

Deleting and restoring a package

In this article

- Package deletion and restoration support on GitHub
- Packages API support
- Required permissions to delete or restore a package
- Deleting a package version
- Deleting an entire package
- Restoring packages

Learn how to delete or restore a package.

Package deletion and restoration support on GitHub



On GitHub if you have the required access, you can delete:

- an entire private package
- an entire public package, if there's not more than 5000 downloads of any version of the package
- a specific version of a private package
- a specific version of a public package, if the package version doesn't have more than 5,000 downloads

Note:

- You cannot delete a public package if any version of the package has more than 5,000 downloads. In this scenario, contact us through the [GitHub Support portal](#) for further assistance.
- When deleting public packages, be aware that you may break projects that depend on your package.

On GitHub, you can also restore an entire package or package version, if:

- You restore the package within 30 days of its deletion.
- The same package namespace is still available and not used for a new package.

Packages API support

GitHub Packages only supports authentication using a personal access token (classic). For more information, see "[Managing your personal access tokens](#)."

You can use the REST API to manage your packages. For more information, see the "[Packages](#)."

Note: The ability for GitHub Actions workflows to delete and restore packages using the REST API is currently in public beta and subject to change.

With registries that support granular permissions, you can use a `GITHUB_TOKEN` in a GitHub Actions workflow to delete or restore packages using the REST API. The token must have `admin` permission to the package. If your workflow publishes a package, the `admin` role is granted by default to the repository where the workflow is stored. For existing packages not published by a workflow, you need to grant the repository the `admin` role to be able to use a GitHub Actions workflow to delete or restore packages using the REST API. For more information, see "[Configuring a package's access control and visibility](#)."

For certain registries, you can use GraphQL to delete a version of a private package.

You cannot use the GitHub Packages GraphQL API with registries that support granular permissions. For the registries that **only** support repository-scoped permissions, and can be used with the GraphQL API, see "[About permissions for GitHub Packages](#)."

Required permissions to delete or restore a package



With registries that support granular permissions, you can choose to allow packages to be scoped to a user or an organization, or linked to a repository.

To delete a package that has granular permissions separate from a repository, such as container images stored at `https://containers.HOSTNAME/NAMESPACE/PACKAGE-NAME` (where `NAMESPACE` is the name of the personal account or organization to which the package is scoped), you must have admin access to the package. For more information, see "[About permissions for GitHub Packages](#)."

For packages that inherit their access permissions from repositories, you can delete a package if you have admin permissions to the repository.

Some registries **only** support repository-scoped packages. For a list of these registries, see "[About permissions for GitHub Packages](#)."

Deleting a package version

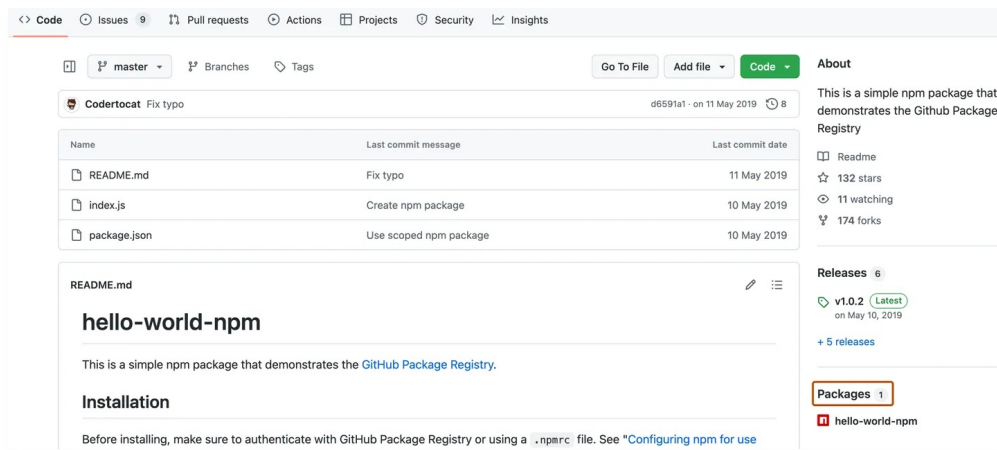


Deleting a version of a repository-scoped package on GitHub



To delete a version of a repository-scoped package, you must have admin permissions to the repository in which the package is published. For more information, see "[Required permissions](#)."

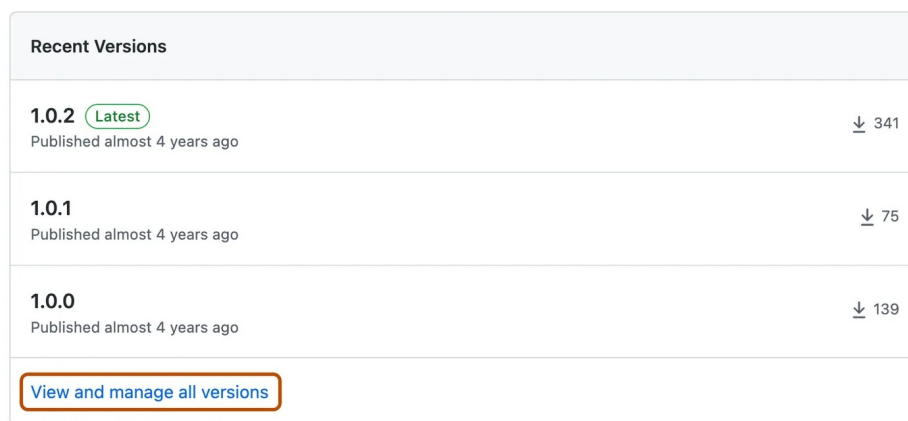
- 1 On your GitHub Enterprise Server instance, navigate to the main page of the repository.
- 2 In the right sidebar of your repository, click **Packages**.



3 Search for and then click the name of the package that you want to manage.

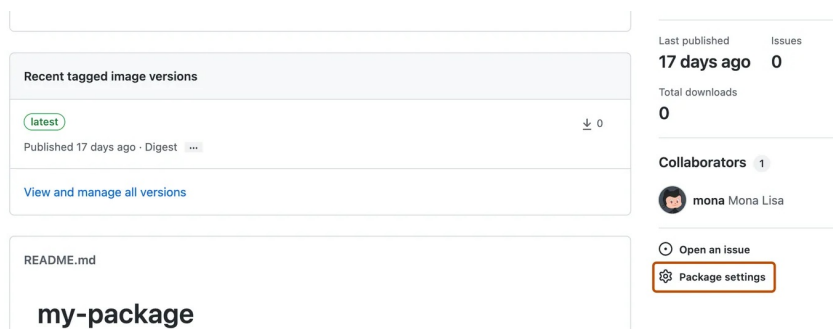
4 Navigate to where you can manage versions for your type of package.

- If your package is a container, under the "Recent Versions" section, click **View and manage all versions**.

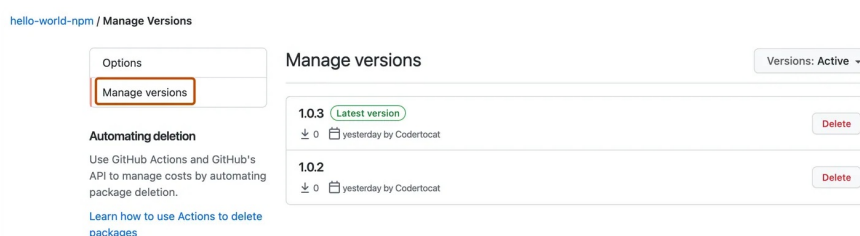


- For types of packages other than containers:

a. On the right-hand side, click **Package settings**.

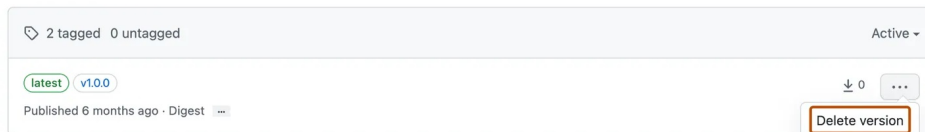


b. On the left click **Manage versions**.



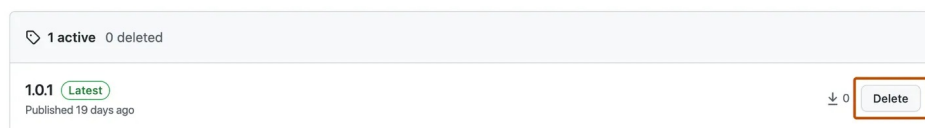
- 5 In the list of packages, find the version of the package that you want to delete.
- *If your package is a container*, to the right of the package version click **...**, then select **Delete version** from the dropdown menu.

hello-world / All Versions



- *For types of packages other than containers*, to the right of the package version click **Delete**.

hello-world-npm / All Versions



- 6 To confirm deletion, type the package name and click **I understand the consequences, delete this version**.

Deleting a version of a repository-scoped package with GraphQL [↗](#)

For certain registries, you can use GraphQL to delete a version of a private package.

You cannot use the GitHub Packages GraphQL API with registries that support granular permissions. For the registries that **only** support repository-scoped permissions, and can be used with the GraphQL API, see "[About permissions for GitHub Packages](#)." For information on using the REST API instead, see the "[Packages](#)."

Use the `deletePackageVersion` mutation in the GraphQL API. You must use a personal access token (classic) with the `read:packages`, `delete:packages`, and `repo` scopes. For more information about personal access tokens (classic), see "[Introduction to GitHub Packages](#)."

The following example demonstrates how to delete a package version, using a `packageVersionId` of `MDIy0lJlZ2lzdHJ5UGFja2FnZVZlcnNpb243MTEwNg`.

```
curl -X POST \
-H "Accept: application/vnd.github.package-deletes-preview+json" \
-H "Authorization: bearer TOKEN" \
-d '{"query":"mutation { deletePackageVersion(input: {packageVersionId:\\"MDIy0lJlZ2lzdHJ5UGFja2FnZVZlcnNpb243MTEwNg==\"}) { success }}"}' \
HOSTNAME/graphql
```

To find all of the private packages you have published to GitHub Packages, along with the version IDs for the packages, you can use the `packages` connection through the `repository` object. You will need a personal access token (classic) with the `read:packages` and `repo` scopes. For more information, see the [packages](#) connection or the [PackageOwner](#) interface.

For more information about the `deletePackageVersion` mutation, see "[Mutations](#)."

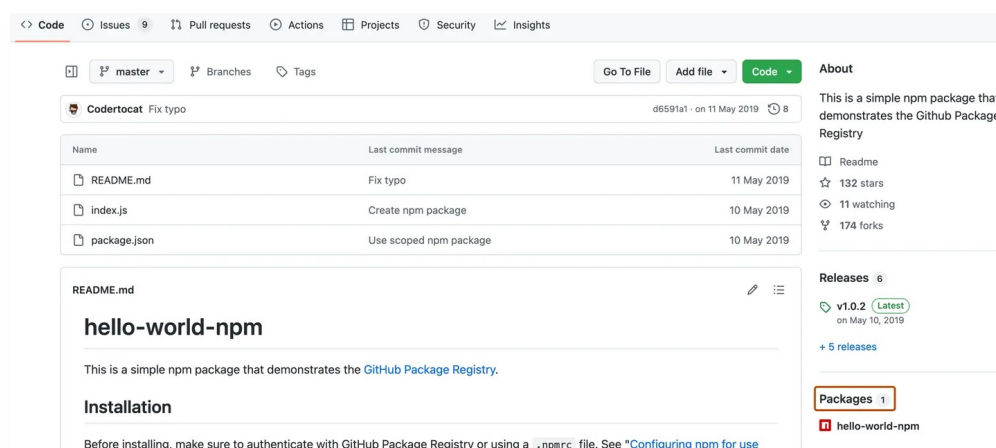
You cannot directly delete an entire package using GraphQL, but if you delete every version of a package, the package will no longer show on GitHub Enterprise Server.


Deleting an entire package [↗](#)

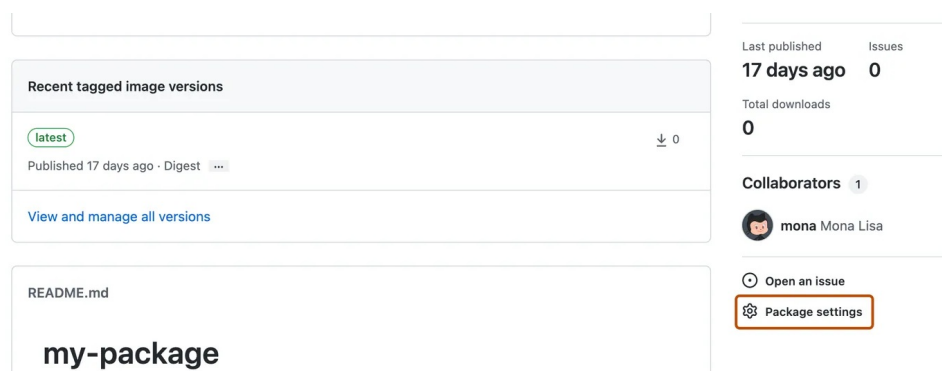
Deleting an entire repository-scoped package on GitHub [↗](#)

To delete an entire repository-scoped package, you must have admin permissions to the repository that owns the package. For more information, see "[Required permissions](#)."

- 1 On your GitHub Enterprise Server instance, navigate to the main page of the repository.
- 2 In the right sidebar of your repository, click **Packages**.



- 3 Search for and then click the name of the package that you want to manage.
- 4 On your package's landing page, on the right-hand side, click  **Package settings**.



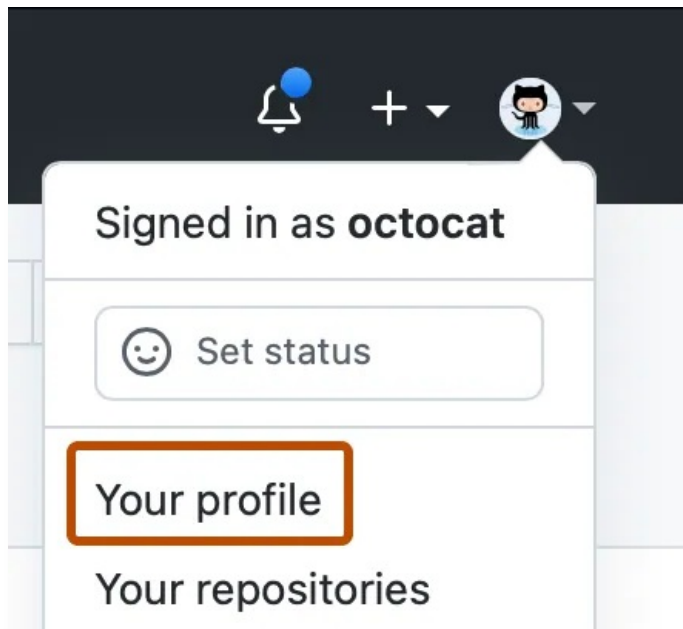
- 5 At the bottom of the page, under "Danger Zone", click **Delete this package**.
- 6 To confirm, review the confirmation message, enter your package name, and click **I understand, delete this package**.



Deleting an entire user-scoped package on GitHub [↗](#)

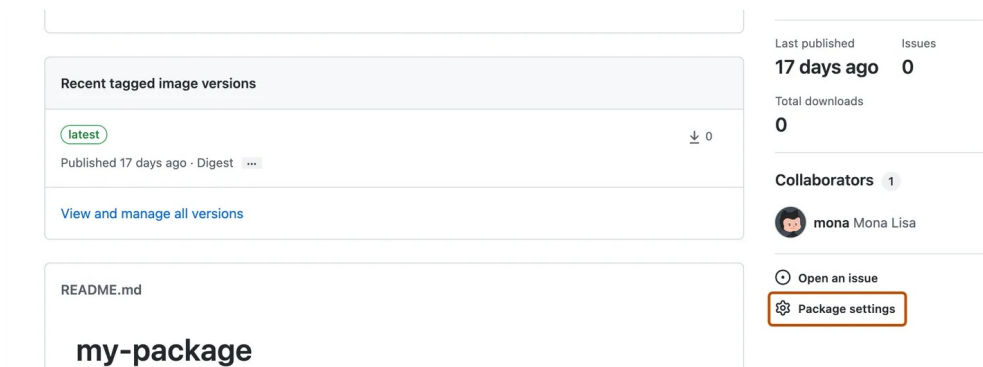
To review who can delete a package, see "[Required permissions](#)."

- 1 On GitHub, navigate to the main page of your personal account.
- 2 In the top right corner of GitHub Enterprise Server, click your profile photo, then

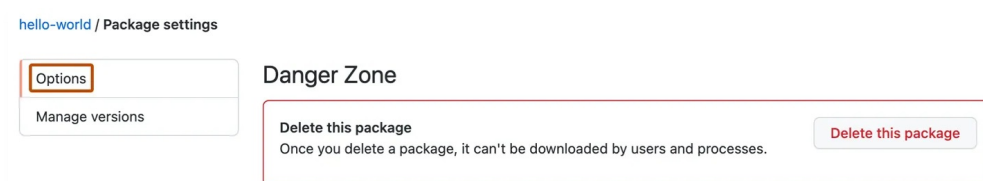
click **Your profile**.



- 3 On your profile page, in the header, click the  **Packages** tab.
- 4 Search for and then click the name of the package that you want to manage.
- 5 On your package's landing page, on the right-hand side, click  **Package settings**.




- 6 On the left click **Options**.

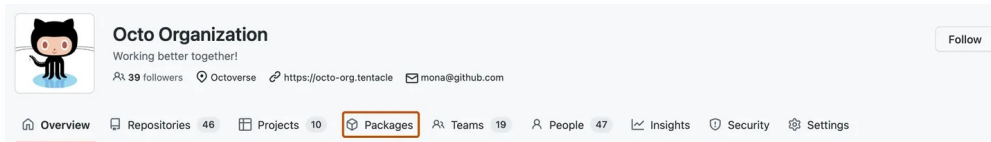



- 7 At the bottom of the page, under "Danger zone", click **Delete this package**.
- 8 In the confirmation box, type the name of the package to confirm you want to delete it.
- 9 Click **I understand the consequences, delete this package**.

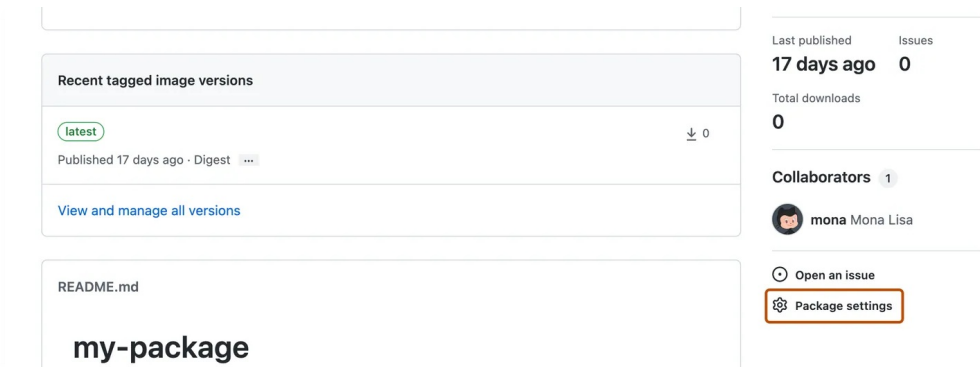
Deleting an entire organization-scoped package on GitHub

To review who can delete a package, see "[Required permissions](#)."

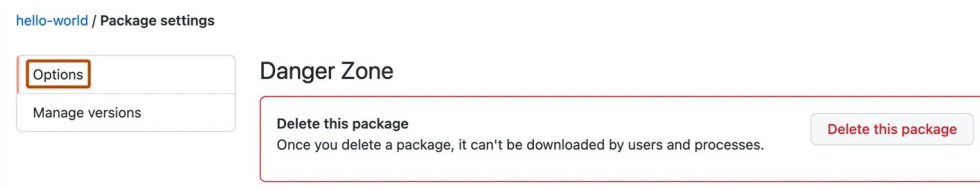
- 1 On GitHub, navigate to the main page of your organization.
- 2 Under your organization name, click the  **Packages** tab.



- 3 Search for and then click the name of the package that you want to manage.
- 4 On your package's landing page, on the right-hand side, click  **Package settings**.



- 5 On the left click **Options**.



- 6 At the bottom of the page, under "Danger zone", click **Delete this package**.
- 7 In the confirmation box, type the name of the package to confirm you want to delete it.
- 8 Click **I understand the consequences, delete this package**.

Restoring packages

You can restore a deleted package or version if:

- You restore the package within 30 days of its deletion.
- The same package namespace and version is still available and not reused for a new package.

For example, if you're the user `octocat`, and you have a deleted RubyGems package named `my-package` that was scoped to the repo `octocat/my-repo`, then you can only restore the package if the package namespace `rubygem.pkg.github.com/octocat/my-repo/my-package` is still available, and 30 days have not yet passed.

To delete a package, you must also have admin permissions to the repository in which the package is published.


For more information, see "[Required permissions](#)."

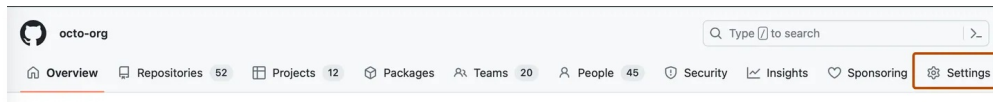
Once the package is restored, the package will use the same namespace it did before. If the same package namespace is not available, you will not be able to restore your package. In this scenario, to restore the deleted package, you must delete the new package that uses the deleted package's namespace first.

Restoring a package in an organization [↗](#)

You can restore a deleted package through your organization account settings, as long as the package was in a repository owned by the organization or had granular permissions and was scoped to your organization account.

To review who can restore a package in an organization, see "[Required permissions](#)."

- 1 On your GitHub Enterprise Server instance, navigate to the main page of the organization.
- 2 Under your organization name, click  **Settings**. If you cannot see the "Settings" tab, select the ... dropdown menu, then click **Settings**.

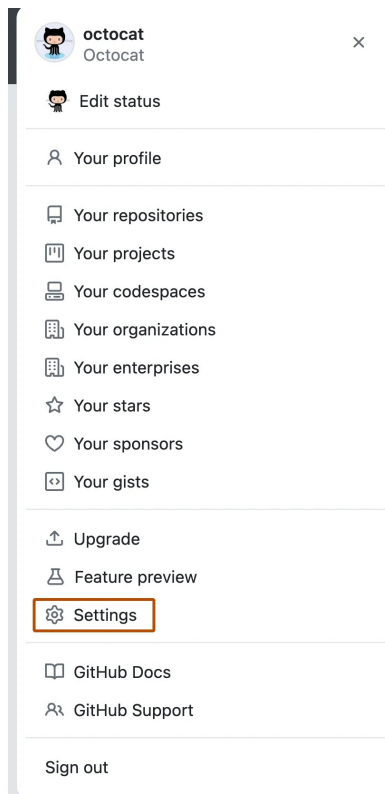


- 3 On the left, click **Packages**.
- 4 Under "Deleted Packages", next to the package you want to restore, click **Restore**.
- 5 To confirm, type the name of the package and click **I understand the consequences, restore this package**.

Restoring a user-account scoped package [↗](#)

You can restore a deleted package through your personal account settings, if the package was in one of your repositories or scoped to your personal account. For more information, see "[Required permissions](#)."


- 1 In the upper-right corner of any page, click your profile photo, then click **Settings**.

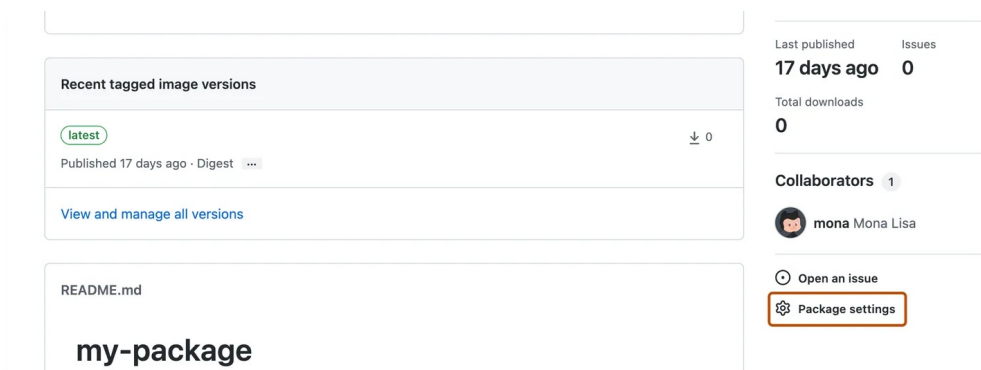


- 2 In the left sidebar, click **Packages**.
- 3 Under "Deleted Packages", next to the package you want to restore, click **Restore**.
- 4 To confirm, type the name of the package and click **I understand the consequences, restore this package**.

Restoring a package version [🔗](#)


You can restore a package version from your package's landing page. To review who can restore a package, see "[Required permissions](#)."

- 1 Navigate to your package's landing page.
- 2 Search for and then click the name of the package that you want to manage.
- 3 On your package's landing page, on the right-hand side, click  **Package settings**.



- 4 Navigate to where you can manage versions for your type of package.
 - If your package is a container, under the "Recent Versions" section, click **View and manage all versions**.

Recent Versions		
1.0.2	Latest Published almost 4 years ago	↓ 341
1.0.1	Published almost 4 years ago	↓ 75
1.0.0	Published almost 4 years ago	↓ 139
View and manage all versions		

- o For types of packages other than containers:
 - a. On the right-hand side, click  **Package settings**.

Recent tagged image versions

latest

Published 17 days ago · Digest ...

[View and manage all versions](#)

README.md


my-package


Last published17 days ago0


Issues

Total downloads0

Collaborators1

 mona Mona Lisa

 Open an issue

 Package settings

- b. On the left click **Manage versions**.

[hello-world-npm](#) / [Manage Versions](#)

Options

Manage versions

Manage versions

Versions: Active ▾

1.0.3 Latest version

⬇ 0 yesterday by Codertocat

Delete

1.0.2

⬇ 0 yesterday by Codertocat

Delete

- At the top right corner of the list of package versions, use the **Select versions view** dropdown and select **Deleted**.

- Next to the deleted package version you want to restore, click **Restore**.
- To confirm, click **I understand the consequences, restore this version**.

Legal

