**POM.XML:**

**build Angular frontend resources**

The provided Maven plugin configuration in the **pom.xml** file is for the **exec-maven-plugin**. This plugin allows you to execute arbitrary commands during the Maven build lifecycle. In this specific case, it is used to install and build the frontend part of an Angular application before packaging it with a Spring Boot application.

Let's break down the configuration:

1. **Plugin Declaration:**
   * **<groupId>**, **<artifactId>**, and **<version>**: These specify the coordinates of the Maven plugin.
   * **<executions>**: This section contains one or more executions, each representing a set of commands to be executed at a specific phase of the Maven build lifecycle.
2. **Execution for Front-end Install:**
   * **<id>**: A unique identifier for this execution.
   * **<goals>**: Specifies the Maven goals associated with this execution. In this case, the goal is **exec**.
   * **<phase>**: The Maven build phase at which this execution should occur. Here, it's set to **prepare-package**.
   * **<configuration>**: Configures the parameters for the execution.
     + **<executable>**: The executable to run. In this case, it's **npm**.
     + **<arguments>**: Specifies the arguments to pass to the executable. Here, it's **install**.
3. **Execution for Front-end Build:**
   * Similar to the first execution but with different parameters.
   * **<arguments>**: In this case, it's **run build**, indicating that the **npm run build** command will be executed.
4. **Common Configuration:**
   * **<configuration>** outside of **<executions>** is used to set configuration options that apply to all executions.
   * **<workingDirectory>**: Sets the working directory for the executions. In this case, it's set to **${basedir}/frontend**, meaning the commands will be executed in the **frontend** directory relative to the project's base directory.

In summary, this Maven configuration is defining two executions for the **exec-maven-plugin**. The first execution installs frontend dependencies using **npm install**, and the second execution builds the frontend using **npm run build**. These commands are run during the **prepare-package** phase of the Maven build lifecycle, ensuring that the frontend is installed and built before the Spring Boot application is packaged. The working directory for these commands is set to the **frontend** directory within the project

Copy Angular resources to Spring Boot resource files

1. **Copy Front-end Assets Execution (to src/main/resources/static):**
   * **<id>**: A unique identifier for this execution.
   * **<goals>**: Specifies the Maven goals associated with this execution. In this case, the goal is **copy-resources**.
   * **<phase>**: The Maven build phase at which this execution should occur. Here, it's set to **prepare-package**.
   * **<configuration>**: Configures the parameters for the execution.
     + **<outputDirectory>**: Specifies the target directory where resources should be copied. In this case, it's **${basedir}/src/main/resources/static**.
     + **<resources>**: Specifies the resources to be copied.
       - **<resource>**: Defines a resource to be copied.
         * **<directory>**: Specifies the source directory of the resources. In this case, it's **frontend/dist/frontend**.
2. **Copy Front-end Assets Execution (to target/classes/static):**
   * Similar to the first execution but with a different target directory.
   * **<outputDirectory>**: This time, it's set to **${basedir}/target/classes/static**.

In summary, these executions are copying the frontend assets (presumably built Angular application files) from **frontend/dist/frontend** to two different locations within the Spring Boot project. One set of assets is copied to **src/main/resources/static**, and the other set is copied to **target/classes/static**. These directories are common locations where static resources are served from in a Spring Boot application. This ensures that the frontend assets are included in the final packaged JAR or WAR file and can be served by the Spring Boot application.