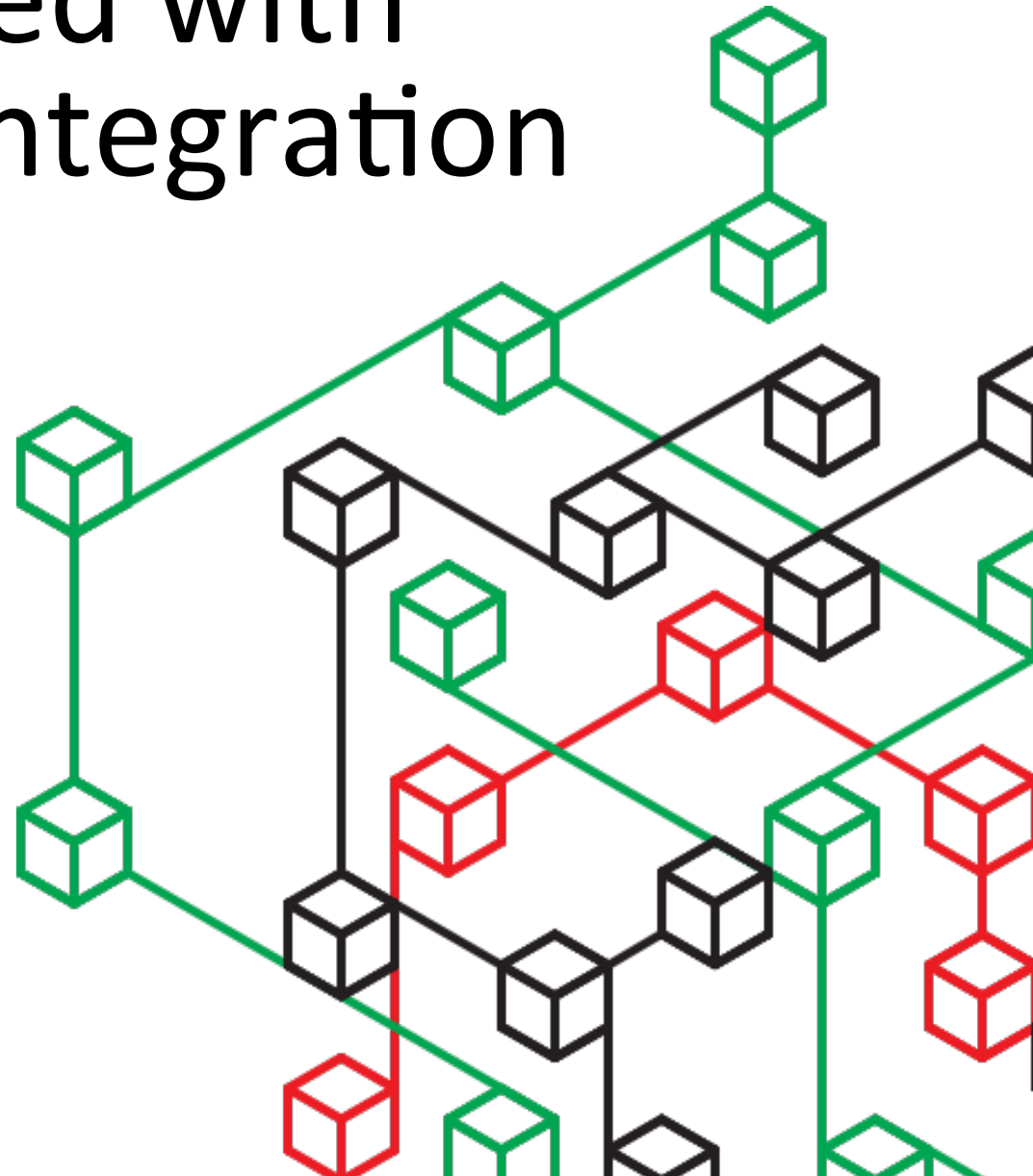


# Getting Started with Continuous Integration

Beth Tucker Long  
@e3betht



# Who am I?

## Beth Tucker Long (@e3betht)

- PHP Developer at Code Climate
- Stay-at-home mom
- User group leader
- Mentor



# Audience Participation?

- Yes, there will be. So, when I ask the audience a question, don't be shy about answering.

# Continuous Integration

That's only for the big guys.  
My team is small, my projects are small.  
So, why am I up here?



# What is continuous integration?

According to Wikipedia:

In software engineering, continuous integration (CI) implements continuous processes of applying quality control — small pieces of effort, applied frequently. Continuous integration aims to improve the quality of software, and to reduce the time taken to deliver it, by replacing the traditional practice of applying quality control after completing all development.

[http://en.wikipedia.org/wiki/Continuous\\_integration](http://en.wikipedia.org/wiki/Continuous_integration)

Martin Fowler -

<http://martinfowler.com/articles/continuousIntegration.html>



# 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. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly.

# Step 1

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.



## Step 2

Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible.

# Code Quality

## Why?

- Easier to test
- Easier to measure
- Easier to follow
- Streamlines the development process

# Measuring Code Quality

- PEAR coding standard -  
<http://pear.php.net/manual/en/standards.php>
- PEAR2 coding standard -  
<http://pear.php.net/manual/en/pear2cs.php>
- PHP Framework Interoperability Group (PHP-FIG) -  
<http://www.php-fig.org/>

# PEAR Coding Standard

- Use an indent of 4 spaces, with no tabs.
- Control Structures:

```
if ((condition1) || (condition2)) {  
    action1;  
} elseif ((condition3) && (condition4)) {  
    action2;  
} else {  
    defaultAction;  
}
```

# Custom Standards

- Broad
- Strict, but flexible
- Based on a “standard” standard
- Everyone must follow

# PHP\_CodeSniffer

"tokenizes your PHP, JavaScript and CSS files and detects violations of a defined set of coding standards"

[http://pear.php.net/package/PHP\\_CodeSniffer/](http://pear.php.net/package/PHP_CodeSniffer/)

- PEAR package
- Single file or entire directory
- Preset and customizable
- Will fix items for you automatically

# Output

```
$ phpcs /myDir/myFile.php  
FILE: /myDir/myFile.php
```

```
-----  
FOUND 3 ERROR(S) AFFECTING 3 LINE(S)  
-----
```

```
 2 | ERROR | Missing file doc comment  
20 | ERROR | PHP keywords must be lowercase;  
    |         | expected "false" but found "FALSE"  
47 | ERROR | Line not indented correctly;  
    |         | expected 4 spaces but found 1  
-----
```

# Testing Code

Sensio Insight

<https://insight.sensiolabs.com/>

- Free for open source
- Integrated with Git
- Symfony-focused, but works with any PHP code
- Checks PHP, XML, YAML, Twig templates, and Composer dependencies

Rule #11-013  
Database queries should use parameter

SQL Injection is possible because of code looking like this:

```
1 $query = 'SELECT * FROM User WHERE username = '.$username.  
password = '.md5($password).'';
```

In such a request, you can assign to `$username` value `toto" OR 1 = 1`. The SQL request will be:

```
1 SELECT * FROM User WHERE username="toto" OR 1 = 1 AND pas  
71dbe52628a3f83a77ab494817525c6"
```

Severity	Category	Developer
3 Critical	13 Architecture	13 John Smith
27 Major	8 Bugrisk	6 Anne Wood
62 Minor	11 Deadcode	3 Larry Fuller
2 Info	1 Security	2 Nancy Jones

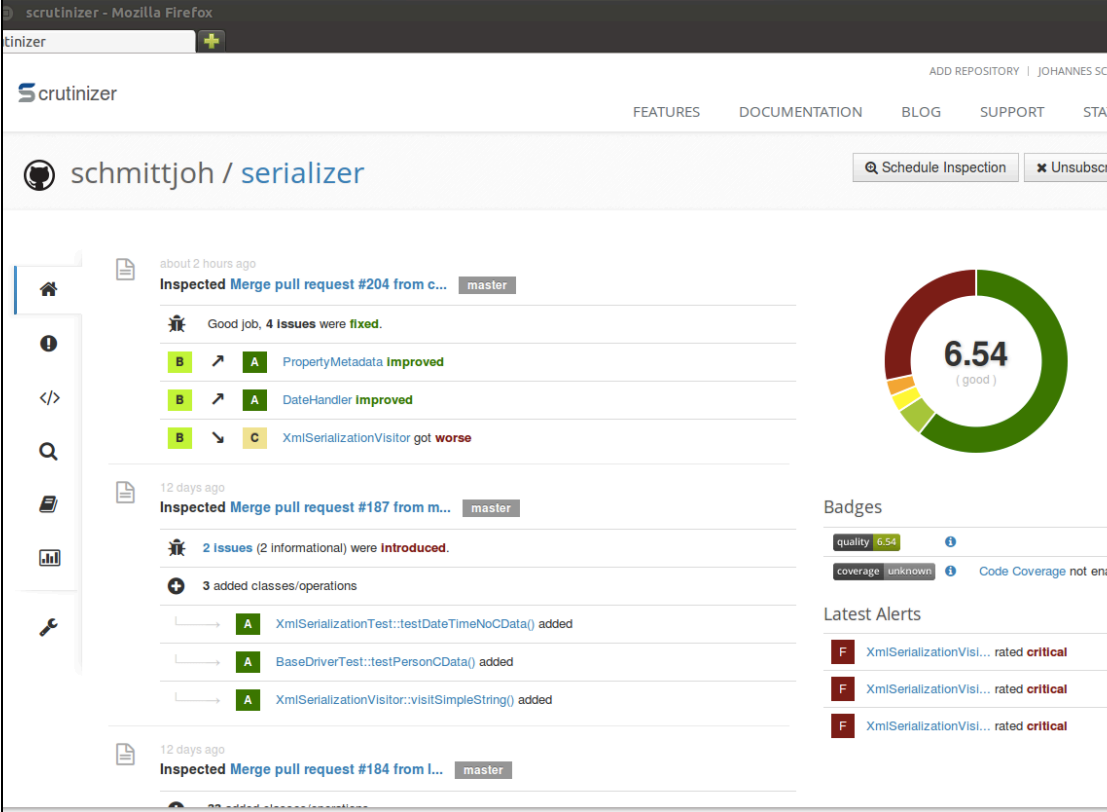


# Testing Code

## Scrutinizer

<https://scrutinizer-ci.com/>

- Free for open source
- Integrated with Git
- Integrates open source checking tools like PHP\_CodeSniffer and PHP Mess Detector
- Checks PHP, Python, and Ruby



The screenshot displays the Scrutinizer web interface for a repository named 'schmittjoh / serializer'. The interface shows a list of inspected merge pull requests. The most recent one, 'Merge pull request #204 from c... master', was inspected 'about 2 hours ago' and resulted in a 'Good job, 4 issues were fixed'. The issues are categorized by severity: B (Warning), A (Info), and C (Error). The quality score for this pull request is 6.54 (good), shown in a circular gauge. The interface also displays 'Badges' for quality (6.54) and coverage (unknown), and 'Latest Alerts' for critical issues.

Issue	Severity	Message
PropertyMetadata	Improved (A)	
DateHandler	Improved (A)	
XmlSerializationVisitor	got worse (C)	

Issue	Severity	Message
XmlSerializationTest::testDateTimeNoCData()	added (A)	
BaseDriverTest::testPersonCData()	added (A)	
XmlSerializationVisitor::visitSimpleString()	added (A)	

# Testing Code

## Code Climate

<https://codeclimate.com/>

- Free for open source
- Integrated with Git
- Quality, security, style, and test coverage checks
- Checks PHP, JavaScript, and Ruby
- Takes security seriously: [codeclimate.com/security](https://codeclimate.com/security)

The screenshot shows the Code Climate interface for the repository 'aws/aws-sdk-php'. The top navigation bar includes 'Home', 'Tour', 'Pricing', 'Sign Up with GitHub', and 'Login'. Below the navigation, there are tabs for 'Feed', 'Code', 'Issues', 'Branches', and 'Trends'. The main content area displays a 'Feed' with several items:

- Summary of January 12th - 18th**: 7 files changed, 176 insertions, 21 deletions.
- Summary of January 5th - 11th**: 28 files changed, 10,291 insertions, 6,178 deletions.
- Quality Change**: A green circle 'A' transitions to a yellow circle 'B', indicating that `src/Aws/Ecs/EcsClient.php` has gotten worse 14 days ago.
- Code Changes**: A plus sign icon indicates that six classes/modules were added 14 days ago. The list includes:
  - `src/Aws/CloudHsm/CloudHsmClient.php` (Grade B)
  - `src/Aws/CloudHsm/Exception/CloudHsmException.php` (Grade A)
  - `src/Aws/CloudHsm/Resources/cloudhsm-2014-05-30.php` (Grade F)
  - `src/Aws/Rds/Resources/rds-2014-10-31.php` (Grade F)
  - `tests/Aws/Tests/CloudHsm/CloudHsmClientTest.php` (Grade A)
  - `tests/Aws/Tests/CloudHsm/Integration/IntegrationTest.php` (Grade A)
- More Changes**: A plus sign icon indicates that five classes/modules were added 16 days ago. The list includes:
  - `src/Aws/Ecs/EcsClient.php` (Grade A)

On the right side, there is a search bar 'Search by name' and a large green circular gauge showing a **3.53 GPA**. Below the gauge, it shows 'code climate 3.5' and 'coverage unknown'. There is also a 'Hotspots' section with a list of files and their grades: `src/Aws/AutoScaling/Auto...` (F), `src/Aws/CloudFormation/...` (F), `src/Aws/CloudWatch/Clou...` (F), `src/Aws/ElastiCache/Elasti...` (F), and `src/Aws/ElasticBeanstalk/...` (F).

# Unit Tests

**Unit** - the smallest piece of testable code within my application or script.

**Unit test** - a piece of code that executes the unit and then evaluates the result returned.

# Tips

- Make sure the unit is small enough so the test is testing a single function.
- Make sure the test is efficient enough to run repeatedly, perhaps even a thousand times a day.
- Make sure the tests do not depend on each other. Each test should be able to run completely separately from other tests.

# Saving Time

```
function validateName($name) {
    if ((strlen($name) > 1) && (strlen($name) < 50)) {
        if (ctype_alpha(str_replace(array(" ", ",", "-", ""), "", $name))) {
            return true;
        }
        else {
            return false;
        }
    }
    else {
        return false;
    }
}

assert(validateName("Beth's Test Name"));
```

# How Many Tests?

Enough to test every basic function of the code.

# Testing Frameworks

- Standardize test format
- Easily run tests
- Analyze results

# PHPUnit

<http://www.phpunit.de>

## Pros:

- Good documentation
- Lots of examples online
- Integrates with many other popular tools and platforms

## Cons:

- Command line only



# SimpleTest

<http://simpletest.sourceforge.net/>

## Pros:

- Run on command line or in browser
- Can test front-end functionality

## Cons:

- Not as integrated as PHPUnit
- No longer in active development

# Selenium-WebDriver

<http://seleniumhq.org/>

## Pros:

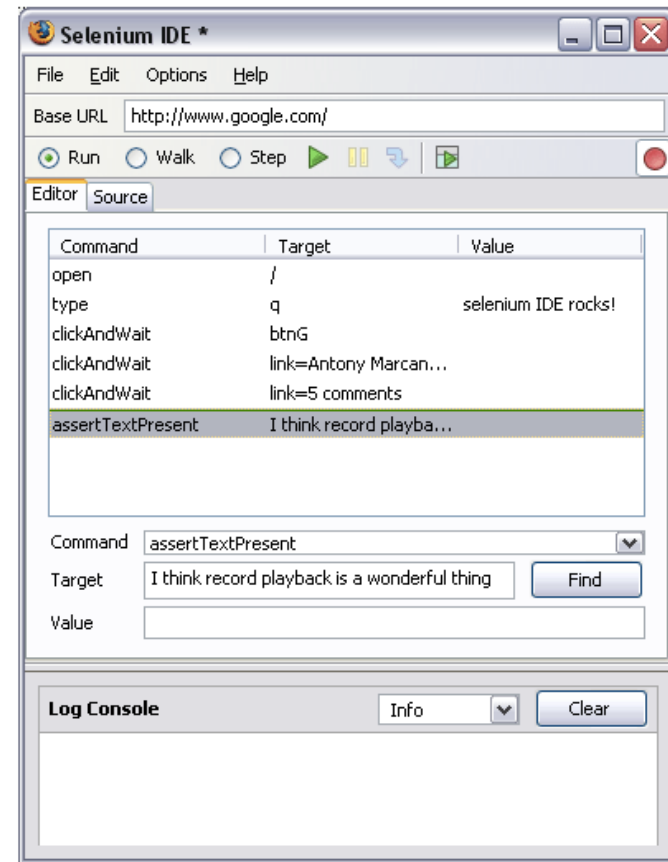
- Can test front-end functionality
- Makes direct calls to the browser using each browser's native support for automation

## Cons:

- Not a native PHP tool, but bindings are available from several third-parties, including one from Facebook
- phpUnit Integration:  
<https://github.com/giorgiosironi/phpunit-selenium>

# Selenium IDE

- Firefox extension
- Record or write scripts by hand



# Automate The Build

Perform a DB query to update the schema, clear a cache, upload files, run cron tasks, etc.

# Phing

<http://phing.info>

- PHP project build system
- Based on Apache Ant
- XML build files and PHP "task" classes
- Integrates with both PHPUnit and SimpleTest as well as phpDocumentor
- Platform independent
- No required external dependencies

# Maven

<http://maven.apache.org>

- Supports Ant tasks
- Large library of third-party plug-ins to integrate other continuous integration tools
- Helps shield you from the details of the build
- For Java-based projects, so you'll need Maven for PHP: <http://www.php-maven.org/>

## Phing Buildfile:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<project name="FooBar" default="dist">
```

```
<!-- ===== -->
```

```
<!-- Target: prepare -->
```

```
<!-- ===== -->
```

```
<target name="prepare">
```

```
  <echo msg="Making directory ./build" />
```

```
  <mkdir dir="./build" />
```

```
</target>
```

```
<!-- ===== -->
```

```
<!-- Target: build -->
```

```
<!-- ===== -->
```

```
<target name="build" depends="prepare">
```

```
  <echo msg="Copying ./about.php to ./build directory..." />
```

```
  <copy file="./about.php" tofile="./build/about.php" />
```

```
</target>
```

```
<!-- ===== -->
```

```
<!-- (DEFAULT) Target: dist -->
```

```
<!-- ===== -->
```

```
<target name="dist" depends="build">
```

```
  <echo msg="Creating archive..." />
```

```
  <tar destfile="./build/build.tar.gz"  
    compression="gzip">
```

```
    <fileset dir="./build">
```

```
      <include name="*" />
```

```
    </fileset>
```

```
</tar>
```

```
  <echo msg="Files copied and compressed in build  
  directory OK!" />
```

```
</target>
```

```
</project>
```

# Documentation

phpDocumentor 2: <http://www.phpdoc.org/>

- Automates documentation
- Tutorial:  
[http://manual.phpdoc.org/HTMLSmartyConverter/HandS/phpDocumentor/tutorial\\_phpDocumentor.howto.pkg.html](http://manual.phpdoc.org/HTMLSmartyConverter/HandS/phpDocumentor/tutorial_phpDocumentor.howto.pkg.html)



```
/**  
 * Put your short description here.  
 *  
 * Put your long description here.  
 * You may use multiple lines.  
 * You can even use Markdown.  
 *  
 * @author Beth Tucker Long <beth@musketeers.me>  
 *  
 * @since 1.0  
 *  
 * @param int $exampleA This is a method parameter description.  
 * @param string $exampleB This is another example.  
 */
```



\ global

## 📍 NAMESPACES

📁 *global*

## 📁 Functions

Creates a packager object with all basic options set.

```
createPackager(string $original_file, string[] $options) : \  
PEAR_Error | \PEAR_PackageFileManager2
```

### Parameters

#### **\$original\_file**

`string` Path of the original package.xml.

#### **\$options**

`string[]` Set of options to merge in.

### Returns

`\PEAR_Error` `\PEAR_PackageFileManager2`

Generate a XHPProf Display View given the various URL parameters as arguments.

```
displayXHPProfReport(object $xhprof_runs_impl, array $url_par  
ams, string $source, string $run, string $wts, string $symbo  
l, $sort, string $run1, string $run2)
```

# CruiseControl

<http://cruisecontrol.sourceforge.net>

- Written in Java
- Binary distribution, a Windows Installer and the source distribution
- Flexible scheduling system
- Notifications via e-mail, messaging or viewing HTML reports
- Integrates with Phing and Maven
- PHPUnderControl - optional add-on application for integrating PHP\_CodeSniffer and PHPUnit

# Jenkins

<http://jenkins-ci.org/>

- Built on Java
- Installed via native packages or a war file
- Easily configured via a GUI web interface
- Extensive library of third-party plug-ins
- RSS, e-mail or instant messaging options for build notifications
- Template for Jenkins Jobs for PHP Projects (by Sebastian Bergmann)

# Reporting

SonarQube

<http://www.sonarqube.org/>

- Integrates with Hudson and Jenkins
- PHP plug-in to integrate it directly with other PHP-based tools
- Web-based application
- Overall “health” of project, drill down for details
- Includes TimeMachine

# Technical Debt

Integrated into core as of version 4.

Assigns a technical debt value

- **The debt ratio** - The debt ratio gives a percentage of the current technical debt of the project versus the total possible debt for the project.
- **The cost to reimburse** – A dollar amount for what it would cost to clean all defects.
- **The work to reimburse** - The cost expressed in work days.

# A Little Help

TeamCity by JetBrains is a user-friendly continuous integration (CI) server for professional developers and build engineers, like ourselves. It is trivial to set up and absolutely free for small teams.

<http://www.jetbrains.com/teamcity/>

# A Little Help

NetBeans has support for continuous integration  
(Template for Jenkins Jobs for PHP Projects)

More info:

[https://blogs.oracle.com/netbeansphp/entry/  
continuous integration support](https://blogs.oracle.com/netbeansphp/entry/continuous_integration_support)



# Yes, But...

- Project is small, budget is small...
- Evaluate which tools are worthwhile to your specific project.

# Make It a Deliverable

Consider including unit tests or code cost/coverage reports in your deliverables to your customers as an added value to them (and you down the road).

# Quick Recap

- Coding Standards -> PHP\_CodeSniffer
- Code Review -> Insight, Scrutinizer, Code Climate
- Unit Tests -> PHPUnit, SimpleTest, Selenium
- Build -> Phing or Maven
- CI Tools -> CruiseControl, Jenkins
- Documentation -> PHP\_Documentor
- Reporting -> SonarQube

# Project

A customer hires you to create a registration form for a one-time event. It's a small customer with a small budget. It should take a couple hundred lines of code in a single file, results will be e-mailed. It will be tested by the event staff and the marketing department on the live site as they do not have a test environment, and it will only be live for two months.

# Ideas

## What they need:

1. If they do not have an in-house standard for you to follow, write it using one of the main coding standards, like PEAR.
2. Create unit tests for it.

## What they don't need:

1. In-depth reporting
2. Full automation, including build.
3. Documentation

# Project

A customer hires you for an ongoing project. On the 15<sup>th</sup> of every month, they need you to go in and add a new survey to collect data and write it to a database. The previous month's survey data needs to be backed up and cleared out of the database when the new survey goes live.

# Ideas

## What they need:

1. If they do not have an in-house standard for you to follow, write it using one of the main coding standards, like PEAR.
2. Create unit tests for it and use a testing framework.
3. Automate the build.

## What they don't need:

1. In-depth reporting (Maybe)
2. Documentation (Maybe)

# Project

A customer hires you to write one part of a very large application. Other consultants that you do not have access to will be working on other parts of the application at the same time.



# Ideas

What they need:

1. All of it

In this situation, see if you can convince them to get everyone working on a unified continuous integration platform utilizing a complete suite of continuous integration tools from standards to documentation and fully automated everywhere in between.

# Take Away #1

Not everything is beneficial enough to use in every situation, so choose the right tools for your project and needs.

## Take Away #2

The fewer steps I have to remember to do manually, the more successful my project will be.

## Resources

CruiseControl - <http://cruisecontrol.sourceforge.net>

Guide to writing your own PHP\_CodeSniffer standards (Official) -

<http://pear.php.net/manual/en/package.php.php-codesniffer.coding-standard-tutorial.php>

Guide to writing your own PHP\_CodeSniffer standards (Alternate) -

<http://luhman.org/blog/2009/12/17/setting-custom-coding-standards-php-codesniffer>

Jenkins - <http://jenkins-ci.org>

Maven - <http://www.php-maven.org>

PEAR coding standard - <http://pear.php.net/manual/en/standards.php>

PEAR Package Manager Installation - <http://pear.php.net/manual/en/installation.php>

PEAR Packages Installation - <http://pear.php.net/manual/en/guide.users.commandline.installing.php>

PEAR2 coding standard - <http://pear.php.net/manual/en/pear2cs.rules.php>

Phing - <http://phing.info>

PHP Standards Working Group - <http://groups.google.com/group/php-standards>

PHP\_CodeSniffer - [http://pear.php.net/package/PHP\\_CodeSniffer](http://pear.php.net/package/PHP_CodeSniffer)

phpDocumentor 2 - <http://www.phpdoc.org/>

PHPUnit - <http://www.phpunit.de/manual/3.6/en/automating-tests.html>

phpUnderControl - <http://phpundercontrol.org>

Selenium - <http://seleniumhq.org/>

SimpleTest - <http://simpletest.sourceforge.net/>

SonarQube – <http://www.sonarqube.org/>

Sonar PHP Plug-in - <http://docs.codehaus.org/display/SONAR/PHP+Plugin>

Sonar Technical Debt Plugin - <http://docs.codehaus.org/display/SONAR/Technical+Debt+Plugin>

Template for Jenkins Jobs for PHP Projects by Sebastian Bergmann - <http://jenkins-php.org>

# Find Me

- Twitter: e3betht
- Madison PHP User Group (Meetup)  
<http://www.madisonphp.com>
- Slides Available on:  
<http://www.TreelineDesign.com/slides>

# Feedback

<https://joind.in/13119>

E-mail:

Beth@CodeClimate.com