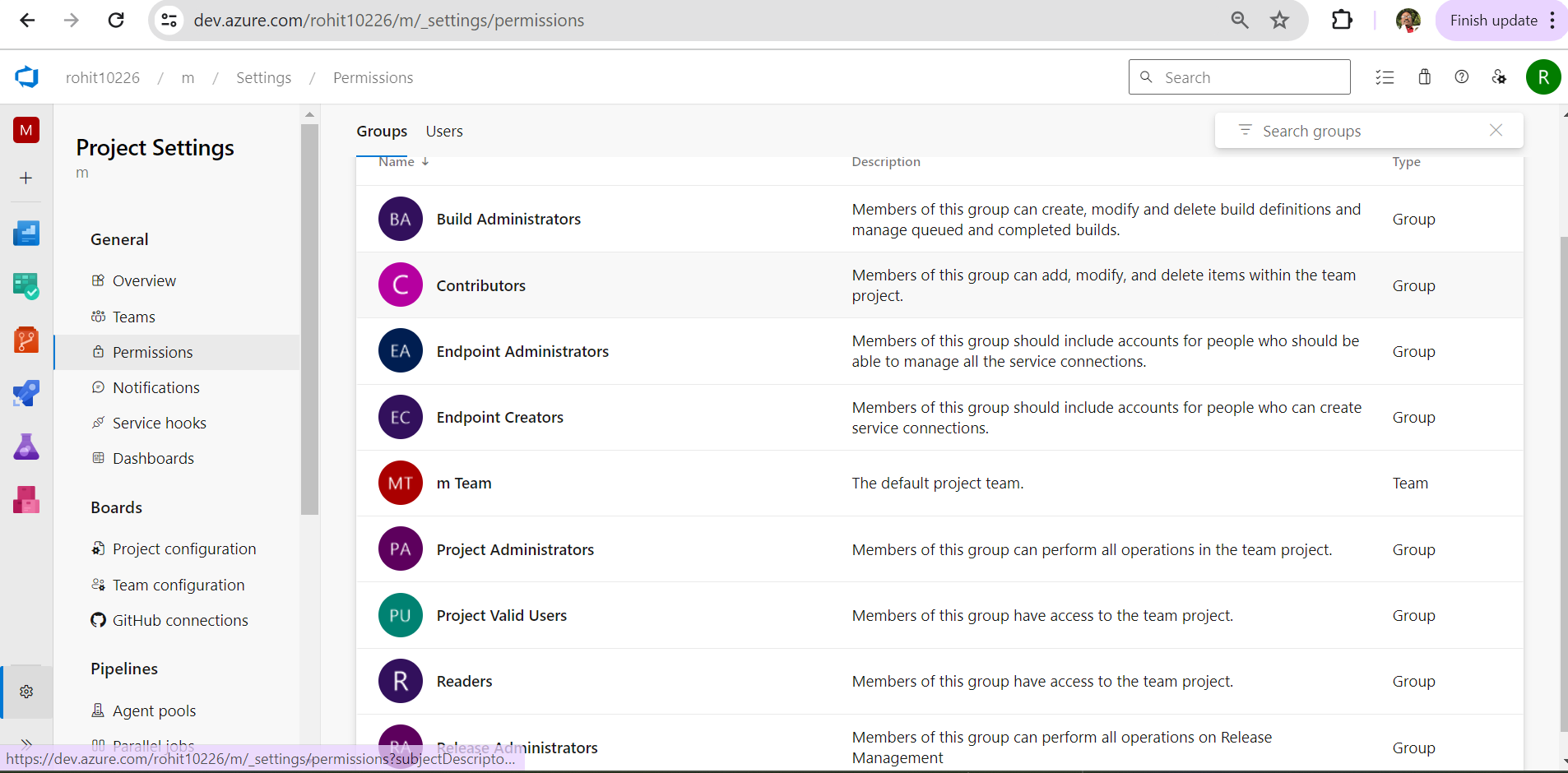
GROUPS OR USERS WHO USES PROJECT



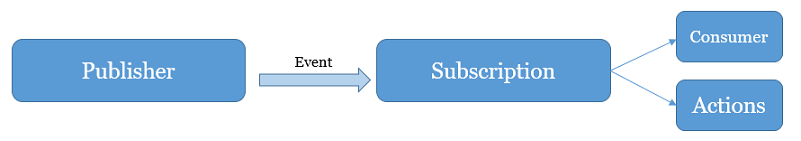
SERVICE HOOKS

Service hooks let you run tasks on other services when events happen in your project in Azure DevOps.

For example, you can create a card in Trello when a work item gets created or send a push notification to your team's mobile devices when a build fails. You can also use service hooks in custom apps and services as a more efficient way to drive activities when events happen in your projects.

Service hook publishers define a set of *events* that you can subscribe to. Subscriptions listen for these *events* and define actions to take based on the event.

Subscriptions also target consumers, which are external services that can run their own actions when events occur.



MASTER & SLAVE

In computer networking, master/slave is a model for a communication in which one device or process (known as the master) controls one or more other devices or processes (known as slaves). Once the master/slave relationship is established, the direction of control is always from the master to the slave(s).

AGENT POOL

An **agent pool** is a collection of agents. Instead of managing each agent individually, you organize agents into **agent pools**

**Difference between agent and deployment pool**

**Deployment group tags are labels that you assign to deployment targets based on their characteristics, such as environment, location, role, or configuration. Agent pools are collections of deployment agents that run on deployment targets or on separate machines.**

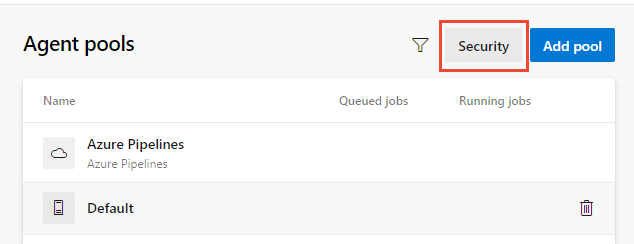
## **Agent pool security roles, project-level**

**You** [**add users to the following security roles**](https://learn.microsoft.com/en-us/azure/devops/pipelines/policies/set-permissions?view=azure-devops) **from the project-level admin context, Agent Pools page. For information on adding and managing agent pools, see** [**Agent pools**](https://learn.microsoft.com/en-us/azure/devops/pipelines/agents/pools-queues?view=azure-devops)**.**

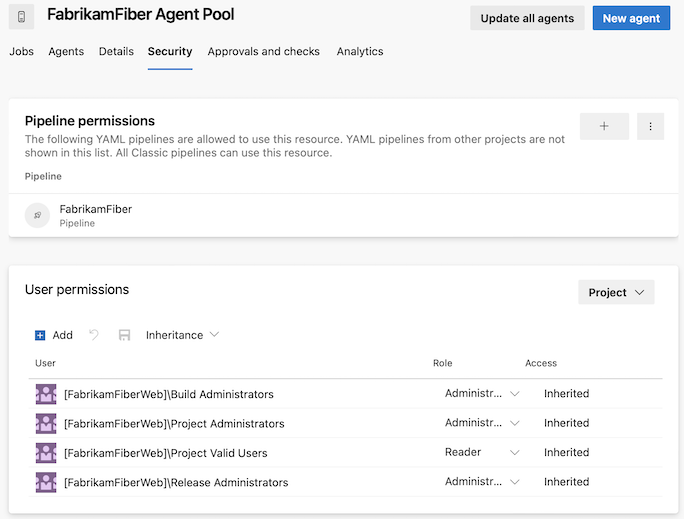
**Expand table**

| **Role (project-level)** | **Description** |
| --- | --- |
| **Reader** | **Can view the pool. You typically add operators to this role that are responsible for monitoring the build and deployment jobs in that pool.** |
| **User** | **Can view and use the pool when authoring build or release pipelines.** |
| **Creator** | **Can create and use the pool when authoring build or release pipelines.** |
| **Administrator** | **Can manage membership for all roles of the pool, as well as view and use the pools. The user that created a pool is automatically added to the Administrator role for that pool.** |

**To manage role settings for a project agent pool, open Project settings and choose Agent Pools.**

* **To set permissions for all pools within the project, choose Security, and then add a user and select their role.  
  **

**To set permissions for a specific pool, choose the pool and then Security. Under Pipeline permissions, you can see which pipelines have access to the pool. You can explicitly allow a pipeline by using the + button or allow all pipelines by using the ⋮ button. Under User permissions, you can add a user or group and select their role.**

****

## **Deployment group security roles**

**You** [**add users to the following roles**](https://learn.microsoft.com/en-us/azure/devops/pipelines/policies/set-permissions?view=azure-devops) **from Pipelines or Build and Release. For information on adding and managing deployment groups, see** [**Deployment groups**](https://learn.microsoft.com/en-us/azure/devops/pipelines/release/deployment-groups/?view=azure-devops)**.**

**Expand table**

| **Role** | **Description** |
| --- | --- |
| **Reader** | **Can only view deployment groups.** |
| **Creator** | **Can view and create deployment groups.** |
| **User** | **Can view and use but cannot manage or create deployment groups.** |
| **Administrator** | **Can administer roles, manage, view and use deployment groups.** |

## **Deployment pool security roles**

**You** [**add users to the following roles**](https://learn.microsoft.com/en-us/azure/devops/pipelines/policies/set-permissions?view=azure-devops) **from the collection-level admin context, Deployment Pools page. To create and manage deployment pools, see** [**Deployment groups**](https://learn.microsoft.com/en-us/azure/devops/pipelines/release/deployment-groups/?view=azure-devops)**.**

**Expand table**

| **Role** | **Description** |
| --- | --- |
| **Reader** | **Can only view deployment pools.** |
| **Service Account** | **Can view agents, create sessions, and listen for jobs from the agent pool.** |
| **User** | **Can view and use the deployment pool for creating deployment groups.** |
| **Administrator** | **Can administer, manage, view and use deployment pools.** |

## **Library asset security roles: Variable groups and secure files**

**You** [**add users to a library role**](https://learn.microsoft.com/en-us/azure/devops/pipelines/policies/set-permissions?view=azure-devops) **from Pipelines or Build and Release. To learn more about using these library assets, see** [**Variable groups**](https://learn.microsoft.com/en-us/azure/devops/pipelines/library/variable-groups?view=azure-devops) **and** [**Secure files**](https://learn.microsoft.com/en-us/azure/devops/pipelines/library/secure-files?view=azure-devops)

Expand table

| **Role** | **Description** |
| --- | --- |
| **Administrator** | **Can edit/delete and manage security for library items.** |
| **Creator** | **Can create library items.** |
| **Reader** | **Can only read library items.** |
| **User** | **Can consume library items in pipelines.** |

## **Service connection security roles**

**You** [**add users to the following roles**](https://learn.microsoft.com/en-us/azure/devops/pipelines/policies/set-permissions?view=azure-devops) **from the project-level admin context, Services page. To create and manage these resources, see** [**Service connections for build and release**](https://learn.microsoft.com/en-us/azure/devops/pipelines/library/service-endpoints?view=azure-devops)**.**

**Expand table**

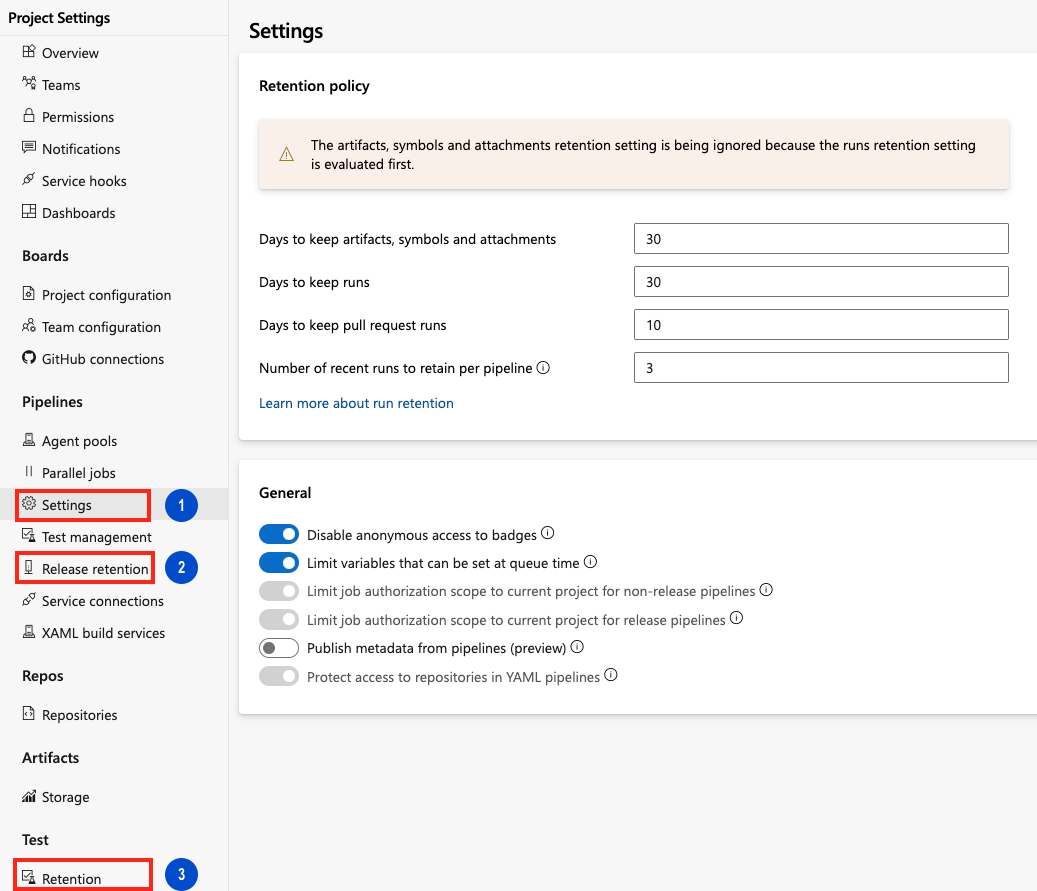
| **Role** | **Description** |
| --- | --- |
| **User** | **Can use the endpoint when authoring build or release pipelines.** |
| **Administrator** | **Can manage membership of all other roles for the service connection as well as use the endpoint to author build or release pipelines. The system automatically adds the user that created the service connection to the Administrator role for that pool.** |

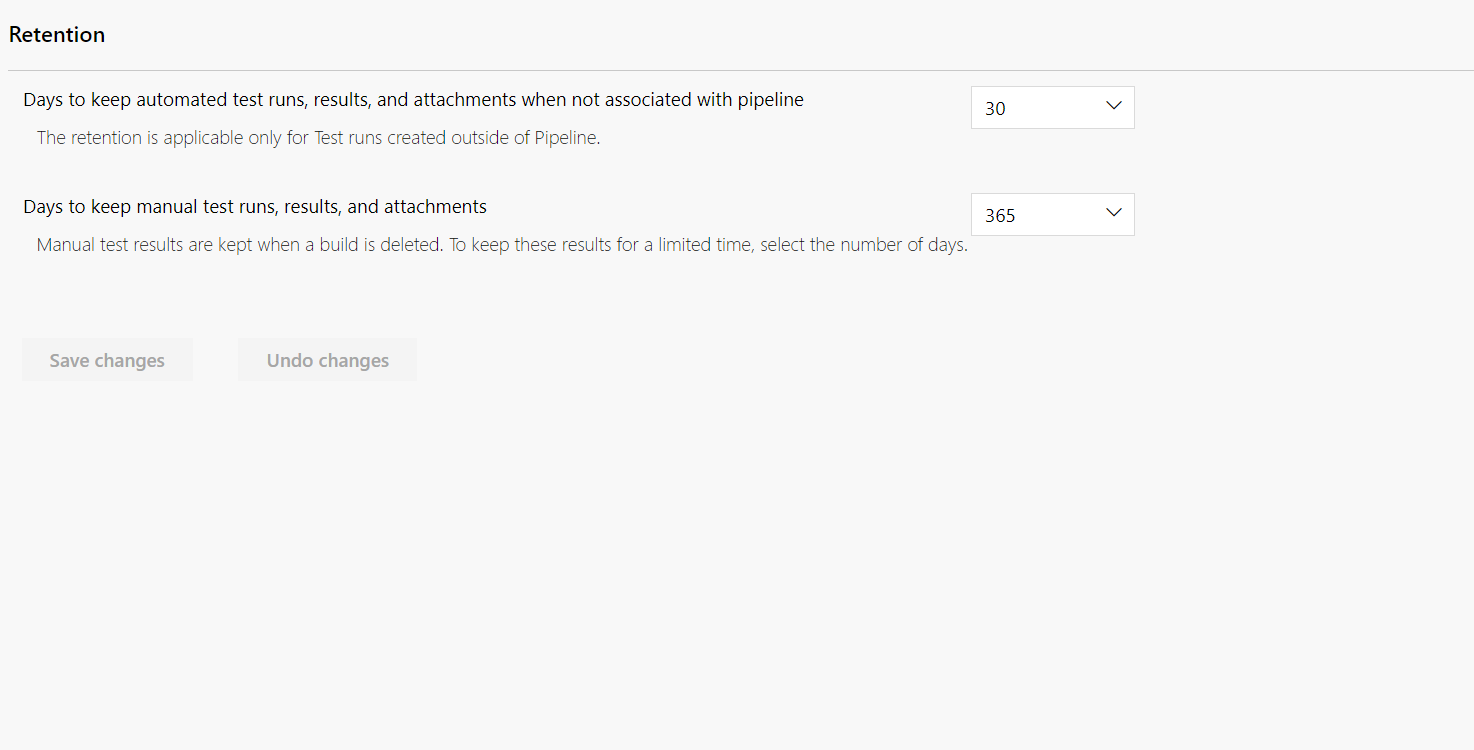
Retention policy

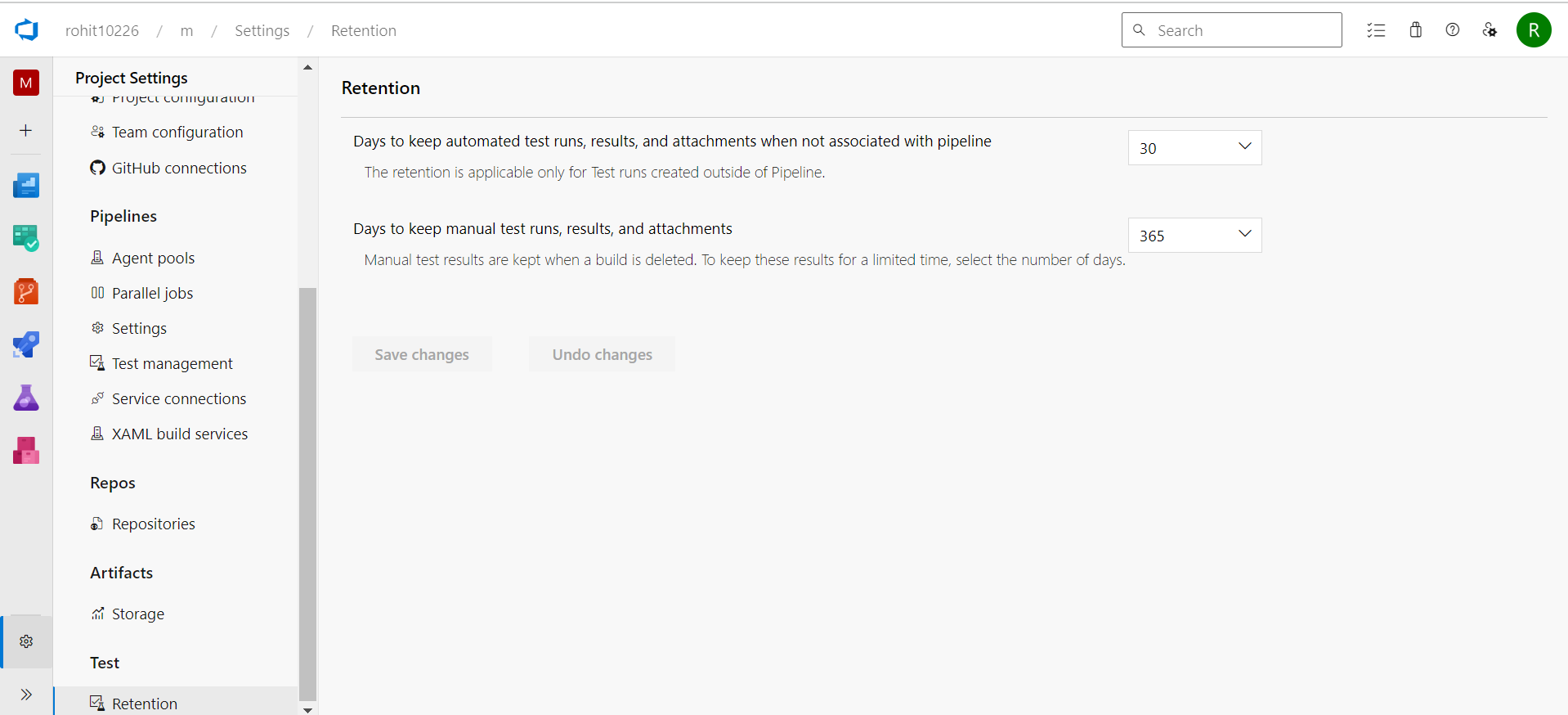
Retention policies let you set how long to keep runs, releases, and tests stored in the system. To save storage space, you want to delete older runs, tests, and releases.

The following retention policies are available in Azure DevOps in your Project settings:

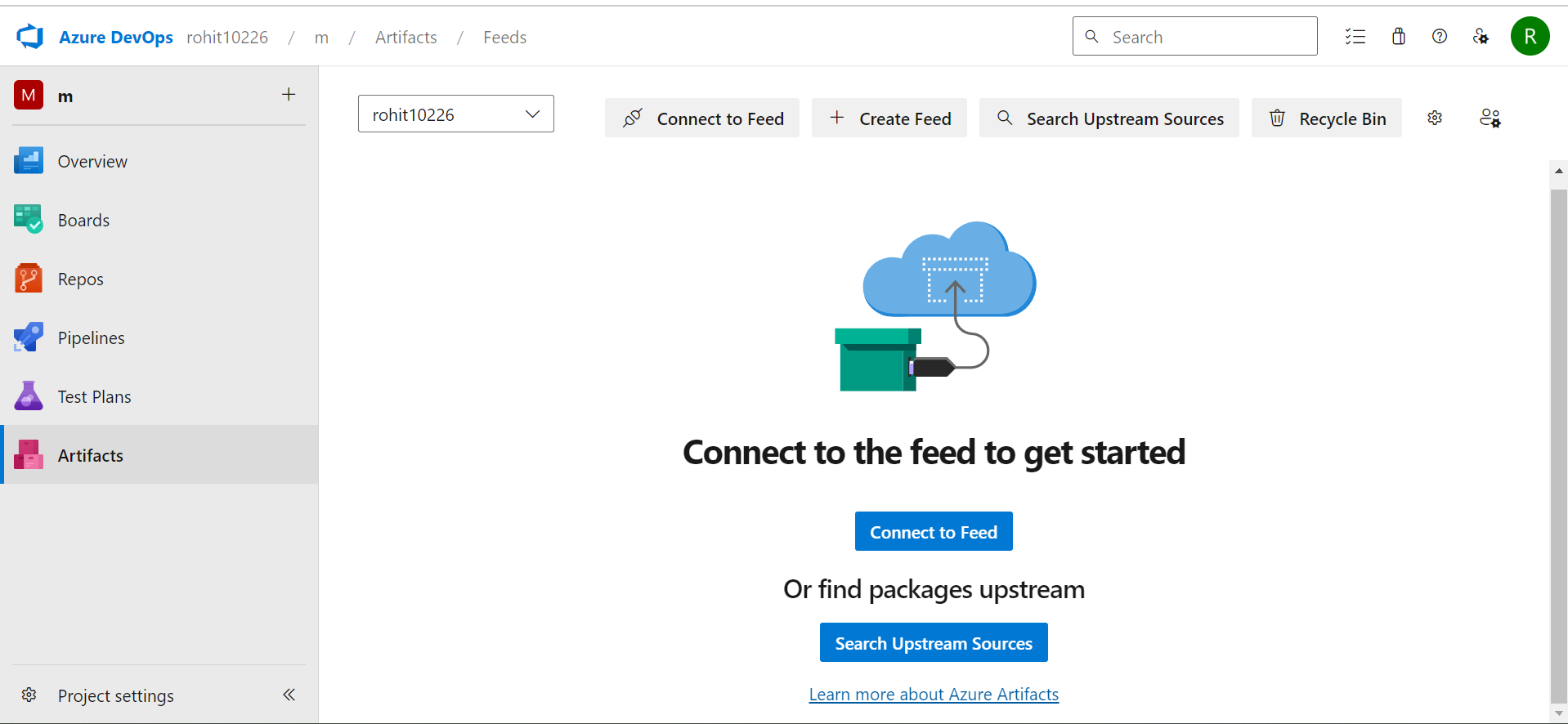
1. Pipeline - Set how long to keep artifacts, symbols, attachments, runs, and pull request runs.
2. Release (classic) - Set whether to save builds and view the default and maximum retention settings.
3. Test - Set how long to keep automated and manual test runs, results, and attachments.







Azure Artifacts enables developers to efficiently manage all their dependencies from one place. With Azure Artifacts, developers can publish packages to their feeds and share them within their team, across organizations, and even publicly across the internet. Azure Artifacts also allows developers to consume packages from different feeds and public registries such as NuGet.org or npmjs.com. Azure Artifacts supports the following package types: NuGet, npm, Python, Maven, Cargo, and Universal Packages.



##### **Q: How can I manage the retention duration for Artifacts?**

A: You can set up the retention policies to automatically delete packages. For more information, see [How to use retention policies to delete old packages](https://learn.microsoft.com/en-gb/azure/devops/artifacts/how-to/delete-and-recover-packages?view=azure-devops#delete-packages-automatically-with-retention-policies).

##### **Q: How do I delete specific packages?**

A: See [Delete and recover packages](https://learn.microsoft.com/en-gb/azure/devops/artifacts/how-to/delete-and-recover-packages?view=azure-devops) for more details.

##### **Q: Which artifacts contribute to my total billed storage?**

A: You will be charged for all package types (npm, NuGet, Python, Maven, Cargo, and Universal Packages), including packages stored from upstream sources. However, there will be no charges for Pipeline Artifacts and Pipeline Caching

# **What are feeds?**

Artifacts Feeds are organizational constructs that allow you to store, manage, and group your packages and control who to share it with. Feeds are not package-type dependent. You can store all the following package types in a single feed: npm, NuGet, Maven, Python, and Universal packages.

## **Project-scoped vs Organization-scoped feeds**

Previously, all feeds were scoped to an organization, they could be viewed and accessed in the Azure Artifacts hub from any project within an organization. With the introduction of public feeds, we also introduced project-scoped feeds. This type of feed can only be accessed from within the hosting project.

Only project-scoped feeds can be made public. You can learn more about [public feeds](https://learn.microsoft.com/en-us/azure/devops/artifacts/concepts/feeds?view=azure-devops&source=recommendations#public-feeds) later in this article. See [Feeds visibility](https://learn.microsoft.com/en-us/azure/devops/artifacts/feeds/project-scoped-feeds?view=azure-devops) to understand the differences between project-scoped and organization-scoped feeds.

Note

To access a feed in a different organization, a user must be given access to the project hosting that feed.

## **Public feeds**

Public feeds are used to share your packages publicly with anyone on the Internet. Users won't have to be a member of your organization or your enterprise. They can access the packages even if they don't have an Azure DevOps account.

Public feeds are project-scoped feeds and it will inherit the visibility settings of the hosting project.

There some important things to note regarding public feeds:

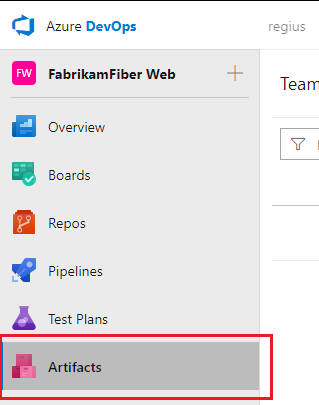
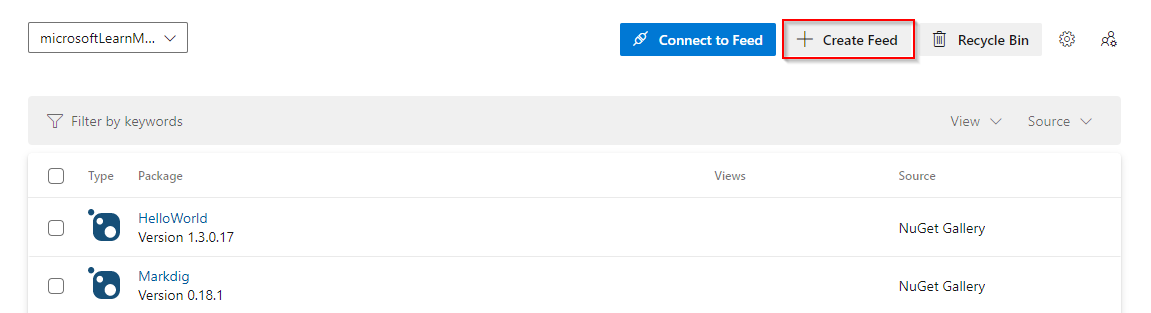
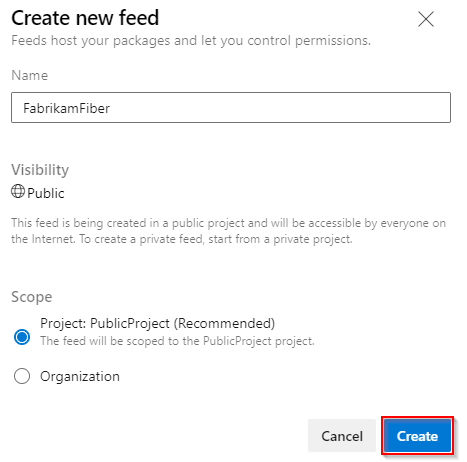
* Public feeds can only be created inside of public projects.
* Public feeds aren't intended as a replacement for existing package management platforms (NuGet.org, npmjs.com, etc.).
* Public users cannot currently download universal packages. All other package types are supported for public access.

Note

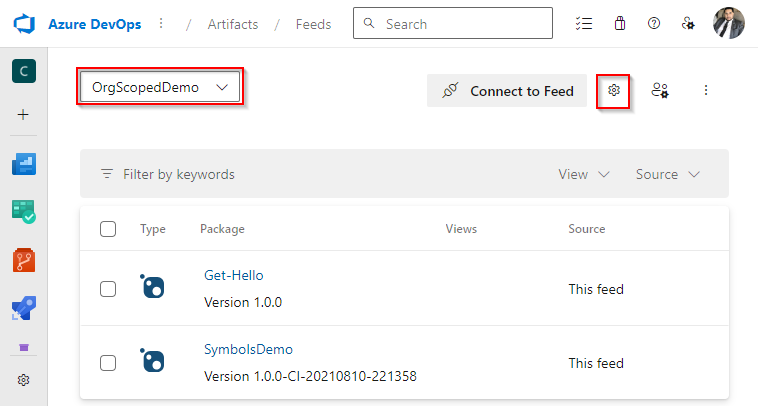
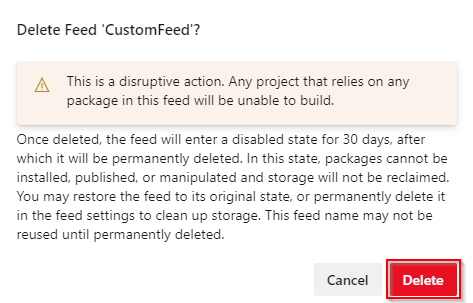
All feed views in a public project are accessible to everyone on the internet.

## **Create public feeds**

Public feeds are project-scoped feeds in a public project.

1. Select Artifacts.  
   
2. Select Create Feed.  
   
3. Give your feed a Name, and then select Project for your feed's scope.  
   
4. Select Create when you are done.

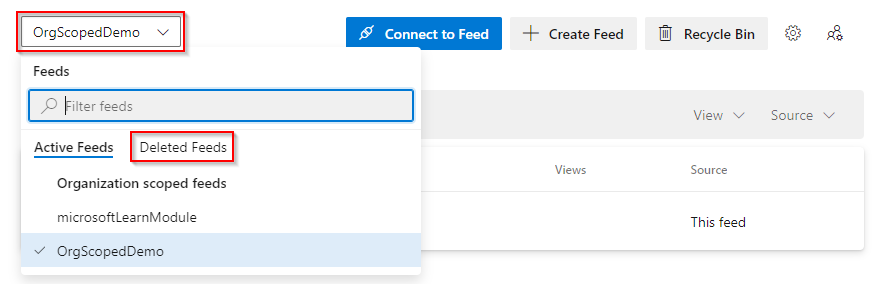
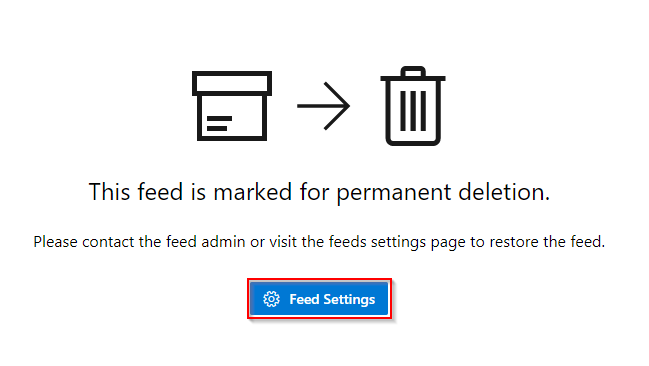
## **Delete a feed**

1. Select Artifacts, and then select your feed from the dropdown menu.
2. Select the gear icon gear icon to navigate to your feed's settings.  
   
3. Select Delete feed.
4. Select Delete when you are ready.  
   

## **Restore deleted feeds**

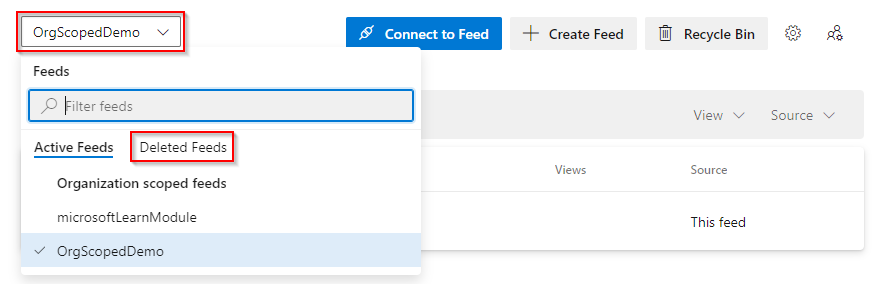
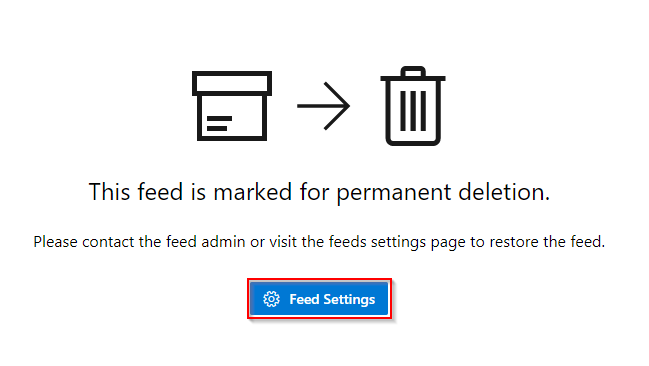
If you accidentally delete a feed, Azure Artifacts provides a 30 days window to recover your feed to its original state. After the 30 days, the feed will be deleted permanently. During the recovery window, the name of the feed remains reserved, packages are unavailable for download, and write access is suspended for that feed.

You can view the feeds that are pending permanent deletion in the feed picker dropdown list under the Deleted Feeds tab.

1. Select Artifacts.
2. Select the feed picker dropdown menu, and then select Deleted Feeds  
   
3. Select the feed you want to restore, and then select Feed Settings.  
   
4. Select Restore Feed.

## **Permanently deleting a feed**

A feed pending deletion will still use storage space. If you want to permanently delete your feed before the 30 days period is up, you can do this as follows:

1. Select Artifacts.
2. Select the feed picker dropdown menu, and then select Deleted Feeds  
   
3. Select the feed you want to restore, and then select Feed Settings.  
   
4. Select Permanently Delete Feed, and then select Delete.

Once the feed is permanently deleted, users won't be able to view or restore its packages. The feed name will be available for reuse 15 minutes after the deletion.

## **Count limits of Artifacts**

* 5000 versions per package ID.
* Unlimited package IDs per feed.
* 20 upstreams per package type per feed.

Note

You can use [retention policies](https://learn.microsoft.com/en-us/azure/devops/artifacts/how-to/delete-and-recover-packages?view=azure-devops#delete-packages-automatically-with-retention-policies) to automatically delete older package versions.

## **Size limits of artifacts**

* NuGet packages: limited to 500 MB per file.
* Npm packages: limited to 500 MB per file.
* Maven packages: limited to 500 MB per file.
* Python packages: limited to 500 MB per file.
* Universal Packages: up to 4 TB per file (Recommended for large binary files).

Note

Universal Packages are only available in Azure DevOps Services.

# **Promote packages and manage feed views in Azure Artifacts**

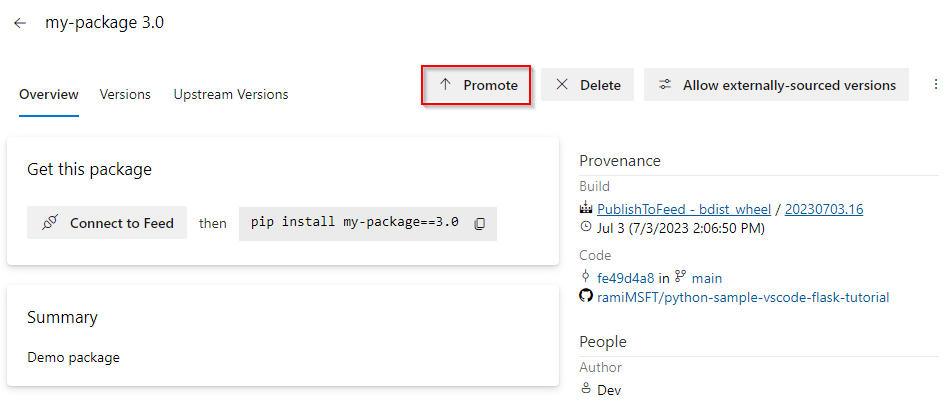
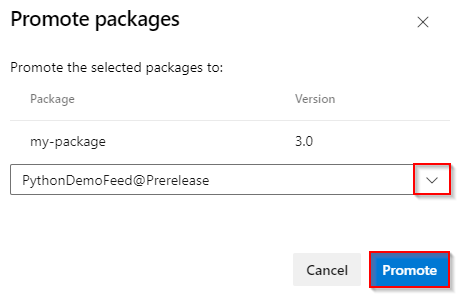
Feed views are a way to enable users to share some packages while keeping other packages private. Views filter the feed to a subset of packages that meet a set of criteria defined by that view.

By default, Azure Artifacts comes with three views: @Local, @Prerelease, and @Release. @local is the default view that contains all the published packages and all the packages saved from upstream sources. All views support NuGet, npm, Maven, Python, and Universal packages.

Note

Azure Artifacts only supports publishing and restoring packages from and to the default view - *@Local*.

## **Promote packages**

1. Sign in to your Azure DevOps organization, and then navigate to your project.
2. Select Artifacts, and then select your feed from the dropdown menu.
3. Select the package you wish to promote, and then select Promote.  
   
4. Select a view from the dropdown menu, and then select Promote.  
   

Note

Package demotion is not supported. If you want this feature to be added to a future release, please feel free to Suggest a feature on [Azure DevOps Developer Community](https://developercommunity.visualstudio.com/spaces/21/index.html).

## **Promote packages using the REST API**

In addition to using the Azure Artifacts user interface, you can also promote packages using the REST API. The URI varies based on the package type:

Use the actual user-facing name and version of the package for the {packageName} and {packageVersion} fields, respectively. If your feed is organization-scoped, omit the {project} field.

The body of the request is a [JSON Patch](https://jsonpatch.com/) document adding the view to the end of the views array. See [Get started with the REST API](https://learn.microsoft.com/en-us/azure/devops/integrate/how-to/call-rest-api?view=azure-devops) and the [REST API samples](https://learn.microsoft.com/en-us/azure/devops/integrate/get-started/rest/samples?view=azure-devops) for more information on how to interact with Azure DevOps REST API.

* [NuGet](https://learn.microsoft.com/en-us/azure/devops/artifacts/feeds/views?view=azure-devops&source=recommendations&tabs=nuget#tabpanel_1_nuget)
* [Npm](https://learn.microsoft.com/en-us/azure/devops/artifacts/feeds/views?view=azure-devops&source=recommendations&tabs=nuget#tabpanel_1_npm)
* [Python](https://learn.microsoft.com/en-us/azure/devops/artifacts/feeds/views?view=azure-devops&source=recommendations&tabs=nuget#tabpanel_1_python)
* [Maven](https://learn.microsoft.com/en-us/azure/devops/artifacts/feeds/views?view=azure-devops&source=recommendations&tabs=nuget#tabpanel_1_maven)
* [Universal Packages](https://learn.microsoft.com/en-us/azure/devops/artifacts/feeds/views?view=azure-devops&source=recommendations&tabs=nuget#tabpanel_1_universalpackages)
* Organization scoped feed:
* HTTP
* Copy

PATCH https://pkgs.dev.azure.com/{organization}/\_apis/packaging/feeds/{feedId}/nuget/packages/{packageName}/versions/{packageVersion}?api-version=7.1-preview.1

* Project scoped feed:
* HTTP
* Copy

PATCH https://pkgs.dev.azure.com/{organization}/{project}/\_apis/packaging/feeds/{feedId}/nuget/packages/{packageName}/versions/{packageVersion}?api-version=7.1-preview.1

* Use [JsonPatchOperation](https://learn.microsoft.com/en-us/rest/api/azure/devops/artifactspackagetypes/nuget/update%20package%20version?view=azure-devops-rest-5.1&preserve-view=true#jsonpatchoperation) to construct the body of your request. See [NuGet - update package version](https://learn.microsoft.com/en-us/rest/api/azure/devops/artifactspackagetypes/nuget/update%20package%20version?view=azure-devops-rest-7.1&preserve-view=true) for more details.
* Example:

HTTP

Copy

PATCH https://pkgs.dev.azure.com/fabrikam-fiber-inc/litware/\_apis/packaging/feeds/litware-tools/nuget/packages/LitWare.Common/versions/1.0.0?api-version=5.1-preview.1 HTTP/1.1

Content-Type: application/json-patch+json

{

"views": {

"op": "add",

"path": "/views/-",

"value": "Release"

}

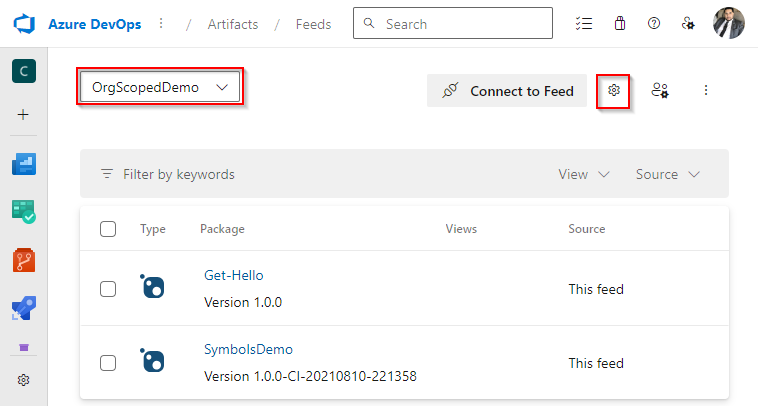
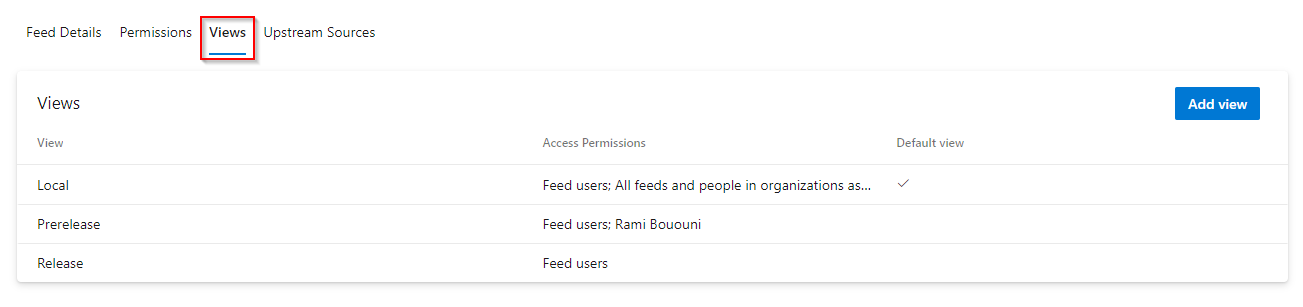
}

## **Manage views**

You can create your own views or rename and delete existing ones from your feed's settings.

Note

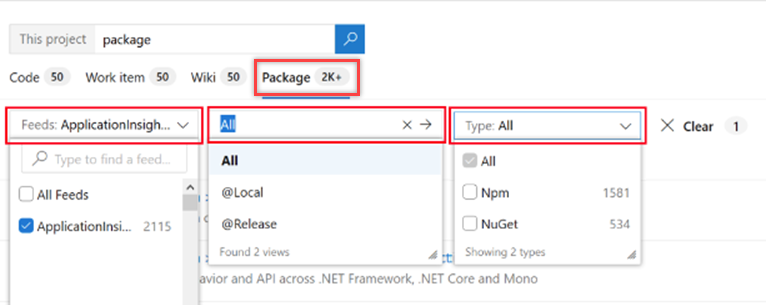
All feed views in a public project are accessible to everyone on the internet.

1. Select Artifacts.
2. Select your feed from the dropdown menu.
3. Select the gear icon  to access your feed's settings.  
   
4. Select Views.  
   
5. Select a view, and then select Edit to edit your view or select Add view if you want to add a new view.
6. Select Save when you're done.

Important

For public feeds, if you change the access permissions of a certain view to Specific people your view will not be available as an upstream source.

## **Search packages**

1. Sign in to your project (https://dev.azure.com/{your\_organization}/{your\_project}).
2. Enter "package" in the search box.
3. Select from the dropdown menus to search by feeds, views, or package types.  
   

You can search within all feeds of the organization by default, regardless of the project you’re in.

The Views filter shows up only when you select a single feed from the Feeds filter. This filter lets you display packages from a particular view.

You can use the Type filter to choose the package type you want to search for (for example, NuGet packages).

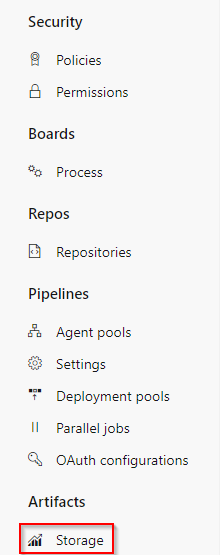
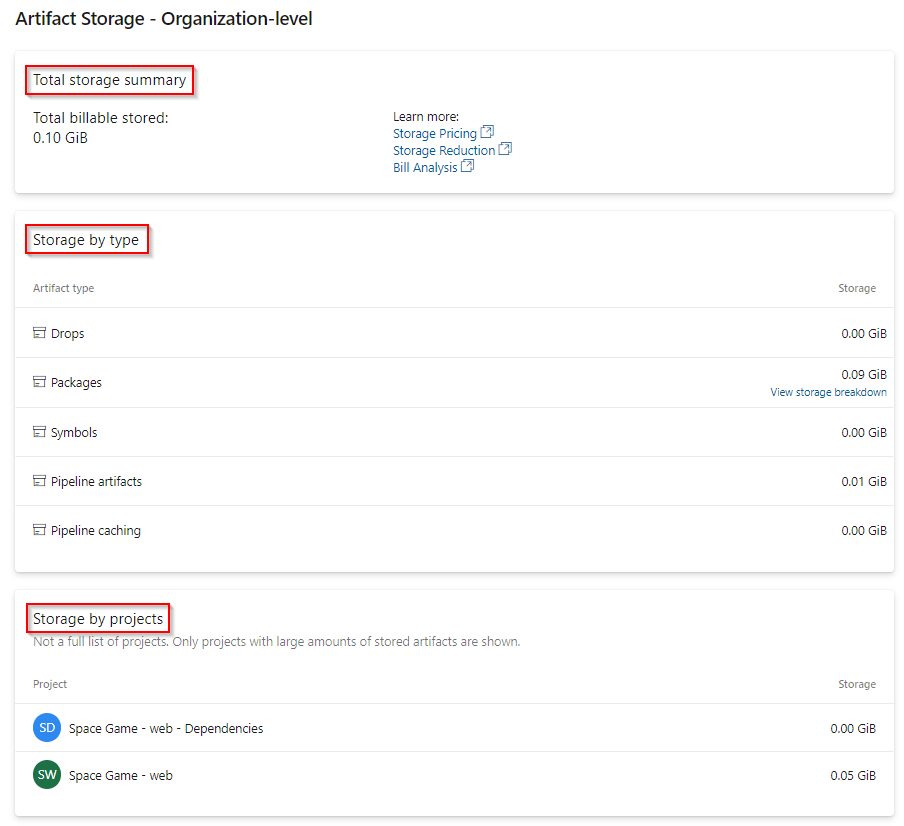
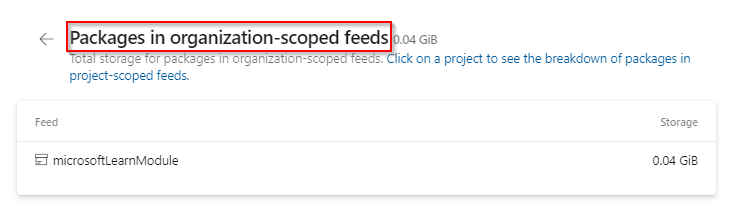
# **Artifacts storage consumption**

Azure Artifacts adopts a consumption-based billing model for all the package types it supports, such as NuGet, npm, Python, Maven, and Universal packages. The free-tier plan provides a storage capacity of two Gibibytes (GiB) to store various package types. If you exceed the storage limit, you can either upgrade to a paid subscription or remove some of your existing artifacts.

The artifact storage UI available in your organization/project settings allows you to monitor your storage usage at the organization and project levels. Storage is also grouped by project and artifact type.

## **Organization-level storage**

The organization-level view provides an overview of your total storage usage as well as the storage consumption by artifact type and by project.

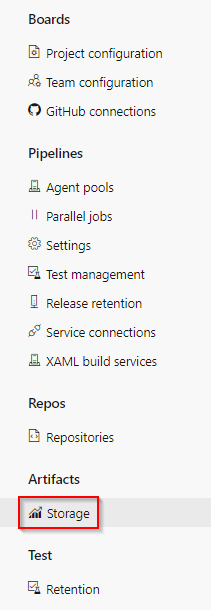
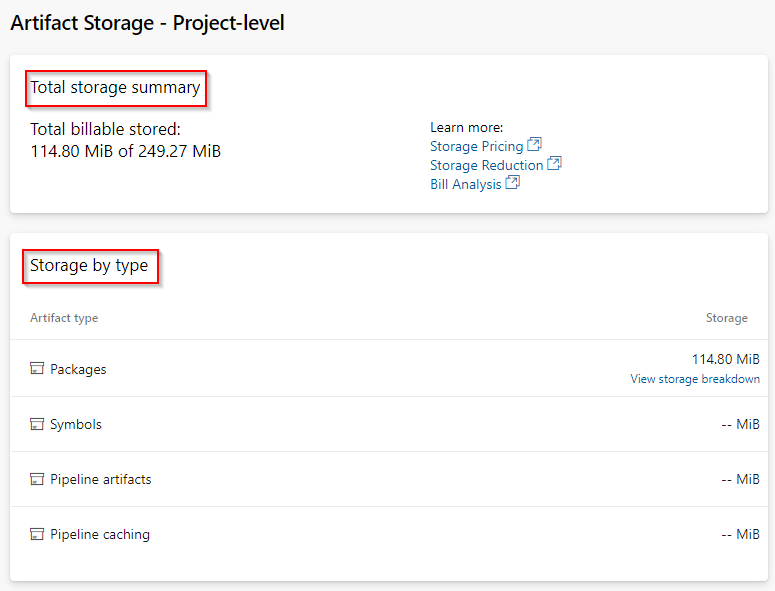
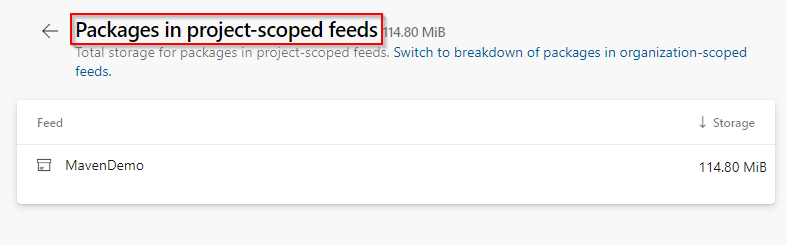
1. Sign in to your Azure DevOps organization.
2. From within your organization, select gear icon Organization settings.
3. Select Storage from the left pane.  
   
4. You can view your total storage summary, storage by artifact type, and storage by projects in your organization.  
   
5. Select View storage breakdown from Storage by type to view the total storage for packages in your organization-scoped feeds.  
   

Note

The list of Storage by projects only includes projects with the largest storage consumption and not the complete list of projects in your organization.

## **Project-level storage**

The project-level view provides an overview of your total storage usage as well as the storage consumption by artifact type.

1. Sign in to your Azure DevOps organization, and then navigate to your project.
2. From within your project, select gear icon Project settings.
3. Select Storage from the left pane.  
   
4. You can view your total storage summary and storage consumption by artifact type for your project.  
   
5. Select View storage breakdown from Storage by type to view the total storage for packages in project-scoped feeds.  
   

Note

Azure Artifacts provides 2GiB of free storage for each organization. After reaching the maximum storage limit, you need to [set up bil](https://learn.microsoft.com/en-us/azure/devops/organizations/billing/set-up-billing-for-your-organization-vs?view=azure-devops)ling

# **About pipeline tests**

| **Term** | **Definition** |
| --- | --- |
| Duration | Time elapsed in execution of a test, test run, or entire test execution in a build or release pipeline. |
| Owner | Owner of a test or test run. The test owner is typically specified as an attribute in the test code. See [Publish Test Results](https://learn.microsoft.com/en-us/azure/devops/pipelines/tasks/reference/publish-test-results-v2) task to view the mapping of the Owner attribute for supported test result formats. |
| Failing build | Reference to the build having the first occurrence of consecutive failures of a test case. |
| Failing release | Reference to the release having the first occurrence of consecutive failures of a test case. |
| Outcome | There are 15 possible outcomes for a test result: Aborted, Blocked, Error, Failed, Inconclusive, In progress, None, Not applicable, Not executed, Not impacted, Passed, Paused, Timeout, Unspecified, and Warning.  Some of the commonly used outcomes are:  - Aborted: Test execution terminated abruptly due to internal or external factors, e.g., bad code, environment issues.  - Failed: Test not meeting the desired outcome.  - Inconclusive: Test without a definitive outcome.  - Not executed: Test marked as skipped for execution.  - Not impacted: Test not impacted by the code change that triggered the pipeline.  - Passed: Test executed successfully.  - Timeout: Test execution duration exceeding the specified threshold. |
| Flaky test | A test with non-deterministic behavior. For example, the test may result in different outcomes for the same configuration, code, or inputs. |
| Filter | Mechanism to search for the test results within the result set, using the available attributes. [Learn more](https://learn.microsoft.com/en-us/azure/devops/pipelines/test/review-continuous-test-results-after-build?view=azure-devops). |
| Grouping | An aid to organizing the test results view based on available attributes such as Requirement, Test files, Priority, and more. Both [test report](https://learn.microsoft.com/en-us/azure/devops/pipelines/test/review-continuous-test-results-after-build?view=azure-devops) and [test analytics](https://learn.microsoft.com/en-us/azure/devops/pipelines/test/test-analytics?view=azure-devops) provide support for grouping test results. |
| Pass percentage | Measure of the success of test outcome for a single instance of execution or over a period of time. |
| Priority | Specifies the degree of importance or criticality of a test. Priority is typically specified as an attribute in the test code. See [Publish Test Results](https://learn.microsoft.com/en-us/azure/devops/pipelines/tasks/reference/publish-test-results-v2) task to view the mapping of the Priority attribute for supported test result formats. |
| Test analytics | A [view of the historical test data](https://learn.microsoft.com/en-us/azure/devops/pipelines/test/test-analytics?view=azure-devops) to provide meaningful insights. |
| Test case | Uniquely identifies a single test within the specified branch. |
| Test files | Group tests based on the way they are packaged; such as files, DLLs, or other formats. |
| Test report | A [view of single instance of test execution](https://learn.microsoft.com/en-us/azure/devops/pipelines/test/review-continuous-test-results-after-build?view=azure-devops) in the pipeline that contains details of status and help for troubleshooting, traceability, and more. |
| Test result | Single instance of execution of a test case with a specific outcome and details. |
| Test run | Logical grouping of test results based on:  - Test executed using built-in tasks: All tests executed using a single task such as [Visual Studio Test](https://learn.microsoft.com/en-us/azure/devops/pipelines/tasks/reference/vstest-v2), [Ant](https://learn.microsoft.com/en-us/azure/devops/pipelines/tasks/reference/ant-v1), [Maven](https://learn.microsoft.com/en-us/azure/devops/pipelines/tasks/reference/maven-v3), [Gulp](https://learn.microsoft.com/en-us/azure/devops/pipelines/tasks/reference/gulp-v1), [Grunt](https://learn.microsoft.com/en-us/azure/devops/pipelines/tasks/reference/grunt-v0) or [Xcode](https://learn.microsoft.com/en-us/azure/devops/pipelines/tasks/reference/xcode-v5) will be reported under a single test run  - Results published using [Publish Test Results](https://learn.microsoft.com/en-us/azure/devops/pipelines/tasks/reference/publish-test-results-v2) task: Provides an option to group all test results from one or more test results files into a single run, or individual runs per file  - Tests results published using API(s): [API(s)](https://learn.microsoft.com/en-us/rest/api/azure/devops/test/runs) provide the flexibility to create test runs and organize test results for each run as required. |
| Traceability | Ability to [trace](https://learn.microsoft.com/en-us/azure/devops/pipelines/test/requirements-traceability?view=azure-devops) forward or backward to a requirement, bug, or source code from a test result. |