

Assign issue to last user who executed a certain transition in the workflow

On this page

- [Features used to implement the example](#)
- [Example: Assign issue to last user who executed a certain transition in the workflow](#)
- [Other examples of that function](#)
- [Related Usage Examples](#)

Features used to implement the example

- [Copy parsed text to a field](#)
- Virtual field **"Assignee"**: writing a user name into this field will assign the issue to the corresponding user.
- Parser function **"usersWhoTransitioned()"**

FUNCTION	RETURNED VALUE
usersWhoTransitioned (string origin_status , string destination_status) : string list Available since version 2.2.7	returns a string list with the names of the users who transitioned current issue from origin_status to destination_status , order ascending by time. An empty string as argument is interpreted as any status . Example: <code>last(usersWhoTransitioned("Open", "In Progress"))</code> returns the name of the user who executed transition "Start Progress" more recently.

Example: Assign issue to last user who executed a certain transition in the workflow

We want to assign current issue to the last user who executed the transition **Start Progress**, which goes from **Open** to **In Progress** status.

For implementing that behavior we simply use [Copy parsed text to a field](#) post-function with the following configuration:

Target field:

Assignee - [User]

Field to be written with the resulting parsed text.

☐ Don't overwrite target field if it's already set.

Parsing Mode:

Basic

Basic mode: Insert field codes anywhere in the text, and they will be replaced with corresponding field values. Field code formats are `{nnnnn}`, and `{nnnnn.i}` for Cascading Select fields (i = 0 for base level).

Advanced

Advanced mode: Strings literals are written in double quotes ("This is a string."). Operator '+' is used to concatenate strings, and field codes are like in basic mode, e.g., "Issue key is " + `{00015}` + ".". More information at [parser syntax documentation](#).

Text to be parsed and then copied to target field:

[Line 1 / Col 51] [Syntax Specification](#) [Check Syntax](#)

1

`last(usersWhoTransitioned("Open", "In Progress"))`

Summary - [Text] - `{00000}`

Insert String Value

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

Insert Numeric Value

- **Compose dynamic text** by inserting field codes (`{nnnnn}`) that will be replaced with corresponding field values prior to be copied to target field.

- You can reference parent and child values of **cascading select fields** writing `{nnnnn.0}` for parent value, and `{nnnnn.1}` for child value. Use greater indexes to read **multi-level cascading select** custom fields.

- You can change **reporter**, **assignee**, **due date**, **issue status**, **priority**, **resolution**, **labels**, **components**, **fixed versions**, **affected versions**, **original estimate**, **estimated**, **time spent**, and **security level** by choosing the suitable target field and value to be assigned.

- To assign **cascading selects**, **multi selects**, **multi checkboxes**, **components**, **labels**, **fixed versions** and **affected versions** you should use comma or semicolon separated values.

- Additionally, fields of type **Select list**, **Radio button**, **Multi select**, **Multicheck box**, **Multi user**, **Multi groups**, **Components** and **Versions** can be set through regular expressions: options that matches a regular expression can be set by writing `/(regular_expression)/`.

- **Setting and unsetting individual values** in multi-valued fields, leaving the rest untouched, can be achieved simply by inserting a character '+' or '-' preceding the value or the list of values. You can insert more than one '+' / '-' character in a sole setting operation.

- Fields **Attachments** (**only new attachments will be added**) and **Attachments** (**all current attachments will be replaced**) expect one or more issue keys whose attachments will be copied to current issue.

- You can also use this post-function to **cast a string into a number**.

Conditional execution:

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

[\(Syntax Specification\)](#)

1

`count(usersWhoTransitioned("Open", "In Progress")) > 0`

Leave the field empty for executing the post-function unconditionally.

[Collection of Examples](#)

[Line 1 / Col 56]

[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] - `{00000}`

[Numeric/Date Field Code Injector:](#)

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

Run as:

Select the user that will be used to execute this feature. JIRA will apply restrictions according to the permissions, project roles and groups of the selected user.

Current user

User defined by a field.

Input a specific user.

Text to be parsed: `count(usersWhoTransitioned("Open", "In Progress")) > 0`

Conditional execution: `last(usersWhoTransitioned("Open", "In Progress"))`

We use `count(usersWhoTransitioned("Open", "In Progress")) > 0` for checking whether there is at least one user who transitioned the issue from **Open** to **In Progress**, and in affirmative case we use `last(usersWhoTransitioned("Open", "In Progress"))`.

Other examples of that function

[Page: Add all assignees of certain sub-task types to a "Multi-User Picker" custom field](#)
[Page: Add and remove a single or a set of items from multi valued fields](#)
[Page: Add current user to comment](#)
[Page: Add or remove request participants](#)
[Page: Add watchers from a part of the issue summary: "Summary_text - watcher1, watcher2, watcher3, ..."](#)
[Page: Assign issue based on the value of a Cascading Select custom field](#)
[Page: Assign issue to last user who executed a certain transition in the workflow](#)
[Page: Automatically close resolved sub-tasks when parent issue is closed](#)
[Page: Automatically reopen parent issue when one of its sub-tasks is reopened](#)
[Page: Calculate the time elapsed between 2 transition executions](#)
[Page: Close parent issue when all sub-tasks are closed](#)
[Page: Combine the values of several Multi-User picker fields](#)
[Page: Compose a parsed text including the "full name" or a user selected in a User Picker custom field](#)
[Page: Compose dynamic text by inserting field values in a text template](#)
[Page: Copy issue labels to a custom field](#)
[Page: Copy the value of a user property into a user picker](#)
[Page: Create a comment in sub-tasks when parent transitions](#)
[Page: Execute transition in epic](#)
[Page: Getting the number of selected values in a custom field of type Multi Select](#)
[Page: Limit the number of hours a user can log per day](#)
[Page: Make a sub-task's status match parent issue's current status on creation](#)
[Page: Make parent issue progress through its workflow](#)
[Page: Moving story to "In Progress" when one of its sub-tasks is moved to "In Progress"](#)
[Page: Moving story to "Ready for QA" once all its sub-tasks are in "Ready for QA" status](#)
[Page: Parse Email addresses to watchers list](#)
[Page: Parsing text from last comment and appending it to issue's summary](#)
[Page: Remove versions selected in a version picker custom field](#)
[Page: Replace certain issue link types with different ones](#)
[Page: Restrict parent issue from closing if it has sub-tasks that were created during a given parent issue status](#)
[Page: Set a Select or Multi-Select field using regular expression to express the values to be assigned](#)
[Page: Set assignee depending on issue type](#)
[Page: Set field depending on time passed since issue creation](#)
[Page: Set priority for issues that have been in a certain status for longer than 24 hours](#)
[Page: Set security level based on groups and project roles the reporter or creator are in](#)
[Page: Transition linked issues in currently active sprint](#)
[Page: Transition only a sub-task among several ones](#)
[Page: Transition parent issue only when certain issue sub-task types are done](#)
[Page: Update Cascading Select custom field with a value of the field in parent issue](#)
[Page: Update checkboxes custom field if a file has been attached during a transition](#)
[Page: Validation on issue attachments](#)
[Page: Validation on MIME types of issue attachments](#)
[Page: Writing a comment to blocked issues when blocking issues are resolved](#)

Related Usage Examples

- [Block or unblock a transition after an issue rested a specific time in a status](#)
 - [example](#)
 - [condition](#)
 - [validator](#)
 - [transition](#)
- [Block transition until all sub-tasks are in a specific status category](#)
 - [example](#)
 - [transition](#)
 - [condition](#)
- [Validation and condition based on time expressions](#)
 - [example](#)
 - [condition](#)
 - [validator](#)
 - [transition](#)
- [Validation on sibling sub-tasks depending on issue type and status](#)
 - [example](#)
 - [validator](#)
 - [sub-task](#)
 - [transition](#)
- [Set a condition in a global transition which only applies in a certain status](#)
 - [example](#)
 - [condition](#)
 - [transition](#)
- [Block a transition until all sub-tasks have certain fields populated](#)
 - [example](#)
 - [condition](#)
 - [validator](#)
 - [sub-task](#)
 - [transition](#)
- [Block an epic's transition depending on linked issues status and due date](#)
 - [example](#)
 - [validator](#)
 - [issue-links](#)
 - [transition](#)
- [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](#)
- [Change parent's status depending on sub-task's summary \(Transition issues\)](#)
 - [example](#)
 - [post-function](#)
 - [transition](#)

- Change parent's status depending on sub-task's summary
 - example
 - post-function
 - sub-task
 - transition
 - outdated
- Moving story to "In Progress" when one of its sub-tasks is moved to "In Progress"
 - example
 - post-function
 - sub-task
 - transition
 - outdated