Script rules examples

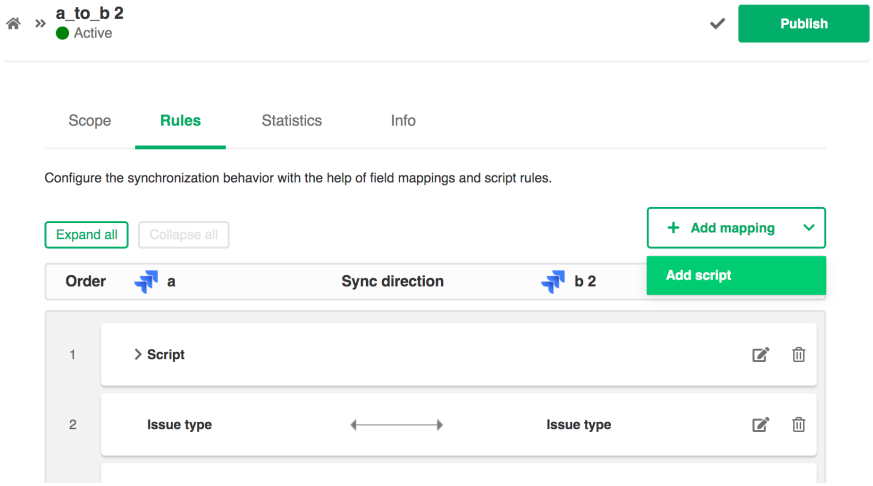
JIRA CLOUD | ZENDESK | JIRA SERVER | AZURE DEVOPS

This article describes how to write a script rule in Exalate visual mode.

Intro

Script rules can be used when the standard mapping rules are not sufficient to cover the use case.
Creating a script rule is straightforward.

- Edit your visual connection
- Select the rules
- Select the 'add script'



How does it work

The scripts are groovy-based - meaning that all groovy structures can be used to define the behavior of the mapping.

For instance, if a mapping is needed between the assignees of one side with the instance name 'left' and another side with the instance name 'right', the following code snippet will implement the mapping:

```
// define the mapping

def leftToRightAssignee = [
    // left Assignee ---> right Assignee
    "peter@acme.com" : "peter.pan@acme.com",
    "cinderella@acme.com" : "cinderalla.white@acme.com",
]


// look up the corresponding email, default to team@acme.com
def targetUserEmail = leftToRightAssignee[left.issue.assignee?.email] ?: "team@acme.com"

// assign to right issue

right.issue.assignee = nodeHelper.getUserByEmail(targetUserEmail)
```

Examples

Below you can find examples of the most common field types that are usually synchronized

Labels

```
your_instance_shortname.issue.labels = remote_instance_shortname.issue.labels
```

Components

```
your_instance_shortname.issue.components = remote_instance_shortname.issue.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() // can also be null
    )
}
```

Resolution

Set the local resolution same as on the remote side, if there's no such resolution on your side don't set anything

```
if(nodeHelper.getResolution(remote_instance_shortname.issue.resolution?.name)) {
     your_instance_shortname.issue.resolution = remote_instance_shortname.issue.resolution
}
```

Versions

```
// assign fix versions from JIRA A to JIRA B
your_instance_shortname.issue.fixVersions = remote_instance_shortname.
  .fixVersions
  // ensure that all the fixVersions are available on B
  .collect { v -> nodeHelper.createVersion(issue, v.name, v.description) }
// assign affected versions from JIRA A to JIRA B
your_instance_shortname.issue.affectedVersions = remote_instance_shortname
  .affectedVersions
  .collect { v -> nodeHelper.createVersion(issue, v.name, v.description) }
```

User fields

- assignee

  ```
  your_instance_shortname.issue.assignee = nodeHelper.getUser(remote_instance_shortname.issue.assignee?.
  key)
  ```

- reporter

  ```
  your_instance_shortname.issue.reporter = nodeHelper.getUser(remote_instance_shortname.issue.reporter?.
  key)
  ```

Custom fields

Text/String custom fields

Sync value from "remote side select list custom field" to the local "select list custom field"

```
your_instance_shortname.issue.customFields."text custom field".value = remote_instance_shortname.issue.
customFields."remote side text custom field".value
```

Set a fixed value in the local custom field

```
your_instance_shortname.issue.customFields."My CF".value = "Exalate"
```

Single select list/radio button

Sync value from "remote side select list custom field" to the local "select list custom field"

```
your_instance_shortname.issue.customFields."select list custom field".value = remote_instance_shortname.issue.
customFields."remote side select list custom field".value
```

Set a fixed value in the local custom fields "My select list"

```
your_instance_shortname.issue.customFields."My Select list".value = "Red"
```

Multi-select list/Checkbox

```
// sync value from "remote multi-select list custom field" to the local "select list multiple choice"
your_instance_shortname.issue.customFields."select list multiple choice".value = remote_instance_shortname.
issue.customFields."remote multi-select list custom field".value?.value
// Add "Red" as a value in the custom fields "My multi-select list"
your_instance_shortname.issue.customFields."My multi-select list".value += nodeHelper.getOption("Red")
```

Multi-cascade custom fields

Sync only existing option values

```
def sourceRegion = remote_instance_shortname.issue.customFields."Source Region/Country"?.value?.parent?.value
def sourceCountry = remote_instance_shortname.issue.customFields."Source Region/Country"?.value?.child?.value

def region = nodeHelper.getOption(
  issue,
  "Destination Region/Country",
  sourceRegion
)
def country = region.childOptions.find{it.value == sourceCountry}
if ( region != null && (sourceCountry == null || country != null)) {
  your_instance_shortname.issue.customFields."Destination Region/Country"?.value = nodeHelper.getCascadingSelect
(
              region,
              country
  )
} else if (sourceRegion == null) {
  your_instance_shortname.issue.customFields."Destination Region/Country"?.value = null
}
```

Date/DateTime custom fields

```
// if you have a custom field called "My Date" (of type Date Picker or Date Time Picker)
// on your Side and you'd like to populate it from
// "Their Date" of remote Side (of type Date Picker or Date Time Picker)
your_instance_shortname.issue.customFields."My Date".value = remote_instance_shortname.issue.customFields."
Their Date".value
// or if you'd like to assign a fixed moment in time:
your_instance_shortname.issue.customFields."My Date".value = new java.text.SimpleDateFormat("yyyy-MM-dd HH:mm:
ss z")
    .parse("2019-10-24 13:30:59 EET")
```

URL custom fields

```
// sync value from "remote side url custom field" to the local "url custom field"
your_instance_shortname.issue.customFields."url custom field".value = remote_instance_shortname.issue.
customFields."remote side url custom field".value

// Set a fixed value "https://exalate.com" in the custom field with name  "My url custom field"
your_instance_shortname.issue.customFields."My url custom field".value = "https://exalate.com"
```

Label custom fields

```
// sync value from "remote side labels" to the local "My labels"
your_instance_shortname.issue.customFields."My labels".value = remote_instance_shortname.issue.customFields."
remote side labels".value
// add "attention" to the custom field "My labels"
your_instance_shortname.issue.customFields."My labels".value += nodeHelper.getLabel("attention")
```

User picker custom fields

```
// sync value from "remote side user picker custom field" to the local "user picker custom field"
your_instance_shortname.issue.customFields."user picker custom field".value = nodeHelper.getUser
(remote_instance_shortname.issue.customFields."remote side user picker custom field".value)
// Set a fixed value "557358:bda57a72g56a9-4219-9c29-7d666481388f" (id for a user in your system) in the custom
field with name  "My user picker"
your_instance_shortname.issue.customFields."My user picker".value = "557358:bda57a72g56a9-4219-9c29-
7d666481388f"
```

Number custom fields

```
your_instance_shortname.issue.customFields."numeric custom field".value = remote_instance_shortname.issue.
customFields."remote side numeric custom field".value
```