Jenkins Administration

Do the Labs in Order!

Be sure to complete all exercises in the follow-along and do-it-yourself labs in the order they appear in the class! You are building up the configuration of your cluster; later exercises depend on steps take in previous exercises.

"Solutions" are provided for do-it-yourself labs if you need them, but we recommend that you try to do the exercises before looking at the solution.

Local VM: Vagrant + Virtualbox

A Virtual Machine (VM) is used for hosting your **Lab Environment**:

- It does not break any local environment
- It does not depend on the host operating system you are using
- It is portable with the same behavior for everyone

This VM runs using the VirtualBox hypervisor, and is managed and automated by Vagrant (see requirements below).

Common requirements

An HTML5 compliant web browser is required: **Mozilla Firefox**, **Google Chrome**, **Microsoft Edge**, **Apple Safari**, or **Opera**.

Internet Explorer is not supported.

The following ports must be allowed access to your instance’s domain:

- 5000
- 5001
- 5002
- 5003
- 20000
- 30001
- 30002

The following protocols must be allowed by any antivirus/firewall software:

- HTTP
- HTTPS
- Websockets

Software requirements

Your machine must meet the following **software** requirements:

- For **All** operating systems, download and install the latest (64 Bits) versions of:
  - *VirtualBox* (An Open Source Hypervisor from Oracle):
    - Downloads page: https://www.virtualbox.org/wiki/Downloads
  - Vagrant *(An Open Source VM manager):*
    - Downloads page: https://www.vagrantup.com/downloads.html
- For **Windows** only, download latest version of Git for Windows

Getting lab resources

After installing the software prerequisites:

- Right click this link to the virtual machine's **ZIP** archive to open it in a new tab or window. When you click **OK**, the archive downloads to your local disc.
- Extract the virtual machine **ZIP** archive to your local disc. This archive contains your virtual machine image and automated settings in a folder named `jenkins-training`.

Starting the lab environment

- Open a command line on your host operating system:
  - On Windows, open **Start Menu** , **Git Bash**
  - On Linux, this can be named **Command Line** or **Terminal**

The command line is required to start the Virtual Machine without having to care to any specific configuration.

- Using the command line `cd`, navigate to the un-archived folder that should be located on your Desktop:

```
cd ~/Desktop/jenkins-training/
```

- The `~` special character means "Full path to the user home folder"
- `Desktop` may vary depending on your operating system; it can be lower case, or localized in your operating system's language.

- Use the command line `ls` to check the content of this directory. We need to have a file named `Vagrantfile` here:
-
```
ls -1
  Vagrantfile
```

- Now you are able to start the Virtual Machine, using the `vagrant` command:

```
vagrant up
```

You must be able to stop and start the Virtual Machine whenever you want:

- From the jenkins-training folder that contains a `Vagrantfile`:
  - Stop the VM "gracefully" with the vagrant "halt" command:

    ```
    vagrant halt
    ```

    Once the VM is in the *stopped* state, you can safely do anything else, like stopping your computer

  - Restart the Virtual Machine:

    ```
    vagrant up
    ```

Any *Vagrant* command can be used here. For more informations, please check [Vagrant Documentation - https://www.vagrantup.com/docs/cli/]

Accessing the lab environment

Your **Lab Environment** provides a Home Page to use as the entry point. This page is available on your web browser when you open your lab instance.

```
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Clearing any previously set forwarded ports...
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
==> default: Forwarding ports...
    default: 5000 (guest) => 5000 (host) (adapter 1)
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Running 'pre-boot' VM customizations...
==> default: Booting VM...
```

Opent it using this URL : http://localhost:5000

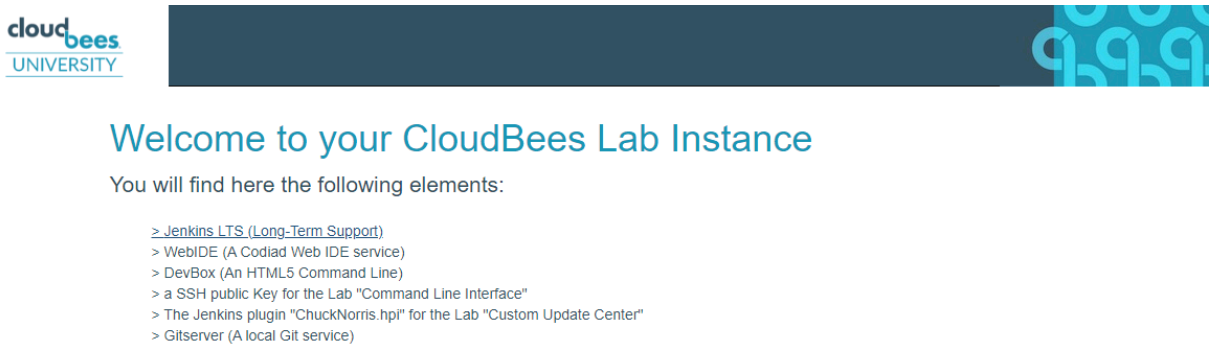
Unless specified differently, any authentication asked by any service in the Lab Environment uses the following:

- Username: butler
- Password: butler

You will see an HTML page that lists the services hosted on your Lab Environment.

Meet your lab instance

Lab home page



When you click a link from this page, it opens in a new tab.

These links are to work environments:

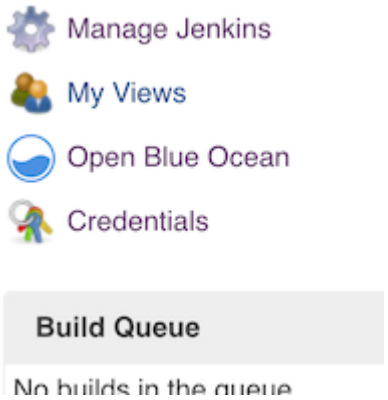
- *Jenkins controller* — The Jenkins controller dashboard; this is the environment you will use for most of the work in this course.
- *Gitserver* — Git repository page for the projects in this course. If you are familiar with GitHub, this page will look familiar even though it is actually running Gitea. Gitea provides a local advanced Git server with a web interface from which to browse repositories, authenticate, do pull requests, and perform reviews.

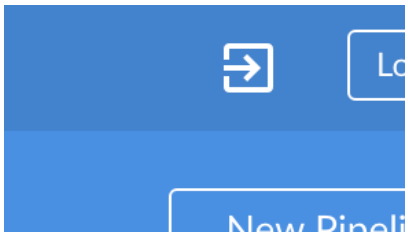
Enter the username `butler` and the password `butler` to login to the Jenkins controller.

Blue Ocean

The Blue Ocean plugin is installed in your lab environment. To open it:

- Open the Jenkins controller dashboard
- Select **Open Blue Ocean** in the left frame:



- **You just got CD-in-a-box!**
- Switch to Classic Web UI.
  - Click the arrow button to switch to "Jenkins Web UI":
    - 
  - Click **Open Blue Ocean** in the side bar to switch back.

The icon to "Switch to Classic Web UI" is available at the top of most pages in Blue Ocean.

Configure system

Use the **Manage Jenkins System Configuration** page to configure many standard aspects of your Jenkins server, including:

- JDK installations
- Build tools - Ant and Maven installations
- Version control tools
- Email configuration

Plugins could add sections to this page.

General system-wide configurations

Use the top of the **System Configuration** page to set basic information about the controller:

- **Home Directory** — Display the Jenkins home directory. To change this location, modify the value of the `JENKINS_HOME` environment variable.
- **System Message** — Text that is displayed at the top of your Jenkins home page. Use this page to display the name of the server and a short description of its purpose. It is also used to display announcements about scheduled down time and other information of interest to all Jenkins users.

  This field can include HTML tags or a pointer to a Wiki page and information about who to contact.

Inline help is available on most **Manage Jenkins** pages. To access the help, select the **?** icon to the right of each field. For example, the help screen for the **System Message** box is shown here.

Click the **?** icon again to hide the help text.

Executors on controller

The **# of executors** field configures the number of executors on the controller. Typically, this should be set to 0 to prevent builds from running on the controller. We will discuss executors for nodes and agents in a later section.

Quiet period

This field defines an interval to wait before actually starting a build. This can be used to collapse multiple adjacent commits into one build.

Jenkins location

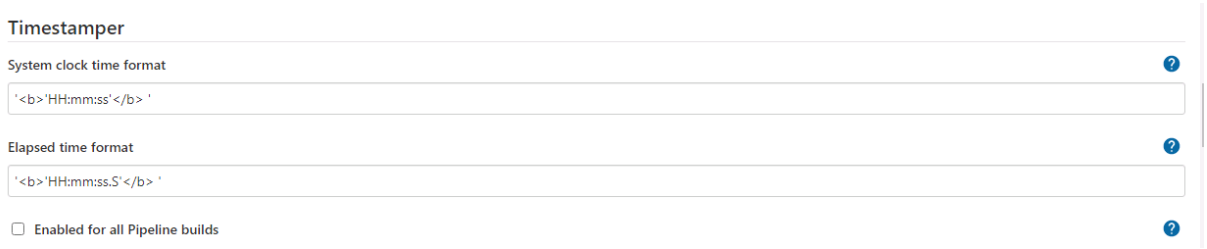
You must configure the **Jenkins Location** area to identify this particular instance of Jenkins.

Two items must be defined here:

- **Jenkins URL** — Set to the location used to access this Jenkins instance. This information can be:
  - The actual URL that people use to access Jenkins.
  - The host alias as defined by the name service. If your IT team cannot provide this, use dynamic DNS. This makes your instance easier for your users to access and makes your service relocatable.
- **System Admin e-mail address** — Define the email address to which email about general Jenkins issues are sent. This can be an individual's email address or an email alias that contains multiple email addresses.

Timestamper

The **Timestamper** fields define the format used to display timestamp information:



More about system configuration

Jenkins is very flexible and powerful. Default configurations can be used for most fields while you are getting started. As you learn more about how Jenkins works and how your project uses Jenkins, you can modify the values of settings on this page. Use these settings to fine-tune the behavior of your Jenkins instance.

When you are done modifying this screen, scroll to the bottom and click **Save** to apply your configuration changes.

Provide build environments

Administrators must configure build environments that Pipeline developers can use. Two main elements are:

- Configure the tools that Pipeline developers can use.
- Configure the nodes and agents on which Pipelines execute.

Management options

Select **Managment Options** to start, stop, and reload the Jenkins service.

To reload the configuration from disk, Jenkins rereads the configuration from `$JENKINS_HOME` without restarting Jenkins.

The **Prepare for Shutdown** option prevents new jobs from being started and waits for running jobs to complete. Use this option to ensure a clean job termination before you shut down or restart Jenkins.

The Management page provides Jenkins Upgrade management. If you are running a standalone server, you can perform a one-click upgrade. Otherwise, you have a direct link to the latest WAR.

Manage Old Data helps to handle dropped configuration items after an upgrade.

Global tool configuration

Select **Manage Jenkins Global Tool Configuration** to configure the tools used for Pipeline development.

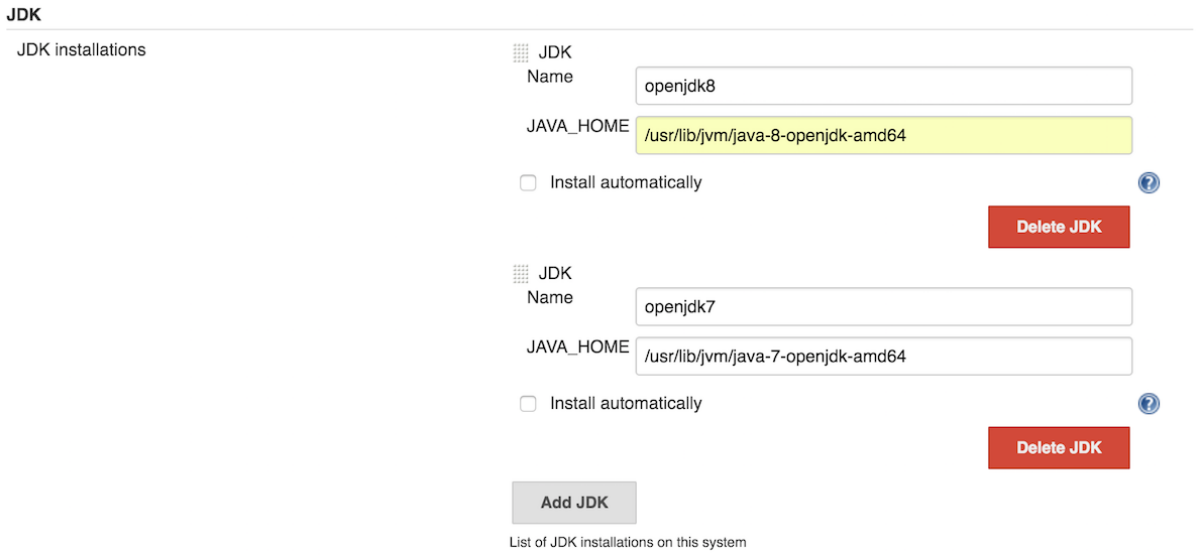
Some tools are listed here by default, while others are listed only if the appropriate plugins are installed.

- JDK and other languages
- Build tools: Maven, Gradle, Ant and others
- Source Code Management: Git, Mecurial and others
- Containers: Docker and Kubernetes (when installed)

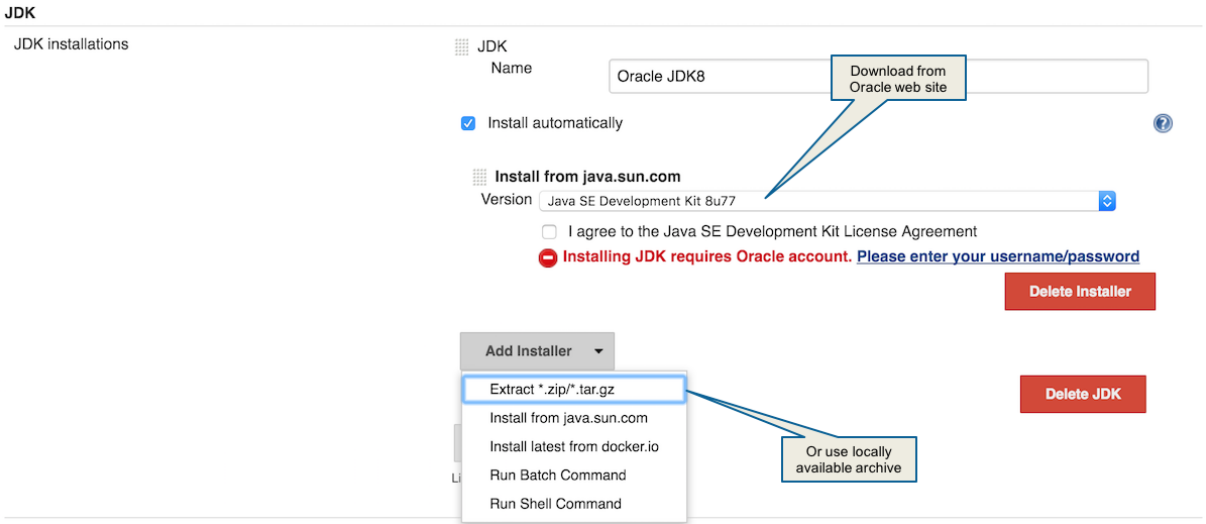
When you select the **Add** button for a tool, Jenkins provides fields where you can supply the information required to install that tool. Many tools allow you to configure multiple versions, and some tools also support auto installation.

JDK

Use the **JDK** section to define the JDK versions you need for your projects. You can define multiple versions for different projects.



You can also install JDK versions automatically.



Maven

Use the **Maven** section to automatically install Maven.