

CIBox. Full stack OpenSource Continuous Integration workflow.

Andrii Podanenko, Architect, FFW
Yuriy Gerasimov, Architect, FFW



Formerly known as Bysted, Propeople, Blink Reaction,
Chainbizz and Geekpolis



Join the conversation on Twitter



FFWglobal



FFWdcon

Oldschool development workflow



Development infrastructure

- single development environment
- separate company dev server (multiple vhosts one per project)
- SaaS solutions (Acquia, Pantheon, ...)

Coding process

- all commits directly to master
- master deployed to dev site (sometimes automatically)
- code review on dev site after deployment

Configuration process

- master database, get pulled to local dev environments (backup & migrate)
- changes to configuration happens on dev site manually

Problems

- shared resources on dev environment -> dev conflicts (cache, solr, mysql, apache)
- too much time to configure all services on local dev desktop (varnish, solr)
- large gap between Dev and Ops (only one guru able to do production release)

“Newschool” development workflow



Local development

- local virtualized environment (vagrant)
- based on ansible provisioning
- based on ubuntu 14.04 (possible 12.04)

Database

- no configuration on demo/stage/prod sites
- code driven development
- database/profile workflow

Code process

- Github pull requests workflow
- code review before merge to master
- code style checks, test runs
- security tests
- QA on builds

Extra tools

- os monitoring to see resources consuming operations (e.g. multi node munin)
- automated complex deployments Acquia deployment
- visual regression testing
- urls health checks

Visual regression

- <http://backtrac.io> SaaS
- screenshots before / after release, diff
- automated scheduled screenshots / diffs
- authenticated user
- register and start using now!

diffs



URLs health checks

- <https://github.com/ygerasimov/website-size-scan>
- scans all URLs in the file, checks sizes of images
- logs 404s, 5xx, etc.
- written on golang

Welcome CIBox

<http://bit.ly/ffw-cibox>



CIBox code structure

CI for project and VM with Drupal initial codebase

**Continuous Integration
Server for a project**

jenkinsbox.yml

**Project tree initial creation
playbook**

github.yml

How to deploy whole CI system

Steps for getting started

1. CI server

- Get virtual or real server from hosting provider (*Ubuntu LTS 64 bit only for now*)
- Make initial config changes to `jenkinsbox.yml` and inventory
- Run ***ansible-playbook jenkinsbox.yml*** from a shell
- Make changes to jenkins UI with credentials to github repo

2. Repository initialization

- Make needed changes to `github.yml`
- Run ***ansible-playbook github.yml***
- push generated codebase folder to github repo
- check Pull Request builder with newly created change to `readme.md`
- profit

jenkinsbox.yml

ansible playbook for installing CI server

- Jenkins installation
- Needed Jenkins's plugins
- LAMP stack + SSL
- PHP Code Sniffer, scss-lint, security linters
- Java JDK
- Jetty & Apache Solr with Drupal configs
- Selenium & Behat packages
- Optimized and preconfigured configs for mysql for speed

Preconfigured Jenkins

A bunch of jobs with scripts for running playbooks

All		
S	W	Name ↓
		ASC_DEMO_TESTS
		ASC_PR_BUILDER
		BACKUP_PROD_DB
		DEMO
		DISK_USAGE_TRIGGER
		SERVER_CLEANER

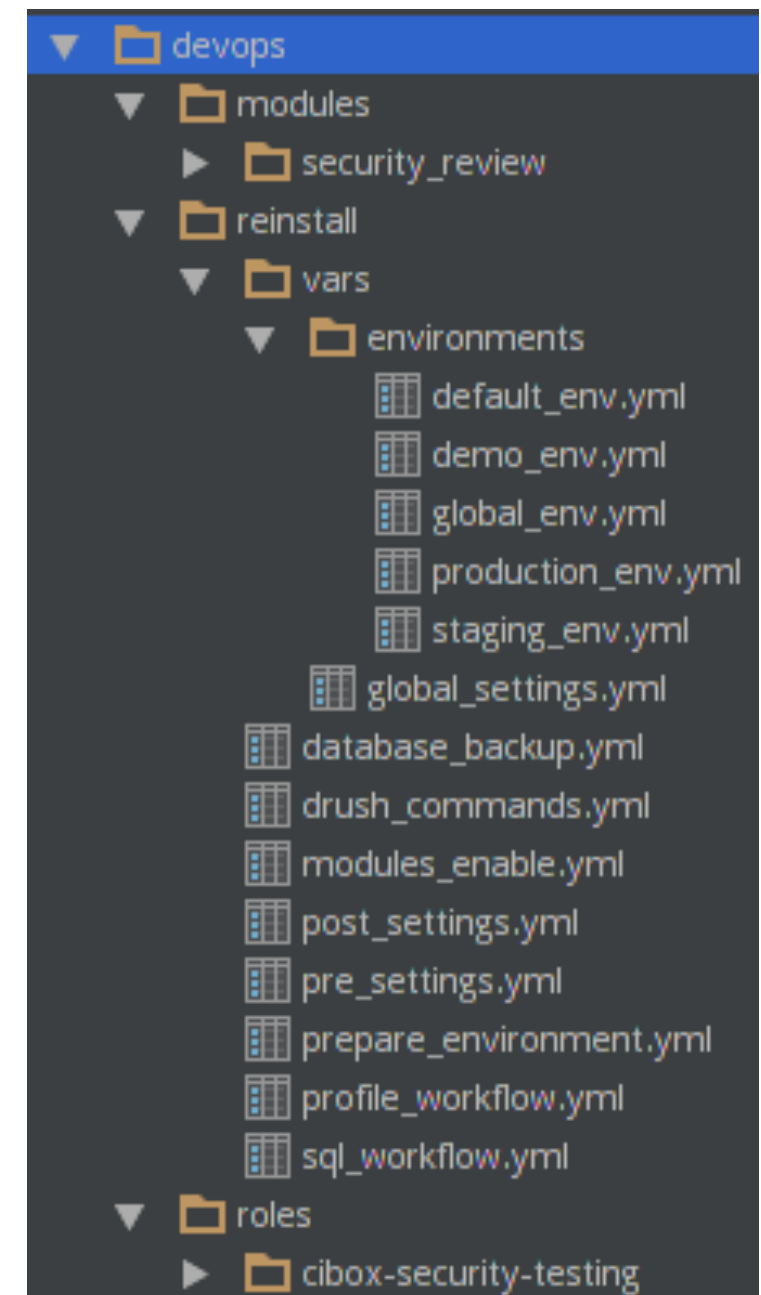
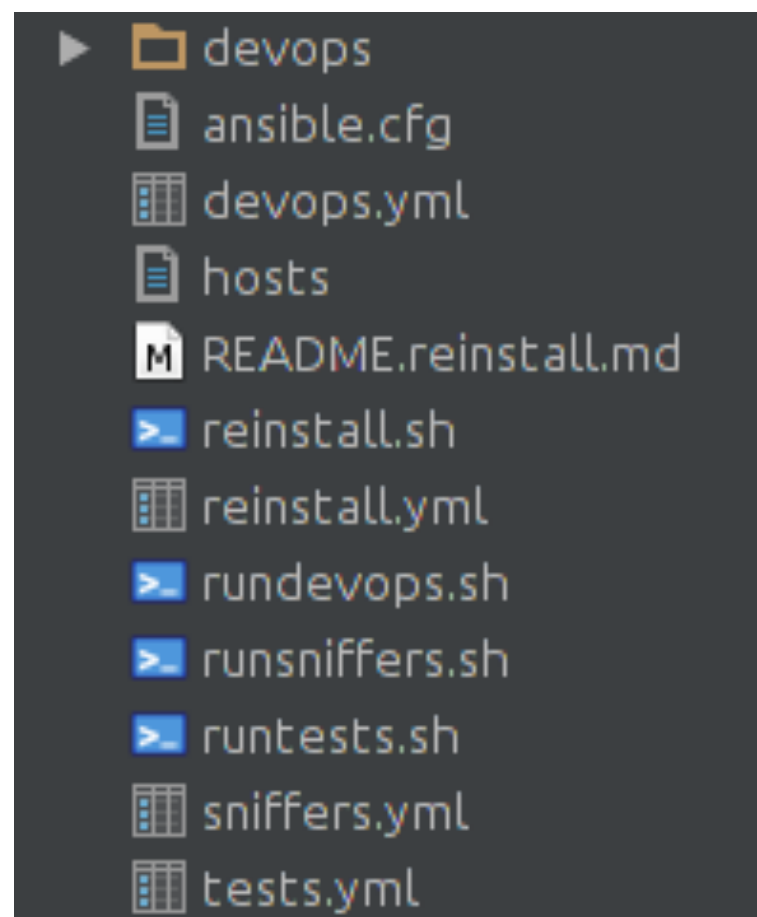
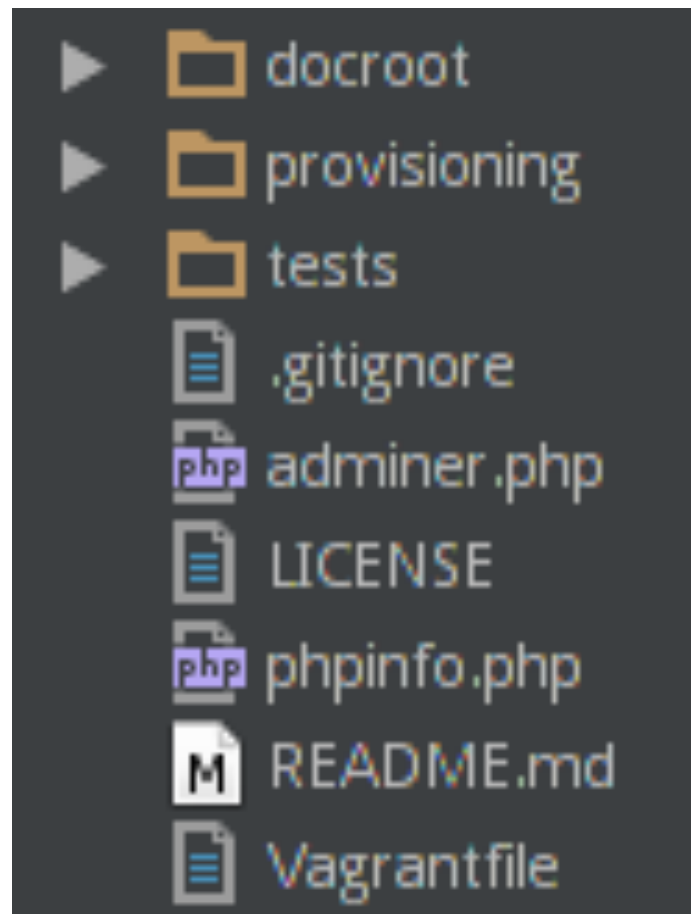
Значок: [S](#) [M](#) [L](#)

- Pull Request Builder
- Skeleton for Backup production database
- Demo site builder
- Disk space cleaner

Main project code structure

Latest drupal, adminer, devops scripts, basic profile

During run of **github.yml** you'll get a codebase that has latest drupal in drupal folder and scripts for future CI builds and tests with Vagrant VM.



Vagrant box

Vagrant + virtualbox (optional lxc) + ansible provisioner

We are using trick for sharing ansible roles between CI server and VM provisioning scripts for making sure we have equal environments for both.

Basic stuff for now (all are inside splitted ansible roles):

composer, pear, ansible, apache, memcached, mysql, php, sdebug, shprof, selenium, behat, drush, jetty solr, phpdaemon, php codesniffer, apache ssl, custom swap.

Just ***vagrant up*** and you are ready to go coding.

Developer point of view

how to work with ClBox

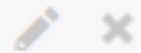


Comments by CI bot



ppbot commented on 1 Apr

Collaborator



Build comment file:

CodeSniffer: Drupal standard file <http://188.40.239.202/build59/Drupalsniff.txt>

CodeSniffer: DrupalPractice standard file <http://188.40.239.202/build59/DrupalPracticesniff.txt>

CodeSniffer: DrupalSecure standard file <http://188.40.239.202/build59/DrupalSecuresniff.txt>

CodeSniffer: Features Drupal standard file <http://188.40.239.202/build59/FeaturesDrupalsniff.txt>

CodeSniffer: Features DrupalPractice standard file

<http://188.40.239.202/build59/FeaturesDrupalPracticesniff.txt>

CodeSniffer: Features DrupalSecure standard file

<http://188.40.239.202/build59/FeaturesDrupalSecuresniff.txt>

JSHint: modules standard file <http://188.40.239.202/build59/modulesjshint.txt>

JSHint: themes standard file <http://188.40.239.202/build59/themesjshint.txt>

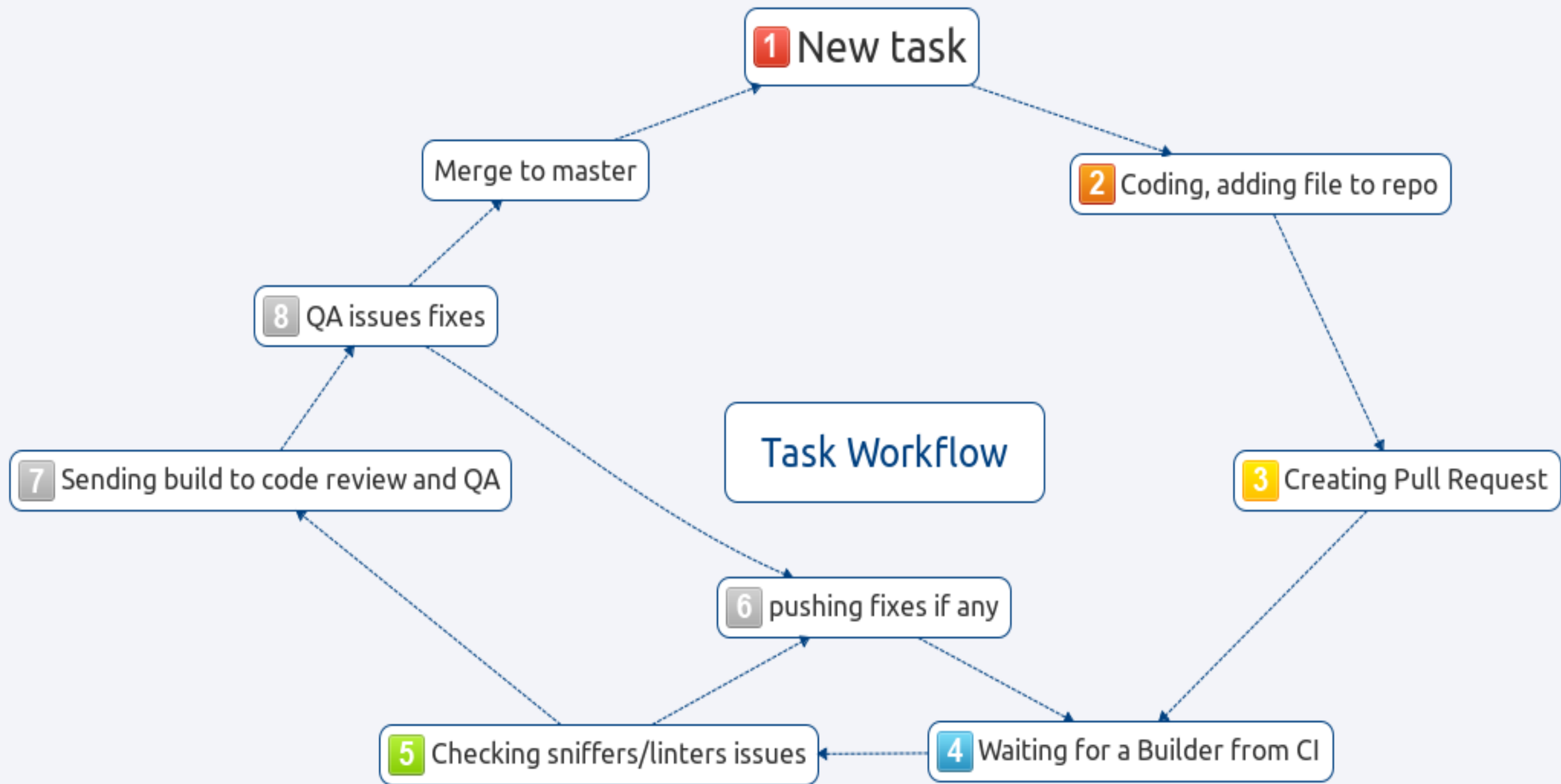
SCSS-lint: themes standard file <http://188.40.239.202/build59/scsslintthemes.txt>

Build site installed at <http://188.40.239.202/build59>



Single task flow

development team point of view



QA: Steps for Review



AndreyMaximov commented 3 days ago

<https://propeople-us.atlassian.net/browse/ANW-737>

Prototype: <http://muledesign.com/clientwork/audubon/review/pages/news/slideshow.html>

There are a lot of degradations came with PR #1187.

Steps for review:

1) Go to

<http://nas.wearepropeople.md/build9792/photos/staring-contest> – standalone slideshow

<http://nas.wearepropeople.md/build9792/magazine/january-february-2015/which-came-first-dinosaur-or-bird> – inline slideshow

<http://nas.wearepropeople.md/build9792/magazine/may-june-2015/2015-audubon-photography-awards-top-100#67> – hero slideshow

2) Make sure standalone slideshow and its control elements look well and similar to prototype.

3) Make sure there is no overlapping with text like <http://screencast.com/t/WottLBhu>

4) Make sure url hashes are correctly red and set

5) Make sure hero slideshow's images are not stretched. Compare with production.

6) Check on mobiles/tablets

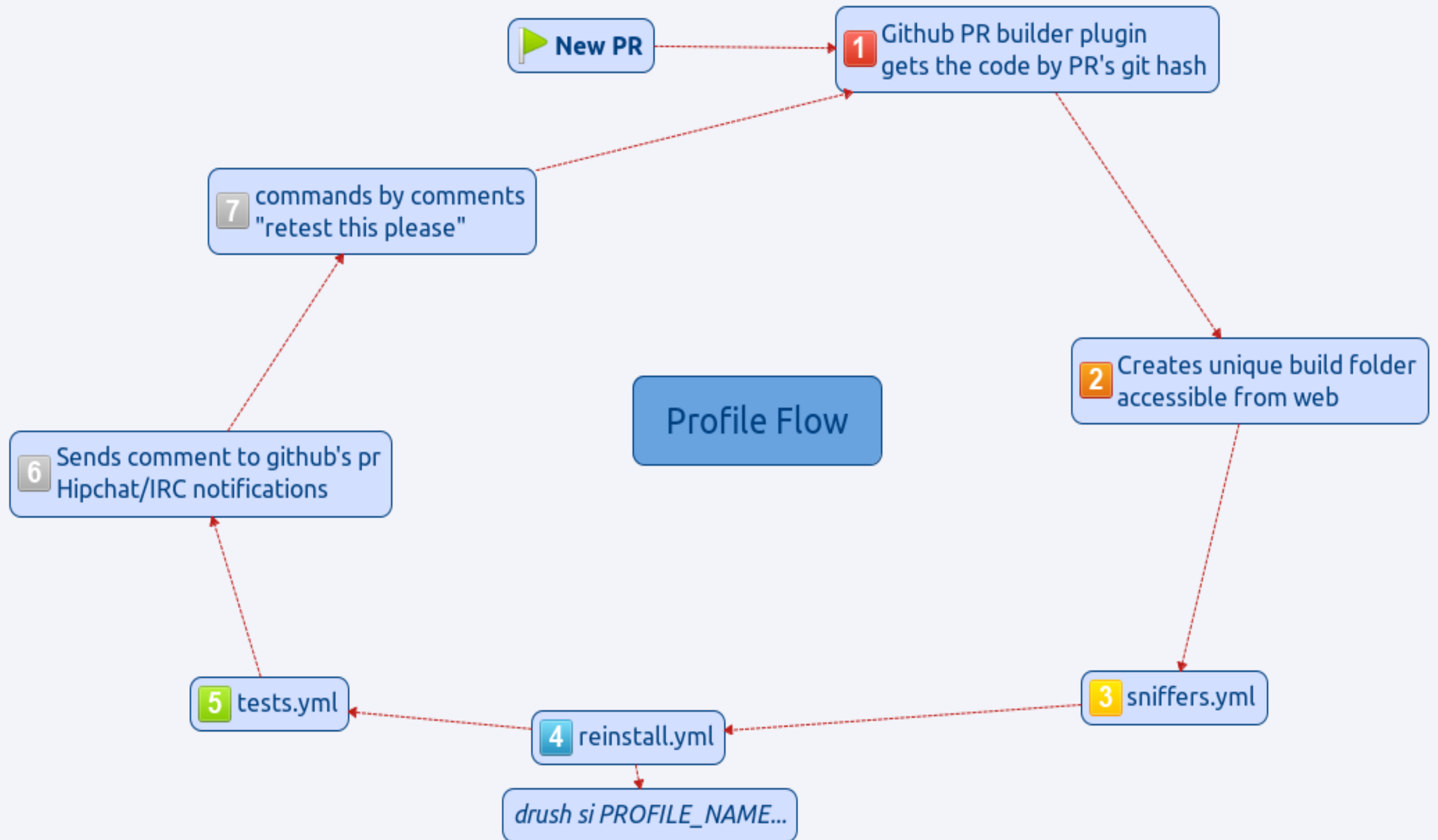
How CI works

technical information



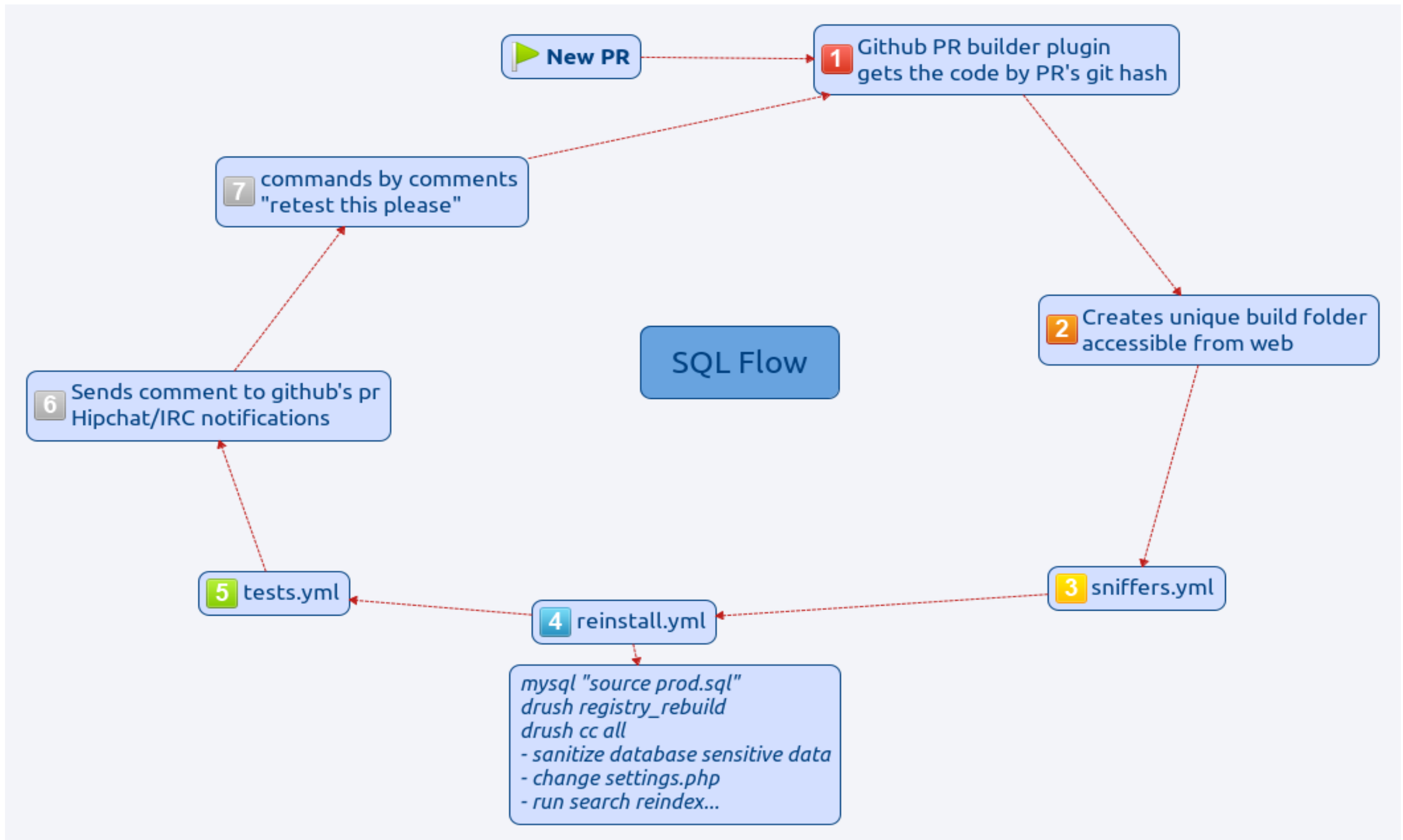
Profile based flow

Reinstalls Drupal from scratch every builder time



SQL based flow

Imports SQL dump every build and prepares it to codebase



Team rules

- One CI bot and per team
- One CI server per project
- Never merge own Pull Request
- Always add Steps for Review
- Select somebody for review by round-robin principle
- Never push directly to main repo master* branch
- Keep two siblings for every role in a team(optional)
- Assign bug fixing task for the guy, who did review for the code with a bug.

****master branch is stable***

Responsibility shift

Due to the fact all DevOps scripts are in the same repo with a project itself - any developer can change workflow at any point.

* Team does manage all the steps for DevOps scripts, no need to involve Ops into the team for full time.

Flow Bottlenecks

- Dependency from github(gitlab, bitbucket)
- If CI server down - team gets stopped on code review step
- New developers should follow new rules. (Coder is tough)
- DevOps must be a team member(s)
- Manual Code Review gets hurt
- Builds are slow on huge projects (Ops needed)
- Decent desktops for a team (SSD is a must)
- Minimal task ≥ 1 hour
- Overall system is pretty complex at start
- Not so easy to start for new teams or companies

How to start using ClBox in your team

- **Contribute*** to ClBox and get more familiar with its subsystems.
- Install the flow for some internal project without tough deadline
- Organize codesprint with ClBox as workflow
- Start to use 1-2 parts of the system and add new parts every following project
 - sniffers.yml
 - tests.yml
 - reinstall.yml
 - Vagrant box
 - ansible scripting(playbooks)
 - Jenkins
 - Github PR for manual code review

ClBox is **opensource, based on popular*

CIBox stable release 1.0.0

- Jenkins 1.624
- A bunch of plugins, stucked to specific, well tested versions
- Apache SOLR 4.x with drupal configs
- LAMP stack (Apache 2.4, PHP 5.6, memcache)
- Speed powered mysql config
- Nice HipChat notification templates out of the box
- Quick Start documentation
- Best practices wiki



Upcoming Features for CIBox 1.1.0

- Acquia deployment plans
- GULP frontend workflow (sass compiling on server side, no more css conflicts within git repo)
- Bigtree-CMS integration (~20 hours for adding any CMS to CIBox afterwards)
- PM docs skeletons within project documentation
- Drupal 8 out of the box (right after D8 release)





Usefull links

- <https://github.com/propeoplemd/cibox>
- <https://github.com/propeoplemd/cibox/wiki>

Presentations

- <http://sanchiz.net/blog/happy-ever-afters-ci-workflow-drupalcamp-kyiv-2015-september-5>
- <http://events.drupal.org/losangeles2015/sessions/multidimensional-testing-workflow-merge-master>
- <http://druler.com/node/888>
- <http://www.slideshare.net/podarok/drupal-continuous-integration-workflow>
- <http://www.slideshare.net/podarok/start-using-vagrant-now>
- <http://www.slideshare.net/podarok/live-deployment-ci-drupal>
- <http://www.slideshare.net/ygerasimov/ci-drupal-camp-berlin-2014>
- <http://www.slideshare.net/ygerasimov/vagrant-stanford-drupalcamp-2014>
- <http://www.slideshare.net/ygerasimov/continuous-integration-stanford-2014>

Blog posts

- <http://wearepropeople.com/blog/how-we-use-vagrant-in-our-drupal-development-workflow>
- <http://wearepropeople.com/blog/building-quality-into-drupal-development-workflow>

Thank you

Visit us at booth 501

Questions?

<http://bit.ly/ffw-cibox>

