# Spring REST Controller - Dev Process

# Development Process

Step-By-Step

# Development Process

Step-By-Step

1. Add Maven dependency for Spring MVC and Jackson project

# Development Process

Step-By-Step

1. Add Maven dependency for Spring MVC and Jackson project

2. Add code for All Java Config: @Configuration

# Development Process

Step-By-Step

1. Add Maven dependency for Spring MVC and Jackson project

2. Add code for All Java Config: @Configuration

3. Add code for All Java Config: Servlet Initializer

# Development Process

Step-By-Step

1. Add Maven dependency for Spring MVC and Jackson project

2. Add code for All Java Config: @Configuration

3. Add code for All Java Config: Servlet Initializer

4. Create Spring REST Service using @RestController

# Step 1: Add Maven Dependencies

**File: pom.xml**

# Step 1: Add Maven Dependencies

**File: pom.xml**

```
<!-- Add Spring MVC and REST support -->
<dependency>
   <groupId>org.springframework</groupId>
   <artifactId>spring-webmvc</artifactId>
   <version>...</version>
</dependency>
```

**Will load all supporting dependencies:**
**spring-core, logging etc …**

# Step 1: Add Maven Dependencies

**File: pom.xml**

```
<!-- Add Spring MVC and REST support -->
<dependency>
   <groupId>org.springframework</groupId>
   <artifactId>spring-webmvc</artifactId>
   <version>...</version>
</dependency>

<!-- Add Jackson for JSON converters -->
<dependency>
   <groupId>com.fasterxml.jackson.core</groupId>
   <artifactId>jackson-databind</artifactId>
   <version>...</version>
</dependency>
```

**Add Jackson support**
**we'll need it later for converting**
**JSON <--> POJO**

# Step 1: Add Maven Dependencies

**File: pom.xml**

```
<!-- Add Spring MVC and REST support -->
<dependency>
   <groupId>org.springframework</groupId>
   <artifactId>spring-webmvc</artifactId>
   <version>...</version>
</dependency>

<!-- Add Jackson for JSON converters -->
<dependency>
   <groupId>com.fasterxml.jackson.core</groupId>
   <artifactId>jackson-databind</artifactId>
   <version>...</version>
</dependency>

<!-- Add Servlet support for
    Spring's AbstractAnnotationConfigDispatcherServletInitializer -->
<dependency>
   <groupId>javax.servlet</groupId>
   <artifactId>javax.servlet-api</artifactId>
   <version>...</version>
</dependency>
```

**Add Servlet support**

# Step 2: All Java Config: @Configuration

**File: DemoAppConfig.java**

```
@Configuration
@EnableWebMvc
@ComponentScan(basePackages="com.luv2code.springdemo")
public class DemoAppConfig {

}
```

# Web App Initializer

# Web App Initializer

- Spring MVC provides support for web app initialization

# Web App Initializer

- Spring MVC provides support for web app initialization
- Makes sure your code is automatically detected

# Web App Initializer

- Spring MVC provides support for web app initialization
- Makes sure your code is automatically detected
- Your code is used to initialize the servlet container

# Web App Initializer

- Spring MVC provides support for web app initialization
- Makes sure your code is automatically detected
- Your code is used to initialize the servlet container

**AbstractAnnotationConfigDispatcherServletInitializer**

# Web App Initializer (more info)

**AbstractAnnotationConfigDispatcherServletInitializer**

# Web App Initializer (more info)

**AbstractAnnotationConfigDispatcherServletInitializer**

- Your  TO DO list

# Web App Initializer (more info)

**AbstractAnnotationConfigDispatcherServletInitializer**

- Your  TO DO list
  - Extend this abstract base class

# Web App Initializer (more info)

**AbstractAnnotationConfigDispatcherServletInitializer**

- Your TO DO list
  - Extend this abstract base class
  - Override required methods

# Web App Initializer (more info)

**AbstractAnnotationConfigDispatcherServletInitializer**

- Your TO DO list
  - Extend this abstract base class
  - Override required methods
  - Specify servlet mapping and location of your app config

# Step 3: All Java Config: Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {









}
```

# Step 3: All Java Config: Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

   @Override
   protected Class<?>[] getRootConfigClasses() {
      // TODO Auto-generated method stub
      return null;
   }









}
```

# Step 3: All Java Config: Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

   @Override
   protected Class<?>[] getRootConfigClasses() {
      // TODO Auto-generated method stub
      return null;
   }

   @Override
   protected Class<?>[] getServletConfigClasses() {
      return new Class[] { DemoAppConfig.class };
   }





}
```

# Step 3: All Java Config: Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

   @Override
   protected Class<?>[] getRootConfigClasses() {
      // TODO Auto-generated method stub
      return null;
   }

   @Override
   protected Class<?>[] getServletConfigClasses() {
      return new Class[] { DemoAppConfig.class };
   }



}
```

Our config class from Step 2

# Step 3: All Java Config: Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

   @Override
   protected Class<?>[] getRootConfigClasses() {
      // TODO Auto-generated method stub
      return null;
   }

   @Override
   protected Class<?>[] getServletConfigClasses() {
      return new Class[] { DemoAppConfig.class };
   }





}
```

# Step 3: All Java Config: Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

   @Override
   protected Class<?>[] getRootConfigClasses() {
      // TODO Auto-generated method stub
      return null;
   }

   @Override
   protected Class<?>[] getServletConfigClasses() {
      return new Class[] { DemoAppConfig.class };
   }

   @Override
   protected String[] getServletMappings() {
      return new String[] { "/" };
   }

}
```

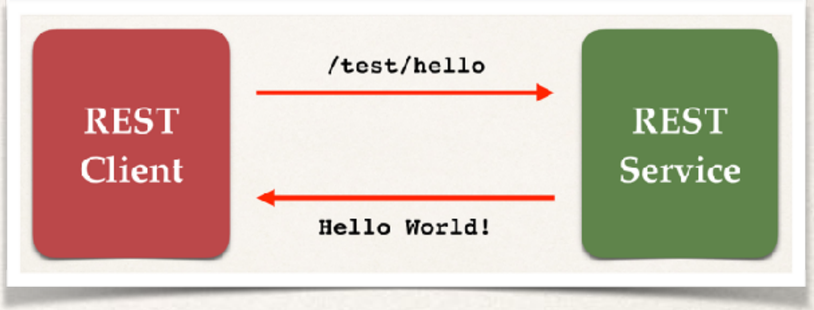
Whew!!!!

Now for the easy stuff

Create Spring REST Controller

easy peazy ...

# Step 4: Create Spring REST Service



# Step 4: Create Spring REST Service

```
@RestController
@RequestMapping("/test")
public class DemoRestController {

  @GetMapping("/hello")
  public String sayHello() {
    return "Hello World!";
  }

}
```