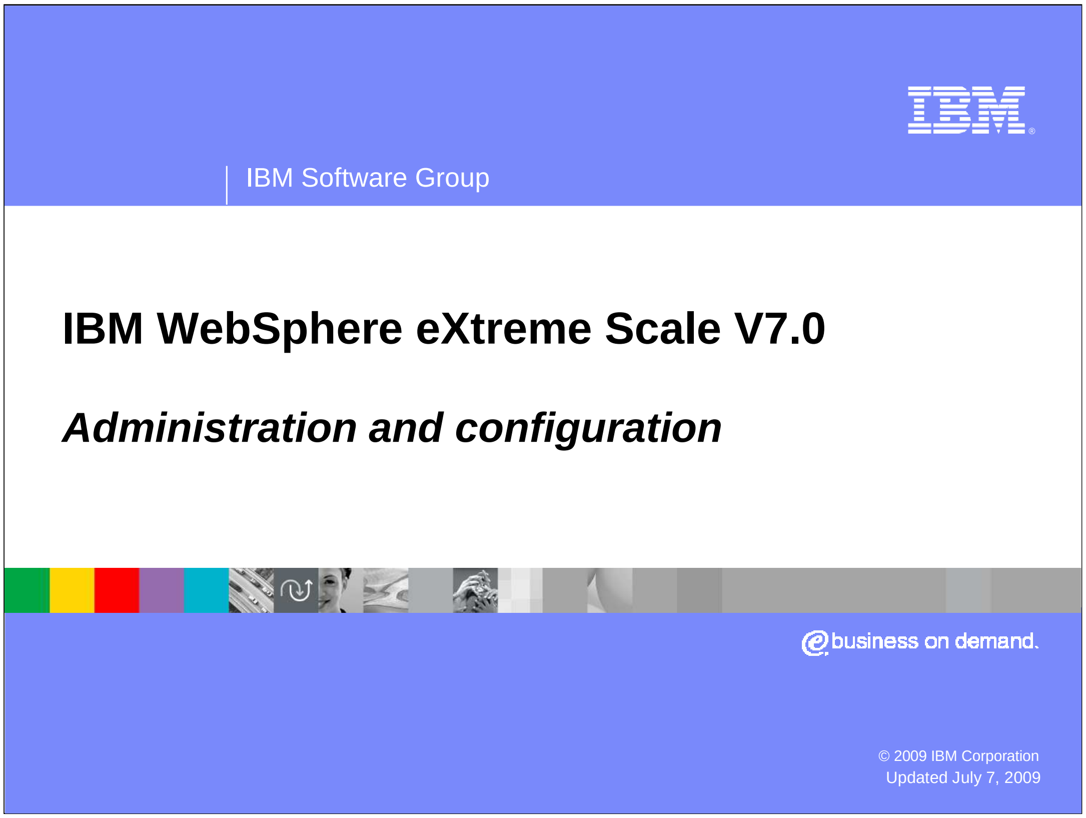


IBM Software Group

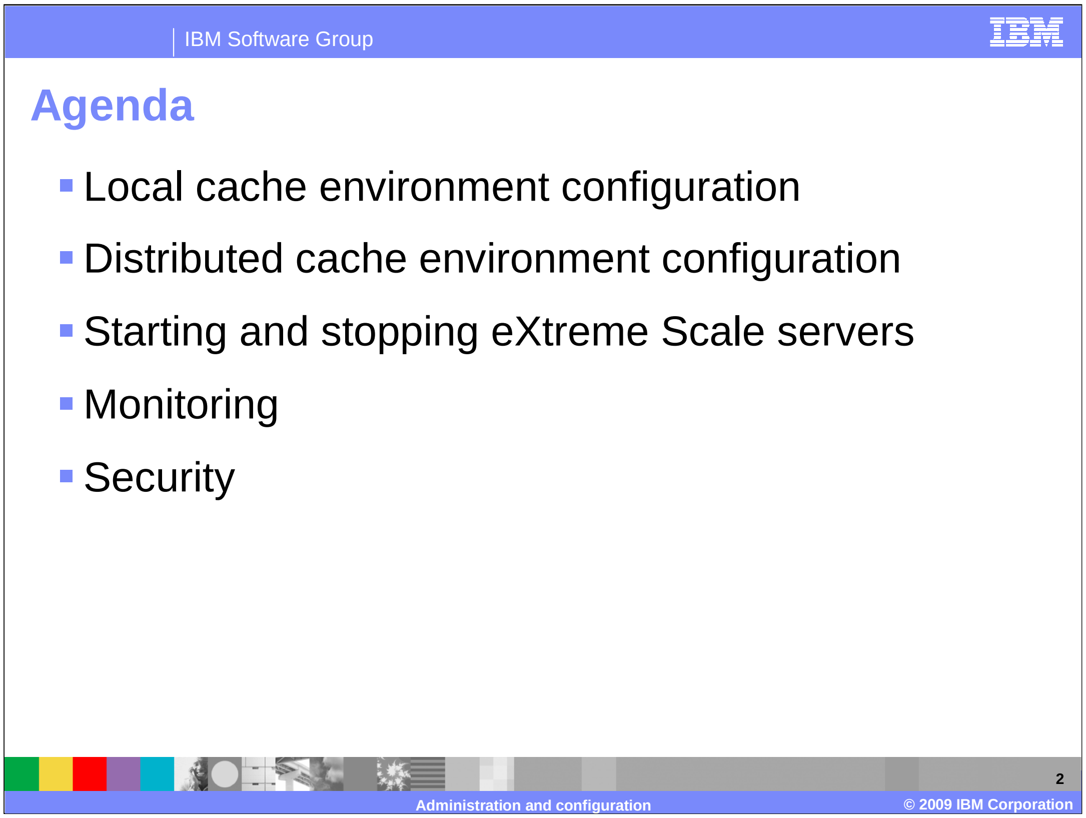
IBM WebSphere eXtreme Scale V7.0

Administration and configuration

e business on demand.

Updated July 7, 2009

This presentation will cover WebSphere® eXtreme Scale V7.0 administration and configuration.



IBM Software Group

Agenda

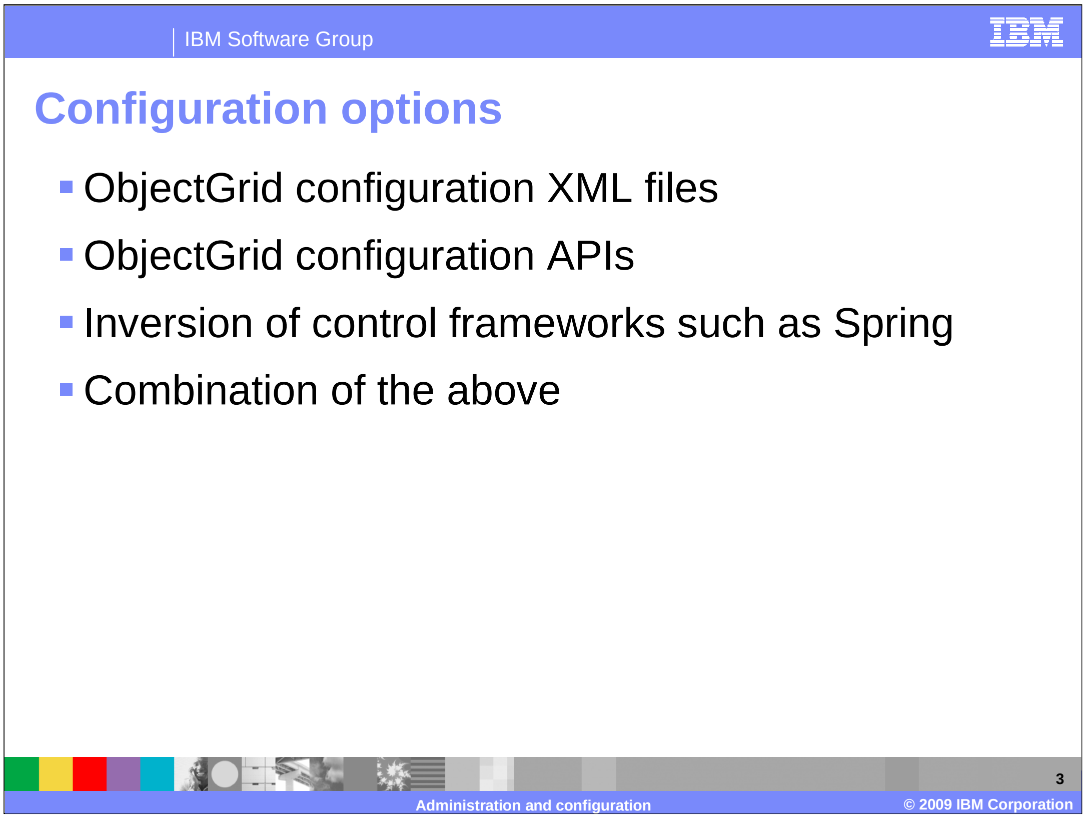
- Local cache environment configuration
- Distributed cache environment configuration
- Starting and stopping eXtreme Scale servers
- Monitoring
- Security

2

Administration and configuration

© 2009 IBM Corporation

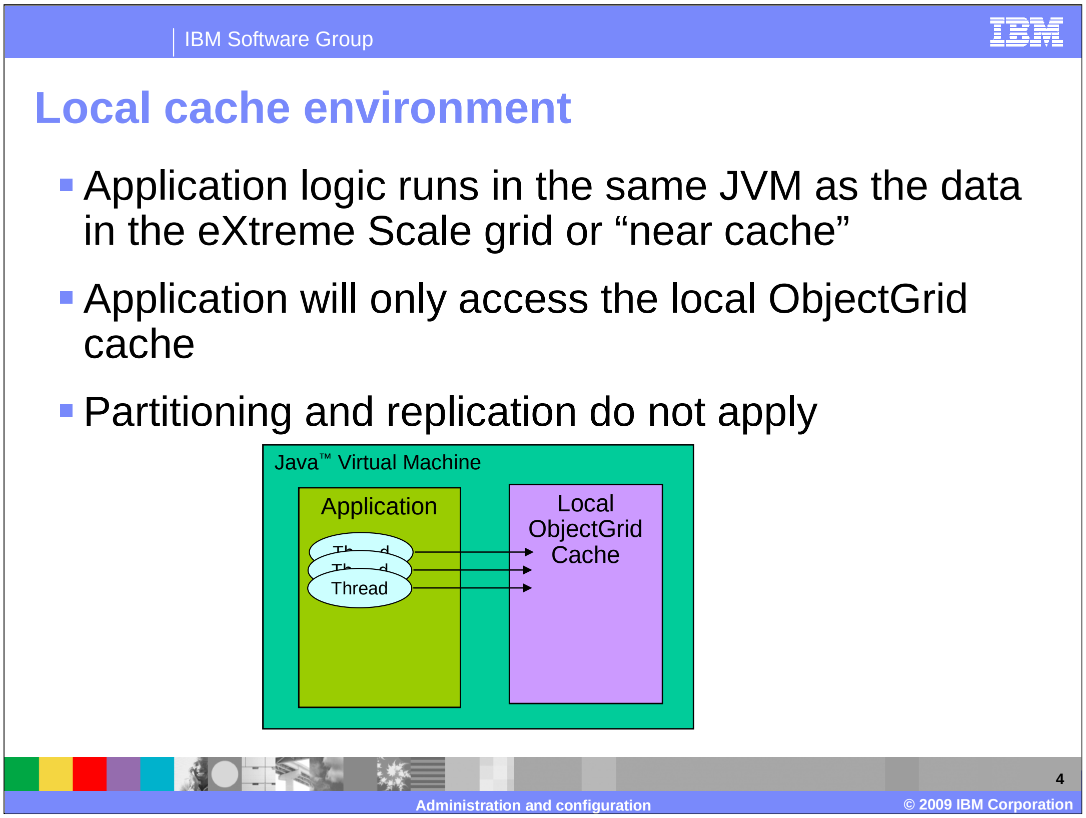
The presentation will begin with the configuration of the local cache and the distributed cache WebSphere eXtreme Scale environments. Then administration topics starting and stopping servers, monitoring and security are covered.



Configuration options

- ObjectGrid configuration XML files
- ObjectGrid configuration APIs
- Inversion of control frameworks such as Spring
- Combination of the above

To cache objects using eXtreme Scale, you must create an ObjectGrid instance within your application. The instance can be created based on configuration data stored in an XML file, configured programmatically. created and enhanced to integrate with control frameworks, or through a combination of these options.

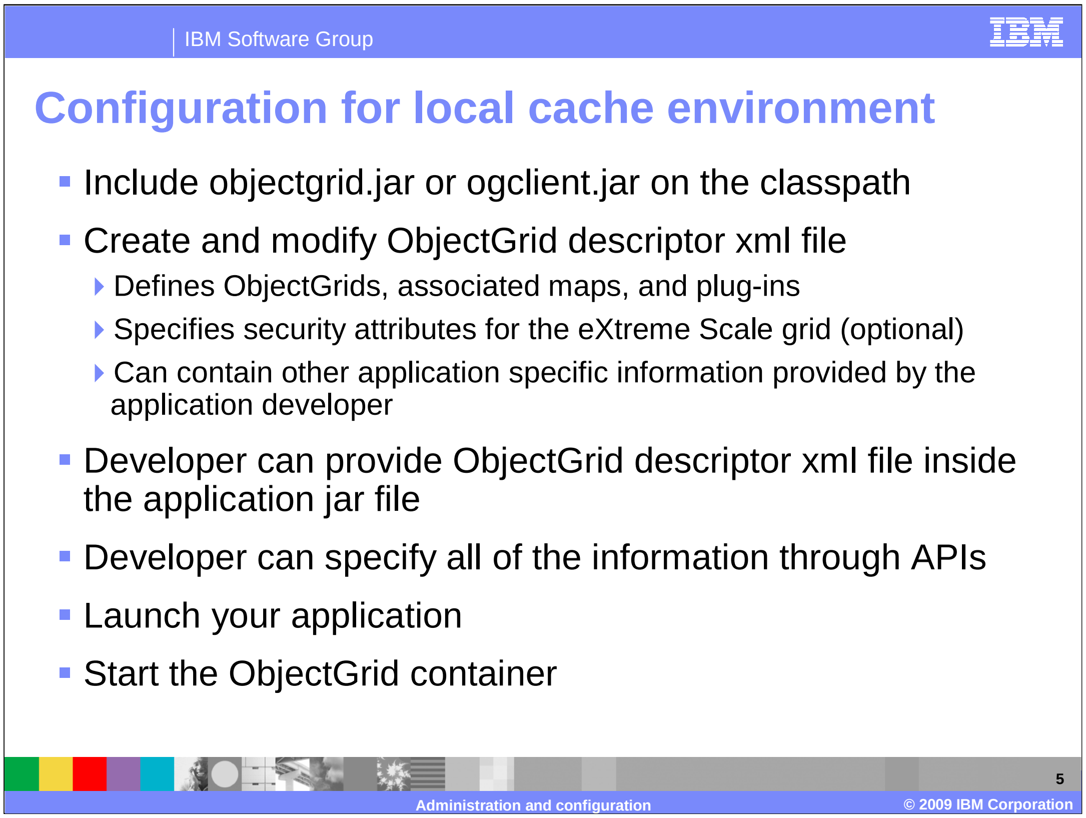


Local cache environment

- Application logic runs in the same JVM as the data in the eXtreme Scale grid or “near cache”
- Application will only access the local ObjectGrid cache
- Partitioning and replication do not apply

Java™ Virtual Machine
Application
Thread
Thread
Thread
Local ObjectGrid Cache

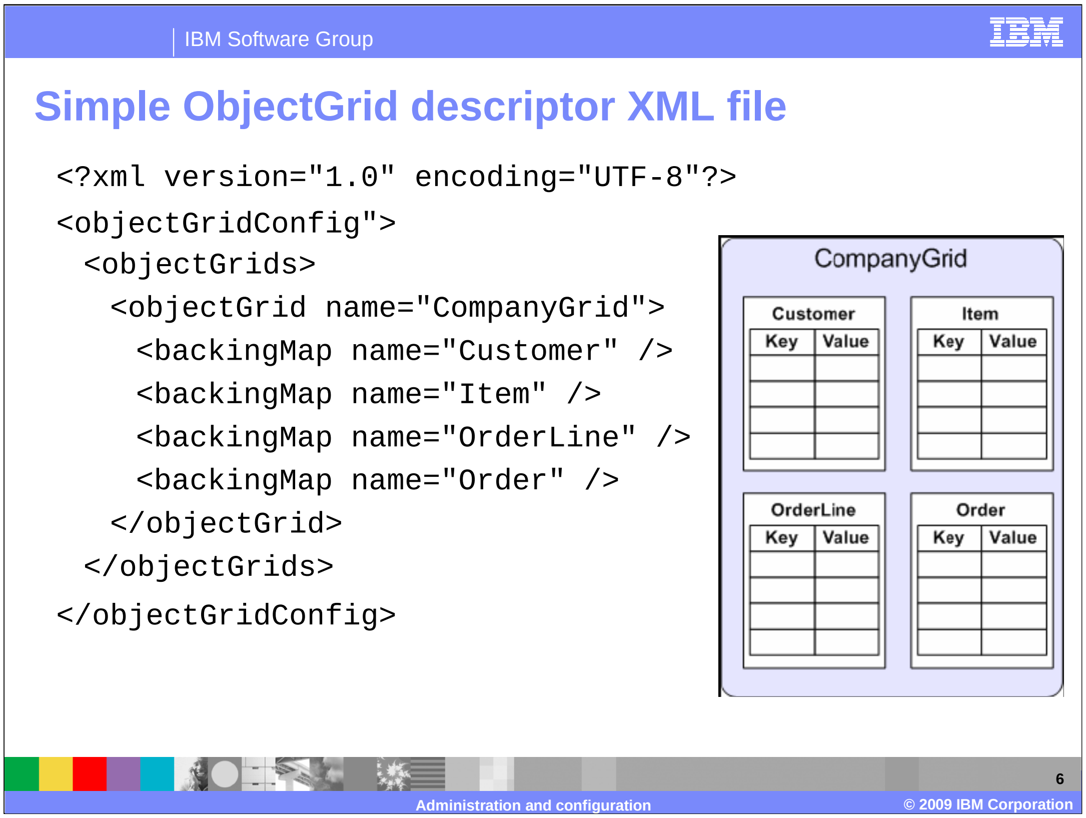
In the local cache environment, or “near-cache”, the application logic runs in the same JVM as the data in the eXtreme Scale grid. Each application will only access the local ObjectGrid instance to store or retrieve data from its cache.



Configuration for local cache environment

- Include objectgrid.jar or ogclient.jar on the classpath
- Create and modify ObjectGrid descriptor xml file
  - Defines ObjectGrids, associated maps, and plug-ins
  - Specifies security attributes for the eXtreme Scale grid (optional)
  - Can contain other application specific information provided by the application developer
- Developer can provide ObjectGrid descriptor xml file inside the application jar file
- Developer can specify all of the information through APIs
- Launch your application
- Start the ObjectGrid container

In a local cache environment, first ensure the objectgrid.jar or objectclient.jar file is including in the classpath. A local ObjectGrid can then be configured by using an ObjectGrid descriptor xml file or ObjectGrid APIs. The configuration elements include the definition of the ObjectGrid, associated maps that are used with the application, many customizable plug-ins, and security attributes. Finally, launch your application and grid container.

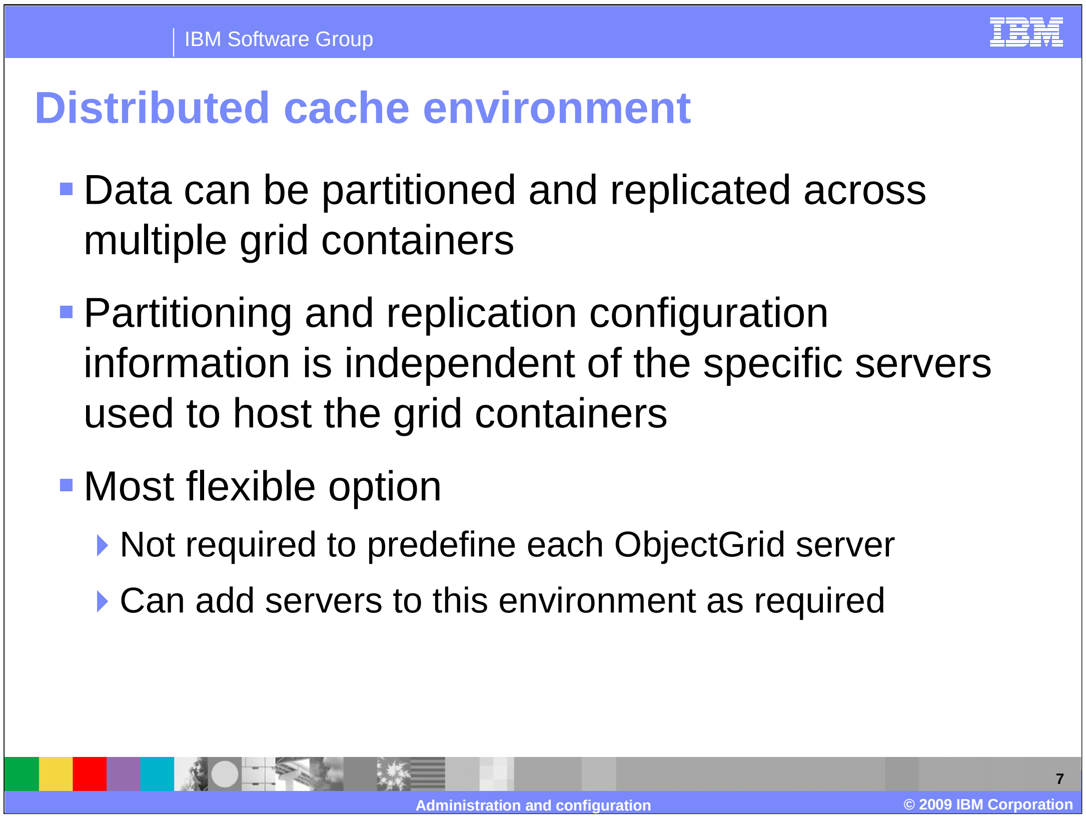


Simple ObjectGrid descriptor XML file

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig">
  <objectGrids>
    <objectGrid name="CompanyGrid">
      <backingMap name="Customer" />
      <backingMap name="Item" />
      <backingMap name="OrderLine" />
      <backingMap name="Order" />
    </objectGrid>
  </objectGrids>
</objectGridConfig>
```

CompanyGrid
Customer
Key
Value
Item
Key
Value
OrderLine
Key
Value
Order
Key
Value

This simple ObjectGrid descriptor XML file begins with the required header for each ObjectGrid XML file. It then defines the CompanyGrid ObjectGrid with Customer, Item, OrderLine, and Order BackingMaps.



IBM Software Group

Distributed cache environment

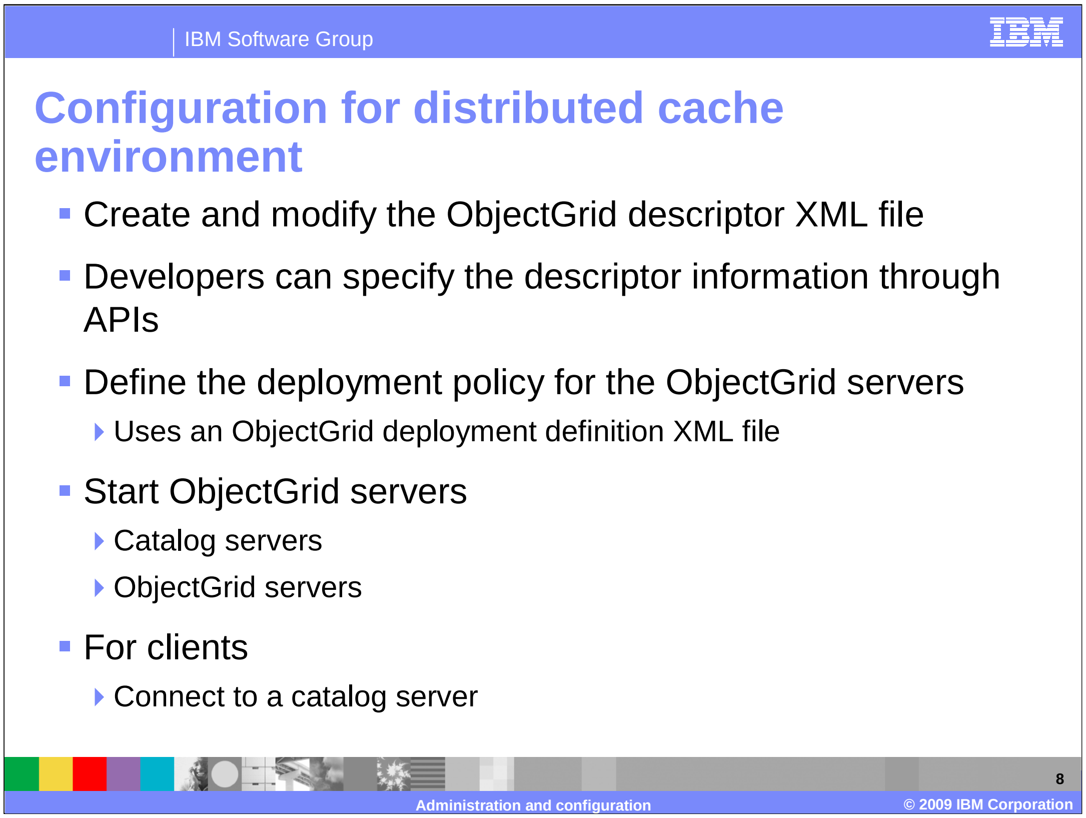
- Data can be partitioned and replicated across multiple grid containers
- Partitioning and replication configuration information is independent of the specific servers used to host the grid containers
- Most flexible option
  - Not required to predefine each ObjectGrid server
  - Can add servers to this environment as required

7

Administration and configuration

© 2009 IBM Corporation

In a distributed cache environment, the application logic can run on applications servers separate from the grid servers. In that case, the application servers host a grid client which can communicate with the grid servers to access data from the eXtreme Scale grid. The data stored in the grid is spread across all the JVMs that have WebSphere eXtreme Scale installed and configured. The data in the eXtreme Scale grid can be partitioned and replicated across multiple grid containers to ensure fault tolerance and high availability. Since all the connected grid containers act as a single entity, more JVMs can be added dynamically to increase the storage space available in the grid.



IBM Software Group

Configuration for distributed cache environment

- Create and modify the ObjectGrid descriptor XML file
- Developers can specify the descriptor information through APIs
- Define the deployment policy for the ObjectGrid servers
  - Uses an ObjectGrid deployment definition XML file
- Start ObjectGrid servers
  - Catalog servers
  - ObjectGrid servers
- For clients
  - Connect to a catalog server

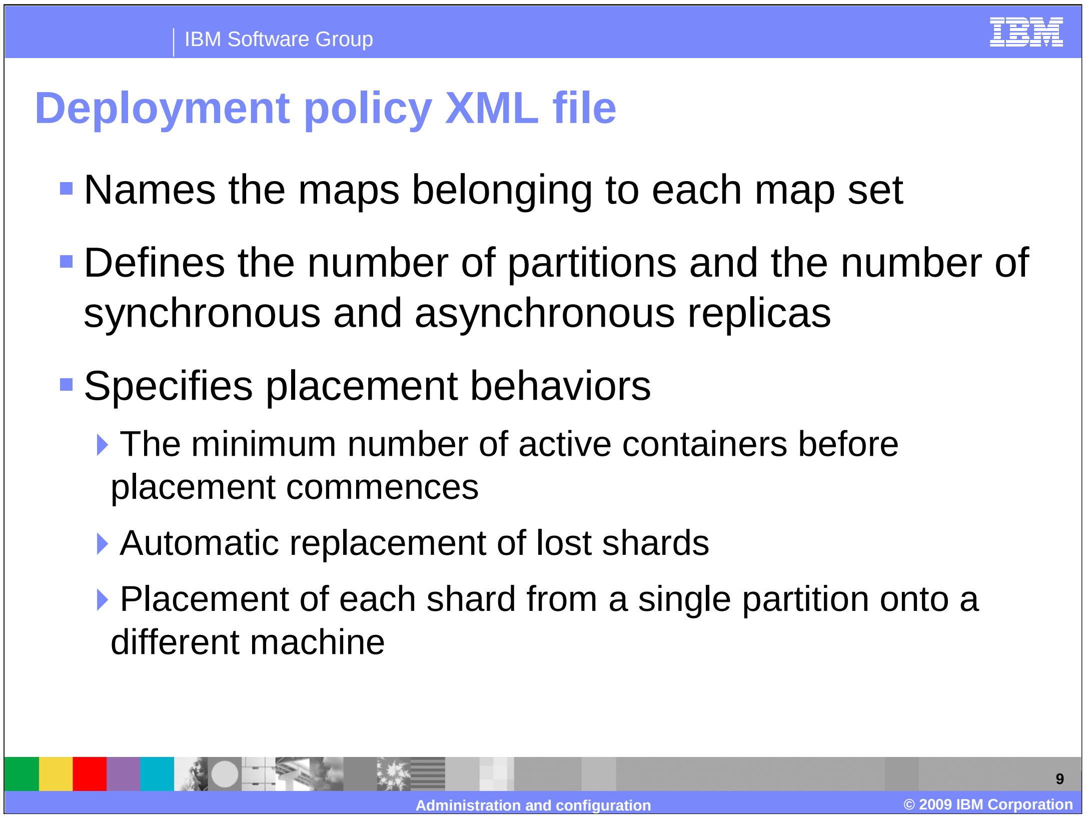
8

Administration and configuration

A distributed ObjectGrid cache can be configured by using a ObjectGrid descriptor xml file or ObjectGrid APIs. The configuration elements define each ObjectGrid, any plug-ins or event handlers and the maps within those grids. Optionally, the deployment policy XML file that is compatible with the descriptor xml file can be used to manage the deployment of an ObjectGrid into a dynamic environment.

Server topology information is not pre-configured. There are no server names or physical topology information found in the deployment policy. All shards in an eXtreme Scale grid are automatically placed into grid containers by the catalog service. The catalog service uses the constraints defined by the deployment policy to automatically manage shard placement. This allows very large grids to be easily configured. It also allows you to add servers to your environment as needed.

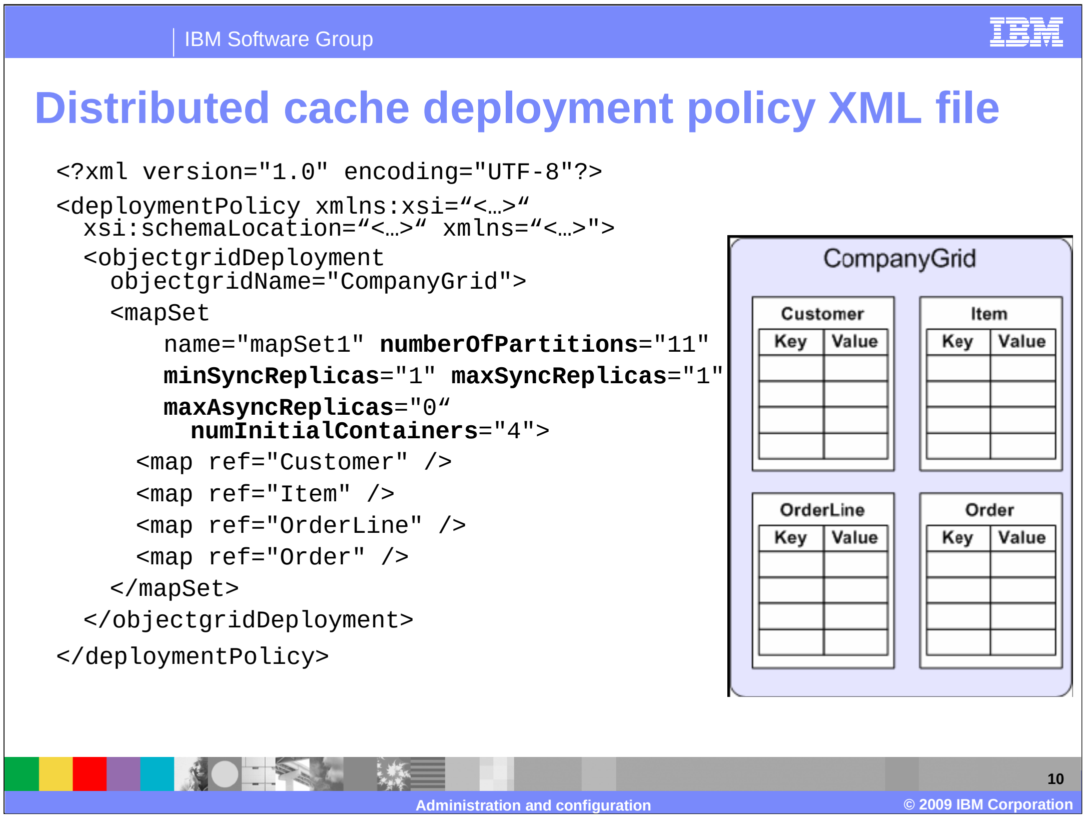
Catalog servers must be started first and in parallel if there is more than one. Then ObjectGrid servers and clients are started and communicate with the primary catalog server to connect into the grid.



Deployment policy XML file

- Names the maps belonging to each map set
- Defines the number of partitions and the number of synchronous and asynchronous replicas
- Specifies placement behaviors
  - The minimum number of active containers before placement commences
  - Automatic replacement of lost shards
  - Placement of each shard from a single partition onto a different machine

The deployment policy XML file specified the maps belonging to each map set, the number of partitions and the number of synchronous and asynchronous replicas. It also dictates these placement behaviors. The minimum number of active containers before placement will commence, the automatic placement of lost shards, and the placement of each shard from a single partition onto a different machine.



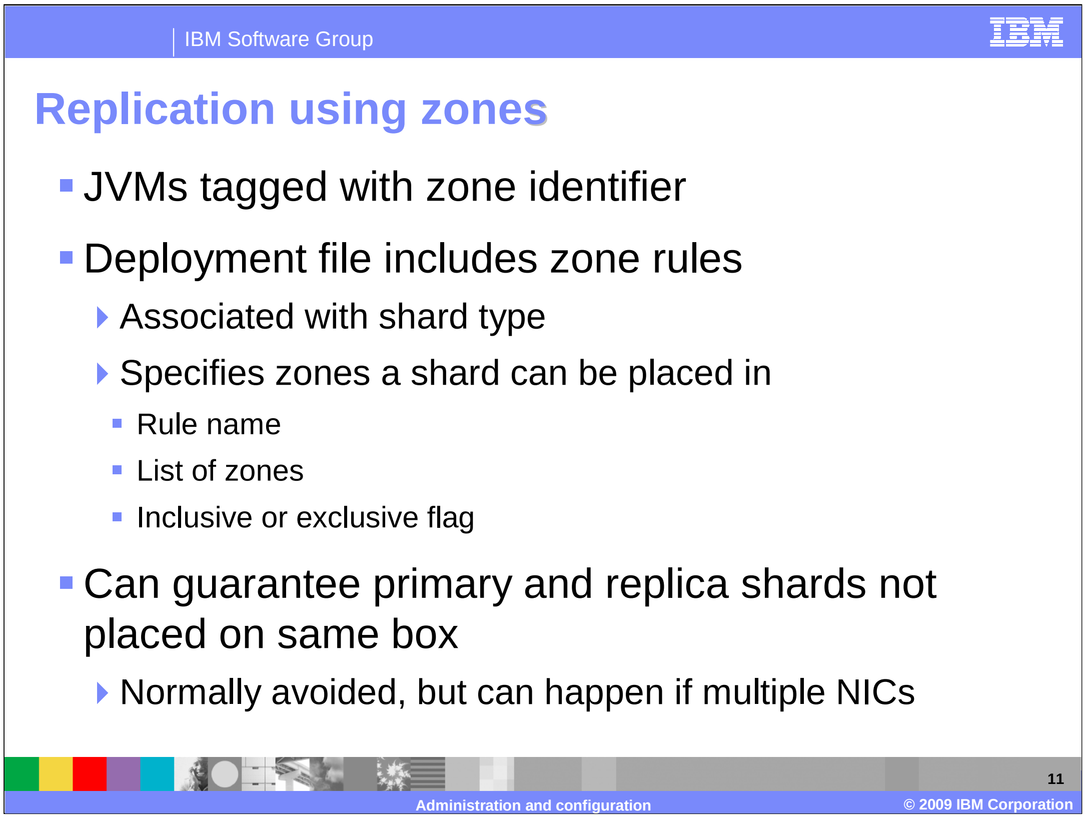
Distributed cache deployment policy XML file

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi=“<…>“
  xsi:schemaLocation=“<…>“ xmlns=“<…>">
  <objectgridDeployment
    objectgridName="CompanyGrid">
    <mapSet
        name="mapSet1" numberOfPartitions="11"
        minSyncReplicas="1" maxSyncReplicas="1"
        maxAsyncReplicas="0“
          numInitialContainers="4">
      <map ref="Customer" />
      <map ref="Item" />
      <map ref="OrderLine" />
      <map ref="Order" />
    </mapSet>
  </objectgridDeployment>
</deploymentPolicy>
```

CompanyGrid
Customer
Key
Value
Item
Key
Value
OrderLine
Key
Value
Order
Key
Value

The distributed ObjectGrid deployment policy XML is intended to be paired with the corresponding ObjectGrid descriptor XML. The deployment policy illustrated on this slide is compatible with the simple XML shown earlier in this presentation.

This deployment policy file begins with the required header for each ObjectGrid XML file. It then defines the CompanyGrid ObjectGrid with one mapSet that is divided into 11 partitions. Each partition must have exactly one synchronous replica which is dictated by the minSynchReplicas and maxSyncreplicas attributes. The inumInitialContainers instructs the catalog service to defer placement until four containers are available to support this ObjectGrid. Customer, Item, OrderLine, and Order BackingMaps are contained in the mapSet.



IBM Software Group

Replication using zones

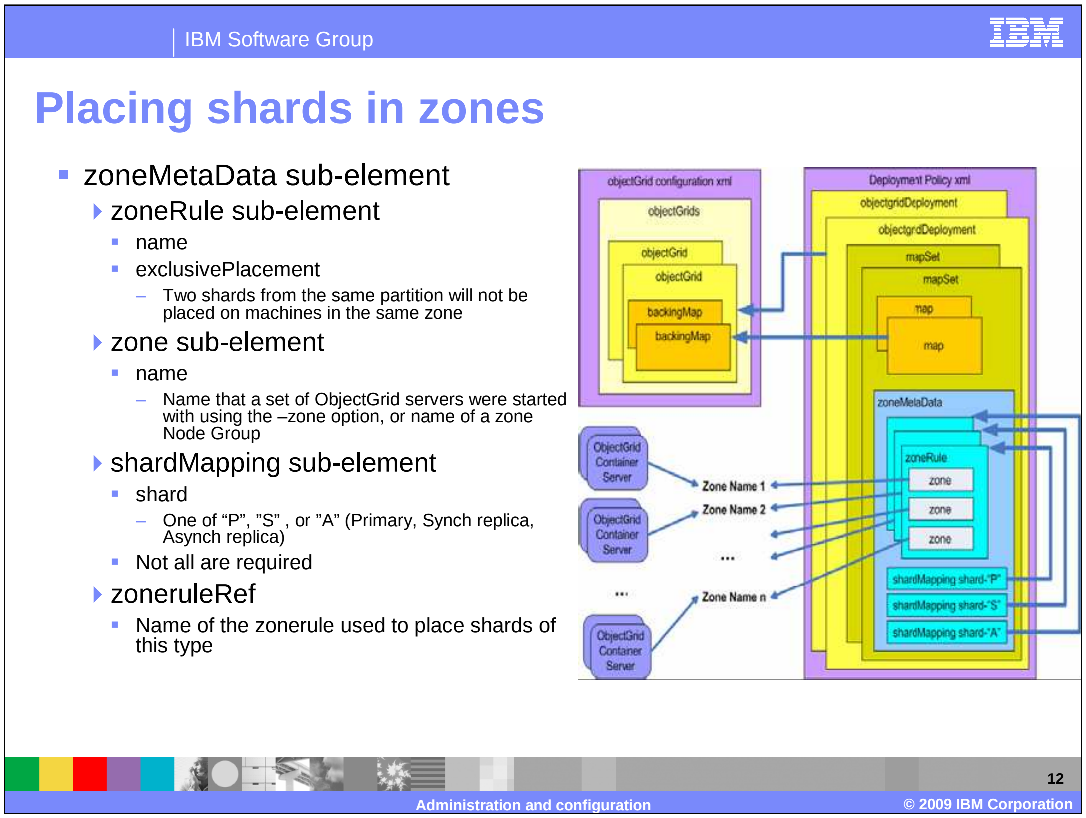
- JVMs tagged with zone identifier
- Deployment file includes zone rules
  - Associated with shard type
  - Specifies zones a shard can be placed in
    - Rule name
    - List of zones
    - Inclusive or exclusive flag
- Can guarantee primary and replica shards not placed on same box
  - Normally avoided, but can happen if multiple NICs

11

Administration and configuration

© 2009 IBM Corporation

Zoning allows more control over how eXtreme Scale places shards in a grid and can guarantee primary and replica shards are not place in the same physical location. To use zoning, JVMs that host an ObjectGrid sever must be tagged with a zone identifier. The deployment file can then include one or more zone rules. The zone rules are associated with a shard type, such as primary, synchronous replica or asynchronous replica. The rule also specifies the possible set of zones a shard can be placed in. The **inclusive** flag means that once a shard is place in a zone from the list then all other shards will also be placed in that zone. An **exclusive** setting means that each shard for a partition is placed in a different zone in the zone list. This means if there are three shards (primary and two synchronous replicas) then the zone list must have three zones in it.

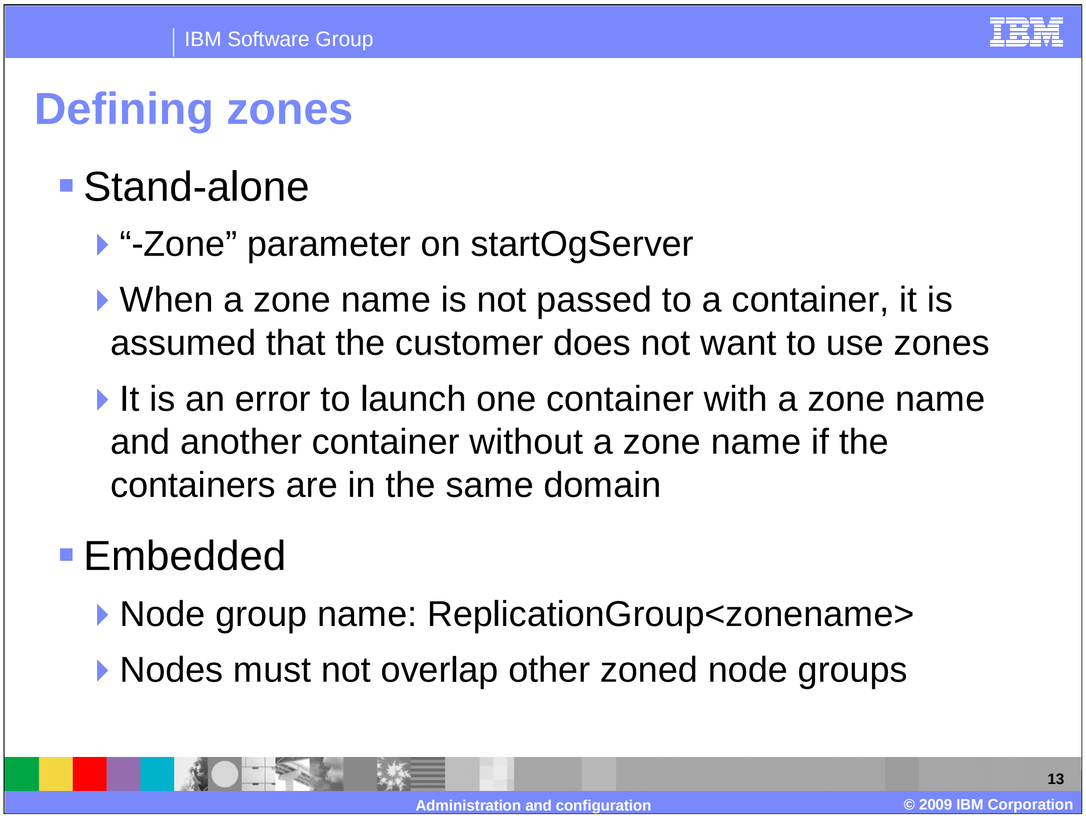


IBM Software Group

Placing shards in zones

- zoneMetaData sub-element
  - zoneRule sub-element
    - name
    - exclusivePlacement
      - Two shards from the same partition will not be placed on machines in the same zone
  - zone sub-element
    - name
      - Name that a set of ObjectGrid servers were started with using the –zone option, or name of a zone Node Group
  - shardMapping sub-element
    - shard
      - One of "P", "S" , or "A" (Primary, Synch replica, Asynch replica)
    - Not all are required
  - zoneruleRef
    - Name of the zonerule used to place shards of this type

To configure zoning, the elements listed on this slide can be included in the ObjectGrid deployment policy xml file to define the rules for placing shards in zones.



IBM Software Group

Defining zones

- Stand-alone
  - “-Zone” parameter on startOgServer
  - When a zone name is not passed to a container, it is assumed that the customer does not want to use zones
  - It is an error to launch one container with a zone name and another container without a zone name if the containers are in the same domain
- Embedded
  - Node group name: ReplicationGroup<zonename>
  - Nodes must not overlap other zoned node groups

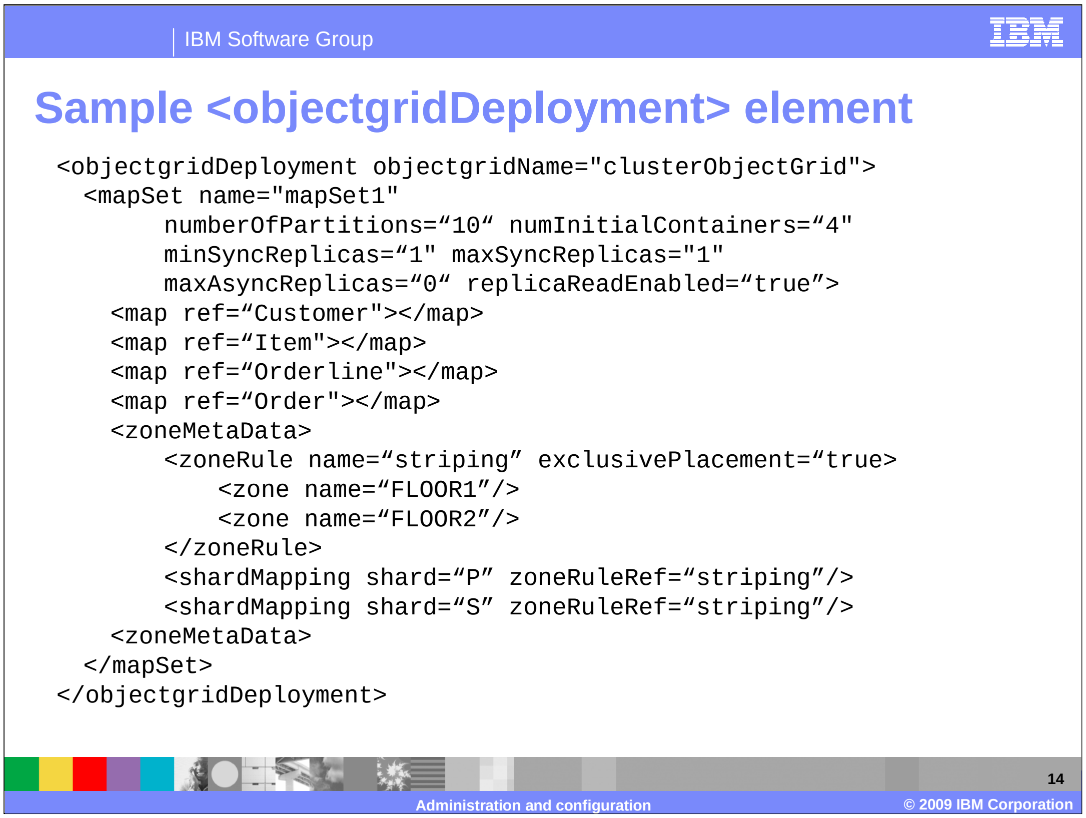
13

Administration and configuration

© 2009 IBM Corporation

When you use the startOgServer script to start a stand-alone server, specify the –zone parameter to specify the zone to use for all containers within the server. If a zone name is not passed, then it is assumed that the customer does not want to use zones. If two containers reside in the same domain and one container is launched with a zone name while the other is not, then you will receive an error.

For embedded servers, zones are identified by node group membership. Node groups can be named using the convention "ReplicationGroupZONENAME". Any nodes in such a group are members of the zone "ZONENAME". Care needs to be taken to ensure that these cluster members are not a member of two or more such node groups. A cluster member JVM checks for zone membership at start only. Adding a new node group or changing the membership will only have an impact on newly started or restarted JVMs.



IBM Software Group

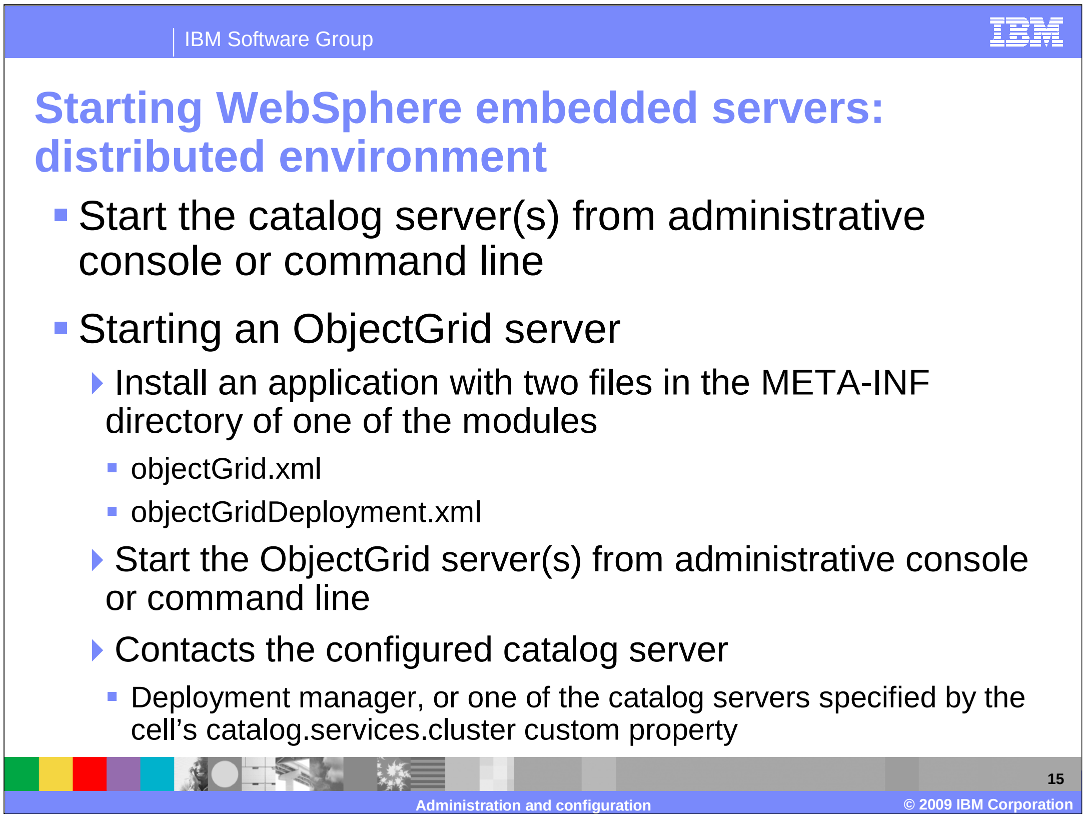
Sample <objectgridDeployment> element

```
<objectgridDeployment objectgridName="clusterObjectGrid">
  <mapSet name="mapSet1"
        numberOfPartitions=“10“ numInitialContainers=“4"
        minSyncReplicas=“1" maxSyncReplicas="1"
        maxAsyncReplicas=“0“ replicaReadEnabled=“true”>
    <map ref=“Customer"></map>
    <map ref=“Item"></map>
    <map ref=“Orderline"></map>
    <map ref=“Order"></map>
    <zoneMetaData>
        <zoneRule name=“striping” exclusivePlacement=“true>
            <zone name=“FLOOR1”/>
            <zone name=“FLOOR2”/>
        </zoneRule>
        <shardMapping shard=“P” zoneRuleRef=“striping”/>
        <shardMapping shard=“S” zoneRuleRef=“striping”/>
    <zoneMetaData>
  </mapSet>
</objectgridDeployment>
```

14

Administration and configuration

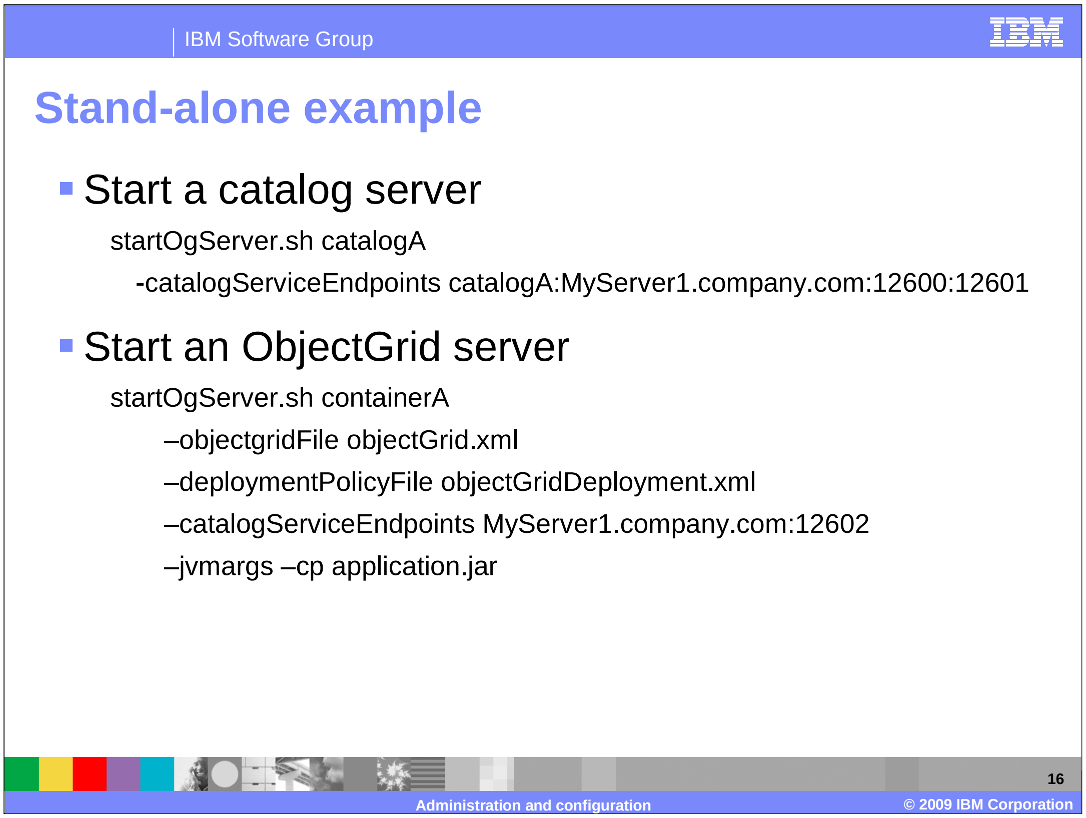
This ObjectGrid deployment policy XML file contains zone rule information, including the zone names, shard type, and whether to use exclusive placement or not.



Starting WebSphere embedded servers: distributed environment

- Start the catalog server(s) from administrative console or command line
- Starting an ObjectGrid server
  - Install an application with two files in the META-INF directory of one of the modules
    - objectGrid.xml
    - objectGridDeployment.xml
  - Start the ObjectGrid server(s) from administrative console or command line
  - Contacts the configured catalog server
    - Deployment manager, or one of the catalog servers specified by the cell’s catalog.services.cluster custom property

Launching a catalog service is the first step in bringing a distributed cache environment online. Start the catalog server or catalog server cluster from either the administrative console or command line like you would a typical WebSphere server or cluster.



Stand-alone example

- Start a catalog server

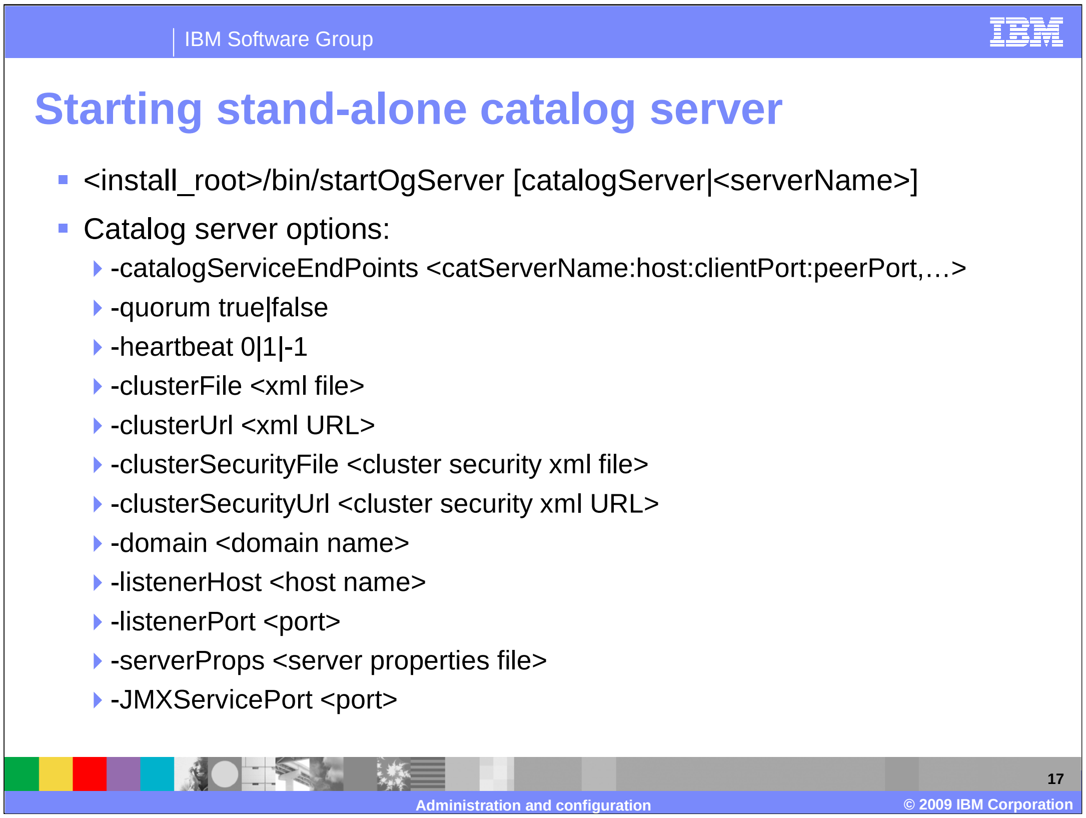
```
startOgServer.sh catalogA
  -catalogServiceEndpoints catalogA:MyServer1.company.com:12600:12601
```

- Start an ObjectGrid server

```
startOgServer.sh containerA
      –objectgridFile objectGrid.xml
      –deploymentPolicyFile objectGridDeployment.xml
      –catalogServiceEndpoints MyServer1.company.com:12602
      –jvmargs –cp application.jar
```

In this example, a server named catalogA on MyServer1.company.com is started. The server’s name is the first argument that is passed to the script and it will become the primary catalog server. During initialization of catalogA, the catalogServiceEndpoints are examined to determine which ports are allocated for this process. Other catalog servers can be specified here, so catalogA would know what other servers to accept connections from if a catalog server cluster existed.

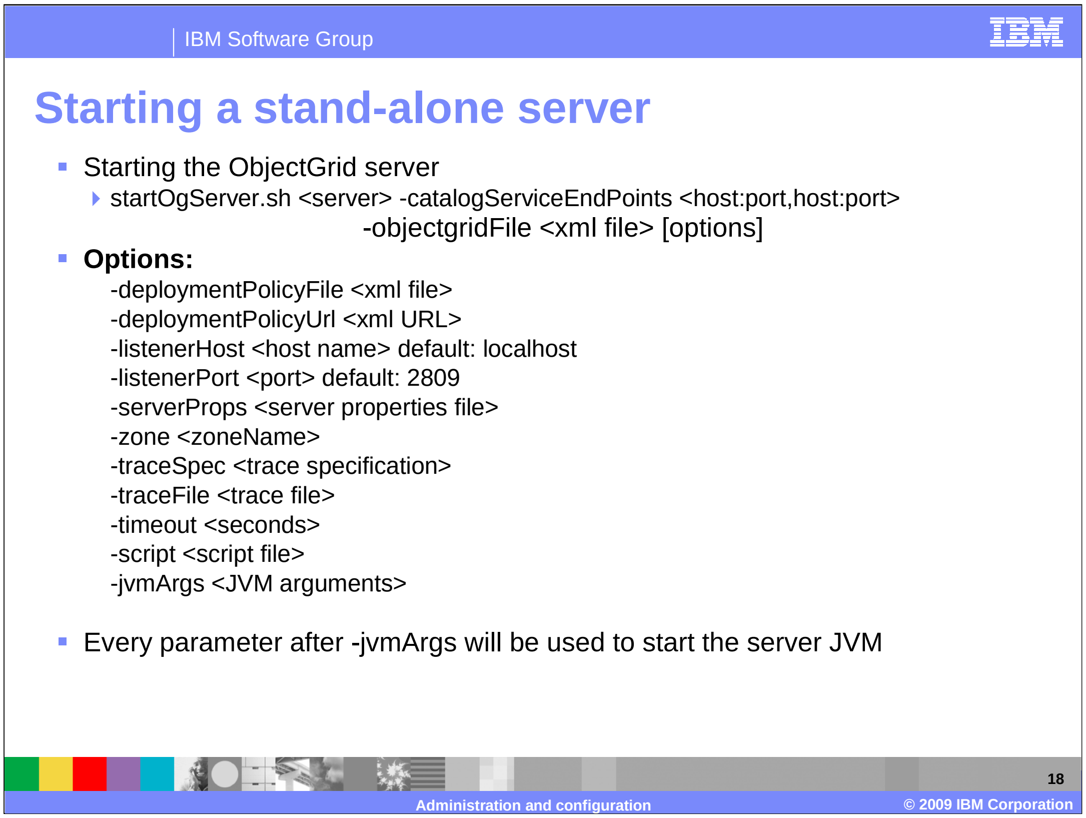
Next, an ObjectGrid server named containerA is started. The ObjectGrid descriptor file, objectgrid.xml, and the deployment policy objectGridDeployment.xml are defined. When the server’s grid container is started, it will publish its deployment policy to the catalog service specified in the cataServiceEndpints parameter. The catalog service will examine objectGridDeployment.xml to set up partitioning and replication for the eXtreme Scale grid. The –jvmargs option is used to place the application.jar file on the classpath.



Starting stand-alone catalog server

- <install_root>/bin/startOgServer [catalogServer|<serverName>]
- Catalog server options:
  - -catalogServiceEndPoints <catServerName:host:clientPort:peerPort,…>
  - -quorum true|false
  - -heartbeat 0|1|-1
  - -clusterFile <xml file>
  - -clusterUrl <xml URL>
  - -clusterSecurityFile <cluster security xml file>
  - -clusterSecurityUrl <cluster security xml URL>
  - -domain <domain name>
  - -listenerHost <host name>
  - -listenerPort <port>
  - -serverProps <server properties file>
  - -JMXServicePort <port>

Launching a catalog service is the first step in bringing a distributed cache environment online. In the stand-alone environment, the process is started using the startOgServer script. The parameter options are listed here and are also described in detail in the eXtreme Scale information center. To start a catalog server cluster, specify use the –catalogServiceEndPoints option. You must start all clustered catalog servers in parallel.



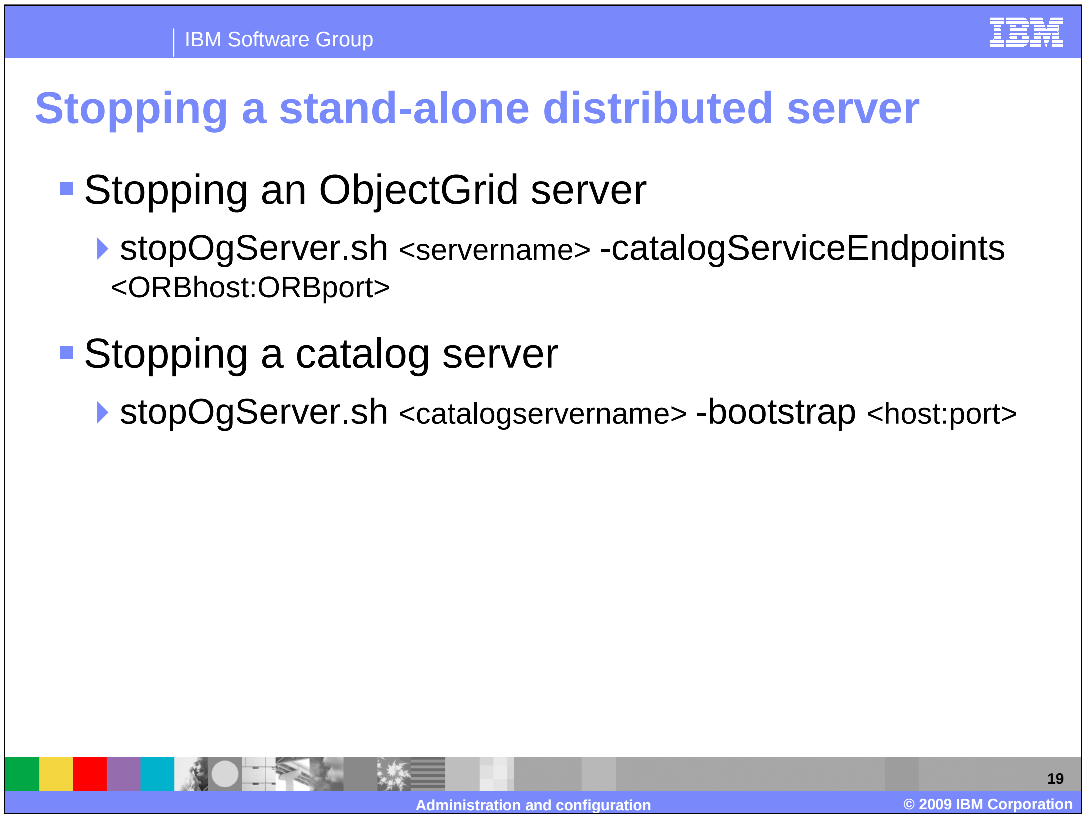
Starting a stand-alone server

- Starting the ObjectGrid server
  - startOgServer.sh <server> -catalogServiceEndPoints <host:port,host:port> -objectgridFile <xml file> [options]
- **Options:**

  -deploymentPolicyFile <xml file>
  -deploymentPolicyUrl <xml URL>
  -listenerHost <host name> default: localhost
  -listenerPort <port> default: 2809
  -serverProps <server properties file>
  -zone <zoneName>
  -traceSpec <trace specification>
  -traceFile <trace file>
  -timeout <seconds>
  -script <script file>
  -jvmArgs <JVM arguments>

- Every parameter after -jvmArgs will be used to start the server JVM

To launch a stand-alone distributed server, use the startOgServer script. The parameter options are listed here and are also described in detail in the eXtreme Scale information center. Every parameter listed after –jvmArgs will be used to start the server JVM, so ensure that it is the last argument specified.

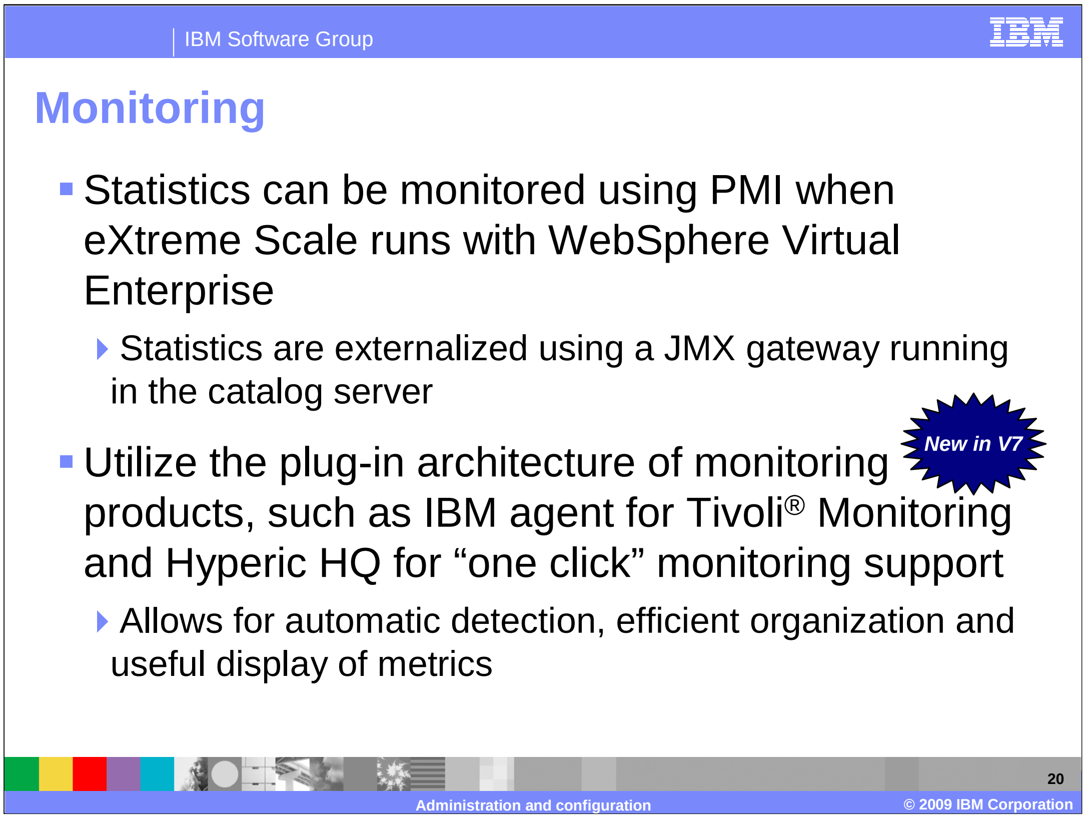


Stopping a stand-alone distributed server

- Stopping an ObjectGrid server
  - stopOgServer.sh <servername> -catalogServiceEndpoints <ORBhost:ORBport>
- Stopping a catalog server
  - stopOgServer.sh <catalogservername> -bootstrap <host:port>

Once an ObjectGrid container process has been started, you can stop that process with the stopOgServer script. The –catalogServiceEndpoints argument is used to reference the Object Reference Broker, or ORB, host and port on the catalog service. The -catalogServiceEndpoints option should NOT be used to reference the hosts and ports that were passed to the -catalogServiceEndpoints on the catalog service.

To stop a catalog service process, the stopOgServer script is also used, but with the bootstrap argument specified.



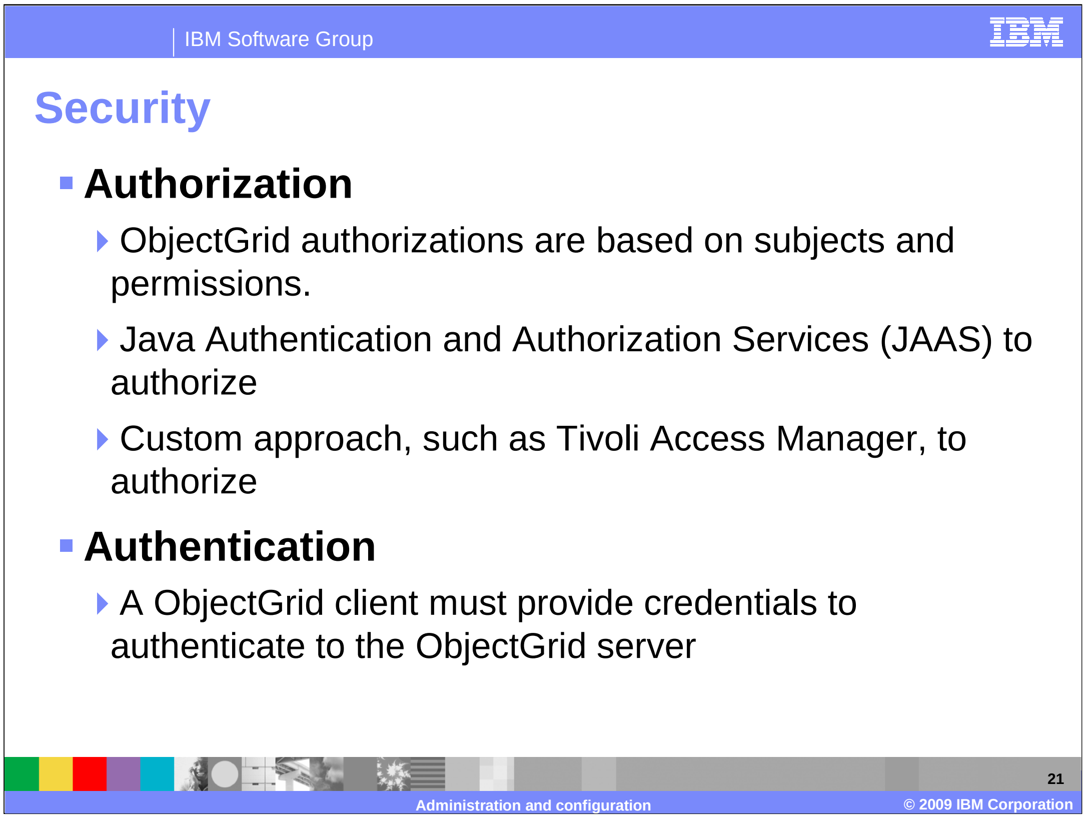
Monitoring

- Statistics can be monitored using PMI when eXtreme Scale runs with WebSphere Virtual Enterprise
  - Statistics are externalized using a JMX gateway running in the catalog server
- Utilize the plug-in architecture of monitoring products, such as IBM agent for Tivoli® Monitoring and Hyperic HQ for "one click" monitoring support
  - Allows for automatic detection, efficient organization and useful display of metrics

New in V7

WebSphere eXtreme Scale includes statistics modules for use with the WebSphere Virtual Enterprise Performance Monitoring Infrastructure. You can access statistics using a standard Java Management Extensions gateway such as Tivoli Performance Viewer.

WebSphere eXtreme Scale V7.0 can also use the plug-in architecture of monitoring products, such as Hyperic HQ and Tivoli. This will allow for automatic detection, efficient organization, and useful display of metrics.

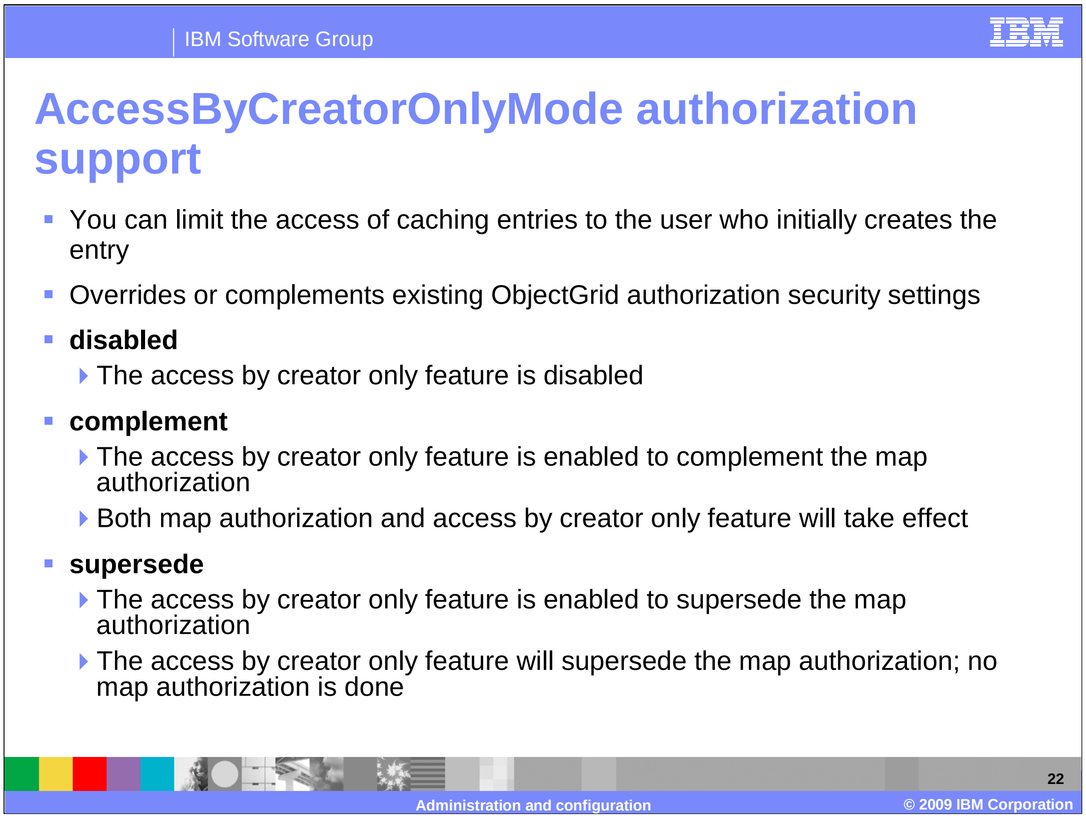


Security

- **Authorization**
  - ObjectGrid authorizations are based on subjects and permissions.
  - Java Authentication and Authorization Services (JAAS) to authorize
  - Custom approach, such as Tivoli Access Manager, to authorize
- **Authentication**
  - A ObjectGrid client must provide credentials to authenticate to the ObjectGrid server

You can enforce security in your eXtreme Scale environment by configuring an external security implementation. Authorizations are based on subjects and permissions and can be configured by providing a security descriptor XML file to the catalog server. eXtreme Scale can successfully integrate with Java Authentication and Authorization Service, Tivoli Access Manager, Kerberos, Lightweight Directory Access Protocol (LDAP), or WebSphere Application Server security.

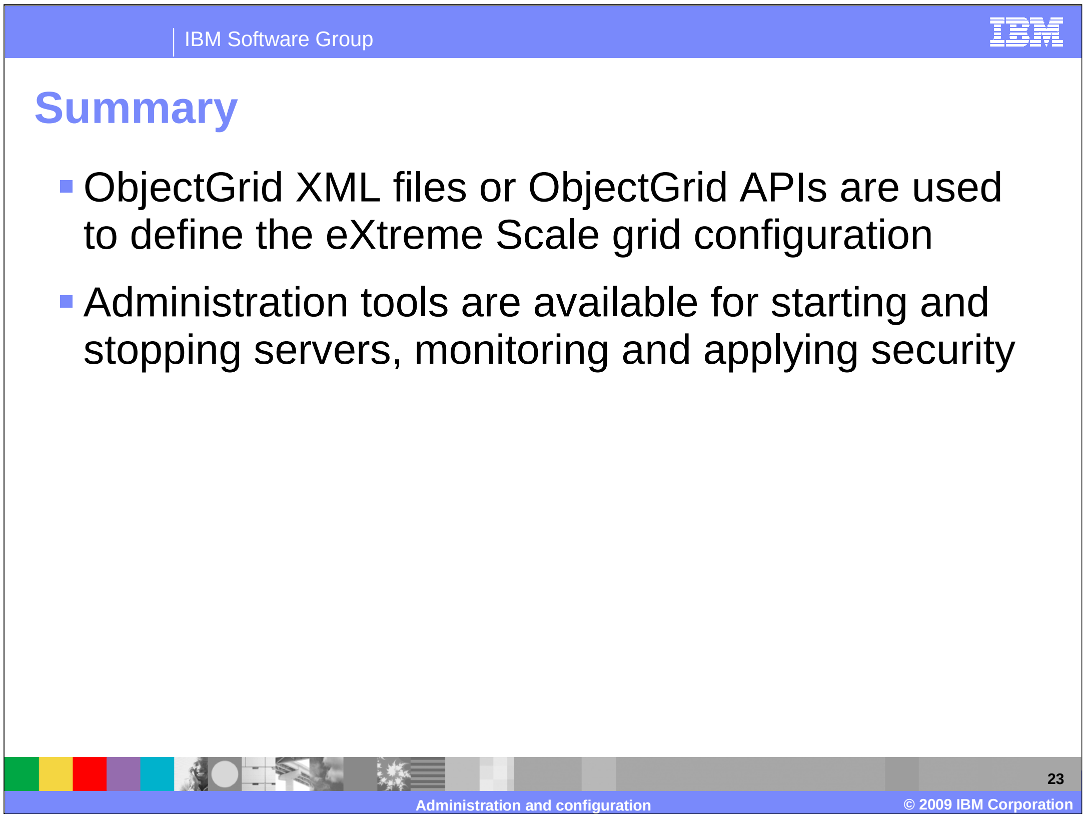
eXtreme Scale uses the security provider to authorize which users access a particular cache entry or to specify what can be done with certain ObjectGrid artifacts. Also, security must be enabled on both the client and server in order to be able to successfully authenticate with the eXtreme Scale grid.



AccessByCreatorOnlyMode authorization support

- You can limit the access of caching entries to the user who initially creates the entry
- Overrides or complements existing ObjectGrid authorization security settings
- **disabled**
  - The access by creator only feature is disabled
- **complement**
  - The access by creator only feature is enabled to complement the map authorization
  - Both map authorization and access by creator only feature will take effect
- **supersede**
  - The access by creator only feature is enabled to supersede the map authorization
  - The access by creator only feature will supersede the map authorization; no map authorization is done

The AccessByCreatorOnlyMode is a form of authorization support that limits access to cache entries to only the user who initially created the entry. This mode can compliment or override any existing ObjectGrid authorization security settings.

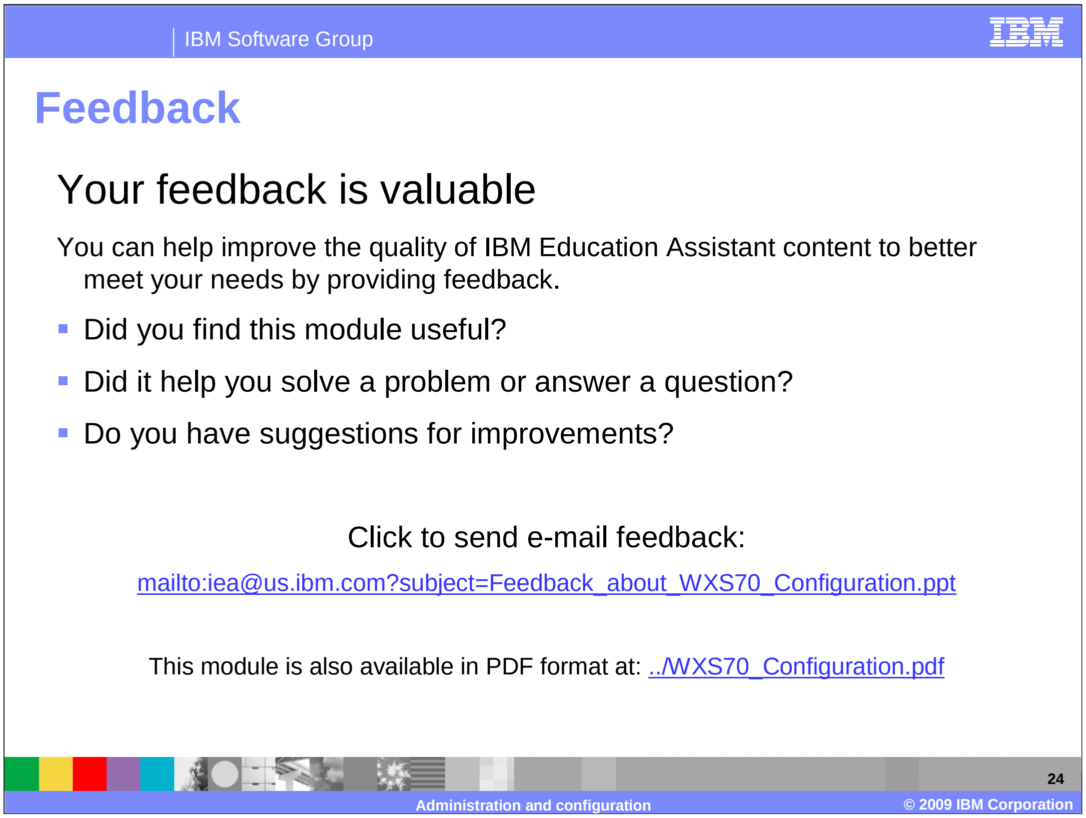


Summary

- ObjectGrid XML files or ObjectGrid APIs are used to define the eXtreme Scale grid configuration
- Administration tools are available for starting and stopping servers, monitoring and applying security

ObjectGrid XML files or ObjectGrid APIs can be used to define and configure the ObjectGrid environment. In a distributed environment, all shards in an eXtreme Scale grid are automatically placed into grid containers by the catalog service following the constraints defined by the configuration code. This allows very large grids to be easily configured and allows you to add servers to your environment as needed.

WebSphere eXtreme Scale also provides all of the administration tools needed to start and stop the servers, monitor the grid environment and apply security.



Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

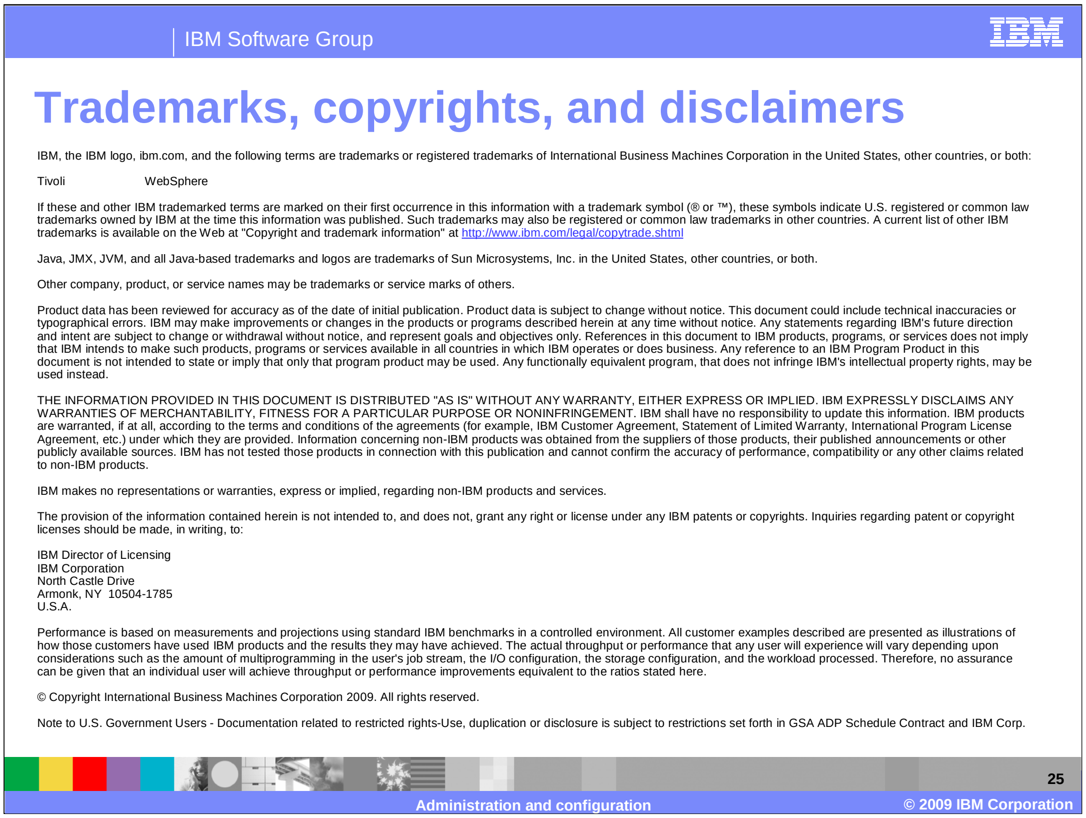
- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_WXS70_Configuration.ppt

This module is also available in PDF format at: ../WXS70_Configuration.pdf

You can help improve the quality of IBM Education Assistant content by providing feedback.



Trademarks, copyrights, and disclaimers

IBM, the IBM logo, ibm.com, and the following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

Tivoli WebSphere

If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of other IBM trademarks is available on the Web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml

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

Other company, product, or service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.