1. **Differentiate between the different dependency scopes: compile, runtime, test, provided using different dependencies being defined in your pom.xml.**

**Ans:-**

**Compile**:  
This is the default scope, used if none is specified. Compile dependencies are available in all classpaths of a project. Furthermore, those dependencies are propagated to dependent projects.

<dependencies>

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.14</version>

<!-- You can omit this because it is default -->

**<scope>compile</scope>**

</dependency>

</dependencies>

**Test**:

This scope indicates that the dependency is not required for normal use of the application, and is only available for the test compilation and execution phases.

<dependencies>

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.14</version>

**<scope>test</scope>**

</dependency>

</dependencies>

**Runtime**:

This scope indicates that the dependency is not required for compilation, but is for execution. It is in the runtime and test classpaths, but not the compile classpath.

<dependencies>

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.14</version>

**<scope>runtime</scope>**

</dependency>

</dependencies>

**Provided**:

This is much like compile, but indicates you expect the JDK or a container to provide the dependency at runtime. For eg:- when you run an application on a server some of the dependencies are provided implicitly by the server itself

<dependencies>

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.14</version>

**<scope>provided</scope>**

</dependency>

</dependencies>