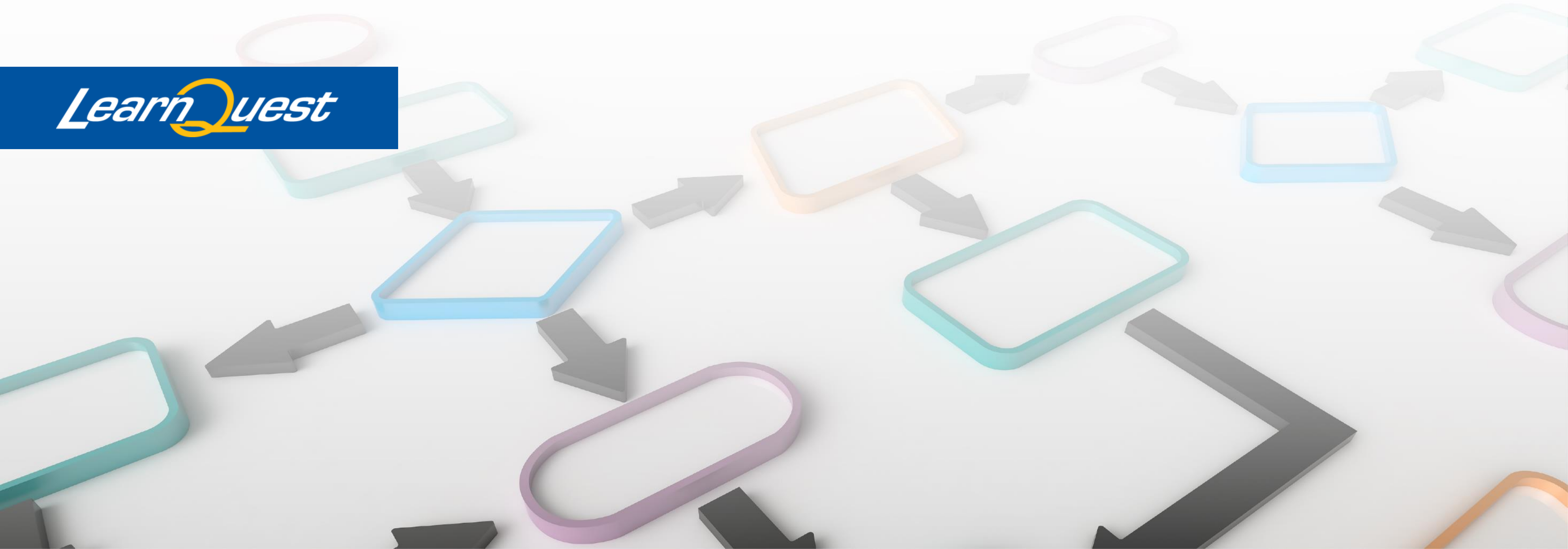
Linux Cloud and DevOps

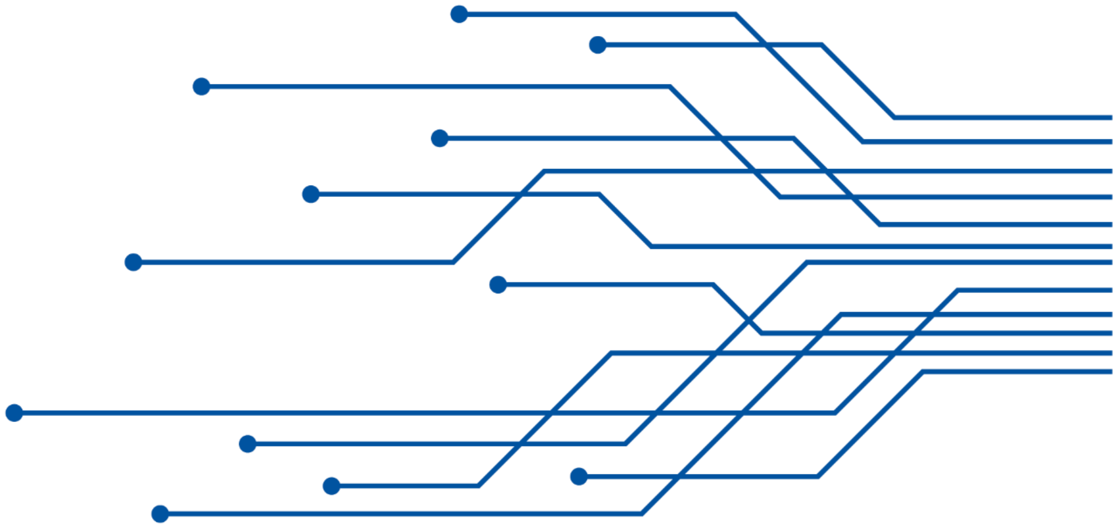
4th Course in Linux Foundations Specialization



Version Control

In this module, we look at how we can manage versions of source control in the cloud using the Git version control system.

3



Learning Objectives

Version Control

Upon completion of this module, learners will be able to:

- Describe Version Control
- Commit Source Code with Git
- Merge Versions with Git

Lesson 1

Version Control

In this lesson we look at what Version Control is and what it is used for

Version control

A method or system that organizes various project files and protects modifications to them

Version control system (VCS) provides a common central place to store and merge project files, so latest project version is accessible

Git

- Created by Linus Torvalds (creator of Linux)
- Distributed VCS

Git Components

Working Directory - Typically a home subdirectory where all source files are created, modified, and reviewed

Staging Area - Hidden subdirectory named .git

- Created by git init command
- Working directory source files are registered into this area via git add command

Local Repository - Contains each project file's history

Remote Repository - Typically a cloud-based location

Popular Remote Repositories

- GitHub
- GitLab
- BitBucket
- Launchpad

Lesson 1 Review

The working directory has the local copy of source files

The remote repository holds the permanent copy of source file and versions

GitHub is a popular Git Repository

Lesson 2

Committing Changes

In this lesson we look at Committing Changes to a repository

Setting up the Local Git Environment

1. Create a working directory
2. Initialize the .git/ directory
3. Set up local repository options
4. Establish your remote repository

Committing Source Files with Git

Create or modify	Create or modify the source file(s)
Add	Add the file(s) to the staging area
Commit	Commit the file(s) to the local repository
Push	Push the file(s) to the remote repository

Git Configuration Commands

```
git config --global user.name "[firstname lastname]"
```

- set a name that is identifiable for credit when reviewing version history

```
git config --global user.email "[valid-email]"
```

- set an email address that will be associated with each history marker

```
git config --global color.ui auto
```

- set automatic command line coloring for Git for easy reviewing

Git Setup Commands

git init

- initialize an existing directory as a Git repository

git clone [url]

- retrieve an entire repository from a hosted location via URL

Git Commit Commands

git status

- show modified files in working directory, staged for your next commit

git add [file]

- add a file as it looks now to your next commit

git commit -m "[descriptive message]"

- commit the staged content as a new commit snapshot

Lesson 2 Review

Git add puts the file in the queue for the next commit

Git commit pushes the staged content into a new snapshot

A snapshot is just the Git term for a revision

Lesson 3

Branches

In this lesson we look at Git Branches

Git Snapshot Command

git reset [file]	git diff	git diff –staged
• unstage a file while retaining the changes in working directory	• diff of what is changed but not staged	• diff of what is staged but not yet committed

Git Branches

An area within a local repository for a particular project section

By default, Git stores work in the master branch

Can have multiple branches for a project. An example is:

- Master – production software
- Development – software being developed
- Test – software being tested

Git Branch Commands

git branch

- list your branches. an * will appear next to the currently active branch

git branch [branch-name]

- create a new branch at the current commit

git checkout

- switch to another branch and check it out into your working directory

git merge [branch]

- merge the specified branch's history into the current one

git log

- show all commits in the current branch's history

Lesson 3 Review

Unstaging is the term for changes in Git but not marked for commit

A Git branch is an area in a project

A merge conflict happens when the same part of a file is changed differently