Configure the Connection in Script Mode for Jira Cloud

Last Modified on 03/18/2024 10:30 am EDT

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**.

- If you just created a connection, select **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.resolution   = issue.resolution
10  replica.status       = issue.status
11  replica.parentId     = issue.parentId
12  replica.priority     = issue.priority
13  replica.attachments  = issue.attachments
14  replica.project      = issue.project
15
16  /*
17     Uncomment these lines if you want to send the full list of versions and components of the source project.
18     replica.project.versions = []
19     replica.project.components = []
20  */
21
22  /*
23     Custom Fields (CF)
24        How to send any field value from the source side to the destination side.
25           1/ Add the value to the replica object using the Display Name of the specific field.
26           2/ Uncomment this next statement out and change accordingly:
27           replica.customFields."CF Name" = issue.customFields."CF Name"
28  */
29
30     // Exalate API Reference Documentation: https://docs.exalate.com/docs/exalate-api-reference-documentation
31
```

```
Incoming sync
1   if (firstSync) {
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/ Find the Display Name of the CF. Note: If you have multiple custom 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 how the value is coming over from the source side, by checking the "Entity Sync Status"
19           of an issue in sync and then 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  /*
26     Status Synchronization
27        For Status sync, we map the source status, to the destination status with a hash map.
28        The syntax is as follows:
29        def statusMap = [
30           "remote status name": "local status name"
31        ]
32        Go to Entity Sync Status, put in the entity key, and it will show you where to find the remote replica
33        by clicking on Show remote replica.
34        def statusMap = [
35           "New"   : "Open",
36           "Done"  : "Resolved"
37        ]
38        def remoteStatusName = replica.status.name
39        issue.setStatus(statusMap[remoteStatusName] ?: remoteStatusName)
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).