# How to sync issue priority fields in Jira Cloud

Last Modified on 02/28/2024 6:56 am EST

This article shows how to sync issue priority.

**Warning**: Despite our best efforts, code can change without notice due to a variety of factors. If you encounter an issue in any of the code shown here and find that a specific block of code is not correct, or is causing errors, please check with the Community to find an updated version.

## Source side

To send priority field data to the destination side add the code below to your incoming sync processor.

### Outgoing sync (Data Filter)

```
replica.priority       = issue.priority
```

## Destination side

- Jira Cloud
- Jira Server 7.5 and lower/ Zendesk
- Jira Server 7.6 and higher

### Jira Cloud

Set the local priority based on the received priority value. If there's no such priority on your side - set the default priority to "**Major**".

```
def defaultPriority = nodeHelper.getPriority("Major")
issue.priority = nodeHelper.getPriority(replica.priority?.name) ?: defaultPriority
```

- Sync priority with mapping:

Set local priority based on the received urgency from the Servicenow side. If the priority name does not exist on the local side - set default priority to "Low".

```
def priorityMapping = [
        // remote side priority <-> local side priority
          "Critical" : "High",
          "Medium" : "Medium",
          "Minor" : "Low"
    ]
def priorityName = priorityMapping[replica.priority?.name] ?: "Low" // set default priority in case the proper urgency could not be found
issue.priority = nodeHelper.getPriority(priorityName)
```

## Jira Server 7.5 and lower/ Zendesk

```
if (replica.assignee.displayName == "Steve Jobs") {
   issue.priority = nodeHelper.getPriority("Critical")
}
```

- Sync priority with mapping

Set local priority based on the received urgency from the Servicenow side. If the priority name does not exist on the local side - set default priority to "Low".

```
def priorityMapping = [
        // remote side priority <-> local side priority
          "Critical" : "High",
          "Medium" : "Medium",
          "Minor" : "Low"
    ]
def priorityName = priorityMapping[replica.priority?.name] ?: "Low" // set default priority in case the proper urgency could not be found
issue.priority = nodeHelper.getPriority(priorityName)
```

## Jira Server 7.6 and higher

Use a nodeHelper.getPriority method to find local priority by the remote *priorityName.*

### Incoming sync (Create/Change processor)

There are several ways to manage incoming sync data related to the priorities:

- Find local priority by the remote *priorityName.* If there's no such priority on your side, don't sync the priority.

In this case, you need to be sure that both sides have the same priorities.

```
issue.priority = nodeHelper.getPriority(replica.priority?.name, issue)
```

- Set a default priority if the local priority is not found by the remote priority name.

```
issue.priority = nodeHelper.getPriority(replica.priority?.name, issue) ?: nodeHelper.getPriority('Major', issue)
```

- Sync priority with mapping:

Set local priority based on the received urgency from the Servicenow side. If the priority name does not exist on the local side - set default priority to "Low".

```
def priorityMapping = [
        // remote side priority <-> local side priority
          "Critical" : "High",
          "Medium" : "Medium",
          "Minor" : "Low"
    ]
def priorityName = priorityMapping[replica.priority?.name] ?: "Low" // set default priority in case the proper urgency c
ould not be found
issue.priority = nodeHelper.getPriority(priorityName)
```

ON THIS PAGE

Source side

Destination side