IBM WebSphere Message Broker V7.0 System Administration Workshop

(Course code WM643 / VM643)

Instructor Guide

ERC 1.0

Authorized
Training

WebSphere Education

Trademarks

IBM® is a registered trademark of International Business Machines Corporation.

The following are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

AIX®
CICS®
DataPower®
DB2®
developerWorks®
Everyplace®
HACMP™
IMS™
Informix®
Language Environment®
MQSeries®
Notes®
POWER®
Rational®
Redbooks®
System z®
Tivoli®
VisualAge®
WebSphere®
z/OS®
zSeries®

VMware® and the VMware “boxes” logo and design, Virtual SMP and VMotion are registered trademarks or trademarks (the “Marks”) of VMware, Inc. in the United States and/or other jurisdictions.

Adobe is either a registered trademark or a trademark of Adobe Systems Incorporated in the United States, and/or other countries.

Intel, Intel Core, Itanium and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux® is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT and Windows Vista are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX® is a registered trademark of The Open Group in the United States and other countries.

Other product and service names might be trademarks of IBM or other companies.

November 2010 edition

The information contained in this document has not been submitted to any formal IBM test and is distributed on an “as is” basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer’s ability to evaluate and integrate them into the customer’s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will result elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Contents

Trademarks xv

Instructor course overview. xvii

Course description xxiii

Agenda xxvii

Unit 1. WebSphere Message Broker overview 1-1
Unit objectives 1-3
1.1. IBM WebSphere reference architecture, SOA, and ESB 1-5
IBM WebSphere reference architecture 1-6
Service-oriented architecture (SOA) 1-8
Integration techniques 1-10
ESB advantages 1-12
WebSphere ESB products 1-14
WebSphere Message Broker integration with an ESB 1-16
1.2. WebSphere Message Broker connectivity 1-19
Enterprise messaging 1-20
Message brokering 1-23
Publish/subscribe implementation 1-25
WebSphere Message Broker and WebSphere MQ publish/subscribe interoperability 1-27
Web services support 1-29
Adapters, bridges, and clients 1-31
WebSphere MQ interoperability 1-33
Prerequisite WebSphere MQ skills 1-35
1.3. Development and runtime components 1-37
Interaction of WebSphere Message Broker main components 1-38
The WebSphere Message Broker Toolkit 1-40
Broker components 1-42
Broker administration tools 1-44
WebSphere Message Broker Explorer 1-46
Command-line utilities 1-48
Basic commands 1-50
Administration API 1-52
Message flows 1-54
Message sets 1-56
Message flow example 1-58
MQInput node 1-60
WebSphere Message Broker node examples 1-62
WebSphere Information Center 1-64
Unit summary 1-66
Checkpoint (1 of 2) 1-68

Checkpoint answers (1 of 2) 1-70
Checkpoint (2 of 2) 1-72
Checkpoint answers (2 of 2) 1-74

Unit 2. Product installation, configuration, and security planning 2-1
Unit objectives 2-2
2.1. Hardware and software requirements 2-5
WebSphere Message Broker Toolkit system requirements 2-6
WebSphere Message Broker Explorer system requirements 2-8
WebSphere Message Broker platform summary 2-10
WebSphere Message Broker system requirements 2-12
Operation modes 2-14
Supported databases 2-16
Other software requirements 2-18
Some essential background information 2-20
Product package contents 2-22
Optional software and support 2-24
2.2. Installation and configuration 2-27
Before you begin installation 2-28
Installation 2-30
Installation using the Launchpad (Windows only) 2-32
Configuration overview 2-34
Create a broker using mqsicreatebroker 2-36
Actions performed by mqsicreatebroker 2-38
SYSTEM.BROKER queues 2-40
Start and stop components from command console 2-42
Create an execution group using mqsicreateexecutiongroup 2-44
Create a broker using Message Broker Explorer 2-46
Start and stop broker from Message Broker Explorer 2-48
Create an execution group using Message Broker Explorer 2-50
Connect to broker from Message Broker Explorer 2-52
Create components using the Default Configuration wizard 2-54
Verify broker setup using mqsicvp 2-56
Broker properties 2-58
Report component details using mqsiservice 2-60
List installed brokers using mqsilist 2-62
mqsilist command examples 2-64
2.3. Security considerations 2-67
Security considerations overview 2-68
Security features overview 2-70
Security checks 2-72
Security settings scenario 2-74
Unit summary 2-76
Checkpoint (1 of 2) 2-78
Checkpoint answers (1 of 2) 2-80
Checkpoint (2 of 2) 2-82
Checkpoint answers (2 of 2) 2-84
Exercise 1: Broker setup and customization 2-86

Exercise objectives 2-88

Unit 3. Using the WebSphere Message Broker Toolkit 3-1
Unit objectives 3-2
3.1. WebSphere Message Broker Toolkit concepts 3-5
Eclipse platform 3-6
Development and runtime environment 3-8
Workbench 3-10
Broker Application Development perspective 3-12
Workspace 3-14
Customizing the workspace 3-16
Projects 3-18
Project maintenance 3-20
Active working sets 3-22
Workspace import and export 3-24
WebSphere Information Center 3-26
Integrated help 3-28
3.2. Basic configuration tasks 3-31
WebSphere Message Broker Toolkit Welcome view 3-32
Broker Development views 3-34
Brokers view 3-36
Basic configuration 3-38
Importing a resource (1 of 4) 3-40
Importing a resource (2 of 4) 3-42
Importing a resource (3 of 4) 3-44
Importing a resource (4 of 4) 3-46
Create or modify a message flow 3-48
MQInput node properties examples 3-50
MQOutput node properties example 3-52
Compute node and ESQL editor 3-54
Save your work 3-56
3.3. Deploying message flows. 3-59
Deployment overview 3-60
Creating or modifying a BAR file 3-62
Creating the BAR file (1 of 2) 3-64
Creating the BAR file (2 of 2) 3-66
Modifying a BAR file 3-68
Creating or modifying a BAR file 3-70
BAR message flow properties 3-72
BAR node properties 3-75
Compiling and deploying 3-77
Deploying 3-80
Deployment Log 3-82
3.4. Testing message flows. 3-85
Integrated Test Client 3-86
Integrated Test Client configuration 3-88
Integrated Test Client events 3-90
Specifying the Test Client deployment location (1 of 2) 3-92

Specifying the Test Client deployment location (2 of 2) 3-94
Test using the Test Client (1 of 2) 3-96
Test using the Test Client (2 of 2) 3-98
RfhUtil (Support Pac IH03) 3-100
RfhUtil: Main tab 3-102
RfhUtil: Data tab 3-104
Unit summary 3-106
Checkpoint 3-108
Checkpoint answers 3-110
Exercise 2: Using the WebSphere Message Broker Toolkit 3-112
Exercise objectives 3-114

Unit 4. Configuring and administering the broker 4-1
Unit objectives 4-2
4.1. Administering WebSphere Message Broker using the command-line. 4-5
Using administrative commands 4-6
Starting, stopping, and deleting a broker 4-9
Password prompt option 4-11
Modifying a broker 4-13
Execution group commands 4-15
Message flow commands 4-17
Create a BAR file from a command line 4-19
Deploy a BAR file from a command line 4-22
Security commands 4-24
Modify broker properties command 4-26
Report broker properties command 4-28
Problem determination commands 4-30
Creating command line scripts 4-33
Example batch file 4-35
Example Ant script 4-37
4.2. Administering Message Broker using Message Broker Explorer. 4-39
WebSphere Message Broker Explorer 4-40
Graphical administration consistent with MQ 4-43
WebSphere Message Broker Explorer: Navigator 4-45
WebSphere Message Broker status icons 4-47
Grouping brokers using broker sets 4-49
Creating a broker set (1 of 3) 4-51
Creating a broker set (2 of 3) 4-53
Creating a broker set (3 of 3) 4-55
Deployment process 4-57
Multiple deployment methods 4-60
Broker Archive File content 4-62
Broker views 4-64
Broker properties 4-66
Execution group management (1 of 2) 4-68
Execution group management (2 of 2) 4-70
Message flow management options 4-72
Message flow and node properties 4-74

Configurable services 4-76
Configurable Service wizard 4-78
Administration Queue 4-80
WebSphere Message Broker Explorer preferences 4-82
4.3. Administration API (CMP API) overview. 4-85
Administration API overview 4-86
Benefits of the Administration API 4-88
Administration API service trace 4-90
Application API samples 4-92
CMP API Exerciser 4-94
View and manager a broker in the CMP API Exerciser 4-96
4.4. Backing up and restoring resources. 4-99
Backup and restore 4-100
Backup the broker 4-102
Restore the broker 4-104
Backing up and restoring Message Broker Toolkit 4-106
Backing up application databases 4-108
Unit summary 4-110
Checkpoint 4-112
Checkpoint answers 4-114
Exercise 3: Administering the broker runtime components 4-116
Exercise objectives 4-118

Unit 5. Implementing WebSphere Message Broker security. 5-1
Unit objectives 5-2
5.1. Broker component security. 5-5
WebSphere Message Broker security 5-6
Groups and authorities 5-8
Broker security considerations 5-10
Service user account on Windows 5-12
Security exits 5-14
Creating a security exit 5-16
5.2. Broker administration security 5-19
Administration security 5-20
Broker component authorizations 5-22
Actions subject to authority checking 5-24
Activating broker administration authority 5-26
Authorization queues 5-28
Authorization queue management 5-30
Authorization process 5-32
Required authority for connecting to the broker 5-34
Authorization queue limitations 5-36
Setting MQ permissions for Message Broker 5-38
Tasks and authorizations 5-40
Creating authorization permissions (1 of 2) 5-42
Creating authorization permissions (1 of 2) 5-44
Granting and revoking authority from a command line 5-46
setmqaut examples 5-48

Deactivating broker administration authority 5-50
5.3. Message flow security 5-53
Message flow security 5-54
Security profiles 5-56
Security profile property 5-58
Creating a security profile 5-60
Security profile configuration example 5-62
Identity and security token propagation 5-64
Configuring for identity propagation 5-66
Secure sockets layer (SSL) 5-68
Implementing SSL authentication in Message Broker 5-70
Configuring PKI at the broker level (1 of 2) 5-72
Configuring PKI at the broker level (2 of 2) 5-74
Configuring PKI at the execution group level (1 of 2) 5-76
Configuring PKI at the execution group level (2 of 2) 5-78
SSL and JMS nodes configuration 5-80
Unit summary 5-82
Checkpoint 5-84
Checkpoint answers 5-86
Exercise 4 5-88
Exercise objectives 5-90

Unit 6. Monitoring and problem determination 6-1
Unit objectives 6-2
6.1. Broker default behavior in case of runtime errors 6-5
Message flow: Case1 6-6
Message flow: Case 2 6-8
Message flow error behavior: BOTHRESH not exceeded 6-10
Message flow error behavior: BOTHRESH exceeded 6-12
Important notes 6-14
Transaction support 6-16
Globally coordinated transaction 6-19
6.2. Testing and problem determination tools. 6-21
Troubleshooting tools 6-22
Local error log (syslog) 6-24
Flow debugger overview 6-26
Flow debugger 6-28
Configuring the flow debug port 6-30
Enabling the flow debug port 6-32
Configuring flow debug port from a command line 6-34
Launching the flow debugger 6-36
Flow debugger example 6-38
Trace 6-40
User trace 6-42
Trace commands 6-44
Activating a user trace from WebSphere Message Broker Explorer 6-46
Usertrace.cmd utility 6-48
GetTrace.cmd utility 6-50

Service trace 6-52
Activating a service trace from a command line 6-54
Activating a service trace from WebSphere Message Broker Explorer 6-56
Trace node 6-58
Configuring the Trace node 6-60
Comparing test and debugging methods 6-62
Windows test tools 6-65
WebSphere MQ test tools 6-67
Exercise 5 6-69
Exercise objectives 6-71
6.3. Troubleshooting common problems 6-73
Some common problems for administrators 6-74
Making initial checks 6-76
Debugging deployment problems 6-78
Message Broker Toolkit performance issues 6-81
Data flow engine memory usage 6-83
WebSphere MQ or database issues 6-86
WebSphere Message Broker as an WebSphere MQ application 6-88
Message flow hangs or loops 6-90
Searching knowledge bases 6-92
Help system (information center) 6-94
Getting product fixes 6-96
Unit summary 6-98
Checkpoint (1 of 2) 6-100
Checkpoint answers (1 of 2) 6-102
Checkpoint (2 of 2) 6-104
Checkpoint answers (2 of 2) 6-106
Exercise 6: Identifying runtime problems 6-108
Exercise objectives 6-110

Unit 7. Monitoring broker and message flow performance 7-1
Unit objectives 7-3
7.1. Resource and message flow accounting and statistics 7-5
Monitoring performance 7-6
Resource statistics 7-8
Resource statistics resource types 7-10
Viewing resource statistics 7-12
Starting resource statistics collection in Message Broker Explorer 7-14
Viewing resource statistics in Message Broker Explorer 7-16
Filtering resource statistics 7-18
Message flow accounting and statistics 7-20
When is statistics data written? 7-22
Message flow accounting and statistics details 7-24
Message flow operation statistics commands (1 of 2) 7-27
Message flow operation statistics commands (2 of 2) 7-29
Message flow statistics in WebSphere Message Broker Explorer 7-31
Subscribing to message flow statistic reports 7-33
XML publish/subscribe example output 7-35

7.2. Event monitoring 7-37
Event monitoring and auditing (1 of 2) 7-38
Event monitoring and auditing (2 of 2) 7-40
Event monitoring profiles 7-42
Event monitoring profile example 7-44
Event monitoring commands 7-46
mqsichangeflowmonitoring flags 7-48
mqsireportflowmonitoring flags 7-50
7.3. Performance Tuning 7-53
Introduction 7-54
Configuration 7-56
Common causes of performance problems 7-58
WebSphere Message Broker tuning options 7-60
Multiple copies of a message flow 7-62
Scaling message throughput 7-65
Broker tuning 7-68
Broker queue manager tuning 7-70
Database tuning 7-73
Detecting and removing obstructions 7-76
Unit summary 7-79
Checkpoint 7-81
Checkpoint answers (1 of 2) 7-83
Checkpoint answers (2 of 2) 7-85

Unit 8. Publish/subscribe implementation overview 8-1
Unit objectives 8-2
8.1. Using publish/subscribe in message flows 8-5
What is publish and subscribe 8-7
Publisher and subscriber applications 8-9
Topics and filters 8-11
Topic filters: Example 8-13
Typical publish/subscribe applications 8-15
Publish/subscribe message flows and applications 8-17
Register subscription in WebSphere Message Broker (1 of 2) 8-19
Register subscription in Message Broker (2 of 2) 8-21
Publication flows 8-23
WebSphere MQ publish/subscribe exploitation 8-25
Publication node 8-27
No-match and Out terminals 8-29
Content-based filtering 8-31
Content-based filtering configuration 8-33
Setting content-based filtering in WebSphere Message Broker Explorer 8-35
8.2. Managing subscriptions and topics 8-37
Managing subscriptions 8-38
Managing topics with WebSphere Message Broker Explorer 8-40
Managing subscriptions with WebSphere Message Broker Explorer 8-42
RfhUtil PubSub tab (1 of 2) 8-44
RfhUtil PubSub tab (2 of 2) 8-46

RfhUtil pscr tab 8-48
Unit summary 8-50
Checkpoint 8-52
Checkpoint answers 8-54
Exercise 7: Accessing broker statistics 8-56
Exercise objectives 8-58

Unit 9. Administering web services and JMS 9-1
Unit objectives 9-3
9.1. Introduction to web services 9-5
What is a web service? 9-6
Web services context 9-8
SOAP 9-10
Web services description language (WSDL) 9-12
Universal description, discovery and integration (UDDI) 9-14
9.2. Implementing WebSphere Message Broker in a web services environment 9-17
WebSphere Message Broker and web services scenarios 9-18
Conceptual overview 9-20
Message flow as HTTP/HTTPS client 9-22
Message flow as HTTP/HTTPS service provider 9-24
Technical implementation 9-26
HTTPInput and HTTPReply node customization 9-29
HTTPRequest node customization 9-31
9.3. Implementing SSL with HTTP nodes 9-35
HTTPS basics 9-36
Enabling SSL for HTTP nodes (1 of 2) 9-38
Enabling SSL for HTTP nodes (2 of 2) 9-40
HTTPRequest node details for SSL 9-42
HTTPInput node and SSL 9-44
Conditions for SSL activation 9-46
What if it is not working? 9-48
9.4. Capacity considerations for HTTP nodes 9-51
Capacity and the HTTP nodes: client mode 9-52
Capacity and HTTP nodes: server mode 9-54
An alternative to broker internal HTTPListener 9-56
Another alternative to broker internal HTTPListener 9-58
Some capacity planning conclusions 9-60
9.5. Creating policy sets 9-63
SOAP nodes overview 9-64
Policy sets 9-66
Default policy set and bindings: WSS10Default 9-68
Importing a policy set and policy set binding (1 of 2) 9-70
Importing a policy set and policy set binding (2 of 2) 9-72
Exporting a policy set and policy set binding 9-74
Display policy set associated with a binding 9-76
Defining policy sets in Message Broker Explorer 9-78
Accessing policy sets and policy set bindings 9-80
Policy set editor 9-82

Policy set binding editor 9-84
9.6. Preparing to use JMS nodes 9-87
JMS architecture 9-88
Why JMS nodes are useful 9-90
JMS nodes 9-92
JMS feature overview 9-94
JMSInput node 9-97
JMSOutput node 9-99
Other JMS nodes 9-101
JMS provider configurable services 9-103
JMS provider configuration steps (1 of 2) 9-105
JMS provider configuration steps (2 of 2) 9-107
JMS Admin utility 9-109
WebSphere MQ Explorer for JMS 9-111
Unit summary 9-113
Checkpoint 9-115
Checkpoint answers 9-117
Exercise 8 9-119
Exercise objectives 9-121

Unit 10. Extending WebSphere Message Broker 10-1
Unit objectives 10-3
10.1. Database connectivity 10-5
Using databases in message flows 10-6
Configure database access for the broker 10-8
Verifying connectivity and compatibility 10-10
Database error messages 10-12
Node properties for database access 10-14
Transaction support 10-16
Transaction coordinator options 10-18
WebSphere Message Broker ODBC Database Extender (IE02) 10-20
10.2. Configuring for SFTP 10-23
Configuring SFTP file transfer 10-24
FTPserver configurable service properties 10-26
Known host checking 10-28
Steps for implementing SFTP 10-30
FileInput node FTP properties 10-32
SFTP configurable service: command line 10-34
SFTP configurable service: WebSphere Message Broker Explorer 10-36
SFTP connection failure troubleshooting 10-38
10.3. User-defined extensions 10-41
What is a user-defined extension? 10-42
Create a user-defined extension 10-44
Message flow plug-in node project contents 10-46
Distribute user-defined extensions 10-48
Implementing a node or a parser 10-50
More on nodes and parsers 10-52
10.4. WebSphere Message Broker SupportPacs 10-55

Before you start programming 10-56
Finding WebSphere Message Broker product extensions 10-58
WebSphere Message Broker SupportPacs 10-60
SupportPac examples 10-62
WebSphere Message Broker information roadmap 10-64
Unit summary 10-66
Checkpoint 10-68
Checkpoint answers 10-70
Exercise 9 10-72
Exercise objectives 10-74

Unit 11. Course summary 11-1
Unit objectives 11-2
Course learning objectives 11-4
Class evaluation 11-6
To learn more on this subject 11-8
References 11-10
Unit summary 11-12

Appendix A. List of abbreviations A-1

Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM® is a registered trademark of International Business Machines Corporation.

The following are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

AIX®
CICS®
DataPower®
DB2®
developerWorks®
Everyplace®
HACMP™
IMS™
Informix®
Language Environment®
MQSeries®
Notes®
POWER®
Rational®
Redbooks®
System z®
Tivoli®
VisualAge®
WebSphere®
z/OS®
zSeries®

VMware® and the VMware “boxes” logo and design, Virtual SMP and VMotion are registered trademarks or trademarks (the “Marks”) of VMware, Inc. in the United States and/or other jurisdictions.

Adobe is either a registered trademark or a trademark of Adobe Systems Incorporated in the United States, and/or other countries.

Intel, Intel Core, Itanium and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux® is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT and Windows Vista are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX® is a registered trademark of The Open Group in the United States and other countries.

Other product and service names might be trademarks of IBM or other companies.

Instructor course overview

This course exposes students to the major administrative interfaces to the WebSphere Message Broker V7.0.0.1. A background in WebSphere MQ administration is assumed.

The product is complex. Accordingly, not all the administrative interfaces are exposed in the time allotted to the course. It is expected that students will continue working with the product and refine their skills, after completing the class.

In general, the exercises take the students through the essential tasks of configuring the environment and establishing broker connections. After established, the remaining exercises lead the students through the various administrative actions required to ensure that the broker operates in an efficient and productive manner.

Interfaces used include the WebSphere Message Broker Explorer, WebSphere Message Broker Toolkit, and the WebSphere Message Broker commands. Students will learn which interface is appropriate for common tasks.

Course strategy

The course strategy is built around lectures, to introduce and describe the administrative aspects of the product, reinforced with exercises that permit the students to administer the product in a pseudo-realistic work environment.

The early exercises (through Exercise 5) build on one another; the exercises must be completed in the designated order. Being an administration class, message flow development is de-emphasized. Where appropriate, students are directed to import message flows rather than develop them in class.

Summary of changes

WM643, ERC 1.0

This course update contains many changes to the previous version, which was written for WebSphere Message Broker V6.0.

General changes are as follows:

- Course renamed from WM642 to WM643/VM643.
- Updated for WebSphere Message Broker V7.0, including many changes to graphics and text
- Changed order of end of unit slide to be consistent with standards. The new order is: Unit summary, Checkpoint, Checkpoint answers, Exercise title, Exercise objectives
- Updated to current standards for slide order, graphics, and font sizes.
- The “Backup and Restore” unit has been merged into Unit 4 as much of the content is obsolete (domains, broker databases, and configuration manager).
- The “Additional transport protocols” unit has been removed as the protocols covered in this unit, except for JMS, have been deprecated. The JMS transport topic has been added to Unit 9, “Implementing web services and JMS”.
- Added more slides on command-line and WebSphere Message Broker Explorer so that units show administrator how to accomplish a task using either option.
- Changed references to CMP API to Administration API
- Added Course Summary unit
- Updated Checkpoint and Checkpoint answers to standards and included questions that are appropriate for SPVC.

Unit 0: Added new course introduction unit

Unit 1 changes:

- Removed references to domains, configuration manager, and user name server
- Added slide to emphasize reliance on WebSphere MQ.
- Added introduction slides on WebSphere Message Broker Explorer
- Added a slide to list some of the nodes available in WebSphere Message Broker
- Added slide on Information Center to introduce online help as a key product component
- Added a slide introducing message sets
- Added a slide for Administration API overview
- Updated graphics to adhere to WebSphere Education graphics standards.

Unit 2 changes:

- Removed specific build and version information and replaced with references to websites and summary slides.
- Split Configuration Overview slide into two slides: one containing overview information and one with command-line information
- Updated screen captures for WebSphere Message Broker V7
- Removed broker topology slides
- Added slides for configuration using WebSphere Message Broker Explorer
- Added slides for commands: mqsicvp, mqsiservice, mqsilist
- Moved some of the security slides to Unit 5, “Implementing Message Broker security.”
- Removed slides referencing access control lists and domains
- Added a slide for WebSphere Message Broker operation modes.
- Added a slide for overview of SYSTEM.BROKER queues

Unit 3 changes:

- Added slides to emphasize the Information Center and online help.
- Added slide for Brokers view
- Removed slides that referenced domain connection and setup, and registering a broker
- Updated slides for building a `.bar` file for WebSphere Message Broker V7
- Added a slide for the Deployment log
- Added slides for creating and deploying the `.bar` file from a command line
- Added more slides for the Integrated Test Client
- Added a slide introducing RfhUtil

Unit 4 changes:

- Removed slides referencing user name server, configuration manager, and broker database
- Replaced “Miscellaneous commands” slide with slides that provide more information and example for the commands.
- Added slide for creating command-line scripts and batch files
- Added slides that show example command-line scripts and Ant scripts
- Added a topic on “Administering Message Broker using Message Broker Explorer”
- Removed slides referencing the configuration manager.
- Added slide for the CMP API Exerciser as this tool is used in later exercise.
- Moved information about backup and restore from into this unit.
- Add slide for backing up databases
- Added a slide for Administration queue

Unit 5: New unit covering broker component security, message flow security, security profiles, SSL, and broker administration authority

Unit 6 changes:

- Updated “Message flow error behavior” slide with new slides adhering to graphics standards
- Update “Message flow debugger” slide screen capture and instructions for WebSphere Message Broker V7 changes.
- Added a slide for configuring the debug port from a command line
- Added slides on activating a user trace and service trace from a command line and from WebSphere Message Broker Explorer.
- Removed slides for Integrated Test Client: some of this is already covered in Unit 3 and some of it is not relevant to an administrator as they do not typically have the message flow knowledge to use the message flow debugger.
- Added slide “Making Initial checks”
- Added slide “Debugging deployment problems”
- Added slide “Searching knowledge bases”
- Added slide “Help system”
- Added slide “Getting product fixes”

Unit 7 changes:

- Recorded content to focus on resource statistics and message flow statistics
- Added content on activating and viewing resource and message flow statistics and graphs in WebSphere Message Broker Explorer
- Added slides for “Message flow statistics commands”
- Added topic for Event Monitoring configuration

Unit 8 changes:

- Updated publish/subscribe slides for new architecture that is more reliant on WebSphere MQ
- Added slides for managing topics and subscriptions using WebSphere MQ Explorer
- Added two slides on RfhUtil tabs for publish/subscribe support.
- Removed topic “Access Control on topic level”
- Removed topic “Configuring multi-broker publish/subscribe topologies”
- Removed topic “Differences and migration from WebSphere MQ publish/subscribe”

Unit 9 (WM642 Unit 10) changes:

- Added new topic for “Creating policy sets”
- Added JMS transport topic from “Message transport protocols” unit in WM642

Unit 10 (WM642 Unit 11) changes:

- Added topic “Database connectivity”
- Added topic “Configuring for SFTP” topic
- Added topic “WebSphere Message Broker SupportPacs”
- Removed “Complex event processing” from this unit. Superseded by the event processing topic in Unit 7.

Unit 11: New summary unit

Course description

IBM WebSphere Message Broker V7.0 System Administration Workshop

Duration: 4 days

Purpose

This 4-day instructor-led workshop provides an intermediate-level overview of the IBM WebSphere Message Broker product, concentrating on administering WebSphere Message Broker V7 on distributed platforms such as Windows and AIX. The course is designed specifically for students who will administer the Message Broker environment, including Message Broker developers who also work in an administrative capacity.

This course teaches students how to use common WebSphere Message Broker administrative interfaces, including the WebSphere Message Broker Toolkit, WebSphere Message Broker Explorer, and the command-line interface. Topics include initial installation and subsequent maintenance tasks, environment configuration, and routine administration tasks such as backup and recovery of the environment. Interactive discussions cover topics such as product requirements planning, problem determination and resolution, performance monitoring and tuning, and techniques for extending the capabilities of WebSphere Message Broker. This course also covers the publish/subscribe model and provides students with a review of the Java Message Service (JMS) transport protocol and web services.

In the hands-on lab exercises, students learn how to perform essential configuration and administrative tasks for the WebSphere Message Broker environment, such as:

- Configuring the installed WebSphere Message Broker environment by using the WebSphere Message Broker Toolkit, WebSphere Message Broker Explorer, and command-line interfaces
- Creating and configuring a broker
- Ensuring the efficient operation of a broker
- Backing up and restoring the WebSphere Message Broker environment
- Configuring security parameters
- Using tracing and debugging tools for problem determination and resolution

The exercises also provide hands-on experience with the publish/subscribe message topology, administering WebSphere Message Broker web services and security, and extending the base functionality of WebSphere Message Broker by implementing user-defined extensions and SupportPacs.

Audience

This course is designed for WebSphere Message Broker administrators, including WebSphere Message Broker developers who act in an administrative capacity in their environments.

Prerequisites

Before taking this course, students should successfully complete one of the IBM WebSphere MQ V7 System Administration courses (WM201, VM201, WM202, or VM202), and have hands-on experience with the Windows XP operating system environment.

Objectives

After completing this course, you should be able to:

- Install and configure a WebSphere Message Broker instance
- Establish, maintain, and manage a broker
- Perform basic administration tasks in the management of a broker
- Use the command-line interface and Message Broker Explorer for dedicated administration
- Use problem determination aids to diagnose and solve development and runtime errors
- Implement WebSphere Message Broker security

Contents

Instructional units in this class include, but are not limited to: using the WebSphere Message Broker Toolkit, web services, and collecting broker accounting and statistical data.

Curriculum relationship

Courses providing prerequisite knowledge:

- WM201, *IBM WebSphere MQ V7 System Administration (Windows)*

- WM202, *IBM WebSphere MQ V7 System Administration (AIX Labs)*

Agenda

Day 1

(00:30) Course introduction
(01:00) Unit 1: WebSphere Message Broker overview
(01:00) Unit 2: Product installation, configuration, and security planning
(01:00) Exercise 1: Broker setup and customization
(01:00) Unit 3: Using the WebSphere Message Broker Toolkit
(01:00) Exercise 2: Using the WebSphere Message Broker Toolkit

Day 2

(01:30) Unit 4: Configuring and administering the broker
(01:30) Exercise 3: Administering the broker runtime components
(01:30) Unit 5: Implementing WebSphere Message Broker security
(01:00) Exercise 4: Administering broker security

Day 3

(01:30) Unit 6: Monitoring and problem determination
(00:30) Exercise 5: Using trace facilities
(01:00) Exercise 6: Identifying runtime problems
(01:00) Unit 7: Monitoring broker and message flow performance
(01:00) Unit 8: Publish/subscribe implementation overview

Day 4

(01:00) Exercise 7: Accessing broker statistics
(01:30) Unit 9: Administering web services and JMS
(01:00) Exercise 8: Implementing web services and web services security
(01:00) Unit 10: Extending WebSphere Message Broker
(00:30) Exercise 9: Implementing a user-defined extension
(00:30) Unit 11: Course summary

Unit 1. WebSphere Message Broker overview

Estimated time

01:00 lecture

What this unit is about

This unit introduces the IBM WebSphere reference architecture. You will learn about the position of the WebSphere Message Broker in a service-oriented architecture (SOA) and an enterprise service bus. Finally, this unit introduces WebSphere Message Broker application connectivity and provides an overview of the product components.

What you should be able to do

After completing this unit, you should be able to:

- Describe the major functions of an enterprise service bus (ESB)
- Position WebSphere Message Broker within the IBM WebSphere reference architecture
- Describe the main functions and components of WebSphere Message Broker

How you will check your progress

Accountability:

- Review checkpoint questions

References

www.ibm.com/software/integration/wbimessagebroker
IBM WebSphere Message Broker home page

www.ibm.com/software/integration/wbimessagebroker/library
IBM WebSphere Message Broker Library

www.redbooks.ibm.com

IBM Redbooks

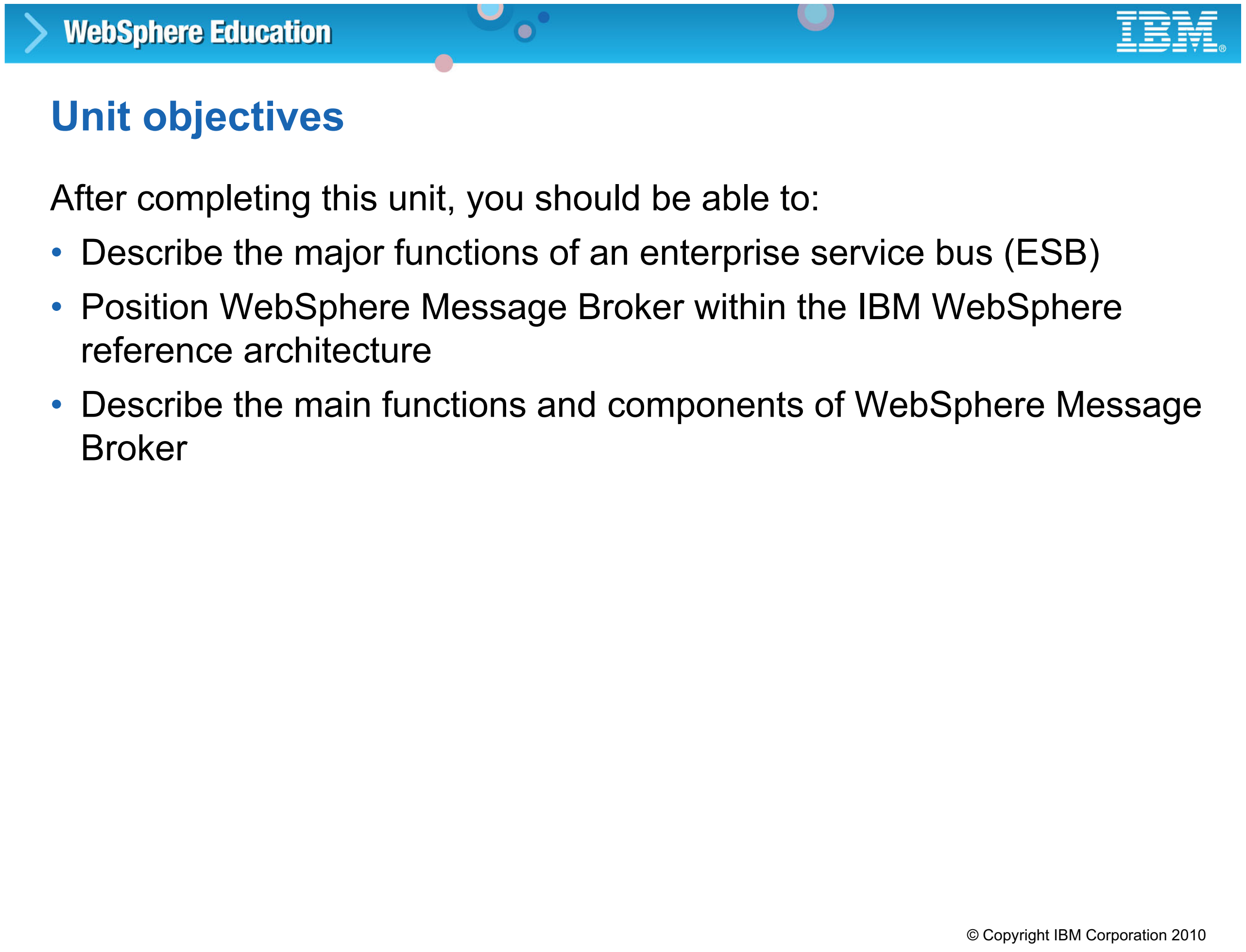
includes:

Enabling SOA Using WebSphere Messaging, SG24-7163-00

Patterns: Integrating Enterprise Service Buses in a Service-Oriented Architecture, SG24-6773-00

Patterns: SOA Design Using WebSphere Message Broker and WebSphere ESB, SG24-7369-00

WebSphere Message Broker Basics, SG24-7137-00



WebSphere Education

Unit objectives

After completing this unit, you should be able to:

- Describe the major functions of an enterprise service bus (ESB)
- Position WebSphere Message Broker within the IBM WebSphere reference architecture
- Describe the main functions and components of WebSphere Message Broker

Figure 1-1. Unit objectives

WM643 / VM6431.0

Notes:

Instructor notes:

Purpose — List unit objectives.

Details —

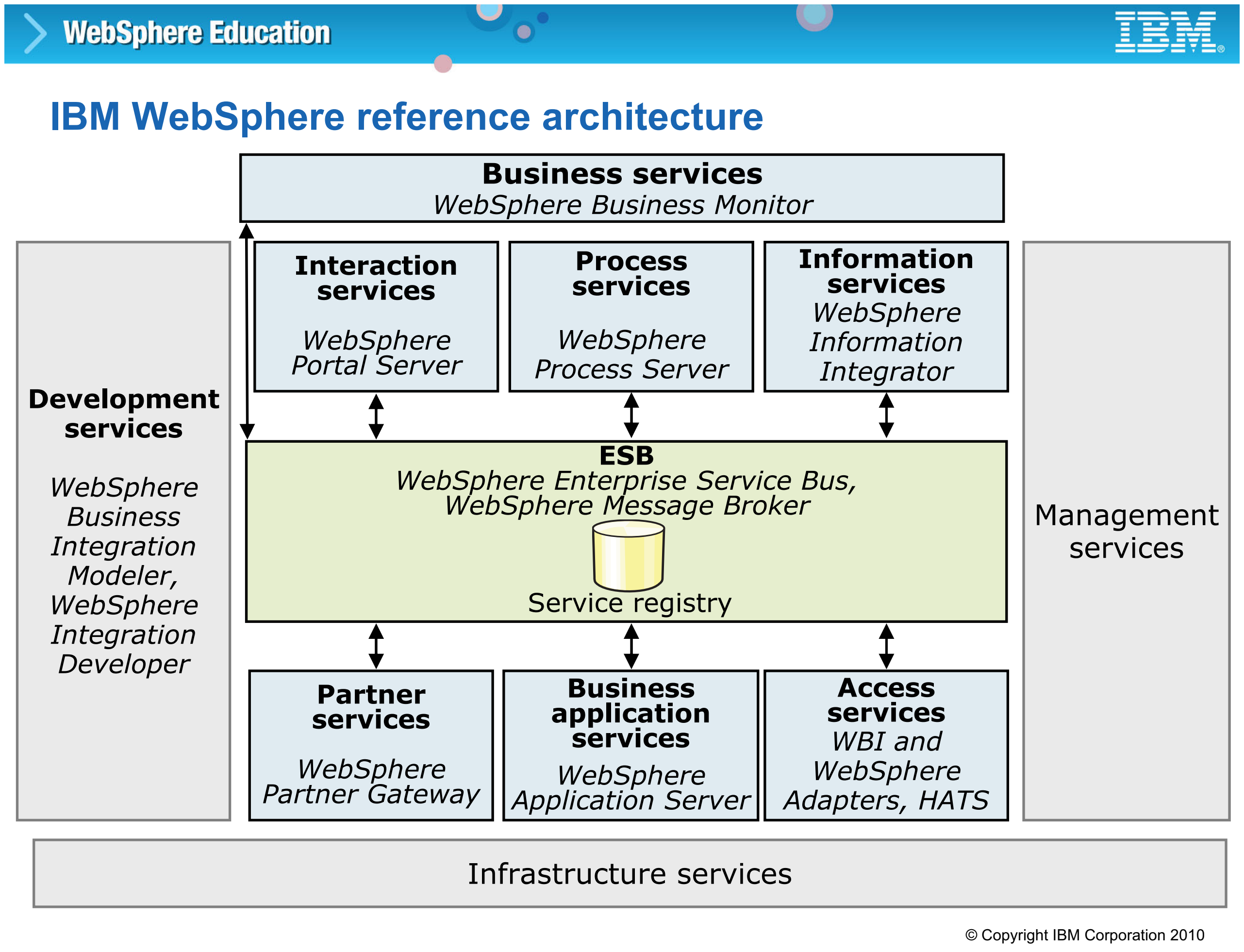
Additional information —

Transition statement — Next: IBM WebSphere reference architecture

1.1. IBM WebSphere reference architecture, SOA, and ESB

Instructor topic introduction

This topic provides high-level overview of the IBM WebSphere products. Do not spend much time on it. The topic is provided to show students where WebSphere Message Broker is positioned within the IBM WebSphere product portfolio.



WebSphere Education

IBM WebSphere reference architecture

Business services
WebSphere Business Monitor

Development services
WebSphere Business Integration Modeler, WebSphere Integration Developer

Interaction services
WebSphere Portal Server

Process services
WebSphere Process Server

Information services
WebSphere Information Integrator

ESB
WebSphere Enterprise Service Bus, WebSphere Message Broker

Service registry

Management services

Partner services
WebSphere Partner Gateway

Business application services
WebSphere Application Server

Access services
WBI and WebSphere Adapters, HATS

Infrastructure services

Figure 1-2. IBM WebSphere reference architecture WM643 / VM6431.0

Notes:

The WebSphere platform provides the software foundation for on-demand business. WebSphere started as a transactional application server and has grown with customer requirements to provide software products that deliver a breadth of business integration. While it emphasizes the value of the entire platform, clients can start easily by using just the functions required at the time, and then to progressively add functions as needed. The IBM middleware platform is implemented as a service-oriented architecture (SOA), which IBM calls the IBM WebSphere Integration Reference Architecture.

The architecture diagram shows the relationship between each of the products in the WebSphere suite and where they are positioned in the reference architecture.

Instructor notes:

Purpose — Describe the WebSphere integration architecture. Indicate how the layers are complementary. Different solutions might use subsets of the architecture functions but the most value, and what is required for e-business on demand, is obtained when all the function of the architecture is used.

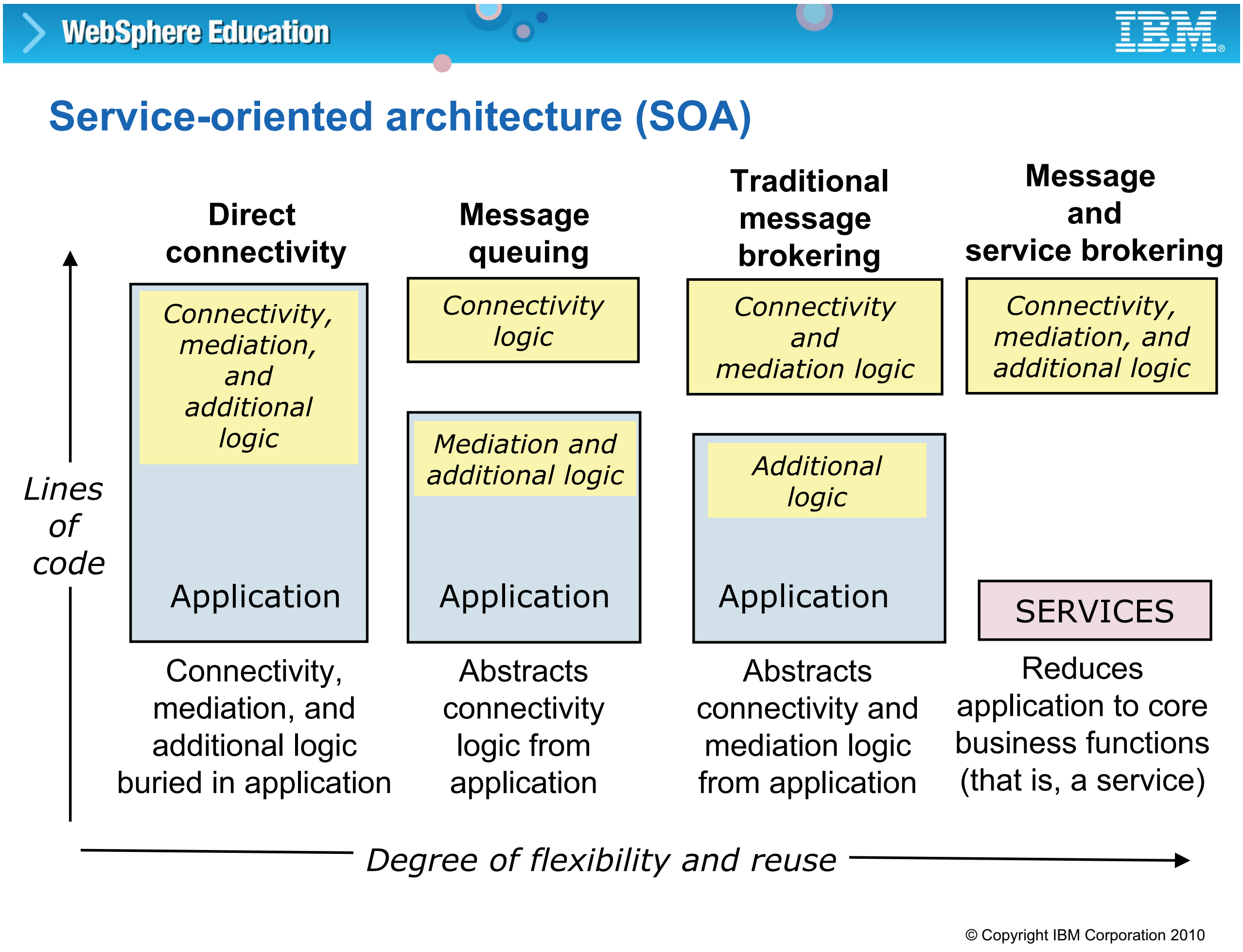
Details — Start at the bottom of the chart and work your way up the stack.

Integration solutions usually begin with the fundamental need to interconnect multiple applications. Integration solutions include prepackaged applications, such as SAP and PeopleSoft, and often include applications that were built years ago with no requirement or design considerations for interconnecting and interacting with other applications or components in a distributed system. There are three basic layers within the architecture:

- Application connectivity provides integration middleware. Information can flow between the applications in a way that abstracts the details of the information flow from the applications themselves. This layer provides:
  - Connectivity management attaches applications to a transport, isolating the details of the connection from the internals of the application. Connectivity management is provided through adapters.
  - Information delivery management determines the appropriate destination for the information flow and ensure that it is in the form required by the destination.
- Process integration provides integration middleware that manages the flow of execution of function across various heterogeneous, connected applications in a way that abstracts the details of the flow of activity from the applications themselves. This layer provides functions that:
  - Catch and handle application events that must be propagated to other applications
  - Control the flow of actions required among the interconnected applications
  - Allow the interaction of people and applications
  - Provide control and state management required for long-running processes
- Modeling and monitoring can be used to graphically create runtime assets that implement a business process, and view, collect, analyze, and use data from the runtime system to upgrade the processes, effectively providing the necessary tools for continuous process improvement. This layer provides the functions required for:
  - Efficient implementation of business processes
  - Analyzing and assessing process efficiency and effectiveness
  - Continuous business improvements through process management and change

Additional information — None.

Transition statement — SOA helps to make interfaces simpler and easier to maintain



WebSphere Education

Service-oriented architecture (SOA)

Direct connectivity

Connectivity, mediation, and additional logic

Application

Connectivity, mediation, and additional logic buried in application

Message queuing

Connectivity logic

Mediation and additional logic

Application

Abstracts connectivity logic from application

Traditional message brokering

Connectivity and mediation logic

Additional logic

Application

Abstracts connectivity and mediation logic from application

Message and service brokering

Connectivity, mediation, and additional logic

SERVICES

Reduces application to core business functions (that is, a service)

Lines of code

Degree of flexibility and reuse

Figure 1-3. Service-oriented architecture (SOA) WM643 / VM6431.0

Notes:

The objective of SOA is to reduce the service down to the bare business logic. The diagram in this figure illustrates that message-queuing by itself does not help to deliver information in the correct format. Nor does it help direct information to different targets based on message content. Interface logic is required to communicate directly with application interfaces.

A broker eliminates the second level of interface logic by removing transformation and routing logic from the application interfaces. A broker can also augment content and reroute messages based on message content, and translate between different protocols and programming models.

Additional standards, such as web services description language (WSDL), are required to further package the application as a service.

Instructor notes:

Purpose — Introduce SOA.

Details — WebSphere MQ decouples applications by abstracting connections through the use of queues. Instead of the application communicating directly with another application, it communicates with an intermediate queue which can then be directed to any application. Message queuing also eliminates the need for the application to deal with the intricacies of different platforms or to figure out whether the other application is busy or even online. It also ensures message delivery without duplication. All of this capability eliminates interface code from your application.

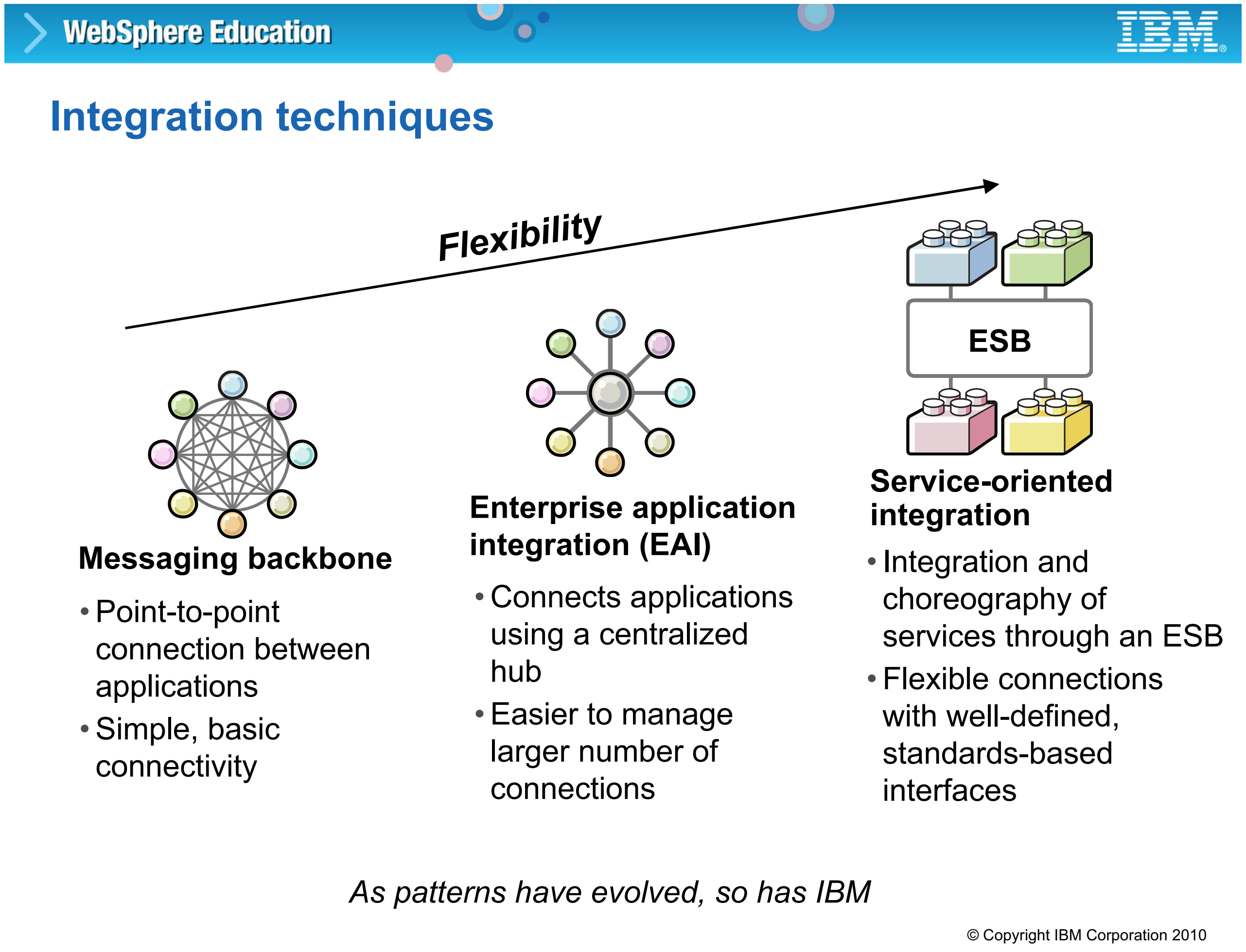
But message queuing itself does nothing to help deliver information in the right format. Nor does it help you direct information to different target destinations, the determination of which might be based on message content. You still must build all of this interface logic directly into your application.

To eliminate this second level of interface logic, you need a different type of intermediary such as a broker. A broker removes the transformation and routing logic from the application interfaces. A broker can also augment content and reroute messages based on content. They can also translate between different protocols and programming models.

Nevertheless, even with a “traditional” broker, some logic might remain in your applications. To reduce the logic to a *service,* some additional standards are needed. First is the use of web services description language (WSDL). Second is the programming model for mediating between different programming models, such as queue names (used with queuing), topics (used with publish/subscribe), and services (used by web services). You might also perform other functions for the application, such as the automation of request/reply calls. Finally, locating and binding to existing services, or to the various transformations that might be required, is another aspect that you might not want to handle from within your application program.

Additional information — None.

Transition statement — The next slide shows some of the integration techniques used in the recent past.



WebSphere Education

Integration techniques

Messaging backbone

- Point-to-point connection between applications
- Simple, basic connectivity

Enterprise application integration (EAI)

- Connects applications using a centralized hub
- Easier to manage larger number of connections

Service-oriented integration

- Integration and choreography of services through an ESB
- Flexible connections with well-defined, standards-based interfaces

As patterns have evolved, so has IBM

Figure 1-4. Integration techniques WM643 / VM6431.0

Notes:

The desire to make information technology (IT) more flexible is not new. Each of the integration techniques that are shown in the figure, has its place and is appropriate for handling certain situations.

SOA blends the best of all these concepts into one new architecture. But it is important to recognize that SOA is not the end of the road either. It is the next step in the evolution of flexible infrastructures.

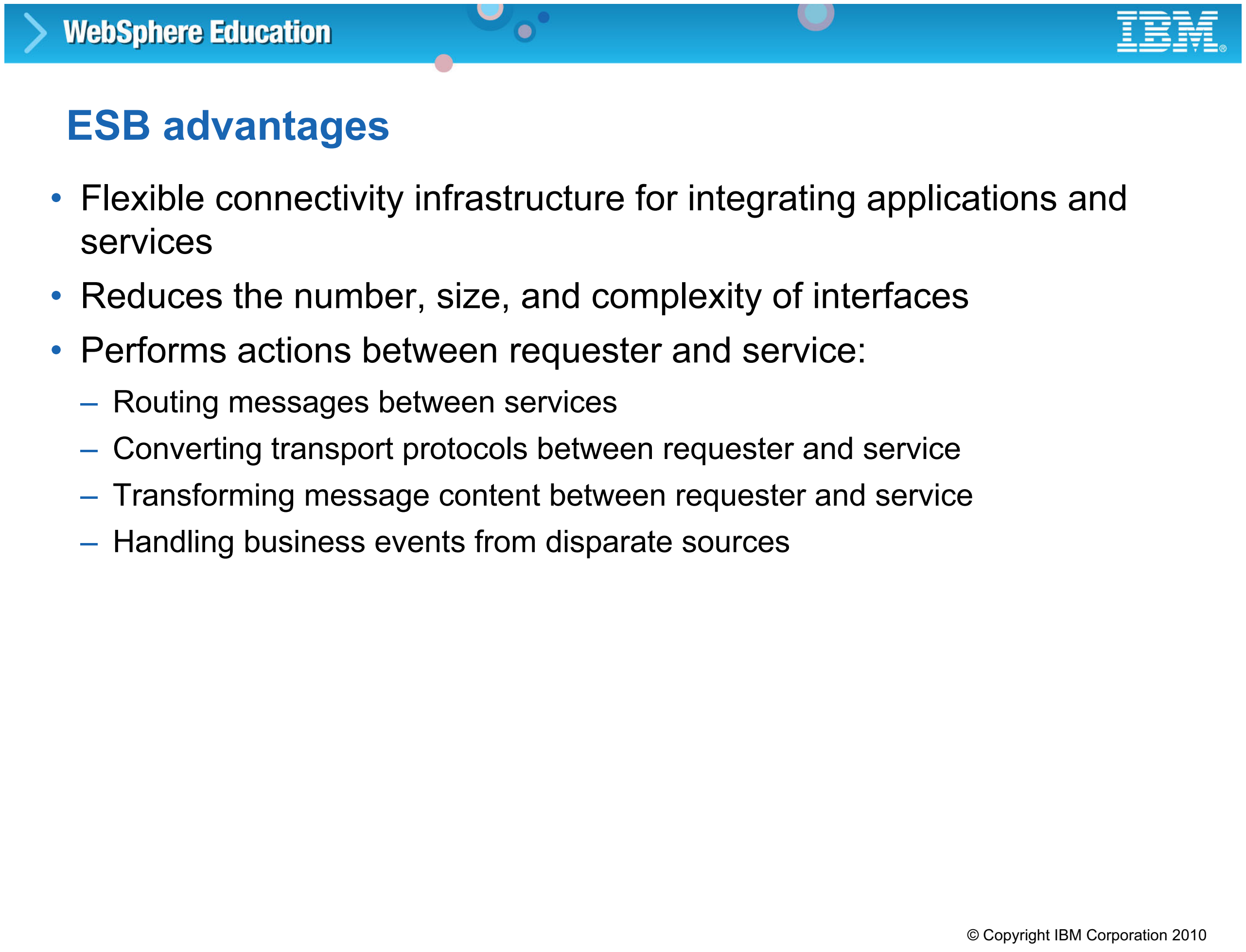
Instructor notes:

Purpose — Show the evolution of integration techniques.

Details — None.

Additional information — None.

Transition statement — So what is an ESB?



WebSphere Education

ESB advantages

- Flexible connectivity infrastructure for integrating applications and services
- Reduces the number, size, and complexity of interfaces
- Performs actions between requester and service:
  - Routing messages between services
  - Converting transport protocols between requester and service
  - Transforming message content between requester and service
  - Handling business events from disparate sources

Figure 1-5. ESB advantages

WM643 / VM6431.0

Notes:

An ESB is a flexible connectivity infrastructure for integrating applications and services. An ESB can help you achieve an SOA by reducing the number, size, and complexity of interfaces.

The ESB provides four main functions:

1. It routes messages between services.
2. It converts transport protocols between requester and service.
3. It transforms message formats between requester and service.
4. It handles business events from disparate sources.

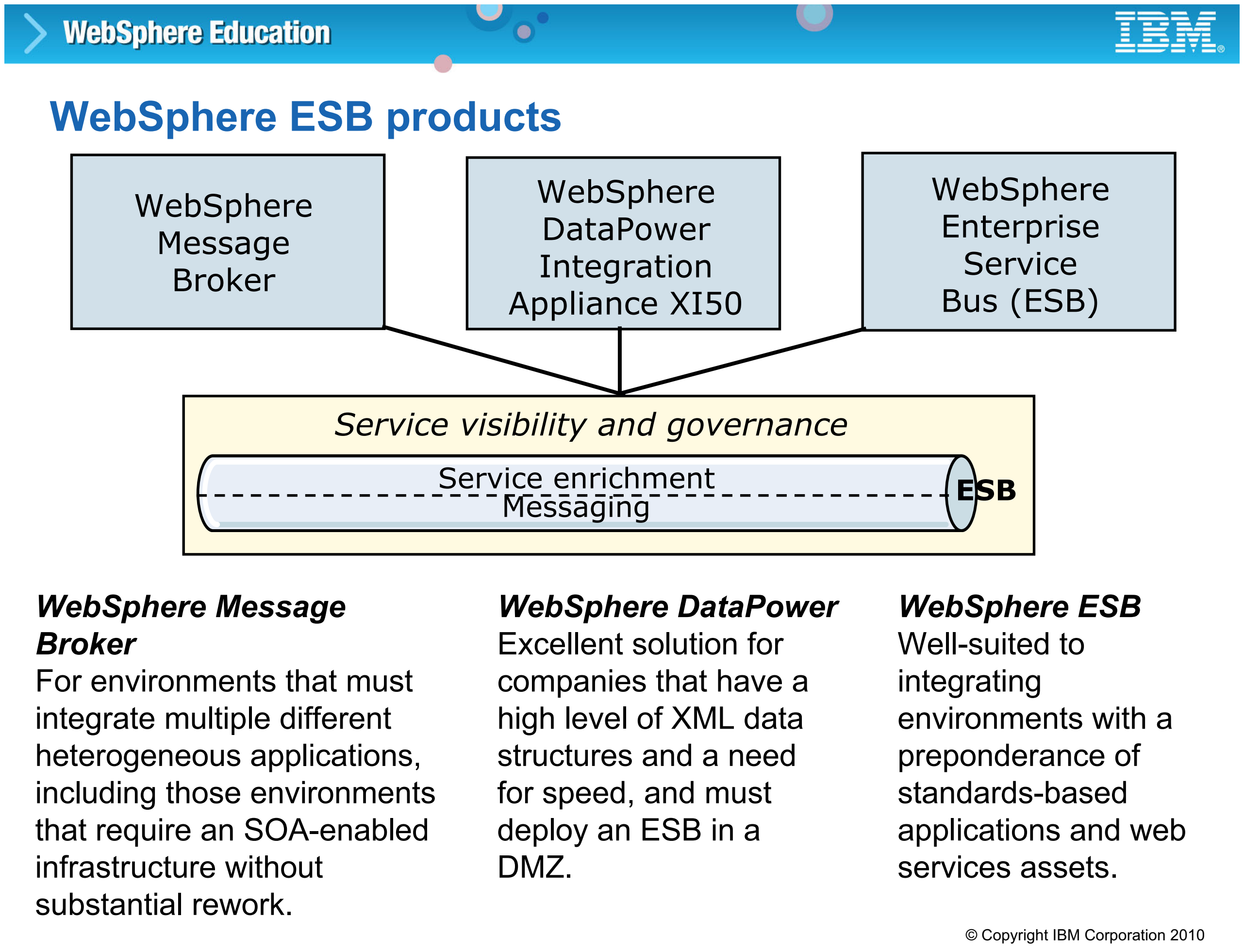
Instructor notes:

Purpose — Define an ESB.

Details — These criteria are central to every IBM ESB product. A product must support all four of the actions listed on the figure to be considered as an ESB product.

Additional information — None.

Transition statement — An ESB is an architecture and it can be implemented using various WebSphere products.



WebSphere Education

WebSphere ESB products

WebSphere Message Broker
WebSphere DataPower Integration Appliance XI50
WebSphere Enterprise Service Bus (ESB)
Service visibility and governance
Service enrichment
Messaging
ESB

WebSphere Message Broker
For environments that must integrate multiple different heterogeneous applications, including those environments that require an SOA-enabled infrastructure without substantial rework.

WebSphere DataPower
Excellent solution for companies that have a high level of XML data structures and a need for speed, and must deploy an ESB in a DMZ.

WebSphere ESB
Well-suited to integrating environments with a preponderance of standards-based applications and web services assets.

Figure 1-6. WebSphere ESB products

WM643 / VM6431.0

Notes:

IBM offers three ESB products, each tailored to suit various deployment needs.

- WebSphere Enterprise Service Bus provides flexible connectivity and integration for web services-focused applications and services.
- WebSphere DataPower Integration Appliance XI50 is purpose-built for simplified deployment and hardened security. You can enable extreme reliability by securing services at the network layer.
- WebSphere Message Broker delivers an advanced ESB, which provides universal connectivity and data transformation for applications, whether they comply to standards.

The WebSphere ESB products can be deployed separately to meet individual requirements, or in any combination to meet sets of different business connectivity requirements that span an entire infrastructure. Each product has its strengths for specific integration project types, and deployment environments.

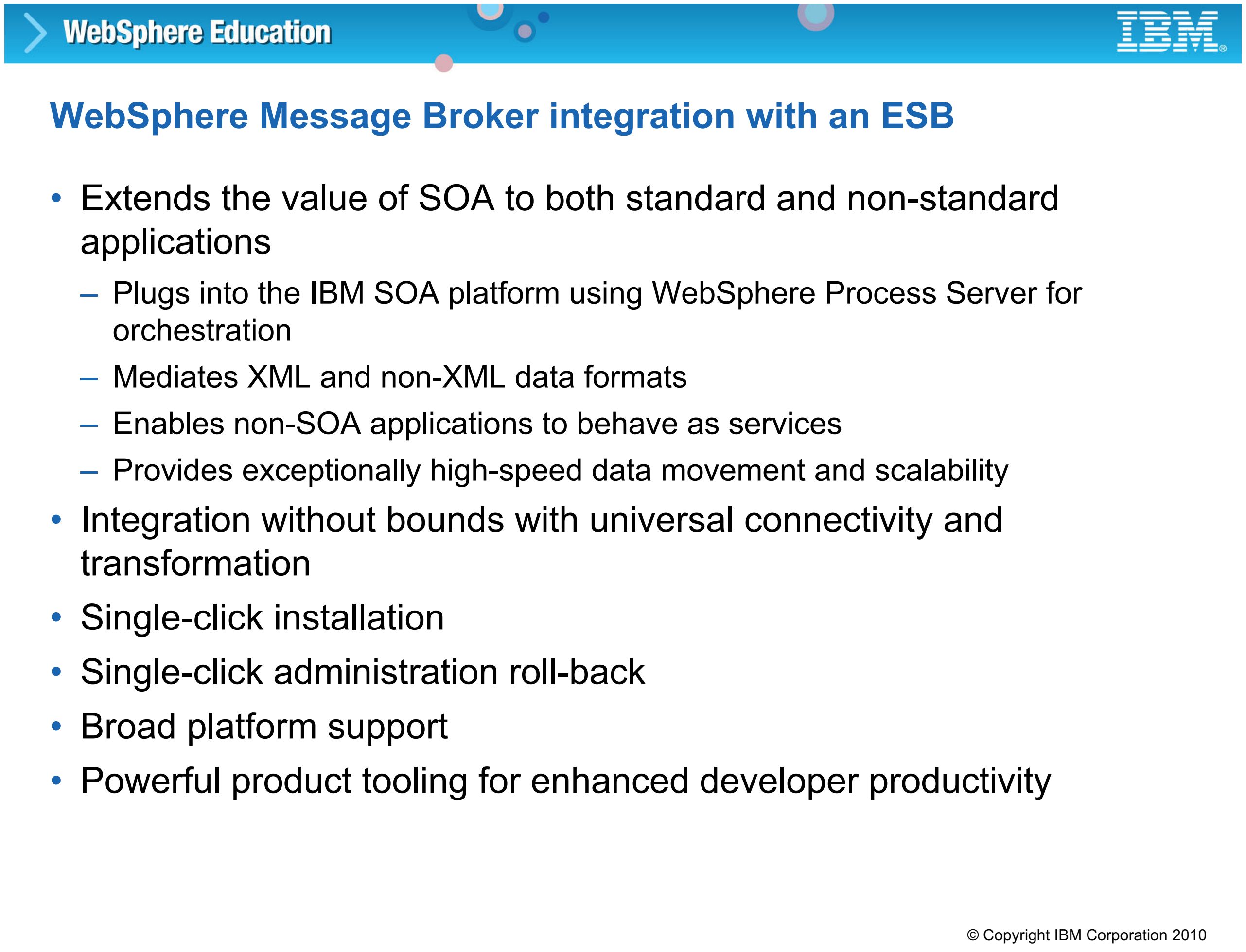
Instructor notes:

Purpose — Position the IBM ESB products.

Details — None.

Additional information — None.

Transition statement — This course focuses on WebSphere Message Broker.



WebSphere Education
IBM

WebSphere Message Broker integration with an ESB

- Extends the value of SOA to both standard and non-standard applications
  - Plugs into the IBM SOA platform using WebSphere Process Server for orchestration
  - Mediates XML and non-XML data formats
  - Enables non-SOA applications to behave as services
  - Provides exceptionally high-speed data movement and scalability
- Integration without bounds with universal connectivity and transformation
- Single-click installation
- Single-click administration roll-back
- Broad platform support
- Powerful product tooling for enhanced developer productivity

Figure 1-7. WebSphere Message Broker integration with an ESB WM643 / VM6431.0

Notes:

Enhancements to WebSphere Message Broker have enabled it to act as a part of a wider ESB.

WebSphere Message Broker can integrate virtually any system. It is fully customizable, but it also offers many prebuilt integration components. It also offers a comprehensive integration platform that supports universal integration between any application and any system because it can send and receive messages using the built-in WebSphere MQ transport layer, or it can read and write from the file system or databases. WebSphere Message Broker can even send and receive e-mail messages and take advantage of other methods of connectivity to provide the most comprehensive integration platform possible.

Recent extensions can locate web services through registries such as IBM WebSphere Service Registry and Repository, and build new web services front-end interfaces to existing applications.

Instructor notes:

Purpose — Describe WebSphere Message Broker integration with an ESB.

Details — This figure provides an overview of some of the integration and ease-of-use features.

Additional information — None

Transition statement — None.

1.2. WebSphere Message Broker connectivity

Instructor topic introduction

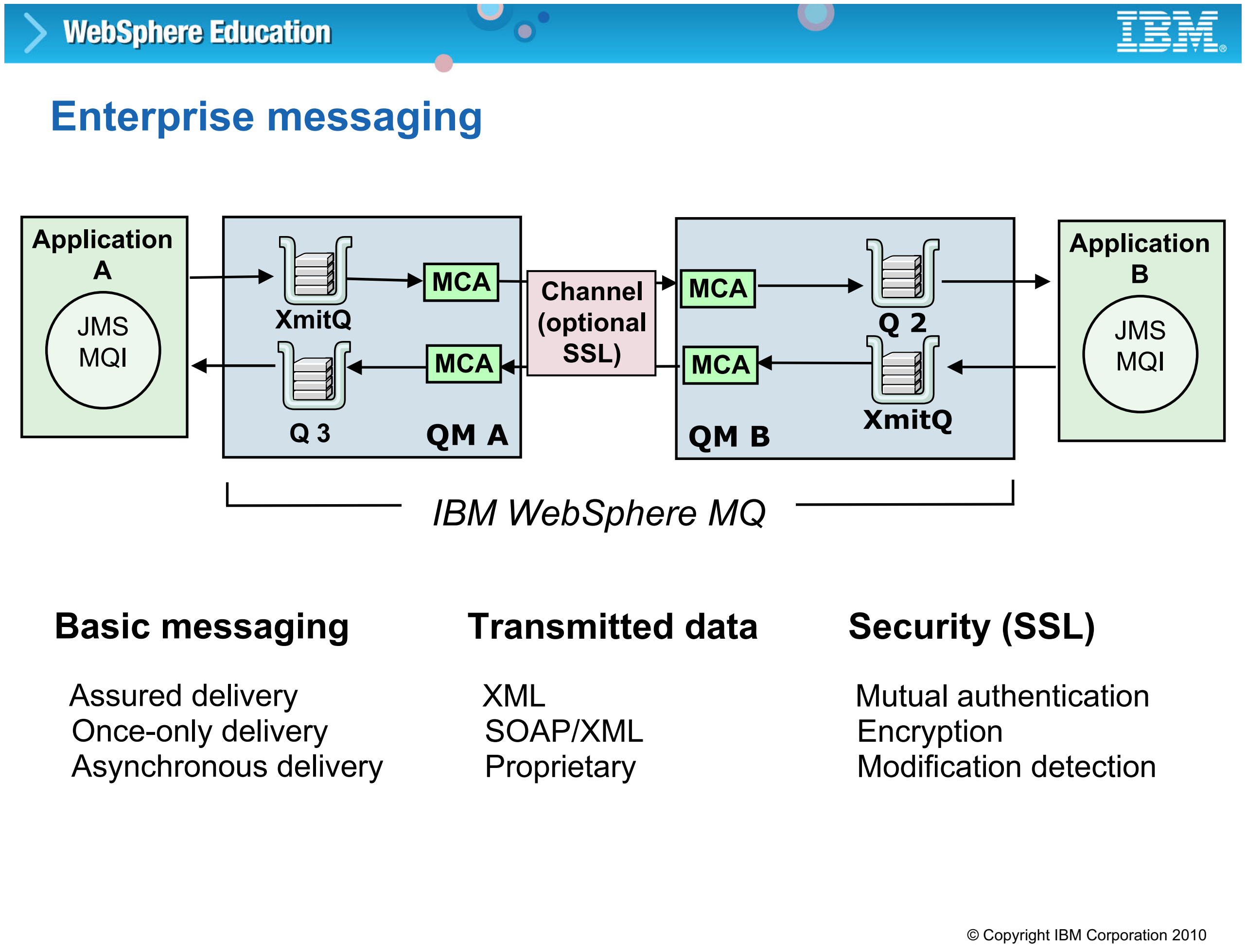
This topic provides an overview on different types of application connectivity.

If students are not familiar with WebSphere MQ concepts (which is a course prerequisite), take some time to describe them now. It might be a good idea to refer students to the WebSphere MQ product demonstration (30 minutes) or WebSphere MQ Primer for self study.

WebSphere MQ product demonstration:
`http://www.ibm.com/software/integration/wmq/quicktour/`

WebSphere MQ Primer:

`http://www.redbooks.ibm.com/abstracts/redp0021.html?Open`
`for self-study.`



Enterprise messaging

Basic messaging

Assured delivery
Once-only delivery
Asynchronous delivery

Transmitted data

XML
SOAP/XML
Proprietary

Security (SSL)

Mutual authentication
Encryption
Modification detection

Figure 1-8. Enterprise messaging WM643 / VM6431.0

Notes:

Enterprise messaging is a general term that describes the orchestrated exchange of business data between disparate applications in complex environments. The main features of enterprise messaging are that it provides application decoupling, hides operating system specifics, and relieves applications from dealing with communications protocols. The common building block of data exchange is the message, which can include records, files, events, web services, requests, and responses.

As shown in the figure, enterprise messaging, data transmission, and security functions are provided by WebSphere MQ. WebSphere MQ, offers assured, once-only delivery of messages between applications and systems. It provides the basic transport on which information flows, asynchronous delivery, and a transport mechanism that can deliver any type of data such as XML, SOAP, and proprietary data, as the situation requires. As more applications participate in the SOA environment, the need to exchange data and messages also arises, requiring a more robust, reliable, timely, and secure way for exchanging messages. The messaging backbone is the foundation for SOA connectivity.

The enterprise messaging, data transmission, and security functions provided by WebSphere MQ are sufficient if the number of applications being interconnected is small. As the number of applications being interconnected increases, additional application connectivity services are required to implement a more efficient and effective solution.

Instructor notes:

Purpose — Review the role of WebSphere MQ.

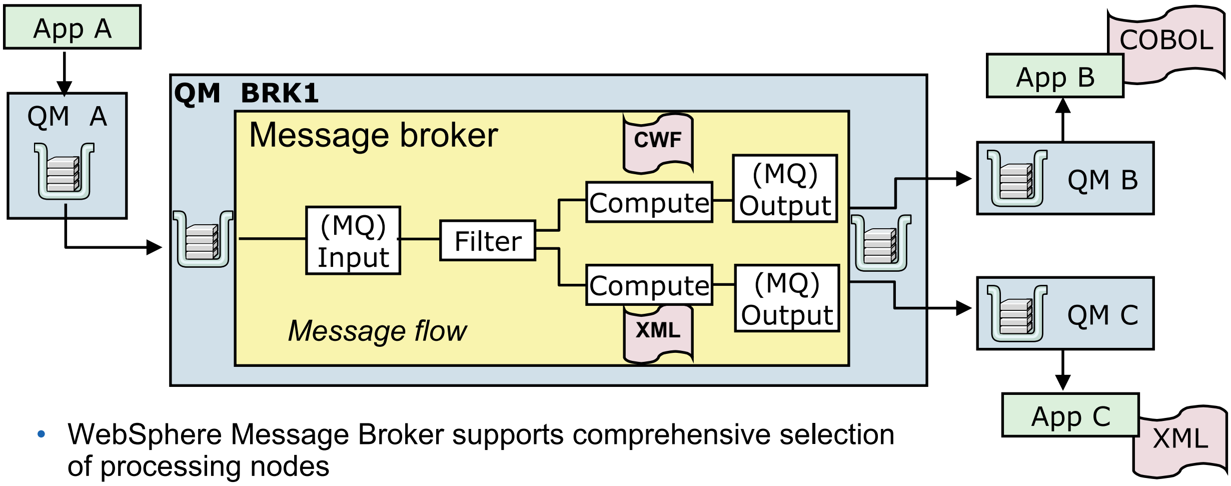
Details — Ideally, students should already be familiar with WebSphere MQ and the basic connectivity.

Additional information — In fact, WebSphere MQ has 75 - 80% market share among customers who buy messaging software.

Transition statement — Discuss some of the services needed for implementing integration across a large (large being greater than four) number of applications.

WebSphere Education

Message brokering



- WebSphere Message Broker supports comprehensive selection of processing nodes
- When connected, nodes create a message flow
  - Receive and route messages
  - Transform a message to an alternative representation
  - Select a message for further processing based upon the message content
  - Interact with an external database to augment a message or store the whole or part of a message
  - Respond to events and errors

Figure 1-9. Message brokering WM643 / VM6431.0

Notes:

When working in a large environment, message brokering functions are required. Fundamentally, message brokers are used to centralize routing and transformation functions, and these functions are handled by a message flow.

A shown in the figure, a message flow is a sequence of operations performed on a message by a series of message processing nodes. The actions are defined in terms of the message format, its content, and the results of individual actions along the message flow. Notice, however, that the direction of the flow is an extension of the path from the source application to the destination application. In other words, it is assumed that the destination wants, expects, and handles all messages sent to it.

Basically, WebSphere Message Broker is a WebSphere MQ application that routes and transforms messages.

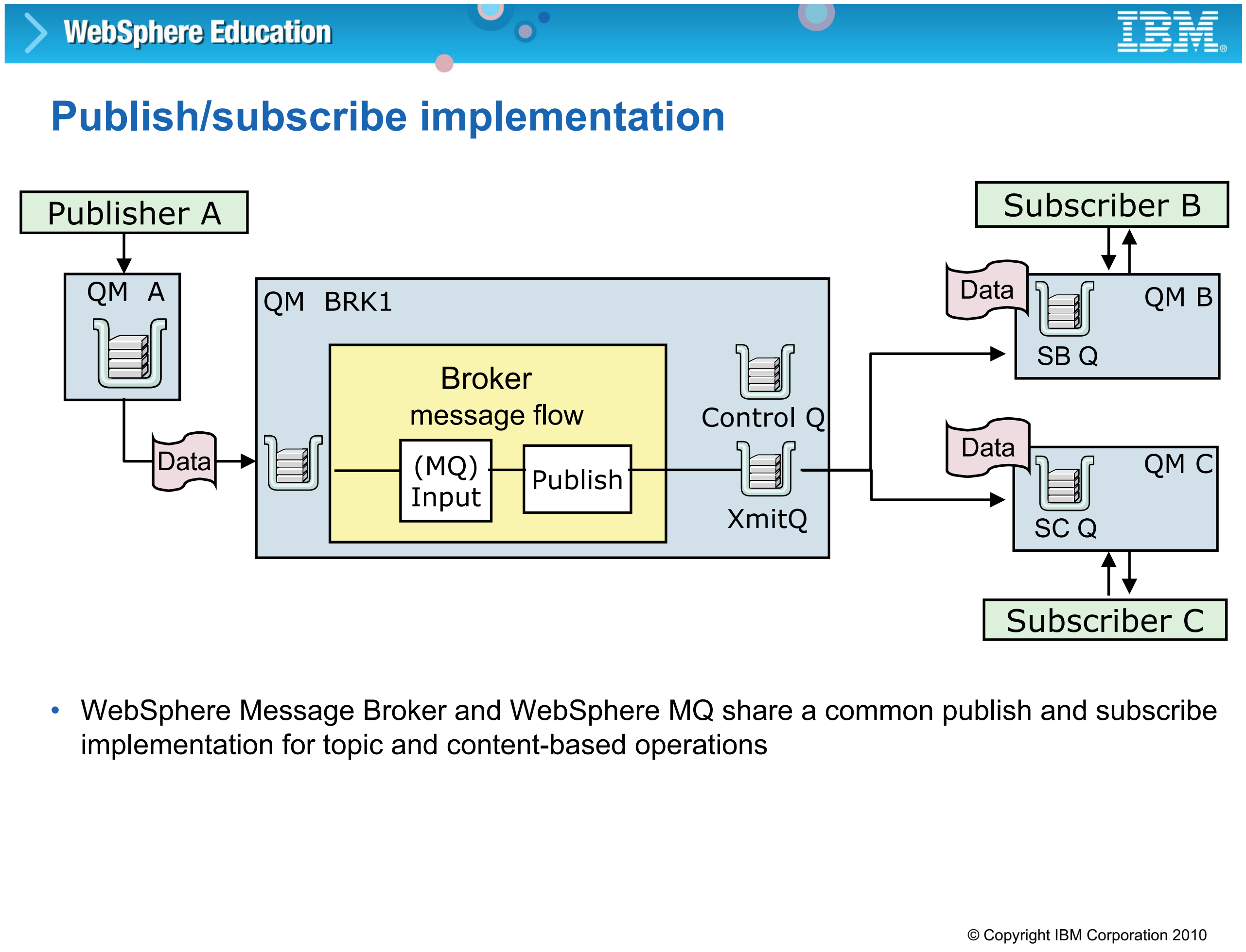
Instructor notes:

Purpose — Introduce the brokering role in application connectivity.

Details — Emphasize the role of the WebSphere MQ queue manager in the broker architecture.

Additional information — None.

Transition statement — There is additional functional support in WebSphere that allows the destination application to dynamically participate in the control of which messages it receives. This function is known as publish/subscribe.


WebSphere Education

Publish/subscribe implementation

- WebSphere Message Broker and WebSphere MQ share a common publish and subscribe implementation for topic and content-based operations

Figure 1-10. Publish/subscribe implementation WM643 / VM6431.0

Notes:

Publish/subscribe is a style of asynchronous messaging where the senders (publishers) of information are decoupled from the consumers (subscribers) of that information.

As shown in the figure, messages are supplied to the message flow by applications that publish messages, referred to as *publishers*, and retrieved from the message flow by applications that have registered a subscription with a broker, referred to as *subscribers*. A *subscription* defines a subscriber's interest in published messages.

WebSphere Message Broker V7 uses the publish/subscribe engine provided by WebSphere MQ V7.0.1 to control the interactions between the publishers and subscribers. It is the publish/subscribe engine in the queue manager that receives the messages from publishers, subscriptions from the subscribers, and eventually routes the published data to the target subscribers.

WebSphere MQ V7.0.1 supports filtering on the header of the message only. WebSphere Message Broker supports filtering on the whole message, including the body of the message.

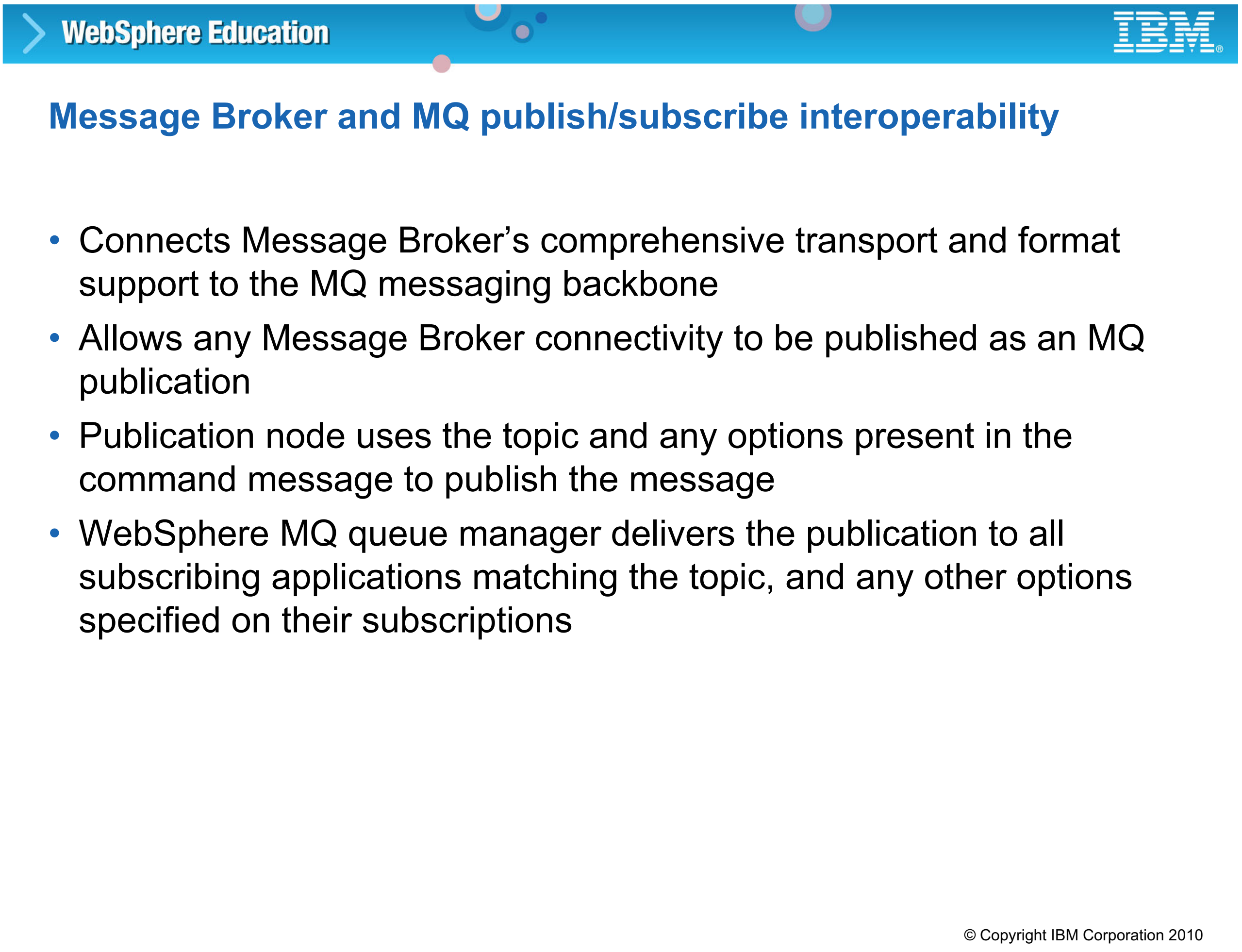
Instructor notes:

Purpose — Introduce publish/subscribe as a means of indirect communication.

Details — None.

Additional information — None.

Transition statement — Interoperability between WebSphere Message Broker and WebSphere MQ provides a number of advantages.



WebSphere Education

Message Broker and MQ publish/subscribe interoperability

- Connects Message Broker’s comprehensive transport and format support to the MQ messaging backbone
- Allows any Message Broker connectivity to be published as an MQ publication
- Publication node uses the topic and any options present in the command message to publish the message
- WebSphere MQ queue manager delivers the publication to all subscribing applications matching the topic, and any other options specified on their subscriptions

Figure 1-11. WebSphere Message Broker and WebSphere MQ publish/subscribe interoperability WM643 / VM6431.0

Notes:

All topic-based publish/subscribe operations are handled by WebSphere MQ. You can use WebSphere Message Broker facilities to extend publish/subscribe options to include content-based publishers and subscribers.

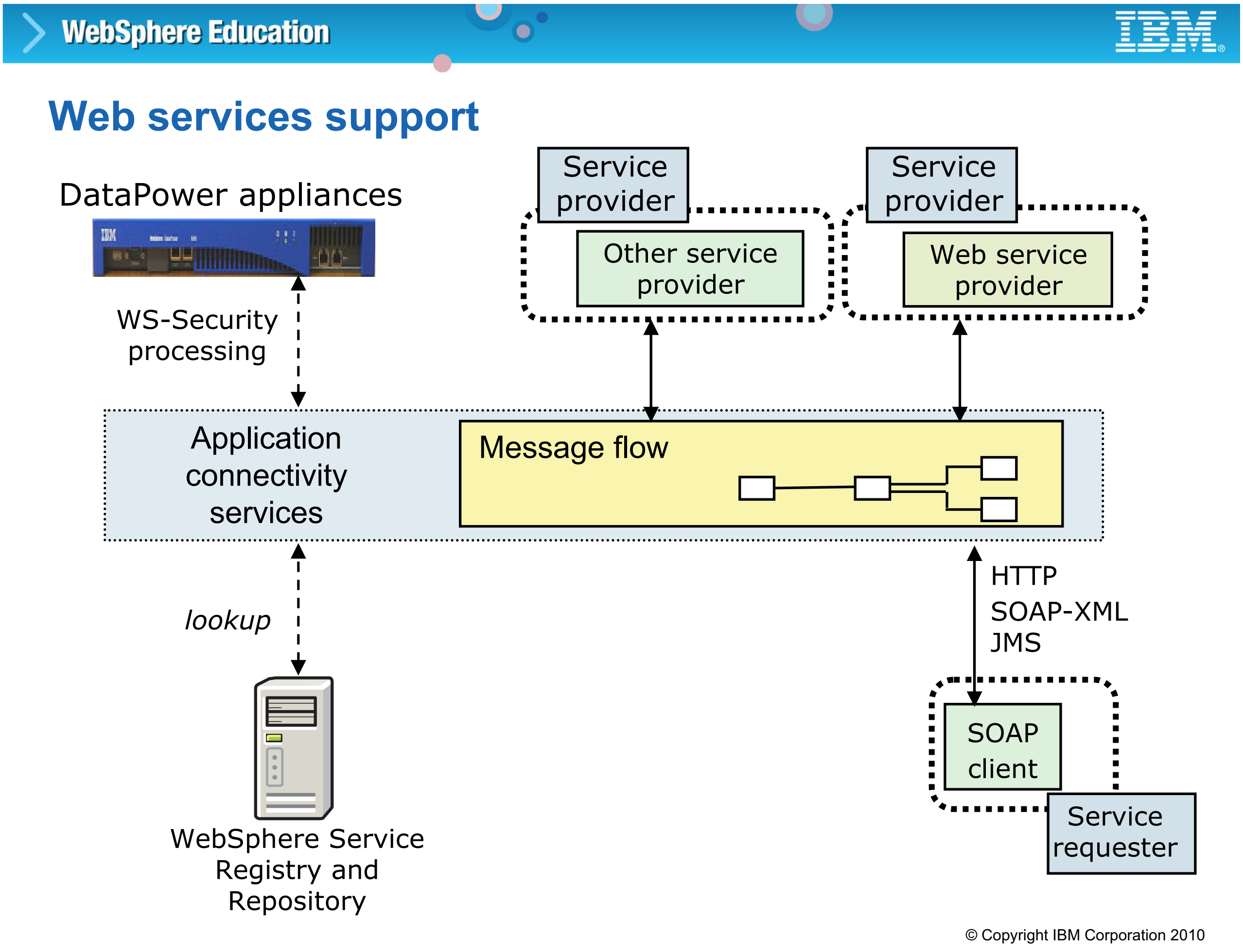
Instructor notes:

Purpose — Describe key features of WebSphere Message Broker and WebSphere MQ for publish/subscribe interoperability.

Details — One of the major enhancements for WebSphere Message Broker V7 is more reliance on WebSphere MQ to provide the publish/subscribe backbone services.

Additional information —

Transition statement — Although the general concept of input to a message flow or output from a message flow implies a WebSphere MQ message, that is not always the case. WebSphere Message Broker can also take advantage of web services. In fact, a message flow can itself be exposed as a web service.



WebSphere Education

Web services support

DataPower appliances

WS-Security processing

Service provider

Other service provider

Service provider

Web service provider

Application connectivity services

Message flow

HTTP
SOAP-XML
JMS

lookup

WebSphere Service Registry and Repository

SOAP client

Service requester

Figure 1-12. Web services support WM643 / VM6431.0

Notes:

WebSphere Message Broker provides facilities for using web services in the application connectivity services layer. SOAP messages are the form of data that is passed between web services. SOAP messages are expressed in XML and can be delivered using HTTP or JMS transports. Both transport protocols are directly supported by WebSphere Message Broker. Web services can be requested and aggregated in a message flow. A message flow can be exposed as a web service, listening on HTTP or JMS for incoming SOAP requests.

WebSphere Message Broker can use the IBM DataPower Appliance to handle its web services security (WS-Security) processing through the use of HTTP or HTTPS encryption and decryption.

Web services can be referenced dynamically in IBM WebSphere Service Registry and Repository (WSRR) which is a central repository of documents describing services, service interfaces, and associated policies that control access mechanisms such as WS-Policy documents.

Instructor notes:

Purpose — Emphasize support for web services, natively and through the use of the DataPower appliance.

Details — WS-Security describes enhancements to SOAP messaging to provide quality of protection through message integrity, message confidentiality, and single message authentication. WS-Security mechanisms can be used to accommodate a wide variety of security models and encryption technologies.

Generic XML documents, WSDL, service creation description language (SCDL), and other formats can be stored in the WebSphere Service Registry and Repository, although some queries might only apply to certain document types. For example, a query for a port type can only be done on WSDL documents.

Additional information —

Transition statement — Next: Adapters, bridges, and clients

Adapters, bridges, and clients

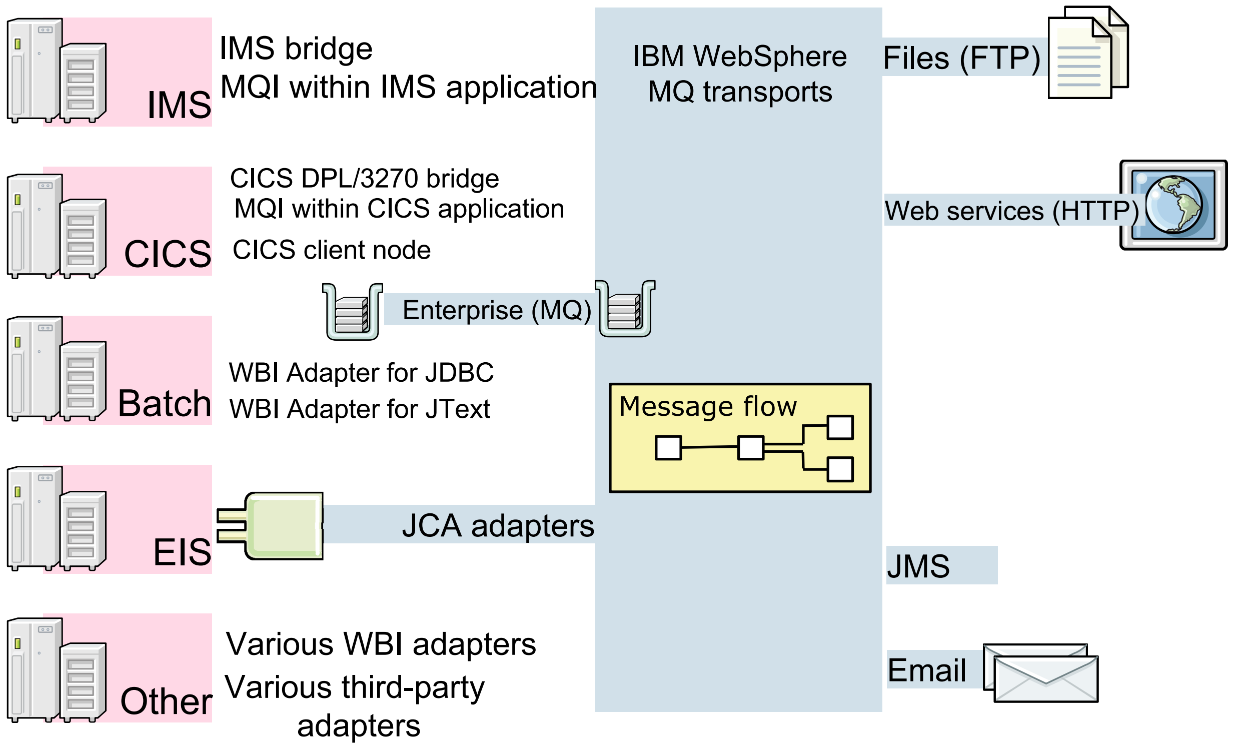


Figure 1-13. Adapters, bridges, and clients WM643 / VM6431.0

Notes:

Some applications do not directly support a WebSphere MQ transport protocol. In this case, various adapters, bridges, and clients are available to connect these legacy applications with WebSphere Message Broker.

Adapters are available from many sources. They might be part of the product, such as the CICS and IMS bridges, a separate IBM product, freeware, a service offering, or vendor software.

This figure shows some of the adapters, bridges, and clients supported by IBM WebSphere MQ, and WebSphere Message Broker.

Instructor notes:

Purpose — Explain that after the data is put onto a WebSphere MQ transport, the WebSphere Message Brokers can handle it. It does not matter where the message originated or where it is to be delivered.

Details — None.

Additional information — Perhaps now is a good time for a postal service analogy. Usually, mail is acquired from mailboxes or what might be considered a postal service *native* input source. However, mail drops exist in nonpostal service facilities, such as a *pack-and-ship* retail outlet. From the postal services perspective, the pack-and-ship store functions like one of the adapters presented here.

Transition statement — WebSphere Message Broker relies on many WebSphere MQ services, including administration services.

WebSphere Education

WebSphere MQ interoperability

- Single administrative explorer for WebSphere Message Broker and WebSphere MQ operations and security
- Publish/subscribe topology defined with WebSphere MQ V7 facilities
- Support and exploitation of WebSphere MQ multiple instance queue managers for high availability

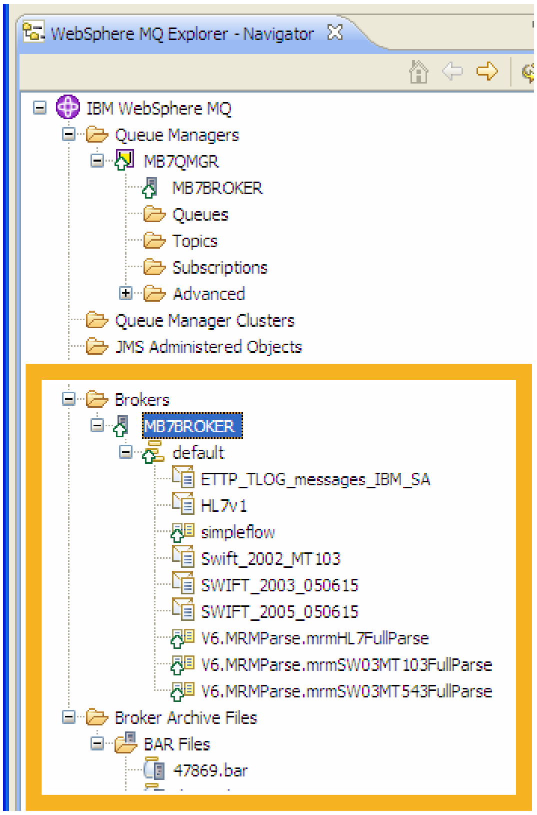


Figure 1-14. WebSphere MQ interoperability WM643 / VM6431.0

Notes:

WebSphere Message Broker relies on WebSphere MQ for a number of key functions. These include message transport, administration, security, publish/subscribe topology, and high availability.

As shown in the figure, WebSphere Message Broker Explorer is an extension to WebSphere MQ Explorer, providing a single administration interface for managing queues, queue managers, topics and subscriptions, and broker components.

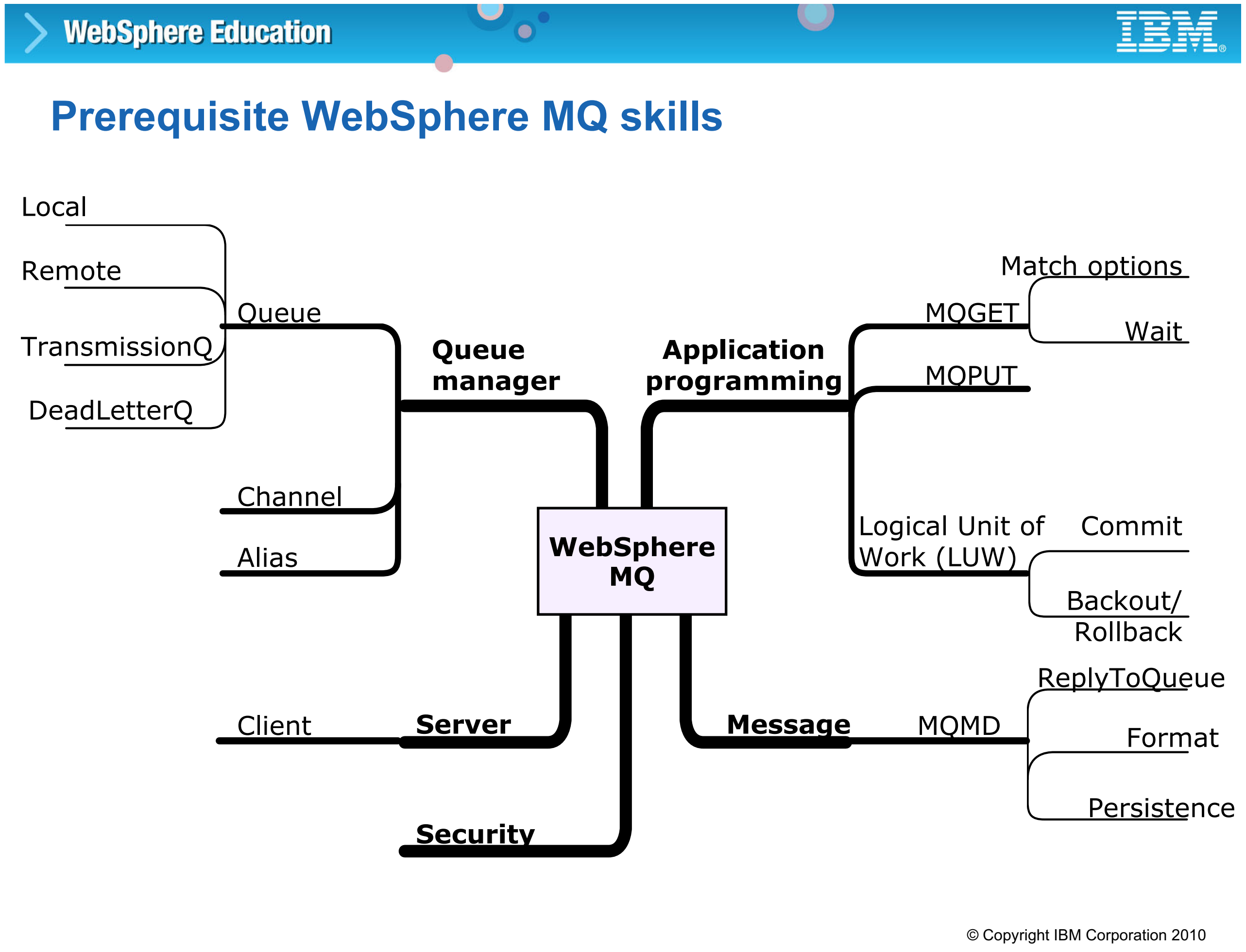
Instructor notes:

Purpose — Provide an overview of some of the WebSphere Message Broker functions provided by WebSphere MQ.

Details — Some key services that were supported in prior versions of WebSphere Message Broker are now provided by WebSphere MQ.

Additional information —

Transition statement — The tighter integration between WebSphere Message Broker and WebSphere MQ requires a prerequisite knowledge of WebSphere MQ.



WebSphere Education

Prerequisite WebSphere MQ skills

WebSphere MQ

- Queue manager
  - Queue
    - Local
    - Remote
    - TransmissionQ
    - DeadLetterQ
  - Channel
  - Alias
- Application programming
  - MQGET
    - Match options
    - Wait
  - MQPUT
  - Logical Unit of Work (LUW)
    - Commit
    - Backout/ Rollback
- Server
  - Client
- Security
- Message
  - MQMD
    - ReplyToQueue
    - Format
    - Persistence

Figure 1-15. Prerequisite WebSphere MQ skills

WM643 / VM6431.0

Notes:

WebSphere Message Broker is basically a WebSphere MQ application, so knowledge of WebSphere MQ is essential.

The figure identifies some of the key WebSphere MQ concepts. Check the terms and mark each:

x = I know that.

! = I think I know, but some explanation might be helpful.

? = I need help understanding the concepts.

Instructor notes:

Purpose — Set the level of prerequisite knowledge about WebSphere MQ.

Details — Even though successful completion of a WebSphere MQ administration course is a prerequisite, some students might have little technical knowledge of WebSphere MQ. This small exercise might help to expose the deficiencies in students' WebSphere MQ skills without making the students feel uncomfortable.

Give the students a moment to mark the terms on the visual.

Then, give them a chance to ask questions. Use any marking tools available (dry erase pens on a whiteboard, the *marker tool* of Acrobat Reader or Freelance) to help draw attention to the items that require more clarification in this class.

Take the time to give a short explanation of the topics that have been marked. Spending time now might save you much time later in the course.

Also consult the instructor notes on the topic introduction page for this topic for references to other sources that can help students understand the basic concepts of WebSphere MQ.

Additional information — None.

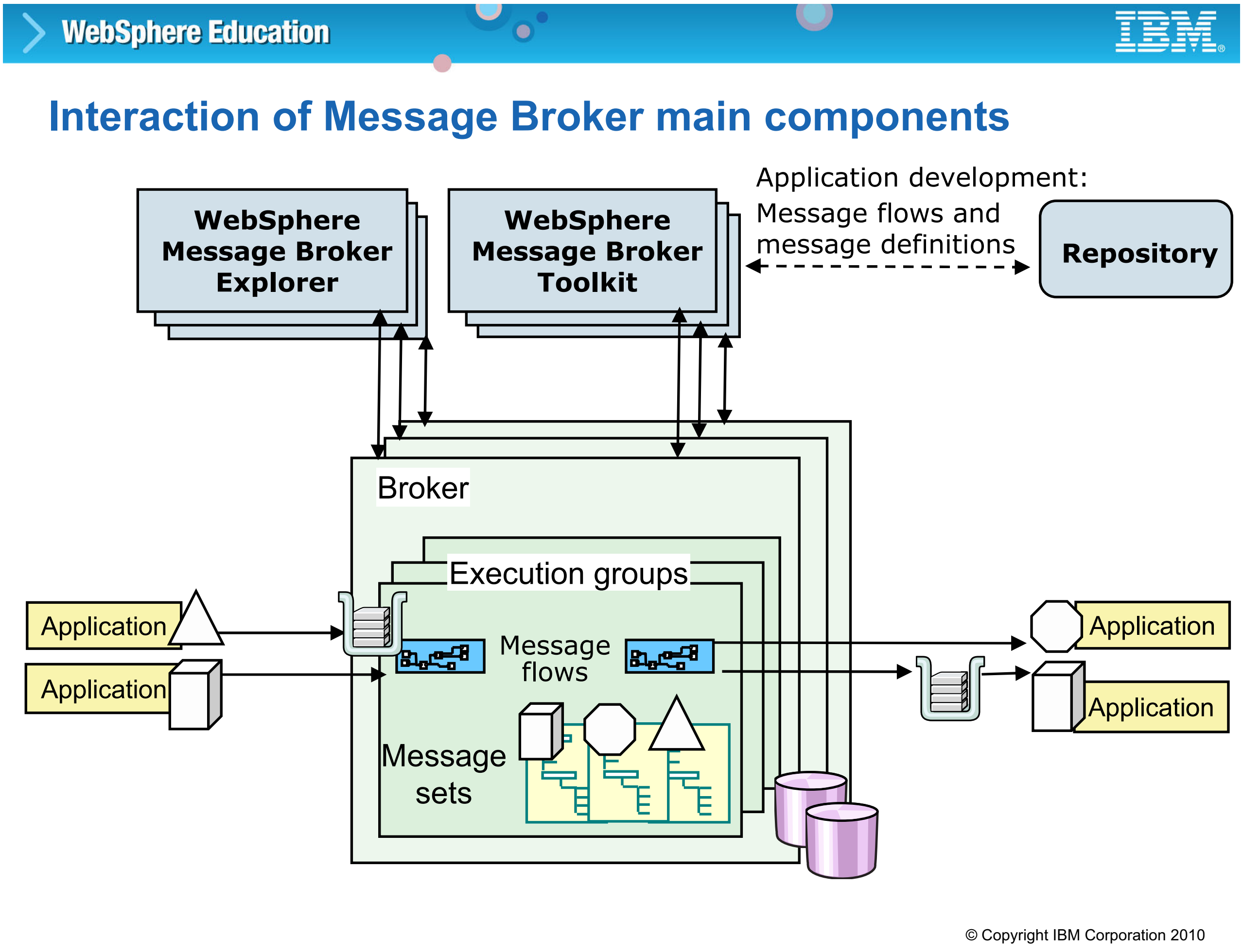
Transition statement — Next you look at the architecture and components of WebSphere Message Broker.

1.3. Development and runtime components

Instructor topic introduction

In this topic, students learn that there is a single runtime component, the broker, and multiple options to manage the broker and its resources. The concept of a single runtime component a crucial part of the course, so take the time to explain thoroughly and encourage discussion

How this will help students on their job — Students will be able to better plan their own configuration architecture.



WebSphere Education

Interaction of Message Broker main components

WebSphere Message Broker Explorer
WebSphere Message Broker Toolkit
Application development: Message flows and message definitions
Repository
Broker
Execution groups
Message flows
Message sets
Application
Application
Application
Application

Figure 1-16. Interaction of WebSphere Message Broker main components

WM643 / VM6431.0

Notes:

WebSphere Message Broker consists of a single runtime component, the broker. WebSphere Message Broker Explorer, WebSphere Message Broker Toolkit, commands, and applications that are written to the WebSphere Message Broker Application (CMP) API for can be used to connect directly to the broker to manage the broker and its resources.

WebSphere Message Broker Toolkit is an integrated development environment for developing and testing message flows and message flow components. Development artifacts can be stored on the local drive and managed by an external repository.

WebSphere Message Broker Explorer is the administration interface for creating and managing WebSphere Message Broker components such as brokers and execution groups.

The broker is the runtime component where message flows run in processes called *execution groups*. A message flow might need message dictionaries (*message sets*) to parse and construct predefined messages. There can be many brokers, and each can be running on a different system which provides protection against failure, and can separate work across different divisions in a business.

Instructor notes:

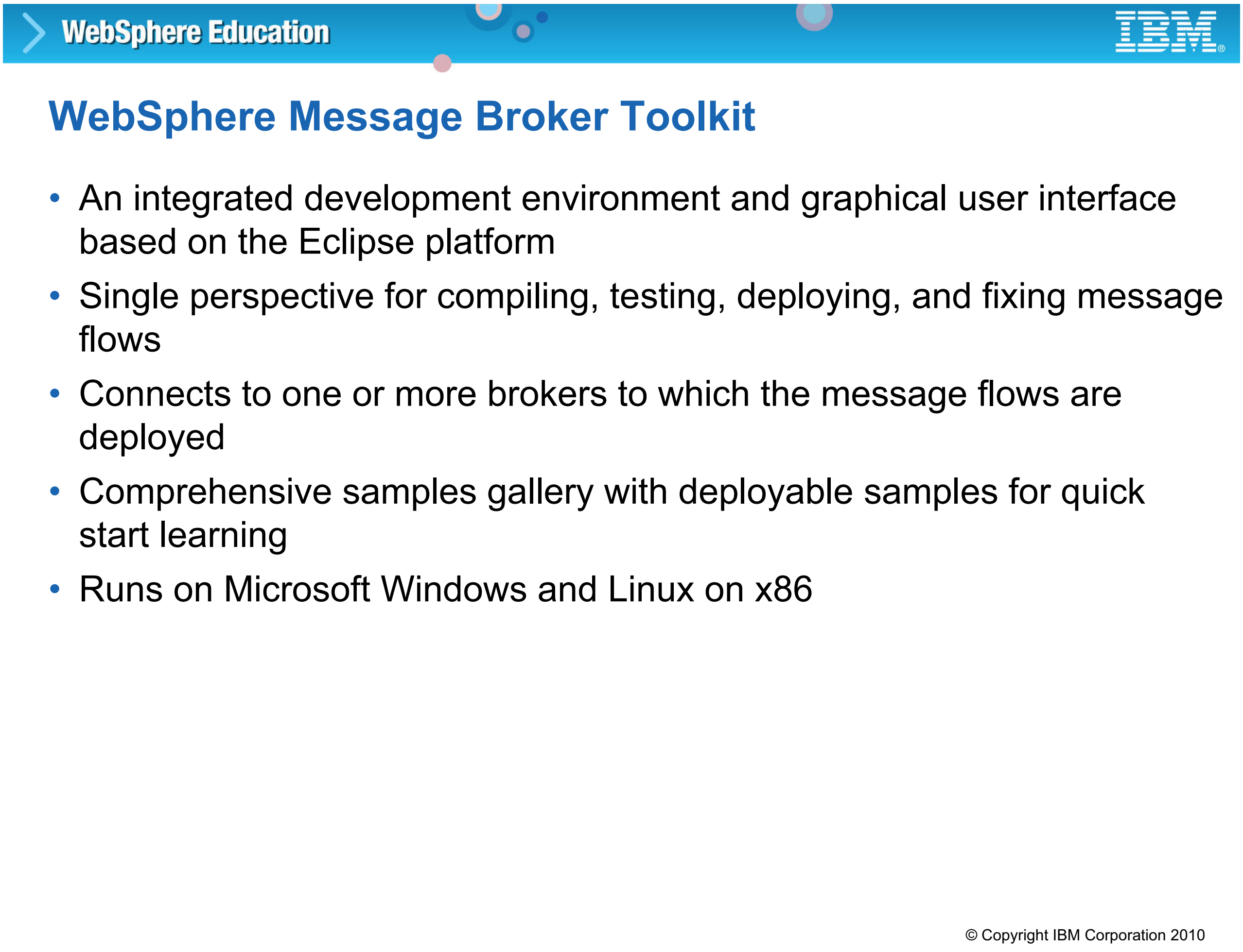
Purpose — Introduce the components that make up a broker domain.

Details — The WebSphere Message Broker Toolkit is an integrated development environment. Development objects are stored in the local file system. They might also be stored and managed by an optional external repository product.

The persistent configuration of the broker is now stored exclusively on the file system, which means that there is no longer a requirement to create and maintain a system database, and no database product is required unless you want to access user databases.

Additional information — More information about message flows and message sets is provided later in this unit.

Transition statement — You now take a closer look at each component, starting with the WebSphere Message Broker Toolkit.



WebSphere Education
IBM

WebSphere Message Broker Toolkit

- An integrated development environment and graphical user interface based on the Eclipse platform
- Single perspective for compiling, testing, deploying, and fixing message flows
- Connects to one or more brokers to which the message flows are deployed
- Comprehensive samples gallery with deployable samples for quick start learning
- Runs on Microsoft Windows and Linux on x86

Figure 1-17. The WebSphere Message Broker Toolkit WM643 / VM6431.0

Notes:

The WebSphere Message Broker Toolkit is an integrated development environment (IDE) and graphical user interface (GUI) based on the open source Eclipse platform. Application developers work in separate instances of the WebSphere Message Broker Toolkit (also known as workbenches) to develop message sets and message flows.

The WebSphere Message Broker Toolkit runs on Windows and Linux and uses the local file system to store its components.

The main tasks that can be performed in the toolkit include:

- Defining message flows
- Defining and importing message definitions
- Deploying message flows and message sets to brokers
- Starting, stopping, and tracing message flows running in brokers

The WebSphere Message Broker Toolkit requires a server-connection channel for connecting to a remote broker. A default SVRCONN channel called `SYSTEM.BKR.CONFIG` is created when you create a broker.

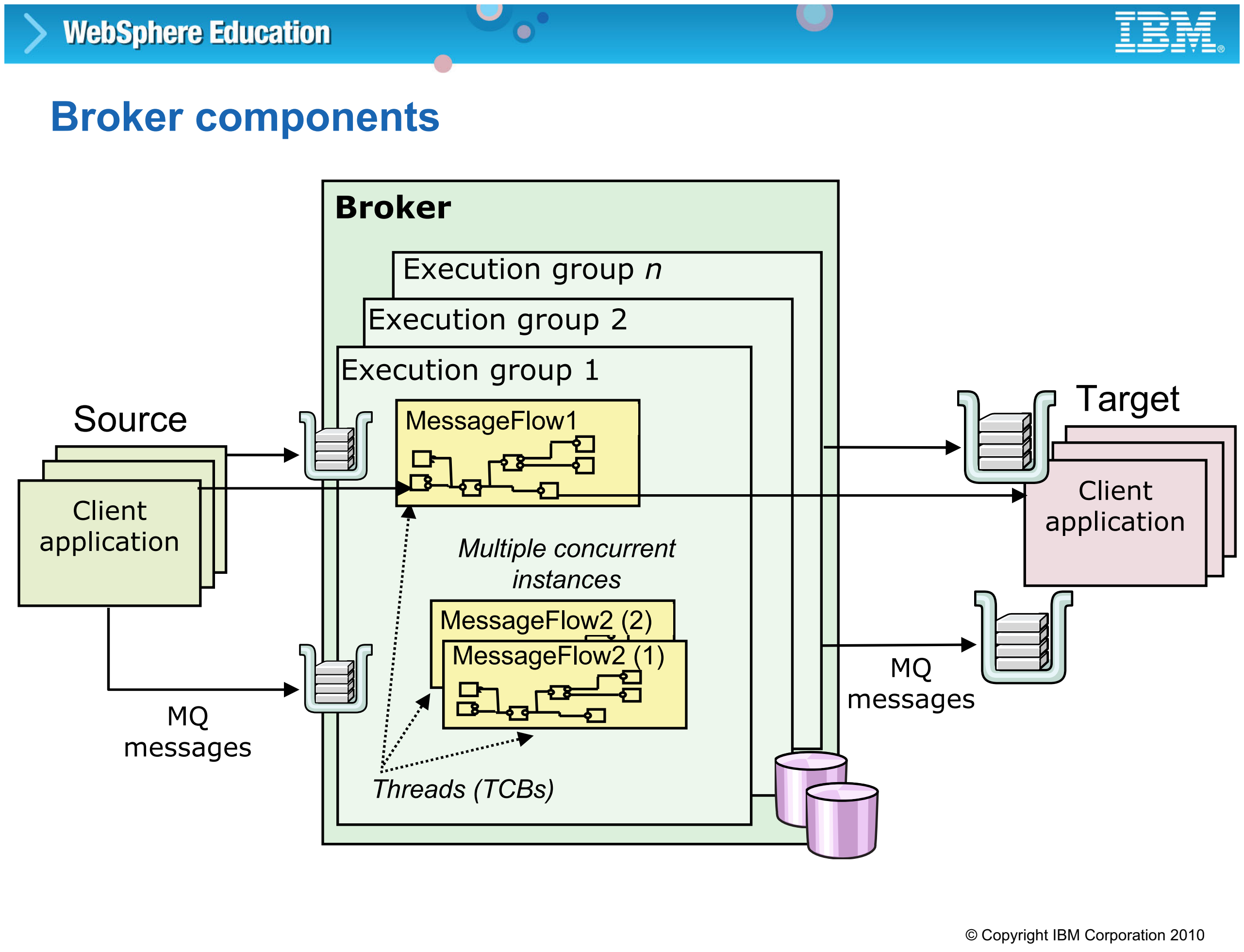
Instructor notes:

Purpose — Introduce WebSphere Message Broker Toolkit concepts.

Details — An upcoming unit and exercise are dedicated to using the WebSphere Message Broker Toolkit to import, deploy, and test message flows.

Additional information — You might suggest that security plays a part in determining who performs specific tasks from a toolkit instance. There is more on security in the next lecture unit and subsequent units.

Transition statement — Next, you examine the broker, or runtime, component.



WebSphere Education

Broker components

Figure 1-18. Broker components WM643 / VM6431.0

Notes:

The broker can run on various Windows platforms, Linux, Sun Solaris, HP Itanium, and z/OS. A broker is a system service on Windows platforms, a daemon process on UNIX platforms, or a started task on zSeries platforms. It controls processes that run message flows.

Applications often send messages to the broker using WebSphere MQ queues and connections. The broker routes each message using the rules defined in message flows and message sets, and transforms the data into the structure required by the receiving applications. The broker also uses a set of queues during its operations. These queues (named `SYSTEM.BROKER.`*`xxx`*`.`*`yyy`*) are automatically created when the broker is created. Because each broker uses queues of the same name, only one broker can be associated a WebSphere MQ queue manager.

The broker contains one or more processes called *execution groups*. which are the engines responsible for running the message flows. Message flows run as threads within an execution group. When a message flow is deployed, the broker automatically starts an instance of the message flow.

Instructor notes:

Purpose — Explain some broker concepts.

Details — It is possible to have up to 255 additional (concurrent) instances of the same message flow within one execution group. This means that messages can be processed in random order and therefore should not have affinities. An input option permits messages originating from the same user ID to be processed in sequence which is one way to ensure message processing order.

Each message flow can be assigned to one or more execution groups, providing memory separation. Running the same message flow in multiple execution groups increases throughput, but there is no way to influence the order in which messages are processed.

The broker also uses a set of queues during its operations. These queues (named SYSTEM.BROKER.xxx.yyy) are automatically created when the broker is created. Because each broker uses queues of the same name, only one broker can be associated a WebSphere MQ queue manager.

Execution groups can handle many different message flows. The number of total threads within an execution group is a restriction of the environment, not the execution group. Similarly, the number of execution groups assigned to a broker is limited by the environment, not the product.

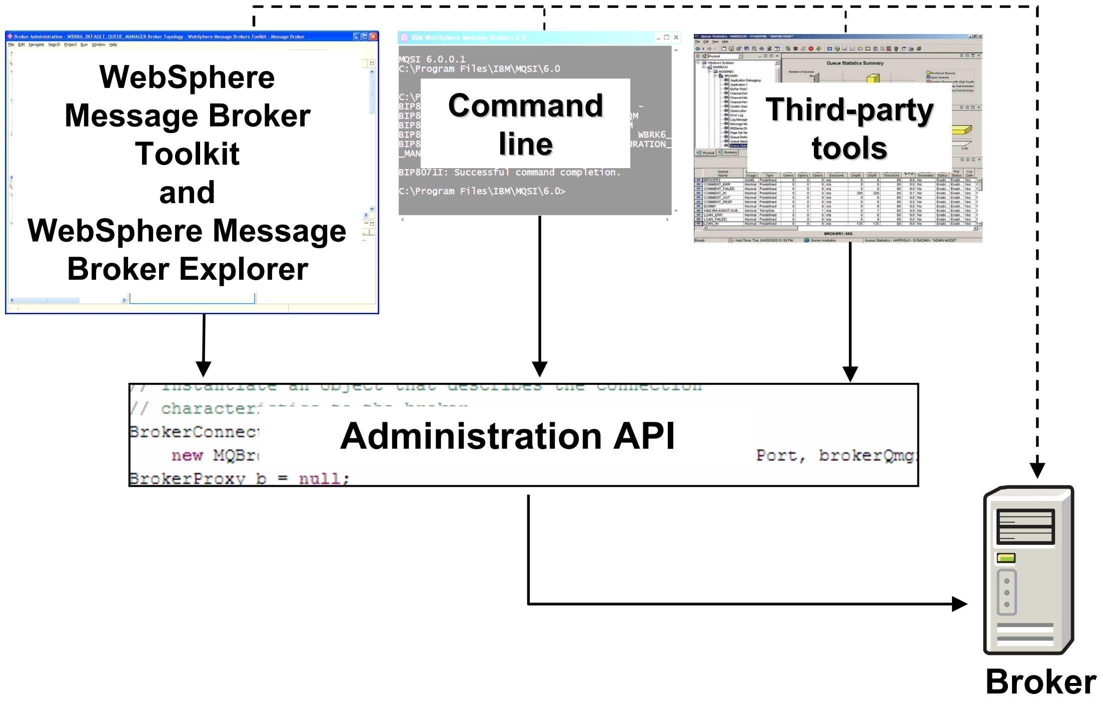
There can be one or many brokers running on a mixture of all supported operating systems. Each broker has an individual name. The various components in a domain communicate with each other using WebSphere MQ messages. The format of this information is XML.

Additional information — Broker queues are described in more detail in the next unit. Security plays a role in broker administration as well.

Transition statement — None.

WebSphere Education

Broker administration tools


WebSphere Message Broker Toolkit and WebSphere Message Broker Explorer
Command line
Third-party tools
Administration API
Broker

© Copyright IBM Corporation 2010

Figure 1-19. Broker administration tools WM643 / VM6431.0

Notes:

The broker can be administered using the WebSphere Message Broker Explorer, the **Brokers** view in the WebSphere Message Broker Toolkit, or the product commands. You can also write your own programs to use the Administration API (also known as the CMP API) for WebSphere Message Broker.

A default WebSphere MQ SVRCONN channel with a fixed name `SYSTEM.BKR.CONFIG` is used by the WebSphere Message Broker Toolkit, the WebSphere Message Broker Explorer, and applications that use the Administration API (CMP API) to connect to the broker. The server connection channel is created when the broker is created.

Instructor notes:

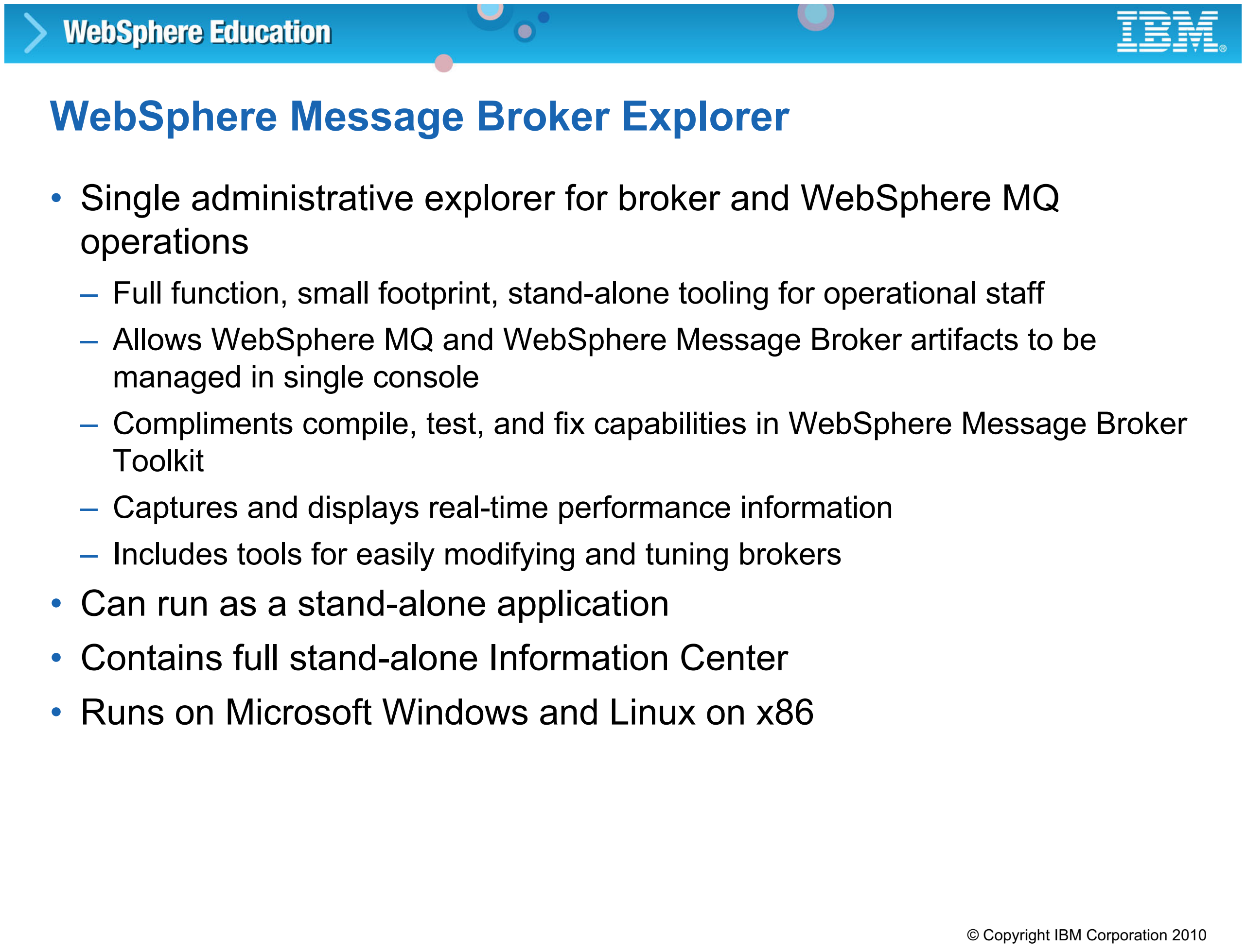
Purpose — Discuss the various environments where the administrator might work.

Details — As a rule, anything you can do in the WebSphere Message Broker Explorer can be achieved programmatically using the Administration API.

Additional information — In previous versions of WebSphere Message Broker, the API was known as the Configuration Manager Proxy (CMP) API. Applications provided by WebSphere Message Broker, such as the WebSphere Message Broker Toolkit and some commands, and applications that you wrote yourself, connected to the Configuration Manager to achieve specific tasks.

The Configuration Manager has been removed from Version 7, and the full name of the API has changed. However, the terms CMP application and CMP API have been retained, and are used in the information center to refer to the Administration API, for continuity and consistency with the JAR file `ConfigManagerProxy.jar t`hat supplies all the classes.

Transition statement — The next series of slides look at each of the administration tools in more detail. First, look at WebSphere Message Broker Explorer; the administrative console for both queue managers and brokers.



WebSphere Education

WebSphere Message Broker Explorer

- Single administrative explorer for broker and WebSphere MQ operations
  - Full function, small footprint, stand-alone tooling for operational staff
  - Allows WebSphere MQ and WebSphere Message Broker artifacts to be managed in single console
  - Compliments compile, test, and fix capabilities in WebSphere Message Broker Toolkit
  - Captures and displays real-time performance information
  - Includes tools for easily modifying and tuning brokers
- Can run as a stand-alone application
- Contains full stand-alone Information Center
- Runs on Microsoft Windows and Linux on x86

Figure 1-20. WebSphere Message Broker Explorer WM643 / VM6431.0

Notes:

WebSphere Message Broker Explorer is a stand-alone tool that installs into WebSphere MQ Explorer. You can manage brokers and queue managers from within a single interface. Most operators are already using WebSphere MQ Explorer for queue management. This single console for both queues and brokers uses the same interfaces as WebSphere MQ, so there is no requirement to learn another interface to manage brokers.

Note

You can view and interact only with brokers that you have been created in WebSphere Message Broker V7.0.

Using WebSphere Message Broker you can create local brokers (Windows or Linux only) and manage local and remote brokers.

Instructor notes:

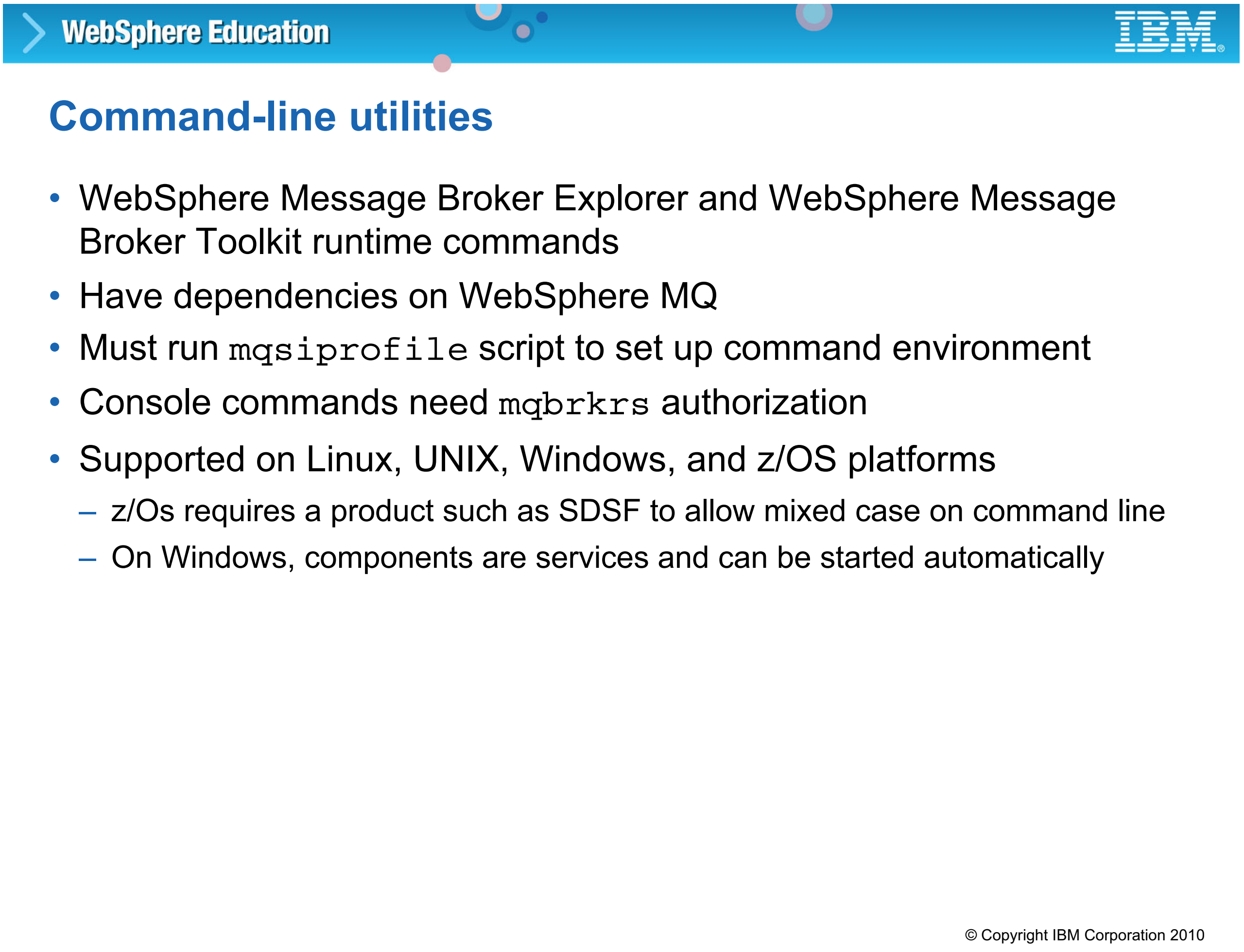
Purpose — Describe the features of WebSphere Message Broker Explorer.

Details — WebSphere Message Broker Explorer is described in more detail in the remaining units of this course.

Emphasize that WebSphere Message Broker Explorer can be used to create and manage local brokers on platforms running WebSphere Message Broker Explorer and that it can also be used to manage remote brokers.

Additional information — None

Transition statement — None.



WebSphere Education

Command-line utilities

- WebSphere Message Broker Explorer and WebSphere Message Broker Toolkit runtime commands
- Have dependencies on WebSphere MQ
- Must run `mqsiprofile` script to set up command environment
- Console commands need `mqbrkrs` authorization
- Supported on Linux, UNIX, Windows, and z/OS platforms
  - z/Os requires a product such as SDSF to allow mixed case on command line
  - On Windows, components are services and can be started automatically

Figure 1-21. Command-line utilities WM643 / VM6431.0

Notes:

WebSphere Message Broker administrative tasks can be performed in a command window on all platforms. Some commands, such as `mqsistart`, access the local components directly. Other commands, such as `mqsicreatebar`, require access to the WebSphere Message Broker Toolkit workspace resources and can only run on Windows or Linux.

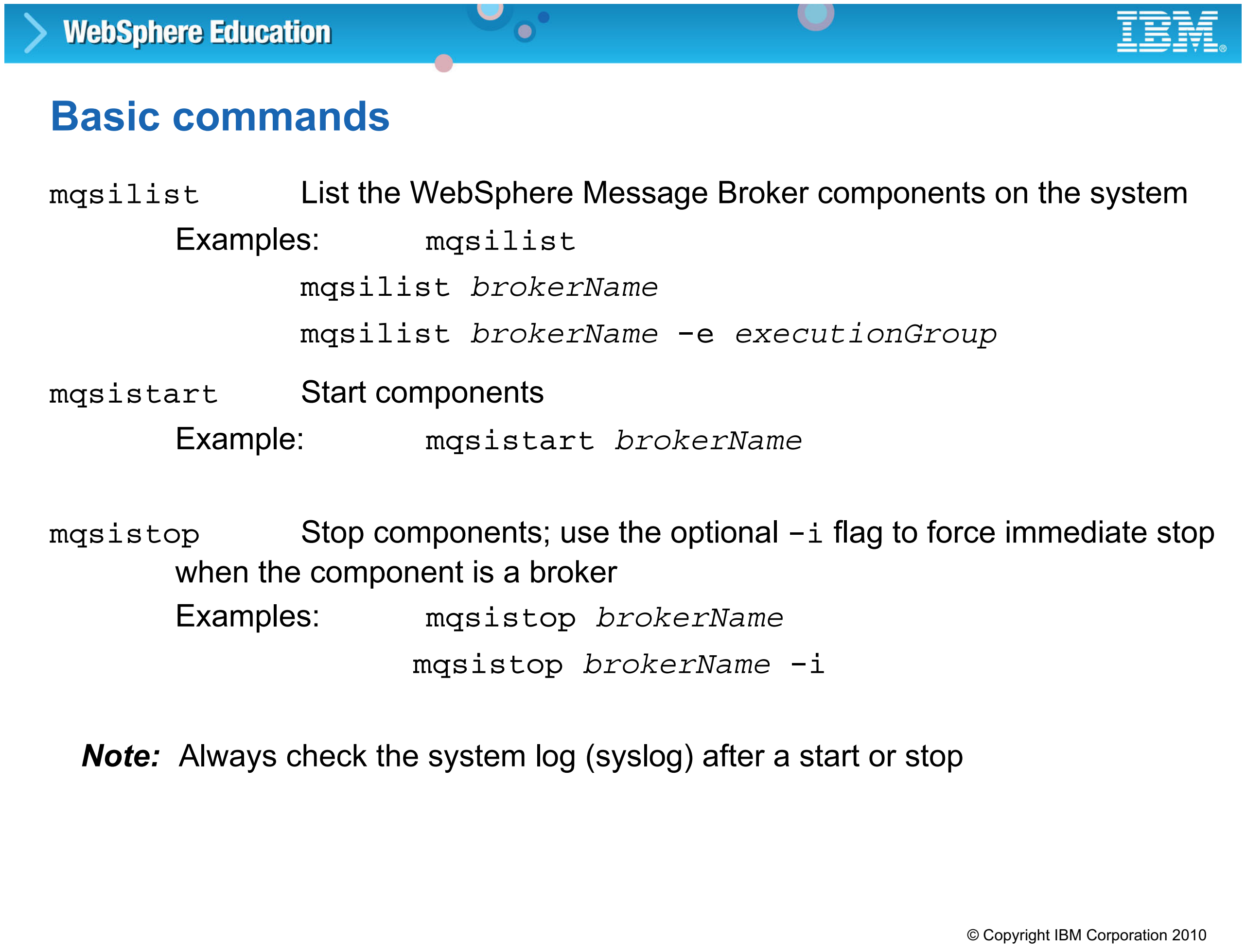
Instructor notes:

Purpose — Provide overview of command-line utilities.

Details — Students should be somewhat familiar with using a command line to manage WebSphere MQ.

Additional information — There is much more detail on commands in later units.

Transition statement — Here are some of the basic commands you need to know.



Basic commands

`mqsilist` List the WebSphere Message Broker components on the system

Examples:

```
mqsilist
mqsilist brokerName
mqsilist brokerName -e executionGroup
```

`mqsistart` Start components

Example:

```
mqsistart brokerName
```

`mqsistop` Stop components; use the optional `–i` flag to force immediate stop when the component is a broker

Examples:

```
mqsistop brokerName
mqsistop brokerName –i
```

Note: Always check the system log (syslog) after a start or stop

© Copyright IBM Corporation 2010

Figure 1-22. Basic commands

WM643 / VM6431.0

Notes:

After installation, some additional steps are required before the WebSphere Message Broker can be used. A WebSphere Message Broker Toolkit wizard is available to guide you through a process that builds and starts a default configuration on a stand-alone system.

The WebSphere Message Broker Explorer or commands must be used to check the existence of WebSphere Message Broker components on a system, and to start or stop a component. The figure lists some of basic commands that can be used to control and monitor the broker.

In Windows, components can also be started from the Windows Services GUI accessed using the Control Panel folder. It is useful to configure them start automatically. Check at the same time that the service “IBM MQSeries” is also set to start automatically.

If services are set to start automatically, the order in which they start cannot be predicted or forced.

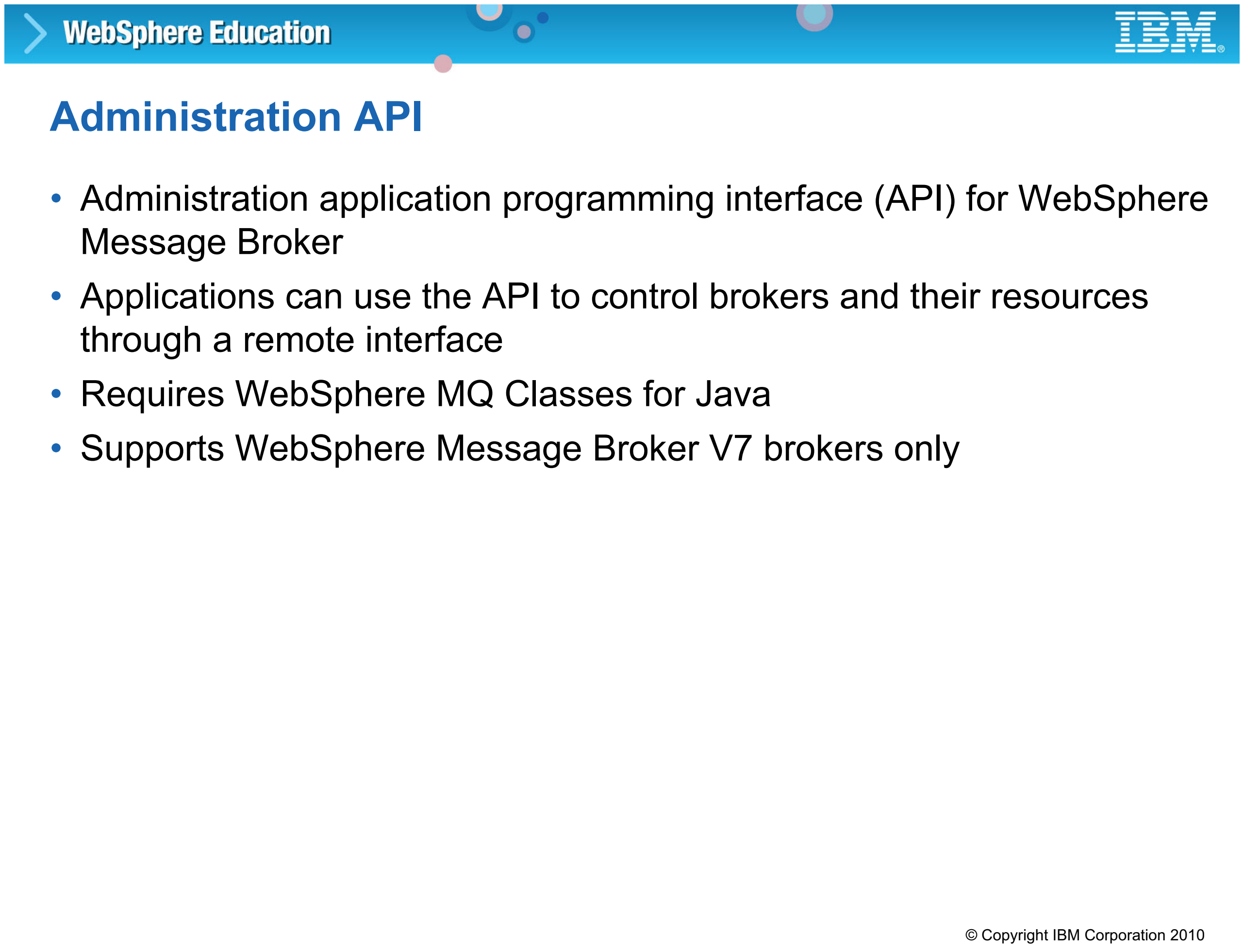
Instructor notes:

Purpose — Describe some useful commands.

Details — Do not go into detail on the specific commands at this point. The next unit describes these commands in much more detail.

Additional information — In Exercise 1, students manually create, and start a complete configuration, using the default configuration wizard.

Transition statement — If you want to create your own applications to manage the broker, you can connect to the Administration API for WebSphere Message Broker.



WebSphere Education

Administration API

- Administration application programming interface (API) for WebSphere Message Broker
- Applications can use the API to control brokers and their resources through a remote interface
- Requires WebSphere MQ Classes for Java
- Supports WebSphere Message Broker V7 brokers only

Figure 1-23. Administration API WM643 / VM6431.0

Notes:

The Administration API (formerly known as the Configuration Manager Proxy APi) is a lightweight set of Java classes that sit logically between the user application and the broker. It is an alternative interface for administering brokers.

Note

The WebSphere Message Broker V7 Administration API cannot be used to manage brokers from previous versions of WebSphere Message Broker.

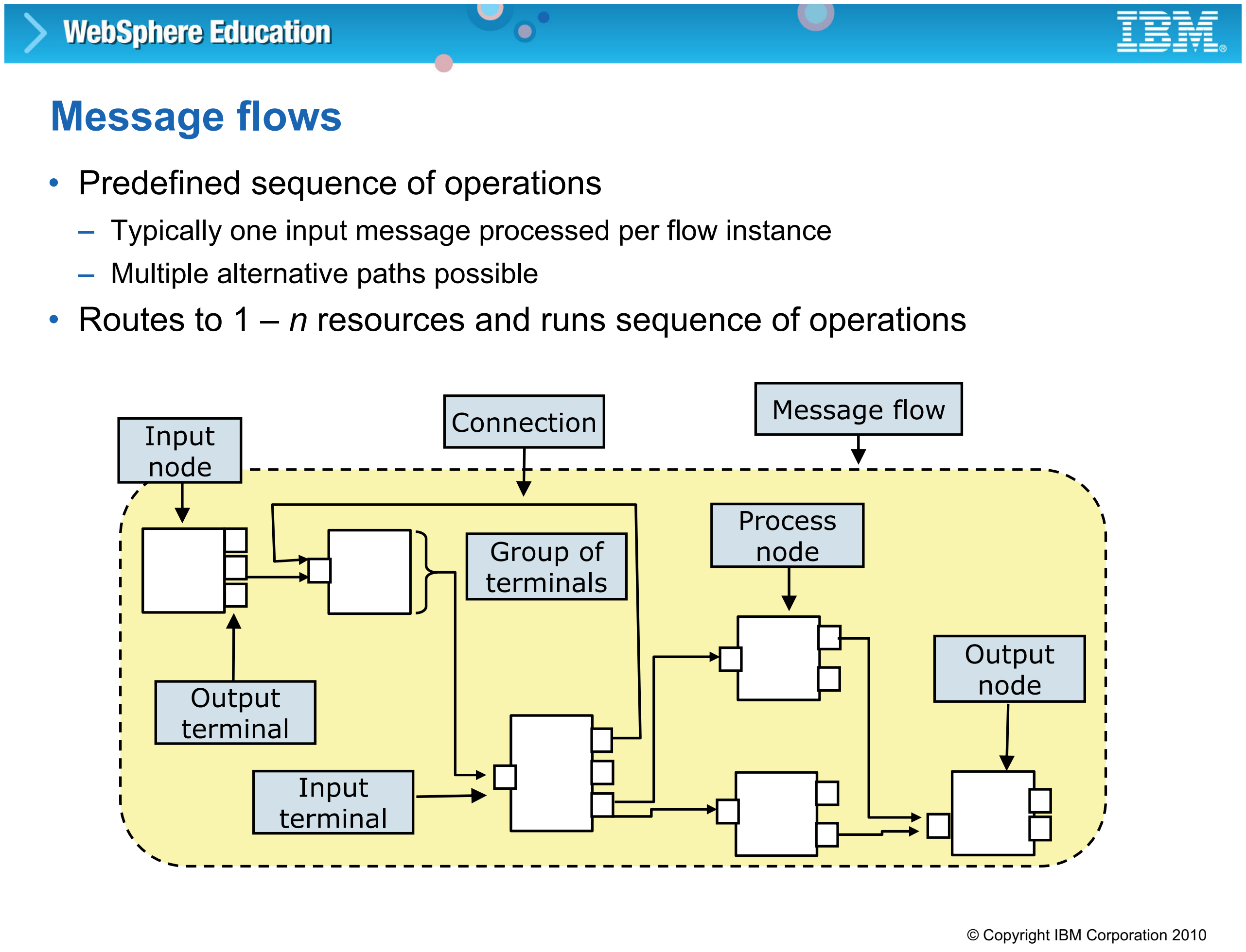
Instructor notes:

Purpose — Introduce the Administration API.

Details — The Administration API is covered in more detail in a later unit.

Additional information — None.

Transition statement — Now that you have learned a little about the management interfaces, it is time to examine the contents of a message flow in more detail.



WebSphere Education

Message flows

- Predefined sequence of operations
  - Typically one input message processed per flow instance
  - Multiple alternative paths possible
- Routes to 1 – *n* resources and runs sequence of operations

Figure 1-24. Message flows

WM643 / VM6431.0

Notes:

A *message flow* is a program created using graphical tools. The WebSphere Message Broker Toolkit provides both the environment (the workbench) and tools (Eclipse plug-ins) for defining the message flow. A typical message flow is *stateless*; each instance of a message flow is treated as an independent transaction that is unrelated to any previous request.

The figure shows the components of a message flow. It starts with at least one *input node*. A *process node* is an intermediate node where messages are used for various business processes. An *output node* usually places messages on a transport. It is possible to end a flow without an output node. *Input terminals* and *output terminals* are the connectors between nodes in a flow. The terminals of nodes are “wired” together with connections and indicate message flow direction.

A terminal might have multiple connections which results in a duplication of the message, or multiple, different messages. The order in which any of the outbound paths is taken is random, but the paths are run sequentially. Each path is followed until it ends, and then the next path is taken.

Instructor notes:

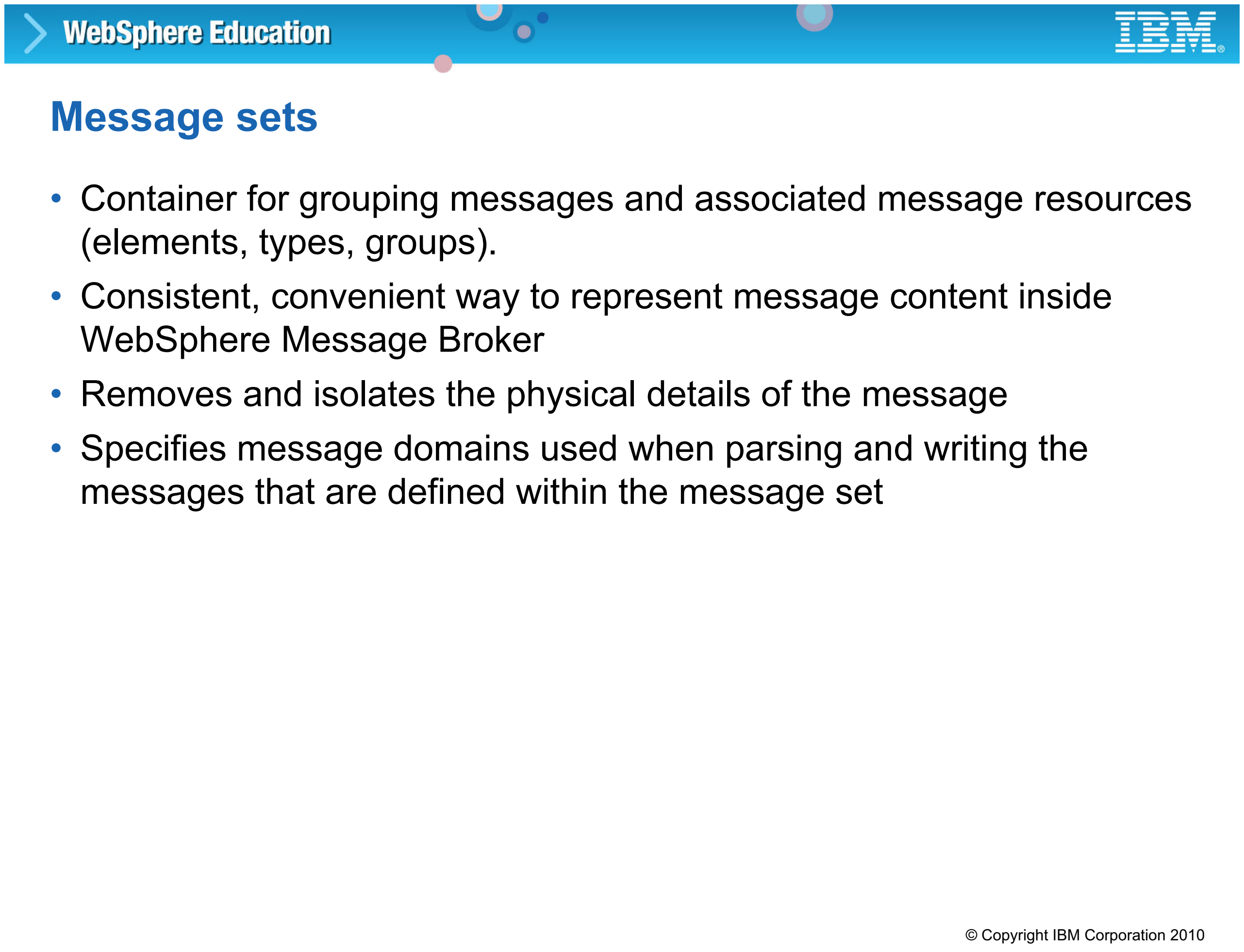
Purpose — Examine the parts that make up a message flow.

Details — Remind students of something stated earlier: a message flow is just an extension of the message originating program. Basic connectivity is being enhanced by the service the message flow provides.

This series of slides starts with a high-level view of a message flow, followed by a more detailed example of a message flow as it will appear in the toolkit. This is then followed by a more in-depth look at a node. These three slides should give administrators a good understanding of a message flow and nodes. The slide listing some of the nodes was added so that administrators can see the types of operations that can be supported in a message flow. It is not intended that the instructor go into any detail on any of the nodes.

Additional information — Having multiple connections to the same input terminal is not message aggregation.

Transition statement — Next: Message sets



WebSphere Education

Message sets

- Container for grouping messages and associated message resources (elements, types, groups).
- Consistent, convenient way to represent message content inside WebSphere Message Broker
- Removes and isolates the physical details of the message
- Specifies message domains used when parsing and writing the messages that are defined within the message set

Figure 1-25. Message sets WM643 / VM6431.0

Notes:

A message flow might need message dictionaries (*message sets*) to parse and construct predefined messages.

The message set specifies the domains used when parsing and writing messages. WebSphere Message Broker provides built-in support for messages in the following message domains by providing message body parsers:

- MRM
- XMLNSC, XMLNS, and XML
- SOAP
- Data Object
- JMSMap and JMSStream
- MIME
- BLOB

As an administrator, you need to know if the message flow refers to a message set. If the message flow refers to a message set, the message set file (`.mset`) must be deployed with the message flow file (`.msgflow`).

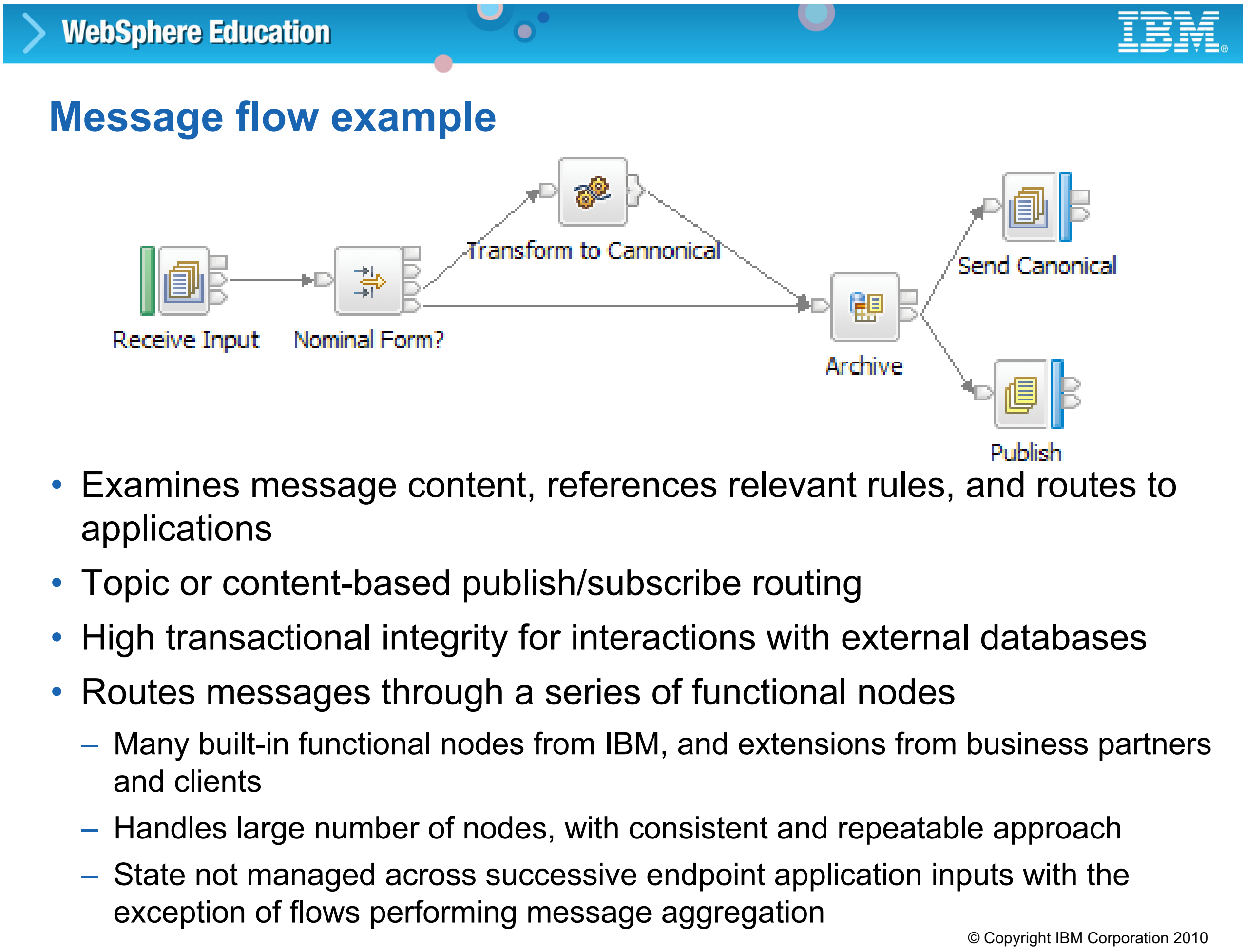
Instructor notes:

Purpose — Provide an overview of a message set.

Details — The important concept is that the message set is a deployable object.

Additional information — None.

Transition statement — Next is example of actual message flow as it would appear in the Message Flow editor in the WebSphere Message Broker Toolkit.



Message flow example

- Examines message content, references relevant rules, and routes to applications
- Topic or content-based publish/subscribe routing
- High transactional integrity for interactions with external databases
- Routes messages through a series of functional nodes
  - Many built-in functional nodes from IBM, and extensions from business partners and clients
  - Handles large number of nodes, with consistent and repeatable approach
  - State not managed across successive endpoint application inputs with the exception of flows performing message aggregation

Figure 1-26. Message flow example WM643 / VM6431.0

Notes:

Message flows can route messages based on message content. In the message flow, each node represents an operation.

In the figure, the first node is an MQInput node. It retrieves messages from a specified queue. The directional arrows are referred to as wires and show the direction of the message flow.

The message flow that is shown in the figure retrieves a message from a queue and transforms the message as it flows through the intermediate process nodes. It then sends the message to an WebSphere MQ output queue and publishes a topic.

Instructor notes:

Purpose — Provide an example of a message flow.

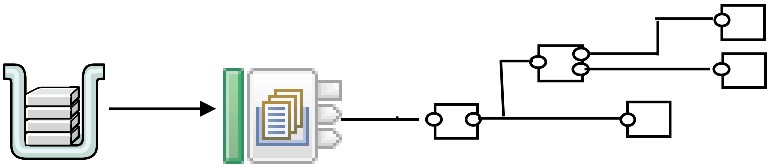
Details — Describe the message flow and emphasize the concepts of input nodes, output nodes, and terminals.

Additional information — None.

Transition statement — The behavior of the flow is ultimately controlled by the configuration of the input node. One of the most common input nodes is the MQInput node.

WebSphere Education

Node example: MQInput



- Polls a local MQ queue for incoming messages that match search criteria
  - Repeated MQGET_with_WAIT operations
  - All GetMessageOptions supplied using the GUI
  - Message can optionally be browsed only
- Default values for parser selection and publish/subscribe topic
- Transaction mode for entire message flow
- Starts its own thread
- Multiple MQInput nodes in a message flow identify alternative input sources
- One message received per thread

Figure 1-27. MQInput node WM643 / VM6431.0

Notes:

The MQInput node retrieves a message from a specified WebSphere MQ message queue.

MQGET is a call function that might appear in a C or COBOL program. It is the way user software can acquire a message from a WebSphere MQ queue.

The MQInput node has three output terminals:

- **Failure** is the output terminal to which the message is routed if an error occurs.
- **Out** is the output terminal to which the message is routed if it is successfully retrieved from the WebSphere MQ queue.
- **Catch** is the output terminal to which the message is routed if an exception is thrown downstream and caught by this node

Node properties determine the node behavior and characteristics, such as the name and location of the WebSphere MQ input queue.

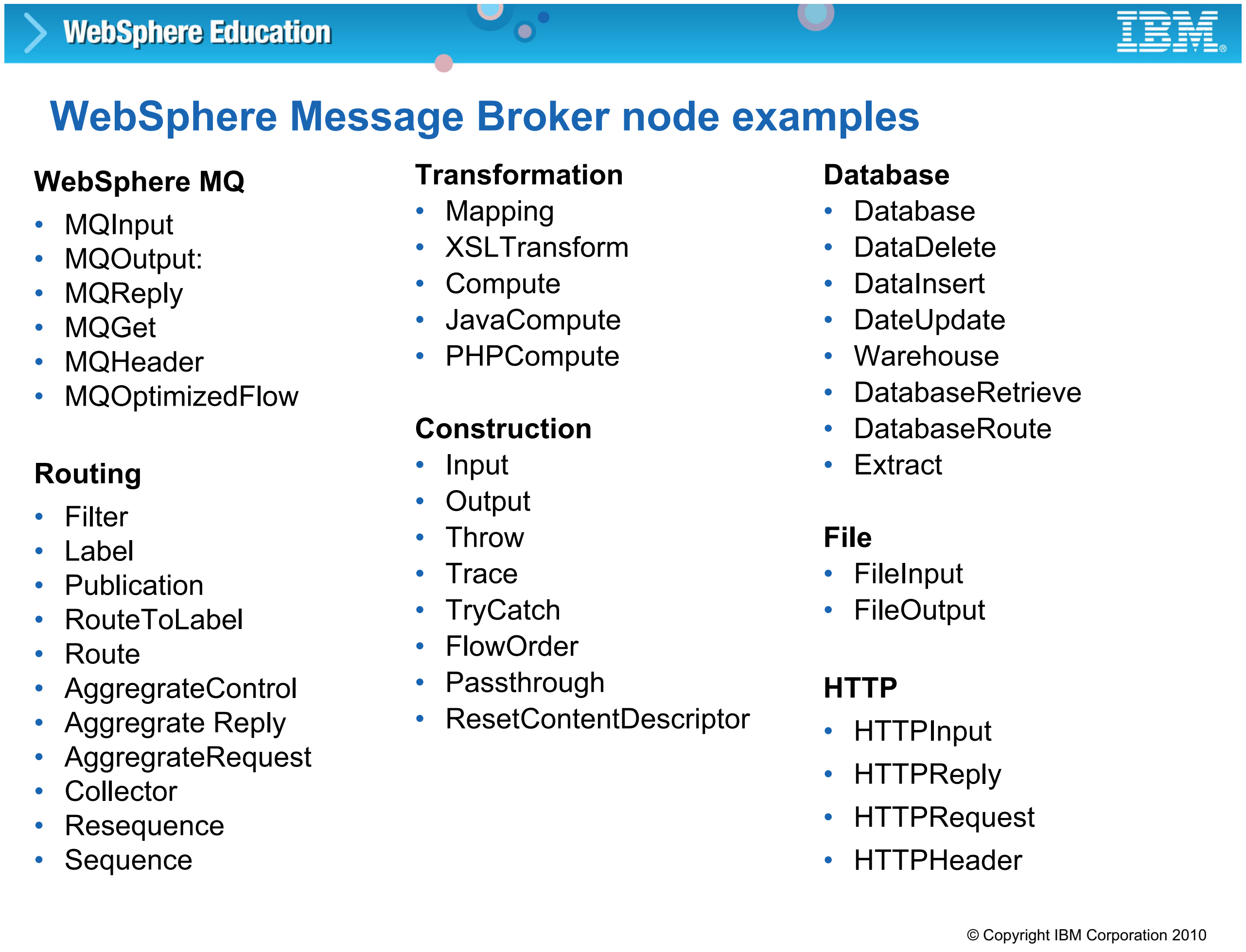
Instructor notes:

Purpose — Introduce the MQInput node as an example of a message flow node.

Details — Default option settings and transaction mode are covered in more detail later. Make sure that students understand how the MQInput node starts and influences the behavior of a message flow.

Additional information — None.

Transition statement — WebSphere Message Broker supports many built-in operations.



WebSphere Education

WebSphere Message Broker node examples

WebSphere MQ

- MQInput
- MQOutput:
- MQReply
- MQGet
- MQHeader
- MQOptimizedFlow

Routing

- Filter
- Label
- Publication
- RouteToLabel
- Route
- AggregateControl
- Aggregate Reply
- AggregateRequest
- Collector
- Resequence
- Sequence

Transformation

- Mapping
- XSLTransform
- Compute
- JavaCompute
- PHPCompute

Construction

- Input
- Output
- Throw
- Trace
- TryCatch
- FlowOrder
- Passthrough
- ResetContentDescriptor

Database

- Database
- DataDelete
- DataInsert
- DateUpdate
- Warehouse
- DatabaseRetrieve
- DatabaseRoute
- Extract

File

- FileInput
- FileOutput

HTTP

- HTTPInput
- HTTPReply
- HTTPRequest
- HTTPHeader

Figure 1-28. WebSphere Message Broker node examples WM643 / VM6431.0

Notes:

The figure is a partial list of the nodes available to the developer during the creation of message flow.

Note

See the WebSphere Message Broker Information Center for detailed information about all the nodes available in WebSphere Message Broker.

Instructor notes:

Purpose — List some of the nodes available in WebSphere Message Broker.

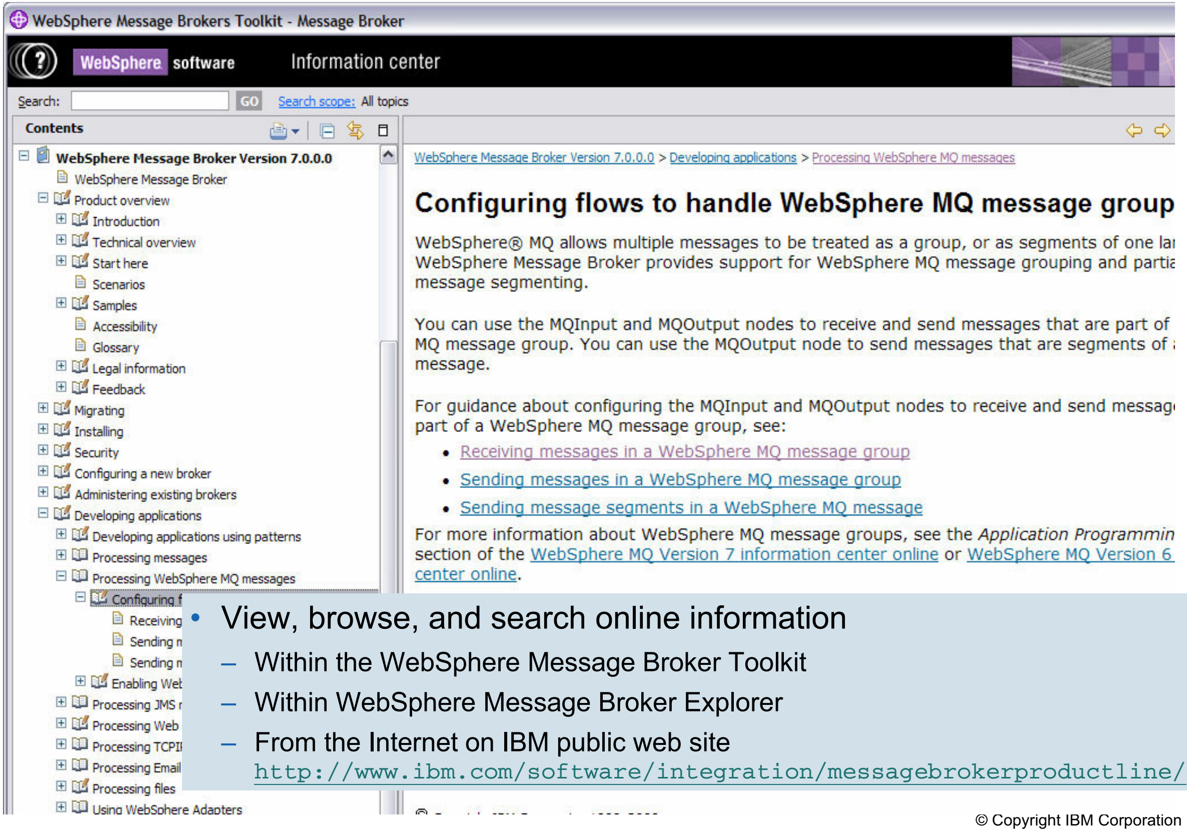
Details — This slide is intended to give administrators a high-level overview of the types of operations that are supported by WebSphere Message Broker. Highlight some of the nodes that are used in the message flows in the exercises for this course such as the MQOutput node and Compute node. Do not go into depth at this point as this is an overview unit.

Additional information — None.

Transition statement — A description of the nodes and node properties is available in the WebSphere Message Broker information center.

WebSphere Education IBM

WebSphere Information Center



WebSphere Message Brokers Toolkit - Message Broker

WebSphere Message Broker Version 7.0.0.0 > Developing applications > Processing WebSphere MQ messages

Configuring flows to handle WebSphere MQ message group

WebSphere® MQ allows multiple messages to be treated as a group, or as segments of one lar WebSphere Message Broker provides support for WebSphere MQ message grouping and partia message segmenting.

You can use the MQInput and MQOutput nodes to receive and send messages that are part of MQ message group. You can use the MQOutput node to send messages that are segments of a message.

For guidance about configuring the MQInput and MQOutput nodes to receive and send messag part of a WebSphere MQ message group, see:

- Receiving messages in a WebSphere MQ message group
- Sending messages in a WebSphere MQ message group
- Sending message segments in a WebSphere MQ message

For more information about WebSphere MQ message groups, see the *Application Programmin* section of the WebSphere MQ Version 7 information center online or WebSphere MQ Version 6 center online.

- View, browse, and search online information
  - Within the WebSphere Message Broker Toolkit
  - Within WebSphere Message Broker Explorer
  - From the Internet on IBM public web site
    `http://www.ibm.com/software/integration/messagebrokerproductline/`

Figure 1-29. WebSphere Information Center WM643 / VM6431.0

Notes:

It is important to know where to go to get additional information about the product and product application. The WebSphere Message Broker Information Center contains a complete set of application documentation. It is available online and with WebSphere Message Broker Toolkit.

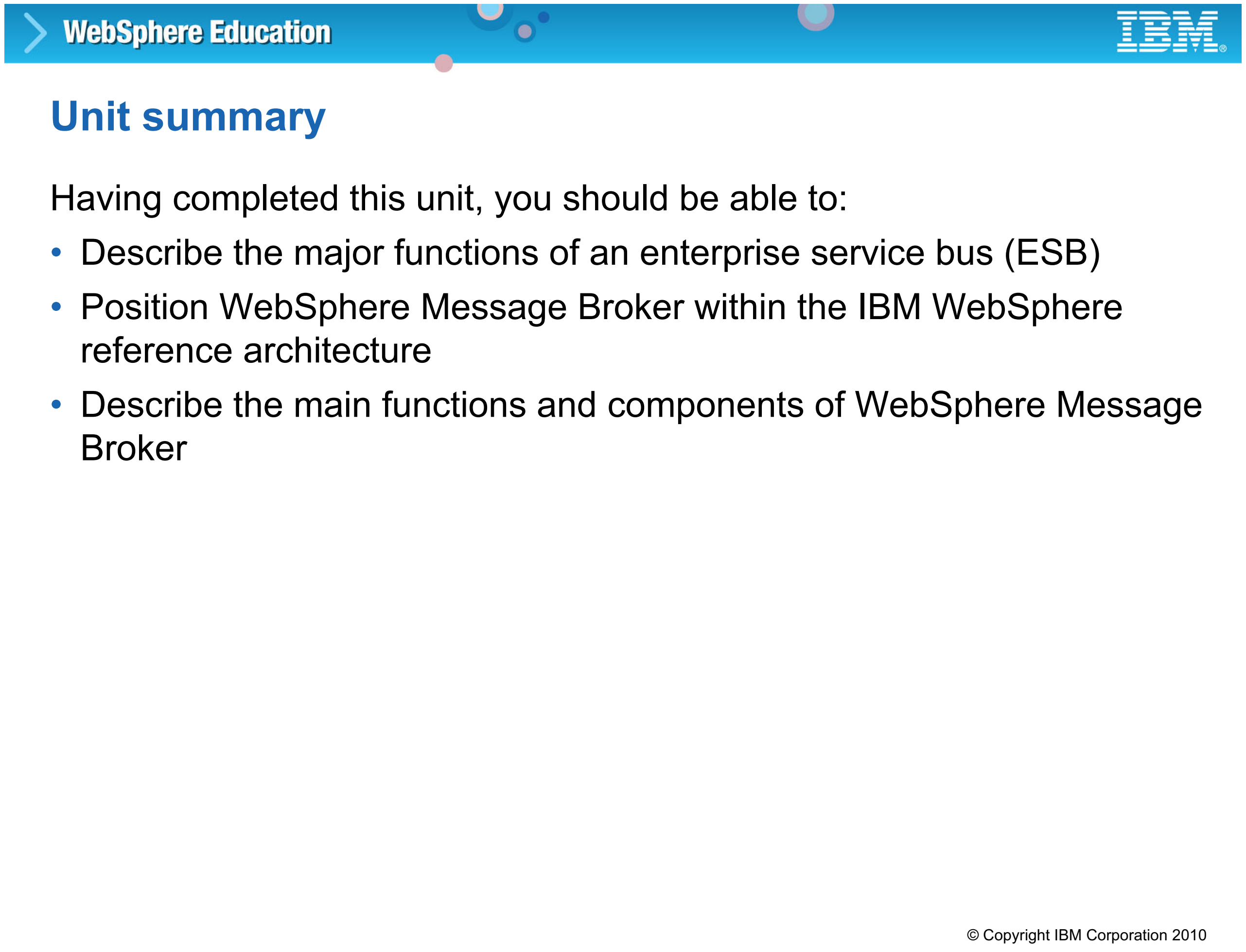
Instructor notes:

Purpose — Introduce the WebSphere Information Center.

Details — It is important for students know where to go to get information about WebSphere Message Broker. Show the students how to access the documentation from the WebSphere Message Broker Toolkit.

Additional information — The Information Center and help functions are described in more detail in the next unit.

Transition statement — What did you learn in this unit?



WebSphere Education

Unit summary

Having completed this unit, you should be able to:

- Describe the major functions of an enterprise service bus (ESB)
- Position WebSphere Message Broker within the IBM WebSphere reference architecture
- Describe the main functions and components of WebSphere Message Broker

Figure 1-30. Unit summary WM643 / VM6431.0

Notes:

Instructor notes:

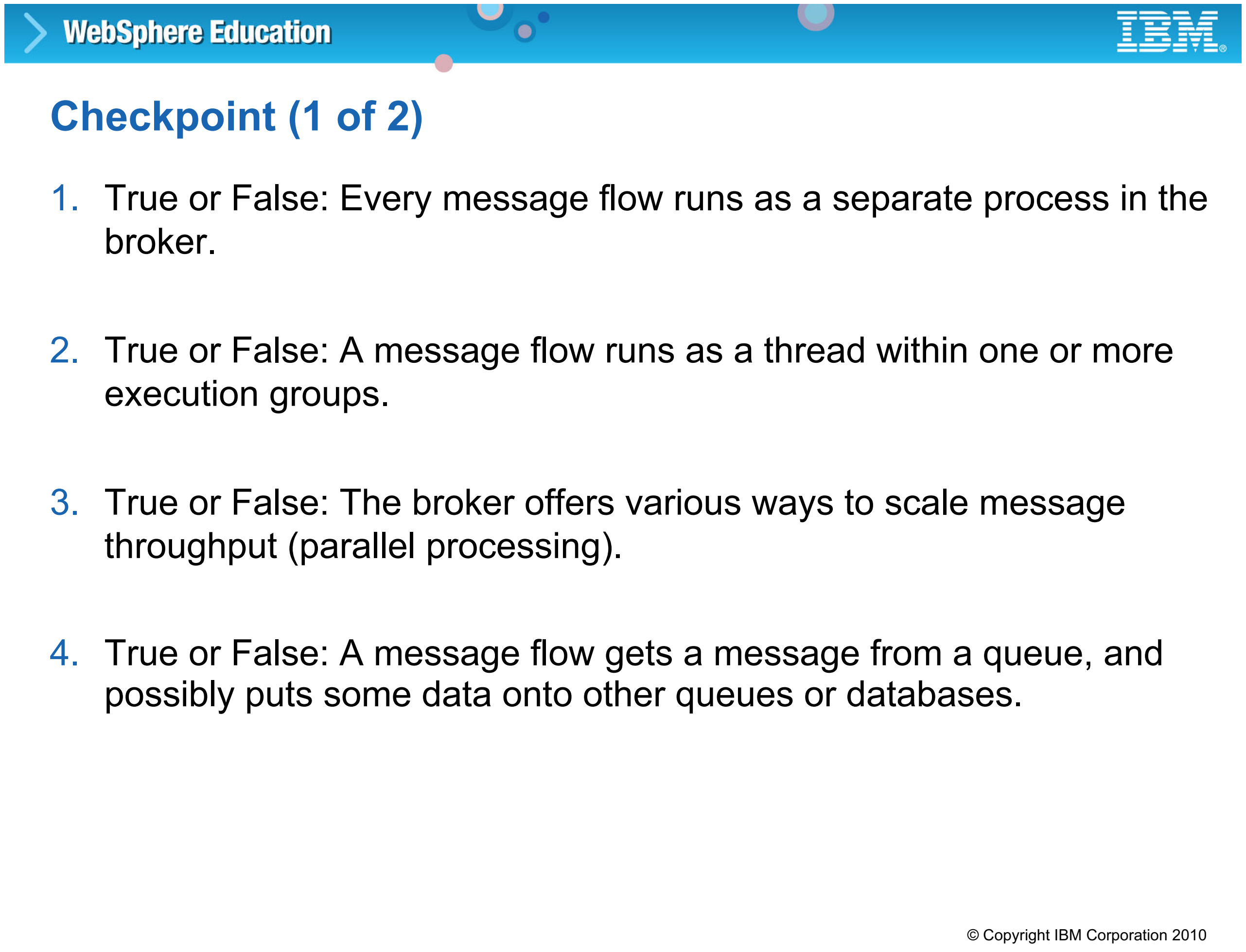
Purpose — Summarize the unit.

Details — This unit should have provided a feel for the products and helped the students understand (at a high level) what the component parts are and how they work together.

Although it might take some time to get through it all, laying this foundation is necessary if the students are to complete the course successfully.

Additional information — None.

Transition statement — In the next unit, you will learn about installation and setup and have an opportunity to work with the product. First, some checkpoints test your knowledge to ensure that you grasp the key concepts presented in this unit.



WebSphere Education
IBM

Checkpoint (1 of 2)

1. True or False: Every message flow runs as a separate process in the broker.

2. True or False: A message flow runs as a thread within one or more execution groups.

3. True or False: The broker offers various ways to scale message throughput (parallel processing).

4. True or False: A message flow gets a message from a queue, and possibly puts some data onto other queues or databases.

Figure 1-31. Checkpoint (1 of 2) WM643 / VM6431.0

Notes:

Each statement requires a true or false answer. Record your answers here:

1.

2.

3.

4.

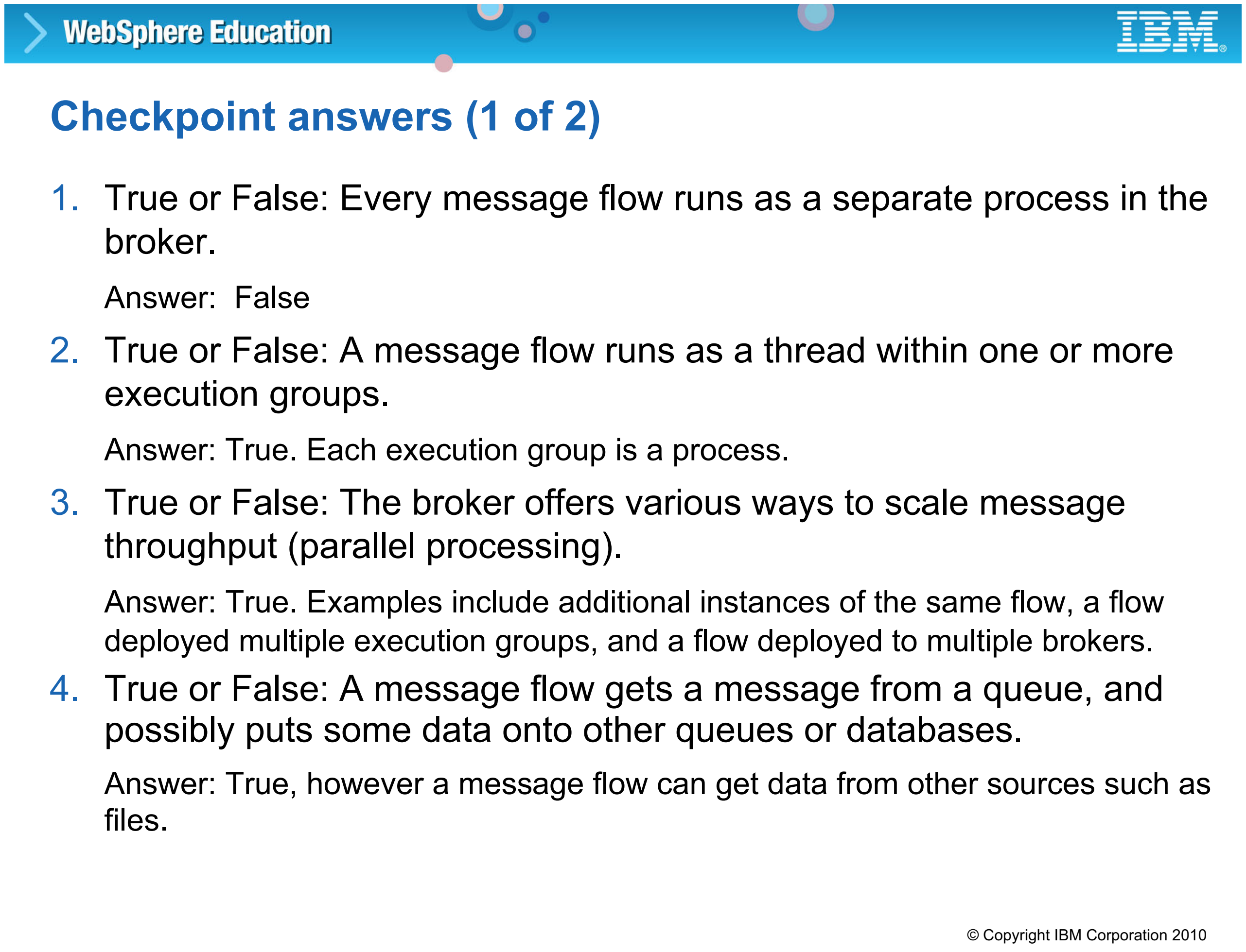
Instructor notes:

Purpose — Test student knowledge.

Details —

Additional information —

Transition statement — Next: Checkpoint answers (1 of 2)



WebSphere Education

Checkpoint answers (1 of 2)

1. True or False: Every message flow runs as a separate process in the broker.

   Answer: False

2. True or False: A message flow runs as a thread within one or more execution groups.

   Answer: True. Each execution group is a process.

3. True or False: The broker offers various ways to scale message throughput (parallel processing).

   Answer: True. Examples include additional instances of the same flow, a flow deployed multiple execution groups, and a flow deployed to multiple brokers.

4. True or False: A message flow gets a message from a queue, and possibly puts some data onto other queues or databases.

   Answer: True, however a message flow can get data from other sources such as files.

Figure 1-32. Checkpoint answers (1 of 2)

WM643 / VM6431.0

Notes:

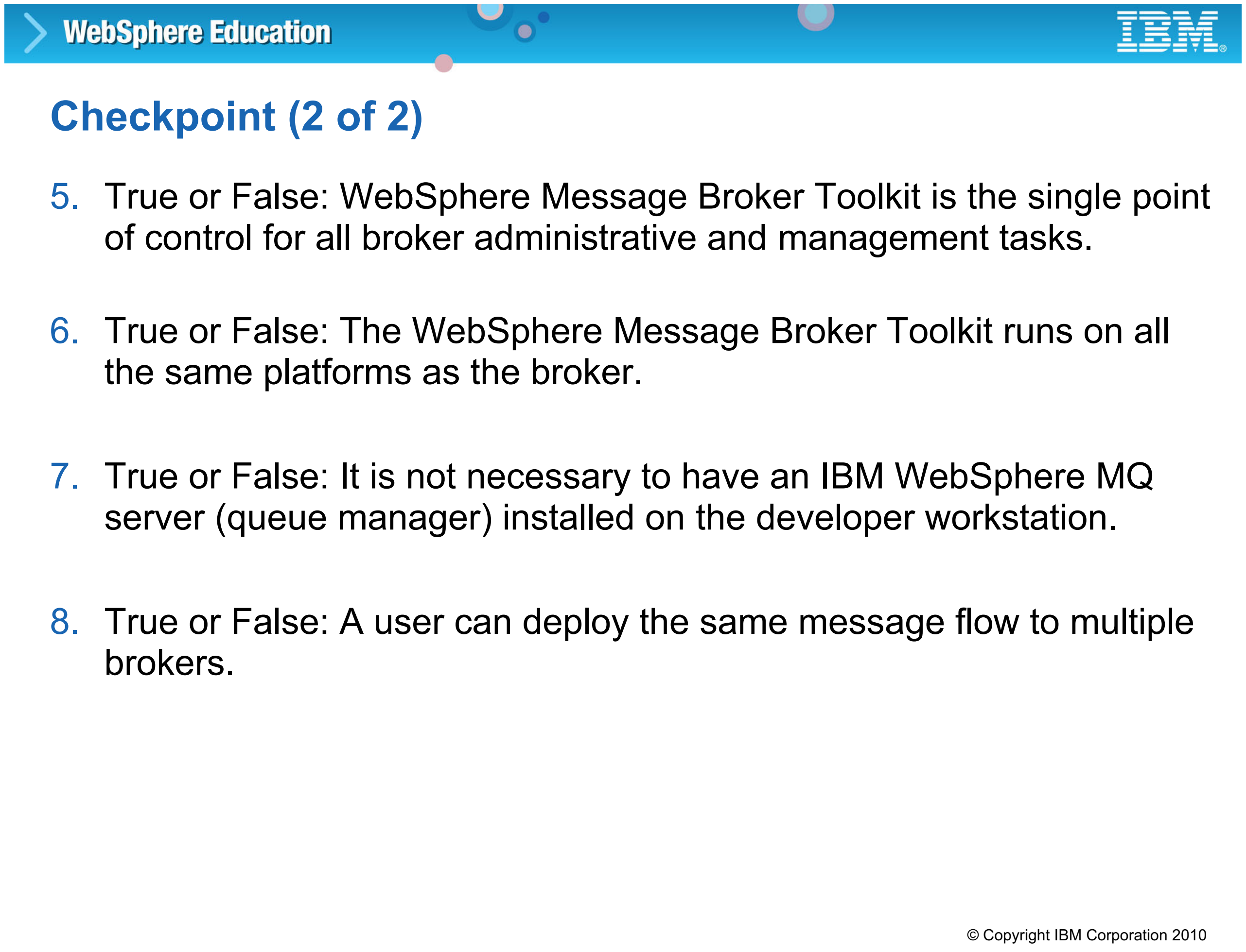
Instructor notes:

Purpose — Checkpoint answers

Details —

Additional information —

Transition statement — Next: Checkpoint (2 of 2)



WebSphere Education

Checkpoint (2 of 2)

5. True or False: WebSphere Message Broker Toolkit is the single point of control for all broker administrative and management tasks.

6. True or False: The WebSphere Message Broker Toolkit runs on all the same platforms as the broker.

7. True or False: It is not necessary to have an IBM WebSphere MQ server (queue manager) installed on the developer workstation.

8. True or False: A user can deploy the same message flow to multiple brokers.

Figure 1-33. Checkpoint (2 of 2) WM643 / VM6431.0

Notes:

Each statement requires a true or false answer. Record your answers here:

5.

6.

7.

8.

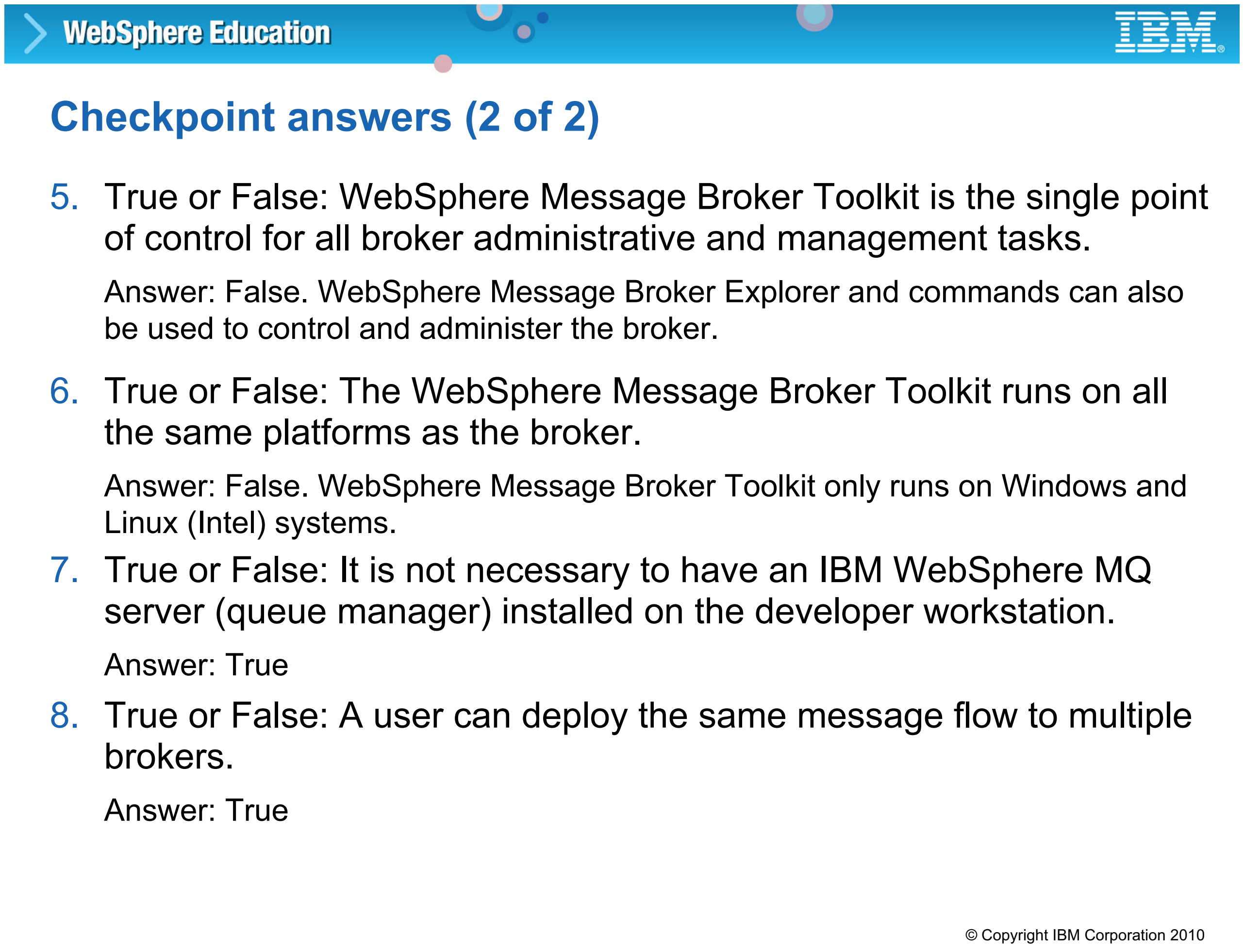
Instructor notes:

Purpose — Test student knowledge.

Details —

Additional information —

Transition statement — Next: Checkpoint answers (2 of 2)



WebSphere Education

Checkpoint answers (2 of 2)

5. True or False: WebSphere Message Broker Toolkit is the single point of control for all broker administrative and management tasks.

   Answer: False. WebSphere Message Broker Explorer and commands can also be used to control and administer the broker.

6. True or False: The WebSphere Message Broker Toolkit runs on all the same platforms as the broker.

   Answer: False. WebSphere Message Broker Toolkit only runs on Windows and Linux (Intel) systems.

7. True or False: It is not necessary to have an IBM WebSphere MQ server (queue manager) installed on the developer workstation.

   Answer: True

8. True or False: A user can deploy the same message flow to multiple brokers.

   Answer: True

Figure 1-34. Checkpoint answers (2 of 2) WM643 / VM6431.0

Notes:

Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —

Unit 2. Product installation, configuration, and security planning

Estimated time

01:00 lecture, 01:00 exercise

What this unit is about

This unit describes the installation preparations and procedures for the WebSphere Message Broker. Post-installation setup is just as important and is covered as well. Last but not least, security considerations and complementary product features are described.

What you should be able to do

After completing this unit, you should be able to:

- List the prerequisite hardware and software for WebSphere Message Broker
- Explain how to install WebSphere Message Broker
- Create queue managers, brokers, and execution groups
- Plan for runtime security

How you will check your progress

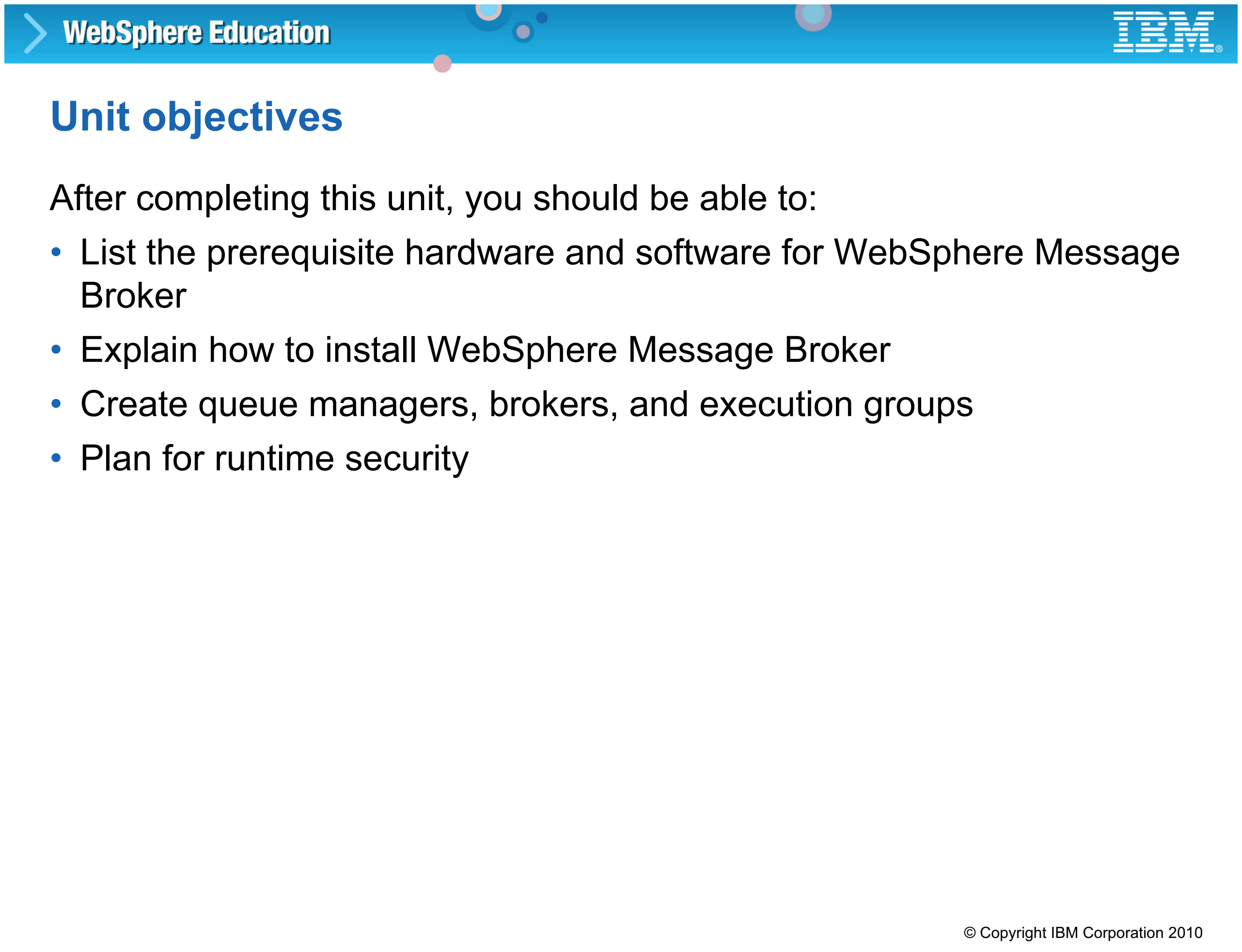
Accountability:

- Review checkpoint questions
- Exercise 1, Setup and customization

References

`http://www.ibm.com/software/integration/`
`wbimessagebroker/library`
IBM WebSphere Message Broker Library

`http://www.ibm.com/software/integration/`
`wbimessagebroker/requirements`
IBM WebSphere Message Broker system requirements



WebSphere Education

Unit objectives

After completing this unit, you should be able to:

- List the prerequisite hardware and software for WebSphere Message Broker
- Explain how to install WebSphere Message Broker
- Create queue managers, brokers, and execution groups
- Plan for runtime security

Figure 2-1. Unit objectives

WM643 / VM6431.0

Notes:

Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —

2.1. Hardware and software requirements

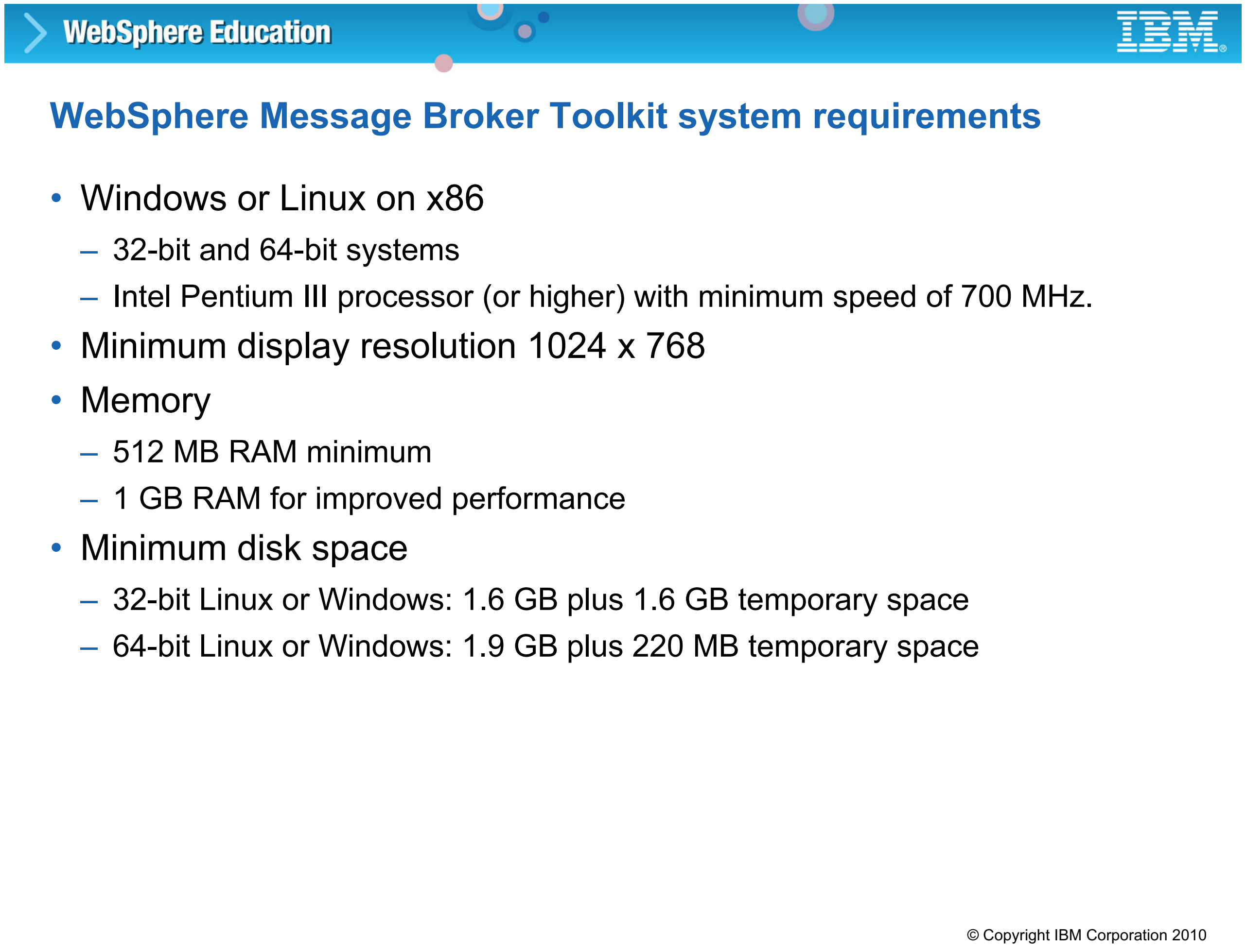
Instructor topic introduction

What students will do — Students will learn about the hardware and software prerequisites for installing WebSphere Message Broker for the Windows, distributed, and zSeries platforms.

How students will do it — Listen to the lecture and participate in classroom discussion.

What students will learn — Students will discover the basic hardware environments required to install WebSphere Message Broker. They will also learn about the operating system and other software requirements needed before beginning installation. The information presented here might be out of date, so it is important that students understand where they can retrieve the most current information (the WebSphere Message Broker website).

How this will help students in their job — Students will use this information to help determine whether their computing environments are ready to support a WebSphere Message Broker runtime environment or the WebSphere Message Broker Toolkit.



WebSphere Education

WebSphere Message Broker Toolkit system requirements

- Windows or Linux on x86
  - 32-bit and 64-bit systems
  - Intel Pentium III processor (or higher) with minimum speed of 700 MHz.
- Minimum display resolution 1024 x 768
- Memory
  - 512 MB RAM minimum
  - 1 GB RAM for improved performance
- Minimum disk space
  - 32-bit Linux or Windows: 1.6 GB plus 1.6 GB temporary space
  - 64-bit Linux or Windows: 1.9 GB plus 220 MB temporary space

Figure 2-2. WebSphere Message Broker Toolkit system requirements WM643 / VM6431.0

Notes:

The figure lists the system hardware requirements for WebSphere Message Broker Toolkit.

The values stated are minimums. You can achieve better performance by increasing the amount of memory and processor speed.

Message definitions and message flows are stored in the local file system. As the number of message definitions and message flows increases, your disk space requirements might increase as well. With careful planning, you can manage your resources without a negative impact to operations.

See the WebSphere Message Broker system requirements website and the release notes for detailed requirements.

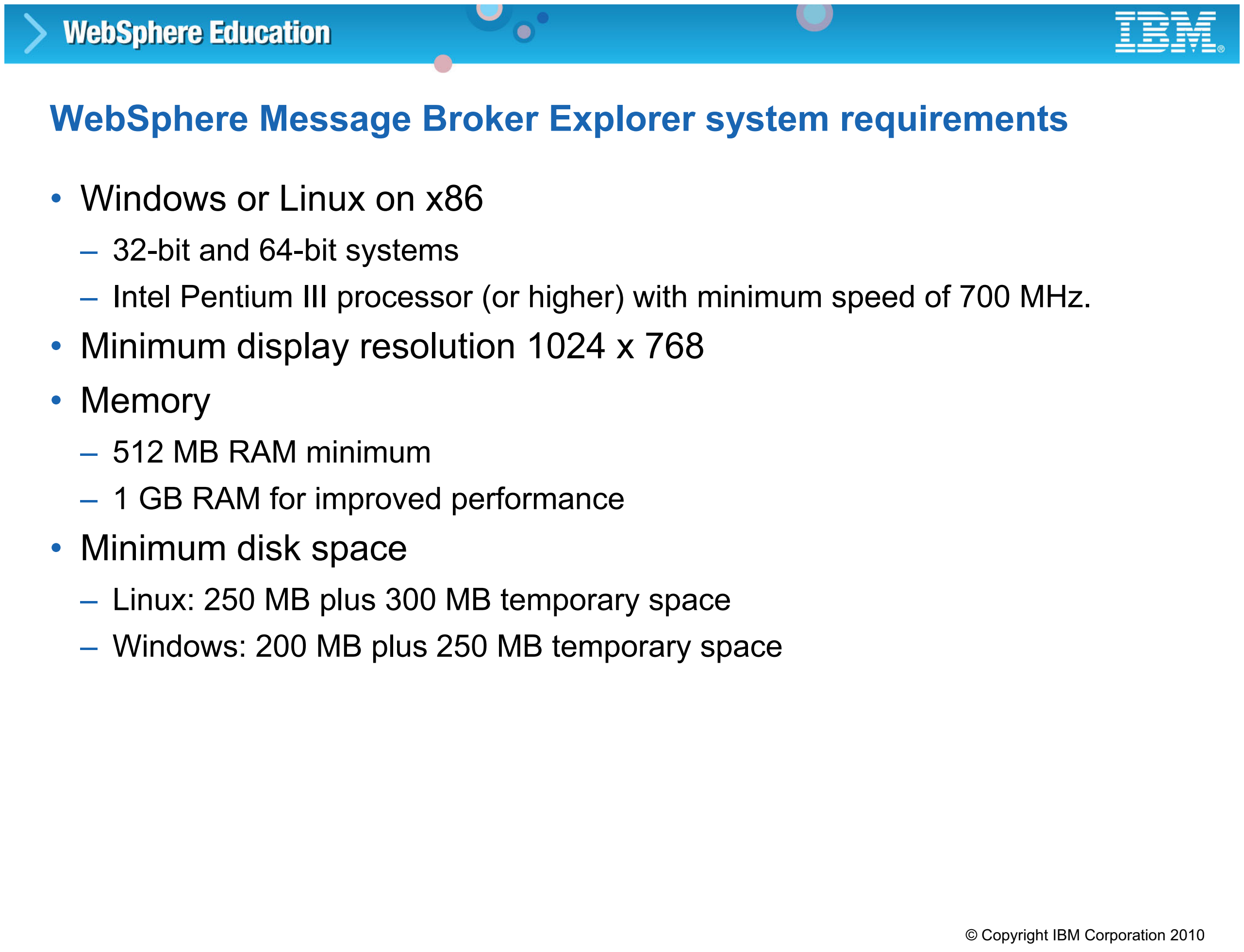
Instructor notes:

Purpose — List the WebSphere Message Broker Toolkit system requirements.

Details — Emphasize to students the importance of checking the online system requirements for the latest information and operating system and hardware details.

Additional information —

Transition statement —



WebSphere Education

WebSphere Message Broker Explorer system requirements

- Windows or Linux on x86
  - 32-bit and 64-bit systems
  - Intel Pentium III processor (or higher) with minimum speed of 700 MHz.
- Minimum display resolution 1024 x 768
- Memory
  - 512 MB RAM minimum
  - 1 GB RAM for improved performance
- Minimum disk space
  - Linux: 250 MB plus 300 MB temporary space
  - Windows: 200 MB plus 250 MB temporary space

Figure 2-3. WebSphere Message Broker Explorer system requirements WM643 / VM6431.0

Notes:

The hardware and software requirements in this figure are extracted from `http://www.ibm.com/software/integration/wbimessagebroker/library`, the *IBM WebSphere Message Broker Library.*

Consult the *IBM WebSphere Message Broker Library* for detailed, current information. You might also want to consult the WebSphere Message Broker system requirements at `http://www.ibm.com/software/integration/wbimessagebroker/requirements.`

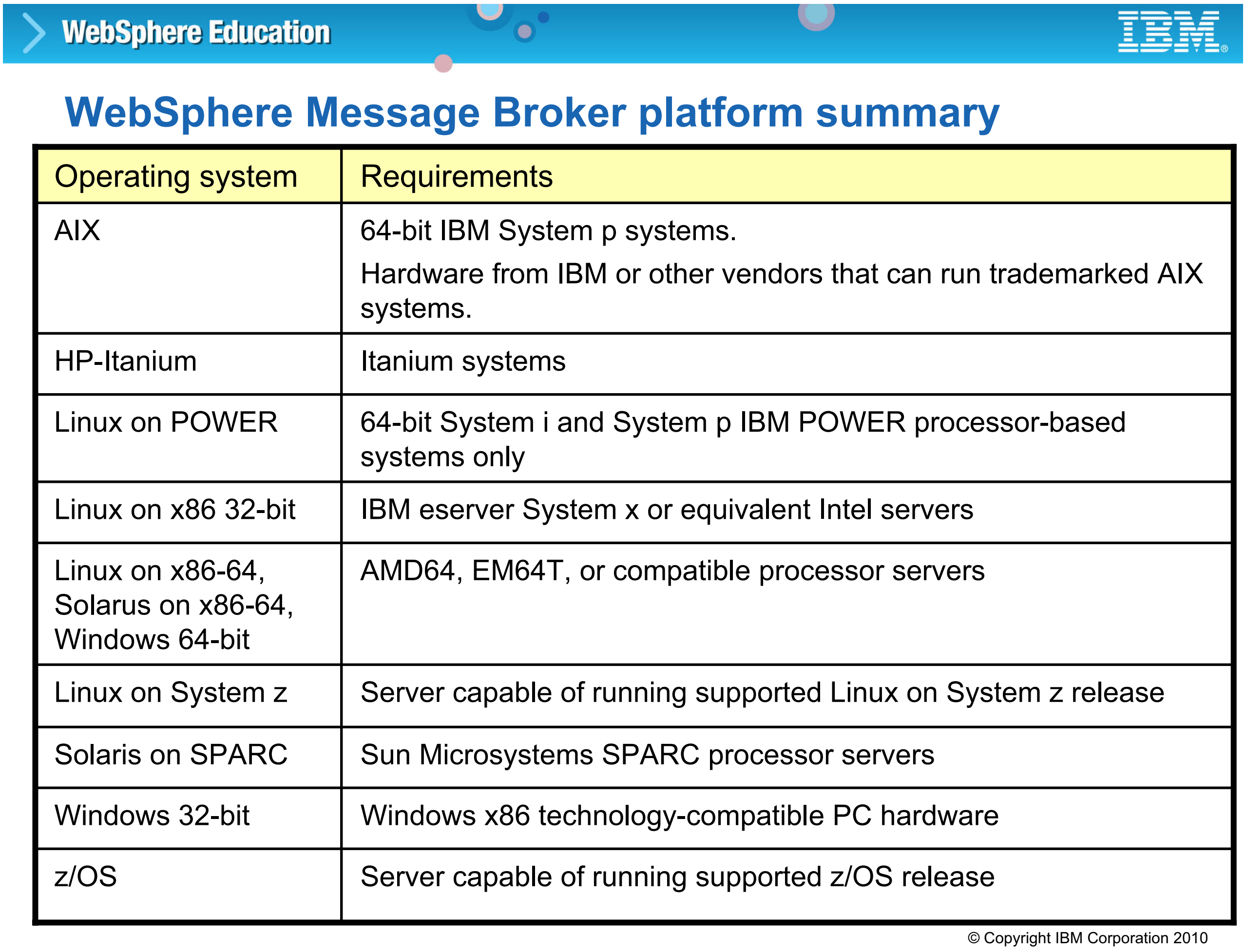
Instructor notes:

Purpose — List the system requirements for WebSphere Message Broker Explorer.

Details — The disk space size does not include the space requirements for WebSphere MQ.

Additional information —

Transition statement —



WebSphere Education

WebSphere Message Broker platform summary

Operating system	Requirements
AIX	64-bit IBM System p systems. Hardware from IBM or other vendors that can run trademarked AIX systems.
HP-Itanium	Itanium systems
Linux on POWER	64-bit System i and System p IBM POWER processor-based systems only
Linux on x86 32-bit	IBM eserver System x or equivalent Intel servers
Linux on x86-64, Solarus on x86-64, Windows 64-bit	AMD64, EM64T, or compatible processor servers
Linux on System z	Server capable of running supported Linux on System z release
Solaris on SPARC	Sun Microsystems SPARC processor servers
Windows 32-bit	Windows x86 technology-compatible PC hardware
z/OS	Server capable of running supported z/OS release

Figure 2-4. WebSphere Message Broker platform summary WM643 / VM6431.0

Notes:

The hardware and software requirements that are shown in this figure were extracted from the WebSphere Message Broker system requirements website at `http://www.ibm.com/software/integration/wbimessagebroker/requirements`. Always consult this website for the latest information, and platform details.

Instructor notes:

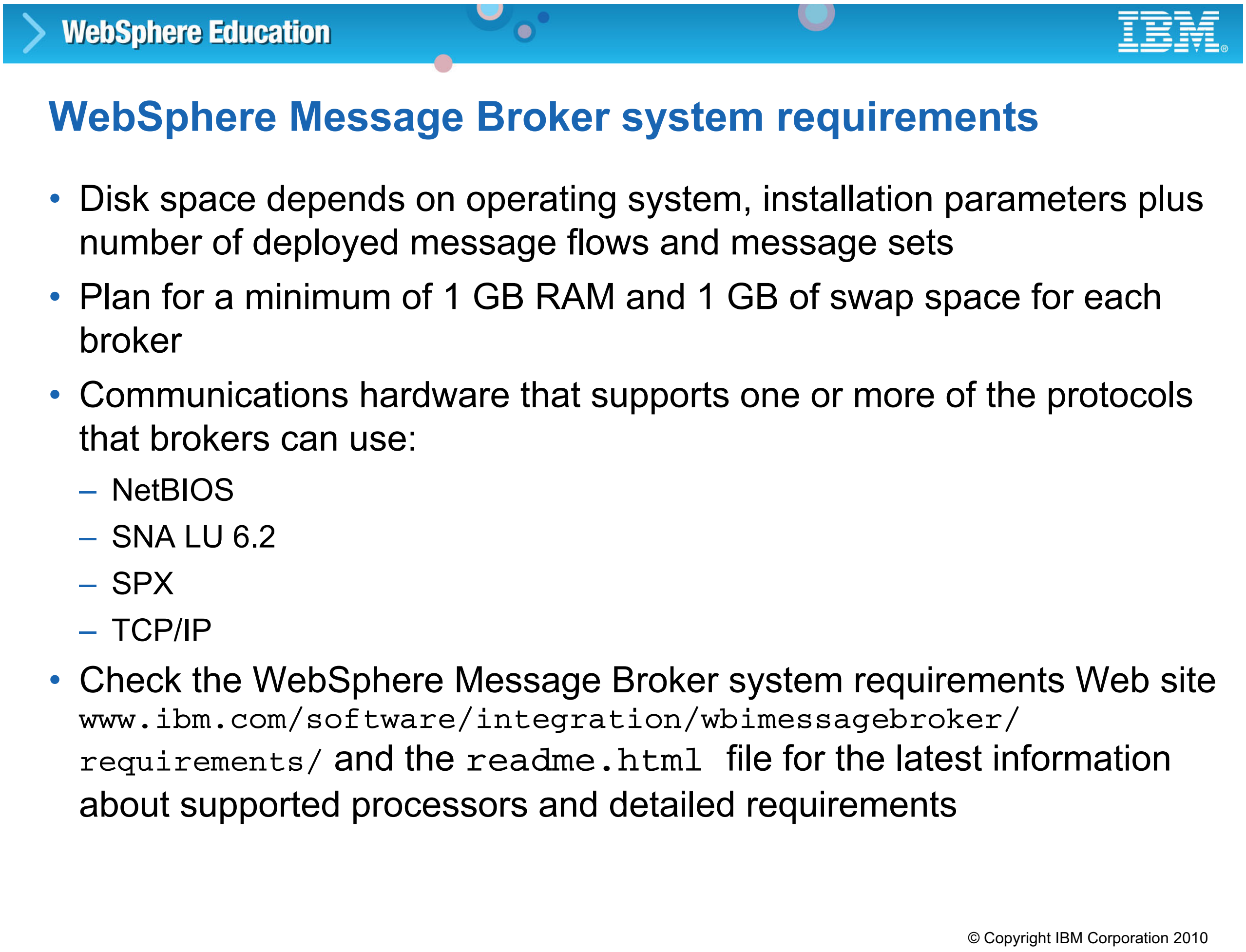
Purpose — Discuss supported platforms for the broker.

Details — Supported platforms can change.

Show the students how to find the platform information for the version of WebSphere Message Broker they are using on the IBM website at `http://www.ibm.com/software/integration/wbimessagebroker/requirements/`.

Additional information — This figure provides an overview of the supported platforms for the broker.

Transition statement — Next are the requirements for the broker.



WebSphere Education

WebSphere Message Broker system requirements

- Disk space depends on operating system, installation parameters plus number of deployed message flows and message sets
- Plan for a minimum of 1 GB RAM and 1 GB of swap space for each broker
- Communications hardware that supports one or more of the protocols that brokers can use:
  - NetBIOS
  - SNA LU 6.2
  - SPX
  - TCP/IP
- Check the WebSphere Message Broker system requirements Web site `www.ibm.com/software/integration/wbimessagebroker/requirements/` and the `readme.html` file for the latest information about supported processors and detailed requirements

Figure 2-5. WebSphere Message Broker system requirements WM643 / VM6431.0

Notes:

The hardware listed is the **minimum**. Your installation might require more capacity, depending on other applications running on your system and how much memory you have installed. In general, more memory provides better performance.

Disk speed is a factor for I/O-intensive message flows.

Recommendations, along with monitoring and measuring techniques, are available in a series of platform-specific, performance-oriented SupportPacs accessible at `http://www.ibm.com/software/integration/wbimessagebroker/support`

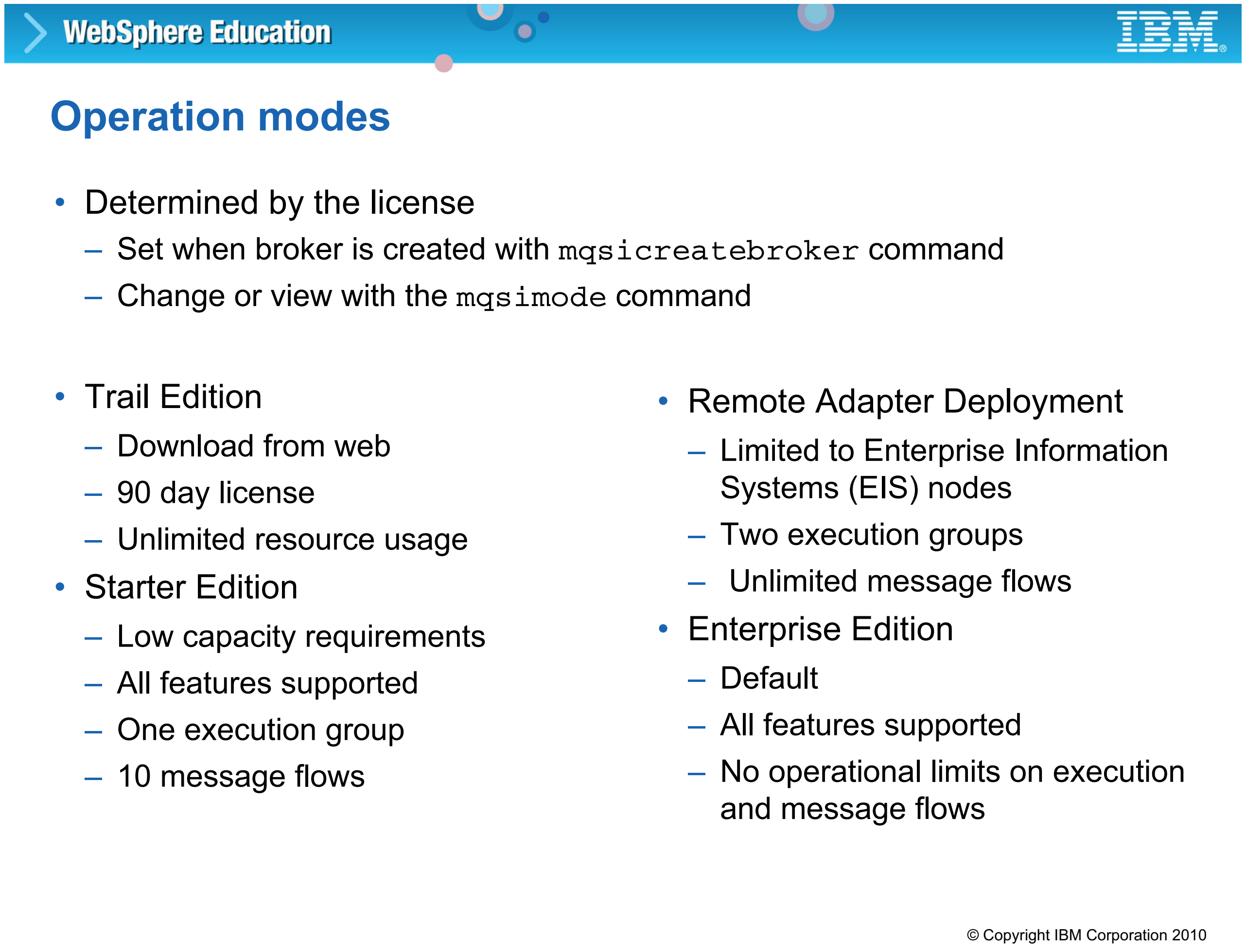
Instructor notes:

Purpose — Continue describing minimum recommended system requirements.

Details — None.

Additional information — Promote the SupportPac reference. A wealth of information is available.

Transition statement — WebSphere Message Broker supports four operation modes.



WebSphere Education

Operation modes

- Determined by the license
  - Set when broker is created with `mqsicreatebroker` command
  - Change or view with the `mqsimode` command
- Trail Edition
  - Download from web
  - 90 day license
  - Unlimited resource usage
- Starter Edition
  - Low capacity requirements
  - All features supported
  - One execution group
  - 10 message flows
- Remote Adapter Deployment
  - Limited to Enterprise Information Systems (EIS) nodes
  - Two execution groups
  - Unlimited message flows
- Enterprise Edition
  - Default
  - All features supported
  - No operational limits on execution and message flows

Figure 2-6. Operation modes

WM643 / VM6431.0

Notes:

The operation mode that you use for your broker is determined by the license that you purchase.

The following modes are supported:

- Trial Edition mode. All features are enabled, but you can use the product for only 90 days after installation.
- Starter Edition mode. All features are enabled, but the number of execution groups that you can create, and the number of message flows that you can deploy, are limited.
- Remote Adapter Deployment mode. Only adapter-related features are enabled, and the types of node that you can use, and the number of execution groups that you can create, are limited.
- Enterprise mode. All features are enabled and no restrictions or limits are imposed. This mode is the default mode, unless you have the Trial Edition.

You must ensure that your brokers are running in the correct operation mode. You can set the operation mode when you create a broker by using the `mqsicreatebroker` command.

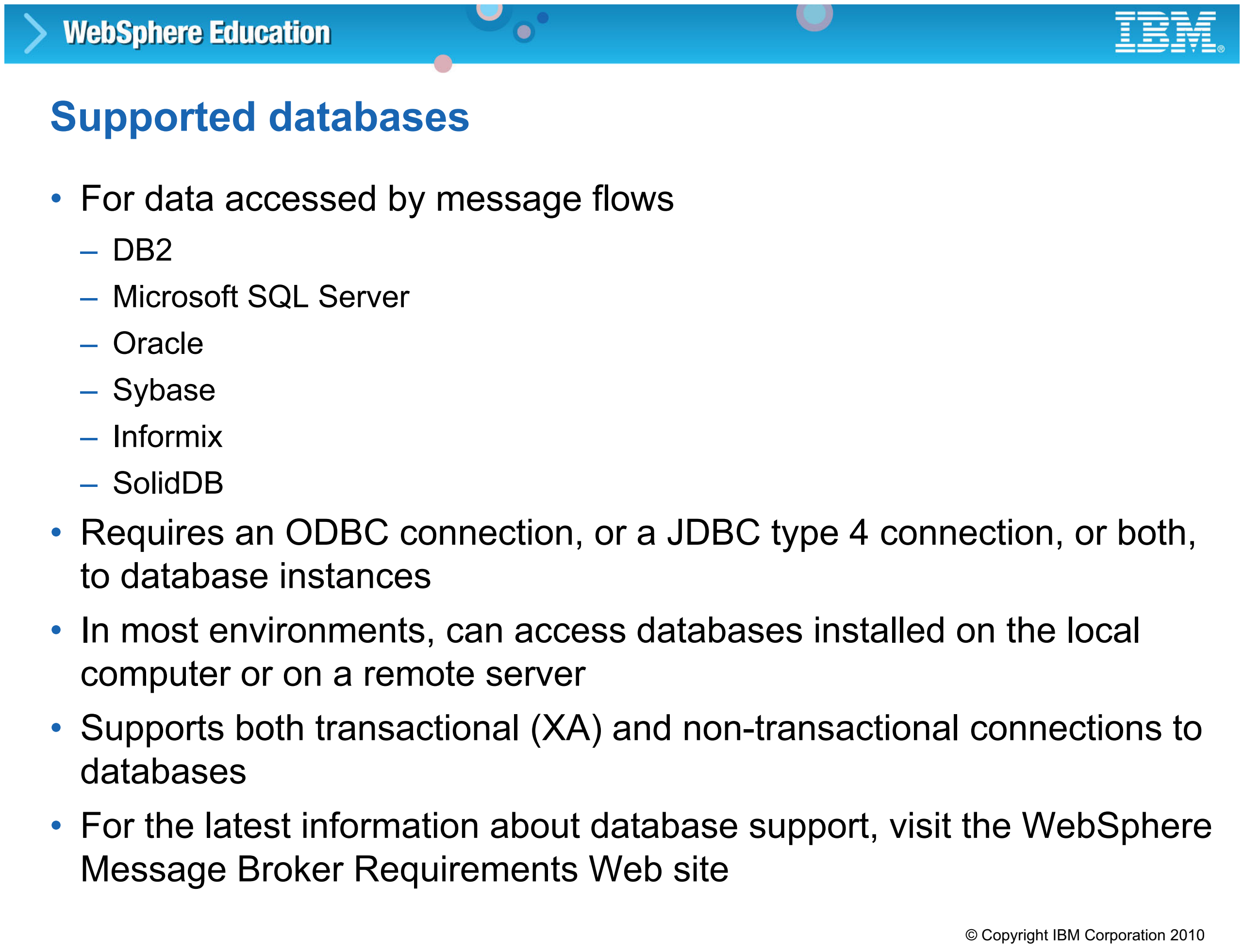
Instructor notes:

Purpose — Provide a description of the WebSphere Message Broker operation modes.

Details —

Additional information — Use the `mqsimode` command to configure and retrieve operation mode information. For example, the check the mode, enter: `mqsimode <`*`brokerName`*`>`.

Transition statement — WebSphere Message Broker also supports many of the popular relational database management systems.



WebSphere Education

Supported databases

- For data accessed by message flows
  - DB2
  - Microsoft SQL Server
  - Oracle
  - Sybase
  - Informix
  - SolidDB
- Requires an ODBC connection, or a JDBC type 4 connection, or both, to database instances
- In most environments, can access databases installed on the local computer or on a remote server
- Supports both transactional (XA) and non-transactional connections to databases
- For the latest information about database support, visit the WebSphere Message Broker Requirements Web site

Figure 2-7. Supported databases WM643 / VM6431.0

Notes:

WebSphere Message Broker message flows can be configured to access databases at run time provided there is an ODBC or JDBC type 4 connection.

Supported releases of DB2 UDB, Oracle, Sybase, and Informix can participate as a resource manager in a distributed XA transaction, and can be coordinated by WebSphere MQ as the XA transaction manager. This can be an important consideration depending on how the application developers are using WebSphere Message Broker.

The information that is shown in this figure indicates the support for databases valid when the WebSphere Message Broker Information Center was published. However, database support might be enhanced after the release is made available. For the latest information about database support, visit the WebSphere Message Broker Requirements website.

Instructor notes:

Purpose — The database environment is an important consideration for a WebSphere Message Broker installation.

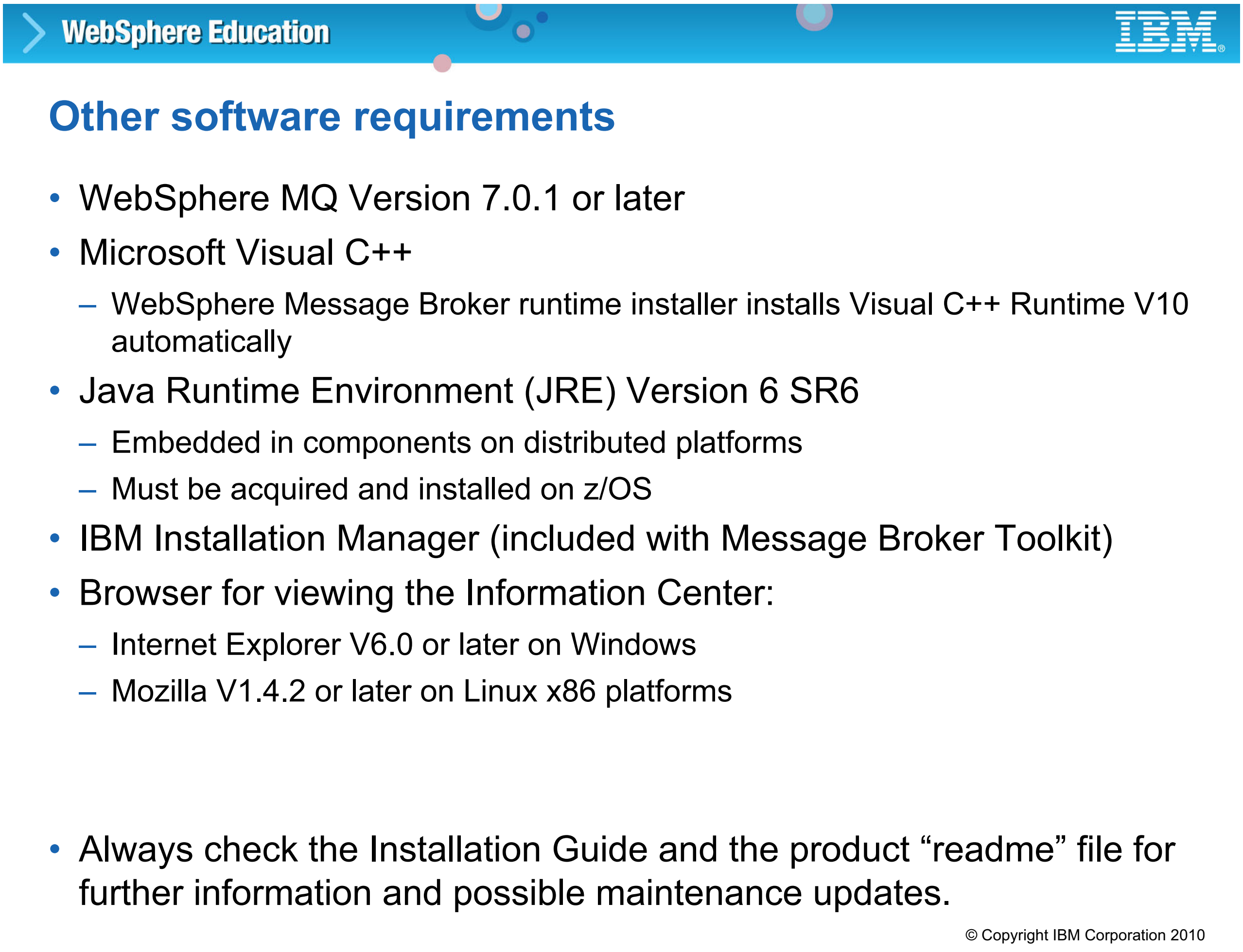
Details — As of WebSphere Message Broker V7, a database is no longer required to maintain state information about the broker environment as it is operating. Database connectivity is still supported for application developers who need to access databases to retrieve or store data as part of a message flow.

If students ask about ODBC or JDBC support for a database, open the WebSphere Message Broker Information Center and search for ‘Supported databases’. The table lists all the databases and supported platforms. Emphasize that they should always check the IBM website for the latest information.

Database connectivity is described in more detail in a later unit in this course.

Additional information — None.

Transition statement — Now have a look at the other software that is required for the WebSphere Message Broker environment.



WebSphere Education

Other software requirements

- WebSphere MQ Version 7.0.1 or later
- Microsoft Visual C++
  - WebSphere Message Broker runtime installer installs Visual C++ Runtime V10 automatically
- Java Runtime Environment (JRE) Version 6 SR6
  - Embedded in components on distributed platforms
  - Must be acquired and installed on z/OS
- IBM Installation Manager (included with Message Broker Toolkit)
- Browser for viewing the Information Center:
  - Internet Explorer V6.0 or later on Windows
  - Mozilla V1.4.2 or later on Linux x86 platforms
- Always check the Installation Guide and the product “readme” file for further information and possible maintenance updates.

Figure 2-8. Other software requirements WM643 / VM6431.0

Notes:

Other software products are required for the WebSphere Message Broker environment. Consult the Installation Guide and product “readme file” for more detailed information.

Some products are dependent upon the operating system being used. For example, WebSphere Message Broker for zSeries uses various other services, such as z/OS UNIX System Services (USS), Language Environment (LE), automatic restart manager (ARM), Resource Recovery Services (RRS), and the syslog daemon (SYSLOGD).

To help with the installation prerequisites, the installation Launchpad tool determines whether certain software products are already installed, and assists with installing them if they are not already present.

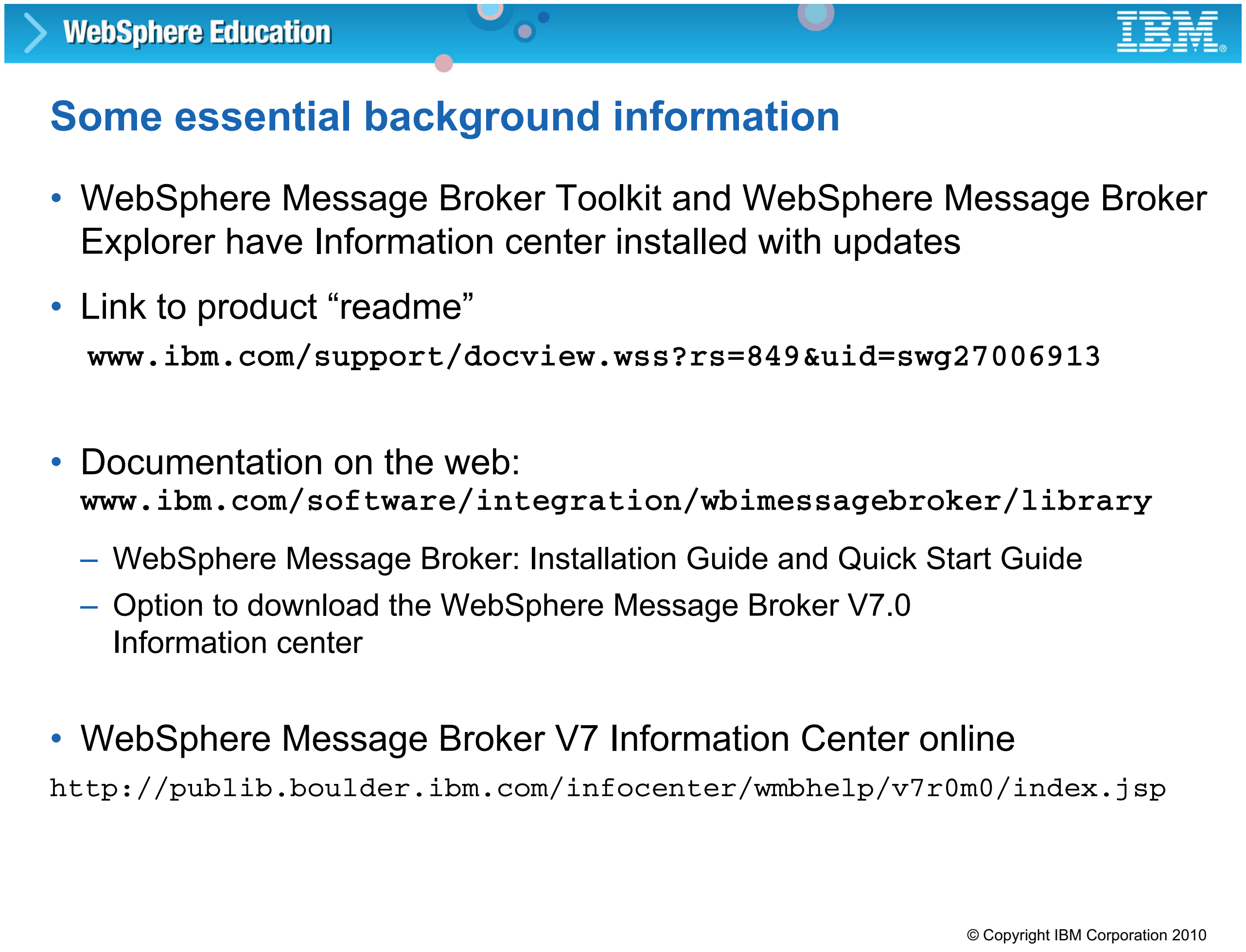
Instructor notes:

Purpose — Highlight software prerequisites and dependencies.

Details — None.

Additional information — Emphasize the importance of reviewing the website for the most recent software prerequisite information.

Transition statement — Some valuable websites of interest are next.



WebSphere Education
IBM

Some essential background information

- WebSphere Message Broker Toolkit and WebSphere Message Broker Explorer have Information center installed with updates
- Link to product “readme”

  `www.ibm.com/support/docview.wss?rs=849&uid=swg27006913`

- Documentation on the web:
  `www.ibm.com/software/integration/wbimessagebroker/library`
  - WebSphere Message Broker: Installation Guide and Quick Start Guide
  - Option to download the WebSphere Message Broker V7.0 Information center

- WebSphere Message Broker V7 Information Center online

`http://publib.boulder.ibm.com/infocenter/wmbhelp/v7r0m0/index.jsp`

Figure 2-9. Some essential background information

WM643 / VM6431.0

Notes:

WebSphere Message Broker is constantly being improved. You should regularly monitor changes by checking the websites identified in the figure.

The documentation can be installed on a Windows or Linux server, independent of the WebSphere Message Broker Toolkit.

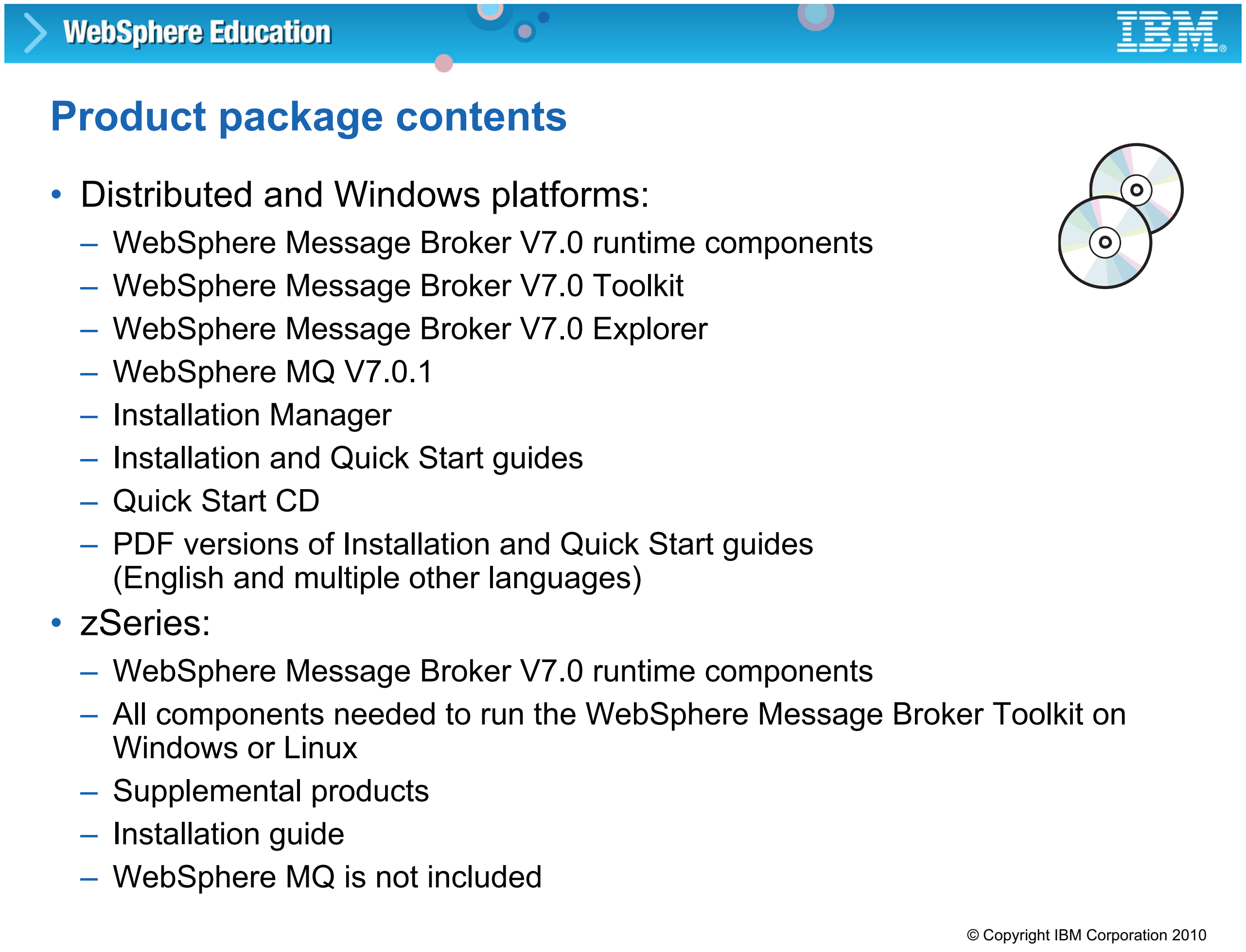
Instructor notes:

Purpose — Alert students to the web locations of key product information.

Details — None.

Additional information — None.

Transition statement — What do you get when you open the box?



WebSphere Education

Product package contents

- Distributed and Windows platforms:
  - WebSphere Message Broker V7.0 runtime components
  - WebSphere Message Broker V7.0 Toolkit
  - WebSphere Message Broker V7.0 Explorer
  - WebSphere MQ V7.0.1
  - Installation Manager
  - Installation and Quick Start guides
  - Quick Start CD
  - PDF versions of Installation and Quick Start guides (English and multiple other languages)
- zSeries:
  - WebSphere Message Broker V7.0 runtime components
  - All components needed to run the WebSphere Message Broker Toolkit on Windows or Linux
  - Supplemental products
  - Installation guide
  - WebSphere MQ is not included

Figure 2-10. Product package contents

WM643 / VM6431.0

Notes:

The products included with WebSphere Message Broker are licensed for use only within the scope of the WebSphere Message Broker products. This means that no other use of these products is permitted.

Always read the accompanying “readme file.”

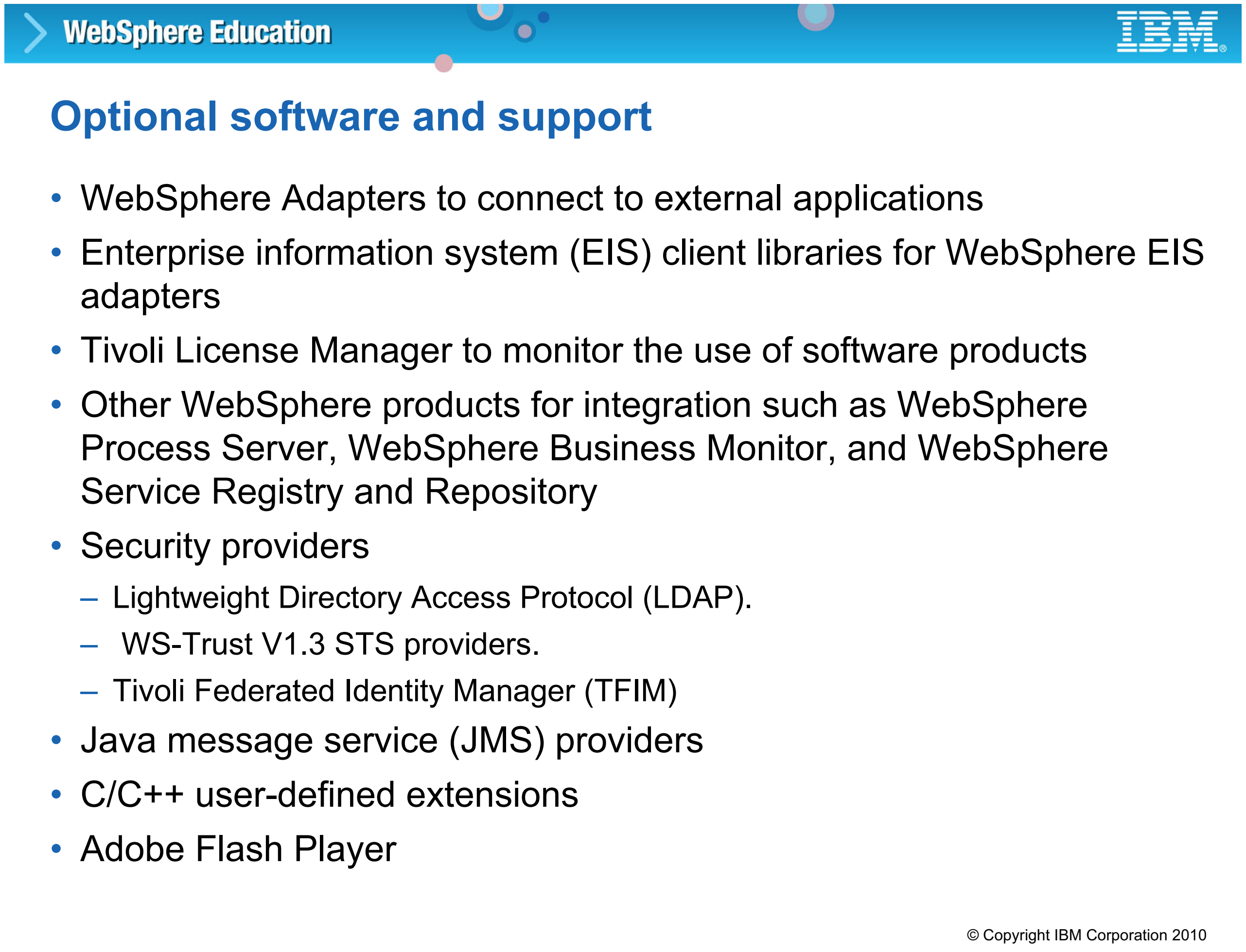
Instructor notes:

Purpose — Discuss package contents.

Details — None.

Additional information — Except where zSeries is identified, the information generally applies to both the Windows and UNIX platforms.

Transition statement — What else is included in the installation media carton?



WebSphere Education

Optional software and support

- WebSphere Adapters to connect to external applications
- Enterprise information system (EIS) client libraries for WebSphere EIS adapters
- Tivoli License Manager to monitor the use of software products
- Other WebSphere products for integration such as WebSphere Process Server, WebSphere Business Monitor, and WebSphere Service Registry and Repository
- Security providers
  - Lightweight Directory Access Protocol (LDAP).
  - WS-Trust V1.3 STS providers.
  - Tivoli Federated Identity Manager (TFIM)
- Java message service (JMS) providers
- C/C++ user-defined extensions
- Adobe Flash Player

Figure 2-11. Optional software and support WM643 / VM6431.0

Notes:

The products listed in the figure are not required, but might be useful. Except where stated, these products are not supplied with WebSphere Message Broker.

WebSphere Adapters are required to connect to external applications. You must obtain the appropriate EIS client libraries from the EIS vendor. For details of the versions of WebSphere Adapters supported by WebSphere Message Broker Version 7.0, visit the WebSphere Message Broker Requirements website.

WebSphere Message Broker SCA nodes allow interoperability with WebSphere Process Server, a business integration server that supports solutions that are based on SOA.

WebSphere Service Registry and Repository nodes allow a message flow to retrieve data dynamically from the repository at run time, and to use and expose those resources in the message flow.

JMS providers are required if any message flows use the JMSInput and JMSOutput nodes.

Adobe Flash Player is required to run the Quick Tour from the Information Center.

Instructor notes:

Purpose — List some of the other software applications that might be useful.

Details — C/C++ user-defined extensions that were built with older levels of compilers can be used when migrating an existing broker or setting up a new broker.

Additional information —

Transition statement — What about the actual installation procedure?

2.2. Installation and configuration

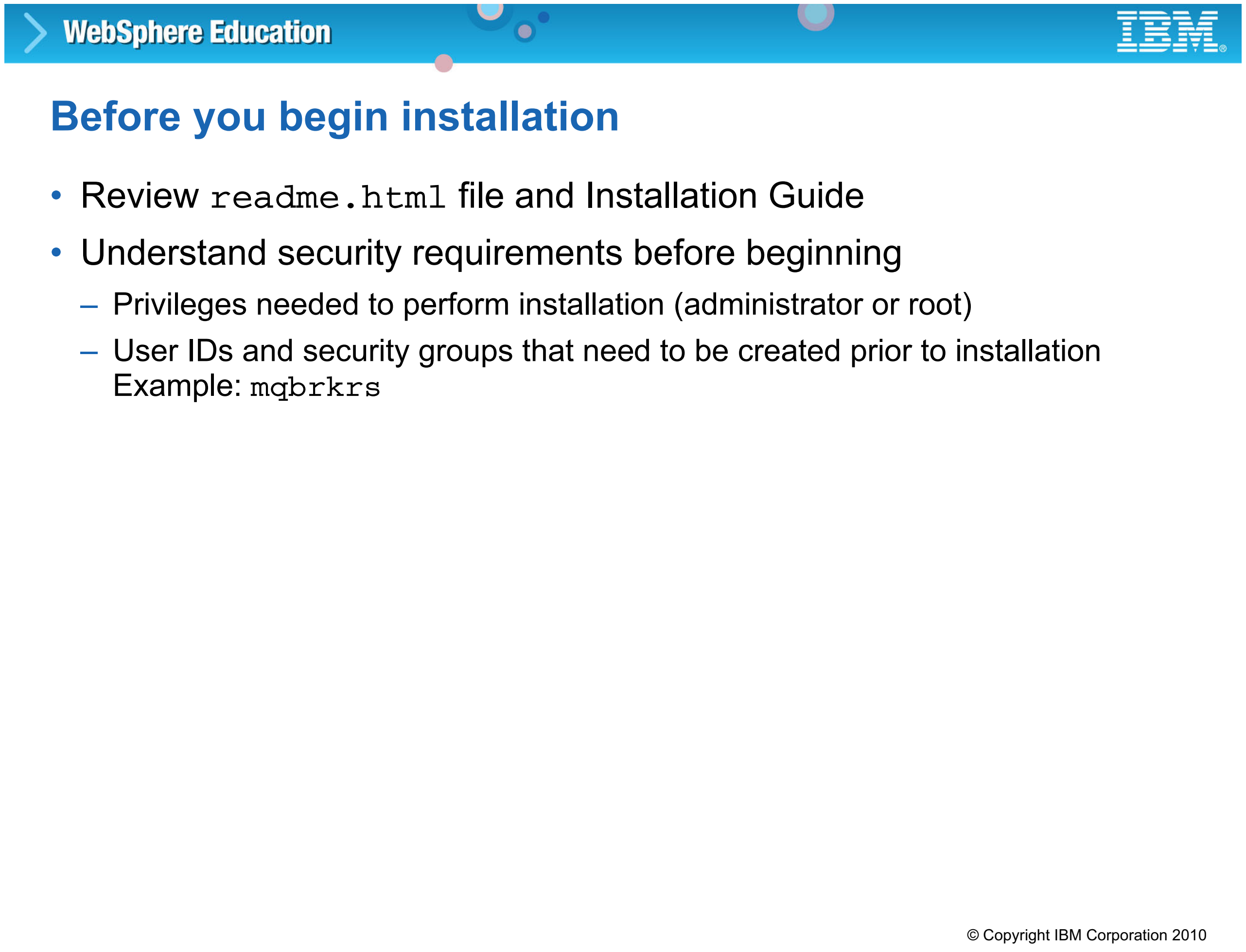
Instructor topic introduction

What students will do — Students will learn about installing and configuring WebSphere Message Broker on Windows and UNIX.

How students will do it — Listen to the lecture and participate in classroom discussion. In the accompanying exercise (exercise 1), assuming an Internet connection in the classroom, students apply the latest fix pack (also known as CSD) and partially configure a preinstalled WebSphere Message Broker on their classroom workstations. Additional configuration is performed in Exercise 2.

What students will learn — Students are exposed to the installation process, which includes WebSphere Message Broker, WebSphere Message Broker Toolkit, and WebSphere Message Broker Explorer.

How this will help students in their job — Students are able to complete the installation and setup of the product in a Windows environment.



WebSphere Education

Before you begin installation

- Review `readme.html` file and Installation Guide
- Understand security requirements before beginning
  - Privileges needed to perform installation (administrator or root)
  - User IDs and security groups that need to be created prior to installation
    Example: `mqbrkrs`

Figure 2-12. Before you begin installation WM643 / VM6431.0

Notes:

Ensure that your system meets the hardware and software prerequisites before you attempt to install any WebSphere Message Broker component.

Security control of WebSphere Message Broker components, resources, and tasks depends on the definition of users and groups of users (principals) to the security subsystem of the operating system. Check that you have the correct authority, and that the required principals are in place, before you install WebSphere Message Broker.

There are platform-specific restrictions on IDs for users and services:

- Windows IDs: 12-character maximum
- Linux, UNIX, zSeries iDs: eight character maximum
- Mixed-environment IDs: eight character maximum
- Ensure that case (uppercase or lowercase) is consistent. Some platforms consider “User1” and “USER1” to be the same ID, others do not.

The Installation Guide describes the security tasks that you must complete before, during, and after installation for the supported platforms.

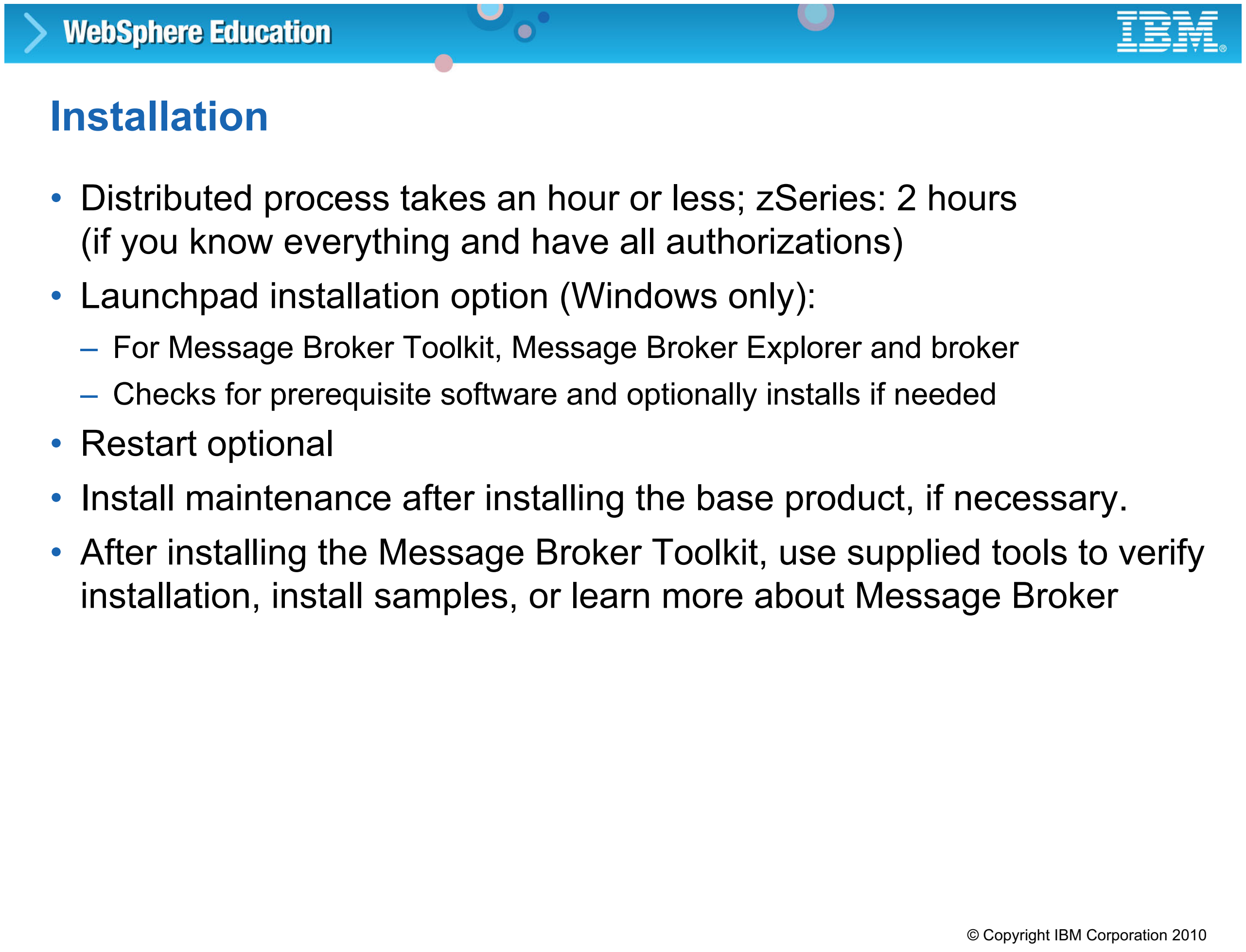
Instructor notes:

Purpose — Discuss the installation prerequisites.

Details — Stress that the students must look at the readme file ***before*** attempting an installation. It contains some critical information.

Additional information —

Transition statement —



WebSphere Education

Installation

- Distributed process takes an hour or less; zSeries: 2 hours (if you know everything and have all authorizations)
- Launchpad installation option (Windows only):
  - For Message Broker Toolkit, Message Broker Explorer and broker
  - Checks for prerequisite software and optionally installs if needed
- Restart optional
- Install maintenance after installing the base product, if necessary.
- After installing the Message Broker Toolkit, use supplied tools to verify installation, install samples, or learn more about Message Broker

Figure 2-13. Installation

WM643 / VM6431.0

Notes:

Installation time depends on which parts of the product are selected to be installed, whether all of the prerequisite software is already in place, and the speed of the processor. A typical installation takes less than an hour.

You should always check for the availability of maintenance updates (fix packs) and new releases for your version of WebSphere Message Broker. You can check for software updates by selecting **Help > Software** updates from the WebSphere Message Broker Explorer or WebSphere Message Broker Toolkit. You can also check the list of recommended fixes at `www.ibm.com/support/docview.wss?rs=849&uid=swg27006041`

Some of the tools available to verify that the installation is successful are described later in this unit.

Instructor notes:

Purpose — Establish some installation parameters.

Details — None.

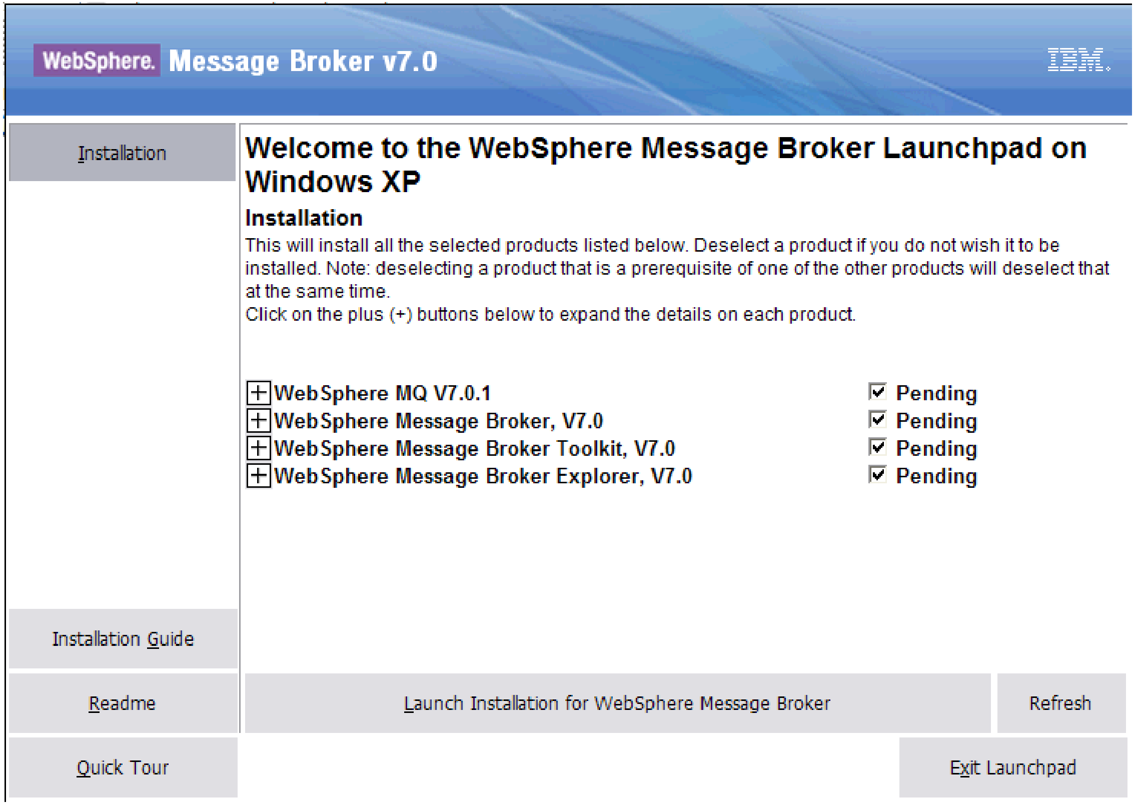
Additional information — Stress that the students must look at the readme file ***before*** attempting an installation. It contains some critical information.

Transition statement — If you are installing WebSphere Message Broker in the Windows environment, you can use the built-in Launchpad utility to assist with the installation of the base product and prerequisite software.

V

WebSphere Education

Installation using the Launchpad (Windows only)


WebSphere. Message Broker v7.0
IBM.
Installation
Installation Guide
Readme
Quick Tour
Launch Installation for WebSphere Message Broker
Refresh
Exit Launchpad

Welcome to the WebSphere Message Broker Launchpad on Windows XP

Installation

This will install all the selected products listed below. Deselect a product if you do not wish it to be installed. Note: deselecting a product that is a prerequisite of one of the other products will deselect that at the same time.
Click on the plus (+) buttons below to expand the details on each product.

- ⊞ WebSphere MQ V7.0.1 ☑ Pending
- ⊞ WebSphere Message Broker, V7.0 ☑ Pending
- ⊞ WebSphere Message Broker Toolkit, V7.0 ☑ Pending
- ⊞ WebSphere Message Broker Explorer, V7.0 ☑ Pending

Figure 2-14. Installation using the Launchpad (Windows only) WM643 / VM6431.0

Notes:

The Launchpad walks you through the installation process.

The Launchpad Installation window lists the minimum set of products required for a default configuration. It checks to see if the prerequisite software has been installed, and installs it if not. For example, WebSphere MQ Explorer requires the WebSphere Eclipse Platform to be installed; when you select WebSphere MQ V7.0.1, the Eclipse Platform is automatically installed.

When the Launchpad starts each installation, it updates the status from *Pending* to I*n Progress*, and finally to *Installed*.

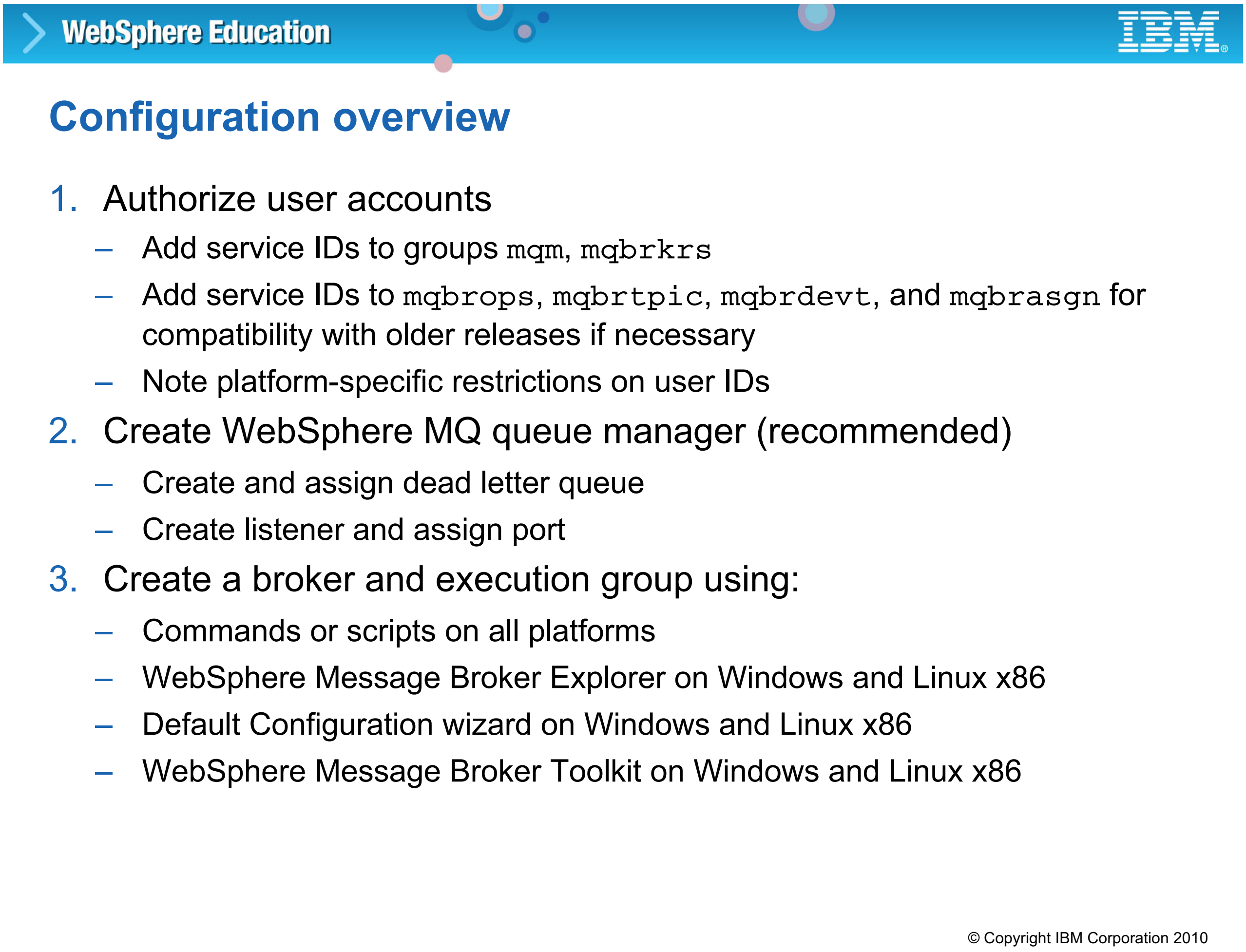
Instructor notes:

Purpose — Describe how the Launchpad helps to simplify installation.

Details — Launchpad is only available on Windows.

Additional information —

Transition statement — After the components have been installed, the administrator must perform some configuration steps.



WebSphere Education

Configuration overview

1. Authorize user accounts
   - Add service IDs to groups `mqm`, `mqbrkrs`
   - Add service IDs to `mqbrops`, `mqbrtpic`, `mqbrdevt`, and `mqbrasgn` for compatibility with older releases if necessary
   - Note platform-specific restrictions on user IDs
2. Create WebSphere MQ queue manager (recommended)
   - Create and assign dead letter queue
   - Create listener and assign port
3. Create a broker and execution group using:
   - Commands or scripts on all platforms
   - WebSphere Message Broker Explorer on Windows and Linux x86
   - Default Configuration wizard on Windows and Linux x86
   - WebSphere Message Broker Toolkit on Windows and Linux x86

Figure 2-15. Configuration overview WM643 / VM6431.0

Notes:

Some additional administrative actions are required before the WebSphere Message Broker can be used.

Administrators and authorized users must be added to the appropriate security groups and granted authorization to deploy and manage brokers and broker artifacts. Some security setup steps occur automatically during installation. For example, the `mqbrkrs` security group is defined, and the installing user ID is added to that group. If WebSphere MQ is installed after WebSphere Message Broker, you must create the `mqbrkrs` group manually. The `mqsisetsecurity` command performs that function. There is more on security later in this unit.

Each broker requires a queue manager which can be created before creating the broker or at the same time the broker is created. If you create the queue manager before creating the broker you can assign a dead letter queue, the listener, and port for the queue manager. Brokers can access only local queue managers, so you cannot create a broker on a queue manager that is on a remote system. You can create the queue manager using scripts or the WebSphere Message Broker Explorer.

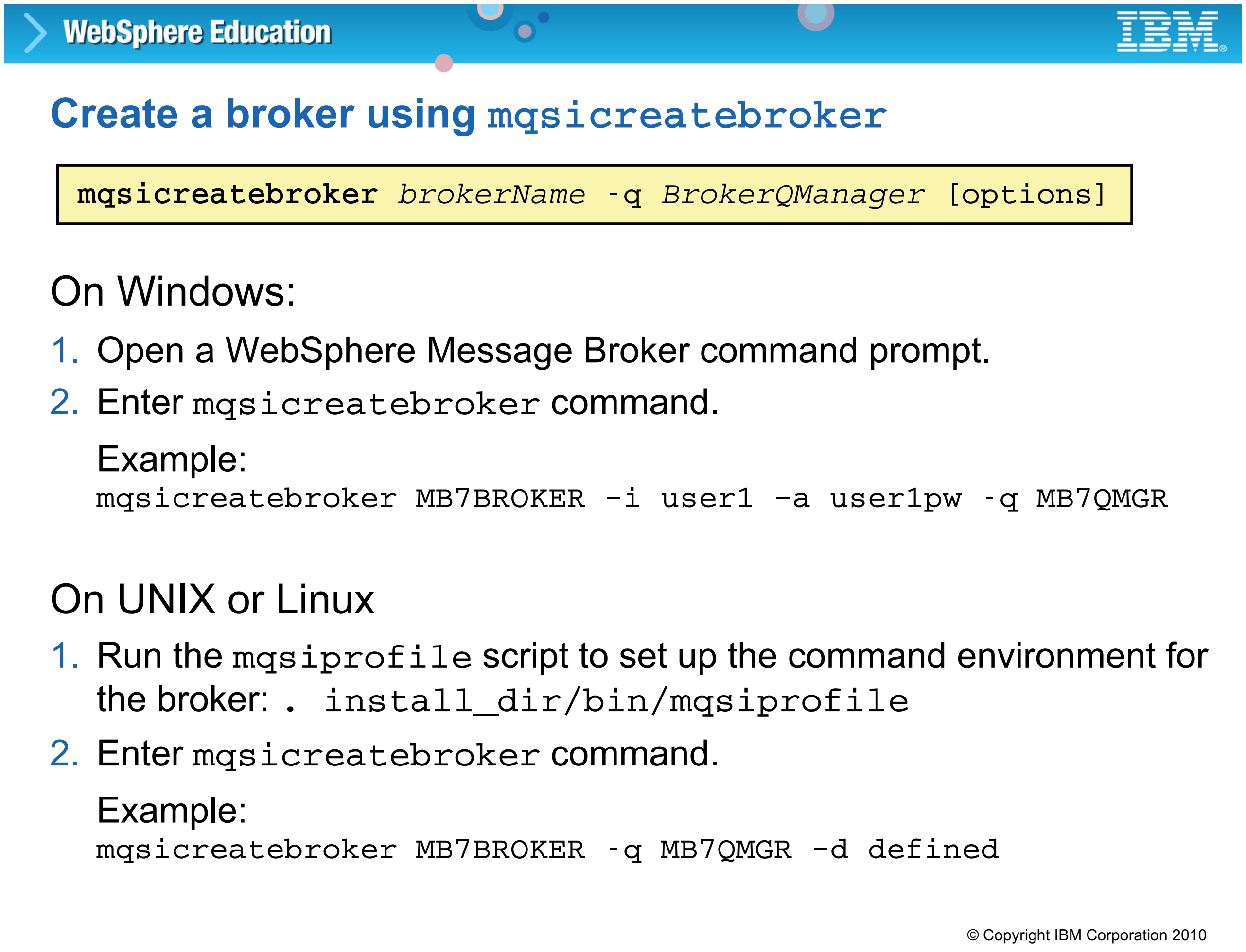
Instructor notes:

Purpose — Identify post-installation setup requirements and tools available to accomplish these tasks.

Details — Some degree of security setup occurs during installation. The mqbrkrs group is defined and the installing user ID is added. On occasion, it might be required to create the mqbrkrs group manually. The `mqsisetsecurity` command serves this purpose.

Additional information — If you want to operate your WebSphere Message Broker instances in a highly available configuration, you can set up your brokers to work either with a high availability manager, such as HACMP, or with WebSphere MQ multi-instance queue managers. You also have the option to set up a queue manager cluster. The number of required channel definitions are two per broker queue manager combination. No transmission queues are required, but listeners on the other hand still must be created and started

Transition statement — Creating and activating a component are two different tasks. So, how do you create broker components?



WebSphere Education

Create a broker using `mqsicreatebroker`

```
mqsicreatebroker brokerName -q BrokerQManager [options]
```

On Windows:

1. Open a WebSphere Message Broker command prompt.
2. Enter `mqsicreatebroker` command.

   Example:

   ```
   mqsicreatebroker MB7BROKER -i user1 -a user1pw -q MB7QMGR
   ```

On UNIX or Linux

1. Run the `mqsiprofile` script to set up the command environment for the broker: `. install_dir/bin/mqsiprofile`
2. Enter `mqsicreatebroker` command.

   Example:

   ```
   mqsicreatebroker MB7BROKER -q MB7QMGR -d defined
   ```

Figure 2-16. Create a broker using mqsicreatebroker WM643 / VM6431.0

Notes:

The `mqsicreatebroker` command can be used to create a broker and its associated resources.

In the example for Windows, the `mqsicreatebroker` command creates a broker named `MB7BROKER` with a service ID of `user1` (`-i`) and password of `user1pw` (`-a`), and associates the queue manager `MB7QMGR` (`-q`) with the broker instance.

In the example for UNIX or Linux, the `mqsicreatebroker` command creates a broker named `MB7BROKER` associated with queue manager named `MB7QMGR` (`-q`). The broker has been defined to start and stop when queue manager starts and stops (`-d defined`).

If you run this command on Linux, UNIX, and Windows, it creates a queue manager if it does not exist.

See the WebSphere Message Broker Information Center for a detailed description of the `mqsicreatebroker` command and command syntax options.

You can check the existence of the broker using: `mqsilist`

Obtain a report of the details of the broker with: `mqsiservice <Component_Name>`

Instructor notes:

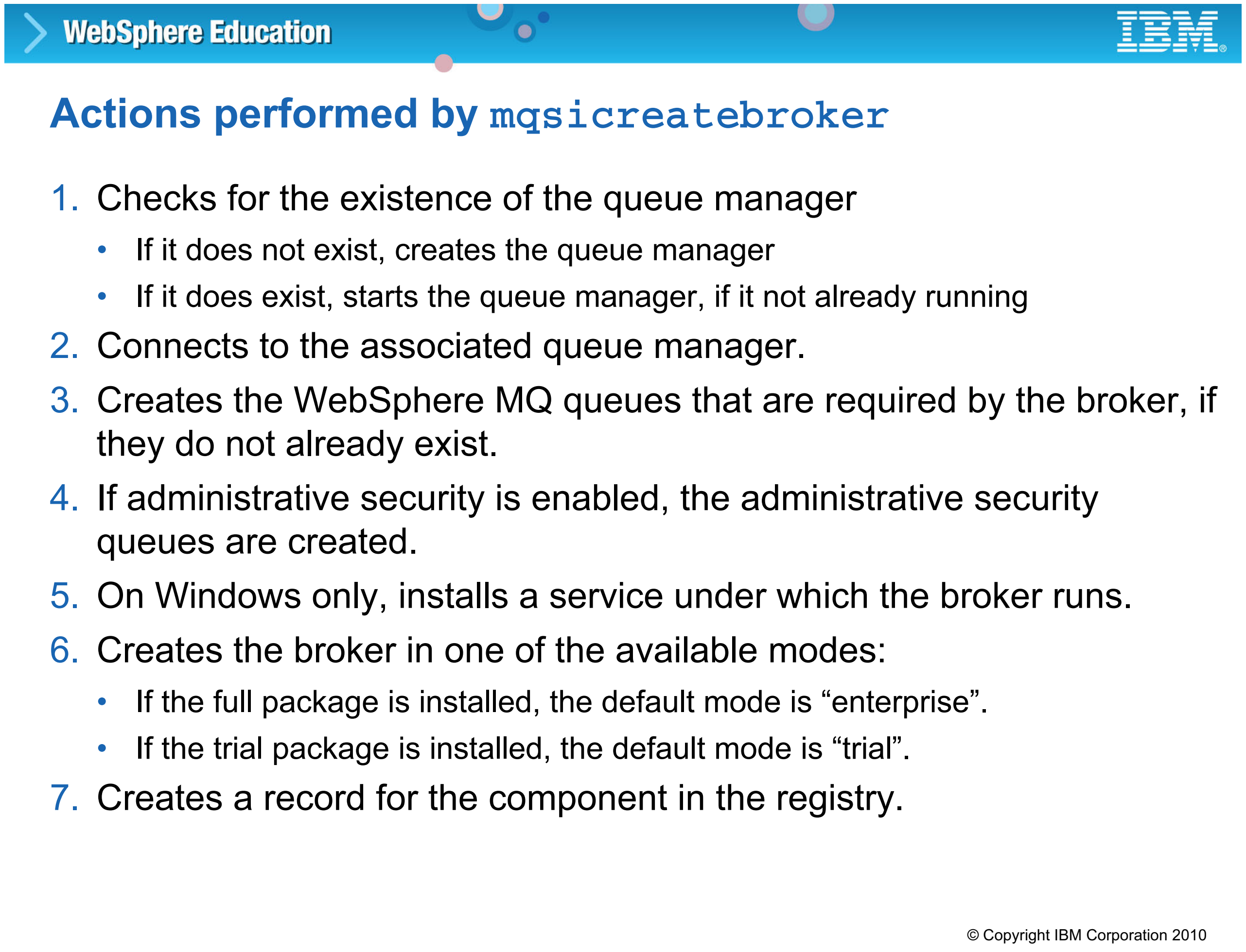
Purpose — Describe examples of the `mqsicreatebroker` command

Details — Emphasize that the command syntax options vary between operating systems.

The `mqsilist` and `mqsiservice` commands are described in detail later in this unit,

Additional information —

Transition statement — Besides creating the broker, the `mqsicreatebroker` command performs a number of related actions.



WebSphere Education

Actions performed by `mqsicreatebroker`

1. Checks for the existence of the queue manager
   - If it does not exist, creates the queue manager
   - If it does exist, starts the queue manager, if it not already running
2. Connects to the associated queue manager.
3. Creates the WebSphere MQ queues that are required by the broker, if they do not already exist.
4. If administrative security is enabled, the administrative security queues are created.
5. On Windows only, installs a service under which the broker runs.
6. Creates the broker in one of the available modes:
   - If the full package is installed, the default mode is “enterprise”.
   - If the trial package is installed, the default mode is “trial”.
7. Creates a record for the component in the registry.

Figure 2-17. Actions performed by mqsicreatebroker WM643 / VM6431.0

Notes:

The `mqsicreatebroker` command performs a number of actions, besides creating the broker and queue manager.

Note

The `mqsicreatebroker` command does not automatically create the queue manager on z/OS.

If the queue manager is created on Windows by the `mqsicreatebroker` command, it is not started as a service.

On Windows platforms, Linux, and UNIX systems, a command option is available so that you to specify, when the broker is created, whether the broker can be started and stopped as a WebSphere MQ service.

Instructor notes:

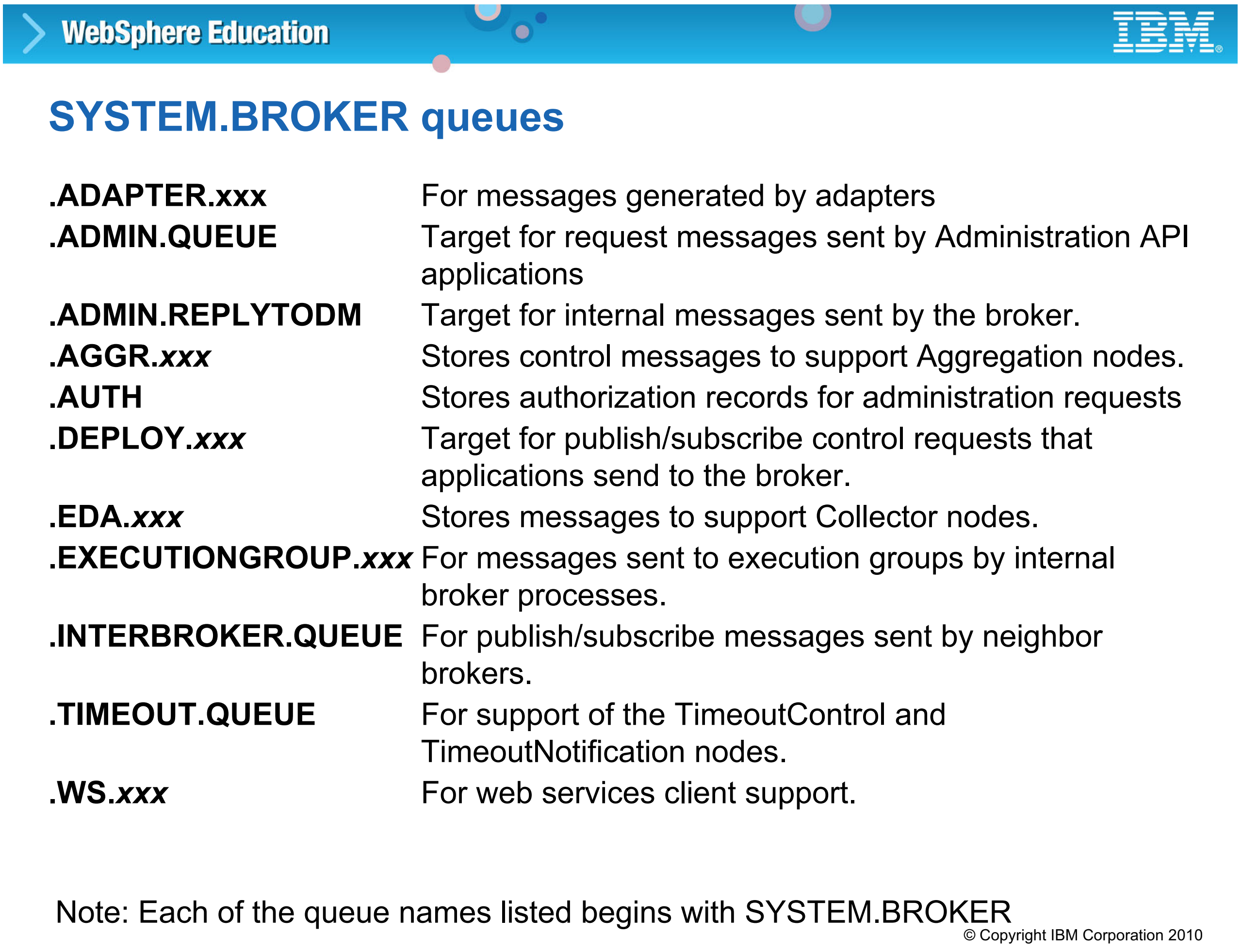
Purpose — List the additional actions performed by `mqsicreatebroker` command.

Details — Highlight the differences in the actions based on the platform.

Additional information — The operation mode is determined by the license:

- Enterprise mode. All features are enabled and no restrictions or limits are imposed. This mode is the default mode, unless you have the Trial Edition.
- Starter Edition mode. All features are enabled, but the number of execution groups and deployed message flows are limited.
- Remote Adapter Deployment mode. Only adapter-related features are enabled. The types of node that can be used and the number of execution groups are limited.
- Trial Edition mode. All features are enabled, but the product can only be used for 90 days after installation.

Transition statement —



WebSphere Education
IBM

SYSTEM.BROKER queues

.ADAPTER.xxx	For messages generated by adapters
.ADMIN.QUEUE	Target for request messages sent by Administration API applications
.ADMIN.REPLYTODM	Target for internal messages sent by the broker.
.AGGR.*xxx*	Stores control messages to support Aggregation nodes.
.AUTH	Stores authorization records for administration requests
.DEPLOY.*xxx*	Target for publish/subscribe control requests that applications send to the broker.
.EDA.*xxx*	Stores messages to support Collector nodes.
.EXECUTIONGROUP.*xxx*	For messages sent to execution groups by internal broker processes.
.INTERBROKER.QUEUE	For publish/subscribe messages sent by neighbor brokers.
.TIMEOUT.QUEUE	For support of the TimeoutControl and TimeoutNotification nodes.
.WS.*xxx*	For web services client support.

Note: Each of the queue names listed begins with SYSTEM.BROKER

Figure 2-18. SYSTEM.BROKER queues WM643 / VM6431.0

Notes:

One of the actions during the process of creating a broker is to create the WebSphere MQ queues required by broker, if they do not already exist.

The names of these queues begin with the reserved characters `SYSTEM.BROKER`. The resources are additional to the default WebSphere MQ objects that are created when you install that product.

The figure summarizes the SYSTEM.BROKER queues. You will work with some of these queues such as the `SYSTEM.BROKER.AUTH` queue, later in this course.

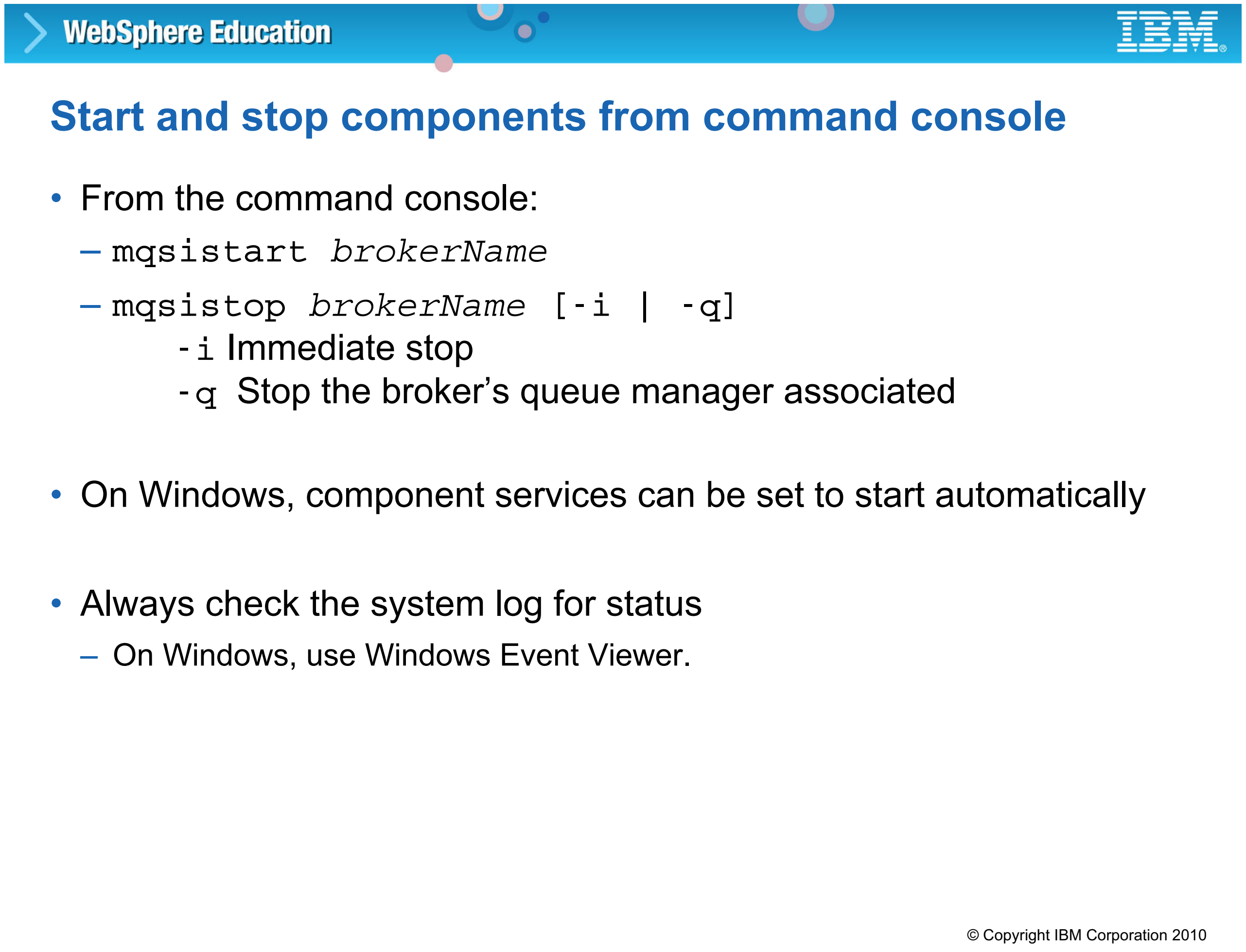
Instructor notes:

Purpose — Provide an overview of the system queues created for the broker.

Details — To demonstrate, open WebSphere Message Broker Explorer and show the system queues for a broker.

Additional information — None.

Transition statement —



WebSphere Education

Start and stop components from command console

- From the command console:
  - `mqsistart` *`brokerName`*
  - `mqsistop` *`brokerName`* `[-i | -q]`

    `-i` Immediate stop
    `-q` Stop the broker's queue manager associated

- On Windows, component services can be set to start automatically

- Always check the system log for status
  - On Windows, use Windows Event Viewer.

Figure 2-19. Start and stop components from command console WM643 / VM6431.0

Notes:

The `mqsistart` and `mqsistop` commands are supported on Windows, Linux, UNIX, and z/OS.

In Windows, the broker can be started from the Windows Services control panel. It is useful to set it to start automatically when the Windows server starts. When verifying running status, check that the *IBM MQSeries* service is also active.

The `mqsistart` command performs a number of actions, such as starting the queue manager on distributed systems, and verifying that the broker environment is set up correctly.

See the system event log messages if any errors occur while processing the `mqsistart` or `mqsistop` commands.

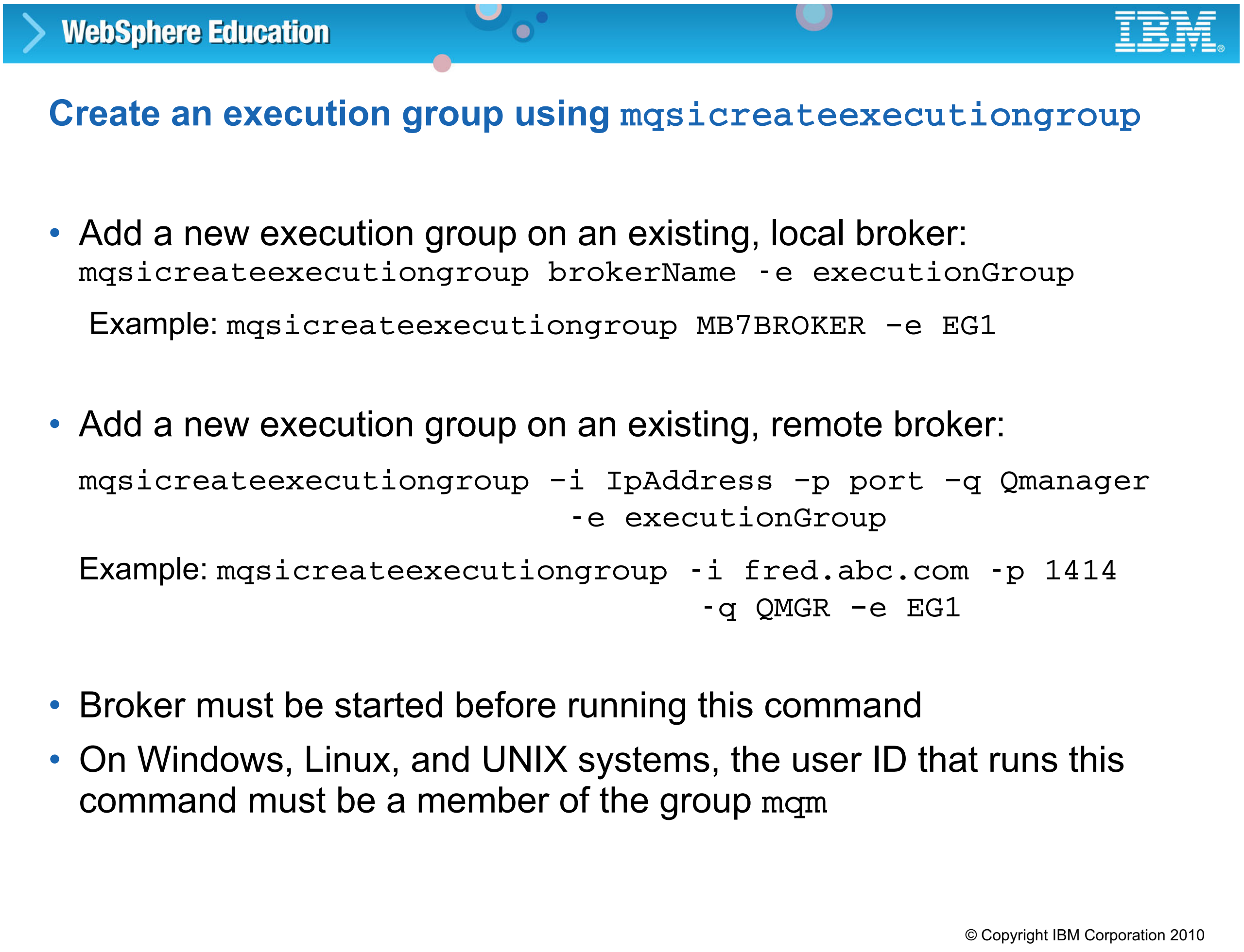
Instructor notes:

Purpose — Establish the need to start the WebSphere Message Broker components.

Details — None.

Additional information — None.

Transition statement — After the broker is created and running, the next step is to create execution groups.



WebSphere Education

Create an execution group using `mqsicreateexecutiongroup`

- Add a new execution group on an existing, local broker:

```
mqsicreateexecutiongroup brokerName -e executionGroup
```

  Example: `mqsicreateexecutiongroup MB7BROKER -e EG1`

- Add a new execution group on an existing, remote broker:

```
mqsicreateexecutiongroup -i IpAddress -p port -q Qmanager
                          -e executionGroup
```

  Example:

```
mqsicreateexecutiongroup -i fred.abc.com -p 1414
                                -q QMGR -e EG1
```

- Broker must be started before running this command
- On Windows, Linux, and UNIX systems, the user ID that runs this command must be a member of the group `mqm`

Figure 2-20. Create an execution group using mqsicreateexecutiongroup WM643 / VM6431.0

Notes:

Execution groups can be created from a command console, and on Windows and Linux from WebSphere Message Broker Explorer and WebSphere Message Broker Toolkit.

Security authorization can be implemented to restrict users who can create execution groups. Security authorization is described in a later unit.

The first example of the `mqsicreateexecutiongroup` command creates an execution group named `EG1` (-e) on a local broker named `MB7BROKER`.

The second example of the `mqsicreateexecutiongroup` command creates an execution group named `EG1` (`-e`) on the broker that is hosted by the queue manager named `QMGR` (`-q`) which is listening on `server fred.abc.com` (`-i`) using port 1414 (`-p`).

See the WebSphere Message Broker Information Center for more information about the `mqsicreateexecutiongroup` command and command options.

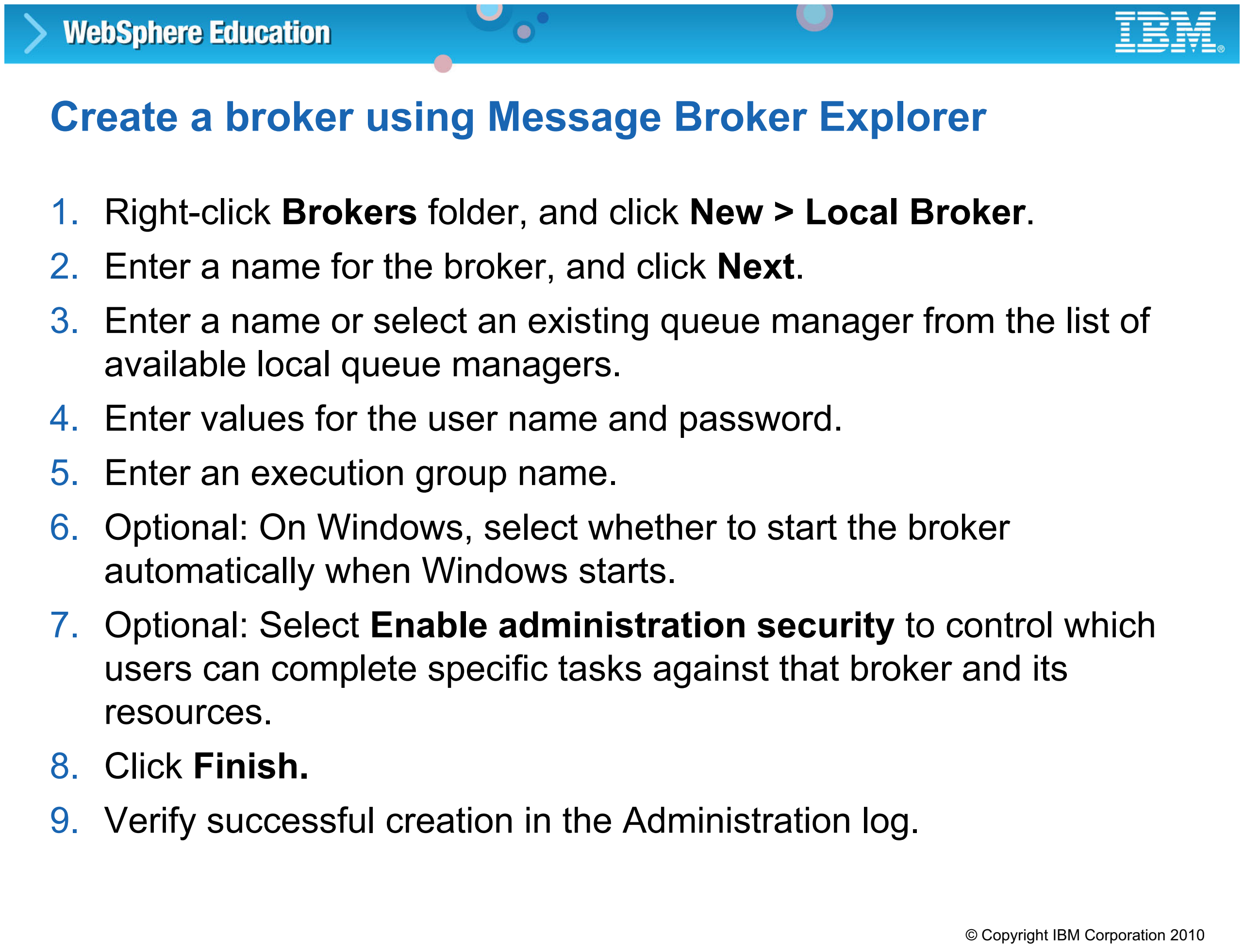
Instructor notes:

Purpose — Describe the function and syntax of the `mqsicreateexecutiongroup` command to create execution groups on local and remote brokers.

Details —

Additional information — Additional security authorization can be implemented. Security implementation is described in a later unit in this course.

Transition statement — Local brokers and execution groups can be created using WebSphere Message Broker Explorer on Windows and Linux.



WebSphere Education

Create a broker using Message Broker Explorer

1. Right-click **Brokers** folder, and click **New > Local Broker**.
2. Enter a name for the broker, and click **Next**.
3. Enter a name or select an existing queue manager from the list of available local queue managers.
4. Enter values for the user name and password.
5. Enter an execution group name.
6. Optional: On Windows, select whether to start the broker automatically when Windows starts.
7. Optional: Select **Enable administration security** to control which users can complete specific tasks against that broker and its resources.
8. Click **Finish.**
9. Verify successful creation in the Administration log.

Figure 2-21. Create a broker using Message Broker Explorer WM643 / VM6431.0

Notes:

The figure lists the steps for creating a local broker using WebSphere Message Broker Explorer.

To start WebSphere Message Broker Explorer on Linux, enter the `strmqcfg` command on a command line, or run `/usr/bin/strmqcfg`.

To start WebSphere Message Broker Explorer on Windows, select **IBM WebSphere MQ > WebSphere MQ Explorer** or **IBM WebSphere Message Broker 7.0 > IBM WebSphere Message Broker Explorer** from the Windows **Program** menu.

On Windows, you must have administrator access rights to create brokers using WebSphere Message Broker Explorer. When creating a broker, you might be prompted to agree to the use of administrator rights or to enter an administrator user ID and password.

If the WebSphere MQ queue manager does not exist, the broker creation wizard automatically creates the queue manager. The wizard requires that you create an execution group on the broker. You can create additional execution groups after the broker is created.

Instructor notes:

Purpose — List the steps for creating a local broker on Windows or Linux using WebSphere Message Broker Explorer.

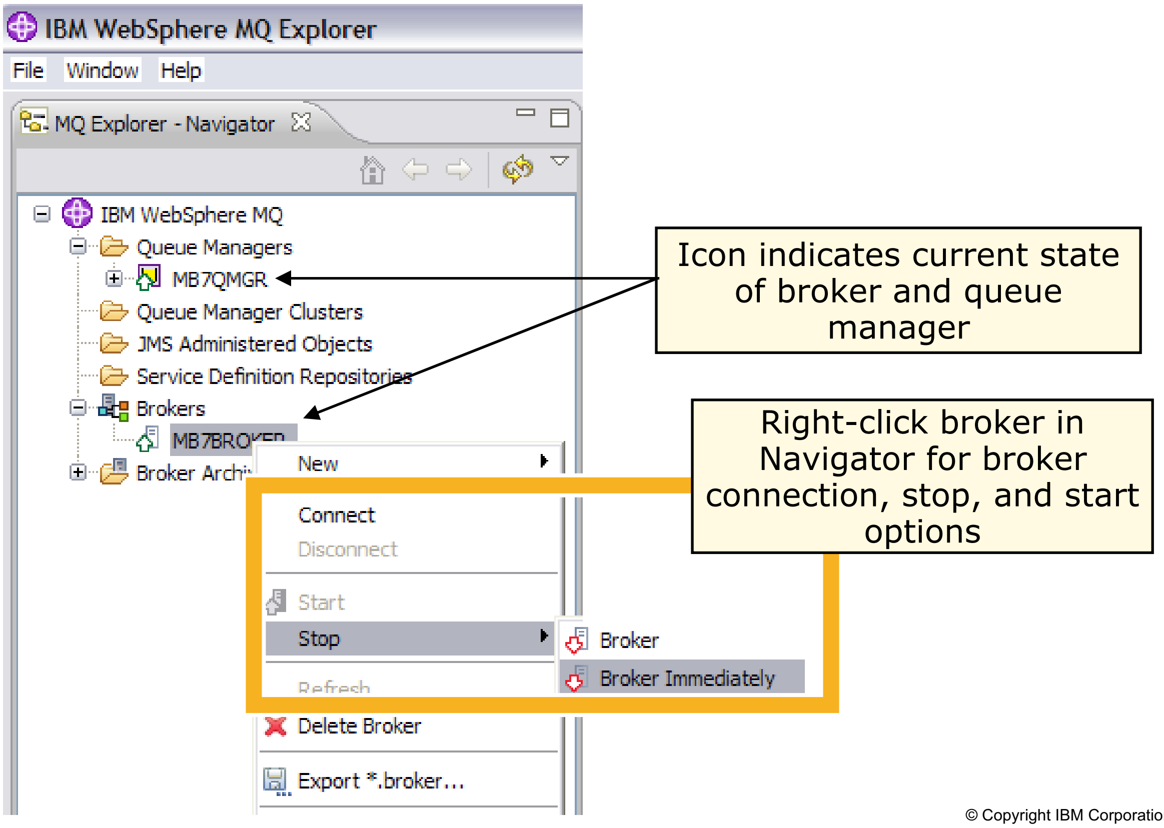
Details — Demonstrate creating a local broker using WebSphere Message Broker Explorer.

Additional information — Students will create a broker using commands and WebSphere Message Broker Explorer in a lab exercise.

Transition statement —

WebSphere Education

Start and stop broker from Message Broker Explorer



Icon indicates current state of broker and queue manager

Right-click broker in Navigator for broker connection, stop, and start options

Figure 2-22. Start and stop broker from Message Broker Explorer WM643 / VM6431.0

Notes:

A broker can be stopped and started from WebSphere Message Broker Explorer.

The icon colors and appearance indicate the current state of the broker. For example, an icon with a green arrow that is pointing up, indicates that the broker is running.

When stopping the broker, you also have the option of stopping the broker immediately by selecting **Stop > WebSphere Message Broker Immediately**. Specify this option only if you have already tried, and failed, to stop the broker in a controlled fashion by selecting **Stop > WebSphere Message Broker.**

Instructor notes:

Purpose — Show that many of the same functions that are performed using a console command can also be performed using WebSphere Message Broker Explorer.

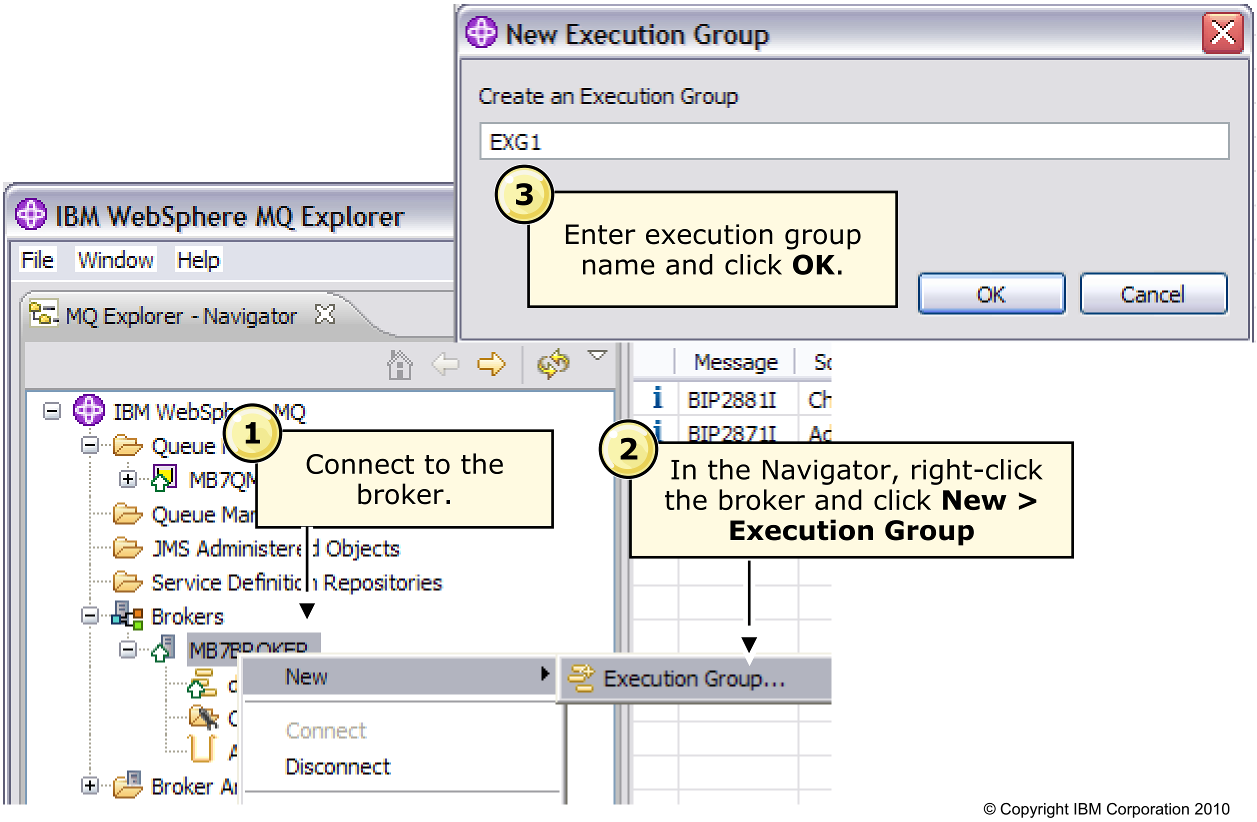
Details — The students should already be familiar with the state icons from WebSphere MQ Explorer.

Additional information —

Transition statement —

WebSphere Education

Create an execution group using Message Broker Explorer



1 Connect to the broker.

2 In the Navigator, right-click the broker and click **New > Execution Group**

3 Enter execution group name and click **OK**.

Figure 2-23. Create an execution group using Message Broker Explorer WM643 / VM6431.0

Notes:

To create an execution group from WebSphere Message Broker Explorer:

1. If the broker is not already connected, right-click the broker in the Navigator and select **Connect** from the menu.
2. Right-click the broker and select **New** > **Execution Group** from the menu.
3. Enter the name of the new execution group on the **New Execution Group** window and then click **OK**.

Instructor notes:

Purpose — Show the steps for creating an execution group on a local broker using WebSphere Message Broker Explorer

Details —

Additional information —

Transition statement — You can connect to both local and remote brokers from WebSphere Message Broker Explorer.

WebSphere Education IBM

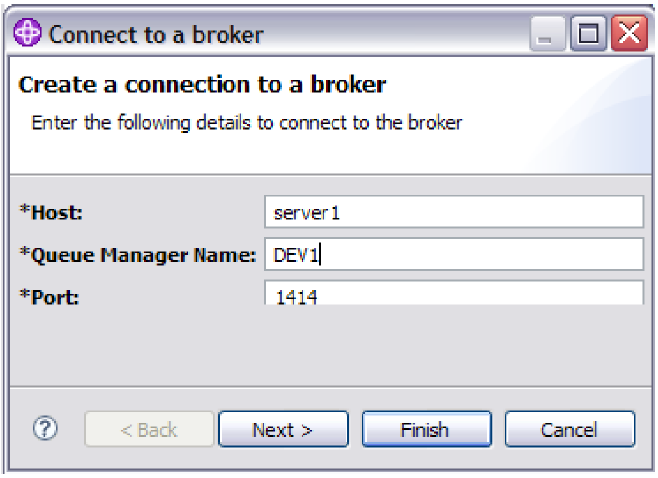
Connect to broker from Message Broker Explorer

To connect to local broker:

1. In the Navigator view, expand the **Brokers** folder.
2. Right-click the local broker and click **Connect.**

To connect to remote broker:

1. Right-click the **Brokers** folder, and click **Connect to a Remote Broker**.
2. Enter the remote broker queue manager name.
3. Enter the host name or IP address of the remote server.
4. Enter the WebSphere MQ queue manager TCP listening port (the default is 1414).
5. Optional: Click **Next** and enter channel security parameters.
6. Click **Finish**.



Connect to a broker

Create a connection to a broker

Enter the following details to connect to the broker

*Host: server1

*Queue Manager Name: DEV1

*Port: 1414

Figure 2-24. Connect to broker from Message Broker Explorer WM643 / VM6431.0

Notes:

In WebSphere Message Broker Explorer, you can choose to connect and disconnect from both local and remote brokers. You can also choose to automatically reconnect to a broker when you start WebSphere Message Broker Explorer.

Channel security is described in more detail in Unit 5, “Implementing WebSphere Message Broker security”.

Instructor notes:

Purpose — Describe the steps for connecting to remote and local broker from WebSphere Message Broker Explorer.

Details —

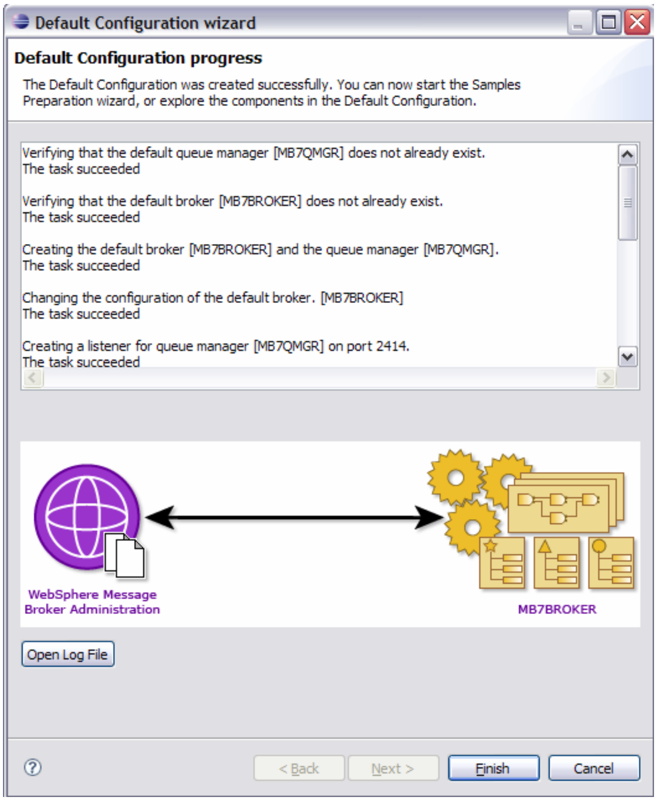
Additional information —

Transition statement — New users and developers can create a queue manager, broker, and execution group using the Default Configuration wizard.

WebSphere Education IBM

Create components using the Default Configuration wizard

- Accessed through WebSphere Message Broker Toolkit
- Creates complete local unit test configuration
  - Broker named **MB7BROKER**
  - Queue manager named **MB7QMGR**
  - Queue manager listener on port 2414
  - Execution group named **default**
- Requires Administrator privileges and local user ID on Windows


Default Configuration wizard
WebSphere Message Broker Administration
MB7BROKER
Open Log File
< Back
Next >
Finish
Cancel

Default Configuration progress

The Default Configuration was created successfully. You can now start the Samples Preparation wizard, or explore the components in the Default Configuration.

```
Verifying that the default queue manager [MB7QMGR] does not already exist.
The task succeeded

Verifying that the default broker [MB7BROKER] does not already exist.
The task succeeded

Creating the default broker [MB7BROKER] and the queue manager [MB7QMGR].
The task succeeded

Changing the configuration of the default broker. [MB7BROKER]
The task succeeded

Creating a listener for queue manager [MB7QMGR] on port 2414.
The task succeeded
```

Figure 2-25. Create components using the Default Configuration wizard WM643 / VM6431.0

Notes:

The Default Configuration wizard creates a broker and all the components that are required to import and deploy the WebSphere Message Broker samples, and to develop and test message flows.

The Default Configuration wizard can be used to create a simple configuration on your workstation for message flow testing.

The default configuration assumes port 2414 can be used for the WebSphere MQ queue manager listener. You should verify that this port is not being used by another application.

To start the Default Configuration wizard from the WebSphere Message Broker Toolkit:

1. Select **Help > Welcome**.
2. On the Welcome page, click **Get Started**.
3. Click **Create the default configuration**.

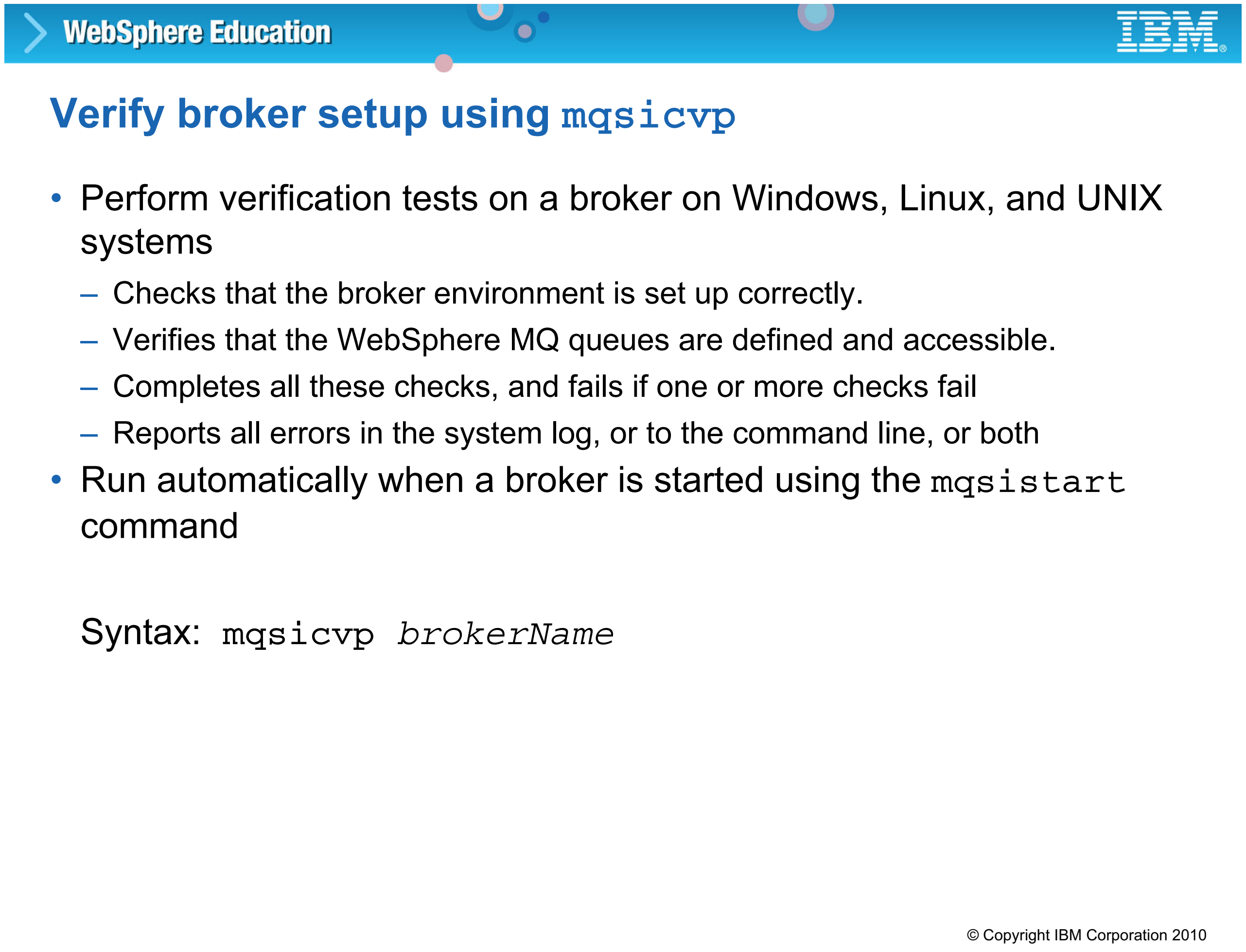
Instructor notes:

Purpose — Describe the Default Configuration wizard

Details — Show the student's where the option to start the Default Configuration wizard is located.

Additional information —

Transition statement — You can verify the broker setup using the `mqsicvp` command.



WebSphere Education

Verify broker setup using `mqsicvp`

- Perform verification tests on a broker on Windows, Linux, and UNIX systems
  - Checks that the broker environment is set up correctly.
  - Verifies that the WebSphere MQ queues are defined and accessible.
  - Completes all these checks, and fails if one or more checks fail
  - Reports all errors in the system log, or to the command line, or both
- Run automatically when a broker is started using the `mqsistart` command

Syntax: `mqsicvp` *`brokerName`*

Figure 2-26. Verify broker setup using mqsicvp WM643 / VM6431.0

Notes:

Use the `mqsicvp` command to perform verification tests on a broker.

The `mqsicvp` command checks that the broker environment is set up correctly; for example, that the installed level of Java is supported

You can run this command against a broker that is running, or is not running. If the broker is not running, the verification tests are performed, but the broker is not started.

Instructor notes:

Purpose — Describe how to use the mqsicvp command to verify the broker setup.

Details — The `mqsicvp` can also be used as an ODBC test. This application of the mqsicvp command is described in Unit 10, “Extending WebSphere Message Broker”

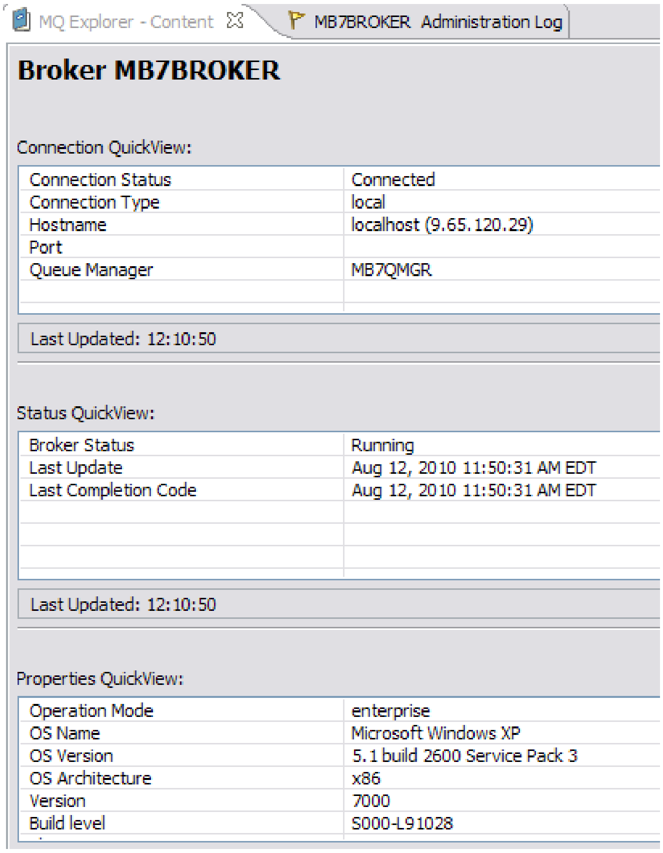
Additional information —

Transition statement —

WebSphere Education

Broker properties

- WebSphere Message Broker maintains a set of properties for each broker
- View broker properties using:
  - `mqsireportbroker` command
  - WebSphere Message Broker Explorer



Broker MB7BROKER

Connection QuickView:

Connection Status	Connected
Connection Type	local
Hostname	localhost (9.65.120.29)
Port	
Queue Manager	MB7QMGR

Last Updated: 12:10:50

Status QuickView:

Broker Status	Running
Last Update	Aug 12, 2010 11:50:31 AM EDT
Last Completion Code	Aug 12, 2010 11:50:31 AM EDT

Last Updated: 12:10:50

Properties QuickView:

Operation Mode	enterprise
OS Name	Microsoft Windows XP
OS Version	5.1 build 2600 Service Pack 3
OS Architecture	x86
Version	7000
Build level	S000-L91028

Figure 2-27. Broker properties WM643 / VM6431.0

Notes:

WebSphere Message Broker maintains a set of properties for each broker. Broker properties include the host name, queue manager name, and status.

You can view these properties in WebSphere Message Broker Explorer or in the command console using the `mqsireportbroker` command.

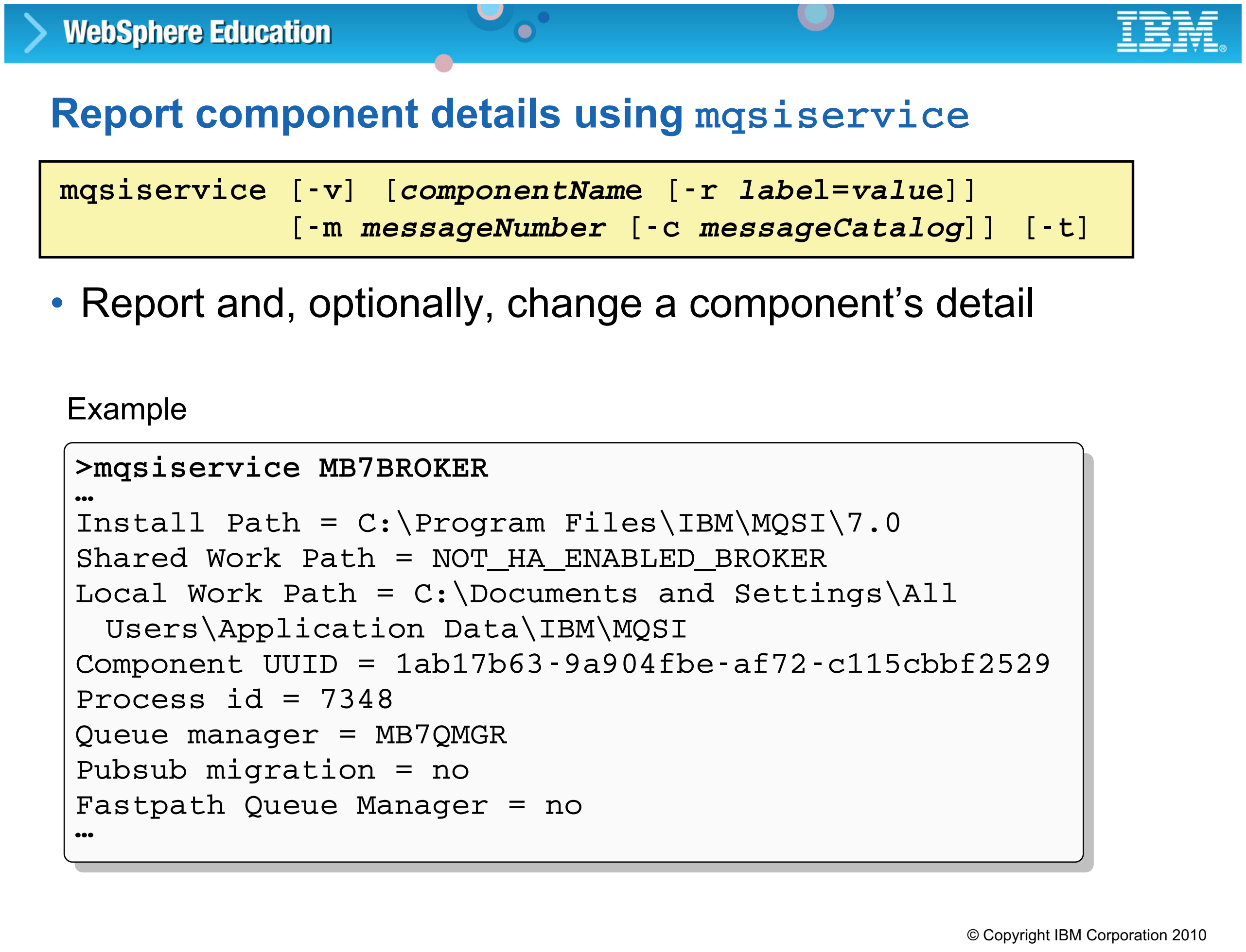
Instructor notes:

Purpose — Provide an overview of broker properties.

Details — Open WebSphere Message Broker Explorer and review some of the broker properties.

Additional information —

Transition statement —



WebSphere Education

Report component details using `mqsiservice`

```
mqsiservice [-v] [componentName [-r label=value]]
            [-m messageNumber [-c messageCatalog]] [-t]
```

- Report and, optionally, change a component’s detail

Example

```
>mqsiservice MB7BROKER
…
Install Path = C:\Program Files\IBM\MQSI\7.0
Shared Work Path = NOT_HA_ENABLED_BROKER
Local Work Path = C:\Documents and Settings\All
  Users\Application Data\IBM\MQSI
Component UUID = 1ab17b63-9a904fbe-af72-c115cbbf2529
Process id = 7348
Queue manager = MB7QMGR
Pubsub migration = no
Fastpath Queue Manager = no
…
```

Figure 2-28. Report component details using mqsiservice WM643 / VM6431.0

Notes:

The `mqsiservice` command can be run in a command console to report on and, optionally, change a component’s detail.

The figure includes an example of the `mqsiservice` command and an excerpt of the results of the `mqsiservice` command when the component is a broker.

Command options include:

`-v` is the product version information.

`componentName` is the name of the component to query for its registry data.

`-r label=value` sets the registry key 'label' to 'value'. Use with caution.

`-m messageNumber` is a message number to output.

`-c messageCatalog` is the name of the message catalog to use.

`-t` outputs information about current time and time zone.

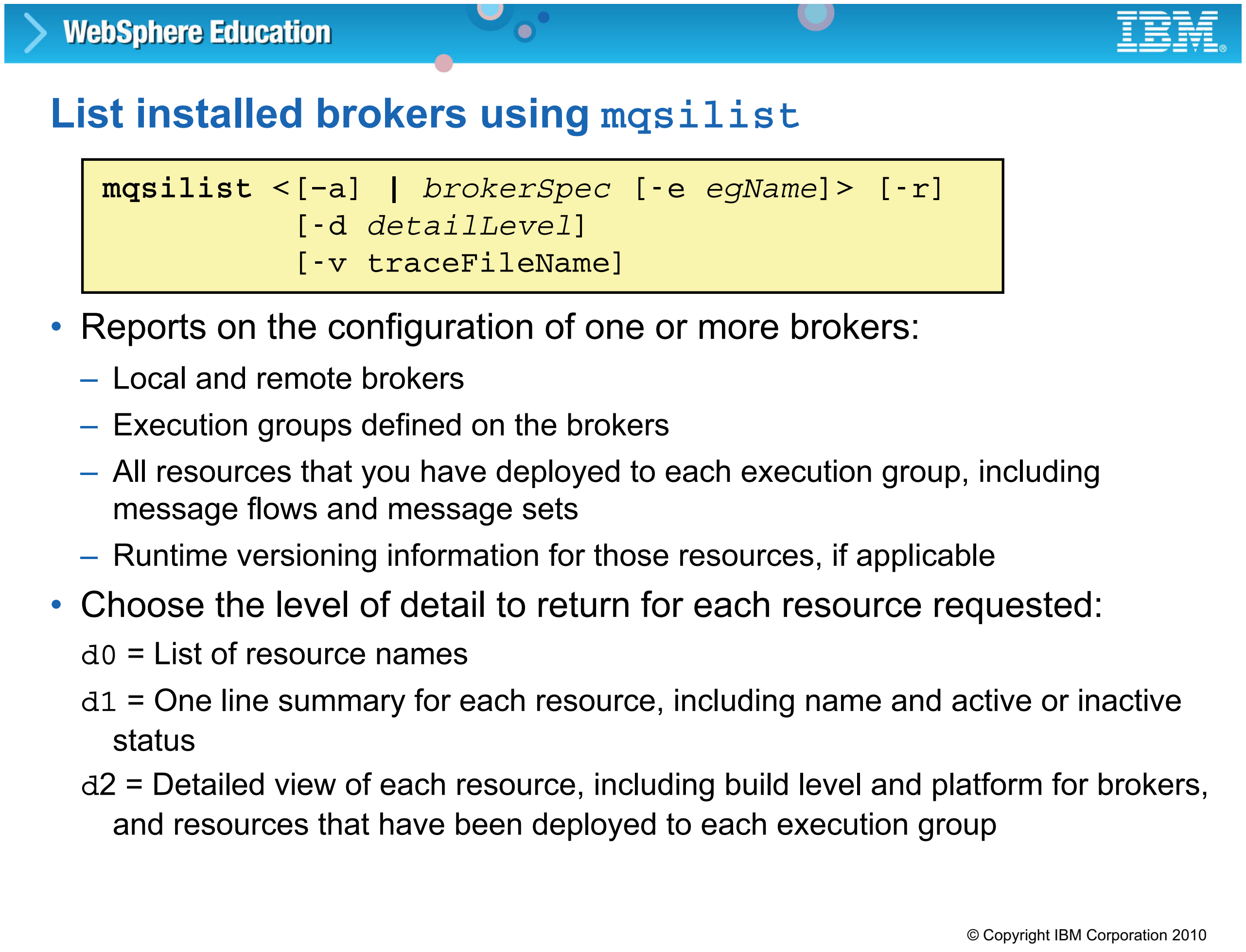
Instructor notes:

Purpose — Describe the use of the `mqsiservice` command to get broker details.

Details — Open a command console and demonstrate the use of the mqsiservice command to display broker details.

Additional information — This command is described under BIP8124.

Transition statement — Another command available for getting broker information is the `mqsilist` command.



WebSphere Education
IBM

List installed brokers using `mqsilist`

```
mqsilist <[-a] | brokerSpec [-e egName]> [-r]
           [-d detailLevel]
           [-v traceFileName]
```

- Reports on the configuration of one or more brokers:
  - Local and remote brokers
  - Execution groups defined on the brokers
  - All resources that you have deployed to each execution group, including message flows and message sets
  - Runtime versioning information for those resources, if applicable
- Choose the level of detail to return for each resource requested:

  `d0` = List of resource names

  `d1` = One line summary for each resource, including name and active or inactive status

  `d2` = Detailed view of each resource, including build level and platform for brokers, and resources that have been deployed to each execution group

Figure 2-29. List installed brokers using mqsilist

WM643 / VM6431.0

Notes:

Check the existence of the components you have created using the `mqsilist` command.

Obtain a report of the details of each component with: `mqsiservice <Component_Name>`

In the figure, optional parameters are enclosed in brackets [ ].

The `-a` option lists all the brokers installed on the local computer, in all installations.

The `-d` option specifies the type of detail to provide:

`-d0` returns only the broker name and the names of their associated queue managers.

`-d1` returns a one line summary of each resource.

`-d2` returns detailed information about each resource

The `-r` option causes the command to run the command recursively and display information about subcomponents.

See the WebSphere Message Broker Information Center for more information about command syntax and options.

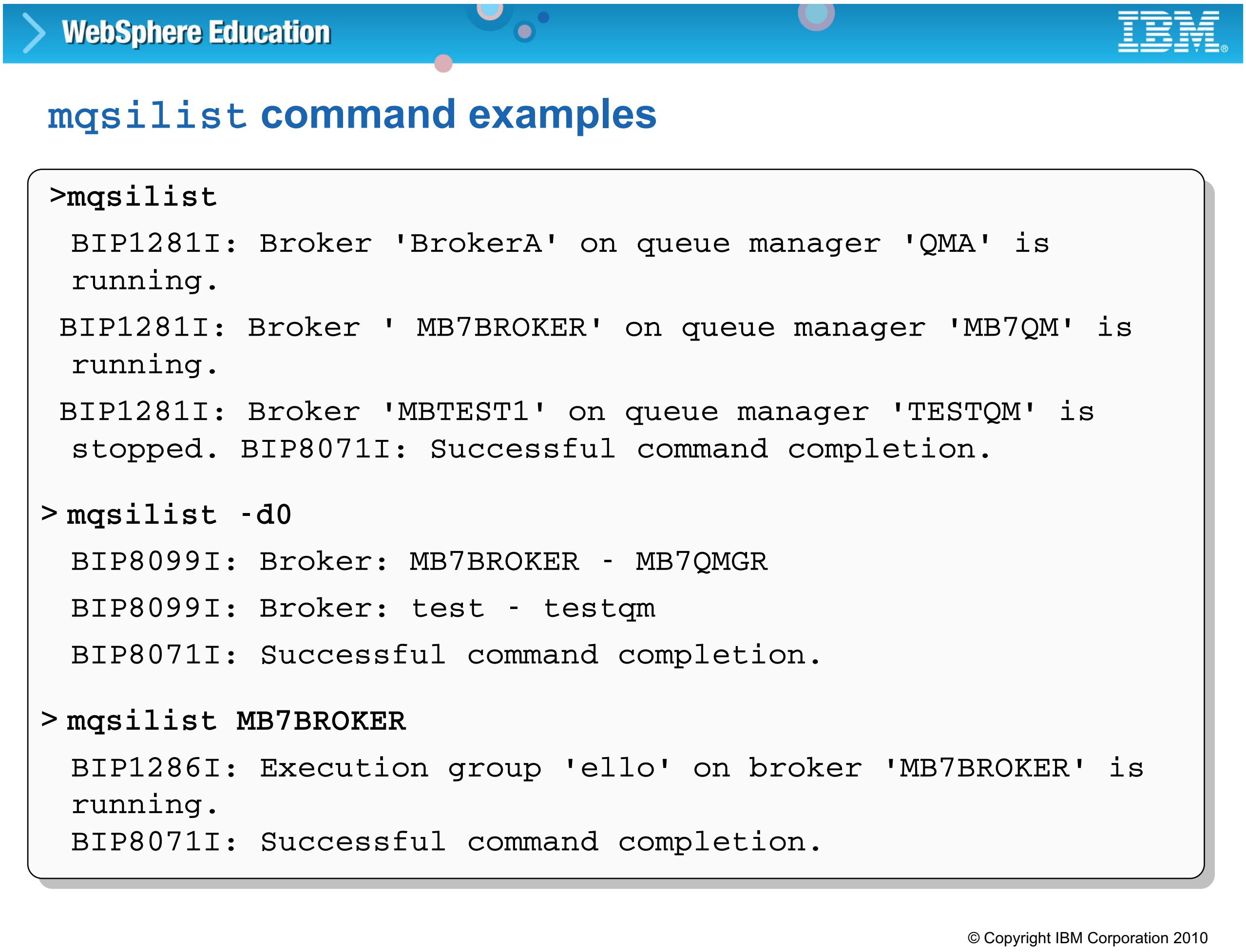
Instructor notes:

Purpose — Describe the `mqsilist` command to report on broker configuration.

Details — Demonstrate the `mqsilist` command.

Additional information — The Windows registry and equivalent UNIX files are yet another place to find details of the installed products and defined components.

Transition statement — The next slide shows some examples of the `mqsilist` command.



WebSphere Education

mqsilist command examples

```
>mqsilist
  BIP1281I: Broker 'BrokerA' on queue manager 'QMA' is
  running.
 BIP1281I: Broker ' MB7BROKER' on queue manager 'MB7QM' is
  running.
 BIP1281I: Broker 'MBTEST1' on queue manager 'TESTQM' is
  stopped. BIP8071I: Successful command completion.

> mqsilist -d0
  BIP8099I: Broker: MB7BROKER - MB7QMGR
  BIP8099I: Broker: test - testqm
  BIP8071I: Successful command completion.

> mqsilist MB7BROKER
  BIP1286I: Execution group 'ello' on broker 'MB7BROKER' is
  running.
  BIP8071I: Successful command completion.
```

Figure 2-30. mqsilist command examples WM643 / VM6431.0

Notes:

The figure shows some examples of the `mqsilist` command and the results.

In the first example, the `mqsilist` is run without any additional options. It reports the status of each broker on this system.

In the second example, the `mqsilist` is run with the `-d0` option which returns the broker and its associated queue manager.

In the third example, the `mqsilist` command is run with the broker name as the only option. It returns the status of the execution group running on the specified broker.

Instructor notes:

Purpose — Show some examples of the `mqsilist` command and command results.

Details — Demonstrate various invocations of the `mqsilist` command.

Additional information —

Transition statement —

2.3. Security considerations

Instructor topic introduction

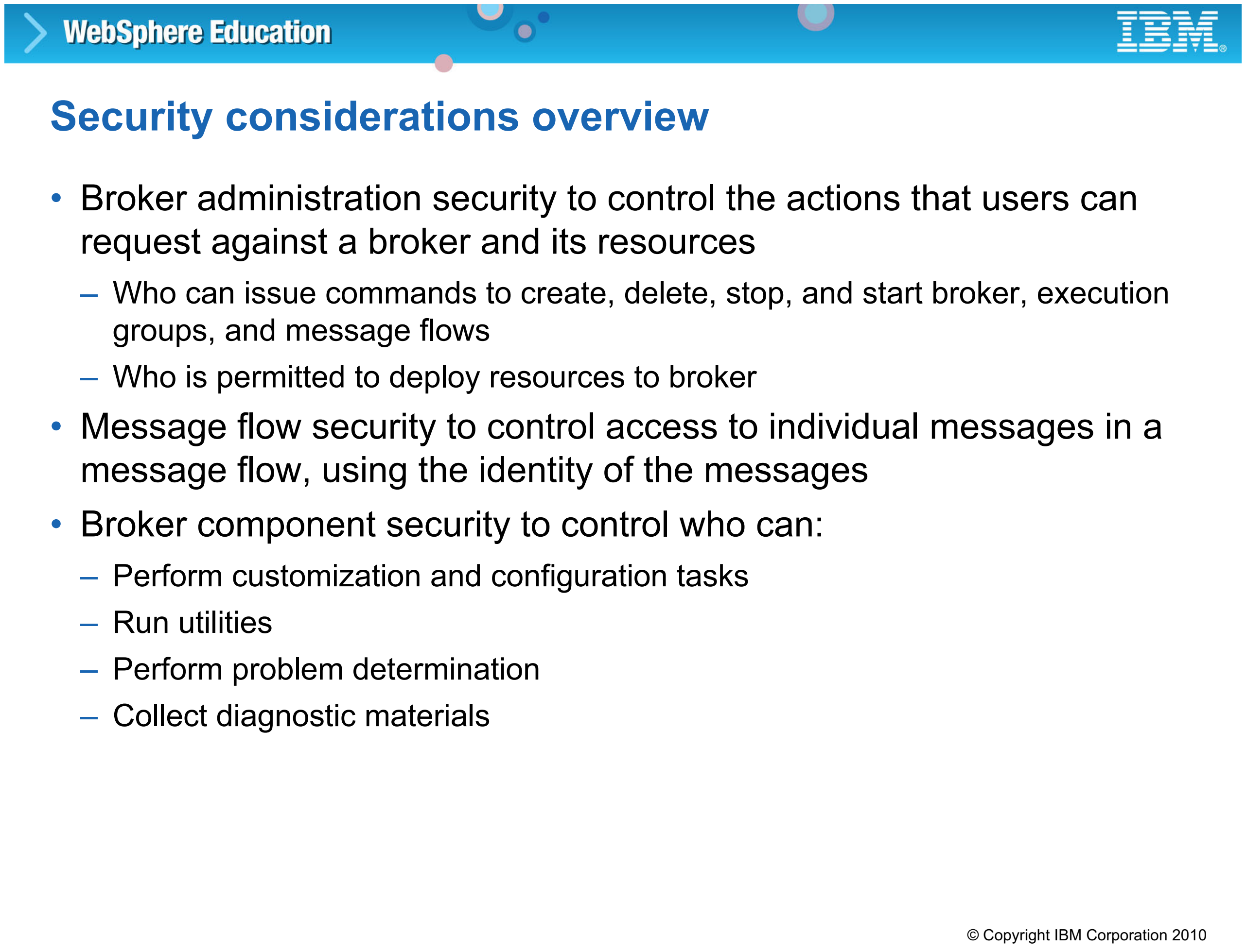
This topic is an overview of the security options available for WebSphere Message Broker. A detailed description of security implementation is provided in a dedicated unit later in this course.

What students will do — Students will learn about the implementation of various WebSphere Message Broker security features.

How students will do it — Students will listen to the lecture and participate in classroom discussion.

What students will learn — How to plan for broker security in the local and domain environments.

How this will help students in their job — Students will be able to plan security parameters in a local and domain environment. This will help them in planning and implementing a security infrastructure for their broker topology and help manage the applications running on those brokers.



WebSphere Education

Security considerations overview

- Broker administration security to control the actions that users can request against a broker and its resources
  - Who can issue commands to create, delete, stop, and start broker, execution groups, and message flows
  - Who is permitted to deploy resources to broker
- Message flow security to control access to individual messages in a message flow, using the identity of the messages
- Broker component security to control who can:
  - Perform customization and configuration tasks
  - Run utilities
  - Perform problem determination
  - Collect diagnostic materials

Figure 2-31. Security considerations overview WM643 / VM6431.0

Notes:

There are several aspects related to security within WebSphere Message Broker. There is security around the broker objects themselves, which controls who can access the broker and other broker resources. There is security as it relates to the individual message flows, which allows the ID associated with a message that starts a message flow to be authenticated, authorized, and propagated through the message flow. Other kinds of security considerations deal with Web services security, authorization for installation and configuration tasks, securing development resources, security exits, and other areas.

If you do not enable message flow security, the default WebSphere Message Broker facilities are based on transport-specific facilities. WebSphere Message Broker processes all messages that can be physically delivered to it, using the broker service ID as a proxy for all incoming messages. In this case, even if a specific user ID is present in an incoming message, it is ignored.

This course concentrates on domain security aspects. However, you should review the *Configuration, Administration, and Security* guide to familiarize yourself with all aspects of security administration.

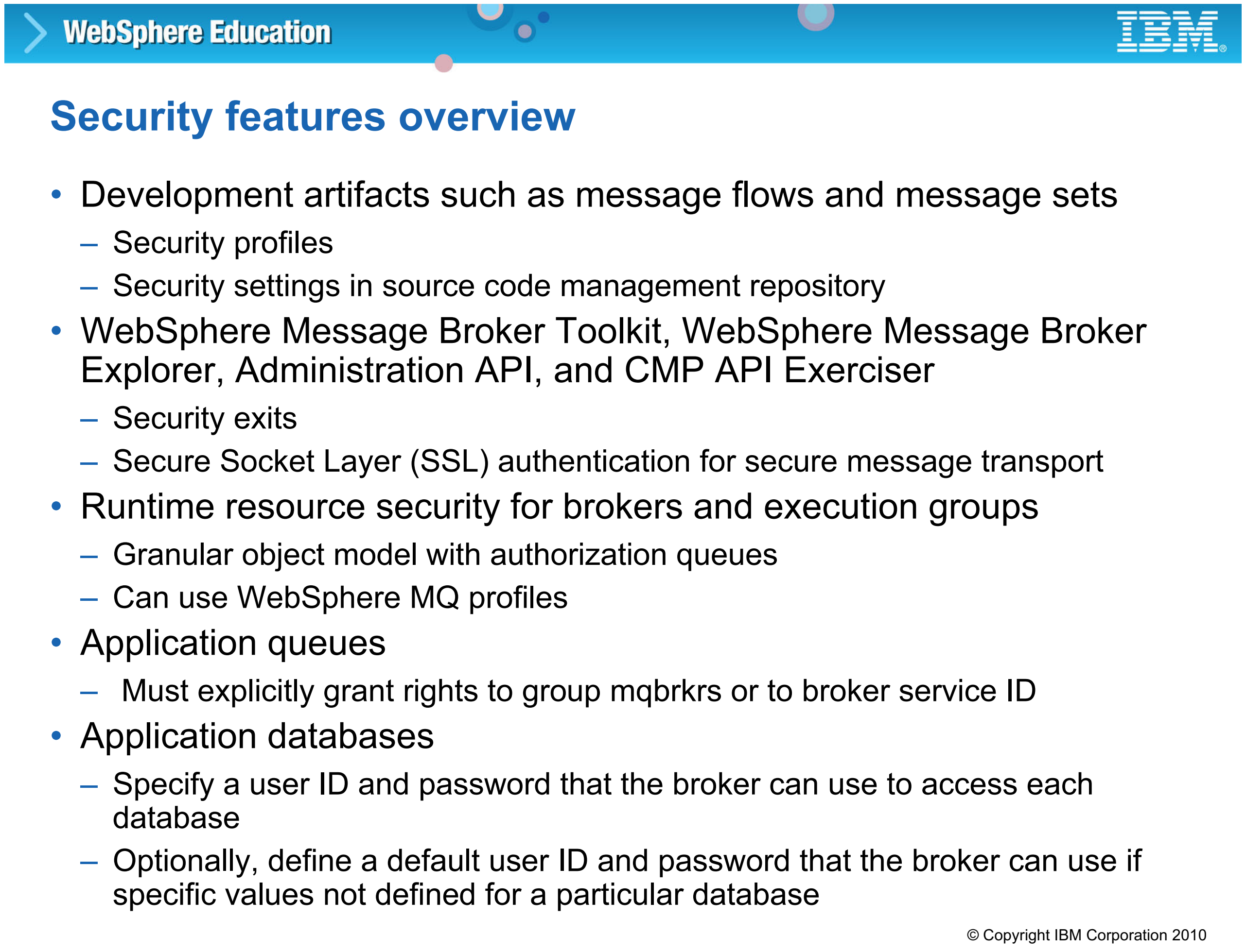
Instructor notes:

Purpose — Provide a frame of reference for the various types of security in which a WebSphere Message Broker administrator might be involved.

Details —

Additional information — Message flow security using LDAP or Tivoli Federated Identity Manager is not required. If they are not used, message flow security is handled at the message transport level.

Transition statement —



WebSphere Education

Security features overview

- Development artifacts such as message flows and message sets
  - Security profiles
  - Security settings in source code management repository
- WebSphere Message Broker Toolkit, WebSphere Message Broker Explorer, Administration API, and CMP API Exerciser
  - Security exits
  - Secure Socket Layer (SSL) authentication for secure message transport
- Runtime resource security for brokers and execution groups
  - Granular object model with authorization queues
  - Can use WebSphere MQ profiles
- Application queues
  - Must explicitly grant rights to group mqbrkrs or to broker service ID
- Application databases
  - Specify a user ID and password that the broker can use to access each database
  - Optionally, define a default user ID and password that the broker can use if specific values not defined for a particular database

Figure 2-32. Security features overview WM643 / VM6431.0

Notes:

Databases can be accessed with the component’s service ID or with a specific database ID and password.

The default message flow behavior is to access all user databases and all user queues using the broker service ID. The `mqsisetdbparms` command can be used to manage (set, reset, or remove) the user database (ODBC DSN) user ID and password pairs for a broker instance. This program can be run against a registered broker in the stopped state. It does not verify that the user database (ODBC DSN), the user ID, or the password exists, or that any of them are correct.

For security of the runtime configuration, you can choose between (or mix) two models:

- Authorize each object separately or
- Use group memberships that reflect the user’s role, such as developer or operator

If you are running in a heterogeneous environment, ensure that you limit all the principal IDs to eight characters or less.

Instructor notes:

Purpose — Provide some summary information about the various security features available with the WebSphere Message Brokers.

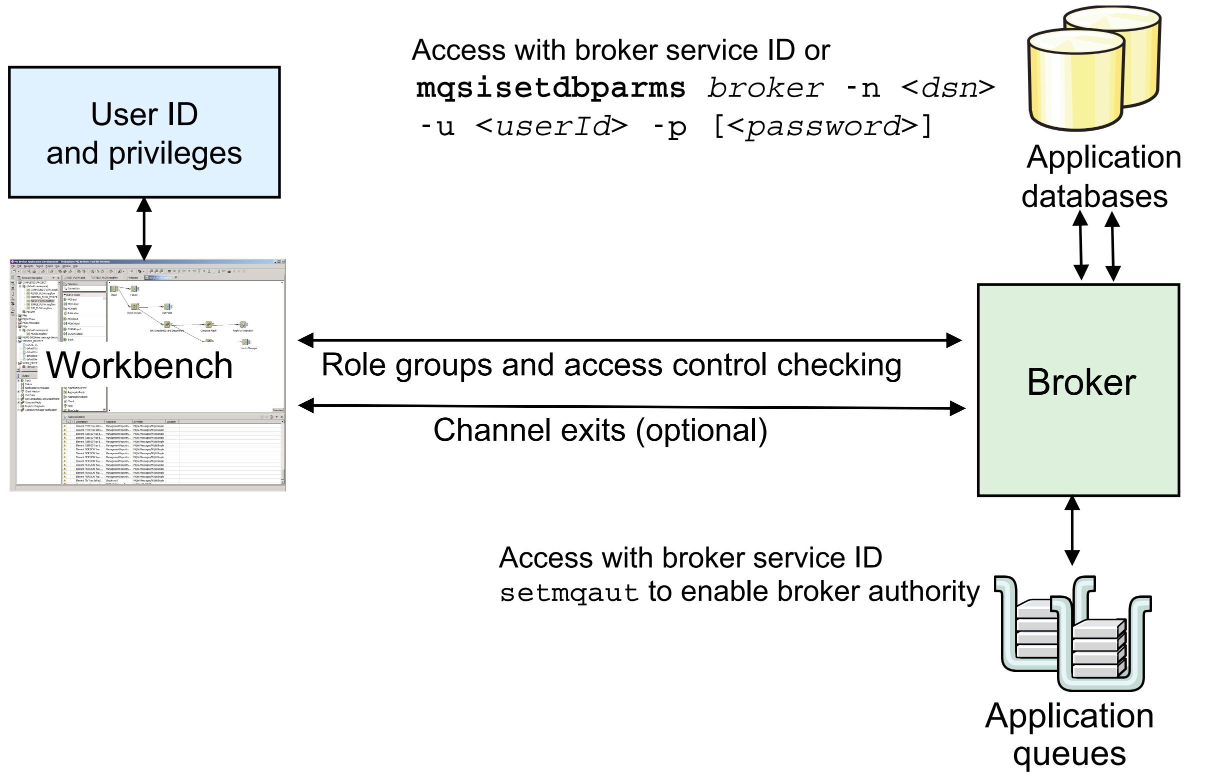
Details — None.

Additional information — None.

Transition statement — Security can be implemented at multiple levels.

WebSphere Education

Security checks



Access with broker service ID or
`mqsisetdbparms` *broker* `-n` *<dsn>*
`-u` *<userId>* `-p` [*<password>*]

Application databases

User ID and privileges

Workbench

Role groups and access control checking

Channel exits (optional)

Broker

Access with broker service ID
`setmqaut` to enable broker authority

Application queues

Figure 2-33. Security checks WM643 / VM6431.0

Notes:

Security in the WebSphere Message Broker environment involves multiple components, including the operating system, WebSphere MQ, and databases. In addition, you can implement WebSphere MQ channel security exits between the WebSphere Message Broker Toolkit and component queue managers. Secure Sockets Layer (SSL) security is a viable alternative in each case.

You can also grant or revoke authority to one or more groups or users to complete specific tasks against a broker running on Linux, UNIX, or Windows. You can use WebSphere MQ commands to set up and manage your required security levels. If you prefer, you can also make authorization changes to the security queues by using WebSphere MQ Explorer.

For security reasons, it is important that authorities are set correctly. The `setmqaut` command grants and revokes authorities cumulatively. To avoid retaining unwanted pre-existing authorities, set authorities explicitly on each `setmqaut` command, rather than granting and revoking individual authorities.

Security between the broker and application databases allows flows running on the broker to read or write to the databases as required in the flow.

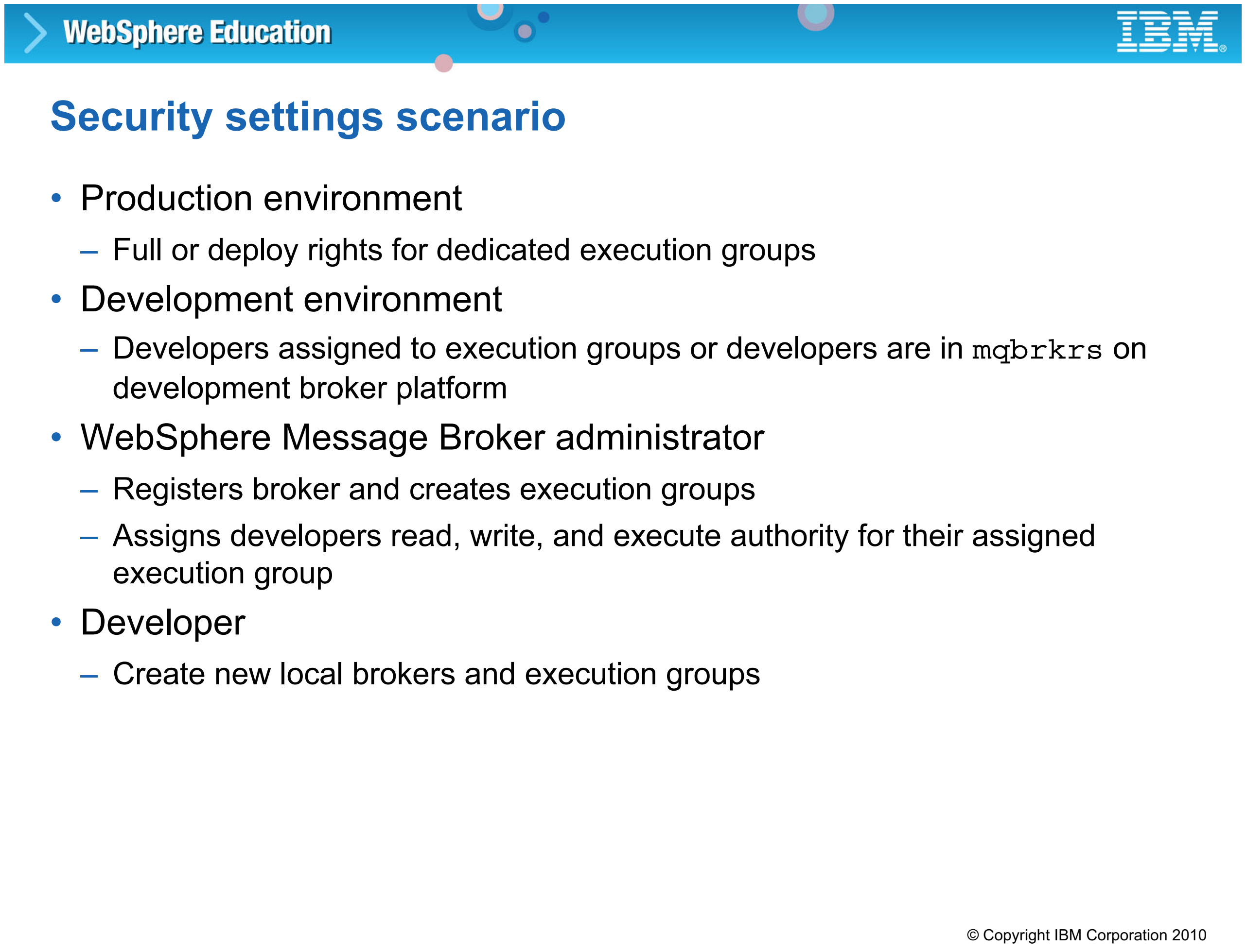
Instructor notes:

Purpose — Give an overview of the various security checks that can be performed in the broker environment.

Details — None.

Additional information — Security in the WebSphere Message Brokers equates to access control.

Transition statement — A number of security features are identified in this figure. Next you look at an example of how security can be implemented.



WebSphere Education
IBM

Security settings scenario

- Production environment
  - Full or deploy rights for dedicated execution groups
- Development environment
  - Developers assigned to execution groups or developers are in `mqbrkrs` on development broker platform
- WebSphere Message Broker administrator
  - Registers broker and creates execution groups
  - Assigns developers read, write, and execute authority for their assigned execution group
- Developer
  - Create new local brokers and execution groups

Figure 2-34. Security settings scenario WM643 / VM6431.0

Notes:

The figure identifies the groups of users and some suggestions as to how best to use the various security authorizations depending on a user’s job requirements.

For example, in a development environment, you should allow developer’s to deploy to and control development brokers and execution groups. In a production environment, however, access to brokers should be limited to administrators.

Instructor notes:

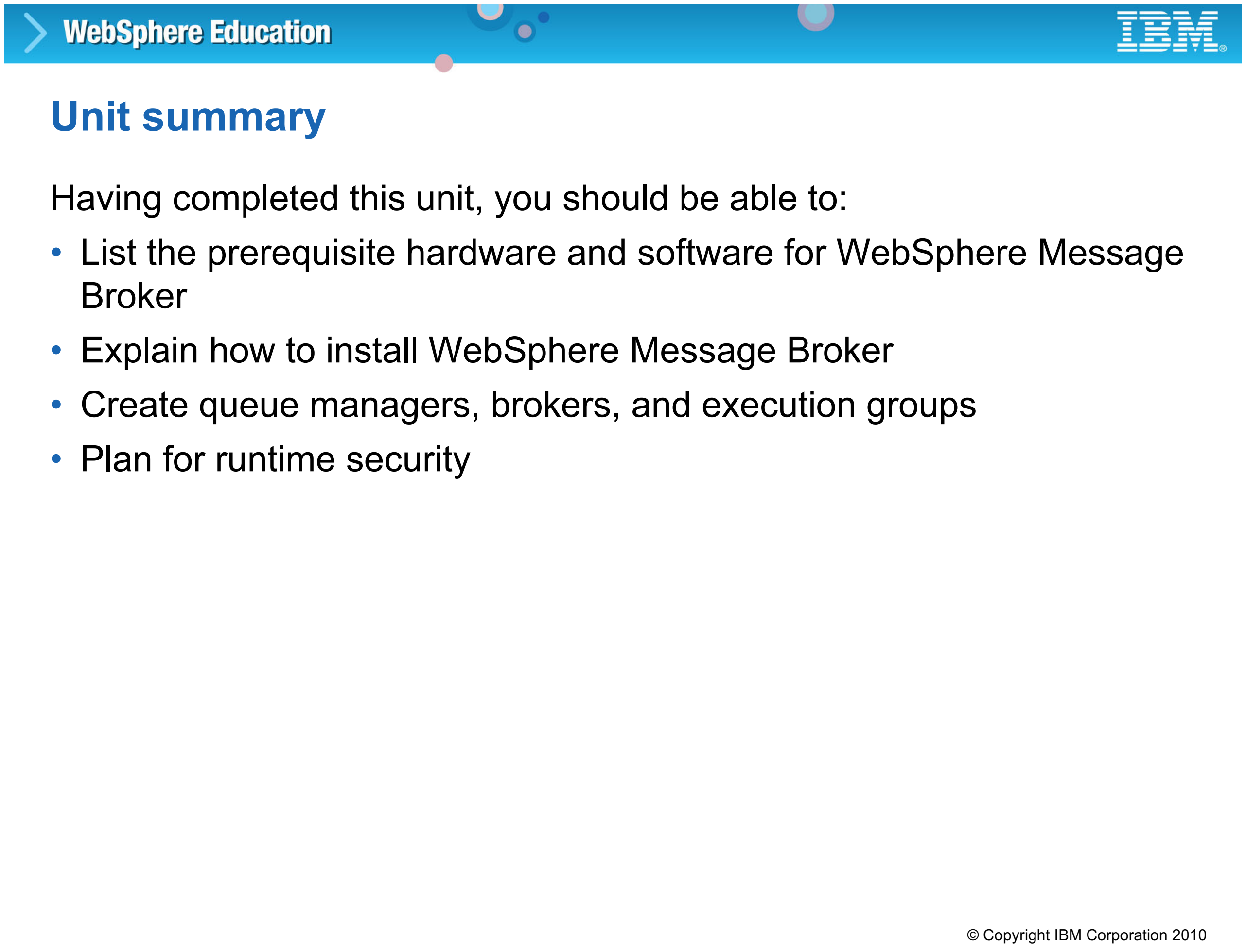
Purpose — Show an example of how security could be implemented.

Details — None.

Additional information — Administrators can control how developers use broker archive files to deploy message flows and message sets by building groups to grant specific deployment permission to a specific broker or execution group instance based on individual developer user IDs.

The `mqbrdevt` group, although considered obsolete from the version 6 product point of view, can still be used to provide authorizations to developers

Transition statement —



WebSphere Education

Unit summary

Having completed this unit, you should be able to:

- List the prerequisite hardware and software for WebSphere Message Broker
- Explain how to install WebSphere Message Broker
- Create queue managers, brokers, and execution groups
- Plan for runtime security

Figure 2-35. Unit summary WM643 / VM6431.0

Notes:

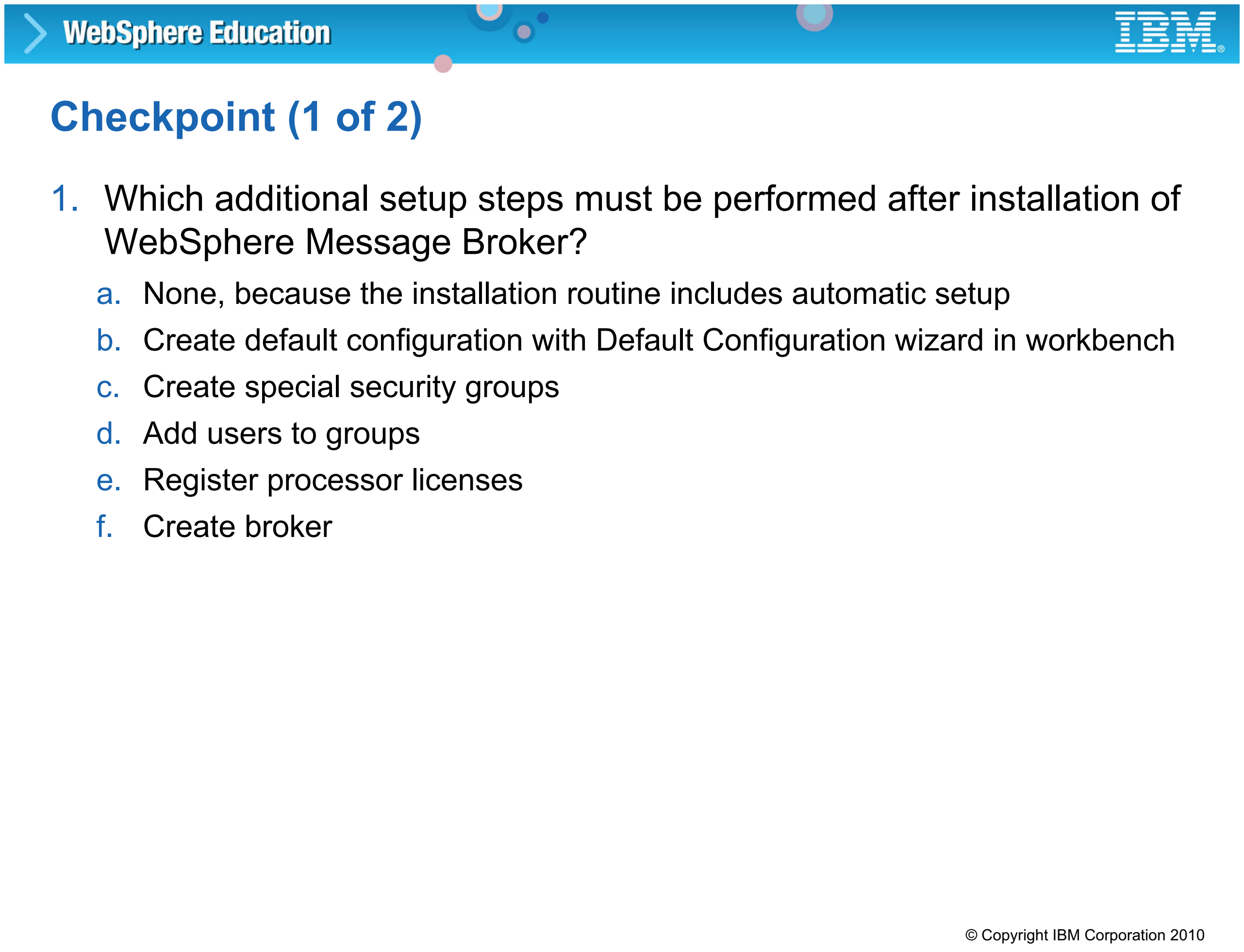
Instructor notes:

Purpose — Complete the unit with a summary of the salient points.

Details — None.

Additional information — None.

Transition statement — Now you will put what you have learned into practice by completing the Checkpoints and Exercise 1. In the next unit, you will use the WebSphere Message Broker Toolkit so that you have a good understanding of the WebSphere Message Broker concepts and the tasks performed by a developer.



WebSphere Education

Checkpoint (1 of 2)

1. Which additional setup steps must be performed after installation of WebSphere Message Broker?
   a. None, because the installation routine includes automatic setup
   b. Create default configuration with Default Configuration wizard in workbench
   c. Create special security groups
   d. Add users to groups
   e. Register processor licenses
   f. Create broker

© Copyright IBM Corporation 2010

Figure 2-36. Checkpoint (1 of 2)

WM643 / VM6431.0

Notes:

Write your answer here:

1.

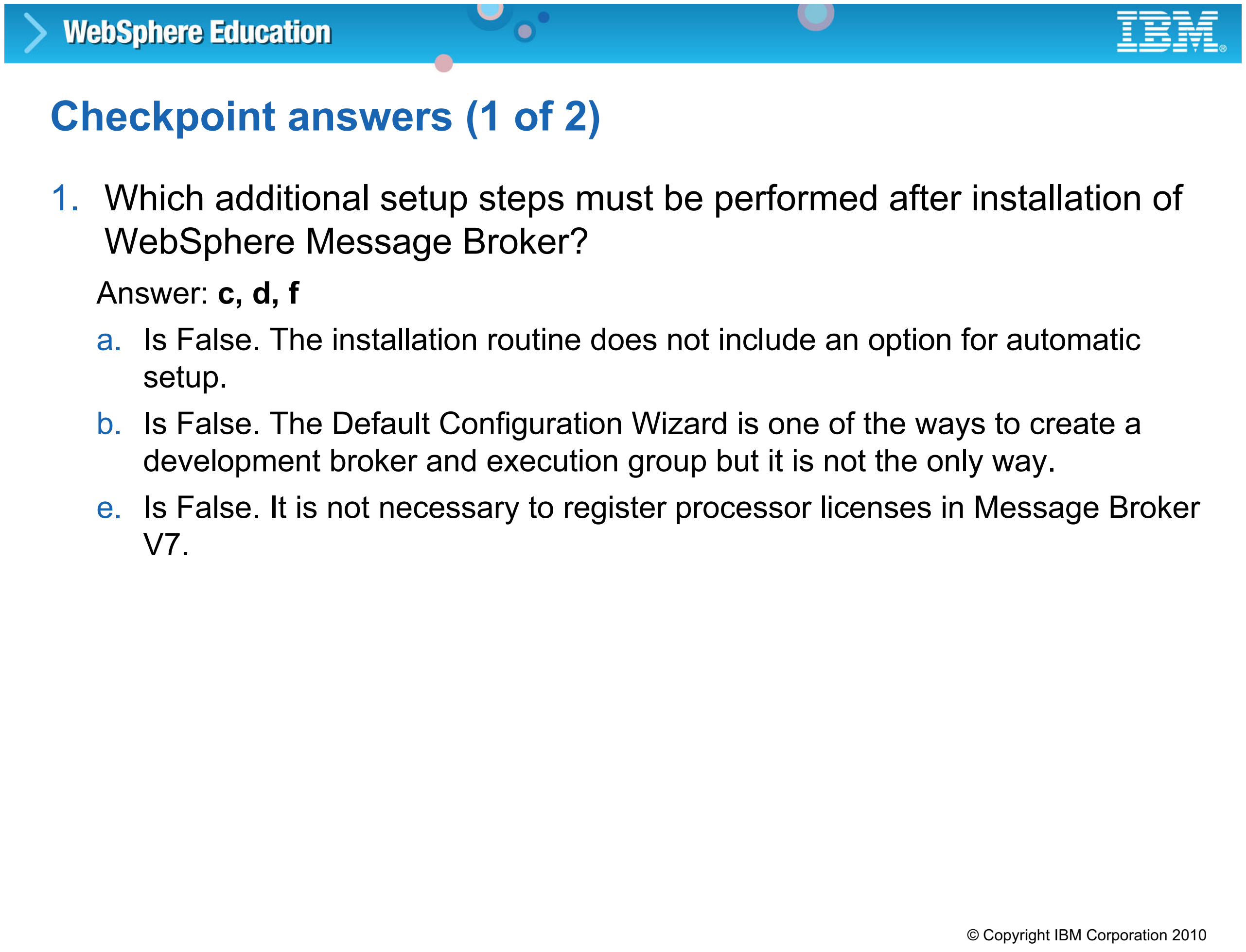
Instructor notes:

Purpose — Let students work on checkpoint questions, either individually or in teams.

Details — Checkpoint answers are on the next page.

Additional information — Discuss each question and answer.

Transition statement — Here are the answers, in case you missed something during the discussion.



WebSphere Education

Checkpoint answers (1 of 2)

1. Which additional setup steps must be performed after installation of WebSphere Message Broker?

Answer: **c, d, f**

a. Is False. The installation routine does not include an option for automatic setup.

b. Is False. The Default Configuration Wizard is one of the ways to create a development broker and execution group but it is not the only way.

e. Is False. It is not necessary to register processor licenses in Message Broker V7.

Figure 2-37. Checkpoint answers (1 of 2)

WM643 / VM6431.0

Notes:

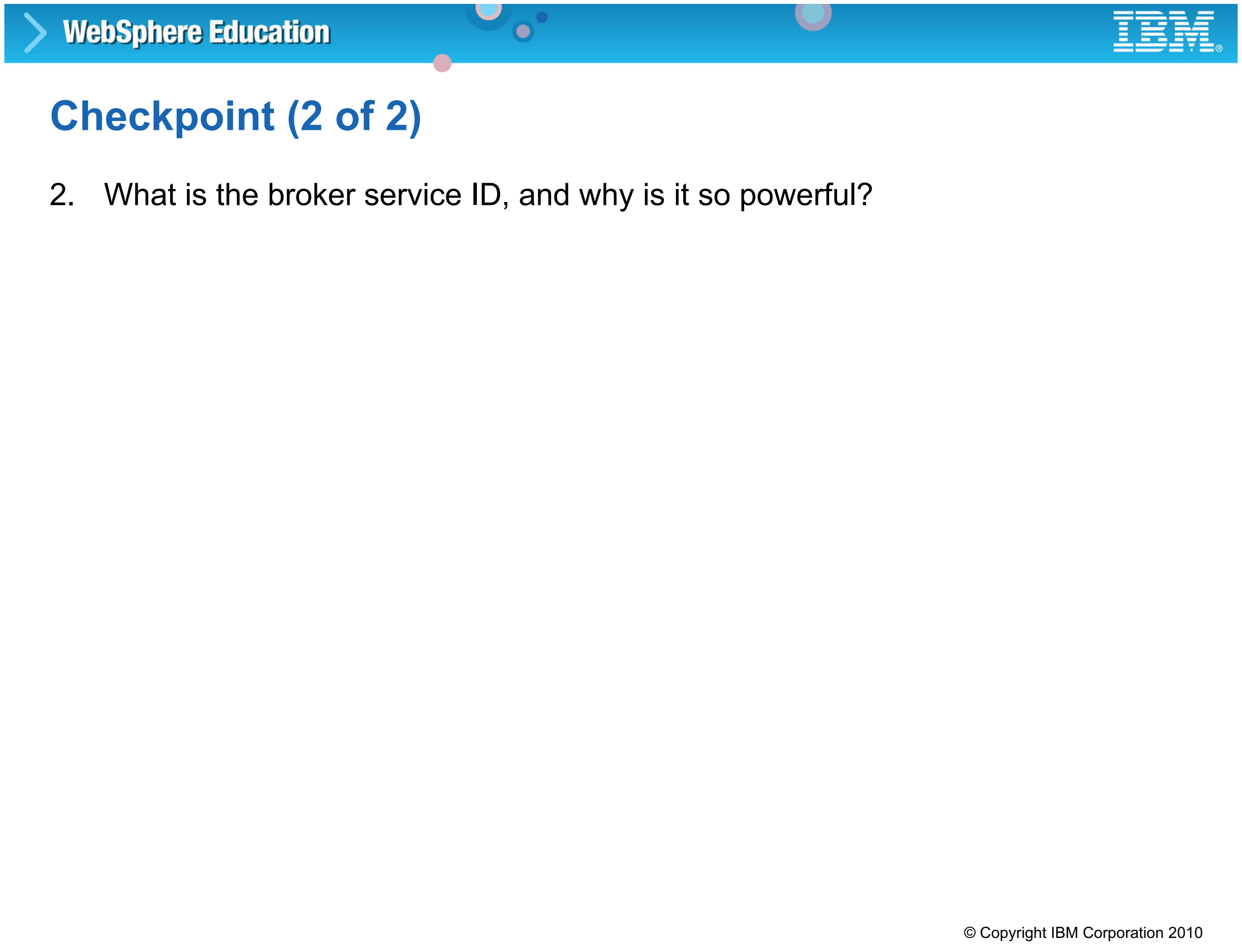
Instructor notes:

Purpose — Present checkpoint answers

Details —

Additional information —

Transition statement —



WebSphere Education
IBM

Checkpoint (2 of 2)

2. What is the broker service ID, and why is it so powerful?

Figure 2-38. Checkpoint (2 of 2)

WM643 / VM6431.0

Notes:

Write your answer here:

2.

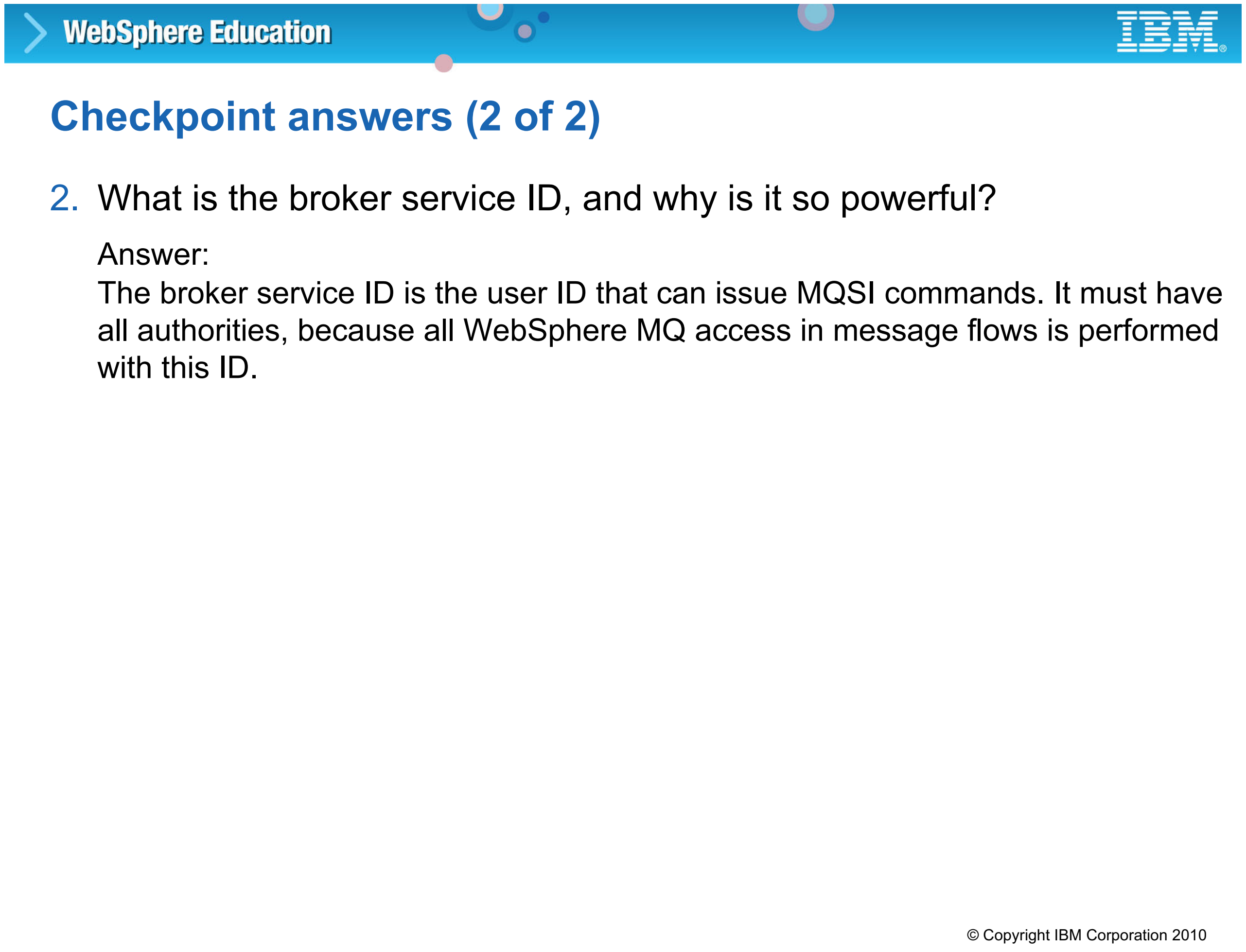
Instructor notes:

Purpose — Let students work on checkpoint questions, either individually or in teams.

Details —

Additional information —

Transition statement —



WebSphere Education

Checkpoint answers (2 of 2)

2. What is the broker service ID, and why is it so powerful?

   Answer:
   The broker service ID is the user ID that can issue MQSI commands. It must have all authorities, because all WebSphere MQ access in message flows is performed with this ID.

Figure 2-39. Checkpoint answers (2 of 2) WM643 / VM6431.0

Notes:

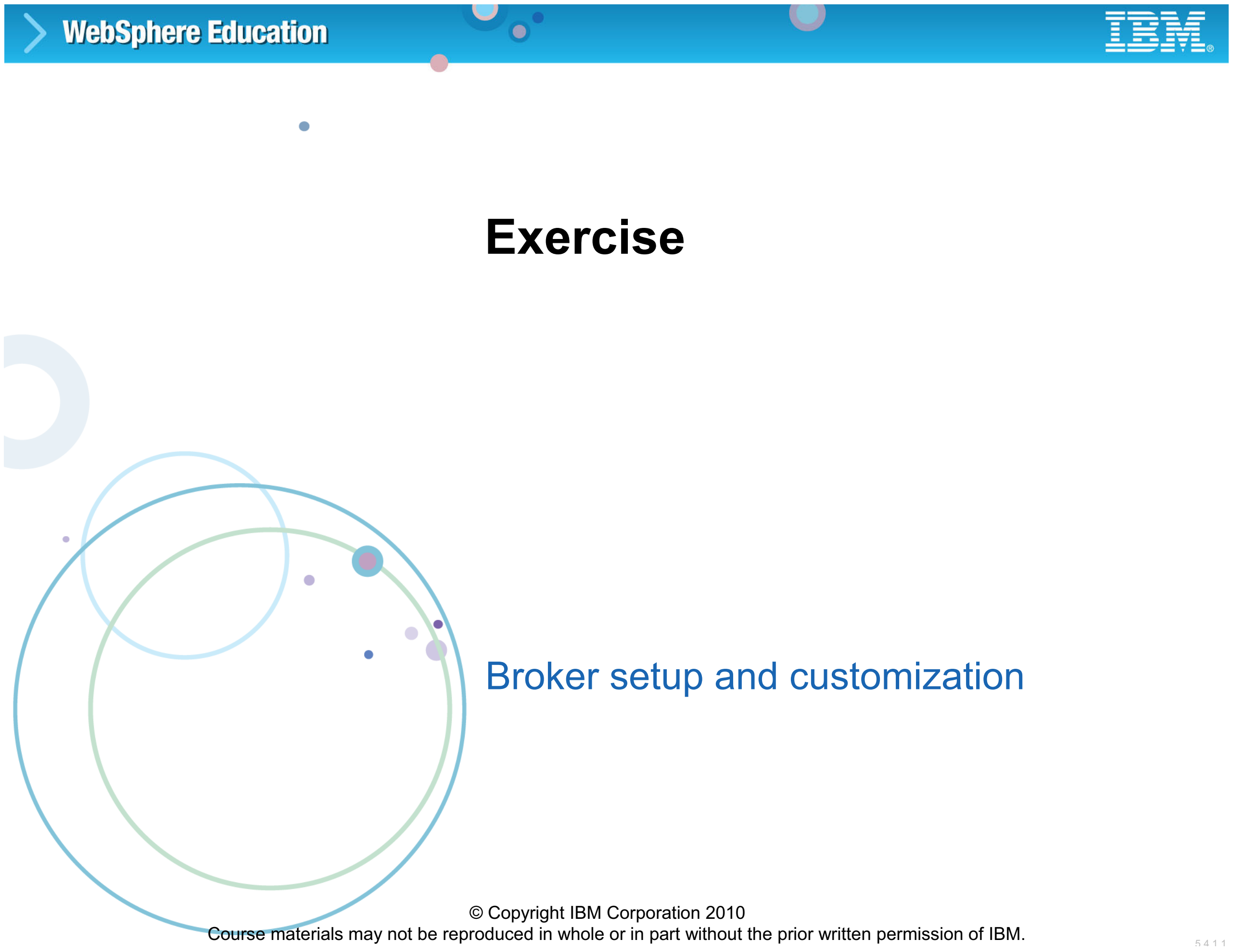
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —



WebSphere Education

IBM

Exercise

Broker setup and customization

5411

Figure 2-40. Exercise 1: Broker setup and customization WM643 / VM6431.0

Notes:

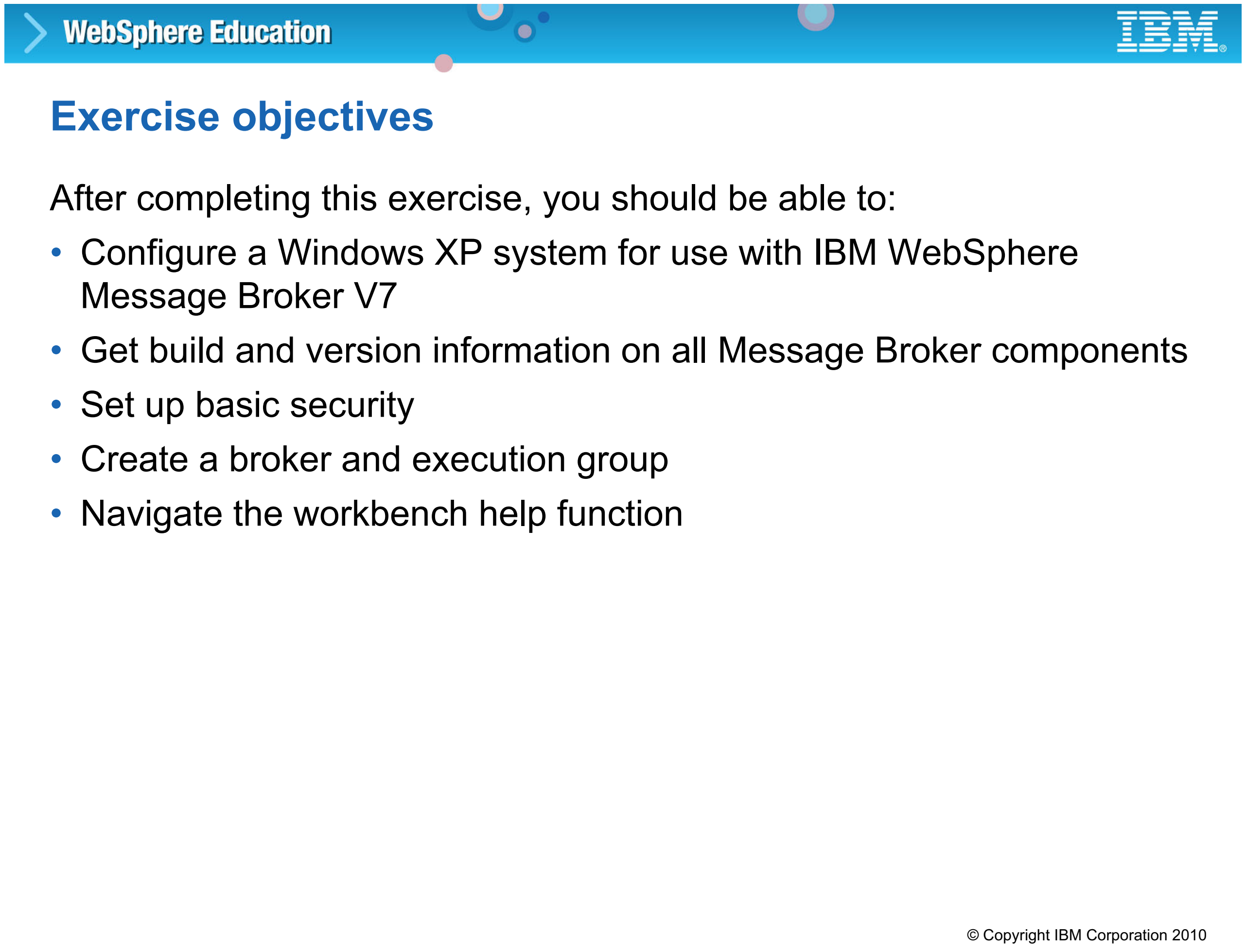
Instructor notes:

Purpose — Have students complete the exercise.

Details — This exercise has students set up WebSphere Message Broker, start a workbench instance, and use the help function to access the Information Center.

Additional information —

Transition statement —



WebSphere Education

Exercise objectives

After completing this exercise, you should be able to:

- Configure a Windows XP system for use with IBM WebSphere Message Broker V7
- Get build and version information on all Message Broker components
- Set up basic security
- Create a broker and execution group
- Navigate the workbench help function

Figure 2-41. Exercise objectives WM643 / VM6431.0

Notes:

See the *Student Exercises Guide* for detailed instructions.

Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —

Unit 3. Using the WebSphere Message Broker Toolkit

Estimated time

01:00 lecture, 01:00 exercise

What this unit is about

This unit describes the basic administrative tasks that can be performed using the WebSphere Message Broker Toolkit.

What you should be able to do

After completing this unit, using only the toolkit, you should be able to:

- Explain the function of the WebSphere Message Broker Toolkit
- Use the WebSphere Message Broker Toolkit to:
  - Configure the Message Broker development environment
  - Import a simple message flow
  - Deploy a broker archive file containing a message flow to a broker
- Test a deployed message flow

How you will check your progress

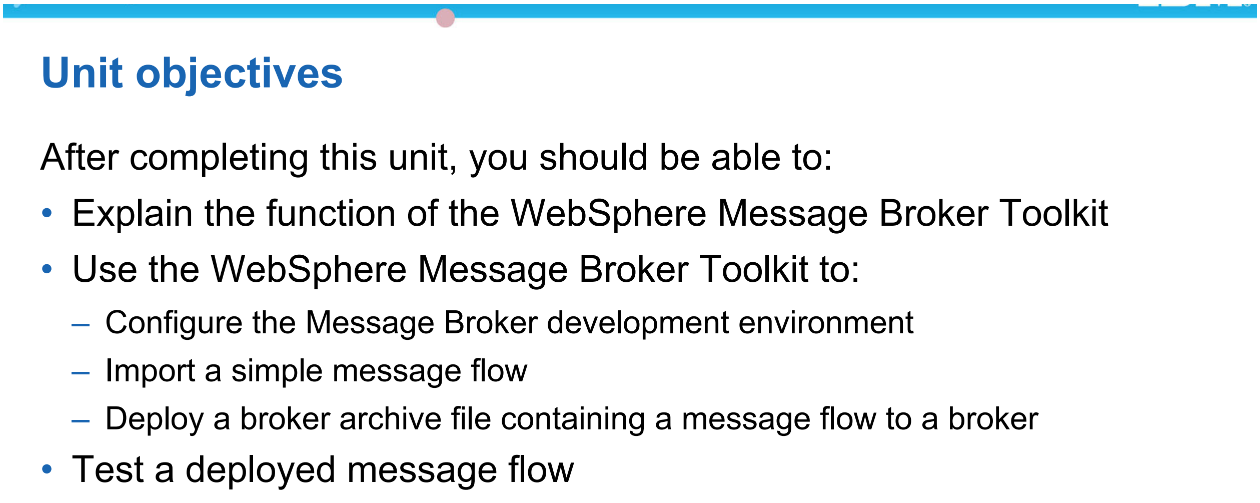
Accountability:

- Review checkpoint questions
- Exercise 2: Using the WebSphere Message Broker Toolkit

References

`www.ibm.com/software/integration/wbimessagebroker/library`
IBM WebSphere Message Broker Library

`www.ibm.com/software/integration/wmq/library`
IBM *WebSphere MQ Library*



Unit objectives

After completing this unit, you should be able to:

- Explain the function of the WebSphere Message Broker Toolkit
- Use the WebSphere Message Broker Toolkit to:
  - Configure the Message Broker development environment
  - Import a simple message flow
  - Deploy a broker archive file containing a message flow to a broker
- Test a deployed message flow

Figure 3-1. Unit objectives WM643 / VM6431.0

Notes:

The WebSphere Message Broker Toolkit, sometimes referred to as the “workbench”, is an integrated development and administrative environment.

Using the WebSphere Message Broker Toolkit developers can create, deploy, test, and debug message flows and message sets.

This unit explores the functions of the WebSphere Message Broker Toolkit. If you require in-depth training on developing and testing message flow, see course WM663/VM663, *IBM WebSphere Message Broker V7.0 Developer Workshop*.

Instructor notes:

Purpose — This unit explains the functions of the WebSphere Message Broker Toolkit. Although many administrators likely are not developing message flows (such as a developer would), understanding how the toolkit functions is important for administrators to understand.

Details — None.

Additional information — This unit is basic training for administrators. Most of the *fundamental* toolkit navigation and usage functions are covered here.

Transition statement — Now you will spend a few minutes in an overview to understand the concepts.

3.1. WebSphere Message Broker Toolkit concepts

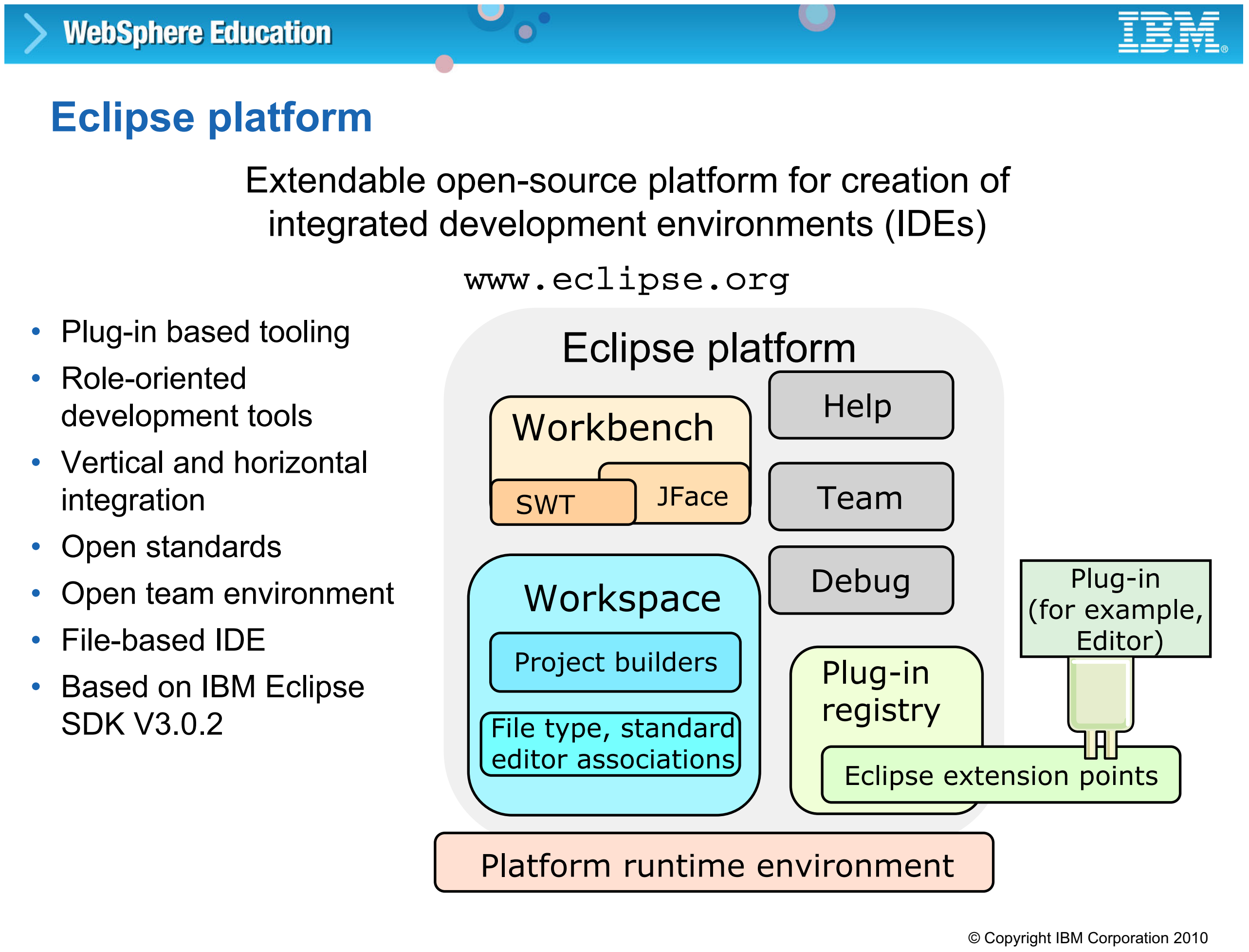
Instructor topic introduction

What students will do — Learn about the WebSphere Message Broker Toolkit to become familiar with the concepts.

How students will do it — Listen to a lecture and participate in classroom discussion.

What students will learn — They will learn the high-level information necessary to prepare them to use the WebSphere Message Broker Toolkit.

How this will help students on their job — By understanding the basic concepts, the use of the WebSphere Message Broker Toolkit will be easier for them.



WebSphere Education

Eclipse platform

Extendable open-source platform for creation of integrated development environments (IDEs)

`www.eclipse.org`

- Plug-in based tooling
- Role-oriented development tools
- Vertical and horizontal integration
- Open standards
- Open team environment
- File-based IDE
- Based on IBM Eclipse SDK V3.0.2

Eclipse platform

Workbench

SWT

JFace

Help

Team

Debug

Workspace

Project builders

File type, standard editor associations

Plug-in registry

Plug-in (for example, Editor)

Eclipse extension points

Platform runtime environment

Figure 3-2. Eclipse platform

Notes:

The WebSphere Message Broker Toolkit provides an integrated development environment to perform application development and broker administration tasks for brokers. It is based on Eclipse V3.0.2 (formerly known as the WebSphere Studio Workbench). This development environment is used as a common tool across most of the WebSphere products.

The Eclipse platform provides a core architecture that can be enhanced and extended by tool vendors or open source projects using plug-ins. The standard Eclipse platform includes plug-ins for team programming support, the workbench, standard user interface controls, Java development, the help subsystems, and others. There are now over 30 tool vendors providing Eclipse tools in such areas as change management, project management, languages, and graphics support.

The Eclipse website `www.eclipse.org` has more information about Eclipse and available plug-ins.

Instructor notes:

Purpose — Introduce Eclipse and Toolkit

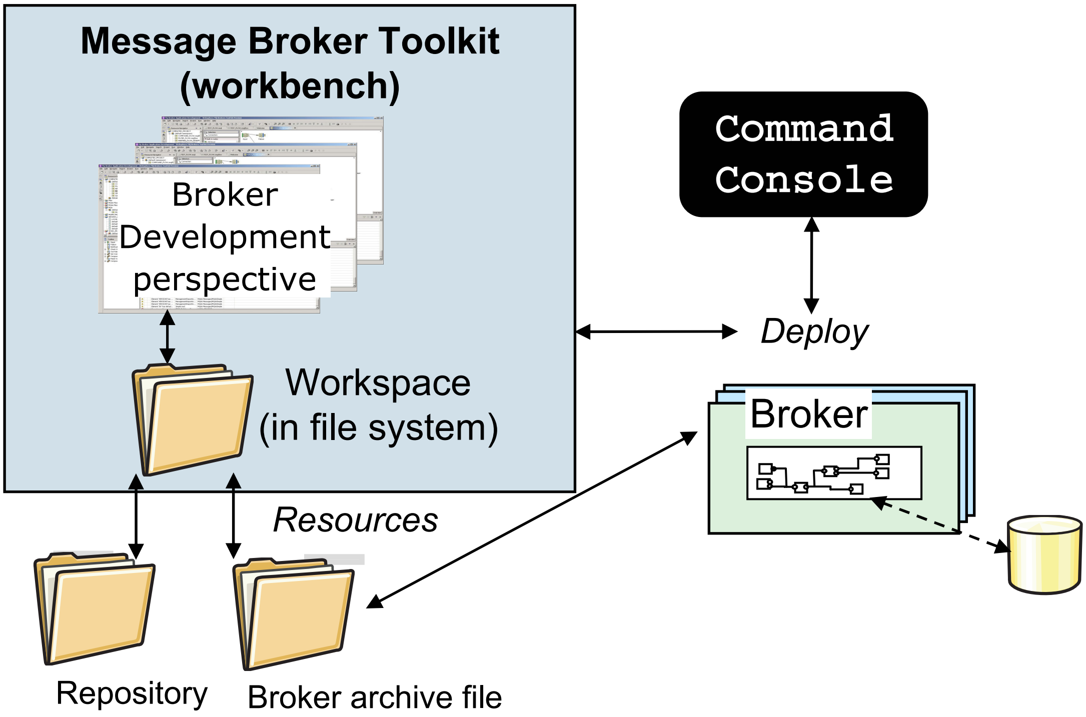
Details — None.

Additional information — The Eclipse platform is an open source project which provides an open platform for tool integration. The Eclipse project was started in November 2001 when IBM donated $40 million of tool development products based on the VisualAge products to start the project.

Transition statement — So how does the WebSphere Message Broker Toolkit work with the other components?

WebSphere Education

Development and runtime environment


Message Broker Toolkit (workbench)
Broker Development perspective
Workspace (in file system)
Command Console
Deploy
Broker
Resources
Repository
Broker archive file

© Copyright IBM Corporation 2010

Figure 3-3. Development and runtime environment WM643 / VM6431.0

Notes:

The WebSphere Message Broker Toolkit consists of a workbench window which displays one or more perspectives. A perspective is a group of views and editors required to perform tasks associated with a role.

The projects, folders, and files that exist in the workbench are called *resources*. They are collectively called the *workspace*. The workspace is a private work area for the individual developer. It holds the environment metadata and the loaded projects' data in regular files. Every time you save a resource, the changes are reflected in the file system. Normally this is your local file system, but you can choose to store your workspace in a network file system (NFS) as well.

To run any developed resource, it must be packaged into a *broker archive file*. Then that broker archive file must be deployed to the broker.

An external source code management system (SCM) can be used as a repository for development artifacts.

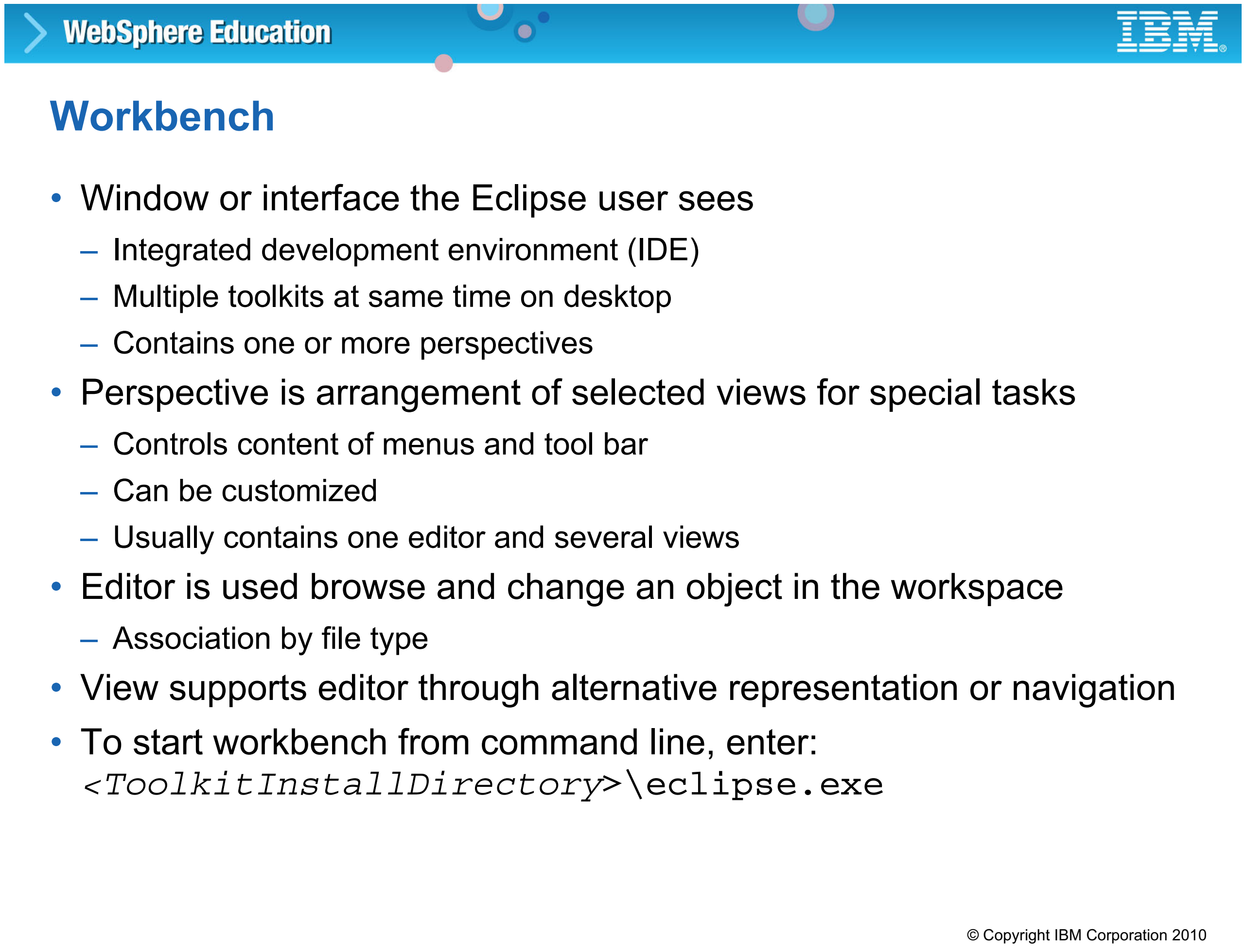
Instructor notes:

Purpose — Outline connections between components and interfaces.

Details — None.

Additional information — None.

Transition statement — What exactly is a workbench?



WebSphere Education
IBM

Workbench

- Window or interface the Eclipse user sees
  - Integrated development environment (IDE)
  - Multiple toolkits at same time on desktop
  - Contains one or more perspectives
- Perspective is arrangement of selected views for special tasks
  - Controls content of menus and tool bar
  - Can be customized
  - Usually contains one editor and several views
- Editor is used browse and change an object in the workspace
  - Association by file type
- View supports editor through alternative representation or navigation
- To start workbench from command line, enter:
  `<ToolkitInstallDirectory>\eclipse.exe`

Figure 3-4. Workbench WM643 / VM6431.0

Notes:

A workbench is an instance (session) of the WebSphere Message Broker Toolkit. A workbench window can have several separate perspectives, only one of which is visible at any given moment.

You also can use the WebSphere Message Broker Toolkit to open more than one workbench at a time. To open a new workbench in another window, click **Window** > **New Window** and a new workbench with the same perspective opens in a new window.

Each perspective has its own views and editors that are arranged (tiled, stacked, or detached) for presentation on the screen (some might be hidden at any given moment). Several different types of views and editors can be open at the same time within a perspective.

A perspective controls initial view visibility, layout, and action visibility. The user can quickly switch perspectives to work on a different task, and can easily rearrange and customize a perspective to better suit a particular task. Customized perspectives can be saved as new perspectives if required.

Instructor notes:

Purpose — Explain the concept of the workbench

Details —

Additional information — The platform provides standard perspectives for general resource navigation, online help, and team support tasks. Additional perspectives are supplied by other plug-ins.

Editors allow the user to open, edit, and save objects. They follow an open-save-close life cycle much like file system-based tools, but are more tightly integrated into the workbench. When active, an editor can contribute actions to the workbench menus and toolbar. The platform provides a standard editor for text resources. Other editors are supplied by plug-ins or applications.

Views provide information about some object that the user is working with in the workbench. A view can assist an editor by providing information about the document being edited.

For example, the standard content outline view shows a structured outline for the content of the active editor if one is available. A view can augment other views by providing information about the currently selected object. For example, the standard properties view presents the properties of the object selected in another view.

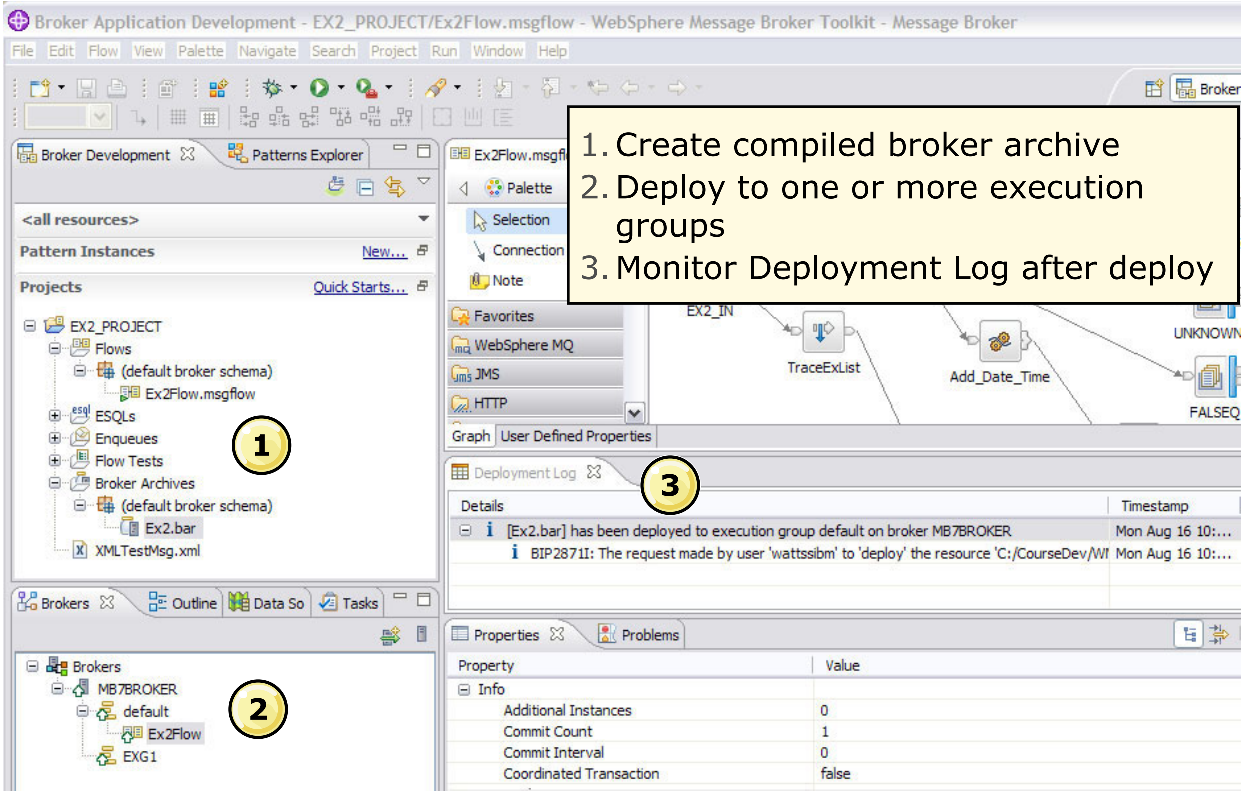
Views have a simpler life cycle than editors. Modifications made in a view (such as changing a property value) are generally saved immediately, and the changes are reflected immediately in other related parts of the user interface.

The platform provides several standard views. Additional views are supplied by other plug-ins.

Transition statement — The Broker Application Development perspective provides some basic administration capabilities.

WebSphere Education

Broker Application Development perspective



1. Create compiled broker archive
2. Deploy to one or more execution groups
3. Monitor Deployment Log after deploy

```
[Ex2.bar] has been deployed to execution group default on broker MB7BROKER    Mon Aug 16 10:...
BIP2871I: The request made by user 'wattssibm' to 'deploy' the resource 'C:/CourseDev/WI   Mon Aug 16 10:...
```

Property	Value
Info	
Additional Instances	0
Commit Count	1
Commit Interval	0
Coordinated Transaction	false

© Copyright IBM Corporation 2010

Figure 3-5. Broker Application Development perspective WM643 / VM6431.0

Notes:

The Broker Application Development perspective supports some basic administration functions such as creating local brokers and execution groups, creating broker archive files, deploying broker archive files, and monitoring deployment activities.

Many of the broker administration functions available in the WebSphere Message Broker Toolkit are also available as command-line utilities. Scripting commands can be used to automate operation for such tasks as deployment and configuration.

Many additional perspectives are available for specific tasks, for example Debug, Java, plug-in Development, or CVS Repository.

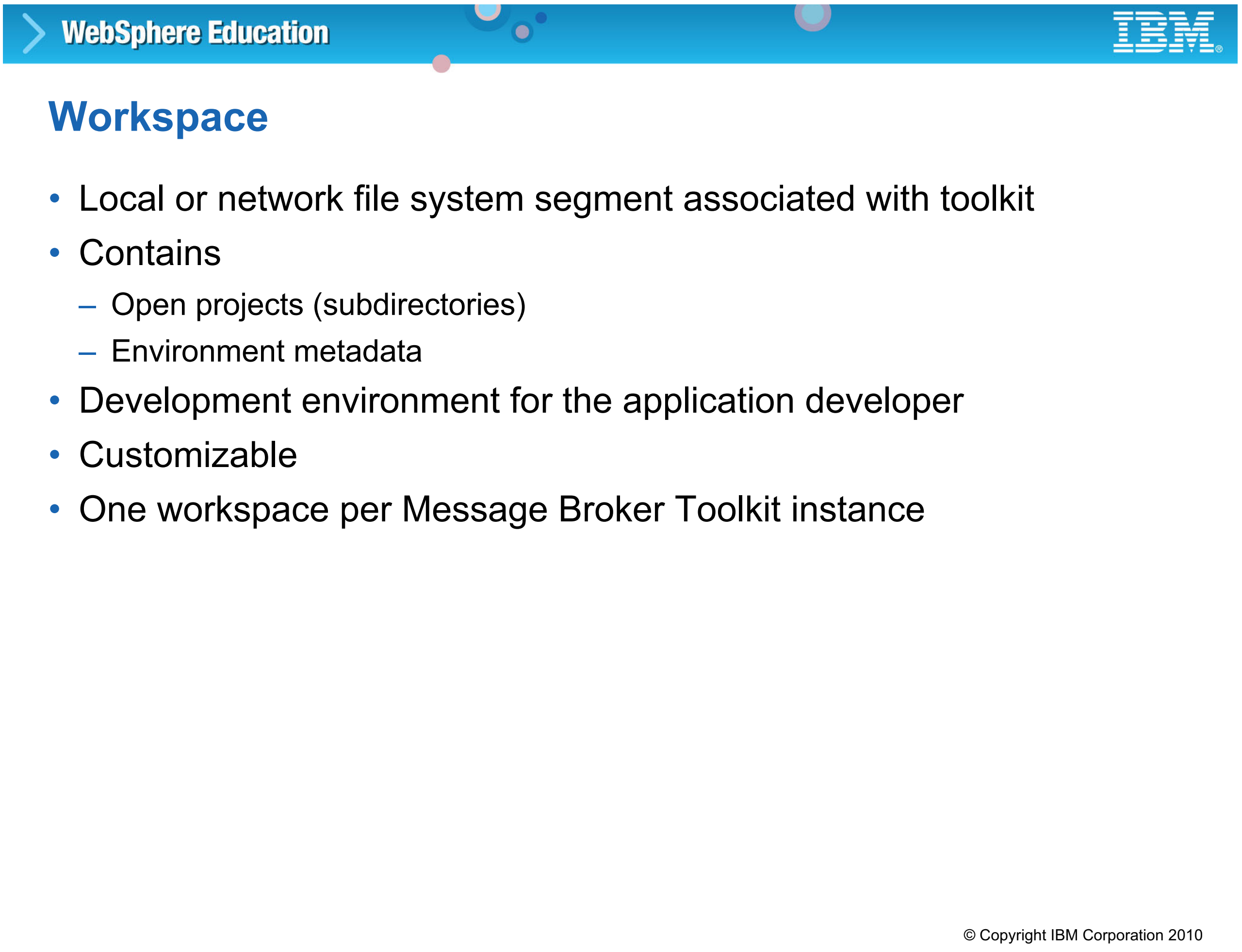
Instructor notes:

Purpose — Introduce perspectives.

Details — Emphasize the fact that even though the students role is that of an administrator, they might not always stick to a single perspective. For instance, they can add the WebSphere MQ Explorer perspective and check on queues or use the Debug perspective to debug a message flow. The latter of these are discussed further in Unit 6, “Monitoring and problem determination”. Mention additional perspectives that are available in the WebSphere Message Broker Toolkit.

Additional information — A main problem for students is that they are looking for a resource or a menu entry and cannot find it, simply because they are in the wrong perspective. The best thing is to demonstrate this behavior.

Transition statement — Next, you will look at workspaces.



WebSphere Education

Workspace

- Local or network file system segment associated with toolkit
- Contains
  - Open projects (subdirectories)
  - Environment metadata
- Development environment for the application developer
- Customizable
- One workspace per Message Broker Toolkit instance

Figure 3-6. Workspace

WM643 / VM6431.0

Notes:

The projects, folders, and files that exist in the workbench are called resources. They are collectively called the workspace and they reside in your local file system.

Instructor notes:

Purpose — Explain the workspace concept.

Details — Do a live demonstration.

Additional information — It is essential that students understand the difference between workbench and workspace, and the relation between projects and the file system.

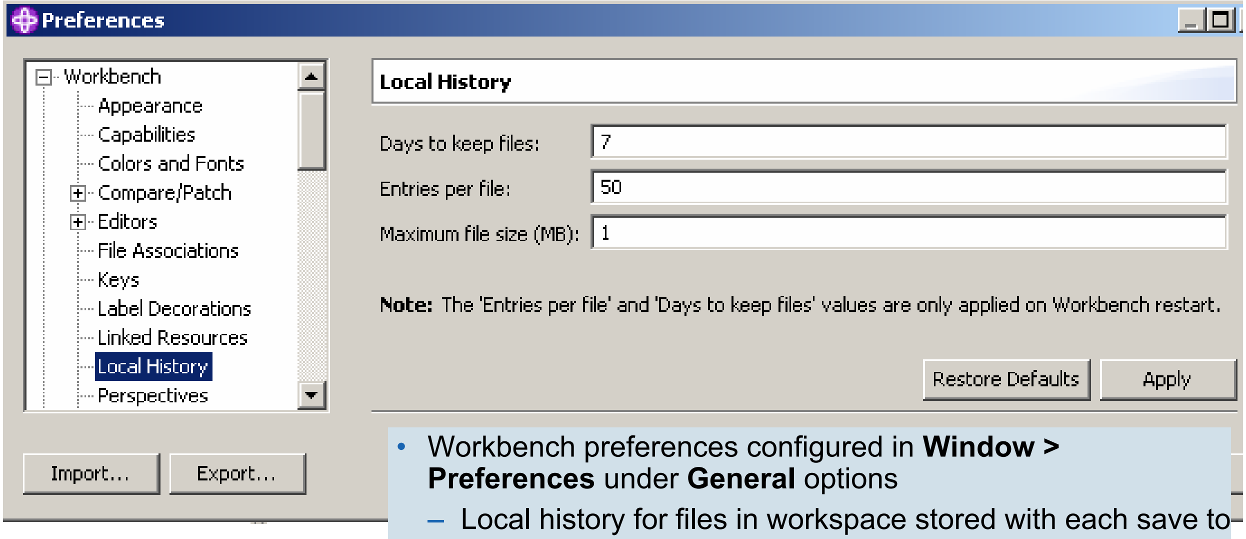
An Eclipse novice might not absorb all the details of what you can do (customize) in the workbench. It is the conceptual knowledge that matters. The exercises later in the course provide enough opportunity and detailed instructions to navigate and operate in the workbench.

Emphasize that in the workbench, there are often at least two alternative ways to achieve the same result. You need practice to find “your way” to do things.

Transition statement — The workspace is highly customizable.

WebSphere Education

Customizing the workspace



Preferences

Local History

Days to keep files: 7

Entries per file: 50

Maximum file size (MB): 1

Note: The 'Entries per file' and 'Days to keep files' values are only applied on Workbench restart.

- Workbench preferences configured in **Window > Preferences** under **General** options
  - Local history for files in workspace stored with each save to allow for comparisons
  - Perspectives
  - Short cut keys
- All options can be reset to default value

Figure 3-7. Customizing the workspace WM643 / VM6431.0

Notes:

Almost every aspect of the workbench is customizable in the **Preferences** menu. Preferences are stored in your workspace and can be imported and exported.

You can also customize the views in every perspective and store your own, named perspective.

A local edit history of a file is maintained when you create or modify a file. A copy is saved each time you edit and save the file. This allows you to replace the current file with a previous edit or even restore a deleted file. You can also compare the content of all the local edits. Each edit in the local history is uniquely represented by the data and time the file has been saved.

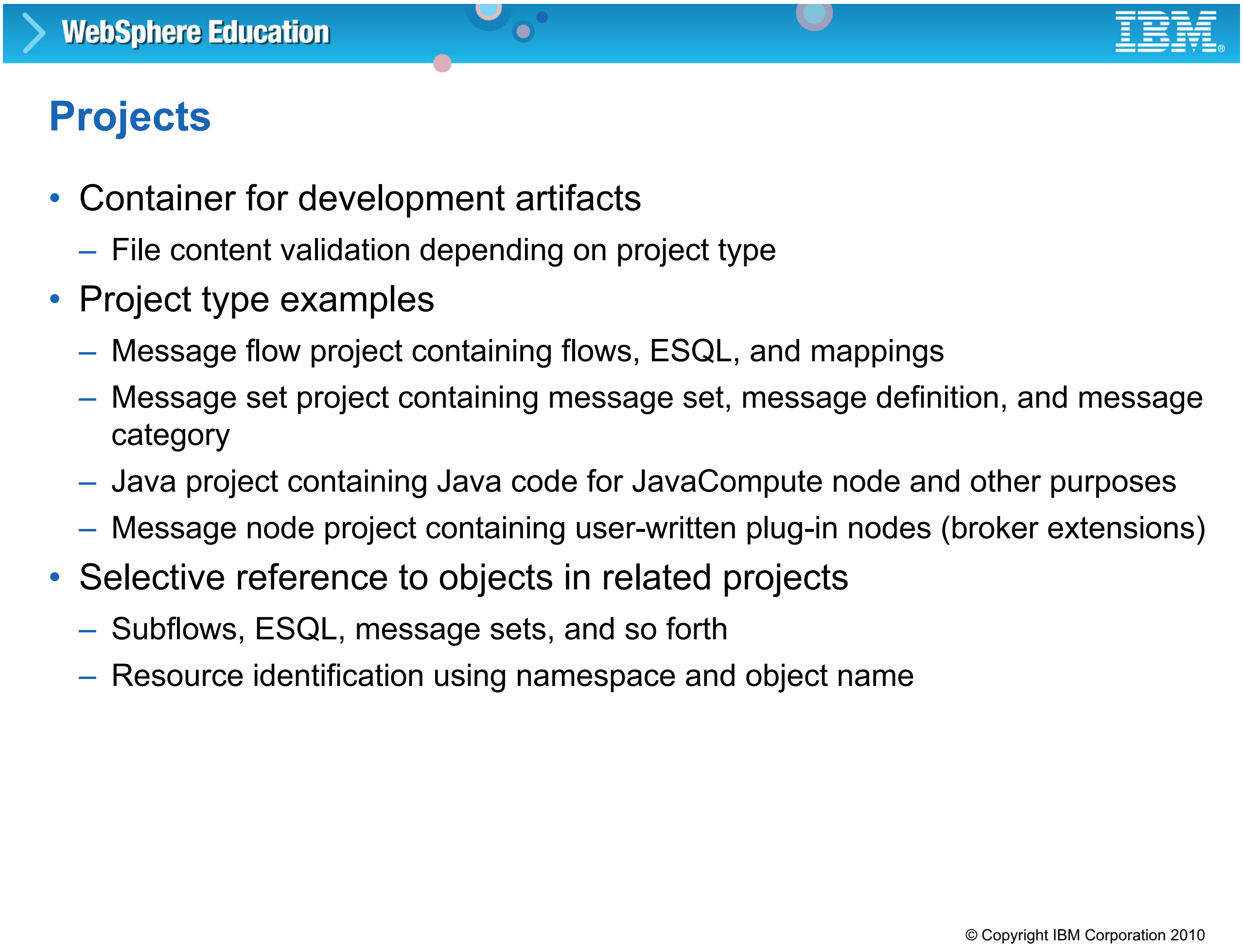
Instructor notes:

Purpose — Introduce local history and customization options

Details — This is best shown in a demonstration.

Additional information — None.

Transition statement — Now you will look into the toolkit in more detail.



WebSphere Education

Projects

- Container for development artifacts
  - File content validation depending on project type
- Project type examples
  - Message flow project containing flows, ESQL, and mappings
  - Message set project containing message set, message definition, and message category
  - Java project containing Java code for JavaCompute node and other purposes
  - Message node project containing user-written plug-in nodes (broker extensions)
- Selective reference to objects in related projects
  - Subflows, ESQL, message sets, and so forth
  - Resource identification using namespace and object name

Figure 3-8. Projects WM643 / VM6431.0

Notes:

WebSphere Message Broker Toolkit organizes all resources into *projects*. A project is the top-level construct for organizing the different resources. It contains files and folders. In the workbench you can create different kinds of projects, and they can have a different structure.

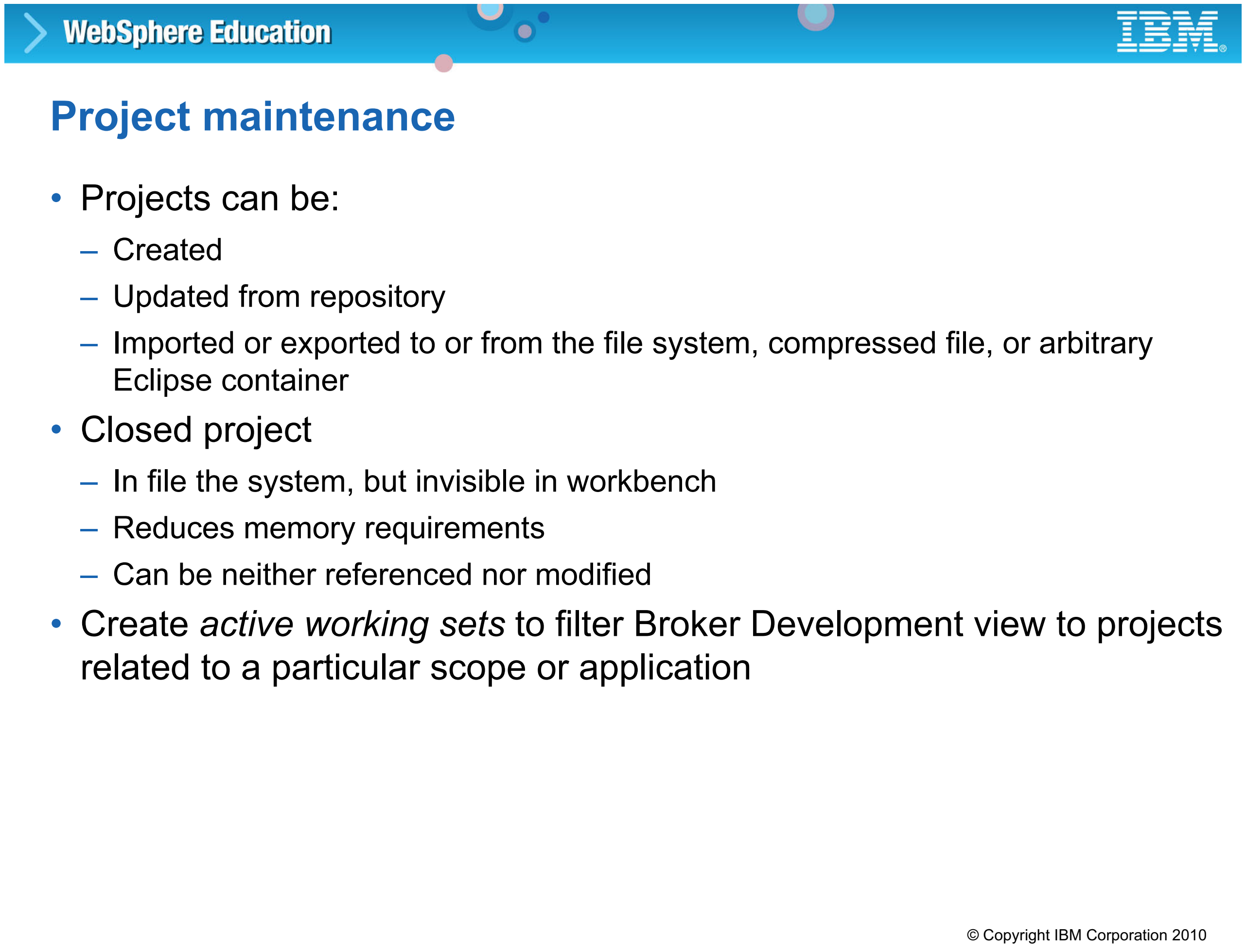
Instructor notes:

Purpose — Introduce the concepts of projects and project types.

Details — None.

Additional information — None.

Transition statement — How can you work with projects?



WebSphere Education

Project maintenance

- Projects can be:
  - Created
  - Updated from repository
  - Imported or exported to or from the file system, compressed file, or arbitrary Eclipse container
- Closed project
  - In file the system, but invisible in workbench
  - Reduces memory requirements
  - Can be neither referenced nor modified
- Create *active working sets* to filter Broker Development view to projects related to a particular scope or application

Figure 3-9. Project maintenance WM643 / VM6431.0

Notes:

Unless specified differently, projects are created in the workspace directory of the WebSphere Message Broker Toolkit installation folder. Also the metadata is stored in the workspace directory. The workspace `.metadata` subdirectory stores important information about the workspace structure, such as a project reference or resource properties.

A project is either open or closed. When a project is closed, it cannot be changed in the workbench and it cannot be referenced from other projects. The resources of a closed project do not appear in the workbench, but they do reside in the local file system. Closed projects require less memory. Because they are not examined during builds, closing a project can improve the build time.

Instructor notes:

Purpose — Outline actions on projects and project references.

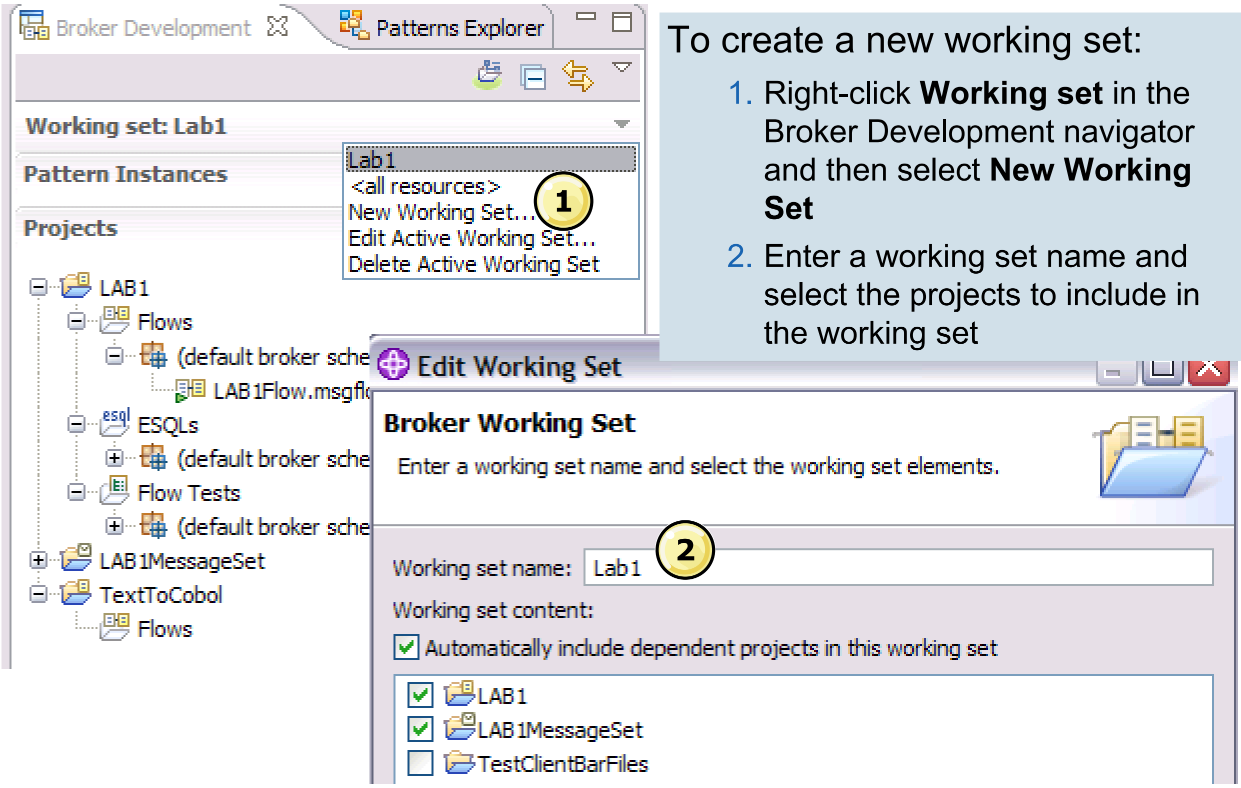
Details — In the exercises, students practice many kinds of actions on projects. Do not spend too much time with theory now. It includes too many details.

Additional information — None.

Transition statement — Projects can be further organized into working sets.

WebSphere Education

Active working sets



Broker Working Set

Enter a working set name and select the working set elements.

Working set name: Lab1

Working set content:

☑ Automatically include dependent projects in this working set

☑ LAB1
☑ LAB1MessageSet
□ TestClientBarFiles

To create a new working set:

1. Right-click **Working set** in the Broker Development navigator and then select **New Working Set**
2. Enter a working set name and select the projects to include in the working set

Figure 3-10. Active working sets WM643 / VM6431.0

Notes:

A *working set* is a logical collection of application projects, that you can use to limit the number of resources that are displayed in the Broker Development view.

Creating and using a working set allows you to reduce the visual complexity of what is displayed in the Broker Development view, making it easier to manage and work with your application projects.

To create a new working set:

1. Right-click **Working set** at the top of the Broker Development navigator and then select **New Working Set** from the drop-down menu.
2. Enter a working set and then select the projects to include in the working set. Click **Finish**.

In addition to creating new working sets, you can also select, edit, and delete existing working sets using the options in the Broker Development view menu.

Instructor notes:

Purpose — Describe a technique for managing projects.

Details — Demonstrate creating a working set.

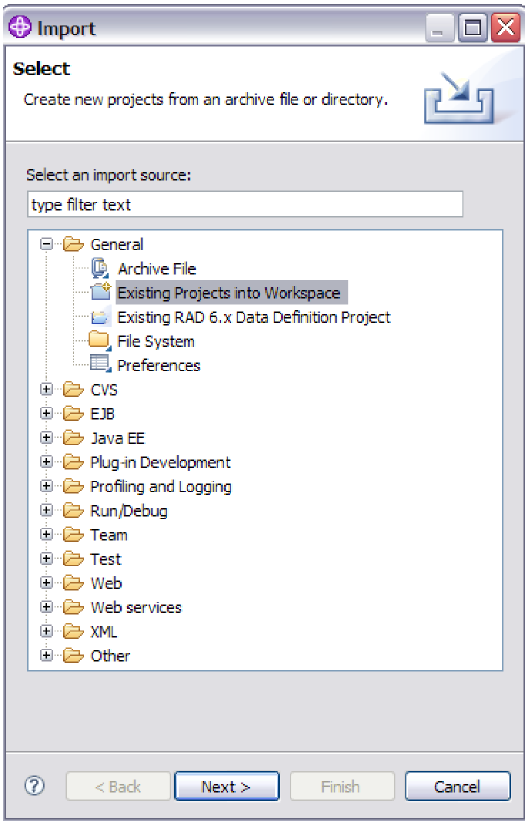
Additional information —

Transition statement — Import and export utilities can be used to share projects and project artifacts.

WebSphere Education IBM

Workspace import and export

- Eclipse operations
  - No transformation
  - No Message Broker involvement
- Files are copied between workspace and:
  - The file system
  - A compressed file
  - Any other files container supported by Eclipse
- File import wizard copies resources into existing project, or adds workspace view on project existing anywhere in file system
- Drag or cut and paste resources between the file system and the existing project



Create new projects from an archive file or directory.

Figure 3-11. Workspace import and export WM643 / VM6431.0

Notes:

Never move resources by moving their associated files in the file system. You must use the workbench to move resources to ensure that all references are corrected to reflect the new organization.

There are several methods for importing files into the WebSphere Message Broker Toolkit. Typically you create a project and then use the “import from file system” or “import Project Interchange file” actions. Both of these methods create a file in the file system and copy the source into that new file.

Altering the new file does not affect the original source files. However, there is also an “import existing project” capability. Use this method with care, since it does ***not*** create a file in the file system to hold the imported content. Instead, it points directly to the existing project. If you alter that file, you are altering the original project, not a copy of it.

Instructor notes:

Purpose — Explain more basic tasks with projects and resources.

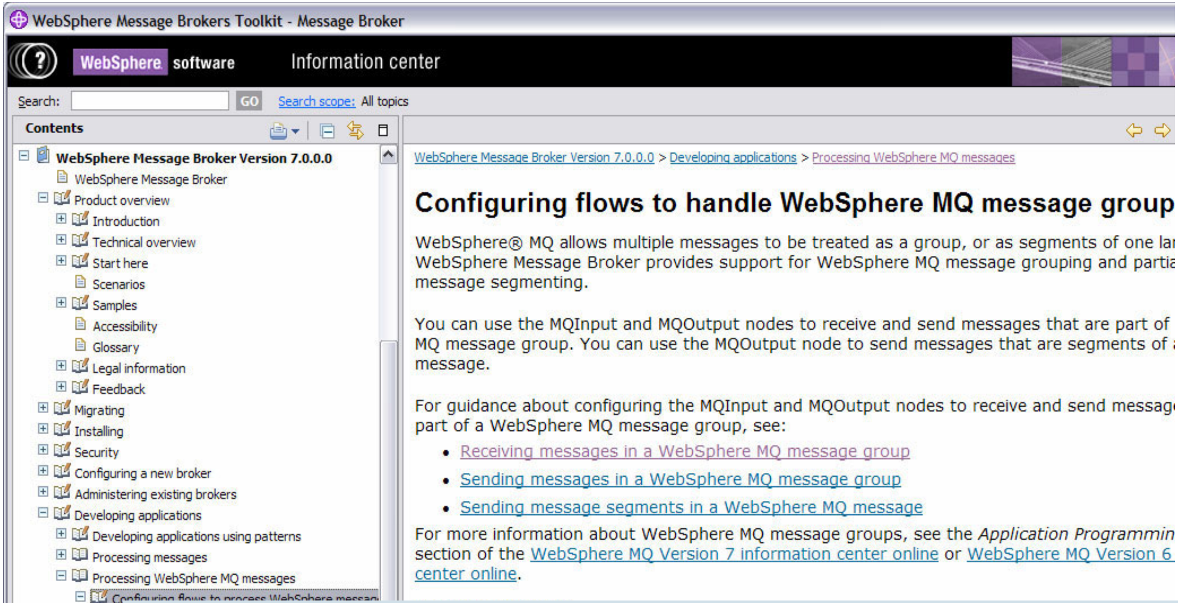
Details — When you import an existing project into the workspace, it is ***not*** moved or copied into the workspace directory. Rather, the project in the workspace contains a “pointer” to the physical name and location in file system.

Additional information — None.

Transition statement — The WebSphere Message Broker Toolkit includes interactive help and access to the WebSphere Message Broker Information Center.

WebSphere Education

WebSphere Information Center



WebSphere Message Brokers Toolkit - Message Broker

- WebSphere Message Broker Version 7.0.0.0
  - WebSphere Message Broker
  - Product overview
    - Introduction
    - Technical overview
    - Start here
    - Scenarios
    - Samples
    - Accessibility
    - Glossary
    - Legal information
    - Feedback
  - Migrating
  - Installing
  - Security
  - Configuring a new broker
  - Administering existing brokers
  - Developing applications
    - Developing applications using patterns
    - Processing messages
    - Processing WebSphere MQ messages
      - Configuring flows to process WebSphere messag
      - Er
    - Proce
    - Proce
    - Proce
    - Proce
    - Proce
    - Using WebSphere Adapters

WebSphere Message Broker Version 7.0.0.0 > Developing applications > Processing WebSphere MQ messages

Configuring flows to handle WebSphere MQ message group

WebSphere® MQ allows multiple messages to be treated as a group, or as segments of one lar WebSphere Message Broker provides support for WebSphere MQ message grouping and partia message segmenting.

You can use the MQInput and MQOutput nodes to receive and send messages that are part of MQ message group. You can use the MQOutput node to send messages that are segments of a message.

For guidance about configuring the MQInput and MQOutput nodes to receive and send messag part of a WebSphere MQ message group, see:

- Receiving messages in a WebSphere MQ message group
- Sending messages in a WebSphere MQ message group
- Sending message segments in a WebSphere MQ message

For more information about WebSphere MQ message groups, see the *Application Programmin* section of the WebSphere MQ Version 7 information center online or WebSphere MQ Version 6 center online.

- View, browse, and search online information
  – Within the WebSphere Message Broker Toolkit
  – Within WebSphere Message Broker Explorer
  – From the Internet on IBM public Web site
    `http://www.ibm.com/software/integration/messagebrokerproductline/`

Figure 3-12. WebSphere Information Center WM643 / VM6431.0

Notes:

It is important to know where to go to get additional information about the product and product application. The WebSphere Information Center contains a complete set of online application documentation.

Instructor notes:

Purpose — Describe the different ways to access the online documentation.

Details — It is critical that students know where to go for additional information so this slide is worth repeating.

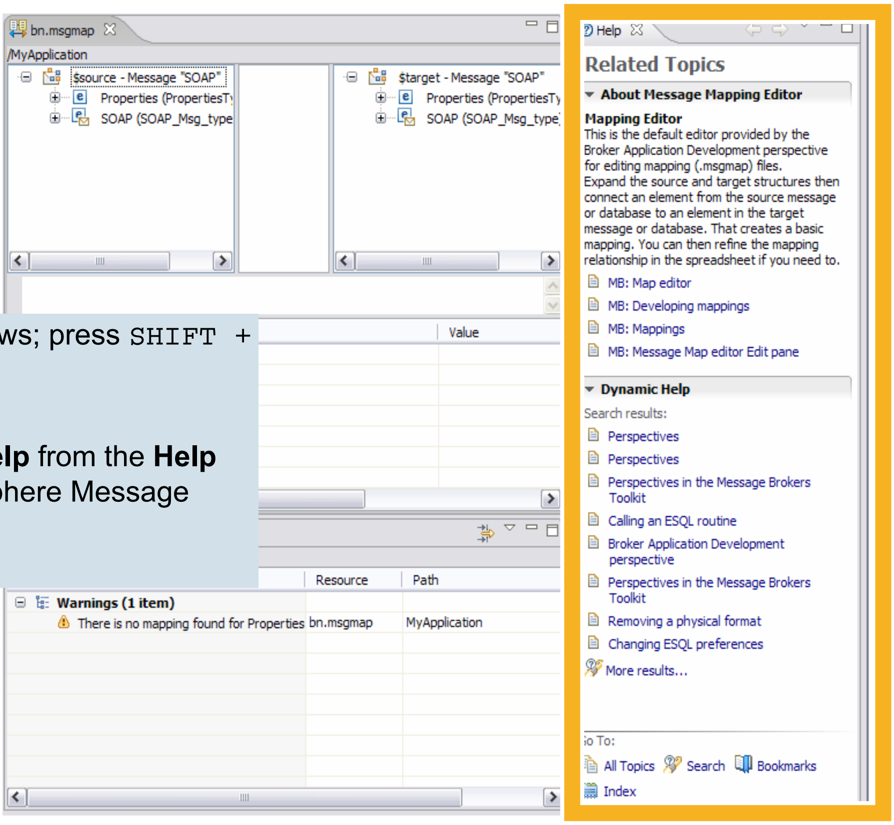
Additional information —

Transition statement — Context help makes finding documentation easier.

Integrated help

- Press F1 on Windows; press SHIFT + F1 on Linux

  or
- Select **Dynamic Help** from the **Help** menu in the WebSphere Message Broker Toolkit



There is no mapping found for Properties bn.msgmap MyApplication

Mapping Editor
This is the default editor provided by the Broker Application Development perspective for editing mapping (.msgmap) files.
Expand the source and target structures then connect an element from the source message or database to an element in the target message or database. That creates a basic mapping. You can then refine the mapping relationship in the spreadsheet if you need to.

- MB: Map editor
- MB: Developing mappings
- MB: Mappings
- MB: Message Map editor Edit pane

- Perspectives
- Perspectives
- Perspectives in the Message Brokers Toolkit
- Calling an ESQL routine
- Broker Application Development perspective
- Perspectives in the Message Brokers Toolkit
- Removing a physical format
- Changing ESQL preferences
- More results...

Figure 3-13. Integrated help WM643 / VM6431.0

Notes:

A Help view is provided in the WebSphere Message Broker Toolkit. To view the Help view in the WebSphere Message Broker Toolkit, press **F1** on Windows, or **SHIFT** + **F1** on Linux, or select **Dynamic Help** from the **Help** menu.

The default view for Help contains **Related Topics**. In the top section there is a small description of the interface element, such as a view, or an object, such as a node, that you last selected with the mouse. This description also contains links to topics in the Information Center. Click a link to open any topic in the Help view. Right-click the topic link and select **Open in Help Contents** to open the topic in the Information Center. The topic Help opens in a a separate window.

The bottom section of the Help view contains relevant links from the Information Center. These links are generated using a search on the values of the perspective that you are in and the name of the interface element that you selected. The relevancy of the search results is variable depending on the interface element selected.

When you click a new interface element, the content of the **Related Topics** changes to match the new object.

Instructor notes:

Purpose — Describe how to access the context-sensitive help in the WebSphere Message Broker Toolkit.

Details —

Additional information —

Transition statement — Next: WebSphere Message Broker Toolkit Welcome view

3.2. Basic configuration tasks

Instructor topic introduction

What students will do — Review the graphical nature of the WebSphere Message Broker through the Broker Development perspective, in this case to perform post-install operations.

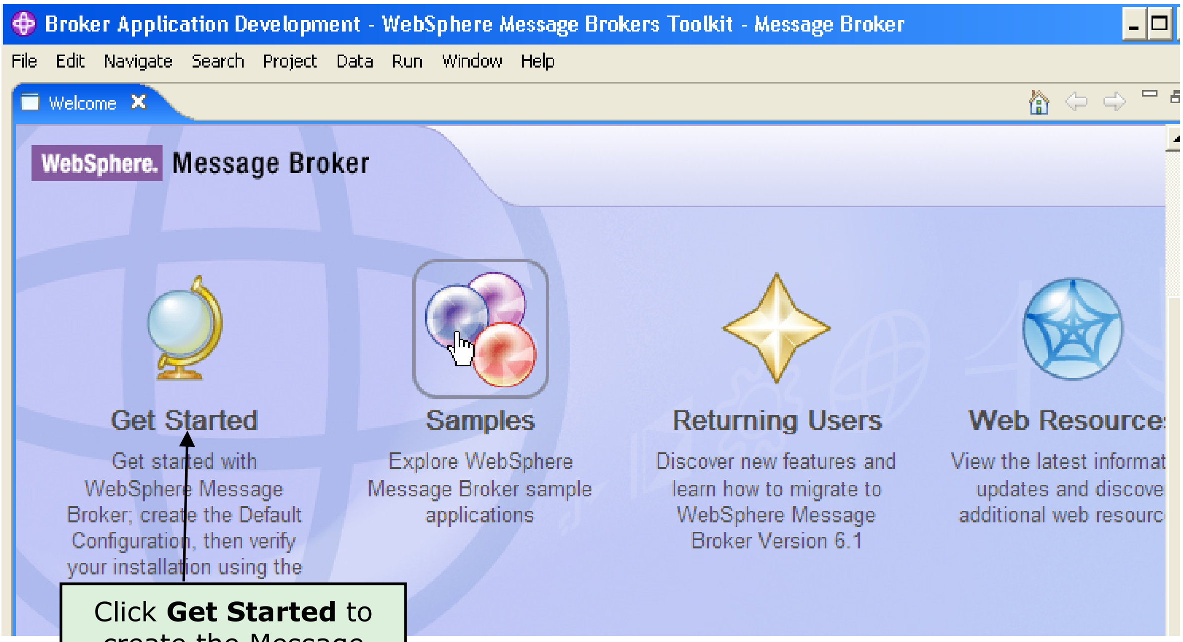
How students will do it — Listen to the lecture and participate in classroom discussion. Later exercises (2 and 4) gives students the opportunity to practice the procedures and operations covered in this topic.

What students will learn — Some basic functions provided by the WebSphere Message Broker Toolkit, available through the Broker Development perspective, are used for post-install tasks.

How this will help students on their job — WebSphere Message Broker V7 administration is performed through the WebSphere Message Broker Toolkit, WebSphere Message Broker Explorer, command line, and a set of programming APIs called the Application API. You begin here with basic skill development using the basic administration support provided in the WebSphere Message Broker Toolkit, with an initial focus on post-install configuration tasks.

WebSphere Education

WebSphere Message Broker Toolkit Welcome view


Broker Application Development - WebSphere Message Brokers Toolkit - Message Broker
File Edit Navigate Search Project Data Run Window Help
Welcome
WebSphere. Message Broker
Get Started
Samples
Returning Users
Web Resource

Get started with WebSphere Message Broker; create the Default Configuration, then verify your installation using the

Explore WebSphere Message Broker sample applications

Discover new features and learn how to migrate to WebSphere Message Broker Version 6.1

View the latest informat updates and discove additional web resourc

Click **Get Started** to create the Message Broker default configuration

Figure 3-14. WebSphere Message Broker Toolkit Welcome view WM643 / VM6431.0

Notes:

The Welcome view, shown in the figure, is displayed the first time the WebSphere Message Broker Toolkit is started. You should have seen this view when you started the WebSphere Message Broker Toolkit in Exercise 1. The Welcome view provides access to the Default Configuration wizard, sample, upgrade information, and web resources.

Closing the initial view by clicking the **X** on the **Welcome** tab, yields the second (default) view, the Broker Development perspective. As the name implies, developers spend most of their time in this perspective.

Additional perspectives can be accessed by selecting **Open Perspective** from the **Window** menu.

The title bar and a corresponding icon in the upper right of the screen confirm a particular perspective selection.

Instructor notes:

Purpose — A starting point for this topic.

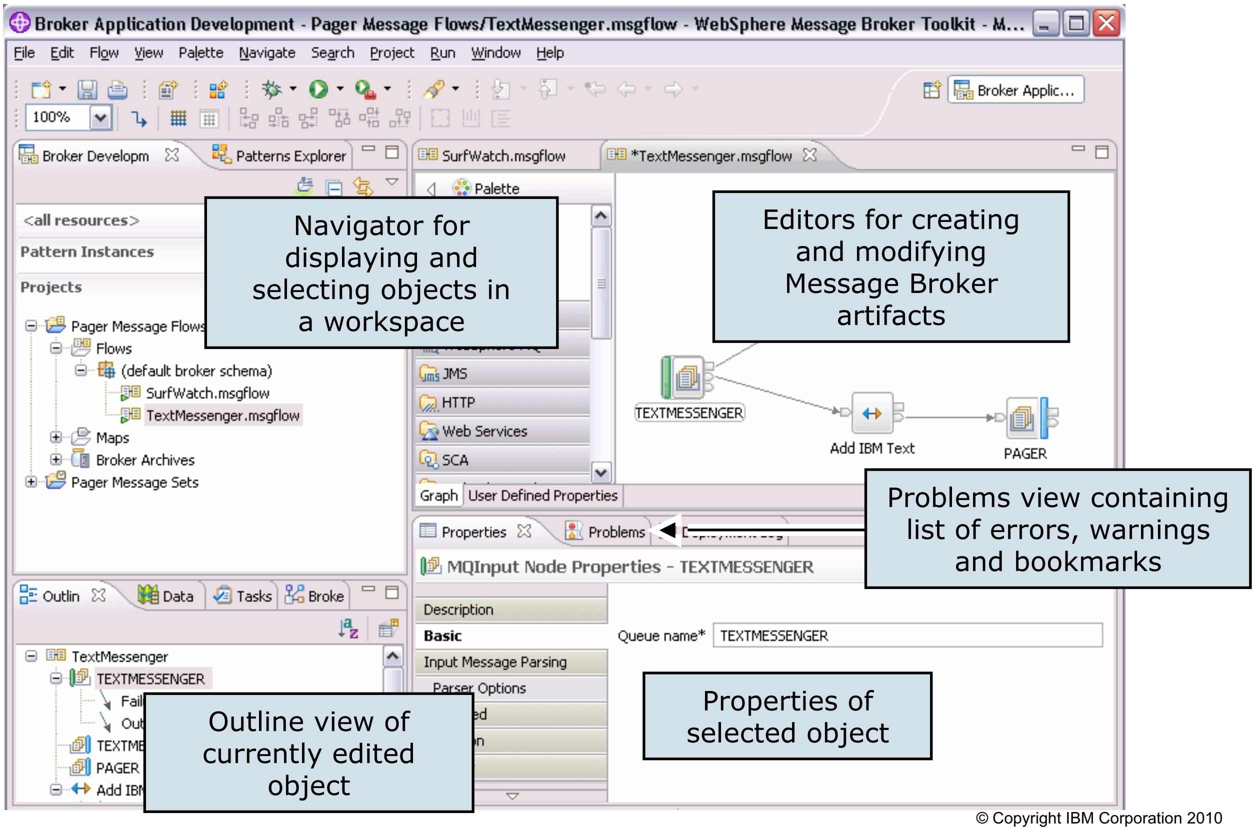
Details — None.

Additional information — None.

Transition statement — Next: Broker Development views

WebSphere Education

Broker Development views



Navigator for displaying and selecting objects in a workspace

Editors for creating and modifying Message Broker artifacts

Problems view containing list of errors, warnings and bookmarks

Outline view of currently edited object

Properties of selected object

Figure 3-15. Broker Development views WM643 / VM6431.0

Notes:

The figure shows some of the most common views that are available in the WebSphere Message Broker Development perspective:

- The Message Flow Editor view contains the message flow nodes in a palette and an area for developing message flows and wiring message nodes together.
- The Navigator is used to display and select objects in a workspace. Right-clicking most objects in the Navigator launches a menu. Double-clicking most objects opens the object in an editor.
- There are a number of tabbed views provided in the lower right of the Broker Development perspective. The Outline view provides an outline of currently edited objects.
- The Properties view provides the configuration properties for the currently selected object.
- The Problems view contains the list of current errors or warnings

Instructor notes:

Purpose — Provide an overview of the most commonly used views in the Broker Development perspective.

Details — Open the Broker Development perspective with a message flow. Quickly demonstrate view navigation and the effect of selecting, right-clicking, and double-clicking an object.

Additional information — None.

Transition statement — Next: Brokers view

Brokers view

- Hosted with Outline view
- Allows application developer to:
  - Create and delete local brokers
  - Connect to remote broker
  - Create and delete execution group
  - Configure debug port and launch debugger for execution group
  - Deploy message flows to execution group
  - Remove deployed artifacts from execution group
  - Start and stop brokers, execution groups, and message flows

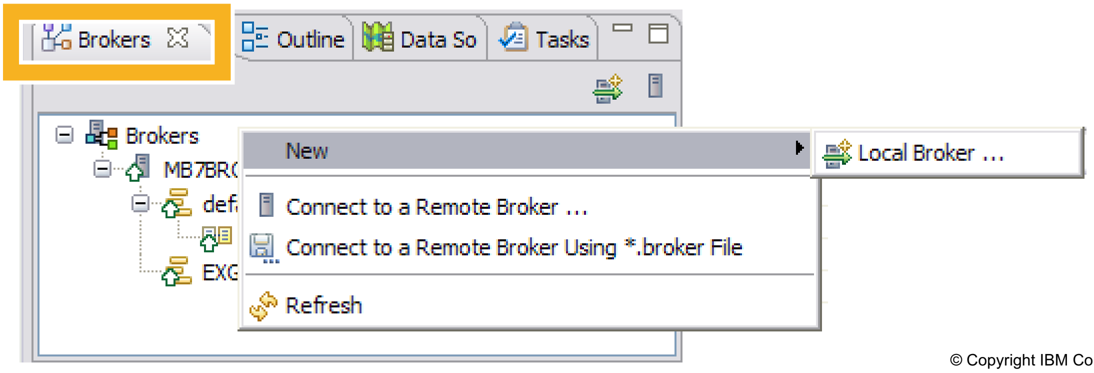

Brokers
Outline
Data So
Tasks
Brokers
MB7BR(
def.
EXG
New
Local Broker …
Connect to a Remote Broker …
Connect to a Remote Broker Using *.broker File
Refresh

Figure 3-16. Brokers view WM643 / VM6431.0

Notes:

The Brokers view can be used to perform some of the broker administrative tasks, such as creating, deleting, and starting and stopping local brokers, connecting to remote brokers, and starting and stopping local or remote execution groups.

You cannot create remote brokers from the WebSphere Message Broker Toolkit.

Instructor notes:

Purpose — Describes the Brokers view

Details — Show the Brokers view in the WebSphere Message Broker Toolkit Development perspective.

Additional information —

Transition statement — Next: Basic configuration

WebSphere Education

Basic configuration

1. Create a local broker or connect to remote broker.
2. Start the broker.
3. Create an execution group.

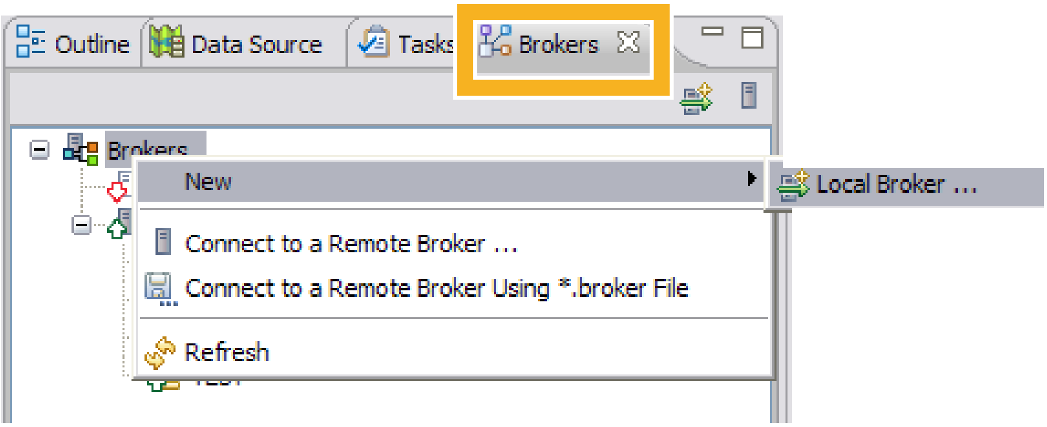

Outline
Data Source
Tasks
Brokers
Brokers
New
Local Broker …
Connect to a Remote Broker …
Connect to a Remote Broker Using *.broker File
Refresh

Figure 3-17. Basic configuration

WM643 / VM6431.0

Notes:

Developers can perform basic configuration tasks from the WebSphere Message Broker Toolkit.

A basic configuration includes a connection to a broker and at least one execution group. In the WebSphere Message Broker Toolkit, local brokers are automatically connected. You must manually connect to remote brokers

Creation assumes that queue manager has already been created.

Instructor notes:

Purpose — List the basic configuration tasks.

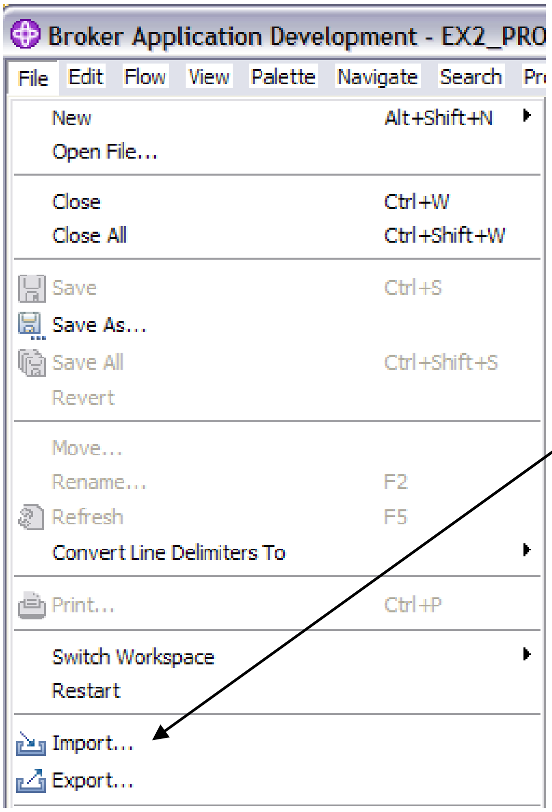
Details — The Default Configuration wizard creates a basic configuration that developer's can use to test message flows.

Additional information —

Transition statement — After the basic configuration is created, you can create or import message flow artifacts.

WebSphere Education

Importing a resource (1 of 4)



Broker Application Development - EX2_PRO

File Edit Flow View Palette Navigate Search Pr

New	Alt+Shift+N
Open File...	
Close	Ctrl+W
Close All	Ctrl+Shift+W
Save	Ctrl+S
Save As...	
Save All	Ctrl+Shift+S
Revert	
Move...	
Rename...	F2
Refresh	F5
Convert Line Delimiters To	
Print...	Ctrl+P
Switch Workspace	
Restart	
Import...	
Export...	

1 Select **File > Import** from the Broker Application Development menu

Figure 3-18. Importing a resource (1 of 4) WM643 / VM6431.0

Notes:

The Broker Development navigator presents, as a tree structure, a view of the user's workspace as it exists on the local file system. One way to add something to this space is through an import operation. Selection can be made through the menu bar as **File** > **Import**.

This starts another wizard, which is the subject of the next figure.

Instructor notes:

Purpose — Show the navigation path to import, in this case, a message flow project.

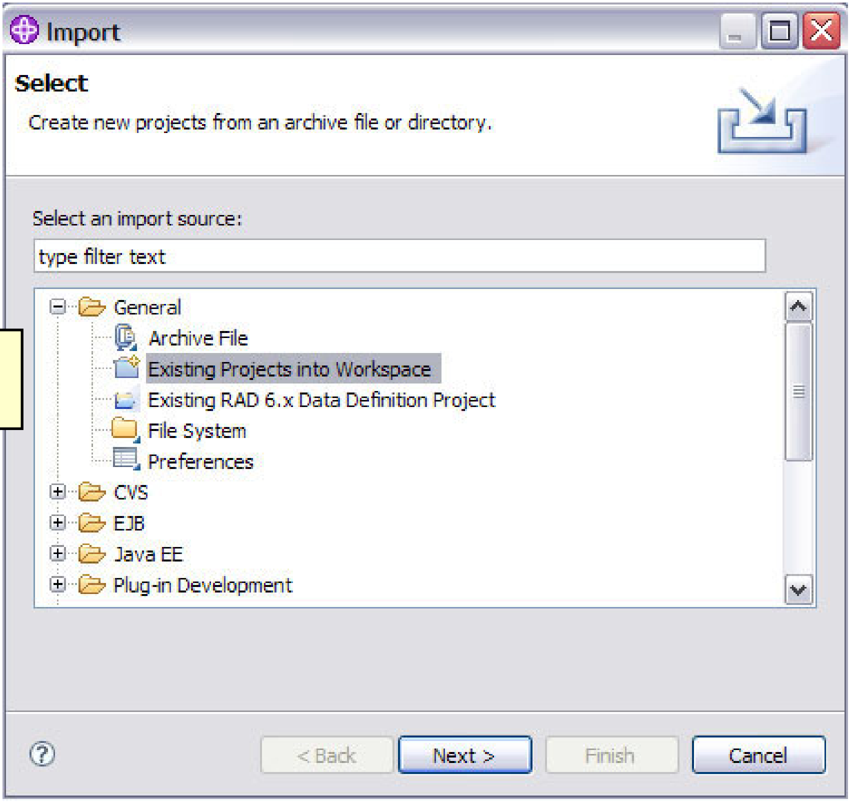
Details — None.

Additional information — Alternatively, to create a message flow from scratch, one can go to the menu bar and select **File > New > Message Flow Project**.

Transition statement — The wizard prompts you to identify the source of the object you want to import.

WebSphere Education

Importing a resource (2 of 4)



Create new projects from an archive file or directory.

Select an import source:

type filter text

- General
  - Archive File
  - Existing Projects into Workspace
  - Existing RAD 6.x Data Definition Project
  - File System
  - Preferences
- CVS
- EJB
- Java EE
- Plug-in Development

2 Select import source

Figure 3-19. Importing a resource (2 of 4) WM643 / VM6431.0

Notes:

As the figure suggests, there are many options associated with bringing pre-existing objects into a workspace. Here, you bring in a completed message flow project, which was previously exported. This is a common technique to move completed work from developer to administrator, or from a test environment to production.

Later, you use the **File System** selection to bring other objects into your workspace, namely test messages.

Instructor notes:

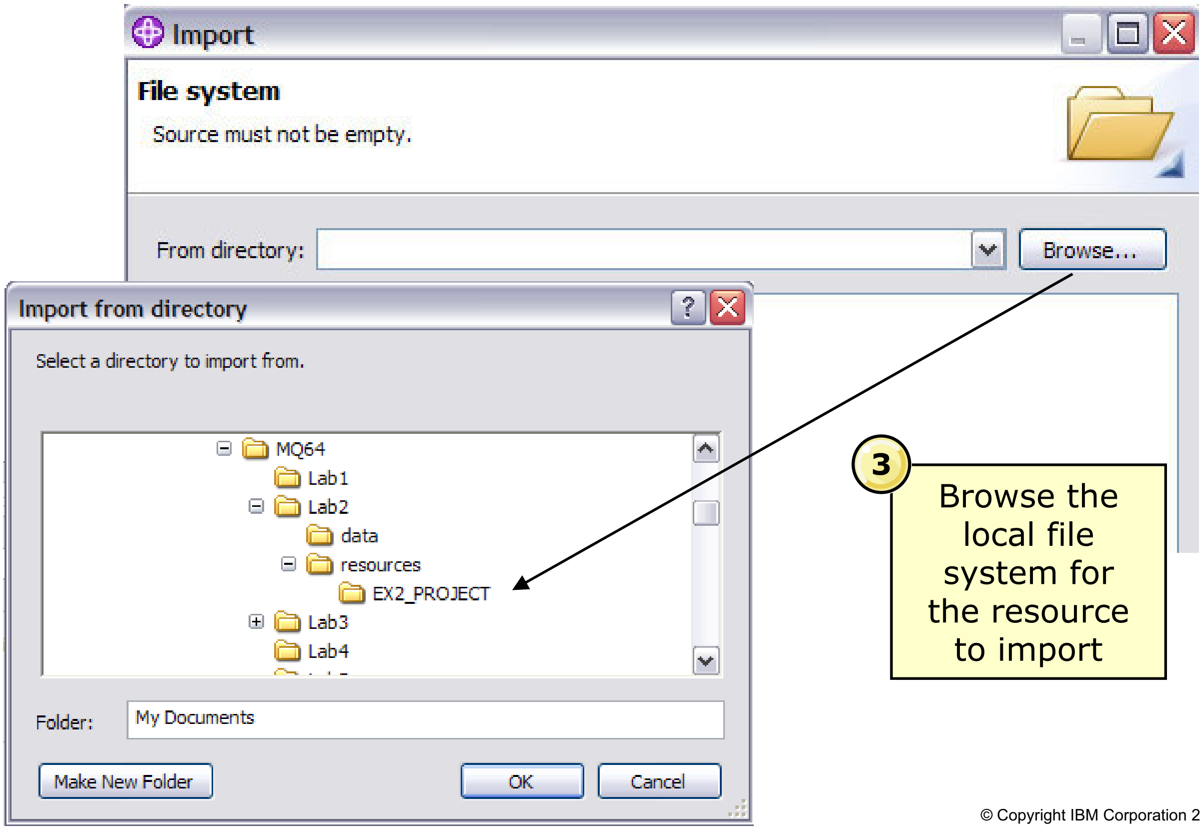
Purpose — Show the versatility of the import function.

Details — None.

Additional information — None.

Transition statement — Now you browse the local file system and select the wanted object, in this case a message flow project.

Importing a resource (3 of 4)



Import

File system

Source must not be empty.

From directory:

Import from directory

Select a directory to import from.

Folder: My Documents

3 Browse the local file system for the resource to import

Figure 3-20. Importing a resource (3 of 4) WM643 / VM6431.0

Notes:

The **Browse** button is used to go to the local file system and select the resource to import. The object shown in the figure (**EX2_PROJECT**) agrees with what you are doing in the upcoming exercise.

Instructor notes:

Purpose — Illustrate one of several links between the WebSphere Message Broker Toolkit and the local file system.

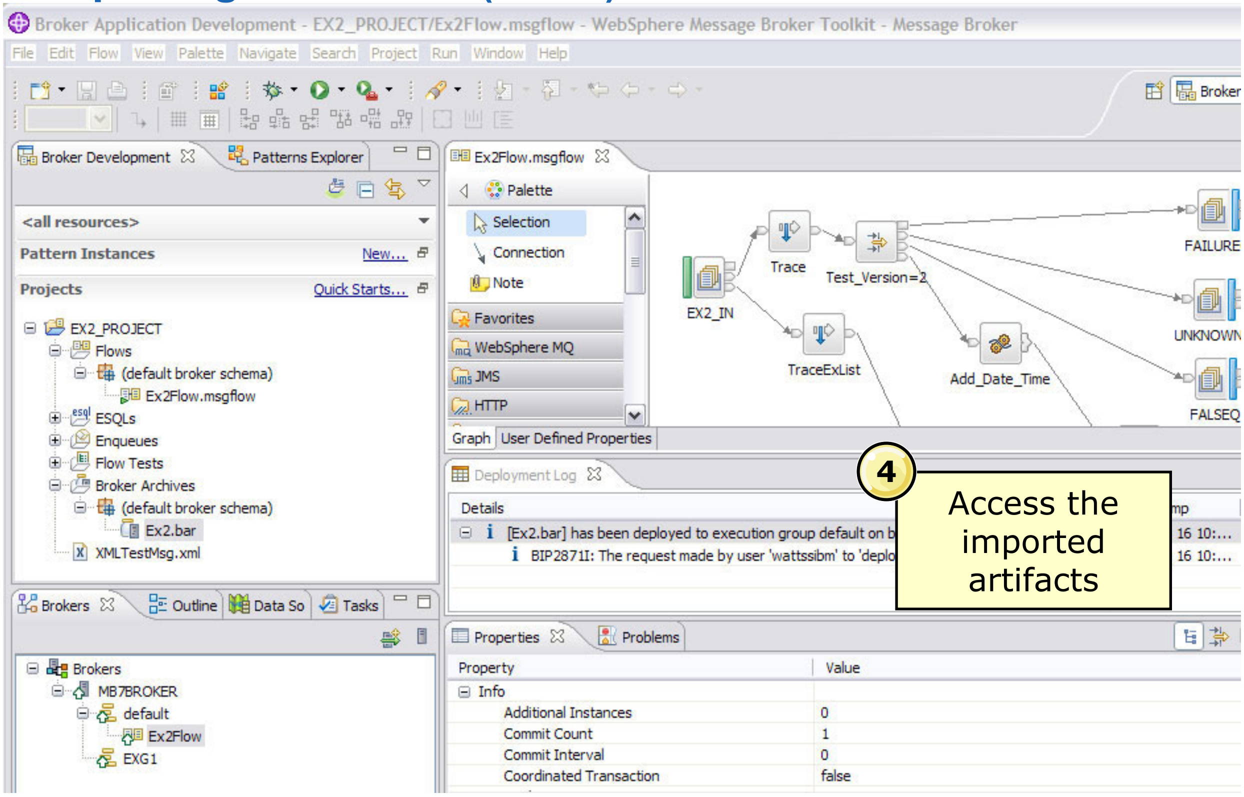
Details — None.

Additional information — None.

Transition statement — Having made a selection, you see what changes have taken place in the workspace.

WebSphere Education

Importing a resource (4 of 4)



[Ex2.bar] has been deployed to execution group default on b

BIP2871I: The request made by user 'wattssibm' to 'deplo

16 10:...

16 10:...

Property	Value
Info	
Additional Instances	0
Commit Count	1
Commit Interval	0
Coordinated Transaction	false

4 Access the imported artifacts

© Copyright IBM Corporation 2010

Figure 3-21. Importing a resource (4 of 4) WM643 / VM6431.0

Notes:

A successful import, in this case of a message flow project, positions the imported object in the navigator. Expanding the object reveals, in this case, a `.msgflow` and a `.bar` file.

Double-clicking opens the selected object. In the figure, you see the details of the `Ex2.msgflow` file selected from the navigator. You will see this graphic again when you begin Exercise 2.

Instructor notes:

Purpose — Expose students (administrators) to the work product of developers.

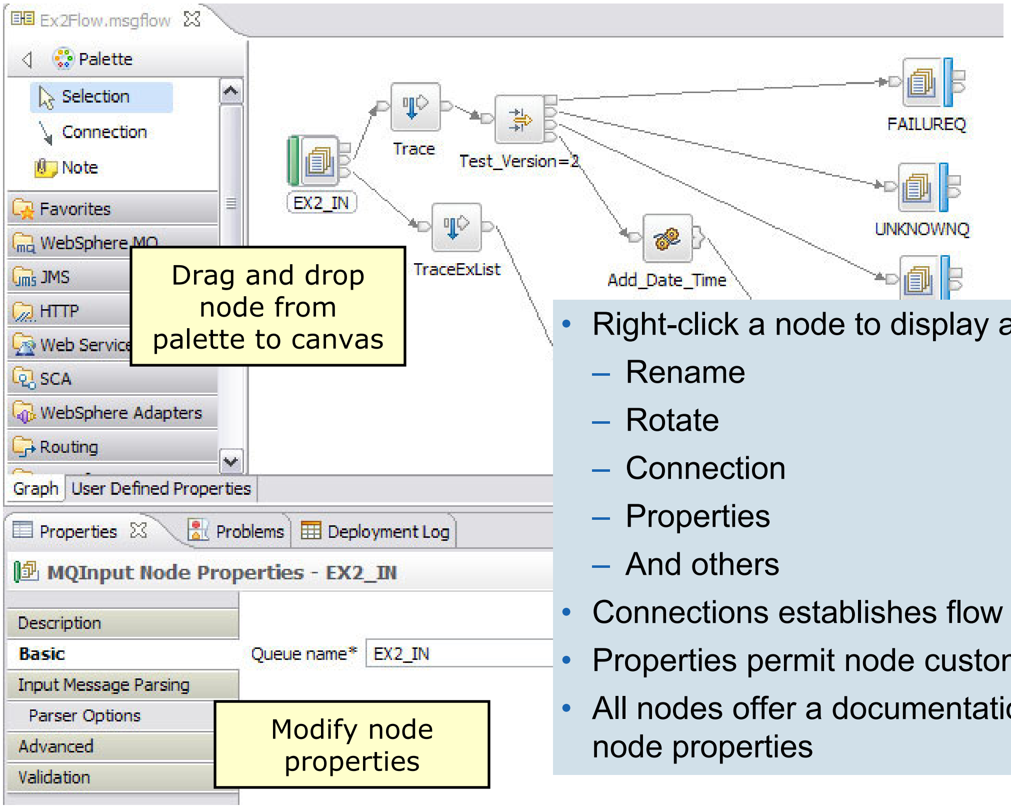
Details — None.

Additional information — None.

Transition statement — You might ask how a message flow is developed. Briefly explain.

WebSphere Education

Create or modify a message flow



Queue name* EX2_IN

Drag and drop node from palette to canvas

Modify node properties

- Right-click a node to display available options
  - Rename
  - Rotate
  - Connection
  - Properties
  - And others
- Connections establishes flow sequence
- Properties permit node customization
- All nodes offer a documentation option in node properties

Figure 3-22. Create or modify a message flow WM643 / VM6431.0

Notes:

Message flows are made up of message processing nodes, represented as icons and positioned in the Message Flow editor. Connections between the nodes establish their execution sequence. Each node contributes to the overall purpose or intent of the message flow through node properties. The next few figures provide some additional detail.

Creating and customizing a message flow, or altering an existing message flow, is procedural in nature. As the figure suggests, a number of steps are followed, not necessarily in a specific order.

Instructor notes:

Purpose — Establish a framework for message flow development.

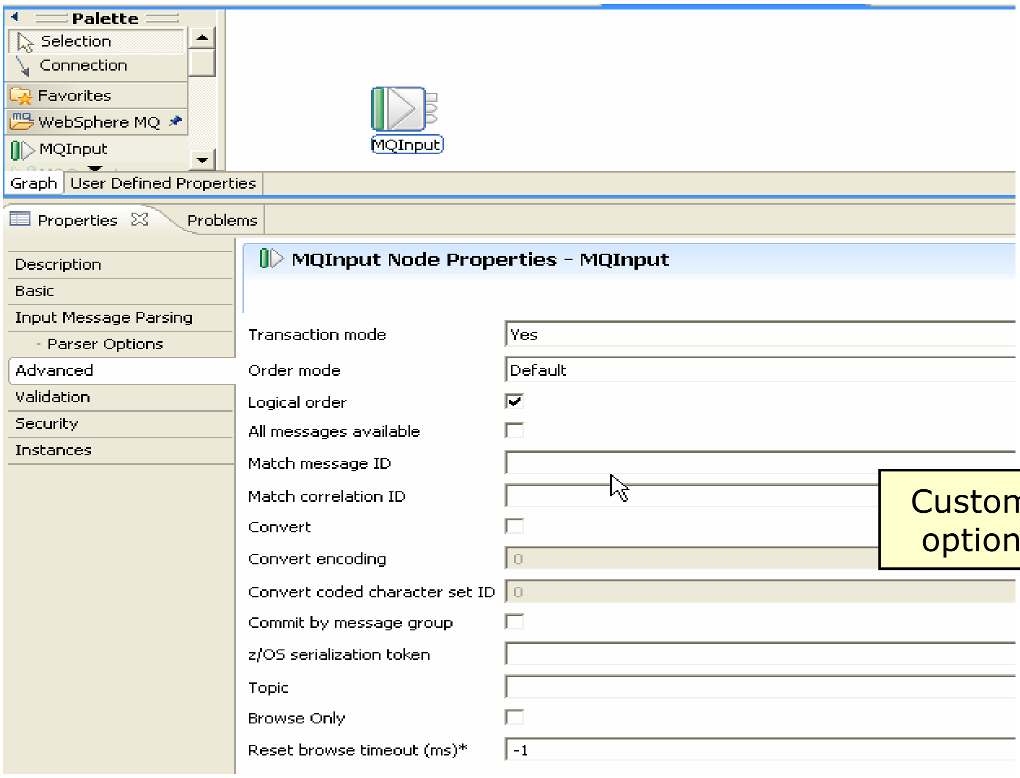
Details — None.

Additional information — Order is more logical than anything else, and individuals will likely establish their own way of getting the job done by personal taste.

Transition statement — Nodes take on behavioral characteristics by setting node properties. An example follows.

WebSphere Education IBM

MQInput node properties example



MQInput Node Properties - MQInput

Transaction mode: Yes
Order mode: Default
Logical order ☑
All messages available □
Match message ID
Match correlation ID
Convert □
Convert encoding: 0
Convert coded character set ID: 0
Commit by message group □
z/OS serialization token
Topic
Browse Only □
Reset browse timeout (ms)*: -1

Customization options page

© Copyright IBM Corporation 2010

Figure 3-23. MQInput node properties examples WM643 / VM6431.0

Notes:

Nodes are named in a way to convey their intent and purpose. For example, an MQInput node is used to start a message flow by delivering to the flow a message obtained from a WebSphere MQ queue (in essence, it performs an MQGET operation). One of the node properties of an MQInput node is the name of the WebSphere MQ queue from which it is to obtain messages. Developers would likely provide the case-sensitive queue name, as shown in the figure.

The figure shows the customization option pages obtained by clicking the node instance. The **Properties** view appears, by default, at the bottom of the Broker Development perspective. By continuing through the several **Properties** pages**,** the developer has the opportunity to customize the node in a manner that is consistent with the node's role in the message flow.

As expected, customization options vary among nodes and are directly associated with the intents and purposes of each node. For example, customizing an MQOutput node will likely identify a WebSphere MQ queue to which a processed message is directed by MQPUT — but not always.

Instructor notes:

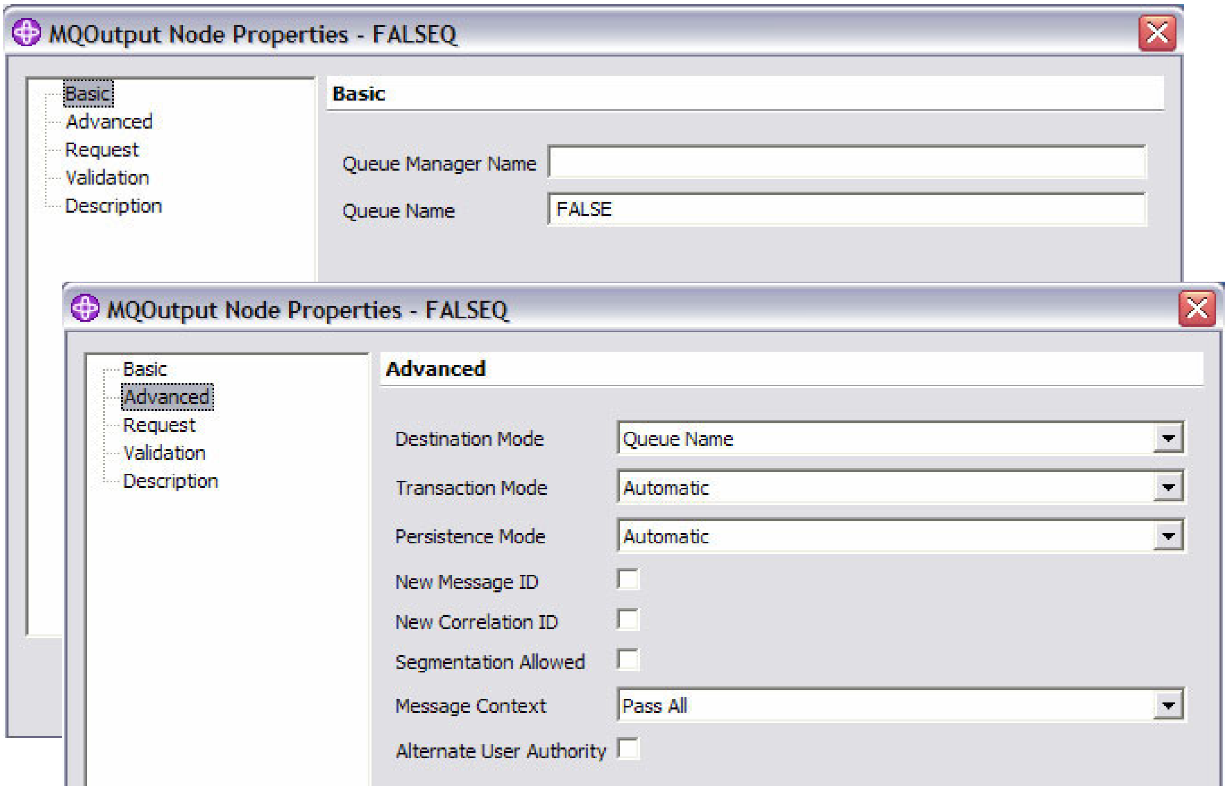
Purpose — Establish an overview of what a message flow is (that is, its appearance), how it is constructed (nodes and connections), and how it performs the job the developer intends (node properties).

Details — The screen capture on this slide shows the Properties view as it is displayed by default. Show the students how to select the different pages.

Additional information — A message flow is nothing more than a program created with graphical tools rather than a source programming language. Since language skills are not required by the developer, the thought is that less skilled people can still produce some complicated code.

Transition statement — The next figure looks at some of the properties of an MQOutput node.

MQOutput node properties example



MQOutput Node Properties - FALSEQ

- Basic
- Advanced
- Request
- Validation
- Description

Basic

Queue Manager Name:

Queue Name: FALSE

MQOutput Node Properties - FALSEQ

- Basic
- Advanced
- Request
- Validation
- Description

Advanced

Destination Mode: Queue Name

Transaction Mode: Automatic

Persistence Mode: Automatic

New Message ID □

New Correlation ID □

Segmentation Allowed □

Message Context: Pass All

Alternate User Authority □

Figure 3-24. MQOutput node properties example WM643 / VM6431.0

Notes:

The figure shows some of the dialog that results from clicking an MQOutput node and selecting the **Properties** view.

Note in the figure the list of property pages (on the left) through which related node properties can be set. Many have default settings that usually suffice.

Required parameters are identified with an asterisk. If not provided, a red decorator and a suitable error message appear at the top of the dialog. In the case of an MQOutput node, it is interesting to note that a queue name is not always required.

Note, in the figure, the page with **Advanced** highlighted and examine the setting for **Destination mode**. When **Destination mode** is set to **Queue name**, then a queue name must be specified on the **Basic** page. Other output options available to a developer include **Reply to queue** and **Destination list**. You will use the **Reply to queue** option in Exercise 2.

Instructor notes:

Purpose — Take a closer look at node properties, using the MQOutput node as an example.

Details — It might be best to open the message flow students will be deploying in Exercise 2 and walk through the nodes in as much detail as required. This is a class for administrators so try not to spend too much time on the node details. The focus should be on properties that an administrator might need to change when the message flow is deployed to a production environment, such as the node name.

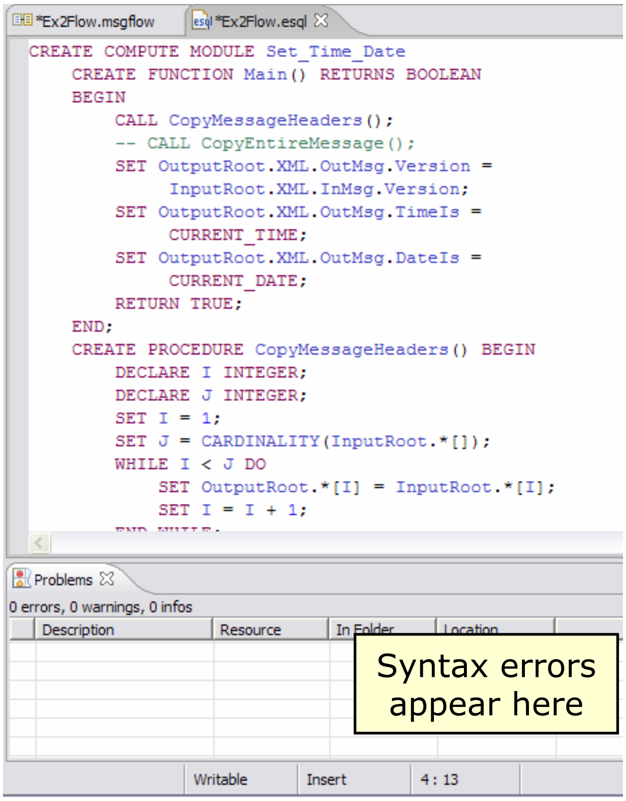
Additional information — Depending on the nature of the node, properties can be extensive or nonexistent.

Transition statement — Some nodes are programmable and their behavior can be truly customized with programming — either ESQL or Java.

WebSphere Education

Compute node and ESQL editor

- Use Compute node to:
  - Build a new message using a set of assignment statements
  - Copy messages between parsers
  - Convert messages from one code set to another
  - Transform message from one format to another
  - Specify how new messages are created by coding ESQL
- ESQL editor
  - Default editor for editing ESQL (`.esql`) files
  - Context sensitive Content Assist



```
CREATE COMPUTE MODULE Set_Time_Date
    CREATE FUNCTION Main() RETURNS BOOLEAN
    BEGIN
        CALL CopyMessageHeaders();
        -- CALL CopyEntireMessage();
        SET OutputRoot.XML.OutMsg.Version =
             InputRoot.XML.InMsg.Version;
        SET OutputRoot.XML.OutMsg.TimeIs =
             CURRENT_TIME;
        SET OutputRoot.XML.OutMsg.DateIs =
             CURRENT_DATE;
        RETURN TRUE;
    END;
    CREATE PROCEDURE CopyMessageHeaders() BEGIN
        DECLARE I INTEGER;
        DECLARE J INTEGER;
        SET I = 1;
        SET J = CARDINALITY(InputRoot.*[]);
        WHILE I < J DO
            SET OutputRoot.*[I] = InputRoot.*[I];
            SET I = I + 1;
        END WHILE;
```

0 errors, 0 warnings, 0 infos

Description	Resource	In Folder	Location

Syntax errors appear here

Figure 3-25. Compute node and ESQL editor WM643 / VM6431.0

Notes:

The Compute node creates an output message by coding ESQL which is a WebSphere Message Broker programming language that defines and manipulates data within a message flow.

Open ESQL in an option of the Compute node pull-down menu**.** Selecting this option opens the ESQL editor and, on first use, creates a `.esql` file into which the developer can write any ESQL statements appropriate to the flow.

The ESQL statements are grouped in modules (also known as stanzas) where each module affects the behavior of a corresponding node. This association is obvious by looking at the module name, which takes the form `<Flow_name>_<node_name>`. You can see this in the first line of the figure, which reads: `CREATE COMPUTE MODULE Set_Time_Date`

A series of ESQL statements is terminated with an **END** statement while the module itself is ended with: `END MODULE;`

The ESQL embedded between the `BEGIN` and `END` statements constitutes a program and is run concurrent with each message processed by the parent node.

Instructor notes:

Purpose — Expose the notion that some nodes are *programmable,* some with ESQL statements.

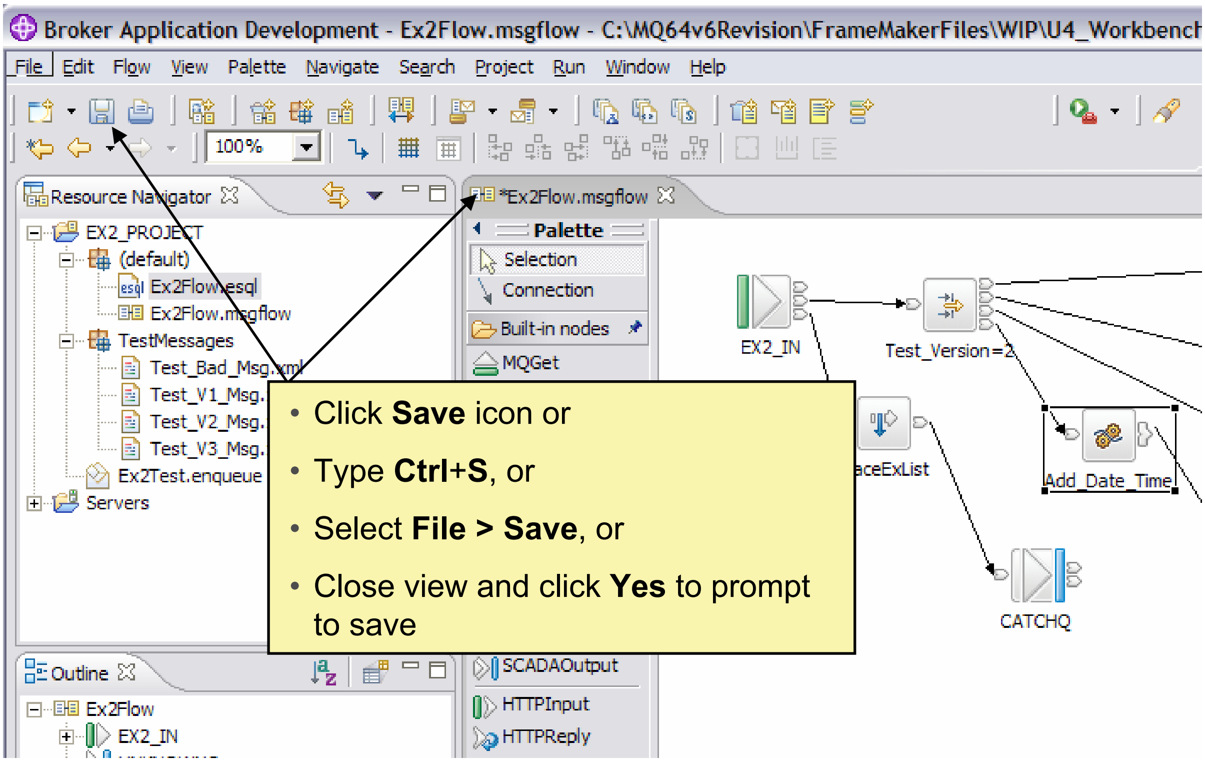
Details — Do not spend much time on ESQL. If the student wants to learn more about the this topic, they should be directed to the WebSphere Message Broker Developer Workshop (WM663/VM663).

Additional information — Draw a distinction between customizing a node, through Properties, and programming a node, here with ESQL statements. It might also be wise to point out that only a few nodes have this ESQL programmable feature (for example, filter, compute, and database).

Transition statement — Adding or changing the details of a message flow or an ESQL file requires a save for the changes to become permanent on the local file system.

WebSphere Education

Save your work



- Click **Save** icon or
- Type **Ctrl+S**, or
- Select **File > Save**, or
- Close view and click **Yes** to prompt to save

Figure 3-26. Save your work WM643 / VM6431.0

Notes:

Working in the ESQL and Message Flow editors requires a save to transfer your work from memory to local storage. The save can be invoked in one of several ways, as identified in the figure.

ESQL syntax errors (as shown in the previous figure) are not reported in the **Problems** view until a save is initiated.

Similarly, message flow errors are shown by the addition of a decorator attached to the node instance considered in error.

In the next topic, you learn how to deliver a message flow to a broker for execution.

Instructor notes:

Purpose — Make the point that working with a toolkit editor has no effect on local storage until a save is performed.

Details — None.

Additional information — Remind students that with the Eclipse tooling, all their work eventually ends up in a file on their local file system, but only if they save it. However, the program is somewhat forgiving in the sense that it prompts the user to save when it recognizes work in progress (which is common among editors in general).

Transition statement — The next topic examines the techniques available to deliver work to a broker for execution, a process called *deployment.*

However, you can pause here and get started on Exercise 2 (Using the WebSphere Message Broker Toolkit) while the information about configuration and message flow development is still fresh. When class reconvenes, you learn about deployment and testing, and then complete Exercise 2.

3.3. Deploying message flows

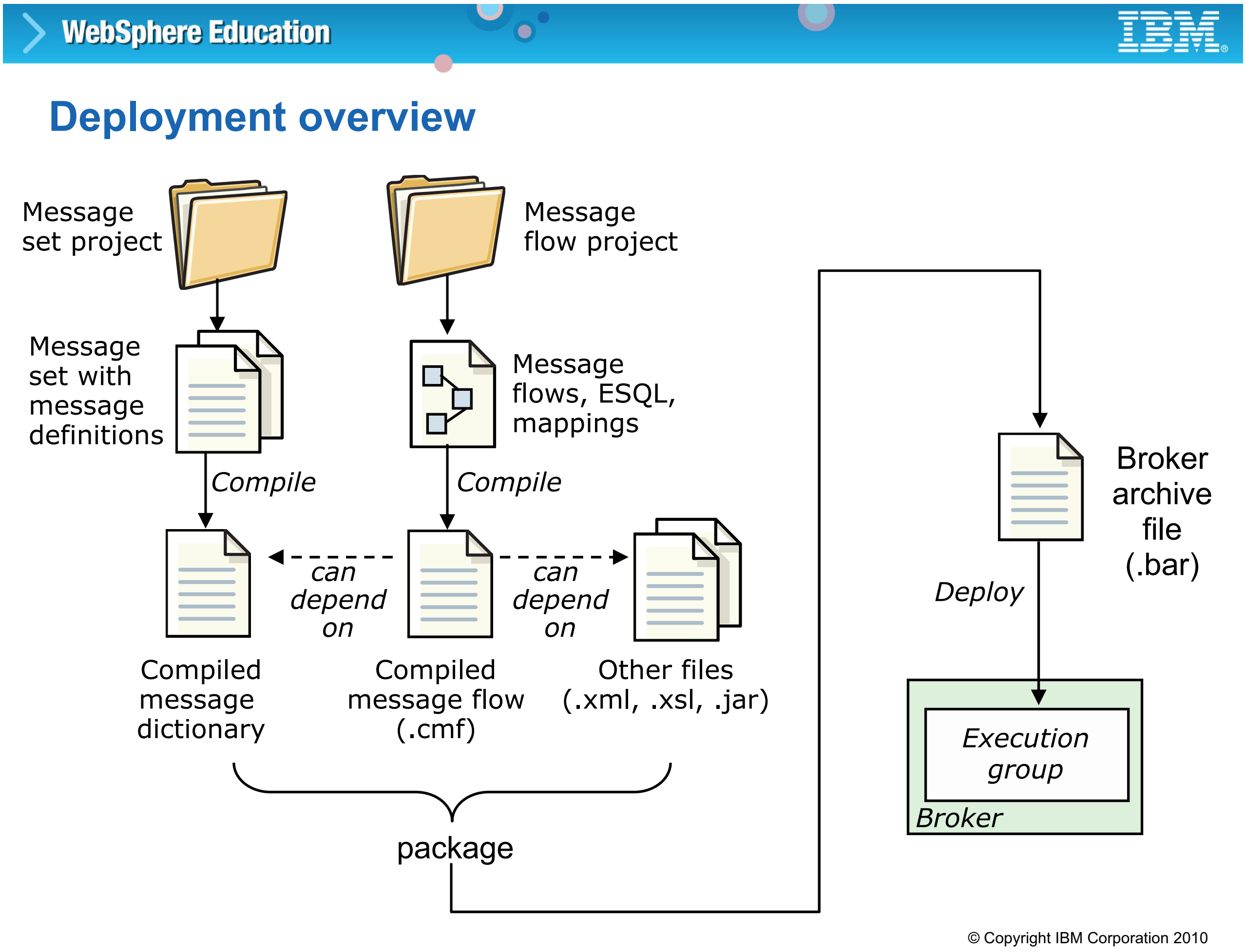
Instructor topic introduction

What students will do — Learn how to transfer work from the development environment to the runtime environment, that is, deploy to a broker.

How students will do it — Listen to the lecture and participate in classroom discussion. A later exercise (Exercise 2) will give the students the opportunity to practice the procedures and operations covered in this topic.

What students will learn — Deploying, or transferring work to a broker, involves the use of a broker archive file, also known as a `.bar` file.

How this will help students on their job — Deploying to a broker is fundamental to WebSphere Message Broker operations. It is one of several basic tasks described in this unit. In a production environment, deploying might be an administrative task (as opposed to a development environment, where the developer himself likely performs the deployment).



WebSphere Education

Deployment overview

Message set project

Message set with message definitions

Compile

Compiled message dictionary

Message flow project

Message flows, ESQL, mappings

Compile

Compiled message flow (.cmf)

can depend on

can depend on

Other files (.xml, .xsl, .jar)

package

Broker archive file (.bar)

Deploy

Execution group

Broker

© Copyright IBM Corporation 2010

Figure 3-27. Deployment overview WM643 / VM6431.0

Notes:

Deployment is the process of moving message flow components (and possibly message set components) from the WebSphere Message Broker Toolkit into a broker environment for execution.

A deployment operation takes some time; exactly how much depends on the nature of network connections and the volume of information being exchanged.

Administrators have some control over the volume of information exchanged in that they can select between a complete or incremental deployment.

- In a **complete** deployment, files that are already deployed to the execution group are removed before the complete contents of the `.bar` file are deployed. Therefore, nothing is left in the execution group from any previous deployment.
- In an **incremental** deployment, deployed files are added to the execution group. Files that exist in the execution group are replaced by newer versions.

As expected, efficiency can be achieved with respect to the packaging of the deployment data. Reducing the volume, by compression, reduces the exchange time.

Instructor notes:

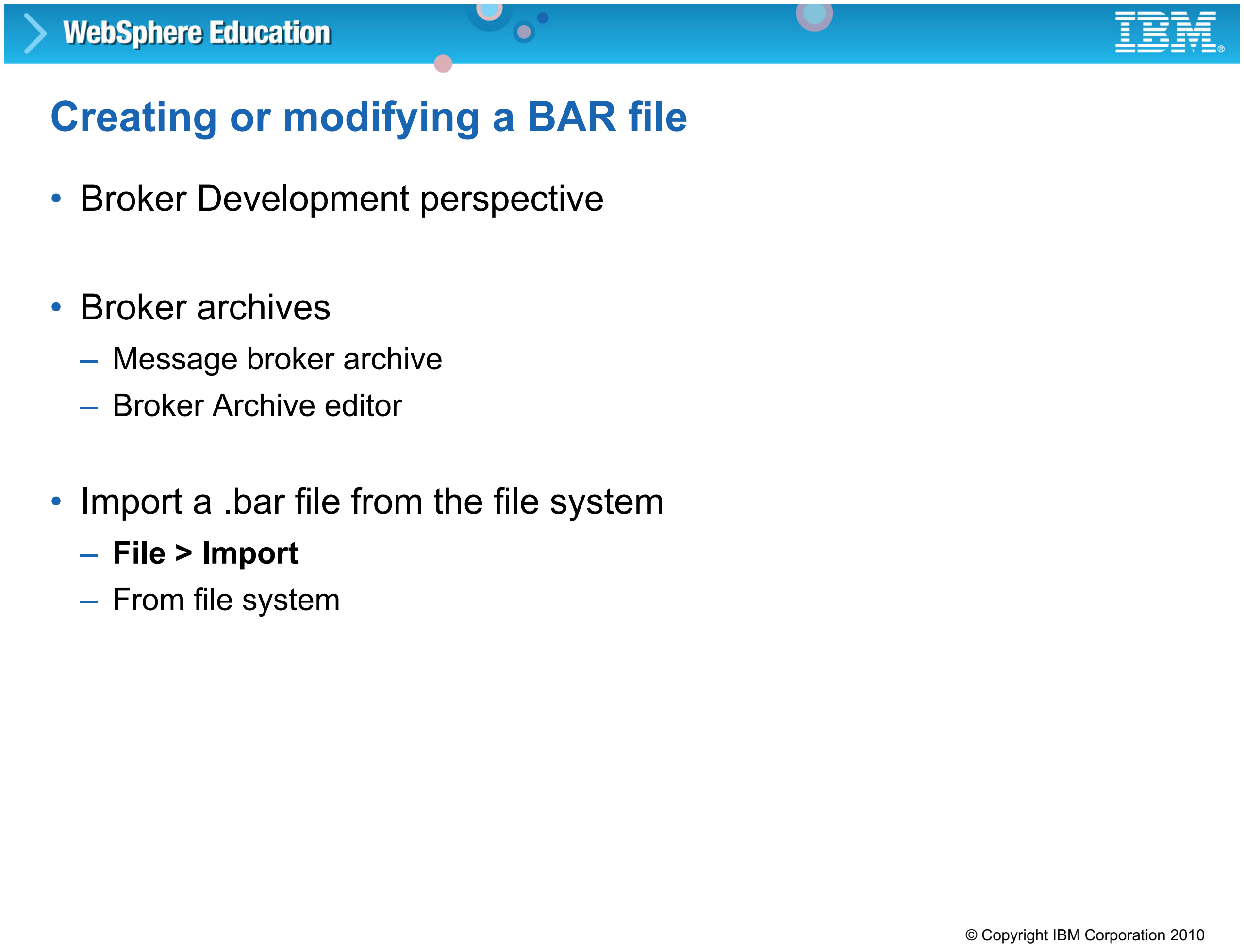
Purpose — Highlight the scope of the deployment process.

Details — Be sure to explain the difference between the deployment types and why you might choose one over the other:

- When ***not*** to use complete deployment: if you want to merge the existing contents of the execution group with the contents of the `.bar` file. This can only be done by using an incremental deployment.
- When ***not*** to use incremental deployment: if you want to completely clear the contents of the execution group before the `.bar` file is deployed. This can only be done by using a complete deployment.

Additional information — None.

Transition statement — So, what options exist regarding the use of a `.bar` file?



WebSphere Education

Creating or modifying a BAR file

- Broker Development perspective
- Broker archives
  - Message broker archive
  - Broker Archive editor
- Import a .bar file from the file system
  - **File > Import**
  - From file system

Figure 3-28. Creating or modifying a BAR file WM643 / VM6431.0

Notes:

BAR files are normally managed and stored in a project. You can create a BAR file or import a BAR file that has previously been built.

After a BAR file exists, further usability options are available through the `broker.xml` file.

Instructor notes:

Purpose — Orient the student as to where in the WebSphere Message Broker Toolkit BAR files are created and managed.

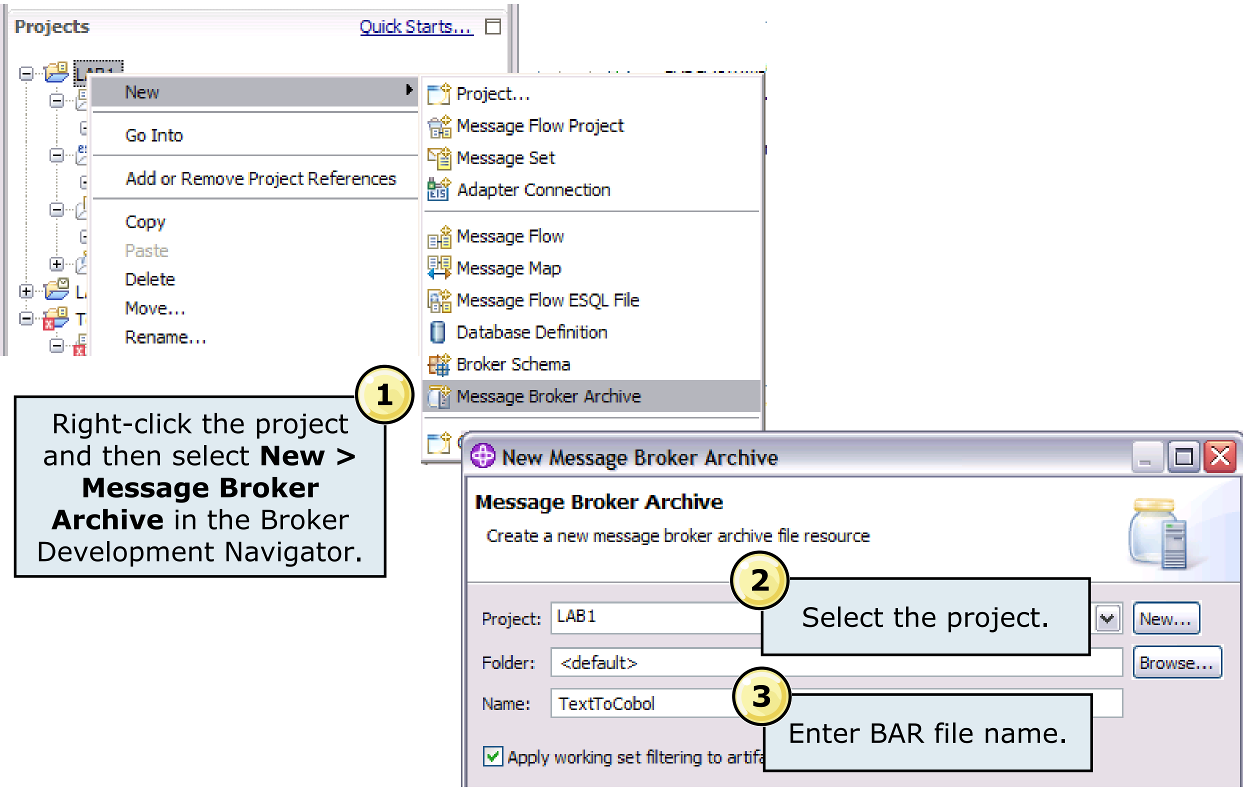
Details — None.

Additional information — None.

Transition statement — A look at the Broker Administration perspective helps put things in focus.

WebSphere Education

Creating the BAR file (1 of 2)



Right-click the project and then select **New > Message Broker Archive** in the Broker Development Navigator.

Select the project.

Enter BAR file name.

Figure 3-29. Creating the BAR file (1 of 2) WM643 / VM6431.0

Notes:

The Broker Application Development **Projects** view contains a subtree called **Broker Archives**. This element (among other things) serves as a container for broker archive (`.bar`) files.

You can create a broker archive file in the WebSphere Message Broker Toolkit or using the `mqsicreatebar` command.

To create a broker archive file in the WebSphere Message Broker Toolkit:

1. Right-click a project and then select **New > Message Broker Archive**.
2. In the **New Message Broker Archive** dialog, select the project that will contain the broker archive file.
3. Enter the broker archive file name. The broker archive file name is added to the **Broker Archives** subtree in the Projects view and the BAR File editor opens on the **Prepare** view.

Instructor notes:

Purpose — Identify the broker archives subtree and one of the ways to start the wizard to build a broker archive.

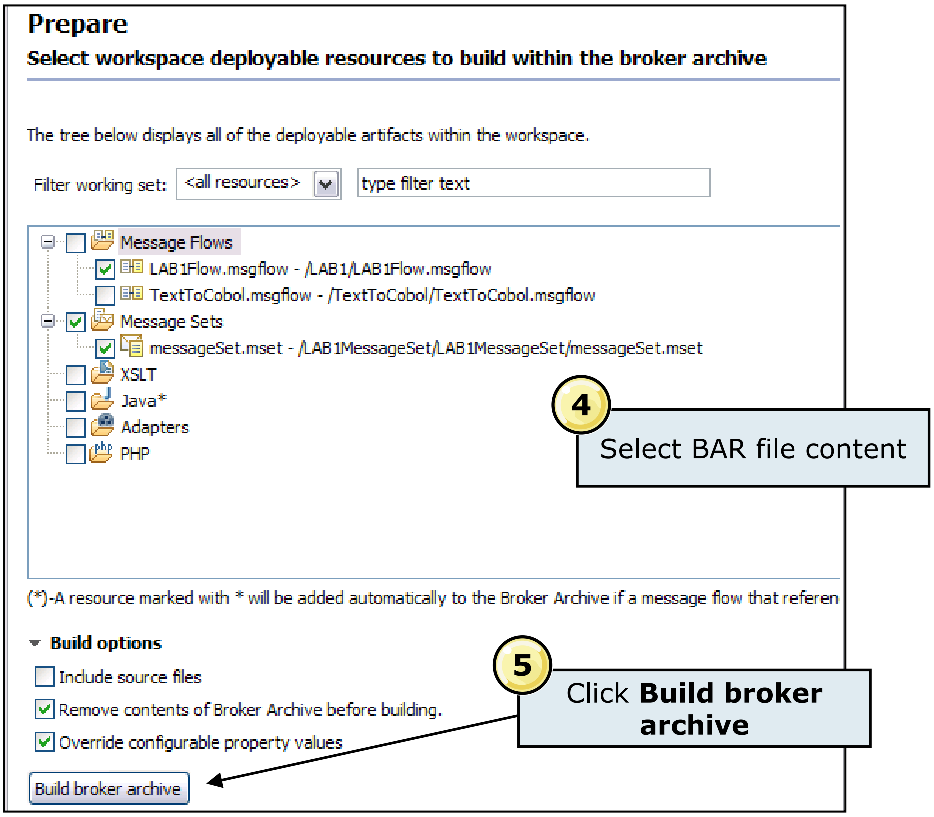
Details — None.

Additional information — None.

Transition statement — Next look at the BAR File editor where the composition of the `.bar` file is established.

WebSphere Education

Creating the BAR file (2 of 2)



Prepare

Select workspace deployable resources to build within the broker archive

The tree below displays all of the deployable artifacts within the workspace.

Filter working set: <all resources> type filter text

- □ Message Flows
  - ☑ LAB1Flow.msgflow - /LAB1/LAB1Flow.msgflow
  - □ TextToCobol.msgflow - /TextToCobol/TextToCobol.msgflow
- ☑ Message Sets
  - ☑ messageSet.mset - /LAB1MessageSet/LAB1MessageSet/messageSet.mset
- □ XSLT
- □ Java*
- □ Adapters
- □ PHP

(*)-A resource marked with * will be added automatically to the Broker Archive if a message flow that referen

Build options

□ Include source files

☑ Remove contents of Broker Archive before building.

☑ Override configurable property values

Build broker archive

4 Select BAR file content

5 Click **Build broker archive**

Figure 3-30. Creating the BAR file (2 of 2) WM643 / VM6431.0

Notes:

The **Prepare** view displays the options for creating a BAR file. All available resources in the current workspace (including Java, XML and XSL/XSLT files, message flows, message sets, and adapters) are included with each build type.

The next steps in creating a BAR file are to:

4. Select the artifacts to include in the BAR file.

5. Click **Build broker archive**.

The Prepare view includes some additional build options:

- **Include source files** to include source files in the BAR file.
- **Remove contents of Broker Archive before building** to remove all existing contents of the broker archive file before building the new broker archive file.
- **Override configurable properties values** to override any value set by the flow compiler, that is, the values from the message flow. If you do not check this option the properties in the `broker.xml` file are used when compiling a message flow.

Instructor notes:

Purpose — Define the contents of the BAR file.

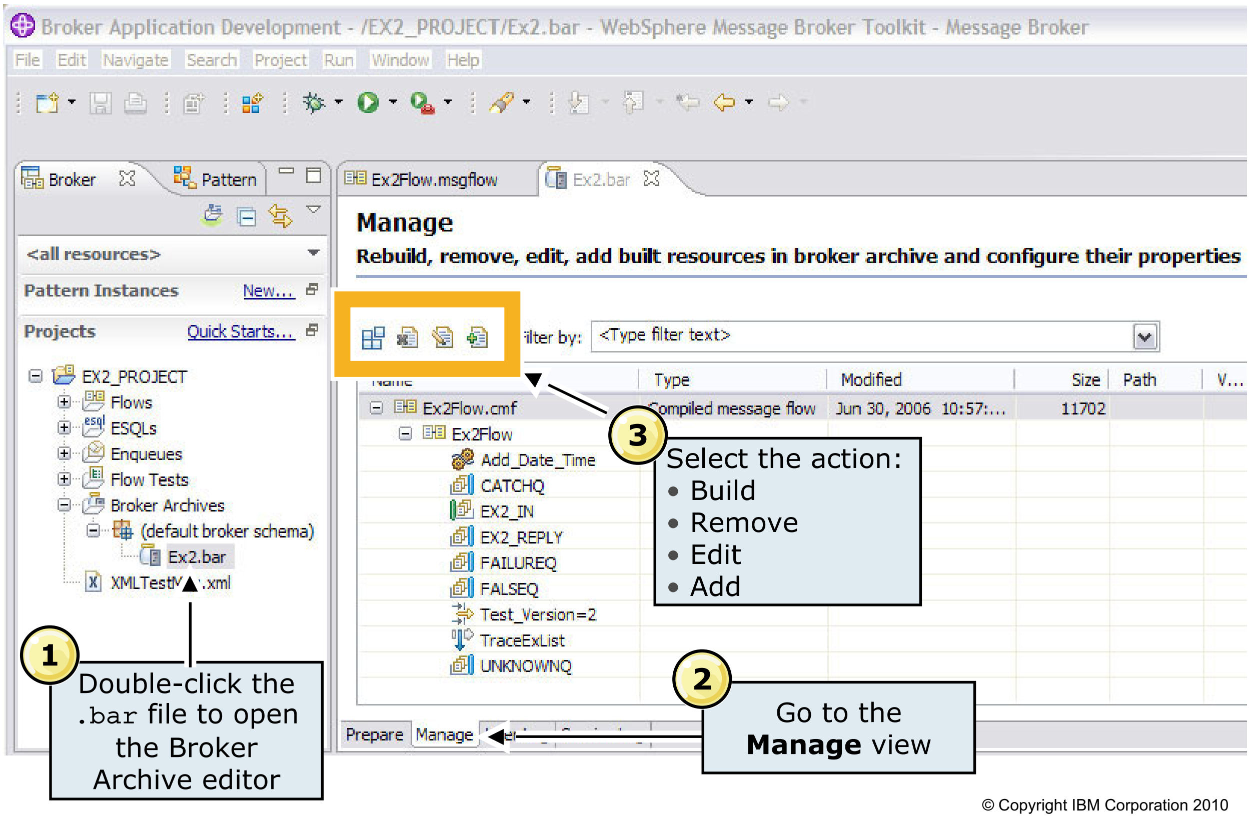
Details — None.

Additional information — None.

Transition statement — You also modify the BAR file in the Broker Archive editor.

WebSphere Education

Modifying a BAR file



Manage

Rebuild, remove, edit, add built resources in broker archive and configure their properties

Name	Type	Modified	Size	Path	V...
Ex2Flow.cmf	Compiled message flow	Jun 30, 2006 10:57:...	11702		
Ex2Flow					
Add_Date_Time					
CATCHQ					
EX2_IN					
EX2_REPLY					
FAILUREQ					
FALSEQ					
Test_Version=2					
TraceExList					
UNKNOWNQ					

1 Double-click the .bar file to open the Broker Archive editor

2 Go to the **Manage** view

3 Select the action:
- Build
- Remove
- Edit
- Add

© Copyright IBM Corporation 2010

Figure 3-31. Modifying a BAR file WM643 / VM6431.0

Notes:

The **Manage** view displays the message flows, message sets, and other files that are currently in the BAR file. Use the icons to add, edit, and remove files from the BAR file.

To modify a BAR file:

1. Double-click the `.bar` file to open the Broker Archive editor.
2. Go to the **Manage** view.
3. Select the action.

The action icons on the toolbar in the **Manage** view are used to:

- Build a BAR file.
- Remove message flows, message sets, or other items from the BAR file.
- Edit selected message flows, message sets, or other items in the BAR file.
- Add message flows, message sets, or other items to the BAR file.

Instructor notes:

Purpose — Illustrate the Broker Archive editor.

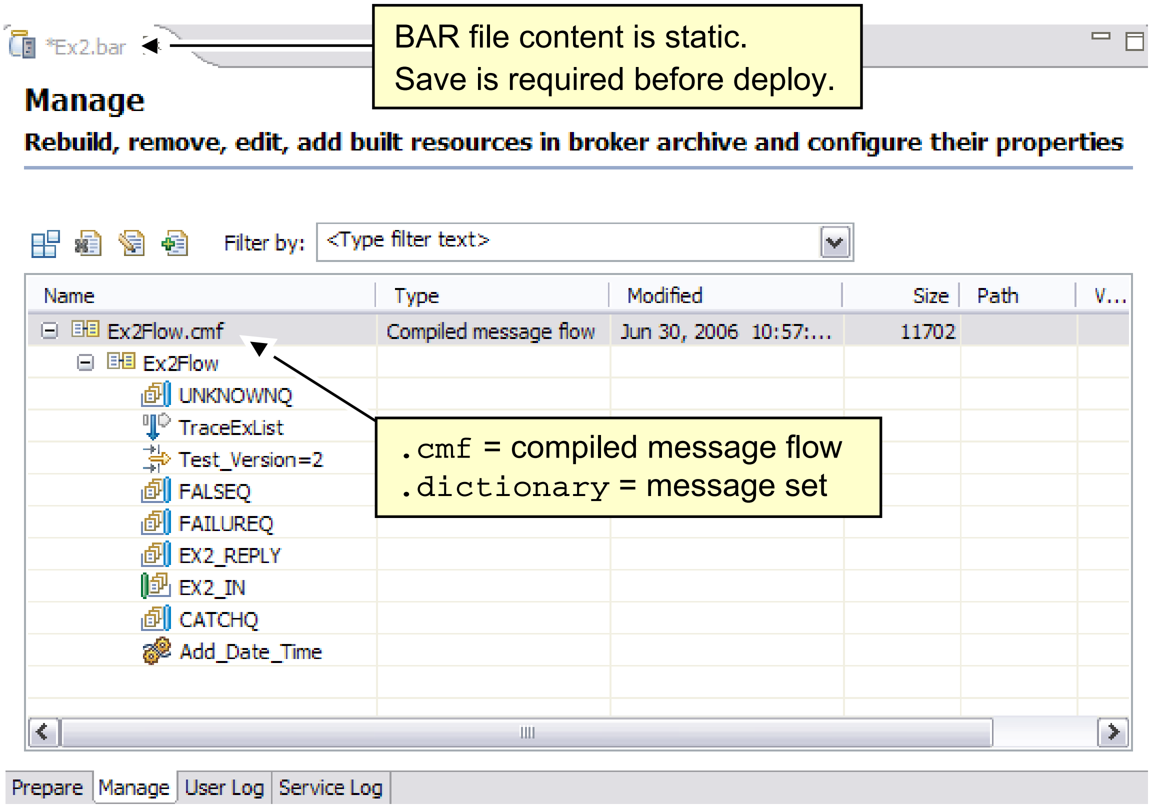
Details — None.

Additional information — None.

Transition statement — Now, you will add some content to the `.bar` file.

WebSphere Education

Creating or modifying a BAR file



*Ex2.bar

BAR file content is static.
Save is required before deploy.

Manage

Rebuild, remove, edit, add built resources in broker archive and configure their properties

Filter by: <Type filter text>

Name	Type	Modified	Size	Path	V...
⊟ Ex2Flow.cmf	Compiled message flow	Jun 30, 2006 10:57:...	11702		
⊟ Ex2Flow					
UNKNOWNQ					
TraceExList					
Test_Version=2					
FALSEQ					
FAILUREQ					
EX2_REPLY					
EX2_IN					
CATCHQ					
Add_Date_Time					

`.cmf` = compiled message flow
`.dictionary` = message set

Figure 3-32. Creating or modifying a BAR file WM643 / VM6431.0

Notes:

The results of the compilation yield an addition to the `.bar` file, that is, a `.cmf` file (*compiled message flow*) for the previously selected message flow. The selection process can be repeated as needed to add more resources to the BAR file.

Save the BAR file after your modifications are complete. If you attempt to proceed with the deployment without saving, you are prompted to save the content as shown in the Broker Archive editor.

Instructor notes:

Purpose — Make the association between building a `.bar` file and committing it to the local file system.

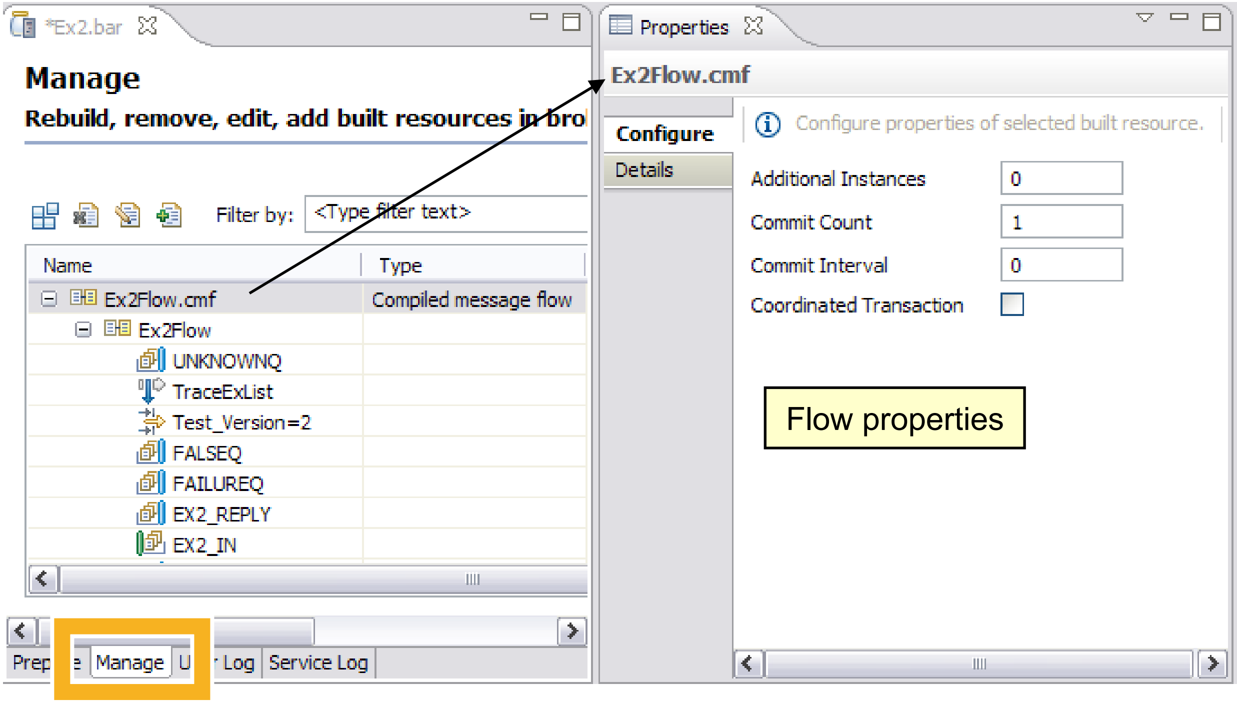
Details — None.

Additional information — None.

Transition statement — Select the **Configure** tab to see what else the BAR file editor has to offer.

WebSphere Education

BAR message flow properties



Manage

Rebuild, remove, edit, add built resources in bro

Filter by: <Type filter text>

Name	Type
Ex2Flow.cmf	Compiled message flow
Ex2Flow	
UNKNOWNQ	
TraceExList	
Test_Version=2	
FALSEQ	
FAILUREQ	
EX2_REPLY	
EX2_IN	

Ex2Flow.cmf

Configure | Details

Configure properties of selected built resource.

Additional Instances	0
Commit Count	1
Commit Interval	0
Coordinated Transaction	□

© Copyright IBM Corporation 2010

Figure 3-33. BAR message flow properties WM643 / VM6431.0

Notes:

The **Manager** view of the Broker Archive editor permits the setting of various variables that give the BAR file additional or alternate runtime properties. In this manner, the content of the BAR file (that is, message flows and message sets) can remain unchanged, meaning that the selection process and the compilation process do not have to be repeated. However, subsequent deployments release packages with attributes that vary from one package to another.

These system-defined properties that can be modified without changing the underlying message flow itself are known as “configurable properties”. Another type of property, known as a user-defined property (UDP), can be defined by message flow developers when they are creating the message flow. A UDP value can also be changed at deployment time.

Configurable properties eliminate the need to modify message flow source code (with respect to certain node properties) as the flow is migrated through the life cycle (for example, development to test to production).

In this example, the message flow has the following configurable properties:

- **Additional instances** is set to the number of additional threads that the broker can use to service the message flow. Additional threads are created only if there are sufficient input messages. You can use up to 256 threads per message flow. The default is `0`.
- **Commit Count** is set to the number of input messages that are processed by a message flow before a sync point is taken. The default is `1`.
- **Commit Interval** is the time interval at which a commit is taken when the **Commit Count** property is greater than 1 (that is, where the message flow is batching messages).
- **Coordinated Transaction** specifies whether message flow includes one or more nodes that access databases that are to participate in a globally coordinated transaction.

Instructor notes:

Purpose — Identify the configurable runtime options associated with a `.bar` file.

Details — None.

Additional information — Spend just a moment explaining each of the options shown in the figure: **Coordinated Transaction**, **Commit Interval**, and so on. As the audience is made up of administrators, it is likely that these settable attributes are in their specific area of interest.

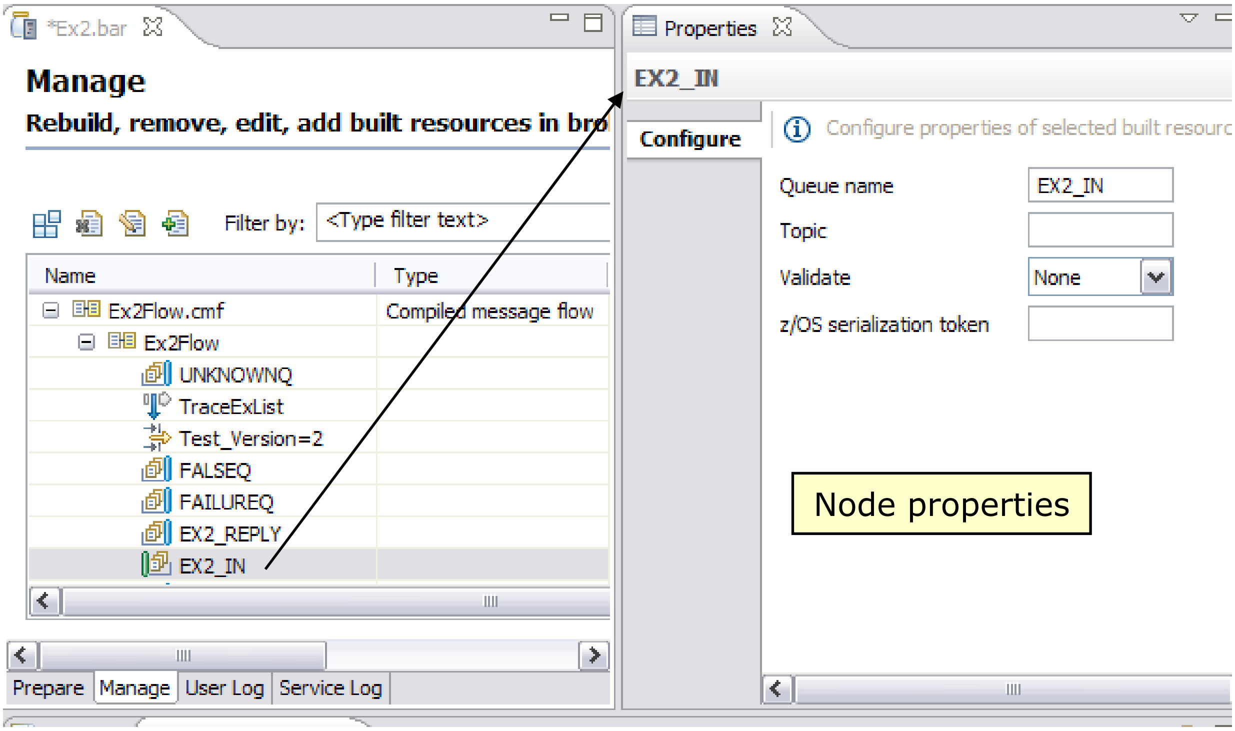
You can relate additional instances to what you already presented in Unit 2.

Also point out which of these attributes will be further explained later in the class. For example, you will revisit **Coordinated Transaction**, **Commit Interval**, and **Commit Count** during the Problem determination unit.

Transition statement — What happens when a flow migrates out of the development environment and through the next stage (for example, unit test, system test, QA test, production)? What source changes are required (if any)?

WebSphere Education

BAR node properties



*Ex2.bar

Manage

Rebuild, remove, edit, add built resources in bro

Filter by: <Type filter text>

Name	Type
Ex2Flow.cmf	Compiled message flow
Ex2Flow	
UNKNOWNQ	
TraceExList	
Test_Version=2	
FALSEQ	
FAILUREQ	
EX2_REPLY	
EX2_IN	

Properties

EX2_IN

Configure

Configure properties of selected built resourc

Queue name: EX2_IN

Topic:

Validate: None

z/OS serialization token:

Node properties

Figure 3-34. BAR node properties WM643 / VM6431.0

Notes:

Consider the case of a simple message flow. It uses an input queue with a (development) name of `EX2_IN`. On the test system, the corresponding input queue might be named `TEST_01_IN`. In the **Manage** tab of the Broker Archive editor, right-clicking the MQInput node name displays the node properties where you can override the associated input queue. In this manner, an administrator can alter the nature of the message flow by replacing the developer's input queue name (`EX2_IN`) with that of the test system (`TEST_01_IN`). Likewise, when the message flow migrates to the production system, the input queue name can be changed. You can also view the properties that can be configured for your message flows by selecting **Configurable properties** from the **Filter by** list.

In addition to queue names, the data source (that is, the data base name for nodes that permit data base interaction) can also be modified using the **Manage** tab.

The collection of configurable properties of a message flow is referred to as the `.bar` file *deployment descriptor*; they are recorded in an XML-format file named `broker.xml` and stored in the workspace associated with the broker archive.

Instructor notes:

Purpose — Identify the `.bar` file deployment descriptor and the utility it provides as message flows migrate from one runtime environment to another, such as test to production.

Details — None.

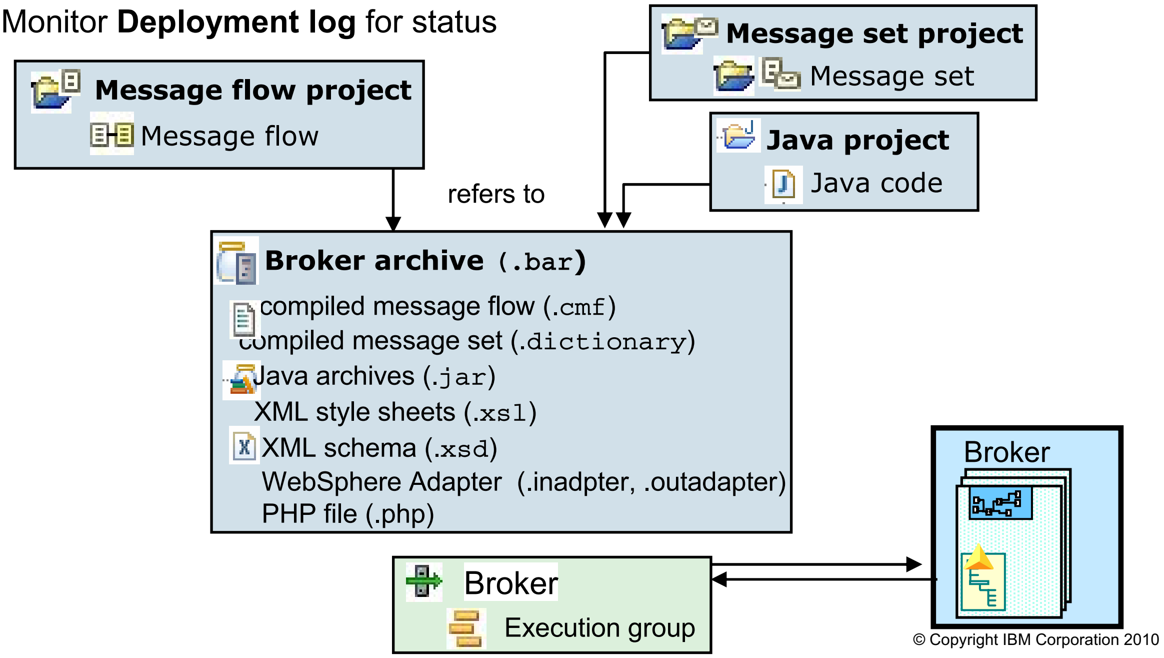
Additional information — None.

Transition statement — Having completed the BAR file, the next step is to deliver it to the broker by initiating a deployment operation.

WebSphere Education

Compiling and deploying

- Create BAR file and include contents
- Configure queues, DSNs for broker environment (deployment descriptor)
- Deploy by dragging BAR file into execution group or using command for script
- Monitor **Deployment log** for status



Message flow project
Message flow

Message set project
Message set

Java project
Java code

refers to

Broker archive (.bar)
compiled message flow (.cmf)
compiled message set (.dictionary)
Java archives (.jar)
XML style sheets (.xsl)
XML schema (.xsd)
WebSphere Adapter (.inadpter, .outadapter)
PHP file (.php)

Broker
Execution group

Broker

© Copyright IBM Corporation 2010

Figure 3-35. Compiling and deploying WM643 / VM6431.0

Notes:

The deployment process sends the BAR file to a broker. The BAR file is a compressed file which can contain a number of deployment resources:

- One `.cmf` file for each message flow. This is a compiled version of the message flow.
- One `.dictionary` file for each message set dictionary.
- One `broker.xml` broker deployment descriptor file in XML format. This file is in the META-INF folder of the compressed file and can be modified using a text editor or shell script.
- Any number of XML files (`.xml`) and style sheets (`.xsl` files) for use with the XSLTransform node.
- Any number of `.jar` files for use with the JavaCompute node.
- Any number of `.inadapter` or `.outadapter` files for use with the WebSphere Adapter nodes.
- Any number of `.php`, files for use with the PHPCompute node.

As a compressed archive, the BAR file can also contain any additional files you need. For example, you might want to include source files or test data for future reference. Only the file types listed in the figure are added to the execution group, replacing any deployed objects with the identical name.

Instructor notes:

Purpose — Identify the two ways to initiate a deployment.

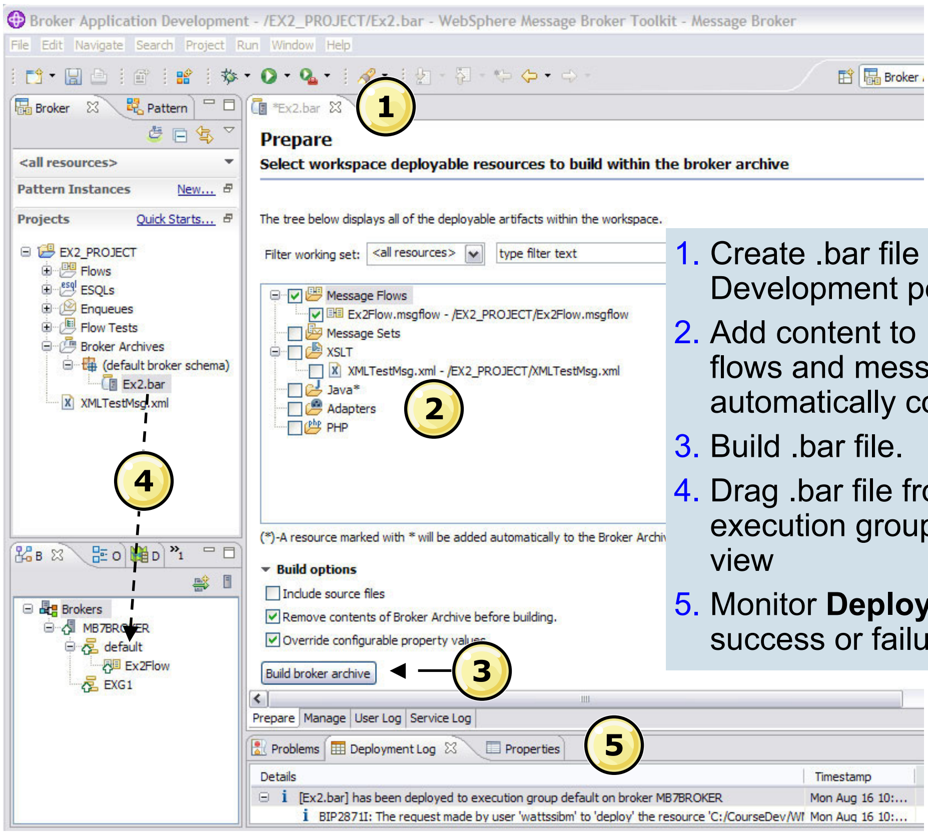
Details — None.

Additional information — None.

Transition statement — A wizard permits a broker or execution group to be selected as the target for the deployment.

WebSphere Education

Deploying



Broker Application Development - /EX2_PROJECT/Ex2.bar - WebSphere Message Broker Toolkit - Message Broker
File Edit Navigate Search Project Run Window Help
Broker
Pattern
*Ex2.bar
1
2
3
4
5

Prepare

Select workspace deployable resources to build within the broker archive

The tree below displays all of the deployable artifacts within the workspace.

Filter working set: <all resources> type filter text

- ☑ Message Flows
  - ☑ Ex2Flow.msgflow - /EX2_PROJECT/Ex2Flow.msgflow
- □ Message Sets
- □ XSLT
  - □ XMLTestMsg.xml - /EX2_PROJECT/XMLTestMsg.xml
- □ Java*
- □ Adapters
- □ PHP

(*)-A resource marked with * will be added automatically to the Broker Archi

Build options

- □ Include source files
- ☑ Remove contents of Broker Archive before building.
- ☑ Override configurable property values

Build broker archive

Prepare | Manage | User Log | Service Log

<all resources>

Pattern Instances New...

Projects Quick Starts...

- EX2_PROJECT
  - Flows
  - ESQLs
  - Enqueues
  - Flow Tests
  - Broker Archives
    - (default broker schema)
      - Ex2.bar
  - XMLTestMsg.xml

- Brokers
  - MB7BROKER
    - default
      - Ex2Flow
    - EXG1

Problems | Deployment Log | Properties

Details	Timestamp
[Ex2.bar] has been deployed to execution group default on broker MB7BROKER	Mon Aug 16 10:...
BIP2871I: The request made by user 'wattssibm' to 'deploy' the resource 'C:/CourseDev/WI	Mon Aug 16 10:...

1. Create .bar file in Broker Development perspective.
2. Add content to .bar file: Message flows and message sets are automatically compiled
3. Build .bar file.
4. Drag .bar file from navigator onto execution group (deploy) in **Brokers** view
5. Monitor **Deployment Log** for success or failure

Figure 3-36. Deploying WM643 / VM6431.0

Notes:

The first task in deploying is to populate the `.bar` file with artifacts from the workspace on the **Prepare** view.

Next, select the resources that you need and then click **Build broker archive** to create the `.bar` file.

There are a number of ways to deploy the BAR file. One way is to select the `.bar` file in the Broker Development navigator and drag-and-drop it onto the execution group in the **Brokers** view. Alternatively, you can right-click an execution group to select a broker archive file to deploy to the selected execution group.

You can deploy message flows directly onto execution groups without having to build BAR files or change perspectives. Deployment results are displayed synchronously in a Deployment log, which lets you quickly verify that message flows are deployed and working as expected.

Instructor notes:

Purpose — Show the second of the two techniques to initiate a deployment.

Details — None.

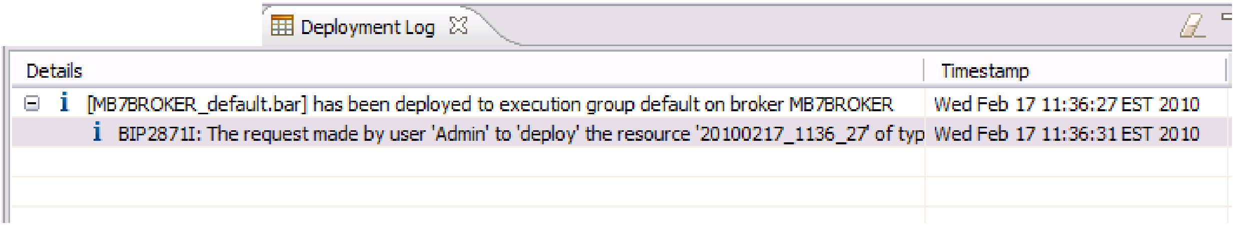
Additional information — None.

Transition statement — After the `.bar` file has been constructed, now it is necessary to perform the deployment.

WebSphere Education

Deployment Log

- Shows the result of deployment actions on brokers
  - Message number
  - Name of the broker that the deployment message has come from
  - Detail of the deployment message
- **Clear Deployment Log**
  - Clears all log entries from the Deployment Log view
  - Does not delete the log entries from the Administration log in Message Broker Explorer
- Messages are automatically cleared after 72 hours.



Details	Timestamp
[MB7BROKER_default.bar] has been deployed to execution group default on broker MB7BROKER	Wed Feb 17 11:36:27 EST 2010
BIP2871I: The request made by user 'Admin' to 'deploy' the resource '20100217_1136_27' of typ	Wed Feb 17 11:36:31 EST 2010

Figure 3-37. Deployment Log

WM643 / VM6431.0

Notes:

The **Deployment Log** view shows the result of deployment actions on brokers in the WebSphere Message Broker Toolkit. Use the **Deployment Log** view to verify that the message flow has been deployed successfully.

The **Details** column displays the message number, the name of the broker that the deployment message has come from, and the detail of the deployment message.

Instructor notes:

Purpose — Describe the Deployment Log.

Details —

Additional information — Deployment information is contained the Administration log in the Message Broker Explorer.

Transition statement — You can also use MQSI commands to build a `.bar` file and deploy the `.bar` file.

3.4. Testing message flows

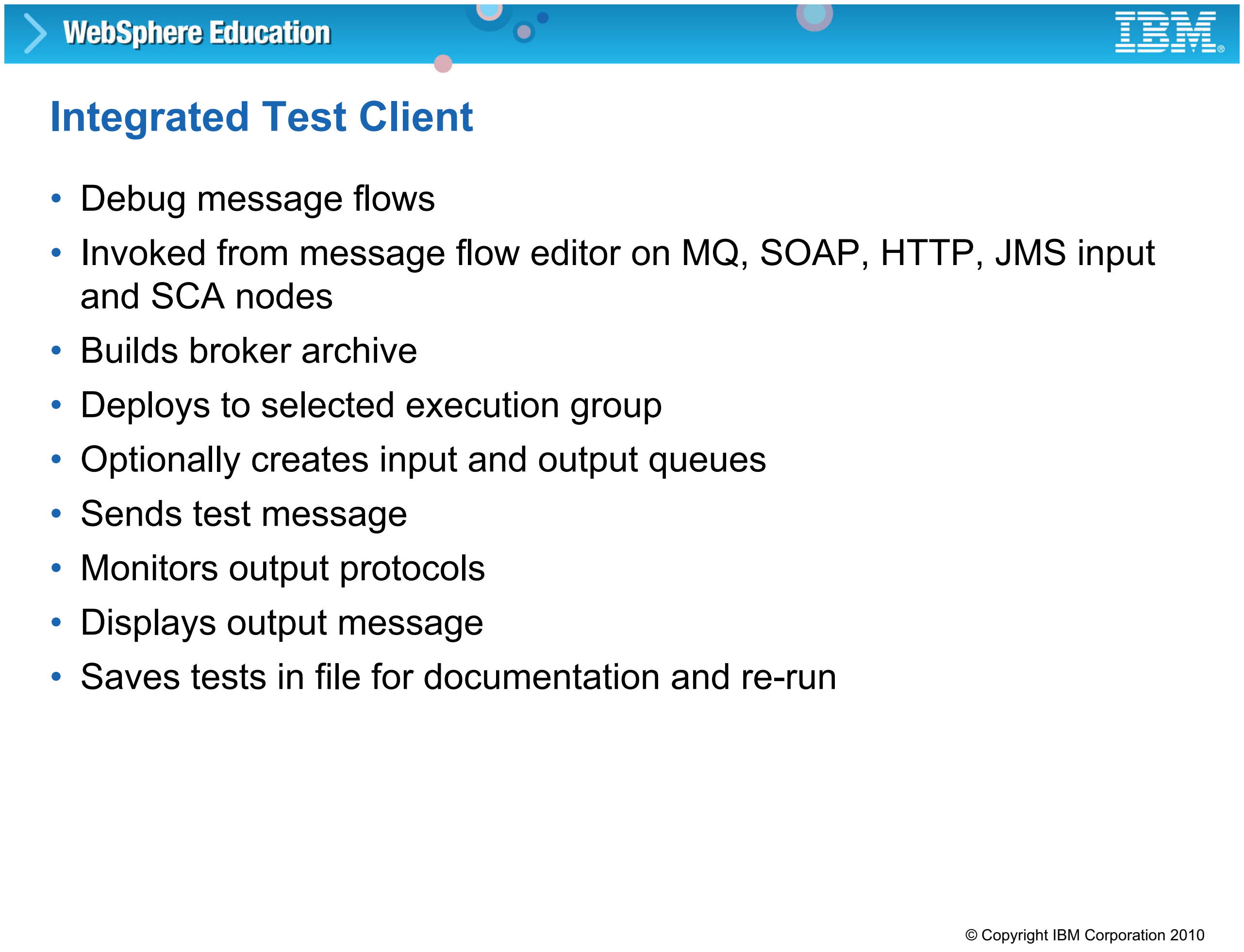
Instructor topic introduction

What students will do — Become acquainted with Message Broker Toolkit functions designed to integrate testing along with development and administration.

How students will do it — Listen to the lecture and participate in classroom discussion. The exercise (Exercise 2) gives students the opportunity to practice the procedures and operations covered in this topic.

What students will learn — To locate and use the various test tools built into the Message Broker Toolkit.

How this will help students on their job — Even administrators may be called upon to assist developers in debugging message flows. Thorough testing leads to stable systems.



WebSphere Education

Integrated Test Client

- Debug message flows
- Invoked from message flow editor on MQ, SOAP, HTTP, JMS input and SCA nodes
- Builds broker archive
- Deploys to selected execution group
- Optionally creates input and output queues
- Sends test message
- Monitors output protocols
- Displays output message
- Saves tests in file for documentation and re-run

Figure 3-38. Integrated Test Client

WM643 / VM6431.0

Notes:

The Integrated Test Client perform all the steps necessary to test a message flow. After some initial configuration, you can rerun the test with just one single click.

To start the Integrated Test Client:

1. Right-click the input node of the flow you are testing.
2. Select the **Test** action on the menu. This opens the test client dialog on the **Events** tab. When you right-click the message body, you are presented with a number of options, depending on the type of input node.

You can select the structure of an XML test message from the contents of message sets in your workspace and enter values for the elements. These values can be stored and reused in later test runs.

Instructor notes:

Purpose — Describe the Integrated Test Client. It is easy to configure and execute a test.

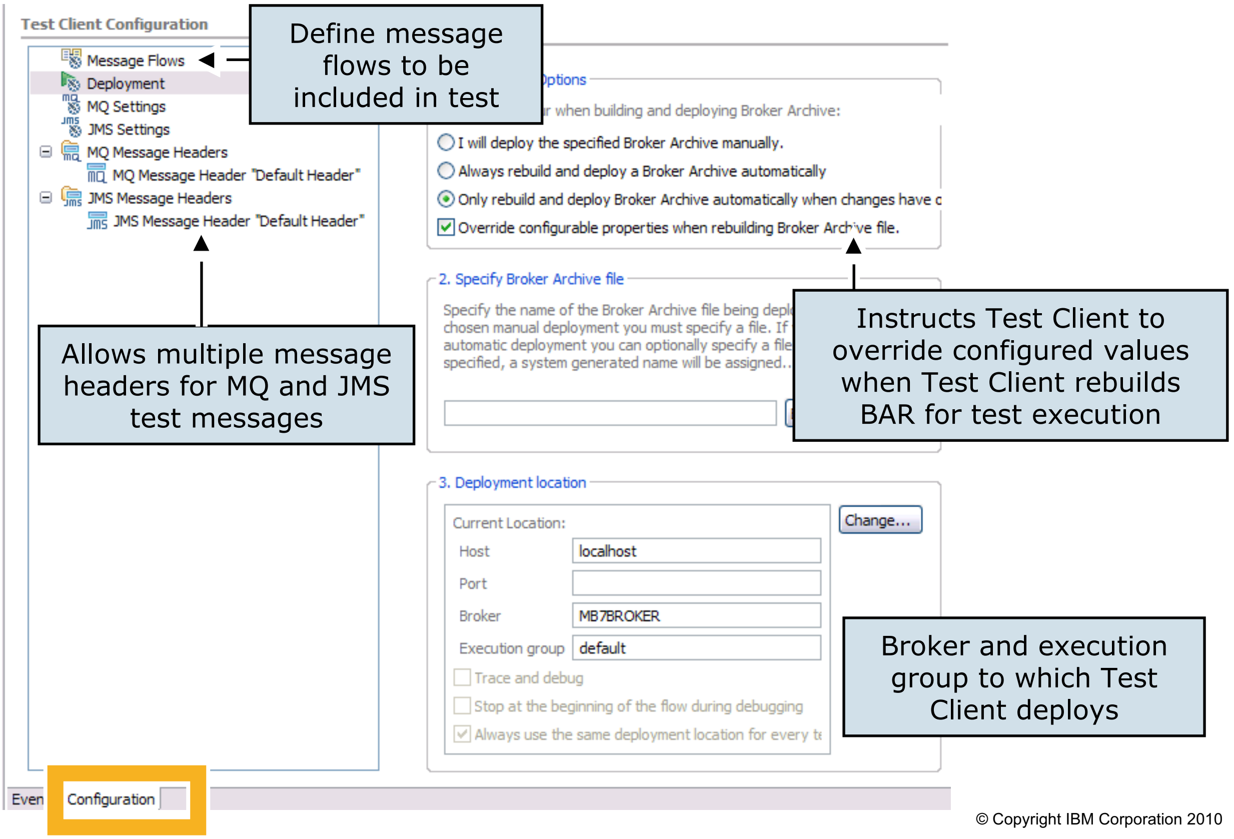
Details — Now that the message flow has been deployed to a broker, the message flow can be tested by executing it with test data. The message flow can be tested with the Integrated Test Client or RfhUtil. Both of these methods are used in the exercises in this course.

Additional information —

Transition statement — There are some configurable parameters for the Integrated Test Client. Have a look.

WebSphere Education

Integrated Test Client configuration



Test Client Configuration

- Message Flows
- Deployment
- MQ Settings
- JMS Settings
- MQ Message Headers
  - MQ Message Header "Default Header"
- JMS Message Headers
  - JMS Message Header "Default Header"

Define message flows to be included in test

Allows multiple message headers for MQ and JMS test messages

ptions

r when building and deploying Broker Archive:

○ I will deploy the specified Broker Archive manually.

○ Always rebuild and deploy a Broker Archive automatically

● Only rebuild and deploy Broker Archive automatically when changes have c

☑ Override configurable properties when rebuilding Broker Archive file.

2. Specify Broker Archive file

Specify the name of the Broker Archive file being depl chosen manual deployment you must specify a file. If automatic deployment you can optionally specify a file specified, a system generated name will be assigned.

Instructs Test Client to override configured values when Test Client rebuilds BAR for test execution

3. Deployment location

Current Location:

Host: localhost

Port:

Broker: MB7BROKER

Execution group: default

□ Trace and debug

□ Stop at the beginning of the flow during debugging

☑ Always use the same deployment location for every te

Change...

Broker and execution group to which Test Client deploys

Even | Configuration

© Copyright IBM Corporation 2010

Figure 3-39. Integrated Test Client configuration WM643 / VM6431.0

Notes:

Before you run the flow with the test client, you may need to specify certain additional parameters. This is done by switching to the **Configuration** tab.

If your main message flow needs additional message flows to execute, you must specify these in the test client.

You can specify the values for MQMD and JMS message headers. If you want to test alternative header values, you can add additional headers and select which one should be used in a test run.

You can specify how you want to the test client to manage the deployment of the message flow that is being tested. You can choose to deploy the message flow manually, or have the test client deploy if there are changes, or have the test client deploy every time.

You can manually open the BAR editor to configure some configurable properties for the message flows being tested. The **Override configurable properties when rebuilding bar file** option instructs the test client to override user-configured values when the test client rebuilds the `.bar` file during test execution.

Instructor notes:

Purpose — Walk through the test client configuration options.

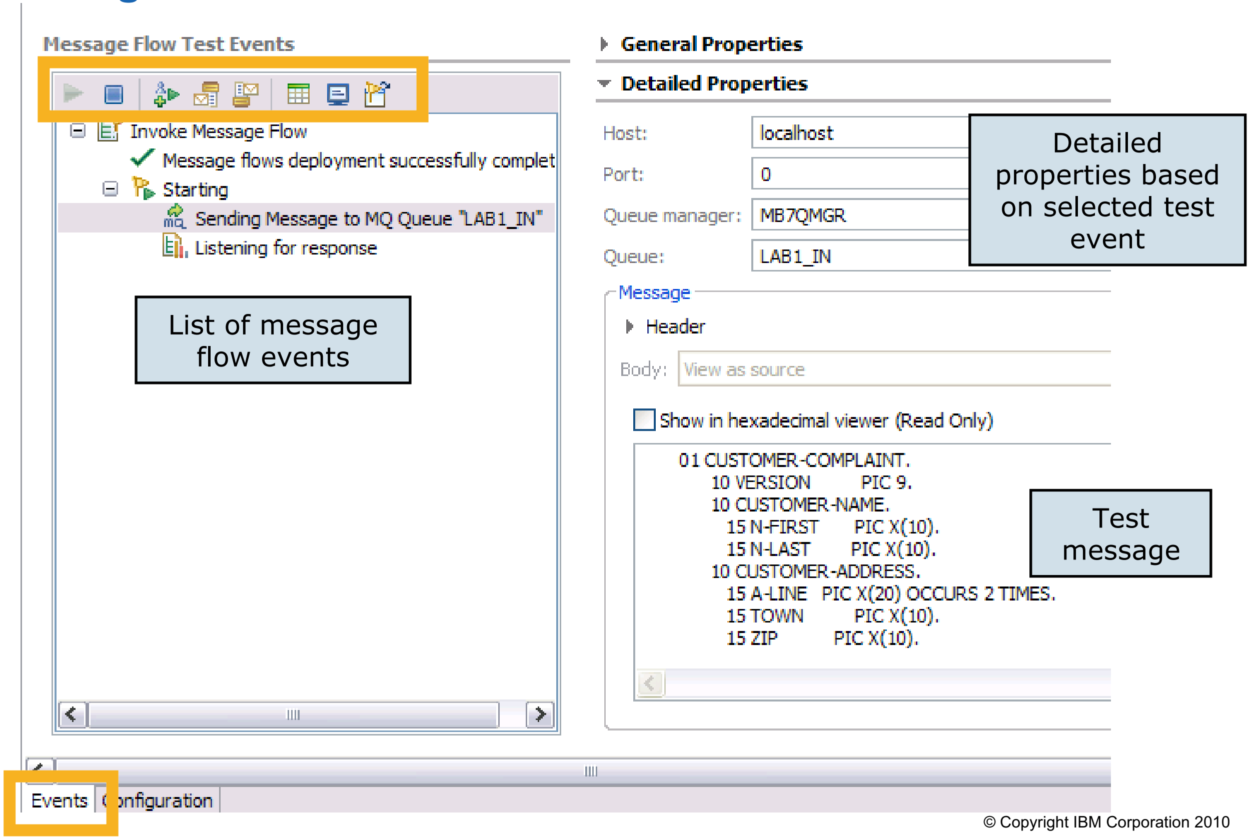
Details — The test client does not let you specify an RFH2 header explicitly. If you want to send a test message with an RFH2 header, you would configure an WebSphere MQ message header with `Format = 'MQHRF2  '` in the test client. Then you load a source test file which includes the RFH2 header. Such a file can be written from RFHUTIL, for example.

Additional information — Note the test client can be started in debug mode and offers a complete component trace. This is discussed in the unit on debugging.

Transition statement — Next: Integrated Test Client events

WebSphere Education

Integrated Test Client events



Message Flow Test Events

- Invoke Message Flow
  - ✓ Message flows deployment successfully complet
  - Starting
    - Sending Message to MQ Queue "LAB1_IN"
    - Listening for response

List of message flow events

General Properties

Detailed Properties

Host:	localhost
Port:	0
Queue manager:	MB7QMGR
Queue:	LAB1_IN

Detailed properties based on selected test event

Message

Header

Body: View as source

□ Show in hexadecimal viewer (Read Only)

```
 01 CUSTOMER-COMPLAINT.
      10 VERSION          PIC 9.
      10 CUSTOMER-NAME.
         15 N-FIRST       PIC X(10).
         15 N-LAST        PIC X(10).
      10 CUSTOMER-ADDRESS.
         15 A-LINE   PIC X(20) OCCURS 2 TIMES.
         15 TOWN          PIC X(10).
         15 ZIP        PIC X(10).
```

Figure 3-40. Integrated Test Client events WM643 / VM6431.0

Notes:

To invoke the Integrated Test Client, right-click the input node of the flow you are testing and select Test from the menu. This opens the Test Client dialog on the **Events** tab, shown in the figure.

While running the Test Client, the list of message flow events appears in on the left side of the view. The right side of the view contains the event properties and information on the test message.

The Integrated Test Client includes icons to start, stop, invoke, enqueue, dequeue, get saved messages, show console, and show Event Viewer.

Instructor notes:

Purpose — Overview of Integrated Test Client **Events** tab.

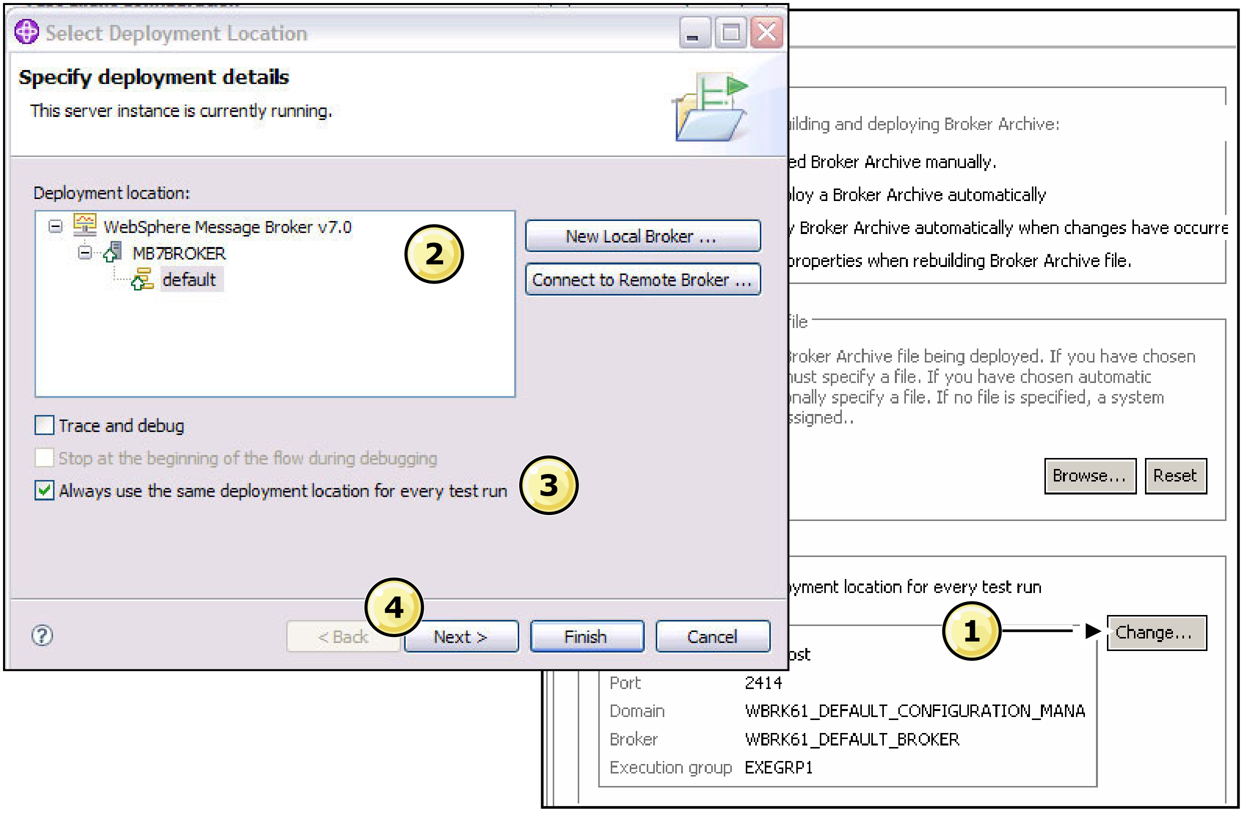
Details —

Additional information —

Transition statement — When you use the Integrated Test Client, you must identify the deployment location for the `.bar` file.

WebSphere Education

Specifying the Test Client deployment location (1 of 2)



Select Deployment Location

Specify deployment details

This server instance is currently running.

Deployment location:

- WebSphere Message Broker v7.0
  - MB7BROKER
    - default

□ Trace and debug

□ Stop at the beginning of the flow during debugging

☑ Always use the same deployment location for every test run

ilding and deploying Broker Archive:

ed Broker Archive manually.

loy a Broker Archive automatically

y Broker Archive automatically when changes have occurre

properties when rebuilding Broker Archive file.

file

roker Archive file being deployed. If you have chosen
ust specify a file. If you have chosen automatic
nally specify a file. If no file is specified, a system
ssigned..

yment location for every test run

ost

Port	2414
Domain	WBRK61_DEFAULT_CONFIGURATION_MANA
Broker	WBRK61_DEFAULT_BROKER
Execution group	EXEGRP1

Figure 3-41. Specifying the Test Client deployment location (1 of 2) WM643 / VM6431.0

Notes:

To specify the Test Client deployment location:

1. Click **Change** on the Test Client **Events** tab.
2. Specify a deployment location for a local broker or click **New local broker** to define a new local broker and deployment location.
3. Optionally, check **Always use the same deployment location** for every test run.
4. Click **Next**.

Instructor notes:

Purpose — Show the steps for specifying the deployment location.

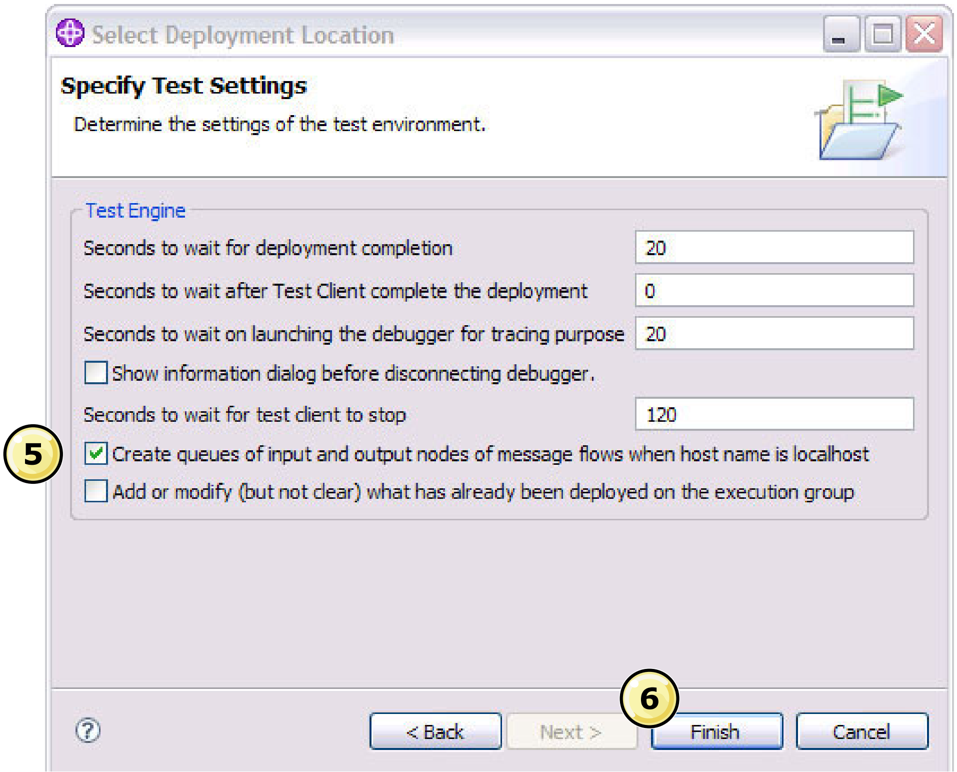
Details —

Additional information —

Transition statement — Next: Specifying the Test Client deployment location (2 of 2)

WebSphere Education IBM.

Specifying the Test Client deployment location (2 of 2)



Select Deployment Location

Specify Test Settings

Determine the settings of the test environment.

Test Engine

Seconds to wait for deployment completion: 20

Seconds to wait after Test Client complete the deployment: 0

Seconds to wait on launching the debugger for tracing purpose: 20

□ Show information dialog before disconnecting debugger.

Seconds to wait for test client to stop: 120

5 ☑ Create queues of input and output nodes of message flows when host name is localhost

□ Add or modify (but not clear) what has already been deployed on the execution group

Figure 3-42. Specifying the Test Client deployment location (2 of 2) WM643 / VM6431.0

Notes:

5. Additional configuration options are available on the **Specify Test Settings** dialog, shown in the figure.

   For example, to have the Test Client automatically create any input and output queues referenced in the test flow, check **Create queues of input and output nodes of message flows when host name is localhost**.

6. Click **Finish** to save the configuration options.

Instructor notes:

Purpose — Show the remaining steps for specifying Test Client options.

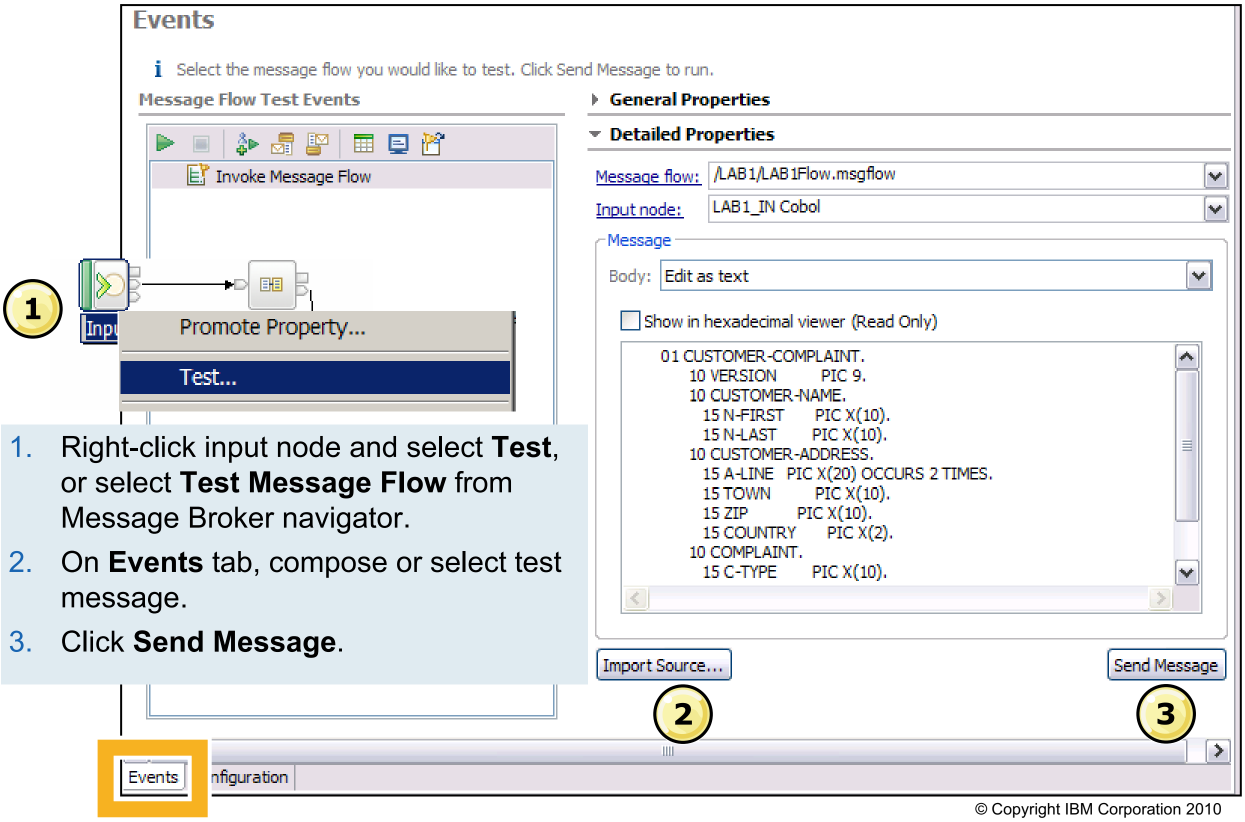
Details — Emphasize the option to automatically create the queues based on the flow.

Additional information —

Transition statement — Next: Test using the Test Client (1 of 2)

WebSphere Education

Test using the Test Client (1 of 2)



Select the message flow you would like to test. Click Send Message to run.

Message flow: /LAB1/LAB1Flow.msgflow

Input node: LAB1_IN Cobol

Body: Edit as text

□ Show in hexadecimal viewer (Read Only)

```
 01 CUSTOMER-COMPLAINT.
    10 VERSION       PIC 9.
    10 CUSTOMER-NAME.
      15 N-FIRST     PIC X(10).
      15 N-LAST      PIC X(10).
    10 CUSTOMER-ADDRESS.
      15 A-LINE  PIC X(20) OCCURS 2 TIMES.
      15 TOWN        PIC X(10).
      15 ZIP         PIC X(10).
      15 COUNTRY     PIC X(2).
    10 COMPLAINT.
      15 C-TYPE      PIC X(10).
```

1. Right-click input node and select **Test**, or select **Test Message Flow** from Message Broker navigator.
2. On **Events** tab, compose or select test message.
3. Click **Send Message**.

Figure 3-43. Test using the Test Client (1 of 2) WM643 / VM6431.0

Notes:

The next steps for using the Test Client are shown on this and the following figures.

1. Launch the Test Client and configure the deployment location and options on the **Configuration** tab.
2. On the **Events** tab, select the source for a node by clicking Import Source and then browsing to the source file. The test data properties vary depending on the input node.

   If the node is an MQInput or JMSInput, and if the message type was not specified on the input node, select the message by choosing the **Add Message Part** action.

   If the node is a SOAPInput node, the available WSDL operations are shown. Select the required operation from those available. The Test Client populates the message structure which corresponds to the chosen operation.

   If the node is an HTTP input node, the Test Client shows a generic SOAP envelope. To select the specific message that you require, right-click the body element.
3. Click **Send Message**.

Instructor notes:

Purpose — Show the steps for running the test.

Details — When you right-click the **Message Body**, you are presented with a number of options, depending on the type of input node.

You can select the structure of an XML test message from the contents of **Message Sets i**n your workspace and enter values for the elements. These values can be stored and reused in later test runs. If you have a pre-built file containing the test data input (XML or any other format), you can load this by clicking the **Viewer** pull-down, and select **Source**.

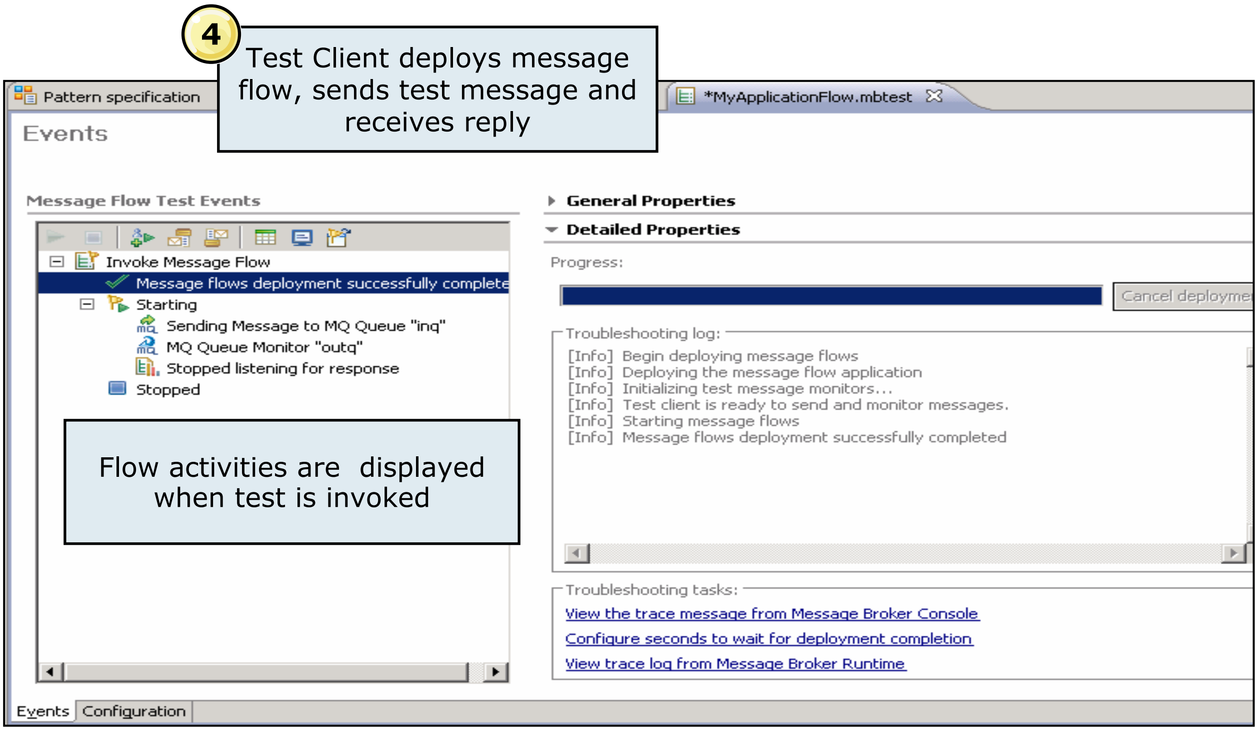
Demonstrate launching the Test Client and selecting the input message

Additional information —

Transition statement — Next: Test using the Test Client (2 of 2)

WebSphere Education

Test using the Test Client (2 of 2)



4 Test Client deploys message flow, sends test message and receives reply

Pattern specification | *MyApplicationFlow.mbtest

Events

Message Flow Test Events

- Invoke Message Flow
  - ✓ Message flows deployment successfully complete
  - Starting
    - Sending Message to MQ Queue "inq"
    - MQ Queue Monitor "outq"
    - Stopped listening for response
  - Stopped

Flow activities are displayed when test is invoked

General Properties

Detailed Properties

Progress:

Troubleshooting log:

```
[Info] Begin deploying message flows
[Info] Deploying the message flow application
[Info] Initializing test message monitors...
[Info] Test client is ready to send and monitor messages.
[Info] Starting message flows
[Info] Message flows deployment successfully completed
```

Troubleshooting tasks:

View the trace message from Message Broker Console

Configure seconds to wait for deployment completion

View trace log from Message Broker Runtime

Figure 3-44. Test using the Test Client (2 of 2) WM643 / VM6431.0

Notes:

The figure shows an example of a flow that has completed successfully. The message flow events are shown on the left side of the Events tab.

4. Click any message flow event to display the event details on the right side of the Events tab.

If the flow completes successfully, it makes a transition to the *Stopped* state and the output message is displayed on the right side

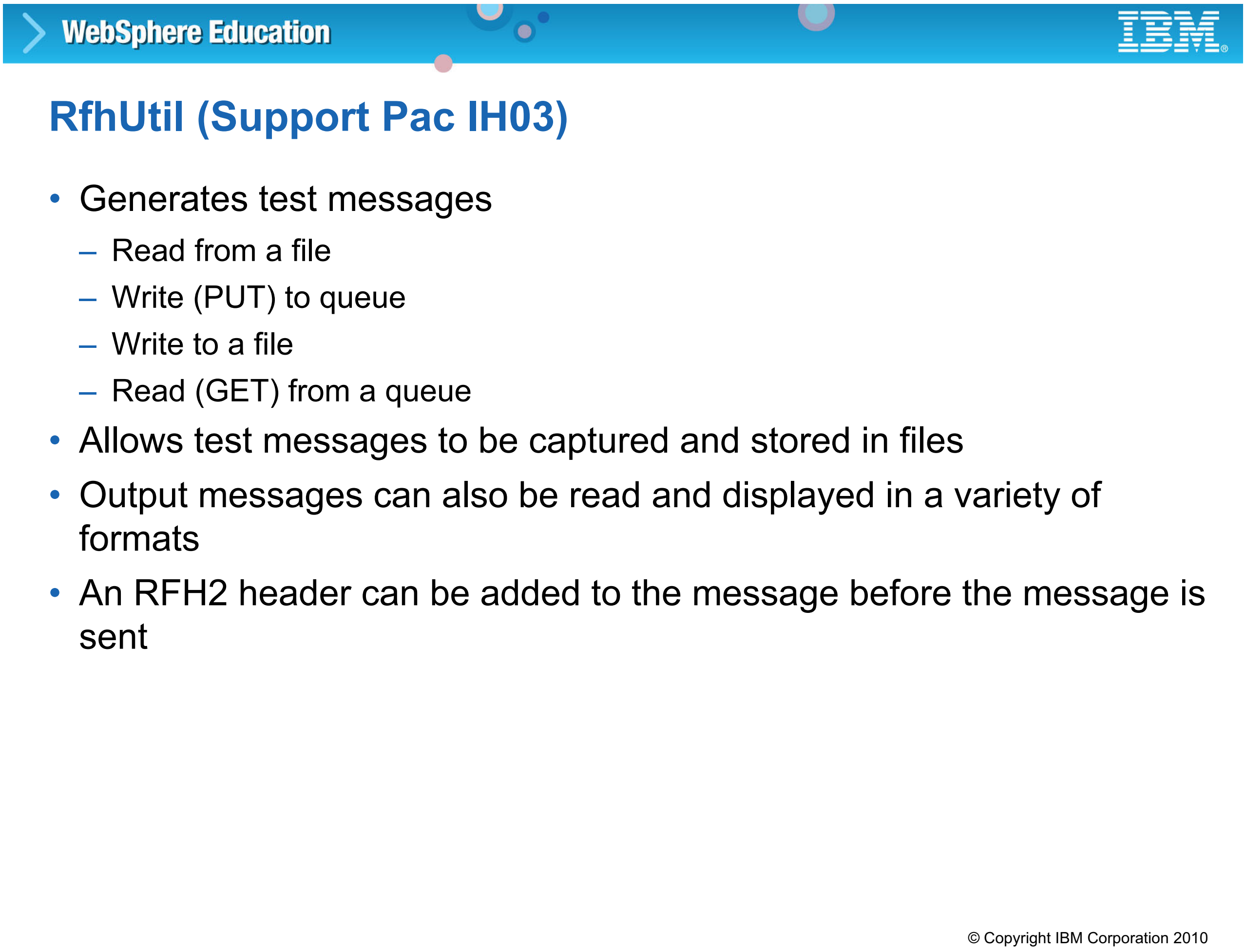
Instructor notes:

Purpose — Show the Events tab with a successful test.

Details —

Additional information —

Transition statement — The message flow can also be tested with a stand-alone test client.



WebSphere Education

RfhUtil (Support Pac IH03)

- Generates test messages
  - Read from a file
  - Write (PUT) to queue
  - Write to a file
  - Read (GET) from a queue
- Allows test messages to be captured and stored in files
- Output messages can also be read and displayed in a variety of formats
- An RFH2 header can be added to the message before the message is sent

Figure 3-45. RfhUtil (Support Pac IH03) WM643 / VM6431.0

Notes:

The RfhUtil program provided with SupportPac IH03 reads data from files, queues or both; writes data to files, queues or both; and displays data in various formats.

The user data portion of the message can be displayed in various formats but cannot be changed. Another program must be used to create or change the user data.

The utility program can add rules and formatting headers (RFH) to messages or files. It writes and formats these headers when found in messages or files it reads. The headers can include publish and subscribe commands.

Instructor notes:

Purpose — Introduce RfhUtil.

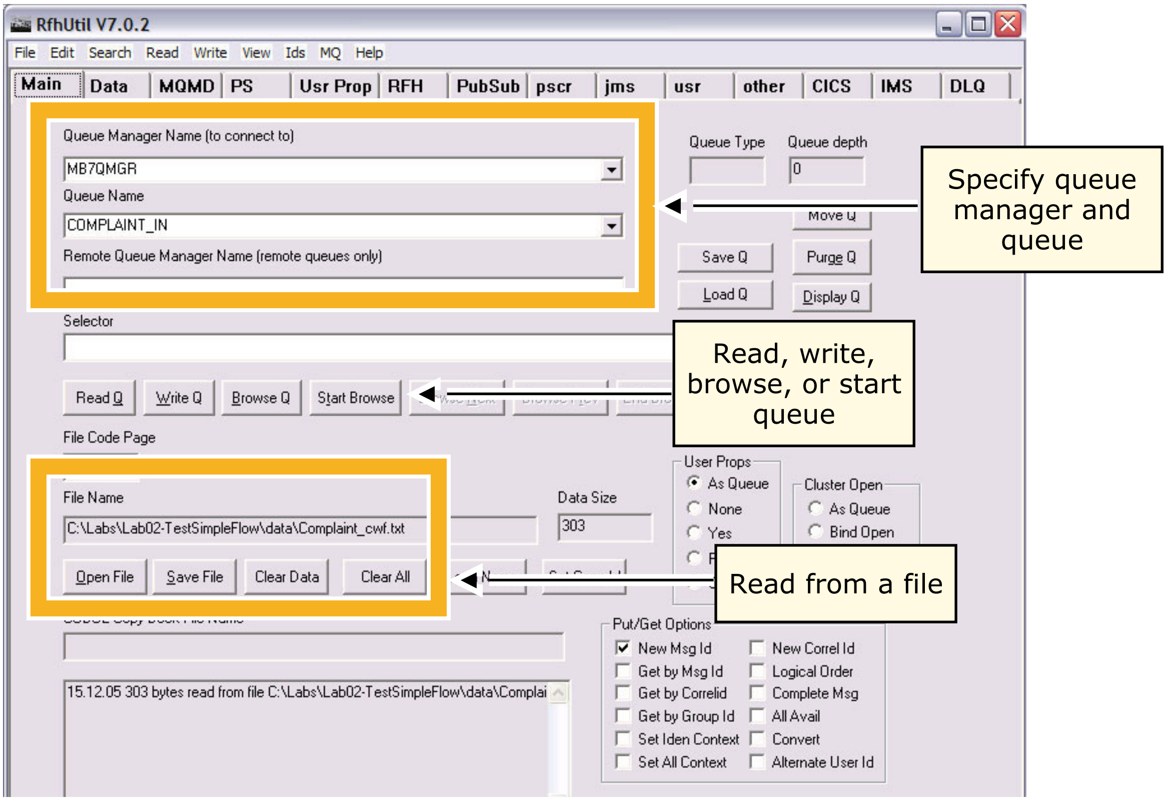
Details — RFHutil is a category 2 SupportPac, that is, free and fully supported. Demonstrate using RfhUtil. Students use this application in the exercise for this unit.

Additional information —

Transition statement — RfhUtil has a number of tabs for configuring test parameters and viewing test results.

WebSphere Education

RfhUtil: Main tab



Queue Manager Name (to connect to): MB7QMGR

Queue Name: COMPLAINT_IN

Remote Queue Manager Name (remote queues only):

Queue depth: 0

File Name: C:\Labs\Lab02-TestSimpleFlow\data\Complaint_cwf.txt

Data Size: 303

User Props: ● As Queue ○ None ○ Yes ○ F

Cluster Open: ○ As Queue ○ Bind Open

Put/Get Options:
☑ New Msg Id
□ Get by Msg Id
□ Get by CorrelId
□ Get by Group Id
□ Set Iden Context
□ Set All Context
□ New Correl Id
□ Logical Order
□ Complete Msg
□ All Avail
□ Convert
□ Alternate User Id

```
15.12.05 303 bytes read from file C:\Labs\Lab02-TestSimpleFlow\data\Complai
```

Specify queue manager and queue

Read, write, browse, or start queue

Read from a file

Figure 3-46. RfhUtil: Main tab WM643 / VM6431.0

Notes:

The **Main** tab in RfhUtil is where you specify the queue manager and queue to get or put messages.

Buttons are provided for the purposes of reading or writing files or messages.

- To read data from a file, click the **Open File** button.
- To write data press the **Save File** button.

To read data from a queue, enter the name of the queue. Enter the name of the local queue manager to connect to if you are not using the default queue manager or if there is no default queue manager.

Instructor notes:

Purpose — Use the RfhUtil program to test message flows.

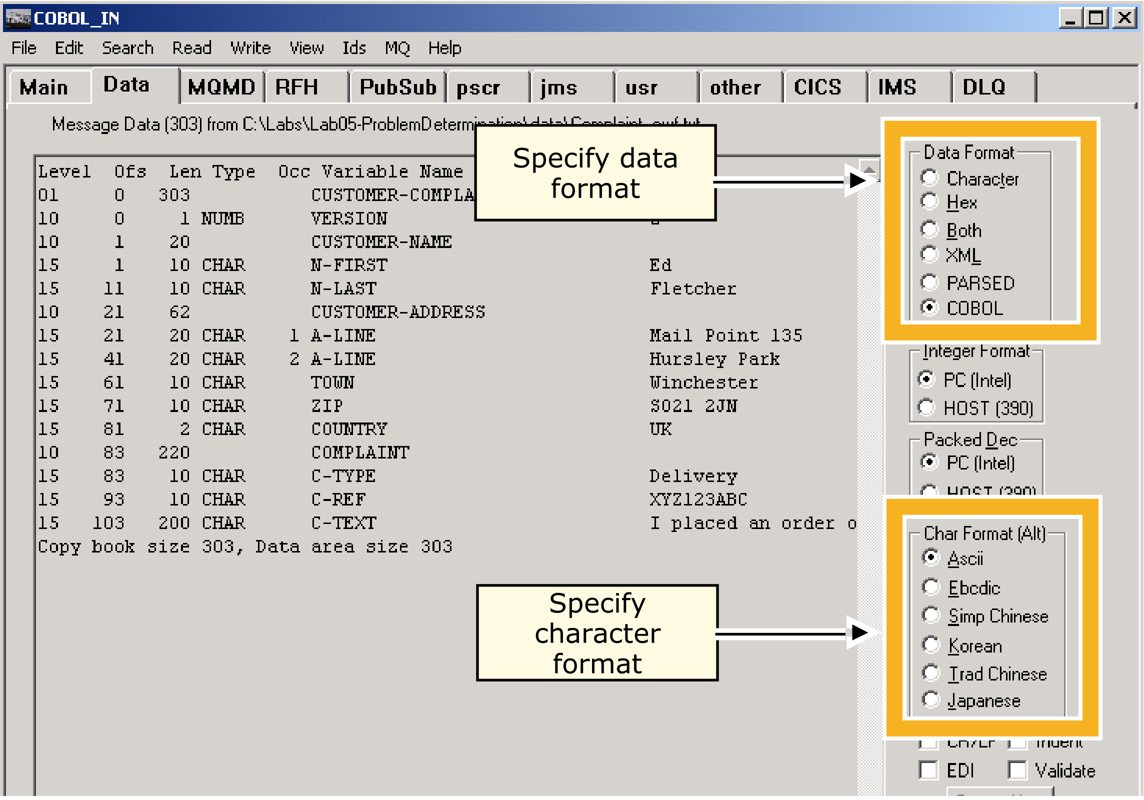
Details —

Additional information —

Transition statement — Next: RfhUtil: Data tab

WebSphere Education

RfhUtil: Data tab



Message Data (303) from C:\Labs\Lab05-ProblemDeterm

```
Level  Ofs  Len Type  Occ Variable Name
01      0   303          CUSTOMER-COMPLA
10      0     1 NUMB     VERSION
10      1    20          CUSTOMER-NAME
15      1    10 CHAR     N-FIRST                    Ed
15     11    10 CHAR     N-LAST                     Fletcher
10     21    62          CUSTOMER-ADDRESS
15     21    20 CHAR    1 A-LINE                    Mail Point 135
15     41    20 CHAR    2 A-LINE                    Hursley Park
15     61    10 CHAR     TOWN                       Winchester
15     71    10 CHAR     ZIP                        S021 2JN
15     81     2 CHAR     COUNTRY                    UK
10     83   220          COMPLAINT
15     83    10 CHAR     C-TYPE                     Delivery
15     93    10 CHAR     C-REF                      XYZ123ABC
15    103   200 CHAR     C-TEXT                     I placed an order o
Copy book size 303, Data area size 303
```

Data Format: ○ Character ○ Hex ○ Both ○ XML ○ PARSED ● COBOL

Integer Format: ● PC (Intel) ○ HOST (390)

Packed Dec: ● PC (Intel) ○ HOST (390)

Char Format (Alt): ● Ascii ○ Ebcdic ○ Simp Chinese ○ Korean ○ Trad Chinese ○ Japanese

□ CR/LF □ Indent

□ EDI □ Validate

Specify data format

Specify character format

© Copyright IBM Corporation 2010

Figure 3-47. RfhUtil: Data tab WM643 / VM6431.0

Notes:

The **Data** tab contains a large data window and a number of radio buttons that control the format that the data displayed in the data buffer. The radio buttons only affect the data display and do not change the contents of the data in the data buffer.

The data in the data buffer can be displayed in six different formats:

- Character
- Hexadecimal
- Both character and hexadecimal
- XML
- Parsed
- COBOL

The character format radio buttons indicate whether the data in the data buffer should be translated from EBCDIC to ASCII before being displayed. It is used with all display formats except hexadecimal. This selection does not affect the data in the data buffer.

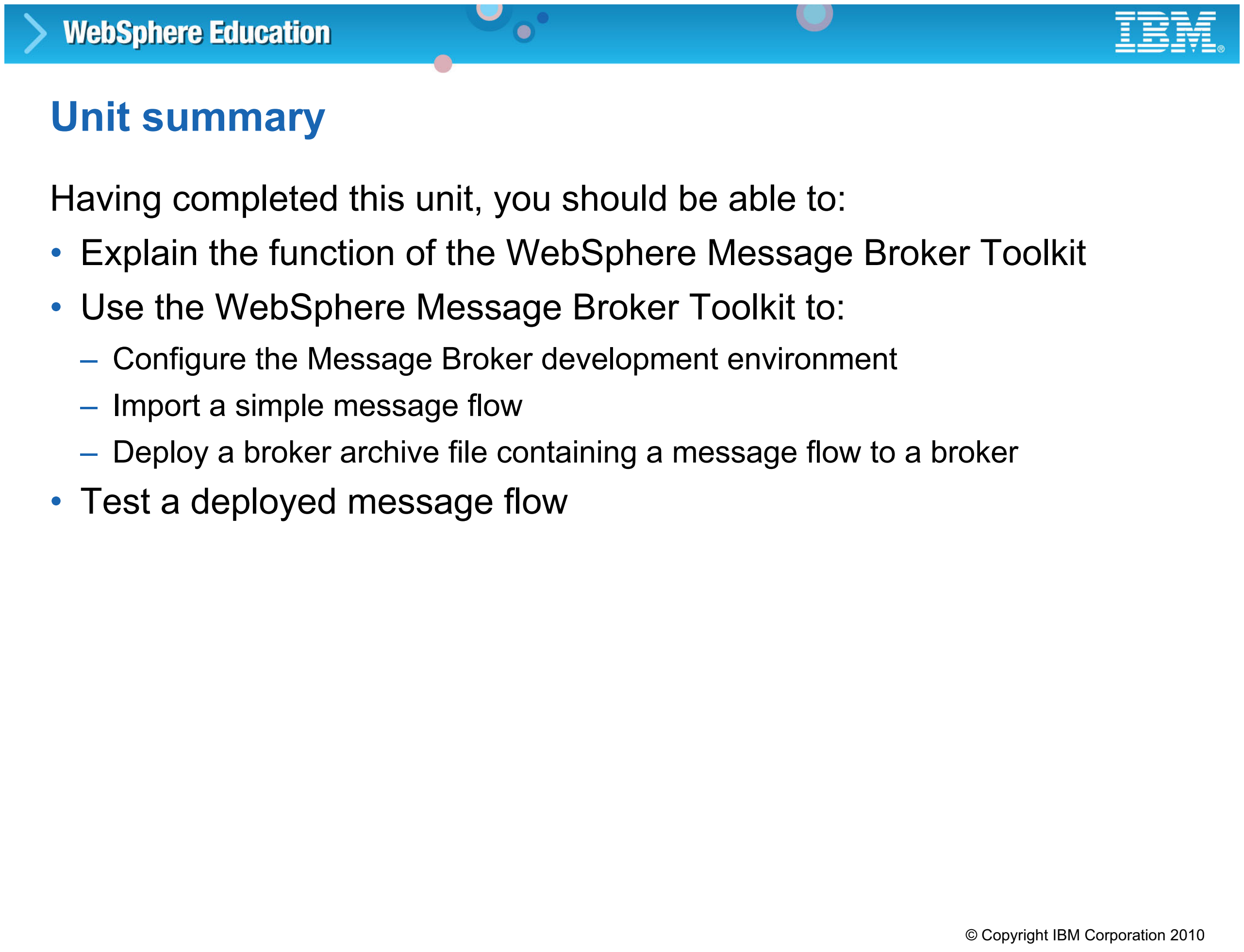
Instructor notes:

Purpose — Show the Data tab and display options.

Details —

Additional information —

Transition statement — Next: Unit summary



WebSphere Education

Unit summary

Having completed this unit, you should be able to:

- Explain the function of the WebSphere Message Broker Toolkit
- Use the WebSphere Message Broker Toolkit to:
  - Configure the Message Broker development environment
  - Import a simple message flow
  - Deploy a broker archive file containing a message flow to a broker
- Test a deployed message flow

© Copyright IBM Corporation 2010

Figure 3-48. Unit summary WM643 / VM6431.0

Notes:

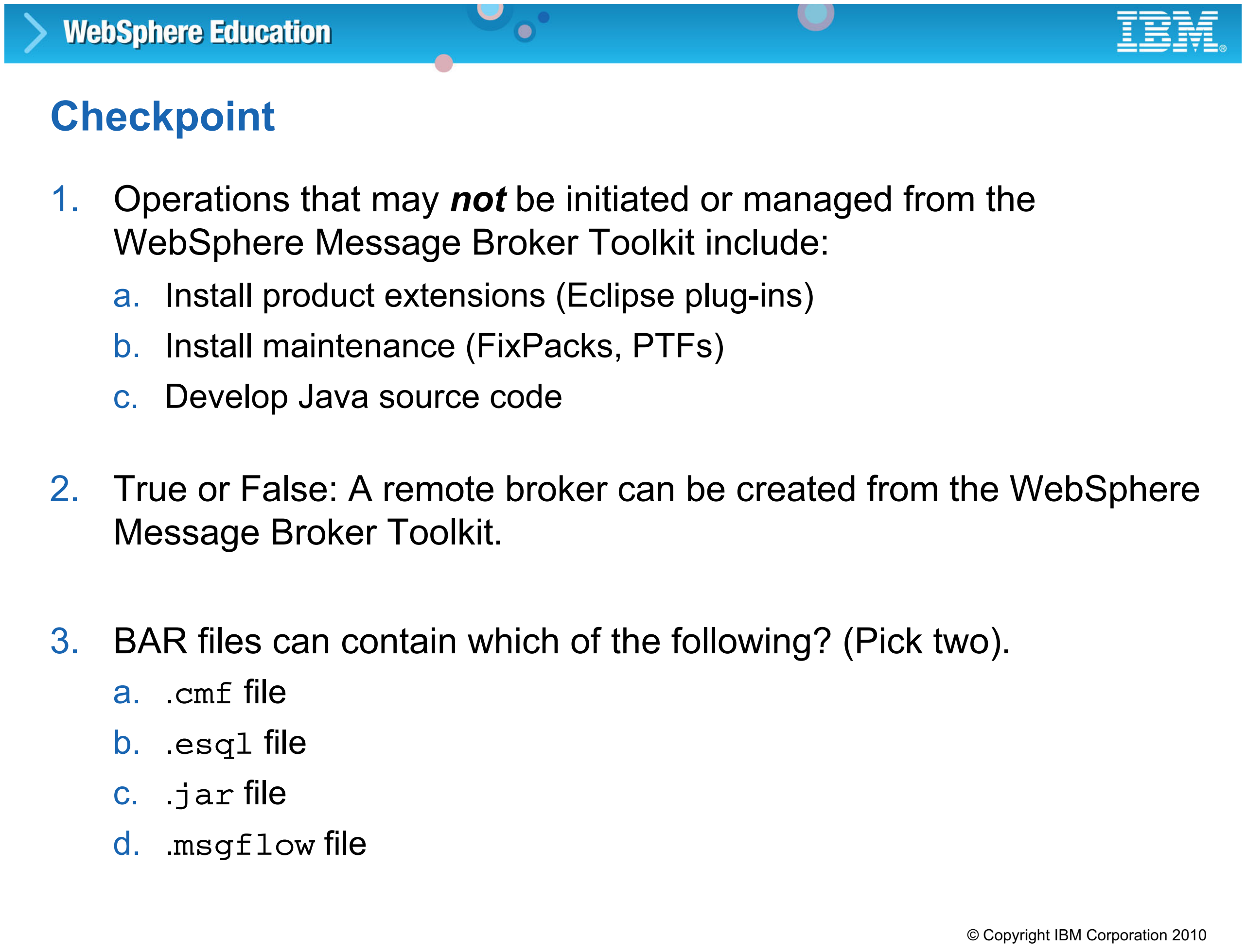
Instructor notes:

Purpose — Unit summary.

Details —

Additional information —

Transition statement — This leads into the next lab exercise, which gives you the opportunity to build and test a message flow.



WebSphere Education

Checkpoint

1. Operations that may ***not*** be initiated or managed from the WebSphere Message Broker Toolkit include:
   a. Install product extensions (Eclipse plug-ins)
   b. Install maintenance (FixPacks, PTFs)
   c. Develop Java source code

2. True or False: A remote broker can be created from the WebSphere Message Broker Toolkit.

3. BAR files can contain which of the following? (Pick two).
   a. `.cmf` file
   b. `.esql` file
   c. `.jar` file
   d. `.msgflow` file

Figure 3-49. Checkpoint WM643 / VM6431.0

Notes:

Write your answers here:

1.

2.

3.

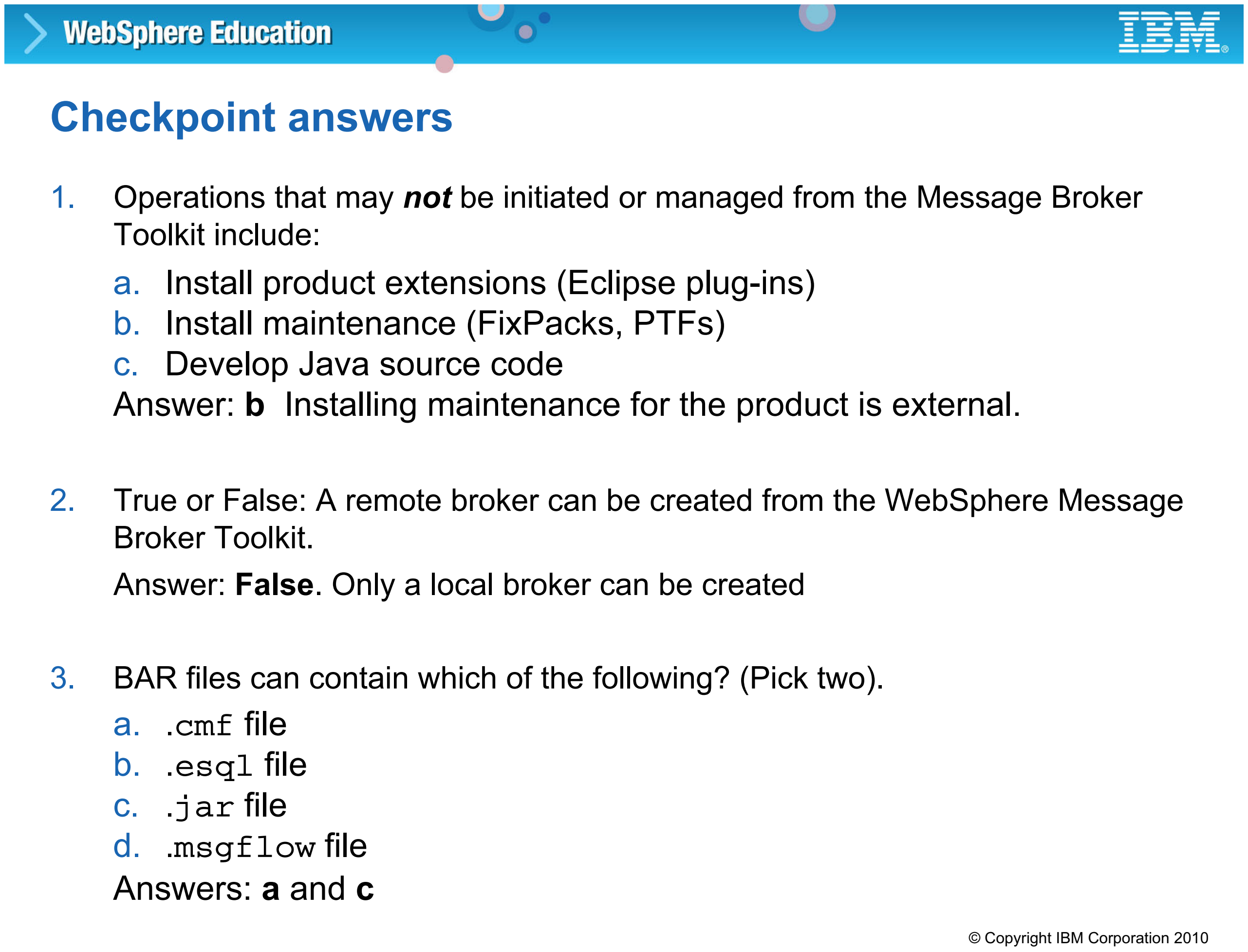
Instructor notes:

Purpose — Checkpoint questions

Details —

Additional information —

Transition statement — And now here are the answers.



WebSphere Education

Checkpoint answers

1. Operations that may ***not*** be initiated or managed from the Message Broker Toolkit include:
   a. Install product extensions (Eclipse plug-ins)
   b. Install maintenance (FixPacks, PTFs)
   c. Develop Java source code
   Answer: **b** Installing maintenance for the product is external.

2. True or False: A remote broker can be created from the WebSphere Message Broker Toolkit.

   Answer: **False**. Only a local broker can be created

3. BAR files can contain which of the following? (Pick two).
   a. `.cmf` file
   b. `.esql` file
   c. `.jar` file
   d. `.msgflow` file
   Answers: **a** and **c**

Figure 3-50. Checkpoint answers

WM643 / VM6431.0

Notes:

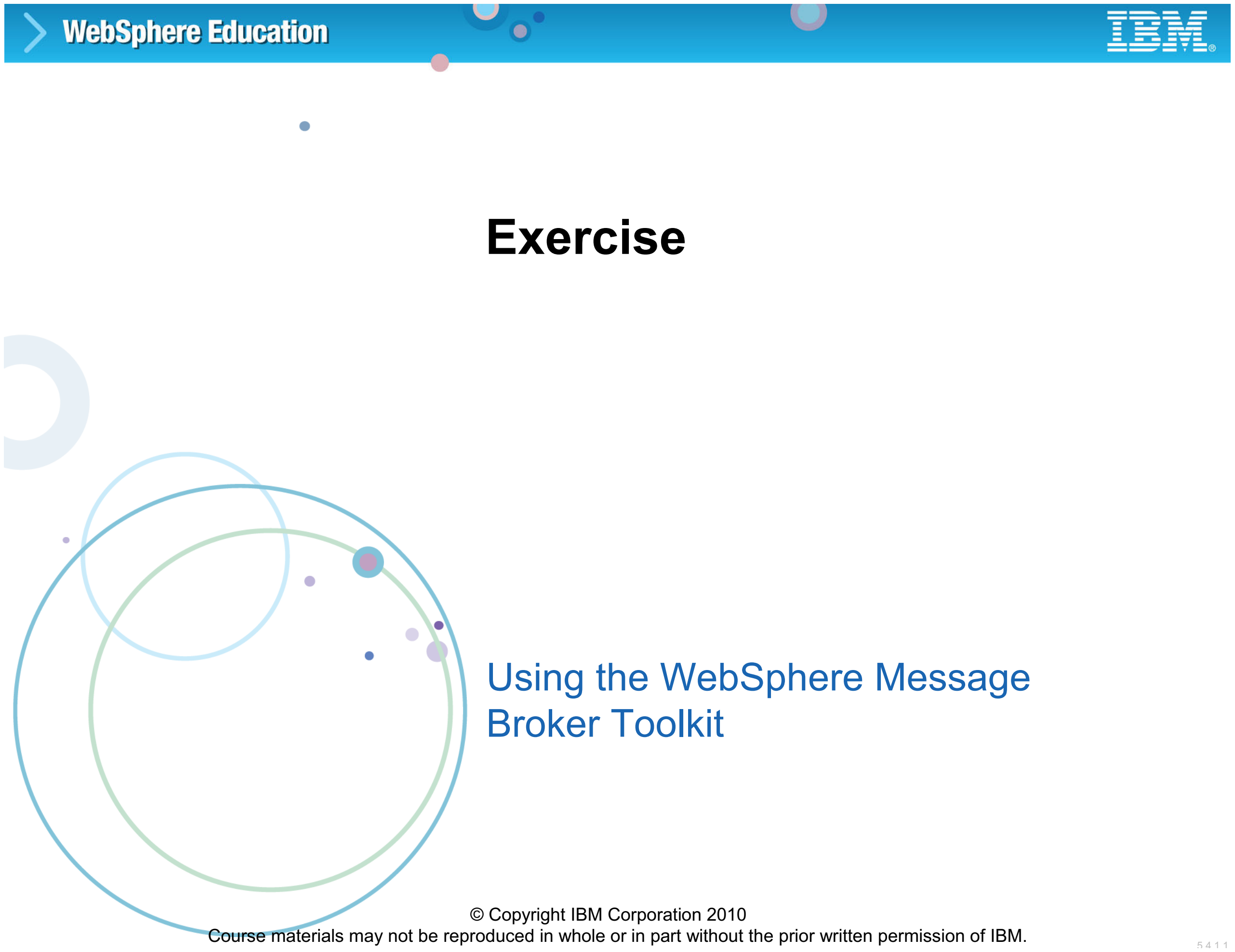
Instructor notes:

Purpose — Checkpoint answers

Details —

Additional information —

Transition statement — Next: Exercise 2: Using the WebSphere Message Broker Toolkit



WebSphere Education

IBM

Exercise

Using the WebSphere Message Broker Toolkit

5411

Figure 3-51. Exercise 2: Using the WebSphere Message Broker Toolkit WM643 / VM6431.0

Notes:

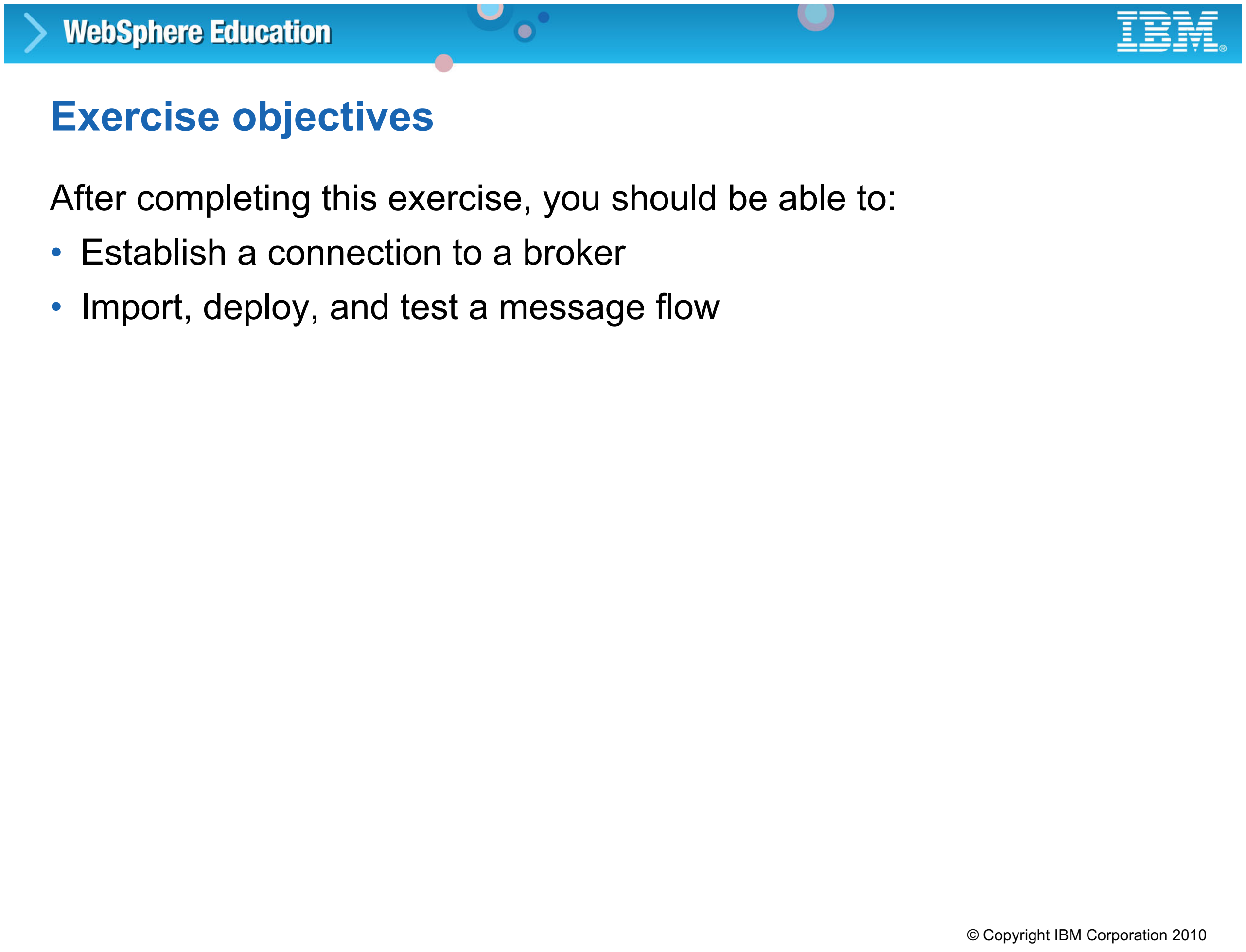
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement — Next: Exercise objectives



WebSphere Education

Exercise objectives

After completing this exercise, you should be able to:

- Establish a connection to a broker
- Import, deploy, and test a message flow

Figure 3-52. Exercise objectives

WM643 / VM6431.0

Notes:

See the Student Exercises Guide for detailed instructions.

Instructor notes:

Purpose — List exercise objectives.

Details — Have students complete Exercise 2.

Additional information —

Transition statement —

Unit 4. Configuring and administering the broker

Estimated time

01:30 lecture, 01:30 exercise

What this unit is about

This unit introduces using commands and WebSphere Message Broker Explorer to configure and administer the broker and broker components. This unit also describes how to back up and restore critical resources.

The Administration API (CMP API) provides an API that can be used to tailor the management and manipulation of the message broker environment. Although not an objective for this unit, it also includes a topic on the Administration API so that you have some information about how to use the Administration API.

What you should be able to do

After completing this unit, you should be able to:

- Perform broker administration using commands
- Perform broker administration using the WebSphere Message Broker Explorer
- Back up and restore critical resources

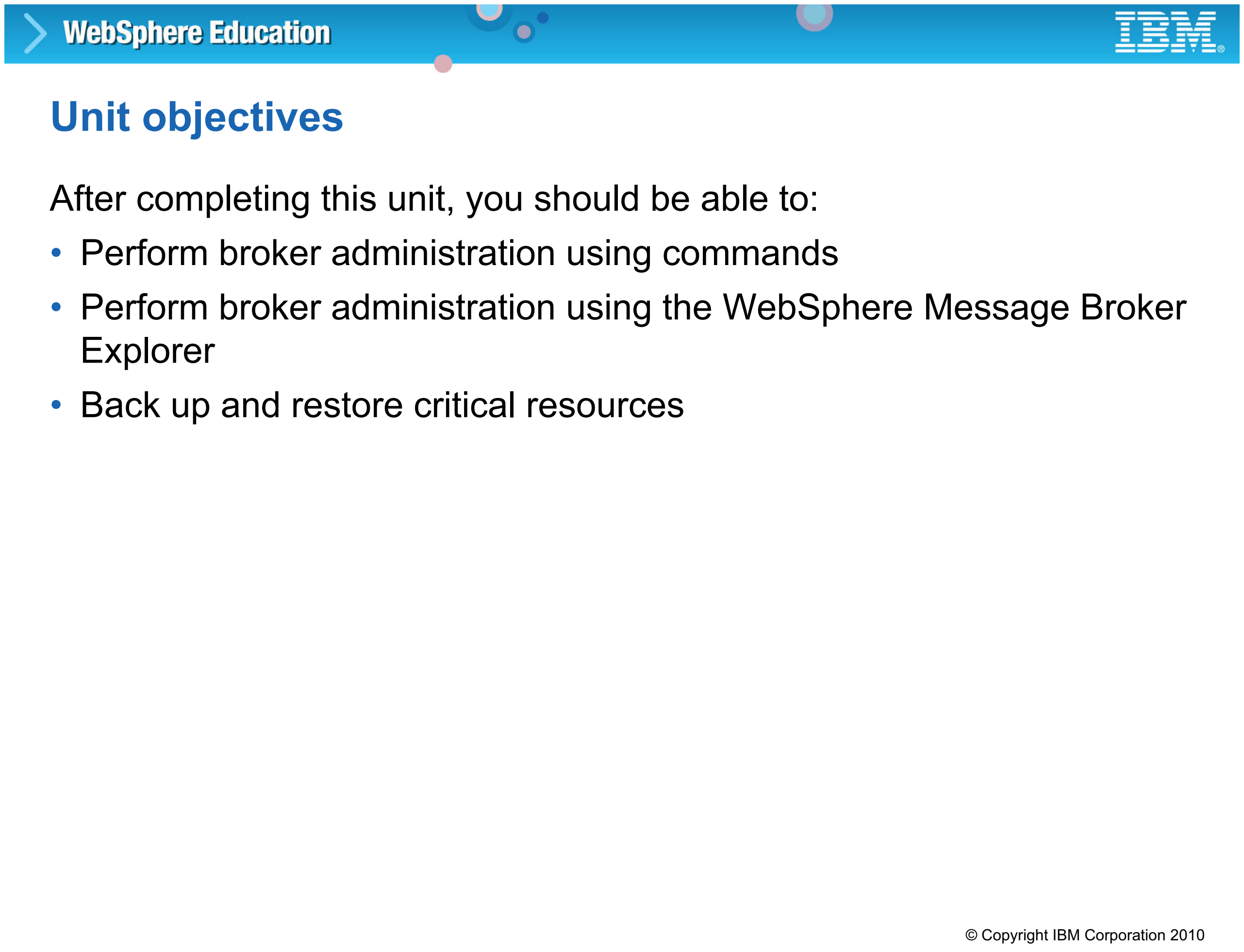
How you will check your progress

Accountability:

- Review checkpoint questions
- Exercise 3, Administering the broker runtime components

References

`www.ibm.com/software/integration/wbimessagebroker/library`
IBM WebSphere Message Broker Library



WebSphere Education

Unit objectives

After completing this unit, you should be able to:

- Perform broker administration using commands
- Perform broker administration using the WebSphere Message Broker Explorer
- Back up and restore critical resources

Figure 4-1. Unit objectives WM643 / VM6431.0

Notes:

The intent of this unit is to demonstrate the broker administration interfaces that do not require the WebSphere Message Broker Toolkit.

- The command console is an environment that permits using commands to operate on nearly all portions of the message broker environment.
- Message Broker Explorer is a stand-alone application that enhances WebSphere MQ Explorer with message broker administration tools.

This unit also includes a topic on backup and restore utilities for broker resources.

Instructor notes:

Purpose — Establish the scope of this unit.

Details — It is important to impress on students that they will not be doing programming with the CMP API. They will make use of a supplied sample. The API will not be covered in detail.

Additional information — None.

Transition statement — First, you will look into command-line administration.

4.1. Administering WebSphere Message Broker using the command-line

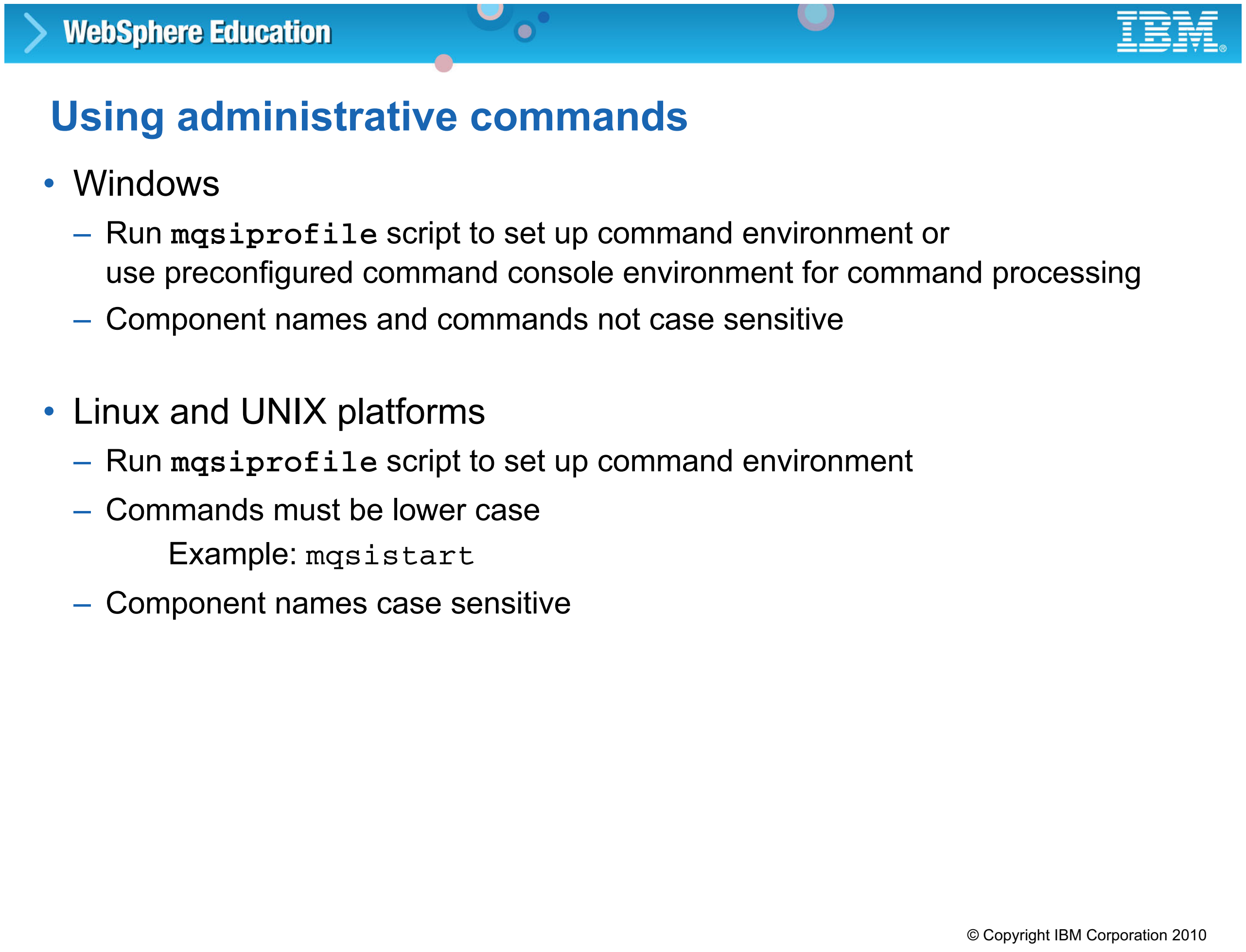
Instructor topic introduction

What students will do — Students will learn about the command-line interface, including many of the commands available for management and manipulation of the broker. They will learn that this is an alternative to using the WebSphere Message Broker Explorer and offers a way to build automated scripts for some functions.

How students will do it — Listen to the lecture and participate in classroom discussion. Exercise 3 gives students the chance to work with some of the more common commands using the command console.

What students will learn — Students will be able to appreciate the use of commands as a means to control the operation of the message broker. They can start to see how these commands can be put into a script for use in management tasks.

How this will help students in their job — Students will be prepared to consider using the command-line interface to automate some of their normal tasks.



WebSphere Education

Using administrative commands

- Windows
  - Run `mqsiprofile` script to set up command environment or use preconfigured command console environment for command processing
  - Component names and commands not case sensitive
- Linux and UNIX platforms
  - Run `mqsiprofile` script to set up command environment
  - Commands must be lower case

    Example: `mqsistart`
  - Component names case sensitive

© Copyright IBM Corporation 2010

Figure 4-2. Using administrative commands WM643 / VM6431.0

Notes:

All WebSphere Message Broker functions can be controlled through the use of commands. These commands begin with “`mqsi`”.

The administrative command line processor is essentially the same for Windows, Linux, and UNIX. In all cases, the command environment (such as file paths and environment variables) must be initialized after product installation, as well as before you run a line command, such as before you start a broker. This initialization function is performed by the `mqsiprofile` command. On Windows platforms, you can start the Command Console, which runs the `mqsiprofile` command in the background, then displays a Windows command shell from which you enter the administration commands.

In the UNIX and Linux environments, you must manually run the `mqsiprofile` command to set up the command environment; you can then run administration commands from the command line.

On all platforms, you can also have user-written scripts run as part of the initialization process.

Before running the `mqsiprofile` command, be sure to change the contents of the `.profile` file by adding this line:

```
/opt/ibm/mqsi/7.0/bin/mqsiprofile
```

Warning

Commands are case-sensitive on the Linux and UNIX platforms.

The `mqsiprofile` command is found in the `bin` directory of the installation directory. Do not change the location of the `mqsiprofile` command, since it can be replaced or updated when you apply product maintenance.

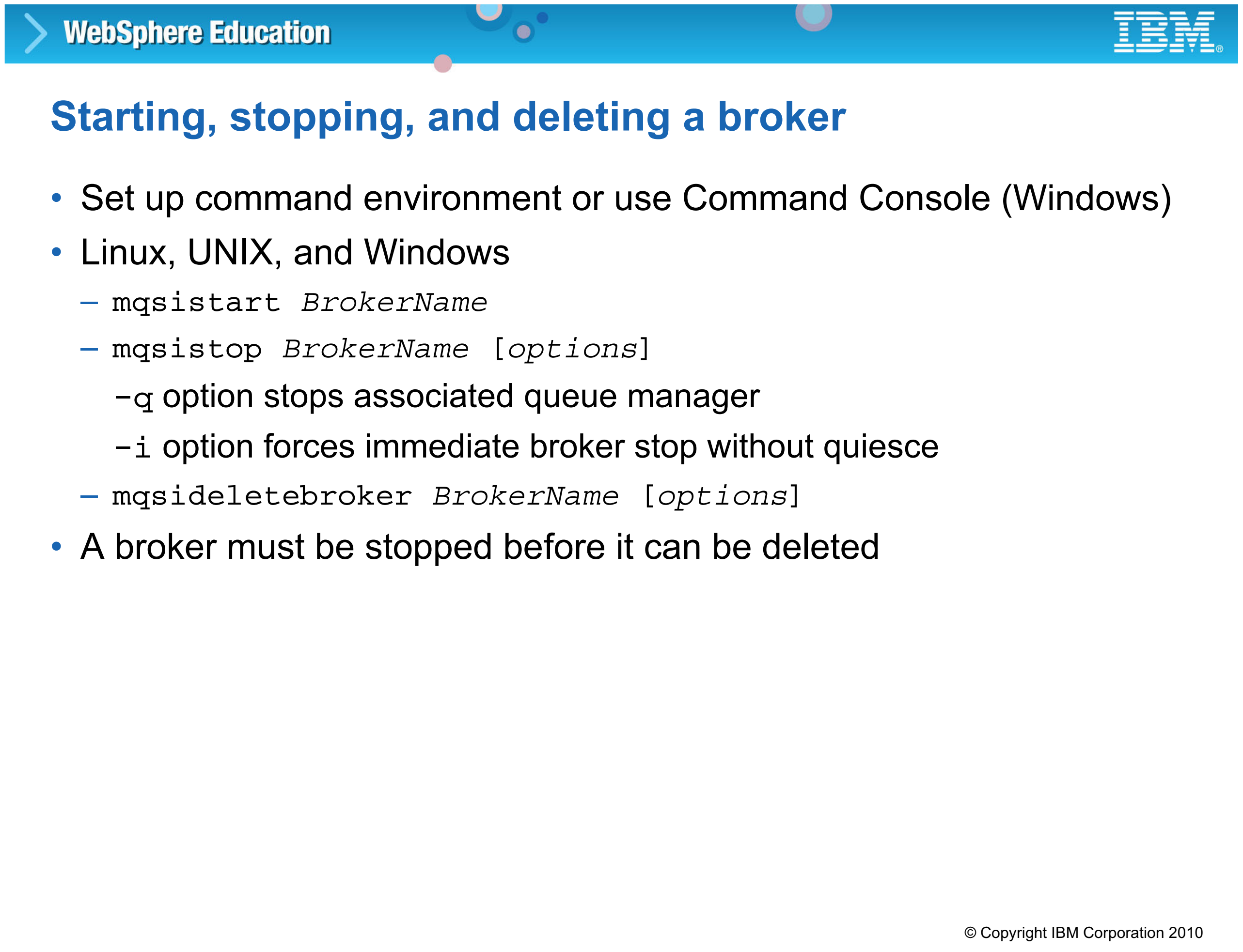
Instructor notes:

Purpose — Introduce the command environment.

Details — It is important to impress on students, especially those who might have used previous versions of WebSphere Message Broker, that there is a requirement to set the environment for processing any command. On Windows, use of the command console simply sets up that environment for you. In any case, whether Windows, UNIX, or Linux, the environment must be initialized properly before entering commands.

Additional information — None.

Transition statement — Examine some of the line commands that you will use as a WebSphere Message Broker administrator.


WebSphere Education
IBM

Starting, stopping, and deleting a broker

- Set up command environment or use Command Console (Windows)
- Linux, UNIX, and Windows
  - `mqsistart` *`BrokerName`*
  - `mqsistop` *`BrokerName`* `[`*`options`*`]`
    - `-q` option stops associated queue manager
    - `-i` option forces immediate broker stop without quiesce
  - `mqsideletebroker` *`BrokerName`* `[`*`options`*`]`
- A broker must be stopped before it can be deleted

Figure 4-3. Starting, stopping, and deleting a broker WM643 / VM6431.0

Notes:

In earlier units, you learned how to create a message broker. After a broker has been created, you must know how to start and stop it. You also must know how to delete it as well.

Be sure that the broker is stopped before attempting to delete it. When deleting a broker, all broker-related queues are removed. On Windows, the broker service is also deleted.

The `mqsitop -q` option stops the queue manager before stopping the broker. The `-i` option immediately stops the broker. Specify this flag only if starting the command without `-i` has failed to stop the broker.

The options provided on the `mqsideletebroker` command allow for stopping, if necessary, and deleting the accompanying queue manager (`-q`) and removing all work path files associated with the broker (`-w`). If the `-q` option is used and other components use the queue manager, the command fails and the queue manager is not deleted.

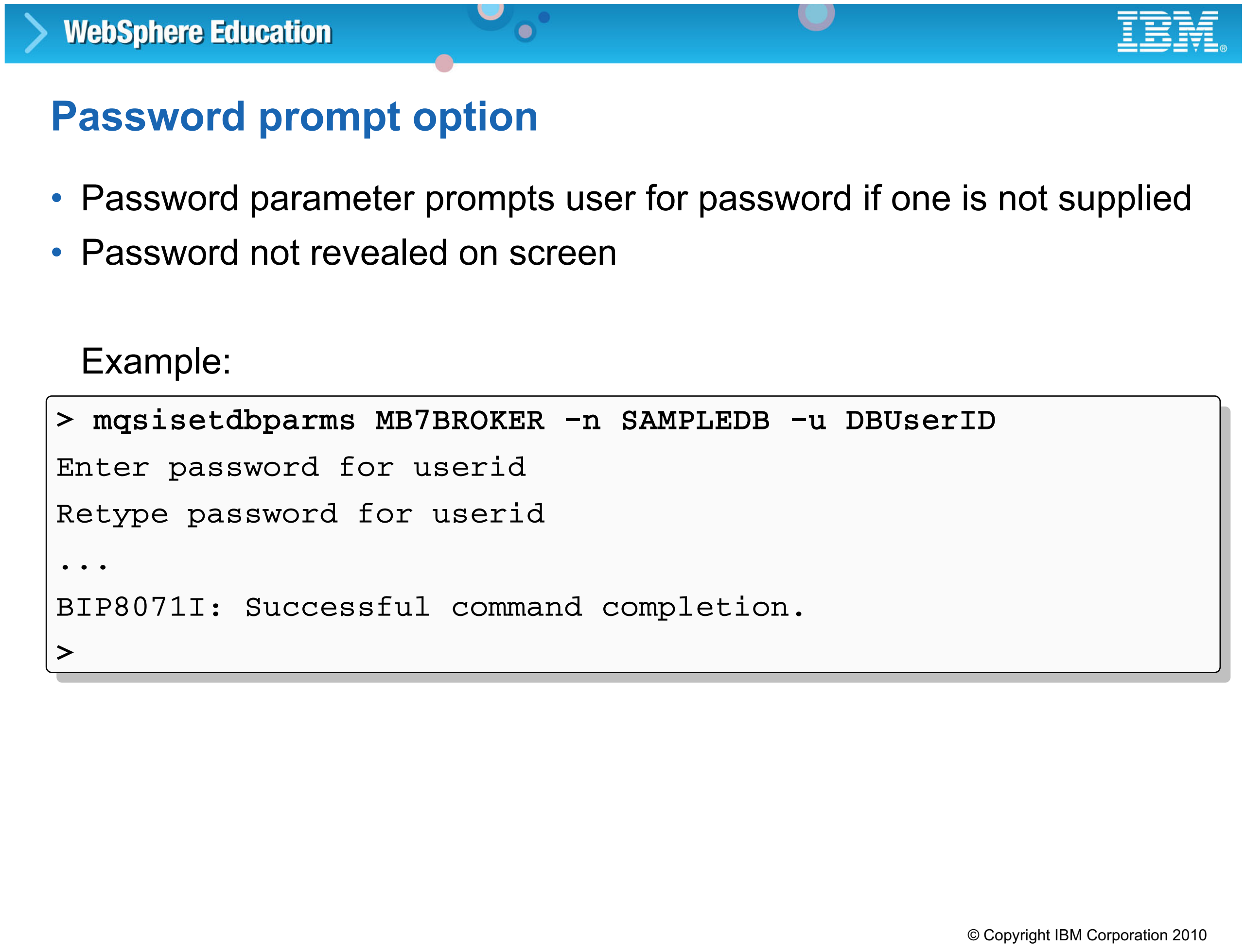
Instructor notes:

Purpose — Discuss starting, stopping, and deleting brokers.

Details — Later, students learn about publish/subscribe. An additional possible action of the `mqsideletebroker` command is in regard to neighboring queue managers in an MQ publish/subscribe network. The command causes a `clrmqbrk` command to be issued at each of the neighboring brokers to notify them to delete the WebSphere Message Broker that is being deleted.

Additional information — None.

Transition statement — When a password is required on a command, it is requested if it is not supplied.



WebSphere Education

Password prompt option

- Password parameter prompts user for password if one is not supplied
- Password not revealed on screen

Example:

```
> mqsisetdbparms MB7BROKER -n SAMPLEDB -u DBUserID
Enter password for userid
Retype password for userid
...
BIP8071I: Successful command completion.
>
```

© Copyright IBM Corporation 2010

Figure 4-4. Password prompt option

WM643 / VM6431.0

Notes:

When a command that requires a password is run without using one, you are prompted for the password.

In the example in the figure, a user ID for the database has been provided (`-u DBUserID`), but a password has not. A prompt for the password is automatically displayed.

The response to the prompt is not displayed on the console when you type it.

Be careful when using commands that require a password in a script since the prompt cannot be responded to interactively.

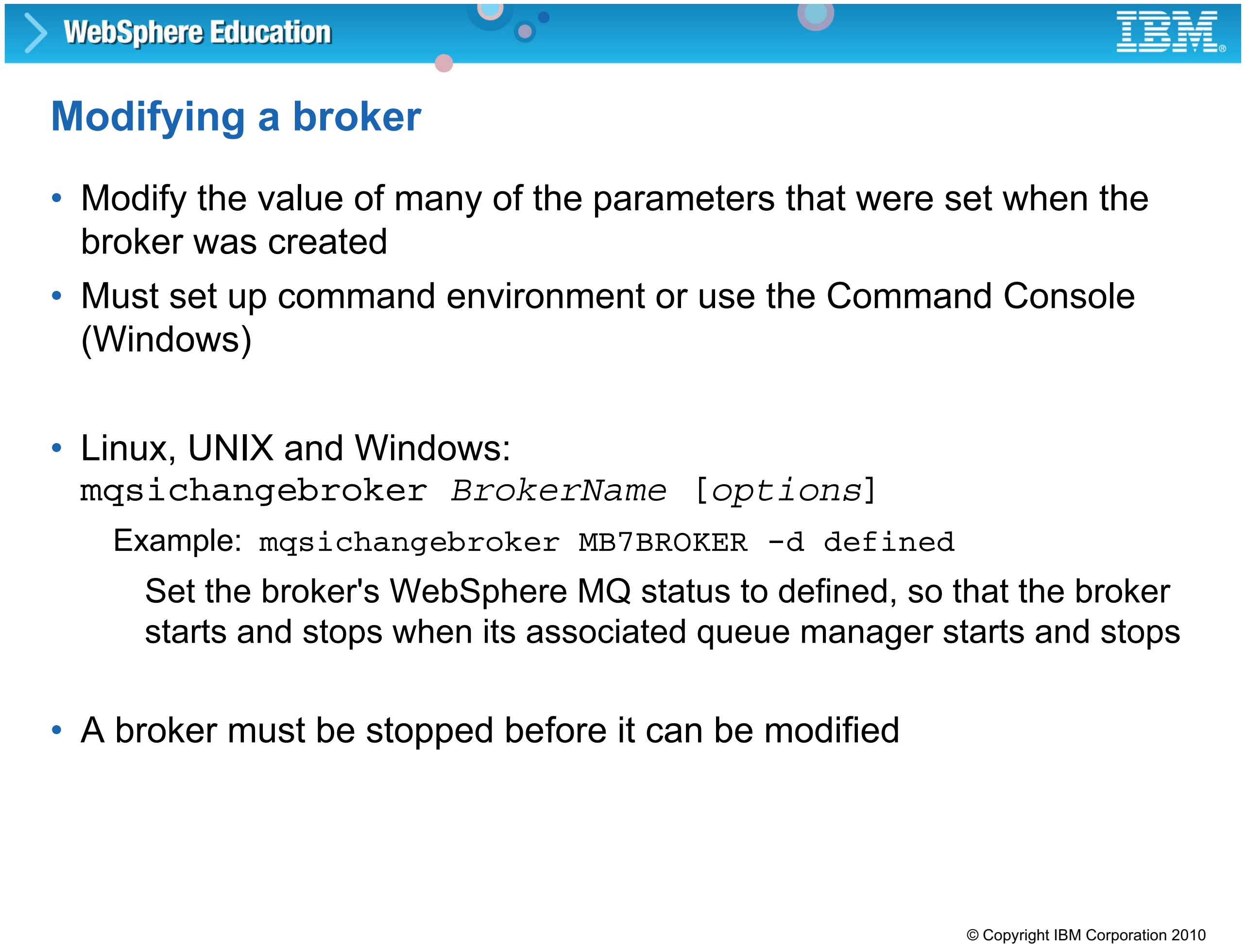
Instructor notes:

Purpose — Mention the password prompt option.

Details — The password prompt option removes the entry of the password as part of the command invocation where it is necessary to show the password.

Additional information — None.

Transition statement — There are some things that can be changed on a broker after it has been created.



WebSphere Education

Modifying a broker

- Modify the value of many of the parameters that were set when the broker was created
- Must set up command environment or use the Command Console (Windows)
- Linux, UNIX and Windows:
  `mqsichangebroker` *`BrokerName`* `[`*`options`*`]`

  Example: `mqsichangebroker MB7BROKER -d defined`

  Set the broker's WebSphere MQ status to defined, so that the broker starts and stops when its associated queue manager starts and stops
- A broker must be stopped before it can be modified

Figure 4-5. Modifying a broker WM643 / VM6431.0

Notes:

Not all broker properties can be changed using the `mqsichangebroker` command.

Here is a partial list of some commonly used options:

`-t` Broker runs as MQ trusted application

`-n` Brokers ceases to run as a WebSphere MQ trusted application

`-g` Maximum time to wait for a response from an execution group.

`-k` Maximum time to wait for a response from a broker

The `-g` and `-k` options are related. The total of the two values is the amount of time the broker allows before sending a negative response. The default values are 300 for -g and 60 seconds for -k.

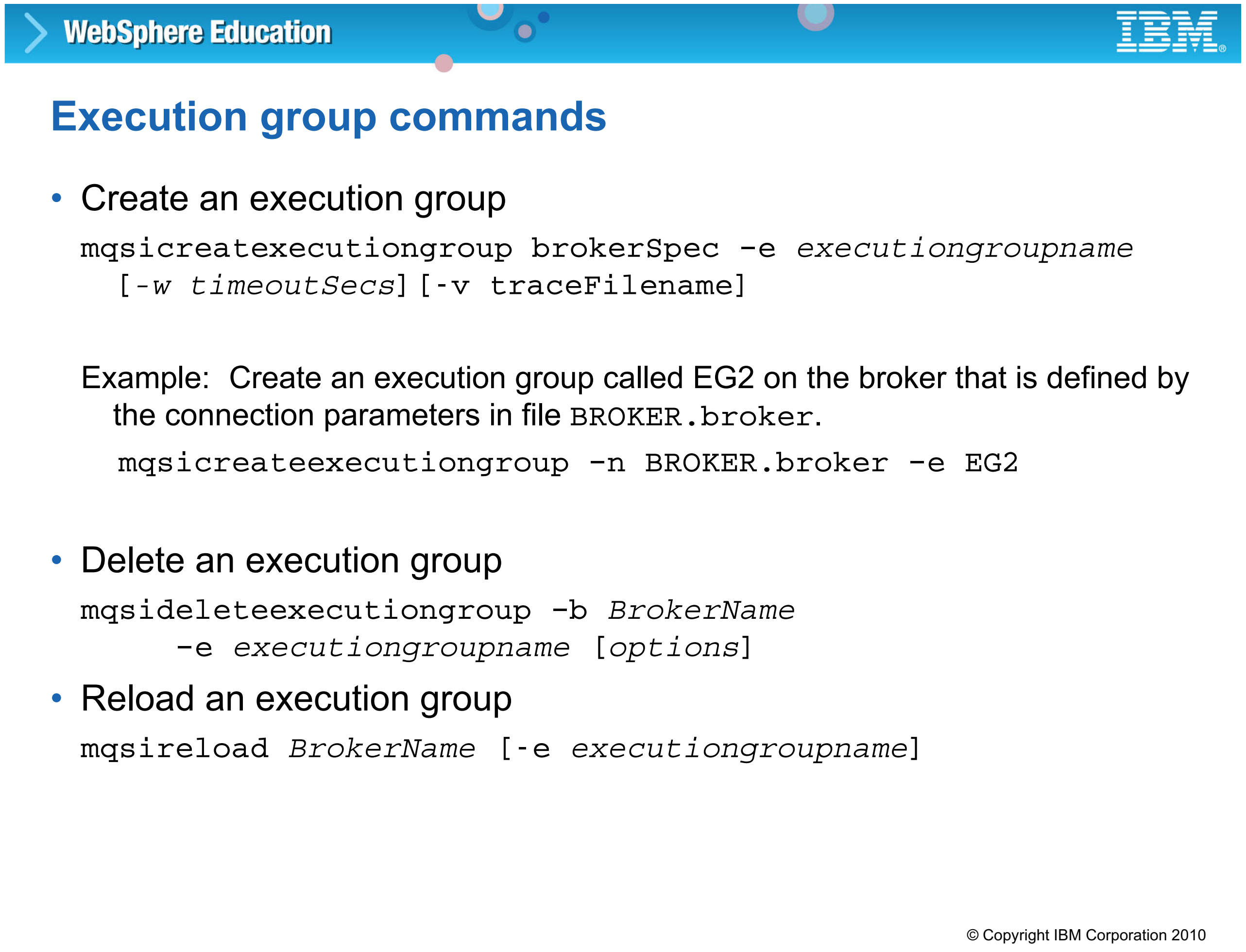
Instructor notes:

Purpose — Provide an overview of some of the options when using the `mqsichangebroker` command.

Details — There are many flags for this command. If students seem interested, take them to the online documentation and review them all, if time permits.

Additional information — None.

Transition statement — Next: Execution group commands



WebSphere Education

Execution group commands

- Create an execution group

```
mqsicreateexecutiongroup brokerSpec -e executiongroupname
  [-w timeoutSecs][-v traceFilename]
```

Example: Create an execution group called EG2 on the broker that is defined by the connection parameters in file `BROKER.broker`.

```
mqsicreateexecutiongroup -n BROKER.broker -e EG2
```

- Delete an execution group

```
mqsideleteexecutiongroup -b BrokerName
     -e executiongroupname [options]
```

- Reload an execution group

```
mqsireload BrokerName [-e executiongroupname]
```

Figure 4-6. Execution group commands WM643 / VM6431.0

Notes:

An alternative to creating an execution group through the WebSphere Message Broker Toolkit is to use the command-line interface. Additionally, the required parameters and optional flags can be specified in a file which is passed to the appropriate command using the `-n` flag.

See the WebSphere Message Broker Information Center for a complete list of command options and syntax.

It is also possible to delete an execution group using the `mqsideleteexecutiongroup` command.

The `mqsireload` command issues a message to the broker to stop and restart all its execution groups or a specific execution group, depending on the command form. You would issue the `mqsireload` command to set the debug port for message flow debugger, for example, or to set content filter for publish/subscribe.

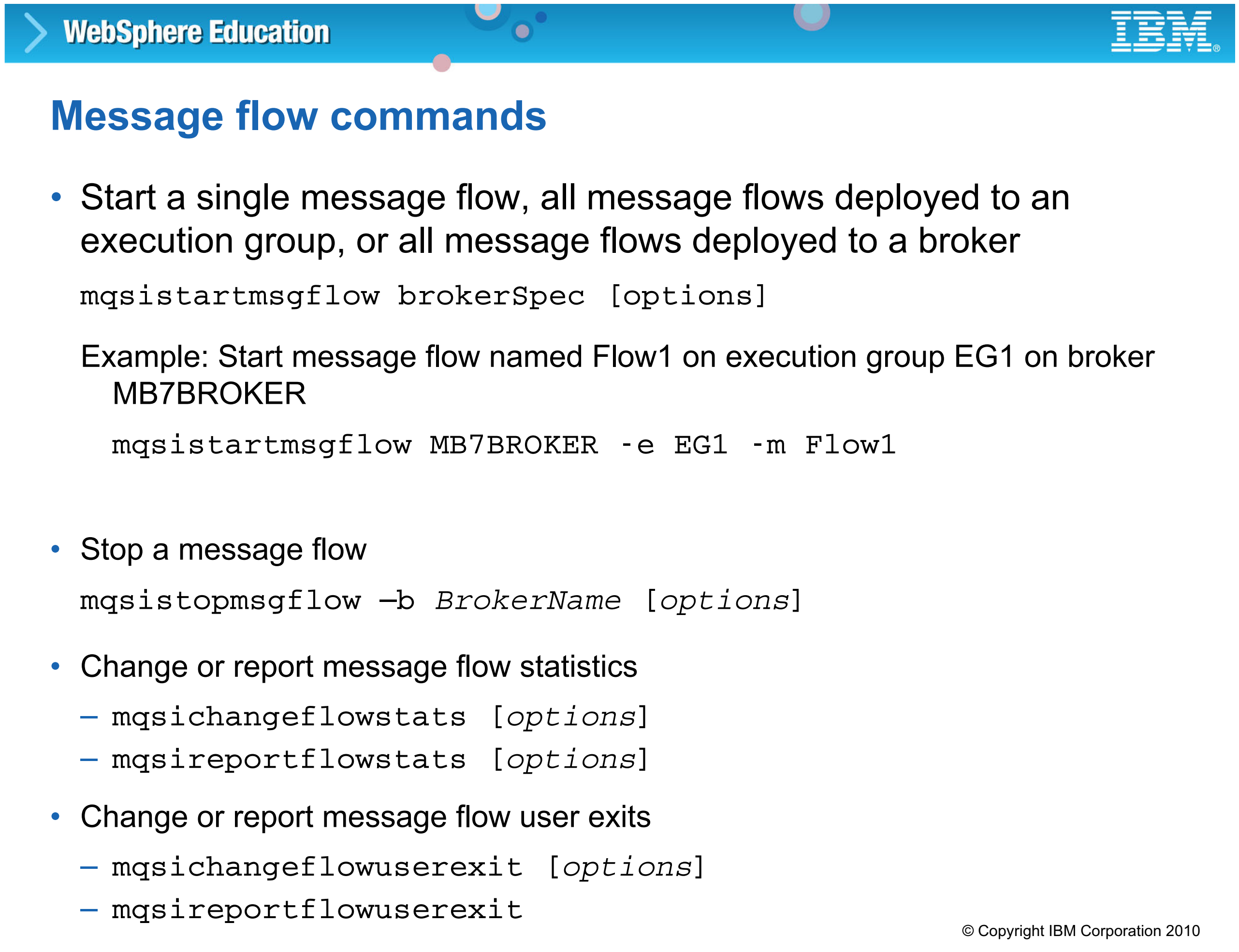
Instructor notes:

Purpose — Offer a description of commands for execution groups.

Details — None.

Additional information —

Transition statement — Next: Message flow commands



WebSphere Education

Message flow commands

- Start a single message flow, all message flows deployed to an execution group, or all message flows deployed to a broker

  ```
  mqsistartmsgflow brokerSpec [options]
  ```

  Example: Start message flow named Flow1 on execution group EG1 on broker MB7BROKER

  ```
  mqsistartmsgflow MB7BROKER -e EG1 -m Flow1
  ```

- Stop a message flow

  ```
  mqsistopmsgflow –b BrokerName [options]
  ```

- Change or report message flow statistics
  – `mqsichangeflowstats [options]`
  – `mqsireportflowstats [options]`
- Change or report message flow user exits
  – `mqsichangeflowuserexit [options]`
  – `mqsireportflowuserexit`

Figure 4-7. Message flow commands WM643 / VM6431.0

Notes:

After a message flow is deployed, it can be manually started and stopped using the `mqsistartmsgflow` and `mqsistopmsgflow` commands. The commands for message flows have a number of options for controlling how the message flows are started and stopped. For example, you can control whether message flows are started in a single execution group by using the `-e` flag. You can use the `-m` flag to specify a specific flow within the execution group (which means that the `-e` flag must also be present).

The use of the `mqsichangeflowstats` and `mqsireportflowstats` are discussed in the accounting and statistics unit later in this course.

A user exit is user-provided custom code, written in C, to track data passing through a message flow. If user exits are implemented, the `mqsichangeflowuserexits` command can be used to set which exits are active and inactive. The `mqsireportflowuserexits` command can be used to report a list of active and inactive exits.

See the Information Center for a complete description of all command and command syntax.

Instructor notes:

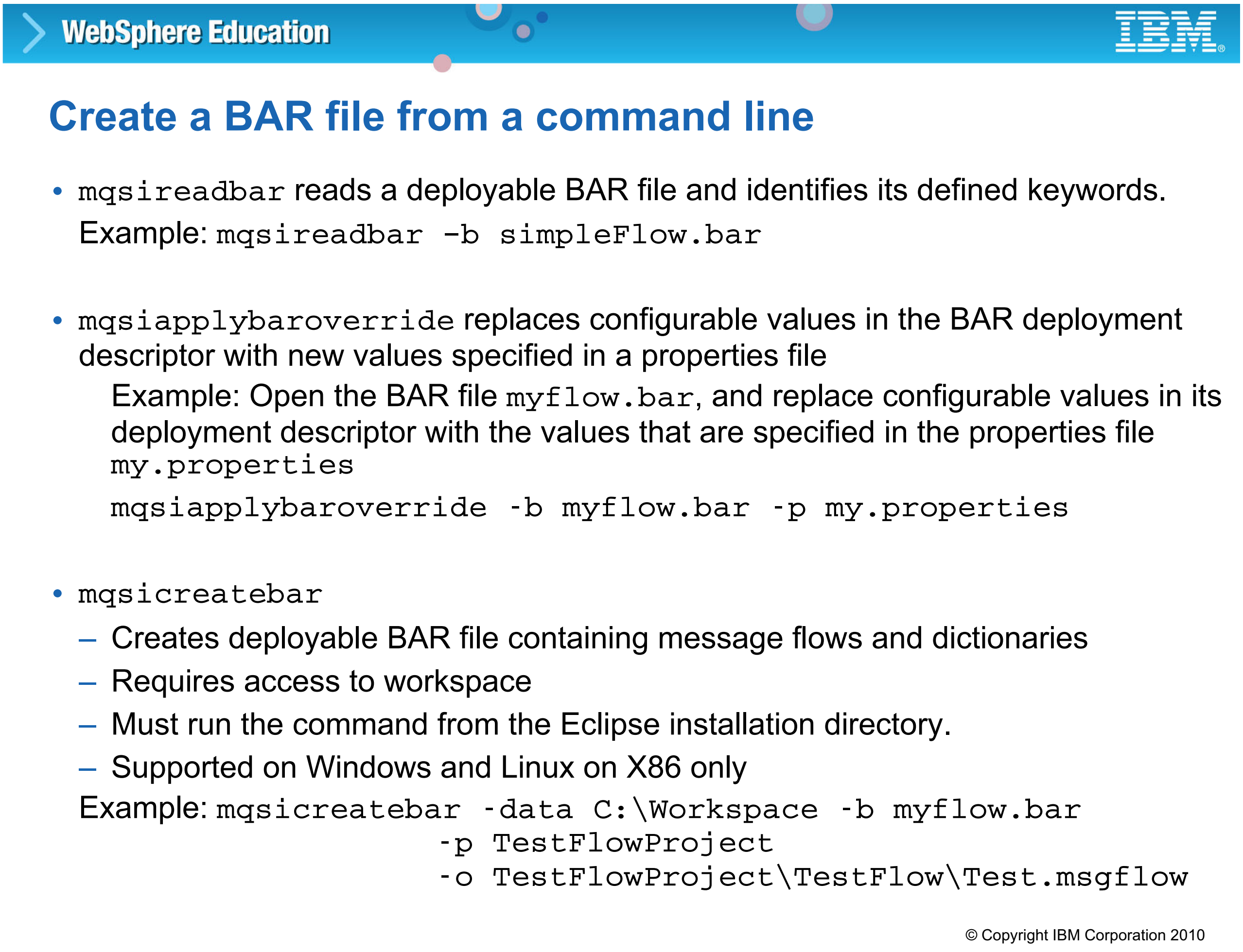
Purpose — Offer an overview of commands for message flows

Details — Show the students how to locate command syntax information and examples in the Information Center.

Additional information — User-provided functions can be started at specific points during the life cycle of a message while it passes through the message flow, and can start utility functions to query information about the point in the flow, and the contents of the message assembly. The user exits can be started when one or more of the following events occurs:

- End of unit-of-work (UOW) or transaction (COMMIT or ROLLBACK)
- A message passing between two nodes
- A message dequeued from the input source

Transition statement — There are some commands that are used for managing broker archive (`.bar`) files.



WebSphere Education

Create a BAR file from a command line

- `mqsireadbar` reads a deployable BAR file and identifies its defined keywords.
  Example: `mqsireadbar -b simpleFlow.bar`

- `mqsiapplybaroverride` replaces configurable values in the BAR deployment descriptor with new values specified in a properties file
  Example: Open the BAR file `myflow.bar`, and replace configurable values in its deployment descriptor with the values that are specified in the properties file `my.properties`
  ```
  mqsiapplybaroverride -b myflow.bar -p my.properties
  ```

- `mqsicreatebar`
  - Creates deployable BAR file containing message flows and dictionaries
  - Requires access to workspace
  - Must run the command from the Eclipse installation directory.
  - Supported on Windows and Linux on X86 only

  Example:
  ```
  mqsicreatebar -data C:\Workspace -b myflow.bar
                -p TestFlowProject
                -o TestFlowProject\TestFlow\Test.msgflow
  ```

Figure 4-8. Create a BAR file from a command line WM643 / VM6431.0

Notes:

The `mqsicreatebar` command provides a command-line compiler that creates deployable broker archive files containing message flows and dictionaries. If you use a repository to store your message flows and dictionaries, you can write scripts or batch files to deploy the message flow applications using the `mqsicreatebar` command and the repository's command-line tools.

Important

The fully qualified path is required when specifying the BAR file name in the commands.

The `mqsireadbar` command lists the keywords defined for each deployable file within a deployable broker archive file. For example:

```
C:\test.bar
     BAR Entry: simpleflow.cmf (07/10/07 10:43:44)
          Author = Matt
          VERSION = v1.1
          Information = This flow simply removes messages from IN.Q
```

It also displays deployment descriptors and the list of any properties in the BAR file that can be overridden, together with the new values of any overrides that are currently applied. For example:

```
Deployment descriptor:
    simpleflow#additionalInstances
    simpleflow#commitCount
    simpleflow#commitInterval
    simpleflow#coordinatedTransaction
    simpleflow#MQInput.topicProperty
    simpleflow#MQInput.validateMaster
    simpleflow#MQInput.queueName = OVERRIDDEN.Q
    simpleflow#MQInput.serializationToken
```

You can use the `mqsiapplybaroverride` command to replace configurable values in the broker archive deployment descriptor, `META-INF/broker.xml` with new values that you specify in a properties file. Configurable properties allow an administrator to update target-dependent properties, such as queue names, queue manager names, and database connections. By changing configurable properties, you can customize a BAR file for a new environment, for example a test system, without needing to edit and rebuild the message flows, message mappings, or ESQL transformation program.

Follow these steps to edit properties using the `mqsiapplybaroverride` command:

1. Open a command window that is configured for your environment.
2. Create a text file (with a `.properties` file extension).
3. Enter the command, typed on a single line, specifying the location of your broker archive deployment descriptor (typically `broker.xml`) and the file that contains the properties to be changed. A file with a `.bar` extension is created.

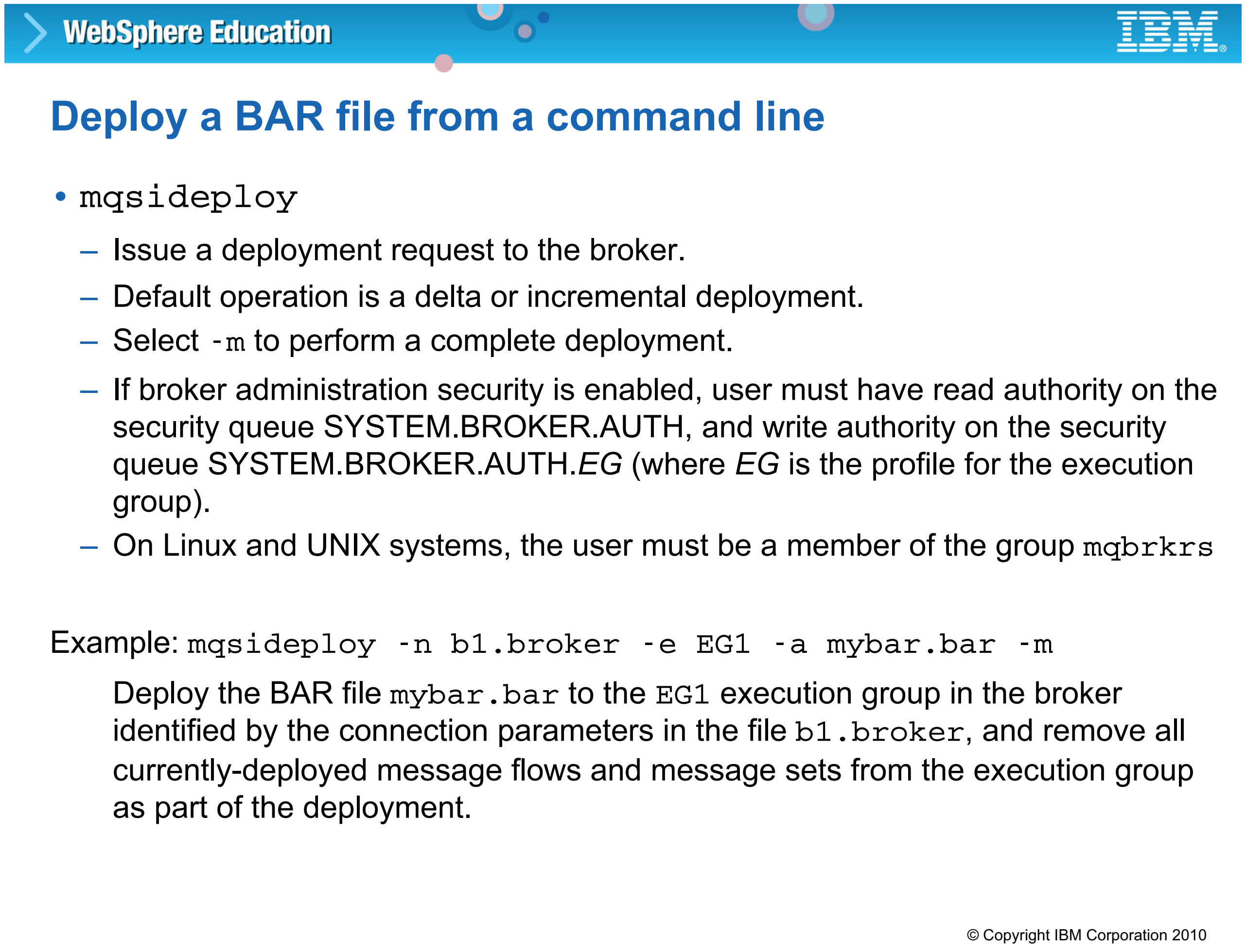
Instructor notes:

Purpose — Provide information on commands pertaining to broker archive files that may be of use to a Message Broker administrator.

Details — The commands listed allow the creation, viewing or listing the contents of, and the modification of `.bar` file parameters. The `mqsiapplybaroverride` command is a command that most administrators are apt to use frequently.

Additional information — None.

Transition statement — Now you will look at the `mqsideploy` command.



WebSphere Education

Deploy a BAR file from a command line

- `mqsideploy`
  - Issue a deployment request to the broker.
  - Default operation is a delta or incremental deployment.
  - Select `-m` to perform a complete deployment.
  - If broker administration security is enabled, user must have read authority on the security queue SYSTEM.BROKER.AUTH, and write authority on the security queue SYSTEM.BROKER.AUTH.*EG* (where *EG* is the profile for the execution group).
  - On Linux and UNIX systems, the user must be a member of the group `mqbrkrs`

Example: `mqsideploy -n b1.broker -e EG1 -a mybar.bar -m`

Deploy the BAR file `mybar.bar` to the `EG1` execution group in the broker identified by the connection parameters in the file `b1.broker`, and remove all currently-deployed message flows and message sets from the execution group as part of the deployment.

Figure 4-9. Deploy a BAR file from a command line WM643 / VM6431.0

Notes:

The `mqsideploy` command is used to send a deployment request. Typically this command is used to deploy a broker archive file to an execution group. The `mqsideploy` command is used frequently in scripts.

It is also possible to use the `mqsideploy` command to remove objects from an execution group. Use the `-d` flag followed by one of more objects, separated by a colon "`:`".

For example,

```
mqsideploy -n MB7BROKER.broker -e EG1 -d Admin1.cmf:Admin2.cmf
```

removes the two message flows called `Admin1` and `Admin2` from execution group `EG1`.

Instructor notes:

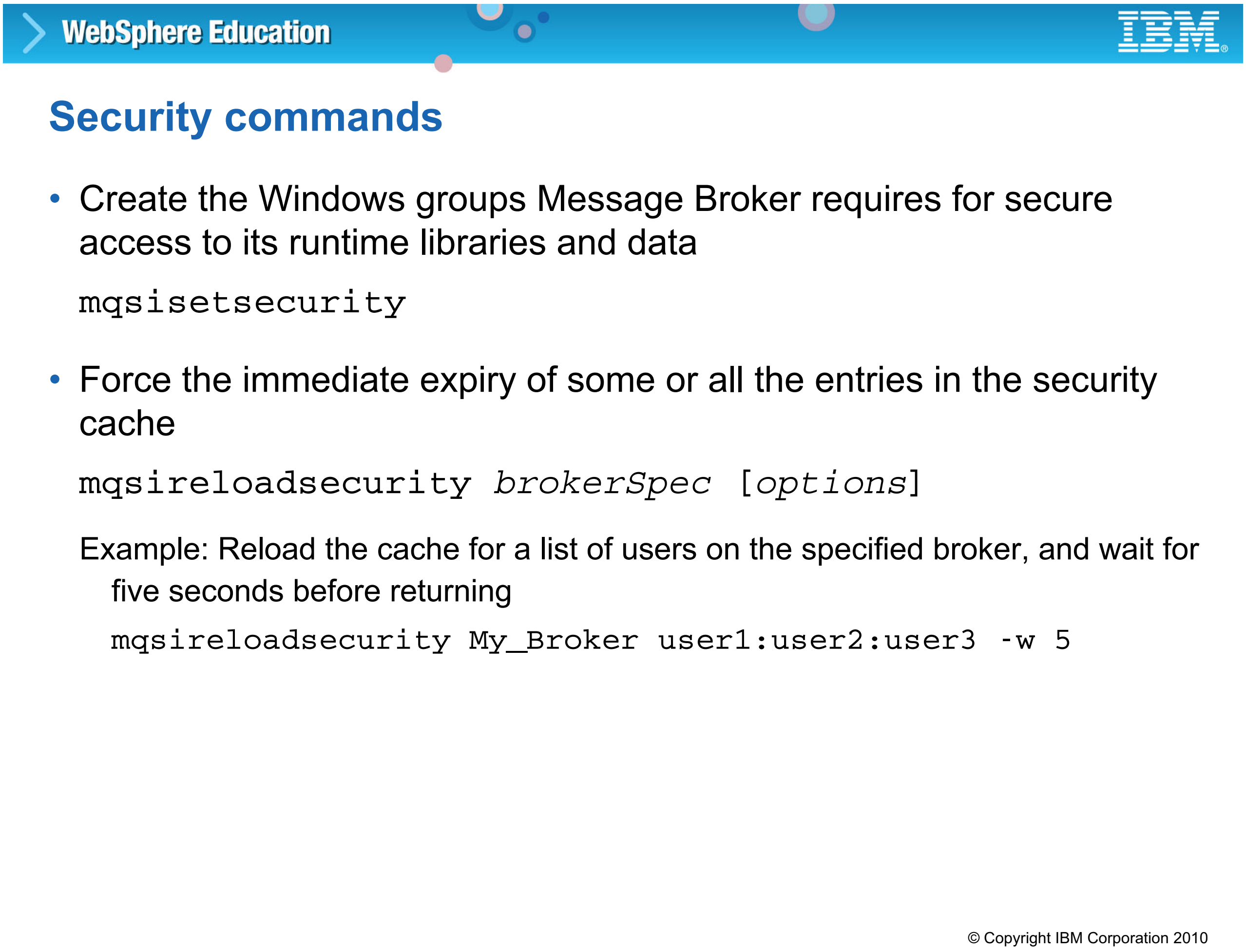
Purpose — Discuss the `mqsideploy` command in more detail.

Details — There are many possibilities with the `mqsideploy` command. Students should understand that the `mqsideploy` command has many uses. It is important that you constantly direct to the Information Center as it is not the intent to duplicate the online documentation in the student notes.

They should be investigating it if they think they want to handle such tasks using scripts or batch files.

Additional information — The option to cancel deployment (`-c`) has been removed in Message Broker V7.0.

Transition statement — There are also commands for implementing security.



WebSphere Education
IBM

Security commands

- Create the Windows groups Message Broker requires for secure access to its runtime libraries and data

  `mqsisetsecurity`

- Force the immediate expiry of some or all the entries in the security cache

  `mqsireloadsecurity` *`brokerSpec`* `[`*`options`*`]`

  Example: Reload the cache for a list of users on the specified broker, and wait for five seconds before returning

  ```
  mqsireloadsecurity My_Broker user1:user2:user3 -w 5
  ```

Figure 4-10. Security commands WM643 / VM6431.0

Notes:

The `mqsisetsecurity` command applies to Windows environments. During installation of WebSphere Message Broker, this command is automatically run. However, it is possible that WebSphere MQ might be installed *after* the WebSphere Message Broker code. If so, you can issue this command to add your account to the `mqm` group provided you have **Administrator** authority.

Use the `mqsireloadsecurity` command to force the immediate expiration of some or all of the entries in the security cache. Entries in the security cache are valid for a specified length of time, after which the entries are marked as “expired”. When an entry is marked as expired, it must be reauthenticated with the security provider before it can be reused and its expiration time reset.

Security is described in more detail later in this course.

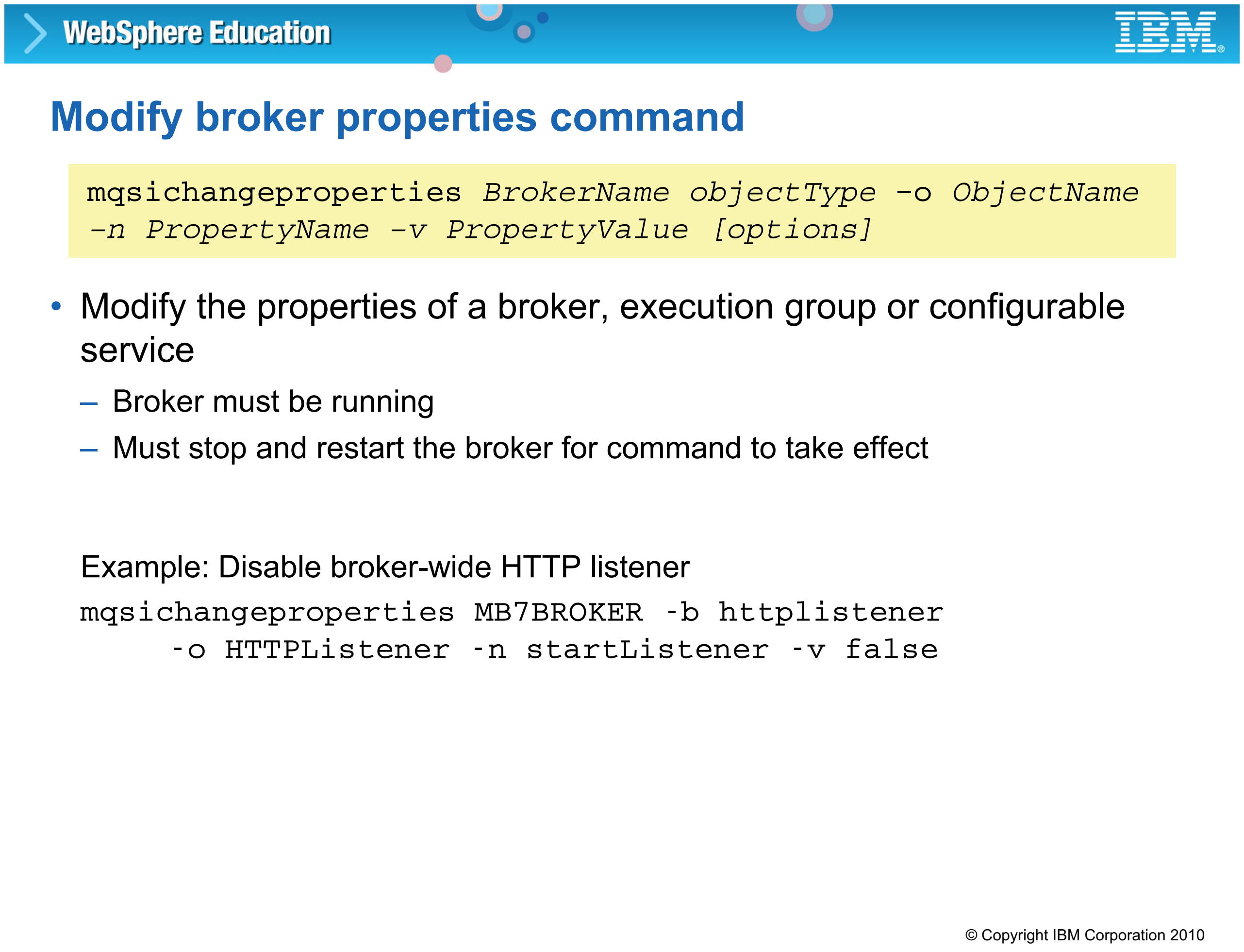
Instructor notes:

Purpose — Review the commands that apply to security.

Details — The `mqsisetsecurity` command is not frequently used but is included for completeness.

Additional information — None.

Transition statement — Next is the `mqsichangeproperties` command.



WebSphere Education
IBM

Modify broker properties command

```
mqsichangeproperties BrokerName objectType -o ObjectName
-n PropertyName -v PropertyValue [options]
```

- Modify the properties of a broker, execution group or configurable service
  – Broker must be running
  – Must stop and restart the broker for command to take effect

Example: Disable broker-wide HTTP listener

```
mqsichangeproperties MB7BROKER -b httplistener
     -o HTTPListener -n startListener -v false
```

Figure 4-11. Modify broker properties command WM643 / VM6431.0

Notes:

Configurable services are typically runtime properties related to external services on which the broker relies.

You can use the `mqsichangeproperties` command to modify broker properties and configurable properties of broker resources.

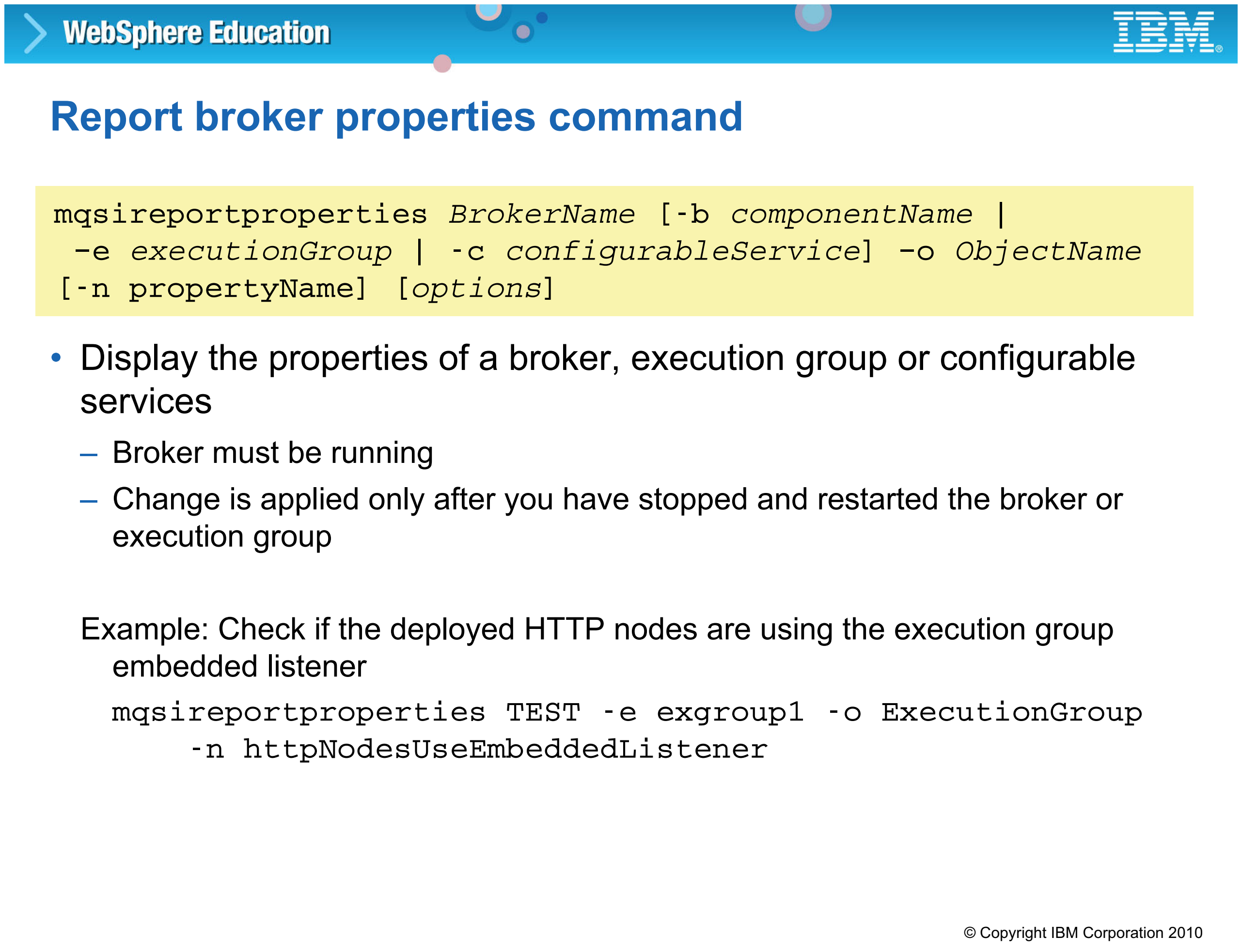
Instructor notes:

Purpose — Discuss the `mqsichangeproperties` command.

Details —

Additional information —

Transition statement — There is also a command to display the broker properties.



WebSphere Education
IBM

Report broker properties command

```
mqsireportproperties BrokerName [-b componentName |
 -e executionGroup | -c configurableService] -o ObjectName
[-n propertyName] [options]
```

- Display the properties of a broker, execution group or configurable services
  - Broker must be running
  - Change is applied only after you have stopped and restarted the broker or execution group

Example: Check if the deployed HTTP nodes are using the execution group embedded listener

```
mqsireportproperties TEST -e exgroup1 -o ExecutionGroup
    -n httpNodesUseEmbeddedListener
```

Figure 4-12. Report broker properties command WM643 / VM6431.0

Notes:

The `mqsireportproperties` command displays broker properties and configurable properties of broker resources. The figure shows a summary of the syntax for the command.

> `-b` is the name of the component selected. Valid values are `httplistener`, `securitycache`, and `servicefederation`.
>
> `-c` is type of the configurable service. Specify a value of AllTypes to report on all configurable service types. For a list of supplied configurable services, and their properties and values, see “Configurable services properties” in the Information Center.
>
> `-e` is the name of execution group for which a report is required.
>
> `-o` is the name of the object whose properties you want to read.

See the Information Center for a complete description of the command syntax.

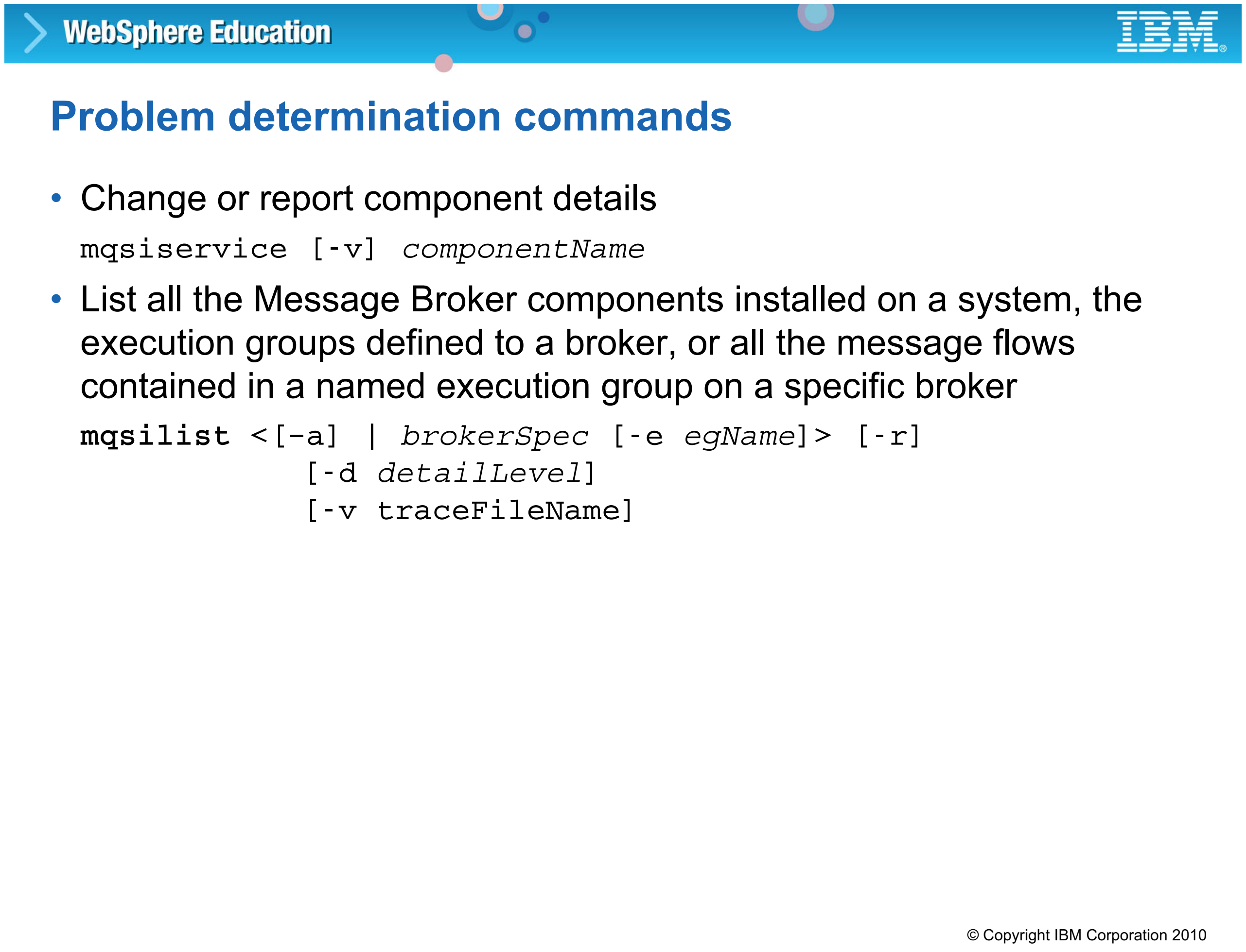
Instructor notes:

Purpose — Describe the command that is used to report broker properties.

Details —

Additional information —

Transition statement — There are also commands for problem determination.



WebSphere Education

IBM

Problem determination commands

- Change or report component details

  `mqsiservice [-v] componentName`

- List all the Message Broker components installed on a system, the execution groups defined to a broker, or all the message flows contained in a named execution group on a specific broker

```
mqsilist <[-a] | brokerSpec [-e egName]> [-r]
            [-d detailLevel]
            [-v traceFileName]
```

© Copyright IBM Corporation 2010

Figure 4-13. Problem determination commands WM643 / VM6431.0

Notes:

Use the `mqsiservice` command to report on the configuration of one or more brokers.

Use the `mqsilist` command with no options to list all of the components installed on the system. For example:

```
C:\Program Files\IBM\MQSI\7.0>mqsiservice MB7BROKER
...
Install Path = C:\Program Files\IBM\MQSI\7.0
Shared Work Path = NOT_HA_ENABLED_BROKER
Local Work Path = C:\Documents and Settings\All Users\Application
Data\IBM\MQSI
Component UUID = 57ec3c14-a000-45b4-bb30-09849a5cfbbd
process id = 4980
broker db userId = JSMITH
broker db password =
queue manager = MB7QMGR
pubsub migration = no
```

```
fastpath Queue Manager = no
configuration timeout = 300
configuration delay timeout = 60
statistics major interval = 60
ComponentType = Broker
Fixpack capability level =  (effective level 7.0.0.1)

BIP8071I: Successful command completion.
```

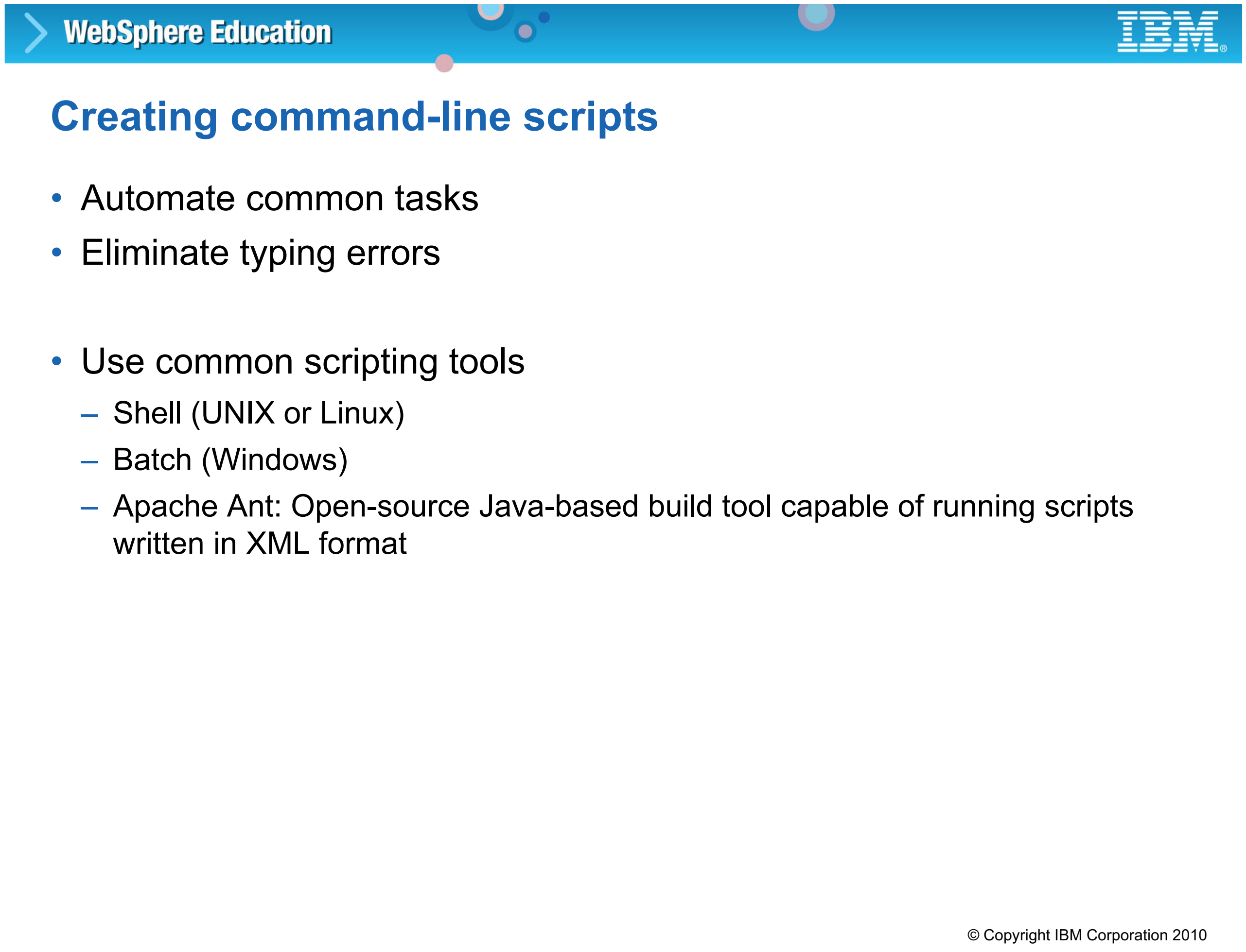
Instructor notes:

Purpose — Introduce the trace facilities and associated commands.

Details — These commands were introduced in Unit 2. This is review.

Additional information — None.

Transition statement — Next: Creating command line scripts



WebSphere Education

Creating command-line scripts

- Automate common tasks
- Eliminate typing errors

- Use common scripting tools
  - Shell (UNIX or Linux)
  - Batch (Windows)
  - Apache Ant: Open-source Java-based build tool capable of running scripts written in XML format

Figure 4-14. Creating command line scripts WM643 / VM6431.0

Notes:

WebSphere Message Broker commands can be incorporated into batch files and command-line scripts to automate common tasks such as modifying configurable properties in a BAR file.

Any scripting tool can be used, provided the user running the script has the correct permissions. One of the newer tools is Apache Ant, which is an open source Java-based build tool.

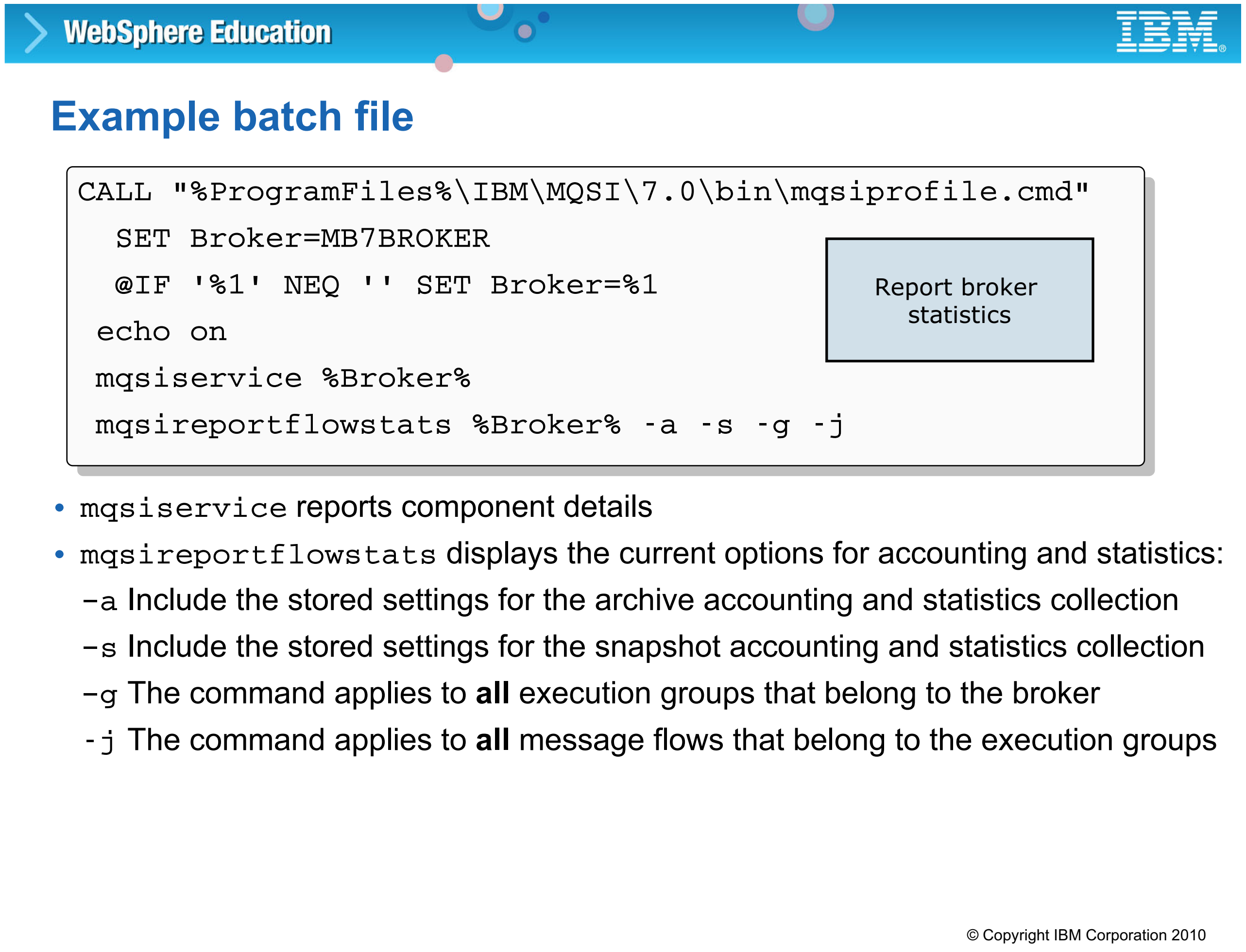
Instructor notes:

Purpose — Provide an overview of command-line scripts.

Details —

Additional information —

Transition statement — The next pages show examples of batch and Ant scripts.



WebSphere Education

Example batch file

```
CALL "%ProgramFiles%\IBM\MQSI\7.0\bin\mqsiprofile.cmd"
  SET Broker=MB7BROKER
  @IF '%1' NEQ '' SET Broker=%1
 echo on
 mqsiservice %Broker%
 mqsireportflowstats %Broker% -a -s -g -j
```

Report broker statistics

- `mqsiservice` reports component details
- `mqsireportflowstats` displays the current options for accounting and statistics:
  - `-a` Include the stored settings for the archive accounting and statistics collection
  - `-s` Include the stored settings for the snapshot accounting and statistics collection
  - `-g` The command applies to **all** execution groups that belong to the broker
  - `-j` The command applies to **all** message flows that belong to the execution groups

Figure 4-15. Example batch file

WM643 / VM6431.0

Notes:

The example batch file in the figure, reports broker statistics for all execution groups and all message flows.

The command has an optional argument for specifying the broker. If a broker is not specified, the command is run against the broker named `MB7BROKER`.

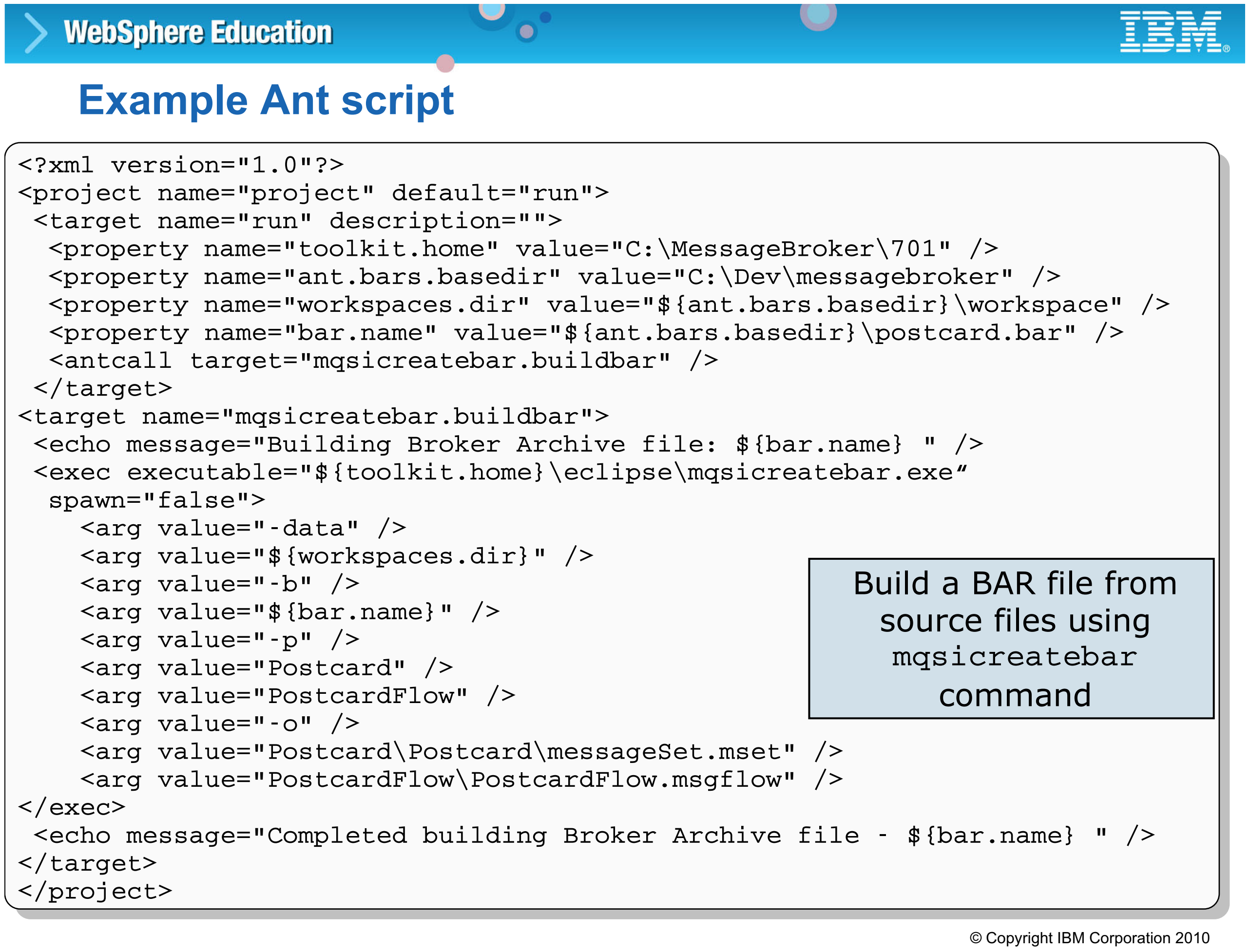
Instructor notes:

Purpose — Show an example batch file.

Details —

Additional information —

Transition statement — Next: Example Ant script



WebSphere Education

Example Ant script

```
<?xml version="1.0"?>
<project name="project" default="run">
 <target name="run" description="">
  <property name="toolkit.home" value="C:\MessageBroker\701" />
  <property name="ant.bars.basedir" value="C:\Dev\messagebroker" />
  <property name="workspaces.dir" value="${ant.bars.basedir}\workspace" />
  <property name="bar.name" value="${ant.bars.basedir}\postcard.bar" />
  <antcall target="mqsicreatebar.buildbar" />
 </target>
<target name="mqsicreatebar.buildbar">
 <echo message="Building Broker Archive file: ${bar.name} " />
 <exec executable="${toolkit.home}\eclipse\mqsicreatebar.exe“
  spawn="false">
    <arg value="-data" />
    <arg value="${workspaces.dir}" />
    <arg value="-b" />
    <arg value="${bar.name}" />
    <arg value="-p" />
    <arg value="Postcard" />
    <arg value="PostcardFlow" />
    <arg value="-o" />
    <arg value="Postcard\Postcard\messageSet.mset" />
    <arg value="PostcardFlow\PostcardFlow.msgflow" />
</exec>
 <echo message="Completed building Broker Archive file - ${bar.name} " />
</target>
</project>
```

Build a BAR file from source files using `mqsicreatebar` command

Figure 4-16. Example Ant script WM643 / VM6431.0

Notes:

The figure shows an example of an Ant script that builds a BAR file from source files using the `mqsicreatebar` command. Ant is like a Java version of “make”. Ant scripts are written in XML, and as with “make”, Ant targets can depend on other targets.

The script builds and runs the `mqsicreatebar` command with the following options:

> `-data` identifies the workspace directory that contains the Message Set and Message Flow projects.
>
> `-b` identifies the BAR file name.
>
> `-p` specifies the projects to include in the BAR file.
>
> `-o` identifies the workspace relative path (including the project) of a message flow (`.msgflow`) file or message set (`.mset`) file to add to the BAR file.

See the IBM developerWorks topic “WebSphere Message Broker deployment scripting using Ant” for more details.

Instructor notes:

Purpose — Provide an example of an Ant script.

Details — Using Ant to modify configurable properties reduces manual operations and avoids common build errors, thus providing a consistent and reliable way to build and maintain the WebSphere Message Broker deployment artifacts known as broker archive (BAR) files.

A developerWorks topic (`http://www.ibm.com/developerworks/websphere/library/techarticles/0706_spriet/0706_spriet.html`) shows how to use Ant to automate the build and deployment process.

Additional information — None.

Transition statement — Next: WebSphere Message Broker Explorer

4.2. Administering Message Broker using Message Broker Explorer

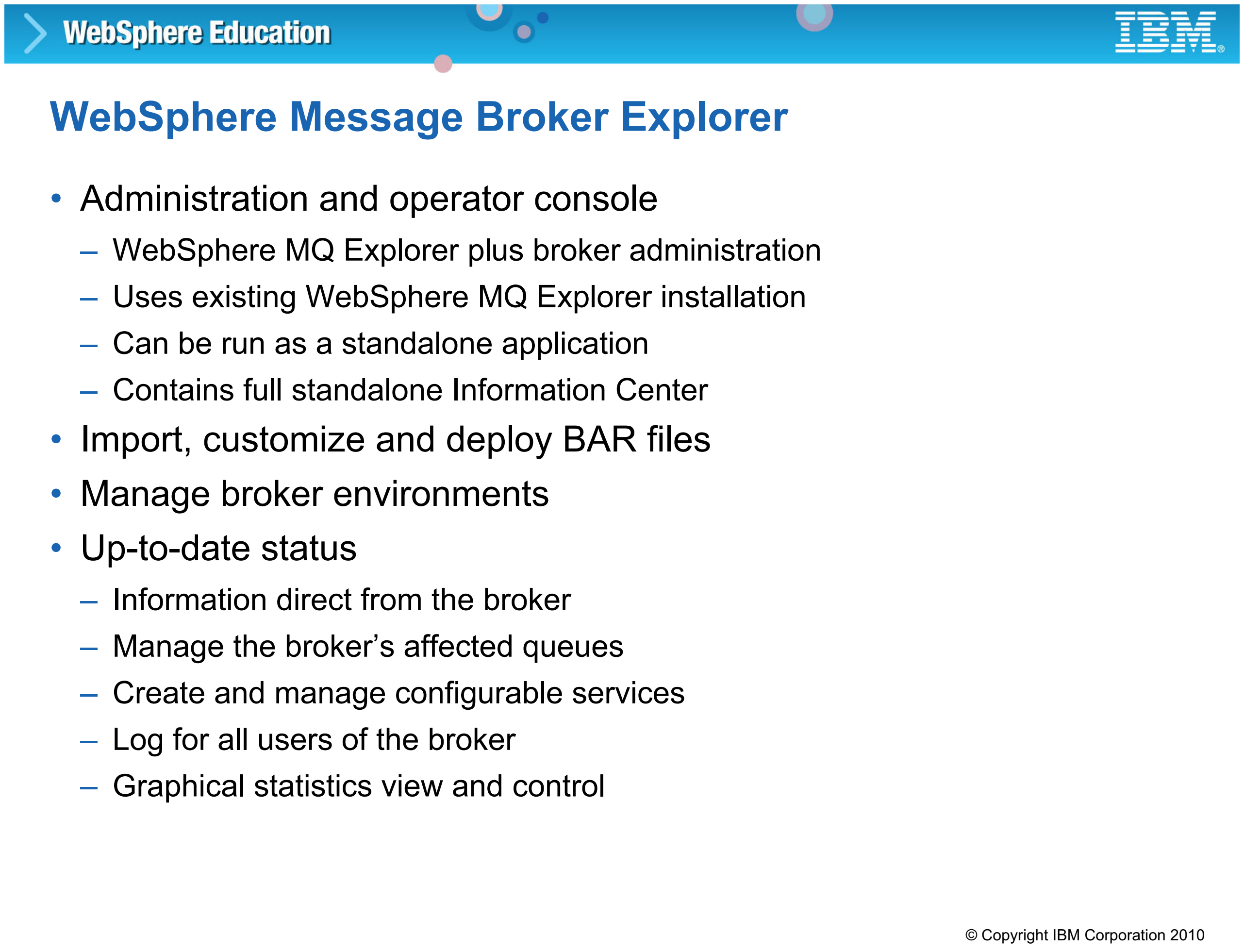
Instructor topic introduction

What students will do — Students will learn about WebSphere Message Broker Explorer, including many of the options available for management of the broker. They will learn that this is an alternative to using the command-line on a Windows or Linux platform.

How students will do it — Listen to the lecture and participate in classroom discussion. Exercise 3 will give students the chance to work with WebSphere Message Broker Explorer.

What students will learn — Students will be able to appreciate the use of Message Broker Explorer as a means to control the operation of the broker.

How this will help students in their job — Students will be prepared to consider using the WebSphere Message Broker Explorer to perform some of their normal administrative tasks.



WebSphere Education

WebSphere Message Broker Explorer

- Administration and operator console
  - WebSphere MQ Explorer plus broker administration
  - Uses existing WebSphere MQ Explorer installation
  - Can be run as a standalone application
  - Contains full standalone Information Center
- Import, customize and deploy BAR files
- Manage broker environments
- Up-to-date status
  - Information direct from the broker
  - Manage the broker’s affected queues
  - Create and manage configurable services
  - Log for all users of the broker
  - Graphical statistics view and control

Figure 4-17. WebSphere Message Broker Explorer WM643 / VM6431.0

Notes:

WebSphere Message Broker Explorer is used to administer brokers in a WebSphere Message Broker V7 environment. It consists of WebSphere MQ Explorer plus WebSphere Message Broker administration.

With the tight integration between WebSphere MQ and WebSphere Message Broker, it makes sense to have a single administration interface and take advantage of some of the WebSphere MQ features. It makes managing your brokers just like managing your queue managers. It also reduces the time to learn operational skills for WebSphere Message Broker; if the operator is already familiar with WebSphere MQ Explorer, they do not need to learn a new interface.

WebSphere Message Broker Explorer is a stand-alone application; there is no requirement to have the full toolkit installed in administration-only environments. WebSphere Message Broker Explorer has minimal effect on broker performance.

WebSphere Message Broker Explorer also contains a complete, stand-alone Information Center for access to documentation and help.

Instructor notes:

Purpose — Review the features of WebSphere Message Broker Explorer.

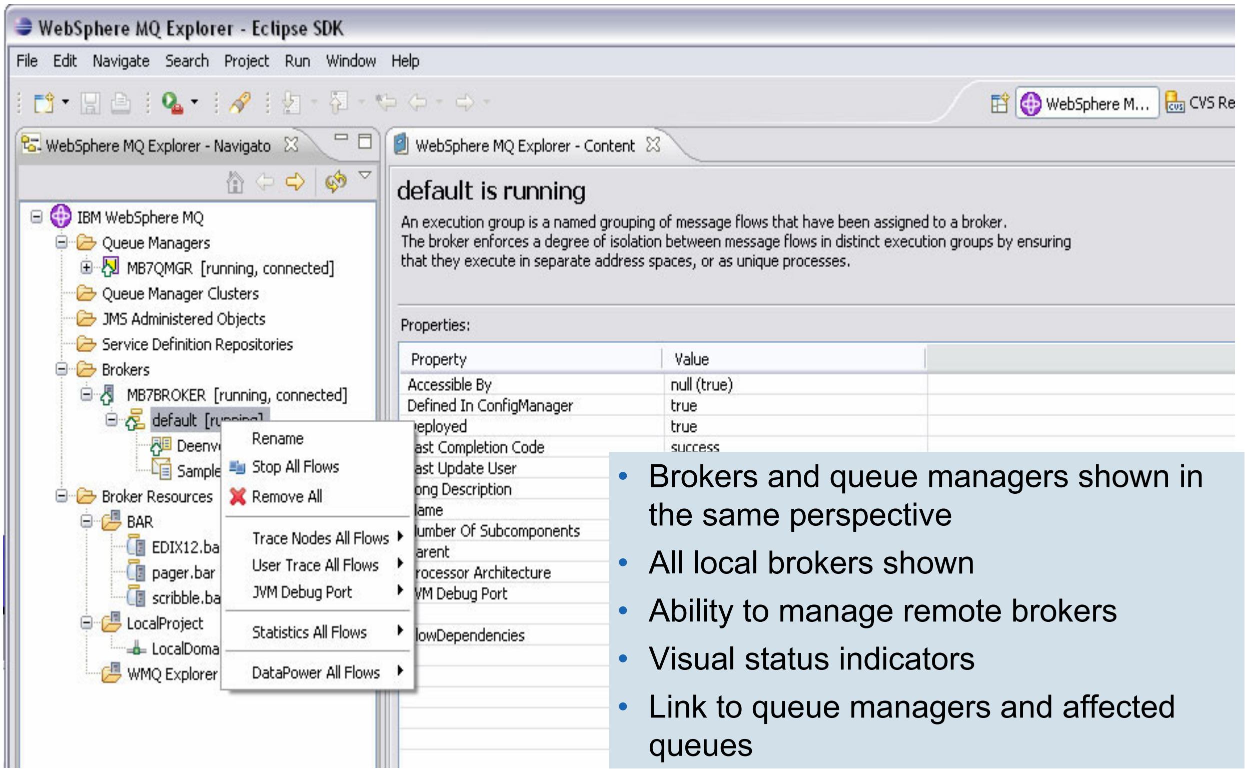
Details — WebSphere Message Broker Explorer was introduced in an earlier unit so this is a review.

Additional information — None.

Transition statement — WebSphere Message Broker Explorer is basically WebSphere MQ Explorer with enhancements for managing broker and broker components.

WebSphere Education

Graphical administration consistent with MQ



default is running

An execution group is a named grouping of message flows that have been assigned to a broker. The broker enforces a degree of isolation between message flows in distinct execution groups by ensuring that they execute in separate address spaces, or as unique processes.

Properties:

Property	Value
Accessible By	null (true)
Defined In ConfigManager	true
eployed	true
ast Completion Code	success
ast Update User	
ong Description	
lame	
lumber Of Subcomponents	
arent	
rocessor Architecture	
VM Debug Port	
lowDependencies	

- Brokers and queue managers shown in the same perspective
- All local brokers shown
- Ability to manage remote brokers
- Visual status indicators
- Link to queue managers and affected queues

Figure 4-18. Graphical administration consistent with MQ WM643 / VM6431.0

Notes:

WebSphere Message Broker Explorer is WebSphere MQ Explorer with broker and broker resources, making it easy to open queues and queue managers, and navigate between brokers and queue managers.

Use the **Brokers** folder to create, view, and modify brokers. Local brokers are discovered automatically.

Use the ***Broker Archive Files*** folder to import, view, and modify broker archive files before deploying them to your brokers.

Instructor notes:

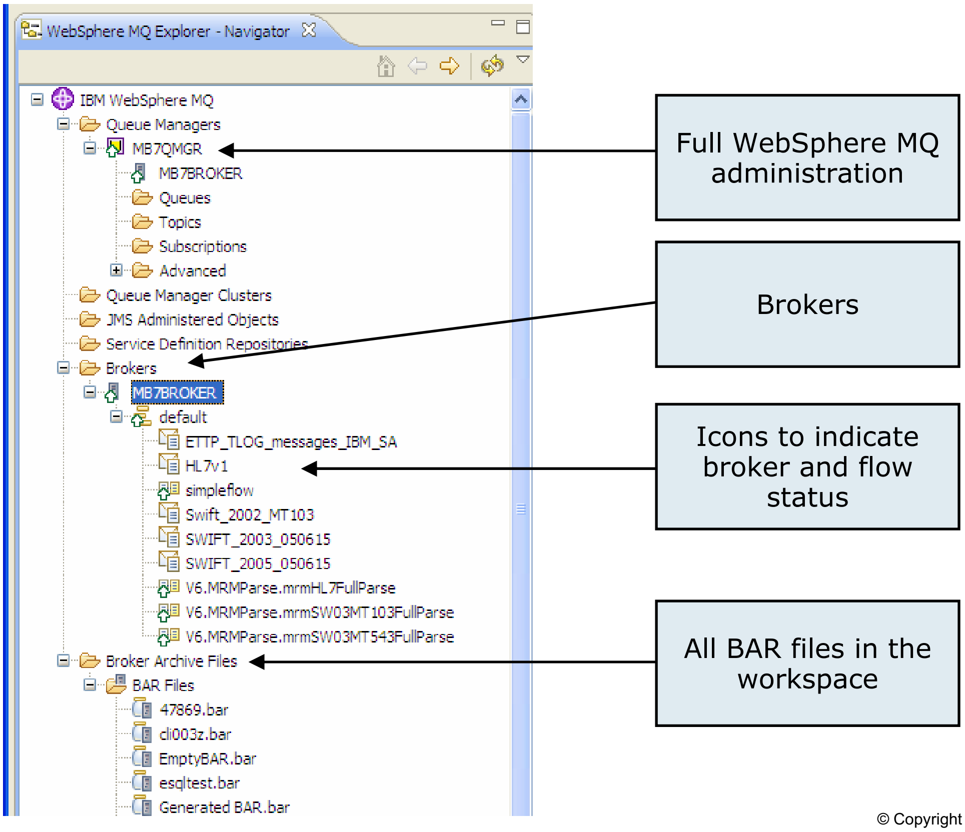
Purpose — Show WebSphere Message Broker Explorer and describe the relationship with WebSphere MQ Explorer.

Details — Emphasize that WebSphere Message Broker Explorer is only available on Windows and Linux on X86, so it does not replace the command line.

Additional information — None

Transition statement — The WebSphere Message Broker Explorer navigator is the starting point for managing and configuring the broker and broker components.

WebSphere Message Broker Explorer: Navigator



WebSphere MQ Explorer - Navigator

```
IBM WebSphere MQ
  Queue Managers
    MB7QMGR
      MB7BROKER
      Queues
      Topics
      Subscriptions
      Advanced
  Queue Manager Clusters
  JMS Administered Objects
  Service Definition Repositories
  Brokers
    MB7BROKER
      default
        ETTP_TLOG_messages_IBM_SA
        HL7v1
        simpleflow
        Swift_2002_MT103
        SWIFT_2003_050615
        SWIFT_2005_050615
        V6.MRMParse.mrmHL7FullParse
        V6.MRMParse.mrmSW03MT103FullParse
        V6.MRMParse.mrmSW03MT543FullParse
  Broker Archive Files
    BAR Files
      47869.bar
      cli003z.bar
      EmptyBAR.bar
      esqltest.bar
      Generated BAR.bar
```

Full WebSphere MQ administration

Brokers

Icons to indicate broker and flow status

All BAR files in the workspace

Figure 4-19. WebSphere Message Broker Explorer: Navigator WM643 / VM6431.0

Notes:

WebSphere Message Broker Explorer has a separate folder in the navigator for brokers and broker archive files.

The **Brokers** folder contains a subtree for each broker. All local brokers display automatically; remote brokers must be added manually. The example in the figure shows a local broker named `MB7BROKER` which is the default broker created by the Default Configuration wizard. Each broker subtree contains the execution groups defined on the broker and the message flows deployed to each execution group.

Similar to WebSphere MQ Explorer, the navigator in WebSphere Message Broker Explorer uses status indicators to identify the current state of the broker, execution groups, and message flows.

The **Broker Archive Files** folder contains the BAR files. All BAR files in the workspace are discovered automatically and displayed in a separate subfolder in the navigator. To deploy a BAR file, you can drag the BAR file to the execution group, also listed in the navigator, even though the BAR files might actually exist in separate directories on disk.

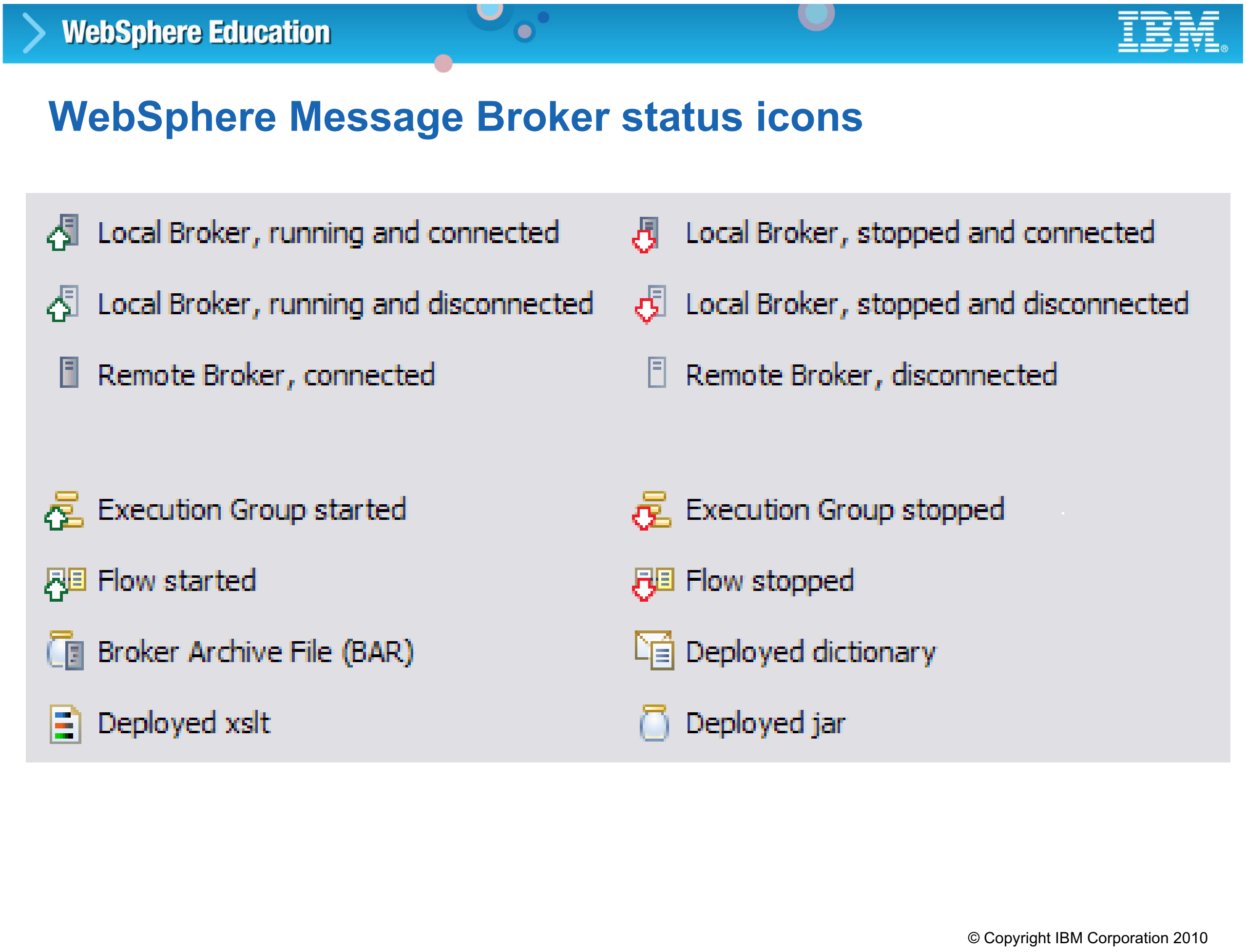
Instructor notes:

Purpose — Highlight the folders added to WebSphere MQ for managing brokers and broker components.

Details —

Additional information —

Transition statement — The navigator in WebSphere Message Broker Explorer uses status indicators to identify the current state of the broker, execution groups, and message flows.



WebSphere Education

IBM

WebSphere Message Broker status icons

Local Broker, running and connected

Local Broker, stopped and connected

Local Broker, running and disconnected

Local Broker, stopped and disconnected

Remote Broker, connected

Remote Broker, disconnected

Execution Group started

Execution Group stopped

Flow started

Flow stopped

Broker Archive File (BAR)

Deployed dictionary

Deployed xslt

Deployed jar

Figure 4-20. WebSphere Message Broker status icons WM643 / VM6431.0

Notes:

The icons displayed in the WebSphere Message Broker Explorer Navigator view indicate the status of the WebSphere MQ and broker objects.

The figure shows the status icons that might appear in the Navigator view.

Instructor notes:

Purpose — Describe the WebSphere Message Broker Explorer Navigator icons.

Details —

Additional information —

Transition statement — If you are managing a large network with many brokers, you can create broker sets to make it easier to monitor broker status.

WebSphere Education IBM

Grouping brokers using broker sets

- Create a broker set to visually group brokers in WebSphere Message Broker Explorer using tags
  - Manual broker set is empty until brokers are added to the broker set from the **All** broker set or another broker set
  - Automatic broker set uses broker tags to dynamically add and remove brokers from a set based on provided values or the current state of the brokers
- Similar to grouping queue managers

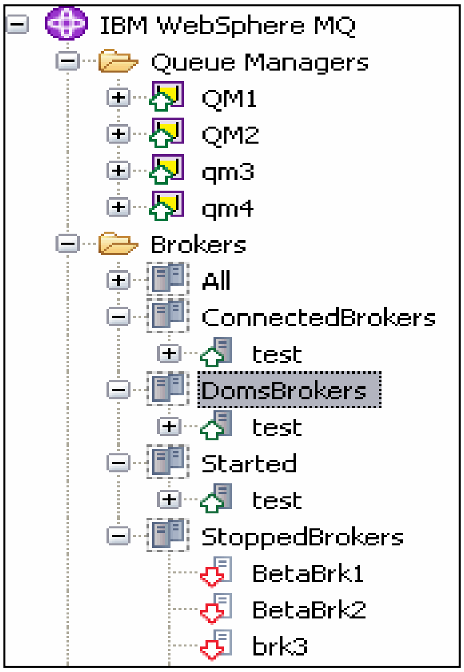


Figure 4-21. Grouping brokers using broker sets WM643 / VM6431.0

Notes:

Within WebSphere Message Broker Explorer, you can group brokers using tags such as broker state, for example, similar to grouping queue managers. Dynamic tags are also supported for started, stopped, connected, and disconnected,

WebSphere Message Broker Explorer supports dynamic grouping. For example, you can select a group of brokers that are stopped, and put them into *Started* group and they will all start.

The figure shows an example of brokers grouped by status, such as **StoppedBrokers**, and usage, such as **DomsBrokers**.

Instructor notes:

Purpose — Describe the purpose of broker sets.

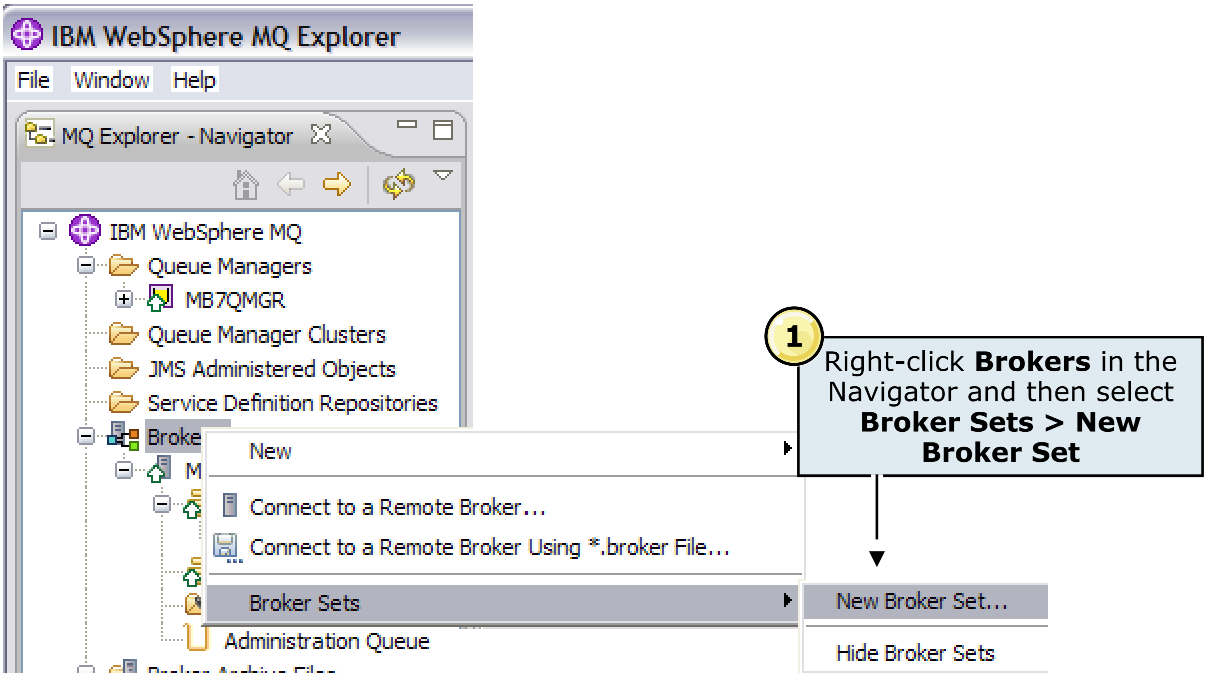
Details —

Additional information —

Transition statement — The next series of slides show the steps for creating broker sets.

WebSphere Education

Creating a broker set (1 of 3)



1 Right-click **Brokers** in the Navigator and then select **Broker Sets > New Broker Set**

Figure 4-22. Creating a broker set (1 of 3)

WM643 / VM6431.0

Notes:

To create a broker set in WebSphere Message Broker Explorer:

1. Right-click **Brokers** in the Navigator and then select **Broker Sets > New Broker Set** from the menu.

Instructor notes:

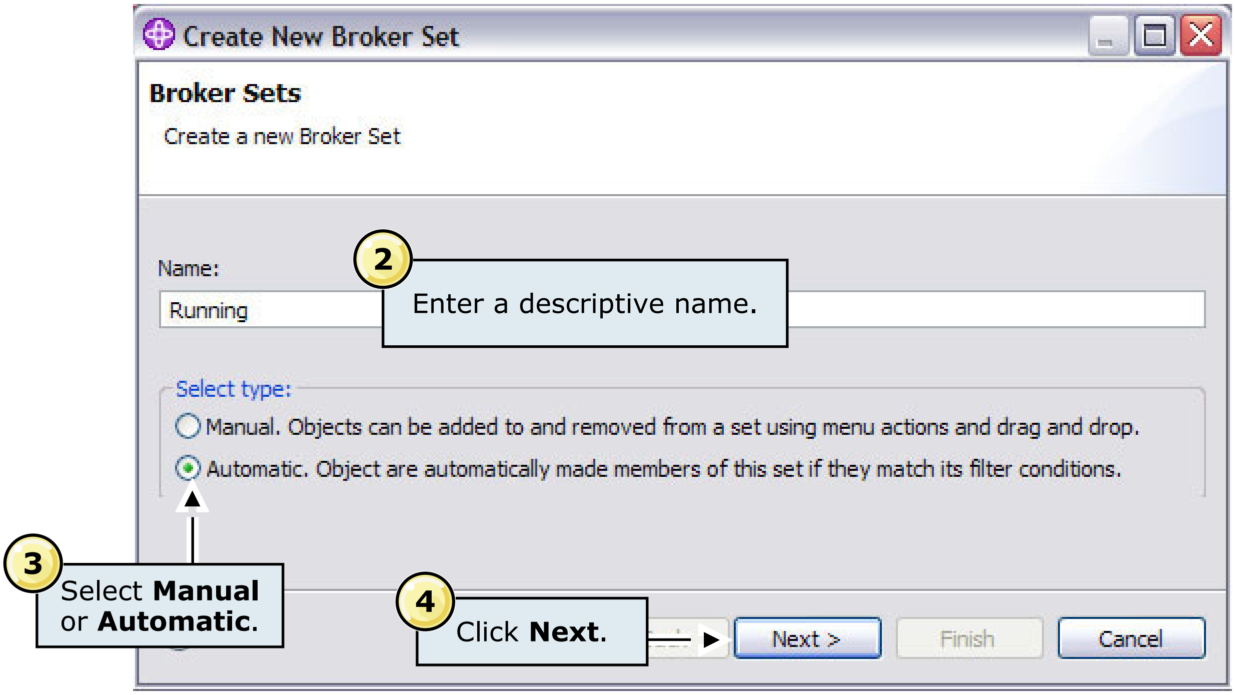
Purpose — Shows the first step for creating a broker set.

Details —

Additional information —

Transition statement — Next: Creating a broker set (2 of 3)

Creating a broker set (2 of 3)



Create New Broker Set

Broker Sets

Create a new Broker Set

Name: Running

Select type:

○ Manual. Objects can be added to and removed from a set using menu actions and drag and drop.

● Automatic. Object are automatically made members of this set if they match its filter conditions.

2 Enter a descriptive name.

3 Select **Manual** or **Automatic**.

4 Click **Next**.

Figure 4-23. Creating a broker set (2 of 3) WM643 / VM6431.0

Notes:

In the Create New Broker Set wizard:

2. Enter a name for the broker set in the **Name** field.
3. Select the **Manual** or **Automatic** radio button to indicate whether the broker should be added to the group automatically based on matching filter conditions, or manually.
4. Click **Next**.

Instructor notes:

Purpose — Show the next steps for creating a broker set.

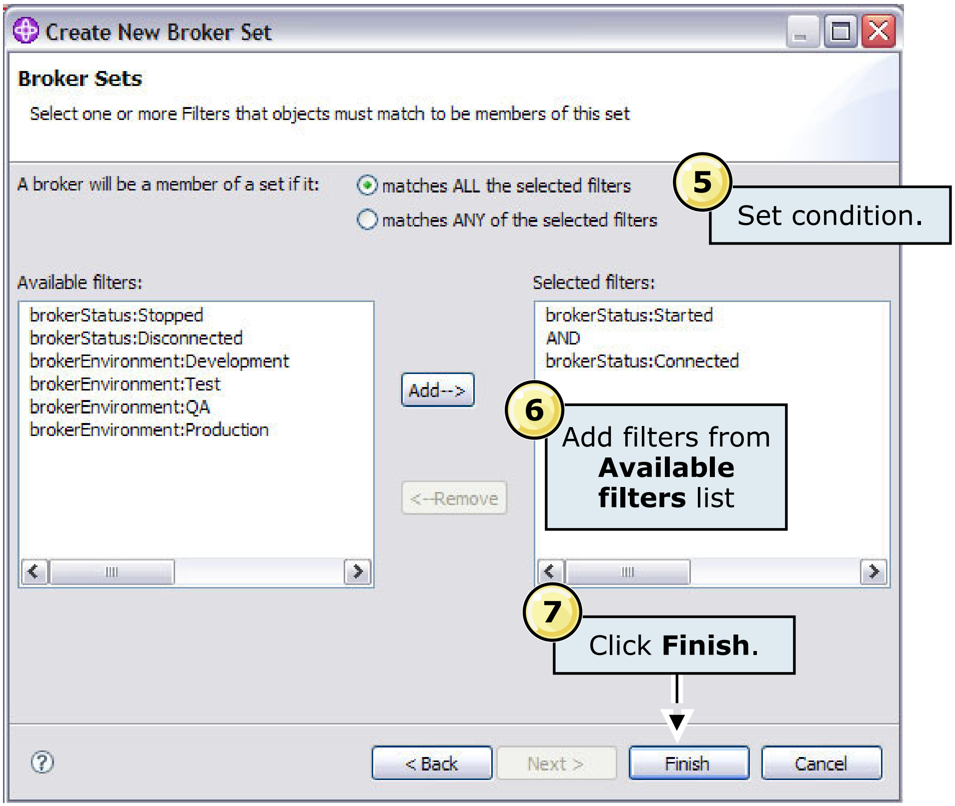
Details —

Additional information —

Transition statement — Next: Creating a broker set (3 of 3)

WebSphere Education

Creating a broker set (3 of 3)



Create New Broker Set

Broker Sets

Select one or more Filters that objects must match to be members of this set

A broker will be a member of a set if it:

● matches ALL the selected filters

○ matches ANY of the selected filters

Available filters:

brokerStatus:Stopped
brokerStatus:Disconnected
brokerEnvironment:Development
brokerEnvironment:Test
brokerEnvironment:QA
brokerEnvironment:Production

Selected filters:

brokerStatus:Started
AND
brokerStatus:Connected

5 Set condition.

6 Add filters from **Available filters** list

7 Click **Finish**.

Figure 4-24. Creating a broker set (3 of 3) WM643 / VM6431.0

Notes:

The final steps for creating a broker set are to define the filters:

5. Set the condition that determines whether a broker is a member of a broker set. The options are:

   - A broker is a member of a set if it matches **all** the selected filters.
   - A broker is a member of a set if it matches **any** of the selected filters.

6. Create the filters by selecting the filter in the Available filters list and clicking **Add >.**

   If you selected the **matches all of the selected filters** radio button, an AND appears between each filter, indicating that all the filters must evaluate to ‘true’ before the broker is added to the broker set.

   If you selected the **matches any of the selected filters** radio button, an OR appears between each filter, indicating that only one of the filters must evaluate to ‘true’ before the broker is added to the broker set.

7. When you have finished defining the filters, click **Finish**.

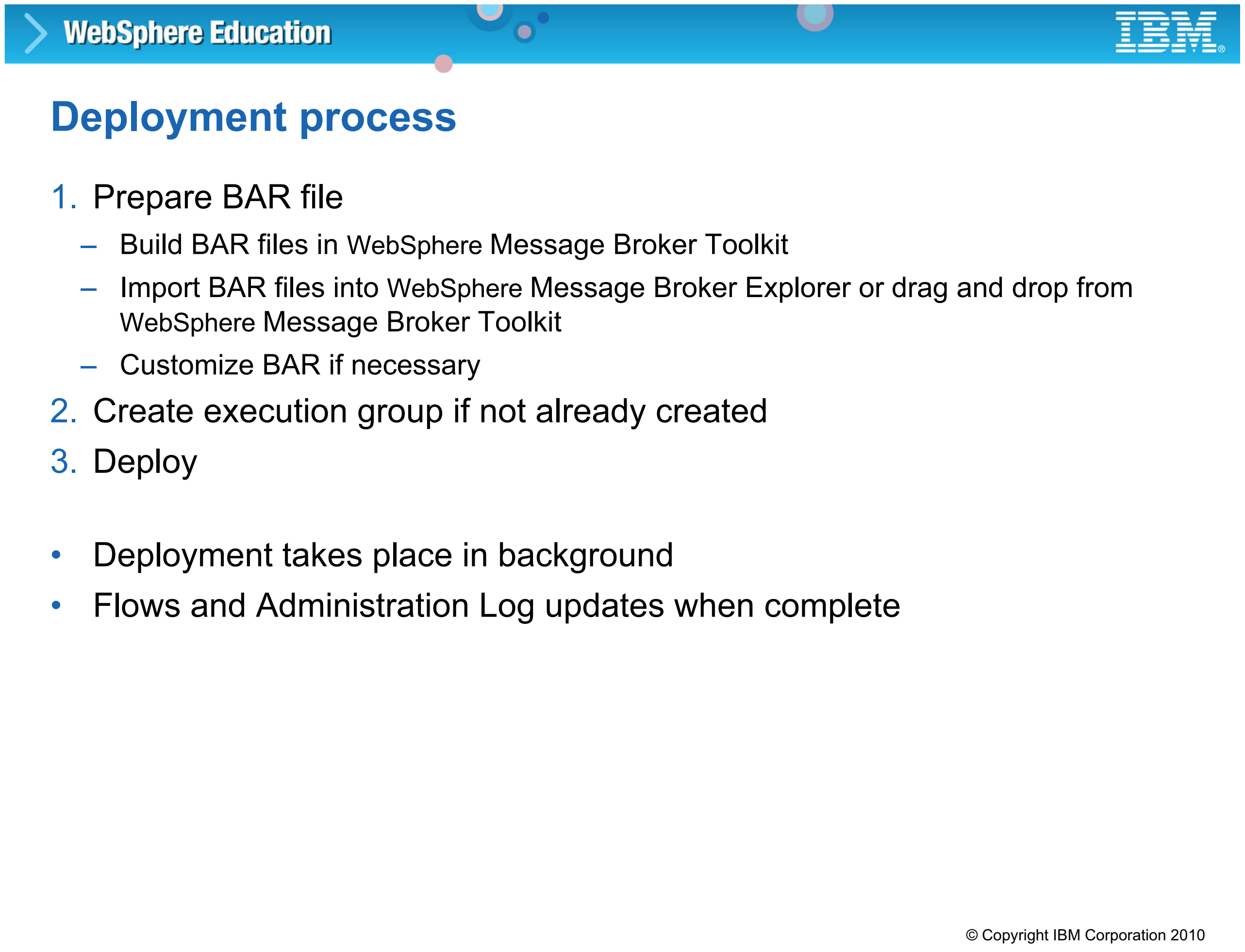
Instructor notes:

Purpose — Shows the final steps for creating a broker set.

Details — In the example, the broker must be started and connected before it appears in the broker set.

Additional information —

Transition statement — You can also deploy BAR files from the WebSphere Message Broker Explorer.



WebSphere Education

Deployment process

1. Prepare BAR file
   - Build BAR files in WebSphere Message Broker Toolkit
   - Import BAR files into WebSphere Message Broker Explorer or drag and drop from WebSphere Message Broker Toolkit
   - Customize BAR if necessary
2. Create execution group if not already created
3. Deploy

- Deployment takes place in background
- Flows and Administration Log updates when complete

Figure 4-25. Deployment process WM643 / VM6431.0

Notes:

With WebSphere Message Broker Explorer, the deployment process is much more flexible and non-intrusive.

The administration log information is written to the **Administration Log** view in the WebSphere Message Broker Explorer. The Administration Log view displays actions performed on the broker by all users. Alternatively, you can view the results of deployment actions from a single user using the **Deployment Log** view in the WebSphere Message Broker Toolkit.

The Administration Log view contains information about events that occur within your brokers. These events can be information, errors, or warnings and relate to your own actions. To view events for a particular broker, look for the name of the broker in the **Source** column.

Each event contains the following information:

Message: The event number.

Source: Where the event has come from.

TimeStamp: The date and time that the event occurred. Time stamps are taken from the computer that is hosting the broker.

Details: What has caused the event and what action is needed to rectify it.

.

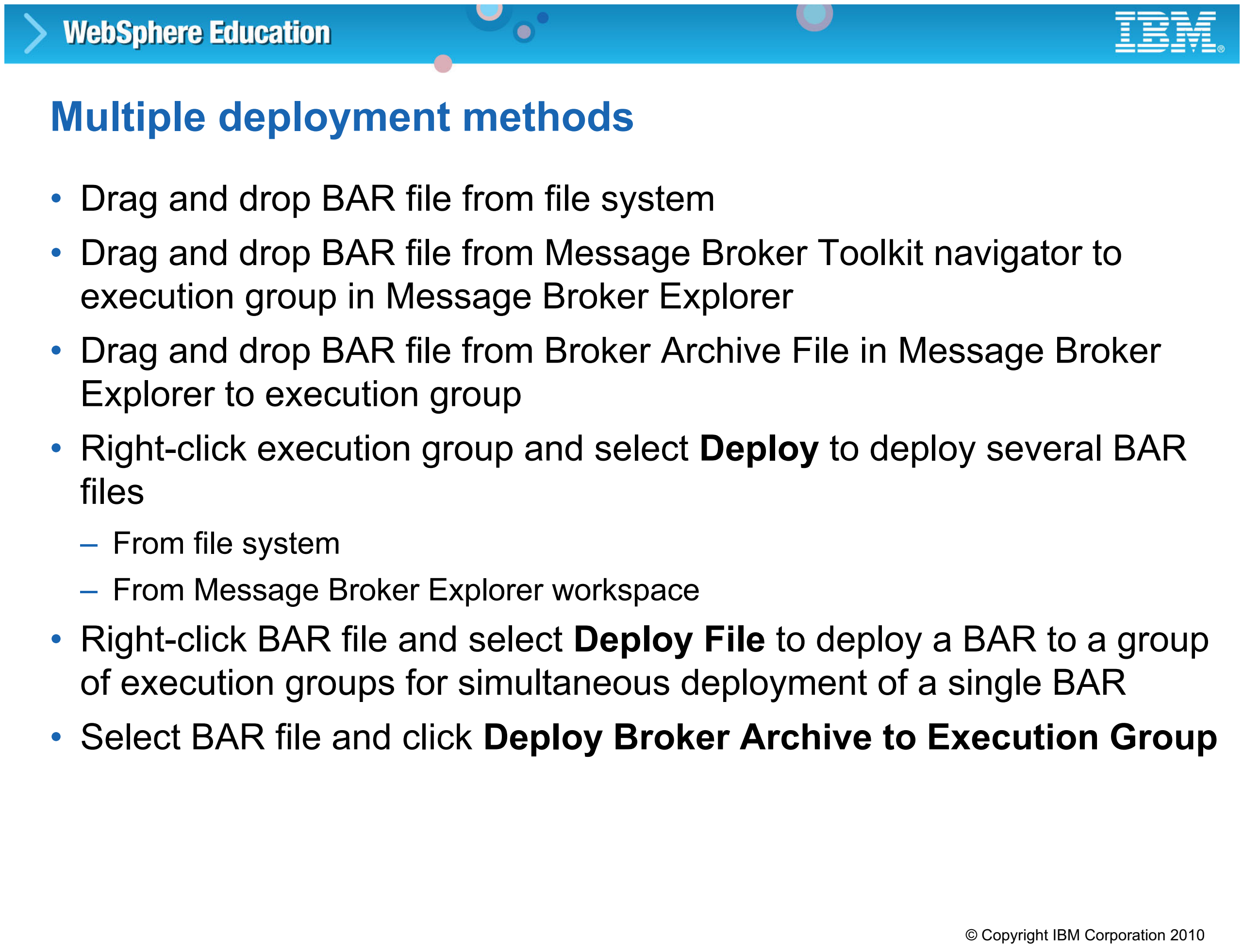
Instructor notes:

Purpose — Describe the BAR file process using WebSphere Message Broker Explorer.

Details — The Administration Log in the WebSphere Message Broker Explorer is comparable to the Deployment Log in the WebSphere Message Broker Toolkit.

Additional information —

Transition statement — There are many ways to deploy a BAR file using WebSphere Message Broker Explorer



WebSphere Education
IBM

Multiple deployment methods

- Drag and drop BAR file from file system
- Drag and drop BAR file from Message Broker Toolkit navigator to execution group in Message Broker Explorer
- Drag and drop BAR file from Broker Archive File in Message Broker Explorer to execution group
- Right-click execution group and select **Deploy** to deploy several BAR files
  - From file system
  - From Message Broker Explorer workspace
- Right-click BAR file and select **Deploy File** to deploy a BAR to a group of execution groups for simultaneous deployment of a single BAR
- Select BAR file and click **Deploy Broker Archive to Execution Group**

Figure 4-26. Multiple deployment methods WM643 / VM6431.0

Notes:

This figure lists some of the ways that BAR files can be deployed. It is even possible to drag and drop the BAR file from Windows Explorer into WebSphere Message Broker Explorer.

Instructor notes:

Purpose — List many of the ways a BAR file can be deployed using WebSphere Message Broker Explorer.

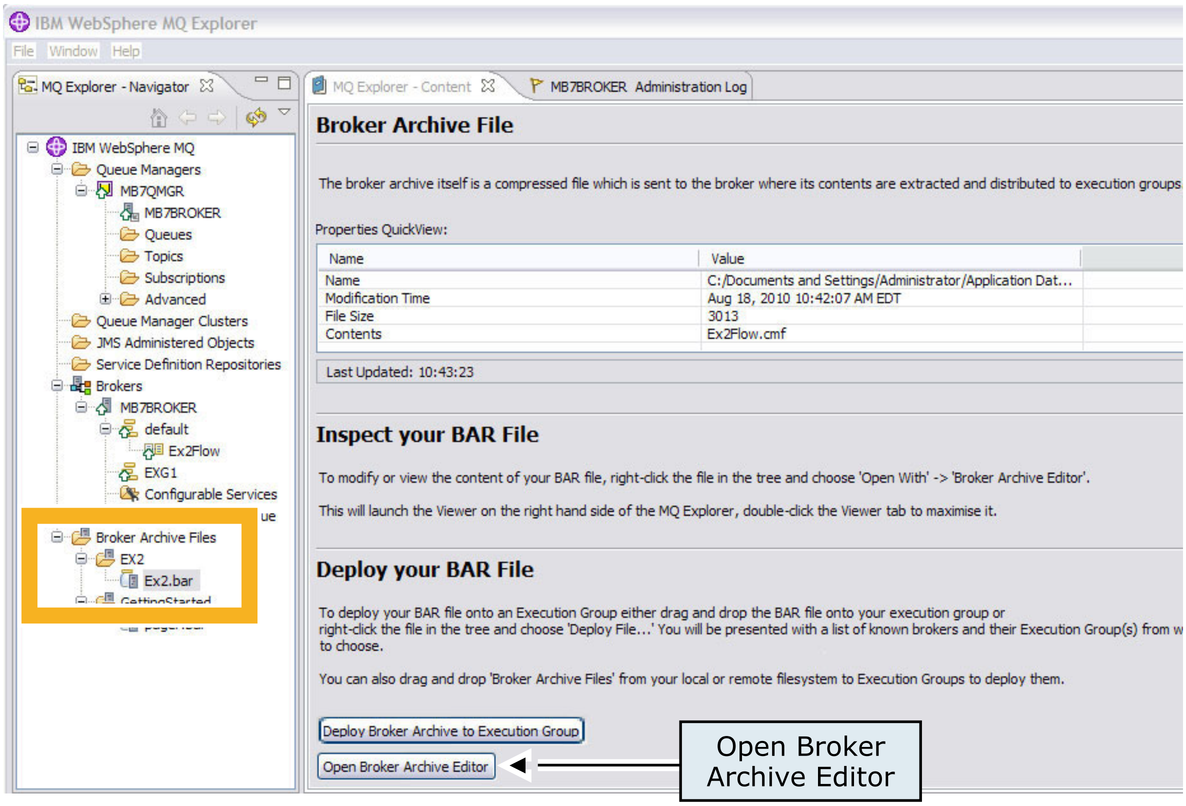
Details —

Additional information —

Transition statement — The Broker Archive Content view displays BAR file properties.

WebSphere Education

Broker Archive File content



Broker Archive File

The broker archive itself is a compressed file which is sent to the broker where its contents are extracted and distributed to execution groups

Properties QuickView:

Name	Value
Name	C:/Documents and Settings/Administrator/Application Dat...
Modification Time	Aug 18, 2010 10:42:07 AM EDT
File Size	3013
Contents	Ex2Flow.cmf

Last Updated: 10:43:23

Inspect your BAR File

To modify or view the content of your BAR file, right-click the file in the tree and choose 'Open With' -> 'Broker Archive Editor'.

This will launch the Viewer on the right hand side of the MQ Explorer, double-click the Viewer tab to maximise it.

Deploy your BAR File

To deploy your BAR file onto an Execution Group either drag and drop the BAR file onto your execution group or right-click the file in the tree and choose 'Deploy File...' You will be presented with a list of known brokers and their Execution Group(s) from w to choose.

You can also drag and drop 'Broker Archive Files' from your local or remote filesystem to Execution Groups to deploy them.

Open Broker Archive Editor

Figure 4-27. Broker Archive File content WM643 / VM6431.0

Notes:

The **Broker Archive File** content view displays properties associated with the BAR file selected in the Navigator view.

From the **Broker Archive File** view you can:

- Review the BAR file properties
- Launch the Deploy wizard to deploy a BAR file to an execution group
- Launch the Broker Archive editor

Instructor notes:

Purpose — Show an example of the Broker Archive File content view.

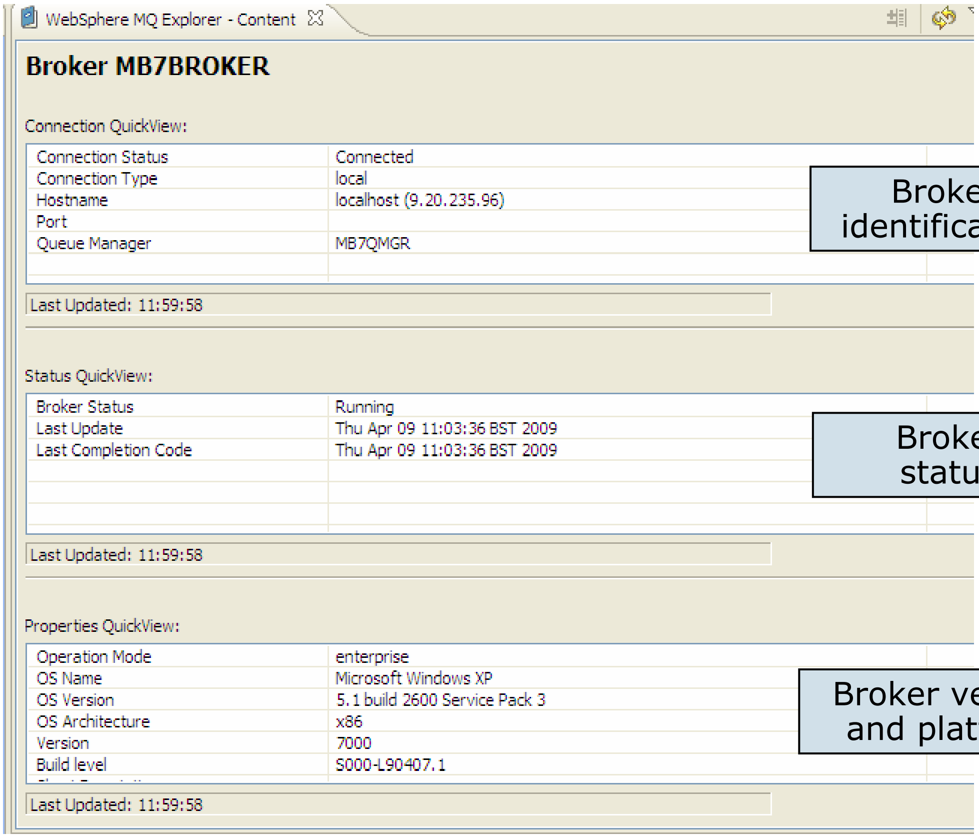
Details — The Broker Archive editor is the same editor that is included in the WebSphere Message Broker Toolkit. Demonstrate launching the Broker Archive editor.

Additional information —

Transition statement — The Content view also shows the properties for brokers, execution groups, and message flows.

WebSphere Education

Broker views



WebSphere MQ Explorer - Content

Broker MB7BROKER

Connection QuickView:

Connection Status	Connected
Connection Type	local
Hostname	localhost (9.20.235.96)
Port	
Queue Manager	MB7QMGR

Last Updated: 11:59:58

Status QuickView:

Broker Status	Running
Last Update	Thu Apr 09 11:03:36 BST 2009
Last Completion Code	Thu Apr 09 11:03:36 BST 2009

Last Updated: 11:59:58

Properties QuickView:

Operation Mode	enterprise
OS Name	Microsoft Windows XP
OS Version	5.1 build 2600 Service Pack 3
OS Architecture	x86
Version	7000
Build level	S000-L90407.1

Last Updated: 11:59:58

Broker identification

Broker status

Broker version and platform

© Copyright IBM Corporation 2010

Figure 4-28. Broker views WM643 / VM6431.0

Notes:

If you select a broker in the WebSphere Message Broker Explorer Navigator, the content view shows three Quick View. Quick Views in WebSphere Message Broker Explorer, similar to Quick Views in WebSphere MQ, provide easy access to important information.

The broker Quick Views contain detailed broker information such the name of the broker, broker status, and broker platform including version and build level of source code.

Instructor notes:

Purpose — Show the broker content view for displaying broker properties

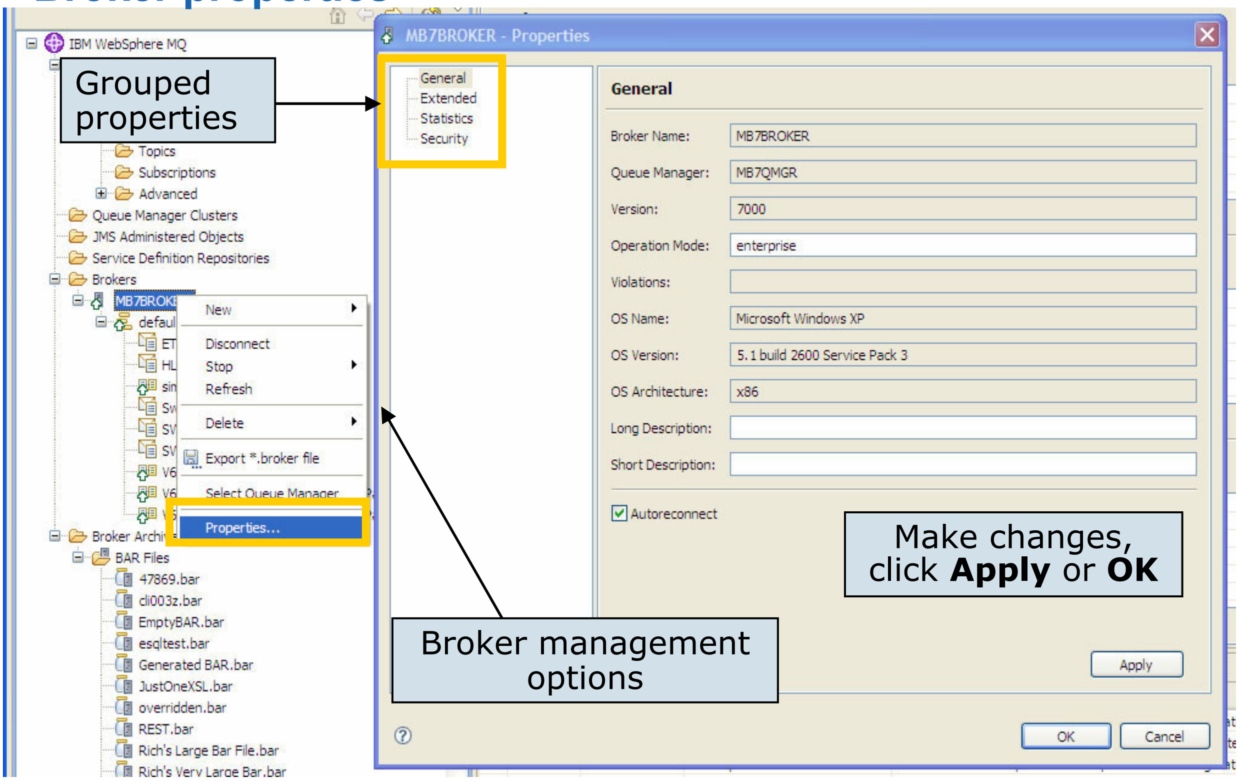
Details —

Additional information —

Transition statement — The QuickViews are read-only, but you can modify some of the broker properties from WebSphere Message Broker Explorer.

WebSphere Education IBM.

Broker properties



General

Broker Name: MB7BROKER

Queue Manager: MB7QMGR

Version: 7000

Operation Mode: enterprise

Violations:

OS Name: Microsoft Windows XP

OS Version: 5.1 build 2600 Service Pack 3

OS Architecture: x86

Long Description:

Short Description:

☑ Autoreconnect

Grouped properties

Broker management options

Make changes, click **Apply** or **OK**

© Copyright IBM Corporation 2010

Figure 4-29. Broker properties WM643 / VM6431.0

Notes:

Options to manage and configure brokers are available by right-clicking a broker in the Navigator. Options include the ability to create new execution groups, connect and disconnect, select a new queue manager, and modify some of the broker properties.

To display and modify some of the broker properties, right-click the broker in the navigator and select **Properties** from the menu.

Property categories in the Broker properties are consistent with WebSphere MQ. The categories are General, Extended, Statistics, and Security. Any fields that are not gray, can be modified. For example, under the General category the operation mode, long description, short description, and autoconnect setting can be modified.

Instructor notes:

Purpose — Show that some broker properties can be modified from WebSphere Message Broker Explorer.

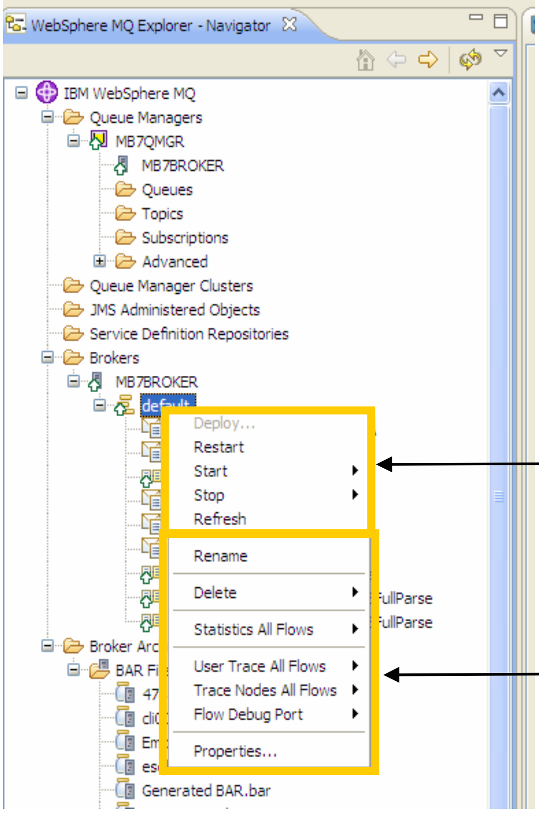
Details —

Additional information —

Transition statement — WebSphere Message Broker Explorer also includes a number of options for managing execution groups.

WebSphere Education

Execution group management (1 of 2)



Control options to deploy, restart, start, stop, and refresh execution groups

Configuration options to rename, delete, enable statistics, trace, and modify execution group properties

Figure 4-30. Execution group management (1 of 2) WM643 / VM6431.0

Notes:

Right-click any execution group in the WebSphere Message Broker Explorer navigator to access the control and configuration options, and display the execution group Quick View of the flows and message sets deployed to the execution group with a status.

Control options are **Restart**, **Start**, **Stop**, and **Refresh**.

Configuration options are **Rename**, **Delete**, **Statistics All Flows**, **User Trace All Flows**, **Trace Nodes All Flows**, **Flow Debug Port,** and **Properties**.

Instructor notes:

Purpose — Show options for managing execution groups in WebSphere Message Broker Explorer.

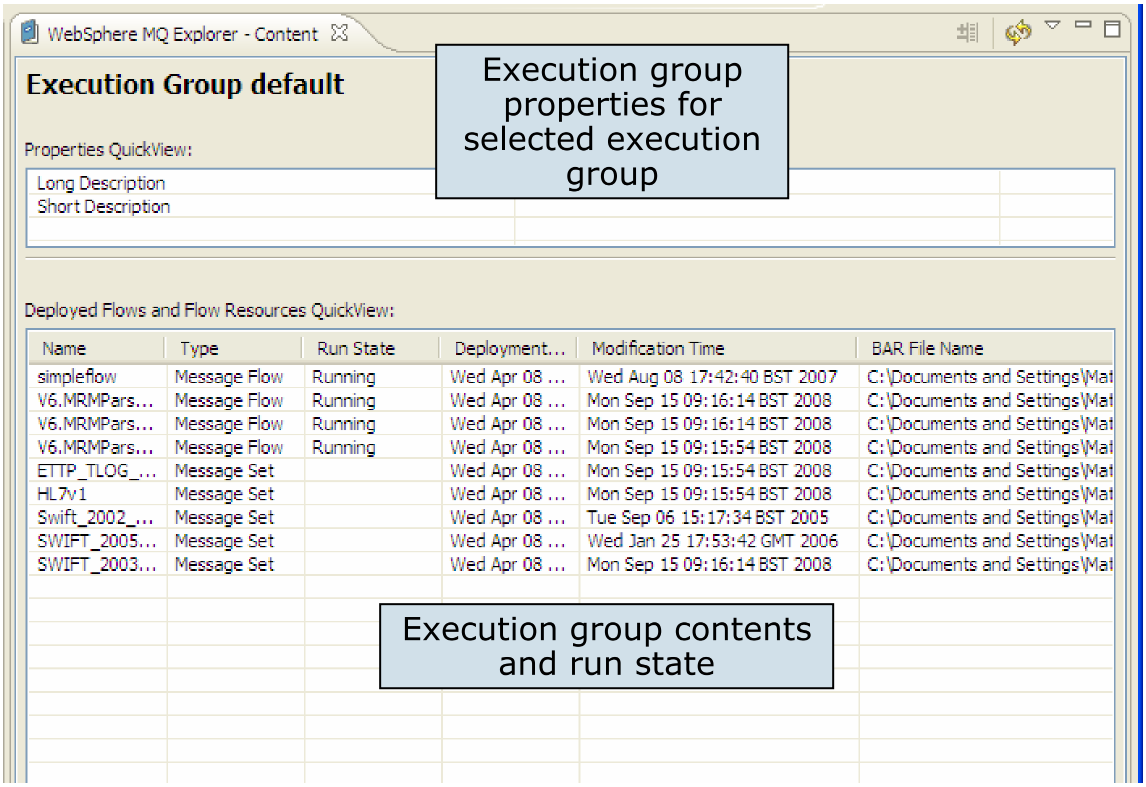
Details — Statistics, user trace, trace nodes, debug ports are described in a later unit.

Additional information —

Transition statement — Selecting any execution group in the Navigator displays the execution group Quick Views.

WebSphere Education

Execution group management (2 of 2)



WebSphere MQ Explorer - Content

Execution Group default

Properties QuickView:

Long Description		
Short Description		

Deployed Flows and Flow Resources QuickView:

Name	Type	Run State	Deployment...	Modification Time	BAR File Name
simpleflow	Message Flow	Running	Wed Apr 08 ...	Wed Aug 08 17:42:40 BST 2007	C:\Documents and Settings\Mat
V6.MRMPars...	Message Flow	Running	Wed Apr 08 ...	Mon Sep 15 09:16:14 BST 2008	C:\Documents and Settings\Mat
V6.MRMPars...	Message Flow	Running	Wed Apr 08 ...	Mon Sep 15 09:16:14 BST 2008	C:\Documents and Settings\Mat
V6.MRMPars...	Message Flow	Running	Wed Apr 08 ...	Mon Sep 15 09:15:54 BST 2008	C:\Documents and Settings\Mat
ETTP_TLOG_...	Message Set		Wed Apr 08 ...	Mon Sep 15 09:15:54 BST 2008	C:\Documents and Settings\Mat
HL7v1	Message Set		Wed Apr 08 ...	Mon Sep 15 09:15:54 BST 2008	C:\Documents and Settings\Mat
Swift_2002_...	Message Set		Wed Apr 08 ...	Tue Sep 06 15:17:34 BST 2005	C:\Documents and Settings\Mat
SWIFT_2005...	Message Set		Wed Apr 08 ...	Wed Jan 25 17:53:42 GMT 2006	C:\Documents and Settings\Mat
SWIFT_2003...	Message Set		Wed Apr 08 ...	Mon Sep 15 09:16:14 BST 2008	C:\Documents and Settings\Mat

Execution group properties for selected execution group

Execution group contents and run state

© Copyright IBM Corporation 2010

Figure 4-31. Execution group management (2 of 2)

WM643 / VM6431.0

Notes:

Selecting an execution group in the Navigator displays the execution group Quick Views; that shows the execution group properties, execution group contents, and run state.

The execution group **Properties QuickView** includes a list of the queues that are used and flow debug port.

The **Deployed Flows and Flow Resources QuickView** lists the contents of the execution group. If the content is a message flow, the QuickView also shows the current state of the message flow.

Similar to brokers, you can also modify some of the execution group properties by right-clicking the execution group in the Navigator view and selecting **Properties** from the menu.

Instructor notes:

Purpose — Show the execution group Quick Views.

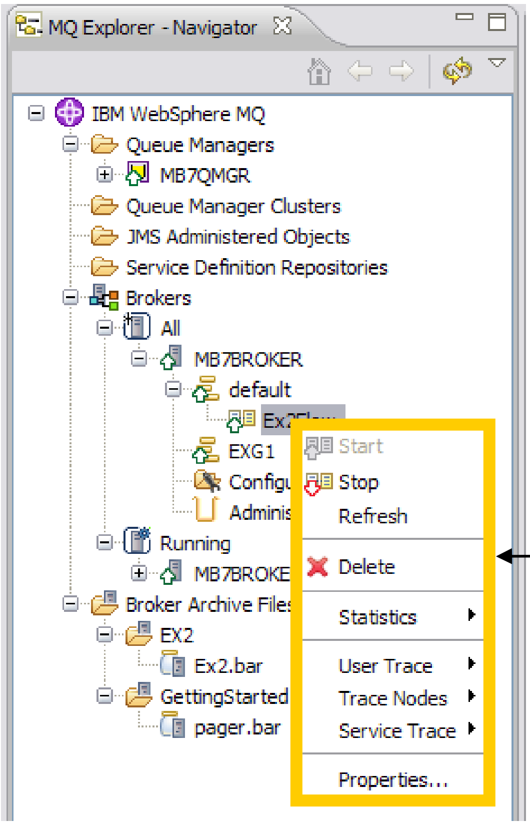
Details — Demonstrate the execution group Quick Views by selecting an execution group in the Navigator view. Also demonstrate that some of the properties can be modified by right-clicking the execution groups and selecting **Properties**.

Additional information —

Transition statement — At the lowest level, you also manage message flows.

WebSphere Education

Message flow management options



Control and configuration options

© Copyright IBM Corporation 2010

Figure 4-32. Message flow management options WM643 / VM6431.0

Notes:

Right-click on any message flow in the Navigator view to display the control and configuration options on a menu.

With WebSphere Message Broker Explorer, it is also possible to change message flow properties from an administrator console instead of in the BAR file by selecting **Properties** from the menu and making the modifications.

Instructor notes:

Purpose — Show the message flow management options in WebSphere Message Broker Explorer.

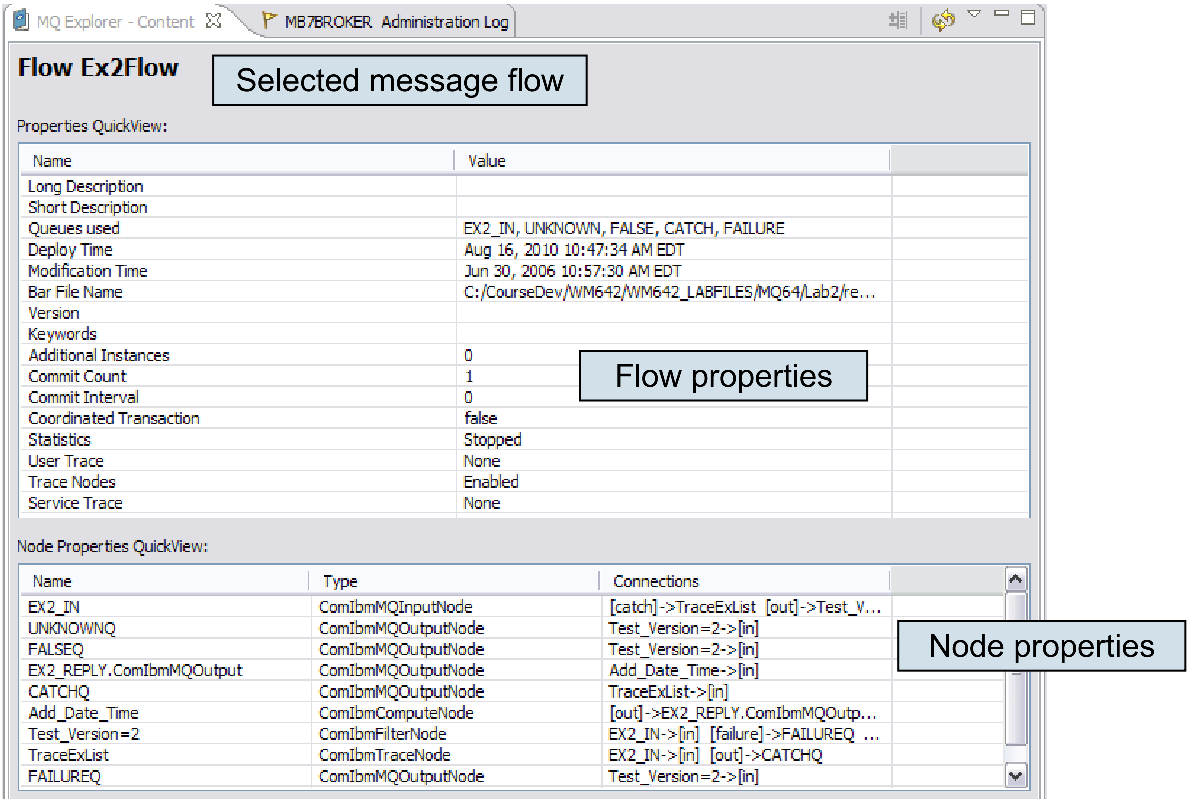
Details — Right-click a message flow and select Properties to show the message flow properties that can be modified from WebSphere Message Broker Explorer.

Additional information —

Transition statement — Selecting any message flow in the Navigator, displays the message flow Quick Views.

WebSphere Education

Message flow and node properties



Flow Ex2Flow

Selected message flow

Properties QuickView:

Name	Value
Long Description	
Short Description	
Queues used	EX2_IN, UNKNOWN, FALSE, CATCH, FAILURE
Deploy Time	Aug 16, 2010 10:47:34 AM EDT
Modification Time	Jun 30, 2006 10:57:30 AM EDT
Bar File Name	C:/CourseDev/WM642/WM642_LABFILES/MQ64/Lab2/re...
Version	
Keywords	
Additional Instances	0
Commit Count	1
Commit Interval	0
Coordinated Transaction	false
Statistics	Stopped
User Trace	None
Trace Nodes	Enabled
Service Trace	None

Flow properties

Node Properties QuickView:

Name	Type	Connections
EX2_IN	ComIbmMQInputNode	[catch]->TraceExList [out]->Test_V...
UNKNOWNQ	ComIbmMQOutputNode	Test_Version=2->[in]
FALSEQ	ComIbmMQOutputNode	Test_Version=2->[in]
EX2_REPLY.ComIbmMQOutput	ComIbmMQOutputNode	Add_Date_Time->[in]
CATCHQ	ComIbmMQOutputNode	TraceExList->[in]
Add_Date_Time	ComIbmComputeNode	[out]->EX2_REPLY.ComIbmMQOutp...
Test_Version=2	ComIbmFilterNode	EX2_IN->[in] [failure]->FAILUREQ ...
TraceExList	ComIbmTraceNode	EX2_IN->[in] [out]->CATCHQ
FAILUREQ	ComIbmMQOutputNode	Test_Version=2->[in]

Node properties

Figure 4-33. Message flow and node properties WM643 / VM6431.0

Notes:

The message flow Quick Views show the key flow properties, such as the BAR file name and commit count, additional instances value, and node properties, including the node names, type, and connections.

Instructor notes:

Purpose — Show the message flow Quick Views.

Details —

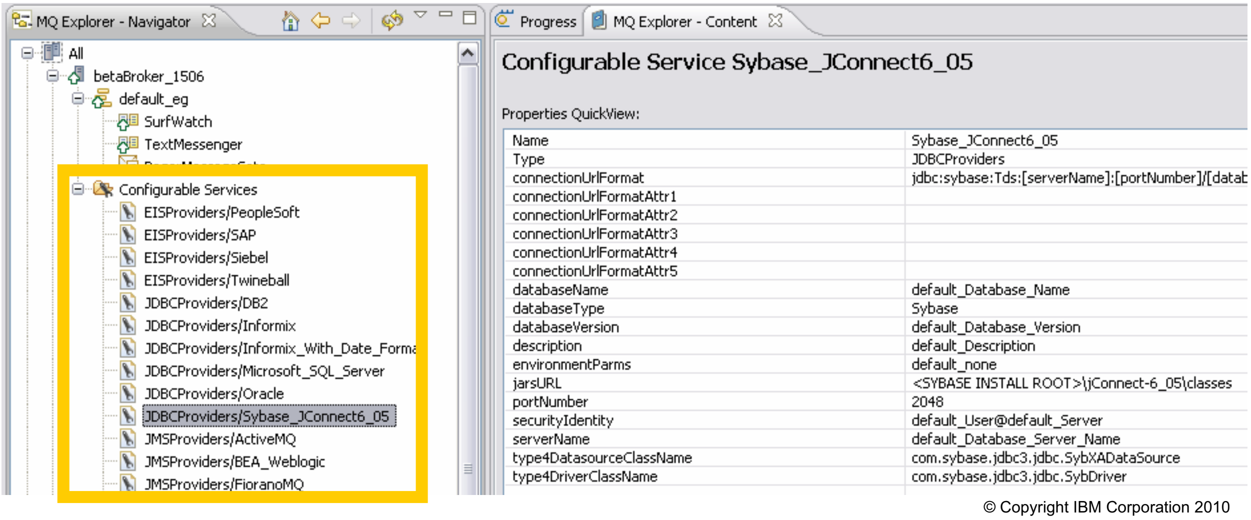
Additional information —

Transition statement — The WebSphere Message Broker Explorer can also be used to manage configurable services.

WebSphere Education

Configurable services

- Configurable services are typically runtime properties related to external services on which the broker relies
  - Use the Configurable Services wizard in WebSphere Message Broker Explorer to base a configurable service on an existing one
  - Import and export configurable services files to the broker from other brokers
  - Drag onto **Configurable Services** directory between brokers



All
betaBroker_1506
default_eg
SurfWatch
TextMessenger
Configurable Services
EISProviders/PeopleSoft
EISProviders/SAP
EISProviders/Siebel
EISProviders/Twineball
JDBCProviders/DB2
JDBCProviders/Informix
JDBCProviders/Informix_With_Date_Forma
JDBCProviders/Microsoft_SQL_Server
JDBCProviders/Oracle
JDBCProviders/Sybase_JConnect6_05
JMSProviders/ActiveMQ
JMSProviders/BEA_Weblogic
JMSProviders/FioranoMQ

Configurable Service Sybase_JConnect6_05

Properties QuickView:

Name	Sybase_JConnect6_05
Type	JDBCProviders
connectionUrlFormat	jdbc:sybase:Tds:[serverName]:[portNumber]/[datat
connectionUrlFormatAttr1	
connectionUrlFormatAttr2	
connectionUrlFormatAttr3	
connectionUrlFormatAttr4	
connectionUrlFormatAttr5	
databaseName	default_Database_Name
databaseType	Sybase
databaseVersion	default_Database_Version
description	default_Description
environmentParms	default_none
jarsURL	<SYBASE INSTALL ROOT>\jConnect-6_05\classes
portNumber	2048
securityIdentity	default_User@default_Server
serverName	default_Database_Server_Name
type4DatasourceClassName	com.sybase.jdbc3.jdbc.SybXADataSource
type4DriverClassName	com.sybase.jdbc3.jdbc.SybDriver

Figure 4-34. Configurable services WM643 / VM6431.0

Notes:

Instead of defining properties on the node or message flow, you can create *configurable services* so that nodes and message flows can refer to them to find properties at run time. For example, you can use a configurable service to define properties that are related to a JMS provider.

WebSphere Message Broker Explorer provides an interface to create and manage configurable services. This makes it possible to import and export configurable services, and quickly update configurable services.

Some configurable services require a restart of execution group to take effect.

Instructor notes:

Purpose — Describe configurable services creation and management tools available in WebSphere Message Broker Explorer.

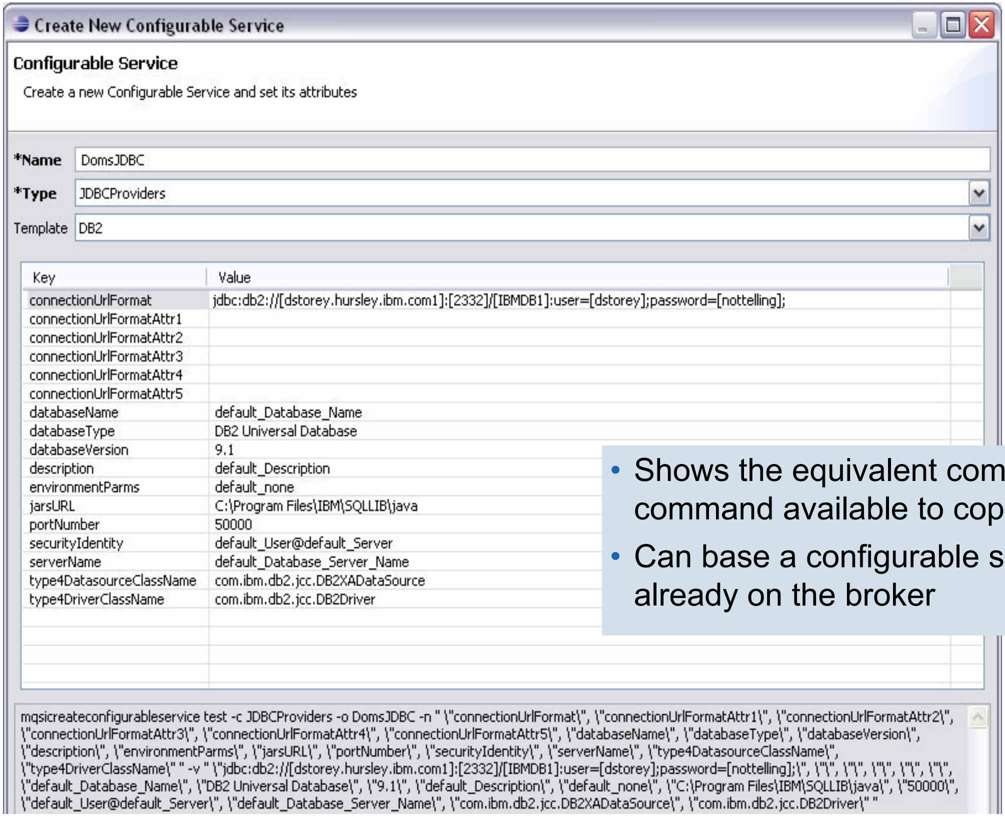
Details —

Additional information —

Transition statement — WebSphere Message Broker contains a number of configurable services that can be used as a template for creating new services.

WebSphere Education

Configurable Service wizard



Configurable Service

Create a new Configurable Service and set its attributes

*Name: DomsJDBC

*Type: JDBCProviders

Template: DB2

Key	Value
connectionUrlFormat	jdbc:db2://[dstorey.hursley.ibm.com1]:[2332]/[IBMDB1]:user=[dstorey];password=[nottelling];
connectionUrlFormatAttr1	
connectionUrlFormatAttr2	
connectionUrlFormatAttr3	
connectionUrlFormatAttr4	
connectionUrlFormatAttr5	
databaseName	default_Database_Name
databaseType	DB2 Universal Database
databaseVersion	9.1
description	default_Description
environmentParms	default_none
jarsURL	C:\Program Files\IBM\SQLLIB\java
portNumber	50000
securityIdentity	default_User@default_Server
serverName	default_Database_Server_Name
type4DatasourceClassName	com.ibm.db2.jcc.DB2XADataSource
type4DriverClassName	com.ibm.db2.jcc.DB2Driver

```
mqsicreateconfigurableservice test -c JDBCProviders -o DomsJDBC -n " \"connectionUrlFormat\", \"connectionUrlFormatAttr1\", \"connectionUrlFormatAttr2\",
\"connectionUrlFormatAttr3\", \"connectionUrlFormatAttr4\", \"connectionUrlFormatAttr5\", \"databaseName\", \"databaseType\", \"databaseVersion\",
\"description\", \"environmentParms\", \"jarsURL\", \"portNumber\", \"securityIdentity\", \"serverName\", \"type4DatasourceClassName\",
\"type4DriverClassName\" " -v " \"jdbc:db2://[dstorey.hursley.ibm.com1]:[2332]/[IBMDB1]:user=[dstorey];password=[nottelling];\", \"\", \"\", \"\", \"\", \"\",
\"default_Database_Name\", \"DB2 Universal Database\", \"9.1\", \"default_Description\", \"default_none\", \"C:\Program Files\IBM\SQLLIB\java\", \"50000\",
\"default_User@default_Server\", \"default_Database_Server_Name\", \"com.ibm.db2.jcc.DB2XADataSource\", \"com.ibm.db2.jcc.DB2Driver\" "
```

- Shows the equivalent command-line command available to copy and paste
- Can base a configurable service on one already on the broker

Figure 4-35. Configurable Service wizard WM643 / VM6431.0

Notes:

It is possible to use a configurable service as a template for another configurable service. This speeds implementation time; by starting with a base set of services, you only need to modify those values required for the new configurable service.

To create a new configurable service, right-click Configurable Services in the Navigator view and select **New > Configurable Service** from the menu.

In addition, the command line describing the configurable service as configured in Configurable Services wizard can be copied and pasted for command-line execution.

Alternatively, you can use the following commands to work with configurable services:

- Use the `mqsicreateconfigurableservice` command to create configurable services.
- Use the `mqsideleteconfigurableservice` command to delete configurable services.
- Use the `mqsichangeproperties` command to set attributes after you have created the configurable services.
- Use the `mqsireportproperties` command to report attributes.

Instructor notes:

Purpose — Show the Configurable Service wizard for creating a new configurable service.

Details — Expand **Configurable Services** in the WebSphere Message Broker Explorer Navigator view and show students some of the configurable services that are provided with WebSphere Message Broker.

Additional information —

Transition statement — WebSphere Message Broker Explorer includes an option to view pending and submitted tasks.

WebSphere Education

Administration Queue

- Shows pending and submitted tasks that have been sent to the broker, and are waiting to be processed
- Use WebSphere Message Broker Explorer to view task properties, and to cancel tasks in the queue
  - Order
  - Status
  - Username
  - Operation Type
  - Object Name
  - Object Type
  - Creation Time
  - Elapsed Time
  - Identifier

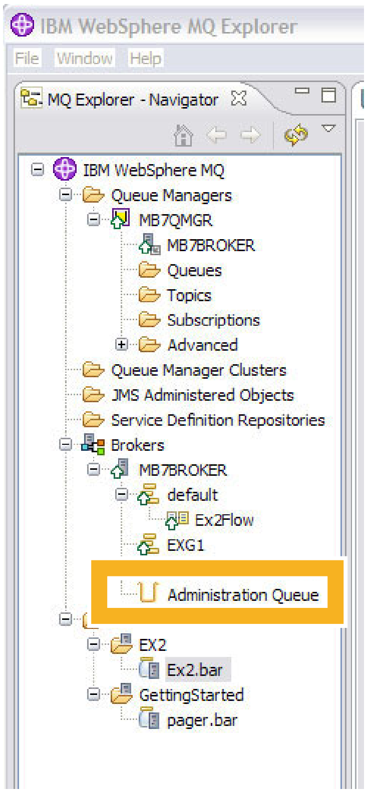


Figure 4-36. Administration Queue WM643 / VM6431.0

Notes:

Each broker that is connected in the WebSphere Message Broker Explorer displays an icon labeled **Administration Queue**.

When you click the **Administration Queue** icon, tasks submitted to the broker are displayed in the Administration Queue QuickView in the Content pane. The Administration Queue icon changes to indicate the number of messages on the Administration Queue.

You can remove tasks from the Administration Queue if you want to cancel requests that have been sent to a broker. To remove tasks from the Administration Queue:

1. In the Navigator view, expand the **Brokers** folder.
2. Expand the broker with which you want to work, and right-click **Administration Queue**.
3. Click **Cancel Work Items**. The Administration Queue window is displayed.
4. Select the task or tasks that you want to remove from the Administration Queue, and click **Cancel Work Items**.

Instructor notes:

Purpose — Introduce the Administration Queue.

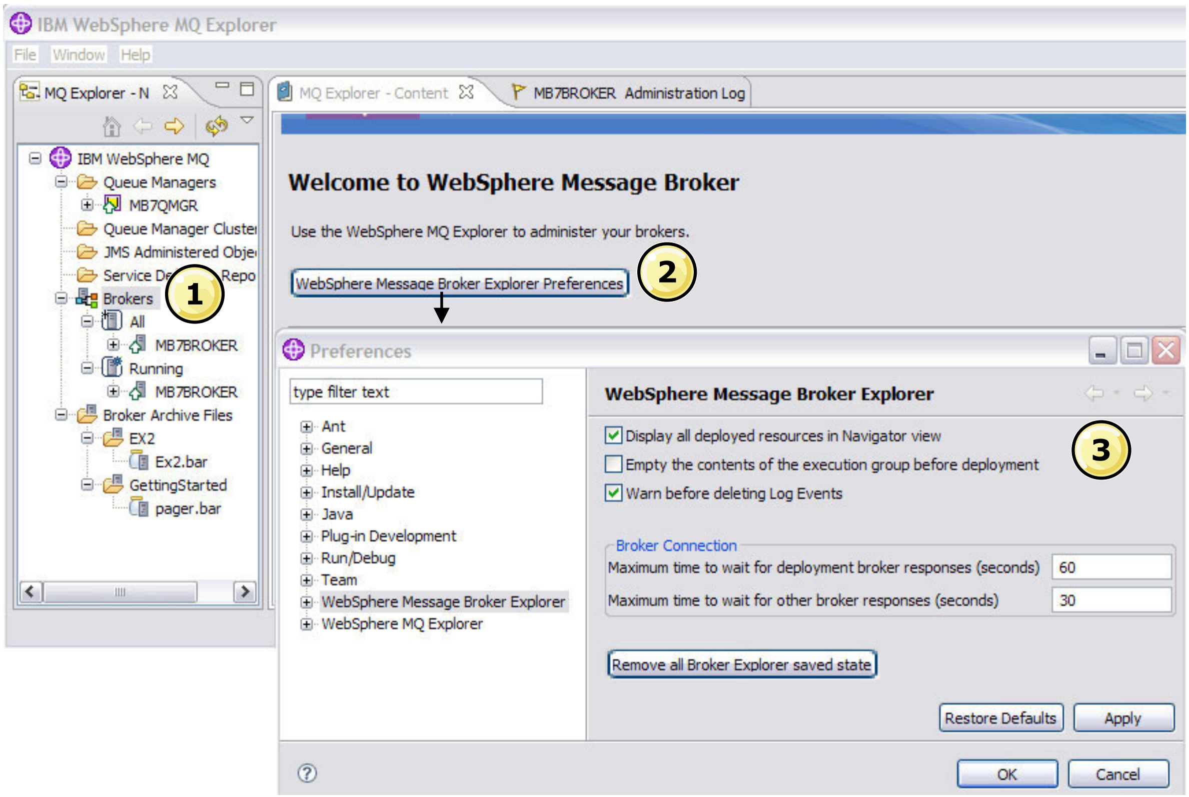
Details — None.

Additional information — None.

Transition statement — WebSphere Message Broker has a number of preferences

WebSphere Education

WebSphere Message Broker Explorer preferences



Welcome to WebSphere Message Broker

Use the WebSphere MQ Explorer to administer your brokers.

WebSphere Message Broker Explorer

☑ Display all deployed resources in Navigator view

□ Empty the contents of the execution group before deployment

☑ Warn before deleting Log Events

Broker Connection

Maximum time to wait for deployment broker responses (seconds) 60

Maximum time to wait for other broker responses (seconds) 30

© Copyright IBM Corporation 2010

Figure 4-37. WebSphere Message Broker Explorer preferences WM643 / VM6431.0

Notes:

You can change the following preferences to tailor how you work in the WebSphere Message Broker Explorer:

- Display deployed resources in the Navigator view.
- Empty the contents of the execution group before deployment
- Warn before deleting log events
- Broker connection wait times
- DataPower configuration properties
- Service trace properties
- SSL parameters

To change WebSphere Message Broker Explorer preferences:

1. Select the Brokers folder in the Navigator view.
2. Click **WebSphere Message Explorer Broker Preferences**.
3. Select the preferences, click **Apply**, and then click **OK**.

Instructor notes:

Purpose — Change preferences in WebSphere Message Broker Explorer,

Details — The Welcome page also has options for creating and removing the default configuration.

Additional information —

Transition statement — You can also configure and manage the broker using the Administration API.

4.3. Administration API (CMP API) overview

Instructor topic introduction

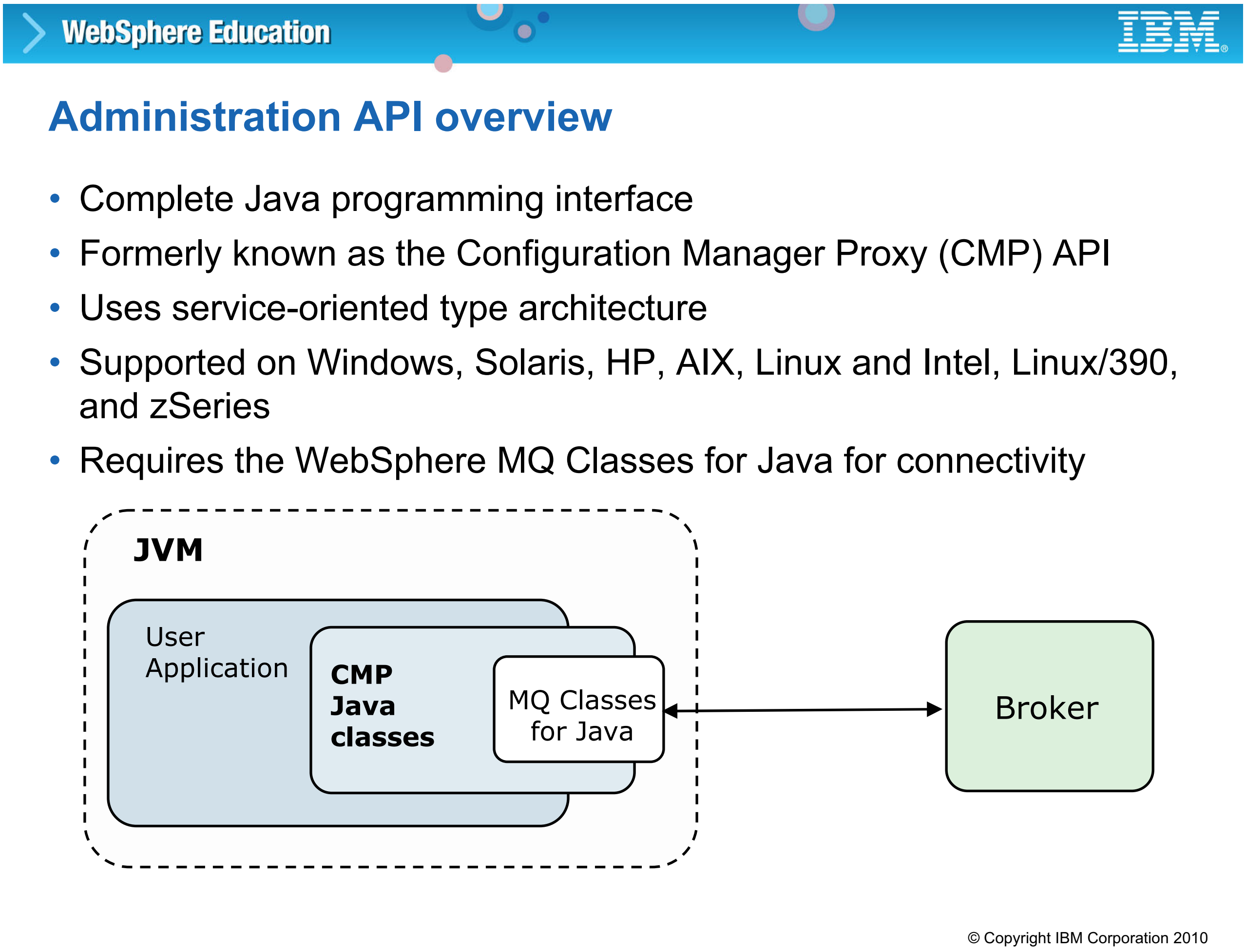
What students will do — Learn about the Administration API.

There will be some simple examples of the Administration API, but no programming skills are expected to be used during class.

How students will do it — Listen to the lecture and participate in classroom discussion. In the accompanying exercise (Exercise 3), students will use the Configuration Manager Proxy API Exerciser, a sample shipped with WebSphere Message Broker.

What students will learn — Students will be exposed to the Administration API and have an opportunity to consider how they might use the interface.

How this will help students in their job — Students will be able to make recommendations for use of the Administration API in their own environment.



WebSphere Education

Administration API overview

- Complete Java programming interface
- Formerly known as the Configuration Manager Proxy (CMP) API
- Uses service-oriented type architecture
- Supported on Windows, Solaris, HP, AIX, Linux and Intel, Linux/390, and zSeries
- Requires the WebSphere MQ Classes for Java for connectivity

Figure 4-38. Administration API overview WM643 / VM6431.0

Notes:

The Administration (CMP) API for WebSphere Message Broker is a programming interface that your applications can use to control brokers and their resources through a remote interface.

As shown in the figure, the CMP Java classes sit logically between the user application and the broker, inside the Java Virtual Machine (JVM) of the user application. The API requires the WebSphere MQ Classes for Java for connectivity.

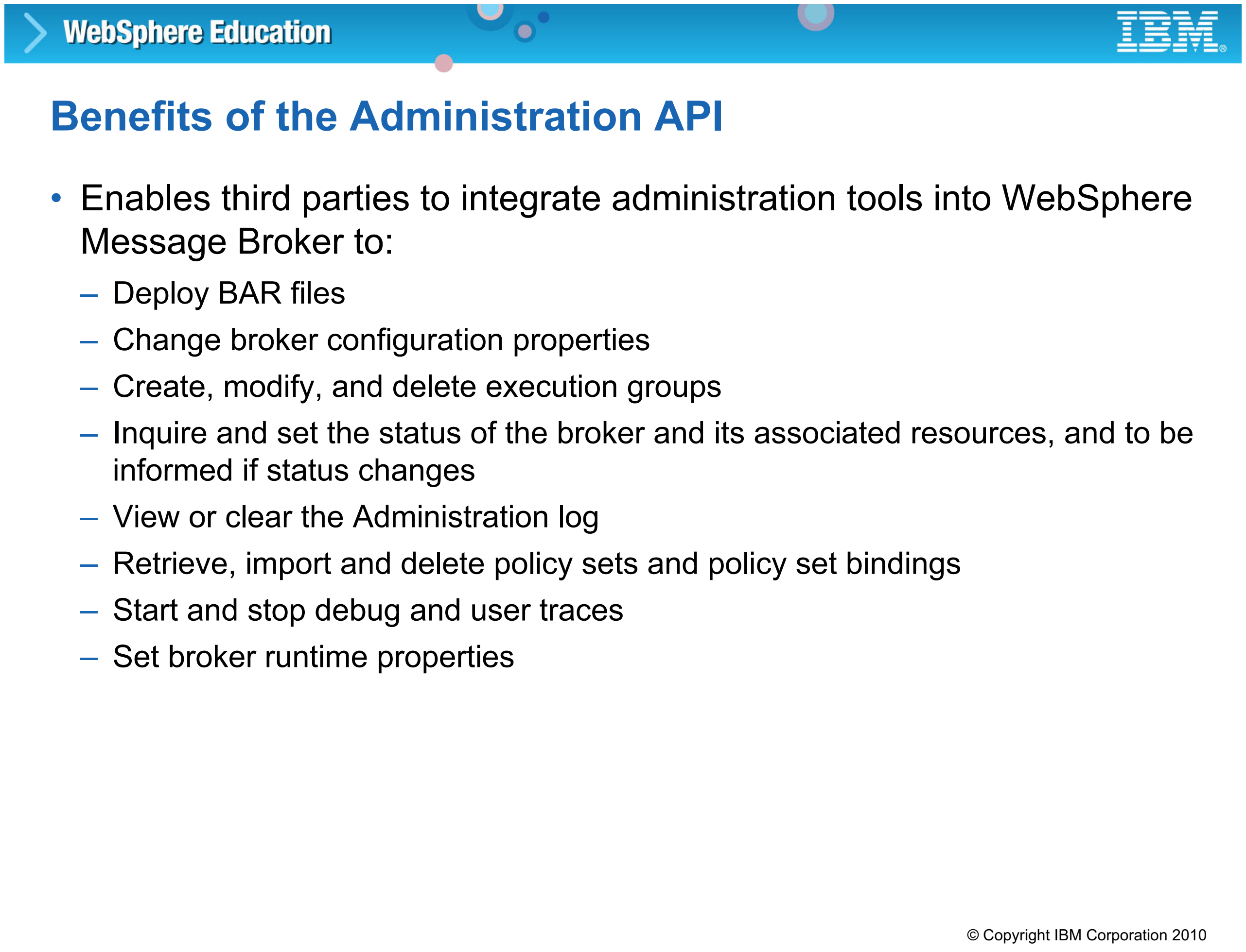
Instructor notes:

Purpose — Introduce the CMP.

Details — There is no need to discuss the API in any detail other than to introduce it, since this class is for administrators.

Additional information — None.

Transition statement — Why did IBM create the Administration API?



WebSphere Education
IBM

Benefits of the Administration API

- Enables third parties to integrate administration tools into WebSphere Message Broker to:
  - Deploy BAR files
  - Change broker configuration properties
  - Create, modify, and delete execution groups
  - Inquire and set the status of the broker and its associated resources, and to be informed if status changes
  - View or clear the Administration log
  - Retrieve, import and delete policy sets and policy set bindings
  - Start and stop debug and user traces
  - Set broker runtime properties

Figure 4-39. Benefits of the Administration API WM643 / VM6431.0

Notes:

The Administration API provides an interface that can be used to manipulate the broker and broker resources allowing users to write functions to perform the various tasks.

Your applications have complete access to the broker functions and resources through the set of Java classes that constitute the Administration API.

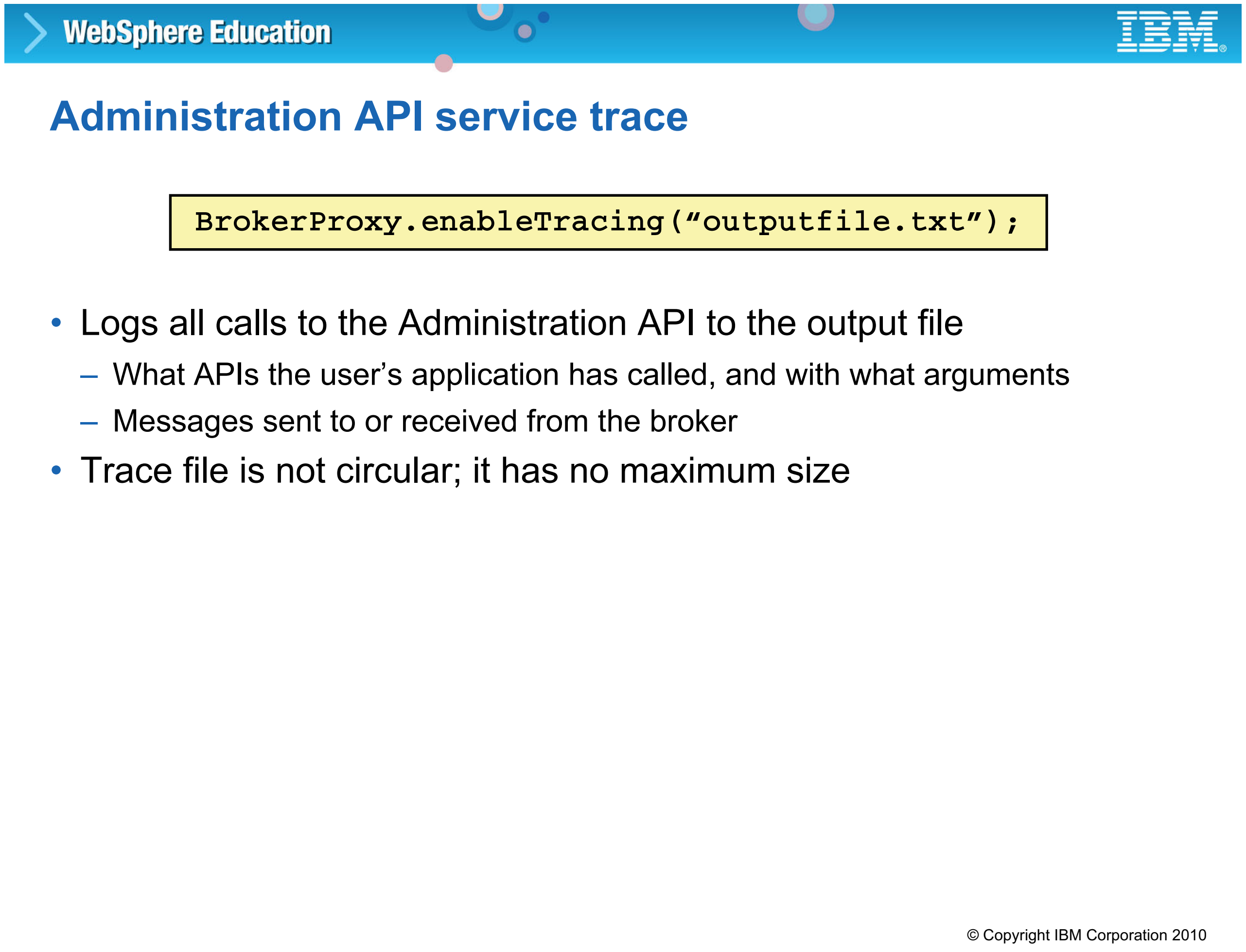
Instructor notes:

Purpose — Offer an explanation of the reason for the Administration API.

Details — There is not much here but a quick answer if someone asks “How come?”.

Additional information — None.

Transition statement — During development or troubleshooting it might be necessary to generate a service trace.



WebSphere Education

Administration API service trace

```
BrokerProxy.enableTracing(“outputfile.txt”);
```

- Logs all calls to the Administration API to the output file
  - What APIs the user’s application has called, and with what arguments
  - Messages sent to or received from the broker
- Trace file is not circular; it has no maximum size

Figure 4-40. Administration API service trace

WM643 / VM6431.0

Notes:

Use Administration API tracing to detail the API calls used, the arguments passed to the API, and the messages going between the API and the broker, which could potentially expose any exceptions.

The description of this method can be found in the BrokerProxy class.

To enable tracing for the Administration API for your application, use the following command in your code:

```
BrokerProxy.enableTracing("outputfile.txt");
```

This request logs all calls to the CMP API to the outputfile.txt file in the current directory. All CMP API activity in the entire Java™ Virtual Machine is logged.

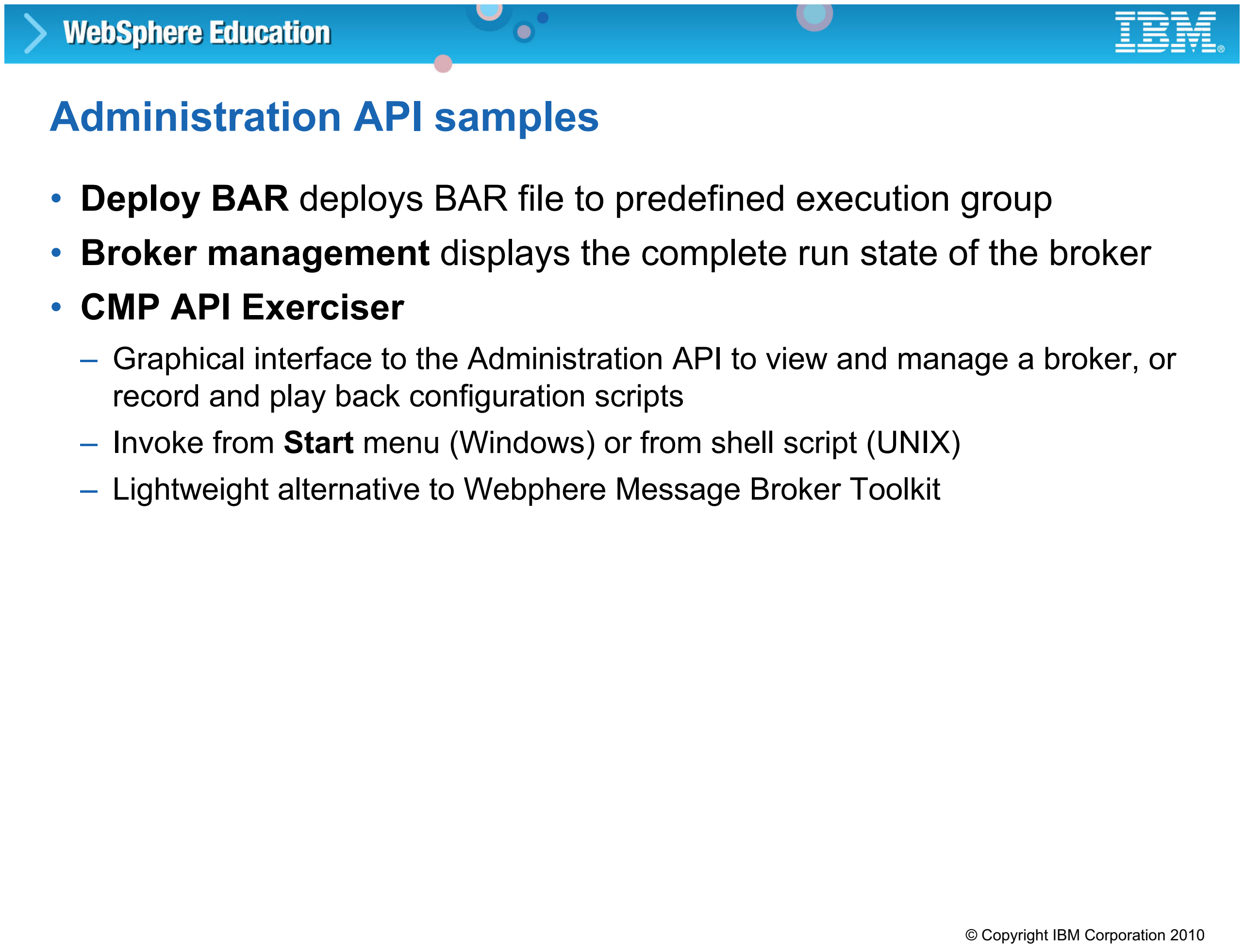
Instructor notes:

Purpose — Talk about tracing the Administration API.

Details — This is a relatively straightforward method in the BrokerProxy class. It might require a change in the application to activate and deactivate the trace or some means of passing a parameter.

Additional information — You can also disable CMP API service trace from the File menu of the CMP API Exerciser.

Transition statement — None.



WebSphere Education

Administration API samples

- **Deploy BAR** deploys BAR file to predefined execution group
- **Broker management** displays the complete run state of the broker
- **CMP API Exerciser**
  - Graphical interface to the Administration API to view and manage a broker, or record and play back configuration scripts
  - Invoke from **Start** menu (Windows) or from shell script (UNIX)
  - Lightweight alternative to Webphere Message Broker Toolkit

Figure 4-41. Application API samples WM643 / VM6431.0

Notes:

Explore the samples to learn the basic features that are provided by the Administration API.

The listed samples are available for use with the product and are documented in the *WebSphere Message Broker Application API Programming Guide.* They can also serve as a basis for applications you may want to develop to automate more of your administration work.

Instructor notes:

Purpose — List the supplied samples.

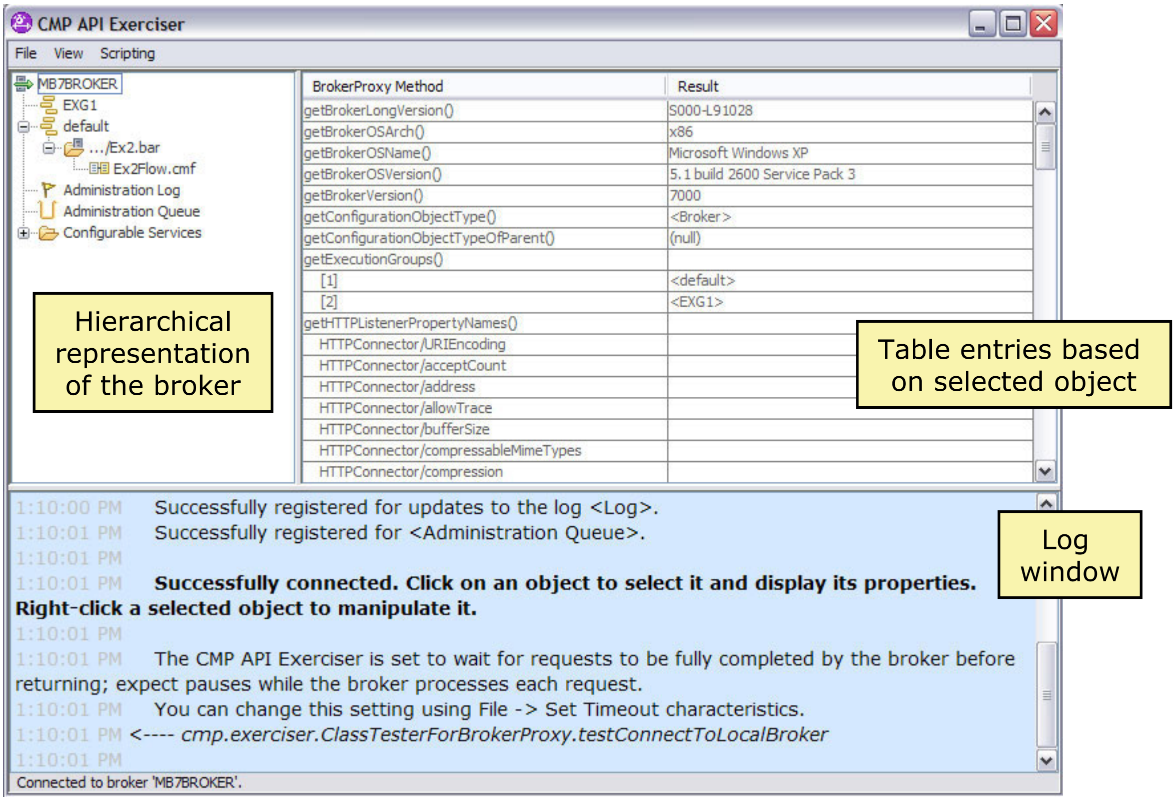
Details — Students get to use the CMP API Exerciser in the labs.

Additional information — None.

Transition statement — One of the samples is the CMP API Exerciser. You can use the CMP API Exerciser to view and manipulate brokers.

WebSphere Education

CMP API Exerciser



BrokerProxy Method	Result
getBrokerLongVersion()	S000-L91028
getBrokerOSArch()	x86
getBrokerOSName()	Microsoft Windows XP
getBrokerOSVersion()	5.1 build 2600 Service Pack 3
getBrokerVersion()	7000
getConfigurationObjectType()	<Broker>
getConfigurationObjectTypeOfParent()	(null)
getExecutionGroups()	
[1]	<default>
[2]	<EXG1>
getHTTPListenerPropertyNames()	
HTTPConnector/URIEncoding	
HTTPConnector/acceptCount	
HTTPConnector/address	
HTTPConnector/allowTrace	
HTTPConnector/bufferSize	
HTTPConnector/compressableMimeTypes	
HTTPConnector/compression	

```
1:10:00 PM    Successfully registered for updates to the log <Log>.
1:10:01 PM    Successfully registered for <Administration Queue>.
1:10:01 PM
1:10:01 PM    Successfully connected. Click on an object to select it and display its properties.
Right-click a selected object to manipulate it.
1:10:01 PM
1:10:01 PM    The CMP API Exerciser is set to wait for requests to be fully completed by the broker before
returning; expect pauses while the broker processes each request.
1:10:01 PM    You can change this setting using File -> Set Timeout characteristics.
1:10:01 PM <---- cmp.exerciser.ClassTesterForBrokerProxy.testConnectToLocalBroker
1:10:01 PM
```

Connected to broker 'MB7BROKER'.

Hierarchical representation of the broker

Table entries based on selected object

Log window

Figure 4-42. CMP API Exerciser

WM643 / VM6431.0

Notes:

You can use the CMP API Exerciser sample to view and manage a broker, or record and play back configuration scripts.

The upper left portion of the CMP API Exercise contains a hierarchical representation of the broker to which you are connected. Selecting objects in the tree causes the table on the right to change, reflecting the attributes of the object that you select.

The **Method** column lists Application API methods that you can call in your own Java applications.

The **Result** column indicates the data that is returned by calling the Application API method on the selected object.

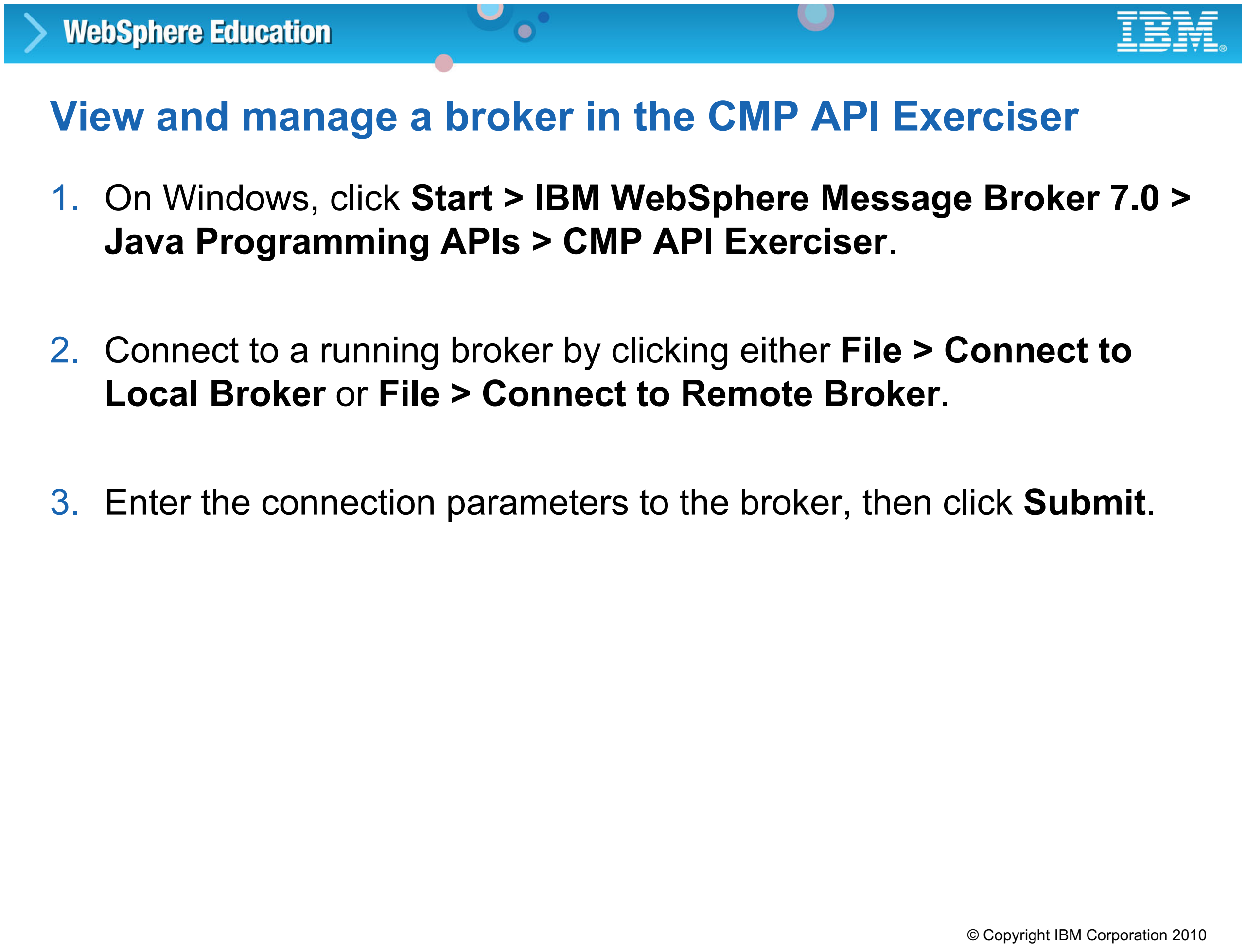
Instructor notes:

Purpose — Introduce the CMP API Exerciser

Details — Students use the CMP API Exercise in an exercise.

Additional information —

Transition statement — Next: View and manager a broker in the CMP API Exerciser



WebSphere Education

View and manage a broker in the CMP API Exerciser

1. On Windows, click **Start > IBM WebSphere Message Broker 7.0 > Java Programming APIs > CMP API Exerciser**.

2. Connect to a running broker by clicking either **File > Connect to Local Broker** or **File > Connect to Remote Broker**.

3. Enter the connection parameters to the broker, then click **Submit**.

Figure 4-43. View and manager a broker in the CMP API Exerciser WM643 / VM6431.0

Notes:

The figure lists the steps for starting the CMP API Exerciser on Windows.

To start the CMPI API Exerciser on other platforms:

1. Start a broker command environment by running `mqsiprofile`, or follow the guidance provided in the `StartConfigManagerProxyExerciser` shell script to configure the correct CLASSPATH for your environment.
2. Ensure that your user ID has writer permission to the current directory. The CMP API Exerciser stores its configuration settings in a file in this directory.
3. Run the shell script:

   `install_dir``\sample\ConfigManagerProxy\StartConfigManagerProxyExerciser`

   where *`install_dir`* is the Message Broker installation directory.

Instructor notes:

Purpose — List the steps for starting the CMP API Exerciser.

Details —

Additional information — None.

Transition statement — It is important to back up the broker and broker resources.

4.4. Backing up and restoring resources

Instructor topic introduction

What students will do — Students will learn how to plan for backup and recover of the broker resources as well as the Message Broker Toolkit.

How students will do it — Listen to lecture and participate in classroom discussion.

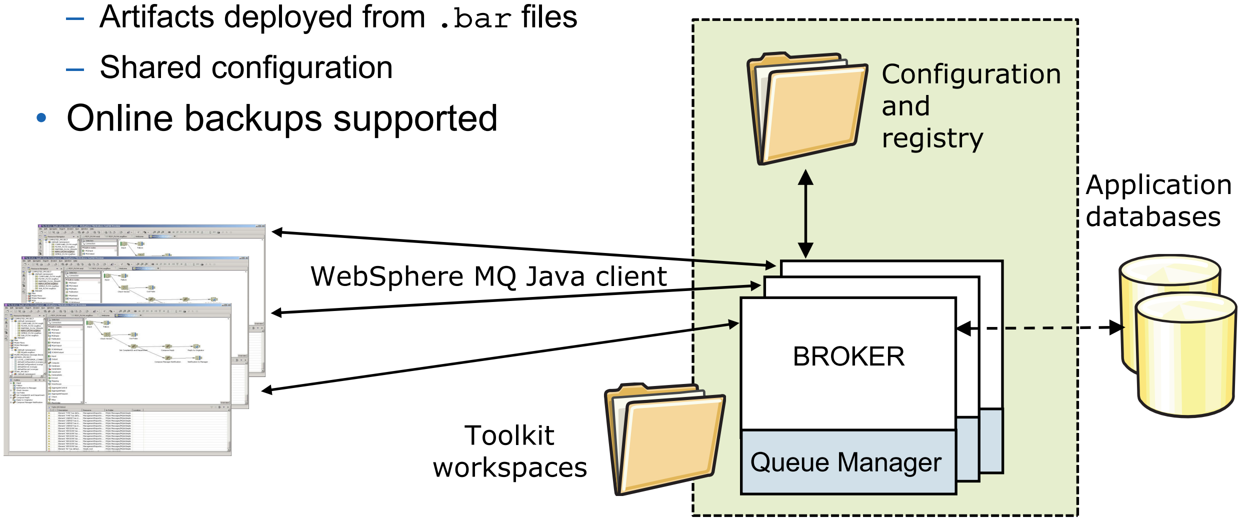
What students will learn — What steps must be taken to ensure all resources are backed up, as well as the steps necessary to complete a recovery scenario.

How this will help students on their job — Students will be able to backup and restore the Message Broker environment

WebSphere Education

Backup and restore

- Backup and restore commands for disaster recovery (operational state not saved)
- Backs up configuration state from the file system
  - Internally managed broker configuration store
  - Registry
  - Artifacts deployed from `.bar` files
  - Shared configuration
- Online backups supported



© Copyright IBM Corporation 2010

Figure 4-44. Backup and restore WM643 / VM6431.0

Notes:

Planning a backup and recovery strategy is an important administrative task.

WebSphere Message Broker has implemented two commands for disaster recovery: `mqsibackupbroker` and `mqsirestorebroker`

Each command requires the broker name and a location on disk for the archive file.

If it necessary to restore, you can restore directly from the backup file as long as the broker name and platform are the same as were used to create the backup

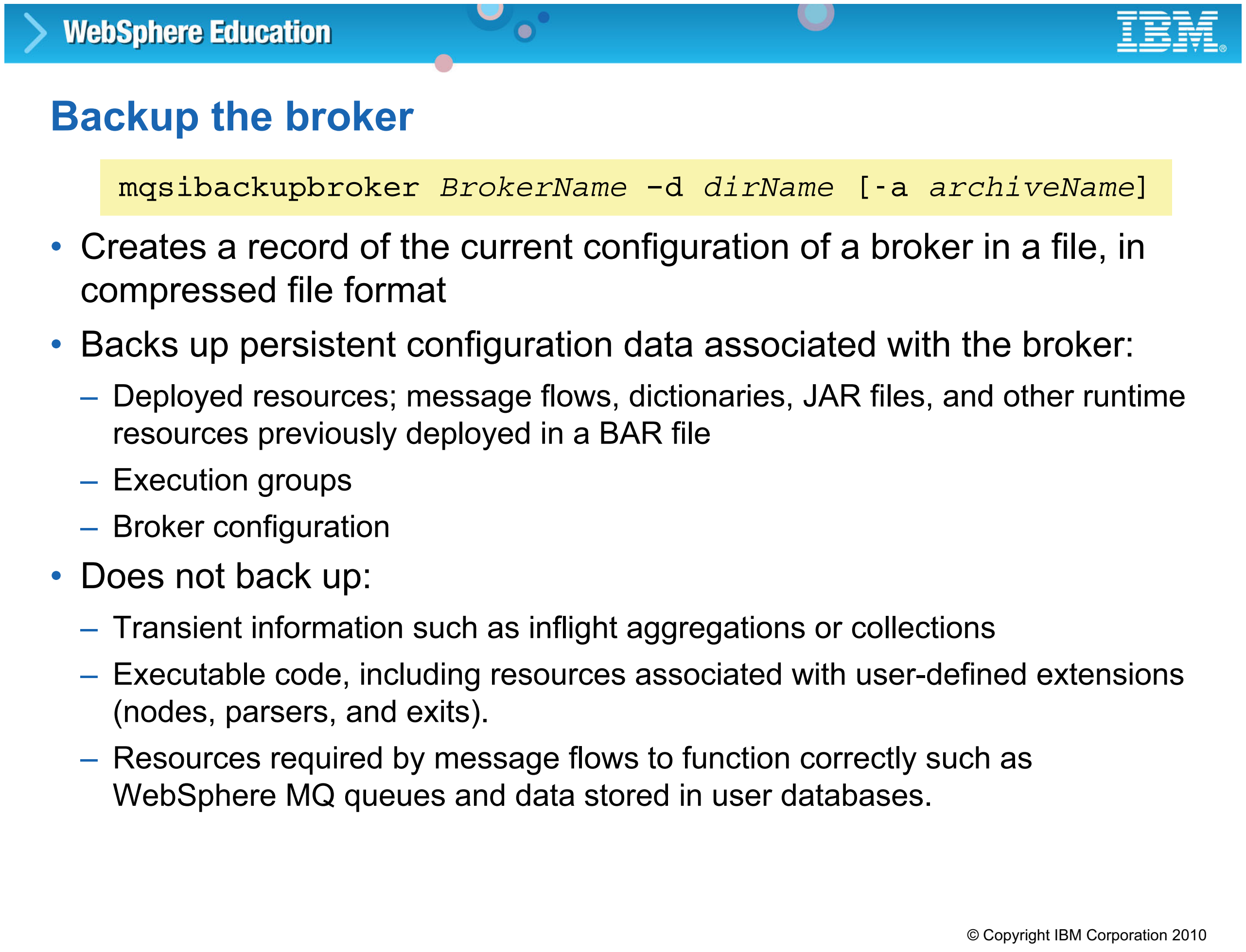
Instructor notes:

Purpose — Introduce what needs to be considered for backup and recovery.

Details — There will not be a great deal of time spent on setting up the WebSphere MQ and database backup and recovery strategy. These topics must be considered with the appropriate administration staff for those products.

Additional information —

Transition statement — Next: Backup the broker



WebSphere Education

Backup the broker

```
mqsibackupbroker BrokerName -d dirName [-a archiveName]
```

- Creates a record of the current configuration of a broker in a file, in compressed file format
- Backs up persistent configuration data associated with the broker:
  - Deployed resources; message flows, dictionaries, JAR files, and other runtime resources previously deployed in a BAR file
  - Execution groups
  - Broker configuration
- Does not back up:
  - Transient information such as inflight aggregations or collections
  - Executable code, including resources associated with user-defined extensions (nodes, parsers, and exits).
  - Resources required by message flows to function correctly such as WebSphere MQ queues and data stored in user databases.

Figure 4-45. Backup the broker

WM643 / VM6431.0

Notes:

The backup command creates a compressed file containing the internal configuration store, registry, and artifacts from the BAR file such as stylesheets. Backup can run while the broker is running.

The `mqsibackupbroker` command includes the following parameters:

`-d` is the required directory in which the backup file is created. You must specify a directory that is on a file system that can be accessed by the computer on which you run this command.

-a is the optional backup (archive) file name. The file is created in compressed file format. If you do not specify this parameter, the default name *`brokerName`*`_yyMMdd_HHmmss.zip` is used.

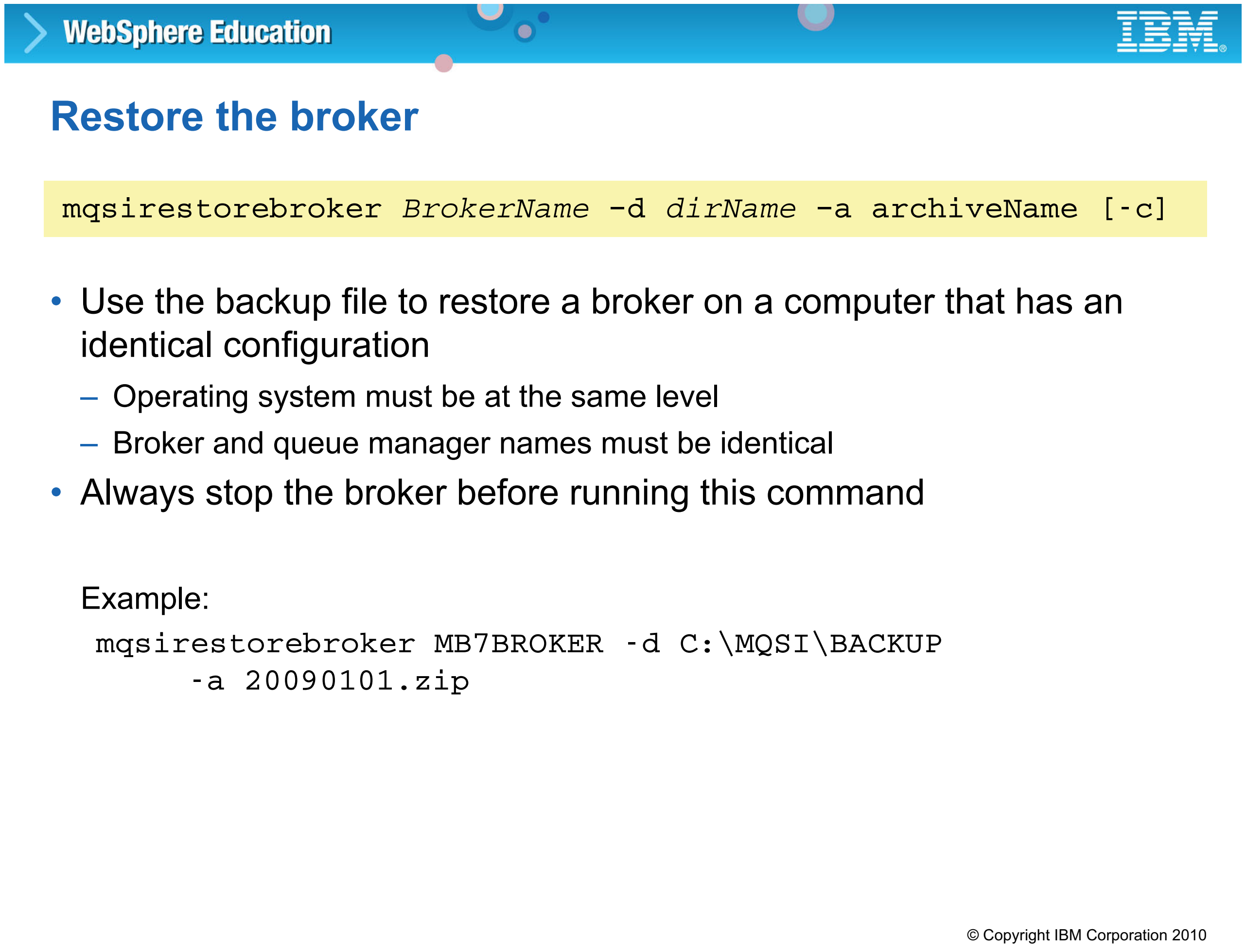
Instructor notes:

Purpose — Describe the syntax of the `mqsibackupbroker` command.

Details — No specific authority is required to run this command.

Additional information — None.

Transition statement — The next command allows you to restore a broker from a backup.



WebSphere Education

Restore the broker

```
mqsirestorebroker BrokerName -d dirName -a archiveName [-c]
```

- Use the backup file to restore a broker on a computer that has an identical configuration
  - Operating system must be at the same level
  - Broker and queue manager names must be identical
- Always stop the broker before running this command

Example:

```
mqsirestorebroker MB7BROKER -d C:\MQSI\BACKUP
     -a 20090101.zip
```

Figure 4-46. Restore the broker WM643 / VM6431.0

Notes:

If it necessary to restore, you can restore directly from the backup file as long as the broker name and platform are the same as were used to create the backup.

The `mqsirestorebroker` command includes the following parameters:

`-d` is the directory in which the backup file is stored. The file must be stored in a file system that can be accessed by the computer on which you run this command.

`-a` is the name of the backup (archive) file.

`-c` restores all configuration data that is shared with other brokers on the same computer; for example, profiles.

Always stop the broker before you run this command. If you specify `-c` to restore common configuration data that is shared with other brokers, you must also stop all brokers with which this broker shares that data. If you run this command when a relevant broker is active, the results are unpredictable.

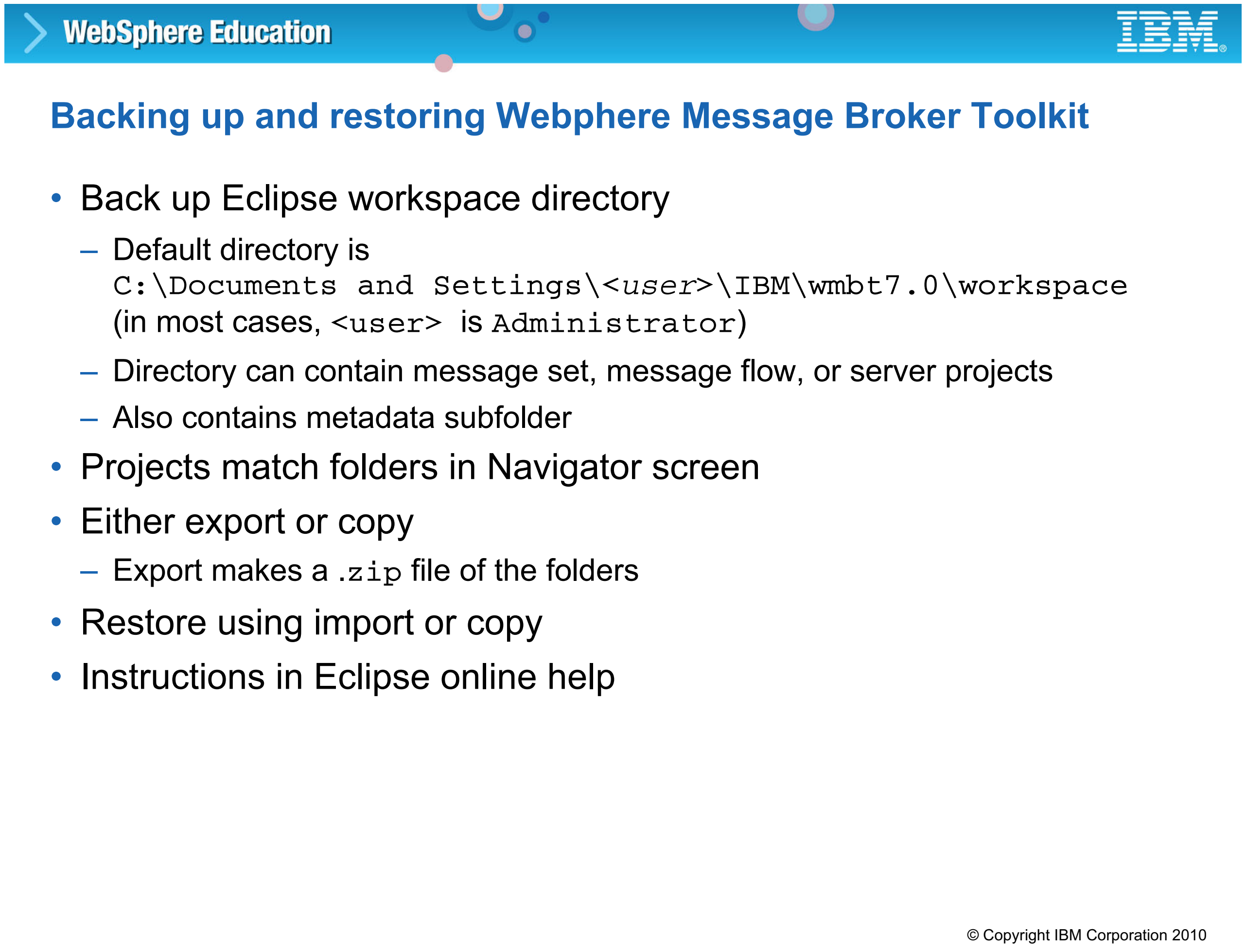
Instructor notes:

Purpose — Describe the syntax for `mqsirestorebroker` command.

Details — None.

Additional information — None.

Transition statement — You also must consider a strategy for backing up development artifacts.



WebSphere Education

Backing up and restoring Webphere Message Broker Toolkit

- Back up Eclipse workspace directory
  - Default directory is `C:\Documents and Settings\<user>\IBM\wmbt7.0\workspace` (in most cases, `<user>` is `Administrator`)
  - Directory can contain message set, message flow, or server projects
  - Also contains metadata subfolder
- Projects match folders in Navigator screen
- Either export or copy
  - Export makes a `.zip` file of the folders
- Restore using import or copy
- Instructions in Eclipse online help

Figure 4-47. Backing up and restoring Message Broker Toolkit WM643 / VM6431.0

Notes:

Since the toolkit is made up of many files, you must plan to back up your entire workspace directory (or directories). It is possible to export projects using the **File>Export** option or you can copy the files. The **Export** option generates a compressed file which is easier to manage.

The workspace contains a `.metadata` subfolder, which is essential because it contains all the settings, plug-ins, references to projects, and so on. Projects, however, are not necessarily subfolders in workspace directory. They can reside anywhere in the (local or shared) file system.

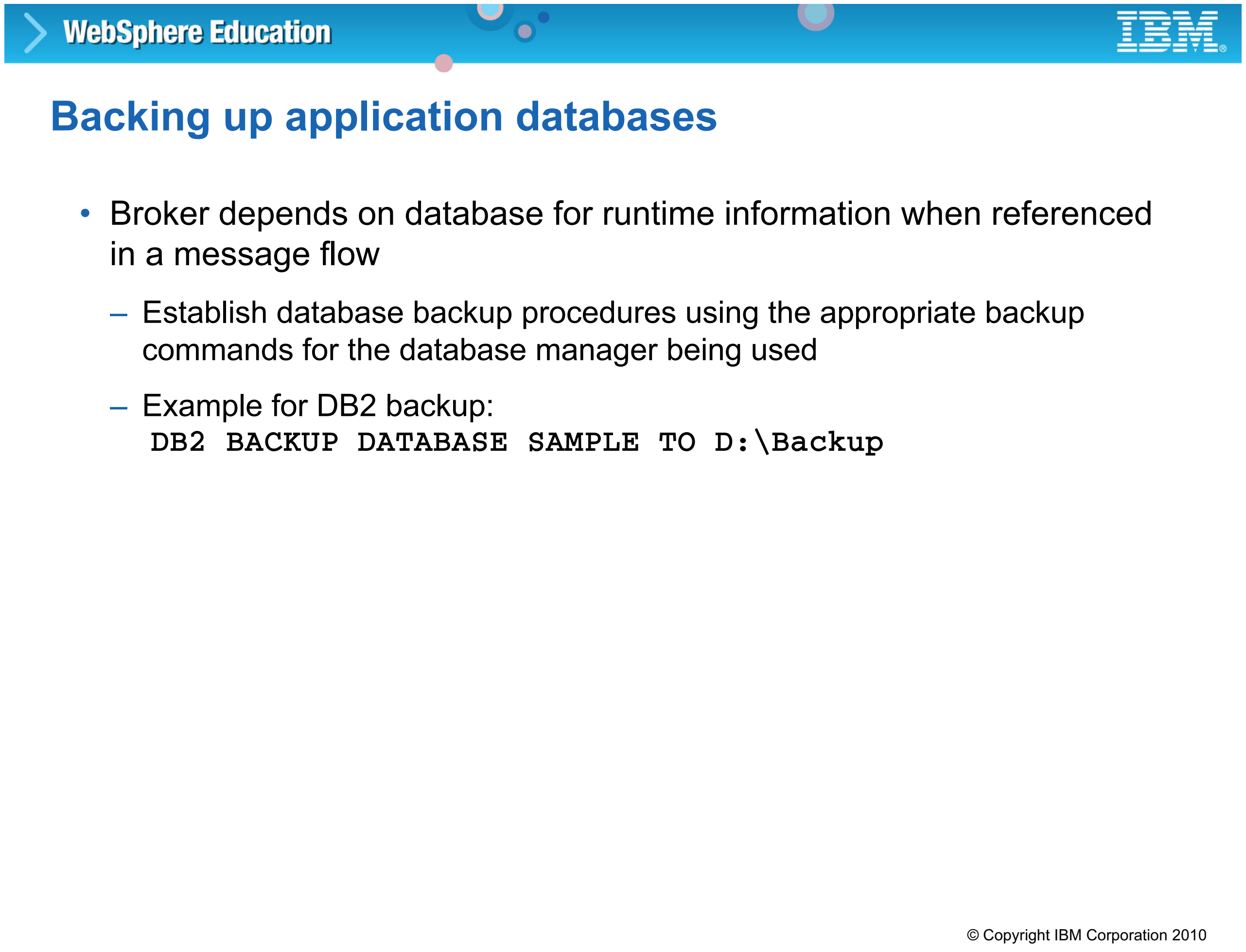
Instructor notes:

Purpose — Remind students about backing up their toolkit files

Details — Use of an external repository (version control) is useful in ensuring you have all the necessary resources for a development environment. However, sometimes resources are in development stages that have not yet caused them to be stored outside of a workspace

Additional information —

Transition statement — You should also back up any databases referenced by message flows running on the broker.



WebSphere Education

Backing up application databases

- Broker depends on database for runtime information when referenced in a message flow
  - Establish database backup procedures using the appropriate backup commands for the database manager being used
  - Example for DB2 backup:

```
DB2 BACKUP DATABASE SAMPLE TO D:\Backup
```

Figure 4-48. Backing up application databases

WM643 / VM6431.0

Notes:

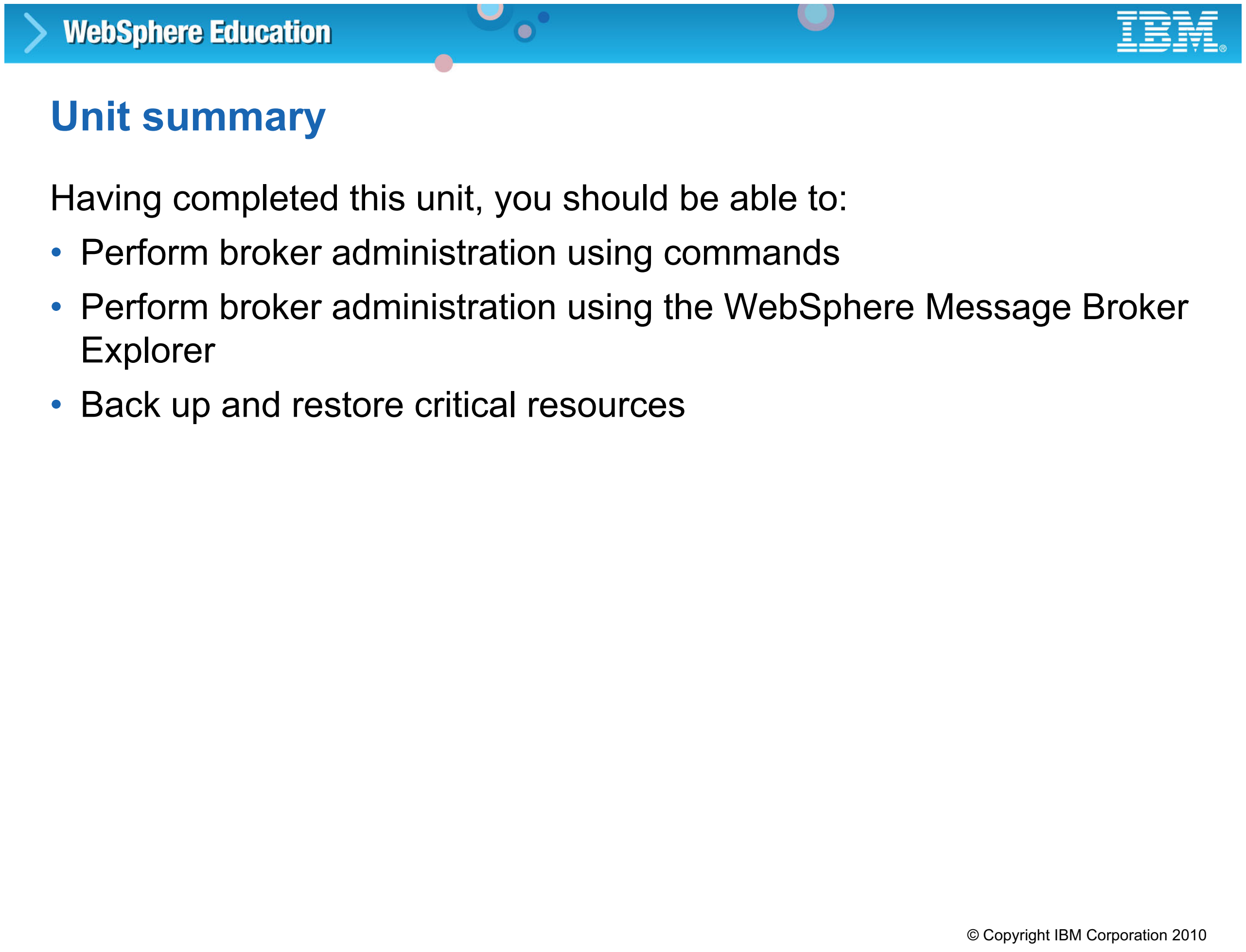
Instructor notes:

Purpose — Discuss what must be backed up for the brokers.

Details —

Additional information —

Transition statement — Next: Unit summary



WebSphere Education

Unit summary

Having completed this unit, you should be able to:

- Perform broker administration using commands
- Perform broker administration using the WebSphere Message Broker Explorer
- Back up and restore critical resources

Figure 4-49. Unit summary WM643 / VM6431.0

Notes:

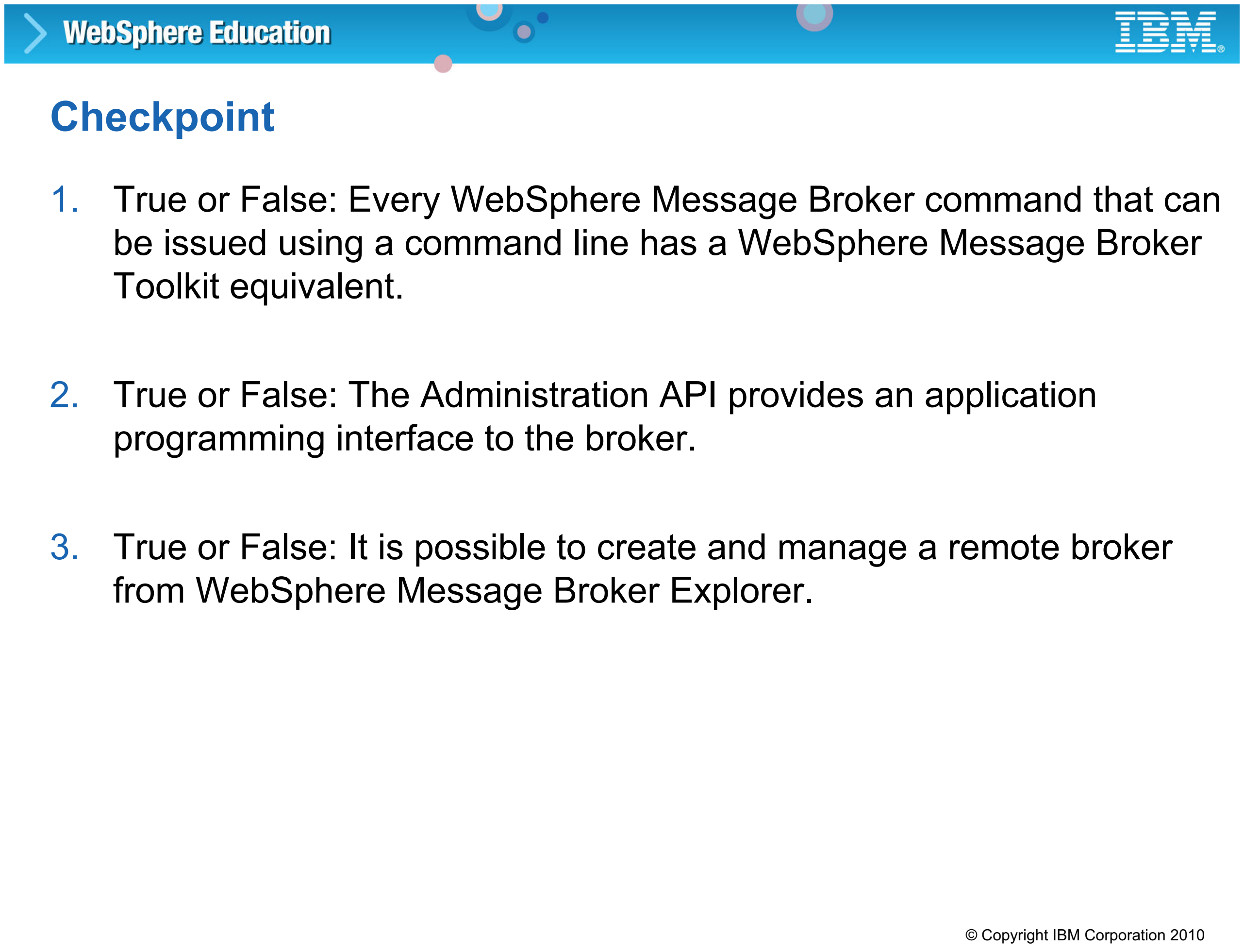
Instructor notes:

Purpose — Summarize the unit.

Details — None.

Additional information — None.

Transition statement — This concludes the administration unit. Now it is time to check your know with some checkpoint questions and an exercise.



WebSphere Education

Checkpoint

1. True or False: Every WebSphere Message Broker command that can be issued using a command line has a WebSphere Message Broker Toolkit equivalent.

2. True or False: The Administration API provides an application programming interface to the broker.

3. True or False: It is possible to create and manage a remote broker from WebSphere Message Broker Explorer.

Figure 4-50. Checkpoint WM643 / VM6431.0

Notes:

Write your answers here:

1.

2.

3.

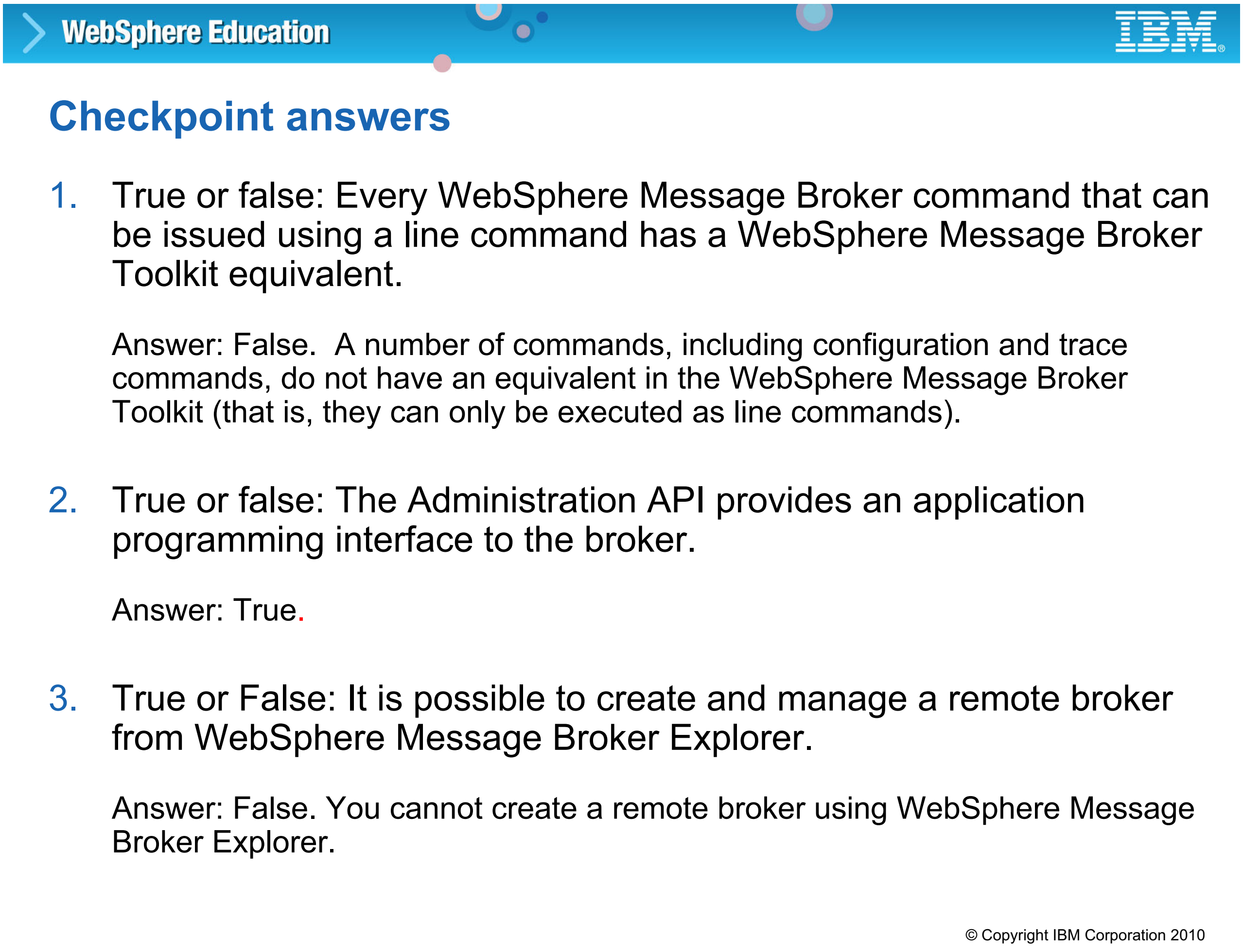
Instructor notes:

Purpose — Check student’s knowledge.

Details —

Additional information —

Transition statement — Next: Checkpoint answers



WebSphere Education

Checkpoint answers

1. True or false: Every WebSphere Message Broker command that can be issued using a line command has a WebSphere Message Broker Toolkit equivalent.

   Answer: False. A number of commands, including configuration and trace commands, do not have an equivalent in the WebSphere Message Broker Toolkit (that is, they can only be executed as line commands).

2. True or false: The Administration API provides an application programming interface to the broker.

   Answer: True.

3. True or False: It is possible to create and manage a remote broker from WebSphere Message Broker Explorer.

   Answer: False. You cannot create a remote broker using WebSphere Message Broker Explorer.

Figure 4-51. Checkpoint answers WM643 / VM6431.0

Notes:

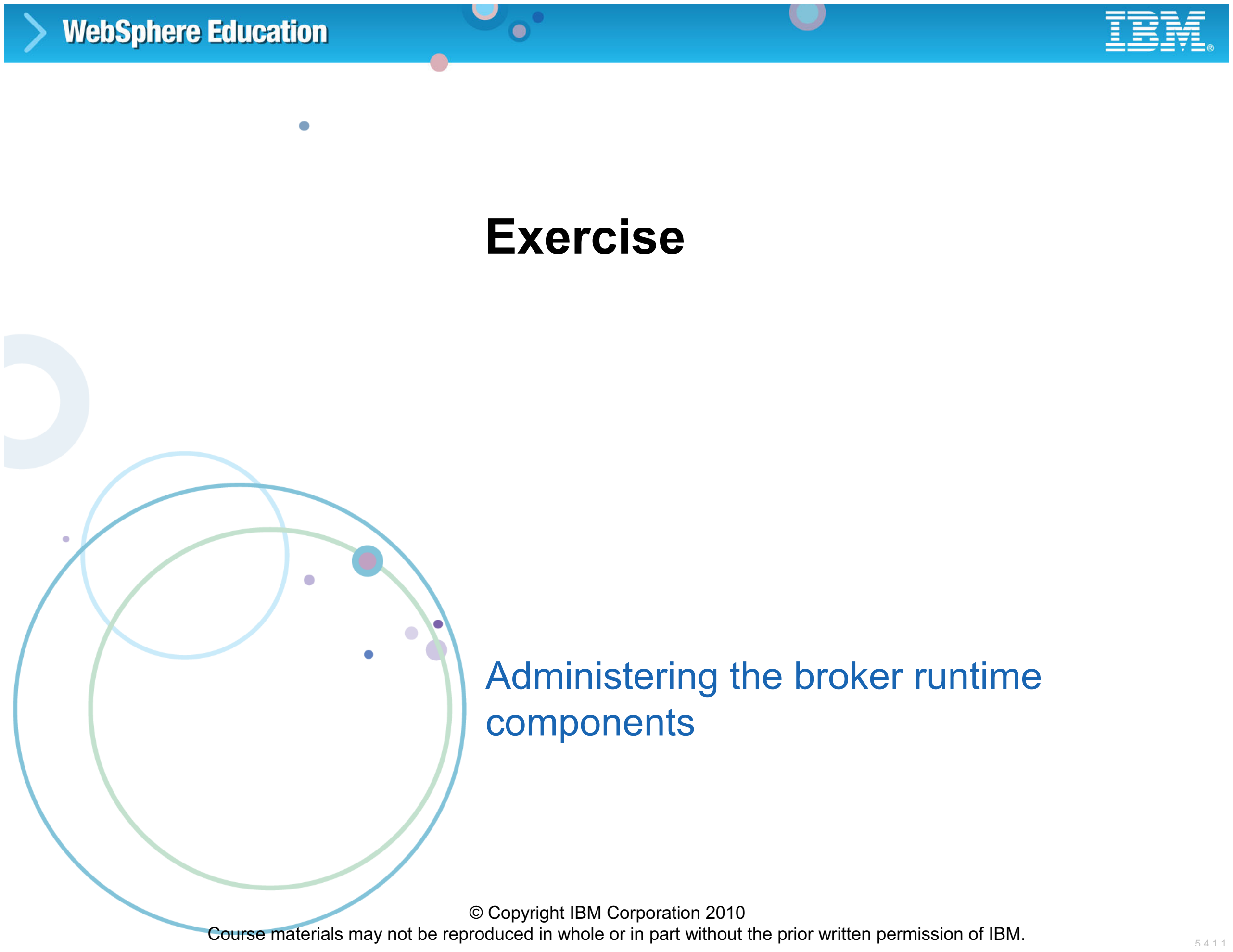
Instructor notes:

Purpose — Show checkpoint answers.

Details —

Additional information —

Transition statement — Next: Exercise 3: Administering the broker runtime components



WebSphere Education
IBM

Exercise

Administering the broker runtime components

5411

Figure 4-52. Exercise 3: Administering the broker runtime components WM643 / VM6431.0

Notes:

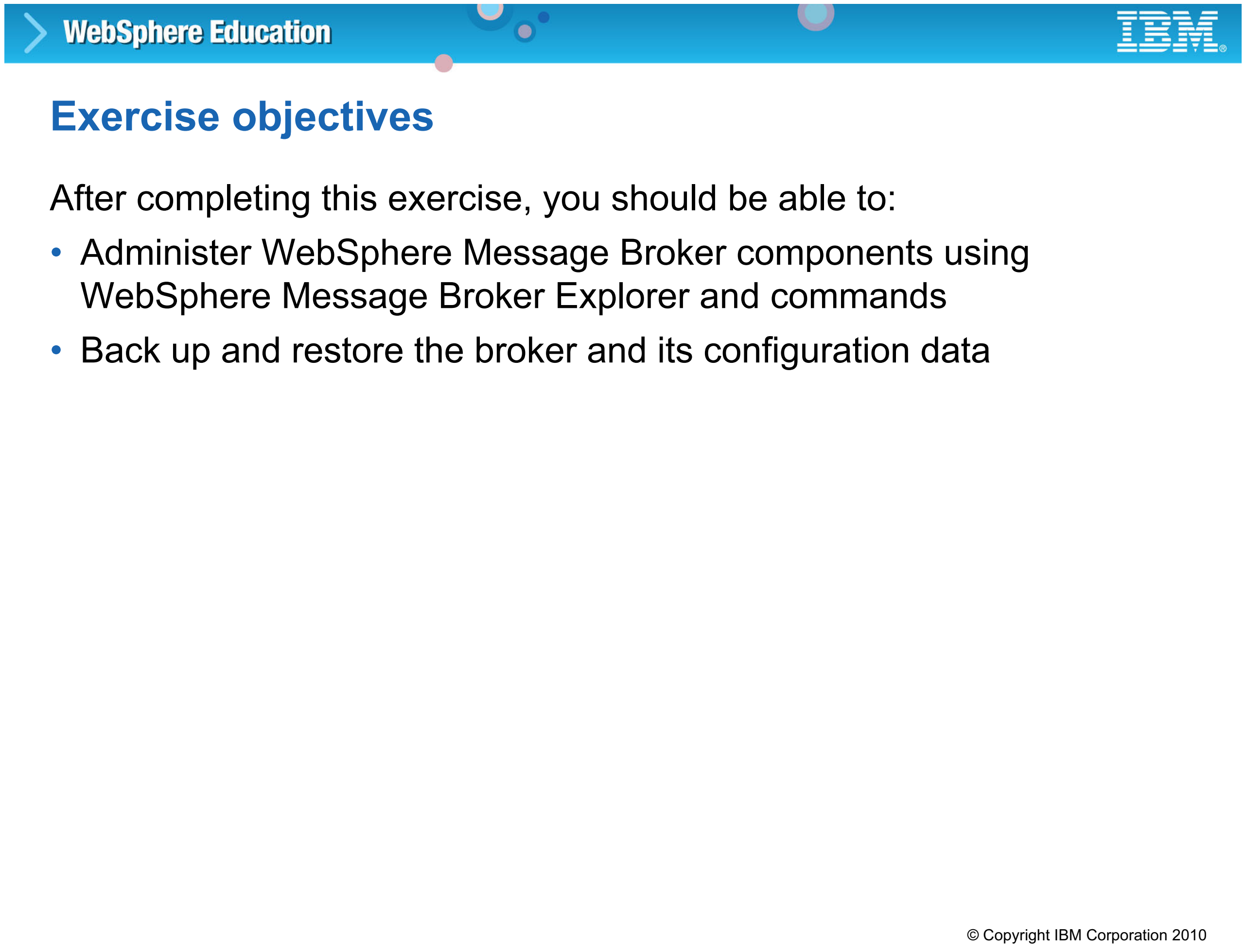
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement — Next: Exercise objectives



WebSphere Education

Exercise objectives

After completing this exercise, you should be able to:

- Administer WebSphere Message Broker components using WebSphere Message Broker Explorer and commands
- Back up and restore the broker and its configuration data

Figure 4-53. Exercise objectives WM643 / VM6431.0

Notes:

See the *Student Exercises Guide* for detailed instructions.

Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —

Unit 5. Implementing WebSphere Message Broker security

Estimated time

01:30 lecture, 01:00 exercise

What this unit is about

This unit describes the WebSphere Message Broker implementations for broker components security, message flow security, and broker administration security.

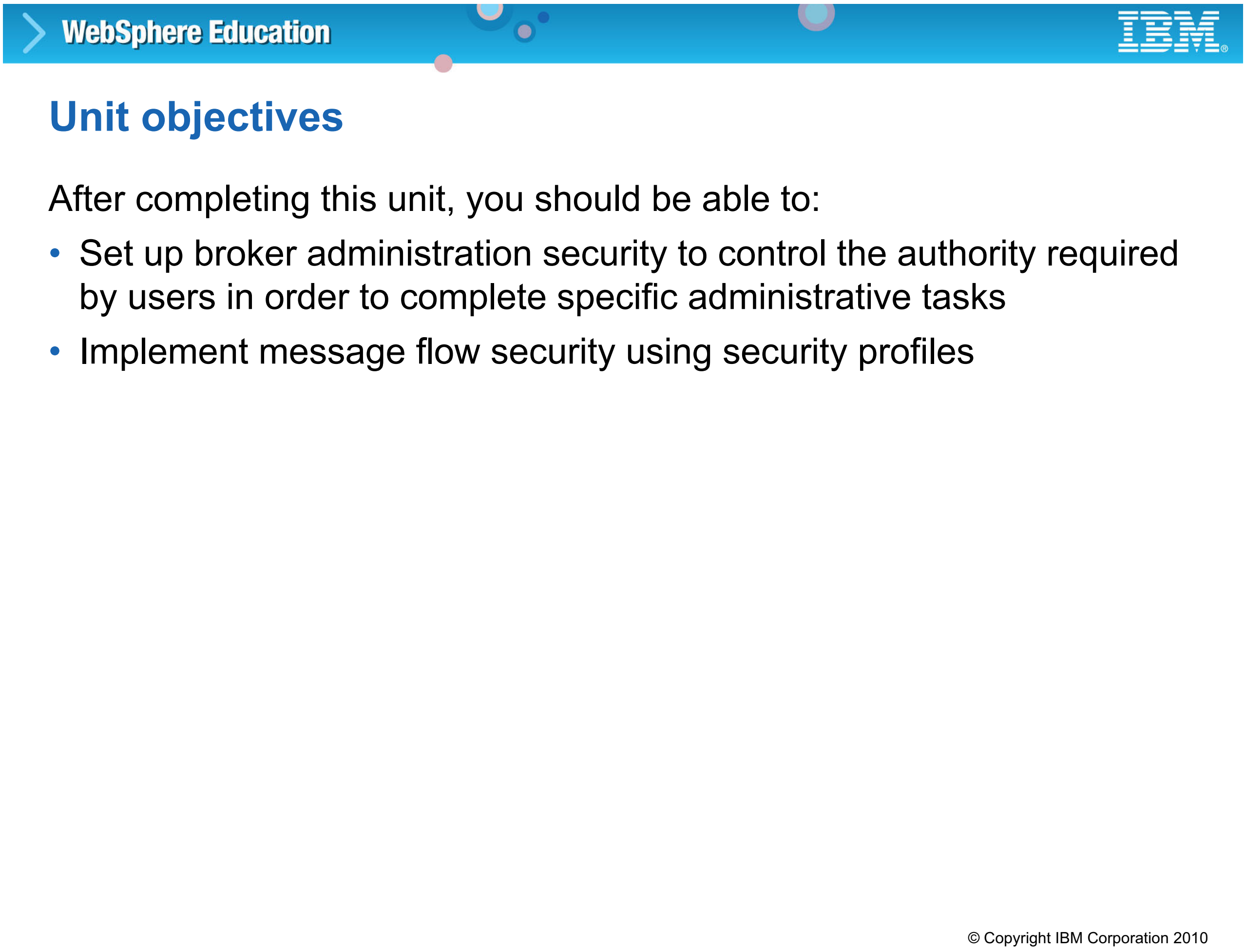
What you should be able to do

After completing this unit, you should be able to:

- Set up broker administration security to control the authority required by users in order to complete specific administrative tasks
- Implement message flow security using security profiles

How you will check your progress

- Review checkpoint questions
- Exercise 4: Administering broker security



WebSphere Education

Unit objectives

After completing this unit, you should be able to:

- Set up broker administration security to control the authority required by users in order to complete specific administrative tasks
- Implement message flow security using security profiles

Figure 5-1. Unit objectives WM643 / VM6431.0

Notes:

Instructor notes:

Purpose — List unit objectives

Details — Security is an important consideration for both developers of WebSphere Message Broker applications, and for system administrators configuring WebSphere Message Broker authorities.

This unit describes the implementation of broker security. Security specific to Web services and Web service nodes is discussed in Unit 9.

When you are designing a WebSphere Message Broker application, it is important to consider the security measures that are needed to protect the information in the system.

Additional information —

Transition statement — Next: WebSphere Message Broker security

5.1. Broker component security

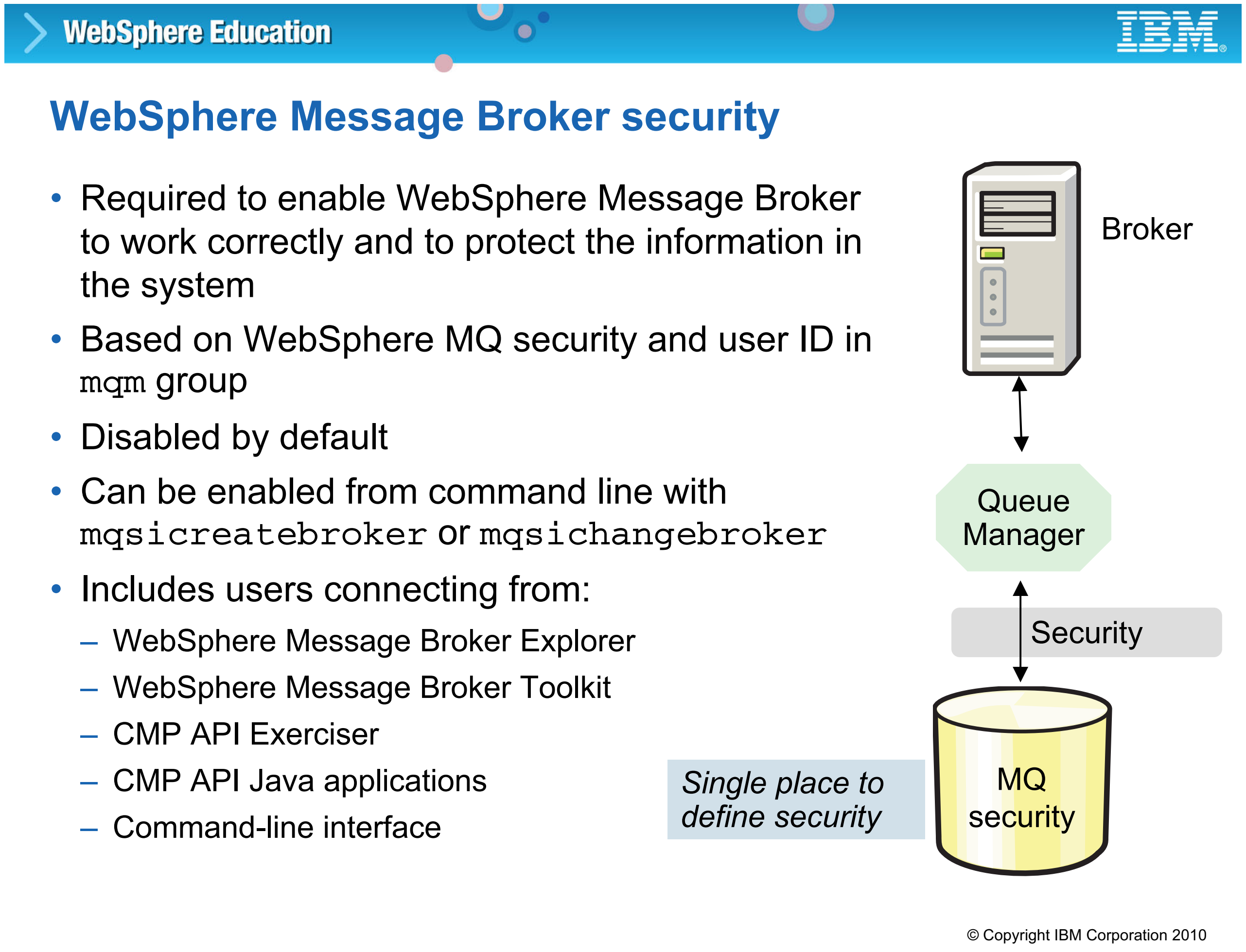
Instructor topic introduction

What students will do — Learn how to implement security for broker components such as broker queues and the broker registry.

How students will do it — Listen to lecture, answer checkpoint questions, and complete a hands-on exercise.

What students will learn — How to implement security for their environments.

How this will help students on their job — Prevent unauthorized users from gaining access to broker components.



WebSphere Message Broker security

- Required to enable WebSphere Message Broker to work correctly and to protect the information in the system
- Based on WebSphere MQ security and user ID in `mqm` group
- Disabled by default
- Can be enabled from command line with `mqsicreatebroker` or `mqsichangebroker`
- Includes users connecting from:
  - WebSphere Message Broker Explorer
  - WebSphere Message Broker Toolkit
  - CMP API Exerciser
  - CMP API Java applications
  - Command-line interface

Single place to define security

© Copyright IBM Corporation 2010

Figure 5-2. WebSphere Message Broker security WM643 / VM6431.0

Notes:

In WebSphere Message Broker, the broker is the policy enforcement point, and WebSphere MQ is the policy decision point. This results in a single place to define security; building on existing skills by taking advantage of security support in WebSphere MQ.

WebSphere Message Broker security is based on permissions assigned to users and groups based on the user ID in the WebSphere MQ message descriptor. Security is disabled by default but can be enabled using commands, and through the WebSphere Message Broker Explorer or CMP API Exerciser. Security can also be specified when you create a broker.

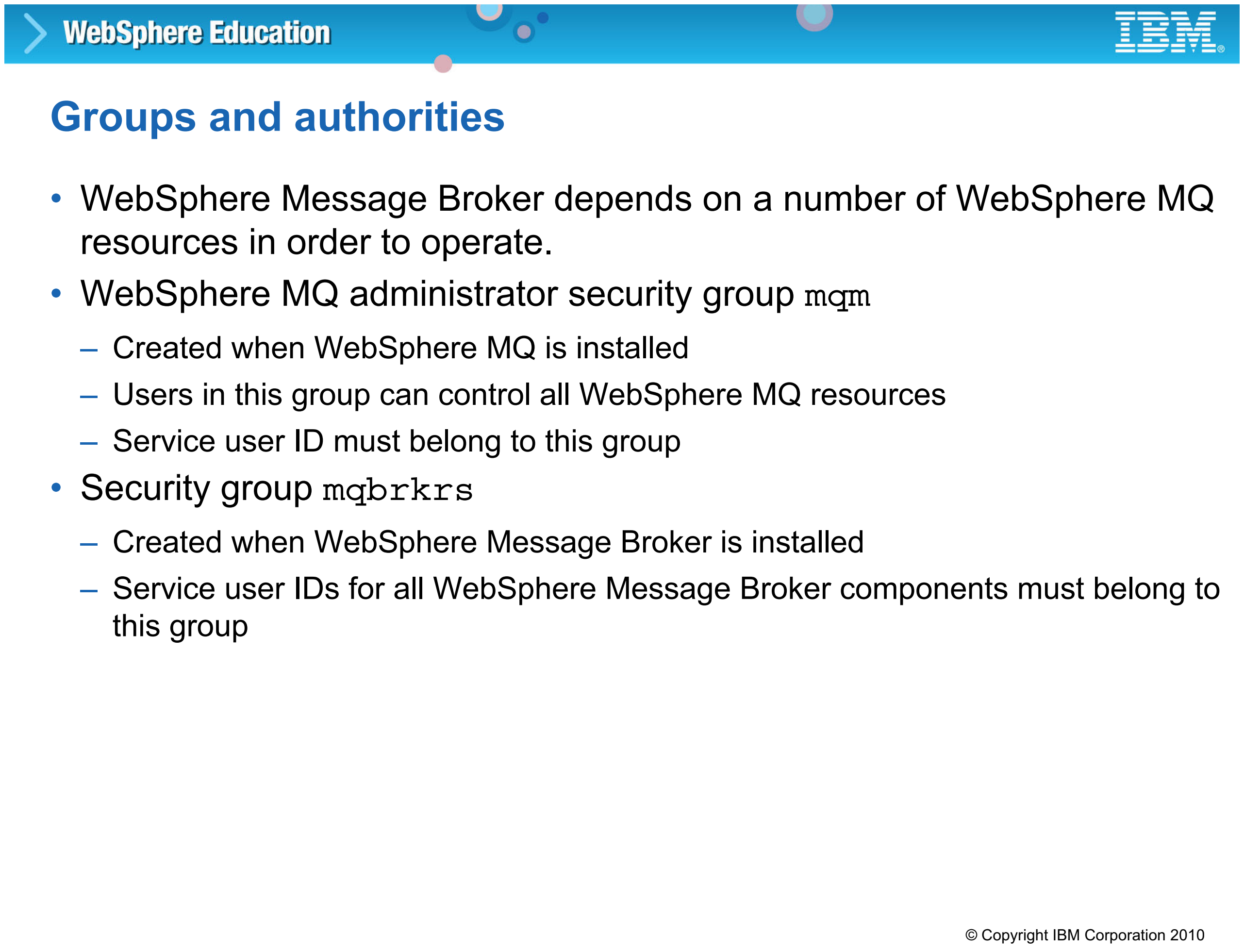
Instructor notes:

Purpose — Provide an overview of WebSphere Message Broker security.

Details — Some of this information has already been covered in a previous unit. Focus on any new concepts.

Additional information —

Transition statement — WebSphere Message Broker depends on WebSphere MQ security groups.



Groups and authorities

- WebSphere Message Broker depends on a number of WebSphere MQ resources in order to operate.
- WebSphere MQ administrator security group `mqm`
  - Created when WebSphere MQ is installed
  - Users in this group can control all WebSphere MQ resources
  - Service user ID must belong to this group
- Security group `mqbrkrs`
  - Created when WebSphere Message Broker is installed
  - Service user IDs for all WebSphere Message Broker components must belong to this group

Figure 5-3. Groups and authorities

WM643 / VM6431.0

Notes:

WebSphere MQ is the security policy decision point for WebSphere Message Broker.

The WebSphere MQ `mqm` administrator security group is created when WebSphere MQ is installed. It identifies users that can control all WebSphere MQ resources including the broker queue manager. Any new users that must be able to control WebSphere MQ resources must be added to this group. On Windows, the broker service ID must also belong to this group.

The security group `mqbrkrs` is created when WebSphere Message Broker is installed. Members of the `mqbrkrs` group can modify a broker, back up and restore a broker, and delete a broker. On Windows, the broker service ID must belong to this group.

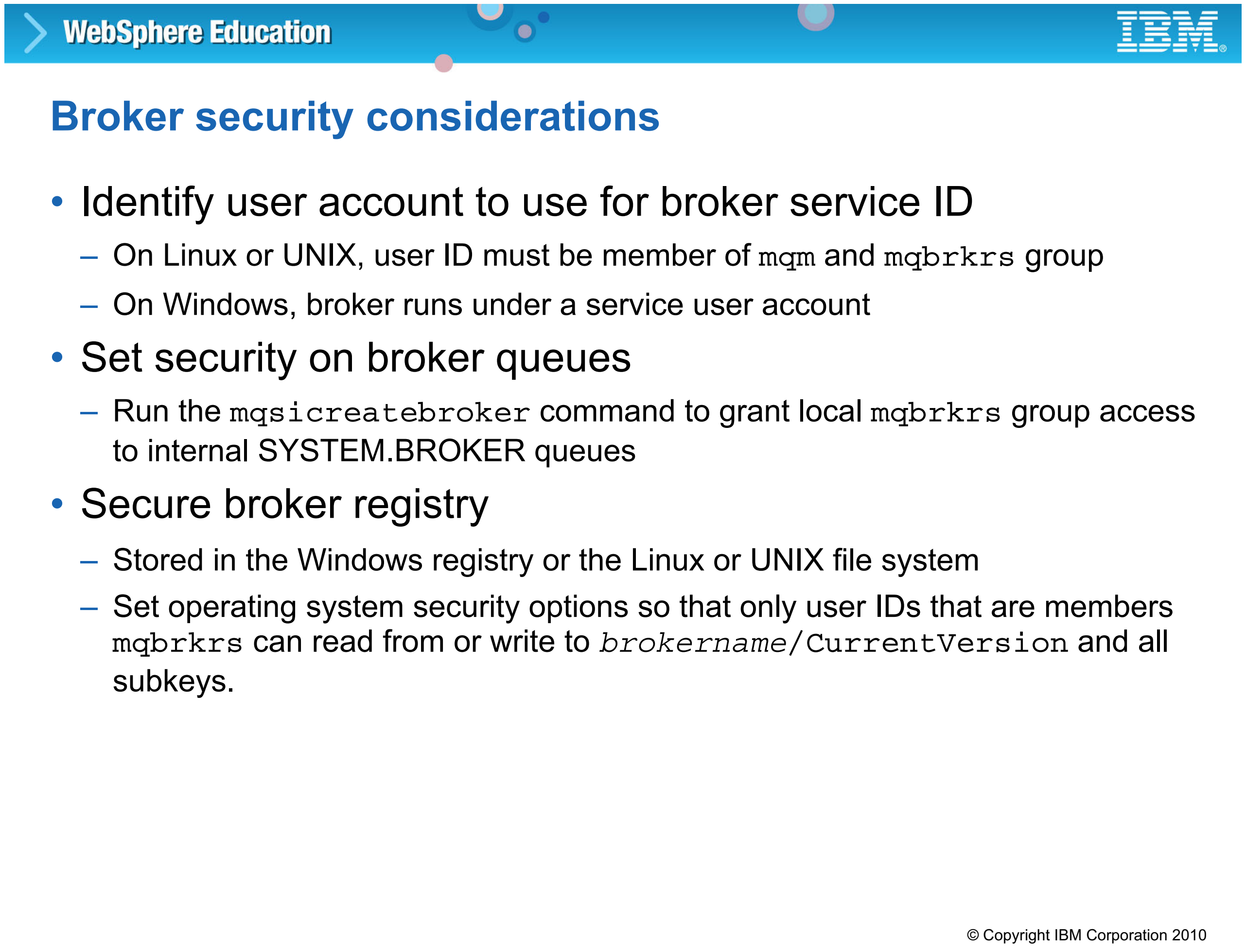
Instructor notes:

Purpose — Describe the WebSphere MQ security groups that identify the users that can manage queues and brokers.

Details — On all Windows platforms, there is no longer any requirement for the service user ID to be a member of the **Administrators** group. The only requirement is that the service user ID is a member of the `mqbrkrks` group. The LocalSystem account can be used as the service user ID by specifying LocalSystem for the `-i` parameter on the `mqsicreatebroker` command.

Additional information — None.

Transition statement — What must you consider when implementing broker security?



WebSphere Education
IBM

Broker security considerations

- Identify user account to use for broker service ID
  - On Linux or UNIX, user ID must be member of `mqm` and `mqbrkrs` group
  - On Windows, broker runs under a service user account
- Set security on broker queues
  - Run the `mqsicreatebroker` command to grant local `mqbrkrs` group access to internal SYSTEM.BROKER queues
- Secure broker registry
  - Stored in the Windows registry or the Linux or UNIX file system
  - Set operating system security options so that only user IDs that are members `mqbrkrs` can read from or write to *`brokername`*`/CurrentVersion` and all subkeys.

© Copyright IBM Corporation 2010

Figure 5-4. Broker security considerations WM643 / VM6431.0

Notes:

Consider several factors when you are deciding which users can run broker commands, and which users can control security for other broker resources. Although most security for the broker and broker resources is optional, you might find it appropriate to restrict the tasks that some user IDs can perform. You can then apply greater control to monitor changes.

On Windows, the broker runs under a service user account. On Linux or UNIX, the user ID under which you run the `mqsistart` command becomes the user ID under which the broker component process runs. The `mqm` and `mqbrkrs` groups are used to identify users that can control the broker service.

When you run the `mqsicreatebroker` command, the local `mqbrkrs` group is granted access to internal queues whose names begin with the characters `SYSTEM.BROKER.`

Broker operation depends on the information in the broker registry, which must be secured to guard against accidental corruption. The broker registry is stored in the Windows registry or in the `qm.ini` file on the Linux or UNIX file system.

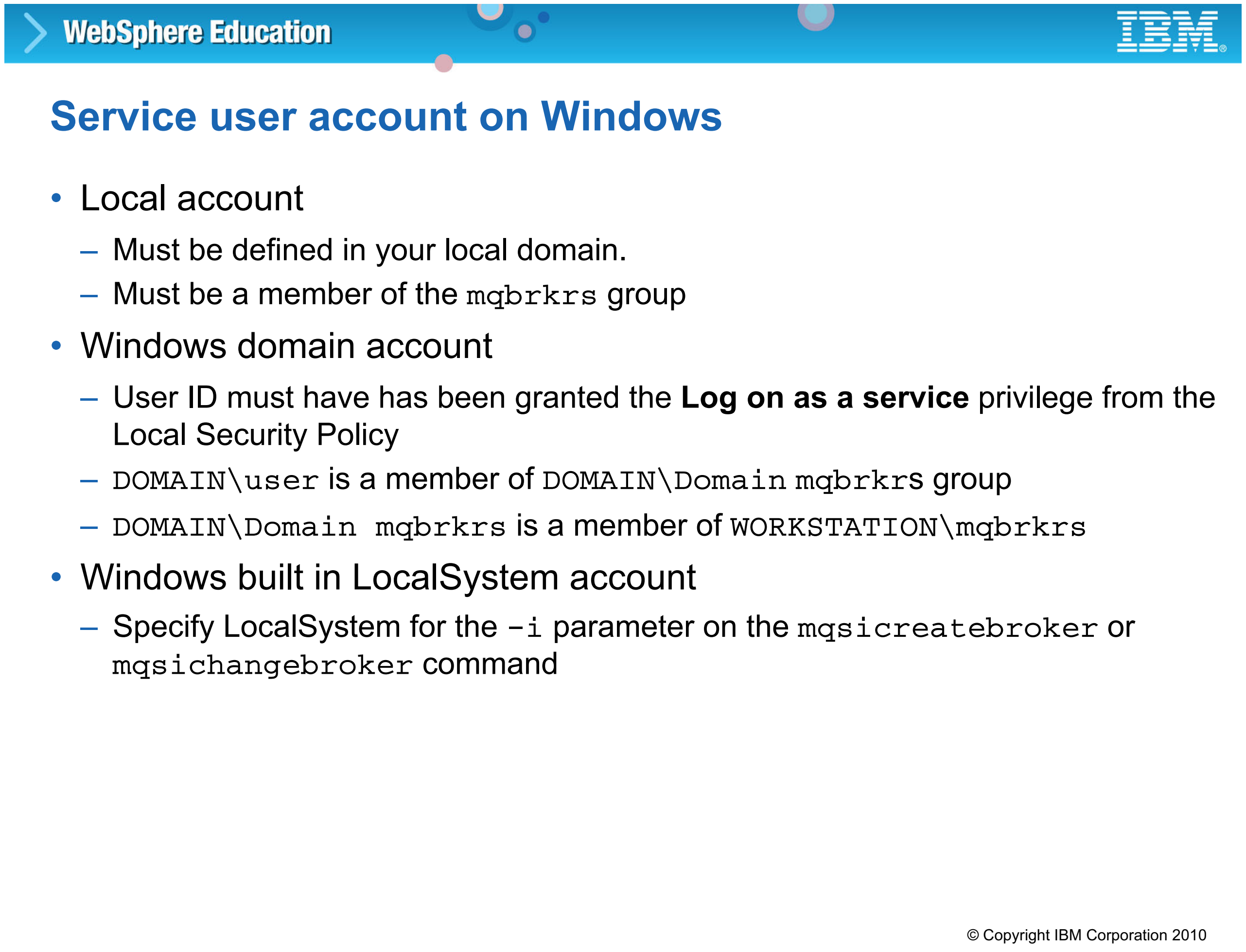
Instructor notes:

Purpose — List the three tasks that must be considered when implementing security on the broker, broker queues, and broker registry.

Details — You must consider several security aspects when you are setting up brokers running on Windows, Linux, or UNIX platforms.

Additional information — None

Transition statement — You have three options when configuring the user account on Windows,


WebSphere Education
IBM

Service user account on Windows

- Local account
  - Must be defined in your local domain.
  - Must be a member of the `mqbrkrs` group
- Windows domain account
  - User ID must have has been granted the **Log on as a service** privilege from the Local Security Policy
  - `DOMAIN\user` is a member of `DOMAIN\Domain mqbrkrs` group
  - `DOMAIN\Domain mqbrkrs` is a member of `WORKSTATION\mqbrkrs`
- Windows built in LocalSystem account
  - Specify LocalSystem for the `-i` parameter on the `mqsicreatebroker` or `mqsichangebroker` command

Figure 5-5. Service user account on Windows WM643 / VM6431.0

Notes:

On the Windows platform, the broker runs under a service user account. The service user account can be a Windows local account, a Windows domain account, or the built-in LocalSystem account. The figure lists the requirements for each of the options.

Instructor notes:

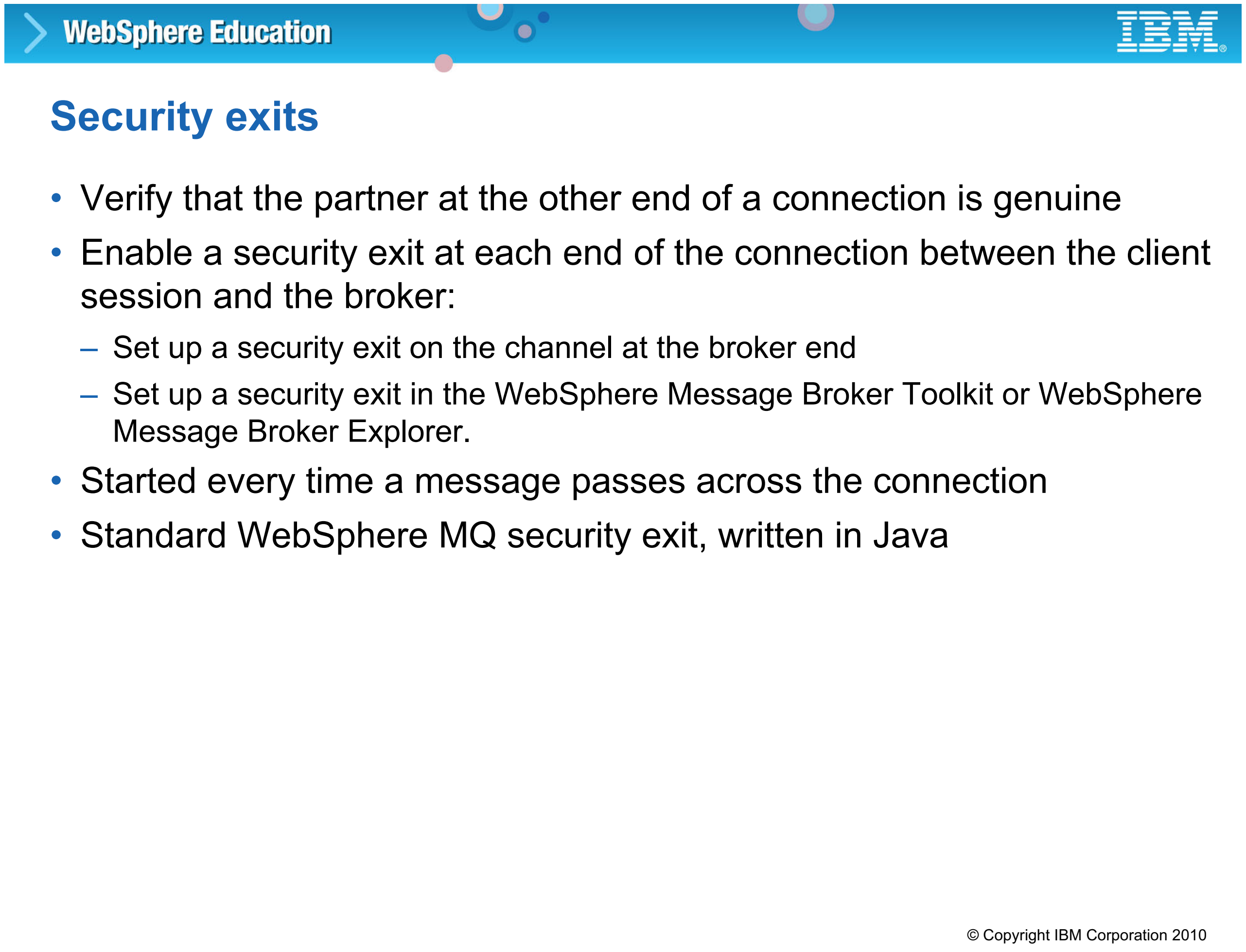
Purpose — Describe the three options for a service user account on Windows.

Details — For cases 1 and 2, the user ID chosen must be granted the *Logon as a service* privilege. This is normally done automatically by the `mqsichangebroker` command or the `mqsichangeproperties` command when a service user ID is specified that does not have this privilege.

However, if you want to do this manually before running these commands, you can do this by using the Local Security Policy tool in Windows, which you can access by selecting **Control Panel > Administrative Tools > Local Security Policy**.

Additional information — None.

Transition statement — Another important security point is the connection between WebSphere Message Broker Toolkit or WebSphere Message Broker and remote brokers.



WebSphere Education
IBM

Security exits

- Verify that the partner at the other end of a connection is genuine
- Enable a security exit at each end of the connection between the client session and the broker:
  - Set up a security exit on the channel at the broker end
  - Set up a security exit in the WebSphere Message Broker Toolkit or WebSphere Message Broker Explorer.
- Started every time a message passes across the connection
- Standard WebSphere MQ security exit, written in Java

Figure 5-6. Security exits WM643 / VM6431.0

Notes:

When you connect from the WebSphere Message Broker Toolkit or WebSphere Message Broker Explorer to a broker on another computer, you might want to protect access to the broker from client programs. Security exit programs verify that the partner at the other end of a connection is genuine. This security exit at the broker end has no special requirements; you can use the WebSphere MQ security exit facility.

You can enable a security exit at each end of the connection between your client session and the broker:

- Set up a security exit on the channel at the broker end. This security exit has no special requirements; you can provide a standard security exit.
- Set up a security exit in the WebSphere Message Broker Toolkit or WebSphere Message Broker Explorer. Identify the security exit properties when you connect to the broker.

For an overview of security exits and details in their implementation, see "Channel security exit programs" in the WebSphere MQ Version 7 Information Center.

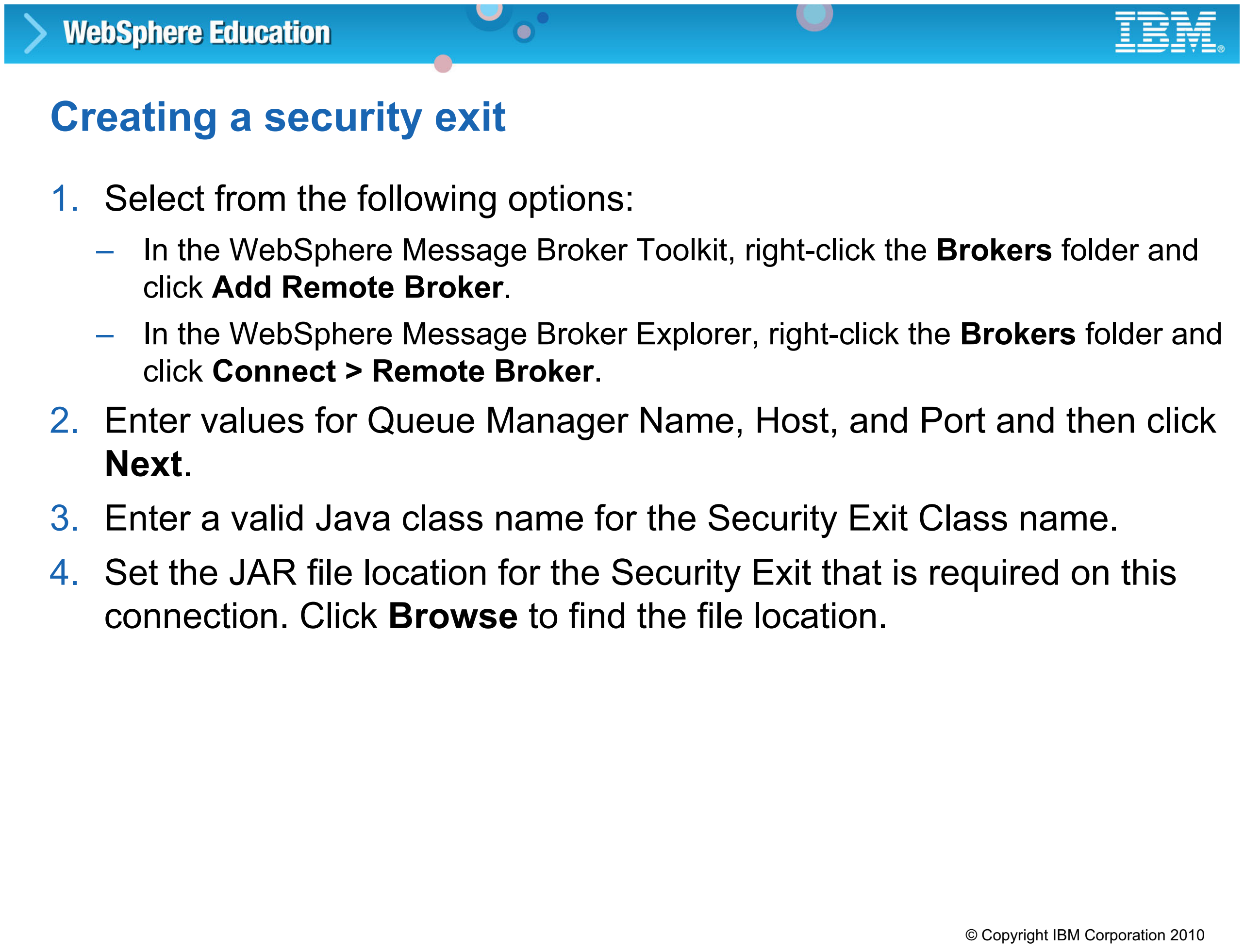
Instructor notes:

Purpose — Provide an overview of security exit implementation for WebSphere Message Broker.

Details — WebSphere MQ security exits were covered in the IBM WebSphere MQ V7 System Administration course, which is a prerequisite for this class so an in-depth discussion is not included here.

Additional information — None.

Transition statement — The next slide provides an overview of the steps for creating a security exit.



WebSphere Education

Creating a security exit

1. Select from the following options:
   - In the WebSphere Message Broker Toolkit, right-click the **Brokers** folder and click **Add Remote Broker**.
   - In the WebSphere Message Broker Explorer, right-click the **Brokers** folder and click **Connect > Remote Broker**.
2. Enter values for Queue Manager Name, Host, and Port and then click **Next**.
3. Enter a valid Java class name for the Security Exit Class name.
4. Set the JAR file location for the Security Exit that is required on this connection. Click **Browse** to find the file location.

Figure 5-7. Creating a security exit WM643 / VM6431.0

Notes:

To create a security exit on the WebSphere MQ channel SYSTEM.BKR.SVRCONN that you define for communications between the WebSphere Message Broker Toolkit, or WebSphere Message Broker Explorer and the broker, you must define a security exit when you create the connection.

The figure lists the steps for creating a security exit.

Instructor notes:

Purpose — List the steps for creating a security exit.

Details — The Security Exit Class name must be a valid Java class name.

Additional information — Alternatively, you can use SSL to communicate between the Administration API for WebSphere Message Broker (also known as the CMP API) and the broker. SSL is discussed later in this unit.

Transition statement — The next topic describes broker administration security to control the users that can manage brokers and execution groups

5.2. Broker administration security

Instructor topic introduction

What students will do — Activate broker administration security and set authorization for users and groups.

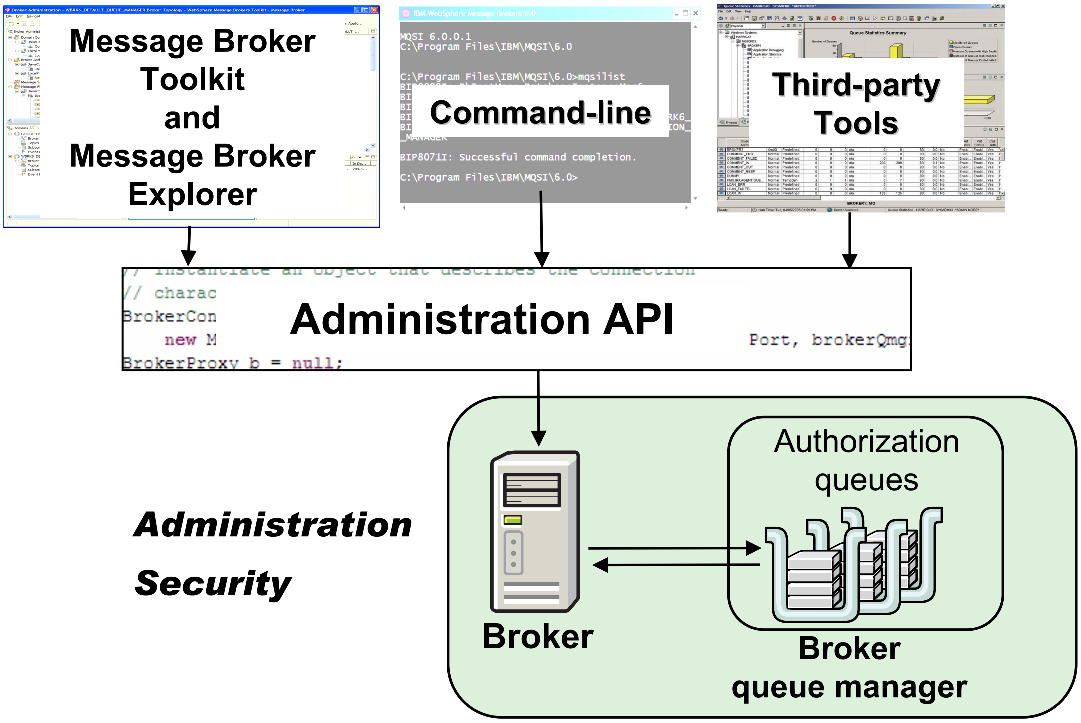
How students will do it — Listening to lecture, answering checkpoint questions, and completing a hands-on exercise.

What students will learn — How to implement broker administration security.

How this will help students on their job — Allow administrators to limit control of broker and broker resources to authorized users only.

WebSphere Education

Administration security



```
MQSI 6.0.0.1
C:\Program Files\IBM\MQSI\6.0

C:\Program Files\IBM\MQSI\6.0>mqsilist
BI
BI
BI
BI                                                RK6_
BI                                                ION_
_MANAGER

BIP8071I: Successful command completion.

C:\Program Files\IBM\MQSI\6.0>
```

```
// charac
BrokerCon                                   Port, brokerQmg
    new M
BrokerProxy b = null;
```

Figure 5-8. Administration security WM643 / VM6431.0

Notes:

The diagram in the figure shows another way to view the WebSphere Message Broker architecture. The top layer, shows all the communication links to the broker through the Administration (CMP) API. These include the WebSphere Message Broker Toolkit and WebSphere Message Broker Explorer, command-line interface, and third-party tools that might be using the Administration API to communicate with the broker.

In WebSphere Message Broker V7, the broker communicates with authorization queues which are used to grant or deny users authority to act on broker objects. There is one authorization queue for each broker (SYSTEM.BROKER.AUTH) and one authorization queue for each execution group (SYSTEM.BROKER.AUTH.*EG* where *EG* is the name of the execution group).

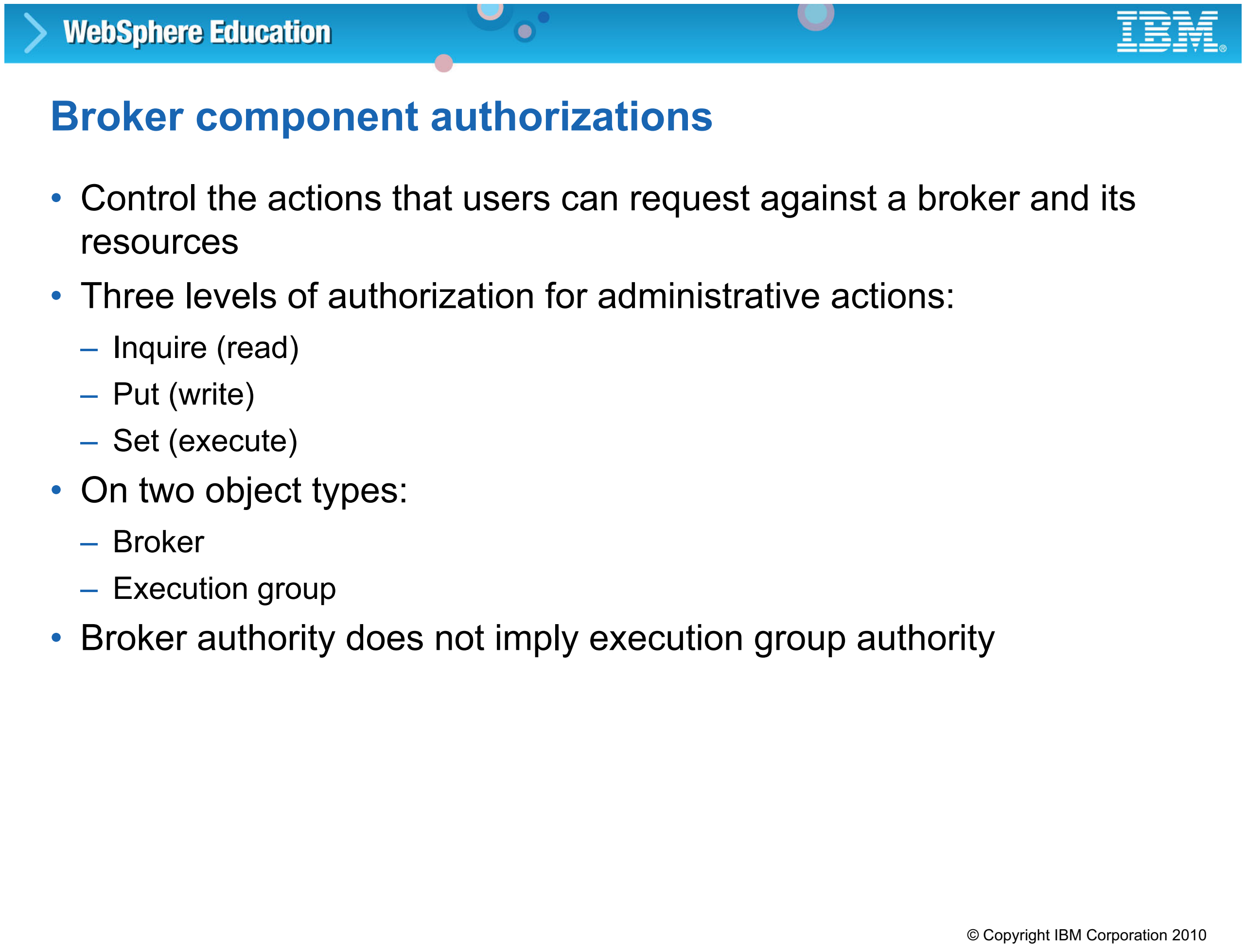
Instructor notes:

Purpose — Provide an overview of the broker administration security architecture.

Details — This diagram is an overview only.

Additional information —

Transition statement — Next: Broker component authorizations



WebSphere Education

Broker component authorizations

- Control the actions that users can request against a broker and its resources
- Three levels of authorization for administrative actions:
  - Inquire (read)
  - Put (write)
  - Set (execute)
- On two object types:
  - Broker
  - Execution group
- Broker authority does not imply execution group authority

Figure 5-9. Broker component authorizations WM643 / VM6431.0

Notes:

WebSphere Message Broker used to have many different levels of authorization. Now there are only three levels of authorization: inquire (read), put (write) and set (execute).

Permissions are configured on only two types of objects: brokers and execution group.

Broker and execution group authority are separate.

If execution groups are renamed, the existing authorization queue is deleted, and a new authorization queue is created, if it does not exist. The permissions set on the original queue are lost.

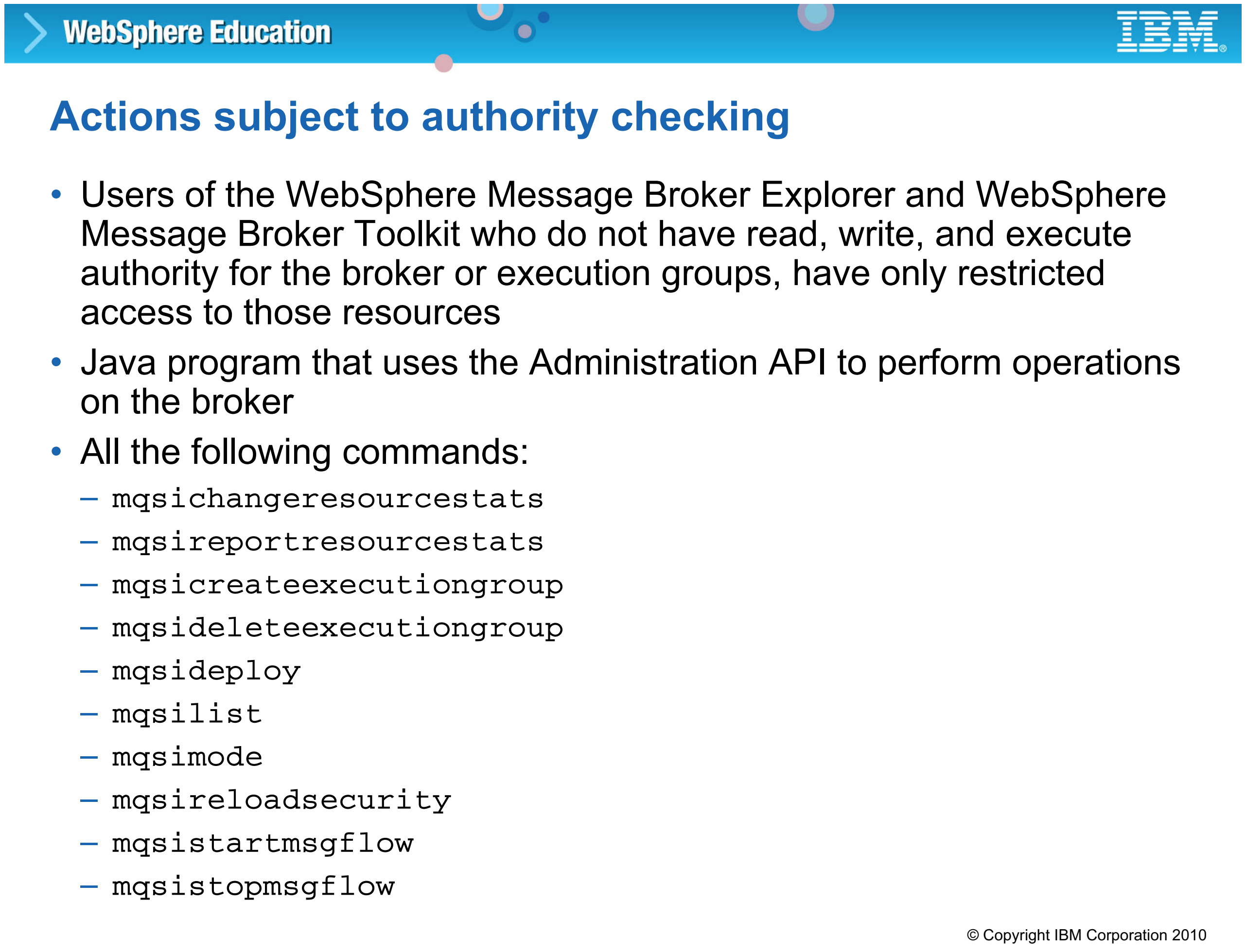
Instructor notes:

Purpose — Provide an overview of broker component authorization.

Details — The interface to set authorizations uses the terms get, set, and inquire instead of read, write, and execute.

Additional information — None.

Transition statement — When broker component authorization is activated, certain actions are subject to authority checking.



Actions subject to authority checking

- Users of the WebSphere Message Broker Explorer and WebSphere Message Broker Toolkit who do not have read, write, and execute authority for the broker or execution groups, have only restricted access to those resources
- Java program that uses the Administration API to perform operations on the broker
- All the following commands:
  - `mqsichangeresourcestats`
  - `mqsireportresourcestats`
  - `mqsicreateexecutiongroup`
  - `mqsideleteexecutiongroup`
  - `mqsideploy`
  - `mqsilist`
  - `mqsimode`
  - `mqsireloadsecurity`
  - `mqsistartmsgflow`
  - `mqsistopmsgflow`

Figure 5-10. Actions subject to authority checking WM643 / VM6431.0

Notes:

When broker authorization security is activated, certain actions are restricted based on the access that has been configured for each user or group.

For example, if developers are working with existing execution groups, and pre-production resources such as BAR files, you must give them inquire, put, and set permissions on the execution group authorization queues.

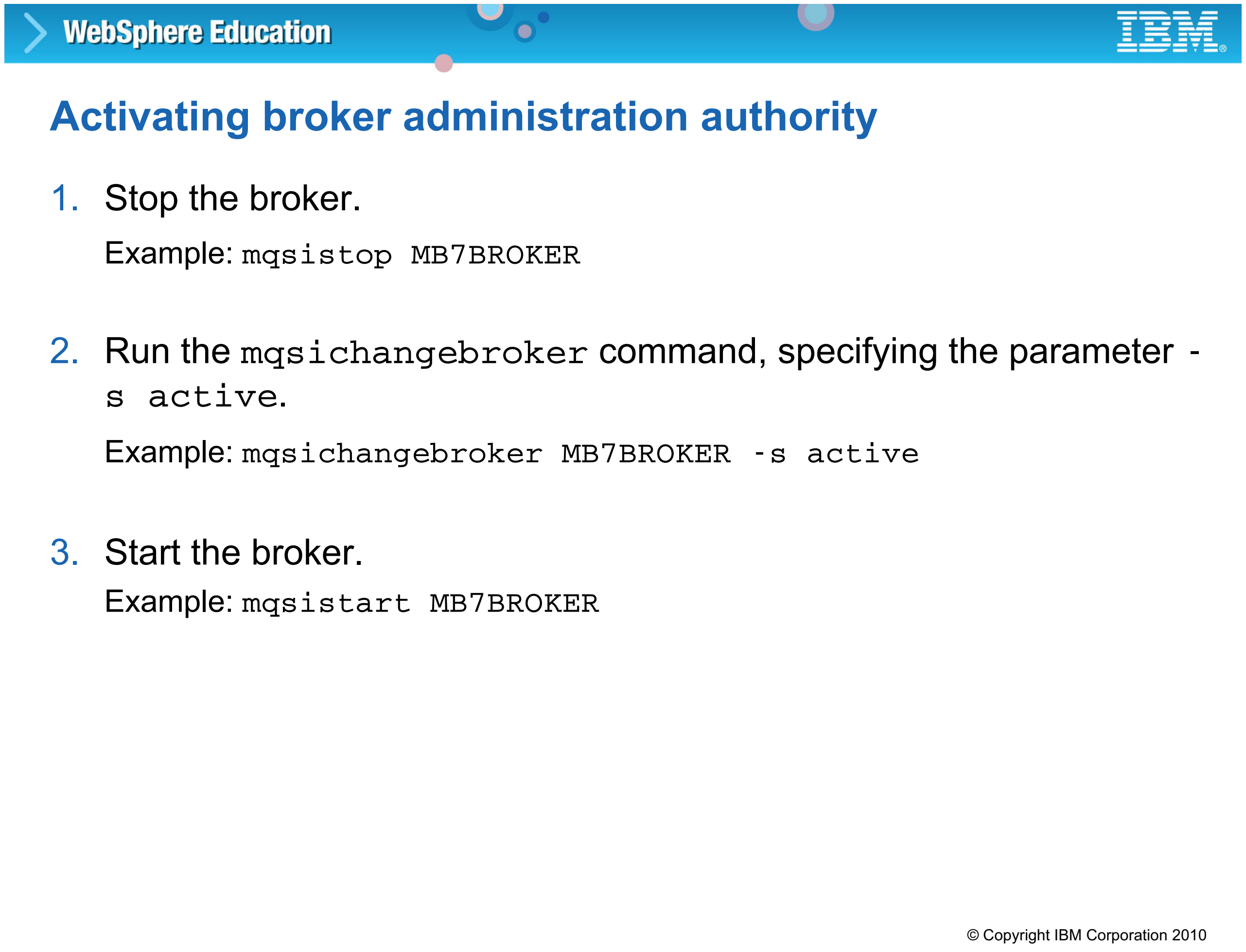
Instructor notes:

Purpose — Describe the actions and commands that are subject to authority checking if broker security is activated.

Details —

Additional information —

Transition statement — You activate broker authorization from a command line.



WebSphere Education

Activating broker administration authority

1. Stop the broker.

   Example: `mqsistop MB7BROKER`

2. Run the `mqsichangebroker` command, specifying the parameter `-s active`.

   Example: `mqsichangebroker MB7BROKER -s active`

3. Start the broker.

   Example: `mqsistart MB7BROKER`

Figure 5-11. Activating broker administration authority WM643 / VM6431.0

Notes:

Broker administration authority is an option. You must activate broker administration security for the broker before you grant and revoke authority for requests sent to that broker.

To activate broker administration authority:

1. Stop the broker.
2. Run the `mqsichangebroker` command and specify the broker followed by the `-s active` parameter.
3. Restart the broker.

The examples in the figure use the command line to stop and start the broker. You can also use WebSphere Message Broker Explorer, the WebSphere Message Broker Toolkit or Window Services to stop or start the broker.

Instructor notes:

Purpose — List steps for activating broker administration authority from a command line.

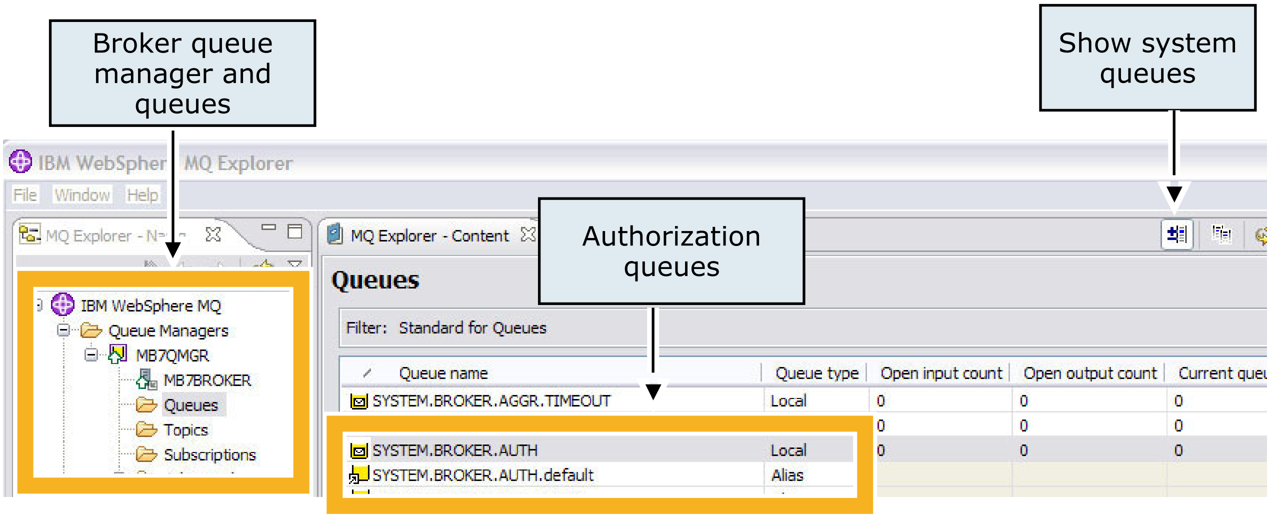
Details —

Additional information —

Transition statement — You manage the authorization queues from a command line or from WebSphere Message Broker Explorer.

WebSphere Education

Authorization queues



Queues

Filter: Standard for Queues

Queue name	Queue type	Open input count	Open output count	Current queu
SYSTEM.BROKER.AGGR.TIMEOUT	Local	0	0	0
		0	0	0
SYSTEM.BROKER.AUTH	Local	0	0	0
SYSTEM.BROKER.AUTH.default	Alias			

- One queue for broker authorization
- One queue for each execution group for execution group authorization

Figure 5-12. Authorization queues

WM643 / VM6431.0

Notes:

Each broker queue manager contains one authorization queue for the broker and one authorization queue for each execution group.

The WebSphere MQ broker group (mqbrks) is given authorization to the authorization queues by default when security is enabled. Users are prevented from performing any actions on brokers or execution groups until they are given authorization by the administrator.

Click the **Show system queues** icon to show the authorization queues in WebSphere Message Broker Explorer.

No messages are written to the authorization queues; they are only used to identify broker and execution group permissions. The depth on any authorization queues should always be zero.

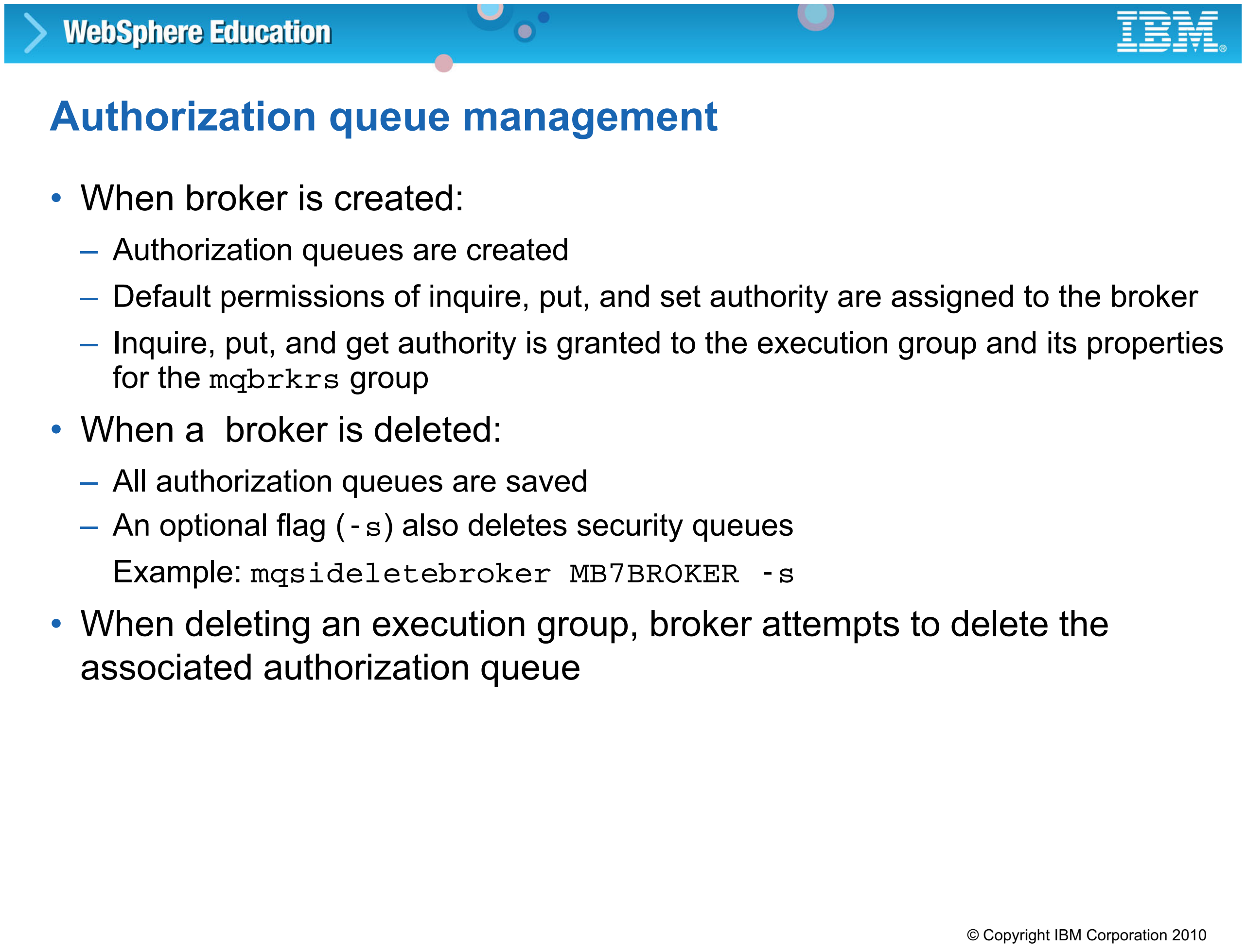
Instructor notes:

Purpose — Show broker authorization queues in WebSphere Message Broker Explorer.

Details — Activate broker authority and open WebSphere Message Broker Explorer to show the broker system queues. The concept of WebSphere MQ system queues should be reviewed for the students.

Additional information — None.

Transition statement — What happens when a new broker is created or an existing broker is deleted?



Authorization queue management

- When broker is created:
  - Authorization queues are created
  - Default permissions of inquire, put, and set authority are assigned to the broker
  - Inquire, put, and get authority is granted to the execution group and its properties for the `mqbrkrs` group
- When a broker is deleted:
  - All authorization queues are saved
  - An optional flag (`-s`) also deletes security queues

    Example: `mqsideletebroker MB7BROKER -s`
- When deleting an execution group, broker attempts to delete the associated authorization queue

Figure 5-13. Authorization queue management WM643 / VM6431.0

Notes:

The broker authorization queue is created when you create a broker or when you migrate a broker to WebSphere Message Broker V7, if security is enabled. If, for some reason, the broker cannot create the queues dynamically, a message is written to the system log. In this case, the broker administrator must then either manually create the queue, or run the `mqsichangebroker` to create the queue.

An execution group authorization queue is created when an execution group is created. When an execution group is renamed, a new authorization queue is created. The old queue is then deleted.

The broker authorization queue is not deleted when the broker is deleted unless you specifically indicate that the queue should be deleted.

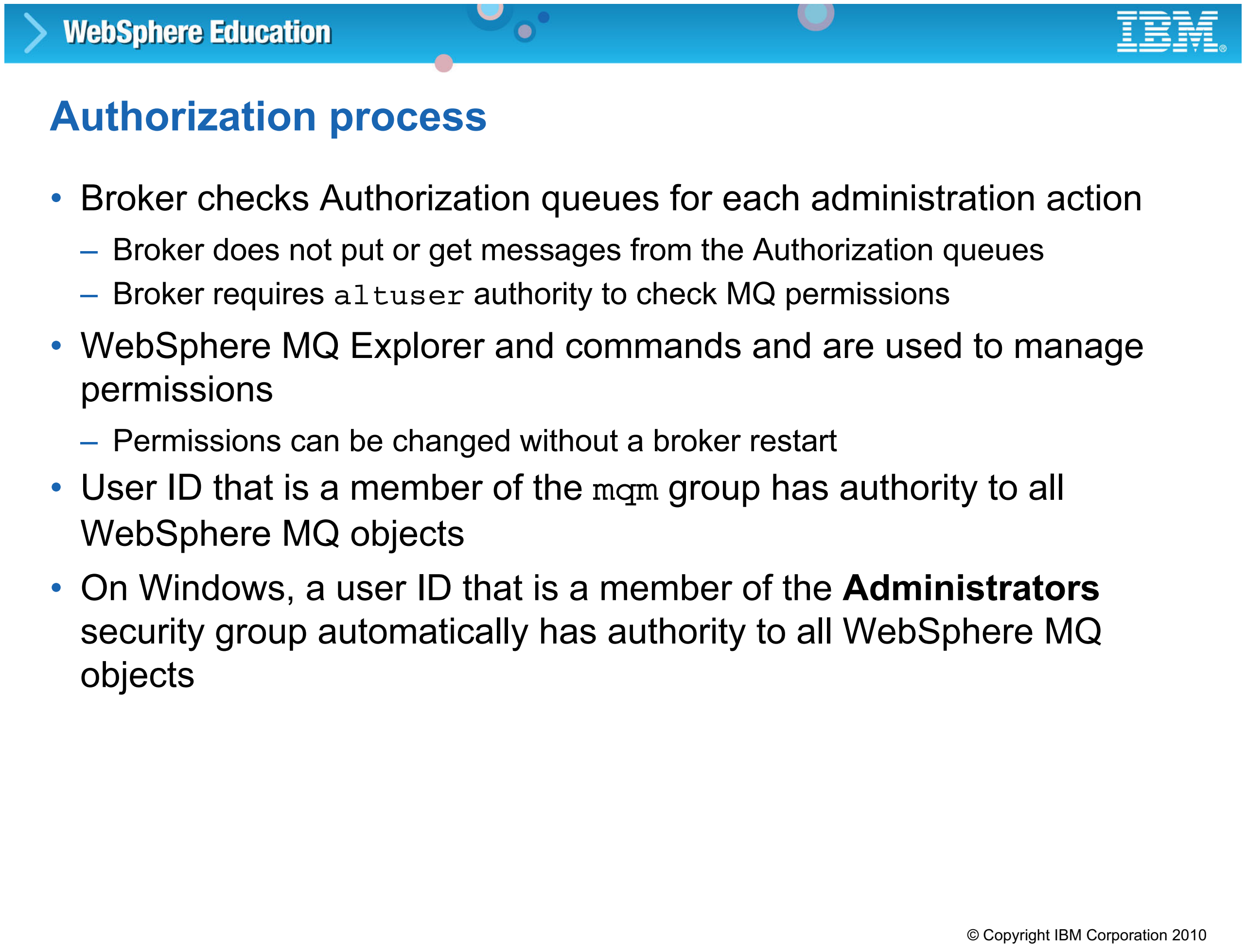
Instructor notes:

Purpose — Describe the actions that occur when a broker or execution group is deleted and broker security is activated.

Details — User IDs are used to determine whether the user has authority. A user needs separate authorization to delete execution groups and brokers as the broker authorization and execution group authorizations are handled by separate queues.

Additional information —

Transition statement — The broker has a specific process for checking authorization.



WebSphere Education

Authorization process

- Broker checks Authorization queues for each administration action
  - Broker does not put or get messages from the Authorization queues
  - Broker requires `altuser` authority to check MQ permissions
- WebSphere MQ Explorer and commands and are used to manage permissions
  - Permissions can be changed without a broker restart
- User ID that is a member of the `mqm` group has authority to all WebSphere MQ objects
- On Windows, a user ID that is a member of the **Administrators** security group automatically has authority to all WebSphere MQ objects

Figure 5-14. Authorization process WM643 / VM6431.0

Notes:

In WebSphere Message Broker, the broker checks the authorization queues for every action. If permission is granted, the action is performed.

The administrator can change permissions at any time without restarting the broker. The broker automatically recognizes the change the next time the user attempts to perform the action.

Any user that has WebSphere MQ alternate user command authority can check the broker authorization permissions.

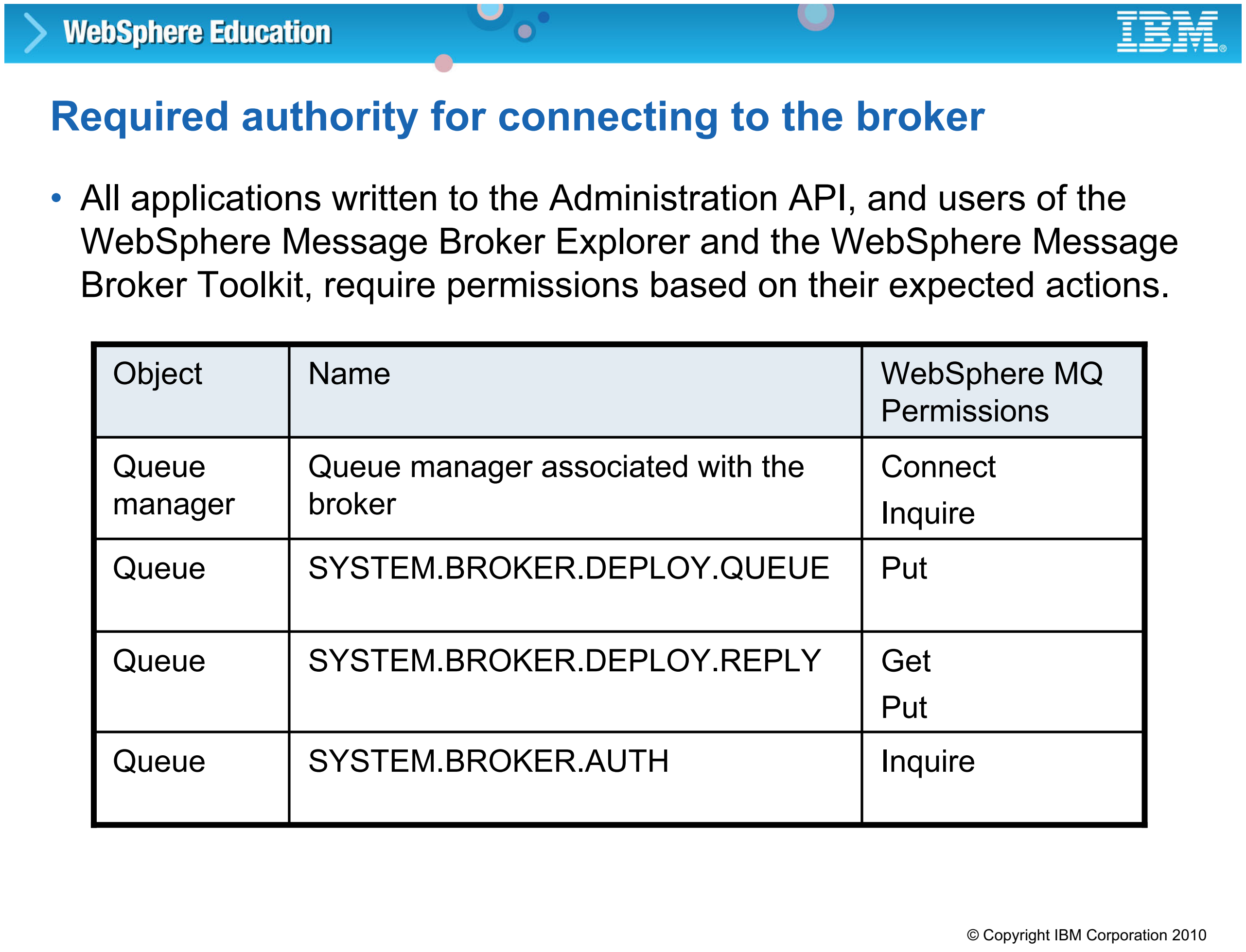
Instructor notes:

Purpose — Describe the authorization process.

Details — The alternate user ID used to validate access to the queue when it is opened. Students should know this if they meet the course prerequisites.

Additional information — None.

Transition statement — There is some basic authorization required to connect to brokers.



WebSphere Education

Required authority for connecting to the broker

- All applications written to the Administration API, and users of the WebSphere Message Broker Explorer and the WebSphere Message Broker Toolkit, require permissions based on their expected actions.

Object	Name	WebSphere MQ Permissions
Queue manager	Queue manager associated with the broker	Connect Inquire
Queue	SYSTEM.BROKER.DEPLOY.QUEUE	Put
Queue	SYSTEM.BROKER.DEPLOY.REPLY	Get Put
Queue	SYSTEM.BROKER.AUTH	Inquire

Figure 5-15. Required authority for connecting to the broker WM643 / VM6431.0

Notes:

If a user or application wants to connect to a broker, you must grant them the appropriate permissions. All applications written to the Administration (CMP) API, and users of the WebSphere Message Broker Explorer and the WebSphere Message Broker Toolkit, require permissions based on their expected actions.

The table in the figure shows the WebSphere MQ permissions that are required to connect to the broker.

The SYSTEM.BROKER.DEPLOY.QUEUE queue s the target for publish/subscribe control requests that applications send to the broker.

The SYSTEM.BROKER.DEPLOY.REPLY queue is the target for publish/subscribe control requests that applications send to the broker

The SYSTEM.BROKER.AUTH queue stores authorization records for administration requests against the broker.

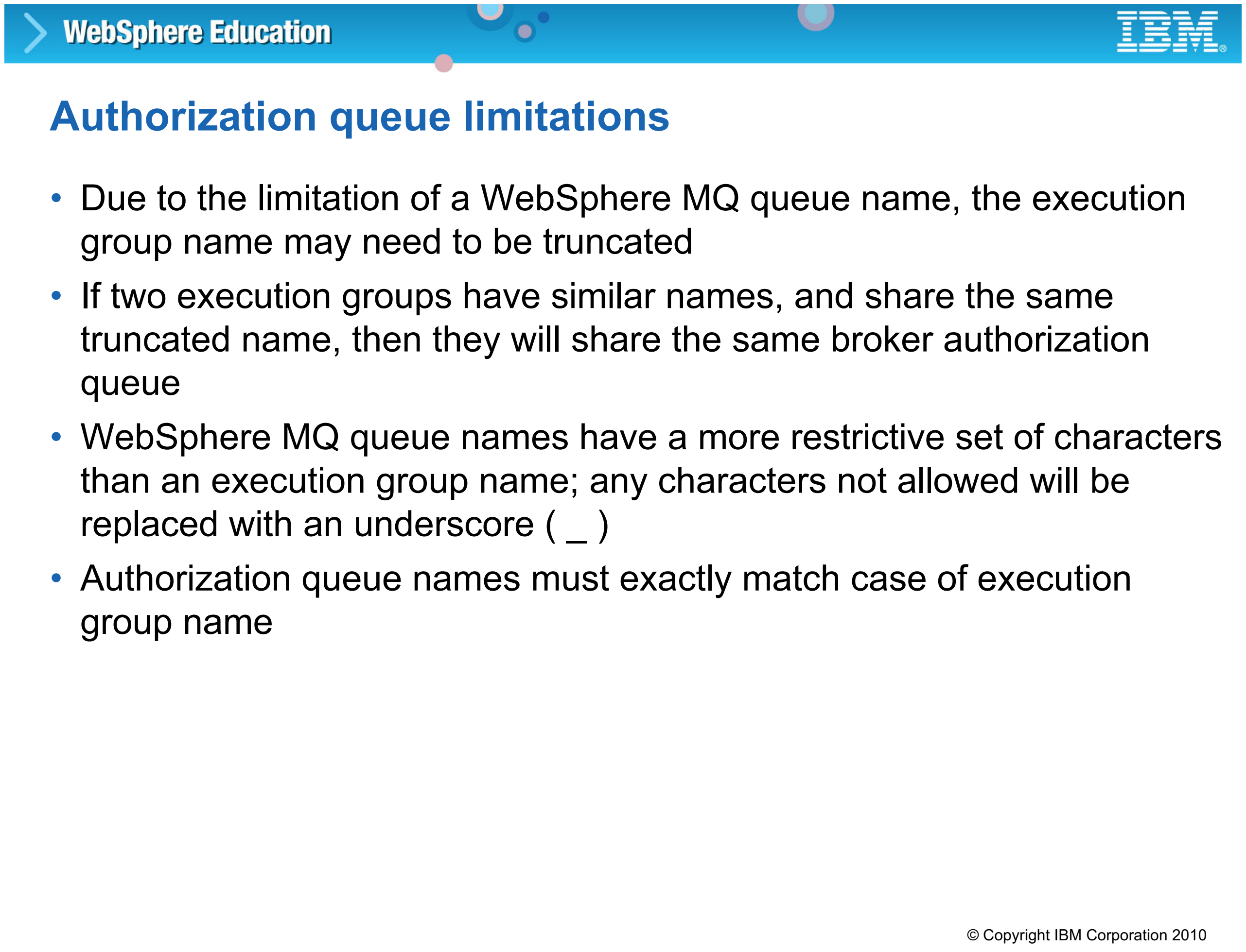
Instructor notes:

Purpose — Describe authority required to users who must be able to connect to the broker.

Details — Users and applications can connect to the broker without Inquire permissions on the SYSTEM.BROKER.AUTH queue, but they are not able to request actions against the broker, including viewing properties.

Additional information — None.

Transition statement — Execution group names are used for creating authorization queue names. For this reason, you should be aware of the limitations on authorization queue names so that you can name your execution groups appropriately.



WebSphere Education

Authorization queue limitations

- Due to the limitation of a WebSphere MQ queue name, the execution group name may need to be truncated
- If two execution groups have similar names, and share the same truncated name, then they will share the same broker authorization queue
- WebSphere MQ queue names have a more restrictive set of characters than an execution group name; any characters not allowed will be replaced with an underscore (_)
- Authorization queue names must exactly match case of execution group name

Figure 5-16. Authorization queue limitations WM643 / VM6431.0

Notes:

If you create an execution group with a name longer than 30 characters, it is truncated to 30 characters and the truncated name is used for authorization queue name.

If two execution groups share the same truncated name, they share the same authorization queue. For example, if execution group names of two execution groups are 40 characters, but the first 30 characters are the same, they would share the same queue.

Important

The queue names must exactly match the case of the execution group name. This is not a problem when WebSphere Message Broker creates the queues; it is handled automatically. It might be a problem if the administrator wants to create the queues in advance or the queues are renamed.

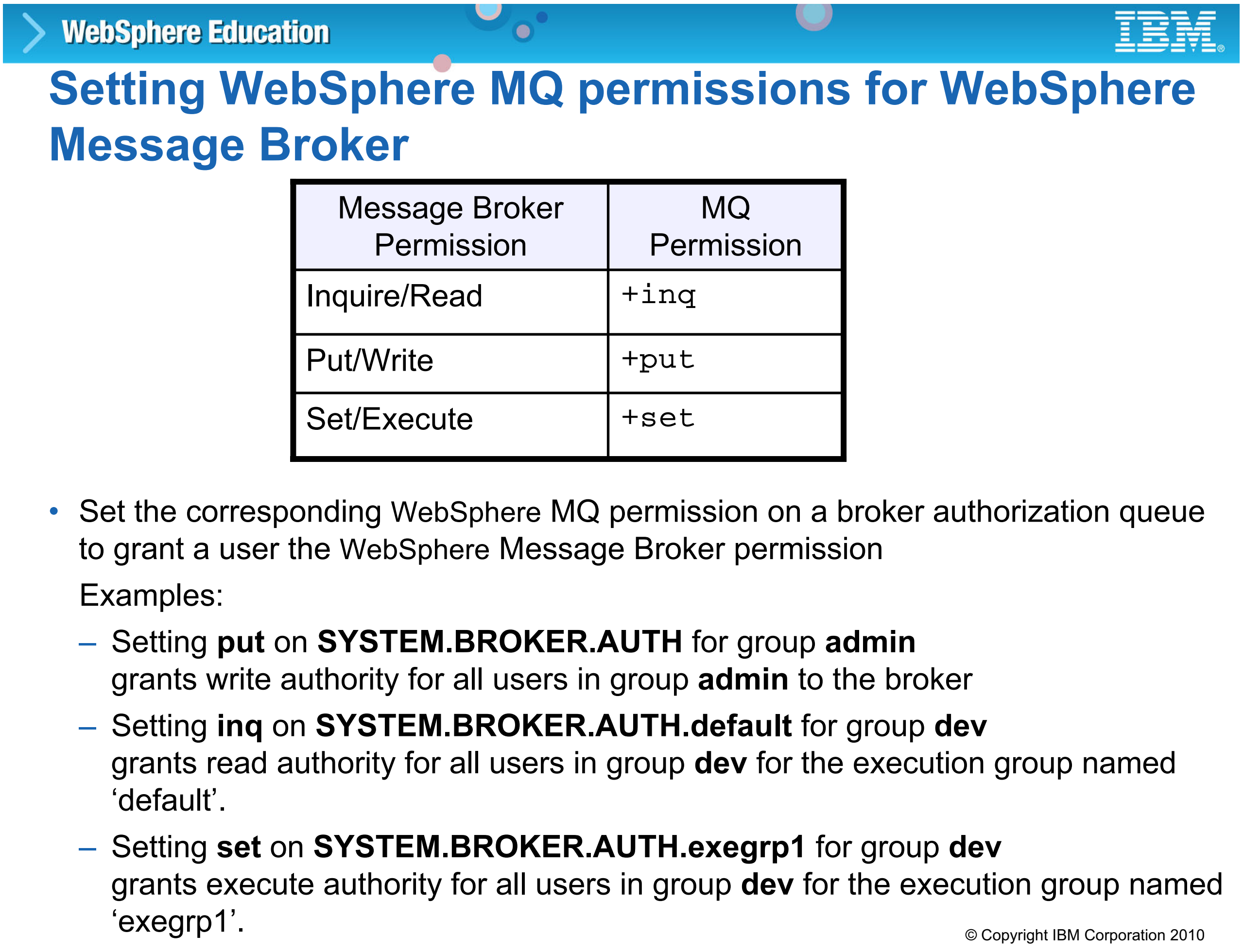
Instructor notes:

Purpose — Describe limitations on queue names for broker and execution group authorization queues.

Details —

Additional information —

Transition statement — Next: Setting MQ permissions for Message Broker



WebSphere Education
IBM

Setting WebSphere MQ permissions for WebSphere Message Broker

Message Broker Permission	MQ Permission
Inquire/Read	`+inq`
Put/Write	`+put`
Set/Execute	`+set`

- Set the corresponding WebSphere MQ permission on a broker authorization queue to grant a user the WebSphere Message Broker permission

  Examples:

  - Setting **put** on **SYSTEM.BROKER.AUTH** for group **admin** grants write authority for all users in group **admin** to the broker
  - Setting **inq** on **SYSTEM.BROKER.AUTH.default** for group **dev** grants read authority for all users in group **dev** for the execution group named ‘default’.
  - Setting **set** on **SYSTEM.BROKER.AUTH.exegrp1** for group **dev** grants execute authority for all users in group **dev** for the execution group named ‘exegrp1’.

Figure 5-17. Setting MQ permissions for Message Broker WM643 / VM6431.0

Notes:

This table shows the mapping between WebSphere Message Broker permissions and WebSphere MQ permissions:

- Read equals ‘+inq’
- Write equals ‘+put’
- Execute equals ‘+set’.

For example, to grant write authority to the broker for all users in the `admin` group, enable ‘put’ on the SYSTEM.BROKER.AUTH queue using the WebSphere MQ authorization commands, WebSphere MQ Explorer, or WebSphere Message Broker Explorer.

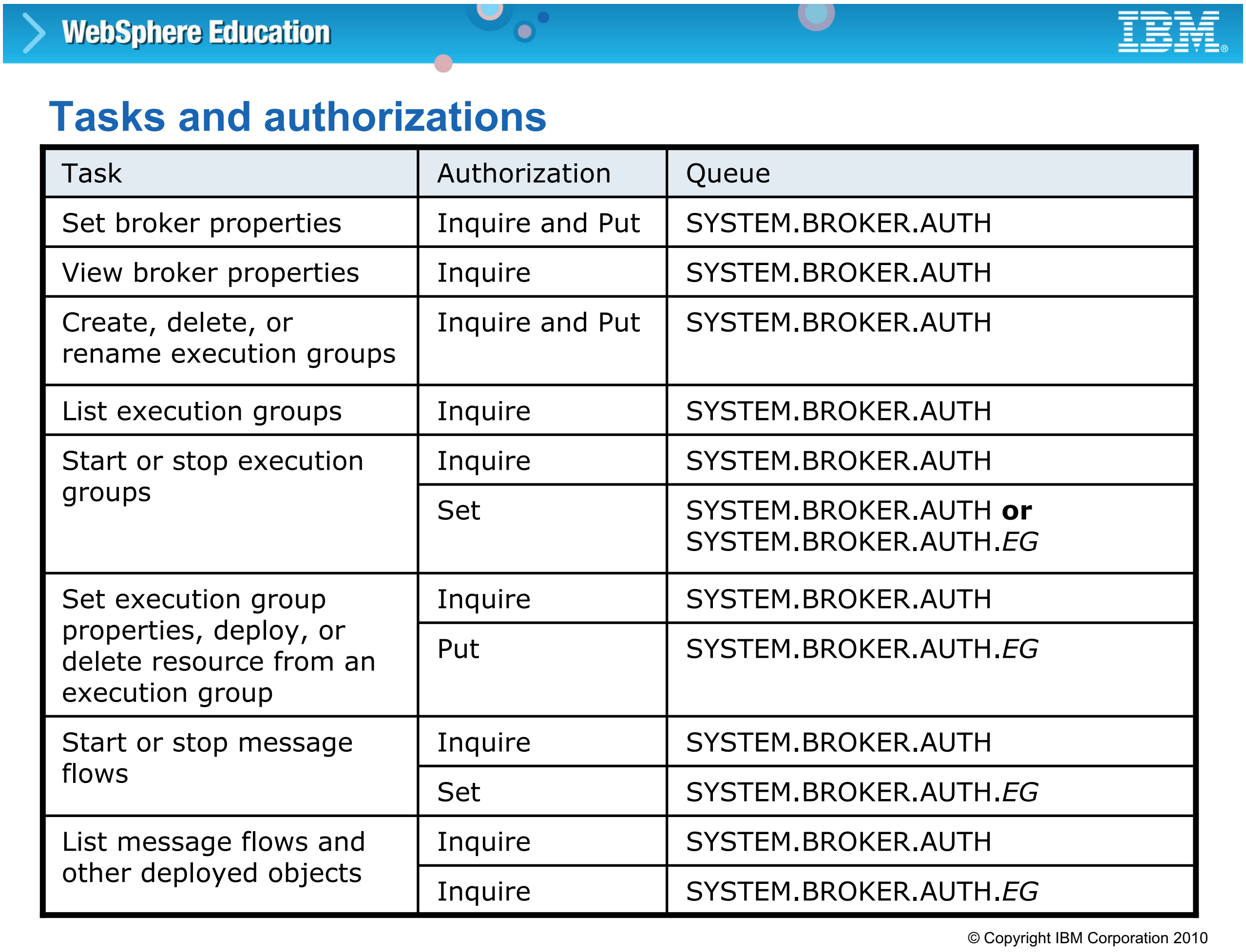
Instructor notes:

Purpose — Show the relationship between WebSphere Message Broker permissions and WebSphere MQ permissions.

Details —

Additional information —

Transition statement — The table on the next page provides some guidance on setting permissions.



WebSphere Education

Tasks and authorizations

Task	Authorization	Queue
Set broker properties	Inquire and Put	SYSTEM.BROKER.AUTH
View broker properties	Inquire	SYSTEM.BROKER.AUTH
Create, delete, or rename execution groups	Inquire and Put	SYSTEM.BROKER.AUTH
List execution groups	Inquire	SYSTEM.BROKER.AUTH
Start or stop execution groups	Inquire	SYSTEM.BROKER.AUTH
	Set	SYSTEM.BROKER.AUTH **or** SYSTEM.BROKER.AUTH.*EG*
Set execution group properties, deploy, or delete resource from an execution group	Inquire	SYSTEM.BROKER.AUTH
	Put	SYSTEM.BROKER.AUTH.*EG*
Start or stop message flows	Inquire	SYSTEM.BROKER.AUTH
	Set	SYSTEM.BROKER.AUTH.*EG*
List message flows and other deployed objects	Inquire	SYSTEM.BROKER.AUTH
	Inquire	SYSTEM.BROKER.AUTH.*EG*

© Copyright IBM Corporation 2010

Figure 5-18. Tasks and authorizations WM643 / VM6431.0

Notes:

The table in the figure lists some common tasks and the authorization that must be set on the queue.

EG in the table represents execution group. For example, if a developer must be able to start and stop message flows on an execution group named `default`, Set permission must be enabled on the SYSTEM.BROKER.AUTH.default queue.

See the WebSphere Message Broker Information Center for a complete list of tasks and permissions.

Instructor notes:

Purpose — List some common tasks and permission requirements.

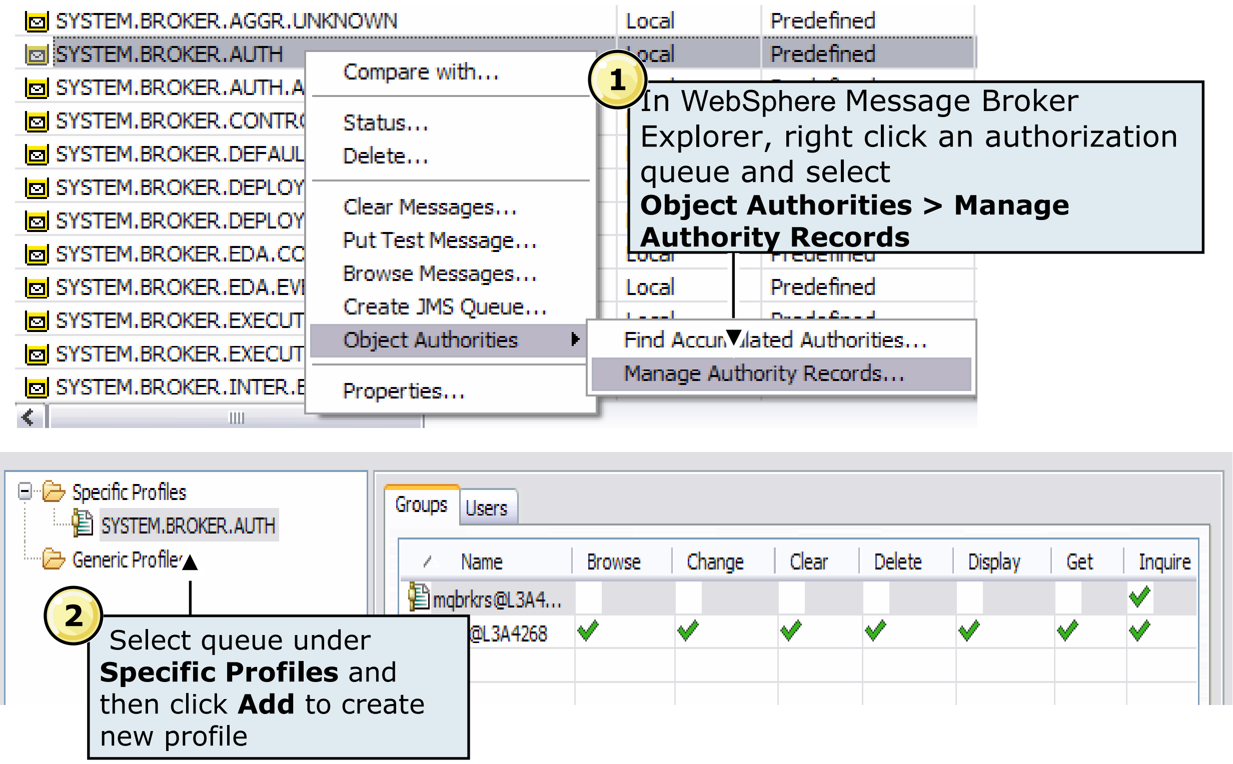
Details — Students will change permissions and test the effect of the change in the lab exercise for this unit.

Additional information —

Transition statement — You can create and modify authorization permissions in WebSphere Message Broker Explorer.

WebSphere Education IBM.

Creating authorization permissions (1 of 2)



Name	Browse	Change	Clear	Delete	Display	Get	Inquire
mqbrkrs@L3A4...							✓
@L3A4268	✓	✓	✓	✓	✓	✓	✓

1 In WebSphere Message Broker Explorer, right click an authorization queue and select **Object Authorities > Manage Authority Records**

2 Select queue under **Specific Profiles** and then click **Add** to create new profile

© Copyright IBM Corporation 2010

Figure 5-19. Creating authorization permissions (1 of 2) WM643 / VM6431.0

Notes:

WebSphere Message Broker Explorer and WebSphere MQ Explorer can be used to set authorization permissions, as shown here.

The screen capture on the top shows a partial list of all the queues starting with SYSTEM.BROKER.AUTH.

To set authorization on specific queue:

1. Right-click the queue name and then select **Object Authorities > Manage Authority Records**.
2. The security profile for the selected queue is displayed. The screen capture on the bottom of the figure shows the permissions page for SYSTEM.BROKER.AUTH.

   From this page you can either add users or groups to the authorization profile, delete, or modify existing permissions.

Instructor notes:

Purpose — Show how to launch security authorization window in WebSphere Message Broker Explorer.

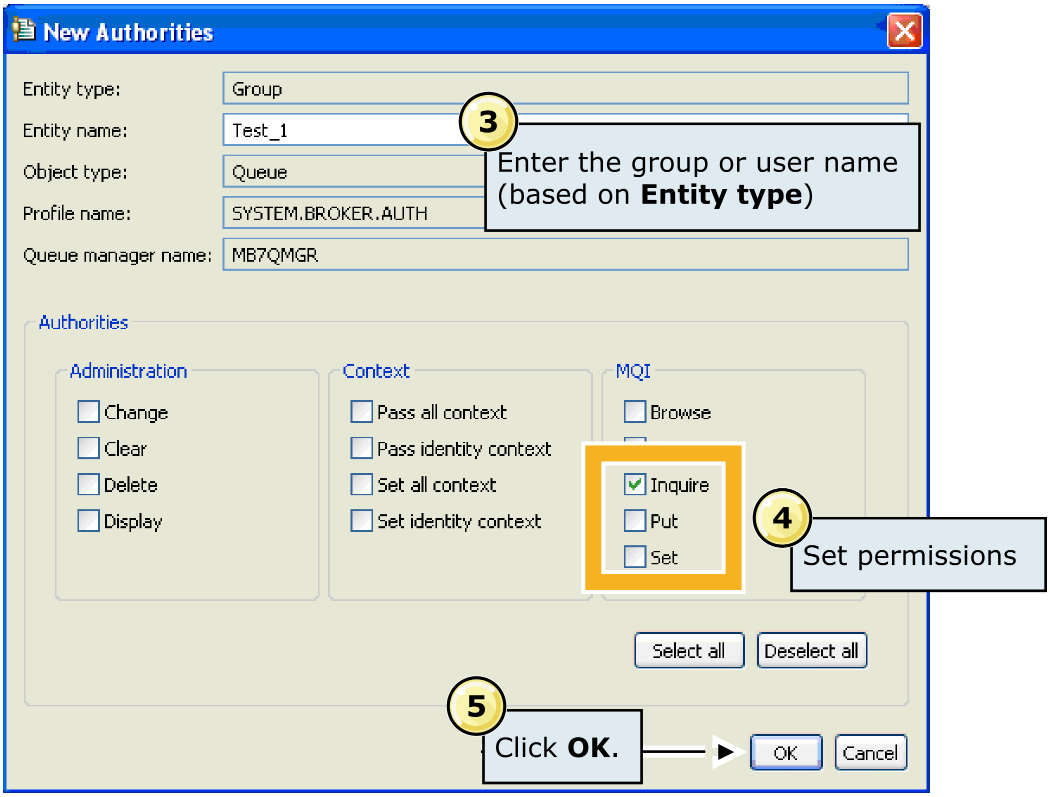
Details — Open the authority records for the SYSTEM.BROKER.AUTH queue. Highlight the Users and Groups tabs allowing administrators to set permissions by groups or by users.

Additional information —

Transition statement — You can modify existing permissions or add new authorities.

WebSphere Education

Creating authorization permissions (1 of 2)



New Authorities

Entity type: Group

Entity name: Test_1

Object type: Queue

Profile name: SYSTEM.BROKER.AUTH

Queue manager name: MB7QMGR

Authorities

Administration
- □ Change
- □ Clear
- □ Delete
- □ Display

Context
- □ Pass all context
- □ Pass identity context
- □ Set all context
- □ Set identity context

MQI
- □ Browse
- ☑ Inquire
- □ Put
- □ Set

3 Enter the group or user name (based on **Entity type**)

4 Set permissions

5 Click **OK**.

Figure 5-20. Creating authorization permissions (1 of 2) WM643 / VM6431.0

Notes:

The next step is to define the authorities. You can assign authority by group or by user.

3. The **Entity type** field identifies the entity as a group or a user.

   If you are adding a new group, specify the group name in the **Entity name** field.

   If you are adding a new user, enter the user name in the **Entity name** field.

4. Set the permissions. WebSphere Message Broker permissions are set under the MQI column.

5. Click **OK** to save the changes. The changes are immediate; there is no need to restart the broker.

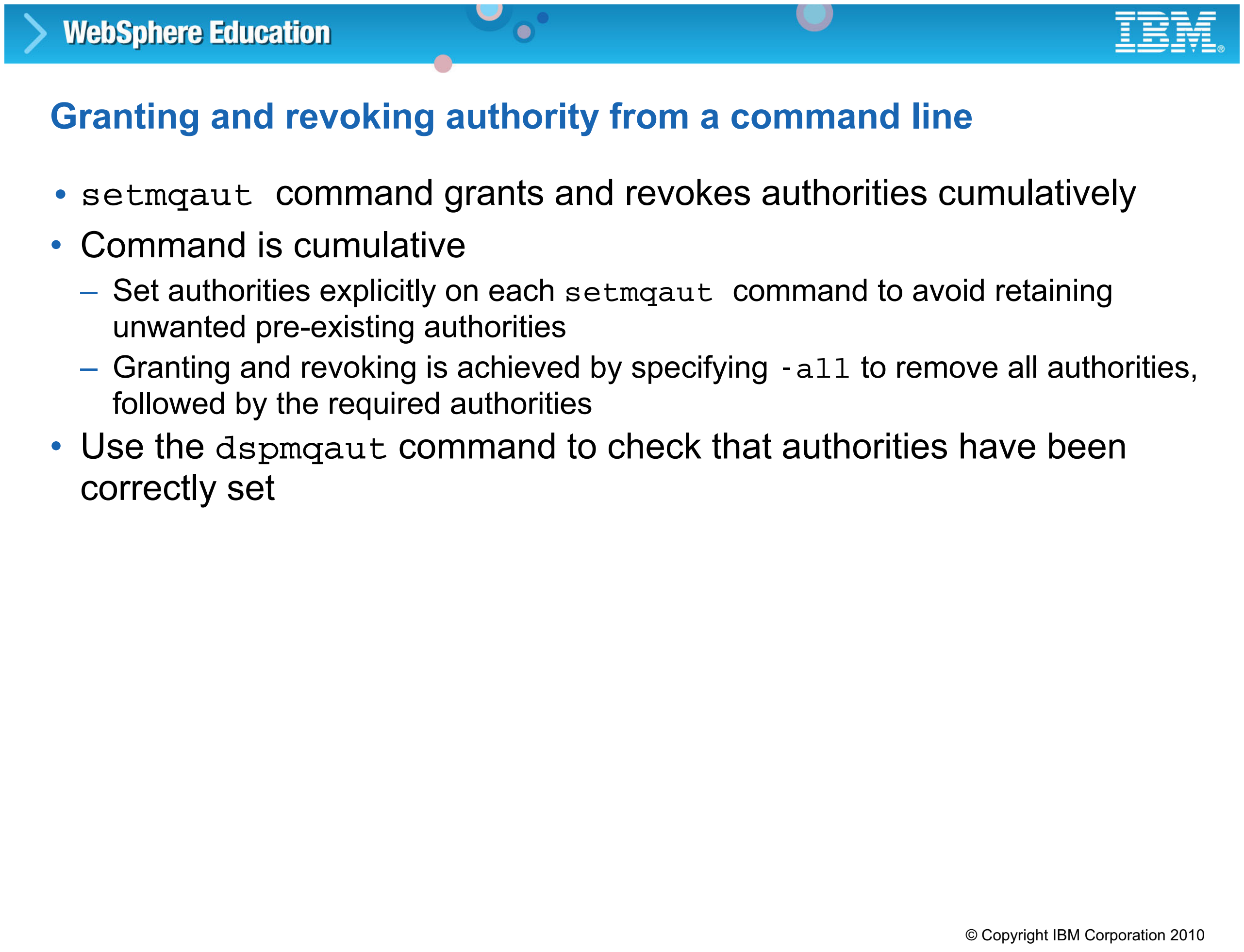
Instructor notes:

Purpose — Show the steps for setting authority permissions.

Details — Demonstrate setting permissions. Students will perform these steps in the lab exercise.

Additional information — None.

Transition statement — WebSphere Message Broker Explorer does not run on all platforms. It might be necessary to set permissions from a command line.



WebSphere Education

IBM

Granting and revoking authority from a command line

- `setmqaut` command grants and revokes authorities cumulatively
- Command is cumulative
  - Set authorities explicitly on each `setmqaut` command to avoid retaining unwanted pre-existing authorities
  - Granting and revoking is achieved by specifying `-all` to remove all authorities, followed by the required authorities
- Use the `dspmqaut` command to check that authorities have been correctly set

Figure 5-21. Granting and revoking authority from a command line WM643 / VM6431.0

Notes:

You can use the `setmqaut` command to grant and revoke permissions; however, commands are cumulative so always use the `dspmqaut` command to check that the authorities are correct.

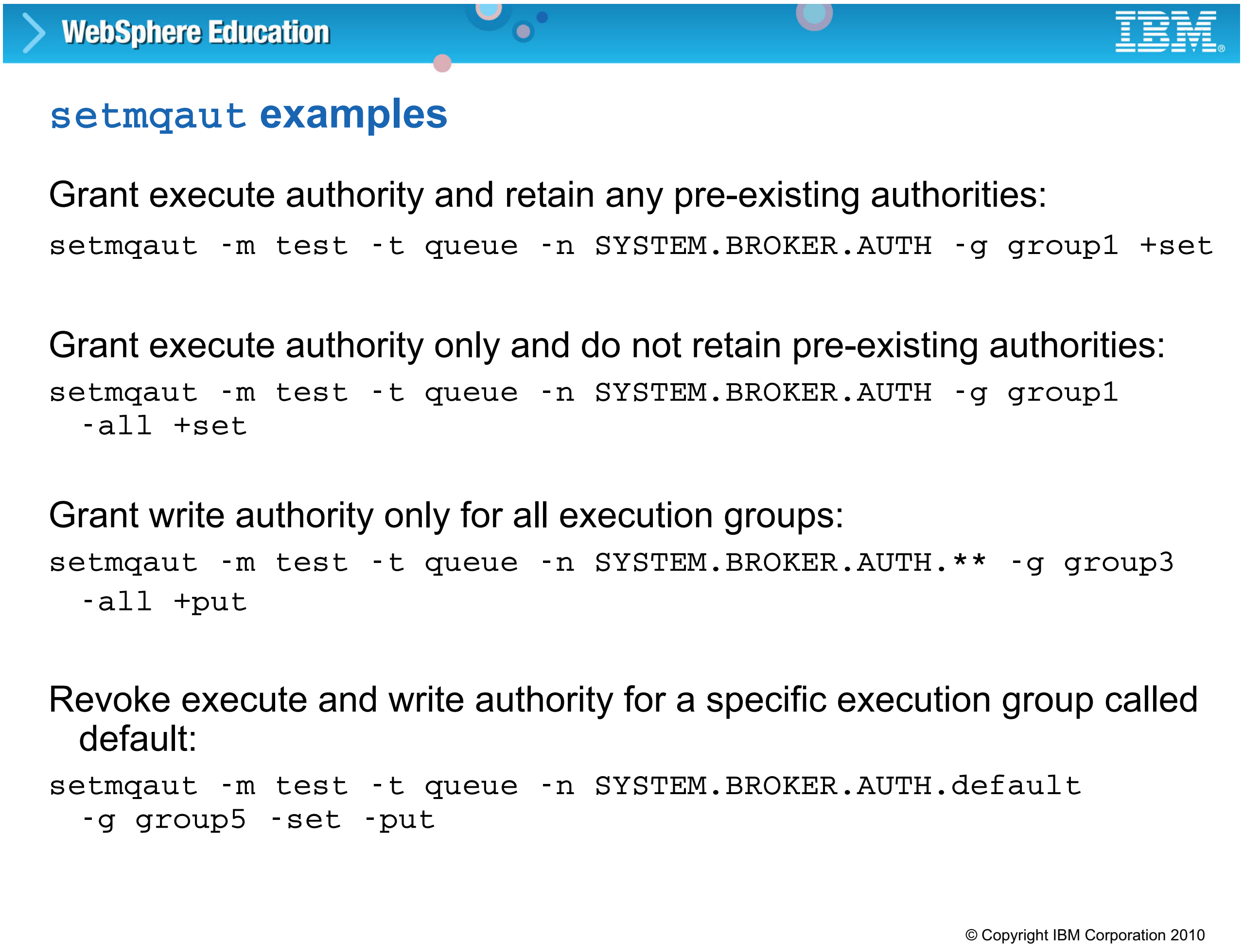
Instructor notes:

Purpose — Describe the `setmqaut` command for setting authority permissions.

Details —

Additional information —

Transition statement — Look at some examples of the `setmqaut` command.



WebSphere Education

setmqaut examples

Grant execute authority and retain any pre-existing authorities:

```
setmqaut -m test -t queue -n SYSTEM.BROKER.AUTH -g group1 +set
```

Grant execute authority only and do not retain pre-existing authorities:

```
setmqaut -m test -t queue -n SYSTEM.BROKER.AUTH -g group1
  -all +set
```

Grant write authority only for all execution groups:

```
setmqaut -m test -t queue -n SYSTEM.BROKER.AUTH.** -g group3
  -all +put
```

Revoke execute and write authority for a specific execution group called default:

```
setmqaut -m test -t queue -n SYSTEM.BROKER.AUTH.default
  -g group5 -set -put
```

Figure 5-22. setmqaut examples WM643 / VM6431.0

Notes:

The figure shows a number of examples for using the setmqaut command to set authority permissions.

A plus sign before the authorization, adds the permission. For example `+set` grants Set (execute) authority.

A minus sign before the permission, removes the permission. For example, -put removes Put (write) authority.

The options shown for the `setmqaut` command are:

`-m` is queue manager name
`-t` is type
`-n` is the name of queue to change
`-g` is the group name
`-p` is the principal
`-all` to remove all authorities before granting the authorities that follow

Instructor notes:

Purpose — Show some examples of using the `setmqaut` command to set broker authority from a command line.

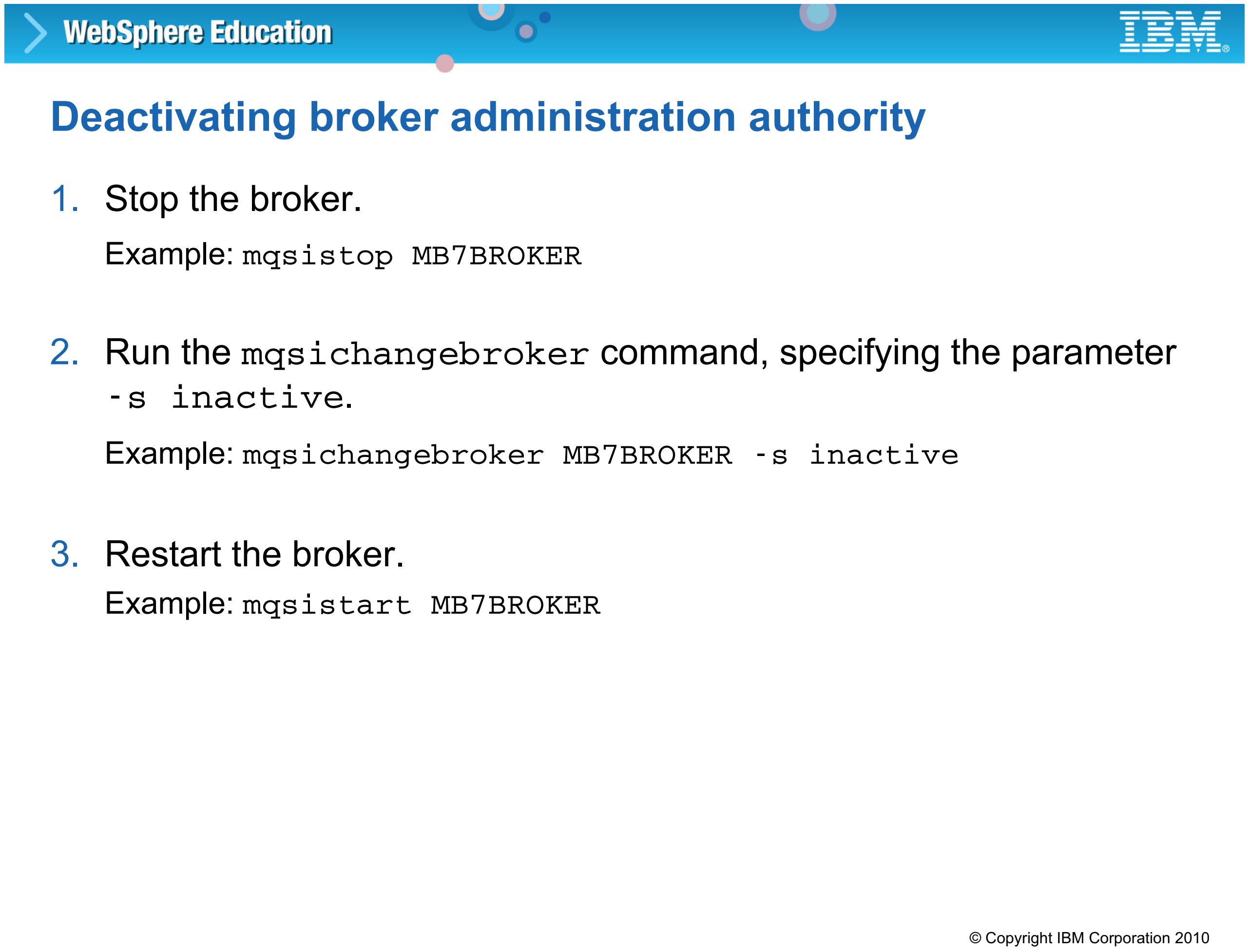
Details — To avoid retaining unwanted pre-existing authorities, set authorities explicitly on each `setmqaut` command, rather than granting and revoking individual authorities.

Granting and revoking is achieved by specifying `-all` (to remove all authorities) followed by the required authorities.

If time permits, demonstrate this command to show the cumulative effects of setting permissions. Also mention that this command can be included in a script or batch file allowing administrators to easily set permissions as new users are added.

Additional information — None.

Transition statement — An administrator can deactivate broker security at any time.


WebSphere Education

Deactivating broker administration authority

1. Stop the broker.

   Example: `mqsistop MB7BROKER`

2. Run the `mqsichangebroker` command, specifying the parameter `-s inactive`.

   Example: `mqsichangebroker MB7BROKER -s inactive`

3. Restart the broker.

   Example: `mqsistart MB7BROKER`

Figure 5-23. Deactivating broker administration authority WM643 / VM6431.0

Notes:

When you deactivate broker administration security, the authorization queues that are associated with this broker are retained. If you activate security again, these queues are reused, therefore you must ensure that the authorizations defined by these queues are correct.

Instructor notes:

Purpose — List the steps for deactivating broker authority on Linux, UNIX, and Windows.

Details —

Additional information —

Transition statement — WebSphere Message Broker provides a security manager, with which you can control access to individual messages in a message flow, using the identity of the message.

5.3. Message flow security

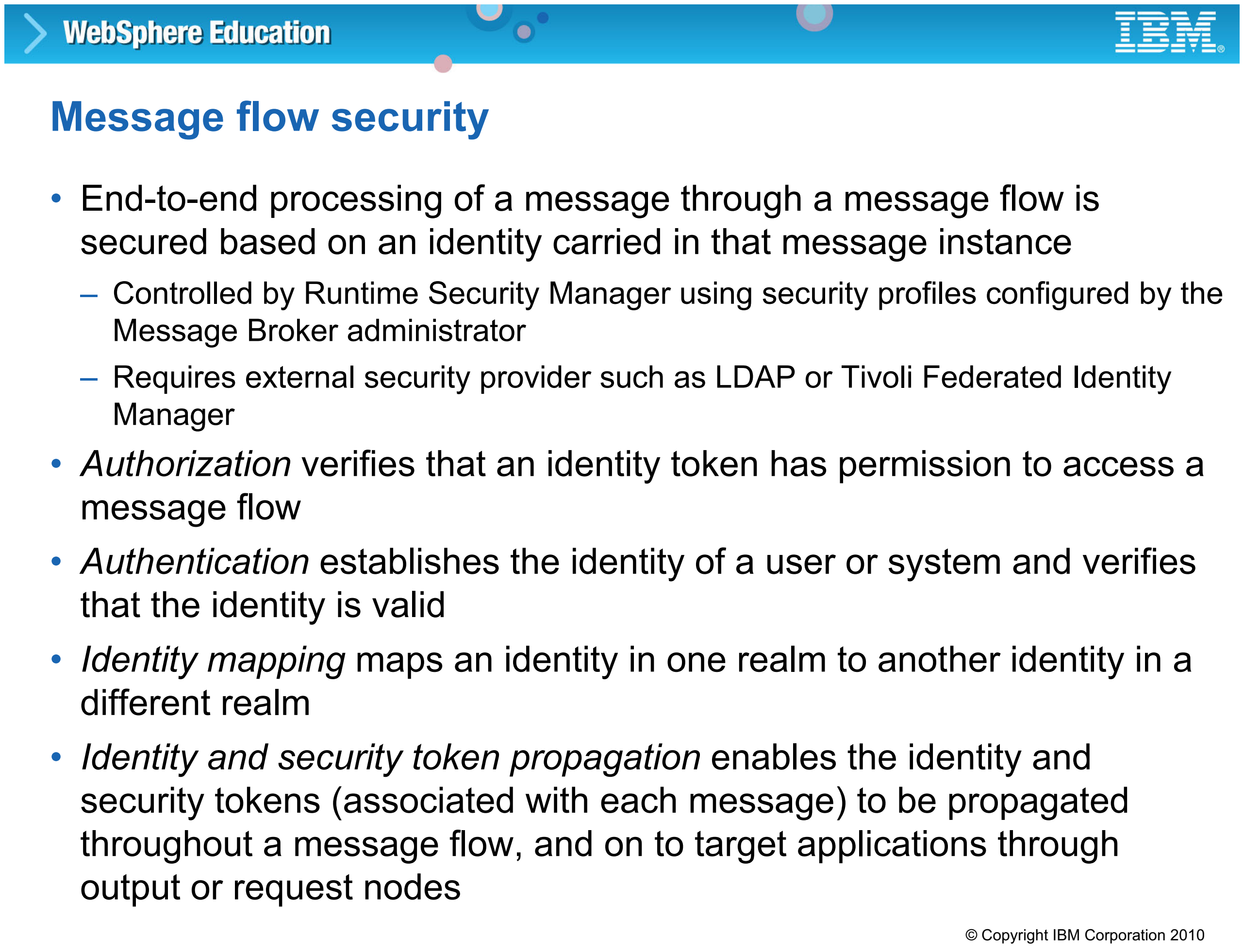
Instructor topic introduction

What students will do — Learn how to implement and administer message flow security.

How students will do it — By listening to lecture, answering checkpoint questions, and completing a hands-on lab exercise.

What students will learn — Students will learn the options for implementing message flow security.

How this will help students on their job — Administrators will be able to identify points of security and implement message flow security.



WebSphere Education
IBM

Message flow security

- End-to-end processing of a message through a message flow is secured based on an identity carried in that message instance
  - Controlled by Runtime Security Manager using security profiles configured by the Message Broker administrator
  - Requires external security provider such as LDAP or Tivoli Federated Identity Manager
- *Authorization* verifies that an identity token has permission to access a message flow
- *Authentication* establishes the identity of a user or system and verifies that the identity is valid
- *Identity mapping* maps an identity in one realm to another identity in a different realm
- *Identity and security token propagation* enables the identity and security tokens (associated with each message) to be propagated throughout a message flow, and on to target applications through output or request nodes

Figure 5-24. Message flow security WM643 / VM6431.0

Notes:

WebSphere Message Broker provides a security manager, which enables you to control access to individual messages in a message flow, using the identity of the message.

You can configure the broker to perform end-to-end processing of an identity carried in a message through a message flow. This security manager can:

- Extract the identity from an inbound message.
- Authenticate the identity (using an external security provider).
- Map the identity to an alternative identity (using an external security provider).
- Check that either the alternative identity or the original identity is authorized to access the message flow (using an external security provider).
- Propagate either the alternative identity or the original identity with an outbound message.

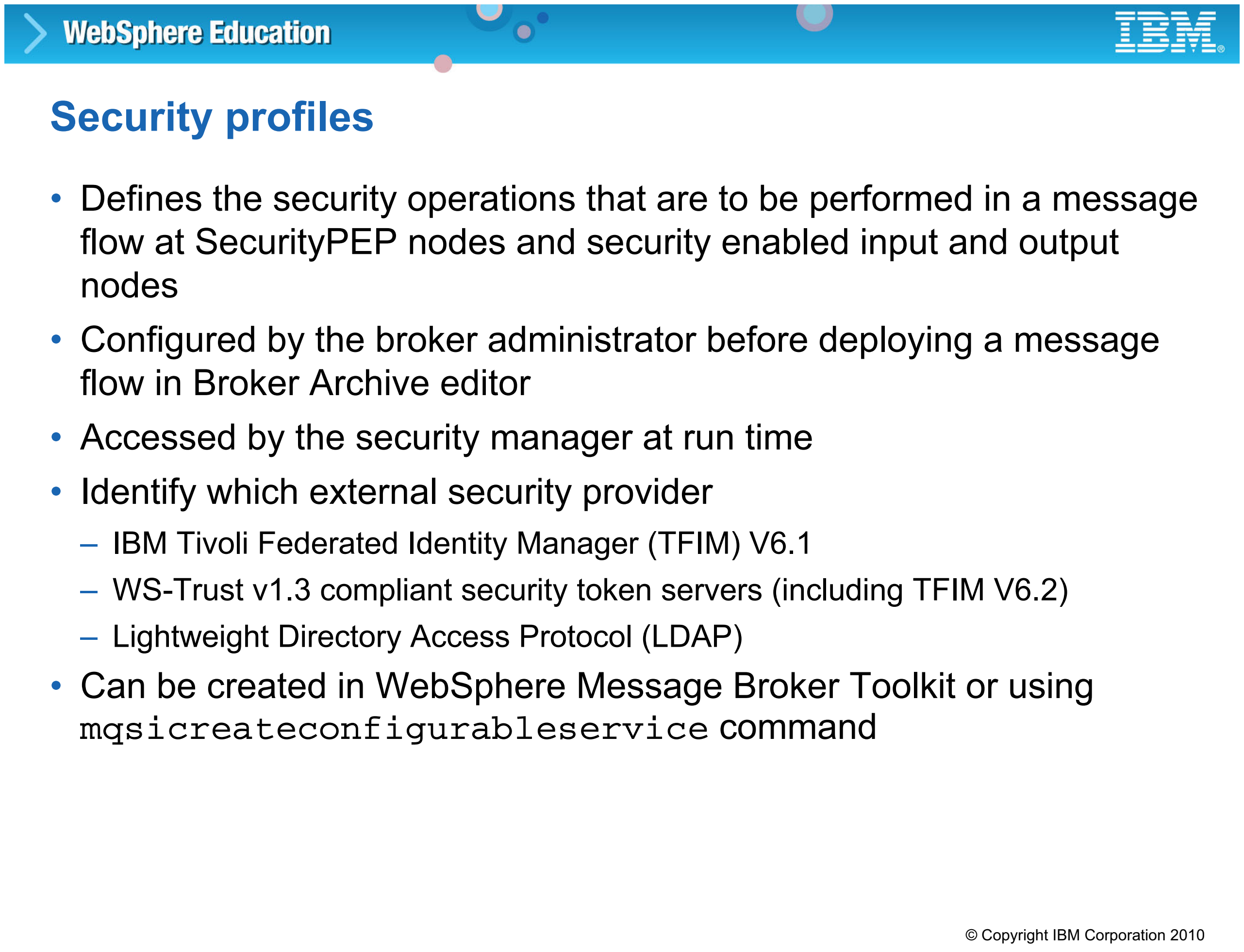
Instructor notes:

Purpose — Provide an overview of message flow security

Details —

Additional information —

Transition statement — The security functions that are associated with a message flow are controlled by using security profiles.



WebSphere Education

Security profiles

- Defines the security operations that are to be performed in a message flow at SecurityPEP nodes and security enabled input and output nodes
- Configured by the broker administrator before deploying a message flow in Broker Archive editor
- Accessed by the security manager at run time
- Identify which external security provider
  - IBM Tivoli Federated Identity Manager (TFIM) V6.1
  - WS-Trust v1.3 compliant security token servers (including TFIM V6.2)
  - Lightweight Directory Access Protocol (LDAP)
- Can be created in WebSphere Message Broker Toolkit or using `mqsicreateconfigurableservice` command

Figure 5-25. Security profiles WM643 / VM6431.0

Notes:

Security profiles are created by the broker administrator and accessed by the security manager at run time.

The following external security providers are supported:

- IBM Tivoli Federated Identity Manager V6.1 for authentication, mapping, and authorization
- WS-Trust V1.3 compliant security token servers (including Tivoli Federated Identity Manager V6.2) for authentication, mapping, and authorization
- Lightweight Directory Access Protocol (LDAP) for authentication and authorization

Security profiles apply to the SecurityPEP node and to security enabled input, output, and request nodes, and are configured by the administrator at deployment time in the Broker Archive editor or using the `mqsicreateconfigurableservice` command.

Instructor notes:

Purpose — Describe a security profile and the supported security managers.

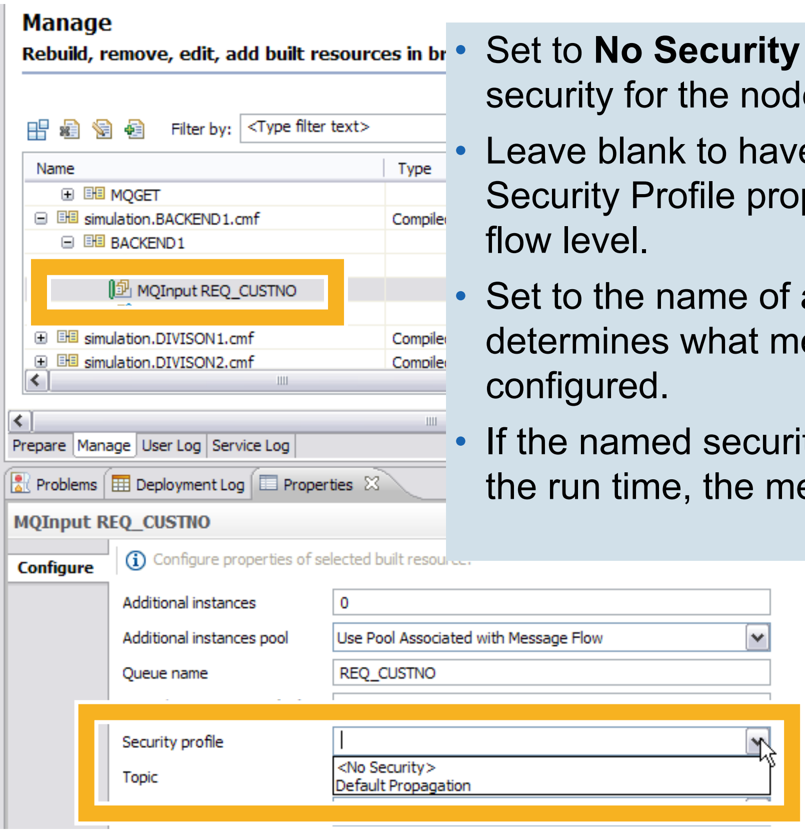
Details — Security profiles apply to the SecurityPEP node and to security enabled input, output, and request nodes, and are configured by the administrator at deployment time in the Broker Archive editor.

Additional information — None.

Transition statement — Nodes that support security profiles have a **Security Profile** property.

WebSphere Education

Security profile property



Additional instances: 0

Additional instances pool: Use Pool Associated with Message Flow

Queue name: REQ_CUSTNO

Security profile
<No Security>
Default Propagation

Topic

- Set to **No Security** to explicitly turn off security for the node.
- Leave blank to have the node inherit the Security Profile property set at the message flow level.
- Set to the name of a specific security profile determines what message flow security is configured.
- If the named security profile does not exist in the run time, the message flow fails to deploy

Figure 5-26. Security profile property WM643 / VM6431.0

Notes:

Nodes that support runtime security have a **Security profile** property which can be left blank, set to **No Security**, or set to a specific security profile name.

- Set **Security profile** to **No Security** to explicitly turn off security for the node.
- If you leave the **Security profile** property blank, the node inherits the security profile property that is set at the message flow level. If you leave the **Security profile** property blank at both levels, security is turned off for the node.
- When **Security profile** is set to the name of a specific security profile, that profile determines what message flow security is configured. If the named security profile does not exist in the run time, the message flow fails to deploy. If you want to extract and propagate an identity without security enforcement or mapping, you can use the supplied security profile called **Default Propagation**.

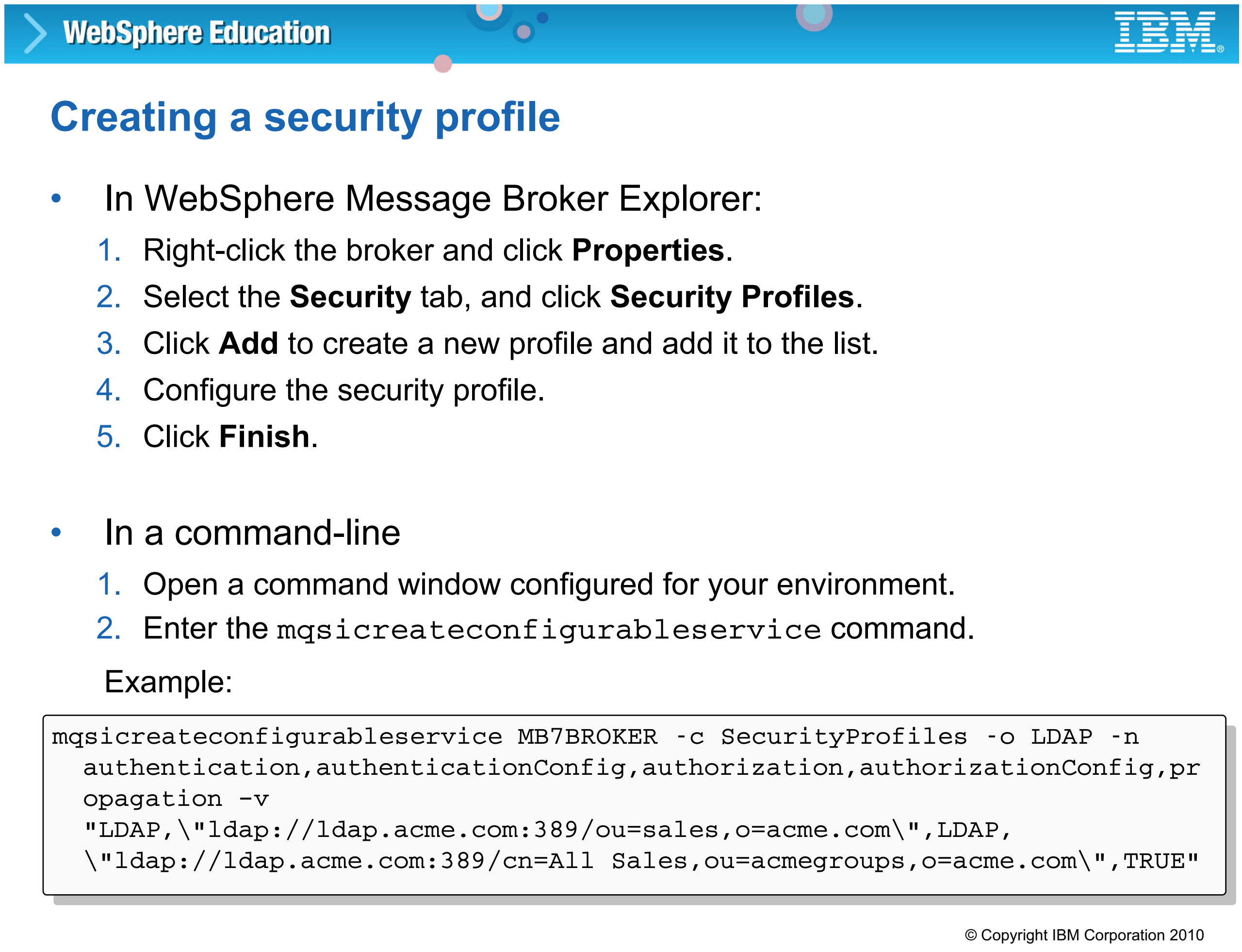
Instructor notes:

Purpose — Shown an example of the Security profile property in a node.

Details — The Default Propagation profile is a predefined profile that requests only identity propagation. Identity and security token propagation is described later in this topic.

Additional information — None.

Transition statement — A security profile can be created in WebSphere Message Broker Explorer or in a command line.



WebSphere Education

Creating a security profile

- In WebSphere Message Broker Explorer:
  1. Right-click the broker and click **Properties**.
  2. Select the **Security** tab, and click **Security Profiles**.
  3. Click **Add** to create a new profile and add it to the list.
  4. Configure the security profile.
  5. Click **Finish**.

- In a command-line
  1. Open a command window configured for your environment.
  2. Enter the `mqsicreateconfigurableservice` command.

  Example:

```
mqsicreateconfigurableservice MB7BROKER -c SecurityProfiles -o LDAP -n
  authentication,authenticationConfig,authorization,authorizationConfig,pr
  opagation -v
  "LDAP,\"ldap://ldap.acme.com:389/ou=sales,o=acme.com\",LDAP,
  \"ldap://ldap.acme.com:389/cn=All Sales,ou=acmegroups,o=acme.com\",TRUE"
```

© Copyright IBM Corporation 2010

Figure 5-27. Creating a security profile WM643 / VM6431.0

Notes:

You can create a security profile on using WebSphere Message Broker Explorer or using the `mqsicreateconfigurableservice` command line.

The command-line example shows an example of the syntax for creating an LDAP-based security profile. The arguments after -n are the property names separated by a comma. The arguments after -v are the values for each property separate by comma. The example sets the `authentication` property to `LDAP`, the `authenticationConfig` property to `ldap://ldap.acme.com:389/ou=sales, o=acme.com\`, and so on.

Note

You must enclose the LDAP URL (which contains commas) with escaped double quotation marks (\" and \") so that the URL commas are not confused with the comma separator of the value parameter of `mqsicreateconfigurableservice`.

Instructor notes:

Purpose — List the high-level steps for creating a Security profile using WebSphere Message Broker Explorer and provide an example of using the mqsicreateconfigurableservice command to create an LDAP security profile.

Details — The `authenticationConfig` using the following syntax:

```
ldap[s]://server[:port]/baseDN[?[uid_attr][?[base|sub]]]
```

Open the Information Center and search for “Creating a security profile for LDAP”.

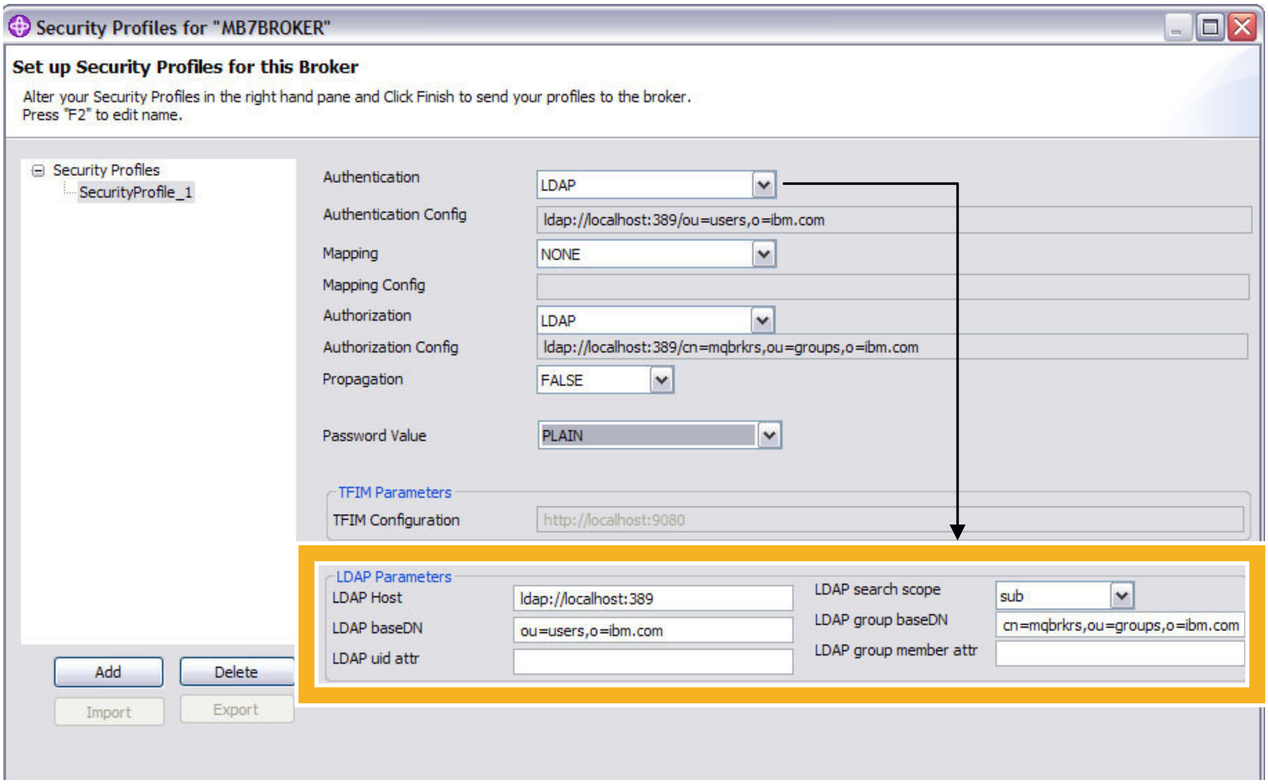
The next page has an example of the Security profile configuration options.

Additional information — None.

Transition statement — Next: Security profile configuration example

WebSphere Education

Security profile configuration example



Security Profiles for "MB7BROKER"

Set up Security Profiles for this Broker

Alter your Security Profiles in the right hand pane and Click Finish to send your profiles to the broker.
Press "F2" to edit name.

- Security Profiles
  - SecurityProfile_1

Authentication: LDAP
Authentication Config: ldap://localhost:389/ou=users,o=ibm.com
Mapping: NONE
Mapping Config:
Authorization: LDAP
Authorization Config: ldap://localhost:389/cn=mqbrkrs,ou=groups,o=ibm.com
Propagation: FALSE
Password Value: PLAIN

TFIM Parameters
TFIM Configuration: http://localhost:9080

LDAP Parameters
LDAP Host: ldap://localhost:389
LDAP baseDN: ou=users,o=ibm.com
LDAP uid attr:
LDAP search scope: sub
LDAP group baseDN: cn=mqbrkrs,ou=groups,o=ibm.com
LDAP group member attr:

Figure 5-28. Security profile configuration example WM643 / VM6431.0

Notes:

The Security Profiles window contains a list of existing security profiles for the broker on the left, and a pane in which you can configure the profile on the right.

To create a security profile for LDAP, for example:

1. Select the type of **Authentication** to LDAP.
2. In the **Propagation** field, specify whether you require the identity to be propagated. The default is False.
3. In the **Password Value** field, select the way in which the password is displayed in the properties folder.
4. Edit the following fields in the **LDAP Parameters** section.

The values that you enter in the LDAP Parameters fields create a configuration string, which is displayed in the **Authorization Config** field.

Instructor notes:

Purpose — Show an example of the Security profile configuration dialog in WebSphere Message Broker Explorer. The options are set for LDAP.

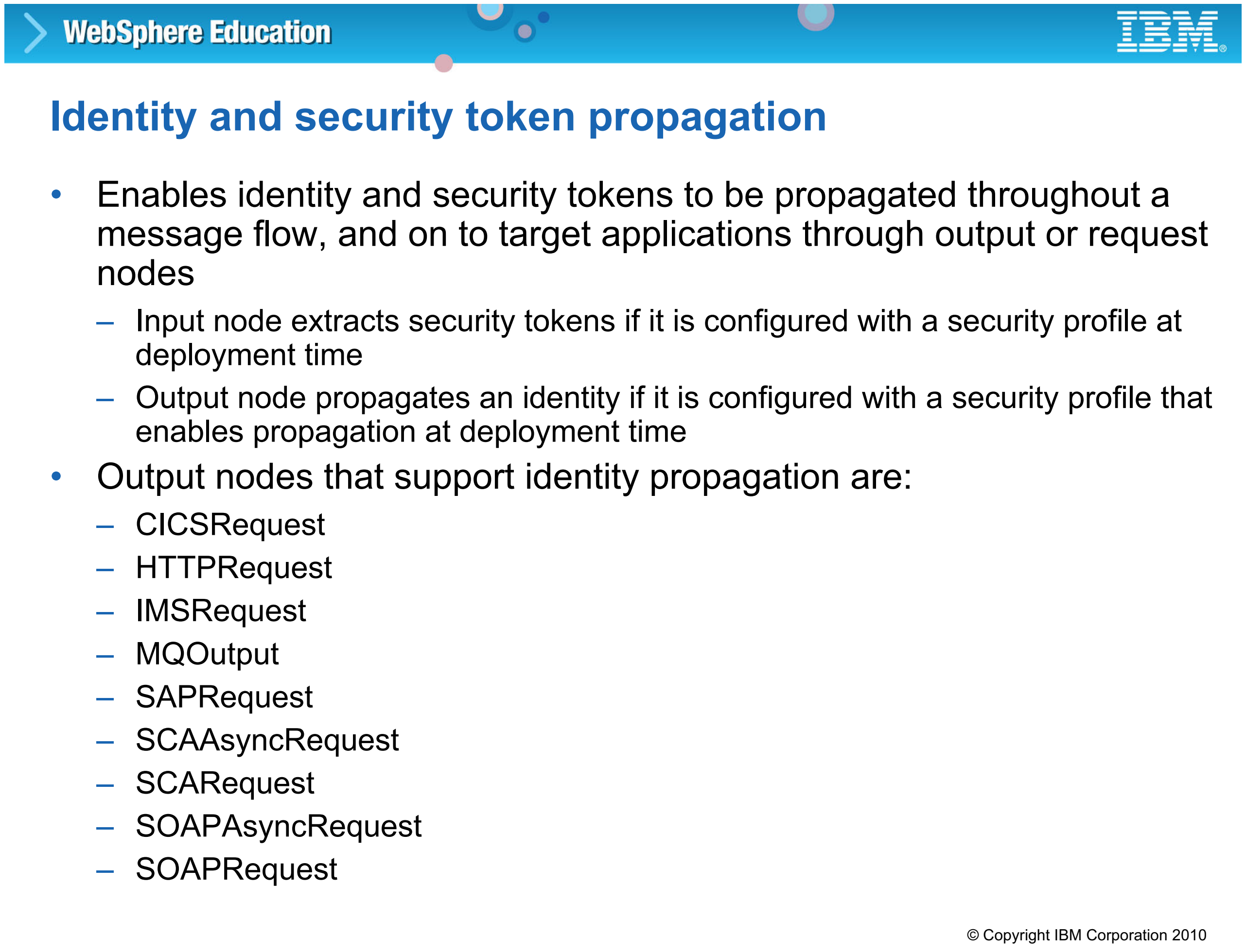
Details — The options for Password value are:

- **PLAIN:** The password is shown in the Properties folder as plain text.
- **OBFUSCATE:** The password is shown in the Properties folder as base64 encoding.
- **MASK:** The password is shown in the Properties folder as four asterisks (****).

Additional information — LDAP options and settings are usually determined by security personnel. You must have an LDAP server that is LDAP Version 3 compliant, for example:

- IBM® Tivoli® Directory Server
- Microsoft® Active Directory
- OpenLDAP.

Transition statement — WebSphere Message Broker also supports identify and security token propagation is enabled in the configuration by setting **Propagation** to True.



WebSphere Education

Identity and security token propagation

- Enables identity and security tokens to be propagated throughout a message flow, and on to target applications through output or request nodes
  - Input node extracts security tokens if it is configured with a security profile at deployment time
  - Output node propagates an identity if it is configured with a security profile that enables propagation at deployment time
- Output nodes that support identity propagation are:
  - CICSRequest
  - HTTPRequest
  - IMSRequest
  - MQOutput
  - SAPRequest
  - SCAAsyncRequest
  - SCARequest
  - SOAPAsyncRequest
  - SOAPRequest

Figure 5-29. Identity and security token propagation WM643 / VM6431.0

Notes:

The propagation of an identity ensures that the logical identity is kept throughout the system by mapping between the various physical forms as necessary. For example, a message might enter the system using a certificate, but a user name token might be required for server processing of the message. Identity mapping is used to convert from the certificate to the user name token, and identity propagation ensures that the mapped identity is placed in the correct place for the outbound transport.

The figure lists the output nodes that support identity propagation.

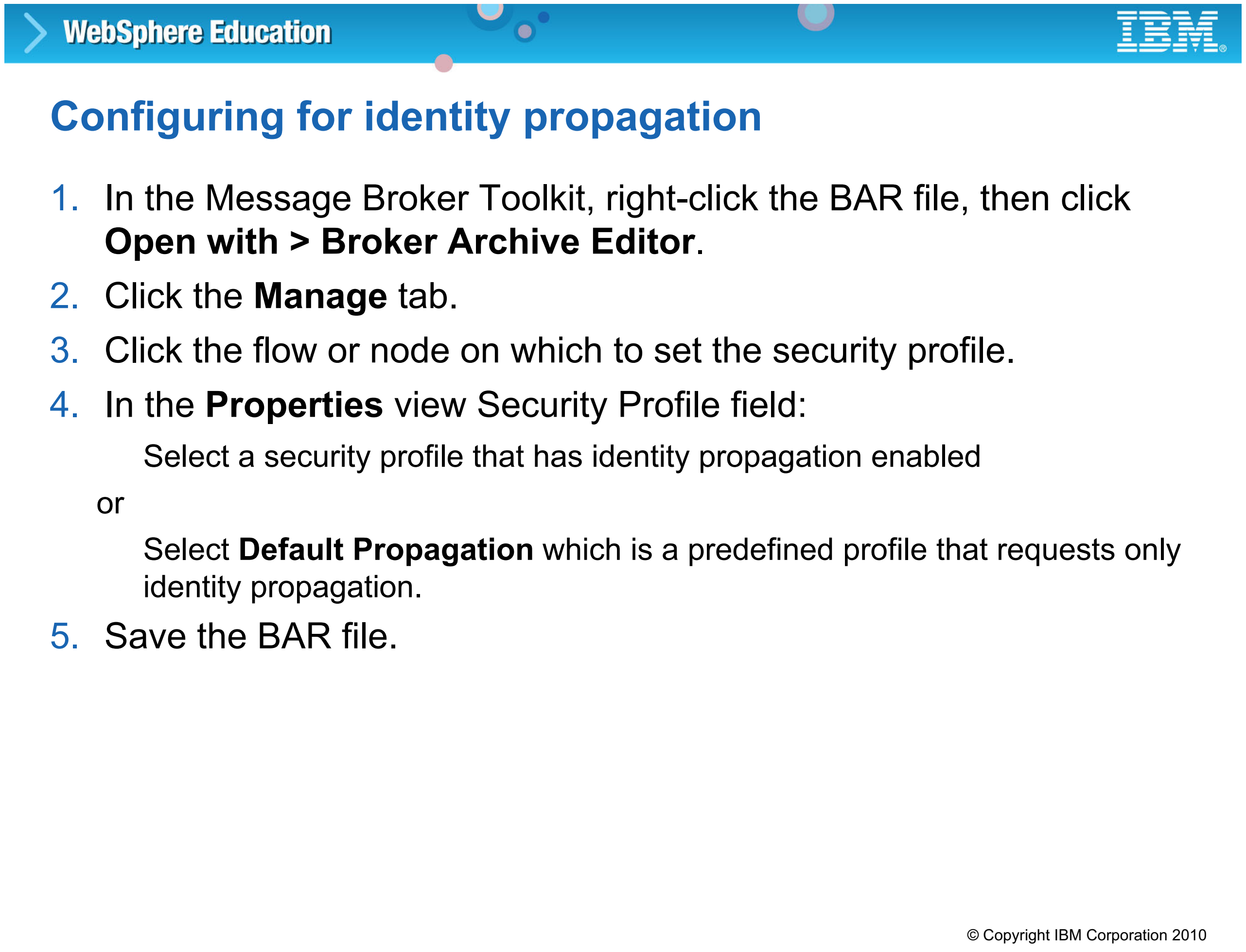
Instructor notes:

Purpose — Introduce the concept of identity and security token propagation.

Details — When an output or request node propagates an identity, the mapped identity is used. If the mapped identity is not set, or if it has a token type that is not supported by the node, the source identity is used. If no identity is set, or if the mapped and source identity do not have a token type that is supported by the node, a security exception is thrown by the node.

Additional information — This topic provides an overview to identity and security propagation only.

Transition statement — Next: Configuring for identity propagation



WebSphere Education

Configuring for identity propagation

1. In the Message Broker Toolkit, right-click the BAR file, then click **Open with > Broker Archive Editor**.
2. Click the **Manage** tab.
3. Click the flow or node on which to set the security profile.
4. In the **Properties** view Security Profile field:

   Select a security profile that has identity propagation enabled

   or

   Select **Default Propagation** which is a predefined profile that requests only identity propagation.
5. Save the BAR file.

Figure 5-30. Configuring for identity propagation WM643 / VM6431.0

Notes:

Before you can configure a message flow to perform identity propagation, you must check that an appropriate security profile exists, or create a security profile with propagation enabled. You can use the **Default Propagation** profile, which is a predefined profile that requests only identity propagation.

You can set a security profile on a message flow or on individual input and output nodes. If no security profile is set for the input and output nodes, the setting is inherited from the setting on the message flow.

The properties that you can configure for the message flow or for the node at run time are displayed in the **Properties** view in the WebSphere Message Broker Toolkit.

Instructor notes:

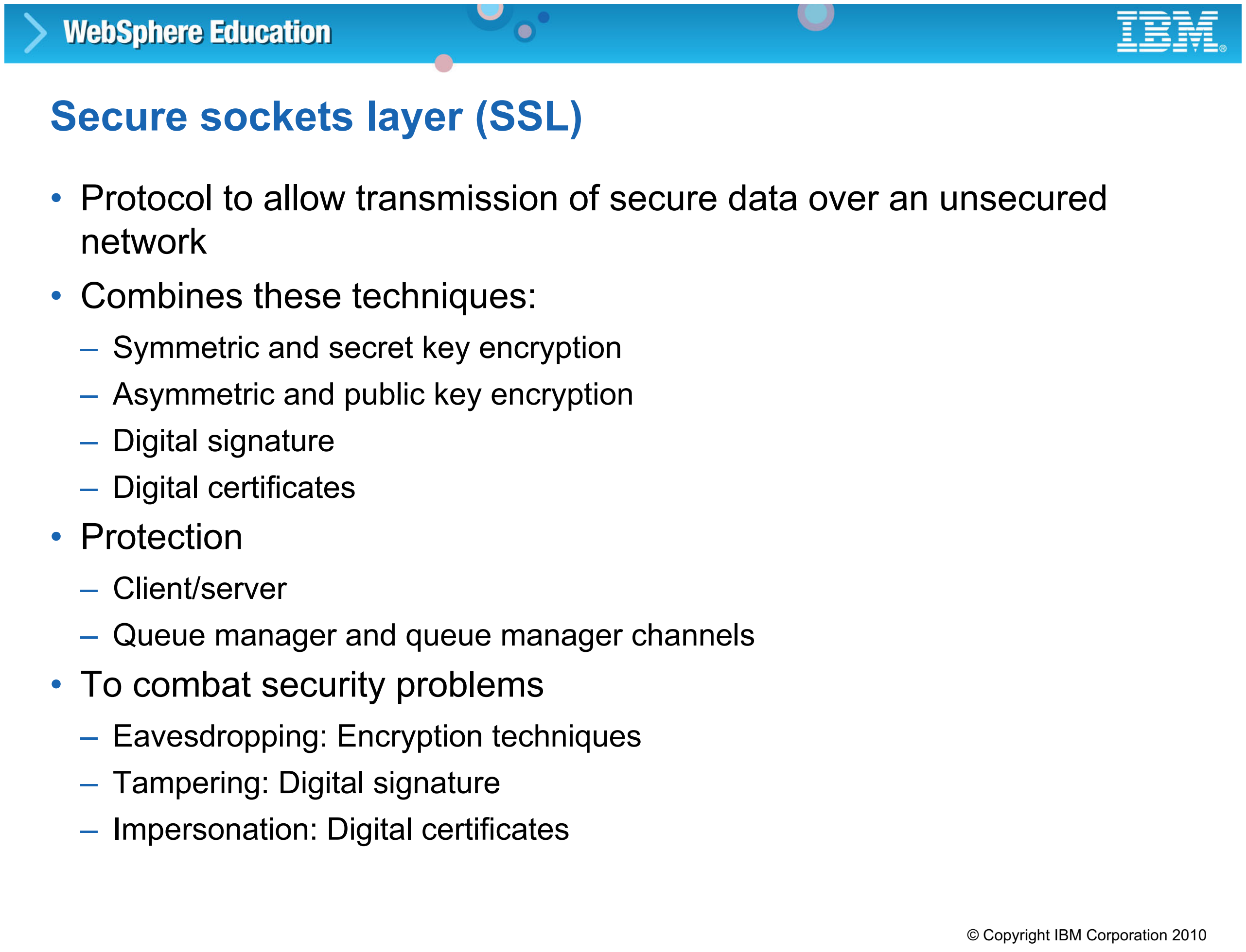
Purpose — List the steps for configuring for identify propagation in WebSphere Message Broker Toolkit.

Details — The Security profile configuration example slide can be used to show where the Propagation option is enabled in the security profile.

Additional information — If the message identity does not contain enough information for identity propagation, you can use any of the following methods to acquire the necessary information:

- Take the information from the message body. For example, if the message comes from WebSphere MQ with only a user name token, and the output is an HTTP request node requiring a Username + Password token, the password might be present in the body of the incoming message. For more information, see Configuring the extraction of an identity or security token.
- Configure an identity mapper using Tivoli Federated Identity Manager. For more information, see the IBM Tivoli Federated Identity Manager Information Center.
- Use ESQL or Java to set the Mapped Identity fields in the Properties tree.

Transition statement — Next: Secure sockets layer (SSL)



WebSphere Education

Secure sockets layer (SSL)

- Protocol to allow transmission of secure data over an unsecured network
- Combines these techniques:
  - Symmetric and secret key encryption
  - Asymmetric and public key encryption
  - Digital signature
  - Digital certificates
- Protection
  - Client/server
  - Queue manager and queue manager channels
- To combat security problems
  - Eavesdropping: Encryption techniques
  - Tampering: Digital signature
  - Impersonation: Digital certificates

Figure 5-31. Secure sockets layer (SSL) WM643 / VM6431.0

Notes:

Secure Sockets Layer (SSL) is an industry-standard protocol for secure communications, involving encryption, authentication, and integrity of data. As an alternative to security exits, SSL can be used to communicate between the Administration API for WebSphere Message Broker and the broker.

SSL is supported in both client/server and qmgr/qmgr channels (including clusters). There are many flexible capabilities built-in, including the ability to select who are prepared to accept communications from based on their fully authenticated identity. This removes, for many people, the need to set up channel exits, where they were used for security purposes.

SSL, widely accepted in the Internet community, has been subjected to significant testing by the hacker community.

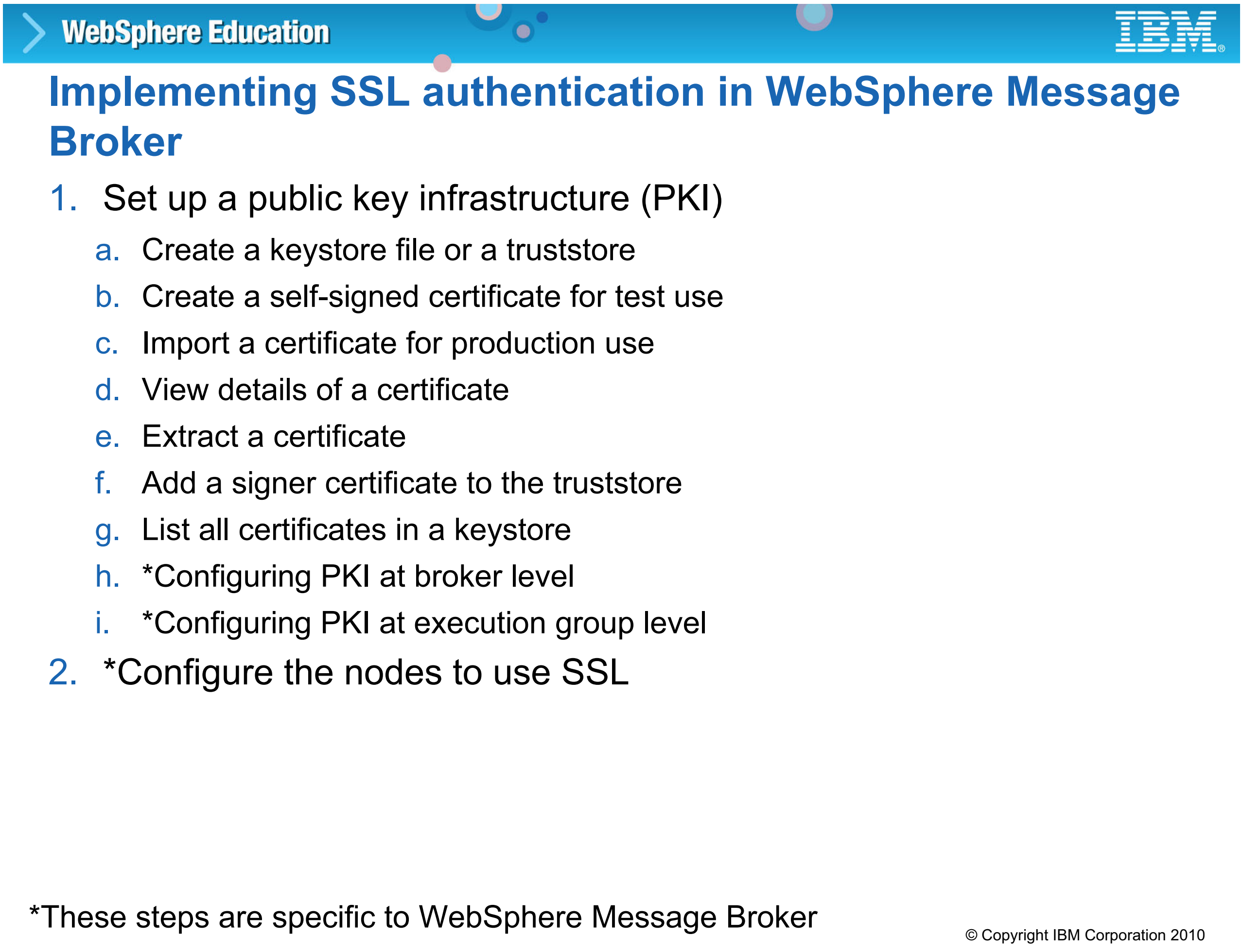
Instructor notes:

Purpose — Provide an overview of SSL.

Details —

Additional information —

Transition statement — One of the main tasks for configuring for SSL is setting up the public key infrastructure.



WebSphere Education

Implementing SSL authentication in WebSphere Message Broker

1. Set up a public key infrastructure (PKI)
   a. Create a keystore file or a truststore
   b. Create a self-signed certificate for test use
   c. Import a certificate for production use
   d. View details of a certificate
   e. Extract a certificate
   f. Add a signer certificate to the truststore
   g. List all certificates in a keystore
   h. *Configuring PKI at broker level
   i. *Configuring PKI at execution group level
2. *Configure the nodes to use SSL

*These steps are specific to WebSphere Message Broker

Figure 5-32. Implementing SSL authentication in Message Broker WM643 / VM6431.0

Notes:

The figure lists the high-level steps for setting up the public key infrastructure (PKI).

You can configure keystores and truststores at either broker level (one keystore, one truststore, and one personal certificate for each broker) or at execution group level (one keystore, one truststore, and one personal certificate for each execution group). Execution groups that do not have PKI configured use the broker-level PKI configuration. The HTTP nodes support PKI configuration only at broker level; the SOAP nodes support both levels.

Steps a through g in the figure are configured using the Global Secure Toolkit (gsk7cmd), supplied with WebSphere MQ is used to create and populate keystores and truststores.

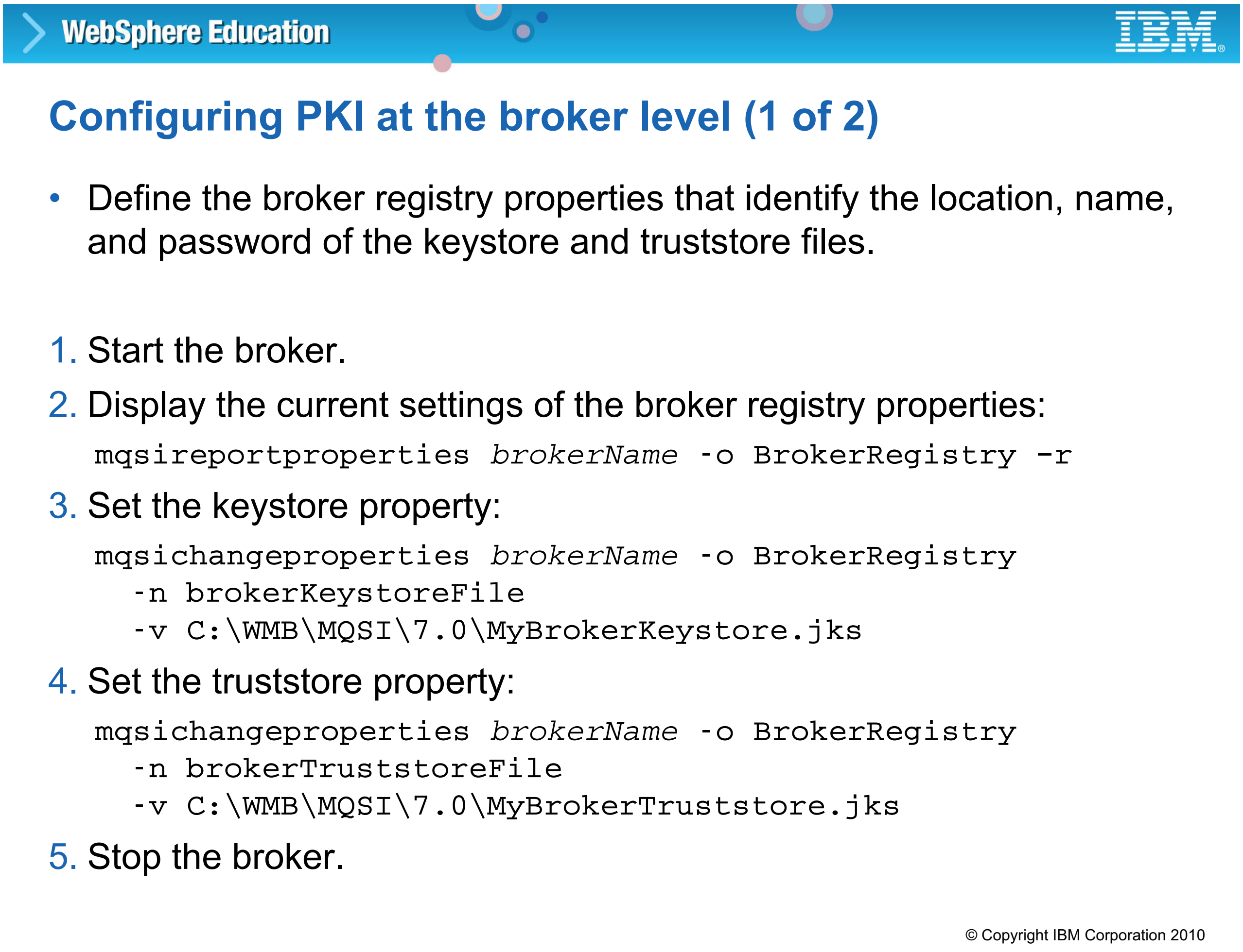
Instructor notes:

Purpose — List the steps required to set up a PKI.

Details — Steps a through g are not specific to WebSphere Message Broker and so are not described in detail in this course. The Global Secure Toolkit (gsk7cmd), supplied with WebSphere MQ is used to create and populate keystores and truststores.

Additional information —

Transition statement — The next series of slides provide detail on configuring PKI at the broker level and at the execution group level.



Configuring PKI at the broker level (1 of 2)

- Define the broker registry properties that identify the location, name, and password of the keystore and truststore files.

1. Start the broker.
2. Display the current settings of the broker registry properties:

   ```
   mqsireportproperties brokerName -o BrokerRegistry -r
   ```

3. Set the keystore property:

   ```
   mqsichangeproperties brokerName -o BrokerRegistry
     -n brokerKeystoreFile
     -v C:\WMB\MQSI\7.0\MyBrokerKeystore.jks
   ```

4. Set the truststore property:

   ```
   mqsichangeproperties brokerName -o BrokerRegistry
     -n brokerTruststoreFile
     -v C:\WMB\MQSI\7.0\MyBrokerTruststore.jks
   ```

5. Stop the broker.

Figure 5-33. Configuring PKI at the broker level (1 of 2) WM643 / VM6431.0

Notes:

This figure and the next list the steps for defining the broker registry properties that identify the location, name, and password of the keystore and truststore files at the broker level.

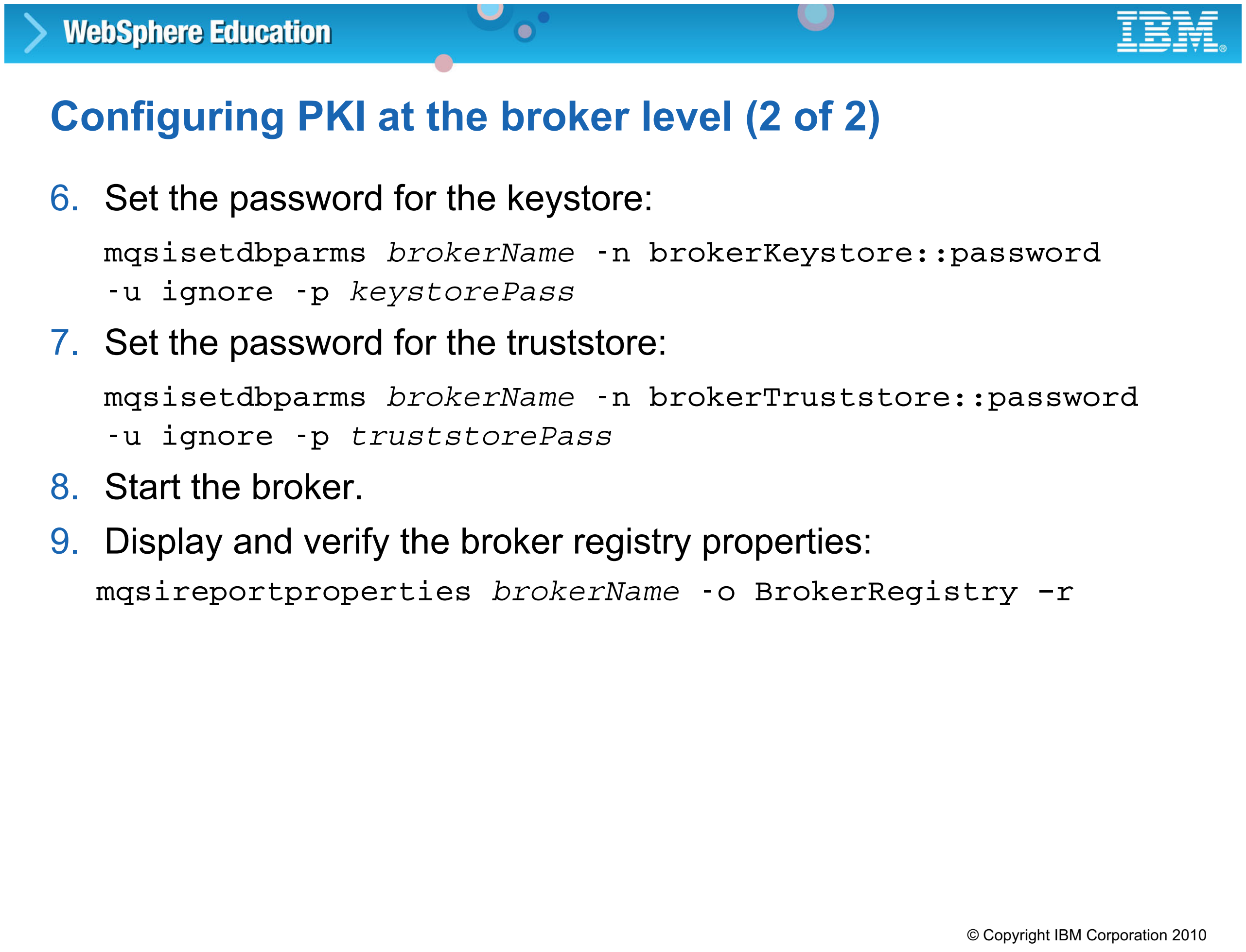
Instructor notes:

Purpose — List steps 1 through 5 for configuring a PKI at the broker level.

Details — These steps assume that the steps to set up the keystore and truststore have already been completed.

Additional information — None.

Transition statement — Next: Configuring PKI at the broker level (2 of 2)



Configuring PKI at the broker level (2 of 2)

6. Set the password for the keystore:

```
mqsisetdbparms brokerName -n brokerKeystore::password
-u ignore -p keystorePass
```

7. Set the password for the truststore:

```
mqsisetdbparms brokerName -n brokerTruststore::password
-u ignore -p truststorePass
```

8. Start the broker.
9. Display and verify the broker registry properties:

```
mqsireportproperties brokerName -o BrokerRegistry -r
```

Figure 5-34. Configuring PKI at the broker level (2 of 2) WM643 / VM6431.0

Notes:

The result of the `mqsireportproperties` command has the following format:

```
ServiceFederationManager
  uuid='ServiceFederationManager'
  userTraceLevel='none'
  traceLevel='none'
  userTraceFilter='none'
  traceFilter='none'
  port='8811'
  securePort='8844'
  maxWaitTime='180'
  proxyURLHostName=' mbhost.ibm.com '
  proxyPathPrefix='proxy'
  proxyPathPrefixesEnabled='TRUE'
  creationTime='2009-09-11 15:37:52.639219'
  nextProxyPathPrefixCount='8'
  ownedProxyGroupCount='2'
```

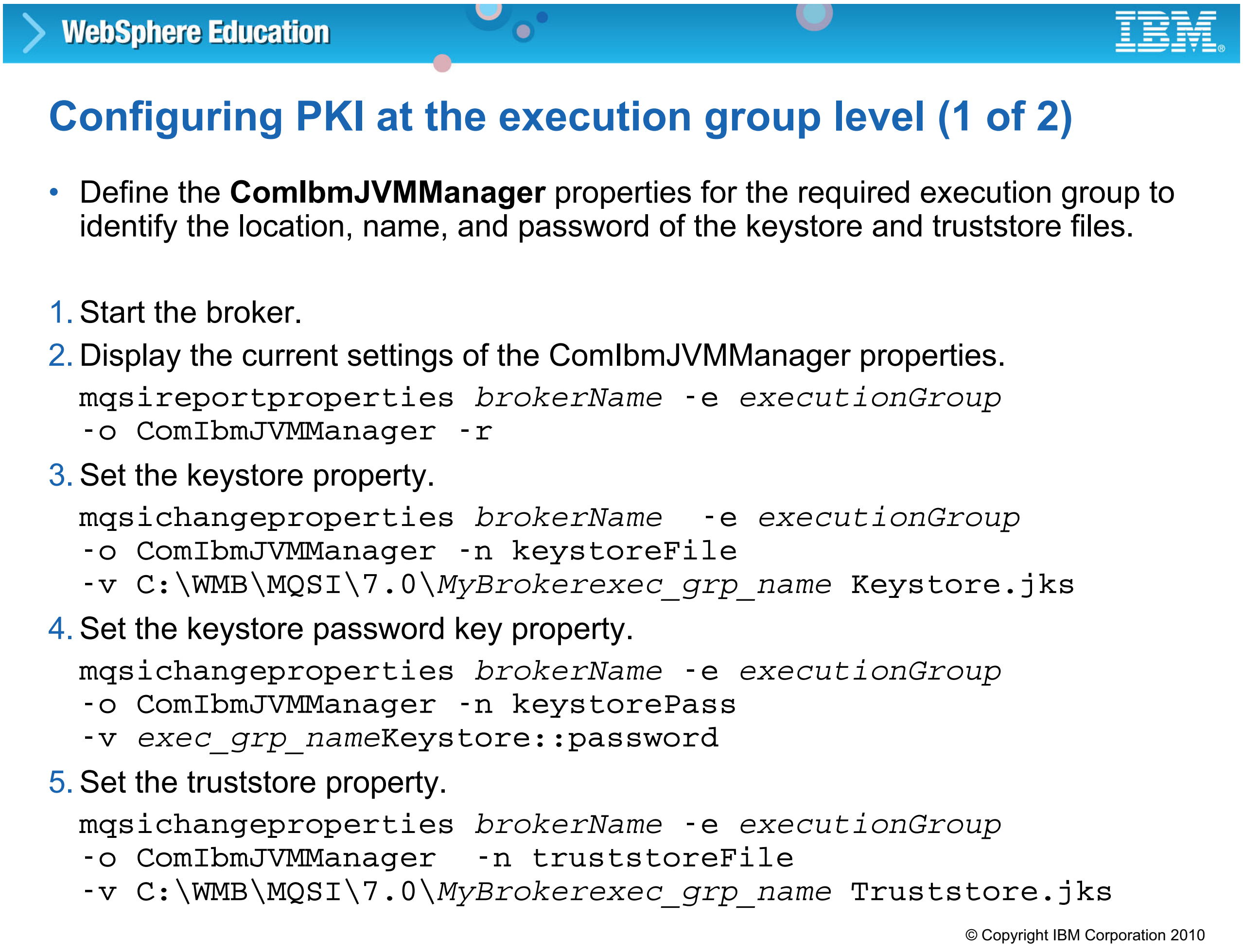
Instructor notes:

Purpose — List steps 6 through 10 for configuring PKI at the broker level.

Details —

Additional information —

Transition statement — Next: Configuring PKI at the execution group level (1 of 2)


WebSphere Education
IBM

Configuring PKI at the execution group level (1 of 2)

- Define the **ComIbmJVMManager** properties for the required execution group to identify the location, name, and password of the keystore and truststore files.

1. Start the broker.
2. Display the current settings of the ComIbmJVMManager properties.

```
mqsireportproperties brokerName -e executionGroup
-o ComIbmJVMManager -r
```

3. Set the keystore property.

```
mqsichangeproperties brokerName  -e executionGroup
-o ComIbmJVMManager -n keystoreFile
-v C:\WMB\MQSI\7.0\MyBrokerexec_grp_name Keystore.jks
```

4. Set the keystore password key property.

```
mqsichangeproperties brokerName -e executionGroup
-o ComIbmJVMManager -n keystorePass
-v exec_grp_nameKeystore::password
```

5. Set the truststore property.

```
mqsichangeproperties brokerName -e executionGroup
-o ComIbmJVMManager  -n truststoreFile
-v C:\WMB\MQSI\7.0\MyBrokerexec_grp_name Truststore.jks
```

Figure 5-35. Configuring PKI at the execution group level (1 of 2) WM643 / VM6431.0

Notes:

To configure PKI at the execution group level, define the ComIbmJVMManager properties for the required execution group to identify the location, name, and password of the keystore and truststore files.

This figure and the next, list the steps for configuring PKI at the execution group level.

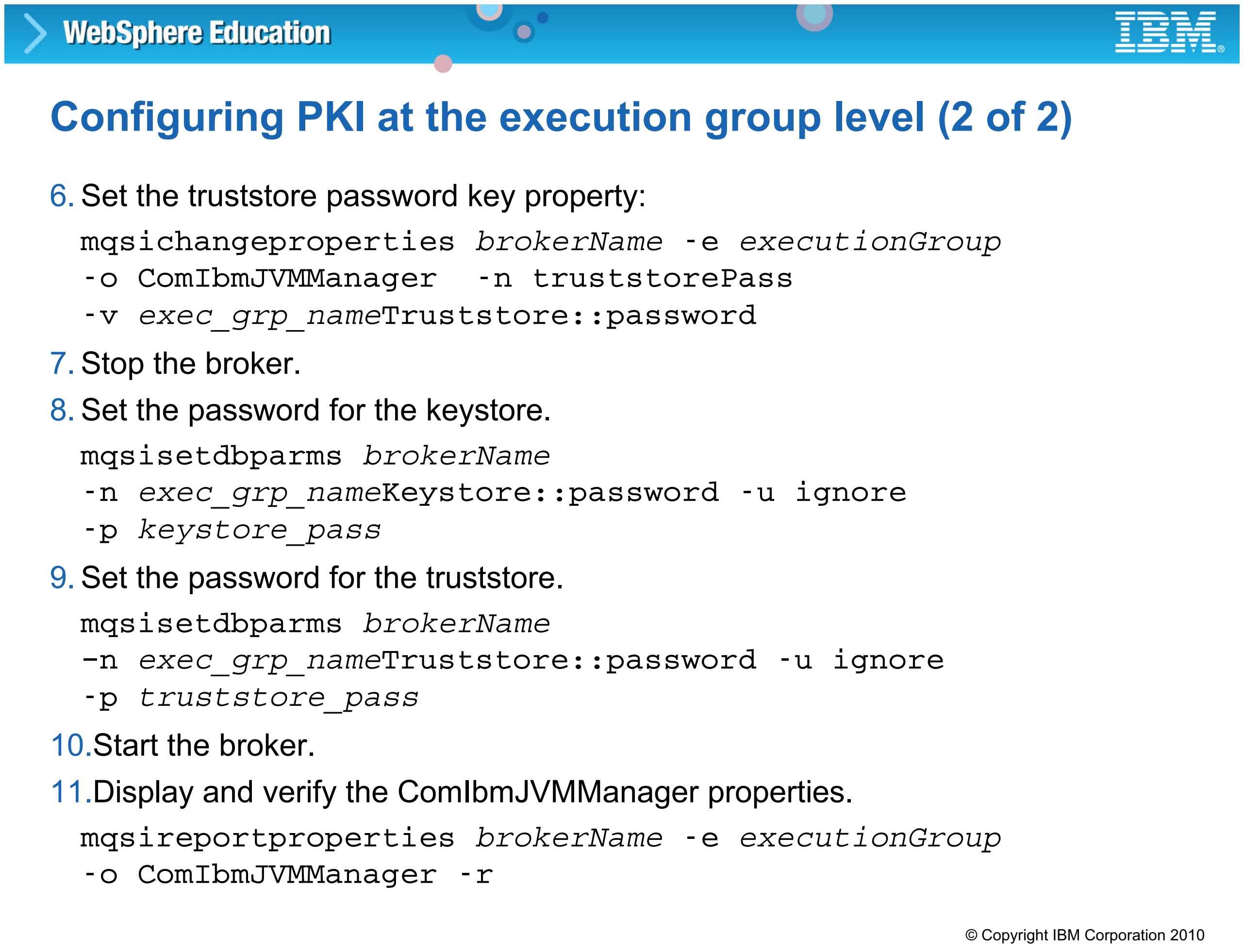
Instructor notes:

Purpose — List the steps for configuring the PKI at the execution group level.

Details — There steps assume that the keystore and truststore have already been created.

Additional information —

Transition statement — The remaining steps for creating the PKI at the execution group level are on the next slide.



WebSphere Education

Configuring PKI at the execution group level (2 of 2)

6. Set the truststore password key property:

```
mqsichangeproperties brokerName -e executionGroup
-o ComIbmJVMManager  -n truststorePass
-v exec_grp_nameTruststore::password
```

7. Stop the broker.
8. Set the password for the keystore.

```
mqsisetdbparms brokerName
-n exec_grp_nameKeystore::password -u ignore
-p keystore_pass
```

9. Set the password for the truststore.

```
mqsisetdbparms brokerName
-n exec_grp_nameTruststore::password -u ignore
-p truststore_pass
```

10. Start the broker.
11. Display and verify the ComIbmJVMManager properties.

```
mqsireportproperties brokerName -e executionGroup
-o ComIbmJVMManager -r
```

Figure 5-36. Configuring PKI at the execution group level (2 of 2) WM643 / VM6431.0

Notes:

Instructor notes:

Purpose — List the remaining steps for configuring the PKI for the execution group.

Details —

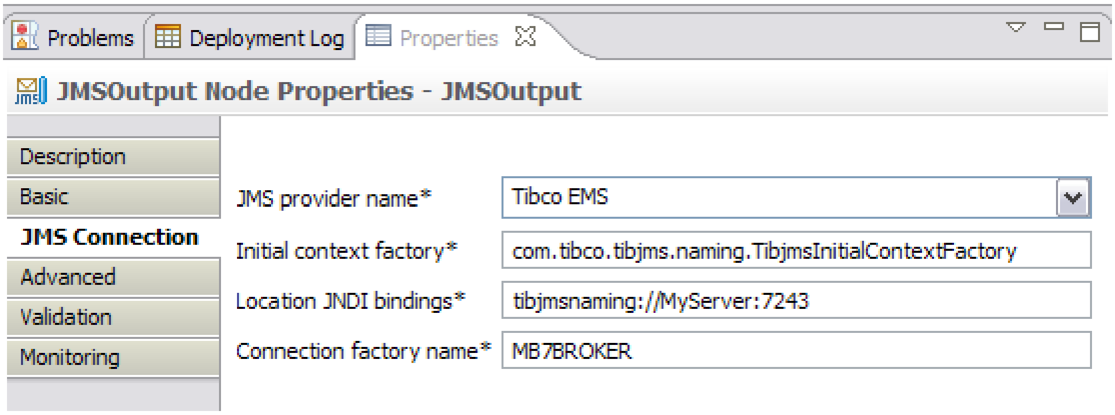
Additional information — None.

Transition statement — After the PKI has been set up, the next step is to configure the node to use SSL.

WebSphere Education

SSL and JMS nodes configuration

1. In the JMS node properties:
   - Select JMS provider name
   - Configure **Location JNDI bindings** with the URL that points to the JNDI bindings
   - Configure the **Connection factory name** to specify a pre-configured connection factory that is enabled for SSL connectivity
2. Make the client JAR files available to the broker



JMSOutput Node Properties - JMSOutput

JMS provider name* Tibco EMS

Initial context factory* com.tibco.tibjms.naming.TibjmsInitialContextFactory

Location JNDI bindings* tibjmsnaming://MyServer:7243

Connection factory name* MB7BROKER

Figure 5-37. SSL and JMS nodes configuration WM643 / VM6431.0

Notes:

JMS nodes use Java Naming and Directory Interface (JNDI) to look up a connection factory object that is used to create JMS connections.

The figure lists the steps for using SSL with the JMSOutput node. The node properties in the figure are set to use the TIBCO Enterprise Message Service (EMS).

To make the TIBCO EMS client JAR files available to the broker, for example, use the `mqsicreateconfigurableservice` or the `mqsichangeproperties` command to set the JMSProviders configurable service property `jarsURL` to point to the directory that contains the JMS provider client JAR files and the SSL vendor JAR files.

Note

Configuration of the JNDI administered objects used by the JMS nodes is specific to each JMS provider. Check the documentation supplied by the JMS provider.

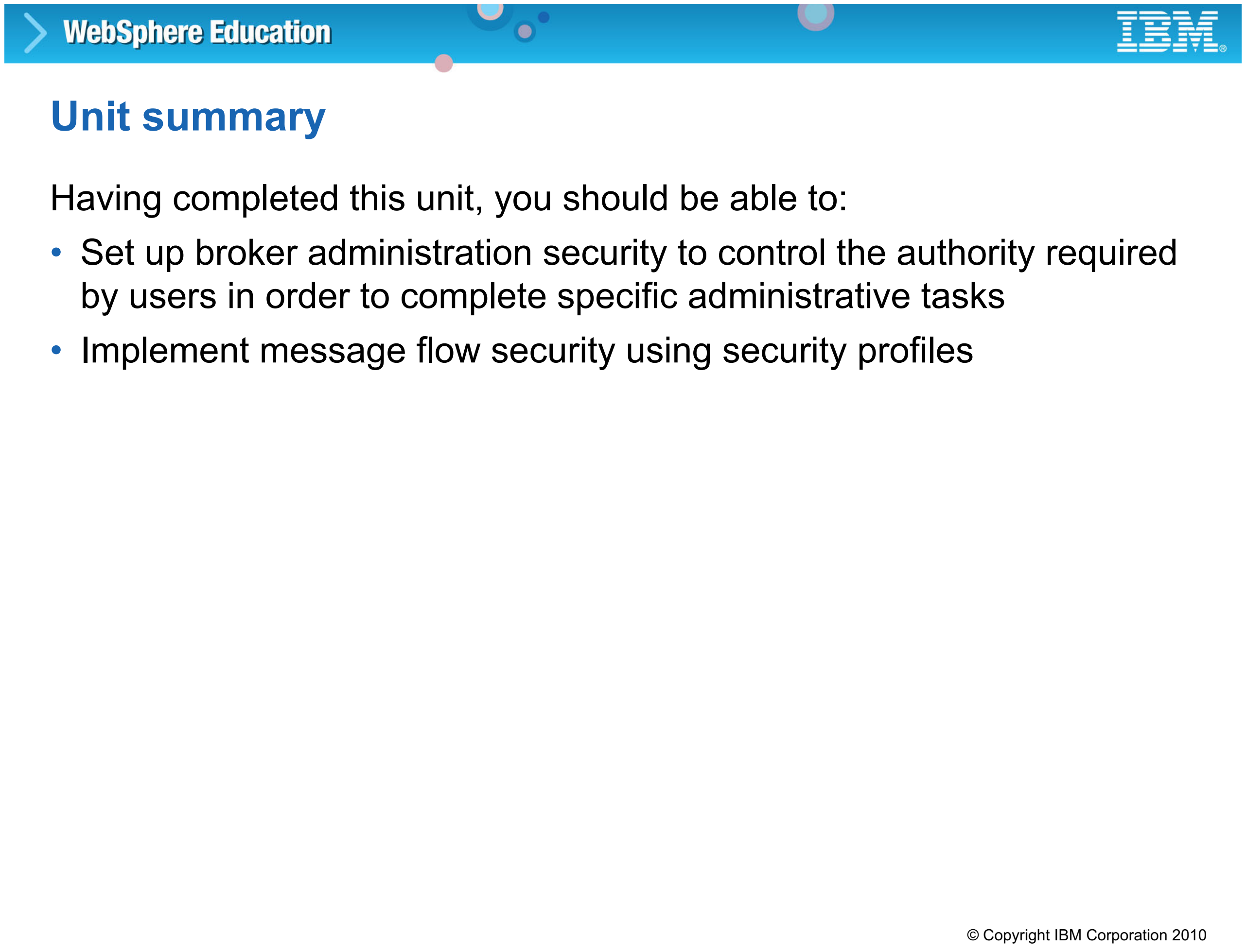
Instructor notes:

Purpose — Describe the steps for configuring JMS nodes to use SSL-enabled JNDI administered objects.

Details — The three built-in nodes JMSInput, JMSOutput, and JMSReply are referred to in this topic by the generic term JMS nodes.

Additional information — The JMS 1.1 Specification states that JMS does not provide features for controlling or configuring message integrity or message privacy. JMS providers typically support these additional features, and provide their own administration tools to configure these services.

Transition statement — Next: Unit summary



WebSphere Education

Unit summary

Having completed this unit, you should be able to:

- Set up broker administration security to control the authority required by users in order to complete specific administrative tasks
- Implement message flow security using security profiles

Figure 5-38. Unit summary WM643 / VM6431.0

Notes:

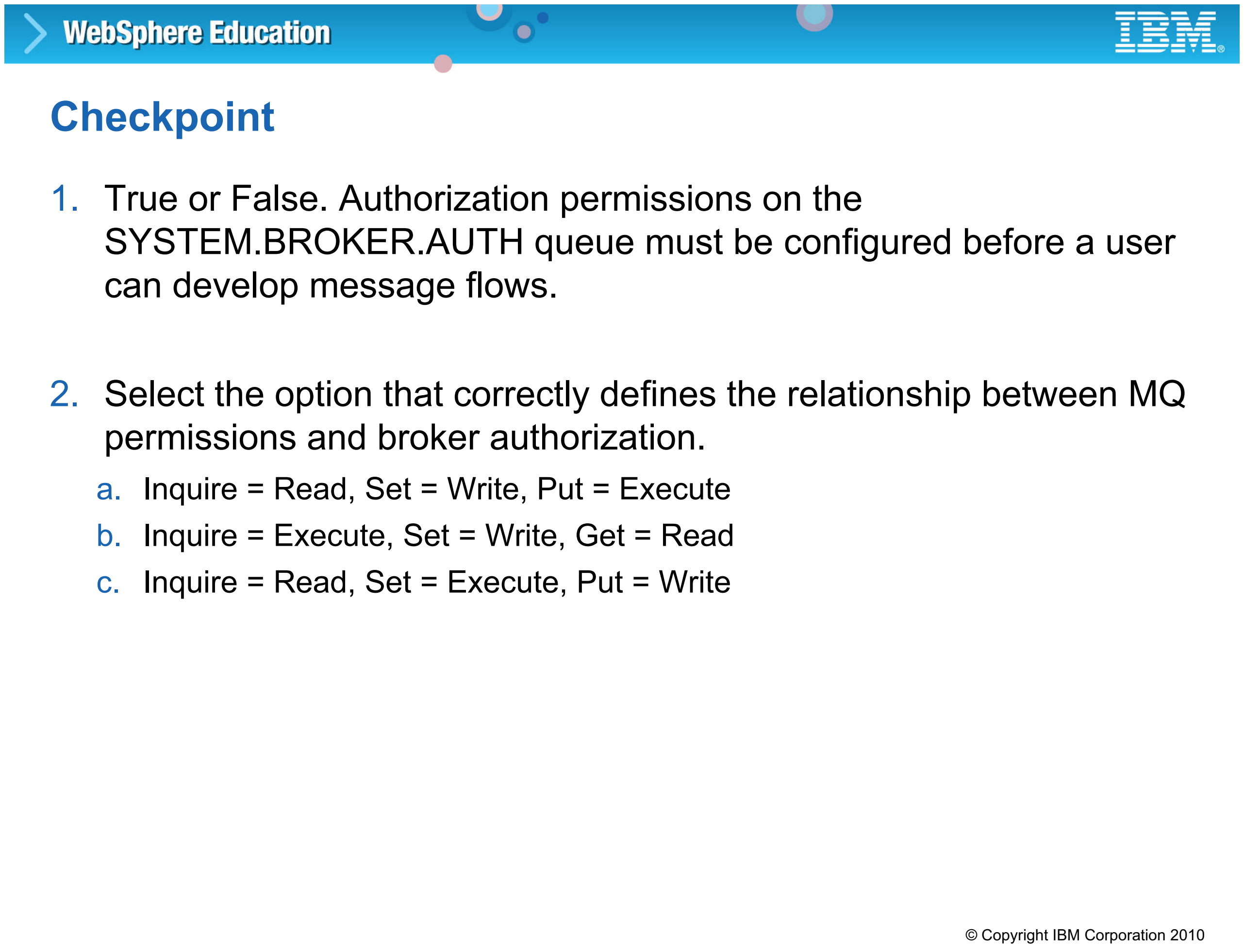
Instructor notes:

Purpose — Summarize the unit.

Details —

Additional information —

Transition statement — Some checkpoint questions test your knowledge before the lab exercise.



WebSphere Education

Checkpoint

1. True or False. Authorization permissions on the SYSTEM.BROKER.AUTH queue must be configured before a user can develop message flows.

2. Select the option that correctly defines the relationship between MQ permissions and broker authorization.
   a. Inquire = Read, Set = Write, Put = Execute
   b. Inquire = Execute, Set = Write, Get = Read
   c. Inquire = Read, Set = Execute, Put = Write

Figure 5-39. Checkpoint WM643 / VM6431.0

Notes:

Write your answers here:

1.

2.

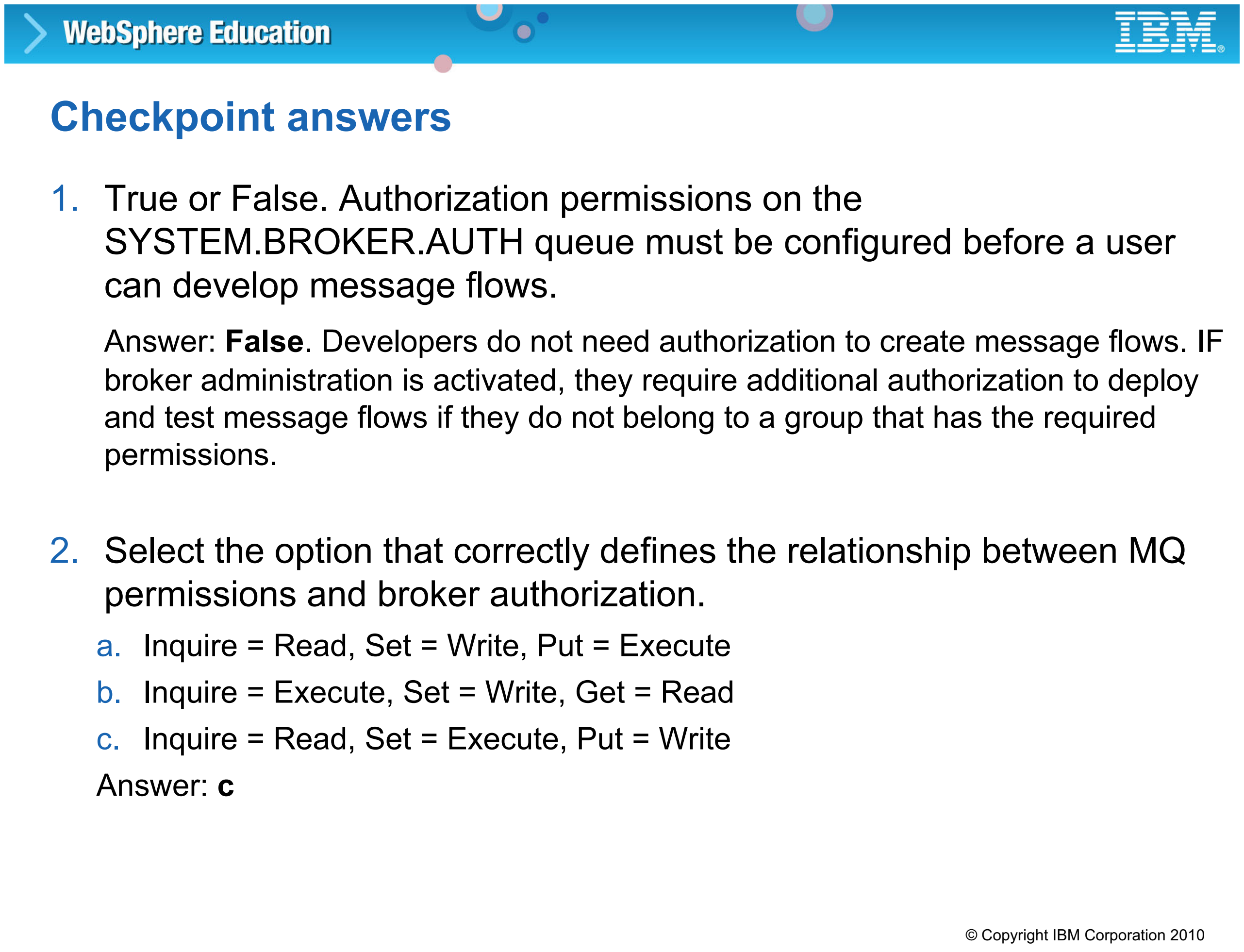
Instructor notes:

Purpose — Checkpoint questions to test student’s knowledge.

Details — The answers are on the page.

Additional information —

Transition statement — Next: Checkpoint answers



WebSphere Education

Checkpoint answers

1. True or False. Authorization permissions on the SYSTEM.BROKER.AUTH queue must be configured before a user can develop message flows.

   Answer: **False**. Developers do not need authorization to create message flows. IF broker administration is activated, they require additional authorization to deploy and test message flows if they do not belong to a group that has the required permissions.

2. Select the option that correctly defines the relationship between MQ permissions and broker authorization.

   a. Inquire = Read, Set = Write, Put = Execute

   b. Inquire = Execute, Set = Write, Get = Read

   c. Inquire = Read, Set = Execute, Put = Write

   Answer: **c**

Figure 5-40. Checkpoint answers WM643 / VM6431.0

Notes:

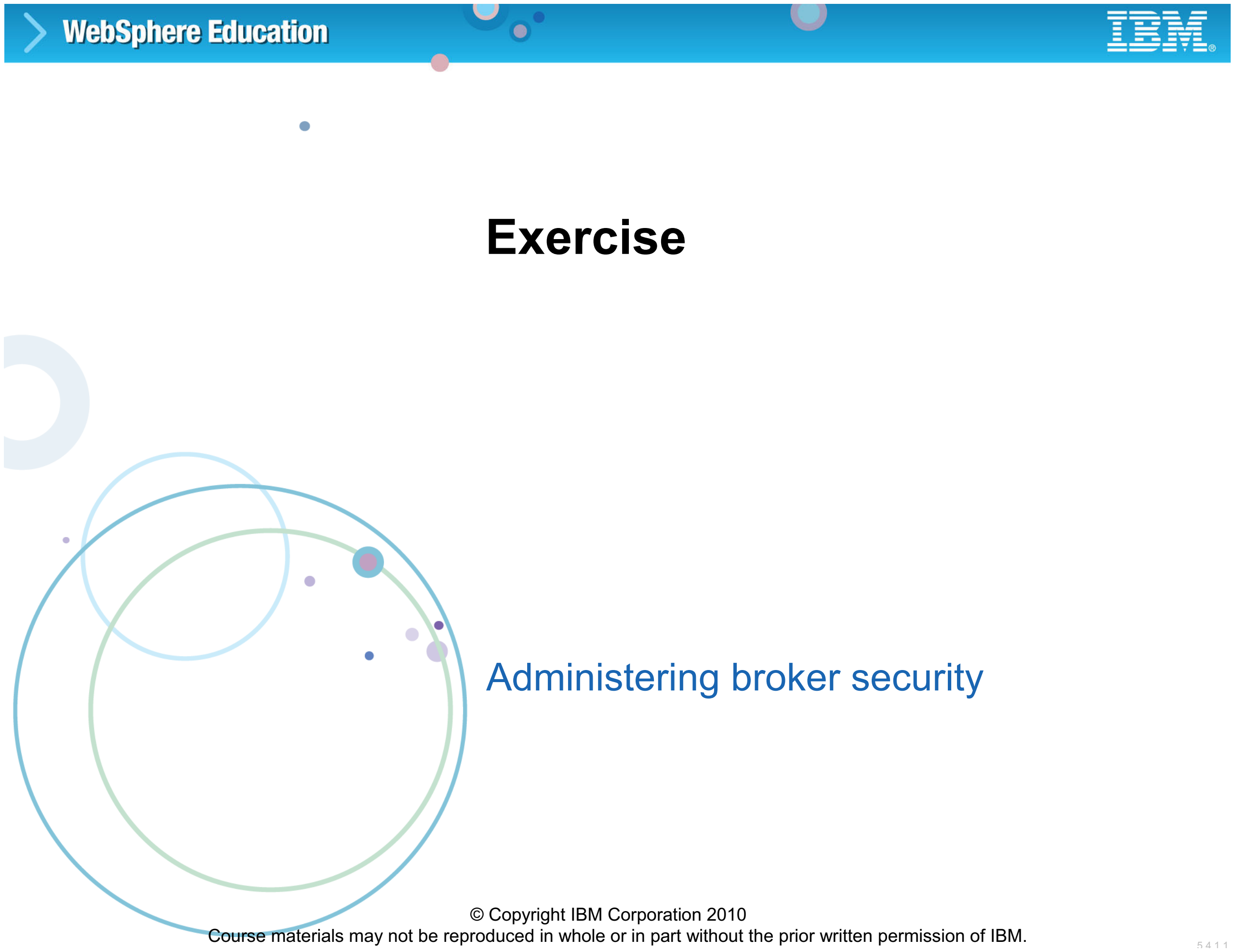
Instructor notes:

Purpose — Provide checkpoint answers.

Details —

Additional information —

Transition statement — Next: Exercise 4



WebSphere Education

IBM

Exercise

Administering broker security

5411

Figure 5-41. Exercise 4

WM643 / VM6431.0

Notes:

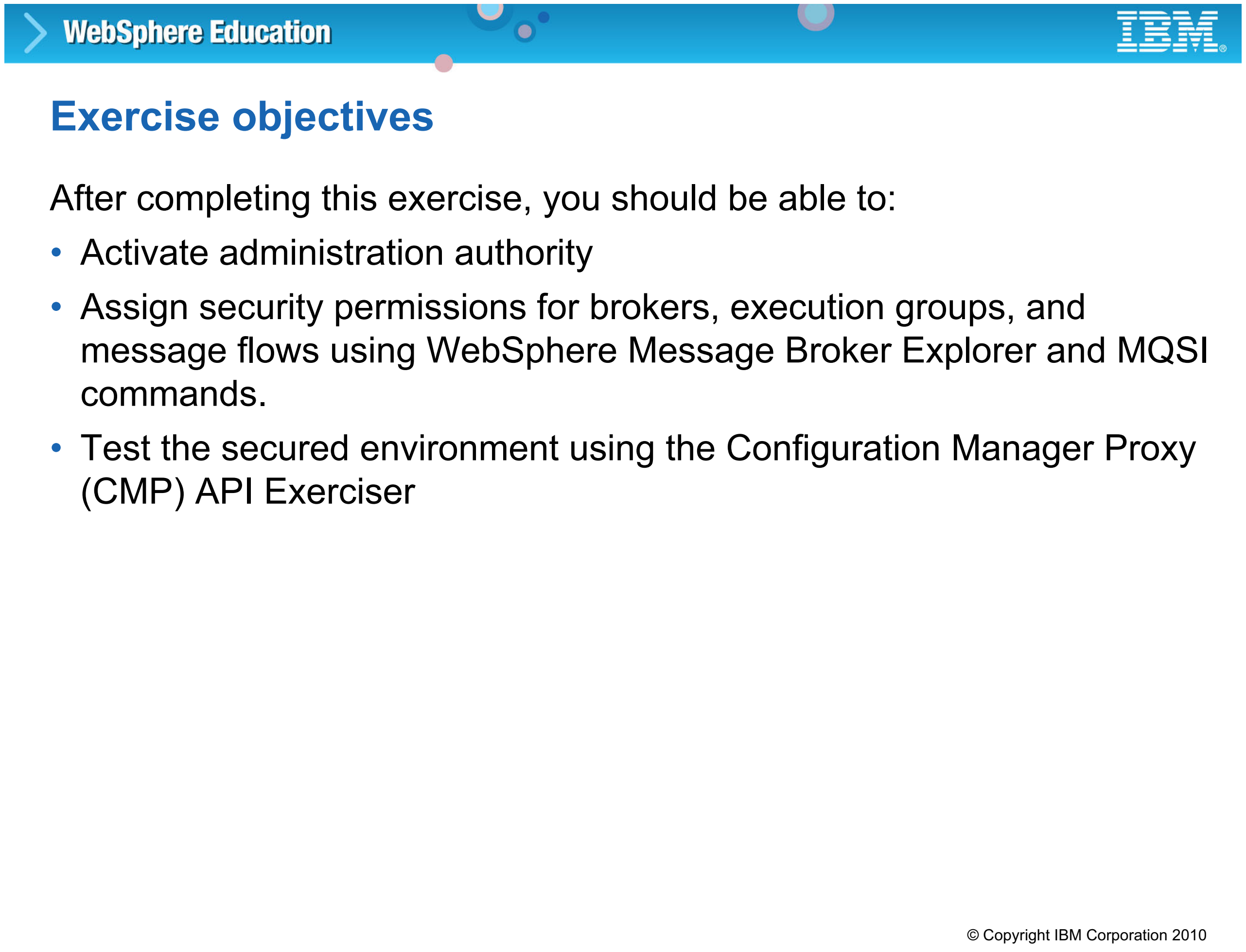
Instructor notes:

Purpose — Introduce the exercise.

Details — This slide provides an introduction to a group exercise.

Additional information —

Transition statement — Next: Exercise objectives



WebSphere Education
IBM

Exercise objectives

After completing this exercise, you should be able to:

- Activate administration authority
- Assign security permissions for brokers, execution groups, and message flows using WebSphere Message Broker Explorer and MQSI commands.
- Test the secured environment using the Configuration Manager Proxy (CMP) API Exerciser

Figure 5-42. Exercise objectives WM643 / VM6431.0

Notes:

See the *Student Exercises Guide* for detailed instructions.

Instructor notes:

Purpose — Introduce the exercise.

Details — This slide provides an introduction to a discussion or group exercise.

Additional information —

Transition statement —

Unit 6. Monitoring and problem determination

Estimated time

01:30 lecture, 01:30 exercises

What this unit is about

This unit introduces some of the tools and techniques that WebSphere Message Broker offers for problem determination and debugging.

First, however, the discussion covers default product behavior and investigate ways to alter this behavior by changing the message flow transaction mode. This, in turn, has an effect on message handling by the broker when runtime errors are encountered.

What you should be able to do

After completing this unit, you should be able to:

- Describe the default message flow behavior when an error occurs
- Configure message flows for transactional behavior
- Troubleshoot common problems for administrators

How you will check your progress

Accountability:

- Checkpoint questions
- Exercise 5, Using trace facilities
- Exercise 6, Identifying runtime problems

References

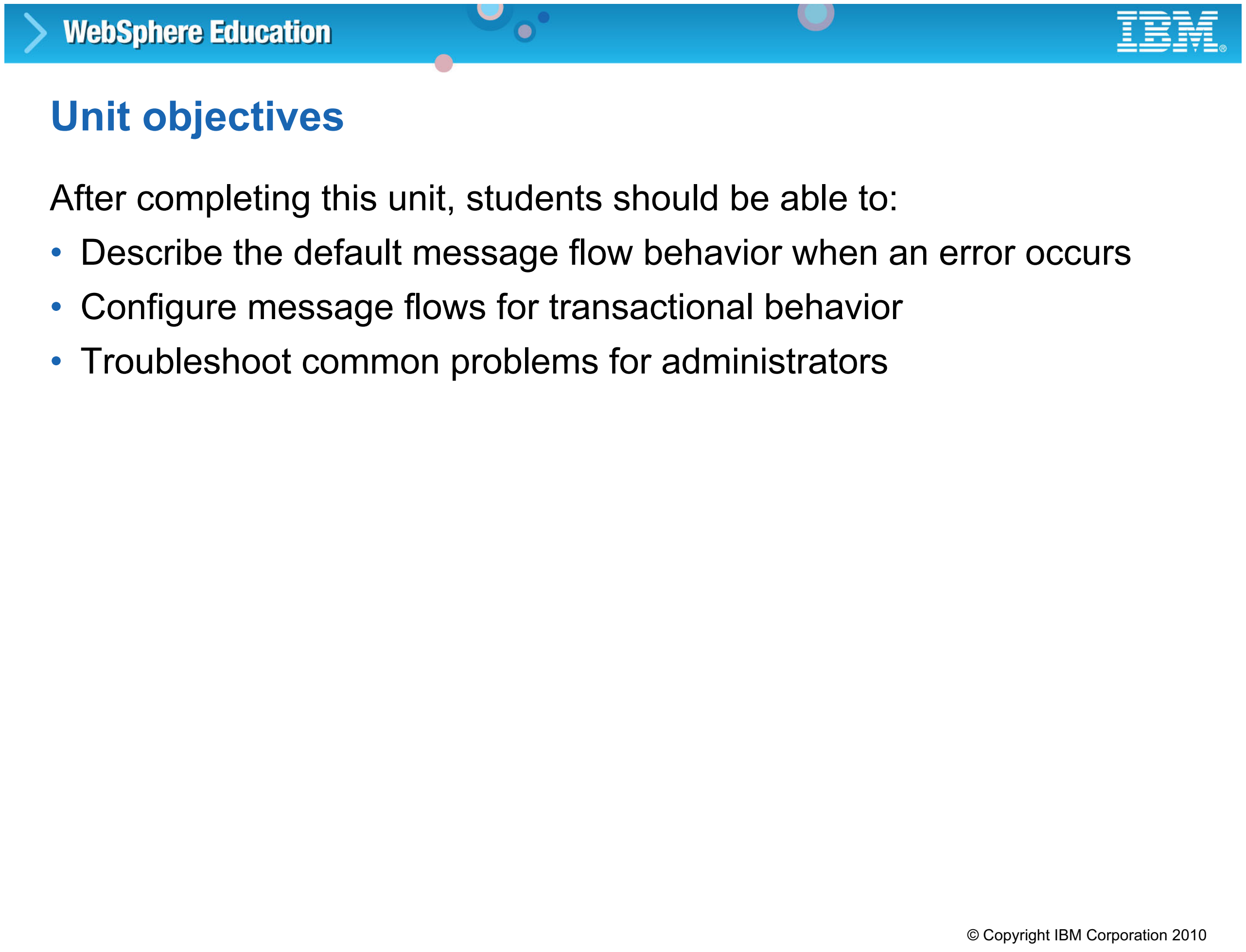
`www.ibm.com/software/integration/wbimessagebroker/library`

IBM WebSphere Message Broker Library

WebSphere Message Broker stand-alone online help system

`www.ibm.com/software/integration/wmq/library`

IBM WebSphere MQ Library



WebSphere Education

Unit objectives

After completing this unit, students should be able to:

- Describe the default message flow behavior when an error occurs
- Configure message flows for transactional behavior
- Troubleshoot common problems for administrators

Figure 6-1. Unit objectives WM643 / VM6431.0

Notes:

Instructor notes:

Purpose —

Details — This unit focuses on problem determination and troubleshooting. The message flow troubleshooting and error handling in the first topic should be presented with a focus on the administrator tasks involved in message flow troubleshooting such as creating debug ports and enabling and disabling trace.

Additional information —

Transition statement — Next: Message flow: Case1

6.1. Broker default behavior in case of runtime errors

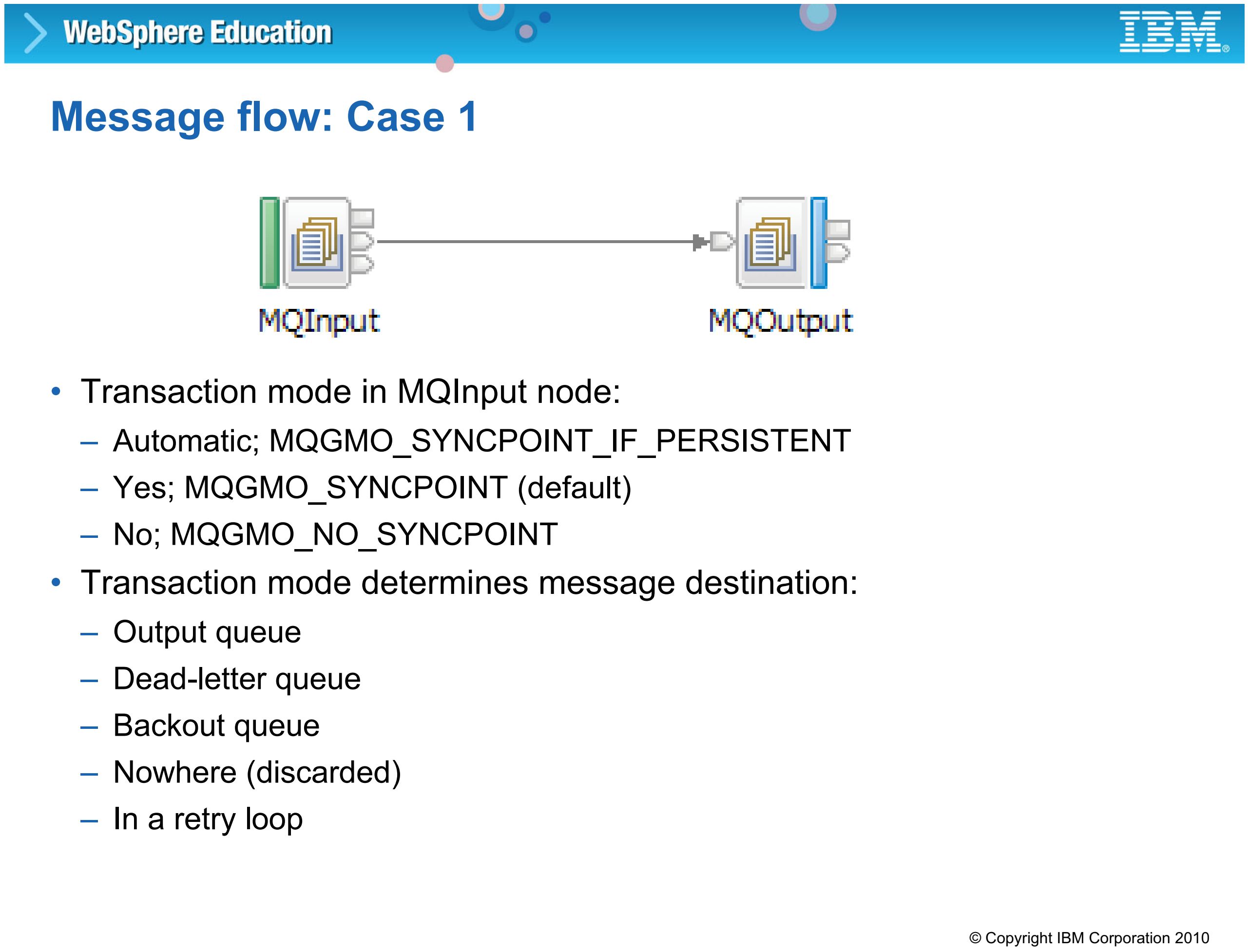
Instructor topic introduction

What students will do — Students will become acquainted with several default behaviors that the WebSphere Message Brokers exhibit in an error situation. Students will also learn how the broker and queue manager work together to ensure message and database integrity.

How students will so it — Listen to the lecture and participate in classroom discussion. Exercise 5 will give students the opportunity to experience and deal with some runtime error conditions.

What students will learn — Understanding message flow behavior is fundamental to problem determination. It is important to know what to expect in order to apply the appropriate problem determination tools. It is also important to understand the value and cost of executing message flows with and without transactional behavior.

How this will help students in their job — Students will be better prepared to deal with runtime errors common to message flows. In turn, students are more likely to consider all possible message flow outcomes. In addition to understanding message flow default behavior, students benefit from an understanding of the runtime effect of running message flows with and without transactional protection.


WebSphere Education
IBM
MQInput
MQOutput

Message flow: Case 1

- Transaction mode in MQInput node:
  - Automatic; MQGMO_SYNCPOINT_IF_PERSISTENT
  - Yes; MQGMO_SYNCPOINT (default)
  - No; MQGMO_NO_SYNCPOINT
- Transaction mode determines message destination:
  - Output queue
  - Dead-letter queue
  - Backout queue
  - Nowhere (discarded)
  - In a retry loop

Figure 6-2. Message flow: Case1

WM643 / VM6431.0

Notes:

The message flow in this case is a simple flow connecting an MQInput node to an MQOutput node.

The MQInput node properties include a choice of three transaction modes: **Automatic**, **Yes** (the default setting), and **No**.

- If the transaction mode is set to **No**, and an error is encountered in the message flow, even a persistent message could be discarded.
- With transaction mode set to **Yes**, all messages are rolled back in the event of an error. This might be inefficient, given the volatile nature of nonpersistent messages.
- The **Automatic** setting affords transaction protection for persistent messages while denying the same for nonpersistent messages.

If the flow is successful, the messages are put on the output queue, if that is the intended destination. However, depending on what other terminals are connected, how the input queue is defined to WebSphere MQ, or the transaction mode, the message might be routed to one of the several destinations, identified in the figure.

Instructor notes:

Purpose — Identify default behavior characteristics of message flows.

Details — Students might not realize all the things that can affect message flow behavior. Start them here but let them know there is more.

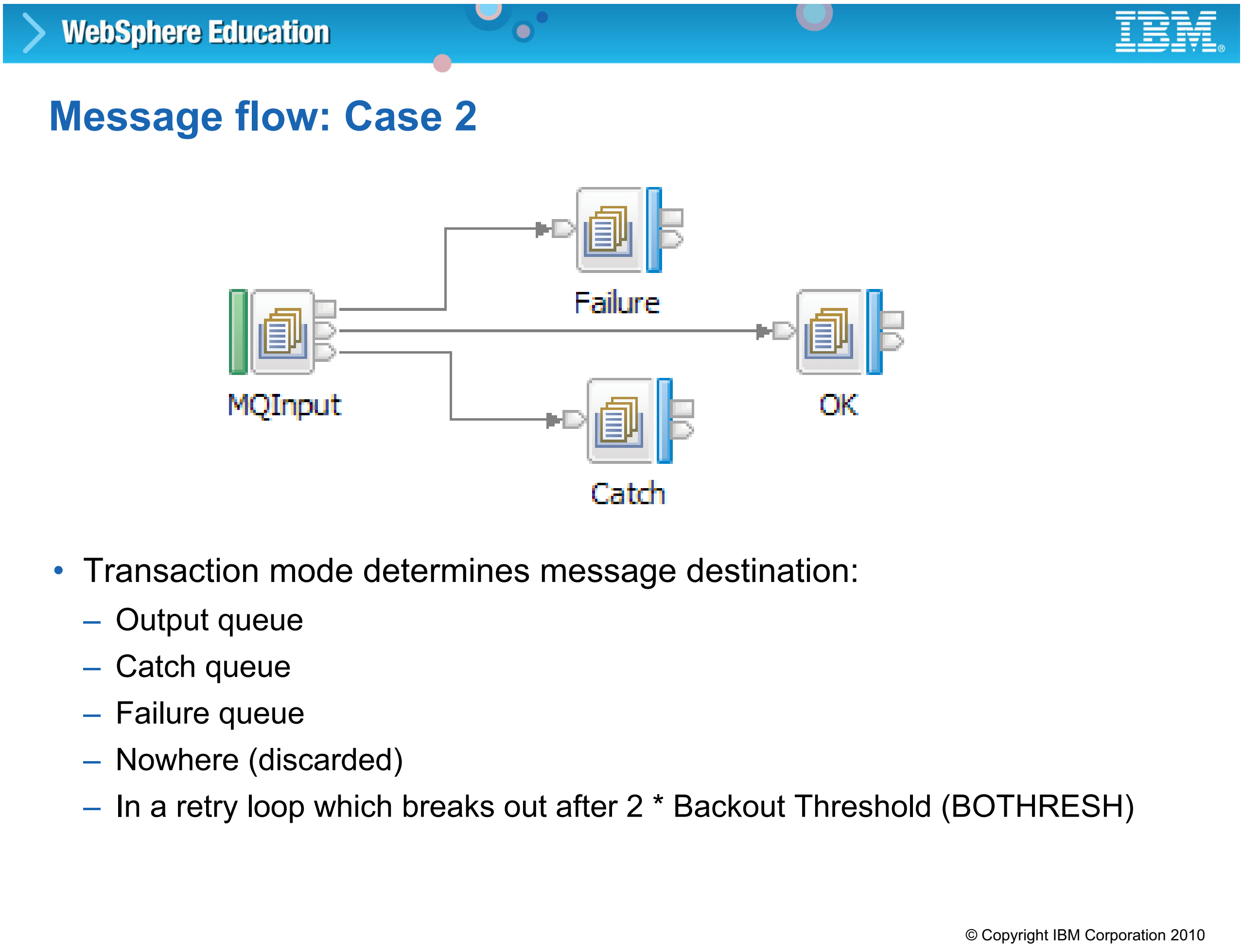
The figure makes some assumptions about the students' level of understanding of WebSphere MQ. Poll the class and ask. You might need to take some time to explain some details about WebSphere MQ transaction behavior (that is, messages are rolled back to the input queue) based on an `MQGET` with a get message option of `Syncpoint`.

While on the WebSphere MQ subject, identify the intent and purpose of the backout requeue queue as a means to avoid the use of the queue manager's dead-letter queue. This queue attribute works with the backout threshold and can assist in the handling of messages that encounter an error in a message flow.

Emphasize that the figure uses *only* the Out terminal of the MQInput node. This is key to explaining the potential results listed. The next figure describes some error handling alternatives.

Additional information —

Transition statement — The message destination can be altered by using the other MQInput node terminals.



Message flow: Case 2

- Transaction mode determines message destination:
  - Output queue
  - Catch queue
  - Failure queue
  - Nowhere (discarded)
  - In a retry loop which breaks out after 2 * Backout Threshold (BOTHRESH)

Figure 6-3. Message flow: Case 2

WM643 / VM6431.0

Notes:

A message flow developer can choose to ignore or handle runtime errors within a message flow. The previous figure ignored error handling.

In this figure, the Failure and Catch terminals of the MQInput node have been connected. In this example, the message flow developer is explicitly handling error conditions and by so doing, overriding the default behavior of WebSphere Message Broker.

If there is an error that causes a message to be rolled back, the message is first directed to the MQOutput node labeled CATCH for disposition. Should another error occur, the rollback continues, and the message is next directed to the MQOutput node labeled FAILURE for disposition. In the unlikely event of a third error, the message ends up in a retry loop between the MQInput node and the MQOutput node labeled FAILURE. However, to avoid a continuous loop, after a number of attempts equal to twice the value of the BackoutThreshold for that queue, the message breaks out of the rollback and is directed to the backout requeue queue or the dead-letter queue.

With this design, a message is never directed to the queue manager’s dead-letter queue or a user-defined backout queue.

Instructor notes:

Purpose — Further discuss message flow behavior.

Details — None.

Additional information — In this figure, messages do not go to the backout queue or the queue manager's dead-letter queue until after (2 * queue's BackoutThreshold) attempts. This is the time to remind students of the previous figure where the only wired terminal was the Out terminal. Therefore, the conclusion to draw is that the more options designed into the flow, the less default behavior takes place.

Transition statement — Review the possible combinations and what results can be expected.

Message flow error behavior: BOTHRESH not exceeded

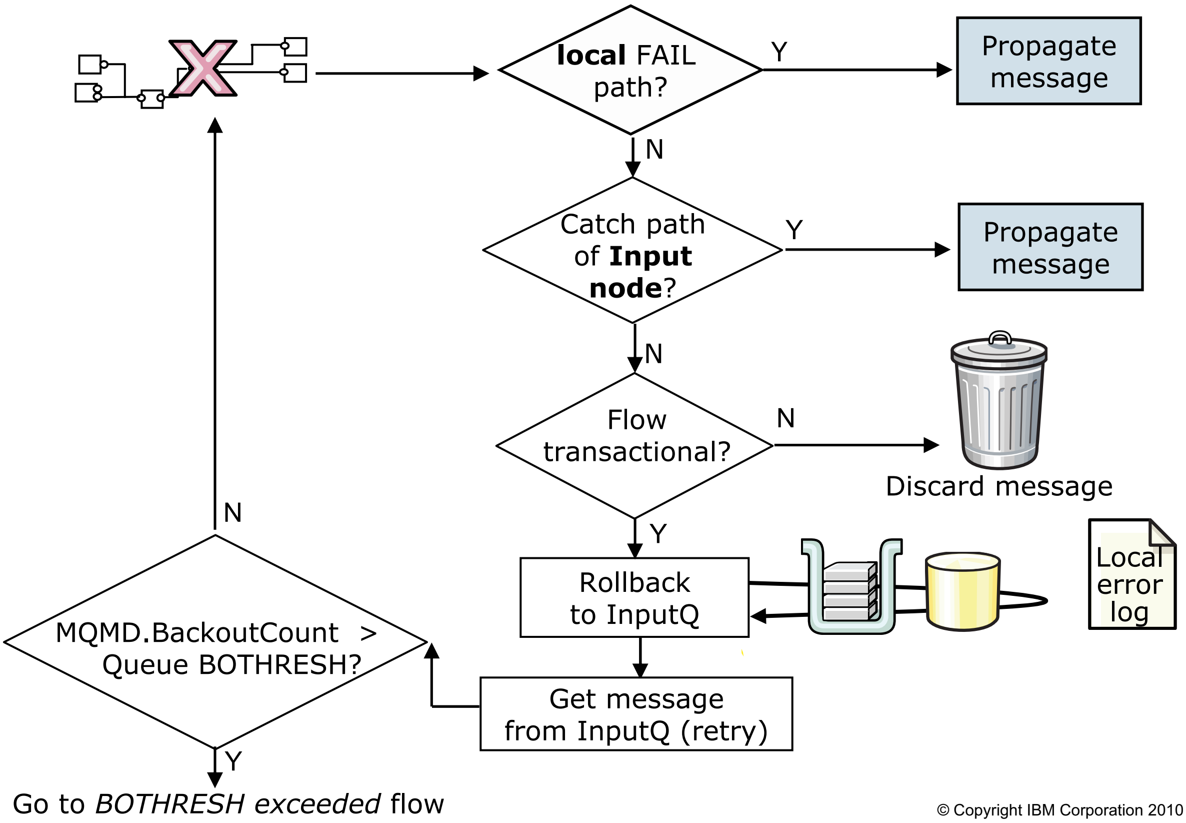

local FAIL path?
Y
Propagate message
N
Catch path of Input node?
Y
Propagate message
N
Flow transactional?
N
Discard message
Y
Rollback to InputQ
Local error log
Get message from InputQ (retry)
MQMD.BackoutCount > Queue BOTHRESH?
N
Y
Go to BOTHRESH exceeded flow
© Copyright IBM Corporation 2010

Figure 6-4. Message flow error behavior: BOTHRESH not exceeded WM643 / VM6431.0

Notes:

The figure shows the events that occur when an error occurs in a processing node. The levels of error handling are like filters. If the FAILURE terminal of the node where the exception occurred is not connected, the message falls through to level 2. If the CATCH terminal of the MQInput node is not connected, the MQInput node transaction mode is checked; if the message is not transactional, it is discarded. Transactional messages are rolled back to the input queue, and the message flow restarts. The retry attempt might not work and would end with another backout.

A backout threshold (BOTHRESH) property on the input queue limits the number of retry attempts. The next page shows the behavior that occurs when the BOTHRESH is exceeded.

Instructor notes:

Purpose — Discuss possible behaviors based on MQInput node properties and terminal connections.

Details — The error behavior flow is shown in two figures. This first figure shows the error flow behavior when the BOTHRESH count is not exceeded.

Additional information — The figure looks intimidating but it is not. After a ‘true’ condition is encountered, that behavior is performed.

Transition statement — What happens when the BOTHRESH is exceeded?

WebSphere Education

Message flow error behavior: BOTHRESH exceeded

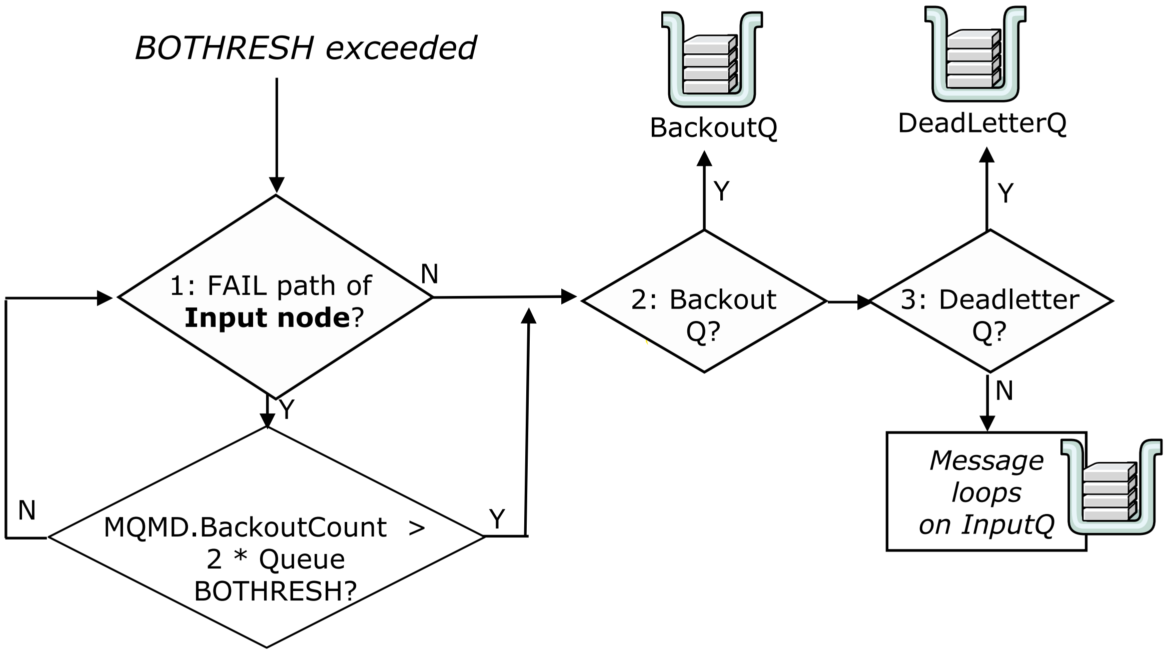


Figure 6-5. Message flow error behavior: BOTHRESH exceeded WM643 / VM6431.0

Notes:

When the BOTHRESH value is reached, the message destination depends on the existence of the FAIL path of an input node, the backout queue (BOQ) of the input queue, or the DeadLetterQ of the queue manager (they are checked in exactly this order).

If a failure occurs beyond the FAILURE terminal, further retry attempts occur until the backout count in the MQMD is twice the BOTHRESH set for the input queue. When this limit is reached, the node attempts to put the message to either the BOQ or DLQ. If neither of these exist, or if any of these fail, the message loops on the input queue, and prevents any new messages from being processed. In these cases, the error condition must be corrected manually.

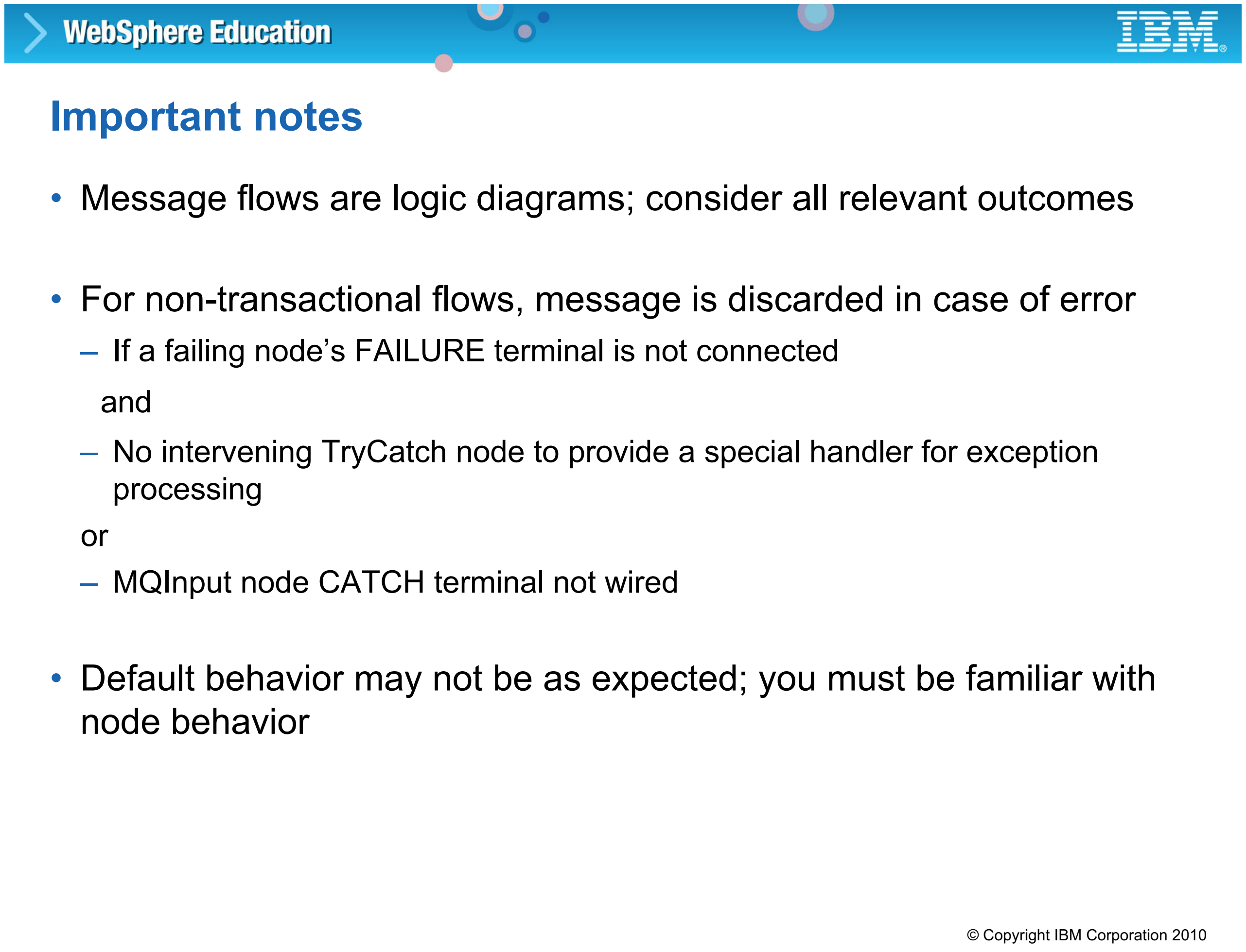
Instructor notes:

Purpose — Show the possible destinations of the message when an error occurs and the BOTHRESH is exceeded.

Details —

Additional information —

Transition statement — Summarize this topic about default and optional behaviors



WebSphere Education

Important notes

- Message flows are logic diagrams; consider all relevant outcomes
- For non-transactional flows, message is discarded in case of error
  - If a failing node's FAILURE terminal is not connected

    and
  - No intervening TryCatch node to provide a special handler for exception processing

  or
  - MQInput node CATCH terminal not wired
- Default behavior may not be as expected; you must be familiar with node behavior

Figure 6-6. Important notes WM643 / VM6431.0

Notes:

Message flows that rely on default behavior, that is, with node output terminals unwired, put the administrator in an awkward position. It is the administrator who must be aware of possible default behaviors. The default behavior includes a message flow loop as a final action. A symptom of such a loop is a message count (current depth) on an input queue not decrementing (or incrementing, implying a backlog of unprocessed messages).

Developers should take ownership of the message flows they develop insofar as anticipating and handling errors within the flow. Otherwise, messages can be delivered to unexpected queues, or discarded; or looping can occur.

More complex message flows, such as those that use filter and compute nodes, whose behavior is controlled by ESQL statements, which can produce errors, can introduce even more possible destinations, depending on how a particular message flow is wired to handle or ignore errors.

The developer can add a TryCatch node to provide a special handler for exception processing. If a downstream node throws an exception, the TryCatch node catches it and routes the original message to its Catch terminal.

Instructor notes:

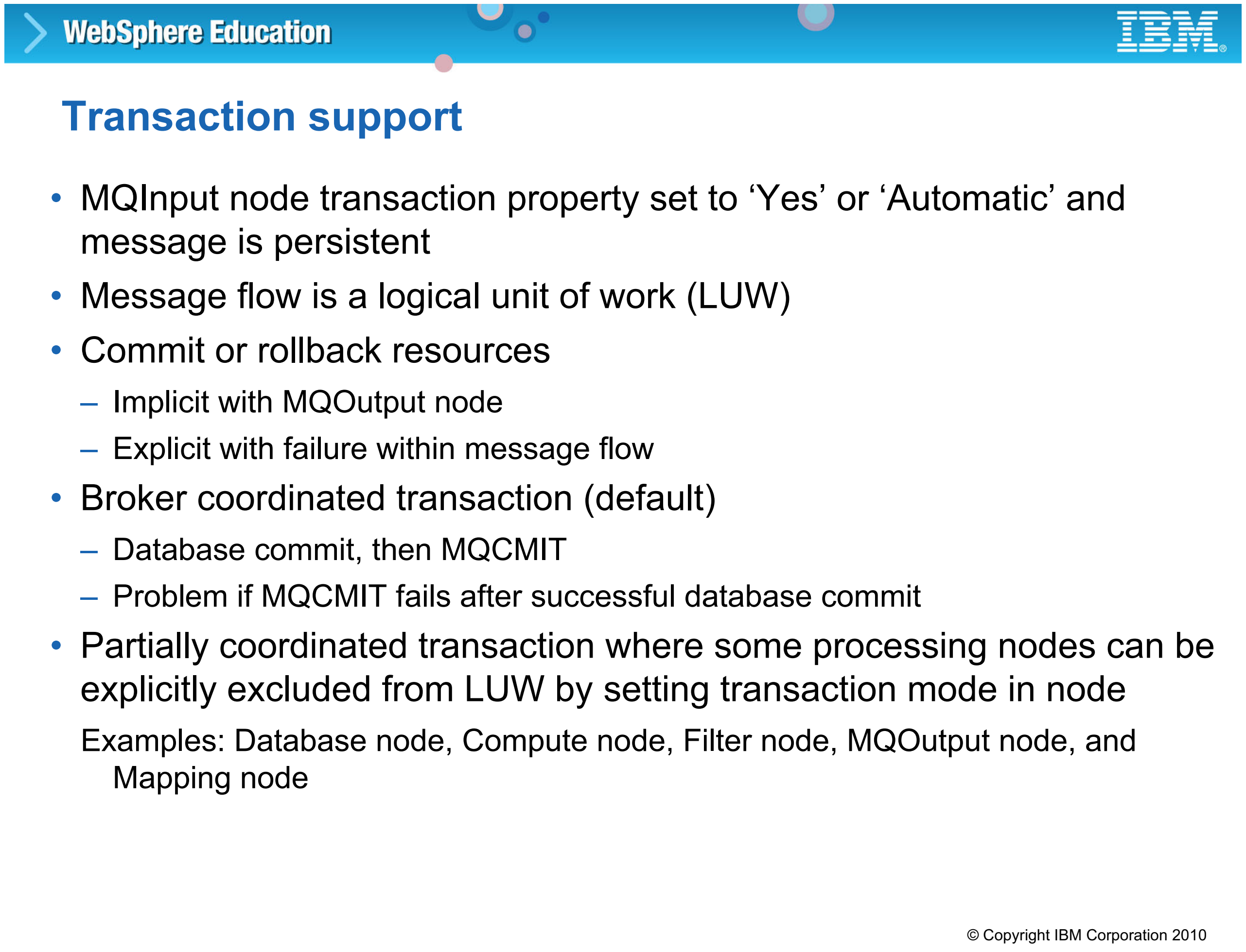
Purpose — Be sure that everyone really understands the behaviors that have been covered.

Remind students that WebSphere MQ, by default, attempts to prevent a message from ever being lost. If the default behavior is overridden, the likelihood of a message being discarded increases. It is important that students understand the error in the message flow error behavior.

Details — Transactions are discussed in more detail on the next page.

Additional information — None.

Transition statement — Next: Transaction support



WebSphere Education

Transaction support

- MQInput node transaction property set to 'Yes' or 'Automatic' and message is persistent
- Message flow is a logical unit of work (LUW)
- Commit or rollback resources
  - Implicit with MQOutput node
  - Explicit with failure within message flow
- Broker coordinated transaction (default)
  - Database commit, then MQCMIT
  - Problem if MQCMIT fails after successful database commit
- Partially coordinated transaction where some processing nodes can be explicitly excluded from LUW by setting transaction mode in node

  Examples: Database node, Compute node, Filter node, MQOutput node, and Mapping node

Figure 6-7. Transaction support WM643 / VM6431.0

Notes:

WebSphere Message Broker supports both transactional (XA) and non-transactional connections to databases. You can define an ODBC connection, or a JDBC type 4 connection, or both, to your database instances. XA support is referred to as a globally coordinated message flow.

On a single broker, you can use ODBC XA, or JDBC XA, but not both. This restriction applies to all supported platforms, and to all database servers for which XA is supported.

The default broker behavior is to coordinate transactions. It explicitly and separately commits (or back out) database resources at *flow completion*, and does the same for WebSphere MQ resources.

The default broker behavior might work in most cases, but there is the chance of a problem if the first commit (database) succeeds, but the second (WebSphere MQ) does not.

Using a two-phase commit with a single transaction coordinator prevents this problem. WebSphere MQ can act as the transaction coordinator for its own (queues) and for database resources, provided the queue manager is suitably configured. The WebSphere

Message Broker uses this WebSphere MQ facility when supervising a globally coordinated transaction (XA-coordination with a two-phase commit).

Support and restrictions vary based on database and platforms. See the WebSphere Message Broker Information Center and search on the strings “ODBC support” and “JDBC type 4 support” for specific versions and platforms.

Instructor notes:

Purpose — Detail broker coordinated and globally coordinated transactions, and partially coordinated transactions.

Details — None.

Additional information — Distributed platform and database support is shown below. RRS provides transaction support for zSeries. The following table lists the supported database servers for each broker platform for non-XA support. XA connections are supported by most of these servers.

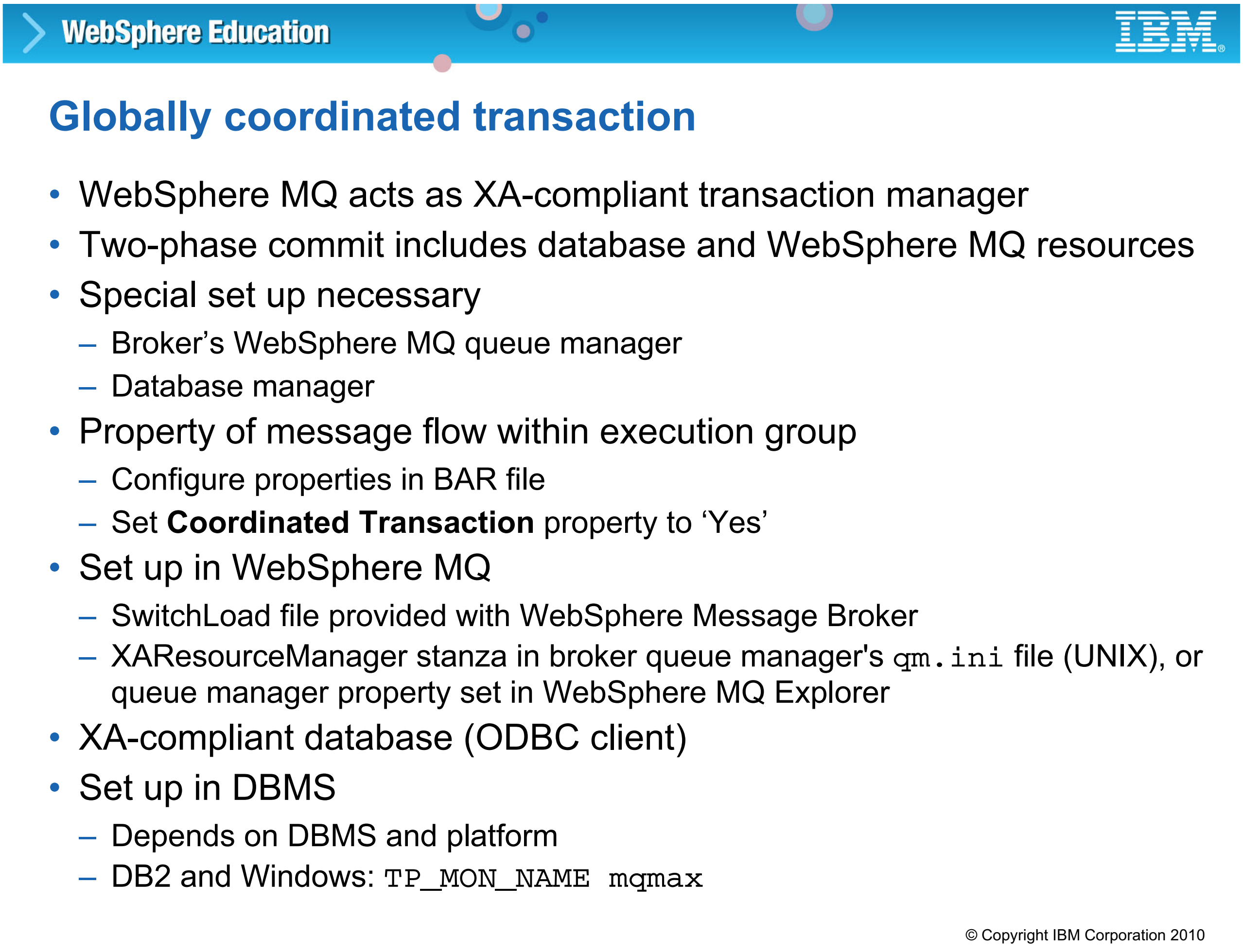
Note

This table is a summary only. You should refer to the Information Center for the latest information about platforms, version support, and exceptions.

Table 1: ODBC support

Platform	DB2	Microsoft SQL Server	Informix	Oracle	Sybase
AIX	Yes	Yes	Yes	Yes	Yes
HP-UX on Itanium	Yes	Yes	Yes	Yes	Yes
Linux on POWER	Yes	No	Yes	Yes	Yes
Linux on x86	Yes	Yes	Yes	Yes	Yes
Linux on x86-64	Yes	Yes	Yes	Yes	Yes
Linux on System z	Yes	No	Yes	Yes	No
Solaris on SPARC	Yes	Yes	Yes	Yes	Yes
Solaris on x86-64	Yes	No	Yes	Yes	Yes
Windows XP and Server 2003	Yes	Yes	Yes	Yes	Yes
Windows Vista	Yes	Yes	Yes	Yes	Yes
Windows Server 2008	Yes	Yes	Yes	Yes	Yes
z/OS	Yes	No	No	No	No

Transition statement — So how do you activate a globally coordinated transaction?



WebSphere Education

Globally coordinated transaction

- WebSphere MQ acts as XA-compliant transaction manager
- Two-phase commit includes database and WebSphere MQ resources
- Special set up necessary
  - Broker’s WebSphere MQ queue manager
  - Database manager
- Property of message flow within execution group
  - Configure properties in BAR file
  - Set **Coordinated Transaction** property to ‘Yes’
- Set up in WebSphere MQ
  - SwitchLoad file provided with WebSphere Message Broker
  - XAResourceManager stanza in broker queue manager's `qm.ini` file (UNIX), or queue manager property set in WebSphere MQ Explorer
- XA-compliant database (ODBC client)
- Set up in DBMS
  - Depends on DBMS and platform
  - DB2 and Windows: `TP_MON_NAME mqmax`

Figure 6-8. Globally coordinated transaction WM643 / VM6431.0

Notes:

Three actions are required to activate a globally coordinated transaction: one within the message flow (or possibly altered by the administrator in the `.bar` file), one within the WebSphere MQ queue manager, and one within the database manager.

The setup steps are described in detail in the WebSphere MQ documentation, such as the distributed platform WebSphere MQ System Administration guide.

Instructor notes:

Purpose — Point out that the assigned message flow, the queue manager, and the database manager, must be customized for globally coordinated transactions.

Details — None.

Additional information — None.

Transition statement — The discussion touched upon runtime errors but did not go into any detail. Now it is time to cover where runtime error information can be found.

6.2. Testing and problem determination tools

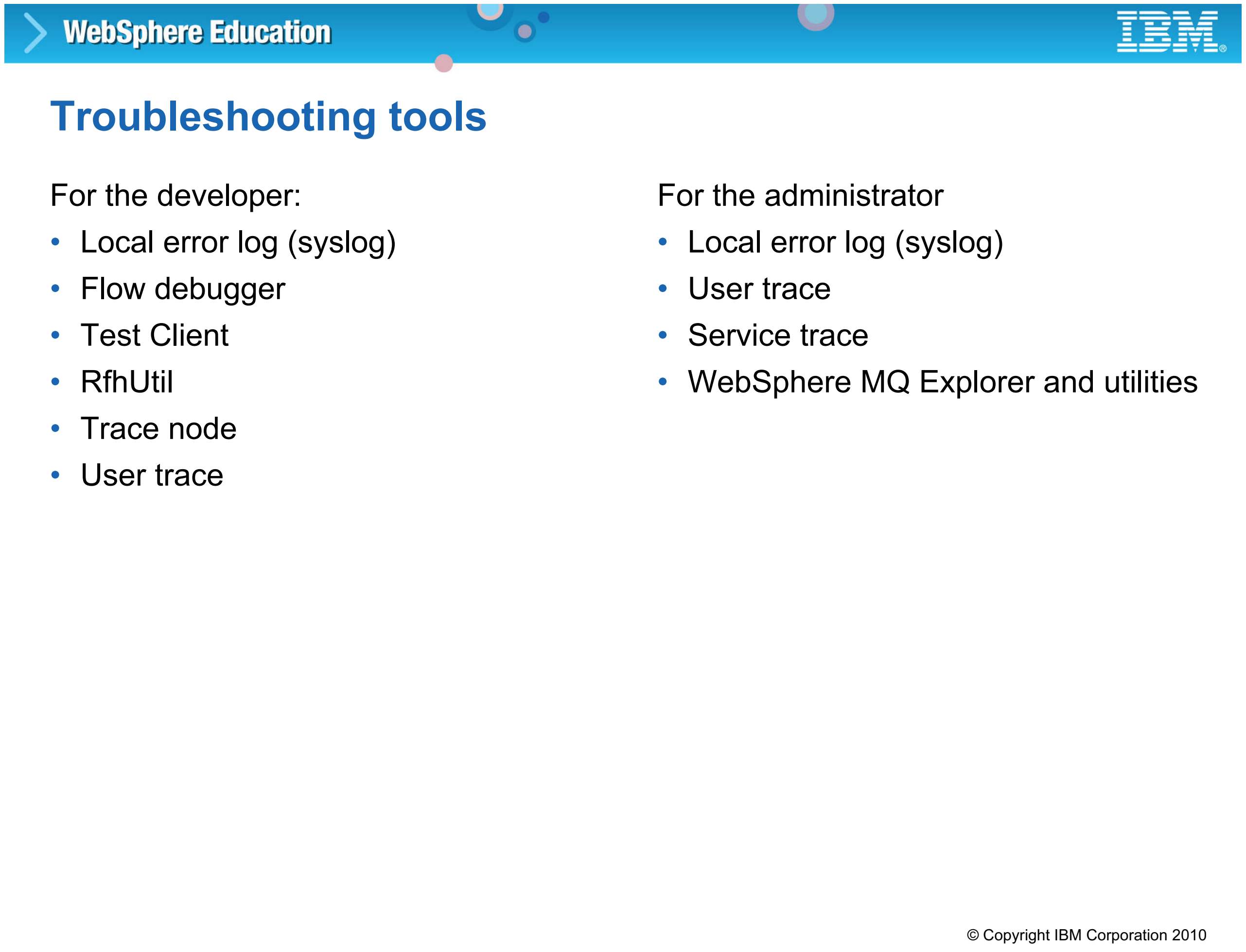
Instructor topic introduction

What students will do — Students learn about the basic problem determination tools available with the WebSphere Message Brokers and use them in a later exercise.

How students will do it — Students listen to a brief lecture about the available problem determination tools and use them in subsequent exercises.

What students will learn — Students will be able to differentiate between tools intended to be used in debugging message flows and tools designed for other purposes. They will also learn how to determine when the WebSphere Message Brokers are running as expected and when they are not.

How this will help students in their job — Students will be prepared to operate and evaluate a running WebSphere Message Broker using the logs and tools available, ensuring that there are no errors or problems.



WebSphere Education

Troubleshooting tools

For the developer:

- Local error log (syslog)
- Flow debugger
- Test Client
- RfhUtil
- Trace node
- User trace

For the administrator

- Local error log (syslog)
- User trace
- Service trace
- WebSphere MQ Explorer and utilities

Figure 6-9. Troubleshooting tools WM643 / VM6431.0

Notes:

As shown in the figure, there are a number of tools available for troubleshooting the broker and broker components, message flows, and runtime problems. Some of the tools, such as the local error log are useful to both developers and administrators.

Instructor notes:

Purpose — Identify some of the tools available to the developer and administrator to locate runtime problems.

Details — This topic describes the troubleshooting tools available to the administrator. This is not a comprehensive list as other operating-specific tools might be available.

Additional information —

Transition statement — A common location for looking for error information is the local error log.

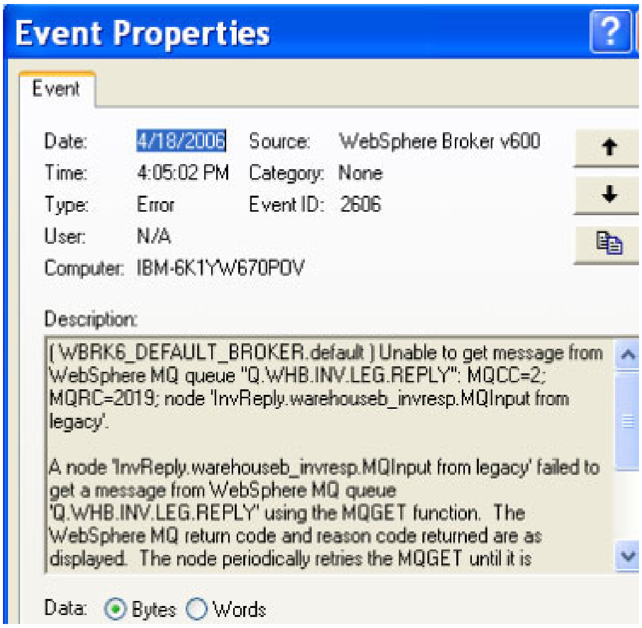
WebSphere Education

Local error log (syslog)

- Multiple messages per runtime error, must review all of them
  - Status of broker
  - Run time errors
  - Some database errors may be logged
- Always check after system startup, deploy, and during testing of message flows
- Location differs per platform:
  - Windows uses Event Viewer
  - UNIX location based on entry in `/etc/syslog.conf`

```
user.debug /var/mqsi/user.debuglog
user.err /var/mqsi/user.errlog
```

  - z/OS: Address space joblog and z/OS syslog



Date: 4/18/2006 Source: WebSphere Broker v600
Time: 4:05:02 PM Category: None
Type: Error Event ID: 2606
User: N/A
Computer: IBM-6K1YW670POV

Description:

(WBRK6_DEFAULT_BROKER.default) Unable to get message from WebSphere MQ queue "Q.WHB.INV.LEG.REPLY": MQCC=2; MQRC=2019; node 'InvReply.warehouseb_invresp.MQInput from legacy'.

A node 'InvReply.warehouseb_invresp.MQInput from legacy' failed to get a message from WebSphere MQ queue 'Q.WHB.INV.LEG.REPLY' using the MQGET function. The WebSphere MQ return code and reason code returned are as displayed. The node periodically retries the MQGET until it is

Data: ● Bytes ○ Words

Figure 6-10. Local error log (syslog) WM643 / VM6431.0

Notes:

The local error log should be the primary source of runtime problem determination, because all errors and warnings are recorded there. The WebSphere Message Broker Toolkit does ***not*** show any runtime errors in the broker, it only shows deployment errors.

Additionally, it is possible that errors beyond the broker can occur; for instance, with a database being referenced in a message flow. It is important to check the local error log regularly. Errors can pile up in the local error log and actually fill it if not acted on.

In Windows, WebSphere Message Broker automatically writes to the Event Viewer Application log. In UNIX, the syslog must be explicitly configured.

On non-Windows platforms, the syslog only contains the first line of the message, and the user is expected to look up the remainder of the message in the messages book.

Instructor notes:

Purpose — Remind students about the error log on each platform.

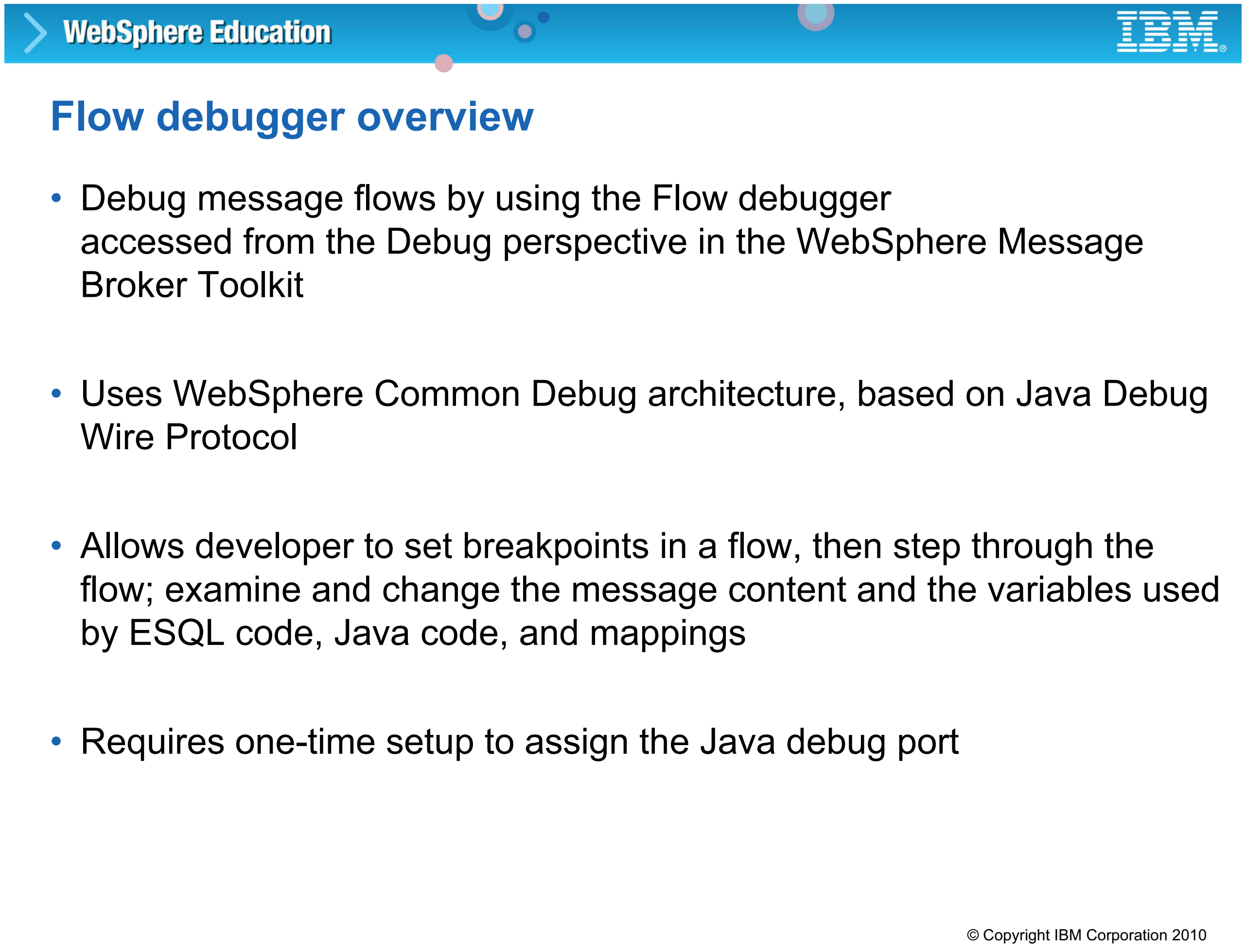
Details — None.

Additional information — In some rare cases, deploys look as if they have worked according to the toolkit log but due to database and other related problems, they have failed. This shows up in the syslog or Windows Event Viewer Application log. It is critical to get students to look at the log regularly.

Many UNIX systems provide a command-line utility, known as logger, to help test and refine the configuration of the syslog subsystem.

On UNIX, syslog entries are restricted in length and messages that are sent to the syslog are truncated by the new line character. To record a large amount of data in a log on UNIX, set the **Destination** property on the Trace node to **File** or **User Trace** instead of Local Error Log

Transition statement — The message flow debugger is a valuable tool for anyone who must analyze a message flow, node-by-node, or ESQL, line-by-line.



WebSphere Education

Flow debugger overview

- Debug message flows by using the Flow debugger accessed from the Debug perspective in the WebSphere Message Broker Toolkit
- Uses WebSphere Common Debug architecture, based on Java Debug Wire Protocol
- Allows developer to set breakpoints in a flow, then step through the flow; examine and change the message content and the variables used by ESQL code, Java code, and mappings
- Requires one-time setup to assign the Java debug port

Figure 6-11. Flow debugger overview WM643 / VM6431.0

Notes:

The Flow debugger can be used to step through a message flow and determine where a problem is occurring.

Instructor notes:

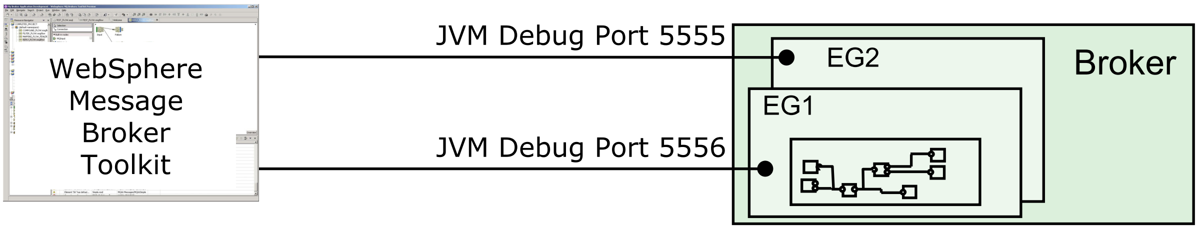
Purpose — Provide an overview of the message flow debugger.

Details — This slide provides overview information for the debugger.

Additional information — The architecture of the debugger changed in WebSphere Message Broker V6.1. Previously it used the Rational Agent Controller facility for debugging; now it uses a Common Debug Architecture used by other WebSphere products

Transition statement — Look at more introductory information about the debugger.

Flow debugger



- Eclipse-based debugging tool attaches to execution groups
  - Can attach to multiple flow engines at same time allowing for debugging of multiple flows in multiple execution groups from same workbench
  - One execution group can be debugged by only one workbench at a time
- Debugs flows, subflows, ESQL and Java code, maps
- Message flow must:
  - Be deployed and running
  - Loaded in the workspace

Figure 6-12. Flow debugger

Notes:

The flow debugger is visual interface for flow testing. If there is an exception in the flow, the debugger pauses automatically at the exception location. Breakpoints can also be added between nodes to allow interruption of flow execution at specific points.

The flow debugger operates in the Debug perspective in the toolkit and is based on the Java debugger. The flow debugger can be used with the test client.

Some one-time configuration is required in order to use the flow debugger:

- You must have a flow deployed.
- The Java (JVM) Debug Port must be set on each execution group that hosts a flow to be debugged. The execution group (or the broker) must be restarted before the change takes effect.
- Debugger configuration must select a DataFlowEngine to attach to (the execution group running the flow).
- Debugger configuration must point at the projects that contain source for the message flow.

Instructor notes:

Purpose — Introduce the debugger.

Details — The debugger is based on IBM Common Debug Architecture (CDA) used by other WebSphere products (WebSphere Integration Developer, WebSphere Enterprise Service Bus). CDA is a reliable and robust Java Debug Channel (Channel) transport technology based on standard Java Debug Wire Protocol (JDWP). If the broker runtime is separated from the Toolkit by a firewall, then the firewall should be configured to permit this port.

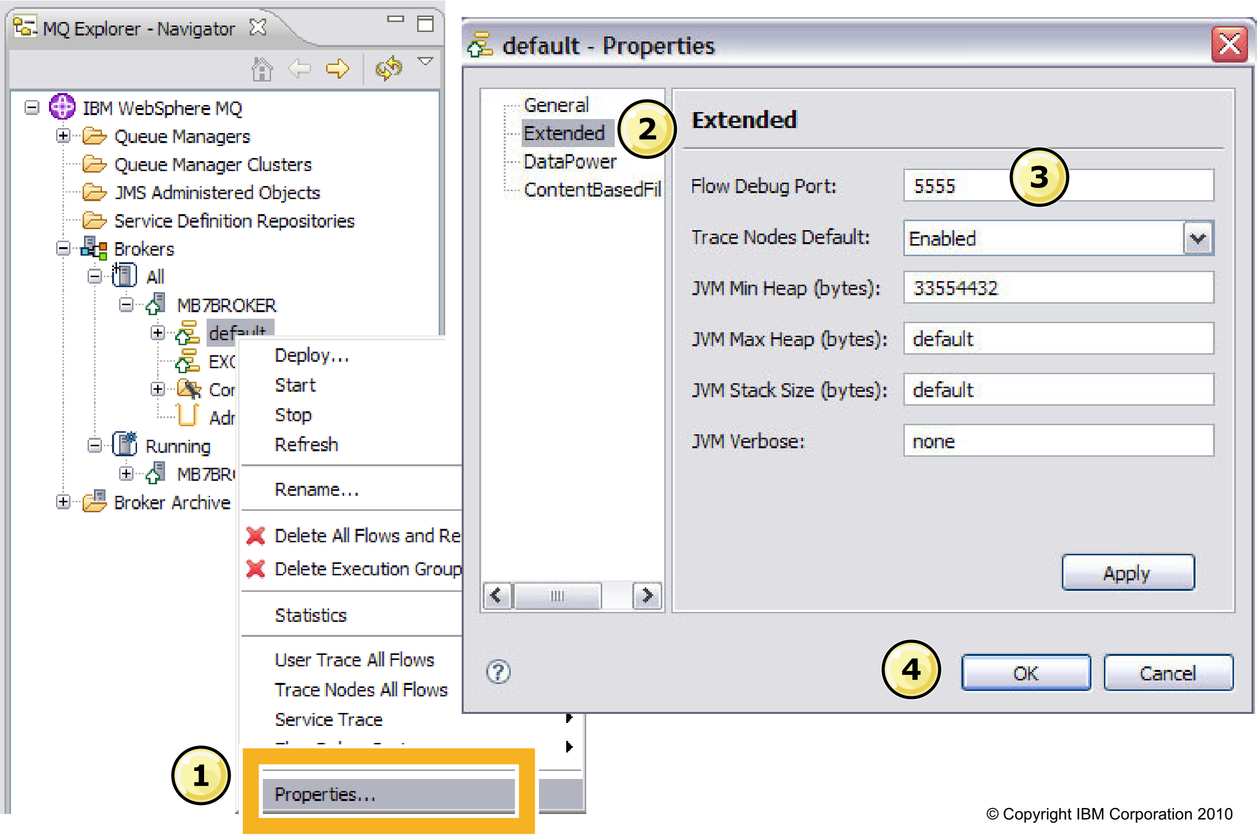
This facility is fully integrated into WebSphere Message Broker 7, and you do not have to install a separate product or component.

Channel uses a single fixed TCP/IP port and is compatible with firewalls

Additional information — The debugger has removed the dependency on Rational Agent Controller. This has been replaced by the Java Debug Channel, and this makes the configuration of the debug function easier than in earlier versions. This needs the configuration of a single TCP/IP port on the runtime component.

Transition statement — Before you use the debugger for the first time, you must perform some one-time setup tasks.

Configuring the flow debug port



Extended

Flow Debug Port: 5555

Trace Nodes Default: Enabled

JVM Min Heap (bytes): 33554432

JVM Max Heap (bytes): default

JVM Stack Size (bytes): default

JVM Verbose: none

© Copyright IBM Corporation 2010

Figure 6-13. Configuring the flow debug port WM643 / VM6431.0

Notes:

Before using the debugger, you must configure the required port on the broker’s execution group. You can configure the port in the WebSphere Message Broker Explorer Navigator view:

1. Right-click the execution group, and select **Properties** from the menu.
2. In the execution group **Properties** window, click **Extended** to show the extended properties.
3. Enter the **Flow Debug Port** number.
4. Click **OK** (this example shows port 5555 being used.) When you click **OK**, the execution group is automatically restarted.

Instructor notes:

Purpose — Demonstrate one-time setup for the debugger.

Details — The commands to restart the broker of the execution group must be issued from the broker system's command console, and special authority is needed.

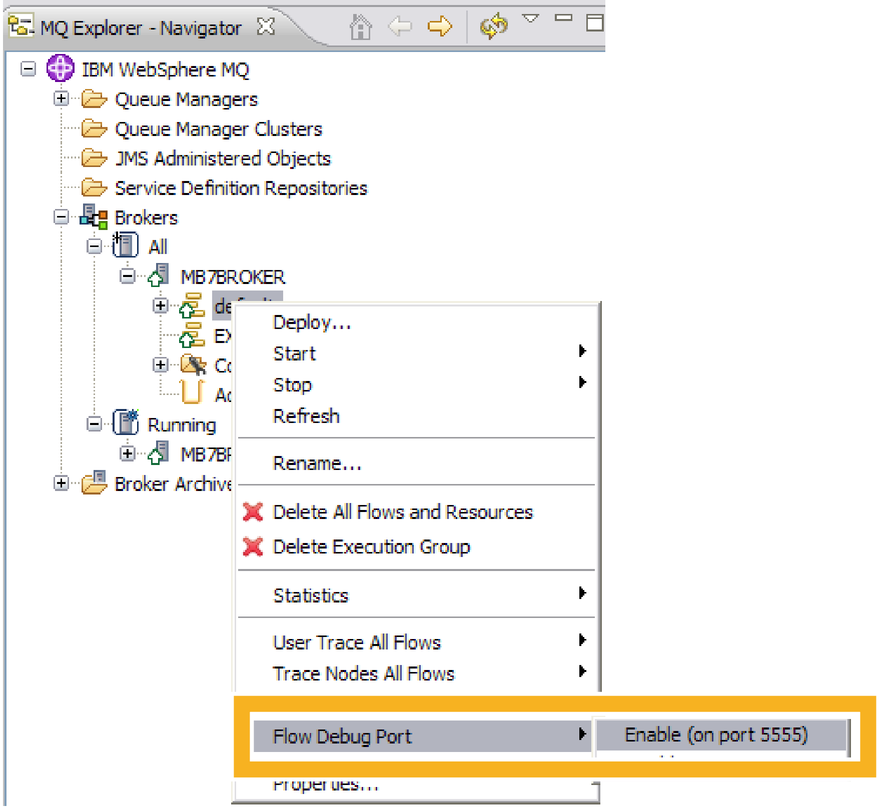
Additional information —

Transition statement — How do you set up your workspace environment for the debugger?

WebSphere Education

IBM

Enabling the flow debug port



© Copyright IBM Corporation 2010

Figure 6-14. Enabling the flow debug port

WM643 / VM6431.0

Notes:

The flow debug port must be enabled before the message flow can be debugged.

To enable the flow debug port, right-click the execution group in the WebSphere Message Broker Explorer navigator and select **Flow Debug Port > Enable** from the menu.

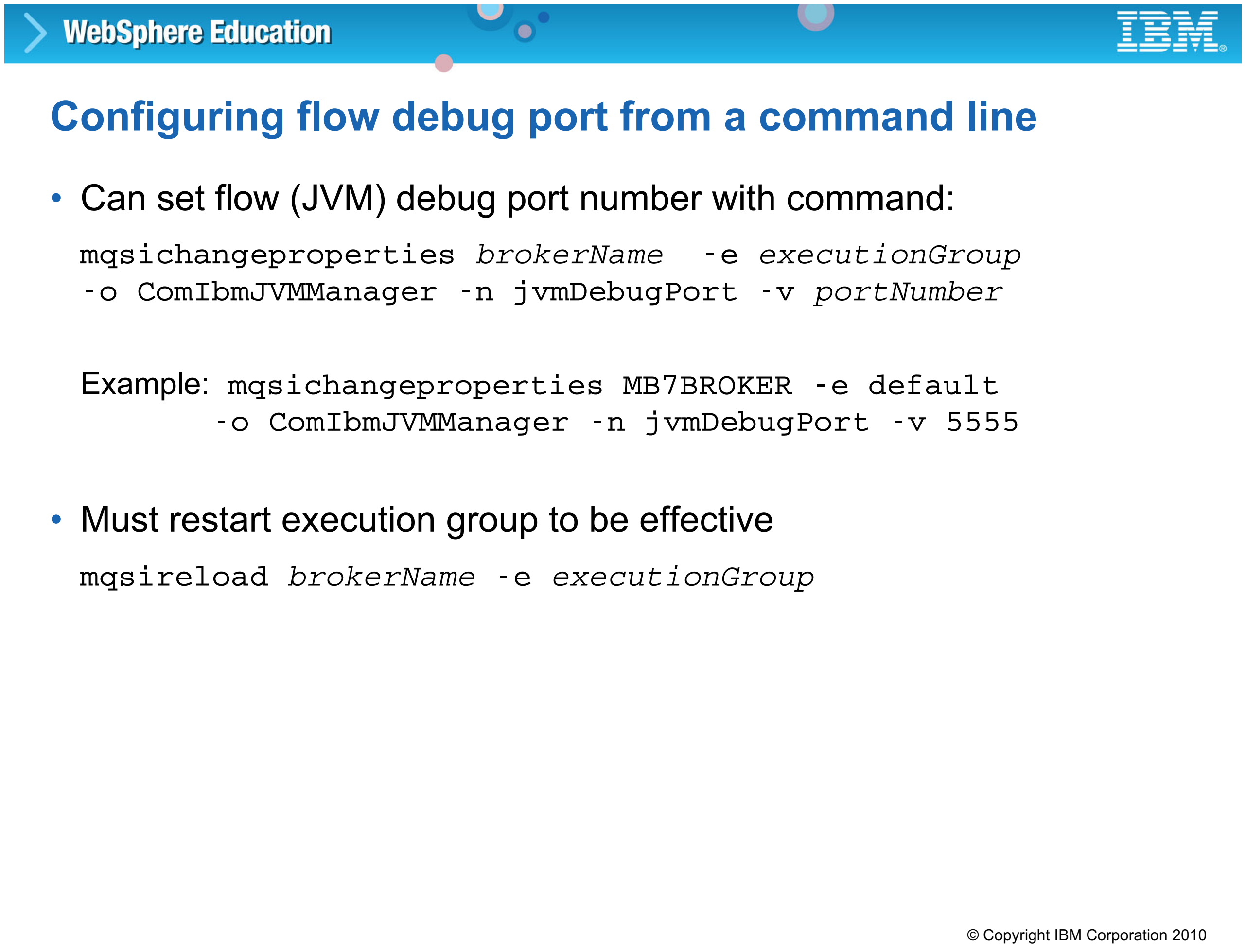
Instructor notes:

Purpose — Demonstrate enabling a flow debug port.

Details — This topic is showing debug configuration in the WebSphere Message Broker Explorer as this is the primary interface for administrators. Mention that developers have the same capabilities in WebSphere Message Broker Toolkit.

Additional information —

Transition statement — The flow debug port can also be specified using the `mqsichangeproperties` command.



WebSphere Education

Configuring flow debug port from a command line

- Can set flow (JVM) debug port number with command:

```
mqsichangeproperties brokerName  -e executionGroup
-o ComIbmJVMManager -n jvmDebugPort -v portNumber
```

Example:
```
mqsichangeproperties MB7BROKER -e default
       -o ComIbmJVMManager -n jvmDebugPort -v 5555
```

- Must restart execution group to be effective

```
mqsireload brokerName -e executionGroup
```

Figure 6-15. Configuring flow debug port from a command line WM643 / VM6431.0

Notes:

As shown in the figure, the flow debug port can also be specified using the `mqsichangeproperties` command line.

The change of properties becomes active after a restart of the execution group, or the entire broker. You can use the `mqsireload` command, as shown in the figure, or the `mqsistop` and `mqsistart` commands.

Instructor notes:

Purpose — As an alternative, use WebSphere Message Broker commands to define the flow debug port and restart the execution group.

Details —

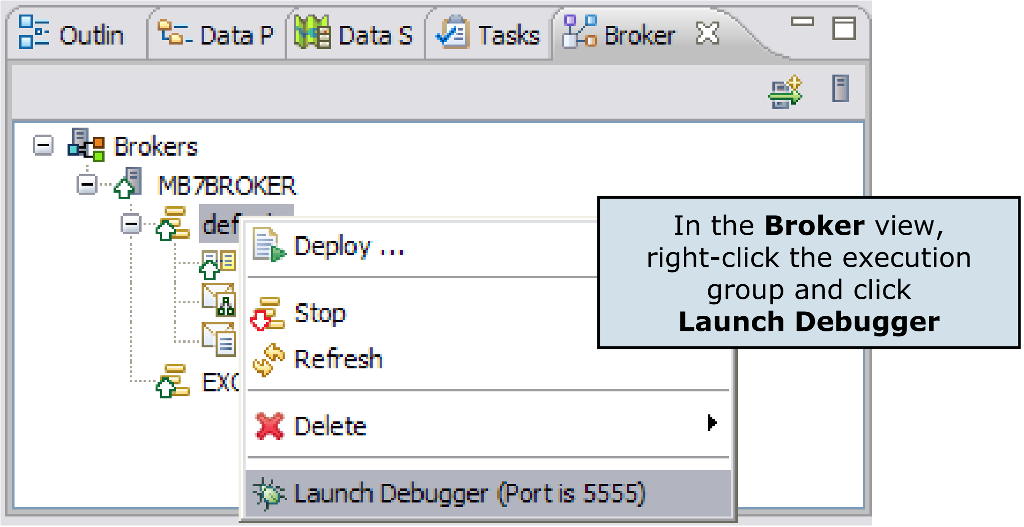
Additional information —

Transition statement — After the debugger is configured, you can use it for debugging a message flow.

WebSphere Education IBM.

Launching the Flow debugger

- If you used the `mqsichangeproperties` or WebSphere Message Broker Explorer to configure the flow debug port, attach the Flow debugger to the execution group in the WebSphere Message Broker Toolkit



In the **Broker** view, right-click the execution group and click **Launch Debugger**

© Copyright IBM Corporation 2010

Figure 6-16. Launching the flow debugger WM643 / VM6431.0

Notes:

If you used the `mqsichangeproperties` command or the WebSphere Message Broker Explorer to configure the flow debug port, you must use the **Broker** view in the WebSphere Message Broker Toolkit to attach the flow debugger to the execution group.

Right-click the execution group with which you want to work, select **Launch Debugger**, and click **OK** to attach the debugger to the selected execution group.

Instructor notes:

Purpose — Show the extra step that is required to launch the debugger if the flow debug port was configured using the command line or WebSphere Message Broker Explorer.

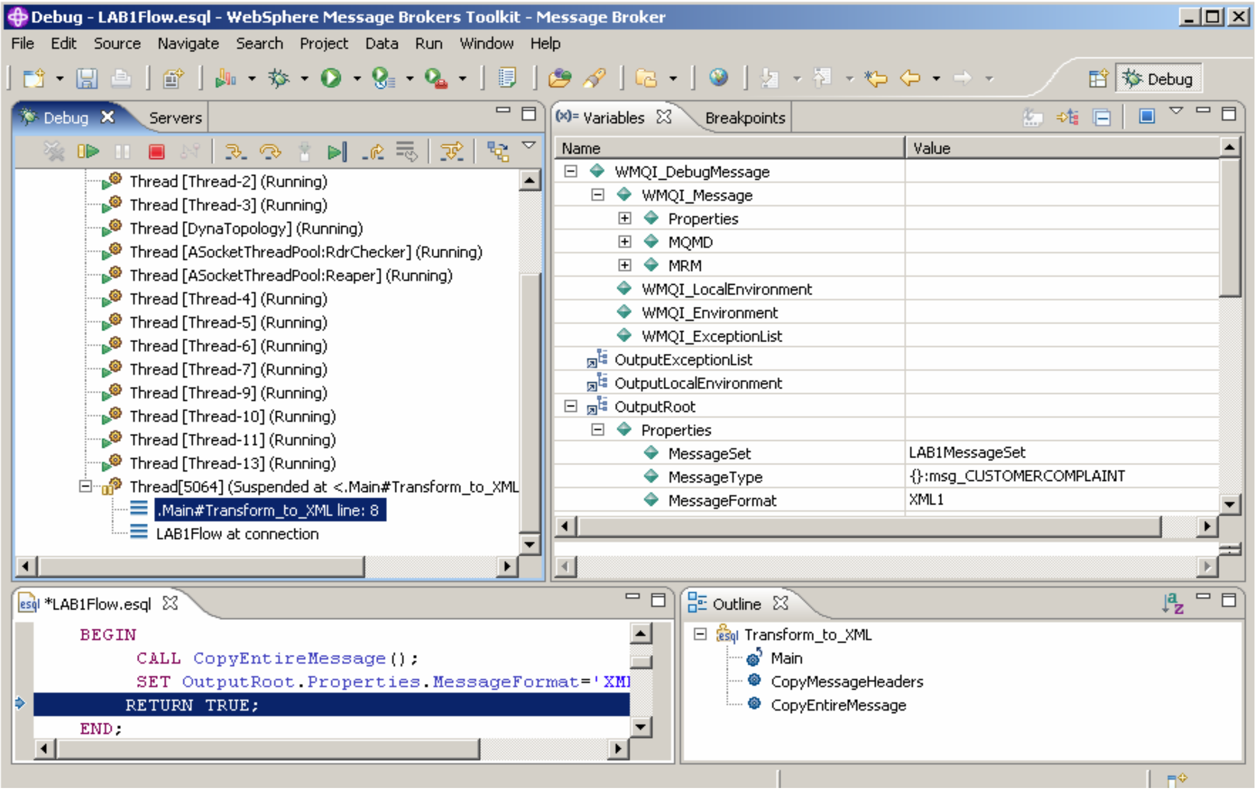
Details —

Additional information —

Transition statement — Here is an example of the debugger in action.

WebSphere Education

Flow debugger example



Debug - LAB1Flow.esql - WebSphere Message Brokers Toolkit - Message Broker
File Edit Source Navigate Search Project Data Run Window Help
Debug
Debug
Servers
Variables
Breakpoints
*LAB1Flow.esql
Outline

```
Thread [Thread-2] (Running)
Thread [Thread-3] (Running)
Thread [DynaTopology] (Running)
Thread [ASocketThreadPool:RdrChecker] (Running)
Thread [ASocketThreadPool:Reaper] (Running)
Thread [Thread-4] (Running)
Thread [Thread-5] (Running)
Thread [Thread-6] (Running)
Thread [Thread-7] (Running)
Thread [Thread-9] (Running)
Thread [Thread-10] (Running)
Thread [Thread-11] (Running)
Thread [Thread-13] (Running)
Thread[5064] (Suspended at <.Main#Transform_to_XML
    .Main#Transform_to_XML line: 8
    LAB1Flow at connection
```

Name	Value
WMQI_DebugMessage	
WMQI_Message	
Properties	
MQMD	
MRM	
WMQI_LocalEnvironment	
WMQI_Environment	
WMQI_ExceptionList	
OutputExceptionList	
OutputLocalEnvironment	
OutputRoot	
Properties	
MessageSet	LAB1MessageSet
MessageType	{}:msg_CUSTOMERCOMPLAINT
MessageFormat	XML1

```
BEGIN
        CALL CopyEntireMessage();
        SET OutputRoot.Properties.MessageFormat='XMI
      RETURN TRUE;
END;
```

```
Transform_to_XML
    Main
    CopyMessageHeaders
    CopyEntireMessage
```

© Copyright IBM Corporation 2010

Figure 6-17. Flow debugger example WM643 / VM6431.0

Notes:

The message flow runs until it encounters a breakpoint previously defined in the message flow. When a breakpoint is encountered, control is returned to you. At this point, the current content of message trees can be examined and modified.

If the next node to be run supports source code (the mapping node, compute node (ESQL), or JavaCompute node), you can step into that node and debug at the source level. You can also step into and debug a subflow.

Execution can be resumed in several modes:

- **Step Over** runs the next node, then takes an implicit breakpoint.
- **Resume Execution** runs to the next defined breakpoint.
- **Run To Completion** ignores any configured breakpoints and runs to completion.

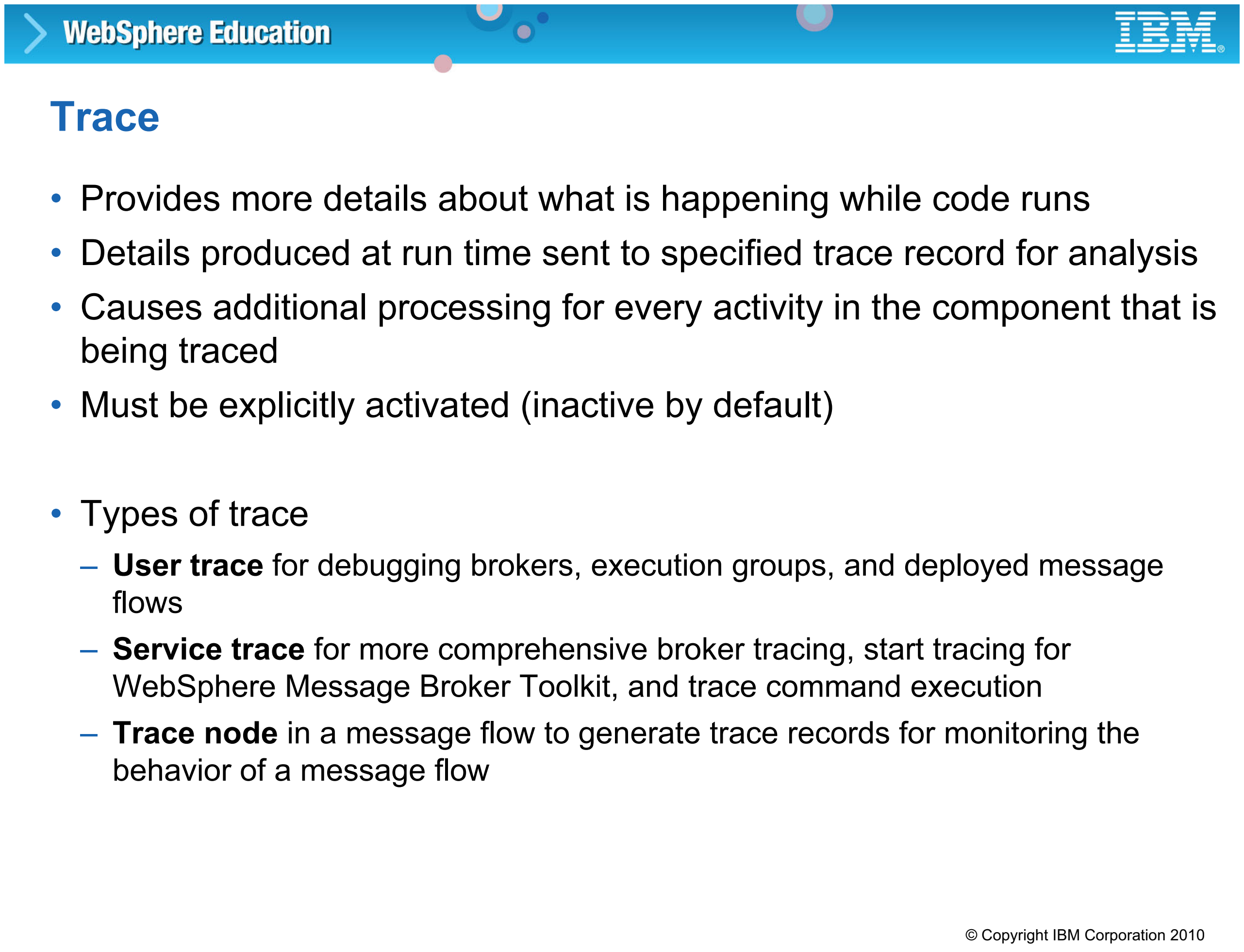
Instructor notes:

Purpose — Show how the debugger displays message trees and breakpoints.

Details —

Additional information — Students will practice using the debugger in the next lab.

Transition statement — Another set of useful debugging tools are the trace facilities available in WebSphere Message Broker.



WebSphere Education

Trace

- Provides more details about what is happening while code runs
- Details produced at run time sent to specified trace record for analysis
- Causes additional processing for every activity in the component that is being traced
- Must be explicitly activated (inactive by default)

- Types of trace
  - **User trace** for debugging brokers, execution groups, and deployed message flows
  - **Service trace** for more comprehensive broker tracing, start tracing for WebSphere Message Broker Toolkit, and trace command execution
  - **Trace node** in a message flow to generate trace records for monitoring the behavior of a message flow

Figure 6-18. Trace WM643 / VM6431.0

Notes:

If you cannot get enough information about a particular problem from the entries that are available in the various logs, the next troubleshooting method to consider is using trace. Trace provides more details about what is happening while code executes. The information produced from trace is sent to a specified trace record, so that you or IBM support personnel can analyze it to discover the cause of your problem.

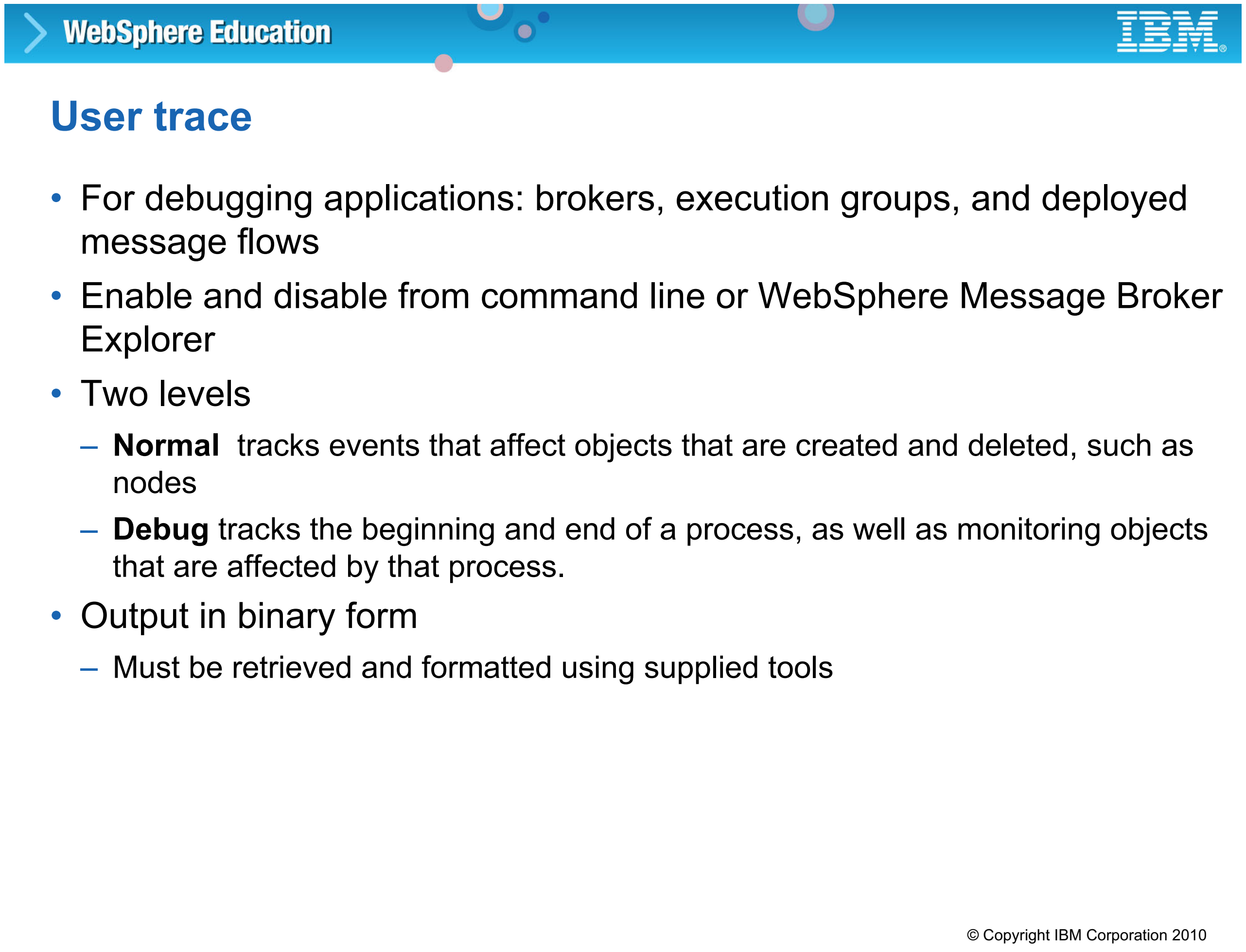
Instructor notes:

Purpose — Overview of trace options available for WebSphere Message Broker

Details — User trace, service trace, trace nodes are described in more detail on the next pages.

Additional information —

Transition statement — Next: User trace



WebSphere Education

User trace

- For debugging applications: brokers, execution groups, and deployed message flows
- Enable and disable from command line or WebSphere Message Broker Explorer
- Two levels
  - **Normal** tracks events that affect objects that are created and deleted, such as nodes
  - **Debug** tracks the beginning and end of a process, as well as monitoring objects that are affected by that process.
- Output in binary form
  - Must be retrieved and formatted using supplied tools

Figure 6-19. User trace WM643 / VM6431.0

Notes:

You typically use a user trace for debugging message flows, but you can trace brokers and execution groups as well.

When you set user tracing to `on`, you are causing additional processing for every activity in the component you are tracing. Large quantities of data are generated during execution. You must therefore expect to see some affect on performance while tracing is active. However, you can limit this additional processing by being selective about what you trace, and by restricting the time during which the trace is active.

You can activate user trace from the WebSphere Message Broker Explorer, and from a line command (`mqsichangetrace`).

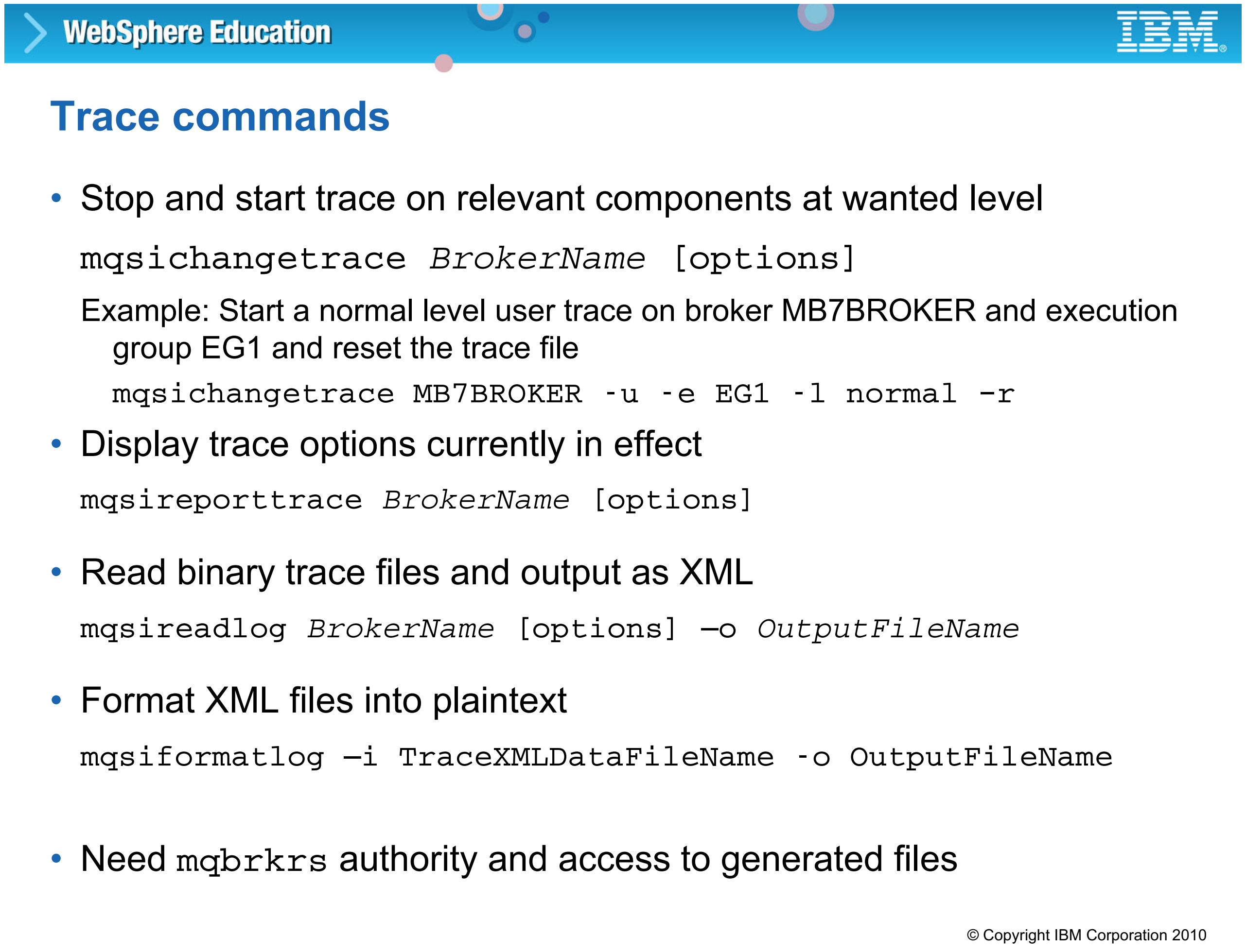
Instructor notes:

Purpose — Identify and discuss user trace operations and options.

Details — Remind students that after user trace is turned on, it remains on until explicitly turned off. With user trace running, performance is affected, especially if tracing a broker which produces significant amounts of trace data.

Additional information —

Transition statement — See how user trace can be activated from the command line.



WebSphere Education

Trace commands

- Stop and start trace on relevant components at wanted level

  ```
  mqsichangetrace BrokerName [options]
  ```

  Example: Start a normal level user trace on broker MB7BROKER and execution group EG1 and reset the trace file

  ```
  mqsichangetrace MB7BROKER -u -e EG1 -l normal -r
  ```

- Display trace options currently in effect

  ```
  mqsireporttrace BrokerName [options]
  ```

- Read binary trace files and output as XML

  ```
  mqsireadlog BrokerName [options] —o OutputFileName
  ```

- Format XML files into plaintext

  ```
  mqsiformatlog —i TraceXMLDataFileName -o OutputFileName
  ```

- Need `mqbrkrs` authority and access to generated files

Figure 6-20. Trace commands WM643 / VM6431.0

Notes:

You can activate user trace from the **`mqsichangetrace`** command line.

The `mqsireporttrace` command displays the trace options currently in effect.

After trace data is recorded (in a binary form), you can use the **`mqsireadlog`** command to read the trace. The output is a file in XML format. In order to get the XML data in plain text, you must run an additional command, **`mqsiformatlog`**.

These commands are typically run in series, and for simplicity, can easily be packaged into a script.

To determine currently active trace status, from the command console, use the **`mqsireporttrace`** command

Instructor notes:

Purpose — Describe the trace commands

Details —

Additional information —

Transition statement — You can also activate a user trace from WebSphere Message Broker Explorer.

WebSphere Education

Activating a user trace from WebSphere Message Broker Explorer

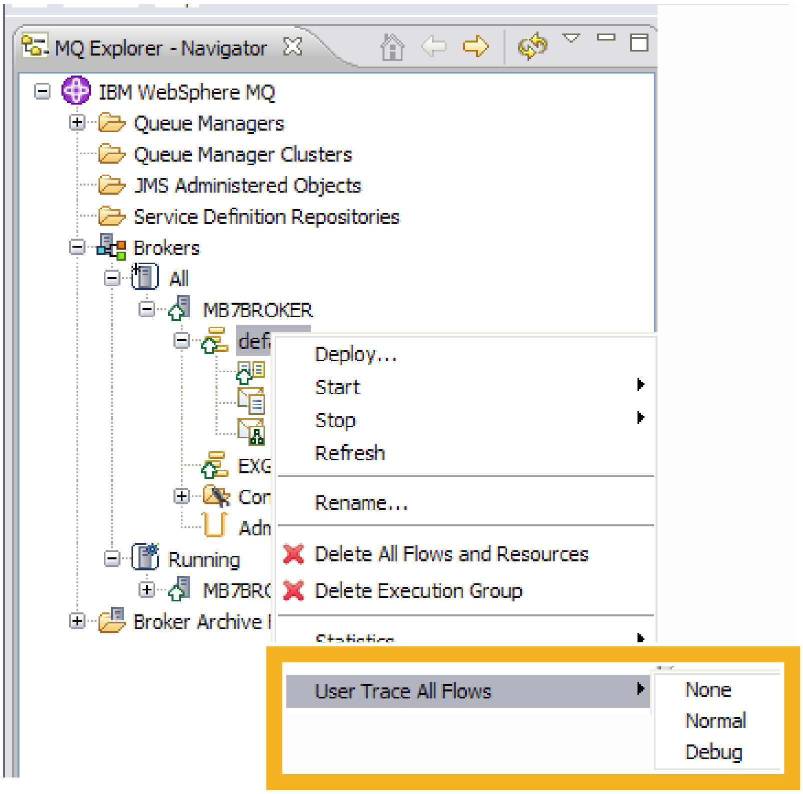


Figure 6-21. Activating a user trace from WebSphere Message Broker Explorer WM643 / VM6431.0

Notes:

From WebSphere Message Broker Explorer, you can enable a user trace on all flows, by selecting the execution group, or on a specific flow, by selecting a specific message flow.

There are two levels of tracing:

- **Normal** provides a basic level of trace information.
- **Debug** provides a more comprehensive trace.

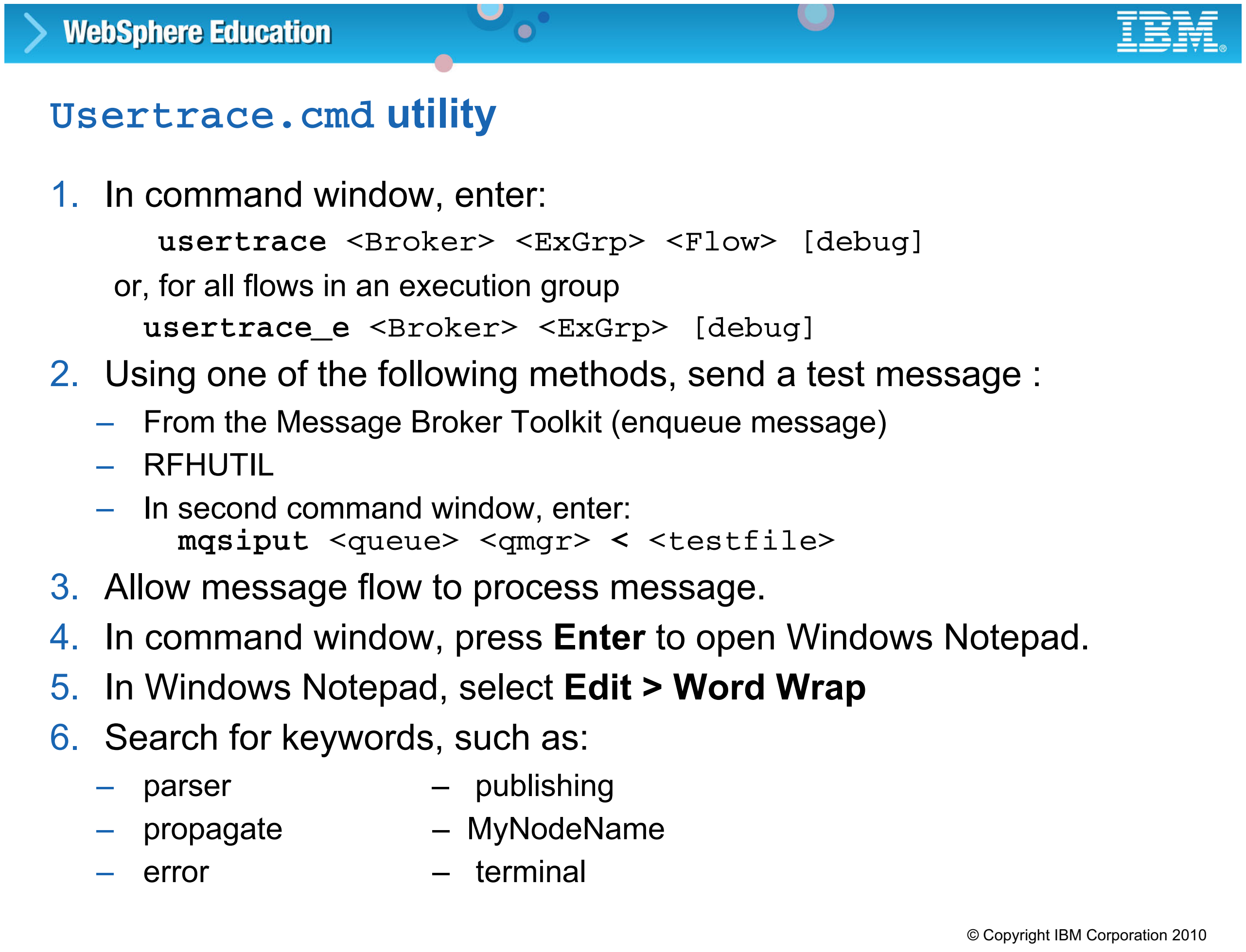
Instructor notes:

Purpose — Show how to activate a user trace from WebSphere Message Broker Explorer.

Details — Consider a toolkit demonstration to show how user trace can be turned on for a message flow. Point out the icon change (yellow triangle indicator) to show that user trace is active, and the entry in the **Alerts** view (`Message Flow is tracing at level....`). Send a message through the flow and format the trace data using **`GetTrace.cmd`** to show the results.

Additional information —

Transition statement — In this course, you will use two classroom utilities to create and read trace data.



WebSphere Education

Usertrace.cmd utility

1. In command window, enter:

```
    usertrace <Broker> <ExGrp> <Flow> [debug]
```

   or, for all flows in an execution group

```
  usertrace_e <Broker> <ExGrp> [debug]
```

2. Using one of the following methods, send a test message :
   - From the Message Broker Toolkit (enqueue message)
   - RFHUTIL
   - In second command window, enter:

```
     mqsiput <queue> <qmgr> < <testfile>
```

3. Allow message flow to process message.
4. In command window, press **Enter** to open Windows Notepad.
5. In Windows Notepad, select **Edit > Word Wrap**
6. Search for keywords, such as:
   - parser
   - propagate
   - error
   - publishing
   - MyNodeName
   - terminal

Figure 6-22. Usertrace.cmd utility WM643 / VM6431.0

Notes:

The `usertrace.cmd` utility activates a user trace for a message flow. The `usertrace_e.cmd` utility traces an execution group. Both of these utilities are working examples of command-line scripts.

The figure provides some guidance on how to run a user trace and use the trace results using the utilities supplied in the classroom.

The `usertrace.cmd` is *not* supplied with the product. It is a utility for this class. It is a simple command script which you can easily build yourself.

Another useful utility, `mqsiput`, is available as WebSphere Message Broker SupportPac IH02.

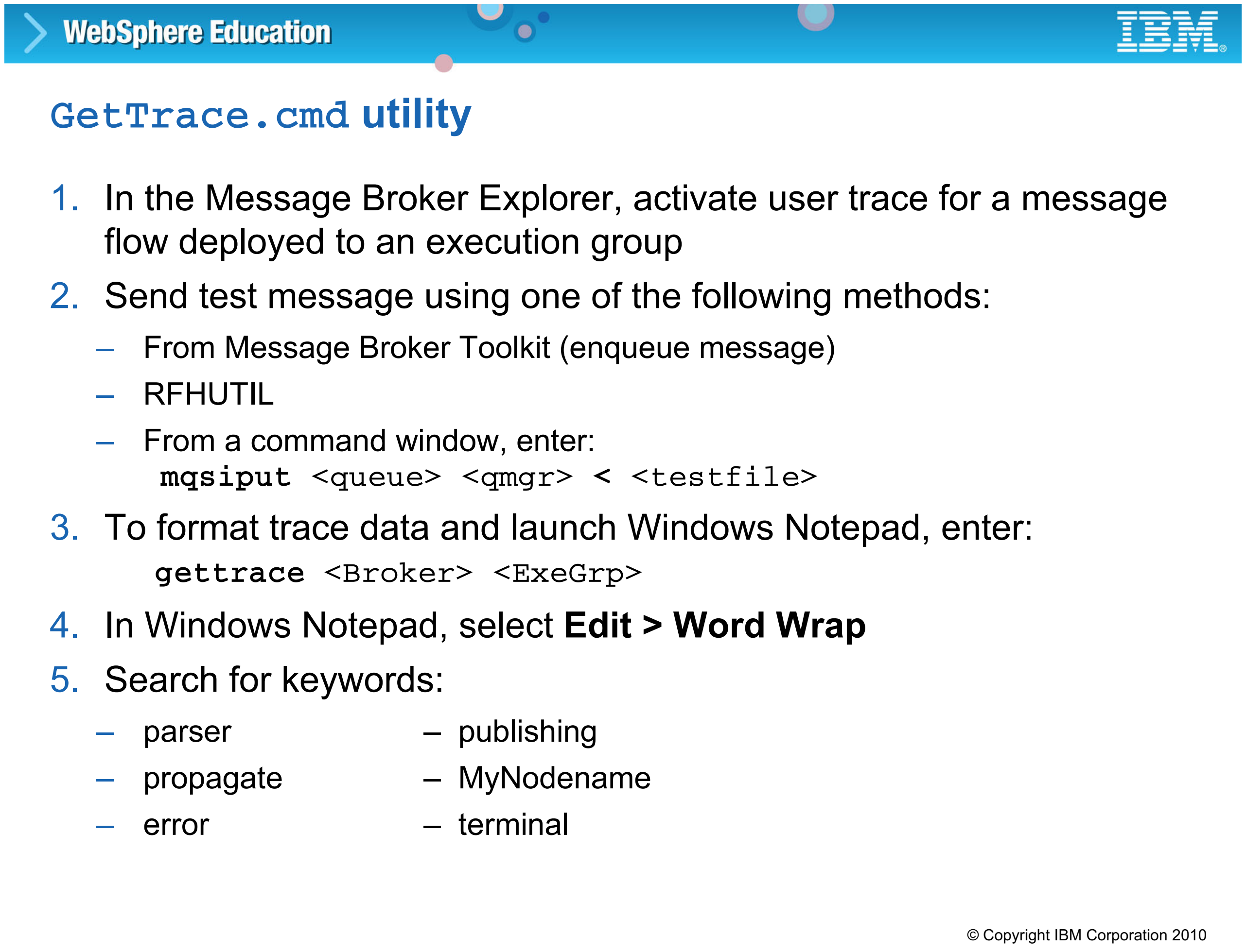
Instructor notes:

Purpose — Explain the UserTrace.cmd classroom utility.

Details — Usertrace.cmd is a utility provided for this course to simplify tracing. It is in the Tools directory (`C:\Labs\Tools`) in the lab files. Do urge students to use it if they seem reluctant to do so, certainly at the beginning of the course. User trace really helps to understand what the broker and message flows are doing and where messages are routed.

Additional information — Usertrace_e.cmd is beneficial in the classroom environment. It presumes a single flow running in a named execution group. Therefore, there is no need to identify (that is, type) an explicit message flow name (a constant source of typographical errors). However, it should not be used with an execution group running many message flows.

Transition statement — Although you can start trace from the WebSphere Message Broker Toolkit, it results in binary output. There is another utility available to format the trace output.



WebSphere Education

GetTrace.cmd utility

1. In the Message Broker Explorer, activate user trace for a message flow deployed to an execution group
2. Send test message using one of the following methods:
   - From Message Broker Toolkit (enqueue message)
   - RFHUTIL
   - From a command window, enter:
     ```
     mqsiput <queue> <qmgr> < <testfile>
     ```
3. To format trace data and launch Windows Notepad, enter:
   ```
   gettrace <Broker> <ExeGrp>
   ```
4. In Windows Notepad, select **Edit > Word Wrap**
5. Search for keywords:
   - parser
   - propagate
   - error
   - publishing
   - MyNodename
   - terminal

Figure 6-23. GetTrace.cmd utility WM643 / VM6431.0

Notes:

The `GetTrace.cmd` utility converts the binary trace data to XML and reformats the XML to plain text. `GetTrace.cmd` is nothing more than a container for `mqsireadlog` and `mqsiformatlog`.

The `GetTrace.cmd` is *not* supplied with WebSphere Message Broker and is not an IBM supported utility.

The `GetTrace.cmd` utility is run from a WebSphere Message Broker command console, not from a Windows command prompt.

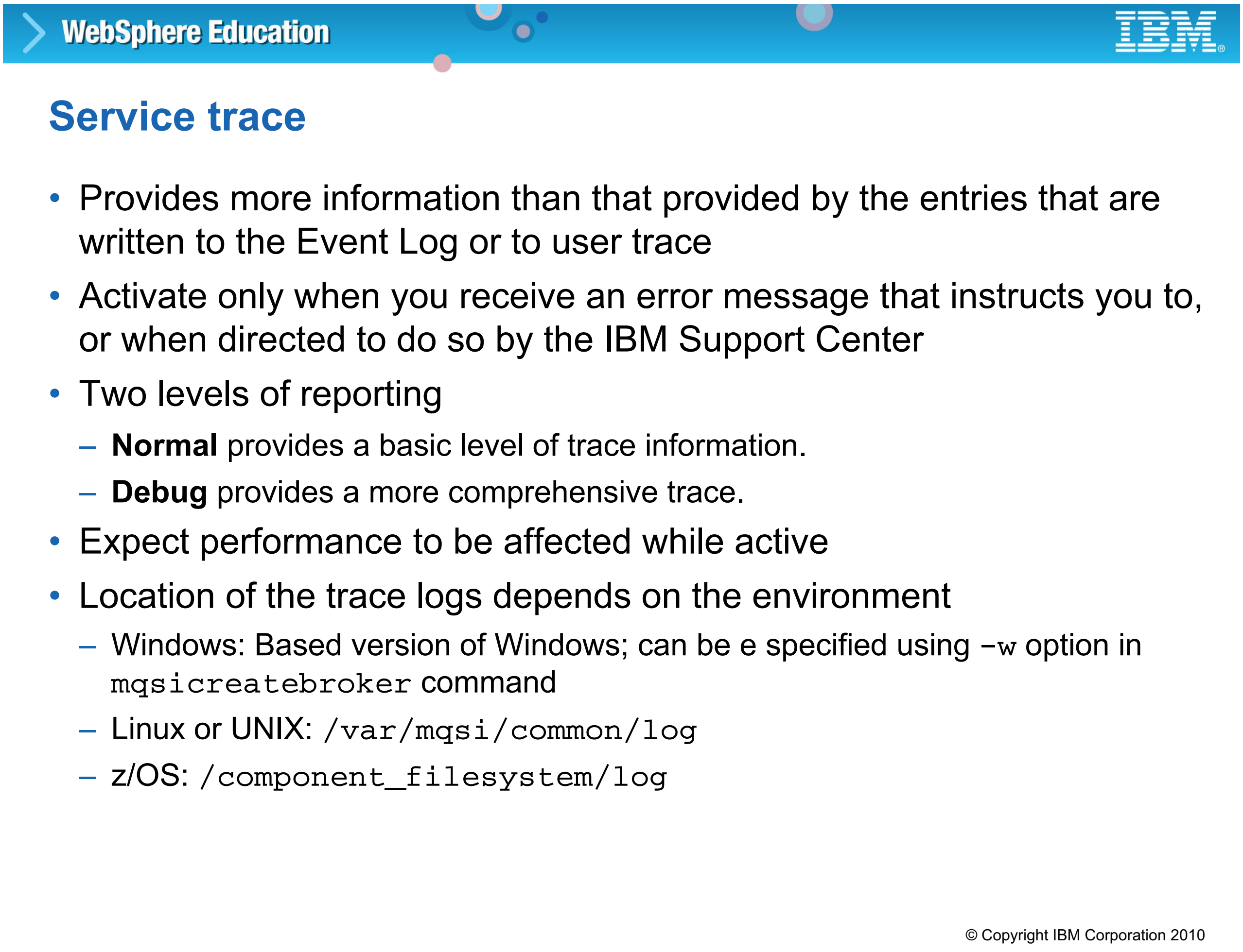
Instructor notes:

Purpose — Introduce the `GetTrace.cmd` utility that can be used in the exercises.

Details — None.

Additional information — None.

Transition statement — Next: Service trace



WebSphere Education
IBM

Service trace

- Provides more information than that provided by the entries that are written to the Event Log or to user trace
- Activate only when you receive an error message that instructs you to, or when directed to do so by the IBM Support Center
- Two levels of reporting
  - **Normal** provides a basic level of trace information.
  - **Debug** provides a more comprehensive trace.
- Expect performance to be affected while active
- Location of the trace logs depends on the environment
  - Windows: Based version of Windows; can be e specified using `-w` option in `mqsicreatebroker` command
  - Linux or UNIX: `/var/mqsi/common/log`
  - z/OS: `/component_filesystem/log`

Figure 6-24. Service trace WM643 / VM6431.0

Notes:

With service trace, you can activate more comprehensive broker tracing, and start tracing for the WebSphere Message Broker Toolkit. You can also trace the execution of all the commands, including the trace commands themselves.

When you activate service trace, you cause additional processing for every activity in the component that you are tracing. Large quantities of data are generated by the components. Expect performance to be affected while service trace is active. You can limit this additional processing by being selective about what you trace, and by restricting the time during which trace is active.

The location of the trace log depends on the environment. If you have set the work path by using the `-w` parameter of the `mqsicreatebroker` command, the location is `workpath\log`.

If you have not specified the broker work path, the default location is `%ALLUSERSPROFILE%\Application Data\IBM\MQSI\common\log` where `%ALLUSERSPROFILE%` is the environment variable that defines the system working directory.

Instructor notes:

Purpose — Provide an overview of service trace.

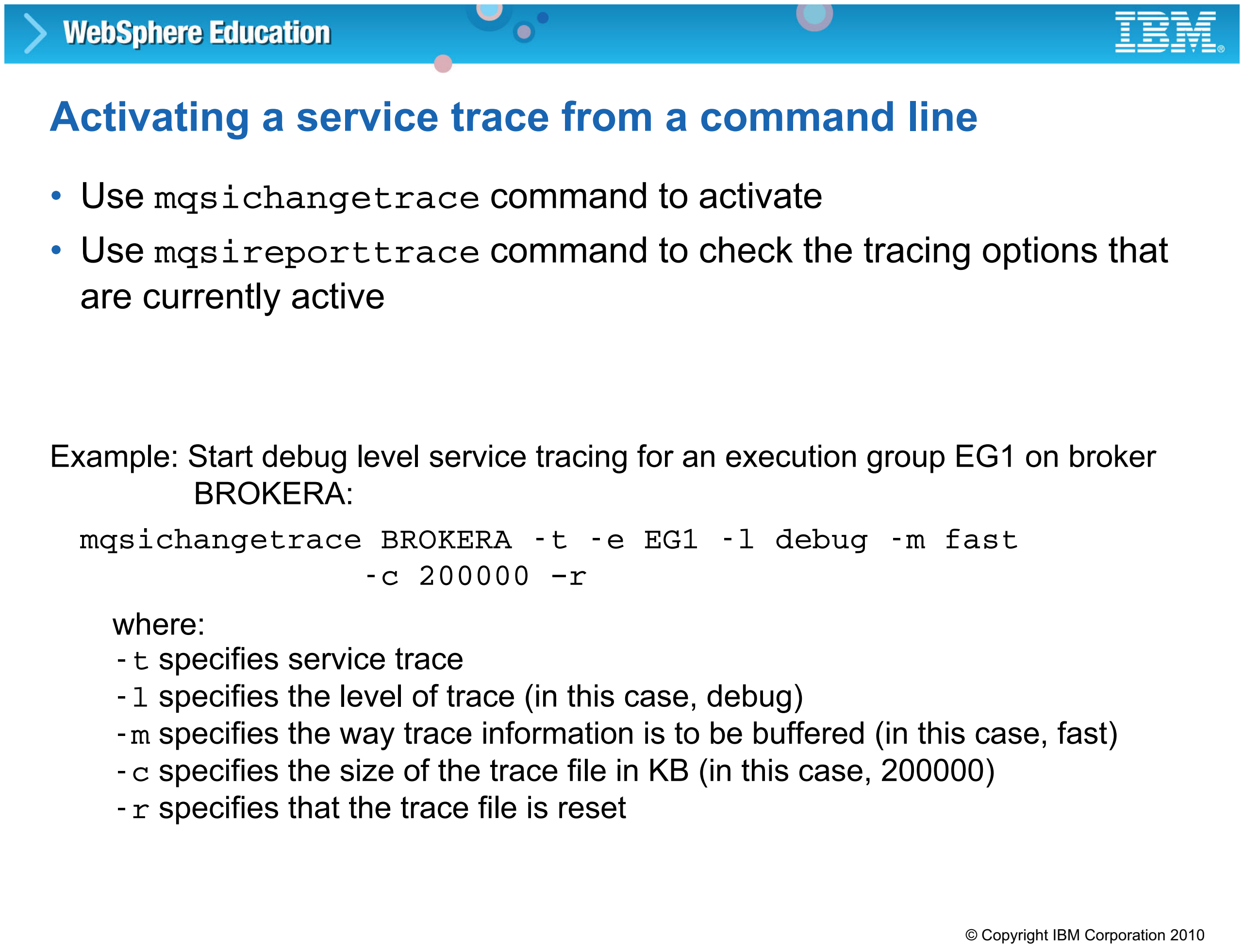
Details — The default directory depends on the operating system:

On Windows XP and Windows Server 2003: `C:\Documents and Settings\All Users\Application Data\IBM\MQSI\common\log`

On Windows Vista and Windows Server 2008: `C:\ProgramData\IBM\MQSI\common\log`.

Additional information —

Transition statement — Like a user trace, you can activate a service trace from a command line.



WebSphere Education

Activating a service trace from a command line

- Use `mqsichangetrace` command to activate
- Use `mqsireporttrace` command to check the tracing options that are currently active

Example: Start debug level service tracing for an execution group EG1 on broker BROKERA:

```
mqsichangetrace BROKERA -t -e EG1 -l debug -m fast
                  -c 200000 -r
```

where:
`-t` specifies service trace
`-l` specifies the level of trace (in this case, debug)
`-m` specifies the way trace information is to be buffered (in this case, fast)
`-c` specifies the size of the trace file in KB (in this case, 200000)
`-r` specifies that the trace file is reset

Figure 6-25. Activating a service trace from a command line WM643 / VM6431.0

Notes:

The `mqsichangetrace` command can be used to activate a service trace.

The `mqsireporttrace` command can be used to check the currently active tracing options.

The figure shows an example of the command to activate a service trace.

Instructor notes:

Purpose — Show the command line for activating a service trace.

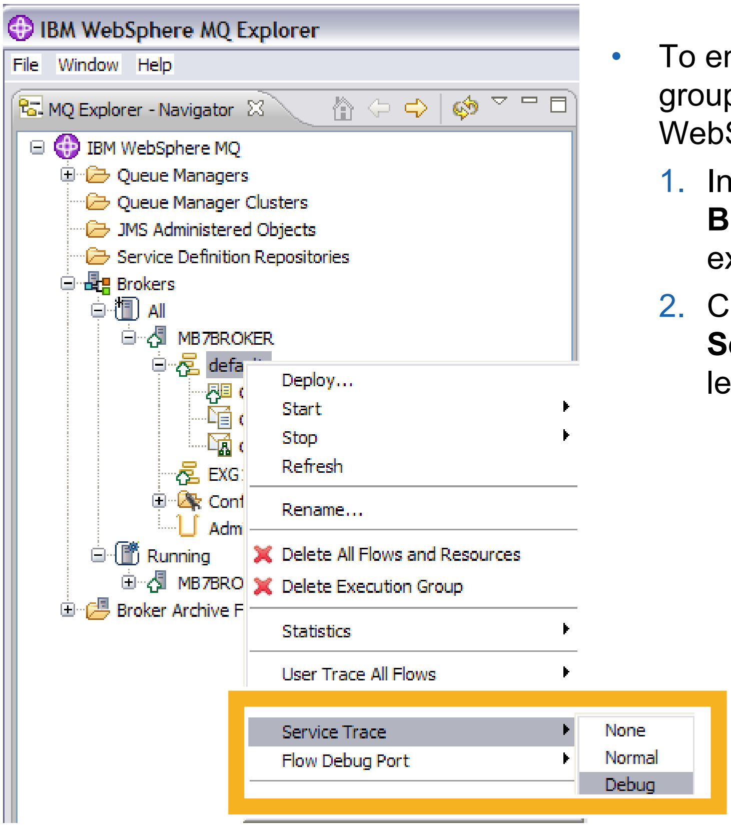
Details — Highlight the `-r` option which resets the trace file. If the trace file is not reset, new information is appended to the existing trace file.

Additional information —

Transition statement — You can also enable and disable a service trace from WebSphere Message Broker Explorer.

WebSphere Education IBM

Activating a service trace from WebSphere Message Broker Explorer



- To enable service trace for execution groups or messages flows in the WebSphere Message Broker Explorer:
  1. In the Navigator view, expand the **Brokers** folder and right-click the execution group or message flow.
  2. Click **Service Trace > Normal** or **Service Trace > Debug** to select the level of service trace

Figure 6-26. Activating a service trace from WebSphere Message Broker Explorer WM643 / VM6431.0

Notes:

You can activate a normal or debug level service from WebSphere Message Broker Explorer by selecting the execution group and selecting **Service Trace.**

Set **Service Trace** to **None** to deactivate service trace.

Instructor notes:

Purpose — Show how to enable a service trace from WebSphere Message Broker Explorer.

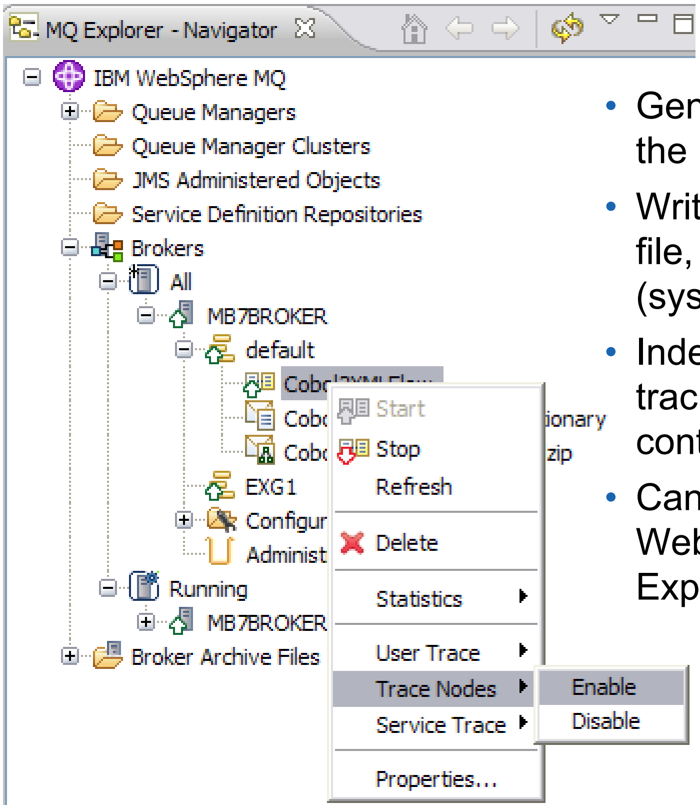
Details —

Additional information —

Transition statement — Sometimes you must see the message content when in flight between processing nodes of a message flow. This requirement can sometimes be satisfied with a trace node.

WebSphere Education

Trace node



- Generate trace records to monitor the behavior of a message flow
- Write the records to the User trace file, another file, or the local error log (syslog)
- Independent of the setting of user tracing for the message flow that contains it
- Can be enabled and disabled in WebSphere Message Broker Explorer

Figure 6-27. Trace node WM643 / VM6431.0

Notes:

You can insert a Trace node within a message flow to capture message data at a specific point in the flow. The information can be sent to the user trace, the local error log, or to a file (on the broker system).

After a Trace node is wired into a message flow, it is not necessary to remove it to disable the trace output it generates. The tracing with the Trace node can be disabled or enabled at the message flow level from the WebSphere Message Broker Explorer or using `mqsichangetrace` command line.

There is no performance penalty for inactive Trace nodes in a message flow. This means that an application developer can move a message flow that contains trace nodes into a production environment without being concerned with the overhead that tracing can cause.

Records that are written by the Trace node to the user trace log are written even if user trace is not currently active for the message flow.

Instructor notes:

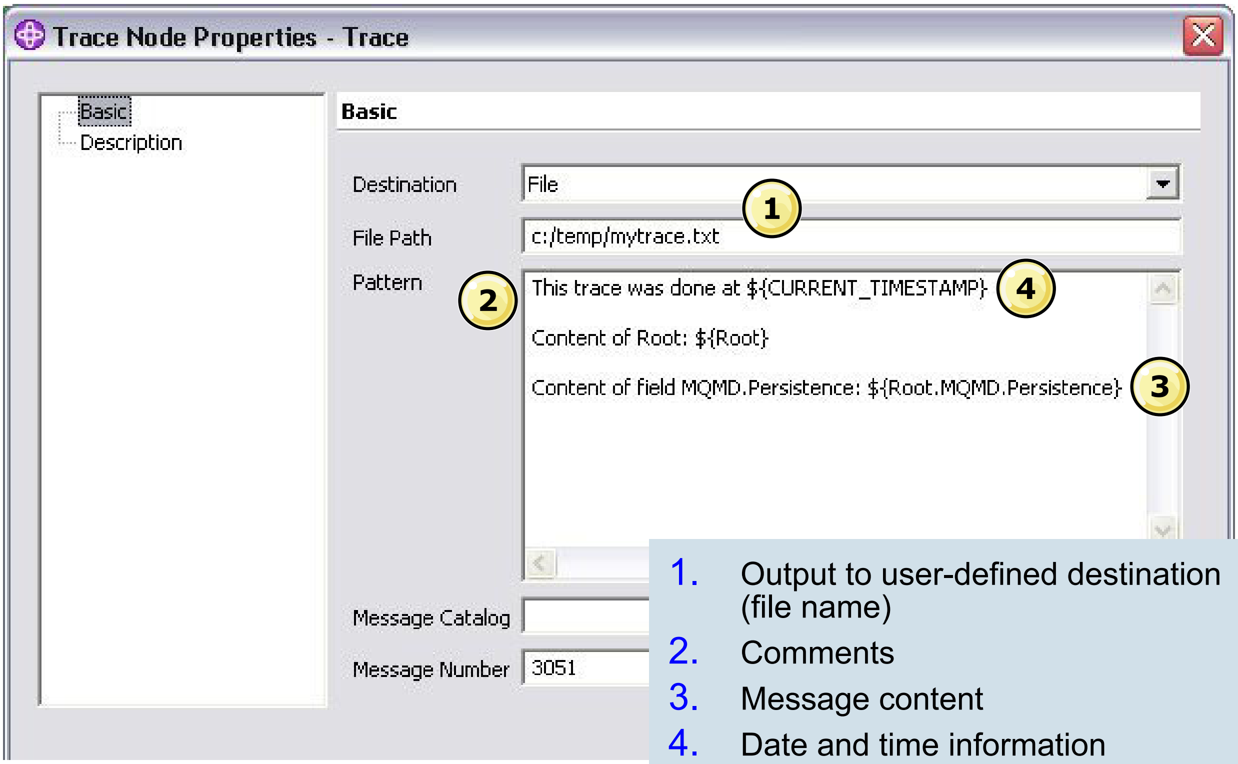
Purpose — Introduce the trace node.

Details — The trace node can be used to look at any part of the message tree that is flowing within the message flow. What is recorded depends on the patterns specified in the properties of the trace node.

Additional information — Recommend that students set the destination to **File** instead of the default of binary trace file which causes the data written to be in plain text. This makes it easier and faster to see trace output. Also, since this is usually a debug function, the students will probably want their test results isolated.

Transition statement — What is recorded in the trace file depends on the patterns specified in the properties of the Trace node.

Configuring the Trace node



Destination: File

File Path: c:/temp/mytrace.txt

Pattern:

This trace was done at ${CURRENT_TIMESTAMP}

Content of Root: ${Root}

Content of field MQMD.Persistence: ${Root.MQMD.Persistence}

Message Catalog:

Message Number: 3051

1. Output to user-defined destination (file name)
2. Comments
3. Message content
4. Date and time information

Figure 6-28. Configuring the Trace node WM643 / VM6431.0

Notes:

You must specify, in node properties, an ESQL pattern to specify what information is to be captured:

- You can insert plain text (comments) into the trace record; it is copied exactly as you enter it.
- You can identify parts of the message to be written to the trace record, specifying the full field identifiers enclosed within the characters `$`, `{`, and `}`. For example, if you want to record the entire message, specify `${Root}`. Other useful parts of the message tree that you might want to examine include `${Environment}`, `${LocalEnvironment}`, and `${ExceptionList}`.
- You can use ESQL functions to provide additional information. For example, you can use the ESQL function, `CURRENT_TIMESTAMP`, to record the current date, local time, or both, at which the message is processed.

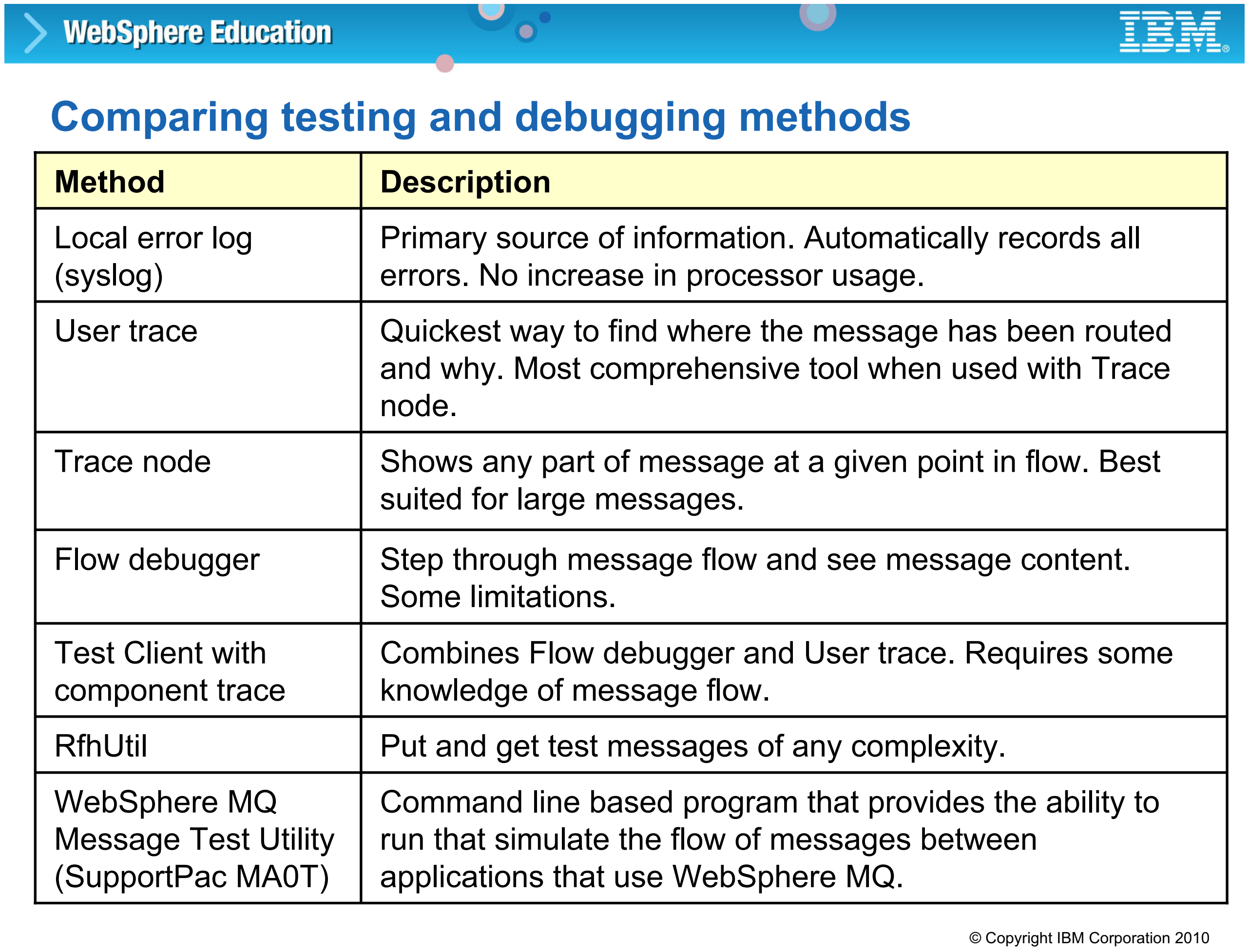
Instructor notes:

Purpose — Show an example of the Trace node properties and sample pattern.

Details — ESQL is not covered in this course.

Additional information —

Transition statement — What is the best tool to use for testing and debugging message problems?



WebSphere Education

Comparing testing and debugging methods

Method	Description
Local error log (syslog)	Primary source of information. Automatically records all errors. No increase in processor usage.
User trace	Quickest way to find where the message has been routed and why. Most comprehensive tool when used with Trace node.
Trace node	Shows any part of message at a given point in flow. Best suited for large messages.
Flow debugger	Step through message flow and see message content. Some limitations.
Test Client with component trace	Combines Flow debugger and User trace. Requires some knowledge of message flow.
RfhUtil	Put and get test messages of any complexity.
WebSphere MQ Message Test Utility (SupportPac MA0T)	Command line based program that provides the ability to run that simulate the flow of messages between applications that use WebSphere MQ.

Figure 6-29. Comparing test and debugging methods WM643 / VM6431.0

Notes:

The table summarizes testing and debugging methods. In many cases, one debugging method is not sufficient; it might be necessary to use a combination of debugging methods.

The Flow debugger is an excellent tool for identifying problems in message flows. The Flow debugger does assume an understanding of the message flow design making it, primarily, a development tool.

User trace should be the standard tool for diagnosis, combined with Trace nodes. The debugger does not always give an accurate representation of the message tree. A Trace node is always accurate, and it can optionally write its output to the user trace.

Exceptions are often longer than a single line, and they usually come in groups of two or three. User trace shows the full text, with inserts, of all the exceptions in the stack. Some parsing or modeling problems are hard to diagnose without this information.

Important

Do not use the flow debugger with component trace when messages are greater than 1 megabyte. You can have performance problems in your workspace due to excessive memory requirements.

The WebSphere MQ Message Test Utility can be used to:

- Load data from a file to a WebSphere MQ queue so that an application that reads from the queue can be tested.
- Read messages from a WebSphere MQ queue and compare the actual message with a file to see if it matches the expected result.
- Get messages from one queue and put them on another.
- Put a message with an RFH2 header on a queue to test a message flow written for the WebSphere Message Broker

See the IBM WebSphere MQ SupportPacs website at `http://www.ibm.com/software/integration/support/supportpacs` for more information about the WebSphere MQ Message Test Utility.

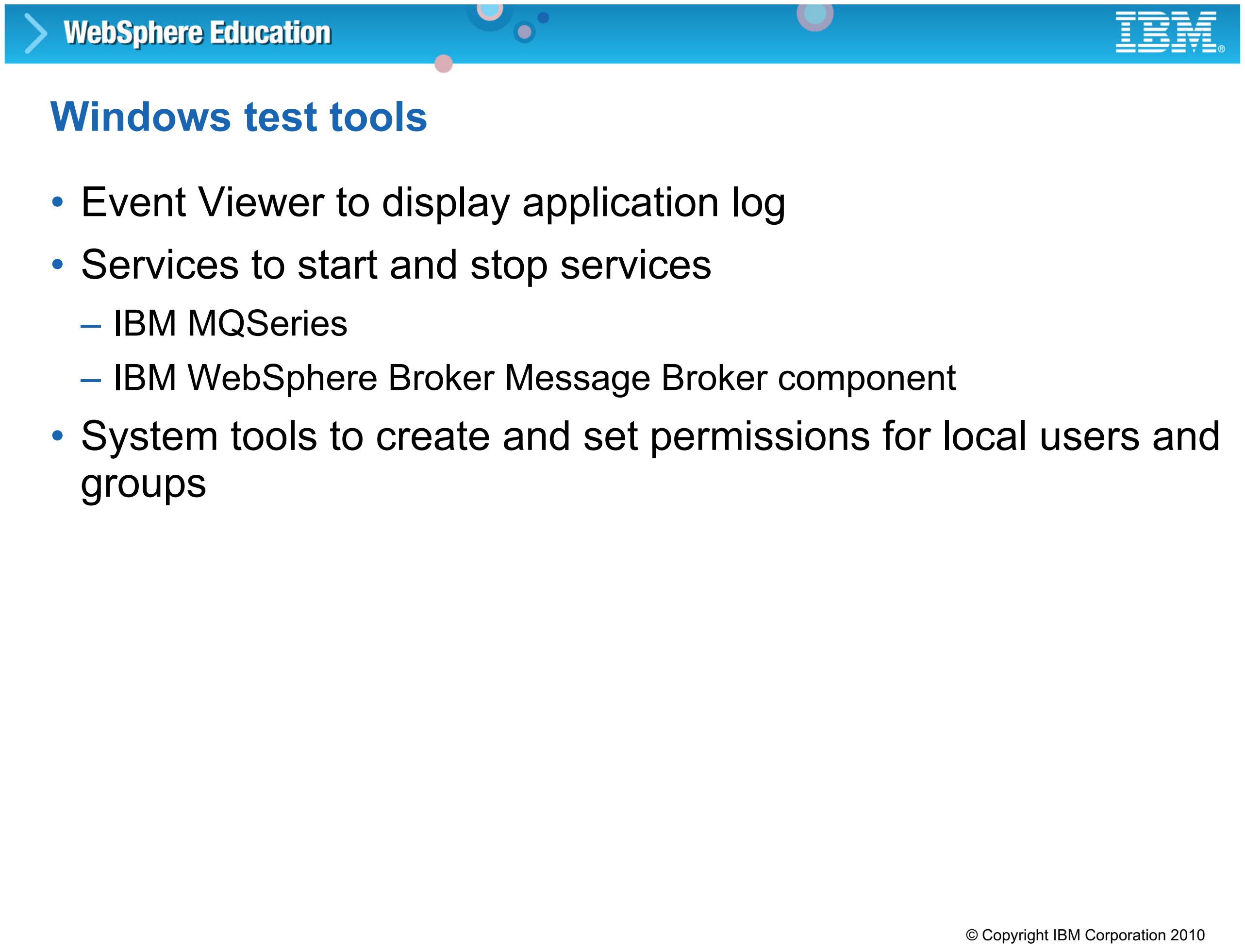
Instructor notes:

Purpose — Compare problem determination methods.

Details — None.

Additional information — None.

Transition statement — In the exercises, you can use some additional tools to put and inspect test messages.



WebSphere Education

Windows test tools

- Event Viewer to display application log
- Services to start and stop services
  - IBM MQSeries
  - IBM WebSphere Broker Message Broker component
- System tools to create and set permissions for local users and groups

Figure 6-30. Windows test tools WM643 / VM6431.0

Notes:

The lab environment for this course is based on Windows XP Professional. There are a number of built-in Windows applications that can be used to set up test environments and identify problems. The exercise instructions guide you on how to use these utilities.

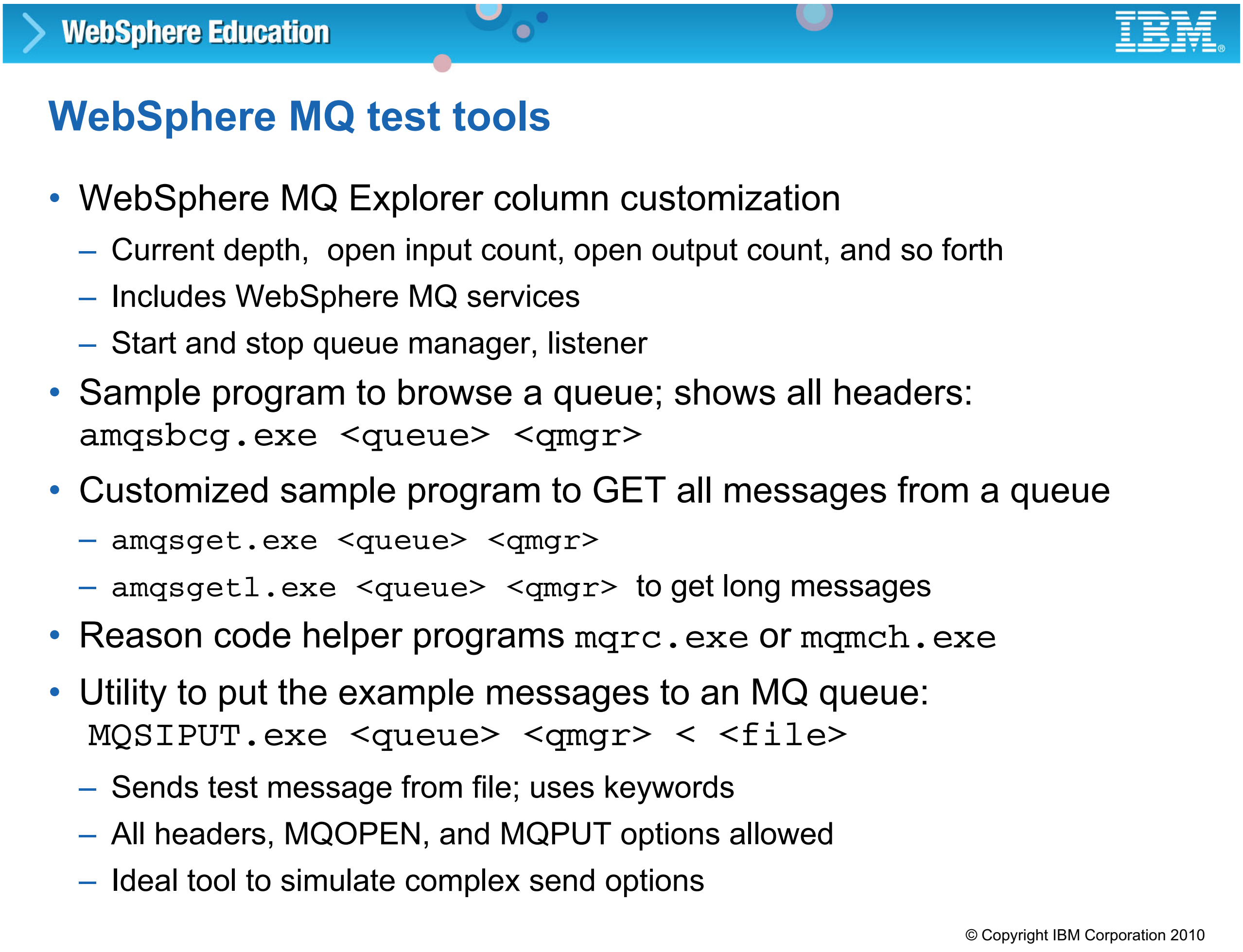
Instructor notes:

Purpose — Identify the various tools available, for the classroom and back on the job.

Details — Take time to explain each of the tools. It will save students (and you) much time during the exercises.

Additional information — Since all of these tools are available at no cost, encourage the students to do a little web surfing for additional familiarization. Also encourage students to take home whatever they think is of value from the exercise directory.

Transition statement — Here are some more tools of interest.



WebSphere Education

WebSphere MQ test tools

- WebSphere MQ Explorer column customization
  - Current depth, open input count, open output count, and so forth
  - Includes WebSphere MQ services
  - Start and stop queue manager, listener
- Sample program to browse a queue; shows all headers: `amqsbcg.exe <queue> <qmgr>`
- Customized sample program to GET all messages from a queue
  - `amqsget.exe <queue> <qmgr>`
  - `amqsgetl.exe <queue> <qmgr>` to get long messages
- Reason code helper programs `mqrc.exe` or `mqmch.exe`
- Utility to put the example messages to an MQ queue: `MQSIPUT.exe <queue> <qmgr> < <file>`
  - Sends test message from file; uses keywords
  - All headers, MQOPEN, and MQPUT options allowed
  - Ideal tool to simulate complex send options

Figure 6-31. WebSphere MQ test tools WM643 / VM6431.0

Notes:

Some of these tools are available as SupportPacs: MQSIPUT (IH02) and MQMCH (MA0K)

Look for the “SupportPacs” link on the WebSphere MQ product support website:

`www.ibm.com/software/integration/support/supportpacs`

For extensive testing before going into production with a message flow, you might also consider using the *WebSphere MQ Integrator Message Generator and Coverage Analyzer* tool, which is available as SupportPac IH04

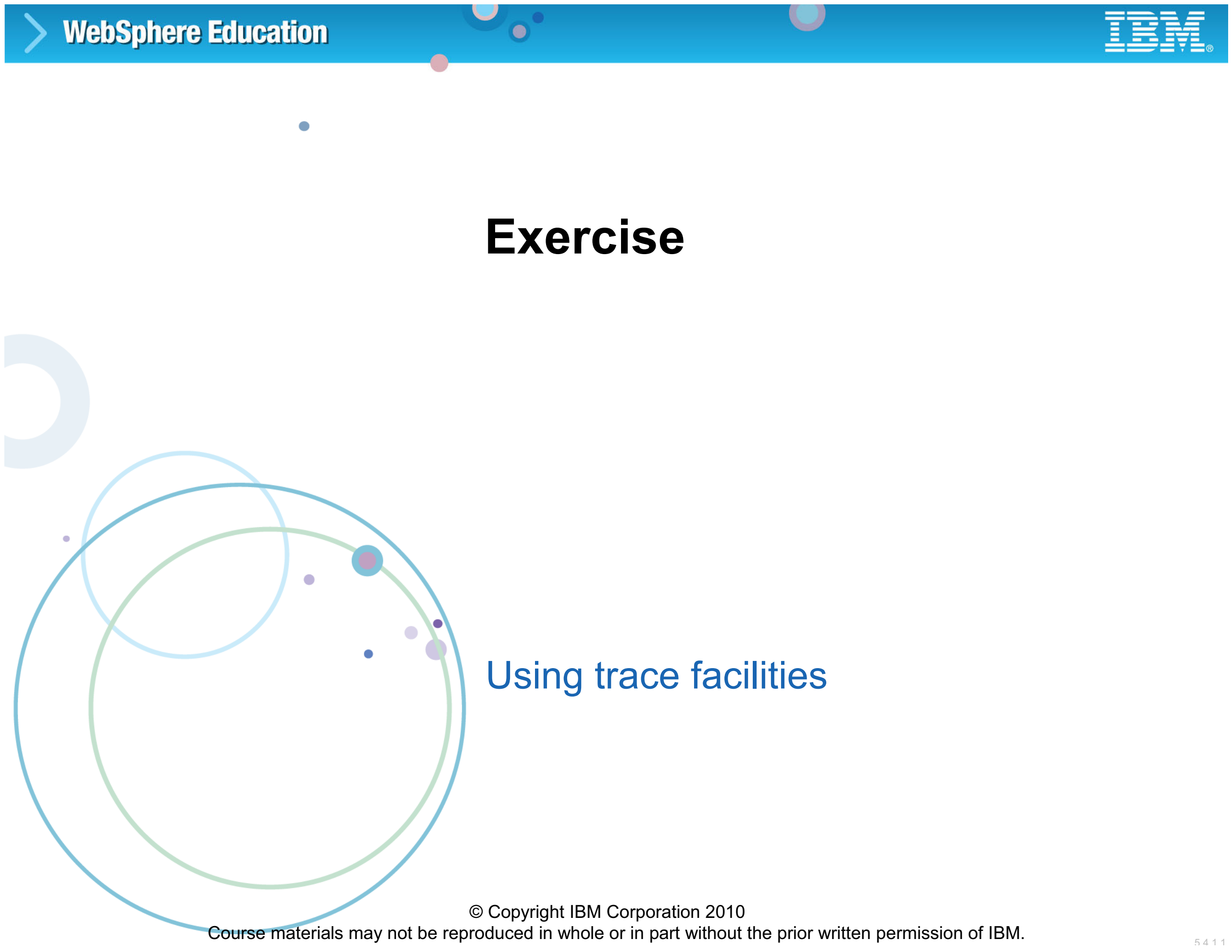
Instructor notes:

Purpose — Close this topic with a summary of available test tools.

Details — None.

Additional information — None.

Transition statement — Next: Exercise 5



WebSphere Education

Exercise

Using trace facilities

5411

Figure 6-32. Exercise 5 WM643 / VM6431.0

Notes:

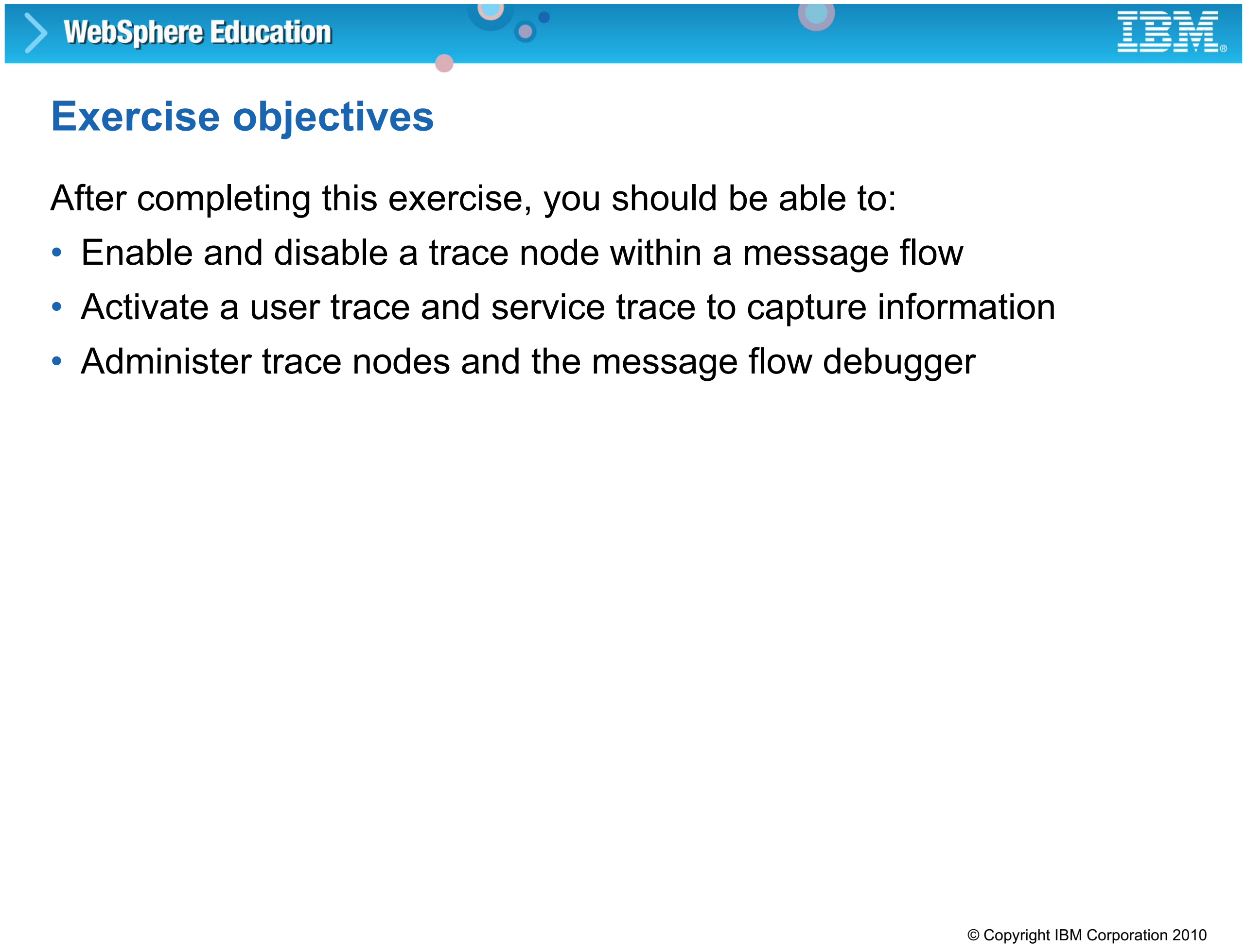
Instructor notes:

Purpose — This is a placeholder slide for exercise 5.

Details — None.

Additional information — None.

Transition statement — Next: Exercise objectives



WebSphere Education

Exercise objectives

After completing this exercise, you should be able to:

- Enable and disable a trace node within a message flow
- Activate a user trace and service trace to capture information
- Administer trace nodes and the message flow debugger

Figure 6-33. Exercise objectives

WM643 / VM6431.0

Notes:

In this exercise, you will use the various tracing and debugging tools presented in this topic to identify problems in message flows.

Proceed to the *Student Exercise Guide* and complete Exercise 5.

Instructor notes:

Purpose — Introduce the lab exercise objectives.

Details — Have students complete Exercise 5.

Additional information —

Transition statement — The administrator is responsible for more than just the operation of message flows. The next topic identifies some administrator-specific problems and provides some guidance in helping to isolate and solve the problem.

6.3. Troubleshooting common problems

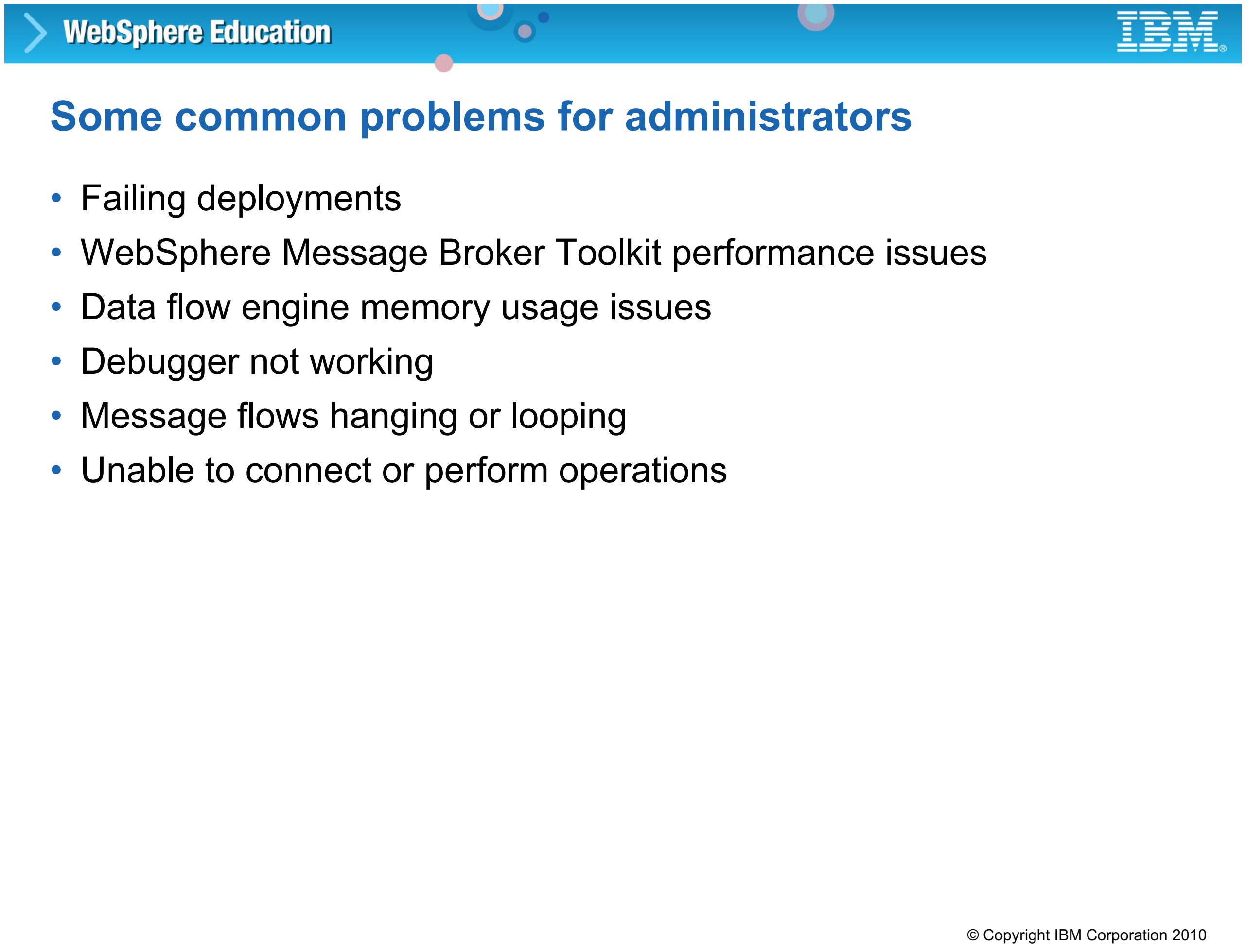
Instructor topic introduction

What students will do — Students will learn about some of the common administrator problems encountered and how to troubleshoot them.

How students will do it — Students will listen to a brief lecture about the available troubleshooting and will use some of the methods mentioned in subsequent exercises.

What students will learn — Students will be able to consider some possible causes of problems and to decide on an approach to solving them.

How this will help students in their job — Students will be prepared to consider how to approach some common problems to reach a faster resolution.



WebSphere Education

Some common problems for administrators

- Failing deployments
- WebSphere Message Broker Toolkit performance issues
- Data flow engine memory usage issues
- Debugger not working
- Message flows hanging or looping
- Unable to connect or perform operations

Figure 6-34. Some common problems for administrators WM643 / VM6431.0

Notes:

One of the best resources for identifying common problems is the IBM Support Center. The following items have been excerpted from information the Support Center has put together to aid in identifying some of the most common problems and to decide how to approach resolving them.

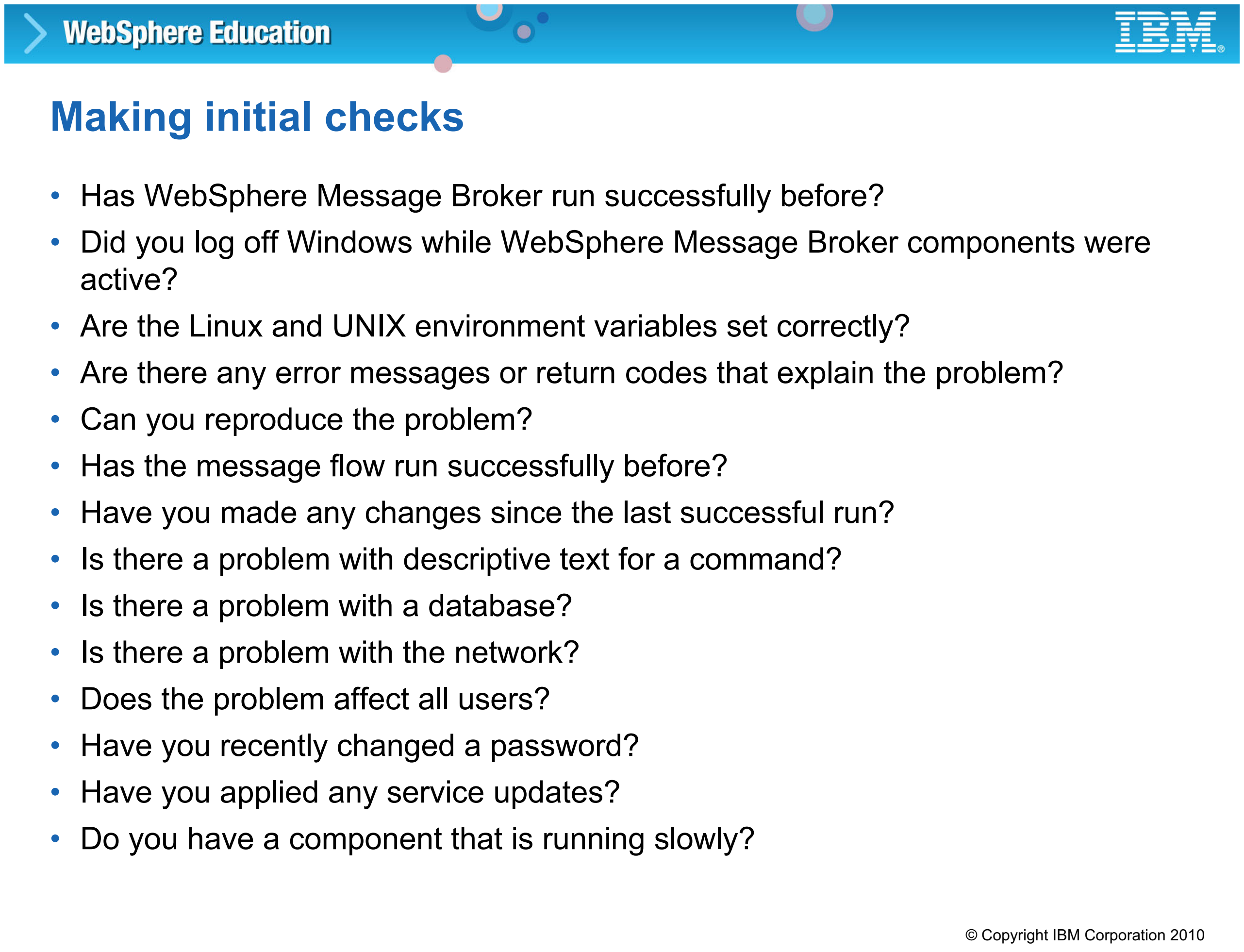
Instructor notes:

Purpose — List some of the most commonly encountered problems for an administrator.

Details — This information comes from a “top 10” list provided by the IBM Support Center. It is not exhaustive; rather, it is intended to provide some items that can be considered if certain types of problems are encountered.

Additional information — The information provided by the support center listed several APARs (authorized program analysis reports). However, since this information is meant to be a bit more generic than recommending a specific APAR, that information has not been included here.

Transition statement — There are some checks you can make immediately to help you to isolate a problem.



WebSphere Education
IBM

Making initial checks

- Has WebSphere Message Broker run successfully before?
- Did you log off Windows while WebSphere Message Broker components were active?
- Are the Linux and UNIX environment variables set correctly?
- Are there any error messages or return codes that explain the problem?
- Can you reproduce the problem?
- Has the message flow run successfully before?
- Have you made any changes since the last successful run?
- Is there a problem with descriptive text for a command?
- Is there a problem with a database?
- Is there a problem with the network?
- Does the problem affect all users?
- Have you recently changed a password?
- Have you applied any service updates?
- Do you have a component that is running slowly?

Figure 6-35. Making initial checks WM643 / VM6431.0

Notes:

Before you start problem determination in detail, consider whether there is an obvious cause of the problem, or an area of investigation that is likely to give useful results. This approach to diagnosis can often save much work by highlighting a simple error, or by narrowing down the range of possibilities.

The figure lists a number of questions you should ask when starting the problem determination process.

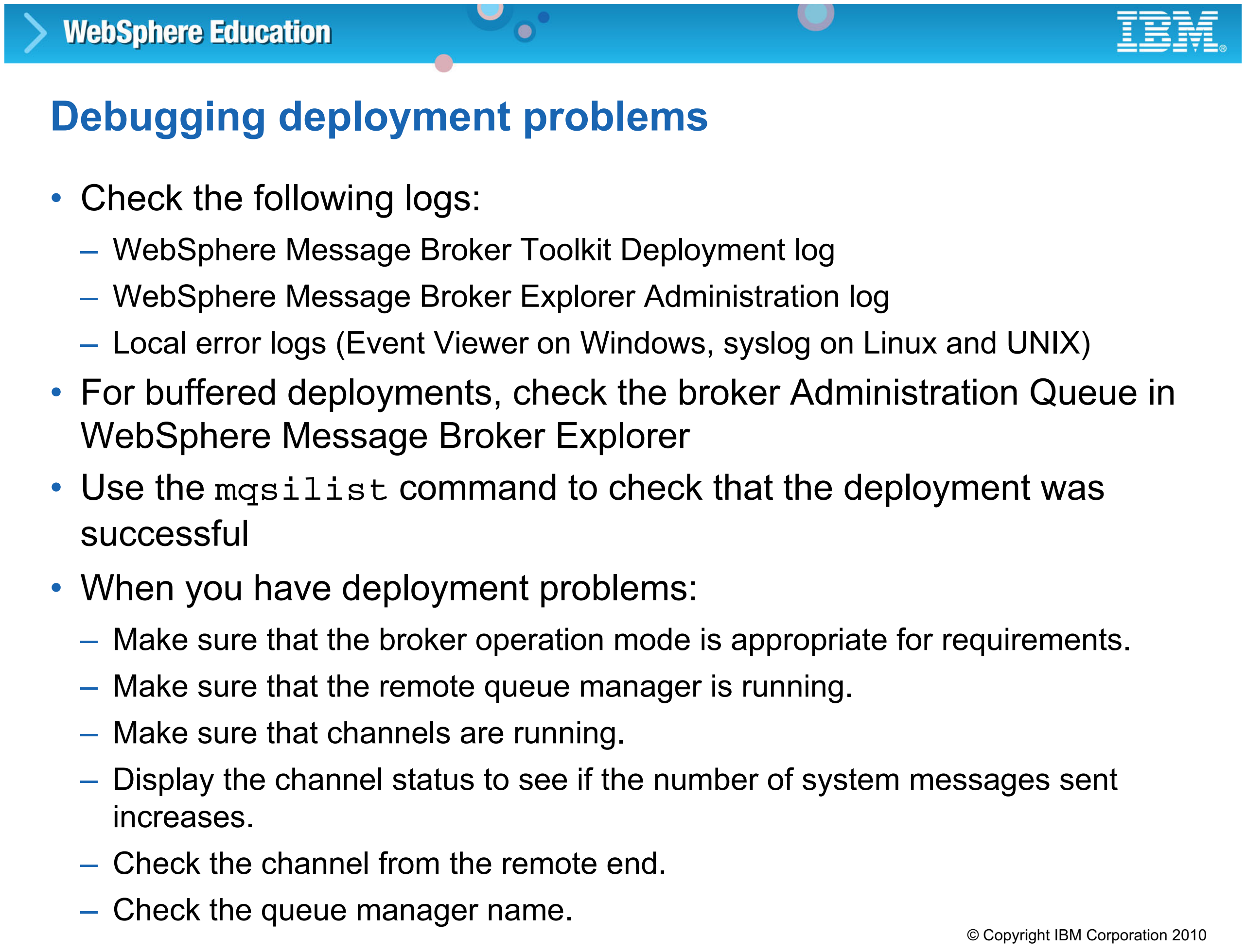
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement — It is not uncommon to have problems occur during deployment.



WebSphere Education

Debugging deployment problems

- Check the following logs:
  - WebSphere Message Broker Toolkit Deployment log
  - WebSphere Message Broker Explorer Administration log
  - Local error logs (Event Viewer on Windows, syslog on Linux and UNIX)
- For buffered deployments, check the broker Administration Queue in WebSphere Message Broker Explorer
- Use the `mqsilist` command to check that the deployment was successful
- When you have deployment problems:
  - Make sure that the broker operation mode is appropriate for requirements.
  - Make sure that the remote queue manager is running.
  - Make sure that channels are running.
  - Display the channel status to see if the number of system messages sent increases.
  - Check the channel from the remote end.
  - Check the queue manager name.

Figure 6-36. Debugging deployment problems WM643 / VM6431.0

Notes:

Some possible deployment errors are:

- No response
- Negative response
- Error on deployment

Sometimes it seems as if a deployment has not been started. It is not a good idea to resubmit or try the deployment again. This generally causes a series of deployments to be stacked up. If a deployment is not responding, check the local error logs and the Deployment log. Look for WebSphere MQ errors where WebSphere MQ was unable to send the deployment messages.

Negative responses can be received if the user does not have the correct permissions. Check that the broker service user ID and the user ID are authorized on the broker’s system. It is also possible that the configuration timeout and ConfigurationDelayTimeout values are not adequate. These can be set using `mqsichangebroker`.

Smaller deploys work better; avoid using complete deploys and also deploying large groups of message flows and message sets in one deployment.

You can view the progress of a deployment by setting the MQSI_SHOW_DEPLOY environment variable. If a deployment fails, check for operating system and WebSphere MQ errors in the appropriate logs. To set the environment variable (the example is for a UNIX system):

```
export MQSI_SHOW_DEPLOY=1
```

The broker logs BIP8099I messages to the event viewer or syslog at every logical step of the deployment operation. This helps to isolate the problems. The broker must be restarted to recognize the environment variable.

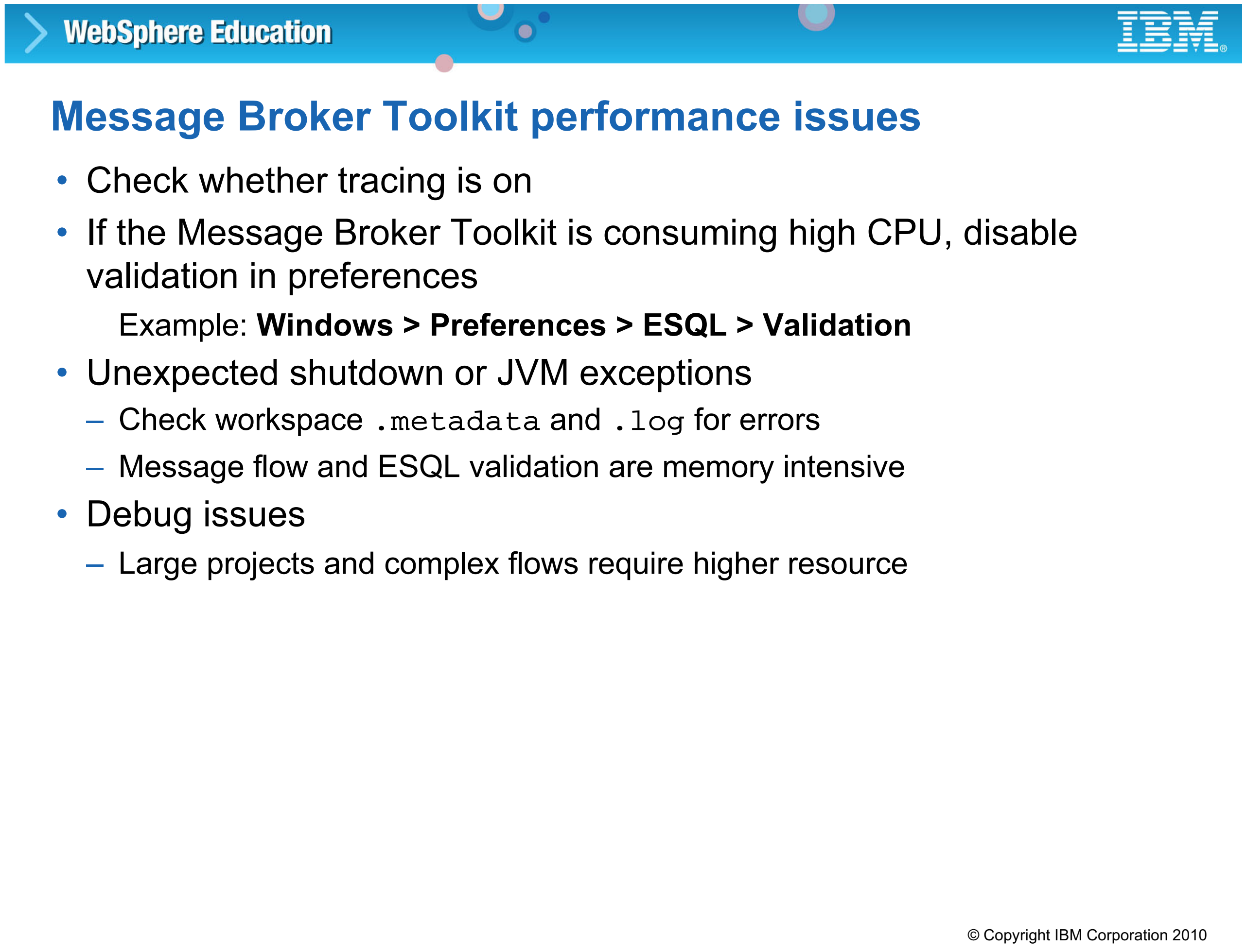
Instructor notes:

Purpose — Offer an overview of possible deployment issues.

Details — This is not a thorough discussion. Time does not permit. However, these hints should offer some simple things to consider if a deployment is not responding or is failing.

Additional information — The Administration queue is covered in Unit 2. The Administration log is covered in Unit 4. Students should be familiar with the Administration log as they used it in a previous exercise.

Transition statement — Sometimes, the toolkit can seem to be having performance problems.



WebSphere Education

Message Broker Toolkit performance issues

- Check whether tracing is on
- If the Message Broker Toolkit is consuming high CPU, disable validation in preferences

  Example: **Windows > Preferences > ESQL > Validation**
- Unexpected shutdown or JVM exceptions
  - Check workspace `.metadata` and `.log` for errors
  - Message flow and ESQL validation are memory intensive
- Debug issues
  - Large projects and complex flows require higher resource

Figure 6-37. Message Broker Toolkit performance issues WM643 / VM6431.0

Notes:

If the WebSphere Message Broker Toolkit seems to be slow when saving a project or when creating a broker archive, make sure that validation is turned off. Validation of ESQL is very memory-intensive.

Large projects, debugging complex flows, and testing large messages in the WebSphere Message Broker Toolkit are resource-intensive; you might need to increase the size of the heap size used by the WebSphere Message Broker Toolkit. If the WebSphere Message Broker Toolkit shuts down, it is possible it is related to a JVM exception; be sure to check the messages in the local error log.

The WebSphere Message Broker Toolkit (`mb.exe`) creates the JVM with a default maximum heap size. The JVM default maximum heap size is half the real memory on the server. Increase the WebSphere Message Broker Toolkit JVM maximum heap size by starting the WebSphere Message Broker Toolkit from the command line. For example:

> `<ToolkitInstDir>\mb.exe -data <path>WorkspaceDir -vmargs`
> `-Xmx1024M` starts the toolkit with JVM storage of 1 GB.

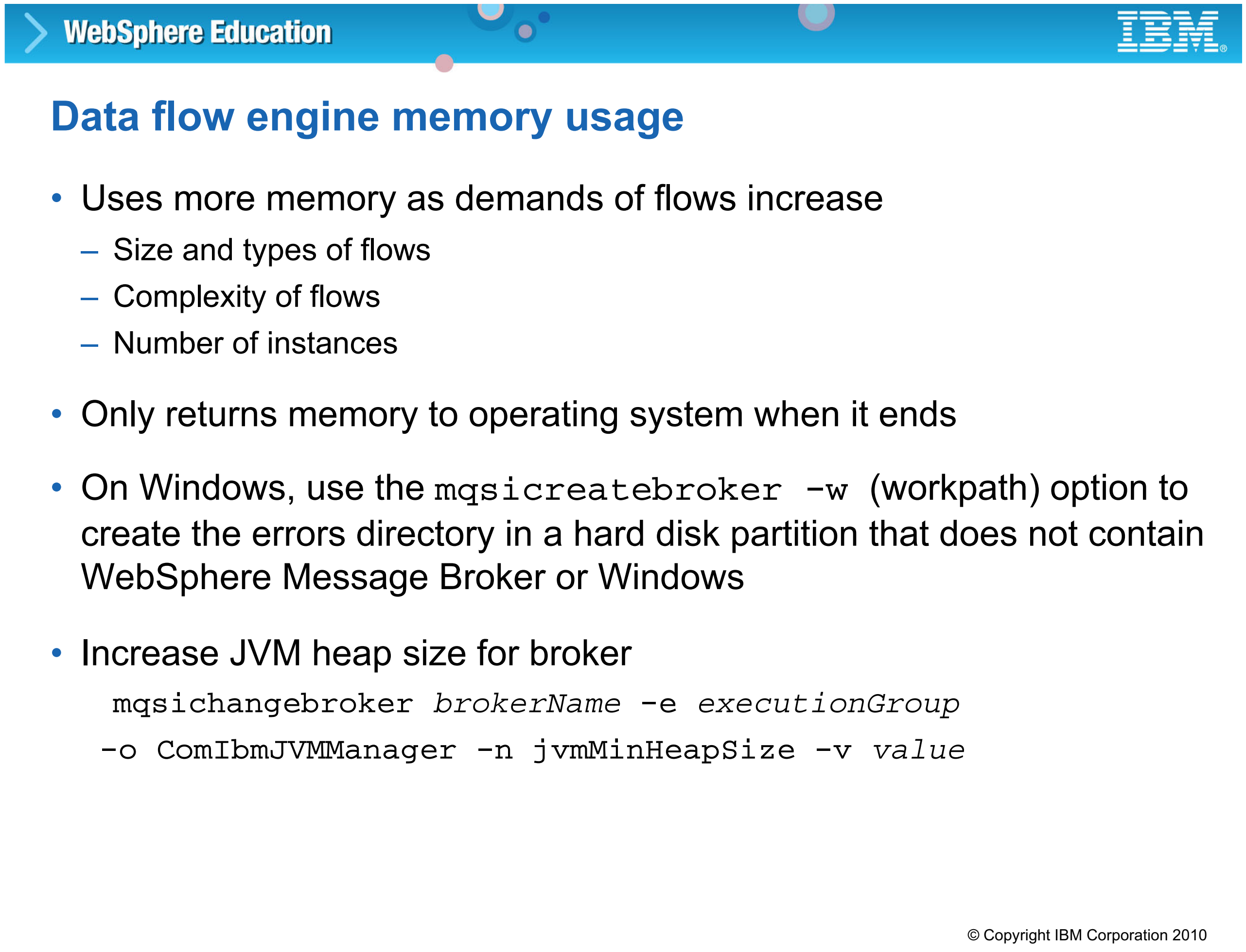
Instructor notes:

Purpose — Offer some hints about how to speed up the WebSphere Message Broker Toolkit response.

Details — It is important to carefully review the documentation before making wholesale changes. In addition, be sure that students are aware of SupportPacs that are available with advice on performance and tuning, such as IP04, WebSphere Business Integration, and Designing for Performance.

Additional information — None.

Transition statement — Sometimes the execution groups, known as data flow engines, have issues with performance.



WebSphere Education

Data flow engine memory usage

- Uses more memory as demands of flows increase
  - Size and types of flows
  - Complexity of flows
  - Number of instances
- Only returns memory to operating system when it ends
- On Windows, use the `mqsicreatebroker -w` (workpath) option to create the errors directory in a hard disk partition that does not contain WebSphere Message Broker or Windows
- Increase JVM heap size for broker

```
  mqsichangebroker brokerName -e executionGroup
 -o ComIbmJVMManager -n jvmMinHeapSize -v value
```

Figure 6-38. Data flow engine memory usage

WM643 / VM6431.0

Notes:

Each execution group starts one JVM and then uses it for all Java code running in all flows on that JVM. Each time a new message flow is added to an execution group, memory consumption is increased in that execution group. These data flow engines (execution group) have a finite limit on the number of threads each can support (255). As those threads are created, more memory is required.

If a particular flow is processing-intensive and must process larger volumes of messages, it might be better to deploy the flow to multiple execution groups, rather than increasing the number of instances within an execution group.

- A message flow issues a “free” on all its storage after completing message processing, but the “free” does not release memory to the operating system.
- Memory is returned to the operating system only upon termination of the application.
- The operating system does not retrieve storage from a process until termination; memory gets released on a process restart.

The data flow engine can be restarted with a command:

```
mqsireload <Broker> -e <Execution Group>
```

Here are some things to remember:

- The data flow engine increases in memory usage depending on the number of flows deployed, types of messages, and message sizes processed. Memory increases when processing large messages.
- Message flows run as threads in a process and request memory for message processing.
- The data flow engine is a multithreaded process where many threads can request storage at the same time, causing memory requests to the operating system.

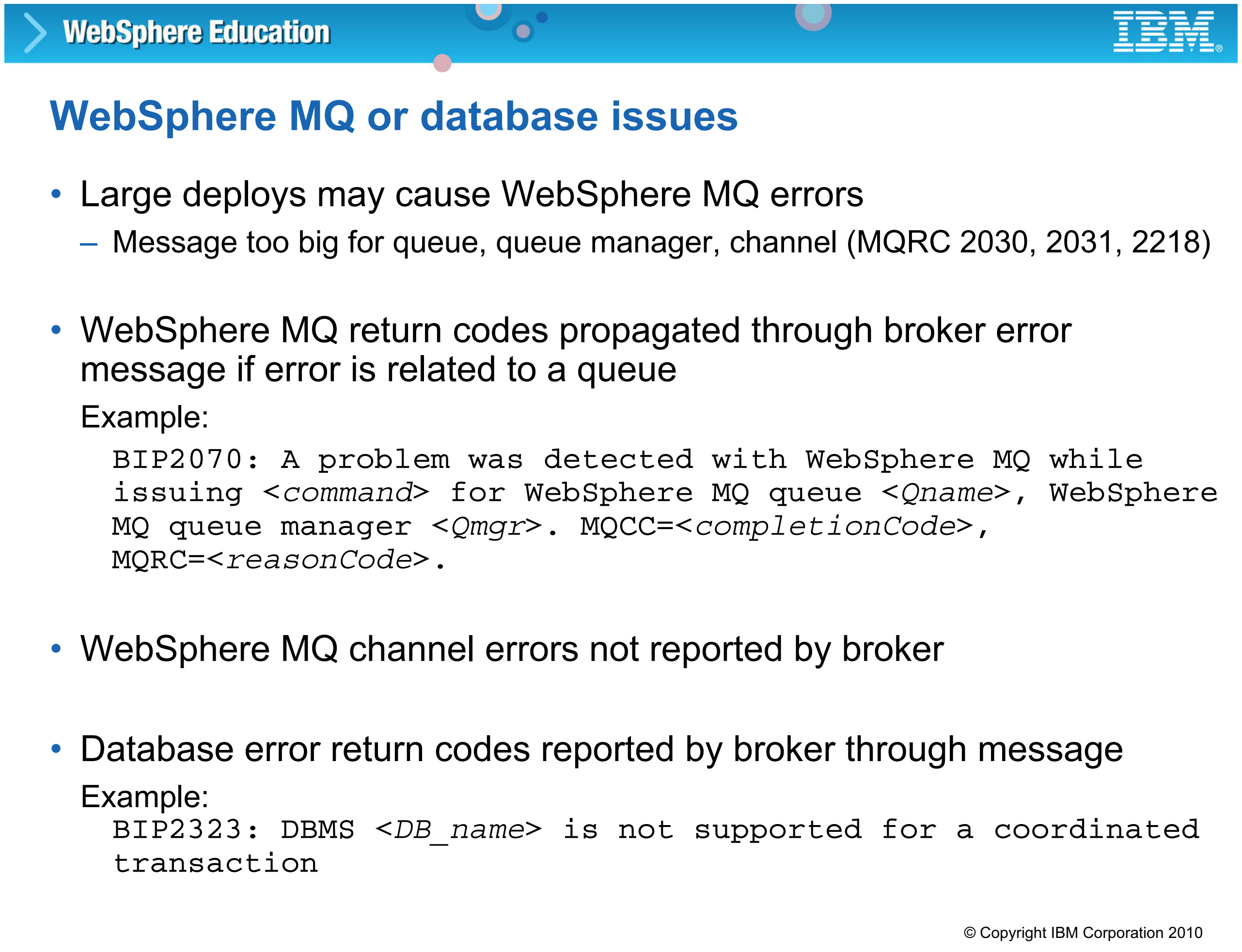
Instructor notes:

Purpose — Mention some possible issues with the memory use of an execution group.

Details — None.

Additional information — The current broker behavior might change in future versions of WebSphere Message Broker.

Transition statement — It is possible that database or WebSphere MQ problems can cause the broker environment to have problems.



WebSphere Education

WebSphere MQ or database issues

- Large deploys may cause WebSphere MQ errors
  - Message too big for queue, queue manager, channel (MQRC 2030, 2031, 2218)

- WebSphere MQ return codes propagated through broker error message if error is related to a queue

  Example:

```
BIP2070: A problem was detected with WebSphere MQ while
issuing <command> for WebSphere MQ queue <Qname>, WebSphere
MQ queue manager <Qmgr>. MQCC=<completionCode>,
MQRC=<reasonCode>.
```

- WebSphere MQ channel errors not reported by broker

- Database error return codes reported by broker through message

  Example:

```
BIP2323: DBMS <DB_name> is not supported for a coordinated
transaction
```

Figure 6-39. WebSphere MQ or database issues WM643 / VM6431.0

Notes:

Several WebSphere MQ errors can be reported when working with a broker. One to be aware of is MQRC_OBJECT_IN_USE (MQRC 2042). This can occur when altering a queue used in an MQInput node after stopping a message flow. Although the flow stops, the queue is not closed. It might be necessary to remove the message flow from the execution group so it no longer is in the runtime environment, change the queue attributes, and then redeploy the message flow.

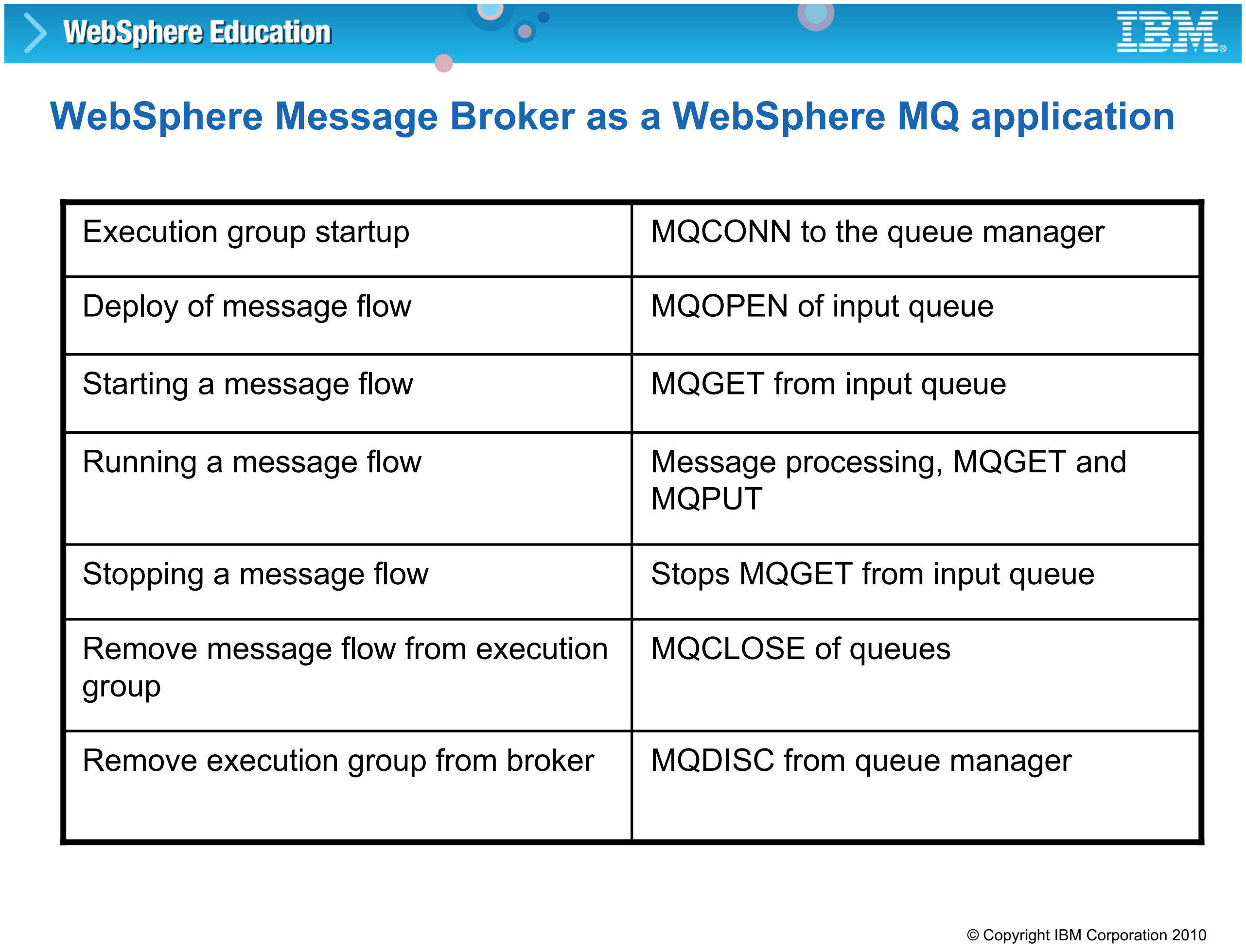
Instructor notes:

Purpose — Remind students that some WebSphere MQ and database errors can also cause problems for the broker.

Details — None.

Additional information — None.

Transition statement — The message broker is, after all, an WebSphere MQ application. So, consider it from that aspect.



WebSphere Education

WebSphere Message Broker as a WebSphere MQ application

Execution group startup	MQCONN to the queue manager
Deploy of message flow	MQOPEN of input queue
Starting a message flow	MQGET from input queue
Running a message flow	Message processing, MQGET and MQPUT
Stopping a message flow	Stops MQGET from input queue
Remove message flow from execution group	MQCLOSE of queues
Remove execution group from broker	MQDISC from queue manager

Figure 6-40. WebSphere Message Broker as an WebSphere MQ application WM643 / VM6431.0

Notes:

The table relates execution group and message flow operations to WebSphere MQ functions.

If you are not familiar with WebSphere MQ API calls, do not let this chart cause you concern. If you are concerned, it might help you as you reflect on the broker and how it works. This is a good place to point out that the MQGETs are no longer issued if a message flow is stopped but the queue is not closed.

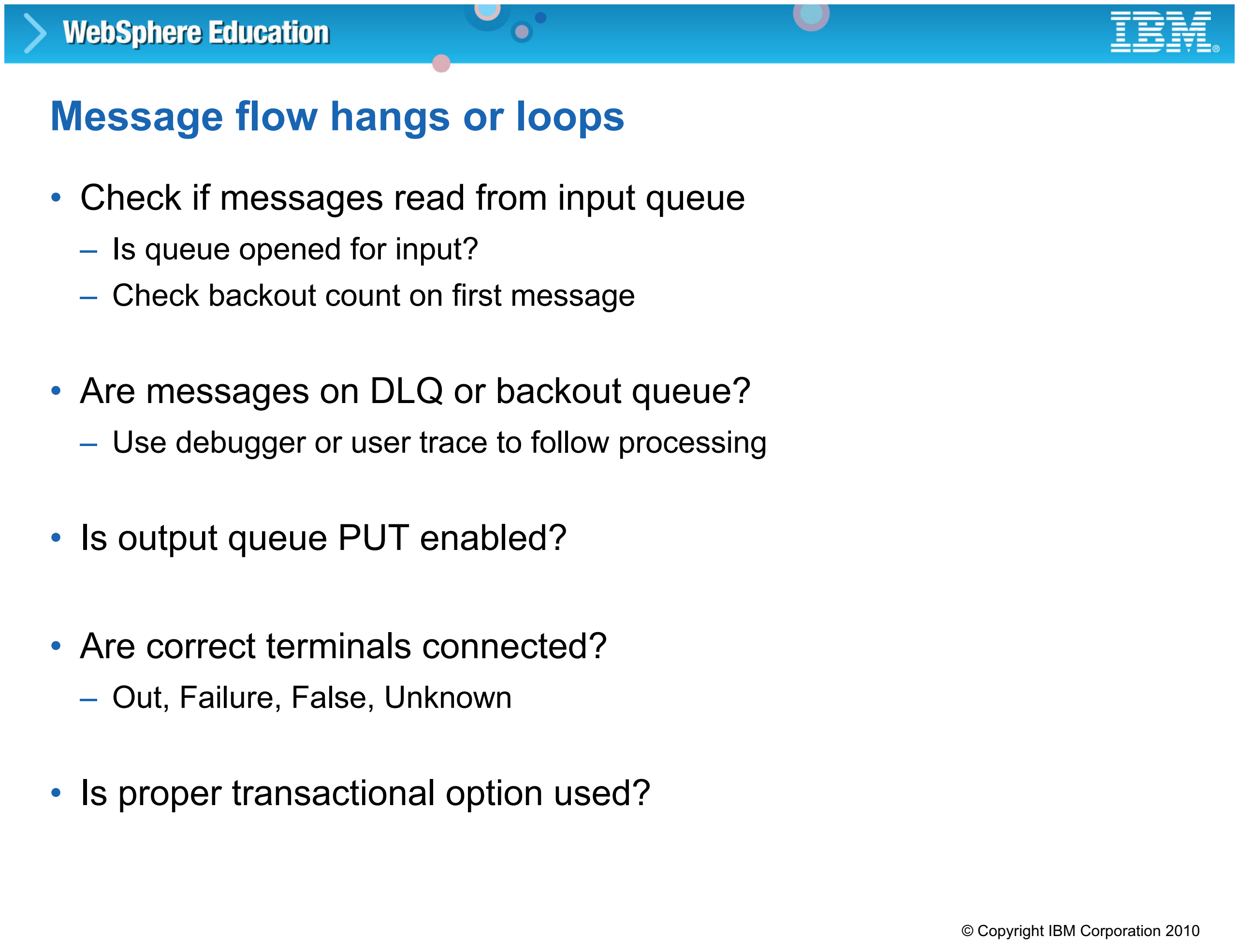
Instructor notes:

Purpose — Relate the stages in an execution group and message flow operation to WebSphere MQ functions.

Details — None.

Additional information — None.

Transition statement — The debugger is useful; however, there might be problems using it if you are trying to debug message flow loops.



WebSphere Education

Message flow hangs or loops

- Check if messages read from input queue
  - Is queue opened for input?
  - Check backout count on first message
- Are messages on DLQ or backout queue?
  - Use debugger or user trace to follow processing
- Is output queue PUT enabled?
- Are correct terminals connected?
  - Out, Failure, False, Unknown
- Is proper transactional option used?

Figure 6-41. Message flow hangs or loops WM643 / VM6431.0

Notes:

After deploying a new flow, always check to see if the input queue is opened for input. In some cases, the problem can be that the flow is not set up to use the specified queue. The problem can also be that the deployment failed; so always check the results of a deployment in the WebSphere Message Broker Explorer Deployment log. If the messages appear on the input queue, look at the backout count of the first message. If it is more than 0, there is likely a problem where the message has no place to go; check the output queues to make sure PUT is enabled.

If messages have disappeared from the input queue, check the backout requeue queue or dead-letter queue if no explicit failure processing is in the flow. It is possible that a flow is in some type of loop, perhaps where there is a propagation in a Compute node but the loop’s conditions are set up incorrectly; you will need assistance from the developer to correct this. It can be detected using a user trace.

Messages can also be discarded because a flow has not been properly set up. If a message is transactional, it is not thrown away; however, it is possible that incorrect (or missing) terminal connections can cause a flow to end at a point that was not intended.

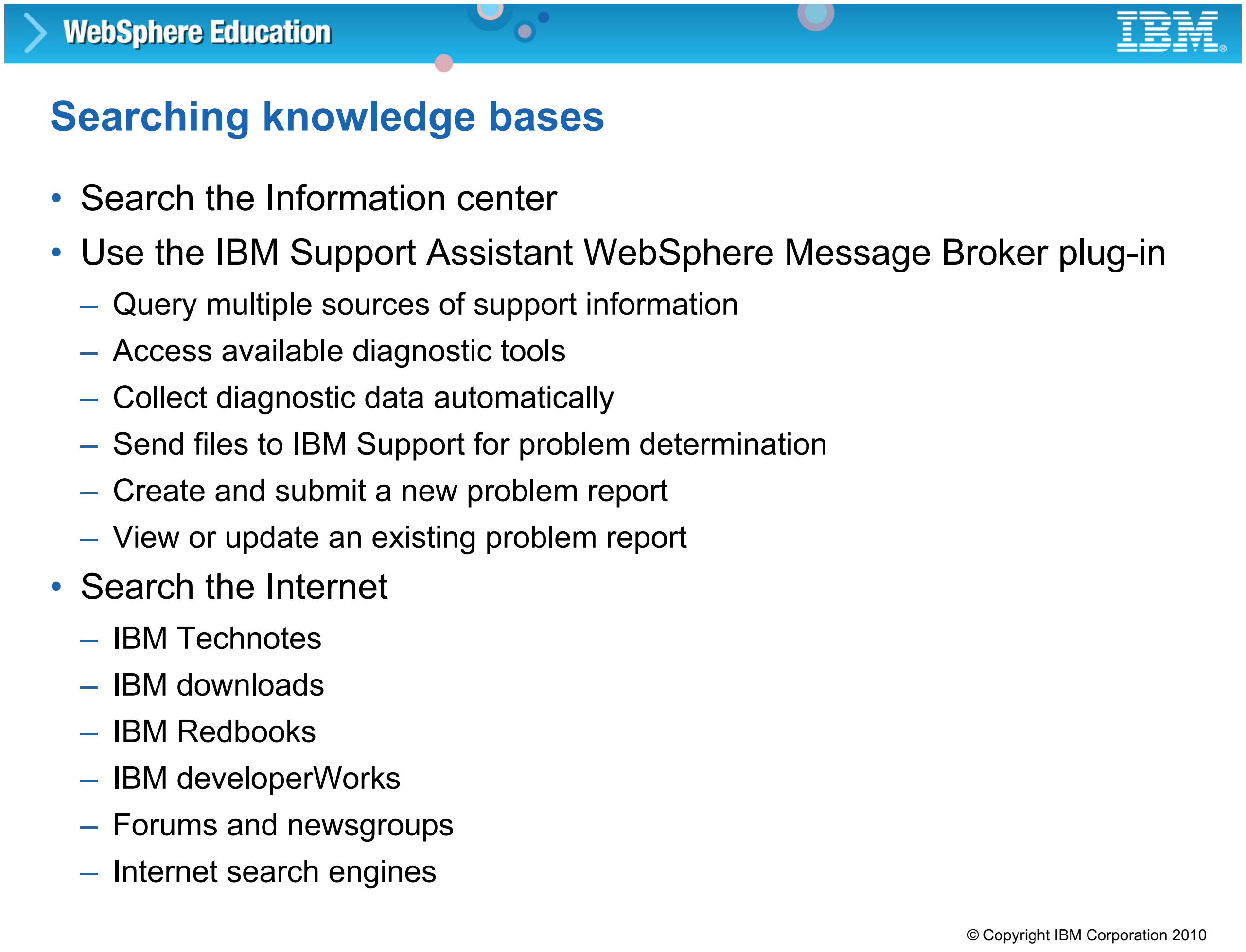
Instructor notes:

Purpose — Review and remind students of some potential pitfalls in message flows.

Details — Some of this is a review but since this is one of the most common areas of issues with problem determination (where is your message), it is worth visiting again.

Additional information — None.

Transition statement — There are many knowledge bases available for helping you to troubleshoot problems.



WebSphere Education

Searching knowledge bases

- Search the Information center
- Use the IBM Support Assistant WebSphere Message Broker plug-in
  - Query multiple sources of support information
  - Access available diagnostic tools
  - Collect diagnostic data automatically
  - Send files to IBM Support for problem determination
  - Create and submit a new problem report
  - View or update an existing problem report
- Search the Internet
  - IBM Technotes
  - IBM downloads
  - IBM Redbooks
  - IBM developerWorks
  - Forums and newsgroups
  - Internet search engines

Figure 6-42. Searching knowledge bases WM643 / VM6431.0

Notes:

If you have a problem with WebSphere Message Broker, you want it resolved quickly. Begin by searching the available knowledge bases to determine whether the resolution to your problem is already documented.

IBM provides extensive documentation in the form of online information centers, forums, and Redbooks.

IBM Support Assistant (ISA) is available at no charge to install on your computer; you then install the relevant product add-on. ISA has a built-in user guide, and the ISA download package includes a quick start installation and configuration guide.

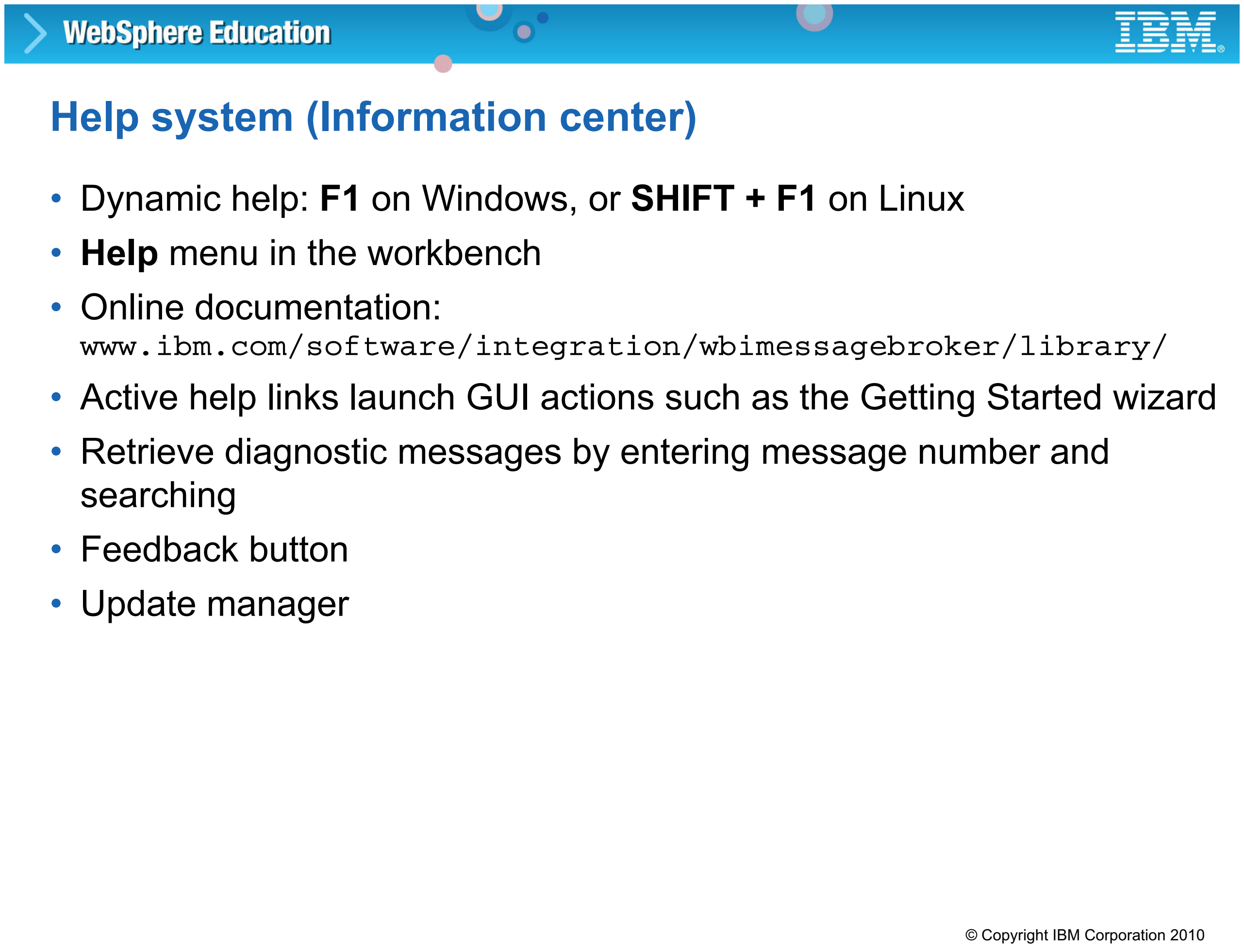
Instructor notes:

Purpose — Introduce sources of information for administrators.

Details — An important aspect of this course is teaching administrators where to go to get help. Many times the problem they are experiencing is a problem that has a documented solution.

Additional information — None.

Transition statement — The first place to start is the WebSphere Message Broker Information Center.



WebSphere Education
IBM

Help system (Information center)

- Dynamic help: **F1** on Windows, or **SHIFT + F1** on Linux
- **Help** menu in the workbench
- Online documentation:
  `www.ibm.com/software/integration/wbimessagebroker/library/`
- Active help links launch GUI actions such as the Getting Started wizard
- Retrieve diagnostic messages by entering message number and searching
- Feedback button
- Update manager

Figure 6-43. Help system (information center) WM643 / VM6431.0

Notes:

An information center can be installed on your local workstation or on a local intranet. An information center can also be viewed on the IBM website. You can use the powerful search function of the information center to query conceptual and reference information and detailed instructions for completing tasks.

You can also start the online WebSphere Message Broker information center:
`http://publib.boulder.ibm.com/infocenter/wmbhelp/v7r0m0/index.jsp`

You can download manuals in PDF format from
`ftp://ftp.software.ibm.com/software/integration/wbibrokers/docs/V7.0/`

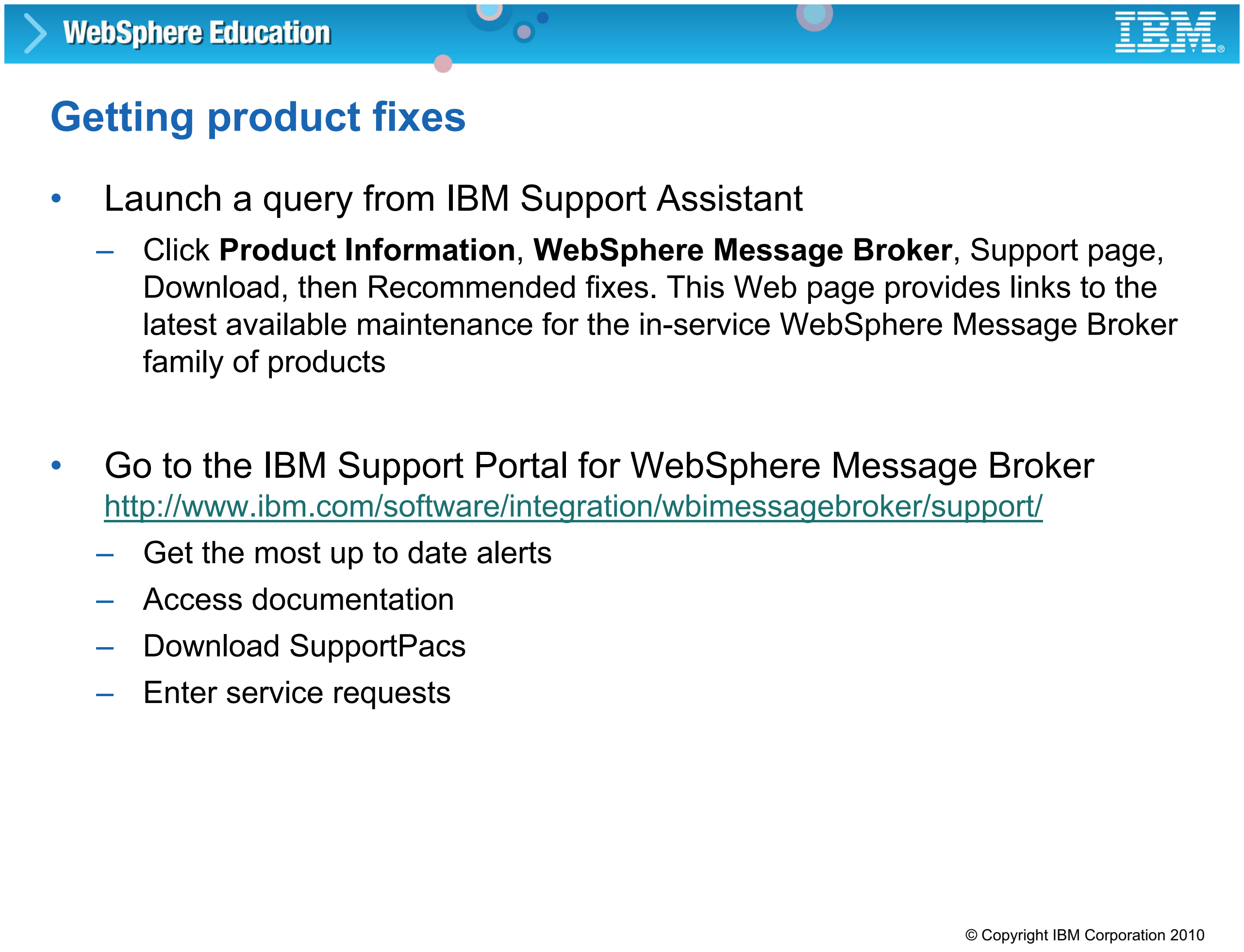
Instructor notes:

Purpose — Review the WebSphere Message Broker Information Center.

Details — The Information center was introduced in an earlier unit but it is important to remind students that the Information center is the first place to look for information.

Additional information —

Transition statement — In some cases, a product fix might be required to solve your problem.



WebSphere Education

Getting product fixes

- Launch a query from IBM Support Assistant
  - Click **Product Information**, **WebSphere Message Broker**, Support page, Download, then Recommended fixes. This Web page provides links to the latest available maintenance for the in-service WebSphere Message Broker family of products
- Go to the IBM Support Portal for WebSphere Message Broker http://www.ibm.com/software/integration/wbimessagebroker/support/
  - Get the most up to date alerts
  - Access documentation
  - Download SupportPacs
  - Enter service requests

Figure 6-44. Getting product fixes WM643 / VM6431.0

Notes:

A product fix might be available to resolve your problem. You can determine what fixes are available by launching a query from the IBM Support Assistant.

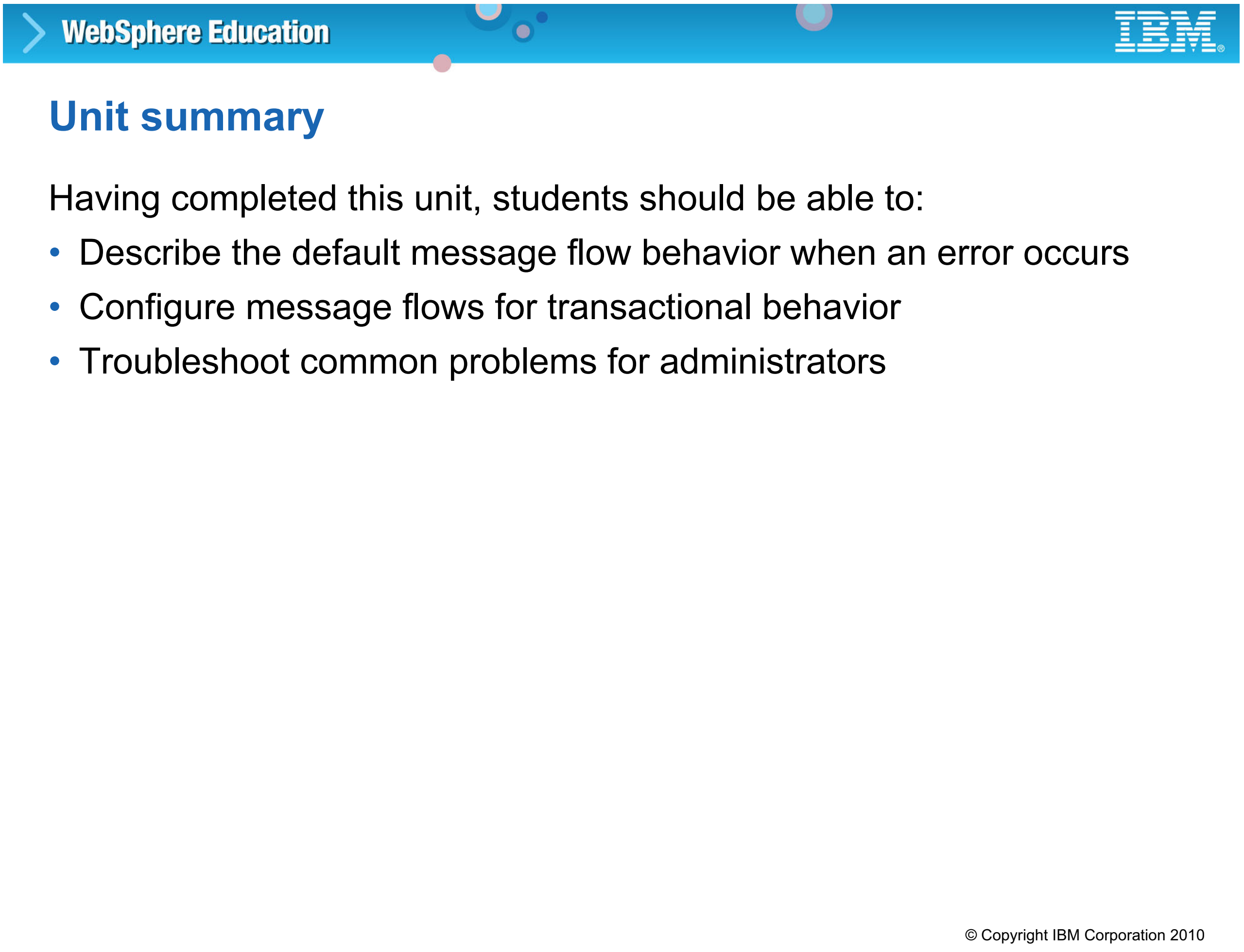
Instructor notes:

Purpose — Provide information about how to get product fixes.

Details —

Additional information —

Transition statement — Next: Unit summary



WebSphere Education
IBM

Unit summary

Having completed this unit, students should be able to:

- Describe the default message flow behavior when an error occurs
- Configure message flows for transactional behavior
- Troubleshoot common problems for administrators

Figure 6-45. Unit summary WM643 / VM6431.0

Notes:

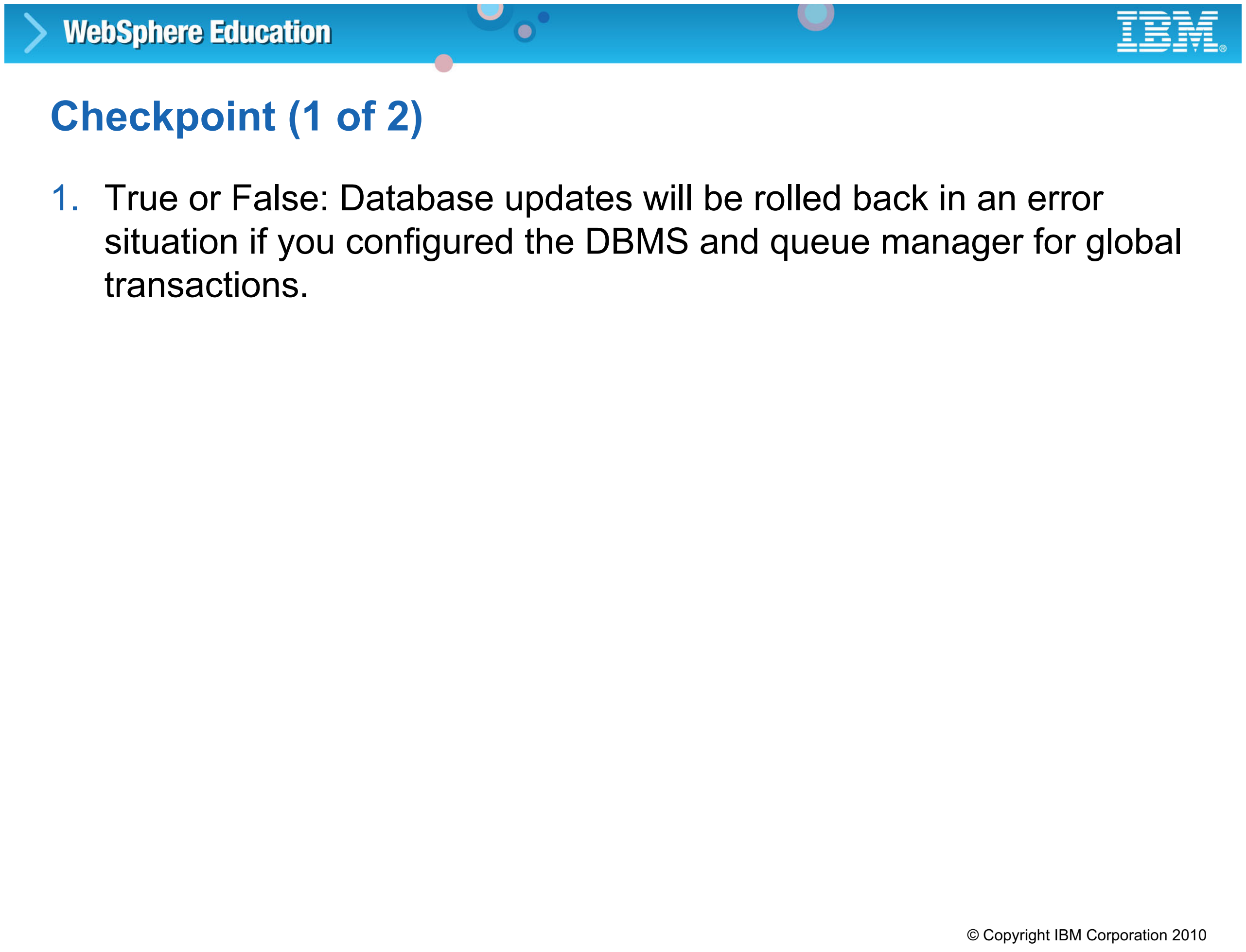
Instructor notes:

Purpose — Summarize the unit.

Details — None.

Additional information — None.

Transition statement — When you have finished the exercise, you will learn about some performance considerations when working with the WebSphere Message Broker.



WebSphere Education

Checkpoint (1 of 2)

1. True or False: Database updates will be rolled back in an error situation if you configured the DBMS and queue manager for global transactions.

Figure 6-46. Checkpoint (1 of 2) WM643 / VM6431.0

Notes:

Write your answers here:

1.

.

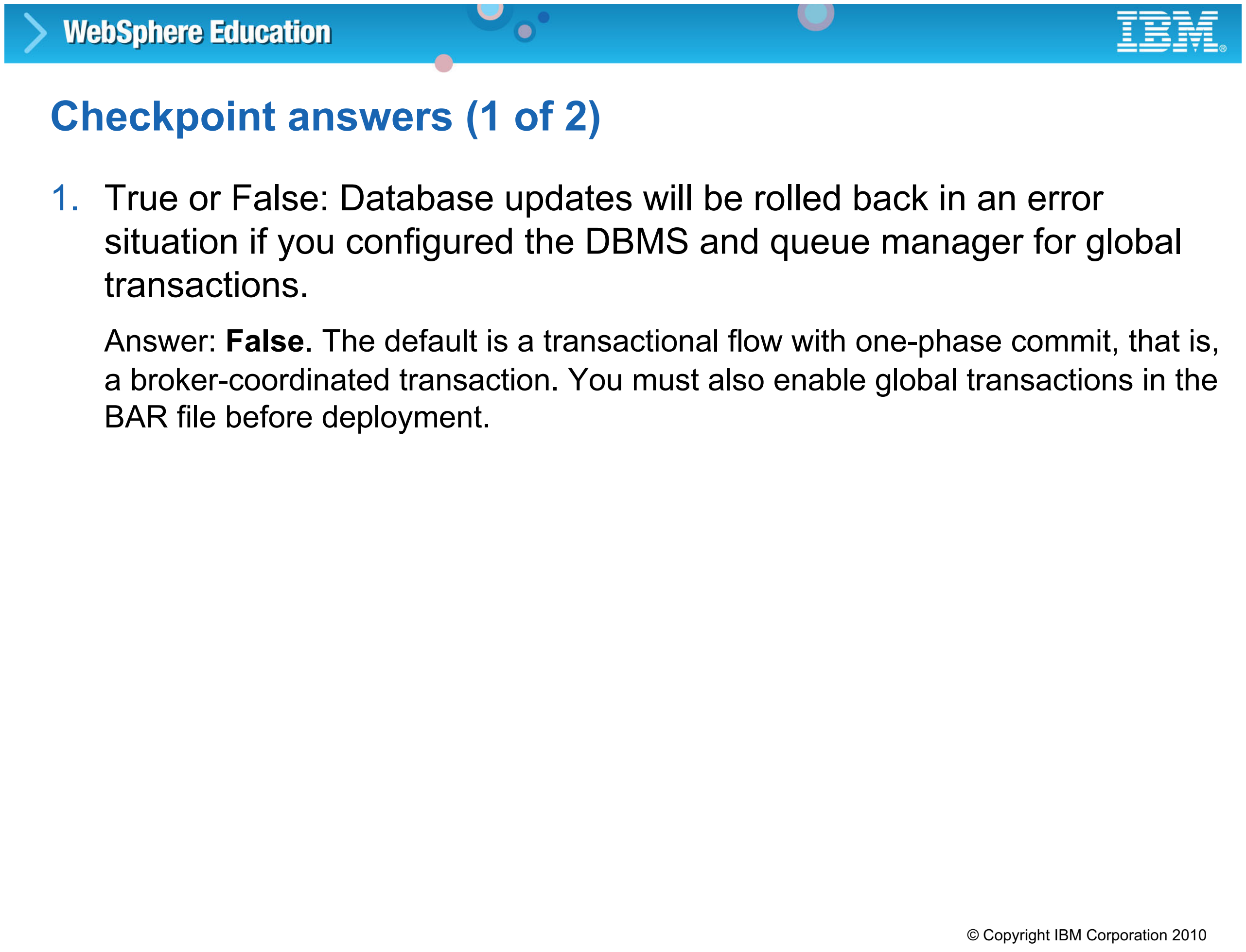
Instructor notes:

Purpose — Review the unit with some checkpoint questions.

Details — These are objective questions appropriate for SPVC.

Additional information —

Transition statement — Next: Checkpoint answers (1 of 2)



WebSphere Education

Checkpoint answers (1 of 2)

1. True or False: Database updates will be rolled back in an error situation if you configured the DBMS and queue manager for global transactions.

   Answer: **False**. The default is a transactional flow with one-phase commit, that is, a broker-coordinated transaction. You must also enable global transactions in the BAR file before deployment.

Figure 6-47. Checkpoint answers (1 of 2) WM643 / VM6431.0

Notes:

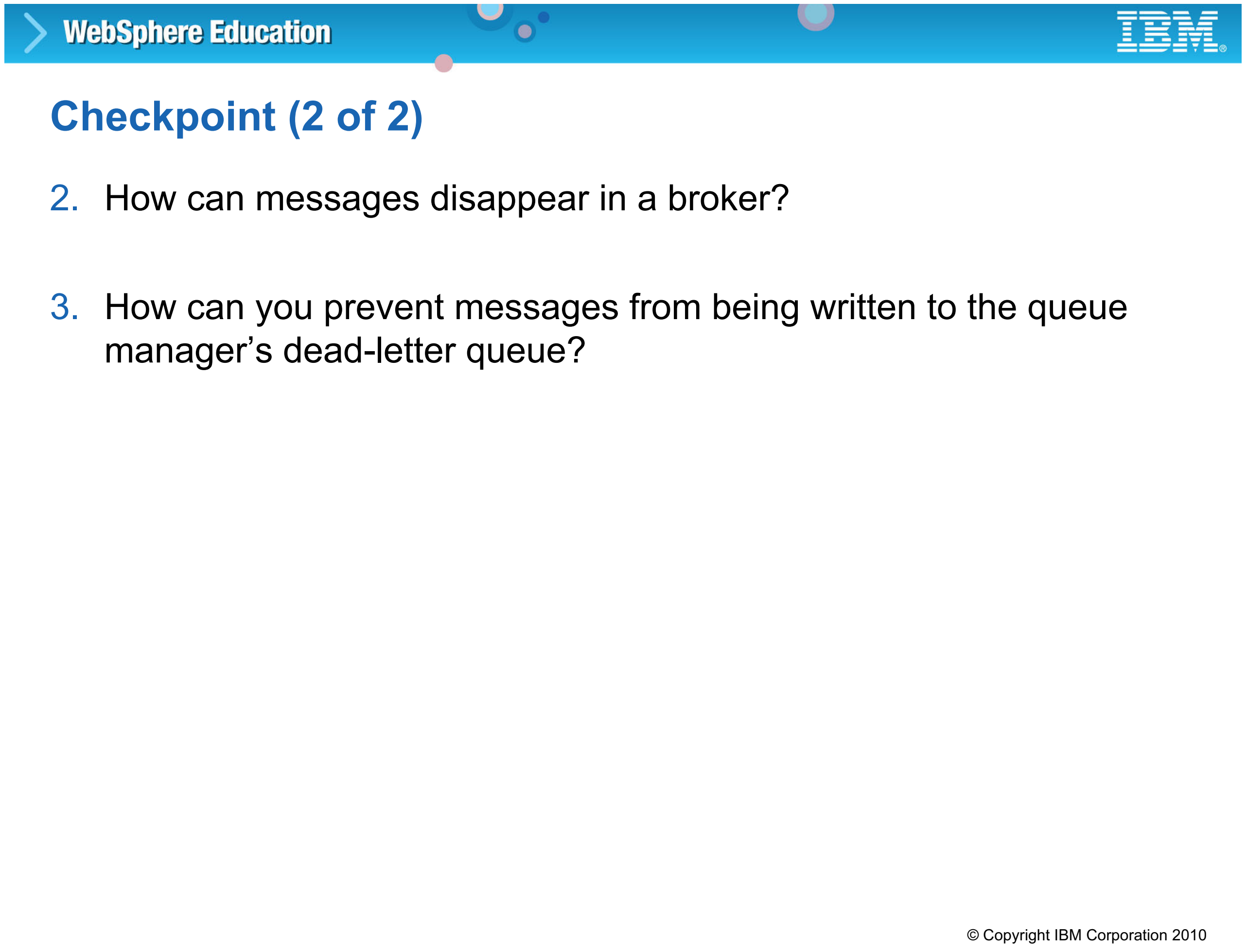
Instructor notes:

Purpose — Review the unit with some checkpoint questions.

Details —

Additional information —

Transition statement — Next: Checkpoint (2 of 2)



WebSphere Education

Checkpoint (2 of 2)

2. How can messages disappear in a broker?

3. How can you prevent messages from being written to the queue manager's dead-letter queue?

Figure 6-48. Checkpoint (2 of 2) WM643 / VM6431.0

Notes:

Write your answers here:

2.

3.

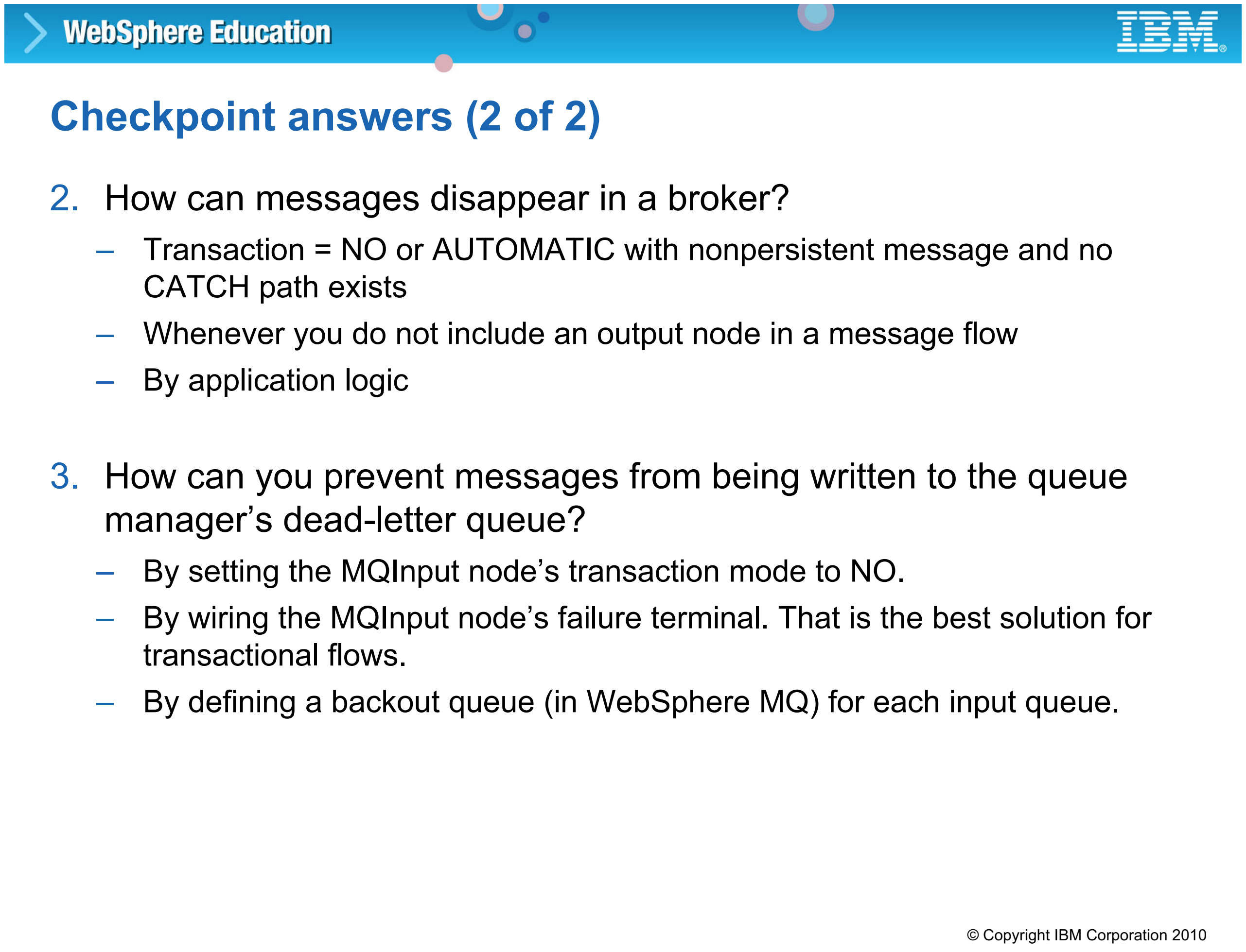
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement — Next: Checkpoint answers (2 of 2)



WebSphere Education

Checkpoint answers (2 of 2)

2. How can messages disappear in a broker?
   - Transaction = NO or AUTOMATIC with nonpersistent message and no CATCH path exists
   - Whenever you do not include an output node in a message flow
   - By application logic

3. How can you prevent messages from being written to the queue manager’s dead-letter queue?
   - By setting the MQInput node’s transaction mode to NO.
   - By wiring the MQInput node’s failure terminal. That is the best solution for transactional flows.
   - By defining a backout queue (in WebSphere MQ) for each input queue.

Figure 6-49. Checkpoint answers (2 of 2) WM643 / VM6431.0

Notes:

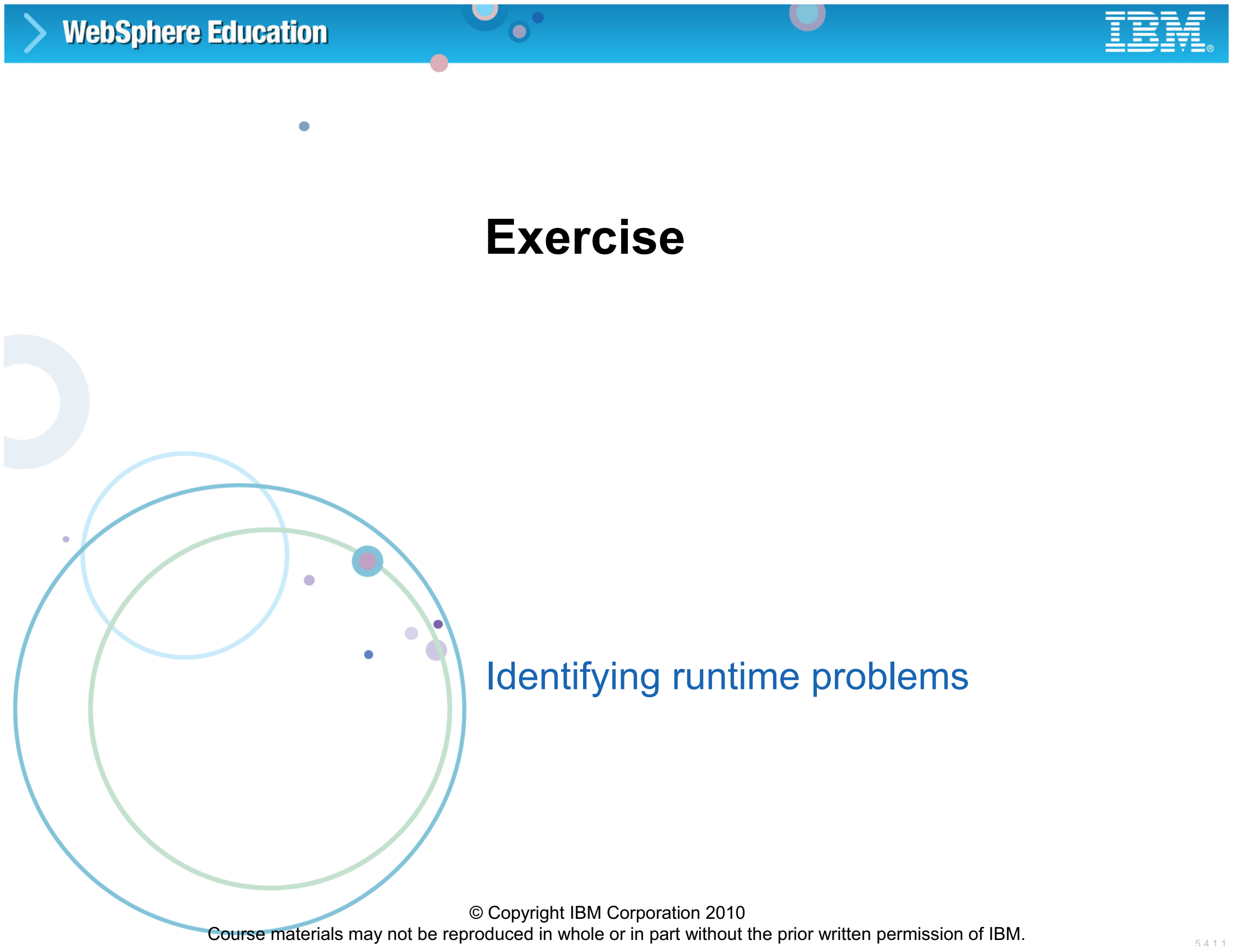
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement — Next: Exercise 6: Identifying runtime problems



Exercise

Identifying runtime problems

Figure 6-50. Exercise 6: Identifying runtime problems WM643 / VM6431.0

Notes:

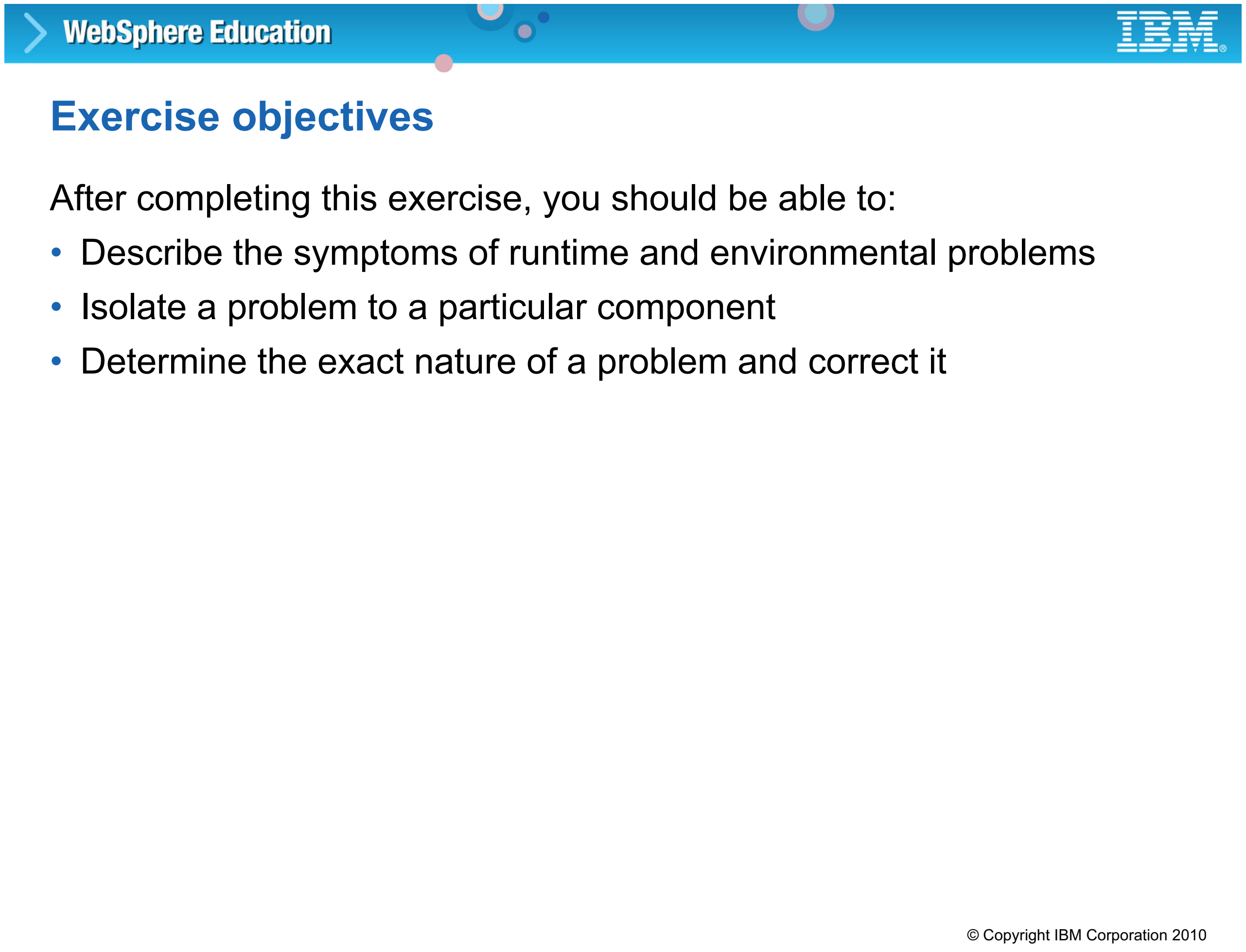
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement — Next: Exercise objectives



WebSphere Education

Exercise objectives

After completing this exercise, you should be able to:

- Describe the symptoms of runtime and environmental problems
- Isolate a problem to a particular component
- Determine the exact nature of a problem and correct it

Figure 6-51. Exercise objectives WM643 / VM6431.0

Notes:

In this exercise, you will analyze and correct some error situations that involve the WebSphere Message Broker and the WebSphere MQ queue manager running on the local computer.

Proceed to the *Student Exercise Guide* and complete Exercise 6.

Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —

Unit 7. Monitoring broker and message flow performance

Estimated time

01:00 lecture

What this unit is about

The first part of this unit describes the tools available to enable performance statistics monitoring for a running broker, broker resources, and message flows. Statistics can be used to determine broker activity, for either charge-back purposes or to determine broker workload.

The second part of this unit, describes message flow event monitoring and how to create a monitoring profile that can be used to update monitoring options at run time.

The third part of this unit provides some guidance on performance tuning.

What you should be able to do

- Explain broker runtime measurement capabilities
- Describe the tools used to format gathered information, including text output and XML output
- View runtime statistics in WebSphere Message Broker Explorer
- Create an event monitoring profile
- Identify opportunities for improving performance

How you will check your progress

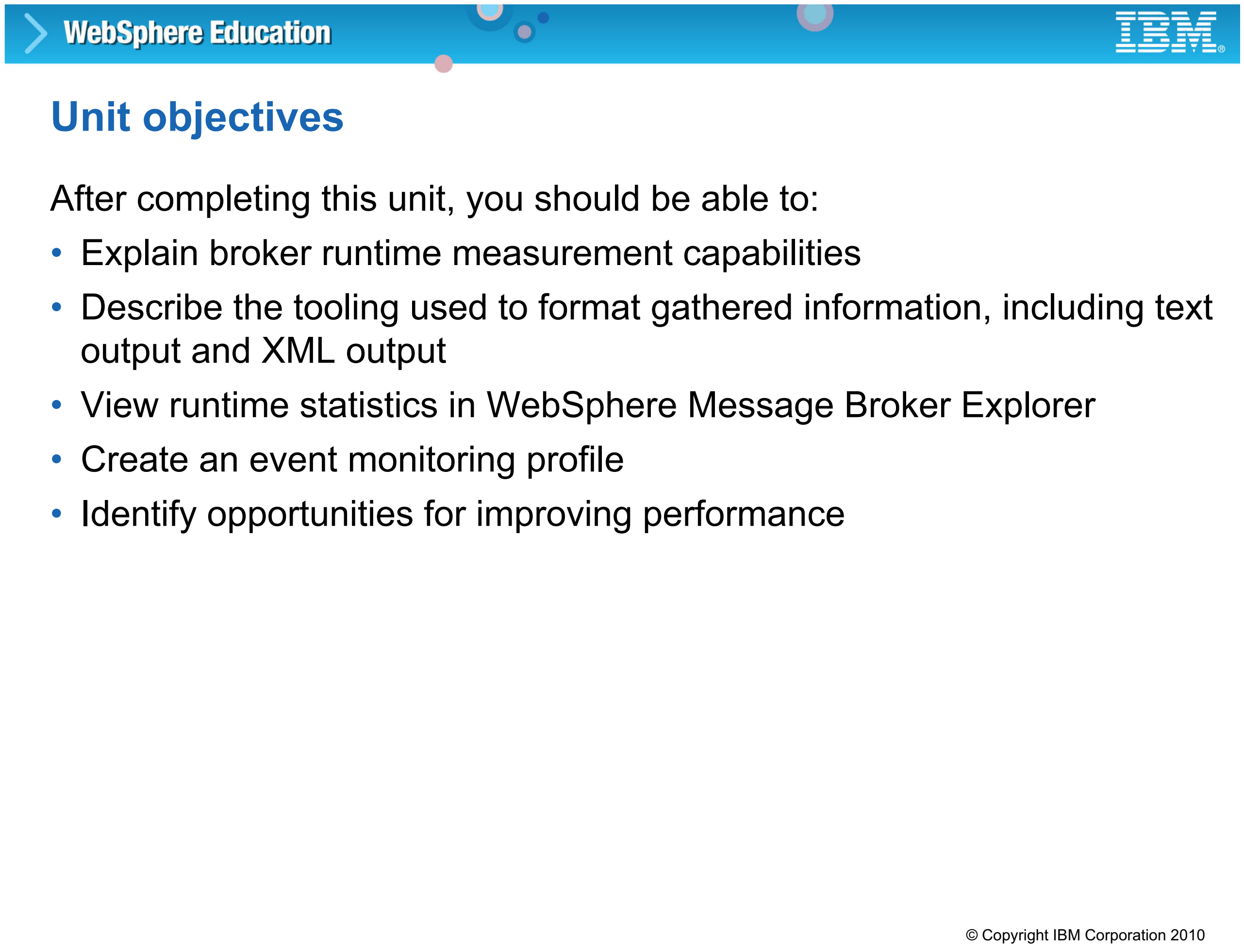
Accountability:

- Checkpoint questions
- Exercise 7, Accessing broker statistics (after the next unit)

References

`www.ibm.com/software/integration/wbimessagebroker/library`
IBM WebSphere Message Broker Library

`www.redbooks.ibm.com`
Managing WebSphere Message Broker Resources in a Production Environment (SG24-7283-00)



WebSphere Education
IBM

Unit objectives

After completing this unit, you should be able to:

- Explain broker runtime measurement capabilities
- Describe the tooling used to format gathered information, including text output and XML output
- View runtime statistics in WebSphere Message Broker Explorer
- Create an event monitoring profile
- Identify opportunities for improving performance

Figure 7-1. Unit objectives

WM643 / VM6431.0

Notes:

Instructor notes:

Purpose — Give students an overview of the topics included in this unit.

Details — None.

Additional information —

Transition statement — The first topic describes how to capture broker and message flow statistics that can be used to help you better manage system resources.

7.1. Resource and message flow accounting and statistics

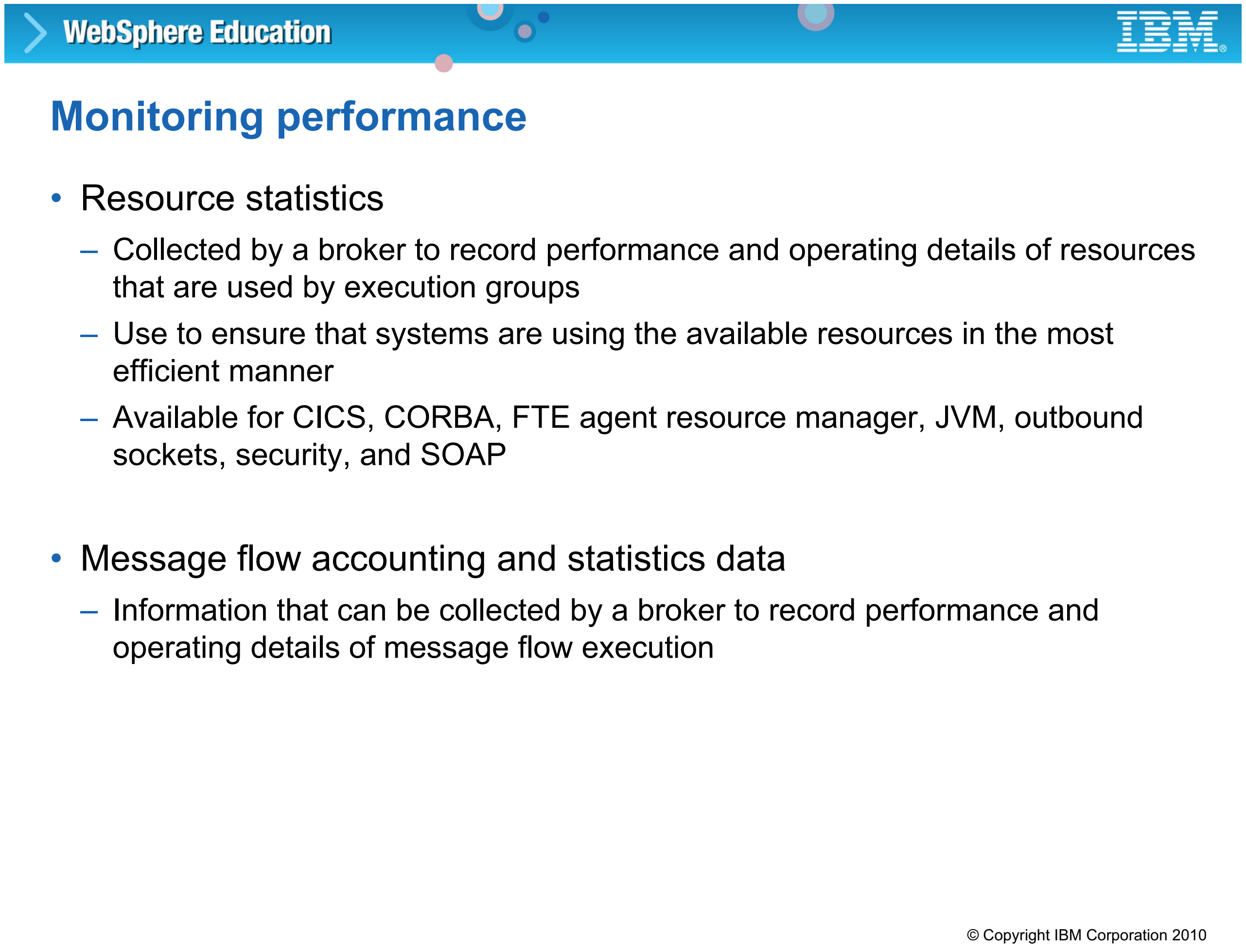
Instructor topic introduction

What students will do — Students will be presented with a review of broker statistics recording capabilities and the tools available to convert this information into forms suitable for analysis.

How students will do it — Listen to the lecture and participate in classroom discussion. The exercise at the end of the next unit gives students the opportunity to gather and format broker statistics.

What students will learn — To use broker recording facilities to derive meaningful, accurate information regarding broker activity.

How this will help students in their job — WebSphere Message Broker statistics can form the basis for customer billing (charge back) and broker tuning options.



WebSphere Education
IBM

Monitoring performance

- Resource statistics
  - Collected by a broker to record performance and operating details of resources that are used by execution groups
  - Use to ensure that systems are using the available resources in the most efficient manner
  - Available for CICS, CORBA, FTE agent resource manager, JVM, outbound sockets, security, and SOAP
- Message flow accounting and statistics data
  - Information that can be collected by a broker to record performance and operating details of message flow execution

Figure 7-2. Monitoring performance

WM643 / VM6431.0

Notes:

Many enterprises would like to be able to charge their customers according to how much they use the broker. For instance, they might assign a nominal dollar cost to each message processed by a particular flow. In another case, a broker might not be performing optimally, or the infrastructure or functional area might want to understand in more detail how a message flow or node is performing. Using the WebSphere Message Broker accounting and statistics features, administrators can dynamically measure various runtime attributes of broker resources. For example, the total number of messages processed in a given time interval, itemizing the associated processor cost, average message size, and even the total number of bytes processed can be accumulated.

The WebSphere Message Broker accounting and statistics features can also be used to understand how messages are being processed. For example, it can demonstrate that a particular error path is being taken with greater than expected frequency. This could lead to system or message flow modifications to make the message processing more effective.

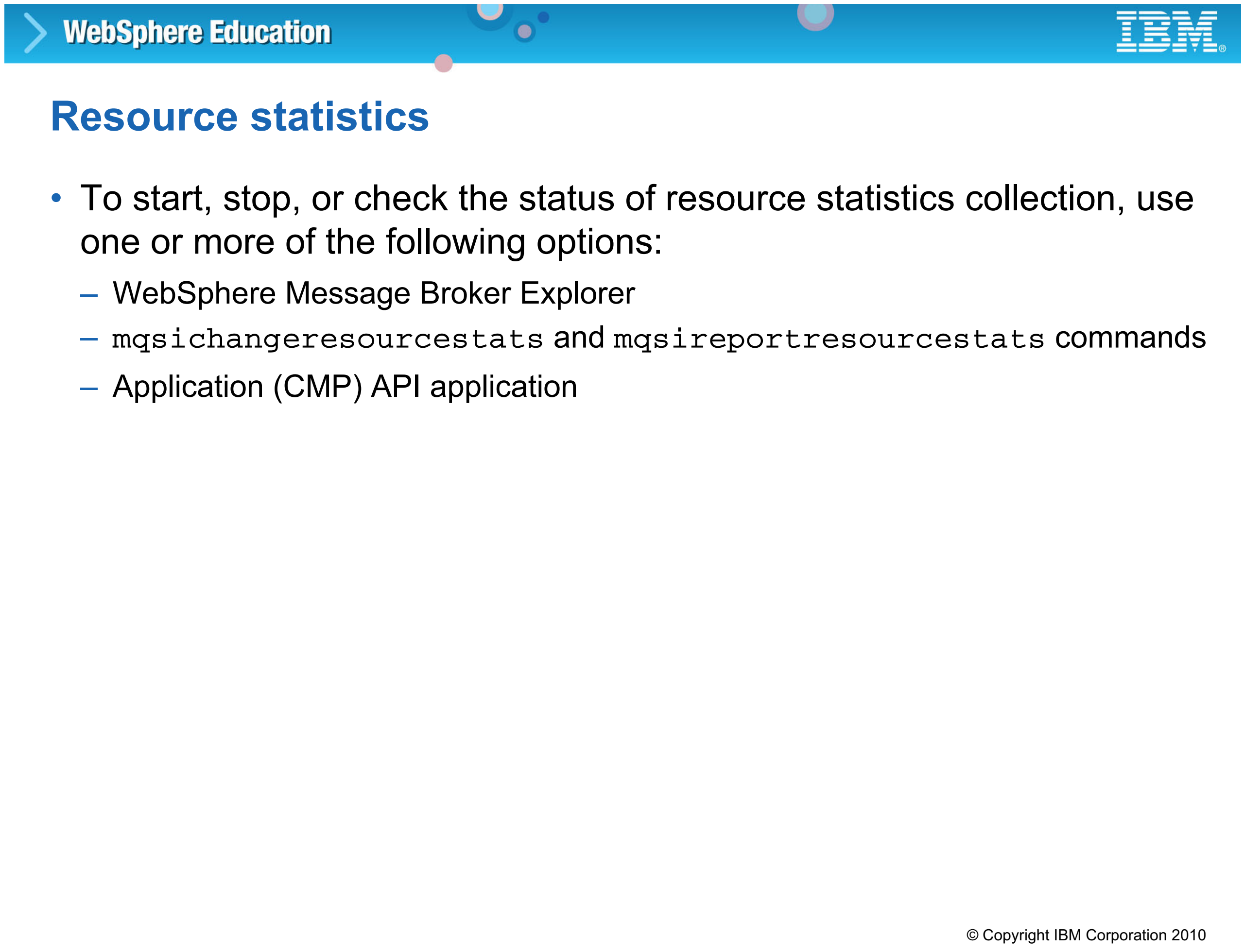
Instructor notes:

Purpose — Establish some bounds on what information is collected and when.

Details — None.

Additional information — None.

Transition statement — It is important to understand when statistics data is written so that you know whether it represents a complete time interval.



WebSphere Education

Resource statistics

- To start, stop, or check the status of resource statistics collection, use one or more of the following options:
  - WebSphere Message Broker Explorer
  - `mqsichangeresourcestats` and `mqsireportresourcestats` commands
  - Application (CMP) API application

Figure 7-3. Resource statistics WM643 / VM6431.0

Notes:

WebSphere Message Broker resource statistics can be used to ensure that systems are using the available resources in the most efficient manner. If you detect that system resources are under pressure, you can examine the statistics collected by the broker to assess whether the cause of the concern is the use of those resources by processes in WebSphere Message Broker.

Resource statistic collection can be managed from WebSphere Message Broker Explorer, the command-line interface, or from the Application API for WebSphere Message Broker.

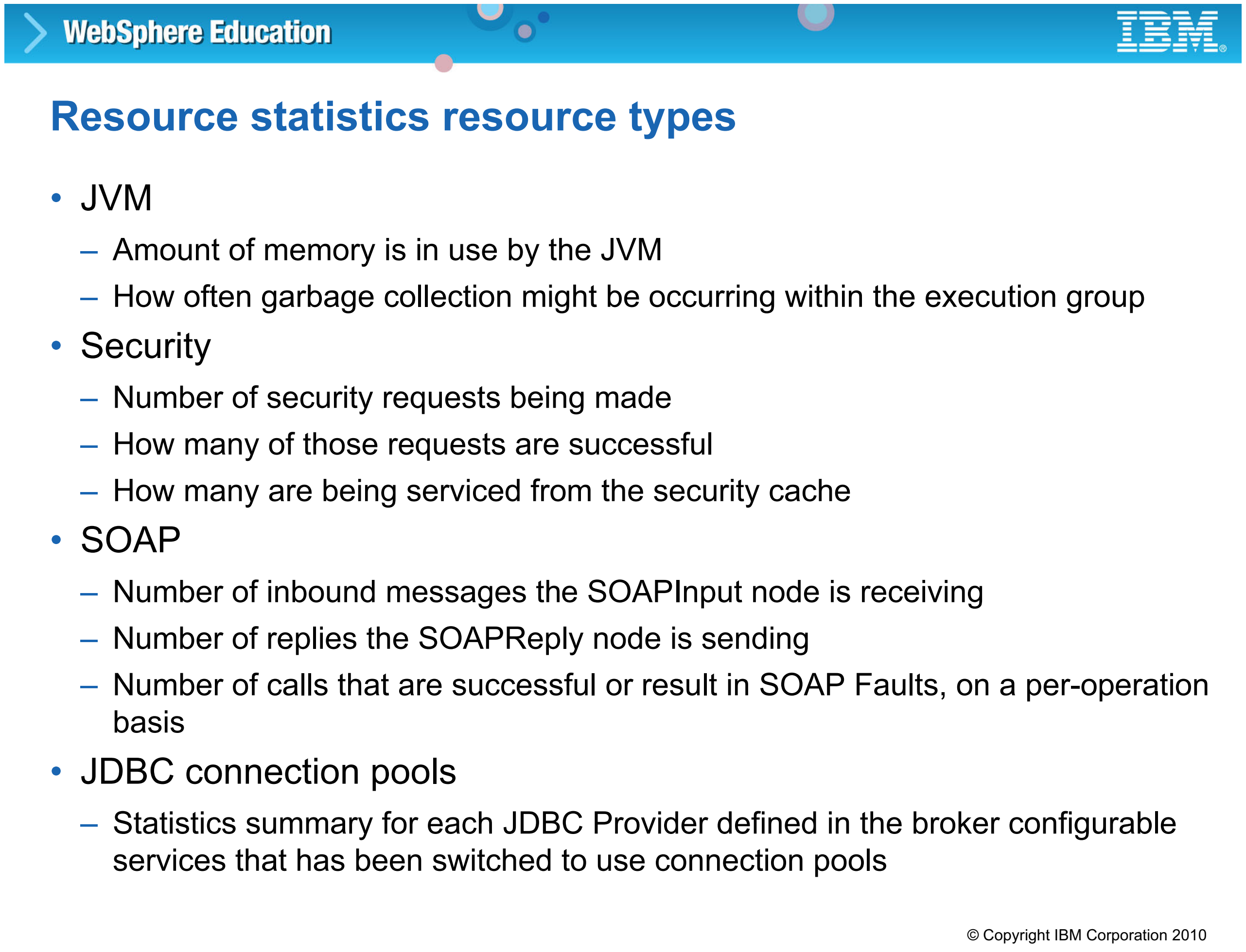
Instructor notes:

Purpose — Introduce resource statistics.

Details — This topic starts with a discussion of resource statistics. Message flow statistics are described later in this topic.

Additional information — None.

Transition statement — Resource statistics are available for a number of resource types.



WebSphere Education

Resource statistics resource types

- JVM
  - Amount of memory is in use by the JVM
  - How often garbage collection might be occurring within the execution group
- Security
  - Number of security requests being made
  - How many of those requests are successful
  - How many are being serviced from the security cache
- SOAP
  - Number of inbound messages the SOAPInput node is receiving
  - Number of replies the SOAPReply node is sending
  - Number of calls that are successful or result in SOAP Faults, on a per-operation basis
- JDBC connection pools
  - Statistics summary for each JDBC Provider defined in the broker configurable services that has been switched to use connection pools

Figure 7-4. Resource statistics resource types WM643 / VM6431.0

Notes:

The figure lists some of the resource types for which statistics can be collected.

Other resource types are:

- Outbound sockets
- FTEAgent
- CICS
- CORBA

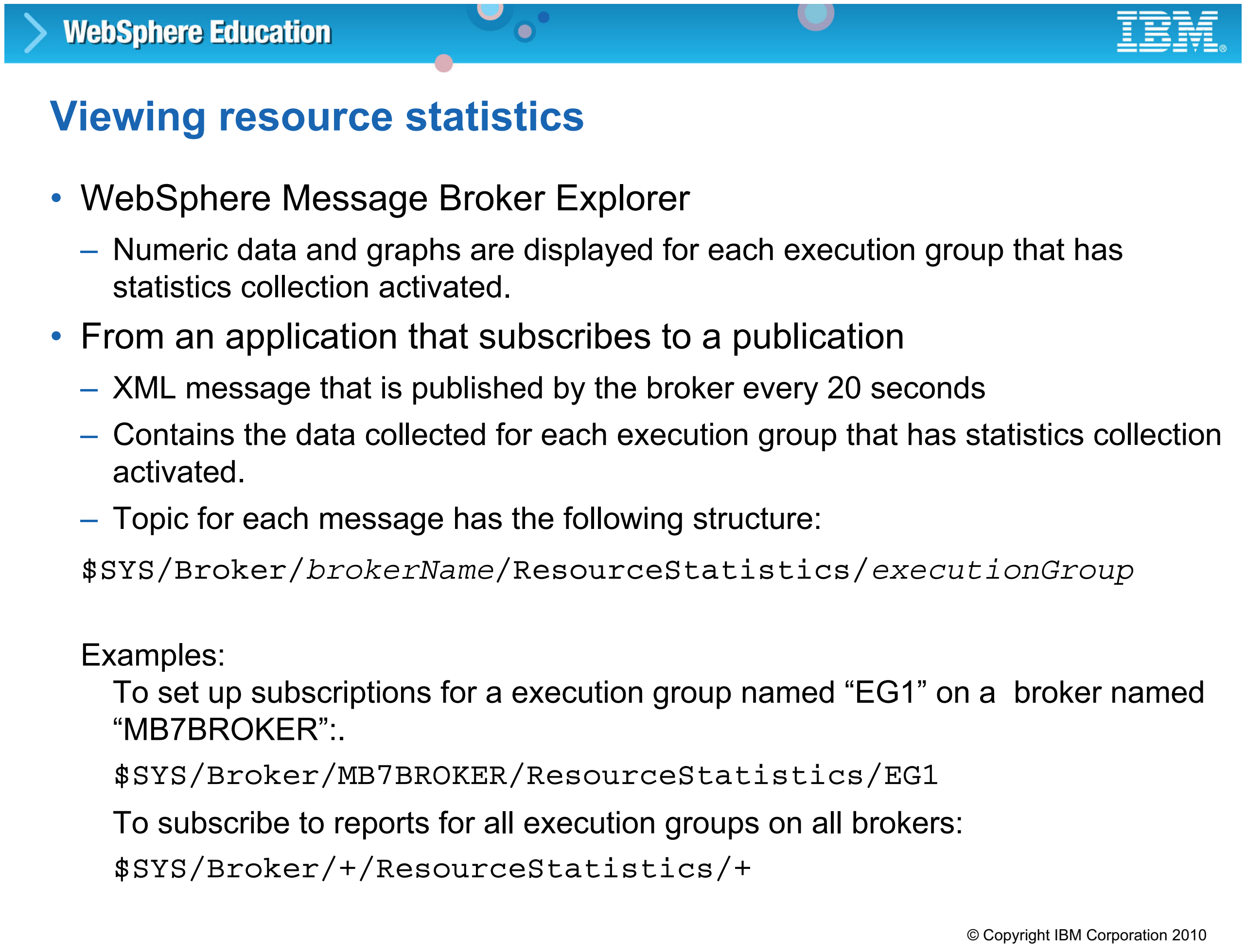
Instructor notes:

Purpose — List some of the resource types and statistics that can be collected.

Details — This figure only lists some of the resource types. It is not a comprehensive list.

Additional information — The Information Center includes a full list of all the statistics that can be collected for any resource.

Transition statement — Statistic reports can be generated to include detailed information written to a choice of locations.



WebSphere Education

Viewing resource statistics

- WebSphere Message Broker Explorer
  - Numeric data and graphs are displayed for each execution group that has statistics collection activated.
- From an application that subscribes to a publication
  - XML message that is published by the broker every 20 seconds
  - Contains the data collected for each execution group that has statistics collection activated.
  - Topic for each message has the following structure:

`$SYS/Broker/`*`brokerName`*`/ResourceStatistics/`*`executionGroup`*

Examples:

To set up subscriptions for a execution group named “EG1” on a broker named “MB7BROKER”:.

```
$SYS/Broker/MB7BROKER/ResourceStatistics/EG1
```

To subscribe to reports for all execution groups on all brokers:

```
$SYS/Broker/+/ResourceStatistics/+
```

Figure 7-5. Viewing resource statistics WM643 / VM6431.0

Notes:

You can examine the statistics collected by the broker to assess whether the cause of the concern is the use of those resources by processes in WebSphere Message Broker.

You can view the statistics in WebSphere Message Broker Explorer or from an application that subscribes to a publication that is published by the broker every 20 seconds.

Snapshot publications are produced according to the topic tree:

```
$SYS/Broker/brokerName/StatisticsAccounting/Snapshot/
executionGroup/messageFlow/
```

Archive publications are produced according to the topic tree:

```
$SYS/Broker/.../StatisticsAccounting/Archive/.../.../
```

By using ESQL filters you can further refine your subscriptions to examine the content of accounting and statistics messages. You can register a subscription specifying information with only certain values, maybe to generate alerts when certain node terminals (such as CATCH) are being driven more than you expect.

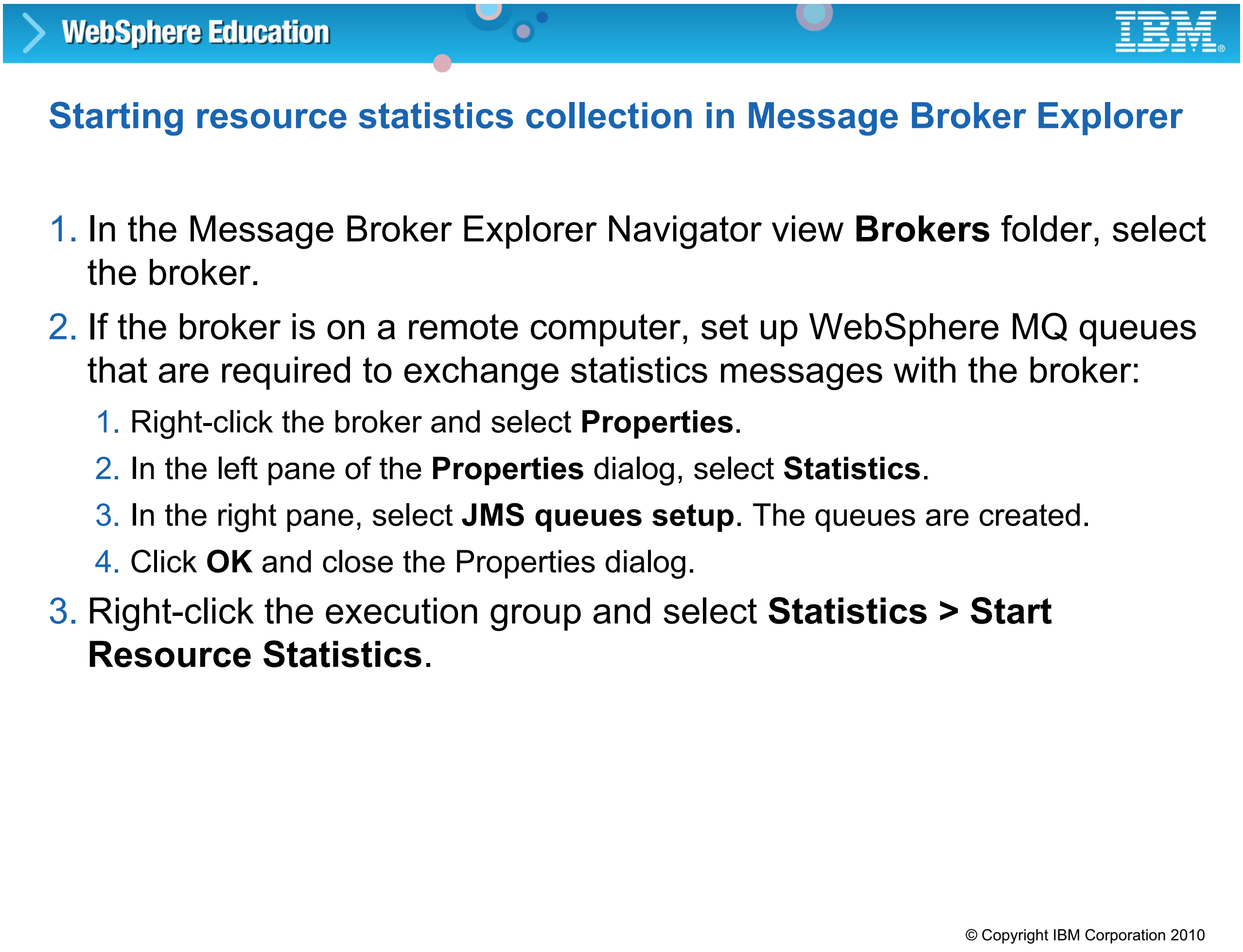
Instructor notes:

Purpose — Describe options for viewing resource statistics.

Details — None.

Additional information — WebSphere Studio Application Developer can be used to browse and format the XML statistics messages. The message must be imported to WebSphere Studio Application Developer, formatted, and exported again into a file. The exported file can then be imported to a spreadsheet for analysis. Many other procedures can be used to retrieve the XML statistics messages and inject them into a spreadsheet.

Transition statement — Viewing statistics in WebSphere Message Broker Explorer is a simple process, but first, you must start resource collection.



WebSphere Education

Starting resource statistics collection in Message Broker Explorer

1. In the Message Broker Explorer Navigator view **Brokers** folder, select the broker.
2. If the broker is on a remote computer, set up WebSphere MQ queues that are required to exchange statistics messages with the broker:
    1. Right-click the broker and select **Properties**.
    2. In the left pane of the **Properties** dialog, select **Statistics**.
    3. In the right pane, select **JMS queues setup**. The queues are created.
    4. Click **OK** and close the Properties dialog.
3. Right-click the execution group and select **Statistics > Start Resource Statistics**.

Figure 7-6. Starting resource statistics collection in Message Broker Explorer WM643 / VM6431.0

Notes:

Before you can view resource statistics, you must start resource statistics collection, as described in the figure.

If you want statistics for several execution groups on a broker, select more than one by using standard operating system interfaces; for example, on Windows systems, hold down the **Ctrl** key and select each execution group before you right-click to open the menu.

When you click **Statistics > Start Resource Statistics**, a message is sent to the broker to start collecting data for the selected execution groups. The property **Resource Statistics Active** is updated in the properties view of each affected execution group to indicate that data collection is now active. A warning message is displayed in the execution group properties window to warn you that performance might be affected by your action.

Instructor notes:

Purpose — List the steps for activating resource statistics collection.

Details — The command-line equivalent is covered later in this topic.

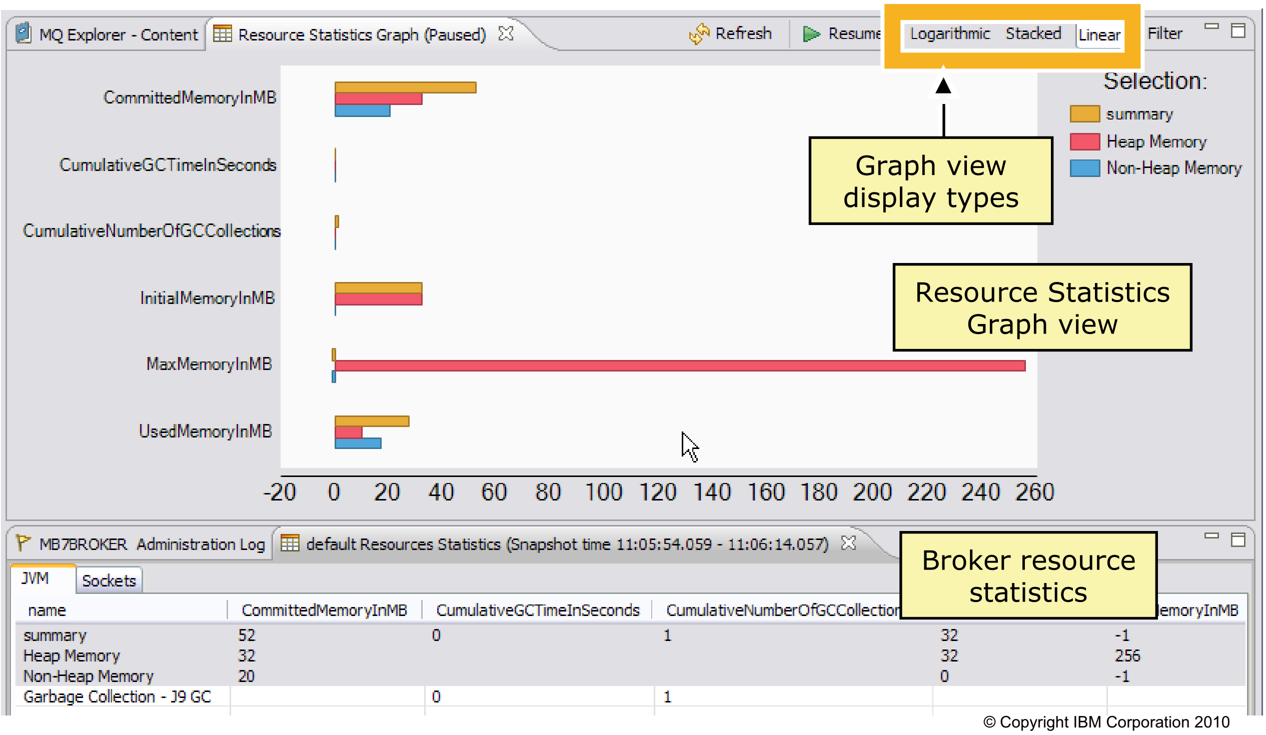
Additional information — None.

Transition statement — After statistics collection is started, it only takes a few clicks to view the statistics.

WebSphere Education

Viewing resource statistics in Message Broker Explorer

- Select **Window > Show View > Resource Statistics**


MQ Explorer - Content
Resource Statistics Graph (Paused)
Refresh
Resume
Logarithmic
Stacked
Linear
Filter
Selection:
summary
Heap Memory
Non-Heap Memory
CommittedMemoryInMB
CumulativeGCTimeInSeconds
CumulativeNumberOfGCCollections
InitialMemoryInMB
MaxMemoryInMB
UsedMemoryInMB
-20
0
20
40
60
80
100
120
140
160
180
200
220
240
260
Graph view display types
Resource Statistics Graph view
Broker resource statistics
MB7BROKER Administration Log
default Resources Statistics (Snapshot time 11:05:54.059 - 11:06:14.057)
JVM
Sockets

name	CommittedMemoryInMB	CumulativeGCTimeInSeconds	CumulativeNumberOfGCCollection		emoryInMB
summary	52	0	1	32	-1
Heap Memory	32			32	256
Non-Heap Memory	20			0	-1
Garbage Collection - J9 GC		0	1		

Figure 7-7. Viewing resource statistics in Message Broker Explorer WM643 / VM6431.0

Notes:

To view statistics in WebSphere Message Broker Explorer, click **Window > Show View > Resource Statistics** to open the Broker resources and Resource Statistics Graph views.

If you are displaying statistics for an execution group for the first time, the views might be empty until the first data is received. Update messages are sent every 20 seconds, and the views refresh automatically.

The Broker resources and Resource Statistics Graph views are displayed together. If you close one of the views, the other view is also closed.

The **Resume**/**Pause** toggle allows you to stop updating the statistics in the views.

The Logarithmic, Stacked, and Linear buttons allow you to select the type of graph to display.

Instructor notes:

Purpose — Show an example of the broker and resource statistics views in the WebSphere Message Broker Explorer.

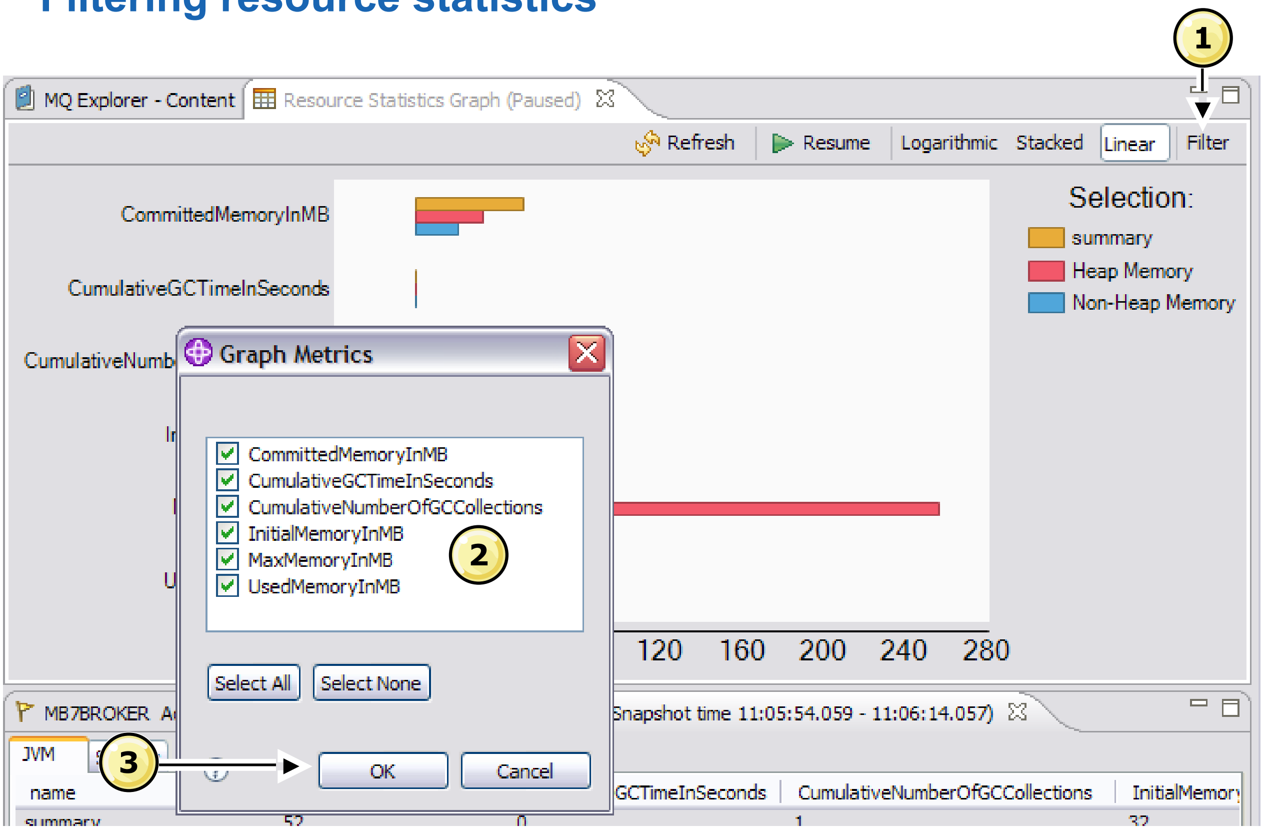
Details — The graph is color-coded; the key is on the right side of the window. The graph can be displayed as Linear, Stacked, and Logarithmic. The example in the figure is a linear graph.

Additional information — None.

Transition statement — You can filter the statistics to only display specific metrics.

WebSphere Education

Filtering resource statistics



☑ CommittedMemoryInMB
☑ CumulativeGCTimeInSeconds
☑ CumulativeNumberOfGCCollections
☑ InitialMemoryInMB
☑ MaxMemoryInMB
☑ UsedMemoryInMB

Figure 7-8. Filtering resource statistics WM643 / VM6431.0

Notes:

To filter statistics in the WebSphere Message Broker Explorer:

1. Click the **Filter** button in the Resource Statistics Graph view to display the Graph Metrics window.
2. Select the metrics that you want to display from the list.

   Clear all the metrics that you do not want to display in the Broker Statistics and Broker Statistics Graph view.
3. Click **OK**.

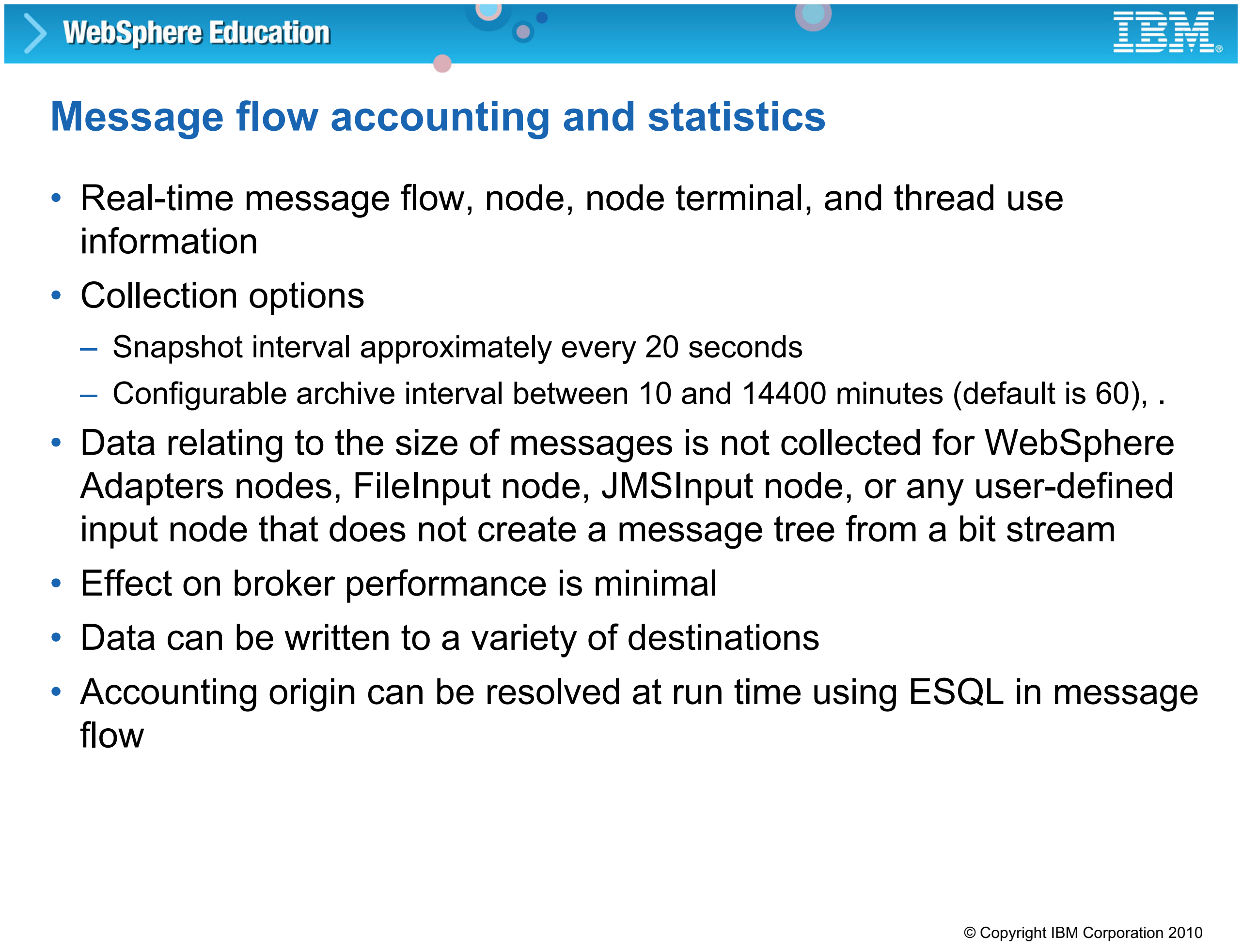
Instructor notes:

Purpose — Show how to filter resource metrics in WebSphere Message Broker Explorer.

Details — None.

Additional information — None.

Transition statement — You can also obtain statistics on message flow performance.



WebSphere Education

Message flow accounting and statistics

- Real-time message flow, node, node terminal, and thread use information
- Collection options
  - Snapshot interval approximately every 20 seconds
  - Configurable archive interval between 10 and 14400 minutes (default is 60), .
- Data relating to the size of messages is not collected for WebSphere Adapters nodes, FileInput node, JMSInput node, or any user-defined input node that does not create a message tree from a bit stream
- Effect on broker performance is minimal
- Data can be written to a variety of destinations
- Accounting origin can be resolved at run time using ESQL in message flow

Figure 7-9. Message flow accounting and statistics WM643 / VM6431.0

Notes:

You can configure your message flow to collect statistics about the operation and behavior of your message flows. Use this information to assess the performance of your message flow

Warning

The broker takes information about statistics and accounting from the operating system. On some operating systems, such as Windows, Linux, and UNIX, rounding can occur because the system calls that are used to determine the processor times are not sufficiently granular. This rounding might affect the accuracy of the data.

The effect of monitoring on broker performance is minimal. However, collecting more data than the default message flow statistics can generate high volumes of report data that might affect performance slightly.

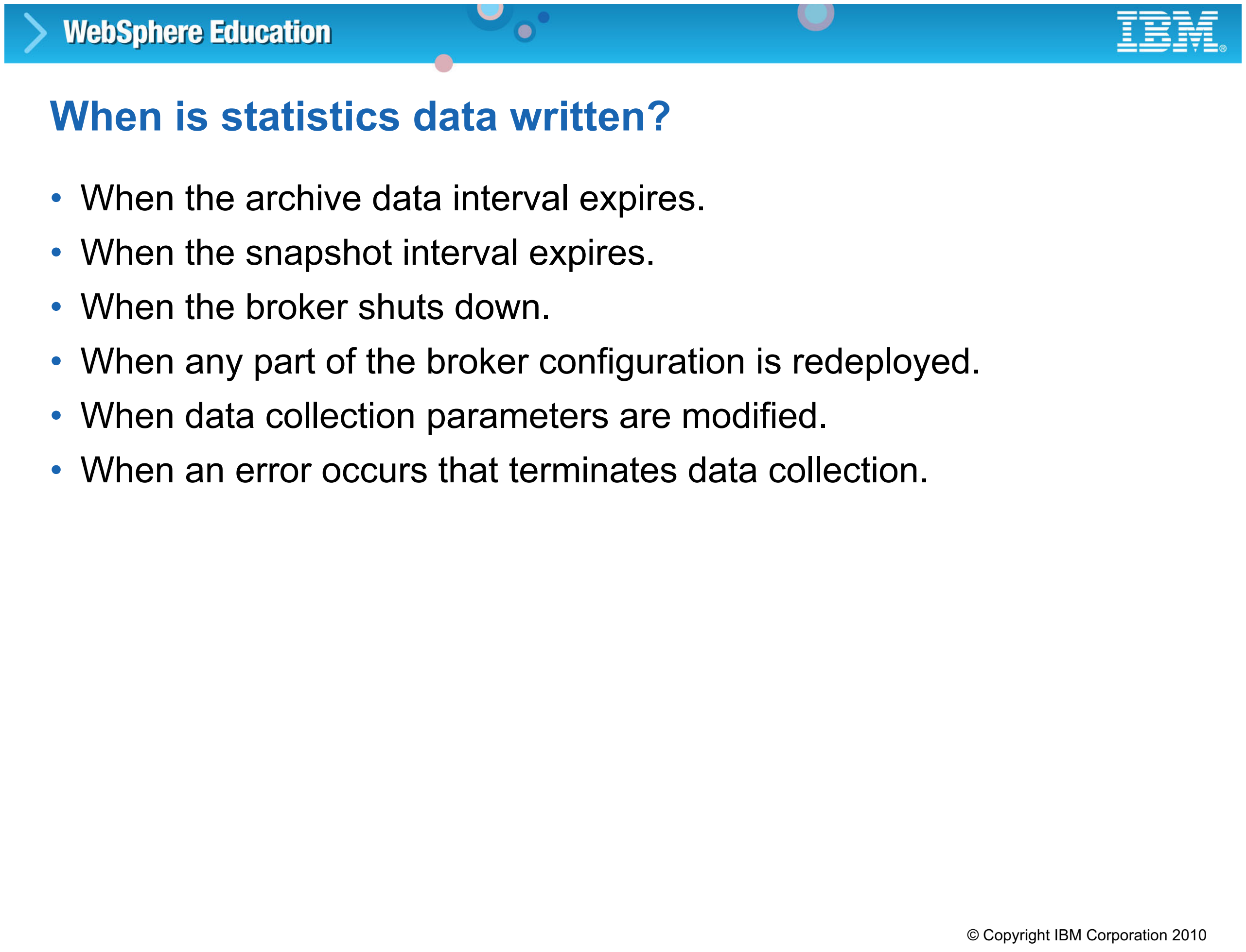
Instructor notes:

Purpose — Introduce message flow accounting and statistics.

Details — None.

Additional information — None.

Transition statement — You can collect message flow, thread, node, and terminal statistics for message flows.



WebSphere Education

When is statistics data written?

- When the archive data interval expires.
- When the snapshot interval expires.
- When the broker shuts down.
- When any part of the broker configuration is redeployed.
- When data collection parameters are modified.
- When an error occurs that terminates data collection.

Figure 7-10. When is statistics data written? WM643 / VM6431.0

Notes:

The figure lists the circumstances under which statistics data is written to the specified output location.

During broker shutdown, any data that has been collected by the broker, but has not yet been written to the specified output destination, is written to the output location. It might therefore represent data for an incomplete interval.

Redeployed configuration data might contain an updated configuration that is not consistent with the existing record structure (for example, a message flow might include an additional node). Therefore the current data, which might represent an incomplete interval, is written to the output destination.

If you update the parameters that you have set for data collection, all data that is collected for the message flow (or message flows) is written to the output destination to retain data integrity. Statistics collection is restarted according to the new parameters.

Instructor notes:

Purpose — Identify when statistics data is written.

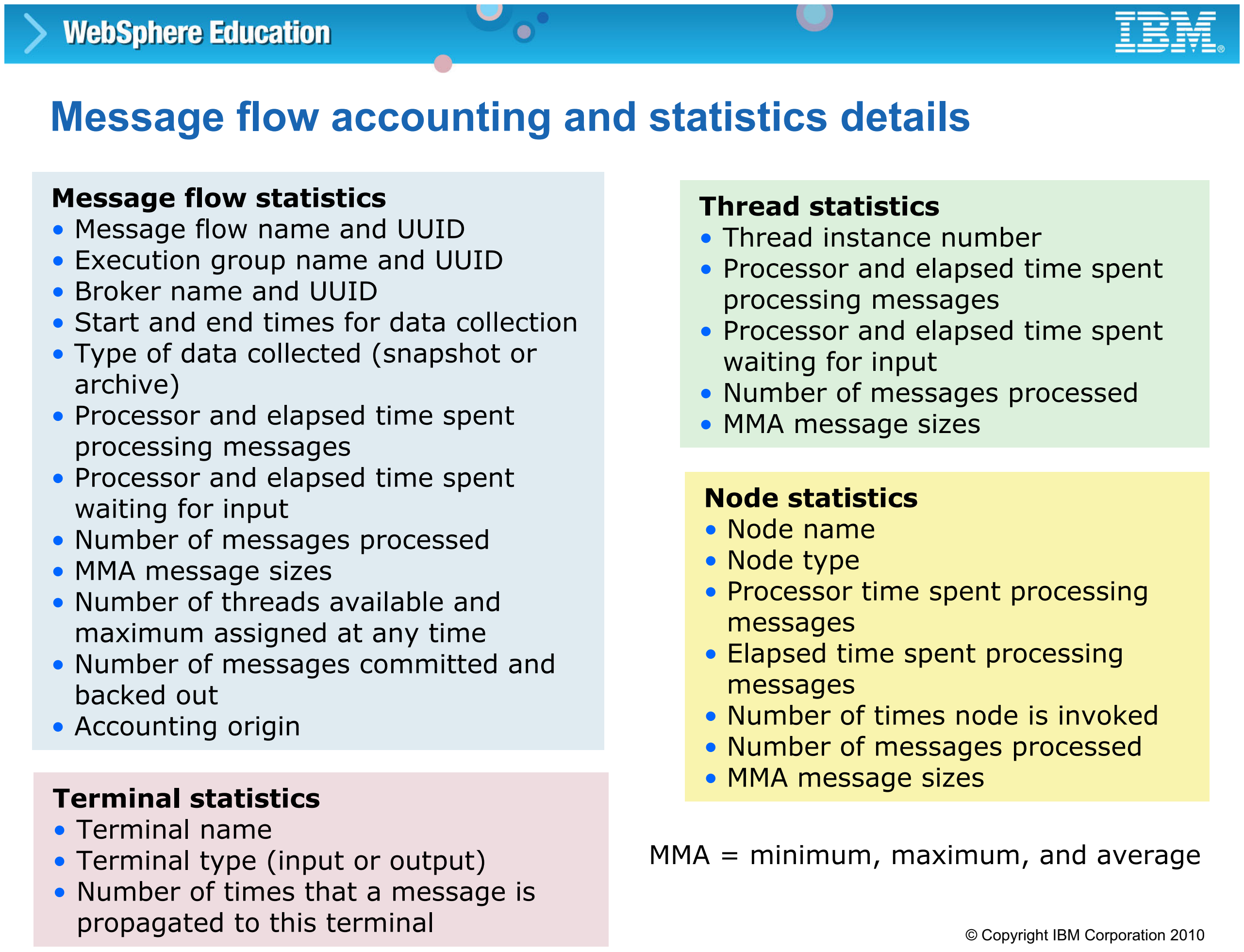
Details — There are two time periods over which statistics are available: *snapshot* (seconds) and *archive* (minutes to hours). Snapshot is usually used when there is a specific performance problem to analyze; archive statistics are used for charge-back and long-term performance monitoring.

Data collection continues for the redeployed configuration until the data collection parameters are changed or data collection is stopped.

You must restart data collection manually when an error occurs that terminates data collection.

Additional information — None.

Transition statement — Statistics can be collected for broker resources and message flows.



WebSphere Education

Message flow accounting and statistics details

Message flow statistics
- Message flow name and UUID
- Execution group name and UUID
- Broker name and UUID
- Start and end times for data collection
- Type of data collected (snapshot or archive)
- Processor and elapsed time spent processing messages
- Processor and elapsed time spent waiting for input
- Number of messages processed
- MMA message sizes
- Number of threads available and maximum assigned at any time
- Number of messages committed and backed out
- Accounting origin

Terminal statistics
- Terminal name
- Terminal type (input or output)
- Number of times that a message is propagated to this terminal

Thread statistics
- Thread instance number
- Processor and elapsed time spent processing messages
- Processor and elapsed time spent waiting for input
- Number of messages processed
- MMA message sizes

Node statistics
- Node name
- Node type
- Processor time spent processing messages
- Elapsed time spent processing messages
- Number of times node is invoked
- Number of messages processed
- MMA message sizes

MMA = minimum, maximum, and average

© Copyright IBM Corporation 2010

Figure 7-11. Message flow accounting and statistics details WM643 / VM6431.0

Notes:

There are four types of data records collected by the WebSphere Message Broker accounting and statistics feature. These correspond to message flows, processing nodes, node terminals, and threads (message flow instances).

Message flow statistics include the execution group name, broker name, and message flow name. You also receive the message flow labels and UUIDs. Message flow statistics also include:

- Type of data collected indicating whether the data was collected during an archive (long-term) or snapshot (short-term) interval. Since it is possible to record both during the same interval, it is up to you to make sure that events are not counted twice during analysis.
- The number of messages and their minimum, maximum, and average size during the interval. Also provided is the total number of messages and bytes processed in the interval by a message flow.

- Elapsed, processor, and wait times. These statistics show how much processor and elapsed time the flow used to process messages. For cases where I/O is an important factor, the wait times might prove more interesting.
- Time that a message flow is waiting for a message to process. This gives you an idea of the activity level of message flows.
- Total and maximum threads the flow has assigned to it, and the maximum in use at any given time during the interval. Use the thread statistics to determine whether the `.bar` file variable, **Additional Instances,** is set appropriately.
- How many messages were backed out or committed during the interval. This is a useful statistic to verify that messages are being processed as expected.

Thread statistics include:

- Arbitrary thread number. This has no significance other than identification. One record is produced per thread.
- Total messages processed by the thread during the interval.
- Elapsed, CPU, wait, and input wait times. Processing times are like those for message flows, but you can see how effectively the (number of) threads are (or are not) supporting the message flow's processing needs.
- Minimum and maximum message input size. Various statistical sizes of messages processed by the thread.

Node statistics include one statistical record per node.

- Elapsed, CPU, and wait times. Standard statistical times to give node level granularity of processing. Allows isolation of hot spots, or charging according to specific processing.
- The number of times the node has been traversed. This is useful to better understand, or identify, a critical path in a message flow.
- Message size, number processed, and bytes processed. Various statistical sizes of messages processed by the thread.

Terminals statistics provide one record for each node terminal. Statistics include labels of each terminal within a node and count of the number of times traversed. This is useful for identifying a critical path and exception processing frequency.

Instructor notes:

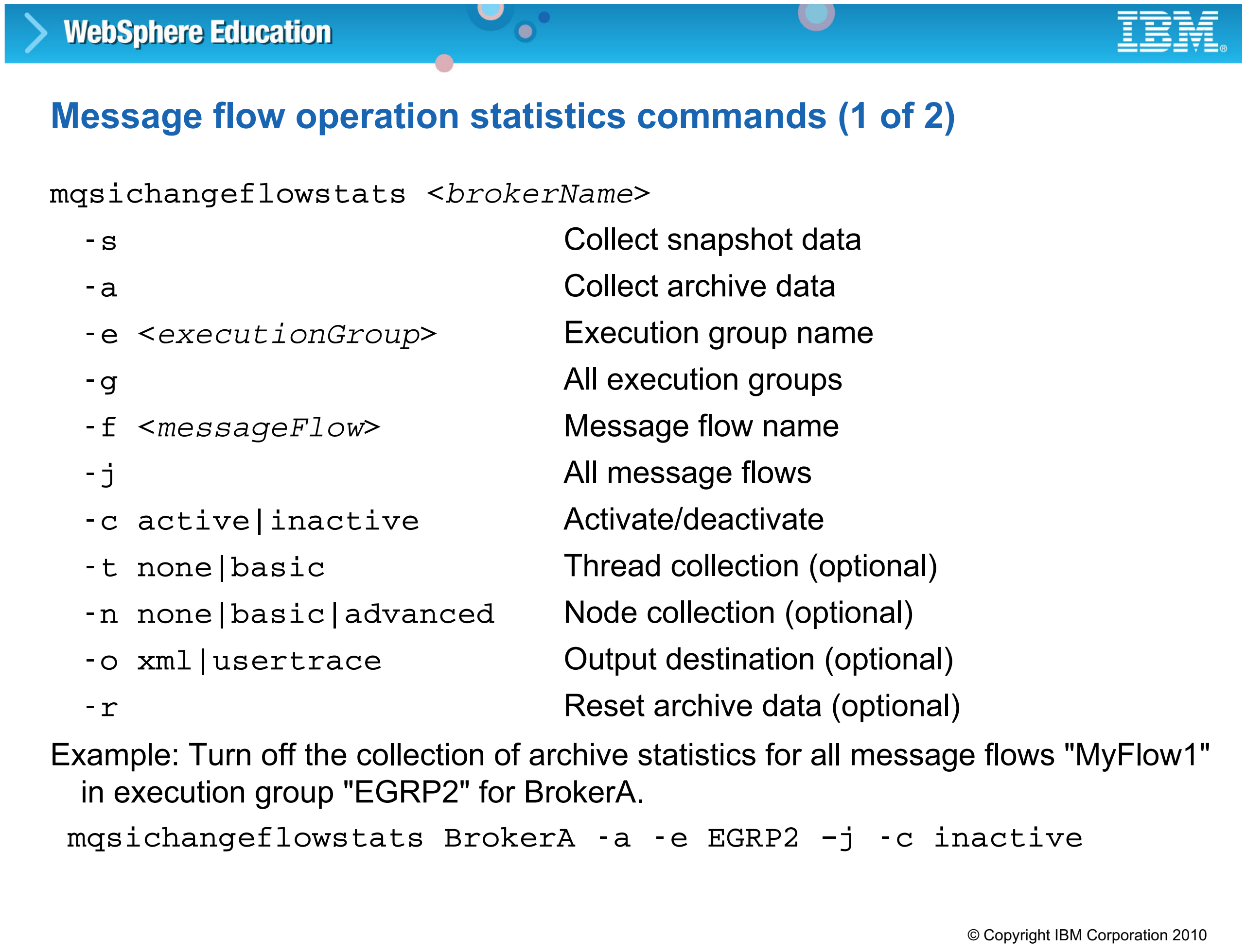
Purpose — Identify the granularity of the statistical data that can be collected.

Details — None.

Additional information — Data relating to the size of messages is not collected for WebSphere Adapters nodes (for example, the SAPInput node), the FileInput node, the JMSInput node, or any user-defined input node that does not create a message tree from a bit stream.

zSeries can use ENF37 (event notification flag 37) to drive SMF, user trace, and publish/subscribe intervals. ENF is also important to allow consolidated reporting of SMF information across major subsystems. For example, you might coordinate queue manager and broker activity to best understand how to tune your queue manager for particular message flows.

Transition statement — You can also use the command-line interface to manage statistics collection.



WebSphere Education

Message flow operation statistics commands (1 of 2)

```
mqsichangeflowstats <brokerName>
```

`-s`	Collect snapshot data
`-a`	Collect archive data
`-e <executionGroup>`	Execution group name
`-g`	All execution groups
`-f <messageFlow>`	Message flow name
`-j`	All message flows
`-c active|inactive`	Activate/deactivate
`-t none|basic`	Thread collection (optional)
`-n none|basic|advanced`	Node collection (optional)
`-o xml|usertrace`	Output destination (optional)
`-r`	Reset archive data (optional)

Example: Turn off the collection of archive statistics for all message flows "MyFlow1" in execution group "EGRP2" for BrokerA.

```
mqsichangeflowstats BrokerA -a -e EGRP2 –j -c inactive
```

Figure 7-12. Message flow operation statistics commands (1 of 2) WM643 / VM6431.0

Notes:

Use the `mqsichangeflowstats` command to:

- Turn on or off accounting and statistics snapshot publication, or archive record output.
- Specify that the command is applied to a specific flow message flow, or all flows in an execution group, or all execution groups belonging to a broker.
- Modify the granularity of the data collected in addition to the standard message flow accounting and statistics. This extra data can include thread-related data, node-related data, node terminal-related data, or a mixture of this data
- Specify the output destination. User trace is the default output destination on all platforms.

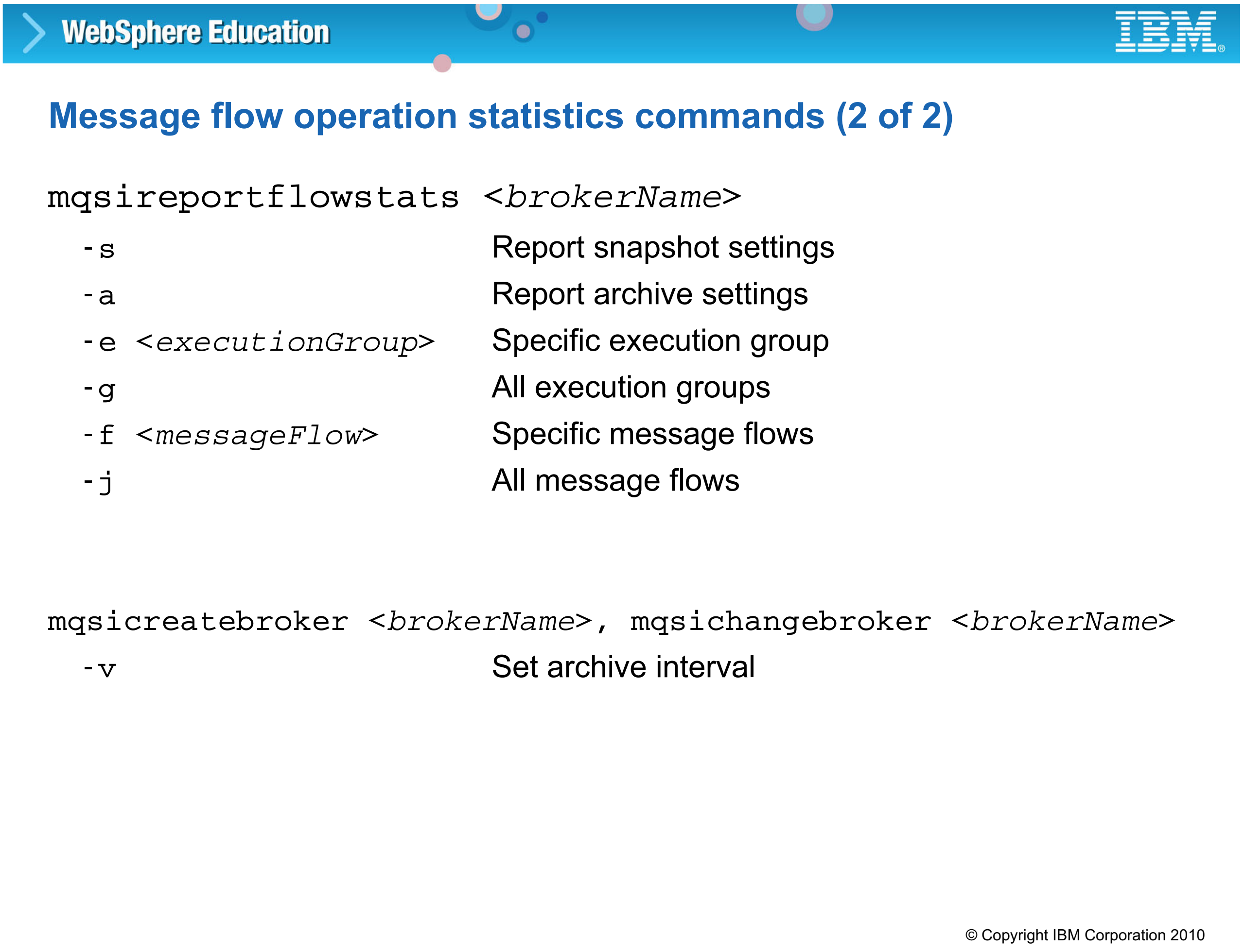
Instructor notes:

Purpose — Show the command used to activate statistical recording.

Details — At the end of the next unit, students record and inspect their own accounting and statistics data in an exercise.

Additional information — None.

Transition statement — There is also a command to show you the current options for accounting and statistics.



WebSphere Education
IBM

Message flow operation statistics commands (2 of 2)

```
mqsireportflowstats <brokerName>
  -s                     Report snapshot settings
  -a                     Report archive settings
  -e <executionGroup>    Specific execution group
  -g                     All execution groups
  -f <messageFlow>       Specific message flows
  -j                     All message flows


mqsicreatebroker <brokerName>, mqsichangebroker <brokerName>
  -v                     Set archive interval
```

Figure 7-13. Message flow operation statistics commands (2 of 2) WM643 / VM6431.0

Notes:

Use the `mqsireportflowstats` command to display the current options for accounting and statistics that have been set using the `mqsichangeflowstats` command. The mqsichangeflowstats has the following syntax requirements:

- You must specify at least one option for snapshot settings (`-a`, `-s` or both).
- You must either specify a specific execution group (`-e`) or all execution groups (`-g`).
- You must either specify a specific message flow (`-f`) or all message flows (`-j`).

When defining or modifying a broker using the command line, use the `-v` flag to specify the time interval (in minutes) at which statistics and accounting archive records are written. For internal accounting, the valid range is from 10 minutes to 14400 minutes; the default value is 60. An interval of zero minutes indicates that the operating system has an external method of notification (the ENF timer), and is not using an internal timer within WebSphere Message Broker.

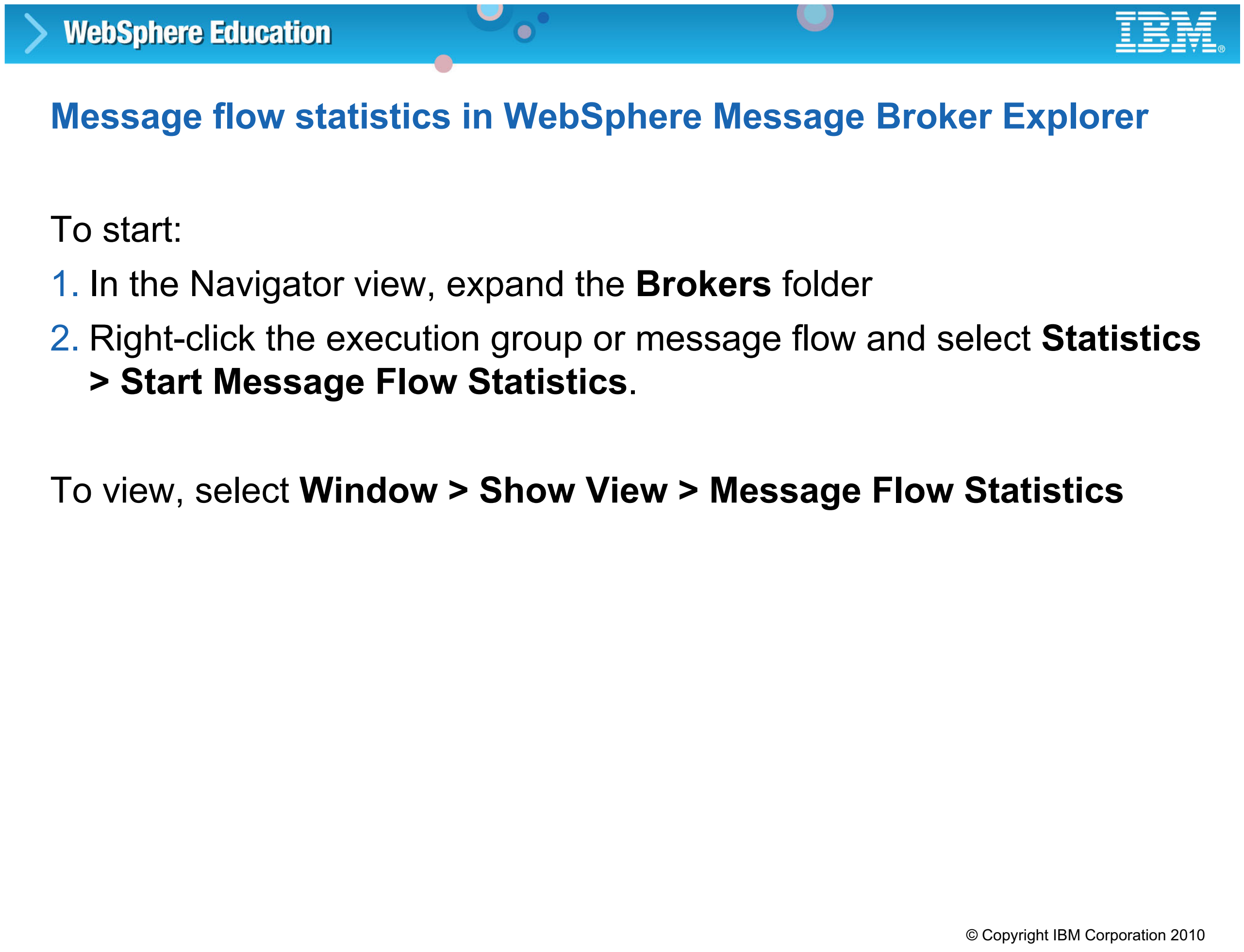
Instructor notes:

Purpose — Review the `mqsireportflowstats` command for reporting on current settings.

Details — On Windows, Linux, and UNIX systems, the user ID used to run this command must be a member of the `mqbrkrs` group.

Additional information — None.

Transition statement — You can display the statistics in WebSphere Message Broker Explorer.



WebSphere Education

IBM

Message flow statistics in WebSphere Message Broker Explorer

To start:

1. In the Navigator view, expand the **Brokers** folder
2. Right-click the execution group or message flow and select **Statistics > Start Message Flow Statistics**.

To view, select **Window > Show View > Message Flow Statistics**

Figure 7-14. Message flow statistics in WebSphere Message Broker Explorer WM643 / VM6431.0

Notes:

To collect statistics for all message flows on an execution group, right-click the execution group and select **Statistics** > **Start Message Flow Statistics**.

To collect statistics for a specific message flow, right-click the message flow and select **Statistics** > **Start Message Flow Statistics**.

To view the statistics, select **Window** > **Show View** > **Message Flow Statistics**.

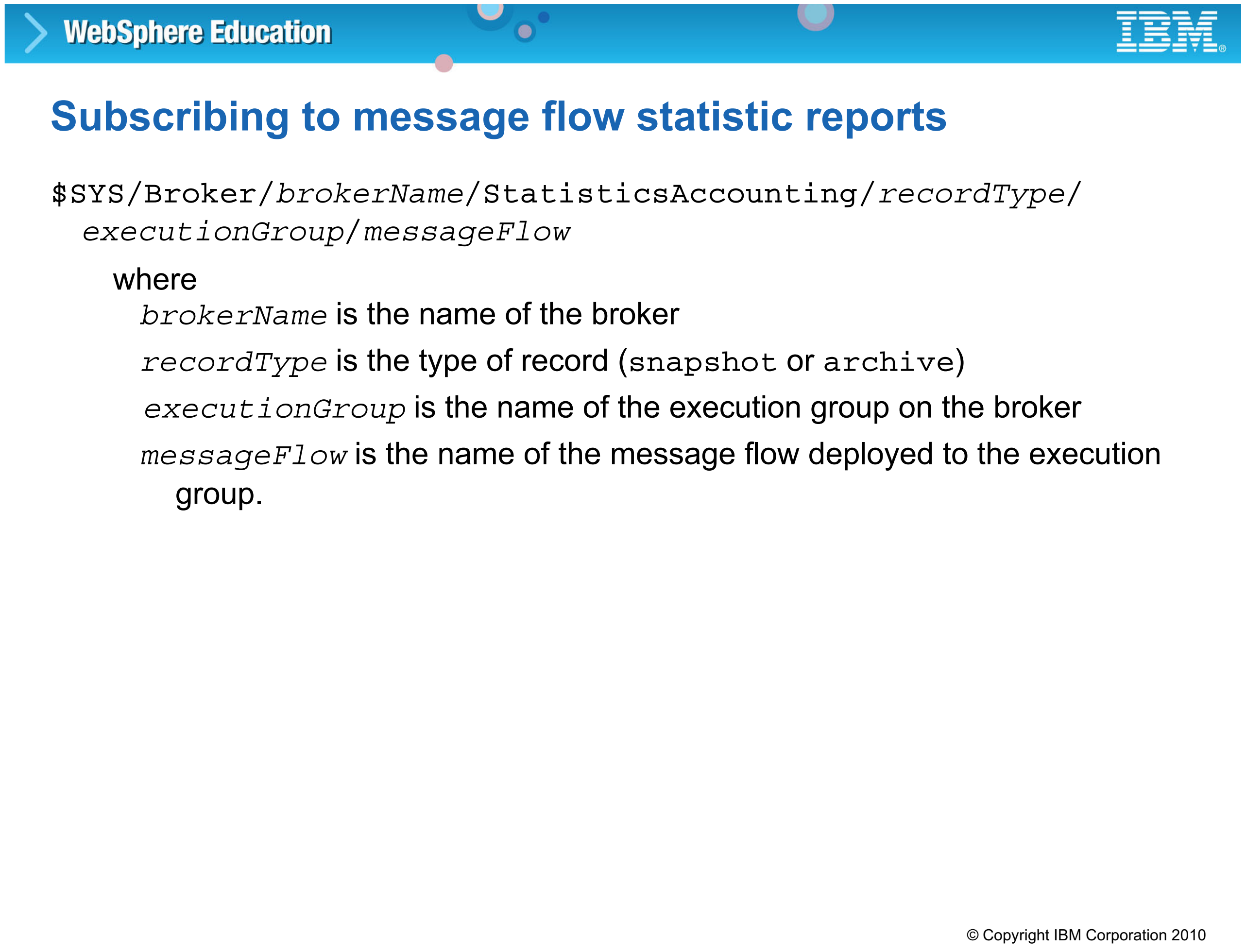
Instructor notes:

Purpose — Describe the steps for starting and viewing message flow statistics in WebSphere Message Broker Explorer.

Details — None.

Additional information — None.

Transition statement — An external application can access statistics by subscribing to the statistics report.



WebSphere Education

Subscribing to message flow statistic reports

```
$SYS/Broker/brokerName/StatisticsAccounting/recordType/
  executionGroup/messageFlow
```

where

brokerName is the name of the broker

recordType is the type of record (`snapshot` or `archive`)

executionGroup is the name of the execution group on the broker

messageFlow is the name of the message flow deployed to the execution group.

Figure 7-15. Subscribing to message flow statistic reports WM643 / VM6431.0

Notes:

You can subscribe to the message flow statistics. Use WebSphere MQ as a publication delivery mechanism to ensure that these messages are not lost, which would spell disaster if you are using them for charge-back purposes.

All publications are global, which means they can be collected anywhere in the broker domain. Furthermore, multiple subscribers are supported, meaning that the data can be collected by more than one analysis program.

Publication topics allow subscribers to retrieve a broad range of information.

The publication tree contains the broker name, execution group name, and message flow name, which allows subscribers anywhere in the broker domain to request a highly flexible range of information. This can vary from information about a specific flow running in a specific execution group, to all information about every flow running on the broker.

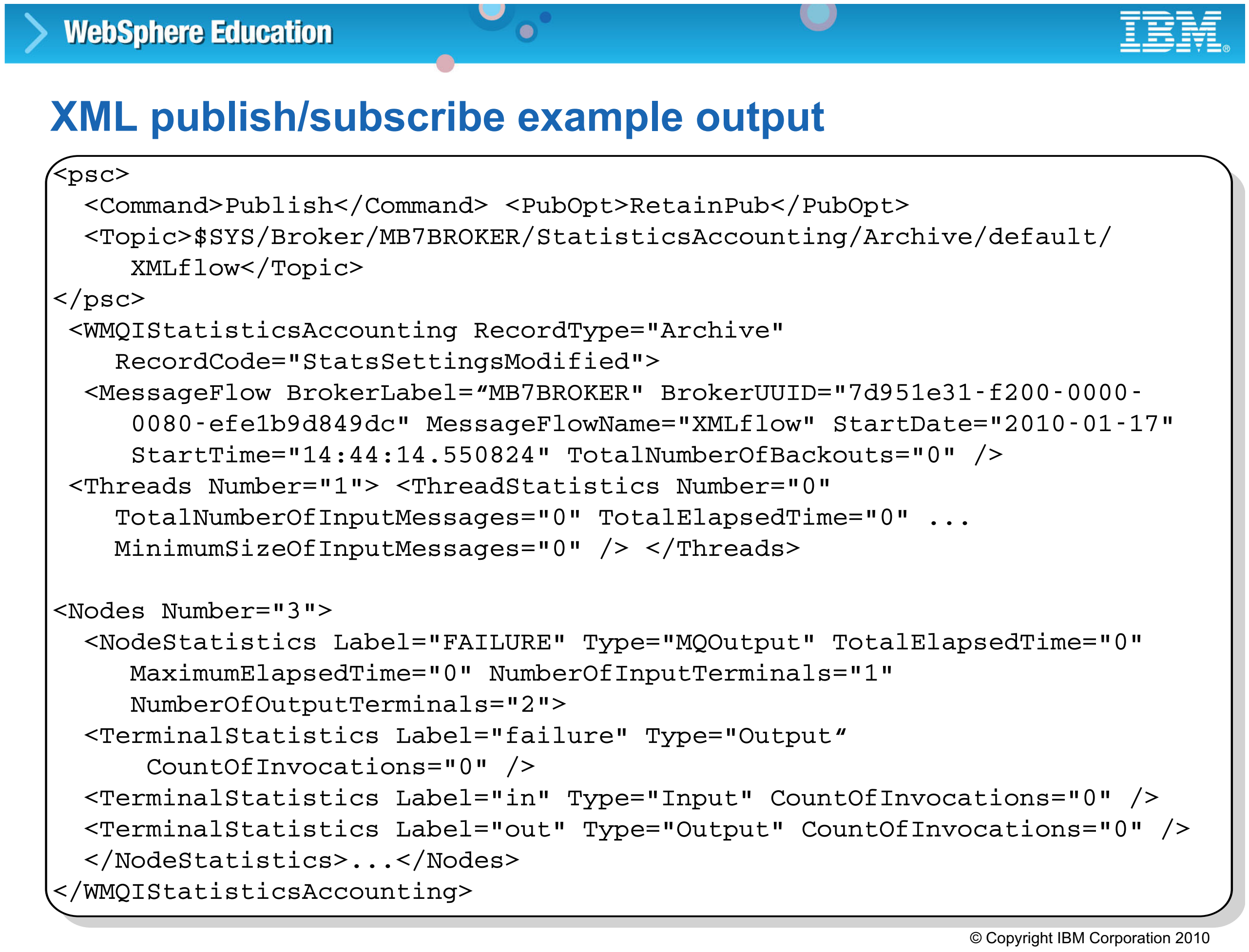
Instructor notes:

Purpose — Provide information about message flow statistics subscription.

Details — None.

Additional information — None.

Transition statement — The next slide shows a sample of the XML publication.



WebSphere Education

XML publish/subscribe example output

```
<psc>
  <Command>Publish</Command> <PubOpt>RetainPub</PubOpt>
  <Topic>$SYS/Broker/MB7BROKER/StatisticsAccounting/Archive/default/
     XMLflow</Topic>
</psc>
 <WMQIStatisticsAccounting RecordType="Archive"
    RecordCode="StatsSettingsModified">
  <MessageFlow BrokerLabel=“MB7BROKER" BrokerUUID="7d951e31-f200-0000-
     0080-efe1b9d849dc" MessageFlowName="XMLflow" StartDate="2010-01-17"
     StartTime="14:44:14.550824" TotalNumberOfBackouts="0" />
 <Threads Number="1"> <ThreadStatistics Number="0"
    TotalNumberOfInputMessages="0" TotalElapsedTime="0" ...
    MinimumSizeOfInputMessages="0" /> </Threads>

<Nodes Number="3">
  <NodeStatistics Label="FAILURE" Type="MQOutput" TotalElapsedTime="0"
     MaximumElapsedTime="0" NumberOfInputTerminals="1"
     NumberOfOutputTerminals="2">
  <TerminalStatistics Label="failure" Type="Output“
      CountOfInvocations="0" />
  <TerminalStatistics Label="in" Type="Input" CountOfInvocations="0" />
  <TerminalStatistics Label="out" Type="Output" CountOfInvocations="0" />
  </NodeStatistics>...</Nodes>
</WMQIStatisticsAccounting>
```

Figure 7-16. XML publish/subscribe example output WM643 / VM6431.0

Notes:

This figure shows an example of a simple statistics message published in XML. Note the five sections:

- Publish/subscribe elements in the MQRFH2 denoted by the `<psc>` tag.
- Message flow attributes denoted by the `<Message Flow>` tag.
- Thread attributes denoted by the `<Threads>` tag.
- Nodes attributes denoted by the `<Nodes>` tag. This folder can contain several terminal folders, if this level of granularity of reporting has been requested.

Both short-term snapshot and longer term archive data messages are published. Since the snapshot data is intended to be used for performance analysis (being sent to a real-time monitor, where, for example, it is graphically displayed), it is published as a retained and nonpersistent message. When archive data is used for accounting and long-term performance, an audit trail is more important, and therefore published persistently and retained. In this manner, late subscribers can get the most recent update.

Instructor notes:

Purpose — Close this topic with an example of actual accounting and statistics output.

Details — None.

Additional information — None.

Transition statement — WebSphere Message Broker can capture data from the message flow using event monitoring.

7.2. Event monitoring

Instructor topic introduction

What students will do — Learn how to create a monitoring profile to collect information from message flows.

How students will do it — Listen to lecture.

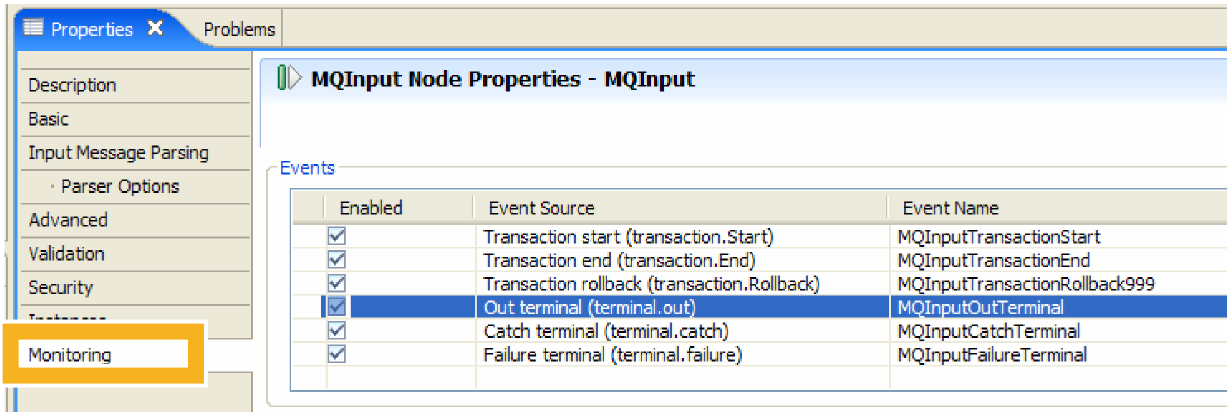
What students will learn — How to create a monitoring profile and use event monitoring to audit transactions.

How this will help students on their job — Administrators will know how to create and deploy an event monitoring profile.

WebSphere Education IBM

Event monitoring and auditing (1 of 2)

- Generate monitoring and audit events from message flow
  - Every Message Broker node includes a **Monitoring** tab to generate events
  - Transaction start, end, rollback issued from input nodes from any terminal on any node
  - Configure payload data, content style, identity, correlation, and sequencing data
  - Event filters to limit exact conditions for event creation; for example: `msg.Price>100`



Enabled	Event Source	Event Name
☑	Transaction start (transaction.Start)	MQInputTransactionStart
☑	Transaction end (transaction.End)	MQInputTransactionEnd
☑	Transaction rollback (transaction.Rollback)	MQInputTransactionRollback999
☑	Out terminal (terminal.out)	MQInputOutTerminal
☑	Catch terminal (terminal.catch)	MQInputCatchTerminal
☑	Failure terminal (terminal.failure)	MQInputFailureTerminal

Figure 7-17. Event monitoring and auditing (1 of 2) WM643 / VM6431.0

Notes:

Message flows can be configured to emit events. The events can be read and used by other applications for transaction monitoring, transaction auditing, and business process monitoring.

For example, it might be necessary to capture the first three fields of an input message at a certain point in the flow or to capture the entire payload of the output message. In a development environment, the monitoring requirements are defined in the WebSphere Message Broker Toolkit using the **Monitoring** tab on the node properties. You can use the **Monitoring** tab to declare the event you want to see from a node, such as an input event, by checking the box next to the event. In the example, the MQInput node has monitoring enabled at all points for a transaction: transaction start, transaction end, transaction rollback, out terminal, catch terminal, and failure terminal.

It is also possible to declare sequencing information, correlating information, what the payload looks like, certain fields, and other information.

Event filters can be defined to only generate events when a certain condition applies, such as when a field in the message exceeds a specified value.

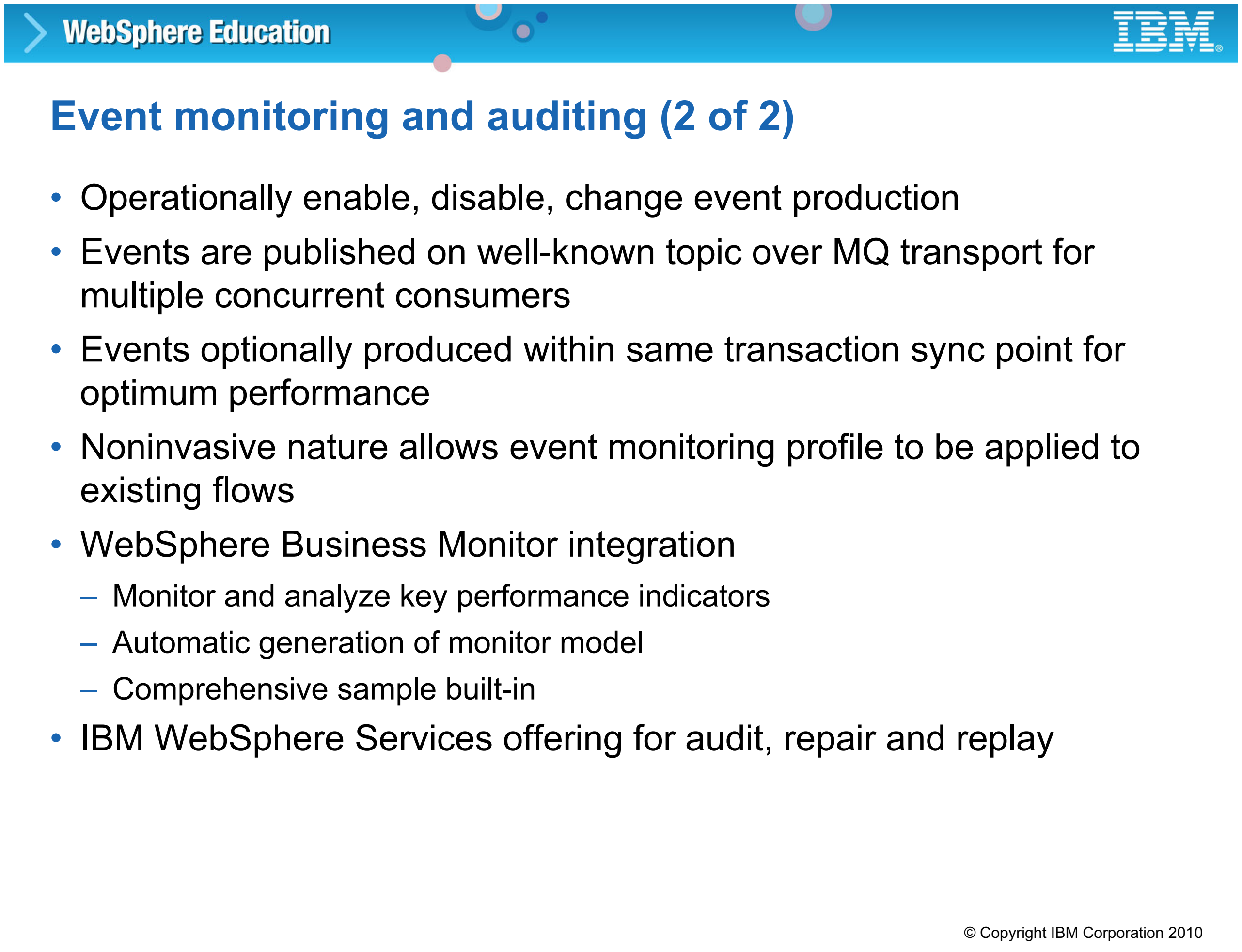
Instructor notes:

Purpose — Introduce event monitoring.

Details — Administrators do not necessarily need to know how to configure event monitoring but they might want to work with monitoring profiles.

Additional information — None.

Transition statement — None.



WebSphere Education

Event monitoring and auditing (2 of 2)

- Operationally enable, disable, change event production
- Events are published on well-known topic over MQ transport for multiple concurrent consumers
- Events optionally produced within same transaction sync point for optimum performance
- Noninvasive nature allows event monitoring profile to be applied to existing flows
- WebSphere Business Monitor integration
  - Monitor and analyze key performance indicators
  - Automatic generation of monitor model
  - Comprehensive sample built-in
- IBM WebSphere Services offering for audit, repair and replay

Figure 7-18. Event monitoring and auditing (2 of 2)

WM643 / VM6431.0

Notes:

Events are published over a publish/subscribe network as a well-defined topic. The topic is named with the name of broker, execution group, and the name given to the event as it is generated. It is also possible to generate an event at design-time or operationally at any point throughout the flow, containing the entire payload or any piece of it.

A message driven bean (MDB) captures the publication and pushes it into a common event infrastructure (CEI) for integration with WebSphere Business Monitor. WebSphere Business Monitor can be used to trend the data, for example looking at volume changes, or breaking out the information by geographical area.

An IBM Software Services offering is available for full auditing capability in the broker including the ability to repair and replay the flow.

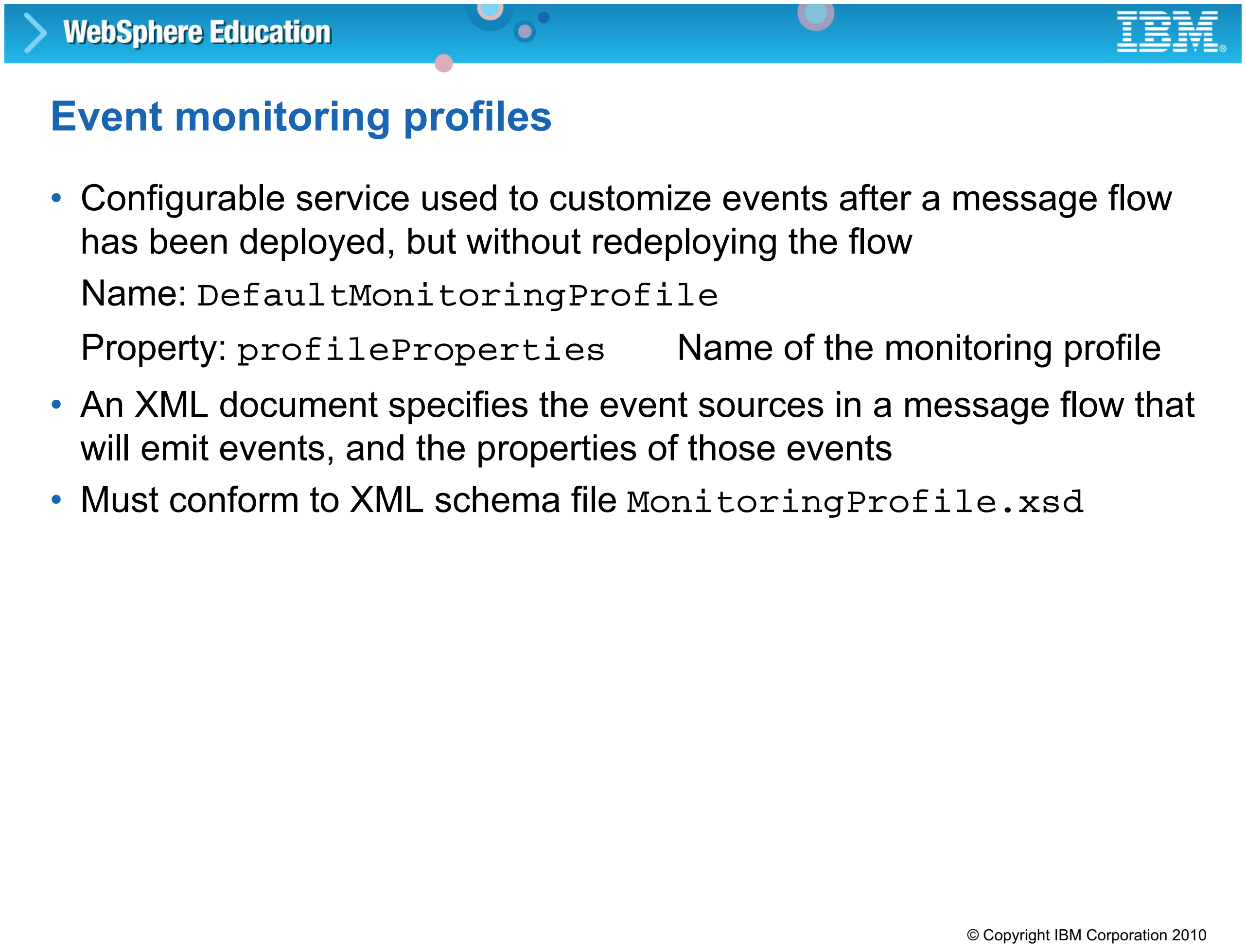
Instructor notes:

Purpose — Provide more information about event monitoring.

Details — None.

Additional information — None.

Transition statement — As an administrator, you might not have to configure event monitoring in a flow but you might have to work with monitoring profiles.



WebSphere Education

Event monitoring profiles

- Configurable service used to customize events after a message flow has been deployed, but without redeploying the flow

  Name: `DefaultMonitoringProfile`

  Property: `profileProperties`   Name of the monitoring profile
- An XML document specifies the event sources in a message flow that will emit events, and the properties of those events
- Must conform to XML schema file `MonitoringProfile.xsd`

Figure 7-19. Event monitoring profiles WM643 / VM6431.0

Notes:

A monitoring profile configurable service can be used to customize events after a message flow has been deployed, but without redeploying the flow. The configurable service has two parameters: the configurable service name and the name of the monitoring profile file.

The WebSphere Business Monitor sample provided with WebSphere Message Broker includes the monitoring profile schema file (`MonitoringProfile.xsd`).

Hint

If the deployed message flow has event monitoring properties configured using the WebSphere Message Broker Toolkit, use the `mqsireportflowmonitoring` command to create the equivalent monitoring profile XML file for the message flow. Use this profile as a starting point for creating other monitoring profiles.

Instructor notes:

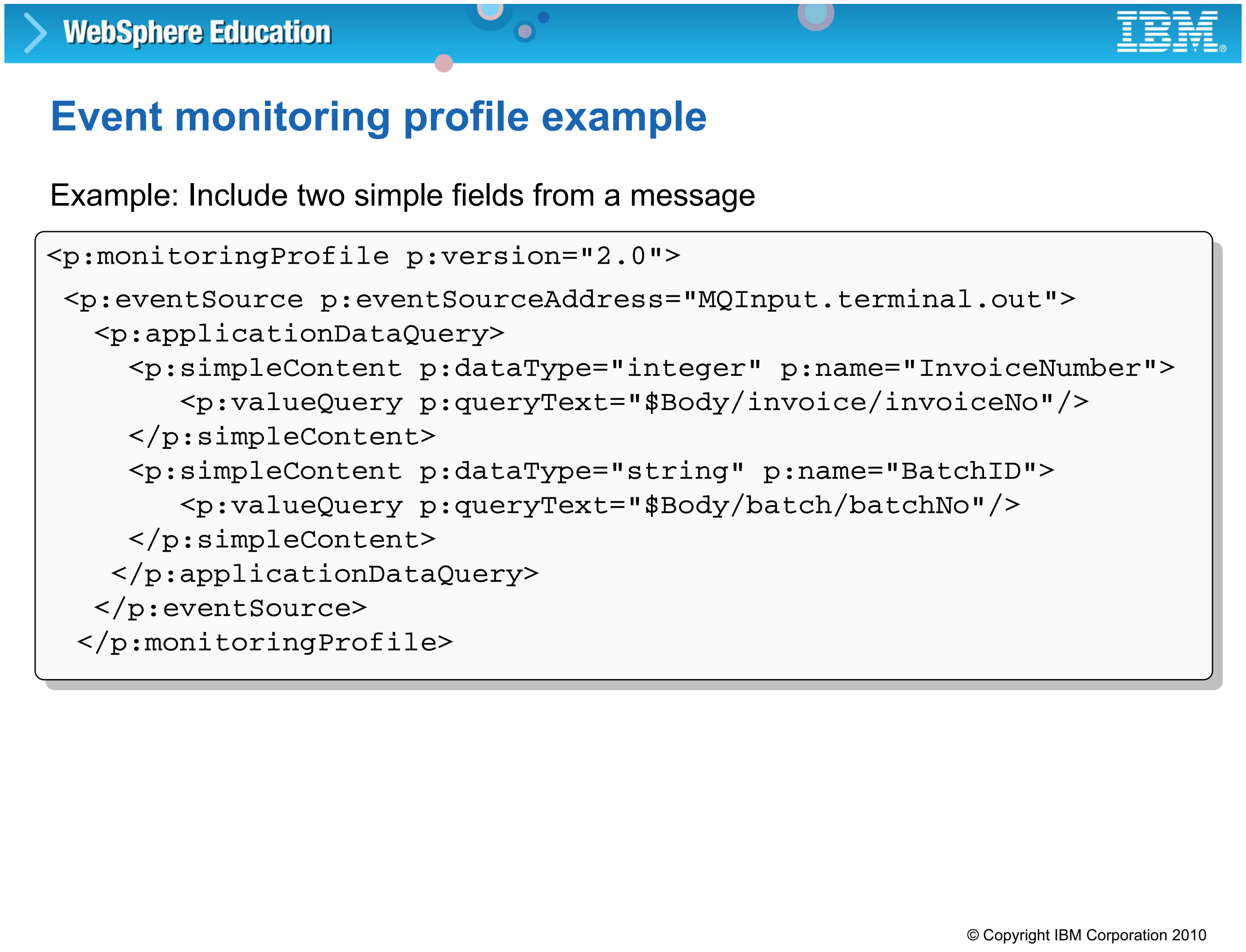
Purpose — Introduce the concept of a monitoring profile.

Details — It is best to start a new monitoring profile using the existing profile as starting point.

Open the Information Center and search on monitoring profile to show students the detailed information available on the schema.

Additional information — The Information Center has extensive information about the XML schema structure and elements.

Transition statement — An example of a monitoring profile is shown on the next slide.



Event monitoring profile example

Example: Include two simple fields from a message

```
<p:monitoringProfile p:version="2.0">
 <p:eventSource p:eventSourceAddress="MQInput.terminal.out">
   <p:applicationDataQuery>
     <p:simpleContent p:dataType="integer" p:name="InvoiceNumber">
        <p:valueQuery p:queryText="$Body/invoice/invoiceNo"/>
     </p:simpleContent>
     <p:simpleContent p:dataType="string" p:name="BatchID">
        <p:valueQuery p:queryText="$Body/batch/batchNo"/>
     </p:simpleContent>
    </p:applicationDataQuery>
   </p:eventSource>
  </p:monitoringProfile>
```

Figure 7-20. Event monitoring profile example

WM643 / VM6431.0

Notes:

The figure shows an example of an event monitoring profile.

The profile in the example captures the invoice number and batch ID from the message body. The message is captured at the MQInput node Out terminal (the event source).

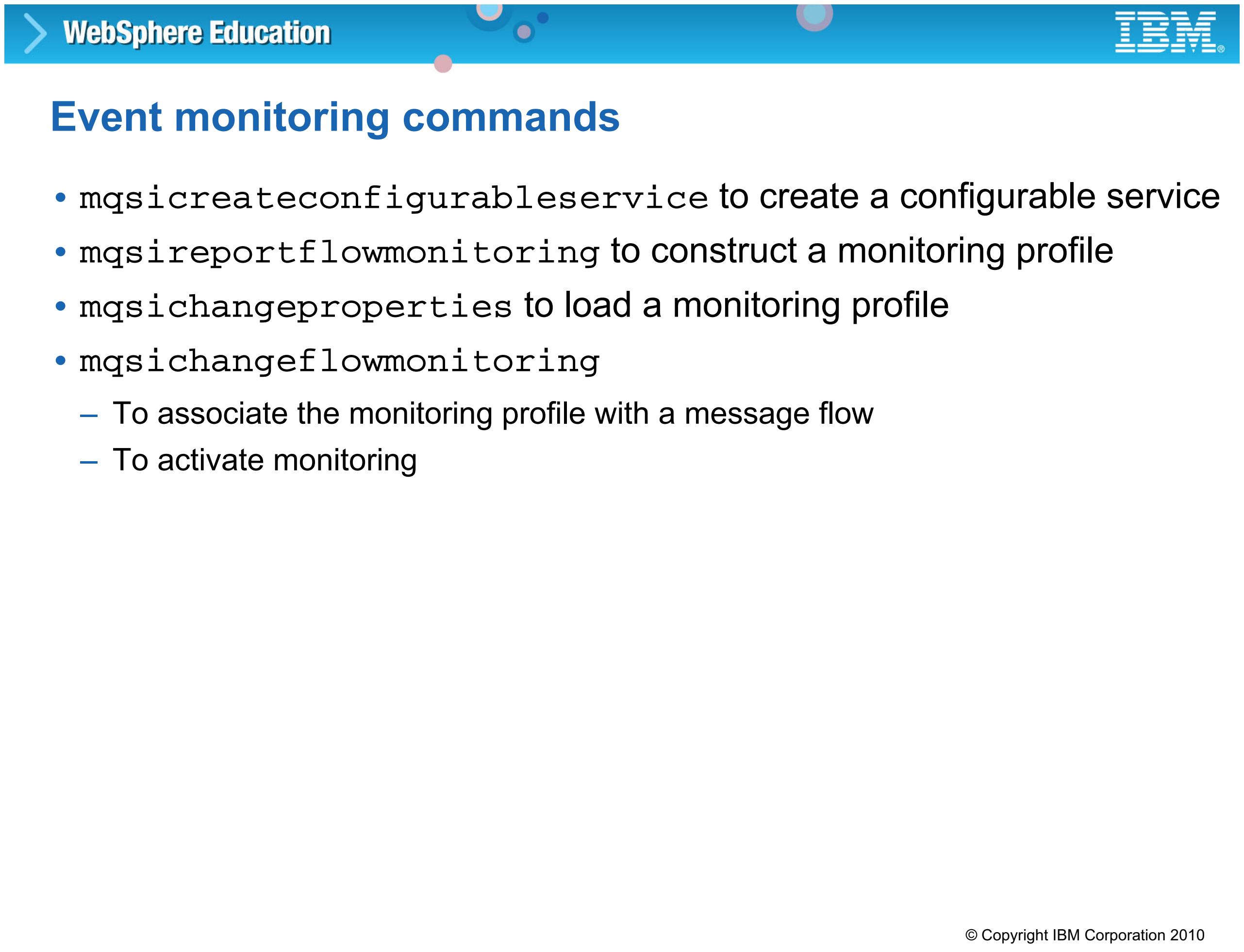
Instructor notes:

Purpose — Show an example of a simple monitoring profile.

Details — All students might not be familiar with XML schemas. Try not to spend too much time on the semantics. They should be able to recognize the elements, however, based on the tag names.

Additional information — None.

Transition statement — Event monitoring commands can be used to construct and load a monitoring profile.



WebSphere Education

Event monitoring commands

- `mqsicreateconfigurableservice` to create a configurable service
- `mqsireportflowmonitoring` to construct a monitoring profile
- `mqsichangeproperties` to load a monitoring profile
- `mqsichangeflowmonitoring`
  - To associate the monitoring profile with a message flow
  - To activate monitoring

Figure 7-21. Event monitoring commands WM643 / VM6431.0

Notes:

Event monitoring can be configured from a command line, without using the WebSphere Message Broker Toolkit.

The first step is to create a configurable service using the `mqsicreateconfigurableservice` command. Next, generate a base profile that you can use as a template using the `mqsireportflowmonitoring` command.

When the monitoring profile is ready, load it using the `mqsichangeproperties` command. Finally, associate the monitoring profile with a message flow using the `mqsichangeflowmonitoring` command.

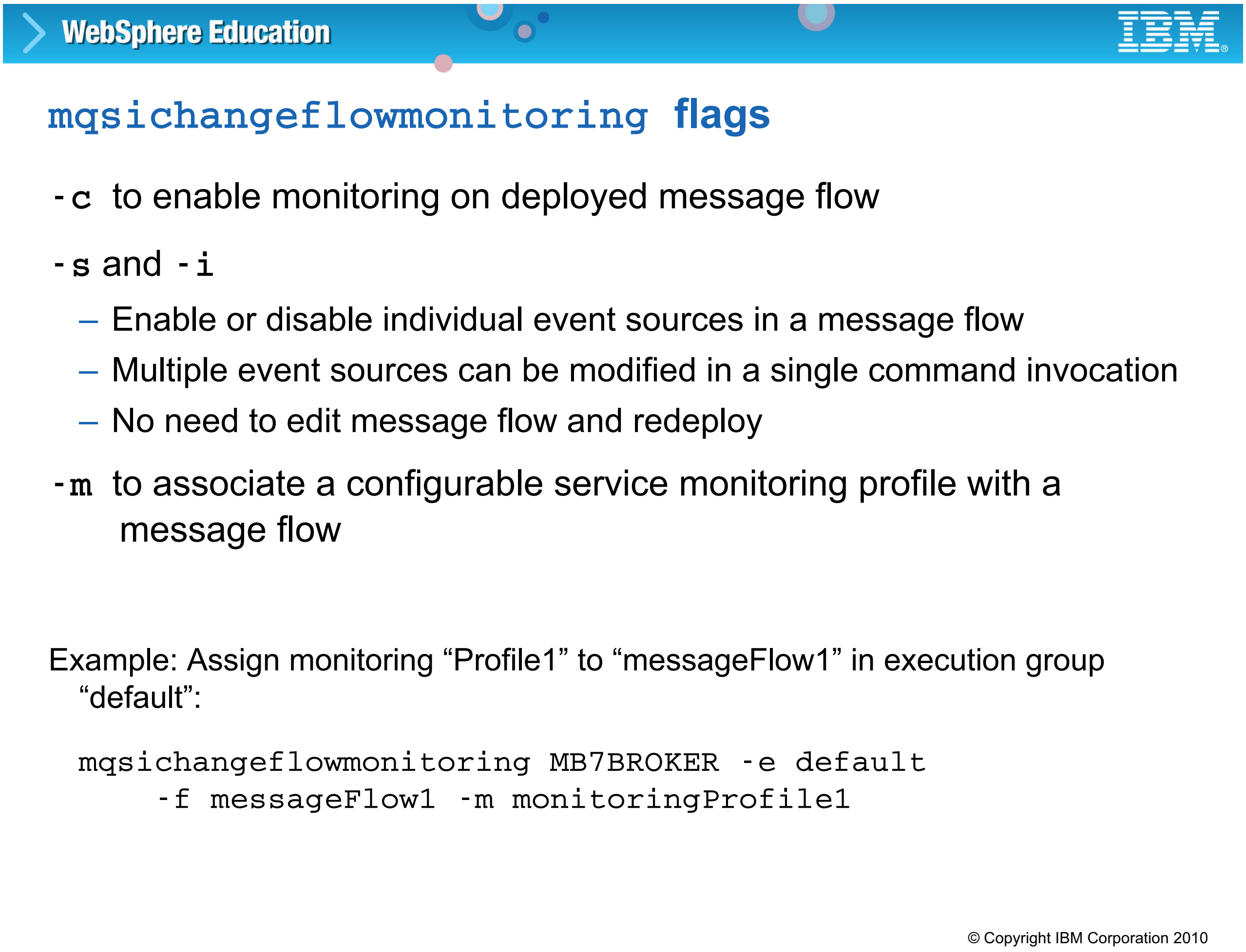
Instructor notes:

Purpose — Identify the commands that can be used to create and deploy a monitoring profile.

Details — None.

Additional information — None.

Transition statement — The next slides have more information about the commands.



WebSphere Education

`mqsichangeflowmonitoring` flags

`-c` to enable monitoring on deployed message flow

`-s` and `-i`

- Enable or disable individual event sources in a message flow
- Multiple event sources can be modified in a single command invocation
- No need to edit message flow and redeploy

`-m` to associate a configurable service monitoring profile with a message flow

Example: Assign monitoring “Profile1” to “messageFlow1” in execution group “default”:

```
mqsichangeflowmonitoring MB7BROKER -e default
    -f messageFlow1 -m monitoringProfile1
```

© Copyright IBM Corporation 2010

Figure 7-22. mqsichangeflowmonitoring flags WM643 / VM6431.0

Notes:

The change flow monitoring command identifies the flow to process using the broker name, execution group, and name of the flow.

Whether you are configuring monitoring using the WebSphere Message Broker Toolkit or the command line, you must activate monitoring using the `mqsichangeflowmonitoring` command with the `–c` flag. If you redeploy a flow, you must reactivate monitoring if you delete the existing flow, and then deploy.

You can use the `–m` flag to change the monitoring configuration without redeploying the message flow.

See the WebSphere Message Broker Information Center for the complete command syntax.

Instructor notes:

Purpose — Provide more information about the mqsichangeflowmonitoring command.

Details — You can also use this command to directly enable event sources:

```
mqsichangeflowmonitoring MB7BROKER -e default -f myMessageFlow

     -s "SOAP Input1.terminal.out,MQOutput1.terminal.in" -i enabled
```

`-i` specifies the string value that controls monitoring for the specified event source. Possible values are:

`enable` - enable monitoring for the specified event sources.

`disable` - disable monitoring for the specified event sources.

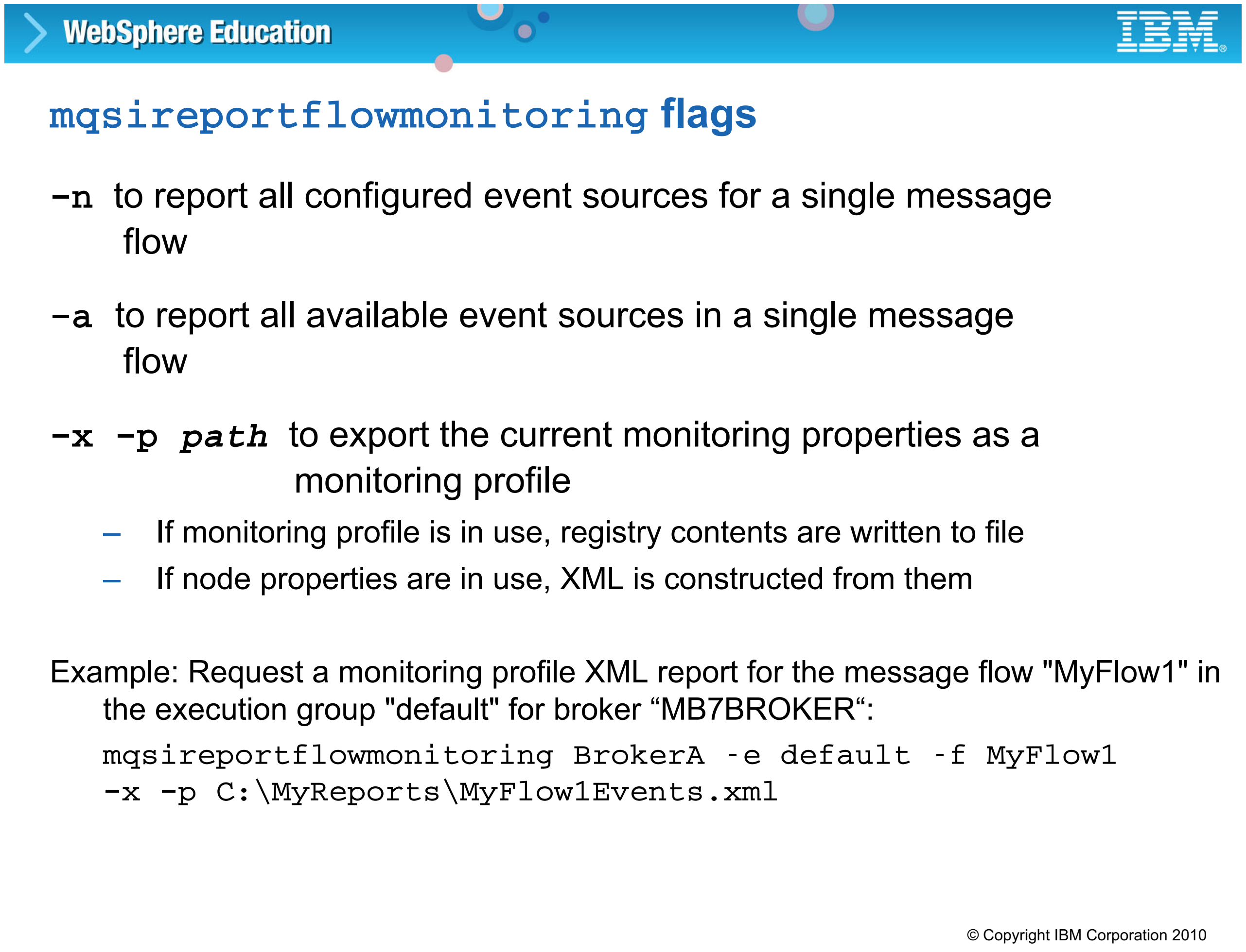
`-s` is a comma-separated list of the event sources to be enabled or disabled. This value takes the form <node name>.<event source>, where <event source> is one of the following values:

```
'terminal.<terminal name>'

'transaction.Start'

'transaction.End'

'transaction.Rollback'
```

If a message flow contains two or more nodes with identical names, the event sources on those nodes cannot be accurately addressed. If this is attempted, behavior is undefined.

Additional information — None.

Transition statement — None.



WebSphere Education

`mqsireportflowmonitoring` flags

`-n` to report all configured event sources for a single message flow

`-a` to report all available event sources in a single message flow

`-x -p` ***`path`*** to export the current monitoring properties as a monitoring profile

- If monitoring profile is in use, registry contents are written to file
- If node properties are in use, XML is constructed from them

Example: Request a monitoring profile XML report for the message flow "MyFlow1" in the execution group "default" for broker “MB7BROKER“:

```
mqsireportflowmonitoring BrokerA -e default -f MyFlow1
-x -p C:\MyReports\MyFlow1Events.xml
```

Figure 7-23. mqsireportflowmonitoring flags WM643 / VM6431.0

Notes:

The report flow monitoring command can be used to construct a monitoring profile, rather than handcrafting it in a schema editor.

- The `-n` flag is equivalent to selecting the message flow canvas.
- The `-a` flag is useful when you must discover the event source addresses of the available event sources.
- The `-x` and `-p` flags allows you to extract the monitoring profile to an XML file. If you extract the current configuration, you can modify it and reactivate monitoring using the change flow monitoring command.

See the WebSphere Message Broker Information Center for the complete command syntax.

Instructor notes:

Purpose — Identify `mqsireportflowmonitoring` commands specific to generating event reports and monitoring profiles.

Details — This same command can be used to generate the monitoring profile and to check that the event settings after a profile has been deployed.

Additional information — None.

Transition statement — Now that you have learned about the monitoring capabilities available in WebSphere Message Broker, the next topic describes some of the ways you can use this information to improve the performance of the broker and message flows.

7.3. Performance Tuning

Instructor topic introduction

*************************** **NOTE TO INSTRUCTORS:** *************************************

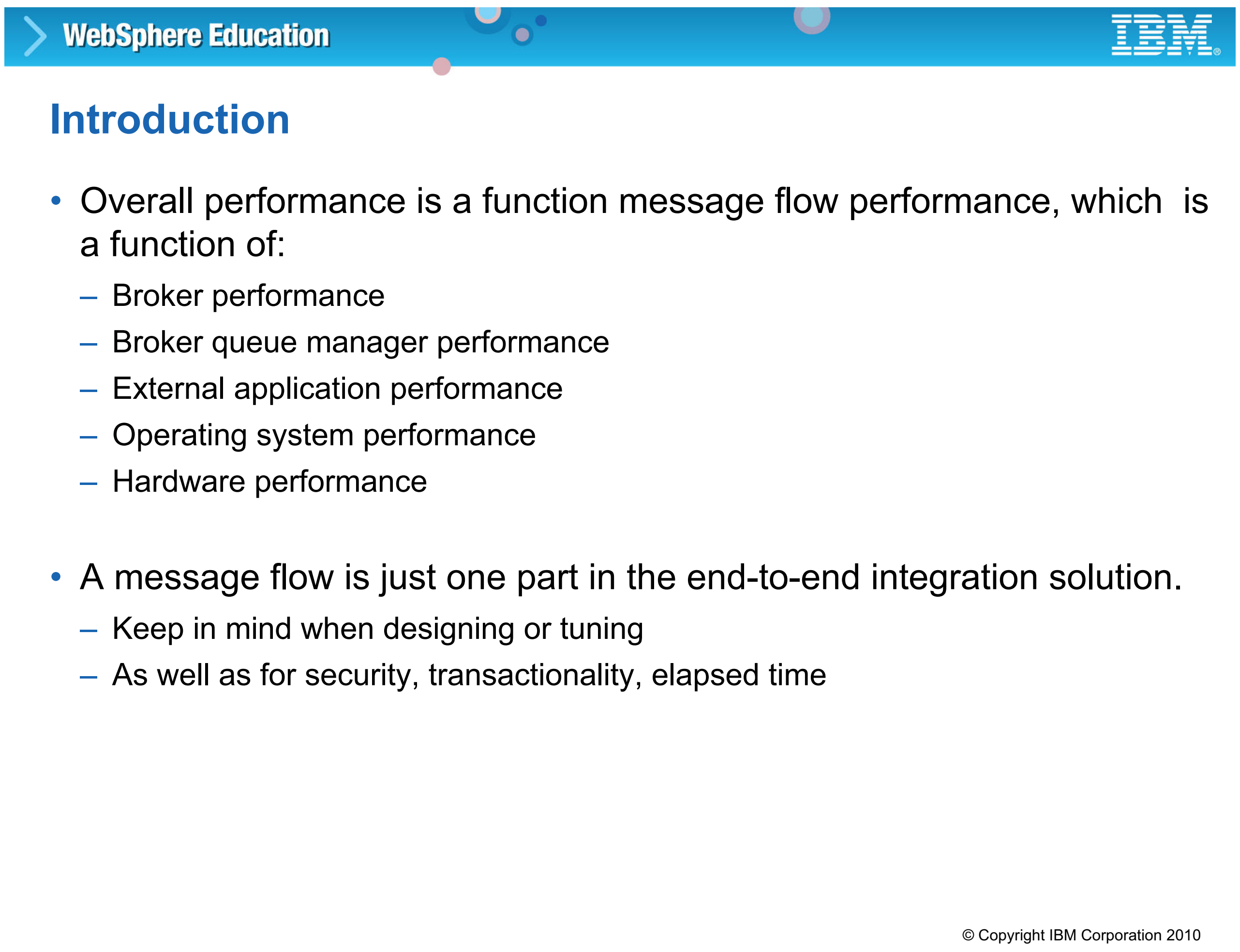
Performance statistics have not been included in the presentation material. Rather than focusing on specific performance statistics, you should concentrate on general principles of performance measurement, troubleshooting, and resolution.

What students will do — Listen to a lecture on performance influences and tuning possibilities of the WebSphere messaging products. Only the administrative perspective is covered here. A lot more could be said about design and programming, but this is covered in other courses.

How students will do it — Listen to the lecture and participate in classroom discussion.

What students will learn — Having recorded broker statistics, one use is to interpret to determine how well the broker is performing. Tuning options then take on a new meaning.

How this will help students in their job — Improving system efficiency is likely an entry on every system administrator’s job description. Knowing how to get the most out of the WebSphere messaging components will contribute to high marks in this category.



WebSphere Education

Introduction

- Overall performance is a function message flow performance, which is a function of:
  - Broker performance
  - Broker queue manager performance
  - External application performance
  - Operating system performance
  - Hardware performance

- A message flow is just one part in the end-to-end integration solution.
  - Keep in mind when designing or tuning
  - As well as for security, transactionality, elapsed time

Figure 7-24. Introduction WM643 / VM6431.0

Notes:

The message rate that is possible to achieve when running any given message flow is a function of the performance characteristics of a number of different but related items:

- Message flow: The way in which it is written.
- Broker: The way in which it is configured.
- Databases: Frequency of use by message flows and the type of operation.
- Broker queue manager: The efficiency of MQGET and MQPUT calls issued from message flows.
- Operating system: Affects overall performance.
- Hardware: The speed and number of processors, memory, number, speed, and frequency of access to disk storage, all contribute.

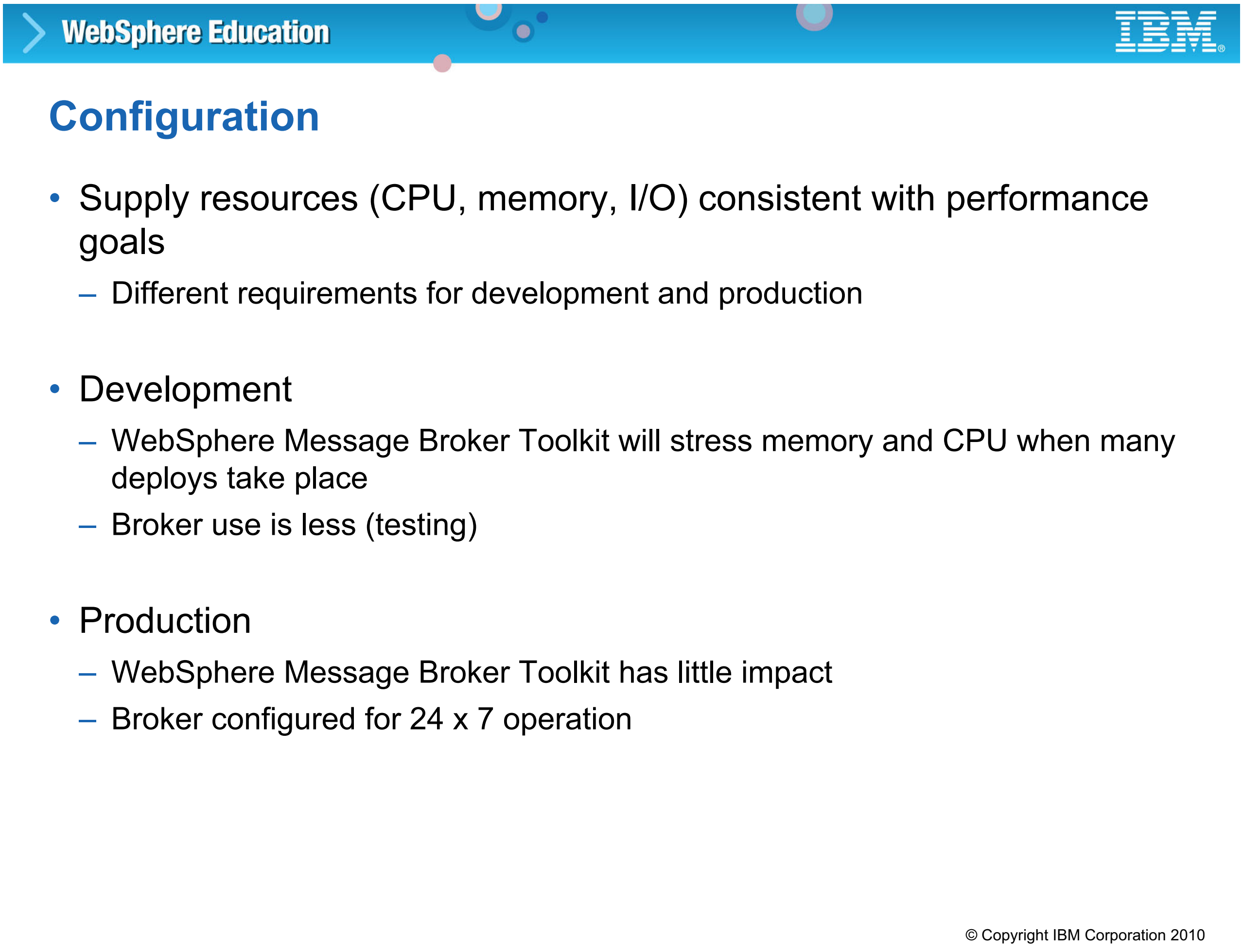
Instructor notes:

Purpose — Introduce performance variables.

Details — In trying to improve the rate at which messages can be processed by a message flow, it is important that you address each of the components listed. Changes to any one component have some effect on overall performance.

Additional information — None.

Transition statement — Look first at how the configuration of the components can affect performance.



WebSphere Education

Configuration

- Supply resources (CPU, memory, I/O) consistent with performance goals
  - Different requirements for development and production
- Development
  - WebSphere Message Broker Toolkit will stress memory and CPU when many deploys take place
  - Broker use is less (testing)
- Production
  - WebSphere Message Broker Toolkit has little impact
  - Broker configured for 24 x 7 operation

Figure 7-25. Configuration WM643 / VM6431.0

Notes:

In order to get the level of performance you expect, it is essential to have sufficient resources allocated. These include processor speed, the number of processors, memory, and the I/O subsystem.

It is also important to consider the difference in resource demands in a development environment running the WebSphere Message Broker Toolkit and a test broker as compared to a production environment requiring high throughput and running many flows on, potentially, many brokers and execution groups.

The key to good configuration is to match the capabilities of the equipment with the needs of the job. Too much is as bad as too little.

Instructor notes:

Purpose — Discuss the differences in configuration needs for development and production.

Details — There is often a tendency to concentrate on just the processors and memory and pay little attention to I/O. There are significant gains in performance to be had by using disks with a fast write nonvolatile cache for the queue manager log. This can lead to a three or four times increase in the persistent message rate that you are able to achieve.

Think about the different levels of resource that you are likely to need for developing message flows compared to running them in production.

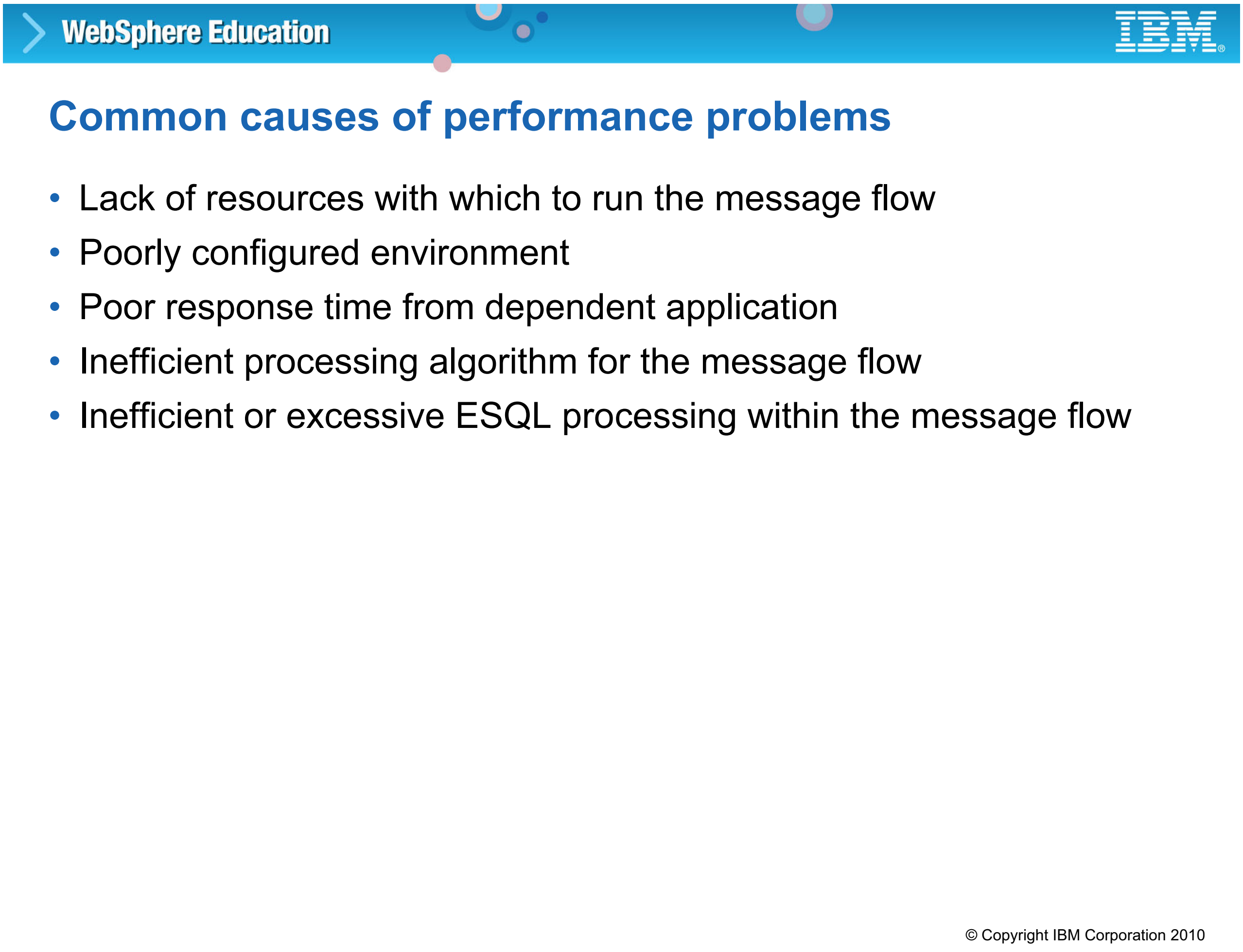
In the development environment, a user’s time is spent with the workbench, creating, and testing (meaning deploying) message flows. The volume of messages handled by the broker is low. Now the workbench has high processor and memory requirements. The broker has much lower requirements.

When running in a production environment (presumably 24 hours per day, 7 days per week), it is highly likely that the workbench has little use. The broker is the busy component as it continually processing messages. The focus has now shifted to the broker, which has high processor and memory requirements.

Given this change in emphasis as a project develops, you might want to change the allocation of your resources.

Additional information — None.

Transition statement — None.



WebSphere Education

Common causes of performance problems

- Lack of resources with which to run the message flow
- Poorly configured environment
- Poor response time from dependent application
- Inefficient processing algorithm for the message flow
- Inefficient or excessive ESQL processing within the message flow

Figure 7-26. Common causes of performance problems WM643 / VM6431.0

Notes:

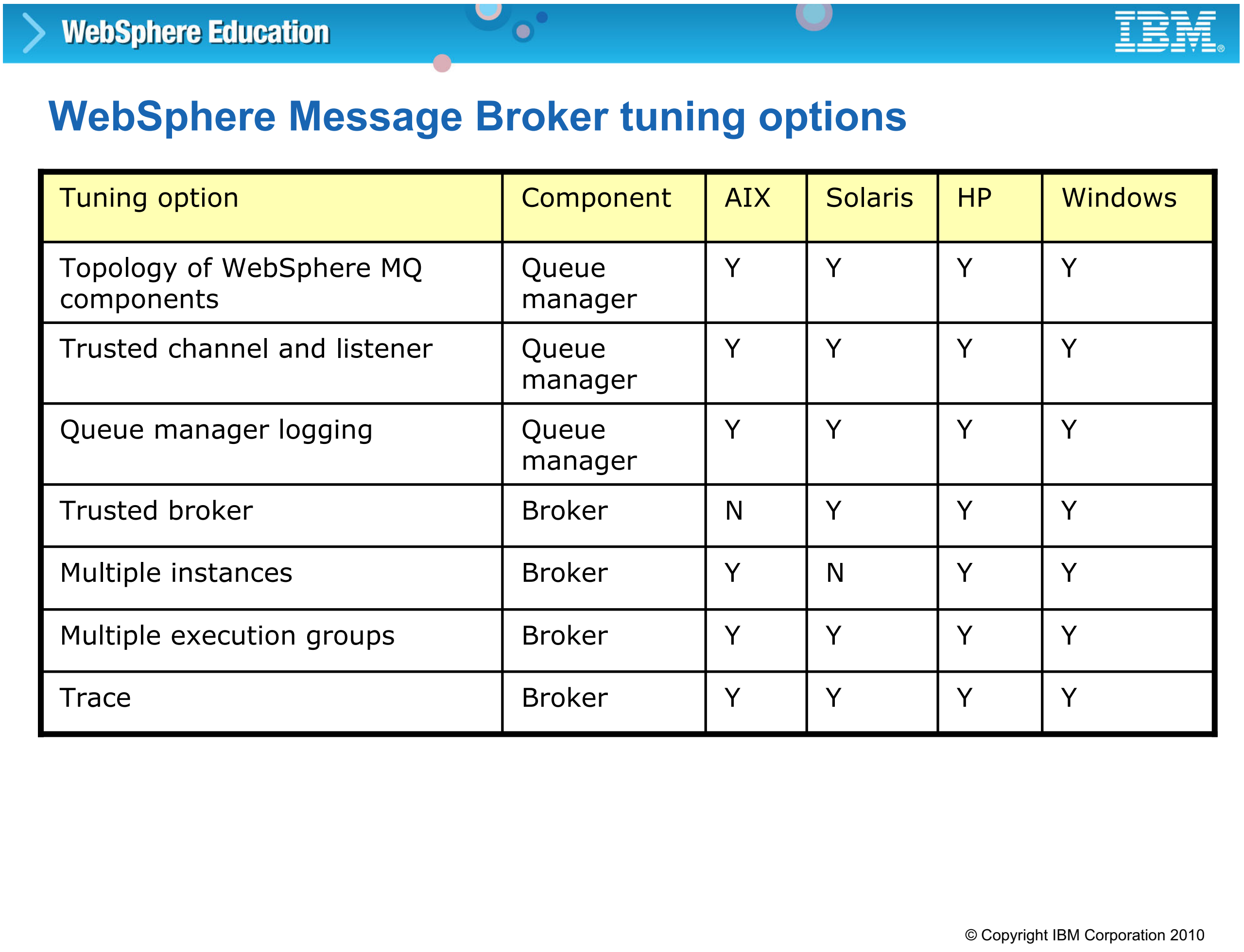
Instructor notes:

Purpose — List some of the common causes of performance problems.

Details — Some of these problems, such as resource allocation and environment are under the administrator’s control; other problems, such as inefficient ESQL are not.

Additional information — None.

Transition statement — There are quite a few broker tuning options available to improve performance.



WebSphere Education
IBM

WebSphere Message Broker tuning options

Tuning option	Component	AIX	Solaris	HP	Windows
Topology of WebSphere MQ components	Queue manager	Y	Y	Y	Y
Trusted channel and listener	Queue manager	Y	Y	Y	Y
Queue manager logging	Queue manager	Y	Y	Y	Y
Trusted broker	Broker	N	Y	Y	Y
Multiple instances	Broker	Y	N	Y	Y
Multiple execution groups	Broker	Y	Y	Y	Y
Trace	Broker	Y	Y	Y	Y

Figure 7-27. WebSphere Message Broker tuning options WM643 / VM6431.0

Notes:

On some platforms, it is possible to run a broker as a WebSphere MQ trusted application to streamline MQPUT and MQGET calls by the broker. Performance is improved because the broker interacts directly with the queue manager rather than an intermediary agent process. The increase in performance is at the cost of not isolating the queue manager from errant program behavior, such as poorly coded plug-in node, which can cause the broker queue manager to fail. A broker can be modified to run as a trusted WebSphere MQ application by using the `mqsichangebroker` command with the `-t` flag.

The benefit that is obtained from running a broker in trusted mode is dependent on the processing within the message flows. Different message flows running in the same broker might see different results. A message flow that has a high proportion of MQPUT or MQGET processing of nonpersistent messages will see more benefit. This processing is equivalent to a simple MQInput to MQOutput test.

Message flows in which most of the processing is concerned with manipulating the contents of a message will see a much lower benefit from using a trusted broker and might see no gain at all.

Instructor notes:

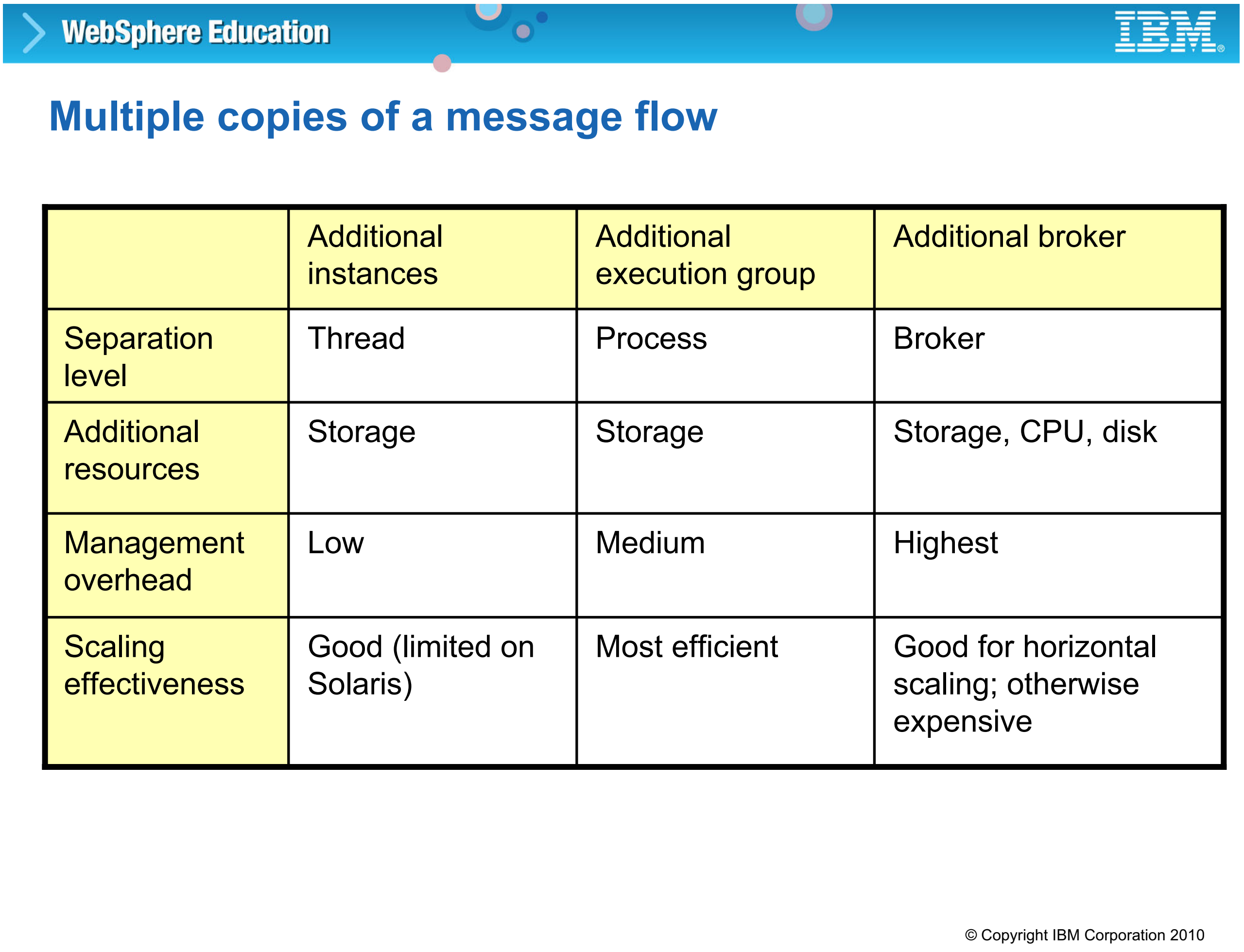
Purpose — Summarize the tuning options that are possible with message brokers.

Details — Tuning options cover the broker, its associated queue manager, and any databases containing business data across each of the platforms on which WebSphere Message Brokers are implemented.

In the figure, a value of **Y** indicates that the option applies or is available. It is not the default value but can be enabled. A value of **N** indicates the option is unavailable or is not recommended. You will look at each of these options in more detail later in the presentation.

Additional information — None.

Transition statement — What options exist to run multiple copies of a message flow?


WebSphere Education
IBM

Multiple copies of a message flow

	Additional instances	Additional execution group	Additional broker
Separation level	Thread	Process	Broker
Additional resources	Storage	Storage	Storage, CPU, disk
Management overhead	Low	Medium	Highest
Scaling effectiveness	Good (limited on Solaris)	Most efficient	Good for horizontal scaling; otherwise expensive

Figure 7-28. Multiple copies of a message flow

WM643 / VM6431.0

Notes:

An additional message flow instance (within an execution group) involves another thread running in that execution group process. In some situations this does not provide sufficient separation. Different execution groups provide separation at the process or address space level. This is enforced by the operating system. With multiple brokers it would be possible to a run common message flow on different servers. This would provide the maximum level of separation but is the most expensive in hardware resources.

In the figure, the **Management overhead** row indicates how much extra effort is involved with each of the techniques. The amount of work to add instances is low, since it is only the number of copies in the execution group which is changed. If there are many execution groups in existence, managing them in the WebSphere Message Broker Toolkit can become an issue.

Creating multiple brokers has the highest overhead. Each broker queue manager must be connected into the existing topology. As a new broker and broker queue manager have to be created, additional memory, processor, and disk space are required on the broker server.

When you need to add additional copies of a message flow, first use additional instances to increase message throughput. If adding more instances is not an option, try multiple execution groups to fully use the server on which the broker is running, or to use as much of it as possible. If further increases in message throughput are needed, use additional brokers. Establish the best use of additional instances and execution groups before replicating the broker.

When assigning message flows to execution groups, there might be business requirements which dictate that certain message flows should not run in the same execution group, such as flows for different companies when the broker is run by a service bureau. This might affect the policy you use to assign message flows to execution groups.

In some situations a message flow might have heavy resource requirements, such as the need for a large amount of processor cycles or memory. By assigning such flows to a restricted number of execution groups, the effects can be limited. This can be true where a message flow has a large high water mark for virtual memory use. If such a flow is assigned to all execution groups, the total virtual memory requirements are large, which places a greater demand on the memory of the server. This could lead to a noticeable increase in paging or swapping. If the message flow is only assigned to a limited number of execution groups, the effect is contained.

In order to determine how many copies of the message flow are required, you must know (or estimate) what the target message rate is and compare that to the rate achieved with one copy of the message flow. This also means that measurements are required.

Next you must establish an overall goal. Do you want to achieve a particular message rate, reach a particular processor utilization, or run in the most efficient way (although that might not lead to the maximum message rate)?

Run a single copy of the message flow with a realistic message. If the message flow processes multiple types of messages, make sure that all of the required message types are present and in their correct proportions. Continuously load the input queues to get the message flow running in a steady state. Measure the message throughput with tools, such as `vmstat`, to also determine the relative state of the system.

From these initial measurements it should be possible to see whether the message flow is processor bound or I/O bound, which affects how the message rate increases if additional copies of the message flow are deployed. It might also mean that you must change the hardware configuration in order to overcome an I/O constraint.

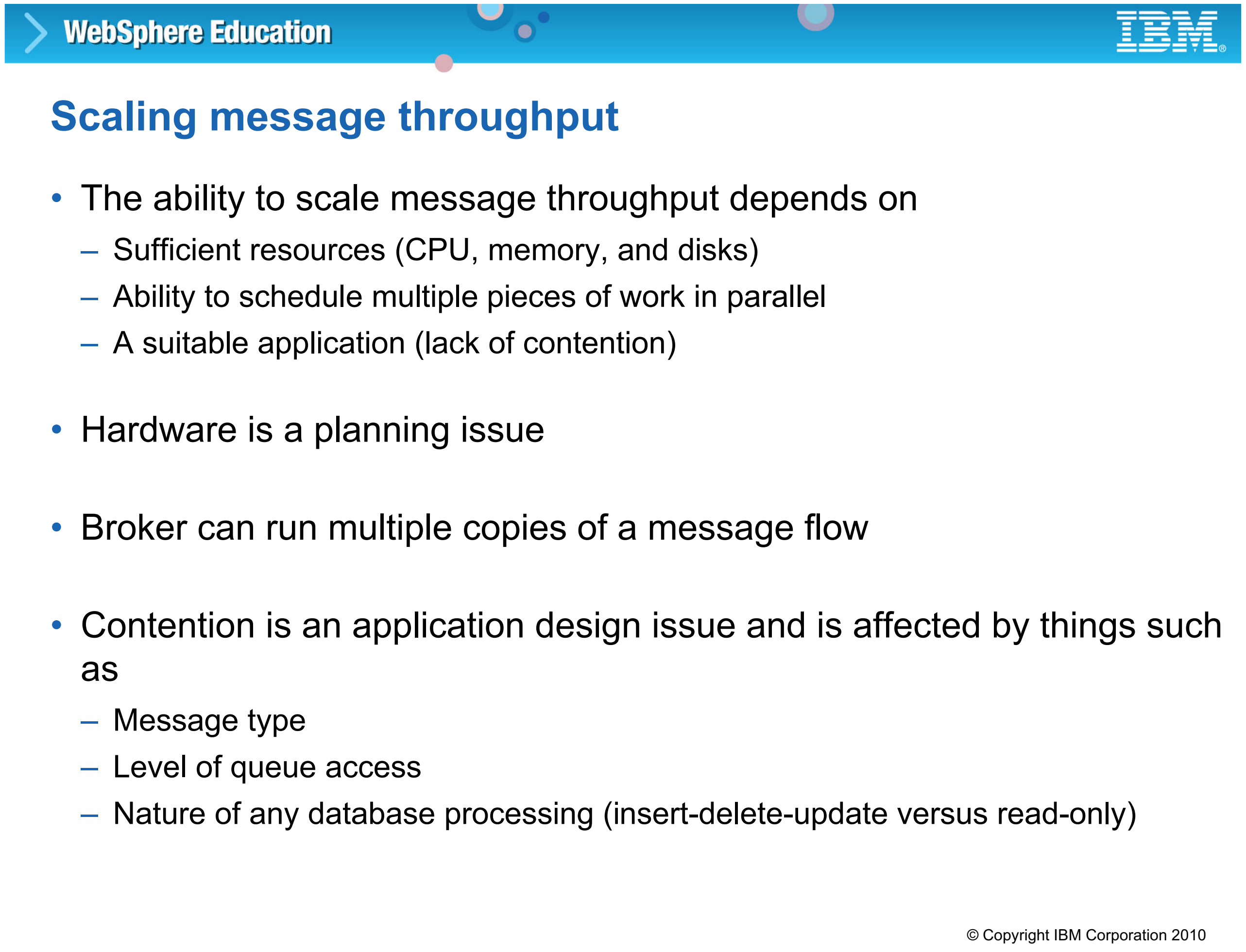
Instructor notes:

Purpose — Discuss how multiple copies of a message flow can affect message throughput rates.

Details — None.

Additional information — None.

Transition statement — When you try to scale message throughput by running multiple instances, the results can vary.



WebSphere Education

Scaling message throughput

- The ability to scale message throughput depends on
  - Sufficient resources (CPU, memory, and disks)
  - Ability to schedule multiple pieces of work in parallel
  - A suitable application (lack of contention)
- Hardware is a planning issue
- Broker can run multiple copies of a message flow
- Contention is an application design issue and is affected by things such as
  - Message type
  - Level of queue access
  - Nature of any database processing (insert-delete-update versus read-only)

Figure 7-29. Scaling message throughput WM643 / VM6431.0

Notes:

It is difficult to accurately predict the precise benefits that are obtained when running multiple copies of a message flow since it depends on the design of the message flow. However, there are characteristics that message flows must have in order to be able to scale well. These are:

- They should have low levels of I/O, either queue or database related. I/O normally imposes a lower maximum message rate than is the case when processing is processor bound.
- They should have low levels of queue access.
- They should have low levels of database insert, delete, and update activity. Read processing is far more scalable.

While the ideal is to produce message flows that give the level of required performance with little hardware tuning, additional benefit might be obtained by upgrading to faster processors or disks.

By adding additional copies of a message flow, your aim is to balance throughput and resources. It is often difficult to follow this ideal for a number of reasons. What can often happen is that you reach the limitation of a resource, perhaps the speed at which you can insert messages into a database. In such a case the improvement of additional copies reduces and you start to hit a plateau.

A more extreme case of reducing benefit is where you actually reduce overall throughput by adding more copies of the message flow. This would be the case where you started to experience deadlocks for key resources. The task in this case, is to identify the contention and eliminate it. At the least, reduce the number of copies running so that you get the maximum throughput.

The ability to increase message throughput in line with running additional copies of the message flow is an important consideration for many users. They must know that there are no inherent limitations which prevent them from growing vertically (within a server) and horizontally (across servers) as they require.

In order to be able to increase throughput with any piece of transaction or messaging software, a number of fundamentals must be in place.

- Physical resources must be in place. That is, processor, memory, and disks.
- There must be the ability to schedule concurrent pieces of work (multiprogramming).
- There must be minimum of contention within the application so that there is no conflict between running multiple copies concurrently. If each processing element needed access to a central control record, this would be a point of contention and so would limit throughput gains from running multiple copies.

Ensuring that sufficient hardware is available is really a case of recognizing what is required. In order to do this, you must implement the message flows, test, and measure to adequately determine the requirements.

Contention between message flows is not something you can predict. It depends on how the message flows are coded and the data that they reference. Items to be aware of are high levels of access to few queues and database processing. If messages are located over multiple input queues, there is less chance of contention than if a single queue is used for all message processing.

Message type is also a factor. Persistent messages are locked for a longer time than nonpersistent messages, as I/O takes place in the log. In addition, the maximum message rate that is achieved with persistent messages is lower than messages as there is an obvious I/O limitation.

Consider carefully any database processing. Avoid control records that are used by multiple message flows. Realize that insert, delete, and update processing is more intensive than read processing. With insert, delete, and update activity, changes must be logged.

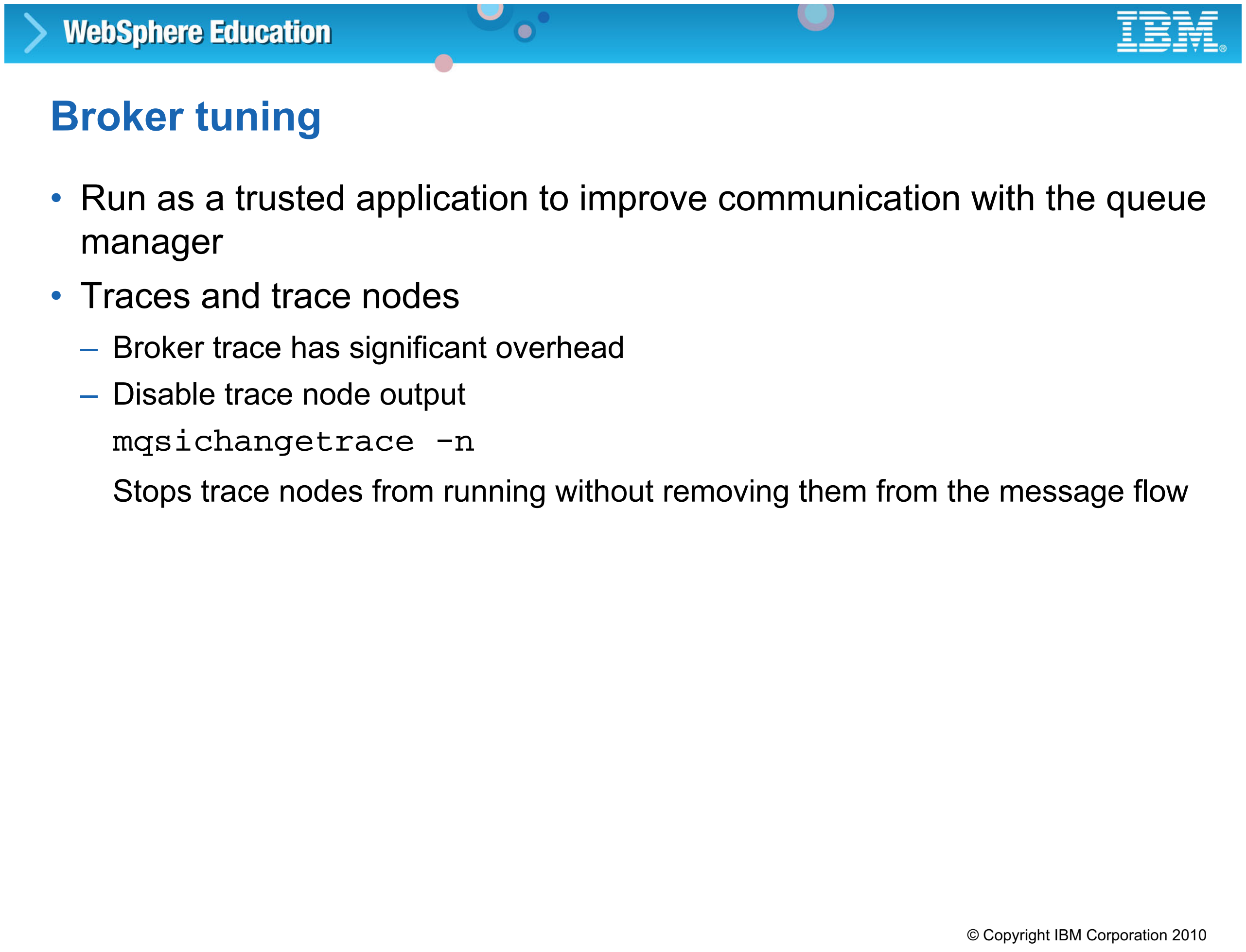
Instructor notes:

Purpose — Discuss the pluses and minuses associated with additional message flow instances.

Details — None.

Additional information — None.

Transition statement — What else influences scalability?



WebSphere Education

Broker tuning

- Run as a trusted application to improve communication with the queue manager
- Traces and trace nodes
  - Broker trace has significant overhead
  - Disable trace node output

    `mqsichangetrace -n`

    Stops trace nodes from running without removing them from the message flow

Figure 7-30. Broker tuning WM643 / VM6431.0

Notes:

Turn all tracing off within the broker. This includes user and service level trace.

Disable trace node output by using `mqsichangetrace`. This command allows you to suppress trace node output without having to delete the trace nodes from the message flow. There is essentially no overhead incurred by the trace nodes that have been deactivated.

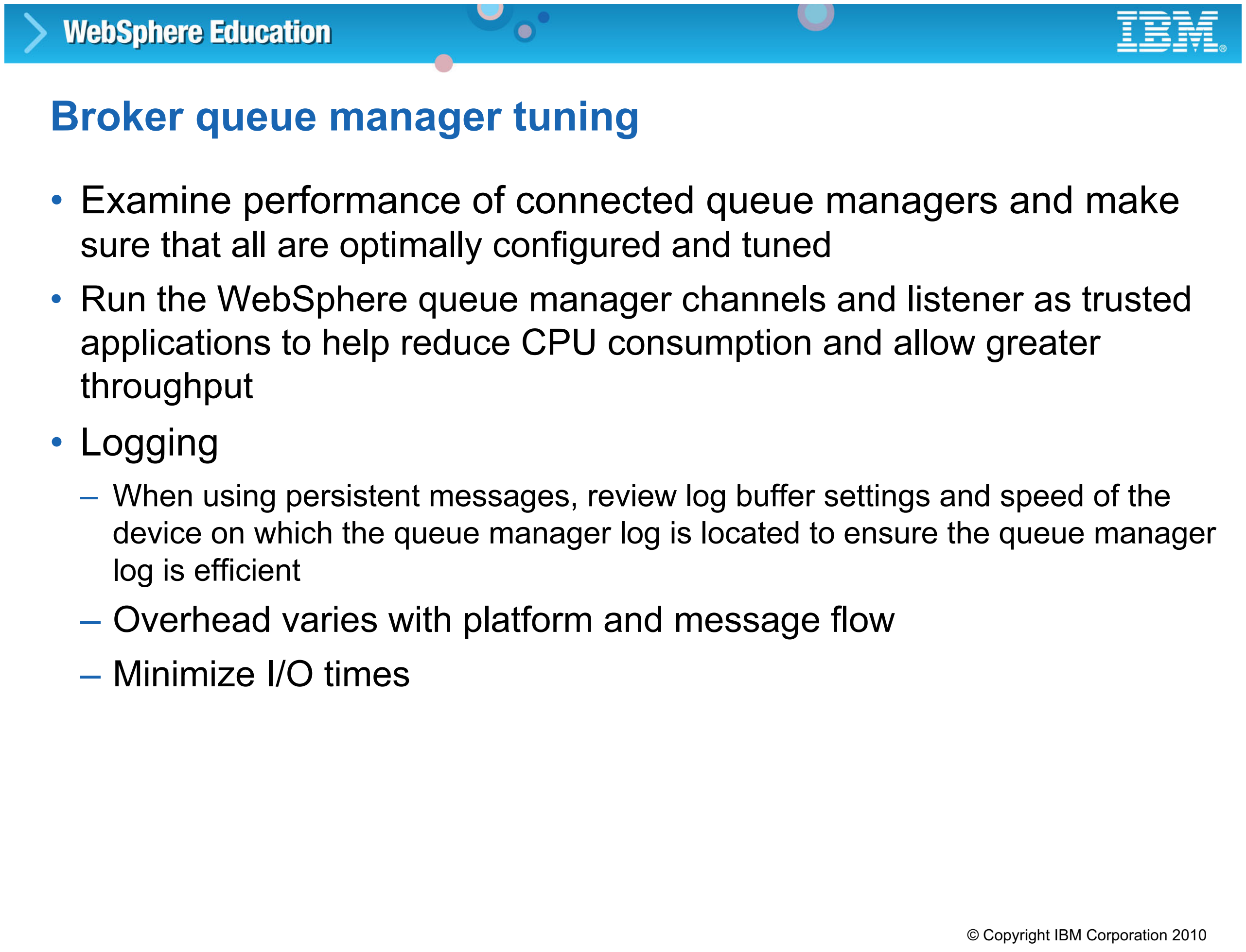
Instructor notes:

Purpose — Focus on some broker tuning parameters.

Details —

Additional information — At the end of the student notes are some suggestions about trace overhead and how to reduce it.

Transition statement — Do not forget the broker’s queue manager. WebSphere MQ has some tuning parameters to consider.



WebSphere Education

Broker queue manager tuning

- Examine performance of connected queue managers and make sure that all are optimally configured and tuned
- Run the WebSphere queue manager channels and listener as trusted applications to help reduce CPU consumption and allow greater throughput
- Logging
  - When using persistent messages, review log buffer settings and speed of the device on which the queue manager log is located to ensure the queue manager log is efficient
  - Overhead varies with platform and message flow
  - Minimize I/O times

Figure 7-31. Broker queue manager tuning WM643 / VM6431.0

Notes:

Where possible, ensure that the broker is located as close as possible to the messages to be processed. If output messages must be moved to a remote queue manager, processing cost increases for those messages. This is true if the messages are persistent. With persistent messages the performance of logging over multiple queue managers contributes to slower throughput. Each message must be logged multiple times as it passes from one queue manager to another.

On some platforms it is possible to run both the broker queue manager channels (MCA) and listener as trusted applications. This can be achieved by setting the MQIBindType to a value of `FASTPATH` in the channels stanza of the Windows registry, or `qm.ini` file of the (UNIX) queue managers. If the environment variable MQ_CONNECT_TYPE is present, it must be set to a value of `FASTPATH`. The use of trusted channels and listener is only recommended in a stable environment.

When persistent messages are used, the queue manager must successfully issue a synchronous write at commit time to the log before a unit of work is complete. This can significantly add to the overhead of message processing. This becomes apparent when you compare the rates at which persistent and nonpersistent messages can be processed.

When tuning logging parameters, also consider changing those related to log buffer and extent size. Large log extents mean less frequent switching between extents. The overhead resulting from the logging also varies with platform.

In a system in which queue managers are interconnected, it is important that both queue managers are configured for efficient logging, since the log on both queue managers is used as the messages move across the WebSphere MQ channel connecting the two queue managers.

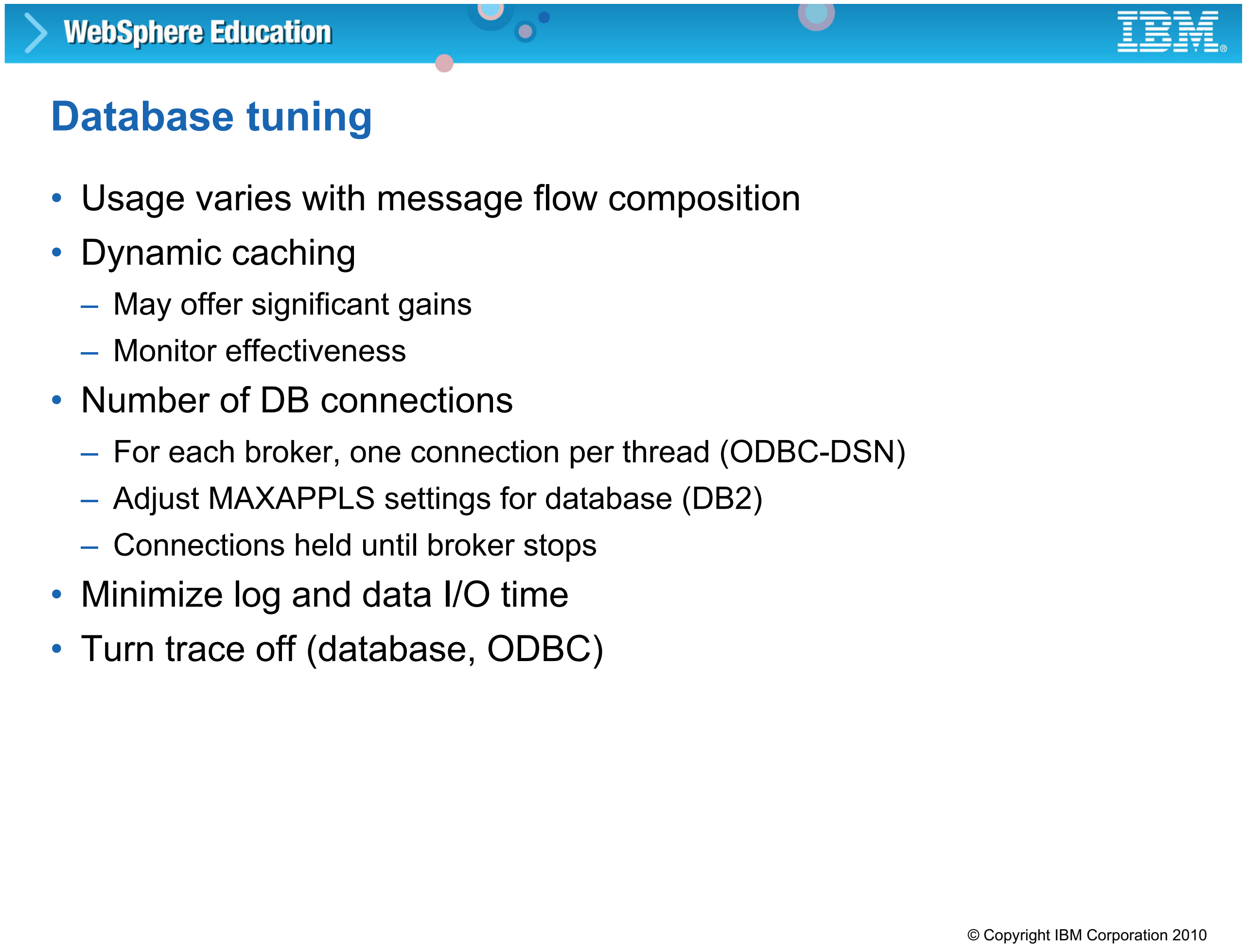
Instructor notes:

Purpose — Focus on some queue manager tuning options.

Details — None.

Additional information — None.

Transition statement — In some cases, database tuning can have a significant, positive effect.


WebSphere Education
IBM

Database tuning

- Usage varies with message flow composition
- Dynamic caching
  - May offer significant gains
  - Monitor effectiveness
- Number of DB connections
  - For each broker, one connection per thread (ODBC-DSN)
  - Adjust MAXAPPLS settings for database (DB2)
  - Connections held until broker stops
- Minimize log and data I/O time
- Turn trace off (database, ODBC)

Figure 7-32. Database tuning

Notes:

The extent to which a database containing business data is accessed in a message flow depends on the composition of the message flow. Obviously if there is no access, no tuning is required. Usage of business data can vary from read-only activity through update-insert-delete. The type of processing that takes place determines where you must place your tuning effort.

Ensure that both database and ODBC trace are turned off. ODBC tracing is controlled differently on each of the different platform types.

Where message flows frequently access a database, a significant performance benefit can be obtained by caching the dynamic SQL, which is generated when an access occurs. Without such caching, each data access must be compiled using an `SQL PREPARE` statement and then run using an `SQL EXECUTE` statement. The compilation is relatively expensive. The aim is to significantly reduce the number of times it takes place.

Ensure that each database is configured with sufficient connections. The connection from the message flow to the database manager is made on the first database request in the flow. Depending on the number of connections defined within the database manager and the number of instances of the message flow, not all requests can be connected, which obviously reduces the throughput potential.

Examine the allocation of data and index buffers, and ensure that there are enough to meet your needs. Also verify that good buffer hit ratios are obtained.

Consider creating a read-only view of those tables, which are rarely, if ever, updated. This reduces locking by the database manager and leads to better read performance.

When database insert, delete, and update activity takes place, a synchronous write to the database manager log is required. This is regardless of whether the WebSphere MQ message is persistent or not. In order to ensure maximum throughput, it is important to ensure that the database log is located on a fast device.

A BLOB written to a database is an immediate write to disk. It is not buffered like other data. If BLOB I/O is frequent, it is beneficial to locate the data portion of the database on a fast device.

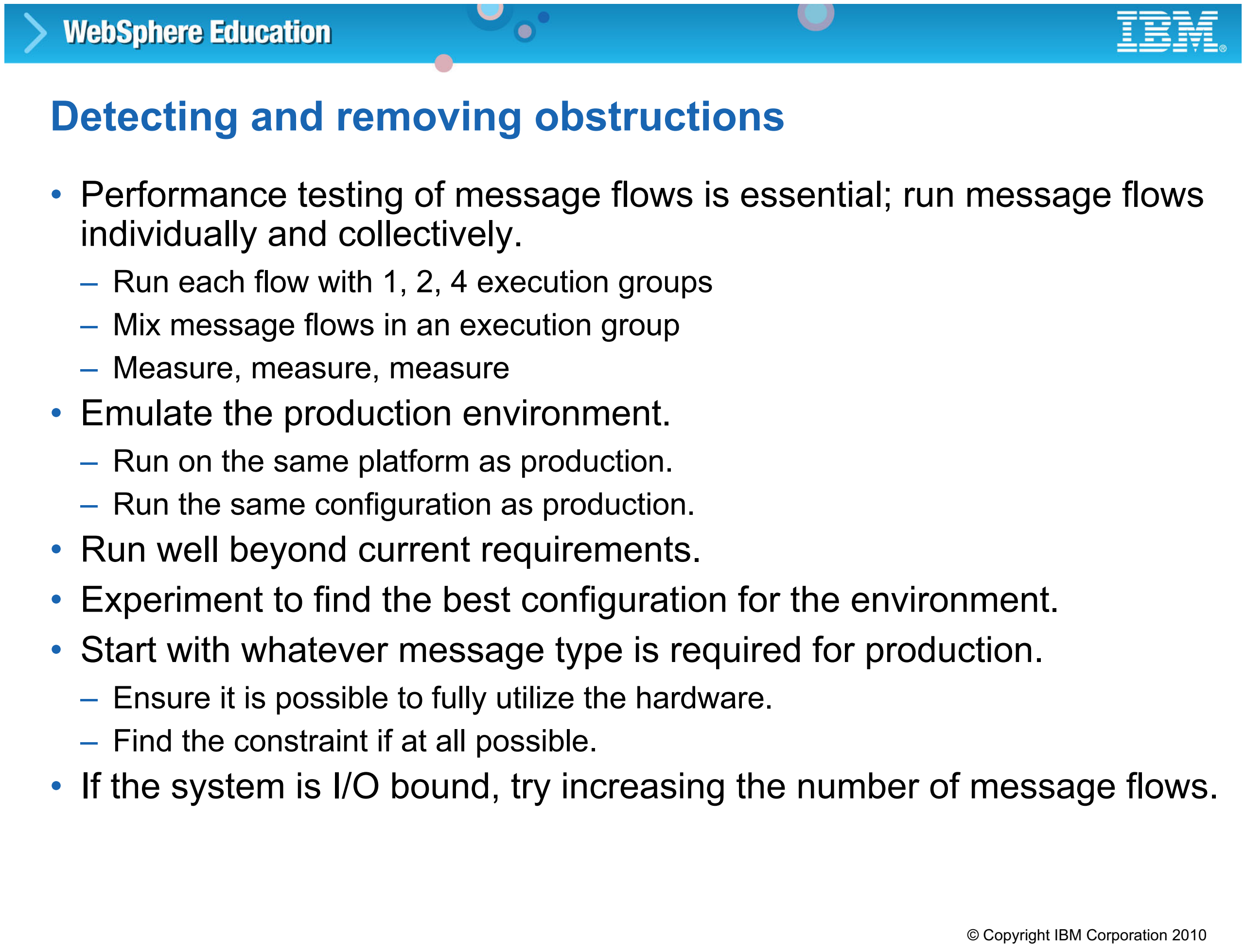
Instructor notes:

Purpose — Identify some database tuning options.

Details —

Additional information — None.

Transition statement — The message flow transaction mode can also affect performance.



WebSphere Education

Detecting and removing obstructions

- Performance testing of message flows is essential; run message flows individually and collectively.
  - Run each flow with 1, 2, 4 execution groups
  - Mix message flows in an execution group
  - Measure, measure, measure
- Emulate the production environment.
  - Run on the same platform as production.
  - Run the same configuration as production.
- Run well beyond current requirements.
- Experiment to find the best configuration for the environment.
- Start with whatever message type is required for production.
  - Ensure it is possible to fully utilize the hardware.
  - Find the constraint if at all possible.
- If the system is I/O bound, try increasing the number of message flows.

Figure 7-33. Detecting and removing obstructions WM643 / VM6431.0

Notes:

It is essential to undertake performance testing of message flows before production. You might observe that all is well when running few copies of a message flow but that things change when more copies are running. It is essential to find this out before being in production.

First, determine the effects of running an increasing number of copies of each message flow. Run message flows in isolation, and in combination, in order to see if there are any adverse effects.

Be sure to conduct this testing in the same environment as production, ideally using the same

Test with nonpersistent messages at first, even if you intend to use persistent messages in production. Using nonpersistent messages remove the limit (I/O impact) that is imposed by the use of persistent messages. Such testing can make a constraint apparent which would not have surfaced otherwise.

In performance testing it is important to reflect the production environment as fully as possible. In most cases it is typical to have a continual flow of messages through the system, rather than processing messages in batches. What often happens in testing is that many messages are loaded onto a queue and then processed. This approach produces different results from a continual topping up of the input queues.

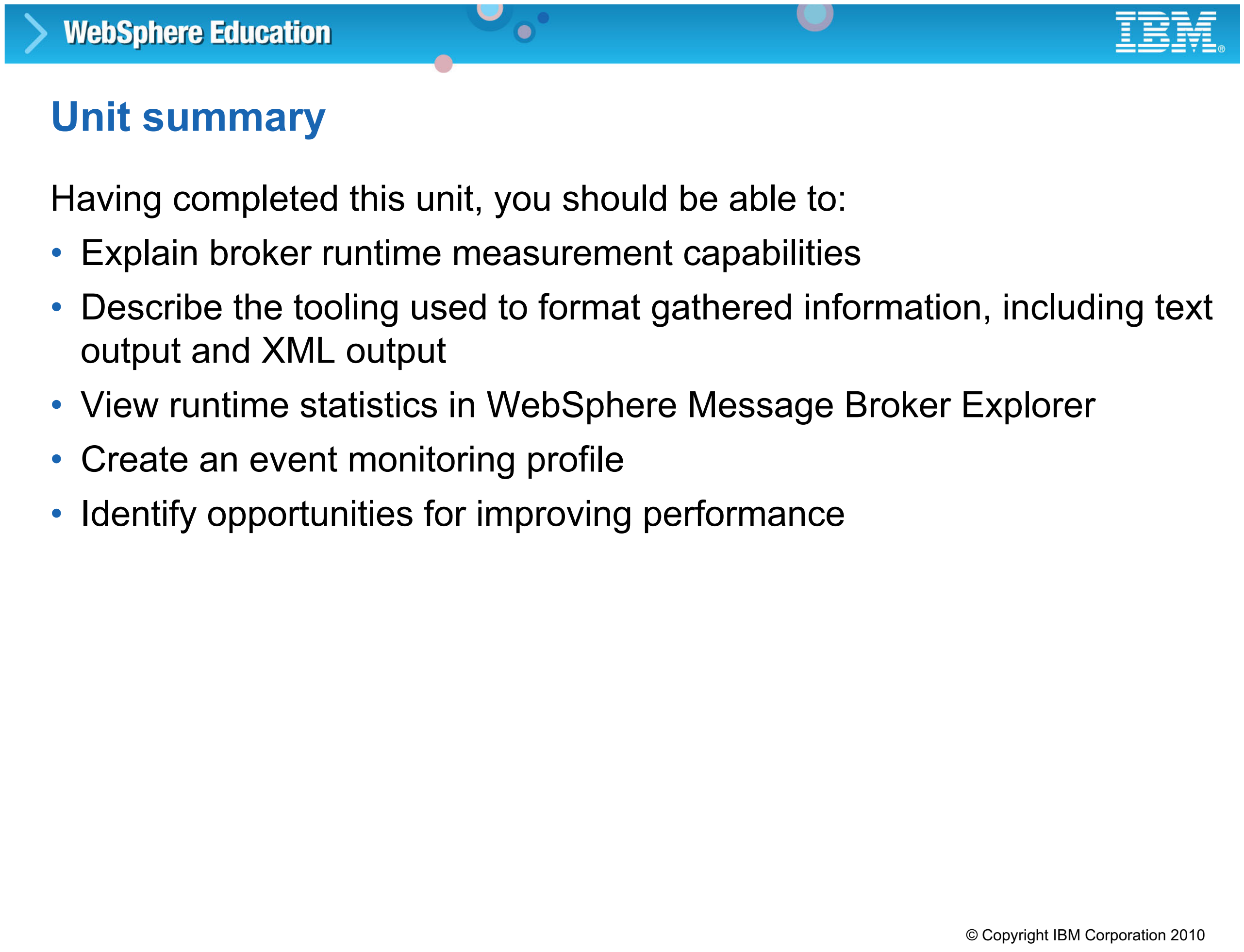
Instructor notes:

Purpose — Address the subject of obstructions in throughput and how they might be identified through testing.

Details — None.

Additional information — None.

Transition statement — Next: Unit summary



WebSphere Education

Unit summary

Having completed this unit, you should be able to:

- Explain broker runtime measurement capabilities
- Describe the tooling used to format gathered information, including text output and XML output
- View runtime statistics in WebSphere Message Broker Explorer
- Create an event monitoring profile
- Identify opportunities for improving performance

Figure 7-34. Unit summary WM643 / VM6431.0

Notes:

There are multiple aspects to improving performance. It is rare to change one parameter and have a dramatic effect on message throughput. In practice, the large changes come from improvements in a number of areas. It is important to have a consistent and reliable view of message throughput and resource usage as you tune a system. Without consistency, you cannot be sure of the benefits from making a change.

Tuning is an iterative process. Early changes tend to focus on the most significant metrics. With experience, later changes become less significant as do the returns. However, it is still worth addressing all areas in order to make the best possible improvement to your configuration.

Finally, you saw how it is possible to estimate message throughput for a message flow not yet developed, if you have a good understanding of the high-level requirements. Such estimates can be useful and can help to influence early design decisions when it becomes clear that a certain implementation choice is not going to meet objectives. This makes it possible to influence and improve design without ever having coded the message flow.

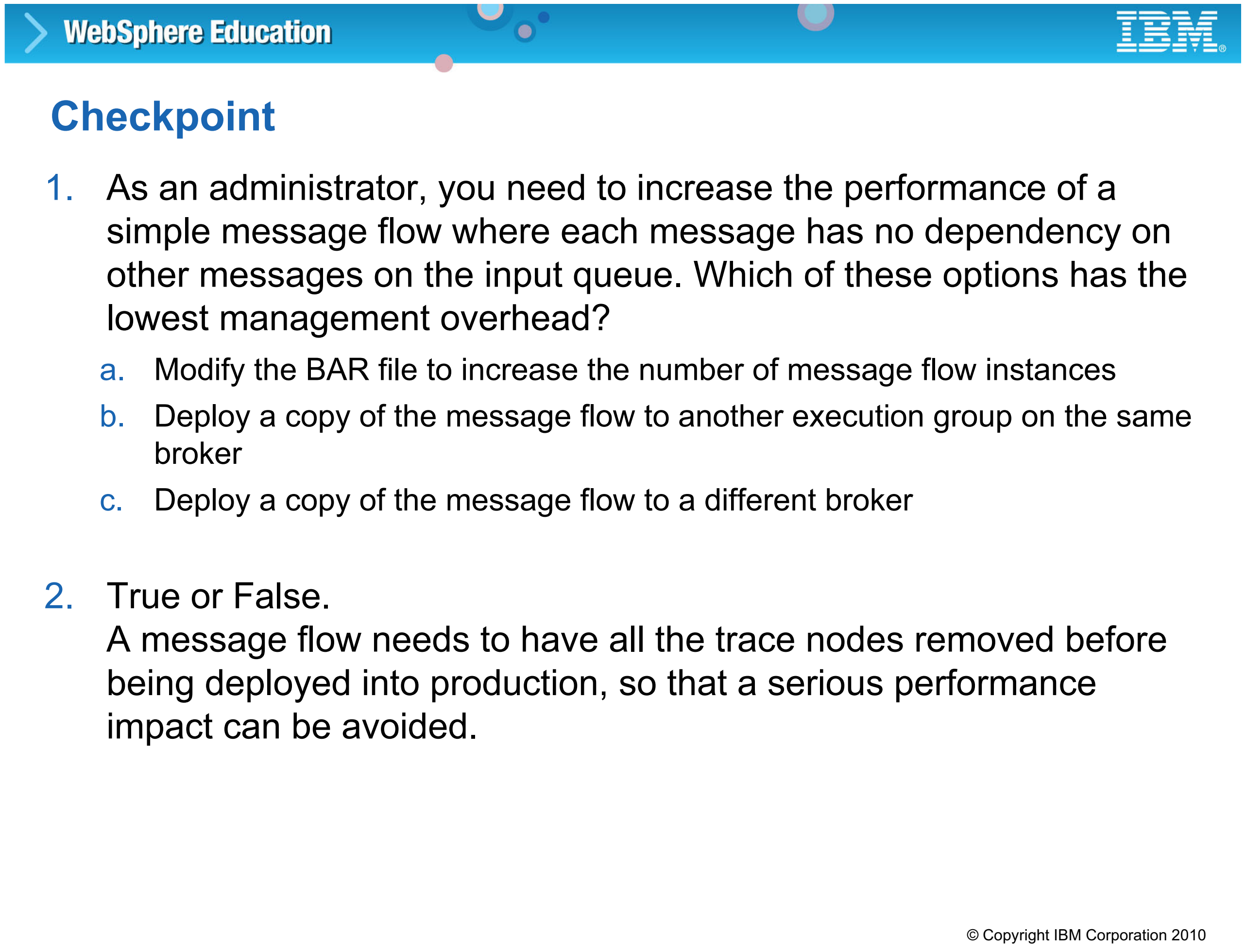
Instructor notes:

Purpose — Summarize the highlights of this unit.

Details — None.

Additional information — None.

Transition statement — An exercise on capturing broker statistics is included after the next unit which reviews publish/subscribe implementation.



WebSphere Education
IBM

Checkpoint

1. As an administrator, you need to increase the performance of a simple message flow where each message has no dependency on other messages on the input queue. Which of these options has the lowest management overhead?
   a. Modify the BAR file to increase the number of message flow instances
   b. Deploy a copy of the message flow to another execution group on the same broker
   c. Deploy a copy of the message flow to a different broker

2. True or False.
   A message flow needs to have all the trace nodes removed before being deployed into production, so that a serious performance impact can be avoided.

Figure 7-35. Checkpoint WM643 / VM6431.0

Notes:

Write your answers here:

1.

2.

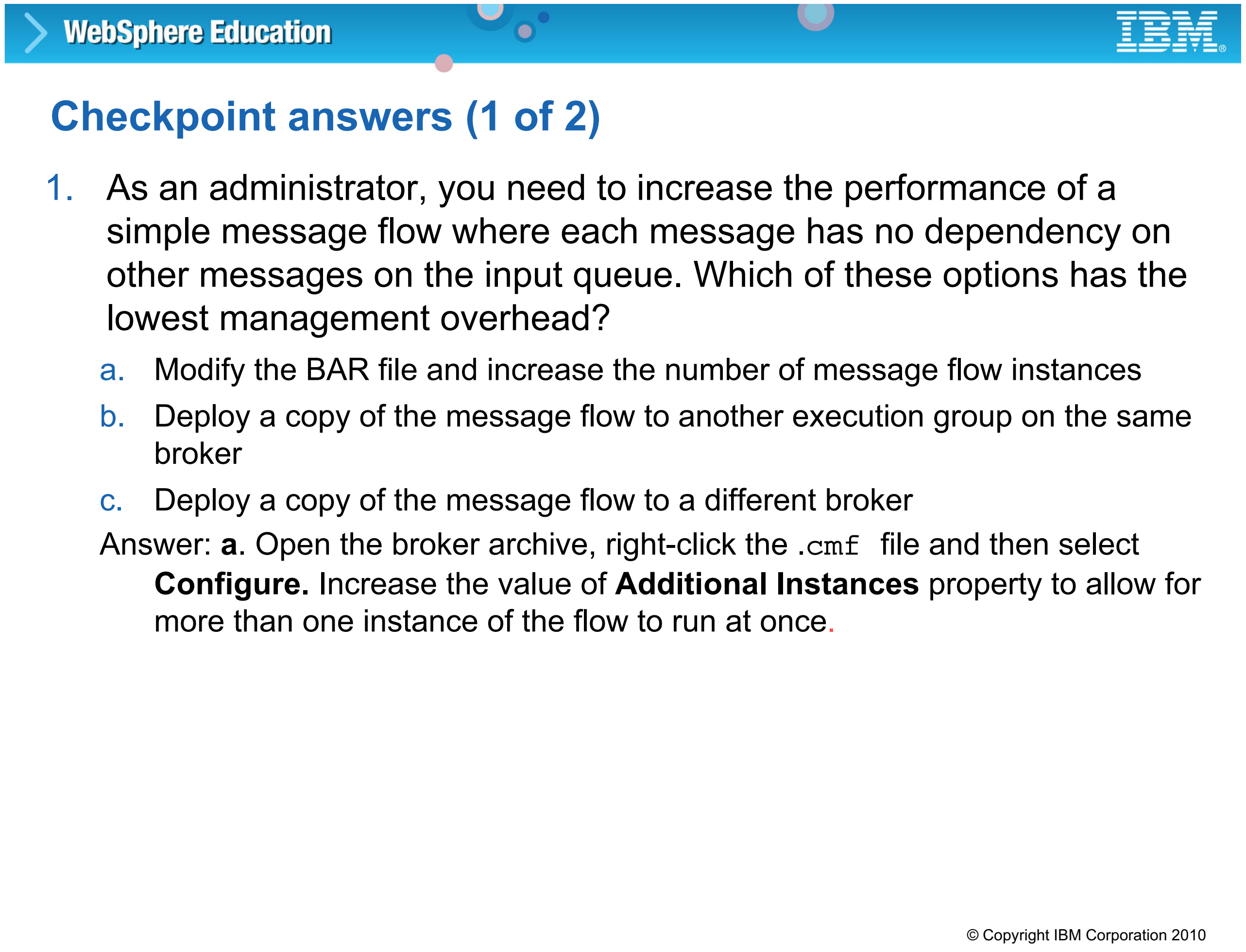
Instructor notes:

Purpose — End the unit with some questions.

Details —

Additional information —

Transition statement — Next: Checkpoint answers (1 of 2)



WebSphere Education
IBM

Checkpoint answers (1 of 2)

1. As an administrator, you need to increase the performance of a simple message flow where each message has no dependency on other messages on the input queue. Which of these options has the lowest management overhead?
   - a. Modify the BAR file and increase the number of message flow instances
   - b. Deploy a copy of the message flow to another execution group on the same broker
   - c. Deploy a copy of the message flow to a different broker

   Answer: **a**. Open the broker archive, right-click the `.cmf` file and then select **Configure.** Increase the value of **Additional Instances** property to allow for more than one instance of the flow to run at once.

Figure 7-36. Checkpoint answers (1 of 2)

WM643 / VM6431.0

Notes:

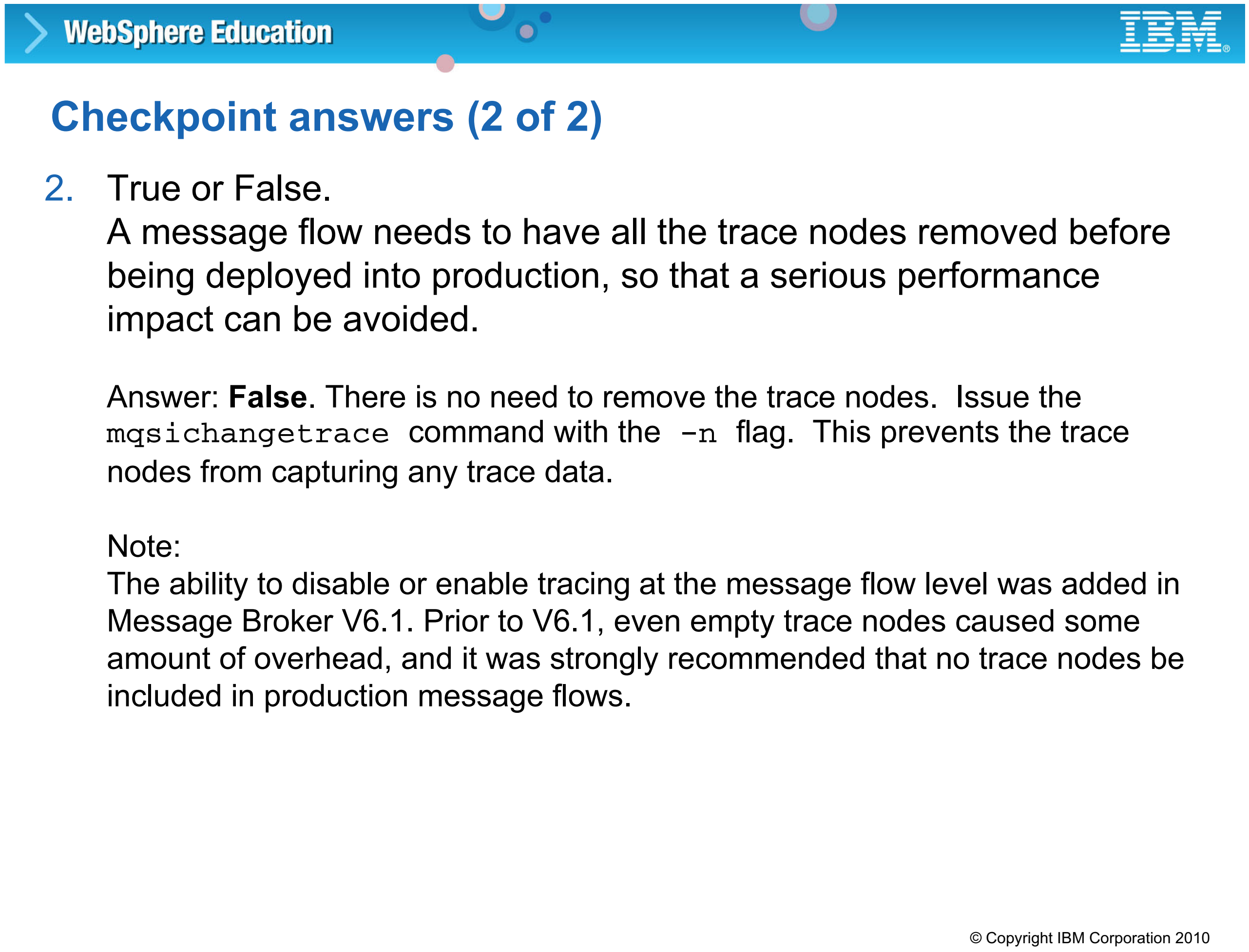
Instructor notes:

Purpose — Provide checkpoint answers.

Details —

Additional information —

Transition statement — Next: Checkpoint answers (2 of 2)



WebSphere Education

Checkpoint answers (2 of 2)

2. True or False.
   A message flow needs to have all the trace nodes removed before being deployed into production, so that a serious performance impact can be avoided.

   Answer: **False**. There is no need to remove the trace nodes. Issue the `mqsichangetrace` command with the `-n` flag. This prevents the trace nodes from capturing any trace data.

   Note:
   The ability to disable or enable tracing at the message flow level was added in Message Broker V6.1. Prior to V6.1, even empty trace nodes caused some amount of overhead, and it was strongly recommended that no trace nodes be included in production message flows.

Figure 7-37. Checkpoint answers (2 of 2)

Notes:

Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —

Unit 8. Publish/subscribe implementation overview

Estimated time

01:00 lecture, 01:00 exercise

What this unit is about

This unit reviews WebSphere MQ publish/subscribe and provides an overview of publish/subscribe implementation in WebSphere Message Broker.

What you should be able to do

After completing this unit, you should be able to:

- Use publish/subscribe in message flows
- Manage subscriptions using RFHUTIL and WebSphere Message Broker Explorer

How you will check your progress

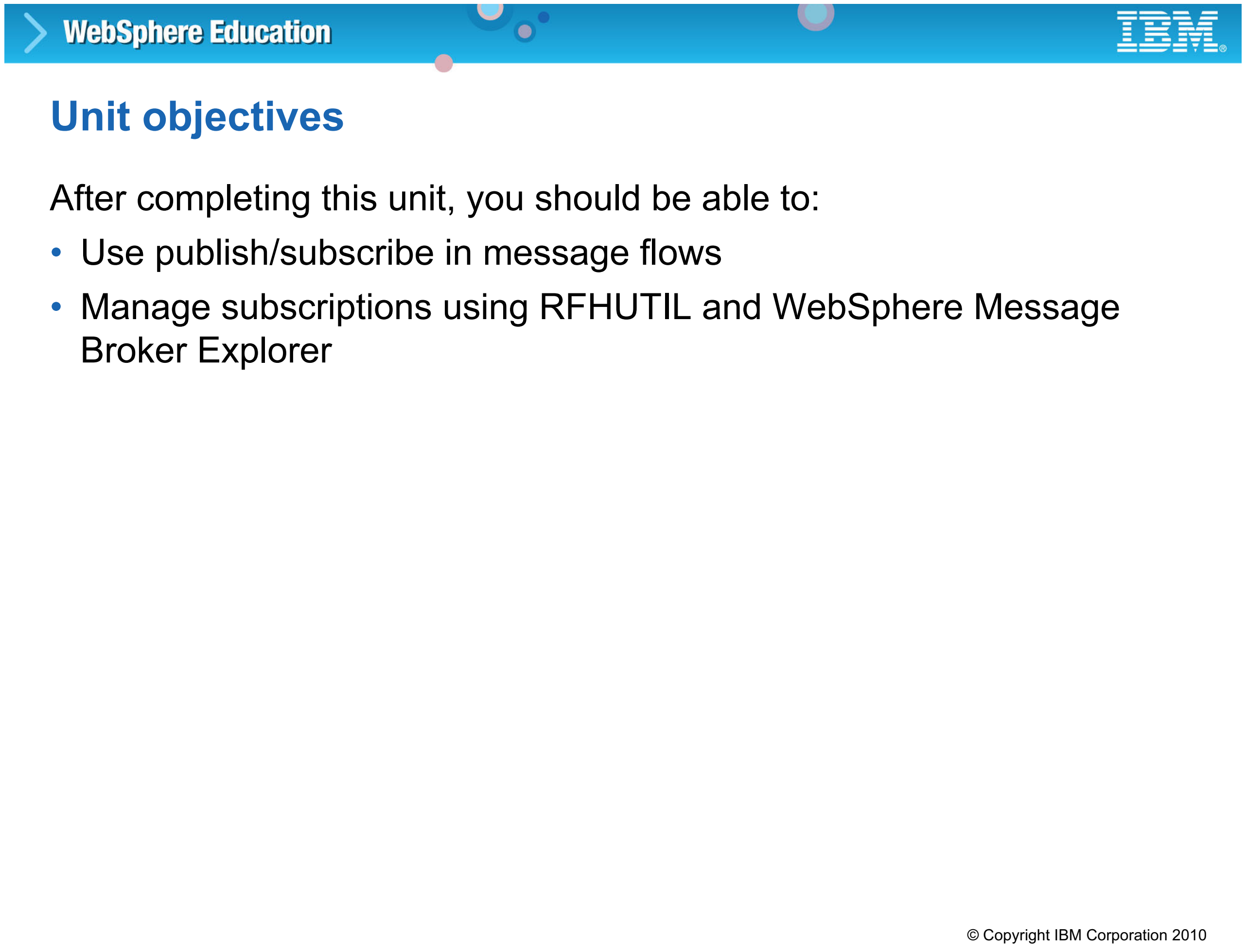
Accountability:

- Checkpoint questions
- Exercise 7, Accessing broker statistics

References

`www.ibm.com/software/integration/wbimessagebroker/library`
IBM WebSphere Message Broker Library

WebSphere Message Broker: Publish/Subscribe



WebSphere Education
IBM

Unit objectives

After completing this unit, you should be able to:

- Use publish/subscribe in message flows
- Manage subscriptions using RFHUTIL and WebSphere Message Broker Explorer

Figure 8-1. Unit objectives WM643 / VM6431.0

Notes:

Instructor notes:

Purpose —

Details —

Additional information —

Transition statement — Next: What is publish and subscribe

8.1. Using publish/subscribe in message flows

Instructor topic introduction

An introduction to publish/subscribe was presented in the prerequisite course WM201, IBM WebSphere MQ System Administration. This unit has some review information but concentrates more on giving the administrator an understanding for how WebSphere Message Broker interacts with WebSphere MQ for publish/subscribe support.

WebSphere Message Broker is a good sample of how publish/subscribe can be used for flexibility: brokers publish configuration and operational changes, and operational warnings (in XML). The format of these publications is described in the online help system, so that clients can write their own monitors, which register as subscribers for the appropriate broker topics.

What students will do — Students will listen to a lecture about general publish/subscribe messaging concepts, without going into detail about the WebSphere Message Broker implementation.

How students will do it — Students will answer checkpoint questions and complete an exercise.

What students will learn — How publish/subscribe is implemented in WebSphere Message Broker.

How this will help students in their job — Publish/subscribe represents a viable and simple means to use a reliable, robust messaging system such as that provided by WebSphere MQ.

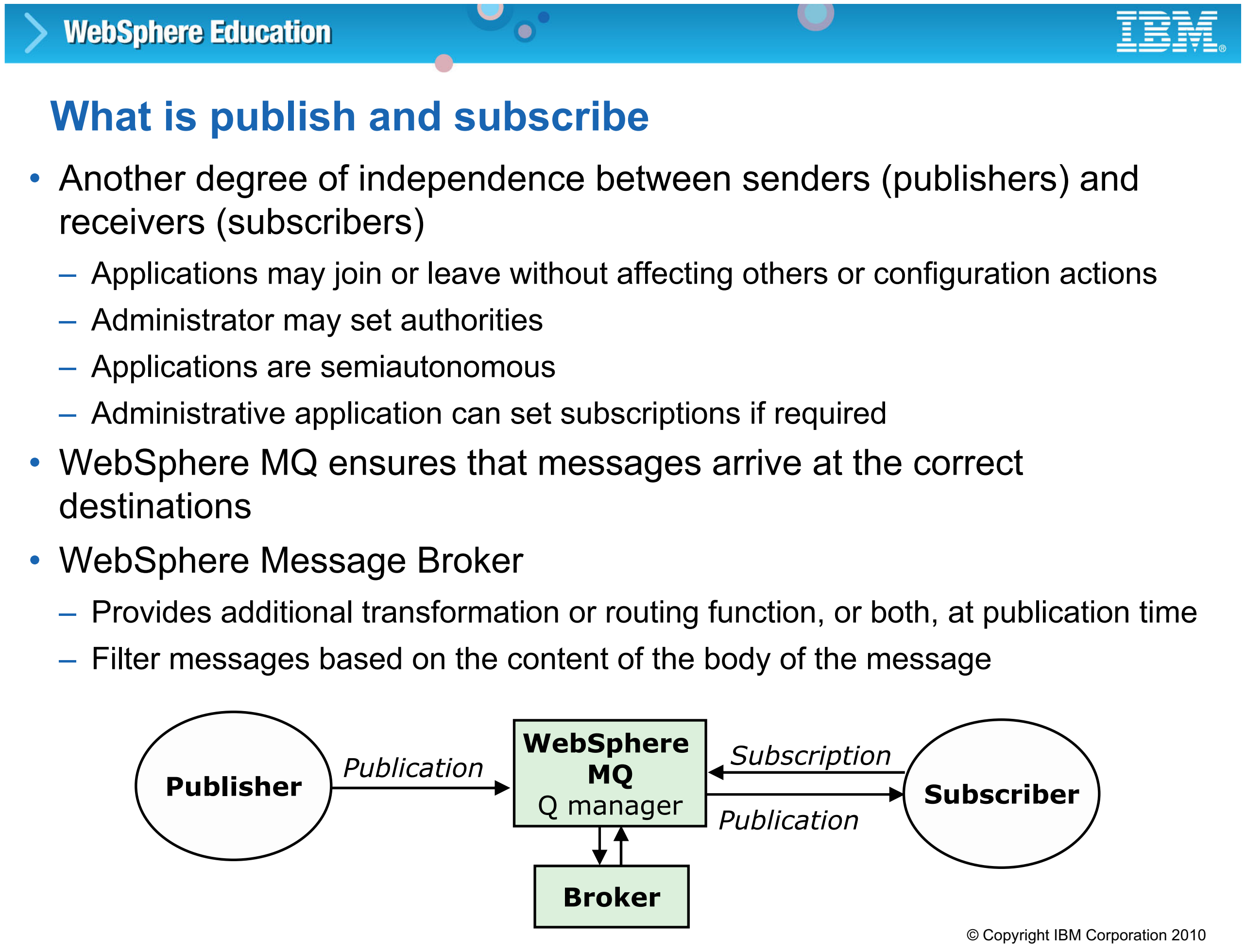
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —



WebSphere Education

What is publish and subscribe

- Another degree of independence between senders (publishers) and receivers (subscribers)
  - Applications may join or leave without affecting others or configuration actions
  - Administrator may set authorities
  - Applications are semiautonomous
  - Administrative application can set subscriptions if required
- WebSphere MQ ensures that messages arrive at the correct destinations
- WebSphere Message Broker
  - Provides additional transformation or routing function, or both, at publication time
  - Filter messages based on the content of the body of the message

© Copyright IBM Corporation 2010

Figure 8-2. What is publish and subscribe WM643 / VM6431.0

Notes:

Publish/subscribe decouples the sender (publisher, or producer) and the receiver (subscriber, or consumer) of the message. Applications can register and unregister to a topic through WebSphere MQ. No configuration overhead occurs.

WebSphere MQ provides much of the support for publish/subscribe. WebSphere Message Broker supplements this support by providing additional transformation or routing, and filtering based on topic. A *filter* is an expression applied to the content of a publication message to determine whether it matches a subscription

The advantage of a publish/subscribe system is that it can remove the unmanageable aspects of a point-to-point network and replace it with a simple network of a publisher, a broker, and all the subscribers. New subscription clients or services can be added without any affect or interruption in the service to other users. This provides a superior means of streamlined and efficient integration and growth across an enterprise, and, since WebSphere MQ can be used as the backbone for message delivery, all the benefits and features of WebSphere MQ are inherited.

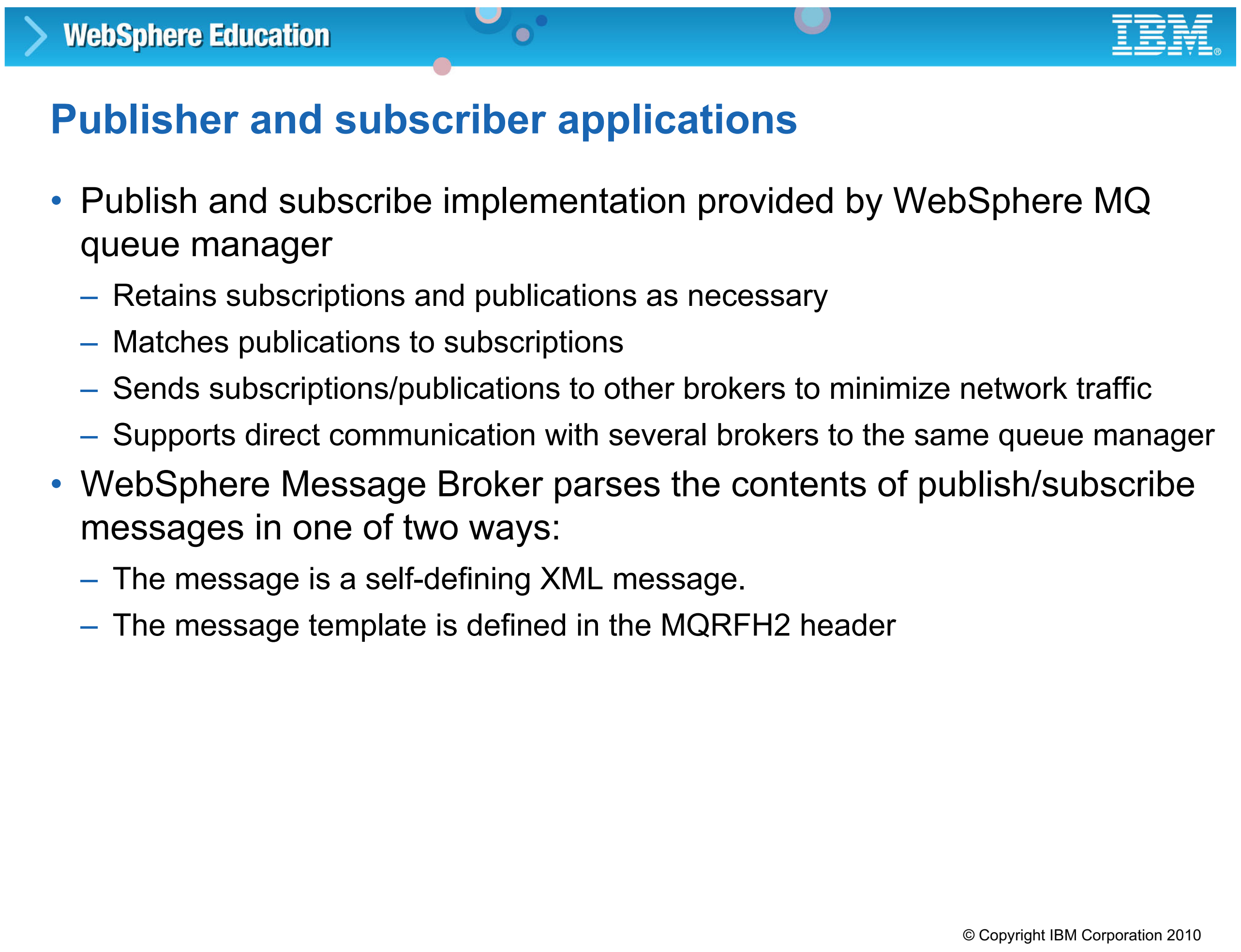
Instructor notes:

Purpose — Introduce publish/subscribe as an alternative messaging model, in addition to request/reply, and fire-and-forget. You also discuss advantages of the publish/subscribe messaging model.

Details — None.

Additional information — Alert students to the downside: the risk of misuse in the form of a spam generator with the possible consequence of flooding a network with nuisance messages. This threat can be countered with appropriate security controls that are identified and described later in the unit.

Transition statement — Who are the participants in publish/subscribe messaging?



WebSphere Education IBM

Publisher and subscriber applications

- Publish and subscribe implementation provided by WebSphere MQ queue manager
  - Retains subscriptions and publications as necessary
  - Matches publications to subscriptions
  - Sends subscriptions/publications to other brokers to minimize network traffic
  - Supports direct communication with several brokers to the same queue manager
- WebSphere Message Broker parses the contents of publish/subscribe messages in one of two ways:
  - The message is a self-defining XML message.
  - The message template is defined in the MQRFH2 header

Figure 8-3. Publisher and subscriber applications WM643 / VM6431.0

Notes:

Publish/subscribe applications involve these components:

Publisher

An application that generates publications, identified by a topic.

Subscriber

An application that subscribes to and receives publications, identified by a topic.

Publish/subscribe data

The message payload that the publisher and subscriber send to the broker are in free form. The MQRFH or MQRFH2 header carries the publish/subscribe commands (such as `Publish`) and supporting data (such as `Topic = Sports`). These commands can be supplied by a user application or added in a compute node within a message flow.

WebSphere Message Broker only parses a publish/subscribe message if content-based filtering is required for the message.

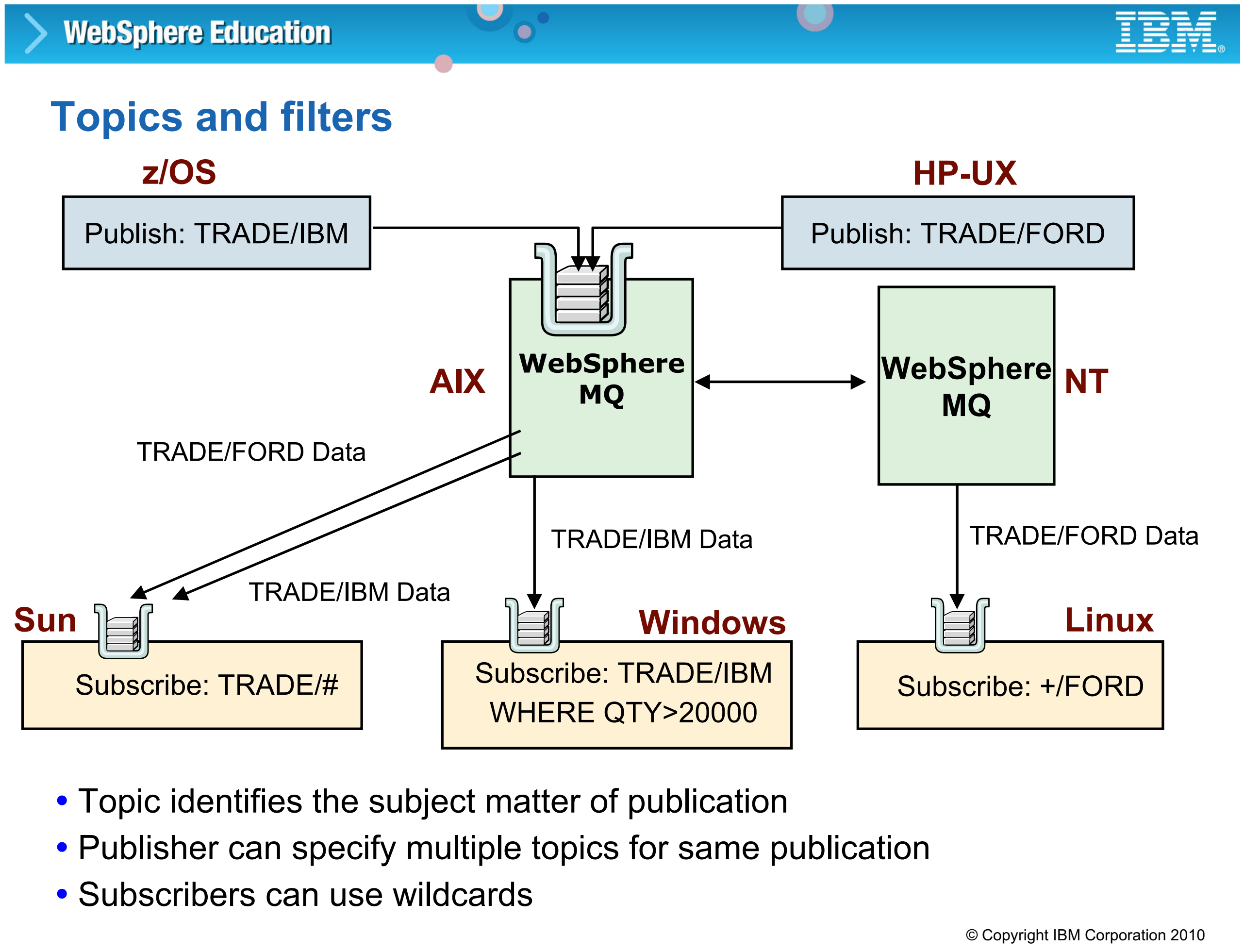
Instructor notes:

Purpose — Identify the participants in publish/subscribe messaging and explain the role of each.

Details — Publisher and subscriber are usually normal WebSphere MQ applications sending free-form messages with specific publish/subscribe commands in an MQRFH or MQRFH2 header directed to appropriate broker queues.

Additional information — None.

Transition statement — Look at how publications are identified; that is, the composition of a topic.



WebSphere Education

Topics and filters

z/OS

Publish: TRADE/IBM

HP-UX

Publish: TRADE/FORD

AIX

WebSphere MQ

WebSphere MQ

NT

TRADE/FORD Data

TRADE/IBM Data

TRADE/FORD Data

TRADE/IBM Data

Sun

Subscribe: TRADE/#

Windows

Subscribe: TRADE/IBM WHERE QTY>20000

Linux

Subscribe: +/FORD

- Topic identifies the subject matter of publication
- Publisher can specify multiple topics for same publication
- Subscribers can use wildcards

Figure 8-4. Topics and filters WM643 / VM6431.0

Notes:

The topic identifier represents the subject matter of a publication, specified by the publisher. The topic string can be any length. A topic tree structure can be built by qualifying the topic with the slash (/) character, which is recognized by the WebSphere Message Broker, but is not apparent to (base) WebSphere MQ publish/subscribe. Publishers can specify multiple topics for the same publication.

Subscribers subscribe to topics and can use wildcards:

+ matches with exactly one topic level (between / characters)
# matches with zero or more topic levels (at the beginning or end of a topic)

Subscribers can optionally include a content filter (specified in the MQRFH2) to further refine a request for matching publications. Wildcards are also permitted when specifying a content filter expression. In this figure, the “WHERE QTY > 20000” represents the content filter.

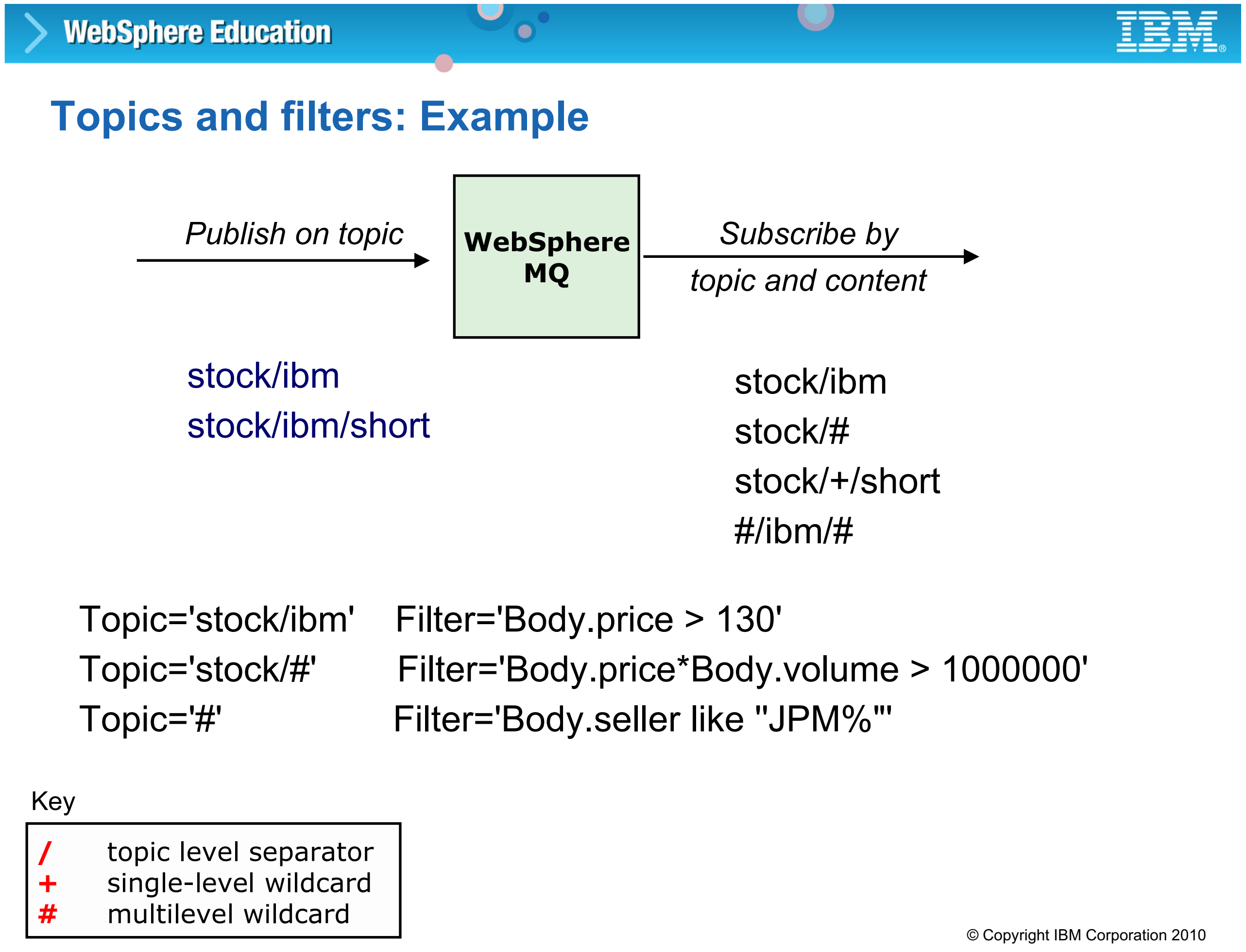
Instructor notes:

Purpose — Explain topics and filters and how they are specified.

Details — A content filter is specified using the same syntax as an ESQL expression used in a filter node. Likewise, it is evaluated for a true or false result. Wildcard use conforms to SQL specifications.

Additional information — The slash (/) denotes a topic hierarchy.

Transition statement — Next is an example of topics and filters.



WebSphere Education

Topics and filters: Example

Publish on topic → **WebSphere MQ** → *Subscribe by topic and content*

stock/ibm
stock/ibm/short

stock/ibm
stock/#
stock/+/short
#/ibm/#

Topic='stock/ibm'   Filter='Body.price > 130'
Topic='stock/#'   Filter='Body.price*Body.volume > 1000000'
Topic='#'   Filter='Body.seller like "JPM%"'

Key

/	topic level separator
+	single-level wildcard
#	multilevel wildcard

Figure 8-5. Topic filters: Example   WM643 / VM6431.0

Notes:

A subscription must include a topic. A generic topic is denoted by a number sign (#). Filters are optional. The syntax of subscription filters is exactly like those in Filter node (this is covered later in this course).

Note

Use only single quotation marks (‘) in filters.

Subscribers subscribe to topics and can use wildcards:

The figure shows examples of valid topic and filter strings using topic level separators, single-level wildcards, and multilevel wildcards. The plus sign (+) matches with exactly one topic level, between forward slash (/) characters. The number sign (#) matches with zero or more topic levels, at the beginning or end of the topic.

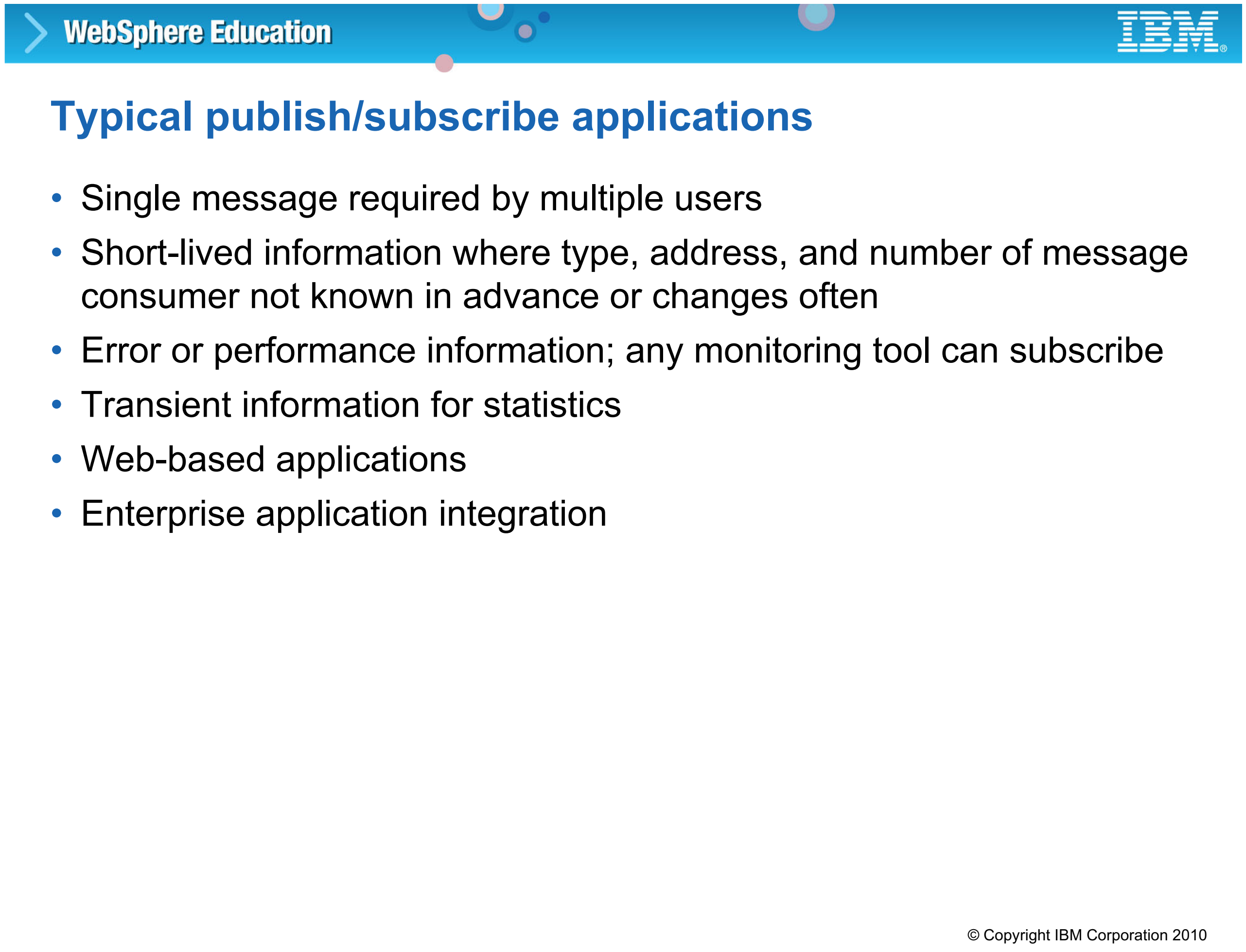
Instructor notes:

Purpose — Look at some examples of topics and filters.

Details — This is one more opportunity to point out the use of SQL wild cards for filters.

Additional information — None.

Transition statement — So what are examples of typical publish/subscribe applications?



WebSphere Education

Typical publish/subscribe applications

- Single message required by multiple users
- Short-lived information where type, address, and number of message consumer not known in advance or changes often
- Error or performance information; any monitoring tool can subscribe
- Transient information for statistics
- Web-based applications
- Enterprise application integration

Figure 8-6. Typical publish/subscribe applications WM643 / VM6431.0

Notes:

Publish/subscribe is a useful messaging model, offering many benefits when applied to the correct environment. It is most useful when there are many dynamic receiving applications, that is, applications that subscribe to varying and changing topics. Also, there is no link required between publishers and subscribers.

If there is a requirement to link applications together, for example, to bill a subscriber based on type and source of information, then it might be necessary to add more functionality to the publish/subscribe application. Doing so might reduce the benefit of low central administration (an inherent benefit of publish/subscribe).

If all the subscribers are known and remain relatively static and the type of information they receive is also a known quantity, it might be more suitable to use a simple point-to-point system which uses the broker to examine messages for content (perhaps with a filter node) and route to the relevant destination (rather than using the publish/subscribe model). In this case, there would be no need to formulate topic strings and have the receiving applications subscribe to these topics, especially if the information they request (that is, the topic) does not change and they require the same information about a regular basis.

Instructor notes:

Purpose — Discuss some general application scenarios with students. Suggestion: have them think of other use cases.

Details — None.

Additional information — None.

Transition statement — None.

WebSphere Education IBM

Publish/subscribe message flows and applications

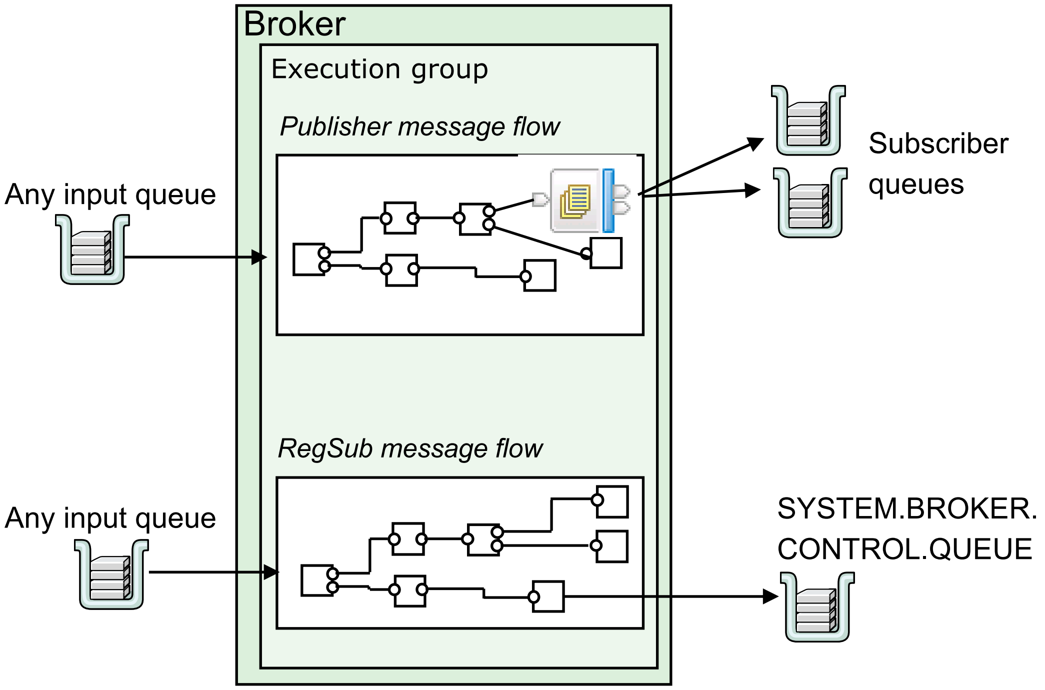


Figure 8-7. Publish/subscribe message flows and applications WM643 / VM6431.0

Notes:

Message flows can perform publish/subscribe tasks for applications. So existing applications can publish messages or register as subscribers without being modified with publish/subscribe commands.

Subscriptions can be durable, that is, the publications are stored in a queue and the consumer can get them whenever it fits. Subscriptions over WebSphere MQ Enterprise transport are always durable.

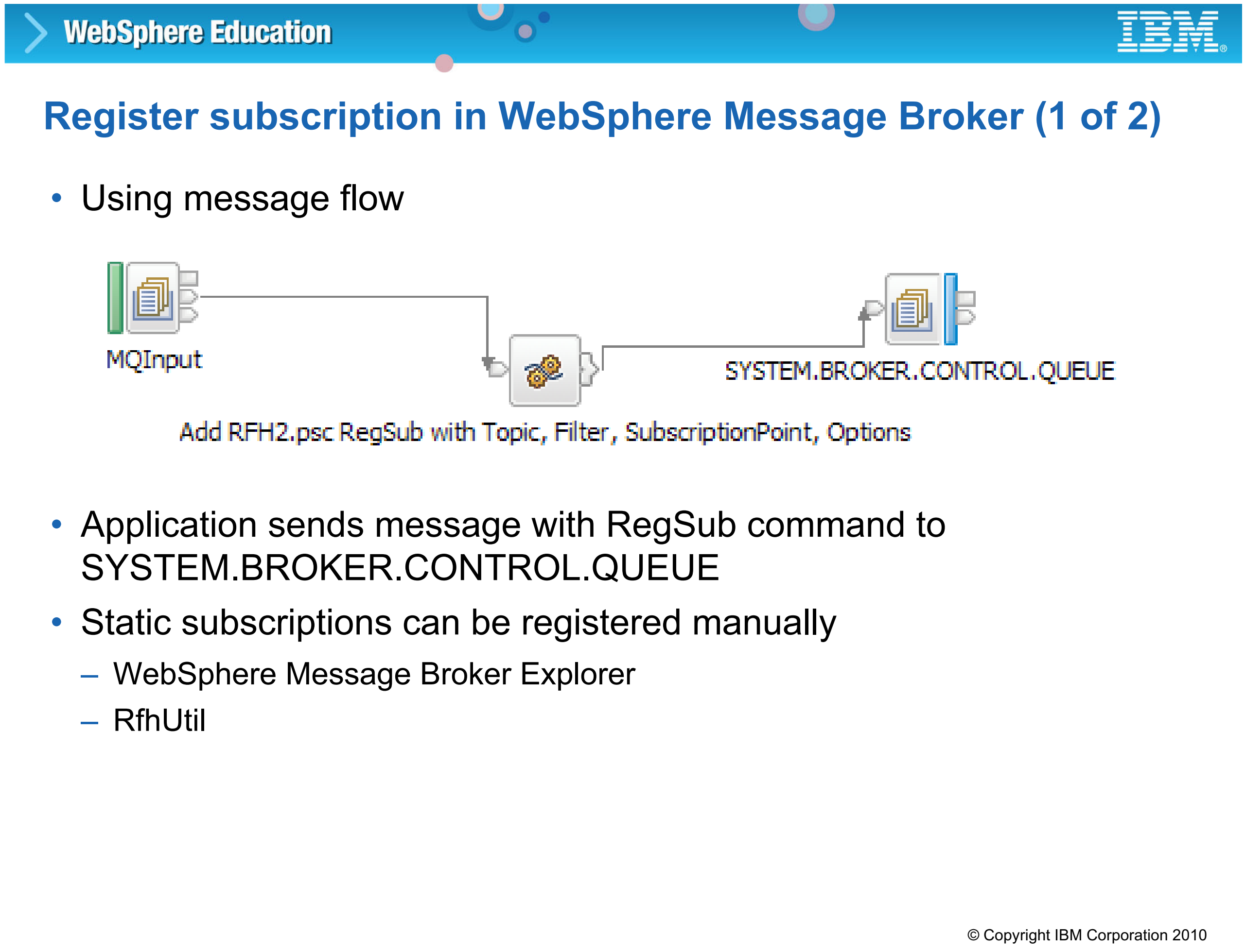
Instructor notes:

Purpose — Describe the queues (control and otherwise) used for publish/subscribe.

Details — None.

Additional information — None.

Transition statement — How can you register a subscription with the broker?



WebSphere Education

Register subscription in WebSphere Message Broker (1 of 2)

- Using message flow

MQInput
SYSTEM.BROKER.CONTROL.QUEUE
Add RFH2.psc RegSub with Topic, Filter, SubscriptionPoint, Options

- Application sends message with RegSub command to SYSTEM.BROKER.CONTROL.QUEUE
- Static subscriptions can be registered manually
  - WebSphere Message Broker Explorer
  - RfhUtil

Figure 8-8. Register subscription in WebSphere Message Broker (1 of 2) WM643 / VM6431.0

Notes:

The next few pages provide some rather in-depth development information. However, it is always useful to understand how a mechanism is implemented.

There is no specific *Subscribe* node provided by the WebSphere Message Brokers. However, you can build a message flow that receives messages on any input queue, builds an MQRFH2 adding the `RegSub` PSC command, and sends this message to the SYSTEM.BROKER.CONTROL.QUEUE.

The alternative is to use a tool such as RFHUTIL or WebSphere MQ to build a subscription. You should be aware that static registration means that the administrators have taken responsibility for the maintenance, resubscription, and subscription deletion of subscribing applications. So, static subscriber models do not fully use the dynamic nature of subscriptions.

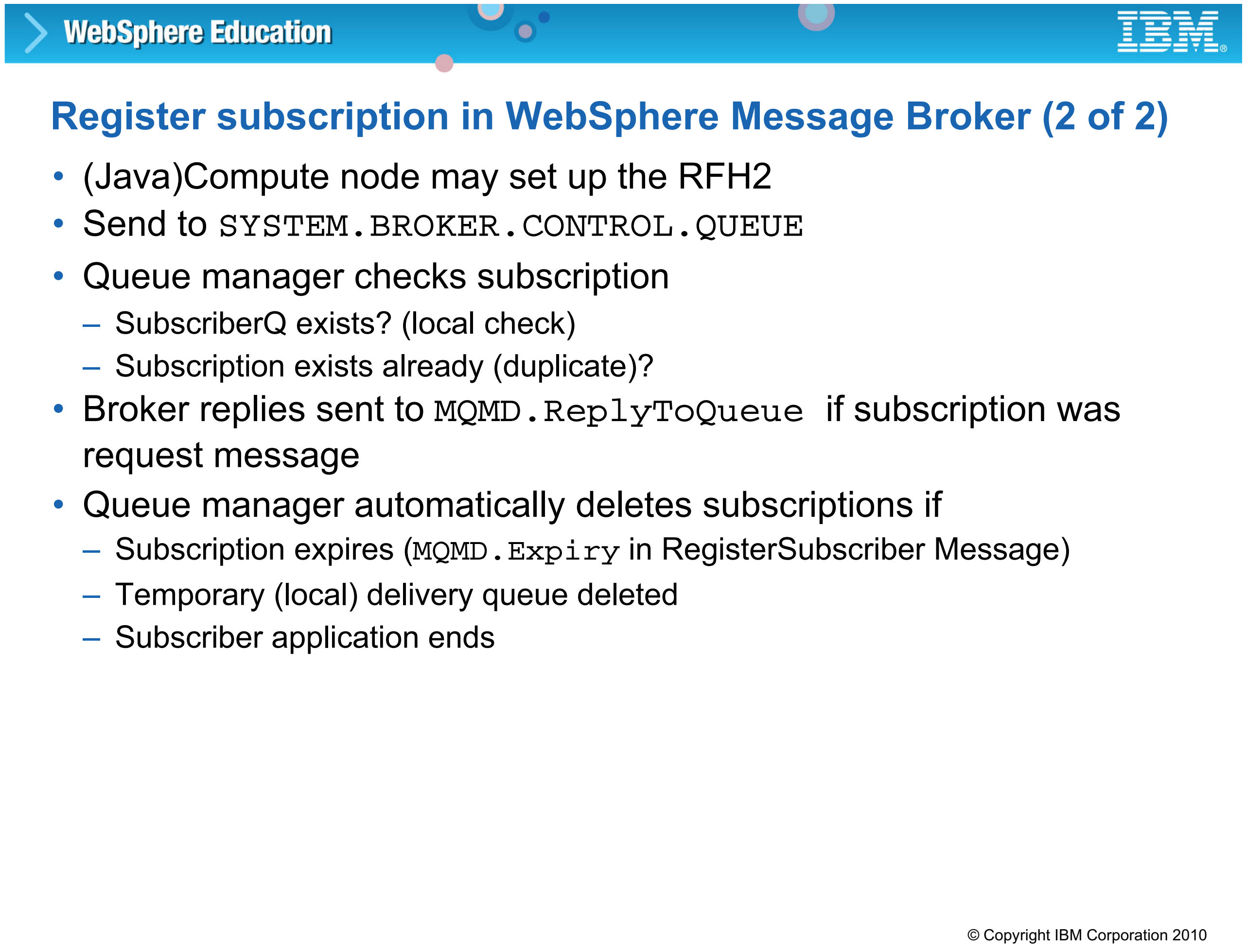
Instructor notes:

Purpose — Describe different ways to register a subscription in WebSphere Message Broker.

Details — None.

Additional information — Historically, the MQRFH (RFH stands for Rules Format Header) came on the scene with MQSeries Integrator V1. Because of its flexible nature, it was adopted by publish/subscribe (rather than designing an additional, unique publish/subscribe header). The MQRFH2 is the second generation of the MQRFH, first implemented with WebSphere MQ Integrator V2.0; and, as expected, it includes even more flexibility than its predecessor.

Transition statement — You have learned quite a bit about the MQRFH and MQRFH2 headers. You now will examine these headers more closely.



WebSphere Education

Register subscription in WebSphere Message Broker (2 of 2)

- (Java)Compute node may set up the RFH2
- Send to `SYSTEM.BROKER.CONTROL.QUEUE`
- Queue manager checks subscription
  - SubscriberQ exists? (local check)
  - Subscription exists already (duplicate)?
- Broker replies sent to `MQMD.ReplyToQueue` if subscription was request message
- Queue manager automatically deletes subscriptions if
  - Subscription expires (`MQMD.Expiry` in RegisterSubscriber Message)
  - Temporary (local) delivery queue deleted
  - Subscriber application ends

Figure 8-9. Register subscription in Message Broker (2 of 2) WM643 / VM6431.0

Notes:

On receipt of a RegSub command from SYSTEM.BROKER.CONTROL.QUEUE, the broker first checks the validity of the subscription. It is not possible to subscribe with a subscription queue that does not exist in normal WebSphere MQ distributed messaging. This means that either a local queue or a local definition of a remote queue must exist on the broker queue manager for `psc.QName`, a queue manager alias, or a transmission queue with the name of `psc.QMgrName`. The broker cannot determine if the definitions on the remote queue manager are valid.

If there is no `psc.QName` specified in the MQRFH2, the MQMD.ReplyToQueue is used as the subscriber queue instead.

The broker also checks subscription conditions so that there is no registered subscription that would result in the same publication being delivered to any subscriber queue twice.

You can check the success of the RegSub command using the normal WebSphere MQ request/reply mechanism. The broker sends a reply message (consisting of `MQRFH2.pscr` with Completion, Response, and Reason fields) to the reply queue as specified in the MQMD of the subscription message.

Instructor notes:

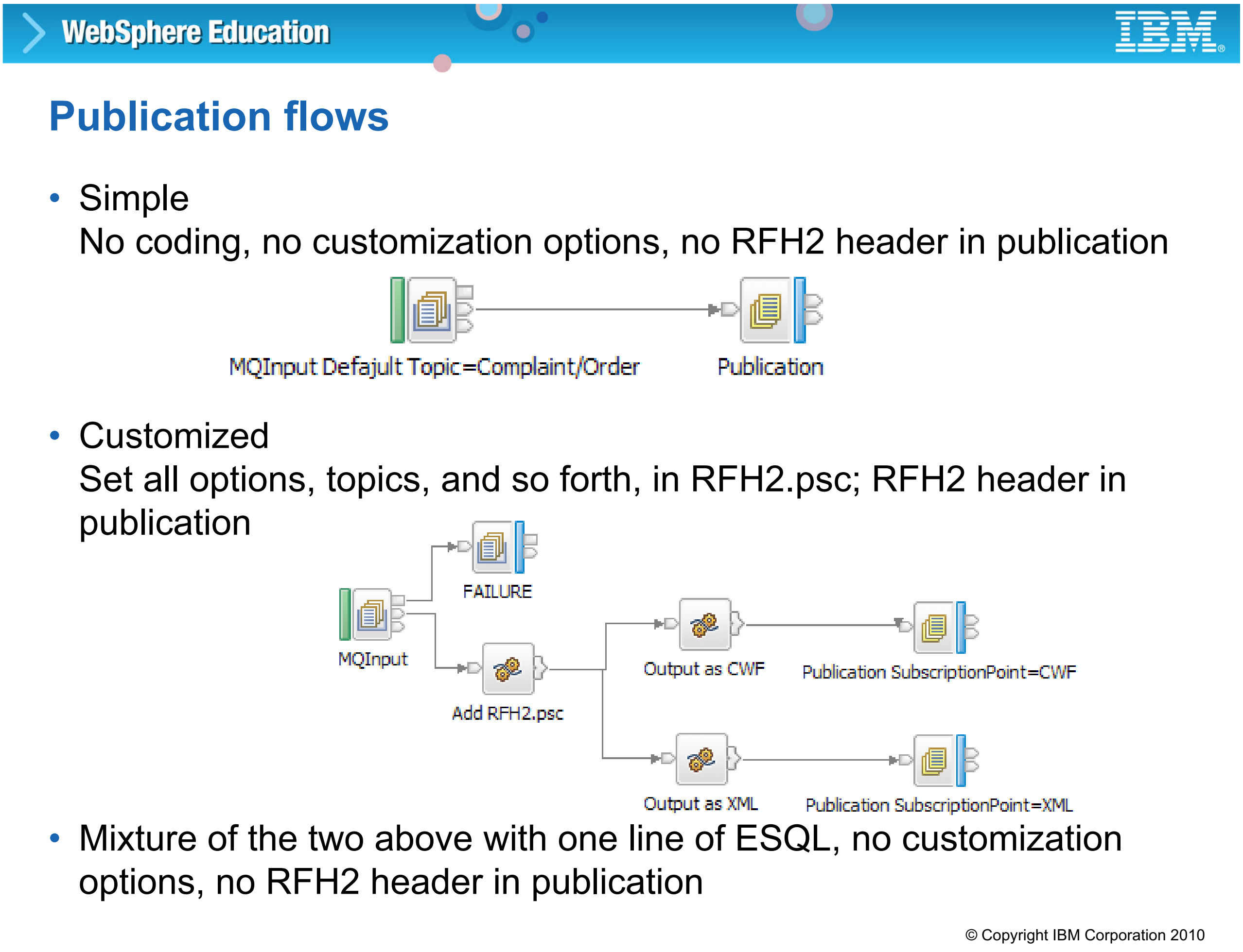
Purpose — Describe the subscription process a bit more.

Details — It is possible to issue a subscription within a message flow rather than in a typical user-written application.

Topic subscriptions use the typical publish/subscribe wildcards, but content filters do not. You must use SQL wildcards to set the content filters — the same syntax as in the filter node. Do not use quotation marks in the filter expression, however, because they are converted to `&apos;` and then do not match.

Additional information — The *Online Help System* provides more details on programming RFH2 headers for publish/subscribe (in Reference, publish/subscribe).

Transition statement — Now you will review some subscription examples and, based on what you have learned so far, determine what messages are delivered.



WebSphere Education

Publication flows

- Simple
  No coding, no customization options, no RFH2 header in publication

MQInput Defajult Topic=Complaint/Order
Publication

- Customized
  Set all options, topics, and so forth, in RFH2.psc; RFH2 header in publication

FAILURE
MQInput
Add RFH2.psc
Output as CWF
Publication SubscriptionPoint=CWF
Output as XML
Publication SubscriptionPoint=XML

- Mixture of the two above with one line of ESQL, no customization options, no RFH2 header in publication

Figure 8-10. Publication flows WM643 / VM6431.0

Notes:

WebSphere Message Broker offers a special publication node that implicitly sends the publication message to the SYSTEM.BROKER.CONTROL.QUEUE. Publication nodes are used instead of MQOutput nodes. The Publication node has no output terminal, not even on Failure (the broker answers on the ReplyToQueue).

Before publishing, you must at least specify a topic. Even if your message contains no `RFH2.psc` commands, it is possible to specify a default topic in the MQInput node. In that case, the publication does not contain an RFH2 header, which can be an advantage for legacy receiver applications that cannot handle this header. An alternative is to use a Compute node to set the OutputRoot.Properties.Topic field, which has the same effect, plus the additional advantage of being able to set the topic dynamically.

To specify further publish/subscribe options, such as retained publication and local publication, you must set up the appropriate PubSub Commands in the RFH or RFH2 header. If the sending application does not supply this, use a Compute Node in the message flow to add the RFH2 PubSub commands.

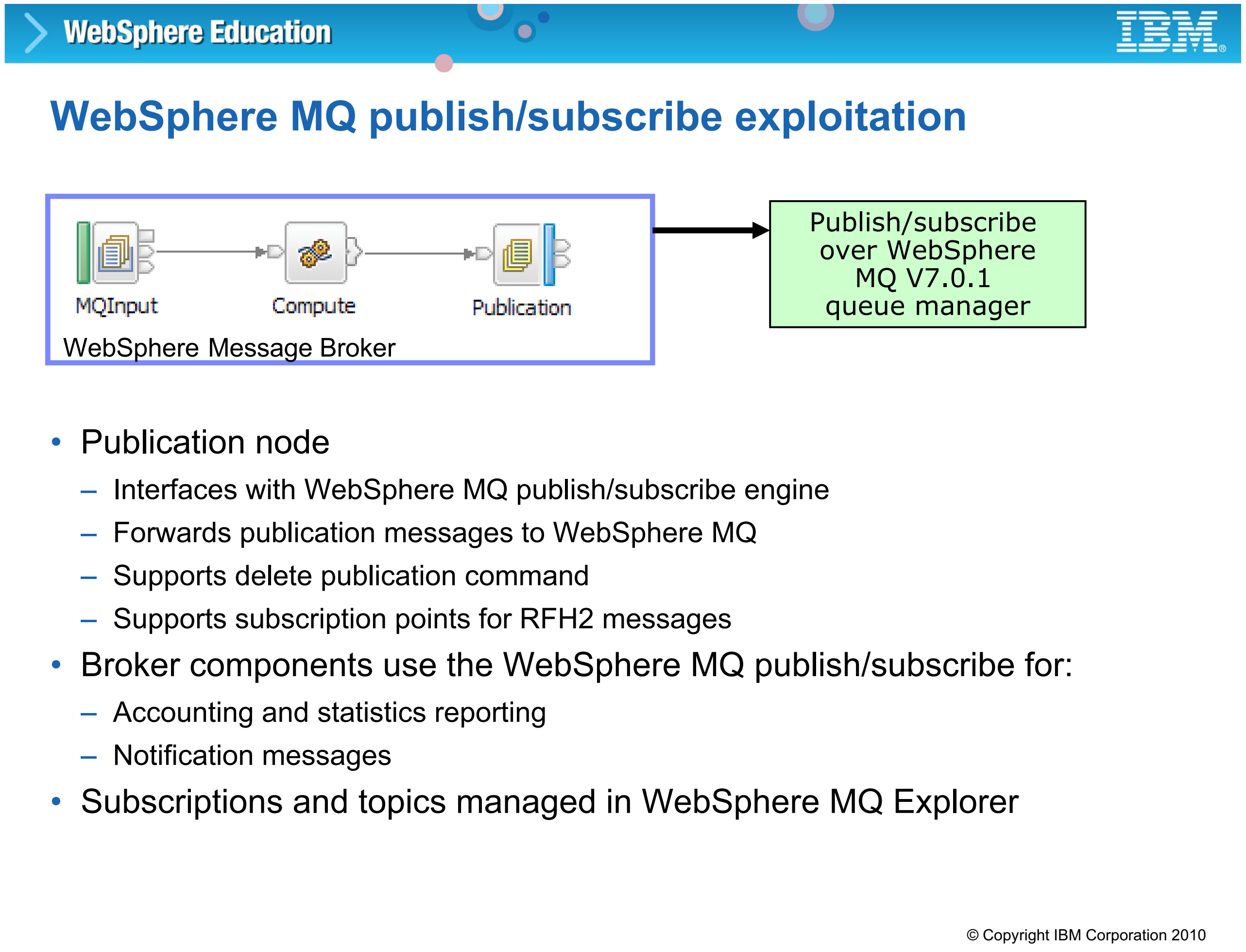
Instructor notes:

Purpose — Introduce the publication node.

Details — Try to avoid programming questions (or detailed questions on ESQL and the compute node) since this is an administration course. Being administrators, they are not expected to master some of the node internals as is the case with developers.

Additional information — None.

Transition statement — Now there is just a bit more detail about the publication node.



WebSphere Education

WebSphere MQ publish/subscribe exploitation

- Publication node
  - Interfaces with WebSphere MQ publish/subscribe engine
  - Forwards publication messages to WebSphere MQ
  - Supports delete publication command
  - Supports subscription points for RFH2 messages
- Broker components use the WebSphere MQ publish/subscribe for:
  - Accounting and statistics reporting
  - Notification messages
- Subscriptions and topics managed in WebSphere MQ Explorer

Figure 8-11. WebSphere MQ publish/subscribe exploitation

Notes:

In WebSphere Message Broker V7, the publication messages are all sent to WebSphere MQ. WebSphere Message Broker also uses WebSphere MQ publish/subscribe for other functions, such as accounting and statistics and for monitoring, and for internal messages that are passing between internal components.

WebSphere Message Broker parses the message being propagated to the Publication node and calls the WebSphere MQ API so that WebSphere MQ can match and deliver to subscribers at the correct subscription level.

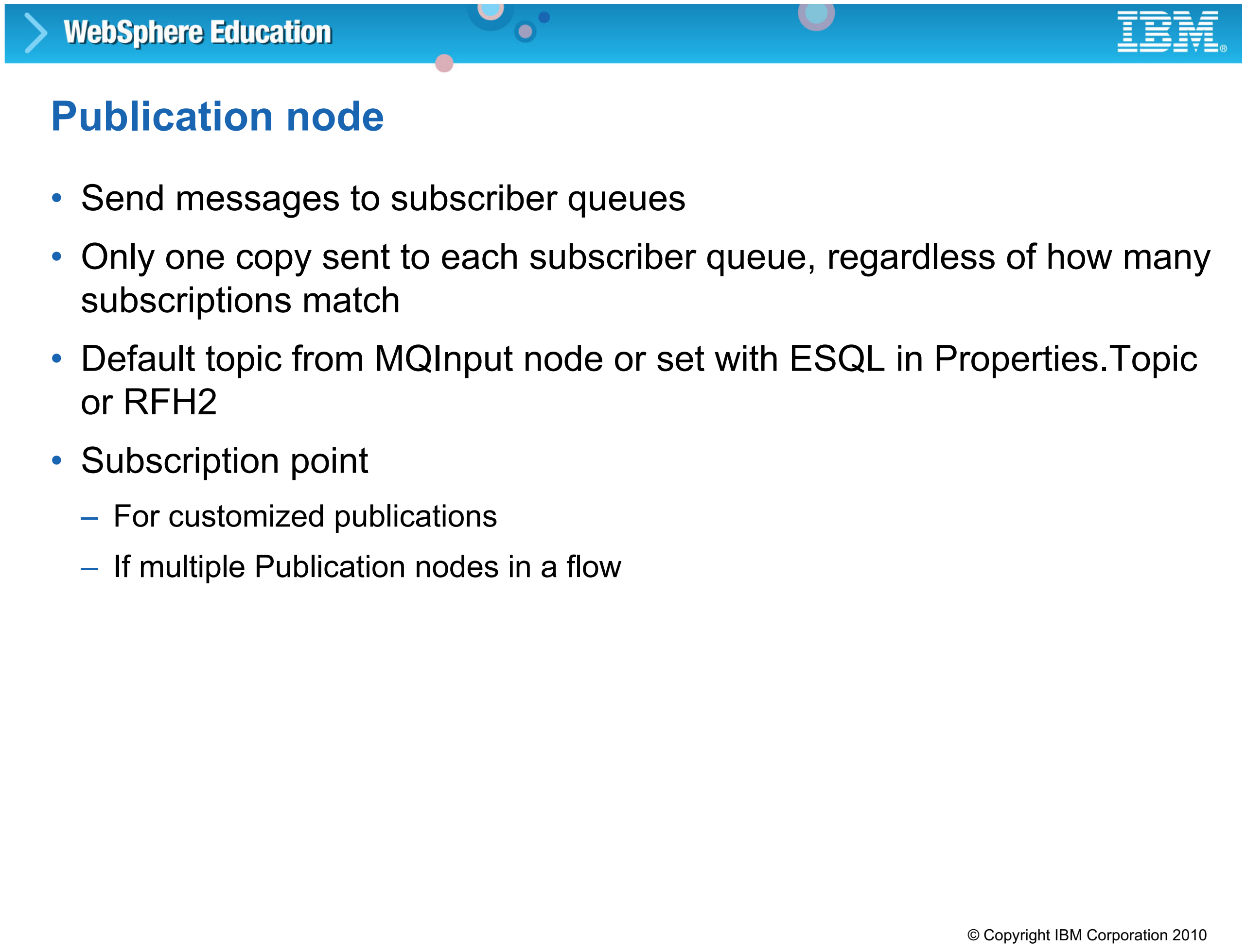
Instructor notes:

Purpose — Describe how WebSphere MQ and WebSphere Message Broker interact to provide publish/subscribe support.

Details — Although the internals have changed, the external components, such as RFH1 + RFH2, streams, and subscription points, all function as in previous versions

Additional information — None.

Transition statement — None.



WebSphere Education

Publication node

- Send messages to subscriber queues
- Only one copy sent to each subscriber queue, regardless of how many subscriptions match
- Default topic from MQInput node or set with ESQL in Properties.Topic or RFH2
- Subscription point
  - For customized publications
  - If multiple Publication nodes in a flow

Figure 8-12. Publication node WM643 / VM6431.0

Notes:

The Publication node is relatively simple. An input terminal receives the data to be published. You can identify a subscription point within the properties of the Publication node. The subscription point differentiates multiple Publication nodes in the same message flow and therefore represents a specific path through the message flow, for example, if the same message is published in different formats. Subscribers can choose to which subscription point they subscribe. An unnamed publication node (that is, one without a specific subscription point) is known as the default publication node.

The Publication node always tries to deliver its publication. If the subscriber queue (or, in the case of a remote subscriber queue, the transmission queue) is not reachable, the broker tries to put the message to the DLQ. If this is not possible, and the publication flow is transactional which causes **all** publications to be rolled back.

Having 0 (zero) registered subscribers is not an error. The message is sent nowhere with no warning.

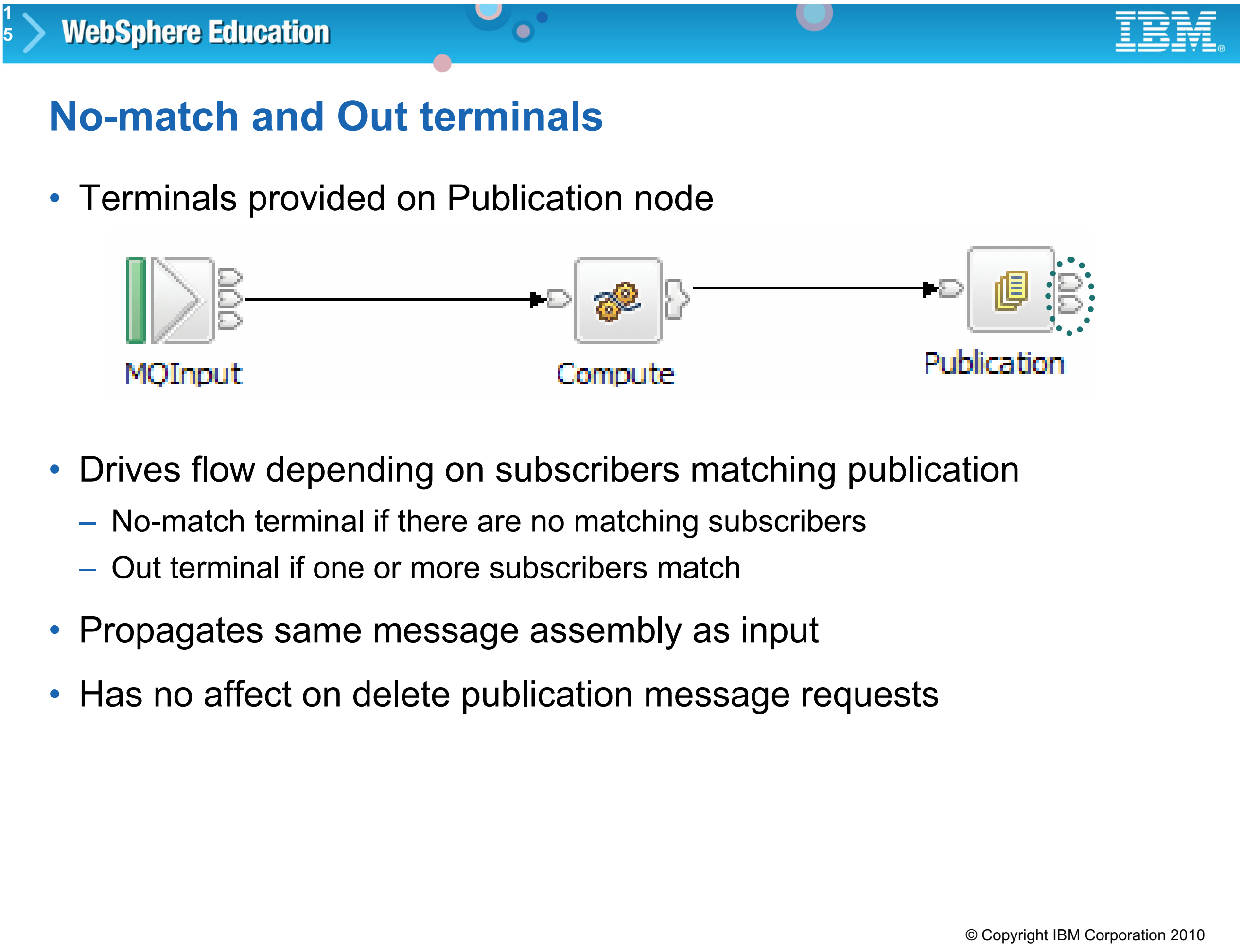
Instructor notes:

Purpose — Provide additional detail regarding the publication node.

Details — None.

Additional information — None.

Transition statement — Let us now talk about explicit error handling in message flows.



No-match and Out terminals

- Terminals provided on Publication node
- Drives flow depending on subscribers matching publication
  - No-match terminal if there are no matching subscribers
  - Out terminal if one or more subscribers match
- Propagates same message assembly as input
- Has no affect on delete publication message requests

Figure 8-13. No-match and Out terminals

WM643 / VM6431.0

Notes:

The Publication node can be configured to detect whether subscribers exist for the publication. This is supported through the introduction of two terminals on the Publication node:

- No-match terminal: The publication message is propagated down this terminal if no subscribers match the publication message
- Out terminal: The publication message is propagated down this terminal if at least one subscriber matched this publication message

If there are subscribers, a destination list can be exposed in the message tree. By default, if both terminals are connected, no destination list is created. Creation of the destination list is an option, as the list can be large and generating the list might affect performance.

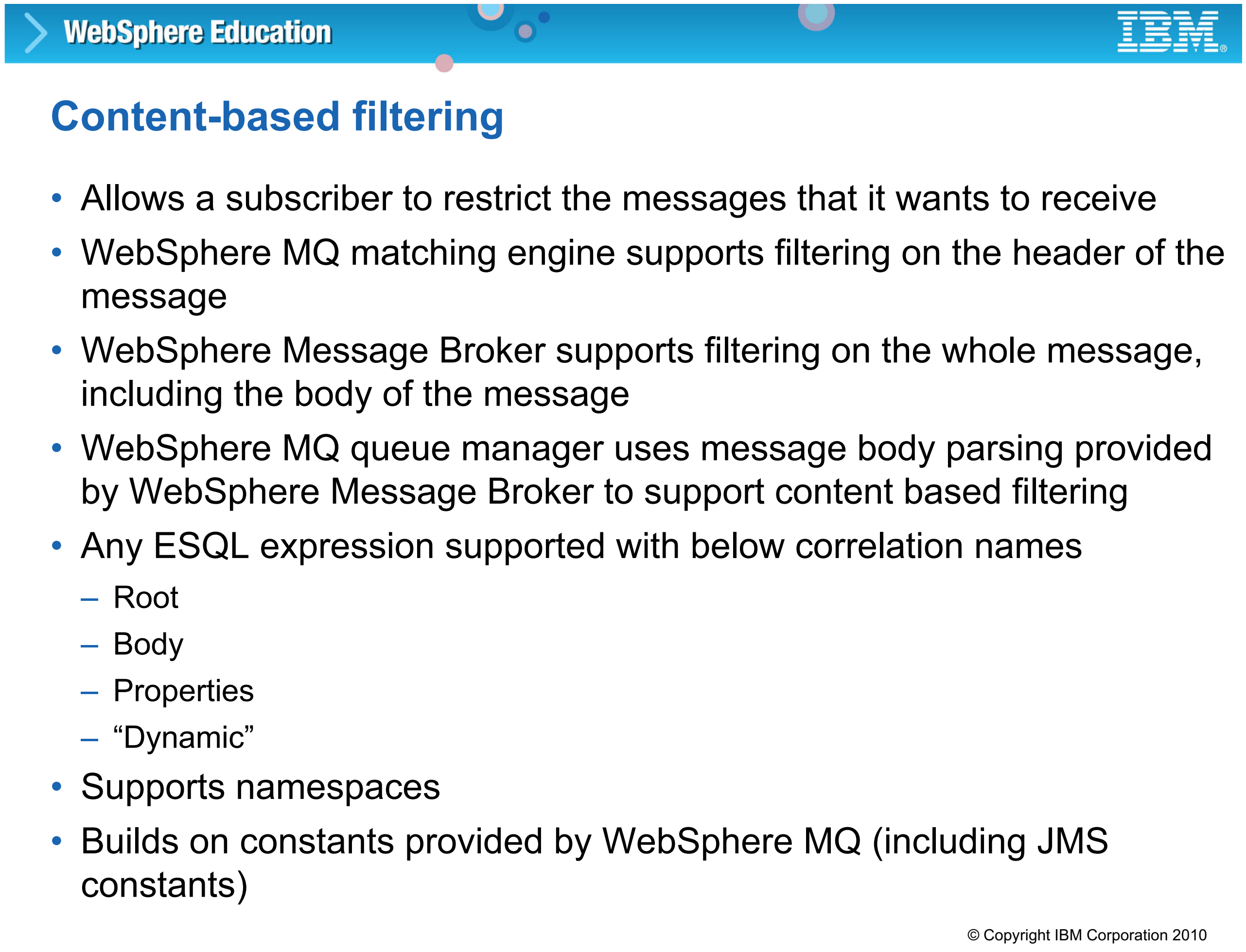
Instructor notes:

Purpose — Describes how the WebSphere Message Broker can handle errors such as no subscribers.

Details — None.

Additional information — None.

Transition statement — WebSphere Message Broker supplements WebSphere MQ publish/subscribe by providing content-based filtering.



WebSphere Education

Content-based filtering

- Allows a subscriber to restrict the messages that it wants to receive
- WebSphere MQ matching engine supports filtering on the header of the message
- WebSphere Message Broker supports filtering on the whole message, including the body of the message
- WebSphere MQ queue manager uses message body parsing provided by WebSphere Message Broker to support content based filtering
- Any ESQL expression supported with below correlation names
  - Root
  - Body
  - Properties
  - “Dynamic”
- Supports namespaces
- Builds on constants provided by WebSphere MQ (including JMS constants)

© Copyright IBM Corporation 2010

Figure 8-14. Content-based filtering WM643 / VM6431.0

Notes:

Content-based filtering allows a subscriber to restrict or filter messages that it wants to receive, in addition to the topics that it specified. This filter is specified as an SQL expression, such as the following: `Body.Name LIKE 'Smit%'`

WebSphere MQ supports a limited amount filtering on the header and message properties, but it cannot filter on the body of the message. This is because it typically cannot parse this part of the message. WebSphere Message Broker supports content-based filtering on the body of the message, so it is important that this is supported.

Content-based filtering builds on constants provided by WebSphere MQ, such as JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSPriority, JMSReplyTo, JMSTimestamp, and JMSType.

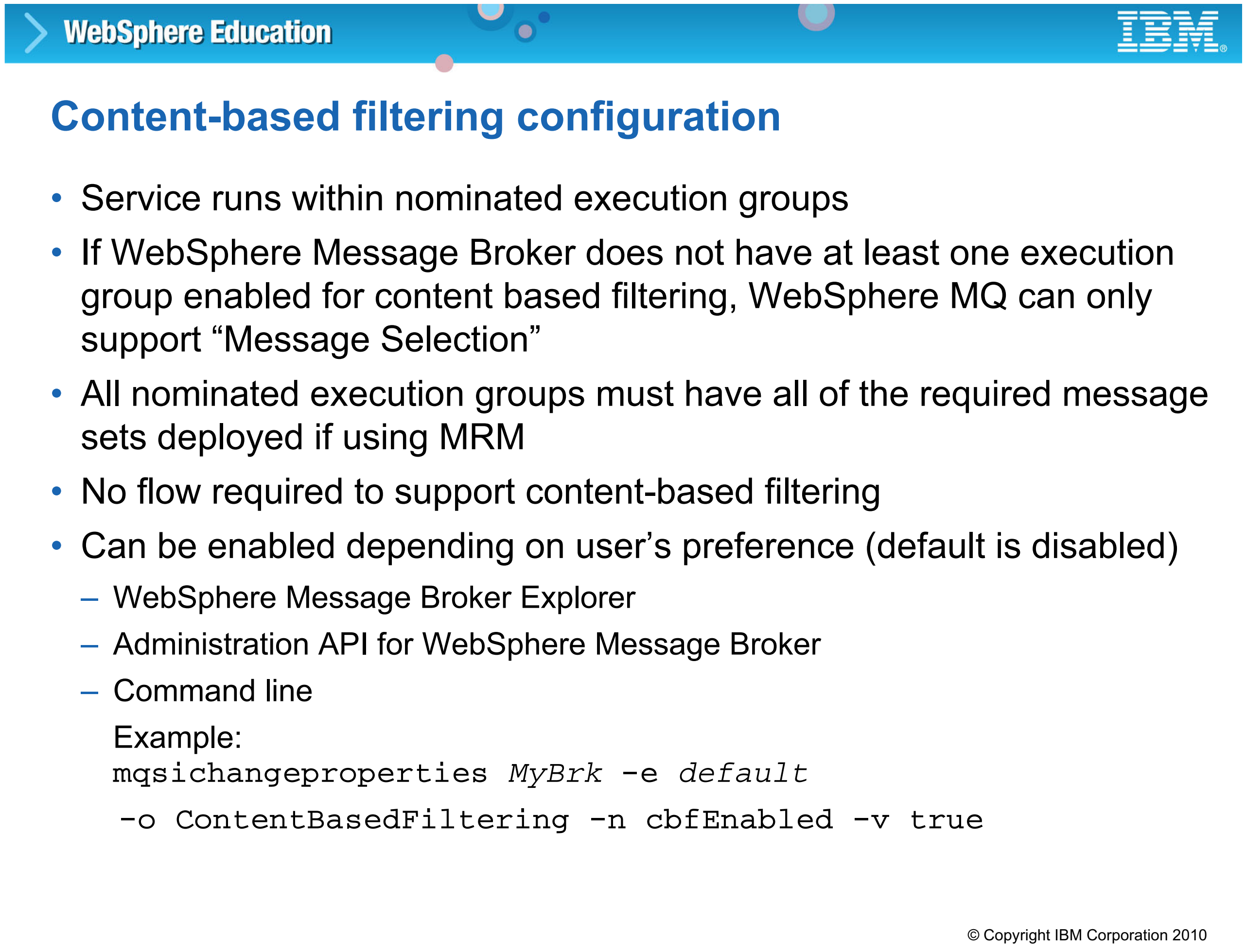
Instructor notes:

Purpose — Describe content-based filtering.

Details — None.

Additional information — None.

Transition statement — Some additional configuration is required to support content-based filtering.



WebSphere Education

Content-based filtering configuration

- Service runs within nominated execution groups
- If WebSphere Message Broker does not have at least one execution group enabled for content based filtering, WebSphere MQ can only support “Message Selection”
- All nominated execution groups must have all of the required message sets deployed if using MRM
- No flow required to support content-based filtering
- Can be enabled depending on user’s preference (default is disabled)
  - WebSphere Message Broker Explorer
  - Administration API for WebSphere Message Broker
  - Command line

    Example:

```
mqsichangeproperties MyBrk -e default
 -o ContentBasedFiltering -n cbfEnabled -v true
```

Figure 8-15. Content-based filtering configuration

Notes:

WebSphere Message Broker enables WebSphere MQ to support extended message selection and content-based filtering.

Additional configuration options are:

- Content-based filtering is enabled or disabled
- Number of validation threads (1 - 32)
- Number of evaluation threads (1 - 32)

Instructor notes:

Purpose — Describe the configuration requirements for content-based filtering.

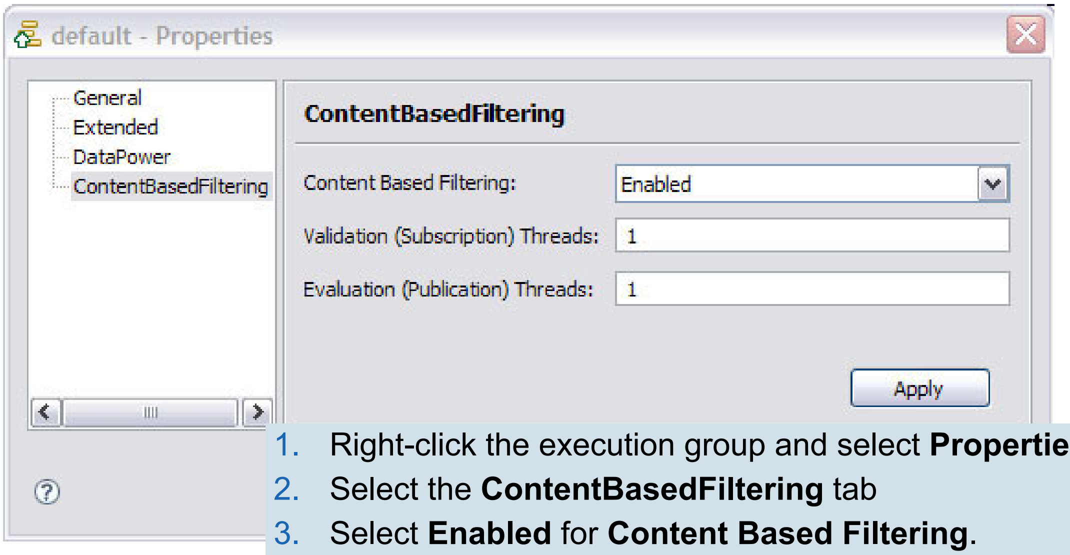
Details — None.

Additional information — None.

Transition statement — None.

WebSphere Education IBM

Setting content-based filtering in WebSphere Message Broker Explorer



ContentBasedFiltering

Content Based Filtering: Enabled

Validation (Subscription) Threads: 1

Evaluation (Publication) Threads: 1

1. Right-click the execution group and select **Properties.**
2. Select the **ContentBasedFiltering** tab
3. Select **Enabled** for **Content Based Filtering**.
4. Enter the number of **Validation (Subscription) Threads** and **Evaluation (Publication) Threads**.
5. Click **OK**.
6. Restart the execution group.

Figure 8-16. Setting content-based filtering in WebSphere Message Broker Explorer WM643 / VM6431.0

Notes:

You can configure content based filtering using the WebSphere Message Broker Explorer, as shown in the figure.

Evaluation threads are used to validate content filters against a given publication at publication time. A network with a high number of publications might require the **evaluationThreads** property to be increased (up to a maximum of 32) to handle the workload at publication time.

Validation threads are used to validate syntax of content filters at subscription time. A publish/subscribe network with a high number of subscribers, especially if dynamic subscribers, might require the **validationThreads** property to be increased (up to a maximum of 32) to handle the high throughput of subscription requests.

A valid value for each of these options is a number in the range 1 - 32. Entering any other value causes an error to appear in the **Properties** dialog.

Instructor notes:

Purpose — Show how to configure content-based filtering in WebSphere Message Broker Explorer.

Details —

Additional information —

Transition statement — Next: Managing subscriptions

8.2. Managing subscriptions and topics

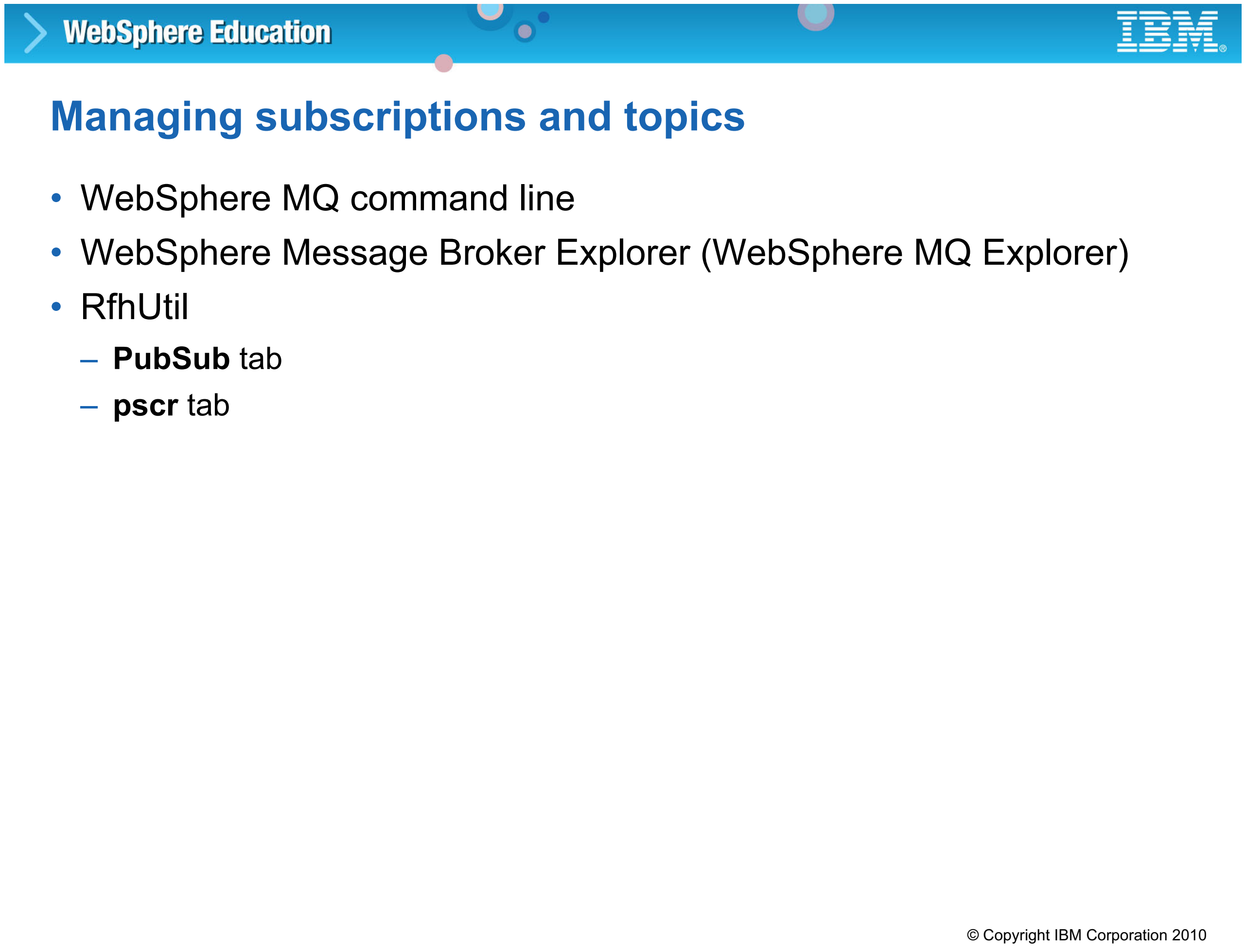
Instructor topic introduction

What students will do — Students will learn how to manage subscriptions and topics using WebSphere MQ, WebSphere Message Broker Explorer, and Rfhutil.

How students will do it — Students will listen to lecture and complete an exercise.

What students will learn — How to manage subscriptions.

How this will help students on their job — Administrators will know how to enable subscriptions for broker and message flow statistics.



WebSphere Education

Managing subscriptions and topics

- WebSphere MQ command line
- WebSphere Message Broker Explorer (WebSphere MQ Explorer)
- RfhUtil
  - **PubSub** tab
  - **pscr** tab

Figure 8-17. Managing subscriptions WM643 / VM6431.0

Notes:

In publish/subscribe, applications are responsible for registering as subscribers to receive published documents. Managing subscriptions can be performed by server applications, using the Application API for WebSphere Message Broker, WebSphere Message Broker Explorer (based on WebSphere MQ Explorer), and RfhUtil.

You will also see some subscriptions to topics beginning with `$SYS`. Generally, these represent the topics shared by the queue manager and its brokers. The format of the various publications is described in the online help system. Understanding the `$SYS` topic structure permits monitoring applications to subscribe to, and share, these system publications.

Instructor notes:

Purpose — Introduce the Subscriptions Query editor view in the toolkit.

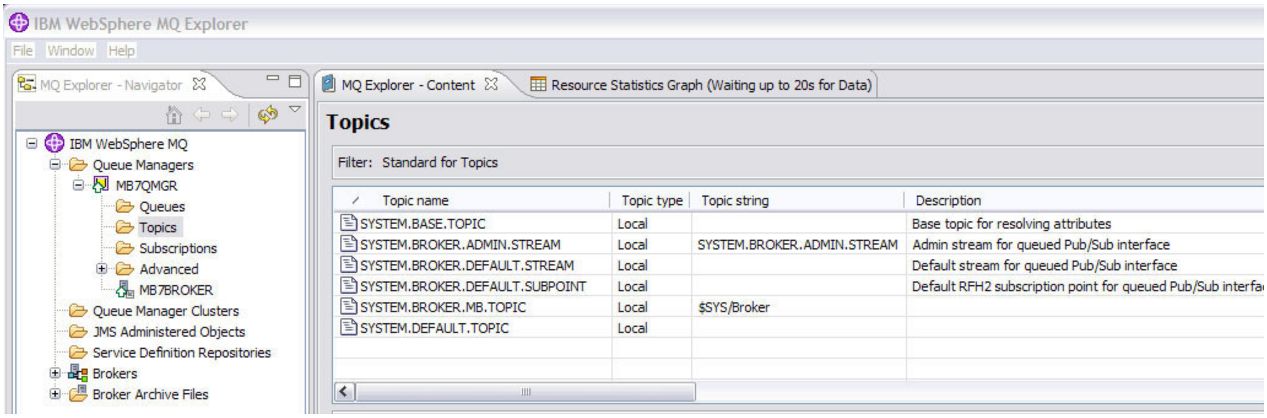
Details — None.

Additional information — None.

Transition statement — Next: Managing topics with WebSphere Message Broker Explorer

WebSphere Education IBM

Managing topics with WebSphere Message Broker Explorer



Topics

Filter: Standard for Topics

Topic name	Topic type	Topic string	Description
SYSTEM.BASE.TOPIC	Local		Base topic for resolving attributes
SYSTEM.BROKER.ADMIN.STREAM	Local	SYSTEM.BROKER.ADMIN.STREAM	Admin stream for queued Pub/Sub interface
SYSTEM.BROKER.DEFAULT.STREAM	Local		Default stream for queued Pub/Sub interface
SYSTEM.BROKER.DEFAULT.SUBPOINT	Local		Default RFH2 subscription point for queued Pub/Sub interfa
SYSTEM.BROKER.MB.TOPIC	Local	$SYS/Broker	
SYSTEM.DEFAULT.TOPIC	Local		

- **Topics** view lists the properties of all the topics on the selected queue manager
  - Create new topics
  - Check topic status
  - Test publication
  - Run tests
  - Manage authority records

Figure 8-18. Managing topics with WebSphere Message Broker Explorer WM643 / VM6431.0

Notes:

The figure lists some of the tasks that can be performed using WebSphere Message Broker Explorer.

Instructor notes:

Purpose — Provide an overview of the publish/subscribe management tasks that can be performed using WebSphere Message Broker Explorer.

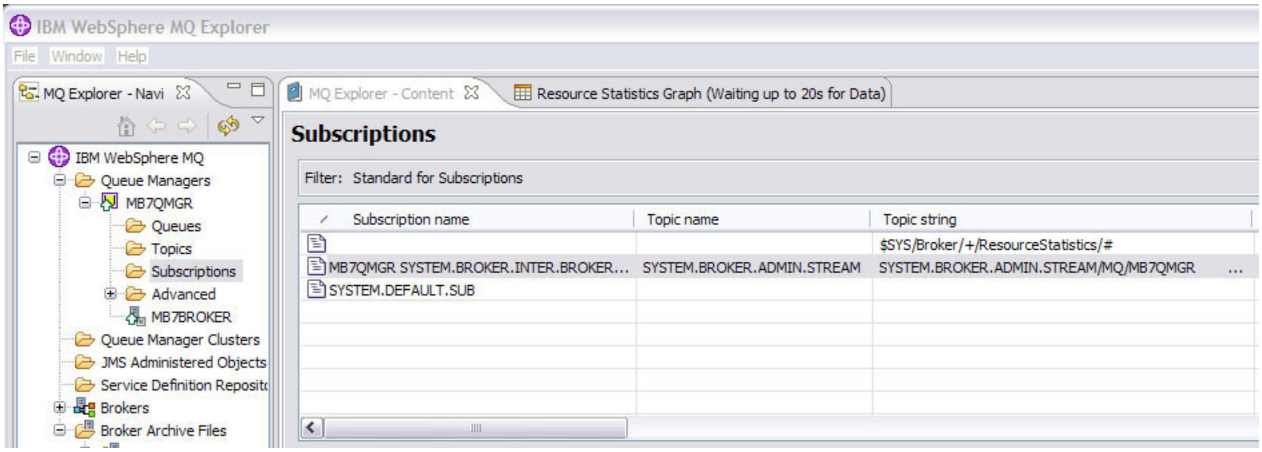
Details —

Additional information —

Transition statement — Next: Managing subscriptions with WebSphere Message Broker Explorer

WebSphere Education

Managing subscriptions with WebSphere Message Broker Explorer



Subscriptions

Filter: Standard for Subscriptions

Subscription name	Topic name	Topic string
		$SYS/Broker/+/ResourceStatistics/#
MB7QMGR SYSTEM.BROKER.INTER.BROKER...	SYSTEM.BROKER.ADMIN.STREAM	SYSTEM.BROKER.ADMIN.STREAM/MQ/MB7QMGR
SYSTEM.DEFAULT.SUB		

- **Subscriptions** view lists the properties of all the subscriptions on the selected queue manager
  - Create new subscriptions
  - Check subscription status
  - Test subscriptions

Figure 8-19. Managing subscriptions with WebSphere Message Broker Explorer WM643 / VM6431.0

Notes:

The figure lists the subscription management tasks that can be performed using WebSphere Message Broker Explorer.

Instructor notes:

Purpose — List the subscription management tasks that can be performed using WebSphere Message Broker Explorer.

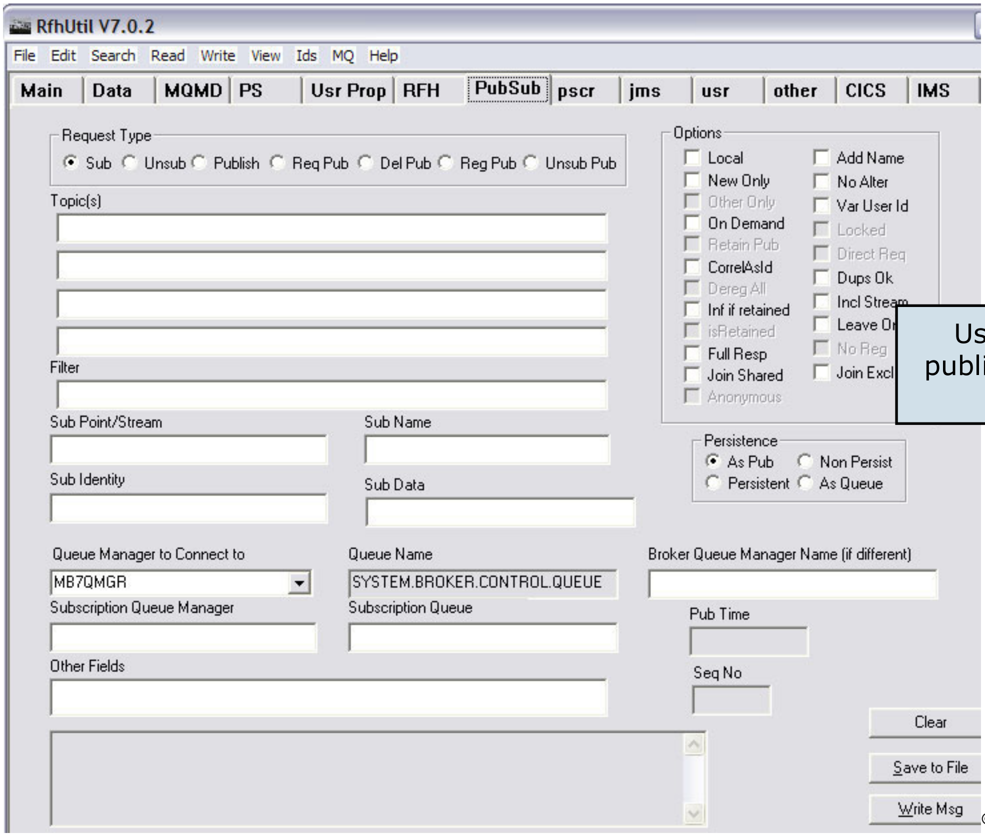
Details —

Additional information —

Transition statement — Next: RfhUtil PubSub tab (1 of 2)

WebSphere Education

RfhUtil PubSub tab (1 of 2)


RfhUtil V7.0.2
File Edit Search Read Write View Ids MQ Help
Main Data MQMD PS Usr Prop RFH PubSub pscr jms usr other CICS IMS
Clear
Save to File
Write Msg

Request Type: ● Sub ○ Unsub ○ Publish ○ Req Pub ○ Del Pub ○ Reg Pub ○ Unsub Pub

Topic(s)

Filter

Sub Point/Stream

Sub Name

Sub Identity

Sub Data

Options: ☐ Local ☐ New Only ☐ Other Only ☐ On Demand ☐ Retain Pub ☐ CorrelAsId ☐ Dereg All ☐ Inf if retained ☐ isRetained ☐ Full Resp ☐ Join Shared ☐ Anonymous ☐ Add Name ☐ No Alter ☐ Var User Id ☐ Locked ☐ Direct Req ☐ Dups Ok ☐ Incl Stream ☐ Leave O ☐ No Reg ☐ Join Excl

Persistence: ● As Pub ○ Non Persist ○ Persistent ○ As Queue

Queue Manager to Connect to: MB7QMGR

Queue Name: SYSTEM.BROKER.CONTROL.QUEUE

Broker Queue Manager Name (if different)

Subscription Queue Manager

Subscription Queue

Pub Time

Seq No

Other Fields

Used to generate publish and subscribe requests

Figure 8-20. RfhUtil PubSub tab (1 of 2) WM643 / VM6431.0

Notes:

The RfhUtil **PubSub** tab is used to generate publish and subscribe requests. This tab will allow most but not all publish/subscribe commands to be entered or displayed. Different publish and subscribe brokers might not support all of the options that are available.

The **Request Type** specifies the type of request. After the request type is selected, the appropriate input fields for that type of request are available.

The key fields on this tab are described on the next page.

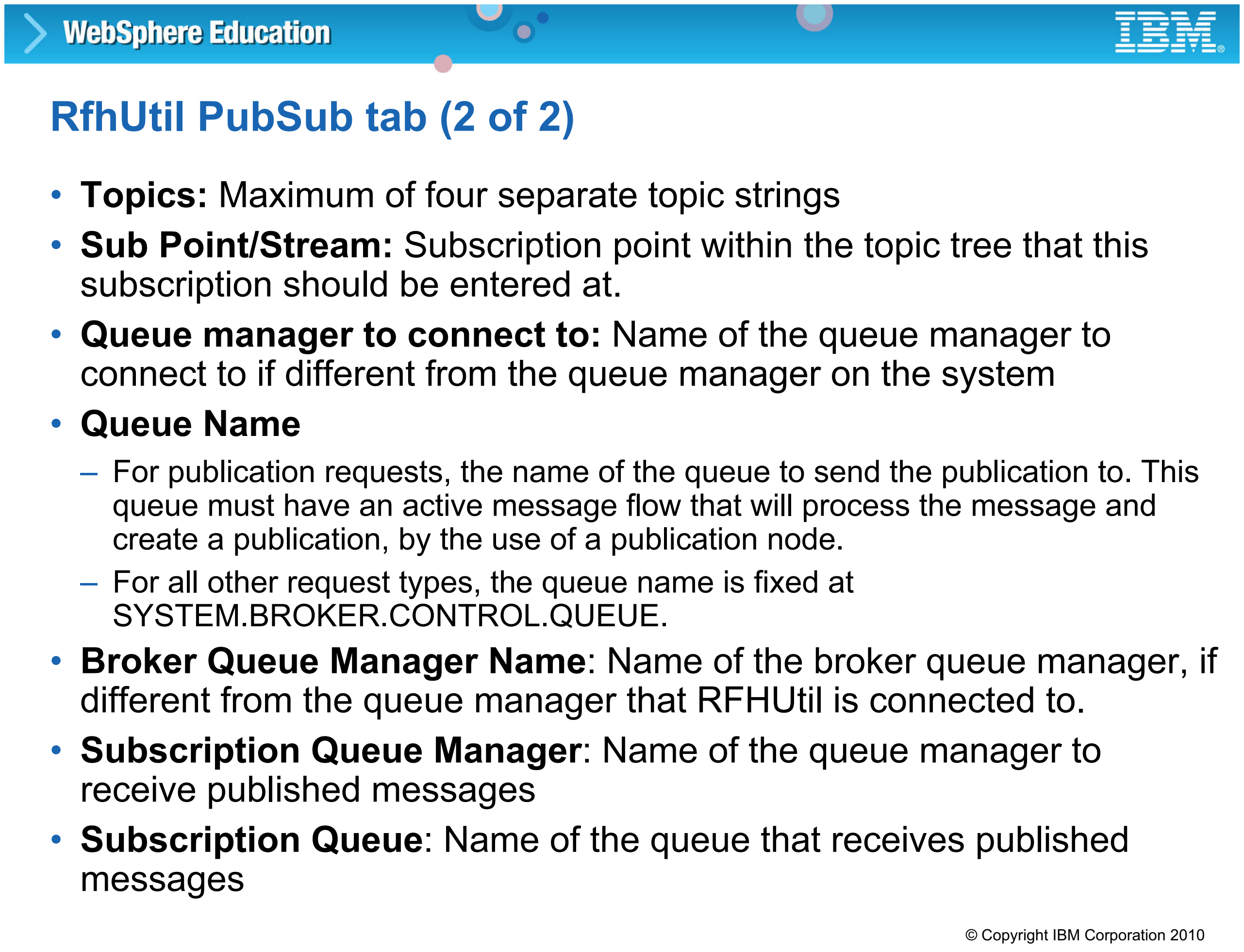
Instructor notes:

Purpose — Show the RfhUtil PubSub tab.

Details — Details are on the next page.

Additional information — None.

Transition statement — None.



WebSphere Education

RfhUtil PubSub tab (2 of 2)

- **Topics:** Maximum of four separate topic strings
- **Sub Point/Stream:** Subscription point within the topic tree that this subscription should be entered at.
- **Queue manager to connect to:** Name of the queue manager to connect to if different from the queue manager on the system
- **Queue Name**
  - For publication requests, the name of the queue to send the publication to. This queue must have an active message flow that will process the message and create a publication, by the use of a publication node.
  - For all other request types, the queue name is fixed at SYSTEM.BROKER.CONTROL.QUEUE.
- **Broker Queue Manager Name**: Name of the broker queue manager, if different from the queue manager that RFHUtil is connected to.
- **Subscription Queue Manager**: Name of the queue manager to receive published messages
- **Subscription Queue**: Name of the queue that receives published messages

Figure 8-21. RfhUtil PubSub tab (2 of 2) WM643 / VM6431.0

Notes:

The figure provides a brief description of some of the fields on the RfhUtil **PubSub** tab.

See the *IBM WebSphere Message Broker Display Test and Performance utilities* guide for more information.

Instructor notes:

Purpose — Describe some of the fields on the PubSub tab.

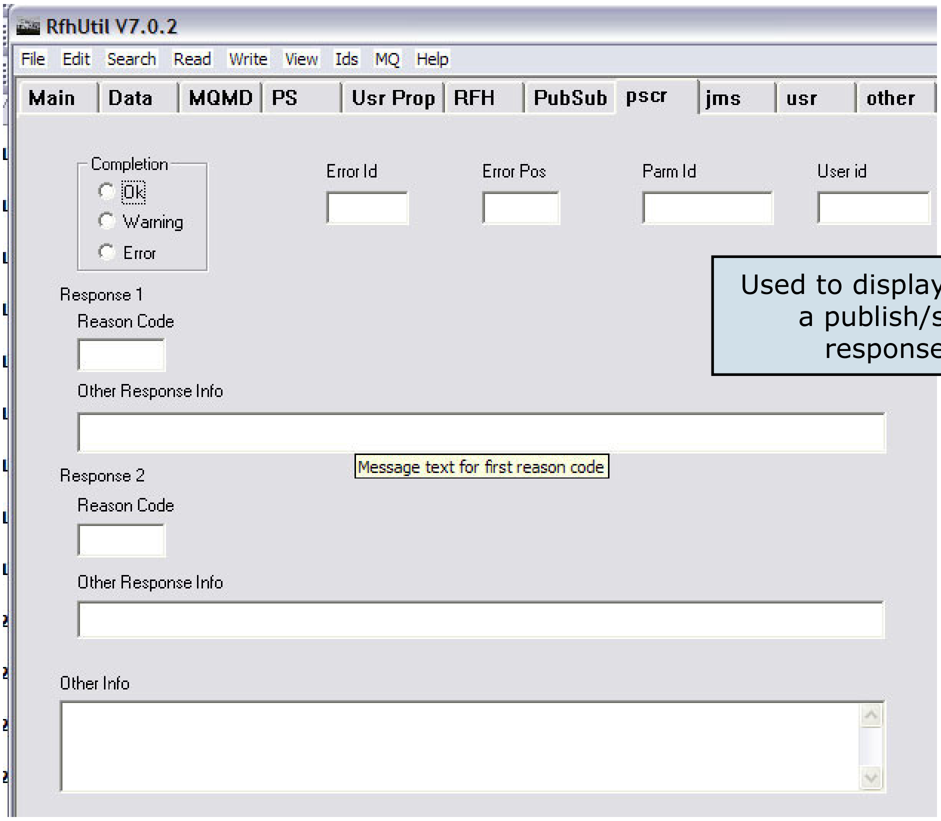
Details — None.

Additional information — None.

Transition statement — Next is the **pscr** tab.

WebSphere Education

RfhUtil pscr tab



Used to display fields within a publish/subscribe response folder

Figure 8-22. RfhUtil pscr tab WM643 / VM6431.0

Notes:

The **pscr** tab is used to display fields within a publish/subscribe response folder.

This tab is populated with the contents of a pscr (publish/subscribe response) folder in an RFH 2 header that is found in a message or file that is read by the RfhUtil program. It is also used to display fields found in a version 1 header that are contained in broker responses. A maximum of two responses is displayed.

Responses are generally sent by the broker and not created by an application.

Instructor notes:

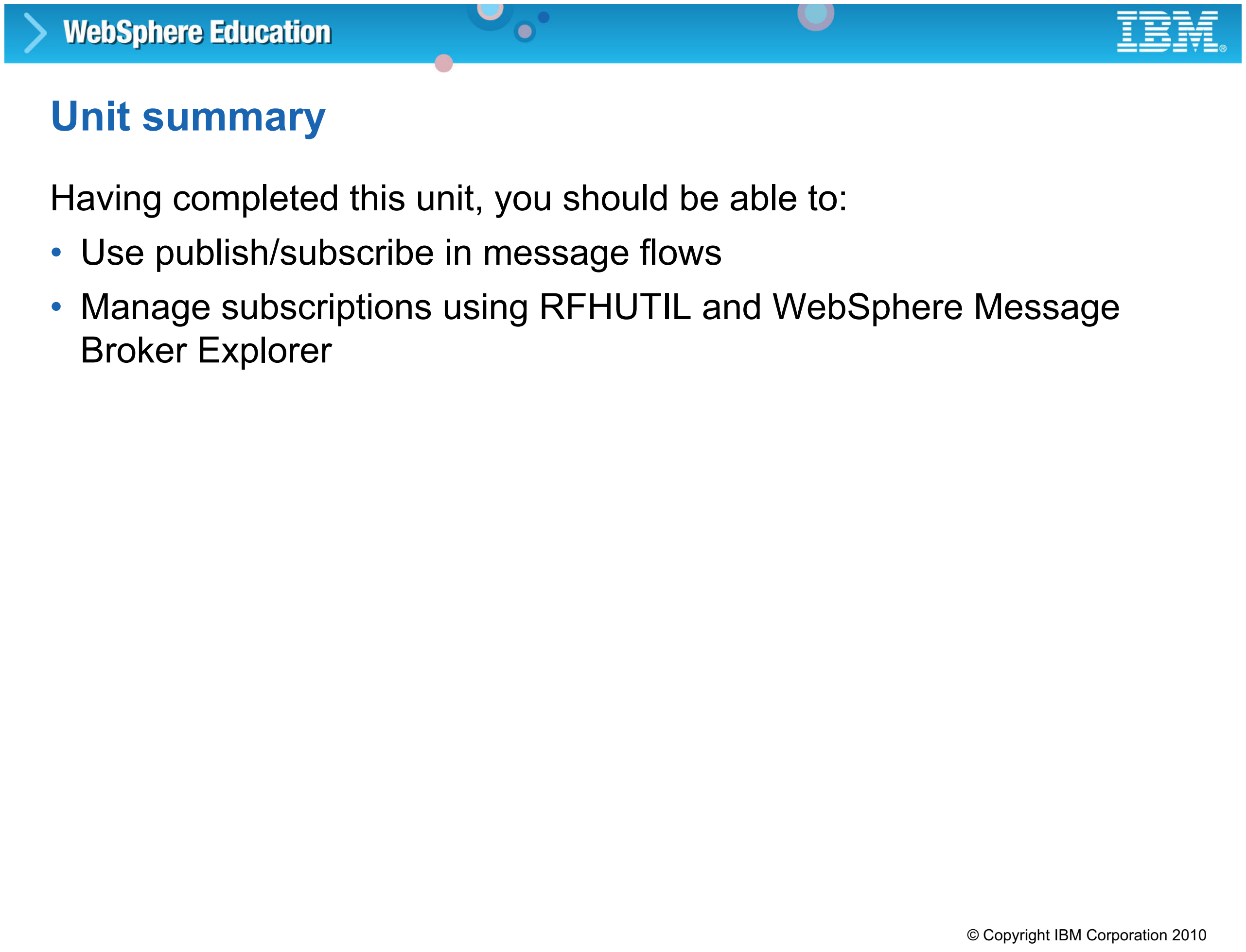
Purpose — Provide an overview of the RfhUtil **pscr** tab.

Details — The **Error ID**, **Error Pos**, Parm ID and **user ID** fields are only used with a version 1 RFH. The Other Info field can contain fields that are not recognized in a version 1 RFH.

If any entries are found in the variable section of an RFH version 1 header that are not recognized by the program, the information is added to the **Other Info** field and the **Resp** check box is selected on the RFH tab.

Additional information — None

Transition statement — Summarize the concepts covered in this unit.



WebSphere Education

Unit summary

Having completed this unit, you should be able to:

- Use publish/subscribe in message flows
- Manage subscriptions using RFHUTIL and WebSphere Message Broker Explorer

Figure 8-23. Unit summary WM643 / VM6431.0

Notes:

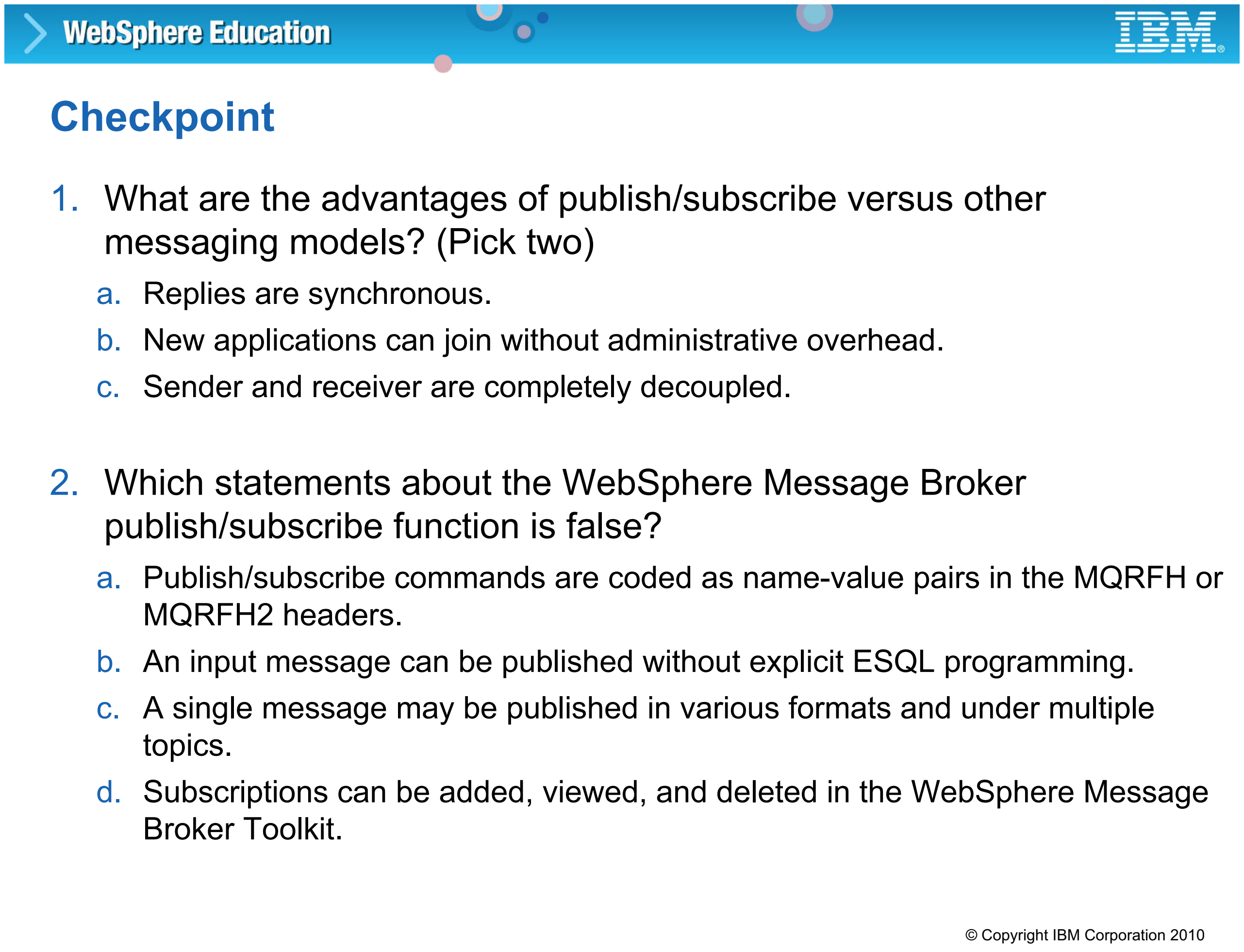
Instructor notes:

Purpose — Summarize the unit.

Details — None.

Additional information — None.

Transition statement — After the exercise, the next unit will describe using web services with WebSphere Message Broker.



WebSphere Education

Checkpoint

1. What are the advantages of publish/subscribe versus other messaging models? (Pick two)
   a. Replies are synchronous.
   b. New applications can join without administrative overhead.
   c. Sender and receiver are completely decoupled.

2. Which statements about the WebSphere Message Broker publish/subscribe function is false?
   a. Publish/subscribe commands are coded as name-value pairs in the MQRFH or MQRFH2 headers.
   b. An input message can be published without explicit ESQL programming.
   c. A single message may be published in various formats and under multiple topics.
   d. Subscriptions can be added, viewed, and deleted in the WebSphere Message Broker Toolkit.

Figure 8-24. Checkpoint

WM643 / VM6431.0

Notes:

Write your answers here:

1.

2.

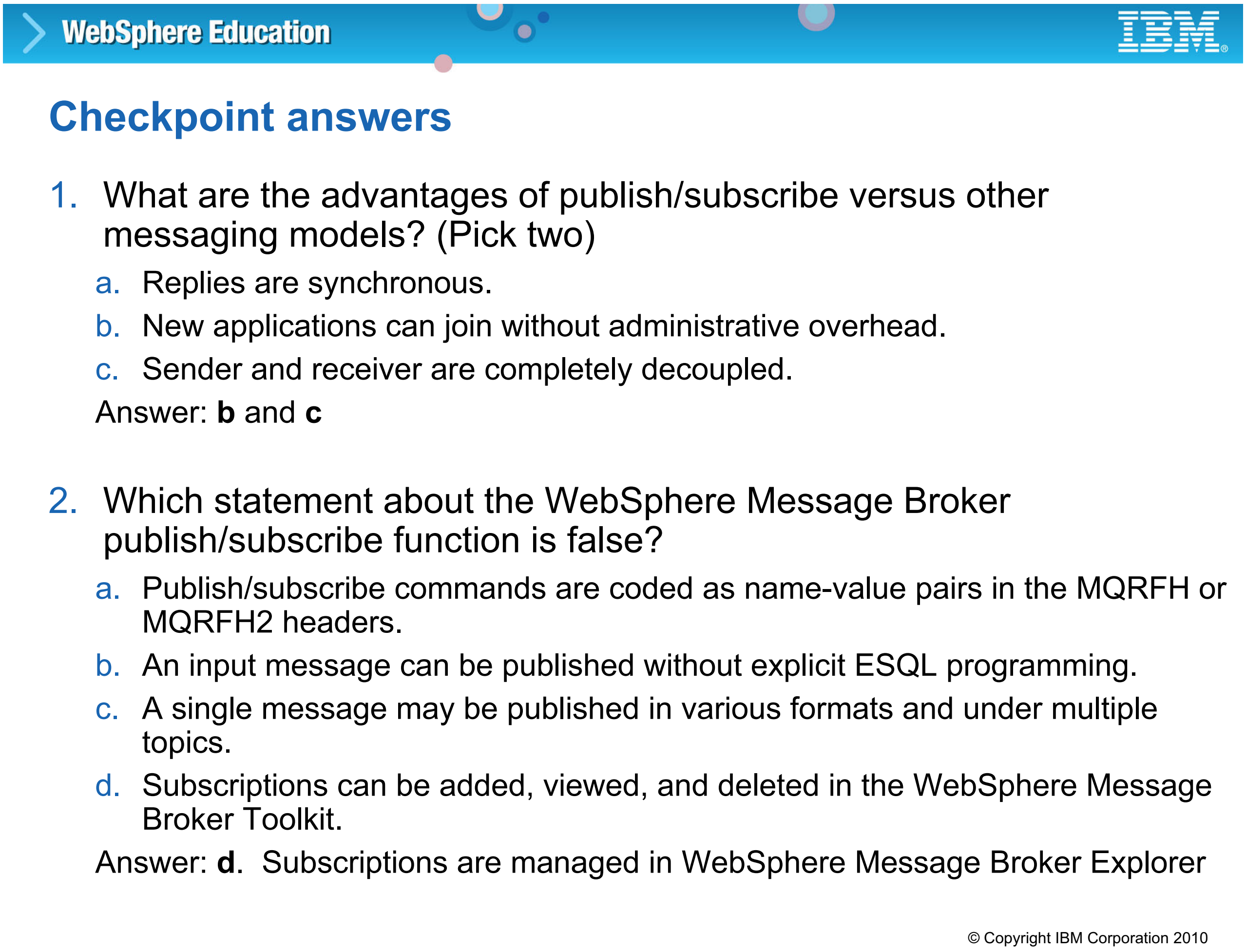
Instructor notes:

Purpose — Let students reflect on the material presented and answer the questions. Discussion to follow.

Details —

Additional information — Discuss the answers, covering both the correct and the incorrect responses.

Transition statement — Next: Checkpoint answers



WebSphere Education

Checkpoint answers

1. What are the advantages of publish/subscribe versus other messaging models? (Pick two)
   a. Replies are synchronous.
   b. New applications can join without administrative overhead.
   c. Sender and receiver are completely decoupled.

   Answer: **b** and **c**

2. Which statement about the WebSphere Message Broker publish/subscribe function is false?
   a. Publish/subscribe commands are coded as name-value pairs in the MQRFH or MQRFH2 headers.
   b. An input message can be published without explicit ESQL programming.
   c. A single message may be published in various formats and under multiple topics.
   d. Subscriptions can be added, viewed, and deleted in the WebSphere Message Broker Toolkit.

   Answer: **d**. Subscriptions are managed in WebSphere Message Broker Explorer

Figure 8-25. Checkpoint answers WM643 / VM6431.0

Notes:

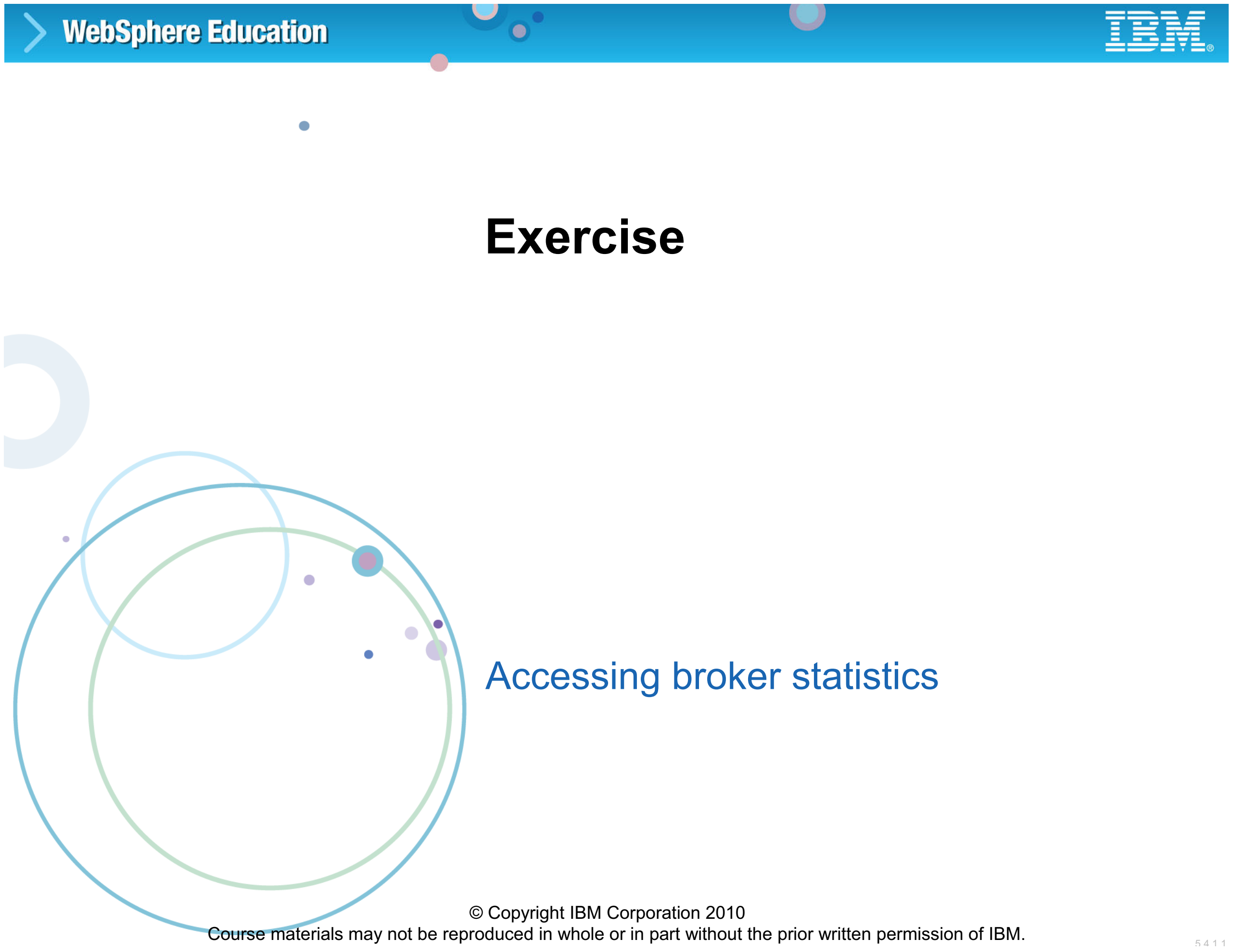
Instructor notes:

Purpose — Review the checkpoint answers

Details —

Additional information —

Transition statement — Next: Exercise 7: Accessing broker statistics



WebSphere Education

Exercise

Accessing broker statistics

5411

Figure 8-26. Exercise 7: Accessing broker statistics WM643 / VM6431.0

Notes:

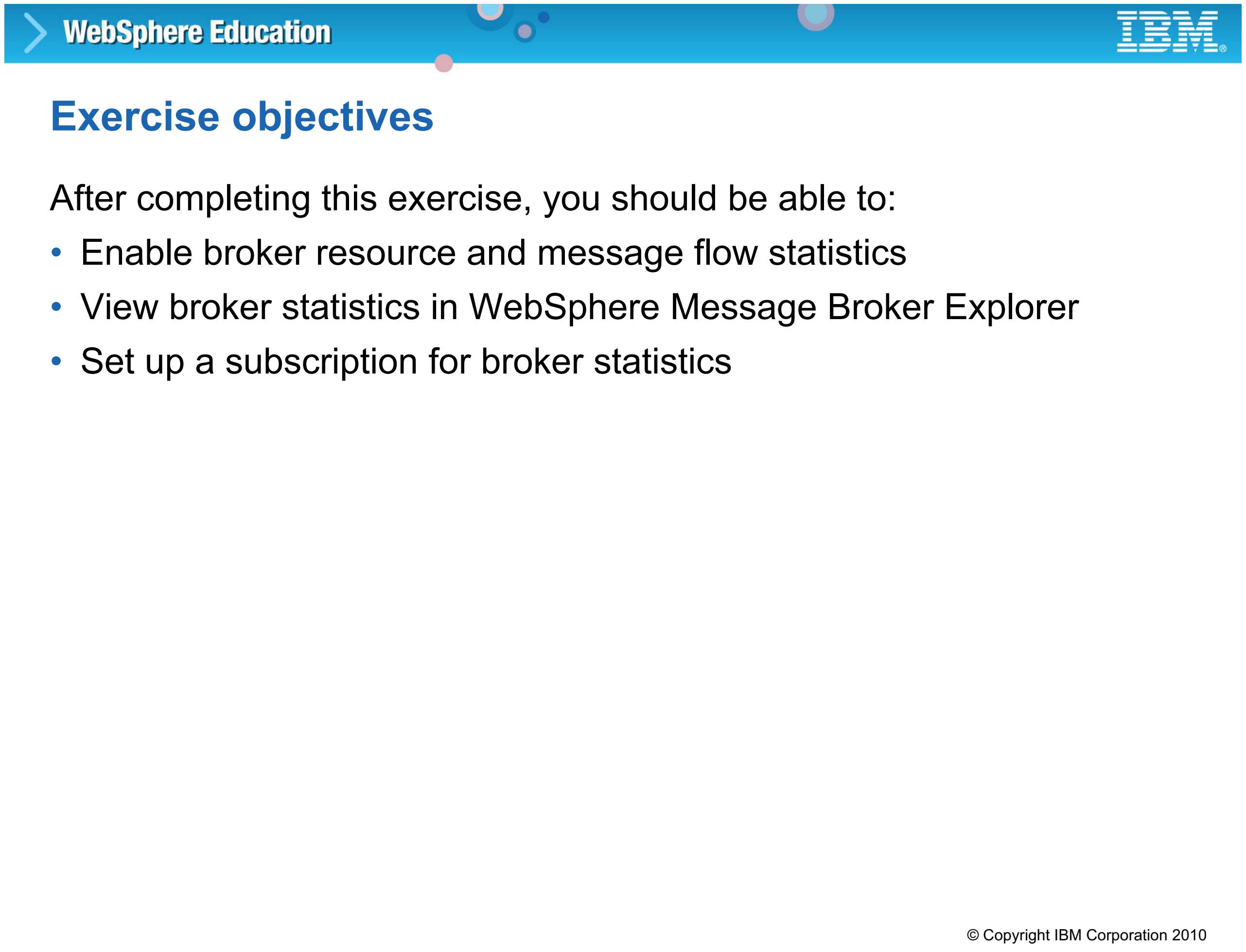
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement — Next: Exercise objectives


WebSphere Education

Exercise objectives

After completing this exercise, you should be able to:

- Enable broker resource and message flow statistics
- View broker statistics in WebSphere Message Broker Explorer
- Set up a subscription for broker statistics

Figure 8-27. Exercise objectives WM643 / VM6431.0

Notes:

Proceed to the *Student Exercise Guide* and complete Exercise 7.

Instructor notes:

Purpose — Have students complete Exercise 7.

Details —

Additional information —

Transition statement —

Unit 9. Administering web services and JMS

Estimated time

01:30 lecture, 01:00 exercise

What this unit is about

This unit describes how to administer message flows that use HTTP and HTTPS to provide web services. This unit also describes providing security on SOAP nodes using policy sets and policy set bindings, and administration tasks for JMS nodes.

What you should be able to do

After completing this unit, you should be able to:

- Describe the implementation of the HTTP and HTTPS nodes in WebSphere Message Broker
- Explain the implementation of Secure Sockets Layer (SSL) in support of HTTP nodes
- Create a policy set
- Use capacity planning methods when implementing HTTP nodes
- Configure a Java Message Service (JMS) provider for use with the JMS nodes

How you will check your progress

Accountability:

- Checkpoint questions
- Exercise 8, Implementing web services and web services security

References

`www.ibm.com/redbooks`

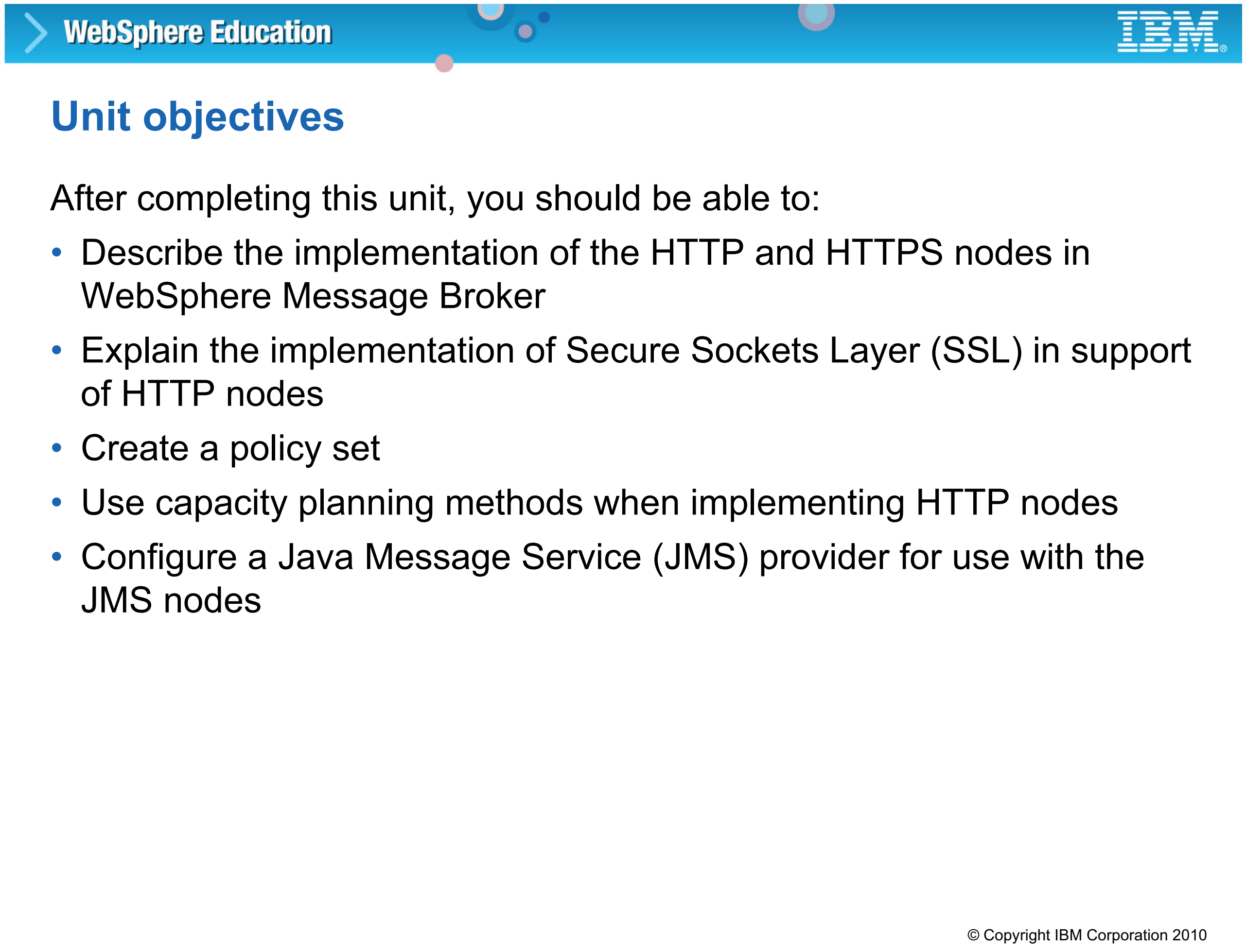
Using Web Services for Business Integration
SG24-6583

`www.ibm.com/software/integration/wbimessagebroker/library/`

WebSphere Message Broker stand-alone online help system

`http://www.ibm.com/support/docview.wss?uid=swg24009370`

SupportPac IE01: WebSphere Message Broker Proxy Servlet for HTTP nodes



WebSphere Education

Unit objectives

After completing this unit, you should be able to:

- Describe the implementation of the HTTP and HTTPS nodes in WebSphere Message Broker
- Explain the implementation of Secure Sockets Layer (SSL) in support of HTTP nodes
- Create a policy set
- Use capacity planning methods when implementing HTTP nodes
- Configure a Java Message Service (JMS) provider for use with the JMS nodes

Figure 9-1. Unit objectives

WM643 / VM6431.0

Notes:

Instructor notes:

Purpose — List the unit objectives.

Details —

Additional information —

Transition statement — This first topic is an overview of web services and basic terminology.

9.1. Introduction to web services

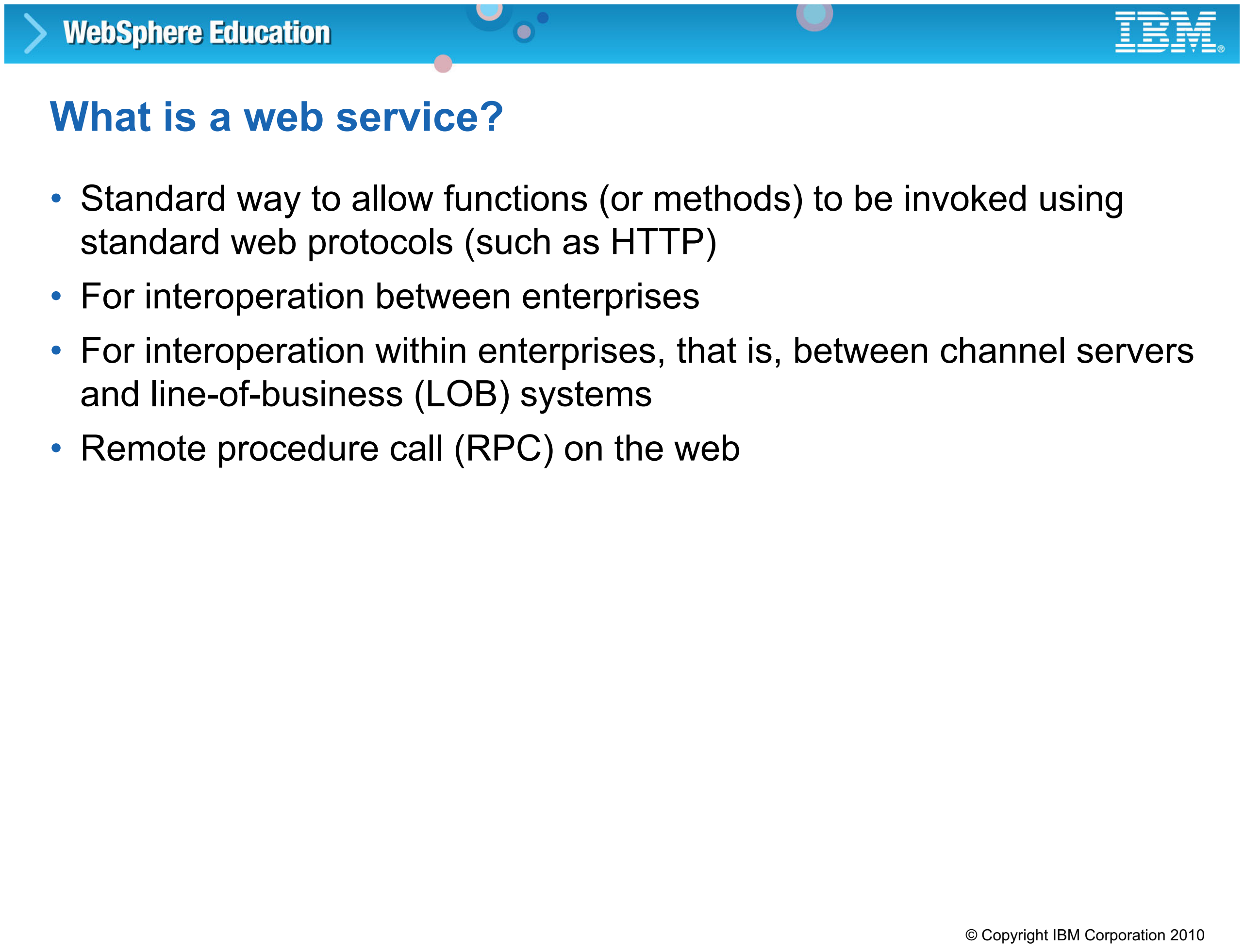
Instructor topic introduction

What students will do — Learn about web services. This first topic can be considered ***optional*** and only needs to be covered if students express an interest.

How students will do it — Listen to the lecture and participate in classroom discussion. Exercise 8 will permit students to configure an HTTP node and exercise a web service supported by WebSphere Message Broker.

What students will learn — How a web service can be supplied by, and invoked by, a message flow.

How this will help students on their job — Web services offer a vast server potential that should be considered as an alternative to a comparable in-house function.



WebSphere Education

What is a web service?

- Standard way to allow functions (or methods) to be invoked using standard web protocols (such as HTTP)
- For interoperation between enterprises
- For interoperation within enterprises, that is, between channel servers and line-of-business (LOB) systems
- Remote procedure call (RPC) on the web

Figure 9-2. What is a web service? WM643 / VM6431.0

Notes:

The key to web services is a common program-to-program communications model, built on existing and emerging standards such as HTTP, Extensible Markup Language (XML), SOAP, Web Services Description Language (WSDL), and Universal Description, Discovery, and Integration (UDDI).

A web service is an *interface* that describes a collection of operations that are network accessible through standardized XML messaging. A web service is described using a standard, formal XML notion, called its *service description*. It covers all the details necessary to interact with the service, including message formats (which detail the operations), transport protocols, and network location. The interface hides the implementation details of the service, allowing it to be used independently of the hardware or software platform on which it is implemented and also independently of the programming language in which it is written. This allows and encourages web service applications to be loosely coupled, component-oriented, cross-technology implementations.

Web services fulfill a specific task or a set of tasks. They can be used alone or with other web services to carry out a complex aggregation or a business transaction.

Instructor notes:

Purpose — A short level set.

Details — Web services can be confusing. When the term *web services* is used, it can mean different things to different people. This situation results from conflicting and overlapping standards, emerging standards, and numerous implementations that lay a claim, but are deficient when it comes to interoperability.

Most of the concepts of web services are not new or unique. Almost all have been used before, but have not been standardized in an easily accessible way.

Web services can be defined as *a standard way to allow functions (or methods) to be invoked using standard web protocols* (such as HTTP or JMS). This has always been possible using CGI-BIN scripts, but implementation was often complicated and proprietary. Using HTTP allows universal access (firewalls let it through). XML (with XML schema) provides platform independence.

Remote Procedure Call (RPC) usage is the most common. SOAP provides a format for encoding RPC requests and responses. Special message formats for errors (called *faults*) are defined. In all, a SOAP request can be sent (using many different transports, such as HTTP, JMS, and so on) and successfully received.

Web Services Invocation Framework: this is an IBM-developed set of libraries for enabling RPC use by programmers. It understands WSDL, handles encoding and marshaling of data, and hides transport complexity. No attempt is made to provide reliable messaging or transactional support. The *once-and-once-only* style of messaging is not supported in web services. When a transport link goes down in the middle of a call, programmers must work out for themselves what state cleanup is required. Standards are being developed to better describe ways to resolve these situations.

Not all web services conform to the SOAP/HTTP model. There are various other RPC schemes that use XML and HTTP:

- Proprietary formats: if both applications are maintained by the same organization, then the format can be anything, even RPC over HTML using email.
- JAX RPC: a Java standard.

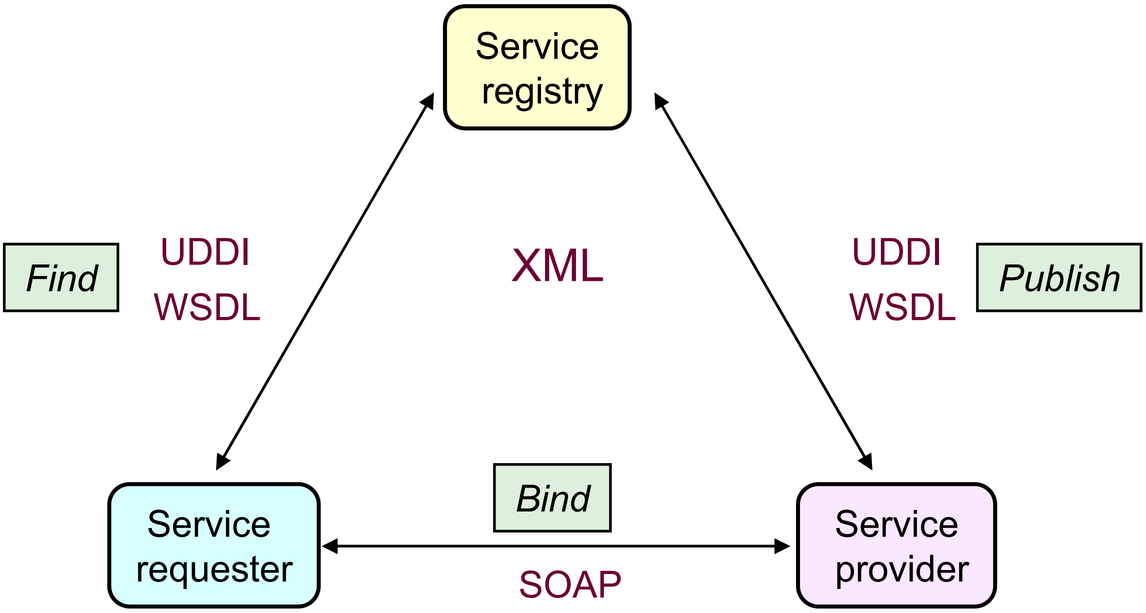
There is a school of thought within IBM which believes that any business-related function or method call can be considered a web service. This view is based on method calls made on local Java objects, which includes almost everything.

Additional information — None.

Transition statement — Presuming you know what a web service is, can you determine what it provides?

WebSphere Education

Web services context



Service registry

Find

UDDI

WSDL

XML

UDDI

WSDL

Publish

Bind

Service requester

SOAP

Service provider

Figure 9-3. Web services context

WM643 / VM6431.0

Notes:

Assume that an application needs to discover the location and details of a business service. It issues a request to search the registry using UDDI. The UDDI specification covers both the data model of the data held in the registry and the API used to search it. The reply is a WSDL message, describing various aspects of the service, including where it is, the protocol used to access it, how to invoke it, and the format of the request and reply messages. All data that flows will be in XML, in a format defined as part of the SOAP standard.

During early implementations, this scenario is likely to be implemented within an organization (intranet rather than Internet). This would certainly be a powerful method of publishing business services; for example, during a merger, it could be an invaluable tool to ease the integration of the IT processes of the organizations involved.

Instructor notes:

Purpose — Identify the participant roles in web services.

Details — Roles in a web services architecture:

- Service provider: From a business perspective, this is the owner of the service. From an architectural perspective, this is the platform that hosts access to the service.
- Service requester: From a business perspective, this is the business that requires certain functions to be satisfied. From an architectural perspective, this is the application that is looking for and invoking or initiating an interaction with a service.
- Service registry: This is a searchable registry where service providers publish their service descriptions.

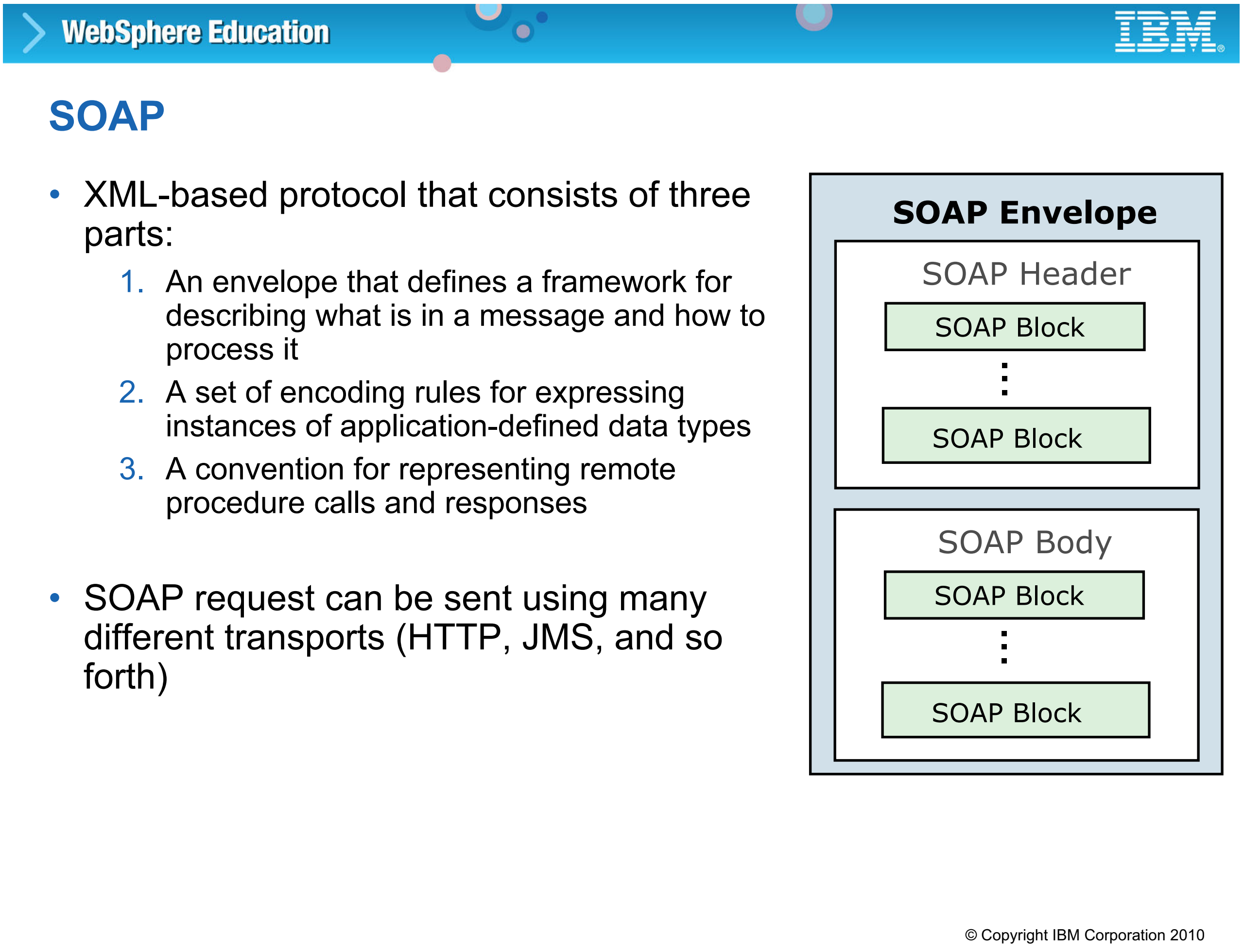
For an application to take advantage of web services, three behaviors must take place: publication of service descriptions, lookup or finding service descriptions, and binding or invoking services based on the service description. These behaviors can occur singly or iteratively. In detail, these operations are:

- Publish: To be accessible, a service description must be published so that the service requester can find it. Where it is published can vary depending upon the requirements of the application.
- Find: In the find operation, the service requester retrieves a service description directly or queries the service registry for the type of service required. The find operation can be involved in two different life cycle phases for the service requester: at design time, to retrieve the service's interface description for program development; and at run time, to retrieve the service's binding and location description for invocation.
- Bind: Eventually, a service must be invoked. In the bind operation, the service requester invokes or initiates an interaction with the service at run time using the binding details in the service description to locate, contact, and invoke the service.

The web services architecture explains how to instantiate the elements and implement the operations in an interoperable manner.

Additional information — None.

Transition statement — Standards abound with web services.



WebSphere Education

SOAP

- XML-based protocol that consists of three parts:
  1. An envelope that defines a framework for describing what is in a message and how to process it
  2. A set of encoding rules for expressing instances of application-defined data types
  3. A convention for representing remote procedure calls and responses
- SOAP request can be sent using many different transports (HTTP, JMS, and so forth)

Figure 9-4. SOAP

WM643 / VM6431.0

Notes:

SOAP is a simple and lightweight XML-based mechanism for exchanging structured data between network applications.

SOAP consists of three parts:

- an envelope that defines a framework for describing what is in a message
- a set of encoding rules for expressing instances of application-defined data types
- a convention for representing remote procedure calls (RPCs) and responses.
- SOAP can be used in combination with, or re-enveloped by, various network protocols such as HTTP, SMTP, FTP, RMI over IIOP, or WebSphere MQ.

Instructor notes:

Purpose — Provide an overview of SOAP and SOAP message structure

Details —

Additional information —

Transition statement — The next key component to web services is the WSDL.

WebSphere Education

Web services description language (WSDL)

- XML-based metadata document that defines Web service and how to access it
- Allows programs to discover services and call them without administrative setup
- Provides functional and technical information
  - **Type**: Container for data type definitions used in messages
  - **Message**: Typed definition of data that is input to and output from operation
  - **Port type**: Set of operations supported by one or more endpoints
  - **Operation**: Description of an action supported by the service
  - **Service**: Collection of related endpoints (ports)
  - **Port**: One connection point; consists of binding plus network address
  - **Binding**: Protocol and data format specifications for a port type

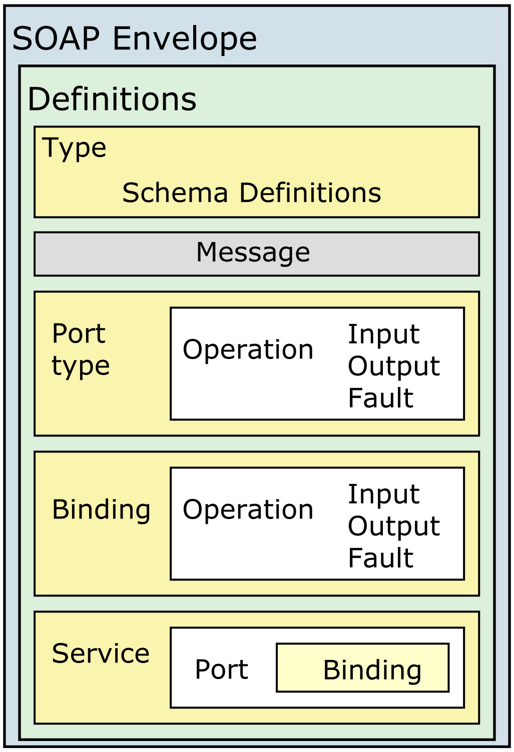


Figure 9-5. Web services description language (WSDL) WM643 / VM6431.0

Notes:

WSDL is an XML format for describing network services as a set of endpoints, operating on messages containing either document-oriented or procedure-oriented information.

The operations and messages are described abstractly, and then bound to a concrete network protocol and message format, to define an endpoint. Related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate.

Instructor notes:

Purpose — Provide an overview of a WSDL.

Details — The WSDL document defines a service in terms of a number of ports (WSDL V1.1) or endpoints (WSDL V1.2). These essentially define where the service is available.

Each named port also defines a mechanism for accessing it, called a *binding*. There is a separate binding for each protocol supported, for example, SOAP over HTTP. Each binding refers to a named portType (WSDL V1.1) or interface (WSDL V1.2)

A binding defines the message wire format and transport details. A portType/interface is the logical interface to the web service. Both binding and portType/interface define operations.

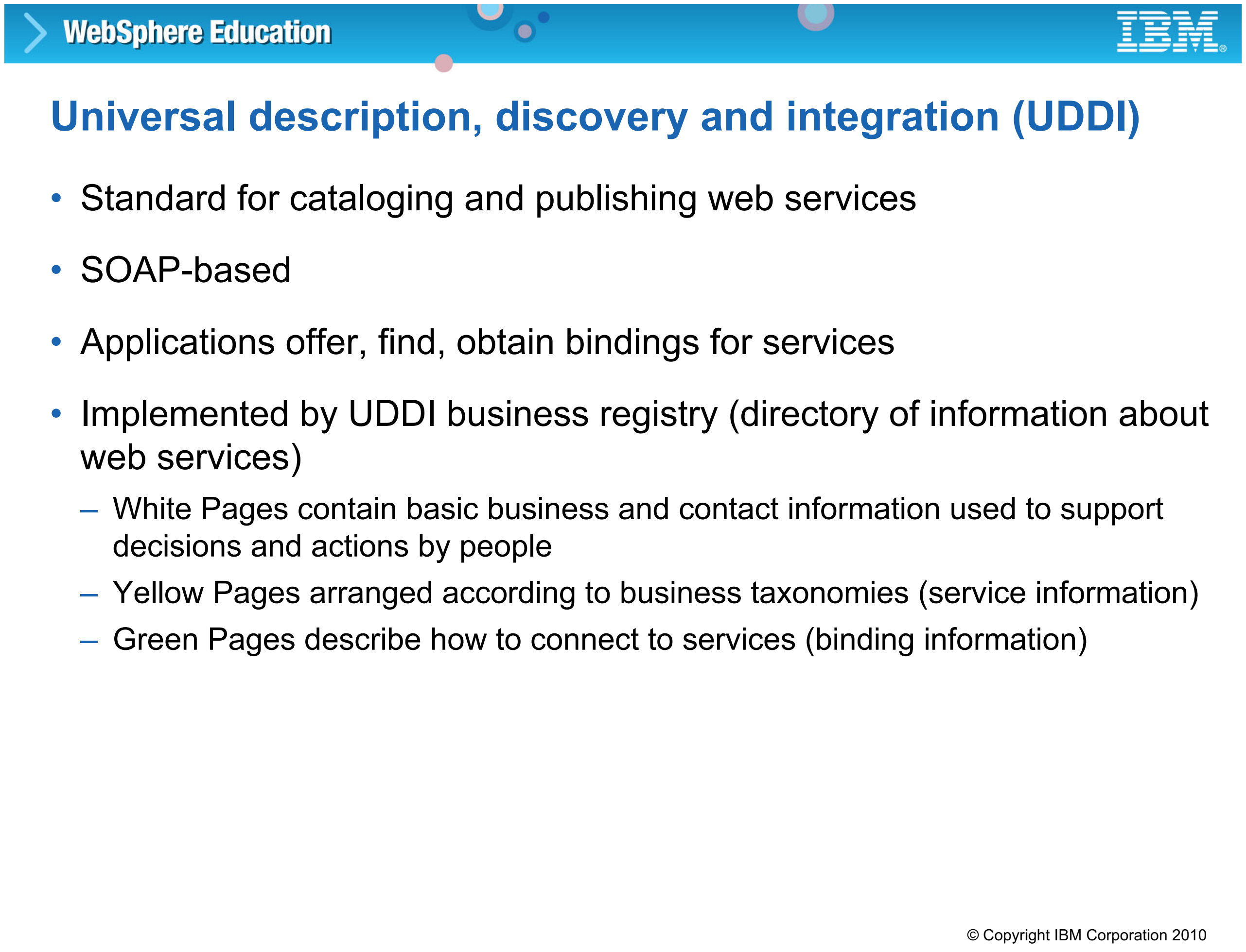
Each operation comprises input and output elements defined in terms of messages or message parts.

The message elements define a logical message in terms of one or more parts. (Each part might correspond to a parameter on a method call.)

A part is defined either as an XML schema element or as an XML schema type.

Additional information — None.

Transition statement — The final key component is the UDDI.



WebSphere Education

Universal description, discovery and integration (UDDI)

- Standard for cataloging and publishing web services
- SOAP-based
- Applications offer, find, obtain bindings for services
- Implemented by UDDI business registry (directory of information about web services)
  - White Pages contain basic business and contact information used to support decisions and actions by people
  - Yellow Pages arranged according to business taxonomies (service information)
  - Green Pages describe how to connect to services (binding information)

Figure 9-6. Universal description, discovery and integration (UDDI) WM643 / VM6431.0

Notes:

UDDI is an XML-based registry for businesses worldwide to list themselves on the Internet. Its ultimate goal is to streamline online transactions by enabling companies to find one another on the web and make their systems interoperable for e-commerce. UDDI is often compared to a telephone book's white and yellow pages. It allows businesses to list themselves by name, product, location, or the web services they offer.

Instructor notes:

Purpose — Provide an overview of UDDI

Details — Service requesters find services and obtain binding information (in the service descriptions) for services during development (for static binding) or during execution (for dynamic binding). For statically bound service requesters, the service registry is an optional role in the architecture (as an alternative, a service provider can send the description directly to service requesters).

Likewise, service requesters can obtain a service description from other sources besides a service registry, such as a local file, FTP site, website, Advertisement and Discovery of Services (ADS), or Discovery of Web Services (DISCO)

Additional information — None.

Transition statement — Next, you will see how WebSphere Message Broker message flows can provide web services or act as a client.

9.2. Implementing WebSphere Message Broker in a web services environment

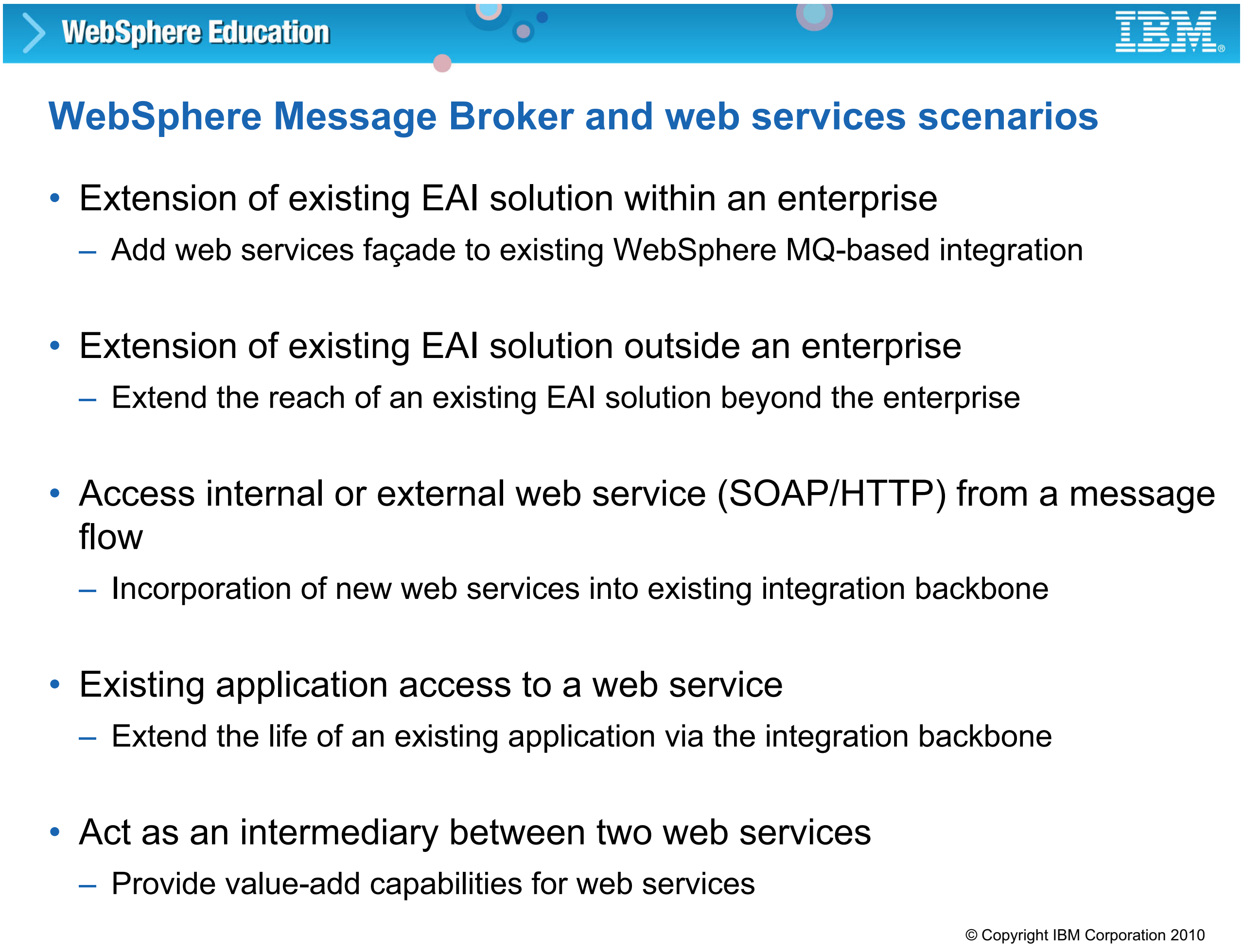
Instructor topic introduction

What students will do — Learn that message flows can function as a web service, or call a web service, anywhere in the network the broker is a part of.

How students will do it — Listen to the lecture and participate in classroom discussion. Exercise 8 will permit students to test a web service message flow.

What students will learn — WebSphere Message Broker can host web services or serve as an intermediary in calling a web service.

How this will help students on their job — A program function that can be called, rather than developed or purchased, represents a viable alternative to application design.



WebSphere Education

WebSphere Message Broker and web services scenarios

- Extension of existing EAI solution within an enterprise
  - Add web services façade to existing WebSphere MQ-based integration
- Extension of existing EAI solution outside an enterprise
  - Extend the reach of an existing EAI solution beyond the enterprise
- Access internal or external web service (SOAP/HTTP) from a message flow
  - Incorporation of new web services into existing integration backbone
- Existing application access to a web service
  - Extend the life of an existing application via the integration backbone
- Act as an intermediary between two web services
  - Provide value-add capabilities for web services

Figure 9-7. WebSphere Message Broker and web services scenarios WM643 / VM6431.0

Notes:

There are a number of scenarios where WebSphere Message Broker can be implemented with web services.

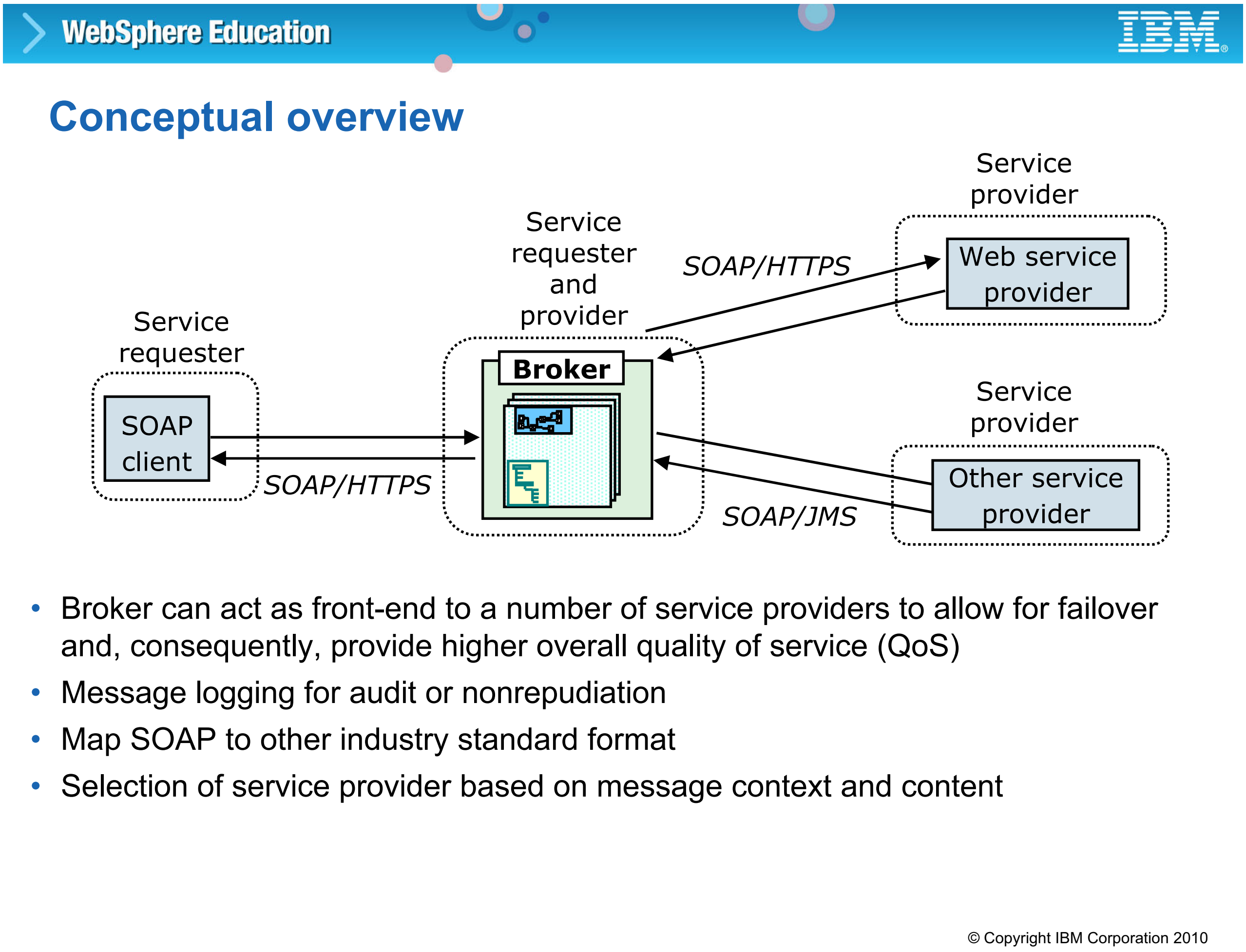
Instructor notes:

Purpose — Highlight the options that message flows and corresponding broker support offer.

Details — None.

Additional information — None.

Transition statement — Look at the concept, graphically.



WebSphere Education

Conceptual overview

- Broker can act as front-end to a number of service providers to allow for failover and, consequently, provide higher overall quality of service (QoS)
- Message logging for audit or nonrepudiation
- Map SOAP to other industry standard format
- Selection of service provider based on message context and content

Figure 9-8. Conceptual overview WM643 / VM6431.0

Notes:

It is not an accident that the message flow is shown in the center of the diagram since it can serve as the interface between client and service provider.

WebSphere Message Broker is a convenient central point for web services brokering. You can wrap legacy services or access web services from legacy applications. Broker can act as a SOAP intermediary, providing first point of contact functions like hiding or changing service implementation, transformation between WSDL definitions, monitoring, data warehousing, and auditing. Normal broker strengths are applied to and enhance web services.

A message flow can be requester or client. Usually, an MQInput node drives a flow that calls a web service (over HTTP or JMS bindings), the result of which is used to augment the message sent back using an MQOutput node.

A message flow can also be the service provider, allowing web service clients to invoke an existing message flow. In that case, you would replace MQInput and MQOutput nodes with HTTP or JMS nodes and accept SOAP messages.

Instructor notes:

Purpose — Position the broker and message flow from a high level.

Details — None.

Additional information — None.

Transition statement — One option is for the requested service to be external to the broker, in which case the broker acts as an intermediary for service communication.

WebSphere Education

Message flow as HTTP/HTTPS client

- Legacy clients accessing web services
- WebSphere Message Broker as buffer between changing set of service providers (web service and other)
- Aggregation of web services

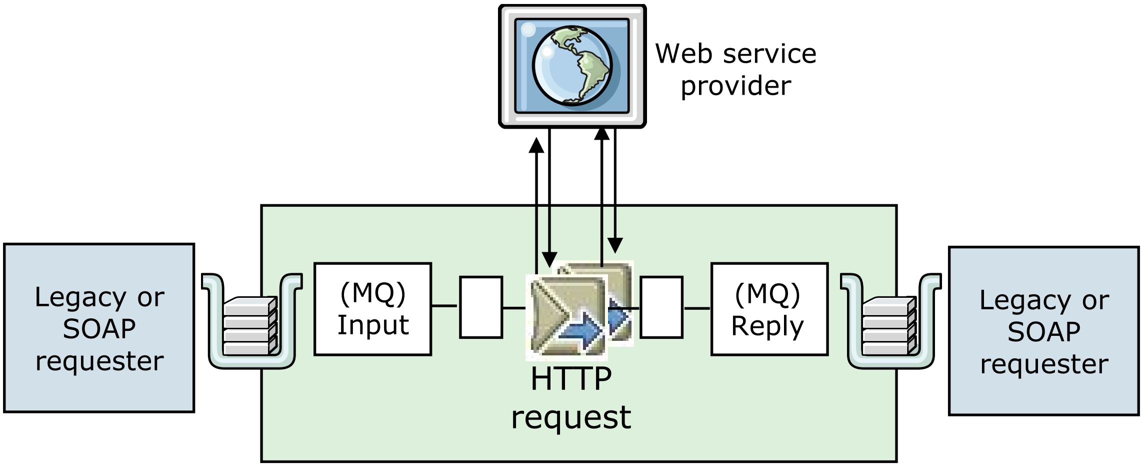


Figure 9-9. Message flow as HTTP/HTTPS client

WM643 / VM6431.0

Notes:

Use the HTTPRequest node to interact with a web service by using all or part of the input message as the request sent to that service. You can also configure the node to create an output message from the contents of the input message, augmented by the contents of the web service response, before you propagate the message to subsequent nodes in the message flow.

Depending on the configuration, this node constructs an HTTP or an HTTP over SSL (HTTPS) request from the specified contents of the input message. The node sends the request to the web service and then receives the response from the web service and parses the response for inclusion in the output tree. The node generates HTTP headers if they are required by your configuration.

You can set the property **Default Web service URL** on the HTTPRequest node to determine the destination URL for a web service request. A Compute node can be configured before the HTTPRequest node within the message flow to override the value set in the property.

Instructor notes:

Purpose — Overview of how WebSphere Message Broker can be implemented to act as a web service client.

Details — The HTTPRequest node performs full HTTP Request/Reply synchronously. HTTPRequest node uses HTTPS if it is specified on the URL. There are options to build the request header automatically. It includes an empty SOAPAction if it is automatically built. Content-length is entered automatically on request. It only handles HTTP V1.0 requests. You can specify the format of output messages in the node. It currently throws an exception if there is a 4XX HTTP response code. Exceptions are thrown on TCP/IP errors.

You can code ESQL that stores a URL string in `LocalEnvironment.Destination.HTTP.RequestURL`. This is retrieved by the HTTPRequest node and used in place of the node property value. You can also set the request URL in the special header X-Original-HTTP-URL in the HTTPRequestHeader section of the request message (which overrides all other settings) in a Compute node. Use the LocalEnvironment content for this purpose.

Additional information — None.

Transition statement — WebSphere Message Broker can also act as service provider.

WebSphere Education

Message flow as HTTP/HTTPS service provider

- Broker acts as intermediary to the web service to provide routing
- Many more combinations

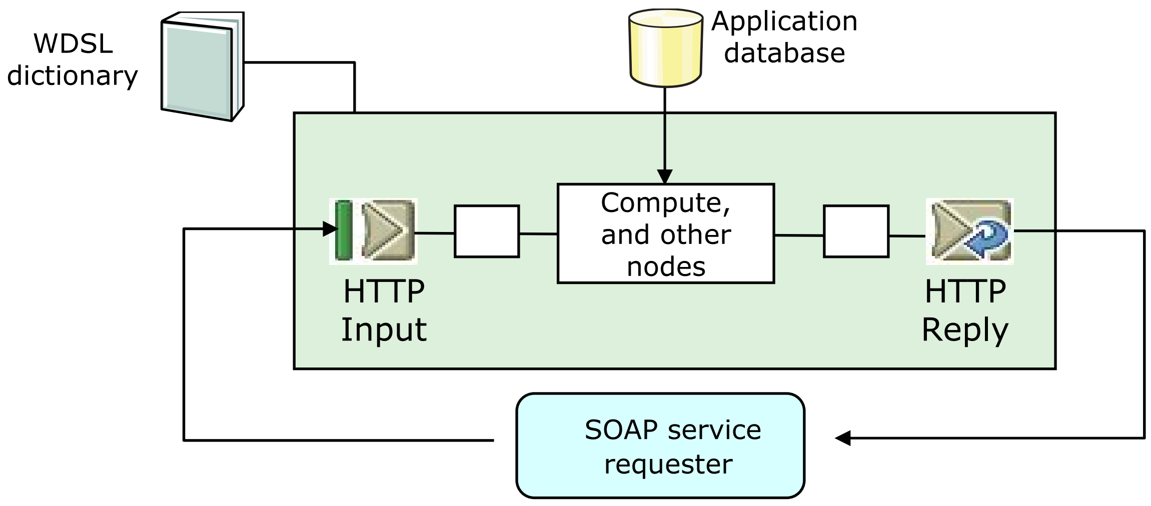


Figure 9-10. Message flow as HTTP/HTTPS service provider WM643 / VM6431.0

Notes:

Each broker has a single TCP/IP port on which incoming HTTP requests are accepted. Client applications can post to a predefined URL or a URL that has been already looked up. An attribute on each HTTPInput node is used to qualify the requests for which the node is responsible.

A message flow can act as an intermediary and transform, route, or aggregate web service requests. A message flow parses the request, and forwards it on to one or more destinations based on content, perhaps after transformation. Reply data from the destinations are returned to the original client, after further possible transformation or aggregation.

A message flow can transform HTTP to MQ. It parses the incoming HTTP request, and forwards the transformed request onto an MQ-enabled application that does not handle HTTP. The reply is then transformed into the client's native format and returned.

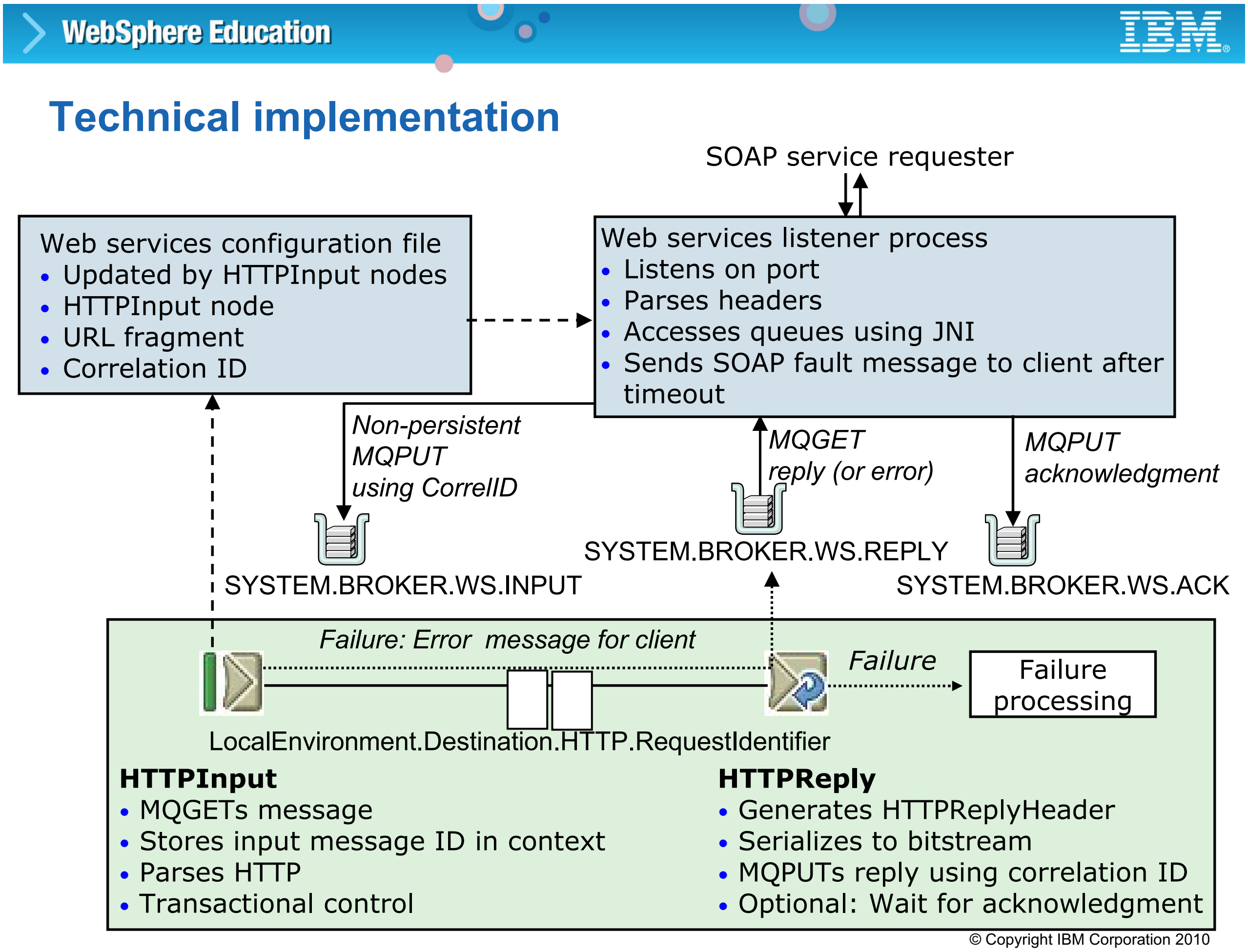
Instructor notes:

Purpose — Introduce the HTTPInput and HTTPReply nodes and show how WebSphere Message Broker can act as a web service provider.

Details — None.

Additional information — None.

Transition statement — Now you will take a look at the implementation in a bit more detail.



WebSphere Education

Technical implementation

SOAP service requester

Web services listener process
- Listens on port
- Parses headers
- Accesses queues using JNI
- Sends SOAP fault message to client after timeout

Web services configuration file
- Updated by HTTPInput nodes
- HTTPInput node
- URL fragment
- Correlation ID

HTTPInput
- MQGETs message
- Stores input message ID in context
- Parses HTTP
- Transactional control

HTTPReply
- Generates HTTPReplyHeader
- Serializes to bitstream
- MQPUTs reply using correlation ID
- Optional: Wait for acknowledgment

© Copyright IBM Corporation 2010

Figure 9-11. Technical implementation WM643 / VM6431.0

Notes:

The message flow logic (contained in the lower box) that implements a web service receives much broker support (everything shown outside the lower box) designed to retain state information about each incoming request message. Reply processing requires two pieces of information: (1) to which request message the message applies (CorrelID) and (2) where in the network the reply message is to be routed (LocalEnvironment).

Each broker has a single TCP/IP port on which incoming HTTP requests are accepted. It is set with command (the default is 7080). The flag to set the port is an uppercase `-P`. For example: `mqsichangebroker <Broker> -P 7080`

The HTTPInput node is like the MQInput node. It uses an Apache Tomcat servlet engine running in separate process (`biphttplistener`), started when the broker starts. Tomcat is only configurable through the broker. Port is a broker property. The `biphttplistener` communicates with execution groups using queues. It is important to note that the broker uses a single port for all incoming HTTP requests; this is a consideration for capacity planning.

The servlet has a configurable timeout value, after which it gives up waiting for a reply from the DataFlowEngine and sends a SOAP fault message back to the client; these properties are set by the developer as node properties.

After the message is passed on for processing, when it reaches an HTTPReply node, the result is sent back to the servlet engine which returns the result message using HTTP. The HTTPInput node routes each message it retrieves successfully to the Out terminal. If message validation fails, the message is routed to the failure terminal. If it is not connected, the message is discarded, the maximum client wait time expires, and the TCP/IP listener returns an error to the client. There are no other situations in which the message is routed to the failure terminal.

Instructor notes:

Purpose — Put it all in perspective.

Details — None.

HTTPInput node:

The HTTPInput node is like the MQInput node. It uses the Tomcat servlet engine (available from the Apache Jakarta project) running in a separate process (biphttplistener), which starts when the broker starts.

The message is parsed and passed to the node's out terminal for processing. When the message reaches the HTTPReply node, the result is sent back to Tomcat, and Tomcat returns the result to the requester using HTTP. If wanted, the HTTPReply node can set the HTTP headers for the reply message.

The HTTPInput node routes each message it retrieves successfully to its out terminal. If message validation is requested, and fails, the message is routed to the failure terminal. If the failure terminal is not connected, the message is discarded. If the **Maximum Client Wait Time** expires, the broker listener returns an error to the client. The are no other situations in which the message is routed to the failure terminal.

Remember that with the HTTP protocol, the message does not contain an MQMD (Message Queue Message Descriptor) or any other WebSphere MQ header. If an exception is thrown, an attempt is made to catch it at the HTTPInput node, presuming the catch terminal is wired. If not, the message is discarded. Otherwise, a SOAP fault message is sent back to the listener by the HTTPInput node with the exception text as part of the fault message.

HTTPReply node:

This node is based on the MQOutput node. After the MQPUT of the reply data, the HTTPReply node can (if a particular option is not checked) wait for an acknowledgement message from the servlet. Exceptions are thrown if either of the following occurs:

- No ACK message appears within a timeout period.
- An ACK message appears indicating that an *error occurred*; the exception contains the error text from the servlet.

HTTPRequest node:

This node performs full HTTP request/reply functions synchronously. Options exist to build an HTTP request header automatically. It includes an empty SoapAction header, if automatically built. Content length is entered automatically on request. HTTP 1.0 and 1.1 requests are currently supported. You can specify the format of the output message in the node. It currently throws an exception if it receives a 4XX HTTP response code. Exceptions are also thrown on TCP/IP errors.

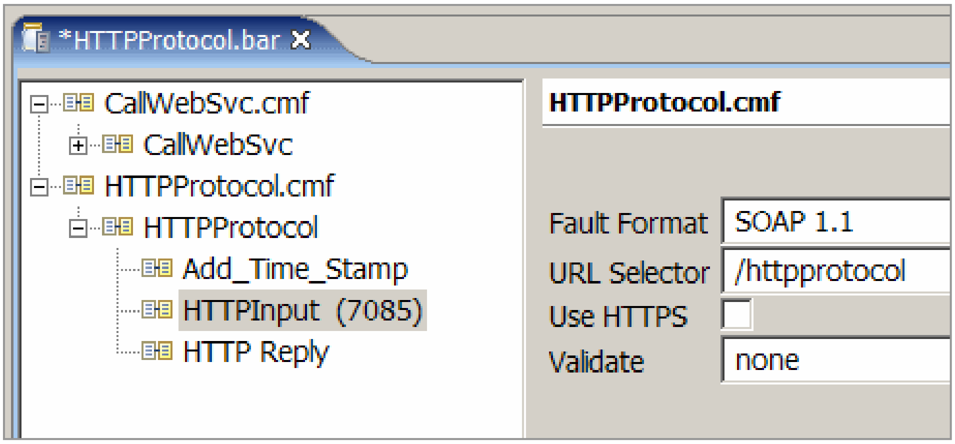
Additional information — None.

Transition statement — The developer can set many properties, but the administrator can only change a limited number.

WebSphere Education IBM

HTTPInput and HTTPReply node customization

- HTTPReply: One configurable property (in `.bar`)
  - Validate
- HTTPInput: Use **Configure** tab for broker archive (`.bar`) for Validate
  - Change the URL
  - Implement SSL



*HTTPProtocol.bar

CallWebSvc.cmf
- CallWebSvc

HTTPProtocol.cmf
- HTTPProtocol
  - Add_Time_Stamp
  - HTTPInput (7085)
  - HTTP Reply

HTTPProtocol.cmf

Fault Format: SOAP 1.1
URL Selector: /httpprotocol
Use HTTPS: □
Validate: none

© Copyright IBM Corporation 2010

Figure 9-12. HTTPInput and HTTPReply node customization WM643 / VM6431.0

Notes:

As an administrator, there are only a few properties that can be altered in a broker archive file before deploying. Other than the **Validate** property, the HTTPReply has no other properties that can be altered in the `.bar` file.

Fault Format defines the format of any HTTP errors that are returned to the client. This value can be `SOAP 1.1`, `SOAP 1.2`, or `HTML`.

URL Selector is the address associated with this invocation.

Use HTTPS indicates that the URL refers to an HTTPS address.

Validate can be altered in all three of the HTTP nodes. However, it is only possible for the administrator to set the property to **none** or **inherit**.

- **none**: No validation is performed.
- **inherit**: Instructs the node to use all the validation options provided with the input message tree in preference to any supplied on the node. Inherit resolves to **none**, **content**, or **content and value**. The last two can only be set by the developer.

Instructor notes:

Purpose — Show what can be controlled by the administrator for HTTPInput and HTTPReply nodes

Details — None.

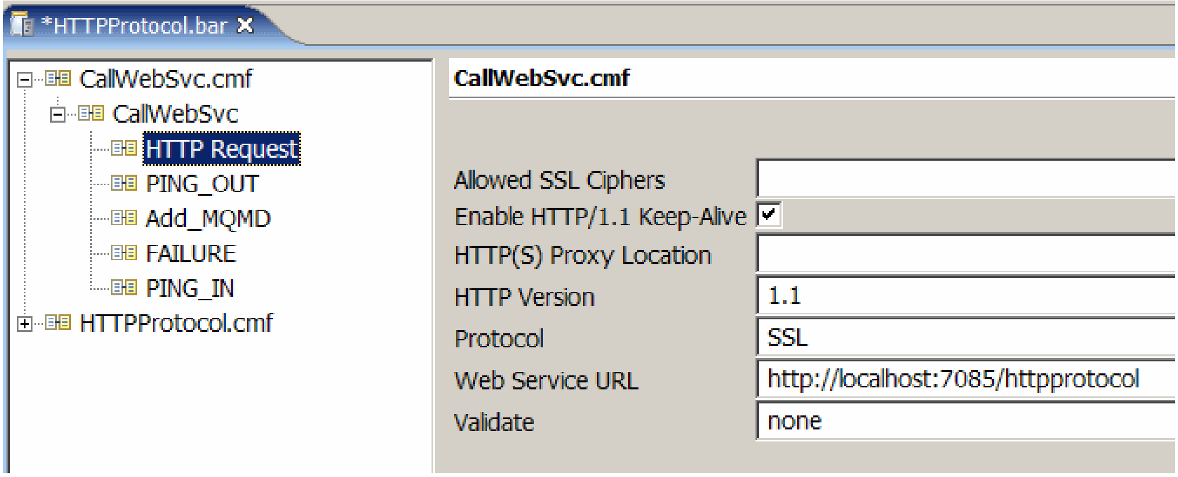
Additional information — None.

Transition statement — The HTTPRequest node can also be customized.

WebSphere Education

HTTPRequest node customization

- Use **Configure** tab for broker archive (`.bar`)
  - Change the URL
  - HTTP properties: version, proxy location, keep-alive
  - Implement or alter SSL
  - Validate



*HTTPProtocol.bar

- CallWebSvc.cmf
  - CallWebSvc
    - HTTP Request
    - PING_OUT
    - Add_MQMD
    - FAILURE
    - PING_IN
- HTTPProtocol.cmf

CallWebSvc.cmf

Allowed SSL Ciphers	
Enable HTTP/1.1 Keep-Alive	☑
HTTP(S) Proxy Location	
HTTP Version	1.1
Protocol	SSL
Web Service URL	http://localhost:7085/httpprotocol
Validate	none

Figure 9-13. HTTPRequest node customization WM643 / VM6431.0

Notes:

The HTTPRequest node properties are:

Allowed SSL Ciphers: Specify a single cipher (such as SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA) or a list of ciphers (comma-separated) that are the only ones used by the connection. This list must include one or more ciphers that are accepted by the remote server. An empty string is the default; it allows the node to use any, or all of the available ciphers during the SSL connection handshake, which allows the greatest scope for making a successful SSL connection. The possible ciphers are dependent on the provider.

Enable HTTP/1.1 Keep-Alive: Only supported with HTTP/1.1.

HTTP(S) Proxy Location: Location of the proxy server to which requests are sent.

HTTP Version: The possible values for the HTTP version property are 1.0 and 1.1. If set to 1.1, then HTTP/1.1 keep-alive is permitted.

Protocol: If implementing SSL, there are three possible values. If SSL is not implemented, any value in this property is ignored.

- **SSL** (the default): This tries to connect using the SSLv3 protocol first, but allows the handshake to fall back to the SSLv2 protocol where the SSLv2 protocol is supported by the underlying JSSE provider.
- **SSLv3**: Tries to connect with the SSLv3 protocol only. Fallback to SSLv2 is not allowed.
- **TLS**: Tries to connect with the TLS protocol only. Fallback to SSLv3 or SSLv2 is not allowed.

Both ends of an SSL connection must agree on the protocol to use, so the selected protocol must be one that the remote server can accept.

Web Service URL: The URL address that is to be used to call the web service.

Validate: The same options as supported in the other HTTP nodes.

Instructor notes:

Purpose — Call attention to the fact that HTTP messages have headers.

Details — None.

Additional information — None.

Transition statement — A change in emphasis takes place here. Next is a discussion on security and the use of secure sockets with the HTTP protocol.

9.3. Implementing SSL with HTTP nodes

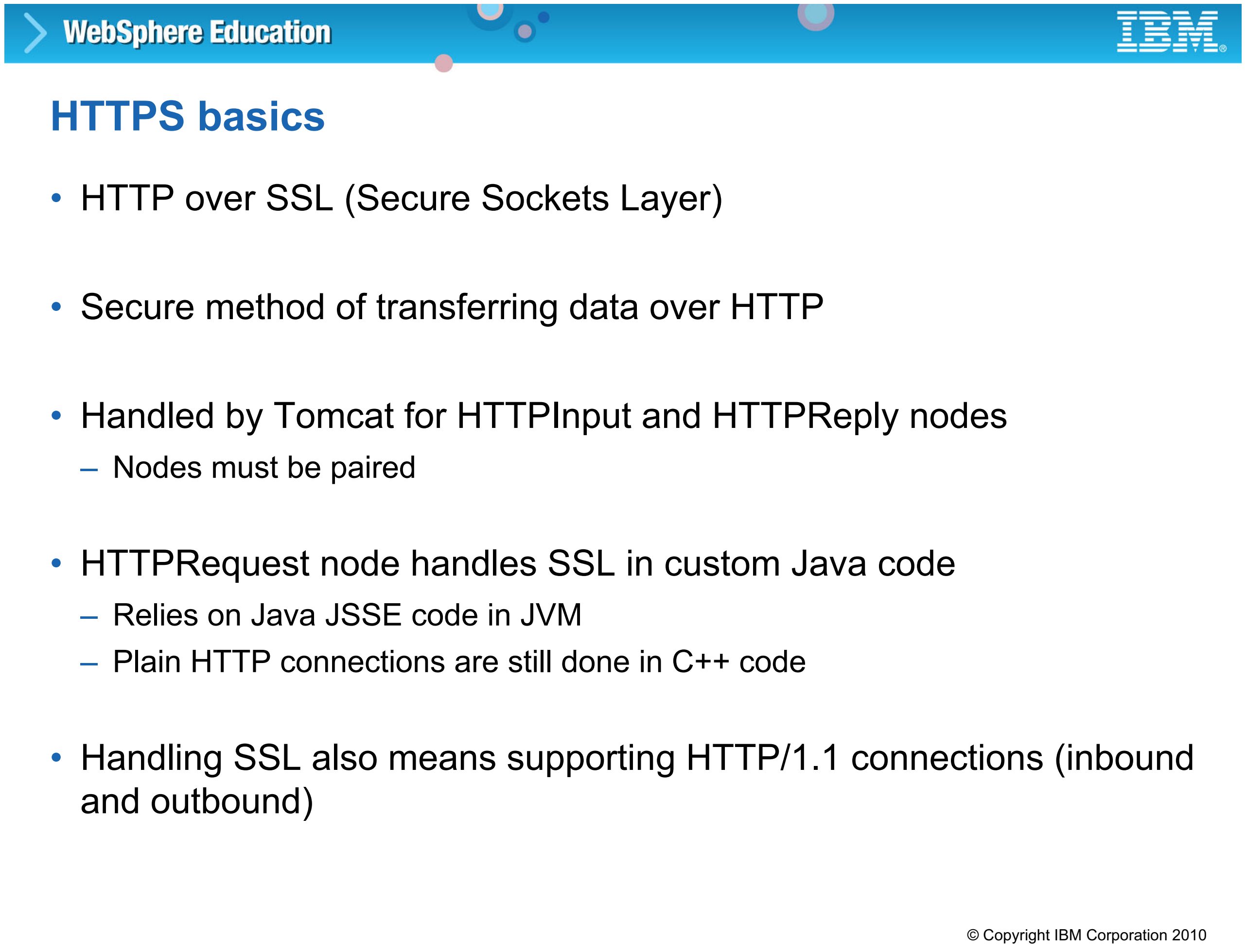
Instructor topic introduction

What students will do — Learn that message flows can function as a web service, or call a web service, anywhere in the network that the broker is a part of.

How students will do it — Listen to the lecture and participate in classroom discussion. Exercise 8 will allow students to monitor a message flow using HTTP. It will also allow students to use the tools available to implement HTTP over SSL for use by these nodes.

What students will learn — A WebSphere Message Broker can host web services or serve as an intermediary in calling a web service.

How this will help students on their job — A program function that can be called, rather than developed or purchased, represents a viable alternative to application design.



WebSphere Education

HTTPS basics

- HTTP over SSL (Secure Sockets Layer)
- Secure method of transferring data over HTTP
- Handled by Tomcat for HTTPInput and HTTPReply nodes
  - Nodes must be paired
- HTTPRequest node handles SSL in custom Java code
  - Relies on Java JSSE code in JVM
  - Plain HTTP connections are still done in C++ code
- Handling SSL also means supporting HTTP/1.1 connections (inbound and outbound)

Figure 9-14. HTTPS basics WM643 / VM6431.0

Notes:

The implementation of HTTPS support allows HTTPRequest nodes to call web services over a secure HTTP connection and allows the HTTPInput node to receive its requests across a secure HTTP connection.

You must use HTTP/1.1 if you implement the SSL support.

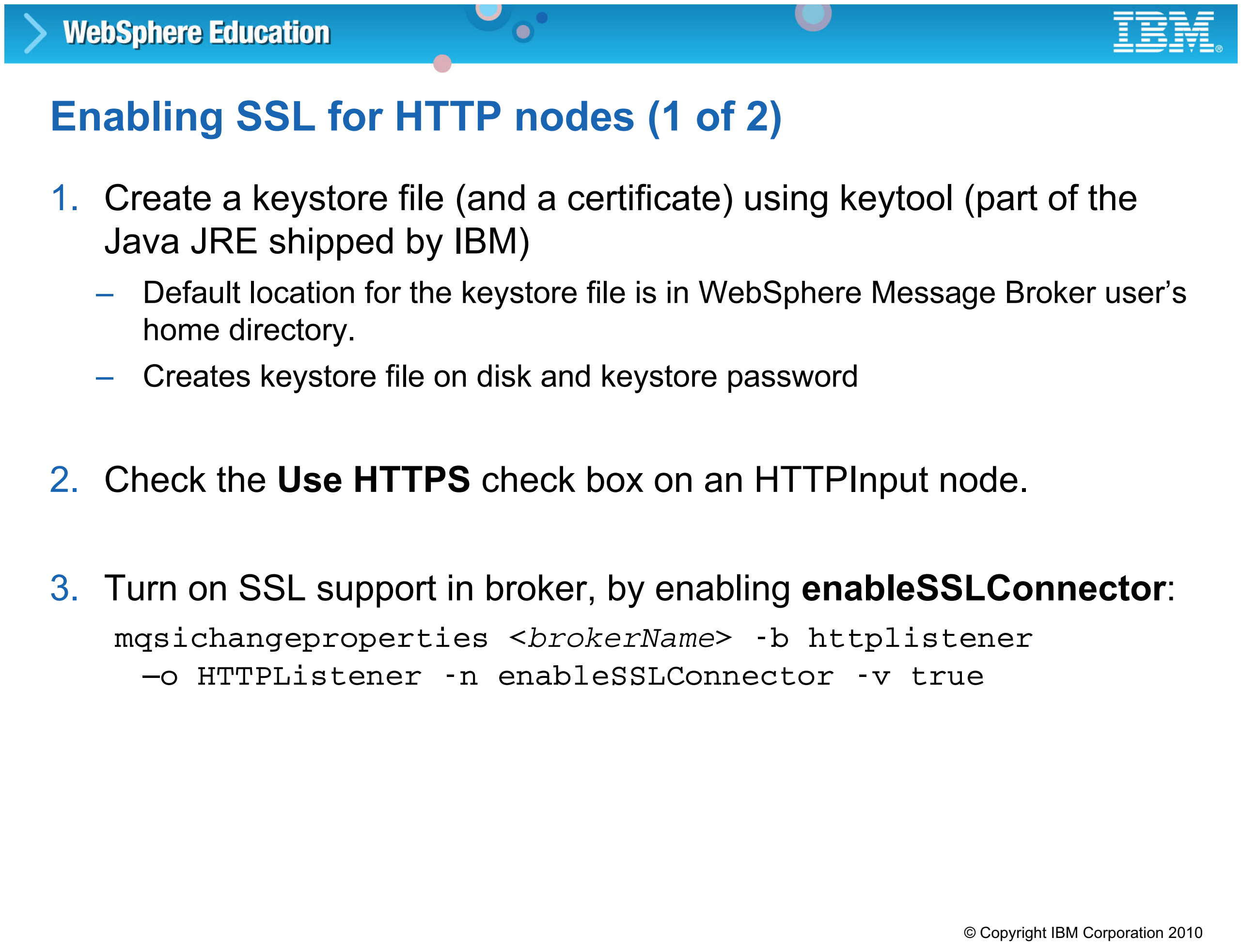
Instructor notes:

Purpose — Provide a bit of information about SSL and HTTPS implemented in the HTTP nodes.

Details — None.

Additional information — None.

Transition statement — Before doing any configuration of the nodes, there are other tasks to perform.



WebSphere Education
IBM

Enabling SSL for HTTP nodes (1 of 2)

1. Create a keystore file (and a certificate) using keytool (part of the Java JRE shipped by IBM)
   - Default location for the keystore file is in WebSphere Message Broker user's home directory.
   - Creates keystore file on disk and keystore password
2. Check the **Use HTTPS** check box on an HTTPInput node.
3. Turn on SSL support in broker, by enabling **enableSSLConnector**:

```
mqsichangeproperties <brokerName> -b httplistener
  –o HTTPListener -n enableSSLConnector -v true
```

Figure 9-15. Enabling SSL for HTTP nodes (1 of 2) WM643 / VM6431.0

Notes:

The keytool utility is used to create the necessary file (and certificates) for use in implementing the SSL support for the nodes. If the path to the keystore file changes, you must be sure to indicate this using the `mqsichangeproperties` command.

If the **Use HTTPS** check box is not set in the HTTPInput node, there is no attempt to connect using the HTTPS connection.

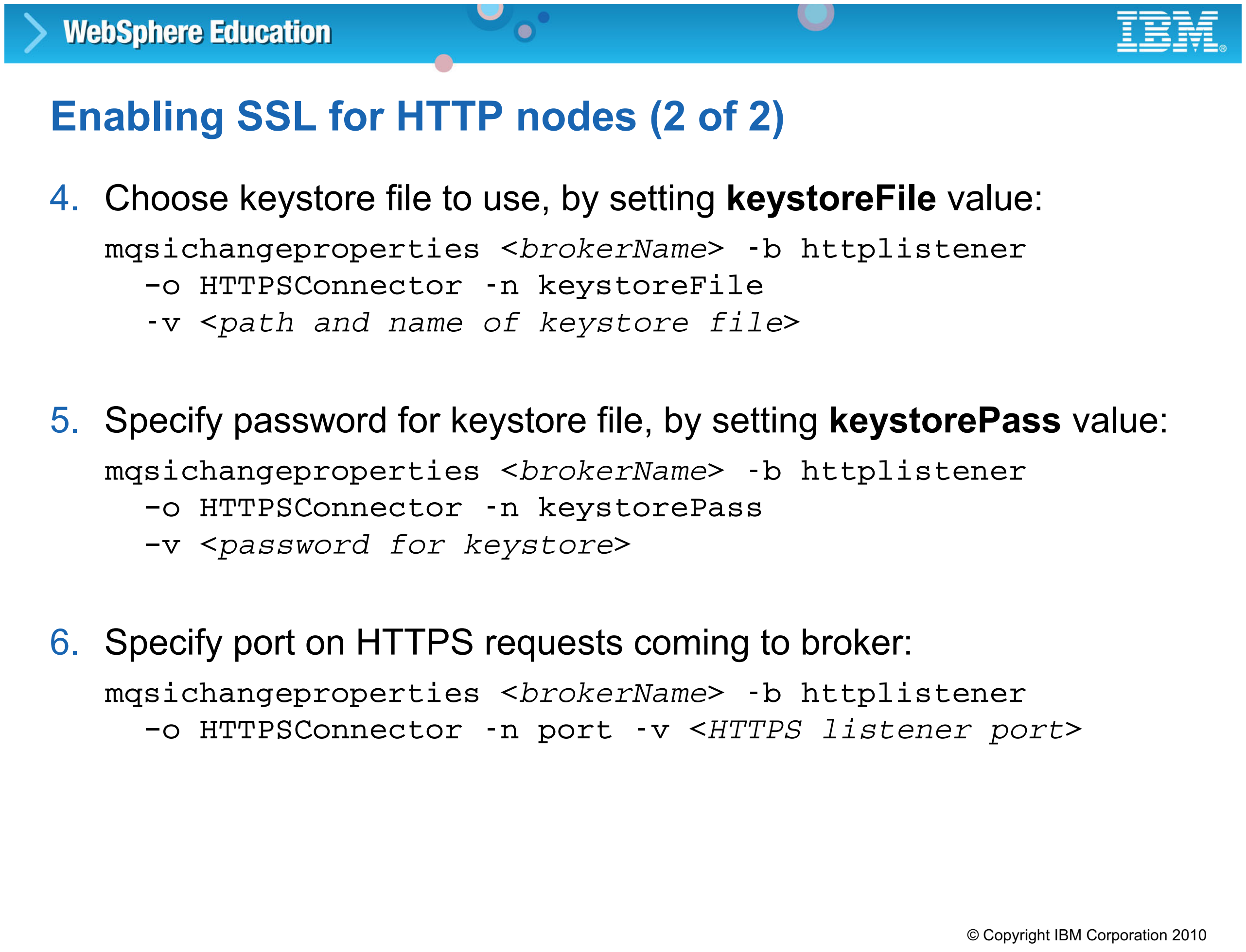
Instructor notes:

Purpose — Mention the keystore file.

Details — More details on the `mqsichangeproperties` command follow and the exercise will allow students to use the keytool.

Additional information — None.

Transition statement — After the keystore is created, you must make sure that it is found and used properly.



WebSphere Education

Enabling SSL for HTTP nodes (2 of 2)

4. Choose keystore file to use, by setting **keystoreFile** value:

```
mqsichangeproperties <brokerName> -b httplistener
  -o HTTPSConnector -n keystoreFile
  -v <path and name of keystore file>
```

5. Specify password for keystore file, by setting **keystorePass** value:

```
mqsichangeproperties <brokerName> -b httplistener
  -o HTTPSConnector -n keystorePass
  -v <password for keystore>
```

6. Specify port on HTTPS requests coming to broker:

```
mqsichangeproperties <brokerName> -b httplistener
  -o HTTPSConnector -n port -v <HTTPS listener port>
```

Figure 9-16. Enabling SSL for HTTP nodes (2 of 2) WM643 / VM6431.0

Notes:

Unlike other invocations of the `mqsichangeproperties` command, the keystore commands do not work unless the broker is active**.**

All of these commands are directed at the HTTPListener. The object within the HTTPListener is the HTTPSConnector (enabled by setting the SSLConnector in the HTTPListener). This allows you to set up the path and password for the key file and establish a separate port within the listener for use by SSL.

Instructor notes:

Purpose — Describe the steps to take using `mqsichangeproperties`.

Details — None.

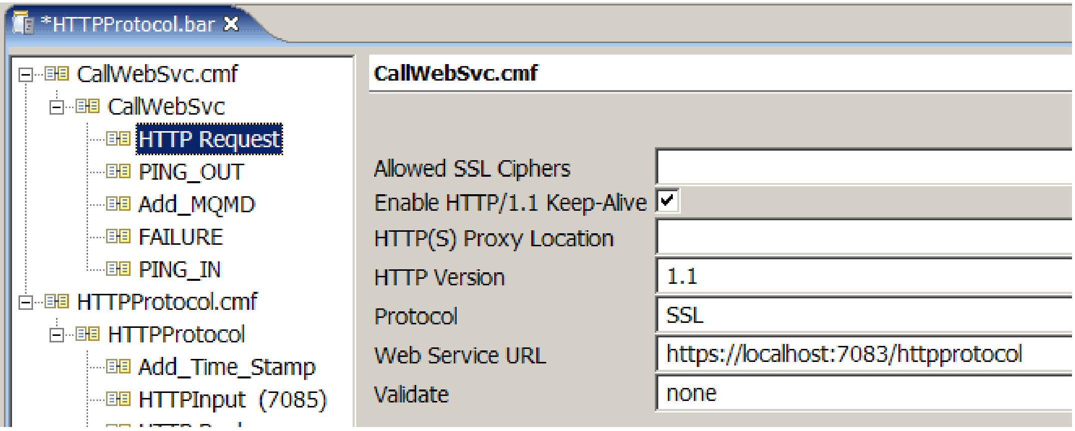
Additional information — None.

Transition statement — After this work has been completed, it is possible to implement SSL support in the nodes.

WebSphere Education

HTTPRequest node details for SSL

- **Protocol** options
  - SSL: try SSLv3 first, allows fallback to SSLv2
  - SSLv3: try SSLv3 only
  - TLS: try TLS only
- **Allowed SSL ciphers**
  - Default of empty means all broker JVM supported ciphers
  - One or more explicit ciphers in list means only these specified ciphers
- **URL**
  - HTTPS and port used by HTTPSConnector



CallWebSvc.cmf

Allowed SSL Ciphers	
Enable HTTP/1.1 Keep-Alive	☑
HTTP(S) Proxy Location	
HTTP Version	1.1
Protocol	SSL
Web Service URL	https://localhost:7083/httpprotocol
Validate	none

Figure 9-17. HTTPRequest node details for SSL

WM643 / VM6431.0

Notes:

Developers can certainly set up their message flows to use SSL by setting the appropriate properties. However, it is also possible for the administrator to implement SSL by tailoring the broker archive (`.bar`) file.

This figure is an example of an HTTPRequest node, the intention being to connect using HTTP over SSL (HTTPS) to a web service found at the URL shown. The port being used is 7083 which is the default SSL port but it can be changed.

By leaving the **Allowed SSL Ciphers** property blank, any cipher is supported.

Instructor notes:

Purpose — Decide what must be changed to implement SSL in an HTTPRequest node.

Details — None.

Additional information — None.

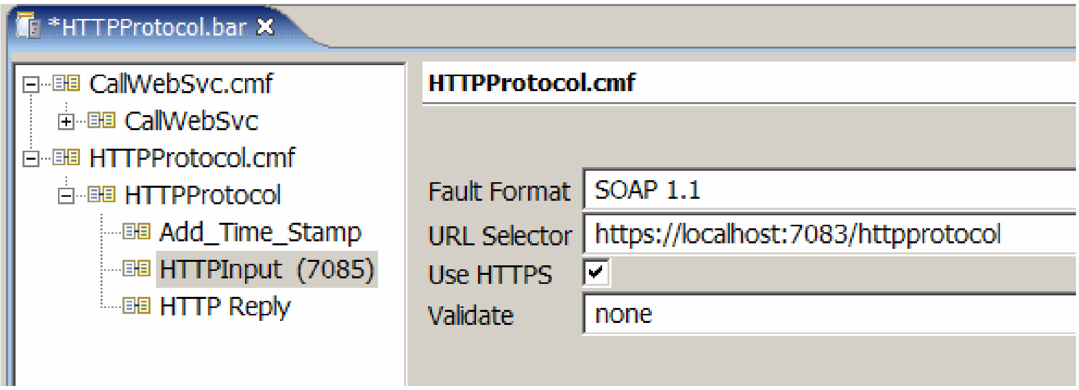
Transition statement — It is also possible to implement HTTP over SSL on incoming requests to an HTTPInput node.

WebSphere Education

HTTPInput node and SSL

- `mqsichangeproperties` in conjunction with a setting on an input node (more in a moment)

- Alter .bar file :
  - URL selector: HTTPs and port used by SSL
  - **Use HTTPS** check box



*HTTPProtocol.bar

HTTPProtocol.cmf

Fault Format: SOAP 1.1
URL Selector: https://localhost:7083/httpprotocol
Use HTTPS: ☑
Validate: none

Figure 9-18. HTTPInput node and SSL

WM643 / VM6431.0

Notes:

The HTTPInput node only needs to have the **Use HTTPS** box checked, and the URL must be set up for SSL. There is no cipher or protocol to select.

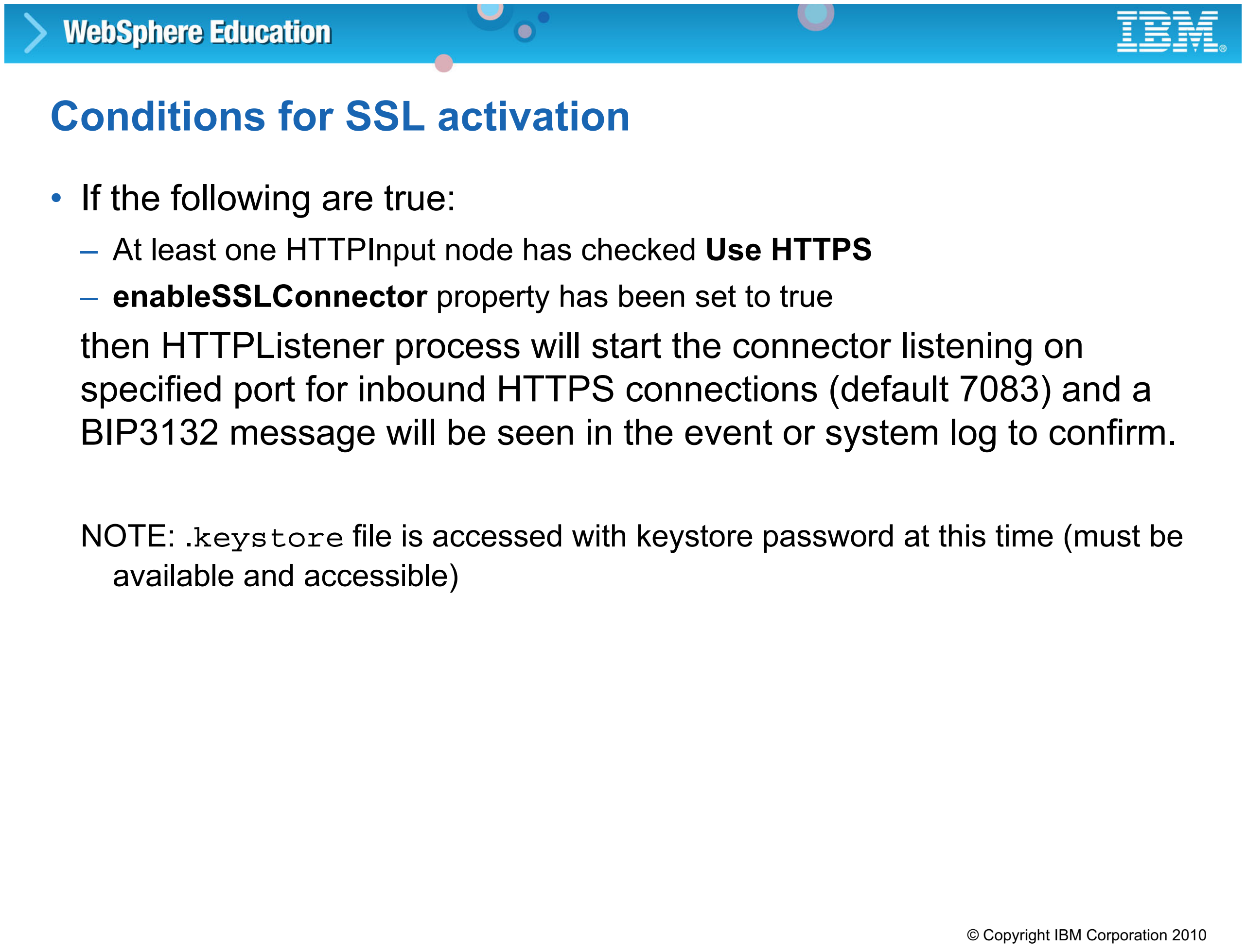
Instructor notes:

Purpose — Look at the HTTPInput node settings.

Details — The HTTPInput node only needs to have the **Use HTTPS** box checked, and the URL must be set up for SSL.

Additional information — None.

Transition statement — So, after all of this has been set up, what happens?



WebSphere Education

Conditions for SSL activation

- If the following are true:
  - At least one HTTPInput node has checked **Use HTTPS**
  - **enableSSLConnector** property has been set to true

  then HTTPListener process will start the connector listening on specified port for inbound HTTPS connections (default 7083) and a BIP3132 message will be seen in the event or system log to confirm.

  NOTE: `.keystore` file is accessed with keystore password at this time (must be available and accessible)

Figure 9-19. Conditions for SSL activation WM643 / VM6431.0

Notes:

If the conditions are all met and the setup has been done properly, after the first message flow using HTTPS is successfully deployed, the listener is started for the SSL port.

It is important that the keystore is properly set up and available before SSL activation, as the keystore file is accessed at that time as well.

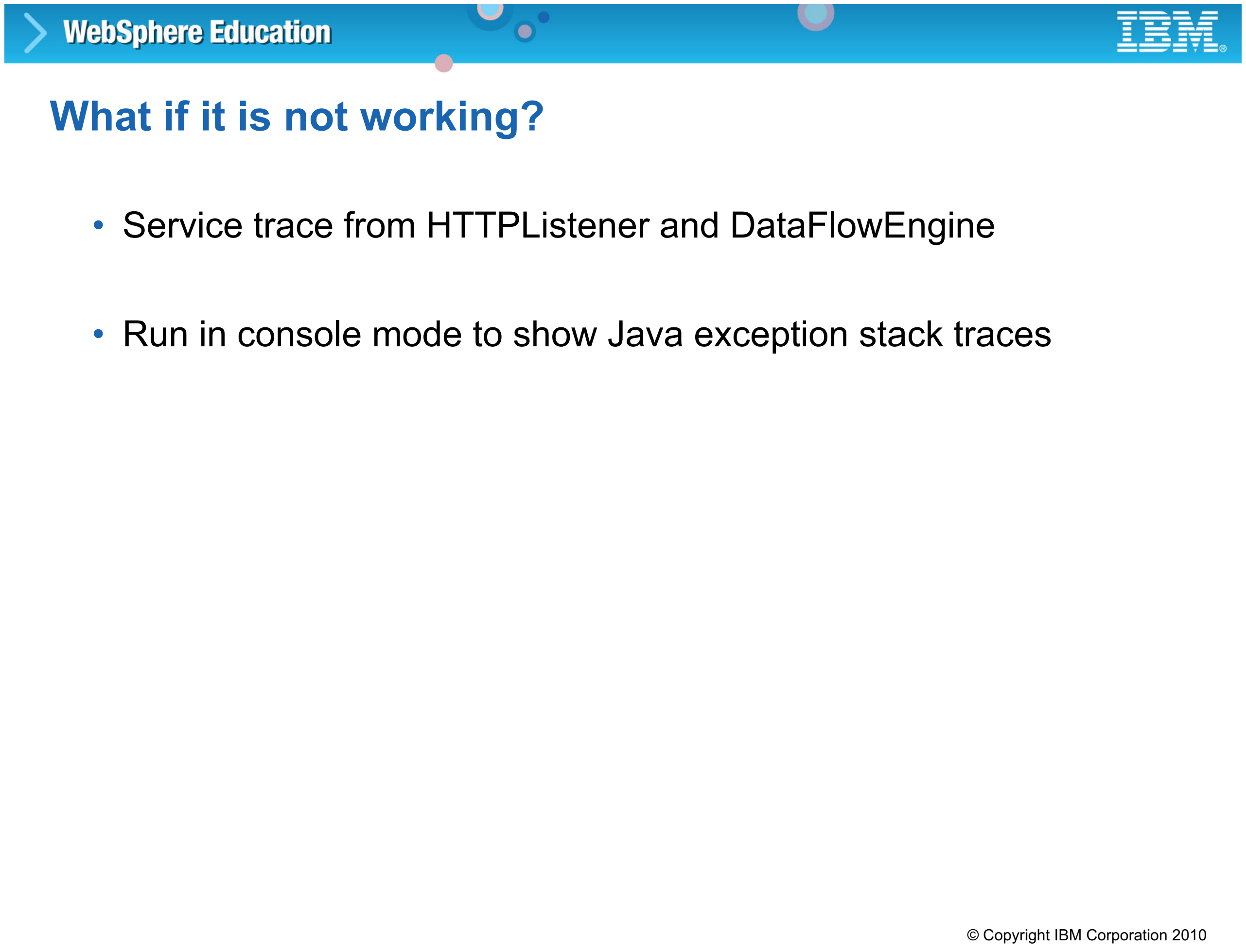
Instructor notes:

Purpose — Review what must be in place before the first deployment.

Details — None.

Additional information — None.

Transition statement — In case something is not working, it is possible to find out why.



WebSphere Education

What if it is not working?

- Service trace from HTTPListener and DataFlowEngine
- Run in console mode to show Java exception stack traces

Figure 9-20. What if it is not working? WM643 / VM6431.0

Notes:

The trace of the HTTPListener and the execution group (service trace, not user trace) provides clues if the listener does not activate or if the message flows do not appear to be working. If the trace is run in console mode, you can obtain additional information about any Java exceptions.

Instructor notes:

Purpose — Mention the traces available for debugging.

Details — Trace was discussed in the Problem determination unit. This is just another use for it.

Additional information — None.

Transition statement — SOAP nodes also support security using policy sets and policy set bindings.

9.4. Capacity considerations for HTTP nodes

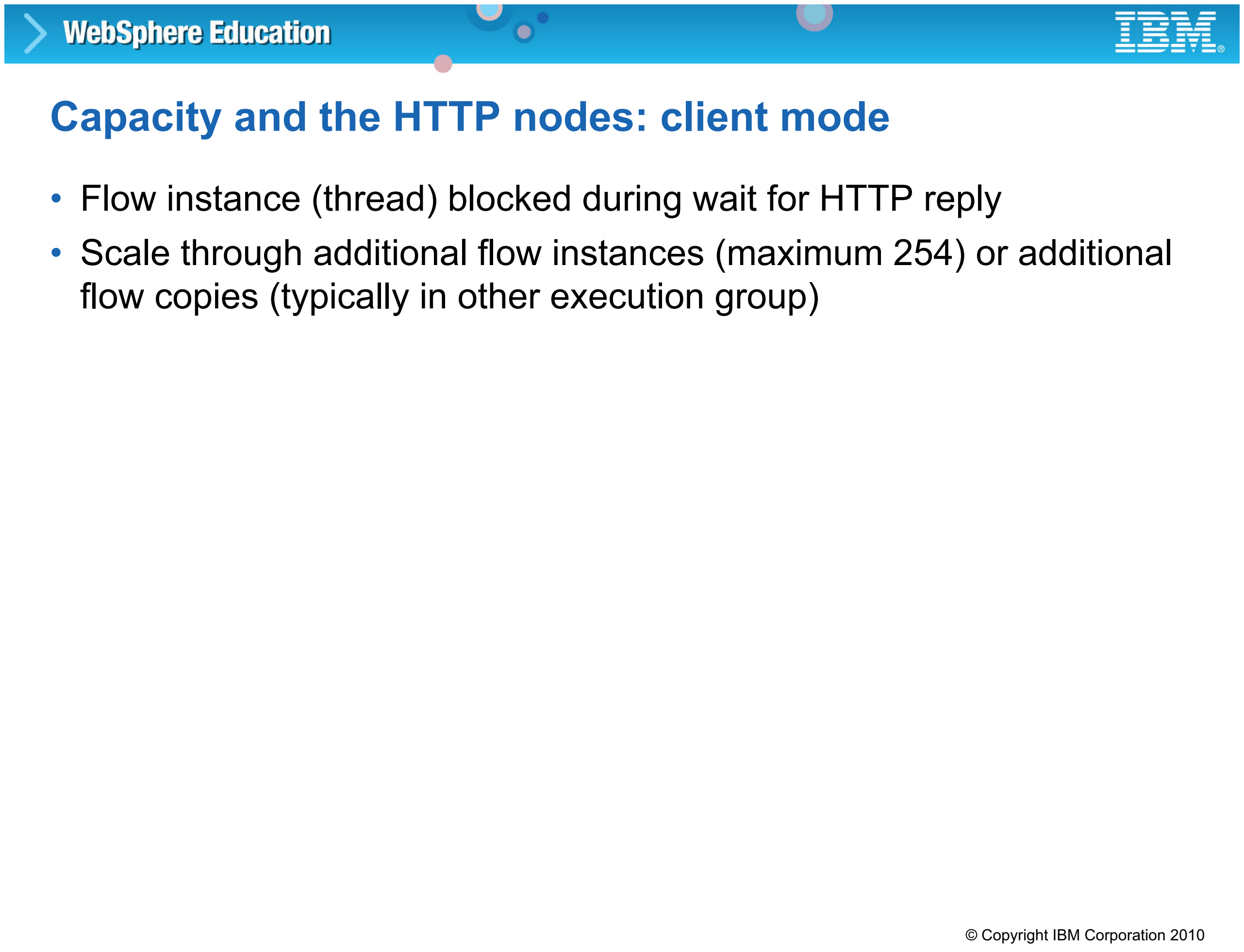
Instructor topic introduction

What students will do — Learn that message flows can function as a web service, or call a web service, anywhere in the network the broker is a part of.

How students will do it — Listen to the lecture and participate in classroom discussion. Exercise 8 will permit students to configure an HTTP node and exercise a web service supported by a WebSphere Message Broker.

What students will learn — A WebSphere Message Broker can host web services or serve as an intermediary in calling a web service.

How this will help students on their job —



WebSphere Education

Capacity and the HTTP nodes: client mode

- Flow instance (thread) blocked during wait for HTTP reply
- Scale through additional flow instances (maximum 254) or additional flow copies (typically in other execution group)

Figure 9-21. Capacity and the HTTP nodes: client mode WM643 / VM6431.0

Notes:

When a flow is invoking an HTTP request, the major thing to remember is to scale the flow by increasing the number of instances (in the broker archive) or by deploying the flow to multiple execution groups.

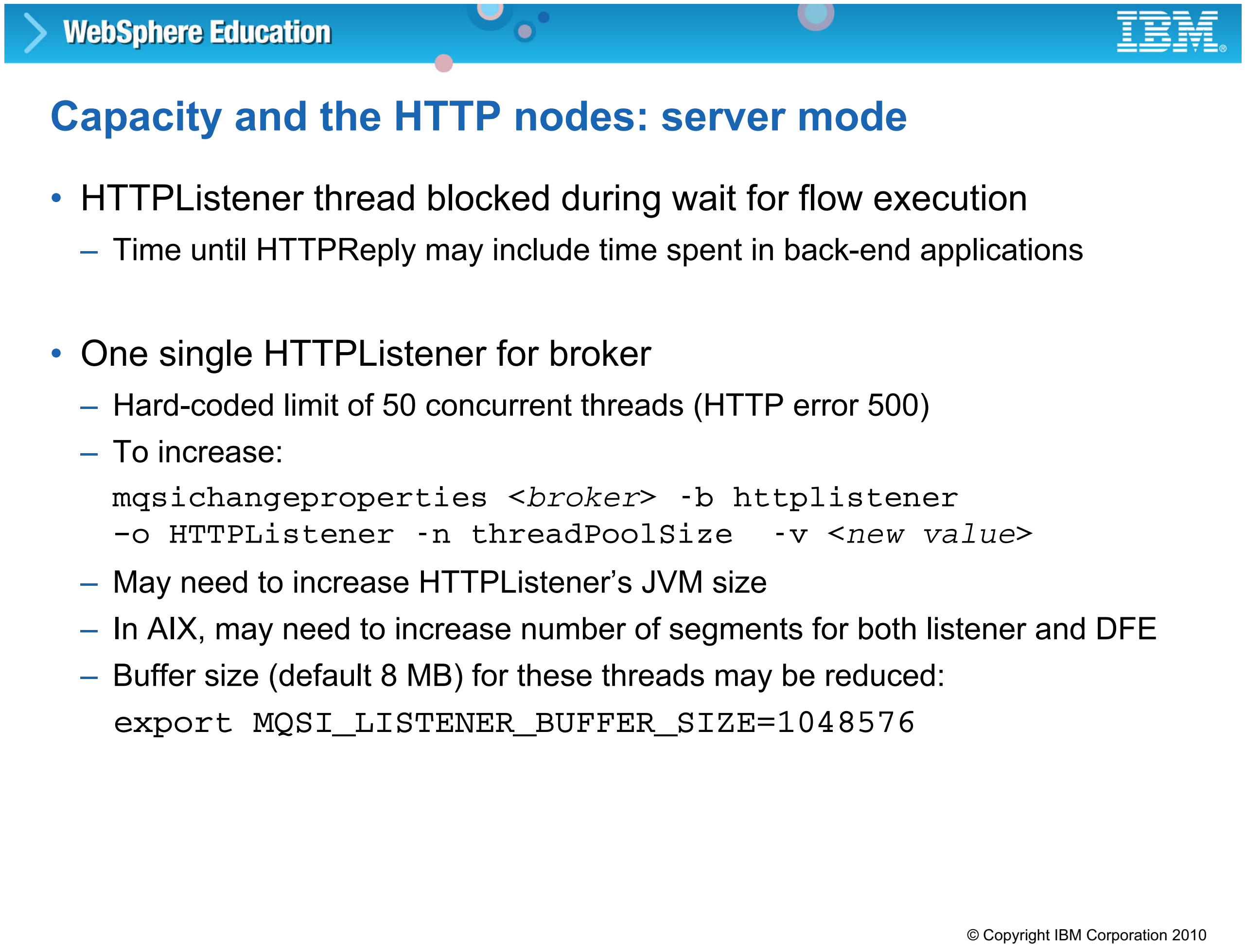
Instructor notes:

Purpose — Provide an overview of some capacity considerations.

Details — None.

Additional information — None.

Transition statement —



WebSphere Education

Capacity and the HTTP nodes: server mode

- HTTPListener thread blocked during wait for flow execution
  – Time until HTTPReply may include time spent in back-end applications
- One single HTTPListener for broker
  – Hard-coded limit of 50 concurrent threads (HTTP error 500)
  – To increase:

```
mqsichangeproperties <broker> -b httplistener
-o HTTPListener -n threadPoolSize  -v <new value>
```

  – May need to increase HTTPListener’s JVM size
  – In AIX, may need to increase number of segments for both listener and DFE
  – Buffer size (default 8 MB) for these threads may be reduced:

```
export MQSI_LISTENER_BUFFER_SIZE=1048576
```

Figure 9-22. Capacity and HTTP nodes: server mode WM643 / VM6431.0

Notes:

When the broker is behaving as an HTTP server (with HTTPInput nodes), there are more possibilities of issues with capacity.

Note

A heavy volume of incoming requests can cause the maximum number of threads to be exceeded.

For heavy loads, you might also consider increasing the JVM size for your HTTPListener.

Finally, AIX has sometimes exhibited issues regarding the number of segments for both the listener and the DataFlowEngine. You might need to tune that environment, perhaps reducing the default buffer size.

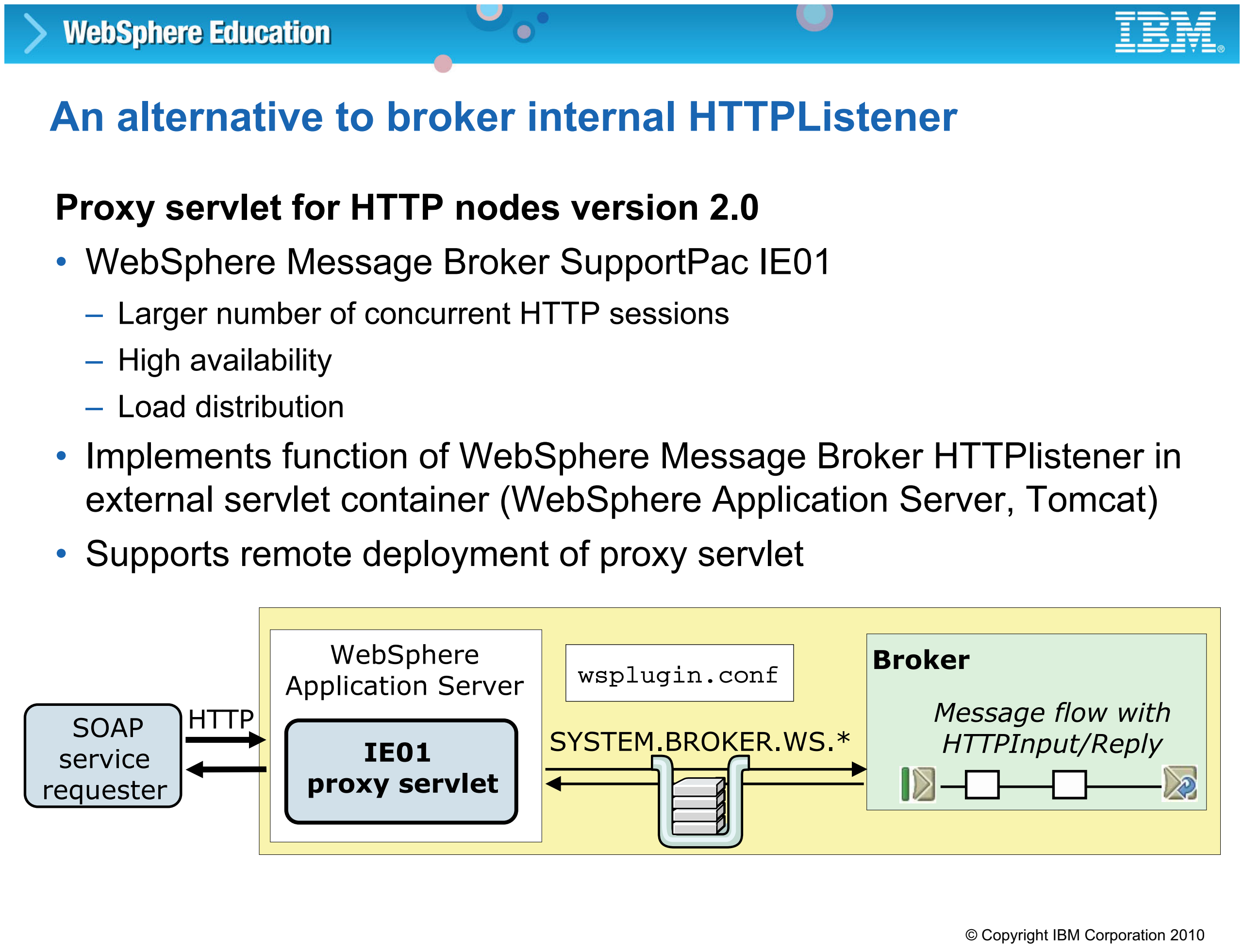
Instructor notes:

Purpose —

Details — The commands to change the maximum threads are different, depending on the version of the product. With version 6, this is another use of the `mqsichangeproperties` command.

Additional information —

Transition statement — There are two alternatives to using the internal listener.



WebSphere Education

IBM

An alternative to broker internal HTTPListener

Proxy servlet for HTTP nodes version 2.0

- WebSphere Message Broker SupportPac IE01
  - Larger number of concurrent HTTP sessions
  - High availability
  - Load distribution
- Implements function of WebSphere Message Broker HTTPlistener in external servlet container (WebSphere Application Server, Tomcat)
- Supports remote deployment of proxy servlet

Figure 9-23. An alternative to broker internal HTTPListener WM643 / VM6431.0

Notes:

The WebSphere Message Broker SupportPac IE01 is a product extension (category 3) which means it is a supported function. It allows for an increased number of HTTP sessions.

Instructor notes:

Purpose — Discuss SupportPac IE01.

Details — If students do not know categories, this is a good time to remind them.

With FixPack 1 of version 6, the implementation has been changed to allow for remote deployment. Before this, the application server and the broker were required to reside on the same server.

Additional information — None.

Transition statement — There is yet another alternative to the broker listener.

WebSphere Education IBM

Another alternative to broker internal HTTPListener

Custom servlets in WebSphere Application Server

- Transforms HTTP to and from JMS
- WebSphere Application Server and Broker connected through JMS
- HTTP traffic scaled through WebSphere Application Server (cluster, Edge Server)
- Broker workload scaled through WebSphere MQ clusters

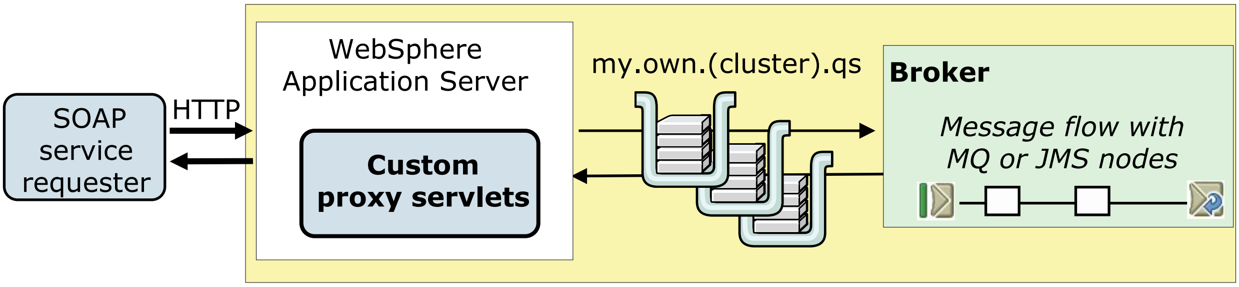


Figure 9-24. Another alternative to broker internal HTTPListener WM643 / VM6431.0

Notes:

Since the broker is now far removed from the implementation, and WebSphere MQ queue manager clusters are in use, the ability to scale the solution is increased even further.

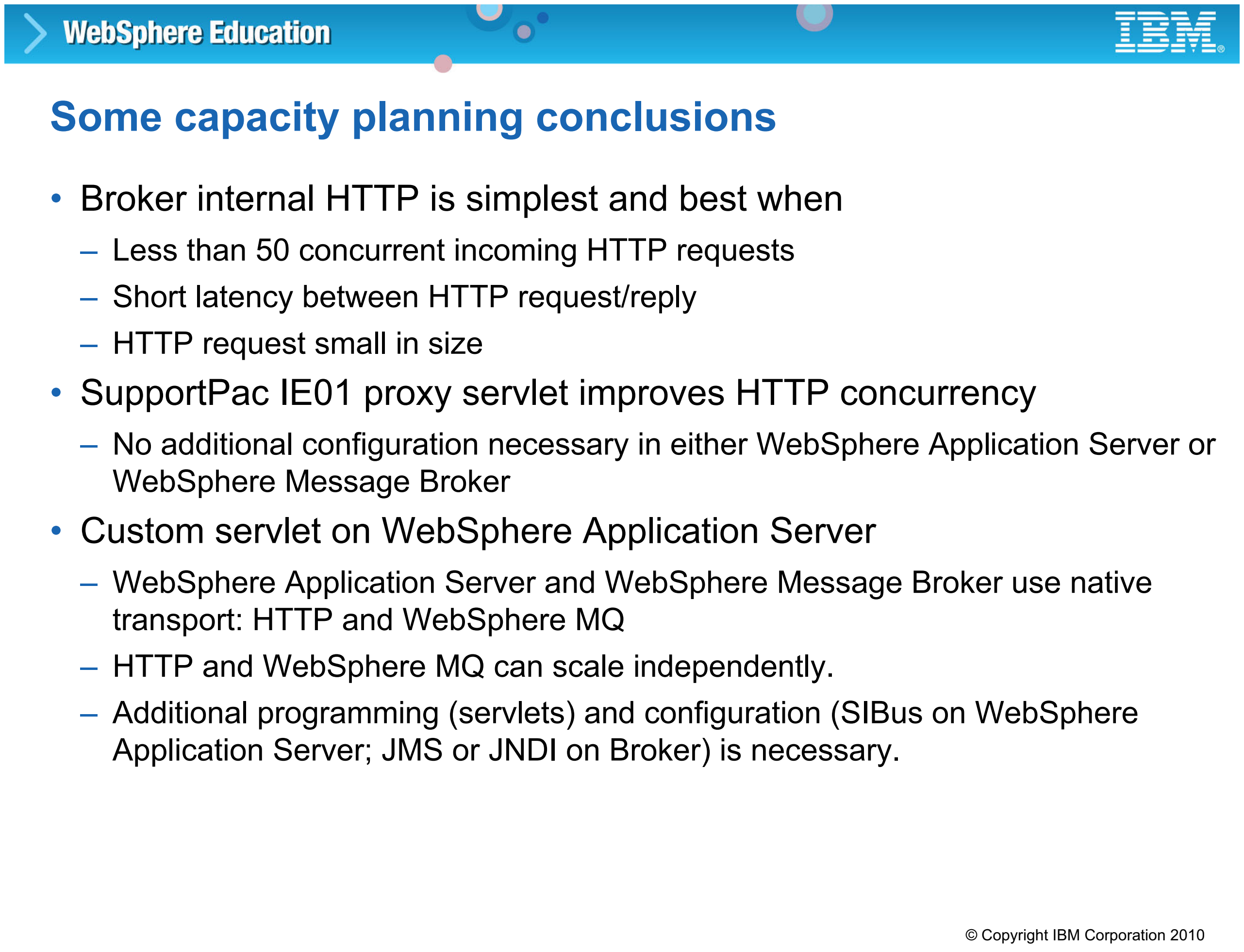
Instructor notes:

Purpose — Show another alternative.

Details — None.

Additional information — None.

Transition statement — Now, you will review what you have seen as far as capacity considerations and draw conclusions.



WebSphere Education

Some capacity planning conclusions

- Broker internal HTTP is simplest and best when
  - Less than 50 concurrent incoming HTTP requests
  - Short latency between HTTP request/reply
  - HTTP request small in size
- SupportPac IE01 proxy servlet improves HTTP concurrency
  - No additional configuration necessary in either WebSphere Application Server or WebSphere Message Broker
- Custom servlet on WebSphere Application Server
  - WebSphere Application Server and WebSphere Message Broker use native transport: HTTP and WebSphere MQ
  - HTTP and WebSphere MQ can scale independently.
  - Additional programming (servlets) and configuration (SIBus on WebSphere Application Server; JMS or JNDI on Broker) is necessary.

Figure 9-25. Some capacity planning conclusions WM643 / VM6431.0

Notes:

The decision to use one solution over another should be considered with the requirements in a particular environment. Some solutions are simpler to implement but offer less scalability.

Instructor notes:

Purpose — Recap the solutions discussed.

Details — None.

Additional information — None.

Transition statement — Security is also available for WebSphere Message Broker SOAP nodes using security policies and security policy bindings.

9.5. Creating policy sets

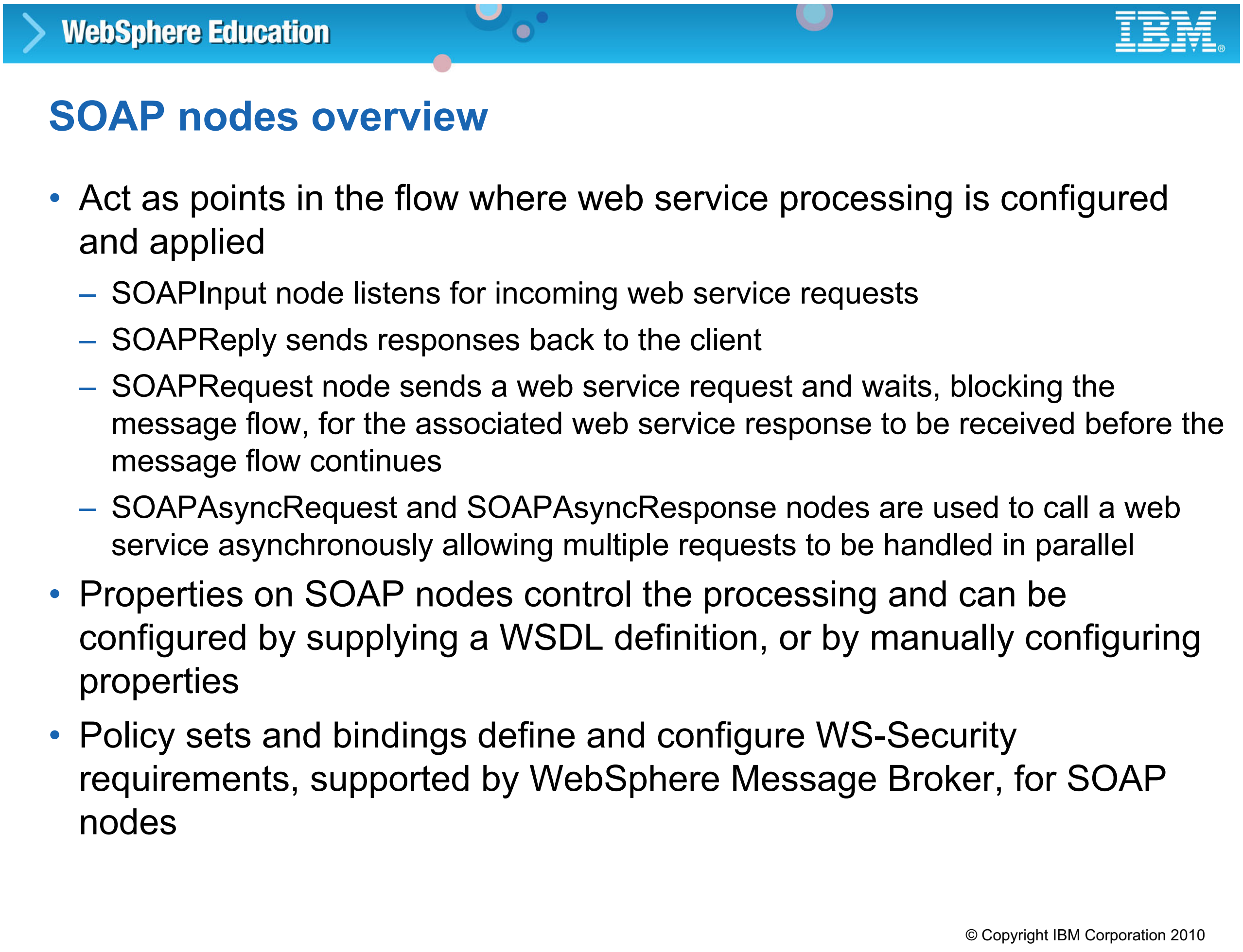
Instructor topic introduction

What students will do — Learn how to create and administer policy sets and policy set bindings for SOAP nodes.

How students will do it — Listen to lecture.

What students will learn — How to create policy sets and policy set bindings, and how to import and export policy sets.

How this will help students on their job — Students will be able to administer policy sets.



WebSphere Education

SOAP nodes overview

- Act as points in the flow where web service processing is configured and applied
  - SOAPInput node listens for incoming web service requests
  - SOAPReply sends responses back to the client
  - SOAPRequest node sends a web service request and waits, blocking the message flow, for the associated web service response to be received before the message flow continues
  - SOAPAsyncRequest and SOAPAsyncResponse nodes are used to call a web service asynchronously allowing multiple requests to be handled in parallel
- Properties on SOAP nodes control the processing and can be configured by supplying a WSDL definition, or by manually configuring properties
- Policy sets and bindings define and configure WS-Security requirements, supported by WebSphere Message Broker, for SOAP nodes

Figure 9-26. SOAP nodes overview WM643 / VM6431.0

Notes:

HTTP and SOAP nodes can both be used to interact with web services. Typically you use SOAP nodes when working with SOAP-based web services.

The figure describes the SOAP nodes available for sending web service requests and receiving web services responses.

Calling a web service asynchronously means that the SOAPAsyncRequest node sends a web service request, but the request does not block the message flow by waiting for the associated web service response to be received because the web service response is received at the SOAPAsyncResponse node, which is in a separate flow. Multiple requests can, therefore, be handled in parallel.

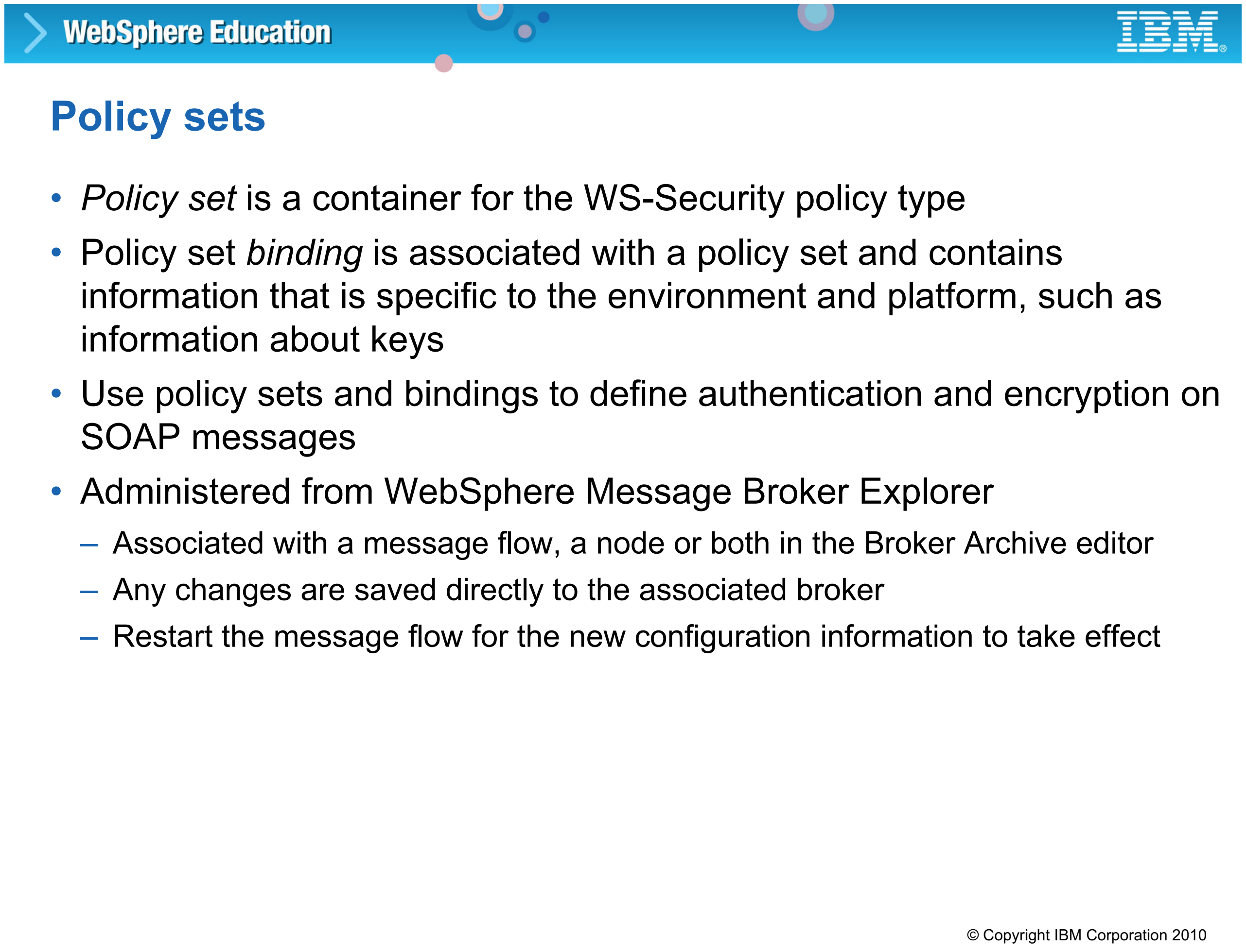
Instructor notes:

Purpose — Provides overview of SOAP nodes as introduction to policy sets.

Details — The SOAPEnvelope and SOAPExtract nodes are not included in the lists as they do not apply when discussing policy sets.

Additional information —

Transition statement — What is a policy set?



WebSphere Education

Policy sets

- *Policy set* is a container for the WS-Security policy type
- Policy set *binding* is associated with a policy set and contains information that is specific to the environment and platform, such as information about keys
- Use policy sets and bindings to define authentication and encryption on SOAP messages
- Administered from WebSphere Message Broker Explorer
  - Associated with a message flow, a node or both in the Broker Archive editor
  - Any changes are saved directly to the associated broker
  - Restart the message flow for the new configuration information to take effect

Figure 9-27. Policy sets

WM643 / VM6431.0

Notes:

Policy sets and bindings define and configure WS-Security requirements, supported by WebSphere Message Broker, for the SOAPInput, SOAPReply, SOAPRequest, SOAPAsyncRequest, and SOAPAsyncResponse nodes.

A *policy set* is a container for the WS-Security policy type.

A *policy set binding* is associated with a policy set and contains information that is specific to the environment and platform, such as information about keys.

Either the whole SOAP message body, or specific parts of the SOAP message header and body can be encrypted and signed.

You administer policy sets and bindings from WebSphere Message Broker Explorer, which can add, delete, display, and edit policy sets and bindings. Any changes to policy sets or bindings are saved directly to the associated broker. You must stop and then restart the message flow for the new configuration information to take effect.

Instructor notes:

Purpose — Introduce the concepts of policy sets and policy set bindings to support WS-Security for SOAP nodes.

Details — Focus on the administration aspects of policy sets and policy set bindings.

Additional information — Use policy sets and bindings to define the following items for both request and response SOAP messages:

Authentication for the following tokens:

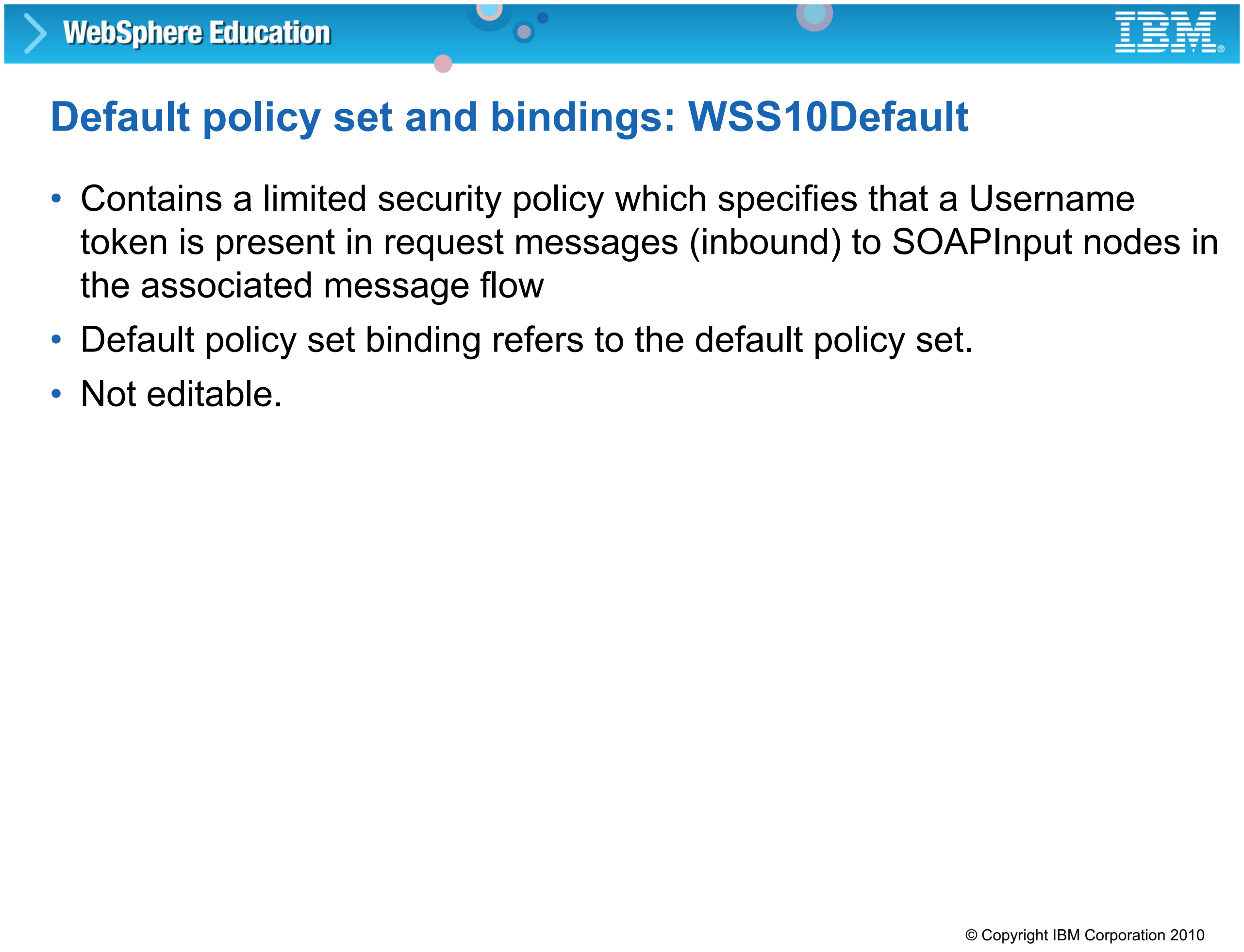
- User name tokens (requires a security profile to specify the external security provider)
- X.509 certificates (requires the broker keystore and truststore, or a security profile to specify the external security provider)
- SAML assertions, using SAML 1.1 or 2.0 pass-through (requires a security profile to specify the external security provider)
- LTPA tokens, using LTPA pass-through (requires a security profile to specify the external security provider)

Asymmetric encryption (confidentiality) using X.509 certificates (requires the broker keystore and truststore)

Symmetric encryption (confidentiality) using Kerberos tokens (requires the host to be configured for Kerberos)

Asymmetric signature (integrity) (requires the broker keystore and truststore)

Transition statement — A default policy set and policy set binding is provided with WebSphere Message Broker.



Default policy set and bindings: WSS10Default

- Contains a limited security policy which specifies that a Username token is present in request messages (inbound) to SOAPInput nodes in the associated message flow
- Default policy set binding refers to the default policy set.
- Not editable.

Figure 9-28. Default policy set and bindings: WSS10Default WM643 / VM6431.0

Notes:

WebSphere Message Broker includes a default policy set and policy set binding, named WSS10Default, that contains a limited security policy.

The WSS10Default policy set and policy set binding cannot be modified.

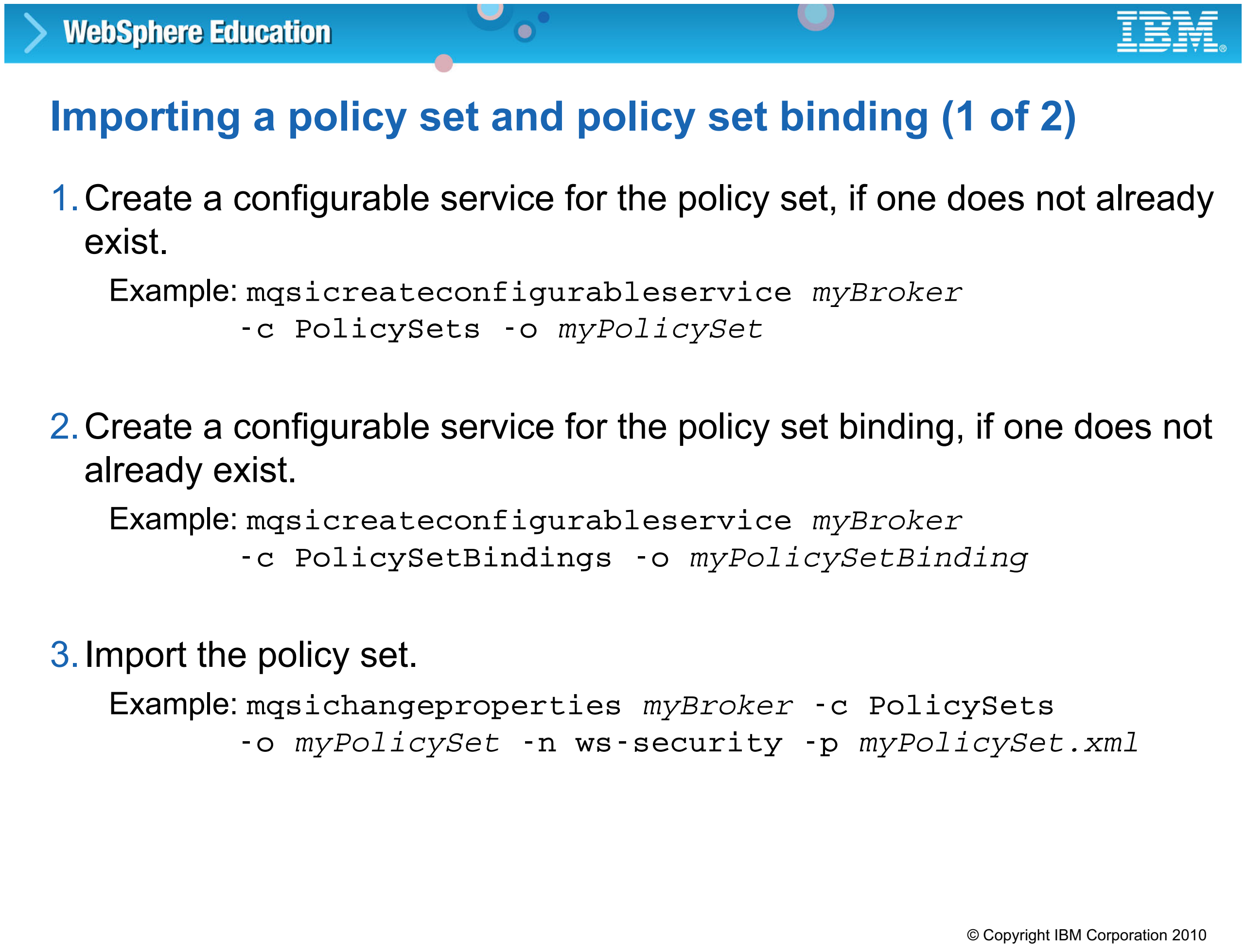
Instructor notes:

Purpose — Explain how the default policy set and policy set binding can be used to provide some security.

Details — None.

Additional information — None.

Transition statement — Policy sets and be imported and exported.



WebSphere Education

Importing a policy set and policy set binding (1 of 2)

1. Create a configurable service for the policy set, if one does not already exist.

   Example:
   ```
   mqsicreateconfigurableservice myBroker
       -c PolicySets -o myPolicySet
   ```

2. Create a configurable service for the policy set binding, if one does not already exist.

   Example:
   ```
   mqsicreateconfigurableservice myBroker
       -c PolicySetBindings -o myPolicySetBinding
   ```

3. Import the policy set.

   Example:
   ```
   mqsichangeproperties myBroker -c PolicySets
       -o myPolicySet -n ws-security -p myPolicySet.xml
   ```

Figure 9-29. Importing a policy set and policy set binding (1 of 2) WM643 / VM6431.0

Notes:

This figure and the next shows the steps for importing an existing policy set and policy set binding from the command line.

The examples show how to import policy set `myPolicySet` to broker `myBroker` from a file called `myPolicySet.xml`. The associated binding is `myPolicySetBinding`, which is imported from `myPolicySetBinding.xml`.

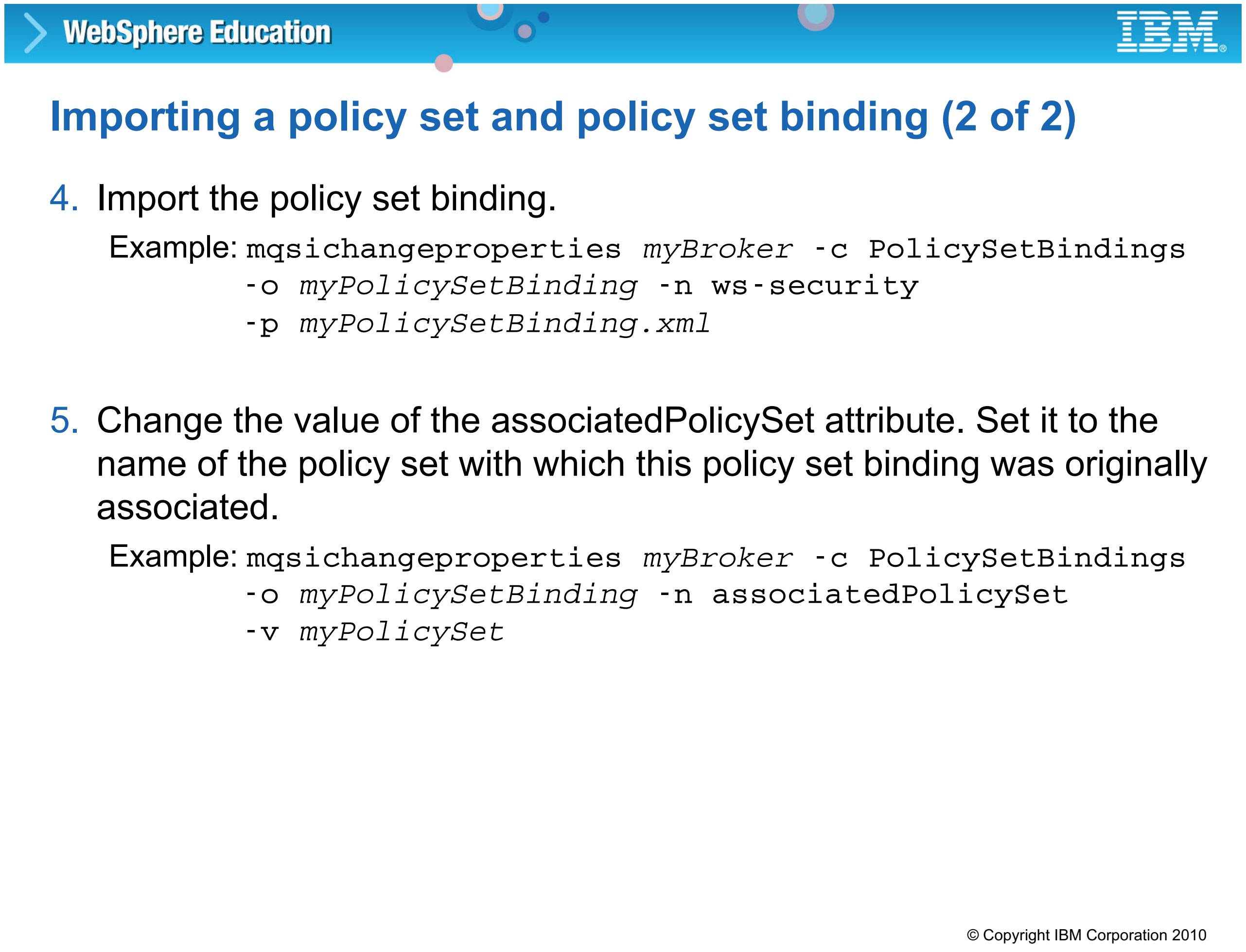
Instructor notes:

Purpose — Describe the steps for importing a policy set and policy set binding.

Details — None.

Additional information — None.

Transition statement — The remaining steps are shown on the next page.



WebSphere Education

Importing a policy set and policy set binding (2 of 2)

4. Import the policy set binding.

   Example:
   ```
   mqsichangeproperties myBroker -c PolicySetBindings
   -o myPolicySetBinding -n ws-security
   -p myPolicySetBinding.xml
   ```

5. Change the value of the associatedPolicySet attribute. Set it to the name of the policy set with which this policy set binding was originally associated.

   Example:
   ```
   mqsichangeproperties myBroker -c PolicySetBindings
   -o myPolicySetBinding -n associatedPolicySet
   -v myPolicySet
   ```

Figure 9-30. Importing a policy set and policy set binding (2 of 2) WM643 / VM6431.0

Notes:

The next steps are to import the policy set binding and then associate the binding with the imported policy set.

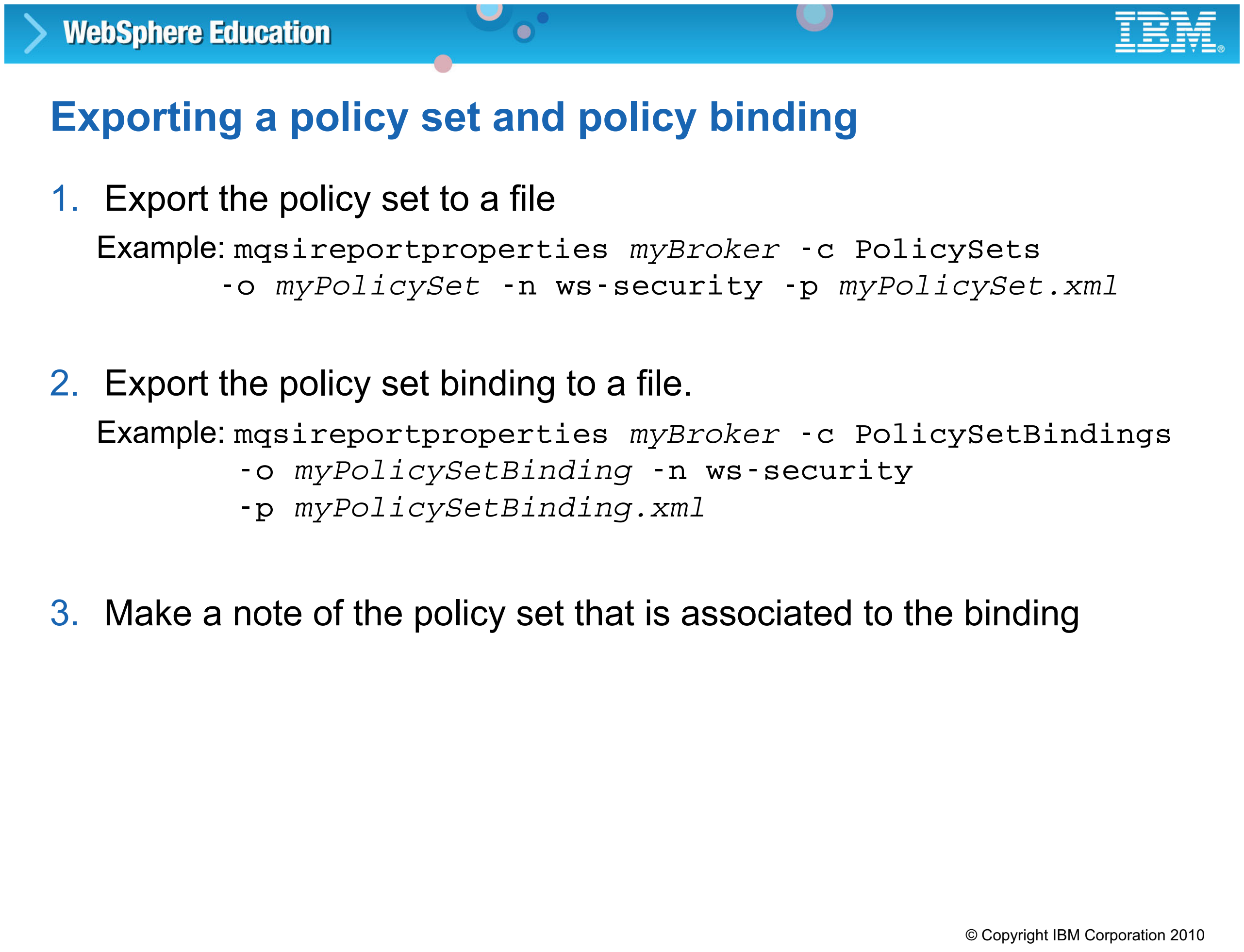
Instructor notes:

Purpose — List remaining steps for importing policy set and policy set binding.

Details — None.

Additional information — None.

Transition statement — Next are the steps for exporting.



WebSphere Education

Exporting a policy set and policy binding

1. Export the policy set to a file

   Example:
   ```
   mqsireportproperties myBroker -c PolicySets
         -o myPolicySet -n ws-security -p myPolicySet.xml
   ```

2. Export the policy set binding to a file.

   Example:
   ```
   mqsireportproperties myBroker -c PolicySetBindings
          -o myPolicySetBinding -n ws-security
          -p myPolicySetBinding.xml
   ```

3. Make a note of the policy set that is associated to the binding

Figure 9-31. Exporting a policy set and policy set binding WM643 / VM6431.0

Notes:

Use the `mqsireportproperties` command to export a policy set and associated binding to a file.

The examples in the figure show how to export policy set `myPolicySet` from broker `myBroker` to a file called `myPolicySet.xml`. The associated binding is `myPolicySetBinding`, which you export to `myPolicySetBinding.xml`.

The final step is to make a note of the policy set that is associated to the binding; you will need this information when you import the policy set and binding.

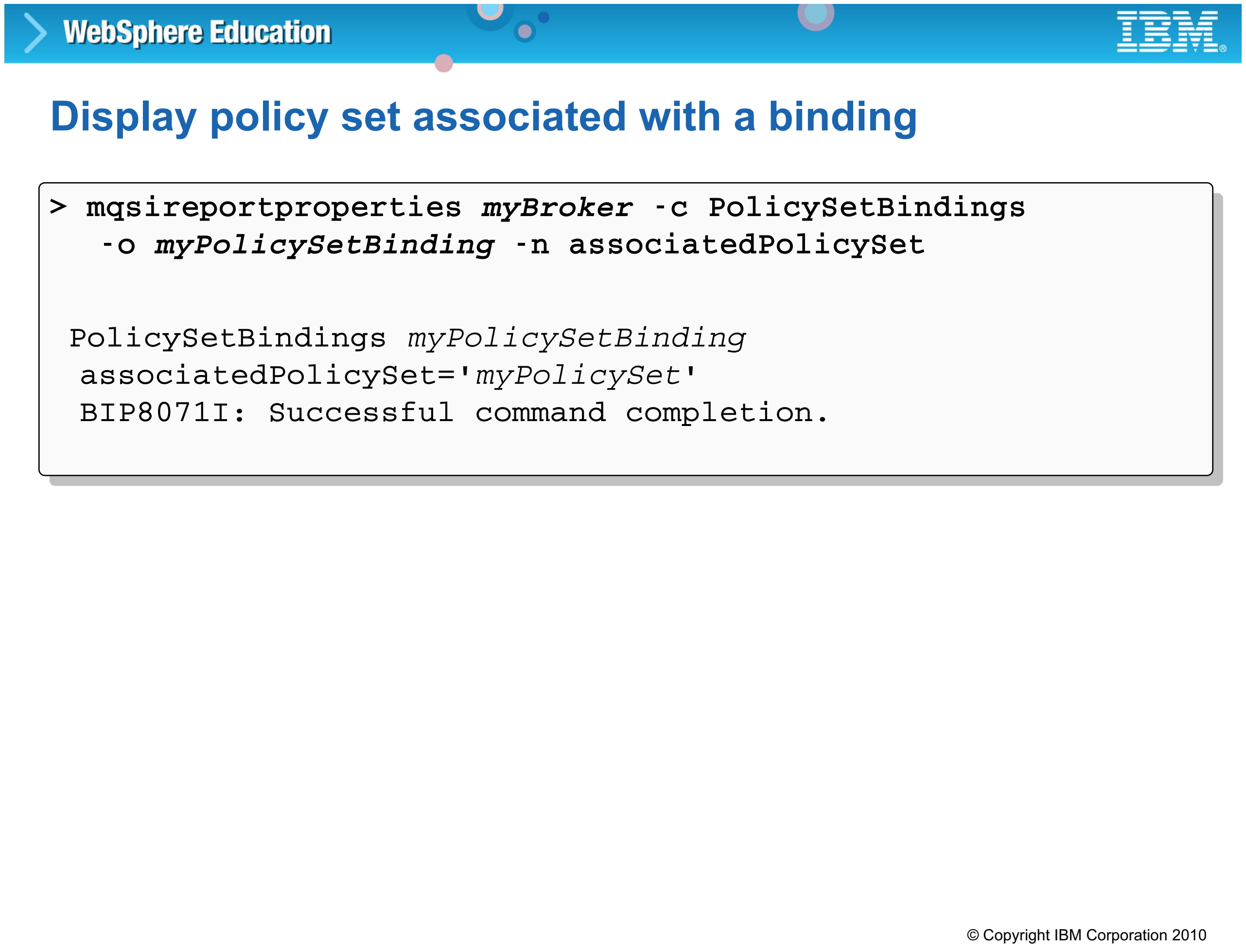
Instructor notes:

Purpose — List steps for exporting a policy set and binding.

Details — None.

Additional information — None.

Transition statement — You can use the `mqsireportproperties` command to identify the policy set associated with the binding.



WebSphere Education

Display policy set associated with a binding

```
> mqsireportproperties myBroker -c PolicySetBindings
   -o myPolicySetBinding -n associatedPolicySet


 PolicySetBindings myPolicySetBinding
  associatedPolicySet='myPolicySet'
  BIP8071I: Successful command completion.
```

Figure 9-32. Display policy set associated with a binding WM643 / VM6431.0

Notes:

This `mqsireportproperties` command displays the policy set associated with a binding.

The example in the figure shows how to display the associated policy set for the policy set binding named `myPolicySetBinding` from the broker named `myBroker`.

Instructor notes:

Purpose — Emphasize the importance of maintaining the policy set and binding association when importing policy sets and binding.

Details — None.

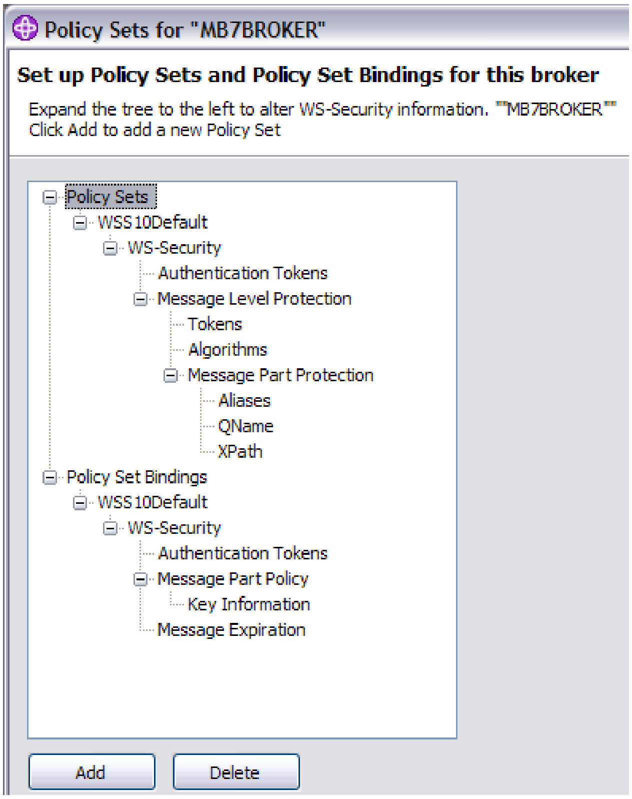
Additional information — None.

Transition statement — If you do not have an existing policy set and binding that fits the requirements, you can create one using WebSphere Message Broker Explorer.

WebSphere Education IBM

Defining policy sets in WebSphere Message Broker Explorer

- Policy Sets and Policy Set Bindings editor
  - For creating and editing a policy set or binding
  - Can define all the WS-Security policies for a single node, or a set of nodes
  - Policy set can be associated by the administrator with either a message flow or a node



Policy Sets for "MB7BROKER"

Set up Policy Sets and Policy Set Bindings for this broker

Expand the tree to the left to alter WS-Security information. ""MB7BROKER""
Click Add to add a new Policy Set

```
⊟ Policy Sets
   ⊟ WSS10Default
      ⊟ WS-Security
         Authentication Tokens
         ⊟ Message Level Protection
            Tokens
            Algorithms
            ⊟ Message Part Protection
               Aliases
               QName
               XPath
⊟ Policy Set Bindings
   ⊟ WSS10Default
      ⊟ WS-Security
         Authentication Tokens
         ⊟ Message Part Policy
            Key Information
         Message Expiration
```

Figure 9-33. Defining policy sets in Message Broker Explorer WM643 / VM6431.0

Notes:

WebSphere Message Broker Explorer includes editor for creating and modifying policy sets and policy set bindings.

Note

You must connect to a broker before you can edit policy sets and bindings.

Instructor notes:

Purpose — Provide an overview of the policy set and binding editors.

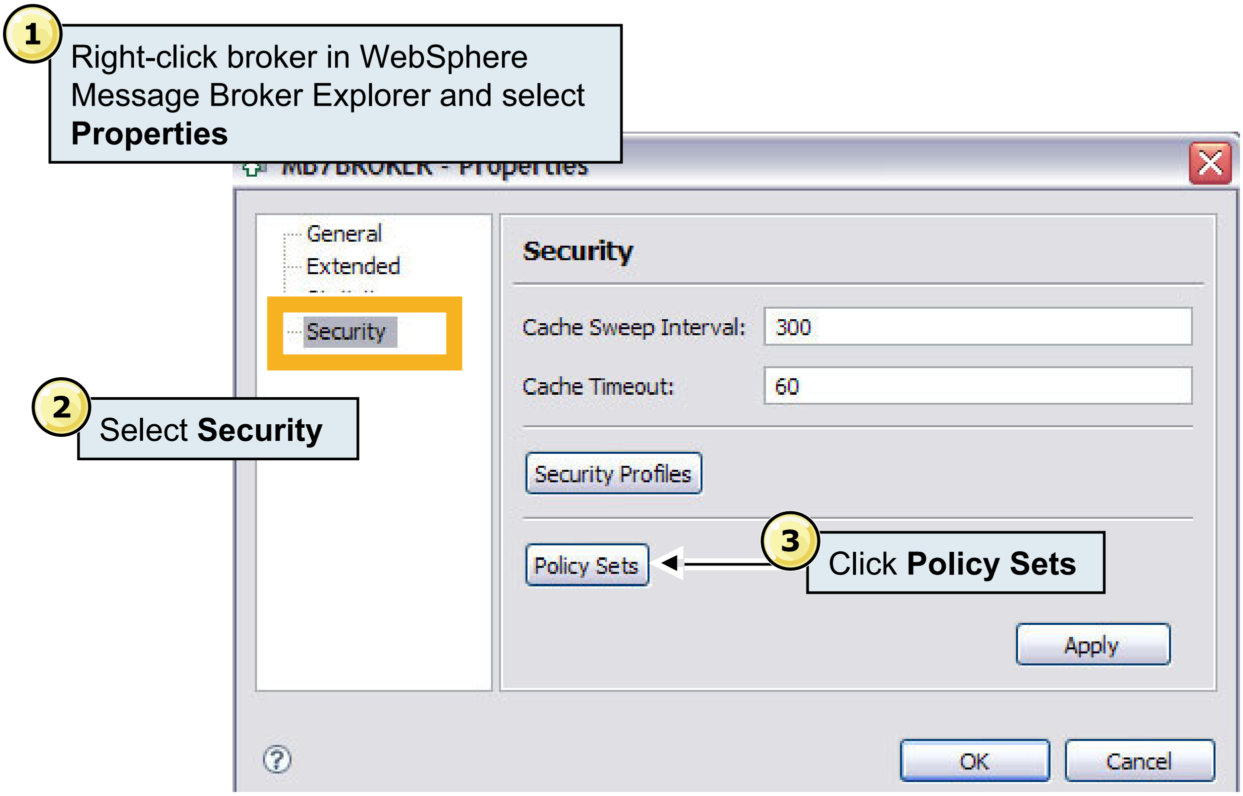
Details — None.

Additional information — None.

Transition statement — How do you access the editors in WebSphere Message Broker Explorer.

WebSphere Education IBM

Accessing policy sets and policy set bindings



1 Right-click broker in WebSphere Message Broker Explorer and select **Properties**

2 Select **Security**

3 Click **Policy Sets**

Figure 9-34. Accessing policy sets and policy set bindings WM643 / VM6431.0

Notes:

To launch the Policy Sets and Policy Set Bindings editor in WebSphere Message Broker Explorer:

1. In the WebSphere Message Broker Explorer Navigator view, right-click the broker with which you want to work, and select **Properties** from the menu.
2. Select the **Security** tab.
3. Click **Policy Sets**.

Instructor notes:

Purpose — Show steps for accessing policy set and bindings editors in WebSphere Message Broker Explorer.

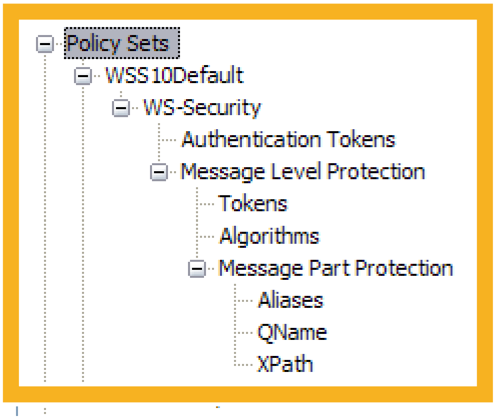
Details — None.

Additional information — None.

Transition statement — The first editor to look at is the Policy Set editor.

WebSphere Education IBM

Policy set editor



- Policy Sets
  - WSS10Default
    - WS-Security
      - Authentication Tokens
      - Message Level Protection
        - Tokens
        - Algorithms
        - Message Part Protection
          - Aliases
          - QName
          - XPath

- **Authentication Tokens** to create Username, X.509, SAML, and LTPA authentication tokens
- **Message Level Protection** to apply signatures and encryption to the whole message, whether inbound or outbound
- **Message Part Protection** to define the parts of a message that encryption and signature apply to
  - **Aliases** to refer to an alias identified in a SOAPInput, SOAPRequest, or SOAPAsyncRequest node
  - **Qname** to define namespaces, and optional elements within those namespaces, within the SOAP message header for which encryption or signature applies
  - **Xpath** to define an XPath expression that refers to an element in the message to which encryption or signature applies

Figure 9-35. Policy set editor WM643 / VM6431.0

Notes:

The Policy set editor contains a number of panels for defining authentication tokens, message level protection, and message part protection.

To create a policy set, select Policy Sets and then click **Add**. A policy set is added with a default name which can be changed.

Instructor notes:

Purpose — Provides an overview of the Policy set editor panels.

Details — The interface includes a number of drop-down menus for selecting options. Open the Policy set editor and quickly show each of the panels.

Additional information — None.

Transition statement — After you have created the policy set, the next step is to create the policy set binding using the Policy set binding editor.

Policy set binding editor

- Associate a policy set binding with a policy set
  - **Authentication Tokens** to further configure any X.509 authentication tokens defined in the associated policy set
  - **Message Part Policy** to further configure any message part protection tokens defined in the associated policy set
  - **Key Information** to further configure any message level protection tokens that are defined in the associated policy set
  - **Message Expiration** to define settings to expire message after the specified time interval

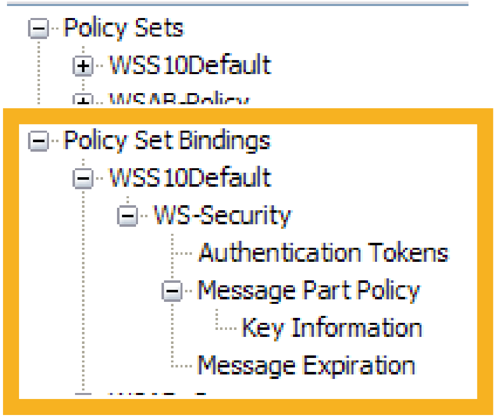


Figure 9-36. Policy set binding editor WM643 / VM6431.0

Notes:

The Policy set binding editor is used to create and maintain policy set bindings. Part of the definition includes associating a policy set binding with a policy set when the policy set binding is added. Information about the binding is then displayed on the main panel of the policy set binding.

You cannot change the policy set to which a binding is associated. You can delete the policy set binding independently of the policy set. However, deleting a policy set also deletes the associated policy set binding.

To create a policy set binding, select **Policy Set Binding** and then click **Add**. A policy set binding is added with a default name which can be changed.

Instructor notes:

Purpose — Summarize the panels in the Policy set binding editor.

Details — To demonstrate, you can open the Policy set binding editor and create a binding.

Additional information — None.

Transition statement — Web services message flows might also include JMS nodes which requires some administrator action.

9.6. Preparing to use JMS nodes

Instructor topic introduction

Some of the information in this topic is review from the prerequisite course, WM201, WebSphere MQ V7 System Administration. It is only included in this unit as a refresher.

What students will do — Students will learn about the JMS nodes and understand the things they must do to make sure that the nodes can be used. They will also learn that they can alter some of the node configurations using `.bar` file changes

How students will do it — Students will listen to a brief lecture and then have an opportunity to complete an exercise to see how to implement JMS node flows.

What students will learn — Students will learn that they must do some work before any JMS nodes can be used in message flows. They will learn how to do this configuration work.

How this will help students on their job — Students will be able to administer flows that use JMS nodes.

WebSphere Education IBM

JMS architecture

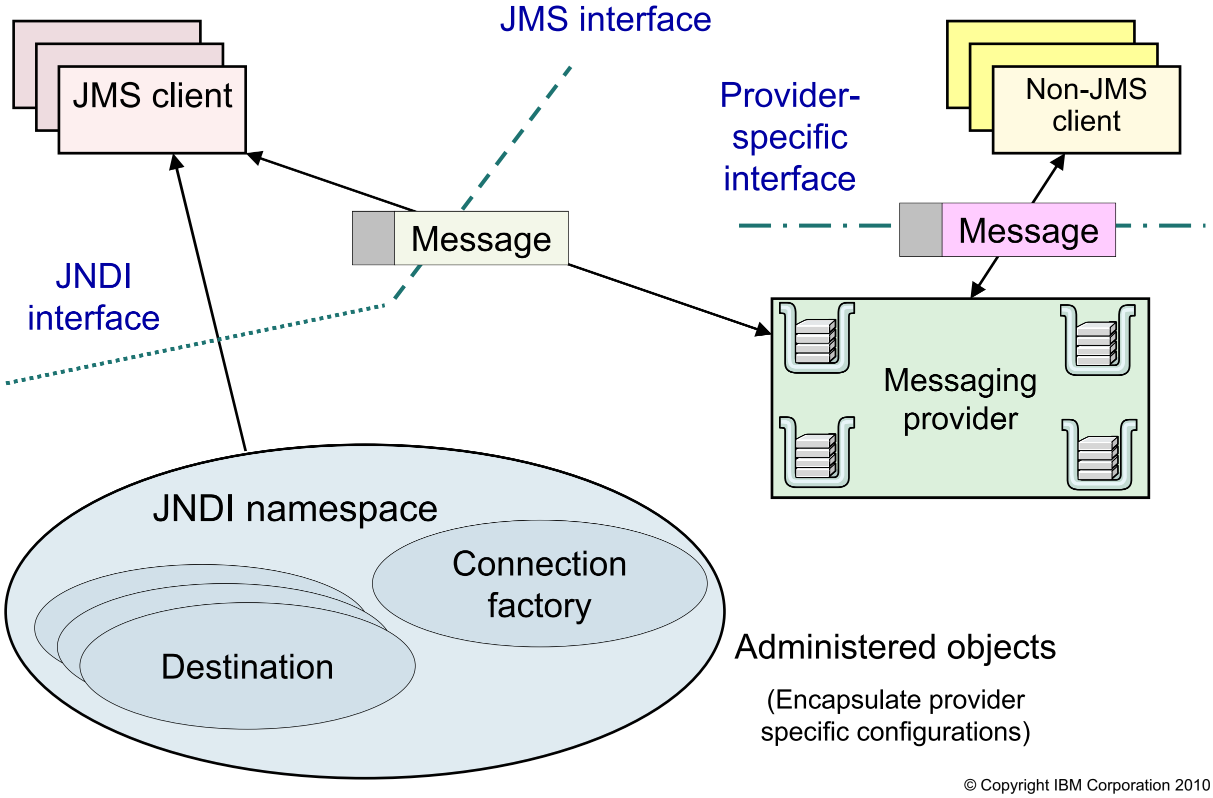


Figure 9-37. JMS architecture WM643 / VM6431.0

Notes:

Messaging applications may consist of the following parts:

- Messaging provider: The enterprise messaging system, for example, WebSphere MQ
- JMS clients: Java programs producing and consuming messages using the JMS
- Non-JMS clients: Programs that use the messaging product API to produce and consume messages
- Messages: The objects that communicate information between clients
- Administered objects: Preconfigured JMS objects created by an administrator for the use of clients

Instructor notes:

Purpose — Explain differences in JMS architecture, as compared to WebSphere MQ. Some understanding is necessary to configure JMS nodes in the broker.

Details — JNDI (Java Naming and Directory Interface) is a standard Java extension that provides a uniform API for accessing various directory and naming services. JMS clients use JNDI to browse a naming service in order to obtain references to administered objects. Administered objects are the JMS connection factory and JMS destination objects, where JMS destination objects are topics and queues. Administered objects are created and configured by a system administrator. To create and configure JNDI administered objects, see the JMS provider documentation.

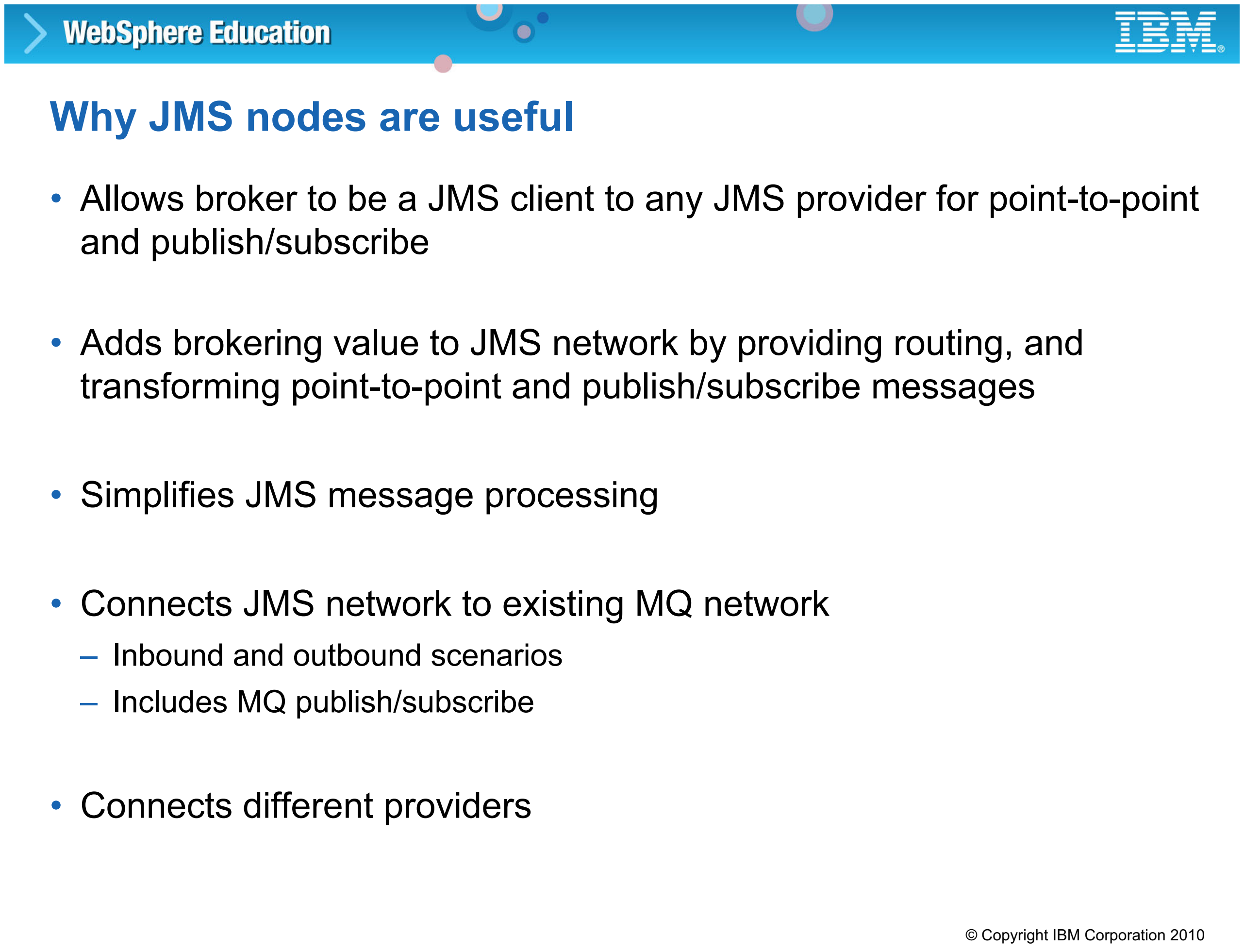
JNDI administered objects are stored in the bindings. This can be either file system based or LDAP based. A naming service associates names with distributed objects so that the administered objects are located by using names and not complex network addresses.

JNDI provides an abstraction that hides the specifics of the naming service, which makes client applications more portable.

A JMS client specifies a JNDI InitialContext to obtain a JNDI connection to the JMS messaging server. The InitialContext is the starting point in any JNDI lookup and acts like the root of a file system. The JMS directory service that is being used determines the properties that are used to create an InitialContext

Additional information — None.

Transition statement — Why are there extra JMS nodes in WebSphere Message Broker?



WebSphere Education

IBM

Why JMS nodes are useful

- Allows broker to be a JMS client to any JMS provider for point-to-point and publish/subscribe
- Adds brokering value to JMS network by providing routing, and transforming point-to-point and publish/subscribe messages
- Simplifies JMS message processing
- Connects JMS network to existing MQ network
  - Inbound and outbound scenarios
  - Includes MQ publish/subscribe
- Connects different providers

© Copyright IBM Corporation 2010

Figure 9-38. Why JMS nodes are useful WM643 / VM6431.0

Notes:

The JMS nodes can be used in applications where messages are produced and consumed from a variety of JMS destinations. In sending and receiving messages, the JMS nodes behave like JMS clients.

The JMS nodes work with the WebSphere MQ JMS provider, WebSphere Application Server, the IBM Service Integration Bus, and any JMS provider that conforms to JMS 1.1.

In order for these nodes to participate in a WebSphere Message Broker coordinated transaction, the JMS provider must support the XAResource interface as defined in JMS 1.1.

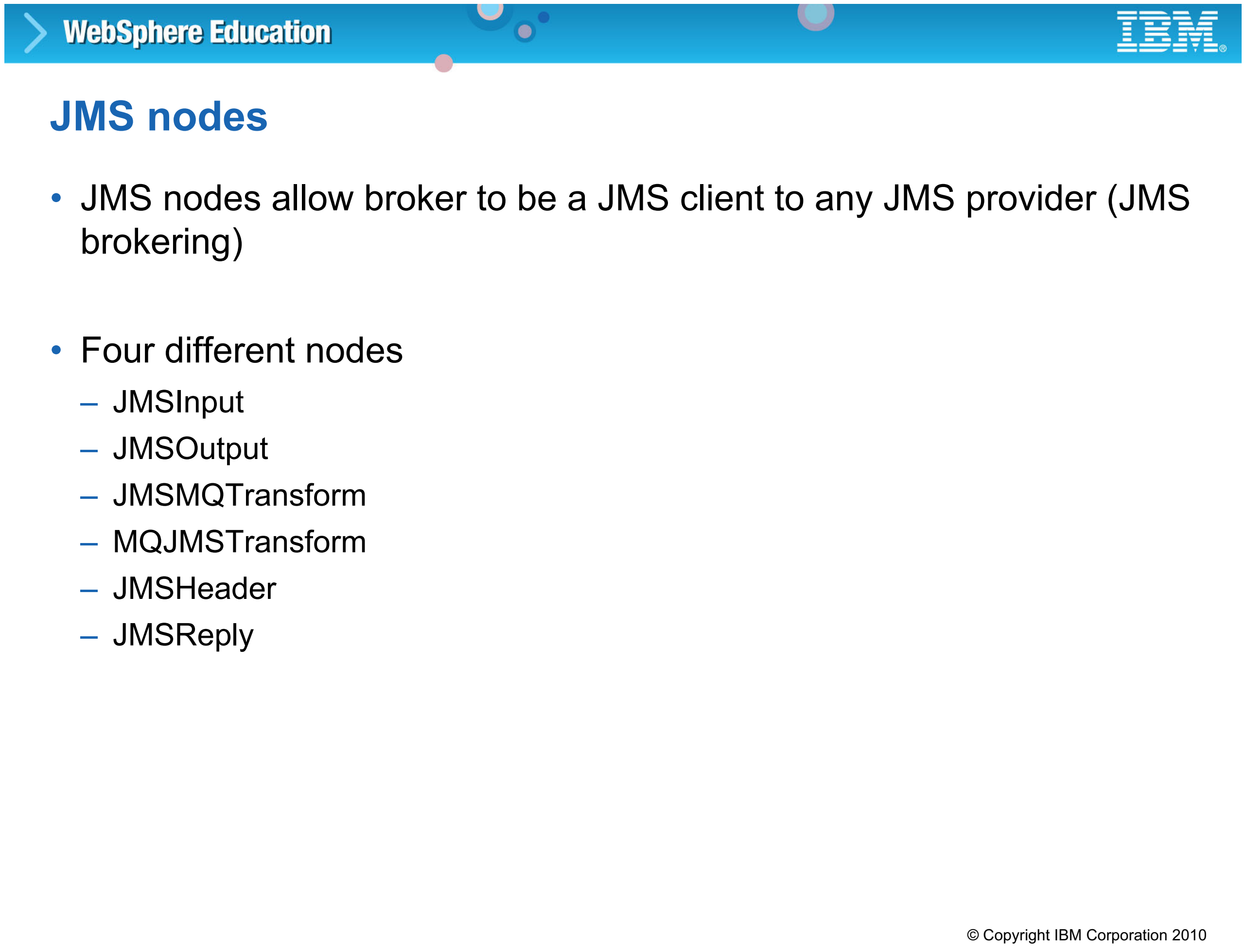
Instructor notes:

Purpose — Discuss advantages of JMS nodes.

Details — None.

Additional information — None.

Transition statement —



WebSphere Education

JMS nodes

- JMS nodes allow broker to be a JMS client to any JMS provider (JMS brokering)

- Four different nodes
  - JMSInput
  - JMSOutput
  - JMSMQTransform
  - MQJMSTransform
  - JMSHeader
  - JMSReply

Figure 9-39. JMS nodes

WM643 / VM6431.0

Notes:

WebSphere Message Broker has a number of nodes that offer brokering in a JMS network. These nodes provide support to the broker, so that it can act as a JMS client.

JMS messages can be sent (JMSOutput node) and received (JMSInput node), and can be transformed into other message formats (JMSMQTransform node) or from other formats into a JMS format (MQJMSTransform node).

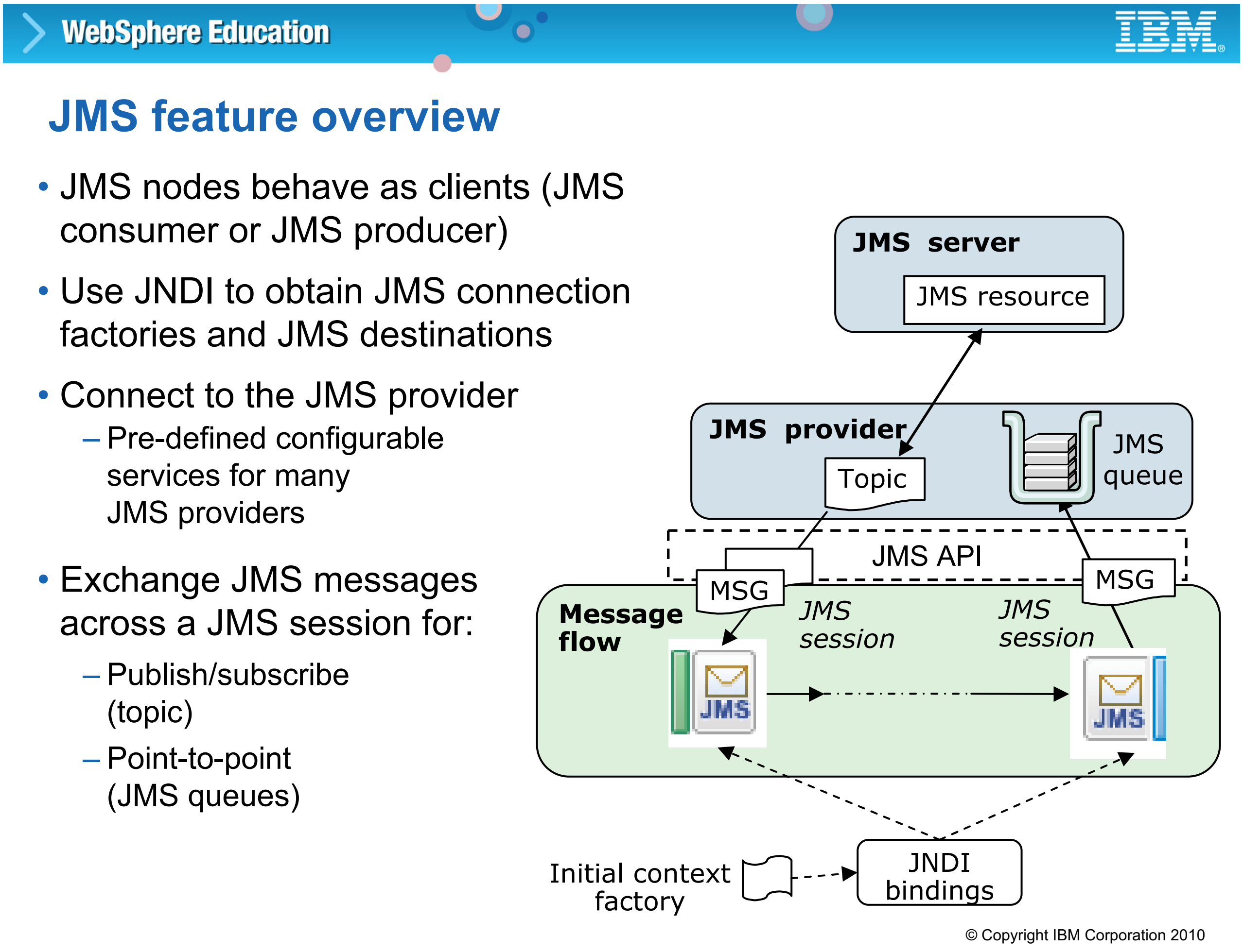
Instructor notes:

Purpose — List the JMS nodes.

Details — None.

Additional information — None.

Transition statement — Perhaps a picture will show how JMS works.



WebSphere Education

JMS feature overview

- JMS nodes behave as clients (JMS consumer or JMS producer)
- Use JNDI to obtain JMS connection factories and JMS destinations
- Connect to the JMS provider
  - Pre-defined configurable services for many JMS providers
- Exchange JMS messages across a JMS session for:
  - Publish/subscribe (topic)
  - Point-to-point (JMS queues)

Figure 9-40. JMS feature overview

WM643 / VM6431.0

Notes:

JNDI is a standard Java extension providing an API to access variety of directory and naming services.

JMS clients implement JNDI to get references to administered objects. Administered objects are JMS connection factory and JMS destination objects (topics and queues). Administrators are responsible to setting up administered objects.

Configuration of JNDI administered objects must be documented by each JMS provider. For the WebSphere MQ JMS provider, a sample JMSAdmin definitions file is included with WebSphere MQ and described in the WebSphere MQ Using Java book.

The JNDI administered objects are stored in bindings, either file system based or LDAP based. LDAP is part of X.500, which is a standard for directory services in a network.

Naming services link names with distributed objects so that the administered objects can be located using names and not complex network addresses. JNDI masks the complexity of the naming service, allowing for more portable client applications.

JMS clients specify an InitialContext to obtain a JNDI connection to the JMS messaging server. This InitialContext is the starting point, acting as the root of a file system. The JMS directory service that is being used determines the properties that are used to create an InitialContext.

JMS clients operate with both publish/subscribe and point-to-point messages. Both of these communication models use virtual channels called destinations. In the publish/subscribe model, the destinations are topics. For the point-to-point model, the destinations are known as queues.

The following application communication model properties can be configured for JMSInput/and JMSOutput nodes (this is a table taken from the broker online help):

Property	Description
Connection factory name	A string name that is passed to JNDI to look up the administered connection factory object. The connection factory object is used to create a connection to the JMS destination. For a client operating as a publish/subscribe client, the connection factory name will be for a TopicConnectionFactory. For a client operating as a point-to-point client, the connection factory name will be for a QueueConnectionFactory.
Subscription topic	The string name that is passed to JNDI to look up the JMS topic destination. The topic is used to create a JMS session when the node is being used to process publish/subscribe messages.
Durable subscription ID	A durable subscription is one that outlasts the client's connection to a message server. When a durable subscriber is disconnected from the server, it is the responsibility of the server to store messages that are published. Therefore, when the durable subscriber reconnects, the message server sends all the unexpired messages. Durable subscriptions cannot be unsubscribed from a message flow. It requires a separate administration task to unsubscribe a previously registered durable subscription. Some JMS providers supply an administration tool to perform this action.
Source queue	The string name that is passed to JNDI to look up the JMS queue destination. The queue is used to create a JMS session when the node is being used to process point-to-point messages.

Instructor notes:

Purpose — Offer some terminology descriptions and an overview.

Details — Try not to get too involved. It is important to point out some of these terms, though, because they are things that the administrator might need to configure.

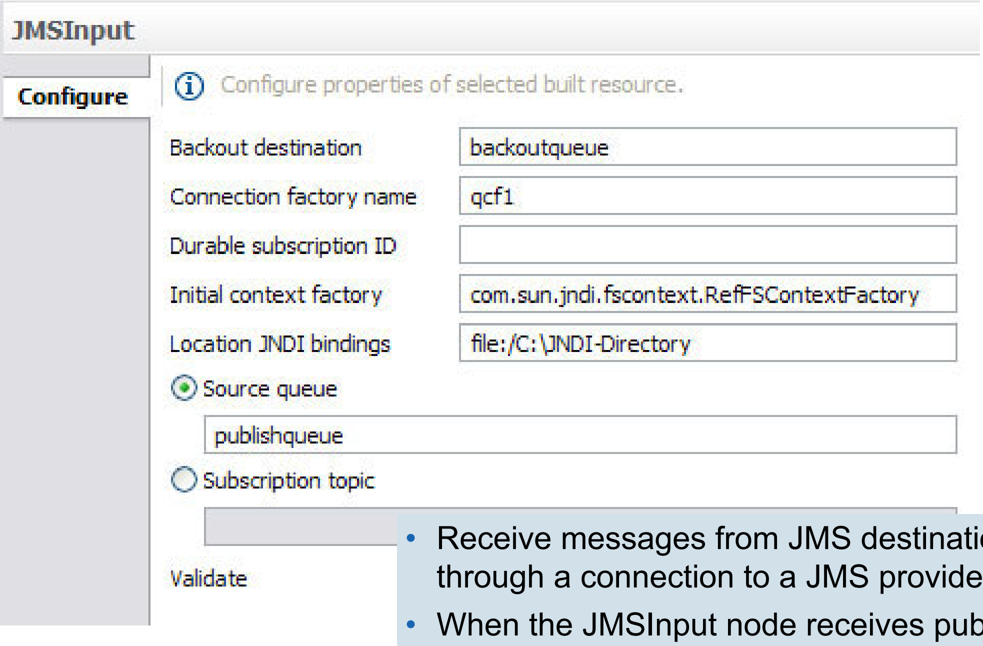
Additional information — Subscription topic and source queue properties are mutually exclusive because they configure the node to work with either the publish/subscribe message model or the point-to-point message model.

A durable subscription ID is not valid without a subscriber topic property.

Transition statement — Now you look at the nodes that implement the JMS mechanism.

WebSphere Education IBM

JMSInput node



JMSInput

Configure

ⓘ Configure properties of selected built resource.

Backout destination	backoutqueue
Connection factory name	qcf1
Durable subscription ID	
Initial context factory	com.sun.jndi.fscontext.RefFSContextFactory
Location JNDI bindings	file:/C:\JNDI-Directory

● Source queue

publishqueue

○ Subscription topic

Validate

- Receive messages from JMS destinations accessed through a connection to a JMS provider
- When the JMSInput node receives publication topics, it internally restricts the message flow property **Additional Instances** to zero to prevent the receipt of duplicate publications

Figure 9-41. JMSInput node WM643 / VM6431.0

Notes:

The JMSInput node receives message from JMS destinations accessed through a connection to a JMS provider. WebSphere Message Broker includes configurable services for many JMS providers or you can create a new configurable service. You can view the configurable services and definitions from the WebSphere Message Broker Explorer.

In the JMSInput node, the incoming message is coming from a JMS provider. Typically, it is reformatted during the flow into another format. However, the JMSInput node permits the establishment of the properties that tell where the JMS message is coming from.

The figure shows the JMSInput node properties that can be modified in the BAR file by the administrator.

Instructor notes:

Purpose — Show the JMSInput node.

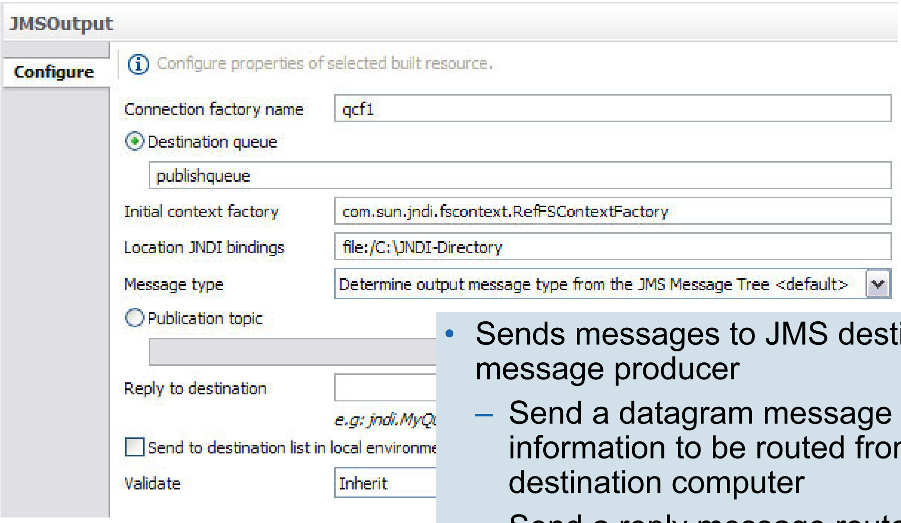
Details — None.

Additional information — None.

Transition statement — It is also possible to send data to a JMS provider.

WebSphere Education

JMSOuput node



JMSOutput

Configure

Configure properties of selected built resource.

Connection factory name: qcf1

● Destination queue

publishqueue

Initial context factory: com.sun.jndi.fscontext.RefFSContextFactory

Location JNDI bindings: file:/C:\JNDI-Directory

Message type: Determine output message type from the JMS Message Tree <default>

○ Publication topic

Reply to destination

e.g: jndi.MyQ

□ Send to destination list in local environme

Validate: Inherit

- Sends messages to JMS destinations and acts as a message producer
  - Send a datagram message that carries sufficient information to be routed from the source to the destination computer
  - Send a reply message routed according to the value in the **JMSReplyTo** property from the request message
  - Send a request message to a JMS destination with the expectation of a response from the message consumer that processes the request

Figure 9-42. JMSOutput node

Notes:

The JMSOutput node is used to route data to a JMS provider. Typically, the data has started out in some other format and has been converted in the flow to a JMS tree.

The figure shows the JMSOutput node properties that can be modified in the BAR file by the administrator.

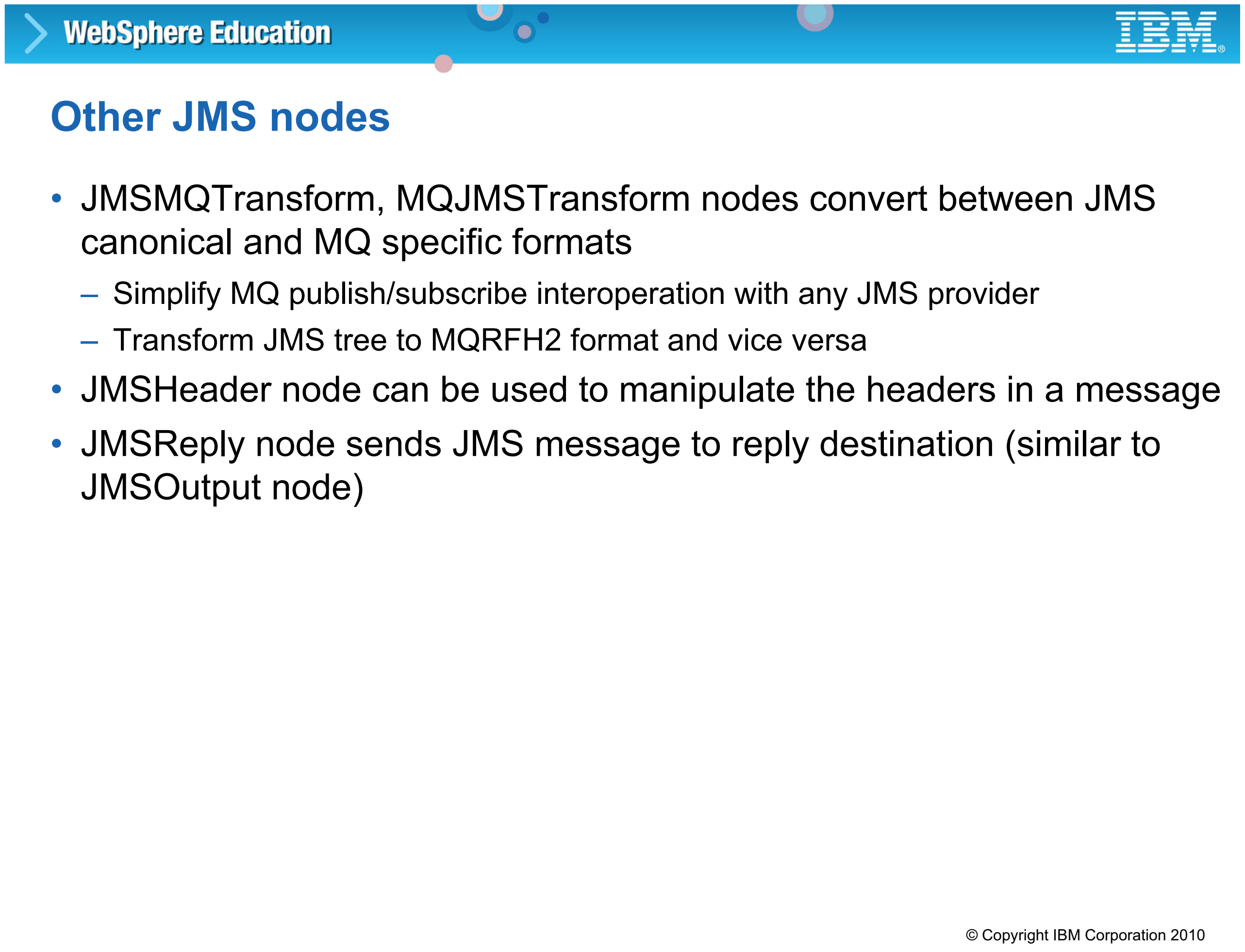
Instructor notes:

Purpose — Show the JMSOutput node.

Details — None.

Additional information — None.

Transition statement — There are other JMS nodes including nodes that transform the message from or to a JMS message tree.



WebSphere Education

Other JMS nodes

- JMSMQTransform, MQJMSTransform nodes convert between JMS canonical and MQ specific formats
  - Simplify MQ publish/subscribe interoperation with any JMS provider
  - Transform JMS tree to MQRFH2 format and vice versa
- JMSHeader node can be used to manipulate the headers in a message
- JMSReply node sends JMS message to reply destination (similar to JMSOutput node)

Figure 9-43. Other JMS nodes WM643 / VM6431.0

Notes:

The JMSMQTransform and MQJMSTransform nodes have no properties other than descriptions. Their purpose is to take an incoming WebSphere MQ message and build a JMS tree or vice versa.

Instructor notes:

Purpose — Mention the other JMS nodes.

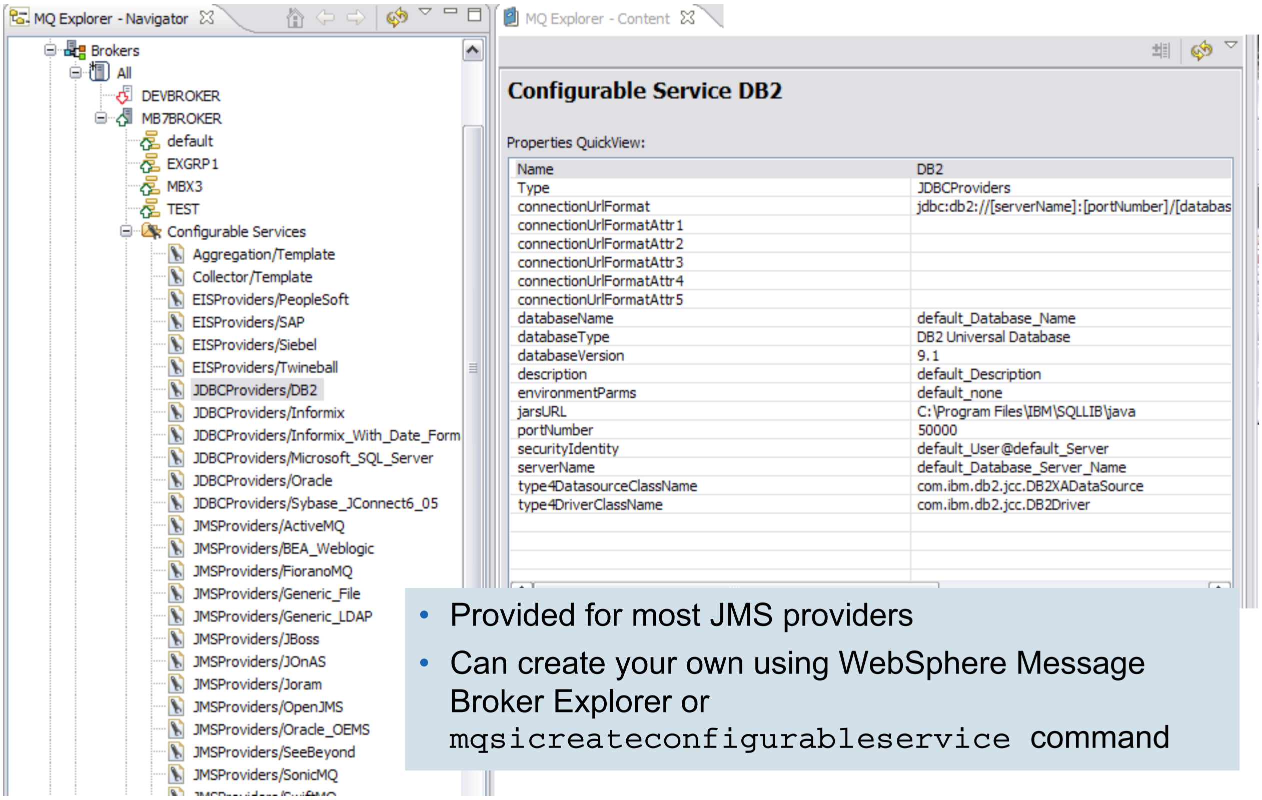
Details — None.

Additional information — None.

Transition statement — So, how does one configure the use of a JMS provider?

WebSphere Education

JMS provider configurable services



- Brokers
  - All
    - DEVBROKER
    - MB7BROKER
      - default
      - EXGRP1
      - MBX3
      - TEST
      - Configurable Services
        - Aggregation/Template
        - Collector/Template
        - EISProviders/PeopleSoft
        - EISProviders/SAP
        - EISProviders/Siebel
        - EISProviders/Twineball
        - JDBCProviders/DB2
        - JDBCProviders/Informix
        - JDBCProviders/Informix_With_Date_Form
        - JDBCProviders/Microsoft_SQL_Server
        - JDBCProviders/Oracle
        - JDBCProviders/Sybase_JConnect6_05
        - JMSProviders/ActiveMQ
        - JMSProviders/BEA_Weblogic
        - JMSProviders/FioranoMQ
        - JMSProviders/Generic_File
        - JMSProviders/Generic_LDAP
        - JMSProviders/JBoss
        - JMSProviders/JOnAS
        - JMSProviders/Joram
        - JMSProviders/OpenJMS
        - JMSProviders/Oracle_OEMS
        - JMSProviders/SeeBeyond
        - JMSProviders/SonicMQ

Configurable Service DB2

Properties QuickView:

Name	DB2
Type	JDBCProviders
connectionUrlFormat	jdbc:db2://[serverName]:[portNumber]/[databas
connectionUrlFormatAttr1	
connectionUrlFormatAttr2	
connectionUrlFormatAttr3	
connectionUrlFormatAttr4	
connectionUrlFormatAttr5	
databaseName	default_Database_Name
databaseType	DB2 Universal Database
databaseVersion	9.1
description	default_Description
environmentParms	default_none
jarsURL	C:\Program Files\IBM\SQLLIB\java
portNumber	50000
securityIdentity	default_User@default_Server
serverName	default_Database_Server_Name
type4DatasourceClassName	com.ibm.db2.jcc.DB2XADataSource
type4DriverClassName	com.ibm.db2.jcc.DB2Driver

- Provided for most JMS providers
- Can create your own using WebSphere Message Broker Explorer or `mqsicreateconfigurableservice` command

Figure 9-44. JMS provider configurable services WM643 / VM6431.0

Notes:

Configurable services are defined for a number of JMS providers. You can choose one of the predefined services, or you can create a service for a new provider, or for one of the existing providers.

If no service is defined for your JMS provider, or if you want to create another service for an existing JMS provider, use WebSphere Message Broker Explorer or the `mqsicreateconfigurableservice` command to identify the new service and to set its properties.

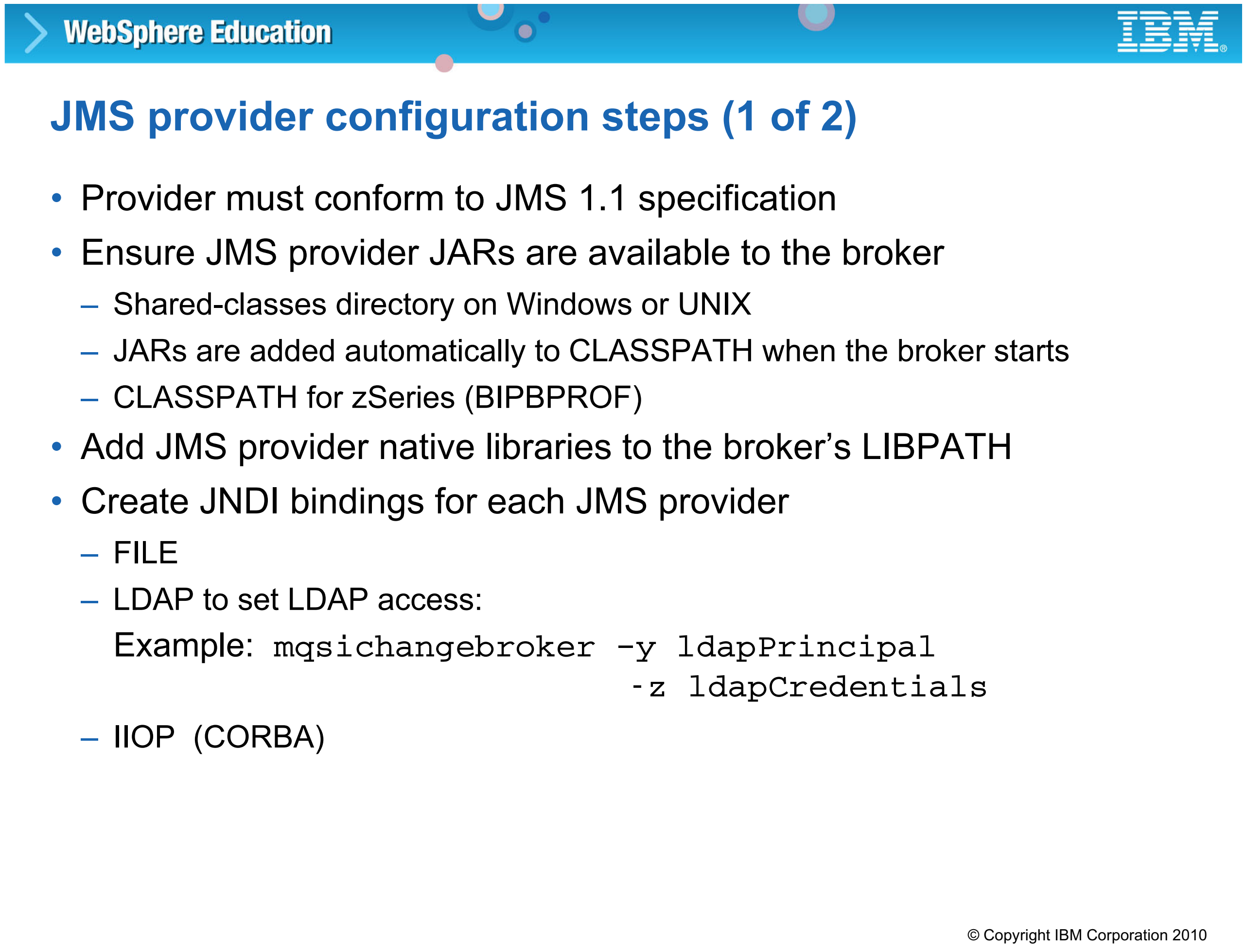
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —



WebSphere Education

JMS provider configuration steps (1 of 2)

- Provider must conform to JMS 1.1 specification
- Ensure JMS provider JARs are available to the broker
  - Shared-classes directory on Windows or UNIX
  - JARs are added automatically to CLASSPATH when the broker starts
  - CLASSPATH for zSeries (BIPBPROF)
- Add JMS provider native libraries to the broker's LIBPATH
- Create JNDI bindings for each JMS provider
  - FILE
  - LDAP to set LDAP access:

    Example:
    ```
    mqsichangebroker -y ldapPrincipal
                     -z ldapCredentials
    ```
  - IIOP (CORBA)

Figure 9-45. JMS provider configuration steps (1 of 2) WM643 / VM6431.0

Notes:

JMS providers document how to do configuration. This description is based on using WebSphere MQ as the JMS provider.

Configure the administration tool with values for the following properties:

- **INITIAL_CONTEXT_FACTORY** The service provider that the tool uses. Examples of supported values are:

  ```
  com.sun.jndi.ldap.LdapCtxFactory
  com.sun.jndi.fscontext.RefFSContextFactory
  com.ibm.ejs.ns.jndi.CNInitialContextFactory
  com.ibm.websphere.naming.WsnInitialContextFactory
  ```

- **PROVIDER_URL** The URL of the session's initial context; the root of all JNDI operations carried out by the tool.
- **SECURITY_AUTHENTICATION** Specifies whether JNDI passes security credentials to your service provider. This property is used only when an LDAP service provider is used.

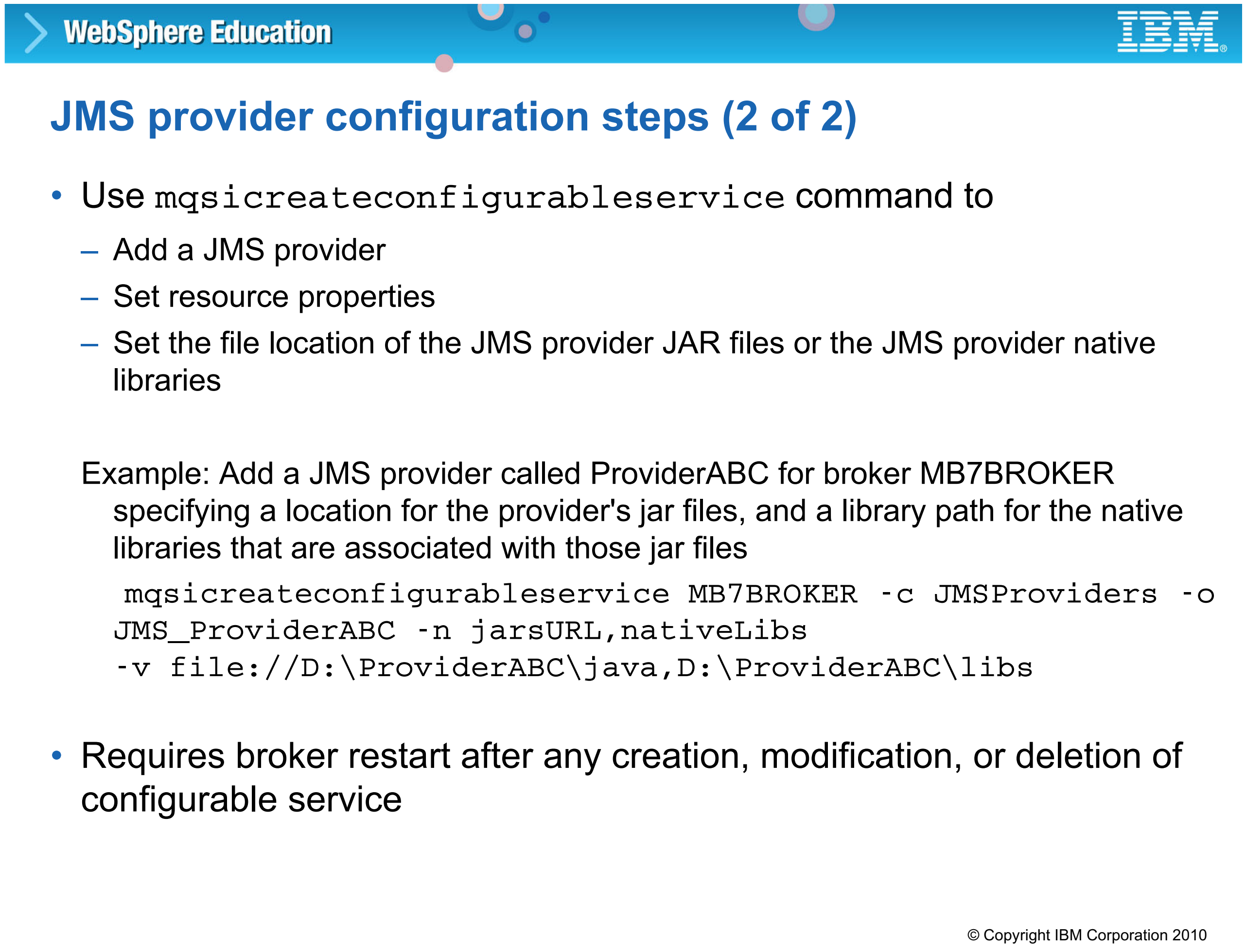
Instructor notes:

Purpose — Introduce the configuration tasks.

Details — None.

Additional information — None.

Transition statement — There are other commands that you can use to configure a JMS provider as well.



WebSphere Education

JMS provider configuration steps (2 of 2)

- Use `mqsicreateconfigurableservice` command to
  - Add a JMS provider
  - Set resource properties
  - Set the file location of the JMS provider JAR files or the JMS provider native libraries

Example: Add a JMS provider called ProviderABC for broker MB7BROKER specifying a location for the provider's jar files, and a library path for the native libraries that are associated with those jar files

```
 mqsicreateconfigurableservice MB7BROKER -c JMSProviders -o
JMS_ProviderABC -n jarsURL,nativeLibs
-v file://D:\ProviderABC\java,D:\ProviderABC\libs
```

- Requires broker restart after any creation, modification, or deletion of configurable service

Figure 9-46. JMS provider configuration steps (2 of 2) WM643 / VM6431.0

Notes:

The `mqsicreateconfigurableservice` command can be used to add a JMS provider and configure its properties, and set the location for the associated library files.

Related commands can be used to report, modify, or delete configurable services:

- Use `mqsichangeproperties` command to modify configurable services.
- Use `mqsideleteconfigurableservice` command to delete configurable services.
- Use `mqsireportproperties` command to view configurable services.

These commands can also be used to configure JDBC properties and FTP services

You must also copy the Java `.jar` files and any native libraries for the JMS provider client into the broker shared-classes directory. For example, on Windows, this is: `C:\Documents and Settings\All Users\Application Data\IBM\MQSI\shared-classes`

This ensures that the Java class path for the JMS nodes is set correctly.

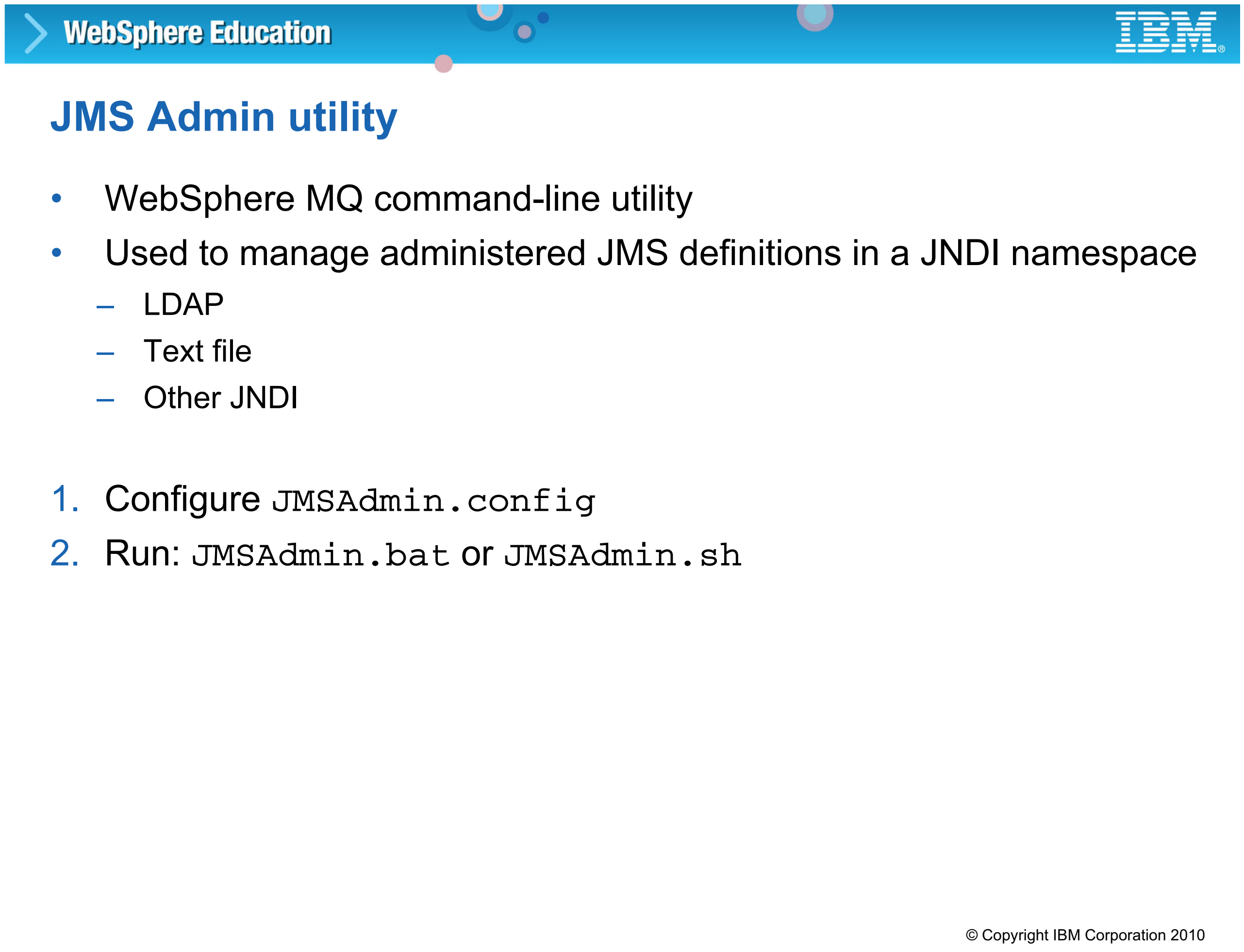
Instructor notes:

Purpose — Use a command line to create the configurable service.

Details — The `mqsiscreateconfigurableservice` and related commands were added in Message Broker V6.1

Additional information — None.

Transition statement — The JMS Admin utility you learned about in the WebSphere MQ System Administration course can also be used to configure JMS for WebSphere Message Broker.



WebSphere Education IBM

JMS Admin utility

- WebSphere MQ command-line utility
- Used to manage administered JMS definitions in a JNDI namespace
  - LDAP
  - Text file
  - Other JNDI

1. Configure `JMSAdmin.config`
2. Run: `JMSAdmin.bat` or `JMSAdmin.sh`

Figure 9-47. JMS Admin utility

Notes:

Use the WebSphere MQ JMSAdmin utility to manager administered JMS definitions in a JNDI namespace.

The `JMSAdmin.config` file must be configured before JMSAdmin can be invoked.

The JMSAdmin utility needs at least two settings:

- INITIAL_CONTEXT_FACTORY which is essentially how to access the JMDI repository
- PROVIDER_URL which indicates the location of the repository

When invoking the JMSAdmin utility, specify the particular configuration file using the `-cfg` parameter. If you do not use the `-cfg` parameter, the utility attempts to use the default configuration file (`JMSAdmin.config`), first looking for it in the current directory, and then in the `<MQ_JAVA_INSTALL_PATH>/bin` directory, where `<MQ_JAVA_INSTALL_PATH>` is the path to your WebSphere MQ JMS installation.

The configuration file is a plain-text file that consists of a set of key-value pairs, separated by an equal sign (=). The hash character (#) in the first column of the line indicates a comment, or a line that is not used.

Instructor notes:

Purpose — Describe the use of the JMSAdmin utility.

Details — The JMSAdmin utility is provided with WebSphere MQ as a tool to manage administered JMS definitions in a connected JNDI namespace. Routines are supplied with JMSAdmin to connect to simple JNDI repositories like LDAP and text files, but it can also be used to connect to other JNDI types, but the connection interface methods must be sourced from the suppliers of those JNDI repositories. Usually, JMS resources are defined using the administration facilities of the application server software, such as WebSphere Application Server administration console. But for stand-alone JMS clients, JMSAdmin provides this interface.

The `JMSAdmin.config` file must be configured to point to the JNDI and indicate the type of JNDI. If LDAP is used, connection details like security authentication can be specified.

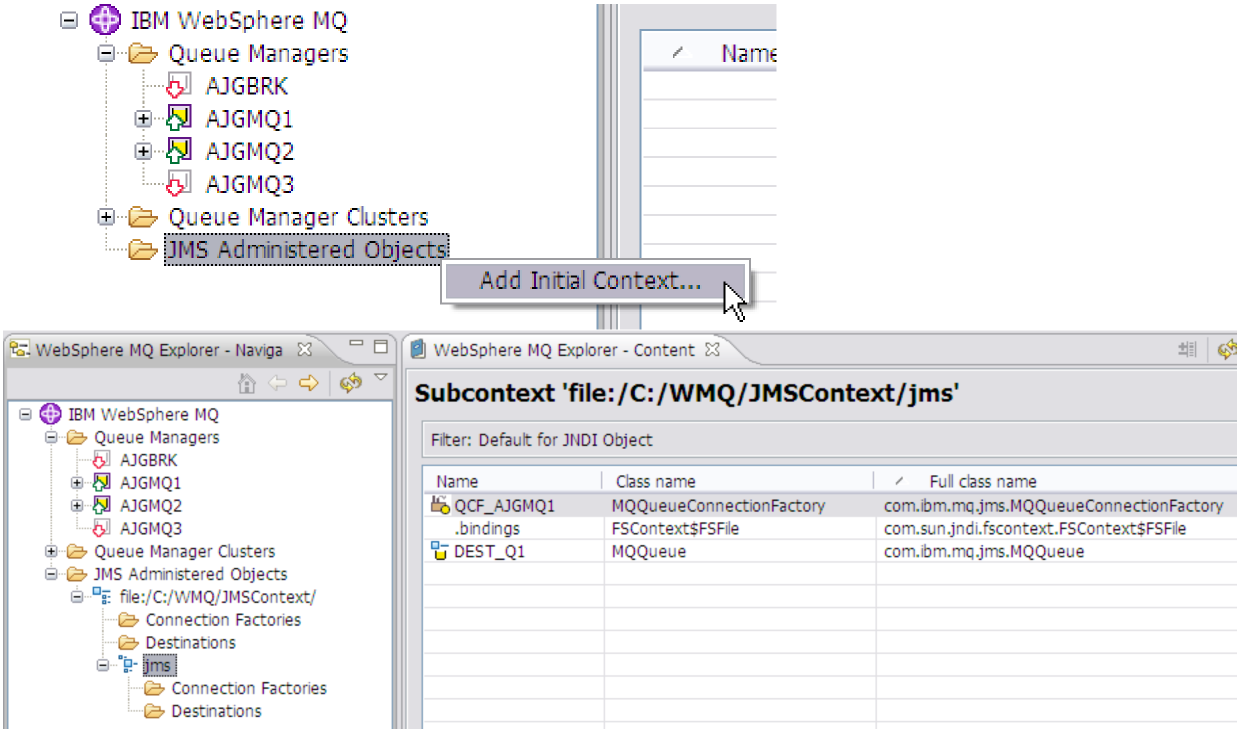
To run the JMSAdmin after the configuration file is complete, run `JMSAdmin.bat` or `JMSAdmin.sh` depending on your platform.

Additional information — This should be review for all students.

Transition statement — WebSphere MQ Explorer can also be used to manage JMS administered objects.

WebSphere Education

WebSphere MQ Explorer facilities for JMS



Subcontext 'file:/C:/WMQ/JMSContext/jms'

Filter: Default for JNDI Object

Name	Class name	Full class name
QCF_AJGMQ1	MQQueueConnectionFactory	com.ibm.mq.jms.MQQueueConnectionFactory
.bindings	FSContext$FSFile	com.sun.jndi.fscontext.FSContext$FSFile
DEST_Q1	MQQueue	com.ibm.mq.jms.MQQueue

Figure 9-48. WebSphere MQ Explorer for JMS WM643 / VM6431.0

Notes:

WebSphere MQ Explorer also contains facilities for creating and managing JMS administered objects.

The same way JMSAdmin must be configured, the WebSphere MQ Explorer first must be told the type and location of the JNDI repository.

After connected, the WebSphere MQ Explorer may be used to perform most JMS administration tasks.

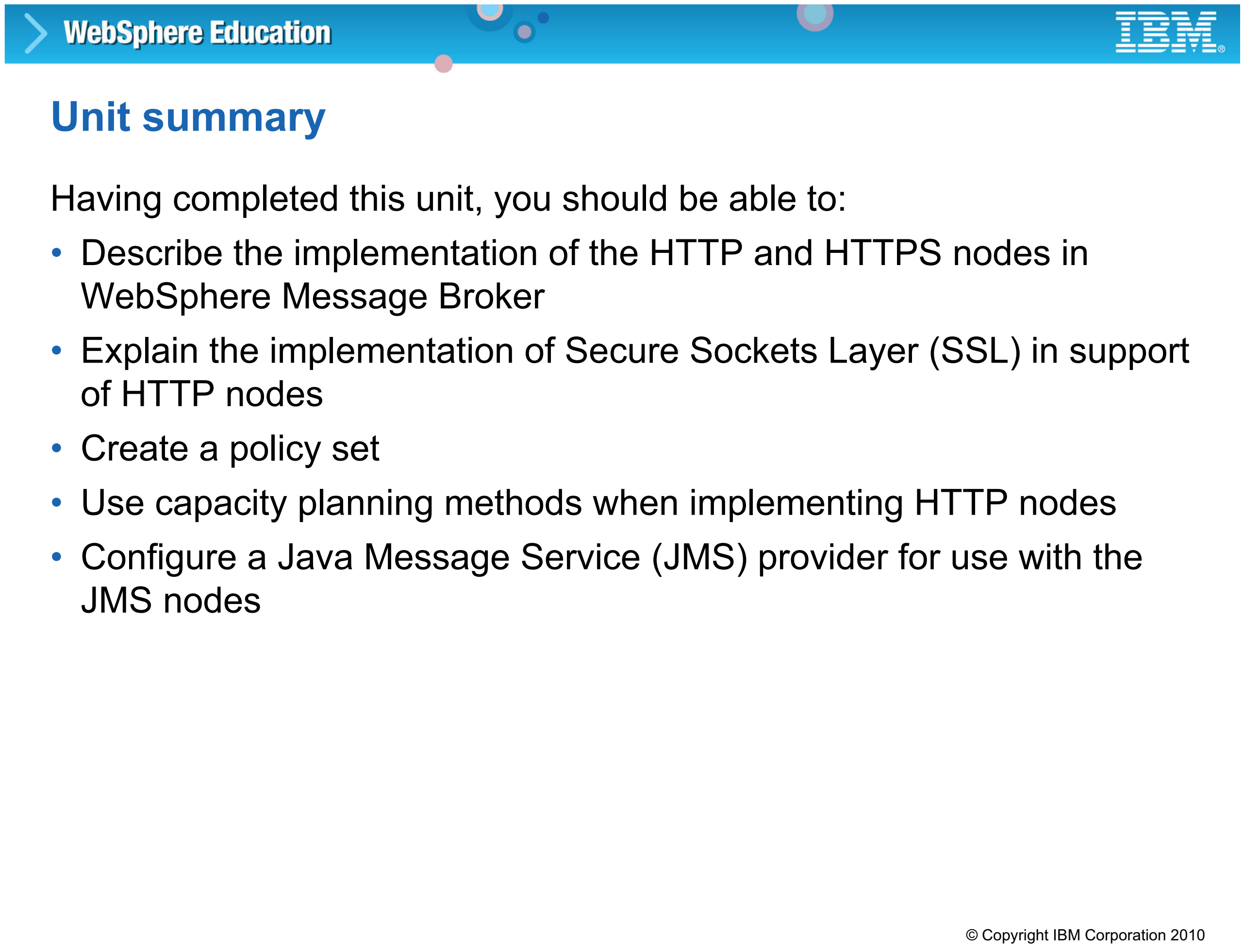
Instructor notes:

Purpose — Describe the JMS facilities in WebSphere MQ Explorer.

Details — The WebSphere MQ Explorer can also be used to manage the JNDI repository. First the initial context must be added, as shown in the top screen capture. The subsequent dialog requests the same configuration information as previously described for JMSAdmin configuration. After this is done, the WebSphere MQ Explorer can connect to the JNDI repository.

Additional information — This should be review for all students if they have attended the prerequisite course.

Transition statement — Now to summarize.



WebSphere Education

Unit summary

Having completed this unit, you should be able to:

- Describe the implementation of the HTTP and HTTPS nodes in WebSphere Message Broker
- Explain the implementation of Secure Sockets Layer (SSL) in support of HTTP nodes
- Create a policy set
- Use capacity planning methods when implementing HTTP nodes
- Configure a Java Message Service (JMS) provider for use with the JMS nodes

Figure 9-49. Unit summary

WM643 / VM6431.0

Notes:

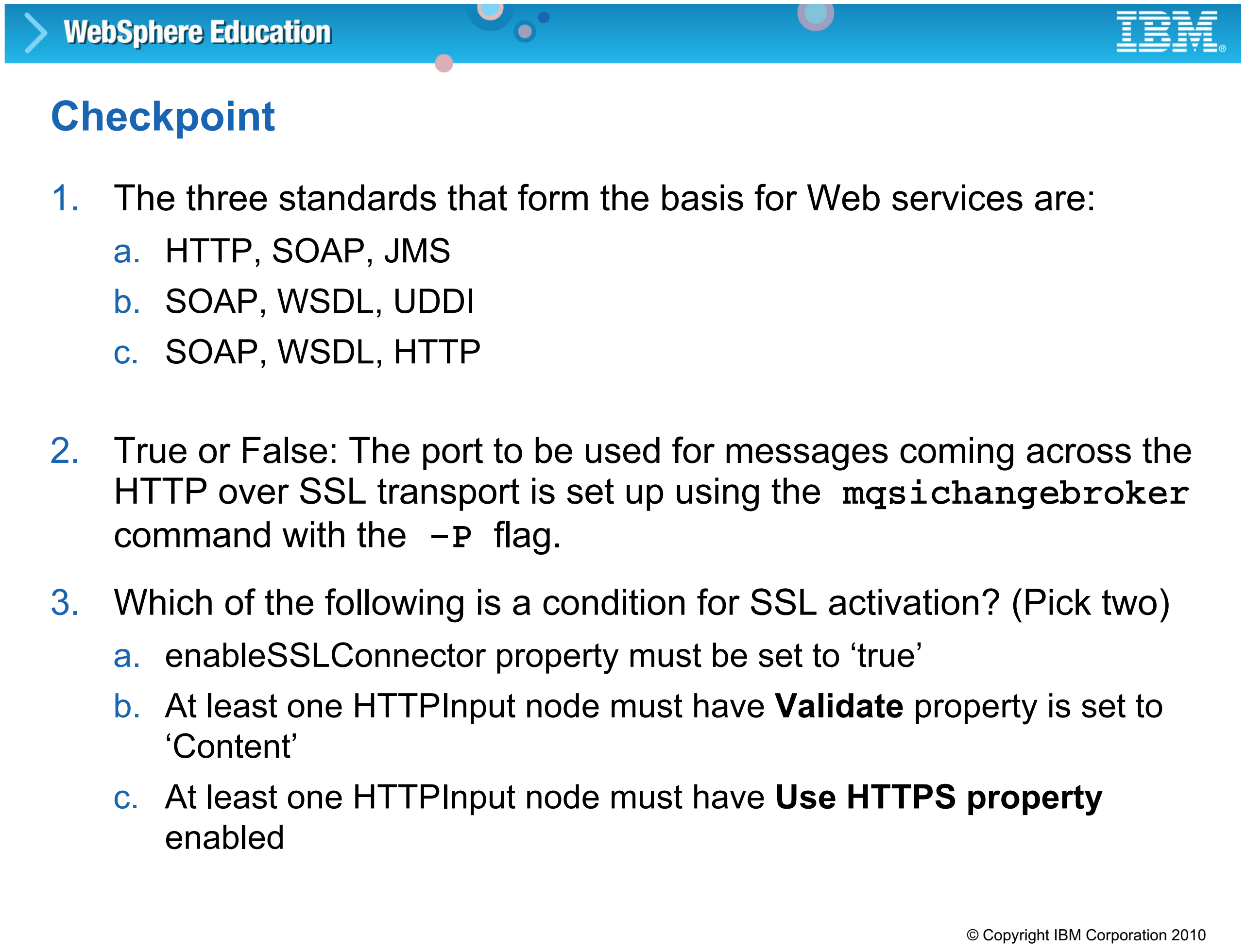
Instructor notes:

Purpose — Close the unit with a summary of highlights.

Details — None.

Additional information — None.

Transition statement — Now you will try Exercise 9, which shows you how to call a web service and how the broker can host a (local) web service. In the next unit, you will talk about user extensions to increase the functionality of the WebSphere Message Brokers beyond what is delivered with the product.



WebSphere Education

Checkpoint

1. The three standards that form the basis for Web services are:
   a. HTTP, SOAP, JMS
   b. SOAP, WSDL, UDDI
   c. SOAP, WSDL, HTTP

2. True or False: The port to be used for messages coming across the HTTP over SSL transport is set up using the `mqsichangebroker` command with the `-P` flag.

3. Which of the following is a condition for SSL activation? (Pick two)
   a. enableSSLConnector property must be set to ‘true’
   b. At least one HTTPInput node must have **Validate** property is set to ‘Content’
   c. At least one HTTPInput node must have **Use HTTPS property** enabled

Figure 9-50. Checkpoint

WM643 / VM6431.0

Notes:

Write your answers here:

1.

2.

3.

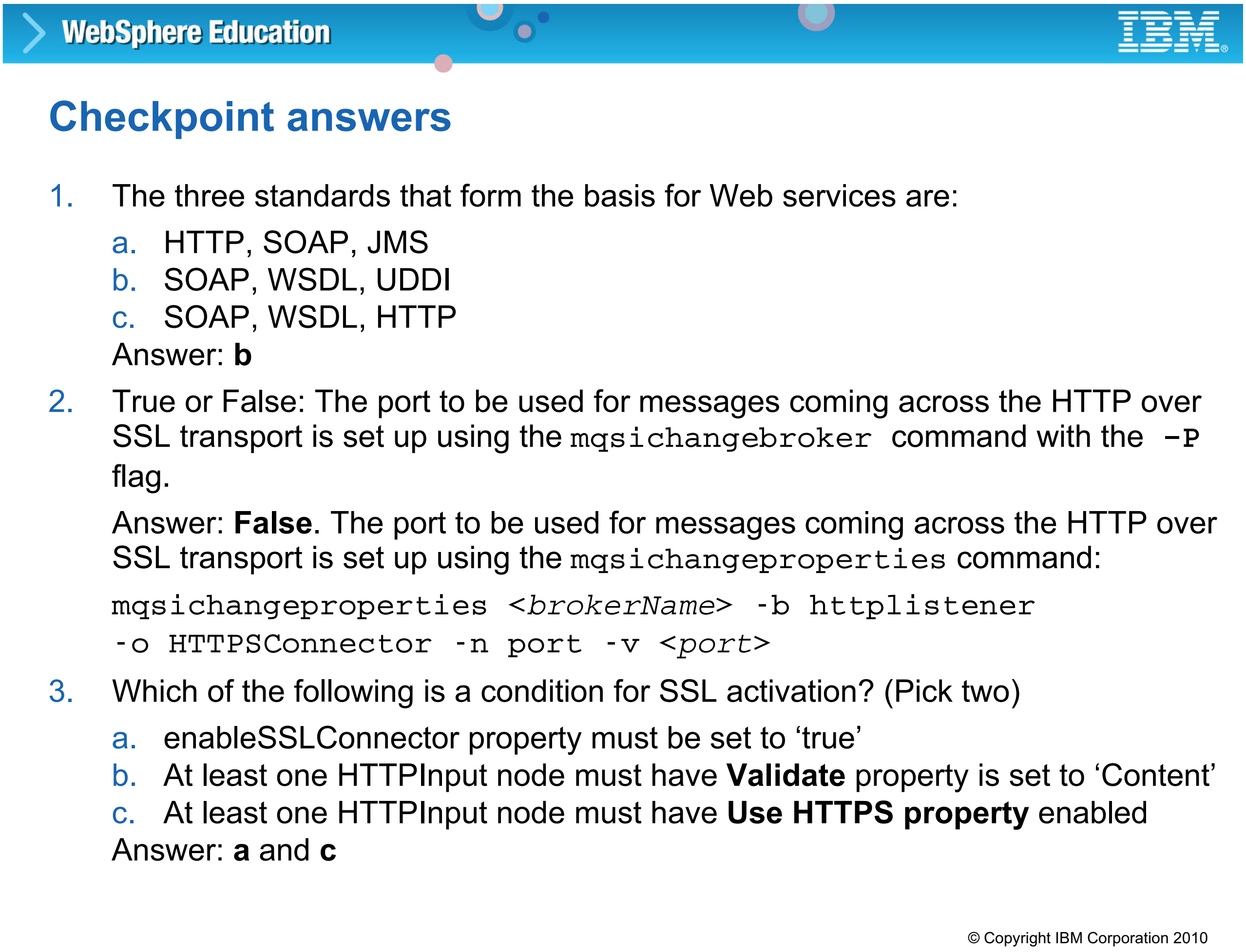
Instructor notes:

Purpose — Review the unit with some questions.

Details —

Additional information —

Transition statement —



WebSphere Education

Checkpoint answers

1. The three standards that form the basis for Web services are:
   - a. HTTP, SOAP, JMS
   - b. SOAP, WSDL, UDDI
   - c. SOAP, WSDL, HTTP

   Answer: **b**

2. True or False: The port to be used for messages coming across the HTTP over SSL transport is set up using the `mqsichangebroker` command with the **`-P`** flag.

   Answer: **False**. The port to be used for messages coming across the HTTP over SSL transport is set up using the `mqsichangeproperties` command:

   ```
   mqsichangeproperties <brokerName> -b httplistener
   -o HTTPSConnector -n port -v <port>
   ```

3. Which of the following is a condition for SSL activation? (Pick two)
   - a. enableSSLConnector property must be set to ‘true’
   - b. At least one HTTPInput node must have **Validate** property is set to ‘Content’
   - c. At least one HTTPInput node must have **Use HTTPS property** enabled

   Answer: **a** and **c**

© Copyright IBM Corporation 2010

Figure 9-51. Checkpoint answers

WM643 / VM6431.0

Notes:

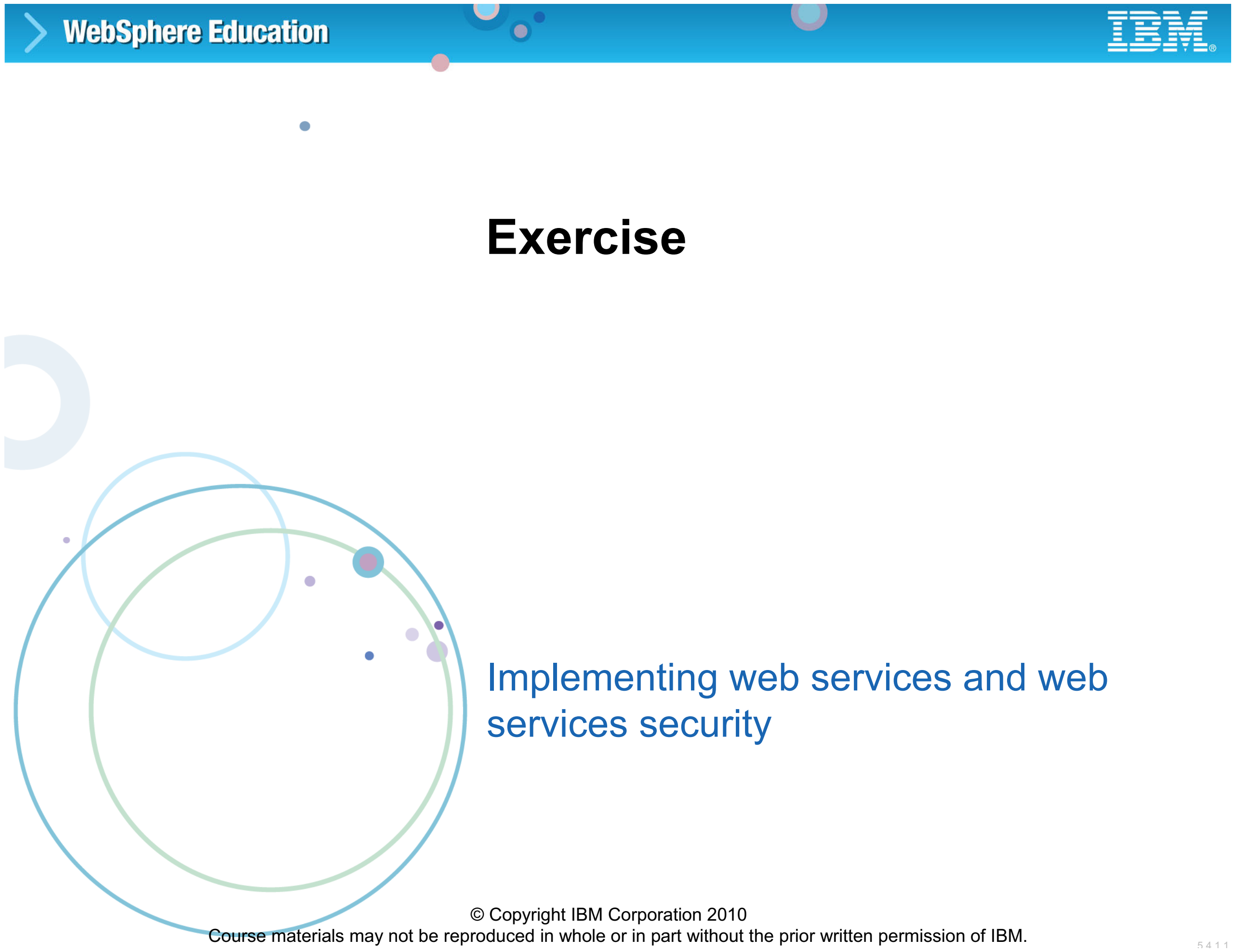
Instructor notes:

Purpose — Discuss the checkpoint answers (right and wrong) with students.

Details —

Additional information —

Transition statement — Next is an exercise.



WebSphere Education

Exercise

Implementing web services and web services security

Figure 9-52. Exercise 8 WM643 / VM6431.0

Notes:

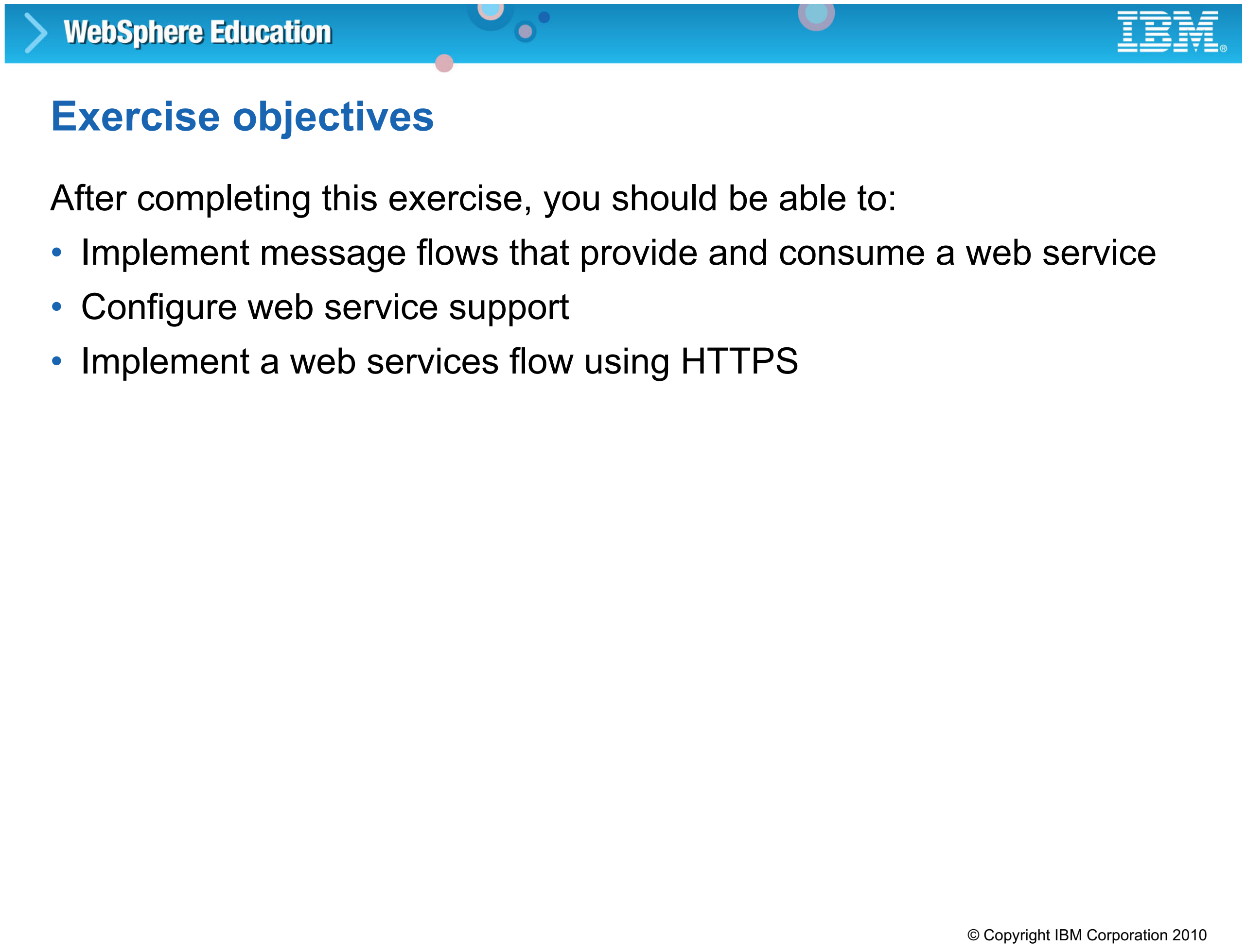
Instructor notes:

Purpose — Reinforce student learning by completing a hands-on exercise.

Details — If you do not have anyone in this class interested in web services you can omit this exercise.

Additional information — None.

Transition statement —



WebSphere Education

Exercise objectives

After completing this exercise, you should be able to:

- Implement message flows that provide and consume a web service
- Configure web service support
- Implement a web services flow using HTTPS

Figure 9-53. Exercise objectives WM643 / VM6431.0

Notes:

Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —

Unit 10. Extending WebSphere Message Broker

Estimated time

01:00 lecture, 00:30 exercise

What this unit is about

This unit describes administration tasks required when extending WebSphere Message Broker to connect to a database or read and write files using secure FTP (SFTP).

It also provides an overview of how to implement a message flow processing node or custom parser to provide additional function to the WebSphere Message Brokers.

What you should be able to do

After completing this unit, you should be able to:

- Connect to a database with ODBC and JDBC
- Configure WebSphere Message Broker for the secure file transfer protocol (SFTP)
- Implement user-defined extensions
- Find and install SupportPac components

How you will check your progress

Accountability:

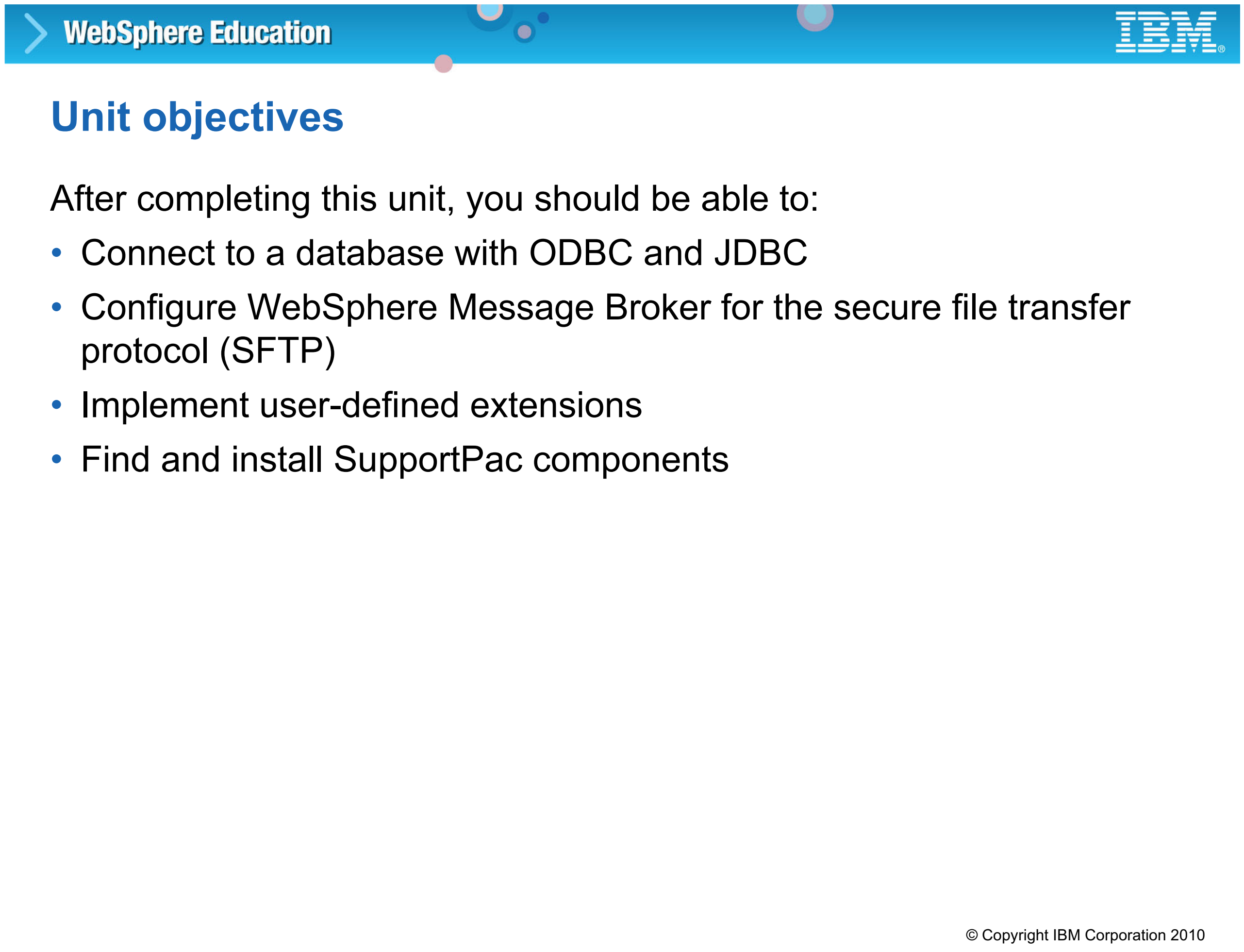
- Checkpoint questions
- Exercise 9, Implementing a user-defined extension

References

`www.ibm.com/software/integration/wbimessagebroker/library/`
WebSphere Message Broker stand-alone online help system

`www.ibm.com/software/integration/support/supportpacs`
WebSphere MQ SupportPacs

`www.ibm.com/software/integration/wmq/partners`
IBM WebSphere MQ Partners



WebSphere Education

Unit objectives

After completing this unit, you should be able to:

- Connect to a database with ODBC and JDBC
- Configure WebSphere Message Broker for the secure file transfer protocol (SFTP)
- Implement user-defined extensions
- Find and install SupportPac components

Figure 10-1. Unit objectives

Notes:

This unit is intended to introduce the concept of extending the WebSphere Message Broker to connect to a database, use SFTP for file transfer, and implementing user-defined extensions. The focus of the topics is on the administration aspects of implementing the extensions.

Instructor notes:

Purpose — Introduce the unit.

Details — None.

Additional information — None.

Transition statement — In many cases, a message flow might need to reference a database for cross-reference information for example.

10.1.Database connectivity

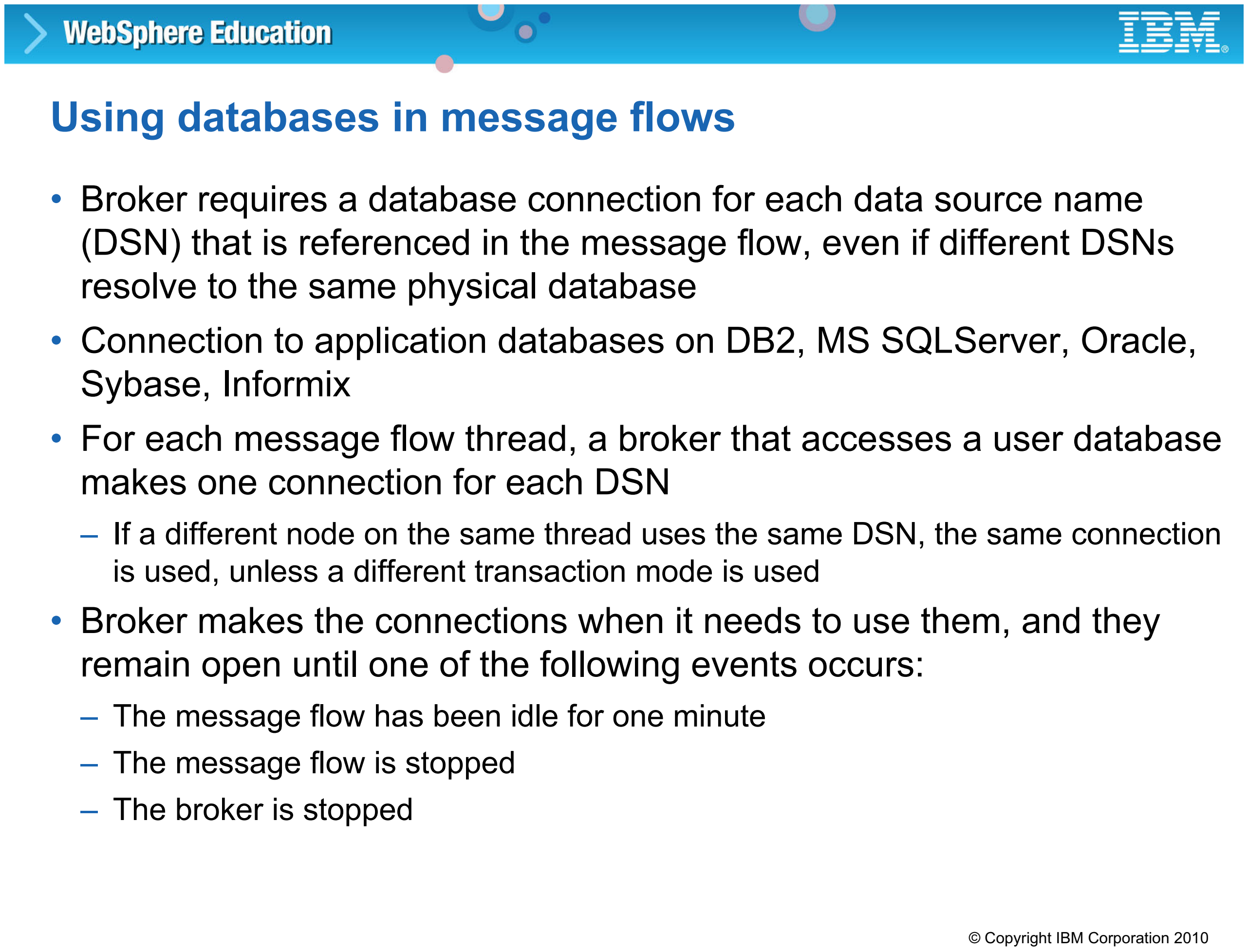
Instructor topic introduction

What students will do — Learn how to administer database connections to WebSphere Message Broker.

How students will do it — By listening to lecture.

What students will learn — The requirements for creating a connection and how to verify a connection.

How this will help students on their job — Administrators will be able to set up database connections and be able to detect problems.



WebSphere Education

Using databases in message flows

- Broker requires a database connection for each data source name (DSN) that is referenced in the message flow, even if different DSNs resolve to the same physical database
- Connection to application databases on DB2, MS SQLServer, Oracle, Sybase, Informix
- For each message flow thread, a broker that accesses a user database makes one connection for each DSN
  - If a different node on the same thread uses the same DSN, the same connection is used, unless a different transaction mode is used
- Broker makes the connections when it needs to use them, and they remain open until one of the following events occurs:
  - The message flow has been idle for one minute
  - The message flow is stopped
  - The broker is stopped

Figure 10-2. Using databases in message flows WM643 / VM6431.0

Notes:

Developers can create and configure message flows to access databases for additional information to enhance or influence the operation of the message flow. A message flow can also modify the contents of a database by adding new information or removing or replacing existing information.

Accessing user databases in a message flow requires the database definitions in the workspace. The WebSphere Message Broker Toolkit can connect to the database, discover the definition, and then store it in the workspace. The administrator must define a DSN for each database and also configure any security requirements.

The broker automatically handles connection pooling and manages the message flow as a single transaction. When a message flow is idle for approximately 1 minute, or if the message flow completes, the broker closes the connection.

Always check the database documentation for information about connections and the limits or restrictions that might apply.

Instructor notes:

Purpose — Introduce user database access.

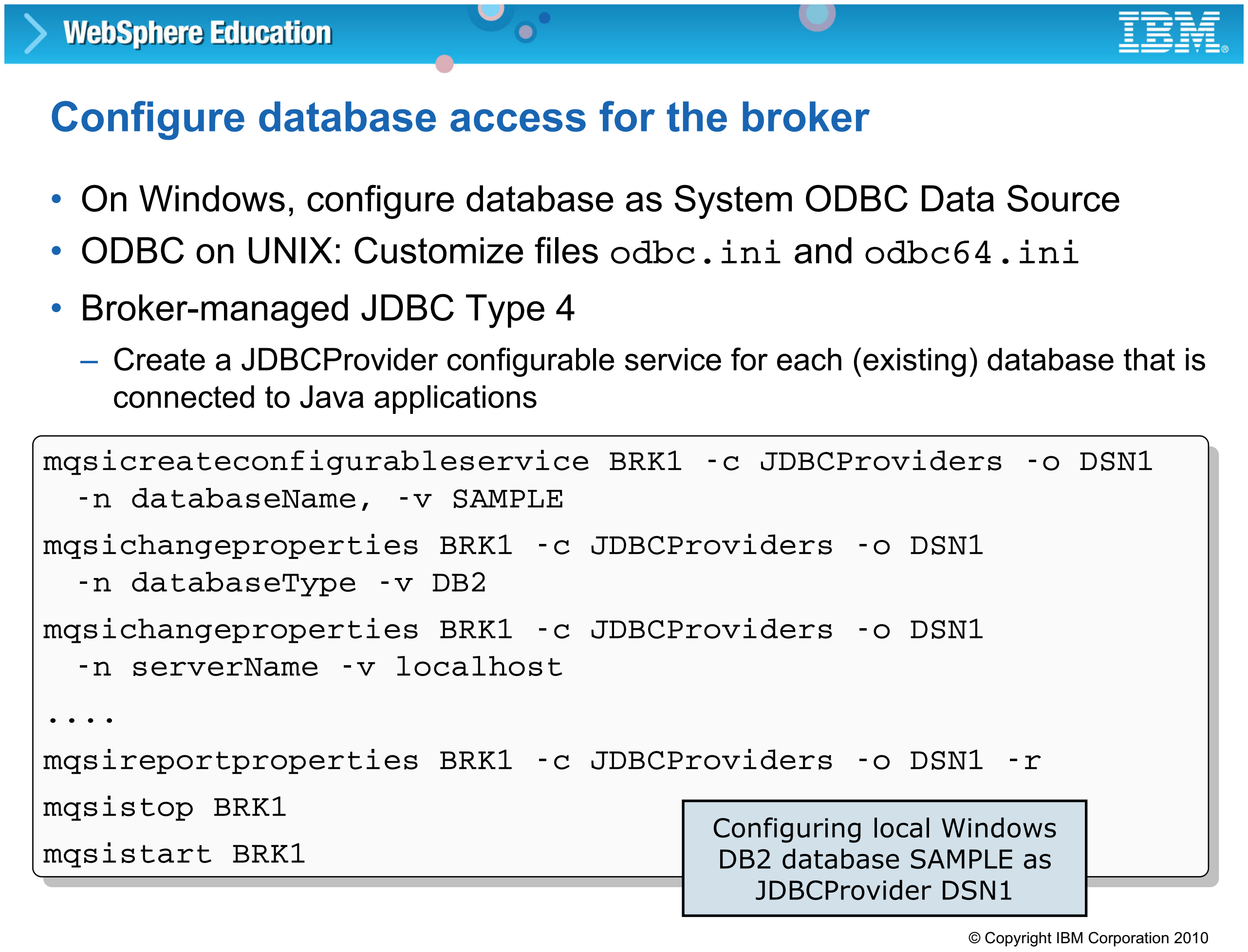
Details — When you start a broker, and while it is running, it opens connections to WebSphere MQ queues and to databases. The broker makes the connections when it needs to use them, and they remain open until a flow has no work. For example, a connection is released if there are no messages in the input queue on a message flow, and the database has not been accessed for 1 minute.

Determine the number of database connections required by a broker for capacity and resource-planning purposes. A database connection is made by the broker to the ODBC data source name (DSN). A connection is made for each DSN even if different DSNs resolve to the same physical database.

The default action taken by DB2 is to limit the number of concurrent connections to a database to the value of the **maxappls** configuration parameter. The default for `maxappls` is 40. If you believe the connections that the broker might require exceed the value for maxappls, you must increase this to a new value based on your calculations.

Additional information — None

Transition statement — What needs to be configured so that the broker can access user databases?



WebSphere Education

Configure database access for the broker

- On Windows, configure database as System ODBC Data Source
- ODBC on UNIX: Customize files `odbc.ini` and `odbc64.ini`
- Broker-managed JDBC Type 4
  - Create a JDBCProvider configurable service for each (existing) database that is connected to Java applications

```
mqsicreateconfigurableservice BRK1 -c JDBCProviders -o DSN1
  -n databaseName, -v SAMPLE
mqsichangeproperties BRK1 -c JDBCProviders -o DSN1
  -n databaseType -v DB2
mqsichangeproperties BRK1 -c JDBCProviders -o DSN1
  -n serverName -v localhost
....
mqsireportproperties BRK1 -c JDBCProviders -o DSN1 -r
mqsistop BRK1
mqsistart BRK1
```

Configuring local Windows DB2 database SAMPLE as JDBCProvider DSN1

Figure 10-3. Configure database access for the broker WM643 / VM6431.0

Notes:

WebSphere Message Broker provides different connection types to application databases, depending on the nodes. Use ODBC and broker-managed JDBC type 4 connections when possible to save on manual programming.

When the message flow includes a database, JavaCompute, or Java user-defined node in a message flow, and interacts with a database in that node, the broker must establish connections with the database to fulfill the operations that are performed by the node. Define a JDBCProvider configurable service to provide the broker with the information that it needs to complete the connection.

The example on the figure shows how to configure a local Windows DB2 database SAMPLE as JDBCProvider DSN1.

You might need to change more properties specific to your database and environment. Some values depend on how and where you have installed the database product; for example, the property `jarsURL` identifies the location of the JAR files that are supplied and installed by the database provider.

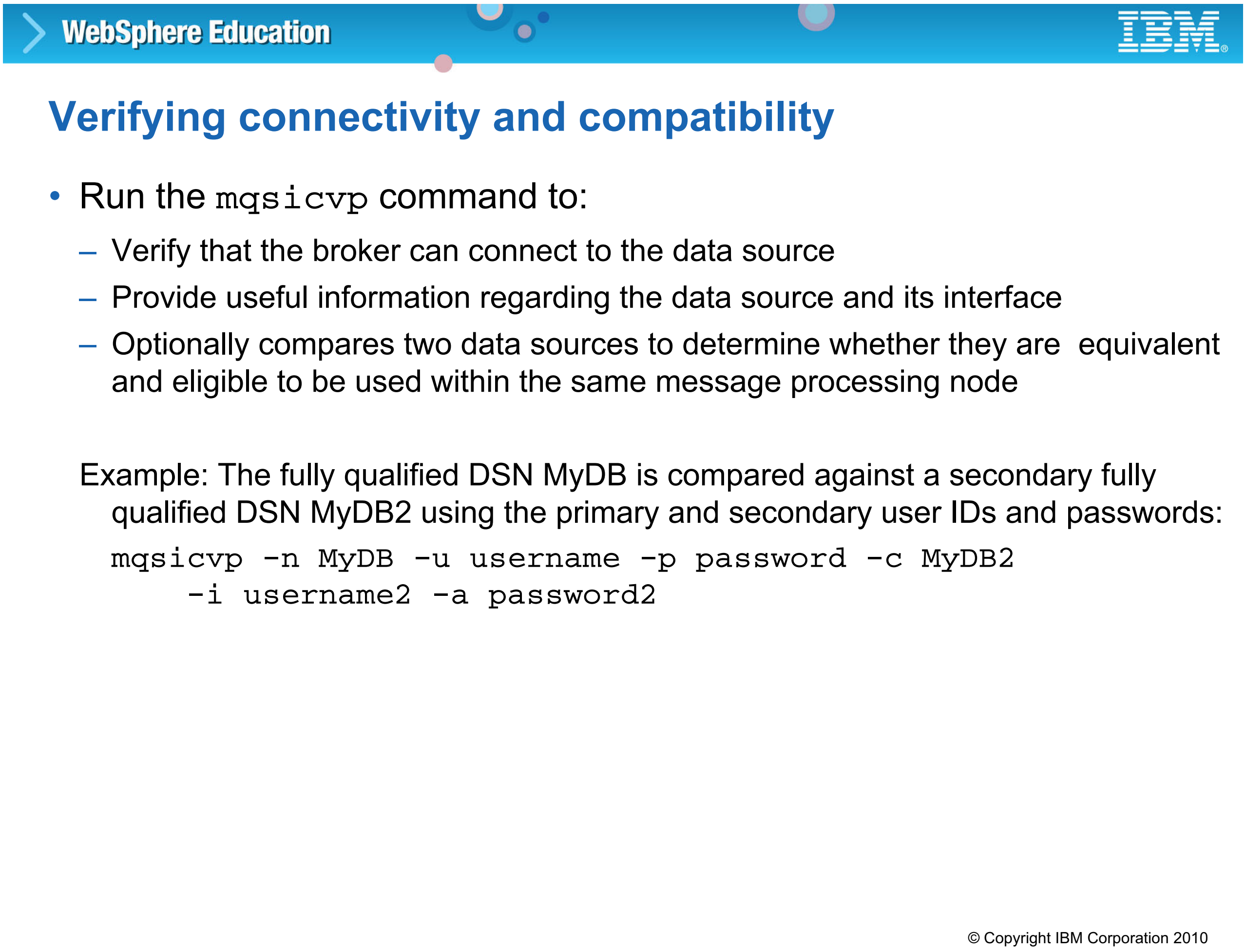
Instructor notes:

Purpose — Explain how to configure ODBC or JDBC for the broker. Do not spend much time on this topic.

Details — The `mqsireportproperties` command is used to examine the values of properties that are set using the `mqsichangeproperties` command, or created using the `mqsicreateconfigurableservice` command

Additional information — None.

Transition statement — The developer can check connectivity in the WebSphere Message Broker Toolkit. As an administrator, you can use a command.



WebSphere Education

Verifying connectivity and compatibility

- Run the `mqsicvp` command to:
  - Verify that the broker can connect to the data source
  - Provide useful information regarding the data source and its interface
  - Optionally compares two data sources to determine whether they are equivalent and eligible to be used within the same message processing node

Example: The fully qualified DSN MyDB is compared against a secondary fully qualified DSN MyDB2 using the primary and secondary user IDs and passwords:

```
mqsicvp -n MyDB -u username -p password -c MyDB2
    -i username2 -a password2
```

Figure 10-4. Verifying connectivity and compatibility WM643 / VM6431.0

Notes:

The `mqsicvp` command has been enhanced to provide database information and act as an ODBC test tool.

When using the `mqsicvp` command as an ODBC test tool, the command issues an informational message on successful connection, providing the name of the data source, database type, and version. If a secondary data source is supplied, the `mqsicvp` command issues a second informational message on successful connection to that data source, with the same information concerning the secondary data source, and informing you that a comparison is to be performed.

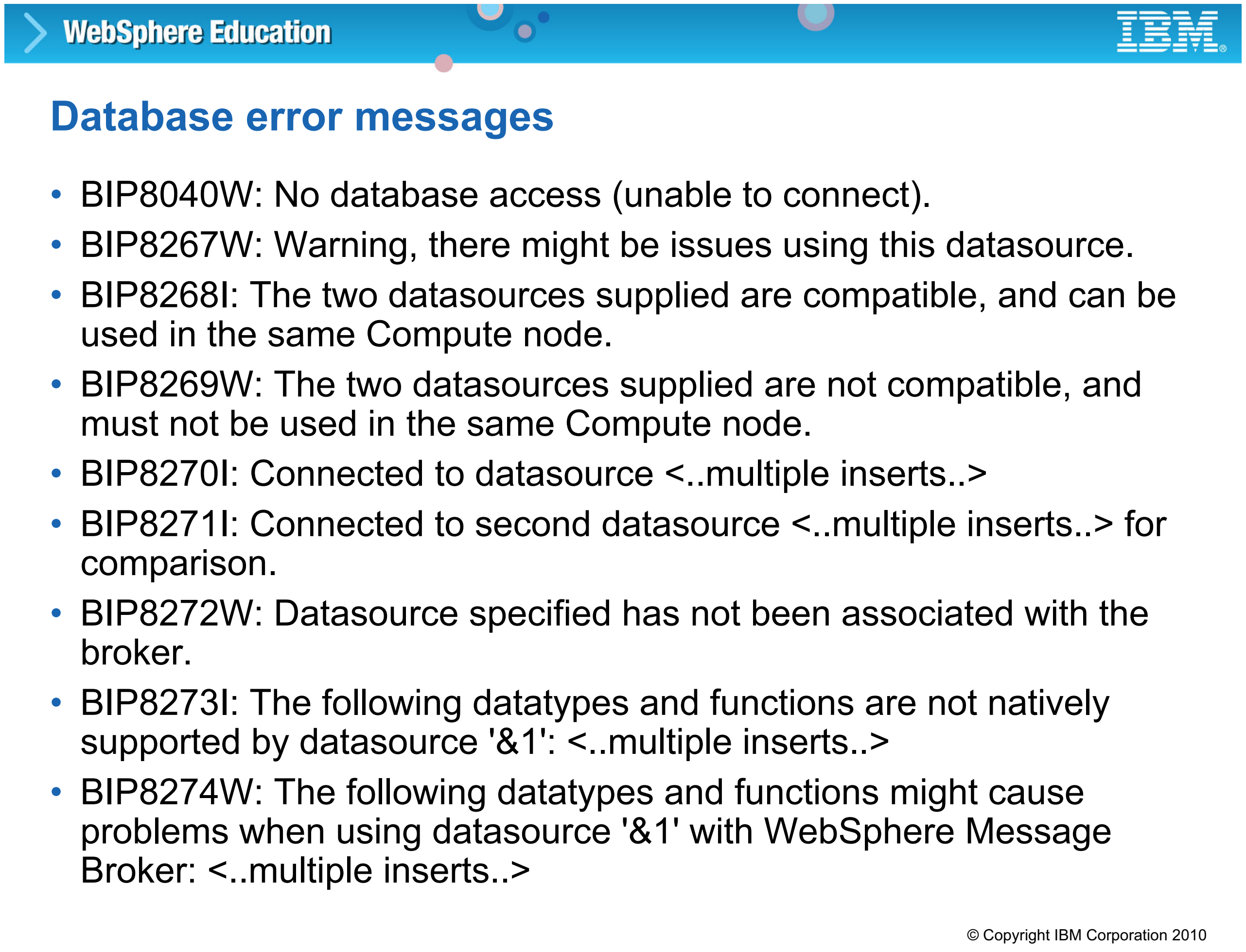
Instructor notes:

Purpose — Describe how to use the `mqsicvp` command to check database connectivity with the broker.

Details — In addition to verifying connectivity, the `mqsicvp` command can also determine whether there are any problems with the data types and whether the database is a suitable replacement for another database. This last feature is especially useful when moving from a development to pre-production environment, for example.

Additional information — None.

Transition statement — WebSphere Message Broker includes a number of specific error messages to help you quickly determine the cause of any problems.



WebSphere Education

Database error messages

- BIP8040W: No database access (unable to connect).
- BIP8267W: Warning, there might be issues using this datasource.
- BIP8268I: The two datasources supplied are compatible, and can be used in the same Compute node.
- BIP8269W: The two datasources supplied are not compatible, and must not be used in the same Compute node.
- BIP8270I: Connected to datasource <..multiple inserts..>
- BIP8271I: Connected to second datasource <..multiple inserts..> for comparison.
- BIP8272W: Datasource specified has not been associated with the broker.
- BIP8273I: The following datatypes and functions are not natively supported by datasource '&1': <..multiple inserts..>
- BIP8274W: The following datatypes and functions might cause problems when using datasource '&1' with WebSphere Message Broker: <..multiple inserts..>

Figure 10-5. Database error messages WM643 / VM6431.0

Notes:

The figure lists the database-specific error messages that might appear in the logs in the event of a database error or as a result of running the `mqsicvp` command.

Instructor notes:

Purpose — Review the types of messages the administrator might see for databases.

Details — Database messages have been improved to provide more information which should help administrators resolve problems.

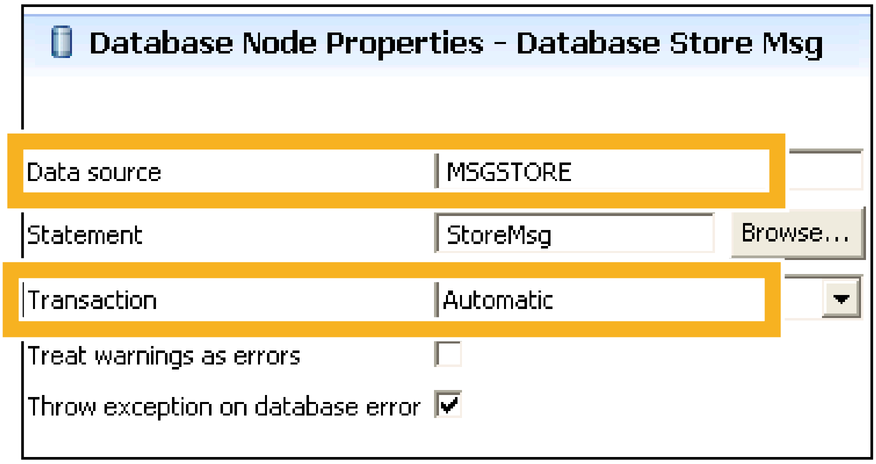
Additional information — None.

Transition statement — The developer selects the DSN in the node properties. As an administrator, you can override this property using the Broker Archive editor.

WebSphere Education

Node properties for database access

- Set **Data source** to use as default DSN (ODBC)
- When Transaction mode = Automatic, database updates are either committed or rolled back with entire message flow
- When Transaction mode = Commit:
  - Updates to default DSN immediately, even if message flow fails later
  - All noncommitted updates made in this message flow to default DSN



Database Node Properties - Database Store Msg

Data source: MSGSTORE

Statement: StoreMsg

Transaction: Automatic

□ Treat warnings as errors

☑ Throw exception on database error

Figure 10-6. Node properties for database access

Notes:

Every node that accesses databases must identify the default data source (DSN) in the node properties.

It there is a database mapping in the message flow, all ESQL nodes in the flow must specify a data source (it can be any data source, it only must exist).

The broker can coordinate the transaction and commit the database update after completion of the flow (Transaction mode = Automatic), or commit all updates on this database, even from upstream nodes, immediately (Transaction mode = Commit).

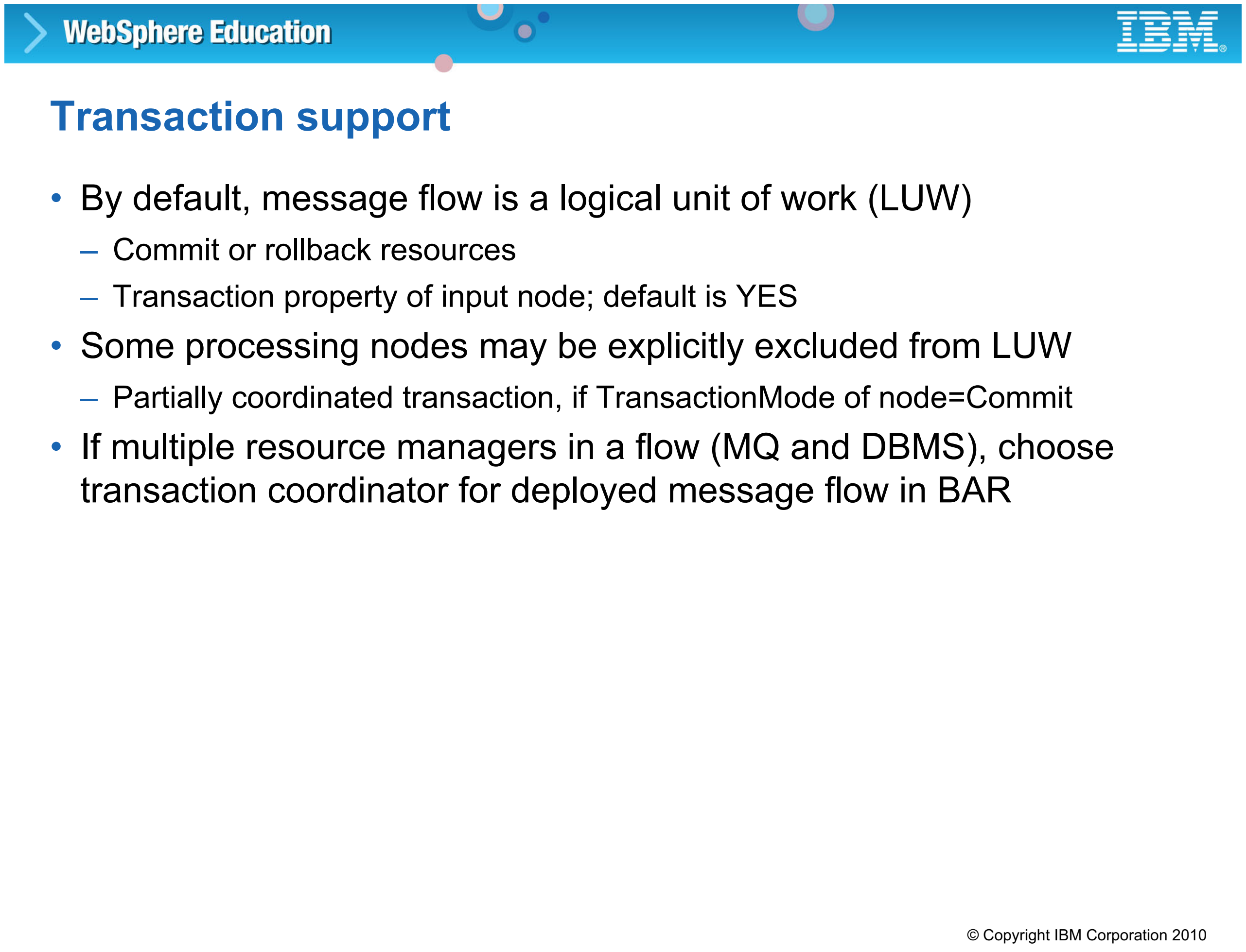
Instructor notes:

Purpose — Show an example of node properties and introduce the concept of transaction control for databases.

Details — None.

Additional information — None.

Transition statement — As an administrator, you might have to modify transaction settings.



WebSphere Education
IBM

Transaction support

- By default, message flow is a logical unit of work (LUW)
  - Commit or rollback resources
  - Transaction property of input node; default is YES
- Some processing nodes may be explicitly excluded from LUW
  - Partially coordinated transaction, if TransactionMode of node=Commit
- If multiple resource managers in a flow (MQ and DBMS), choose transaction coordinator for deployed message flow in BAR

Figure 10-7. Transaction support WM643 / VM6431.0

Notes:

An XA-coordinated transaction is NOT visible for the message flow developer. It is enabled in the BAR file when it is deployed in special scenarios (for example, banking) where data integrity is critical. XA coordination affects processing and resources.

Instructor notes:

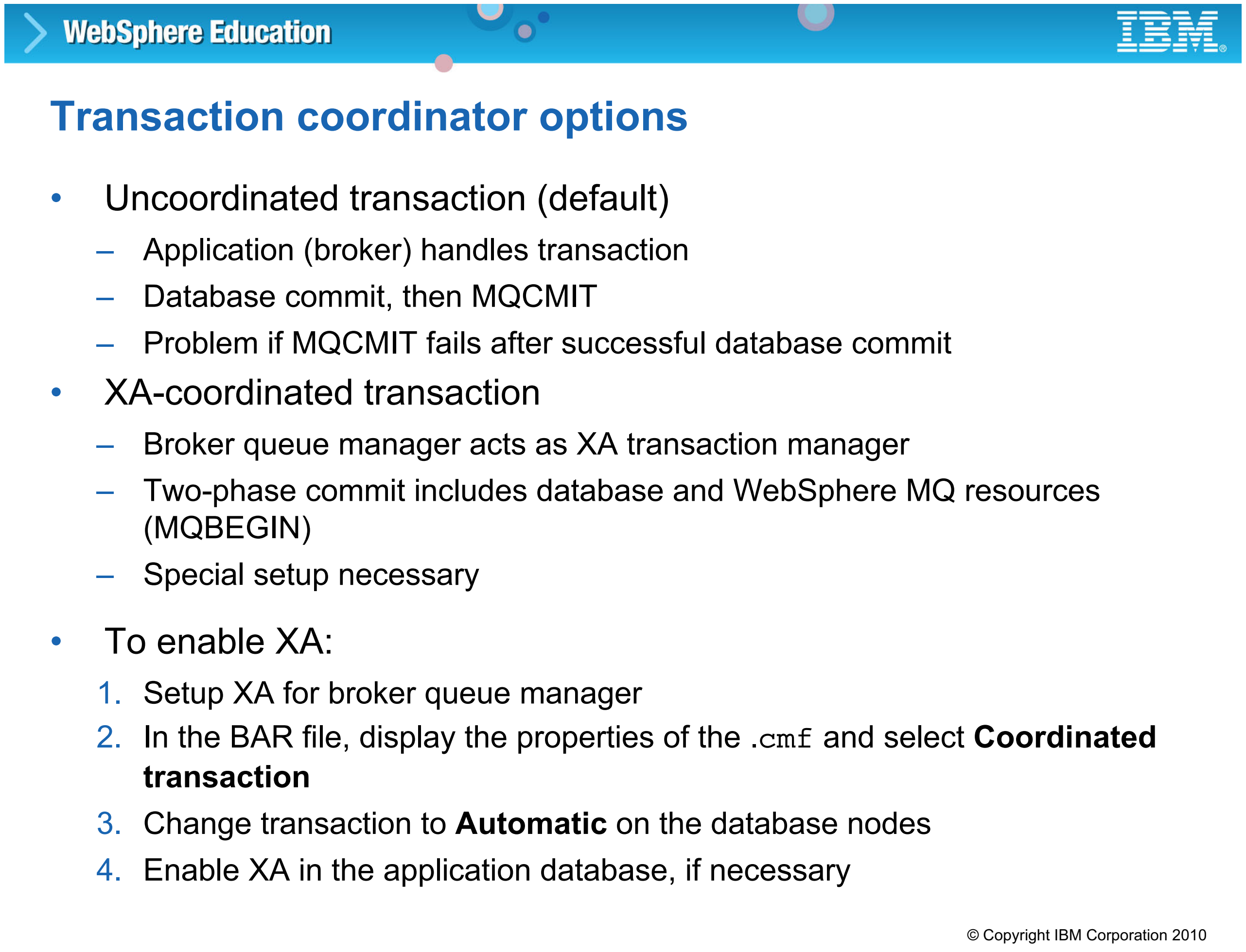
Purpose — Explain the difference between uncoordinated (default), partially coordinated, and coordinated (XA) transactions in WebSphere Message Broker.

Details — WebSphere MQ can be a transaction coordinator.

- It allows updates with external resource manager to be atomic.
- It is provided through XA connection between WebSphere MQ and the resource manager.
- With XA, the application issues one MQCMIT call to commit both WebSphere MQ and DB2 updates, for example.
- Without XA, the application must issue an MQCMIT call to commit WebSphere MQ processing and EXEC SQL SYNCPOINT to commit DB2 updates, for example.
- There is a potential for failure between commits, which can compromise data integrity.
- For maximum performance, ensure that both the queue manager and the resource manager log on low latency disks.

Additional information — None.

Transition statement — What are the transaction coordinator options?



WebSphere Education
IBM

Transaction coordinator options

- Uncoordinated transaction (default)
  - Application (broker) handles transaction
  - Database commit, then MQCMIT
  - Problem if MQCMIT fails after successful database commit
- XA-coordinated transaction
  - Broker queue manager acts as XA transaction manager
  - Two-phase commit includes database and WebSphere MQ resources (MQBEGIN)
  - Special setup necessary
- To enable XA:
  1. Setup XA for broker queue manager
  2. In the BAR file, display the properties of the `.cmf` and select **Coordinated transaction**
  3. Change transaction to **Automatic** on the database nodes
  4. Enable XA in the application database, if necessary

Figure 10-8. Transaction coordinator options WM643 / VM6431.0

Notes:

Uncoordinated flows are flows for which the **Coordinated** property is not set in the BAR file. Updates to resources used by an uncoordinated flow are managed by the separate resource managers. Some resource managers, such as WebSphere MQ, allow updates to be made nontransactionally, or as part of a resource-specific transaction. Other resource managers, such as database managers, always use a resource-specific transaction. A resource-specific transaction is a transaction whose scope is limited to the resources owned by a single resource manager, such as a database or queue manager.

To reduce the window of doubt when multiple separate resource managers are synchronized, transaction coordination using the XA protocol is provided on distributed platforms by WebSphere MQ and on z/OS systems by RRS.

Message flows are always globally coordinated on z/OS, regardless of whether the **Coordinated** property of the message flow is specified as coordinated or not.

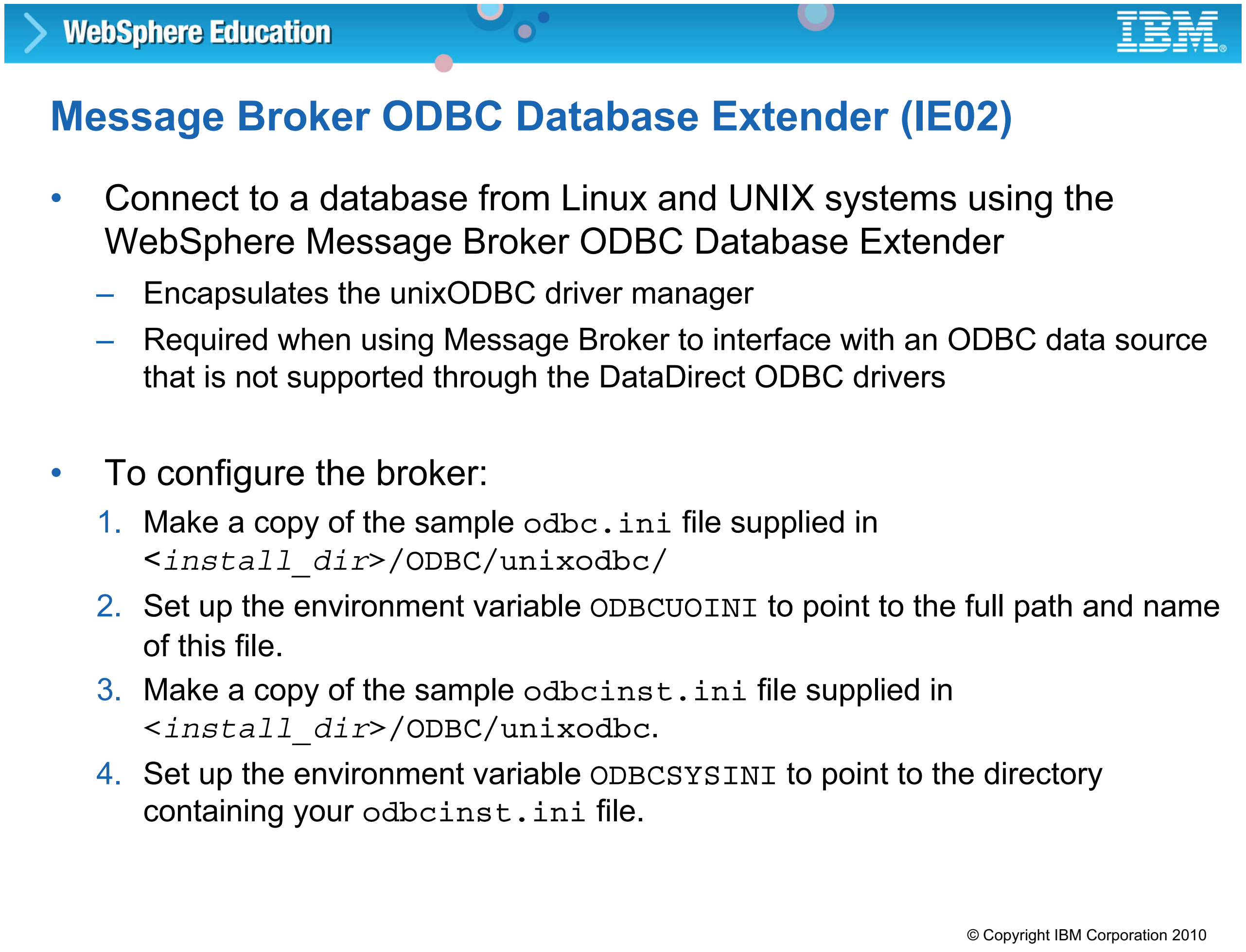
Instructor notes:

Purpose — Explain the difference between uncoordinated (default), partially coordinated, and coordinated (XA) transactions in WebSphere Message Broker.

Details — None.

Additional information — None.

Transition statement — A SupportPac is required when an ODBC data source is not supported using the DataDirect ODBC driver provided with WebSphere Message Broker.



WebSphere Education

Message Broker ODBC Database Extender (IE02)

- Connect to a database from Linux and UNIX systems using the WebSphere Message Broker ODBC Database Extender
  - Encapsulates the unixODBC driver manager
  - Required when using Message Broker to interface with an ODBC data source that is not supported through the DataDirect ODBC drivers

- To configure the broker:
  1. Make a copy of the sample `odbc.ini` file supplied in `<install_dir>/ODBC/unixodbc/`
  2. Set up the environment variable `ODBCUOINI` to point to the full path and name of this file.
  3. Make a copy of the sample `odbcinst.ini` file supplied in `<install_dir>/ODBC/unixodbc`.
  4. Set up the environment variable `ODBCSYSINI` to point to the directory containing your `odbcinst.ini` file.

Figure 10-9. WebSphere Message Broker ODBC Database Extender (IE02) WM643 / VM6431.0

Notes:

WebSphere Message Broker ODBC Database Extender (SupportPac IE02) extends the ability of the WebSphere Message Broker Product to allow additional database support on UNIX and Linux platforms.

The SupportPac includes installation instructions. User documentation is available in the WebSphere Message Broker online Information Center.

Instructor notes:

Purpose — Describe the SupportPac that can be used to extend database support.

Details — This is a new SupportPac. There is not a specific list of databases that are supported. The procedure should be to try to connect using the DataDirect ODBC driver. If that does not work, try this SupportPac.

Additional information — None.

Transition statement — Message flow containing FileInput and FileOutput nodes can be extended to use SFTP for secure file transfer.

10.2.Configuring for SFTP

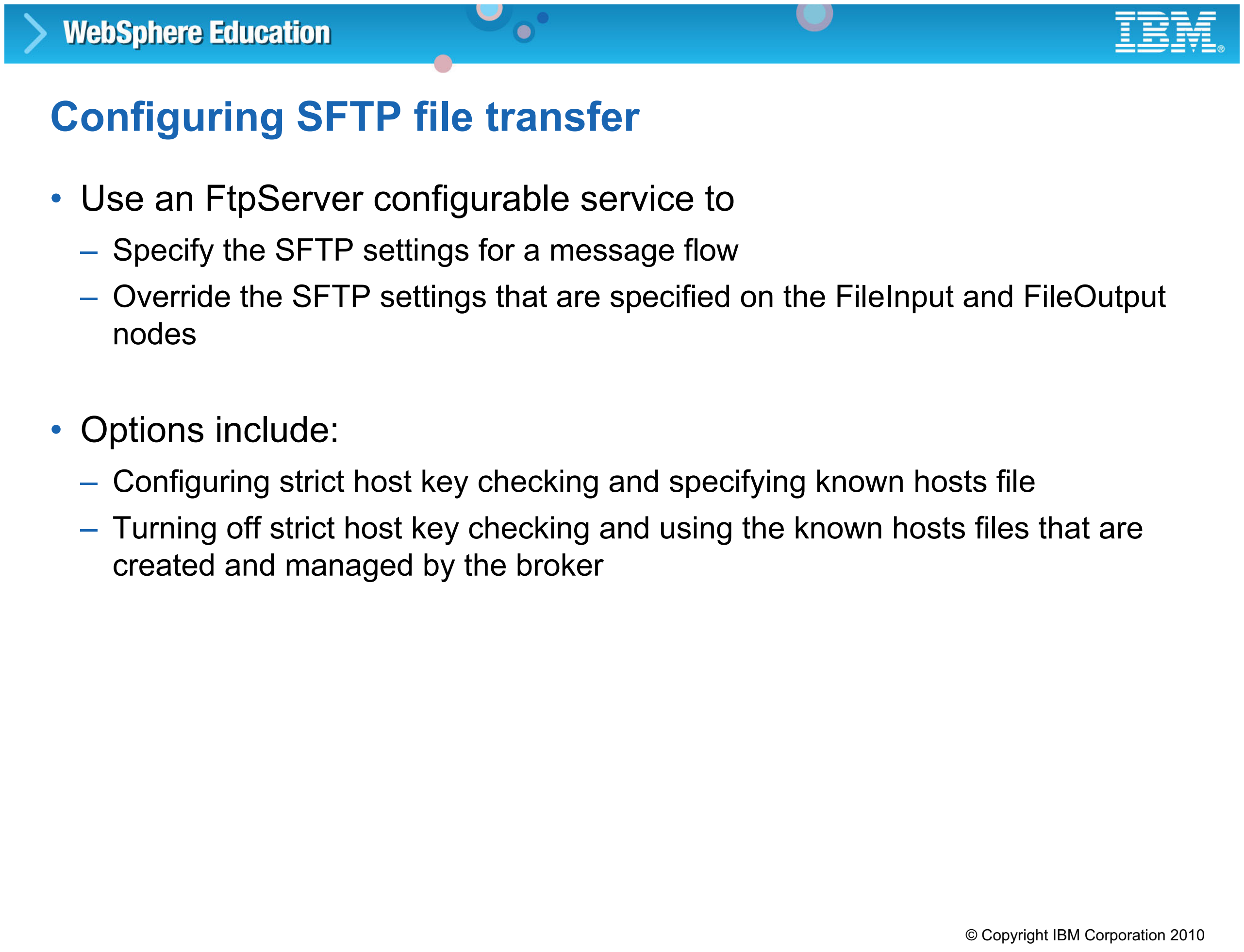
Instructor topic introduction

What students will do — Learn how to perform administration tasks so that message flows containing FileInput or FileOutput nodes can use SFTP for secure file transfers.

How students will do it — Listen to lecture.

What students will learn — How to create an FTPserver configurable service from a command line and from WebSphere Message Broker Explorer.

How this will help students on their job — Administrators will be able to deploy an FTPserver configurable service.



WebSphere Education

Configuring SFTP file transfer

- Use an FtpServer configurable service to
  - Specify the SFTP settings for a message flow
  - Override the SFTP settings that are specified on the FileInput and FileOutput nodes
- Options include:
  - Configuring strict host key checking and specifying known hosts file
  - Turning off strict host key checking and using the known hosts files that are created and managed by the broker

Figure 10-10. Configuring SFTP file transfer WM643 / VM6431.0

Notes:

The settings that you specify by using an FtpServer configurable service are read and validated when the message flow starts, and are used to configure any SFTP connections that are made for the node.

The configurable service can override any or all of the remote transfer properties on the FTP tab of the FileInput and FileOutput nodes.

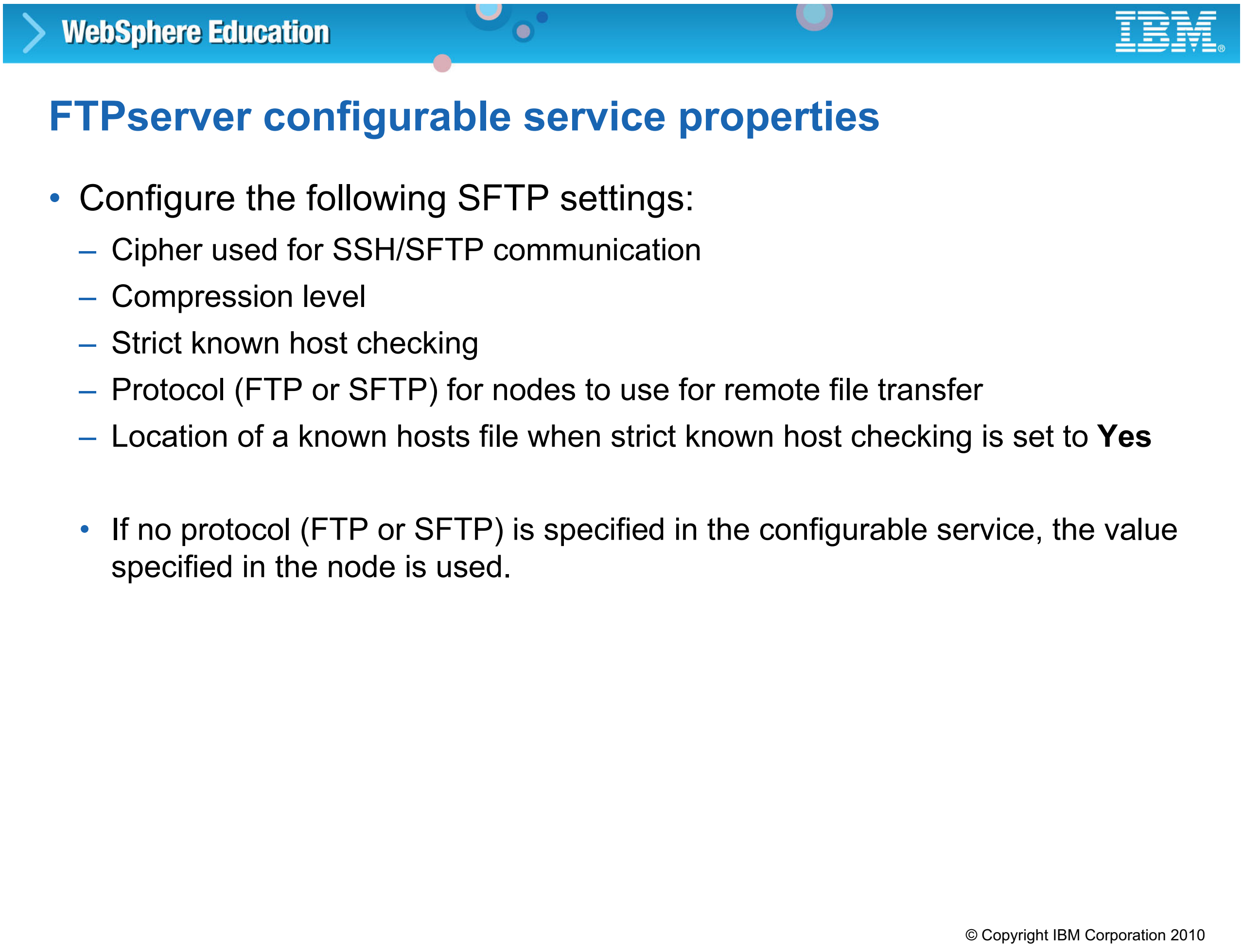
Instructor notes:

Purpose — Give an overview of the requirements for providing SFTP connectivity for message flows.

Details — None.

Additional information — None.

Transition statement — The key to the SFTP implementation is the configurable service.



WebSphere Education

FTPserver configurable service properties

- Configure the following SFTP settings:
  - Cipher used for SSH/SFTP communication
  - Compression level
  - Strict known host checking
  - Protocol (FTP or SFTP) for nodes to use for remote file transfer
  - Location of a known hosts file when strict known host checking is set to **Yes**

- If no protocol (FTP or SFTP) is specified in the configurable service, the value specified in the node is used.

Figure 10-11. FTPserver configurable service properties WM643 / VM6431.0

Notes:

Some of the SFTP properties in the FTPserver configurable service are:

- **Cipher**: The cipher used for encryption. The cipher that you use for encryption depends on your SSH implementation. This property is valid only when SFTP is specified as the protocol. If FTP is used, this property is ignored.
- **Compression**: Specifies the level of compression to be used. This property is valid only when SFTP is specified as the protocol. If FTP is used, this property is ignored.
- **strictHostKeyChecking**: Specifies how host keys are checked during the connection and authentication phase.

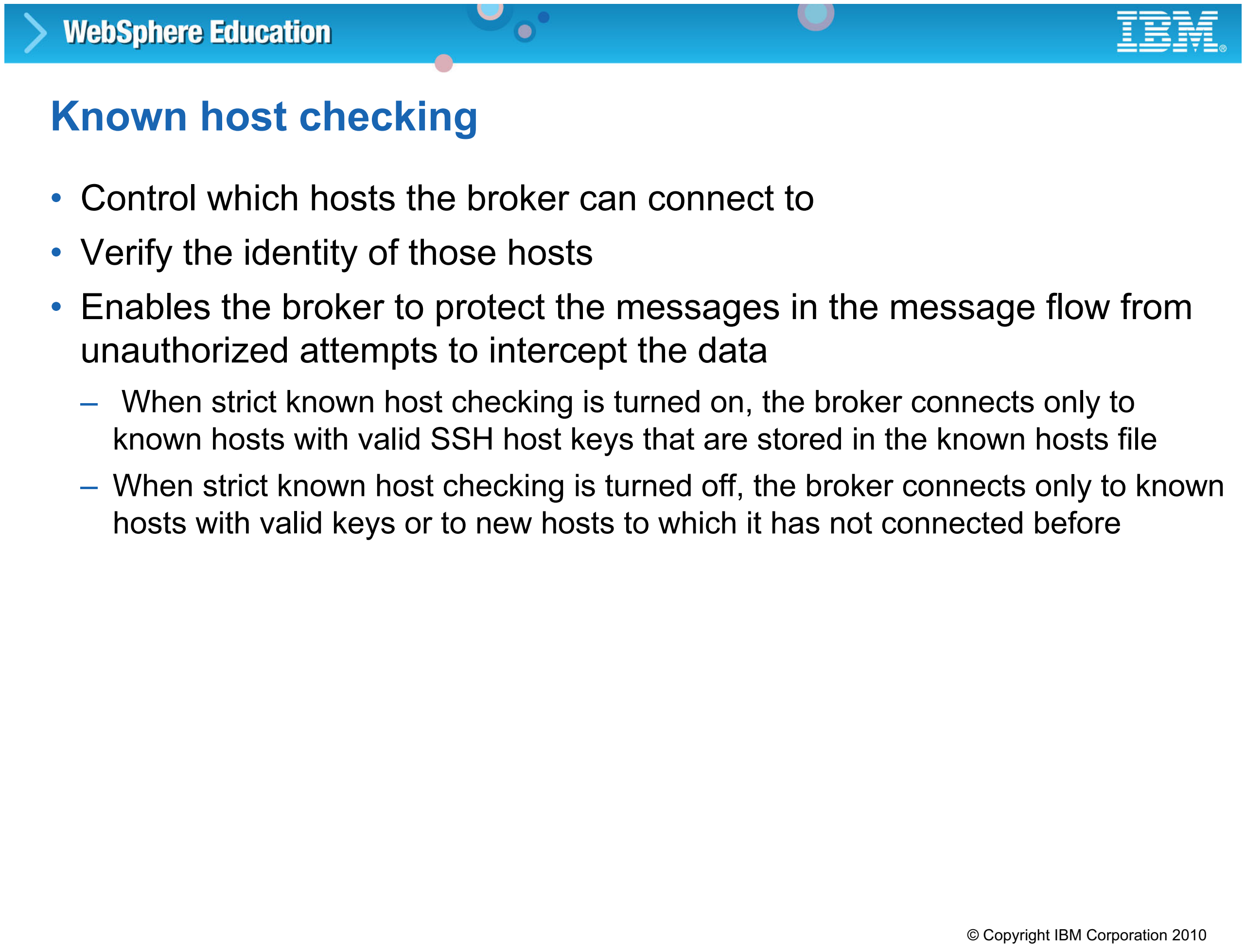
Instructor notes:

Purpose — Provide an overview of the configurable service properties specific to SFTP.

Details — Strict host checking is described in more detail on the next page.

Additional information — None.

Transition statement — Host checking determines how host keys are checked during the connection and authentication phase.



WebSphere Education

Known host checking

- Control which hosts the broker can connect to
- Verify the identity of those hosts
- Enables the broker to protect the messages in the message flow from unauthorized attempts to intercept the data
  - When strict known host checking is turned on, the broker connects only to known hosts with valid SSH host keys that are stored in the known hosts file
  - When strict known host checking is turned off, the broker connects only to known hosts with valid keys or to new hosts to which it has not connected before

Figure 10-12. Known host checking

WM643 / VM6431.0

Notes:

Before it connects to a host to receive or transfer a file (using the FileInput or FileOutput nodes), the broker checks the host key against the keys that are stored in the known hosts file. If the host key does not match an existing entry for the host in the known hosts file, the connection fails. If the host is new (and has no entry in the known hosts file), the result depends on whether strict known that host checking is turned on or off.

When strict known that host checking is turned on (by setting the **strictHostKeyChecking** parameter in the FTPserver configurable service to **Yes**), the broker connects only to known hosts with valid SSH host keys that are stored in the known hosts file

When strict known that host checking is turned off (by setting the **strictHostKeyChecking** parameter in the FTPserver configurable service to **No**) the broker connects only to known hosts with valid keys or to new hosts to which it has not connected before.

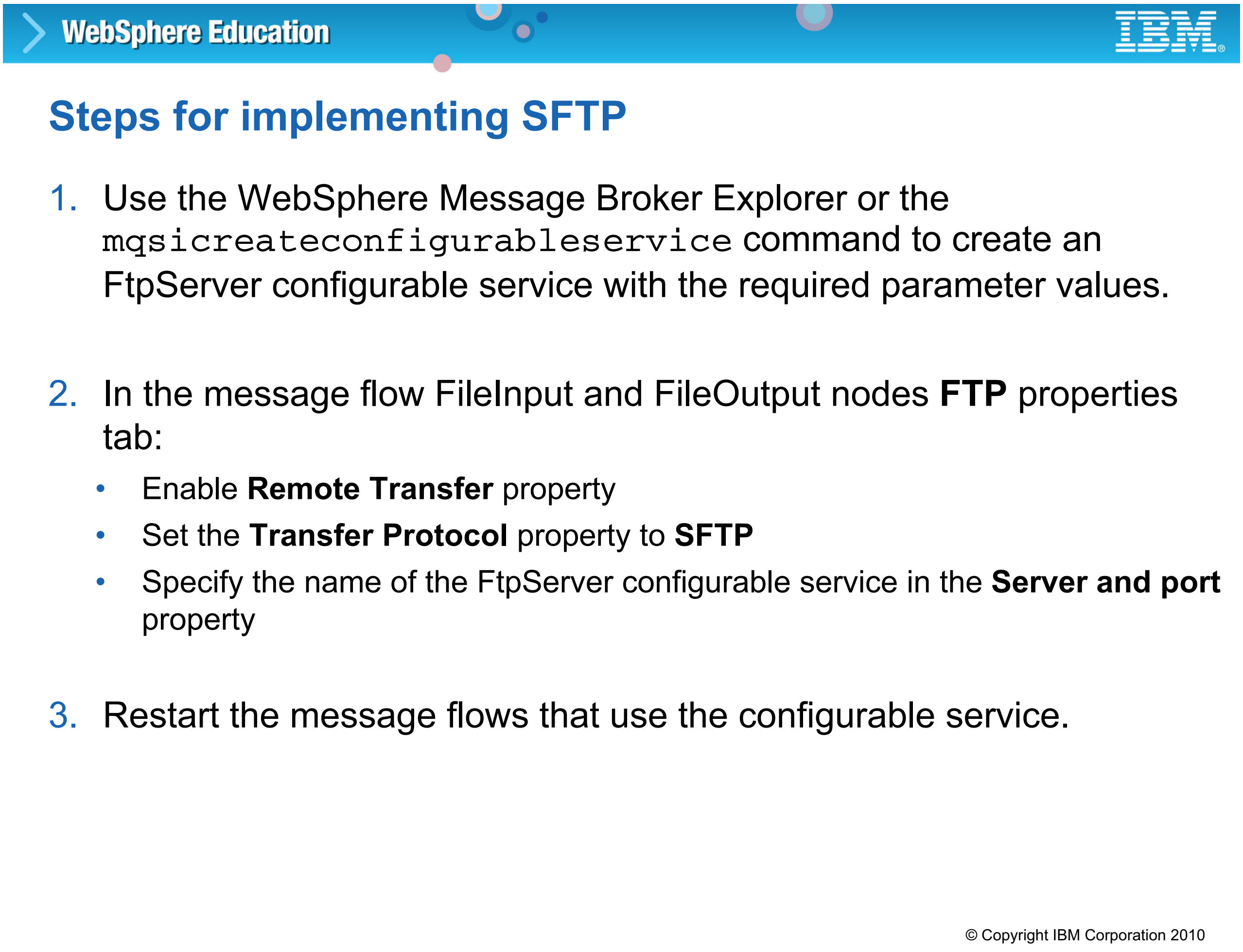
Instructor notes:

Purpose — Provide more information about the known host checking implementation.

Details — None.

Additional information — None.

Transition statement — The steps for implementing SFTP are next.



WebSphere Education
IBM

Steps for implementing SFTP

1. Use the WebSphere Message Broker Explorer or the `mqsicreateconfigurableservice` command to create an FtpServer configurable service with the required parameter values.

2. In the message flow FileInput and FileOutput nodes **FTP** properties tab:
   - Enable **Remote Transfer** property
   - Set the **Transfer Protocol** property to **SFTP**
   - Specify the name of the FtpServer configurable service in the **Server and port** property

3. Restart the message flows that use the configurable service.

Figure 10-13. Steps for implementing SFTP WM643 / VM6431.0

Notes:

The figure lists the steps for implementing SFTP.

You can stop and start the broker to ensure that the configurable service is available to all resources running on the broker.

Instructor notes:

Purpose — List the steps for implementing SFTP.

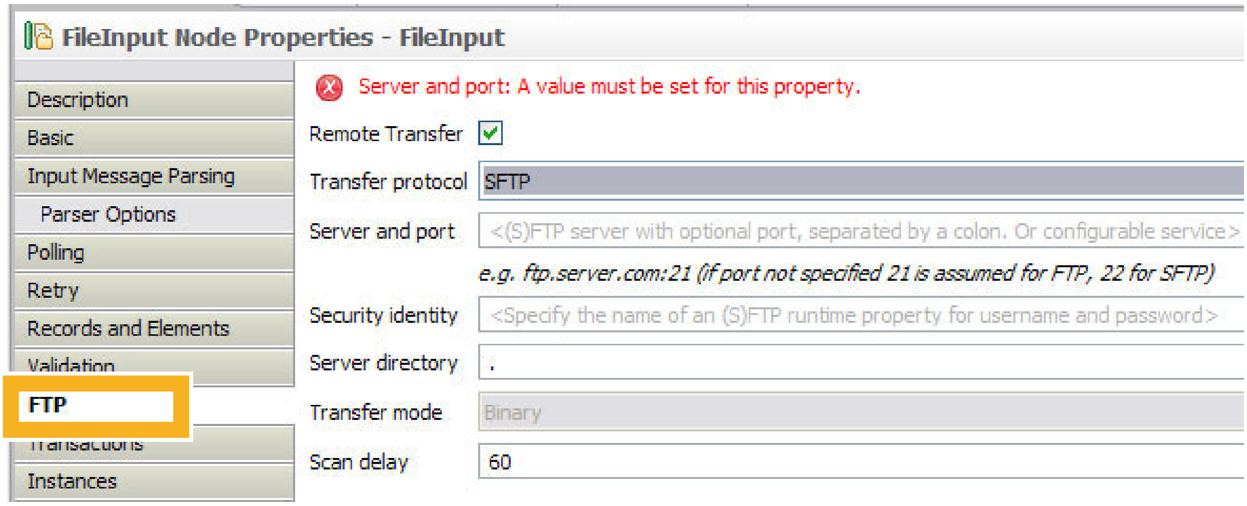
Details — None.

Additional information — None.

Transition statement — An example of a FileInput node shows you the properties that must be set in the node.

WebSphere Education

FileInput node FTP properties



FileInput Node Properties - FileInput

Server and port: A value must be set for this property.

Remote Transfer ☑

Transfer protocol SFTP

Server and port <(S)FTP server with optional port, separated by a colon. Or configurable service>

e.g. ftp.server.com:21 (if port not specified 21 is assumed for FTP, 22 for SFTP)

Security identity <Specify the name of an (S)FTP runtime property for username and password>

Server directory .

Transfer mode Binary

Scan delay 60

- **Remote Transfer** defines whether the node uses the remote file transfer and reads files from either an FTP or SFTP server
- **Transfer protocol** specifies whether to use FTP or SFTP as the protocol for remote transfer
- **Server and port** can be the IP address or name (and, optionally, the port number) of a remote FTP or SFTP server or the name of a configurable service of type FtpServer

Figure 10-14. FileInput node FTP properties

WM643 / VM6431.0

Notes:

The figure highlights the FTP properties available for a FileInput node.

If a configurable service name is specified in the **Server and port** field, any or all the other remote transfer properties on the **FTP** tab can be overridden by the configurable service.

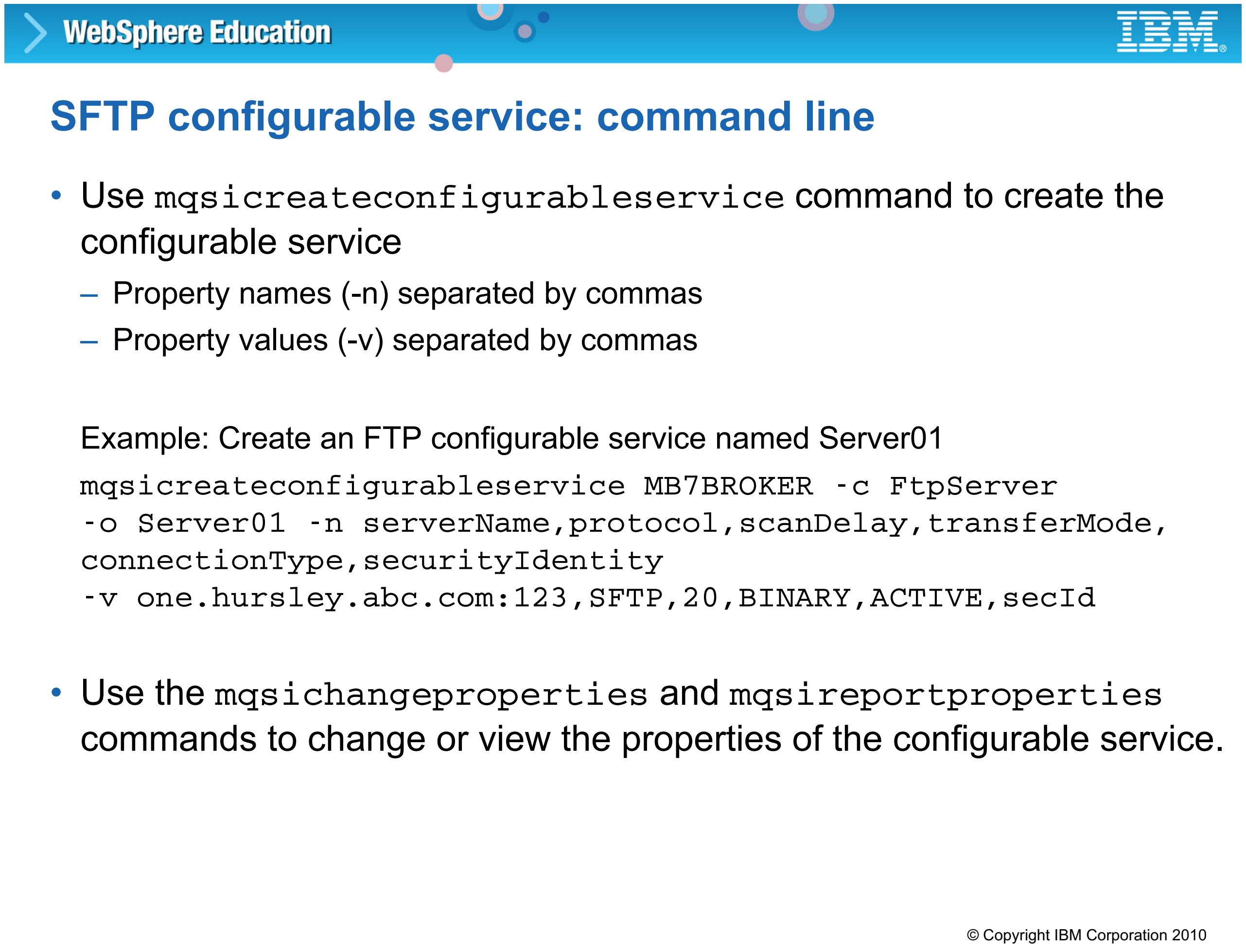
Instructor notes:

Purpose — Highlight the properties specific to SFTP.

Details — None.

Additional information — None.

Transition statement — You can create the FTPserver configurable service from a command line or from WebSphere Message Broker Explorer.



SFTP configurable service: command line

- Use `mqsicreateconfigurableservice` command to create the configurable service
  - Property names (-n) separated by commas
  - Property values (-v) separated by commas

Example: Create an FTP configurable service named Server01

```
mqsicreateconfigurableservice MB7BROKER -c FtpServer
-o Server01 -n serverName,protocol,scanDelay,transferMode,
connectionType,securityIdentity
-v one.hursley.abc.com:123,SFTP,20,BINARY,ACTIVE,secId
```

- Use the `mqsichangeproperties` and `mqsireportproperties` commands to change or view the properties of the configurable service.

Figure 10-15. SFTP configurable service: command line

WM643 / VM6431.0

Notes:

You can use the `mqsicreateconfigurable` service to create the FTPserver configurable service.

You can either enter each name value pair in a separate command or in a single command with each name and value separated by a comma as shown in the example.

Instructor notes:

Purpose — Provide an example of how to create the FTPserver configurable service from a command line.

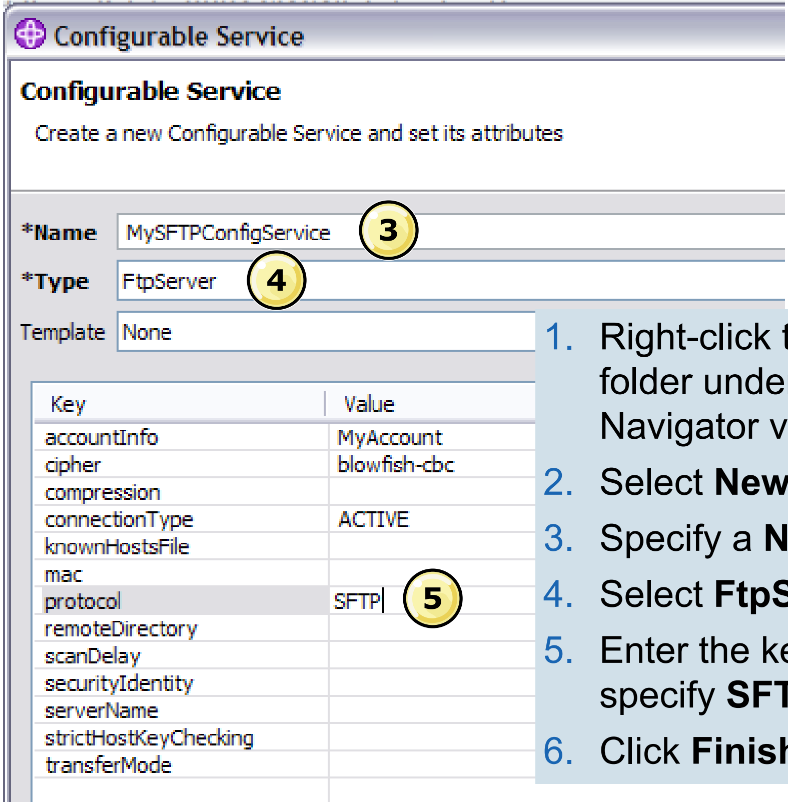
Details — None.

Additional information — None.

Transition statement — You might find it easier to create the configurable service from WebSphere Message Broker Explorer.

WebSphere Education

SFTP configurable service: Message Broker Explorer



Configurable Service

Configurable Service

Create a new Configurable Service and set its attributes

*Name: MySFTPConfigService

*Type: FtpServer

Template: None

Key	Value
accountInfo	MyAccount
cipher	blowfish-cbc
compression	
connectionType	ACTIVE
knownHostsFile	
mac	
protocol	SFTP
remoteDirectory	
scanDelay	
securityIdentity	
serverName	
strictHostKeyChecking	
transferMode	

1. Right-click the **Configurable Services** folder under the broker in the Navigator view.
2. Select **New > Configurable Service**.
3. Specify a **Name**.
4. Select **FtpServer** for the **Type**.
5. Enter the key values being sure to specify **SFTP** as the **protocol**
6. Click **Finish**.

Figure 10-16. SFTP configurable service: WebSphere Message Broker Explorer WM643 / VM6431.0

Notes:

The figure shows the steps for creating a configurable service using WebSphere Message Broker Explorer. See the WebSphere Message Broker Information Center for the valid vales for each key.

When you create a configurable service using WebSphere Message Broker Explorer, the command-line syntax is also created and displayed for reuse.

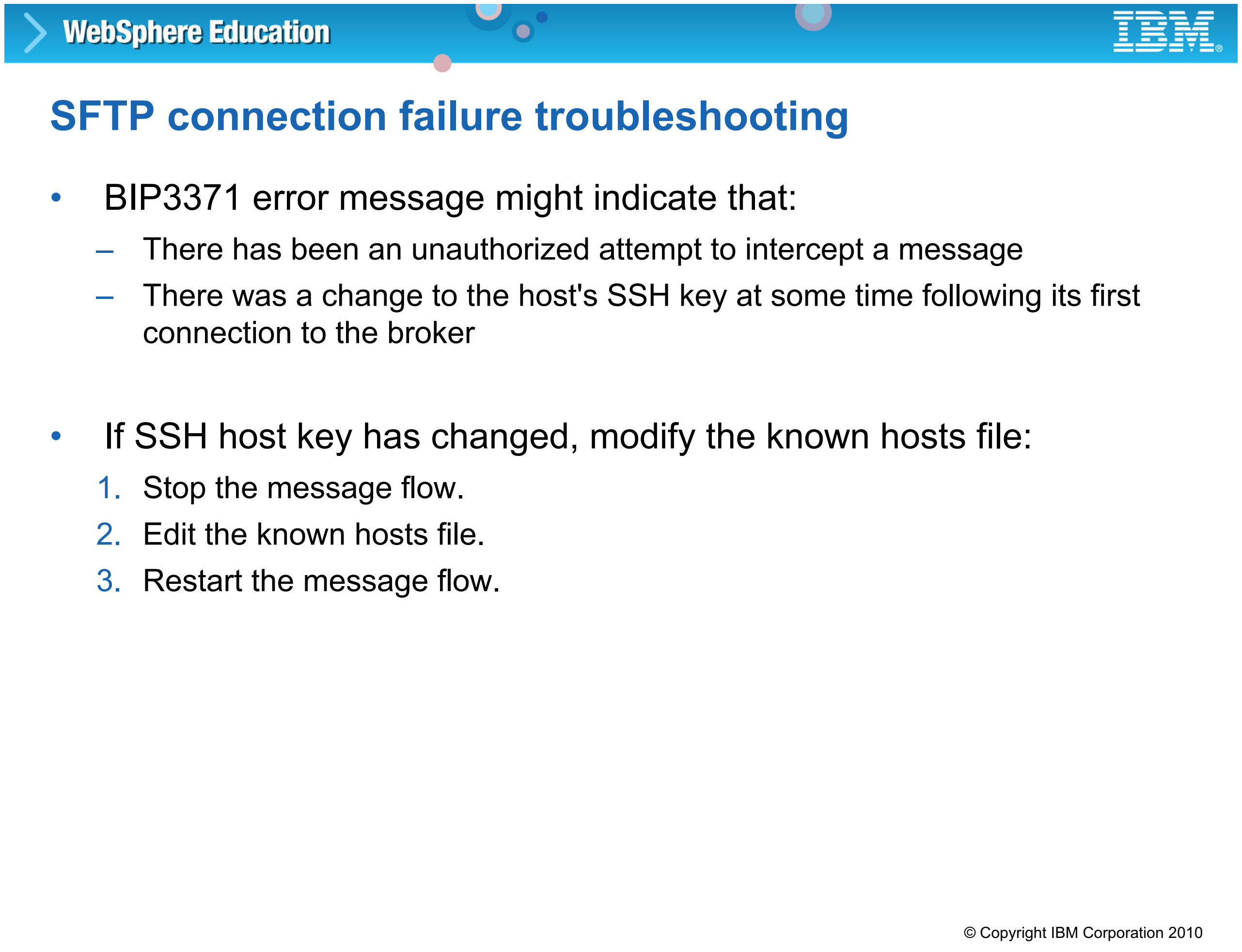
Instructor notes:

Purpose — Show the steps for creating an FTPserver configurable service using WebSphere Message Broker Explorer.

Details — Demonstrate creating the configurable service so that the students can see the command line being created as you enter parameter values.

Additional information — None.

Transition statement — The most common problems with using SFTP are physical connection problems and authorization errors.



WebSphere Education

SFTP connection failure troubleshooting

- BIP3371 error message might indicate that:
  - There has been an unauthorized attempt to intercept a message
  - There was a change to the host's SSH key at some time following its first connection to the broker
- If SSH host key has changed, modify the known hosts file:
  1. Stop the message flow.
  2. Edit the known hosts file.
  3. Restart the message flow.

© Copyright IBM Corporation 2010

Figure 10-17. SFTP connection failure troubleshooting WM643 / VM6431.0

Notes:

If the SSH key has changed and you have strict known host checking turned on, correct the entry for the host in the known hosts file that you specified in the **knownHostsFile** parameter of the FTPserver configurable service.

If you have strict “known host checking” turned off, remove the host's entry from the broker-managed known hosts file. The known hosts file is in the `\components\BROKERNAME\EG-UID\config\known_hosts` subdirectory of the directory in which WebSphere Message Broker is installed.

- On Windows the default directory is `C:\Documents and Settings\All Users\Application Data\IBM\MQSI\components\BROKERNAME\EG-UID\config\known_hosts`.
- On UNIX systems the default directory is `/var/mqsi/components/BROKERNAME/EG-UID/config/known_hosts`.

When the broker attempts to reconnect, it adds the new host key to the known hosts file.

Instructor notes:

Purpose — Provide some information about possible SFTP errors.

Details — None.

Additional information — None.

Transition statement — WebSphere Message Broker also supports user-defined extensions.

10.3.User-defined extensions

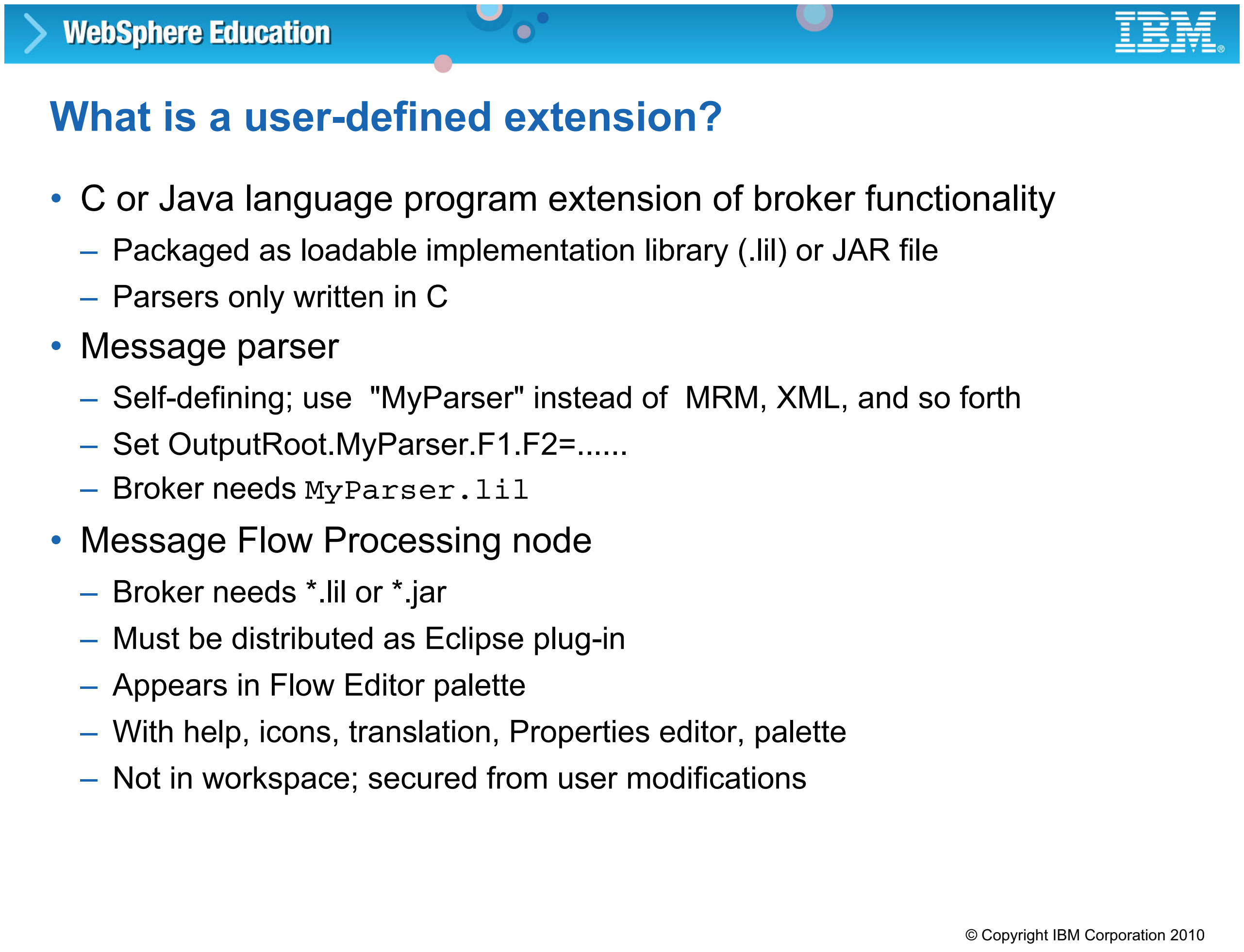
Instructor topic introduction

What students will do — Learn that *implementing* a plug-in node is just a matter of following a specific procedure.

How students will do it — Listen to the lecture and participate in classroom discussion. Exercise 9 will permit the students to implement a plug-in node.

What students will learn — The toolkit components and the procedure to follow to implement a plug-in node.

How this will help students on their job — The job of installing product extensions routinely is given to the administrator to accomplish.



WebSphere Education
IBM

What is a user-defined extension?

- C or Java language program extension of broker functionality
  - Packaged as loadable implementation library (.lil) or JAR file
  - Parsers only written in C
- Message parser
  - Self-defining; use "MyParser" instead of MRM, XML, and so forth
  - Set OutputRoot.MyParser.F1.F2=......
  - Broker needs `MyParser.lil`
- Message Flow Processing node
  - Broker needs *.lil or *.jar
  - Must be distributed as Eclipse plug-in
  - Appears in Flow Editor palette
  - With help, icons, translation, Properties editor, palette
  - Not in workspace; secured from user modifications

Figure 10-18. What is a user-defined extension?

WM643 / VM6431.0

Notes:

You can create and implement the following types of user-defined extensions: Input nodes, message processing nodes, output nodes, and parsers.

User-defined nodes and parsers that you create can be used with both the nodes and parsers supplied with WebSphere Message Broker, and with third-party supplied nodes and parsers. You can also configure a user-defined node to use a user-defined parser that you have written, rather than one of the supplied parsers.

Both nodes and parsers use ANSI C. For nodes only, you can also program in Java. A list of supported compilers is included in the Introduction and Planning Guide.

These extensions to the broker are distributed as dynamic link libraries (DLLs) on Windows NT, and as shared objects in UNIX. For implementation in WebSphere Message Broker, the extension must be changed from `dll` to `lil`. Java nodes are distributed as JARs (Java repository files).

Instructor notes:

Purpose — Discuss some of the requirements necessary to implement a user-defined extension.

Details — Focus more on the administration aspects instead of development.

To write user-defined extensions, you must be a skilled programmer with some knowledge of WebSphere Message Broker and its architecture. Make sure that you have the skills and knowledge required. You also need the time to test and debug your user-defined node or parser, and a safe environment in which to do this.

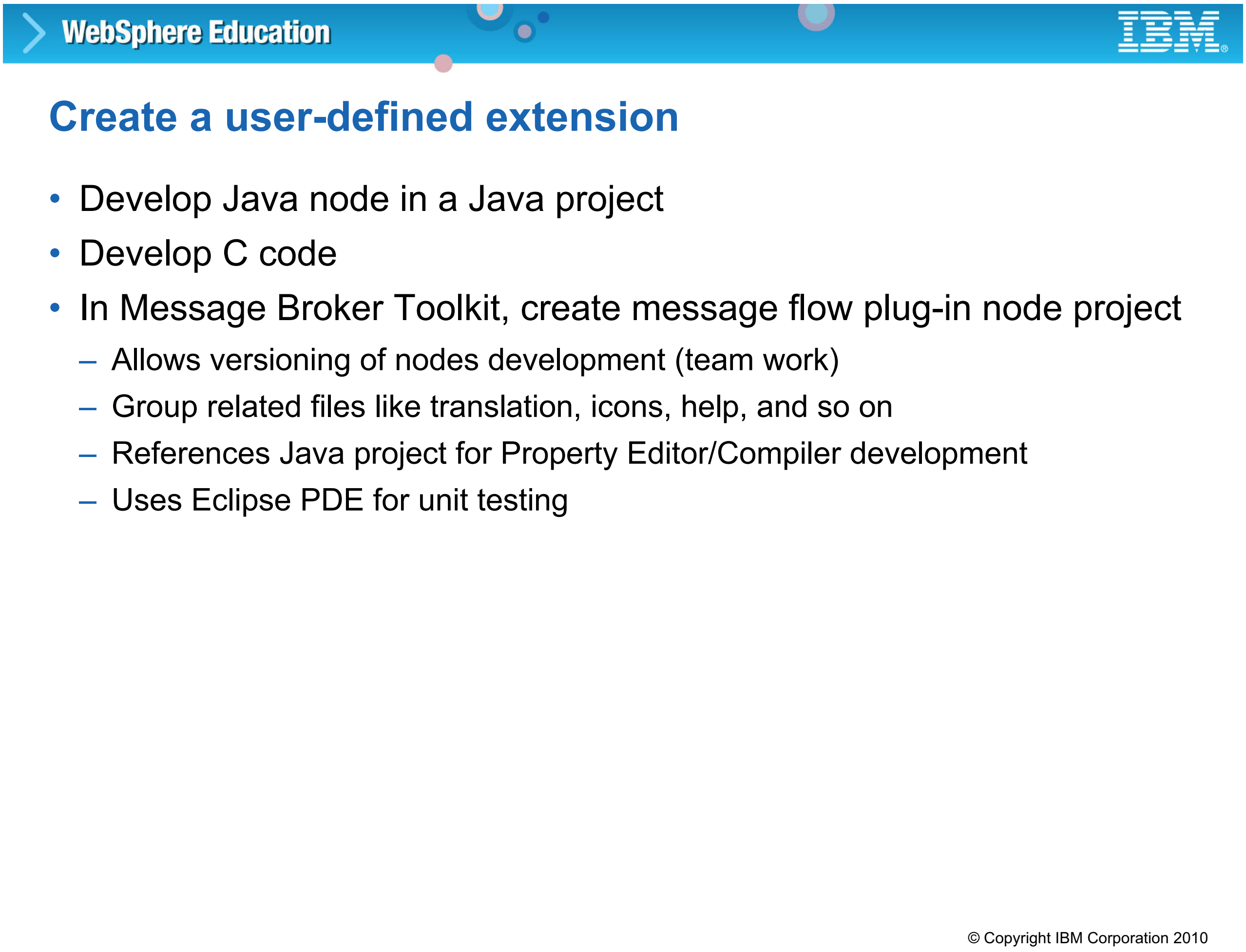
Also note that the maintenance and servicing of user-defined extensions is the client’s responsibility. There will be someone available who can perform future updates or fixes.

A user-defined extension might be appropriate in the following situations:

- When you cannot manipulate the supplied nodes or parsers to perform the function you require. For example, you might want to connect to another software component in your message flow outside of WebSphere MQ. If there is no supplied node for doing this, you must create your own.
- When you can improve performance, ease of use, or reliability by using your own user-defined extensions in place of the supplied nodes or parsers.
- If the available choices are not appropriate for your requirement. You can create user-defined extensions to handle internal, customer-specific, or generic commercial messages formats.

Additional information — The reason for changing the name of the DLL to LIL is because of the way the nodes are loaded. The broker starts an instance of each node when it starts. If the nodes were called DLL, the broker would have no way of differentiating between the nodes and other DLLs.

Transition statement — Here is a brief overview what must be done to develop a user-defined extension.



WebSphere Education

Create a user-defined extension

- Develop Java node in a Java project
- Develop C code
- In Message Broker Toolkit, create message flow plug-in node project
  - Allows versioning of nodes development (team work)
  - Group related files like translation, icons, help, and so on
  - References Java project for Property Editor/Compiler development
  - Uses Eclipse PDE for unit testing

Figure 10-19. Create a user-defined extension WM643 / VM6431.0

Notes:

You can create Java nodes from within the workbench, using the provided plug-in development environment (PDE).

For user-defined nodes only, you need to create the user interface representation of it in the workbench.

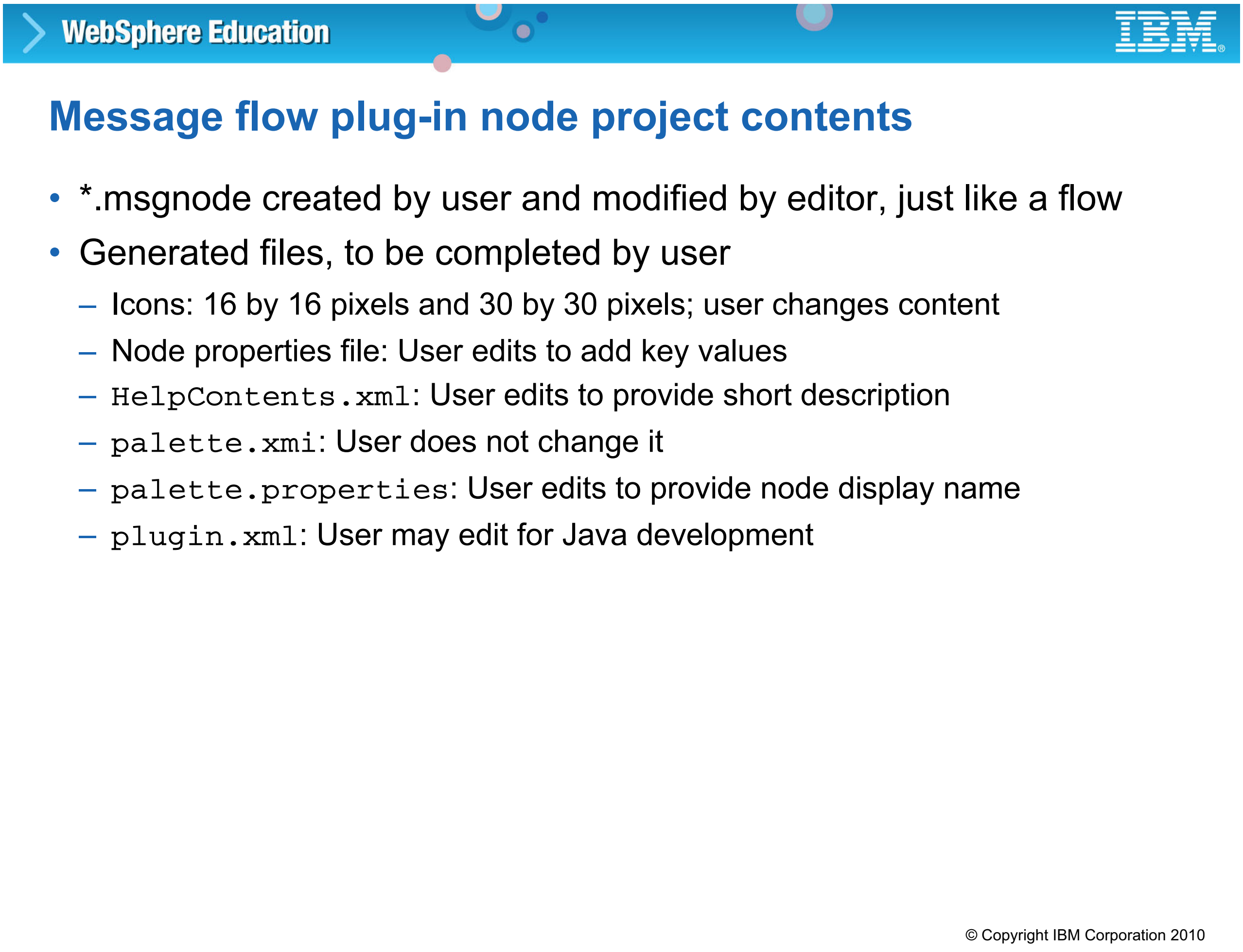
Instructor notes:

Purpose — Provide a brief overview of the tasks that must be done to create user-defined extensions. There will be no detailed further discussion in this course.

Details — A plug-in project can contain any Toolkit-developed plug-in resource. In WebSphere Message Broker, it is used for user-defined message nodes.

Additional information — None.

Transition statement — The next task is to distribute the user-defined extension.



WebSphere Education

Message flow plug-in node project contents

- *.msgnode created by user and modified by editor, just like a flow
- Generated files, to be completed by user
  - Icons: 16 by 16 pixels and 30 by 30 pixels; user changes content
  - Node properties file: User edits to add key values
  - `HelpContents.xml`: User edits to provide short description
  - `palette.xmi`: User does not change it
  - `palette.properties`: User edits to provide node display name
  - `plugin.xml`: User may edit for Java development

Figure 10-20. Message flow plug-in node project contents WM643 / VM6431.0

Notes:

A plug-in project is used for user-defined message flow processing nodes and parsers. Plug-in development projects used to create message flow processing nodes contain these resources:

- `.msgnode` file contains the definition of the node.
- `palette.xmi` file lists the user-defined nodes in the project and the Message editor palette group in which they are shown.
- `.html` file contains the help files for the node.
- `.java` file contains the Java source code for property editors and compilers.
- `plugin.xml` file is the plug-in manifest file that adds the node to the WebSphere Message Broker installation.
- `.properties` file contains the national language properties and the translations for node properties and terminals.
- `.gif` file containing the icon image to associate with the node.

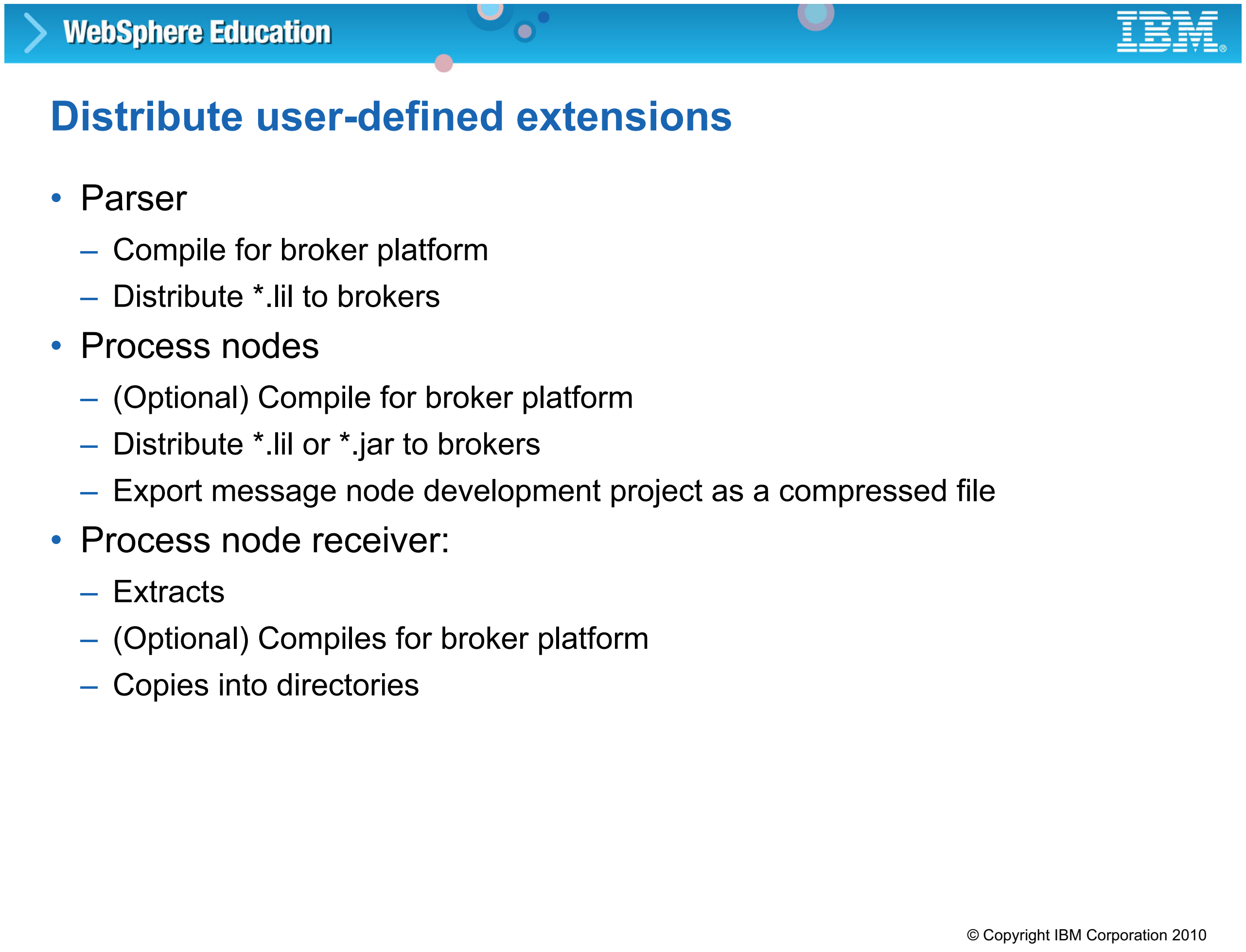
Instructor notes:

Purpose — Identify the programming requirements and the parts-list that goes with a plug-in node.

Details — None.

Additional information — None.

Transition statement — The next job is to do something with the parts-list, but what?



WebSphere Education

Distribute user-defined extensions

- Parser
  - Compile for broker platform
  - Distribute *.lil to brokers
- Process nodes
  - (Optional) Compile for broker platform
  - Distribute *.lil or *.jar to brokers
  - Export message node development project as a compressed file
- Process node receiver:
  - Extracts
  - (Optional) Compiles for broker platform
  - Copies into directories

Figure 10-21. Distribute user-defined extensions WM643 / VM6431.0

Notes:

You can write user-defined nodes in C or Java, or you can use a subflow to create a node. You can write user-defined parsers and exits only in C.

For user-defined nodes only, you must create a WebSphere Message Broker Toolkit Eclipse plug-in. For user-defined nodes created in Java and C, you must also create the run time `.lil` or `.jar` file. The WebSphere Message Broker Toolkit plug-in adds the user-defined node to the node palette in the Message Flow editor, so that you can include the new node in message flows.

A user-defined node is installed differently depending on how it was packaged.

- If the user-defined node was packaged as plug-in JAR files, copy your JAR files into the plug-ins folder in your WebSphere Message Broker Toolkit installation location.
- If the user-defined node was packaged as an update site, install the plug-ins from the update site.

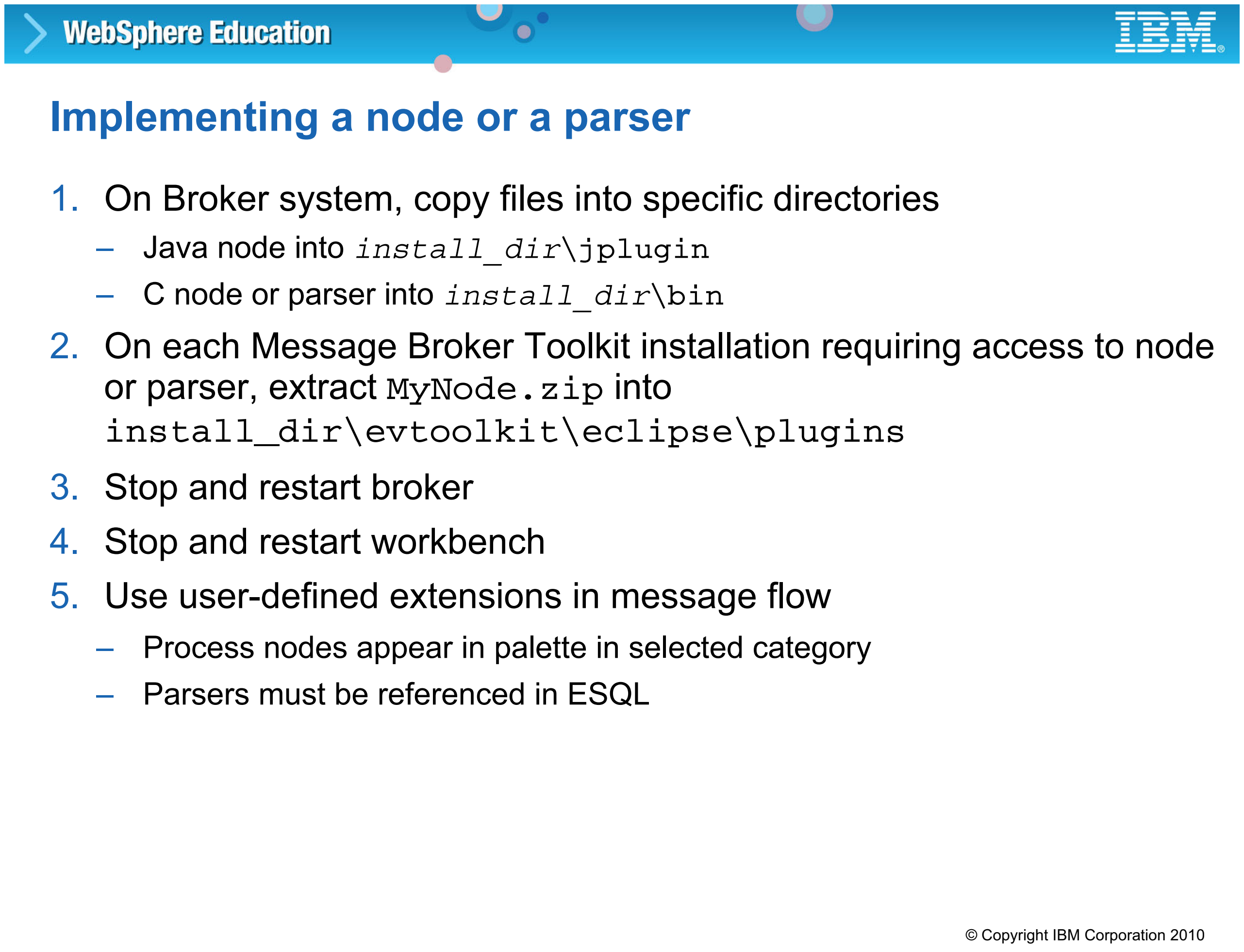
Instructor notes:

Purpose — Provide a summary of the steps for distributing a user-defined extension.

Details — None.

Additional information — None.

Transition statement — Next is more detail on implementing a user-defined node or parser.



WebSphere Education

Implementing a node or a parser

1. On Broker system, copy files into specific directories
   - Java node into *install_dir*\jplugin
   - C node or parser into *install_dir*\bin
2. On each Message Broker Toolkit installation requiring access to node or parser, extract MyNode.zip into install_dir\evtoolkit\eclipse\plugins
3. Stop and restart broker
4. Stop and restart workbench
5. Use user-defined extensions in message flow
   - Process nodes appear in palette in selected category
   - Parsers must be referenced in ESQL

© Copyright IBM Corporation 2010

Figure 10-22. Implementing a node or a parser WM643 / VM6431.0

Notes:

Each user of the workbench must have the node project in the <install_dir>\evtoolkit\eclipse\plugins directory to be able to add the node to their message flows.

Unless you are a developer, there is no need for the node project to be implemented in your toolkit. As an administrator, you want the runtime environment to handle the executable version of the node.

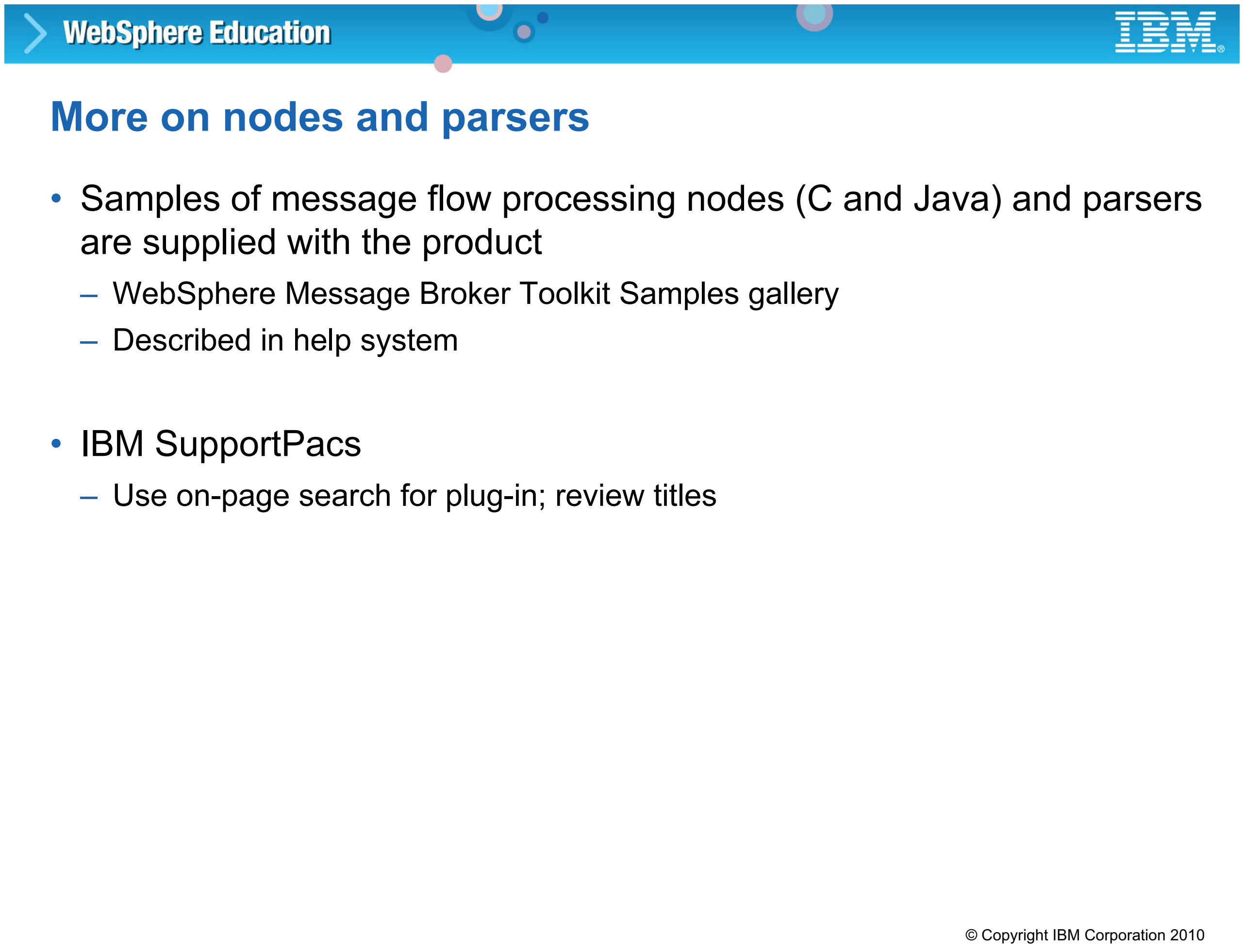
Instructor notes:

Purpose — Explain this most common scenario of implementing a user-defined extension.

Details — None.

Additional information — None.

Transition statement —



WebSphere Education

IBM

More on nodes and parsers

- Samples of message flow processing nodes (C and Java) and parsers are supplied with the product
  - WebSphere Message Broker Toolkit Samples gallery
  - Described in help system
- IBM SupportPacs
  - Use on-page search for plug-in; review titles

Figure 10-23. More on nodes and parsers WM643 / VM6431.0

Notes:

WebSphere Message Broker provides some sample code to help you understand how to write user-defined nodes and parsers. The samples consist of a sample parser, and the following sample nodes:

- SwitchA node, implemented in both C and Java versions, that propagates an input message to one of several output terminals depending on the message content.
- TransformA node, implemented in both C and Java versions, that performs a simple message transformation.

Each sample node consists of the source files and some files that you can use to test each node.

Instructor notes:

Purpose — Provide information about where to get additional information about creating and distributing user-defined extensions.

Details — The samples include well-documented procedures that can be used as a template for user-defined extension implementation.

Additional information — None.

Transition statement — The final topic in this unit provides more information about SupportPacs and WebSphere Message Broker product extensions.

10.4.WebSphere Message Broker SupportPacs

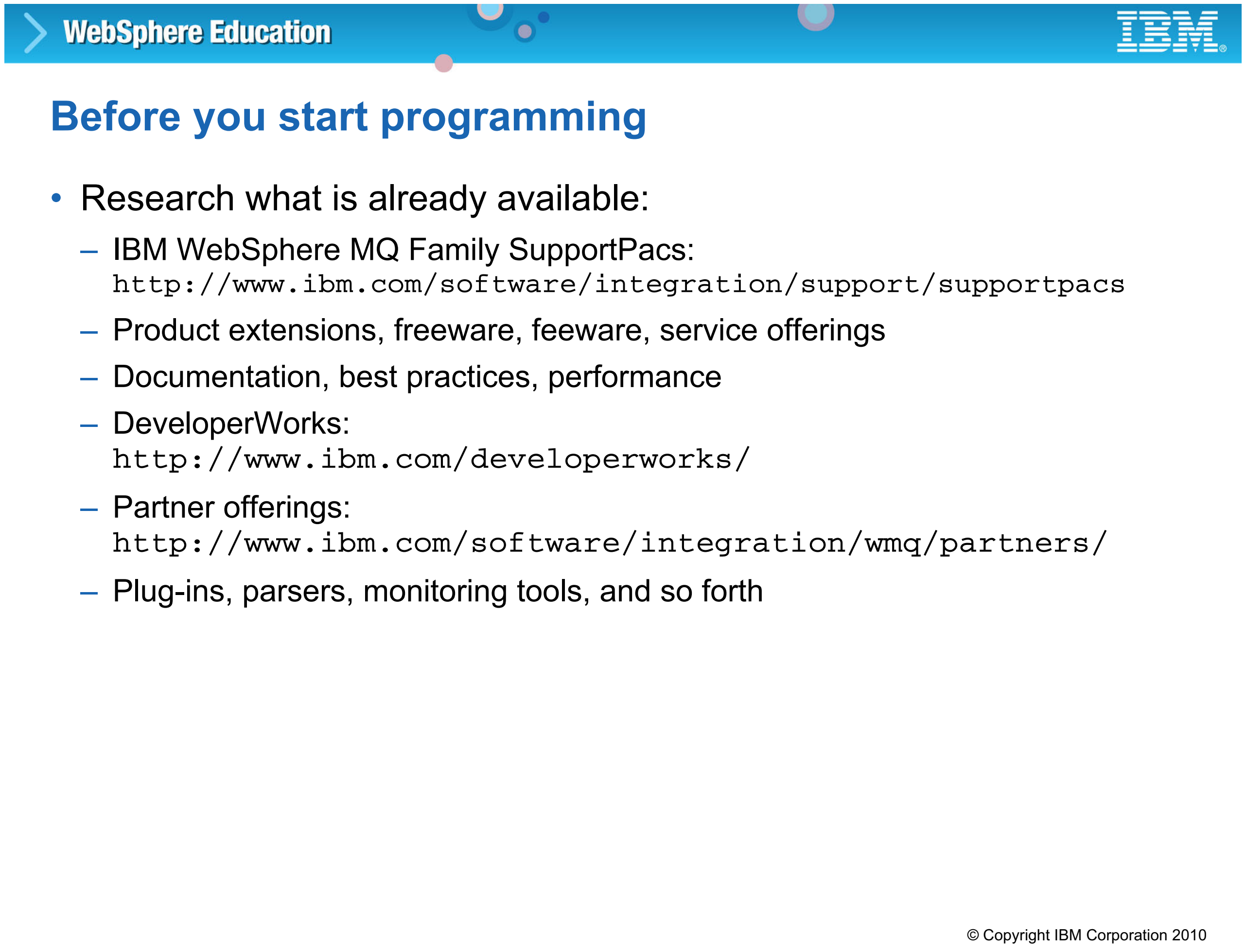
Instructor topic introduction

What students will do — Gain a better understanding of the extensible nature of the WebSphere Message Brokers.

How students will do it — Listen to the lecture and participate in classroom discussion. Exercise 9 will give students an opportunity to obtain and implement a plug-in from the IBM WebSphere Message Brokers SupportPac web page.

What students will learn — The functionality possible with the WebSphere Message Brokers is unlimited given that the architecture permits programming extensions. However, there are some rules programmers must follow.

How this will help students on their job — Students will be able to plan for an implementation of a user-written extension that is being developed.


WebSphere Education
IBM

Before you start programming

- Research what is already available:
  - IBM WebSphere MQ Family SupportPacs: `http://www.ibm.com/software/integration/support/supportpacs`
  - Product extensions, freeware, feeware, service offerings
  - Documentation, best practices, performance
  - DeveloperWorks: `http://www.ibm.com/developerworks/`
  - Partner offerings: `http://www.ibm.com/software/integration/wmq/partners/`
  - Plug-ins, parsers, monitoring tools, and so forth

Figure 10-24. Before you start programming WM643 / VM6431.0

Notes:

If you become involved in a plan to implement user-defined extensions, be sure to first suggest a thorough search of already-written extensions. You might find just what you are looking for.

Partner offerings usually involve a fee; SupportPac material, however, is free. In either case, it might be more practical to have your programmers doing something other than writing specialized code.

As an administrator, you must be involved in the implementation of the finished product, whether purchased, downloaded, or written locally.

Instructor notes:

Purpose — Identify sources of available code, some for free.

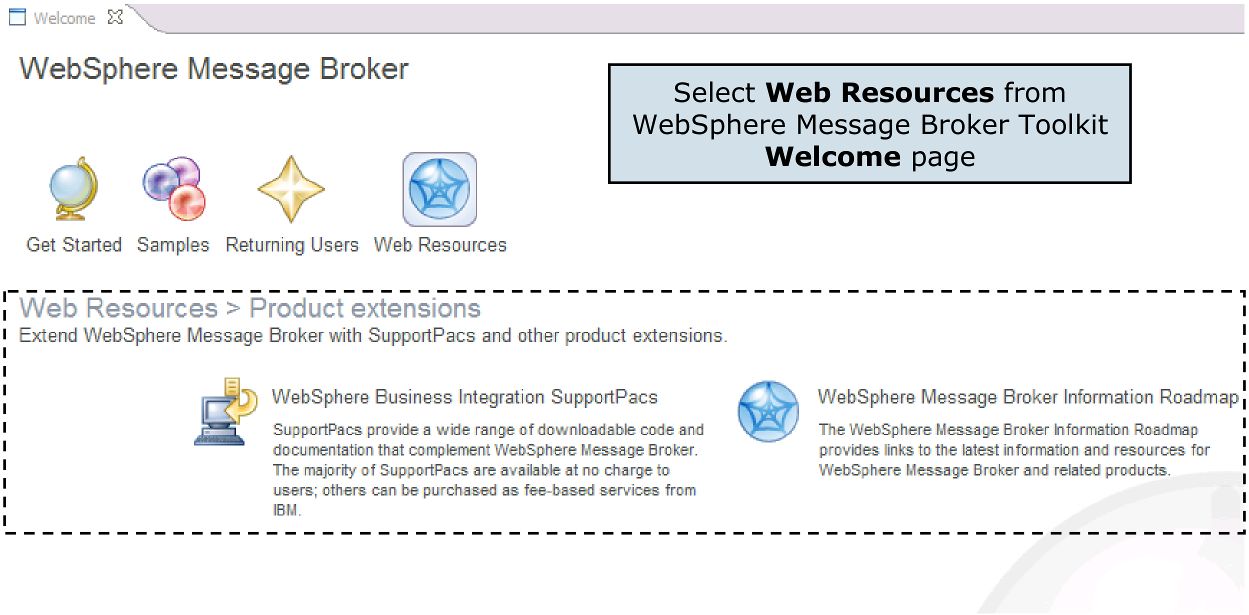
Details — None.

Additional information — None.

Transition statement — The quickest way to find product extensions and SupportPacs is from links provided on the Welcome page in WebSphere Message Broker Explorer and WebSphere Message Broker Toolkit.

WebSphere Education

Finding WebSphere Message Broker product extensions



Select **Web Resources** from WebSphere Message Broker Toolkit **Welcome** page

Web Resources > Product extensions

Extend WebSphere Message Broker with SupportPacs and other product extensions.

WebSphere Business Integration SupportPacs

SupportPacs provide a wide range of downloadable code and documentation that complement WebSphere Message Broker. The majority of SupportPacs are available at no charge to users; others can be purchased as fee-based services from IBM.

WebSphere Message Broker Information Roadmap

The WebSphere Message Broker Information Roadmap provides links to the latest information and resources for WebSphere Message Broker and related products.

Figure 10-25. Finding WebSphere Message Broker product extensions WM643 / VM6431.0

Notes:

The links to WebSphere Message Broker product extensions can be found in WebSphere Message Broker Toolkit under **Help > Web Resources.**

Instructor notes:

Purpose — Show students how to access product extensions.

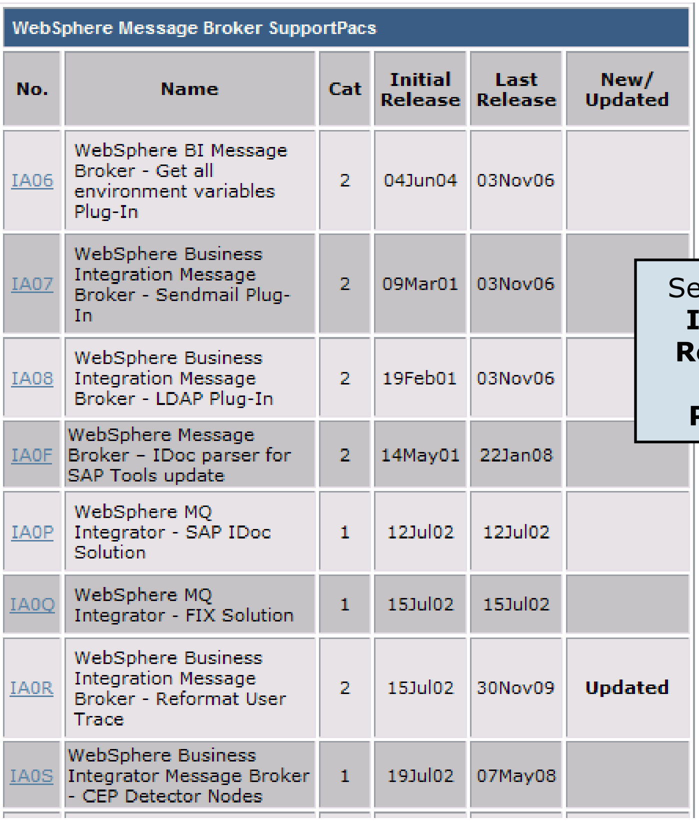
Details — It is not necessary to go into specific product extensions.

Additional information — None.

Transition statement — SupportPacs are listed by identification number.

WebSphere Education

WebSphere Message Broker SupportPacs



WebSphere Message Broker SupportPacs

No.	Name	Cat	Initial Release	Last Release	New/ Updated
IA06	WebSphere BI Message Broker - Get all environment variables Plug-In	2	04Jun04	03Nov06	
IA07	WebSphere Business Integration Message Broker - Sendmail Plug-In	2	09Mar01	03Nov06	
IA08	WebSphere Business Integration Message Broker - LDAP Plug-In	2	19Feb01	03Nov06	
IA0F	WebSphere Message Broker – IDoc parser for SAP Tools update	2	14May01	22Jan08	
IA0P	WebSphere MQ Integrator - SAP IDoc Solution	1	12Jul02	12Jul02	
IA0Q	WebSphere MQ Integrator - FIX Solution	1	15Jul02	15Jul02	
IA0R	WebSphere Business Integration Message Broker - Reformat User Trace	2	15Jul02	30Nov09	**Updated**
IA0S	WebSphere Business Integrator Message Broker - CEP Detector Nodes	1	19Jul02	07May08	

Select **WebSphere Business Integration SupportPacs Resources** from WebSphere Message Broker Toolkit **Product Extensions** page

Figure 10-26. WebSphere Message Broker SupportPacs WM643 / VM6431.0

Notes:

The figure shows an example of the WebSphere Message Broker SupportPacs site, which lists all the available SupportPacs available for download.

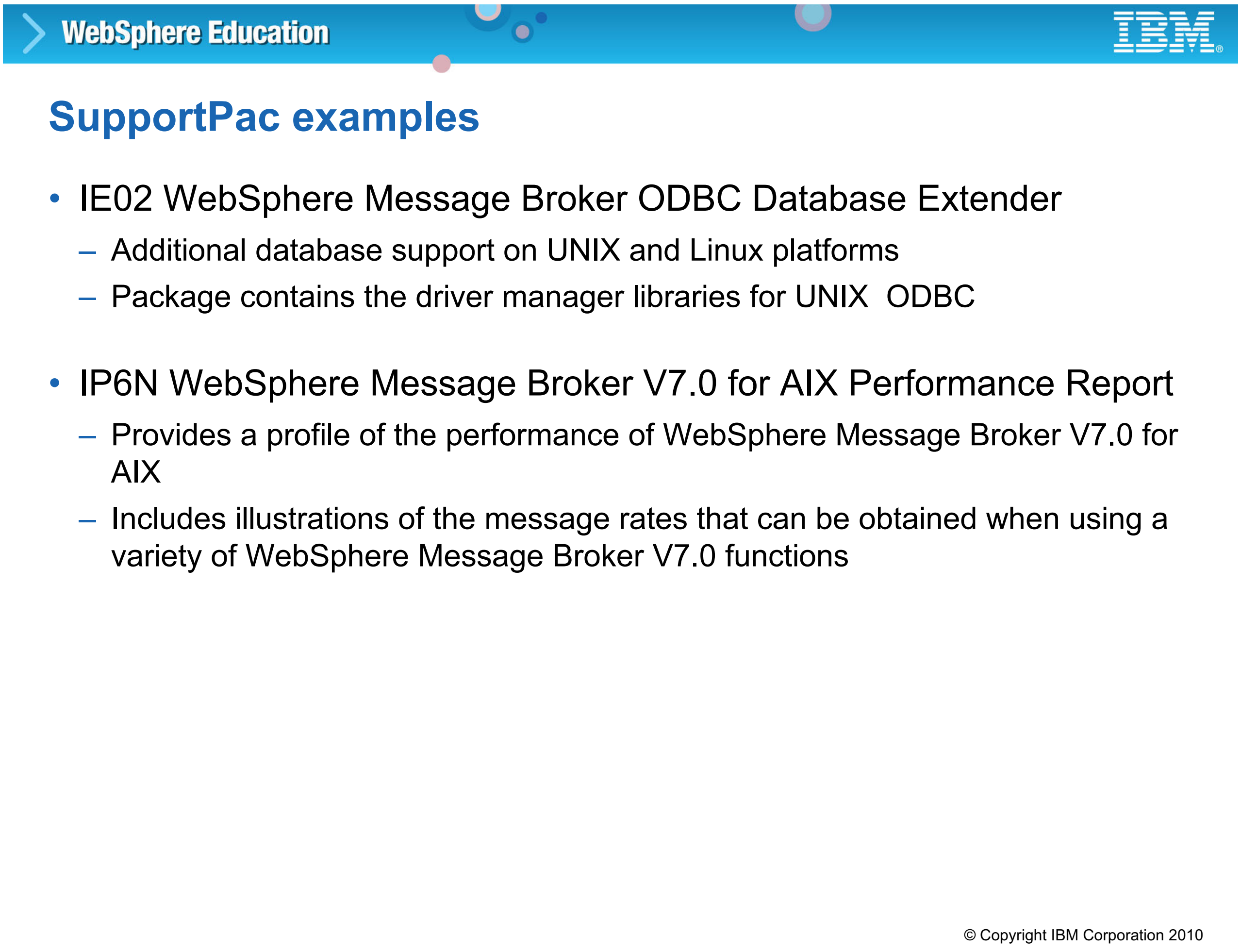
Instructor notes:

Purpose — Show an example of the WebSphere Message Broker SupportPacs web page.

Details — None.

Additional information — None.

Transition statement — An example SupportPac is IE02 (already mentioned) and IP6N.



WebSphere Education

SupportPac examples

- IE02 WebSphere Message Broker ODBC Database Extender
  - Additional database support on UNIX and Linux platforms
  - Package contains the driver manager libraries for UNIX ODBC
- IP6N WebSphere Message Broker V7.0 for AIX Performance Report
  - Provides a profile of the performance of WebSphere Message Broker V7.0 for AIX
  - Includes illustrations of the message rates that can be obtained when using a variety of WebSphere Message Broker V7.0 functions

Figure 10-27. SupportPac examples WM643 / VM6431.0

Notes:

Each SupportPac includes a description and installation instructions.

Instructor notes:

Purpose — Show an example of SupportPac.

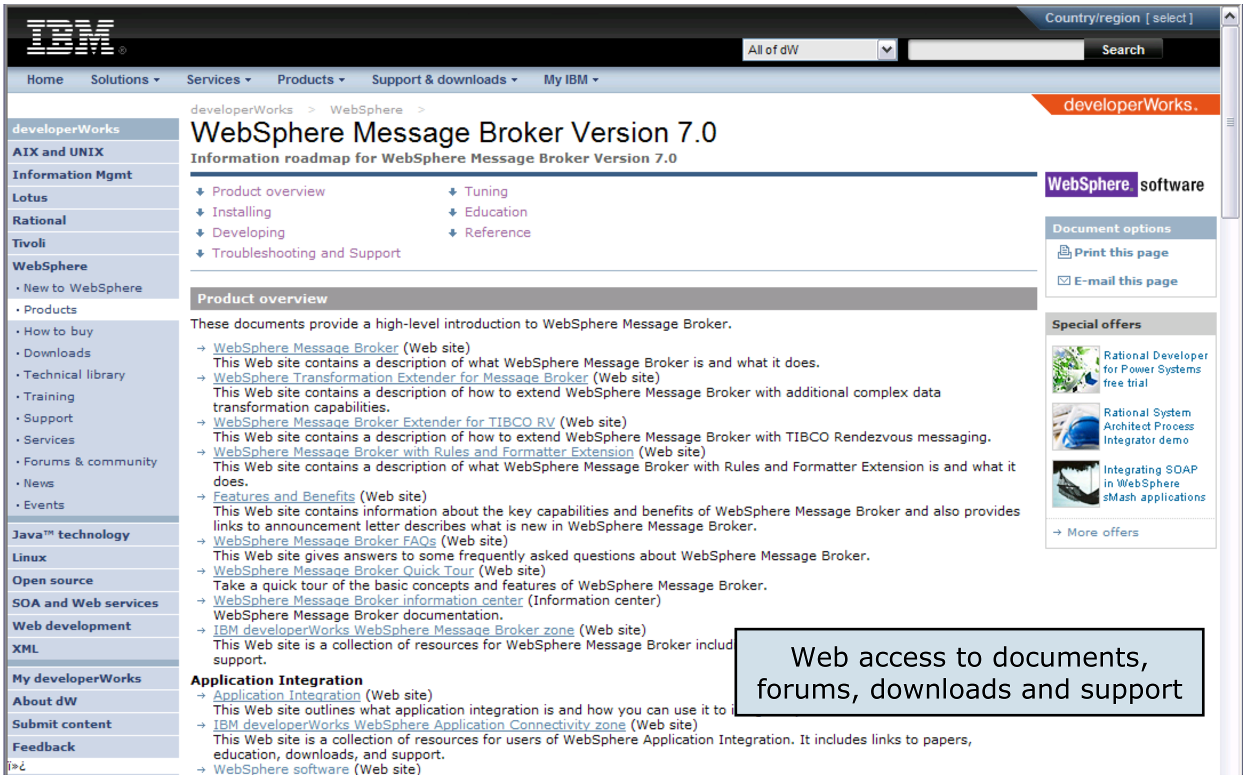
Details — None.

Additional information — None.

Transition statement — The WebSphere Message Broker web page is also a good starting point for information.

WebSphere Education

WebSphere Message Broker information roadmap



WebSphere Message Broker Version 7.0

Information roadmap for WebSphere Message Broker Version 7.0

- Product overview
- Installing
- Developing
- Troubleshooting and Support
- Tuning
- Education
- Reference

Product overview

These documents provide a high-level introduction to WebSphere Message Broker.

→ WebSphere Message Broker (Web site)
This Web site contains a description of what WebSphere Message Broker is and what it does.
→ WebSphere Transformation Extender for Message Broker (Web site)
This Web site contains a description of how to extend WebSphere Message Broker with additional complex data transformation capabilities.
→ WebSphere Message Broker Extender for TIBCO RV (Web site)
This Web site contains a description of how to extend WebSphere Message Broker with TIBCO Rendezvous messaging.
→ WebSphere Message Broker with Rules and Formatter Extension (Web site)
This Web site contains a description of what WebSphere Message Broker with Rules and Formatter Extension is and what it does.
→ Features and Benefits (Web site)
This Web site contains information about the key capabilities and benefits of WebSphere Message Broker and also provides links to announcement letter describes what is new in WebSphere Message Broker.
→ WebSphere Message Broker FAQs (Web site)
This Web site gives answers to some frequently asked questions about WebSphere Message Broker.
→ WebSphere Message Broker Quick Tour (Web site)
Take a quick tour of the basic concepts and features of WebSphere Message Broker.
→ WebSphere Message Broker information center (Information center)
WebSphere Message Broker documentation.
→ IBM developerWorks WebSphere Message Broker zone (Web site)
This Web site is a collection of resources for WebSphere Message Broker includ support.

Application Integration

→ Application Integration (Web site)
This Web site outlines what application integration is and how you can use it to i
→ IBM developerWorks WebSphere Application Connectivity zone (Web site)
This Web site is a collection of resources for users of WebSphere Application Integration. It includes links to papers, education, downloads, and support.
→ WebSphere software (Web site)

Web access to documents, forums, downloads and support

Figure 10-28. WebSphere Message Broker information roadmap WM643 / VM6431.0

Notes:

This figure shows an example of the WebSphere Message Broker information roadmap available on DeveloperWorks. It provides access to documents, forums, downloads, and support.

The link for WebSphere Message Broker DeveloperWorks page is `http://www.ibm.com/developerworks/websphere/zones/businessintegration/wmb.html`

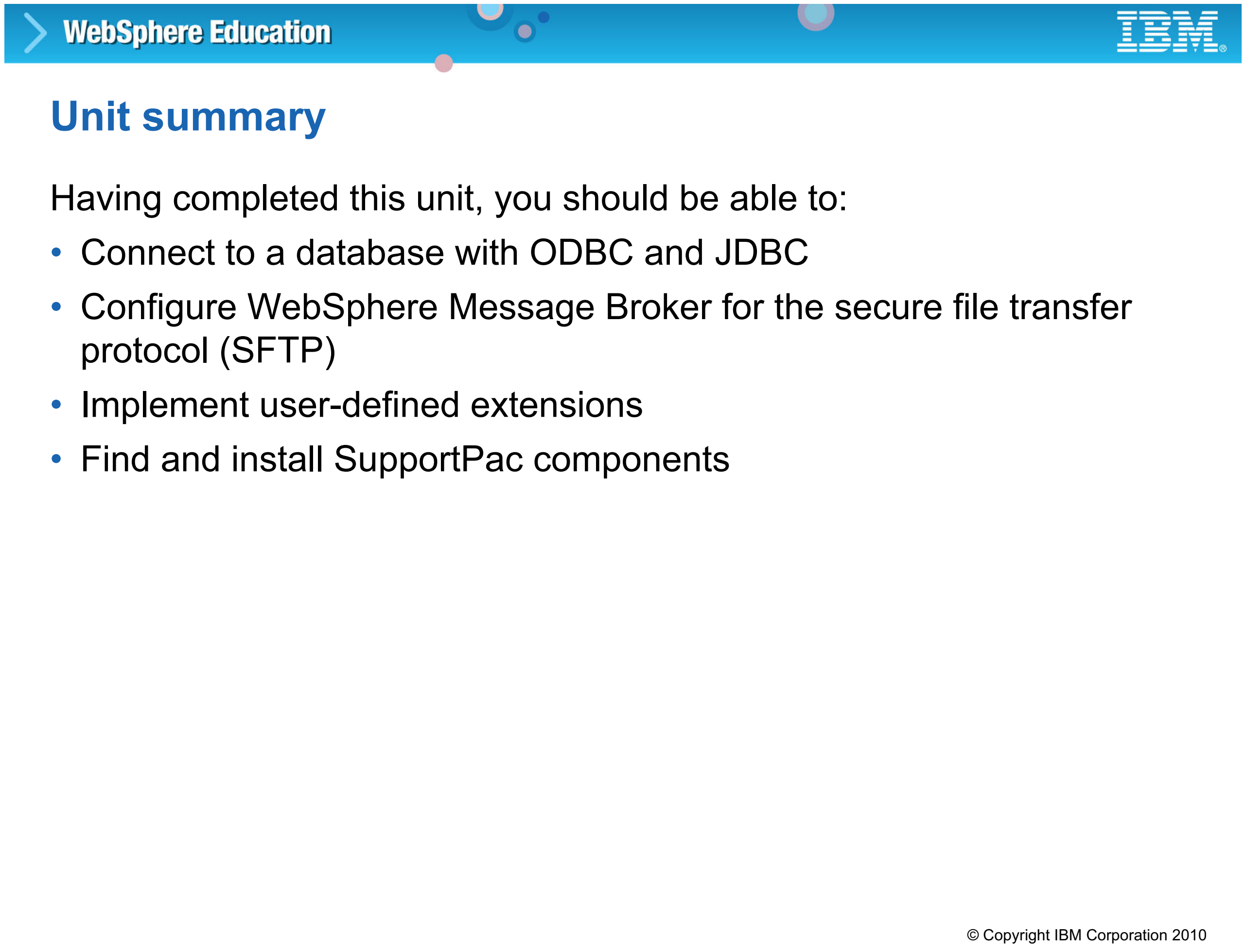
Instructor notes:

Purpose — Tell students about the many online resources available for WebSphere Message Broker support and extensions.

Details — None.

Additional information — None.

Transition statement — Summarize the unit.



WebSphere Education IBM

Unit summary

Having completed this unit, you should be able to:

- Connect to a database with ODBC and JDBC
- Configure WebSphere Message Broker for the secure file transfer protocol (SFTP)
- Implement user-defined extensions
- Find and install SupportPac components

Figure 10-29. Unit summary WM643 / VM6431.0

Notes:

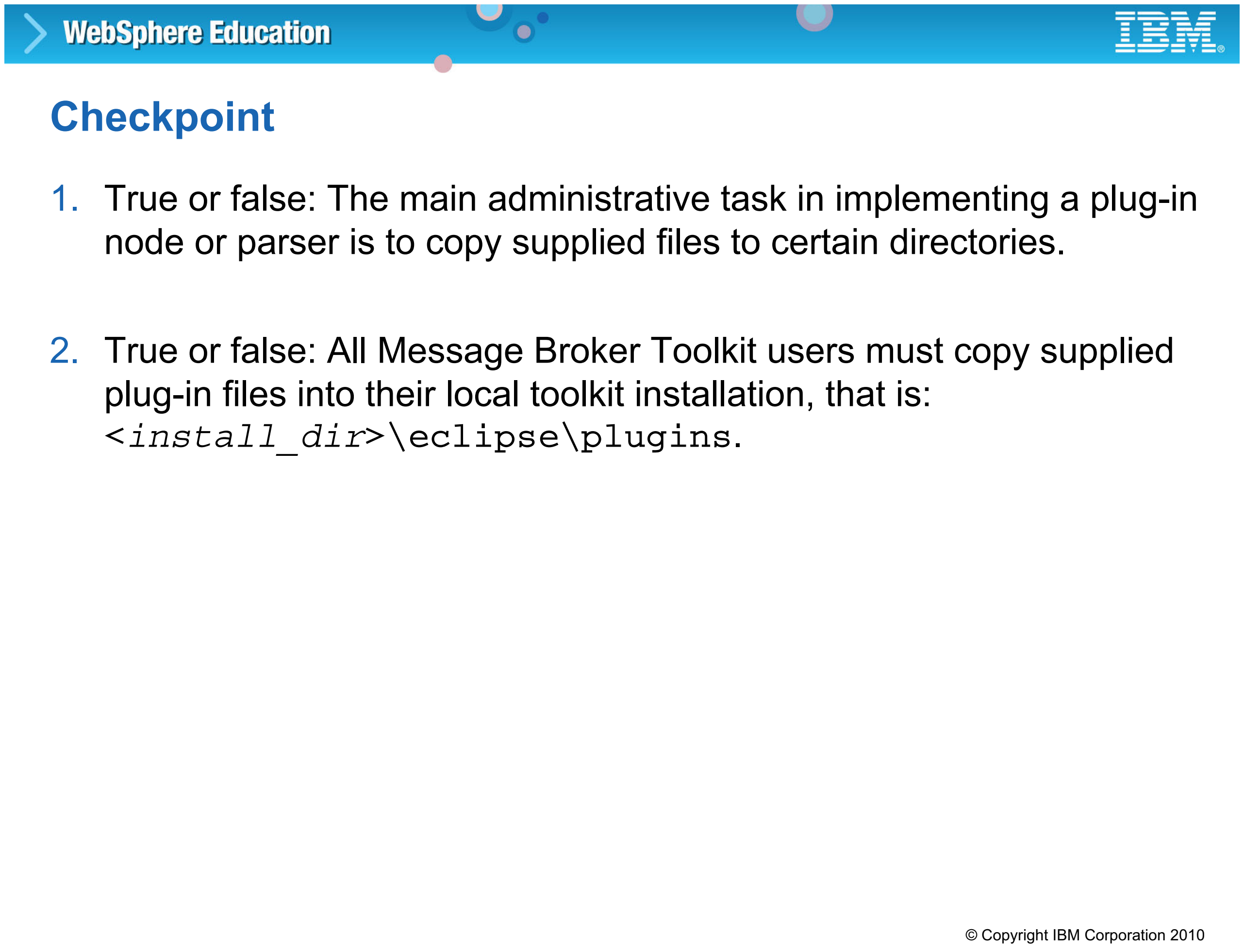
Instructor notes:

Purpose — Close the unit (and the course) by summarizing some highlights.

Details — None.

Additional information — None.

Transition statement — Next are some checkpoint questions to test your knowledge.



WebSphere Education

Checkpoint

1. True or false: The main administrative task in implementing a plug-in node or parser is to copy supplied files to certain directories.

2. True or false: All Message Broker Toolkit users must copy supplied plug-in files into their local toolkit installation, that is: `<install_dir>\eclipse\plugins`.

Figure 10-30. Checkpoint WM643 / VM6431.0

Notes:

Write your answers here:

1.

2.

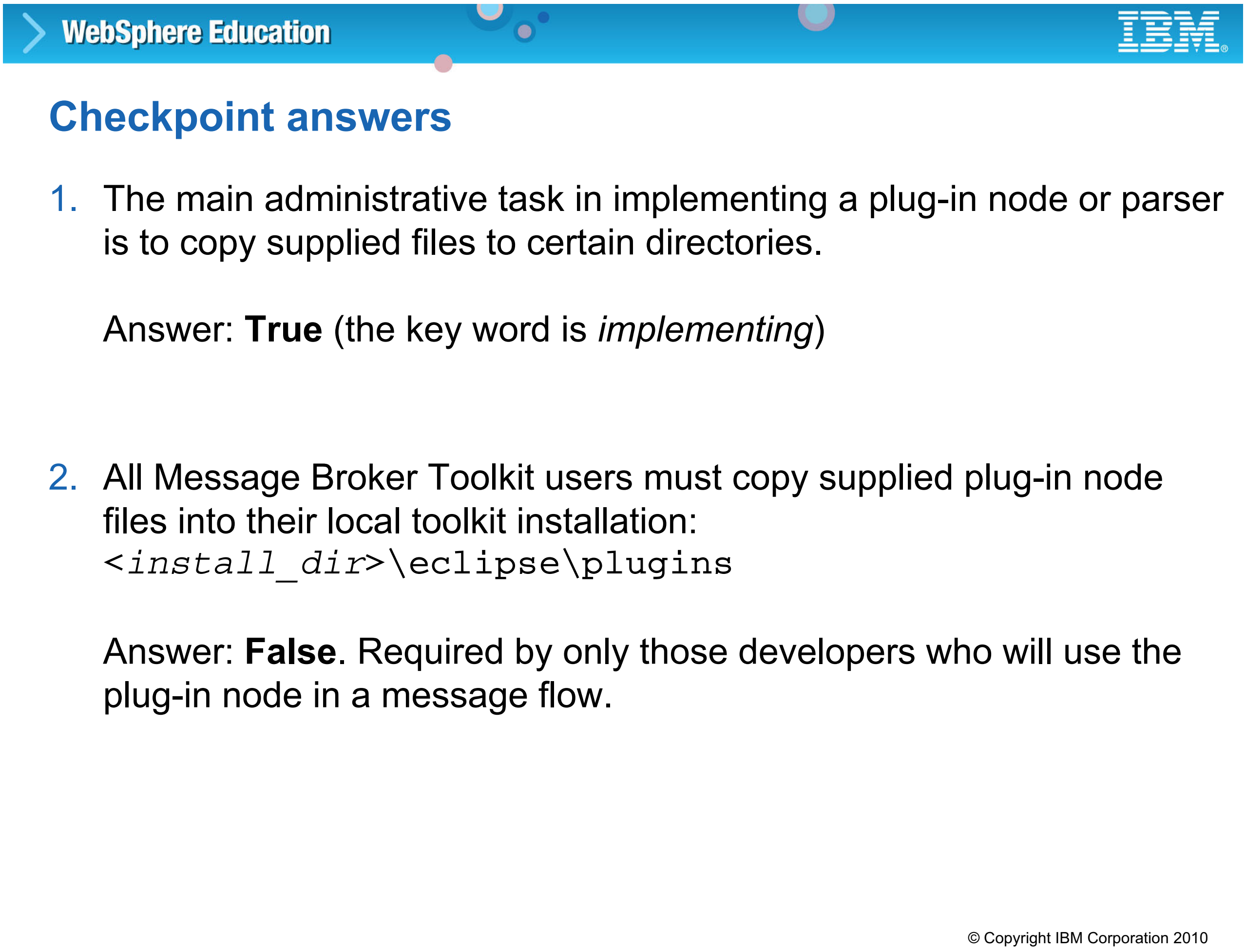
Instructor notes:

Purpose — Review this topic with a couple of questions.

Details —

Additional information — Discuss with students before presenting the answers.

Transition statement — The answers, in case you have missed something during the discussion.



WebSphere Education
IBM

Checkpoint answers

1. The main administrative task in implementing a plug-in node or parser is to copy supplied files to certain directories.

   Answer: **True** (the key word is *implementing*)

2. All Message Broker Toolkit users must copy supplied plug-in node files into their local toolkit installation: `<install_dir>\eclipse\plugins`

   Answer: **False**. Required by only those developers who will use the plug-in node in a message flow.

Figure 10-31. Checkpoint answers WM643 / VM6431.0

Notes:

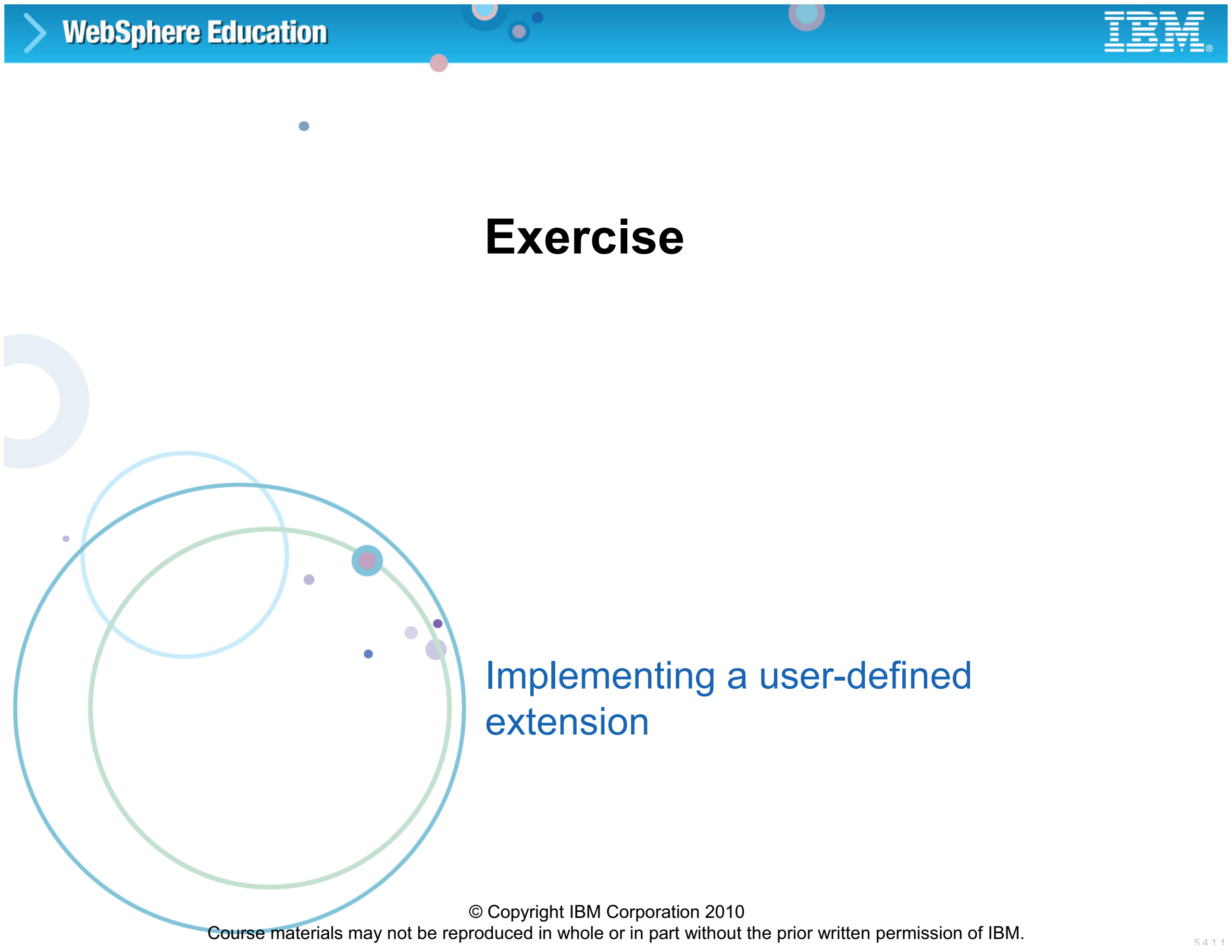
Instructor notes:

Purpose — Review checkpoint answers.

Details — None.

Additional information — None.

Transition statement — Next is a lab exercise.



WebSphere Education

IBM

Exercise

Implementing a user-defined extension

5411

Figure 10-32. Exercise 9

WM643 / VM6431.0

Notes:

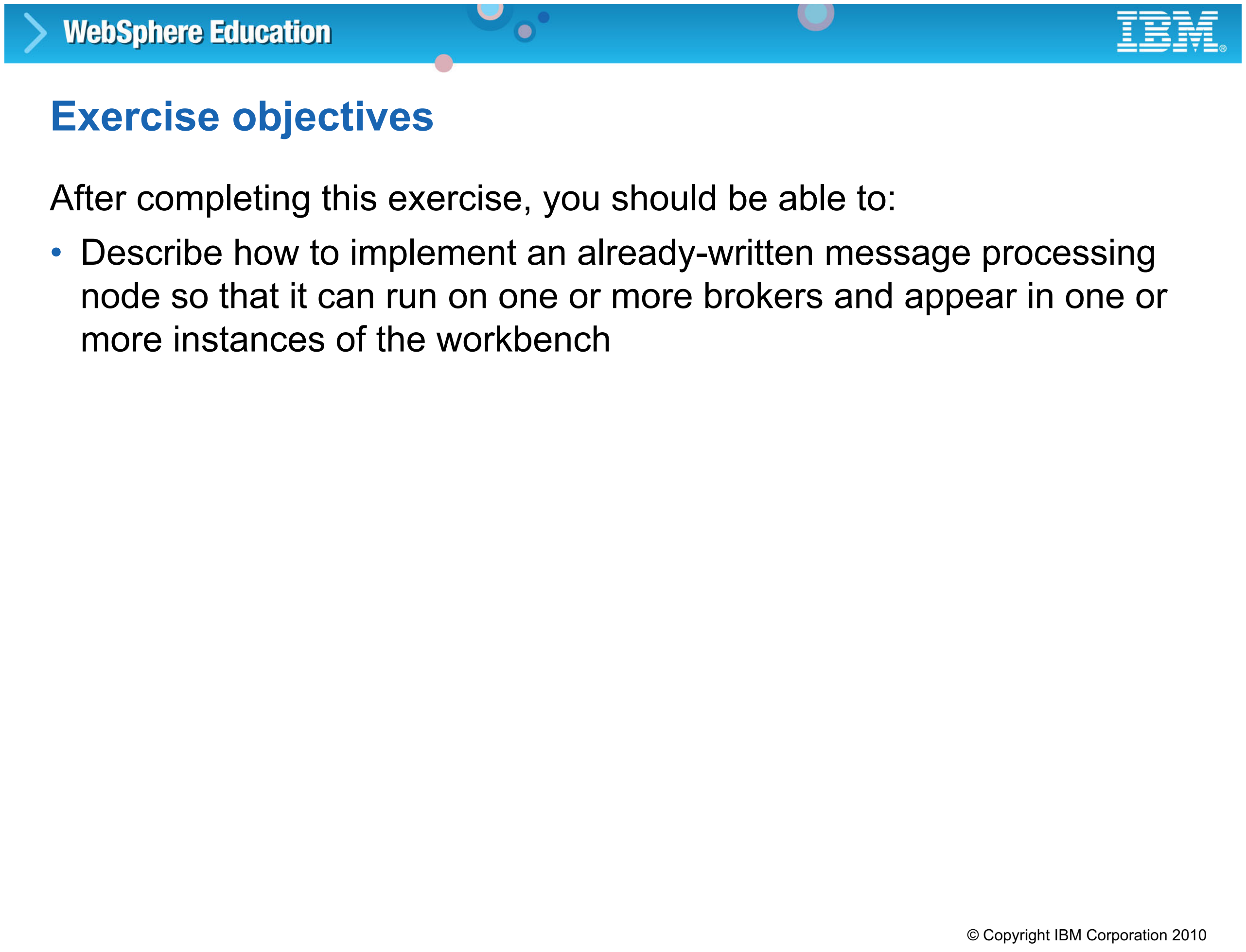
Instructor notes:

Purpose — Introduce the lab.

Details — None.

Additional information — None.

Transition statement — None.



WebSphere Education

Exercise objectives

After completing this exercise, you should be able to:

- Describe how to implement an already-written message processing node so that it can run on one or more brokers and appear in one or more instances of the workbench

Figure 10-33. Exercise objectives

WM643 / VM6431.0

Notes:

Proceed to the *Student Exercise Guide* and complete Exercise 9.

Instructor notes:

Purpose — State exercise objectives.

Details — Have students proceed to Exercise 9 in the Lab Exercise Guide.

Additional information — None.

Transition statement — None.

Unit 11. Course summary

Estimated time

00:30 lecture

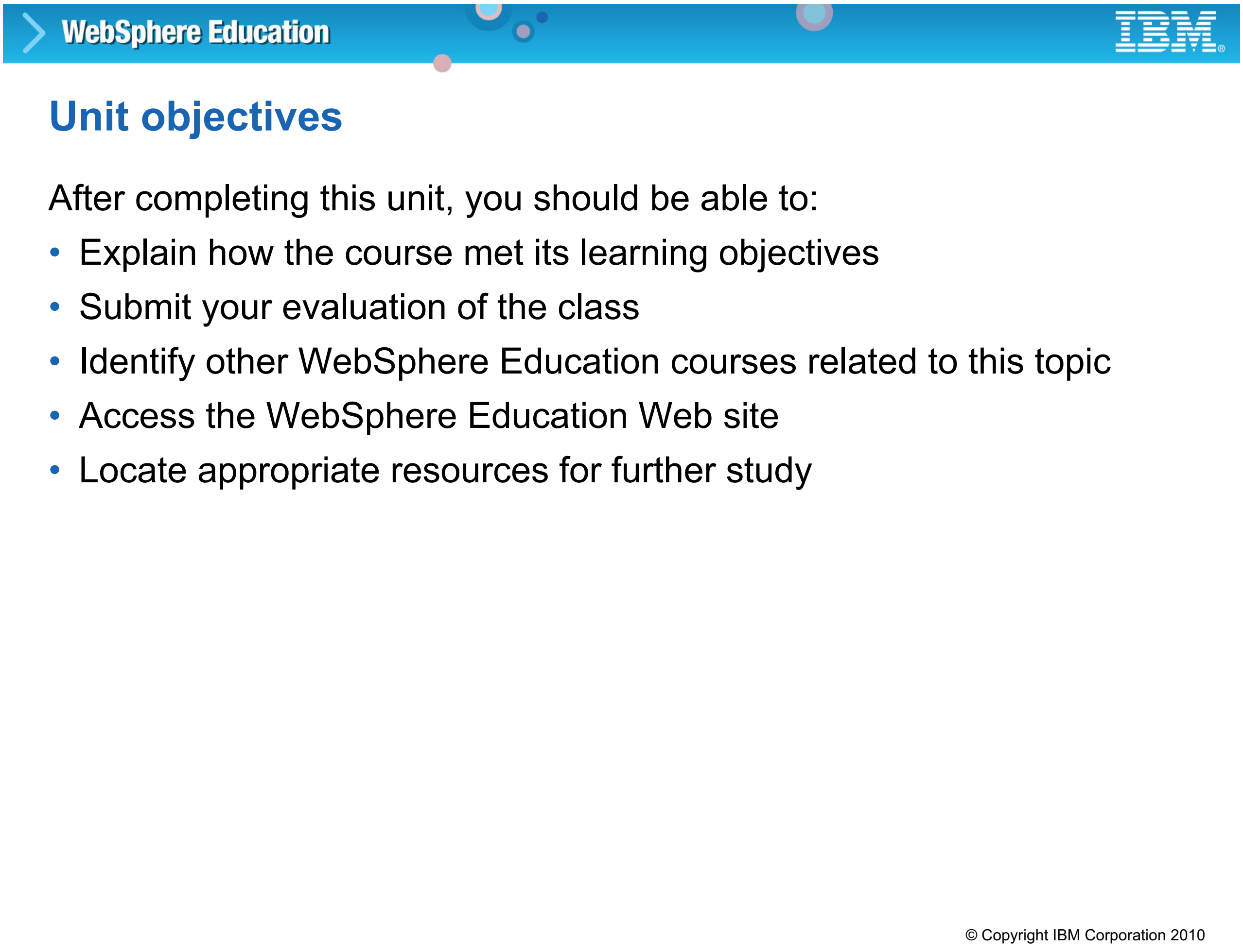
What this unit is about

This unit summarizes the course.

What you should be able to do

After completing this unit, you should be able to:

- Explain how the course met its learning objectives
- Submit your evaluation of the class
- Identify other WebSphere Education courses related to this topic
- Access the WebSphere Education website
- Locate appropriate resources for further study



WebSphere Education

Unit objectives

After completing this unit, you should be able to:

- Explain how the course met its learning objectives
- Submit your evaluation of the class
- Identify other WebSphere Education courses related to this topic
- Access the WebSphere Education Web site
- Locate appropriate resources for further study

Figure 11-1. Unit objectives WM643 / VM6431.0

Notes:

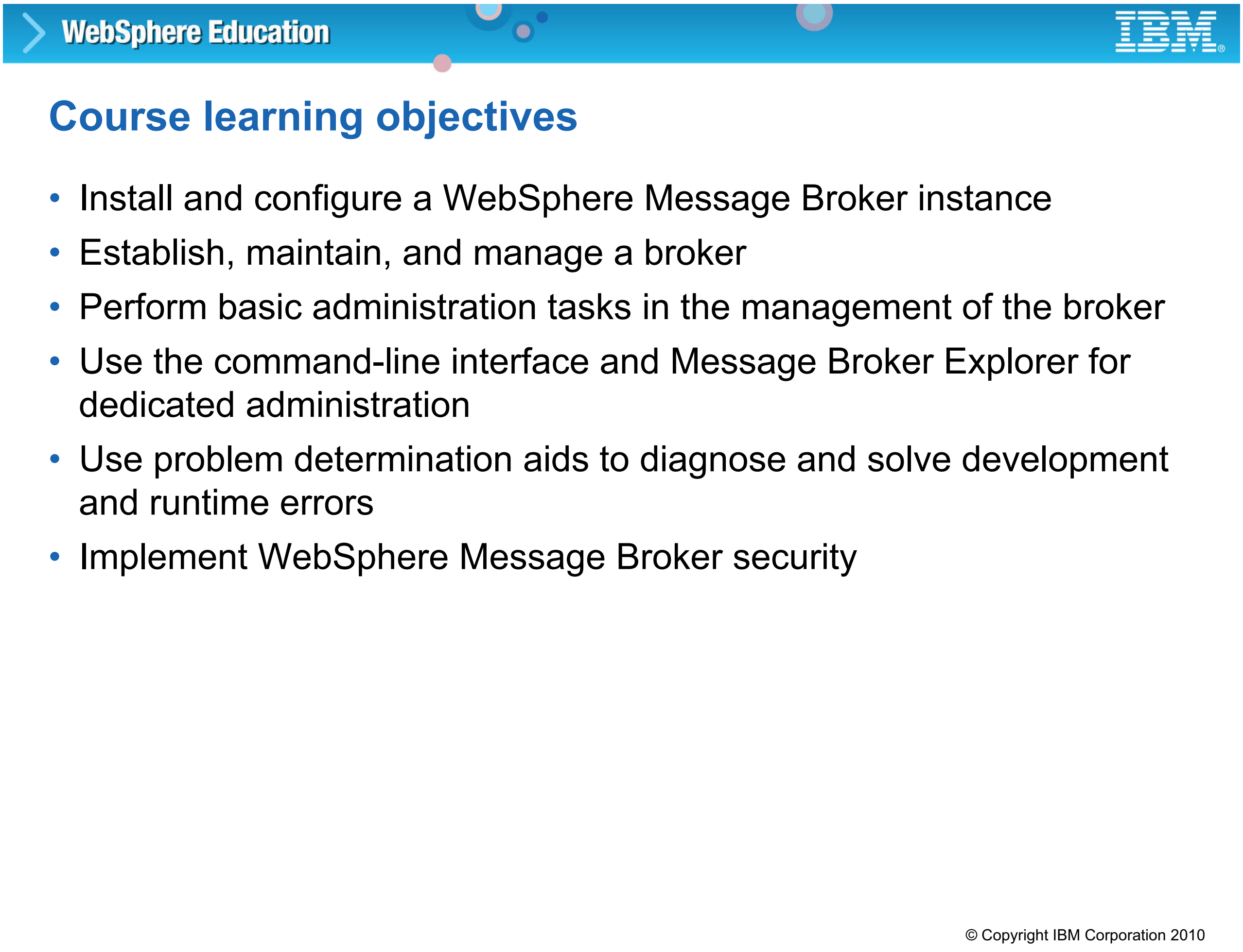
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —



WebSphere Education

Course learning objectives

- Install and configure a WebSphere Message Broker instance
- Establish, maintain, and manage a broker
- Perform basic administration tasks in the management of the broker
- Use the command-line interface and Message Broker Explorer for dedicated administration
- Use problem determination aids to diagnose and solve development and runtime errors
- Implement WebSphere Message Broker security

Figure 11-2. Course learning objectives WM643 / VM6431.0

Notes:

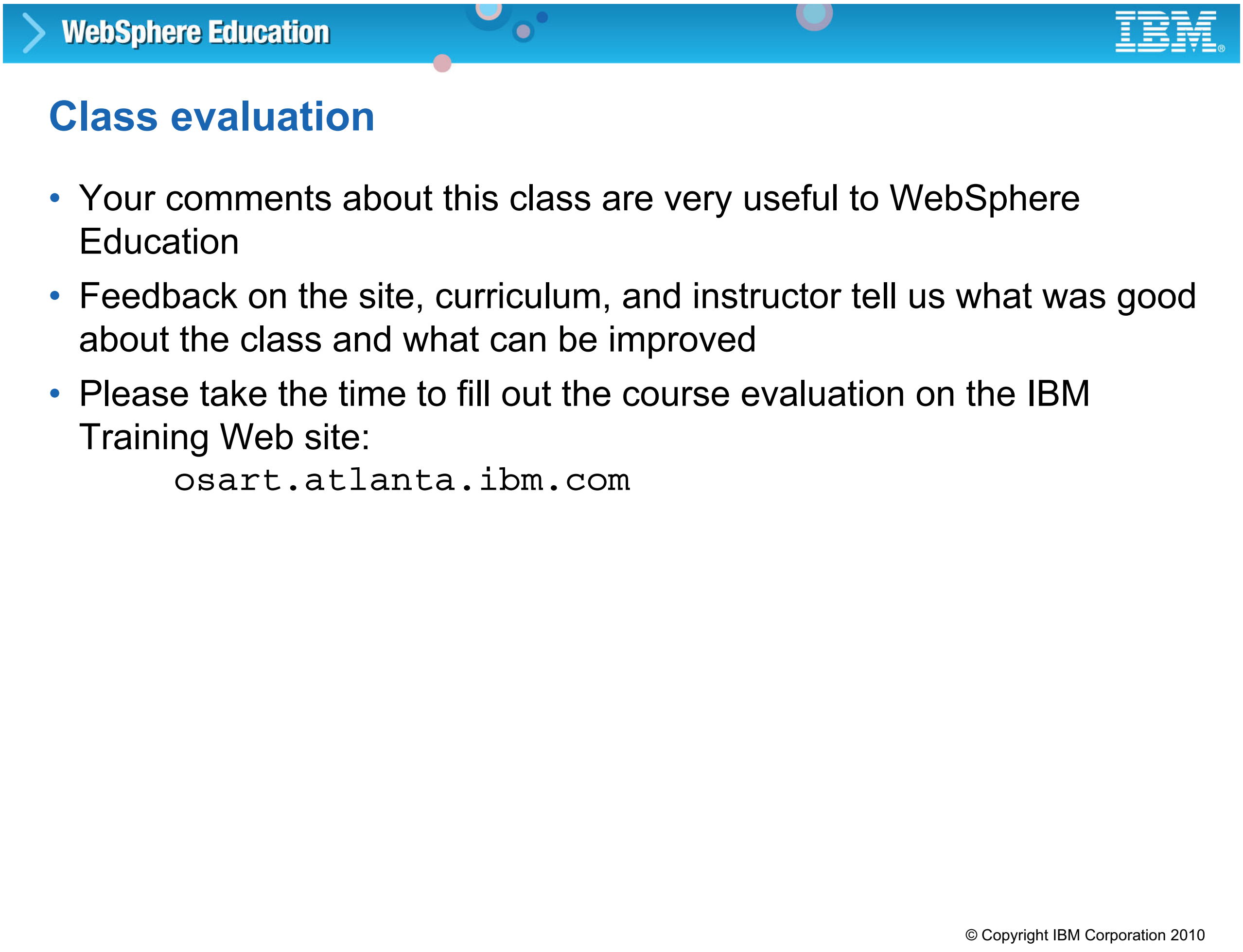
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —


WebSphere Education
IBM

Class evaluation

- Your comments about this class are very useful to WebSphere Education
- Feedback on the site, curriculum, and instructor tell us what was good about the class and what can be improved
- Please take the time to fill out the course evaluation on the IBM Training Web site:

  `osart.atlanta.ibm.com`

Figure 11-3. Class evaluation WM643 / VM6431.0

Notes:

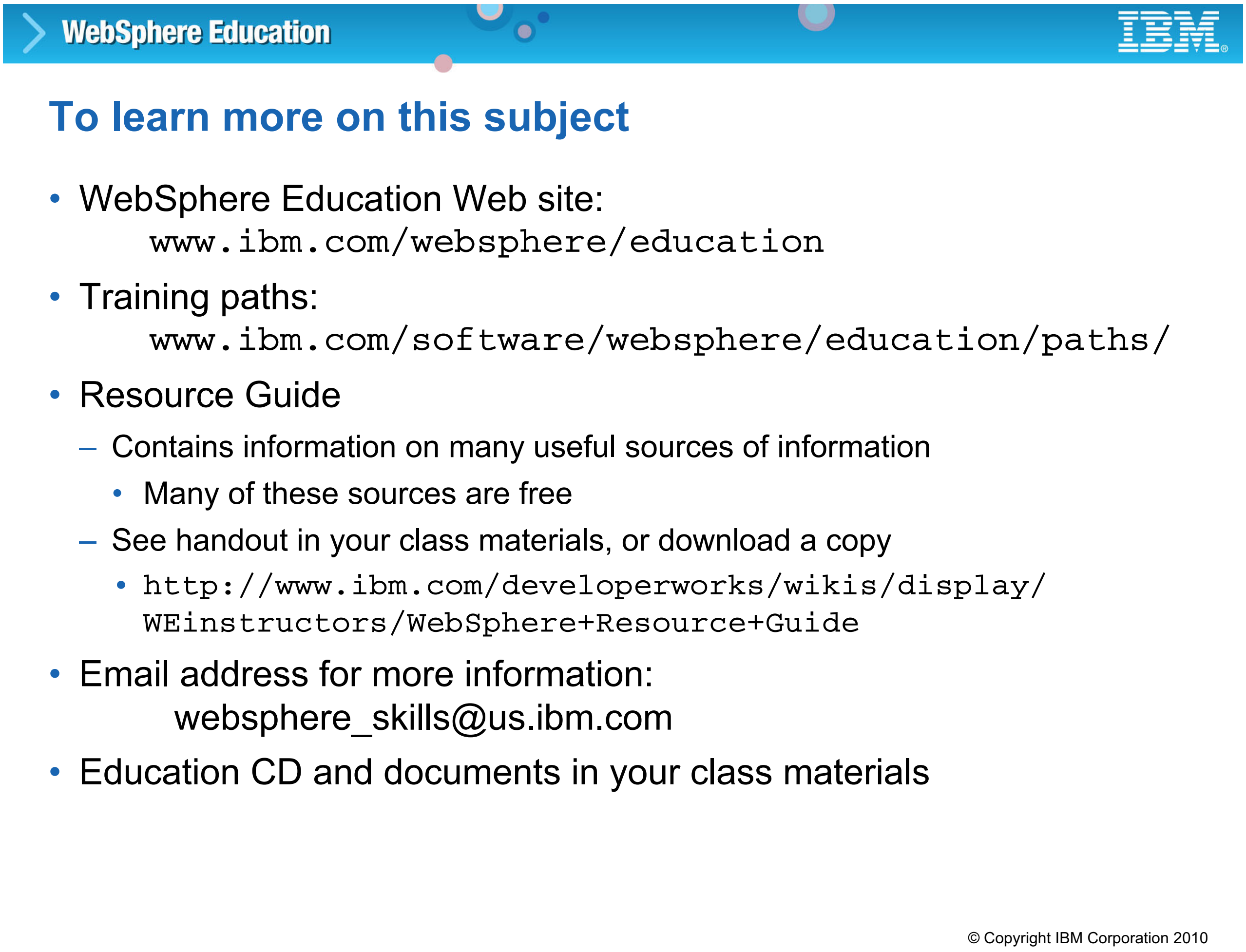
Instructor notes:

Purpose — Review the course objectives.

Details — You should give a brief summary of each objective and the topics and exercises in the course that were included to ensure the objectives were met.

Additional information —

Transition statement —



WebSphere Education

To learn more on this subject

- WebSphere Education Web site:
  `www.ibm.com/websphere/education`
- Training paths:
  `www.ibm.com/software/websphere/education/paths/`
- Resource Guide
  – Contains information on many useful sources of information
    - Many of these sources are free

  – See handout in your class materials, or download a copy
    - `http://www.ibm.com/developerworks/wikis/display/WEinstructors/WebSphere+Resource+Guide`
- Email address for more information:
  websphere_skills@us.ibm.com
- Education CD and documents in your class materials

Figure 11-4. To learn more on this subject WM643 / VM6431.0

Notes:

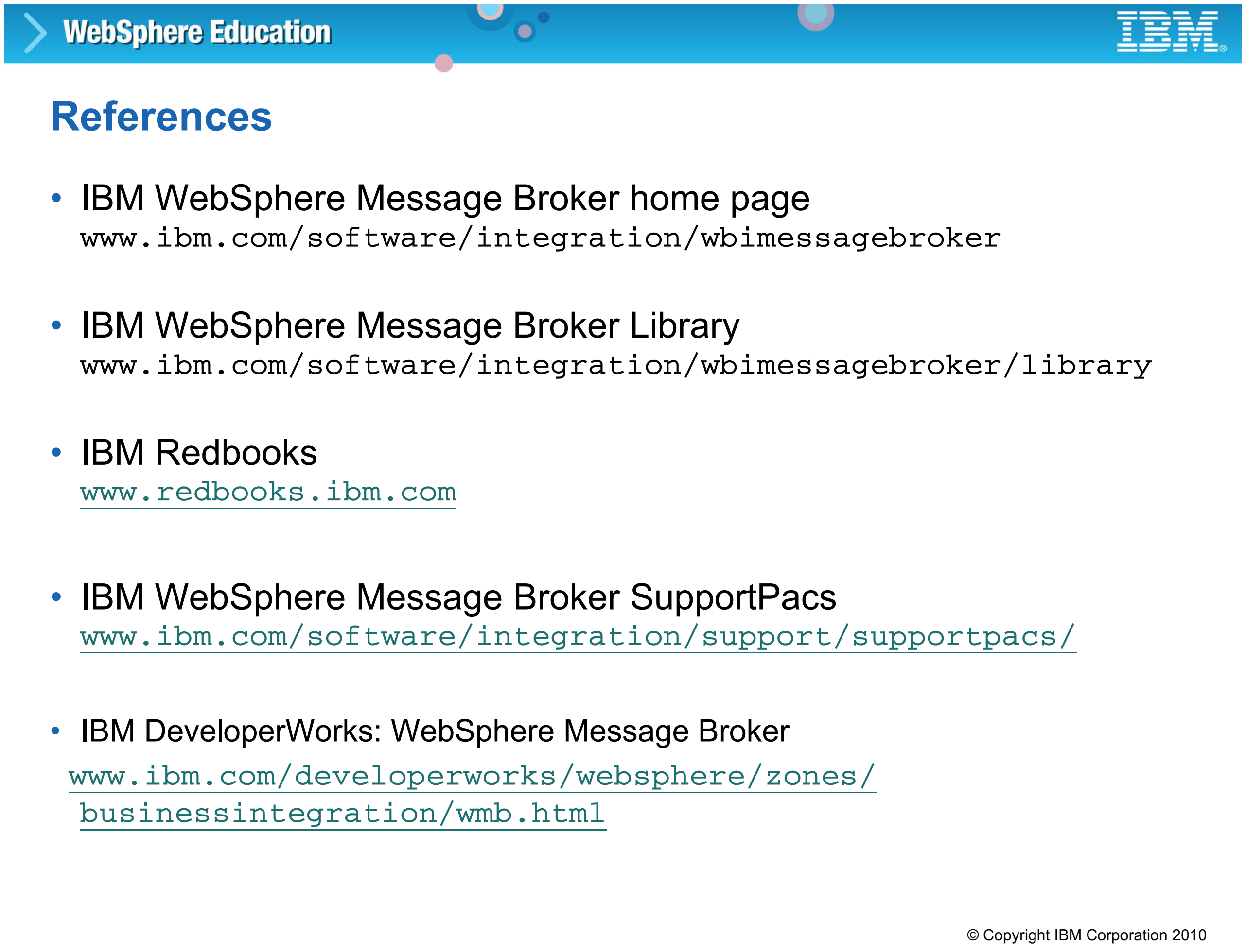
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —


WebSphere Education
IBM

References

- IBM WebSphere Message Broker home page
  www.ibm.com/software/integration/wbimessagebroker
- IBM WebSphere Message Broker Library
  www.ibm.com/software/integration/wbimessagebroker/library
- IBM Redbooks
  www.redbooks.ibm.com
- IBM WebSphere Message Broker SupportPacs
  www.ibm.com/software/integration/support/supportpacs/
- IBM DeveloperWorks: WebSphere Message Broker
  www.ibm.com/developerworks/websphere/zones/businessintegration/wmb.html

Figure 11-5. References WM643 / VM6431.0

Notes:

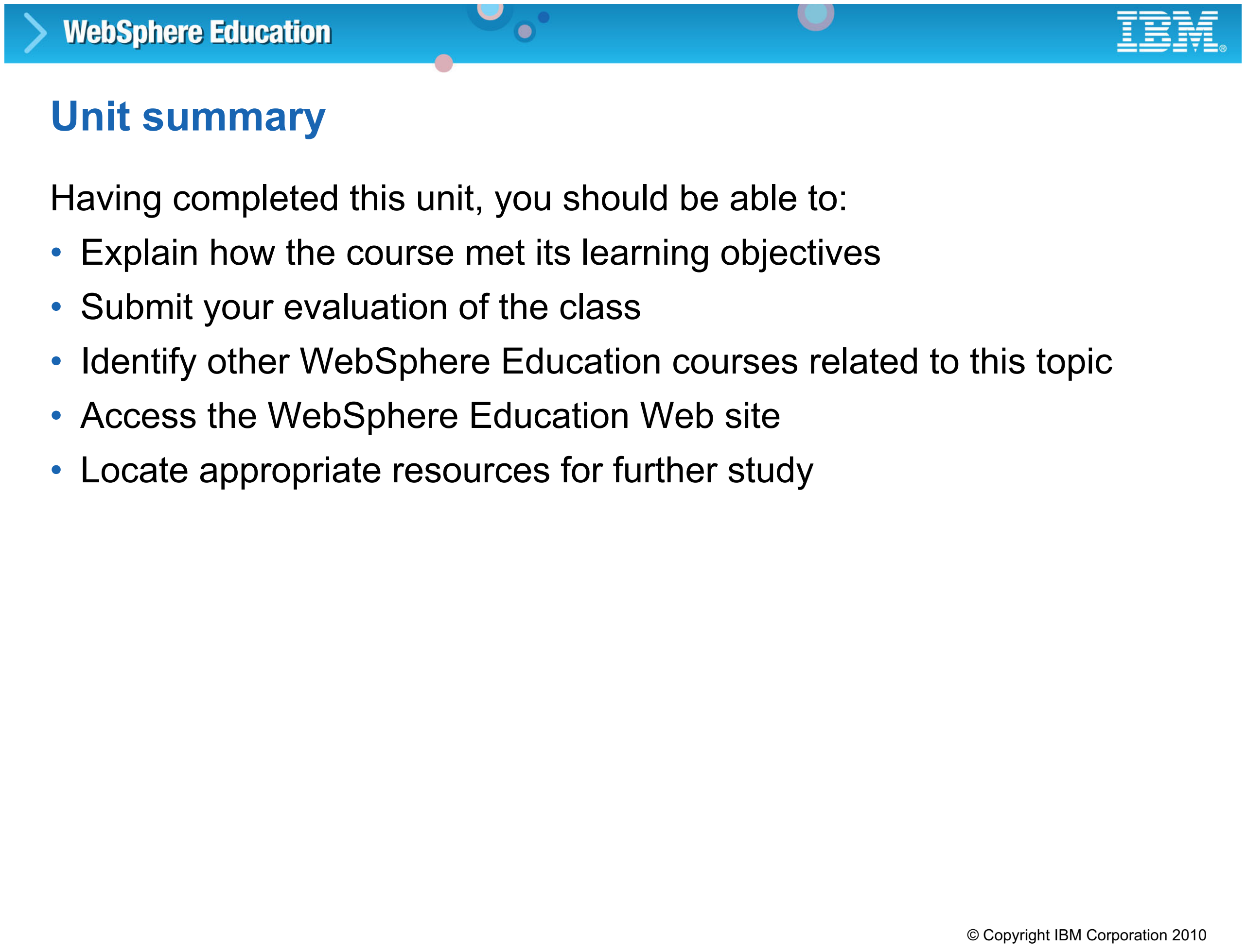
Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —



WebSphere Education

Unit summary

Having completed this unit, you should be able to:

- Explain how the course met its learning objectives
- Submit your evaluation of the class
- Identify other WebSphere Education courses related to this topic
- Access the WebSphere Education Web site
- Locate appropriate resources for further study

Figure 11-6. Unit summary WM643 / VM6431.0

Notes:

Instructor notes:

Purpose —

Details —

Additional information —

Transition statement —

Appendix A. List of abbreviations

ESB	Enterprise service bus
API	Application programming interface
BAR	Broker archive
CORBA	Common object request broker architecture
CMP	Configuration manager proxy
DNS	Domain name service
DSN	Data source name
DTD	Document type definition
EAI	Enterprise application integration
EIS	Enterprise integration system
ESQL	Extended Structured Query Language
FTP	File transfer protocol
GUI	Graphical user interface
IIOP	Internet Inter-Orb protocol
IDE	Integrated development environment
HTTP	Hypertext transport protocol
JAR	Java archive
JMS	Java message service
JNDI	Java naming and directory interface
JVM	Java virtual machine
LDAP	Lightweight directory access protocol
LOB	Line-of-business
LUW	Logical unit of work
OAM	Object authority manager
ODBC	Open database connectivity
QOS	Quality of service
RDB	Relational database
RPC	Remote procedure call
SCA	Service component architecture
SCDL	Service component definition language

SFTP	Secure FTP
SMTP	Simple mail transport protocol
SOA	Service-oriented architecture
SOAP	SOAP originally stood for Simple Object Access Protocol, and lately also Service Oriented Architecture Protocol, but is now simply known as SOAP. The original acronym was dropped with version 1.2 of the standard, which became a W3C Recommendation on June 24, 2003, as it was considered to be misleading.
SQL	Structured query language
SSL	Secure sockets layer
TDS	Tagged delimited string
UDDI	Universal description discovery and integration
UUID	Universally unique identifier
W3C	Web Wide Web consortium
WSDL	Web services description language
XSD	XML schema document
XML	Extensible markup language
XSD	XML schema document
XSL	Extensible stylesheet language
XSLT	XSL transformations

IBM®