CRAN Repositories

Overview

Artifactory natively supports CRAN repositories for the R language, giving you full control of your deployment and resolve process of CRAN packages. The Comprehensive R Archive Network (CRAN) is a collection of sites which carry identical material, consisting of R distributions, contributed extensions, R documentation, and binaries. R is a programming language and free software environment dedicated to statistical computing and graphics that is supported by the R Foundation for Statistical Computing. The R language is widely used for developing statistical software and data analysis by data miners and statisticians.

CRAN repositories in Artifactory offer the following benefits:

- Secure and private local CRAN repositories with fine-grained access control.
- The ability to proxy remote CRAN resources and cache downloaded CRAN packages to keep you independent of the network and the remote resource.
- Virtual CRAN repositories that support a single URL through which to manage the resolution and deployment of all your CRAN packages.
- Metadata calculation of the CRAN packages hosted in the Artifactory local repositories.
- Version management: Archiving older versions of the packages uploaded to local repositories.
- Manage source and binaries.

CRAN version support

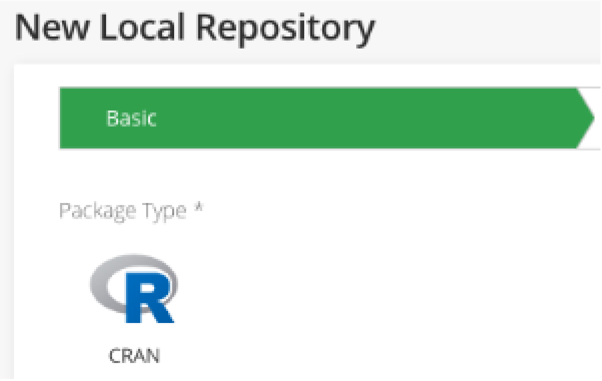
Artifactory supports CRAN version 3.4.0 and above.

Page Contents

Configuration

Local Repositories

To enable calculation of CRAN metadata, in the Administration module, go to **Repositories** | **Repositories** | **Local** and select **CRAN** as the **Package Type** when you create your local repository.



New Local Repository

Local Repository Layout

You need to maintain a specific path structure in order to manage the CRAN packages that are uploaded to CRAN local repositories.

CRAN packages are uploaded to the following locations:

- Source packages are automatically uploaded by default to the relative path: *src/contrib*. For example: *`src/contrib/ArtifactoryRDS_0.1.0.tar.gz`*.
- Binary packages are uploaded to a relative path according to the distribution and R version. For example: *`/myfirstpkg_1.2.tgz`*.

Artifactory will find your packages by performing a property search causing the folder hierarchy not to have an impact on performance.

> **Placing source packages in the recommended path**
>
> When uploading a CRAN package via the UI, the default deploy path is not enforced but is recommended since it allows Artifactory to manage the CRAN packages. Uploading the packages to a different path will cause the packages not to be identified as CRAN packages.

Remote Repositories

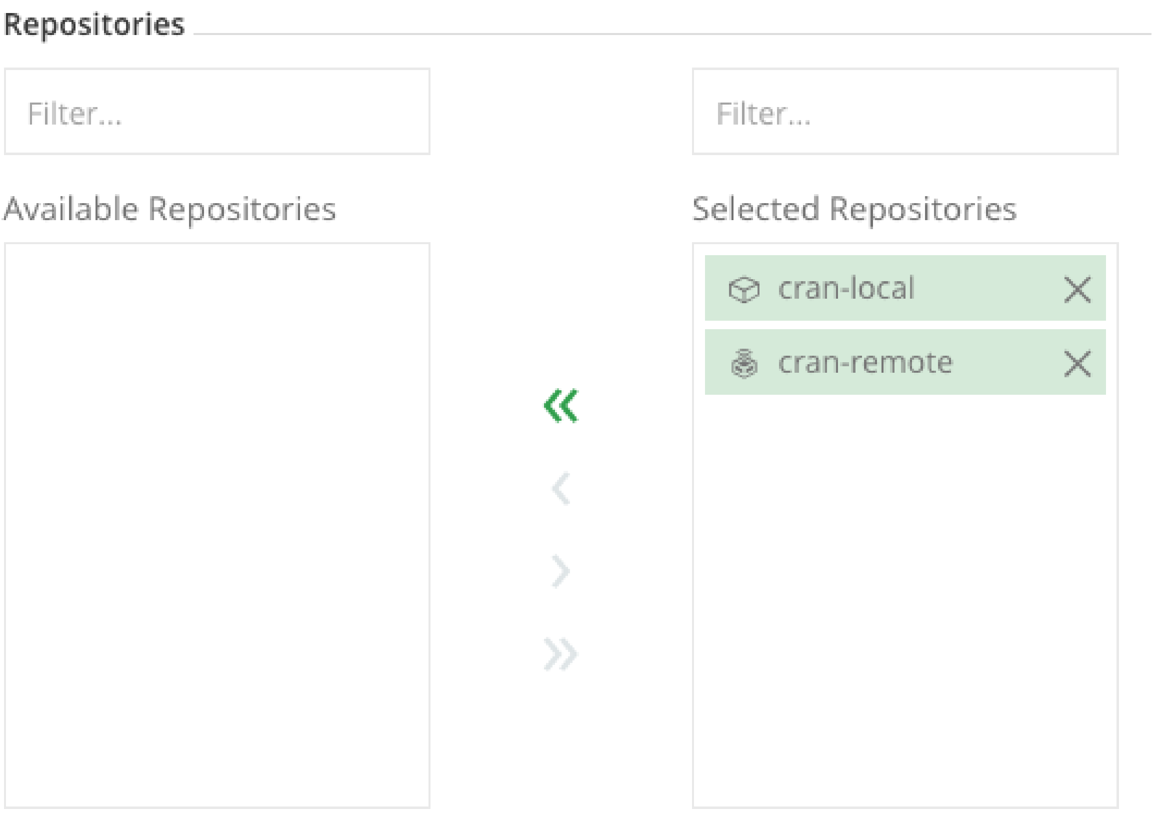
You can create CRAN remote repository to proxy and cache remote repositories or other Artifactory instances.

Note that the index files for remote CRAN repositories are stored and renewed according to the Retrieval Cache Period setting.

Virtual Repositories

A Virtual Repository in Artifactory aggregates packages from both local and remote repositories. This allows you to access both locally hosted CRAN packages and remote proxied CRAN libraries from a single URL defined for the virtual repository.

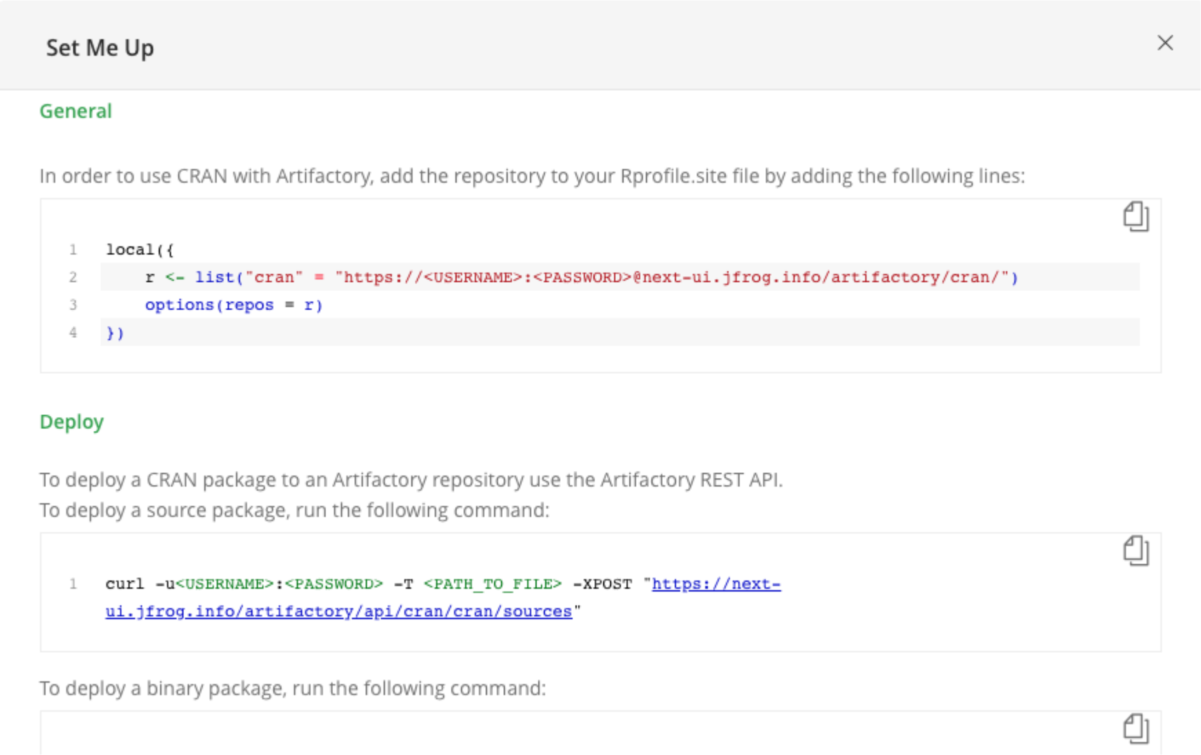
To create a virtual CRAN repository, set **CRAN** as the **Package Type**, and select the underlying local and remote CRAN repositories to include under the **Repositories** section.



Resolving CRAN Packages

Resolving CRAN Packages in the UI

When a CRAN repository is selected in the Artifacts module Tree Browser, click **Set Me Up** to view the code snippets you can use to publish a CRAN package or to configure your R client to resolve artifacts using the selected repository.



Set Me Up

General

In order to use CRAN with Artifactory, add the repository to your Rprofile.site file by adding the following lines:

```
local({
    r <- list("cran" = "https://<USERNAME>:<PASSWORD>@next-ui.jfrog.info/artifactory/cran/")
    options(repos = r)
})
```

Deploy

To deploy a CRAN package to an Artifactory repository use the Artifactory REST API.
To deploy a source package, run the following command:

```
curl -u<USERNAME>:<PASSWORD> -T <PATH_TO_FILE> -XPOST "https://next-ui.jfrog.info/artifactory/api/cran/cran/sources"
```

To deploy a binary package, run the following command:

Resolving CRAN Packages using the R Command Line

1. Run the **Set Me Up** for CRAN.

2. To switch from the current repository to a different resolution CRAN repository.

```
setRepositories()
```

3. View all the available packages for the selected CRAN repository.

```
available.packages()
```

4. Install a package from the CRAN repository.

```
install.packages() - select from a list, on supported clients
install.packages("package") - install by package name
```

Deploying CRAN Packages

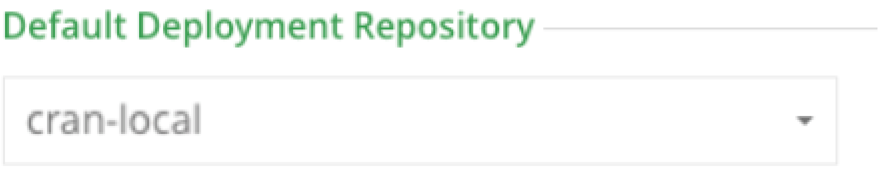
You can deploy packages to a local or virtual CRAN repository using the **Deploy** feature in the UI or using a POST request.

Metadata Updates

The CRAN metadata is automatically calculated and updated when adding, removing, copying or moving CRAN packages. The calculation is only invoked after a package-related action is completed.
It may sometimes take up to 30 seconds to complete as the process is asynchronous and its performance depends on the overall system load.
You can also invoke metadata calculation on the entire repository by selecting **Reindex Packages**.

Setting the Default Deployment Repository

To deploy CRAN packages to a virtual CRAN repository, make sure you have set the **Default Deployment Repository**.



Default Deployment Repository

cran-local

Deploying a Package Using the UI

You can drag and drop, or select a CRAN package to upload in Deploy in the UI. Artifactory will identify if it's a source or binary package.

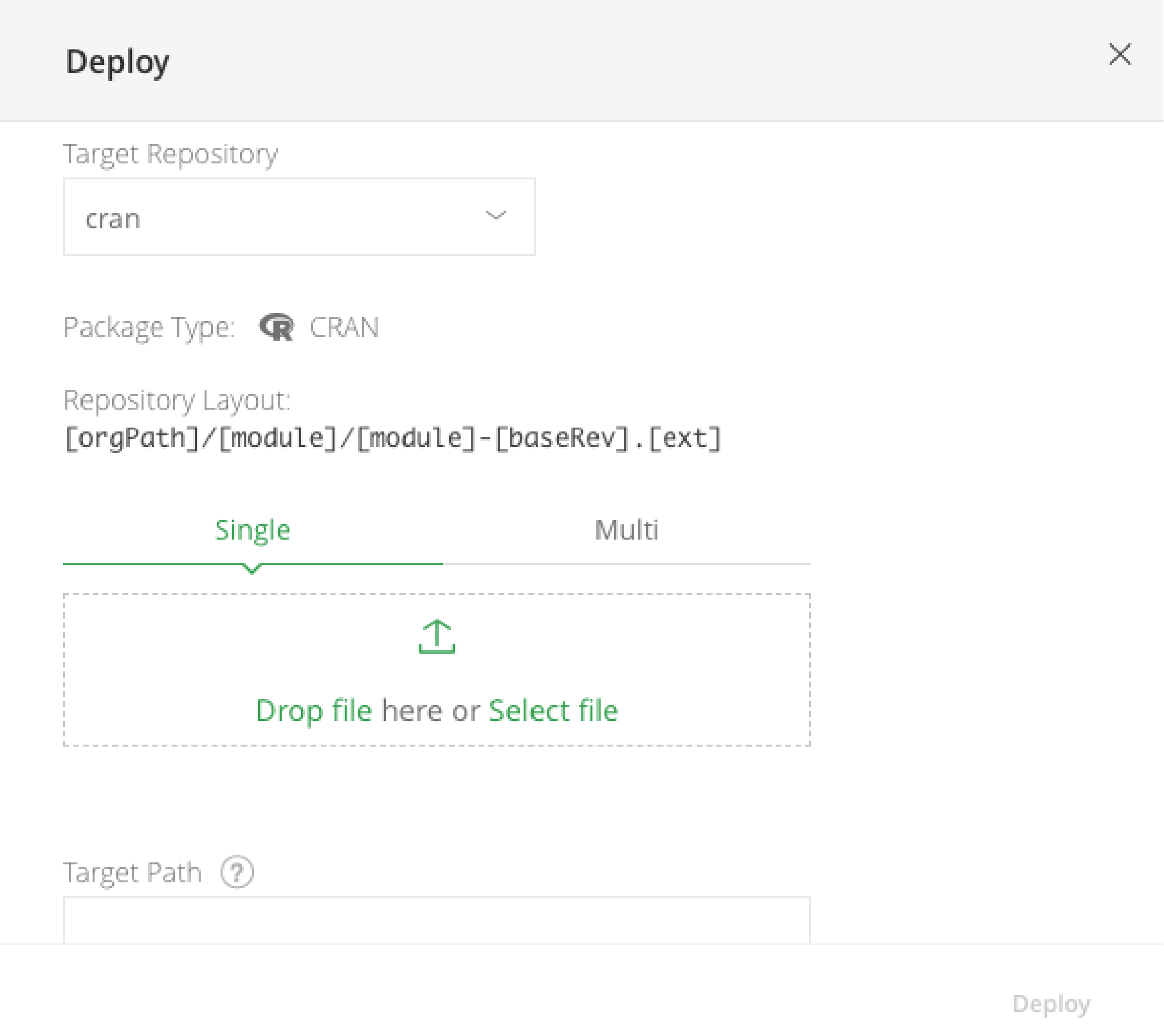
Artifactory supports two types of packages: binaries and sources. They are treated differently in terms of the deployment in the UI.

Deploying a Source Package

When deploying sources deploy, the Target Path is automatically displayed and we recommend not changing this path. Changing the `src/contrib` path will result in Artifactory not identifying the package as a CRAN package since Artifactory will not be able to index it.

Target Path

The Target path is automatically updated and changing it could make Artifactory not invoke the metadata calculation, and this package to not get indexed.



Deploy

Target Repository

cran

Package Type: CRAN

Repository Layout:
[orgPath]/[module]/[module]-[baseRev].[ext]

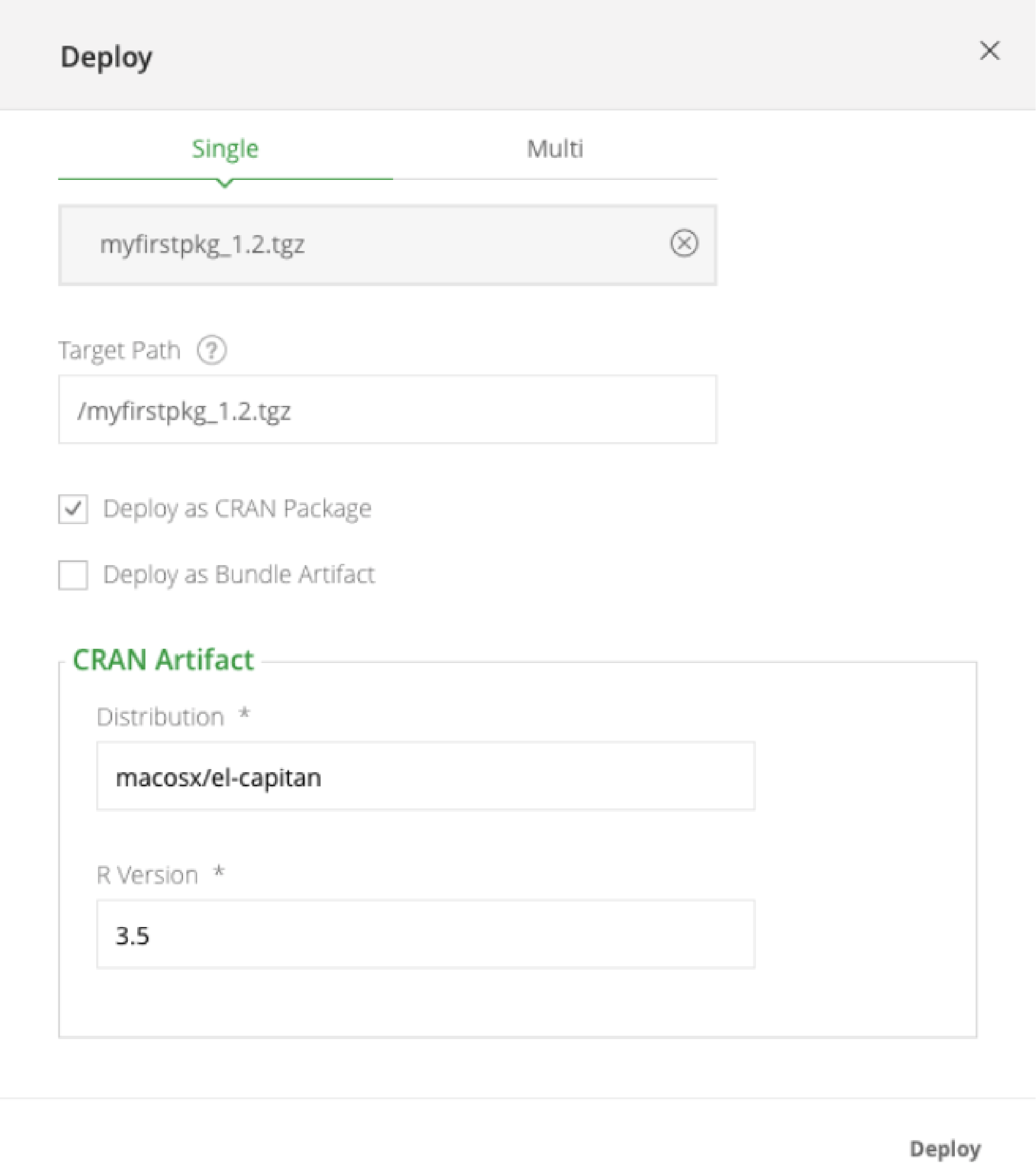
Deploying a Binary Package

In binary deploy, you'll need to fill the CRAN Artifact section.

In the CRAN Artifact section, configure these fields when deploying the CRAN packages. It is mandatory to set these fields and are used to create the destination path of the deployed binary package.

- Distribution: Specifies the operating system.
- R Version: Indicates the R version used.

> **Target Path**
>
> The Target path is updated **after** the file is deployed and there is no need to change it



myfirstpkg_1.2.tgz

Target Path ?

/myfirstpkg_1.2.tgz

☑ Deploy as CRAN Package

□ Deploy as Bundle Artifact

CRAN Artifact

Distribution *

macosx/el-capitan

R Version *

3.5

Deploying a Package Using cURL

Deploy source package

```
curl -XPOST "http://localhost:8080/artifactory/api/cran/cran-local/sources" -T package_1.0.tar.gz
```

Deploy binary package

```
curl -XPOST "http://localhost:8080/artifactory/api/cran/cran-local/binaries?distribution=macosx/el-capitan&rVersion=3.5" -T package_1.0.tgz
curl -XPOST "http://localhost:8080/artifactory/api/cran/cran-local/binaries?distribution=windows&rVersion=3.5" -T package_1.0.zip
```

When deploying a CRAN binary package, you need to specify the distribution and R version as before.

When deploying directly (PUT request to a specific path), make sure the target path is a valid CRAN path:

- **`/src/contrib`** for sources
- **`/bin/{distribution}/contrib/{r-version}`** for binaries.

Deploying a package to a different path will not identify the package as CRAN packages, and will not invoke the metadata indexing.

Applying the CRAN Official Spec on Local CRAN Repositories

From Artifactory version 7.41.1, you can set the artifacts stored in local CRAN the repositories to confine to the official CRAN spec requirements.

Set the following system property in the Artifactory.yml file to save your CRAN archives in the correct hierarchy.

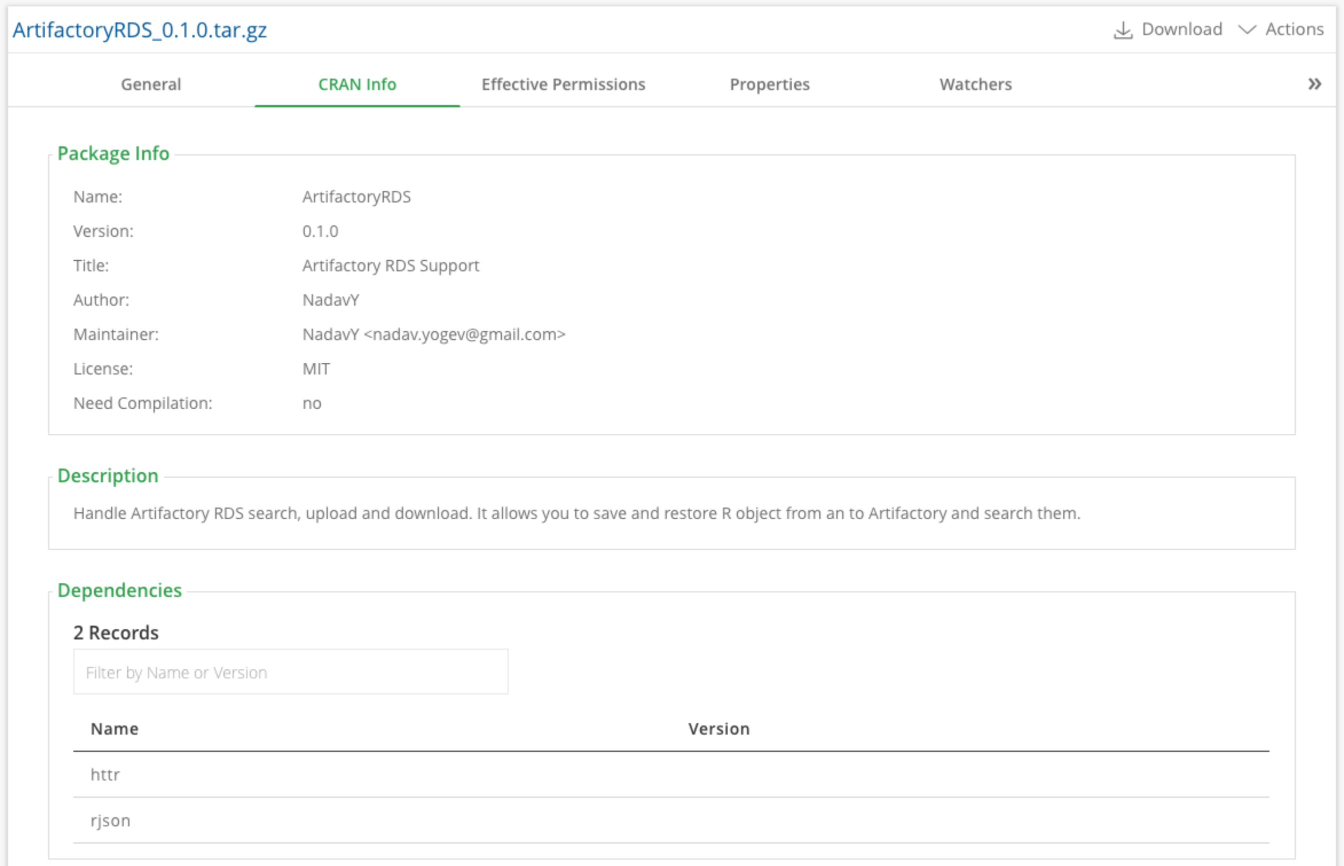
```
"cran.archiveMover.enabled"
```

You can move your existing Archives to the correct path using the following Move CRAN Archives REST API. Requires the `cran.archiveMover.enabled` to be enabled. Add a link

---

Viewing Individual CRAN Package Information

Artifactory lets you view selected metadata of a CRAN package directly from the UI.
In the **Artifact Repository Browser,** select your virtual CRAN repository and scroll down to find and select the package you want to inspect. The metadata is displayed in the **CRAN Info** tab.



ArtifactoryRDS_0.1.0.tar.gz

Package Info

Name:	ArtifactoryRDS
Version:	0.1.0
Title:	Artifactory RDS Support
Author:	NadavY
Maintainer:	NadavY <nadav.yogev@gmail.com>
License:	MIT
Need Compilation:	no

Description

Handle Artifactory RDS search, upload and download. It allows you to save and restore R object from an to Artifactory and search them.

Dependencies

2 Records

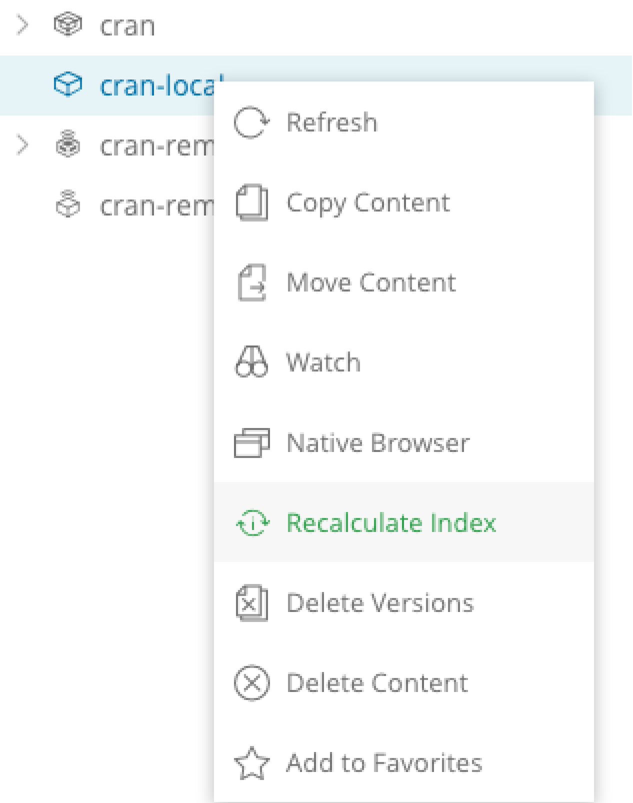
Filter by Name or Version

Name	Version
httr	
rjson	

Reindexing a CRAN Repository

You can trigger an asynchronous reindexing of a local CRAN repository either through the UI or using the REST API.

Through the UI, select your CRAN repository in the Tree Browser and select Recalculate Index from the right-click menu as shown below (requires Admin privileges)



To reindex a CRAN repository through the REST API, please refer to Calculate CRAN Repository Metadata.