In Spring, the @Qualifier and @Primary annotations are used to resolve ambiguity when multiple beans of the same type are present in the application context. Here's a detailed look at why and when to use each:

**@Primary**

* **Purpose**: To designate a default bean when multiple beans of the same type exist.
* **Usage**: Annotate one of the beans with @Primary to indicate that it should be the default choice for autowiring.
* **Example**:

@Bean

@Primary

public MessageService emailService() {

return new EmailService();

}

@Bean

public MessageService smsService() {

return new SMSService();

}

In this example, if there is an autowiring conflict, emailService will be chosen by default because it is marked with @Primary.

**@Qualifier**

* **Purpose**: To specify exactly which bean should be injected when multiple beans of the same type are available.
* **Usage**: Use @Qualifier along with @Autowired to explicitly define which bean to inject.
* **Example**:

@Autowired

@Qualifier("smsService")

private MessageService messageService;

Here, smsService is explicitly chosen for injection, regardless of any @Primary annotations.

**When to Use Each**

* **Use @Primary**: When you want to set a default bean that should be used unless another specific bean is requested.
* **Use @Qualifier**: When you need precise control over which bean to inject, especially in cases where multiple beans of the same type are present and you need to differentiate between them.

**Combining @Primary and @Qualifier**

You can use both annotations together. @Qualifier takes precedence over @Primary. This means if you have a @Primary bean but also use @Qualifier to specify another bean, the @Qualifier bean will be injected.

**Example of Combined Use**

@Bean

@Primary

public MessageService emailService() {

return new EmailService();

}

@Bean

public MessageService smsService() {

return new SMSService();

}

@Autowired

@Qualifier("smsService")

private MessageService messageService;

In this example, even though emailService is marked as @Primary, the smsService will be injected because it is specified with @Qualifier.

**Summary**

* **@Primary**: Sets a default bean.
* **@Qualifier**: Provides precise control over bean selection.
* **Combination**: @Qualifier overrides @Primary when both are used.

These annotations help manage dependency injection more effectively, ensuring that the correct beans are injected where needed [[1]](https://www.javaguides.net/2023/07/spring-qualifier-vs-primary-annotations.html) [[2]](https://stackoverflow.com/questions/56642356/when-to-use-qualifier-and-primary-in-spring) [[3]](https://bing.com/search?q=why+to+use+Qualifier+and+primary+annotations).

Do you have a specific scenario or example in mind where you're considering using these annotations?

**References**

[1] [Difference Between Spring @Qualifier and @Primary Annotations - Java Guides](https://www.javaguides.net/2023/07/spring-qualifier-vs-primary-annotations.html)

[2] [When to use Qualifier and Primary in Spring - Stack Overflow](https://stackoverflow.com/questions/56642356/when-to-use-qualifier-and-primary-in-spring)

[3] [`@Primary` vs `@Qualifier` in Spring – Which to Use and When?](https://bing.com/search?q=why+to+use+Qualifier+and+primary+annotations)