Version Control

What is Version Control?

Version control (also sometimes known as source control) is a system that manages changes to a program, website, or other collection of documents.

Basically, it records the changes made to files so if you need to revert back to a previous version of a file you've been working on, you can then do that.

Why Care About Version Control?

- Say that you're writing some code and you've tried something new that ended up not working. Using version control, you can go back to a previous version of your code that was working instead of spending hours figuring out where you started from.
- Comes in handy when you are working in groups and have multiple people working on the files. That way, everyone can be working on the latest version of the project.
- TL;DR – Using version control = less crying in the long run

Two Types of Version Control

Centralized

-Centralized version control systems are run on one central server, and each collaborator checks out the code from and merges changes into the main server

Distributed

-Distributed version control systems allow each collaborator to maintain a separate repository of the code, which can be periodically reconciled with the main server

Version Control Tools – Intro to Git

There are many different version control tools. Some are open source and others are proprietary. One of the more popular ones in the Detroit area is Git. Git is an open source and distributed version control tool. Using this isn't required for your final project, but you can use it if you want.

Download Git for Windows or Mac here.

Git Tutorial

Git Terms

- ❑Repository (repo) – a self-contained codebase or project
- ❑Local – on your computer
- ❑Remote – on the internet
- ❑Snapshot – the way Git stores changes in a project (Git knows all and sees all – it can spot the differences)

Git Basics

❑ Create a repo – git init

❑ Check status – git status

❑ Make some changes

❑ Staging changes – git add

❑ Staging vs. committing

❑ Committing – git commit -m

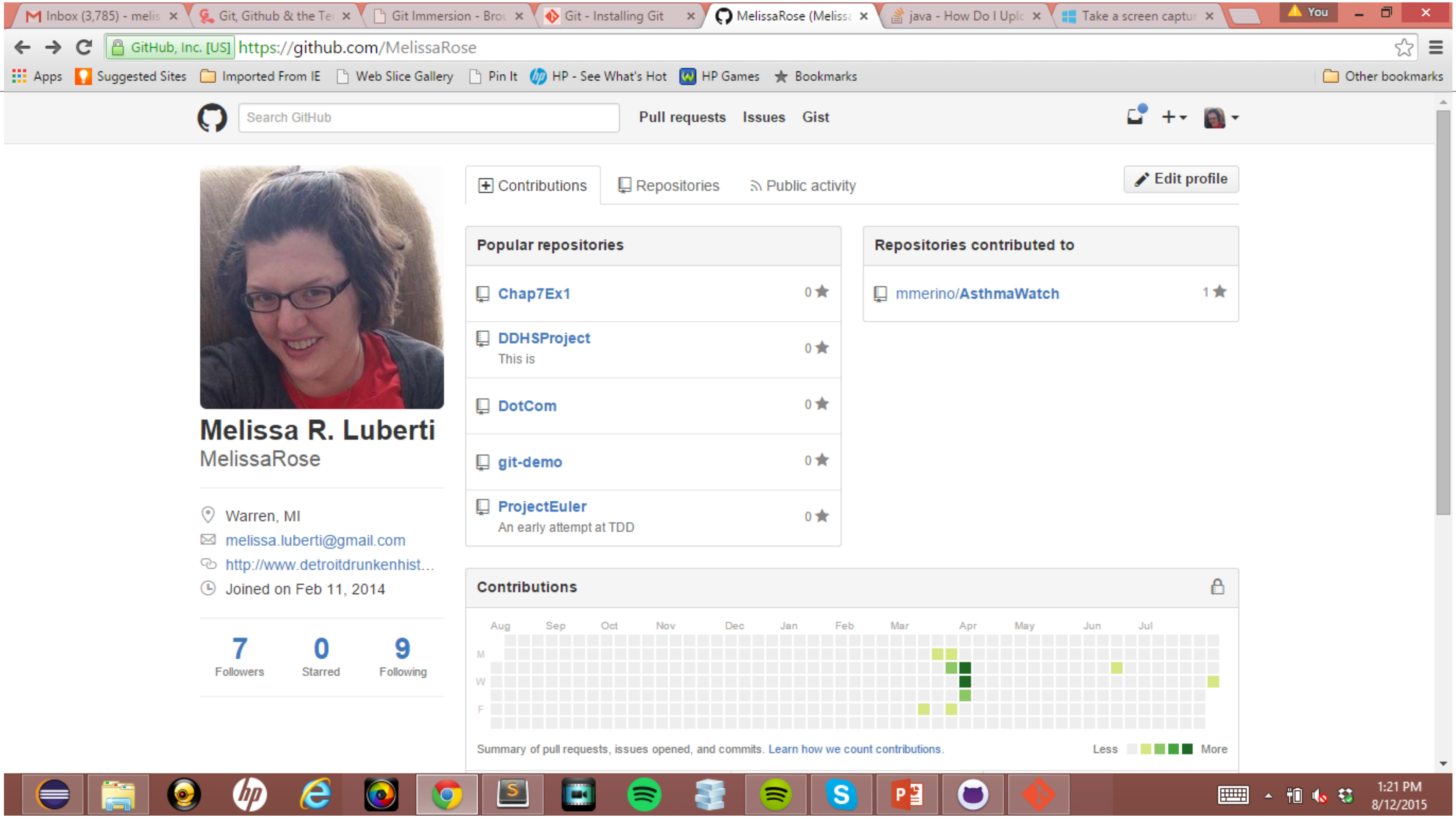
Into to GitHub (Wait, Didn’t We Just Go Over That?)

Please note that GitHub is not the same thing as Git. GitHub is a social coding website for managing projects that uses Git as its version control. Think of GitHub as social media for code. People can share their code on it, which is useful if they’re not working in the same place. Prospective employers also look at your GitHub profile for examples of code you’ve written.

- ❑ You can make repositories public or private, thereby controlling who gets to see your code
- ❑ You can also accept or reject the changes other people have made to you code.

Click here to create a GitHub profile.

GitHub Profile Example



Melissa R. Luberti
MelissaRose

Warren, MI
melissa.luberti@gmail.com
http://www.detroitdrunkenhist...
Joined on Feb 11, 2014

7	0	9
Followers	Starred	Following

Popular repositories

Chap7Ex1	0 ★
DDHSProject This is	0 ★
DotCom	0 ★
git-demo	0 ★
ProjectEuler An early attempt at TDD	0 ★

Repositories contributed to

mmerino/AsthmaWatch	1 ★

Contributions

Summary of pull requests, issues opened, and commits. Learn how we count contributions.

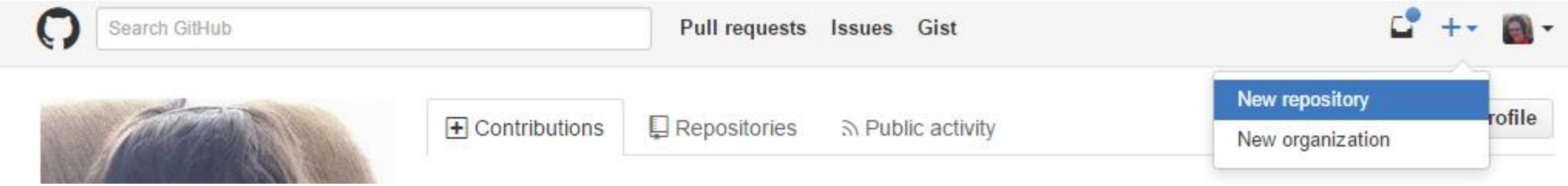
Common GitHub Terms

- Push – Sending the changes you've made locally to the remote repository wo your collaborators can see them.
- Pull – You are fetching changes that have been committed to the repository and merging them with the file you have, thereby getting the most up to date file.
- Pull Request - Proposed changes to a repository submitted by a user and accepted or rejected by a repository's collaborators.
- Merge - Taking the changes from one branch (in the same repository or from a fork), and applying them into another.
- Branch – A parallel version of a repository that is contained within that repository, but doesn't affect the master branch.
- Fork – A personal copy of another user's repository that lives on your account, allowing you to freely make changes without affecting the original.

Creating a Repository in GitHub

After you have installed Git and created a GitHub profile, complete the following steps to create your first repository:

□In your GitHub profile, click on the + and navigate to “New Repository.”



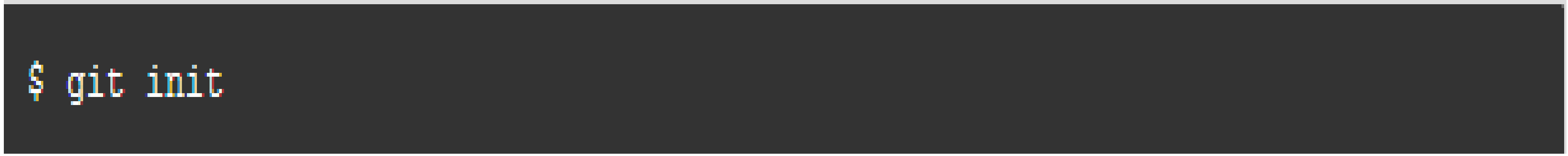
□Name your new repository. *Protip: Name it something clear and to-the-point*

□We’re going to leave the access setting as public.

□Click “Create Repository” button.

Uploading Files to GitHub Using Git

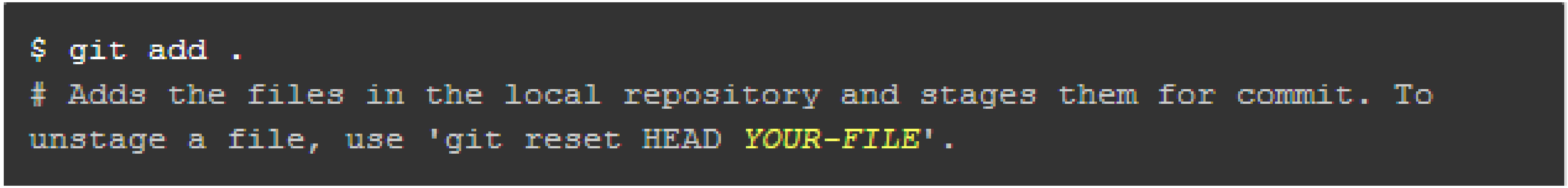
- Open Git Bash
- Change the current working directory to your local project.
- Initialize the local directory as a Git repository.



```
$ git init
```

Uploading Files to GitHub Using Git

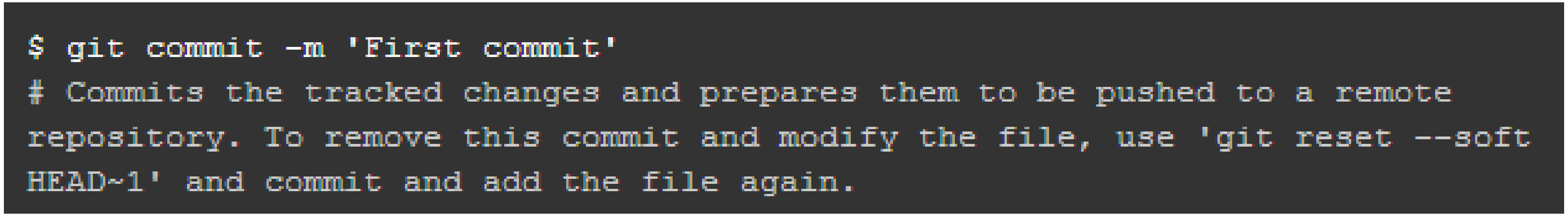
- Add the files in your new local repository. This stages them for the first commit.



```
$ git add .
# Adds the files in the local repository and stages them for commit. To
unstage a file, use 'git reset HEAD YOUR-FILE'.
```

Uploading Files to GitHub Using Git

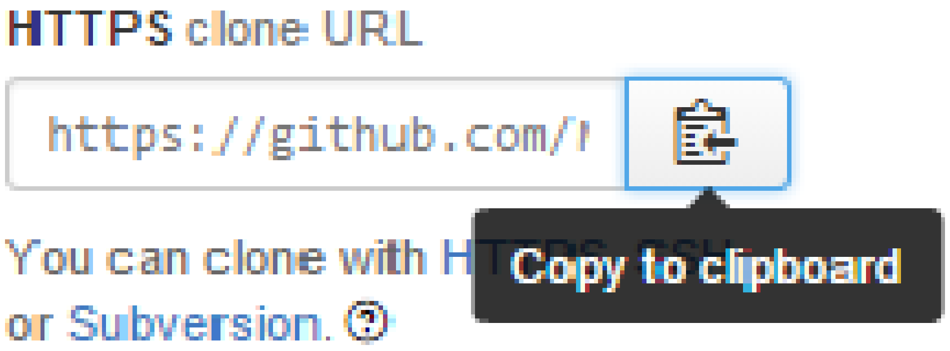
- Commit the files that you've staged in your local repository.



```
$ git commit -m 'First commit'
# Commits the tracked changes and prepares them to be pushed to a remote
repository. To remove this commit and modify the file, use 'git reset --soft
HEAD~1' and commit and add the file again.
```

Uploading Files to GitHub Using Git

- ❑At the top of your GitHub repository's Quick Setup page, click to copy the remote repository URL.



HTTPS clone URL

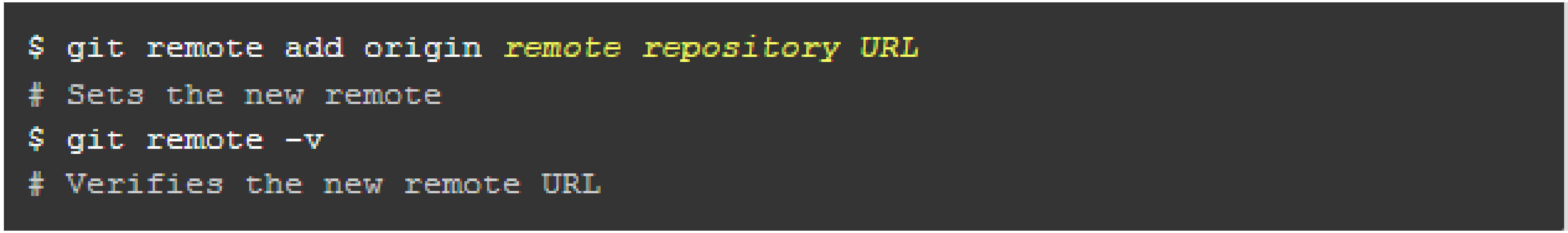
https://github.com/r

Copy to clipboard

You can clone with HTTPS, SSH, or Subversion. ⓘ

Uploading Files to GitHub Using Git

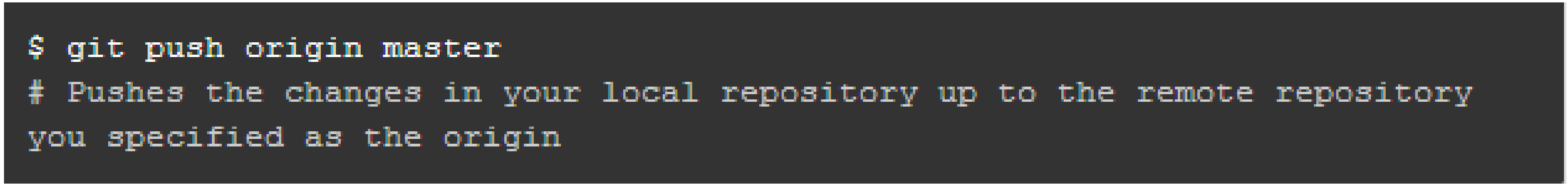
- In the Command prompt, add the URL for the remote repository where your local repository will be pushed.



```
$ git remote add origin remote repository URL
# Sets the new remote
$ git remote -v
# Verifies the new remote URL
```

Uploading Files to GitHub Using Git

- Push the changes in your local repository to GitHub.



```
$ git push origin master
# Pushes the changes in your local repository up to the remote repository
you specified as the origin
```

Congratulations, You’ve Mastered Version Control!