**Maven has three built-in lifecycles:**

1. **default lifecycle** – Handles project deployment.
2. **clean lifecycle** – Handles project cleaning.
3. **site lifecycle** – Handles project site generation.

**1. Default Lifecycle (Main Lifecycle)**

The default lifecycle is the core of the Maven build process and consists of the following phases:

* **validate**: Validates that the project is correct and all required information is available.
* **compile**: Compiles the source code of the project.
* **test**: Runs the unit tests of the project.
* **package**: Packages the compiled code into a distributable format (like JAR, WAR, etc.).
* **verify**: Verifies that the package is valid and meets quality standards (e.g., runs integration tests).
* **install**: Installs the packaged artifact into the local repository for use as a dependency in other projects.
* **deploy**: Copies the final artifact to the remote repository for sharing with other developers and projects.

**2. Clean Lifecycle**

The clean lifecycle is responsible for cleaning up the project and removing any previously generated files, such as compiled files and artifacts.

The phases in the clean lifecycle are:

* **pre-clean**: Executes before the clean phase, allowing custom tasks.
* **clean**: Deletes the target/ directory where build artifacts (e.g., JAR, WAR) are stored.
* **post-clean**: Executes after the clean phase, allowing custom tasks.

**3. Site Lifecycle**

The site lifecycle is responsible for generating project documentation.

The phases in the site lifecycle are:

* **pre-site**: Executes before generating the site.
* **site**: Generates the project’s site (documentation) based on the site plugin.
* **post-site**: Executes after generating the site.
* **site-deploy**: Deploys the generated site to a server.

**1. Group ID**

* The **Group ID** is a unique identifier for a group or organization that is associated with the project.
* It's typically structured in reverse domain name notation (e.g., com.example, org.apache).

**2. Artifact ID**

* The **Artifact ID** is the name of the specific project or module that is being built or referenced.
* It should be unique within a given **Group ID**.
* The **Artifact ID** can be thought of as the name of the final product (e.g., a JAR file, WAR file, etc.).

<artifactId>my-awesome-library</artifactId>

Org.springframework.beans.factory.annotation.Autowired