

Create a comment in sub-tasks when parent transitions

On this page

- [Features used to implement the example](#)
- [Example: Create a comment in sub-tasks when parent transitions](#)
- [Other examples of that functions](#)
- [Related Usage Examples](#)

Features used to implement the example

- [Copy a parsed text to a field](#)
- [Write field on linked issues or sub-tasks](#)

Example: Create a comment in sub-tasks when parent transitions

I'd like to add a post-function so that when a parent issue is resolved all sub-tasks get a new comment in the form of: **"The parent issue of this subtask has been resolved as "<resolution goes here>" <Parent Issue Key>"**

You can do it using 2 post-functions in transition **"Resolve Issue"** of parent's workflow.

Post-function [Copy a parsed text to a field](#) with the following configuration:

Target field:	
Ephemeral string 1 - [Text]	
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 <code>%(nnnnn)</code> , and <code>%(nnnnn.i)</code> 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 " + <code>%(00015)</code> + ". ". More information at parser syntax documentation .
Text to be parsed and then copied to target field:	
[Line 1 / Col 1]	
Syntax Specification	
1	The parent issue of this subtask has been resolved as <code>"%(00028)": %(00015).</code>

Text to be parsed is:

The parent issue of this subtask has been resolved as `"%(00028)": %(00015).`

Note that:

- `%(00028)` is field code for **"Resolution"**
- `%(00015)` is field code for **"Issue Key"**

Post-function [Write field on linked issues or sub-tasks](#) with the following configuration:

Source field to be read in current issue:	Ephemeral string 1 - [Text]
Target field to be written in linked issues or subtasks:	New comment - [Text] ☐ Don't overwrite target field if it's already set.
Filtering by issue link type:	☐ is blocked by ☐ blocks ☐ is cloned by ☐ clones ☐ is duplicated by ☐ duplicates ☐ relates to ☐ relates to Only issues linked to current issue by selected issue link types will be written.
Write also subtasks fulfilling condition on issue type, status and project:	☒ This option only makes sense when current issue itself is not a subtask.
Write also sibling subtasks fulfilling condition on issue type, status and project:	☐ Sibling subtasks are understood as subtasks with the same parent as current issue. This option only makes sense when current issue is itself a subtask.
Filtering linked issues or subtasks by issue type:	☐  Bug ☐  Improvement ☐  New Feature ☐  Task ☐  Sub-task Selected issue types will be written, but if you don't select any, it won't be applied any filter by issue type. In that case all the issue types will be written.
Filtering linked issues or subtasks by status:	☐  Open ☐  In Progress ☐  Reopened ☐  Resolved ☐  Closed Selected statuses will be written, but if you don't select any, it won't be applied any filter by status. In that case issues in any status will be written.
Linked issues or subtasks belong to:	☒ any project ☐ current project ☐ any but current project

1

Filtering by field values:

Optional boolean expression that should be satisfied by issues in order to be selected. [\(Syntax Specification\)](#)

Leave field empty for no filtering.

[Line 1 / Col 1]

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

Comparison operators: =, !=, >, < and <=. Operators ~, !~, in, not in, any in and none in 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.

Numeric/Date Field Code Injector:

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

Field Code for Current Issue

Field Code for Foreign Issues

String Field Code Injector:

Summary - [Text] - %{00000}

Field Code for Current Issue

Field Code for Foreign Issues

Term "Foreign Issues" refers to issues susceptible to being filtered, i.e., linked issues, subtasks, or any issue different from current one.

Example 1: boolean condition {00012} <= ^{00012} will require that issues have Due Date equal or later than current issue's Due Date.

Example 2: boolean condition %{00074} ~ ^{00074} AND ^{00017} in ["Blocker", "Critical"] will require that issues have Fixed versions contained in current issue's Fixed versions and Priority is Blocker or Critical.

Write linked issues and subtasks recursively:

☐

Issues and subtasks transitively linked will also be written, provided they fulfill stated filtering conditions. Issues are written recursively without depth limit, but each selected issue is written only once.

Run as:

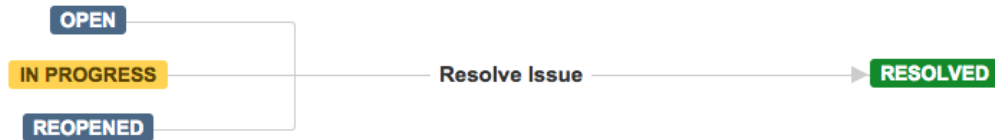
Select the user that will be used to execute this feature. JIRA will apply restrictions according to the permissions, project roles and groups of the selected user.

Current user

User defined by a field.

Input a certain fixed user. Expected value is a user name, don't confuse it with user's full name.

Once configured, transition "Resolve Issue" in parent's workflow will look like this:



Screen: Resolve Issue Screen

Conditions 1

Validators 0

Post Functions 7

The following will be processed after the transition occurs

[Add post function](#)

1. The following text parsed in **basic** mode will be copied to **Ephemeral string 1**:
The parent issue of this subtask has been resolved as "{Resolution}": {Issue key}.
 This feature will be run as user in field **Current user**.
2. Value of field **Ephemeral string 1** in current issue will be copied to field **New comment** in linked issues or subtasks filtering by:
 Inward issue link types: **none**
 Outward issue link types: **none**
Subtasks fulfilling conditions on issue type, status and project **will be written**.
Sibling subtasks won't be written.
 Issue types: **any**
 Statuses: **any**
 Linked issues or subtasks may belong to **any** project.
 This feature will be run as user in field **Current user**.

Other examples of that functions

[Write field on linked issues or sub-tasks](#)

Page: [Add and remove a single or a set of items from multi valued fields](#)

Page: [Automatically become watcher of every issue blocking an issue assigned to you](#)

Page: [Automatically close resolved sub-tasks when parent issue is closed](#)

Page: [Automatically resolve an epic when all its stories are resolved](#)

Page: [Compose dynamic text by inserting field values in a text template](#)

Page: [Copy "Due date" into a date type custom field in a linked issue if it's greater than current issue's "Due date"](#)

Page: [Copy attachments from one issue to another](#)

Page: [Create a comment in sub-tasks when parent transitions](#)

Page: [Creating a Jira Service Desk internal comment](#)

Page: [Creating a Jira Service Desk internal comment on linked issues](#)

Page: [Execute transition in epic](#)

Page: [Make linked issues, sub-tasks and JQL selected issues progress through its workflows](#)

Page: [Moving sub-tasks to "Open" status when parent issue moves to "In Progress"](#)

Page: [Sum sub-task's "Time Spent" \(work logs\) and add it to a certain linked issue](#)

Page: [Transition sub-tasks when parent is transitioned](#)

[Copy a parsed text to a field](#)

Related Usage Examples

- [Validation on sibling sub-tasks depending on issue type and status](#)
 - [example](#)
 - [validator](#)
 - [sub-task](#)
 - [transition](#)
- [Block a transition until all sub-tasks have certain fields populated](#)
 - [example](#)
 - [condition](#)
 - [validator](#)
 - [sub-task](#)
 - [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"](#)

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

- 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
- Automatically close resolved sub-tasks when parent issue is closed
 - example
 - post-function
 - sub-task
 - transition
 - outdated
- 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
- Close parent issue when all sub-tasks are closed
 - example
 - condition
 - validator
 - post-function
 - sub-task
 - transition
- Proceed with a task only when all sub-tasks are completed
 - example
 - condition
 - validator
 - sub-task
 - transition
- Prevent transitioning when there is a blocking issue
 - example
 - validator
 - issue-links
 - sub-task
 - transition
- Transition parent issue only when certain issue sub-task types are done
 - example
 - validator
 - sub-task
 - transition
- Enforce certain type of sub-tasks to be "Resolved" to allow executing a transition
 - example
 - validator
 - sub-task
 - transition
- Restrict parent issue from closing if it has sub-tasks that were created during a given parent issue status
 - example
 - validator
 - post-function
 - sub-task
 - transition