在Spring Boot中，依赖注入（Dependency Injection，简称DI）是核心功能之一，通常用来管理和注入对象实例。Spring框架支持多种方式来实现依赖注入，常见的三种方法是：

**1. 构造器注入（Constructor Injection）**

通过构造函数来注入依赖的对象。Spring会通过构造方法将需要的依赖注入到类的实例中。构造器注入是最推荐的注入方式，因为它是不可变的，确保依赖项在对象创建时就完全初始化。

示例代码：

@Component

public class MyService {

private final MyRepository myRepository;

@Autowired

public MyService(MyRepository myRepository) {

this.myRepository = myRepository;

}

// 使用myRepository的方法

}

解释：

* @Autowired 注解告诉Spring容器注入一个 MyRepository 实例。
* 构造函数注入在Spring中默认支持，@Autowired 可以省略（如果只有一个构造函数）。

**2. 字段注入（Field Injection）**

通过直接在字段上使用 @Autowired 注解来注入依赖对象。这是最简洁的方式，但由于它绕过了构造方法，容易导致测试不便，也不利于进行空值检查，所以在实际开发中不推荐过度使用。

示例代码：

@Component

public class MyService {

@Autowired

private MyRepository myRepository;

// 使用myRepository的方法

}

解释：

* 通过 @Autowired 注解，Spring会自动注入 MyRepository 实例。
* Spring会根据类型自动查找合适的依赖项来注入。

**3. Setter方法注入（Setter Injection）**

通过setter方法来注入依赖。即通过公开的setter方法将依赖对象传入类实例中。这种方式通常用于需要在对象实例化后设置依赖的场景。

示例代码：

@Component

public class MyService {

private MyRepository myRepository;

@Autowired

public void setMyRepository(MyRepository myRepository) {

this.myRepository = myRepository;

}

// 使用myRepository的方法

}

解释：

* 通过 @Autowired 注解在setter方法上，Spring会自动调用setter方法并注入依赖。
* setter方法注入允许在对象创建后修改依赖对象，适用于某些需要可变依赖的场景。

总结：

* 构造器注入 是推荐的方式，因为它保证了依赖关系的不可变性，并且有助于更好的单元测试。
* 字段注入 虽然简洁，但不利于测试和依赖性管理，因此不推荐在复杂项目中大量使用。
* Setter注入 适用于可变依赖的场景，可以在对象创建后注入依赖。