How to Sync Custom Fields in Jira On-premise

Last Modified on 02/24/2026 10:50 am EST

> **This article applies to Exalate Classic only.** If you're using the New Exalate experience, please refer to the New Exalate documentation.

This article describes how to synchronize custom fields in Jira.

Basic Custom Field Synchronization

You can synchronize all standard custom fields with the following custom field types:

- Number
- Text Field (single line/multi-line)
- User Picker
- Version Picker
- Date Picker
- Checkboxes
- Select List (single choice)
- Select List (multiple choices)
- Cascading Select List

There are always two main configuration points: ***Outgoing sync***, which is the sending side, and ***Incoming sync,*** or the receiving side. To send the custom field, you need to get the custom field object and assign it to the replica. To get the custom field object you can use the original *custom field name or ID*.

Get the custom field object examples.

> **Note**: Add the local custom field object to the replica *using the custom field name*
>
> ```
> replica.customFields."Custom field name" = issue.customFields."Local custom field name"
> ```
>
> Add the local custom field object to the replica ***using the custom field ID***
>
> ```
> replica.customFields."17800" = issue.customFields."17800"
> ```

To use the custom field data received from the remote side, you need to assign its value to a local field. This local field can be either a custom field or a standard field. The example below shows how to synchronize simple custom fields:

- single text
- multi-line

- date picker
- lists
- labels and
- others.

Outgoing Sync

Send the custom field to the remote side:

```
replica.customFields."Custom field name" = issue.customFields."Custom field name"
```

Incoming Sync

Set the value of the received custom field in the local custom field using the code below:

```
issue.customFields."Local custom field name".value = replica.customFields."Custom field name".value
```

How to Sync Custom Fields using Shortcuts

Some custom fields like Text or List fields can also be synced using shortcuts. The shortcuts exclude the 'CustomField' word and include the actual custom field name in quotes. This is an example of how to sync the Text Custom field (some_text) with another Text Custom field (my_text):

Text Custom Field: Outgoing Sync

```
replica."some_text" = issue."some_text"
```

Text Custom Field: Incoming Sync

```
issue."my_text" = replica."some_text"
```

This is an example of how to sync the List Custom field (some_list) with another List Custom field (my_list).

List Custom Field: Outgoing Sync

```
replica."some_list" = issue."some_list"
```

List Custom Field: Incoming Sync

```
issue."my_list" = replica."some_list".value
```

How to Set Default Values in a Custom Field

You can add a value to a custom field directly in the Incoming Sync Rules. For example, when the receiving side custom field does not have the relevant option, you can set a default option in the script. To set the value you need to know the custom field value type. It depends on the custom field type itself. Each value type has different properties and fields which you can use during the

synchronization. The table below includes available value types for every custom field type.

JIRA Custom Field Types	Value on script
Checkboxes	List of Option
Date Picker	Date
Date Time Picker	Date
Labels	List of Label
Number Field	Number
Radio Buttons	Option
Select List (cascading)	Cascading
Select List (multiple choices)	List of Option
Select List (single choice)	Option
Text Field (single line)	String
Text Field (multi-line)	String
Url Field	String
User Picker (single user)	User
User Picker (multiple users)	List of Users
Version Picker (multiple versions)	List of Versions

Below you can find examples of how to set a default value for different custom field types.

Text Custom Field

Set a default value to the text custom field "Address" if no custom field was sent from the other side.

```
issue.customFields."Remote Address".value = replica.customFields."Address"?.value ?:
      "1600 Amphitheatre Parkway Mountain View, CA 94043"
```

User Picker(Single/Multi User)

Set a default user to admin if there's no user found or no value received from the remote side.

```
// Single User
issue.customFields."Original User".value = nodeHelper.getUserByEmail(replica.customFields."User"?.value?.email) ?:
nodeHelper.getUserByUsername("admin")

// Multi User

def remoteUsers = replica.customFields."Multi Users"?.value
issue.customFields."Multi Users".value = remoteUsers?.collect{nodeHelper.getUserByEmail(it.email)} ?: []
```

Syncing other Types of Custom Fields

Handling Custom Field with Options

Jira has multiple custom fields with options:

- Select list (Single choice)
- Select list (Multiple choices)
- Select list radio button
- Select list checkbox

Exalate has a specific helper method getOption which handles custom fields that use the value type Option.

Syncing Cascading Select Fields

This type of custom field has a value of type cascading. The field value consists of two options: parent option and child option. For more information, please read how to sync cascading select custom fields.

Syncing Version Picker Fields (Single/Multiple Versions)

Outgoing sync

```
replica.customFields."Multiple Versions" = issue.customFields."Multiple Versions"
```

Incoming sync

> **Note**: The example below handles the following cases:
>
> - when the replica is sent with one version only
> - when the replica is sent with a list of multiple versions
> - when the replica is sent empty to set it to an empty list instead of getting an error

```
def remoteVersions = replica.customFields."Multiple Versions"?.value
issue.customFields."Multiple Versions".value =
  remoteVersions?.collect { v -> nodeHelper.createVersion(issue, v.name, v.description) } ?: []
```

User Picker (Single User)

For the custom field which is the user picker type (single user), you can use the getUserByEmail helper method. Set the Original User based on the received user email.

Incoming Sync

```
issue.customFields."Original User".value = nodeHelper.getUserByEmail(replica.customFields."User"?.value?.email)
```

More Complex Custom Fields Synchronization

Exalate allows syncing any kind of custom field type combination:

- sync labels from the source issue to a text custom field on the other side
- sync select list to a text custom field
- sync select list to user picker

In order to transform custom field data from one value type to another, you need to know:

- Sending custom field type (replica)
- Receiving custom field type (issue)

It is important to know what to expect when calling `.value` on the field.

Examples

A text custom field based on the remote user picker (single user) field. Make sure that after getting the user from the custom field, you get the specific value you need from it (in this case the username):

```
issue.customFields."Text Field Name".value = replica.customFields."Users Custom Fields"?.value.username
```

A text custom field based on the remote SelectList (multiple choices) field. Make sure that after getting the option from the custom field, you get the specific value you need from it (in this case of multiple options):

```
issue.customFields."Text Field Name".value = replica.customFields."Multi Options"?.value?.collect{it.value}.join(',')
```