Maven - Springboot PipelineSample

Prerequisite

- JDK 1.7+
- Maven 3+

Stack

- Java
- Spring Boot

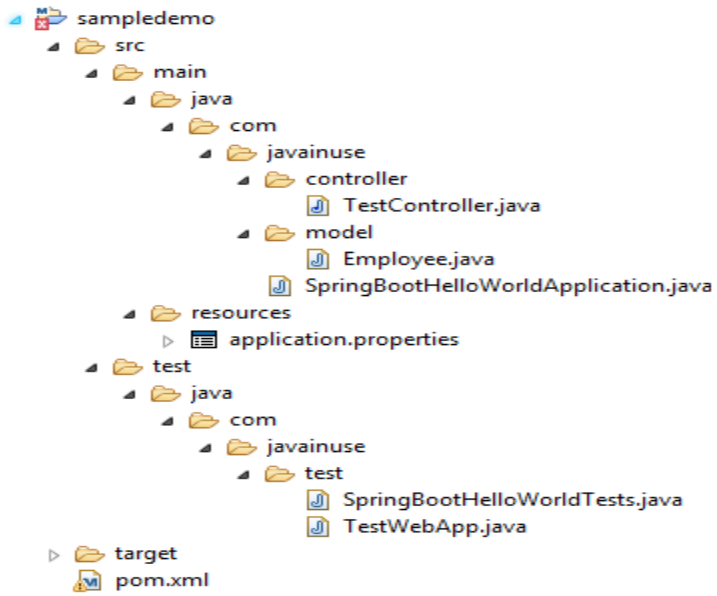
Projects

Create the below services (projects) in maven

1. Sampledemo

2. PipelineSample

Project Structure



Pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.springboot</groupId>
  <artifactId>sampledemo</artifactId>
   <version>1.0</version>
  <packaging>war</packaging>
      <parent>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-parent</artifactId>
            <version>1.4.1.RELEASE</version>
            <relativePath /> <!-- lookup parent from repository -->
      </parent>

      <properties>
            <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
            <project.reporting.outputEncoding>UTF-
8</project.reporting.outputEncoding>
            <java.version>1.8</java.version>
      </properties>

      <dependencies>
            <dependency>
                  <groupId>org.springframework.boot</groupId>
                  <artifactId>spring-boot-starter-web</artifactId>
            </dependency>

<!-- dependency for unit test-- >


            <dependency>
                  <groupId>org.springframework.boot</groupId>
                  <artifactId>spring-boot-starter-test</artifactId>
                  <scope>test</scope>
            </dependency>

      </dependencies>

      <build>
     <plugins>
      <!-- Plugin for unit report generation-- >

                  <plugin>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-maven-plugin</artifactId>
                  </plugin>
                  <plugin>
          <groupId>org.apache.maven.plugins</groupId>
          <artifactId>maven-surefire-plugin</artifactId>
          <version>2.19.1</version>
        </plugin>
```

```
    <!-- Plugin for Code Coverage (cobertura )report generation-- >

       <plugin>
            <groupId>org.codehaus.mojo</groupId>
            <artifactId>cobertura-maven-plugin</artifactId>
            <version>2.5.1</version>
            <configuration>
                <formats>
                    <format>html</format>
                    <format>xml</format>
                </formats>
            </configuration>
         </plugin>


       </plugins>
       </build>
              <reporting>
               <plugins>
               <plugin>
               <groupId>org.apache.maven.plugins</groupId>
               <artifactId>maven-project-info-reports-plugin</artifactId>
               <version>2.7</version>
               <configuration>

<dependencyLocationsEnabled>false</dependencyLocationsEnabled>
               </configuration>

        </plugin>


               </plugins>
               </reporting>

</project>
```

TestController.java
Path → src/main/java/com/javainuse/controller/ TestController.java
package com.javainuse.controller; import org.springframework.web.bind.annotation.RequestMapping; import org.springframework.web.bind.annotation.RequestMethod; import org.springframework.web.bind.annotation.RestController; import com.javainuse.model.Employee; @RestController

```
public class TestController {

	@RequestMapping(value = "/employee", method = RequestMethod.GET)
	public Employee firstPage() {

		Employee emp = new Employee();
		emp.setName("emp1");
		emp.setDesignation("manager");
		emp.setEmpId("1");
		emp.setSalary(3000);

		return emp;
	}

}
```

Employee.java

Path → src/main/java/com/javainuse/model/ Employee.java

```
package com.javainuse.model;

public class Employee {
	private String empId;
	private String name;
	private String designation;
	private double salary;

	public Employee() {
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public String getDesignation() {
		return designation;
	}

	public void setDesignation(String designation) {
		this.designation = designation;
	}

	public double getSalary() {
		return salary;
	}
```

```
	public void setSalary(double salary) {
		this.salary = salary;
	}

	public String getEmpId() {
		return empId;
	}

	public void setEmpId(String empId) {
		this.empId = empId;
	}

}
```

SpringBootHelloWorldApplication.java

Path → src/main/java/com/javainuse/ SpringBootHelloWorldApplication.java

```
package com.javainuse;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class SpringBootHelloWorldApplication {

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

application.properties

Path → src/main/resources/application.properties

```
server.port = 8666
```

SpringBootHelloWorldTests.java

Path → src/test/java/com/javainuse/test/ SpringBootHelloWorldTests.java

```
package com.javainuse.test;
```

```
import org.junit.Test;
import org.junit.runner.RunWith;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.test.context.junit4.SpringRunner;

@RunWith(SpringRunner.class)
@SpringBootTest
public class SpringBootHelloWorldTests {

	@Test
	public void contextLoads() {
	}

}
```

TestWebApp.java

Path → src/test/java/com/javainuse/test/ TestWebApp.java

```
package com.javainuse.test;

import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.get;
import static
org.springframework.test.web.servlet.result.MockMvcResultMatchers.content;
import static
org.springframework.test.web.servlet.result.MockMvcResultMatchers.jsonPath;
import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.status;

import org.junit.Before;
import org.junit.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.test.web.servlet.MockMvc;
import org.springframework.test.web.servlet.setup.MockMvcBuilders;
import org.springframework.web.context.WebApplicationContext;

public class TestWebApp extends SpringBootHelloWorldTests {

	@Autowired
	private WebApplicationContext webApplicationContext;

	private MockMvc mockMvc;

	@Before
	public void setup() {
		mockMvc =
MockMvcBuilders.webAppContextSetup(webApplicationContext).build();
	}

	@Test
	public void testEmployee() throws Exception {
		mockMvc.perform(get("/employee")).andExpect(status().isOk())
```

```
        .andExpect(content().contentType("application/json;charset=UTF-8"))

        .andExpect(jsonPath("$.name").value("emp1")).andExpect(jsonPath("$.designation").
value("manager"))

        .andExpect(jsonPath("$.empId").value("1")).andExpect(jsonPath("$.salary").value(3
000));

        }

}
```