Automatically become watcher of every issue blocking an issue assigned to you

On this page

Features used to implement the example

- **Write field on linked issues or sub-tasks**
- Virtual field "**New watchers**"

Example: Automatically become watcher of every issue blocking an issue assigned to you

We may be interested in automatically making an assignee watcher of every issue blocking the current issue, in order to react as soon as possible to any change in those issues. Post-function **Write field on linked issues or sub-tasks** is used to write the name or current issue's assignee into the virtual field "**New watchers**" of those issues linked with current issue using a "**is blocked by**" issue link. We will also become watcher of **every issue transitively blocking our issue**, i.e., any issue blocking another issue which is itself blocking current issue.

ISSUE A --blocks--> **ISSUE B** --blocks--> **ISSUE C** --blocks--> **ISSUE D,** in this example **issue A** is blocking directly **issue B** and transitively **issue C** and **D**. To implement this behavior we simply have to check option "**Write linked issues and sub-tasks recursively**" in post-function **Write field on linked issues or sub-tasks**.

We use post-function **Write field on linked issues or sub-tasks** to write field "**Assignee**" into virtual field "**New watchers**" in all blocking issues:

Source value that will be written into target field:

Select a source type:

(•) **field** in current issue ( ) **parsed text** (**basic** mode) ( ) **parsed text** (**advanced** mode) ( ) **math** or **date-time** expression

Assignee - [User]

The value of selected field in current issue will be used as source.

Target field that will be set in linked issues or subtasks:

New watchers - [Multi-user]

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

Filtering by issue link type:

- [x] is blocked by
- [ ] blocks
- [ ] is cloned by
- [ ] clones
- [ ] is duplicated by
- [ ] duplicates
- [ ] has Epic
- [ ] is Epic of
- [ ] is caused by
- [ ] causes
- [ ] 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:

- [ ] Epic
- [ ] Story
- [ ] Test Plan
- [ ] Bug
- [ ] New Feature
- [ ] Task
- [ ] Improvement
- [ ] QA Sub-task
- [ ] Sub-task

Selected issue types will be written, 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 written.

Filtering linked issues or subtasks by status:

- [ ] Open
- [ ] In Progress
- [ ] Reopened
- [ ] Resolved
- [ ] Closed
- [ ] To Do
- [ ] Done
- [ ] Acceptance
- [ ] Fail
- [ ] Pass
- [ ] Retest
- [ ] Active
- [ ] Inactive

Selected statuses will be written, 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 written.

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)

```
1
```

Leave field empty for no filtering. [Line 1 / Col 1]

Check Syntax

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}

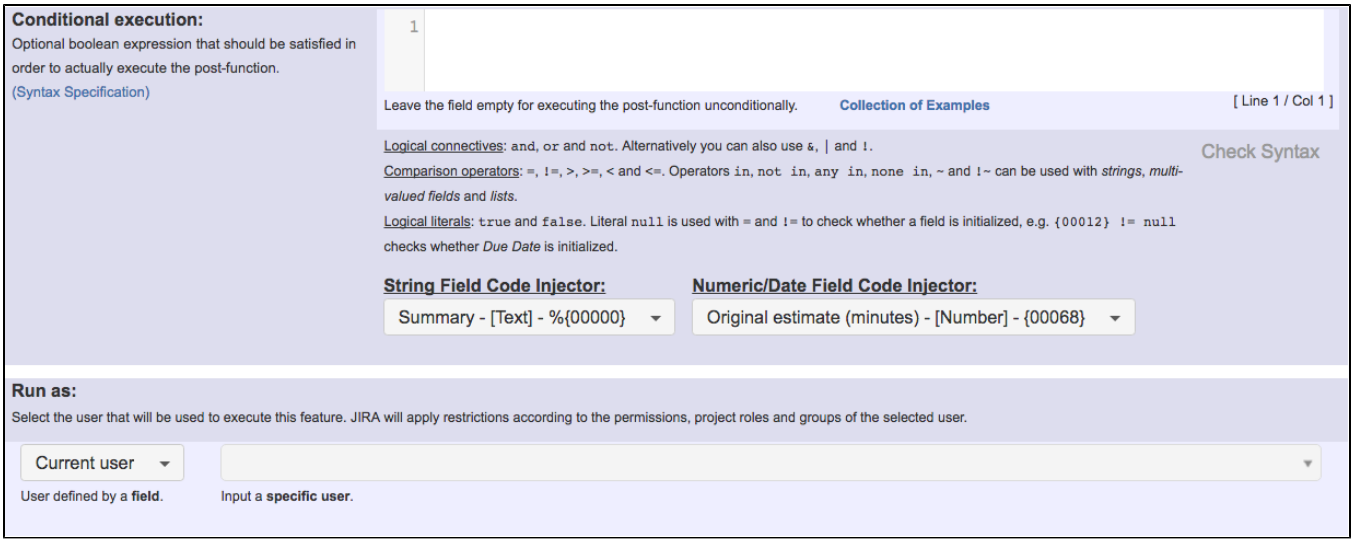
Field Code for **Current** Issue | Field Code for **Linked Issues / Subtasks**

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*.

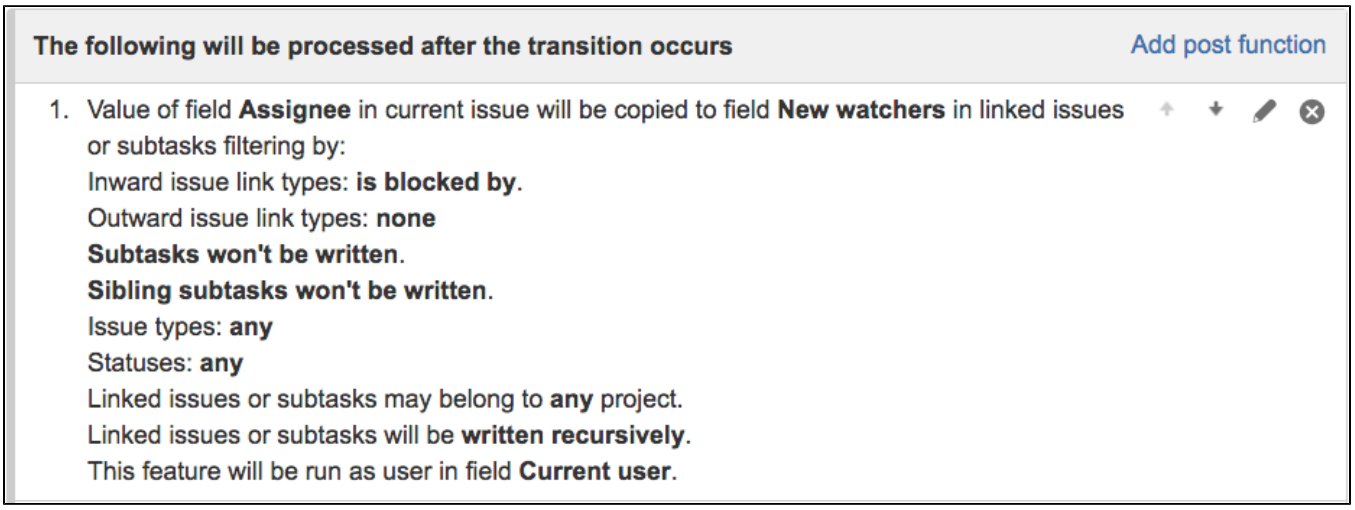
Write linked issues and subtasks recursively:

- [x]

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.



Once configured, the transition will look like this:



Other examples of that function

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

Related Usage Examples

- Validate only issue links created in transition screen
  - example
  - validator
  - issue-links
- Require issue link when resolving as duplicate
  - example
  - validator
  - issue-links
- Ensure that all issues linked with a certain issue link type have "Due Date" field set
  - example
  - validator
  - issue-links
- Block an epic's transition depending on linked issues status and due date
  - example
  - validator
  - issue-links
  - transition
- Add and remove a single or a set of items from multi valued fields

        - example
        - post-function
        - custom-field
        - issue-links
        - sub-task
- Writing a comment to blocked issues when blocking issues are resolved
    - example
    - post-function
    - issue-links
- Prevent issue from moving forward if it's dependent on non-accepted tickets
    - example
    - validator
    - issue-links
    - transition
- Prevent transitioning when there is a blocking issue
    - example
    - validator
    - issue-links
    - sub-task
    - transition
- Enforce linked issues in a specific project to be "Closed" before closing issue
    - example
    - validator
    - issue-links
    - transition
- Block or hide a transition for an issue depending on its issue links
    - example
    - validator
    - issue-links
    - transition
- Prevent issue from being "Closed" if blocking issues aren't yet closed
    - example
    - validator
    - issue-links
    - transition
- Block creation of issue type X if it has not been linked with link type Y to issue type Z on the "Create Issue" screen
    - example
    - validator
    - issue-links
- Prevent issue from being closed if it has links of type "is blocked by" to open issues
    - example
    - condition
    - validator
    - issue-links
    - transition
- Transition linked issues in currently active sprint
    - example
    - post-function
    - issue-links
    - transition
- Make an issue inherit highest priority among those of linked issues
    - example
    - post-function
    - issue-links