Updated 12/6/18

# GitHub Guide

## **Git Installation** (Local, first time setup)

1) Download and install 64-bit Git for Windows from: https://git-scm.com/download/win/
    a. Accept all defaults when installing Git.
2) Create a folder where all of your R project repositories will be stored. 'GitHub' is a common name to use.

## **Initial Setup** (link local Git to online GitHub account – you will only need to do this once)

1) Right click inside the GitHub folder and select Git Bash Here to open a bash terminal window.
2) In Git command line, type: `ssh-keygen`
3) When prompted for a passphrase, simply press Enter. Press Enter again when asked again.
4) Navigate to where the public key is stored using the git bash command line
    a. `cat /c/Users/[MachineUserName]/.ssh/id_rsa.pub`
5) Copy the key to your clipboard
6) Navigate in a web browser window to your personal GitHub account and open Settings in the upper right
    a. Click on "SSH and GPG keys" in the left menu
    b. In the top right corner click "New SSH key"
    c. Name the key something logical (without spaces) in the Title field e.g. "ssh1"
    d. Paste the key from your clipboard into the Key field. Do not make any changes or add spaces. Make sure no trailing space is present in pasted text.
    e. Click "Add SSH key"
7) In Git command line, establish the connection to GitHub: `ssh -T git@github.com`
    a. Type yes if prompted
        i. Note: A window may pop up requiring GitHub username and password to authenticate
    b. Go back to GitHub and refresh the screen- notice the key usually turns green notifying you it has been properly authenticated

---

## **Clone a Repository** (from remote GitHub to local machine)

* If you already have projects started on your GitHub account in individual/organization repositories OR you are starting to work with someone else's repository, you need to clone the repository to your local machine

1) Open a Git command line window from your local GitHub folder: Right Click in folder -> "Git Bash Here"
2) In GitHub, navigate to repository of interest, click "Clone or download" and copy the link to your clipboard.
3) Clone the desired repository in Git by pasting the link from your clipboard after "git clone" to resemble:
    a. `git clone https://github.com/[GitHubAccountName]/[RepositoryName].git`
    b. If SSL certificate problem appears, add `-c http.sslVerify=false` between git and clone commands, e.g.

```
git -c http.sslVerify=false clone https://github.com/EmmaVJones/PermitApp.git
```

Updated 12/6/18

GitHub Guide

## Creating Local Branches

* Users create branches off the master any time they wish to make file modifications.
* Branches allow users to edit versions of a project without affecting the live, production version of the project (master branch)

1) Open Git command line window from your local repository (inside GitHub directory):
    a. Right Click in folder -> Git Bash Here
2) Create a local branch, and switch to that branch
    a. `git branch [BranchName]` *Be sure not to have any spaces in branch names
    b. `git checkout [BranchName]` *Notice the name of the branch you're located in is displayed in blue parentheses within the git bash terminal `(BranchName)`
3) Begin adding/removing/or modifying existing files in the branch

## Commit Local Changes

* Users commit changes when they alter the working copy of a script OR add/remove files from a repository. Commit messages are helpful text strings that are saved with commits to help track changes in workflows.

1) After adding/removing/modifying files in the new branch, add all files in directory to local index (think of this as a staging area):
    a. `git add -A`
    b. Alternatively, you can add single files by name: `git add fileName.R`
2) Commit all local file changes for everything just added to the index (complete with commit message briefly describing the contents/purpose of the commit):
    a. `git commit -a -m 'this is a commit message'`
    b. Git will report how many files changed

## Push Local Commits to Remote Repository (GitHub)

* After you are comfortable with your commit(s) and want to send code to GitHub for backup/sharing/merging with master branch, you need to push your commits to the corresponding GitHub repository.

1) From the previously opened Git command line window from your local repository (inside GitHub directory), push commit(s) to GitHub:
    a. `git push origin [BranchName]`
    b. `[BranchName]` indicates the branch you wish to push commits from, this can be any active branches in the local repository
    c. If you receive SSL certificate errors, insert `-c http.sslVerify=false` between git and push e.g. `git -c http.sslVerify=false push origin [BranchName]`
    d. You will receive a notification that everything is up to date if the push was successful
        i. Note: A window may pop up during your first push requiring GitHub username and password to authenticate
2) You can go into repository on GitHub to verify new branch by using the branch drop down menu

**Merge changes with Master branch** (From GitHub in web browser)

*After a user is finished pushing a local branch to the remote repository, it is best practice to merge the branch to the master branch and delete the working branch.

1) Navigate to the working branch on GitHub
    a. When you are ready to merge the working branch with the master branch, click “New pull request” button. Supply a title (The last commit message will populate the title by default). You can add additional comments describing the contents of the pull request in the comment field.
    b. Click “Create pull request” after reviewing any conflicts the new branch has with the master branch
    c. Click “Merge pull request”
    d. Click “Confirm merge”
    e. Click “Delete branch”
        i. You can verify the branch is deleted on GitHub using the branch drop down menu
2) Navigate back to the master branch in Git: `git checkout master`
    a. Delete the branch you just merged from your local Git: `git branch -D [BranchName]`
        i. e.g. `git branch -D newFeature`
3) Pull the master branch to ensure your local machine is up to date with all merges on GitHub
    a. `git pull origin master`
    b. If you receive SSL certificate errors, insert `-c http.sslVerify=false` between git and pull e.g. `git -c http.sslVerify=false pull origin master`

**General Work Flow**

1) Pull the master branch
2) Create a working branch
3) Add/remove/or modify files
4) Commit and push all working branch updates to GitHub
5) Perform a pull request and merge working branch changes with master
6) Delete working branch on GitHub
7) Delete working branch from local machine
8) Pull the master branch to your local machin