Automatically Linked References in Issues, Pull Requests, and Commit Messages

- User, Team, and Organization Mentions
- Commits
- Code References
- Issues and Pull Requests
- Actionable References in Pull Requests and Commit Messages
- Time Tracking in Pull Requests and Commit Messages
- External Trackers
- Issues and Pull Requests References Summary

When an issue, pull request, or comment is posted, the text description is parsed in search of references. These references will be shown as links in the Issue View and, in some cases, produce certain *actions*.

Likewise, commit messages are parsed when they are listed, and *actions* can be triggered when they are pushed.

To prevent the creation of unintended references, there are certain rules for them to be recognized. For example, they should not be included inside code text. They should also be separated from the surrounding text with spaces.

User, Team, and Organization Mentions

When text in the form `@username` is found and `username` matches the name of an existing user, a *mention* reference is created. This will be shown by changing the text into a link to said user's profile, and possibly creating a notification for the mentioned user depending on whether they have the necessary permission to access the contents.

Example:

> “ @John, can you give this a look?

This is also valid for teams and organizations:

> “ @Documenters, we need to plan for this. @CoolCompanyInc, this issue concerns us all!

Teams will receive email notifications when appropriate, but whole organizations won't.

Commit messages do not produce user notifications.

Commits

Commits can be referenced using their SHA1 hash, or a portion of it of at least seven characters. They will be shown as a link to the corresponding commit.

Example:

> “ This bug was introduced in e59ff077

Code References

It is possible to reference any portion of a file in a repository by using the permalink feature inside the file viewer:

Permalinking of a line range

If used inside an issue or pull request, this is automatically rendered into an inline preview of the portion of that file.

> Alternatively, you can also use the "Reference in a new issue" action to automatically create an issue with the correct permalink.

Example:

> See the code here: https://next.forgejo.org/Mai-Lapyst/test/src/commit/66719555bd242b9ad7c3fb9eef2b388826d87872/hook.go#L191-L200 It sets all of the options for this.

Rendered result of a permalink code reference

There are some restrictions in place:

- Code linked in private repositories will not be displayed if the current viewer has insufficient permissions.
- It can only ever be a permalink; linking to files by a branch or tag will not result in a preview.
- There is a maximum amount of lines that will be rendered. By default, this is 50.
- Only permalinks to the current instance will work.

Issues and Pull Requests

A reference to another issue or pull request can be created using the simple notation `#1234`, where *1234* is the number of an issue or pull request in the same repository. These references will be shown as links to the referenced content.

The effect of creating this type of reference is that a *notice* will be created in the referenced document, provided the creator of the reference has reading permissions on it.

Example:

> This seems related to #1234

Issues and pull requests in other repositories can be referred to as well using the form `owner/repository#1234`:

> This seems related to mike/compiler#1234

Alternatively, the `!1234` notation can be used as well. Even when in Forgejo a pull request is a form of issue, the `#1234` form will always link to an issue; if the linked entry happens to be a pull request instead, Forgejo will redirect as appropriate. With the `!1234` notation, a pull request link will be created, which will be redirected to an issue if required. However, this distinction could be important if an external tracker is used, where links to issues and pull requests are not interchangeable.

Actionable References in Pull Requests and Commit Messages

Sometimes a commit or pull request may fix or bring back a problem documented in a particular issue. Forgejo supports closing and reopening the referenced issues by preceding the reference with a particular *keyword*. Common keywords include "closes", "fixes", "reopens", etc. This list can be customized by the site administrator.

Example:

> This PR *closes* #1234

If the actionable reference is accepted, this will create a notice on the referenced issue announcing that it will be closed when the referencing PR is merged.

For an actionable reference to be accepted, *at least one* of the following conditions must be met:

- The commenter has permissions to close or reopen the issue at the moment of creating the reference.
- The reference is inside a commit message.
- The reference is posted as part of the pull request description.

In the last case, the issue will be closed or reopened only if the merger of the pull request has permissions to do so.

Additionally, only pull requests and commit messages can create an action, and only issues can be closed or reopened this way.

The default *keywords* are:

- **Closing**: close, closes, closed, fix, fixes, fixed, resolve, resolves, resolved
- **Reopening**: reopen, reopens, reopened

Time Tracking in Pull Requests and Commit Messages

When a commit or merging of a pull request results in automatic closing of an issue, it is possible to also add the spent time resolving this issue through the commit message.

To specify spent time on resolving an issue, you need to specify time in the format `@<number><time-unit>` after the issue number. In one commit message, you can specify multiple fixed issues and spent time for each of them.

Supported time units (`<time-unit>`):

- `m` - minutes
- `h` - hours
- `d` - days (equals to 8 hours)
- `w` - weeks (equals to 5 days)
- `mo` - months (equals to 4 weeks)

Numbers to specify time (`<number>`) can be also decimal numbers, e.g., `@1.5h` would result in one and a half hours. Multiple time units can be combined, e.g., `@1h10m` would mean 1 hour and 10 minutes.

Example of commit message:

> Fixed #123 spent @1h, refs #102, fixes #124 @1.5h

This would result in 1 hour added to issue #123 and 1 and a half hours added to issue #124.

External Trackers

Forgejo supports the use of external issue trackers, and references to issues hosted externally can be created in pull requests. However, if the external tracker uses numbers to identify issues, they will be indistinguishable from the pull requests hosted in Forgejo. To address this, Forgejo allows the use of the `!` marker to identify pull requests. For example:

> This is issue #1234, and links to the external tracker. This is pull request !1234, and links to a pull request in Forgejo.

The `!` and `#` can be used interchangeably for issues and pull requests *except* for the case, where a distinction is required. If the repository uses an external tracker, the commit message for a squash merge will use `!` as a reference marker by default.

Issues and Pull Requests References Summary

This table illustrates the different kinds of cross-reference for issues and pull requests. In the examples, `User1/Repo1` refers to the repository where the reference is used, while `UserZ/RepoZ` indicates a different repository.

Reference in User1/Repo1	Repo1 issues are external	RepoZ issues are external	Should render
`#1234`	no	-	A link to issue/pull 1234 in `User1/Repo1`
`!1234`	no	-	A link to issue/pull 1234 in `User1/Repo1`
`#1234`	yes	-	A link to *external issue* 1234 for `User1/Repo1`
`!1234`	yes	-	A link to *PR* 1234 for `User1/Repo1`
`User1/Repo1#1234`	no	-	A link to issue/pull 1234 in `User1/Repo1`
`User1/Repo1!1234`	no	-	A link to issue/pull 1234 in `User1/Repo1`
`User1/Repo1#1234`	yes	-	A link to *external issue* 1234 for `User1/Repo1`

Reference in User1/Repo1	Repo1 issues are external	RepoZ issues are external	Should render
`User1/Repo1!1234`	yes	-	A link to *PR* 1234 for `User1/Repo1`
`UserZ/RepoZ#1234`	-	no	A link to issue/pull 1234 in `UserZ/RepoZ`
`UserZ/RepoZ!1234`	-	no	A link to issue/pull 1234 in `UserZ/RepoZ`
`UserZ/RepoZ#1234`	-	yes	A link to *external issue* 1234 for `UserZ/RepoZ`
`UserZ/RepoZ!1234`	-	yes	A link to *PR* 1234 for `UserZ/RepoZ`
Alphanumeric issue IDs:	-	-	-
`AAA-1234`	yes	-	A link to *external issue* `AAA-1234` for `User1/Repo1`
`!1234`	yes	-	A link to *PR* 1234 for `User1/Repo1`
`User1/Repo1!1234`	yes	-	A link to *PR* 1234 for `User1/Repo1`
Not supported	-	yes	A link to *external issue* `AAA-1234` for `UserZ/RepoZ`
`UserZ/RepoZ!1234`	-	yes	A link to *PR* 1234 in `UserZ/RepoZ`

The last section is for repositories with external issue trackers that use an alphanumeric format.

***-**: not applicable.*

Note: automatic references between repositories with different types of issues (external vs. internal) are not fully supported and may render invalid links.

□□□ #2
□ □□□ □□□ 2026-10-31 13:36:47 UTC
□ □□□ □□□ 2026-10-31 13:36:48 UTC