

# DevOps Foundations: Continuous Delivery

with James Wickett and Ernest Mueller

---

Would you like to know more? This handout provides definitions of terms used in the course as well as links to the topics, tools, and resources we reference.

## Index

[Introduction](#)

[Chapter 1: Continuous Integration and Continuous Delivery](#)

[Must-Read](#)

[Definitions](#)

[Benefits](#)

[Build Pipelines](#)

[Chapter 2: Build Your Own Pipeline](#)

[Version Control](#)

[SCM Tools](#)

[Continuous Integration](#)

[Continuous Integration Practices](#)

[Notes](#)

[CI Server Tools](#)

[Build Tools](#)

[Artifact Repository](#)

[Artifact Repo Tools](#)

[Testing](#)

[Testing Tools](#)

[Deployment](#)

[Deploy Tools](#)

[Chapter 3: Putting It All Together](#)

[The Continuous Delivery Pipeline](#)

## Introduction

Your fearless instructors are James Wickett ([@wickett](#)) and Ernest Mueller ([@ernestmueller](#)).

The Agile Admin blog

<https://theagileadmin.com/>

Signal Sciences

<https://www.signalsciences.com>

<https://labs.signalsciences.com>

AlienVault

<https://alienvault.com>

<https://otx.alienvault.com/>

## Chapter 1: Continuous Integration and Continuous Delivery

Small + Fast = Better

### Must-Read CI/CD Resources

Continuous Delivery, by Jez Humble and David Farley <https://www.amazon.com/Continuous-Delivery-Deployment-Automation-Addison-Wesley/dp/0321601912>

And in website form

<https://continuousdelivery.com/>

### Definitions

Continuous integration is the practice of automatically building and unit testing an entire application frequently, ideally on every source code check-in— dozens of times a day if necessary.

Continuous delivery is the additional practice of deploying every change to a production-like environment and performing automated integration and acceptance testing after it passes its build and unit tests.

Continuous deployment extends this concept to where every change goes through full automated testing and is deployed automatically to the production environment.

### Benefits

**Continuous delivery has the following benefits.**

- Empowering teams
- Lowered cycle time (shortens lead times for changes, time to market goes down, lower MTTR)
- Better security (and quality increases, not decreases)

- Rhythm of practice (limits your work in progress)
- More time to be productive

## Build Pipelines

**CD pipelines consist of the following components.**

- Source code repository
- Build server
- Unit tests
- Artifact repository
- Deployer
- Integration tests
- End-to-end tests
- Security (and other specialized) tests

### **Types of testing**

- Unit testing
- Code hygiene
- Integration testing
- TDD/BDD/ATDD
- Infrastructure testing
- Performance testing
- Security testing

The difference between TDD, BDD, and ATDD

<http://www.assertselenium.com/atdd/difference-between-tdd-bdd-atdd/>

## Chapter 2: Build Your Own Pipeline

The sample word cloud generator app

<https://github.com/wickett/word-cloud-generator>

### Version Control

Get started on GitHub

<https://github.com/>

Setup SSH key

<https://help.github.com/articles/adding-a-new-ssh-key-to-your-github-account/>

### SCM Tools

Git

<https://git-scm.com/>

On a Mac, `brew install git`

But also...

Subversion

<https://subversion.apache.org/>

GitHub

<https://github.com/>

Bitbucket

<https://bitbucket.org/>

Perforce

<https://www.perforce.com/>

### Continuous Integration

Continuous Integration Practices

#### A CI culture

- Start with a clean environment
- Builds should pass the coffee test (<5 minutes)

- Run tests locally before committing
- Don't commit new code on broken builds
- Don't leave the build broken
- Don't remove failing tests

## Notes

Installing go for yourself

<https://golang.org/doc/install>

Compiling go in Jenkins

[https://golang.org/cmd/go/#hdr-Compile\\_and\\_run\\_Go\\_program](https://golang.org/cmd/go/#hdr-Compile_and_run_Go_program)

<http://www.snowfrog.net/2013/06/18/golang-building-with-makefile-and-jenkins/>

Godep for dependencies

<https://www.goinggo.net/2013/10/manage-dependencies-with-godep.html>

Injecting secrets into Jenkins build jobs

<https://support.cloudbees.com/hc/en-us/articles/203802500-Injecting-Secrets-into-Jenkins-Build-Jobs>

Interesting .gitignore so we can keep the jenkins\_home in Git

<https://github.com/github/gitignore/pull/1763/commits/5263ddbf7e4173462838a3461ba827e2bd2b5635>

Some of our build script is making sure the GOROOT and GOPATH are the weird way go expects them.

<https://stackoverflow.com/questions/37262712/jenkin-build-setup-for-go-projects>

## CI Server Tools

Jenkins

<https://jenkins.io/>

[https://hub.docker.com/\\_/jenkins/](https://hub.docker.com/_/jenkins/)

<https://plugins.jenkins.io/>

<https://jenkins.io/doc/book/pipeline/>

But also...

GoCD

<https://www.go.cd/>

Bamboo

<https://www.atlassian.com/software/bamboo>

TeamCity

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

Travis CI

<https://travis-ci.org/>

CircleCI

<https://circleci.com/>

## Build Tools

Make

<https://www.gnu.org/software/make/>

Rake

<https://github.com/ruby/rake>

Maven

<https://maven.apache.org/>

Gulp

<http://gulpjs.com/>

Packer

<https://www.packer.io/>

fpm

<https://github.com/jordansissel/fpm/wiki>

## Artifact Repository

### Use artifacts for

- Reliability
- Composability
- Security
- Shareability

### Plan ahead

1. Packaging format(s)
2. Dependency management
3. Artifact repo

Nexus docs

<https://books.sonatype.com/nexus-book/reference3/index.html>

Jenkins Nexus Artifact Uploader plugin

<https://wiki.jenkins.io/display/JENKINS/Nexus+Artifact+Uploader>

## Artifact Repo Tools

Nexus

<http://www.sonatype.org/nexus/>

<https://hub.docker.com/r/sonatype/nexus3/>

But also...

Artifactory

<https://www.jfrog.com/artifactory/>

Apache Archiva

<https://archiva.apache.org/index.cgi>

fpm

<https://github.com/jordansissel/fpm>

Bintray

<https://bintray.com/>

Docker Hub

<https://hub.docker.com/>

Amazon S3

<https://aws.amazon.com/s3/>

A roundup

<https://binary-repositories-comparison.github.io/>

## Testing

Unit testing is performed at build time on a single unit of code and/or artifact without use of external dependencies or deployment.

Integration testing is performed as you bring together pieces of your application and as it needs to use external dependencies—databases—to actually do its thing.

End-to-end testing, often implemented as UI testing, is when you test more of your application stack in the way an end user actually does.

Security testing looks for flaws in your code and runtime to prevent compromises and leaking of data in production.

TDD, or test-driven development, is the practice of writing a failing test first, and then writing the code that causes the test to pass, and then refactoring it to make it cleaner.

BDD, or behavior-driven development, is a refinement of TDD that focuses on simple sentence-driven testing.

ATDD, or acceptance test-driven development, extends this to where the project team decides on a set of BDD acceptance tests before development begins.

Metrics to track

- Cycle time
- Velocity
- Customer satisfaction

The 70/20/10 guideline

<https://testing.googleblog.com/2015/04/just-say-no-to-more-end-to-end-tests.html>

The PageObject pattern

<https://martinfowler.com/bliki/PageObject.html>

Running that Abao Docker container from our Jenkins Docker container

<https://stackoverflow.com/questions/44999000/permission-denied-error-invoking-docker-on-mac-host-from-inside-docker-ubuntu-co>

<http://jpetazzo.github.io/2015/09/03/do-not-use-docker-in-docker-for-ci/>

RAML 0.8 spec

<https://github.com/raml-org/raml-spec/blob/master/versions/raml-08/raml-08.md>

RAML tutorials

<http://www.baeldung.com/raml-restful-api-modeling-language-tutorial>

<https://raml.org/developers/raml-100-tutorial>

## Testing Tools

GoConvey

<https://github.com/smartystreets/goconvey>

Abao

<https://github.com/cybertk/abao>



Chai (assert library)

<http://chaijs.com/api/assert/>

RAML

<https://raml.org/>

Robot Framework

<http://robotframework.org/>

<https://github.com/robotframework/Selenium2Library>

Gauntlt

<http://gauntlt.org/>

Retire.js

<http://bekk.github.io/retire.js/>

But also...

JUnit

<http://junit.org/junit4/>

golint

<https://github.com/golang/lint>

gofmt

<https://golang.org/cmd/gofmt/>

Rubocop

<http://batsov.com/rubocop/>

FindBugs

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

Protractor

<http://www.protractortest.org/#/>

Cucumber

<https://cucumber.io/>

Selenium

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

[http://www.seleniumhq.org/docs/03\\_webdriver.jsp](http://www.seleniumhq.org/docs/03_webdriver.jsp)

Sauce Labs

<https://saucelabs.com/>

KitchenCI

<http://kitchen.ci/>

ApacheBench

<https://httpd.apache.org/docs/2.4/programs/ab.html>

JMeter

<http://jmeter.apache.org/>

Mittn

<https://github.com/F-Secure/mittn>

## Deployment

### Deploy with

- The same artifact
- The same way
- The same (similar) environment
- The same smoke tests

Chef recipes

<https://docs.chef.io/recipes.html>

Running chef-solo

[https://docs.chef.io/ctl\\_chef\\_solo.html](https://docs.chef.io/ctl_chef_solo.html)

Getting started with Chef

<http://gettingstartedwithchef.com/first-steps-with-chef.html>

Configuring a Chef resource

<https://learn.chef.io/modules/learn-the-basics/ubuntu/bring-your-own-system/configure-a-resource#/>

## Deploy Tools

Chef

<https://www.chef.io/>

<https://learn.chef.io/#/>

<http://gettingstartedwithchef.com/first-steps-with-chef.html>

But also...

Puppet

<https://puppet.com/>

Rundeck

<http://rundeck.org/>

UrbanCode

<https://developer.ibm.com/urbancode/products/urbancode-deploy/>

ThoughtWorks

<https://www.thoughtworks.com/continuous-delivery>

Deployinator

<https://github.com/etsy/deployinator>

## Chapter 3: Putting It All Together

### The Continuous Delivery Pipeline

#### To successfully perform continuous delivery

1. Only build artifacts once.
  2. Artifacts should be immutable.
  3. Deployment should go to a copy of production before going into production.
  4. Stop deploys if it a previous step fails.
  5. Deployments should be idempotent.
- Each developer is responsible for their check-in through deployment
  - Small changes. Build quality in
  - Don't check in on broken builds
    - Take responsibility for your build
    - Immediately address a broken build
    - Revert if fixing takes time
    - No check-ins while the build is broken— the line stops
  - Automate high-quality testing
    - Run tests before check-in
    - Fix flaky tests
    - Don't ignore or disable tests
  - Automate deployment
  - Keep the build and deploy fast
  - Balance your testing

#### "Crazy Fast Build Times"

<https://www.infoq.com/presentations/Crazy-Fast-Build-Times-or-When-10-Seconds-Starts-to-Make-You-Nervous>

#### The Google Testing blog

<http://testing.googleblog.com>

#### Wikipedia on continuous delivery

[https://en.wikipedia.org/wiki/Continuous\\_delivery](https://en.wikipedia.org/wiki/Continuous_delivery)

#### "Dr. Deming's 14 Points for Management"

<https://www.deming.org/theman/theories/fourteenpoints>

gump·tion

'gəmpSH(ə)n/

noun

informal

1. shrewd or spirited initiative and resourcefulness
2. "She had the gumption to put her foot down and head Dan off from those crazy schemes."

Synonyms: initiative, resourcefulness, enterprise, ingenuity, imagination