

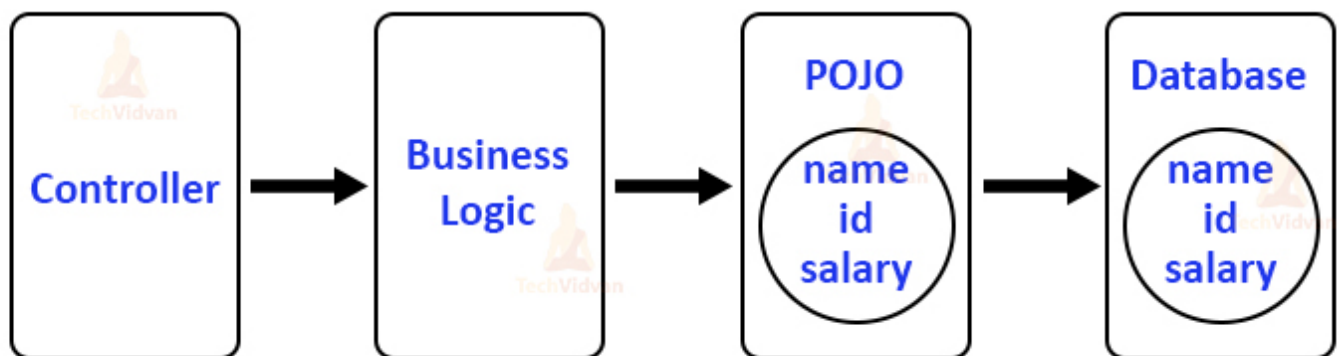
****Important Notes for Demos:****

***[POJO] - Plain Old Java Object**

In software engineering, a plain old Java object is an ordinary Java object, not bound by any special restriction.

Refer for difference between POJO and JavaBean

Java Beans



Spring Demos

- **Demo1** : Constructor-based dependency injection

__Demo~1__

Create a New Maven Project

Select an Archetype

- **Group id**: org.apache.maven.archetypes
- **Artifact id**: maven-archetype-quickstart
- **Version** : 1.4

New Maven Project

- **Group id:** com.ravada
- **Artifact id:** SpringDemo1
- **Package:** com.ravada.SpringDemo1

New Maven Project

New Maven project

Specify Archetype parameters

Group Id:

com.ravada

Artifact Id:

SpringDemo1

Version:

0.0.1-SNAPSHOT

Package:

com.ravada.SpringDemo1

Properties available from archetype:

Name	Value

Add...

Remove

Advanced

?

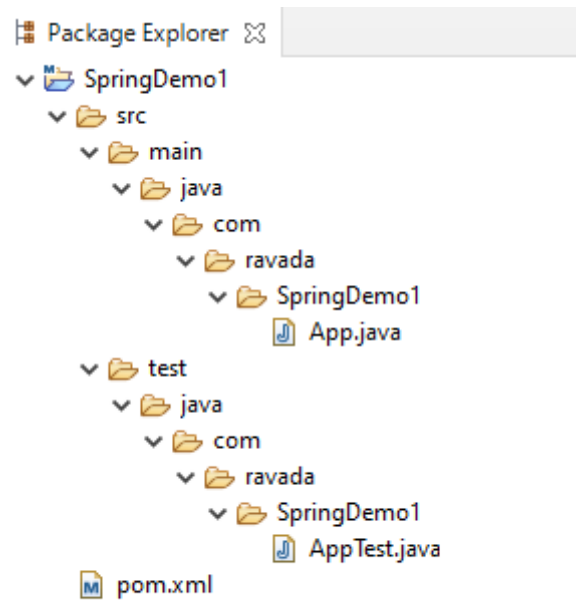
< Back

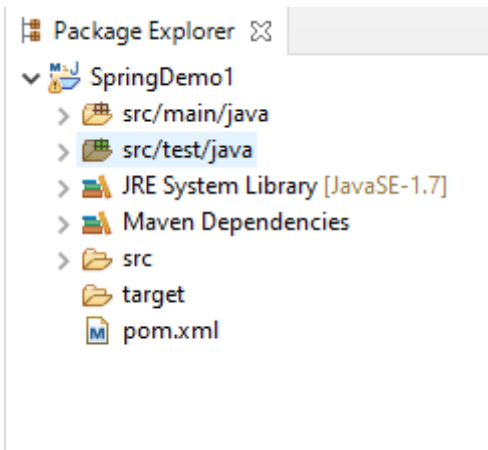
Next >

Finish

Cancel

File Structure



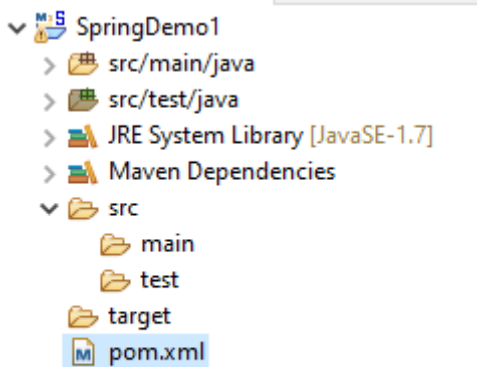


Added dependency to POM.xml file (Project Object Model)

The below dependency added in the POM.xml file

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-context</artifactId>
  <version>5.2.1.RELEASE</version>
</dependency>
```

Save the changes



*SpringDemo1/pom.xml

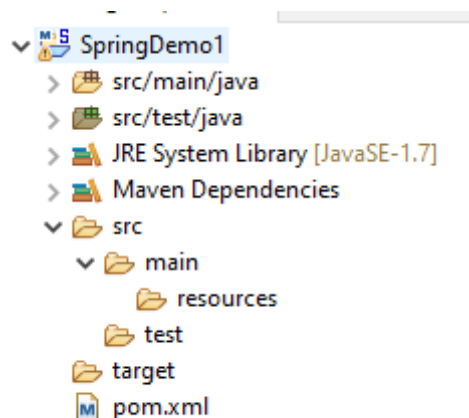
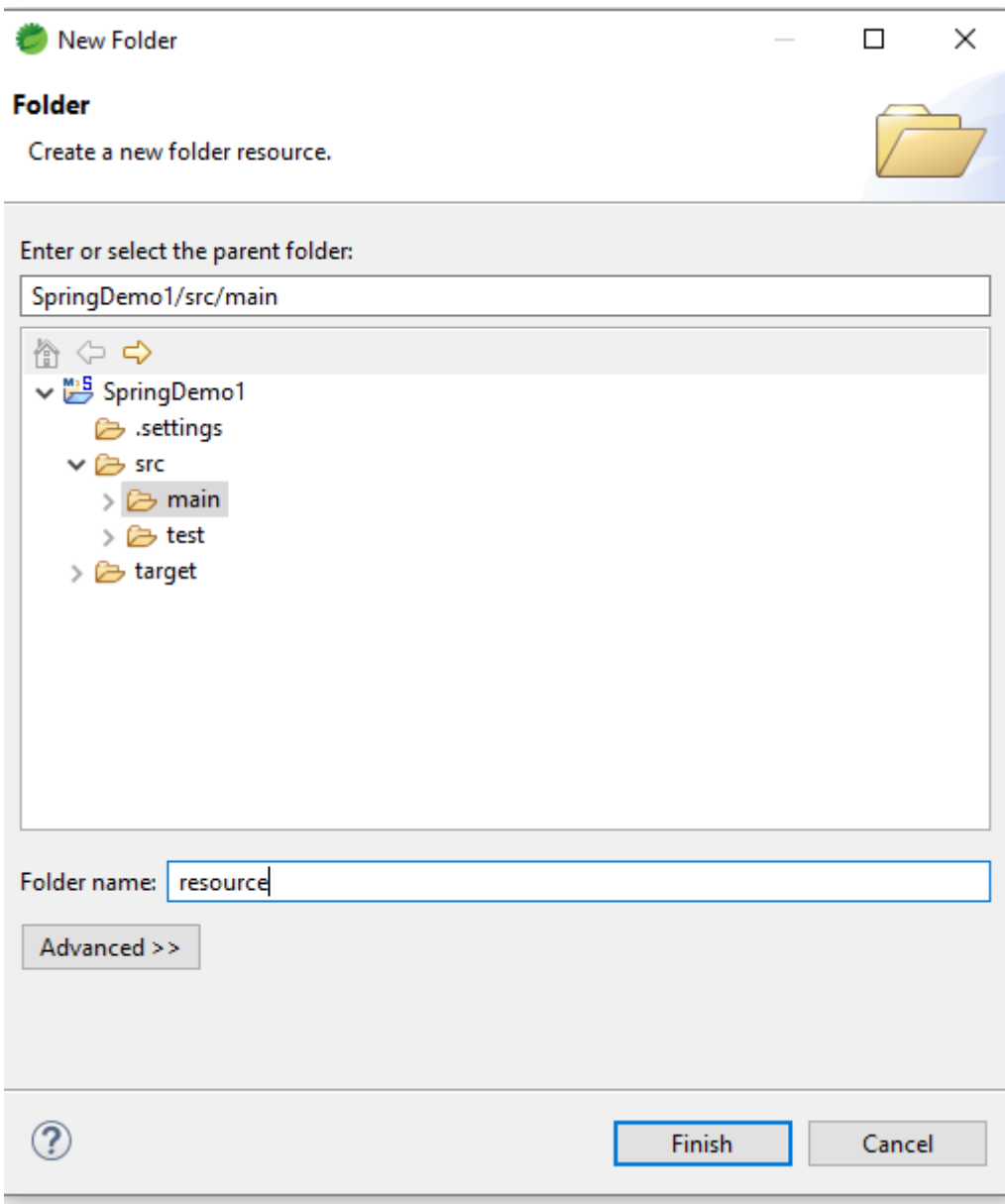
```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5   <modelVersion>4.0.0</modelVersion>
6
7   <groupId>com.ravada</groupId>
8   <artifactId>SpringDemo1</artifactId>
9   <version>0.0.1-SNAPSHOT</version>
10
11   <name>SpringDemo1</name>
12   <!-- FIXME change it to the project's website -->
13   <url>http://www.example.com</url>
14
15   <properties>
16     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
17     <maven.compiler.source>1.7</maven.compiler.source>
18     <maven.compiler.target>1.7</maven.compiler.target>
19   </properties>
20
21   <dependencies>
22     <dependency>
23       <groupId>org.springframework</groupId>
24       <artifactId>spring-context</artifactId>
25       <version>5.2.1.RELEASE</version>
26     </dependency>
27     <dependency>
28       <groupId>junit</groupId>
29       <artifactId>junit</artifactId>
30       <version>4.11</version>
31       <scope>test</scope>
32     </dependency>
33   </dependencies>
34
```

added

Rightclick->Maven->Update Project

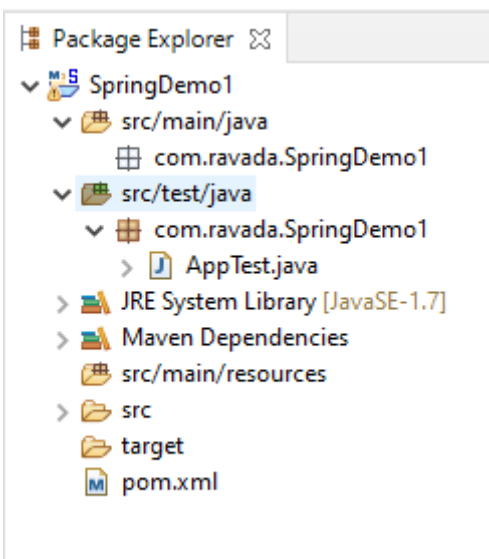
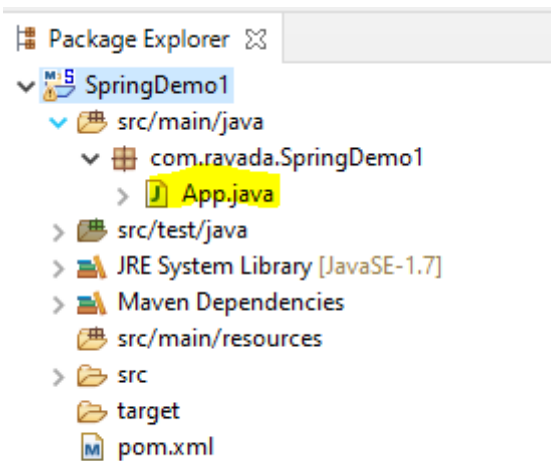
Add new folder **resources** in the src/main.

RClick on main folder->New-> New Folder




Rightclick->Maven->Update Project

Delete the App.java from /src/main/java



Create a new **FlightBean.java** class in Package **com.ravada.SpringDemo1**

Right Click on **com.ravada.SpringDemo1** package->New->Class

 New Java Class

Java Class

Create a new Java class.

Source folder:

Package:

☐ Enclosing type:

Name:

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass:

Interfaces:

Which method stubs would you like to create?

☐ public static void main(String[] args)

☒ Constructors from superclass

☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))

☐ Generate comments

Package Explorer

- SpringDemo1
 - src/main/java
 - com.ravada.SpringDemo1
 - FlightBean.java**
 - src/test/java
 - JRE System Library [JavaSE-1.7]
 - Maven Dependencies
 - src/main/resources
 - src
 - target
 - pom.xml

FlightBean.java

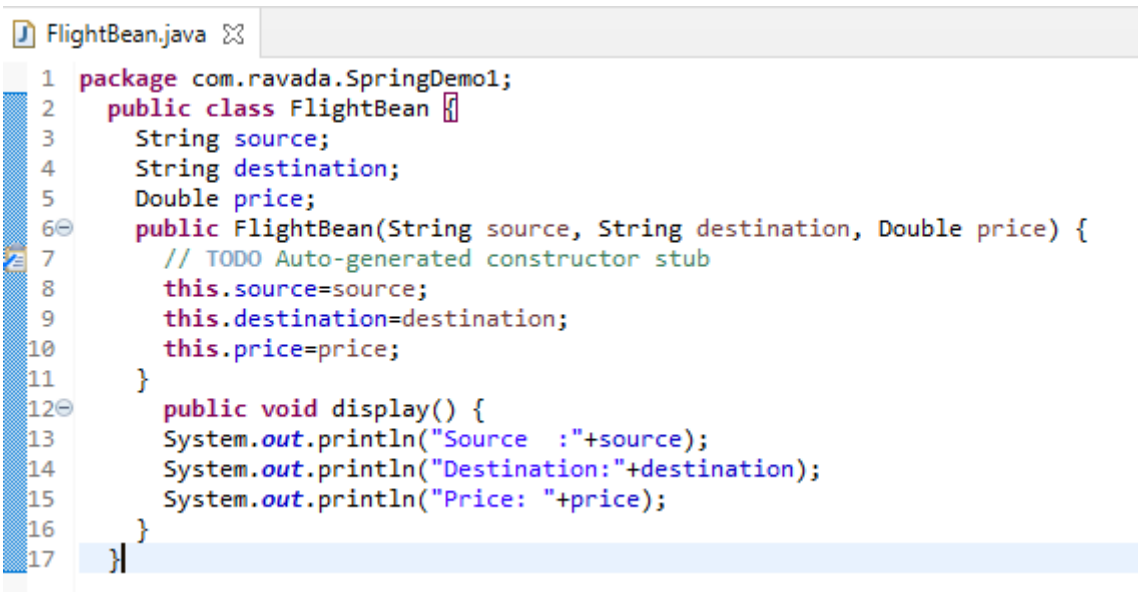
```
1 package com.ravada.SpringDemo1;
2
3 public class FlightBean {
4
5     public FlightBean() {
6         // TODO Auto-generated constructor stub
7     }
8
9 }
10
```

Source code of **FlightBean.java**

```

package com.ravada.SpringDemo1;
public class FlightBean {
    String source;
    String destination;
    Double price;
    public FlightBean(String source, String destination, Double price) {
        // TODO Auto-generated constructor stub
        this.source=source;
        this.destination=destination;
        this.price=price;
    }
    public void display() {
        System.out.println("Source  :"+source);
        System.out.println("Destination:"+destination);
        System.out.println("Price: "+price);
    }
}

```



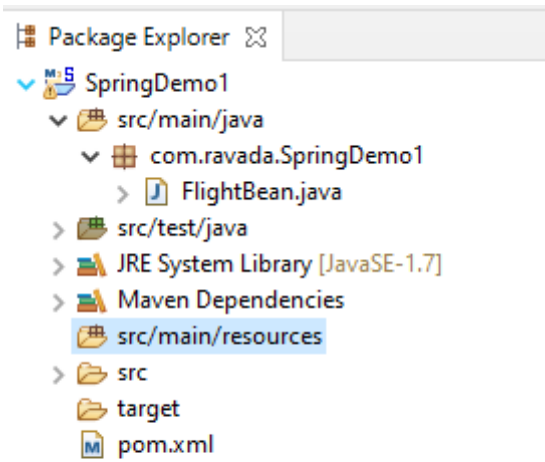
The screenshot shows a code editor window titled "FlightBean.java". The code is as follows:

```

1 package com.ravada.SpringDemo1;
2 public class FlightBean {
3     String source;
4     String destination;
5     Double price;
6     public FlightBean(String source, String destination, Double price) {
7         // TODO Auto-generated constructor stub
8         this.source=source;
9         this.destination=destination;
10        this.price=price;
11    }
12    public void display() {
13        System.out.println("Source  :"+source);
14        System.out.println("Destination:"+destination);
15        System.out.println("Price: "+price);
16    }
17 }

```

Adding a Spring Bean Configuration file to resources



Right-click on **src/main/resources** and select
New->Spring Bean Configuration File

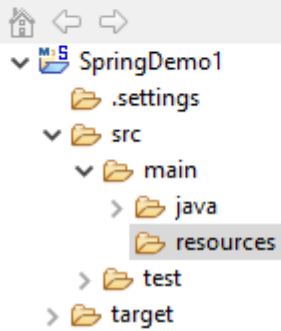
New Spring Bean Definition file

Select the location and give a name for the Spring Bean Definition file



Enter or select the parent folder:

SpringDemo1/src/main/resources



File name: config

Advanced >>

☒ Add Spring project nature if required

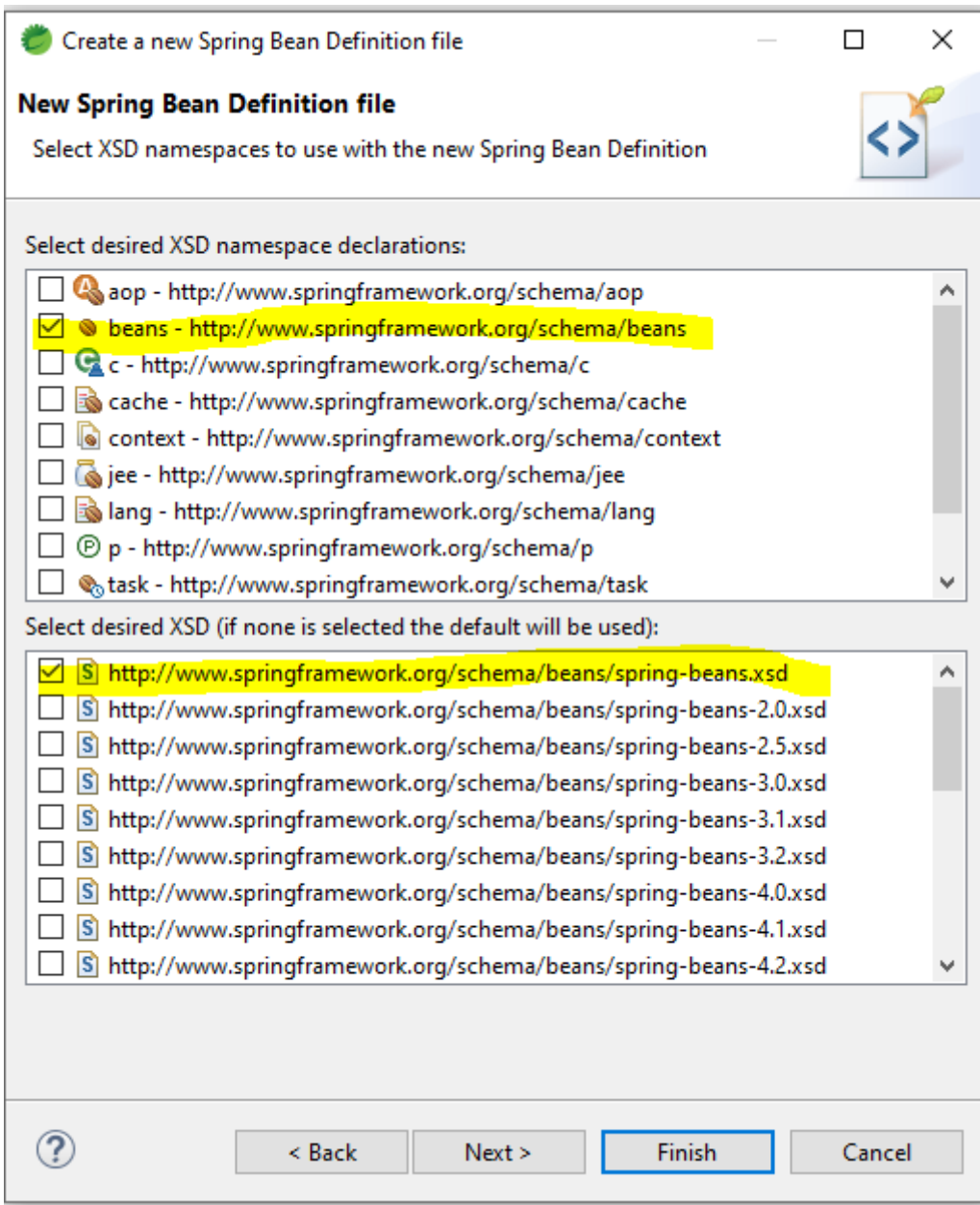


< Back

Next >

Finish

Cancel



Click on **Finish**



Add the below code in the **config.xml** file

```

<bean id="flightBean" class="com.ravada.SpringDemo1.FlightBean">
<constructor-arg name="source" value="Hyderabad"/>
<constructor-arg name="destination" value="Bangalore"/>
<constructor-arg name="price" value="5090.00"/>
</bean>

```

config.xml

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://www.springframework.org/schema/beans h
5
6 <bean id="flightBean" class="com.ravada.SpringDemo1.FlightBean">
7     <constructor-arg name="source" value="Hyderabad"/>
8     <constructor-arg name="destination" value="Bangalore"/>
9     <constructor-arg name="price" value="5090.00"/>
10 </bean>
11
12 </beans>
13

```

Delete the file **AppTest.java** from /src/test/java

Package Explorer


- SpringDemo1
 - src/main/java
 - com.ravada.SpringDemo1
 - FlightBean.java
 - src/test/java
 - com.ravada.SpringDemo1
 - AppTest.java
 - JRE System Library [JavaSE-1.7]
 - Maven Dependencies
 - src/main/resources
 - config.xml
 - src
 - target
 - pom.xml

Add the new **FlightTest.java** class to test the FlightBean

Right-click on package **com.ravada.SpringDemo1** in **SpringDemo1/src/test/java** and select New->Class

Package Explorer

- SpringDemo1
 - src/main/java
 - com.ravada.SpringDemo1
 - FlightBean.java
 - src/test/java
 - com.ravada.SpringDemo1
 - JRE System Library [JavaSE-1.7]
 - Maven Dependencies
 - src/main/resources
 - config.xml
 - src
 - main
 - test
 - target
 - pom.xml

 New Java Class

Java Class
Create a new Java class.

Source folder:

Package:

☐ Enclosing type:

Name:

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass:

Interfaces:

Which method stubs would you like to create?

☒ public static void main(String[] args)
☐ Constructors from superclass
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

Package Explorer

- SpringDemo1
 - src/main/java
 - com.ravada.SpringDemo1
 - FlightBean.java
 - src/test/java
 - com.ravada.SpringDemo1
 - FlightTest.java
 - JRE System Library [JavaSE-1.7]
 - Maven Dependencies
 - src/main/resources
 - config.xml
 - src
 - main
 - test
 - target
 - pom.xml

FlightTest.java

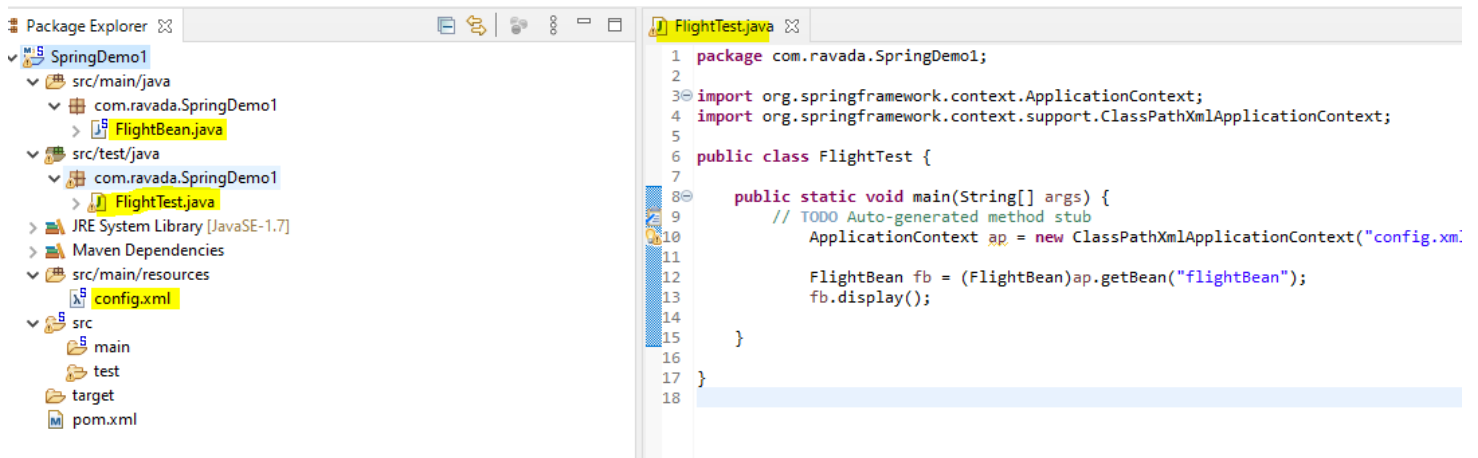
```
1 package com.ravada.SpringDemo1;
2
3 public class FlightTest {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7     }
8
9 }
10
11
```

```

package com.ravada.SpringDemo1;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class FlightTest {
    public static void main(String[] args) {
        ApplicationContext ap = new ClassPathXmlApplicationContext("config.xml");

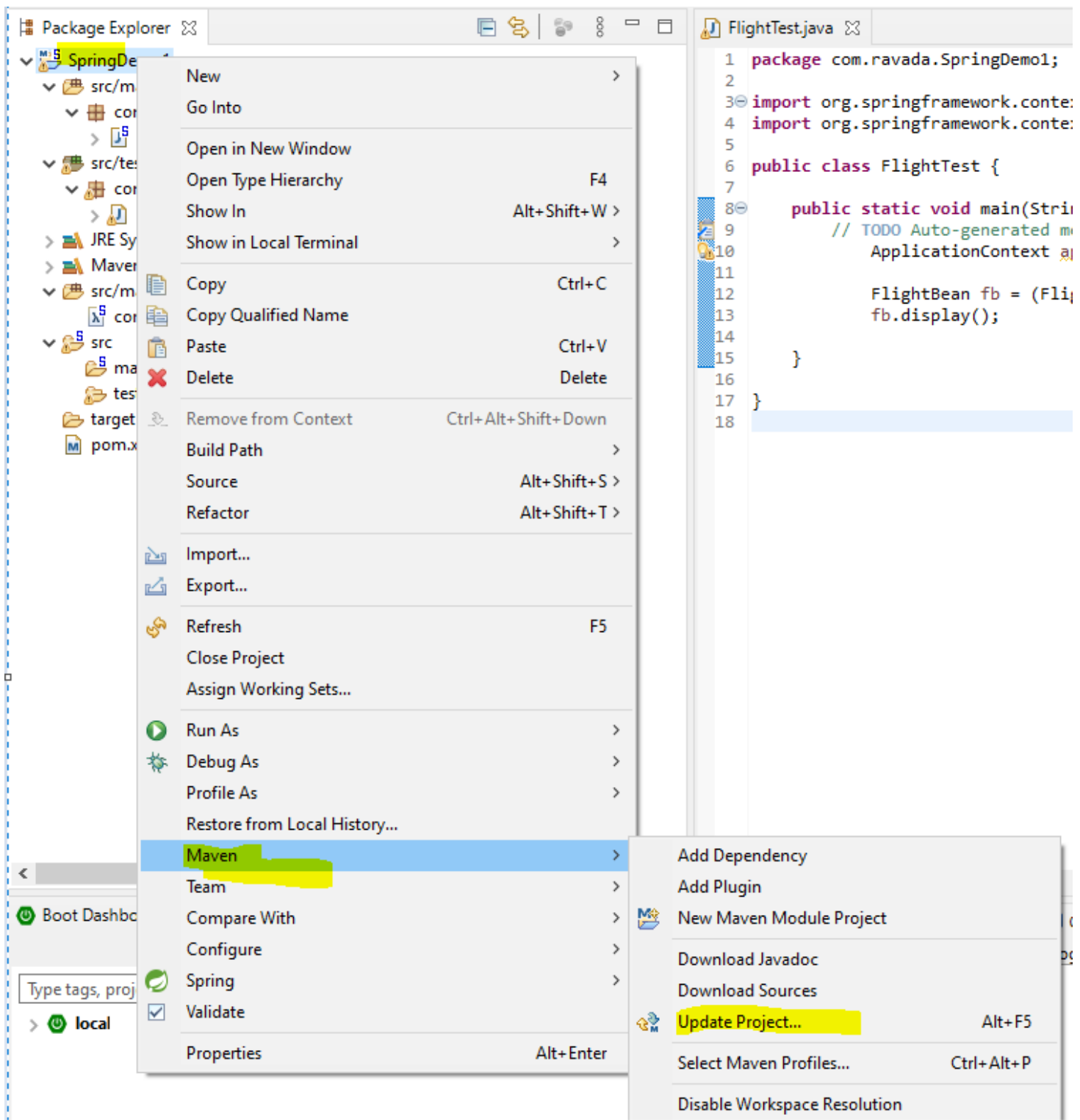
        FlightBean fb = (FlightBean)ap.getBean("flightBean");
        fb.display();
    }
}

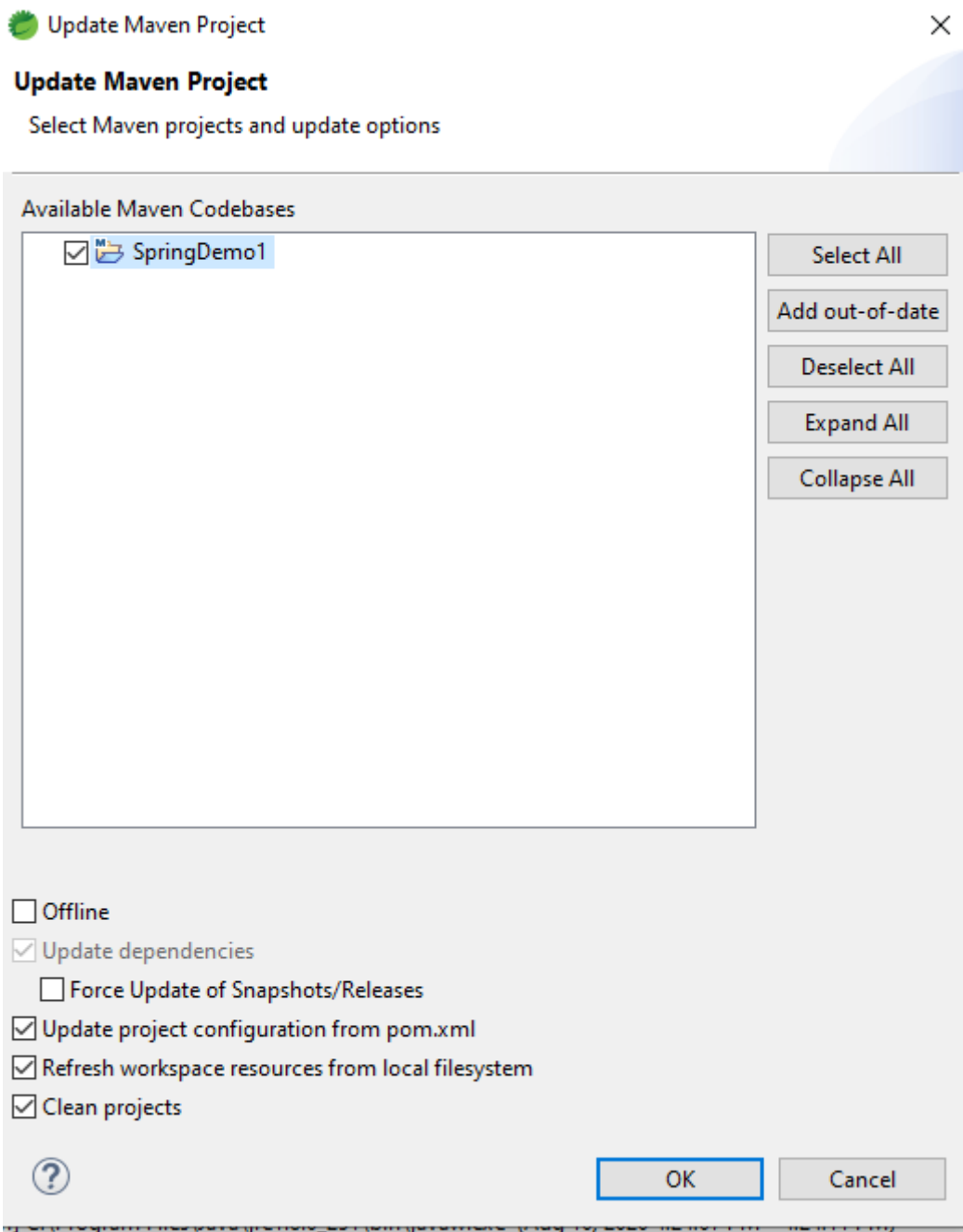
```



To update the Maven Project **SpringDemo1**

RClick on Maven Project **SpringDemo1** and Select Maven-> Update Project (Alt + F5)

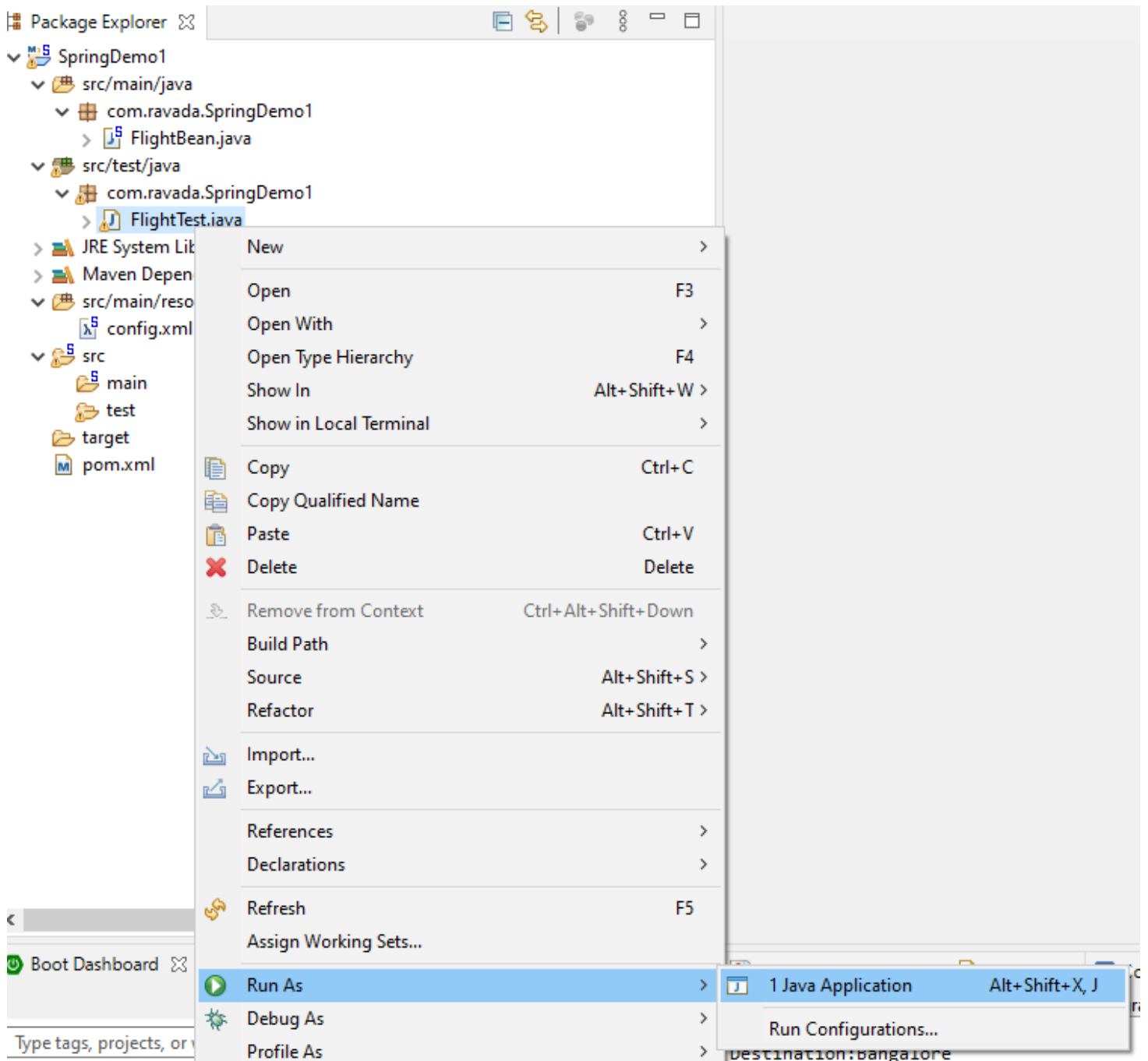




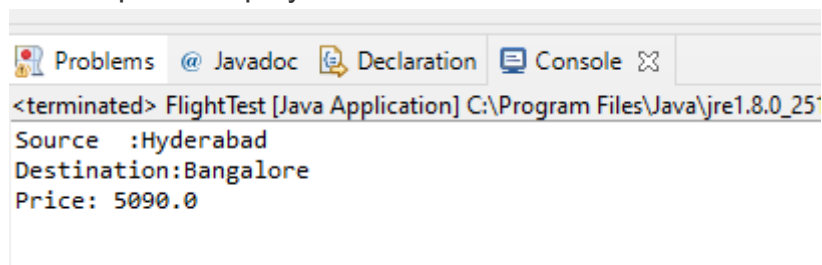
Click **OK** to Update the Maven Project **SpringDemo1**

Run the Spring Application (**FlightTest.java**)

RClick on FlightTest.java -> Run As -> Java Application



The output is displayed in Console screen



Topics:

Constructor-based DI is accomplished when the container invokes a class constructor with a number of arguments, each representing a dependency on other class.

Spring IOC use default constructor to create the object of the class .if your class has parameterized constructor , then you have to call the parameterized constructor to create the object