

CRX Content Package Deployment Plugin

for Jenkins

Mark Adamcin

Acquity Group

Part of Accenture Interactive

mark.j.adamcin@accenture.com

About Me

- Technical Consultant with Acquity Group since 2007, which is now part of Accenture Interactive
- CQ5 Developer and Application Architect since 2009
- Now a Release Management specialist with our Shared Services DevOps group, focused on supporting client application deployments using Jenkins
- Since 2009, I've written five (5!) maven plugins just for building and deploying CRX Content Packages.

This plugin, what is it good for?

- Uploading/Installing CRX Content Packages on CQ/AEM servers
- Downloading CRX Content Packages from CQ/AEM servers
- Doing both of those things without knowing the AEM admin password
- And it's completely free to use!

Wait, no admin password, you say?

- The plugin uses the httpsig-java implementation of a draft HTTP Signature Authentication scheme based on SSH Public Key login.
- Simply install two OSGi bundles and an authorized_keys file on the AEM instance's filesystem to enable support on the server.
- [https://github.com/adamcin/
net.adamcin.sling.auth.httpsig/wiki/getting-started](https://github.com/adamcin/net.adamcin.sling.auth.httpsig/wiki/getting-started)

But why not Maven?

- Maven plugins like the **content-package-maven-plugin** and **maven-vault-plugin** generally only support installing one package on one host per execution.
- Managing additional packages and hosts requires a pom.xml file, usually managed separately from Jenkins in SCM.
- Maven logs are full of non-package-deployment-related verbosity that gets in the way of knowing whether or not installation truly succeeded.

Features

Deploy Packages

- The "Deploy Content Packages to CRX" build step can be added to any job type.
- Flexible in regards to how the package files are created and organized within the workspace, because it will recursively scan and identify any files which have a ".jar" or ".zip" extension.
- Feel free to use this step immediately after a maven build, or after the "Download Content Packages from CRX" step, or after resolving maven artifacts from a repository manager into the workspace.
- Once a package has been identified (i.e. has a valid group:name:version), the package will be checked against the Package ID filter to determine whether it must be uploaded to the configured Base URL(s).
- If a package installation succeeds with errors "(check logs!)", the build will be marked as unstable.

Deploy Packages

Deploy Content Packages to CRX

Deploy CRX Content Package files from the workspace to a configured Adobe Granite repository via the CRX Package Manager. Only package files with **.zip** or **.jar** extensions will be identified and deployed.
(from [CRX Content Package Deployer Plugin](#))

Package ID

Specify a list of package ID filters to match against identified Vault packages located in the workspace. Each filter string must occupy its own line. A Package ID consists of the group, the name, and the version of a package, separated by colons (':'). Package ID filters must follow one of three patterns:

- group:name:version
- group:name
- name

Each segment in the filter may be either omitted or replaced with an asterisk ('*') to represent a wildcard. Matching packages will be uploaded in the order in which the filters are specified. Only the highest matching version of a package identified by 'group:name' will be deployed, and it will only be deployed once per build step, regardless of the number of matching filters.
(from [CRX Content Package Deployer Plugin](#))

Existing package behavior

Install sub-packages ☐

Access Control Handling

Auto Save

Local Directory

Disable for Job Testing ☐

Base URL

Specify the base URL of the Adobe Granite server, including hostname and port. The CRX Package Manager service path will be appended to this value for all requests. For example, <http://localhost:4502> or <https://author.mycorp.com>, etc. Multiple urls can be specified, one per line. All specified packages will be deployed to one server before beginning deployments to the next one.
(from [CRX Content Package Deployer Plugin](#))

Credentials

Request Timeout

Service Timeout

Console Output

```
Started by user anonymous
Building in workspace D:\source\adamsin_github\jenkins-granite-client\work\jobs\test-granite-client\workspace
Beginning to resolve Build Info published dependencies.
Resolving published dependencies with pattern sem-builds:test-packmgr/test-packmgr-client/1.0/test-packmgr-client-1.0.zip
Found 1 dependencies.
Finished resolving Build Info published dependencies.
Beginning to resolve Build Info build dependencies.
Finished resolving Build Info build dependencies.
Deploying packages to http://localhost:4502
Deploying D:\source\adamsin_github\jenkins-granite-client\work\jobs\test-granite-client\workspace\some-subdirectory\test-packmgr\test-packmgr-client\1.0\test-packmgr-client-1.0.zip to
http://localhost:4502/crx/package/index.jsp/etc/packages/test-packmgr/test-packmgr-client-1.0.zip
Will attempt to upload package.
Package uploaded
Will attempt to install package.
Installing content
M Creating snapshot for package test-packmgr:test-packmgr-client:1.0
A META-INF
A META-INF/vault
A META-INF/vault/config.xml
A META-INF/vault/filter.xml
A META-INF/vault/nodetypes.cnd
A META-INF/vault/properties.xml
A /.content.xml
A /test-packmgr
A /test-packmgr/foo
A /test-packmgr/foo/.content.xml
- Aggregation status: 2 of 0 prepared, 1 collected
A META-INF/vault/definition/.content.xml
M Collecting import information...
M Installing node types...
- sling -> http://sling.apache.org/scr/sling/1.0
- nt -> http://www.tcp.org/scr/1.0
- rep -> internal
- sling:Folder
- rep:RepoAccessControllable
M Installing privileges...
M Importing content...
- /test-packmgr/foo
M saving approx 0 nodes...
M Package imported.
Package uploaded
Deploying packages to http://localhost:4503
Deploying D:\source\adamsin_github\jenkins-granite-client\work\jobs\test-granite-client\workspace\some-subdirectory\test-packmgr\test-packmgr-client\1.0\test-packmgr-client-1.0.zip to
http://localhost:4503/crx/package/index.jsp/etc/packages/test-packmgr/test-packmgr-client-1.0.zip
Will attempt to upload package.
Package uploaded
Will attempt to install package.
Installing content
M Creating snapshot for package test-packmgr:test-packmgr-client:1.0
A META-INF
A META-INF/vault
A META-INF/vault/config.xml
A META-INF/vault/filter.xml
A META-INF/vault/nodetypes.cnd
A META-INF/vault/properties.xml
A /.content.xml
A META-INF/vault/definition/.content.xml
M Collecting import information...
M Installing node types...
- sling -> http://sling.apache.org/scr/sling/1.0
- nt -> http://www.tcp.org/scr/1.0
- rep -> internal
- sling:Folder
- rep:RepoAccessControllable
M Installing privileges...
M Importing content...
A /test-packmgr
A /test-packmgr/foo
M saving approx 2 nodes...
M Package imported.
Package uploaded
Build step 'Deploy Content Packages to CRX' changed build result to SUCCESS
Finished: SUCCESS
```


List Packages to Download

Select any number of content packages available for download from a CRX server as a Build Parameter, which can then be used in other components provided by this plugin.

List Packages to Download

CRX Content Package Choice Parameter ⓘ

List Content Packages available for download from a configured Adobe CRX repository. (from [CRX Content Package Deployer Plugin](#))

Name

Description

☒ Multiselect

Visible Item Count ⓘ

Package ID Filter ⓘ

Specify a package ID filter to match against the list of packages returned from the configured Granite server. A Package ID consists of the group, the name, and the version of a package, separated by colons (':'). Package ID filters must follow one of three patterns:

- group:name:version
- group:name
- name

Each segment in the filter may be either omitted or replaced with an asterisk (*) to represent a wildcard. (from [CRX Content Package Deployer Plugin](#))

Full-Text Search ⓘ

Exclude Packages which are not installed ☒ ⓘ

Exclude Packages which have been modified ☒ ⓘ

Base URL ⓘ

Specify the base URL of the Adobe Granite server, including hostname and port. The CRX Package Manager service path will be appended to this value for all requests. For example, **http://localhost:4502** or **https://author.mycorp.com**, etc. (from [CRX Content Package Deployer Plugin](#))

Credentials ⓘ

Select the login credentials with which content packages will be listed from the CRX server. **[Signature]** credentials may be used if the target server supports HTTP Signature Authentication using the keyId format, /\$username/keys/\$fingerprint. Select "-none-" to use the default credentials set in the global **CRX Content Package Deployer - HTTP Client** configuration. (from [CRX Content Package Deployer Plugin](#))

Request Timeout ⓘ

Service Timeout ⓘ

This build requires parameters:

PACKAGE_ID

```
day/cq561/product:cq-content-mac:5.6.20
day/cq561/product:cq-geometrixx-all-pkg:5.6.12
day/cq561/product:cq-geometrixx-commons-pkg:5.6.4
day/cq561/product:cq-geometrixx-login-pkg:5.6.4
day/cq561/product:cq-geometrixx-media-pkg:5.6.8
day/cq561/product:cq-geometrixx-outdoors-pkg:5.6.4
day/cq561/product:cq-geometrixx-pkg:5.6.10
day/cq561/product:cq-geometrixx-users-pkg:5.6.4
day/cq561/product:cq-geometrixx-workflow-pkg:5.6.4
day/cq561/product:cq-media-content:5.6.10
```

List of packages on <http://localhost:4502>.

Download Packages

- The "Download Content Packages from CRX" build step can be used on any job type to download content packages to the workspace.
- Use the CRX Content Package Choice Parameter to select one or more packages from a live CRX Package Manager service.
- Downloaded packages are organized according to their CRX installation path, relative to the workspace path, or local directory, if specified.

Download Packages

Download Content Packages from CRX

Download CRX Content Package files from a configured Adobe Granite repository into the workspace via the CRX Package Manager.
(from [CRX Content Package Deployer Plugin](#))

Package ID
\${PACKAGE_ID}
day/cq561/product:cq-media-content:5.6.10

Specify a list of package IDs to download to the workspace. Each package ID string must occupy its own line. A Package ID consists of the group, the name, and the version of a package, separated by colons (:), as in: "group:name:version".
(from [CRX Content Package Deployer Plugin](#))

Ignore Errors ☐

Local Directory

Base URL

Credentials

Request Timeout

Service Timeout



Console Output

```
Started by user anonymous
Building in workspace D:\source\adamcin_github\jenkins-granite-client\work\jobs\download-granite-client\workspace
Checking for package day/cq561/product:cq-geometrix-login-pkg:5.6.4 on server http://localhost:4502
Found package: http://localhost:4502/crx/packmgr/index.jsp#/etc/packages/day/cq561/product/cq-geometrix-login-pkg-5.6.4.zip
Downloading day/cq561/product:cq-geometrix-login-pkg:5.6.4 to D:\source\adamcin_github\jenkins-granite-client\work\jobs\download-granite-client\workspace
Downloaded 394013 bytes to file D:\source\adamcin_github\jenkins-granite-client\work\jobs\download-granite-client\workspace\etc\packages\day\cq561\product\cq-geometrix-login-pkg-5.6.4.zip.
Verifying downloaded package...
Package verified as day/cq561/product:cq-geometrix-login-pkg:5.6.4.
Checking for package day/cq561/product:cq-geometrix-pkg:5.6.10 on server http://localhost:4502
Found package: http://localhost:4502/crx/packmgr/index.jsp#/etc/packages/day/cq561/product/cq-geometrix-pkg-5.6.10.zip
Downloading day/cq561/product:cq-geometrix-pkg:5.6.10 to D:\source\adamcin_github\jenkins-granite-client\work\jobs\download-granite-client\workspace
Downloaded 36968510 bytes to file D:\source\adamcin_github\jenkins-granite-client\work\jobs\download-granite-client\workspace\etc\packages\day\cq561\product\cq-geometrix-pkg-5.6.10.zip.
Verifying downloaded package...
Package verified as day/cq561/product:cq-geometrix-pkg:5.6.10.
Checking for package day/cq561/product:cq-media-content:5.6.10 on server http://localhost:4502
Found package: http://localhost:4502/crx/packmgr/index.jsp#/etc/packages/day/cq561/product/cq-media-content-5.6.10.zip
Downloading day/cq561/product:cq-media-content:5.6.10 to D:\source\adamcin_github\jenkins-granite-client\work\jobs\download-granite-client\workspace
Downloaded 1296764 bytes to file D:\source\adamcin_github\jenkins-granite-client\work\jobs\download-granite-client\workspace\etc\packages\day\cq561\product\cq-media-content-5.6.10.zip.
Verifying downloaded package...
Package verified as day/cq561/product:cq-media-content:5.6.10.
Finished: SUCCESS
```

Compact Workspace Filters

- each line consists of significant text before an optional comment character (#)
- each line that begins with a "/" begins a new filter root.
- the first non-empty, non-comment line must define a new filter root
- each non-empty, non-comment line after a filter root that begins with a + or - defines an include or exclude rule, respectively.
- everything following the + or - must be a valid regular expression

Compact Workspace Filters

For example, to include everything under /etc except for packages:

```
/etc          # define /etc as the filter root
+/etc(/.*)?   # include everything under /etc
-/etc/packages(/.)? # exclude package paths
```

To create a package for a project "acme" defined in CRX DE Lite, a filter may look like this:

```
/content/acme    # include the site content
/apps/acme        # include the app code
```

Build Package

- The “Build a Content Package on CRX” build step can be added to any job type.
- Specify a Package ID (group:name:version) and a Workspace Filter (using the compact syntax) and the package will be created and built on the specified server.
- If built successfully, the package is downloaded to the workspace, from where it can then be archived or uploaded to other servers.
- This step was designed to support an automated refresh of lower environments using production content.

Build Package

 **Build a Content Package on CRX** 

Package ID

test-packmgr:acme-project 

Workspace Filter

/content/acme

/apps/acme



Download after Build

☒ 

Local Directory



Connection Options...





Console Output

```
Started by user anonymous
Building in workspace /Users/madamcin/projects/jenkins-granite-client/work/jobs/test-build-package/workspace
Checking for package test-packmgr:acme-project: on server http://localhost:4502
Found package: http://localhost:4502/crx/packmgr/index.jsp#/etc/packages/test-packmgr/acme-project.zip
Package updated successfully
Building package test-packmgr:acme-project:.
Building package
A META-INF
A META-INF/vault
A META-INF/vault/config.xml
A META-INF/vault/filter.xml
A META-INF/vault/nodetypes.cnd
A META-INF/vault/properties.xml
A /.content.xml
A /apps
A /apps/.content.xml
A /apps/acme
A /apps/acme/components
A /apps/acme/components/sample
A /apps/acme/components/sample/html.jsp
A /apps/acme/install
A /apps/acme/install/com.example.acme.jar
A /apps/acme/src
A /apps/acme/src/impl
A /apps/acme/src/impl/com.example.acme.bnd
A /apps/acme/src/impl/libs
A /apps/acme/src/impl/src
A /apps/acme/src/impl/src/main
A /apps/acme/src/impl/src/main/java
A /apps/acme/src/impl/src/main/java/com
A /apps/acme/src/impl/src/main/java/com/example
A /apps/acme/src/impl/src/main/java/com/example/acme
A /apps/acme/src/impl/src/main/java/com/example/acme/SampleUtil.java
A /apps/acme/src/impl/src/main/java/com/example/acme/impl
```




Validate Packages

- The “Validate CRX Content Packages” build step can be added to any job type.
- Scan packages for forbidden file extensions and restrict the scope of package workspace filters to control the content that can be deployed along any given pipeline
- Specify a Workspace Filter using the compact syntax, and only the specified filter roots will be allowed. You can even specify mandatory include/exclude rules for each root to ensure that
- This step was designed to support an automated refresh of lower environments using production content.


Validate Packages

 **Validate CRX Content Packages** 


Package ID


Local Directory




Forbidden Extensions



Validation Workspace Filter



Allow Non-covered Roots

☒ 



Console Output

```
Started by user anonymous
Building in workspace /Users/nadancin/projects/jenkins-granite-client/work/jobs/test-build-package/workspace
Validating packages.
Validating package test-packmgr:acme-project: at path /Users/madamcin/projects/jenkins-granite-client/work/jobs/test-build-package/workspace/etc/packages/test-packmgr/acme-project.zip.
FATAL: Package workspace filter defines a filter root which does not list the rules required by its covering validation filter.
ERROR: Invalid filter set:
/apps/acme

ERROR: Covered by validation filter set:
/apps
-/apps(/.*)?/install(/.*)?

Build step 'Validate CRX Content Packages' changed build result to FAILURE
Build step 'Validate CRX Content Packages' marked build as failure
Finished: FAILURE
```

Other Features

- Optionally replicate packages after deployment, or use the separate Replicate Content Packages build step
- Use Jenkins Credentials to define Domain Scopes if deploying to multiple environments with different credentials
- Use Jenkins SSH User Private Key Credentials instead of the admin password to avoid password management headaches.
- Use \${TOKENS} in Base URL and Package ID fields to parameterize deployment jobs

Before you deploy your first package, remember...

- Use the “**Disable for Job Testing**” option to execute a dry run to identify all the packages that would be installed during normal execution
- Clean up your left-over packages from your deploy job workspace when not using SCM or mvn clean!

<input type="checkbox"/>	Set the project description from a file in the workspace (à la GitHub README.md)	1.1
<input checked="" type="checkbox"/>	Workspace Cleanup Plugin This plugin deletes the workspace before the build or when a build is finished and artifacts saved.	0.19
	Config File Provider Plugin	

Build Environment

- ☐ Delete workspace before build starts

Future Features

- ~~Replicate Packages on Install, or as a separate build step (v 1.1)~~
- ~~Create/Build Packages using a simple text format for specifying a workspace filter (v 1.2)~~
- Automatic deletion of old package versions

Get It

- Download Jenkins WAR from <http://jenkins-ci.org/>
- Install the **CRX Content Package Deployment Plugin** from the Plugin Manager

<input type="checkbox"/>	Crittercism dSYM Plugin A Jenkins CI plugin for uploading dSYM files to Crittercism.	1.1
<input checked="" type="checkbox"/>	CRX Content Package Deployer Plugin Deploys content packages to Adobe CRX applications, like Adobe CQ 5.4, CQ 5.5, and AEM 5.6. Also allows downloading packages from one CRX server and uploading them to one or more other CRX servers.	1.0
<input type="checkbox"/>	Deploy Plugin This plugin takes a war/ear file and deploys that to a running remote application server at the end of a build	1.9

Install without restart

Download now and install after restart

Read the Wiki!

- <https://wiki.jenkins-ci.org/display/JENKINS/CRX+Content+Package+Deployer+Plugin>

Get the Source!

- <https://github.com/jenkinsci/crx-content-package-deployer-plugin>
- <https://github.com/adamcin/granite-client-packman>
- <https://github.com/adamcin/httpsig-java>

Get this Deck!

http://adamcin.net/files/jenkins_crx_plugin.pdf

Mark Adamcin

Acquity Group

Part of Accenture Interactive

mark.j.adamcin@accenture.com