**核心注解**

**一、Spring Boot 应用入口注解**

| **注解** | **说明** |
| --- | --- |
| @SpringBootApplication | 标注在主类上，表示是 Spring Boot 应用的入口。是三个注解的组合：  ➡ @Configuration  ➡ @EnableAutoConfiguration  ➡ @ComponentScan |

1. **配置与组件注解**

| **注解** | **说明** |
| --- | --- |
| @Configuration | 表示当前类是一个配置类（可以替代 XML 配置） |
| @Bean | 声明一个 Bean 方法，返回的对象会被注入到 Spring 容器中 |
| @Component | 表示一个通用组件，交给 Spring 容器管理 |
| @Service | 表示一个服务类（业务逻辑层） |
| @Repository | 表示一个持久层类（DAO层） |
| @Controller | 表示一个控制器（用于 MVC） |
| @RestController | @Controller + @ResponseBody，用于返回 JSON 数据 |
| @ComponentScan | 指定包路径，扫描组件 |

1. **自动配置相关注解**

| **注解** | **说明** |
| --- | --- |
| @EnableAutoConfiguration | 启用 Spring Boot 的自动配置（Spring Boot 核心功能之一） |
| @ConditionalOnClass | 某个类存在时才加载配置 |
| @ConditionalOnMissingBean | 某个 Bean 不存在时才注入 |
| @ConditionalOnProperty | 某个配置属性存在/为特定值时才生效 |

1. **依赖注入与配置注入注解**

| **注解** | **说明** |
| --- | --- |
| @Autowired | 自动注入 Bean（构造函数、字段或 setter） |
| @Value | 注入配置文件中的值（如 ${server.port}） |
| @ConfigurationProperties | 将配置文件的内容绑定到 Java 对象上 |
| @Qualifier | 与 @Autowired 联用，指定注入哪个 Bean |

1. **Web 开发相关注解**

| **注解** | **说明** |
| --- | --- |
| @RequestMapping | 用于类或方法上，定义请求路径 |
| @GetMapping / @PostMapping / @PutMapping / @DeleteMapping | 快捷方式，对应 HTTP 方法 |
| @PathVariable | 获取 URL 路径参数 |
| @RequestParam | 获取请求参数（?后面的） |
| @RequestBody | 获取请求体中的 JSON 数据 |
| @ResponseBody | 将返回值直接写入 HTTP 响应体 |

1. **事务与数据库相关注解**

| **注解** | **说明** |
| --- | --- |
| @Transactional | 声明式事务管理，把一个方法标记为“事务性操作”：方法里的所有数据库操作，要么全部成功提交，要么出错时全部回滚。 |
| @EnableTransactionManagement | 开启基于注解的事务控制 |
| @Entity | 声明一个 JPA 实体类 |
| @Table | 映射数据库表名 |
| @Id / @GeneratedValue | 标记主键及主键生成策略 |

1. **测试相关注解**

| **注解** | **说明** |
| --- | --- |
| @SpringBootTest | 表示这是一个 Spring Boot 集成测试类 |
| @Test | JUnit 单元测试方法 |
| @MockBean | 创建一个 Mock 的 Bean 注入到容器中 |
| @DataJpaTest | 用于 JPA 测试，自动配置数据库相关环境 |

1. **其他常用注解**

| **注解** | **说明** |
| --- | --- |
| @EnableScheduling | 开启定时任务 |
| @Scheduled | 定义定时任务的方法 |
| @EnableAsync | 启用异步方法调用 |
| @Async | 将方法变为异步执行 |

**总结图表**

| **类别** | **常用注解** |
| --- | --- |
| 启动入口 | @SpringBootApplication |
| 配置类 | @Configuration, @Bean |
| 组件类 | @Component, @Service, @Repository, @Controller, @RestController |
| 自动配置 | @EnableAutoConfiguration, @ConditionalOn... |
| 注入 | @Autowired, @Value, @ConfigurationProperties |
| Web | @RequestMapping, @GetMapping, @RequestBody |
| 事务 | @Transactional, @EnableTransactionManagement |
| 测试 | @SpringBootTest, @Test, @MockBean |

详细说明：@Component

* 作用：将一个类标记为 Spring 管理的组件。
* 何时使用：用于未涵盖的通用 Spring 管理组件。@Service@Repository@Controller以外

@Service

* 作用：将一个类标记为服务层组件。
* 何时使用：用于定义服务层中的业务逻辑。

@Repository

* 作用：表示某个类是一个数据访问对象（DAO）。
* 何时使用：在持久层使用，与数据库交互。

@Configuration

* 作用：表明一个类包含Spring配置。
* 何时使用：用于定义 bean 配置和定制 Spring 上下文。

@Bean

* 作用：将方法标记为 Spring 管理 bean 的生产者。
* 何时使用：使用内部@Configuration带注释的类来定义 bean。

@Primary

* 作用：当存在多个相同类型的 bean 时，将某个 bean 标记为主 bean。
* 何时使用：当有多个相同类型的 bean，并且想要指定默认值时使用。

@Autowired

* 其功能：将依赖项注入到 Spring 管理的 bean 中。
* 何时使用：用于 Spring 管理的组件中的依赖注入。

@Qualifier

* 作用：当存在多个候选对象时，用于指定注入哪个 bean。
* 何时使用：使用with@Autowired来解决多个bean之间的歧义。

**Web 层注释**

@RestController

* 作用：@Controller@ResponseBody的组合，使得该类提供 RESTful Web 服务。
* 何时使用：在想要定义 REST 端点的类上使用。

@Controller

* 功能：将一个类标记为 Spring MVC 控制器。
* 何时使用：构建 Web 应用程序和处理 HTTP 请求时使用。

@RequestMapping

* 其作用：将 HTTP 请求映射到@Controller或的处理程序方法@RestController。
* 何时使用：用于定义控制器的端点映射。

@GetMapping

其作用：将 HTTP GET 请求映射到处理程序方法。

何时使用：用于获取数据等只读操作。

@PostMapping

其作用：将 HTTP POST 请求映射到处理程序方法。

何时使用：用于创建或提交数据。

@PutMapping

其作用：将 HTTP PUT 请求映射到处理程序方法。

何时使用：用于更新现有资源。

@DeleteMapping

其功能：将 HTTP DELETE 请求映射到处理程序方法。

何时使用：用于删除资源。

@PatchMapping

其功能：将 HTTP PATCH 请求映射到处理程序方法。

何时使用：用于资源的部分更新。

@PathVariable

其作用：从 URL 路径中提取一个值。

何时使用：当您想要将 URL 路径参数传递给方法时使用。

@RequestParam

其作用：从请求中提取查询参数。

何时使用：当您想要从 URL 中提取查询参数时使用。

@RequestBody

其作用：将 HTTP 请求的主体映射到 Java 对象。

何时使用：用于将请求数据（例如 JSON）绑定到方法参数。

@ResponseBody

作用：表示方法的返回值应该绑定到HTTP响应体。

何时使用：在非类中使用@RestController以返回 JSON 或其他格式。

@ResponseStatus

作用：用 HTTP 状态代码标记方法或异常类。

何时使用：用于设置自定义 HTTP 响应代码。

@ExceptionHandler

其作用：处理控制器方法中抛出的异常。

何时使用：在控制器类中使用来定义自定义错误处理。

@CrossOrigin

功能：为特定控制器或方法启用跨域资源共享 (CORS)。

何时使用：用于允许对您的 API 进行跨源请求。

**安全注解**

@EnableWebSecurity

功能：启用 Spring Security 的 Web 安全功能。

何时使用：在配置类上使用来定义身份验证和授权规则。

@Secured

其作用：根据角色限制对方法的访问。

何时使用：用于仅允许具有特定角色的用户访问。

@PreAuthorize

其作用：在调用方法之前应用更复杂的访问控制表达式。

何时使用：当访问控制需要超出角色的条件时使用，例如检查用户权限或属性。

@PostAuthorize

其作用：调用方法后应用访问控制表达式。

何时使用：当访问控制取决于方法的结果时使用。

@RolesAllowed

作用：按照 JSR-250 标准定义，指定允许访问的角色。

何时使用：用作基于角色的访问控制

的替代方案。@Secured

@EnableGlobalMethodSecurity

功能：启用方法级安全注释，如@Secured、@PreAuthorize和@PostAuthorize。

何时使用：在安全配置中使用以激活方法级安全性。

@WithMockUser

其功能：为测试目的设置模拟认证用户。

何时使用：在测试用例中使用，模拟经过身份验证的用户。

持久层注解

@Entity

作用：将一个类标记为 JPA 实体。

何时使用：用于将类映射到数据库表。

@Table

功能：自定义 JPA 实体中的表名和其他表属性。

何时使用：使用with@Entity自定义表映射。

@Id

作用：将某个字段标记为 JPA 实体中的主键。

何时使用：用于定义实体的主键字段。

@GeneratedValue

作用：指定如何生成主键值。

何时使用：使用 with@Id定义生成策略（例如，AUTO，IDENTITY）。

@Column

功能：定制 JPA 实体中字段的列映射。

何时使用：用于指定列属性，如名称、长度等。

@Transactional

其作用：把一个方法或类标记为“事务性操作”：方法里的所有数据库操作，要么全部成功提交，要么出错时全部回滚。

何时使用：在方法或类上使用以自动处理交易。