

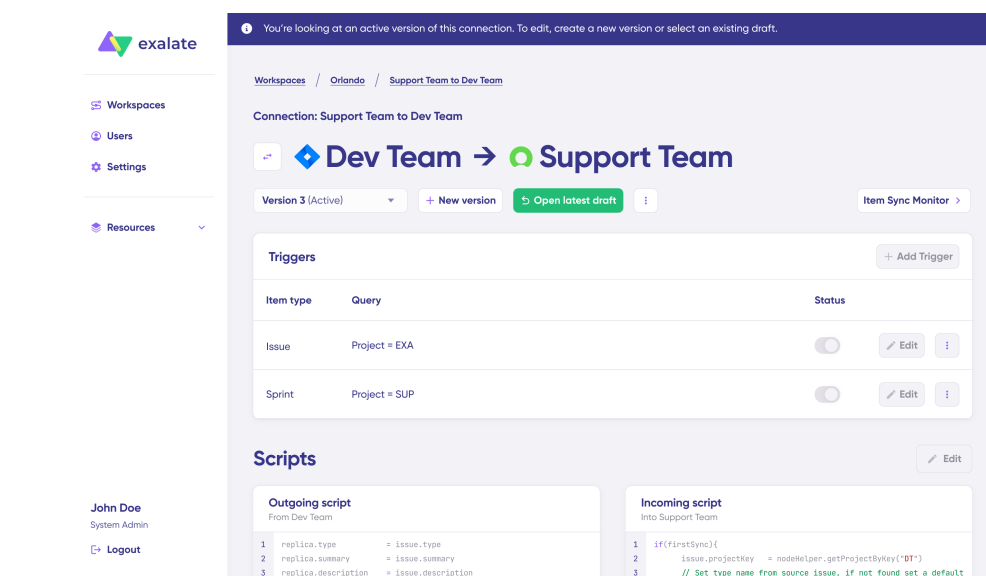
Using Aida AI for Script Configuration

Last Modified on 01/15/2026 9:31 am EST

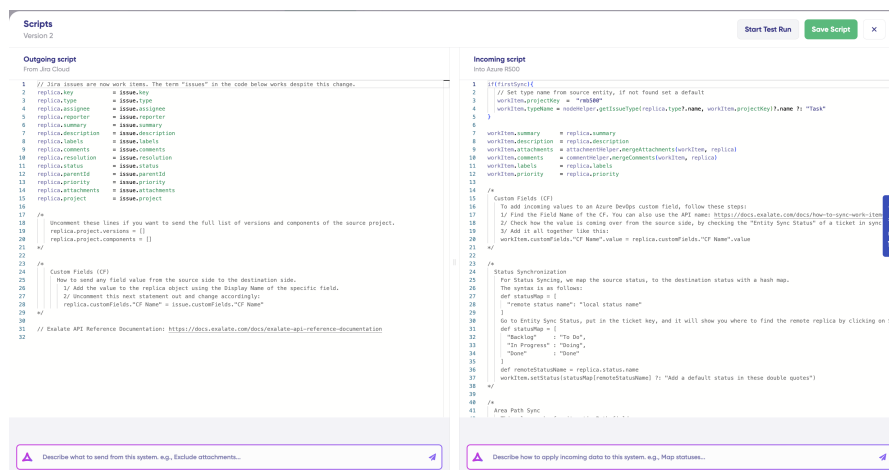
Aida is Exalate's AI-powered integration assistant embedded into the Exalate script editor to help you generate and modify synchronization scripts through natural language prompts. Instead of writing Groovy code manually, you can describe what you want to achieve, and Aida will draft the code for you.

Accessing Aida

1. Navigate to your connection
2. Click **Add new version** (or open the latest draft version)
3. Click **Edit** to enter the script editor



You'll see Aida chat interfaces at the bottom of both the **Outgoing script** and **Incoming script** sections. Each Aida chat works independently, allowing you to configure both sides of your connection simultaneously.



How Aida Works

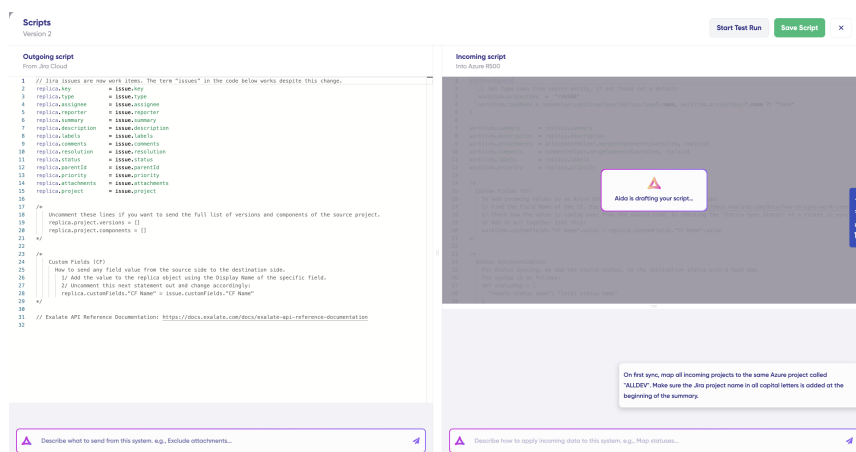
Aida helps you in two ways:

- **For Outgoing scripts:** Describe what data should leave your system. For example, "Exclude attachments" or "Only sync high-priority issues."
- **For Incoming scripts:** Describe how incoming data should be applied to your system. For example, "Map statuses" or "Set a default assignee if the user can't be found."

Using Aida

Submitting a Prompt

1. In the Aida input field, type your request in plain language
2. Click the send button (or press Enter)
3. Aida will begin drafting your script



Example prompts:

- "Exclude attachments from sync."
- "Map priority field to remote system."
- "On first sync, map all incoming projects to the same Azure project called 'ALLDEV'"
- "Set a Reporter/Assignee from the source side, if the user can't be found, set a default use.r"

Reviewing Aida's Changes

Once Aida finishes, you'll see:

- **Green highlighting:** New lines that will be added
- **Red highlighting:** Lines that will be removed
- A floating panel with **Insert** and **Discard** buttons

- Aida's explanation of the changes in the chat

The script editor becomes read-only during review, so you must choose to either insert or discard the changes before making manual edits.

Scripts
Version 2

Outgoing script
From Jira Cloud

```
1 // Jira issues are now work items. The term "issues" in the code below works despite this change.
2 replica.key = issue.key
3 replica.type = issue.type
4 replica.assignee = issue.assignee
5 replica.reporter = issue.reporter
6 replica.summary = issue.summary
7 replica.description = issue.description
8 replica.labels = issue.labels
9 replica.comments = issue.comments
10 replica.resolution = issue.resolution
11 replica.status = issue.status
12 replica.parentId = issue.parentId
13 replica.priority = issue.priority
14 replica.attachments = issue.attachments
15 replica.project = issue.project
16
17 /*
18 Uncomment these lines if you want to send the full list of versions and components of the source project.
19 replica.project.versions = []
20 replica.project.components = []
21 */
22
23 /*
24 Custom Fields (CF)
25 How to send any field value from the source side to the destination side.
26 1/ Add the value to the replica object using the Display Name of the specific field.
27 2/ Uncomment this next statement out and change accordingly:
28 replica.customFields."CF Name" = issue.customFields."CF Name"
29 */
30
31 // Exalate API Reference Documentation: https://docs.exalate.com/docs/exalate-api-reference-documentation
32
```

Incoming script
Into Azure R500

```
1 1 if(!firstSync){
2 // Set type name from source entity, if not found set a default
3 workItem.projectKey = "rb580"
4 // Set all incoming issues to the "ALLDEV" Azure project
5 workItem.projectKey = "ALLDEV"
6
7 // Set type name from source entity, if not found set a default
8 workItem.typeName = modelHelper.getIssueType(replica.type.name, workItem.projectKey).name ? "Task"
9
10 // Prepend the Jira project name (in all caps) to the summary
11 def jiraProjectName = (replica.project.name ? "").toLowerCase()
12 workItem.summary = jiraProjectName ? (jiraProjectName + " " + replica.summary) : replica.summary
13 } else {
14 workItem.summary = replica.summary
15 }
16
17
18 workItem.summary = replica.summary
19
20 workItem.description = replica.description
21
22 workItem.attachments = attachmentHelper.mergeAttachments(workItem, replica)
23
24 workItem.comments = commentHelper.mergeComments(workItem, replica)
25
26 workItem.labels = replica.labels
27
28 workItem.priority = replica.priority
29
30
31
32 Custom Fields (CF)
33 To add incoming values to an Azure DevOps custom field, follow these steps:
34 1/ Find the Field Name of the CF. You can also use the API name: https://docs.exalate.com/docs/how-to-sync-work-its
35 2/ Check how the value is coming over from the source side, by checking the "Entity Sync Status" of a ticket in sys
36 3/ Add it all together like this:
37 workItem.customFields."CF Name".value = replica.customFields."CF Name".value
38
39
40
41 Status Synchronization
```

Feedback

Here's how your requirements have been implemented:
On first sync, all incoming issues are mapped to the Azure DevOps project "ALLDEV".
On first sync, the Jira project name (in all capital letters) is prepended to the summary field.
Would you like to review or further adjust this configuration?

Describe what to send from this system. e.g., Exclude attachments...

Describe how to apply incoming data to this system. e.g., Map statuses...

Working with Scripts on Both Sides (Incoming and Outgoing)

The Outgoing and Incoming Aida chats work independently, so you can:

- Submit a prompt in the Outgoing script while Aida is generating code for the Incoming script
- Maintain separate conversation contexts for each script direction

Adjusting Your Workspace

You can resize the script panels and Aida chat windows to focus on what matters most:

- Drag the handle at the top of the Aida chat to increase or decrease its height
- Drag the divider between Outgoing and Incoming scripts to adjust their widths

Important Notes

- Aida is a helpful assistant, but you should always review the generated code before applying it
- Test your configuration using **Start Test Run** before publishing changes to production
- Aida works best when you provide clear, detailed descriptions of your sync requirements
- You can always manually edit the scripts after applying Aida's suggestions

Product

[Accessing Aida](#)

[About Us](#)

[Using Aida](#)

[Release History](#)

[Glossary](#)

[Adjusting Your Workspace](#)

[API Reference](#)

[Important Notes](#)

[Security](#)

[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](#)