Gerenciando dependências

Tipos de dependências

- **Direta**: dependências declaradas no pom.xml
- **Transitiva**: dependências obrigatórias das dependências declaradas no pom.xml

Transitividade e Escopos

Escopos

O Maven provê escopos para limitar a transitividade das dependências.
Existem 6 tipos de escopos que podemos utilizar.

Classpath

- Runtime
- Test
- Compile

Escopo compile

- Escopo default
- Disponível em todos os classpaths
- É transitivo

Escopo provided

- Indica que a dependência será fornecida em tempo de execução por uma implementação na JDK ou via container
  - Exemplos: Servlet API, Java EE APIs
- A dependência com esse escopo é adicionado no classpath usado para compilação(compile) e teste(test) mas não em runtime;
- Não é transitiva

```
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>javax.servlet-api</artifactId>
  <version>3.1.0</version>
  <scope>provided</scope>
</dependency>
```

Escopo runtime

- Indica que a dependência é necessária para execução e não para compilação
- Maven inclui no classpath dos escopos de runtime e test

```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>6.0.6</version>
    <scope>runtime</scope>
</dependency>
```

Escopo test

- Disponível somente para compilação e execução de testes
- Não é transitivo

```
<dependency>
    <groupId>org.junit.jupiter</groupId>
    <artifactId>junit-jupiter-engine</artifactId>
    <version>${junit.jupiter.version}</version>
    <scope>test</scope>
</dependency>
```

Escopo system

- Similar ao escopo provided exceto por ser necessário prover o JAR explicitamente
- A dependência com esse escopo é adicionado no classpath usado para compilação(compile) e teste(test) mas não em runtime;
- Não é transitiva

```
<dependency>
    <groupId>com.baeldung</groupId>
    <artifactId>custom-dependency</artifactId>
    <version>1.3.2</version>
    <scope>system</scope>
    <systemPath>${project.basedir}/libs/custom-dependency-1.3.2.jar</systemPath>
</dependency>
```

Escopo import

- Este escopo é disponível apenas com uma dependência do tipo pom e com tag <dependencyManagement>
- Indica um processo de reutilizar dependências de um projeto

```
<dependencyManagement>
    <dependencies>
        <dependency>
            <groupId>com.programmergirl</groupId>
            <artifactId>my-project</artifactId>
            <version>1.1</version>
            <type>pom</type>
            <scope>import</scope>
        </dependency>
    </dependencies>
</dependencyManagement>
```

Dica sobre escopos, dependências opcionais e exclusões

Ver o classpath

Comando:

- mvn dependency:build-classpath –DincludeScope=compile
- mvn dependency:build-classpath –DincludeScope=test
- mvn dependency:build-classpath –DincludeScope=runtime

Dependências Opcionais

- Utilizado quando uma dependência não é necessária para os projetos que irão reutilizar seu componente

```
<dependency>
    <groupId>com.google.code.gson</groupId>
    <artifactId>gson</artifactId>
    <version>2.8.8</version>
    <optional>true</optional>
</dependency>
```

Exclusion

- Utilizado quando o componente que você usa compartilha uma biblioteca que você já tem ou não quer ter disponível

```
<dependency>
    <groupId>one.digitalinnovation</groupId>
    <artifactId>component</artifactId>
    <version>1.0.0-SNAPSHOT</version>
    <exclusions>
        <exclusion>
            <groupId>com.google.code.gson</groupId>
            <artifactId>gson</artifactId>
        </exclusion>
    </exclusions>
</dependency>
```