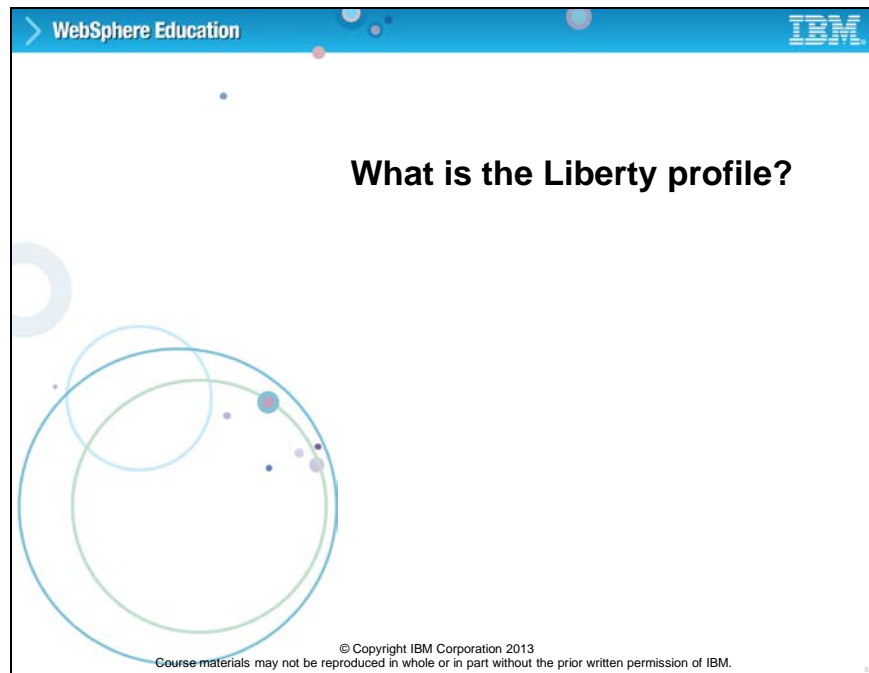


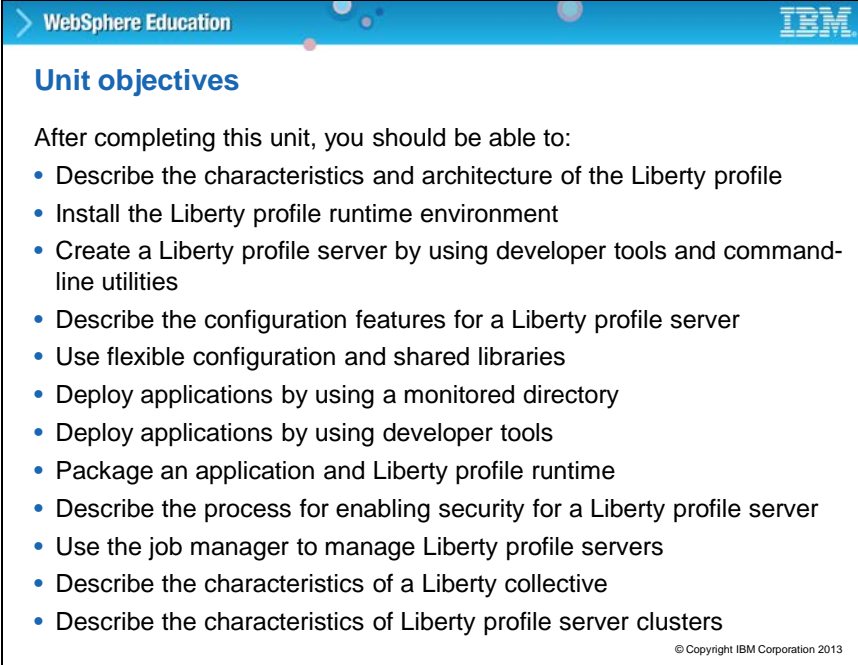
Slide 1



Unit 19: Overview of the Liberty profile

This unit introduces the new Liberty profile and covers the installation of the run time environment. How Liberty profiles servers are created by using developer tools and command-line utilities is also explained. Different methods for deploying applications to the Liberty profile servers are presented. Other topics include flexible configuration, application security, packaging Liberty profile resources, and how to use the job manager to manage multiple servers.

Slide 2



The slide is titled 'Unit objectives' and is part of a 'WebSphere Education' presentation. It lists 13 objectives for completing the unit. The slide includes the IBM logo in the top right corner and a copyright notice for IBM Corporation 2013 in the bottom right corner.

Unit objectives

After completing this unit, you should be able to:

- Describe the characteristics and architecture of the Liberty profile
- Install the Liberty profile runtime environment
- Create a Liberty profile server by using developer tools and command-line utilities
- Describe the configuration features for a Liberty profile server
- Use flexible configuration and shared libraries
- Deploy applications by using a monitored directory
- Deploy applications by using developer tools
- Package an application and Liberty profile runtime
- Describe the process for enabling security for a Liberty profile server
- Use the job manager to manage Liberty profile servers
- Describe the characteristics of a Liberty collective
- Describe the characteristics of Liberty profile server clusters

© Copyright IBM Corporation 2013

Title: Unit objectives

After completing this unit, you should be able to:

- Describe the characteristics and architecture of the Liberty profile
- Install the Liberty profile runtime environment
- Create a Liberty profile server by using developer tools and command-line utilities
- Describe the configuration features for a Liberty profile server
- Use flexible configuration and shared libraries
- Deploy applications by using a monitored directory
- Deploy applications by using developer tools
- Package an application and Liberty profile runtime
- Describe the process for enabling security for a Liberty profile server
- Use the job manager to manage Liberty profile servers
- Describe the characteristics of a Liberty collective
- Describe the characteristics of Liberty profile server clusters

Slide 3

WebSphere Education

Topics

- Introduction to the Liberty profile
- Tools, run time, and installation
- Configurations
- Security
- Using the job manager
- Liberty collectives and clusters

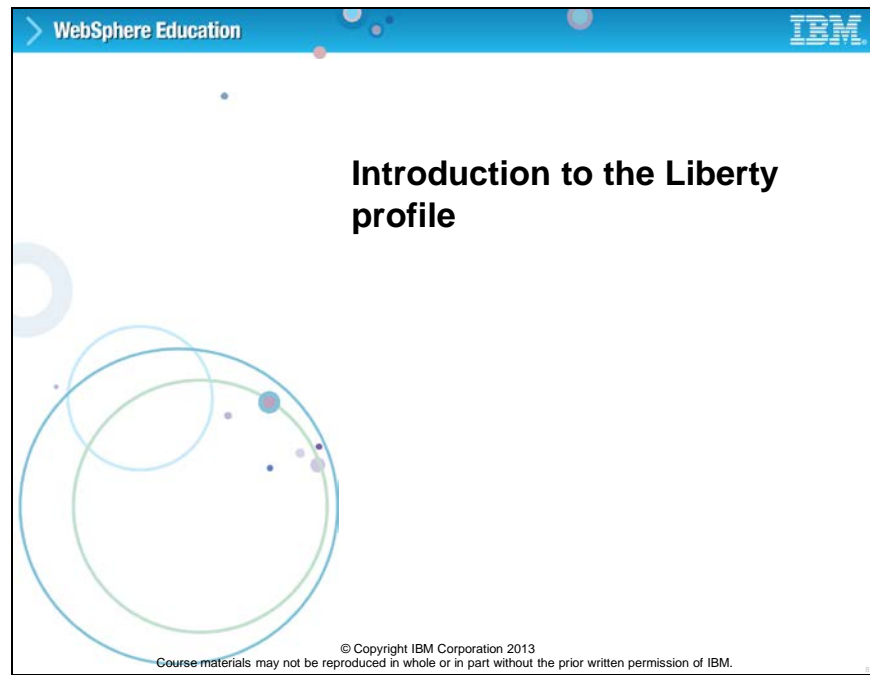
© Copyright IBM Corporation 2013

Title: Topics

This unit contains information about the following topics.

- Introduction to the Liberty profile
- Tools, run time, and installation
- Configurations
- Security
- Using the job manager
- Liberty collectives and clusters


Slide 4



Topic: Introduction to the Liberty profile

This unit describes the architecture and characteristics of the Liberty profile.

Slide 5

WebSphere Education


What is the Liberty profile?

- The Liberty profile is an application server runtime environment
 - Highly composable by using feature elements
 - Configuration changes are dynamic on a running server
 - Server starts quickly
- You can install a Liberty profile runtime environment by extracting a JAR file
 - The Liberty profile does not include a Java runtime environment (JRE), so you must install an IBM or Oracle JRE separately
- Liberty profile servers support two models of application deployment:
 - Deploy an application by dropping it into the `dropins` directory
 - Deploy an application by adding it to the server configuration
- The Liberty profile supports a subset of the full WebSphere Application Server programming model:
 - Web applications
 - OSGi (Open Service Gateway initiative) applications
 - Java Persistence API (JPA)
- Can be easily configured to support the full Java EE 6 Web Profile

© Copyright IBM Corporation 2013

Title:

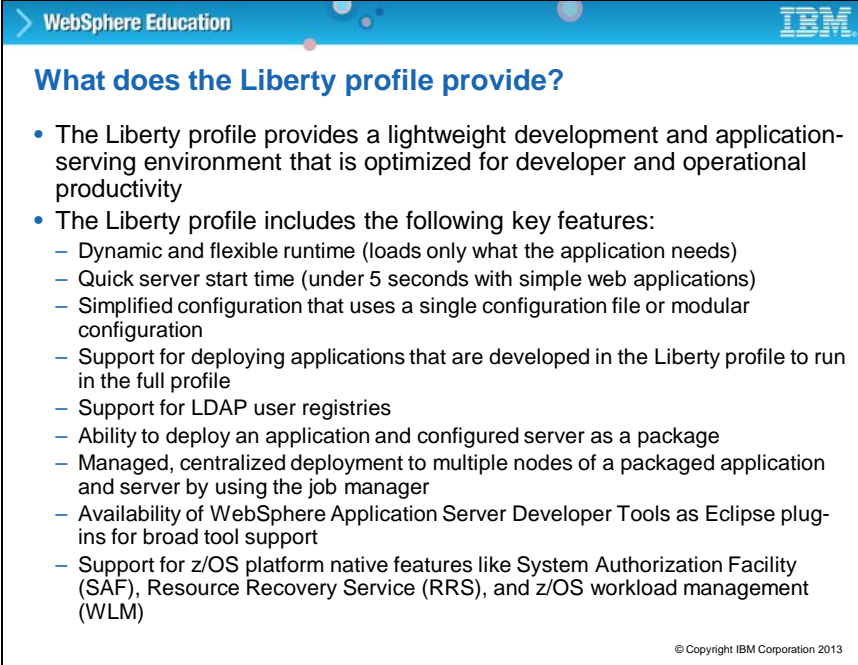
The Liberty profile provides an application server runtime environment that is a highly composable, fast to start, and dynamic. A highly composable system is defined as: “A system that provides components that can be selected and assembled in different combinations to provide specific application requirements.”

The Liberty profile does not include a Java runtime environment (JRE), so you must install an IBM or Oracle JRE or SDK to support the runtime environment.

A Liberty profile server supports two models of application deployment. You can use the monitored directory function and deploy an application by dropping the application archive into the "drop-ins" directory.

You can also deploy an application by storing the application archive in a shareable applications directory and adding location information for it to the server configuration file.

The Liberty profile supports a subset of the following parts of the full WebSphere Application Server programming model: Web applications, OSGi applications, and the Java Persistence API (JPA).



The slide is titled "What does the Liberty profile provide?" and is part of a "WebSphere Education" presentation. It lists the following features:

- The Liberty profile provides a lightweight development and application-serving environment that is optimized for developer and operational productivity
- The Liberty profile includes the following key features:
 - Dynamic and flexible runtime (loads only what the application needs)
 - Quick server start time (under 5 seconds with simple web applications)
 - Simplified configuration that uses a single configuration file or modular configuration
 - Support for deploying applications that are developed in the Liberty profile to run in the full profile
 - Support for LDAP user registries
 - Ability to deploy an application and configured server as a package
 - Managed, centralized deployment to multiple nodes of a packaged application and server by using the job manager
 - Availability of WebSphere Application Server Developer Tools as Eclipse plug-ins for broad tool support
 - Support for z/OS platform native features like System Authorization Facility (SAF), Resource Recovery Service (RRS), and z/OS workload management (WLM)

© Copyright IBM Corporation 2013

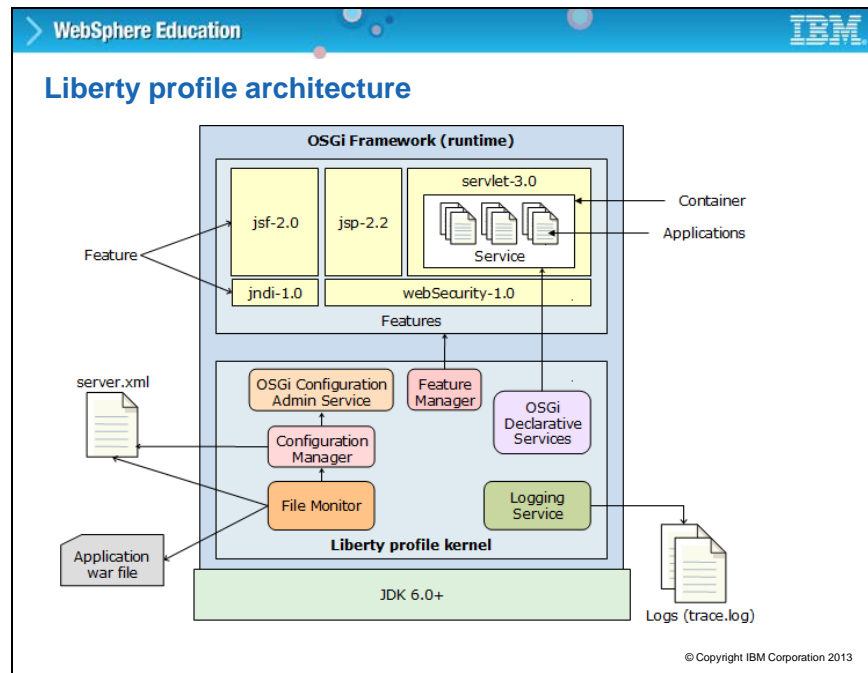
Title: What does the Liberty profile provide?

The Liberty Profile provides a lightweight application server, which is focused on the development and operations experience. When used as a test server, it can have a small footprint that results in the fastest possible server start.

Simplified server configuration is possible since a single XML file is dynamically updated. Configuration data is shareable by “including” common configuration elements from other configuration files. The Liberty profile is compatible with full-profile WebSphere Application Server editions.

The Liberty profile run time is free for developers and does not expire. The runtime environment is focused on web applications and OSGi applications that include Java persistence, transactions, and security.

Slide 7

**Title: Liberty profile architecture**

As shown in the graphic, the Liberty profile is built on OSGi technologies. The server process runs as OSGi bundles and comprises a single Java virtual machine (JVM), the Liberty profile kernel, and any number of optional features.

A functional server is produced by starting the runtime environment with a configuration that includes a list of required features. *Features* are the units of capability, by which the runtime environment is defined and controlled. They are the primary mechanism that makes the server composable. For example, if the servlet feature is specified, the runtime environment operates as a servlet engine or web container. By default, a server runs no features. You can use the feature manager to add the features that are needed. The feature manager is one of the kernel bundles that receives the configuration, resolves each feature to a list of bundles, installs the feature into the framework, and then starts the feature. When the features that are needed are specified, the default configuration of those features provides a rich environment that is designed to cover most common requirements.

The configuration manager reads the server configuration from persistent files, parses the configuration into sets of properties, and then uses those sets of properties to populate the OSGi Configuration Admin service. This service maintains the runtime view of the configuration, and when configuration updates are made, this service injects each set of properties into the service that "owns" them.

Slide 8

WebSphere Education

IBM

Clarification: Profile versus profile

- How does the Liberty profile relate to the application server and custom profiles?
- The term “profile” has another meaning in WebSphere V8.5
 - Installation profile** (full or traditional profile versus Liberty profile): Refers to what runtime is being installed
 - Configuration profile** (deployment manager, application server, custom): Refers to which configuration is being used within the full installation

Install Packages

Select the features to install.

Install Licenses Location Features Summary

Features

IBM WebSphere Application Server Network Deployment 8.5.0.0

WebSphere Application Server Full Profile

☒ EJBProxy tool for pre-EJB 3.0 modules

☒ Stand-alone thin clients, resource adapters and embeddable

☒ Stand-alone thin clients and resource adapters

☒ Embeddable EJB container

☐ Sample applications

☐ WebSphere Application Server Liberty Profile

Installation profiles

Profile Manager

Environment Selection

Configuration profiles

Select a specific type of environment to create.

Environments:

WebSphere Application Server

Cell (deployment manager and a federated application server) Management

Application server

Custom profile

Secure proxy (configuration-only)

© Copyright IBM Corporation 2013

Title: Clarification: Profile versus profile

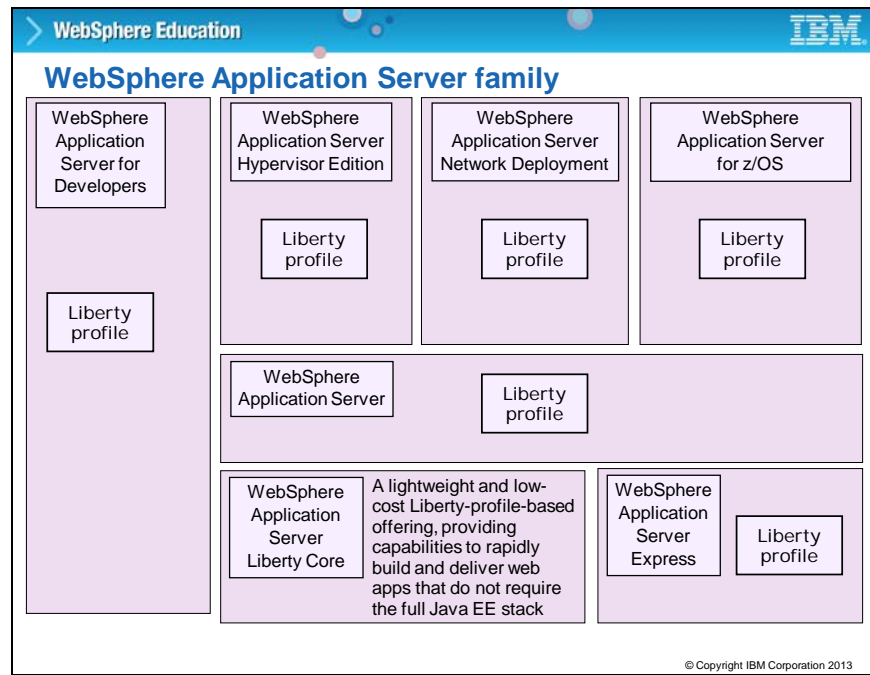
How does the Liberty Profile relate to the Application Server and Custom profiles?

The term “profile” is overloaded with an extra meaning in WebSphere V8.5.

The Liberty profile is a type of installation profile. An installation profile refers to which run time (or product binary files) is being installed, the WebSphere Application Server Full Profile or the WebSphere Application Server Liberty Profile.

Traditional profiles such as, Deployment Manager, Application Server, and Custom are types of configuration profiles and relate to which configuration is being used within the full product installation.

Slide 9



Title: WebSphere Application Server family: V8.5

The Liberty profile is available in all of the WebSphere Application Server family editions.

Slide 10

WebSphere Education

IBM

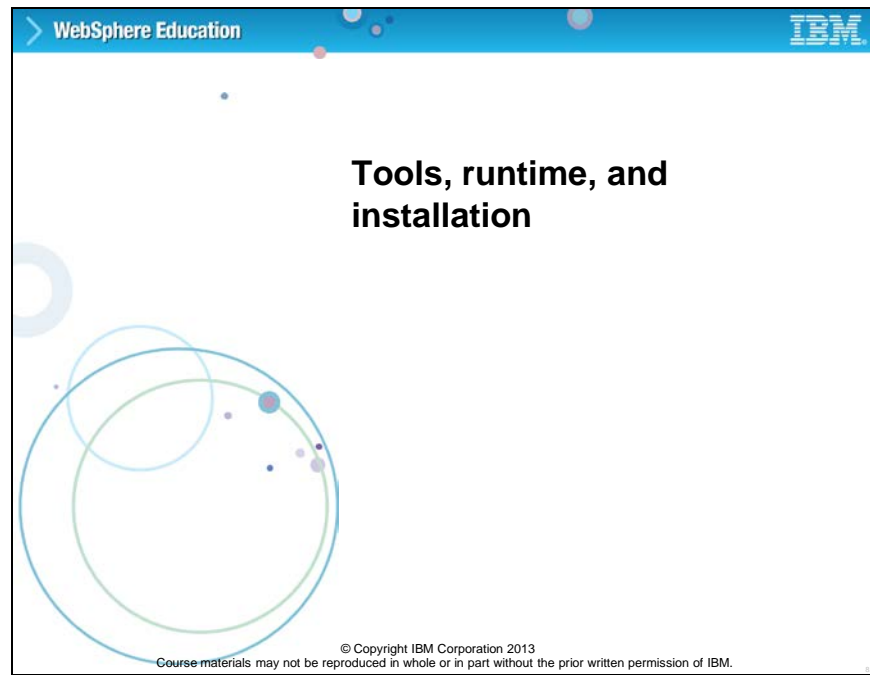
Features available in the Liberty editions

Liberty Core	Base, Express	Network Deployment	z/OS
servlet jsp jsf jpa jndi jdbc json jaxrs wab ssl osgi.jpa monitor sessionDatabase appSecurity blueprint restConnector localConnector beanValidation ejblite cdi managedBeans oauth ldapRegistry webCache concurrent collectiveMember	was.JmsClient was.JmsServer wsSecurity wmq.JmsClient was.JmsSecurity mongodb jaxb jaxws	collectiveController clusterMember	zosSecurity zosTransaction zosWlm

© Copyright IBM Corporation 2013

Different editions of the Liberty profile provide different functions. For example, the ability to create a liberty profile cluster member is only available in the Liberty profile Network Deployment edition.


Slide 11



Title: Tools, run time, and installation

This topic presents some of the development tools that can be used to install the Liberty profile run time.

Slide 12



The slide is titled "Liberty profile developer tools" and is part of a "WebSphere Education" presentation. It lists three developer tools used for developing and testing applications on a Liberty profile server. The tools are: WebSphere Application Server Developer Tools for Eclipse, IBM Assembly and Deploy Tools for WebSphere Administration, and Rational Application Developer. The slide also includes the IBM logo and a copyright notice for IBM Corporation 2013.

- Applications can be developed and tested on a Liberty profile server by using the following developer tools
 - WebSphere Application Server Developer Tools for Eclipse
 - IBM Assembly and Deploy Tools for WebSphere Administration
 - Rational Application Developer

© Copyright IBM Corporation 2013

Title: Liberty profile developer tools

Each of the developer tools that are listed on this slide is Eclipse-based and supports the Liberty Profile Runtime Environment. Using these developer tools, you can install the runtime and create Liberty profile servers. The developer tools provide editors that allow you to configure servers with any features required by the applications that run on the servers.

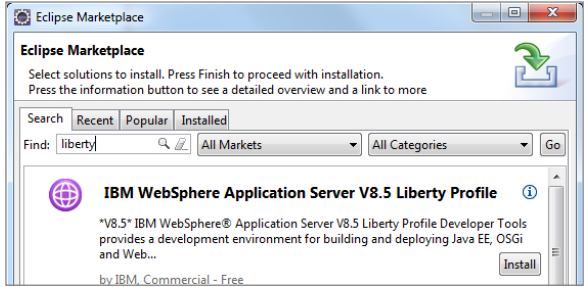
Slide 13

WebSphere Education

IBM

Where to get WebSphere Application Server Developer Tools for Eclipse

- WebSphere Application Server Developer Tools for Eclipse is available for free:
 - From <http://www.wasdev.net>
 - Through the Eclipse Marketplace (**Help > Eclipse Marketplace**)
 - Through the IBM Installation Manager



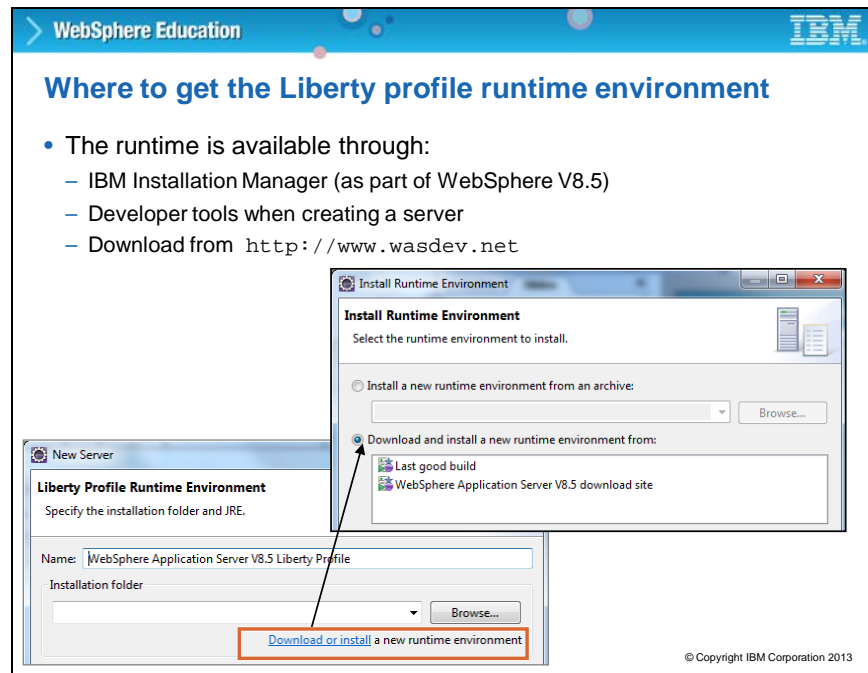
© Copyright IBM Corporation 2013

Title: Where to get WebSphere Application Server Developer Tools for Eclipse

The WebSphere Application Server Developer Tools for Eclipse (WDT) is a free download from the WASdev community website by using the web address that is shown on this slide.

WDT is also available by using the Eclipse Marketplace, which can be accessed in Eclipse by clicking **Help > Eclipse Marketplace**.

WDT can also be installed by using the IBM Installation Manager and accessing the appropriate installation repository.



Title: Where to get the Liberty Profile Runtime Environment

The Liberty Profile run time is available through IBM Installation Manager as part of WebSphere Application Server V8.5.

The Liberty Profile run time is also available through WDT when creating a server, and as a free download from the WASdev community website.

Slide 15

WebSphere Education 

Installing the Liberty profile

- Installing the Liberty profile runtime environment
 - Use developer tools such as WebSphere Application Server Developer Tools, IBM Assembly and DeployTools, or Rational Application Developer
 - Use the IBM Installation Manager
 - Extract from a compressed archive (`java -jar wlp-8500.jar`)



© Copyright IBM Corporation 2013

Title: Installing the Liberty profile

The Liberty Profile Runtime Environment can be installed by using different tools that include: WebSphere Application Server Developer tools (WDT), IBM Installation Manager (IIM), and by downloading a Liberty profile archive and extracting by using command-line tools.

WebSphere Education

IBM

Creating a Liberty profile server

- Creating a server can be done quickly
 - Using the command line
 - Using developer tools (WebSphere Application Server Developer Tools, IBM Assembly and DeployTools, Rational Application Developer)
- From the command line:

Terminal

File Edit View Terminal Tabs Help

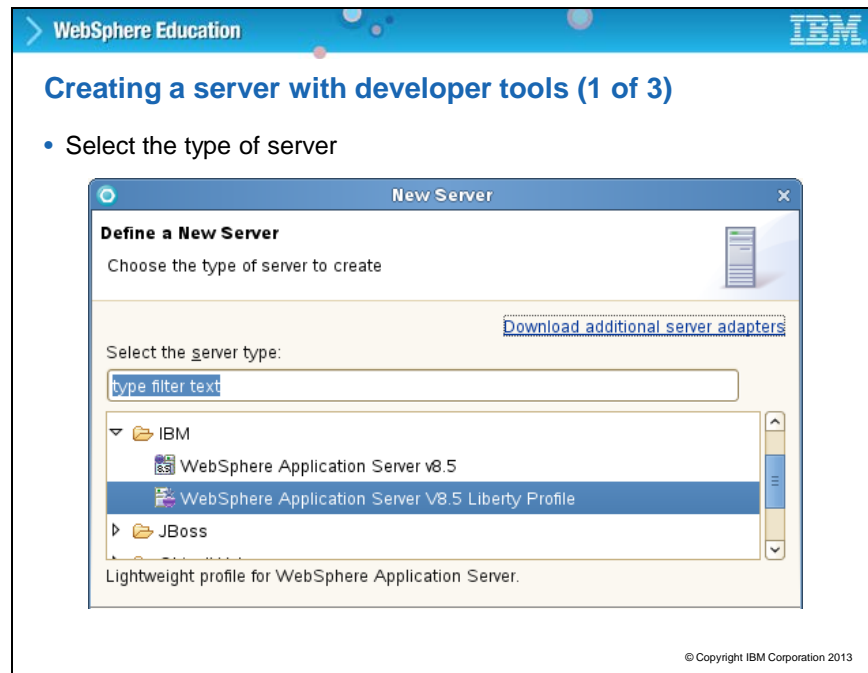
was85host:/opt/IBM/wlp/bin # ./server create server2
Server server2 created.
was85host:/opt/IBM/wlp/bin #

© Copyright IBM Corporation 2013

Title: Creating a Liberty profile server

A Liberty profile server is created from the command line by using the server create command. After installing the Liberty profile run time, there is a bin directory from which you can run a command such as: server create <server_name> as shown on this slide.

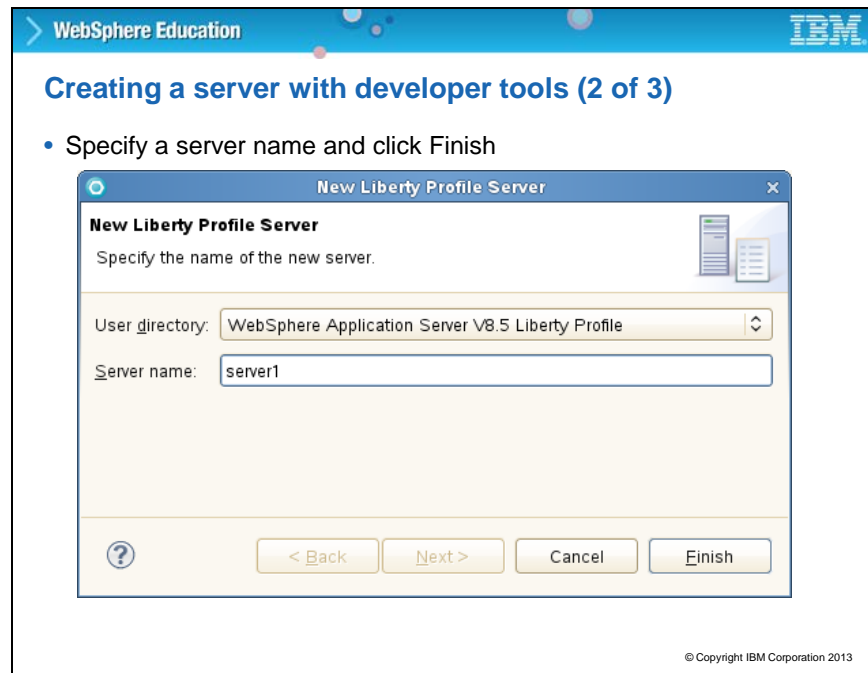
Slide 17



Title: Creating a server with developer tools (1 of 3)

Each of the developer tools provides wizards for creating an instance of a Liberty profile server. The first step is to select the server type: WebSphere Application Server V8.5 Liberty Profile. In this step, you can also specify a host name, and new server name, or accept the defaults.

Slide 18



Title: Creating a server with developer tools (2 of 3)

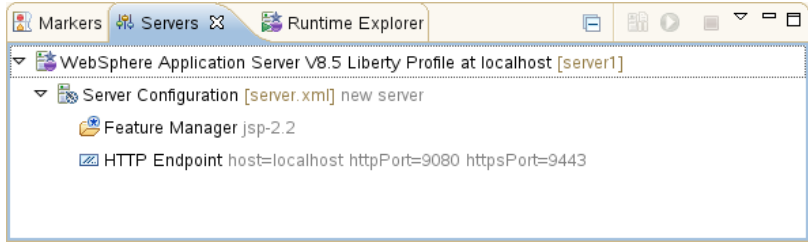
This step allows you to add another server name if you want to change the default name.

Slide 19

WebSphere Education
IBM

Creating a server with developer tools (3 of 3)

- Server1 is created and is now listed in the Servers view
- Expand the server entry and you see the Server Configuration
- Configuration for a new server consists of:
 - The jsp- 2.2 feature
 - The HTTP Endpoint definition (9080 and 9443 are default ports for all new servers)



© Copyright IBM Corporation 2013

Title: Creating a server with developer tools (3 of 3)

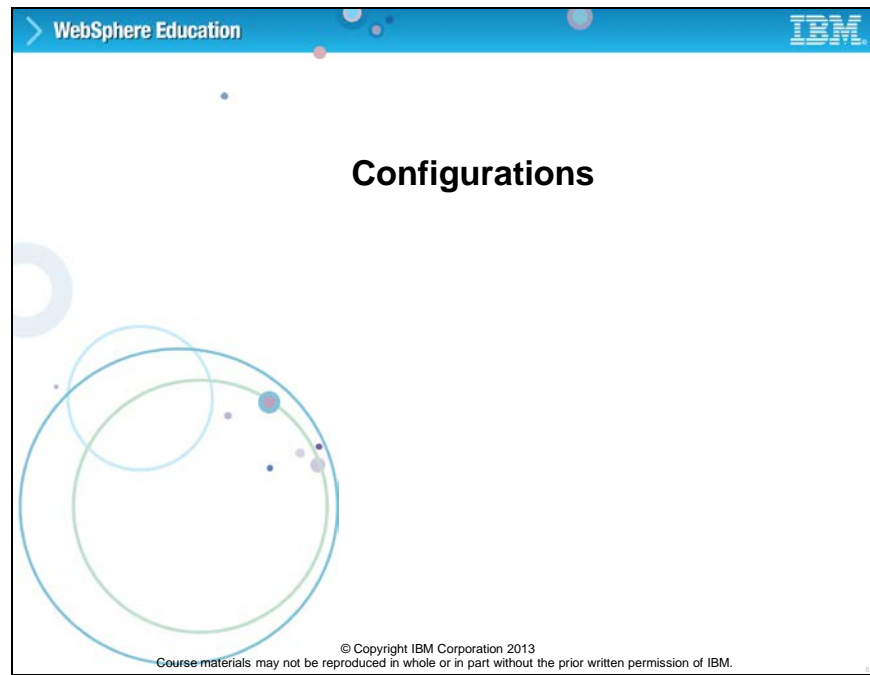
The Liberty profile server instance is created with a minimal configuration that includes the jsp-2.2 feature and the default HTTP endpoint settings.

The Servers view has icons that can be clicked to start or stop a server. In addition, right-clicking the server name brings up a menu with several useful items such as a Restart button and a Utilities link.

The Utilities link allows you to select the following actions:

- Create SSL Certificate
- Generate a Web Server plug-in
- Package Server
- Generate Dump for Support

Slide 20



Topic: Configurations

Some of the ways a Liberty profile server can be configured to support applications

are covered in this topic.Slide 21

WebSphere Education

IBM

Simplified server configuration

- No need for administrative console, wsadmin, or enhanced EARs
 - These tools are not supported
- Configuration in XML files
 - **Simplest case:** one XML file (`server.xml`) for all server configuration
 - Editable within the developer tool or by using a text editor
 - Exportable, shareable, and versionable

server.xml

```

<server description="new server">

  <!-- Enable features -->
  <featureManager>
    <feature>jsp-2.2</feature>
    <feature>localConnector-1.0</feature>
    <feature>jdbc-4.0</feature>
  </featureManager>

  <httpEndpoint host="localhost" httpPort="9080"
    httpsPort="9443" id="defaultHttpEndpoint" />

  <applicationMonitor updateTrigger="mbean" />

  <application id="HelloWorld" location="HelloWorld.war"
    name="HelloWorld" type="war" />

</server>

```

Liberty server configuration

resources.xml
server.xml
...

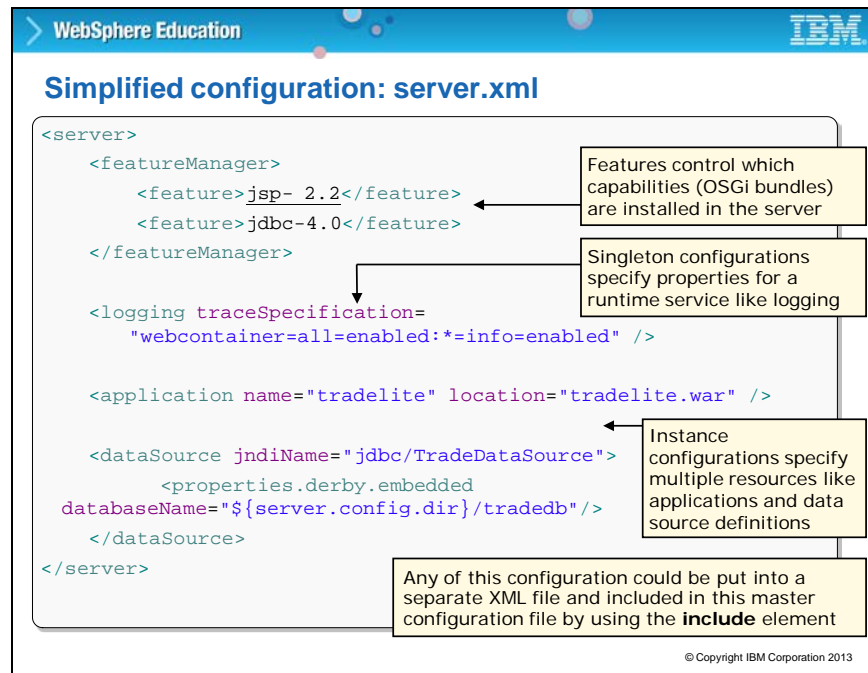
Traditional WebSphere Application Server

© Copyright IBM Corporation 2013

Title: Simplified server configuration

There is no administrative console or wsadmin scripting interface for configuring a Liberty profile server. All of the configuration for a server can be contained in a single XML file called `server.xml`. This file can be manually updated with a text editor and when the changes are saved, the server is dynamically updated. No server restart is required. Other XML configuration files and their contents can be accessed by using an include statement in the `server.xml` file.

In addition, several other files can be used to configure the Liberty profile server. These files include the `bootstrap.properties` file, the `jvm.options` file, and the `server.env` file. The use of these files is fully documented in the information center.



Title: Simplified configuration: server.xml

This slide shows an example of a simple server.xml file for a Liberty profile server.

It consists of a **feature manager** section, a **logging element**, an **application element**, and a **data source element**.

Liberty Profile features control which capabilities (OSGi bundles) are started in the server JVM. *Singleton* configuration elements specify properties for runtime services such as logging and tracing.

Instance configurations can specify multiple resources like applications and data source definitions.

Any of these configuration elements might be put into a separate XML file and 'included' in the server.xml file for multiple servers. For example, multiple Liberty Profile servers might share a data source definition, so that the configuration of the data source can be stored in a single shared XML file.

WebSphere Education

IBM

Flexible configuration

- Shareable configuration snippets

```
server>
...
<include location="http://cfgserver/global.xml/" />
<include location="${shared.config.dir}/datasource.xml" />
</server>
```

server.xml

- Configurations can be divided into components at any level of granularity
 - From a single file to several
- Can use developer tools to associate configuration snippets with a server configuration
- Visualization through developer tools provides a single logical view
- **Team development:** keep the application and configuration components together

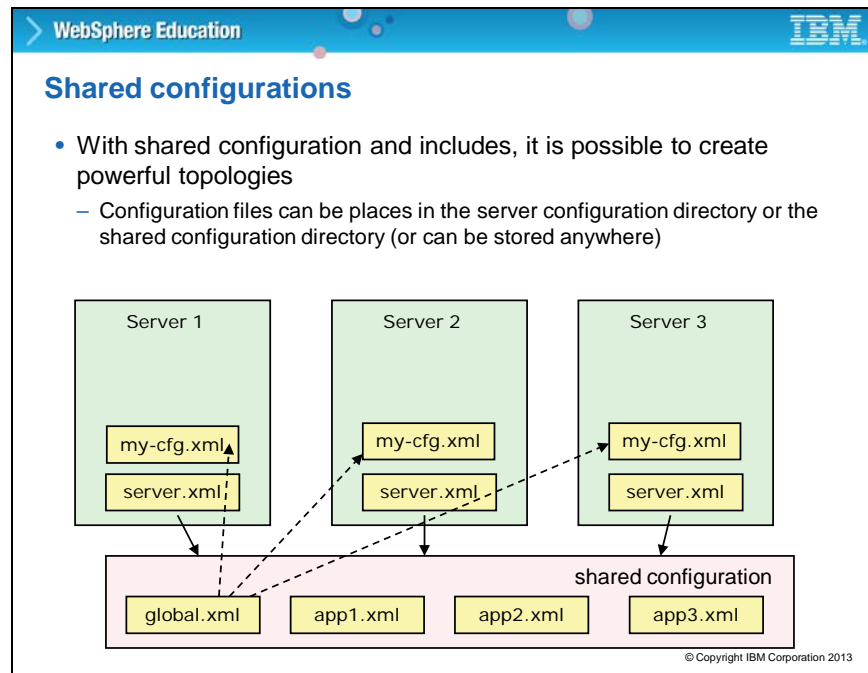
© Copyright IBM Corporation 2013

Title: Flexible configuration

This slide shows an example of how shareable configuration can be accessed in a server.xml file by using the include statement. A URL or environment variable can be used to point to the shareable XML configuration files.

Configurations can be componentized at any level of granularity, from a single XML file to several files.


It is helpful to use WDT (WebSphere Developer Tools) to associate configuration snippets with multiple Liberty profile server configurations. The visualization features of WDT are useful for providing a single logical view. For team development, flexible configuration is helpful for keeping the application and configuration components together.



Title: Shared configurations

With shared configuration and includes, it is possible to create powerful topologies and more easily manage multiple server configurations.

As shown in the diagram on this slide, configuration files can be placed in the server configuration directory, the shared configuration directory, or you can configure any shared file system location. For example, a configuration file named `global.xml` is shared among all servers in an environment. But a local configuration file can be used to customize values that must be unique on the host. The HTTP port numbers must be unique on the host, and clone IDs must be unique across the environment for workload management, so these values can be stored in a local unshared configuration file.

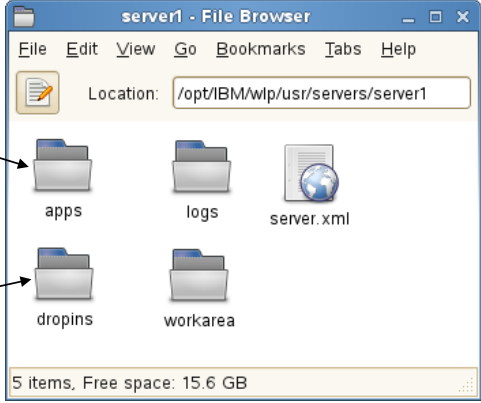
WebSphere Education 

Application deployment

- Applications are deployed by using:
 - Monitored directory (dropins)
 - Configuration (server.xml)
 - Developer tools

Configured applications go here (location can be configured)

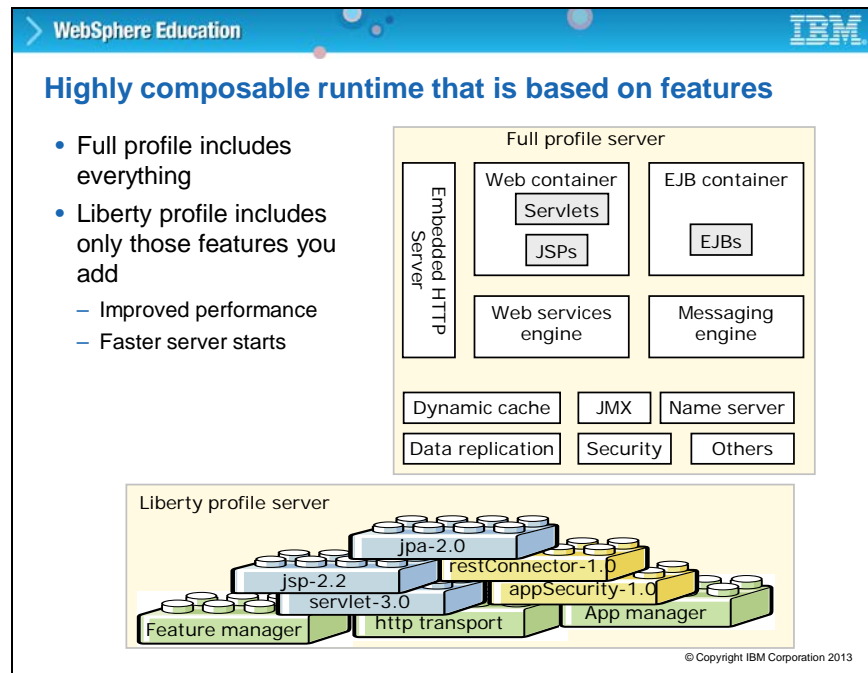
Monitored directory (location can be configured)



© Copyright IBM Corporation 2013

Title: Application deployment

The screen capture on this slide shows different folders in the configuration directory of a Liberty profile server. For application deployment, you can use either the drop-ins folder, which is a monitored directory, or the apps folder which an application configuration element references in the server.xml file.

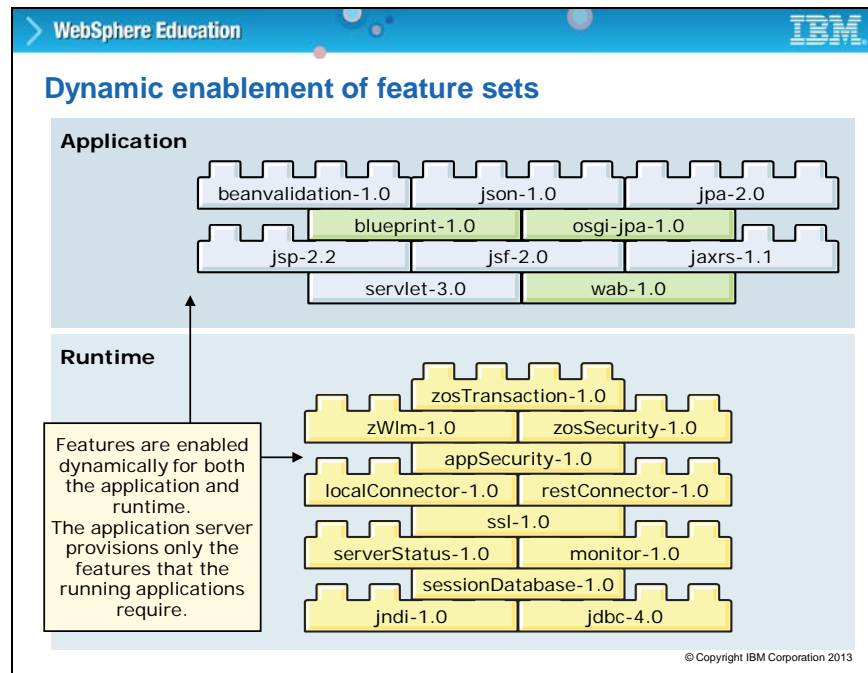


Title: Highly composable run time that is based on features

Application servers in the Full Profile have a JVM shown at the top of this slide that includes all services (features) whether the applications require them or not.

Liberty profile server JVMs start only those features that you add to its server.xml file. By default, a server contains only the jsp- 2.2 feature to support servlet and JSP applications. You use the Liberty profile feature manager to add the features that you need.

The building blocks shown in the graphic on this slide show some of the Liberty profile features that can be defined for the feature manager. These features include, JPA, JSP, servlet, application security, and a remote JMX connector. However, several other features might be configured for a particular server. In addition to the features, you can add HTTP port definitions for the HTTP transport, and application definitions for the application manager.



Title: Dynamic enablement of feature sets

The graphics on this slide show two example feature sets for a Liberty Profile server. All features are dynamically activated when they are saved to the configuration file of the server. No server restart is required. The application features show services that applications require and that run on the server such as bean validation, osgi-jpa, and wab (web application bundle). The runtime features show services that the server runtime itself requires such as serverStatus, monitor, localConnector, restConnector, and ssl.

WebSphere Education
IBM

Class visibility

- Full profile server makes runtime classes visible to applications
- Liberty profile hides runtime classes from applications
- Applications can use open source APIs without the runtime interfering
- Three types of API
 - **spec API**: APIs defined by an external standards group (○)
 - **ibm-api**: Value add APIs provided by IBM (●)
 - **third-party**: APIs provided by open source projects
- By default only **spec** and **ibm-api** are visible to applications
 - third-party can be added by using the `classloader` element

```
<classloader
  allowedApiTypes="spec">
```

© Copyright IBM Corporation 2013

Title: Class visibility

The Full Profile application server exposes runtime classes to applications, but the Liberty profile hides runtime classes from applications.

Applications can use open source classes without interference from the run time.

Three types of API can be enabled through the classloader configuration element.



An external standards group defines the spec API.

The ibm-api are value-added APIs that IBM provides.

The third-party refers to APIs that open source projects provide.

Use the classloader configuration element, and the `apiTypeVisibility` attribute to specify the types of API package the class loader is able to see, as a comma-separated list of any combination of the following APIs: spec, ibm-api, third-party.

By default only spec and ibm-api are exposed to applications, but third-party can be added.



Shared libraries

- Associated with applications
- Move common libraries out of the WAR files

```
<library id="libs">  
  <fileset dir="${shared.resource.dir}/libs"  
    includes="*.jar"/>  
</library>
```

- Share classes between applications

```
<application location="snoop.war">  
  <classloader commonLibraryRef="libs" />  
</application>
```

- Or have an instance per application

```
<application location="snoop.war">  
  <classloader privateLibraryRef="libs" />  
</application>
```

© Copyright IBM Corporation 2013

Title: Shared libraries

The library configuration element can be used to specify shared libraries of classes. If your web applications use common libraries, you can remove the classes from the WAR files and use the fileset attribute to point to the file system location. The first snippet on this slide shows how to use the library configuration element.

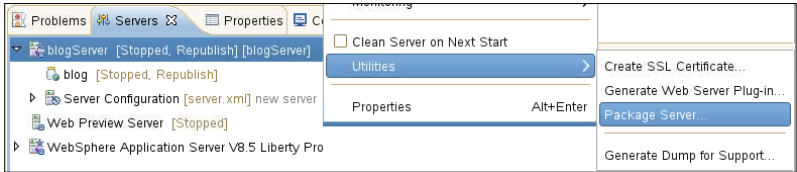
Libraries can also be associated with specific applications by using the classloader feature in the application configuration element. The second snippet on this slide shows how to use the commonLibraryRef attribute to share classes between applications. Library class instances are shared with other class loaders.

The third snippet on this slide shows how to use the privateLibraryRef attribute to isolate classes in an application. Library class instances are unique to this class loader and independent of class instances from other class loaders.

WebSphere Education
IBM

Packaging an application for deployment

- Package an archive of a configured Liberty server along with its applications
 - Directly from Eclipse environment
 - Resulting compressed file can be copied to integration or production environment and uncompressed



The screenshot shows the WebSphere Development Tools (WDT) interface. On the left, a tree view displays the server configuration hierarchy: 'blogServer' (Stopped, Republish), 'blog' (Stopped, Republish), 'Server Configuration [server.xml] new server', 'Web Preview Server' (Stopped), and 'WebSphere Application Server V8.5 Liberty Pro'. On the right, a context menu is open for the 'blogServer' node, showing options like 'Clean Server on Next Start', 'Utilities', 'Properties', 'Alt+Enter', 'Create SSL Certificate...', 'Generate Web Server Plug-in...', 'Package Server...', and 'Generate Dump for Support...'. The 'Package Server...' option is highlighted.

© Copyright IBM Corporation 2013

Title: Packaging an application for deployment

A Liberty profile server configuration and its applications can be packaged into an archive or compressed file. From the WebSphere Development Tools (WDT) or IADT, you can select the **server_name > Utilities > Package Server** to create a compressed file.

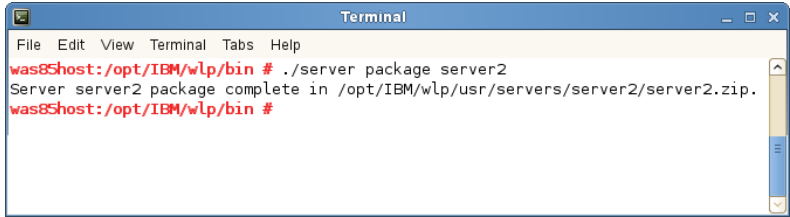
Slide 31

WebSphere Education

IBM

Liberty profile server command-line tools

- For server operations outside of the developers tools, there is a command-line program to manage the lifecycle of server instances and also package it for deployment:
 - Create <server_name>
 - Start and stop <server_name>
 - Package <server_name>
 - Status <server_name>



```
Terminal
File Edit View Terminal Tabs Help
was85host:/opt/IBM/wlp/bin # ./server package server2
Server server2 package complete in /opt/IBM/wlp/usr/servers/server2/server2.zip.
was85host:/opt/IBM/wlp/bin #
```

© Copyright IBM Corporation 2013

Title: Liberty profile server command-line tools

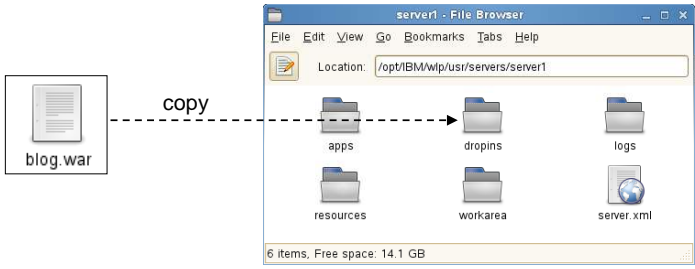
From the command line, you can use the server package command. A compressed file that uses the same server name and a .zip extension is created. In the example on this slide, the compressed file is named server2.zip. The compressed file contains the Liberty Profile runtime directories and the server configuration directory for the specified server.

Slide 32

WebSphere Education IBM

Deploying an application by using the drop-ins directory

- A monitored directory can be used to deploy applications
 - Copy the application file into the **dropins** directory
 - The server installs and starts the application



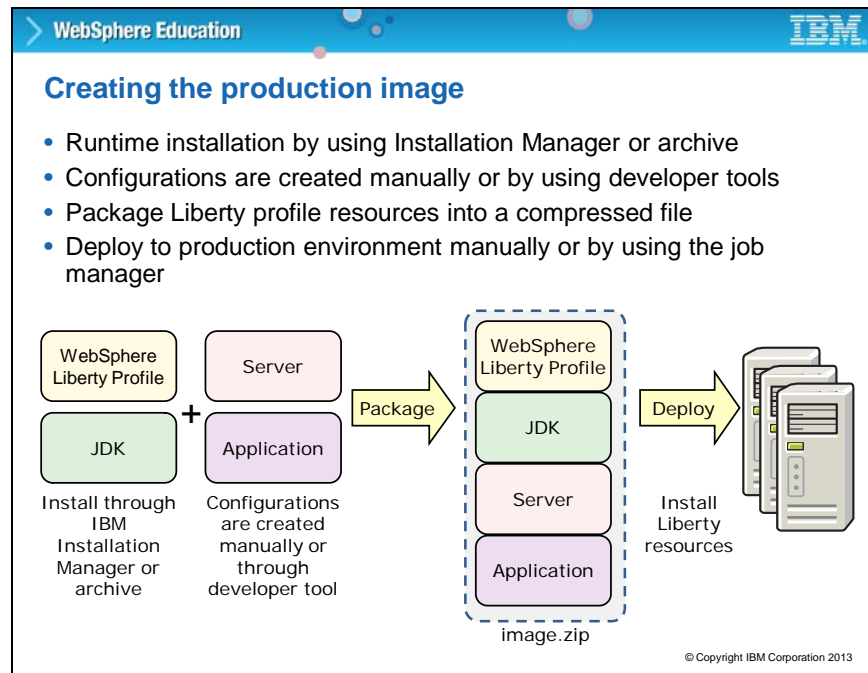
- Removing the file from the dropins directory automatically uninstalls the application

© Copyright IBM Corporation 2013

Title: Deploying an application by using the drop-ins directory

By default, the "drop-ins" directory is automatically monitored. If you drop an application archive file into this directory, the application is automatically deployed on the server. Similarly, if the application file is deleted from the directory, the application is automatically removed from the server. The "drop-ins" directory can be used for applications that do not require any other configuration, such as security role mapping. You do not have to include the application entry or any relevant information in the server configuration. For applications that are not in the "drop-ins" directory, you specify the location by using an application entry in the server configuration. The location can be on the file system, or at a URL.


Three types of dynamic update can be controlled through configuration: changing the server configuration; adding and removing applications; updating installed applications. For all deployed applications, you can configure whether application monitoring is enabled, and how often to check for updates to applications. For the "drop-ins" directory, you can also configure the name and location of the directory, and choose whether to deploy the applications that are in the directory.



Title: Creating the production image

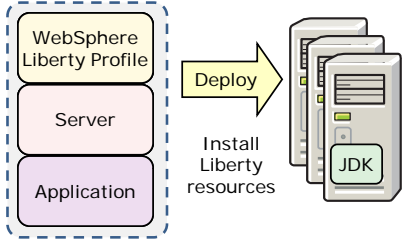
To create a production image that can be deployed to multiple hosts, you must package the Liberty run time, the JDK, server configurations, and application files into a compressed file. If all of these Liberty profile resources are in a single compressed file, the environment that you deploy is said to be “self-contained”. The compressed file can be copied to any host and manually extracted. Alternatively, you can use a job manager to install the Liberty profile resources to a remote target host.

Slide 34

WebSphere Education 

Deploying Liberty topologies

- Choose which parts you must deploy
 - Only the application
 - Only the Liberty profile server
 - The Liberty profile runtime and the server
 - Any combination
- Example: A self-contained topology in which the compressed file contains everything but the JDK
 - The SDK is preinstalled on each host



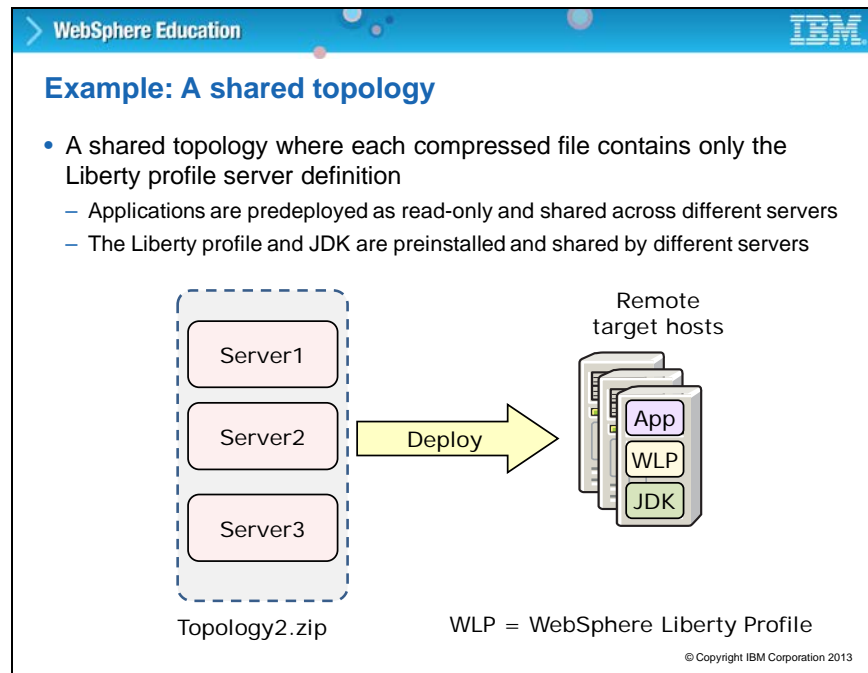
Topology1.zip

© Copyright IBM Corporation 2013

Title: Deploying Liberty topologies

There are multiple scenarios or topologies for deploying Liberty profile resources. In addition to deploying a “self-contained” topology as described on the previous slide, you can choose to have the Liberty runtime and JDK preinstalled on the hosts, and then deploy only the server configurations and applications. Other combinations of “shared topologies” are possible.

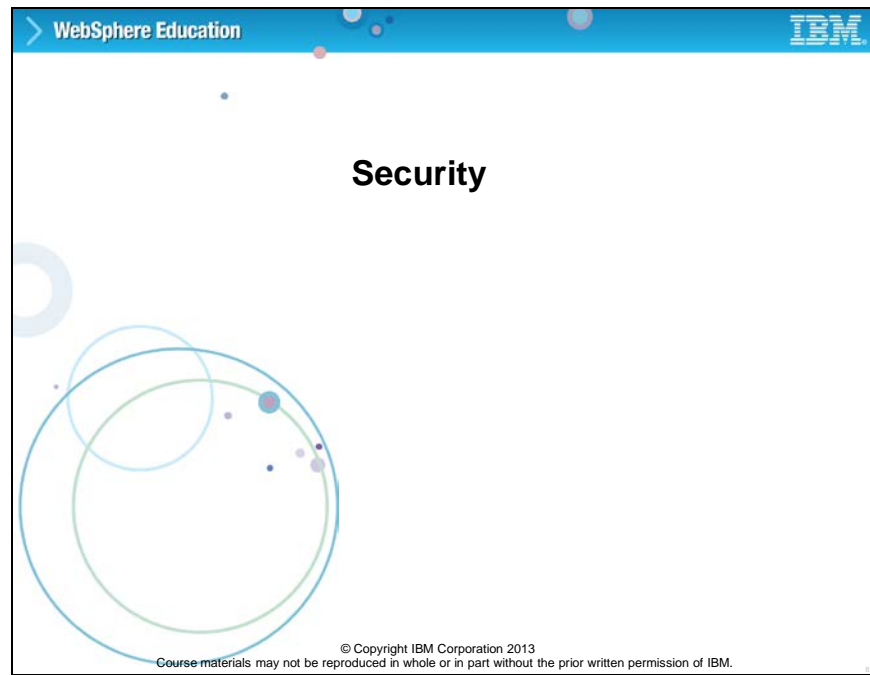
Slide 35

**Title: Example: a shared topology**

The graphic on this slide shows a shared topology where each compressed file contains only the Liberty profile server definition. Applications are predeployed as read-only and shared across different servers. The Liberty profile and JDK are preinstalled and shared among different servers.

A shared topology can be deployed to multiple remote hosts by using the job manager.

Slide 36



Topic: Security

This topic describes different security features that can be used to configure a Liberty profile server, and how to secure an application.

WebSphere Education

IBM

Liberty profile security

- All opened ports are local host only
- No remote management by default
- Seamless transition when enabling security
- Three key security-related features

Feature	Description
ssl-1.0	Includes the SSL-specific code
appSecurity-1.0	Includes all the security services (authentication, registry, authorization) and web-specific security code
zosSecurity-1.0	Includes the SAF registry and authorization code

© Copyright IBM Corporation 2013

Title: Liberty profile security

Security in the Liberty profile supports all the servlet 3.0 security features. In addition, it also secures Java JMX connections. The following server features are applicable to security in the Liberty profile.

The appSecurity-1.0 feature enables security for all web resources.

The ssl-1.0 feature enables SSL connections by using HTTPS.

For z/OS platforms, zosSecurity-1.0 includes the support for SAF Registry and Authorization on the z/OS platform.

In addition, the restConnector-1.0 feature enables remote access by JMX clients through a REST-based connector.

WebSphere Education
IBM

Enable SSL

- Add the SSL feature and provide the keystore password

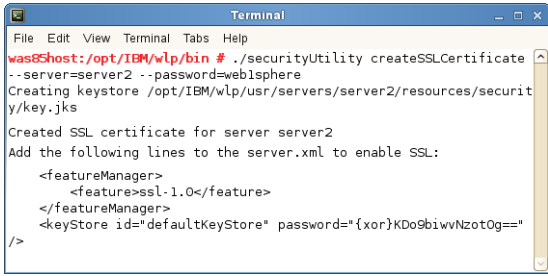
```

<featureManager>
  <feature>ssl-1.0</feature>
</featureManager>

<keyStore id="defaultKeyStore" password="{xor}DFoKyp=" />

```

- Certificate is generated at server startup
- securityUtility can be used to generate a self-signed certificate



```

was8s8host:/opt/IBM/wlp/bin # ./securityUtility createSSLCertificate
--server=server2 --password=weblsphere
Creating keystore /opt/IBM/wlp/usr/servers/server2/resources/security/key.jks
Created SSL certificate for server server2
Add the following lines to the server.xml to enable SSL:
  <featureManager>
    <feature>ssl-1.0</feature>
  </featureManager>
  <keyStore id="defaultKeyStore" password="{xor}KDo9biwvNzot0g=="
/>

```

© Copyright IBM Corporation 2013

Title: Enable SSL

To enable SSL for a server, you must add the ssl-1.0 feature to its configuration file. Also, you must add the keystore configuration element and specify an id and password. You can use the security utility from the command line to generate a self-signed certificate and provide the required configuration data. The screen capture on this slide shows you how to run the security utility and the output it produces. The security utility requires you to specify the server name and a password. The utility creates the keystore under the server configuration directory and provides an encoded version of the password.

WebSphere Education
IBM

Advanced SSL

- Configure per endpoint SSL configuration

```

<featureManager>
  <feature>ssl-1.0</feature>
</featureManager>

<keyStore id="myKeyStore" password="{xor}DFoKyp="
  location="{server.config.dir}/mykeystore.p12"
  type="PKCS12"/>
<keyStore id="myTrustStore" password="{xor}DFoKyp="
  location="{server.config.dir}/mytruststore.p12"
  type="PKCS12"/>
<ssl id="mySSLConfig" keystoreRef="myKeyStore"
  trustStoreRef="myTrustStore"/>

<httpEndpoint id="defaultHttpConfig">
  <sslOptions sslRef="mySSLConfig"/>
</httpEndpoint>

```

© Copyright IBM Corporation 2013

Title: Advanced SSL

The configuration elements in this example show how to configure secure HTTP.

An HTTP endpoint must have an sslOptions attribute, which specifies the SSL protocol options.

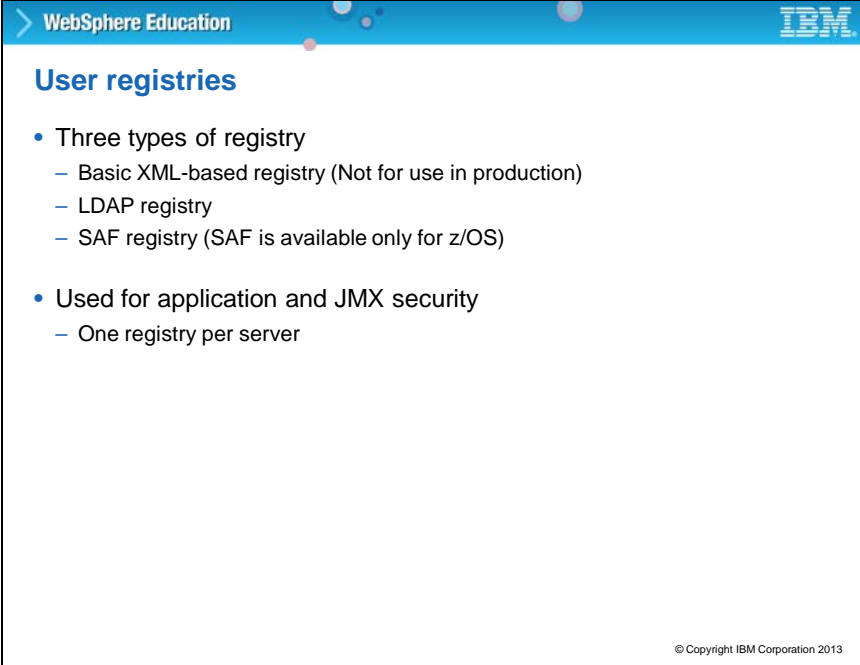
The SSL protocol options in the example use the sslRef attribute, which specifies the SSL configuration to be used as the default.

The following steps define the SSL configuration.

First, the ssl-1.0 feature is enabled. Then, two keystore elements must be configured. One is the keystore for the HTTP endpoint named “myKeyStore” in this example. The other is a truststore for storing certificates of trusted servers named “myTrustStore” in this example. Both of these configuration elements point to keystore files under the configuration directory of the server: mykeystore.p12 and mytruststore.p12.

Next, an ssl configuration, named mySSLConfig in this example, defines a keystoreRef and a truststoreRef in terms of the keystores that were previously configured.

Finally, an httpEndpoint attribute, sslOptions, defines the sslRef in terms of ssl configuration, mySSLConfig.



The slide is titled "User registries" and is part of a "WebSphere Education" presentation. It features a blue header with the IBM logo. The content is organized into a bulleted list. The first bullet point is "Three types of registry", which includes three sub-bullets: "Basic XML-based registry (Not for use in production)", "LDAP registry", and "SAF registry (SAF is available only for z/OS)". The second bullet point is "Used for application and JMX security", which includes one sub-bullet: "One registry per server". A copyright notice "© Copyright IBM Corporation 2013" is located in the bottom right corner of the slide.

- Three types of registry
 - Basic XML-based registry (Not for use in production)
 - LDAP registry
 - SAF registry (SAF is available only for z/OS)
- Used for application and JMX security
 - One registry per server

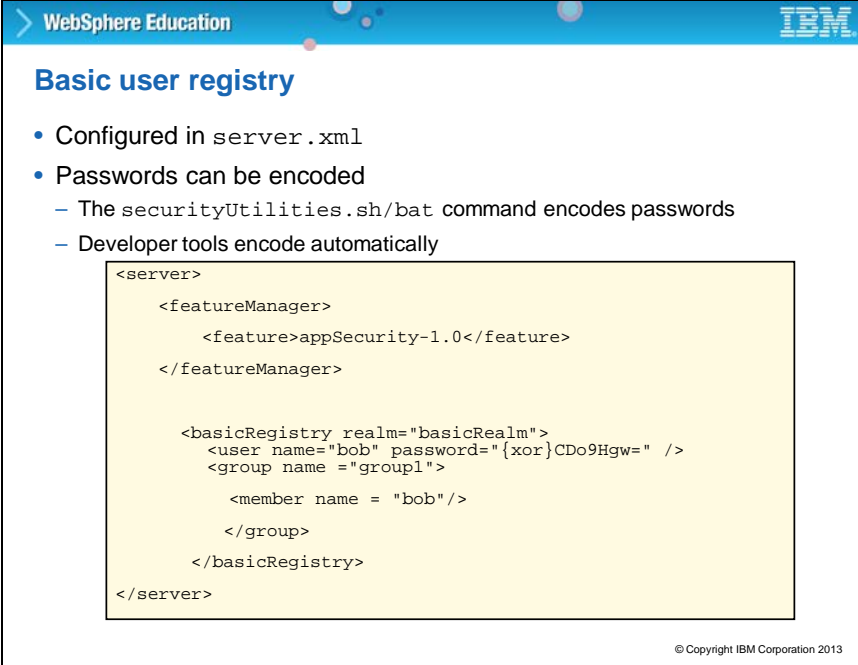
© Copyright IBM Corporation 2013

Title: User registries

The Liberty profile server uses a user registry to authenticate a user and retrieve information about users and groups to perform security-related operations, including authentication and authorization.

When validating the authentication data of a user, the login modules call the user registry that is configured to validate the user information. Liberty profile supports both a Basic XML-based user registry and a more robust LDAP-based repository. For z/OS, a System Authorization Facility (SAF) registry is also supported.

The Liberty profile supports only one user registry per server.



The slide is titled "Basic user registry" and is part of a "WebSphere Education" presentation. It lists two main points: configuration in `server.xml` and password encoding. A code block shows the XML configuration for a basic registry, including a feature manager, a basic registry element with a realm, a user, a group, and a member. The IBM logo is in the top right corner, and a copyright notice is at the bottom right.

WebSphere Education

Basic user registry

- Configured in `server.xml`
- Passwords can be encoded
 - The `securityUtilities.sh/bat` command encodes passwords
 - Developer tools encode automatically



```
<server>
  <featureManager>
    <feature>appSecurity-1.0</feature>
  </featureManager>

  <basicRegistry realm="basicRealm">
    <user name="bob" password="{xor}CDo9Hgw=" />
    <group name="group1">
      <member name="bob"/>
    </group>
  </basicRegistry>
</server>
```

© Copyright IBM Corporation 2013

Title: Basic user registry

A basic registry is configured in the `server.xml` file. The `appSecurity-1.0` feature must be enabled. The `basicRegistry` configuration element is used to define a security realm, users, and groups. Passwords for each user are encoded by using the `securityUtilities` command-line tool. WebSphere Developer Tools (WDT) automatically encodes passwords.



LDAP user registry

- Authenticate by using an LDAP server
- Supports: Microsoft Active Directory, IBM Lotus Domino, Novell eDirectory, IBM Tivoli Directory Server, Sun Java System Directory Server, Netscape Directory Server, IBM SecureWay Directory Server

```
<server>

  <featureManager>

    <feature>appSecurity-1.0</feature>

  </featureManager>

  <ldapRegistry host="myldapserver.ibm.com"
    port="389" baseDN="o=ibm,c=us"
    ldapType="IBM Tivoli Directory Server" />

</server>
```

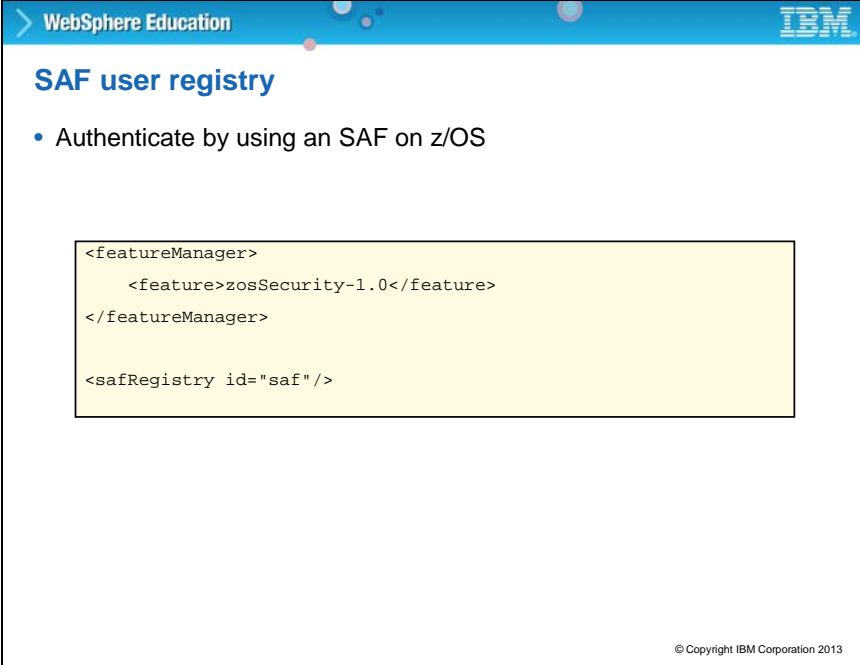
© Copyright IBM Corporation 2013

Title: LDAP user registry

You can use an existing LDAP server for application authentication. To configure an LDAP registry, you add the appSecurity-1.0 server feature to the server.xml file, and specify in the server.xml file the configuration information for connecting to the LDAP server.

Use the ldapRegistry configuration element to specify the LDAP server host name, LDAP port, and base distinguished name (baseDN) for the LDAP directory. The ldapType specifies the LDAP Server product name. Several supported types of LDAP servers are listed on this slide, including IBM Tivoli Directory Server, Microsoft Active Directory, and others.

The example that is shown on this slide is basic. You can use many other attributes to configure the LDAP registry such as: bindDN, bindPassword, userFilter, groupFilter, and others. Also, SSL can be enabled by adding the necessary SSL configuration information.



The slide is titled "SAF user registry" and is part of a "WebSphere Education" presentation. It features a blue header with the IBM logo. A bulleted list indicates the goal is to "Authenticate by using an SAF on z/OS". A yellow box contains the following XML code:

```
<featureManager>
  <feature>zosSecurity-1.0</feature>
</featureManager>

<safRegistry id="saf" />
```

At the bottom right, there is a small copyright notice: "© Copyright IBM Corporation 2013".

Title: SAF user registry

The System Authorization Facility (SAF) registry holds information that is needed to perform security-related functions such as authenticating users and retrieving information about users, groups, or groups that are associated with users. You activate and configure the SAF registry through the server.xml file.



Activate the SAF registry service by adding the zosSecurity-1.0 feature to the server.xml file.

Configure web application security features to use the SAF registry service by adding the appSecurity-1.0 feature.

Configure the SAF registry by using a safRegistry configuration element.

The safRegistry element has the following attributes: ID and realm. The ID uniquely identifies this registry instance. The ID can be anything that you want, but must be unique among other configured registries such as the basic registry and the LDAP registry.

The realm specifies the security realm that is associated with the SAF registry. If you do not specify a realm, the default is the plex name (ECVTSPLX).



Authorization

- Security role mappings are defined in
 - Server configuration
 - `ibm-application-bnd.xml`

```
<application location="secureapp.war">  
  <application-bnd>  
    <security-role name="users">  
      <user name="fred"/>  
      <group name="userGroup"/>  
    </security-role>  
  </application-bnd>  
</application>
```

- SAF



```
<safAuthorization id="saf" />
```

© Copyright IBM Corporation 2013

Title: Authorization

For application authorization, security role mappings must be defined in the `server.xml` file, and the `ibm-application-bnd.xml` file in the application archive.

As part of an application configuration element, use the `application-bind` and `security-role` elements to define users and groups.



Liberty administrative security

- One administrator role
- One user registry for applications and administrators
- Simple configuration for a single administrator user

```
<quickStartSecurity userName="bob"
  userPassword="{xor}Lz4sLCgwLTs" />

<keystore id="DefaultKeyStore"
  password="{xor}DFoKyp=" />
```

- But still easy for multiple users

```
<administrator-role>
  <user>fred</user>
  <group>administratorsGroup</group>
</administrator-role>
```

© Copyright IBM Corporation 2013

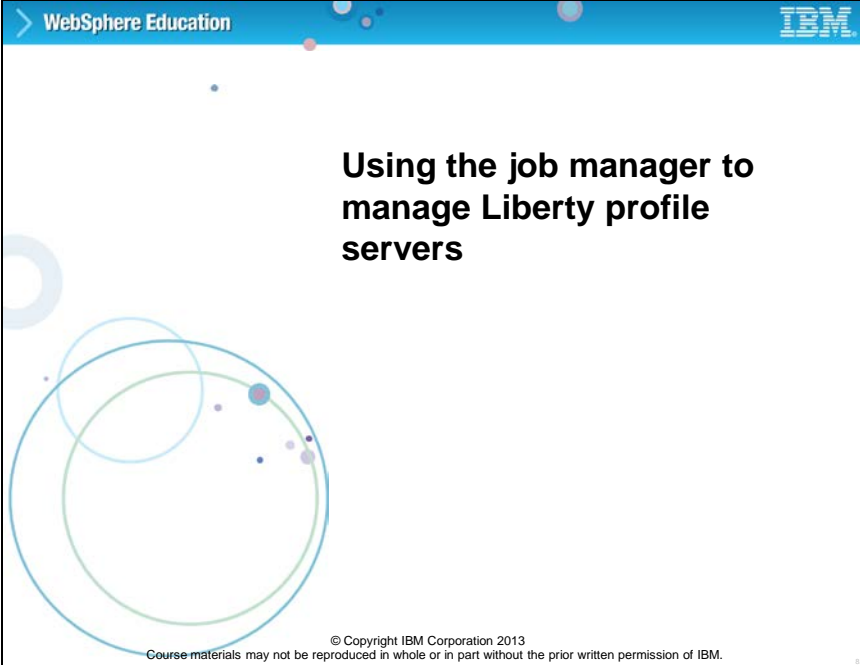
Title: Liberty administrative security

There is only one administrator role for the Liberty profile servers, and a single user registry for both application security and administrative security.

All the JMX methods and MBeans accessed through the REST connector are currently protected with a single role named "administrator". To get started quickly, use the quickStartSecurity element to configure a single user with administrator role and configure the default SSL configuration. If you require only a single administrative user, you can use the quickStartSecurity configuration element. The required attributes are a user name and password.

For authorizing multiple users of administrative functions, use the administratorRole element in the server.xml file to map the users to the administrator role.

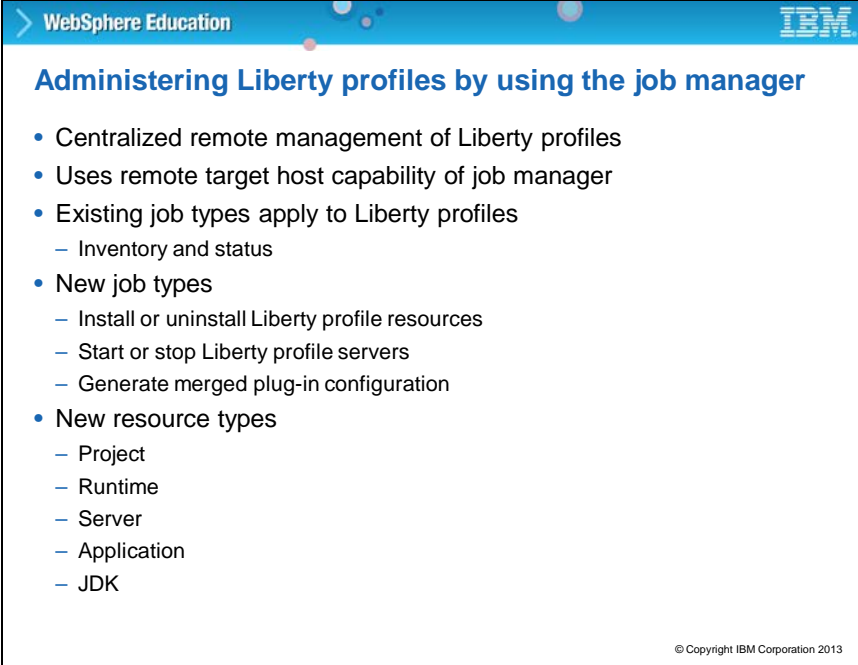
Slide 46



The slide features a blue header bar with the text 'WebSphere Education' on the left and the 'IBM' logo on the right. The main title, 'Using the job manager to manage Liberty profile servers', is centered in a bold, black font. Below the title, there is a decorative graphic consisting of several overlapping circles in light blue and green, with small colored dots scattered around them. At the bottom of the slide, a small copyright notice reads: '© Copyright IBM Corporation 2013. Course materials may not be reproduced in whole or in part without the prior written permission of IBM.'

Topic: Using the job manager to manage Liberty profile servers

This topic presents how the job manager is used to administer Liberty profile servers.



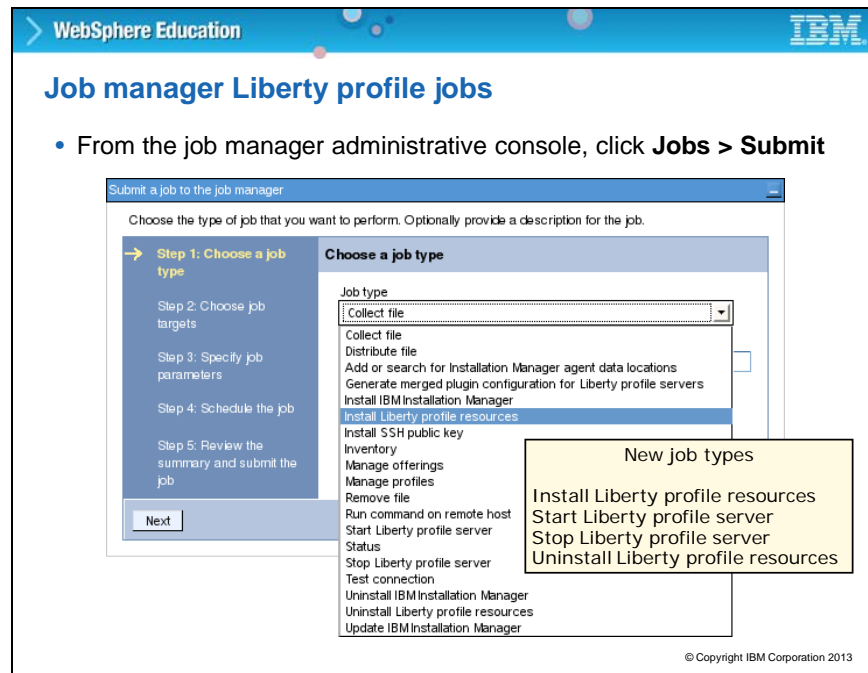
The slide is titled "Administering Liberty profiles by using the job manager" and is part of a "WebSphere Education" presentation. It lists several capabilities of the job manager for Liberty profiles:

- Centralized remote management of Liberty profiles
- Uses remote target host capability of job manager
- Existing job types apply to Liberty profiles
 - Inventory and status
- New job types
 - Install or uninstall Liberty profile resources
 - Start or stop Liberty profile servers
 - Generate merged plug-in configuration
- New resource types
 - Project
 - Runtime
 - Server
 - Application
 - JDK

© Copyright IBM Corporation 2013

Title: Administering Liberty profiles by using the job manager

A job manager environment consists of a job manager and the targets that it manages. The job manager targets can be deployment managers, stand-alone application server nodes that administrative agents manage, and host computers. Setting up a job manager environment involves creating a job manager profile and any other profiles that are needed for the environment. The clocks are then synchronized on all environment computers, and then the targets are registered with the job manager.

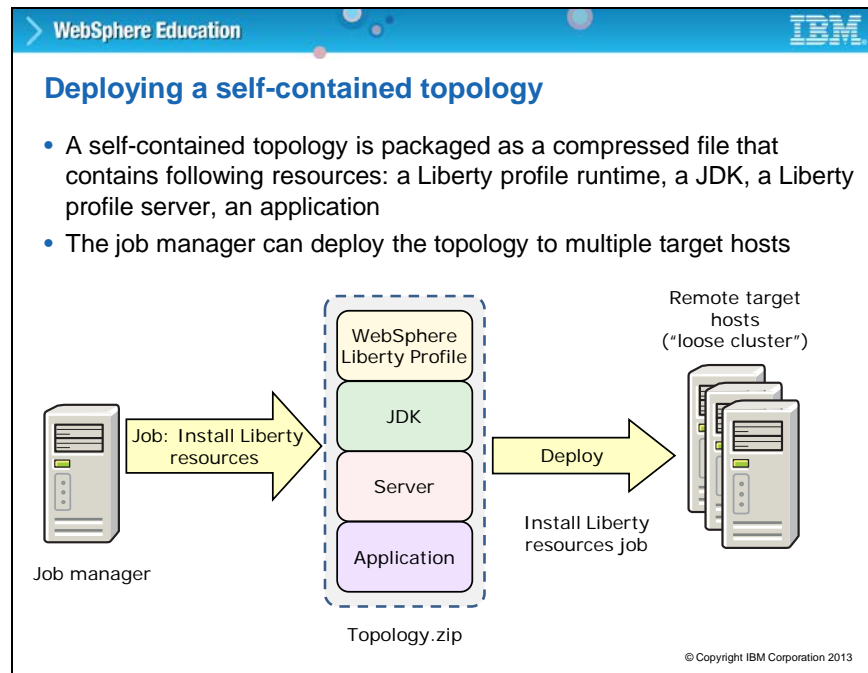


Title: Job manager Liberty profile jobs

You can submit the **Install Liberty profile resources** job to extract resources in a Liberty profile image to destination directories relative to a root directory.

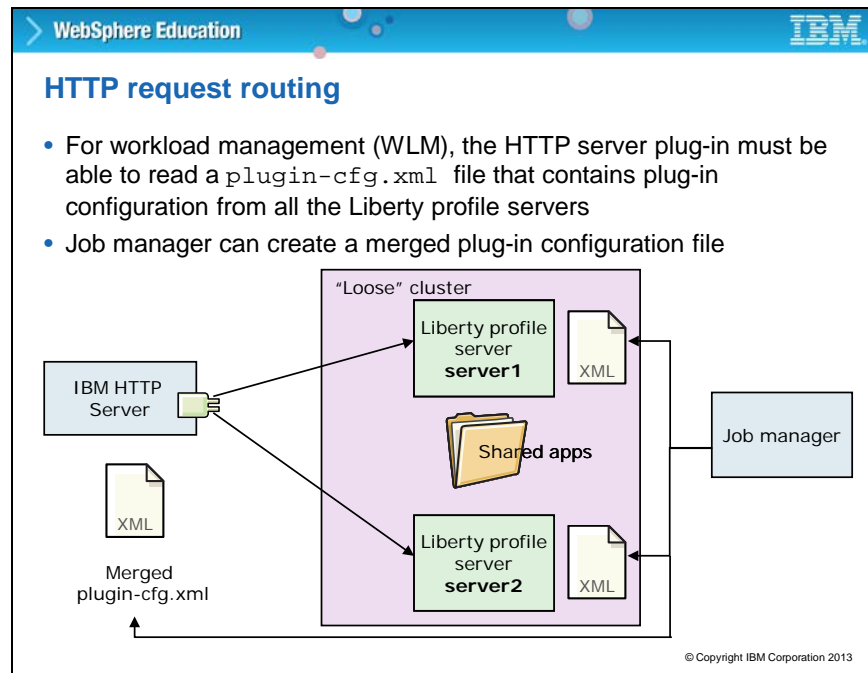
Before running the **Install Liberty profile resources** job, the following conditions must exist:

- The job manager must be running.
- A host computer must be registered with the job manager.
- The image, a compressed file, must contain Liberty profile resources in a directory structure that satisfies job manager rules.
- The root directory to install the resources on the target host must be defined. At minimum, set the WLP_WORKING_DIR variable to a valid directory that is on a target host.
- To install the resources to a shared directory on the target host, you must set the WLP_SHARED_DIR variable to a valid directory.

**Title: Deploying a self-contained topology**

The self-contained topology can be deployed to multiple remote hosts by using the job manager.

One advantage to this topology is that no additional resources are required on the remote hosts.

**Title: HTTP request routing**

When the job manager completes the **Generate merged plug-in configuration** job, it generates a `plugin-cfg.xml` file for each server that is specified, and merges these files into a single file. The merged `plugin-cfg.xml` file can then be accessed with the HTTP server plug-in to route requests to the Liberty profile servers.

WebSphere Education IBM

Job type: Generate merged plug-in configuration

- Use the **Find** facility to specify the resource ID for one server
- Use pattern matching to specify multiple servers
- Servers must be started

Submit a job to the job manager

Step 1: Choose a job type

Step 2: Choose job targets

→ **Step 3: Specify job parameters**

Step 4: Schedule the job

Step 5: Review the summary and submit the job

Specify job parameters

Job type: Generate merged plugin configuration for Liberty profile servers

* Server(s)

User name

Password

Confirm password

© Copyright IBM Corporation 2013

Title: Job type: Generate merged plug-in configuration

Specify the resource name or resource ID of the server to generate a plug-in for. The server or servers must be started for this job to complete successfully.

This job requires a JMX connector or REST connector configuration in the Liberty profile server. Add the feature **localConnector-1.0** to the server.xml file for each server.

The localConnector-1.0 feature provides a local JMX connector that is built into the JVM. It can be used only on the same host machine by someone who is running under the same user ID and the same JDK. It enables local access by JMX clients such as jConsole, or other JMX clients that use the Attach API.

The restConnector-1.0 feature provides a secure JMX connector that can be used locally or remotely when using any JDK. It enables remote access by JMX clients by using a REST-based connector and requires SSL and basic user security configuration.

WebSphere Education

IBM

Merged plugin-cfg.xml

```
<ServerCluster CloneSeparatorChange="false" GetDWLMTable="false"
  IgnoreAffinityRequests="true" LoadBalance="Round Robin"
  Name="Shared_2_Cluster_0" PostBufferSize="64" PostSizeLimit="-1"
  RemoveSpecialHeaders="true" RetryInterval="60">
  <Server CloneID="e9da8378-7892-4993-8ce0-7e838e6bf6d2"
    ConnectTimeout="0" ExtendedHandshake="false"
    MaxConnections="-1" Name="default_node_defaultServer0_1"
    ServerIOTimeout="900" WaitForContinue="false">
    <Transport Hostname="was85host" Port="9041" Protocol="http"/>
  </Server>
  <Server CloneID="306b1428-4119-4b9d-bae0-e7cc0cc5e0a8"
    ConnectTimeout="0" ExtendedHandshake="false"
    MaxConnections="-1" Name="default_node_defaultServer0_0"
    ServerIOTimeout="900" WaitForContinue="false">
    <Transport Hostname="was85host" Port="9040" Protocol="http"/>
  </Server>
  . . .
</ServerCluster>
```

© Copyright IBM Corporation 2013

Title: Merged plugin-cfg.xml

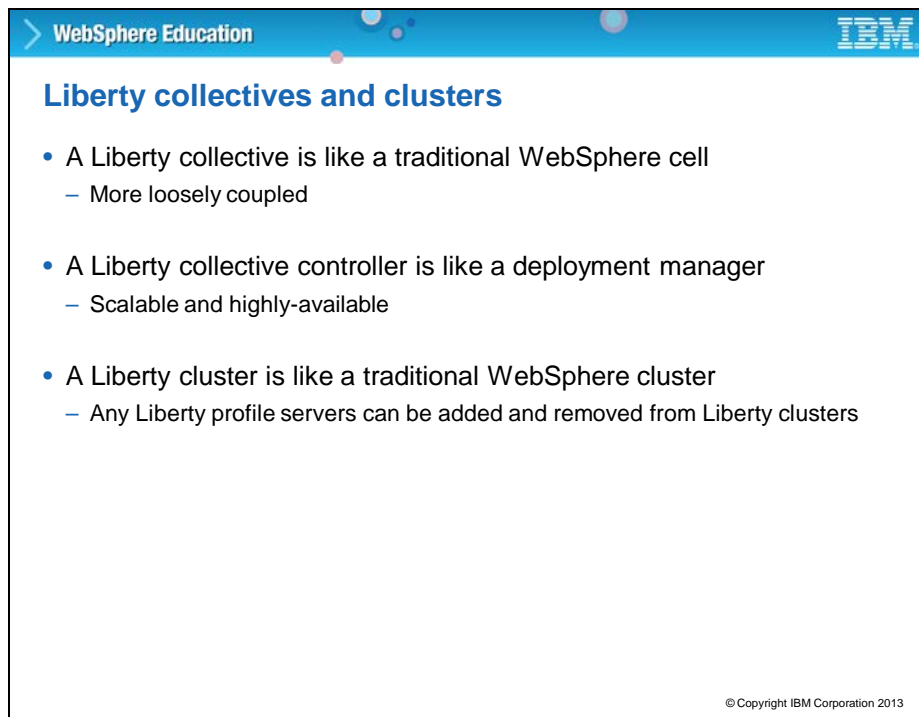
This slide shows a snippet of the merged plugin-cfg.xml file. Notice that it contains the Clone IDs for two servers, server1, and server2.

Slide 53



This topic presents Liberty collectives and clusters.

Slide 54



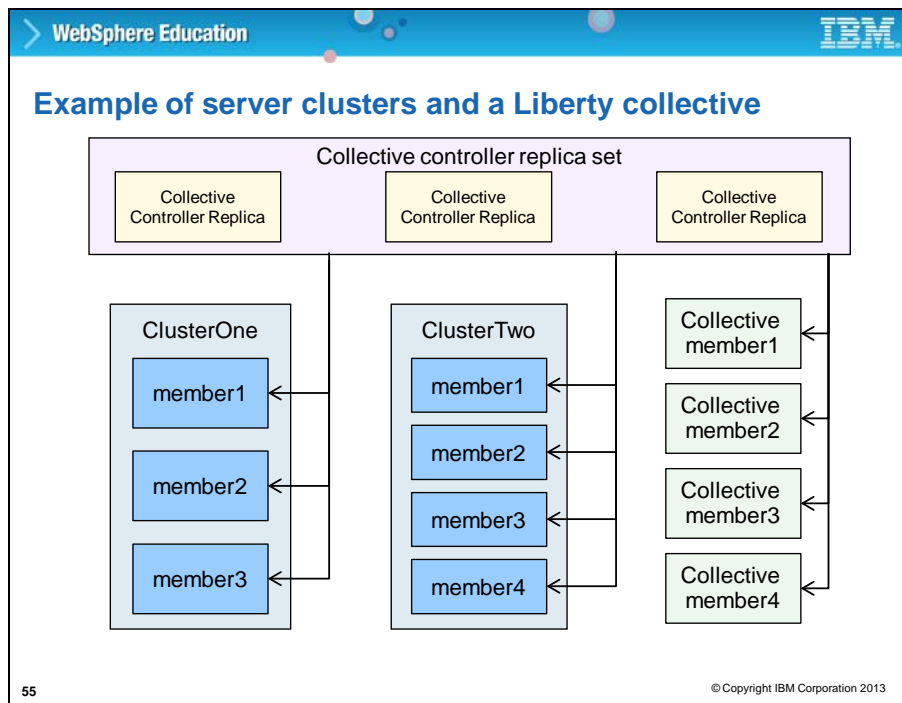
The slide is titled "WebSphere Education" in the top left corner and features the IBM logo in the top right corner. The main title is "Liberty collectives and clusters". Below the title, there is a bulleted list of three points:

- A Liberty collective is like a traditional WebSphere cell
 - More loosely coupled
- A Liberty collective controller is like a deployment manager
 - Scalable and highly-available
- A Liberty cluster is like a traditional WebSphere cluster
 - Any Liberty profile servers can be added and removed from Liberty clusters

At the bottom right of the slide, there is a small copyright notice: "© Copyright IBM Corporation 2013".



A Liberty collective is like a traditional WebSphere cell, but much more loosely coupled. A collective controller is similar to a deployment manager, and a Liberty cluster is similar to a traditional cluster.

Slide 55



Here is a diagram of a possible topology for a Liberty collective. For scalability and availability, there can be multiple controllers for the same collective. Liberty profile servers can be part of a collective whether they are in clusters or not.

Slide 56





Administering Liberty profiles in collectives

- A Liberty collective is an administrative domain for Liberty profiles
 - Analogous to a traditional WebSphere cell
- Standards-based administration API
 - Built on JMX (MBeans)
 - Works with common tools (Jconsole, Jython, and so on)
- Loosely-coupled
 - No centralized master configuration
 - No nodeagents
- Management server is called a collective controller
 - Analogous to a deployment manager
 - Application servers cache sparse configuration data and state in the controller
 - Application servers own their own configuration
- Scalable and resilient
 - Can have multiple controller servers for the same collective
 - Replicate member state and configuration data between collective controllers

© Copyright IBM Corporation 2013

The Liberty collective is more loosely coupled than a traditional WebSphere cell, in that there is no central configuration repository, and no nodeagents must be kept synchronized. A collective controller is similar to a deployment manager, but rather than pushing configuration out to servers, the collective controller accepts configuration information from the servers. There can also be multiple collective controllers for the same collective. The multiple collective controllers replicate information between them. A Liberty collective is administered with the JMX standard.

Slide 57



Liberty profile collective controller and collective members

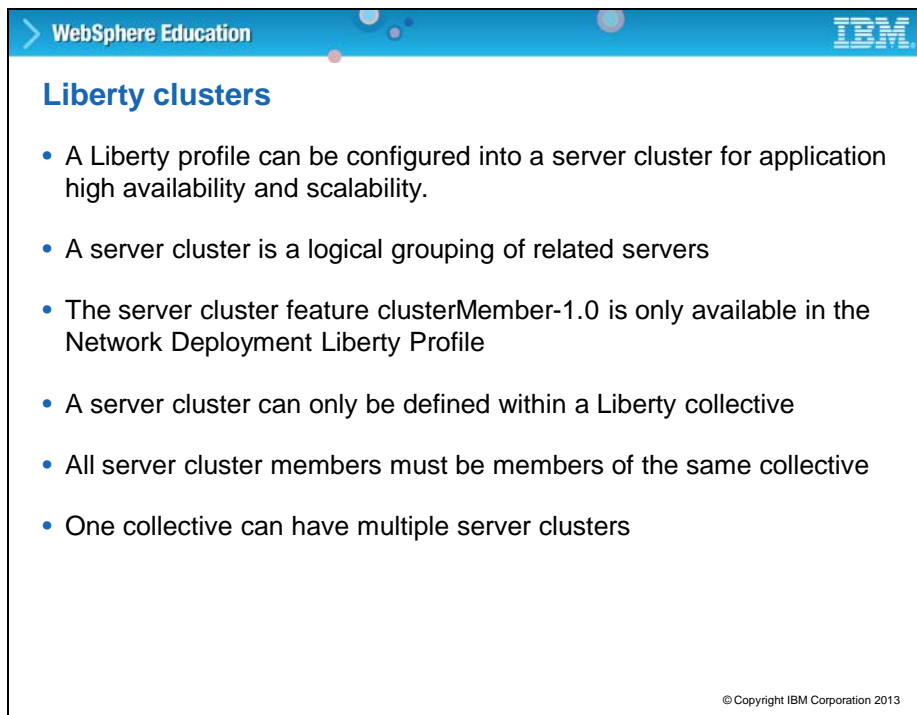
- Collective controller
 - A Liberty server with the collectiveController feature enabled
 - Network Deployment Liberty Profile only
 - Caches the configuration and state of collective members
 - Send commands to collective members
 - Multiple collective controller servers can manage the same collective
- Collective member
 - A Liberty server with the collectiveMember feature enabled
 - All Liberty Profile editions
 - Sends configuration data and state to one of the collective controllers
 - Accepts commands from collective controller servers

© Copyright IBM Corporation 2013

The collectiveController feature for a Liberty profile server is available only in the Network Deployment edition. The collective controller is used to send commands to the collective members.

The collectiveMember feature is available in all Liberty editions. The members of a collective send their state to the collective controllers and accept commands from the collective controllers.

Slide 58

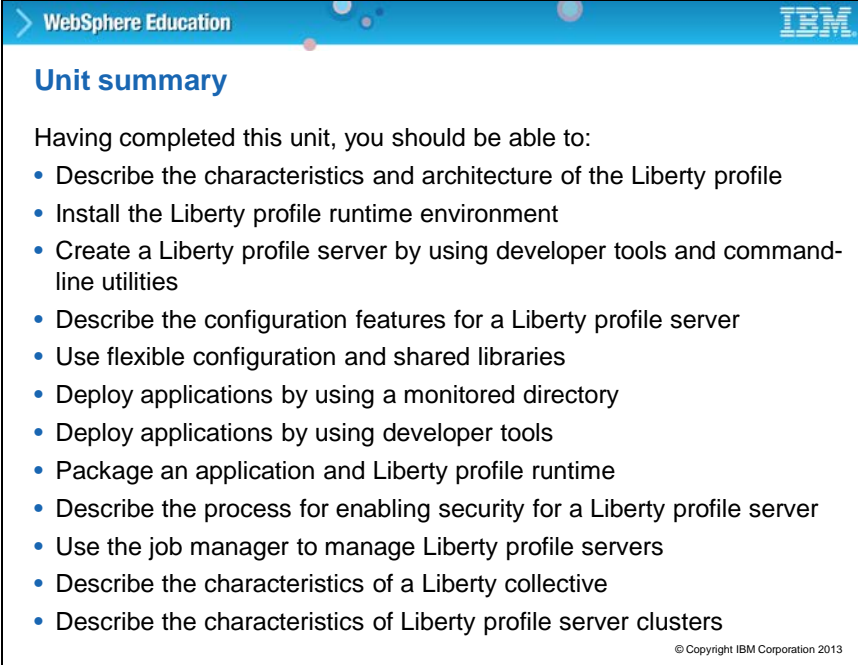


The slide is titled "Liberty clusters" and is part of a "WebSphere Education" presentation. It features a blue header with the IBM logo. The content is a bulleted list of six points regarding Liberty clusters. The footer contains a copyright notice for IBM Corporation 2013.

- A Liberty profile can be configured into a server cluster for application high availability and scalability.
- A server cluster is a logical grouping of related servers
- The server cluster feature clusterMember-1.0 is only available in the Network Deployment Liberty Profile
- A server cluster can only be defined within a Liberty collective
- All server cluster members must be members of the same collective
- One collective can have multiple server clusters

© Copyright IBM Corporation 2013

The clusterMember feature is available only in the Network Deployment edition. Cluster members must be part of a Liberty collective. Liberty profile servers can be clustered according to any logical grouping.



The slide is titled 'Unit summary' and is part of a 'WebSphere Education' presentation, as indicated by the header. It lists 13 learning objectives for the unit. The IBM logo is in the top right corner. A small copyright notice '© Copyright IBM Corporation 2013' is at the bottom right of the slide content area.

Unit summary

Having completed this unit, you should be able to:

- Describe the characteristics and architecture of the Liberty profile
- Install the Liberty profile runtime environment
- Create a Liberty profile server by using developer tools and command-line utilities
- Describe the configuration features for a Liberty profile server
- Use flexible configuration and shared libraries
- Deploy applications by using a monitored directory
- Deploy applications by using developer tools
- Package an application and Liberty profile runtime
- Describe the process for enabling security for a Liberty profile server
- Use the job manager to manage Liberty profile servers
- Describe the characteristics of a Liberty collective
- Describe the characteristics of Liberty profile server clusters

© Copyright IBM Corporation 2013

Title: Unit summary

Having completed this unit, you should be able to:

- Describe the characteristics and architecture of the Liberty profile
- Install the Liberty profile runtime environment
- Create a Liberty profile server by using developer tools and command-line utilities
- Describe the configuration features for a Liberty profile server
- Use flexible configuration and shared libraries
- Deploy applications by using a monitored directory
- Deploy applications by using developer tools
- Package an application and Liberty profile runtime
- Describe the process for enabling security for a Liberty profile server
- Use the job manager to manage Liberty profile servers
- Describe the characteristics of a Liberty collective
- Describe the characteristics of Liberty profile server clusters