git Basics

See nycda.com/git for a step-by-step guide

© New York Code + Design Academy 2016

Learning Objective

- Understand what `git` is and where it is used
- Make your first, "commit"

© New York Code + Design Academy 2016

Getting Setup

Please download git, if you don't already have it:
http://git-scm.com/downloads

If you are a Windows user, use Git Bash (included with Git) for all Terminal and git interactions. To launch, Start Menu -> Programs -> Git -> Git Bash

© New York Code + Design Academy 2016

Where is git?

- Once you've installed git, you can verify it has been installed by opening up the Terminal and typing `git`

```
$ git
```

- If you see a 'command not recognized' error, you probably haven't installed git
- We can solve your issues once the lecture is over - look along with a fellow student for now!

© New York Code + Design Academy 2016

What is `git`?

- A version control system meant to make it easier to have multiple versions of code, sometimes across multiple developers or teams
- At its most simple, `git` helps with the 'indexv1.html, indexv2.html, indexv3FINAL.html' problem
- At its most complex, git allows developers to work together worldwide on code without stepping on each other's toes

© New York Code + Design Academy 2016

Why learn `git`?

- Allows you to see the changes you make to your code and easily revert them
- By pushing your git repositories to GitHub, you'll have a developer portfolio automagically!
- You'll eventually push your code to a remote server using `git` - your application's **host**, so you can release your webpage or web application to the world

© New York Code + Design Academy 2016

Don't get confused - `git` vs GitHub.com

- **git** is a version control system that takes snapshots of your code at certain points in development
- These snapshots are stored in a 'repo', or 'repository' on your local machine
- **GitHub.com** is a website that hosts git repositories on a remote server
- We'll be able to troubleshoot your issues more easily if you're able to identify which of these two separate things you're having trouble with

© New York Code + Design Academy 2016

Configuring git

Your commits will have your name and email attached to them. Before we put anything up in public, let's make sure they're correct!

```
$ git config --global user.name
> should be your name, i.e. Zach Feldman
$ git config --global user.email
>should be your email, i.e. zach@nycda.com
```

To fix either, just add the desired value in quotes after the command:

```
$ git config --global user.name "Zach Feldman"
$ git config --global user.email "zachfeldman@gmail.com"
```

© New York Code + Design Academy 2016

Getting started

- Before using `git`, you'll need a **project** to use it with
- A **project** is any directory (folder) full of files - this could be a website, web application, or really any collection of files
- Use the `cd` command to get to the directory you'll be using git in

© New York Code + Design Academy 2016

Getting started: `git init`

- Once you're in your project's directory, run the `git init` command to initialize git

```
$ git init
```

- You only need to run this command **once** per project
- Unsure if you've initialized yet? Try running `git status` - if you get a `fatal error`, you haven't run `git init` yet

© New York Code + Design Academy 2016

Making your first commit

- A commit is a 'snapshot' of your project at a certain time
- It tracks all of the changes you've made since the last commit
- Two steps to make the commit:
  - **stage** your files: choose which files to commit
  - **commit** tell git to make your commit with an accompanying descriptive message

© New York Code + Design Academy 2016

Staging files

- To stage an individual file, use `git add`

```
$ git add index.html
```

- To stage every file/change since your last commit, just use -A or '.'

```
$ git add -A
-- OR --
$ git add .
```

© New York Code + Design Academy 2016

Making a commit

- Once your files are staged, you can make a commit using the `git commit` command
- Always use the `-m` flag to add a descriptive commit message

```
$ git commit -m "Initial commit."
```

© New York Code + Design Academy 2016

Tip: Use descriptive commit messages

If you use descriptive commit messages, it'll be much easier to see how your project has progressed later on

```
$ git commit -m "Initial commit."
```

```
$ git commit -m "Added 'about' to the navigation bar and a page for it."
```

```
$ git commit -m "Closes #15 by adding a blue background on hover."
```

© New York Code + Design Academy 2016

Tip: If you forget a commit message...

If you forget to add a commit message and hit enter after `git commit`

```
$ git commit [Enter]
```

You may end up in what's called vim. To exit this, simply hit the escape key,
then type the following.

```
$ :q!
```

© New York Code + Design Academy 2016

`git status`

- You can always run this command 'for free', meaning with no repercussions
- Will help you figure out if
  - `git` is initialized for this project
  - what files are staged
  - what files have been changed since the last commit

© New York Code + Design Academy 2016

Tip: you need files!

- A common beginner mistake is to get excited about starting your new project and try to `git init` before you have any files
- git doesn't work unless you have files to track!

© New York Code + Design Academy 2016

Tip: use `git log`

- Curious what commits have been made so far?
- `git log` will show you what commits have been made so far

```
$ git log
commit 4038fb143edfc068264479cce855619730d6edca
Author: Zach Feldman <zach@nycda.com>
Date:   Tue Nov 25 17:05:28 2014 -0500

    GA tracking stuff.

commit 74ee59894ef22fd714bf3ffb06f2ef4cf43be0bc
Merge: de4b141 6c991aa
Author: Zach Feldman <zachfeldman@gmail.com>
Date:   Tue Nov 25 13:01:54 2014 -0500

    Merge pull request #201 from nycda/classes-page-custom-field-fix
```

© New York Code + Design Academy 2016

Exercise: Getting to know git

- Create a new project folder with at least two files, perhaps `index.html` and `style.css`
- Initialize a new git repository in this folder
- Make your initial commit
- Make a change to one or both of the files, then make another commit
- Repeat this process starting from scratch a few times to make sure you have it down

© New York Code + Design Academy 2016

Resources

- Code Academy: Learn Git
- TeamTreeHouse: Git Basics - Getting Started with Git
- Roger Dudler: Git Simple Guide

© New York Code + Design Academy 2016