

Building a dropdown field in ConfiForms backed by webservice call to Jira Rest API - components field

This is a variation of [Building a dropdown field in ConfiForms backed by webservice call to Jira Rest API - createmeta](#) but uses "components" REST API endpoint from Jira

From test project TEST into a dropdown field

```
https://vertuna.atlassian.net/rest/api/2/project/TEST/components
```

Example response returned by the service

```
[
  {
    "self": "https://vertuna.atlassian.net/rest/api/2/component/10001",
    "id": "10001",
    "name": "comp1",
    "description": "my test componebt 1",
    "assigneeType": "PROJECT_DEFAULT",
    "realAssigneeType": "PROJECT_DEFAULT",
    "isAssigneeTypeValid": false,
    "project": "TEST",
    "projectId": 10200
  },
  {
    "self": "https://vertuna.atlassian.net/rest/api/2/component/10002",
    "id": "10002",
    "name": "test 2",
    "description": "My test component 2",
    "assigneeType": "PROJECT_DEFAULT",
    "realAssigneeType": "PROJECT_DEFAULT",
    "isAssigneeTypeValid": false,
    "project": "TEST",
    "projectId": 10200
  }
]
```

Form

Components abccomp1confitest 2vertz

Components multi abccomp1confitest 2vertz

Save

Field configuration

Components field could be configured like this (of course complete configuration depends on YOUR project in Jira...)

Edit 'ConfiForms Field Definition' Macro

Webbservice Multi-select

Webbservice connection

vertuna-test

Manage connections

Service URL

/rest/api/2/project/JTEST/compon

Your service must return an array of JSON objects (or a single JSON object) in order for this field to work. Or you can specify a root to use (Example: fieldname.anotherfield) to navigate to your array of elements.

You can access context variables here: [entry._user], [entry._now], as well as the value for the "ID" field you have mapped, also via [entry.field_name] notation (where field_name is the name of the field you have put in "ID" mapping

Preview

Edit 'ConfiForms Field Definition

parameter, see below)

Root to use

When left empty, the complete JSON document will be taken as source using the mapping below

Field to use as "ID"

id

You can navigate using fields in a JSON tree, like: myfield.anotherfield. If 'Root to use' parameter is given, then you start your navigation from a given root. You can map multiple values by referencing them via [entry.field_name]

Field to use as "Label"

name

You can navigate through the JSON tree using fieldname.subfield.someotherfield to point at the specific value. You can map multiple values by referencing them

Show data

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10032" }
```

abc

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "abc" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10036" }
```

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10032" }
```

abc

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "abc" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10036" }
```

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

entry.components.transform(id).asArrayOfKVPairs(my id)

abc

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "abc" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10036" }
```

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10032" }
```

comp1confi

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "comp1" }, { "name" : "confi" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10032" }, { "my id" : "10038" }
```

abc

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "abc" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10036" }
```

abc

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "abc" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10036" }
```

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10032" }
```

abc

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "abc" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10036" }
```

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

entry.components.transform(id).asArrayOfKVPairs(my id)

comp1confi

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "comp1" }, { "name": "confi" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10032" }, { "my id": "10038" }
```

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

entry.components.transform(id).asArrayOfKVPairs(my id)

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

entry.components.transform(id).asArrayOfKVPairs(my id)

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10032" }
```

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10032" }
```

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10032" }
```

abc

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "abc" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10036" }
```

abc

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "abc" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10036" }
```

confi

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "confi" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10038" }
```

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10032" }
```

confi

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "confi" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10038" }
```

abc

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "abc" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10036" }
```


abccomp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "abc" }, { "name": "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10036" }, { "my id": "10032" }
```

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10032" }
```

confi

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "confi" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10038" }
```

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10032" }
```

comp1test 2

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "comp1" }, { "name": "test 2" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10032" }, { "my id": "10033" }
```

abctest 2

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "abc" }, { "name": "test 2" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10036" }, { "my id": "10033" }
```

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

entry.components.transform(id).asArrayOfKVPairs(my id)

test 2

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "test 2" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10033" }
```

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10032" }
```

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10032" }
```

abccconfi

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "abc" }, { "name": "confi" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10036" }, { "my id": "10038" }
```

confi

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name": "confi" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id": "10038" }
```

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10032" }
```

comp1

Transformation via "asArrayOfKVPairs" [Virtual functions](#) as a bonus 😊

entry.components.transform(name).asArrayOfKVPairs(name)

```
{ "name" : "comp1" }
```

entry.components.transform(id).asArrayOfKVPairs(my id)

```
{ "my id" : "10032" }
```

Storage format (form configuration in page storage format)

```

<ac:structured-macro ac:macro-id="ea177e80-12c8-4166-9541-13ebce02210e" ac:name="confiiform-entry-register" ac:
schema-version="1">
<ac:parameter ac:name="embedded">true</ac:parameter>
<ac:parameter ac:name="atlassian-macro-output-type">INLINE</ac:parameter>
<ac:rich-text-body>
<p>
<br/>
</p>
</ac:rich-text-body>
</ac:structured-macro>
<ac:structured-macro ac:macro-id="81c1440c-989a-4324-b4c3-3840aafa4024" ac:name="confiiform" ac:schema-version="
1">
<ac:parameter ac:name="formName">f</ac:parameter>
<ac:rich-text-body>
<p>
<ac:structured-macro ac:macro-id="232c855f-6dbe-4ea5-b332-f37958081da7" ac:name="confiiform-field-definition" ac:
schema-version="1">
<ac:parameter ac:name="mapping">|id|name</ac:parameter>
<ac:parameter ac:name="fieldName">components</ac:parameter>
<ac:parameter ac:name="fieldLabel">Components</ac:parameter>
<ac:parameter ac:name="values">/rest/api/2/project/JTEST/components</ac:parameter>
<ac:parameter ac:name="extras">1f170724ecd07e7d7046c26f8c71ee81</ac:parameter>
<ac:parameter ac:name="type">wsmultiselect</ac:parameter>
</ac:structured-macro>
</p>
<p>
<ac:structured-macro ac:macro-id="464b50d5-1251-415b-b82a-aa5fab24b95c" ac:name="confiiform-field-definition" ac:
schema-version="1">
<ac:parameter ac:name="mapping">|id|name</ac:parameter>
<ac:parameter ac:name="fieldName">components2</ac:parameter>
<ac:parameter ac:name="fieldLabel">Components multi</ac:parameter>
<ac:parameter ac:name="values">/rest/api/2/project/JTEST/components</ac:parameter>
<ac:parameter ac:name="extras">1f170724ecd07e7d7046c26f8c71ee81</ac:parameter>
<ac:parameter ac:name="type">wsmultiselect</ac:parameter>
</ac:structured-macro>
</p>
</ac:rich-text-body>
</ac:structured-macro>

```

Storage format for the "view"

```

<ac:structured-macro ac:macro-id="5f771dae-aa58-4ee1-84c8-f7046ced1bef" ac:name="confiiform-cleanview" ac:schema-
version="1">
<ac:parameter ac:name="formName">f</ac:parameter>
<ac:rich-text-body>
<p>
<ac:structured-macro ac:macro-id="906e7297-df0d-4662-968d-ae402c0e9212" ac:name="confiiform-field" ac:schema-
version="1">
<ac:parameter ac:name="fieldName">components</ac:parameter>
</ac:structured-macro>
</p>
<p>Tranformation via "asArrayOfKVPairs" <ac:link>
<ri:page ri:content-title="Virtual functions" ri:space-key="CONFIFORMS"/>
</ac:link> as a bonus <ac:emoticon ac:name="smile"/>
</p>
<ac:structured-macro ac:macro-id="7b2569a0-e149-45d8-9842-2a7ee77d9e57" ac:name="code" ac:schema-version="1">
<ac:plain-text-body>[entry.components.transform(name).asArrayOfKVPairs(name)]</ac:plain-text-body>
</ac:structured-macro>
</ac:rich-text-body>
</ac:structured-macro>

```

