SOURCE TO DEPLOYMENT WITH GRADLE & DOCKER

WHO AM I

John Engelman

- Chief Software Technologist
- Ratpack Team Member
- Gradle plugin author and contributor
- DevOps aficionado
- `@johnrengelman`
- github.com/johrengelman

TOPICS

1. Gradle Intro
2. Docker Intro
3. Continuous Integration w/ Gradle
4. Packing An App w/ Docker & Gradle
5. Continuous Deployment w/ Docker

WHAT IS GRADLE?

1. Expressive, declarative, & maintainable build language
2. Dependency Resolver & Manager
3. Build Task Scheduler & Executor
4. Build By Convention

Gradle is an opinionated framework on top of an unopinionated toolkit
- Szczepan Faber

WHAT GRADLE IS NOT!

It is **NOT** Groovy Ant!
(That tool exists -> GANT)

CORE GRADLE FEATURES

1. Build-By-Convention w/ Flexibility
2. Project & Build Groovy DSL
3. Support for Ivy & Maven Dependencies
4. Multi-Project Builds
5. Easy to add custom logic
6. 1st class integration w/ Ant builds
7. Extensive public API and plugin ecosystem
8. Task UP-TO-DATE checking

GRADLE IS FLEXIBLE

Just because it is written in Java

doesn't mean project **must** be Java

```
apply plugin: 'com.moowork.grunt'

npmInstall.inputs.file 'package.json'

installGrunt.dependsOn 'npmInstall'
installGrunt.inputs.file file('Gruntfile.js')

grunt_bowerInstall.dependsOn 'installGrunt'
grunt_bowerInstall.inputs.file file('bower.json')

// Adding dependencies so grunt and NPM get installed
[ grunt_build, grunt_test, grunt_clean ].each{ gtask ->
    gtask.dependsOn 'grunt_bowerInstall'
}

grunt_build.inputs.dir file('app')
grunt_build.outputs.dir file('dist')

assemble.dependsOn grunt_build
build.dependsOn assemble

test.dependsOn grunt_test

task run(description: 'Build and run the application locally.', group
    args = [ 'serve' ]
}

node {
```

```
    // Version of node to use.
    version = '0.12.2'

    // Version of npm to use.
    npmVersion = '2.7.5'

    // Enabled the automatic download. False is the default (for now)
    download = true
}
```

PLUGIN COMPOSITION

Gradle plugins can *react* to other plugins

```
void apply(Project project) {
  project.plugins.withType(JavaPlugin) {
    // Do stuff to Java projects
  }
  project.plugins.withType(GroovyPlugin) {
    // Do extra stuff to Groovy projects
  }
}
```

WHAT IS DOCKER?

- Application Container
- Isolated process
- Portable Packaging

WHY IS THIS GOOD?

- Abstracts application language & build chain
- Write once, run anywhere (*)
- Define OS level dependencies into application package
- Eliminate "works on my machine"
- Increase resource utilization of machines

DOCKER CONTAINERS
VS.
VIRTUAL MACHINES

- Smaller distribution
  - Share the OS
  - Package only the necessities
- Isolate each process
  - Process specific user space, network stack, & filesystem

DOCKER IMAGES & CONTAINERS

DOCKER IMAGE

A docker image is:

- A read-only virtual file system...
- Comprised of layers...
- With atomic changes to the previous (parent) layer...

DOCKER CONTAINER

A docker container is:

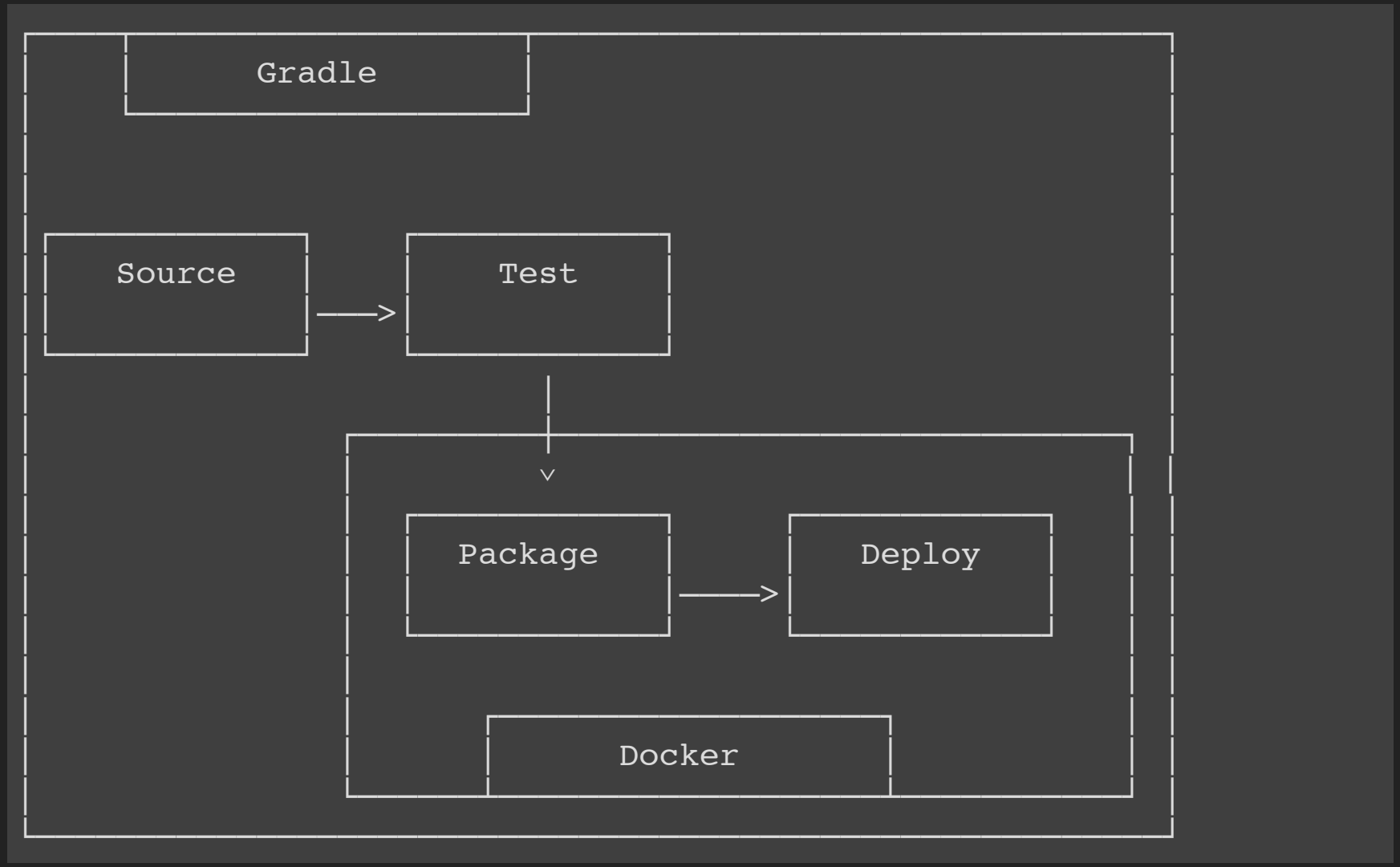
- A process...
- With a read-write layer...
- On top of a image...
- And additional meta-data (env vars, networking config, etc.)

DOCKER & CONTINUOUS DEPLOYMENT

Docker is a useful tool for continuous application deployment.

- Define the entire deployable from the ground up
- On commit, only need to build upwards from the 1st changed layer
- Deploy only the **changed** layers to the server

BUILD PIPELINES


Gradle
Source
Test
Package
Deploy
Docker

KEY COMPONENTS

Gradle Application Plugin

Gradle Docker Plugin

Boot2Docker*

BASIC CONTINUOUS INTEGRATION

```
apply plugin: 'java'

repositories {
  jcenter()
}

dependencies {
  // Add dependencies here
}
```

```
$ gradle build
```

APPLICATION DISTRIBUTIONS

```
apply plugin: 'application'

mainClassName = "sample.MyApplication"
```

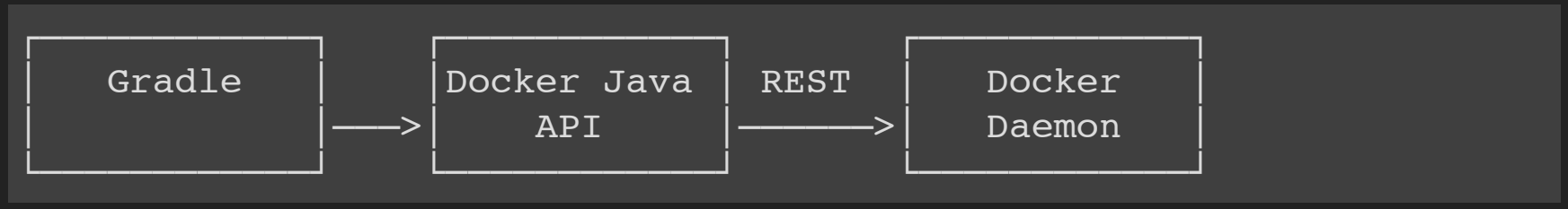
```
$ gradle distTar
```

CONFIGURING GRADLE & BOOT2DOCKER

```
plugins {
  id "com.bmuschko.docker-java-application" version "2.4"
}

docker {
  url = 'https://192.168.59.103:2376'
  certPath = new File(
    System.properties['user.home'],
    '.boot2docker/certs/boot2docker-vm'
  )
}
```

```
$ gradle dockerBuildImage
```


Gradle
Docker Java API
REST
Docker Daemon

DOCKER MACHINE

```
$ docker-machine create \
  --driver amazonec2 \
  --amazonec2-vpc-id <vpc_id> \
  <machine_name>
```

```
$ $(docker-machine env <machine_name>)
```

DEPLOYING DOCKER CONTAINERS

DOCKER SECURITY

CONCERNS

- Docker daemon runs as *root*
- Gradle-Docker integration is unauthenticated