Spring Boot中的条件装配主要以@Profile、@Conditional和一系列@ConditionalOnXXXX注解来实现，其中@ConditionalOnXXX就是Spring Boot中提供的@Conditional的多种预置特殊场景实现，提供了装配bean的多种特殊条件。

1. Spring Boot为应用程序提供了Profile这一概念，用来表示不同的环境，通过@Profile根据不同的环境装配不同的Bean，在运行时加上JVM参数”-Dspring.profiles.active”或在application.properties中加入spring.profiles.active指定运行环境。这种条件装配适用于区别不同场景下的环境配置，例如开发环境、测试环境和生产环境。
2. @ConditionalOnXXXX系列注解，例如：

@ConditionalOnBean:如果容器中有指定的Bean，条件生效；

@ConditionalOnClass:如果class path中有指定的类，条件生效，而不必要在容器中；

@ConditionalOnJava：判断当前jdk版本是否与注解中指定版本一致，满足指定Java版本时加载；

@onditionalOnExpression：根据表达式判断条件是否生效；

@ConditionalOnProperty，通过注解中的name和havingValue与配置文件判断条件是否满足；

1. 如果@ConditionalOnXXXX不能满足要求，需要根据一些更复杂的业务逻辑判断，那么可以使用@Conditional注解自定义判断规则。自定义的Condition类需要实现org.springframework.context.annotation.Condition接口，并且重写其matches（）方法，然后在@Conditional注解上指定该类。