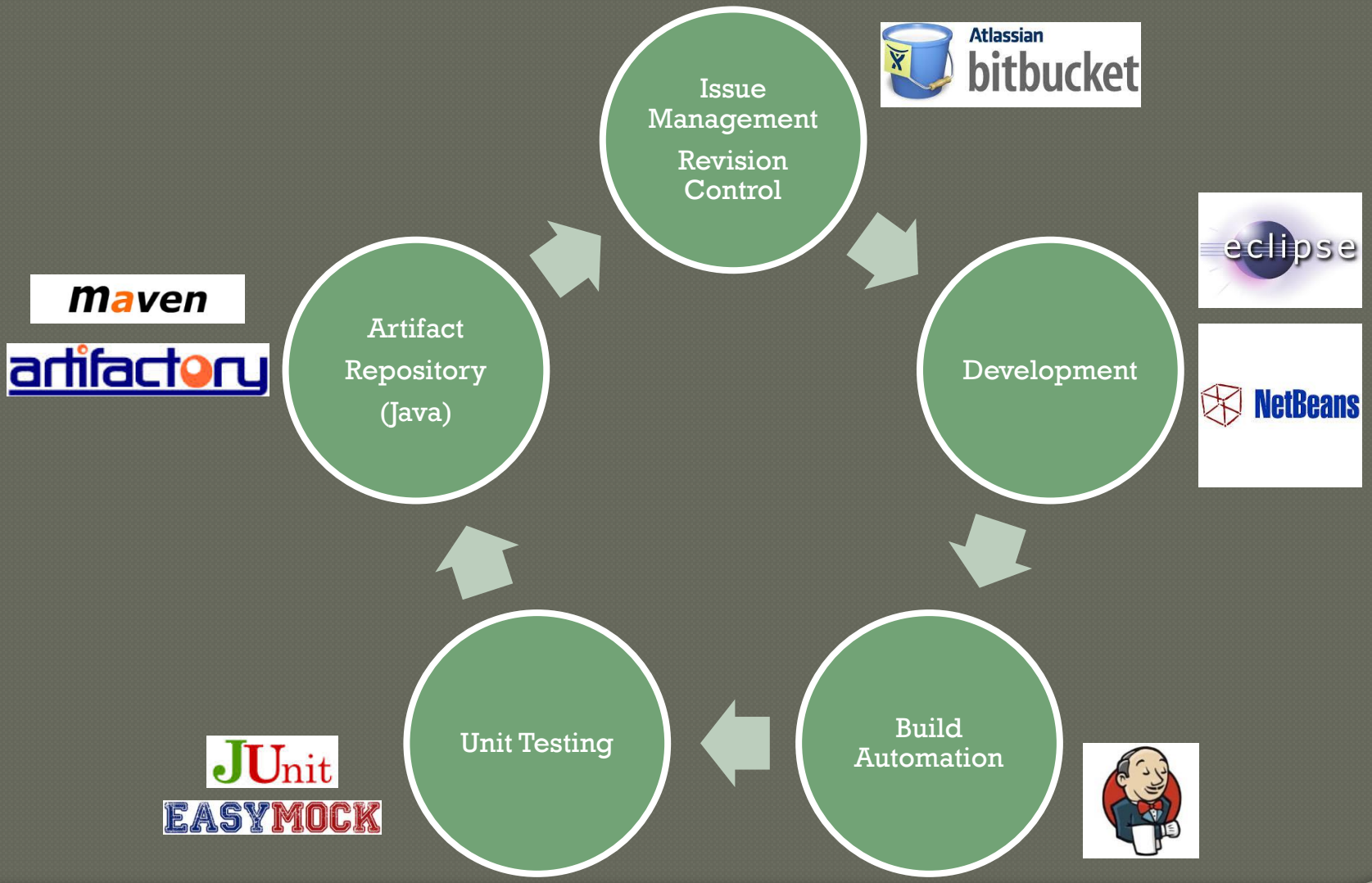
MASS Development Environment

Overview

Matthew Sell, CSSE Student
MASS Research Participant, October 2016

Development Cycle



HELP!!!

- Developer's Reference Guide
  - http://depts.washington.edu/dslab/MASS/docs/dev_quick_reference.pdf
- Git setup, usage help
  - Check Bitbucket for tutorials, videos
  - Command-line: http://git-scm.com
  - "The" book on Git: http://git-scm.com/book
- DSL homepage (manuals)
  - http://depts.washington.edu/dslab/MASS
- Email
  - Professor Fukuda: mfukuda@u.washington.edu
  - Matthew Sell: mrsell@uw.edu

Bitbucket

- Create a free account
  - https://bitbucket.org
  - Use your "UW" email address for upgraded academic account
  - Inform Professor Fukuda or myself to obtain privileges
- Install a Git tool
  - Recommend "Sourcetree" (https://www.sourcetreeapp.com)
- Work from the issue tracker and repository
  - More info on workflow in a bit…

Using Git

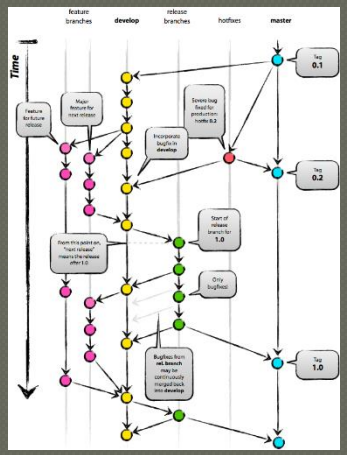
- Branch. *IMMEDIATELY*!
  - Keep your work separate until ready to merge to "develop"
  - Branch name: "<your UW username>_develop"
  - Merging to "develop" means, "My work is ready for release"
- Commit early, commit often
  - Only commit code that compiles!
  - Only commit what YOU changed! (Nothing else!)
  - Provide useful commit messages (and issue #)
  - Commit messages are public!
- Push to origin
  - Multiple development workstations
  - Protects your work (backup)
  - Others can test drive your work

Workflow

- Create issue in Bitbucket
  - Bug / Enhancement / Proposal / Task
- Git Flow – "Start New Feature"
  - "<your initials>-devel" – long running
  - Push to Origin
- Development
  - Unit tests!
- Resolve issue in Bitbucket
- Git Flow – "Finish Feature"
  - Okay to delete branch

Workflow - Git

- Using Git Workflow
  - http://danielkummer.github.io/git-flow-cheatsheet/
  - "master" branch for releases ONLY
  - "develop" branch for merging changes
    - Development branch is always a "Release Candidate"
  - Use feature branching from "develop" for your changes



Jenkins CI

- Automated build from Git repository
- Unit testing / code coverage
- Build / test failure notifications
- Browser based, hosted at UWB
  - http://fukuda-cent-01.css.uwb.edu/jenkins

Assume: "If it doesn't build for Jenkins, it won't for anyone else"

Why Maven?

- Managing dependencies
  - Ensures proper revisioning
  - Dependencies of dependencies (!)
    - More info: http://en.wikipedia.org/wiki/Dependency_hell
- Making consistent builds
  - Between developers
  - Across platforms
  - Using different IDEs

Final Comments…

- Don't commit messy code!
  - Let the IDE help you
  - Think about maintenance
  - Allow interviewers to see your code
  - Tips: https://www.troyhunt.com/10-commandments-of-good-source-control/
- Do something useful with exceptions
- Clever is a "code smell"
  - Others: http://blog.codinghorror.com/code-smells
- Don't reinvent the wheel…
- Unit tests!
- All of these tools and experience is valuable on your resume!

Questions?


YOUR SAMPLE SIZES ARE SMALL
YOUR STANDARD DEVIATIONS ARE HIGH
YOUR CONCLUSION MEANS NOTHING
AND YOU SHOULD FEEL BAD
quickmeme.com