Add all assignees of certain sub-task types to a "Multi-User Picker" custom field

On this page

- Features used to implement the example
- Example: Add all assignees of certain sub-task types to a "Multi-User Picker" custom field
- Other examples of that functions
- Related Usage Examples

Features used to implement the example

- · Read fields from linked issues or sub-tasks
- Copy parsed text to a field

Example: Add all assignees of certain sub-task types to a "Multi-User Picker" custom field

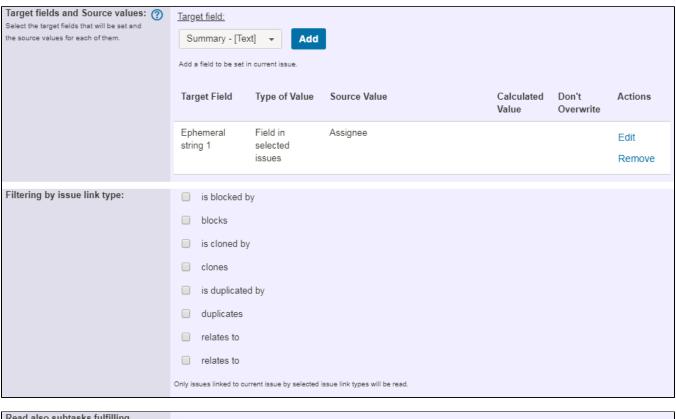
I'm trying to copy the assignee from only a specific type of sub-task to a multi-user field on the parent ticket. There could be any number of the specific type of sub-tasks (0-999) as well.

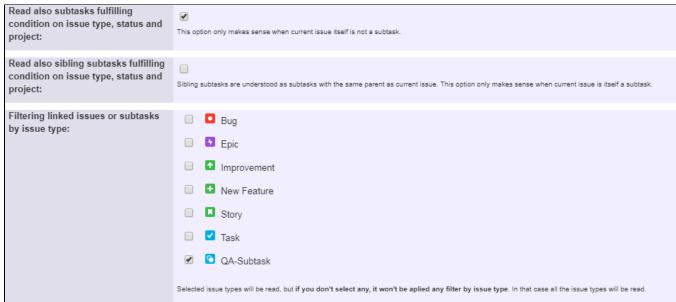
Can this function handle this logic?

I have an additional feature I'm trying to add that is somewhat related as well. For each of a certain type of sub-task, there is a start date field and an end date field. I want to copy the earliest of all the start dates and the latest of all the end dates to a start date field and end date field on the parent ticket.

I explain how to implement first functionality: Let's suppose we have a **Multi User Picker** custom field called "**Team**", and we want to select on that field all the **assignees** of sub-tasks of type "**QA Sub-task**".

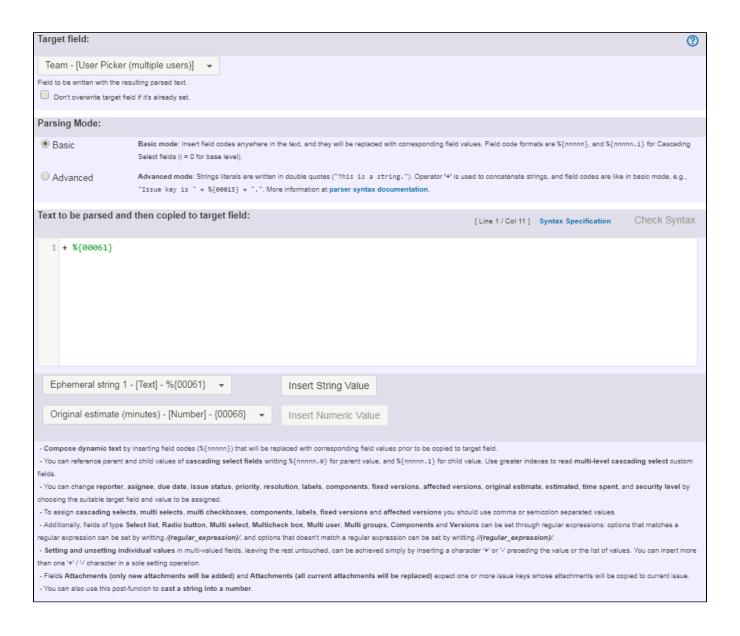
We use post-function Read fields from linked issues or sub-tasks with the following configuration:





Filtering linked issues or subtasks by status:	 → Open ♠ In Progress ♠ Reopened ♠ Resolved ♠ Closed ♠ To Do ♠ Done Selected statuses will be read, but if you don't select any, it won't be aplied any filter by status. In that case issues in any status will be read. 			
Linked issues or subtasks belong to:	any project current project any but current project			
Filtering by field values: Optional boolean expression that should be satisfied by linked issues and subtasks. (Syntax Specification)	Leave field empty for no filtering. 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, multivalued 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 ohecks whether Due Date is initialized. String Field Code Injector: Summary - [Text] - % {00000} Field Code for Current Issue Field Code Injector: Original estimate (minutes) - [Number] - {00068} Field Code for Current Issue Field Code for Linked Issues / Subtasks Example 1: {00012} <= ^{00012} = Field Code for Linked Issues / Subtasks Example 2: % (00074) ~ ^% (00074) AND ^% (00017) in ["Blocker", "Critical"] will require that linked issues and subtasks have Fixed versions contained in current issue's Fixed versions and Priority is Blocker or Critical.			
Read linked issues and subtasks recursively:	Issues and subtasks transitively linked will also be read, provided they fulfill stated filtering conditions. Issues are read recursively without depth limit, but each selected issue is read only once.			
Read also current issue:	Current issue will be included in the issue selection, i.e., current issue's field value will also be read.			
Additional options:	Update issue immediately after field writing. A specific entry will be created in issue history for this field writing.			

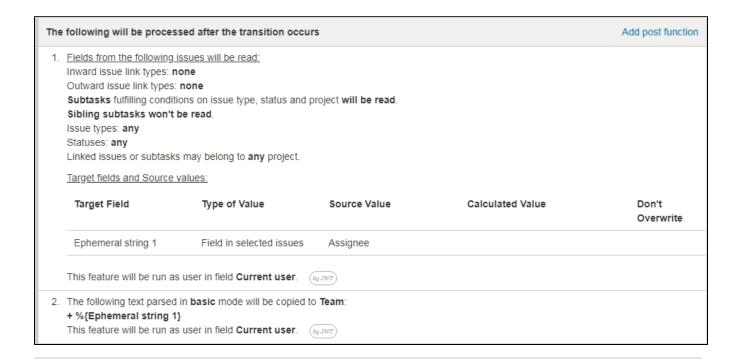
We use post-function **Copy a parsed text to a field** with the following configuration:



Note that:

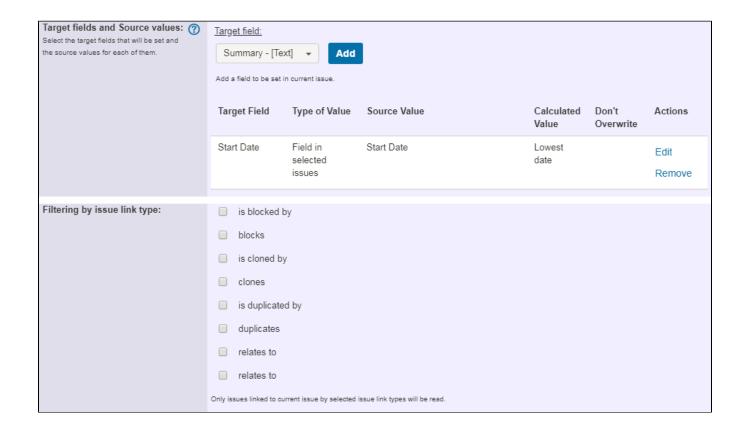
- We use prefix '+' in order to add all users contained in field "Ephemeral string 1". If we used prefix '-' instead, we would remove those users.
 If we didn't use any prefix, we simply would replace currently selected users in field "Team" by those contained in field "Ephemeral string 1"
- %{00061} is field code for virtual field "Ephemeral string 1"

Once configured, transition would look like this:



I explain now how to implement second functionality (third functionality is obvious once explained second one): Let's suppose we want to overwrite **Date e Picker** custom field "**Start Date**" with the **earliest value** of that field in all sub-tasks of type "**QA Sub-task**", or doing nothing in case parent issue contains an earlier value than its sub-tasks do.

We use post-function Read fields from linked issues or sub-tasks with the following configuration:



Read also subtasks fulfilling condition on issue type, status and project:	This option only makes sense when current issue itself is not a subtask.			
Read 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 ■ Epic ■ Improvement ■ New Feature ■ Story ■ Task ■ QA-Subtask			
	Selected issue types will be read, but if you don't select any, it won't be aplied any filter by issue type. In that case all the issue types will be read.			

Filtering linked issues or subtasks by status:	□ → Open			
	□ 🐧 In Progress			
	□ • Reopened			
	□ ♣ Resolved			
	□ ♣ Closed			
	□ 🕴 To Do			
	□ † Done			
	Selected statuses will be read, but if you don't select any, it won't be aplied any filter by status. In that case issues in any status will be read.			
Linked issues or subtasks belong to:	any project current project any but current project			
Filtering by field values:	1			
Optional boolean expression that should be satisfied by linked issues and subtasks. (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.			
	String Field Code Injector:			
	Summary - [Text] - %{00000} ▼			
	Field Code for Current Issue Field Code for Linked Issues / Subtasks			
	Numeric/Date Field Code Injector: Original estimate (minutes) - [Number] - {00068} ▼			
	Field Code for Current Issue Field Code for Linked Issues / Subtasks			
	Example 1: {00012} <= ^{00012} will require that linked issues and subtasks have Due Date equal or later than current issue's Due Date. Example 2: %{00074} ~ ^%{00074} AND ^%{00017} in ["Blocker", "Critical"] will require that linked issues and subtasks have Fixed versions contained in current issue's Fixed versions and Priority is Blocker or Critical.			
Read linked issues and subtasks				
recursively:	Issues and subtasks transitively linked will also be read, provided they fulfill stated filtering conditions. Issues are read recursively without depth limit, but each selected issue is read only once.			
Read also current issue:				
	Current issue will be included in the issue selection, i.e., current issue's field value will also be read.			
Additional options:	Update issue immediately after field writing. A specific entry will be created in issue history for this field writing.			

Once configured, transition will look like this:

The following will be processed after the transition occurs

Add post function

1. Fields from the following issues will be read:

Inward issue link types: none Outward issue link types: none

Subtasks fulfilling conditions on issue type, status and project will be read.

Sibling subtasks won't be read.

Issue types: any Statuses: anv

Linked issues or subtasks may belong to any project.

Current issue will also be read.

Target fields and Source values:

Target Field	Type of Value	Source Value	Calculated Value	Don't Overwrite
Start Date	Field in selected issues	Start Date	Lowest date	

This feature will be run as user in field Current user.



Other examples of that functions

Read fields from linked issues or sub-tasks

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: 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: Make an issue inherit highest priority among those of linked issues Page: Propagate highest priority from blocked issues to blocking issues Page: Sum sub-task's "Time Spent" (work logs) and add it to a certain linked issue

Copy parsed text to a field

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

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

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

Page: Make parent issue progress through its workflow

Related Usage Examples

- · Create a dynamic set of sub-tasks based on checkbox selection with unique summaries
 - example
 - o post-function
 - o custom-field
 - o sub-task
- Add and remove a single or a set of items from multi valued fields
 - o example
 - post-function
 - o custom-field
 - o issue-links
 - o sub-task
- Add all assignees of certain sub-task types to a "Multi-User Picker" custom field
 - o example
 - o post-function
 - o custom-field
- o sub-task Create a sub-task for each user selected in a Multi-User Picker
 - o example
 - post-function
 - o custom-field
 - o sub-task
- Update Cascading Select custom field with a value of the field in parent issue
 - o example
 - o post-function
 - o custom-field
 - o sub-task
- Validation on sibling sub-tasks depending on issue type and status
 - o example
 - o validator
 - sub-task o transition
- Restrict sub-task type creation depending on parent issue type
 - example
 - validator
 - o sub-task
- · Block a transition until all sub-tasks have certains fields populated
 - o example
 - o condition
 - o validator o sub-task

 - transition

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 adresses 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

- Restrict sub-task type creation depending on parent issue status
 - o example
 - o validator
 - o sub-task
- Require at least one sub-task in status "Resolved" or "Closed" when "Testing required" is selected in Check-Box custom field
 - example
 - validator
 - o sub-task
- Transition sub-tasks when parent is transitioned
 - example
 - post-function
 - o sub-task
 - transition
 - o outdated
- Transition only a sub-task among several ones
 - example
 - o post-function
 - o sub-task
 - o transition
 - outdated
- Moving sub-tasks to "Open" status when parent issue moves to "In Progress"
 - example
 - o post-function
 - o sub-task
 - o transition
 - o outdated
- Moving story to "Ready for QA" once all its sub-tasks are in

"Ready for QA" status

- examplepost-function
- o sub-task
- transition
- o outdated
- Change parent's status depending on sub-task's summary
 - example
 - post-function
 - o sub-task
 - transitionoutdated