Course Exercises

Design Application Deployment with IBM UrbanCode Deploy

Course code ZQ410 ERC 1.0

IBM Training

May 2017 edition

Contents

About these exercises

IBM UrbanCode Deploy helps you to plan and automate the deployment of complex applications to development, test, and production environments. It can deploy applications as often as needed, helping you to find problems early in the release cycle and provide predictability late in the release cycle.

In these exercises, you deploy a simple web application with IBM UrbanCode Deploy. You create components, create an application that contains those components, and then deploy the components to an environment.

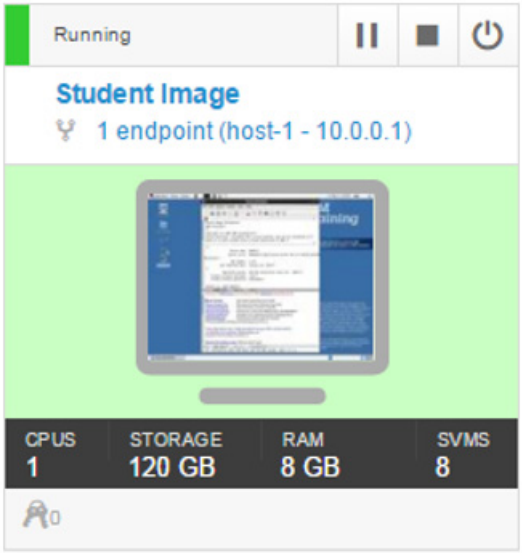
The web application that you deploy is a simple three-tiered web application with a database, web interface, and application logic. You import these three components to the IBM UrbanCode Deploy server and create a single application that deploys these three parts. Then, you create automation instructions that are called processes that describe how to deploy each component. Finally, you use those processes to deploy the components automatically to a target system.

The JPetStore sample application includes three components:

- The application component includes the logic of the application. This component consists of a single web archive file that is named JPetStore.war.
- The web component includes the static web content for the application, including images for the web interface.
- The database component includes the database for the application and scripts that upgrade the database schema to new versions.

Login instructions for this lab

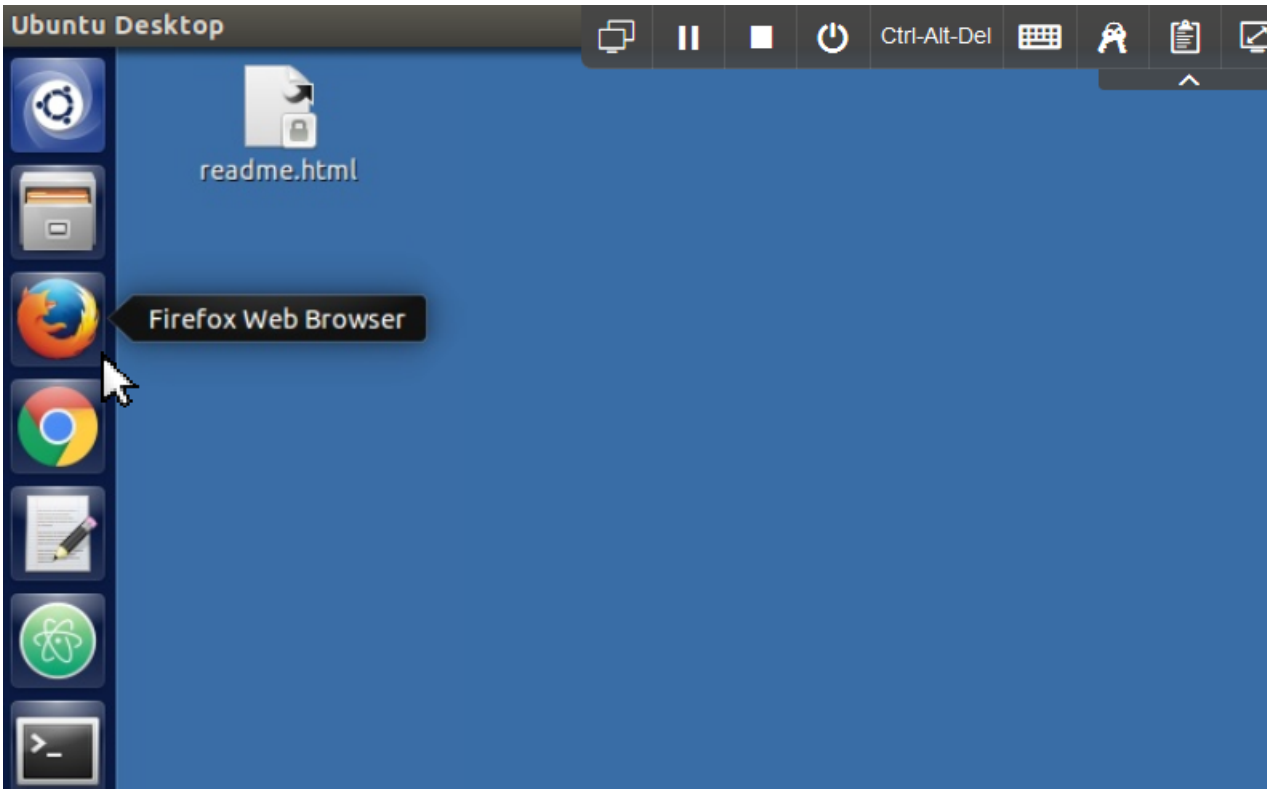
1. To launch the lab environment, click **Start Exercise** after completing each unit.
2. After the student image is in a *Running* state, click the thumbnail image to access the student image.



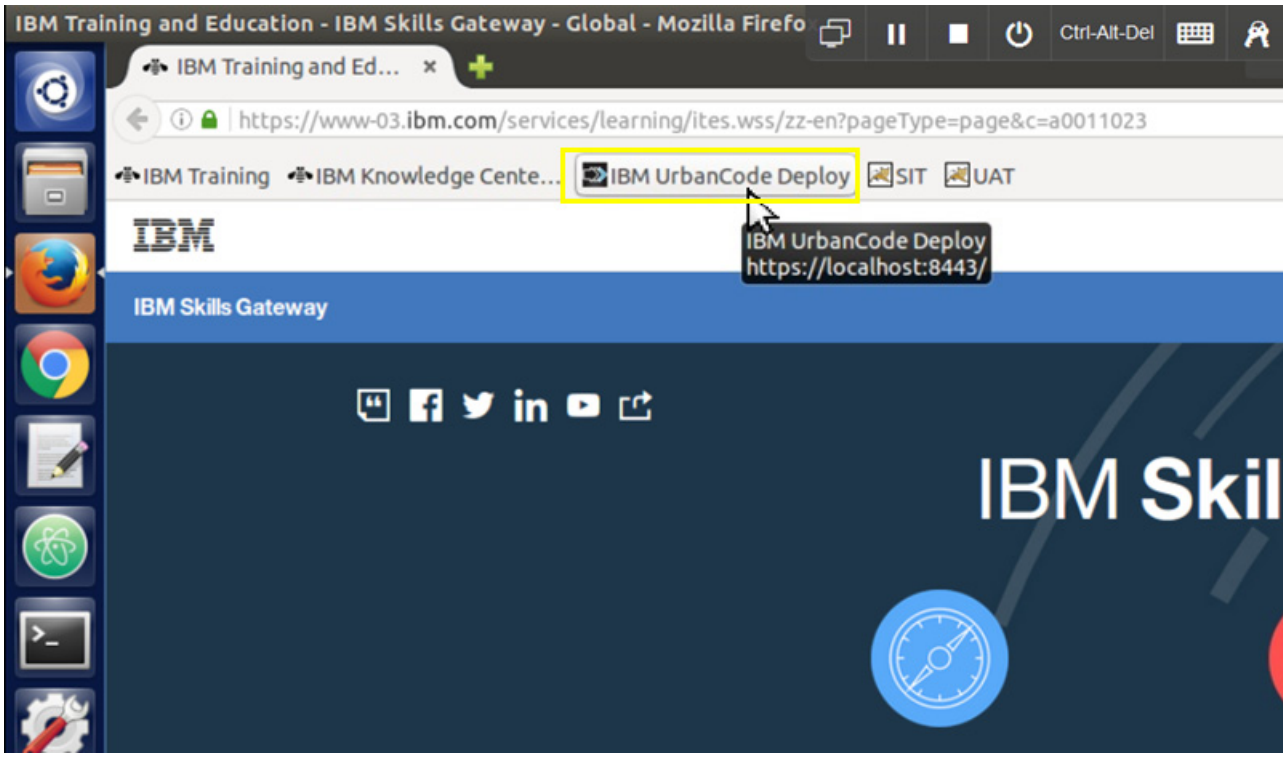
3. When prompted, use these credentials to log on to the student image:
   a. User name: `localuser`
   b. Password: `passw0rd`

Note: Notice that the *o* in password is a zero.

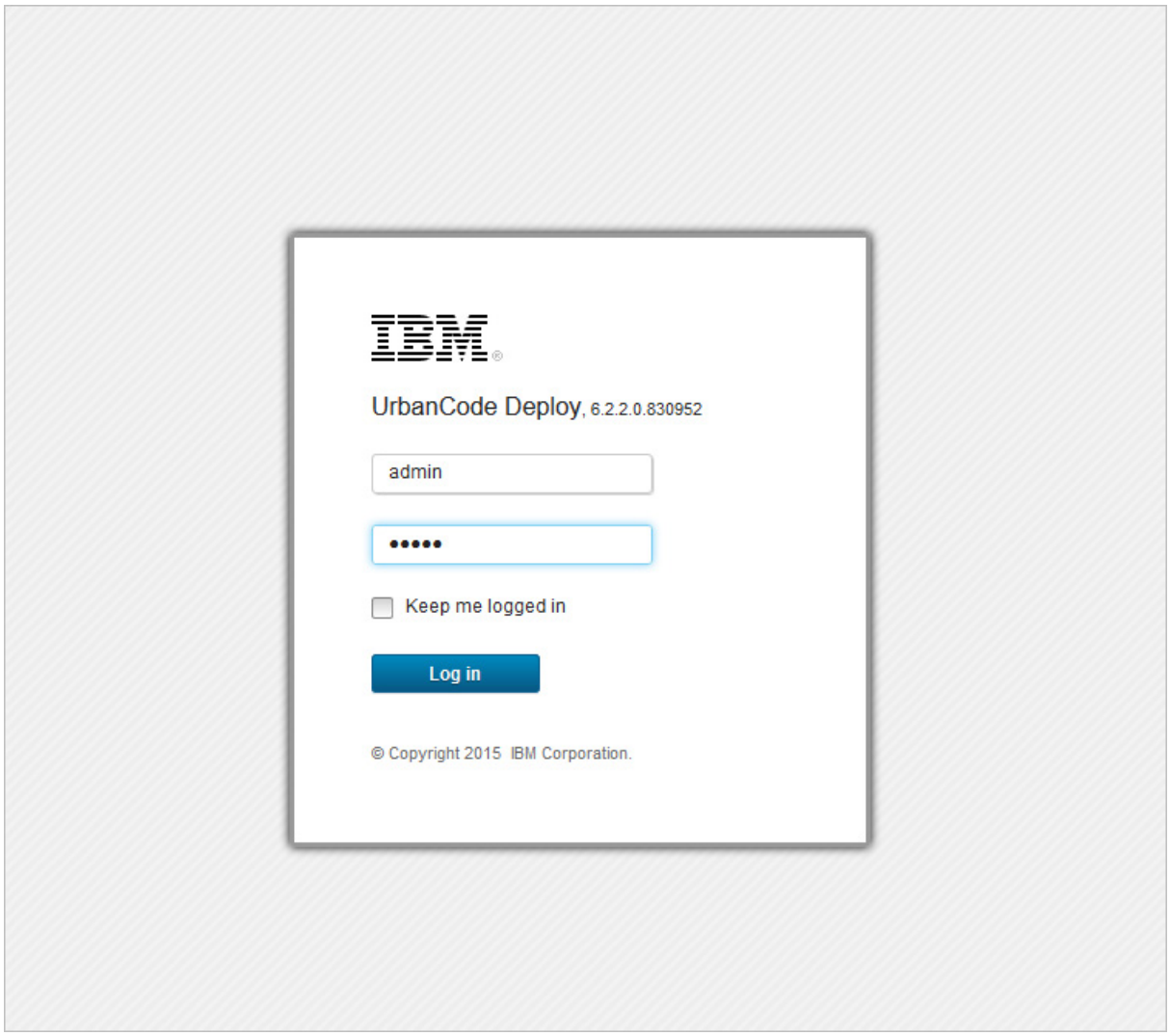
4. From the desktop, open the Firefox browser.



5. To open IBM UrbanCode Deploy in the browser, click the **IBM UrbanCode Deploy** bookmark in the toolbar:



6. At the IBM UrbanCode Deploy log in page, type `admin` in the **User Name** field and `admin` in the **Password** field, and click **Log In**..



Unit 1 Designing software deployment and deploying applications exercises

This unit has no student exercises.

Unit 2 Configuring components and component processes exercises

Components contain artifacts and processes. Artifacts include runnable files, images, databases, configuration instructions, and anything else that is associated with a software project. Processes define the activities that components can perform.

In this exercise, you import artifacts from a build to create components and create processes for deploying the components. You work with the process step types and logic that are used to deploy, install, uninstall, or update components. You create a component template that includes reusable component processes and properties.

After completing these exercises, you will be able to complete these tasks:

- Create components and add user-defined properties to them.
- Import artifacts to create component versions.
- Create component processes.

Creating components

In many cases, artifacts are added to a component by connecting the IBM UrbanCode Deploy server to a computer system that hosts the artifacts. The server can import artifacts from build systems, source-code management systems, and file systems. Imported artifacts are stored in CodeStation, the artifact repository. The component artifacts for this exercise, the JPetStore sample application, come from the file system where the IBM UrbanCode Deploy agent is located.

Hint: The student VM includes the completed files for the components. If you want to skip the creation of one or more components, you can import these files. From the **Components** tab, click **Import Components**, and browse to **Documents** > **Student Files**. You can choose to import the application component (JPetStore-APP.json), the database component (JPetStoreDB.json), and the web component (JPetStore-WEB.json). Be sure to continue with exercise 4 of this unit, whether you created the components manually or imported them.

Exercise 1 Creating the application component

In the exercise, you create the application component that is based on files in the file system, and then you import the initial version into CodeStation, the artifact repository. The application component includes a web archive file that is named JPetStore.war.

If you are not already logged in to the IBM UrbanCode Deploy server, log in with the user name of `admin` and the password of `admin`.

1. Create the application component:
   a. Click the **Components** tab, and click **Create Component.** In this window, you specify where the artifacts for the component are.
   b. In the **Name** field, type `JPetStore-APP`.
   c. In the **Source Configuration Type** list, select **File System (Versioned)**.
   d. In the **Base Path** field, specify the location of the app folder on the server as `/opt/ibm-ucd/agent/opt/JPetStore/shared/app`.
   e. Accept the default values for the other fields on the page.
   f. Click **Save**.
2. Import the component version:
   a. In the **JPetStore-APP** component, click the **Versions** tab.
   b. Click **Import New Versions**.

The import process is displayed in the Currently Running Version Imports section. When the process is finished, the server creates versions of the component based on the folders in the `app` folder. In this case, the server creates a single version of the component that is based on the `app/1.0` folder. Importing versions of small components like this one typically takes a few seconds.

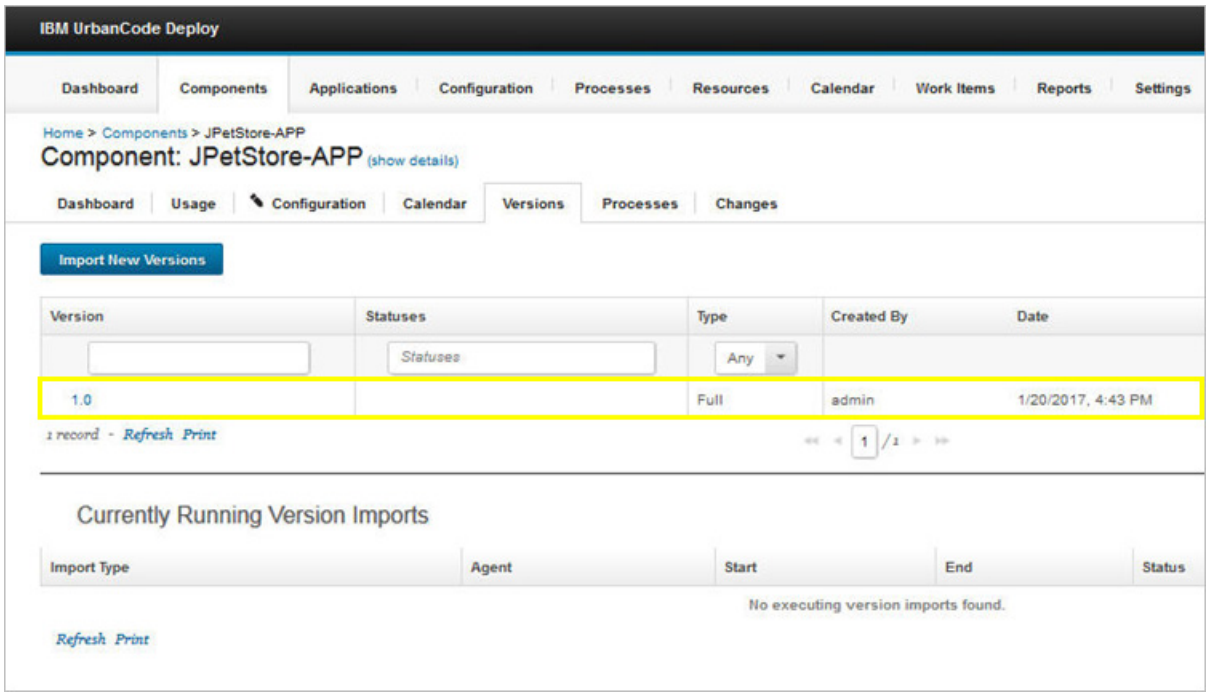
You can track the progress by clicking the **Output Log** icon.

The Output log is similar to this example:

c. Close the log window again.

You might have to refresh the page or click the **Refresh** link at the bottom of the table to see the new version.

d. Verify that the list of versions includes version 1.0 of the component is shown, as in the following figure:



e. If importing the version failed, make sure that the files are available on the agent computer and that the location of these files is correct on the component **Configuration** tab.

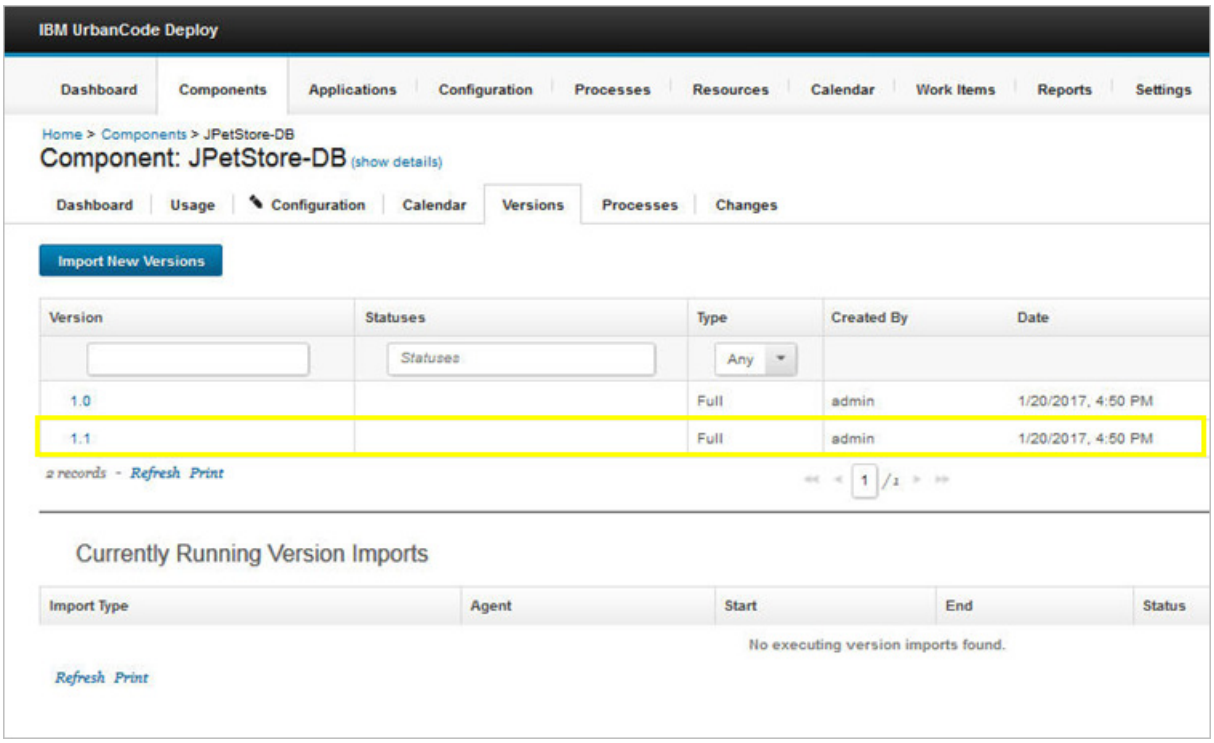
The component is ready to be used in one or more applications.

Exercise 2 Creating the database component

In this exercise, you create the database component in the way that created the application component. The database component includes the database for the application and scripts that upgrade the database schema to new versions.

1. Create the database component:
   a. Click the **Components** tab, and click **Create Component** again.
   b. Specify the name to be `JPetStore-DB`.
   c. In the **Source Configuration Type** list, select **File System (Versioned)**.
   d. In the **Base Path** field, specify the location of the database folder on the server, such as `/opt/ibm-ucd/agent/opt/JPetStore/shared/db`.
   e. Accept the default values for the other fields on the page.
   f. Click **Save**.

2. Import the component versions:
   a. Click the **Versions** tab, and click **Import New Versions.**
   b. Optionally you can repeat the Output log and refresh step as in the previous exercise.
      The server shows two versions of the database component.



Exercise 3 Creating the web component

In this exercise, you create the web component in the way that you created the application and database components. The web component includes the static web content for the application, including images for the web interface.

Create the web component and import the component versions:

1. Click the **Components** tab, and click **Create Component**.
2. Specify the name to be `JPetStore-WEB`.
3. In the **Source Configuration Type** list, select **File System (Versioned)**.
4. In the **Base Path** field, specify the location of the `web` folder on the server, such as `/opt/ibm-ucd/agent/opt/JPetStore/shared/web`.
5. Accept the default values for the other fields on the page.
6. Click **Save**.
7. Click the **Versions** tab, and click **Import New Versions**.

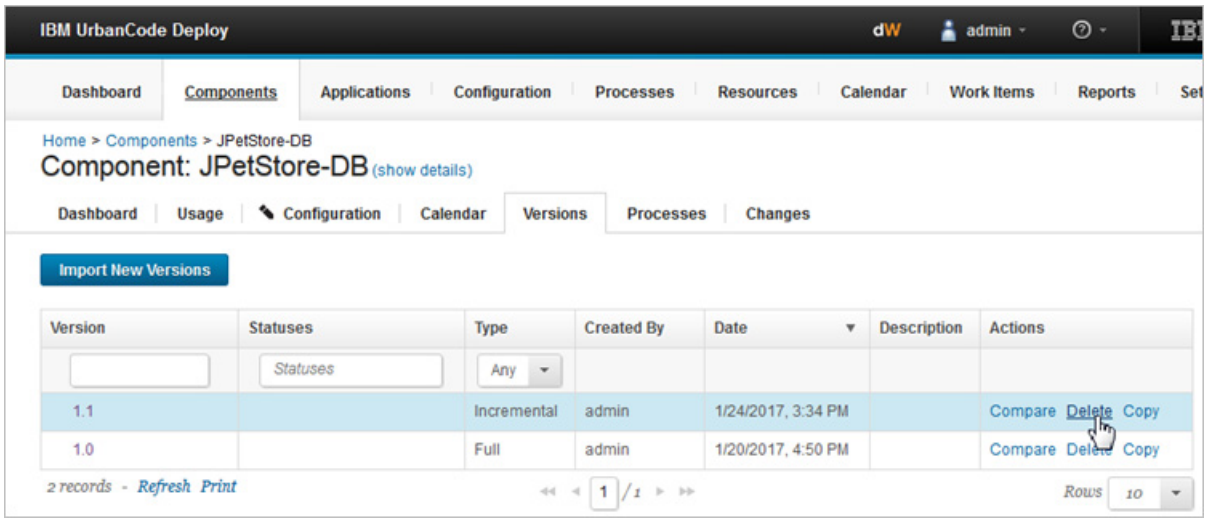
The server shows two versions of the web component.

Exercise 4 Deleting newest versions of components

In later exercises, you will update the server with new component versions. For now, delete the latest versions of the database and web components, so that you can simulate importing the new versions later.

Delete the latest versions:

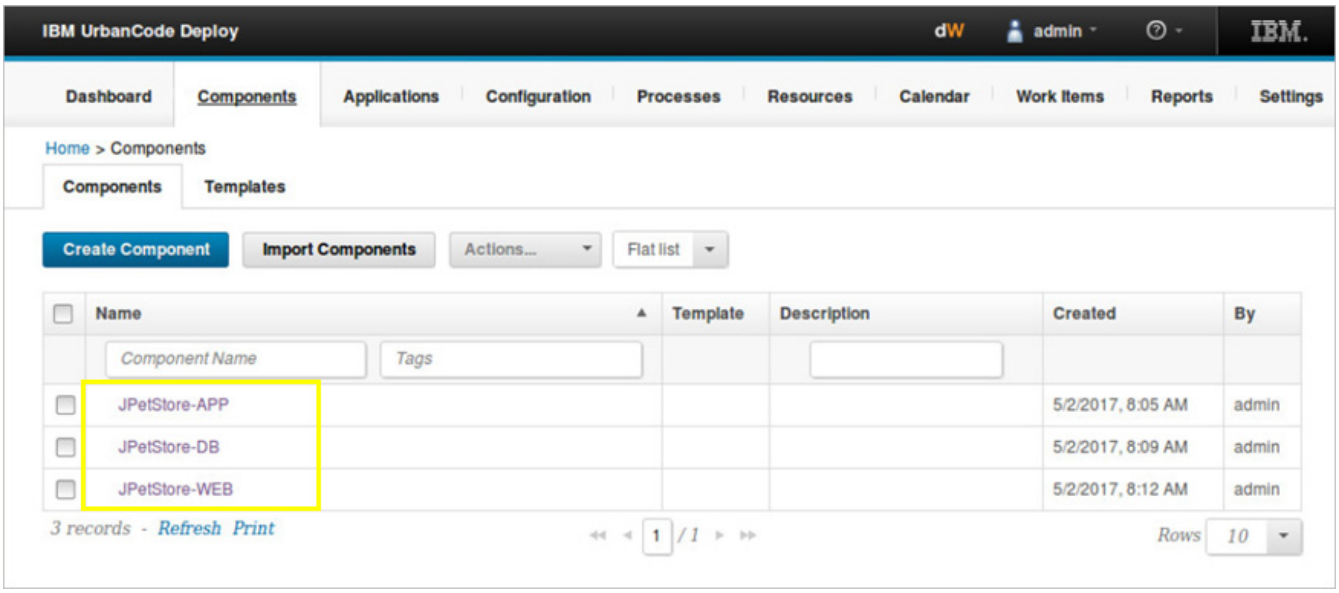
1. Click **Components**, and click the **JPetStore-DB** component.
2. Click **Versions**.
3. In the list of versions, in the same row as the version that is named 1.1, click **Delete**, and click **OK**. You import this version again later.



4. Similarly, delete version 1.1 of the `JPetStore-WEB` component.

In a production scenario, you could connect the component to a source-code management system such as Subversion or a build system, such as Jenkins.

On the **Components** tab, you have three components, each with one version, as shown in the following figure:



Creating component processes

In addition to files and other artifacts, components also contain component processes. Processes are a list of steps and connections between those steps. Each step is an individual command that runs on a target computer. Steps can manipulate files, run system commands, download files, and run programs.

Each component must have at least one component process to deploy or install the component. In this section, you create a deployment process for each of the three components. Later, you create an application process that calls these component processes to deploy each component.

Exercise 5 Creating a component process for the web component

The component process for the web component downloads the most recent version of the component artifacts and puts those artifacts in the correct folder. Note that when you assign properties for the processes you must be sure the syntax is exactly correct. Any error involving stray spaces can cause a process to fail.

Create the component process that deploys the web component:

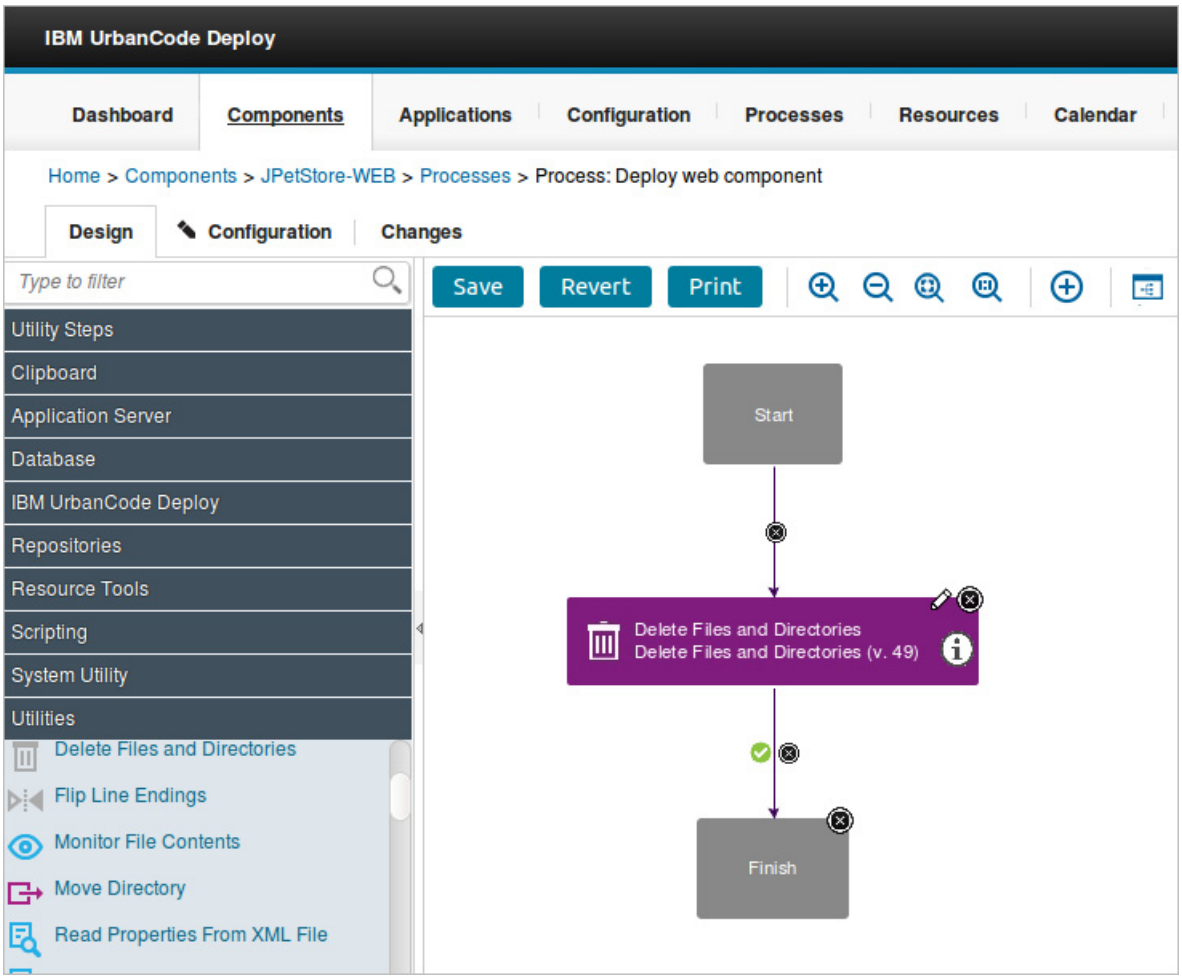
1. From the **Components** tab, click the `JPetStore-WEB` component.
2. From the Components area, click **Processes**, and click **Create Process**.
   c. In the Create Process window, specify the name as `Deploy web component`.
   d. In the **Process Type** list, select **Deployment**. This list has other options for processes that uninstall or configure components.
   e. In the **Default Working Directory** field, type `${p:resource/work.dir}/${p:resource/ResourceWorkDirName}/${p:component.name}`.
   f. Accept the other default values for the other properties, and click **Save**.

Note: The Default Working Directory field becomes the folder that the agent uses to do its work, such as downloading artifacts and creating temporary files.

3. From the list of processes, click the new process.
   The process opens in the process editor. The editor shows the steps in the process in a graphical form as a flowchart. The Start and Finish boxes represent the beginning and the end of the process.

From here, you add steps to the process and link them between the Start and Finish steps to show the order of the steps.

4. Add a step to clean out files in the working directory. The process runs within a working directory. To ensure that you are using the most recent version of the files in the web component, run a command to clean the working directory:
   a. In the **Design** tab at the left of the process editor, expand the **Utilities** drawer. The **Step Palette** menu shows the available steps. You can explore the drawers to find steps, or you can type in the search box at the top of the window.
   b. Under the **FileUtils** section, drag the **Delete Files and Directories** step to the process editor, between the Start and Finish steps.
   c. To open the **Edit Properties** window, click the **Information** icon at the right of the step box. This icon is displayed when a step requires information. You can also click the **Pencil** icon to open the window.



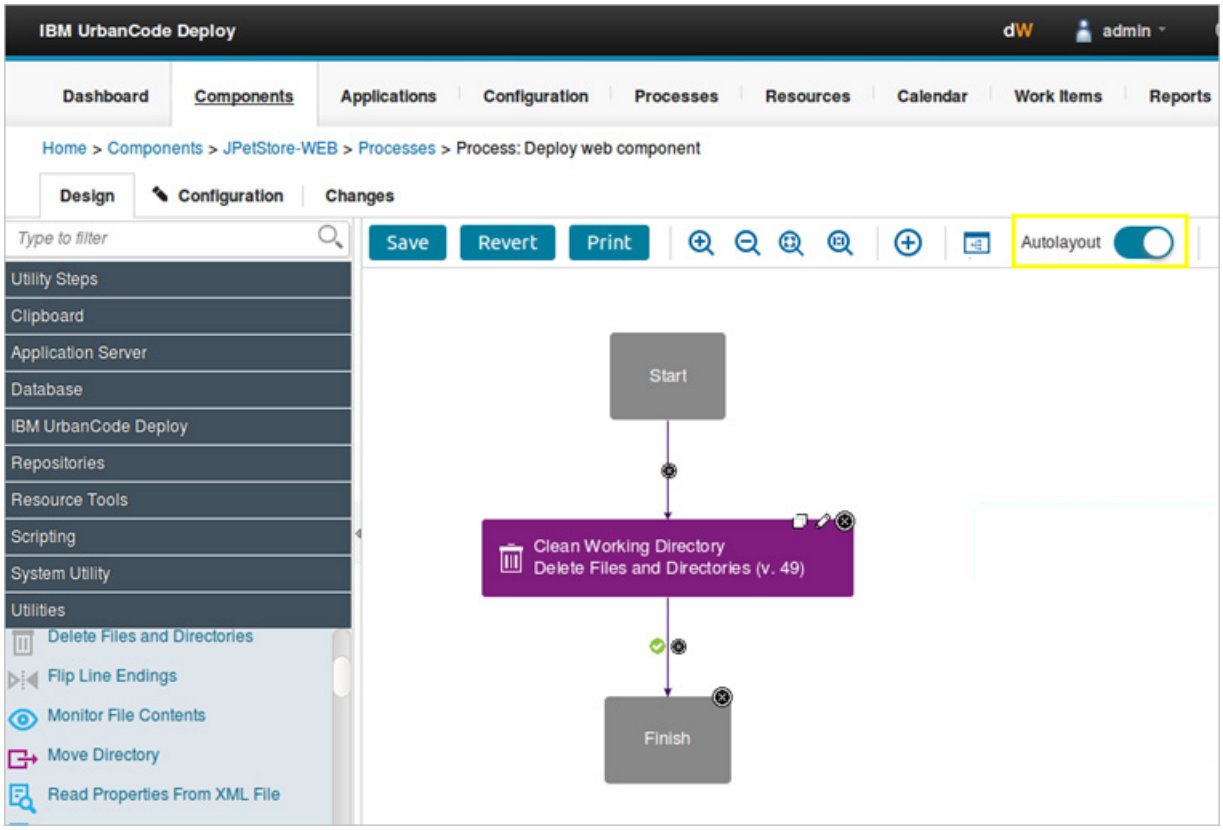
   The Edit Properties window opens, showing the properties for the step. Some of these properties are unique to the step and others are the same for all steps.
   d. In the **Name** field, type the name as `Clean Working Directory`.

e. In the **Base Directory** field, type a single period (.).

f. In the **Include** field, type `**/*`, and click **Save**.

g. Accept the defaults for the other properties, and click **OK**.

The new step is displayed as a box in the process editor. Later, you connect this step to the other steps in the process. The process editor looks like the following figure:



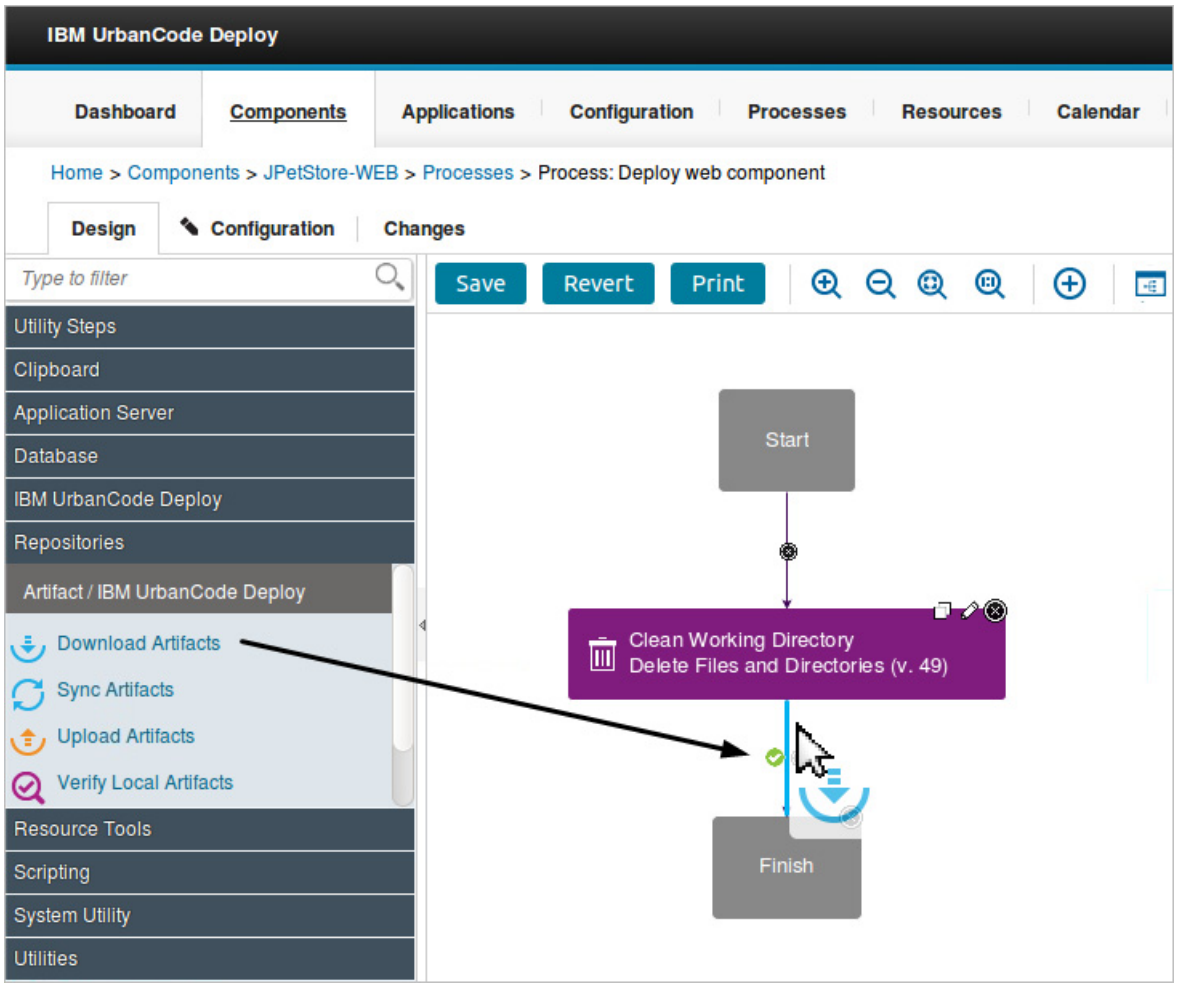
Note: Turn on the auto layout feature to automatically arrange the diagram.

5. Add a step to download the latest version of the component artifacts. The `Download Artifacts` step is used in most deployment processes. This step downloads the specified version of the

component artifacts to the target computer. When you run the process, you specify whether to use the most recent version of the component artifacts or a specific version.

a. In the list of steps, expand **Repositories > Artifact/IBM UrbanCode Deploy**, and drag the **Download Artifacts** step to the process editor, after the Clean Working Directory step.

   A blue line is displayed, indicating where you can drop the step to have the arrows automatically connect.



b. Open the Edit Properties window, accept the default values, and click **OK**.

6. Add a step to remove the old web content from the server:

   a. Expand the **Utilities** drawer.

   b. In the FileUtils section, drag the **Delete Files and Directories** step to the process editor after the Download Artifacts step.

   c. In the Edit Properties window, type the name as `Remove Old Content`.

   d. In the **Base Directory** field, type `webapps/JPetStore`.

   e. In the **Include** field, type `images`, and click **Save**.

This variable represents the main folder of the Tomcat web server. You specify a value for this variable later.

f. In the **Working Directory** field, type `${p:environment/tomcat.home}`.

g. Accept the default values in the other fields, and click **OK**.

7. Add a step to deploy the new content to the server. This step copies the component artifacts to the application server.

   a. From the **Utilities > FileUtils** drawer, drag the **Move Directory** step to the process editor under the Remove Old Content step.

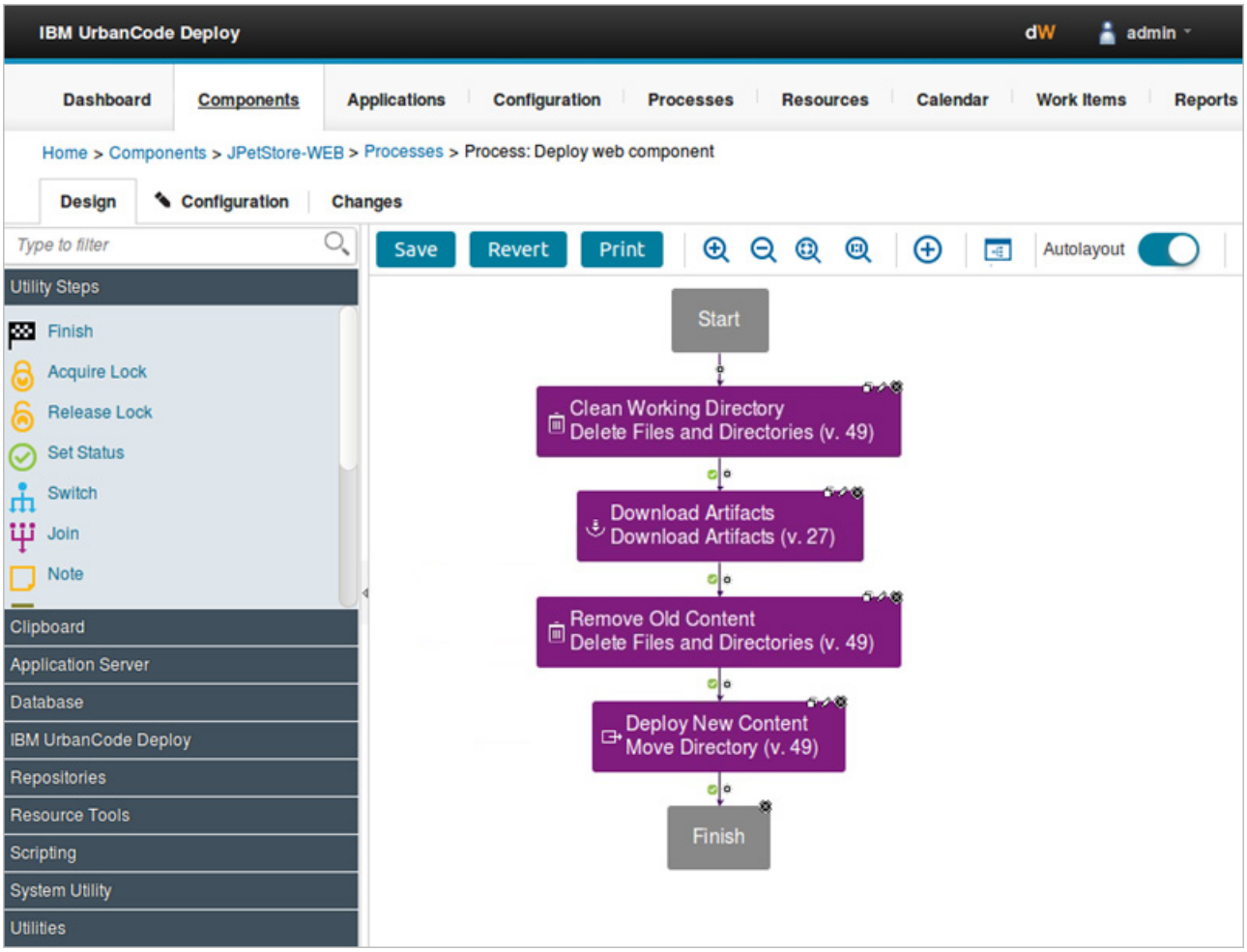
   b. Open the Edit Properties window, and type the name as `Deploy New Content`.

   c. In the **Source Directory** field, type a single period (`.`).

   d. In the **Destination Directory** field, type `${p:environment/tomcat.home}/webapps/JPetStore/`.

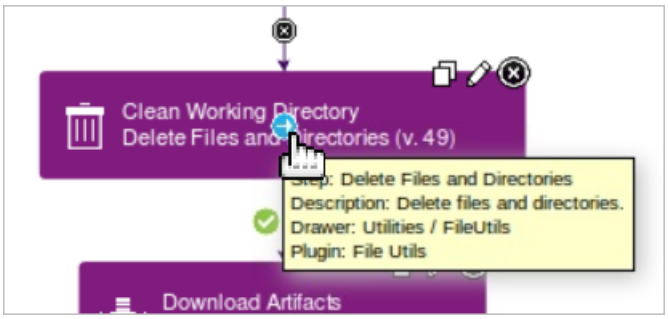
   e. Accept the default values in the other fields, and click **OK**.

   The completed process has four steps, as in the following outline:



Each step has three icons at the upper-right corner of the step. The **Edit** icon opens the Edit Properties window. The **X** icon deletes the step. The duplicate icon creates a copy and puts it into a directory called `Clipboard` in the Step Palette.

If you move the mouse pointer over a step, an arrow icon is displayed over the center of the step. You can create links between steps with this icon.



Note: An information box shows details about the step.

8. If all steps are not connected, connect the steps in the order that they run:
   a. `Start`
   b. `Clean Working Directory`
   c. `Download Artifacts`
   d. `Remove Old Content`
   e. `Deploy New Content`
   f. `Finish`
9. Click **Save** to save the process.

You can connect steps in a sequence like these steps, or you can run steps in parallel with each other by creating more links. However, processes always start with the Start step and end with the End step.

Exercise 6 Creating a component process for the application component

The application component consists of a single WAR file. This component process downloads the most recent version of the WAR file, updates values in the file, and deploys the file on the application server.

Create the component process that deploys the application component:

1. From the **Components** tab, click the `JPetStore-APP` component.
2. On the **Components** tab, click **Processes**, and click **Create Process**.
   a. In the Create Process window, specify the name as `Deploy application component`.
   b. In the **Process Type** list, select **Deployment**.
   c. In the **Default Working Directory** field, type `${p:resource/work.dir}/${p:resource/ResourceWorkDirName}/${p:component.name}`.
   d. Accept the other default values for the other properties, and click **Save**.
3. From the list of processes, click the new process to open it in the process editor.
4. As you did for the web component, add a step to clean the working directory:
   a. In the **Utilities > FileUtils** drawer, drag the **Delete Files and Directories** step to the process editor, between the Start and Finish steps.
   b. Open the Edit Properties window.
   c. In the **Name** field, type `Clean Working Directory`.
   d. In the **Base Directory** field, type a single period (.).
   e. In the **Include** field, type `**/*`, and click **Save**.
   f. Click **OK**.
5. As you did for the web component, add a step to download the artifacts:
   a. Expand the **Repositories > Artifact/IBM UrbanCode Deploy** drawer.
   b. Drag the `Download Artifacts` step to the process editor.
   c. Accept the default properties, and click **OK**.
6. Add a step to extract the WAR file. You must extract the WAR file so that you can update a property file with the database connection information.
   a. Open the **Utilities > FileUtils** drawer, and drag an **Unzip** step to the process editor.
   b. In the Edit Properties window, type the name as `Expand WAR`.

c. In the **Extract directory** field, type the following folder:
`./JPetStore_expanded`

d. In the **.zip files** field, type the following file:
`JPetStore.war`

e. Accept the defaults for the other fields, and click **OK**.

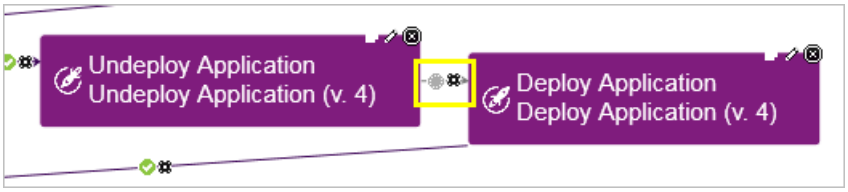
7. Add a step to update the property file with the location of the database:
   a. Expand the **Utilities > FileUtils** drawer, and add an **Update Java Properties File** step to the process editor.
   b. Accept the default name.
   c. In the **Directory Offset** field, type the following folder:
   `./JPetStore_expanded/WEB-INF/classes/properties`
   d. Click the **File Includes** field, type the following file, and click **Save**:
   `database.properties`
   e. Click the **Add/Update properties** field, specify the following code, and click **Save**:
   `url=${p:environment/db.url}`
   This code updates the URL property in the property file with the location of the database. The code `${p:environment/db.url}` is a variable that represents the location of the database component when the application is deployed.
   f. Click **OK**.
8. Add a step to update the WAR file with the new property file.
   a. Expand the **Utilities > FileUtils** drawer, and add a **Create .zip File** step to the process editor.
   b. Specify the name as `Update WAR`.
   c. In the **.zip File Name** field, type the following file:
   the `JPetStore.war`
   d. In the **Base Directory** field, type the following folder:
   `./JPetStore_expanded`
   e. Click the **Include** field, type the following code, and click **Save**:
   `**/*`
   f. Select the **Update Existing** check box. This option updates or overwrites the destination file if a file with that name already exists.
   g. Click **OK**.

9. Add a step to start Tomcat:
   a. Expand the **Application Server > Java/Tomcat** drawer, and add a **Start Tomcat** step to the process editor.
   b. Accept the default name: `Start Tomcat`.
   c. In the **Launcher** field, specify the following code:
      `${p:environment/tomcat.start}`
      This code is another variable that represents the command to start Tomcat.
   d. In the **Startup timeout (in seconds)** field, type `60`.
   e. In the **Port** field, type the port:
      `${p:environment/tomcat.port}`
   f. Click **OK**.
10. Add a step to remove any previous versions of the application:
   a. Expand the **Application Server > Java/Tomcat** drawer, and add an **Undeploy Application** step to the process editor.
   b. Accept the default name: `Undeploy Application`.
   c. In the **Tomcat Manager URL** field, type the following code:
      `${p:environment/tomcat.url}/manager`
   d. In the **Tomcat Manager Username** field, type `tomcatmanager` for the Tomcat server.
   e. In the **Tomcat Manager Password** field, type `tomcatmanager`.
   f. In the **Context Name** field, type the following context name, including the initial forward slash:
      `/JPetStore`
   g. Click **OK**.
11. Add a step to deploy the updated WAR file to the application server:
   a. Expand the **Application Server > Java/Tomcat** drawer, and add a **Deploy Application** step to the process editor.
   b. Accept the default name `Deploy Application`.
   c. In the Tomcat Manager URL field, type the following code:
      `${p:environment/tomcat.url}/manager`
   d. In the **Tomcat Manager Username** field, type `tomcatmanager`.
   e. In the **Tomcat Manager Password** field, type `tomcatmanager`.
   f. In the **Context Name** field, type the following context name:
      `/JPetStore`

g. In the **War File Path** field, type the following path:
`./JPetStore.war`

h. Click **OK**.

12. Ensure the steps are in the following order:
    a. `Start`
    b. `Clean Working Directory`
    c. `Download Artifacts`
    d. `Expand WAR`
    e. `Update Java Property File`
    f. `Update WAR`
    g. `Start Tomcat`
    h. `Undeploy Application`
    i. `Deploy Application`
    j. `Finish`
13. Change the conditional flag on the link between the `Undeploy Application` and `Deploy Application` steps to `both` by clicking the *green* check mark on the connection until it turns to a gray circle, as shown in the following figure:



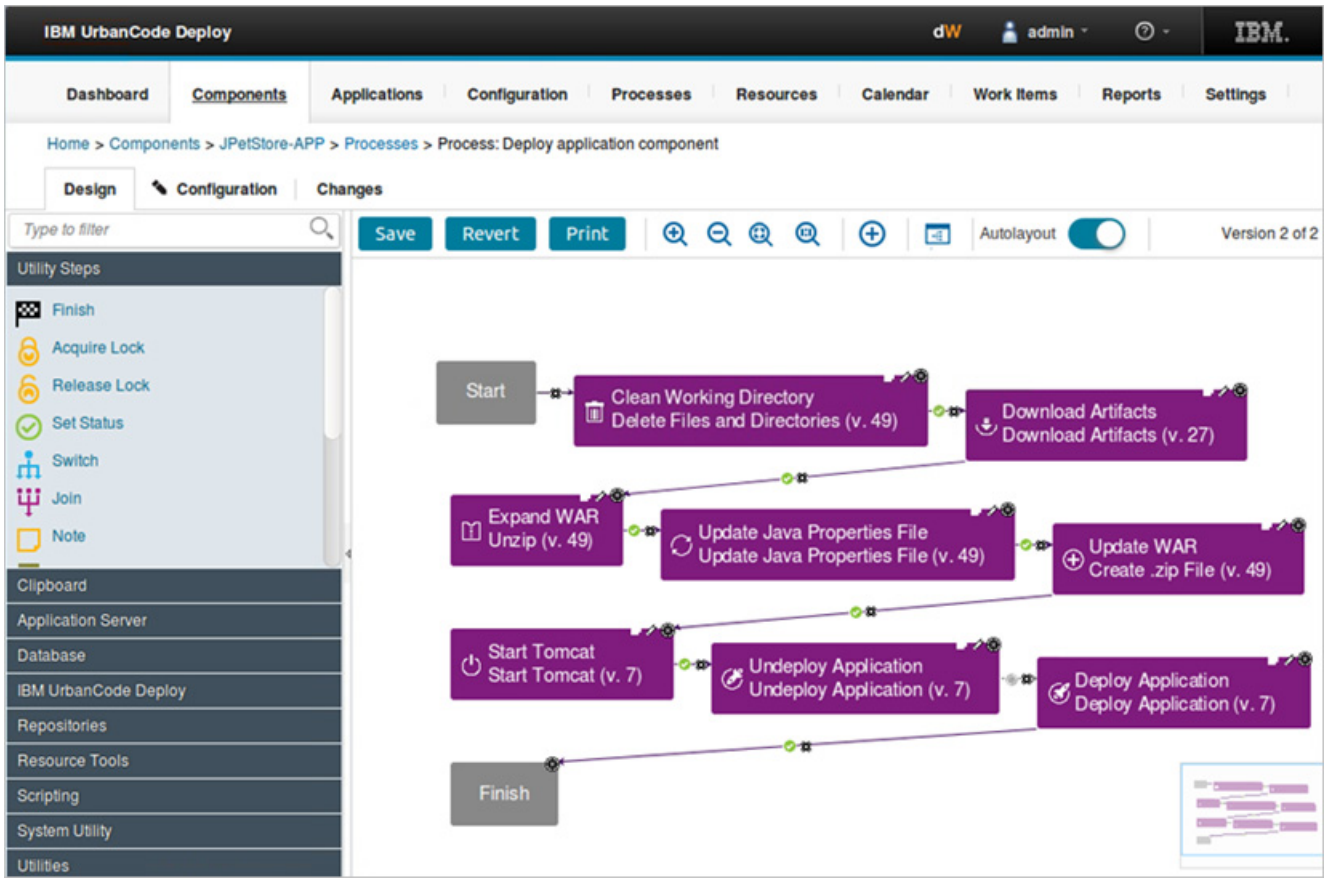
Each connection from one step to another has a *conditional flag*. The conditional flag specifies a condition for the connection. Three conditional flags are available:

- By default, the conditional flag on each connection is set to `success`, which is represented by a green check mark. The process follows these connections when the step is complete.
- The conditional flag `fail` is represented by a red dash (-). The process follows these connections when the step fails.
- The conditional flag `both` is represented by a gray circle. The process follows these connections regardless of whether the step succeeds or fails.

The `Undeploy Application` step fails if no previous versions of the application are installed. In this case, setting the conditional flag on the connection to `both` specifies that the process keeps running regardless of whether the step fails or is completed.

14. Save the process.

The complete component process for the application component looks like the following figure:



Exercise 7 Creating a component process for the database component

The deployment process for the database component updates the database with sample values.

Create the component process that deploys the database component:

1. From the **Components** tab, click the `JPetStore-DB` component.
2. Click **Processes**, and click **Create Process**.
   a. In the Create Process window, specify the name as `Deploy DB component`.
   b. In the **Process Type** list, select **Deployment**.
   c. In the **Default Working Directory** field, type `${p:resource/work.dir}/${p:resource/ResourceWorkDirName}/${p:component.name}`.
   d. Accept the other default values for the other properties, and click **Save**.
3. From the list of processes, click the new process to open it in the process editor.
4. Add a step to clean the working directory:
   a. In the **Utilities > FileUtils** drawer, drag the **Delete Files and Directories** step to the process editor, between the Start and Finish steps.
   b. Open the Edit Properties window.
   c. In the **Name** field, type `Clean Working Directory`.
   d. In the **Base Directory** field, type a single period (.).
   e. In the **Include** field, type `**/*`, and click **Save**.
   f. Click **OK**.
5. Add a step to download the artifacts:
   a. Expand the **Repositories > Artifact/IBM UrbanCode Deploy** drawer.
   b. Drag the `Download Artifacts` step to the process editor.
   c. Accept the default properties, and click **OK**.
6. Add a step to update the database:
   a. In the list of steps, expand the **Database > DBUpgrader** drawer, and drag an **Upgrade DB** step to the process editor.
   b. In the Edit Properties window, in the **Name** field, accept the default value of `Upgrade DB`.

c. In the **Driver Classname** field, type the following class:
`com.mysql.jdbc.Driver`

d. In the **DB Driver Jar** field, type the following path:
`lib/mysql-connector-java-5.1.20-bin.jar`
This file is provided in the source code for the component.

e. In the **URL** field, type the following variable:
`${p:environment/db.url}`

f. In the **User** field, type the user ID of the database user. It is `jpetstore`.

g. In the **Password** field, type the password as the `jppwd` of the database user.

h. In the **SQL File** path field, type a single period: `(.)`

i. In the **SQL File Include** field, type this extension:
`*.xml`

j. Click the **Current Version SQL** field, type the following SQL code, and click **Save**:
`SELECT VER FROM DB_VERSION WHERE RELEASE_NAME = ?`

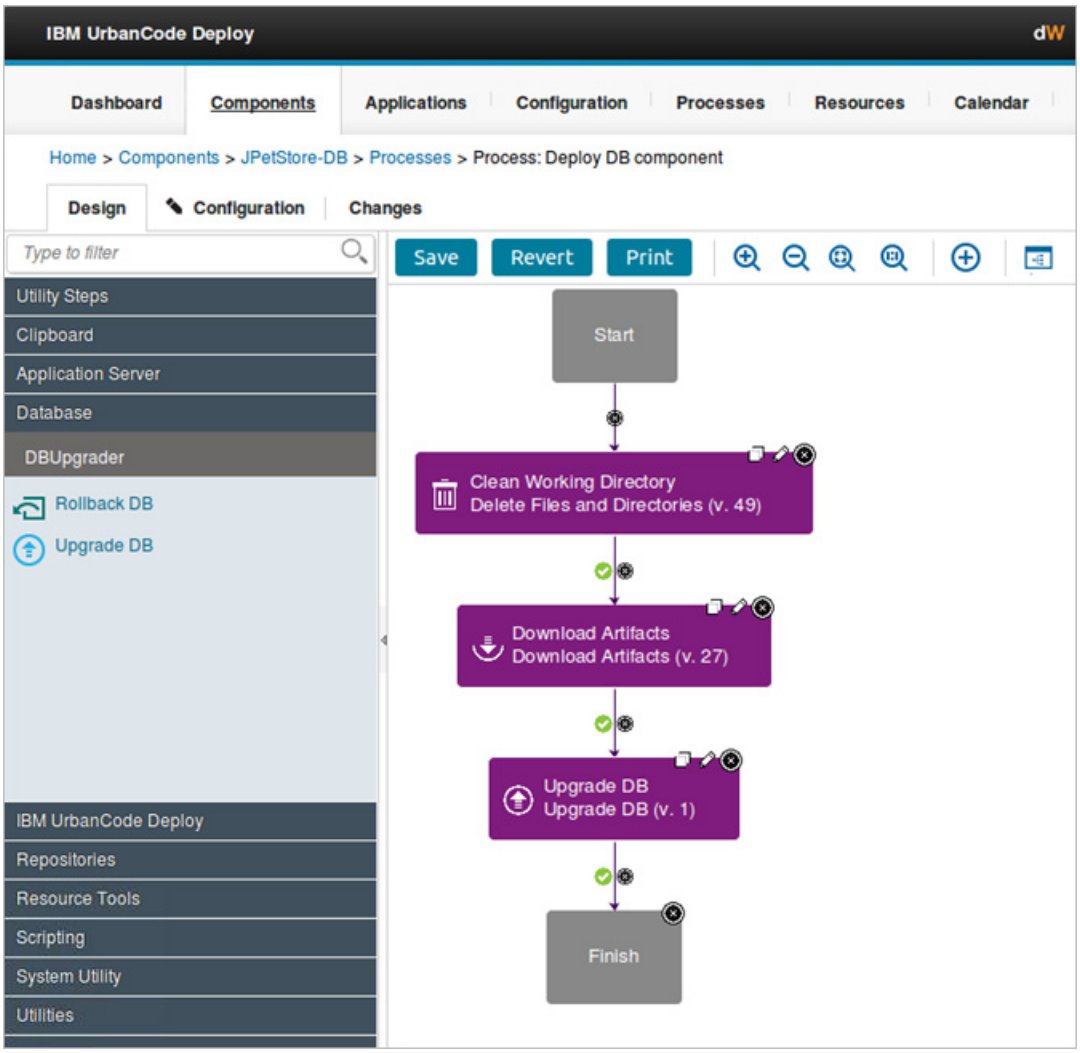
k. Click the **Delete Version SQL** field, type the following SQL code, and click **Save**:
`DELETE FROM DB_VERSION WHERE RELEASE_NAME = ?`

l. Click the **Update Version SQL** field, type the following SQL code, and click **Save**:
`INSERT INTO DB_VERSION (RELEASE_NAME,VER) VALUES(?,?)`

m. Click **OK**.

7. Ensure the steps are connected in the following order:
   a. `Start`
   b. `Clean Working Directory`
   c. `Download Artifacts`
   d. `Upgrade DB`
   e. `Finish`
8. Save the process.

The complete component process for the database looks like the following figure:



Lab checkpoint

In this lab, you created component processes to deploy components. You can use component processes to automate many tasks on components. However, you typically do not run component processes directly. In most cases, you add them to application processes. In the next unit, you create an application process for these three component processes.

Processes consist of steps, which are provided by plug-ins. To see a list of plug-ins that are available and for documentation on those plug-ins and steps, see IBM developerWorks: UrbanCode Deploy Plug-ins.

Unit 3 Defining deployments with resources, environments, and application processes exercises

Many application development teams use multiple environments to form a deployment pipeline. For example, a development team might work on an application in a small environment. When the application is ready, the development team can promote the application to a second environment, such as a quality assurance environment, where a testing team can test the application in a larger, more realistic environment.

In these exercises, you create an application and add the components. Then you configure the resources for two environments, so that you can simulate developing and promoting an application in a later exercise.

After completing this lab, you will be able to complete these tasks:

- Create an application and associate components
- Create environments and associate base resources
- Create an application process to deploy components

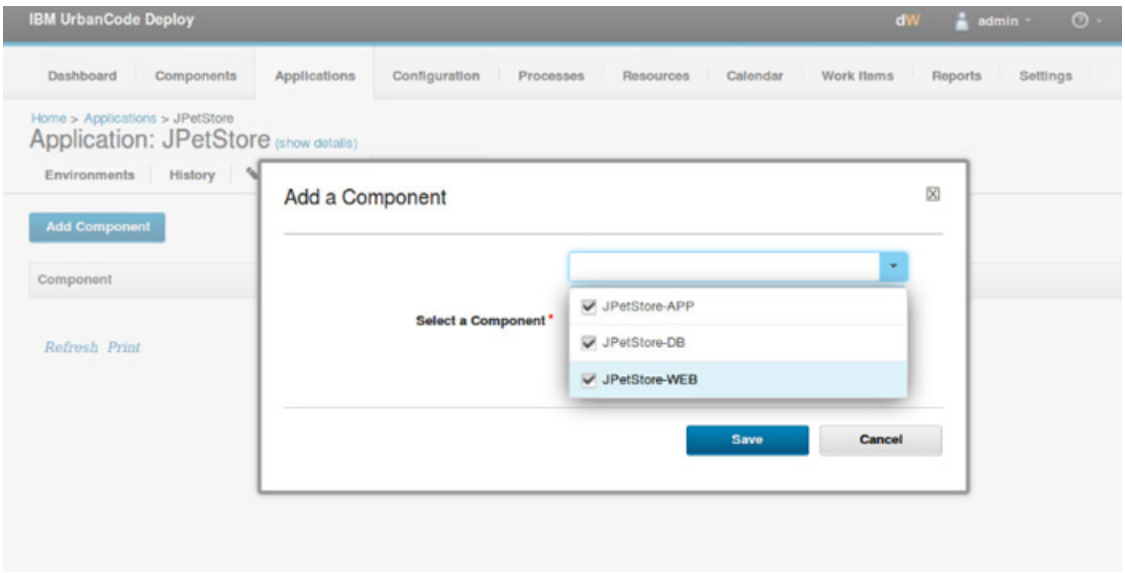
You use the JPetStore APP, DB, and WEB components and processes.

Exercise 1 Creating an application

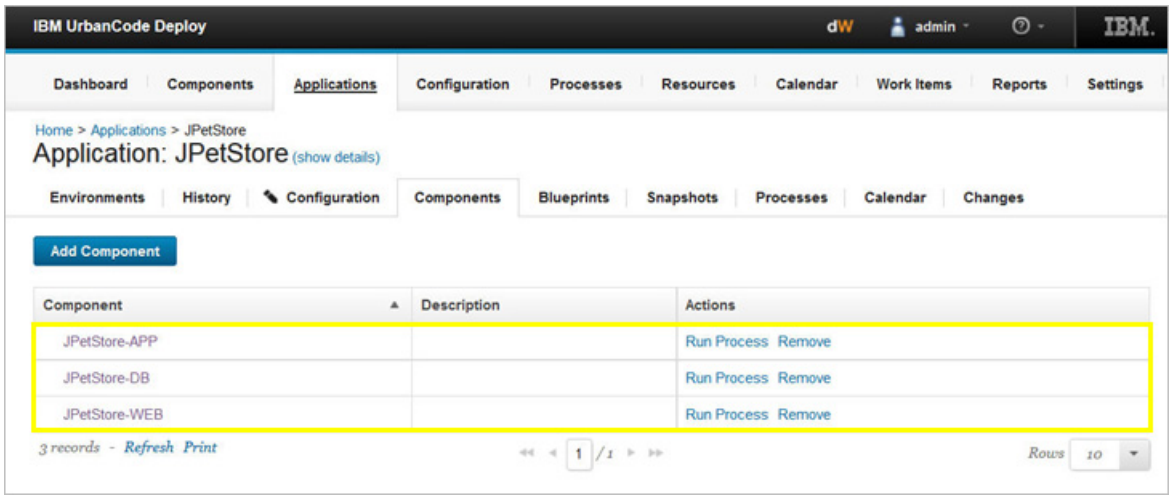
Creating an application involves selecting the components that are in the application. In this exercise, you create an application and add the three components.

Create an application and add components:

1. Click the **Applications** tab. Click **Create Application**.
2. Name the new application `JPetStore`, and click **Save**.
3. Add the components to the application:
   a. Click the **Components** tab for the application, and click **Add Component**.
   b. In the Add a Component window, from the **Select a Component** list, select the three components:
      - `JPetStore-APP`
      - `JPetStore-WEB`
      - `JPetStore-DB`.
   c. Click **Save**.



The application includes the three components, as shown in the following figure. Components can be included in any number of applications.



Exercise 2 Verifying that the agent is running

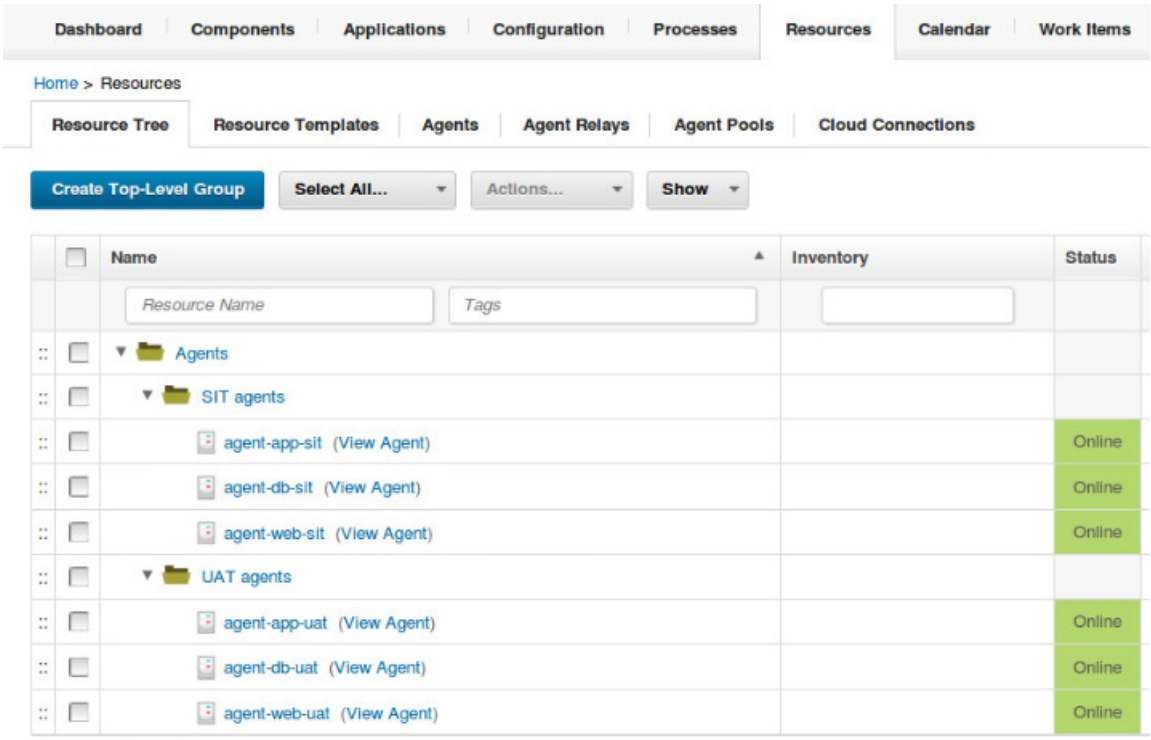
IBM UrbanCode Deploy uses agents and servers. Agents run scripts to complete various actions. In this exercise, you verify that agents are installed and running. Two Tomcat servers are also installed for the lab exercise: one for the Software Integration Test (SIT) environment and one for the User Acceptance Test (UAT) environment. Typically, there is one agent for each environment that you deploy applications to.

A resource is a logical deployment target. You manage resources in a physical or conceptual hierarchy called a *resource tree*. From there, they can represent an agent, a group of agents, a component, or an organizational entity that is used to group other resources.The resource group that you create in this task is a container for the resources for the lab environment.

View the agents:

1. Click the **Resource Tree** tab, and expand the **Agents** resource group.

   Two agent groups are listed for each environment, one for the SIT agents, and one for the UAT agents. Each agent group has three agents: one for the application, one for the database, and one for the web. In the **Status** column, the agents should be listed as `Online`.

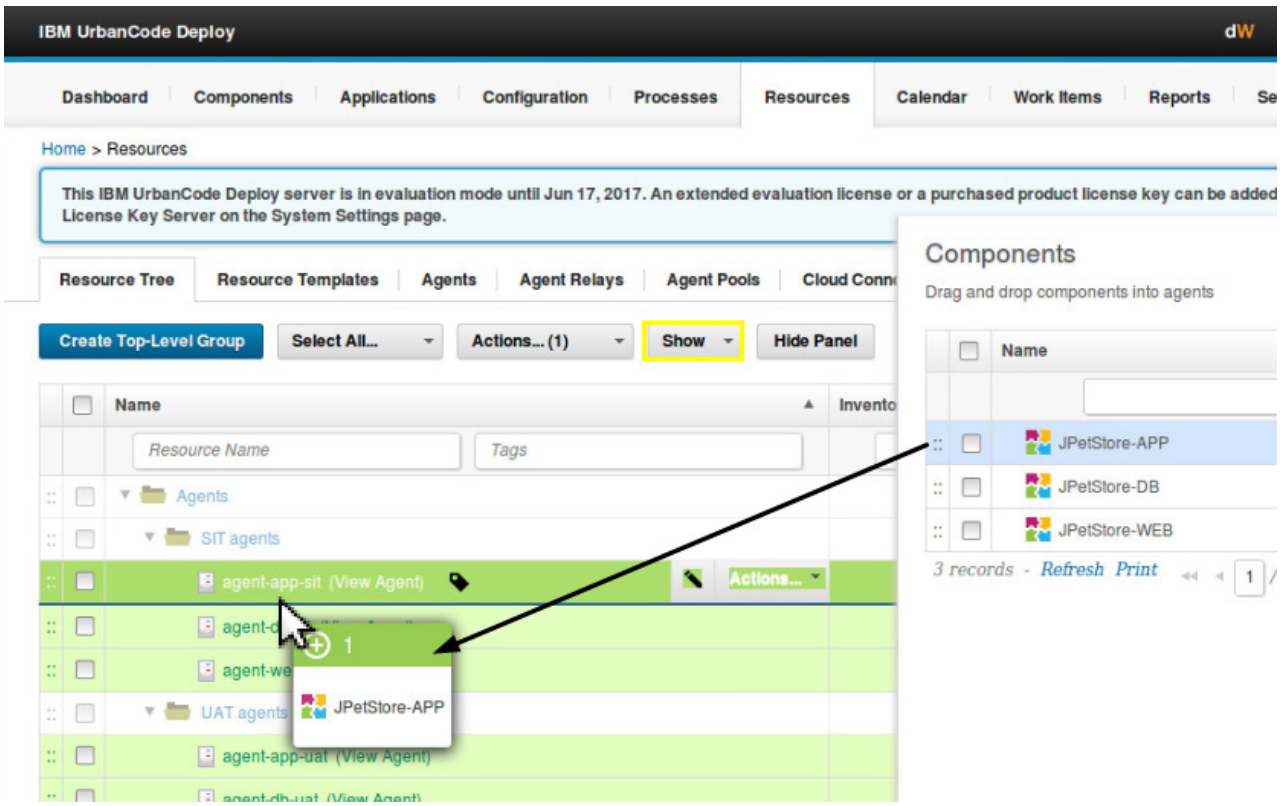


   Resources on the IBM UrbanCode Deploy server represent the system resources that are available to deploy applications to, such as physical hardware or virtual machines. In this exercise, you create a resource for the JPetStore_SIT target environment and associate the resource to a physical agent.

Exercise 3 Adding components to the agent resources

When you map components to an agent resource, you identify all of the components that are being deployed to that environment.

1. Add components to the SIT agent resources:
   a. Click the **Show** button, and then choose **Components**.
   b. In the SIT agents group, on the same row as agent-app-sit, drag the **JPetStore-APP** component from the Components panel.

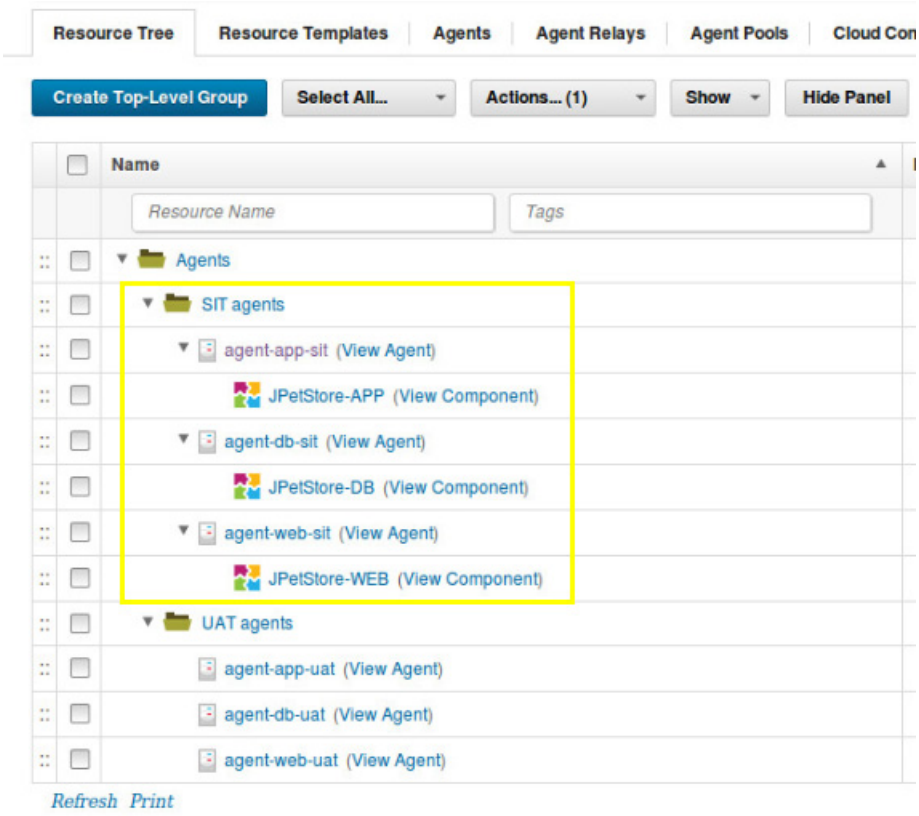


Note: You can also add components by clicking the **Actions** menu and clicking **Add Component**.

   c. Repeat the process to add the component, **JPetStore-WEB**, to agent-web-sit and the component, **JPetStore-DB**, to agent-db-sit.

Attention: Be sure to add each component to the agent resource and not to another component. The components must be at the same hierarchy level, and not as child items of each other.

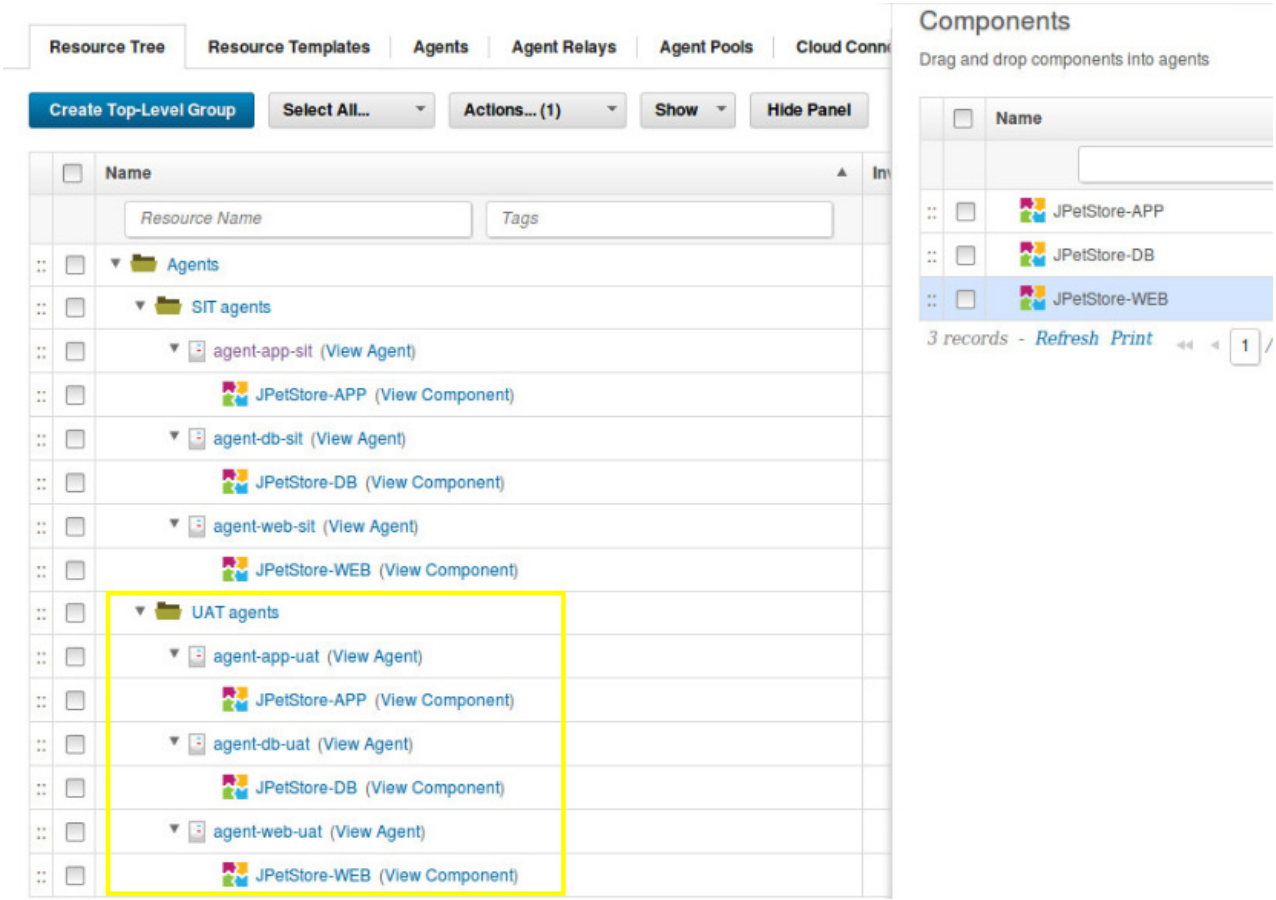
The components are mapped to the target system, as shown in the following figure:



Next, you add components to the JPetStore-UAT agent resources. This environment has three agents, so you can specify which components run on each agent resource.

2. Repeat the process of adding components to agent resources for the UAT resource group:
   a. On the same row as the **agent-app-uat** resource, drag the **JPetStore-APP** component from the Components panel.
   b. On the same row as the **agent-db-uat** resource, drag the **JPetStore-DB** component form the Components panel.
   c. On the same row as the **agent-web-uat** resource, drag the **JPetStore-WEB** component form the Components panel.

The components are mapped to the target system, as shown in the following figure:



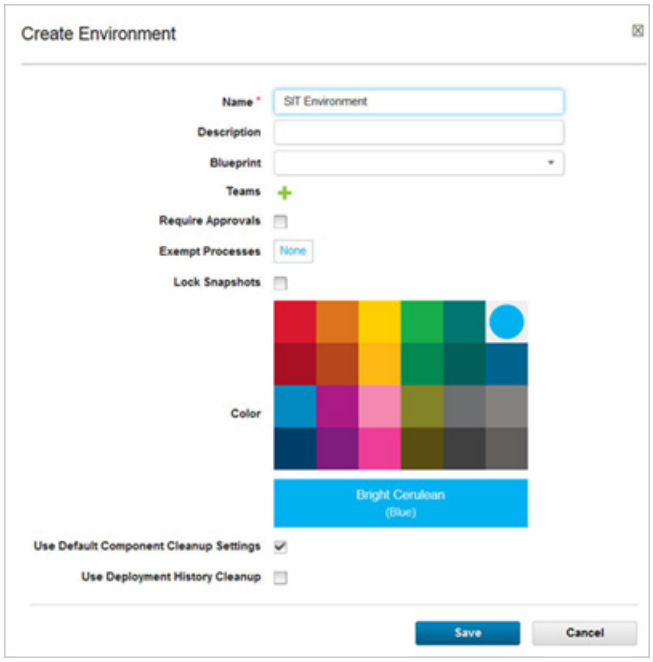
After the environments are prepared and the agents are running, you can run application processes on the environments to deploy the components.

Exercise 4 Creating the SIT environment

Environments represent systems on which you deploy the application components. There are two environments for this class: SIT and UAT.

Create the SIT environment:

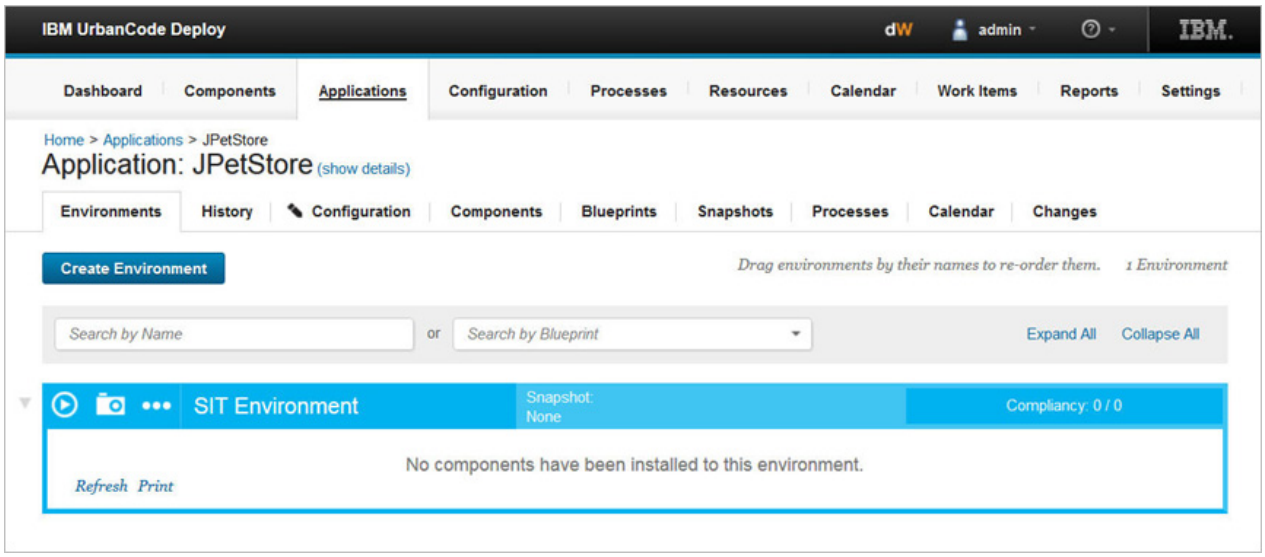
1. Open the **Environments** tab for the JPetStore application.
2. Click **Create Environment**.
3. Specify the name as `SIT Environment`.
4. Accept the default value in the other fields in this window, and click **Save**.



Specify the properties for the environment:

When you created the component processes, you included parameters that specify the location of artifacts on the target system, such as the location of the application server. Because these properties can vary on different target systems, specify these properties as environment properties.

5. Open the environment by clicking the environment name.

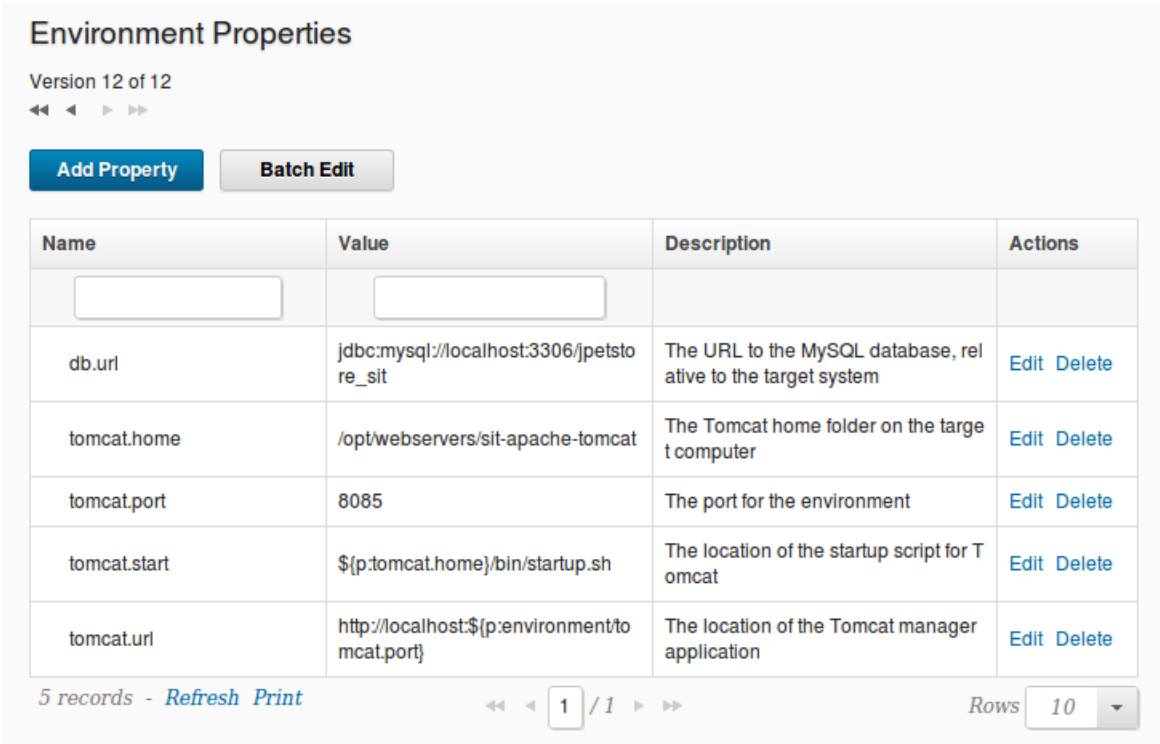


6. Open the environment properties by clicking the **Configuration** tab and clicking **Environment Properties**.

7. Add the following properties, one at a time, by clicking **Add Property** and typing the name and value of the property as shown in the following table:

Table 1 Properties for the SIT environment

Name	Description (optional)	Value
tomcat.home	The Tomcat home folder on the target computer	/opt/webservers/sit-apache-tomcat
db.url	The URL to the MySQL database, relative to the target system	jdbc:mysql://localhost:3306/jpetstore_sit
tomcat.url	The location of the Tomcat manager application	http://localhost:${p:environment/tomcat.port}
tomcat.start	The location of the startup script for Tomcat	${p:tomcat.home}/bin/startup.sh
tomcat.port	The port for the environment	8085



Environment Properties

Version 12 of 12

Add Property | Batch Edit

Name	Value	Description	Actions
db.url	jdbc:mysql://localhost:3306/jpetstore_sit	The URL to the MySQL database, relative to the target system	Edit Delete
tomcat.home	/opt/webservers/sit-apache-tomcat	The Tomcat home folder on the target computer	Edit Delete
tomcat.port	8085	The port for the environment	Edit Delete
tomcat.start	${p:tomcat.home}/bin/startup.sh	The location of the startup script for Tomcat	Edit Delete
tomcat.url	http://localhost:${p:environment/tomcat.port}	The location of the Tomcat manager application	Edit Delete

5 records - Refresh Print

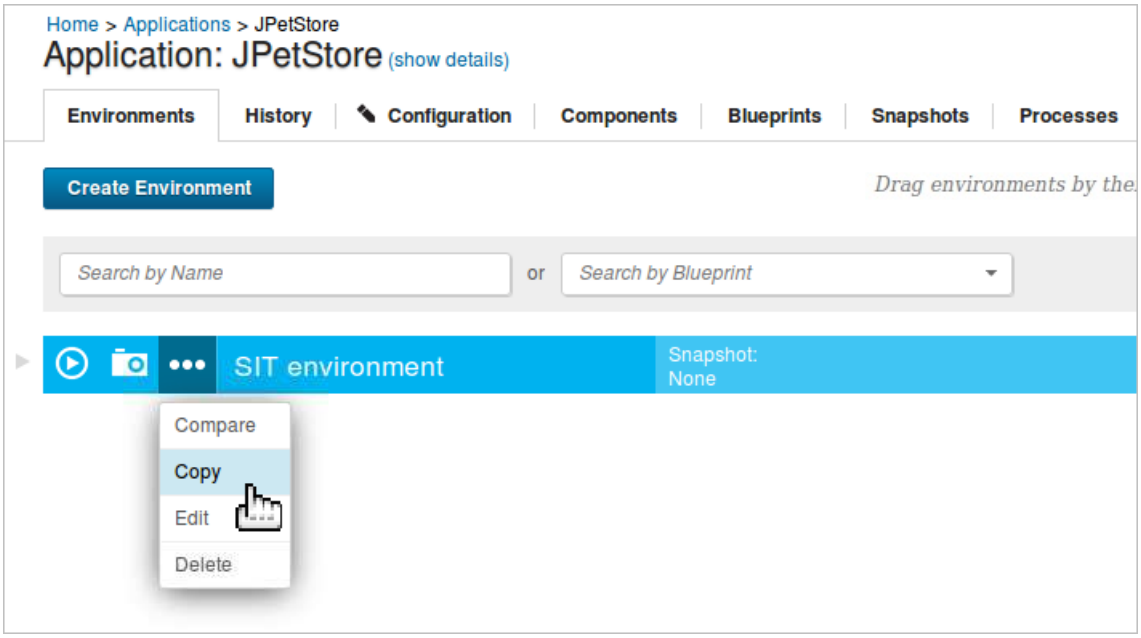
1 / 1

Rows 10

Exercise 5 Creating the UAT environment

Create the UAT environment:

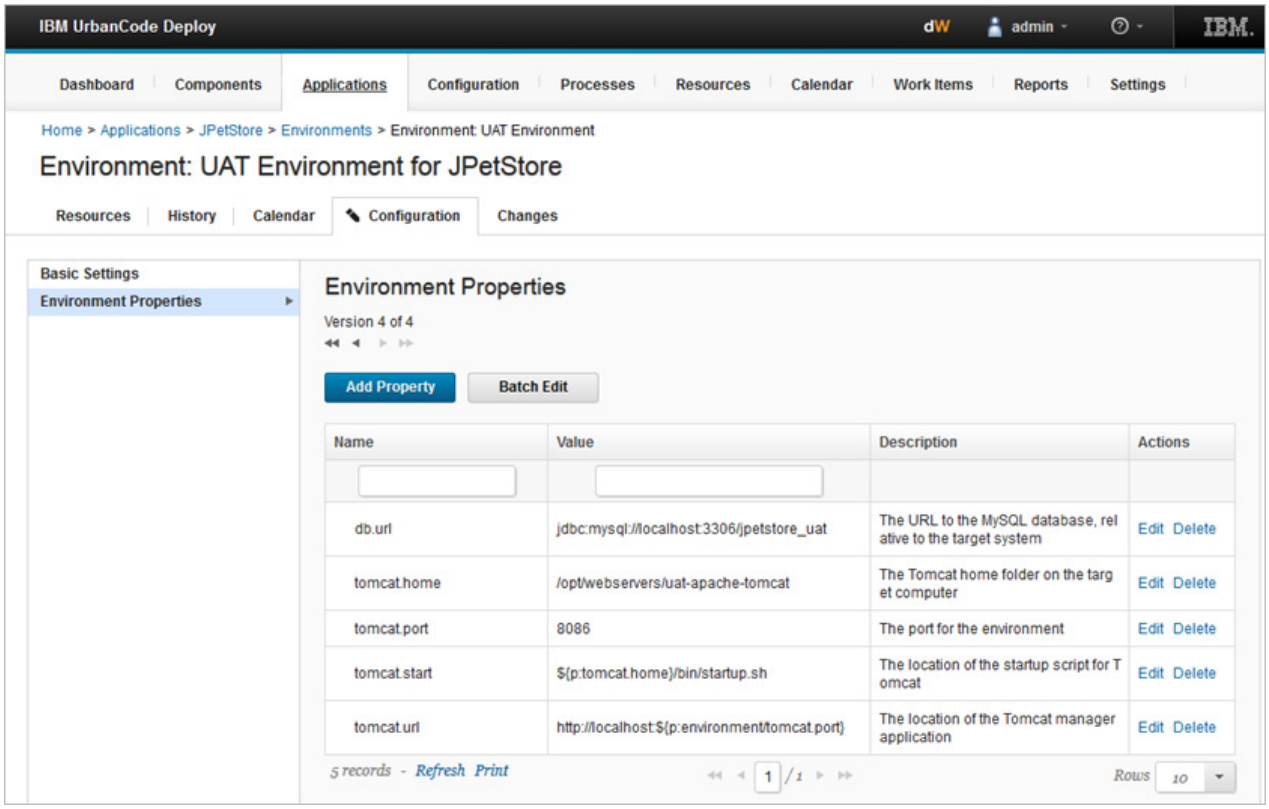
1. Create an environment named `UAT Environment` by copying the existing SIT environment. Click the **More** icon next to the SIT Environment name, and click **Copy**:



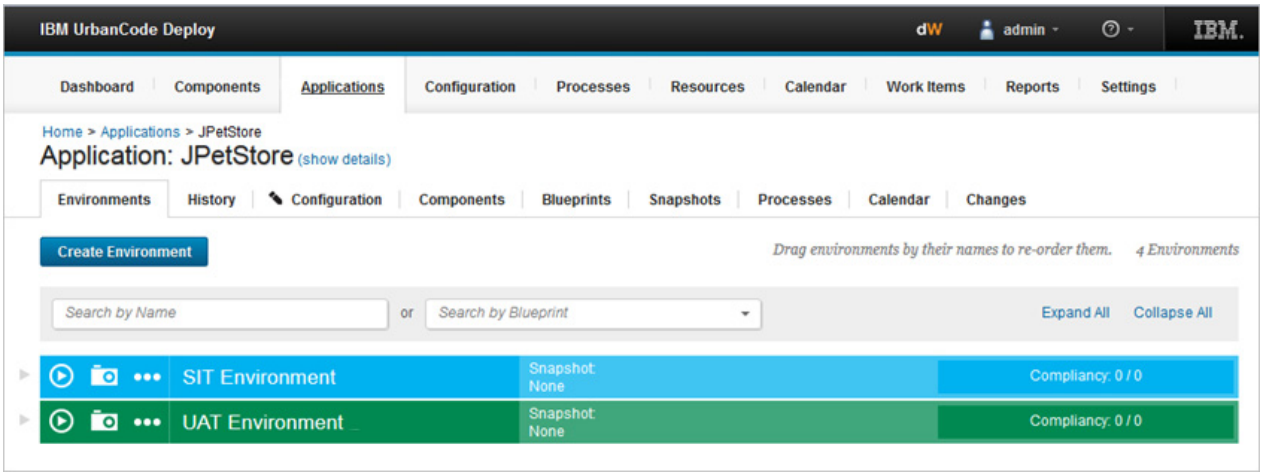
2. Type the name `UAT Environment`, and select a color. Update the properties as shown in the following table (**Configuration** > **Environment Properties**), and click **Save**:

Table 2 Properties for the UAT environment

Name	Description (optional)	Value
tomcat.home	The Tomcat home folder on the target computer	/opt/webservers/uat-apache-tomcat
db.url	The URL to the MySQL database, relative to the target system	jdbc:mysql://localhost:3306/jpetstore_uat
tomcat.url	The location of the Tomcat manager application	http://localhost:${p:environment/tomcat.port}
tomcat.start	The location of the startup script for Tomcat	${p:tomcat.home}/bin/startup.sh
tomcat.port	The port for the environment	8086



You now have two environments.

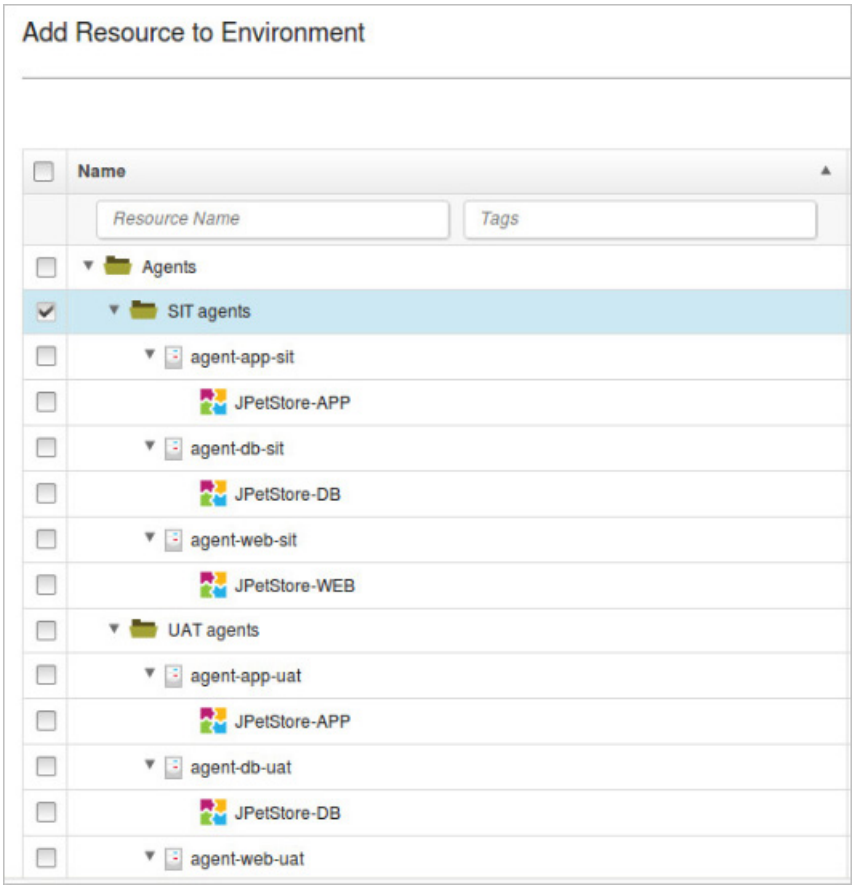


Exercise 6 Adding resources to the SIT environment

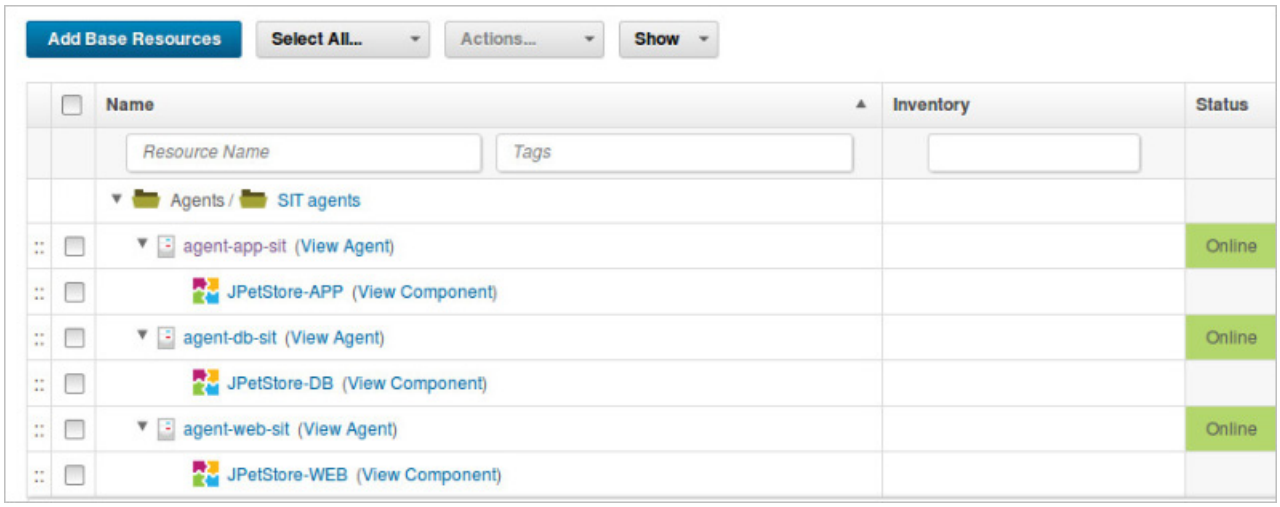
When you set up the agent, you added the agent and components as resources to a resource group. Now you can use that resource as part of the environment.

Add the agent resource group to the SIT environment:

1. Click the **Resources** tab for the environment. Make sure that you are on the **Resources** tab for the **JPetStore-SIT** environment and not the **Resources** tab at the top of the page.
2. Click **Add Base Resources**. The Add Resource to Environment window shows all the resources that are available on the server.
3. Select the check box that is next to the **SIT agents**, and click **OK**.



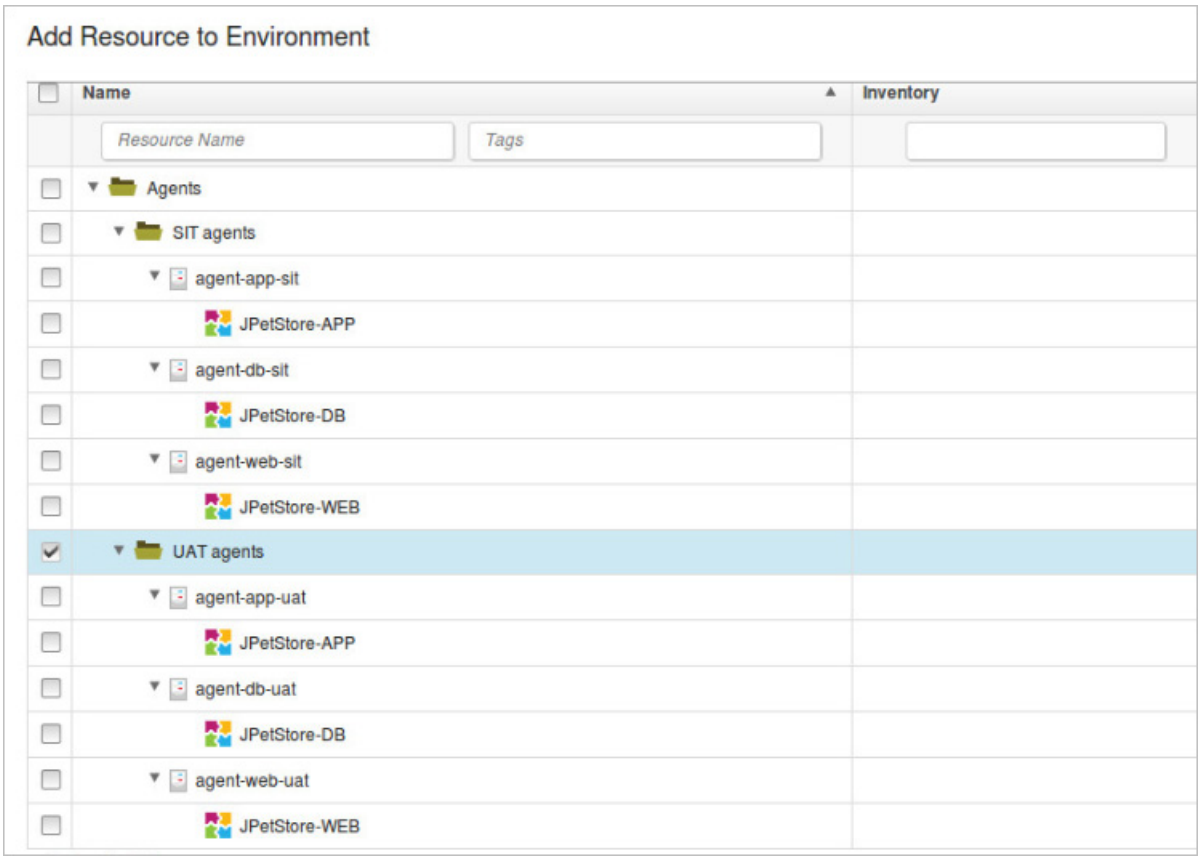
The resource is shown on the **Resources** tab for the environment, as in the following figure:



Exercise 7 Adding resources to the UAT environment

Next you add resources to the UAT environment in the same way you did for the SIT environment.

1. Select the **JPetStore-UAT** environment, and click **Add Base Resources**.
2. Select the **UAT agents** group, and click **OK**. After that, your UAT environment should look like this screen capture:



3. Click **OK**.

Exercise 8 Creating an application process

Application processes tie together component processes. In this task, you create an application process to install the components by calling each component process.

Create the application process:

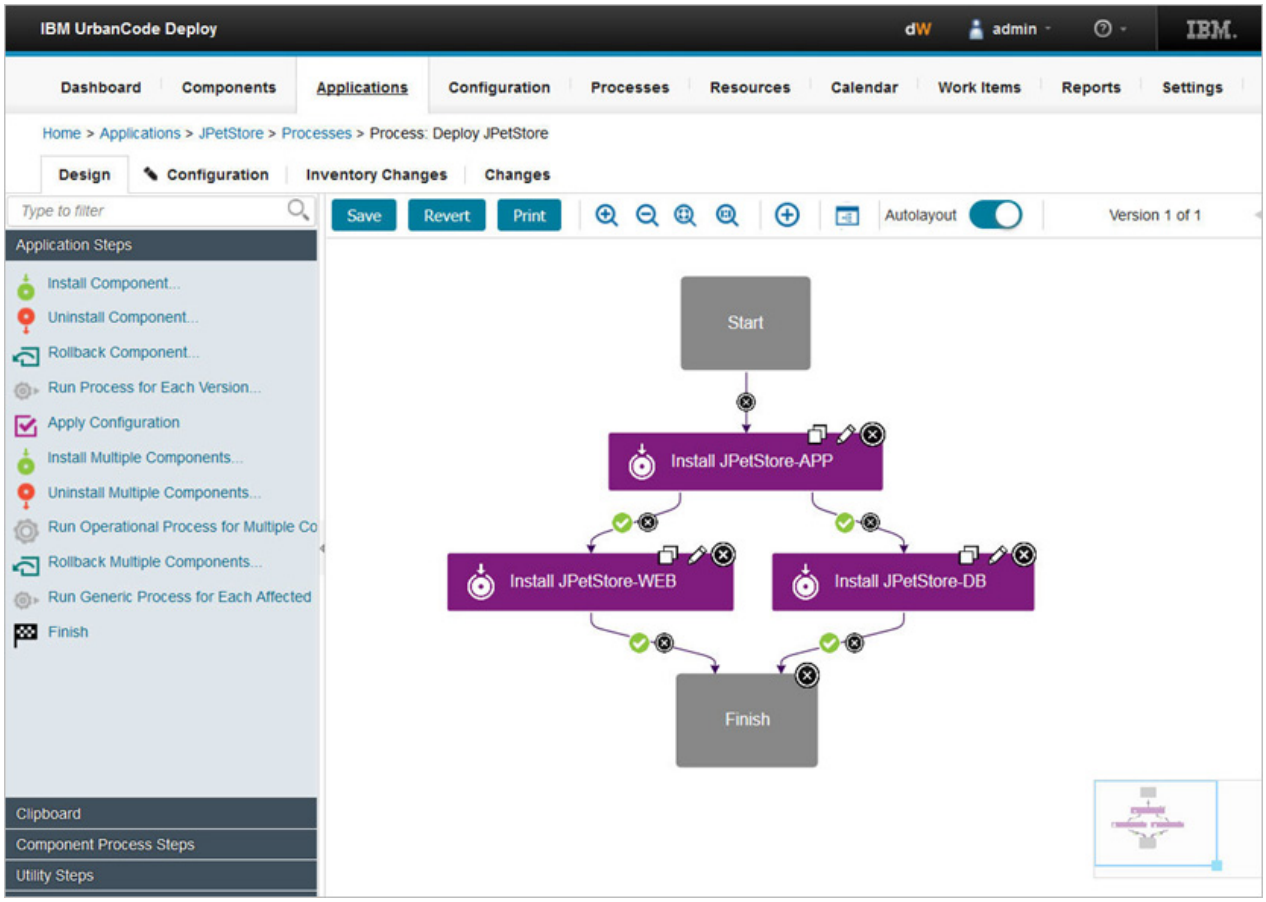
1. Click the **Applications** tab. Click the `JPetStore` application.
2. Click **Processes**, and click **Create Process**.

Hint: Be sure to click the **Processes** tab within the application and not the **Processes** tab at the top of the page, which leads to generic processes.

3. In the Create an Application Process window, name the new application process `Deploy JPetStore`.
4. Accept the default values for the other fields, and click **Save**.
5. Click the new process to open it in the process editor.
   The process editor for application processes is similar to the process editor for component processes. However, for application processes, the steps are limited to calling component processes and a few utility steps.
6. Add a step to deploy the application component:
   a. From the Application Steps drawer, drag the **Install Component** step to the process editor. This step calls component processes that have a process type of `Deployment`.
   b. In the Edit Properties window, name the step `Install JPetStore-APP`.
   c. In the **Component** field, select the `JPetStore-APP` component.
   d. In the **Component Process** field, select the `Deploy application component` process.
   e. Click **OK**.
7. Similarly, add a step that is named `Install JPetStore-WEB` to deploy the web component. Similarly, add a step that is named `Install JPetStore-DB` to deploy the database component.
8. Ensure that the `Start` step to is connected to the `Deploy application component` step.

10. Instead of connecting the remaining steps in a straight line, follow these steps to connect the steps so that they run at the same time. Use the figure at the end of this step as a reference.
    a. Connect the `Deploy application component` step to the `Deploy web component` step.
    b. Connect the `Deploy application component` step to the `Deploy database component` step. These two steps will run at the same time, after the application step.
    c. Connect both the `Deploy web component` step and the `Deploy database component` step to the `Finish` step.
11. Save the process.

The complete application process deploys the application component first and then deploys the web component and database components at the same time. The application process looks like the following figure:



Lab checkpoint

Now that the environment is prepared and the agent is running, you can run application processes on the environment to deploy the components.

Unit 4 Deploying applications to target environments exercises

Now that you have an environment and all the necessary processes, you can deploy the components by running the application process. When you develop a complex application, you use many tests to validate that all the application components work together. After your application passes the tests, you must ensure that you promote the correct collection of application components to the next step in your continuous delivery pipeline. You can use snapshots to specify all of the versions of an application's components that are used in an environment. The snapshot also specifies versions of configuration, such as deployment processes, that are used to deploy those components.

If you use snapshots to deploy to different environments in your continuous delivery pipeline, you no longer need to track and promote individual components through the environments in your pipeline. Instead, you promote configurations of components that were tested together. This ability to promote tested snapshots of components to environments is essential in maintaining a continuous delivery pipeline.

In these exercises, you run an application process to deploy components. After successfully deploying a set of components, you use snapshots to deploy the components, application, and processes that you created for the JPetStore application to a new environment.

After completing this lab, you will be able to complete these tasks:

- Run an application process to deploy components
- Deploy specific component versions to update the application
- Create a snapshot and deploy it to a new environment

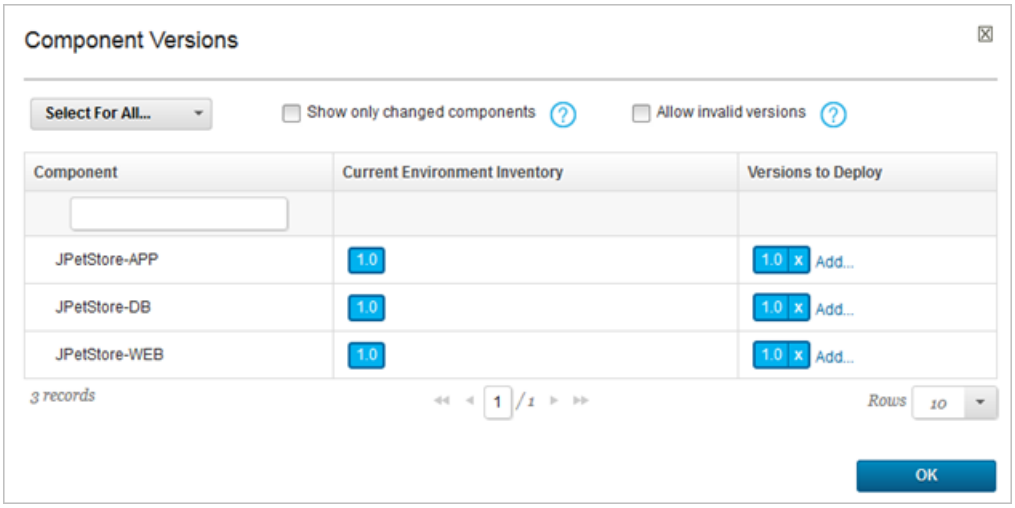
You use the JPetStore APP, DB, WEB components and processes, and the configured SIT and UAT environments.

Exercise 1 Deploying application components

The application process installs each of the components in the application by running their component processes.To deploy the components in the application, run the application process on the SIT environment.

Deploy the components:

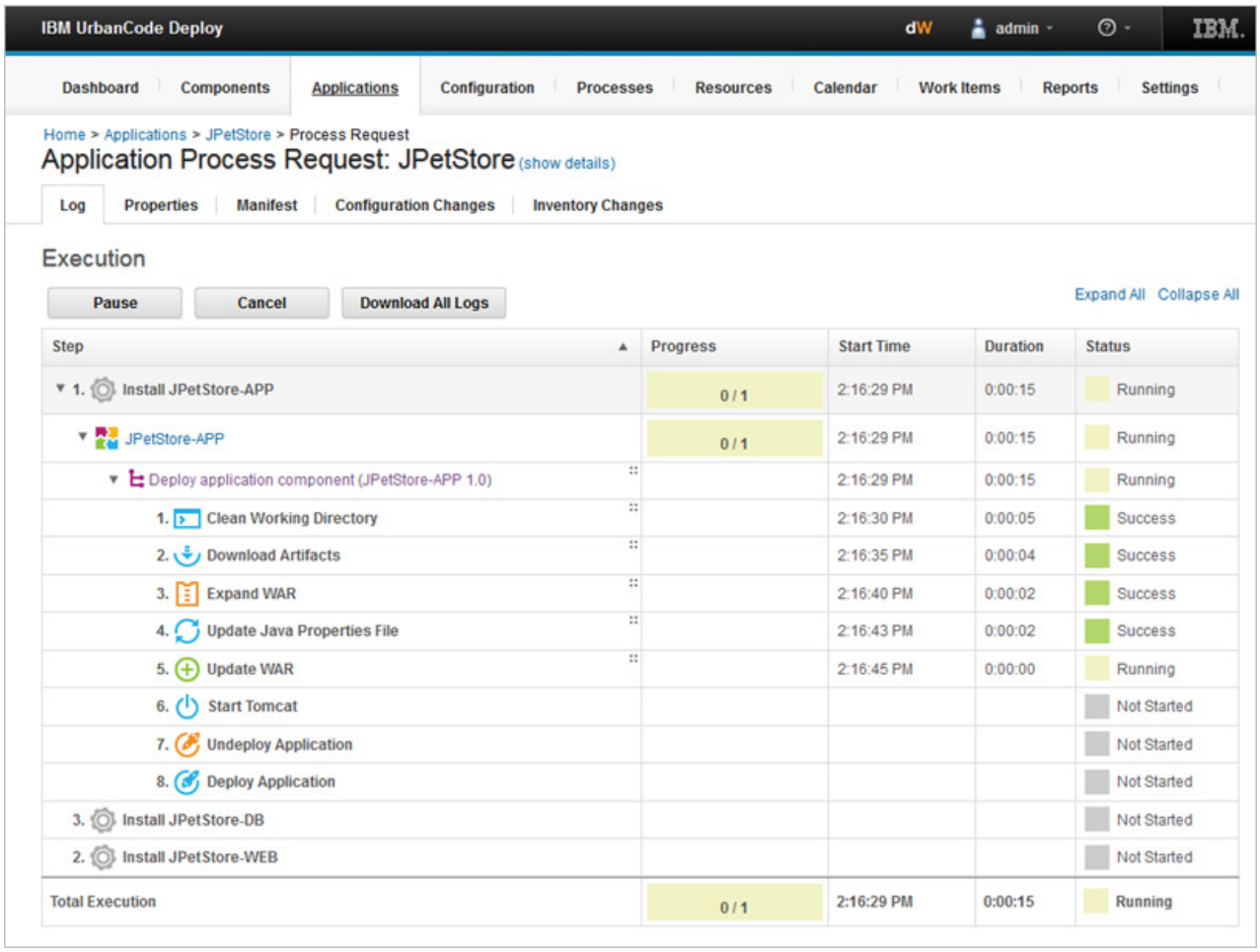
1. Open the application page: click **Applications,** and click the **JPetStore** application name.
2. In the same row as your SIT environment, click the **Request Process**  icon.
3. In the Run Process window, in the **Process** list, select the **`Deploy JPetStore`** process.
4. Under Component Versions, click **Choose Versions** and at the top of the window, click **Select For All > Latest Available**. Make sure that version 1.0 selected for each component, as shown in the following figure:



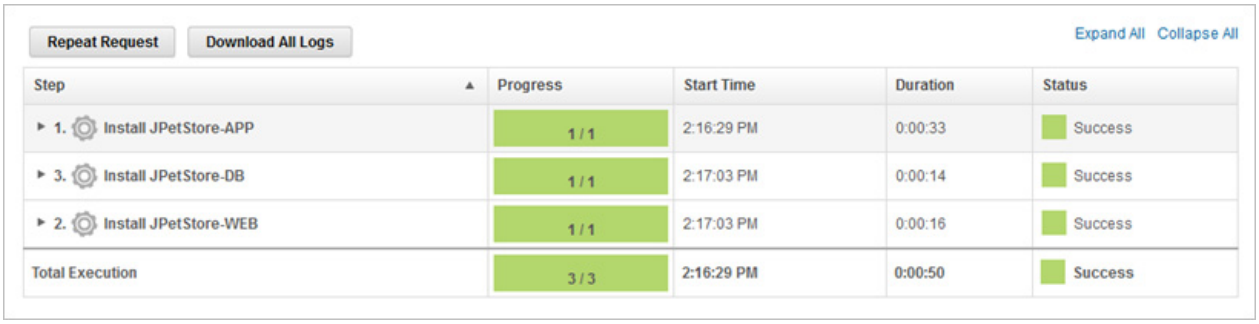
5. Click **OK**. You must select a version for each component; if you do not select a version, that component is not deployed.
6. Click **Submit**.

The web page shows you the progress of the application process request. From this page, you can watch as the processes run. The following figure shows that the application process is partially

completed. The application component process is finished and the other two component processes are running.



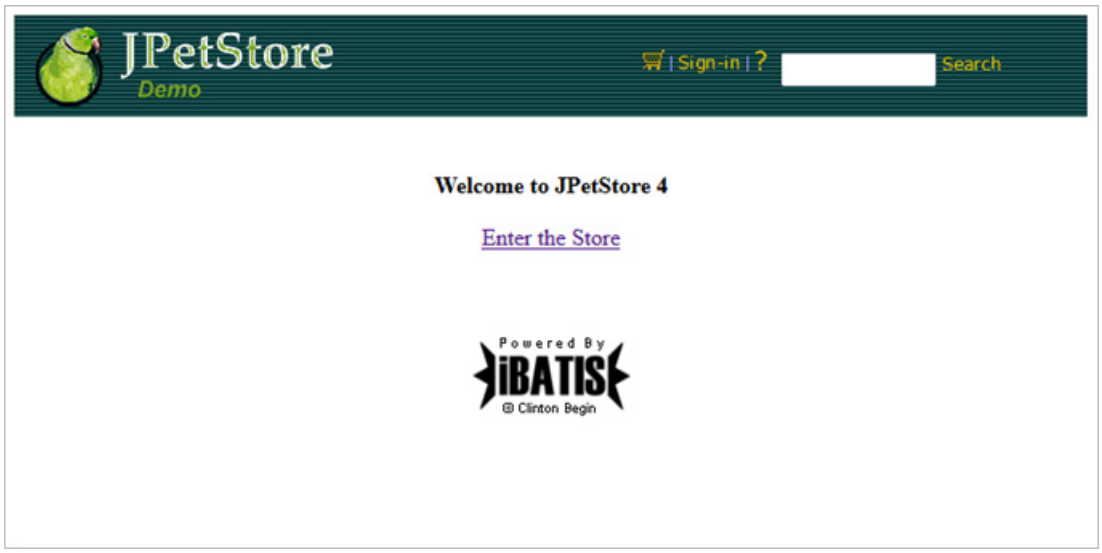
If the process runs to completion, the request shows that each component process is finished, as in the following figure:



7. View the running application at the following URL

   `http://localhost:8085/JPetStore/`

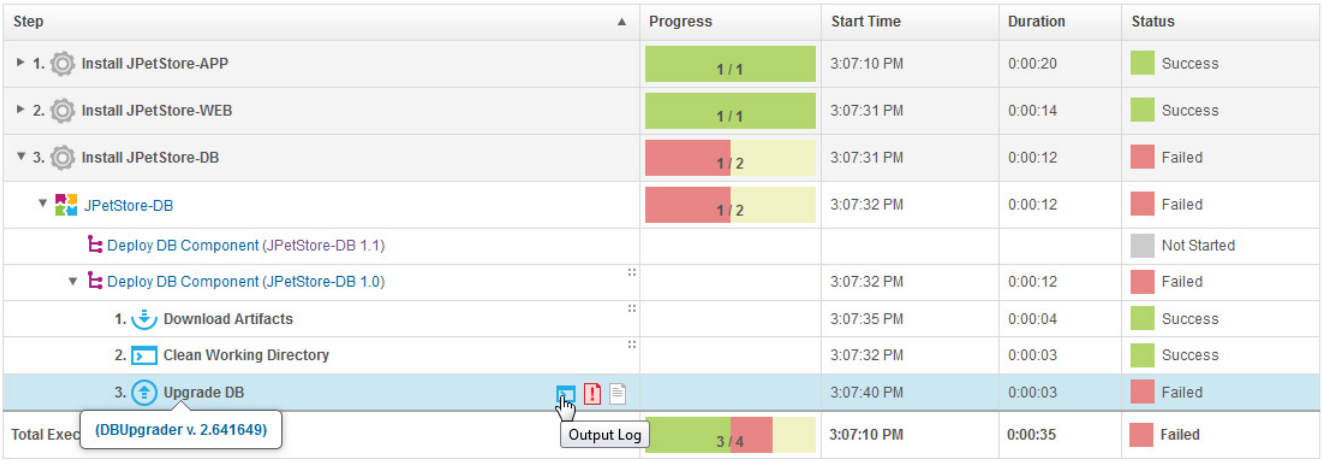
The application shows a simple online shopping site in the form of a pet store. The following figure shows the home page:



8. From the home page, click **Enter the Store**, and browse the items that are available, as shown in the following figure:

If the application process did not run to completion, one of the component processes might have failed. In this case, the process is listed with a status of `Failed`.

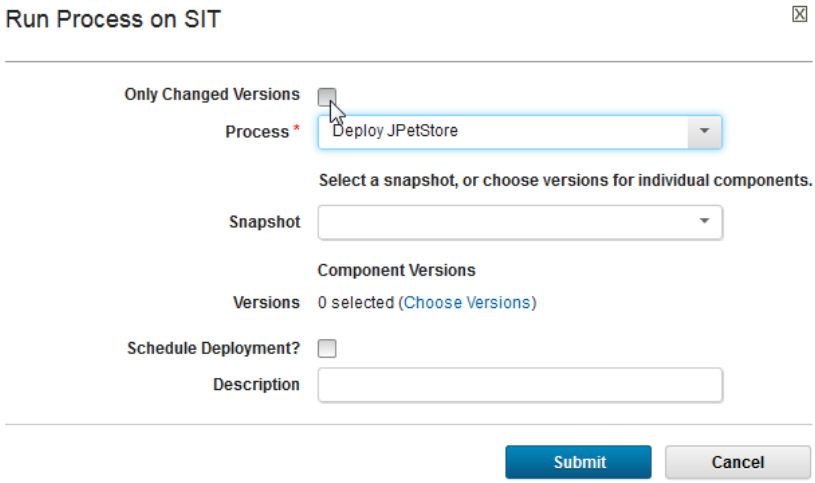
9. To troubleshoot this failed process, expand the log section for the component, and find the step that failed:



10. From here, you can look at the command-line output log for each step in the process by clicking the **Output Log** icon. Or you can inspect the log file to the right of the step, which is the input properties log.

11. If one of your component processes was not complete, identify the step that did not finish. Verify that the properties for that step are correct and that the steps are in the correct order in the process. Run the application process again.

Note: If you run the application process again, be sure to clear the **Only Changed Versions** check box, and select the versions of each component. If you leave the **Only Changed Versions** check box selected, the server runs the component processes only for components that have new versions. When you select **Choose Versions**, the next page is a **Component Versions** page where you can click **Select for All** > **Current Environment Inventory**.



Exercise 2 Updating the application

In the components exercise, you deleted certain versions of the components. In this task, you add those versions again to simulate deploying a new version of the components.

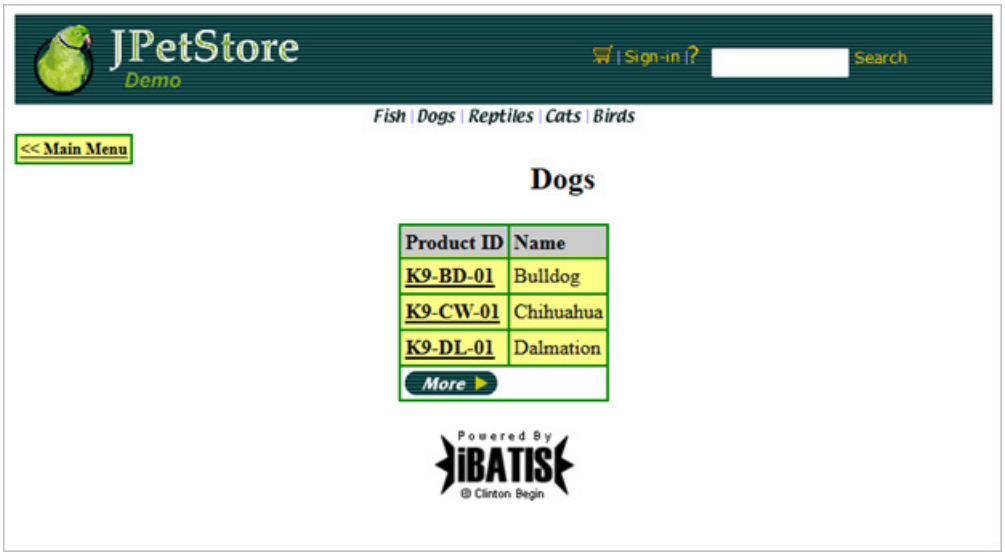
The web application that you deployed in the previous task included version 1.0 of the web component and database component. Version 1.1 of these components includes a new item for the online store: a Bichon in the Dogs category.

Update the application:

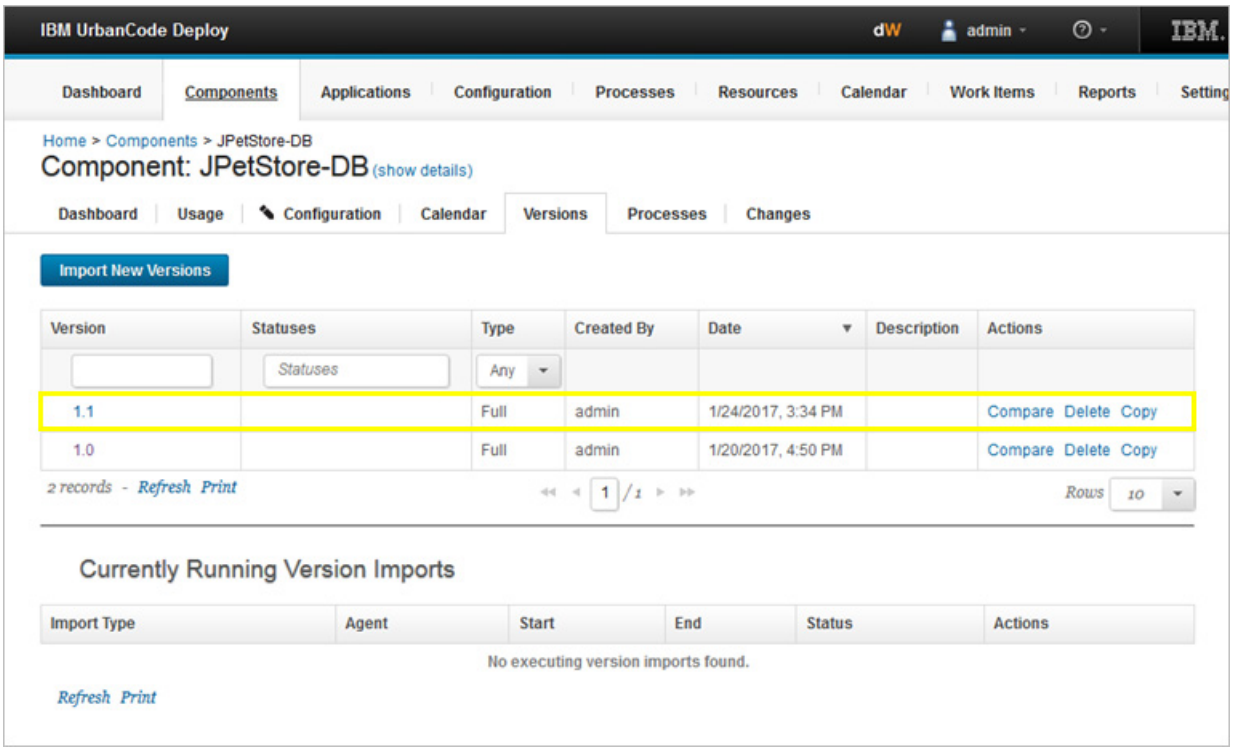
1. Verify that `Bichon` is not available in the Dogs category:
   a. Open the application by opening a web browser to the following URL:
      `http://localhost:8085/JPetStore/`

Note: The URL is case-sensitive.

   b. Click **Enter the Store**.
   c. Browse the Dogs category and verify that `Bichon` is not available. The following figure shows some of the dogs in this category:

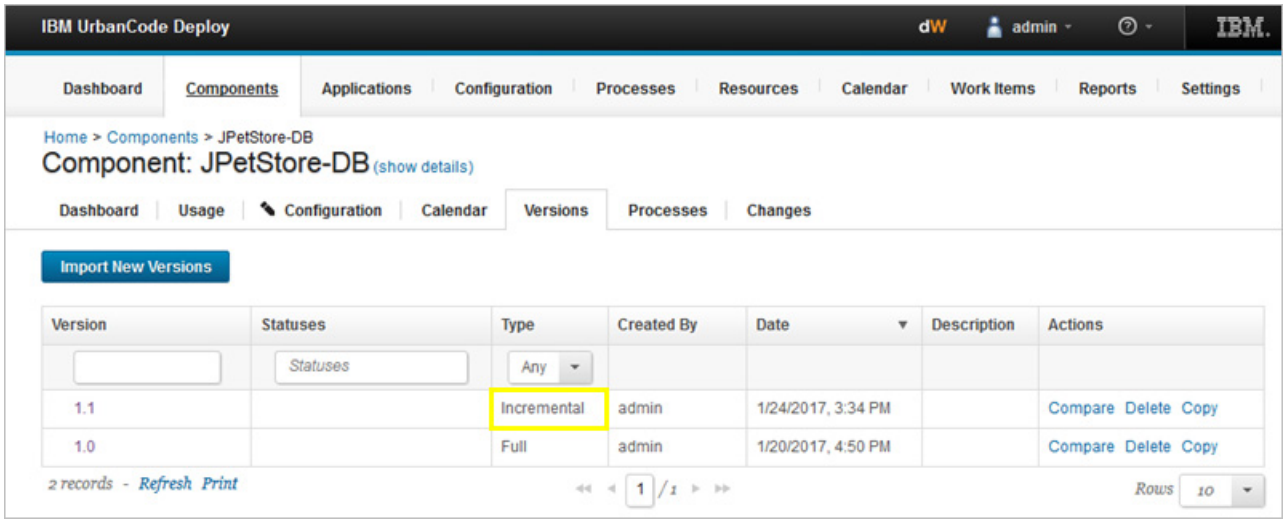


2. Import the new version of the database component:
   a. On the IBM UrbanCode Deploy server, click the **Components** tab, and click the **JPetStore-DB** component.
   b. Click the **Versions** tab, and click **Import New Versions**. Version 1.1 is shown in the list of versions, as in the following figure:



   c. Click the new version of the database component to open its information page.
   d. Click the **Configuration** tab.
   e. Under Basic Settings, in the **Type** list, select **Incremental**, and click **Save**.

f. Select the **Versions** tab again and verify the change:



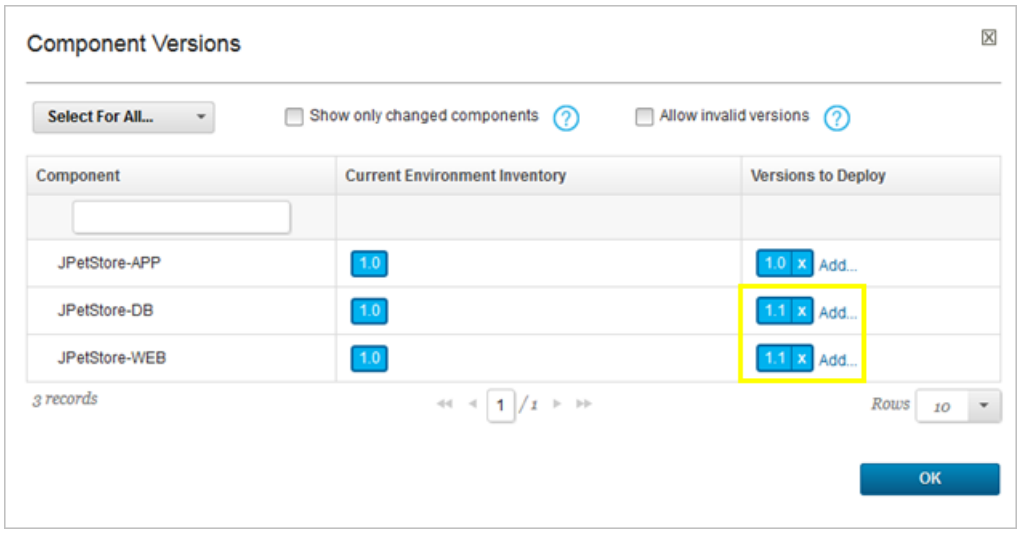
This version is set to be an incremental upgrade rather than a new version. Full versions are installed directly, without installing prior versions first. Incremental upgrades start with previous versions, and then the new version is installed over the earlier versions. Because the new database version is only an addition to the database, it requires the earlier version of the component.

3. In the same way, import version 1.1 of the web component without changing the **Type** setting. The application component does not have any additional versions.

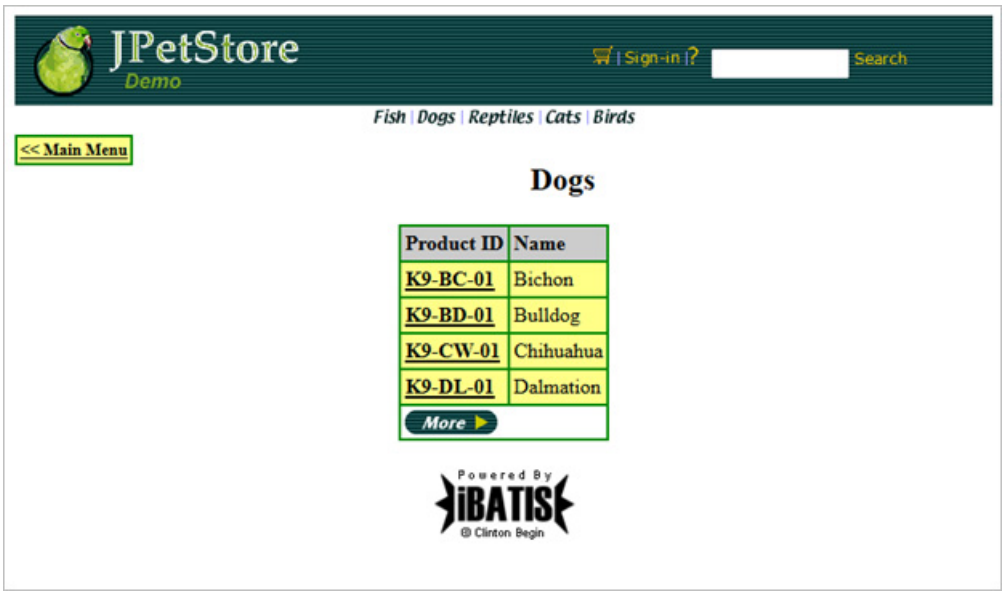
4. On the environment, run the application process again:
   a. In the Run Process window, be sure to select **Latest Version** for each of the components by clicking **Choose Versions** and then clicking **Select For All** > **Latest Available**.

   b. Optionally, you can select the **Only Changed Versions** check box so that the server deploys only the components with new versions.

In this case, the server runs the component processes only for the web and database components.

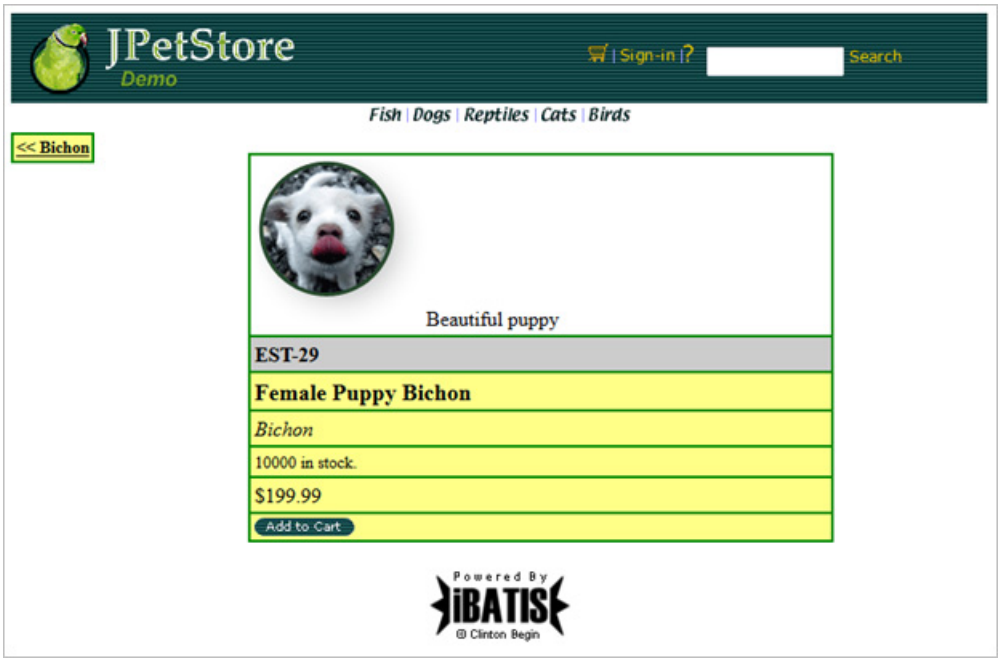


5. After the deployment is finished, in your web browser, reload the web application, and explore the Dogs category to see the new `Bichon`, as shown in the following figure. Click **More** to see the full list of dogs.



This new item demonstrates that the database component is updated to version 1.1.

6. Click the Product ID for `Bichon` and the Item ID to see the product page for `Bichon`. The new graphic for this item demonstrates that the web component is updated to version 1.1.



The application includes the new versions of the components.

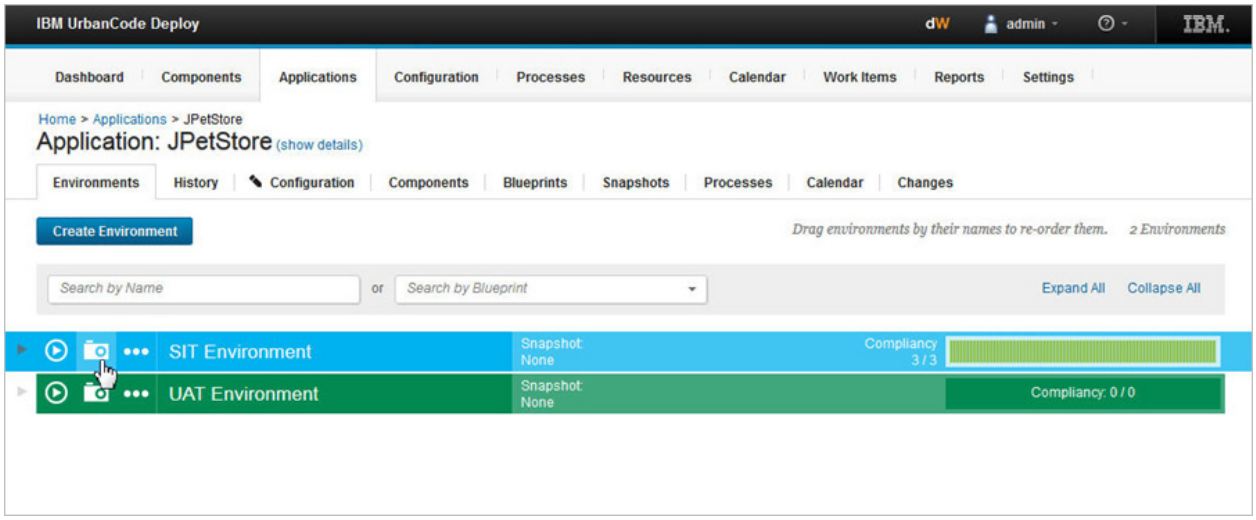
In a production scenario, new versions of components might become available frequently. You can run the application process as often as required to update the deployed components. You can also configure the application process to run automatically when new versions of the components are available.

Exercise 3 Creating a snapshot of the SIT environment

Creating a snapshot involves saving a collection of component versions that are part of an application. In the previous exercises, you deployed the JPetStore application twice to the same environment. In the second deployment, you used two components that contain new information. In this task, you create a snapshot that uses the same components as the deployed environment. You create a snapshot from a functioning development environment in preparation for deployment to the next phase of your continuous delivery pipeline. In a later lesson, you deploy this snapshot to a new environment. By deploying with snapshots in IBM UrbanCode Deploy, you can quickly bring a new environment to a specific state.

Create a snapshot:

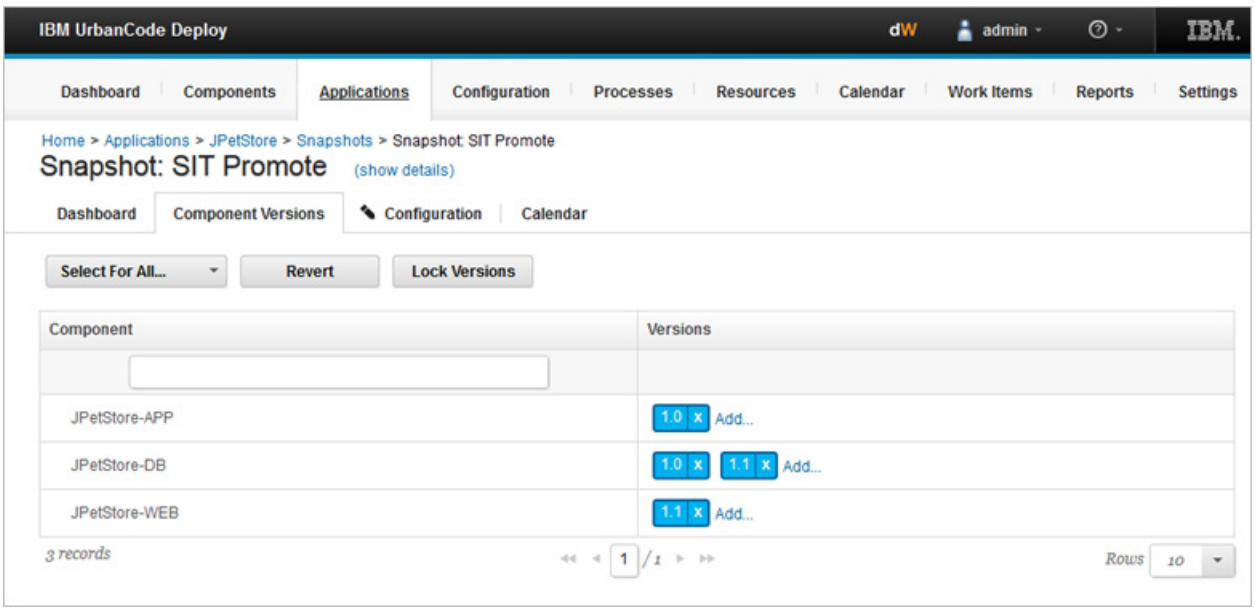
1. To create a new snapshot, navigate to the JPetStore application, and click the **camera** icon that is next to the SIT Environment listing.



2. Name the snapshot `SIT Promote`, and click **Save**.

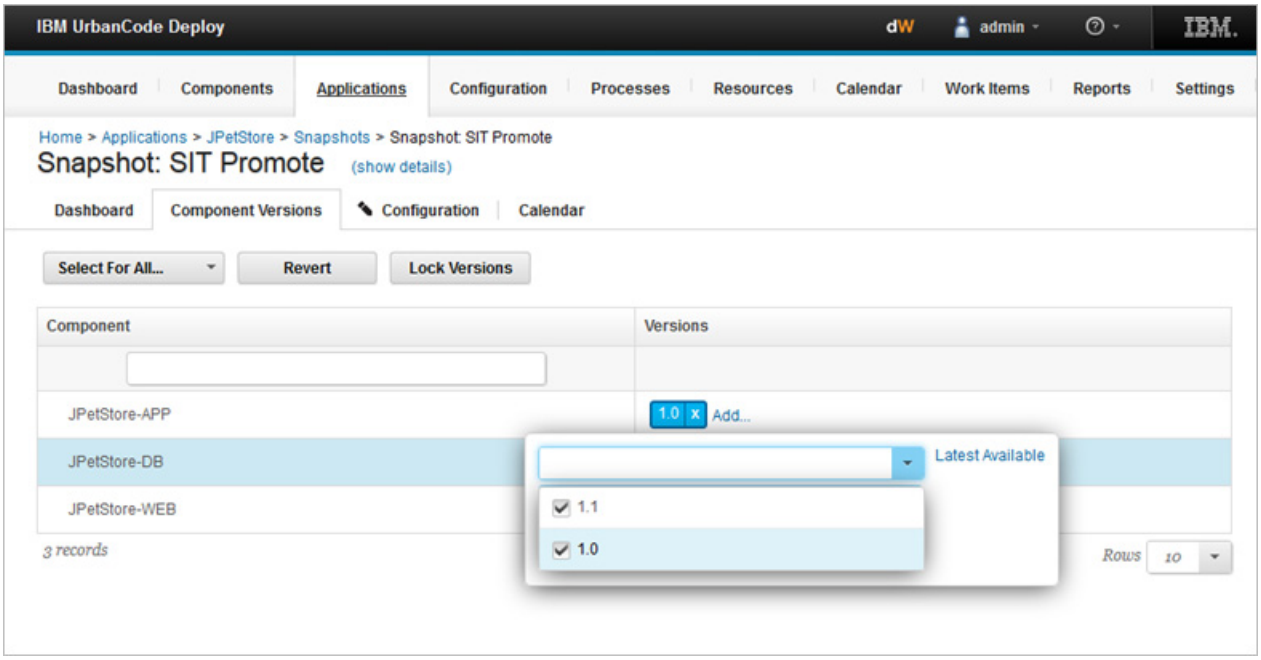
   The snapshot is populated with the content of the current SIT environment.

3. Open the **Component Versions** tab to view the components.



The Database component includes two versions. The Incremental version 1.1 is not sufficient to recreate the environment because it only includes changes to a full version. So both the incremental 1.1 and the original 1.0 are automatically included.

4. If you do not see two versions of the Database component, add the missing version manually by clicking the **Add** button.

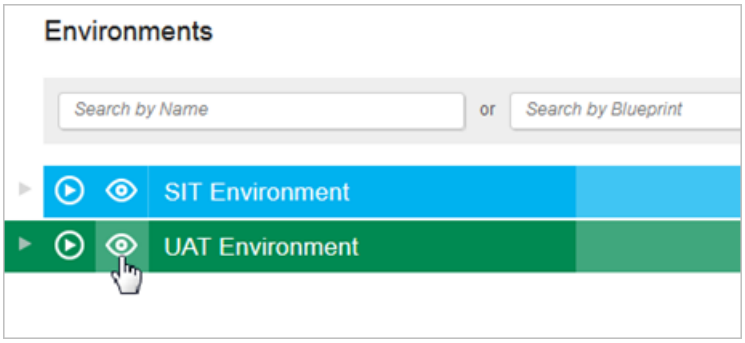


Exercise 4 Previewing the snapshot deployment to the UAT environment

You can preview an application snapshot before you deploy it to see the upcoming changes in resources and properties. On the SIT Promote snapshot, preview what happens when it is deployed to the UAT environment.

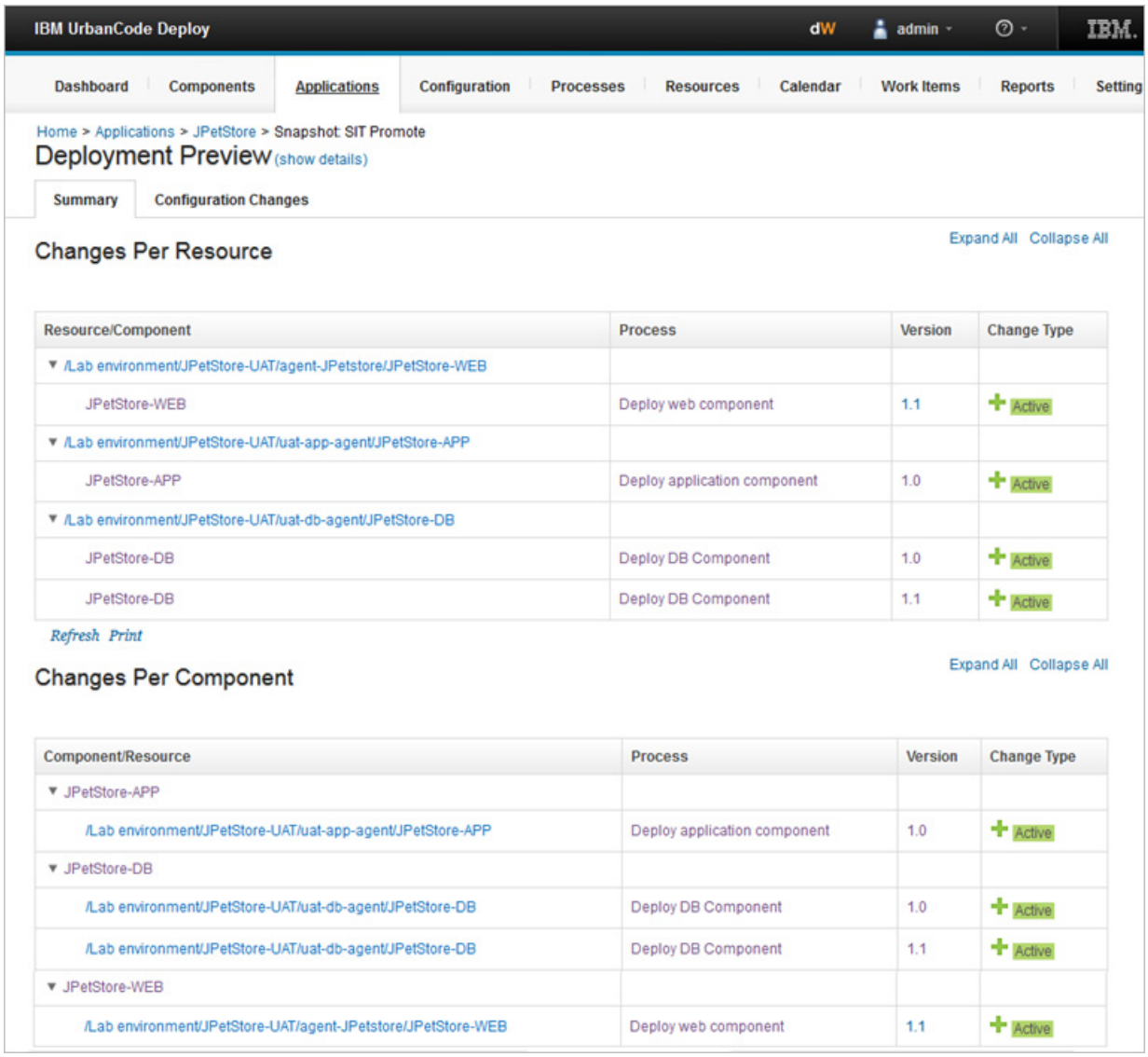
Preview the snapshot deployment:

1. Click the **JPetStore** application, click the **Snapshot** tab, and click the **SIT Promote** snapshot.
2. On the snapshot, in the **UAT** environment row, and click the **Preview** icon.



3. In the Preview Deployment window, select **Deploy JPetStore**, and click **OK**

The changes to be made to resources and components are displayed:

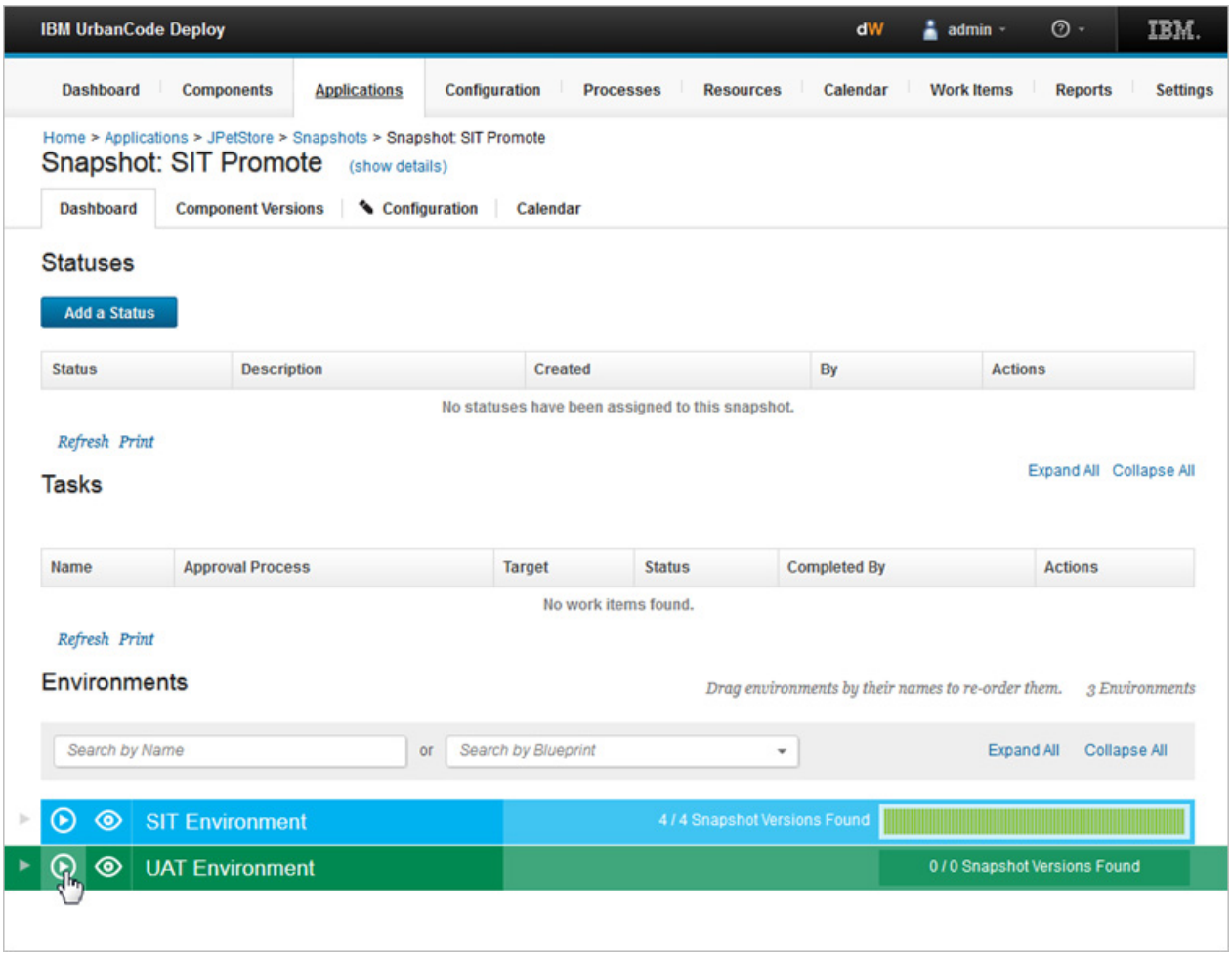


Exercise 5 Requesting deployment to the UAT environment

Deploying a snapshot deploys the components in the snapshot. The components are deployed in the order in which they were added to the snapshot or to the environment on which the snapshot is based.

Deploy a snapshot to the UAT environment:

1. On the snapshot dashboard, next to the **UAT environment,** click the **Request Process** icon.

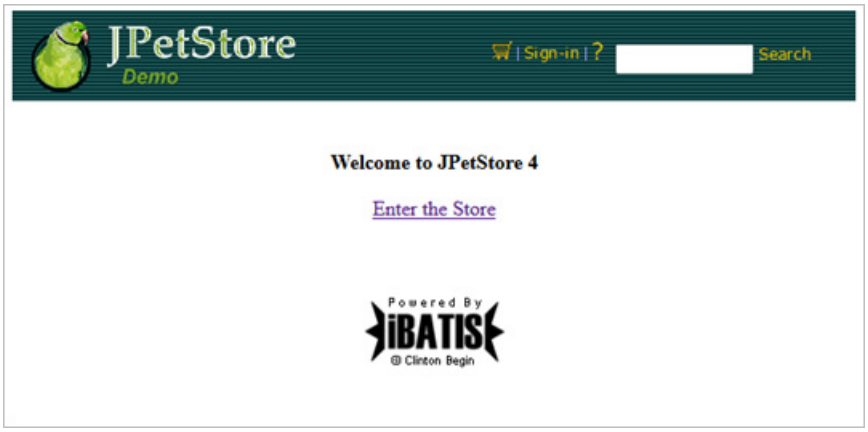


2. Select the **Deploy JPetStore** process and click **Submit**.
3. View the details of the deployment to the UAT environment.
4. If errors occur, troubleshoot the errors similar to the way you did when you initially deployed to the SIT environment.

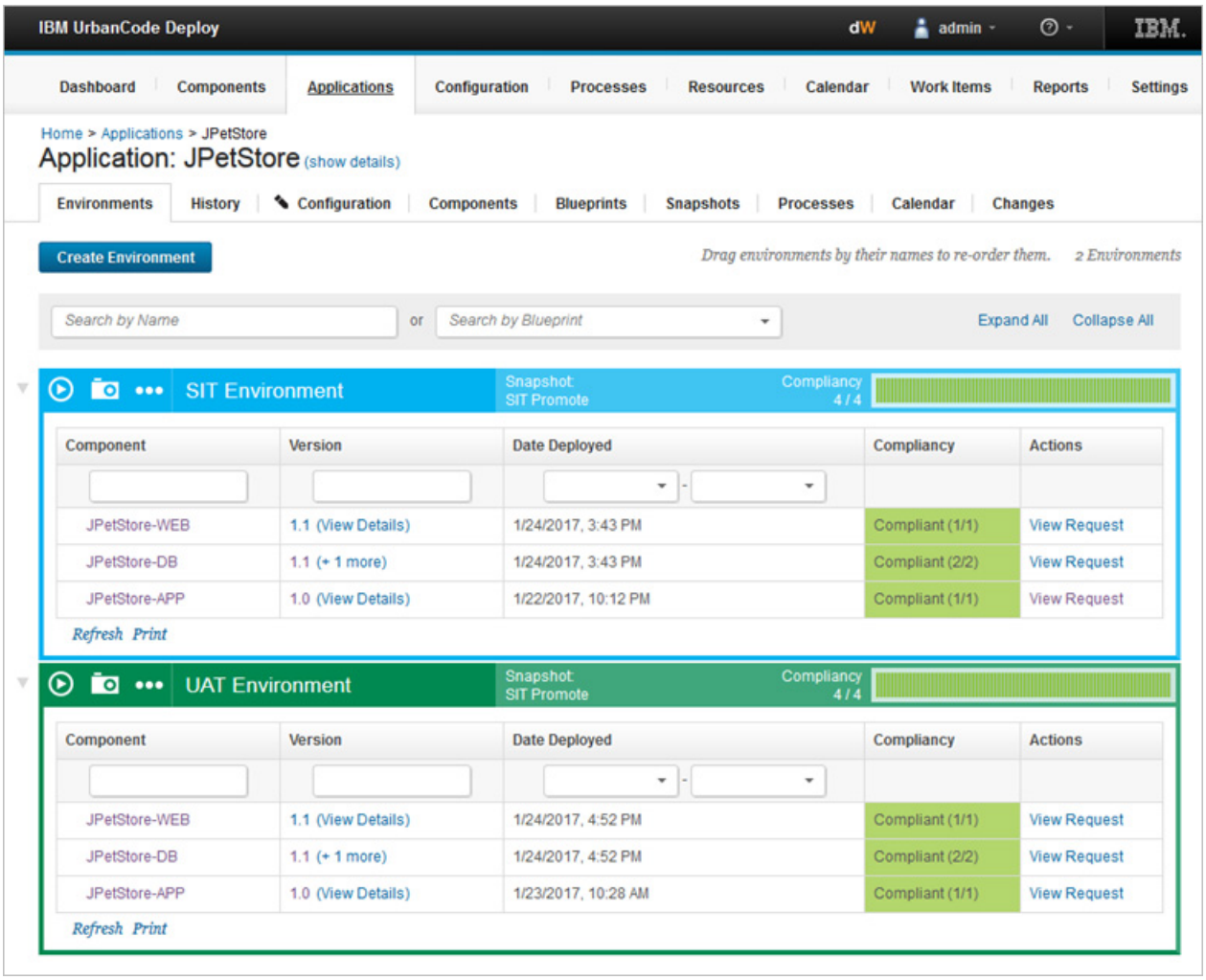
Exercise 6 Verifying the promotion of the updated Tomcat server

Test the application to verify that it runs in the UAT environment:

1. You can view the running application at the following URL. The application shows a simple online shopping site in the form of a pet store. Point your browser to `http://localhost:8086/JPetStore/`



2. To see the deployed components to each environment, click the **Environments** tab of the application



Lab checkpoint

In this exercise, the application process installed each of the components in the application by running their component processes. Then you created a snapshot of the components in the application and deployed the snapshot to different environment.

In most cases, you use snapshots to manage complex deployment environments in IBM UrbanCode Deploy. You typically use snapshots to control deployments to different environments on different servers. However, for simplicity, in this exercise, you used snapshots to deploy an application to two different environments on a single server.

Unit 5 Setting up server security, approvals, and quality gates exercises

Teams and roles

IBM UrbanCode Deploy uses the term *permission* in a way that is familiar to most readers: each permission controls access to a product area or product function. In these exercises, you use authentication for a new user that logs in to IBM UrbanCode Deploy. Authorization is based on a set of roles, that are applied to a set of teams, which are applied to a set of objects. Each role defines a set of actions, and each team contains users who are assigned to roles. Objects include applications, environments, and components.

After completing this lab, you will be able to perform these tasks:

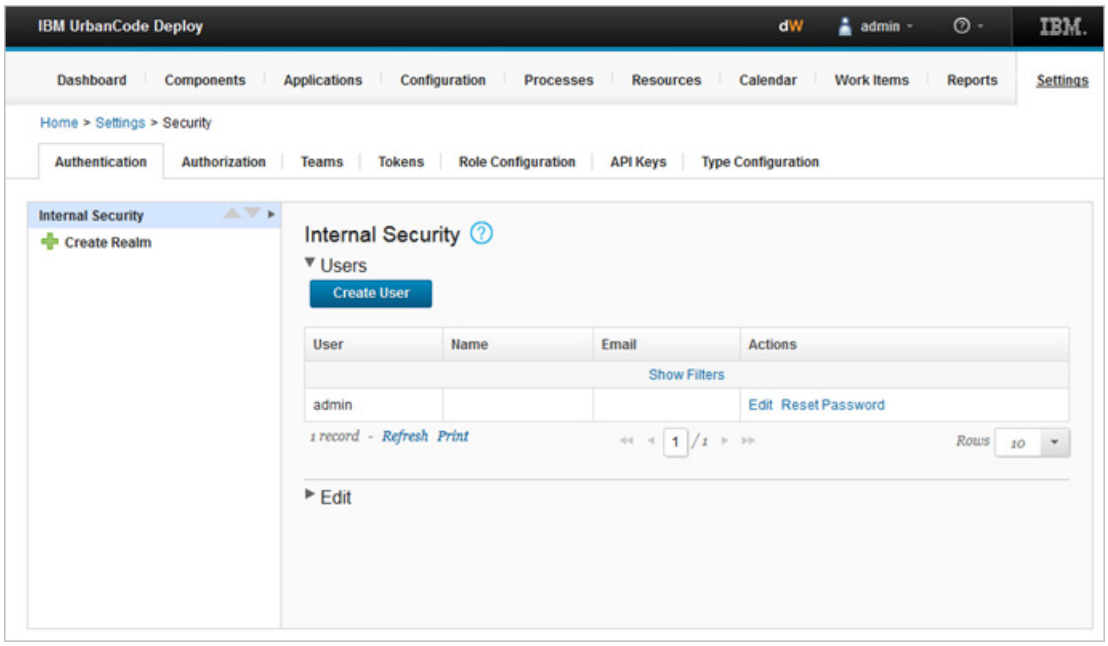
- Create a new user and role
- Create a team
- Associate the team to the existing application and environments

Exercise 1 Creating a new user in the internal security realm

Authentication realms manage users and determine user identity within authorization realms for the server.

Create a user on the server for Internal Security realm:

1. From the top-level **Settings** tab. In the **Security** column, click **Authentication (Users)**.

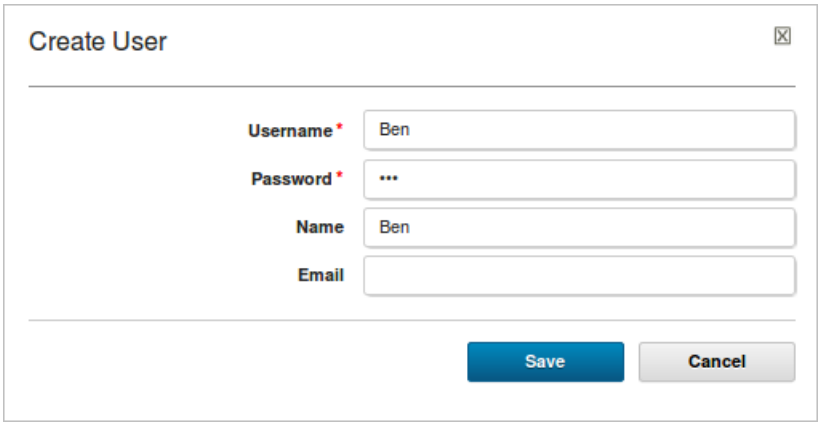


2. Click the **Create User** button and create a new user:

   Username: `Ben`

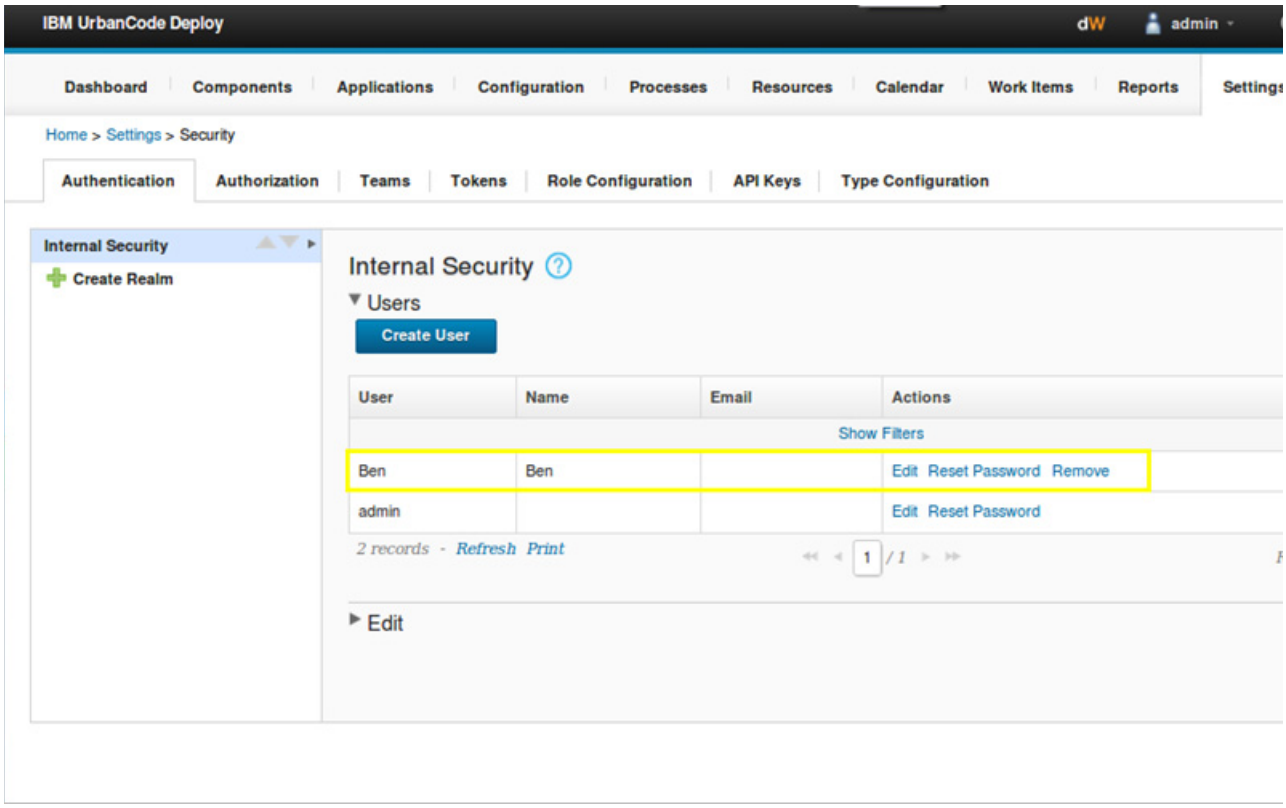
   Password: `ben`

   Name: `Ben`



3. Click **Save**

You now have a second user:



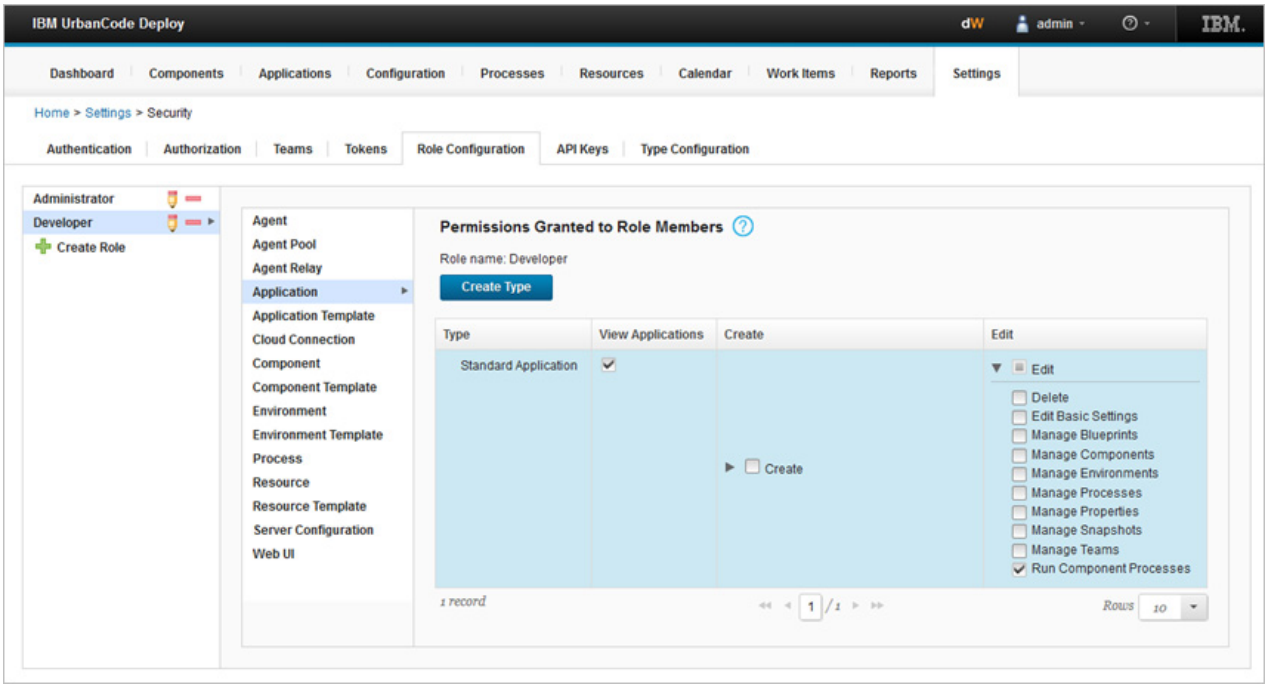
After you add a user to an authentication realm, you must also add the user to groups or teams.

Exercise 2 Creating a Developer role

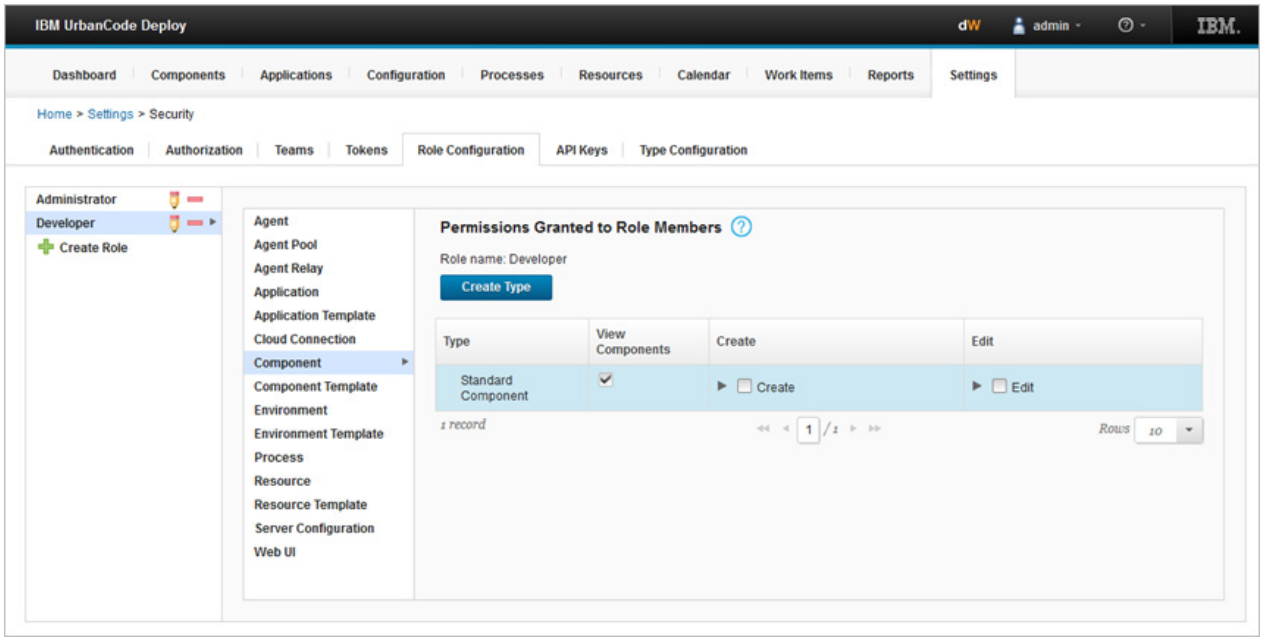
A role is a set of granted permissions. A developer-type role might have permissions for creating applications but not running them in production environments. Alternatively, a deployer-type role might be able to run applications but not create them. You decide the number of roles and their functions. By default, IBM UrbanCode Deploy provides one role, the administrator role, which is granted permissions for all security types. You must create your own roles and specify the permissions for each role.

Create the Developer role:

1. In **Security**, click the **Role Configuration** tab.
2. Click **Create Role** to create a role, and name it **Developer**.
3. Click the **Developer** role in the left column, and do the following actions to add appropriate permissions:
   a. Click **Application** in the second column.
   b. For the standard application, select the check box in the **View Applications** column.
   c. Expand the drop-down list in the **Edit** column, and select **Run Component Processes**.



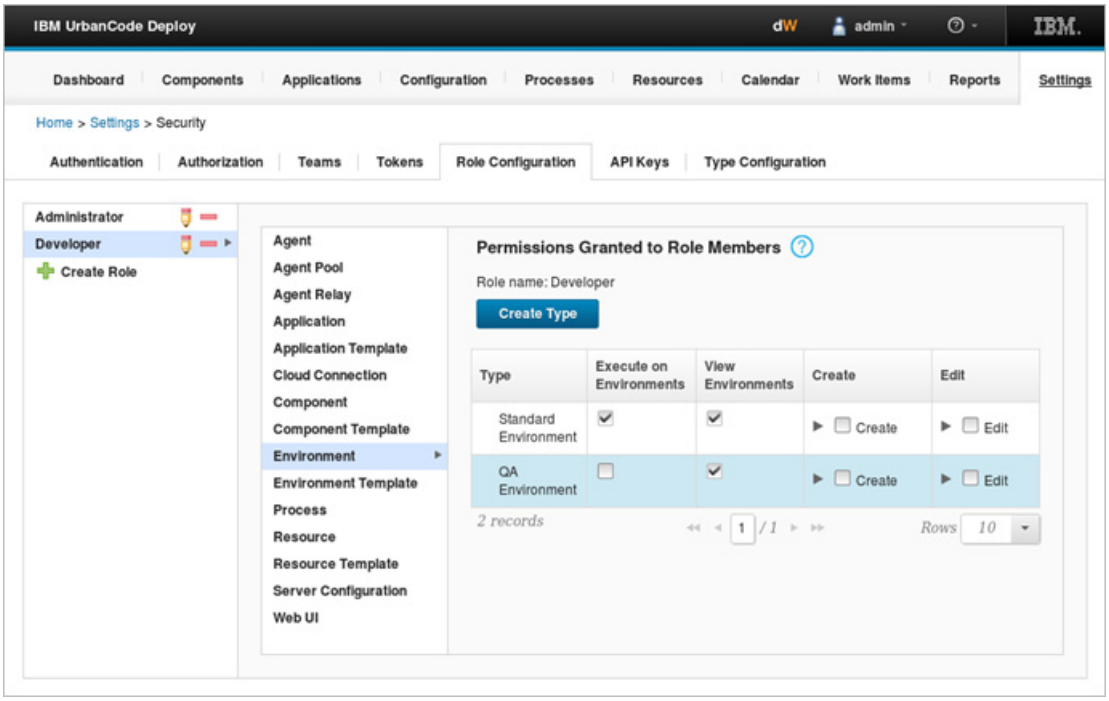
d. Click **Component** in the second column, and select the **View Components** permission for the standard component.



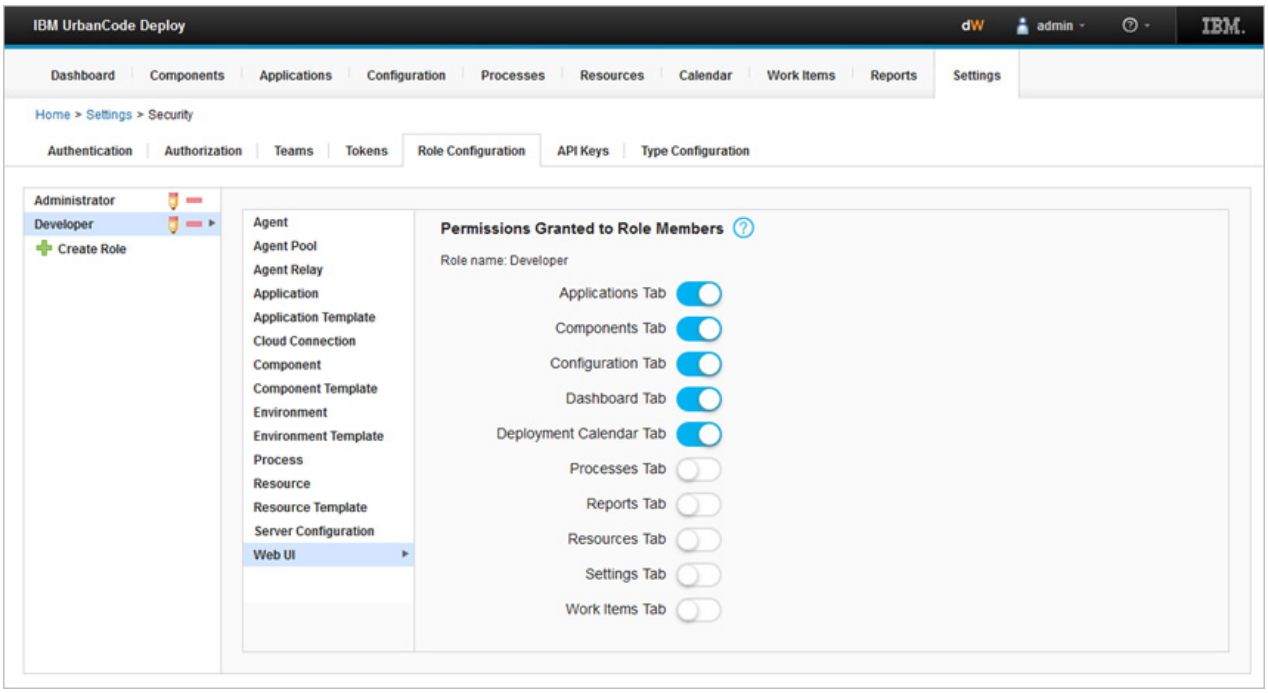
e. Click **Environment**, and click **Create Type** to create a new type called **QA Environment**.

f. For the Standard Environment, select **Execute on Environments** and **View Environments**.

g. For the QA Environment, select **View Environments**.



h. Click **Web UI**, and grant permissions to **Applications Tab**, **Components Tab, Configuration Tab**, **Dashboard Tab**, and **Deployment Calendar Tab**.



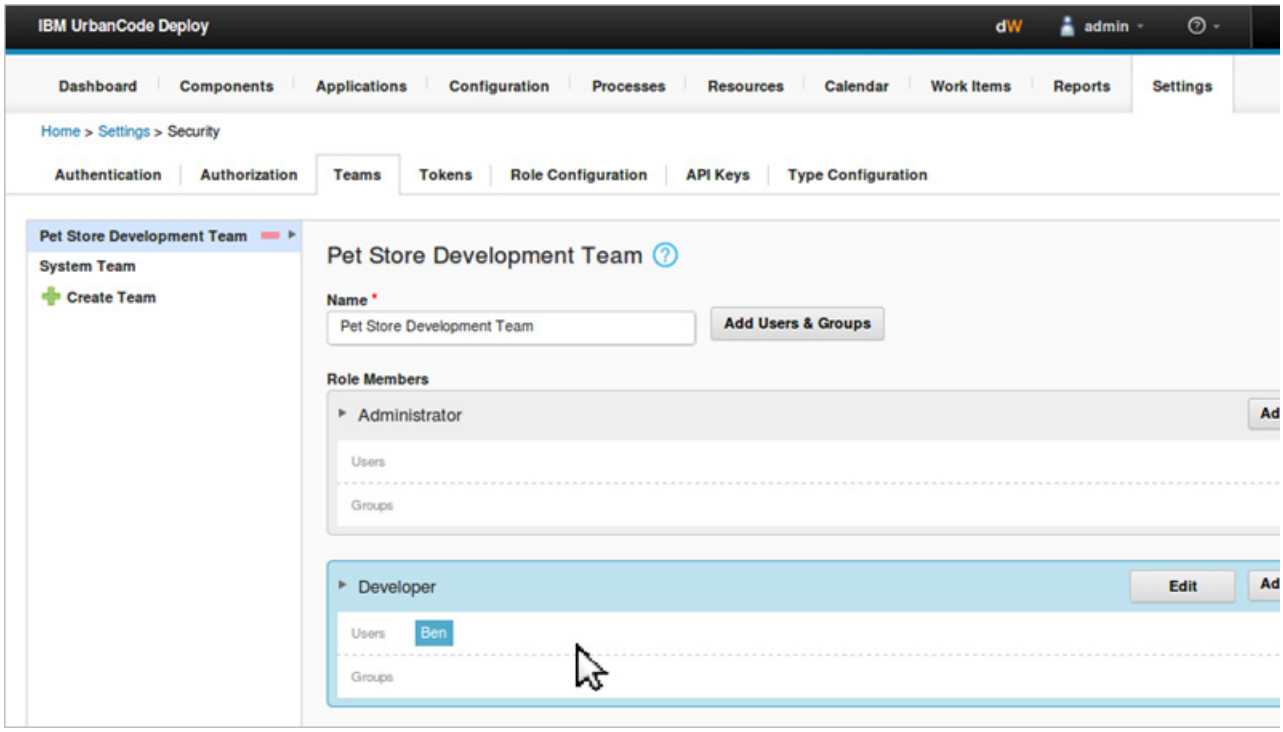
When you create a role, that role becomes available on each team. Then you can assign users to that role for a specific team.

Exercise 3 Creating the Pet Store Development Team

A team is a construct that associates users or groups with roles. When a user is added to a team, that person is assigned to one or more roles. Users cannot be added to a team unless they have a role assignment or to a role that does not have a team assignment. Role members are granted all permissions that are defined for the role.

Create the Pet Store Development team:

1. Still in Security, click the **Teams** tab.
2. Click **Create Team**, name it **Pet Store Development Team**, and click **Save**.
3. In the Developer section, click **Add**, and select **Ben**.
4. Click **OK**.
5. Verify the new definition**.**

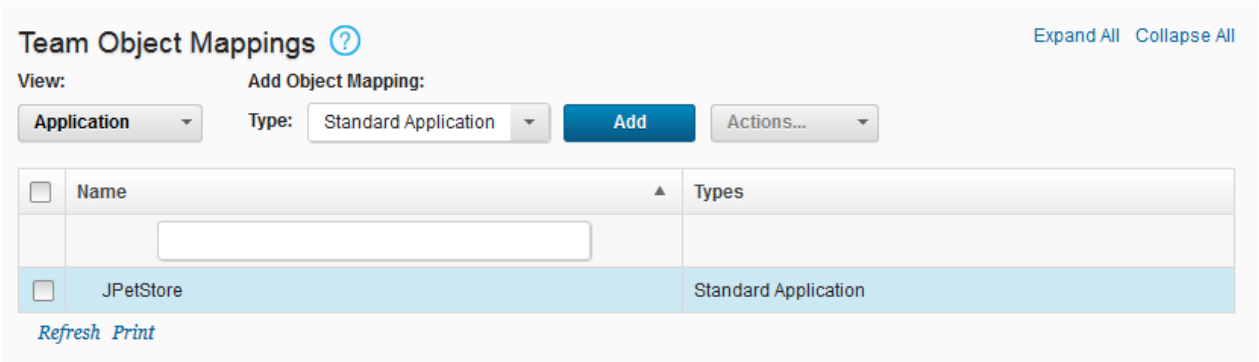


Exercise 4 Applying the team to the existing JPetStore application

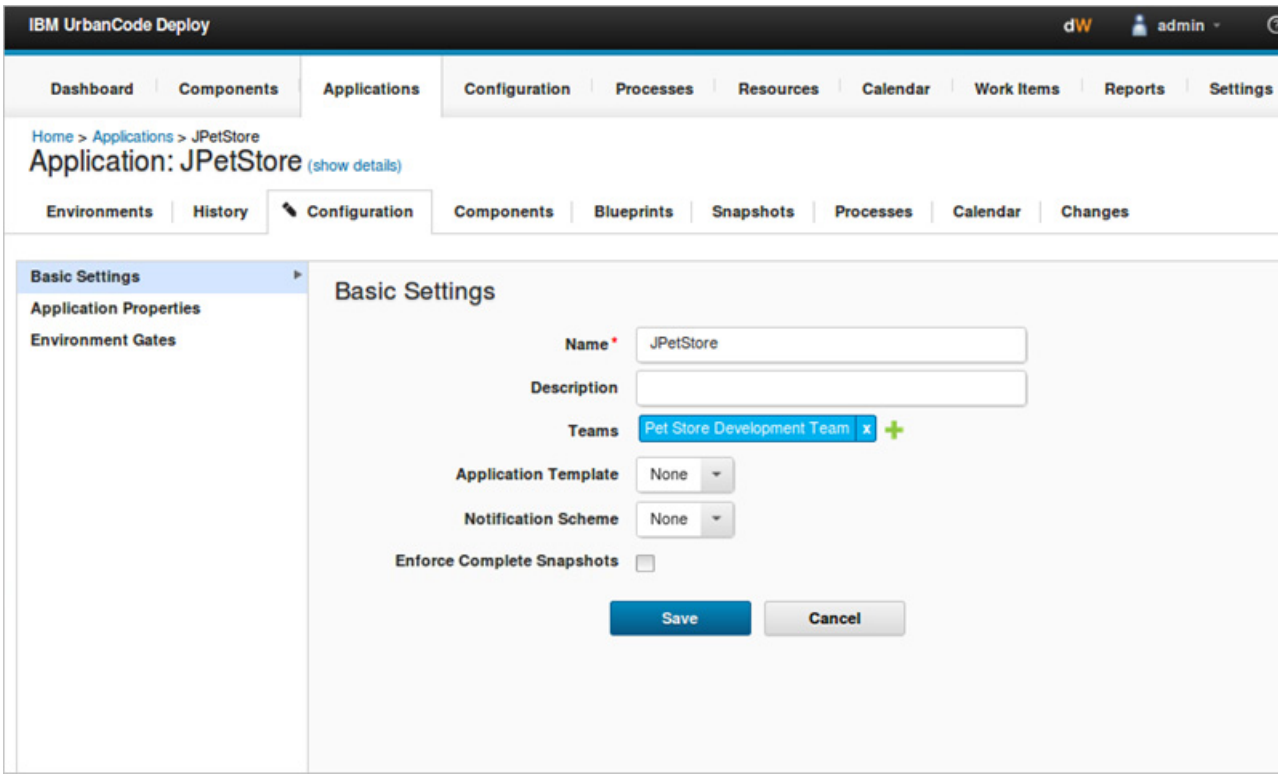
When you assign a team to an object, such as an application, only team members with the appropriate permissions can interact with the affected object.

Add the team to the JPetStore application:

1. On the **Teams** tab**,** at the bottom of the page, go to the **Team Object Mappings**.
2. In the **View** drop-down list, select **Application,** and click **Add**. JPetStore is now listed.
3. Select **JPetStore**, and click **OK**.



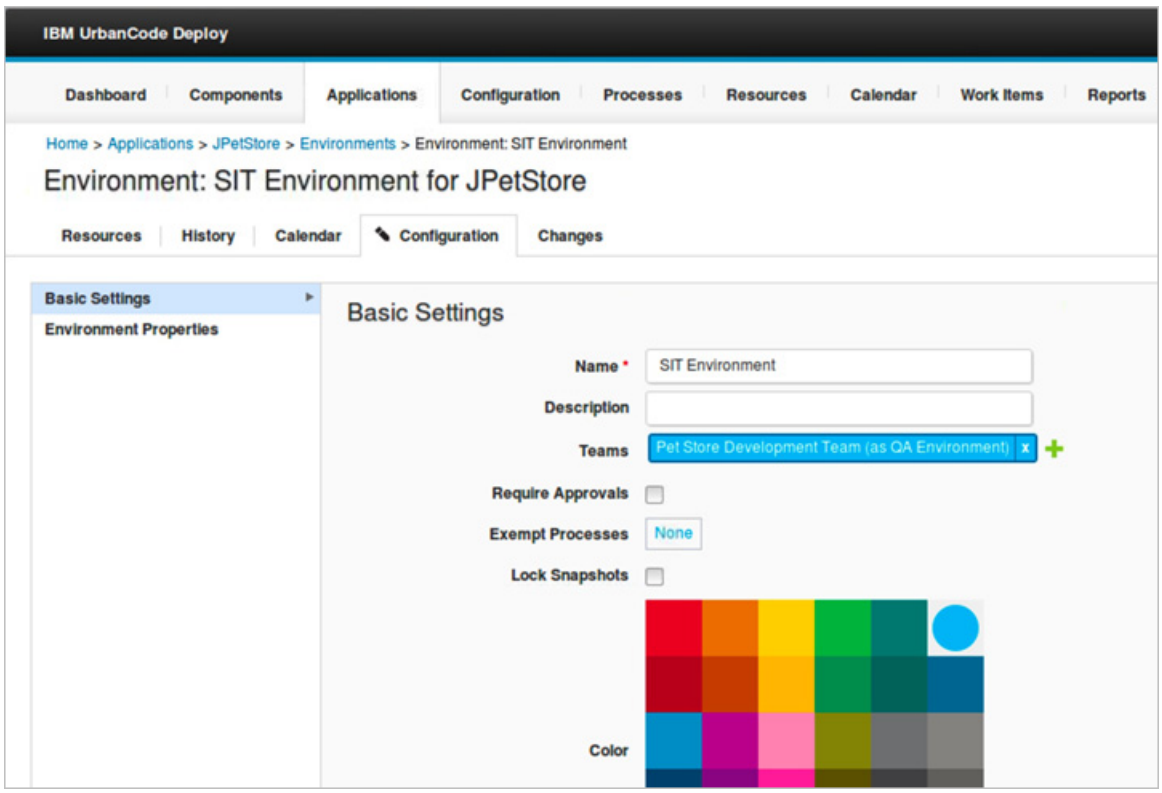
4. View the rules that you defined in the existing application.
   a. In the top-level navigation, click the **Applications** tab, and select **JPetStore**.
   b. Select **Configuration > Basic Settings**.



JPetStore Team is already next to the **Teams** field.

5. Update the SIT environment to include Pet Store Development Team with a Standard Environment role:
   a. From **Applications > JPetStore > SIT Environment**, select **Configuration > Basic Settings**.
   b. Next to the **Teams** field, click the **Add** icon.
   c. In the **Team** field, select **Pet Store Development Team**.
   d. In the **Type** field, select **QA Environment**.

6. Click **Save**.



7. Repeat the same steps for the UAT environment.

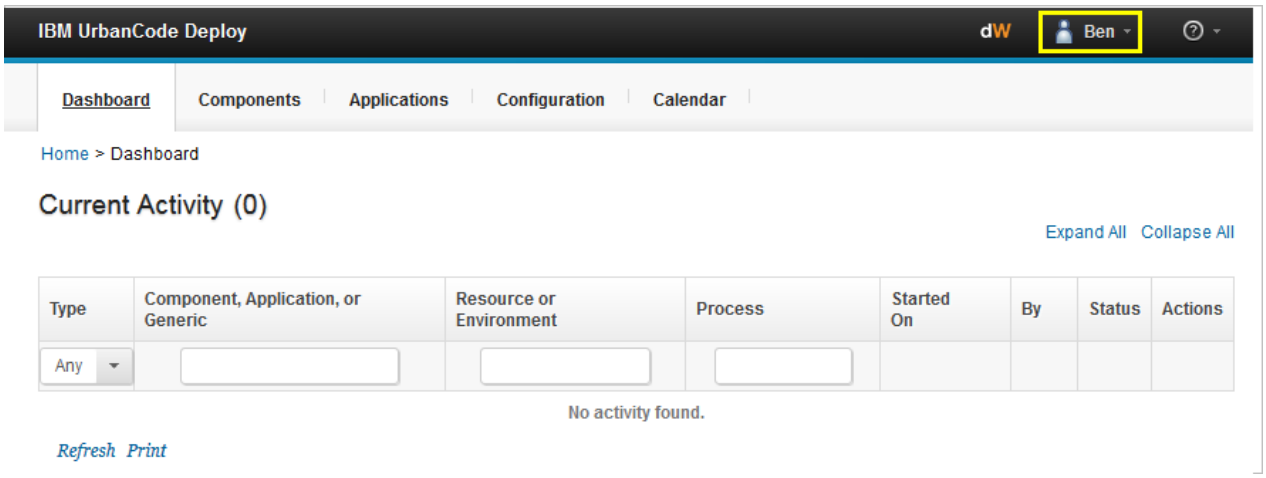
The name of each environment is just a tag. The IBM UrbanCode Deploy system does not understand the semantics of SIT versus UAT. Using environment roles provides a way to define different security rules around the environment type.

Exercise 5 Logging in as a JPetStore developer

Check the permissions of the JPetStore developer:

1. Log out of IBM UrbanCode Deploy as **admin** by clicking **Sign Out** from the **admin** drop-down menu.
2. Log in with user name **Ben** and password **ben**.

Fewer high-level tabs are available because of the configurations to the Web UI. Notice also that, although Ben can see UAT and compare the content against other environments, he does not have the rights to deploy.



Lab checkpoint

In this exercise, you worked with permissions. You created a user in the Internal Security realm. Then you created a developer role with the following permissions:

- View standard applications and components, and run component processes
- Execute on standard environments, but only view environments from the QA environment
- Access the Dashboard, Components, Applications, Configuration, and Calendar tabs of the IBM UrbanCode Deploy web UI

Next you created a team and added Ben as a developer to give him the permissions of the developer role. Finally, you applied the team to the existing JPetStore application and environments to provide the team with access to those assets.

IBM Training