Managing Disk Space Usage

Overview

Artifactory includes features to help you manage the amount of disk space used by your system. This is done by providing alerts, limiting the amount of space allocated for the output of automatic procedures, and by cleaning up unused artifacts in a controlled manner.

Page Contents

- Overview
  - Garbage Collection
  - Storage Quota
- Limiting the Number of Snapshots
  - Limiting Unique Docker Tags
- Deleting Unused Cached Artifacts
- Deleting Complete Versions
- User Plugins
- Manual Cleanup with the REST API
- Discarding old builds with JFrog CLI, Jenkins Artifactory plugin and Azure DevOps Extension

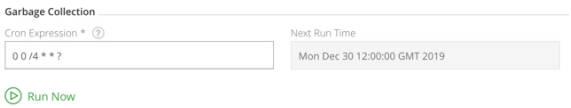
Garbage Collection

When an Artifactory user "deletes" a file, what is actually deleted is the reference from the Artifactory database to the physical file. Before actually deleting a file Artifactory must scan the system to ensure that there are no other users referencing the file. Scanning the system is very CPU intensive, and locks files while the scan is in process, and this may stress the development environment. Therefore this can be scheduled to run periodically as a "Garbage Collection" process during times when demands on the system are low.

This is done in the Artifactory UI **Administration** module under **Artifactory | Maintenance,** where you can schedule an automatic run of Garbage Collection with a

Cron expression

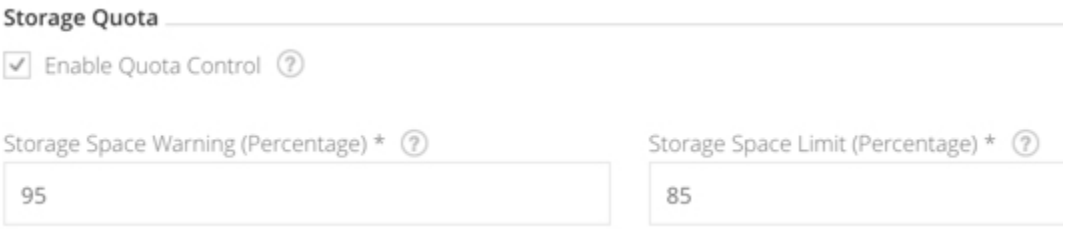
. You can also invoke an immediate run by clicking `Run Storage Garbage Collection`.



Storage Quota

To avoid running out of disk space Artifactory allows you to limit the storage space allocated for your repositories.

In the **Administration** module, under **Artifactory | Maintenance**, set **Enable Quota Control,** and specify **Storage Space Limit** to specify the percentage of disk space that you allocate for your repositories. An attempt to store binaries above the allocated storage percentage will fail with an error. You may also set **Storage Space Warning** to specify at what percentage of disk space usage to receive a warning from Artifactory.



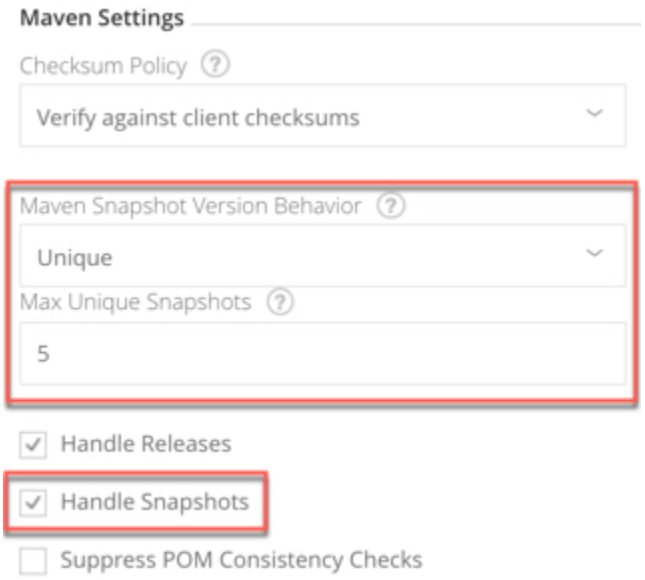
Limiting the Number of Snapshots

Working with snapshots is a standard development practice, however depending on the number of snapshots that are saved, this can use up large quantities of disk space.

To specify the maximum number of snapshots that may be stored, in the **Administration** module, select the **Repositories** and click the repository whose settings you want to edit.

In the **Basic s**ettings, check **Handle Snapshots** and then set the **Max Unique Snapshots** field**.** This value is zero by default, which means that all snapshots are saved.

> **Package Types Supported for Max Unique Snapshots**
>
> Artifactory supports the "Max Unique Snapshots" tag for the following package types: Maven, NuGet, Gradle, Ivy, Docker, SBT.



To avoid issues of concurrency, Artifactory requires that you store a minimum of 2 unique snapshots, however can can control the maximum number of snapshots that are stored.

> **Redundant snapshots are not deleted immediately**
>
> Every time you deploy a snapshot, Artifactory will check the value **Max Unique Snapshots** for the repository, and if exceeded will mark any excess old snapshots for deletion. Then, every 5 minutes, Artifactory runs a background process that deletes those oldest snapshots that have been marked. For example, if you set **Max Unique Snapshots** to 5 and deploy a sixth and seventh snapshot to the repository, then next time the background process runs, it will delete the two oldest snapshots.

Limiting Unique Docker Tags

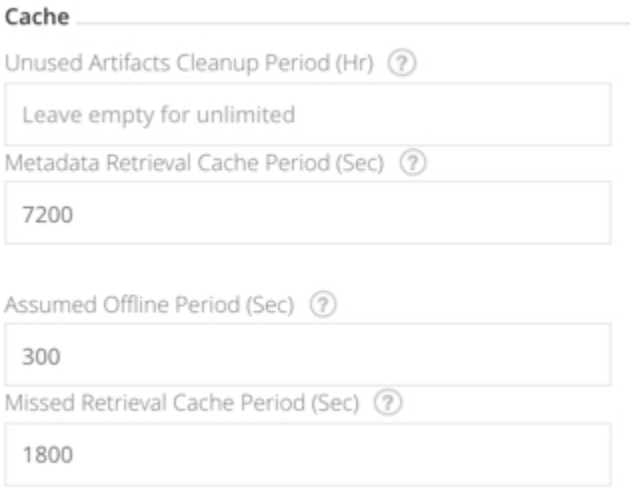
In the case of Docker registries, you can limit the number of unique tags using the **Max Unique Tags** field in the Local Docker Repositories configuration.

Deleting Unused Cached Artifacts

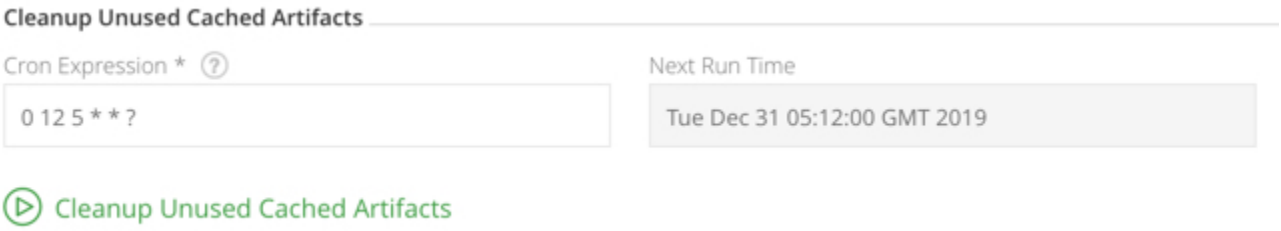
When working with remote repositories, to optimize performance, Artifactory locally caches and aggregates snapshots of remote artifacts that are being used. However, if at some point, these artifacts are no longer used, Artifactory can identify and remove them.

You can control how long an unused artifact will remain cached before it is eligible for cleanup. In the **Edit Repository** screen under **Advanced Settings**, specify the number of hours in the **Unused Artifacts Cleanup Period** field.

By default this value is set to zero which means that an artifacts from the corresponding repository are never removed from the cache.



Cleaning up unused cached artifacts can be scheduled to run automatically during times when demands on the system are low using a Cron expression in the **Administration** module under **Artifactory | Maintenance.** You can also invoke an immediate run by clicking "Run Unused Cached Artifacts Cleanup"



Recommended Frequency for Deleting Unused Cached Artifacts

Deleting unused cached artifacts is a resource-intensive operation, so to avoid concurrency and performance issues it is recommended to do it no more than once or twice a day, and preferably during "quiet time" such as outside of regular working hours.

Deleting Complete Versions

Artifactory supports a complete manual deletion of an installed version. This is fully described in Deleting a Version.

User Plugins

Artifactory supports cleanup by allowing you to write custom User Plugins which you can develop to meet your own specific cleanup requirements.

JFrog provides a number of cleanup scripts on GitHub which you can use as provided or modify to suit your own needs. For example the following artifactCleanup plugin deletes artifacts that have not been downloaded for a specified number of months.

Manual Cleanup with the REST API

Using the Artifactory REST API, you may write scripts to implement virtually any custom cleanup logic. This provides you with an extensive and flexible set of customization capabilities as provided by the REST API.

Examples:

- Use the REST API as described Artifacts Not Downloaded Since, to identify artifacts that have not been downloaded since a specific Java epoch, and then remove them.

- Use the REST API as described in Artifacts Created in Date Range to identify artifacts created within a specific date range and then remove them.

Discarding old builds with JFrog CLI, Jenkins Artifactory plugin and Azure DevOps Extension

When using JFrog CLI, Jenkins or Azure DevOps for continuous integration, you can configure a policy to discard old builds that are stored in Artifactory along with their artifacts.

For more details please refer to the Artifactory Plugin page of the following:

- JFrog CLI
- Jenkins Artifactory Plug-in
- Artifactory Azure DevOps Extension