Spring @Component, @Repository, @Service and @Controller Stereotype Annotations

In this tutorial we would discuss about the **Stereotype Annotations** in Spring. Spring @Component, @Repository, @Service and @Controller are Stereotype Annotations. @Component is generic stereotype annotation for any Spring-managed component. In the previous version **Spring 2.0** introduce the first **Stereotype Annotations**name as ***@Repository.*** The [**@Component annotations**](https://www.dineshonjava.com/spring-component-annotation/) introduced in **Spring 2.5**are really just a continuation of the **“stereotype”** annotations introduced in **Spring 2.0**. **Stereotype annotations** are markers for any class that fulfills a role within an application. This helps remove, or at least greatly reduce, the **Spring XML** configuration required for these components.

These annotations are used to stereotype classes with regard to the application tier that they belong to. Classes that are annotated with one of these annotations will automatically be registered in the Spring application context if **<context:component-scan>** is in the **Spring XML configuration(spring.xml).**

**The Four Types of Spring Stereotype Components and Their Purposes:**

+------------+-----------------------------------------------------+

**| Annotation | Meaning                                             |**

+------------+-----------------------------------------------------+

| [**@Component**](https://www.dineshonjava.com/spring-component-annotation/) | **generic stereotype for any Spring-managed component** |

| **@Repository**| **stereotype for persistence layer**                    |

| **@Service** | **stereotype for service layer**                        |

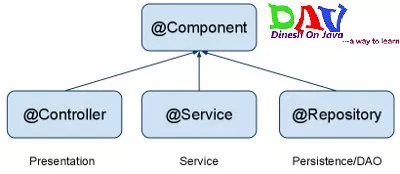
| **@Controller|** **stereotype for presentation layer (spring-mvc)**      |

+------------+-----------------------------------------------------+

[**Annotation @Component**](https://www.dineshonjava.com/spring-component-annotation/)**:**

**Description:**

[**@Component**](https://www.dineshonjava.com/spring-component-annotation/) is a generic stereotype for any Spring-managed component.**@Repository, @Service, and @Controller** are specializations of [**@Component**](https://www.dineshonjava.com/spring-component-annotation/) for more specific use cases, for example, in the persistence, service, and presentation layers, respectively.



@Component

public class Circle

{

private Point center;

}

**Annotation @Repository:**

**Description:**  
In **Spring 2.0** and later, the ***@Repository*** annotation is a marker for any class that fulfills the role or stereotype (**also known as Data Access Object or DAO**) of a repository. Among the uses of this marker is the automatic translation of exceptions.  
A class that serves in the persistence layer of the application as a data access object (**DAO**), otherwise known as a repository in some other technologies. Annotate all your **DAO**classes with ***@Repository***. All your database access logic should be in **DAO**classes.

@Repository

public class CircleDaoImpl implements CircleDao

{

private Point center;

----

}

**Annotation @Service:**

Annotate all your service classes with ***@Service***. All your business logic should be in Service classes.

@Service

public class CircleServiceImpl implements CircleService

{

private Point center;

----

}

**Annotation @Controller:**

**Target:**

**Class**

**Description:**  
The ***@Controller*** is a class level annotation, which indicates that the annotated class is a Spring component of type “**controller**“.  
The ***@Controller*** annotation indicates that a particular class serves the role of a controller. Spring does not require you to extend any controller base class or reference the **Servlet API**. However, you can still reference Servlet-specific features if you need to. In Spring MVC you can make controller class very easily by prefixing ***@Controller*** before any class declaration.

@Controller

public class CircleController

{

private Point center;

----

}

**Enable component scanning**  
Spring by default does not scan means Spring container does create bean for those classes whose annotated with above for stereotype annotations. So we have to enable component scanning explicity by using “context:component-scan” tag in your applicationContext.xml file. So stereotype annotations will be scanned and configured only when they are scanned by DI container of spring framework.

<context:component-scan base-package="com.dineshonjava.app.service" />

<context:component-scan base-package="com.dineshonjava.app.dao" />

<context:component-scan base-package="com.dineshonjava.app.controller" />

The context:component-scan element requires a base-package attribute, the value of base-package attribute should specifies a starting point for a recursive component search. Spring recommends do not use your top package for scanning, so you should declare specific component-scan elements.

**Note:**If you are using component-scan property for context namespace then you no longer need to declare context:annotation-config, because autowiring is implicitly enabled when component scanning is enabled.

**Where to use stereotype annotations?**  
Always use these annotations over concrete classes; not over interfaces.

* **@Controller**annotation is for a class as a Spring Web MVC controller. It is a meta annotation of @Component, so beans annotated with it are automatically imported into the Spring container. If you add the @Controller annotation to a class then you can use handler mappling annotation i.e. @RequestMapping; to map URLs to instance methods of a class.
* **@Service**annotation is for a class as a Service of application.
* **@Repository** annotation is more suitable annotation that provides additional benefits specifically for DAOs. The @Repository annotation is a meta annotation of the @Component annotation with similar use and functionality. In addition to importing the DAOs into the DI container, it also makes the unchecked exceptions eligible for translation into Spring DataAccessException.
* **@Component** should be used when your class does not fall into either of three categories i.e. ***Controllers***, ***Services***and ***DAOs***.

