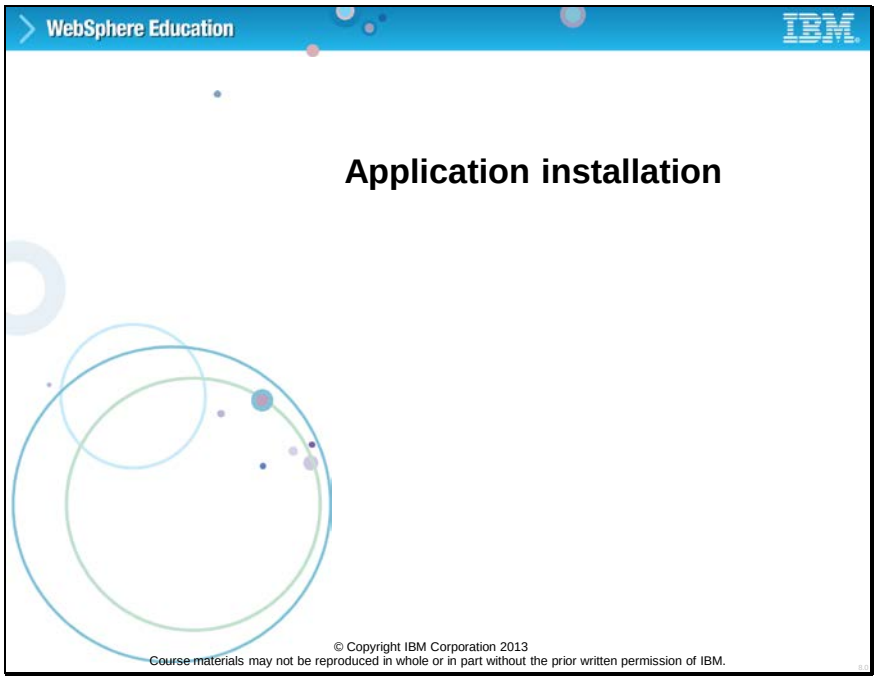
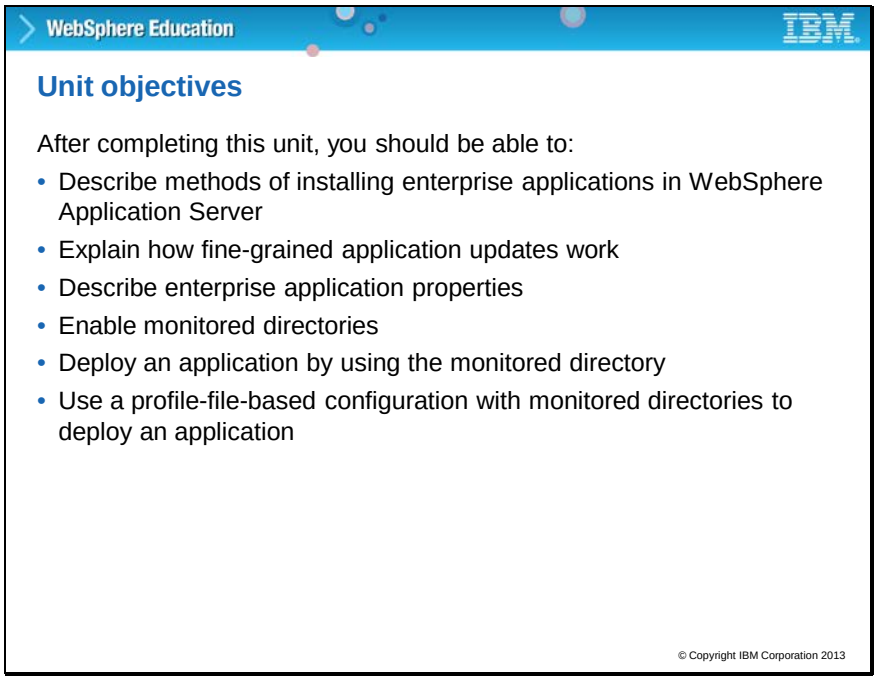
Slide 1



Application installation. This unit describes how to install enterprise applications into WebSphere Application Server.

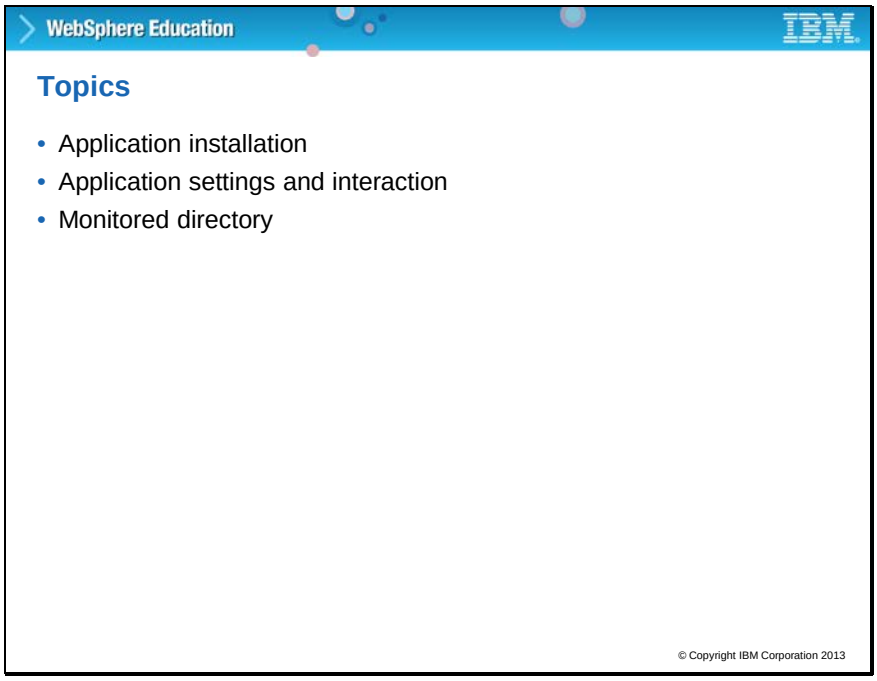
Slide 2



After completing this unit, you should be able to:

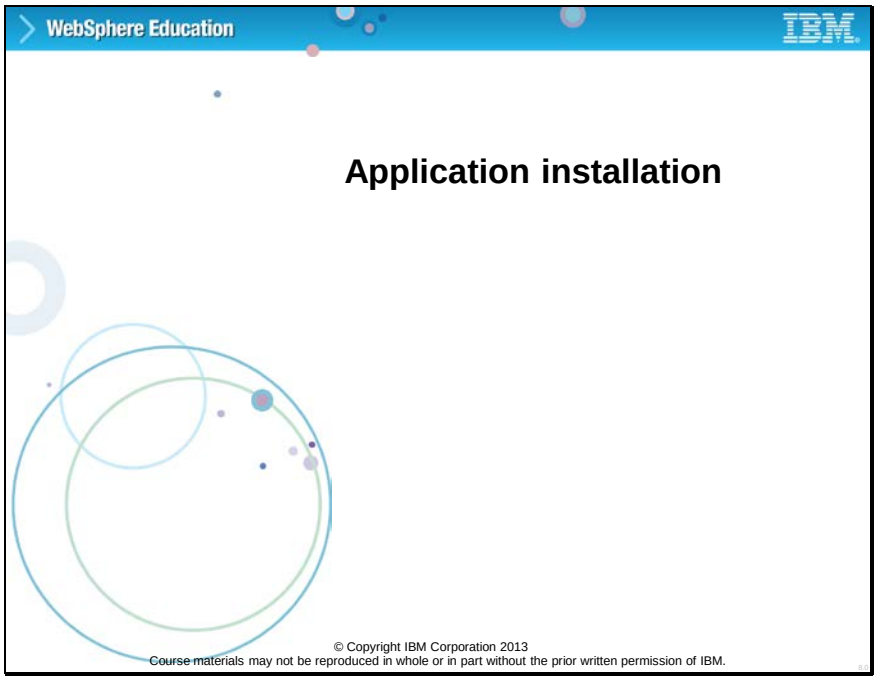
- Describe methods of installing enterprise applications in WebSphere Application Server
- Explain how fine-grained application updates work
- Describe enterprise application properties
- Enable monitored directory
- Deploy an application by using the monitored directory
- Use a property-file-based configuration with monitored directories to deploy an application

Slide 3



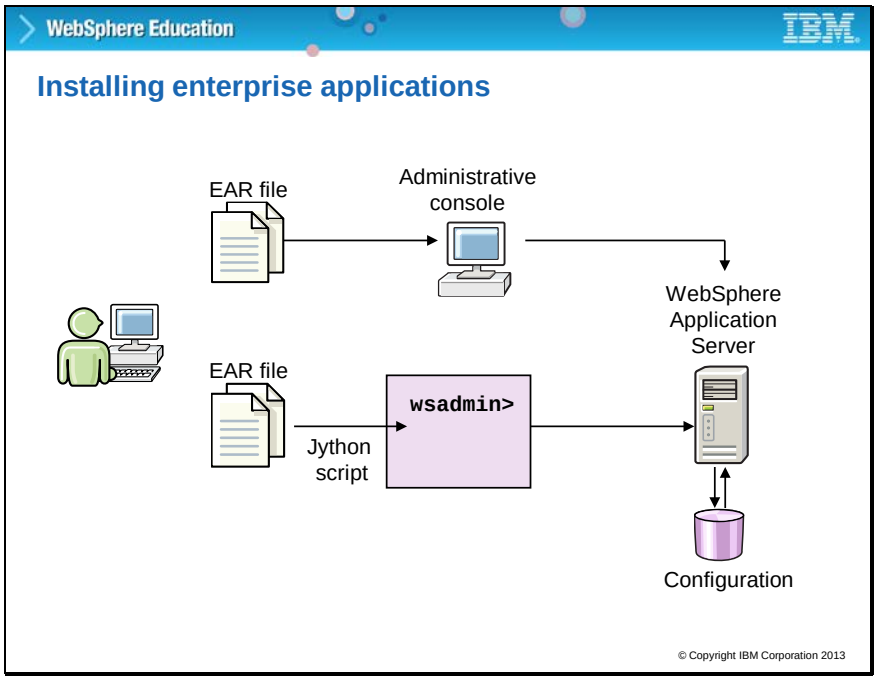
This unit is divided into three topics.

Slide 4



Topic: Application installation. In this topic, you learn the various methods of installing an application.

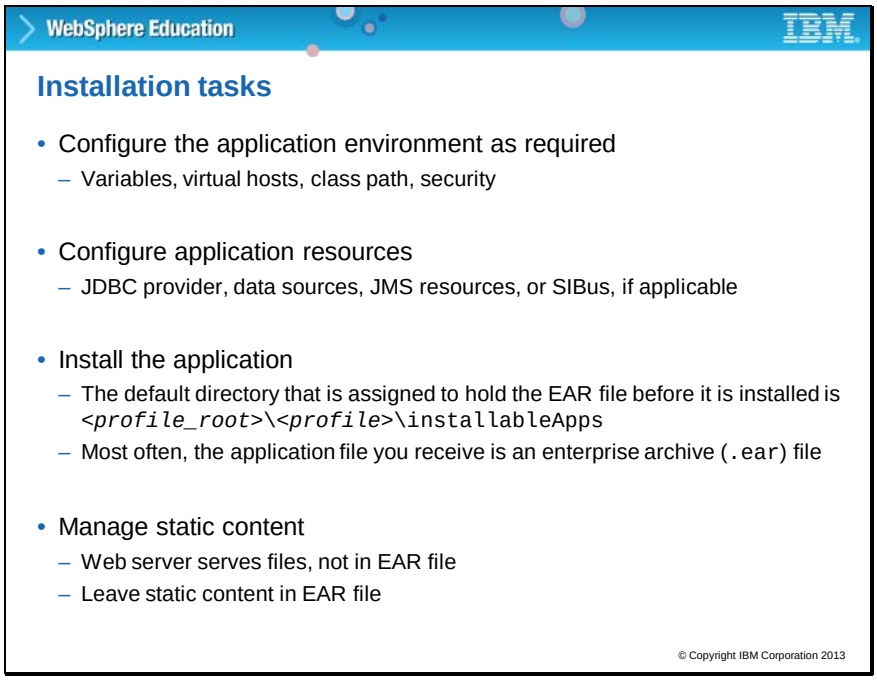
Slide 5



The two most common ways to install enterprise applications on the server are to use the administrative console or wsadmin scripts. The administrative console is the easier method of installing or updating an application. Wizards take you through the process of installing an application and provide helpful information at each step.

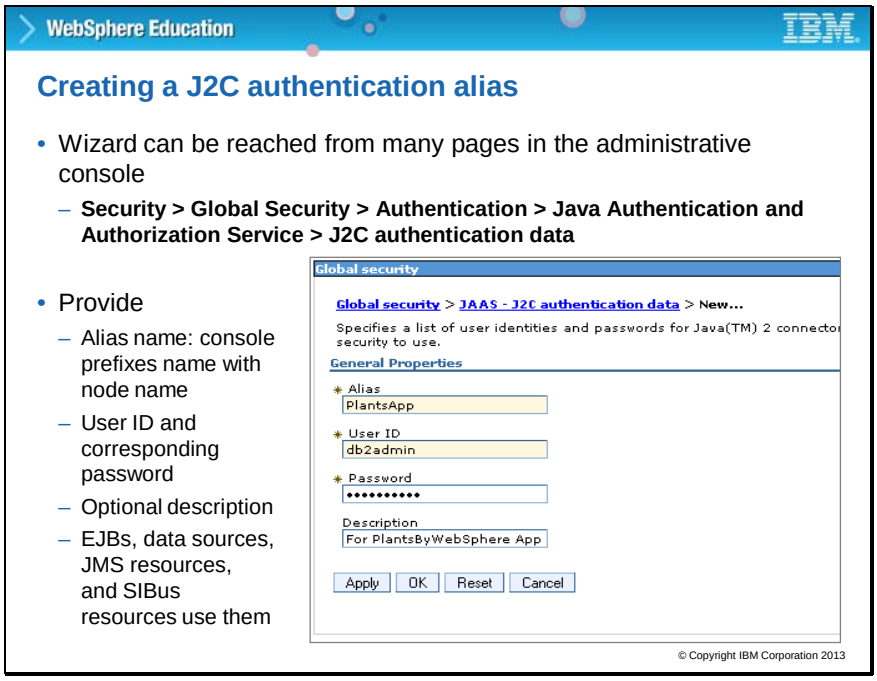
Using wsadmin scripts is more complicated than other methods. It requires skills in at least one of the supported scripting languages, and requires complete understanding of application configuration.

Slide 6



Installing an application involves configuring the runtime environment as required. You can define variables, virtual, hosts, and any other resources that the application needs before you actually install the application. In general, it is a good practice to leave the static content that the application uses in the EAR file and allow the infrastructure to take care of serving and caching the static content. In the end, leaving the static content in the EAR file can be as effective as moving it to the web server.

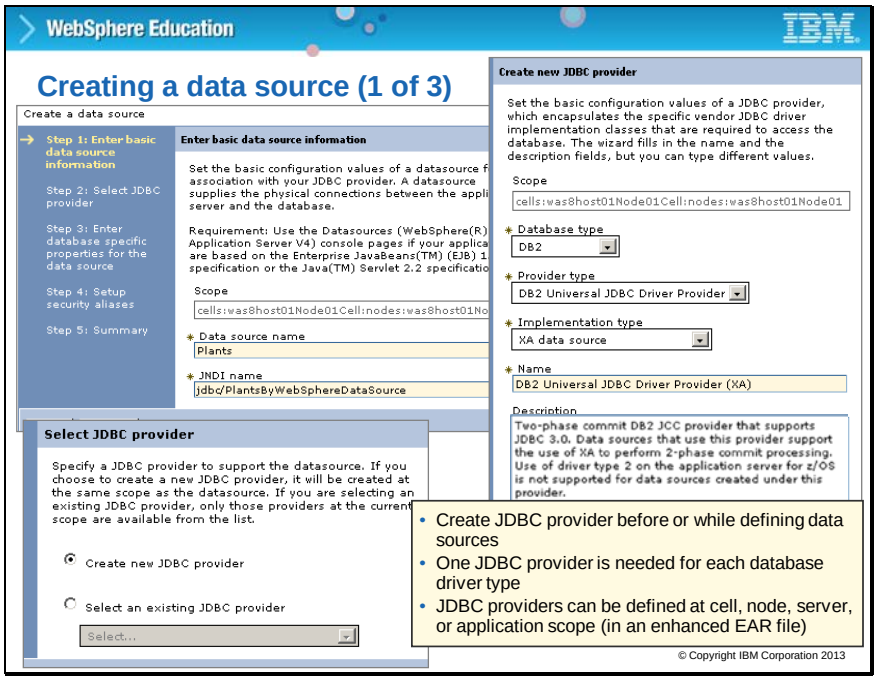
Slide 7



In this scenario, a J2C authentication alias is created for accessing resources that the application uses. You can access the page to create new J2C aliases from most pages in the console where an authentication alias is required. To create a J2C alias, provide a name, user ID, and password.

J2C aliases can also be created through scripting. J2C aliases are not scoped resources, so when installing an enhanced EAR file that has a J2C alias definition within it, the alias is created and available to any resource in the server. The administrative console adds the node name to the alias name you provide.

Slide 8

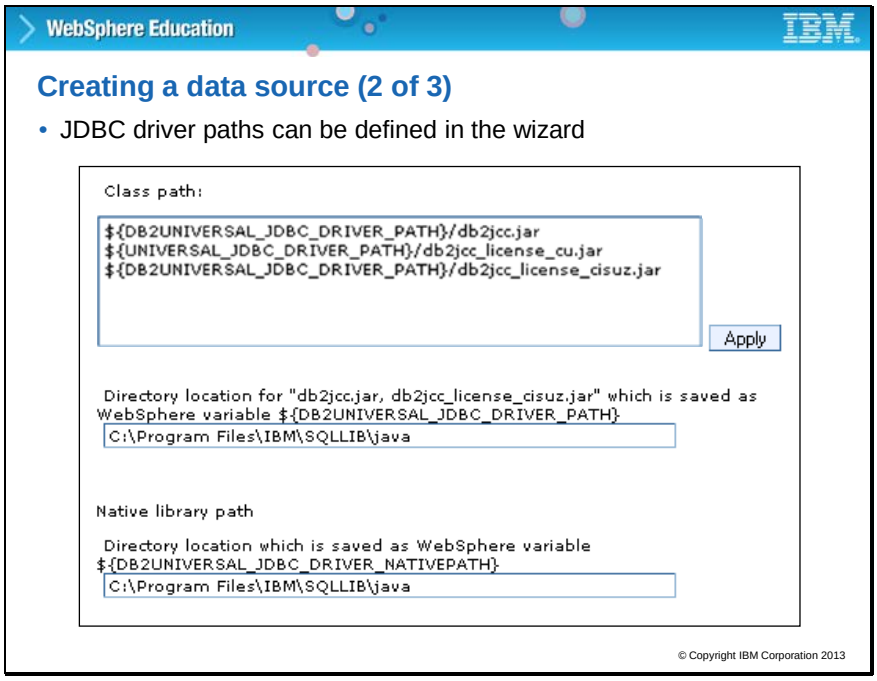


These screen captures show the process for creating the required resources that an application uses to access a database. A data source is associated with a JDBC provider that supplies the specific JDBC driver implementation class. The data source represents the Java EE Connector Architecture (JCA) connection factory for the relational resource adapter.

Application components use the data source to access connection instances to a specific database. A connection pool is associated with each data source. You can create multiple data sources with different settings and associate them with the same JDBC provider. One reason to create multiple data sources with different settings is to provide access to different databases. This series of slides shows you how to create a data source. You can create the JDBC provider for the data source either before or while defining the data source. They can be defined in the data source wizard. One JDBC provider is needed for each database driver type used.

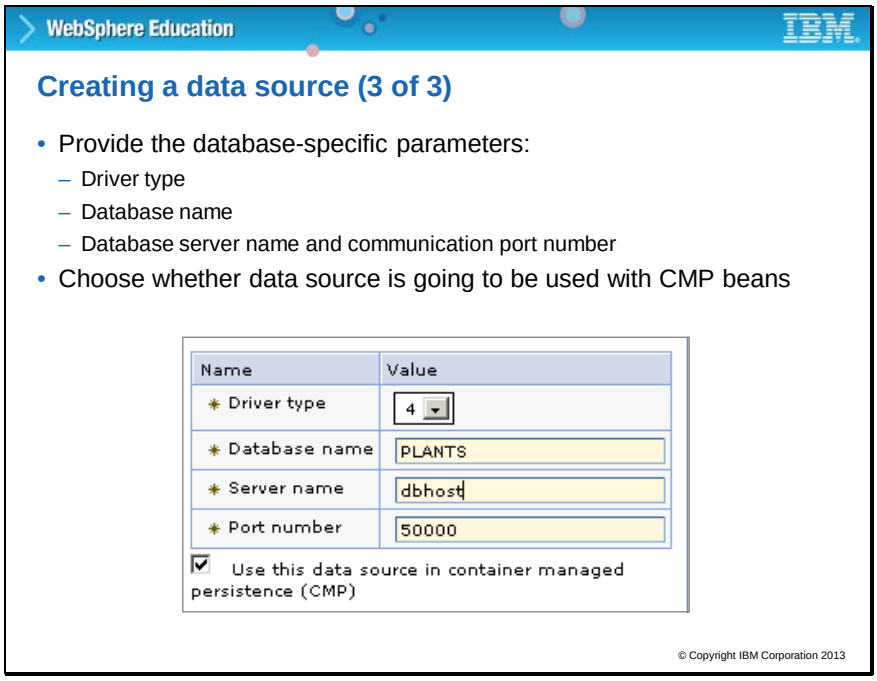
A JDBC provider is a scoped resource that can be defined at the cell, node, or server level, or at the application scope if included in an enhanced EAR file. The data source creation wizard is shown here. After entering a name for the data source, you are prompted to create or select an existing JDBC provider. If you are creating a JDBC provider, you specify that information here.

Slide 9



On this page, you specify the class path, the location of the JDBC driver, and the native library path. The paths are stored in WebSphere variables that the server uses to locate the JDBC driver code.

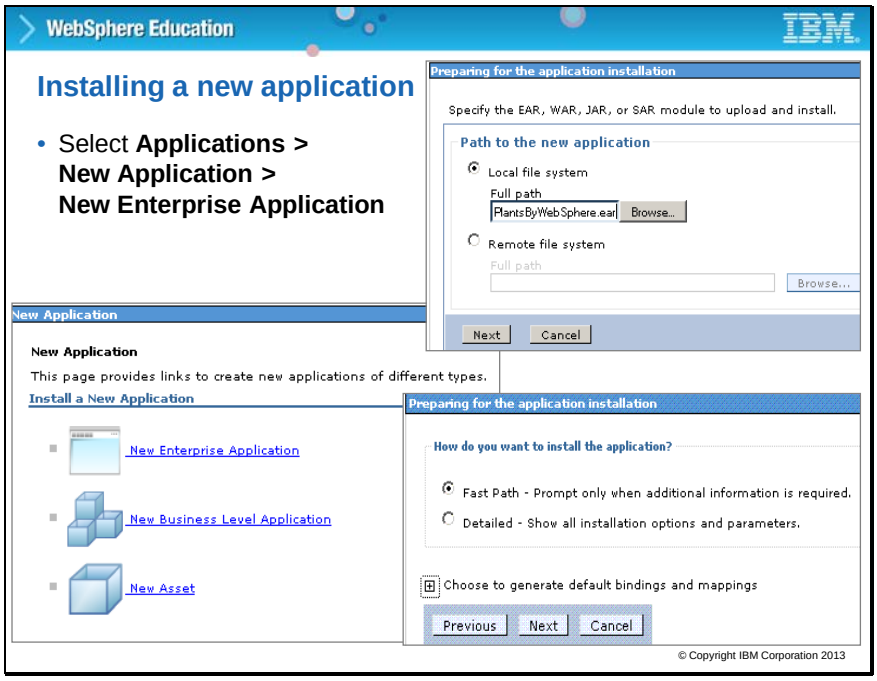
Slide 10



As the database protocol is vendor-specific, you must use separate JDBC drivers that are vendor-supplied to connect to the database. You specify the driver type, database name, server name, and port on this screen.

Type 4 drivers are suggested if available. Type 4 drivers are pure Java drivers that translate JDBC calls to a local database protocol. One advantage of type 4 drivers is that they do not require more database client code on the application server.

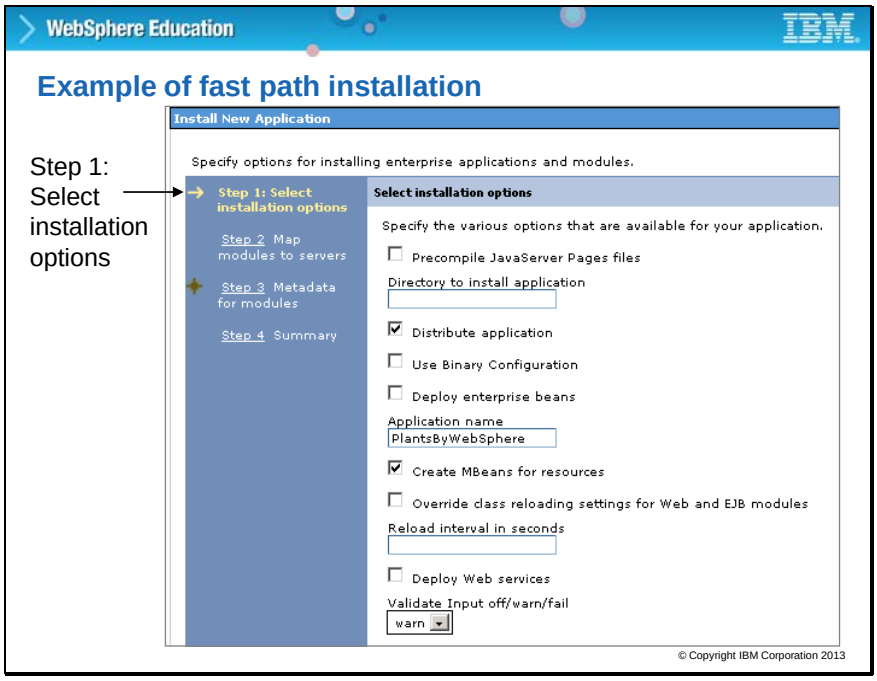
Slide 11



Here are the screen captures for the simple method of installing an application.

To start the application installation wizard, select **Applications > New application > New enterprise application**. Specify the path to the EAR file on the local or remote file system and click Next. You can choose the fast path, which prompts you only when more information is required, or you can choose the detailed path where all of the installation options are shown. You can also choose to generate default bindings and mappings.

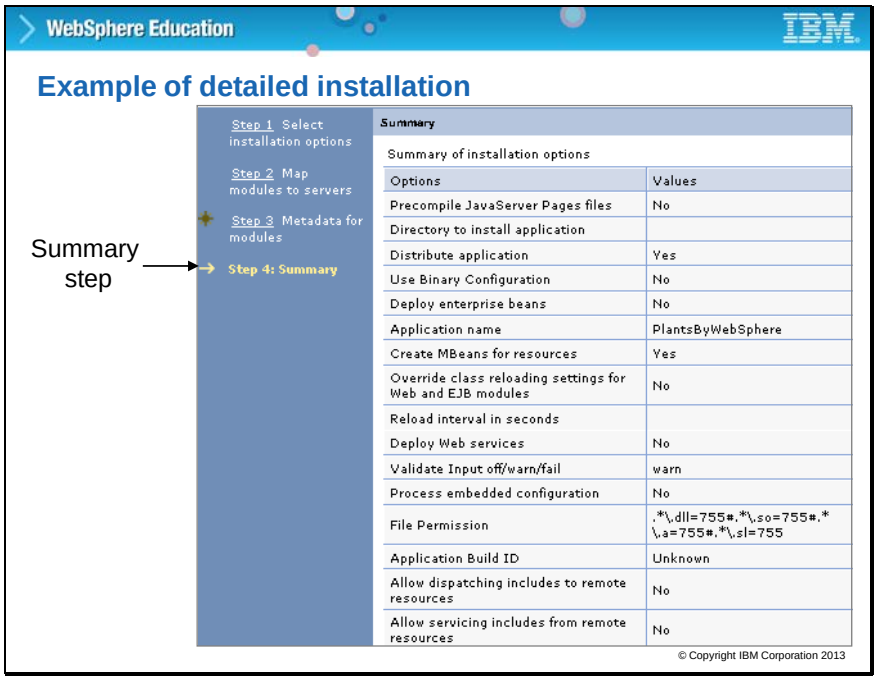
Slide 12



This slide displays an example of what the options look like if you choose the fast path for installing an application. Notice that few steps are listed in the left column. You can skip ahead through these steps by clicking the links. Many options are presented on this page. For example, you can select the box to precompile JSP pages. You can select the box to process embedded configuration settings, which creates the application-scoped resources that are defined in an enhanced EAR file. You can also choose to ignore these settings by leaving that box clear.

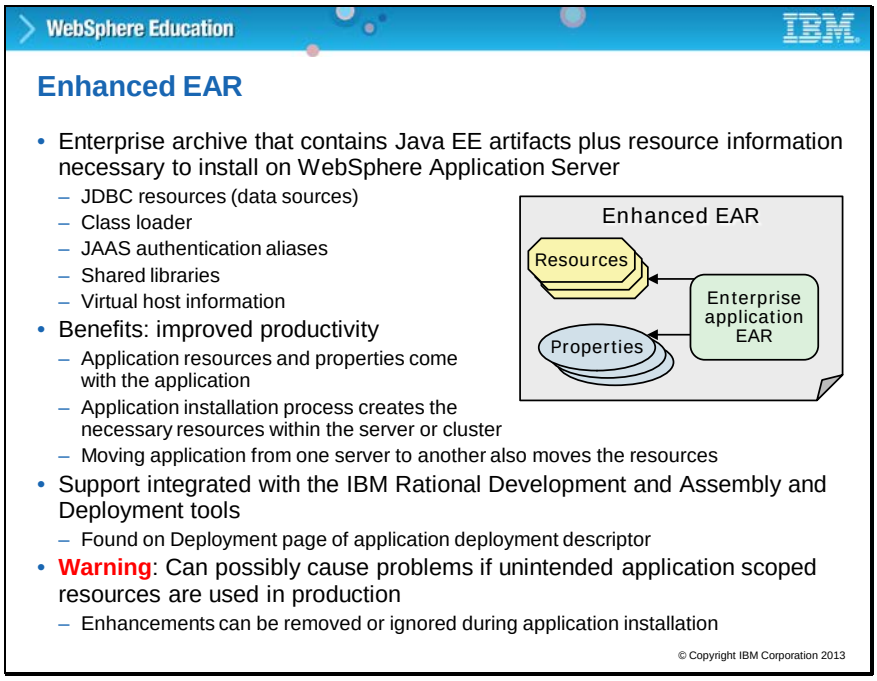
Each module of a deployed application must be mapped to one or more target servers. The target server can be an application server, a cluster of application servers, or a web server.

Slide 13



This screen capture shows the summary of information just before the application is installed. It is not important that you read all of the options on this slide, just that you understand the purpose of this step.

Slide 14

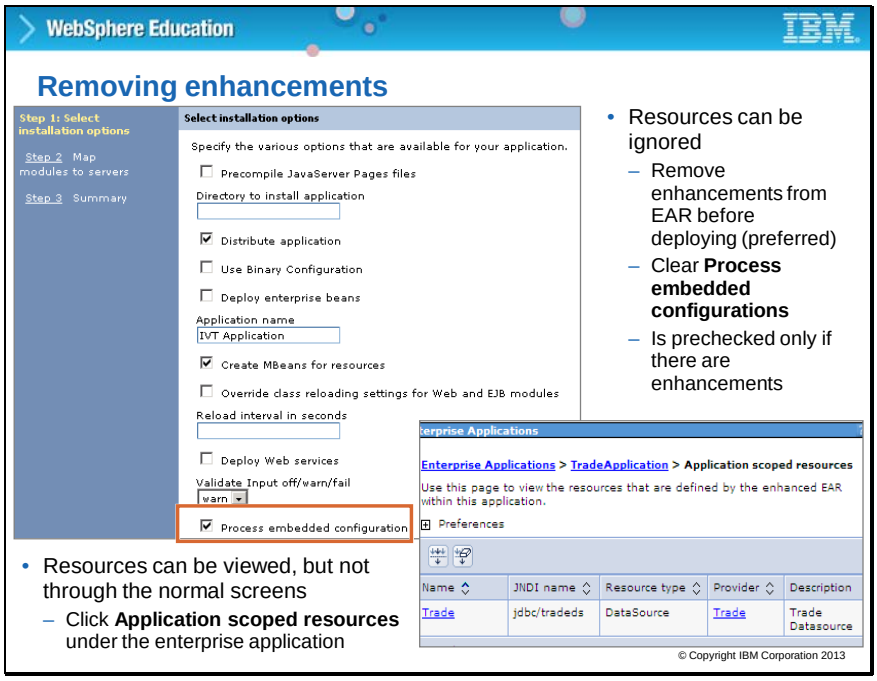


Application-scoped resources are resources that are scoped at the application level, rather than at the cell, node, or server level. An enhanced EAR file contains a simple embedded configuration archive. An enhanced EAR is where you can define resources for an application; for example, JDBC resources, class loaders, virtual hosts, and other items, so that all the information needed to deploy the application is included. In this way, you can deploy an application and its required resource definitions in one shot, and if you move the application, the resource definitions move with it. The benefit is that it can make deploying applications easier. However, it can also cause problems if the application-scoped resources do not coincide with the intended runtime environment. In this case, application scoped resources can be removed before deployment, or ignored during application installation, and the needed resources can be defined at the appropriate scope.

Enhanced EAR files are specific to IBM WebSphere and are not part of Java EE. However, artifacts that are defined within an enhanced EAR are ignored if this EAR file is installed on an application server other than WebSphere.

Developers or administrators can define resources and properties within an enterprise application in tools and import or export the enhanced EAR file. Settings are defined in IBM tools and are stored in the deployment.xml file, which is packaged with the EAR file. Some resources still must be defined in the application server, for example, JMS and JavaMail.

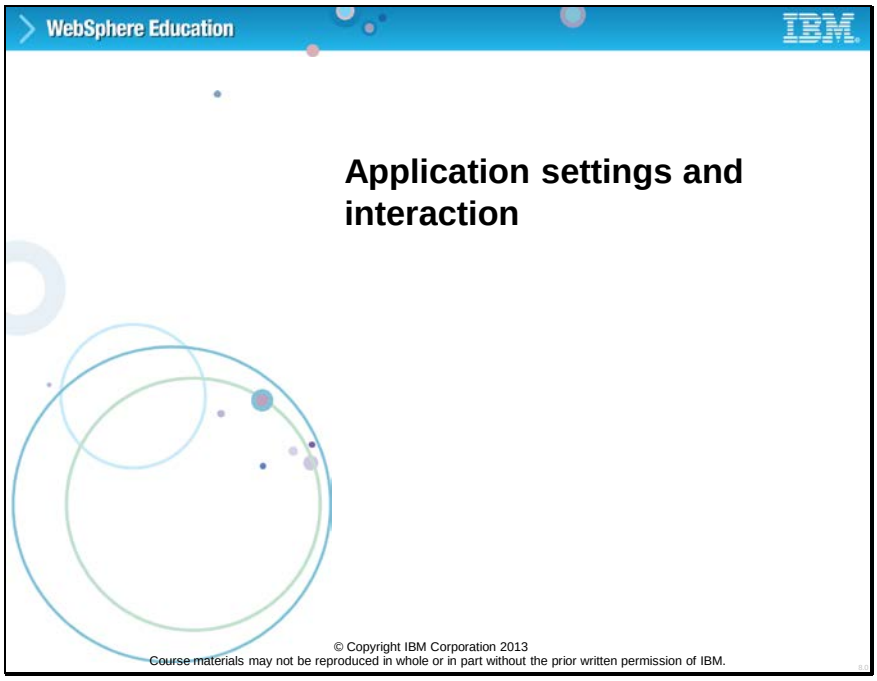
Slide 15



There are several methods to ignore resources in an enhanced EAR file. The preferred method is to remove enhancements from the EAR before deploying. Here is a screen capture of the administrative console for installing an application where you can choose to ignore application-scoped resources. If the box Process embedded configuration is clear, the application-scoped resources are ignored. By default, the box is selected if the application EAR file is an enhanced EAR file.

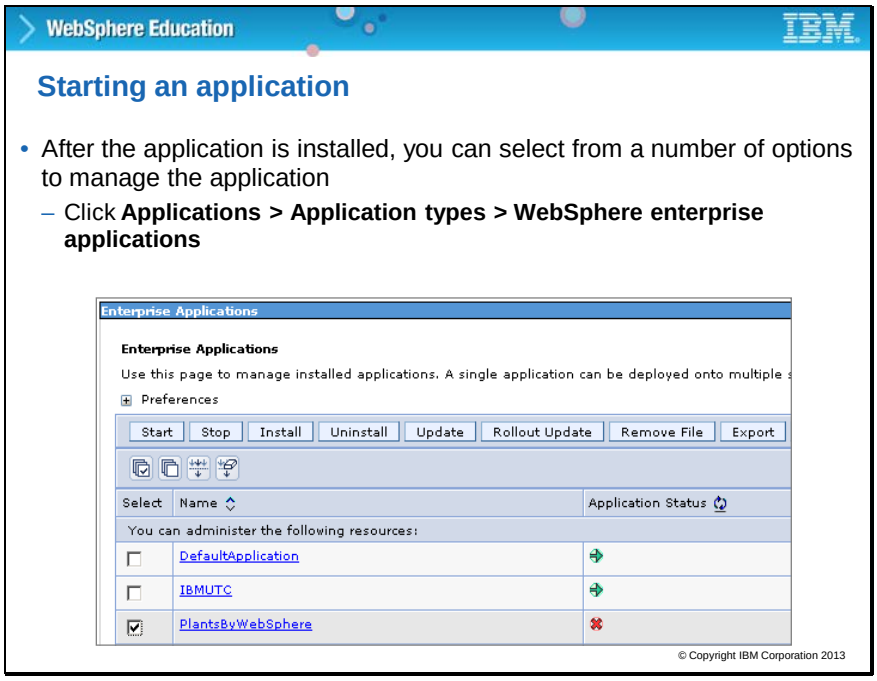
You can also view the application-scoped resources for installed applications in the console under **Enterprise applications > application name > application scoped resources**.

Slide 16



Topic: Application settings and interaction. In this topic, you learn about more application settings and how you can interact with an installed application.

Slide 17



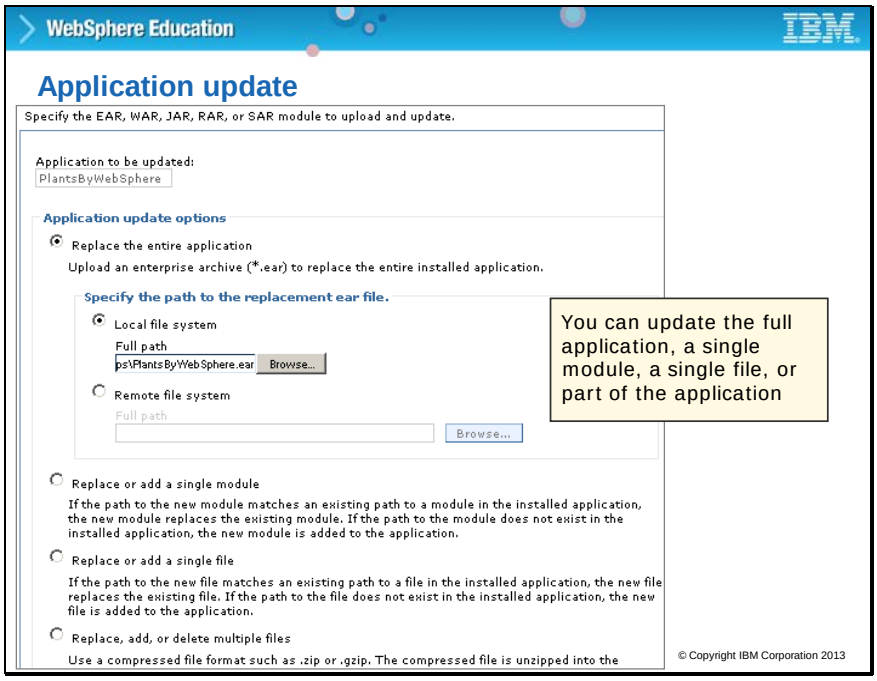
After the application is installed, you can see it listed on the Enterprise Applications page by going to **Applications > Applications types > WebSphere enterprise applications**.

To start the application you installed, select the box beside it, and click **Start**. The application status symbol changes from a red X to a green arrow.

The other actions available are:

- Stop - Stops the selected application.
- Install - Displays the application installation page.
- Uninstall - Deletes a deployed application from the WebSphere Application Server configuration repository. It also deletes application binary files from the file system.
- Update - Replaces an application that is deployed on a server with an updated application. As part of the updating, you can complete steps on the Preparing for application installation and Update application pages.
- Rollout Update - Sequentially updates an application that is installed on multiple cluster members across a cluster.
- Remove File - Removes the specified file from the selected application or module.

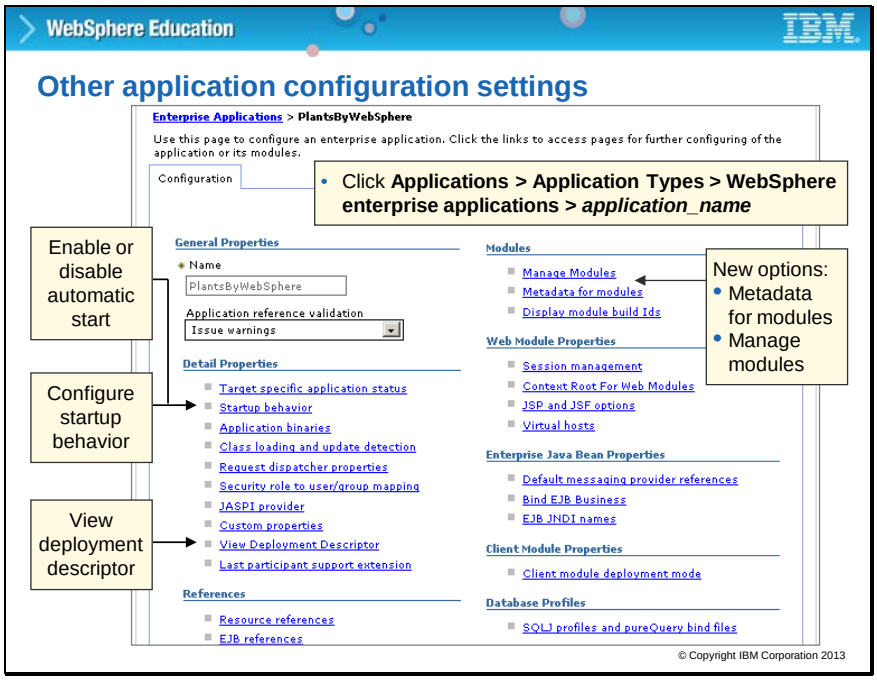
Slide 18



The Update option displays a page that helps you update application files that are deployed in the cell. You can update the full application, a single module, a single file, or part of the application. If a new file or module has the same relative path as a file or module that exists on the server, the new file or module replaces the existing file or module. If the new file or module does not exist on the server, it is added to the deployed application. After replacement of a full application, the product uninstalls the old application. After replacement of a module, file, or partial application, the product removes the old installed module, file, or partial application from the installed application. If a changed application or module is deployed on a cluster, changes are made to all cluster members of the cluster on which the application or module is deployed.

Click **Rollout Update** on the Enterprise applications page to propagate the changed configuration on all members of the cluster on which the application or module is deployed. Rollout update sequentially updates the configuration on the nodes that contain cluster members.

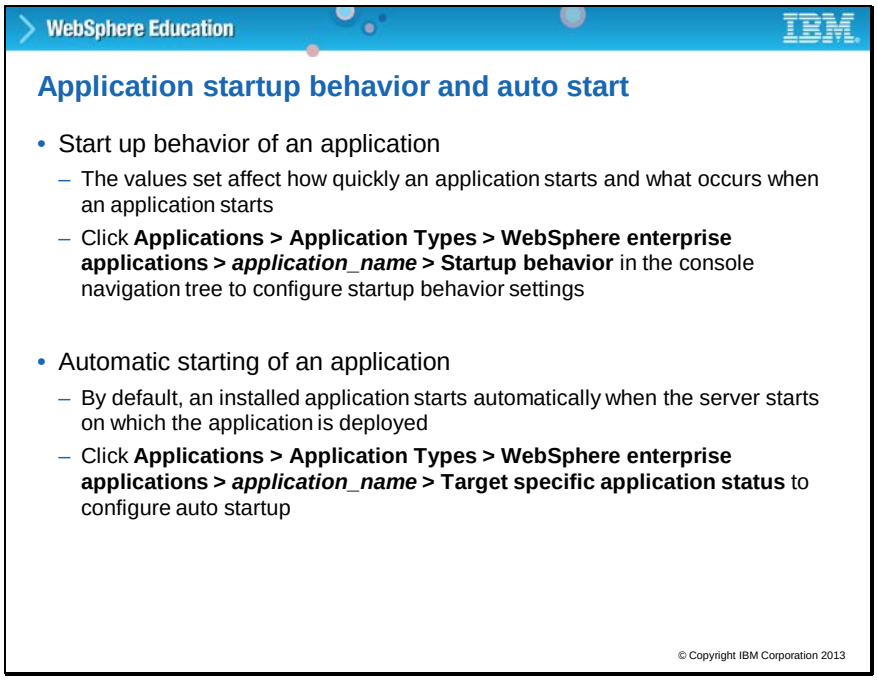
Slide 19



This screen capture shows a broad overview of the metadata for an enterprise application that can be modified or configured through the administrative console.
Click **Applications > Application Types > WebSphere enterprise applications > *application_name*** to change other application settings to get to this page.

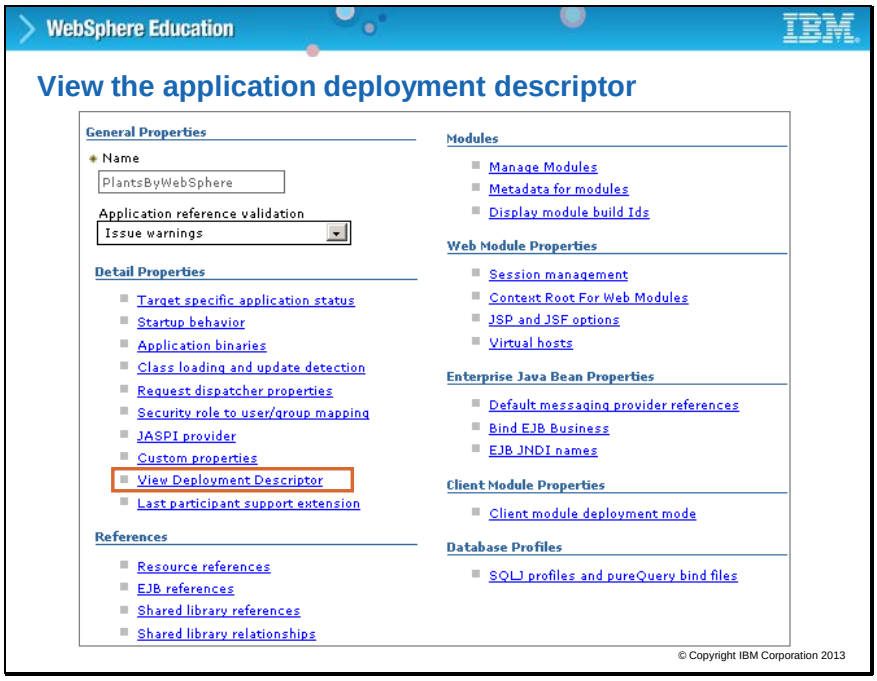
On this page, you can enable or disable automatic start. You can also configure startup behavior, view the deployment descriptor, and initiate other tasks. One of the new features is the capability to view metadata for modules and manage modules for an application.

Slide 20



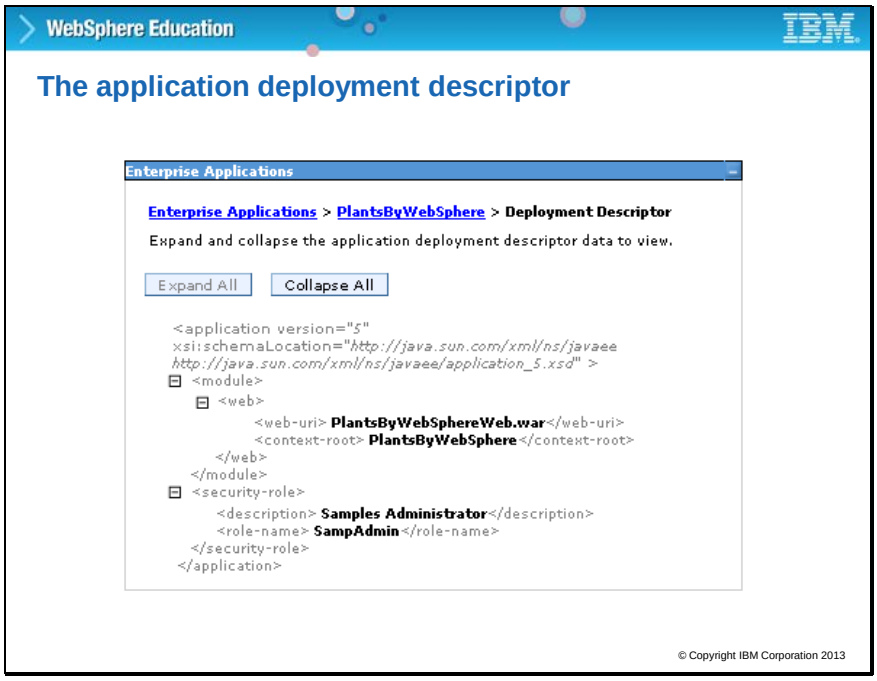
For the startup behavior of an application, the values set affect how quickly an application starts and what occurs when an application starts. By default, an application starts when its parent server starts.

Slide 21



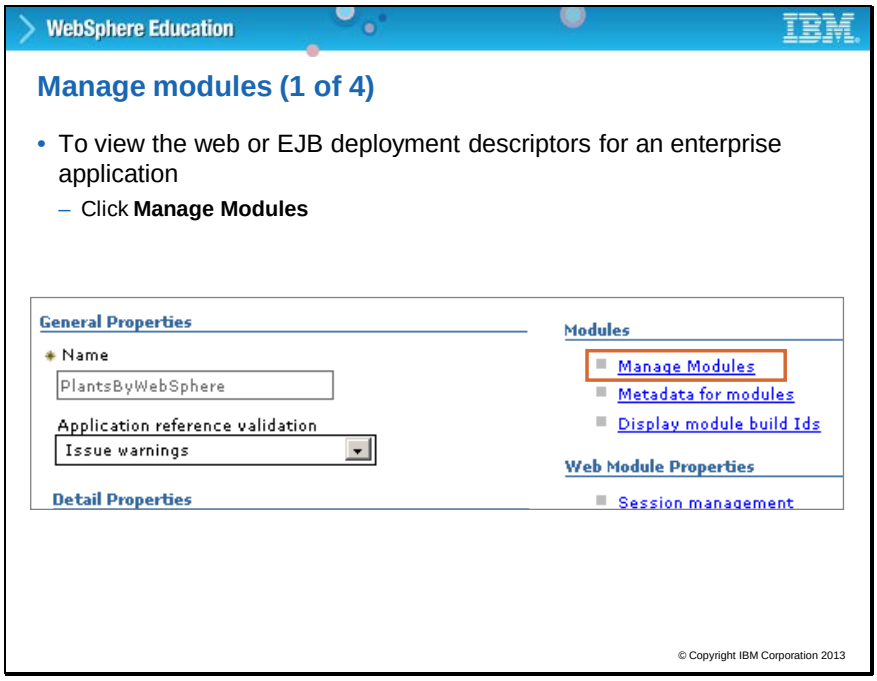
You can view the application.xml deployment descriptor for the installed application from the administrative console. Select **Enterprise applications > *application_name*** and then click **View Deployment Descriptor** under **Detail Properties**.

Slide 22



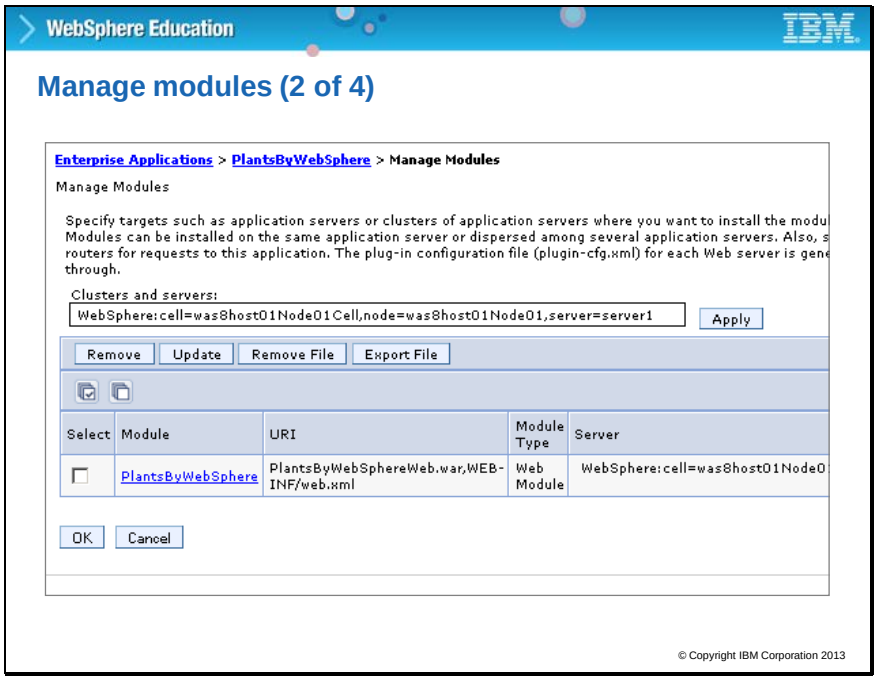
This slide displays the result of selecting View Deployment Descriptor (as seen on the previous slide) for the PlantsByWebSphere application, which is the application deployment descriptor called application.xml. In the example, you can see the context root for the PlantsByWebSphere application. In addition, a security role is seen for this Java Enterprise application.

Slide 23



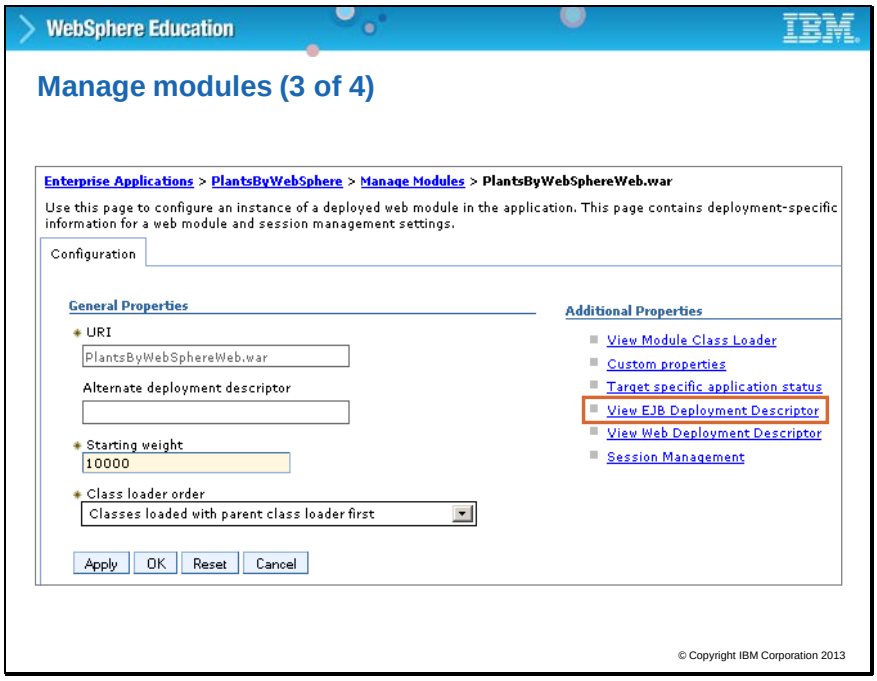
These next four slides show the steps for viewing the deployment descriptor of an EJB JAR file (EJB module). You can view the web or EJB deployment descriptors for the installed application from the administrative console. Select **Enterprise applications > *application_name***; then click **Manage Modules** under **Modules**. In this example, you see the deployment descriptors for the PlantsByWebSphere enterprise application.

Slide 24



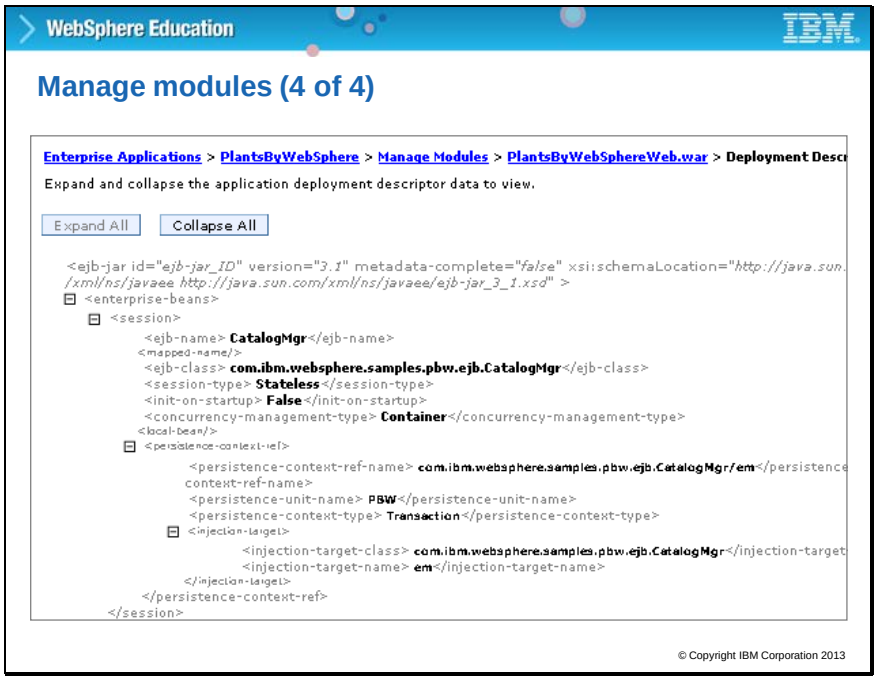
This screen capture shows that the PlantsByWebSphere application has one web module and no EJB module. Click the **module_name** link under the Module column to display the properties of the module.

Slide 25



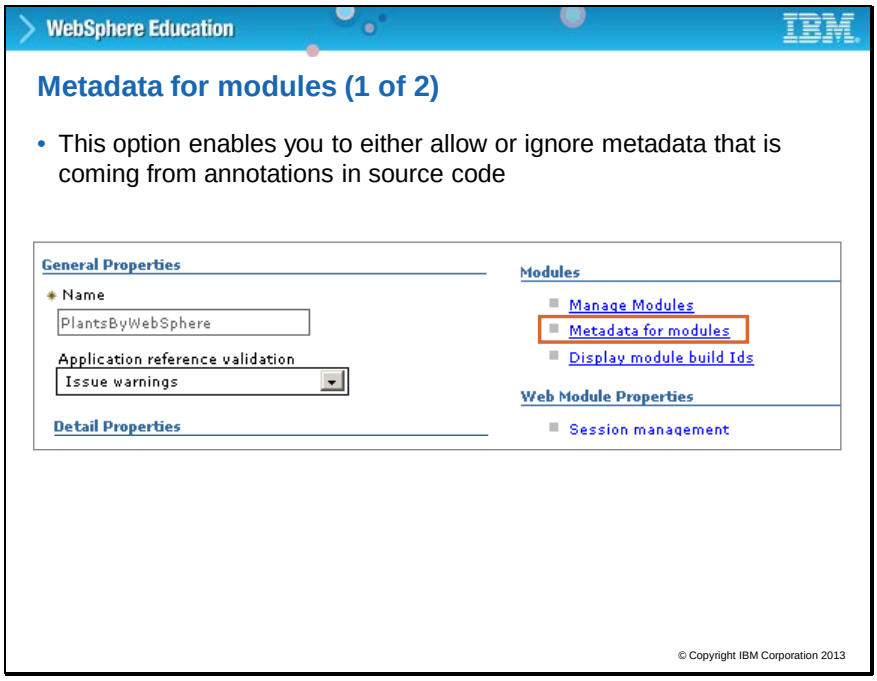
You can now view the ejb-jar.xml deployment descriptor of the PlantsByWebSphere.war module, of the PlantsByWebSphere application, by clicking **View EJB Deployment Descriptor** under **Additional Properties**. In Java EE 6, EJBs can be stored within a WAR file.

Slide 26



This screen capture shows the ejb-jar.xml deployment descriptor for the PlantsByWebSphere application. The part of the deployment descriptor inside the frame is information that WebSphere Application Server generates from annotations in the PlantsByWebSphere.war file and inserted in the ejb-jar.xml file when the PlantsByWebSphere application was deployed.

Slide 27

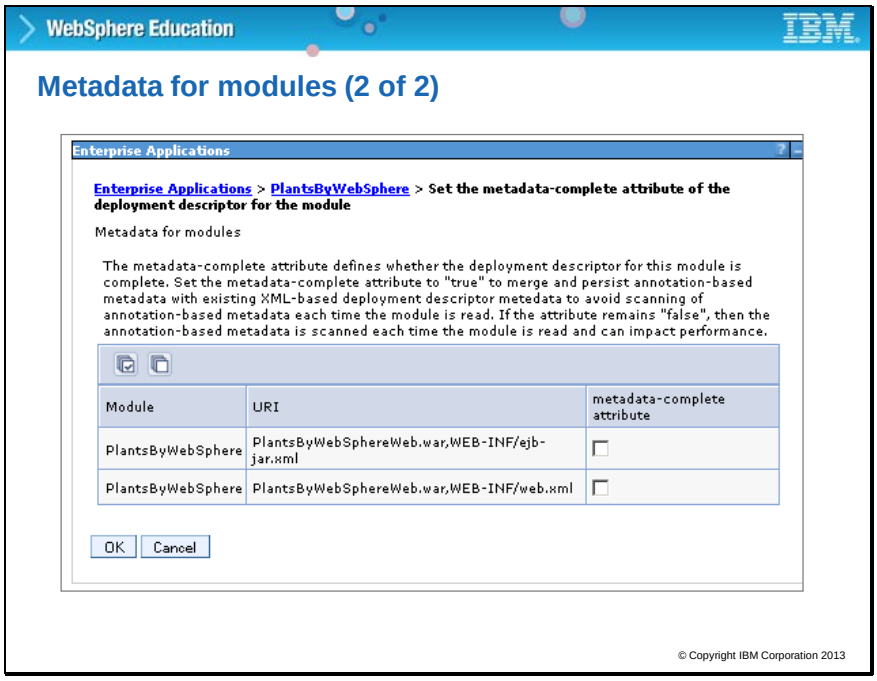


This screen capture shows where you can control the metadata for the application modules. Select **Enterprise applications > *application_name***, and click **Metadata for modules** under **Modules**.

Using this option, you can specify whether the application metadata can be specified in annotations in code together with metadata in deployment descriptors, or whether the metadata can be specified through deployment descriptors. This option is useful because the deployer, or administrator, of the application can change some of the application metadata. For example, an administrator can change the data sources that the enterprise application uses. The administrator can change deployment descriptors only, since in many cases, the application source code is not packaged with the EAR file.

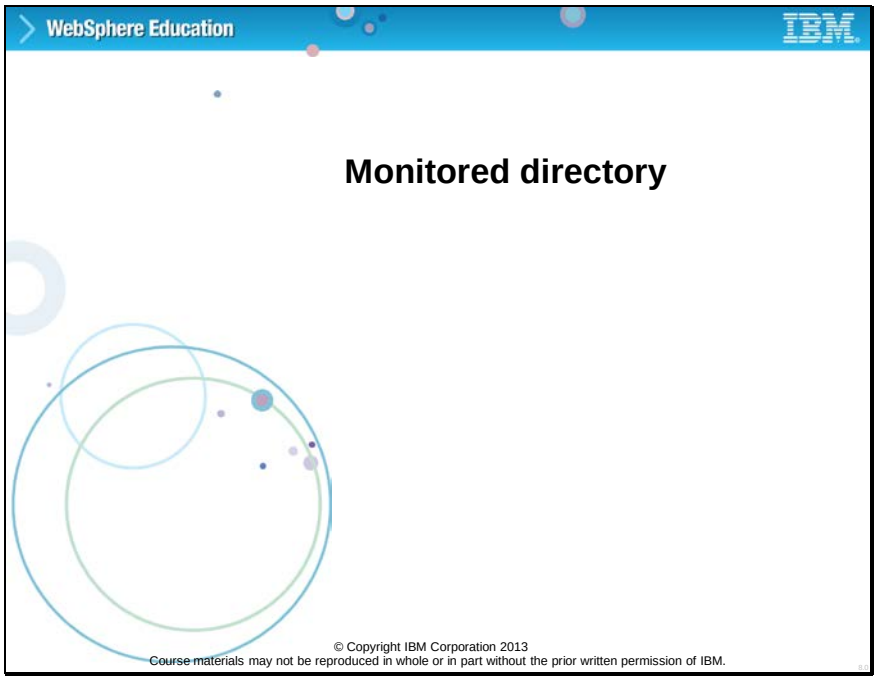
It is not necessarily a recommended practice to set metadata-complete to *true* if your programmers intend to use annotation-based programming techniques. The technique that you adopt can be based on personal preference.

Slide 28



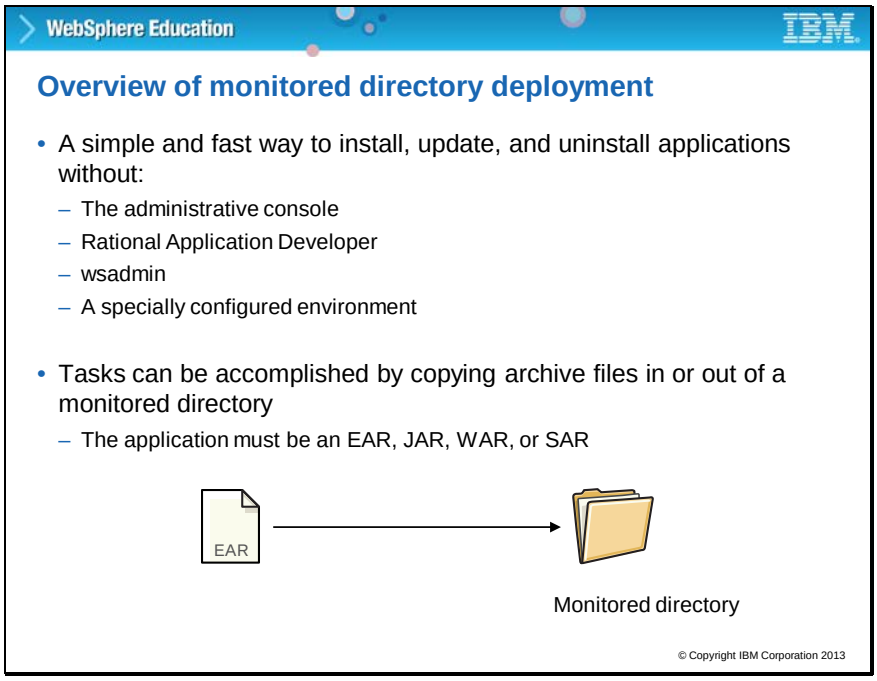
This screen capture displays the metadata-complete attributes for PlantsByWebSphere that are not checked. When the attributes are checked, all annotation metadata is written to the appropriate deployment descriptor and afterward all annotations in the enterprise application are ignored.

Slide 29



Topic: Monitored directory. In this topic, you learn about the feature called “monitored directory”, which is also known as “drag-and-drop.”

Slide 30



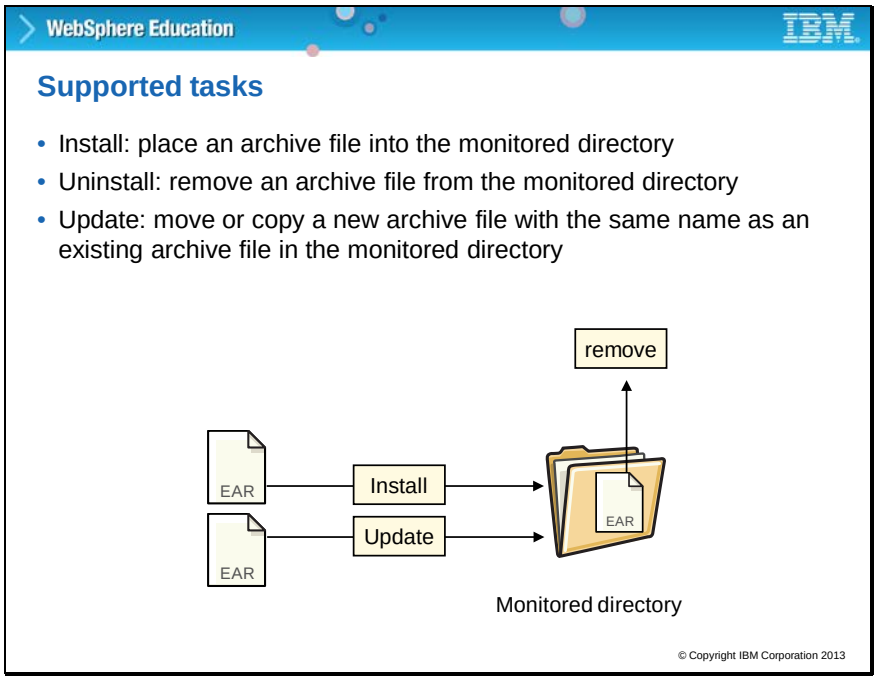
The basic definition for the monitored directory feature is “a place where the WebSphere Application Server looks for updates.” The use of this feature is as simple as copying and deleting files. There are a few more things to know before using the monitored directory feature.

A monitored directory application deployment is a simple and fast way for administrators and developers to install, update, and uninstall applications by moving archive files in or out of a monitored directory. A user who prepackages an application file with all bindings specified, can quickly deploy that application without any tools other than a running application server, or, in a network deployment environment, a deployment manager.

Monitored directory application deployment can be performed with applications packaged as enterprise archive (EAR) files, web archive (WAR) files, Java archive (JAR) files, or SIP application resources (SARs). In addition to the mentioned archive files, an application can be deployed by using a property file.

Property files are different from the other file types. They are not archives. Instead, they contain properties that describe an application, including the source archive location and its installation parameters. Monitored directory deployment with property files relies on the properties file based configuration (PFBC) feature introduced in WebSphere Application Server Version 7. Using a property file can allow a higher degree of control over the deployment than with a plain archive.

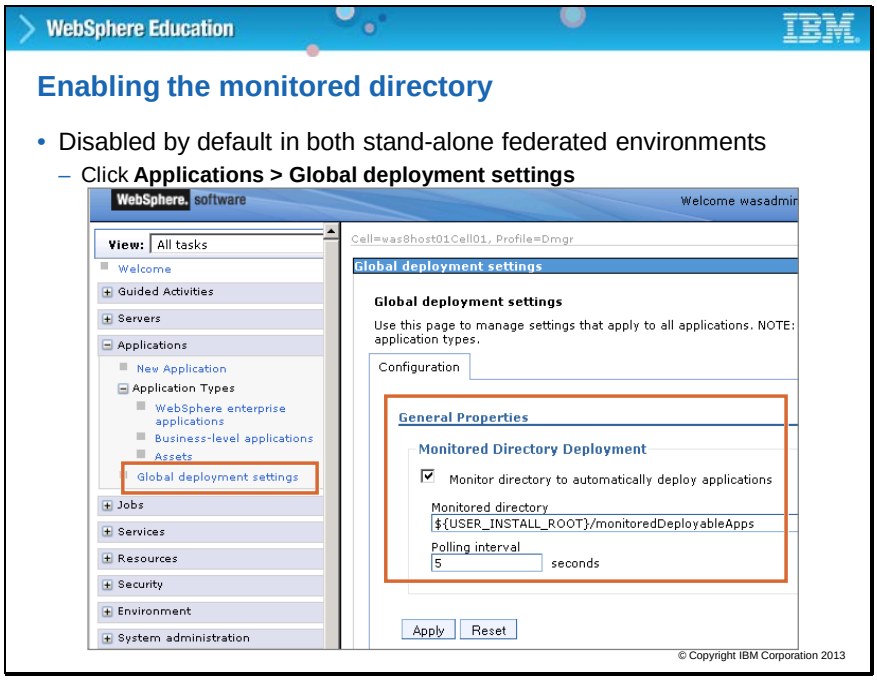
Slide 31



To install an application by using the monitored directory feature, use a graphical file manager or the command line to copy or move its archive file into a monitored directory. Deleting a file from a monitored directory causes uninstallation of the corresponding application.

To perform a full replacement update of a deployed application, move or copy an updated archive file with the same application name into a monitored directory. The archive file name determines the application name. Or, in case of an EAR archive, the display name determines the application name if a display name is specified in the archive.

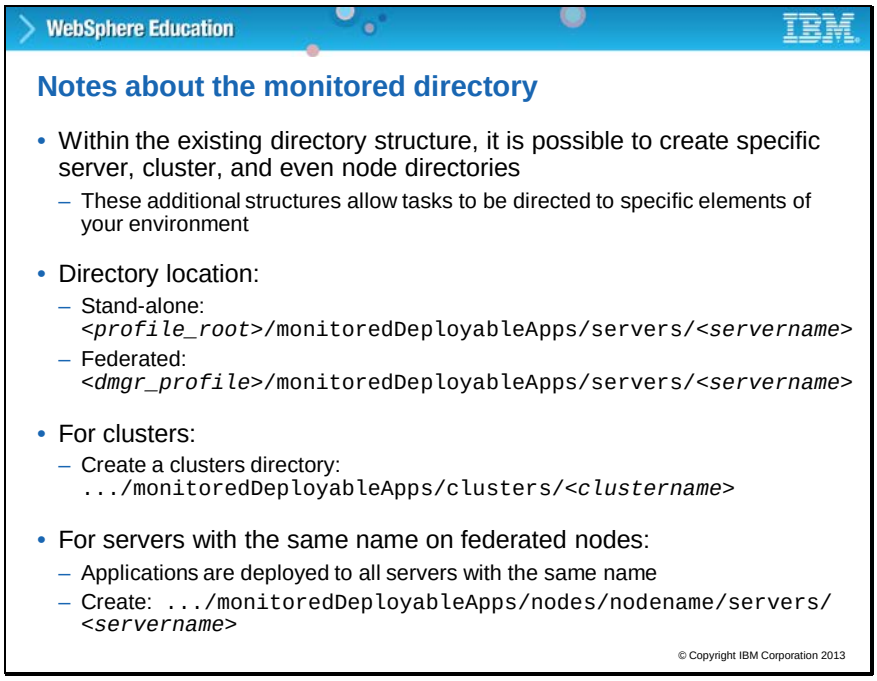
Slide 32



The monitored directory feature is not enabled in the application server after installation by default. Before using the monitored directory, enable it from the **Applications > Global deployment settings** page of the administrative console. This feature requires a path (the directory) and a polling interval (in seconds) that tells the server where to look and how often to check. Enabling the service must be done only once.

The deployment manager, or base server where the service is running, must be restarted to register any changes on the **Global deployment settings** page.

Slide 33



This slide indicates that while the monitored directory feature is defined and described as a single directory, it is composed of several subdirectories. The path to a monitored directory depends on the type of application server, and what the target of the deployment is going to be. To control where the monitored directory applications are deployed, you create more directories to represent their clusters or servers. These names must exactly match what is actually in the cell.

If you are using a stand-alone application server, then the only possible target is the server itself, and the monitored directory is automatically created if the service is enabled. For example, if the profile is called AppSrv01, and the server is named server1, the path is: ../profiles/AppSrv01/monitoredDeployableApps/server1.

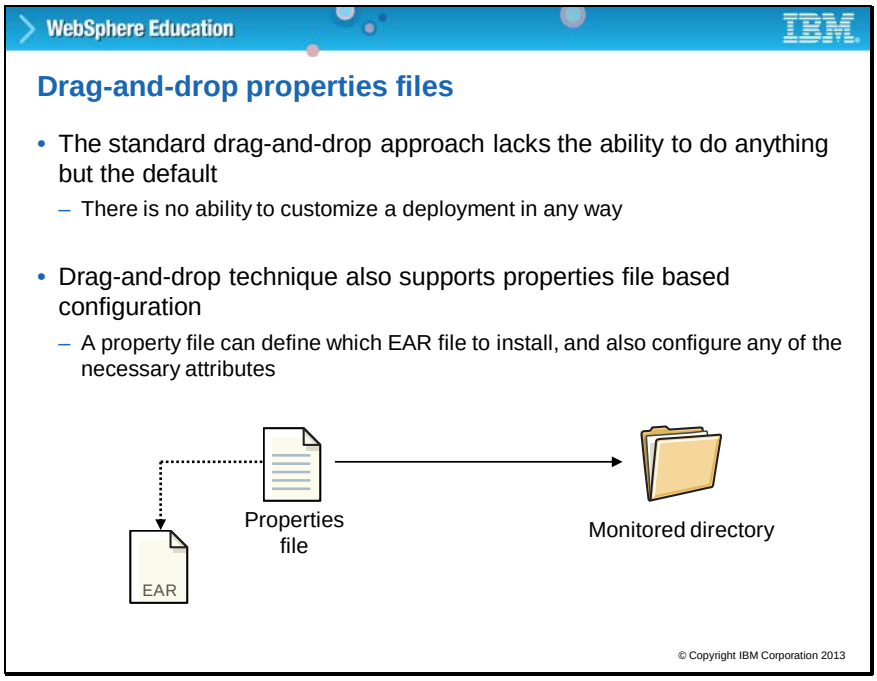
If you are using a network deployment system, it is necessary to create the monitored directories manually. For application servers on a node that is federated with a deployment manager, you must create the monitored directories for servers under the deployment manager profile: ../profiles/dmgrprofilename/monitoredDeployableApps/servers/ server_name.

If multiple servers on different nodes have the same name and you want to target only one of the servers, you can specify the node and server in the path to the monitored directory. Create a directory for the node by using the node name, then servers, and finally the server_name

directory:
../profiles/dmgrprofilename/monitoredDeployableApps/nodes/node_name/servers/server_name.

For clusters, create a monitored directory under the deployment manager profile with the name of the targeted cluster: ../profiles/dmgrprofilename/monitoredDeployableApps/clusters/ .

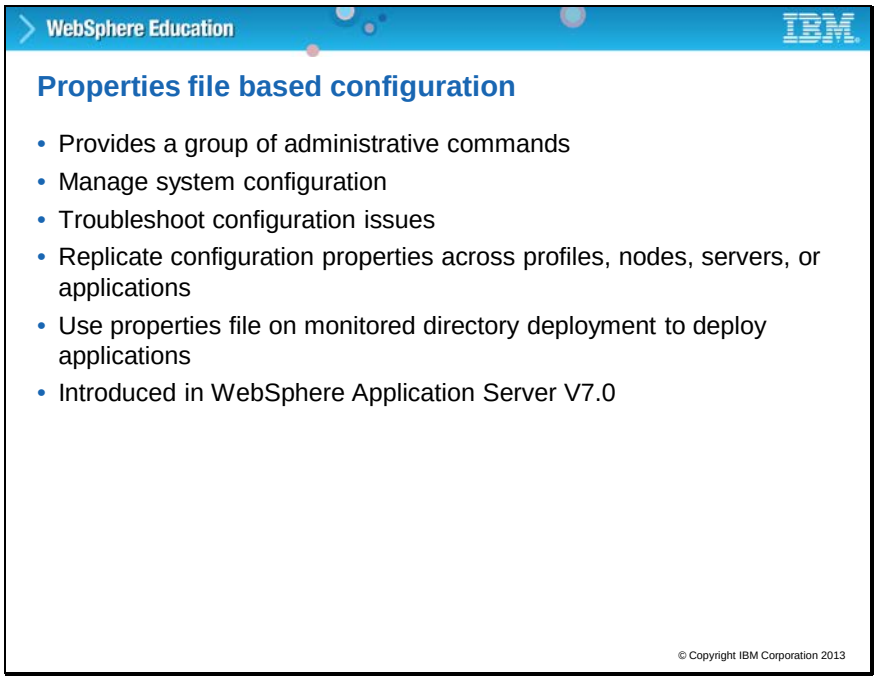
Slide 34



The standard monitored directory feature supports the ability to drag an application from one directory and drop it into another directory. But it does not allow you to customize the deployment.

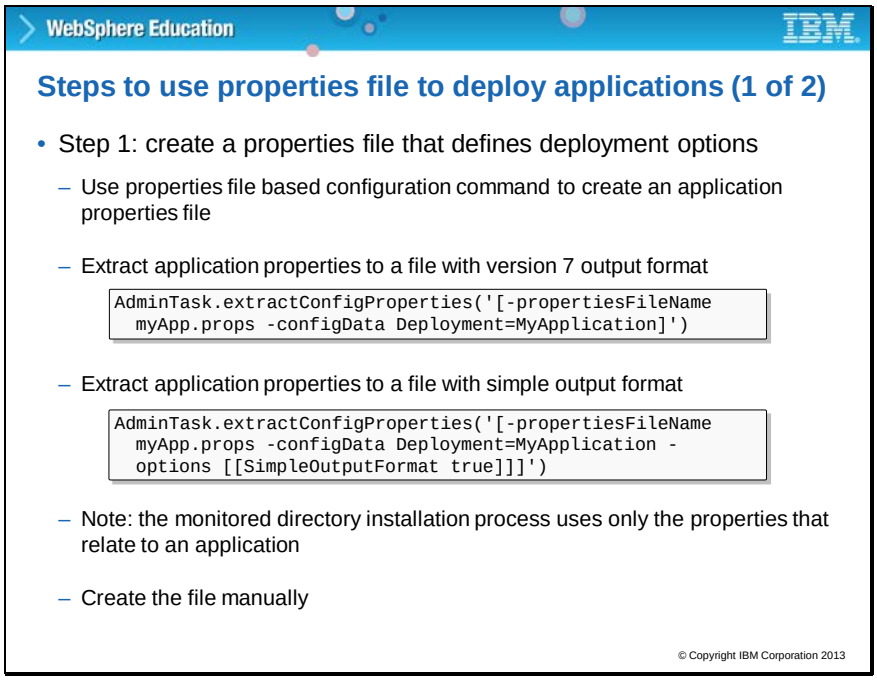
To customize the deployment, you use a properties file. Monitored directory supports the ability to drag and drop a properties file it into another directory, which allows you to customize any installation settings that might be required for monitored directory installations.

Slide 35



The ability of the monitored directory to process property files was introduced with WebSphere Application Server version 7. Property files allow for a select set of administrative commands, which allow for activities such as the ability to manage the system configuration, troubleshoot configuration issues, and replicate configuration properties across profiles, nodes, servers, or applications. In addition, a property file can be used to deploy an application.

Slide 36

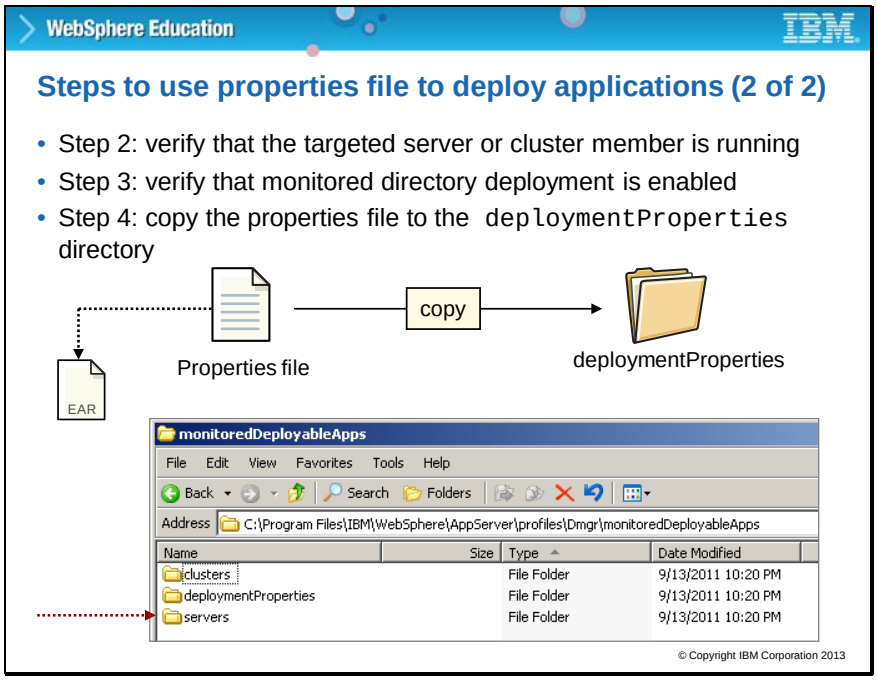


This slide displays an example of using a properties file to deploy applications. The wsadmin AdminTask.extractConfigProperties command is useful for capturing properties for use with the monitored directory feature.

Step 1 shows how to use the properties file based configuration **extractConfigProperties** command to extract application properties to a file. By default, the **extractConfigProperties** command produces output that displays all columns, including all hidden and non-hidden columns of installation tasks and task data values, in separate rows. You can also extract application properties in simple output format to display non-hidden columns of installation task data in columnName=value pairs.

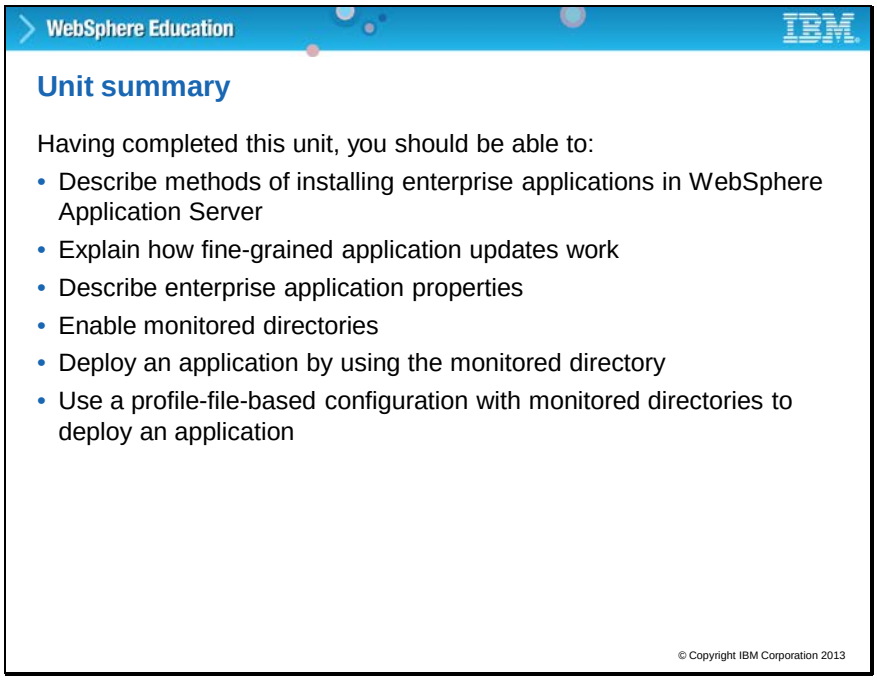
You can also use the properties file examples that are documented in the information center to create an application properties file to deploy, uninstall, or update an application.

Slide 37



Before a property file can be processed, the monitored directory service must be active and the application must be running. The property file must be copied to the deploymentProperties directory for the application. For example, you must copy a property file to the cluster where the application is running for the event to succeed. You can use a file browser to drag the properties file to the monitored directory or use the operating system command to copy the properties file to the directory.

Slide 38



You completed this unit.

Having completed this unit, you should be able to:

- Describe methods of installing enterprise applications in WebSphere Application Server
- Explain how fine-grained application updates work
- Describe enterprise application properties
- Enable monitored directory
- Deploy an application by using the monitored directory
- Use a property-file-based configuration with monitored directories to deploy an application