Course Guide

Design Application Deployment with IBM UrbanCode Deploy

Course code ZQ410G ERC 1.0

IBM Training

June 2017 edition

NOTICES

This information was developed for products and services offered in the USA.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

TRADEMARKS

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

IT Infrastructure Library is a Registered Trade Mark of AXELOS Limited.

ITIL is a Registered Trade Mark of AXELOS Limited.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

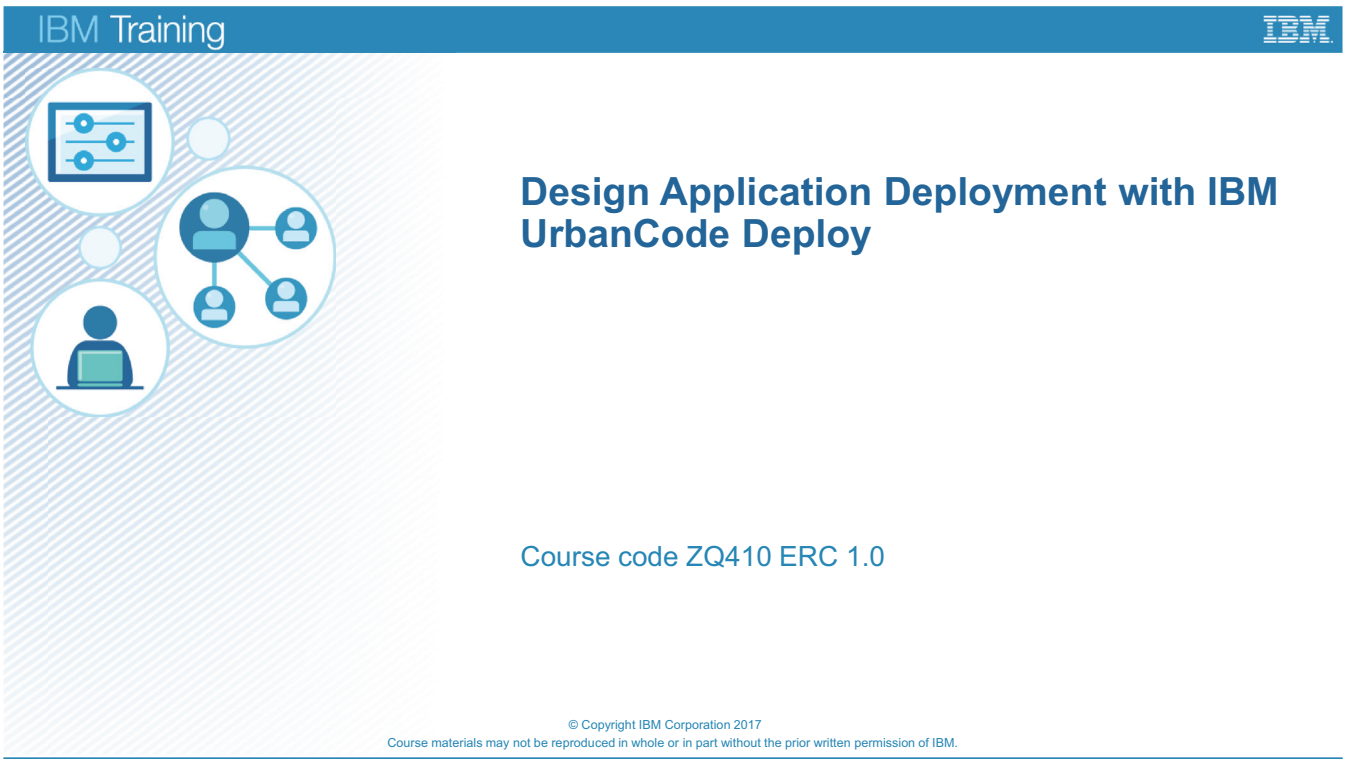
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Contents

About this course



IBM UrbanCode Deploy is a tool for standardizing and simplifying the process of deploying software components to each environment in your development cycle.

This is a beginner-level course for users of IBM UrbanCode Deploy, such as administrators, development teams, and operations leads. In this course, you learn how to plan and automate the deployment of applications to test environments. Hands-on labs use IBM UrbanCode Deploy and cover deploying a simple web application. You create components, create an application that contains those components, and then deploy the components to an environment.

The lab image for course ZQ410 is hosted on Skytap/IRLP. There is no image preparation required.

For information about other related courses, visit the IBM Training website:

http://www.ibm.com/training

	Details
Delivery method	Classroom or instructor-led online (ILO)
Course level	ERC 1.0. This course is an updated course.
Product and version	IBM UrbanCode Deploy 6.2.2
Recommended duration	1 day
Skill level	Beginner

Course objectives

- Describe how IBM UrbanCode Deploy supports the principles and practices of DevOps
- List the main concepts and features of IBM UrbanCode Deploy
- Plan and automate the deployment of applications to test environments
- Configure a security model with roles, approvals, and quality statuses and gates

Course objectives

Audience

This is a basic course for new users of IBM UrbanCode Deploy, such as administrators, performance testers, development teams, and operations leads.

Prerequisites

Before you take this course, you should have a basic understanding of client-server architecture and multitier application models.

IBM Training

Course agenda

- Unit 1: Designing software deployment and deploying applications
- Unit 2: Configuring components and component processes
- Unit 3: Defining deployment with resources, environments, and application processes
- Unit 4: Deploying applications to target environments
- Unit 5: Setting up server security, approvals, and quality gates

Course agenda

The course contains the following units:

1. Designing software deployment and deploying applications

   IBM UrbanCode Deploy is a deployment automation tool that standardizes and simplifies the process of deploying software components to each environment in the development cycle. In this unit, you learn about typical deployment problems and key deployment elements and concepts.

2. Configuring components and component processes

   Components represent deployable items along with user-defined processes that operate on them, usually by deploying them. In this unit, you learn how to create a component, import component versions, create component processes, and work with properties.

3. Defining deployments with resources, environments, and application processes

   Applications identify the component versions that must be deployed together. They also define the different environments that the components must go through on the way to production. In this unit, you learn to create an application and add components, create environments and add resources, and create application processes.

4. Deploying applications to target environments

   You run a deployment by running an application process in a target environment. In this unit, you learn how to deploy to an environment, create and deploy snapshots, and compare environments to view the inventory in each one.

5. Setting up server security, approvals, and quality gates

   IBM UrbanCode Deploy uses a flexible team-based and role-based security model that maps to your organizational structure. In this unit, you learn how to define and configure roles, set up approvals and notifications, and use quality statuses and gates.

Unit 1 Designing software deployment and deploying applications

IBM UrbanCode Deploy is a deployment automation tool that standardizes and simplifies the process of deploying software components to each environment in the development cycle. In this unit, you learn about typical deployment problems and key deployment elements and concepts.

IBM Training

Unit objectives

- Describe typical deployment problems
- Explain the benefits of a DevOps culture
- Define model-driven deployment
- Define key concepts and terms
- Describe the workflow for a basic project

Unit objectives

IBM Training

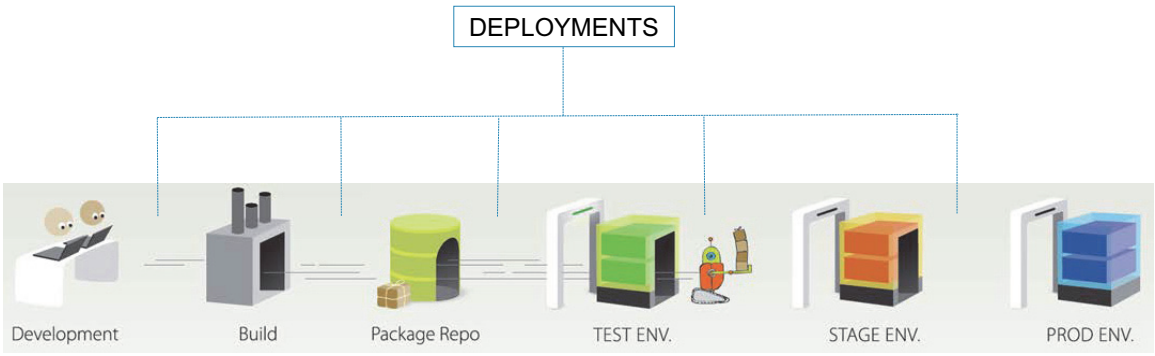
Topics

- Accelerating software delivery and improving quality with IBM DevOps
- Defining key concepts and workflow in IBM UrbanCode Deploy

Topic: Accelerating software delivery and improving quality with IBM DevOps

IBM Training

A deployment is the promotion of an application from one environment to the next



A deployment is the promotion of an application from one environment to the next

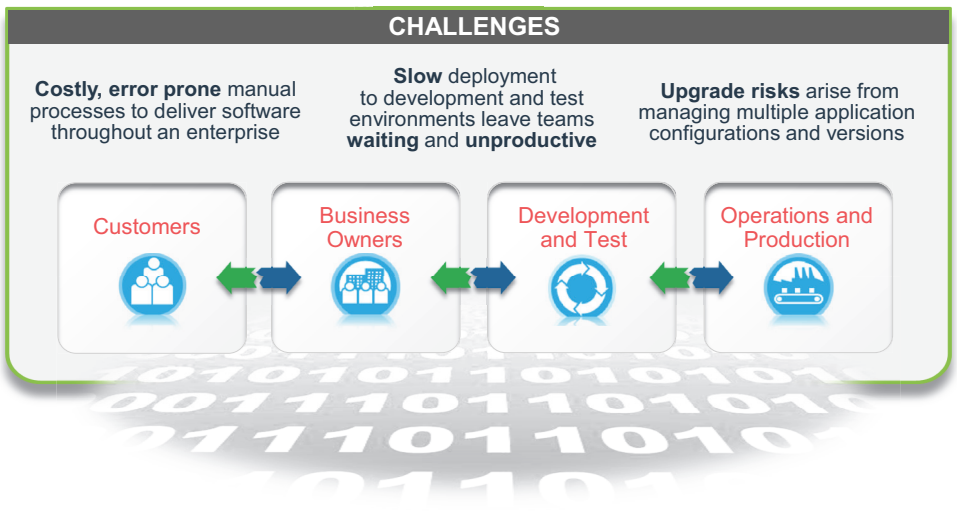
In deployment promotions, environments build on one another to increase the quality of an application before it reaches its intended user.

Applications do not need to go through a set number of environments, but these are typical:

- **Development (DEV)**: In the Development environment, developers build and deploy code in a test lab, and the development team tests the application at the most basic level. When the application meets certain criteria for advancement, it moves to the next environment.
- **System Integration Testing (SIT)**: In the System Integration Testing environment, the application is tested to ensure that it works with existing applications and systems. When the application meets the criteria of this environment, it's deployed to the next environment.
- **User Acceptance Testing (UAT)**: In the User Acceptance Testing environment, the application is tested to ensure that it provides the required features for users. This environment usually is production-like. When the application passes these requirements, it's promoted to the final environment.
- **Production (PROD)**: In the Production environment, the application is made available to users. Feedback is captured by monitoring the application's availability and functionality. Any updates or patches are introduced.

IBM Training

Deployment problems are serious business



Deployment problems are serious business

Some typical deployment challenges include these examples:

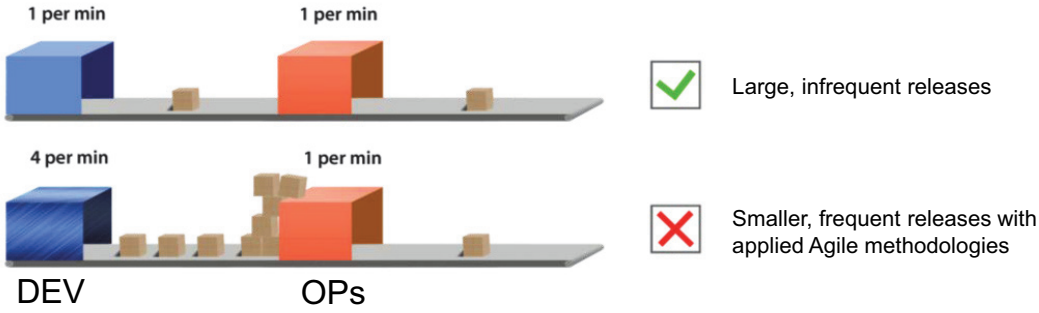
- Differences in development and operations environments and procedures cause failures
- Manual processes for deployments are error prone and lack repeatability and speed
- Major releases take days; hundreds of people are organized by spreadsheet
- No process that alerts team members to errors
- Status can only be obtained through many different systems

Deployment problems can cause an organization both financial loss and damage to its reputation. A few examples include the following cases:

- Knight Capital loses $440 million because of a misconfigured release (in only 45 minutes).
- New Zealand's biggest phone company, Telecom, paid $2.7 million to approximately 47,000 customers who were overcharged after a software glitch.
- A bad software upgrade at RBS Bank left millions unable to access money for 4 days.

IBM Training

Agile development highlights bottlenecks



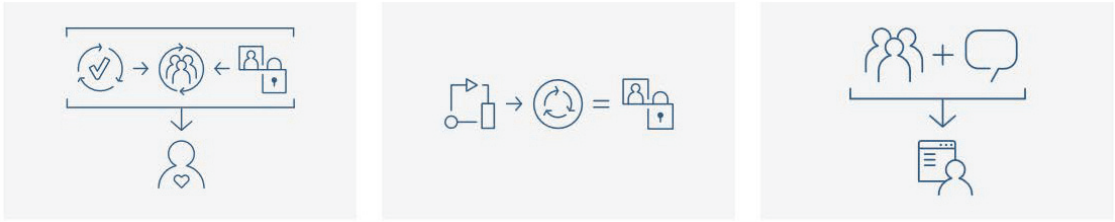
Agile development highlights bottlenecks

Agile methodologies embraced *continuous integration*, a practice in which software developers frequently integrate their work with that of other members of the development team. This process helped development teams to produce smaller releases faster, getting new features developed quickly, but operations teams did not have enough time to ensure stability through quality control and testing, which caused a development backlog.

Test and operations teams had increased pressures to keep up with the increased loads but continued to use traditional tools.

IBM Training

IBM DevOps promotes continuous delivery of software through the entire pipeline



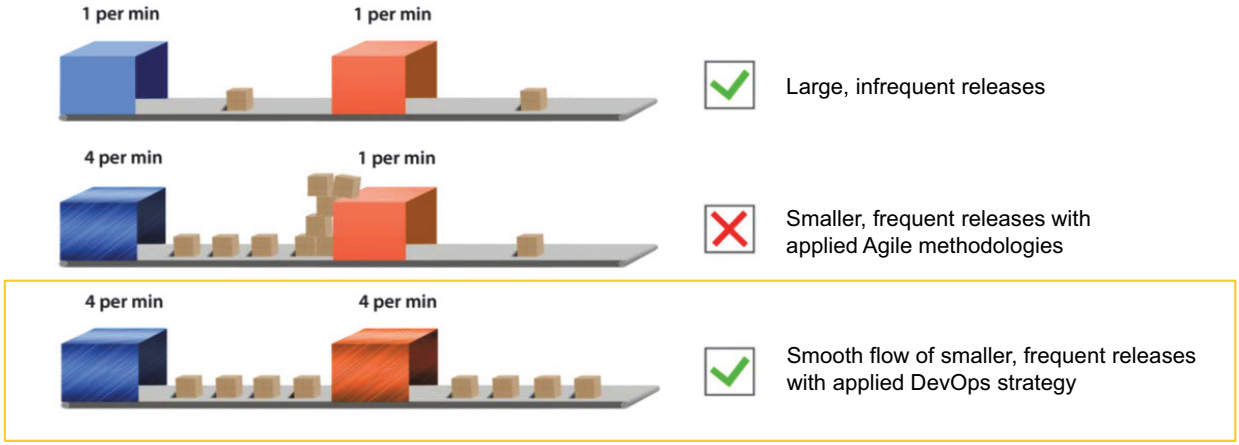
IBM DevOps promotes continuous delivery of software through the entire pipeline

DevOps came into the picture to alleviate some of the pressures between development and operations teams. In DevOps, operations and development engineers participate together in the entire software development lifecycle, from design through the development process to production support. It eliminates the barriers between developing software and production. IBM DevOps supports the philosophy of lean and Agile principles but also focuses on three aspects in software delivery: continuous testing, continuous delivery, and continuous monitoring.

- **Continuous testing** means testing earlier and continuously across the life cycle to achieve continuous feedback on quality. This process is also known as shift-left testing, which stresses integrating development and testing activities to ensure quality is built-in as early in the lifecycle as possible.
- **Continuous delivery** is the process of automating the deployment of the software to all of the environments to achieve consistent results. Using the same automated process with each environment improves efficiency and reduces the risk introduced by inconsistent processes.
- **Continuous monitoring** involves gathering feedback from monitoring data. This data comes from the servers running the application: from development, QA, and production or from metrics tools embedded in the application that capture user actions.

IBM Training

IBM DevOps strategies results in a smooth flow of the delivery pipeline from development to production



IBM DevOps strategies results in a smooth flow of the delivery pipeline from development to production

Applying these DevOps strategies results in the smooth flow of smaller, frequent releases.

IBM Training

IBM DevOps is working for organizations across industries

Computer services

Synchrony Systems cuts development time by up to 90% with IBM Bluemix

Learn more

Watch the video (00:02:06)

Healthcare

Healthcare information provider reduces deployment time from days to minutes

Learn more

Banking

Rizal Commercial Banking Corp. transforms IT to gain 1.2M customers in one year

Learn more

Computer services

DevOps approach helps increase release frequency by 300%

Learn more

Watch the video (00:05:37)

IBM DevOps is working for organizations across industries

DevOps works for a variety of industries. Common themes include cutting deployment time and increasing release frequency. Check out other IBM DevOps case studies at this location:

https://www.ibm.com/ibm/devops/us/en/casestudies/#all

IBM Training

IBM UrbanCode products support the continuous delivery pipeline

IBM UrbanCode products support the continuous delivery pipeline

Your delivery pipeline includes everything from development to production. Deployment is at the heart of the DevOps architecture. IBM UrbanCode Deploy is part of the DevOps story. It is a tool that supports the philosophy and facilitates continuous deployment.

For more information on the IBM UrbanCode Release course, see:

https://www-03.ibm.com/services/learning/ites.wss/zz-en?pageType=course_description&cc=&courseCode=CQ450G

For more information on the IBM UrbanCode Deploy blueprint designer course, see:

https://www-03.ibm.com/services/learning/ites.wss/zz-en?pageType=course_description&cc=&courseCode=CQ401G

IBM Training

IBM UrbanCode Deploy supports the DevOps philosophy

- Automated, consistent deployments and application rollbacks
- Orchestration of changes on servers and tiers
- Configuration and security differences between environments
- Clear visibility: What is deployed where and who changed what
- Integration with middleware, provisioning, and service virtualization

IBM UrbanCode Deploy supports the DevOps philosophy

By using an environment management and deployment tool like IBM UrbanCode Deploy, you can design, deploy, and reuse environments quickly and help accelerate the delivery pipeline. IBM UrbanCode Deploy is an automation tool that manages the software components that get deployed, the middleware components and configurations that need to be updated, the database components that need to be changed, and the configuration changes to the environments to which these components are to be deployed. These tools also capture and automate the processes to carry out these deployments and configuration changes.

IBM Training

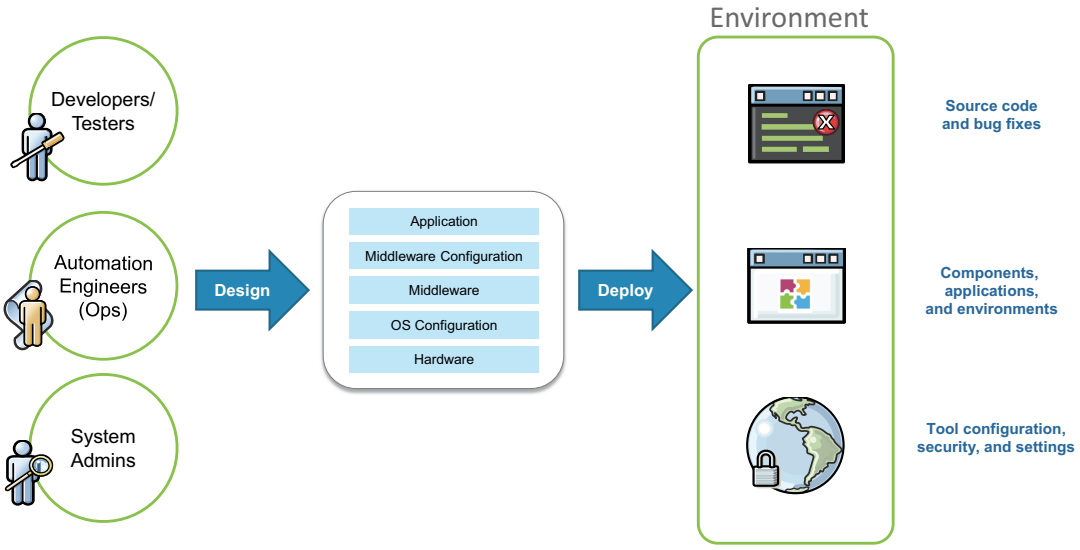
Topics

- Accelerating software delivery and improving quality with IBM DevOps
- Defining the key concepts and workflow in IBM UrbanCode Deploy

Topic: Defining the key concepts and workflow in IBM UrbanCode Deploy

IBM Training

A modular approach permits the division of responsibilities in DevOps organizations



A modular approach permits the division of responsibilities in DevOps organizations

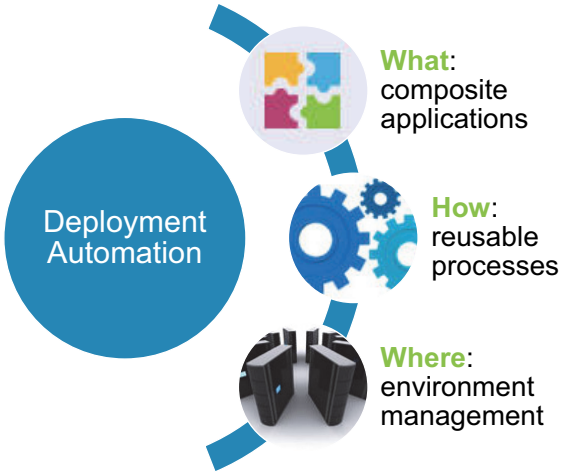
With IBM UrbanCode Deploy, you can separate development and operations responsibilities. Typically, application developers create and deliver code, deploy code to the development environment for unit testing, and fix defects.

Automation engineers author and maintain components and application processes, create templates, define the deployment environments, and promote versions to their next environment (for example, SIT, QA, PERF, PROD).

System administrators set up and configure the tool for use, ensure that the tools are up to date, and set up and maintain the security of the system.

IBM Training

Application models include components, processes, and environments



Model-driven deployment

Application models include components, processes, and environments

IBM UrbanCode Deploy focuses on model-driven deployment. You model a plan to run software deployments, and then you reuse these processes to create automation.

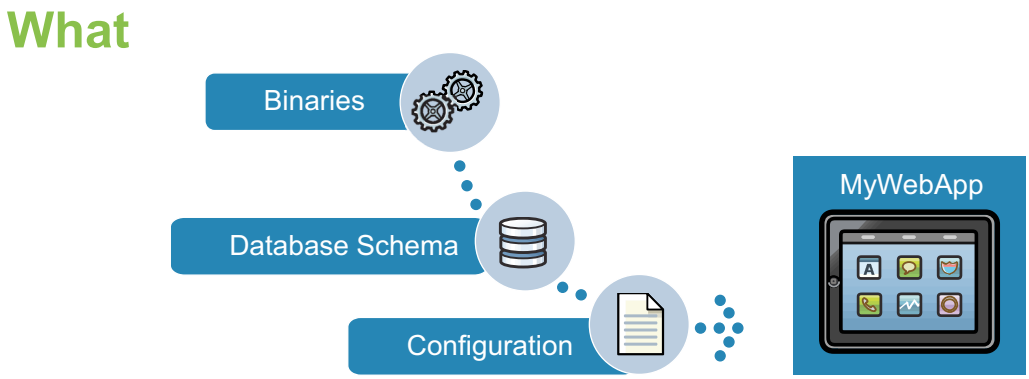
An application brings together the pieces of your deployment and includes three different parts:

1. It houses the components that you want to deploy.
2. It contains the application process that conducts deployment flow.
3. It contains information about the environment where you want to deploy.

This application model answers the questions of what gets deployed (the components), how it gets deployed (the processes), and where it gets deployed (the environments).

IBM Training

A component is the smallest functional unit of an application

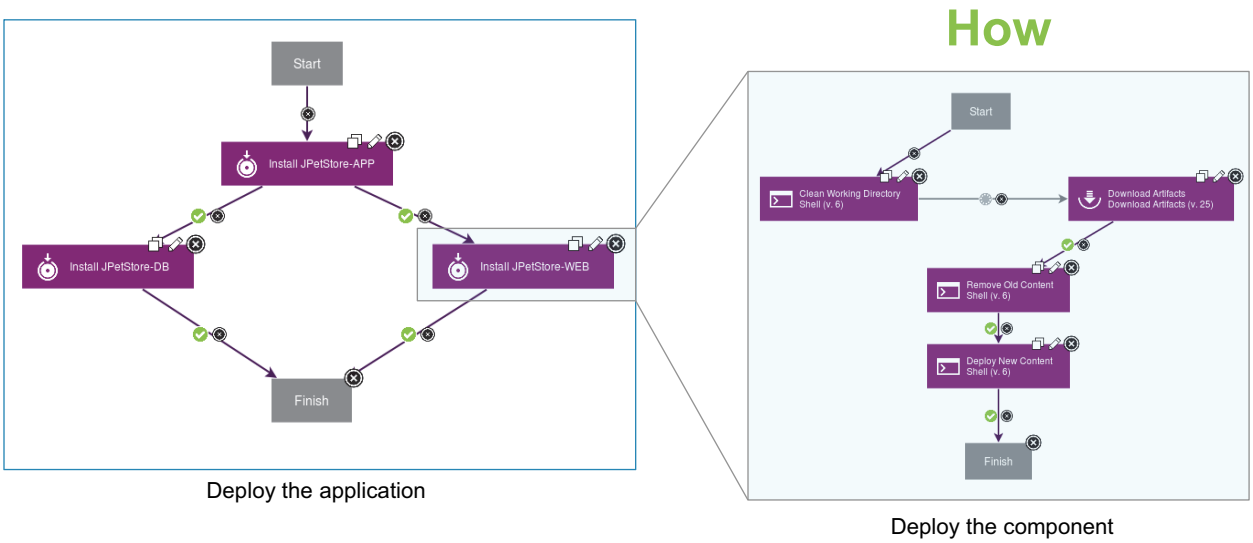


A component is the smallest functional unit of an application

Components are containers for deployable artifacts and identify the “what” of a deployment. This includes any files, images, and databases that are associated with a software project.

IBM Training

Processes are lists of steps and connections between those steps



Deploy the application

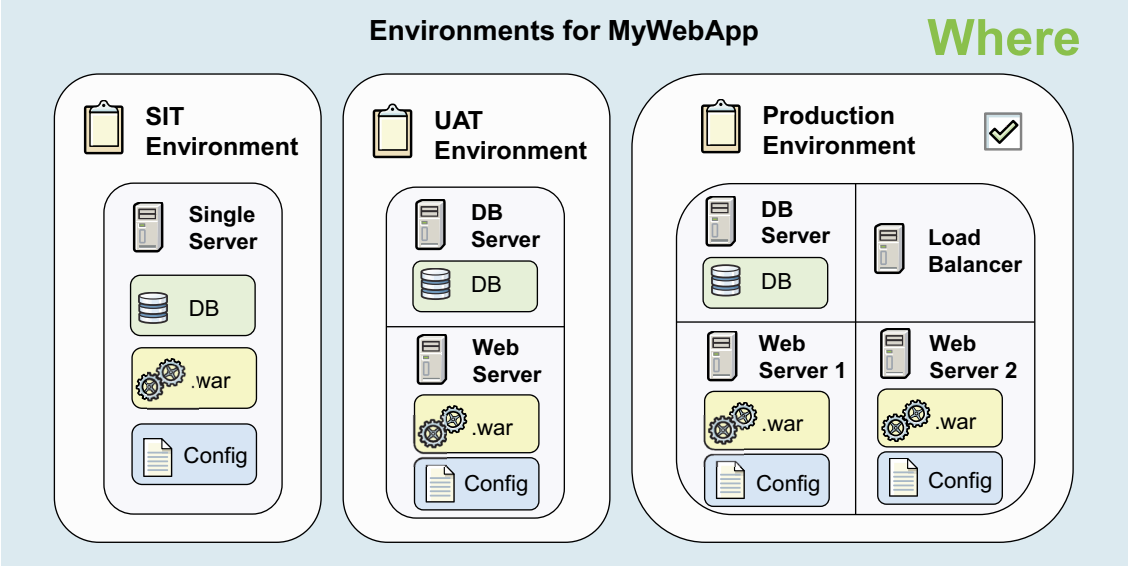
Deploy the component

Processes are lists of steps and connections between those steps

Processes describe the “how” of a deployment and perform actions on components. The component process includes instructions, or steps, that operate on a component. The application process determines what components are deployed together and in what order.

IBM Training

An environment is a collection of resources that hosts an application

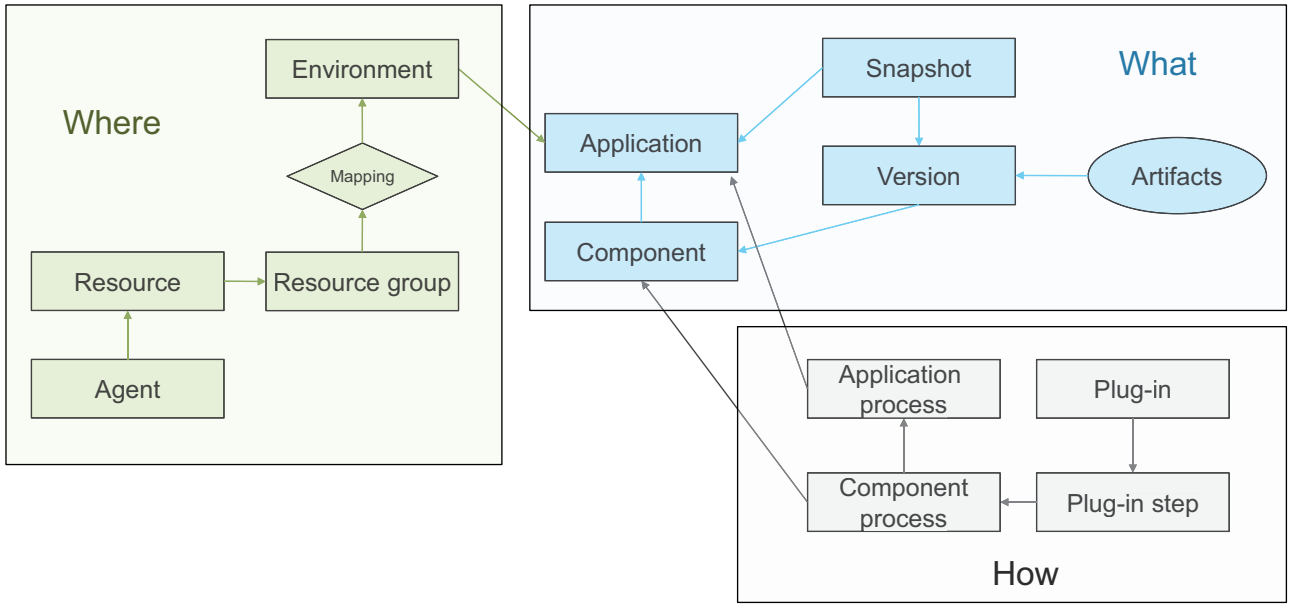


An environment is a collection of resources that hosts an application

The “where” part of the model refers to the target's hosts and environments. You can define where components are deployed and capture configuration settings for each deployment environment for an application.

IBM Training

The objects for modeling software deployments work together



The objects for modeling software deployments work together

The objects for modeling deployments fit into these categories:

What

- **Application**: A logical grouping of artifacts with self-contained definitions of automation and target environment
- **Component**: Files or configuration to be deployed, including the steps to deploy them
- **Version**: A container with the actual deployment artifacts, which are typically produced by an automated build
- **Snapshot**: A group of component versions that guarantees repeatability and reduces user error

How

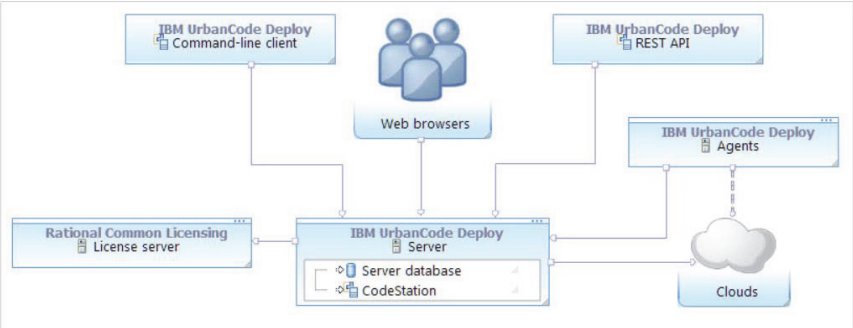
- **Plug-in**: Provides parameterized steps to perform automation; provided by IBM UrbanCode Deploy or through integrations
- **Component process**: Composes individual plug-in steps to define an automated process for a component
- **Application process**: Composes individual steps to define an automated process for a component

Where

- **Agent**: Represents the IBM UrbanCode Deploy agent software that is running on a server and is installed on each server where commands must be executed
- **Resource**: Represents a deployment target, such as a physical server, virtual machine, or database
- **Resource group**: Represents a logical collection of resources that enable collections of resources to be easily reused
- **Environment**: Combines resources into a single logical deployment target

IBM Training

The core installation of IBM UrbanCode Deploy includes a server, agents, and a license server

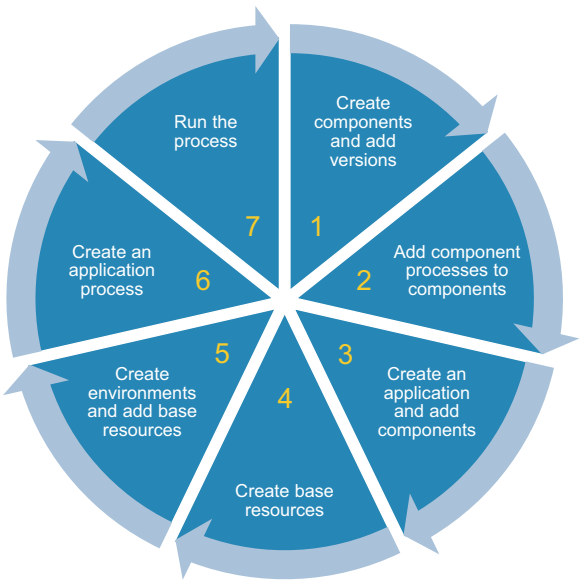


The core installation of IBM UrbanCode Deploy includes a server, agents, and a license server

In this course, you access the server through web browsers, but you can also use REST API or the command-line client. Agents can be installed on cloud environments, virtual machines, containers, or physical systems.

IBM Training

A basic workflow consists of some typical steps



A basic workflow consists of some typical steps

Modeling software deployment in IBM UrbanCode Deploy includes configuring components and component processes and then adding those components to applications. Then you use processes to deploy the components to environments.

A simple workflow might start with creating components and the component processes. Next, you might add those components to an application and create resources and environments. Finally, you create an application process and then run that deployment.

While this slide demonstrates a basic workflow, you can create the pieces of your project in a different order. For instance, you could create your base resources first.

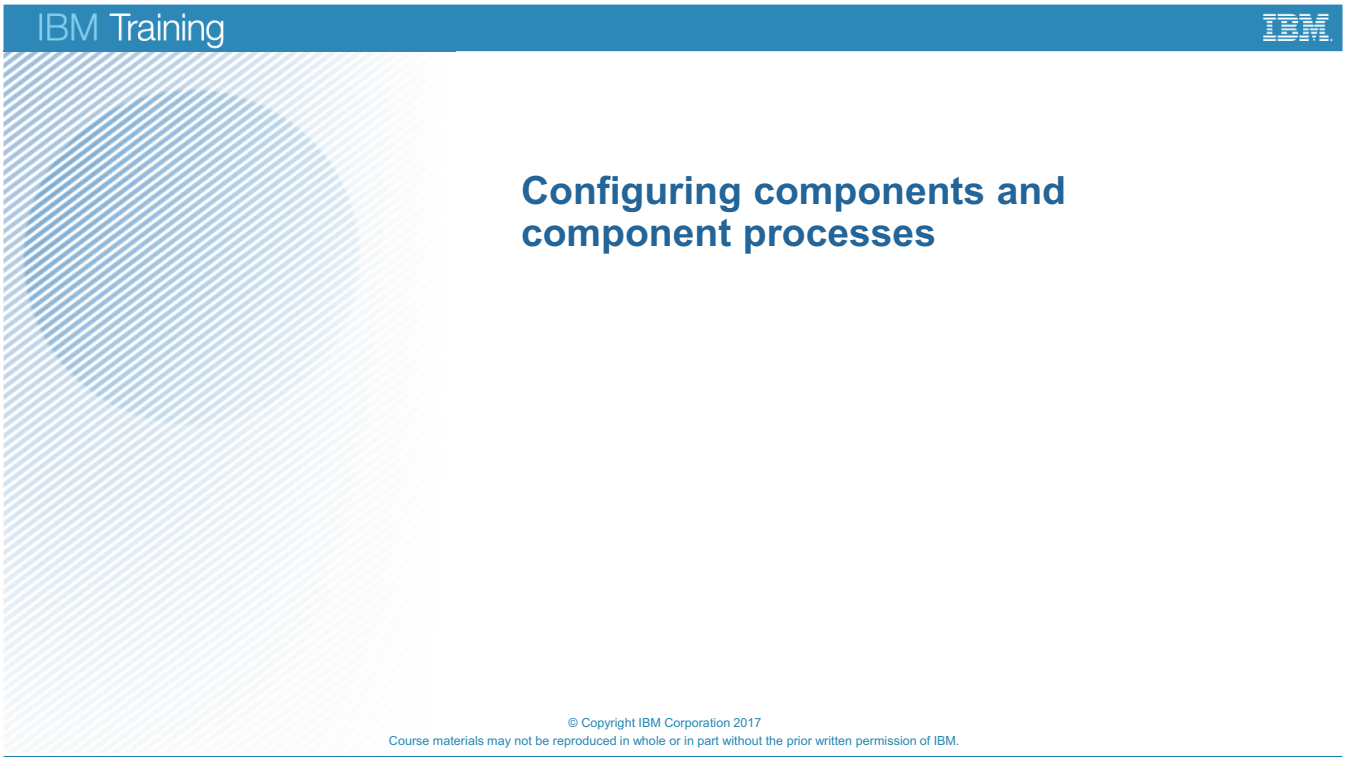
IBM Training

Unit summary

- This unit covered typical deployment problems, the benefits of a DevOps culture, and key concepts and terms in IBM UrbanCode Deploy
- A deployment is the promotion of an application from one environment to the next
- IBM UrbanCode Deploy supports the principles and practices of DevOps
- The goals of DevOps are continuous testing, continuous delivery, and continuous monitoring
- IBM UrbanCode Deploy focuses on model-driven deployment
- Application objects fit into the *what*, *where*, and *how* of the deployment model

Unit summary

Unit 2 Configuring components and component processes



Components represent deployable items along with user-defined processes that operate on them, usually by deploying them. In this unit, you learn how to create a component, import component versions, create component processes, and work with properties.

IBM Training

Unit objectives

- Describe the purpose and structure of a component
- Create components and import component artifacts
- Create component processes to deploy components
- Store component information with properties

Unit objectives

IBM Training

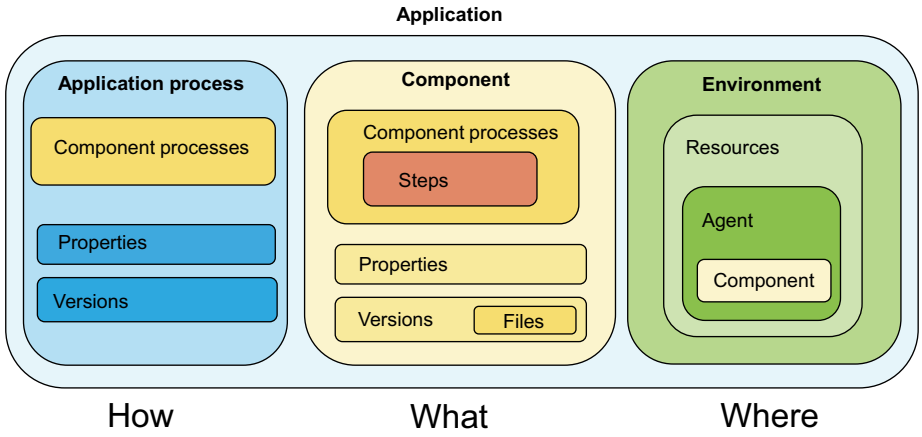
Topics

- Components overview
- Creating components and importing component artifacts
- Creating processes to deploy components
- Storing information with properties

Topic: Component overview

IBM Training

A component is the smallest functional unit of an application

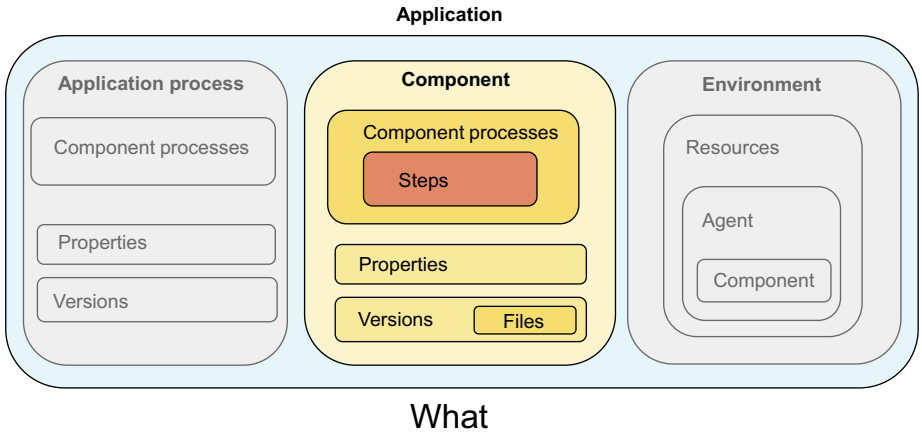


A component is the smallest functional unit of an application

You will see this representation of the objects in IBM UrbanCode Deploy for the next two units. This slide shows a visual that groups the parts of the product that make up an application.

IBM Training

Components describe *what* gets deployed

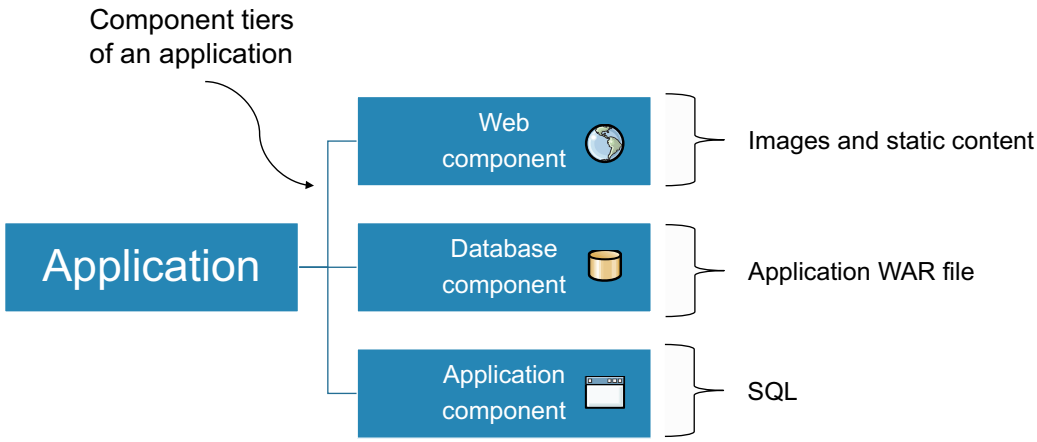


Components describe what gets deployed

This section focuses on ***components***, which are containers for deployable items and their deployment instructions. A component usually contains a closely related set of artifacts, and you deploy its content on a single target.

IBM Training

Each component represents artifacts from a single source



Each component represents artifacts from a single source

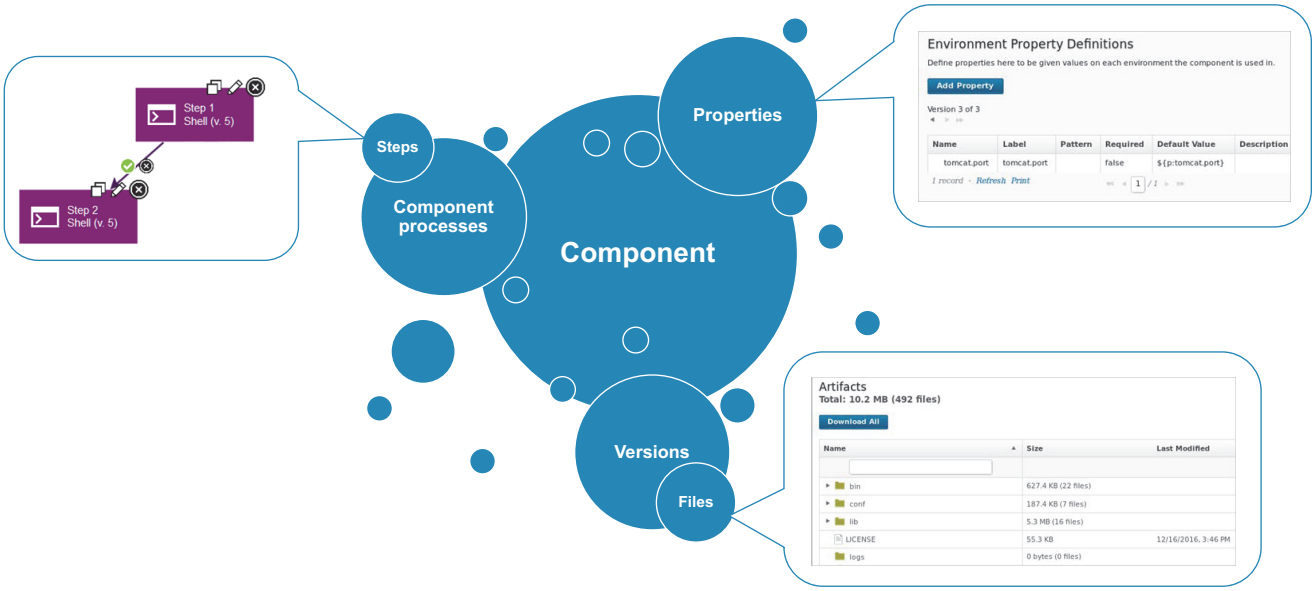
An application consists of a break down of components. In this example, the application has three component tiers:

- One for the web content
- One for the database content
- One for the application content

This arrangement gives flexibility to deployments. You can see what versions of each component are deployed to specific environments. The benefit of IBM UrbanCode Deploy is that it manages the source file and identifies which files have changes. Then it deploys only the changed files.

IBM Training

A component contains versioned files, component processes, and properties



A component contains versioned files, component processes, and properties

From a high-level view, a component has three major parts: versions, component processes, and properties.

- A **version** contains a set of artifacts. Versions ensure that the appropriate set of files is deployed.
- A **component process** is a list of commands that run on a target computer. Steps can manipulate files, run system commands, download files, and run programs.
- **Properties** are variables that store the locations of files, folders, servers, or other resources that might change.

IBM Training

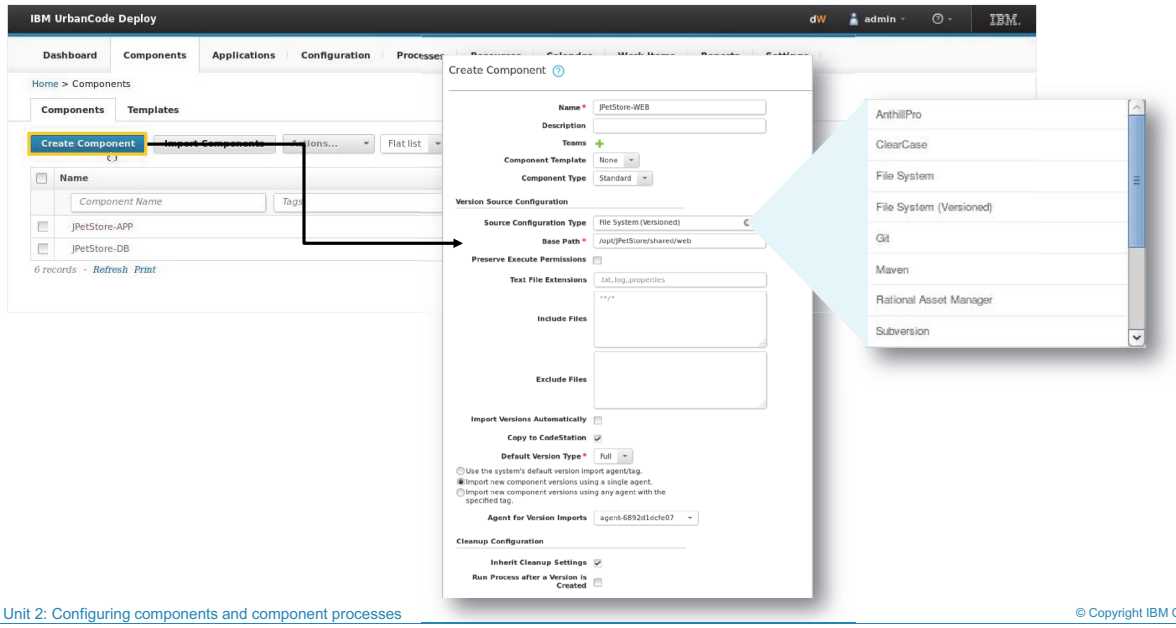
Topics

- Components overview
- Creating components and importing component artifacts
- Creating processes to deploy components
- Storing information with properties

Topic: Creating components and importing component artifacts

IBM Training

Component artifacts can come from various source-code management sources



Component artifacts can come from various source-code management sources

When you create a component, you identify its type of artifacts and the location of the artifacts. In most cases, you add components by connecting the server to the system that hosts the artifacts, such as a Maven repository. In this example, and when you work through the lab, the versions are imported from the file system.

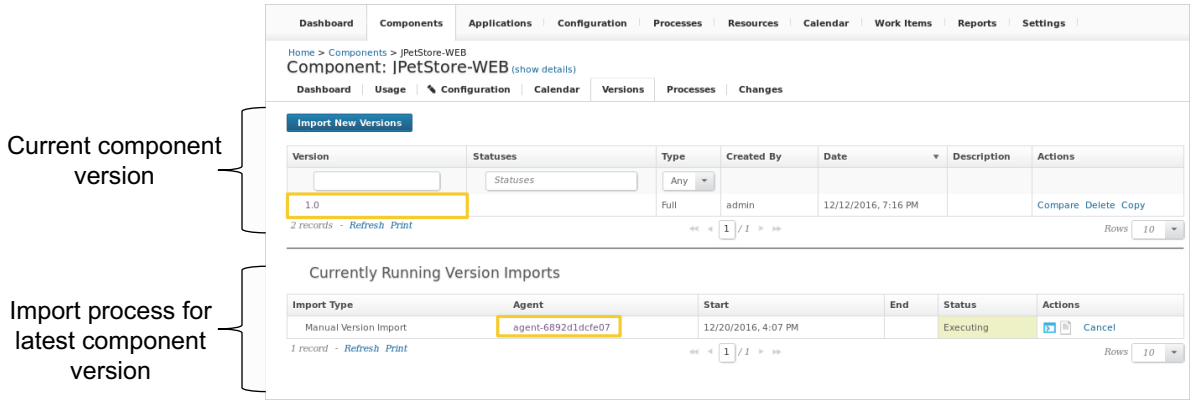
In the Create Component window, you need to consider a few points:

- **Component Template**: First, you decide if you want to base a new component on a template. With templates, you can reuse component processes and properties. You can save time when you create many components with a similar set of attributes. Although this course doesn't cover templates, you can consult the video demonstration in the notes section of this slide for more information. For more information about creating component templates, see this video: https://developer.ibm.com/urbancode/videos/component-templates-ibm-urbancode-deploy-v6-0/
- **Source Configuration Type**: The Source Configuration Type identifies the location of the artifacts. You can choose from many different source control systems.
- **Import Versions Automatically**: If you select the option to import versions automatically, the source location is periodically polled for new versions. Any versions that are found are automatically imported.
- **Run Process after a Version is Created**: In conjunction to automatically importing versions, at the bottom of the window, you can choose to run a process after a version is created. You specify an environment and an application process. Then any time a new version is automatically imported, the process runs. You can use this to automatically deploy a new version any time that it's created.

- **Copy to CodeStation**: By default, a complete copy of the content is imported into CodeStation, which leaves the artifacts in their original location untouched. CodeStation is the embedded artifact repository in IBM UrbanCode Deploy. It tracks file versions and maintains a complete history for all components.
- **Default Versions Type**: When you import a version, it can be full or incremental. You make this selection in the **Default Versions Type** field before you even import the component. A full version contains all component artifacts. An incremental version contains only artifacts that are different from the previous version. It's additive, like a patch.

IBM Training

Components have versions that ensure the appropriate component instances are deployed



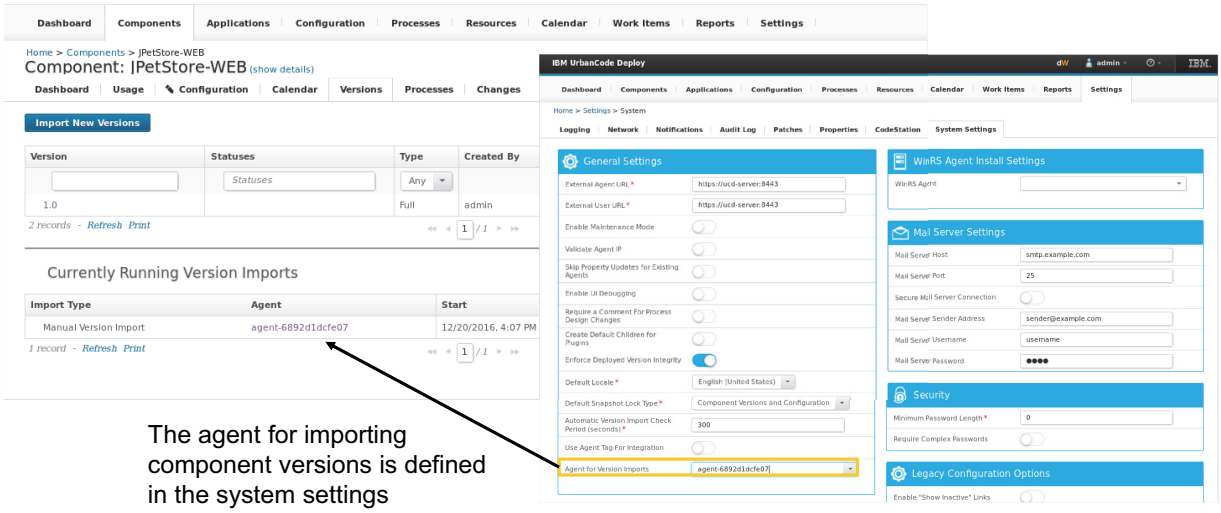
Components have versions that ensure the appropriate component instances are deployed

After you define the source for a component, you import its artifacts on the **Versions** tab. Each time a component is imported, including the first time, a new version of the component is created. This process is called ***versioning***.

To manually import a new version, you click **Import New Versions**. The status of the import process displays in the Currently Running Version Imports section. You can see the agent that is importing the component.

IBM Training

The agent is typically on the same computer where the artifacts are located



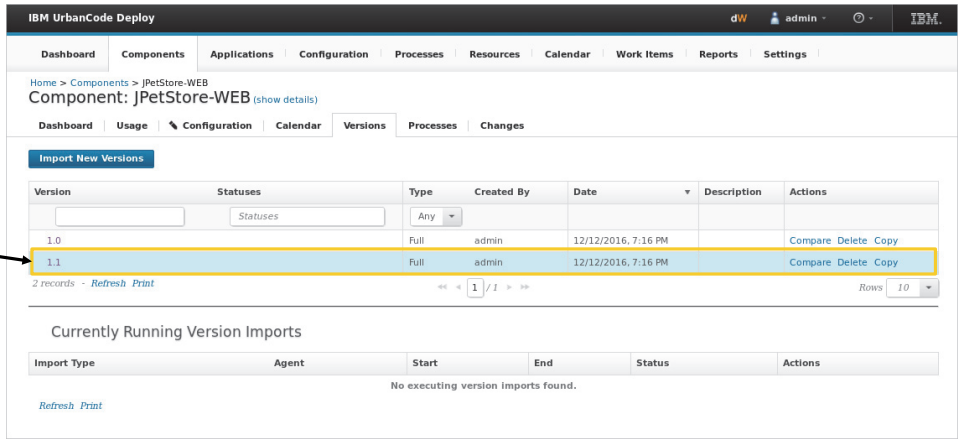
The agent is typically on the same computer where the artifacts are located

Version artifacts are imported by the agent on that system. The agent copies the files from a repository or file system into CodeStation. You can choose the agent that imports versions in the system settings.

IBM Training

A new version of the component is created every time a component's artifacts are modified and imported

New component version successfully imported

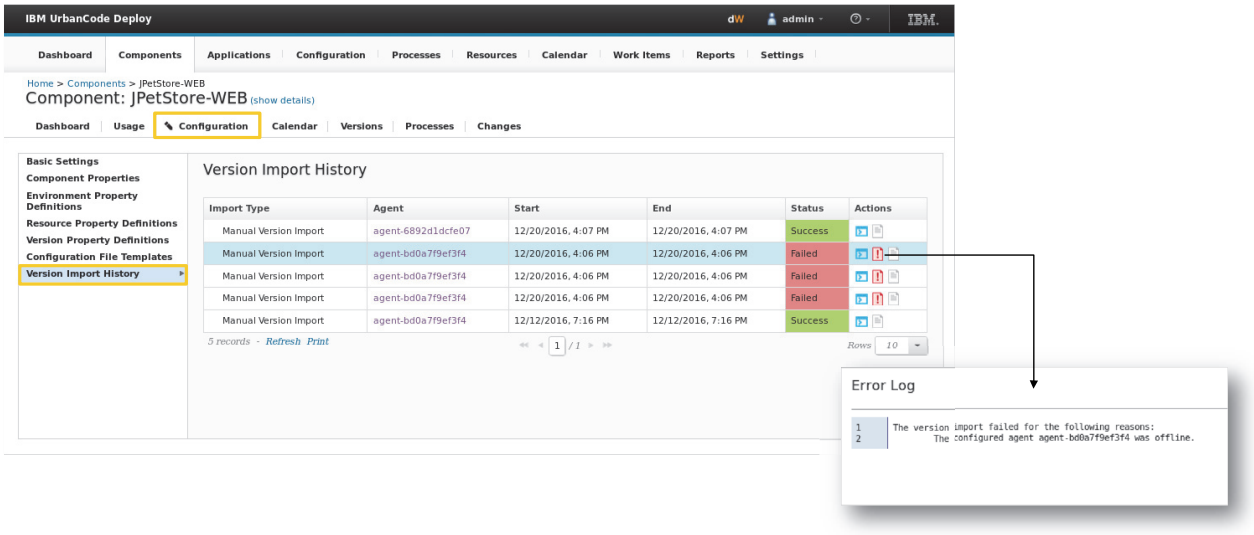


A new version of the component is created every time a component's artifacts are modified and imported

A component might have several versions in CodeStation, and each version has a unique set of files.

IBM Training

The version import history shows the status of attempted imports

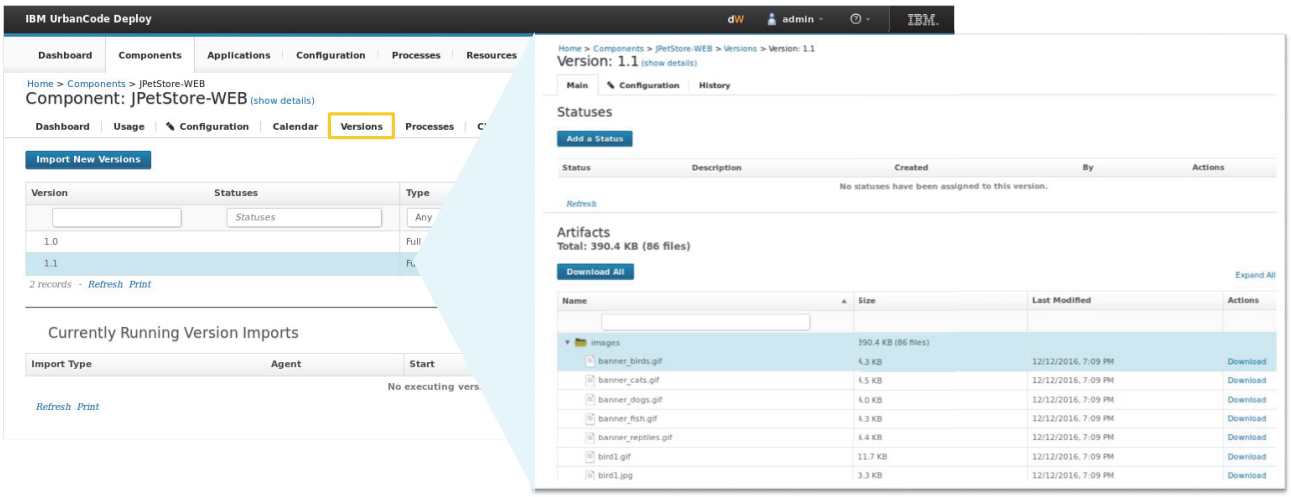


The version import history shows the status of attempted imports

You can see the version import history in the **Configuration** tab. If the version import failed, you can view the error log. Make sure that the files are available on the agent system and that the location of these files is correct on the component.

IBM Training

A version contains a set of deployable artifacts

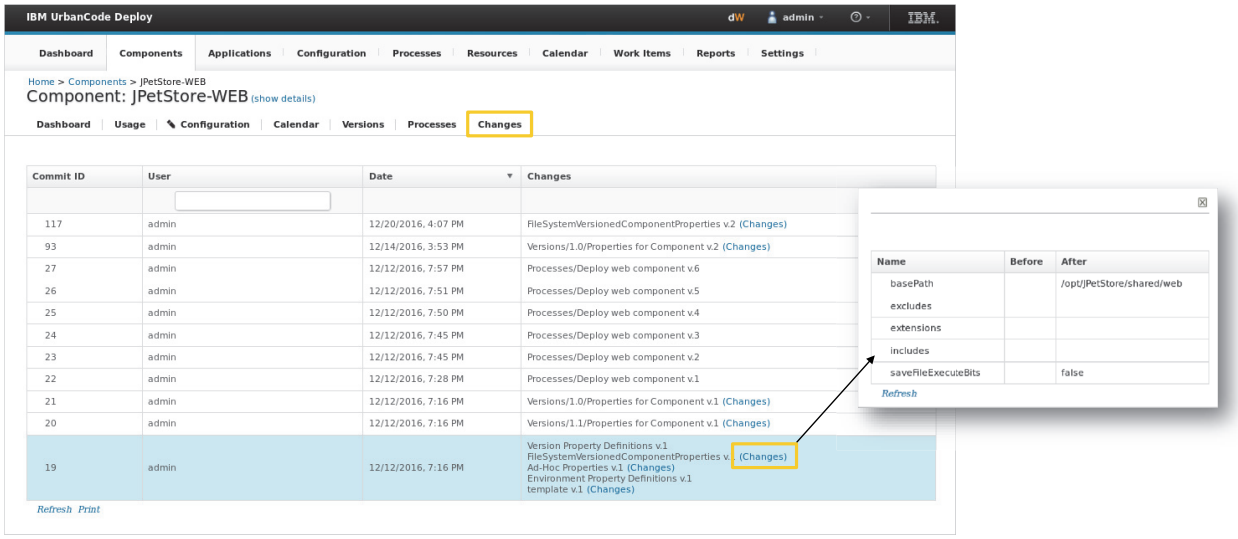


A version contains a set of deployable artifacts

After a version is imported successfully, click the new version to view the files that were imported. From **Versions**, you see all files that are associated with a component. In this example, the images for the web server are listed in the Artifacts section.

IBM Training

Change logs provide information about modifications to components



Change logs provide information about modifications to components

The **Changes** tab tracks changes made to the component. You can see who modified the component and the date and time the change was made. This information provides you with an audit trail of who is changing things. In this example, you can see that the basePath property was changed.

IBM Training

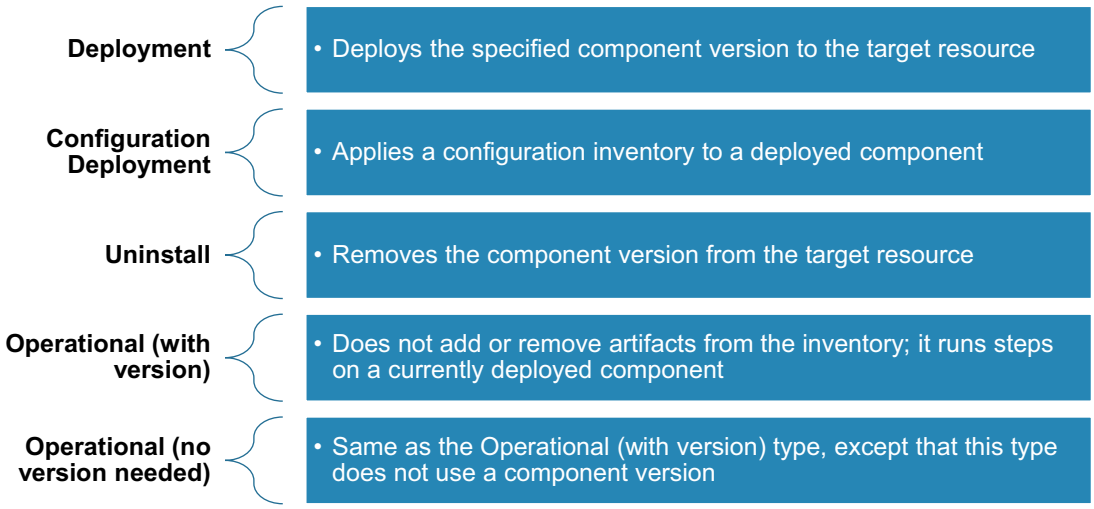
Topics

- Components overview
- Creating components and importing component artifacts
- Creating processes to deploy components
- Storing information with properties

Topic: Creating processes to deploy components

IBM Training

Component processes deploy, configure, or uninstall component versions

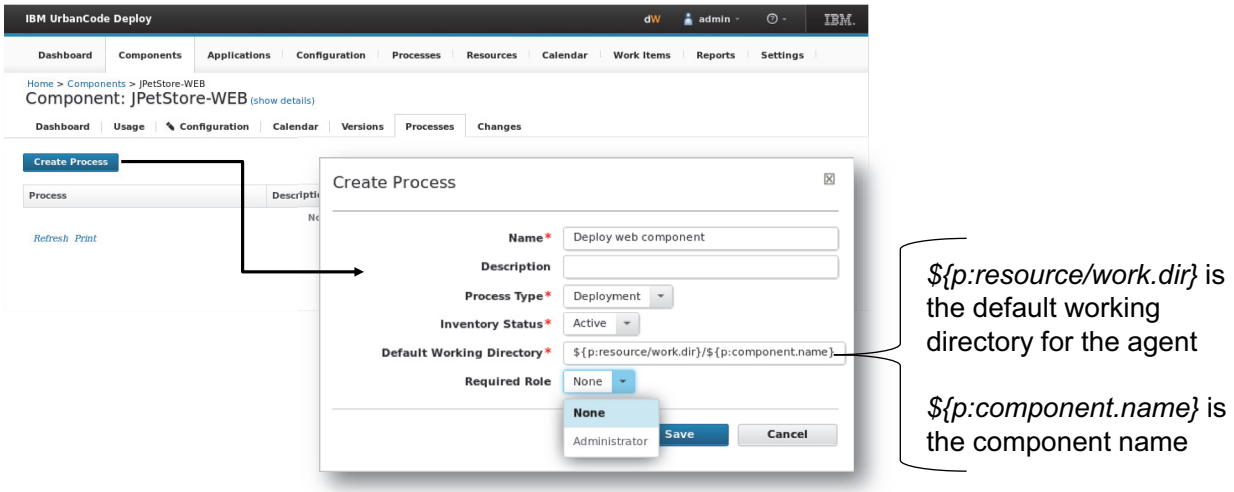


Component processes deploy, configure, or uninstall component versions

At this point, you created your components and imported the versions. Now you see how the actual deployment happens. You deploy components with component processes. Processes define what IBM UrbanCode Deploy does with a component's artifacts. For example, a component process can run tasks on a single component, such as deploying it, uninstalling it, or running configuration tasks on it.

IBM Training

Components contain processes that perform operations on component files



Components contain processes that perform operations on component files

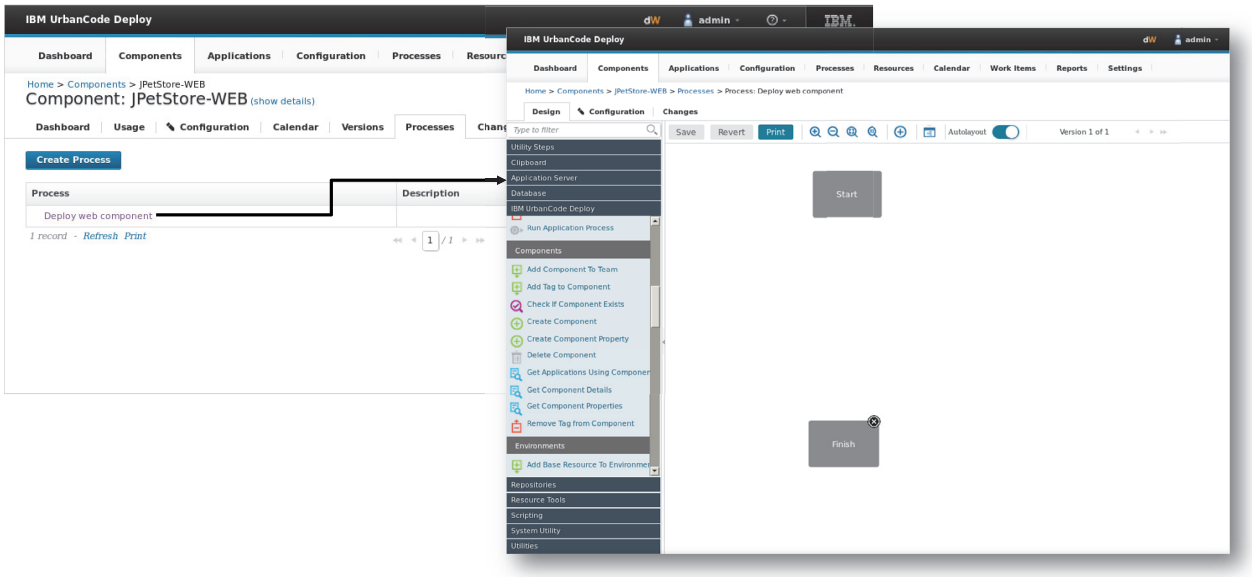
You create a component process in two steps: first, you configure basic information, such as the name and process type; then you use the process editor to assemble the process. A component process is defined for a specific component.

When you create a process, you need to make a few settings:

- **Process type**: The process type determines how the component handles updating the inventory in IBM UrbanCode Deploy. Here you decide whether your process will deploy components or configuration or if it will uninstall a component.
- **Inventory status**: The Inventory status is a status that is applied to a component version after the processes successfully runs. An Active status indicates that the component version is deployed to its target resource.
- **Default working directory**: The default working directory defines the location that the agent uses to run the process. You can see the default value in the example, where ${p:resource/work.dir} is the default working directory for the agent and ${p:component.name} is the component name. The default value resolves to agent_directory\work\component_name_directory and is stored in the componentProcess.defaultWorkDir property.
- **Required role**: You can set a value in the **Required Role** field to restrict access to a specific role of users who can run this process. The available options are derived from the IBM UrbanCode Deploy security system. The default value is **None**, meaning that anyone can run the process.

IBM Training

Component processes are created with the process editor

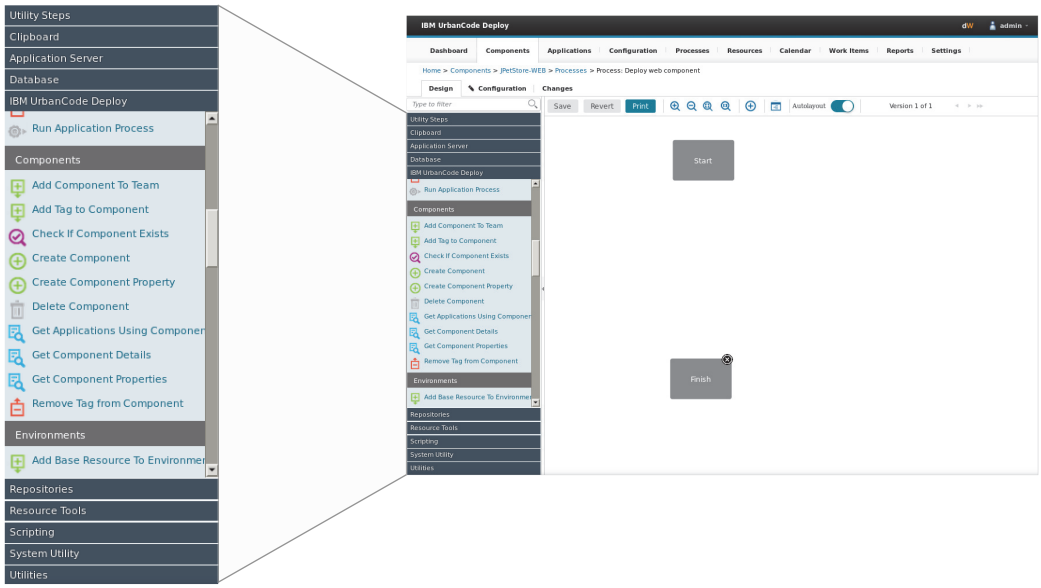


Component processes are created with the process editor

Component processes are created with the process editor. To open the process editor, click the process name. The process editor has a **Steps** palette on the left and a design space on the right. A process begins with the Start step and must end with the Finish step.

IBM Training

Process steps are selected from a menu of standard steps that replace typical deployment scripts and manual processes



Process steps are selected from a menu of standard steps that replace typical deployment scripts and manual processes

The available steps in the palette are determined by the installed plug-ins. Plug-ins consist of distinct processes called steps. Each step consists of these items:

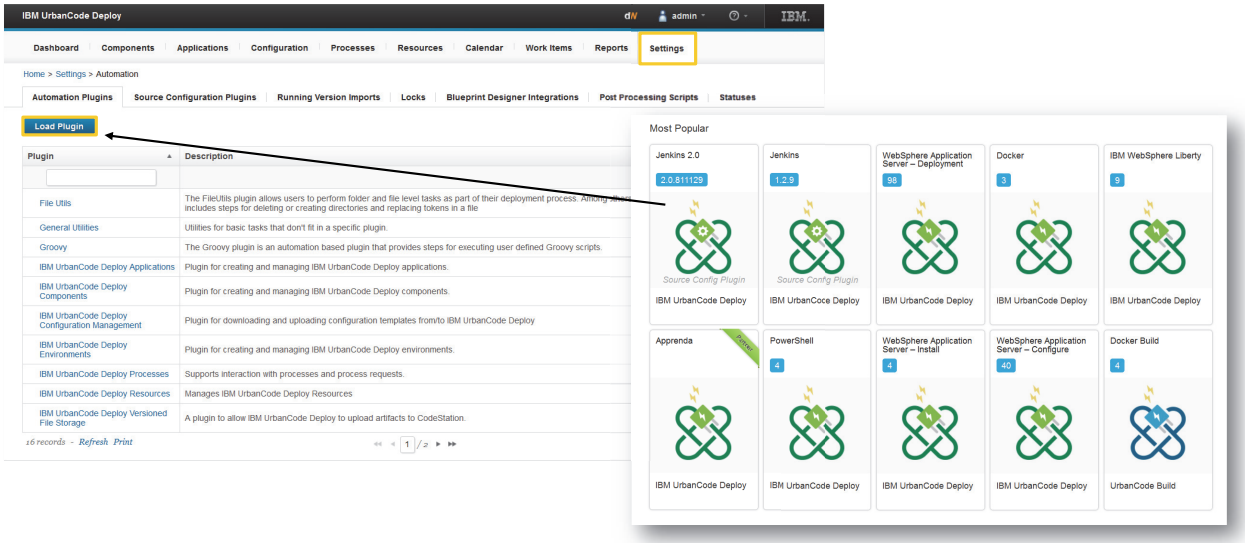
- Properties
- A command that runs the step
- Post-processing instructions

Step properties can serve various purposes, from providing input to the command to supplying some, or all, the actual command itself.

Some plug-ins are included with the product, but you can install others. IBM UrbanCode Deploy provides steps for several utility processes, such as inventory management, and workflow control.

IBM Training

A number of plug-ins come with the product and many others are available that can be downloaded and installed



A number of plug-ins come with the product and many others are available that can be downloaded and installed

Other process steps are provided by plug-ins that provide integration with common deployment tools and application servers. A full list of plug-ins can be found on DeveloperWorks at this address:

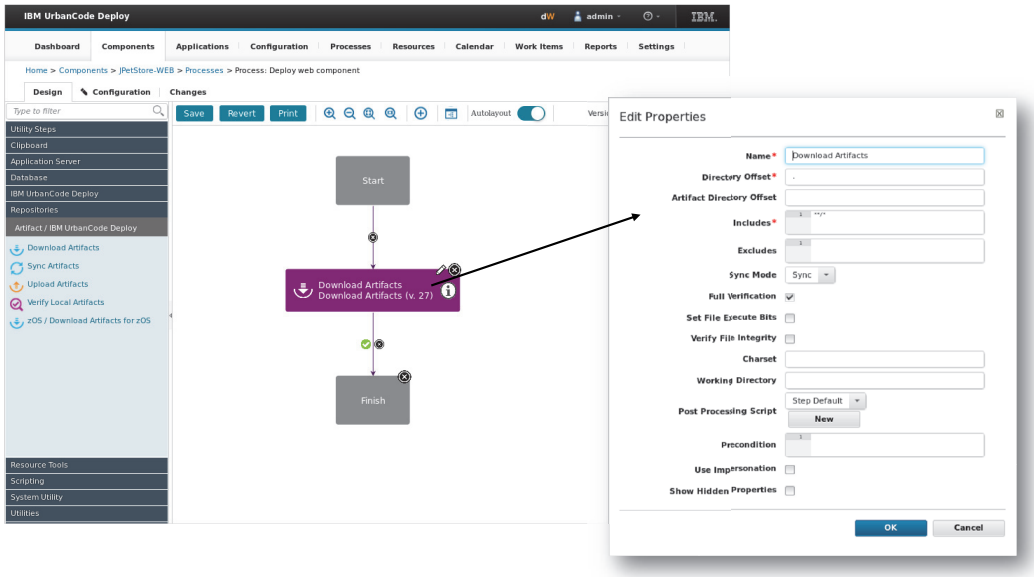
https://developer.ibm.com/urbancode/plugins/

Plug-ins provide processing and integration functions. There are two types of plug-ins:

- **Source-type plug-ins** integrate with external systems to import artifacts and create component versions. Source-type plug-in properties are typically defined when a component is created.
- **Automation-type plug-ins** provide process steps that manipulate components, typically by deploying them. You can set automation-type plug-in properties at design time in the process editor or at run time in the user interface.

IBM Training

The palette provides automation steps that you can drag onto the process editor



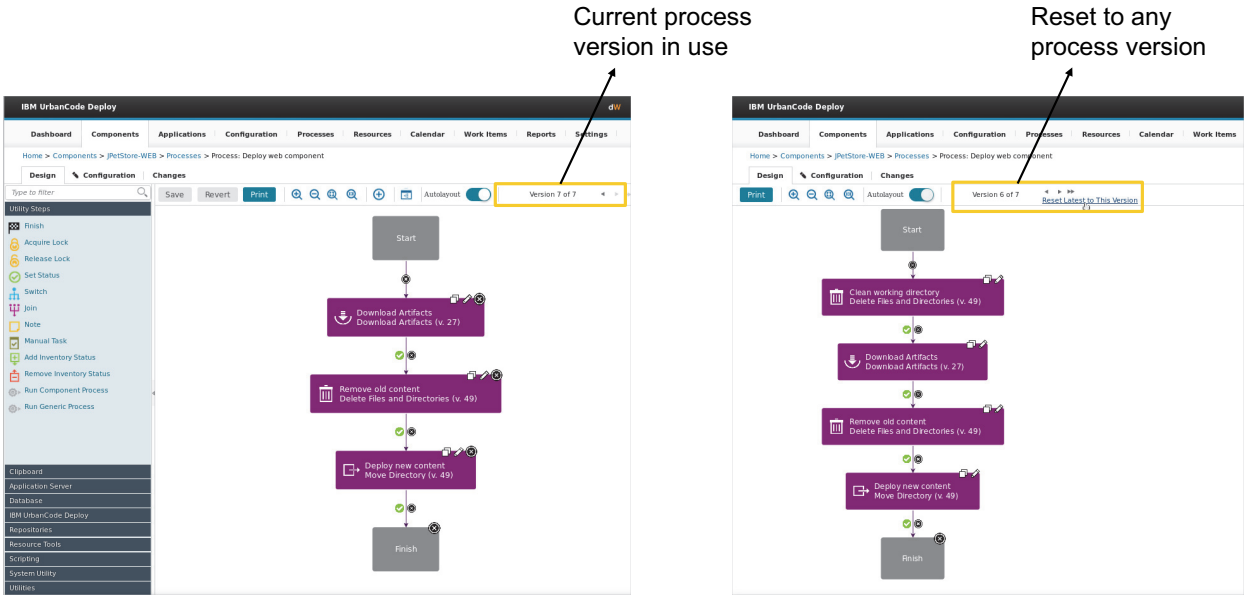
The palette provides automation steps that you can drag onto the process editor

The process editor is a visual drag-and-drop editor where you can drag process steps from the palette onto the design space and configure them as you go. As more steps are placed, you visually define their relationships with one another.

When you drag a step into the process editor, you can edit the fields for that step. In this example, the Download Artifacts step is prepopulated with information that you can use.

IBM Training

Processes are versioned artifacts

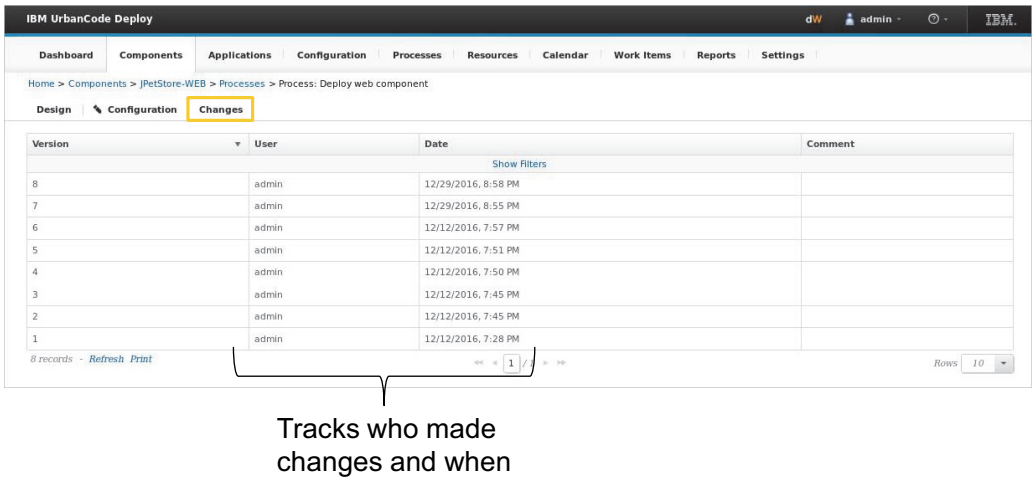


Processes are versioned artifacts

Each time you save a process, a new version of that process is saved. You can scroll through the versions to view and then reset to any process version that you previously saved.

IBM Training

The Changes tab tracks the creation date and author of the versions

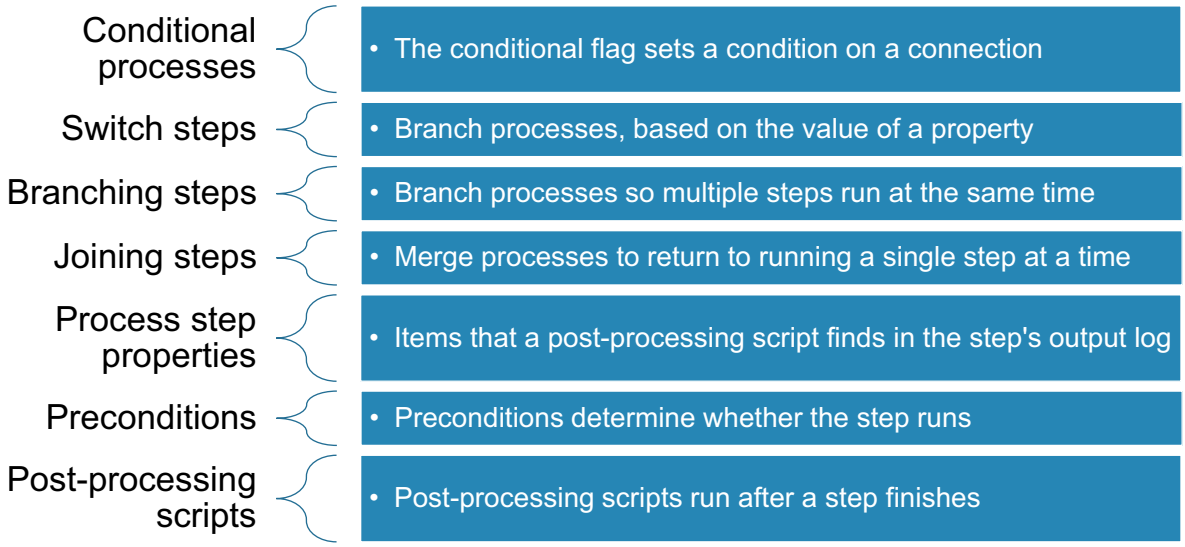


The Changes tab tracks the creation date and author of the versions

When a process is created, information is gathered to inform when a change was made and who made it.

IBM Training

Processes use different kinds of steps and joining techniques

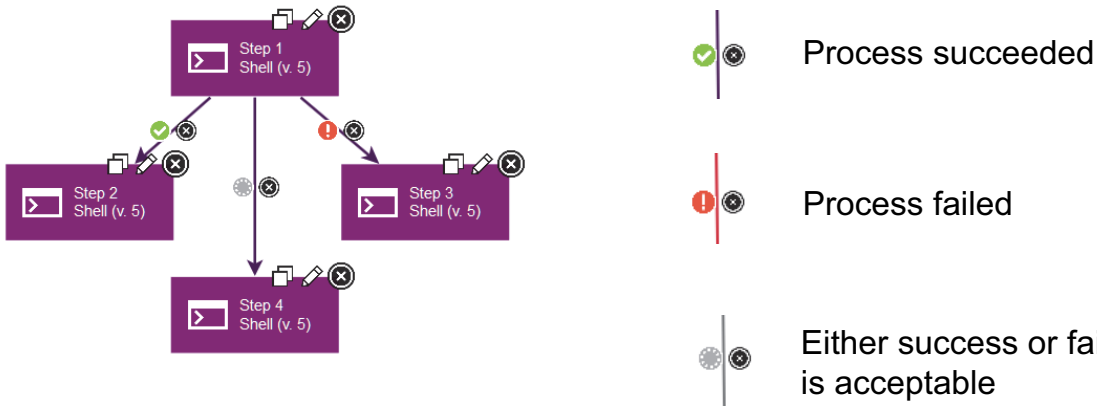


Processes use different kinds of steps and joining techniques

A component process can be as simple as a single step or contain numerous relationships, branches, and process switches. When you design processes, you use different kinds of process steps and joining techniques.

IBM Training

Conditional processes use flags to set conditions on connections



Conditional processes use flags to set conditions on connections

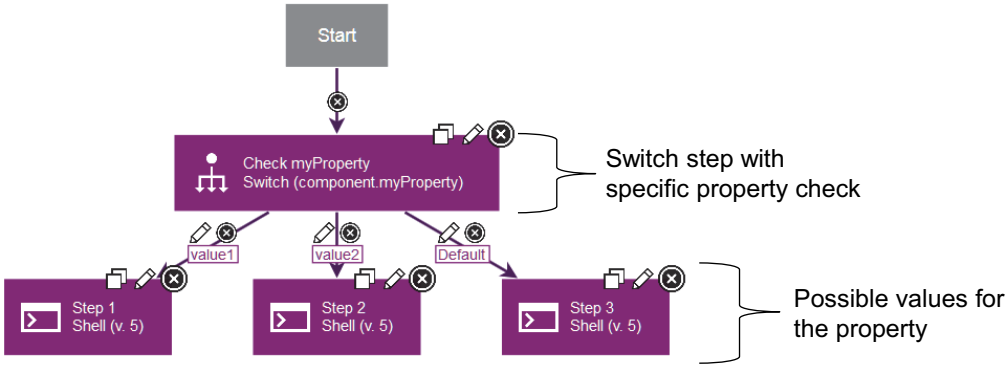
By default, all connections between steps have the flag set to **Success**, which is the green circle with a check. This setting means that the originating step must finish processing before the next step starts to process. If you set the flag to **Fail**, which is the red circle with an explanation point, then the step must have a failed process to reach the next step in that branch. And if you set the flag to a gray circle, the process moves to the next step, whether the previous step's process completed or failed.

The possible conditional flags are as follows:

- **Success**: The process is complete.
- **Fail**: The process did not finish.
- **Both**: The process is complete or the process did not finish. Either status is acceptable, and the next step can start.

IBM Training

Switch steps branch processes based on the value of a property

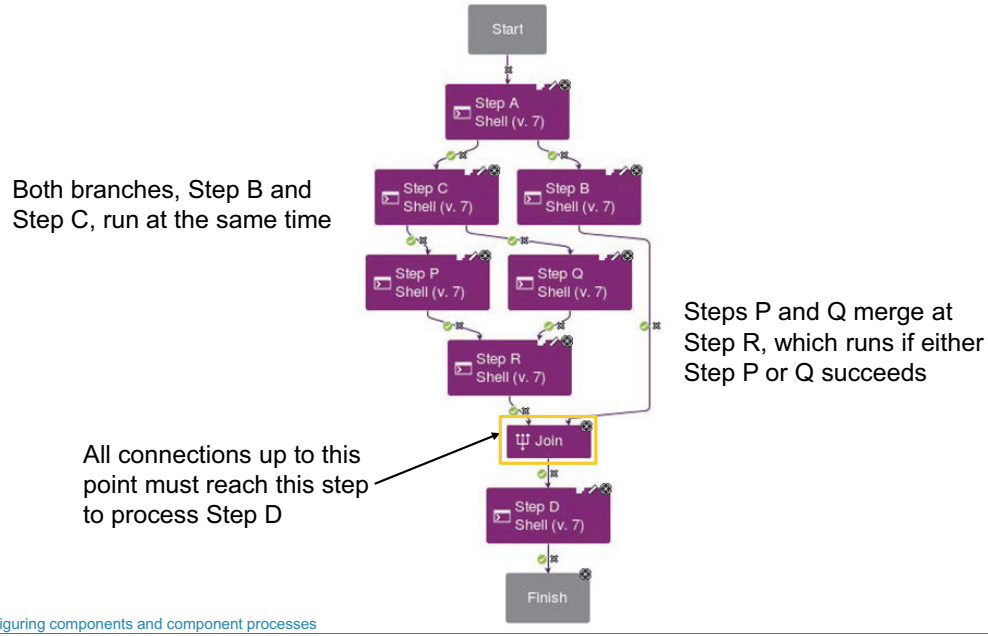


Switch steps branch processes based on the value of a property

When you add a switch step, you specify a property. Then you add two or more outgoing connections from the step, each with a possible value for that property. At run time, the process follows the connection with the matching property value.

IBM Training

Branching steps runs multiple steps at the same time



Branching steps run multiple steps at the same time

You can branch processes so that multiple steps run at the same time by adding multiple outgoing connections from a step. You can use a Join step to merge processes into running a single step at a time. Join steps indicate that all connections to the Join step must succeed. If any of the steps that connect to a Join step fail or do not run, the Join step causes the process to fail. In this example, all steps leading up to Steps R and B must complete, because Steps R and B both connect to a Join step.

IBM Training

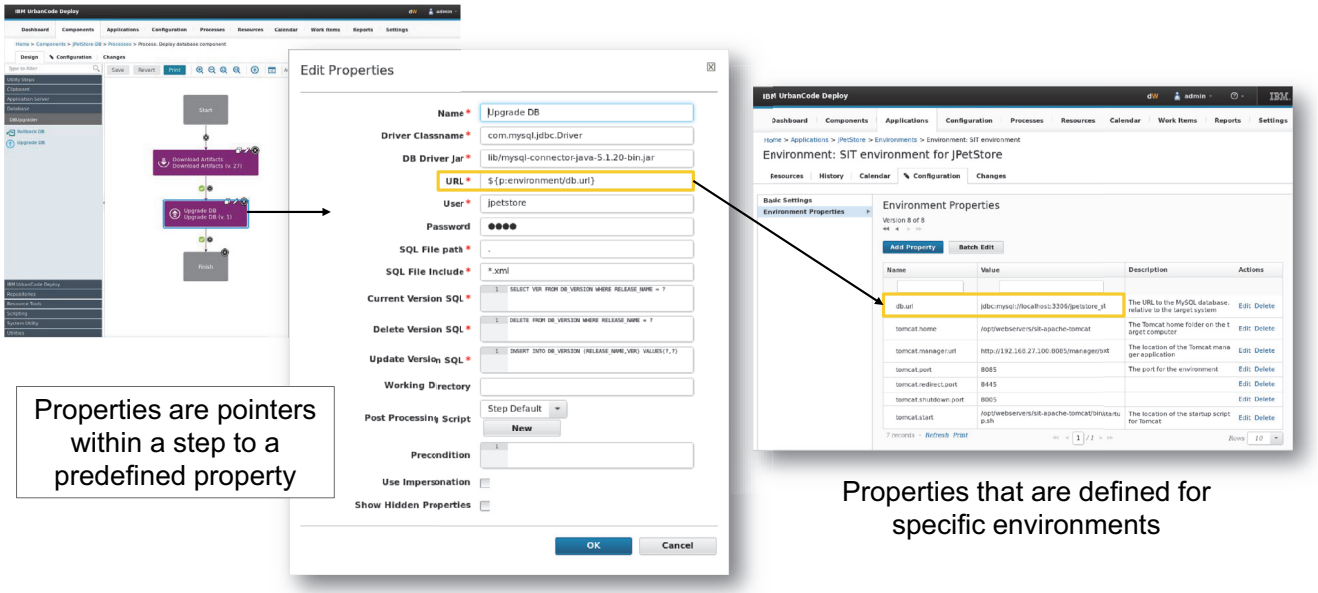
Topics

- Components overview
- Creating components and importing component artifacts
- Creating processes to deploy components
- Storing information with properties

Topic: Storing information with properties

IBM Training

Properties are variables that store information about different elements

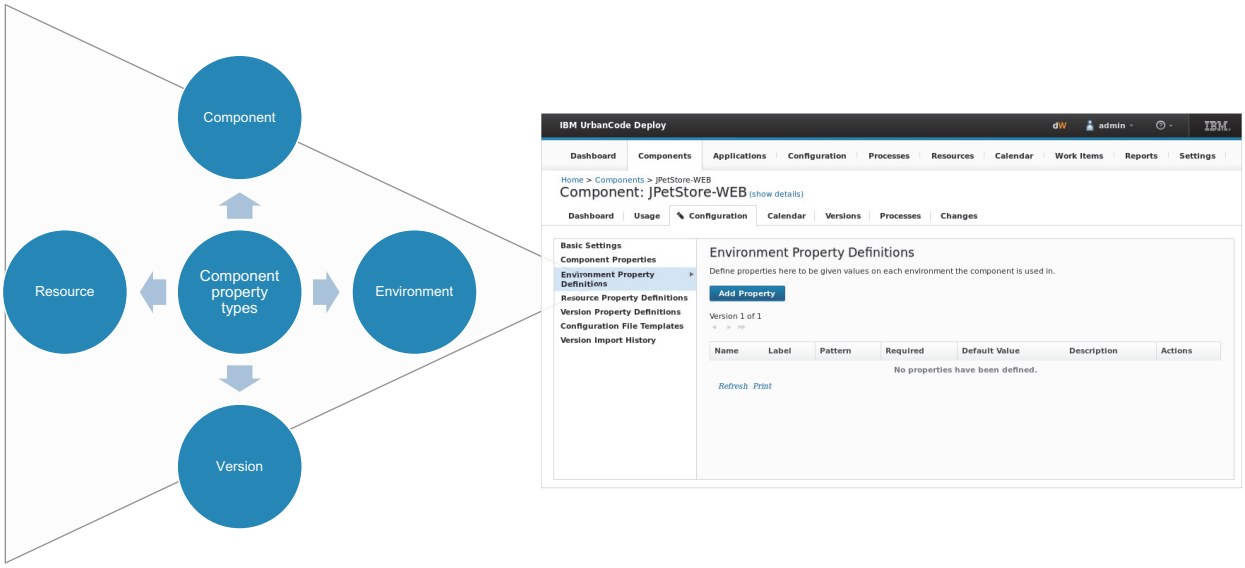


Properties are variables that store information about different elements

Properties are pointers to values. They store information about components, environments, processes, applications, and global properties for the system. In this example of a component step, the URL field points to a value that is defined for the environment. At run time, the environment property, in this case the db.url value, will be substituted here.

IBM Training

Components can have several types of properties



Components can have several types of properties

Component properties: Component properties are values defined and set directly on the component. They stay the same for all versions of the component. So any time you import a new version, the properties are already set.

Property definitions: Components have a special type of property called property definitions. Component property definitions can be used for environments, versions, and resources. You use definitions when you have a value that the component uses that needs to change each time you use that component. These properties are defined on the component, but their values are set in other places. For example, *component environment properties* are defined on a component, but the values are not specified until the component is deployed to an environment. In the previous slide, the URL field pointed to a predefined environment property called db.url. You can therefore assign a different database URL to each environment. You might want to do so if you have a test database at one URL for your development environment and a production database at another.

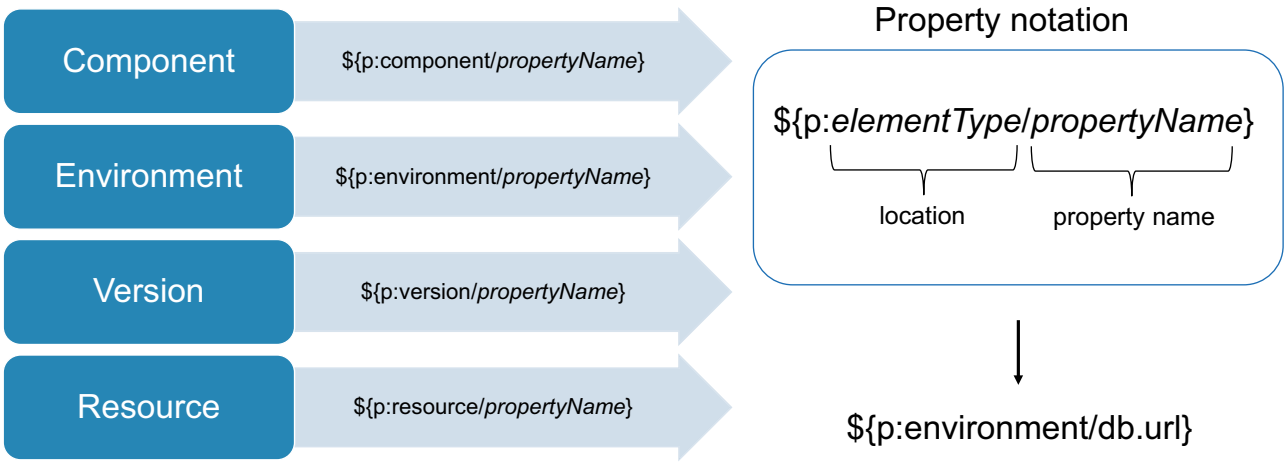
For example, if different database users have access to the development and test databases, you can set the database user name and password in each environment. Then you use a component environment property to point the correct database URL for each environment.

- **Component version properties** are defined on a component, but the values are specified on component versions. This type of property stores information that changes with each version of the component, such as a version number. Each component version can have a different value for its component version property.
- **Component resource properties** are defined on components, but they are not given a value until the component is added to an environment as a resource. Then the property behaves much like a resource property. Because a component can be added to an environment multiple

times, each instance of the component can have a different value for the component resource property. If you need to use two copies of a component in the same environment but use a different value between them, use a resource property definition.

IBM Training

Properties are referred to by scope



Properties are referred to by scope

Properties are referred to by scope. ***Scope*** is the type of artifact that contains the property, such as the environment or component. When you refer to a property, you use the notation that starts with a dollar sign; open curly brace; *p*, which stands for property; and then a colon. Then you identify the artifact, followed by a slash, the property name, and a closed curly brace.

${p:component/prop.name}, where the *p* stands for *property*. It means to look in this location with this name.

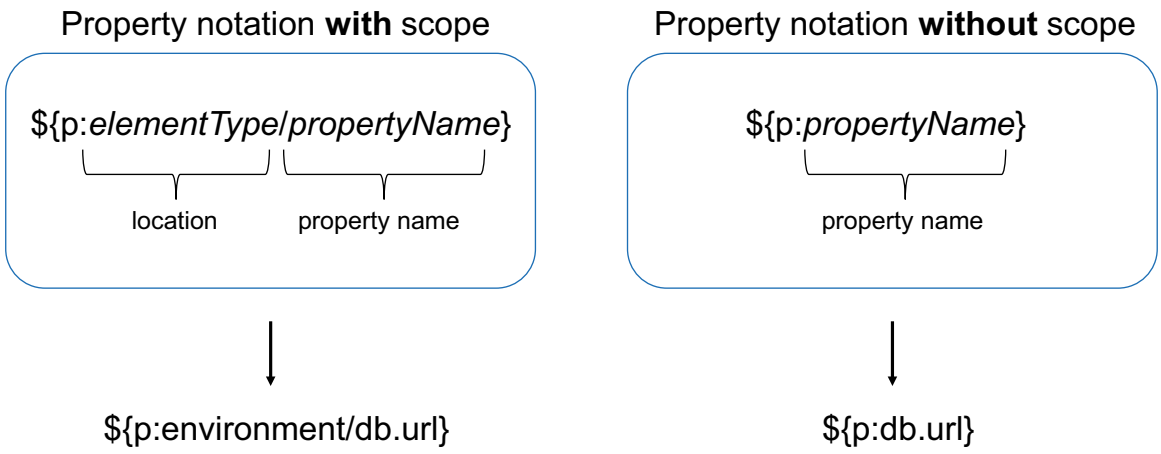
Keep in consideration where values change; that is where you want to set the properties.

This slide shows the most commonly used component properties and definition. Application, environments, resources, the system, and agents also have properties. For more information, see the IBM Knowledge Center:

https://www.ibm.com/support/knowledgecenter/SS4GSP_6.2.3/com.ibm.udeploy.doc/topics/ud_properties_overview.html

IBM Training

Properties can also be referred to without scope

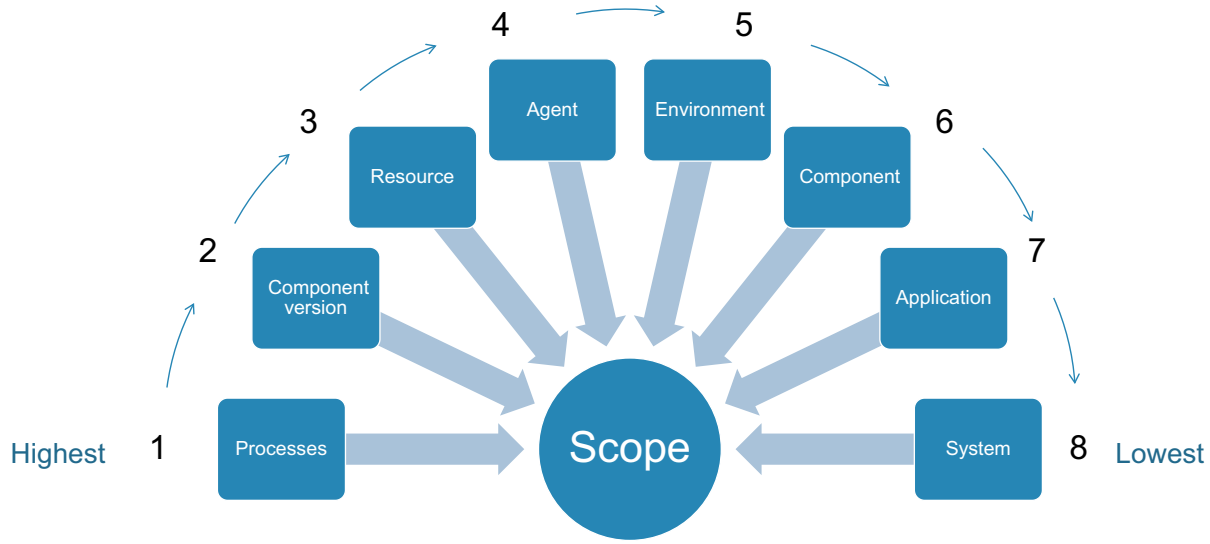


Properties can also be referred to without scope

There might be situations where you don't want to reference a specific artifact. In this case, you can refer to a property without scope by leaving out the artifact (${p:propertyName}). For example, with the environment variable, db.url, you can refer to it with the artifact name, environment, as you see on the left (${p:environment/db.url}). Or you can refer to the same environment variable without scope by removing the artifact name, as shown in the right (${p:db.url}). In this way, you can refer to properties defined in other places, such as in the application, system, or resource.

IBM Training

Property scope determines the precedence of properties with the same name



Unit 2: Configuring components and component processes

Property scope determines the precedence of properties with the same name

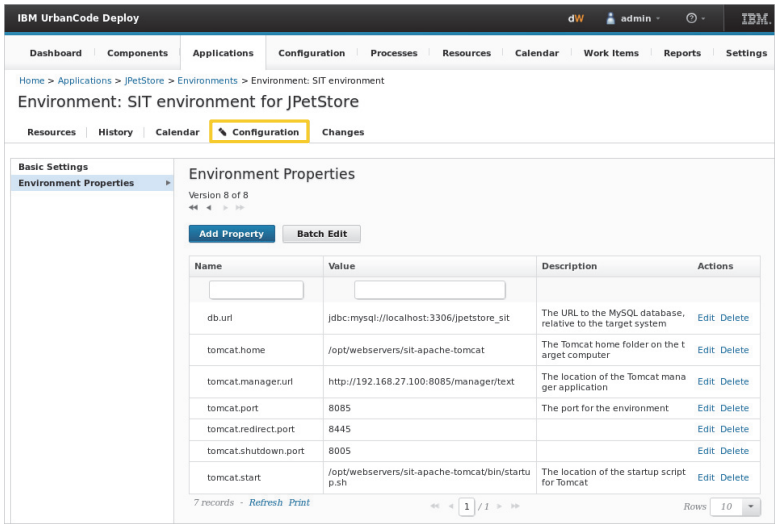
If a property is defined in multiple places, its value is determined by the property's order of precedence. You can use the order of precedence to override properties in specific contexts. If you refer to a property without scope, then the property with the highest order of precedence is used.

This slide shows the order of precedence from highest to lowest. The highest precedence is given to properties that are referenced in processes. The lowest precedence is given to properties referenced in the system.

In this way, you can override properties to change the application behavior without changing the code. For example, an application might have a database for testing and a separate database for production use. In this case, you might create an application property, such as databaseLocation, to store the testing database location. The application and component processes can refer to the property without specifying the application scope, like this: ${p:databaseLocation}. Then, in the production environment, you can override the application property with an environment property that specifies the production database location. The processes still refer to ${p:databaseLocation}, but in the context of the production environment, the environment property takes precedence; so the application uses the production database.

IBM Training

Properties are primarily set from the Configuration tab



You can also set properties here:

- Processes
- Rest API and command line
- Output properties

Properties are primarily set from the Configuration tab

You can set properties in the following locations:

- **Configuration tab**: You can manage properties for applications, components, and environments.
- **Processes**: Several different steps are available to create and set properties in processes.
- **REST API and command-line client**: Several commands in the REST API and in the command-line client can set properties.
- **Output properties**: You can pass properties from one step to another within a process.

Properties for applications, processes, components, component versions, resources, and agents are available on their respective **Configuration** tabs. System (global) properties are available on the **Settings** tab by clicking **Settings** > **Properties**.

IBM Training

Unit summary

- A component has three major parts: versions, processes, and properties
- Components are added by connecting the server to the system that hosts the artifacts
- A component version contains a set of deployable artifacts
- A component process is a series of user-defined steps that operate on a component's artifacts
- Component properties are values that are assigned on individual components
- Property definitions are defined on the component, but their values are set in other places

Unit summary

IBM Training

Exercises: Configuring components and component processes

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

Exercises: Configuring components and component processes

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

Applications identify the component versions that must be deployed together. They also define the different environments that the components must go through on the way to production. In this unit, you learn to create an application and add components, create environments and add resources, and create application processes.

IBM Training

Unit objectives

- Describe the structure of an application
- Explain how agents communicate with the server
- Describe the properties used for resources
- Create an environment for deployments
- Map resources to environments
- Create an application process to install components

Unit objectives

IBM Training

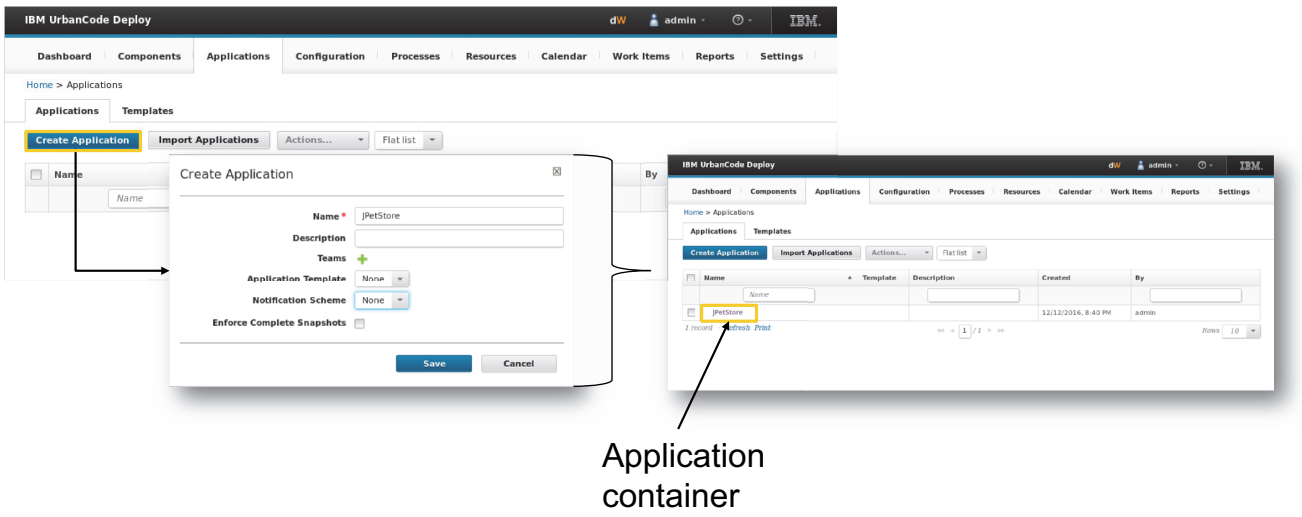
Topics

- Managing components in applications
- Configuring the agent as a deployment-target host
- Organizing resources in a resource tree
- Setting up deployment targets with environments
- Coordinating deployments with application processes

Topic: Managing components in applications

IBM Training

Applications associate resources with environments and define processes to run deployments



Applications associate resources with environments and define processes to run deployments

An application is a container for components, processes, and resources. When you create an application, you determine a few things:

- You'll want to identify who has access to the application. You can grant access to teams, and you can set the application security type.
- You decide if you want to base your application on a template. Application templates can help you to be sure that new applications contain the types of components, environments, and resources that you specify. You can also specify the application properties, environment properties, and environment gates for an application. Although this course doesn't cover templates, you can consult the video demonstration in the notes section of this slide for more information.

  For more information about creating application templates, see the video:

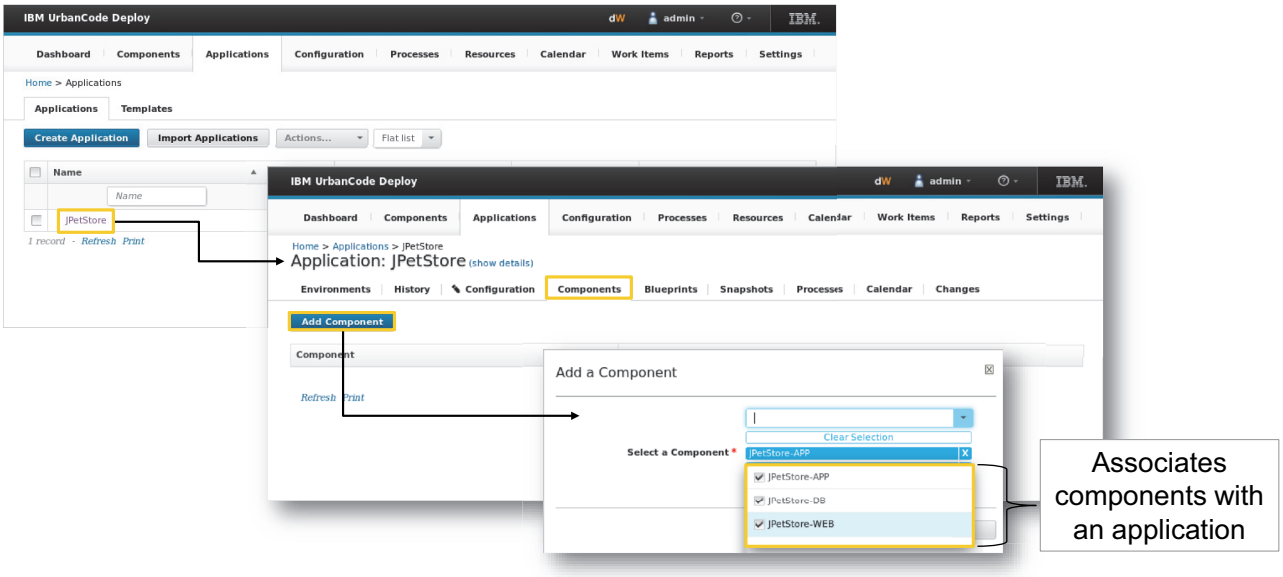
  https://developer.ibm.com/urbancode/videos/creating-application-templates/

  For more information about creating applications based on templates, see the video:

  https://developer.ibm.com/urbancode/videos/creating-applications-based-on-templates-urbancode-deploy-v-6-2/
- You can choose to use a Notification Scheme. A notification scheme sends out notifications that are based on events, such as when an application deployment fails or succeeds.

IBM Training

An application groups components together

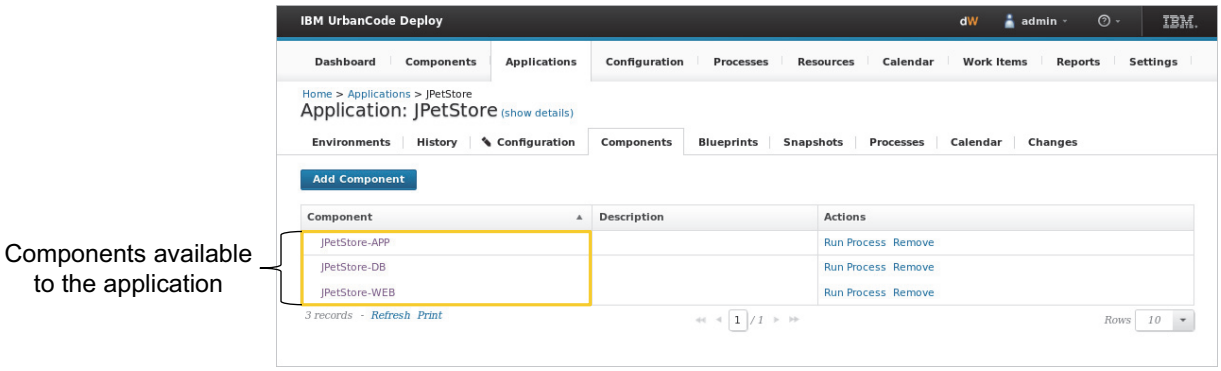


An application groups components together

After defining the application itself, the next step is to identify the components that it manages.

IBM Training

Applications manage components and map the dependencies between them



Applications manage components and map the dependencies between them

When you associate components with an application, they are available to be used in deployments.

IBM Training

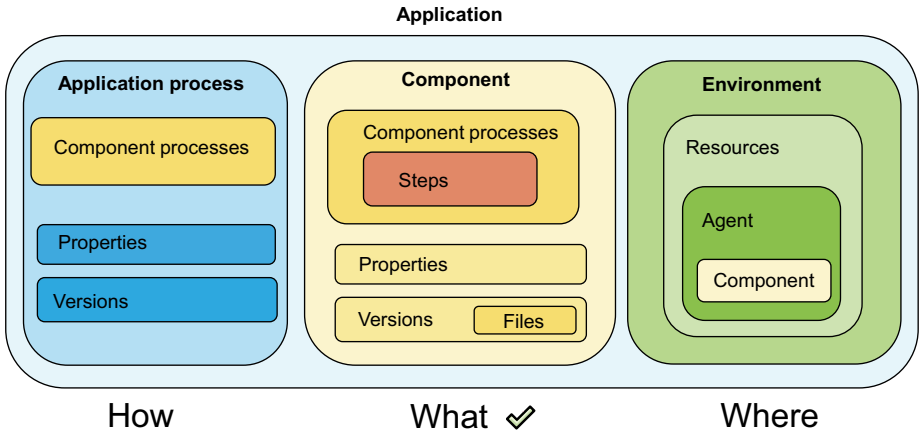
Topics

- Managing components in applications
- Configuring the agent as a deployment-target host
- Organizing resources in a resource tree
- Setting up deployment targets with environments
- Coordinating deployments with application processes

Topic: Configuring the agent as a deployment-target host

IBM Training

Applications determine the *what*, *where*, and *how* of a deployment



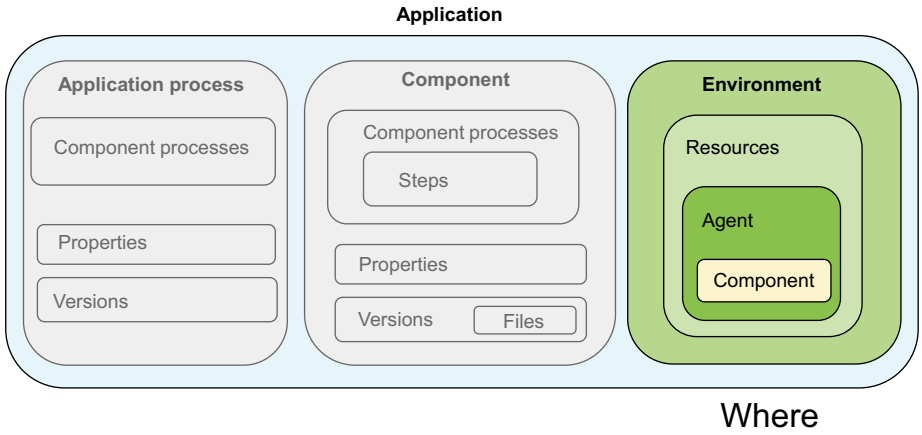
Applications determine the what, where, and how of a deployment

The introduction of this course described how applications bring together the pieces of a deployment. This application model provides the container for what gets deployed (the components), where it gets deployed (the environments), and how it gets deployed (the processes).

In the previous unit, you learned how to set up the components. In this next section, you learn how to set up an environment.

IBM Training

An environment associates components with an agent on the target host

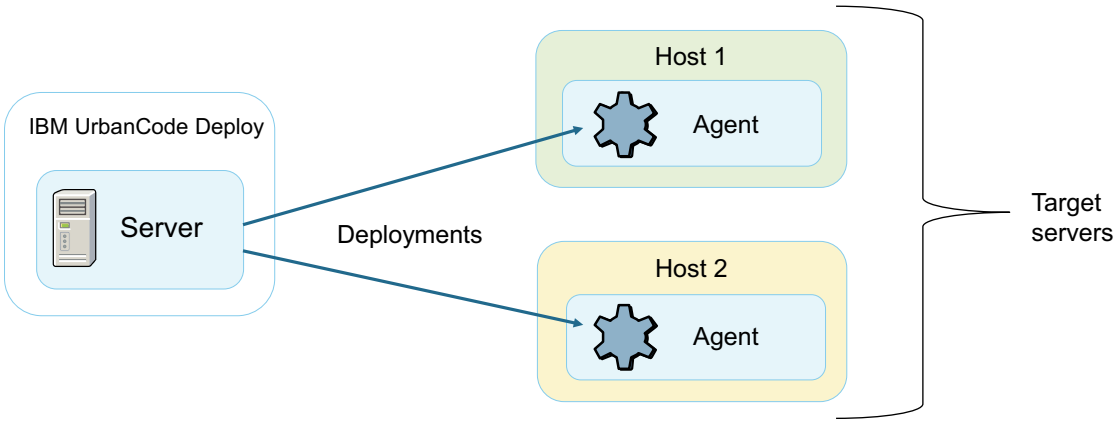


An environment associates components with an agent on the target host

The environment is the “where” of the application model. First, you need to configure an agent. After that, you can create the resources and environments. Next, you associate the resources with the environments.

IBM Training

An agent is a lightweight process that usually runs on a deployment-target host



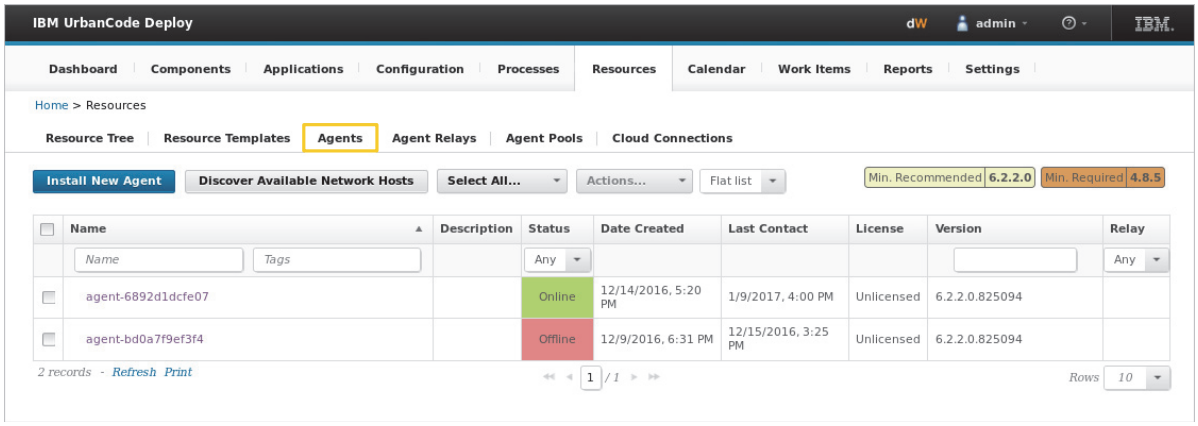
An agent is a lightweight process that usually runs on a deployment-target host

Agents are processes that run on the target environment. They communicate with the server and perform the automated steps that are defined in a component process.

Usually, an agent runs on the same host where the resources that it handles are located.

IBM Training

Agents are installed from the user interface or the command prompt

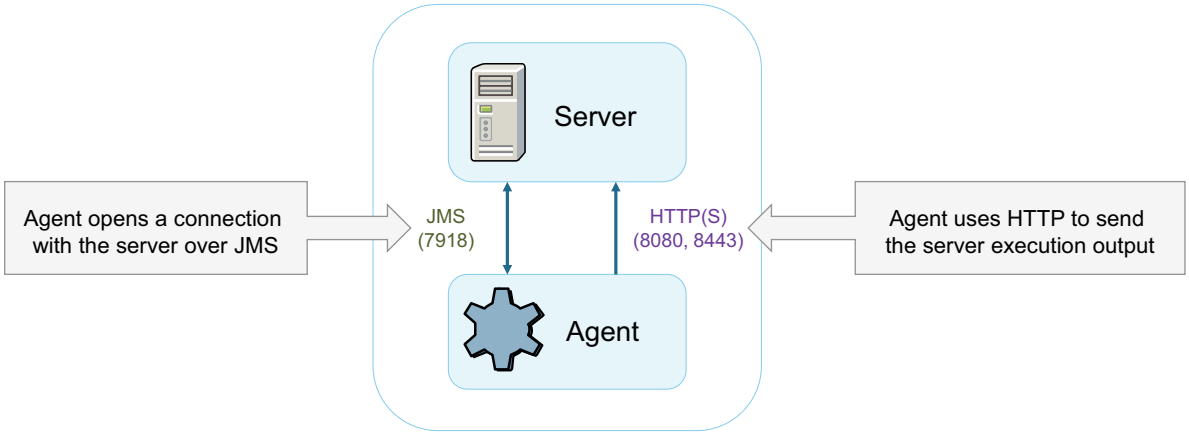


Agents are installed from the user interface or the command prompt

Agents are installed with the installation scripts that are provided with the IBM UrbanCode Deploy installation files. When you bring an agent online for the first time, it creates a resource with the same name as the agent.

IBM Training

Agents open direct connections to the server



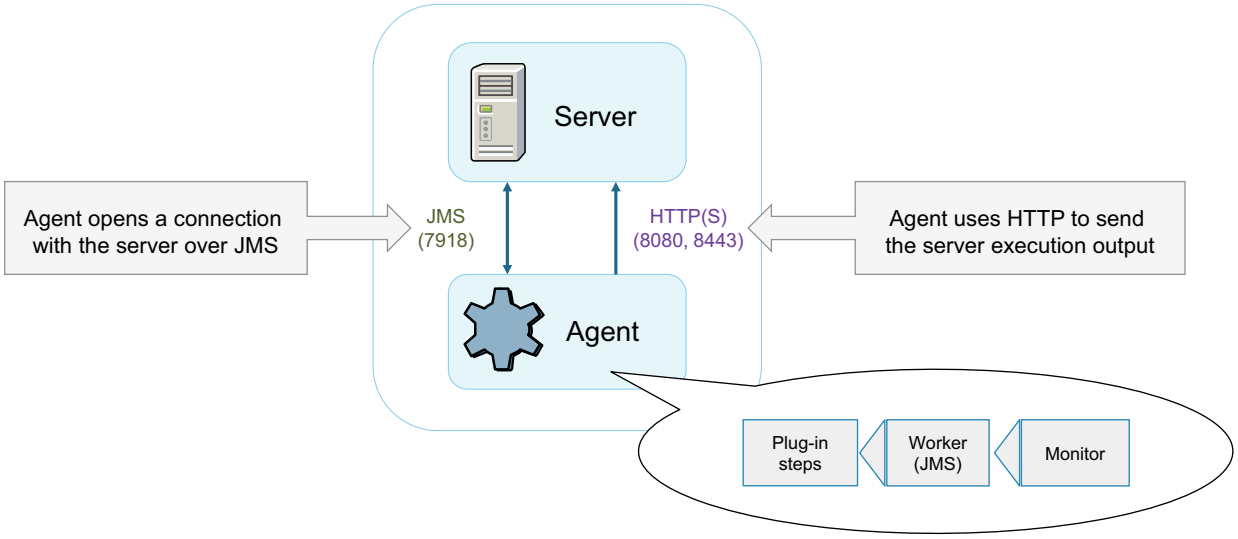
Agents open direct connections to the server

When the agent is installed on the deployment target, it opens access to transfer files and execute commands. All processes that the IBM UrbanCode Deploy server requests, including packaging, configuration, and deployments, run on hardware that is assigned to agents.

Agent communications employ JMS, HTTP, and HTTPS protocols to communicate with the server. First, the agent uses JMS to establish a connection with the server. Over JMS, the server instructs the agent to run a plug-in step and provides a URL for HTTP connectivity. The agent downloads the required plug-in, runs the step, and then sends execution output back to the server over HTTP.

IBM Training

An agent consists of a worker process and a monitor process



An agent consists of a worker process and a monitor process

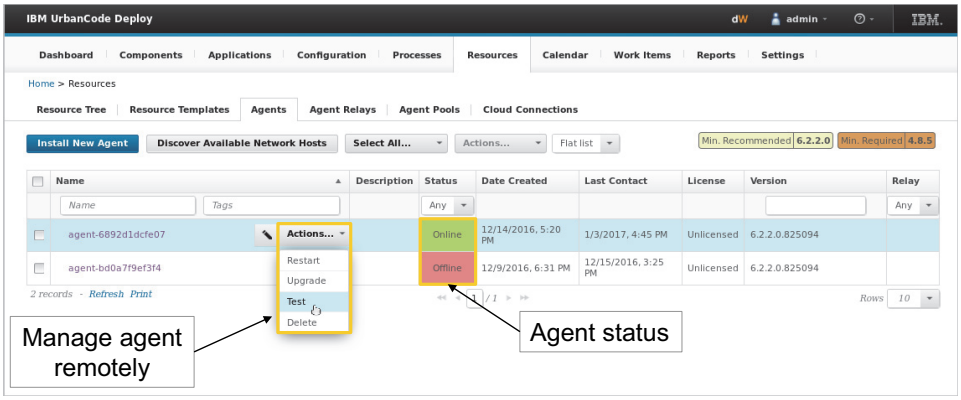
Although an agent is typically considered a single process, an agent consists of both a worker process and a monitor process.

The worker process runs the deployment work after it receives commands from the server. Work commands come from plug-in steps. The worker process uses JMS for system communications and HTTP REST services when it completes plug-in steps or retrieves information from the server.

The monitor process is a service that manages the worker process by starting and stopping, handling restarts, upgrades, and security. The agent monitor service uses JMS for all server communications and for sending commands, such as the run step command, to the worker process.

IBM Training

Many agent features are managed from IBM UrbanCode Deploy



Many agent features are managed from IBM UrbanCode Deploy

After an agent is installed, you can manage many of its features from the **Resources** tab with the monitor process. You can edit, restart, upgrade, test, inactivate, or delete the agent.

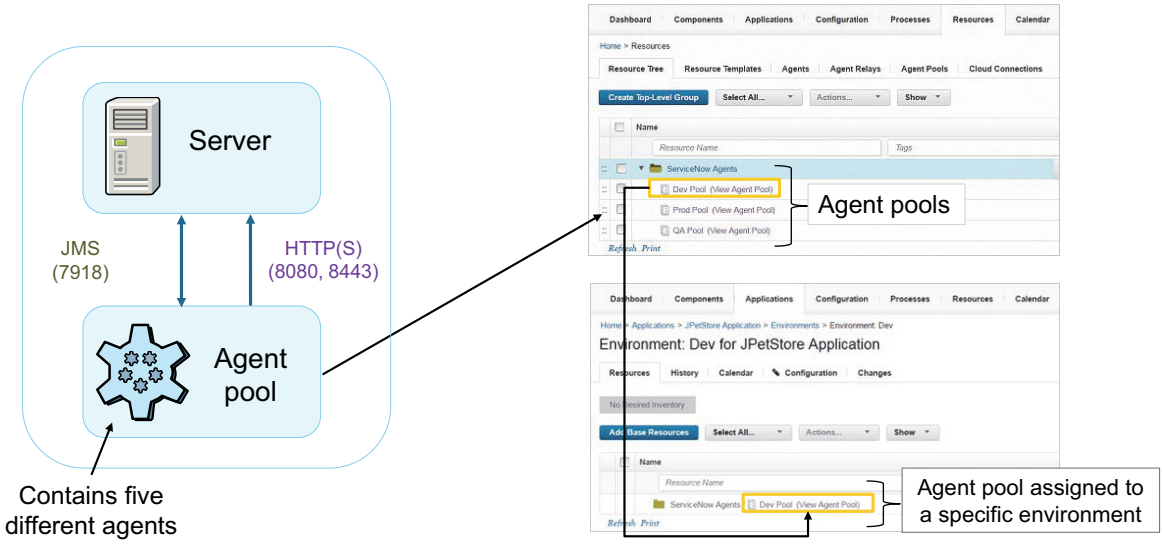
Agents must be online in order to run deployments.

In the status column, you view its state:

- **Offline**: No connection
- **Connecting**: JMS connectivity is available, but there is no HTTP connection. When an agent is in this state, the server can send a signal for upgrade or restart.
- **Online**: HTTP and JMS connectivity

IBM Training

Agent pools manage agents that are installed in different environments



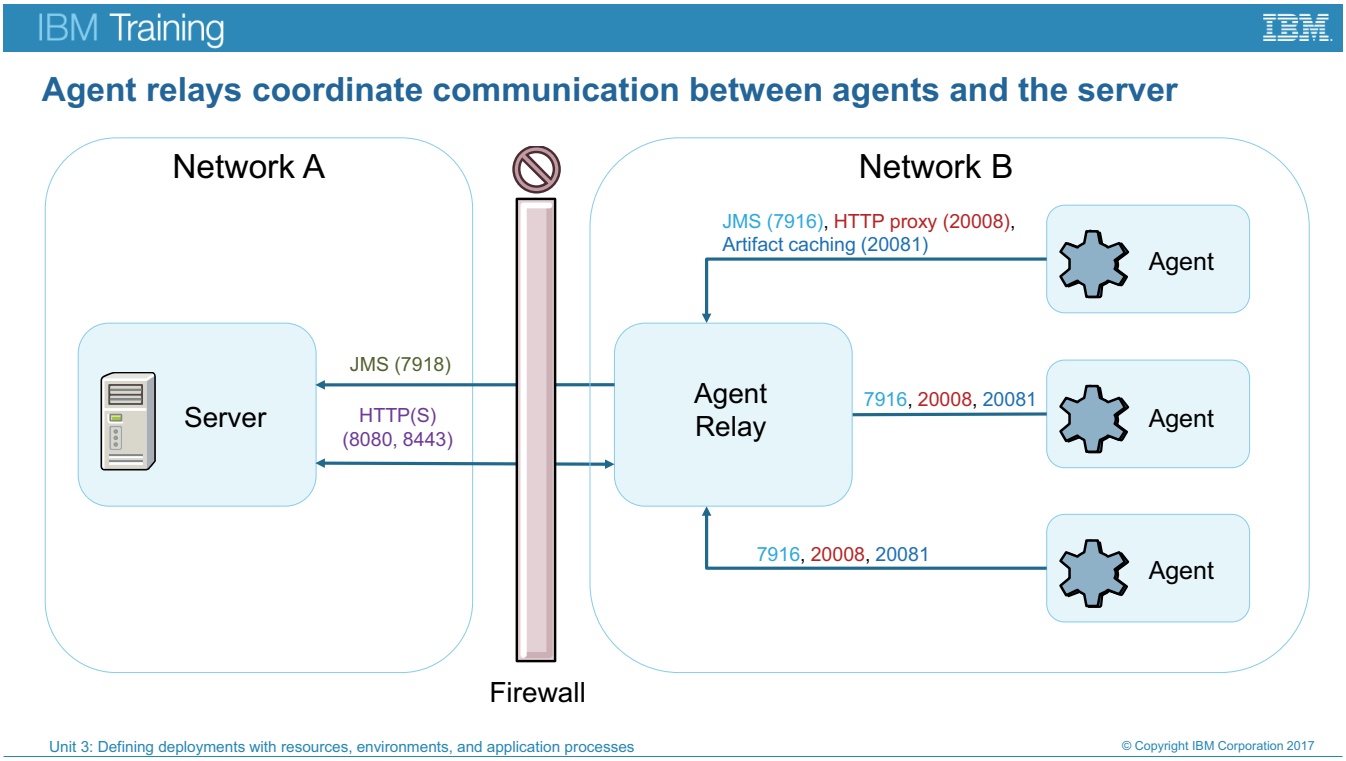
Agent pools manage agents that are installed in different environments

To provide relief to an overburdened agent, you can use an agent pool to spread the deployment processing work among participating agents. You assign agents to pools and pools are assigned to resources, just like lone agents. When an agent pool is assigned to a resource, work items are sent to eligible agents.

The **Resource Tree** tab shows a typical design of the structure of an agent pool. Notice the high-level folder called ServiceNow Agents. The folder contains three agent pools: Dev Pool, QA Pool, and Prod Pool. Agents are assign within a pool specifically configured for the environment.

In the JPetStore environment graphic, notice that the Dev Pool agent pool is mapped to the Dev Environment. For each environment (QA and Prod), the Environments page displays the agent pools associated with the specified environment. With this configuration, the application process runs only on the available agents in the agent pool associated with the environment.

Note: The agent that does the actual work is chosen randomly from available online agents in the pool. The current workload that is running on the agents in the pool is not considered. The agent that was most recently selected to do work is not exempt from subsequent selections. The selected agent is always randomly selected from available online agents in the pool.



Agent relays coordinate communication between agents and the server

In simple configurations, agents communicate directly with the UrbanCode server, as mentioned earlier. In more complex situations, this communication becomes inefficient or impossible. Some examples are when the number of agents increase from tens to hundreds to thousands, network topology becomes complex, or agent-server communication crosses firewalls with port restrictions.

The Agent Relay helps alleviate these challenges by consolidating agent traffic and communicating with remote agents.

- **They can connect large groups of agents to the primary server**: Instead of each agent connecting directly to the server, agents connect to agent relays, which are then connect directly to the server. Using agent relays in this way reduces load on the server because the server has fewer direct connections.
- **They can simplify communication across networks and firewalls**: If multiple agents are in a remote network without an agent relay, each agent must connect to the server individually. In this case, each agent must have network permission to connect to the server, including firewall permissions. With an agent relay, the agents connect only to the relay, and the relay is the only system that contacts the server directly, such as with a VPN or tunnel.

The diagram shows the main default ports that are involved in communication between agents, agent relays, and the server.

IBM Training

Topics

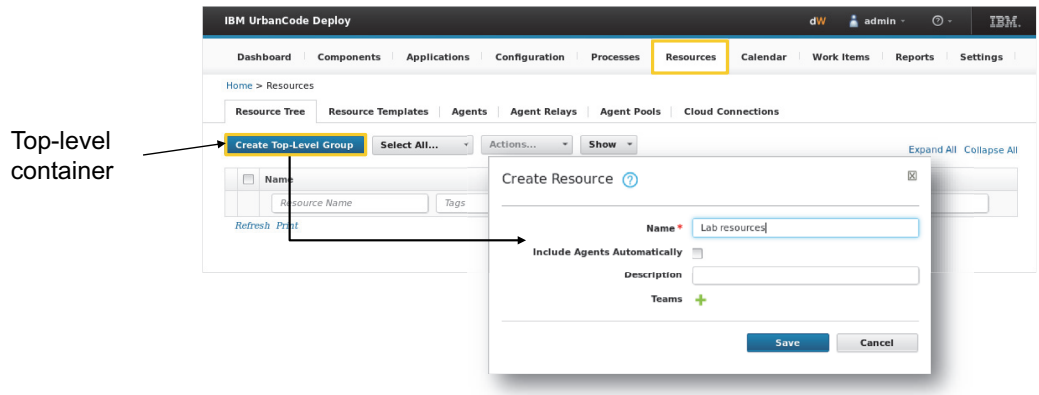
- Managing components in applications
- Configuring the agent as a deployment-target host
- Organizing resources in a resource tree
- Setting up deployment targets with environments
- Coordinating deployments with application processes

Topic: Organizing resources in a resource tree

After installing an agent, you can identify the resources for the deployment.

IBM Training

A resource tree organizes *what* is being deployed and to *where*

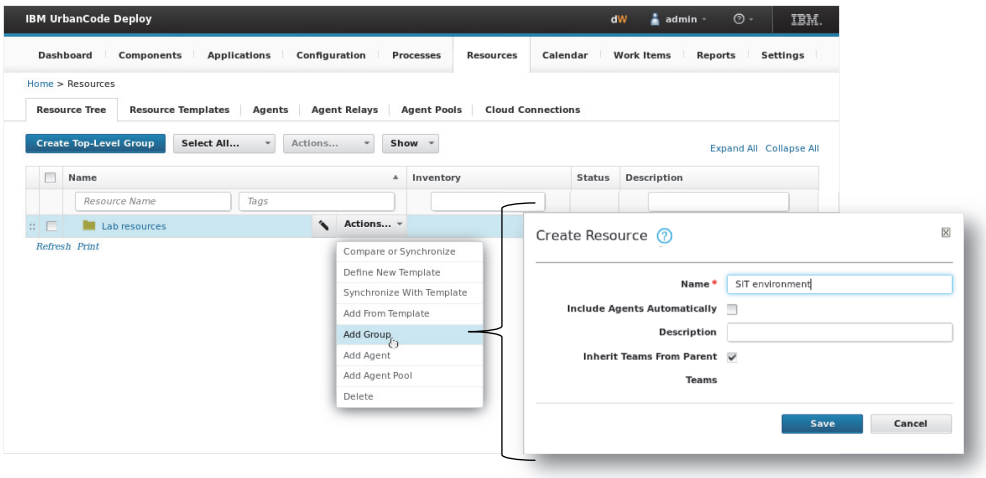


A resource tree organizes what is being deployed and to where

Resources are created on the **Resource Tree** tab. Resources associate agents with components. After they are mapped to an application environment, they provide deployment targets.

IBM Training

Resource groups enable collections of resources to be easily reused

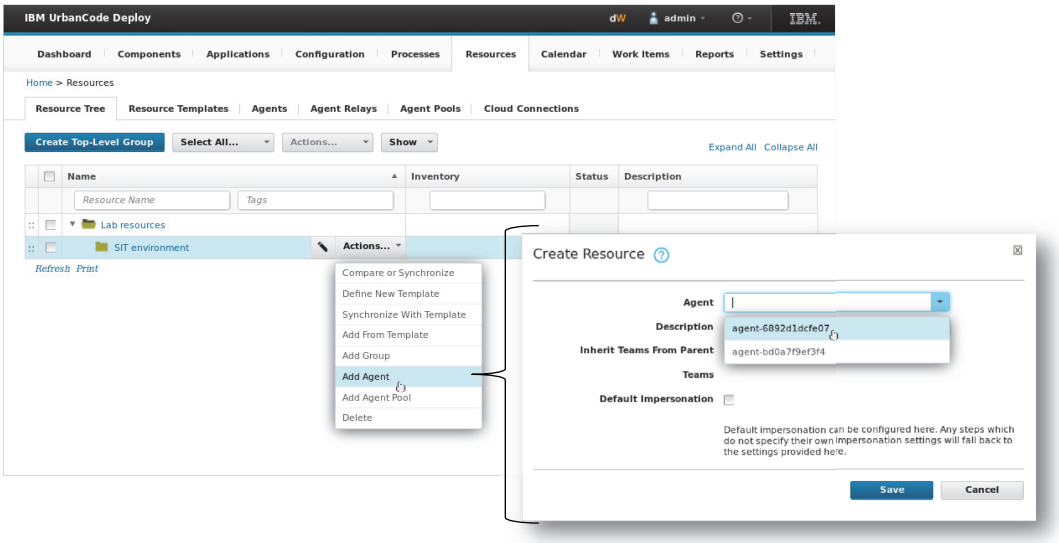


Resource groups enable collections of resources to be easily reused

First, you create a top-level group for all the resources within that group. A resource group is a useful organizational element. You can break down a complex environment into a number of blocks and manage them across hundreds of applications.

IBM Training

Agent-type resources represent the computers where components are deployed

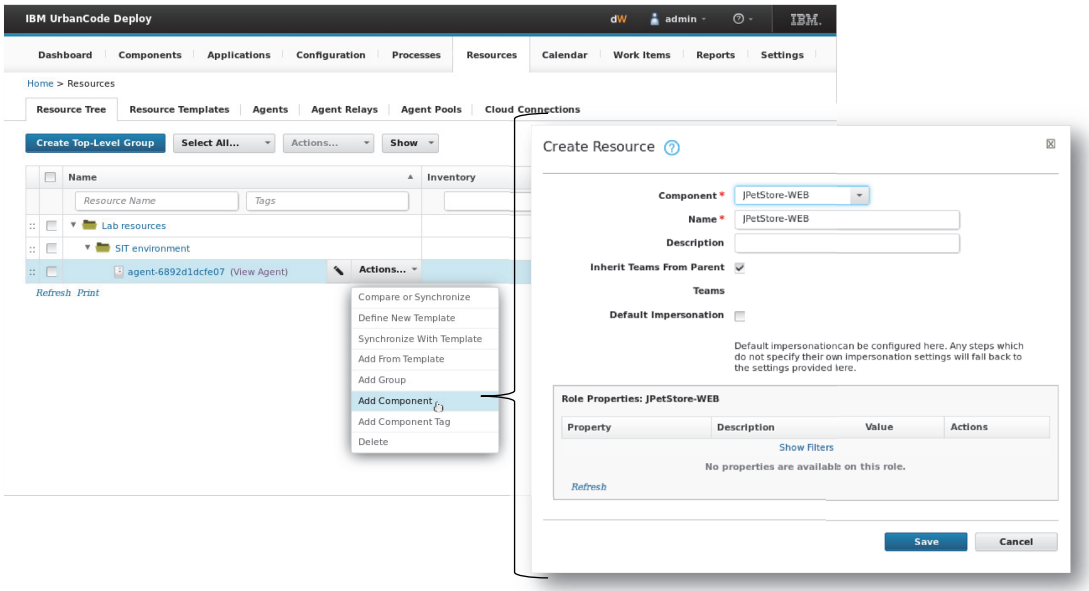


Agent-type resources represent the computers where components are deployed

Next, you add an agent. An agent lives on a resource and performs deployment operations.

IBM Training

Component-type resources represent the components that are deployed to target environments

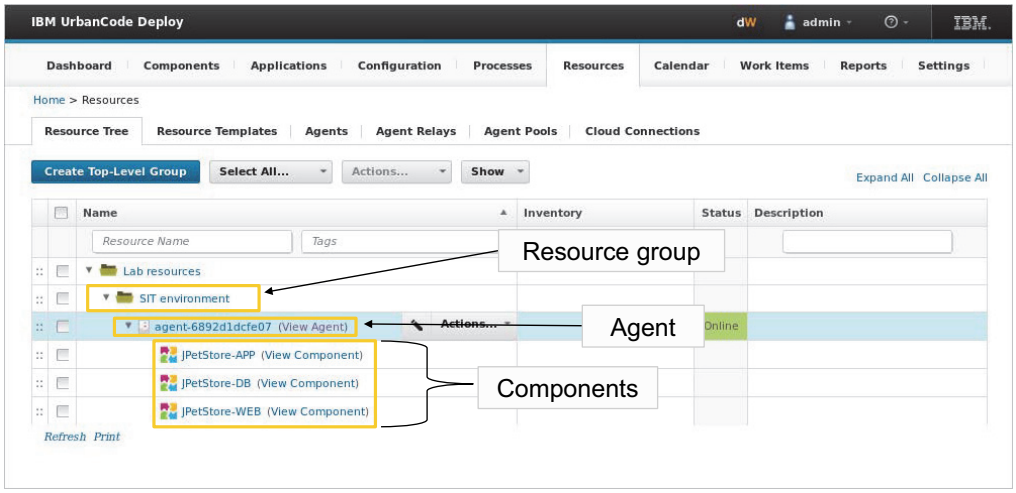


Component-type resources represent the components that are deployed to target environments

Finally, you associate the components. A component-type resource can point directly to the agent or agent pool that deploys it, or it can point to another resource in a hierarchical chain of resources.

IBM Training

The Resource Tree page displays resources in a hierarchical tree structure



The Resource Tree page displays resources in a hierarchical tree structure

A resource tree shows a hierarchical representation of the mapping between logical and physical. Each row in a resource tree represents a resource and has an Action menu that is associated with it. The available actions depend on whether the resource represents an agent, agent pool, component, or organizational group.

IBM Training

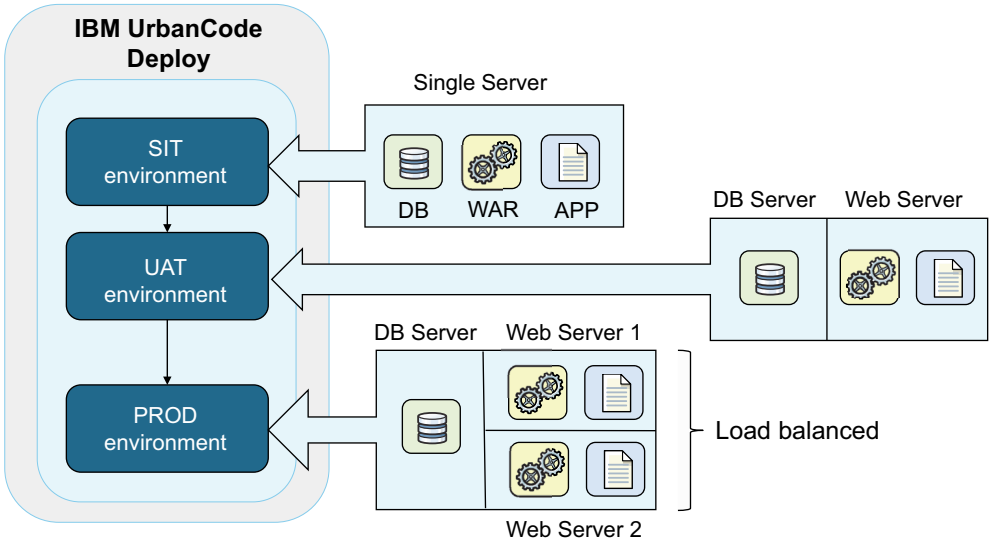
Topics

- Managing components in applications
- Configuring the agent as a deployment-target host
- Organizing resources in a resource tree
- Setting up deployment targets with environments
- Coordinating deployments with application processes

Topic: Setting up deployment targets with environments

IBM Training

Environments can have different topologies, configurations, and settings



Environments can have different topologies, configurations, and settings

An environment is the application's mechanism for bringing together components with the agent that deploys them. Now that you have an agent with mapped components, you can create the environments and associate the resources. Environments typically include host systems and agents. When a deployment is run, it is always done so in an environment.

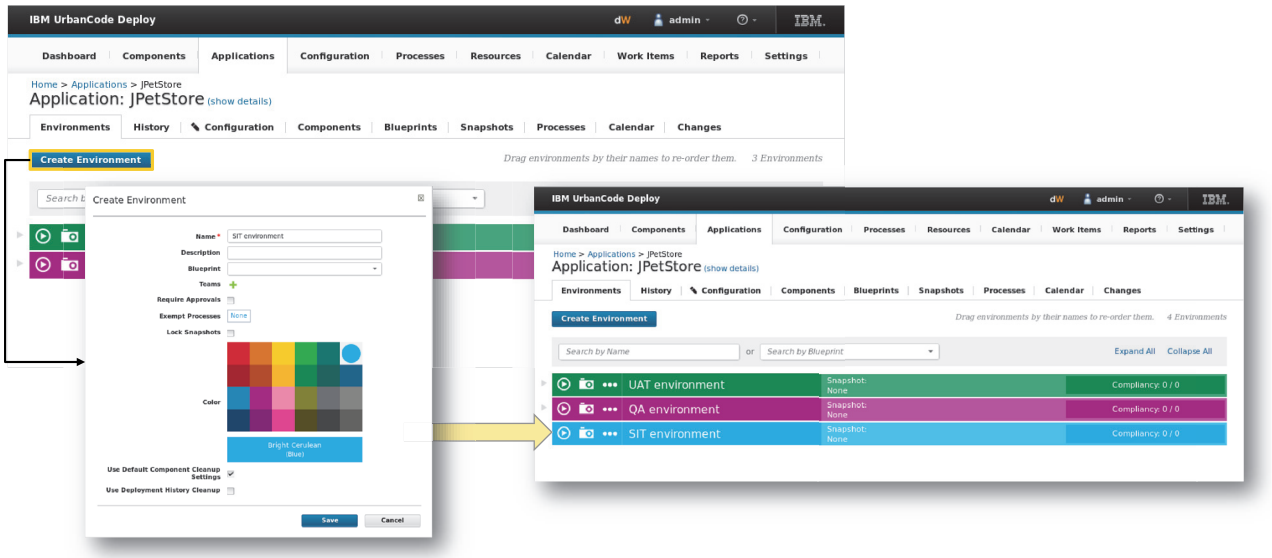
Environments can have different topologies.

The diagram shows three different server topologies:

- The SIT environment has a single server with an installed agent that contains all three components: Database, Web, and Configuration.
- The UAT environment has two servers with an installed agent on each. The database server contains the database component and the web server contains the web and configuration components.
- The PROD environment has a more distributed and production-ready topology. Each server has an installed agent with the database server that still contains the database component, but there are load-balanced web servers that each contain a web and configuration component.

IBM Training

Environments represent the target to which your files are being deployed



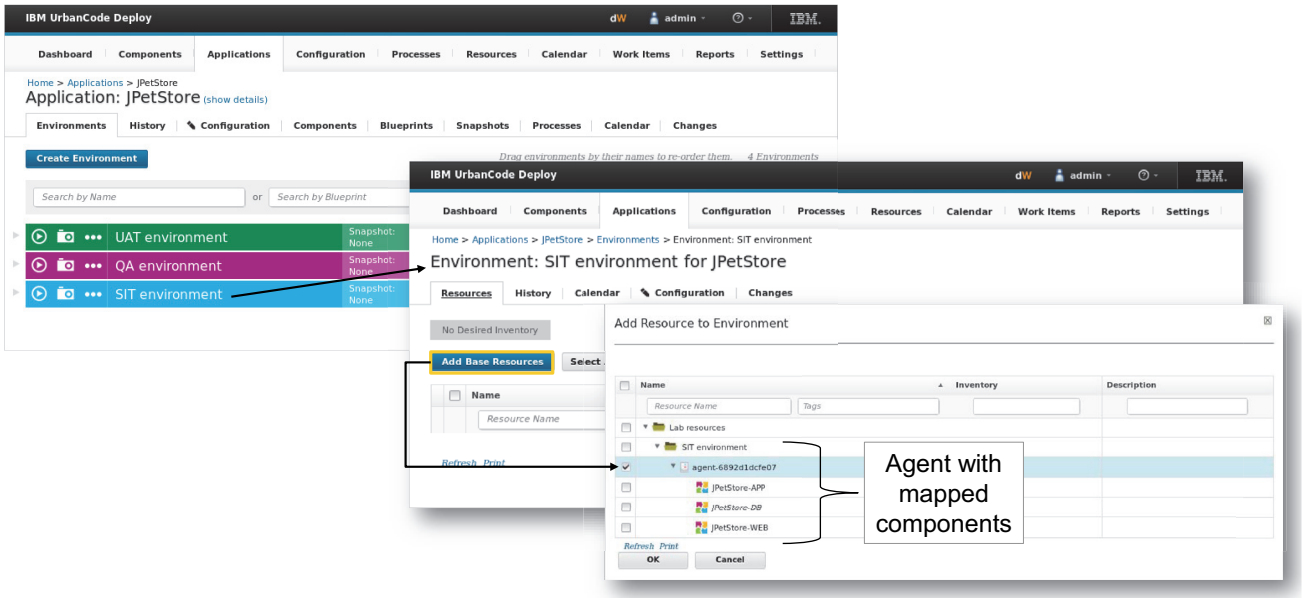
Environments represent the target to which your files are being deployed

You create an environment for each target to which you deploy an application. Environments are typically modeled on some stage of the software project lifecycle, such as development, quality assurance, or production. The initial environment is typically uncontrolled and often is used to create snapshots. A snapshot represents a set of component versions that are known to work together. You look at snapshots later in this course.

Creating an environment involves selecting one or more agent resources with mapped components.

IBM Training

Mapping resources and components to the environment provides a topology

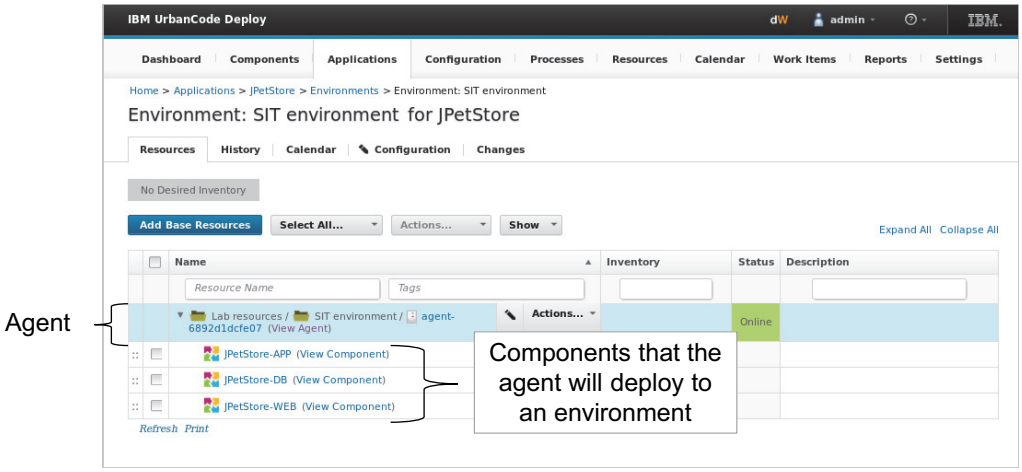


Mapping resources and components to the environment provides a topology

Before you can run a deployment, you must define at least one application environment that associates components with an agent on the target host.

IBM Training

Resources associate agents with components



Resources associate agents with components

After components are mapped to an application environment, resources provide the deployment targets.

IBM Training

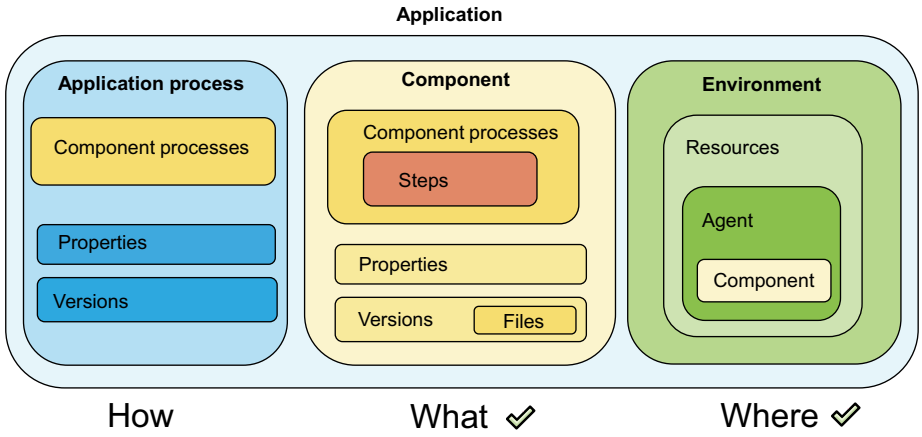
Topics

- Managing components in applications
- Configuring the agent as a deployment-target host
- Organizing resources in a resource tree
- Setting up deployment targets with environments
- Coordinating deployments with application processes

Topic: Coordinating deployments with application processes

IBM Training

An application process orchestrates the order of component process execution

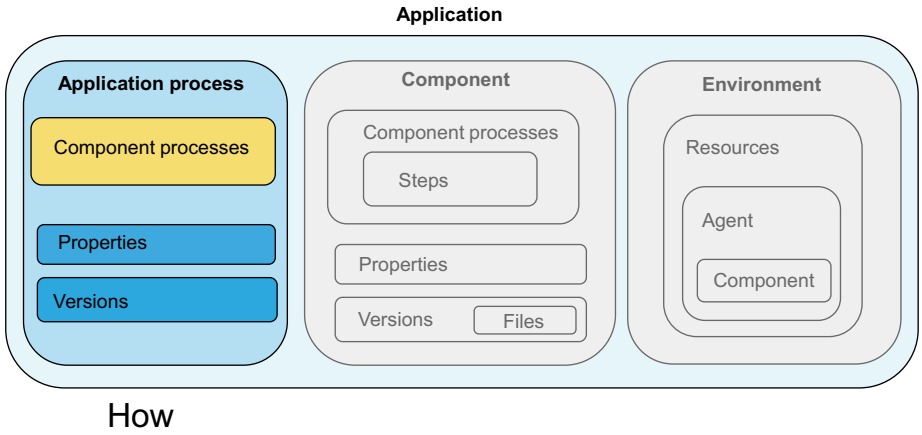


An application process orchestrates the order of component process execution

In many cases, application processes call component processes. For example, an application process can call the component processes that deploy their associated components.

IBM Training

The application process is the *how* of the application model

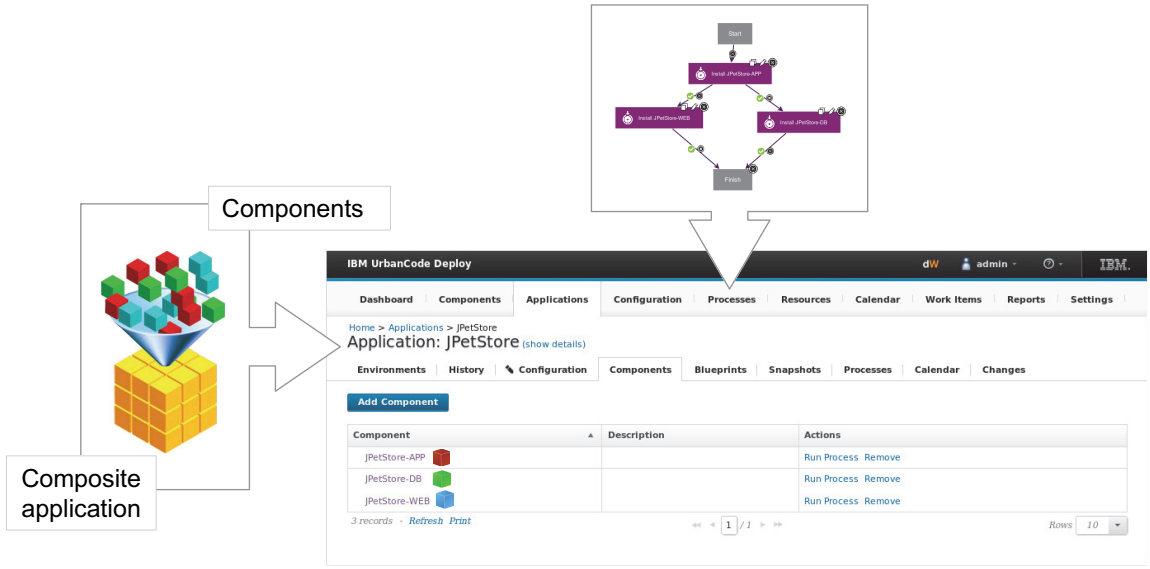


The application process is the how of the application model

In this next section, you look at application processes, which describe the “how” in the application model.

IBM Training

An application process determines what components are deployed together

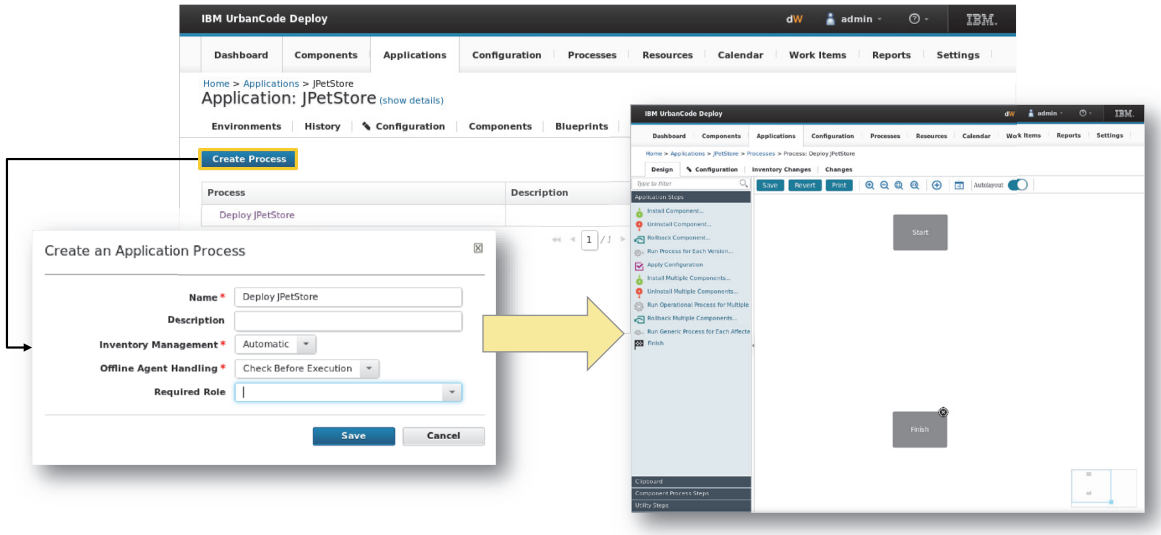


An application process determines what components are deployed together

When you create application processes, you not only determine which components are grouped together, but you also identify the flow of the deployment. Component processes are run in a particular order as managed by the application process.

IBM Training

Application processes are created with the process editor



Application processes are created with the process editor

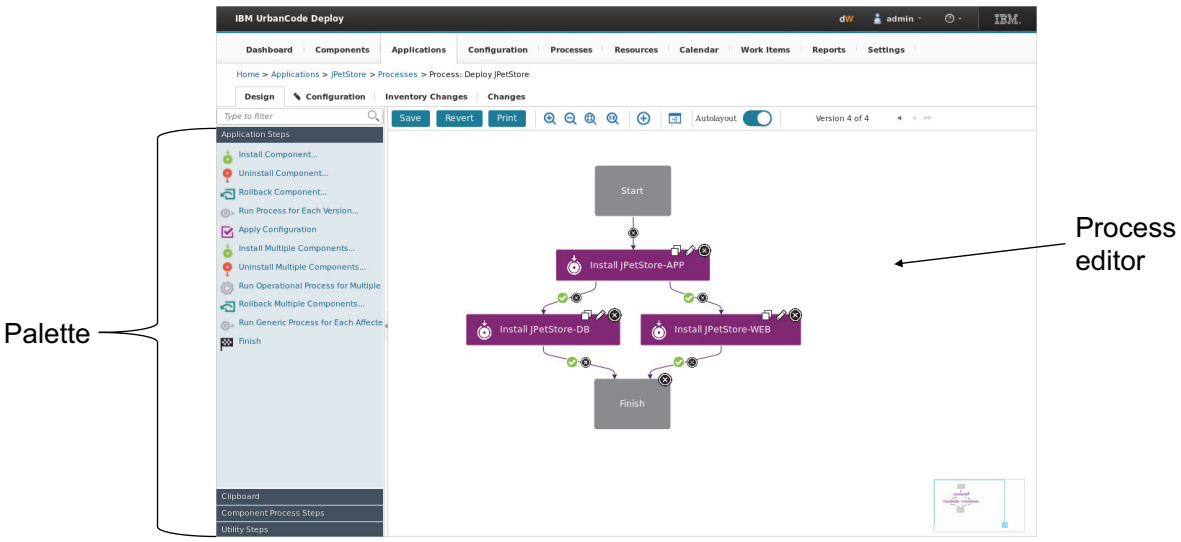
Application processes, like component processes, are created with the process editor. They are authored with a visual drag-and-drop editor and composed of steps that call the component processes.

Similar to component processes, you can use several step types, such as these examples:

- Conditional processes
- Switch steps
- Branching and joining steps
- Process step properties
- Post-processing preconditions

IBM Training

Application processes provide overall deployment orchestration

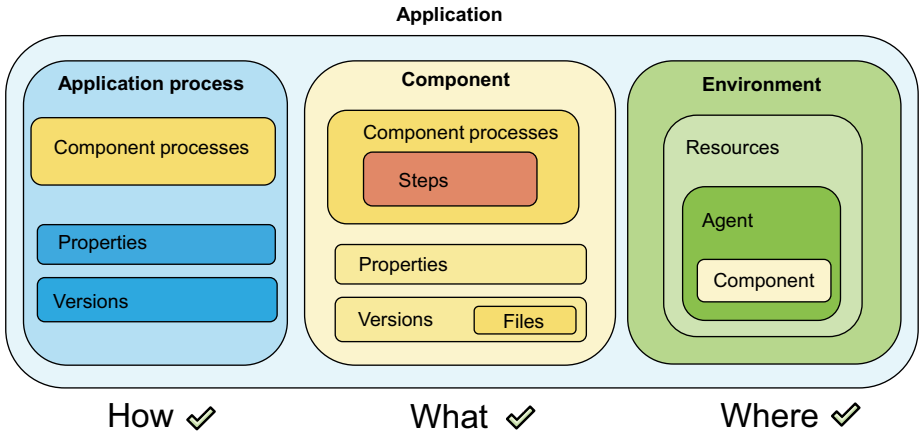


Application processes provide overall deployment orchestration

An application process plays a coordination role. Steps in an application process represent the various component processes and tasks that are required to complete a deployment. While it looks like a component process, these steps are not from plug-ins. Steps for applications are part of IBM UrbanCode Deploy.

IBM Training

With all of the application pieces in place, you are ready for deployment



With all of the application pieces in place, you are ready for deployment

IBM Training

Unit summary

- An application is a container for components, processes, and resources
- Agents are processes that run on the target environment
- Environments represent the target to which your files are being deployed
- Application processes orchestrate the order of component process execution

Unit summary

IBM Training

Exercises: Defining deployments with resources, environments, and application processes

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

Exercises: Defining deployments with resources, environments, and application processes

Unit 4 Deploying applications to target environments

You run a deployment by running an application process in a target environment. In this unit, you learn how to deploy to an environment, create and deploy snapshots, and compare environments to view the inventory in each one.

IBM Training

Objectives

- Describe what happens in a deployment
- Describe the difference between full and incremental deployments
- Describe why and how to use snapshots
- View which versions are deployed to environments
- View the file differences between the environments
- Preview deployments for impact analysis
- Migrate changes from one environment to another

Unit objectives

IBM Training

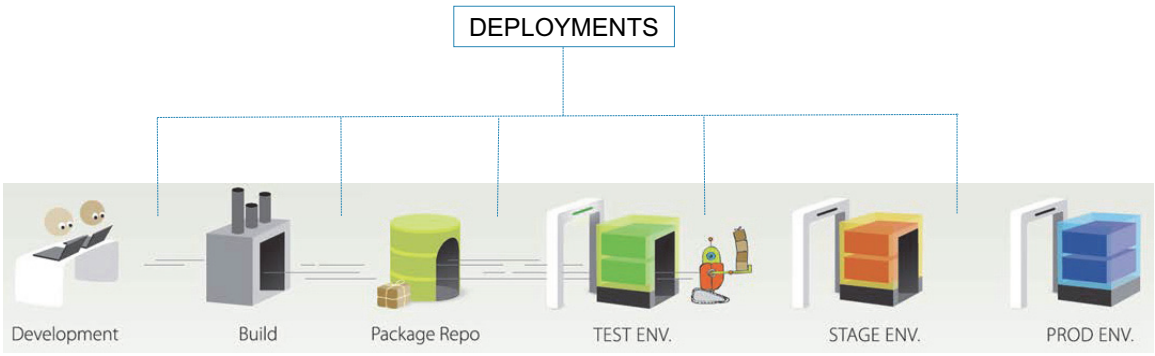
Topics

- Deploying to an environment
- Examining application environment inventory
- Deploying applications with snapshots

Topic: Deployment to an environment

IBM Training

A deployment is the promotion of an application from one environment to the next

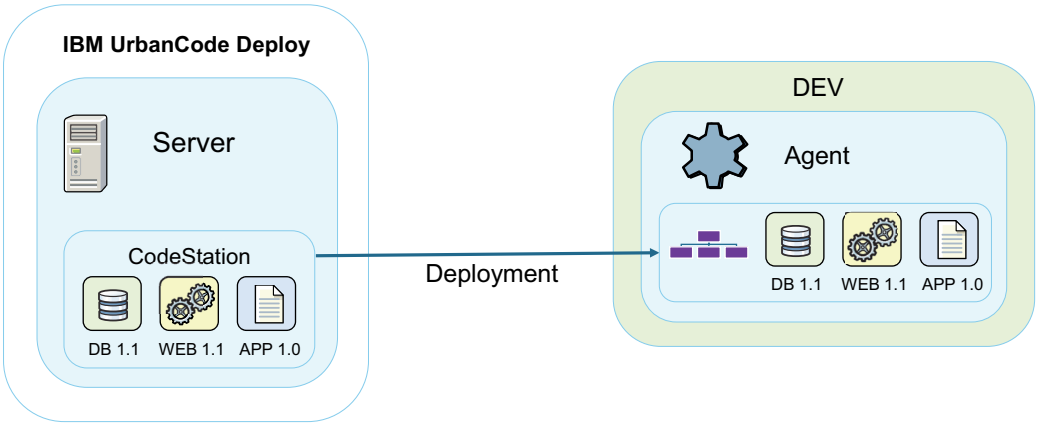


A deployment is the promotion of an application from one environment to the next

As mentioned in the first unit, a deployment is the promotion of an application from one environment to the next.

IBM Training

A deployment configures and installs a deployment package in a specific environment

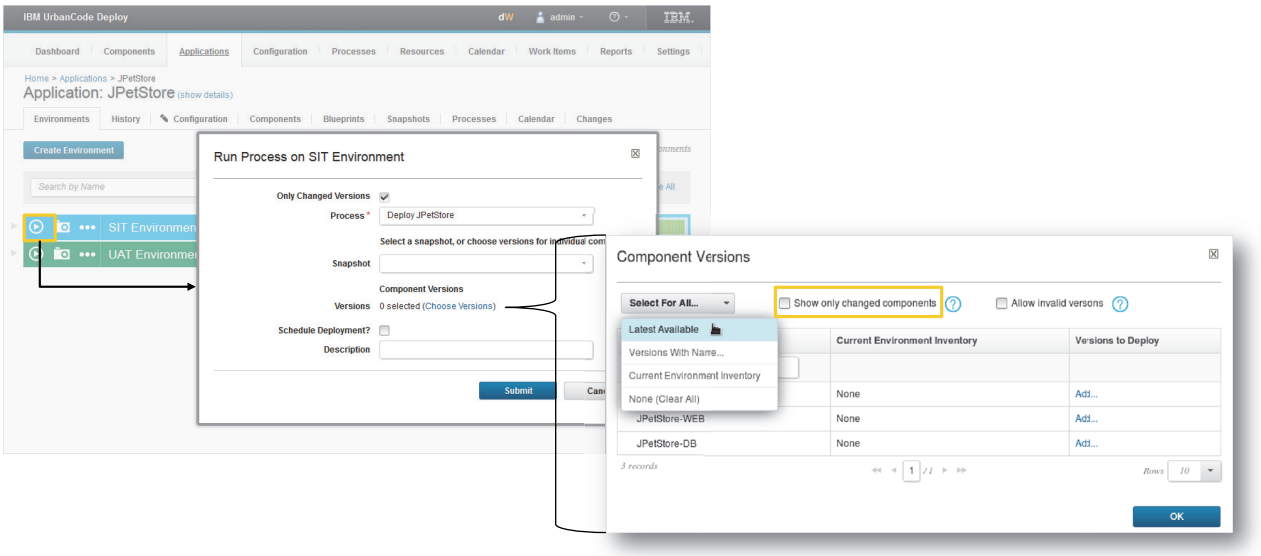


A deployment configures and installs a deployment package in a specific environment

The IBM UrbanCode Deploy server provides a file transfer capability for deploying files to target servers. A deployment involves selecting component versions from a repository, moving those components to the target server, and completing the required tasks for bringing those parts of the application online.

IBM Training

You run a deployment by running an application process in a target environment



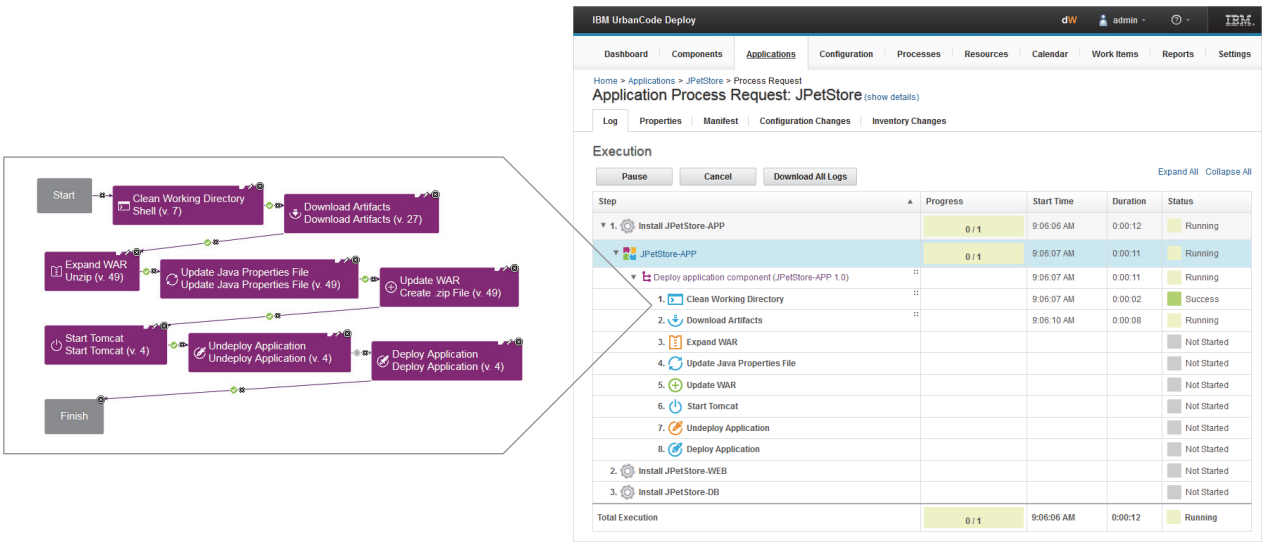
You run a deployment by running an application process in a target environment

When you start a deployment, you identify the application process and the component versions to deploy. You can select any process that is defined for the parent application, and you can use any components that are associated with the application.

If you want to deploy only the changed versions since the previous deployment, select the **Only Changed Versions** check box. If you clear this check box, all of the component versions that you select are deployed, even if they are already in the inventory.

IBM Training

The application process request shows each step in the process

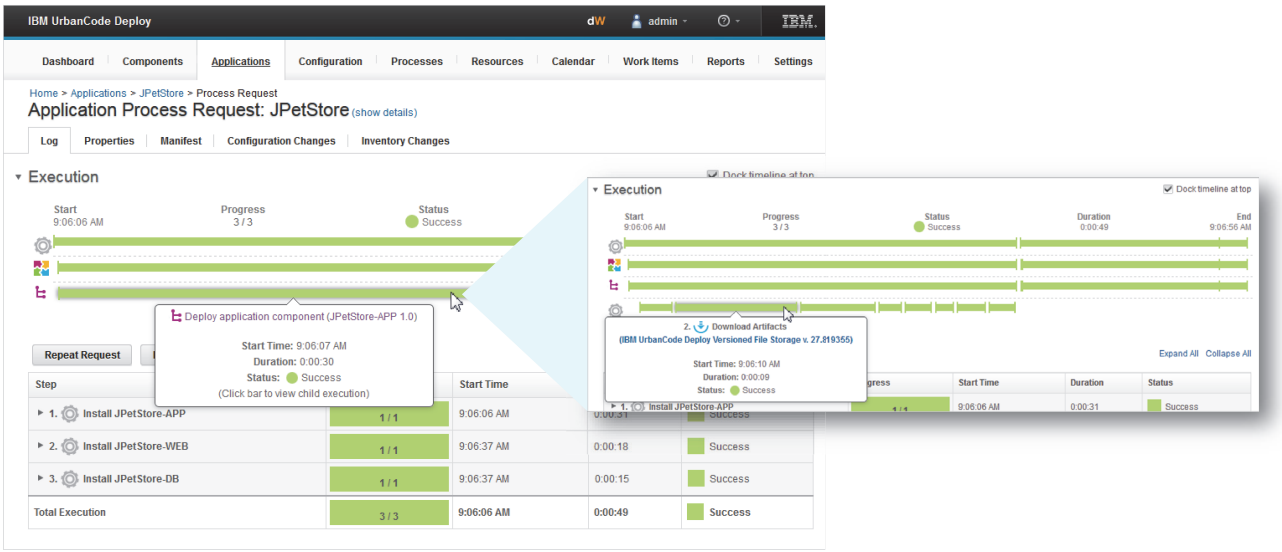


The application process request shows each step in the process

When an application process starts, the Application Process Request page displays information about the deployment status. Notice that the steps in the log are the same as the component process.

IBM Training

The Execution section displays the deployment details



The Execution section displays the deployment details

After a process finishes, the Execution section shows the status. You can click the bars to get more details about the processes and steps.

IBM Training

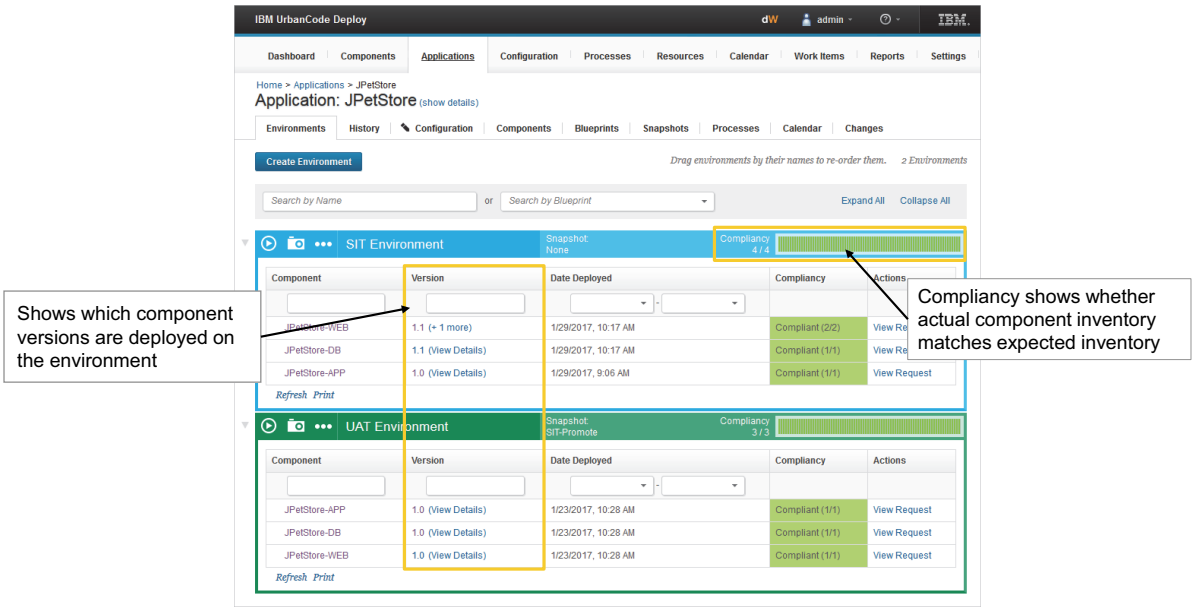
Topics

- Deploying to an environment
- Examining inventory and deployment information
- Deploying applications with snapshots

Topic: Examining inventory and deployment information

IBM Training

Inventory and history are retained for environments and resources



Inventory and history are retained for environments and resources

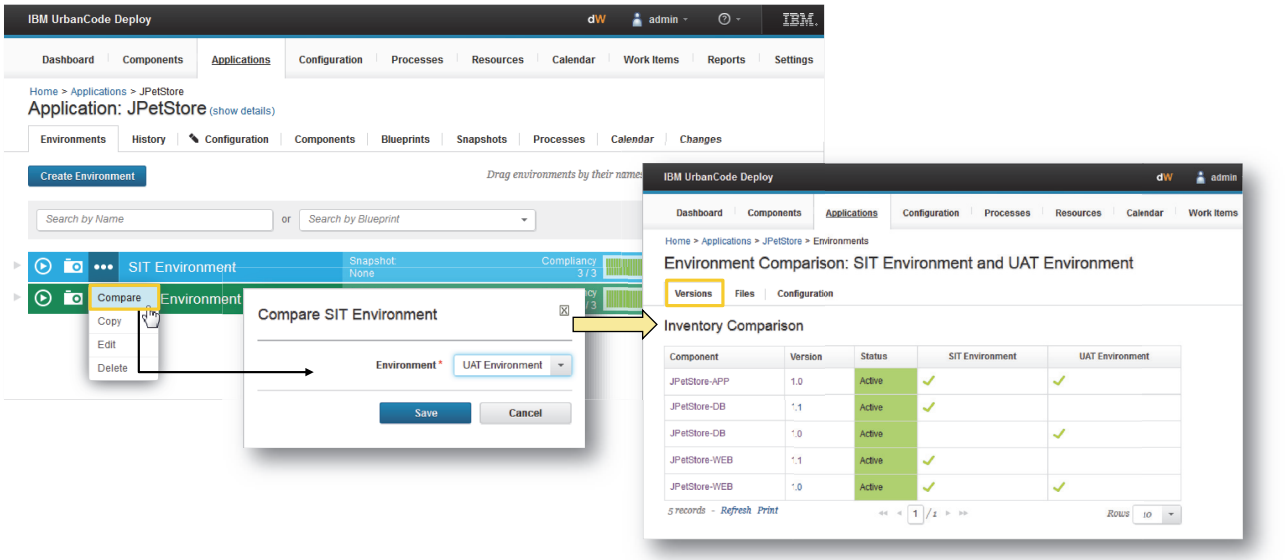
From the **Environments** tab of an application, you can see an inventory system. The inventory system tracks the specified state of each environment and what has been successfully deployed to each target resource. Compliancy indicates whether the environment contains those component versions.

The inventory and compliance for an environment are updated each time that you run an application process or deploy a snapshot to the environment.

In this example, three deployable components are deployed to two environments. You can see that the SIT environment has newer versions.

IBM Training

You can compare inventories between environments

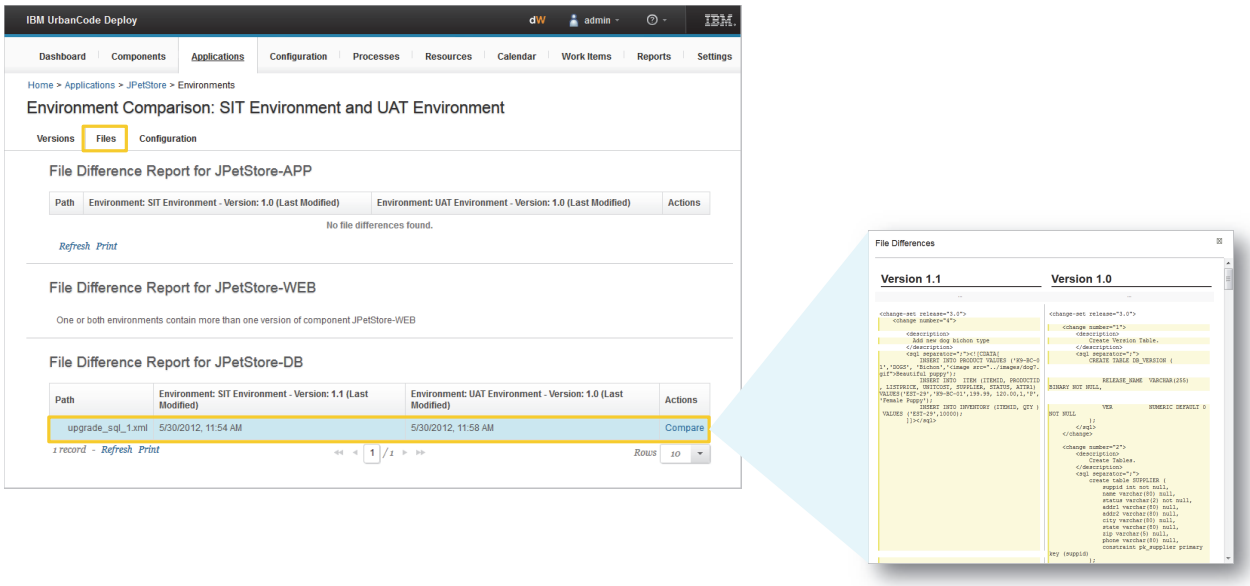


You can compare inventories between environments

IBM UrbanCode Deploy maintains an inventory of every artifact that is deployed to each environment and also tracks the differences between them. You can compare two environments to see differences in the versions that they contain.

IBM Training

File differences are tracked between the environments

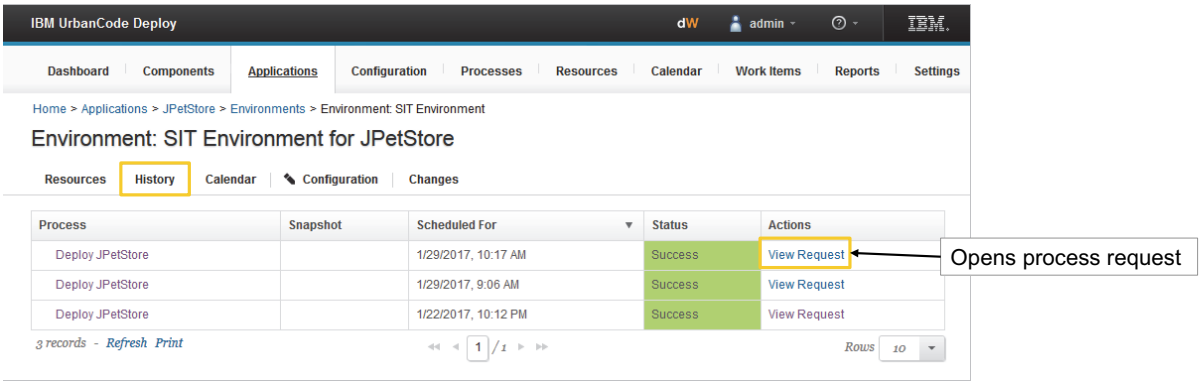


File differences are tracked between the environments

When there is a problem in production or during test, traceability makes it easier to understand what changes are going in that could have caused the problem.

IBM Training

The History tab show the process requests for a specific environment

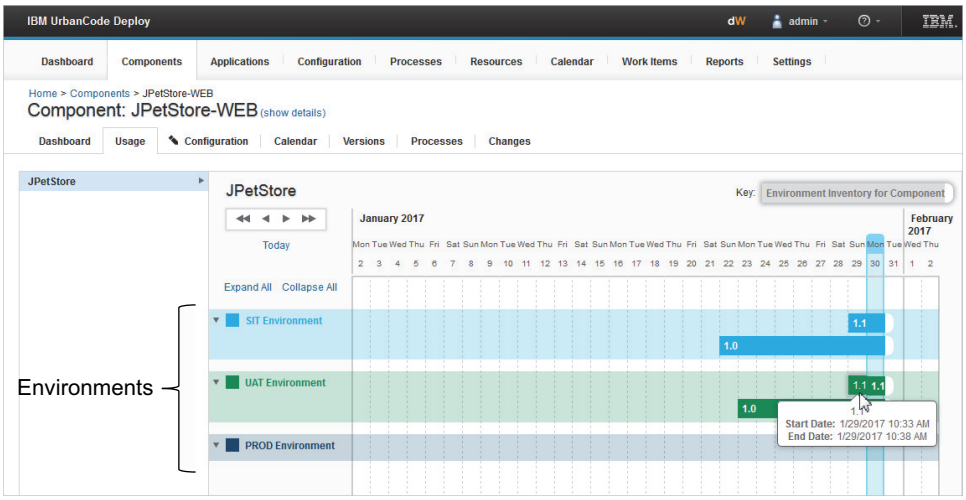


The History tab show the process requests for a specific environment

You can view the progress of the process by going to the **History** tab and opening the process request.

IBM Training

The Usage tab shows the environment inventory for each component

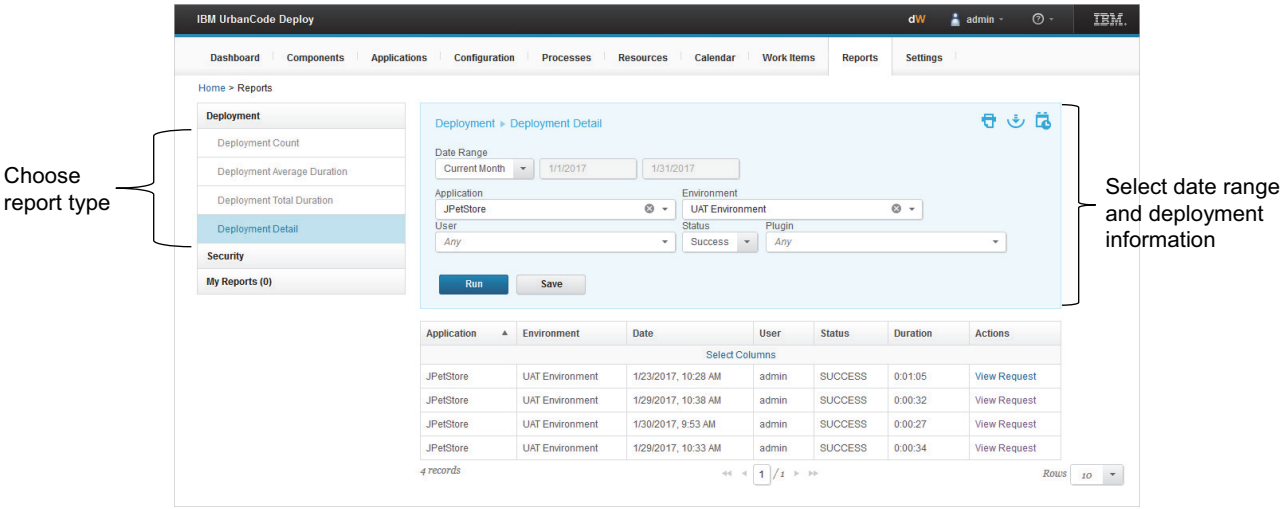


The Usage tab shows the environment inventory for each component

You can track the environment inventory for each component by clicking the component and then going to the **Usage** tab. IBM UrbanCode Deploy tracks both the version that is currently in an environment and also what was in various environments over time. This feature is helpful comparing flows through environments and identifying what changed.

IBM Training

Deployment reports contain historical information about deployments



Deployment reports contain historical information about deployments

You can filter data, print and save reports, and save search criteria with deployment reports. You can obtain information for specific reporting periods, such as the number of deployments, average deployment duration, and total duration. In this example, the Deployment Detail report shows all deployments for the specified applications, environments, and users for a certain time period.

- **Deployment count**: Provides information about the number of deployments that ran during a specified reporting period.
- **Deployment average duration**: Provides average deployment times for applications that ran during a specified reporting period.
- **Deployment total duration**: Provides total deployment times for applications that ran during a specified reporting period.
- **Deployment detail**: Shows all deployments for the specified applications, environments, and users in the specified time period.

IBM Training

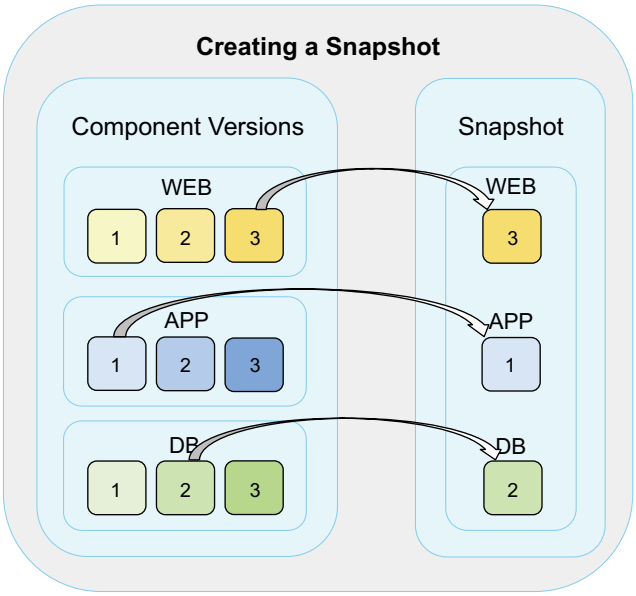
Topics

- Deploying to an environment
- Examining application environment inventory
- Deploying applications with snapshots

Topic: Deploying applications with snapshots

IBM Training

A snapshot is collection of specific versions of components and processes



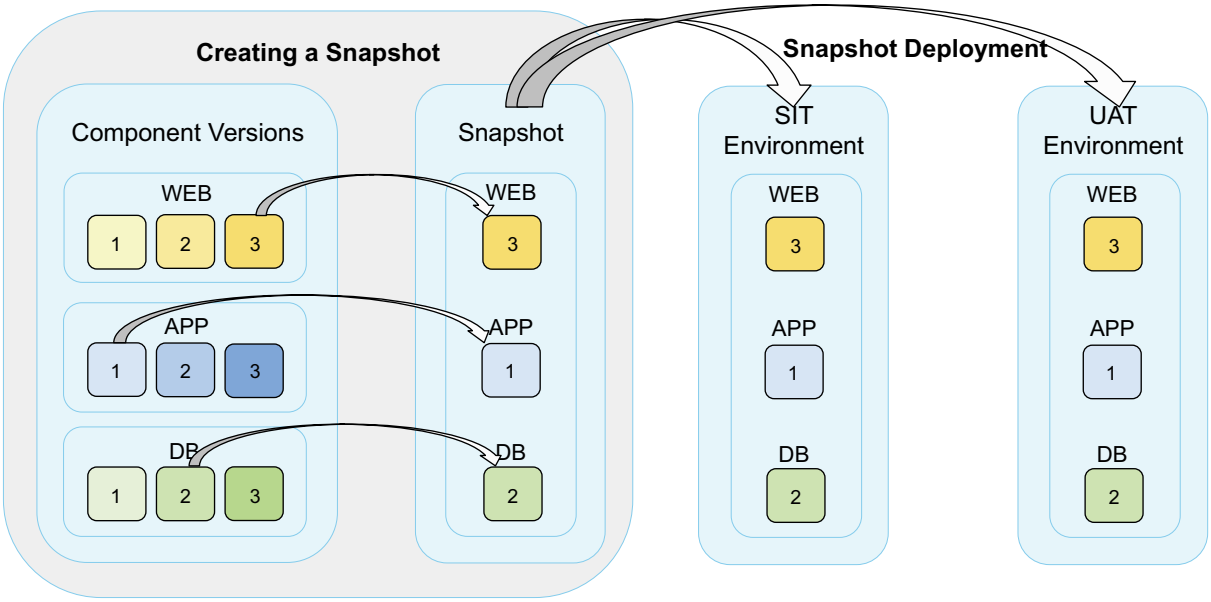
A snapshot is collection of specific versions of components and processes

Snapshots are models that you create before you deploy the application. A snapshot specifies the exact version for each component in the application. When a snapshot is created, information is gathered about the application, including the component versions, for an environment.

Typically, the snapshot is generated in an environment that has no approval gates. This kind of environment is called an uncontrolled environment.

IBM Training

Snapshots ensure consistent deployments from one environment to another

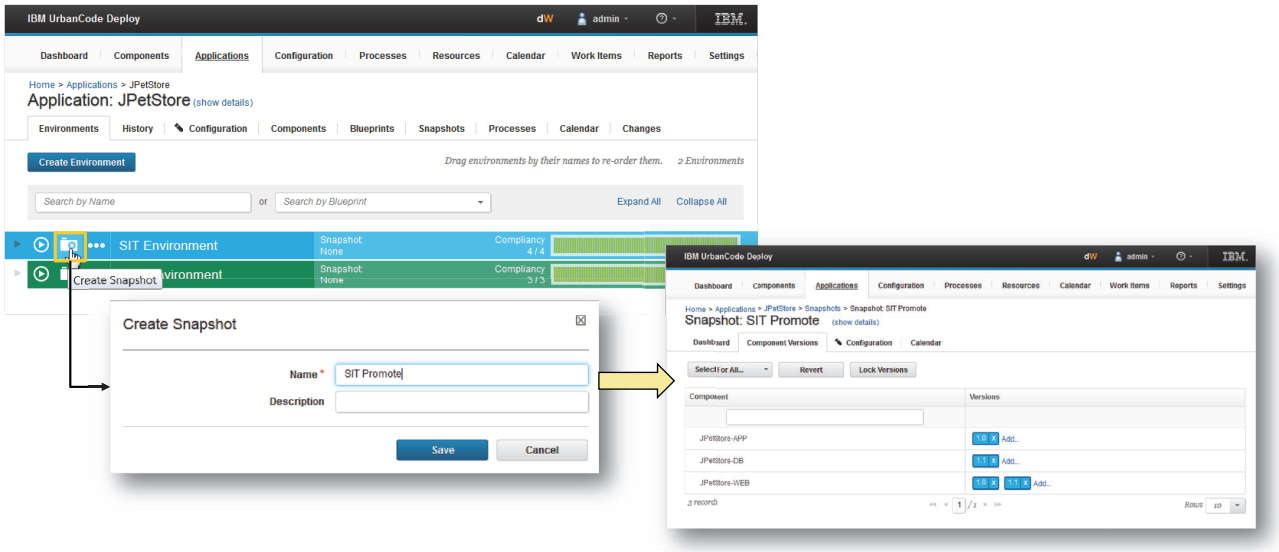


Snapshots ensure consistent deployments from one environment to another

You use snapshots to ensure the same exact deployment happens from one environment to another. If you have several components, selecting a version for each one, each time, could introduce errors. A snapshot is a single deployable item that you can push from environment to environment. After all the appropriate stages and approvals for a snapshot are complete, the snapshot is pushed to production.

IBM Training

A snapshot captures a picture of the application's current state

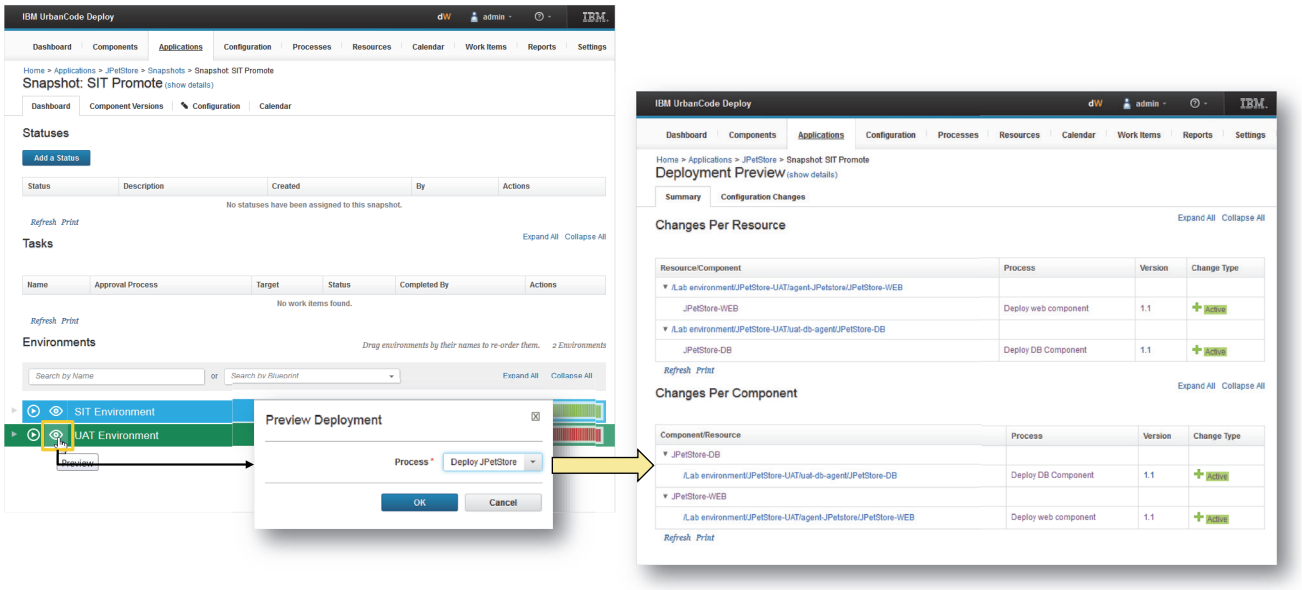


A snapshot captures a picture of the application's current state

To create a snapshot, you choose the component versions and processes to include in the snapshot.

IBM Training

Previewing deployments for impact analysis

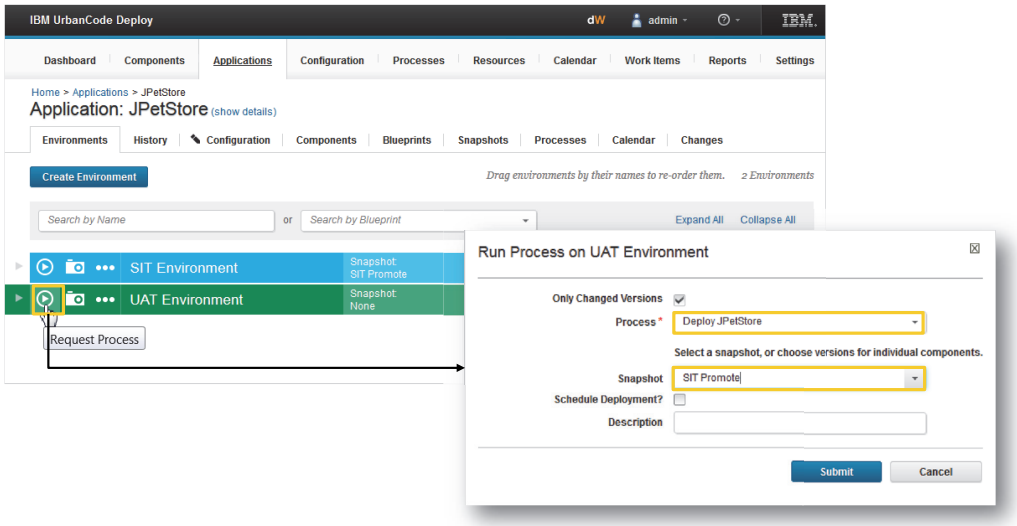


Previewing deployments for impact analysis

You can preview an application snapshot before you deploy it to see the upcoming changes in resources and properties.

IBM Training

Deploying a snapshot deploys the components in the snapshot

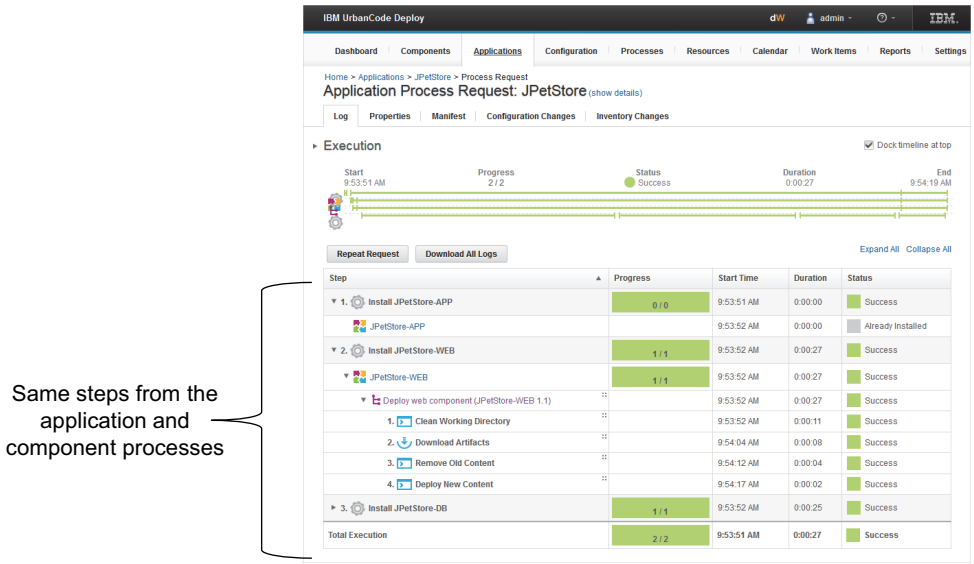


Deploying a snapshot deploys the components in the snapshot

When you have a snapshot, you select both the environment to deploy to and the process to use for the deployment.

IBM Training

Application and component processes are the same

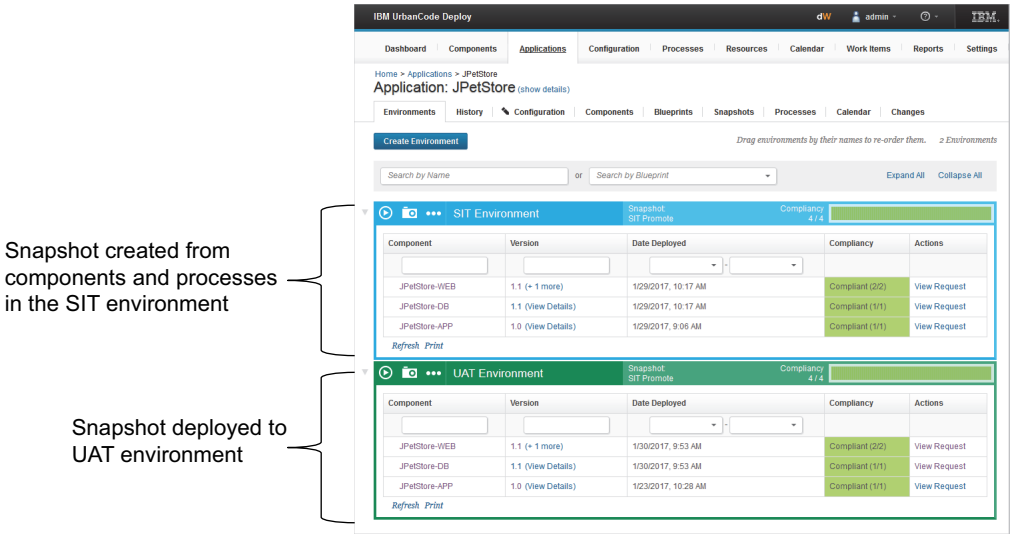


Application and component processes are the same

The deployment looks the same as the original deployment each time the snapshot is used.

IBM Training

The inventory is updated after running the snapshot

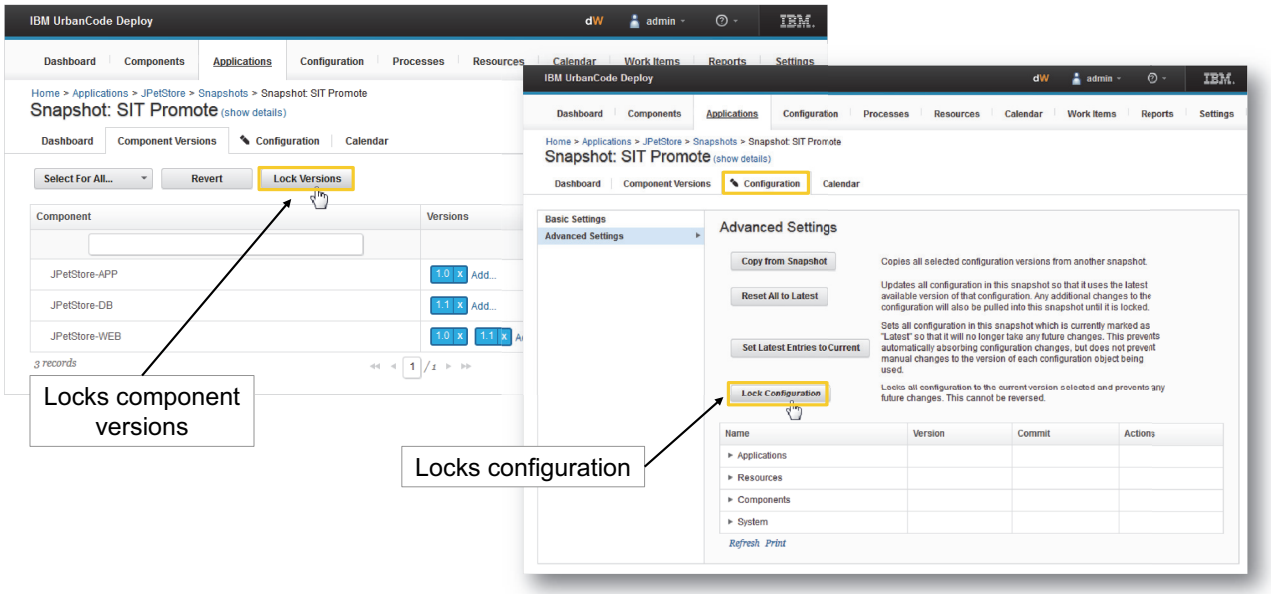


The inventory is updated after running the snapshot

The inventory is immediately updated and can be viewed alongside of other environments in the application.

IBM Training

Environments can be configured to lock snapshot versions or configuration

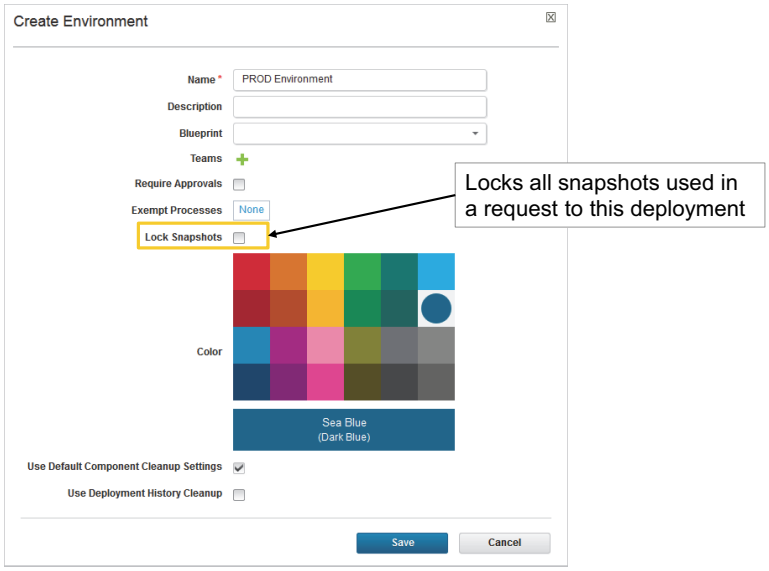


Environments can be configured to lock snapshot versions or configuration

Environments can be configured to specifically lock either snapshot versions or the snapshot configuration, or both, when snapshots are deployed to the environment.

IBM Training

You can lock a snapshot at environment creation



You can lock a snapshot at environment creation

By default, the lock setting for both component versions and snapshot configurations are determined by the option selected in the Lock Snapshots field on the Create Environment window. All snapshots used in a request to this deployment will be locked to prevent further changes.

IBM Training

Unit summary

- A deployment is the promotion of an application from one environment to the next
- You run a deployment by running an application process in a target environment
- You can compare two environments to see differences in the versions that they contain
- Snapshots are models that you create before you deploy the application
- A snapshot specifies the exact version for each component in the application

Unit summary

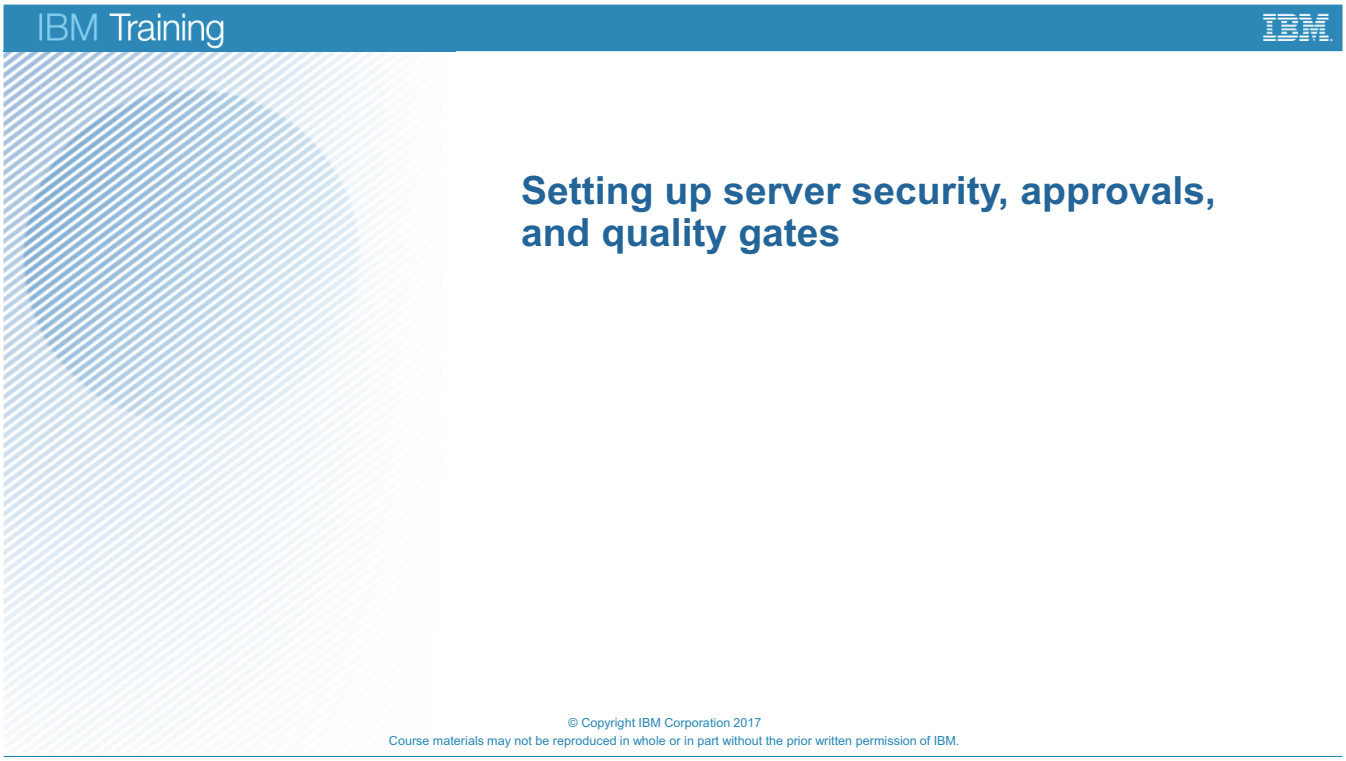
IBM Training

Exercises: Deploying applications to target environments

- 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

Exercises: Deploying applications to target environments

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



IBM UrbanCode Deploy uses a flexible team-based and role-based security model that maps to your organizational structure. In this unit, you learn how to define and configure roles, set up approvals and notifications, and use quality statuses and gates.

IBM Training

Objectives

- Describe the significance of a system team
- Create authorization realms, user groups, and teams
- Create and define roles and permissions
- Set up notifications and approvals
- Implement statuses and gates to ensure testing quality

Unit objectives

IBM Training

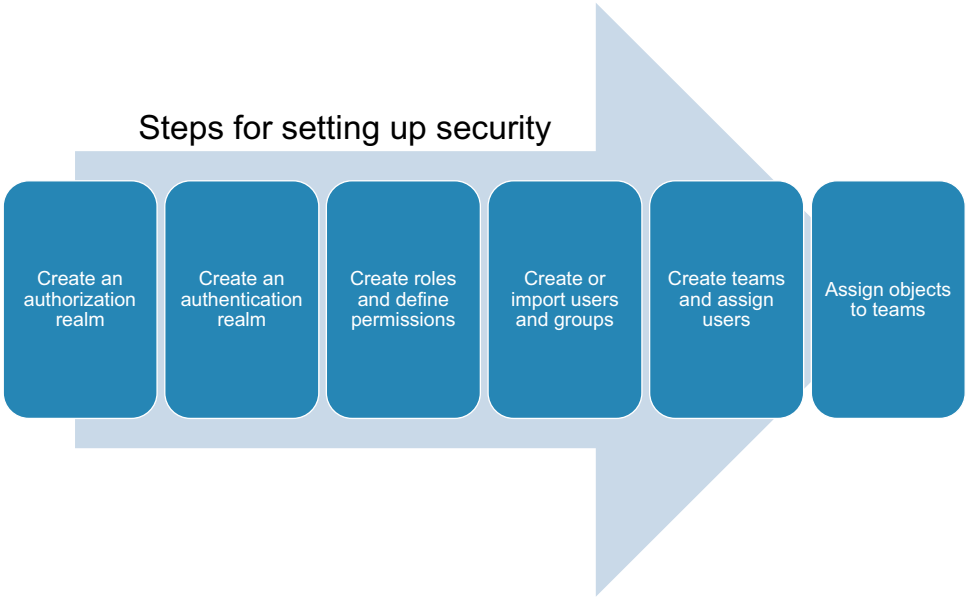
Topics

- Configuring server security
- Setting up notifications
- Setting up an approval process
- Implementing statuses and gates to improve quality

Topic: Configuring server security

IBM Training

The security system for the server consists of an authentication realm, authorization realm, roles, and teams



The security system for the server consists of an authentication realm, authorization realm, roles, and teams

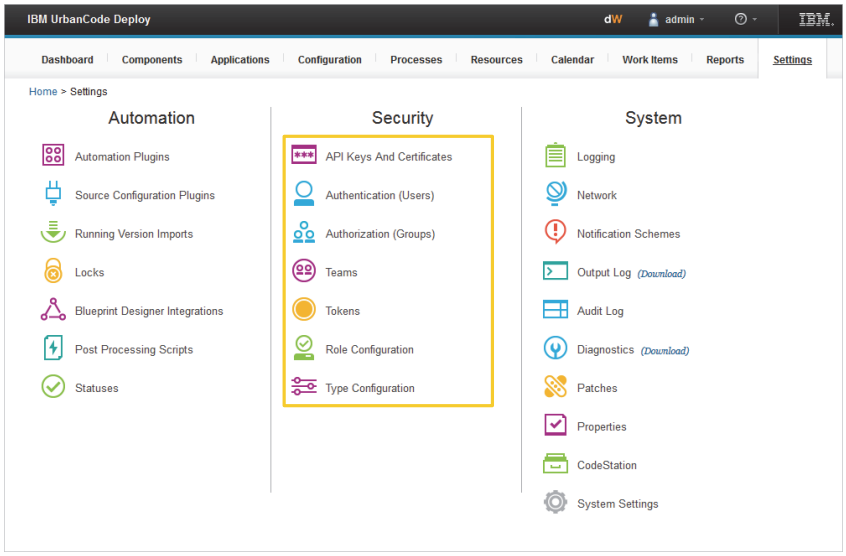
From a high level, the security system for the server consists of an authorization realm, authentication realm, roles, and teams.

The authentication realm verifies the identity of the user or system that is trying to log on to the IBM UrbanCode Deploy server.

The authorization realm manages user groups. Roles manage permissions. Teams bring together users with roles and specify which objects the team can access.

IBM Training

You can access the security system from any of these links on the Settings tab

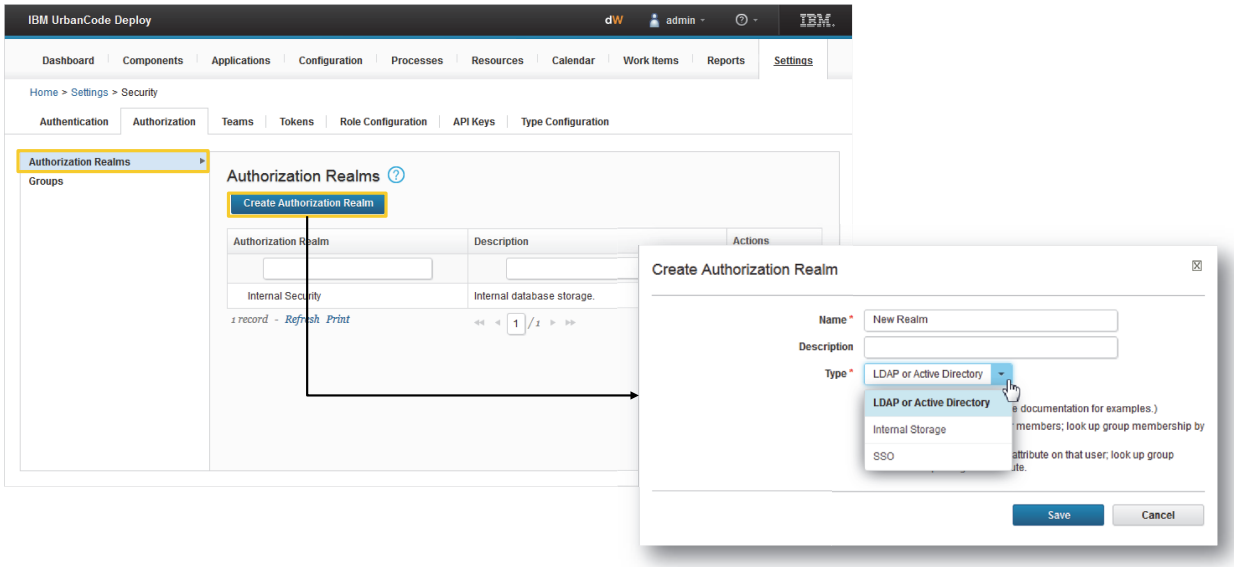


You can access the security system from any of these links on the Settings tab

You access the security features from the **Settings** tab.

IBM Training

Authorization realms manage user groups



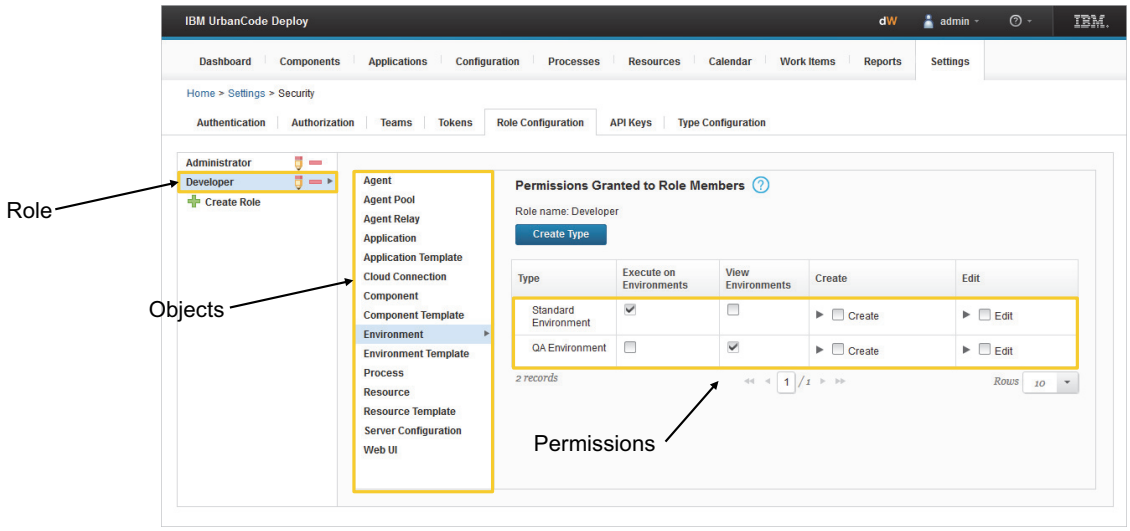
Authorization realms manage user groups

Authorization realms verify the identity of a user or system attempting to login. There are three available types for the server, internal storage authorization, an LDAP authorization realm, and an SSO authorization realm:

- **Internal storage authorization**: Does not retrieve users from any external source. Instead, you add users to internal storage authorization realms manually.
- **LDAP authorization realm (Lightweight Directory Access Protocol)**: LDAP is a widely used protocol for accessing distributed directory information over IP networks. It uses an external LDAP server for authorization.
- **SSO authorization realm (single sign-on)**: Uses an external server for authorization and allows a user to sign on with one set of credentials for multiple applications.

IBM Training

A role is a set of permissions

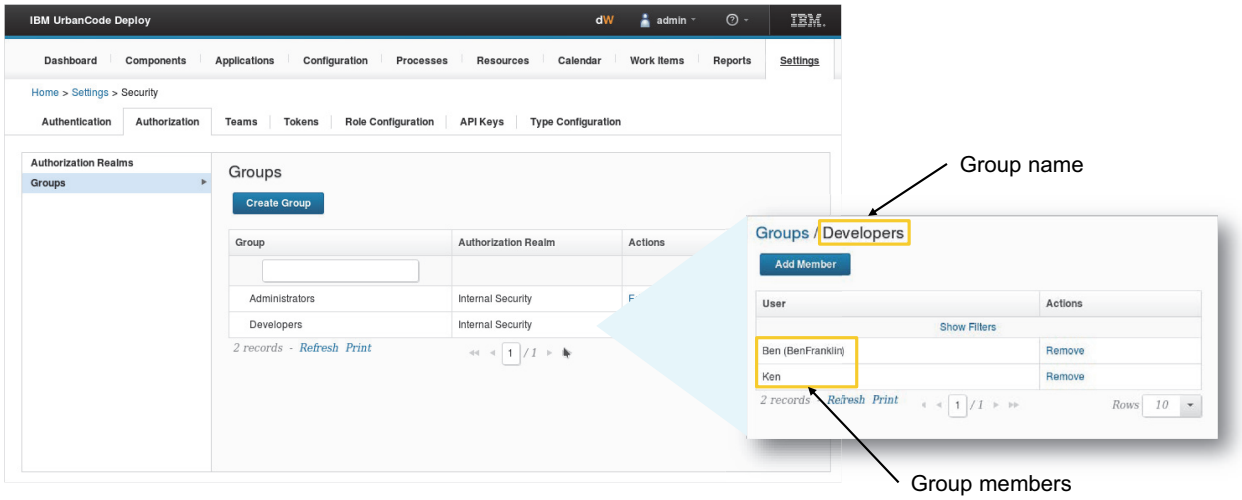


An authentication realm is a source of information about users

Authentication realms manage users and determine user identity within authorization realms for the server. Users can be created manually or imported from external systems, such as LDAP.

IBM Training

Groups identify users with the same role

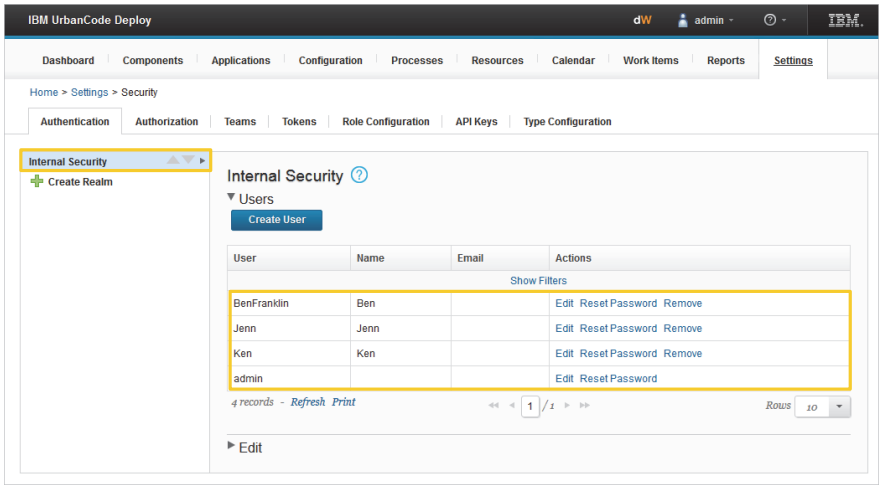


A role is a set of permissions

Permissions define what can be done; roles define who can do it. Separate permissions are available for each type of object on the server, including components, applications, and environments.

IBM Training

An authentication realm is a source of information about users

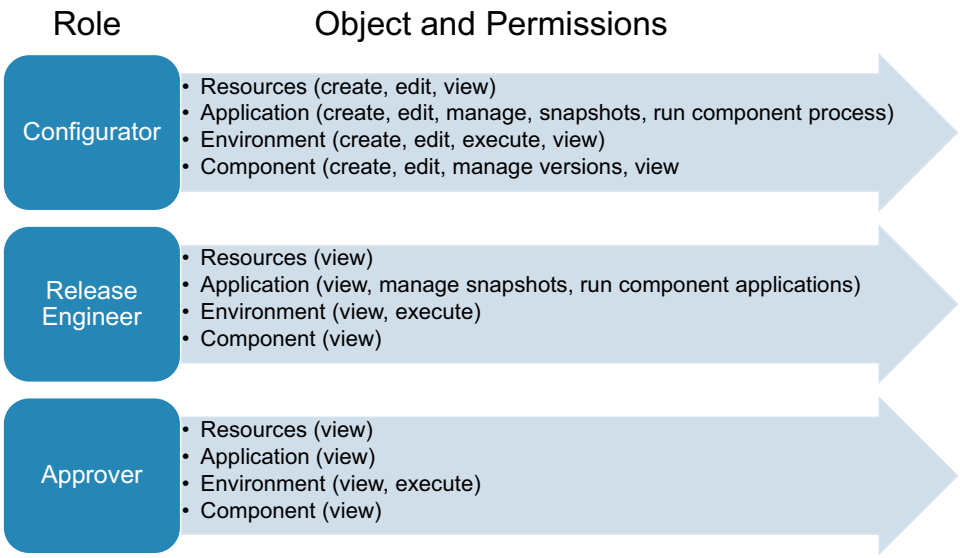


Groups identify users with the same role

Groups are containers that grant permissions to multiple users; members automatically share a group's permissions.

IBM Training

Keep the roles simple but ensure sufficiency for performing work

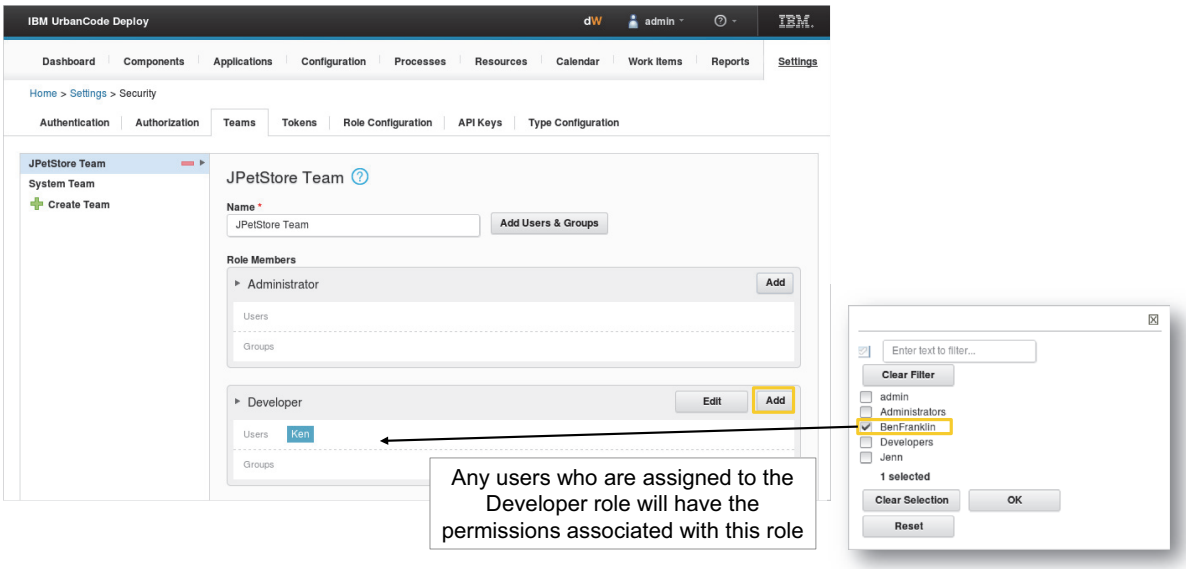


Keep the roles simple but ensure sufficiency for performing work

When defining the roles for your team, start by keeping the roles simple, but make them sufficient to perform the appropriate work.

IBM Training

Users are granted permissions by being assigned to teams

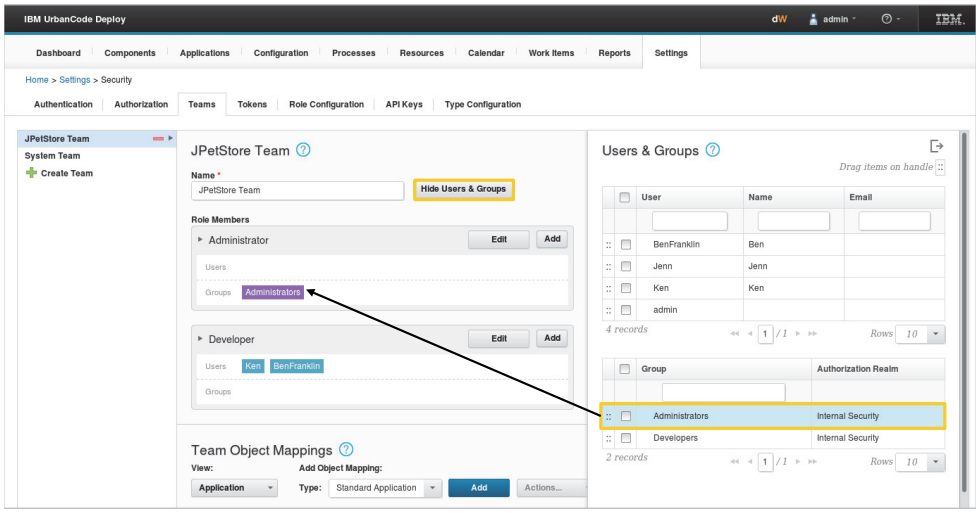


Groups grant permissions to multiple users

You can grant a group of users the same role and permissions by dragging the group to the role.

IBM Training

Groups grant permissions to multiple users



Users are granted permissions by being assigned to teams

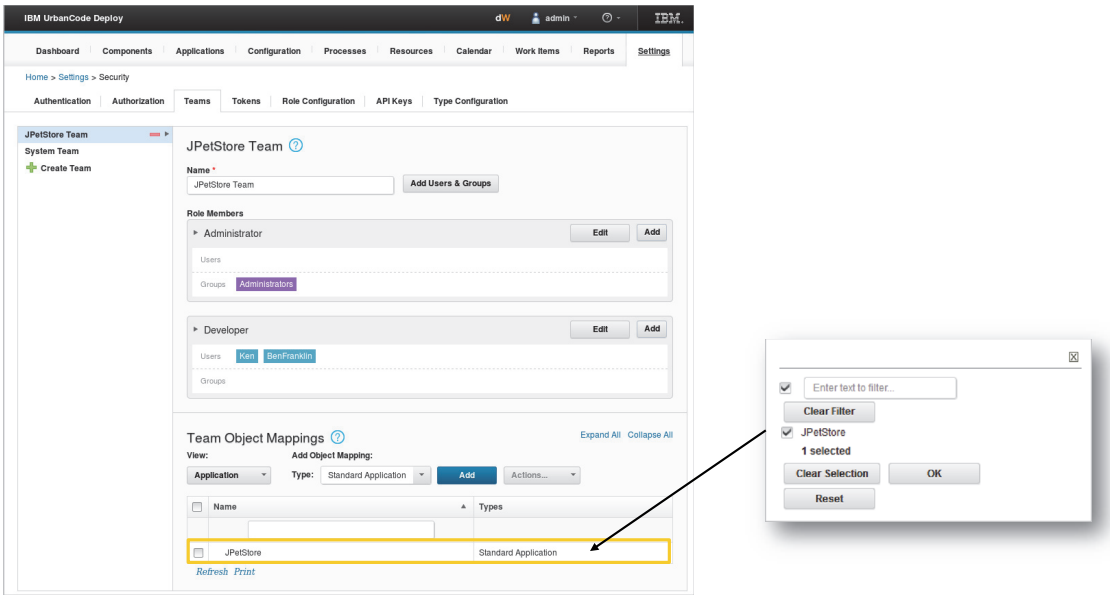
You can also assign individuals to teams to give them the proper permissions.

Teams provide two important functions:

- First, a team provides the mechanism to assign roles to users. When a user is assigned a role, all permissions that are granted to the role are automatically granted to the user, but only for objects that the team has access to.
- Second, teams secure objects such as applications, components, and environments. When a team is attached to an object, only team members with the appropriate permissions can interact with the affected resource.

IBM Training

Restrict permissions by assigning teams to objects

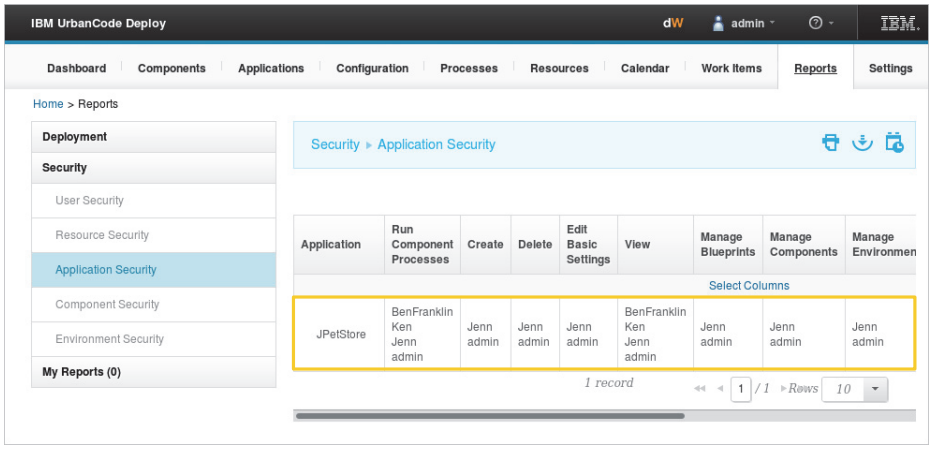


Restrict permissions by assigning teams to objects

You can assign an object, such as an application, to a team in the Team Object Mappings section.

IBM Training

Security reports provide information about user roles and privileges



Security reports provide information about user roles and privileges

Security reports track information about roles and permissions. The application security report shows the user roles and permissions that are set for applications.

IBM Training

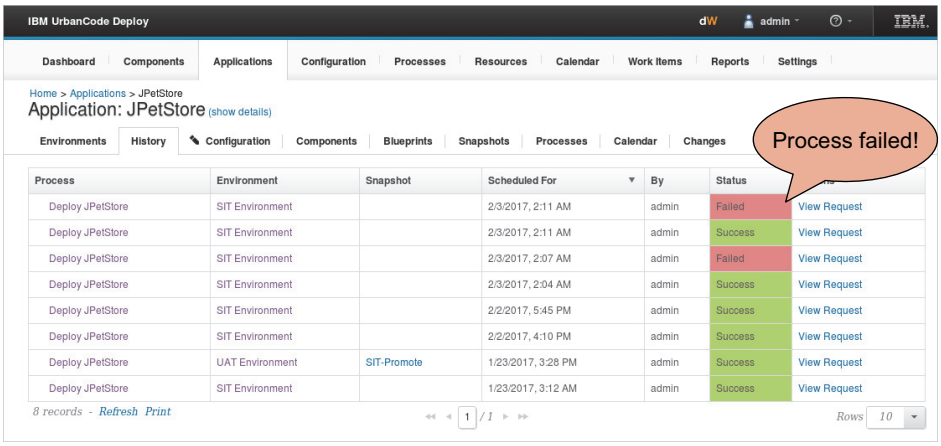
Topics

- Configuring server security
- Setting up notifications
- Setting up an approval process
- Implementing statuses and gates to improve quality

Topic: Setting up notifications

IBM Training

Notifications can be sent when user-defined trigger events occur

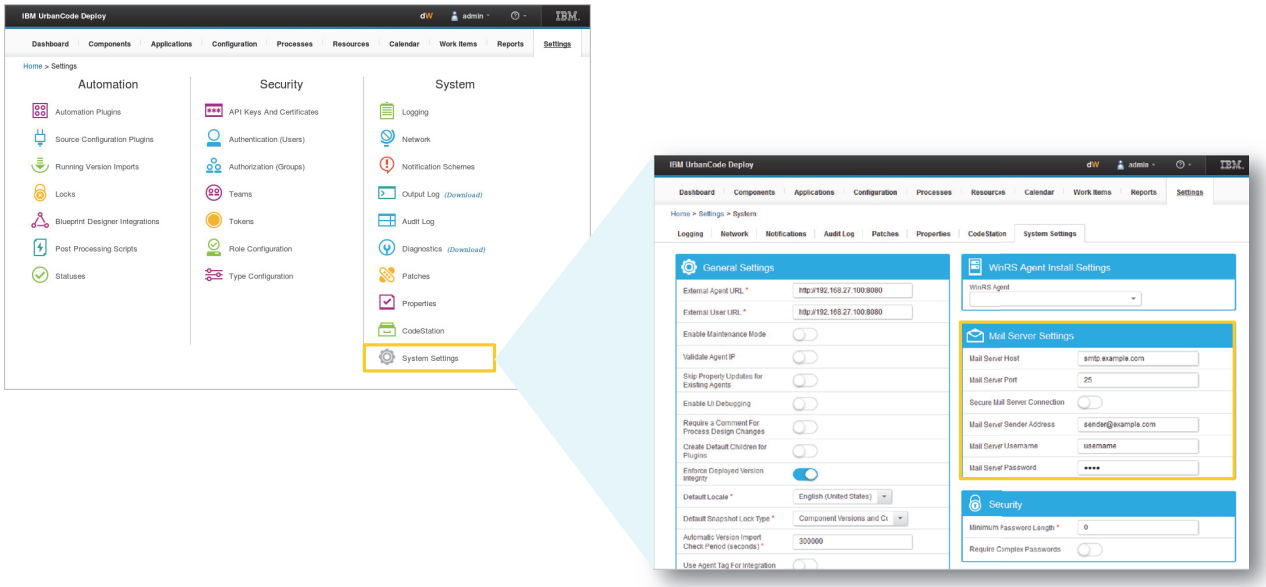


Notifications can be sent when user-defined trigger events occur

Notifications are emails that are sent whenever user-defined trigger events on the server occur, such as when a deployment finishes or an approval is required.

IBM Training

IBM UrbanCode Deploy requires an external SMTP mail server to send notifications

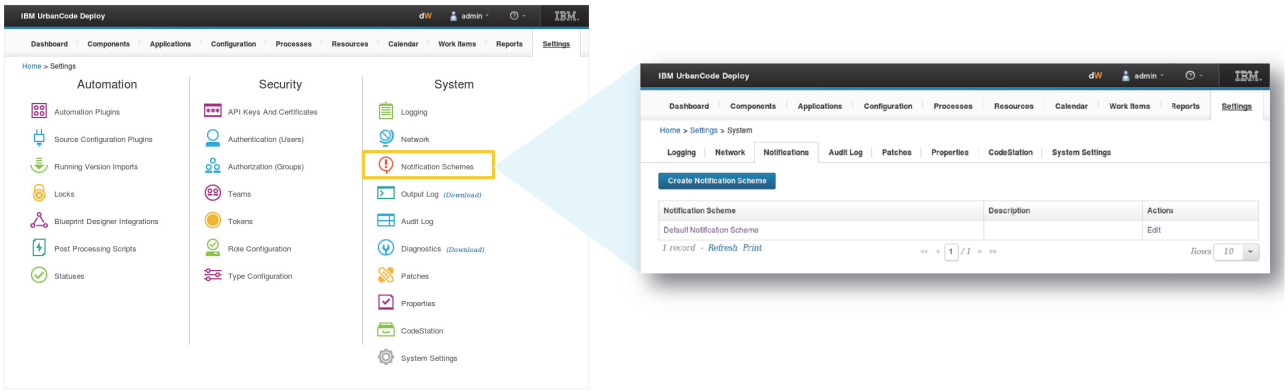


IBM UrbanCode Deploy requires an external SMTP mail server to send notifications

Before you can send notifications, users must have email addresses attached to them on the server; email addresses are not automatically imported.

IBM Training

Notifications are sent to appropriate parties based on scheme

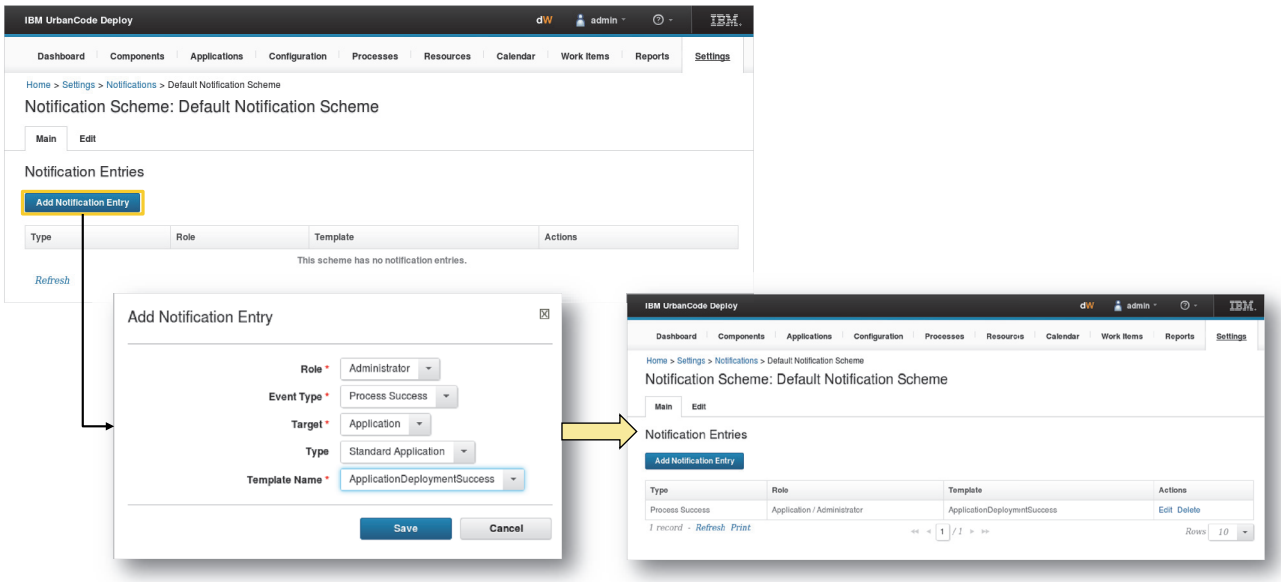


Notifications are sent to appropriate parties based on scheme

To set up notifications, open the Notifications pane by clicking **Settings** > **Notification Schemes**.

IBM Training

Notifications include triggering events and roles

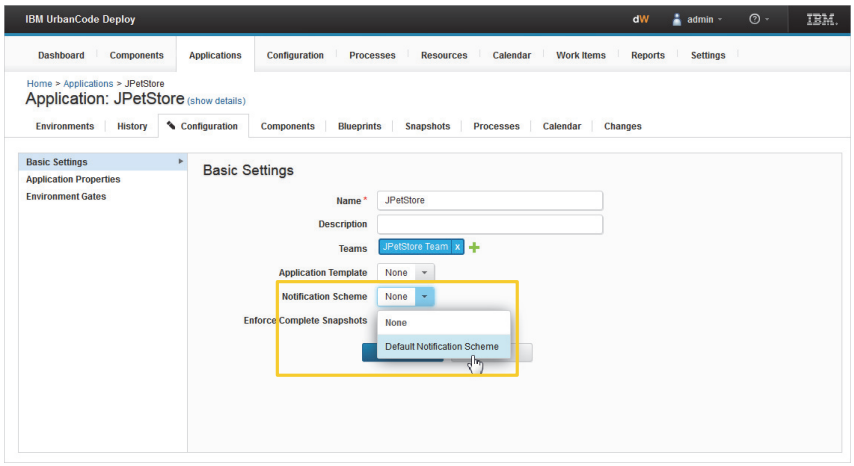


Notifications include triggering events and roles

When you set up the notification, you determine which users receive notification by selecting both the triggering events and the role. The role is inherited from the security system.

IBM Training

Notifications are sent to appropriate parties based on scheme



Notifications are sent to appropriate parties based on scheme

You can assign notification schemes to applications and environments at creation or in the **Configuration** tab.

IBM Training

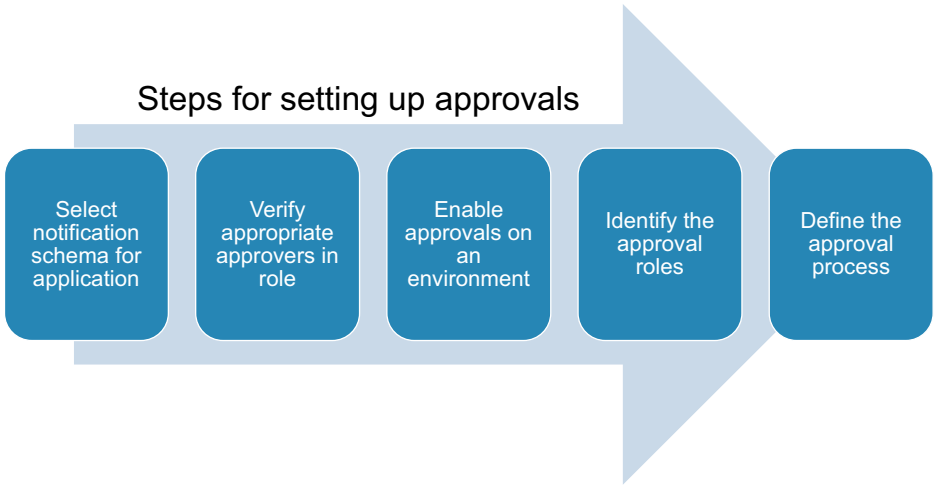
Topics

- Configuring server security
- Setting up notifications
- Setting up an approval process
- Implementing statuses and gates to improve quality

Topic: Setting up an approval process

IBM Training

Approvals provide more visibility into deployments for audit trails

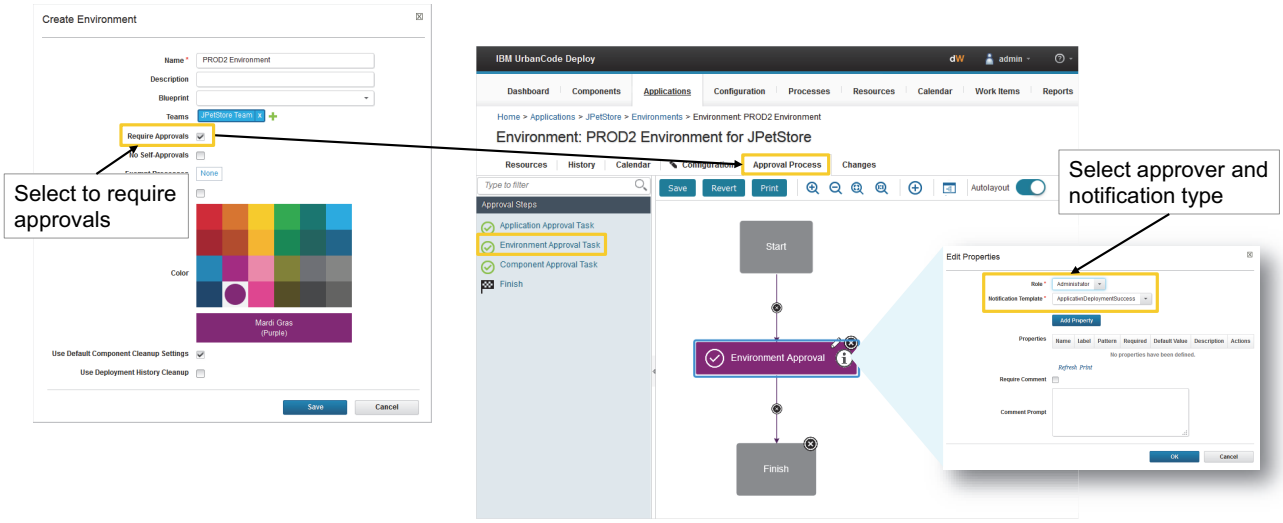


Approvals provide more visibility into deployments for audit trails

An approval specifies the job that needs approval and the role of the approver. The main tasks for setting up an approval process include selecting the notification scheme, enabling approvals on an environment, and defining the approval process.

IBM Training

An approval process specifies the job that needs approval and the role of the approver

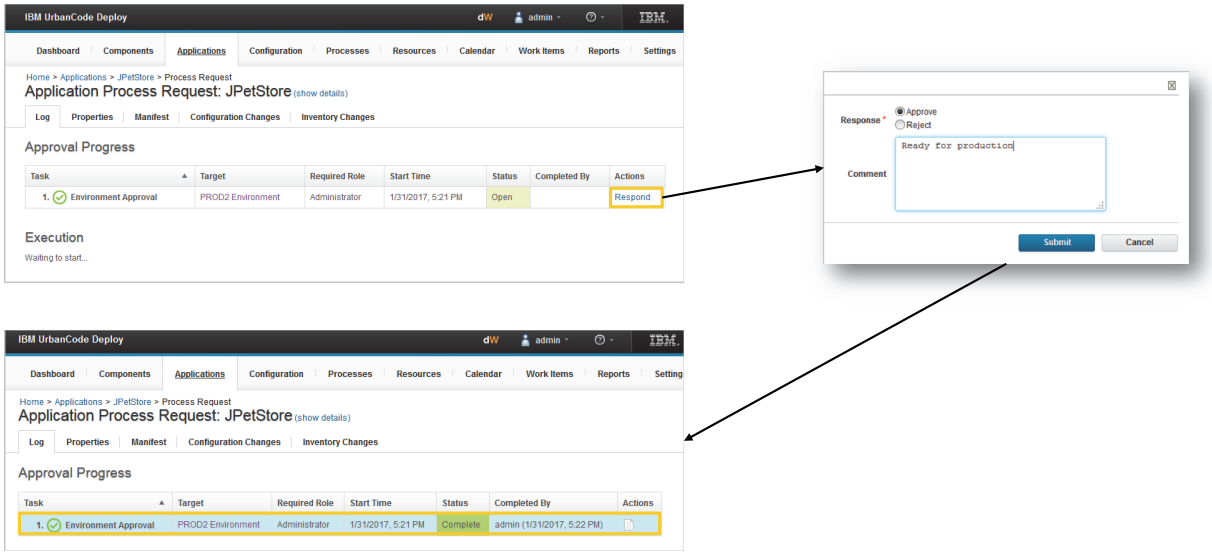


An approval process specifies the job that needs approval and the role of the approver

To ensure that components cannot be deployed to the environment without first being approved, select the **Require Approvals** check box. This option enforces an approval process before the deployment can be deployed to the environment. Approvals are usually attached to environments.

IBM Training

When a request for approval is made, the users with the corresponding role are notified through email

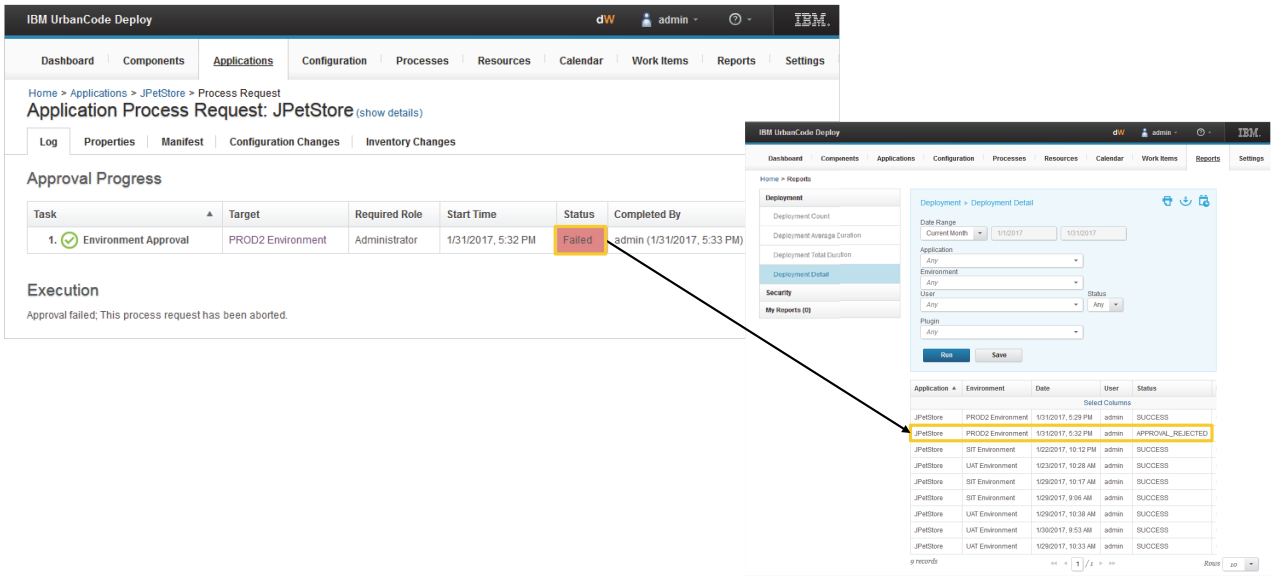


When a request for approval is made, the users with the corresponding role are notified through email

The approver can approve or reject a deployment and also provide comments to the decision.

IBM Training

The Deployment Details report tracks approval status



The Deployment Details report tracks approval status

You can see the status of the approvals by clicking **Reports** > **Deployment Details**.

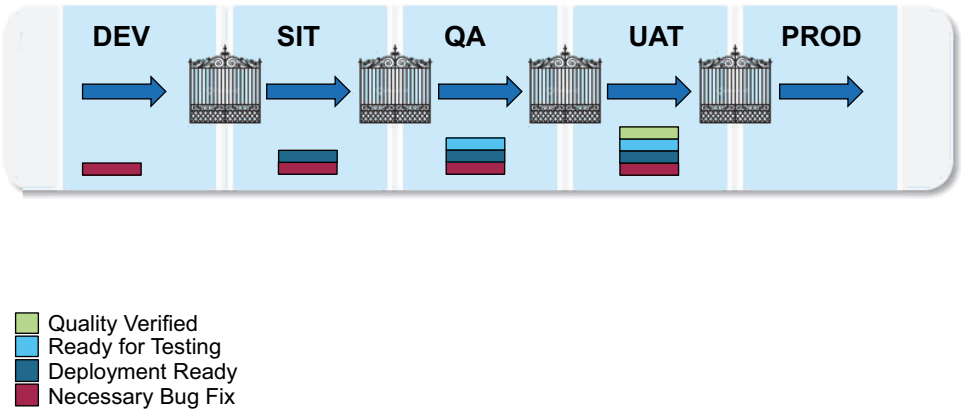
IBM Training

Topics

- Configuring server security
- Setting up notifications
- Setting up an approval process
- Implementing environment gates to improve quality

Topic: Implementing environment gates to improve quality

Gates specify the conditions that must be met for deployment

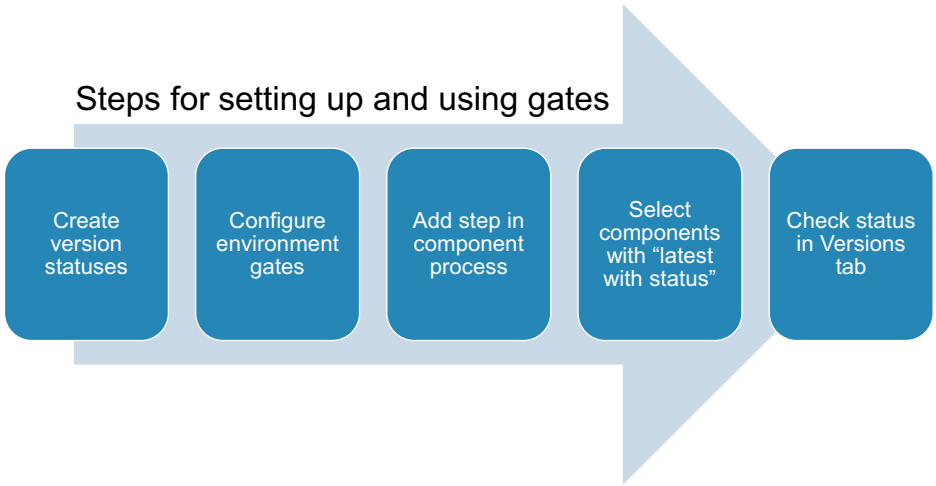


Gates specify the conditions that must be met for deployment

An environment gate is a requirement that must be met before component versions can be deployed to an environment. A version status adds a tag to a version to indicate that it has met those requirements and can be deployed to the environment.

IBM Training

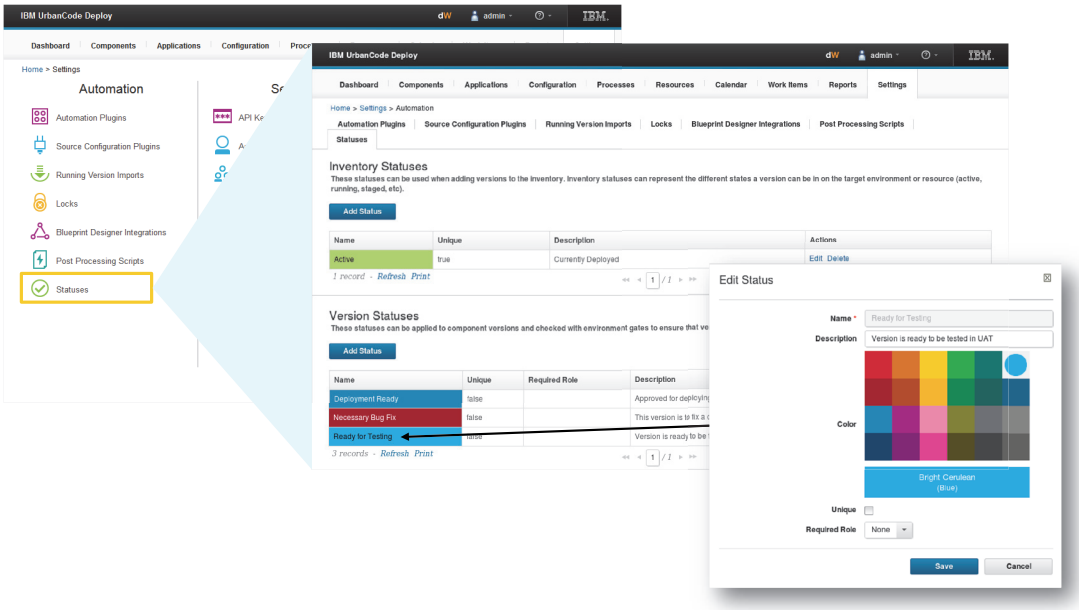
Using statuses and gates requires initial configuration



Using statuses and gates requires initial configuration

When you set up gates, you first need to create the version statuses and configure the environment gates. Then, you add a step in the component process to check the version status. When you select the components for your deployment, choose **Latest with Status** before deployment. This choice deploys only those versions that have met the appropriate status.

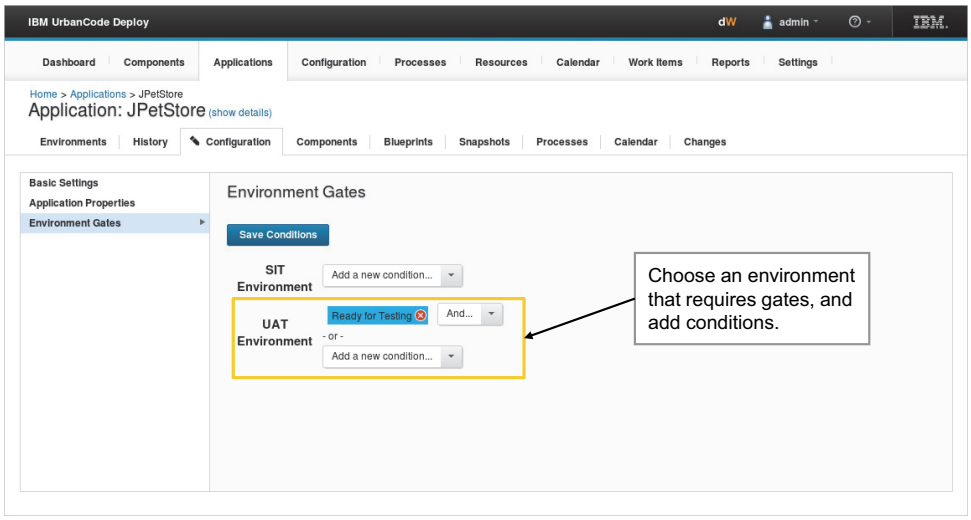
Version statuses identify when component versions meet criteria



Version statuses identify when component versions meet criteria

On the **Statuses** tab, a table shows the currently configured version statuses. When you create a status, you can set the required role field as a specific role to restrict the ability to assign a status to a version.

Environment gates check the version tags on component versions



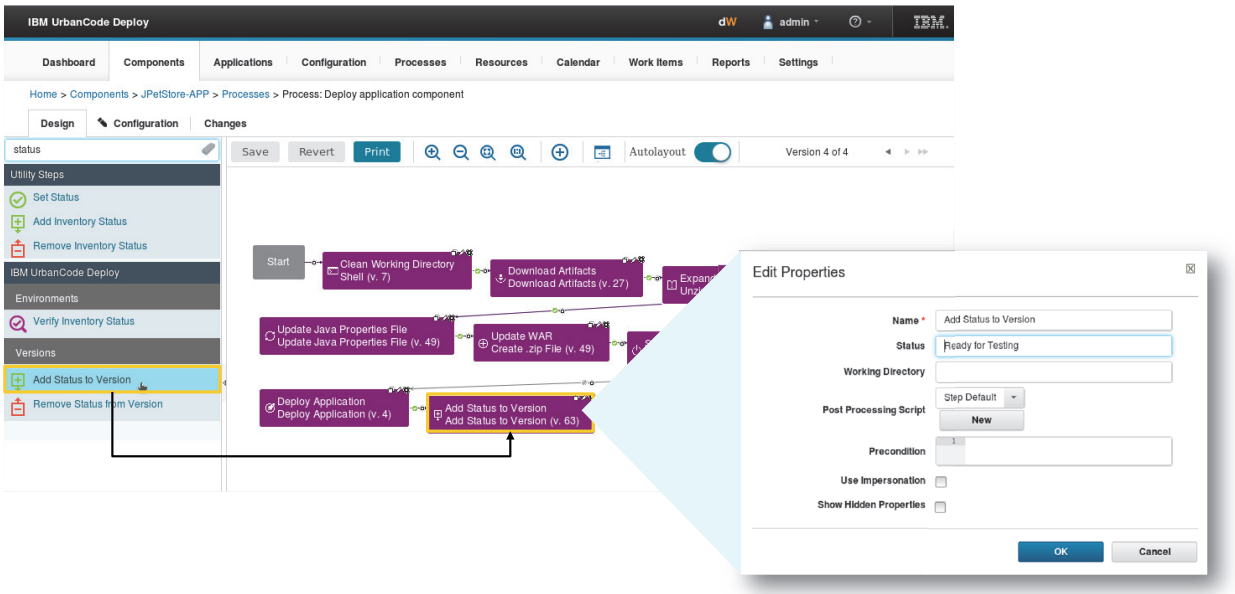
Environment gates check the version tags on component versions

After you have your statuses configured, you can set up the application gates on your environments from the application's **Configuration** tab.

For each environment, you can set version statuses that the version must have before it can be deployed to that environment. In this example, only versions that have the status Ready for Testing are allowed to be deployed to the UAT environment.

IBM Training

Statuses can be added to versions automatically from component processes

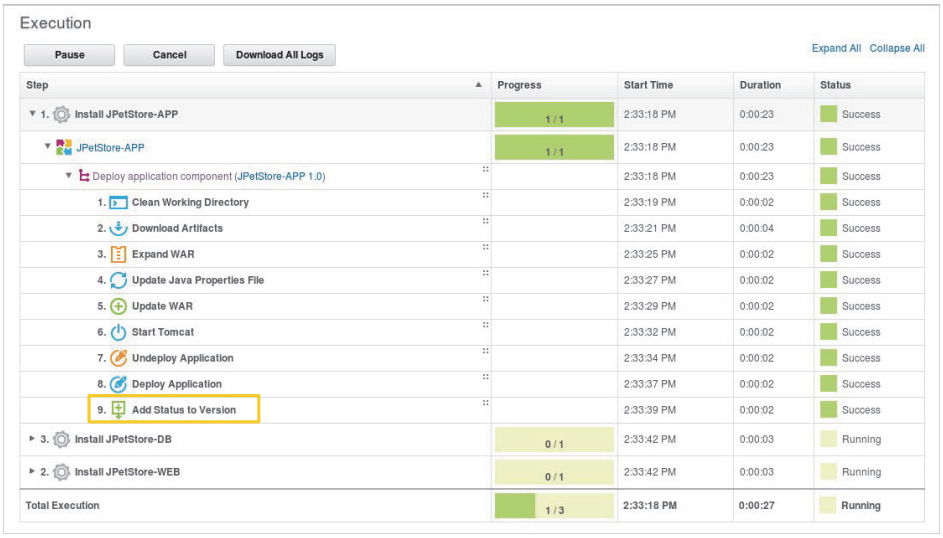


Statuses can be added to versions automatically from component processes

You can add statuses to versions in component processes using the Add Status to Version step in the IBM UrbanCode Deploy Versions plug-in.

IBM Training

Adding version statuses to component versions becomes part of the process

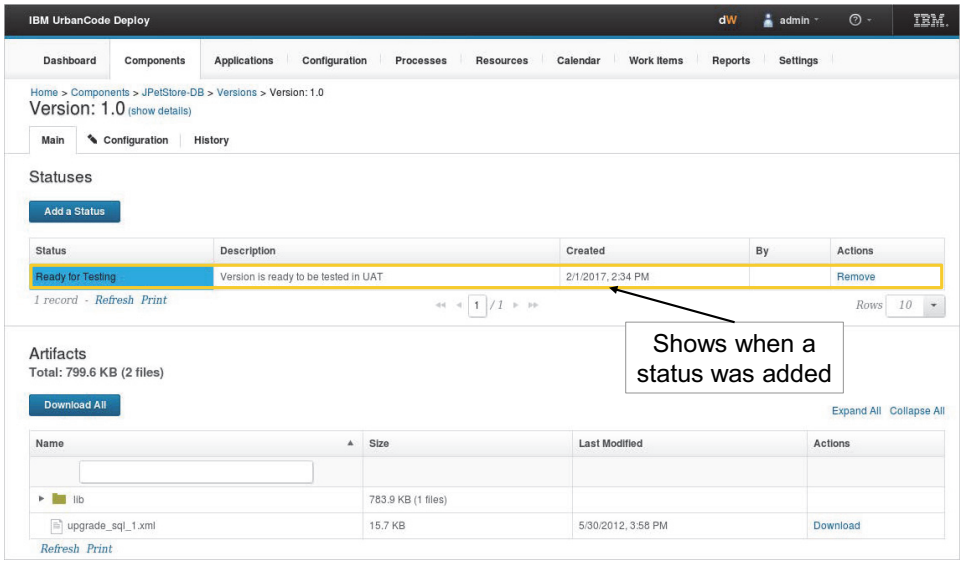


Adding version statuses to component versions becomes part of the process

Now the component process includes adding a version status.

IBM Training

Version statuses are tracked in each component

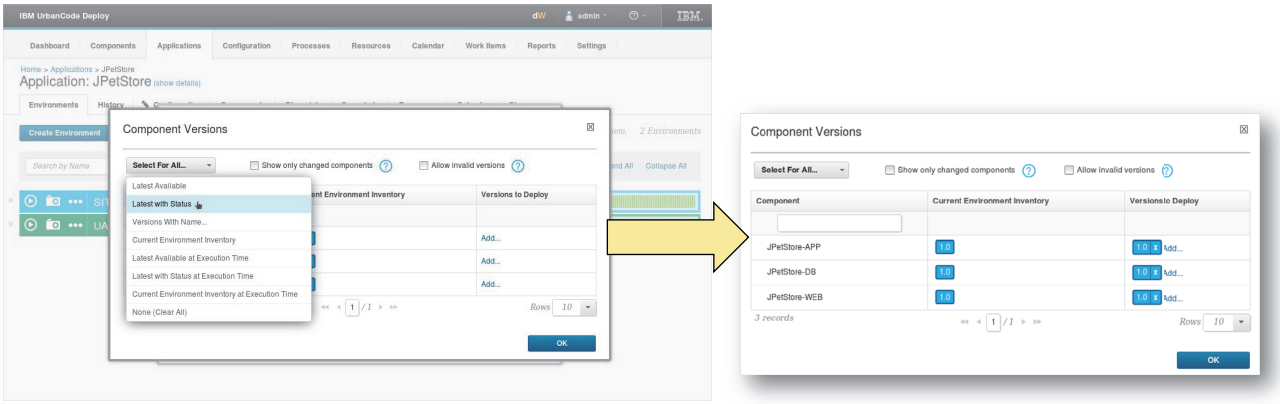


Version statuses are tracked in each component

In order to view a version status, navigate to the component containing the version and then to component's **Versions** tab. Click the version. You see a table containing the version's statuses, and you can add or remove statuses from this interface.

IBM Training

Select the latest components with a status



Select the latest components with a status

When you select components for deployment, select “latest with status” to view the components that have passed the gate.

IBM Training

Unit summary

- The security system for the server consists of an authentication realm, authorization realm, roles, and teams
- Permissions define what can be done, not who can do it; roles define who can do it
- Users are granted permissions by being assigned to teams
- Notifications are emails that are sent whenever user-defined trigger events on the server occur
- Approvals provide more visibility into deployments for audit trails
- An environment gate is a requirement that must be met before component versions can be deployed to an environment
- Version statuses identify when component versions meet criteria

Unit summary

IBM Training

Exercises: Setting up server security

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

Exercises: Setting up server security

IBM Training