How to Sync Components in Jira On-premise

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

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

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 to the **Incoming sync**.

```
issue.components = replica.components.collect{
 nodeHelper.createComponent(
  issue,
  it.name,
  it.description,
  it.leadKey,
  it.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 == "elT
s"}
issue.assignee = component.lead
issue.components += component
```

Another possible way is using getComponent nodeHelper.

```
def project = issue.project ?: nodeHelper.getProject(issue.projectKey)
def component = nodeHelper.getComponent("elTs", project)
issue.assignee = component.lead
issue.components += component
```

ON THIS PAGE

Product

Source Instance

About Us ⧉

Release History ⧉

Destination Instance

Glossary ⧉

How to assign a Synced Issue to an Existing

API Reference ⧉

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 ⧉