010 Section 010 Understanding Component Scanning in Spring Boot

Understanding Component Scanning in Spring Boot

1. What is Component Scanning?

- Component scanning is a mechanism in Spring Boot that automatically detects and registers beans.
- It looks for classes annotated with:
  - `@Component`
  - `@Service`
  - `@Repository`
  - `@Controller`
- The scanning is controlled by `@ComponentScan`, which is included in `@SpringBootApplication`.

2. How Does Component Scanning Work?

- The main application class is annotated with `@SpringBootApplication`.
- By default, Spring Boot scans **only the package of the main class** and all its sub-packages.
- Any class defined outside this package will **not** be detected unless explicitly included.

3. Example Scenarios

Case 1: Components Inside the Base Package (Scanned Automatically)

Application Class (Inside Base Package)

```
@SpringBootApplication
public class MyApplication {
    public static void main(String[] args) {
        SpringApplication.run(MyApplication.class, args);
    }
}
```

Component Class (Inside Base Package)

```
@Component
public class MyComponent {
    public void doSomething() {
        System.out.println("Component is working!");
    }
}
```

✅ `MyComponent` **will be detected** because it is inside the package of `MyApplication`.

Case 2: Components Outside the Base Package (Not Scanned Automatically)

External Component (Outside Base Package)

```
package com.example.service;  // Outside the main package

import org.springframework.stereotype.Component;

@Component
```

```
public class ExternalComponent {
    public void execute() {
        System.out.println("External component executing...");
    }
}
```

❌ `ExternalComponent` **will not be scanned** because it is outside the base package.

4. How to Scan Components Outside the Base Package?

Solution 1: Using `@ComponentScan`

If you want to scan additional packages, use `@ComponentScan`:

```
@SpringBootApplication
@ComponentScan(basePackages = {"com.example", "com.external"})  // Add extra packages
public class MyApplication {
    public static void main(String[] args) {
        SpringApplication.run(MyApplication.class, args);
    }
}
```

✅ Now, Spring Boot will scan both `com.example` (default) and `com.external` packages.

Solution 2: Using `@Import`

Another way to register beans from an external package is by using `@Import`:

```
@Import(ExternalComponent.class)
public class MyConfig {
}
```

✅ This manually registers `ExternalComponent` without modifying `@ComponentScan`.

5. Key Takeaways

- `@SpringBootApplication` includes `@ComponentScan`, which scans the **current package and sub-packages**.
- **Classes outside this package won't be detected** unless explicitly specified using `@ComponentScan`.
- To include additional packages, use `@ComponentScan(basePackages = {"package1", "package2"})`.
- Alternatively, use `@Import(ClassName.class)` to register specific classes manually.

This ensures efficient management of Spring Boot components and their dependencies!