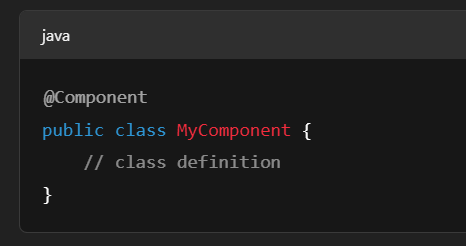
**Core Spring Annotations**

**@Component**

* + **Description:** Marks a class as a Spring component (bean). Spring will automatically detect and register this class as a bean in the application context during classpath scanning.
  + **Use case:** Use it to define general-purpose beans that don’t fall under specialized categories like @Service, @Repository, etc.



**@Service**

* **Description:** A specialization of @Component for service-layer beans, usually containing business logic.
* **Use Case:** Typically used for service classes that perform business logic.

A computer screen shot of a black screen

AI-generated content may be incorrect.

**@Repository**

* **Description:** A specialization of @Component used to annotate data access objects (DAOs). It can provide additional features like exception translation for database-related errors.
* **Use Case:** Used in the persistence layer (data access layer) of your application.

A screen shot of a computer

AI-generated content may be incorrect.

**@Controller**

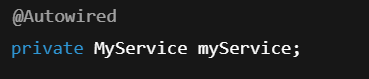
* **Description:** A specialization of @Component used for Spring MVC controllers. It marks a class as a Spring MVC controller, responsible for handling HTTP requests.
* **Use Case:** Used in the presentation layer, specifically for web applications.

**@RestController**

* **Description:** A specialization of @Controller used in RESTful web services. It combines @Controller and @ResponseBody to automatically serialize return objects to JSON/XML.
* **Use Case:** Used for creating RESTful APIs in Spring Boot or Spring MVC.

**@Autowired**

* **Description:** Marks a constructor, field, or setter to be autowired by Spring's DI container. Spring will automatically inject a matching bean into the marked field or method.
* **Use Case:** To automatically wire dependencies.



**@Qualifier**

* **Description:** Used in conjunction with @Autowired to specify which bean to inject when there are multiple beans of the same type.
* **Use Case:** Helps resolve ambiguity when multiple beans of the same type are available.

A black screen with white text

AI-generated content may be incorrect.

**@Inject**

* **Description:** Part of Java's standard dependency injection (JSR-330) specification. It serves the same purpose as @Autowired but is more portable across different DI frameworks.
* **Use Case:** To inject dependencies into a class.

A black background with white text

AI-generated content may be incorrect.

**@Value**

* **Description:** Injects values into fields or methods from properties files or environment variables.
* **Use Case:** Useful for injecting configuration values into beans.

A black background with white text

AI-generated content may be incorrect.

**Configuration and Bean Definition Annotations**

**@Configuration**

* **Description:** Marks a class as a source of bean definitions for the Spring application context. Typically used in Java-based Spring configuration.
* **Use Case:** Used to define configuration classes instead of using XML configuration files.

A screen shot of a computer program

AI-generated content may be incorrect.

**@Bean**

* **Description:** Indicates that a method produces a bean to be managed by the Spring container. Used within @Configuration classes.
* **Use Case:** To define beans in Java-based configuration.

**@Scope**

* **Description:** Specifies the scope of a Spring bean (e.g., singleton, prototype, etc.).
* **Use Case:** Defines the lifecycle and visibility of a Spring bean.

**@RequestMapping**

* **Description:** Maps HTTP requests to handler methods of MVC controllers. It can be used to map a specific HTTP method (GET, POST, etc.) to a method.
* **Use Case:** Commonly used to define the URL mapping of controller methods.
* **Example:**

A black screen with white text

AI-generated content may be incorrect.

**@GetMapping, @PostMapping, @PutMapping, @DeleteMapping**

* **Description:** Specialized versions of @RequestMapping for handling specific HTTP methods.
* **Use Case:** Used to map HTTP GET, POST, PUT, or DELETE requests to controller methods.

A screen shot of a computer

AI-generated content may be incorrect.

**@PathVariable**

* **Description:** Binds a method parameter to a URI template variable.
* **Use Case:** Used values from URI paths.

A computer screen shot of a black background

AI-generated content may be incorrect.

**@RequestParam**

* **Description:** Binds a request parameter to a method parameter.
* **Use Case:** Used to bind query parameters from the URL to method parameters.

A computer screen shot of a black background

AI-generated content may be incorrect.

**Spring Data JPA Annotations**

Spring Data JPA simplifies database access by providing repository functionality and integration with JPA (Java Persistence API).

**@Entity**

* **Description:** Marks a class as an entity that will be persisted to a database.
* **Use Case:** Used to define a JPA entity (a table in the database).

A screen shot of a computer

AI-generated content may be incorrect.

**@Id**

* **Description:** Marks a field as the primary key for the entity.
* **Use Case:** Used to designate the primary key of an entity class.

A black background with white text

AI-generated content may be incorrect.

**@GeneratedValue**

* **Description:** Defines the strategy for generating primary key values for entities.
* **Use Case:** Specifies how the ID of an entity is generated (e.g., auto-increment).

A black background with white text

AI-generated content may be incorrect.

**Spring Boot Annotations**

**@SpringBootApplication**

* **Description:** A convenience annotation that combines @Configuration, @EnableAutoConfiguration, and @ComponentScan. It marks the main class of a Spring Boot application.
* **Use Case:** Used to start a Spring Boot application.

**@EnableAutoConfiguration**

* **Description:** Tells Spring Boot to automatically configure beans based on the dependencies present on the classpath.
* **Use Case:** Typically used in conjunction with @SpringBootApplication, but can also be used on its own.

A screen shot of a computer code

AI-generated content may be incorrect.

**@ComponentScan**

* **Description:** Tells Spring where to look for annotated components (@Component, @Service, @Repository, @Controller, etc.) to register them as beans in the application context. It enables automatic detection and wiring of components.
* **Use Case:** Commonly used to specify the base packages to scan when the component classes are not in the same package (or sub-packages) as the main application class. It can be customized with the basePackages attribute to control which packages are scanned.

**@Value**

* **Description:** Injects values from property files or environment variables into fields.
* **Use Case:** Useful for injecting simple property values into Spring beans.

A black background with white text

AI-generated content may be incorrect.