

Repository Layout - Final

Repository Layout Definition

This is the final layout for the repository available in Maven 2.x and its related Ant tasks.

i This document should be incorporated into the main Maven site shortly, and a link left here as some external people have linked in directly.

Issues with the old layout

- It doesn't scale physically. The disk at ibiblio is completely trashed during peak hours. I've had cases where it used a minute and a half for doing `ls` in the root.
- For a user using browsing the repository through a browser it's hard to locate the artifacts the user want because of the number of files in the directory. This might not be a really big problem if we make a application on top of the repository for browsing.
- The layout doesn't differ between "primary" and "secondary" artifacts. ("secondary" artifacts beeing javadoc, sources etc that do not have a POM of their own).

The current layout looks like this:

```
/$groupId/$type +  
"s"/$artifactId-$version.$type
```

The new layout:

Changes from the old way includes:

- Overall, the entire directory tree is much deeper. This is for both so that it will scale better in terms of number of files/directories in a single directory. This will ensure that the haddisks that are hosting the repository isn't thrashed like they are on ibiblio today and it will be easier for a user to get a overview over a artifact, the versions of a artifact and the secondary artifacts it has.
- For each primary artifact there is a pom file.
- No symlinks.

For primary artifacts:

```
/$groupId[0]/../${groupId[n]}/$artifactId/$ve  
rsion/$artifactId-$version.$extension
```

For secondary artifacts:

```
/$groupId[0]/../$groupId[n]/$artifactId/$version/$artifactId-$version-$classifier.$extension
```

\$groupId is an array of strings made by splitting the groupId's on "." into directories. The group org.apache.maven would then yield: org/apache/maven. This should mostly mirror the package structure in Java, though can apply to any language.

For each primary artifact there will be a POM:

- A \$artifactId-\$version.pom

POMs that are exclusively parents (packaging = pom) will have the same filename.

Secondary artifacts do not need a POM - they will reference the associated primary POM.

For each file that is the repository there must be a file containing the checksum of the file, typically md5 or sha1. There may also be a digital signature (eg .asc an ascii armoured openpgp signature)

A complete example

In the /org/codehaus/plexus directory:

```
plexus/  
  /plexus-RELEASE.version.txt  
  /plexus-RELEASE.version.txt.md5  
plexus/1.0/  
  /plexus-1.0.pom  
  /plexus-1.0.pom.md5  
plexus-container/  
  
/plexus-container-RELEASE.version.txt  
plexus-container/0.15-SNAPSHOT/  
  
/plexus-container-0.15-200407151214.jar  
  
/plexus-container-0.15-200407151214.jar.md5
```

/plexus-container-0.15-200407151214.jar.sha1

/plexus-container-0.15-200407151214.pom

/plexus-container-0.15-200407151214.pom.md5

/plexus-container-0.15-200407151315.jar

/plexus-container-0.15-200407151315.jar.md5

/plexus-container-0.15-200407151315.pom

/plexus-container-0.15-200407151315.pom.md5

/plexus-container-0.15-SNAPSHOT.version.txt

/plexus-container-0.15-SNAPSHOT.version.txt.
md5

plexus-container/0.16-SNAPSHOT/

/plexus-container-0.16-200407151612.jar

/plexus-container-0.16-200407151612.jar.md5

/plexus-container-0.16-200407151612.pom

/plexus-container-0.16-200407151612.pom.md5

/plexus-container-0.16-SNAPSHOT.version.txt

/plexus-container-0.16-SNAPSHOT.version.txt.
md5

plexus-container/0.15/

/plexus-container-0.15.jar

/plexus-container-0.15.jar.md5

/plexus-container-0.15.pom

/plexus-container-0.15.pom.md5

/plexus-container-0.15-javadoc.jar

/plexus-container-0.15-javadoc.jar.md5

/plexus-container-0.15-javasc.jar

/plexus-container-0.15-javasc.jar.md5

plexus-container/0.16/

/plexus-container-0.16.jar

/plexus-container-0.16.jar.md5

/plexus-container-0.16.pom

/plexus-container-0.16.pom.md5

/plexus-container-0.16-javadoc.jar

/plexus-container-0.16-javadoc.jar.md5

/plexus-container-0.16-javasc.jar

/plexus-container-0.16-javasc.jar.md5

/plexus-container-0.16-src.tar.gz

/plexus-container-0.16-src.tar.gz.md5

/plexus-container-0.16-bin.tar.gz

/plexus-container-0.16-bin.tar.gz.md5

some-ejb/1.0/

 /some-ejb-1.0.jar

 /some-ejb-1.0.jar.md5

 /some-ejb-1.0.pom

```
/some-ejb-1.0.pom.md5
/some-ejb-1.0-client.jar
/some-ejb-1.0-client.jar.md5
```

Different Repositories

To ease the different uses and archival policies there will later be added 3 repository types:

1. SNAPSHOT repository, where only snapshots are stored
2. Archive repository, a read only repository where old releases can be downloaded from
3. Release repository, where official, current releases go

Repositories can be any combination of these.

If using an archive repository, it might be considered as a deprecated repository, so you could warn on uses of this for anything other than building old releases themselves.

POM referencing

The POM will always reference the primary artifact.

There can only be one POM for a groupId:artifactId combination - type does not factor into it.

The conflict ID of a dependency is:

groupId:artifactId:type

and the full versioned ID is

groupId:artifactId:type:version

The inclusion of type is to accommodate the coexistence of subtypes: eg ejb and ejb-client.

Subtypes are created by particular mojos: eg ejb packaging will create both an ejb and an ejb-client. apidocs:package will create a JAR of the javadocs.

Dependencies will generally only reference the type corresponding to the packaging. However, in some cases, dependency on a subtype will also be required, eg:
<type>ejb-client</type>.

So two projects with the same group ID and artifact ID, but different packaging are not valid together. However, two dependencies with different types are. Therefore the common use case of having tlds with the same group/artifact ID will work, eg:

```
/taglibs/1.2.2/taglibs-1.2.2.pom
```

```
/taglibs/1.2.2/taglibs-1.2.2.jar
```

```
/taglibs/1.2.2/taglibs-1.2.2.tld
```

```
/taglibs/1.2.2/taglibs-html-1.2.2.jar
```