## Spring Boot and Web Annotations

Use annotations to configure your web application.

- **@SpringBootApplication** - uses @Configuration, @EnableAutoConfiguration and @ComponentScan.
- **@EnableAutoConfiguration** - make Spring guess the configuration based on the classpath.
- **@Controller** - marks the class as web controller, capable of handling the requests.
- **@RestController** - a convenience annotation of a @Controller and @ResponseBody.
- **@ResponseBody** - makes Spring bind method’s return value to the web response body.
- **@RequestMapping** - specify on the method in the controller, to map a HTTP request to the URL to this method.
- **@RequestParam** - bind HTTP parameters into method arguments.
- **@PathVariable** - binds placeholder from the URI to the method parameter.

## Spring Framework Annotations

Spring uses dependency injection to configure and bind your application together.

- **@Configuration** - used to mark a class as a source of the bean definitions.
- **@ComponentScan** - makes Spring scan the packages configured with it for the @Configuration classes.
- **@Import** - loads additional configuration. This one works even when you specify the beans in an XML file.
- **@Component** - turns the class into a Spring bean at the auto-scan time.
- **@Service** - tells Spring that it’s safe to manage @Components with more freedom than regular components.
- **@Autowired** - wires the application parts together, on the fields, constructors, or methods in a component.
- **@Bean** - specifies a returned bean to be managed by Spring context. The returned bean has the same name as the factory method.
- **@Lookup** - tells Spring to return an instance of the method’s return type when we invoke it.

- **@Primary** - gives higher preference to a bean when there are multiple beans of the same type.
- **@Required** - shows that the setter method must be configured to be dependency-injected with a value at configuration time.
- **@Value** - used to assign values into fields in Spring-managed beans. It’s compatible with the constructor, setter, and field injection.
- **@DependsOn** - makes Spring initialize other beans before the annotated one.
- **@Lazy** - makes beans to initialize lazily. @Lazy annotation may be used on any class directly or indirectly annotated with @Component or on methods annotated with @Bean.
- **@Scope** - used to define the scope of a @Component class or a @Bean definition and can be either singleton, prototype, request, session, globalSession, or custom scope.
- **@Profile** - adds beans to the application only when that profile is active.

### Legend:

- **T** - Class
- **F** - Field Annotation
- **C** - Constructor Annotation
- **M** - Method
- **P** - Parameter