



Pivotal

Cloud Native Workshop

Spring Boot and Actuator

Pivotal Cloud Foundry

Pivotal.

Agenda

1. Challenges building non-Boot applications
2. What is Spring Boot?
3. Capabilities

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Agenda

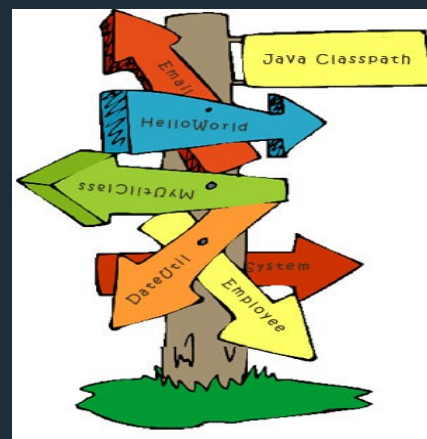
1. Challenges building non-Boot applications
2. What is Spring Boot?
3. Capabilities

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Classpath Hell

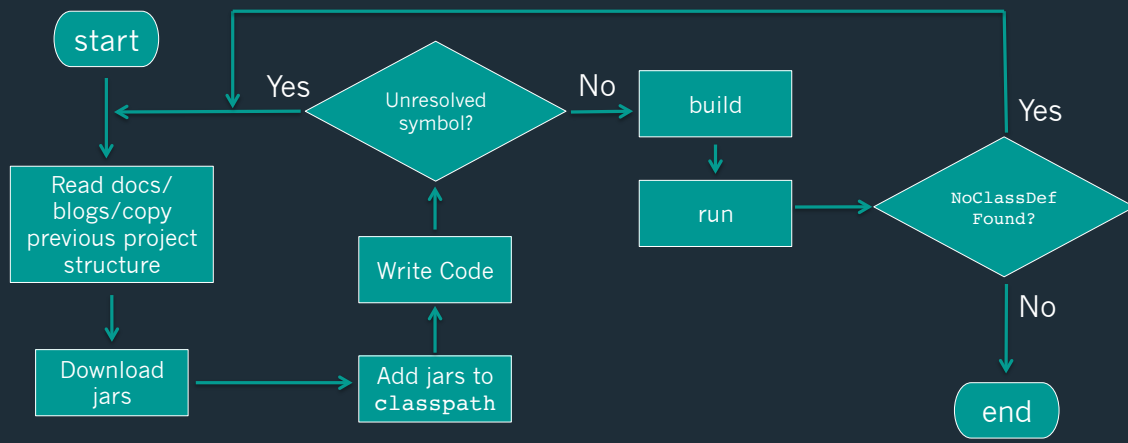
- A Jar is missing
- There is one Jar too many
- A class is not visible where it should be



Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Resolving Classpath Hell



Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Boilerplate code

Every application that accesses a relational database with JDBC needs to configure a `JdbcTemplate` and a `DataSource`

```
@Bean
public JdbcTemplate jdbcTemplate(
    DataSource dataSource) {
    return new JdbcTemplate(dataSource);
}
```

```
@Bean
public DataSource dataSource() {
    return new EmbeddedDatabaseBuilder()
        .setType(EmbeddedDatabaseType.H2)
        .addScripts("schema.sql", "data.sql")
        .build();
}
```

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Boilerplate code

To get a db connection

- Set Driver
- Set Connection URL
- Set Username
- Set Password
- Catch Exceptions

```
public static Connection getConnection() {
    if (dbConnection != null) {
        return dbConnection;
    } else {
        try {
            InputStream inputStream =
                DbUtil.class.getClassLoader().
                    getResourceAsStream("db.properties");
            Properties properties = new Properties();
            if (properties != null) {
                properties.load(inputStream);

                String dbDriver = properties.getProperty("dbDriver");
                String connectionUrl = properties.getProperty("connectionUrl");
                String userName = properties.getProperty("userName");
                String password = properties.getProperty("password");

                Class.forName(dbDriver).newInstance();
                dbConnection = DriverManager.
                    getConnection(connectionUrl, userName, password);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return dbConnection;
    }
}
```

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Boilerplate code

For CRUD Operations

- Prepare Statement
- Set Parameters
- Execute Statement
- Catch Exceptions

```
public void save(String userName, String password, String firstName,
    String lastName, String dateOfBirth, String emailAddress) {
    try {
        PreparedStatement prepStatement = dbConnection.prepareStatement(
            "insert into student(userName, password, " +
            "firstName, lastName, dateOfBirth, emailAddress) " +
            "values (?, ?, ?, ?, ?, ?)");
        prepStatement.setString(1, userName);
        prepStatement.setString(2, password);
        prepStatement.setString(3, firstName);
        prepStatement.setString(4, lastName);
        prepStatement.setDate(5, new java.sql.Date(
            new SimpleDateFormat("MM/dd/yyyy").
                parse(dateOfBirth.substring(0, 10)).getTime()));
        prepStatement.setString(6, emailAddress);

        prepStatement.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    } catch (ParseException e) {
        e.printStackTrace();
    }
}
```

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Agenda

1. Challenges building non-Boot applications
2. What is Spring Boot?
3. Capabilities

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

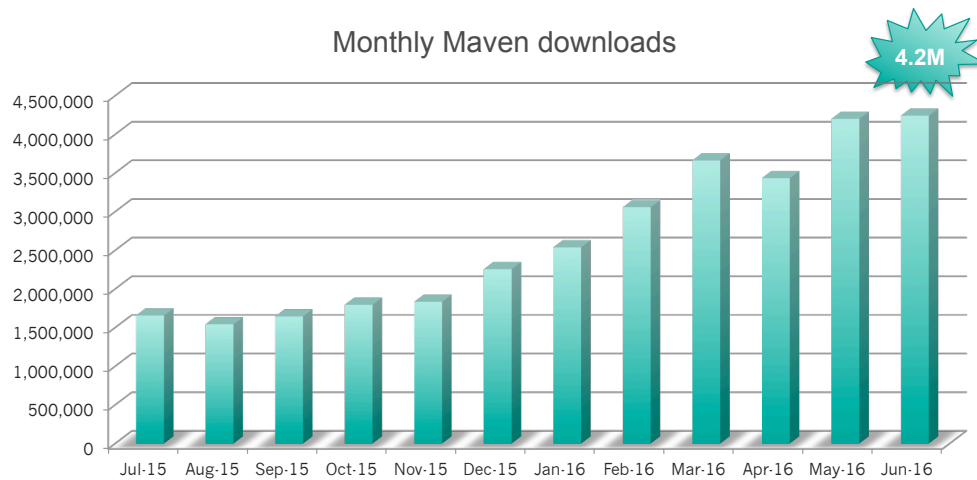
Spring Boot

Spring Boot is an opinionated framework to
simplify bootstrapping and development of new
Spring Applications

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Spring Boot Adoption



Source: oss.sonatype.org

Agenda

1. Challenges building non-Boot applications
2. What is Spring Boot?
3. Capabilities

Capabilities

- Quick start project generation
- Automatic project dependency management
- Configuration drift prevention
- Conditional configuration

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Capabilities

- Developer Productivity Tooling
- Auto-configuration
- Monitoring and management endpoints
- Microservices-friendliness

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Spring Initializr

(Quick start project generation)

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Spring Initializer

Spring Initializr is a configurable service to consistently and easily generate a quick start project

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Spring Initializer

- Generates a Spring Boot project structure
- Provides a Maven/Gradle build specification
- Doesn't generate application code
- You can customize the Spring Initializr
 - <https://github.com/spring-io/initializr/>

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Spring Boot starters

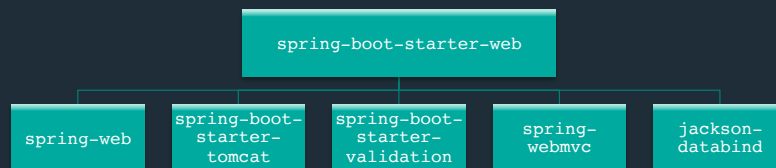
(Automatic project dependency management)

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Spring Boot starters

- Are virtual packages deployed to Maven central
- They pull in other dependencies while containing no code of their own



Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Profiles

(Conditional configuration)

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Profiles

Segregate parts of the application configuration and make it available in certain environments via

- Annotations
- Properties file

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Profiles

Mark `@Component` or `@Configuration` with `@Profile` to limit when it is loaded.

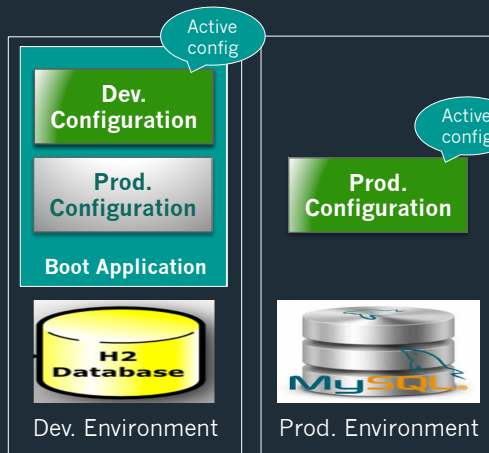
```
@Configuration
@Profile("development")
public class DevelopmentConfiguration {
    // The @Profile requires that the "development"
    // profile be active at runtime
    // for this configuration to be applied
    // Activate this profile in the properties
    // file with spring.profile.active=development
    // command line with --spring.profiles.active=development
}
```

```
@Configuration
@Profile("production")
public class ProductionConfiguration {
    // The @Profile requires that the "production"
    // profile be active at runtime
    // for this configuration to be applied
    // Activate this profile in the properties
    // file with spring.profile.active=production
    // command line with --spring.profiles.active=production
}
```

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Profiles



Checkout code from source control

Build an executable jar

Configuration activated based on active profile

Deploy anywhere

No Code Changes

No Configuration Changes



Pivotal

© Copyright 2015 Pivotal. All rights reserved.

Precedence of externalized configuration

1. Command line arguments.
2. Properties from `SPRING_APPLICATION_JSON` (inline JSON embedded in an environment variable or system property)
3. JNDI attributes from `java:comp/env`.
4. Java System properties (`System.getProperties()`).
5. OS environment variables.
6. A `RandomValuePropertySource` that only has properties in `random.*`.
7. Profile-specific application properties outside of your packaged jar (`application-{profile}.properties` and YAML variants)
8. Profile-specific application properties packaged inside your jar (`application-{profile}.properties` and YAML variants)
9. Application properties outside of your packaged jar (`application.properties` and YAML variants).
10. Application properties packaged inside your jar (`application.properties` and YAML variants).
11. `@PropertySource` annotations on your `@Configuration` classes.
12. Default properties (specified using `SpringApplication.setDefaultProperties`).

Pivotal

© Copyright 2015 Pivotal. All rights reserved.

Actuator

(Monitoring & Management Endpoints)

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Actuator

Offers production-ready features such as monitoring and metrics to improve operator efficiency

```
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-actuator</artifactId>  
</dependency>
```

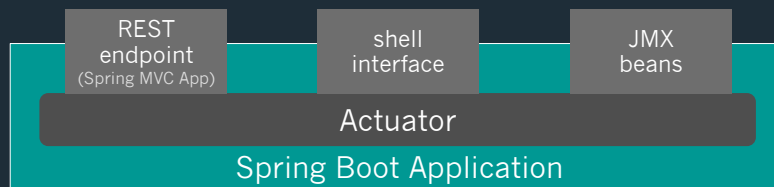
Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Actuator

Actuator Services are exposed through

- HTTP endpoints
- shell interface
- JMX Beans



Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Types of Actuator Endpoints

Configuration
/beans
/autoconfig
/env
/configprops
/controller

Metrics
/metrics
/trace
/dump

Miscellaneous
/shutdown
/info

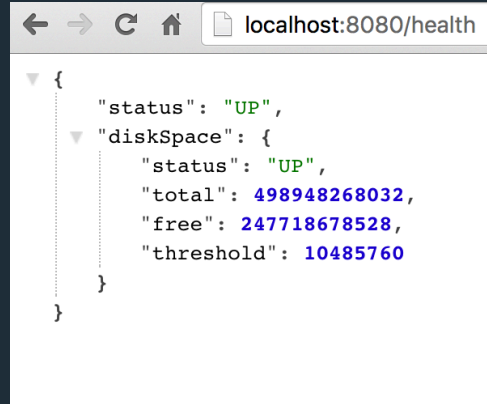
Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

/health

Reports health metrics for the application

- Each check returns a Health object
 - status
 - UP
 - DOWN
 - UNKNOWN
 - OUT_OF_SERVICE
- Plugin your own health definitions



A screenshot of a web browser window displaying the response from the `localhost:8080/health` endpoint. The response is a JSON object with a `status` field set to `"UP"` and a `diskSpace` field containing a nested object. The nested object has `status` set to `"UP"`, `total` set to `498948268032`, `free` set to `247718678528`, and `threshold` set to `10485760`.

```
{
  "status": "UP",
  "diskSpace": {
    "status": "UP",
    "total": 498948268032,
    "free": 247718678528,
    "threshold": 10485760
  }
}
```

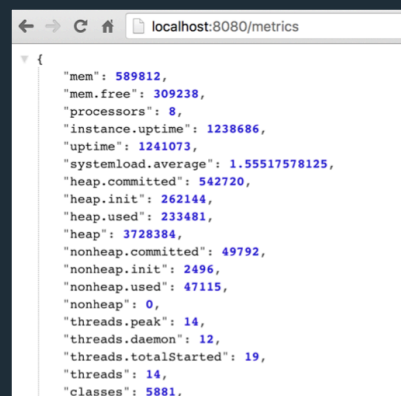
Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

/metrics

Reports application metrics such as

- Memory usage and
- HTTP request counters



A screenshot of a web browser window displaying the response from the `localhost:8080/metrics` endpoint. The response is a JSON object containing various application metrics such as memory usage, system load, heap memory, and thread counts.

```
{
  "mem": 589812,
  "mem.free": 309238,
  "processors": 8,
  "instance.uptime": 1238686,
  "uptime": 1241073,
  "systemload.average": 1.55517578125,
  "heap.committed": 542720,
  "heap.init": 262144,
  "heap.used": 233481,
  "heap": 3728384,
  "nonheap.committed": 49792,
  "nonheap.init": 2496,
  "nonheap.used": 47115,
  "nonheap": 0,
  "threads.peak": 14,
  "threads.daemon": 12,
  "threads.totalStarted": 19,
  "threads": 14,
  "classes": 5881,
}
```

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Microservices

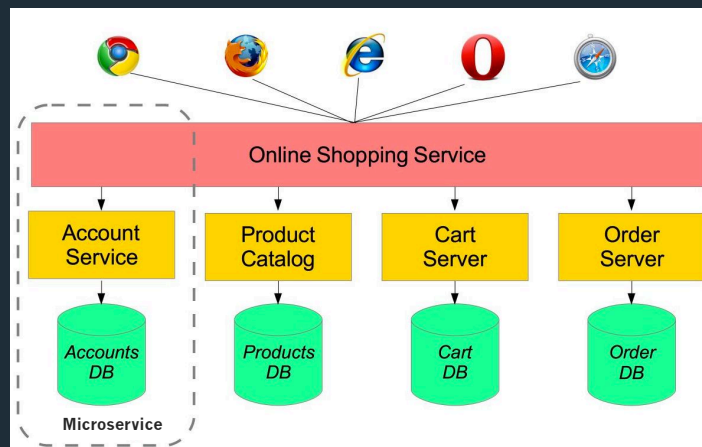
(Microservices-friendliness)

Pivotal.

© Copyright 2015 Pivotal. All rights reserved.

Microservices

Each microservice is a
Spring Boot fat jar



Pivotal.

© Copyright 2015 Pivotal. All rights reserved.



Thank You

