

# Backports

This page describes the support in [Skara](#) for both creating and reviewing backports.

- [Terminology](#)
- [Commit Message](#)
- [Backport Pull Requests](#)
  - [Unknown or Missing Original Commit](#)
- [Backport Commit Command](#)
- [Workflow](#)
  - [Creating backports](#)
    - [Web UI](#)
    - [CLI](#)
  - [Reviewing backports](#)

## Terminology

A *backport commit* (often abbreviated to just *backport*) is a replica of an existing commit applied to a different repository. In most cases the backport commit will be applied to a repository representing an older release, for example creating a backport commit of a commit in the [jdk](#) repository for the [jdk11u-dev](#) repository.

A backport commit is considered *clean* if the changes in the original commit are identical to the changes in the backport commit. Note that only the changes has to be identical, not the changed lines.

## Commit Message

All backport commits features an additional commit message *trailer* (a line in the last paragraph of the commit message) - `Backport-of`. The `Backport-of` trailer describes which commit the backport commit was created from. The following commit message shows an example:

```
8123456: This is a bug

This commit fixes a tricky bug.

Reviewed-by: ehelin
Backport-of: 5a526c1c5716f6d9a7fc94741bcdb2f424d342df
```

In the example above it can be seen that the backport commit fixes the bug "JDK-8123456", has the summary "This commit fixes a tricky bug" and that the backport commit was reviewed by "ehelin". The `Backport-of` trailer shows that the backport commit is a replica of the commit with hash `5a526c1c5716f6d9a7fc94741bcdb2f424d342df`.

Note that author metadata recorded in the backport commit is the author of the backport commit, *not* the author of the original commit. Likewise the reviewers recorded in the commit message are the reviewers of the backport commit, *not* the reviewers of the original commit. If the author of the backport isn't [Author](#) in the project that the backport commit is created for, then the backport commit can be [sponsored](#). In the case of a backport being sponsored, then the author will recorded as the author of the commit and the sponsor as the committer of the commit (just as for regular commits).

## Backport Pull Requests

[Skara](#) features a way to declare a [pull request](#) to be a "backport pull request" in order to ease the creation of backport commits. A "backport pull request" is a pull request which title is of the form "Backport [0-9a-z]{40}", for example "Backport 5a526c1c5716f6d9a7fc94741bcdb2f424d342df". The hash represents the hash of the original commit that is being backported. When the Skara bots encounter a "backport pull request" then they will:

- Find the original commit with the given hash
- Mark the pull request as [solving](#) the same issues that was solved by the original commit
- Update the title to the first issue being solved
- Add the label "backport"
- Set the [summary](#) of the resulting commit of pull request to the summary of the original commit
- Append the correct `Backport-of` trailer to resulting commit when the pull request is integrated

Continuing with the example from the Commit Message section, a "backport pull request" with the title "Backport 5a526c1c5716f6d9a7fc94741bcdb2f424d342df" would result in bots marking the pull request as solving issue "JDK-8123456" and the final commit message to have the summary "This commit fixes a tricky bug". When the pull request is integrated the bots would also append "Backport-of: 5a526c1c5716f6d9a7fc94741bcdb2f424d342df" to the final commit message.

The bots will automatically detect if the commit in a backport pull request is a clean backport. If it is then bots will add the label "clean" to the pull requests. Backport pull requests which commit is a clean backport can be integrated *without* reviewers (configurable per repository). This means that not all commit messages for backport commits will have a "Reviewed-by" line.

The [Skara CLI tools](#) features the tool `git pr create` for creating pull requests. The option `--backport` to `git pr create` makes the title of pull request consist of "Backport" and the given hash, for example `git pr create --backport=5a526c1c5716f6d9a7fc94741bcdb2f424d342df` creates a pull request with the title "Backport 5a526c1c5716f6d9a7fc94741bcdb2f424d342df". The value of `--backport` does not have to be a full hash, it can be an abbreviated hash or a reference.

## Unknown or Missing Original Commit

In certain cases, the original commit for a fix is not publicly available, but a "backport pull request" still needs to be created. For this case, there is an alternate format for the pull request title: "Backport <issueld>" (e.g. "Backport JDK-1234567"). This will initiate the pull request as a "backport pull request", but without any reference to the original commit. This means that the commit message will also not contain the `Backport-of` trailer. The only real advantage of declaring a pull request in this way, compared to a normal pull request, is to avoid getting a warning about the issue not being open, and of course to clearly communicate to reviewers the intent of the change being a backport. Omitting the backport declaration for this kind of pull request will not affect the format of the commit message, nor how a backport issue is created in JBS.

## Backport Commit Command

The [/backport commit command](#) can be used to quickly create a [backport pull request](#) for a given commit. Just navigate to the commit in source code hosting provider's web UI and add a comment consisting of `/backport <repo>`. If the commit does not apply clean on the target repository then a message will be shown for the files with conflicts.

## Workflow

### Creating backports

#### Web UI

Navigate to the commit intended to be backported in the source code hosting provider's web UI and issue the [/backport commit command](#). If the commit applied cleanly onto the target repository then go to the pull request linked in the reply and issue the [/sponsor](#) pull request command.

If the commit did not apply cleanly then the commit must be backported manually and backport pull request must be created manually. See the [CLI](#) section for an example of how to do this using the command-line.

#### CLI

Use the [Skara CLI tool `git-backport`](#) to try to automatically create a backport pull request for the given commit, for example `git backport https://github.com/openjdk/jdk_5a526c1c5716f6d9a7fc94741bcdb2f424d342df`. If the commit applied cleanly onto the target repository then sponsor the resulting backport pull request with [git-pr sponsor](#), for example `git pr sponsor 17`.

If the commit could not be applied cleanly, then the conflicts must be manually resolved. After the conflicts have been resolved then a pull request must be created with the title "Backport <hash>". An example of how to do this is shown below:

```
$ git checkout -b backport-5a526c1c
$ git fetch https://github.com/openjdk/jdk_5a526c1c5716f6d9a7fc94741bcdb2f424d342df
$ git cherry-pick --no-commit FETCH_HEAD
$ # resolve conflicts
$ git commit -m 'Backport 5a526c1c5716f6d9a7fc94741bcdb2f424d342df'
$ git push -u origin backport-5a526c1c
```

The output from the final `git push` will return a link that can be used to create the pull request.

### Reviewing backports