**Beginner’s Guide to Git**

**Digital Scholarship Group, Northeastern University**

**Table of Contents**

* About this Documentation
* About Git
* Downloading and Installing Git
* Setting Up a New Repository
* Copying and Existing Repository
* Configuring Git
* Adding, Modifying, Staging, and Committing Files

**About this Documentation**

* Commands to be used in Command Line (PC) or Terminal (Mac) are shown in **bold**.
* Text in **red** should be replaced with your own information.

**About Git**

Git is a distributed version control system (DVCS) that utilizes command line tools and allows multiple users to create, modify, and store coded data and other files. What makes Git different from other version control systems is its *distributed* structure. This means that all files stored in a project repository are saved locally on all platforms associated with a project. This allows for greater protection against catastrophic loss of project data, because all the information is copied onto ever machine associated with the project, significantly increasing the likelihood that at least one copy of the entire repository will exist.

All authorized users in a Git repository can create files on their own machine and upload them to the shared repository. Every time a file is added to a repository Git takes a snapshot of all files at that moment and stores the history. Unchanged files are represented as links to the file as it was in its most recently changed version. This history is stored on every local platform, allowing for quicker review by individual members of a project. Also, because Git stores all files locally project members can access all project files even when they are offline.

Git has three different file states: committed files, modified files, and staged files. A fourth state, untracked, refers to files that are not currently in your Git workspace but can be staged and later committed using the appropriate commands.

**Downloading and Installing Git**

To download Git, please visit <http://git-scm.com/downloads>. Downloads are available for PC, Mac, Linux, and Solaris.

For information on installing Git on your computer, please visit <http://git-scm.com/book/en/Getting-Started-Installing-Git>.

**Setting Up a New Repository**

1. Create the directory in which you want your Git repository to reside. This can be done one of two ways:
   1. Creating a folder in Finder/My Computer
   2. Using the command line to create a new directory with the **mkdir** command.
2. Navigate to the new directory you just created using the **cd** command.
3. Initialize your Git repository using the **git init** command.
   1. NOTE: If you want to begin working with an existing repository please see the section on *Copying an Existing Repository*.

**Configuring Git**

1. To set your username for the repository you are currently working in, use the command **git config user.name “Your Name”**. To set your username for all repositories on your machine, use the command **git config --global “Your Name”**.
2. To set your email for the repository you are currently working in, use the command **git config user.email your.email@neu.edu**. To set your email for all repositories on your machine, use the command **git config user.email --global** [**your.email@neu.edu**](mailto:your.email@neu.edu).
3. To check your settings, use the command **git config --list** to display the following information:
   1. user.name
   2. user.email
   3. color.status
   4. color.branch
   5. color.interactive
   6. color.diff

**Copying an Existing Repository**

1. Create the directory in which you want the existing Git repository to reside on your machine. This can be done one of two ways:
   1. Creating a folder in Finder/My Computer
   2. Using the command line to create a new directory with the **mkdir** command.
2. Navigate to the new directory you just created using the **cd** command.
3. Find the URL for the repository you want to copy.
4. Use the command **git clone http://github.com/sample**, replacing the text in red with your desired repository URL.
   1. NOTE: The name of the cloned repository will be pulled from the URL. If you wish to change the name of this repository for your personal machine, use the command **git clone http://github.com/sample new\_repository\_name**.

**Adding, Modifying, Staging, and Committing Files**

1. Check the status of your repository using the **git status** command. This should display all the files (both tracked and untracked) that exist in the directory that houses your repository. If it does not, use the **cd** command to navigate to the appropriate directory.
2. To begin tracking an untracked file, use the **git add file\_name.txt** command. If you run the **git status** command again, it should display the file as tracked now, under the heading “Changes to be committed:”.
   1. To untrack a files use the command **git reset** **file\_name.txt**. This command will not work on files that have already been committed. The command **git rm file\_name.txt** can also be used to remove files from the staging area.
3. When you are ready to commit all the files that you have staged using **git add**, use the command **git commit file\_name.txt –m “Short description of commit”**.
4. After committing files, you can use the **git log** command to check the official changes you have made to the repository. There are several options for changing the output of the **git log** command:
   1. **git log –p** shows you the specific changes that have been made to any files that were modified in a commit
   2. **git log –n**, where **n** is replaced with a number, limits the number of commits that will be shown, from most recent to least recent.
   3. **git log --since**