



# Object versioning

Using the Swift API, you have the option to retain multiple versions of an object, rather than overwriting it in place.

## Prerequisites

In order to manage object versioning, be sure that you have **installed and configured** the `swift` command-line interface (CLI). There is presently no way to set object expiry with the `openstack` CLI.

Also, ensure that you have configured a **private container**, i.e. one with an empty Read ACL. The examples in this how-to guide assume that your container is named `private-container`, and that it presently contains a single object named `testobj.txt`.

## Creating a container for non-current versioned objects

In addition to the container holding your live objects, `private-container`, you will also need one for your prior versions. In this example, we use the name `private-container-versions` for that container:

```
swift post private-container-versions
```

This command produces no output.

## Setting `X-Versions-Location`

You must next set the `X-Versions-Location` header on the *original* container:

```
swift post -H "X-Versions-Location: private-container-versions" private-container
```

## Testing object versioning

You can now verify that object versioning is working correctly.

1. Upload a new version of `testobj.txt`:

```
$ echo "bye bye" > testobj.txt  
$ swift upload private-container testobj.txt
```

2. Observe that there is now a newly created object in the `private-container-versions` container:

```
$ swift list private-container-versions  
00btestobj.txt/1670250073.717985
```

3. Issue a command to delete the object:

```
$ swift delete private-container testobj.txt  
testobj.txt
```

4. Read back the object. Since the container is now versioned, the “deletion” in step 3 only caused a roll-back to the object’s prior state:

```
$ swift download -o - private-container testobj.txt  
hello world
```

Object versioning does not prevent deleting the *last remaining* (i.e. first created) version of an object. Do not use object versioning as a prevention mechanism for inadvertent object deletion; it is not suitable for that purpose.

Last update: 2022-12-21

Created: 2022-12-05

Authors: Florian Haas