

Create issue link

This post function automatically **creates** one or multiple **issue links**.

You can create links between virtually **any issue**.



Configuration

Issue link type

Select the issue link type to be created **between** the **source issue** and the **destination issue**. All available link types will be displayed.

Source issues

Select the **source issues** to create the issue links **from**. The following options are available:

Option	Description
Current issue	The link will be created from the current issue.
Issue keys in field	When you select this option, specify the field containing the key(s) of the issues that should be linked from. The field must contain either a single issue key or a list of issue keys , separated by comma or blank.
Set issues manually (parser expression)	When you select this option, you can specify the issues to be linked through an expression in JQL mode .

Destination issues

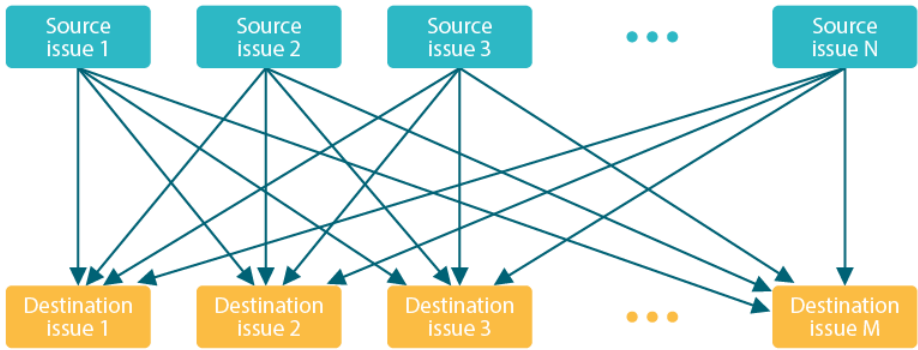
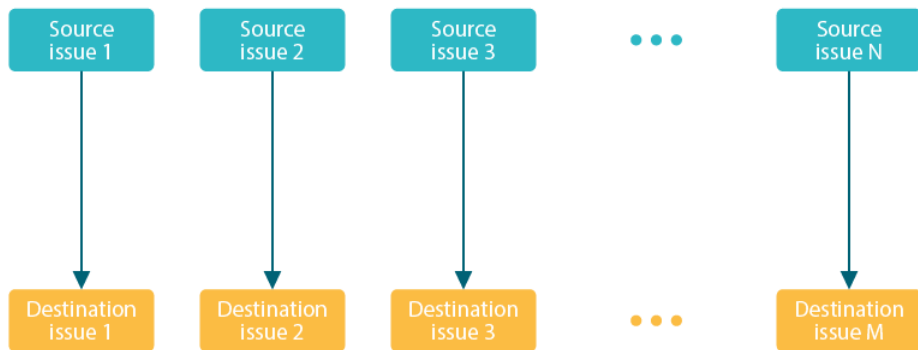
Select the **destination issues** to create the issue links **to**. The following options are available:

Option	Description
Issue keys in field	When you select this option, specify the field containing the key(s) of the issues that should be linked to. The field must contain either a single issue key or a list of issue keys , separated by comma or blank.
Set issues manually (parser expression)	When you select this option, you can specify the issues to be linked through an expression in JQL mode .

Additional options

You can select how the issue links should be created in case there are multiple issues to be linked. The following options are available:

Option	Description
--------	-------------

Many-to-many relationship	<p>Each source issue will be linked to all destination issues.</p>  <p>The diagram shows four teal boxes at the top labeled 'Source issue 1', 'Source issue 2', 'Source issue 3', and 'Source issue N' (with three dots between 3 and N). Below them are four orange boxes labeled 'Destination issue 1', 'Destination issue 2', 'Destination issue 3', and 'Destination issue M' (with three dots between 3 and M). Every teal box has a blue arrow pointing to every orange box, representing a complete bipartite graph where each source issue is linked to all destination issues.</p>
One-to-one relationship	<p>Each source issue will be linked to one destination issue according to order in which they are returned by the JQL query.</p>  <p>The diagram shows four teal boxes at the top labeled 'Source issue 1', 'Source issue 2', 'Source issue 3', and 'Source issue N' (with three dots between 3 and N). Below them are four orange boxes labeled 'Destination issue 1', 'Destination issue 2', 'Destination issue 3', and 'Destination issue M' (with three dots between 3 and M). Each teal box has a single blue arrow pointing down to the corresponding orange box (1 to 1, 2 to 2, 3 to 3, and N to M), representing a one-to-one mapping.</p>

Conditional execution

You can **optionally** specify a [logical expression](#) to define the circumstances (or conditions) under which the post function should be executed.

The result of the logical expression must return a boolean value of either:

- **true** the post function will be executed
- **false** the post function will **not** be executed

Using the **conditional operator**, even complex or multi-layered conditions can be constructed.

Make sure to learn more about defining logical expressions and browse through the various **examples** here: [Logical mode](#)

Run as


Select which **user** will be used to execute this post function. By default this parameter is set to the **current user**. You can also use field codes to run the function as a dynamic user (e.g. current assignee).

Make sure that the user running the post function has all the **relevant permissions** to perform the actions defined in the configuration (e.g. "Update Issues")!

If you want to keep track the actions being performed automatically, we suggest to create a **dedicated JWT account**, granted all relevant **permissions**, and use it in the Run as parameter to identify which changes have been made with JWT.



Use cases and examples

Use case	JWT feature	Workflow function	Parser functions	Label
Link issue to issue keys in its description		Create issue link	findPattern() toString()	

If you still have questions, feel free to refer to our [support](#) team.