

Use field value as a key for referencing an issue in different project and reading field values in referenced issue

On this page

- [Features used to implement the example](#)
- [Example: Use field value as a key for referencing an issue in different project and reading field values in referenced issue](#)
- [Other examples of that functions](#)
- [Related Usage Examples](#)

Features used to implement the example

- [Read field from issues returned by JQL query or issue list](#)

Example: Use field value as a key for referencing an issue in different project and reading field values in referenced issue

I use **Contact Manager** app and **Email this issue** app for Jira Service Desk.

I have 2 projects :

- **Customer relationship Management (CRM)** i had stored all information about the customer.
- **Support team (ST)** Jira Help Desk project

When a external user sends an email, a Mail handler create an issue. "Email this issue" take the email of **"To"** and write it in **"Customer Contact Email"** field in **ST project**.

In ST project there is a custom field for autocomplete the contact of the issue and when we click on it, we have all information stored in CRM project.

With your plugin i want to check the value of **"Customer Contact Email"** that "Email this issue" has set and search in CRM project if there are something with this value :

Example: `JQL: project = CRM and "Customer Contact Email" ~ "albertini.olivier@gmail.com"`

and take the field Contact and write the value in the **"Contact"** field of ST project.

Because now the support team, check the value of **"Customer Contact Email"** and rewrite in **"Contact"** field in order to have an autocomplete and having a link with the issue of **CRM project**.

I have some questions in order to fully understand your use case:

1. Is there a field **"Customer Contact Email"** in both projects **CRM** and **ST**?
2. Is there a field **"Contact"** in both projects **CRM** and **ST**?
3. In case you don't have information in project **CRM** about the customer, is there something special to do?

I need some clarification about the following fragments of your explanation:

1) **"...take the field Contact and write the value in the "Contact" field of ST project."**

Does it mean that field **"Contact"** exists in both projects?

2) **"Because now the support team, check the value of "Customer Contact Email" and rewrite in "Contact" field in order to have an autocomplete and having a link with the issue of CRM project."**

Is the value of field **"Contact"** in project **"CRM"** what you want to write into field **"Contact"** of project **"ST"**?

I explain the way to do it assuming that:

- You want to write the field **Contact** in an issue of project **ST** at the moment of being created.
- The value to be written is the value of field **Contact** in an issue of project **CRM** where the field **Customer Contact Email** has the same value as the field **Customer Contact Email** in the issue being created at project **ST**.

You have to use post-function [Read field from issues returned by JQL query or issue list](#) with the following configuration:

Target fields and Source values: ?

Select the target fields that will be set and the source values for each of them.

Target field:

Summary - [Text] Add

Add a field to be set in current issue.

Target Field	Type of Value	Source Value	Calculated Value	Don't Overwrite	Actions
Contact	Field in selected issues	Contact			Edit Remove

Issue Selection:

Issues whose fields are going to be read.

Issue Selection Mode:

☒ JQL Query ☐ Issue List

[Line 1 / Col 54]

1

project = CRM and "Customer Contact Email" ~ %{10207}

String Field Code Injector:

Customer Contact Email - [Text Field (single line)] - %{10207}

Check Syntax

Numeric/Date Field Code Injector:

Original estimate (minutes) - [Number] - {00068}

- Field codes with format %{nnnnn} may be inserted in the JQL Query, and will be replaced with field values at runtime. Most times it's a good idea to write field codes between double quotes (e.g. "{00001}"), since field values may contain blank spaces that will produce JQL parsing errors at runtime.

- Cascading Select fields and Multi-level Cascading Select fields specific levels can be referenced with %{nnnnn.0} for parent level, %{nnnnn.1} for child level, etc.

Additional options:

☐ Update issue immediately after field writing. A specific entry will be created in issue history for this field writing.

Conditional execution:

Optional boolean expression that should be satisfied in order to actually execute the post-function.
(Syntax Specification)

1

Leave the field empty for executing the post-function unconditionally. [Collection of Examples](#) [Line 1 / Col 1]

Logical connectives: and, or and not. Alternatively you can also use &, | and !.

Comparison operators: =, !=, >, < and <=. Operators in, not in, any in, none in, ~ and != can be used with strings, multi-valued fields and lists.

Logical literals: true and false. Literal null is used with = and != to check whether a field is initialized, e.g. {00012} != null checks whether Due Date is initialized.

String Field Code Injector:

Summary - [Text] - %{000000}

Numeric/Date Field Code Injector:

Original estimate (minutes) - [Number] - {00068}

Check Syntax

Note that:

- `%{10207}` is the field code for field **"Customer Contact Email"**

Once configured, post-function **"Create Issue"** in the workflow of issues at project **"ST"** will look like this:

Create

OPEN

This is the **initial** transition in the workflow.

Screen: None - initial transition does not have a view.

Validators 1

Post Functions 6

The following will be processed after the transition occurs

Add post function

- Creates the issue originally.
- Fields in issues returned by the following **JQL** query will be read:

```
project = CRM and "Customer Contact Email" ~ %{Customer Contact Email}
```

Target fields and Source values:

Target Field	Type of Value	Source Value	Calculated Value	Don't Overwrite
Contact	Field in selected issues	Contact		

This feature will be run as user in field **Current user**. by JWT
- Re-index an issue to keep indexes in sync with the database.
- Fire a **Issue Created** event that can be processed by the listeners.

Other examples of that functions

Page: [Use field value as a key for referencing an issue in different project and reading field values in referenced issue](#)

Related Usage Examples

- [Creating a Jira Service Desk internal comment](#)
 - [example](#)
 - [post-function](#)
- [Limit the number of hours a user can log per day](#)
 - [example](#)
 - [validator](#)
 - [post-function](#)
 - [work-log](#)
- [Using project properties to calculate custom sequence numbers](#)
 - [example](#)
 - [post-function](#)
 - [calculated-field](#)
 - [project-properties](#)
- [Set a date based on current date](#)
 - [example](#)
 - [post-function](#)
- [Setting the priority depending on the multiplication of custom fields](#)
 - [example](#)
 - [calculated-field](#)
 - [post-function](#)
- [Parse Email addresses to watchers list](#)
 - [example](#)
 - [post-function](#)
- [Set the assignee based on a condition](#)
 - [example](#)
 - [post-function](#)
- [Create a dynamic set of sub-tasks based on checkbox selection with unique summaries](#)
 - [example](#)
 - [post-function](#)
 - [custom-field](#)
 - [sub-task](#)
- [Create a static set of sub-tasks with unique summaries](#)
 - [example](#)
 - [post-function](#)

- Triage Jira Service Desk email requests (Move issues)
 - example
 - post-function
 - move
 - transition-issue
- Moving story to "In Progress" when one of its sub-tasks is moved to "In Progress" (Transition issues)
 - example
 - post-function
 - transition
- Transition sub-tasks when parent is transitioned
 - example
 - post-function
 - sub-task
 - transition
 - outdated
- Transition only a sub-task among several ones
 - example
 - post-function
 - sub-task
 - transition
 - outdated
- Moving sub-tasks to "Open" status when parent issue moves to "In Progress"
 - example
 - post-function
 - sub-task
 - transition
 - outdated
- Moving story to "Ready for QA" once all its sub-tasks are in "Ready for QA" status
 - example
 - post-function
 - sub-task
 - transition
 - outdated