### 메이븐 프로젝트 구성

이 절에서는 [pom.xml](https://github.com/logpresso/logpresso-app-examples/blob/main/logpresso-sample-app/pom.xml) 구성을 항목별로 설명합니다. 전체 내용은 소스를 참고하시기 바랍니다.

#### 아티팩트 정보

<groupId>com.logpresso.sonar</groupId><artifactId>logpresso-sample-app</artifactId><version>1.0.2306.0</version><packaging>bundle</packaging><name>Logpresso Sample App</name>

로그프레소 앱의 그룹 ID는 항상 com.logpresso.sonar로 지정합니다. 아티팩트 ID는 logpresso-VENDOR-MODEL 형식으로 정의하는 것을 권장합니다. 버전은 Major.Minor.YYMM.REV 형식으로 정의합니다. 앱 버전에 배포 시점을 표기하는 것은 호환되는 로그프레소 플랫폼을 추정하기 수월하기 때문입니다.

마지막으로 패키징이 bundle로 지정된 것을 특히 주목해야 합니다. 아래에 설명할 maven-bundle-plugin 플러그인이 bundle 패키징을 수행합니다.

#### 환경변수 설정

<properties> <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding> <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding> <jdk.version>11</jdk.version></properties>

운영체제 로케일이 빌드에 영향을 미치지 않도록 파일 인코딩을 UTF-8로 명시하고, JDK 버전을 11로 지정합니다. jdk.version 변수는 아래에서 사용됩니다.

#### 메이븐 저장소

<repositories> <repository> <id>logpresso-repo</id> <name>Logpresso Maven Repository</name> <url>https://maven.logpresso.com/</url> </repository></repositories><pluginRepositories> <pluginRepository> <id>logpresso-plugin-repo</id> <name>Logpresso Maven Repository</name> <url>https://maven.logpresso.com/</url> </pluginRepository></pluginRepositories>

로그프레소는 https://maven.logpresso.com 저장소에서 서드파티 개발에 필요한 라이브러리를 제공합니다. 빌드 단계에서 바이트코드 변경에 필요한 iPOJO 플러그인을 다운로드 받을 수 있도록 <pluginRepository> 설정을 추가하고, 앱 API 및 iPOJO 라이브러리를 다운로드 받을 수 있도록 <repository> 설정을 추가합니다.

#### 빌드 설정

자바 컴파일 시 소스 코드 및 바이트 코드 버전, 디버그 정보 포함 여부, 최적화 여부를 지정합니다.

<plugin> <groupId>org.apache.maven.plugins</groupId> <artifactId>maven-compiler-plugin</artifactId> <version>3.8.1</version> <configuration> <encoding>UTF-8</encoding> <source>${jdk.version}</source> <target>${jdk.version}</target> <debug>true</debug> <optimize>true</optimize> <showDeprecations>true</showDeprecations> </configuration></plugin>

maven-bundle-plugin 플러그인은 JAR 파일에 OSGi 매니페스트를 추가하여 OSGi 번들을 생성합니다. OSGi 번들은 META-INF/MANIFEST.MF 파일에 OSGi 매니페스트가 추가된 JAR 파일을 의미합니다.

<plugin> <groupId>org.apache.felix</groupId> <artifactId>maven-bundle-plugin</artifactId> <version>5.1.4</version> <extensions>true</extensions> <configuration> <instructions> <Bundle-SymbolicName>com.logpresso.sonar.sample</Bundle-SymbolicName> <Export-Package> com.logpresso.sonar.sample;version=${project.version}, </Export-Package> <Import-Package> org.json;version="1.1.0", org.araqne.codec;version="2.2", org.araqne.log.api;version="3.13.0", org.logpresso.api.profile;version="1.1.0", org.logpresso.api.profile.query;version="1.1.0", org.araqne.logdb;version="3.10.0", org.araqne.msgbus;version="1.12.0", com.logpresso.sonar.api.\*;resolution:=optional, \* </Import-Package> <Private-Package> com.logpresso.sonar.sample.impl, com.logpresso.sonar.sample.query, </Private-Package> </instructions> </configuration></plugin>

<instruction>의 4개 항목을 살펴봐야 합니다:

* Bundle-SymbolicName: 번들 식별자를 정의합니다. 자바 패키지명 규칙에 맞춰서 작성합니다.
* Export-Package: 다른 OSGi 번들에게 공개할 패키지 목록을 입력합니다. 일반적으로 인터페이스가 포함된 패키지만 공개합니다. ${project.version} 매크로를 이용하여 공개하는 패키지의 버전을 명시합니다.
* Import-Package: 다른 OSGi 번들에서 가져올 패키지 목록을 입력합니다. 패키지 뒤에 ;version="VERSION"을 정의하여 최소 버전을 명시합니다. 버전을 명시하지 않으면 빌드 플러그인이 <dependencies>에 정의된 메이븐 아티팩트의 버전을 사용하기 때문에, 호환되는 패키지 버전 범위가 의도와 다르게 지정되어 번들 의존성이 해소되지 않을 수 있습니다.
* Private-Package: 다른 OSGi 번들에게 공개하지 않을 내부 패키지 목록을 입력합니다. 일반적으로 구현이 포함된 패키지는 공개하지 않습니다. 외부에는 인터페이스만 공개해야 구현 세부내용을 언제든지 쉽게 변경할 수 있습니다.

Import-Package에는 로그프레소 앱 API 호출에 필요한 패키지의 버전이 미리 정의되어 있습니다. sonar-app-api 라이브러리 버전(현재 4.0.2306.0)에 따라 불러올 수 있는 패키지 버전의 집합은 변경될 수 있습니다.

<plugin> <groupId>org.apache.felix</groupId> <artifactId>maven-ipojo-plugin</artifactId> <version>1.12.1.asm8</version> <executions> <execution> <goals> <goal>ipojo-bundle</goal> </goals> </execution> </executions></plugin>

iPOJO는 선언적으로 OSGi 컴포넌트를 개발할 수 있도록 만들어주는 프레임워크입니다. OSGi는 런타임 플러그인을 가정하기 때문에, 각 번들이 의존하는 기능이 언제든지 새로 설치되거나 삭제될 수 있습니다. 이 때문에 OSGi 인터페이스만을 사용하여 개발하려면 모든 상태 변경에 대해 통지를 받아 연쇄적으로 기능을 비활성화하거나, 활성화해야 해서 구현이 매우 복잡해집니다. iPOJO는 컴포넌트 생명주기 모델을 정의하고 간단한 어노테이션 추가만으로 OSGi 서비스를 선언적으로 개발할 수 있도록 지원합니다.

iPOJO는 빌드 단계에서 iPOJO 어노테이션을 인식하여 바이트코드를 삽입하는 방식으로 자동 변환을 수행합니다. 따라서 반드시 위와 같이 빌드 단계에서 maven-ipojo-plugin이 호출되어야 합니다.

#### 라이브러리 의존성

<dependencies> <dependency> <groupId>junit</groupId> <artifactId>junit</artifactId> <version>4.8.1</version> <scope>test</scope> </dependency> <dependency> <groupId>org.apache.felix</groupId> <artifactId>org.apache.felix.ipojo.annotations</artifactId> <version>1.12.1.asm8</version> </dependency> <dependency> <groupId>org.slf4j</groupId> <artifactId>slf4j-api</artifactId> <version>1.7.12</version> </dependency> <dependency> <groupId>com.logpresso.sonar</groupId> <artifactId>sonar-app-api</artifactId> <version>4.0.2306.0</version> </dependency></dependencies>

로그프레소 앱은 아래의 라이브러리를 사용합니다:

* junit: 단위 테스트 작성에 사용합니다. 단위 테스트 클래스는 src/test/java 디렉터리 아래에 배치합니다.
* org.apache.felix.ipojo.annotations: iPOJO 컴포넌트 어노테이션을 사용합니다.
* slf4j-api: 시스템 로그 기록에 SLF4J를 사용합니다. 로그는 log/araqne.log 파일에 기록되며, 로그 레벨을 런타임에 조정할 수 있습니다.
* sonar-app-api: 로그프레소 앱 API를 사용합니다. 이 라이브러리 버전은 호환되는 로그프레소 플랫폼 버전을 의미합니다.