

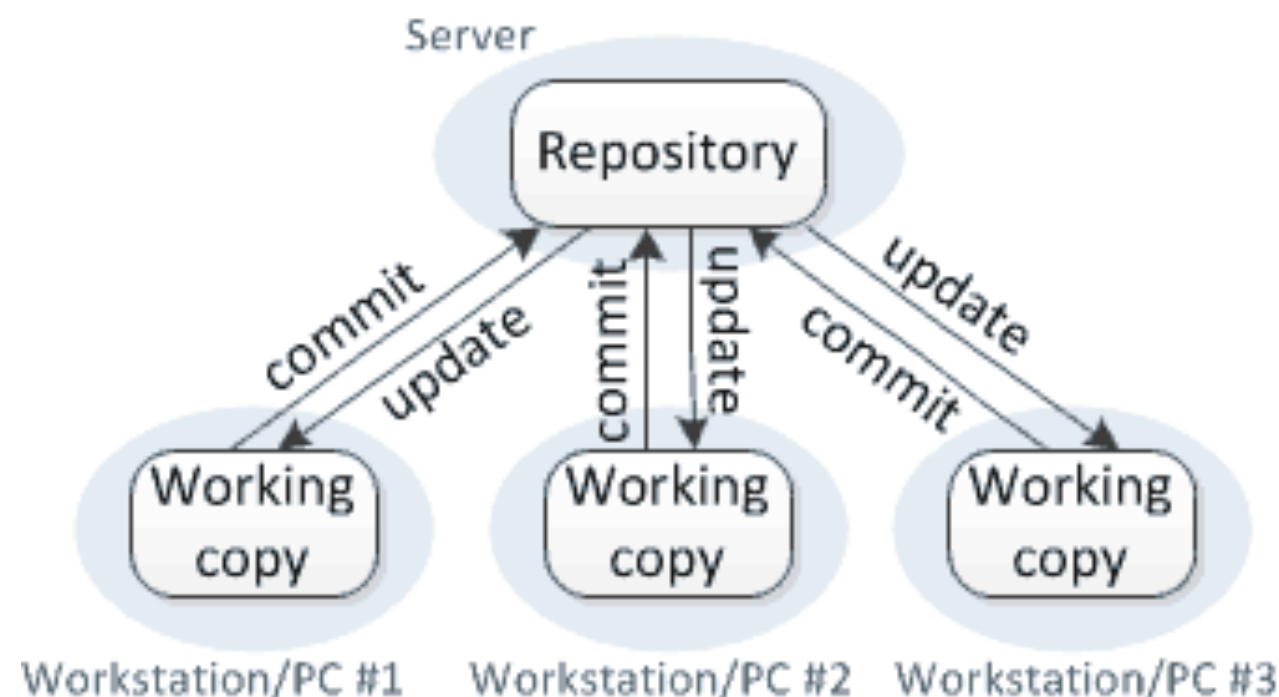
Discussion

- We watched a few videos from the following course on Udacity
 - [Version Control With Git](#)

GIT

Version Control

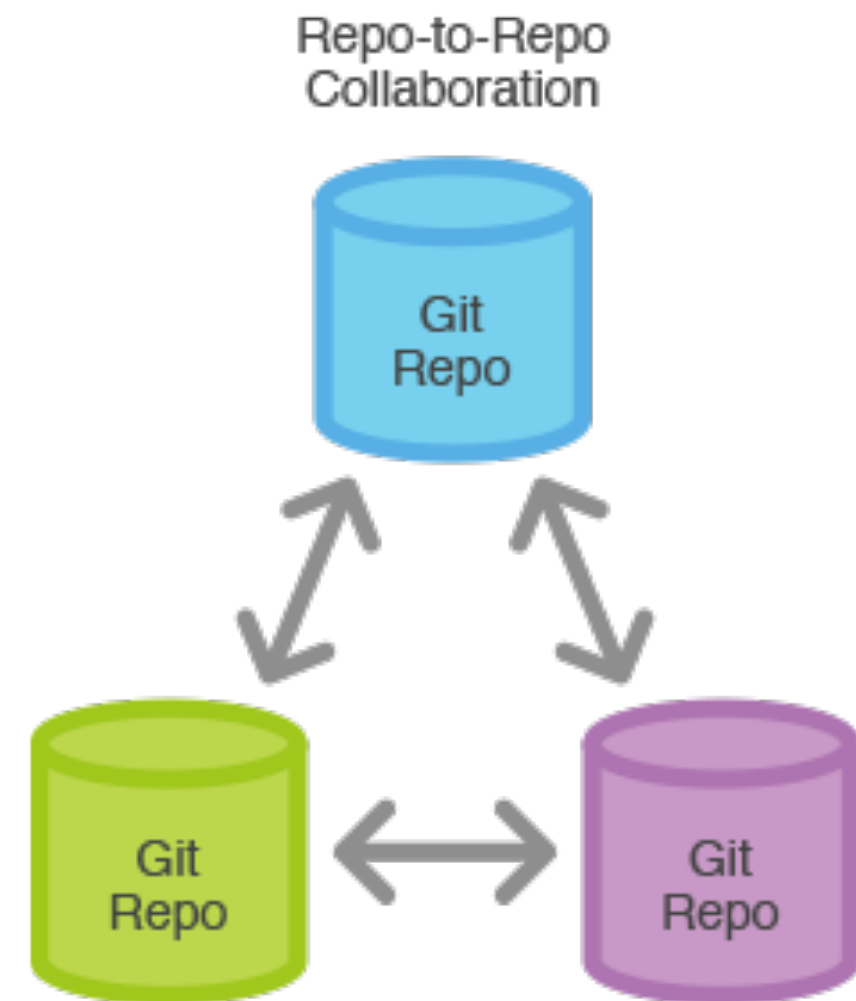
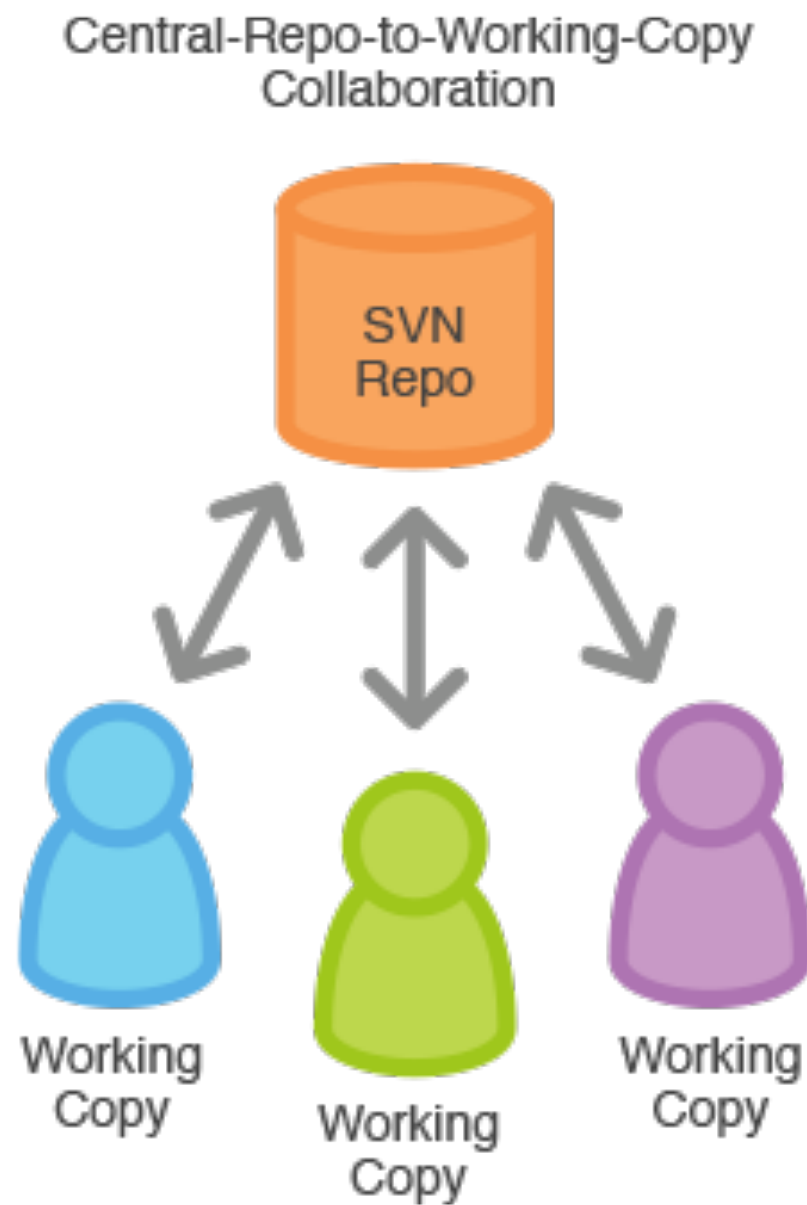
- Collaborative software development necessitates a system for source control and versioning.
 - Facilitates simultaneous updates to the same software.
 - Manages any conflicts created.



Version Control

- There are a few options when it comes to version/source control:
 - **CVS** - Concurrent Versions System
 - **SVN** - Apache Subversion
 - **GIT** - a distributed VCS (*not the only one!*)

Version Control



Git

- Developed by Linus Torvalds, ~2005
- Open source, under the GNU GPL (General Public License)
- **Distributed**, in that every directory on every computer is a full-fledged repository.
 - It has a complete history, version-tracking capabilities.
 - Independent of network access, or a central server

Git

- Use as command-line or via their GUI.
 - EGit is a Git plugin for Eclipse.
 - GitGUI uses GitK

Git Terminology

- Repo - repository
- Init - initialize a repository
- Push/Pull
- Commit
- Clone - copy a remote repo locally
- Branch

Creating a Git Repo

1. Create a new project directory.
2. Move into that new directory.
3. Initialize Git directory.
4. Create/edit project files.
5. Track files in the project directory.
6. Commit tracked files in Git storage.

Creating a Git Repo

1. Create a new project directory.

```
mkdir newProjectDir
```

2. Move into that new directory.

```
cd newProjectDir
```

3. Initialize Git directory.

```
git init
```

4. Create/edit project files.

5. Track files in the project directory.

```
git add .
```

6. Commit tracked files in Git storage.

```
git commit -m "1st commit"
```

Beginning to Use Git

1. Pull down the latest version of the code to local.
2. Add edited files to the source code.
3. Commit edits to the local version.
4. Add a comment about this commit.
5. Push the changes out to the server.

Beginning to Use Git

1. Pull down the latest version of the code to local.

```
git pull
```

2. Add edited files to the source code.

```
git add File.java
```

3. Commit edits to the local version. `git commit -m "comments here"`

4. Add a comment about this commit.

```
git push
```

5. Push the changes out to the server.

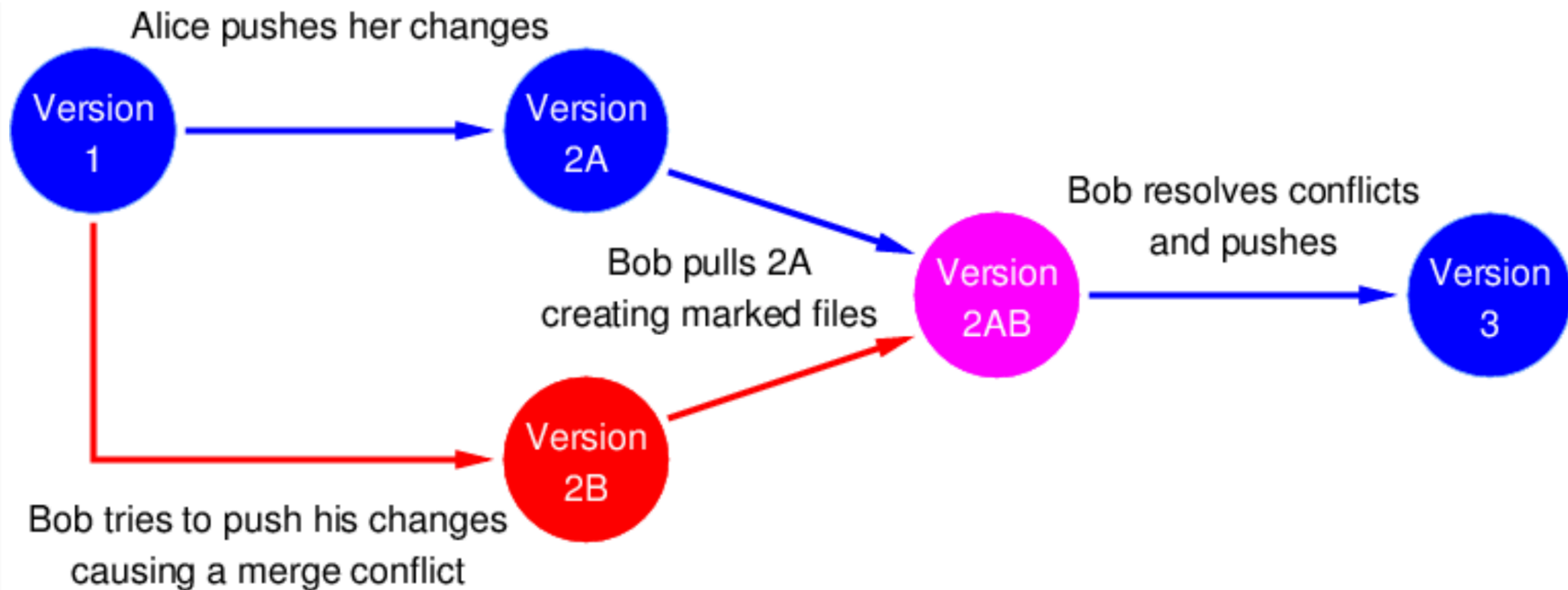
Merge Conflicts



Merging

- **Simultaneous editing** of the same project means that developers may be working within the same class/file, then pushing their changes to the repository..
- Collaborative software development doesn't always run smoothly!
 - **Merge conflicts** occur when changes pushed conflict with the code currently in the repository.

Merge Conflicts



Merging

1. Pull the latest version
2. Edit the files with markers
3. Add (*stage*) the files
4. Commit & push*

Merging

- Step 1: **Pull the latest version**
 - Right click on the project > Team > Pull

```
public class Hello {  
    public static void main(String[] args) {  
local  <<<<<<<< HEAD  
        System.out.println("Hello Universe!!!");  
=====  
remote System.out.println("Hello World!!!");  
>>>>>>>> branch 'master' of https://github.com/UTSA-CS-3443/test-repo.git  
    }  
}
```

Merging

- Step 2: **Edit files with markers**
- Change the code *and remove the markers!*

```
public class Hello {  
    public static void main(String[] args) {  
        System.out.println("Hello World and the Universe!!!");  
    }  
}
```

Merging

- Step 3: **“Add” the files**
 - Right click project > Team > Add to Index
- Step 4: **Push**
 - Right click project > Team > Commit
 - *Depending on the elapsed time since step 1, you may need to pull again!*

You can choose how you want to use git!

- Command line
- Browser <http://github.com>
- GitHub Desktop <http://desktop.github.com>
- EGit (a plugin for Eclipse) <http://www.eclipse.org/egit/>

GitHub

- Web-based Git repository hosting service, with >35 million repos to date.
- Provides cloud storage, public/private repos, free accounts.
 - Home to lots of open source software!



Git Terminology

- Repo - repository
- Init - initialize a repository
- Push/Pull
- Commit
- Clone - copy a remote repo locally
- Branch

Git Resources

For this course:

- [Hints for using eGit \(Eclipse plugin\)](#)
- [Resolving merge conflicts](#)

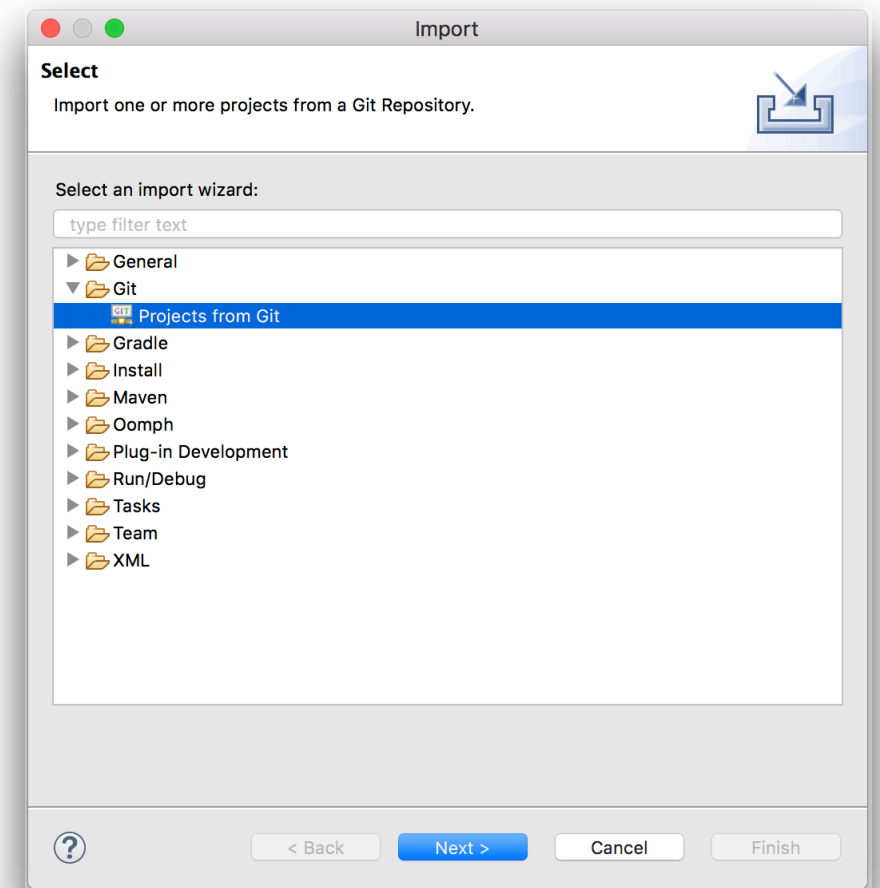
Other resources

- Effective Git: <https://git-scm.com/book/en/v2>
- Tutorial: <https://try.github.io/>

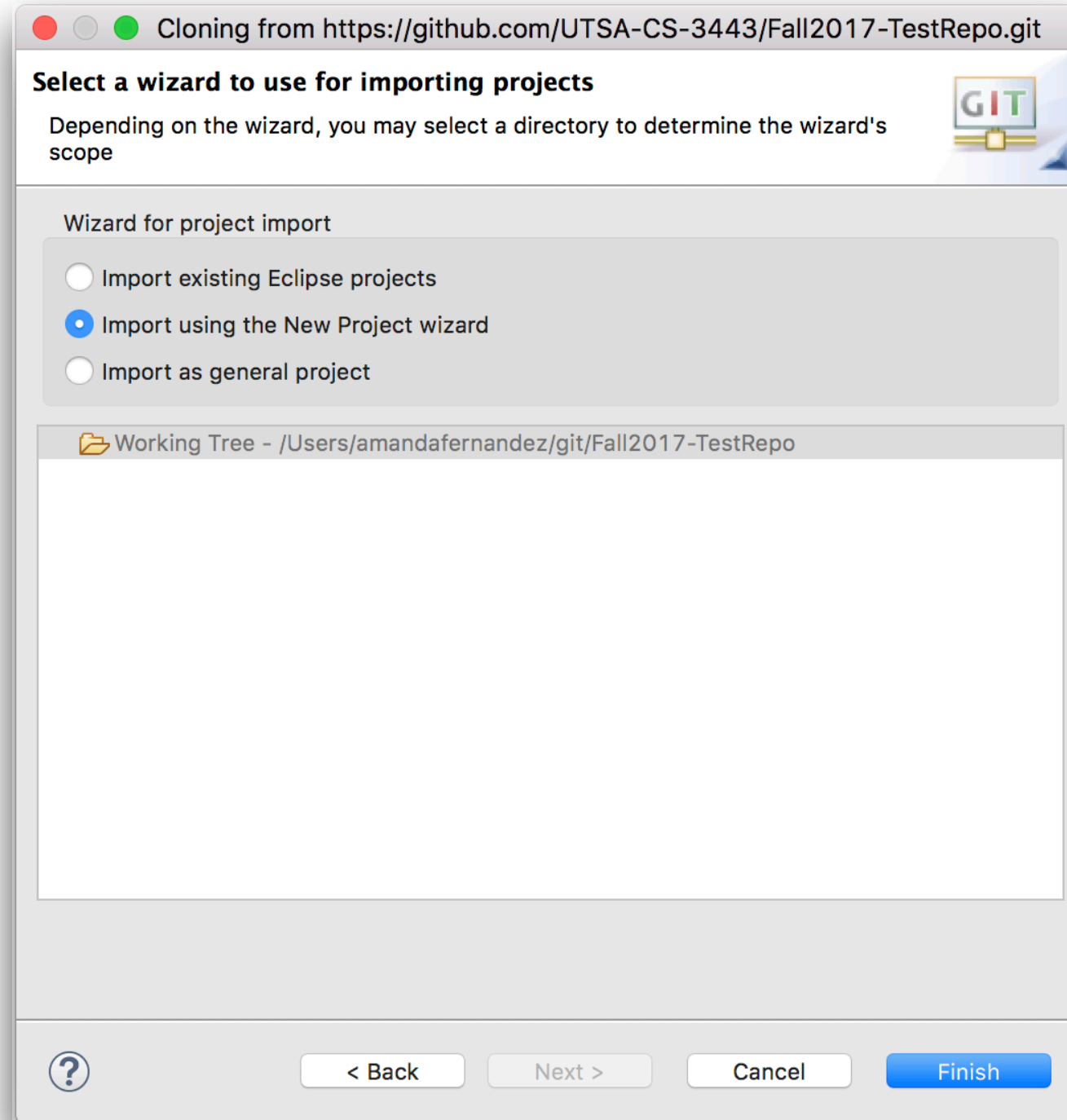
Highly recommend!

EGit - Cloning the repo

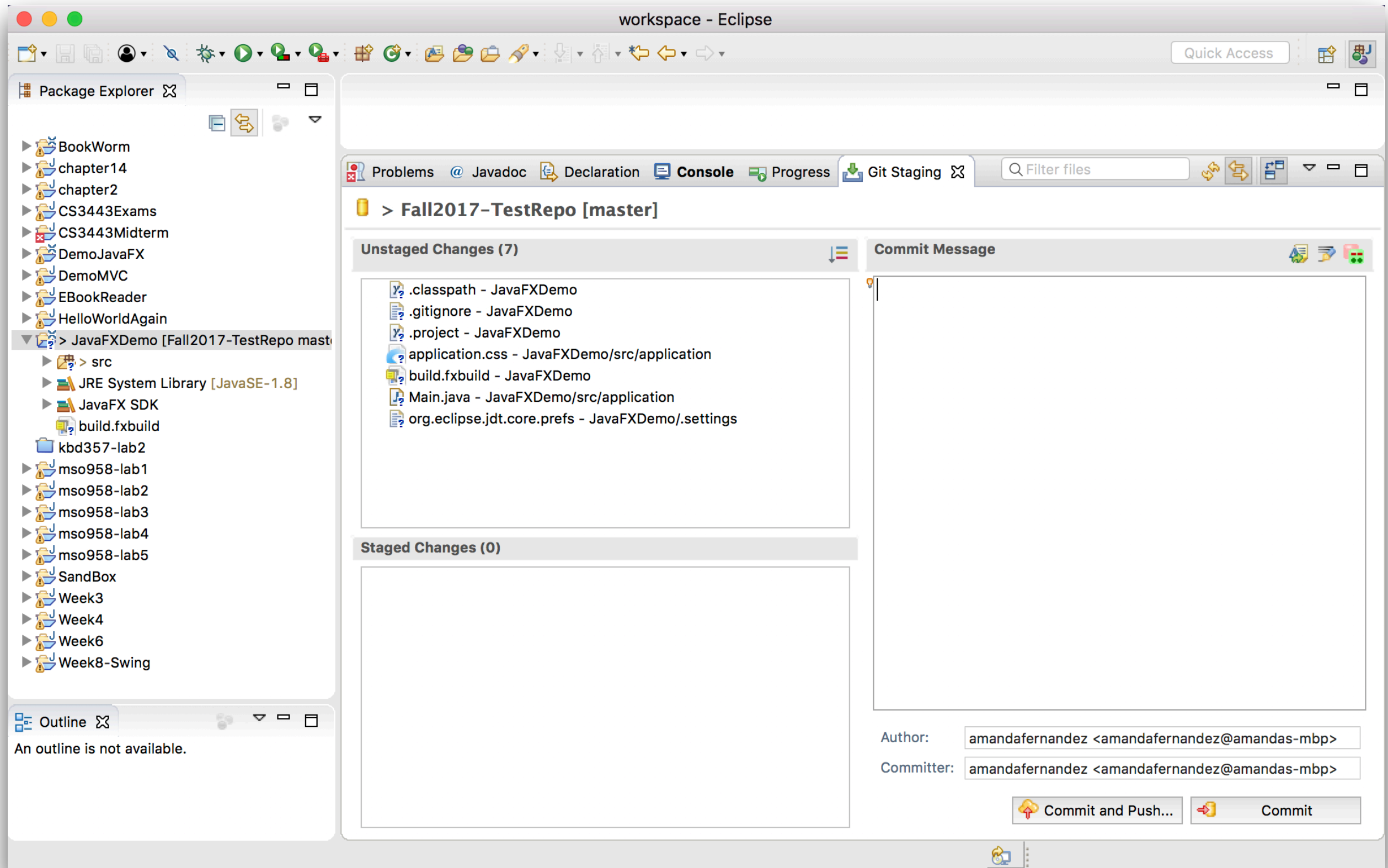
- In Eclipse:
- Select File -> Import -> Git -> Projects from Git
- Click Next
- Select CloneURI, then Click Next
- In this window:
 - URI = <https://github.com/UTSA-CS-3443/reponame.git>
 - User = [your GitHub username]
 - Password = [your GitHub password]
 - Also check "Store in Secure Store".
 - You will be asked to create and confirm a Secure Store password later.
 - For convenience, use your GitHub password as the Secure Store password (we're not aiming for high security here).
 - Keep Clicking Next.



Importing a project from Git



Committing to Git repo



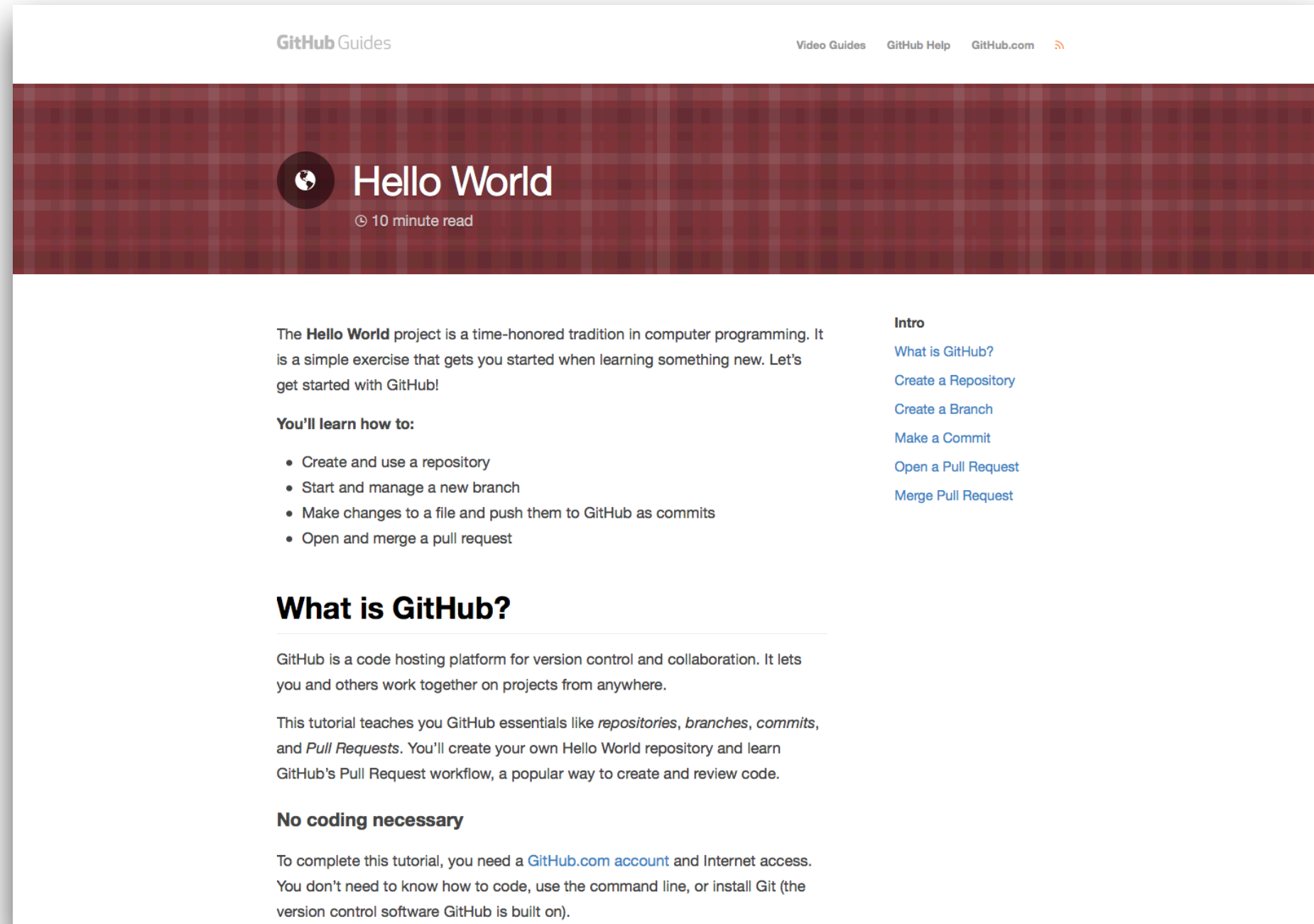
Hello, GitHub!

<https://guides.github.com/activities/hello-world/>

1. Go to [GitHub.com](https://github.com), click “read the guide” button.
2. Follow the instructions for GitHub’s Hello World **tutorial**.

Create the repos on your account, not for your team or a lab.

REMINDER:
Never post any labs or assignments for any course to any online repo!



The screenshot shows the GitHub Guides page for the 'Hello World' tutorial. The page has a dark red header with a grid pattern. Below the header, there's a section titled 'Hello World' with a globe icon and a '10 minute read' indicator. The main content area is white and contains several sections: 'The Hello World project is a time-honored tradition in computer programming. It is a simple exercise that gets you started when learning something new. Let's get started with GitHub!', 'You'll learn how to:' followed by a bulleted list of tasks, 'What is GitHub?' with a brief description, and 'No coding necessary' with instructions on how to complete the tutorial. On the right side, there's a sidebar with an 'Intro' section containing links to 'What is GitHub?', 'Create a Repository', 'Create a Branch', 'Make a Commit', 'Open a Pull Request', and 'Merge Pull Request'.

GitHub Guides

Video Guides GitHub Help GitHub.com

Hello World

🕒 10 minute read

The **Hello World** project is a time-honored tradition in computer programming. It is a simple exercise that gets you started when learning something new. Let's get started with GitHub!

You'll learn how to:

- Create and use a repository
- Start and manage a new branch
- Make changes to a file and push them to GitHub as commits
- Open and merge a pull request

What is GitHub?

GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.

This tutorial teaches you GitHub essentials like *repositories*, *branches*, *commits*, and *Pull Requests*. You'll create your own Hello World repository and learn GitHub's Pull Request workflow, a popular way to create and review code.

No coding necessary

To complete this tutorial, you need a [GitHub.com account](#) and Internet access. You don't need to know how to code, use the command line, or install Git (the version control software GitHub is built on).

Intro

- [What is GitHub?](#)
- [Create a Repository](#)
- [Create a Branch](#)
- [Make a Commit](#)
- [Open a Pull Request](#)
- [Merge Pull Request](#)

JavaFX Tips & Tricks

EGIT + GITHUB

Eclipse & GitHub

- As of August 2021, Eclipse will require a **personal access token** for GitHub authorization (rather than your GitHub password).

Create a Personal Access Token

1. Log into GitHub, click on your account icon in the top right
2. Click Settings
3. Click Personal access tokens
4. Click the button for Generate new token
5. Copy the token to clipboard and paste it as your **password** in Eclipse

The screenshot shows the GitHub web interface. At the top, the navigation bar includes the GitHub logo, a search bar, and links for Pull requests, Issues, Marketplace, and Explore. The user is signed in as 'amandanko'. The main content area is titled 'Settings / Developer settings' and 'Personal access tokens'. On the left, a sidebar menu has three items: 'GitHub Apps', 'OAuth Apps', and 'Personal access tokens', with the third item highlighted and labeled with a blue '3'. The main content area has a heading 'Personal access tokens' and a button 'Generate new token' labeled with a blue '4'. Below the heading, there is explanatory text about API tokens. On the right, a dropdown menu is open, showing options like 'Set status', 'Your profile', 'Your repositories', 'Your codespaces', 'Your organizations', 'Your projects', 'Your stars', 'Your gists', 'Upgrade', 'Feature preview', 'Help', 'Settings', and 'Sign out'. The 'Settings' option is highlighted with a blue '2'.

1

Signed in as amandanko

Set status

Your profile

Your repositories

Your codespaces

Your organizations

Your projects

Your stars

Your gists

Upgrade

Feature preview

Help

Settings

Sign out

2

3

4

Settings / Developer settings

GitHub Apps

OAuth Apps

Personal access tokens

Personal access tokens

Generate new token

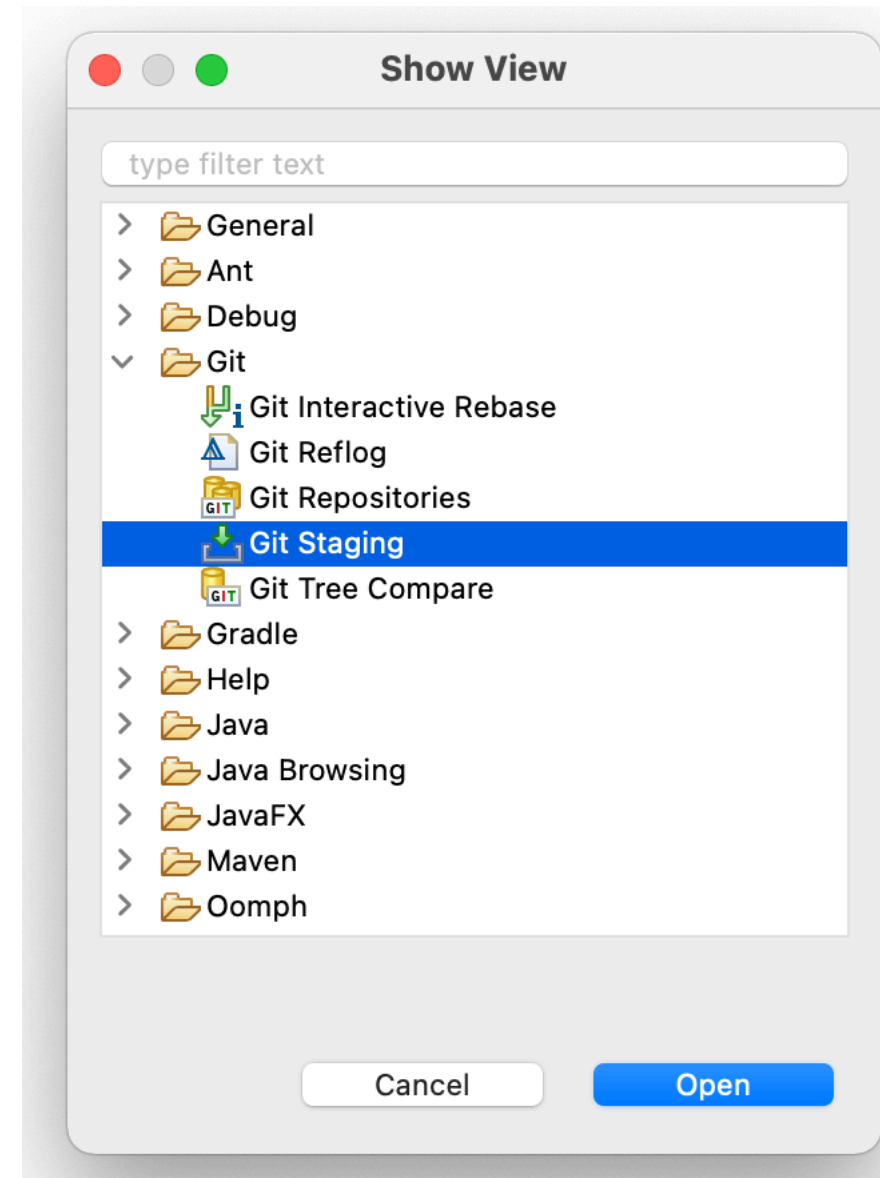
Need an API token for scripts or testing? [Generate a personal access token](#) for quick access to the [GitHub API](#).

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

© 2021 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)

EGIT

- To view git integration in Eclipse, we'll **show the view** for git.
- In the top menu, choose Window > Show View > Other, then select “Git Staging” from the list of Git views (example pictured at right)



Eclipse Git Staging View

