SPRINGBOOT

Basic things to know before getting started…

Spring Boot is a framework that simplifies Java application development by providing pre-configured settings, auto-configuration, and embedded servers. It enables developers to build standalone, production-ready applications with minimal setup and configuration.

- Making use of the spring initializr.
- There is only one @SpringBootApplication that is present inside the main class.
- you can’t run two applications on the same server. Hence, go to application.properties and define a new server using server.port = 8081.
- Before that make sure u do the following : project<clean. This ensures that there is no caching of the project.
- Apache Tomcat is a default server.

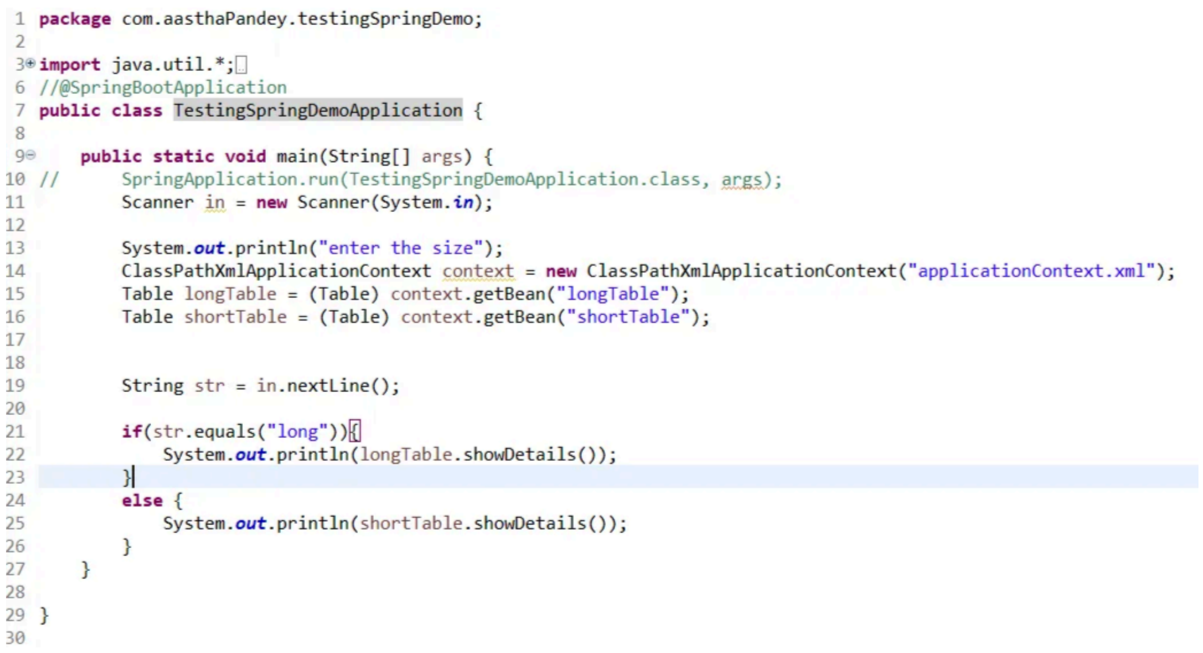
IOC

> INVERSION OF CONTROL

- in lame terms i can say that spring creates object for us and we the users can use it.
- Hence, I don’t actually need to create the obj every time by myself. This makes easy to work with building applications on a large-scale.

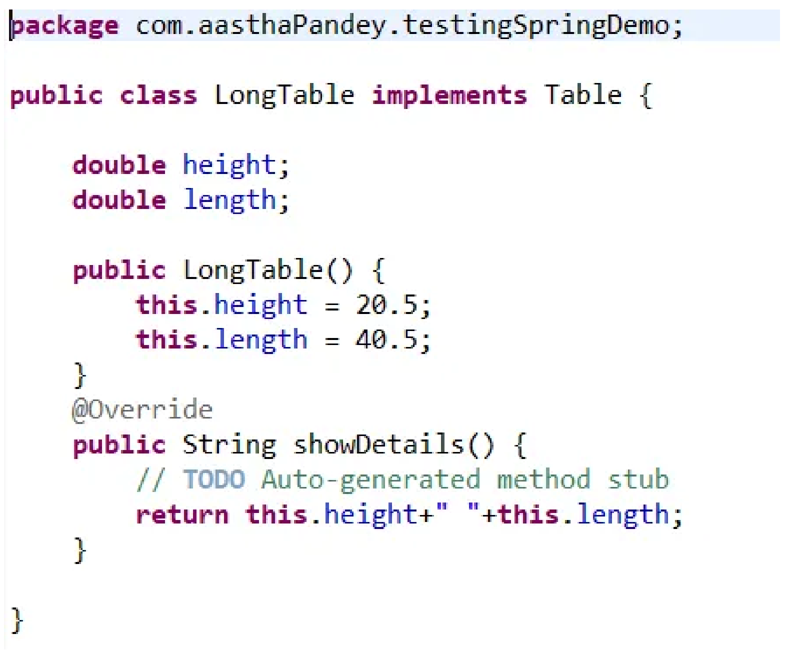
Here’s a simple console-level application that I have attached to have a better look on the working of IOC:

application.java



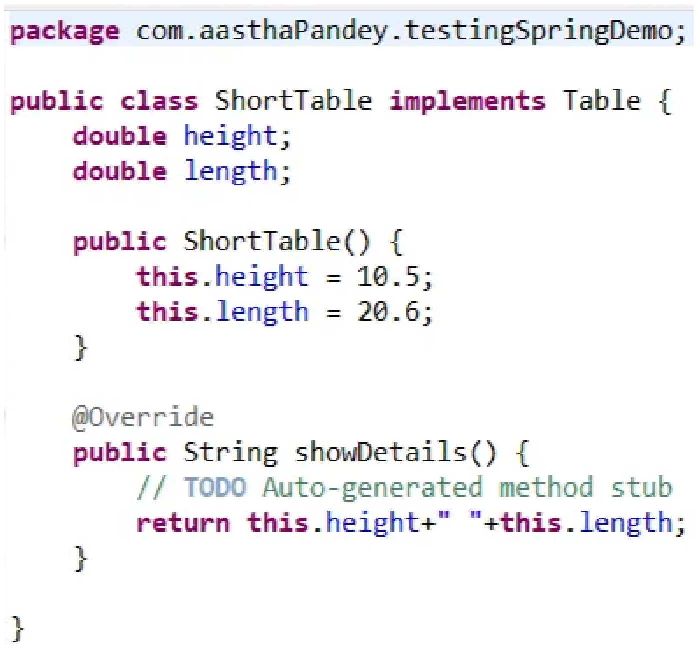
```java
package com.aasthaPandey.testingSpringDemo;

import java.util.*;
//@SpringBootApplication
public class TestingSpringDemoApplication {

    public static void main(String[] args) {
//      SpringApplication.run(TestingSpringDemoApplication.class, args);
        Scanner in = new Scanner(System.in);

        System.out.println("enter the size");
        ClassPathXmlApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
        Table longTable = (Table) context.getBean("longTable");
        Table shortTable = (Table) context.getBean("shortTable");


        String str = in.nextLine();

        if(str.equals("long")){
            System.out.println(longTable.showDetails());
        }
        else {
            System.out.println(shortTable.showDetails());
        }
    }

}

```

LongTable.java



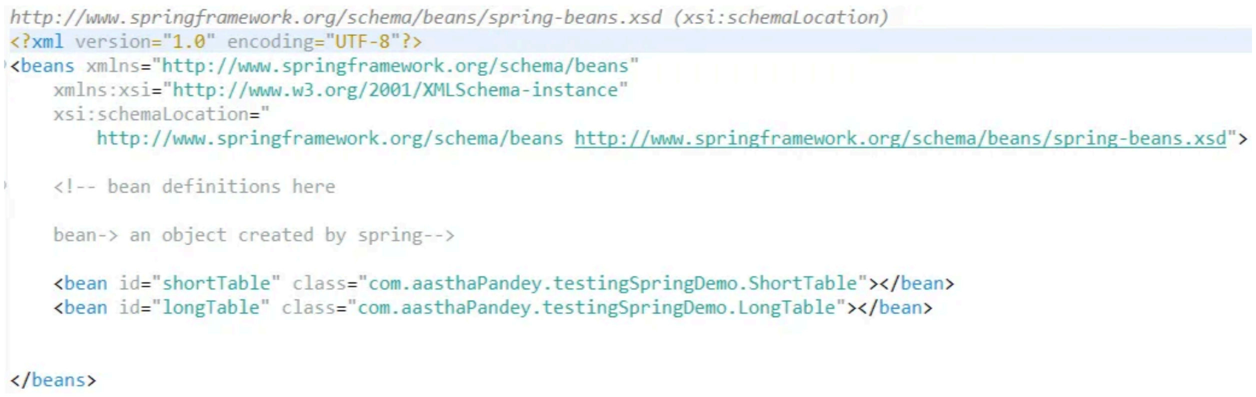
```java
package com.aasthaPandey.testingSpringDemo;

public class LongTable implements Table {

    double height;
    double length;

    public LongTable() {
        this.height = 20.5;
        this.length = 40.5;
    }
    @Override
    public String showDetails() {
        // TODO Auto-generated method stub
        return this.height+" "+this.length;
    }

}
```

ShortTable.java



```java
package com.aasthaPandey.testingSpringDemo;

public class ShortTable implements Table {
    double height;
    double length;

    public ShortTable() {
        this.height = 10.5;
        this.length = 20.6;
    }

    @Override
    public String showDetails() {
        // TODO Auto-generated method stub
        return this.height+" "+this.length;
    }

}
```

applicationContext.xml



```xml
http://www.springframework.org/schema/beans/spring-beans.xsd (xsi:schemaLocation)
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="
        http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">

    <!-- bean definitions here

    bean-> an object created by spring-->

    <bean id="shortTable" class="com.aasthaPandey.testingSpringDemo.ShortTable"></bean>
    <bean id="longTable" class="com.aasthaPandey.testingSpringDemo.LongTable"></bean>


</beans>
```

Key-highlights

- when the obj is created by spring for the developer then it's known as **Bean.**
- In applicationContext.xml we define the beans once and use as many times as we want inside the main class.

DAY-2

DAY-3