Micro Service Architecture

with Spring Boot, Groovy and Friends

About Me

Marco Vermeulen

- Love Coding!
- Worked for Shazam, Associated Newspapers, Burberry, Visa
- Current: Equal Experts at HMRC
- Creator of GVM (Groovy enVironment Manager)
- Blog: **Wired for Code**
- Twitter: **@marcoVermeulen**

The Talk

- Concepts
  - Micro Service Architecture
  - Spring & Spring Boot
  - Spring Boot Components
- Demo
  - Gradle
  - Cucumber, Spock
  - Spring Boot & Groovy
  - Spring Data & MongoDB

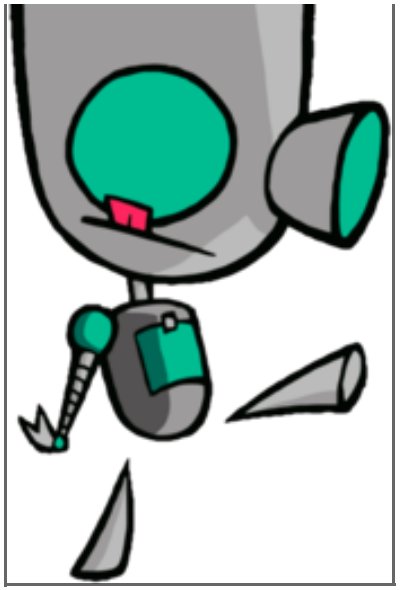
About the Demo

Invader Zim

- American Cartoon
- Created by Jhonen Vasquez
- On Nickelodeon from March, 2001
- Discontinued, with Cult following!
- Theme of IMPENDING DOOM!
- Characters: Zim and GIR

About the Demo

Invader Zim



Zim and GIR

Micro Service Architecture

> *..how we designed and built a Resource Oriented, Event Driven System out of applications about 1000 lines long...*

-- *James Lewis :* ***Java, the Unix Way***

Micro Service Architecture

Small with Single Responsibility

- Many small apps, not monolithic
- Single function
- Few hundred lines of code
- Easy to bin and rewrite!

Micro Service Architecture

Containerless Unix Process

- Embedded Container
- Executable FatJar
- Install with Package Manager (RPM/DEB)
- Use unix service scripts

Micro Service Architecture

Dedicated VCS roots

- Separate Repo per app
- Okay to duplicate domains!
- Common modules can be extracted

Micro Service Architecture

Status Aware and Auto-Scaling

- In-app Metrics
- Ping and Health Checks
- External watchdog process
- Scale on demand

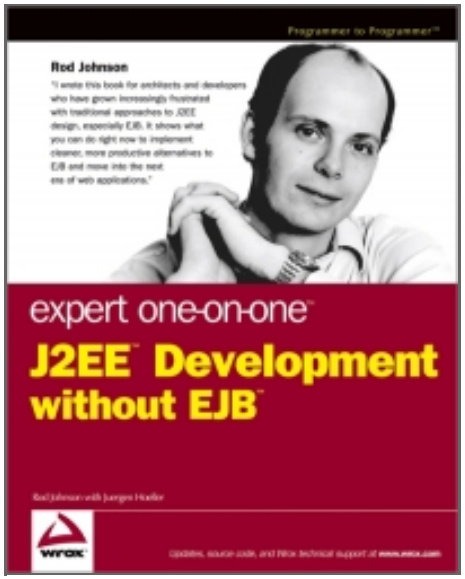
Frameworks

Lots of choices!

- Sinatra
- Play
- Django
- Drop Wizard
- Vertx
- Grails
- Ratpack
- Spring Boot

Spring Ecosystem

In the beginning...



J2EE Development without EJB

Spring Data

Defined

> *the umbrella project which aims to provide a familiar and consistent Spring-based programming model for new datastores while retaining store-specific features and capabilities.*

Spring Data

Command Pattern

- NoSQL datastores (Mongo, Redis, Neo4J, Hadoop)
- Relational stores (JPA & JDBC)
- Umbrella Project has Sub-projects (Official & Community)
- Unified Interface

Spring Data

Spring Data for MongoDB

- Java Config with `AbstractMongoConfiguration`
- `MongoRepository` interface
- Exceptions translate to `DataAccessException`
- Object Mapping
- Much more...

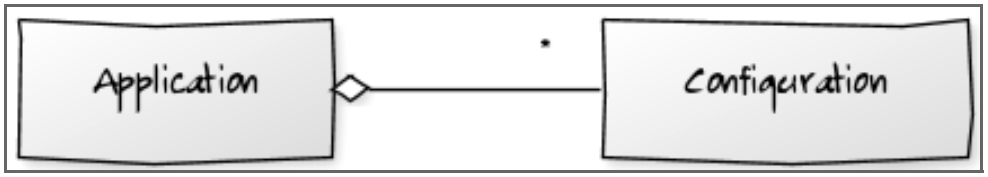
Spring Boot

over Spring

- Opinionated
- Automatic config
- Standalone apps
- Embedded container
- Starter + Example builds
- Metrics
- No XML!
- Groovy!

Spring Boot

Configuration Components



Spring Boot

Components

Application

```
@EnableAutoConfiguration
@ComponentScan("zim")
class Application {
    static void main(String[] args){
        new SpringApplication(Application).run(args)
    }
}
```

Spring Boot

Components

Configuration

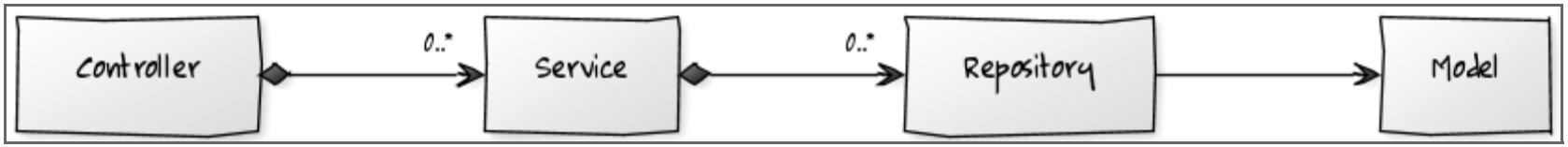
```
@Configuration
class MongoConfiguration extends AbstractMongoConfiguration {

    String getDatabaseName() {
        "invasion"
    }

    Mongo mongo() throws Exception {
        new MongoClient()
    }
}
```

Spring Boot

Component Stereotypes and Domain



Spring Boot

Stereotypes

Controller

```
@Controller
class InvasionController {

    @Autowired
    QuoteRepository repository

    @RequestMapping("/invader/{name}")
    @ResponseBody ResponseEntity quote(@PathVariable String name){
        def quotes = repository.findByName(name)
        if(!quotes) throw new InvaderNotFoundException(name)

        def quote = quotes[(int)(Math.random() * quotes.size())]
        new ResponseEntity(quote, OK)
    }
}
```

context: `/invader/zim`

Spring Boot

Stereotypes

Service

```
@Service
class SomeService {

    Stuff prepare(String stuffing){
        //do stuff
        stuff
    }
}
```

Spring Boot

Stereotypes

Repository

```
@Repository
interface QuoteRepository extends MongoRepository<Quote, BigInteger> {
    List<Quote> findByName(String name)
}
```

Spring Boot

Domain

Model

```
@Document
class Stuffing {
    @Id BigInteger id
    @Field String name
    @Field String description
}
```

Spring Boot

Build

```
buildscript {
    repositories {
        url "http://repo.spring.io/libs-release"
    }
    dependencies {
        classpath "org.springframework.boot:spring-boot-gradle-plugin:1.1.5.RELEASE"
    }
}
apply plugin: 'spring-boot'
apply plugin: 'groovy'
repositories {
    url 'http://repo.spring.io/release'
}
dependencies {
    compile "org.codehaus.groovy:groovy:2.3.7"
    compile "org.springframework.boot:spring-boot-starter-web"
    compile "org.springframework.boot:spring-boot-starter-actuator"
    compile "org.springframework.boot:spring-boot-starter-remote-shell"
    compile "org.springframework.boot:spring-boot-starter-data-mongodb"
}
```

`build.gradle`

Invader Zim and GIR

Quote Service

Demo

Invader Zim (and GIR)

Quote Service

```
curl -s http://localhost:8080/invader/GIR
```

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Sun, 23 Mar 2014 14:17:44 GMT

{
    "id":257235599002375564072234583 22,
    "name":"GIR",
    "message":"Can I be a mongoose dog?"
}
```

Conclusion

Micro Services:

- *evolved from Monoliths*
- *are small*
- *have single responsibility*
- *are dispensible*
- *are self contained*
- *have embedded containers*
- *are self aware*
- **MUST LOOK UP INVADER ZIM!**

Thank You!!!

Q & A