

Using Field Definition Rule to request data from JIRA and set it to ConfiForms fields

 In this tutorial you will learn by example how to setup a rule to lookup data from your JIRA server and set it to ConfiForms fields

 If you are new to ConfiForms, please take a few minutes to read the [Basic concepts](#) Guide, or go through the video tutorial on that page. 😊

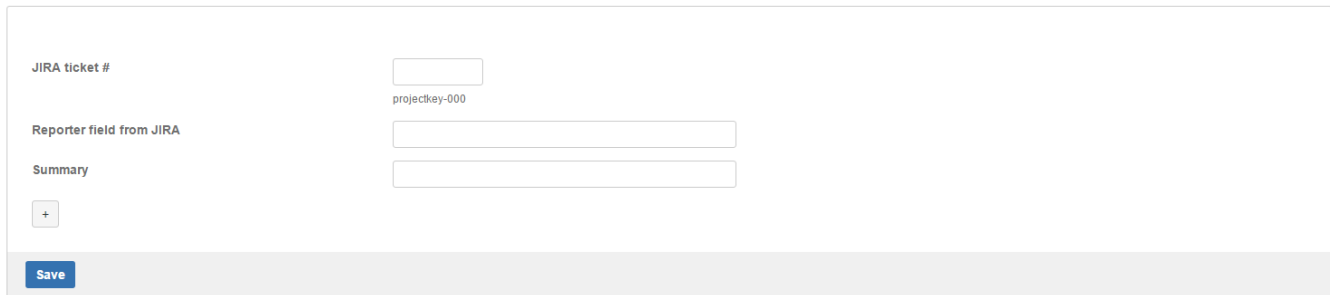
Let's start with a simple form

It will have just 3 fields

- A field (called "keyCF") to ask for JIRA key (can be of any type, as long as the value it holds is a JIRA key, as we will use it to make lookups)
- ConfiForms field (called "reporterCF") to hold Reporter name (reporter username in JIRA)
- ConfiForms field (called "summaryCF") to hold Issue summary

For clarity we name our ConfiForms fields with postfix "CF"

The form looks like this in view mode:



JIRA ticket #

Reporter field from JIRA

Summary

Same form in edit mode:



CONFI: ConfiForms Form (Definition) | formName = f

CONFI: ConfiForms Form Registration Control | embedded = true

CONFI: ConfiForms Form Field (Definition) | fieldName = keyCF | fieldLa...

CONFI: ConfiForms Rules for Field Definition | actionFieldName = keyCF | c...

CONFI: ConfiForms Form Field (Definition) | fieldName = reporterCF | fi...

CONFI: ConfiForms Form Field (Definition) | fieldName = summaryCF | fie...

And it's storage format

```

<ac:structured-macro ac:macro-id="0ac060cf-c01e-4b08-bccf-6ecba9bc13a2" ac:name="conform" ac:schema-version="1">
  <ac:parameter ac:name="formName">f</ac:parameter>
  <ac:rich-text-body>
    <ac:structured-macro ac:macro-id="ffda1fef-a7fc-4eda-8659-ef0107f79012" ac:name="conform-entry-register"
ac:schema-version="1">
      <ac:parameter ac:name="embedded">true</ac:parameter>
      <ac:rich-text-body>
        <p> </p>
      </ac:rich-text-body>
    </ac:structured-macro>
    <p>
      <ac:structured-macro ac:macro-id="e09ebd7b-dde0-4a2b-8b81-aa6d41f703ac" ac:name="conform-field-
definition" ac:schema-version="1">
        <ac:parameter ac:name="css">max-width:100px;width:100px;</ac:parameter>
        <ac:parameter ac:name="fieldName">keyCF</ac:parameter>
        <ac:parameter ac:name="fieldLabel">JIRA ticket #</ac:parameter>
        <ac:parameter ac:name="fieldDescription">projectkey-000</ac:parameter>
        <ac:parameter ac:name="type">text</ac:parameter>
      </ac:structured-macro>
    </p>
    <p>
      <ac:structured-macro ac:macro-id="4371f87c-8e6d-4b7f-9d6b-f1f9b71f66f6" ac:name="conform-field-
definition-rules" ac:schema-version="1">
        <ac:parameter ac:name="condition">!keyCF:[empty]</ac:parameter>
        <ac:parameter ac:name="fieldName">reporterCF=fields.reporter.name&amp;summaryCF=fields.summary</ac:
parameter>
        <ac:parameter ac:name="values">[entry.keyCF]</ac:parameter>
        <ac:parameter ac:name="action">Lookup and set JIRA issue data</ac:parameter>
        <ac:parameter ac:name="actionFieldName">keyCF</ac:parameter>
        <ac:parameter ac:name="values2">JIRA</ac:parameter>
      </ac:structured-macro>
    </p>
    <p>
      <ac:structured-macro ac:macro-id="36f3ebd8-aa8e-458a-bf0d-78a8cf672b6b" ac:name="conform-field-
definition" ac:schema-version="1">
        <ac:parameter ac:name="fieldName">reporterCF</ac:parameter>
        <ac:parameter ac:name="fieldLabel">Reporter field from JIRA</ac:parameter>
        <ac:parameter ac:name="type">text</ac:parameter>
      </ac:structured-macro>
    </p>
    <p>
      <ac:structured-macro ac:macro-id="20302f9b-ba51-4aeb-bb65-e5adffdb09bf" ac:name="conform-field-
definition" ac:schema-version="1">
        <ac:parameter ac:name="fieldName">summaryCF</ac:parameter>
        <ac:parameter ac:name="fieldLabel">Summary</ac:parameter>
        <ac:parameter ac:name="type">text</ac:parameter>
      </ac:structured-macro>
    </p>
  </ac:rich-text-body>
</ac:structured-macro>

```

The most interesting bit is the definition rule itself

```

<ac:structured-macro ac:macro-id="4371f87c-8e6d-4b7f-9d6b-f1f9b71f66f6" ac:name="conform-field-
definition-rules" ac:schema-version="1">
  <ac:parameter ac:name="condition">!keyCF:[empty]</ac:parameter>
  <ac:parameter ac:name="fieldName">reporterCF=fields.reporter.name&amp;summaryCF=fields.summary</ac:
parameter>
  <ac:parameter ac:name="values">[entry.keyCF]</ac:parameter>
  <ac:parameter ac:name="action">Lookup and set JIRA issue data</ac:parameter>
  <ac:parameter ac:name="actionFieldName">keyCF</ac:parameter>
  <ac:parameter ac:name="values2">JIRA</ac:parameter>
</ac:structured-macro>

```

Or in the macro editor:

Edit 'ConfiForms Rules for Field Definition' Macro

Rules for ConfiForms Field
Definitions [Documentation](#)

Field name

Name of the field you want to track
(optional for validation rules and for rules
which should be executed ONLY on users
action)

Condition

Action will be executed when condition is
met. [Same syntax expected as in filters](#)
The scope is current entry/record (for
dataset validation the scope is all records).
You can reference current records fields
via [entry.field_name]

☐ Execute only on user action

 Preview

So, we setup a rule to be executed when "keyCF" field value changes and will execute it when the condition is met and that means when a field has some value (is not empty)

(macro configuration continues)

Action *

Choose Action to perform

Values to set

Context is the JIRA issue you have
LOOKED UP. As key=value pairs,
separated by &. Where key is the field
name in the form where this entry should
be created and value is the value to be
set. You can reference record owner as
[owner]; record modifier, as [modifier]; any
record field as [entry.FIELD_NAME]

JIRA Issue Key

Context is CURRENT record. You can
reference record owner as [owner]; record
modifier, as [modifier]; any record field as

 Preview

As action we select "Lookup and set JIRA issue data" and it will take the value for JIRA lookup from "keyCF"

And the most important bit is "Values to set" parameter and how the JIRA issue data is mapped to ConfiForms fields

```
reporterCF=fields.reporter.name&summaryCF=fields.summary
```

reporterCF and **summaryCF** are ConfiForms fields

and "**fields.reporter.name**" and "**fields.summary**" are fields from JIRA

As the integration between ConfiForms and JIRA is done using JIRA REST API (through application links) - we work on a JSON served by JIRA

Here is an example JSON served by JIRA

```
"expand": "renderedFields,names,schema,operations,editmeta,changelog,versionedRepresentations",
"id": "14401",
"self": "https://vertuna.atlassian.net/rest/api/latest/issue/14401",
"key": "TEST-42",
"fields": {
  "issuetype": { _ }, // 7 items
  "timespent": null,
  "project": { _ }, // 5 items
  "fixVersions": [],
  "aggregatetimespent": null,
  "resolution": null,
  "resolutiondate": null,
  "workratio": -1,
  "lastViewed": "2016-09-26T13:39:40.172+0300",
  "watches": { _ }, // 3 items
  "created": "2016-09-26T13:38:55.579+0300",
  "customfield_10020": null,
  "customfield_10021": null,
  "customfield_10022": null,
  "customfield_10023": "Not started",
  "customfield_10100": null,
  "priority": { _ }, // 4 items
  "customfield_10024": null,
  "customfield_10300": null,
  "labels": [],
  "customfield_10016": null,
  "customfield_10017": null,
  "customfield_10018": null,
  "customfield_10019": null,
  "timeestimate": null,
  "aggregatetimeoriginalestimate": null,
  "versions": [],
  "issuelinks": [],
  "assignee": { _ }, // 8 items
  "updated": "2016-09-26T13:38:55.579+0300",
  "status": { _ }, // 6 items
  "components": [],
  "timeoriginalestimate": null,
  "description": "d \n dd",
  "customfield_10012": "0|i000yn:",
  "customfield_10013": null,
  "customfield_10014": null,
  "customfield_10015": null,
  "timetracking": {},
  "customfield_10007": null,
  "customfield_10009": null,
  "attachment": [],
  "aggregatetimeestimate": null,
  "summary": "no label with label! ",
  "creator": { _ }, // 8 items
  "subtasks": [],
  "reporter": {
    self: "https://vertuna.atlassian.net/rest/api/2/user?username=admin",
    name: "admin",
    key: "admin",
```

To access "reporter's username" You will need to navigate the JSON using the following syntax "fields.reporter.name"

You can navigate any field from the served JSON, including custom fields and complex fields (such as user objects, watches, etc).

Couple of examples:

Finding what	Structure	Syntax
--------------	-----------	--------

assignee email	<pre> "assignee": { "self": "https://", "name": "admin", "key": "admin", "emailAddress": "avatarUrls": { "48x48": "ht "24x24": "ht "16x16": "ht "32x32": "ht }, "displayName": " "active": true, "timeZone": "Eur </pre>	fields.assignee.emailAddress
issue summary		fields.summary
Status name	<pre> "status": { "self": "https://ver "description": "", "iconUrl": "https:// "name": "To Do", "id": "10000", "statusCategory": { "self": "https:// "id": 2, "key": "new", "colorName": "bl "name": "To Do" } }, </pre>	fields.status.name
Status color	<pre> "status": { "self": "https://ver "description": "", "iconUrl": "https:// "name": "To Do", "id": "10000", "statusCategory": { "self": "https:// "id": 2, "key": "new", "colorName": "bl "name": "To Do" } }, </pre>	fields.status.statusCategory.colorName
Issue key	<pre> { "expand": "renderex "id": "14401", "self": "https://ve "key": "TEST-42", "fields": { "issuetype": { "timespent": nu } "project": { _ } "fixVersions": "aggregatetimes "resolution": r "resolutiondate "workratio": -1 "lastViewed": " "watches": { _ } "-----": " </pre>	key (not fields.key! as it is under the JSON root, not under "fields", see the main screenshot with JIRA issue data)

This is it, the tutorial ends here but your joerney with ConfiForms just begins! Welcome and if you have any questions feel free to ask and share your feedback/concerns

Ask for support