Version Control Systems

Student Workbook 1b – Git and GitHub

Version 7.0 Y

Table of Contents

Module 1

Version Control System and Git

Section 1–1

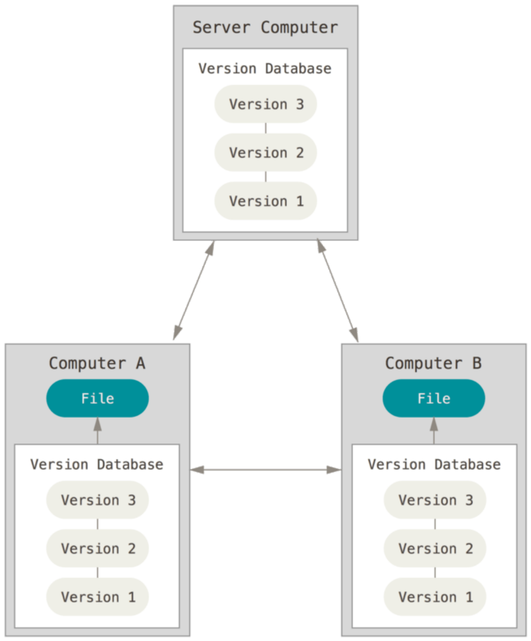
Git

What is Version Control?

- **Version control systems help developers keep track of changes to their source code over time**
  - Commonly referred to as SCM (***Source Code Management***)
  - Or VCS (***Version Control Systems***)
- **The idea is to keep track of every modification made to the code base**
- **You "commit" your changes and store them in a repository**
  - Each time you add "something that works" you generally do a commit
  - This might be a several times per hour if you are productive
- **VCS features allow you to roll back to a previous commit if needed**
  - You can also "roll forward" if you've already rolled back
- **Tools also let you compare code in different commits**
  - It can help you figure out what changes might have caused a bug
- **VCS allows teams of developers to collaborate on one code base**
- **Version control systems can be:**
  - centralized
  - distributed

Version Control Systems

- **Distributed Version Control systems like Git do *not* rely on a central server to store all the versions of a project's files**
  - Each developer has a copy of the repository and with all versions
- **Cloud based services like GitHub, GitLab and BitBucket can provide a centralized (additional) copy of a repository**
  - To work on a project, each developer could create a repository on their local machine or "clone" an existing repository from the cloud service
- **The developer's local repository would contain the full history of the project**
  - A developer can make changes to the code and commit those changes to their local repository
  - When they are ready to share changes with other developers, they "push" their changes and all the attached history to the remote repository
  - Once pushed, other developers can now update their local copies with those changes.



- **Typical Workflow for Git**
  - Developer checks out code from version control or pulls down the latest changes to code already checked out
  - Developer makes some changes and tests their work
  - Developer makes a local "commit" that represents the changes made
  - Developer repeats this cycle locally until they are ready to share their changes with other developers
  - Developer commits changes back to the central repository for others to "pull" into their local copies or provide a patch file to another developer if they choose not to use a service like GitHub

Section 1–2

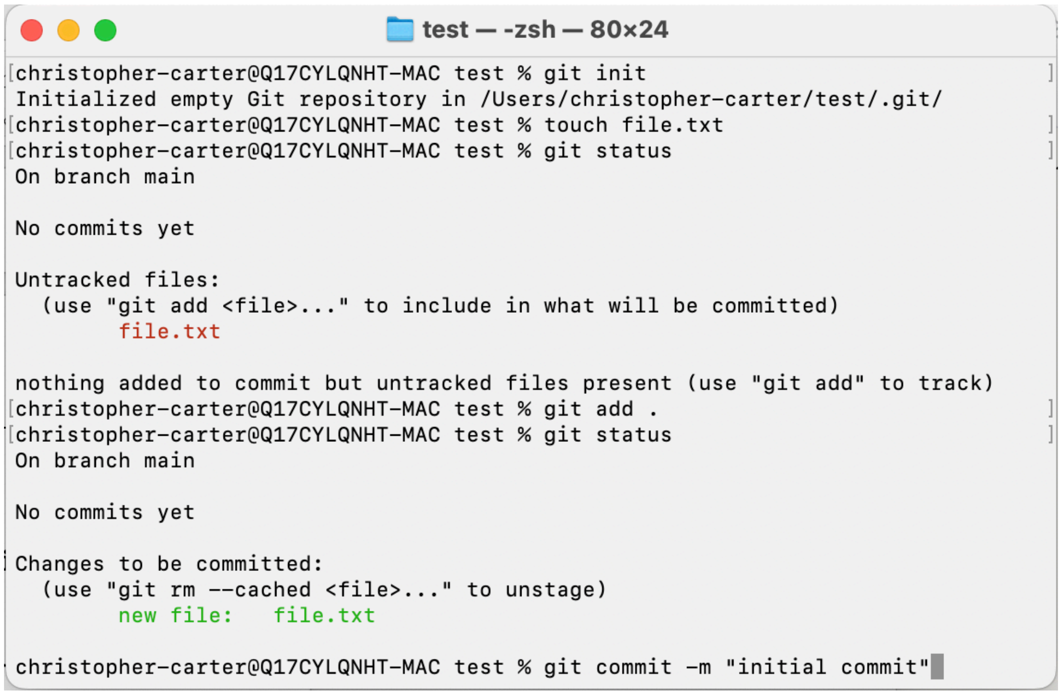
Git Basics and the Local Repository

Git

- **Git is a free, open source distributed version control system**
  - It can handle both small and very large projects
- **It is easy to learn and has a tiny footprint**
  - Most importantly, it is very fast
- **Before you start using Git, install it on your developer machine from `https://git-scm.com/downloads`**
  - Note: Your machine should have been configured before the class started and Git will already be on it
- **To run Git commands, you can use:**
  - Git Bash shell, terminal, a bash shell,
  - or the Windows command prompt window
- **NOTE: We will use IntelliJ's git tooling**
  - Afterwards, you can use Git Bash or the Command Prompt window. Choose whichever you prefer or your company recommends

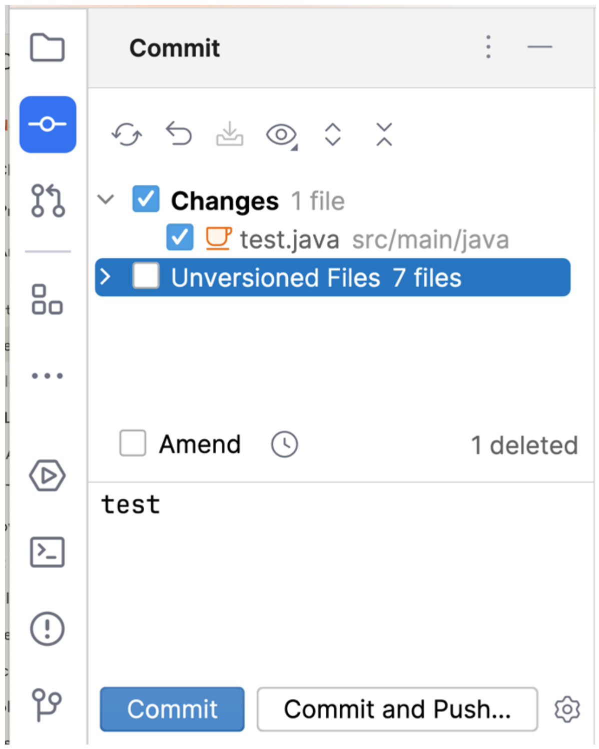
CLI vs IDE vs GUI

- **Command Line Interface (CLI)**
  - Direct interaction with git through terminal commands like `git add`, `git commit`, `git push`
  - Provides complete access to all git features and advanced functionality
  - Requires memorizing commands but offers precise control and is consistent across all operating systems
  - Essential for automation, scripting, and troubleshooting complex git issues

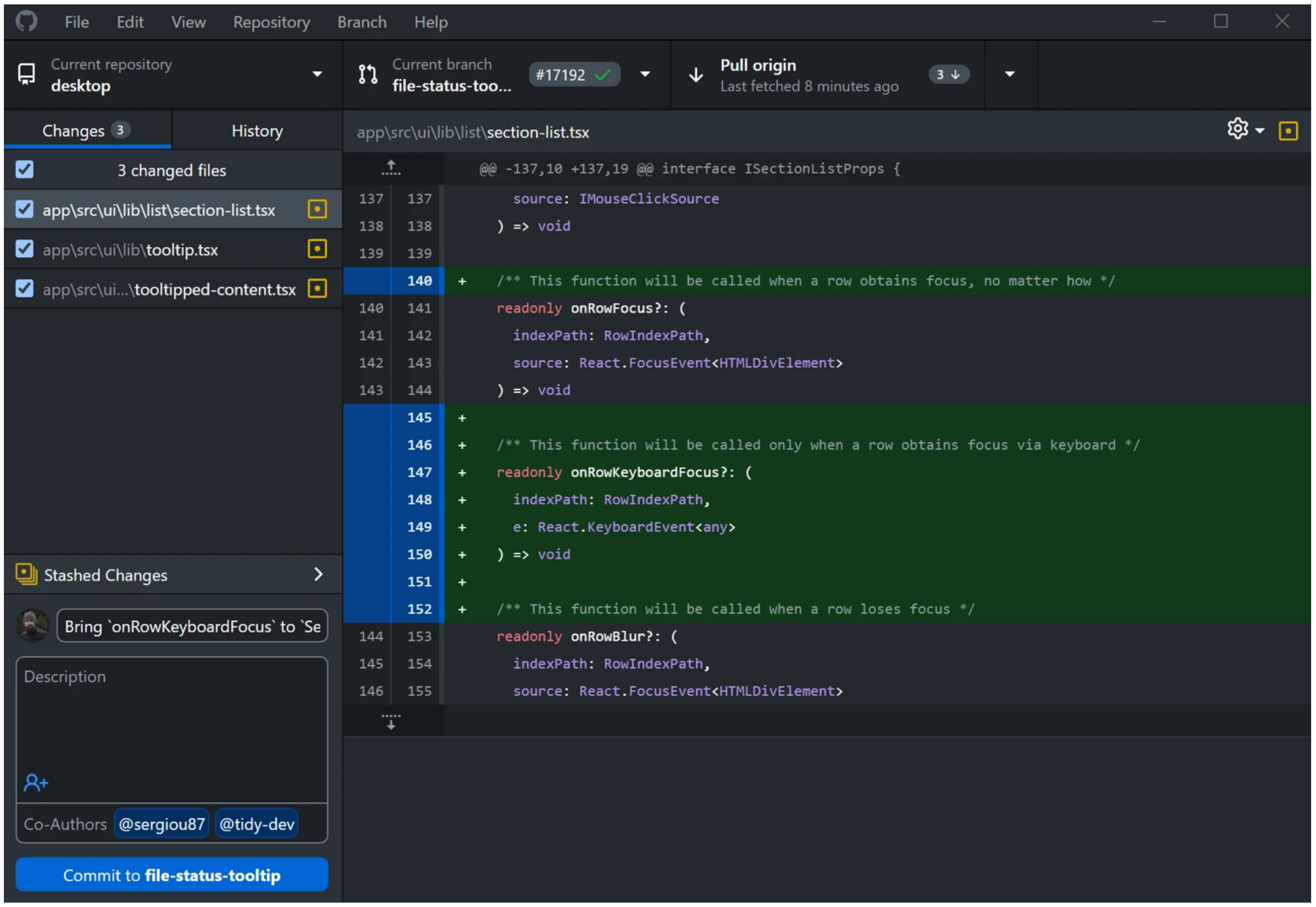


```
christopher-carter@Q17CYLQNHT-MAC test % git init
Initialized empty Git repository in /Users/christopher-carter/test/.git/
christopher-carter@Q17CYLQNHT-MAC test % touch file.txt
christopher-carter@Q17CYLQNHT-MAC test % git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file.txt

nothing added to commit but untracked files present (use "git add" to track)
christopher-carter@Q17CYLQNHT-MAC test % git add .
christopher-carter@Q17CYLQNHT-MAC test % git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file.txt

christopher-carter@Q17CYLQNHT-MAC test % git commit -m "initial commit"
```

- **Integrated Development Environment (IDE)**
  - Git functionality built directly into code editors like VS Code, IntelliJ, or Visual Studio
  - Seamlessly integrates version control with your coding workflow
  - Offers visual indicators for file changes, inline blame annotations, and commit history within your editor
  - Combines the convenience of GUI tools with the context of your actual code



- **Graphical User Interface (GUI)**
  - Standalone visual applications like GitHub Desktop, GitKraken, or Sourcetree
  - Uses buttons, menus, and visual representations instead of text commands
  - Makes complex operations like merge conflict resolution and branch visualization more intuitive
  - Great for beginners or visual learners who prefer point-and-click interactions over command memorization


File
Edit
View
Repository
Branch
Help
Current repository
desktop
Current branch
file-status-too...
#17192
Pull origin
Last fetched 8 minutes ago
3
Changes 3
History
app\src\ui\lib\list\section-list.tsx
3 changed files
app\src\ui\lib\list\section-list.tsx
app\src\ui\lib\tooltip.tsx
app\src\ui...\tooltipped-content.tsx
Stashed Changes
Description
Commit to file-status-tooltip

Bring `onRowKeyboardFocus` to `Se

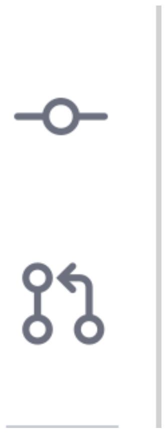
Co-Authors @sergiou87 @tidy-dev

```
@@ -137,10 +137,19 @@ interface ISectionListProps {
      source: IMouseClickSource
    ) => void

+   /** This function will be called when a row obtains focus, no matter how */
    readonly onRowFocus?: (
      indexPath: RowIndexPath,
      source: React.FocusEvent<HTMLDivElement>
    ) => void
+
+   /** This function will be called only when a row obtains focus via keyboard */
+   readonly onRowKeyboardFocus?: (
+     indexPath: RowIndexPath,
+     e: React.KeyboardEvent<any>
+   ) => void
+
+   /** This function will be called when a row loses focus */
    readonly onRowBlur?: (
      indexPath: RowIndexPath,
      source: React.FocusEvent<HTMLDivElement>
```

Managing our Projects using the IntelliJ IDE

- IntelliJ provides built-in git support with visual indicators showing file status changes (by default it's green for new, blue for modified, red for conflicts)

- We can access git operations through the VCS menu, right-click context menus, or the Git tool window at the left of the screen



- We can view commit history, create branches, and merge changes directly within our development environment

Section 1–3

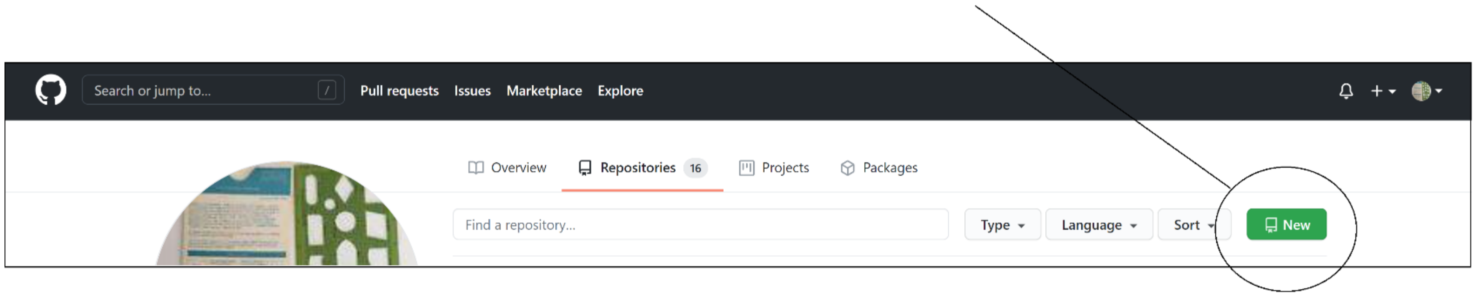
GitHub

GitHub - A Remote Repository Service

- **GitHub is a cloud-based service that can be used to store and manage remote git repositories**
  - Think of it as a backup for your local repo (!)
- **In addition to providing cloud storage for your repositories, it lets you to make your Git repositories available to other developers**
- **Anyone can sign up at GitHub and host public code repositories for free**
  - This makes GitHub very popular as a hosting site for open-source projects
  - Sign up for an account at: `https://github.com`
- **GitHub is a for-profit company and makes money by selling hosted private code repositories and other team-based development services**
- **Many organizations host their own internal GitHub cloud service so that they have complete control over its visibility**
- **However, we will use the public GitHub in this class**

GitHub User Interface

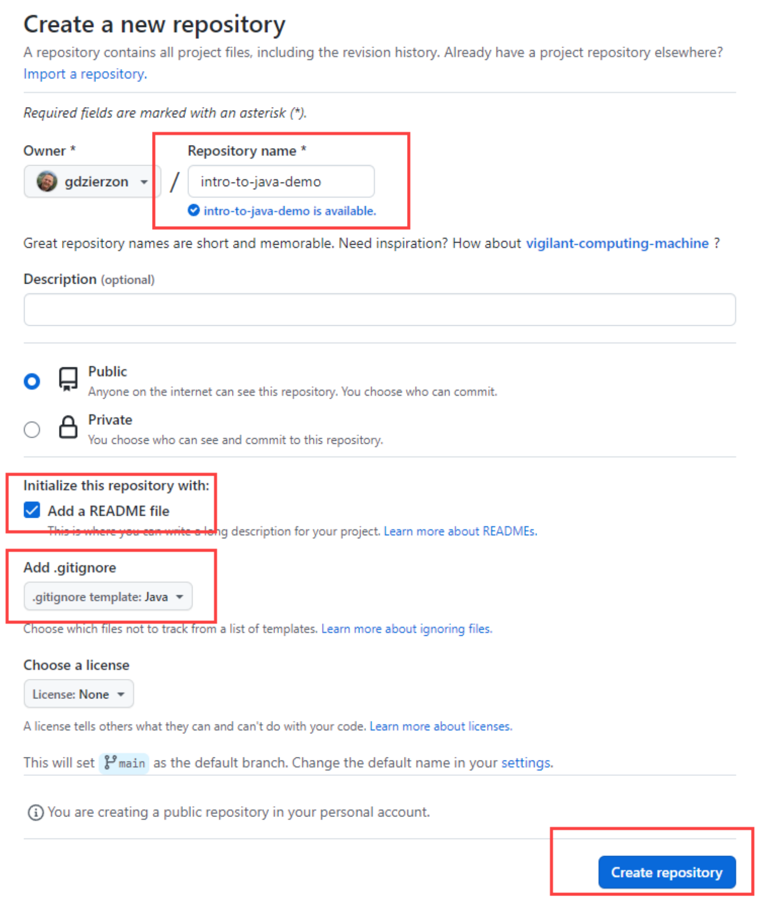
- **GitHub has a web-based graphical interface**
  - It allows you to create new repositories easily



- **It has opinions on how to grant access to your code as well as how people can contribute to your code**

- **It provides several collaboration features for your project, such as:**
  - basic repo management
  - wikis

Creating a Remote Repository


Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Owner * gdzierzon / Repository name * intro-to-java-demo

intro-to-java-demo is available.

Great repository names are short and memorable. Need inspiration? How about vigilant-computing-machine ?

Description (optional)

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Initialize this repository with:

[✓ Add a README file] This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore

.gitignore template: Java

Choose which files not to track from a list of templates. Learn more about ignoring files.

Choose a license

License: None

A license tells others what they can and can't do with your code. Learn more about licenses.

This will set `main` as the default branch. Change the default name in your settings.

You are creating a public repository in your personal account.

- **After clicking the New button, you can:**
  - Name the repo
  - Set the visibility of the repo (public / private)
  - Accept the default main branch name as 'main' or change it to something else
  - Until recently, it defaulted to 'master'
- **There may be small differences on your internal GitHub**

Exercise – My First Repo

In this exercise, you will create a GitHub repository

EXERCISE 2

Step 1: In your browser navigate to `https://github.com` and create a new repository with the following settings:

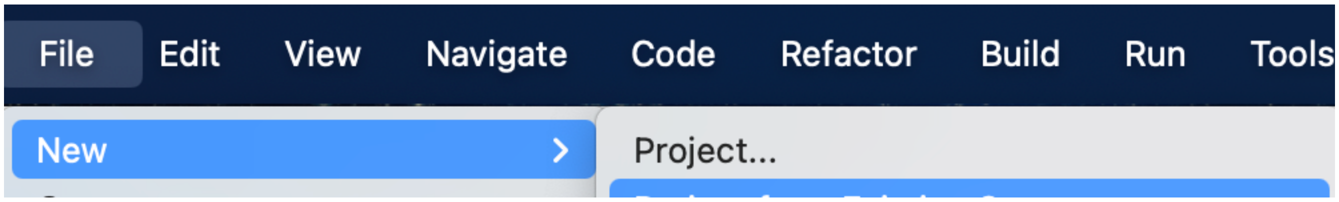
- Name: My-First-Repo
- Add a description
- Have no other selections made

Step 2: Send a link to your GitHub repo to your instructor

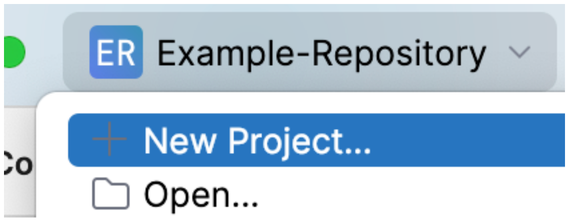
Creating our First Local Repository

- **We will start by creating a new project using IntelliJ**
  - For this project, we won't create any files or write any code!

- **Either select the new Project, under the File dropdown menu**

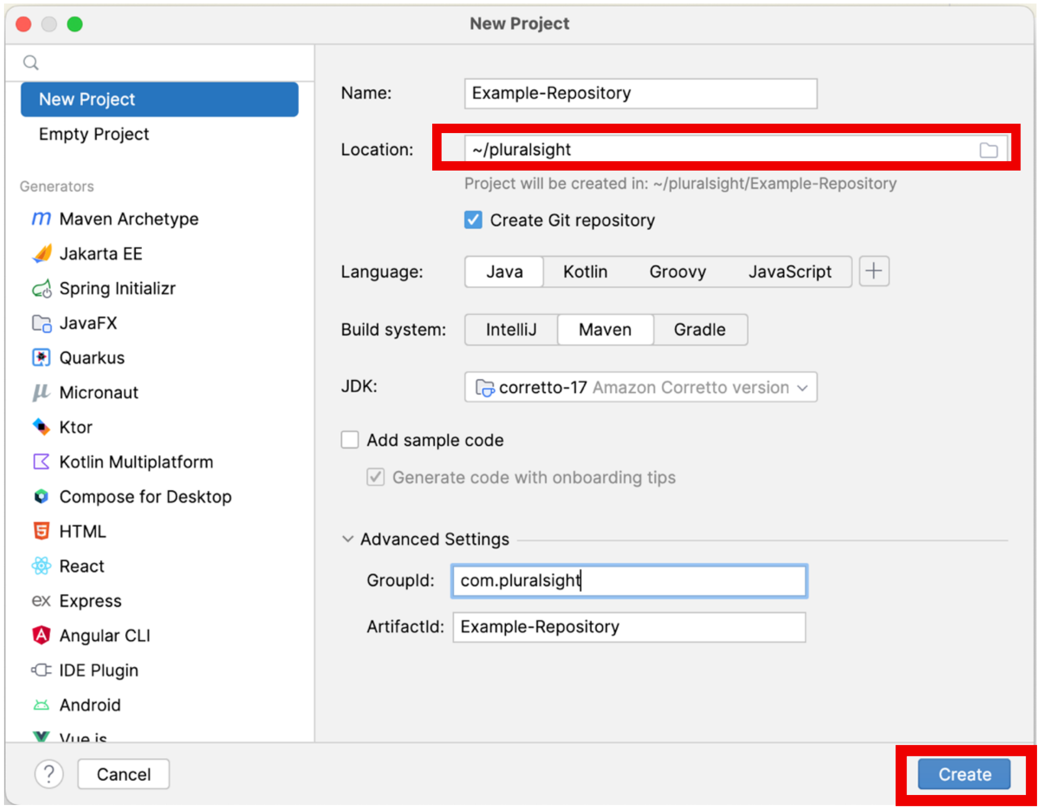


- **Or select the project you are on, and click 'New Project…"**



- **Within the New Project Menu, ensure you have a name, location, and 'Create Git repository' selected**

- **The Location should be:**
  - Mac/Linux: `~/pluralsight`
  - Windows: `C:/Users/[username]/pluralsight`
  - The Language: Java
  - Build System: Maven
  - JDK: corretto-17 Amazon Corretto



New Project

Name: Example-Repository

Location: ~/pluralsight

Project will be created in: ~/pluralsight/Example-Repository

☑ Create Git repository

Language: Java | Kotlin | Groovy | JavaScript

Build system: IntelliJ | Maven | Gradle

JDK: corretto-17 Amazon Corretto version

☐ Add sample code

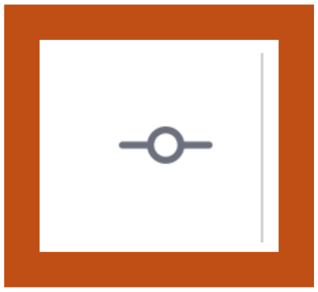
☑ Generate code with onboarding tips

Advanced Settings

GroupId: com.pluralsight

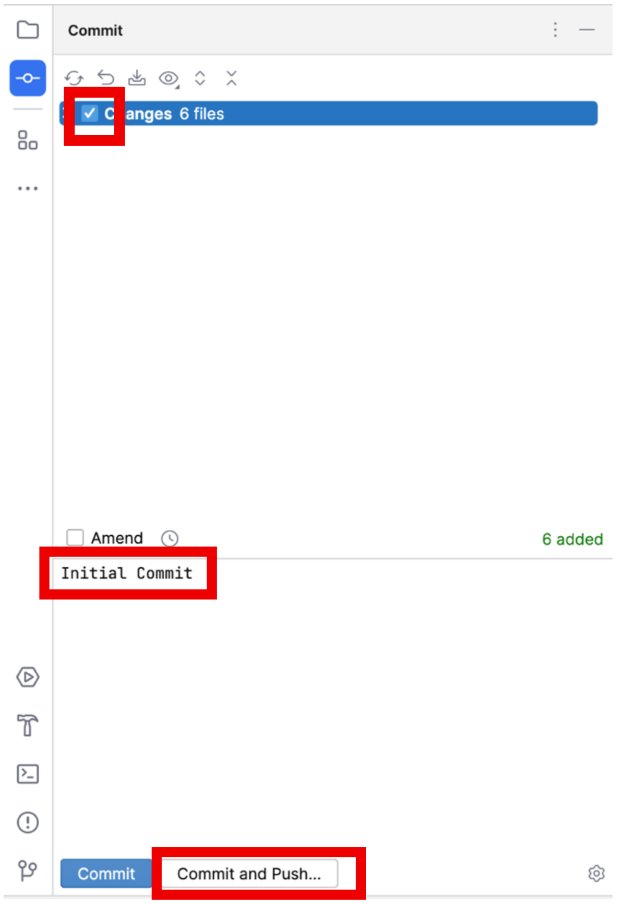
ArtifactId: Example-Repository

- **On the right side menu, click the 'commit' button**



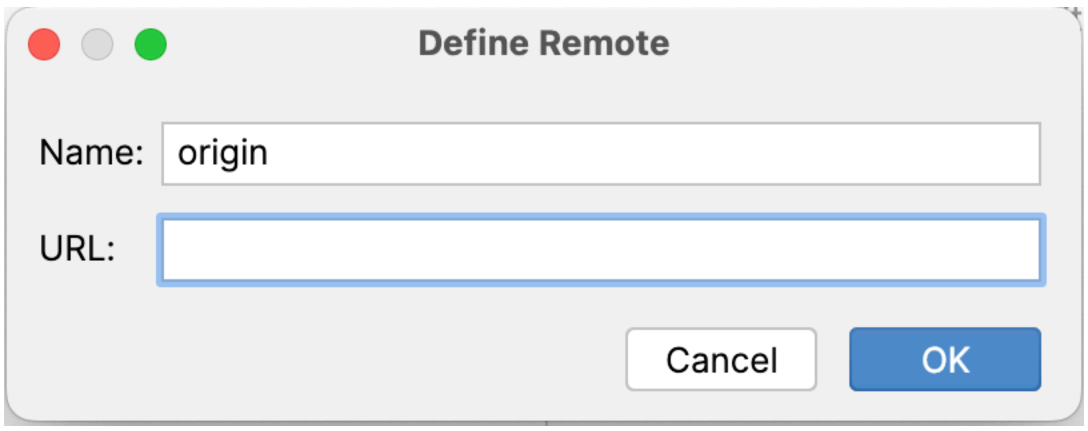
- **Select the changes and click commit and push**

- **Write a commit message**



Initial Commit

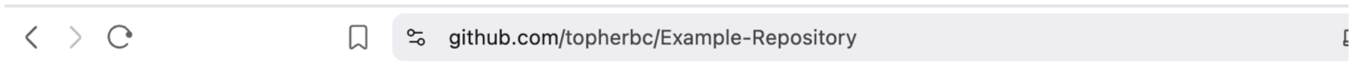
- **This will ask you to connect your Github.com repository you setup earlier**



Name: origin

URL:

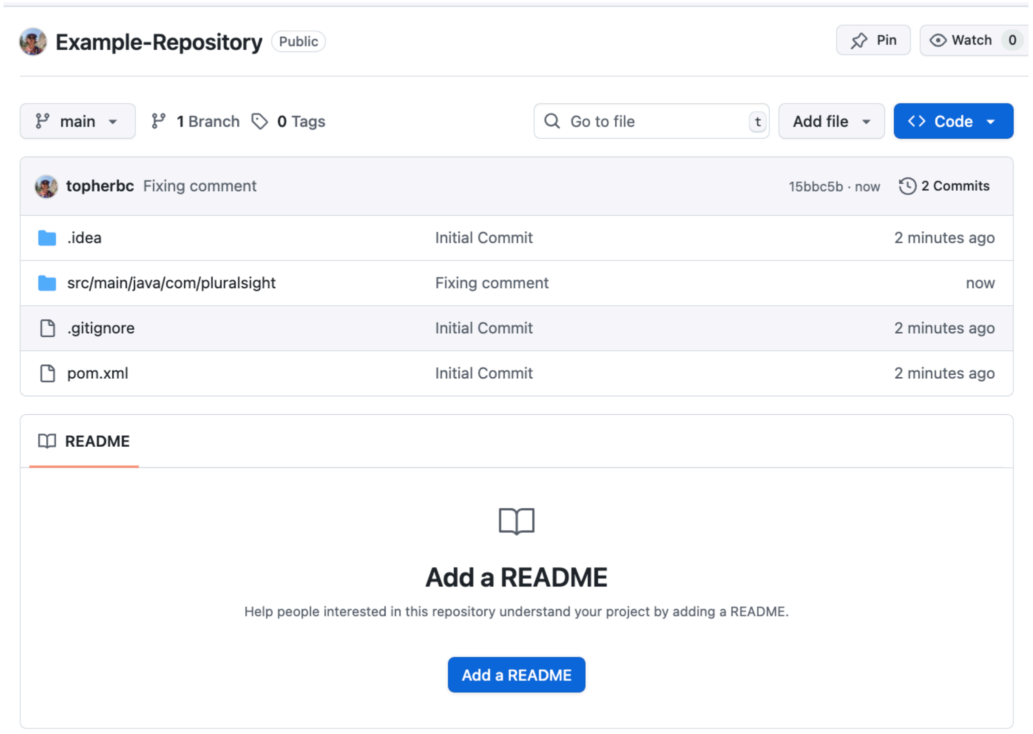
- **Grab the URL from your browsers address bar**



github.com/topherbc/Example-Repository

- **And paste this in the pop-up window in IntelliJ**

- **Now, just refresh and see your changes!**



topherbc Fixing comment 15bbc5b · now 2 Commits

.idea	Initial Commit	2 minutes ago
src/main/java/com/pluralsight	Fixing comment	now
.gitignore	Initial Commit	2 minutes ago
pom.xml	Initial Commit	2 minutes ago

Add a README

Help people interested in this repository understand your project by adding a README.

Exercise – Project Upload

In this exercise you will connect a local project to github.com

- **Step 1:** Start a new project in IntelliJ

    Within the New Project Menu, ensure you have a name, location, and **'Create Git repository'** selected
    The Location should be:

    Mac/Linux: ~/pluralsight

    Windows: C:/Users/[username]/pluralsight

    The Language: **Java**

    Build System: **Maven**

    JDK: **corretto-17 Amazon Corretto**

Step 2: Make your initial commit and push

Step 3: Add the GitHub URL from the previous project, My-First-Repo

Step 4: Refresh the repository page on github.com

Step 5: Send your Instructor a message letting them know you've updated the repository with your first commit