# Table of Contents

Lab Overview - HOL-1921-04-CMP - vRealize Code Stream - DevOps Solutions ........... 2  
Lab Guidance ........................................................................................................... 3  
Module 1 - Introduction to vRealize Code Stream (30 minutes) ................................. 10  
  Introduction ............................................................................................................. 11  
  Introducing vRealize Code Stream ...................................................................... 12  
  Architecture Concepts ......................................................................................... 14  
  Introduction to Lab Scenario ................................................................................ 16  
  Building a Simple Application Release Pipeline ................................................ 18  
  Executing a Simple Application Release Pipeline .............................................. 32  
  Conclusion .............................................................................................................. 39  
Module 2 - Using vRealize Code Stream for Application Release Automation (60 minutes)  
  Introduction to Lab Scenario ................................................................................ 42  
  Power on Brownfield Server ................................................................................ 44  
  Building a "Staging" Stage for QA ..................................................................... 47  
  Build a Gating Rule ............................................................................................... 57  
  Build an Approval Policy and Move to Production ............................................ 63  
  Executing the Production Application Release Pipeline .................................... 69  
  Conclusion .............................................................................................................. 79  
Module 3 - Advanced Pipelining with vRealize Code Stream (30 minutes) ............. 81  
  Introduction ............................................................................................................. 82  
  vRealize Code Stream Pipeline Features ............................................................ 83  
  Hands-on Labs Interactive Simulation: vRealize Code Stream Advanced  
  Pipelines .............................................................................................................. 84  
  Conclusion .............................................................................................................. 85
Lab Overview - HOL-1921-04-CMP - vRealize Code Stream - DevOps Solutions
Lab Guidance

Note: It may take more than 90 minutes to complete this lab. You should expect to only finish 2-3 of the modules during your time. The modules are independent of each other so you can start at the beginning of any module and proceed from there. You can use the Table of Contents to access any module of your choosing.

The Table of Contents can be accessed in the upper right-hand corner of the Lab Manual.

You will explore how to use vRealize Code Stream to orchestrate DevOps release processes and continuous application delivery. The same concepts will then be applied to the Software-Defined Data Center to move infrastructure components as code between environments.

Lab Module List:

- **Module 1 - Introduction to vRealize Code Stream** (30 minutes) (Basic)
  
  Explore how vRealize Code Stream leverages your existing software development tools. Review how to implement application release automation for custom developed applications and IT infrastructure.

- **Module 2 - Using vRealize Code Stream for Application Release Automation** (60 minutes) (Advanced)
  
  Create an application release pipeline with vRealize Code Stream, vRealize Automation, Jenkins, and Artifactory. As a release manager, you will automate the build, test, and deployment of a sample eCommerce application.

- **Module 3 - Advanced Pipelining with vRealize Code Stream** (30 minutes) (Advanced)
  
  Explore the full potential of vRealize Code Stream! Trigger a pipeline execution automatically, work with certified artifacts, and incorporate additional endpoints to fully integrate the code release process.

Lab Captains:

- **Module 1 - Ed Bontempo, Senior Solutions Engineer, USA**
- **Module 2 - Ed Bontempo, Senior Solutions Engineer, USA**
- **Module 3 - Ed Bontempo, Senior Solutions Engineer, USA**

This lab manual can be downloaded from the Hands-on Labs Document site found here:

http://docs.hol.vmware.com

This lab may be available in other languages. To set your language preference and have a localized manual deployed with your lab, you may utilize this document to help guide you through the process:

Introduction to the Scenario

Rainpole Systems is an electronics manufacturer located in Palo Alto, CA. Rainpole designs and manufactures electronic devices for use in everything from aircraft instrumentation to home automation. Given the diversity of their product set, Rainpole needs to develop cloud-based IT services to support the increasing demands of the business units.

Rainpole Systems is planning to use vRealize Automation and vRealize Code Stream to automate the build, testing, and deployment of their custom developed eCommerce application. They also plan to leverage the vRealize Code Stream Management Pack for IT DevOps to promote vRealize Automation application blueprints from development to production to minimize the chance of errors impacting their production cloud environment.

Rainpole Systems is currently leveraging a number of tools in their software development lifecycle including Jenkins, Gitlab, and Selenium. These systems have been loosely tied together through custom scripts, but they lack an overall view of where particular builds are in the application release process. Furthermore, they lack governance as code is promoted from development to test and ultimately to production. They are exploring leveraging vRealize Code Stream to streamline these processes and improve their overall software development lifecycle.

You will take on the roles of a Rainpole Cloud Administrator, a Rainpole Developer, and a Development Manager in this exciting lab exercise in order to experience how VMware and vRealize Code Stream can help make these goals a reality.
1. The area in the RED box contains the Main Console. The Lab Manual is on the tab to the Right of the Main Console.

2. A particular lab may have additional consoles found on separate tabs in the upper left. You will be directed to open another specific console if needed.

3. Your lab starts with 90 minutes on the timer. The lab cannot be saved. All your work must be done during the lab session. But you can click the **EXTEND** to increase your time. If you are at a VMware event, you can extend your lab time twice, for up to 30 minutes. Each click gives you an additional 15 minutes.

   Outside of VMware events, you can extend your lab time up to 9 hours and 30 minutes. Each click gives you an additional hour.

**Alternate Methods of Keyboard Data Entry**

During this module, you will input text into the Main Console. Besides directly typing it in, there are two very helpful methods of entering data which make it easier to enter complex data.
Click and Drag Lab Manual Content Into Console Active Window

You can also click and drag text and Command Line Interface (CLI) commands directly from the Lab Manual into the active window in the Main Console.

Accessing the Online International Keyboard

You can also use the Online International Keyboard found in the Main Console.

1. Click on the Keyboard Icon found on the Windows Quick Launch Task Bar.
Click once in active console window

In this example, you will use the Online Keyboard to enter the "@" sign used in email addresses. The "@" sign is Shift-2 on US keyboard layouts.

1. Click once in the active console window.
2. Click on the Shift key.

Click on the @ key

1. Click on the "@ key".

Notice the @ sign entered in the active console window.
Activation Prompt or Watermark

When you first start your lab, you may notice a watermark on the desktop indicating that Windows is not activated.

One of the major benefits of virtualization is that virtual machines can be moved and run on any platform. The Hands-on Labs utilizes this benefit and we are able to run the labs out of multiple datacenters. However, these datacenters may not have identical processors, which triggers a Microsoft activation check through the Internet.

Rest assured, VMware and the Hands-on Labs are in full compliance with Microsoft licensing requirements. The lab that you are using is a self-contained pod and does not have full access to the Internet, which is required for Windows to verify the activation. Without full access to the Internet, this automated process fails and you see this watermark.

This cosmetic issue has no effect on your lab.

Look at the lower right portion of the screen
Please check to see that your lab is finished all the startup routines and is ready for you to start. If you see anything other than "Ready", please wait a few minutes. If after 5 minutes you lab has not changed to "Ready", please ask for assistance.
Module 1 - Introduction to vRealize Code Stream (30 minutes)
Introduction

This module will demonstrate the power of vRealize Code Stream to rapidly build application release pipelines. With application release pipelines, you can model and automate existing processes to accelerate the delivery of custom application code from the developer's laptop all the way through to production. You will begin with an overview of vRealize Code Stream, how it works, and what problems it solves. Finally, you will create a simple application release pipeline with a test stage.

This HOL-1921-04-CMP lab is broken up into 3 individual modules with varying lengths. Use the "Modules" list to determine which use case(s) you want to complete. You may choose to complete any or all modules, keeping in mind that you have 90 minutes available per session.

If you have not already reviewed it, the Lab Introduction contains details about Rainpole Systems (our example company) as well as important information on text entry, multiple language support, and on-screen keyboard configuration. If desired, you can click here to view this information again.

You will need about 30 minutes to complete this module.
Introducing vRealize Code Stream

VMware vRealize Code Stream is an application release automation and continuous delivery solution that allows developers and operations teams to release software more frequently and efficiently, all the while leveraging their investments in existing developer and operations toolsets.

vRealize Code Stream Introduction

What is **vRealize Code Stream**? Think of Code Stream as a code-release-specific orchestration engine. It allows users to define a visual representation/template of a release pipeline, tying in to all of the existing various SDLC mechanisms that a customer is using, while providing stage-gate checks to verify if a release should continue its path from one stage to the next.

In a release delivery process, after development checks in their enhancements or new features, several additional groups need to test, stage, and accept the release prior to putting it into production. Today this is largely accomplished using siloed processes with a collection of tools that may or may not be consistent across groups. Not only does the process to deploy and test at each stage take too long, but coordination and consistency at each stage also suffers. vRealize Code Stream allows application release teams to model any kind of release process for any kind of application - from simple, single-node, on-premises applications to complex, multi-tiered, cloud-based or hybrid next-gen applications. Pipeline models configure the workflow tasks and governance policies used to build, deploy, and test the software at each stage in the delivery process, as well as the gating rules between stages.

As an introduction, watch this brief video on vRealize Code Stream.
Overview and Benefits

VMware helps enterprise IT to overcome cloud management challenges with the following differentiators:

- **Achieving Fastest Time-to-Cloud Value**: Extensive out-of-the-box capabilities, extensible architecture, and ability to leverage existing investments enables you to create scalable private, public, and desktop cloud services in just days.
- **Rapid Ecosystem Integration**: A full spectrum of extensibility options that empowers IT personnel to enable, adapt, and extend their cloud to work within their existing IT infrastructure and processes, thereby eliminating expensive service engagements while reducing risk.
- **Embracing Consumerization of IT**: Unifies disparate systems and processes into a highly personalized, self-service experience governed by user and organizational awareness, ensuring that the exact needs of the user are met in the context of the business.
- **Boosting IT Efficiency**: VMware's advanced self-service and automation capabilities lead to a significant reduction in operational expenditures (OpEx,) while organizationally aware governance enables a multi-tenant infrastructure, improving hardware utilization and eliminating capital expenditures (CapEx.)
- **Increasing Business Agility**: Enterprise IT is empowered to quickly enable cloud services so that the business can react quickly to changing market demands and capture market share.
- **Hybrid Cloud Automation**: Agnostic provisioning of hardware platforms, operating systems, hypervisors, management tools, and public clouds; vRealize Automation allows customers to rapidly stand up cloud services to deliver quick time to value from your IT investments.
Architecture Concepts

vRealize Code Stream is integrated with vRealize Automation and can work with vCenter Server, Chef, Puppet, or scripts. vRealize Code Stream can also integrate with continuous integration frameworks and testing frameworks, and a variety of other development toolsets.

Architecture

vRealize Code Stream includes three primary modules: a pipeline execution engine, a release dashboard, and reporting. Code Stream also integrates with a number of DevOps-related components, including but not limited to:

- **Repository and Source Code Management** tools, including jFrog Artifactory, git, and Nexus
- **Continuous Integration (CI)** tools, including Jenkins, Bamboo, and Microsoft Team Foundation Server
- **Testing Frameworks** such as Selenium
- **Issue Tracking** tools, including Jira and Bugzilla
- **Change Management and ITSM** toolsets, including BMC Remedy and ServiceNow
- **Provisioning and Configuration Management** tools, including vRealize Automation, Pivotal Cloud Foundry, and Puppet
This integration is provided by vRealize Code Stream's open source Xenon engine, and is available out of the box, via REST API integration, or through vRealize Orchestrator plugins.

**Key Release Automation Concepts**

Use the following definitions to help you understand the release pipeline modeling and the artifact management workflow.

**Artifact:** A script or the output of a build process. The script can be deployed or upgraded in a given stage. Artifact types can be configuration files, application bits, or third-party software.

**Artifact Management:** A service that manages the artifacts over a range of local and remote repositories. For example, managing a WAR file stored in the Maven repository.

**Category:** A task type. Some supported categories are Provision, Custom, Artifact, Deploy, and Test. A task belongs to a provider and to a category.

**Gating Rule:** A set of rules that must be completed before the software changes are promoted to the next stage, starting the next set of tasks. The gating rules include testing rules and compliance rules. Gating rules that are associated with a pipeline are specific to an organization and to applications.

**Instance:** A vRealize Orchestrator plugin scenario that captures specific configurations of a provider. The instance is created by using a vRealize Orchestrator client.

**Pipeline:** A collection of all the stages or environments in which a software change has to pass through independently before it is release into production. For example: development, test, user acceptance test (UAT,) load test, staging, and production.

**Provider:** Plugin vendors that support the categories. For example, the Provision category is supported by vRealize Automation and vCenter Server providers.

**Stage:** Every stage in the pipeline defines a set of activities. For example: deploy, test, approval through gating rules, and custom tasks.

**Task:** An activity in a given stage. For example: provision the machines, resolve the artifact, deploy the artifact, run the test, and so on.
Introduction to Lab Scenario

As you work through the lab's use cases, you'll be assuming roles within a fictional company known as Rainpole Systems, and addressing their business and IT challenges.

Rainpole is an IT company that is under pressure to build private clouds, which means creating a data center architecture that can deliver flexible, scalable computing while maintaining governance and control over their environment. Rainpole's IT Automation team decided to reach out to VMware to help them with their journey to private cloud.

vRealize Automation empowers IT to accelerate the delivery of personalized, business-relevant IT services while improving overall efficiency, by delivering infrastructure, applications, and custom services through a unified IT service catalog. In addition to the need to drastically reduce the time to provision services for their customers, Rainpole also had a need to have a more agile development process that will enable more frequent and reliable software releases. vRealize Code Stream fits in to this category by allowing for the modeling of any release process for any kind of software, while integrating with existing continuous integration tools and artifact management.

Over the next few months, Rainpole's IT Automation team will be starting their private cloud journey with vRealize Automation and vRealize Code Stream.

Rainpole's Fundamental IT Challenge

• Quickly implement a tool which will allow IT to maintain control over their data center resources while allowing engineering to quickly create and destroy environments for development and testing using shared, heterogeneous clouds.
• Provide a personalized self-service experience to the different teams within engineering development (as a starting point, within QE, or Quality Engineering, and potentially other parts of the organization to be added in the future.)
• Decrease service delivery times by automating the end-to-end delivery and management of resources to meet the needs of the agile development environment. Provide an approval mechanism when needed to ensure compliance is met.

In order to support self-service for the engineering and QE teams while leveraging existing infrastructure from both internal and external resources and retaining the necessary controls, the Rainpole IT leadership team has decided to implement VMware's next generation cloud management solution. The Rainpole development team will kick off the implementation as the first group of users, with QE scheduled to follow on shortly after in a second wave.

If you have not already reviewed it, the Lab Introduction contains useful details about Rainpole Systems (our example company,) as well as important information on text entry, multiple language support, and on-screen keyboard configuration. If desired, you can click here to review this information again.
In this module, you will act as the Rainpole Cloud Administrator to explore building and executing a simple application release pipeline within vRealize Code Stream.
Building a Simple Application Release Pipeline

In this exercise, you will build a simple Application Release Pipeline with vRealize Code Stream.

Open Chrome Browser from Windows Quick Launch Task Bar

First you will log in to vRealize Code Stream to begin building a new application release pipeline.

1. Click on the Chrome icon on the Windows Quick Launch Task Bar.

Set Browser Zoom (if Needed)

Note: due to the limited resolution in this lab environment, it may be necessary to adjust the browser zoom in order to properly view all of the vRealize Code Stream interface. While this is not required, if you do prefer to adjust the browser zoom:

1. Click the 3 vertical dots in the upper right corner of the browser window.
2. Click the - (minus sign) in the Zoom section to zoom out from the default. A setting of 90% should suffice, but it may be necessary to click twice to set a value of 80%.

Log In to vRealize Code Stream

1. Log in as rpadmin with the password VMware1!
2. Click Sign In

Passwords for all users will be the same: VMware1!
Examine the MyCommerce Test Blueprint

Before we create a pipeline, we will first examine the vRealize Automation blueprint used in this exercise.

1. Click the **Design** tab, and the Blueprints section will load by default.
2. To view the **MyCommerce Test** blueprint, click on the name (blue text) to open the Blueprint Design Canvas.
View the Blueprint Design Canvas

View the blueprint on the **Design Canvas** and make note of its components. When you deploy a test system with vRealize Code Stream you will select this blueprint, which deploys a CentOS server, performs systems configuration (disabling the firewall,) installs Apache Web Server, and connects the machine to an existing network.

1. Note the **Apache** and **System_Configuration** software components, deployed to the vSphere Machine
2. Note the blueprint’s connection to the **DefaultExternalNetwork**
3. Click **Cancel** to exit the design canvas and begin creating your new pipeline (NOTE: if the Confirm Cancel window appears, click **Yes** to close the blueprint and discard any changes.)
Navigate to Code Stream and Add a New Pipeline

1. Click the Code Stream tab
2. Click the Pipelines sub-tab
3. Finally, click the Add button in the toolbar to begin creating a new pipeline

Build a New Pipeline with a Test Stage

1. Name: HOL - Simple Pipeline
2. Description: Simple pipeline created in Module 1
The first thing you'll want to do once you're in Code Stream is to start building out your own pipeline. Pipelines can be modeled first without specifying any configurations. This can give you a good way to lay out the entire process before entering any details.

1. Enter **HOL - Simple Pipeline** for the name
2. Enter **Simple pipeline created in Module 1** for the description
3. Click **Stages >** to begin modeling the pipeline

### Add a Pipeline Stage

![Add a Pipeline Stage Image]

To add a stage in vRealize Code Stream:

1. Click on **Add Stage**, which will create a basic stage named **Stage0**
2. Double-click **Stage0** and enter **Test** to rename it
3. Click **Add Task** to begin creation of the first task in the pipeline
Add a Provisioning Task

Enter the following information for the task:

1. Select **vRealize Automation 7 - Deploy** for the Provider drop-down
2. For Name, enter **Provision Machine**
3. Click **OK**

Configure Provisioning Task

1. Select the gear icon (not shown) next to the Provision Machine task and then select **Configure** from the drop-down

You will be prompted to save the pipeline configuration (not shown) - click **Yes** when prompted.
Before configuring the task, you should resize the window so it fits better in the screen. Click in the upper left corner and drag down and to the right to resize (not shown here.) You should be able to see the Save and Cancel buttons at the bottom of the Task Configuration window, as well as the scrollbar on the right.

1. In the task editor, click the drop-down and select **vRealize Automation 7** for the Endpoint.
2. Click the drop-down next to Blueprint and select **MyCommerce Test**. Note this is the blueprint you examined earlier in this exercise.
Task vRA Properties, Continued

1. Scroll the window down by dragging the slider on the right side
2. On the left side of the window, click MyCommerce
3. Note the properties that are available for the virtual machine. The options for CPUs, Memory, etc are defined for the vRealize Automation blueprint and are available for selection (if allowed) here
4. Enter **HOL-1921 Code Stream** for the Description
5. Click **Save** to continue to be taken back to the Stages screen
Add a Deploy Website Task

1. Still using the Test stage, select the **Add Task** icon to create a new task

**Deploy Task**

1. Select **Script** for the Provider drop-down
2. For the Name, enter **Install Website**
3. Select **OK**
1. Select the **gear icon** next to the **Install Website** task and then select **Configure** from the drop-down

You will be prompted to save the pipeline configuration (not shown) - click **Yes** when prompted.
1. In the General tab, enter the following information:
   - For Host, select **Read From Pipeline or Task Property**. This is because you want to leverage the VM that gets created as a result of the Provision task you just created.
   - In the Machine Type Property Path (the field directly under the User Defined Hosts radio button,) type `$\{Test.Provision Machine.machines\}` (Note, there is a space between the words *Provision* and *Machine*, and this field is case sensitive.)
   - Select **All Hosts** from the drop-down next to the variable (this value should already be set by default)
   - Username: **root**
   - Password and Confirm Password: **VMware1!**
   - Script Type: **Bash**
   - Execute Script file: `/root/mycommerce/install.sh`

2. Select the **Advanced** tab

**Note**: You can paste the variables or begin typing `$\{` and make the selection from the list presented. The next variable will be presented once you click on the existing variable.
Configure Custom Script Task - Advanced

Under Script Parameters - Other, enter the following parameters:

1. Select Add
2. Enter the following parameter names and values:
   - Parameter Name: JENKINS_SERVER
     - Parameter Value: jenkins-01a.corp.local
   - Parameter Name: JENKINS_JOB_NAME
     - Parameter Value: MyCommerce-Build (Note: The parameter value is case sensitive)

3. Click Save to be taken back to the Stages screen
Completed Test Stage

1. Once both tasks are configured and saved, click **Activate** to activate the pipeline template
Executing a Simple Application Release Pipeline

In the previous exercise, you created a simple application pipeline and activated it. It is now time to execute the pipeline to validate that everything is working as expected.

Execute the Pipeline

1. Click to highlight the pipeline **HOL - Simple Pipeline**
2. Click **Execute**
1. Click **OK**
Enable Auto-Refresh

1. Click the **Pipeline Executions** tab
2. Click **Enable Auto-refresh** to automatically update the pipeline status (it will likely take several minutes for the pipeline to complete)
3. Click the **small arrow** to the left of the pipeline execution to see the details
Successful Pipeline Execution

Once the pipeline Test stage and associated tasks turn green, it has executed successfully.

Deployment Verification

To verify the VM and code deployment worked, you will need the IP address of the VM that was created as part of the pipeline execution.
1. Within vRealize Automation, select the **Items** tab. The Deployments tab should be selected by default.
2. Click the arrow next to the most recent deployment in the list that has a name starting with **MyCommerce Test** to expand the components.
3. Document the **IP Address** in the IP Address column. (Note: it may be necessary to hover the mouse cursor over the IP address, or to expand the IP Address column, in order to see it in its entirety.)

### Website Verification

Once you have the IP address, you can verify that your pipeline execution was a success.

1. Create a **new tab** in Chrome
2. Browse to the IP of the newly created VM in the new lab, which will show the **MyCommerce.com** homepage (Note: the IP address in your lab may differ from this screenshot)
Exercise Clean-Up

Once you have verified successful completion of the pipeline, you will now clean up the lab. Close the browser tab displaying the MyCommerce.com site (not shown,) which will return you to the vRealize Automation tab. Once you've returned to vRealize Automation:

1. Click to highlight the recent **MyCommerce Test** deployment (Note: do not click on the name itself, or you will be taken to the Item Details screen rather than selecting the deployment)
2. Click the arrow next to **Actions**
3. Click **Destroy**
1. Click **Submit** to destroy the virtual machine and **OK** (not shown) to confirm.

You have successfully concluded this exercise.
Conclusion

In this module, you have begun to explore the capabilities of vRealize Code Stream. You learned about Application Release Automation and modeled a simple pipeline to quickly deploy a test system using vRealize Automation. After modeling this pipeline, you executed it and verified the results. This was a very simple example and we encourage you to explore Module 2 in this lab to see more complex pipelines and use cases.

You've finished Module 1

Congratulations on completing Module 1.

If you are looking for additional information on vRealize Code Stream, try one of these:

- Or use your smart device to scan the QRC Code.

Proceed to any module below which interests you most.

- **Module 2 - Using vRealize Code Stream for Application Release Automation** (60 minutes) (Advanced) Create an application release pipeline with vRealize Code Stream, vRealize Automation, Jenkins, and Artifactory. As a release manager, you will automate the build, test, and deployment of a sample eCommerce application.
- **Module 3 - Advanced Pipelining with vRealize Code Stream** (30 minutes) (Advanced) Explore the full potential of vRealize Code Stream! Trigger a pipeline execution automatically, work with certified artifacts, and incorporate additional endpoints to fully integrate the code release process.

How to End Lab
To end your lab click on the **END** button.
Module 2 - Using vRealize Code Stream for Application Release Automation (60 minutes)
Introduction to Lab Scenario

As you work through the lab's use cases, you'll be assuming roles within a fictional company, Rainpole Systems, and addressing their business and IT challenges.

Rainpole is an IT company that is under pressure to build private clouds, which means creating a data center architecture that can deliver a flexible, scalable computing environment while maintaining governance and control over their environment. The Rainpole IT Automation team decided to reach out to VMware to help them with their journey to the private cloud. vRealize Automation empowers IT to accelerate the delivery of personalized, business-relevant IT services while improving overall efficiency by delivering infrastructure, applications, and custom services through a unified IT service catalog. In addition for a need to drastically cut off the time to provision services for their customers, Rainpole also had a need to have a more agile development process that will enable more frequent and reliable software releases. vRealize Code Stream fits into this category by allowing modeling of any release process for any kind of software, while integrating with existing continuous integration tools and artifact management.

Over the next few months, the Rainpole IT Automation team will be starting their private cloud journey with vRealize Automation and vRealize Code Stream.

Rainpole's Fundamental IT Challenge

- Quickly implement a tool which will allow IT to maintain control over their data center resources while allowing engineering to quickly create and destroy environments for development and testing using shared, heterogenous clouds.
- Provide a personalized self-service experience to the different teams within engineering development (as a starting point, with QE and potentially other parts of the organization to be added in the future.)
- Decrease service delivery times by automating the end-to-end delivery and management of resources to meet the needs of the agile development environment. Provide an approval mechanism when needed to ensure compliance is met.

In order to support self-service for the engineering and QE teams while leveraging existing infrastructure from both internal and external resources and retaining the necessary controls, the Rainpole IT leadership team has decided to implement VMware's next generation cloud management solution. The Rainpole development team will kick off the implementation as the first group of users, with QE scheduled to follow on shortly after in a second wave.

If you have not already reviewed it, the Lab Introduction contains details about Rainpole Systems (our example company,) as well as important information on text entry,
multiple language support, and on-screen keyboard configuration. If desired, you can click here to review this information again.

This module will build on the simple application release pipeline created in Module 1. If you did not take Module 1, a starter pipeline with the work from Module 1 is already created so there is no requirement to previously complete it.

In this module, you will act as the Rainpole Administrator and the Rainpole Cloud Administrator to explore vRealize Code Stream.
Power on Brownfield Server

Before we begin the exercises in this module, you will need to power on a server used by the subsequent pipelines. This server will act as a "brownfield" staging and production environment for the MyCommerce.com application.

Open Chrome Browser

1. Click the Chrome icon in the Windows Quick Launch Task Bar. If the browser window is already open following completion of the previous module, open a new tab in the browser instead (not shown.)

Navigate to the vSphere Client and Log In

![vSphere Client Login Process](image)
1. In Chrome, click the **Region A** folder to expand the list of bookmarks
2. Click **HTML 5 Client** in the drop-down
3. Click the checkbox labeled **Use Windows session authentication** to populate the User name and Password fields
4. Click **Login**

**Locate the Brownfield VM**

1. Locate the VM brownfield-01a in the inventory (pictured above,) and click on it. (NOTE: You may need to scroll down to see the brownfield-01a VM in the inventory list.)
Power On Brownfield VM

1. Click the Actions link to expand the menu of available Virtual Machine actions
2. Move the cursor over the Power section to expand the sub-menu
3. Click Power On

Verify Power On and Exit

1. Verify that the brownfield-01a virtual machine icon has changed to include a green arrow, indicating it is powered on. Once verified, you can log out of the vCenter client.
2. Click Administrator@CORP.LOCAL to expand the menu
3. Click Logout (NOTE: if you have completed Module 1 and have multiple browser tabs open, you can close the vSphere Client browser tab. If this is the only browser tab available, then leave it open for now.)
Building a "Staging" Stage for QA

In this exercise, you will expand a simple application release pipeline to include a "Staging" stage.

Navigate to vRealize Automation

First you will log in to vRealize Automation to begin building a new application release pipeline.

1. Click on the **vRealize Automation** on the Chrome bookmarks bar. If vRealize Automation is already open from Module 1, then you may skip this step and the next step.

Set Browser Zoom (if Needed)

Note: due to the limited resolution in this lab environment, it may be necessary to adjust the browser zoom in order to properly view all of the vRealize Code Stream interface. While this is not required, if you do prefer to adjust the browser zoom:

1. Click the **3 vertical dots** in the upper right corner of the browser window.
2. Click the - (minus sign) in the Zoom section to zoom out from the default. A setting of 90% should suffice, but it may be necessary to click twice to set a value of 80%.

Log in to vRealize Automation

1. Log in as rpadmin with the password **VMware1**!
2. Click **Sign in**

Passwords for all users will be the same: **VMware1**!
Navigate to Code Stream and Edit a Pipeline

1. Click the **Code Stream** tab
2. Click the **Pipelines** sub-tab
3. Click the pipeline named **Base - Standard Pipeline**
4. Finally, click the **Edit** button in the toolbar to begin editing the pipeline
Navigate to Stages

1. Click on **Stages** > to navigate to the Stages view

Adding a "Staging" Stage

To add a "Staging" stage in vRealize Code Stream:

1. Click on **Add Stage**, which will create a basic stage named **Stage0**
2. Double-click **Stage0** and enter **Staging**

**Clone an Existing Task**

Next, you're going to create an Install Website task for the "Staging" stage. To save time, you can clone the existing task from the Test stage.

1. Select the gear icon next to the Install Website task and select **Clone** from the drop-down menu.
2. Click **OK** (not shown) on the Clone Task warning that passwords will be reset to null.

**Change the Name of the Install Website Task**

1. Drag the cloned **Install Website-1** task to the Staging stage (not shown)
2. Select the gear icon next to the Install Website-1 task and then click **Configure** in the drop-down
3. You will be prompted to save the pipeline configuration (not shown) - click **Yes** when prompted.

### Change the Name of the Install Website Task

![Task Configuration for Script](image)

1. Click in the Name field of the Task Configuration for Script and change the name to **Install Website**

### Configure Install Website Task - General

![Task Configuration for Script](image)
1. Scroll down the Task Configuration window by dragging the slider on the right side.

2. In the general section, enter the following information:
   ◦ For Host, select User Defined Hosts. This is because you will use an existing machine that has already been provisioned to demonstrate the ability to leverage "brownfield" hosts in a pipeline.
   ◦ In the open text box below User Defined Hosts, type brownfield-01a.corp.local and press Enter.
   ◦ Username: verify it is still set to root.
   ◦ Password and Confirm Password: VMware1!
   ◦ Script type: verify it is still set to Bash.
   ◦ Script File: verify it is still set to /root/mycommerce/install.sh.

3. Click the Advanced tab.

**Configure Install Website Task - Advanced**

![Configure Install Website Task - Advanced](image)

1. Under Script Parameters - Other, verify the following parameters:
   ◦ Name: JENKINS_SERVER
   ◦ Value: jenkins-01a.corp.local
   ◦ Name: JENKINS_JOB_NAME
   ◦ Value: MyCommerce-Build

2. Click Save.
Add a Delete Test Machine Task

Because you no longer need the test machine you provisioned in the test stage, you can create a task to delete the test machine. A vRealize Orchestrator workflow has already been created to call the vRealize Automation API to destroy the virtual machine. You will now add this task as another step in the Staging stage.

1. Click **Add Task** on the Staging stage

**Add a Delete Test Machine Task, Continued**

Enter the following information.

1. Select **vRealize Orchestrator** for the Provider drop-down (Note: you may need to scroll down in the drop-down menu to see vRealize Orchestrator)
2. For Name, enter **Delete Test Machine**
3. Select **OK**
Configure Delete Test Machine Task

1. Select the gear icon next to the Delete Test Machine task, and then select **Configure** in the drop-down

You will be prompted to save the pipeline configuration (not shown) - click **Yes** when prompted.
Configure Custom Script Task - General

1. For Workflow, select **Destroy vRA Machine** from the drop-down
2. In the Input Parameters section, for machineName enter $\{Test.Provision Test Machine.machines\}$ (Note, there is a space between the words Provision and Test and Machine, and that this field is case sensitive.)
3. Click **Save**

**Note:** You can paste the variables above or start typing $\{$ and make the selection from the list presented. The next variable will be presented once you click on the existing variable.

This completes the Staging stage for now. Remain on the **Edit Pipeline** screen for the next exercise.
Build a Gating Rule

Gating rules allow you to control when code is promoted between pipeline stages.

You can create gating rules based upon approvals or upon the results of automated tests performed with testing frameworks like Selenium. In this exercise, you will create a gating rule to promote code from the Test stage to the Staging stage based on the results of a Selenium test executed by Jenkins.

Return to the Pipeline Properties Screen

1. On the Edit Pipeline screen from the previous exercise, click **Previous**
Add a Property Value to the Pipeline

1. Click + Add to add a new property
2. Enter the following for the newly created property:
   ◦ Name: teststring
   ◦ Description: Test String for Selenium Tests
   ◦ Value: MyCommerce.com - Deal of the Day
   ◦ Note: All entries in Step 2 are case sensitive
3. Click Stages > to continue

Create a Selenium Test Task in the Test Stage
1. In the **Test** stage, click **+ Add Task**

**Adding Selenium Test Task**

![Add Task]

Enter the following information:

1. Select **Jenkins** for the Provider drop-down
2. For Name, enter **Selenium Test**
3. Click **OK**

**Configure Selenium Test Task**

![Configure]

1. Select the gear icon next to the Selenium Test task, and then select **Configure** in the drop-down. You will be prompted to save the pipeline configuration (not shown) - click **Yes** when prompted.
(Note: it may be necessary to scroll down in the Test stage to see the Selenium Test task)

**Selenium Test Jenkins Properties**

1. Scroll down the Task Configuration window by dragging the slider on the right side
2. For Endpoint, select **Jenkins** from the drop-down
3. For Job, select **MyCommerce-Test** from the drop-down
4. In the Parameters section, for testurl enter `${Test.Provision Test Machine.machines[0].value.hostIp[0]}`
5. In the Parameters section, for teststring enter `${pipeline.teststring}`
6. Click **Save**

**Note:** You can paste the variables or start typing `{` and make the selection from the list presented. The next variable will be presented once you click on the existing variable, though the last two in the string will need to be manually entered. The next step will explain more about the syntax used above.
**Note About JSON Syntax**

When specifying the `testurl` parameter in the previous step, you may have noticed how the parameter completion didn't work once you got to the "[0]" part. This is because we need to pass the IP of the host to our Selenium job, which is verifying that the website was actually deployed and is running.

These kinds of expressions are possible in vRealize Code Stream, since the values are passed in JSON format. For variable binding in Code Stream, you can autocomplete up to any variable in the pipeline, but the expression after (in our case, after ".machines", ) is a pure JSON path expression.

The string we are parsing contains a "hostIp" parameter, which we need to pass to the Selenium test. We know that it contains this because we first captured the output of `${Test.Provision Machine.machines}`.

For more information about this syntax, refer to [https://github.com/jayway/JsonPath](https://github.com/jayway/JsonPath) outside of this environment.

**Add a Gating Rule**

1. Between the Test and Staging stages, click on the **Gating Rule** icon
Configure the Gating Rule

1. Select the radio button next to **If outcome of a vRealize Orchestrator workflow is successful** (Note: it may be necessary to drag the Gating Rule Configuration window up following this step, in order to view the rest of the window)
2. For Select Workflow, select **Test Acceptance Threshold** from the drop-down
3. In the Input Parameters section, for Threshold Percentage enter **100**
4. In the Input Parameters section, for Test Result enter **${Test.Selenium Test.testResult}** (Note: there is a space between the words Selenium and Test, and this field is case sensitive.)
5. Click **Save**

**Note:** You can paste the variables or start typing ${ and make the selection from the list presented. The next variable will be presented once you click on the existing variable.

This completes the exercise. Next you will build a production stage.
Build an Approval Policy and Move to Production

Next, you will create a new stage called Production with an approval policy that requires the Rainpole Cloud Admin to approve code moves to production.

To save on lab resources, you will re-deploy to the brownfield server and pretend that it's a server in a production environment.

Adding a Production Stage

To add a Production stage in vRealize Code Stream:

1. Click on + Add Stage to create a basic stage named Stage0
2. Double click Stage0 and enter Production
3. Select the gear icon (not shown) next to the Install Website task in the Staging stage, and then select Clone in the drop-down

Click OK (not shown) on the Clone Task warning that passwords will be reset to null.
Configure the Install Website Task

1. Drag the cloned **Install Website-1** task to the Production stage
2. Select the gear icon (not shown) next to the Install Website-1 task and then select **Configure** in the drop-down

You will be prompted to save the pipeline configuration (not shown) - click **Yes** when prompted.

Change the Name of the Install Website Task

1. Click on the Name field of Task Configuration and enter **Install Website**
1. Scroll down the Task Configuration window by dragging the slider on the right side.

2. In the General section, enter the following information:
   ◦ For Host, verify that User Defined Hosts is selected.
   ◦ In the open textbox, verify that brownfield-01a.corp.local is entered
   ◦ Username: verify that it is still set to root
   ◦ For the Password enter VMware1! and Confirm the Password
   ◦ Script type: verify that it is still set to Bash
   ◦ Script File: verify that it is still set to /root/mycommerce/install.sh

3. Click the Advanced tab

Note: The Production and Staging stages utilize the same brownfield server. This would not normally be done in a pipeline, but was done here due to limited lab resources.
Configure Install Website Task - Advanced

1. Under Script Parameters - Other, verify the following parameters:
   - Name: **JENKINS_SERVER**
   - Value: jenkins-01a.corp.local
   - Name: **JENKINS_JOB_NAME**
   - Value: **MyCommerce-Build** (Note: this parameter value is case sensitive)

2. Click **Save**
Add a Gating Rule

1. Between the Staging and Production stages, click on the Gating Rules icon

Configure the Gating Rule

- Select a condition to proceed to next stage: No Gating Rule
  - If outcome of a vRealize Orchestrator workflow is successful
- Select Workflows: Approval
- Input Parameters:
  - Approval Group (DN): vsphere.local/approvalgroup
  - Approval Message: The execution of $releasePipelineName

4. Save
1. Select the radio button next to **If outcome of a vRealize Orchestrator workflow is successful** (Note: it may be necessary to click on the Gating Rule Configuration window title and drag it up in order to view the entire window)
2. For Select Workflow, select **Approval** from the drop-down
3. In the Input Parameters section, for Approval Group (DN) enter `vsphere.local\approvalgroup`
4. Click **Save**

The base lab already has a custom group created in vRealize Automation named approvalgroup with cloudadmin@corp.local as its only member.

**Activate Pipeline**

1. Click **Activate** to activate the pipeline. The pipeline is now ready to be executed.
Executing the Production Application Release Pipeline

In the previous exercise, you created a production application release pipeline and activated it. It is now time to execute the pipeline to validate that everything is working as expected.

Execute the Pipeline

1. Click to highlight the pipeline **Base - Standard Pipeline**
2. Click **Execute**
Execute the Pipeline, Continued

1. Click OK
Enable Auto-Refresh

1. Click the **Pipeline Executions** tab
2. Click **Enable Auto-refresh** to automatically update the status (Note: Auto-refresh may already be enabled if you've completed the previous module, in which case it is not necessary to click the Auto-refresh button)
3. Click the **small arrow** on the left of the pipeline execution to see the details
Pipeline Pause for Approval

1. The pipeline will successfully complete up to the gating rule between Staging and Production. This pause is for the approval policy that you defined in the gating rule.

**NOTE:** The pipeline is paused when a yellow 'pause' symbol appears next to the IN_PROGRESS status. You will also notice a vertical yellow bar next to the gating rule. Continue to the next step when the pipeline pauses. (This process may take 10 or 15 minutes.)

Log Out of vRealize Automation

Now let's log out of vRealize Code Stream in order to approve the request to move code into production.

1. Click **Logout**
Return to Login Page

1. Click **Go back to login page**

Log in to vRealize Automation as Cloud Admin

1. Enter **cloudadmin** as the username.
2. Enter ************ as the password.
3. Click **Sign in**.
1. Log in as **cloudadmin** with the password **VMware1!**
2. Click **Sign in**

### Navigate to the Inbox to Approve the Request

1. Click the **Inbox** tab
2. Click on the **Manual User Action** tab
3. Click on the **request number (example: number 1)** to open the approval tab
Approve the Request

1. Notice the details of the request in the Approver Message field.
2. Enter **This code is ready for production** in the Comments field
3. Click **Submit** to approve

Log Out of vRealize Automation

Now let's log out of vRealize Automation in order to return to the pipeline execution already in progress.

1. Click **Logout**
Return to Login Page

1. Click **Go back to login page**

Log In to vRealize Automation as Rainpole Admin

1. Log in as **rpadmin** with the password **VMware1!**
2. Click **Sign in**

**Successful Pipeline Execution**

![Pipeline Execution](image)

You will now return to the Pipeline Execution sub-tab within Code Stream, and verify that the pipeline completed successfully.

1. Click on the **Code Stream** tab
2. Click the **Pipeline Executions** sub-tab
3. Expand the pipeline
4. Validate that the Status of the pipeline has changed to **Execution finished successfully**.
You can now verify that your pipeline execution was a success.

1. Create a **new tab** in Chrome
2. Browse to [http://brownfield-01a.corp.local](http://brownfield-01a.corp.local) in the new tab, which will show the MyCommerce.com homepage

You have completed this exercise.
Conclusion

In this module, you have explored building a complex application release pipeline with vRealize Code Stream. You learned about gating rules, approvals, automated testing, conditional execution, and viewing pipeline status. vRealize Code Stream is truly the glue that binds together your Software Development Lifecycle (SDLC) processes to increase agility and ensure that processes are repeatable.

You've finished Module 2

Congratulations on completing Module 2.

If you are looking for additional information on integrating software development lifecycle tools with vRealize Code Stream, try one of these:

- View the vRealize Code Stream 2.4 Information Center
- Or use your smart device to scan the QRC Code.

Proceed to any module below which interests you most. [Add any custom(optional information for your lab manual.]

- **Module 1 - Introduction to vRealize Code Stream** (30 minutes) (Basic)
  Explore how vRealize Code Stream leverages your existing software development tools. Review how to implement application release automation for custom developed applications and IT infrastructure.

- **Module 3 - Advanced Pipelining with vRealize Code Stream** (30 minutes) (Advanced)
  Explore the full potential of vRealize Code Stream! Trigger a pipeline execution automatically, work with certified artifacts, and incorporate additional endpoints to fully integrate the code release process.
How to End Lab

To end your lab click on the **END** button.
Module 3 - Advanced Pipelining with vRealize Code Stream (30 minutes)
Introduction

In the previous modules, you learned how to leverage several features of vRealize Code Stream in order to create application release pipelines. However, the functionality demonstrated in modules 1 and 2 is only the beginning of what vRealize Code Stream is capable of.

In this module, you will explore an advanced pipeline that is built on the examples in modules 1 and 2, but also demonstrates additional functionality of vRealize Code Stream. This pipeline will be used first to correct an existing issue with the MyCommerce.com application, and then to enable a new site feature.
vRealize Code Stream Pipeline Features

The previous modules explored several features of vRealize Code Stream. However, the pipelines created in these modules only demonstrate some of the available functionality available within Code Stream pipelines.

Additional pipeline features available in vRealize Code Stream include:

- **Additional Endpoint Integration:** In addition to the Jenkins, vRealize Automation, and vRealize Orchestrator task integration in modules 1 and 2, vRealize Code Stream allows for out of the box integration with Artifactory for artifact management, JIRA for issue creation and tracking, Team Foundation Server for build and test job integration, and more.

- **REST API and Polling Tasks:** In addition to the out of the box integrations above, vRealize Code Stream also allows for creation of tasks leveraging REST API functionality for additional extensibility and integration with software development lifecycle toolsets. Polling tasks can also be created to monitor a URL for specific criteria.

- **Conditional Task Execution:** The creation and use of a pipeline parameter was demonstrated in module 2. Parameters can also be used to configure tasks for conditional execution - for example, only allowing a specific task to be executed based on the specific value of a specified parameter. This allows for pipelines to serve multiple purposes.

- **Parallel Execution:** Tasks within vRealize Code Stream can be configured to execute simultaneously, in order to save time or to streamline the pipeline process.

- **Triggered Pipeline Execution:** vRealize Code Stream pipelines can be executed from external systems via API, and triggered from Git or Gerrit. See the [VMware {code}](https://www.vmware.com) page for more information on Git and Gerrit pipeline triggers.

This module will leverage an interactive simulation in order to show a pipeline that adds some of the additional functionality mentioned here to the pipelines built in modules 1 and 2.
Hands-on Labs Interactive Simulation: vRealize Code Stream Advanced Pipelines

This part of the lab is presented as a Hands-on Labs Interactive Simulation. This will allow you to experience steps which are too time-consuming or resource intensive to do live in the lab environment. In this simulation, you can use the software interface as if you are interacting with a live environment.

1. Click here to open the interactive simulation. It will open in a new browser window or tab.
2. When finished, click the “Return to the lab” link to continue with this lab.

The lab continues to run in the background. If the lab goes into standby mode, you can resume it after completing the module.
Conclusion

In this module, you have begun to explore the capabilities of vRealize Code Stream. You learned about Application Release Automation and modeled a simple pipeline to quickly deploy a test system using vRealize Automation. After modeling this pipeline, you executed it and verified the results. This was a very simple example and we encourage you to explore Module 2 in this lab to see more complex pipelines and use cases.

You've finished Module 3

Congratulations on completing Module 3.

If you are looking for additional information on the more advanced pipeline features of vRealize Code Stream, try one of these:

- Go to the Using Release Automation section of the vRealize Code Stream documentation
- Or use your smart device to scan the QRC Code.

Proceed to any module below which interests you most.

- **Module 1 - Introduction to vRealize Code Stream** (30 minutes) (Basic)
  Explore how vRealize Code Stream leverages your existing software development tools. Review how to implement application release automation for custom developed applications and IT infrastructure.

- **Module 2 - Using vRealize Code Stream for Application Release Automation** (60 minutes) (Advanced)
  Create an application release pipeline with vRealize Code Stream, vRealize Automation, Jenkins, and Artifactory. As a release manager, you will automate the build, test, and deployment of a sample eCommerce application.
How to End Lab

To end your lab click on the **END** button.
Conclusion

Thank you for participating in the VMware Hands-on Labs. Be sure to visit http://hol.vmware.com/ to continue your lab experience online.

Lab SKU: HOL-1921-04-CMP

Version: 20190914-210226