Developer Collaboration

* Why collaborate
  + Develop in timely fashion
  + Check mistakes
  + Divide work
  + Allows developers to focus and be good at one individual thing
* Think of branches as public and private
  + Master should match what is out on production and is public
  + Branches are WIP stuff and private
  + Commits are what will be added to master in the next update
* GIT workflows
  + Feature branch workflow
    - All features have different branches
      * Open a new branch every time you start on a new feature
    - Utilizes pull requests
      * Used to discuss changes before they end up in official project
    - Always work off of a branch, never off master, to avoid breaking things
    - Keeps things separate in case of mistakes
    - Feature branches SHOULD have descriptive names
    - Feature branches should be pushed to the central repo for easy access without messing with official code
  + Pull requests
    - When a feature is completed, don’t immediately push to master. Instead, push to central server and file a pull request asking to merge to master
    - Code review is a major benefit of pull requests, but they’re actually aa generic way to talk about code
    - If you need help with particular feature, file a pull request to get help and start a conversation
* GITflow workflow
* Defines a strict workflow designed around releases
* Solid framework for large projects
* Assigns very specific roles to each branch for how and when to use them
* Also uses feature branches
  + Has individual branches for preparing, maintaining and recording releases
  + Very release driven
* Still uses central repo for communication, only difference is the branch structure of the project
* Historical branches
  + Instead of single master branch, you have 2, master (production), and develop (adding what’s going into next release)
  + Develop branch has commits that will be going into next release
* Feature branches
  + Branch off of develop branch and merge into develop
  + When a feature is complete it gets merged into develop
  + Features shouldn’t interact with master, instead they should be pushed into develop for a buffer
* Release branches
  + Forks off of develop before merging to master
  + Creating this branch starts the next release cycle
  + No new features can be added after this point, just bug fixes etc.
  + Should also be merged into develop to keep it up to date
* Common Conventions
  + Branch off: develop
  + Merge into: master
  + Naming convention: release-# or release/#
* Maintenance branches
  + Hotfixes
  + Used to quickly fix production patch
  + Only branch that should fork directly into master
  + When done, it goes into master and develop
  + Version number should be updated
  + Needs communication to keep all branches up to date for bug fixes
  + Change needs pulled into all active branches to avoid merge conflicts
* Github Pages
  + Github pages use Jekyll
    - Jekyll is a templating language
      * Jekyll uses markdown
  + Github pages use Github