Make linked issues, sub-tasks and JQL selected issues progress through its workflows

On this page

- Features used to implement the example
- Example: Automatically move issues to "In Progress" status once all its blocker issues are moved to "Closed", "Resolved" or "Done" statuses
- · Other examples of that functions
- Related Usage Examples

Features used to implement the example

- · Write field on linked issues or sub-tasks
- Update issue fields
- Block or hide a transition for an issue depending on its issue links
- . Boolean Validator with math, date-time or text-string terms

Jira Workflow Toolbox provides features for making issues automatically progress through the workflow. Issues that can be transitioned this way are:

- Using post-function Write field on linked issues or sub-tasks
 - Linked issues
 - Subtasks
- Using post-function Update issue fields
 - Issues returned by a JQL query
 - o Issues returned by a Issue List expression

Making issues progress through its workflow using a post-function

An issue can transition other issues progress through its workflow by executing a post-function for writing the **name of target status** (e.g., "**Open**", "**In Progress**", "**Closed**", ...) into virtual fields "**Issue status**" or "**Issue status** (**delayed writing**)", or the name of a transition into virtual fields "**Execute transition**" and "**Execute transition** (**delayed execution**)" on other issues.

Care must be taken to write the name of the status or transition exactly as it is. Since version 6.x, Jira shows almost always the status name in upper case. To check the real name of the status, try to edit it at Administration > Issues > Statuses > Edit. Since version 2.1.20, JIRA Workflow Toolbox is case insensitive for issue status, so you can write the name of the status as you see it in UI.

Writing into virtual field "Issue status"

When the **name of a status** (e.g., "**Open**", "**In Progress**", "**Closed**", ...) is written into virtual field "**Issue status**" on another issue, that issue will progress through its workflow to that status, whenever there is a transition from current status to the new one, and conditions and validators in that transition are currently satisfied. **Post-functions in that transition will also be executed**, the same way they are when transition is triggered manually.

Obviously, if there isn't a transition from current status to target status in the workflow of the issue we want to move, it will remain in its current status.

Why is needed another virtual field: "Issue status (delayed writing)"?

"Issue status (delayed writing)" works like virtual field "Issue status", with the only difference that effective status change is carried out after transition execution in current issue (i.e., the issue executing the post-function) has been completed. On the other hand, when writing into "Issue status", status change is done immediately (i.e. within transition). This delayed way of writing is very useful when the issue you want to make progress through its workflow (sub-tasks, linked issues or JQL selected issues) is blocked by the status of current issue. The cause for this blockage might be a condition or a validator that depends on current issue's status, like Block or hide a transition for an issue depending on its issue links, Condition and validation on sub-tasks, Condition and validation based on JQL query and Boolean condition and validator with math. date-time or text-string terms.

Virtual fields "Execute transition" and "Execute transition (delayed execution)"

Since version 2.1.20, there are two additional virtual fields for transitioning issues: "Execute transition" and "Execute transition (delayed execution)". These fields behave similarly to "Issue status" and "Issue status (delayed writing)" respectively, with the only difference that they expect the name of a transition, instead of the name of a status.

These new fields are particularly useful if you have **global transitions in your workflow**, or if you have **more than one transition available for going to target status**. With them you specify which transition you want to be executed.

Hiding a transition to interactive Jira users

Sometimes we add transitions to the workflow, intended to be triggered exclusively by post-functions. In these cases we hide the transition to interactive (human) Jira users using condition "Transition is triggered by Jira Workflow Toolbox post-function".

Features used for transitioning issues

- Post-function Write field on linked issues or sub-tasks or Update issue fields, used to write into virtual fields "Issue status" or "Issue status (delayed writing)" the name of the target status we want to move issues to, or into virtual fields "Execute transition" or "Execute transition (delayed execution)" the name of the transition we want to execute.
- Virtual fields "Issue status" or "Issue status (delayed writing)": when the name of a status is written into any of these fields, the plugin
 tries to executed any transition available from current issue status to the written status. Conditions and validations in the transition should be
 satisfied. These fields are selected in parameter Target field of post-function.
- Virtual fields "Execute transition" or "Execute transition (delayed execution)": when the name of a transition is written into any of these
 fields, the plugin tries to executed a transition with the given name from current issue status. Conditions and validations in the transition
 should be satisfied. These fields are selected in parameter Target field of post-function.

Example: Automatically move issues to "In Progress" status once all its blocker issues are moved to "Closed", "Resolved" or "Done" statuses

In this example we edit 2 transitions in our workflows:

- 1. Transition "Start Progress": we add a validation into transition "Start Progress" in order to prevent its execution whenever there are blocking issues in statuses different from "Done", "Resolved" and "Closed".
- Transitions "Done", "Resolve Issue" and "Close Issue": we add a post-function for to moving blocked issues to "In Progress" status. The
 validator added previously in "Start Progress" transition will prevent the execution of this transition while there still are blocking issues not
 yet closed or resolved.

These 2 transitions might be in the same workflow, or belong to different workflows.

Configuration of transition "Start Progress"

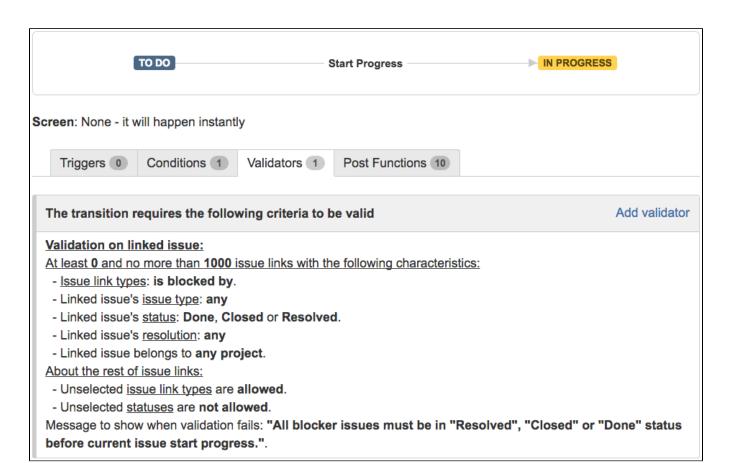
You can use any of these 2 validators in transition "Start progress" to block its execution until all blocking issues are in statuses "Resolved", "Closed" or "Done":

Validator Block or hide a transition for an issue depending on its issue links:

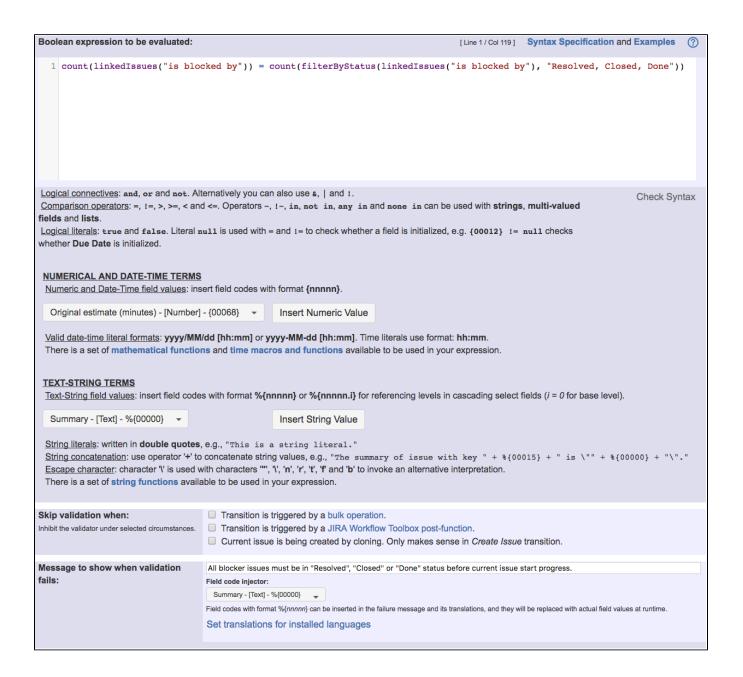
Issue link types:		?			
	blocks				
	is cloned by				
	clones				
	is duplicated by				
	duplicates				
	has Epic				
	is Epic of				
	is caused by				
	causes				
	relates to				
	relates to				
	Selected issue link types will be allowed. Nevertheless, if you didn't select any issue link type, and checked "Allow unselected issue link types", there won't be applied and by issue link type, i.e., every issue link type will be allowed.	y filter			
Issue types for linked issues:	□ S Epic				
	□ Story				
	□				
	□ ■ Bug				
	□ ► New Feature				
	□ ✓ Task				
	□				
	□ Sub-task				
	Linked issues of selected issue types will be allowed. Nevertheless, if you didn't select any issue type, and checked "Allow unselected issue types", there won't be applied filter by issue type, i.e., every issue type will be allowed.				

Statuses for linked issues:	□ → Open
	□ 🐧 In Progress
	□ ♣ Reopened
	□ → To Do
	✓ Å Done
	□ ♣ Acceptance
	□ ♣ Fail
	Pass
	Retest
	Active
	□ • Inactive
	Linked issues in selected statuses will be allowed. Nevertheless, if you didn't select any issue status, and checked "Allow unselected statuses", there won't be applied any filter by status, i.e., every issue status will be allowed.
Resolutions for linked issues:	UNRESOLVED, i.e. no resolution value
	□ Fixed
	□ Won't Fix
	Duplicate
	Incomplete
	Cannot Reproduce
	Linked issues with selected resolutions will be allowed. Nevertheless, if you didn't select any issue resolution, and checked "Allow unselected resolutions", there won't be applied any filter by resolution, i.e., every issue resolution will be allowed.
Linked issues must belong to:	
	current projectany but current project
	oprojects introduced:
	Introduce a comma separated list of project keys. If you write field codes (format %(nnnnn)) they will be replaced with its values. ignore links to issues in other projects



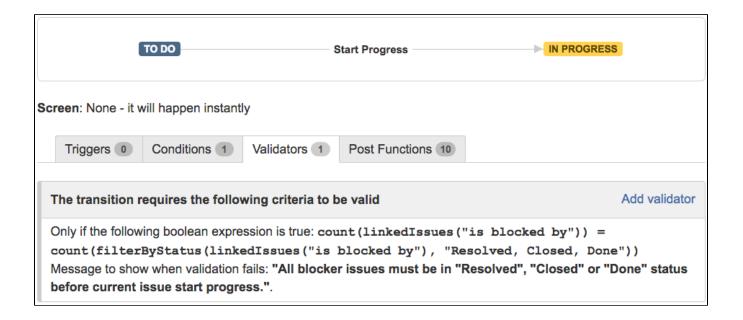


Validator Boolean Validator with math, date-time or text-string terms with the following configuration:



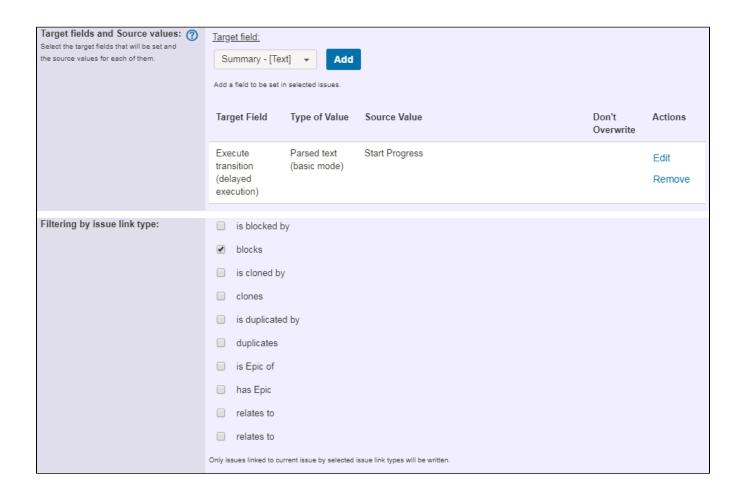
Boolean validation: count(linkedIssues("is blocked by")) = count(filterByStatus(linkedIssues("is blocked by"), "Resolved, Closed, Done"))

Once configured, transition "Start Progress" will look like this:

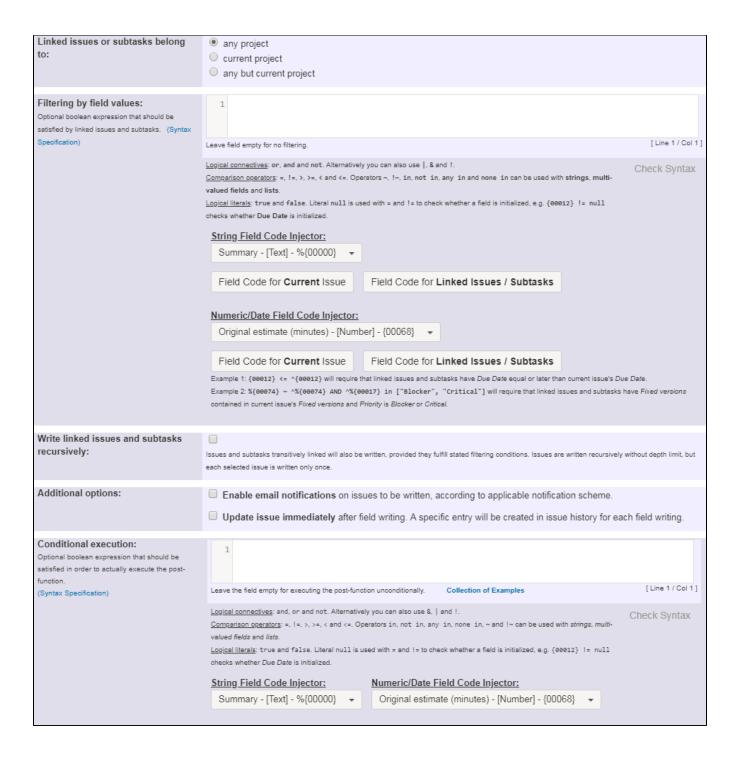


Configuration of transitions "Done", "Resolve Issue" and "Close Issue"

Use post-function Write field on linked issues or sub-tasks to write the name of the transition we want to execute into field "Issue status (delayed writing)" of blocked issue. We use a boolean expression in order to check whether the rest of blocking issues are already resolved or closed, and only when this boolean expression is satisfied the writing operation is actually done:



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 ■ Epic ■ Improvement ■ New Feature ■ Story ■ ✓ 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 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. 		



f) If you are using Jira 7.0 or higher with versions of Jira Workflow Toolbox older than 2.2.8, you should input the following boolean expression in parameter Filtering by field values:

count(filterByStatus(linkedIssues("is blocked by"), "Closed, Resolved, Done")) = count(linkedIssues("is blocked by")) - 1

This way it prevents the transition execution whenever the validation is not satisfied.

Once configured, transition transitions "Resolve Issue", "Close Issue" and "Done" will look like this:

The following will be processed after the transition occurs

Add post function

Fields in the following issues will be written:

Inward issue link types: none Outward issue link types: blocks. Subtasks won't be written.

Sibling subtasks won't be written.

Issue types: any

Statuses: To Do, Open and Reopened.

Linked issues or subtasks may belong to any project.

Target fields and Source values:

Target Field	Type of Value	Source Value	Don't Overwrite
Execute transition (delayed execution)	Parsed text (basic mode)	Start Progress	

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

Page: Transition sub-tasks when parent is transitioned

Update issue fields

Page: Add and remove a single or a set of items from multi valued fields

Page: Compose dynamic text by inserting field values in a text template

Page: Creating a Jira Service Desk internal comment

Page: Creating a Jira Service Desk internal comment on linked issues 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: Parse Email adresses to watchers list

Page: Set priority for issues that have been in a certain status for longer than 24 hours

Page: Transition linked issues in currently active sprint

Page: Transition only a sub-task among several ones

Page: Using project properties to calculate custom sequence numbers

Related Usage Examples

- · Prevent transitioning when there is a blocking issue
 - example
 - o validator
 - o issue-links o sub-task
 - o transition
- Make linked issues, sub-tasks and JQL selected issues progress through its workflows
 - o example
 - o condition
 - o validator
 - post-function
 - o issue-links
 - o sub-task
 - o transition
- Add and remove a single or a set of items from multi valued fields
 - o example
 - o post-function
 - o custom-field
 - o issue-links
 - o sub-task
- Sum sub-task's "Time Spent" (work logs) and add it to a certain linked issue
 - o example
 - o post-function
 - o issue-links
 - o sub-task
 - work-log
- Sum "Time Spent" in all sub-tasks of issues linked with issue link types "LinkA", "LinkB", "LinkC'
 - o example
 - o post-function
 - o issue-links
 - o sub-task o work-log
- Block an epic's transition depending on linked issues status and due date
 - o example
 - o validator
 - o issue-links
 - transition

Page: Writing a comment to blocked issues when blocking issues are resolved

Block or hide a transition for an issue depending on its issue links

Page: Automatically resolve an epic when all its stories are resolved Page: Make linked issues, sub-tasks and JQL selected issues progress through its workflows

Boolean Validator with math, date-time or text-string terms

Page: Block a transition until all sub-tasks have certains fields populated Page: Block an epic's transition depending on linked issues status and due date

Page: Block or hide a transition for an issue depending on its issue links Page: Block or unblock a transition after an issue rested a specific time in a status

Page: Block transition until all sub-tasks are in a specific status category

Page: Close parent issue when all sub-tasks are closed

Page: Enforce a field (Select List) to be set when another field (Radio

Button) has a certain value (works with any kind of field type)

Page: Ensure that all issues linked with a certain issue link type have "Due Date" field set

Page: If field A is populated then, field B must also be populated

Page: Limit issue creation per role and issue type

Page: Limit the number of hours a user can log per day

Page: Limit valid dates for work logs

Page: Make "Time Spent" field required when there is no time logged in the issue

Page: Make a custom field mandatory when priority is "Critical" or "Blocker" and issue type is "Incident"

Page: Make attachment mandatory depending on the value of certain custom field

Page: Make different fields mandatory depending on the value of a Select List custom field

Page: Make linked issues, sub-tasks and JQL selected issues progress through its workflows

Page: Make parent issue progress through its workflow

Page: Prevent issue creation if another issue with same field value already exists

Page: Reject duplicated file names in attachments

Page: Require at least one sub-task in status "Resolved" or "Closed"

when "Testing required" is selected in Check-Box custom field

Page: Require issue link when resolving as duplicate

Page: Restrict parent issue from closing if it has sub-tasks that were created during a given parent issue status

Page: Restrict sub-task type creation depending on parent issue status

Page: Restrict sub-task type creation depending on parent issue type

Page: Set a condition in a global transition which only applies in a certain status

Page: Validate a custom field "Story Points" has been given a value in Fibonacci sequence

Page: Validate compatible values selection among dependent custom fields

Page: Validate only issue links created in transition screen

Page: Validate that multi-user picker custom field A does not contain any user in multi-user picker custom field B

Page: Validation and condition based on time expressions

Page: Validation based on the value of a date type project property

Page: Validation on issue attachments

Page: Validation on MIME types of issue attachments

Page: Validation on sibling sub-tasks depending on issue type and status

Page: Validation on the value of a Cascading Select field

- Prevent issue from moving forward if it's dependent on nonaccepted tickets
 - o example
 - o validator
 - o issue-links
 - transition
- Enforce linked issues in a specific project to be "Closed" before closing issue
 - example
 - o validator
 - o issue-links
 - transition
- Block or hide a transition for an issue depending on its issue links
 - o example
 - validator
 - o issue-links
 - transition
- Prevent issue from being "Closed" if blocking issues aren't yet closed
 - o example
 - validator
 - o issue-links
 - transition
- Prevent issue from being closed if it has links of type "is

blocked by" to open issues

- o example
- o condition
- o validator
- o issue-links
- transition
- · Transition linked issues in currently active sprint
 - example
 - o post-function
 - issue-links
 - transition
- Validation on sibling sub-tasks depending on issue type and status
 - o example
 - validator
 - o sub-task
 - transition
- Block a transition until all sub-tasks have certains fields populated
 - example
 - o condition
 - o validator
 - o sub-task
 - transition
- Transition sub-tasks when parent is transitioned
 - o example
 - post-function
 - o sub-task
 - transitionoutdated