Using Git with WordPress

The Atlanta WordPress Coder's Guild

Mike Schinkel and Micah Wood

@thecodersguild

#gitws

Audience

- **Assumed:**
  - ***Professionals building WordPress sites***
  - ***Maybe never coded***
  - ***Maybe never used terminal/command line***

Definitions

- **WordPress Developers work on:**
  - Themes,
  - Plugins,
  - The Database.

Definitions, *cont.*

- **Themes and Plugins are *"Source Code"***
  - *HTML*
  - *CSS*
  - *PHP*
  - *Javascript*
  - *SQL (for MySQL)*
  - *Maybe other(?)*

Professional Workflow

- **Develop** code on a ***local*** computer
  - *A best practice*
- **Test** code on ***"Testing"*** server
  - *To discover code bugs*
- **Stage** content entry on a ***"Staging"*** server
  - *To discover usability issues*
- **Publish** the live website on a ***"Production"*** server
  - *For obvious reasons?*

What is Version Control?

- **From Wikipedia(-ish):**
  - *Version control is a system (e.g. Git) that records changes to a file or set of files over time so that you can recall specific versions later.*

Why Version Control?

- **To restore earlier versions of code**
- **Allows *"What If?"* Coding**
- **To Recover Lost Source Code**
- **To Working Efficient in a Team**

Restoring Earlier Versions

Checkout specific earlier code:

- *If an older version of the source code for a site is the live site and development on the next version is in progress, version control allows you to recover the source code that is running on the live server back to your development machine, fix a bug, redeploy to the live site, and then return back to working on the next version of the site.*

Allows *"What If?"* Coding

Branch **the current code:**

- *i.e. You want to implement an new feature but are not sure how to do it and do not want to fear breaking the working source code when you first try to implement*

Recover Lost Source Code

Clone the repository:

- *i.e. you forget to backup the source code. Doh! With version control, the source code is never lost; you just recover from version control.*

Working Better in a Team

Merge different commits:

- *If more than one person is changing source code at a time -- i.e. a backend developer, a front-end themer and a Javascript developer -- chances are great one person will overwrite code another has written. Version control acts as the **traffic cop** for multiple developers.*

Which Version Control?

- Subversion?
  - *subversion.apache.org*
- Mecurial?
  - *mercurial.selenic.com*
- Git?
  - *git-scm.com*

Use Git

- Most widely-used version control
- Terminal/Command line tool
- Open-source, free to everyone
- Works on Mac, Windows, Linux

Git Terminology #1

- **Repository:**
  - *A **directory** (`.git`) containing all project files (including docs), and stores each file's revision history.*
- **Initializing:**
  - *To create a **new** repository on the local computer.*
- **Cloning:**
  - *The process of **copying** a repository of source code from a remote computer to your local computer.*

Repository/Init/Cloning Exercise

Git Terminology #2

- **Status:**
  - *The **current state** of all the files in the repository*
- **Commit:**
  - ***Send** a set of files changes to the repository*
- **Adding:**
  - ***Append** file(s) to list of files ready for commit*

Status/Commit/Add Exercise

Git Terminology #3

- **Checkout:**
  - *Extracting previously committed code out of the repo to create a new state of file*
- **Head/Detached Head**
  - *A pointer to the most recent commit/When Head points to a commit ID, not a named branch.*
- **Branch:**
  - *A named collection of the files changes*

Checkout/Head/Branch Exercise

Git Terminology #4

- **Merge:**
  - ***Combine*** *one branch of code with the state of another branch*
- **Merge Conflict**
  - *When the* ***same file(s) in the two branches*** *being merged had their* ***same line(s) of code changed****. Must be manually merged then recommitted.*
- **Reset**
  - *Rolls back changes to a previous commit. (Note:* ***Never*** *reset when you have already sent code to a remote server.*

Merge/Conflict/Reset Exercise

Git Terminology #5

- **Stash:**
  - ***Sets aside*** *uncommitted changes so Git will allow you to checkout an existing branch.*
- **Remove:**
  - ***Subtracts*** *files from list of files to commit (opposite of Add.)*

Stash/Remove Exercise

Working with Remotes

- **Remote Repositories**
  - *Versions of your project that are hosted on the Internet or on your network somewhere on a "Git Server."*

Hosting Your Git Repos

- **Run your own Git Server**
  - *But who really wants to do that?*
- **Or Use a Commercial Git Host**
  - *Like all the pros do.*
  - *Unless you are a large corporate or well funded startup!*

Commercial Git Host

- **GitHub**
  - github.com
- **BitBucket**
  - bitbucket.org
- **Pantheon**
  - pantheon.io

github.com

- Free for open-source repositories
- Charges fee for private repos
- Generally Charges per repo
- Most popular for open-source

bitbucket.org

- Free for open-source repositories
- Free for private repos
- Four (4)+ one(1) users are free
- Charges fee for number of users
- Most popular for small teams

pantheon.io

- A WordPress Web Host
- Deploy via Git *(vs. SFTP)*
- Free for development and testing
- Pay when you deploy to a live site

Git Terminology #5

- **Stash:**
  - ***Sets aside*** *uncommitted changes so Git will allow you to checkout an existing branch.*
- **Remove:**
  - ***Subtracts*** *files from list of files to commit (opposite of Add.)*
- **Remote:**
  - *A repository* ***hosted elsewhere*** *on the Internet or your network somewhere with a version of your project files that can be synced.*

Pantheon Exercise

Git Terminology #6

- **Push:**
  - ***Submit*** *committed changes to a remote Git server to make your changes available to other developers.*
- **Pull:**
  - ***Retrieve*** *committed changes from a remote repo that other developers have pushed.*

Push/Pull Exercise

End of Part 1