

# GIT - Notes

## GIT - Notes

**Alan T. Arnholt**

Last compiled:

[1] "Monday, January 13, 2014 - 13:00:44."

Download and install the latest version of [Git](#).

If you have never used git before, you need to do some setup first. Run the following commands so that git knows your name and email. The commands are all issued in the Terminal (MAC) or at the command prompt (Windows). The Terminal application is usually found in `/applications/Utilities`. To open a command prompt, click on the Windows icon -> All Programs -> Accessories -> Command Prompt. The third line adds pretty command line colors.

```
git config --global user.name "Your Name"
git config --global user.email "your_email@whatever.com"
git config --global color.ui true
```

If you do not want to type your username and password every time you work with a remote server, you will to install the credential helper.

## Creating a GitHub Account

Point your browser to <https://github.com>, type a username in the **Pick a username** box (please use firstlast, for example my username is `alanarnholt`), enter your email (use your school email) in the **Your email** box, type in your password in the **Create a password** box. Then, click the **Sign up for GitHub** box and you will have a GitHub account.

## Creating a GitHub Repository

In order to push your local work to a remote repository, you will first need to create the remote repository. Log into your GitHub account, click the **New repository** button, then give your repository a name and optionally a description. When you finish, click the **Create repository** button and your GitHub repository will be created.

The screenshot shows the GitHub 'Create repository' interface. At the top, there's a navigation bar with the GitHub logo, a search bar, and links for Explore, Gist, Blog, and Help. The user 'alanarnholt' is logged in. The form has two main sections: 'Owner' and 'Repository name'. The 'Owner' is set to 'alanarnholt' and the 'Repository name' is 'GiveItSomeNameHere' with a green checkmark. Below this, a tip suggests repository names should be short and memorable, with 'yolo-ironman' as an example. The 'Description (optional)' field contains 'GiveSomeDescriptionHere'. There are two radio buttons for visibility: 'Public' (selected) and 'Private'. The 'Public' option is described as 'Anyone can see this repository. You choose who can commit.' The 'Private' option is 'You choose who can see and commit to this repository.' There is an unchecked checkbox for 'Initialize this repository with a README', with a note that it will allow cloning immediately. Below this are two dropdown menus: 'Add .gitignore: None' and 'Add a license: None'. A green 'Create repository' button is at the bottom. The footer contains copyright information for 2014 GitHub, Inc., and links for Terms, Privacy, Security, Contact, Status, API, Training, Shop, Blog, and About.

Owner: **alanarnholt** / Repository name: **GiveItSomeNameHere** ✓

Great repository names are short and memorable. Need inspiration? How about **yolo-ironman**.

Description (optional): GiveSomeDescriptionHere

☒ **Public**  
Anyone can see this repository. You choose who can commit.

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

☐ **Initialize this repository with a README**  
This will allow you to `git clone` the repository immediately. Skip this step if you have already run `git init` locally.

Add .gitignore: **None** | Add a license: **None** ⓘ

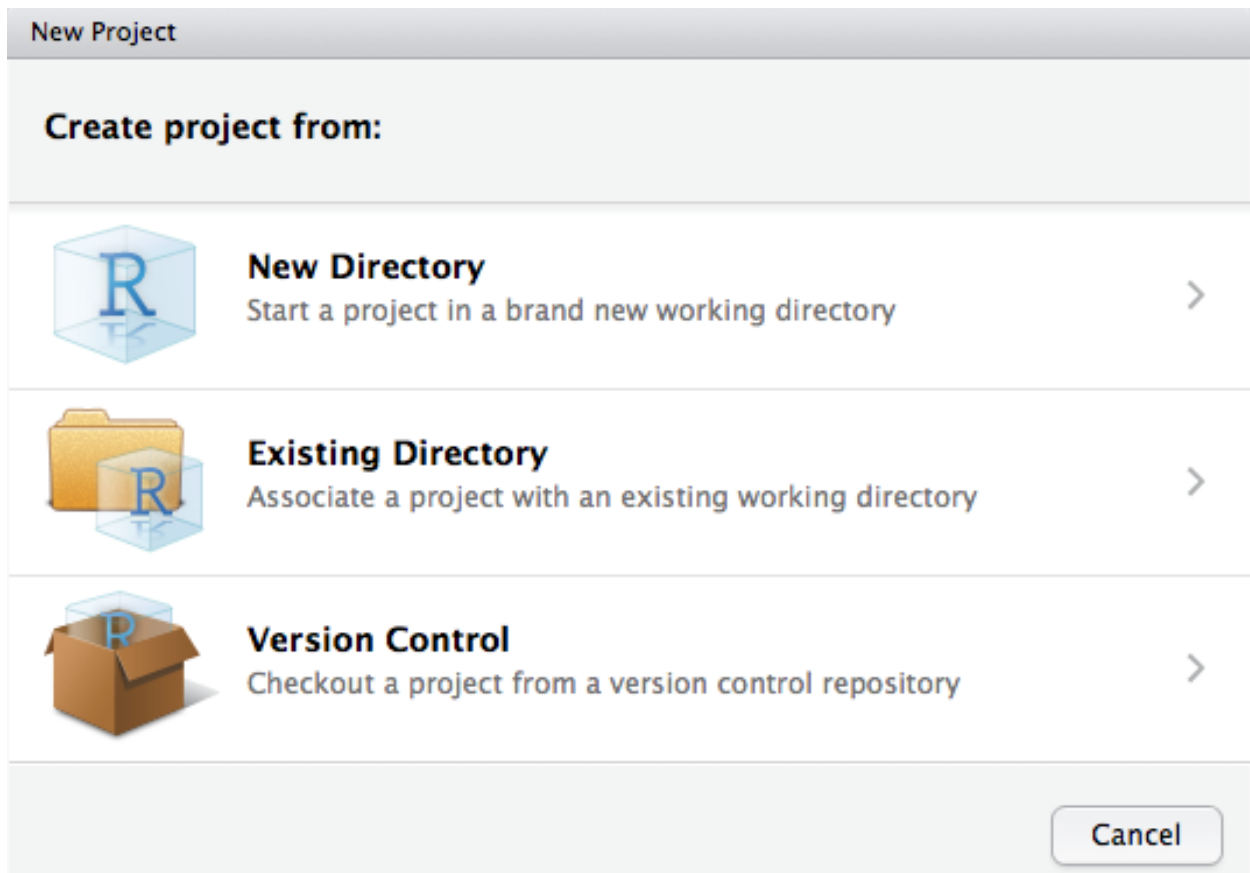
**Create repository**

© 2014 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Contact](#) [Status](#) [API](#) [Training](#) [Shop](#) [Blog](#) [About](#)

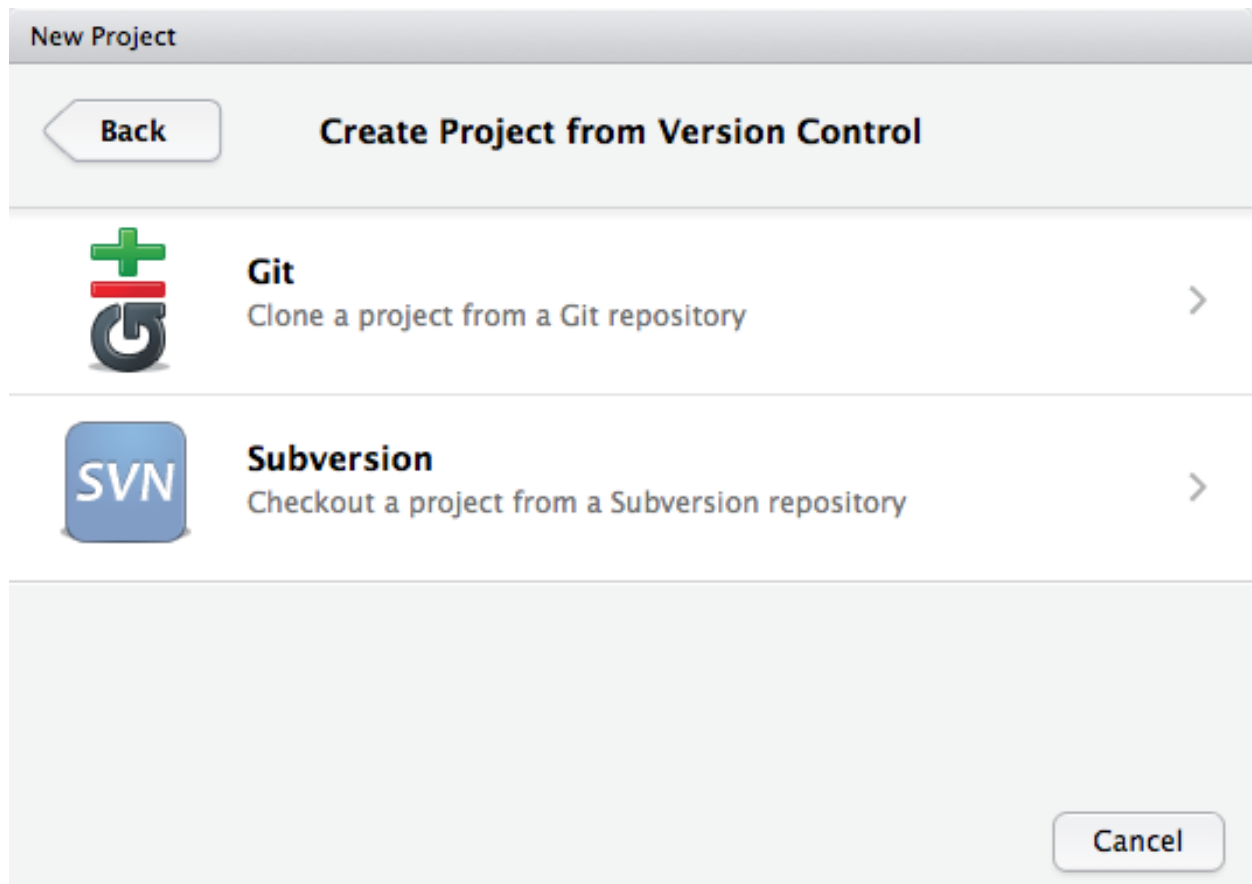
This document is stored in the repository <https://github.com/alanarnholt/SEMINAR> in the folder <https://github.com/alanarnholt/SEMINAR/tree/master/Alan/summaries/GITstuff>.

## Local Repositories

It is possible to set up a local repository using GUI (drop, drag, etc.) commands or to use the command line. I keep my repositories in a folder called *git\_repositories* that is a subfolder of my *USERNAME* directory. Once you have a local folder with files you would like to place under version control, use the `git init` command from your working directory to track your files. If you clone a remote repository to your machine, you will not need to initialize your directory. One way to clone this repo using **RStudio** is to click on File -> New Project



Click Version Control and a new window such as the one below will appear where you will select Git.




In the next window that will appear, shown below you will need to enter the URL for the repository you are cloning. Enter a project name and specify where you want the project to reside on your computer. When you are finished, click the Create Project button and you will have cloned a remote repository.

New Project

Back

Clone Git Repository



Repository URL:

Project directory name:

Create project as subdirectory of:

Browse...

☐ Open in new window

Create Project

Cancel

To check the current status of your repository type:

```
git status
```

```
# On branch master
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in working directory)
#
#   modified:   GIT_LAB1.html
#   modified:   GIT_LAB1.md
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#   ../../../../.DS_Store
no changes added to commit (use "git add" and/or "git commit -a")
```

The `git status` shows us what files are not staged for a commit. Before files can be committed, they must be added to the staging area. Files are added to the staging area with the command `git add file_name`. To add all files in the working directory, one can use `git add .` Next all files are added to the staging area and a snapshot is taken of the commit with the message "staging all files".

```
git add .
git commit -m "staging all files"
```

```
[master f48f765] staging all files
2 files changed, 70 insertions(+), 60 deletions(-)
```

Check the status after the last commit.

```
git status
```

```
# On branch master
# Your branch is ahead of 'origin/master' by 1 commit.
#   (use "git push" to publish your local commits)
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#   ../../../../.DS_Store
nothing added to commit but untracked files present (use "git add" to track)
```

Push changes to the remote repository.

```
git push
```

See if there is anything left to do.

```
git status
```

```
# On branch master
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#   ../../../../.DS_Store
nothing added to commit but untracked files present (use "git add" to track)
```

Show the last three commits with

```
git log -3
```

```
commit f48f765e7a63f7b0305d338f49fbaf017b7c70f9
Author: Alan Arnholt <arnholtat@appstate.edu>
Date:   Mon Jan 13 13:00:44 2014 -0500
```

```
    staging all files
```

```
commit 757bb0448794e42f8b99f024fe20f473f70b39b3
Author: Alan Arnholt <arnholtat@appstate.edu>
Date:   Mon Jan 13 10:20:21 2014 -0500
```

```
    add more directions
```

```
commit 16ac4ac8e65ccae2f73daf5f4878b10facee129c
Author: Alan Arnholt <arnholtat@appstate.edu>
Date:   Mon Jan 13 10:18:45 2014 -0500
```

```
    staging all files
```

That was ugly. Let us try some formatting.

```
git log --pretty=oneline -3
```

```
f48f765e7a63f7b0305d338f49fbaf017b7c70f9 staging all files
757bb0448794e42f8b99f024fe20f473f70b39b3 add more directions
16ac4ac8e65ccae2f73daf5f4878b10facee129c staging all files
```

The previous output was to brief for my likes. Let us try some further formatting.

```
git log --pretty=format:"%h %ad- %s [%an]" -3
```

```
f48f765 Mon Jan 13 13:00:44 2014 -0500- staging all files [Alan Arnholt]
757bb04 Mon Jan 13 10:20:21 2014 -0500- add more directions [Alan Arnholt]
16ac4ac Mon Jan 13 10:18:45 2014 -0500- staging all files [Alan Arnholt]
```

Maybe even some statistics?

```
git log --pretty=format:"%h %ad- %s [%an]" -3 --stat
```

```
f48f765 Mon Jan 13 13:00:44 2014 -0500- staging all files [Alan Arnholt]
  Alan/summaries/GITstuff/GIT_LAB1.html | 69 ++++++-----
  Alan/summaries/GITstuff/GIT_LAB1.md   | 61 ++++++-----
  2 files changed, 70 insertions(+), 60 deletions(-)

757bb04 Mon Jan 13 10:20:21 2014 -0500- add more directions [Alan Arnholt]
  Alan/summaries/GITstuff/GIT_LAB1.html | 48 ++++++-----
  Alan/summaries/GITstuff/GIT_LAB1.md   | 42 ++++++-----
  2 files changed, 39 insertions(+), 51 deletions(-)

16ac4ac Mon Jan 13 10:18:45 2014 -0500- staging all files [Alan Arnholt]
  Alan/summaries/GITstuff/GIT_LAB1.Rmd  |  2 +-
  Alan/summaries/GITstuff/GIT_LAB1.html | 53 ++++++-----
  Alan/summaries/GITstuff/GIT_LAB1.md   | 47 ++++++-----
  3 files changed, 60 insertions(+), 42 deletions(-)
```

Now, just to show how cool this is, we will mix in a little R.

```
library(ggplot2)
ggplot(data = CO2, aes(x = Type, y = uptake)) + geom_boxplot()
```

