Building OSGi projects with Maven

Ray Augé (Liferay)
Tim Ward (Paremus)

The bnd-maven-plugin

What?	The core bnd plugin for Maven, integrates bnd into the maven lifecycle.
Why?	Generates a bundle manifest and other OSGi metadata based on annotations in your code and (optional) configuration data
When?	Apply to the process-classes phase (the default) so that bnd can inspect your classes for annotations and dependencies.
Where?	Everywhere, even in projects that are used outside of OSGi! Remember that an OSGi bundle is still a valid JAR file.

The bnd-maven-plugin (detail)

- Bnd processes your class files (not source) to determine package dependencies
  - No manual maintenance of Import-Package
- Bnd processes all OSGi standard annotations
  - Generates DS/Metatype XML from the annotations
  - Adds Manifest Headers declared by annotations
- For libraries, Bnd provides and processes ServiceLoader annotations and can generate JPMS module info

The bnd-indexer-maven-plugin

What?	A utility plugin for Maven, uses bnd to generate an OSGi repository index
Why?	Converts Maven dependencies into a standard form that can be used by other tools when resolving and deploying
When?	Can be applied in any phase, but is usually *after* the package phase so that the local artifact can be included in the index
Where?	Apply to any project where an index is a useful intermediate, or a desired output
Often Used With	resolver, testing, export and run plugins

The bnd-indexer-maven-plugin (detail)

- Indexer produces an XML index of bundles
  - Can use your Maven Dependencies
  - Can index a folder (maven-dependency-plugin)
- Indexes may be “attached” to the project
  - Allows them to be deployed to Nexus
- Indexes can enforce “local” or “remote” URLs
  - Local URLs are quick and easy for in-build use
  - Remote URLs are necessary for external use

The bnd-resolver-maven-plugin

What?	A utility plugin for Maven, enables the use of the OSGi resolver
Why?	Lets users automatically calculate the list of deployed bundles in a bndrun based on a set of initial requirements
When?	Supports multiple executions, often not bound to a phase, but run from the Maven CLI directly
Where?	Any project that contains a bndrun file
Often Used With	Indexer, export, testing and run plugins

The bnd-resolver-maven-plugin (detail)

- The OSGi resolver attempts to solve requirement graphs
  - Inputs are “initial requirements” in the bndrun
  - Capabilities and Requirements come from a repository
- The result of the resolution is saved to the bndrun
- Can use “implicit” repository with configurable scopes

The bnd-export-maven-plugin

What?	A packaging plugin for Maven, uses bnd to generate an executable application JAR
Why?	Lets users launch an OSGi framework containing their bundles, started using the bnd project launcher
When?	Usually in the package phase
Where?	Supports multiple executions to export using more than one approach
Often Used With	Indexer and resolver plugins

The bnd-export-maven-plugin (detail)

- The exporter consumes one or more bndrun files
  - Contains a list of bundles to deploy
  - Identifies the OSGi framework implementation
  - Bundles and frameworks must be available from a “repository”
- Maven dependencies can be an “implicit” repository
  - Quick and easy, limited control of scope
- Indexes from the indexer can also be repositories
  - A little more work to set up, but very fine-grained control

The bnd-testing-maven-plugin

What?	A testing plugin for Maven, enables the execution of OSGi bundle tests
Why?	Lets users test bundles in a running OSGi framework
When?	Usually bound to the integration-test phase
Where?	A specific OSGi testing project that creates a tester bundle and has one or more testing bndrun files
Often Used With	Indexer and resolver plugins

The bnd-testing-maven-plugin (detail)

- This plugin runs tests in an OSGi framework
  - Tests use Junit and packaged in a bundle
  - Tests listed using a **Test-Cases** header
- The framework is defined using a bndrun file
  - Just like the ones used by the export plugin
  - These bndruns can also be resolved
- These tests should validate bundle operation
  - Not a substitute for Unit testing!

The bnd-run-maven-plugin

What?	A utility plugin for Maven, used to launch an OSGi framework process
Why?	Useful for development, integration testing, and continuous deployment
When?	Useful for live coding from any IDE that can build jars on demand
Where?	Not bound to a lifecycle, run from the cli, usually only one project per Maven build, as the output is the runnable application
Often Used With	Indexer and resolver plugins

The bnd-run-maven-plugin (detail)

- This plugin accepts and executes a bndrun in a process in the foreground
- While executing, any modifications trigger updates
  - Changes to the bndrun/pom.xml may cause the deployed bundles to change
  - Modified project outputs are automatically updated in the running framework

The bnd-baseline-maven-plugin

What?	A validation plugin for Maven, enables enforcement of API semantic versioning
Why?	Automates API versioning policy enforcement
When?	Usually bound to the verify phase
Where?	Any OSGi bundle, but crucially any OSGi bundle which exports API
Often Used With	Bnd maven plugin

The bnd-baseline-maven-plugin (detail)

- This plugin compares OSGi bundles
  - One bundle is the output of the current build
  - The other “baseline” bundle is found from Nexus
- The baseline can be found automatically
  - The highest version lower than the one to compare
  - The baseline can be configured to point at a different coordinate if needed
- Report detail is configurable

The bnd-reporter-maven-plugin

What?	Generate high level reports about your OSGi bundles
Why?	Simplify the task of documenting myriad OSGi details
When?	Whenever you need to generate documentation about OSGi bundles
Where?	Not bound to a lifecycle, execute from the cli
Often Used With	Bnd maven plugin

The bnd-reporter-maven-plugin (detail)

- This plugin is used to generate documentation for your projects
- It uses a template system to enable rich documentation generation
- It uses a two step process
  - Step 1: generates the project metadata into an intermediate format (json or xml)
  - Step 2: converts the metadata using templates
- The system is plugable for extracting arbitrary information

So what about Bndtools?

- The Maven plugins from bnd are IDE agnostic
  - Everything should work from the command line
  - Whatever support for Maven your IDE has will work
- Bndtools *does* still have value, but isn't required
  - Editors for bnd/bndrun
  - Built in resolve buttons
  - M2E bindings for bnd plugins
  - Location appropriate error markers

Bootstrapping your own project

- Setting up a new Maven build can be hard
  - There is documentation for all the bnd plugins
  - A lot of bits to get right
- OSGi enRoute provides easy startup
  - Archetypes for top-level builds and maven projects
  - Useful dependencies gathered into boms
  - All using bnd’s Maven plugins

Demo

Reference

EnRoute

https://enroute.osgi.org/

Bnd Maven Plugins

https://github.com/bndtools/bnd/blob/master/maven/README.md


eclipsecon Europe 2019
OSGi™ Community Event 2019
LUDWIGSBURG, GERMANY | OCTOBER 21 - 24, 2019
EVALUATE THE SESSIONS
Sign in and vote using the conference app or eclipsecon.org
-1 0 +1

Reference

EnRoute

https://enroute.osgi.org/

Bnd Maven Plugins

https://github.com/bndtools/bnd/blob/master/maven/README.md


eclipsecon Europe 2019
OSGi Community Event 2019
LUDWIGSBURG, GERMANY | OCTOBER 21 - 24, 2019
EVALUATE THE SESSIONS
Sign in and vote using the conference app or eclipsecon.org
-1 0 +1