

Close parent issue when all sub-tasks are closed

On this page

- [Features used to implement the example](#)
- [Example: Close parent issue when all sub-tasks are closed](#)
- [Alternative implementation](#)
- [Other examples of that functions](#)
- [Related Usage Examples](#)

Features used to implement the example
















- [Condition and validation on sub-tasks](#)
- [Boolean condition and validator with math, date-time or text-string terms](#)
- [Copy parsed text to a field](#)

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

We want to automatically **close parent** issue when the **all subtasks** has been **closed**. We are going to describe two different solutions to implement this behavior

This solution consist on a validator in parent's workflow, and a post-function in sub-tasks workflow:

A **validation** in **parent issue workflow** to prevent parent issue from being closed manually whenever there still are open sub-tasks. To do it we use [C](#) **ondition and validation on sub-tasks** in transition "**Done**" or "**Close Issue**" on **parent's workflow** with the following configuration:

Subtask's issue types:	<div><div><input type="checkbox"/>  QA Sub-task</div><div><input type="checkbox"/>  Sub-task</div></div> <div>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 any filter by issue type, i.e., every issue type will be allowed.</div>
Subtask's statuses:	<div><div><input type="checkbox"/>  Open</div><div><input type="checkbox"/>  In Progress</div><div><input type="checkbox"/>  Reopened</div><div><input type="checkbox"/>  Resolved</div><div><input checked="" type="checkbox"/>  Closed</div><div><input type="checkbox"/>  To Do</div><div><input checked="" type="checkbox"/>  Done</div><div><input type="checkbox"/>  Acceptance</div><div><input type="checkbox"/>  Fail</div><div><input type="checkbox"/>  Pass</div><div><input type="checkbox"/>  Retest</div><div><input type="checkbox"/>  Active</div><div><input type="checkbox"/>  Inactive</div></div> <div>Subtasks 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.</div>
Subtask's resolutions:	<div><div><input type="checkbox"/> UNRESOLVED, i.e. no resolution value</div><div><input type="checkbox"/> Fixed</div><div><input type="checkbox"/> Won't Fix</div><div><input type="checkbox"/> Duplicate</div><div><input type="checkbox"/> Incomplete</div><div><input type="checkbox"/> Cannot Reproduce</div></div> <div>Subtasks 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 resolution will be allowed.</div>

Filtering by field values: Optional boolean expression that should be satisfied by subtasks. (Syntax Specification)	<div>1</div> <div>Leave field empty for no filtering.</div> <div>[Line 1 / Col 1]</div>
	<div> Logical connectives: or, and and not. Alternatively you can also use <code> </code>, <code>&</code> and <code>!</code>. Comparison operators: <code>=</code>, <code>!=</code>, <code>></code>, <code>>=</code>, <code><</code> and <code><=</code>. Operators <code>!~</code>, <code>in</code>, 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 <code>=</code> and <code>!=</code> to check whether a field is initialized, e.g. <code>{00012} != null</code> checks whether Due Date is initialized. </div> <div> Check Syntax </div> <div> String Field Code Injector: <div>Summary - [Text] - %{00000} ▾</div> <div>Field Code for Current Issue Field Code for Subtasks</div> </div> <div> Numeric/Date Field Code Injector: <div>Original estimate (minutes) - [Number] - {00068} ▾</div> <div>Field Code for Current Issue Field Code for Subtasks</div> </div> <div> <p>Example 1: <code>{00012} <= ^{00012}</code> will require that subtasks have <i>Due Date</i> equal or later than current issue's <i>Due Date</i>.</p> <p>Example 2: <code>!{00074} ~ ^{00074} AND ^{00017} in ["Blocker", "Critical"]</code> will require that subtasks have <i>Fixed versions</i> contained in current issue's <i>Fixed versions</i> and <i>Priority</i> is <i>Blocker</i> or <i>Critical</i>.</p> </div>
Minimum required number of subtasks:	<div>0</div> <div>Minimum number of subtasks required to satisfy selected filtering conditions (issue type, status, resolution and field values).</div>
Maximum allowed number of subtasks:	<div>1000</div> <div>Maximum number of subtasks allowed to satisfy selected filtering conditions (issue type, status, resolution and field values).</div>
Allow unselected issue types:	<div><input checked="" type="checkbox"/></div> <div>Subtasks in unselected issue types will be ignored, i.e., they will be allowed regardless of their status, resolution, field values and number. Nevertheless, if none of the issue types are selected, checking this option will make the condition behave as if you had selected every issue type.</div>
Allow unselected statuses:	<div><input type="checkbox"/></div> <div>Subtasks in unselected statuses will be ignored, i.e., they will be allowed regardless of their resolution, field values and number. Nevertheless, if none of the statuses are selected, checking this option will make the condition behave as if you had selected every status.</div>
Allow unselected resolutions:	<div><input checked="" type="checkbox"/></div> <div>Subtasks in unselected resolutions will be ignored, i.e., they will be allowed regardless of their field values and number. Nevertheless, if none of the resolutions are selected, checking this option will make the condition behave as if you had selected every resolution.</div>
Allow unsatisfied condition on field values:	<div><input type="checkbox"/></div> <div>Subtasks not satisfying filter by field values will be ignored, i.e., they will be allowed regardless of their number.</div>
Skip validation when: Inhibit the validator under selected circumstances.	<div> <input type="checkbox"/> Transition is triggered by a bulk operation. <input type="checkbox"/> Transition is triggered by a JIRA Workflow Toolbox post-function. <input type="checkbox"/> Current issue is being created by cloning. Only makes sense in <i>Create Issue</i> transition. </div>
Message to show when validation fails:	<div>All the sub-tasks must be in "Closed" or "Done" before closing parent issue.</div> <div> Field code injector: <div>Summary - [Text] - %{00000} ▾</div> </div> <div>Field codes with format <code>%{nnnn}</code> can be inserted in the failure message and its translations, and they will be replaced with actual field values at runtime.</div> <div>Set translations for installed languages</div>

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

TO DO

DONE

Screen: None - it will happen instantly

Triggers 0
Conditions 1
Validators 1
Post Functions 6

The transition requires the following criteria to be valid
Add validator

Validation on subtasks:

At least **0** and no more than **1000** subtasks with the following characteristics:

- Issue types: **any**
- Statuses: **Done** or **Closed**.
- Resolutions: **any**

About the rest of subtasks:

- Unselected statuses are **not allowed**.

Message to show when validation fails: **"All the sub-tasks must be in "Closed" or "Done" before closing parent issue."**

Alternatively, we can use [Boolean condition and validator with math, date-time or text-string terms](#) with the following boolean expression:

```
count(subtasks()) = count(filterByStatus(subtasks(), "Closed, Done"))
```

We add a **post-function** in **sub-task's workflow** for trying to close parent issue each time a sub-task is closed. To implement it we use post-function [Copy parsed text to a field](#) in transition **"Done"** or **"Close Issue"** on **sub-task's workflow** with the following configuration:

Target field: ?

Parent's issue status (delayed writing) - [Issue status]

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 1] [Syntax Specification](#) [Check Syntax](#)

1

Done

Once configured, transition **"Done"** or **"Close Issue"** in sub-task's workflow will look like this:

TO DO

Done

DONE

Screen: None - it will happen instantly

Triggers 0

Conditions 1

Validators 0

Post Functions 6

The following will be processed after the transition occurs

Add post function

1. The following text parsed in **basic** mode will be copied to **Parent's issue status (delayed writing)**:
Done
This feature will be run as user in field **Current user**.

Note: 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 **Conditional execution**:

```
count(filterByStatus(siblingSubtasks(), "Closed, Done")) = count(siblingSubtasks())
```

This way we are preventing transition from being executed whenever the validation is not satisfied.

Alternative implementation

Using only a post-function in sub-tasks workflow, the other implementation is usually preferred since it also prevents parent issue from being closed until all sub-tasks are closed, even if you try to do it by manually triggering transition **"Done"** or **"Close Issue"**. Anyway, sometimes you may want to allow manually overriding parent closing, even if there still are open sub-tasks.

This solution uses a post-function [Copy parsed text to a field](#) in transition **"Done"** or **"Close Issue"** in **sub-task's workflow** with the following configuration:

Target field:

Parent's issue status (delayed writing) - [Issue status]

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 1]

Syntax Specification

Check Syntax

1 Done

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)/`, and options that doesn't match 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(subtasks()) = count(filterByStatus(subtasks(), "Closed, Done"))`

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

Collection of Examples

[Line 1 / Col 70]

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.

Conditional execution: `count(subtasks()) = count(filterByStatus(subtasks(), "Closed, Done"))`

Once configured, transition "Close Issue" in sub-task's workflow will look like this:

TO DO

Done

DONE

Screen: None - it will happen instantly

Triggers 0

Conditions 1

Validators 0

Post Functions 6

The following will be processed after the transition occurs

Add post function

1. The following text parsed in **basic** mode will be copied to **Parent's issue status (delayed writing)**:
Done
Post-function will only be executed if the following boolean expression is satisfied: `count(subtasks()) = count(filterByStatus(subtasks(), "Closed, Done"))`
This feature will be run as user in field **Current user**.

Other examples of that functions

Condition and validation on sub-tasks

Page: Close parent issue when all sub-tasks are closed
Page: Enforce certain type of sub-tasks to be "Resolved" to allow executing a transition
Page: Make linked issues, sub-tasks and JQL selected issues progress through its workflows
Page: Make parent issue progress through its workflow
Page: Proceed with a task only when all sub-tasks are completed
Page: Transition parent issue only when certain issue sub-task types are done

Boolean condition and validator with math, date-time or text-string terms

Page: Block a transition until all sub-tasks have certain 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

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"
 - 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

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

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 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

- 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

Page: Block a transition until all sub-tasks have certain 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: Cascading select comparer
 Page: Check project property
 Page: Close parent issue when all sub-tasks are closed
 Page: Condition and validation based on JQL query
 Page: Condition and validation based on regular expression
 Page: Condition and validation on linked issues
 Page: Condition and validation on sub-tasks
 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: Except users in field
 Page: Expression Parser
 Page: If field A is populated then, field B must also be populated
 Page: JWT Release Notes 2.1.26

[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](#)

[Page: JWT Release Notes 2.1.29](#)
[Page: JWT Release Notes 2.2.8](#)
[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: Only users in a field](#)
[Page: Prevent issue creation if another issue with same field value already exists](#)
[Page: Project Properties](#)
[Page: Read a project property](#)
[Page: Read user property](#)
[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](#)
[Page: Virtual Fields](#)