SME Walkthrough Topics – Explanation

---

1. src/main/java

- This folder contains **all your application source code**.
- Under it, your **base package** com.cognizant.springlearn has:
    - **SpringLearnApplication.java** (main class)
    - Other classes like Controllers, Services, Repositories as you build the project.

---

2. src/main/resources

- Contains **configuration files** for your application.
- Currently it has:
    - **application.properties** – used for defining application configurations (e.g. server port, database settings).
- It is also where you keep:
    - Static files, templates, and messages if you add them later.

---

3. src/test/java

- Contains **test classes** for unit and integration testing.
- Initially it has:

SpringLearnApplicationTests.java

- Uses **JUnit** to test if the context loads successfully.

---

4. SpringLearnApplication.java

Explain:

- This is the **entry point** of the Spring Boot application.
- It contains the **main() method**, which:

java

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

This method **bootstraps the Spring Boot application** by calling SpringApplication.run().

---

5. Purpose of @SpringBootApplication annotation

Explain:

- @SpringBootApplication is a **combination annotation** that includes:

@Configuration

@EnableAutoConfiguration

@ComponentScan

Enables Spring Boot's auto-configuration and component scanning in the current package and sub-packages.

---

6. pom.xml

How to walk through pom.xml clearly:

1. **Project metadata**
    - GroupId: com.cognizant
    - ArtifactId: spring-learn
    - Version, packaging, name
2. **Parent**
    - Defines the Spring Boot parent starter for dependency management.
3. **Dependencies**
    - **Spring Boot Starter Web**
        - Includes Spring MVC, Tomcat embedded server.
    - **Spring Boot DevTools**
        - Enables automatic restart and live reload for development.
4. **Build plugins**
    - Spring Boot Maven plugin for packaging the app as an executable jar.

---

7. Dependency Hierarchy

How to check in Eclipse:

1. Double-click **pom.xml**.

2. Go to **Dependency Hierarchy tab (bottom panel)**.
3. Expand each dependency to show **direct and transitive dependencies**.

Explain:

- This shows **all libraries included by your dependencies**, including nested jars brought by Spring Boot starters.

Final Clear Explanation Summary for SME

Created using Spring Initializr with **Group, Artifact, and dependencies**
Extracted, built, and imported into Eclipse
src/main/java: Application source code
src/main/resources: Config files
src/test/java: Unit tests
SpringLearnApplication.java: Entry point with main()
@SpringBootApplication: Combines configuration, auto-config, component scan
pom.xml: Project metadata, dependencies, build plugins
Dependency Hierarchy: Shows entire dependency tree of the project