

**Java Spring Laboratory 6:****Coding programs to demonstrate Spring Autowiring.**

**Program 1:** Write a program that demonstrates component-discovery through enabling of component scanning step of Spring Autowiring. In particular, create a Car interface that specifies public void drive method. Suzuki class should implement Car interface. Enable component-scanning and configure the spring application such that an instance of the Suzuki class gets discovered and registered as a bean in the Spring Application Context. Validate this by looking-up for the suzuki bean in the main method of your program and invoking its drive method. Invocation of drive on a Car instance should print the brand, model and year of the particular Car instance on the console. [Note: Make Suzuki class hard-code model and year parameters so that it can be instantiated through its default constructor. Feel free to have any other assumptions you need to make this work]

**Solution:****Car.java**

```
package nmitd.autowiring.lab6;
```

```
public interface Car {  
  
    public void drive();  
}
```

**Suzuki.java**

```
package nmitd.autowiring.lab6;
```

```
import org.springframework.stereotype.Component;
```

```
@Component
```

```
public class Suzuki implements Car {
```

```
    private String model;
```

```
    private String year;
```

```
    public Suzuki()  
    {
```

```
        model="Dezire";
```

```
        year="2008";
```

```
    }
```

```
    @Override
```

```
    public void drive() {
```

```
        System.out.println("You are driving "+this); //this will invoke toString
```

```
method
```

```
    }
```

```
    public String toString() {
```

```
        return "Suzuki "+model+" (" +year+");
```

```
    }
```

```
}
```

**CarConfig.java**

```
package nmitd.utowiring.lab6;

import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;

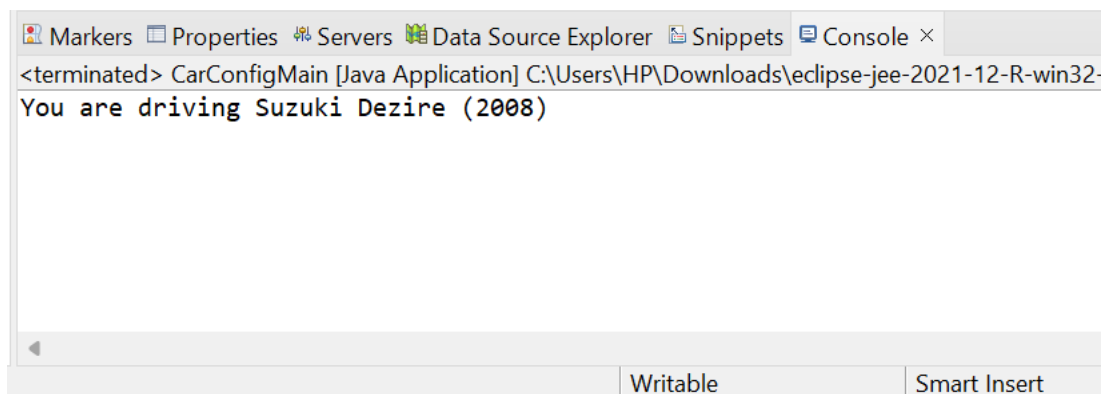
@Configuration
@ComponentScan
public class CarConfig {

}
```

**CarConfigMain.java**

```
package nmitd.utowiring;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;
import nmitd.utowiring.lab6.*;

public class CarConfigMain {
    public static void main(String[] args) {
        AnnotationConfigApplicationContext appCtx = new
AnnotationConfigApplicationContext(CarConfig.class);
        Car c1=appCtx.getBean("suzuki",Suzuki.class);
        c1.drive();
        appCtx.close();
    }
}
```

**Output:**

The screenshot shows the Eclipse IDE's console window. The title bar includes tabs for Markers, Properties, Servers, Data Source Explorer, Snippets, and Console. The console text reads: "<terminated> CarConfigMain [Java Application] C:\Users\HP\Downloads\eclipse-jee-2021-12-R-win32- You are driving Suzuki Dezire (2008)". At the bottom of the console, there are two buttons: "Writable" and "Smart Insert".

**Program 2:** Write a program that demonstrates how dependency injection works in an Auto wired Spring Application. To demonstrate this, extend Program 1 by adding a RichPerson class implementing Person interface. Person interface specifies aboutMe method and driveCar methods specified. RichPerson instance depends on Car instance. Invoking of driveCar method on RichPerson object in turn should invoke the drive method on its depending Car instance. Use @Autowired annotation wherever needed to signal Spring to dependency inject a Car instance into the RichPerson bean. Validate the success of component creation and auto-wired dependency injection by invoking aboutMe and driveCar methods on the richPerson bean in the main method of your program. [Note: Hard-code the first name and last name for the RichPerson class, so that dependency injection of only Car instance is needed. You may have any other assumptions you need to make this work]

**Solution:****Car.java**

```
package nmitd.autowiring.lab6;
```

```
public interface Car {
```

```
    public void drive();
}
```

**Mercedes.java**

```
package nmitd.autowiring.lab6;
```

```
import org.springframework.stereotype.Component;
```

```
@Component
```

```
public class Mercedes implements Car {
```

```
    private String model;
```

```
    private String year;
```

```
    public Mercedes()
```

```
    {
```

```
        model="S-Class";
```

```
        year="2020";
```

```
    }
```

```
    @Override
```

```
    public void drive() {
```

```
        System.out.println("Mercedes "+model+" (" +year+"");
```

```
    }
```

```
}
```

**Person.java**

```
package nmitd.autowiring.lab6;
```

```
public interface Person {
```

```
    public void aboutMe();
```

```
    public void driveCar();
```

```
}
```

**RichPerson.java**

```
package nmitd.autowiring.lab6;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.stereotype.Component;
```

```
@Component
```

```
public class RichPerson implements Person {
```

```
    private Car myCar;
```

```
    private String fname;
```

```
    private String lname;
```

```
    public RichPerson() {
```

```
        fname="Mukesh";
```

```
        lname="Ambani";
```

```
    }
```

```
    @Autowired
```

```
    public void getCar(Car myCar) {
```

```
        this.myCar=myCar;
```

```
    }
```

```
    @Override
```

```
    public void aboutMe() {
```

```
        System.out.println("Myself "+fname+" "+lname+", am a rich person, owning ");
```

```
    }
```

```
    @Override
```

```
    public void driveCar() {
```

```
        // TODO Auto-generated method stub
```

```
        myCar.drive();
```

```
    }
```

```
}
```

**PersonCarConfig.java**

```
package nmitd.autowiring.lab6;
```

```
import org.springframework.context.annotation.ComponentScan;
```

```
import org.springframework.context.annotation.Configuration;
```

```
@Configuration
```

```
@ComponentScan
```

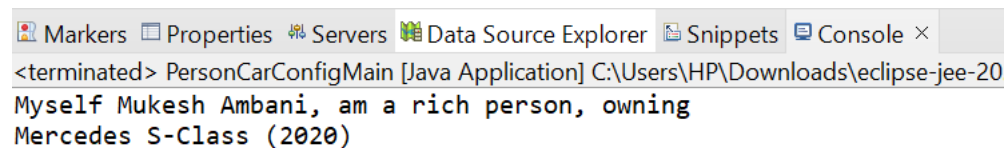
```
public class PersonCarConfig {
```

```
}
```

**PersonCarConfigMain.java**

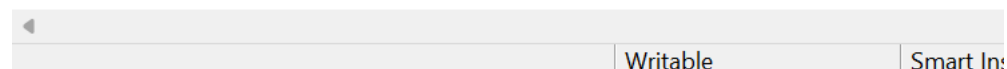
```
package nmitd.autowiring;  
import org.springframework.context.annotation.AnnotationConfigApplicationContext;  
import nmitd.autowiring.lab6.*;
```

```
public class PersonCarConfigMain {  
    public static void main(String[] args) {  
        AnnotationConfigApplicationContext appCtx = new  
AnnotationConfigApplicationContext(PersonCarConfig.class);  
        Person p1=appCtx.getBean(RichPerson.class);  
        p1.aboutMe();  
        p1.driveCar();  
        appCtx.close();  
    }  
}
```

**Output:**

The screenshot shows the Eclipse IDE's console window. The title bar includes tabs for Markers, Properties, Servers, Data Source Explorer, Snippets, and Console. The console output displays the program's execution, which has terminated. The output text is: "<terminated> PersonCarConfigMain [Java Application] C:\Users\HP\Downloads\eclipse-jee-2019-03-MR2-win32-x86\_64.exe Myself Mukesh Ambani, am a rich person, owning Mercedes S-Class (2020)".

```
<terminated> PersonCarConfigMain [Java Application] C:\Users\HP\Downloads\eclipse-jee-2019-03-MR2-win32-x86_64.exe  
Myself Mukesh Ambani, am a rich person, owning  
Mercedes S-Class (2020)
```



The screenshot shows the bottom status bar of the Eclipse IDE. It contains three labels: a left-pointing arrow, 'Writable', and 'Smart In'.

```
◀ Writable Smart In
```