Drupal and Continuous Integration

DrupalCampNJ - 2014

Who we are

Henry Umansky
Princeton University
humansky@princeton.edu

Jason Howe
Drew University
jhowe@drew.edu

What is Continuous Integration?

“Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day.”

-Martin Fowler

Principles of CI?

- Maintain a code repository
- Automate the build
- Make the build self-testing
- Everyone commits to the baseline every day
- Every commit (to baseline) should be built
- Keep the build fast

Principles of CI (cont'd)

- Test in a clone of the production environment
- Make it easy to get the latest deliverables
- Everyone can see the results of the latest build
- Automate deployment

Advantages of CI

- prevent integration problems
- identify failing code early
- immediate unit testing of all changes
- "current" build for testing, demo, or releases

Disadvantages of CI

- Initial setup time required
- Well-developed test-suite required to achieve automated testing advantages

Maintain a code repository

- Version Control Systems: git or svn
- use Features as much as possible
- Strongarm Module

Automate the Build

- Hudson/Jenkins
- Drush
- Build triggers

Make the build self-testing

- Drupal Coder Review/Security Review
- Code Quality - [php | css | js] lint
- PhantomJS and Selenium
- Checkstyles/PHPMD

Daily Commits

- Commit at least once a day
- Reduces potential conflicts
- Triggers automated builds

Demo

Automated Deployment

- Identical /dev/qa/prod systems
- Trivial to move code between environments
- Single button click to perform complex tasks

Automated Deployment

- Single button click to deploy production code to n webservers.
- Deployment is simply a “git pull”, executed via remote ssh.
- Auto deployment to dev upon code commit to dev.

Achieving Identical Environments

- Everything has to be automated, no lovingly handcrafted environments.
- Bash script fired by Jenkins, builds databases, instantiates drupal instance in all environments.
- Tools to sync content from prod->dev and code from dev->prod

Known state of environments

- You always know that Dev contains everyone’s latest contributions.
- You always know that Prod is a copy of what’s in the repo.

From the Operations Perspective

- Keep Server configs in (my.cnf, http.conf, etc) in a configuration repository.
- New webhead is trivial:
  - checkout config repo and run setup script.
  - symlinks config files, checkout drupal.
  - Essentially a self-configuring server
- Works for DR too!

Other Advantages for Ops

- Knowing, without a doubt the current system state.
- Ability to spin up a test environment in minutes.
- Aids in troubleshooting and quick issue resolution. (Good comments commits help too)

Questions?