**Hello Git**

Git is a software that allows you to keep track of changes made to a project over time. Git works by recording the changes you make to a project, storing those changes, then allowing you to reference them as needed.

We'll learn Git by using it to help us write a screenplay called *Harry Programmer and the Sorcerer's Code*.

**git init**

Now that we have started working on the screenplay, let’s turn the **sorcerers-code**directory into a Git project. We do this with:

git init

The word init means *initialize*. The command sets up all the tools Git needs to begin tracking changes made to the project.

# Git Workflow

Nice! We have a Git project. A Git project can be thought of as having three parts:

1. A Working Directory: where you'll be doing all the work: creating, editing, deleting and organizing files
2. A Staging Area: where you'll list changes you make to the working directory
3. A Repository: where Git permanently stores those changes as different versions of the project

The Git workflow consists of editing files in the working directory, adding files to the staging area, and saving changes to a Git repository. In Git, we save changes with a commit, which we will learn more about in this lesson.

# git status

As you write the screenplay, you will be changing the contents of the working directory. You can check the status of those changes with:

git status

# git add

In order for Git to start tracking **scene-1.txt**, the file needs to be added to the staging area.

We can add a file to the staging area with:

git add filename

The word filename here refers to the name of the file you are editing, such as **scene-1.txt**.

**git commit**

A *commit* is the last step in our Git workflow. A commit permanently stores changes from the staging area inside the repository.

git commit is the command we'll do next. However, one more bit of code is needed for a commit: the *option* -m followed by a message. Here's an example:

git commit -m "Complete first line of dialogue"

Standard Conventions for Commit Messages:

* Must be in quotation marks
* Written in the present tense
* Should be brief (50 characters or less) when using –m

# git log

Often with Git, you'll need to refer back to an earlier version of a project. Commits are stored chronologically in the repository and can be viewed with:

git log

**Generalizations**

You have now been introduced to the fundamental Git workflow. You learned a lot! Let's take a moment to generalize:

* Git is the industry-standard version control system for web developers
* Use Git commands to help keep track of changes made to a project:
  + git init creates a new Git repository
  + git status inspects the contents of the working directory and staging area
  + git add adds files from the working directory to the staging area
  + git diff shows the difference between the working directory and the staging area
  + git commit permanently stores file changes from the staging area in the repository
  + git log shows a list of all previous commits