

# How to Sync Components in Jira Cloud

Last Modified on 01/28/2026 6:44 am EST

The components field is one of the standard fields of an issue. You can synchronize them as any other issue field.

This article shows you how to synchronize issue components and create a new component if it does not exist on the destination side.

[Component issue field](#) reference.

## Source Instance

### Outgoing sync

Send issue components to the destination side:

```
replica.components = issue.components
```

## Destination Instance

Exalate uses [CreateComponent](#) nodeHelper to create a new component and add it to the *Components* field. Add the code below into the **Incoming sync** to create components on your side. To update components use the same code in the Incoming sync(change processor).

- [Jira on-premise](#)
- [Jira Cloud](#)
- [How to assign a synced issue to an existing component?](#)

## Jira on-premise

If you sync components with Jira Cloud there're several ways to create components, depending on the information you know about the remote side.

- ***map the remote component lead email to the local component lead email***

### Incoming sync

```

issue.components = replica.components.collect { component ->
def remoteComponentLeadEmail = component.lead?.email
def localComponentLeadName = nodeHelper.getUserByEmail(remoteComponentLeadEmail)
nodeHelper.createComponent(
    issue,
    component.name,
    component.description, // can also be null
    localComponentLeadName?.key, // can also be null
    component.assigneeType?.name() ?: "UNASSIGNED" // set a default as unassigned if there's no assignee type f
or the component
)
}

```

- ***set default component lead if the remote component lead email is not found***

### Incoming sync

```

def defaultUser = nodeHelper.getUserByEmail("default@email.com")
issue.components = replica.components.collect { component ->
def remoteComponentLeadEmail = component.lead?.email
def localComponentLeadName = nodeHelper.getUserByEmail(remoteComponentLeadEmail)?.key
nodeHelper.createComponent(
    issue,
    component.name,
    component.description, // can also be null
    localComponentLeadName ?: defaultUser.key, // can also be null
    component.assigneeType?.name() ?: "UNASSIGNED" // set a default as unassigned if there's no assignee type f
or the component
)
}

```

- ***Do not create a new component but try to look for the same one by name***

### Incoming sync

```

issue.components = replica.components
    .collect { remoteComponent ->
        nodeHelper.getComponent(
            remoteComponent.name,
            nodeHelper.getProject(issue.projectKey)
        )
    }
    .findAll()

```

If you sync components between Jira Server instances and the usernames match, add the code below into the **Incoming sync**

```

issue.components = replica.components.collect{
nodeHelper.createComponent(
    issue,
    it.name,
    it.description,
    it.leadKey,
    it.assigneeType.name()
)
}
...

```

## Jira Cloud

- If you want to sync the components with Jira Server, you need to ***map the remote component lead email to the local component lead email.***

### ***Incoming sync***

```
final def userMapping = [
  "remoteadmin@admin.com" : "localadmin@admin.com",
]
issue.components = replica.components.collect { component ->
  def remoteComponentLeadEmail = component.lead?.email
  def localComponentLeadKey = nodeHelper.getUserByEmail(userMapping[remoteComponentLeadEmail])?.key
  nodeHelper.createComponent(
    issue,
    component.name,
    component.description, // can also be null
    localComponentLeadKey, // can also be null
    component.assigneeType?.name() // can also be null
  )
}
```

- If you sync between Jira Cloud instances use the code below

**Note:** The user with the specified account must be a user in your instance.

```
issue.components = replica.components.collect {
  component ->
  nodeHelper.createComponent(
    issue,
    component.name,
    component.description,
    component.leadKey,
    component.assigneeType?.name()
  )
}
```

## How to assign a synced issue to an existing component?





The receiving side is looking for an existing component and assigns the issue to the lead of this component.

### **Incoming sync**

```
def component = nodeHelper.getProject(issue.project?.key ?: issue.projectKey).components.find {c-> c.name == "eITs"}
issue.assignee = component.lead
issue.components += component
```

Another possible way is using [getComponent](#) nodeHelper.

Product

[About Us](#)   
[Release History](#)   
[Classroom](#)   
[API Reference](#)   
def project = issue.project ?: nodeHelper.getProject(issue.projectKey)  
def component = nodeHelper.getComponent("eITs", project)  
issue.classroom = component.lead  
issue.components += component

[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](#) 