Spring Boot

By Bhagwat Kumar

Agenda

- What and Why?
- Key features of Spring boot
- Prototyping using CLI.
- Gradle primer
- Managing profiles aka environment in grails
- Using Spring data libraries e.g. MongoDB
- Using GORM
- Presentation layer
- Using GSP

What and why?

- Its not a replacement for Spring framework but it presents a small surface area for users to approach and extract value from the rest of Spring.
- Spring-boot provides a quick way to create a Spring based application from dependency management to convention over configuration.
- Grails 3.0 will be based on Spring Boot.

image source : http://spring.io/blog/2013/08/06/spring-boot-simplifying-spring-for-everyone

Key Features

- Stand-alone Spring applications
- No code generation/ No XML Config
- Automatic configuration by creating sensible defaults
- Starter dependencies
- Structure your code as you like
- Supports Gradle and Maven
- Common non-functional requirements for a "real" application
  - embedded servers,
  - security, metrics, health checks
  - externalised configuration

Rapid Prototyping : Spring CLI

- Quickest way to get a spring app off the ground
- Allows you to run groovy scripts without much boilerplate code
- Not recommended for production

Install using GVM

```
$ gvm install springboot
```

Running groovy scripts

```
$ spring run app.groovy
$ spring run --watch app.groovy
$ spring test tests.groovy
```

A quick web application using spring boot

app.groovy

```
@Controller
class Example {
   @RequestMapping("/")
   @ResponseBody
   public String helloWorld() {
      "Hello Spring boot audience!!!"
   }
}
```

```
$ spring run app.groovy
```

What Just happened?

```
// import org.springframework.web.bind.annotation.Controller
// other imports ...
// @Grab("org.springframework.boot:spring-boot-web-starter:0.5.0")

// @EnableAutoConfiguration
@Controller
class Example {
    @RequestMapping("/")
    @ResponseBody
    public String hello() {
        return "Hello World!";
    }

//  public static void main(String[] args) {
//      SpringApplication.run(Example.class, args);
//  }
}
```

Starter POMs

- One-stop-shop for all the Spring and related technology
- A set of convenient dependency descriptors
- Contain a lot of the dependencies that you need to get a project up and running quickly
- All starters follow a similar naming pattern;
  - spring-boot-starter-*
- Examples
  - spring-boot-starter-web
  - spring-boot-starter-data-rest
  - spring-boot-starter-security
  - spring-boot-starter-amqp
  - spring-boot-starter-data-jpa
  - spring-boot-starter-data-elasticsearch
  - spring-boot-starter-data-mongodb
  - spring-boot-starter-actuator

Demo : Starter POMs

```
@Grab('spring-boot-starter-security')
@Grab('spring-boot-starter-actuator')

@Controller
class Example {
    @RequestMapping("/")
    @ResponseBody
    public String helloWorld() {
        return "Hello Audience!!!"
    }
}


//security.user.name : default 'user'
//security.user.password : see log for auto generated password

//actuator endpoints: /beans, /health, /mappings, /metrics etc.
```

Building using Gradle

Lets go beyond prototyping : Gradle

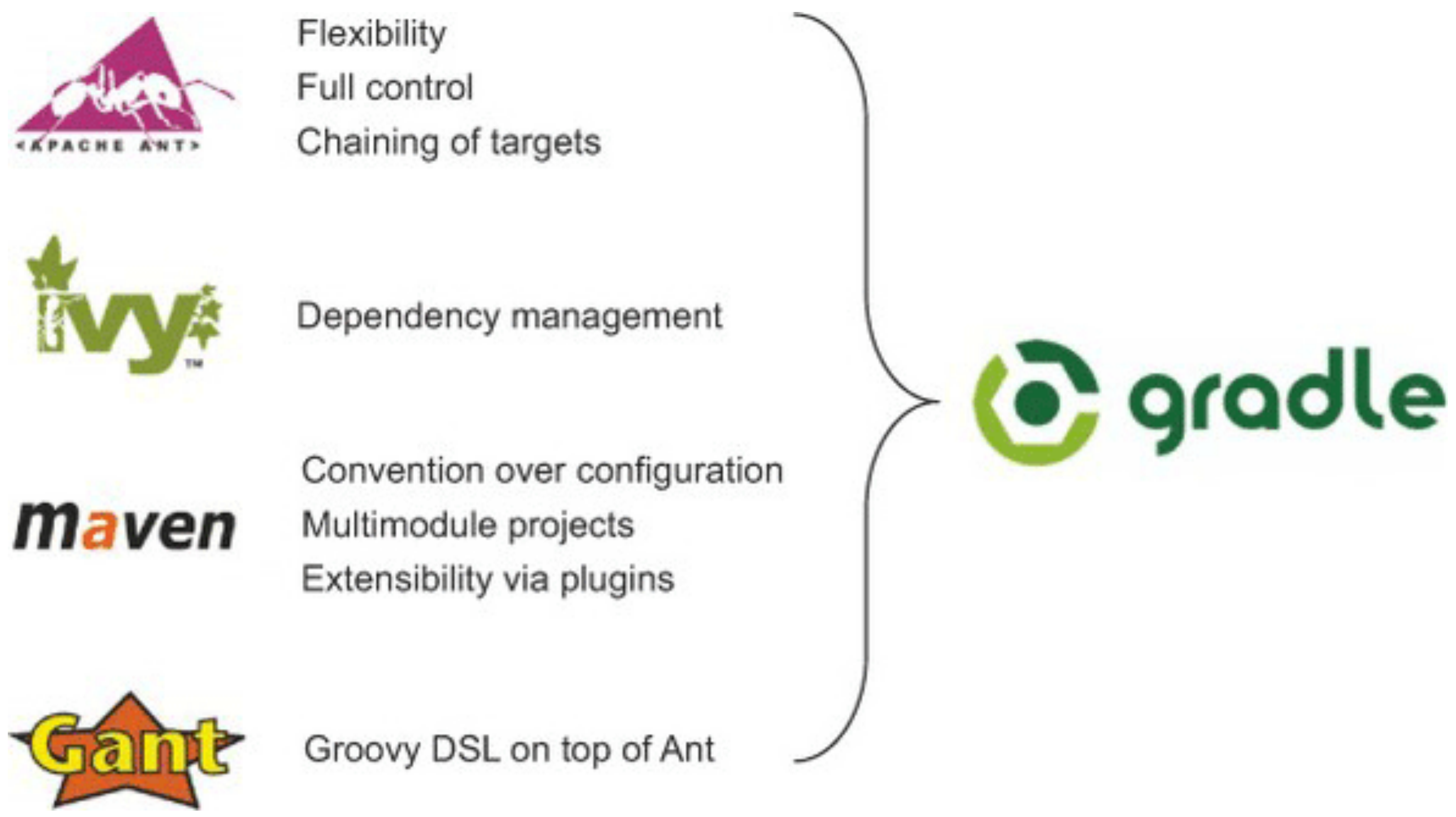


Image source : http://www.drdobbs.com/jvm/why-build-your-java-projects-with-gradle/240168608

build.gradle

```
task hello << {
    println "Hello !!!!"
}
task greet <<{
    println "Welocome Mr. Kumar"
}
task intro(dependsOn: hello) << {
    println "I'm Gradle"
}
hello << { println "Hello extended!!!!" }

greet.dependsOn hello, intro

// gradle tasks   :list all the available tasks
// gradle intro   :executes intro task
// gradle -q greet :bare build output
// gradle --daemon hello  :subsequent execution will be fast
```

build.gradle : using plugin and adding dependencies

```
apply plugin: "groovy"
//look for sources in src/main/groovy folder
//inherits java plugin: src/main/java folder
// tasks compileJava, compileGroovy, build, clean

sourceCompatibility = 1.6

repositories {
    mavenCentral()
}

dependencies {
    compile 'org.codehaus.groovy:groovy-all:2.3.6'
    compile "org.apache.commons:commons-lang3:3.0.1"
    testCompile "junit:unit:4.+"
}
```

build.gradle: for Spring boot app with hot reloading

```
apply plugin: 'groovy'
apply plugin: 'idea'
apply plugin: 'spring-boot'

buildscript {
    repositories { mavenCentral()}
    dependencies {
        classpath("org.springframework.boot:spring-boot-gradle-plugin:1.1.8.RELEASE")
        classpath 'org.springframework:springloaded:1.2.0.RELEASE'
    }
}

idea {
    module {
        inheritOutputDirs = false
        outputDir = file("$buildDir/classes/main/")
    }
}

repositories { mavenCentral() }

dependencies {
    compile 'org.codehaus.groovy:groovy-all'
    compile 'org.springframework.boot:spring-boot-starter-web'
}
```

Environment and Profile aka Grails config

- Put application.properties/application.yml somewhere in classpath
- Easy one: src/main/resources folder

application.yml

```
app:
  name: Springboot+Config+Yml+Demo
  version: 1.0.0
server:
  port: 8080
settings:
  counter: 1
---
spring:
    profiles: development
server:
    port: 9001
```

application.properties

```
app.name=Springboot+Config+Demo
app.version=1.0.0
server.port=8080
settings.coutner=1
```

application-development.properties

```
app.name=Springboot+Config+Demo
app.version=1.0.0
server.port=8080
```

Binding properties

Using ConfigurationProperties annotation

```
import org.springframework.boot.context.properties.ConfigurationProperties
import org.springframework.stereotype.Component
@Component
@ConfigurationProperties(prefix = "app")
class AppInfo {
    String name
    String version
}
```

Using Value annotation

```
import org.springframework.beans.factory.annotation.Value
import org.springframework.stereotype.Component
@Component
class AppConfig {
    @Value('${app.name}')
    String appName

    @Value('${server.port}')
    Integer port
}
```

Examples

OS env variable

```
export SPRING_PROFILES_ACTIVE=development
export SERVER_PORT=8090
gradle bootRun
java -jar build/libs/demo-1.0.0.jar
```

with a -D argument (remember to put it before the main class or jar archive)

```
java -jar -Dspring.profiles.active=development build/libs/dem-1.0.0.jar
java -jar -Dserver.port=8090 build./libs/demo-1.0.0.jar
```

Using Spring data mongoDB

- Add dependency

```
compile 'org.springframework.boot:spring-boot-starter-data-mongodb'
```

- Configure database URL

```
spring.data.mongodb.uri=mongodb://localhost/springtestdev
```

- Add entity class

```
import org.springframework.data.annotation.Id;
class Person{@Id String id, String name}
```

- Add repository interface

```
import org.springframework.data.mongodb.repository.MongoRepository
public interface PersonRepository extends MongoRepository<Person, String> {}
```

- Autowire and use like charm

```
@Autowired PersonRepository  personRepository

personRepository.save(new Person(name:'Spring Boot'))
personRepository.findAll()
personRepository.count()
```

Next level persistence with GORM

- Add dependencies to use GORM-Hibernate

```
compile 'org.springframework.boot:spring-boot-starter-data-jpa'
compile 'org.grails:gorm-hibernate4-spring-boot:1.1.0.RELEASE'
runtime 'com.h2database:h2' //for h2 database
```

- For GORM MongoDB use the following dependencies

```
compile 'org.grails:gorm-mongodb-spring-boot:1.1.0.RELEASE'
```

- Add entity with @grails.persistence.entity

```
import grails.persistence.Entity

@Entity
class Person {
    String name;
    Integer age

    static constraints = {
        name blank: false
        age min: 15
    }
}
```

Further reading https://github.com/grails/grails-data-mapping

Server side view template libraries

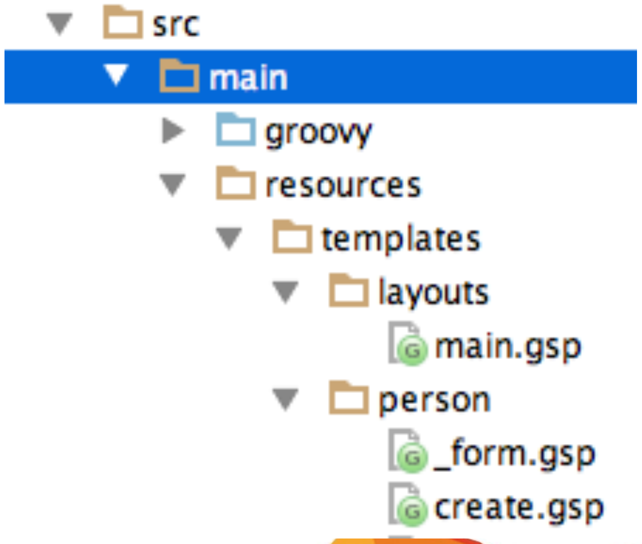
- JSP/JSTL
- Thymeleaf
- Freemarker
- Velocity
- Tiles
- GSP
- Groovy Template Engine

GSP

- Very limited existing tags available
  - https://github.com/grails/grails-boot/issues/3
- Add dependency

```
compile "org.grails:grails-gsp-spring-boot:1.0.0.RC1"
compile "org.grails:grails-web:2.4.0.M2"
```

- Put GSP templates in resources/templates folder



```
src
  main
    groovy
    resources
      templates
        layouts
          main.gsp
        person
          _form.gsp
          create.gsp
```

GSP continued...

- Sample request handler

```
@RequestMapping("/show/{id}")
public ModelAndView show(@PathVariable Long id) {
     Person person = Person.read(id)
     if (person) {
        //render(view:"/person/show", model:[personInstance:personInstance])
         new ModelAndView("/person/show", [personInstance: Person.get(id)])
     } else {
         log.info "No entity fount for id : " + id
         //redirect(controller:"person", action:"list")
         new ModelAndView("redirect:/person/list")
     }
 }
```

Grails Taglib

```
@grails.gsp.TagLib
@org.springframework.stereotype.Component
class ApplicationTagLib {
    static namespace = "app"

    def paginate = { attrs ->
        String action = attrs.action
        Integer total = attrs.total
        Integer currentPage = attrs.currentPage ?: 1
        Integer pages = (total / 10) + 1
        out << render(template: "/shared/pagination",
                model: [action: action, total: total,
                        currentPage: currentPage, pages: pages]
        )
    }
}
```

```
<app:paginate
          total="${personInstanceCount ?: 0}"
          currentPage="${currentPage}"
          action="/person/list"/>
```

Packaging executable jar and war files

Packaging as jar with embedded tomcat

```
$ gradle build
$ java -jar build/libs/mymodule-0.0.1-SNAPSHOT.jar
```

Packaging as war : configure build.groovy

```
//...
apply plugin: 'war'
war {
    baseName = 'myapp'
    version =  '0.5.0'
}
//....
configurations {
    providedRuntime
}

dependencies {
    compile("org.springframework.boot:spring-boot-starter-web")
    providedRuntime("org.springframework.boot:spring-boot-starter-tomcat")
//    ...
}
```

```
$ gradle war
```

Q/A

Thank you.

Blog: http://www.intelligrape.com/blog/author/bhagwat
LinkedIn: http://www.linkedin.com/in/bhagwatkumar
Twitter: http://twitter.com/bhagwatkumar
Mail : bhagwat@intelligrape.com

References

Samples : https://github.com/bhagwat/spring-boot-samples

http://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle
http://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/#getting-started-gvm-cli-installation
https://github.com/spring-projects/spring-boot/tree/master/spring-boot-cli/samples
http://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/#using-boot-starter-poms

http://spring.io/guides/gs/accessing-mongodb-data-rest/
https://spring.io/guides/gs/accessing-data-mongodb/
https://spring.io/guides/gs/accessing-data-jpa/

http://www.gradle.org/

http://www.slideshare.net/Soddino/developing-an-application-with-spring-boot-34661781
http://presos.dsyer.com/decks/spring-boot-intro.html
http://pygments.org/ for nicely formatting code snippets included in presentation