nuxeo | Professional Open Source ECM

Nuxeo 5 Tutorial

Packaging Structure

Last Modification	2007-01-26
Project Code	NUXTUT
Document ID	PACKAGING
Copyright	Copyright © 2007 Nuxeo. All Rights Reserved.

Versioning

Version	Date	Participant(s)	Comments
0.1	2007-01-24	• Florent Guillaume	First draft

Contents

1 Introduction

Nuxeo 5 components are used to extend the base Nuxeo 5 framework and provide application logic, document types, user interface changes, etc.

Because of the way these components are packaged, they are often called “bundles”. This is the term used by OSGi to describe a self-contained component and the meta-information necessary to its deployment and activation. Nuxeo 5 uses OSGi as its component model.

The English term “component” can refer to a bundle, or to a more abstract entity (contained inside a bundle) that is the basis of the Nuxeo 5 plugin architecture. The term “NXRuntime component” always refers to the Nuxeo 5 meaning. The term “component” is therefore a bit ambiguous, but its meaning is generally clear from the context.

Creating and using a Nuxeo 5 component involves several concrete aspects: the source files holding the code and resources, the Jar file used for distribution, and the final layout after the Jar has been deployed by its container.

The source file organization is based on a standard layout that helps development, testing and Jar building.

The distributable bundle, a Jar file, has to contain all the information needed for the Nuxeo 5 framework to find an deploy it correctly, and let it interact with other bundles and components.

The following describes each of these aspects, illustrating them with examples.

2 Development files structure

The following is a typical structure for the development files of a Nuxeo 5 component:

- .classpath
- .project
- META-INF/
  - MANIFEST.MF
- OSGI-INF/
  - deployment-fragment.xml
  - schemas-contrib.xml
  - types-contrib.xml
  - *somemodule*-contrib.xml
  - l10n/
    - messages_fr.properties
    - messages_fr.properties
- build.properties
- build.xml
- pom.xml
- resources/
- src/
  - ...
- test/
  - ...
- web/
  - ...

2.1 .classpath

This file is used by Eclipse to record the dependencies on other Eclipse projects or libraries. It is used to build a classpath when Eclipse need to run or test a file in this project, or to resolve dependencies when checking source code for errors.

2.2 .project

This file is used by Eclipse to record meta-information about this project.

2.3 META-INF/MANIFEST.MF

This file holds the standard Jar manifest, as well as OSGi- and Nuxeo- specific extensions. An example follows:

```
Manifest-Version: 1.0
Bundle-ManifestVersion: 1
Bundle-Name: Sample project
Bundle-SymbolicName: com.nuxeo.project.sample;singleton=true
Bundle-Version: 1.0.0
Bundle-Vendor: Nuxeo
Provide-Package: com.nuxeo.project.sample
Nuxeo-Component: OSGI-INF/schemas-contrib.xml,
 OSGI-INF/types-contrib.xml,
 OSGI-INF/somemodule-contrib.xml
```

2.3.1 OSGi headers

The `Bundle-` headers are are standard OSGi manifest headers, describing the bundle and the way it interacts with other bundles in terms of Java classes requirements and exports. Their use is described in detail in the NXRuntime tutorial.

2.3.2 Nuxeo 5 headers

The Nuxeo-Component header describes the files that should be loaded at deployment time by NXRuntime and interpreted as extension descriptions. These files contain further configuration information that will be applied to the component when it is deployed, they are where plugin registration and use is made. The OSGI-INF section describes them.

2.3.3 Other headers

2.4 OSGI-INF/ folder

This directory contains files relevant to the OSGi deployment of the bundle. They are accessible only during deployment by NXRuntime, but are not accessed after this phase. Most of these files are only used when referenced through the MANIFEST.MF file, but some of them are special.

2.4.1 OSGI-INF/deployment-fragment.xml

This file contains Nuxeo 5 Java EE deployment information. It is read and processed by NXRuntime (actually, NXJBossExtensions). Its goal is to provide instructions about the way the bundle should be integrated with the rest of the Nuxeo 5 Java EE framework, including contributions to:

- the JBoss configuration (jboss-app.xml),
- the EJB 3 configuration (`application.xml`),
- the servlet configuration (web.xml),

- the Faces configuration (`faces-config.xml`).

It also describes what steps should be taken to copy or modify further files needed into the global War file, this is used for instance to insert further translations, or taglibs.

The NXRuntime reference and tutorial detail the contents and semantics of this file.

2.4.2 OSGI-INF/*somemodule*-contrib.xml

All files referenced through the Nuxeo-Component header of the MANIFEST.MF file contain NXRuntime component with their extension point definitions and contributions.

Extension point definitions offer plugin services to other components.

Extension point contributions use the plugins and pass them some configuration information.

The NXRuntime reference and tutorial detail the structure of these component files. The reference and tutorials for each Nuxeo 5 component describe the semantics of each and the syntax to use to contribute to them.

2.4.3 OSGI-INF/l10n

This folder usually contains localization (l10n) data. Its content is copied by the `deployment-fragment.xml` into the global webapp's `classes` folder in order to make it available at runtime for internationalization (i18n) processing.

The name of this folder is just a convention.

2.5 build.properties

This file is used by Ant to define the default values of some variables used during processing of the build.xml file.

2.6 build.xml

This is the Ant build file. Its content is interpreted by Ant (“`ant`” command) and describes various compilation, testing, packaging, deployment or other phases.

2.7 pom.xml

This is the Maven build file. Its content is interpreted by Maven (“`mvn`” command) and describes various compilation, testing, packaging, deployment or other phases. It also describes meta-information about the package seen as a Maven artifact, to describe dependencies on other Maven artifact for instance.

2.8 resources/ folder

This folder contains various files that will be accessible in the final Jar. The Ant or Maven build are responsible for copying its content into the Jar.

The contents of this folder is extremely varied, it can contain for instances schemas, directories SQL configuration, workflow configuration, etc.

2.9 web/ folder

The web/ folder is used by a webapp. In a Nuxeo 5 component, it usually contains images and XHTML files used by the Faces framework.

During packaging, the contents of this file is copied by the Ant or Maven build into a War file placed inside the final Jar.

During deployment, the contents of this War file will be added, as specified by OSGI-INF/deployment-fragment.xml, into the final Nuxeo War file of the Nuxeo Ear.

2.10 src/ folder

This folder contains the source files for the Java classes that will be built. These class files are put into the final Jar by the Ant or Maven build.

2.11 test/ folder

This folder contains the test source files. It is structured like the src/ folder. These files or the classes generated are not included in the final Jar.