Difference Between @Controller and @RestController Annotation in Spring

Last Updated : 23 Jul, 2025

Spring Annotations are a form of metadata that provides data about a program. Annotations are used to provide supplemental information about a program. It does not directly affect the operation of the code they annotate. It does not change the action of the compiled program.

Understanding the **difference between @Controller and @RestController annotations** in Spring is essential for developing web applications and RESTful APIs. Both annotations serve the purpose of handling web requests but differ in how they process and return responses.

The main difference is:

- @Controller is used in Spring MVC applications to handle web requests and return views.
- @RestController is a specialized version of @Controller that combines @ResponseBody, making it ideal for RESTful APIs.

@Controller vs @RestController

The table below demonstrates the differences between @Controller and @RestController annotations in Spring.

Feature	@Controller	@RestController
Definition	This annotation is used to mark classes as Spring MVC controllers.	This is a specialized controller used for RESTful web services.

Feature	@Controller	@RestController
Inheritance	Extends @Component annotation.	Extends @Controller annotation.
View Support	Returns a view in Spring Web MVC.	Cannot return a view.
Response Handling	Requires @ResponseBody for sending responses as JSON or XML.	Assumes @ResponseBody by default, eliminating the need for explicit annotations.
Request Handling	Used for traditional MVC applications where views (JSP, Thymeleaf) are returned.	Used for REST APIs where JSON or XML responses are sent directly to the client.
Spring Version	Introduced in Spring 2.5.	Introduced in Spring 4.0.

@Controller Annotation

Spring @Controller annotation is also a specialization of @Component annotation. The @Controller annotation indicates that a particular class serves the role of a controller. Spring Controller annotation is typically used in combination with annotated handler methods based on the @RequestMapping annotation. It can be applied to classes only. It's used to mark a class as a web request handler. It's mostly used with Spring MVC applications. This annotation acts as a stereotype for the annotated class, indicating its role. The dispatcher scans such annotated classes for mapped methods and detects @RequestMapping annotations.

Example: Using @Controller in a Spring MVC Application

```
@Controller
public class HomeController {

    @GetMapping("/home")
    public String homePage(Model model) {
        model.addAttribute("message", "Welcome to Spring MVC!");

        // Returns a view (home.html or home.jsp)
        return "home";
    }
}
```

Explanation: In the above example,

- The @Controller annotation marks the class as a Spring MVC controller.
- The homePage() method maps to /home and returns a view named home.

@RestController Annotation

The @RestController annotation is used for making restful web services. This annotation is used at the class level and allows the class to handle the requests made by the client. Let’s understand @RestController annotation using an example. The RestController allows to handle all REST APIs such as GET, POST, Delete, and PUT requests.

Example: Using @RestController in a REST API

```
@RestController
public class ProductController {

    @GetMapping("/product")
    public Product getProduct() {
        return new Product(1, "Laptop", 80000);
    }
}
```

Explanation: In the above example,

- The @RestController annotation combines @Controller and @ResponseBody.
- The getProduct() method returns a Product object that is automatically converted into JSON.

GeeksforGeeks
Sanchhaya Education Private Limited

Corporate & Communications Address:
A-143, 7th Floor, Sovereign Corporate
Tower, Sector- 136, Noida, Uttar Pradesh
(201305)

Registered Address:
K 061, Tower K, Gulshan Vivante
Apartment, Sector 137, Noida, Gautam
Buddh Nagar, Uttar Pradesh, 201305