

Limit valid dates for work logs

On this page

- [Features used to implement the example](#)
- [Example: Limit past dates for current work logs](#)
- [Other variation of the usage example](#)
- [Other examples of that function](#)
- [Related Usage Examples](#)

Features used to implement the example

- [Boolean validator with math, date-time or text-string terms](#)

Example: Limit past dates for current work logs

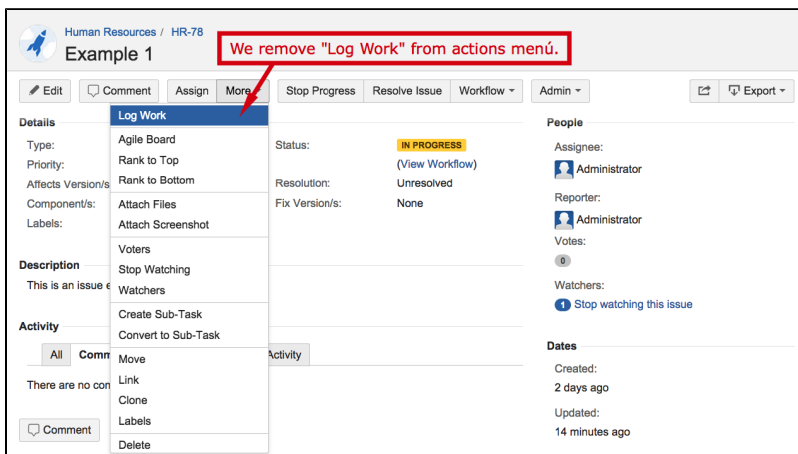
We will explain how to implement a validation for preventing work logs with a beginning date before a time limit in the past.

In particular, we will explain how to **avoid work logs where the associated date is earlier than 10 days in the past**, i.e. If we are on march 25th, today we only allow work logs with march 15th or later as a beginning date.

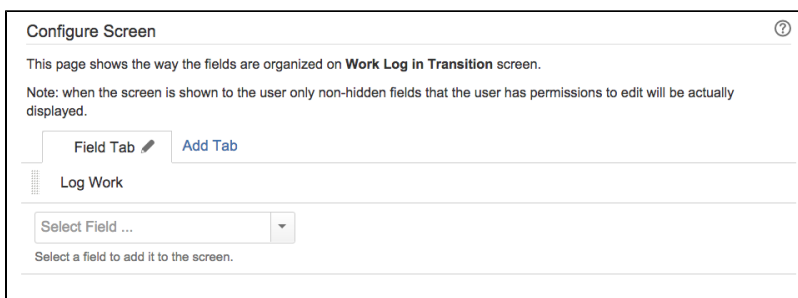
Enforce work logs through transitions

First we need to enforce users to log all the work using transitions in our workflows, instead of using **"Log Work"** operation at the issue screen. To do it we will:

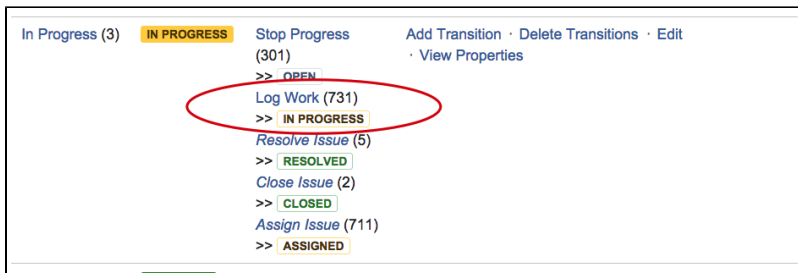
1. **Remove "Log Work" option from operation menu:** Administration > Add-ons > Manage add-ons > Filter by "System" > Issue Operations Plugin > Disable module "View Issue Ops Bar Work Link"



2. **Create a screen with only one field: "Log Work".** You can call it **"Work Log in Transition"**.



3. Add reflexive transitions called "Log Work" in the statuses of our workflow where we want users to be allowed to log work, and we associate them the screen "Work Log in Transition", created in the previous step. These transitions have the **same status as origin and destination**, leaving the issue in the same status, but showing the user a screen where they will be able to log work.



Typically you will add this transition to "In Progress" status, but you can add this transition to all statuses easily using a global reflexive transition.

You can use conditions or validations to limit who can execute these transition, and thus who can log work.

Now we only need to insert **Boolean Validator with math, date-time or text-string terms** in "Log Work" transitions using the following configuration:

Boolean expression to be evaluated:Syntax Specification

```
1 datePart({00057}, LOCAL) - datePart({00166}, LOCAL) <= 10 * {DAY}
```

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.

Boolean expression used is: `datePart({00057}, LOCAL) - datePart({00166}, LOCAL) <= 10 * {DAY}`

Note that:

- **{00057}** is code for numeric value of virtual field "Current date and time"
- **{00166}** is code for numeric value of virtual field "Date and time of work logged in transition"

Once all the post-functions have been inserted, transition "Log Work" will look like this:

IN PROGRESS

Log Work

IN PROGRESS

Screen: [Work Log in Transition](#)

Triggers 0

Conditions 0

Validators 1

Post Functions 9

The transition requires the following criteria to be valid
Add validator

Only if the following boolean expression is true: ***datePart({Current date and time}, LOCAL) - datePart({Date and time of work logged in transition}, LOCAL) <= 10 * {DAY}***
Message to show when validation fails: **"More that 10 days have passed since first work log."**

Other variation of the usage example

If you want to **limit valid work logs to dates within current week** you should use any of the following validations:

Weeks begin on Sundays

```
dayOfTheWeek({00057}, LOCAL) <= (datePart({00166}, LOCAL) - datePart({00057}, LOCAL)) / {DAY} AND (datePart({00166}, LOCAL) - datePart({00057}, LOCAL)) / {DAY} <= 7 - dayOfTheWeek({00057}, LOCAL)
```

Weeks begin on Mondays

```
dayOfTheWeek({00057}, LOCAL) != {SUNDAY} ? (2 - dayOfTheWeek({00057}, LOCAL) <= (datePart({00166}, LOCAL) - datePart({00057}, LOCAL)) / {DAY} AND (datePart({00166}, LOCAL) - datePart({00057}, LOCAL)) / {DAY} <= 8 - dayOfTheWeek({00057}, LOCAL)) : (-6 <= (datePart({00166}, LOCAL) - datePart({00057}, LOCAL)) / {DAY} AND (datePart({00166}, LOCAL) - datePart({00057}, LOCAL)) / {DAY} <= 0)
```

Other examples of that function

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](#)
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](#)

Related Usage Examples

- [Limit the number of hours a user can log per day](#)
 - [example](#)
 - [validator](#)
 - [post-function](#)
 - [work-log](#)
- [Make "Time Spent" field required when there is no time logged in the issue](#)
 - [example](#)
 - [validator](#)
 - [work-log](#)
- [Limit valid dates for work logs](#)
 - [example](#)
 - [validator](#)
 - [work-log](#)
- [Log absence time on another issue](#)
 - [example](#)
 - [post-function](#)
 - [work-log](#)
- [Set "Total time spent" to "Current date and time - date and time of last update"](#)
 - [example](#)
 - [post-function](#)
 - [work-log](#)
- [Sum "Time Spent" in all sub-tasks of issues linked with issue link types "LinkA", "LinkB", "LinkC"](#)
 - [example](#)
 - [post-function](#)
 - [issue-links](#)
 - [sub-task](#)
 - [work-log](#)

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

- Automatic work log with start and stop work transitions
 - example
 - post-function
 - work-log
- Automatically log work time when the user uses a "Stop Progress" transition
 - example
 - post-function
 - custom-field
 - work-log
- Sum sub-task's "Time Spent" (work logs) and add it to a certain linked issue
 - example
 - post-function
 - issue-links
 - sub-task
 - work-log