Git & Github 101

Version control and you.

Git?

- Git is a *distributed version control system*.
- It allows for easy collaboration and version control.

- It was created in 2005 by Linus Torvald for development on Linux. Today, it is one of the most popular versioning systems out there.

How does Git work?

Repositories!

- It holds the history of all changes.
- In a general sense, the repo is a directory.

- You use Git through a CLI (Windows Command Prompt, Terminal, et cetera)

How would *I* use Git?

- Collaborate with others.
- Get the correct solution for a problem.
- Have your work corrected.
- Make your work available to the world!

How do I use Git?

The Clone Wars

To get the code and history of any repository, you *clone* it.

```
git clone git://github.com/eturk/jack
```

History of the World, Part I

To see the history of any repository, you see the *log* of it.

```
git log
```

Commit!

Every change in the history of a project is represented by a *commit*. Each commit has a SHA-1 ID.

Branches

- A *branch* is a different version of the same project.
- You use a branch for keeping some code separate from the main branch (usually called “master”).
  - For example, I use the branch “stable” to differentiate between my development (“master”) and my code that is ready for use (“stable”).

Branches

You can see the current branch you're using.

```
git branch
```

And change the current branch.

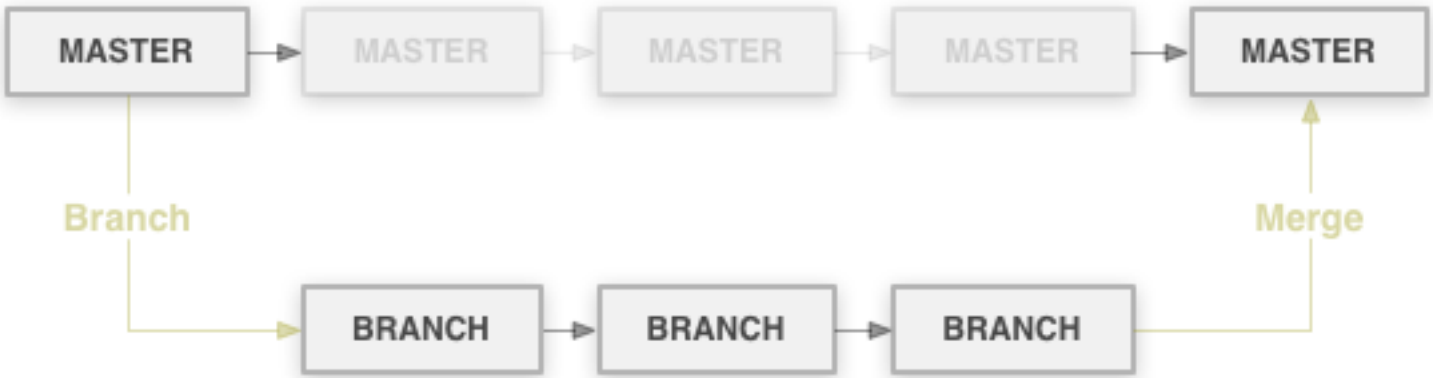
```
git checkout foobar
```

Merging

- Once you've isolated a branch, you'll want to incorporate those changes back into the main branch. You want to select the branch you want to merge into and specify the branch you're merging from.

```
git checkout stable
git merge master
```

Branches & Merging



Branches & Merging

- When you merge, Git will show you a *diff*.
- A diff is the difference between the current code and the code you want to merge in.

```
git diff
```

The Downfall of the Magical Merge

- When the same block of code is edited in two branches, Git doesn't know how to merge.
- Instead, it'll give us a "merge conflict" error and insert markers in the file where the difference is.

The Downfall of the Magical Merge

```
<<<<<<< HEAD
Many Hello World Examples
=======
Hello World Lang Examples
>>>>>>> fix_readme
This project has examples of hello
world in nearly every programming
language.
```

The Downfall of the Magical Merge

- What do I do?!
  - Resolve it manually by changing the file to the correct code. Remove the markers and re-add it with git add.
  - Undo the changes on the current branch by using…

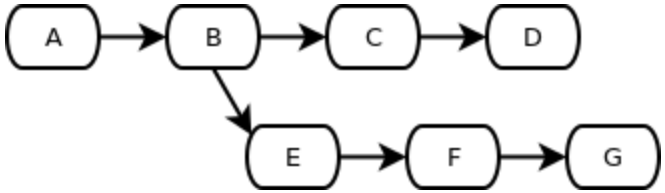
Stash to the rescue!

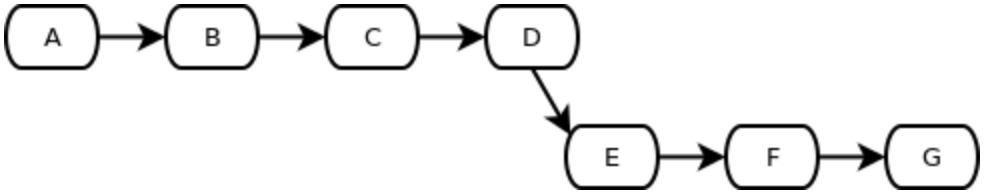
- If you have a merge conflict, using *stash* will throw away the changes to the current branch and allow you to merge.

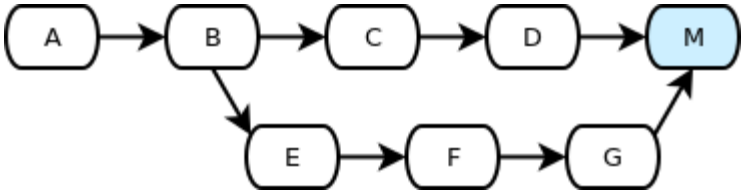
```
git stash
```

All Your Rebase Are Belong to Us

- Instead of merging with last commit, you use the latest commit.







Push & Pull

- To move your changes to a remote repository (a.k.a. GitHub), you *push* the changes.

```
git push [remote] [branch]
```

- To get the latest changes to a remote repository, you *pull* or *fetch* the changes.

```
git pull [remote] [branch]
```

Example Workflow

```
(make some changes)
git add .
git rm some/unneeded/file
git commit –m ‘Made some changes, removed an unneeded file.’
git push origin master
```

Example Workflow

```
git checkout master
(make some changes)
git checkout stable
git merge master
git push origin stable
```

Best Practices

- **Commit Often**
  - When writing a school paper, you save your document often. The same applies to Git.
- **Pull Often**
  - Having the latest version of the code will cut down on the amount of times you have merge conflicts.

Collaboration with Git

You're working on a project called *HelloWorld* with Joe & Bob.

- Everyone has their own repository on their computer and hosted on the Internet.
- Your online repository is the official repository. Joe and Bob's are for development (so when someone wants to use *HelloWorld*, they would get the code from your repository).

Collaboration with Git

- Joe's making a new feature. He would work on it and push to his online (or "remote") repository.
- When you're satisfied that it works, you'd incorporate it into the main repository by "pulling in" his code into a new branch on the remote repository.
- You'd then merge the new branch into the main branch.
- Then, Bob would run git pull on his computer's (or "local") repository to get the latest code.

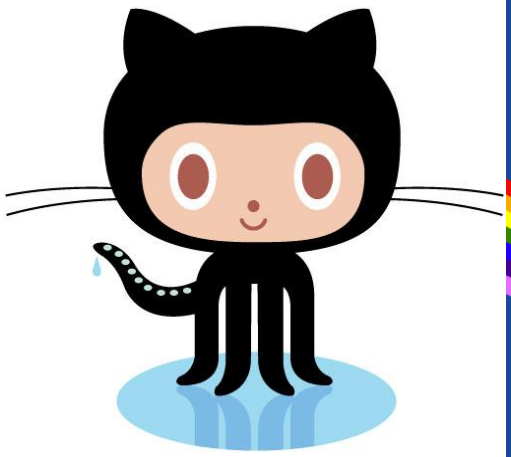
Remember, commit and pull often!

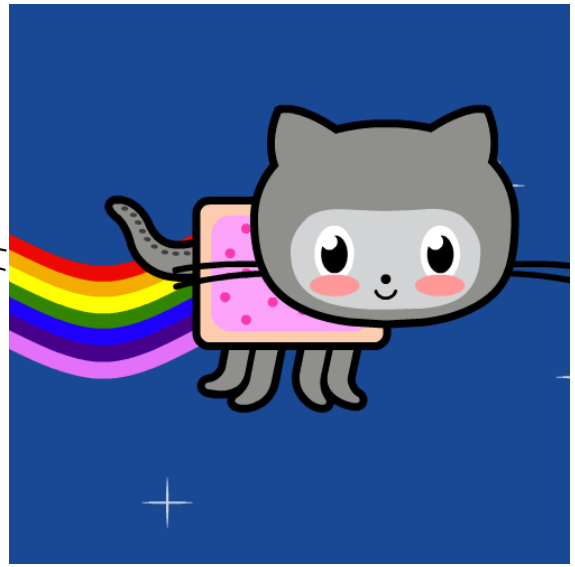
Where do I host online?

- For the longest time, http://repo.or.cz has been the standard… until GitHub came along.

GitHub

Founded in 2008 by Chris Wanstrath, PJ Hyett, and Tom Preston-Warner, it has grown to 1 million users and over 2 million active repositories.



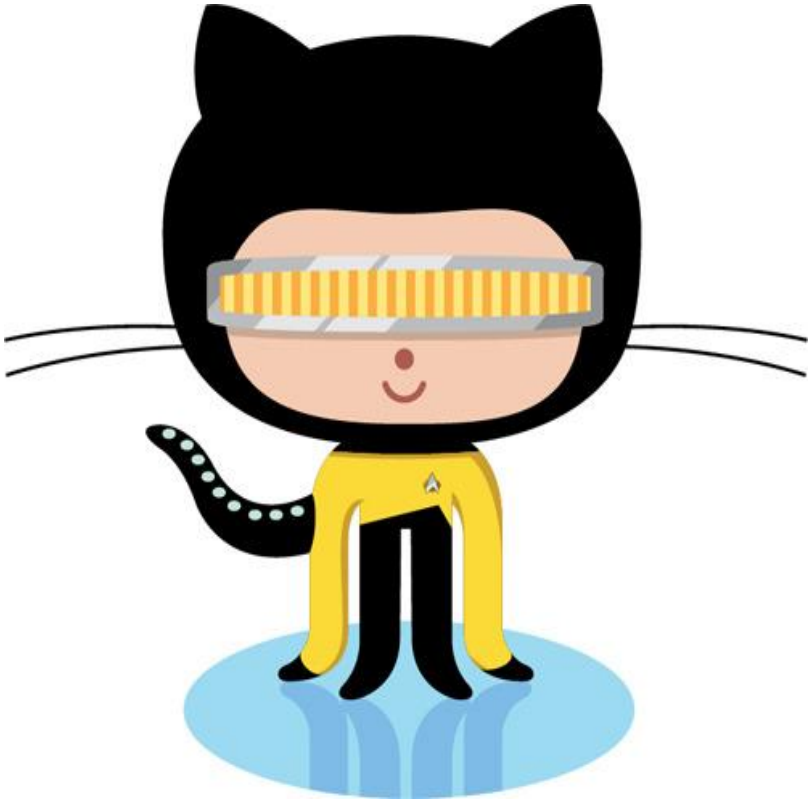


The Slide of Features™

- Browse code online with syntax highlighting.
- View file history.
- Blame and annotations (view who made changes when and where).
- Online editor.
- Git-powered Wikis.
- Integrated issue tracking, with milestones, labels, search.
- Code Review
  - Pull Request = Code + Issue + Comments
  - Comments (comment on commits, files, or even specific lines).
  - Network graph (shows all forks).
  - Compare view (see differences in commits)
- Community
  - Watch repositories and users.
  - Profiles
  - Explore!

Who Uses GitHub?

- Linus Torvald and the Linux kernel
- Twitter
- Facebook
- Rackspace
- Yahoo!
- Shopify
- EMI
- Six Apart
- Sun/Oracle
- Node.js
- Apache



How do I start using GitHub?

1. Go to https://github.com/signup/free and signup.
2. You'll need a SSH key.
    1. Go to Start, All Programs, Git, Git Bash.
    2. Type ssh-keygen -t rsa –C your_email@whatever.com.
    3. Enter a passphrase (twice).
    4. Open S:\.ssh\id_pub.rsa in Notepad. Copy the contents *exactly how they are*.
    5. Go to https://github.com/account/ssh
    6. Click "Add Public Key". Name it something descriptive.

How do I start using GitHub?

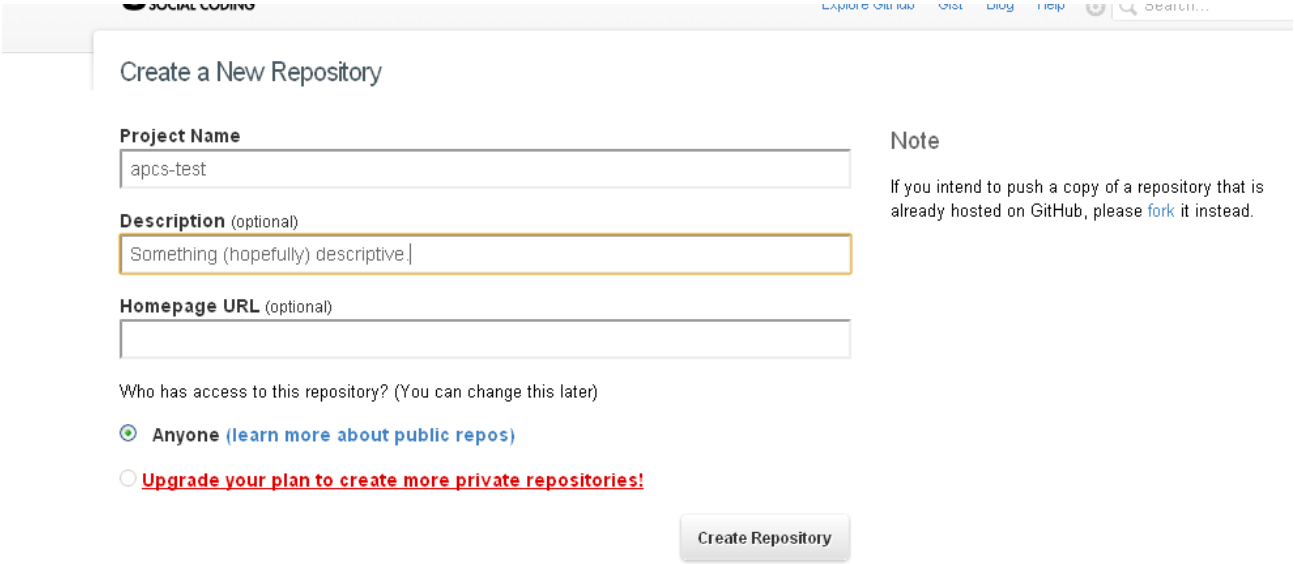
7. Paste the contents into the “Key” field.
8. Click “Add Key”.
9. To make sure it works, enter the command ssh –T git@github.com
10. It should ask you to connect. Type yes and hit enter.
11. Now it should say “Hi, [username]! You've successfully authenticated, but Github does not allow shell access.”

How do I start using GitHub?

2. You'll need to tell Git your name and email. Do this by using the following commands.
   - `git config global --user.name "Your Name"`
   - `git config global --user.email "your@email.com"`

Create a new repository.

- Go to your Dashboard and click “Create Repository”.
- Enter a name and (optionally) a description.



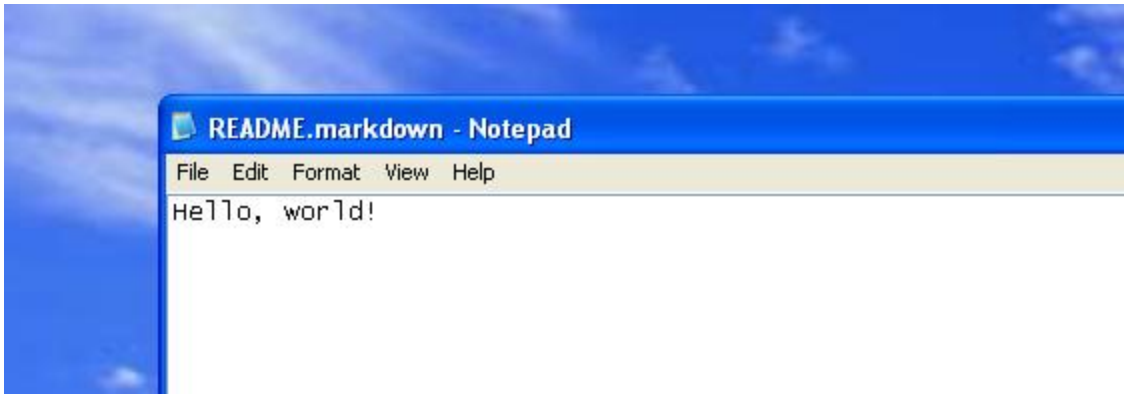
Create your project locally.

In Git bash:

- `mkdir apcs-test && cd apcs-test`
- `git init`
- `touch README.markdown`

Create your project locally.

Open README.markdown in Notepad, and insert some text, like such.



Give Git the remote repository info.

In Git bash:

- `git remote add origin git@github.com:[username]/[repo].git`

Commit & Push!

```
git add .
git commit –m ‘Initial commit!’
git push –u origin master
```

Fork this!

- On GitHub, you can *fork* repositories.
- This will give you your own repository of the code. Clone it, make some changes, and push to your repository.
- Then, on the main repository (the owner's), make a *Pull Request*.

Pull Requests

- Pull Requests alert the project's owner of your changes.
- You are requesting them to merge your changes into the project.
- These can be attached to *Issues*.
- Pull Requests = Code Review

Issues

- Issues are attached to a project.
- If you see something is not working with a project, create an issue. If you want a feature, create an issue.
- Keep track of what you need to do.
- Add Labels and Milestones.
- Issues = Discussion

Gists

- A Gist is an easy way to share code snippets and pastes with others.
- They are Git repositories, so they are versioned, forkable, and you can use them like any other Git repository.

Futher Reading

- Git Tutorial http://coo.ly/smi
- Git Reference http://coo.ly/KDJ
- Version Control for Designers http://coo.ly/vvJ