Spring Professional Exam Tutorial v5.0

Question 16

Question 16 - How does Spring Boot know what to configure?

Spring Boot knows what to configure by usage of Auto Configuration Classes defined in starter modules. Spring Boot searches for `META-INF/spring.factories` on classpath, whenever entry `org.springframework.boot.autoconfigure.EnableAutoConfiguration` is encountered in this file, Auto Configuration Class pointed by this property is loaded.

Auto Configuration class is a regular `@Configuration` class annotated with `@ConditionalOn...` annotation which specifies under which conditions `@Configuration` class should be loaded.

When conditions from `@ConditionalOn...` annotation are matched, `@Configuration` class is loaded which provides beans that integrates your application with specified technology.

Auto Configuration is often used with starter modules. Starter module provides set of dependencies, and optionally may provide Auto Configuration classes.

Question 16 - How does Spring Boot know what to configure?

Spring Boot supports following Conditional Annotations for AutoConfiguration classes:

▶ `ConditionalOnBean` – **presence of Spring Bean**

▶ `ConditionalOnMissingBean` – **absence of Spring Bean**

▶ `ConditionalOnClass` – **presence of class on classpath**

▶ `ConditionalOnMissingClass` – **absence of class on classpath**

▶ `ConditionalOnCloudPlatform` – **if specified cloud platform is active – for example Cloud Foundry**

▶ `ConditionalOnExpression` – **if SpEL expression is true**

▶ `ConditionalOnJava` – **presence of Java in specified version**

▶ `ConditionalOnJndi` – **if JNDI location exists**

▶ `ConditionalOnWebApplication` – **if a web application that uses WebApplicationContext or StandardServletEnvironment**

▶ `ConditionalOnNotWebApplication` – **application that is not a web application**

▶ `ConditionalOnProperty` – **presence of spring property**

▶ `ConditionalOnResource` – **presence of resource**

▶ `ConditionalOnSingleCandidate` – **only one candidate for the bean found**