Spring Boot Core Concepts

Bean, IoC, ApplicationContext, and Annotations

Bean in Spring :

A **Bean** in Spring is simply an **object** that is **managed by the Spring IoC container**.

- You can think of a bean as any Java class that Spring **creates**, **configures**, and **injects** where needed.

Define a Bean :

Using Stereotype Annotations

```
@Component
public class Laptop
{
    // this is now a Spring bean
}
```

Other stereotype annotations:

- `@Service` → for service/business logic layer
- `@Repository` → for database layer
- `@Controller` → for web controller (MVC)
- `@RestController` → combines `@Controller` + `@ResponseBody`

All these annotations automatically register the class as a **Spring bean**.

IoC (Inversion of Control)

Inversion of Control (IoC) means that the control of object creation and wiring is given to the Spring Framework instead of handling it manually in your code.

Instead of writing:

```
Laptop l = new Laptop(); // Tight coupling
```

We let Spring do:

```
@Autowired
Laptop l; // Spring injects it
```

ApplicationContext

`ApplicationContext` is the **Spring container** that:

- Manages the complete **lifecycle of beans**
- Performs **dependency injection**
- Handles **bean scopes**, **events**, **internationalization**, etc.

It **reads configuration** (from annotations or XML) and **initializes all beans**.

Important Annotations for Beans and IoC

Annotation	Purpose
`@Component`	Declares a generic bean
`@Service`	Declares a service-layer bean
`@Repository`	Declares a DAO/data-layer bean
`@Controller`	For MVC controller
`@RestController`	For REST API controllers
`@Bean`	Used inside a `@Configuration` class to declare manual beans
`@Configuration`	Declares a class that provides Spring bean definitions
`@Autowired`	Injects dependencies automatically
`@ComponentScan`	Tells Spring where to look for beans (packages)
`@SpringBootApplication`	Combines `@Configuration`, `@EnableAutoConfiguration`, and `@ComponentScan`