# How to synchronize list custom fields

JIRA SERVER | JIRA CLOUD

This article describes how to synchronize custom fields that have options as a value type.

Jira has different custom field types with options:

- Select list (Single choice)
- Radio buttons (list of radio buttons)
- Select list (Multiple choices)
- Checkboxes (multiple choices)

Below you can find some examples on how to handle the most common cases related to the custom fields with option values.



## Select list (single choice)

Custom fields with types ***Select List(single choice)*** and ***Radio buttons(single choice)*** have values of type Option.

You can  apply the received option in different ways on your side:

- set an option manually directly in the Sync Rules

```
issue.customFields."Hobby".value = "Running"
```

- add the received value from the remote custom field into the local custom field

  If the received custom field does not have any values yet, you can still set a custom field with a default value as empty

```
issue.customFields."Hobby".value = replica.customFields."Interest"?.value
```

- Set a default option in case there's no value received from the remote side

```
issue.customFields."Hobby".value =        replica.customFields."Interest"?.value ?:  "Hobby"
```

- Map options between instances

```
def hobbyMap = ["Futbol": "Soccer", "Beisbol": "Baseball"] // ["remote options" : "local option"]
def remoteHobby = replica.customFields."Hobby"?.value?.value
issue.customFields."Hobby".value = hobbyMap[remoteHobby] ?: remoteHobby
```

## Select list (multiple choices)

Custom fields with types ***Select List(multiple choices)*** and ***Checkboxes(multiple choices)*** have values of type list of Options.

Below you can find some examples of how to handle custom field options in ***multi-value custom fields.***

- Set an option value manually into the local custom field

```
issue.customFields."Multi Select Custom Field name".value =
               (issue.customFields."Multi Select Custom Field name".value ?: []) +
                   nodeHelper.getOption(issue, "Multi Select Custom Field name", "MY STATIC OPTION")
```

- add the value from the remote custom field ***Departments***  into the local custom field ***Sub-divisions***

  leave an empty field in case there's no option, received from the remote side

```
issue.customFields."Sub-divisions".value = replica.customFields."Departments"?.value
```

- Set a default option in case there's no value received from the remote side

  You need to ensure that the value is an array of options.

  ```
  issue.customFields."Hobby".value = replica.customFields."Interest"?.value ?: ["Running"]
  ```

- Map options in a select list

  ```
  def hobbyMap = ["Football": "Sports", "Baseball": "Basketball"]
  def remoteHobbies = replica.customFields."Hobby"?.value?.value
  issue.customFields."Hobby".value = remoteHobbies.collect{hobbyMap[it] ?: it}
  ```

## See also

- Custom fields in Jira Cloud
- How to sync Story Points custom fields in Jira Cloud
- Exalate for Jira Cloud: Setting a project based on the source issue custom field value
- Exalate for Jira Cloud: Displaying a remote issue link in a custom field
- Jira Cloud: Sync between the Select List(single choice) custom field and issue type
- Updating a local custom field with a remote issue key in Exalate for Jira Cloud
- Syncing an assignee to a text custom field in Exalate for Jira Cloud
- Jira Cloud: Sync Select List (single choice) to comment field
- Overlap between custom fields and system fields
- How to sync story points field
- How to update a local custom field with the remote issue key
- Sync between the Select List(single choice) custom field and issue type
- Sync between Select List (single choice) and user picker custom field
- Sync Select List (single choice) to comment field
- How to set the project based on the source issue custom field value