Spring Basics Expanded

Page 1: Introduction to Spring Framework

Spring is an open-source, lightweight framework widely used for building Java applications.

Features:

- Lightweight: Minimal memory consumption.

- Modular: Offers modular architecture.

- Dependency Injection (DI): Simplifies dependency management.

- Integration: Easily integrates with other frameworks.

Why use Spring?

- Simplifies development of enterprise-grade applications.

- Provides excellent support for REST APIs.

Page 2: Core Spring Modules

1. Spring Core:

- Manages the application's configuration using XML or Java annotations.

2. Spring MVC:

- Builds web applications with an MVC architecture.

3. Spring Boot:

- Simplifies project setup with embedded servers and default configurations.

Additional Modules:

- Spring Data, Spring Security, Spring Cloud.

Page 3: Dependency Injection (DI)

DI is a design pattern in which objects receive their dependencies at runtime.

Types of DI:

- Constructor Injection
- Setter Injection

Example:

```
public class TextEditor {
    private SpellChecker spellChecker;
    public TextEditor(SpellChecker spellChecker) {
        this.spellChecker = spellChecker;
    }
}
```

Page 4: Spring Beans and Lifecycle

A Bean is an object managed by Spring's IoC container.

Bean Lifecycle:

- Instantiate: Create an instance.
- Populate properties: Set the object's properties.
- Initialization: Call initialization methods.
- Destruction: Cleanup resources.

Example:

```
<bean id="exampleBean" class="com.example.MyClass">
    <property name="propertyName" value="value"/>
</bean>
```

Page 5: Introduction to Spring Boot

Spring Boot simplifies application setup by:

- Providing default configurations.
- Embedding servers like Tomcat.
- Packaging the application into a single JAR file.

Example REST API:

```
@RestController
public class GreetingController {
    @GetMapping("/greeting")
    public String greet() {
        return "Hello, Spring Boot!";
    }
}
```