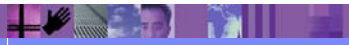

This presentation explains how to extend IBM WebSphere Development Studio Client for iSeries to add your tools.

# Agenda

- Brief Introduction to Eclipse
- WebSphere Studio Workbench
- Extending the Workbench
  - Writing Eclipse plug-ins
  - HelloWorld plug-in example
- WebSphere Development Studio Client
  - iSeries Tools in the Workbench
  - Extending Development Studio Client
- Summary



# What is Eclipse?

## What is Eclipse?


- Eclipse is an empty slate:
  - Except for the Java tools it supplies
  - Use it to build AD products
    - By writing "plug-ins" that offer tools for particular programming needs
    - Plug-ins "snap into" Eclipse seamless
    - Eclipse + useful plug-ins = product
  - Common base for all new IBM AD products
  - Common base for many new 3rd party products
- Eclipse is an opportunity
  - To build and sell plug-ins for all eclipse users
    - All users of Eclipse and Eclipse-based products




So what is Eclipse? With the exception of the built-in Java tools, Eclipse itself is not that interesting. It only gets interesting when you add to it some plug-ins (for example, tools) that do something interesting.

An Eclipse-based product is Eclipse plus a number of interesting plug-ins. IBM is building numerous such Eclipse-based products, including as we have seen Site Developer and Development Studio Client. Further, because Eclipse is free and business partners are free to include it in their products, there will be many other Eclipse-based products from other companies too, including Rational and TogetherSoft.

For business partners or software developers who write and sell application development tools, Eclipse is a fantastic opportunity. By writing plug-ins for Eclipse, those plug-ins can be sold to any developer using any product based on Eclipse or even just the raw Eclipse as downloaded from www.eclipse.org. This opportunity is not lost on iSeries tool vendors, who are all looking at offering Eclipse plug-ins for their tools. This will result in a rich offering of third party plug-ins for developers to choose from, all of which extend their core Development Studio Client development environment. One community, one core development platform, many IBM and 3rd party tools. This is community and excitement!

## Eclipse Open Source Organization

- An open source community chartered to manage the evolution of an open application development integration platform
- Managed by an independent, multi-vendor organization
- Technology licensed via the Common Public License
- Based on Java with initial support for Linux & Windows
- Broad Industry Support


IBM Technology Contribution

eclipse

**www.eclipse.org**



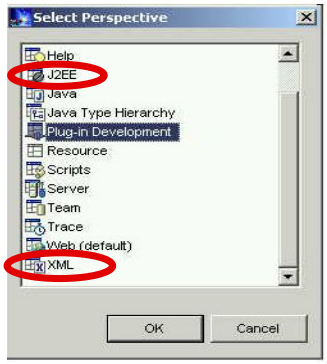

Eclipse was developed by IBM and donated to the open source community. That donation is estimated to be worth $40 million. Anyone can download Eclipse for free, including the source code, from www.eclipse.org. Eclipse has generated extraordinary excitement in the development community and the tools community. It is written in Java, and can be extended by tools that are also written in Java. These tools are known as plug-ins. Out of the box, Eclipse offers an integrated development environment (IDE) that has built-in support for teams and projects and a robust and revolutionary user interface framework. It also has tools built-in to create Eclipse plug-ins. Further, there are extensive and very powerful tools built-in for developing Java applications with Eclipse. So, if all you want is the world's best Java toolset, then all you need is Eclipse. You can't beat the price!

# Eclipse Workbench

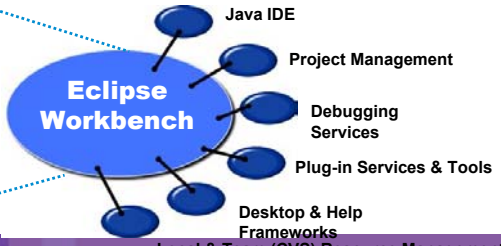
**A 'Portal-like' Tool Integration Platform**

**A personalized, extensible development platform organized via developer**

**"Perspectives"**




Products built with Eclipse inherit these capabilities plus 'plug-ins' built by others




**Local & Team (CVS) Resource Management**




Some of the common services Eclipse supplies include: the Java integrated development environment (IDE), project management, debug, plug-in frameworks, desktop and help frameworks, and resource managers.

# Eclipse – Broad Industry Support

**Over 135 companies have been involved in the Eclipse project, including these industry leaders ...**

- Altoweb
- Bowstreet
- BrowserSoft
- CommerceQuest
- Computer Assoc.
- Compuware
- Crystal Decisions
- Embarcadero
- Holosofx
- Instantiations
- LegacyJ
- Macromedia
- Merant
- Mid-Comp
- Neuvis
- QSSL
- Rational
- Red Hat
- Serena
- Sitraka
- Skyva
- TeleLogic
- TogetherSoft
- Versant
- Versata
- Aldon
- MKS
- Softlanding
- Lansa
- ASC


**www.Eclipse.org**

**"This has long been a dream of developers, and now IBM is providing the integration framework to make it possible."**
Arun Gupta, CTO, Neuvis Software

**"The goal of Eclipse to provide a seamless interface where developers can leverage tools from multiple vendors is a great idea and is something that will add value to our customers. Eclipse is a great way to make it easy for iSeries customers to integrate WebSphere and leading development tools like LANSA".**

**Bill Benjamin**
**Vice President, Business Development**
**LANSA Inc.**

**Not a complete list!!**




IBM is not alone with Eclipse. The open-source consortium that oversees contributions made to it include a number of large companies, and that list is growing. Note that MKS is an iSeries tool vendor. See www.eclipse.org for the latest list.

# What is WebSphere Studio Workbench?

WebSphere. software

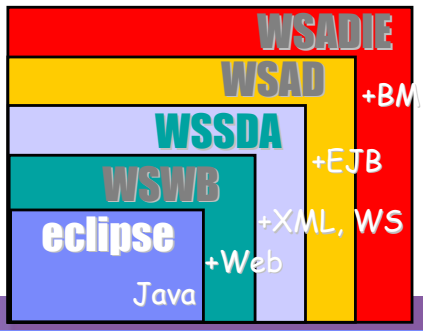
e business software

## WebSphere Studio Workbench

- Basis of all next-gen IBM AD products. Eg:
  - WebSphere Studio Site Developer
  - WebSphere Studio Application Developer
  - WebSphere Studio Integration Edition
- Available to IBM Business Partners
  - Via IBM PartnerWorld for Developers
    - ✓ To build/test plugins
    - ✓ To create products

Each product is a different "configuration" of WebSphere Studio

- ► Refers to product family







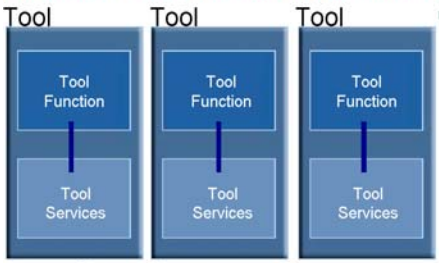
Here you see that Development Studio Client is based on WebSphere Studio Site Developer, while Development Studio Client Advanced is based on WebSphere Studio Application Developer.

The Workbench is based on the open-source Eclipse technology about to be discussed. It is not for sale, but is the basis of all IBM WebSphere Studio products, and is available to business partners.

Site Developer is IBM's entry level offering based on eclipse, and it is for building dynamic Web sites out of non-EJB Java. Application Developer extends Site Developer and adds support for EJBs. Application Developer-Integration Edition extends Application Developer and adds support for JCA Connectors and for Workflow. Enterprise Developer extends Application Developer-Integration Edition and adds support for S/390 and Enterprise Generation Language (EGL), the follow-on to VisualAge Generator.

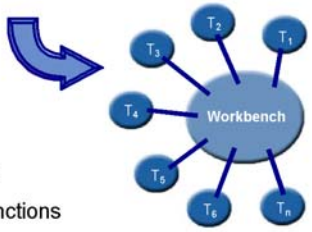
# WebSphere Studio Workbench (WSWB)

**A fundamental change in AD Tooling paradigm**



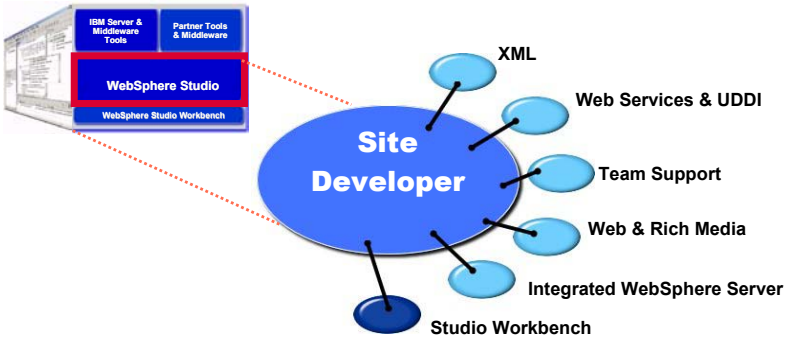

- **Challenges**
  - Difficulty of integration
  - Different Semantics, UI etc
  - Different repositories
  - Not best-of-breed Tool Services
  - Difficult to manage
  - Slow to Market

- **Benefits**
  - Easier integration
  - Single view/mgmt
  - Common look and feel
  - Best-of-breed Tool Services
  - Tool builder focus on tool functions
  - Fast to Market



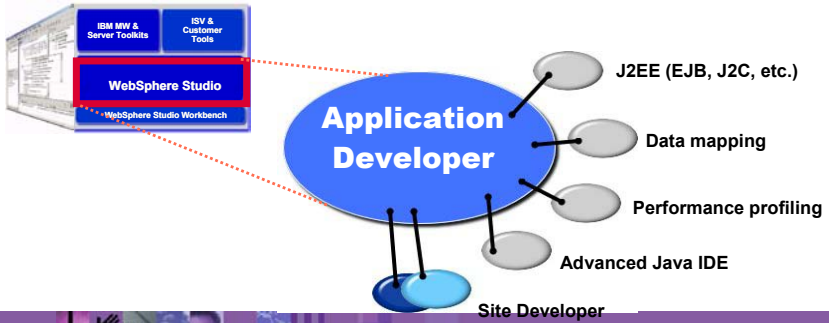
# New WebSphere Studio Site Developer

- For professional developers of dynamic Web applications & sites
- Delivers integrated support for open Web standards, including Java, JSP, XML, Rich Media & Web services tools



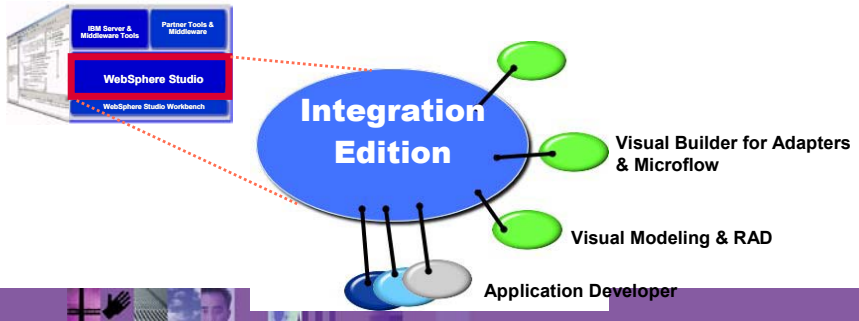
# New WebSphere Studio Application Developer

- For professional developers of Java & J2EE applications, requiring integrated Web, JSP, XML & Web services support
- Advanced tools for code generation, performance tuning & quality
- Includes a highly integrated WebSphere Application Server test environment and deployment automation tools



# New WebSphere Studio Integration Edition

- For developers & integrators of advanced J2EE & Web services
- RAD tools for visual application modeling, composition & adaptability (e.g., business rules)
- Advanced EAI tools for generating adapters & dynamic microflows including integration of remote host assets



IBM Software Group

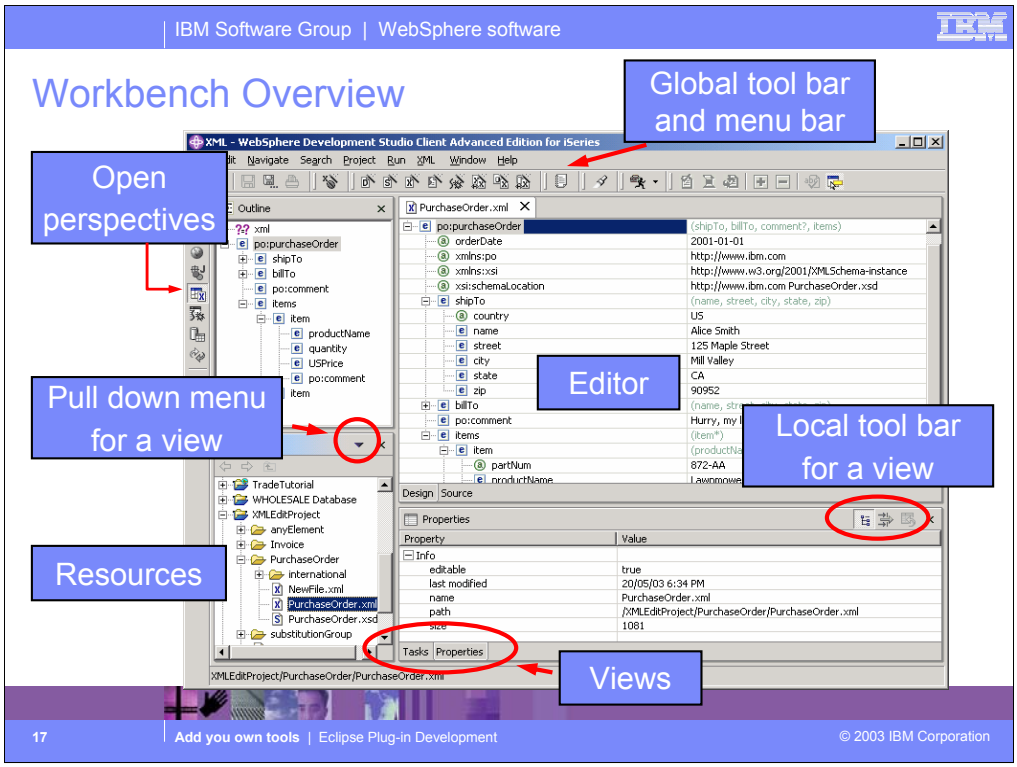
# Eclipse Technology and Terminology


# Workbench


- Perspectives
  - Contain views and editors
  - Are named and have default layout of the views/editors
  - Are targeted towards a specific type of development, i.e. Java, XML, Help, Debug, WebFacing, ...
- Editors
  - That open, edit, save and close files
  - Can contribute to the Workbench menu bar and tool bar
- Views
  - That display information
  - Can have their own toolbars
  - Updates saved automatically
- Miscellaneous
  - Actions, Menus, Toolbars, Wizards, Properties, Preferences

# Workspace

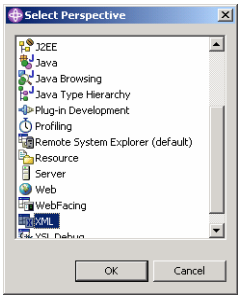
- Workspace is the term given to refer to all development resources currently accessible in the Workbench
- Eclipse Resource Workspace
  - Resources = Projects + Folders + Files
    - Projects map to a folder in local file system
    - Projects contain folders and files
    - Folders contain folders and files
    - Projects are typed
  - Resources are versioned as they change
    - User can compare and restore versions
  - Resources can have properties
    - Properties are displayed in the properties view

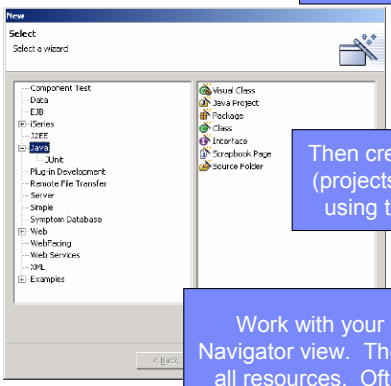




The Eclipse workbench has four open perspectives. You can see them lined up on the left frame of the workbench. The current active perspective is the one whose icon is indented, which in this case is the Java perspective. To open a new perspective, you use Window->Open Perspective giving you the Select Perspective dialog that you saw previously. This particular perspective has a Package Explorer view which drives the Editor and the Outline view.

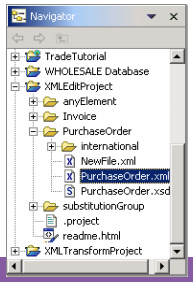
# Using Eclipse

Start by opening a perspective based on the type of development they are working on.



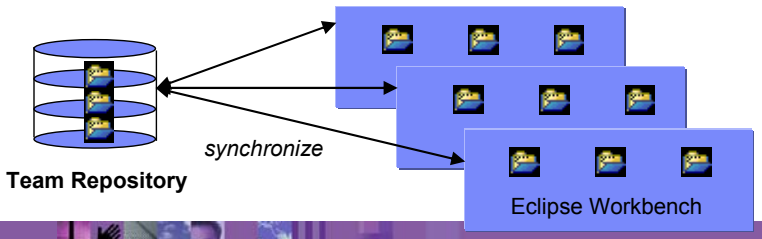

Then create new resources (projects, files and folders) using the “New” wizards




Work with your resources using the Navigator view. The Navigator view shows all resources. Often it is easier to use a local view which filters out projects that are not related. For example the “Packages view” shows only Java projects.

# Eclipse Team Support

- Users create and work with **projects**
  - Contain **folders** and **files**
    - Which are versioned as they change
    - Are “typed” and have type-unique tooling
    - Are **synchronized** between local workspace and team repository







Central to the Eclipse integrated development environment is support for projects. Projects are simply a grouping of folders and files.

Plug-ins have types, such as Java or Web or XML. Tools that plug into Eclipse can define their own new project types. Tools that plug into Eclipse work against resources (folders or files) within projects. They can be scoped to projects of a particular type, if appropriate.

All projects, regardless of type, have common behavior and support. This includes the ability for a team to share a project by using an Eclipse-supporting software change management (SCM) product such as Concurrent Version System (CVS) or Rational Clearcase. This SCM acts as a central repository for one or more projects. Each team member can easily keep their local copy of any project "in synch" with the central repository copy. CVS is a free open-source SCM. It runs on Linux, UNIX and Windows. It comes on the Linux distribution CDs for iSeries Linux LPAR.

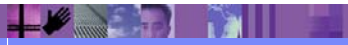
All IBM SCM vendors for iSeries are enabling, or have enabled, their products to Eclipse.

IBM Software Group

# Developing Eclipse Plug-ins


# Plug-in Development

- You add functions to the Workbench by developing plug-ins
- Plug-ins
  - Highest level of organization / structure for adding functionality to the Workbench
    - WebFacing plug-in
    - Remote Systems Explorer plug-in
    - Web Tooling plug-in
    - iSeries extensions to Web Tooling plug-in
    - ...
  - A collection of files which implement your desired function
  - Plugin installed by copying your files to a subdirectory of <root>/plugins
    - There are built in tools for installing new plug-ins
  - Described by the plugin.xml file
    - Eclipse will look for this file in all subdirectories of <root>/plugins
    - Describes what function(s) your plug-in is adding to the Workbench

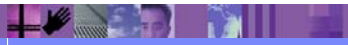


# Anatomy of a Plug-in

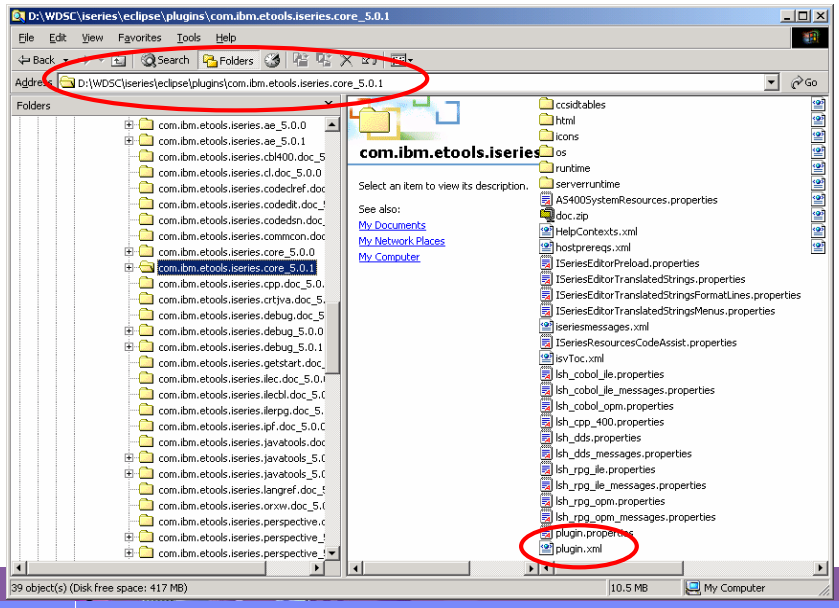
- Files typically found in a plug-in

<root>/plugins/com.ibm.example/

| | |
|---|---|
| plugin.xml | - describes the plug-in |
| plugin.properties | - translatable strings for plugin.xml |
| example.jar | - Java code for the plug-in |
| example.properties | - translatable strings |
| HelpContexts.xml | - structure for F1 help |
| doc.properties | - translatable strings for help |
| doc/ | - plug-in documentation |
| *.html | |
| nl/ | - NL versions of .properties files |
| en_US/ | |
| ... | |
| icons/ | - graphics files for your plug-in |
| ... | |

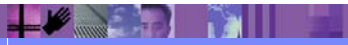


# Example Anatomy of a Plug-in



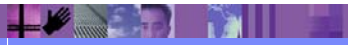
# Plugin.xml File

- An XML file that describes:
  - Prerequisite plug-ins
  - Jar files shipped with your plug-in
  - Extension points implemented by your plug-in
  - Declares new extension points added by your plug-in
- Required for every plug-in
- Special plugin.xml editor included in the Eclipse plug-in development environment

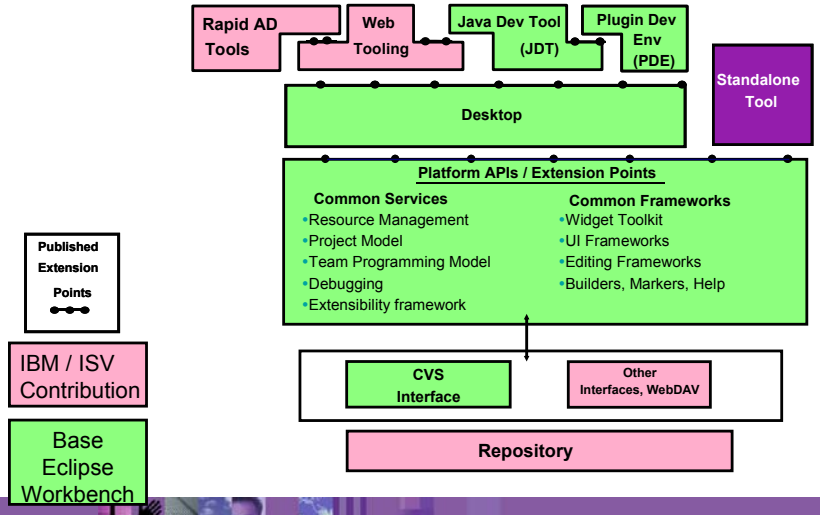


# Extension Points

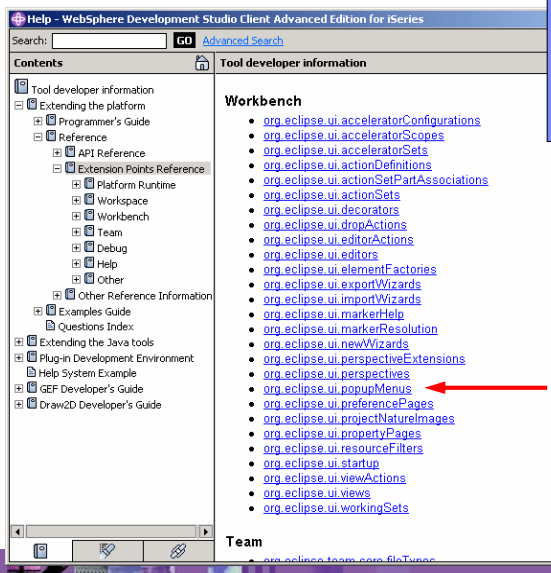
- Allow you to add functions to the workbench
  - Declared in the plugin.xml file
  - Associated Java code is shipped in JAR files
  - First you need to find the extension point for the new function you want to add
    - Workbench extension points are documented in the online help
    - Includes the XML DTD for the extension point and example XML code



# Extension Points



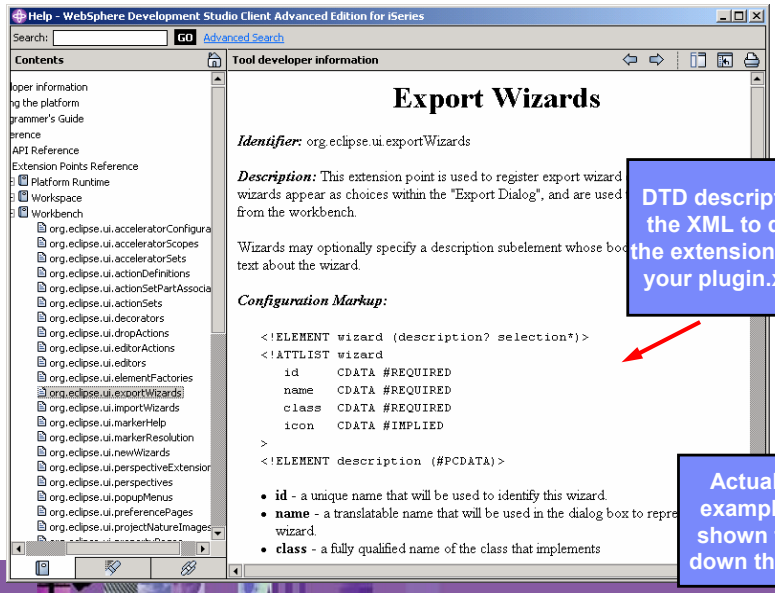
# Example Extension Points




**Documentation for Workbench extension points is available in the online help under *Tool Developer Information***

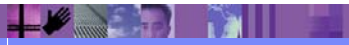
Add your own popup menus to the Workbench

# Extension Point Documentation



# Hello World Plug-in Example

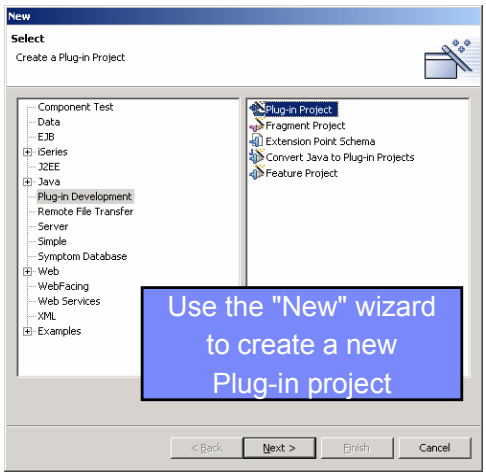
- Create a new plug-in project
- Examine the Plug-in Development perspective
- Add a menu and menu item to the main menu
- Supply an action for the menu item that displays a message box

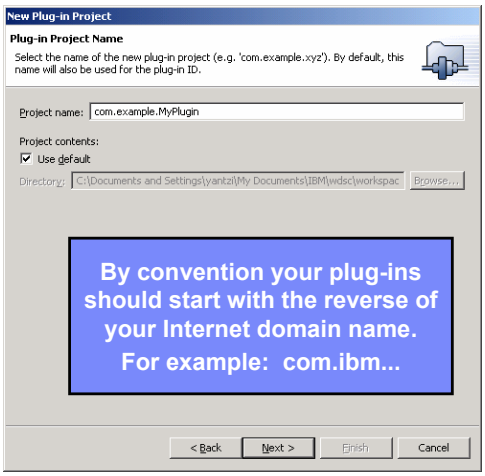




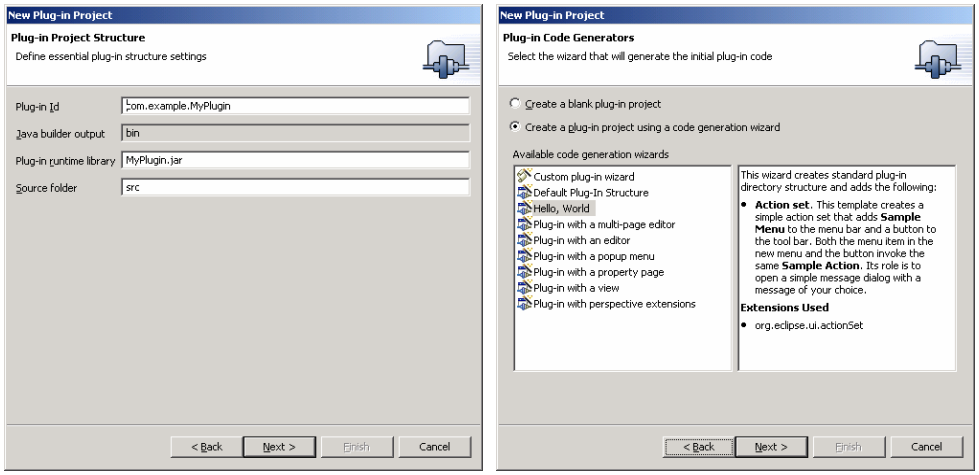

Now lets walk through an example

# Creating a new Plug-in Project

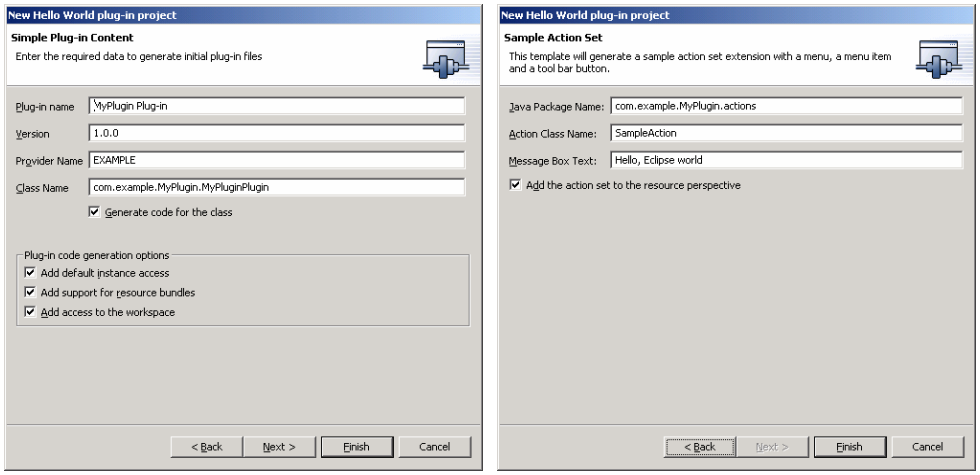


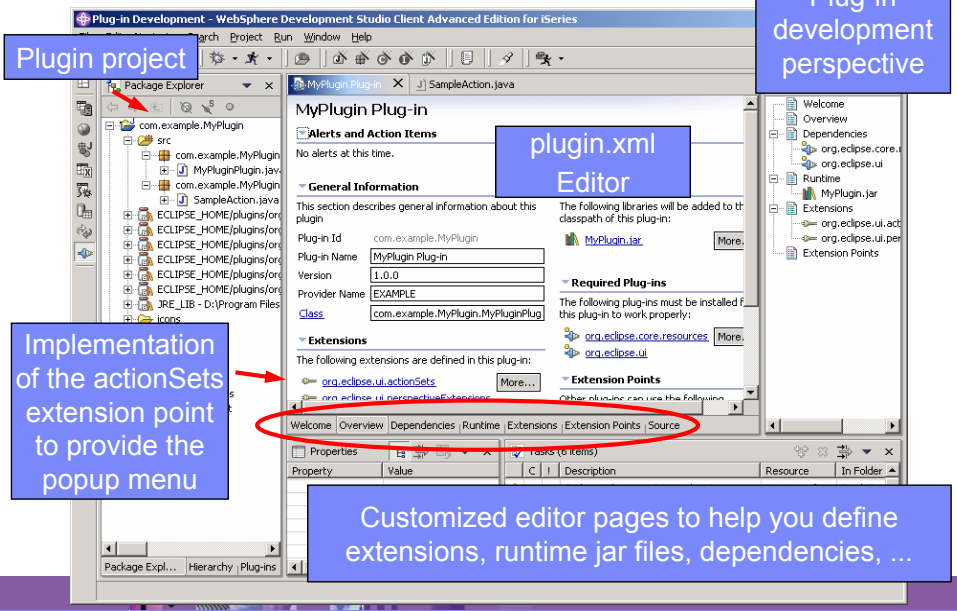
# Creating a New Plug-in Project



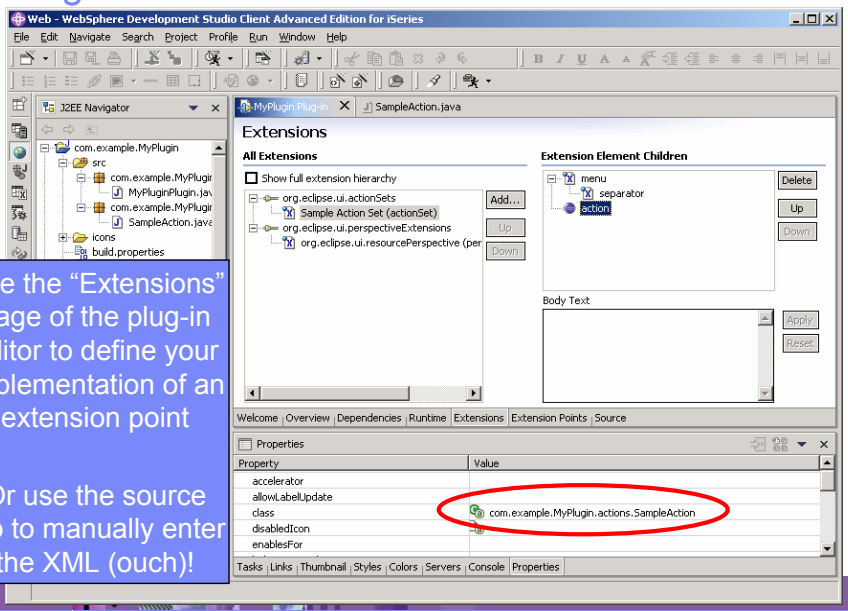
# Creating a New Plug-in Project



# Plug-in Development Environment



# Defining the Extension

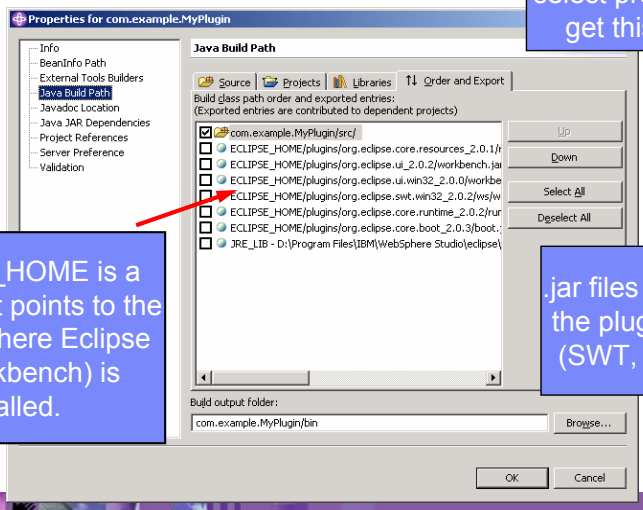



Use the “Extensions” page of the plug-in editor to define your implementation of an extension point

Or use the source tab to manually enter the XML (ouch)!

# Plug-in Build Path

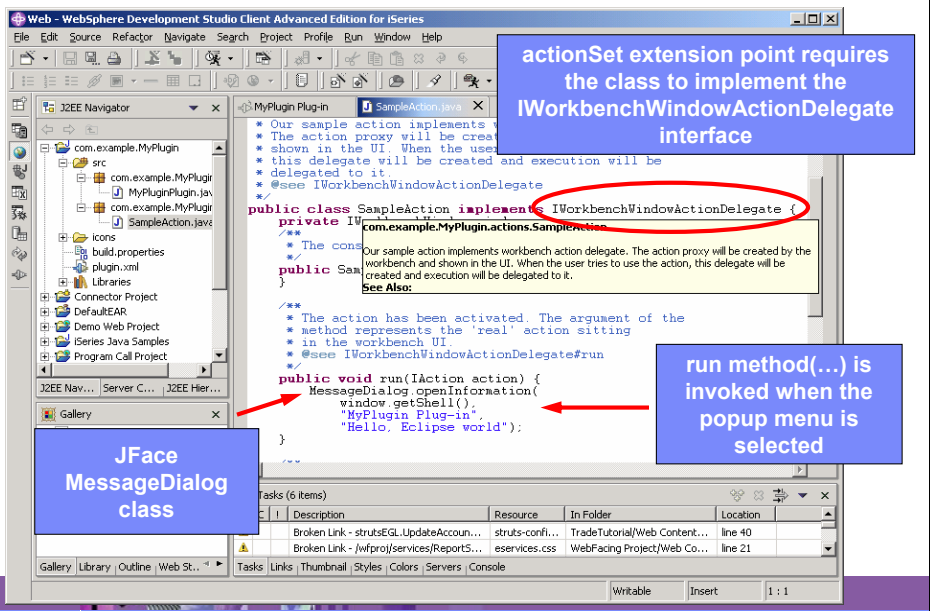
Right click on the PDE project and select properties to get this dialog



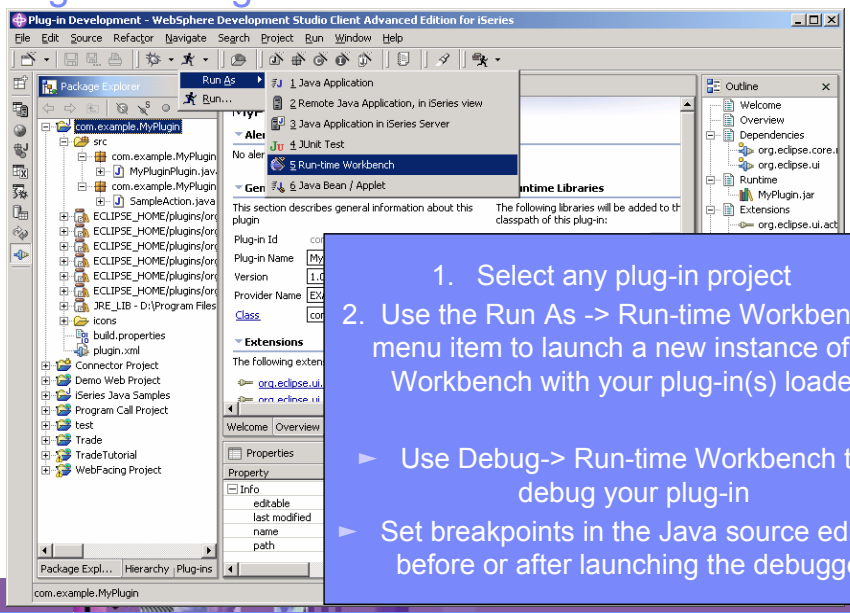

ECLIPSE_HOME is a variable that points to the directory where Eclipse (the workbench) is installed.

.jar files required for the plug-in to build (SWT, JFace, …)

# Action Class (Invoked by the Popup Menu)



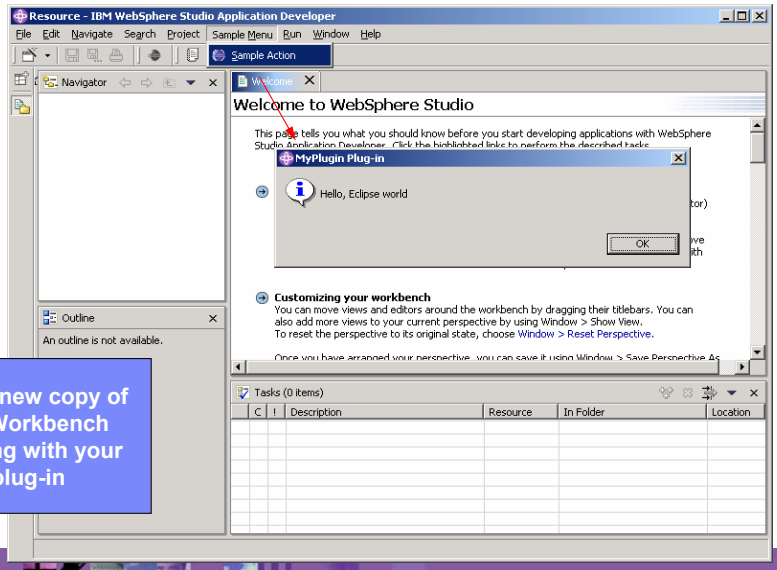
# Testing Your Plug-in




1. Select any plug-in project
2. Use the Run As -> Run-time Workbench menu item to launch a new instance of the Workbench with your plug-in(s) loaded.

- Use Debug-> Run-time Workbench to debug your plug-in
- Set breakpoints in the Java source editor before or after launching the debugger

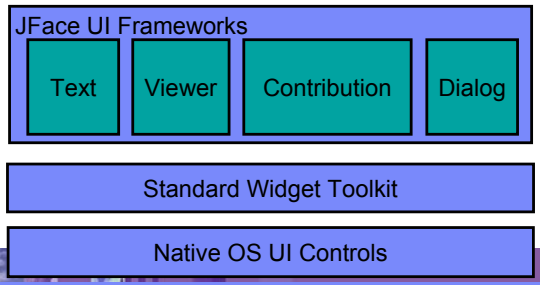
# It Works!




**Brand new copy of the Workbench running with your plug-in**

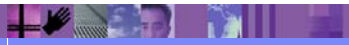
# User Interface

- User Interface for plug-ins should be done using SWT and JFace
  - Provides the best integration with the Workbench
- SWT is a new Java widget toolkit
  - Different than AWT and Swing
- JFace is a UI Framework built on top of SWT for creating wizards, editors, views, dialogs, preference pages, ….



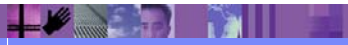
# User Interface

- SWT
  - Fast
  - Native operating system look and feel
  - Provides all the common UI widgets
    - Button, Label, Menu, MenuItem, ProgressBar, ...
- JFace
  - UI frameworks built on top of SWT
    - Provide a higher level of abstraction so you can focus on your added function, not UI details
  - Provide a common look and feel across all Workbench tools
    - Wizards, .., preference pages all look and act the same
    - Provides base set of classes you can extend to add your specific function



# User Interface - JFace

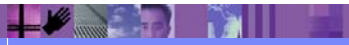
- Text Framework
  - Framework for providing source (text) editors
  - Provides support for
    - Content assist, syntax highlighting, formatting, ...
- Viewer Framework
  - Different types of views into a data model
  - TreeViewer, CheckboxTreeViewer, TableViewer, ...
- Contribution Framework
  - Framework for adding actions to the workbench toolbar and menubar
- Dialog Framework
  - Framework for creating dialogs that interact with the user
  - Example classes:
    - PreferenceDialog, PropertyDialog, WizardDialog, ...
  - Also provides pages for using inside the dialogs
    - PreferencePage, PropertyPage, WizardPage, ...



# Other Things to Consider

# HELP!!! – Lots of ways to provide help

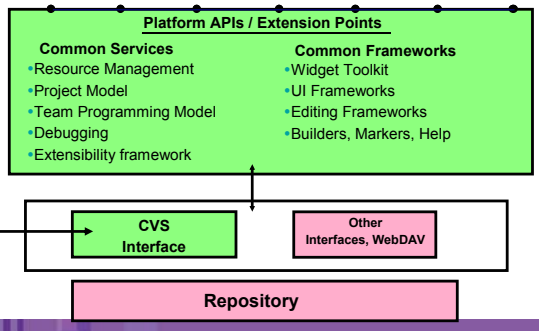
- Tooltip help
  - Also known as "fly-over help"
  - Tooltip is shown when user hovers the mouse pointer over a menu item, button, entry field, ...
  - Use the *setToolTipText(String text)* method on a widget
- F1 key help
  - Context sensitive help
  - Typically used to provide a short description of the current task when the user presses F1 in a view, dialog, wizard, entry field
  - Set context ids using *WorkbenchHelp.setHelp(...)*
  - Eclipse then looks up the context id in HelpContexts.xml
- Full blown online help
  - You can provide complete online documentation integrated into the *Help* perspective
  - This is full topic on its own



# Team Extension Points

- Version / Configuration Management
  - Allows source management providers to plug into the workbench
  - Can be used with by any project in the Workspace without any additional work
    - Java projects, Web projects, iSeries projects, …

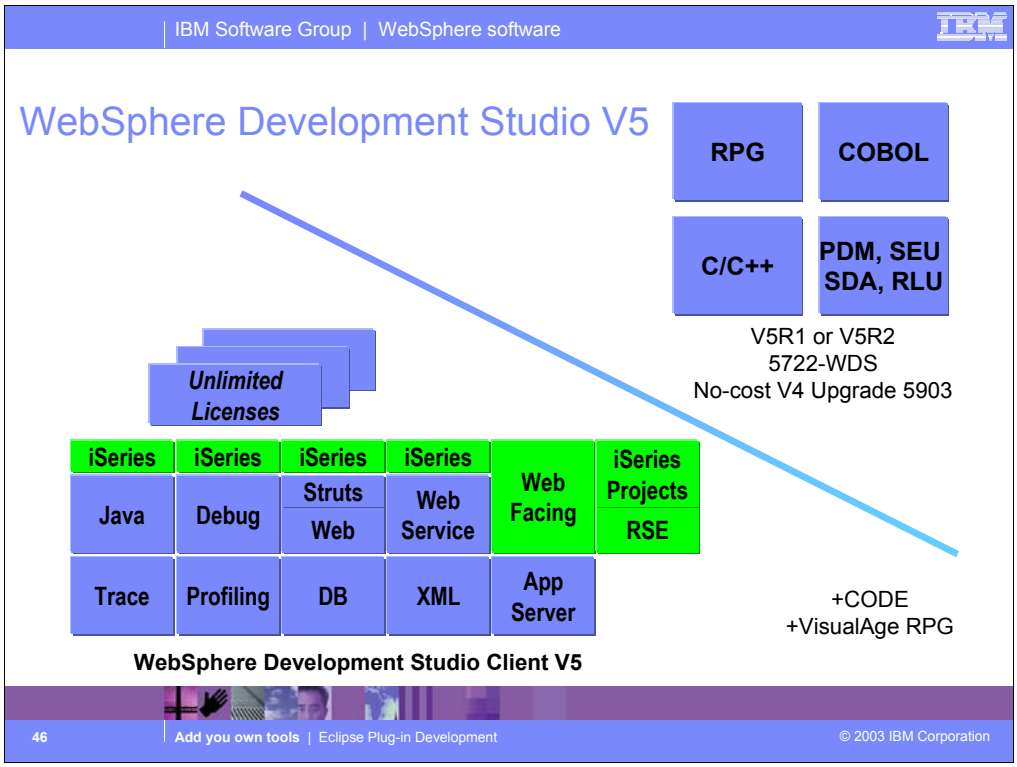
Open source CVS implementation of the VCM APIs is provided as a reference implementation



# What about the iSeries?

WebSphere. software

e business software




There is now only one application development product sold by IBM, for iSeries, as of V4R5. This is WebSphere Development Studio (Development Studio), which includes all four host compilers, all traditional tools (ADTS = PDM+SEU+SDA+RLU+DFU+AFP+CGU), and unlimited licenses of the workstation-based toolset named WebSphere Development Studio Client (formerly WebSphere Development Tools).

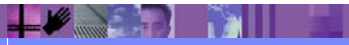
If you are an existing customer who has a subscription, you can upgrade to Development Studio free of charge. Without a Software Subscription, there is an upgrade fee. New licenses of Development Studio are priced very competitive compared to the combined prices of all constituent products. As of V5R1, there is no way to purchase the compilers or tools individually. So if you have RPG at V5R1 or higher, you must have Development Studio and hence are entitled to Development Studio Client.

For consultants who do not have an iSeries of their own, but still wish to have the client tools, Development Studio Client is also made available as a passport advantage product so it can be purchased "off the shelf" from IBM Direct.

Development Studio has been a huge success, with over 80,000 licenses sold. Just as every development machine used to have PDM and SEU, every development machine will now have all the modern Application Development tools from IBM. This ubiquity is especially important for business partners who build and sell software. These Business Partners are now free to build software using any of the technologies or tools in Development Studio, and can assume their customers will have the tools required to tailor everything from RPG to Java and Web user interfaces. This effectively raises the lowest common denominator to a level unparalleled by any other operating system.

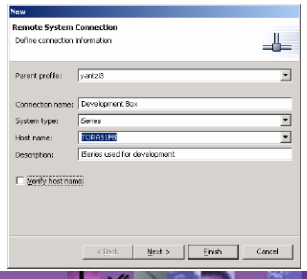
# RPG and COBOL Tools

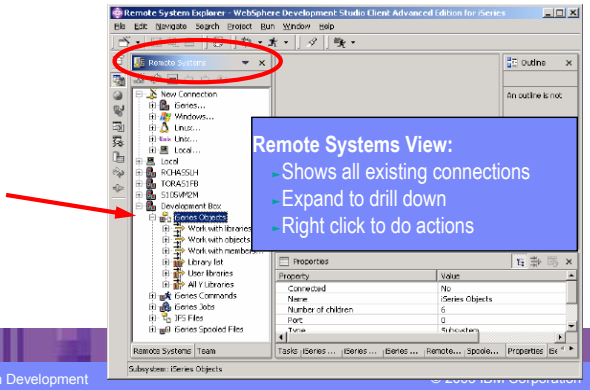
- **Remote Systems Explorer**
  - For random drill-down or filtered access to remote:
    - Library, objects and members in QSYS
    - IFS Folders and Files
    - Linux (including LPAR), Unix and Windows Folders and Files
    - Local Folders and Files
  - For remote command execution of
    - QSYS Commands
    - QShell Commands
    - Linux, Unix and Windows Commands
  - For exploration of Jobs in QSYS
- **iSeries Projects and Navigator**
  - For structured, project-based development
  - Local team-sharable project
  - Holds source files and source members and save files
  - Push action to copy local files to associated library



# Remote System Explorer Perspective

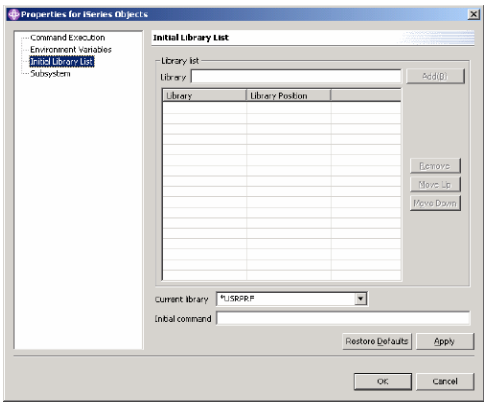
- **Manages connections to remote systems**
  - Where you create and manage connection information
  - Connections subsequently used in all iSeries tools
  - Start by using the New Connection wizard
    - Connection appears in tree view
    - Expand it to see "subsystem" -> functional groupings



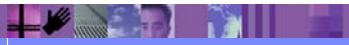
# iSeries Connection Properties

- Use Properties of iSeries Commands to set:
  - Library list information
  - Current library
  - Initial command to call
  - Environment variables
- All information is applied at connect time



# iSeries Connection Subsystems

- Expand a connection in the to see subsystems:
  - **iSeries Objects**
  - **iSeries Commands**
  - **iSeries Jobs**
  - **IFS Files**






**iSeries Objects**

Like WRKXXXPDM. For drilling down on libraries, objects, mbrs

Right click offers wealth of actions like create, rename, delete...

Can edit source members using built-in LPEX or classic CODE

Can create User Defined Actions just like in PDM

**iSeries Commands**

For creating command sets containing one or more CL commands

Right click and select Run to remote run commands

Results are logged in Command console

**iSeries Jobs**

For seeing jobs, filtered by number, user ID or status

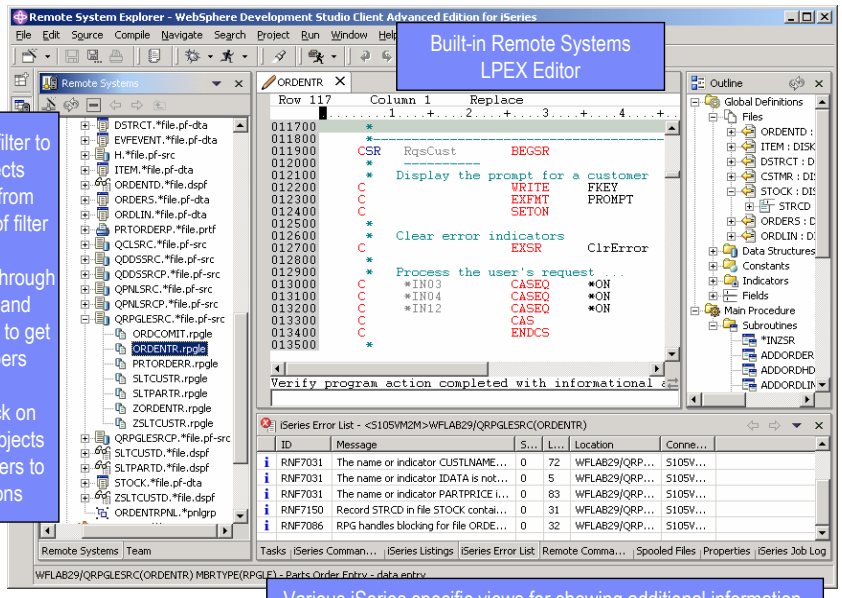
Right click to do common actions on jobs, such as end or hold

**IFS Files**

For drilling down on IFS folders and files

Right click offers wealth of actions like create, rename, delete...

Can edit source members using built-in LPEX or classic CODE



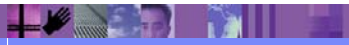

Built-in Remote Systems LPEX Editor

- Expand a filter to see objects resulting from resolution of filter
- Drill down through libraries, and source files to get to members
- Right click on libraries, objects and members to get actions

Various iSeries specific views for showing additional information. iSeries Error List shows compile / verify errors

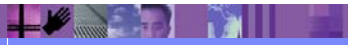
# RSE: Filters and Filter Pools

- Expand a subsystem to see "filters"
  - Like WRKXXXPDM generic name filters but
    - Are named and saved
  - There are some predefined
    - Such as Library List to see *LIBL libraries in iSeries Objects
    - Such as popular commands in iSeries Commands
  - To create your own, right click on subsystem object
  - Each filter can contain multiple filter strings
    - So you can list all libraries that start with A *and* B
- Eventually, you'll have too many filters
  - Time to turn on "Show Filter Pools" via preferences
    - Expanding subsystems will then first show filter pools
  - Filters are grouped into named pools
    - By default they are added to the single default filter pool
  - You can create your own filter pools
    - Then add filters to it
  - Expand a filter pool to see just the filters in it



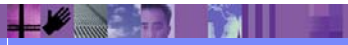
# iSeries Objects Subsystem Filters

- **Library List filter**
  - Displays library list
  - Provides popup actions for working with the library list
- **Library filters**
  - Specify simple, generic or special library names
    - Lists all matching libraries, when expanded
- **Object filters**
  - Specify simple or generic object names, library-qualified
    - Library name can be simple, generic or special
  - Specify simple or generic object types and attributes
    - Can specify one or more type:attribute pairs (OR operation)
    - Lists all objects that meet criteria, when expanded
- **Member filters**
  - Specify simple or generic member names, lib/file-qualified
    - Library name and file name can be simple or generic
  - Specify simple or generic member types
    - Can specify one or more member types (OR operation)
    - Lists all members that meet criteria, when expanded

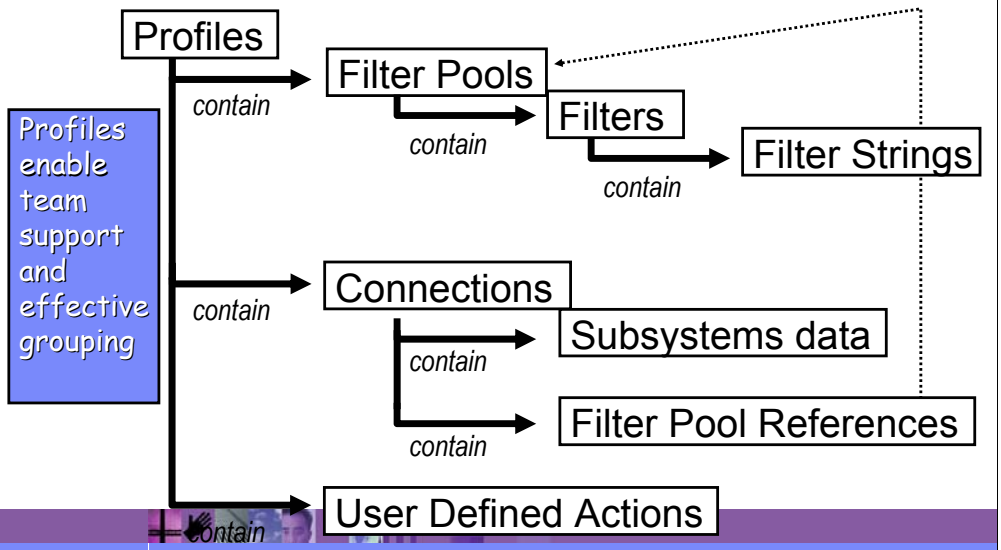


# Remote Systems LPEX Editor

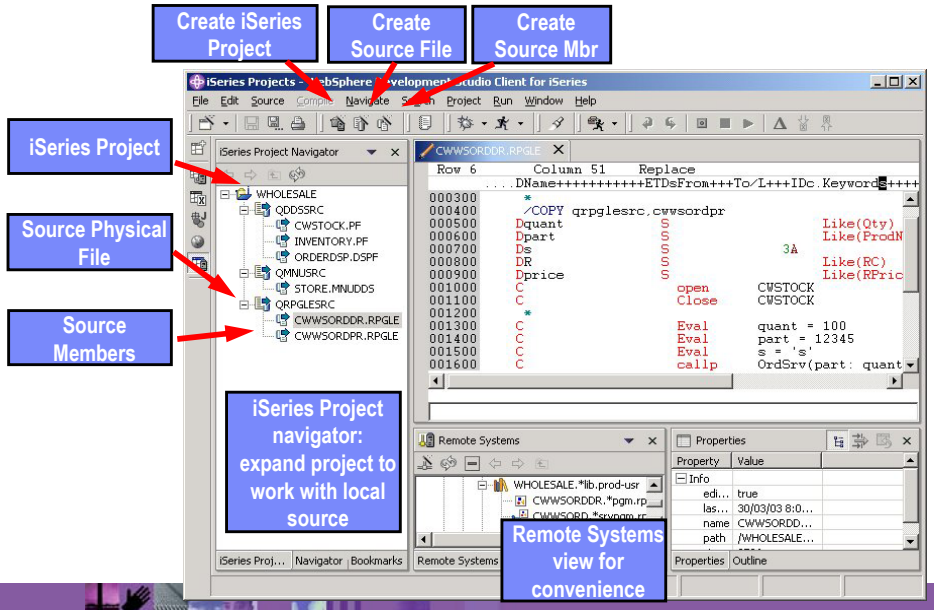
- Remote Systems LPEX Editor
  - Re-write of CODE Editor for Eclipse
- What it has today:
  - All base support from CODE Editor, such as:
    - Alt+L/C/M/D/U/S/J to select/copy/move/delete/unselect/split/join
    - Select streams, lines, blocks and rectangles (Edit menu)
    - Command-line for directly entering editor commands
    - Keystroke recording for repetitive tasks
    - Compare utility
    - Rich search, find and replace
    - Bookmarks and quick marks
    - Ctrl+L to locate a line
    - Line-number prefix area
    - File->Get File to import a file
    - Filter Selection to only show lines containing selected text
  - Plus!
    - Outline view support for ILE RPG and COBOL
    - Content assist for ILE RPG



# Underlying Model of the RSE



# iSeries Projects Perspective

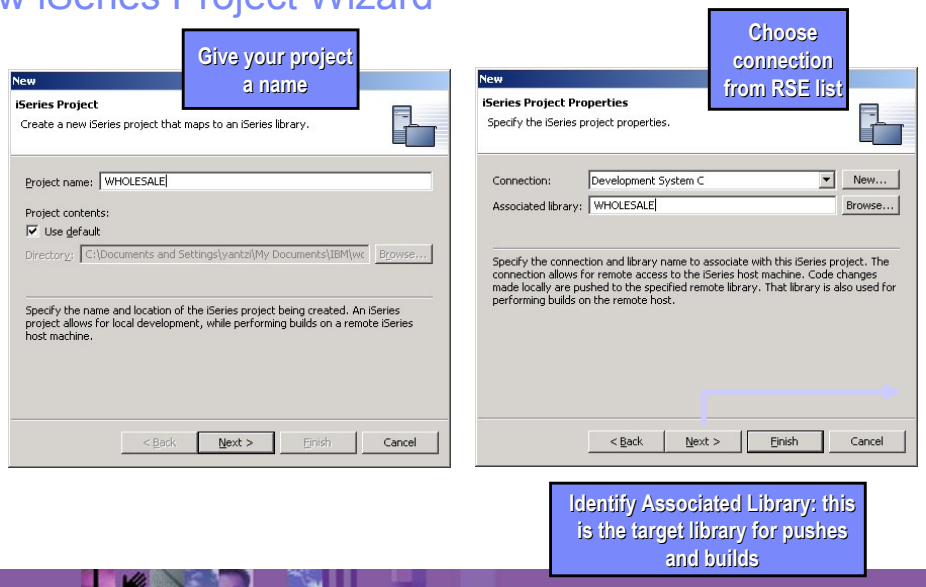






Here we see the iSeries Projects perspective, which you can explicitly open. The primary view is the iSeries Navigator that allows exploration of all existing iSeries Projects. There are wizards lauchable from the toolbar for creating a new iSeries Project, or creating source physical files and source members within an existing project. The iSeries Projects perspective includes a full copy of the Remote Systems Explorer, as a view, for your convenience. This allows you to easily work with the contents of the associated library while simultaneously working with the local copies of the source within the project.

# New iSeries Project Wizard

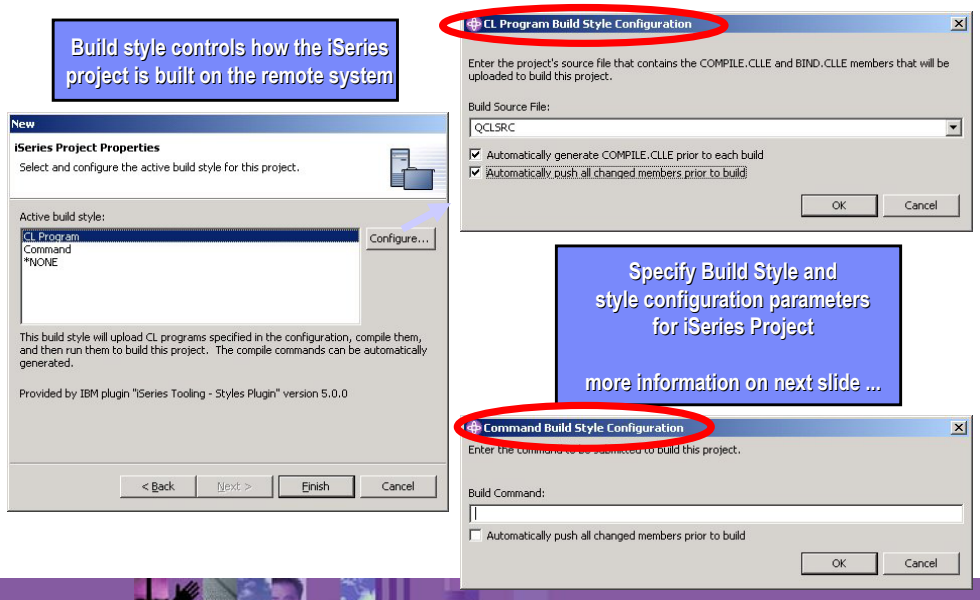






Here we see the rest of the iSeries Project wizard. First, give the project a name. Any name you want! Then select or create a connection (from the RSE!) that identifies the iSeries with which this project is associated. Also select the library on that iSeries where the contents of this project will be pushed to. Also, specify a CL command to call for the "Build" action. If you don't have such a CL command, enter a dummy value or leave it blank.

# iSeries Project Wizard







Here we see the rest of the iSeries Project wizard. First, give the project a name. Any name you want! Then select or create a connection (from the RSE!) that identifies the iSeries with which this project is associated. Also select the library on that iSeries where the contents of this project will be pushed to. Also, specify a CL command to call for the "Build" action. If you don't have such a CL command, enter a dummy value or leave it blank.

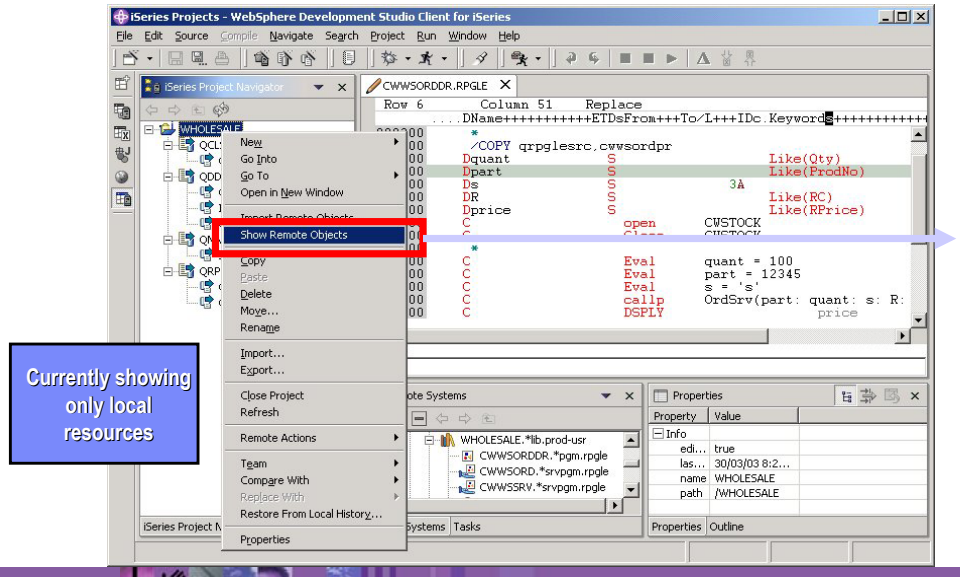
# iSeries Projects Build Styles

- Build Styles
  - Specify how to build the iSeries project on the remote system
  - Three IBM-supplied build styles
  - **CL Program**
    - Automatically generates COMPILE.CLLE and BIND.CLLE
      - Adds compile command for each source file in project
      - Uses last compile command from RSE for specified source type
        - i.e. CRTBNDRPG versus CRTRPGMOD
      - At build time these two members are pushed to iSeries, compiled and run
  - **Command**
    - You specify single command to run build
    - Good if you have an existing build script
  - ***NONE**
    - No build support
- ISVs can plug-in additional build styles
  - To work with their existing iSeries systems




Here we see the wizard for creating a new iSeries source physical file. Within the project, this is actually a folder. When the project is pushed to its associated library, this result in a CRTSRCPF command being run to create a file with the attributes specified in this wizard.

# Project "Bleedthrough"



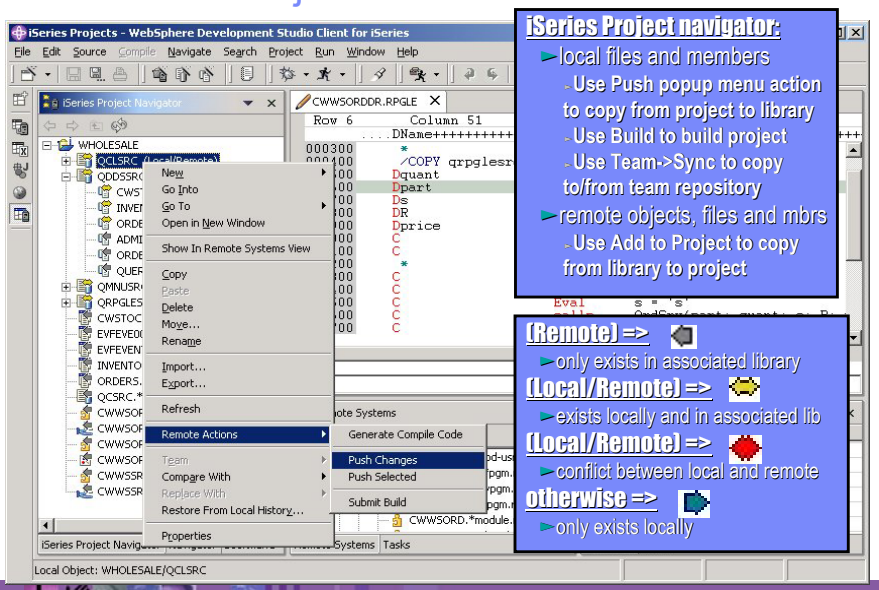




Here we see an iSeries project in the iSeries Navigator, within the iSeries Projects perspective, that has a number of local files and members.

Right-clicking on the project or anything in the project, gives a popup menu with the all-important "Show Remote Objects" menu item....

# Local and Remote Objects



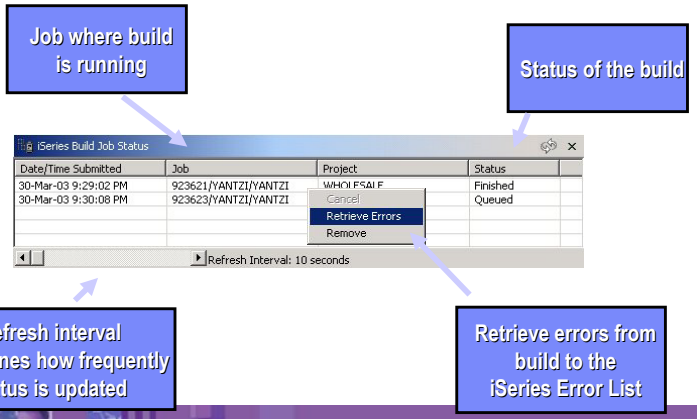
Once "Show Remote Objects" is enabled (it is a toggle) all the objects in the associated library for this project are shown in the iSeries Navigator.

This navigator is called a "bleed through" view because it lists both local files/members and remote objects, files and members. However, if a file or member exists both locally and remotely, it is not shown twice. Rather, it is only shown once, and its icon and bracketed text identify it as existing in both places.

The icons and text enable you to see at a glance which files/members exist only locally, only remotely, or both locally and remotely. In the latter case, the color of the icon indicates if the two are in synch or not. If not, it is an indication you need to "push" your local source to the associated library to get it in synch.

# iSeries Projects Remote Build

- Submit Build
  - Submits build of iSeries project on remote system
  - Adds build job to iSeries Build Job Status view







Once "Show Remote Objects" is enabled (it is a toggle) all the objects in the associated library for this project are shown in the iSeries Navigator.

This navigator is called a "bleed through" view because it lists both local files/members and remote objects, files and members. However, if a file or member exists both locally and remotely, it is not shown twice. Rather, it is only shown once, and its icon and bracketed text identify it as existing in both places.

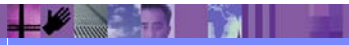
The icons and text enable you to see at a glance which files/members exist only locally, only remotely, or both locally and remotely. In the latter case, the color of the icon indicates if the two are in synch or not. If not, it is an indication you need to "push" your local source to the associated library to get it in synch.


Additional documentation on the following extension points can be found in:

C:\WDSC\iseries\eclipse\plugins\com.ibm.etools.systems.core_5.0.0\plugin.xml

# Benefits of Building On Top of the RSE

- What does the RSE offer you?
  - **Extension points for remote objects**
    - Popup menus
    - Property pages
  - **Extend the remote system framework**
    - Add a new subsystem
    - Add a new system type
      - iSeries, Linux, Unix, Windows and Local are pre-supplied
  - **Java APIs for accessing remote systems**
    - Native
      - Retrieve lists of libraries, objects, members and jobs based on filters
      - Retrieve properties for a library, object, member or job
      - Prompt and run commands
      - Run applications in batch or interactive
    - IFS / Windows / Linux / Unix
      - Retrieve lists of directories / files
      - Run remote commands



# Adding a New RSE System Type

- **Extension point**
  com.ibm.etools.systems.core.systemtype
- **Adds a new system type to the Remote Systems Framework**
  The new system type will appear in the new connection wizard
  No Java code required!
  You can then add new subsystems for this type

```
<extension point="com.ibm.etools.systems.core.systemtype">
   <type name="Windows"
         iconlive="icons/windowsLive.gif">
         icon="icons/windows.gif">
    </type>
</extension>
```

**icon changes when system is connected or disconnected**

# Adding New Subsystems

- **Extension point**
  - com.ibm.etools.systems.core.subsystemfactory
    - A SubSystemFactory creates instances of a specific type of subsystem
- **Adds a new subsystem**
  - New subsystem will appear under connections of specified type

```
<extension point="com.ibm.etools.systems.core.subsystemfactory">
   <factory
      systemtypes="iSeries"
      name="Files"
      icon="icons/filesubsys.gif"
      iconlive="icons/filesubsyslive.gif"
      id="com.acme.etools.mypkg.myfactory"
      class="com.acme.etools.mypkg.MyClass"
      category="files"
      vendor="acme">
   </factory>
</extension>
```

**Class must implement the com.ibm.etools.systems.subsystems.SubSystemFactory interface**

# Adding Property Pages for Remote Objects

- **Extension point**

  com.ibm.etools.systems.core.propertyPages

- **Add property pages for remote objects**

  Property pages will appear in the properties dialog when user selects "Properties..." from the popup menu for a remote object

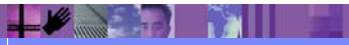
```
<extension point="com.ibm.etools.systems.core.propertyPages">
   <page class="com.acme.myPropertyPage"
      id="com.acme.mypropertypage"
      name="Remote File Info"
      icon="icons/myProperties.gif"
      subsystemfactoryid="ibm.filesLocal"
      typecategoryfilter="files"
      typefilter="file"
      subtypefilter="java"
      namefilter="*.java">
   </page>
</extension>
```

- Filters are subsystem specific
- Each remote object can have a type, subtype and subsubtype
- For example
  - Type: *PGM
  - Subtype: RPGLE
- Can also filter by name

# Adding Popup Menu Actions for Remote Objects

- **Extension point**
  com.ibm.etools.systems.core.popupMenus
- **Add action for remote objects**
  New action will appear in the popup menu of remote objects in the Remote Systems view

Example →



# Adding Popup Menu Actions for Remote Objects

```
<extension point="com.ibm.etools.systems.core.popupMenus">
    <objectContribution
        id="com.acme.displayfile.actions"
        subsystemfactoryid="ibm.files400"
        typecategoryfilter="OBJECT*"
        typefilter="*FILE"
        subtypefilter="DSPF"
        namefilter="*">
        <menu id="com.acme.displayfile.menu
            label="My Display File Actions">
            <separator name="group1"/>
        </menu>
        <action id="com.acme.action1"
            label="My first action"
            class="com.acme.actions.Action1"
            menubarPath="com.acme.displayfile.menu/group1"
            enablesFor="1">
        </action>
        <action id="com.acme.action2"
            label="My second action"
            class="com.acme.actions.Action2"
            menubarPath="com.acme.displayfile.menu/group1"
            enablesFor="+">
        </action>
    </objectContribution>
</extension>
```

Same as property pages

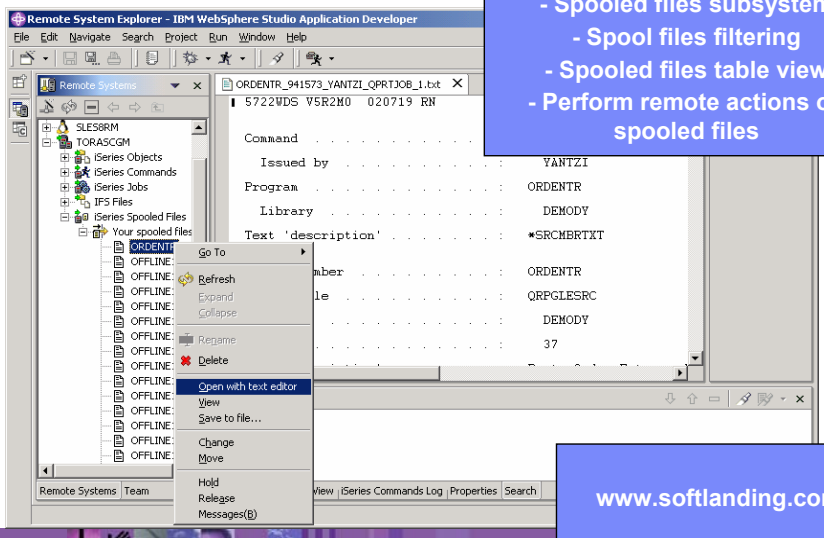
Adds the menu item

Your actions

# Examples from Business Partners

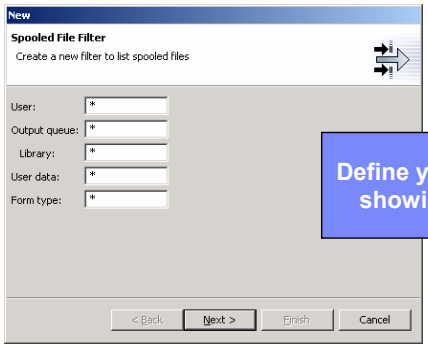
# SoftLanding Systems, Inc.

**Free Spooled File Plug-in for WDSC, includes:**
- Spooled files subsystem
- Spool files filtering
- Spooled files table view
- Perform remote actions on spooled files

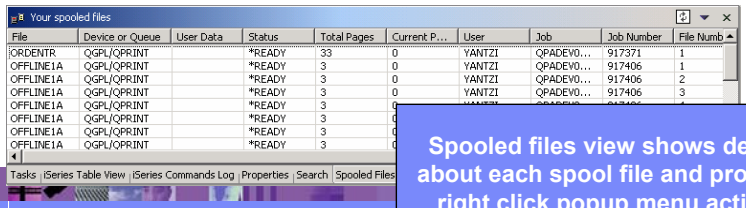



**www.softlanding.com**

# SoftLanding Systems, Inc.




Define your own filters for showing spooled files




Spooled files view shows details about each spool file and provides right click popup menu actions

IBM Software Group

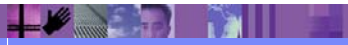
# Plug-in Development Summary

WebSphere. software

e business software

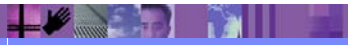
# Summary

- Eclipse and the WebSphere Studio Workbench provide:
  - **An integrated development environment**
  - **Extendable platform**
    - You can focus on your function without worrying about:
      - Resource management
      - Help system
    - Rich set of base classes for quickly developing UI
- WDSc provides:
  - iSeries tooling plugged into Eclipse
  - Remote Systems Framework
    - With extension points so you can add functionality to the RSE



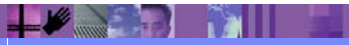
# Additional Resources

- Eclipse
  www.eclipse.org
- WebSphere Development Studio Client
  http://www.ibm.com/software/awdtools/iseries//
- IBM's developerWorks
  www.ibm.com/developer



# Disclaimer

- **Acknowledgement:**
  This presentation is a collaborative effort of the IBM Toronto AS/400 Application Development presentation team, including work done by:
  Phil Coulthard, George Farr, Claus Weiss, Don Yantzi, John Steinbacher, David Slater, Alison Butteril, Linda Cole
- **Disclaimer:**
  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 customers' ability to evaluate and integrate them into the customers' 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 environment do so at their own risk.
- **Reproduction:**
  The base presentation is the property of IBM Corporation. Permission must be obtained PRIOR to making copies of this material for any reason.



IBM Software Group

# Add your own tools to IBM WebSphere Development Studio Client for iSeries: Eclipse Plug-in Development

WebSphere. software

e business software