

Configure the Connection in Script Mode for Jira Cloud

Last Modified on 01/15/2026 7:12 pm EST

Configure the Synchronization (Optional)

Set up Sync Rules

Once you test the Connection, you can choose to configure the synchronization behavior with the help of Sync Rules (Groovy-based scripts).

1. Edit your Connection.

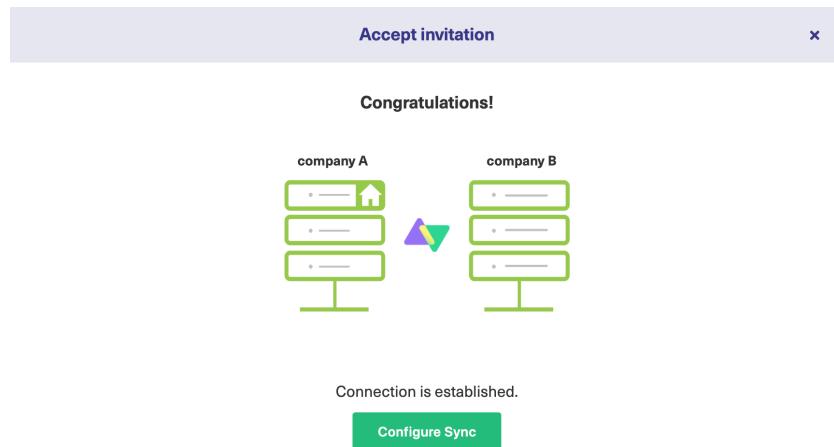
There are multiple ways to edit your connection:

- Navigate to **Connections** →  **Edit connection**.



| Connection | Issues under sync | Last sync | Status |
|--|-------------------|----------------------------------|--|
| Company A to Company B Description of the connection lorem ipsum sit amet | 123 | Issue TEST-777 46 minutes ago | Active    |

- If you just created a connection, select **Configure sync**.



Accept invitation x

Congratulations!

company A company B

Connection is established.

Configure Sync

2. Set up **Outgoing sync** and **Incoming sync** in the **Rules** tab.

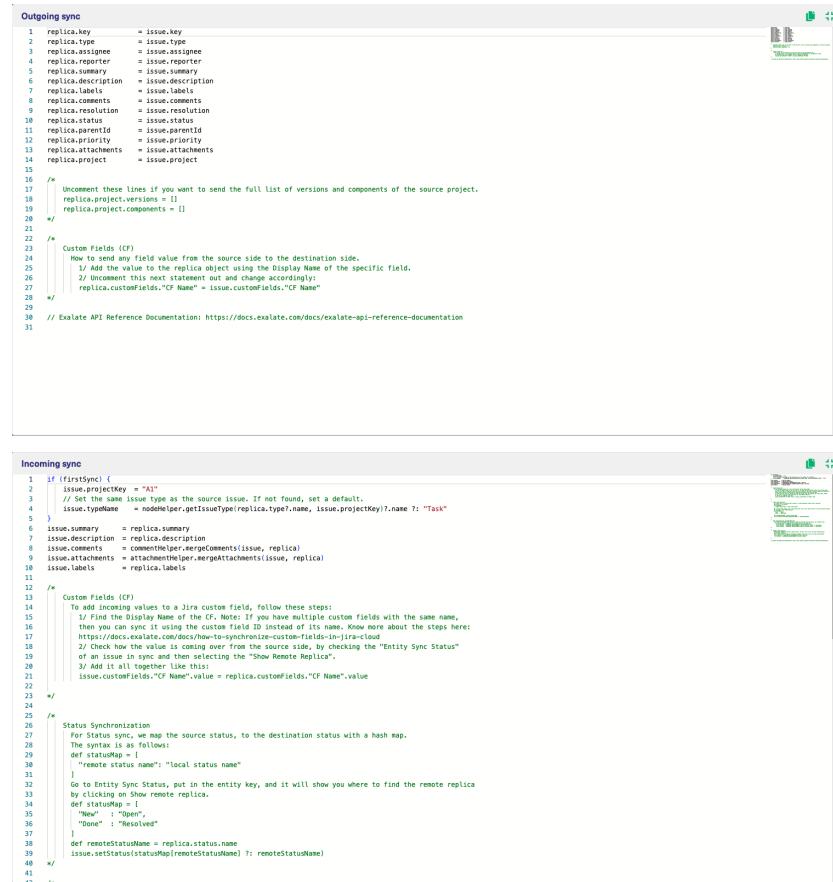
Note: Outgoing sync and Incoming sync scripts:

- are generated by default once the connection has been set up. You can, however, add, edit or remove these scripts to configure your sync according to your requirements.
- are platform specific.
- are present at both the source and the destination instance to give independent control

over what data needs to be sent or received.

In the **Outgoing sync**, you can enter scripts to specify what data you send.

In the **Incoming sync**, you can enter scripts to specify what data you receive.



```
Outgoing sync
1 replica.key      = issue.key
2 replica.type     = issue.type
3 replica.assignee = issue.assignee
4 replica.reporter = issue.reporter
5 replica.summary  = issue.summary
6 replica.description = issue.description
7 replica.labels   = issue.labels
8 replica.comments = issue.comments
9 replica.parentId = issue.parentId
10 replica.status   = issue.status
11 replica.parentId = issue.parentId
12 replica.id      = issue.id
13 replica.attachments = issue.attachments
14 replica.project  = issue.project
15
16 /* Uncomment these lines if you want to send the full list of versions and components of the source project.
17 replica.project.versions = []
18 replica.project.components = []
19 */
20
21 /*
22  * Custom Fields (CF)
23  * How to send any field value from the source side to the destination side.
24  * 1/ Add the value to the replica object using the field name of the specific field.
25  * 2/ Remove this next statement out and change accordingly:
26  *    replica.customFields."CF Name" = issue.customFields."CF Name"
27 */
28
29
30 // Exalate API Reference Documentation: https://docs.exalate.com/docs/exalate-api-reference-documentation
31

Incoming sync
1 if (issue.type) {
2   issue.projectKey = "A1"
3   // Set the same issue type as the source issue. If not found, set a default.
4   issue.typeName = nodeHelper.getIssueType(replica.type.name, issue.projectKey)?."name" ?: "Task"
5
6   issue.summary = replica.summary
7   issue.description = replica.description
8   issue.comments = commentHelper.mergeComments(issue, replica)
9   issue.attachments = attachmentHelper.mergeAttachments(issue, replica)
10  issue.labels = replica.labels
11
12 /*
13  * Custom Fields (CF)
14  * To add incoming values to a Jira custom field, follow these steps:
15  * 1/ Go to Jira and create the CF Name. If you want to sync the destination fields with the same name,
16  * then you can sync it using the custom field ID instead of its name. Know more about the steps here:
17  * https://docs.exalate.com/docs/how-to-synchronize-custom-fields-in-jira-cloud
18  * 2/ Check if the value is coming over from the source side, by checking the "Entity Sync Status"
19  * of the issue type and selecting the "Show Remote Replica".
20  * 3/ Add it all together like this:
21  * issue.customFields."CF Name".value = replica.customFields."CF Name".value
22 */
23
24 /*
25  * Status Synchronization
26  * For Status sync, we map the source status, to the destination status with a hash map.
27  * The syntax is as follows:
28  * def statusMap = {
29  *   "remote status name": "local status name"
30  * }
31  * Go to Entity Sync Status, put in the entity key, and it will show you where to find the remote replicas
32  * by clicking on Show remote replica.
33  * def statusMap = {
34  *   "New" : "Open",
35  *   "Done" : "Resolved"
36  * }
37  * def remoteStatusName = replica.status.name
38  * issue.setStatusStatusMap[remoteStatusName] ?: remoteStatusName
39 */
40
41 ..
```

Automate your Synchronization with Triggers

Specify **Triggers** if necessary.

With triggers, you can set up automatic sync of entities that fit a specific search query.

You can create **Triggers** by editing the connection or by configuring the sync as shown in the previous step. You can also create them on the Exalate Admin Console under the **Triggers** tab.

Note: For more information, check out [How to create a trigger](#).

Publish the changes made to the Connection

Click **Publish** to save a connection.

You can also use these hotkeys to publish a connection:

- **Ctrl + S** on Windows or Linux

- **Cmd + S** on Mac

Note: Changes in existing connections are not applied before you update the entity. Please make sure to update the entity if you want to apply the changes.

What's Next?

Options you can consider next:

- Sit back and relax! Sync happens based on the Sync Rules and Triggers you have set up in this step.
- You can also read the [Configuration Guides](#) to see more synchronization examples.
- You can also read the [Exalate API Reference Documentation](#) to learn more about how to work with incoming and outgoing sync scripts (aka Sync Rules).

ON THIS PAGE

[About Us](#) ?

[Configure the Synchronization](#) (Optional)

[Glossary](#) ?

[Publish the changes made to the Connection](#)

[API Reference](#) ?

[What's Next?](#)

[Pricing and Licensing](#) ?

Resources

[Subscribe for a weekly Exalate hack](#) ?

[Academy](#) ?

[Blog](#) ?

[YouTube Channel](#) ?

[Ebooks](#) ?

Still need help?

[Join our Community](#) ?

[Visit our Service Desk](#) ?

[Find a Partner](#) ?