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

This function has been **renamed** with the **JWT 3.0** release.

Find the new documentation at:

Logical condition

Logical validator

On this page

Purpose

These condition and validator are among the most versatile tools in JIRA Workflow Toolbox, thanks to a **powerful parser for boolean expressions**. They allow writing **complex conditions based on the value of one or more field values**, using simple logic syntax with comparison operators and logical connectives **AND**, **OR** and **NOT**.

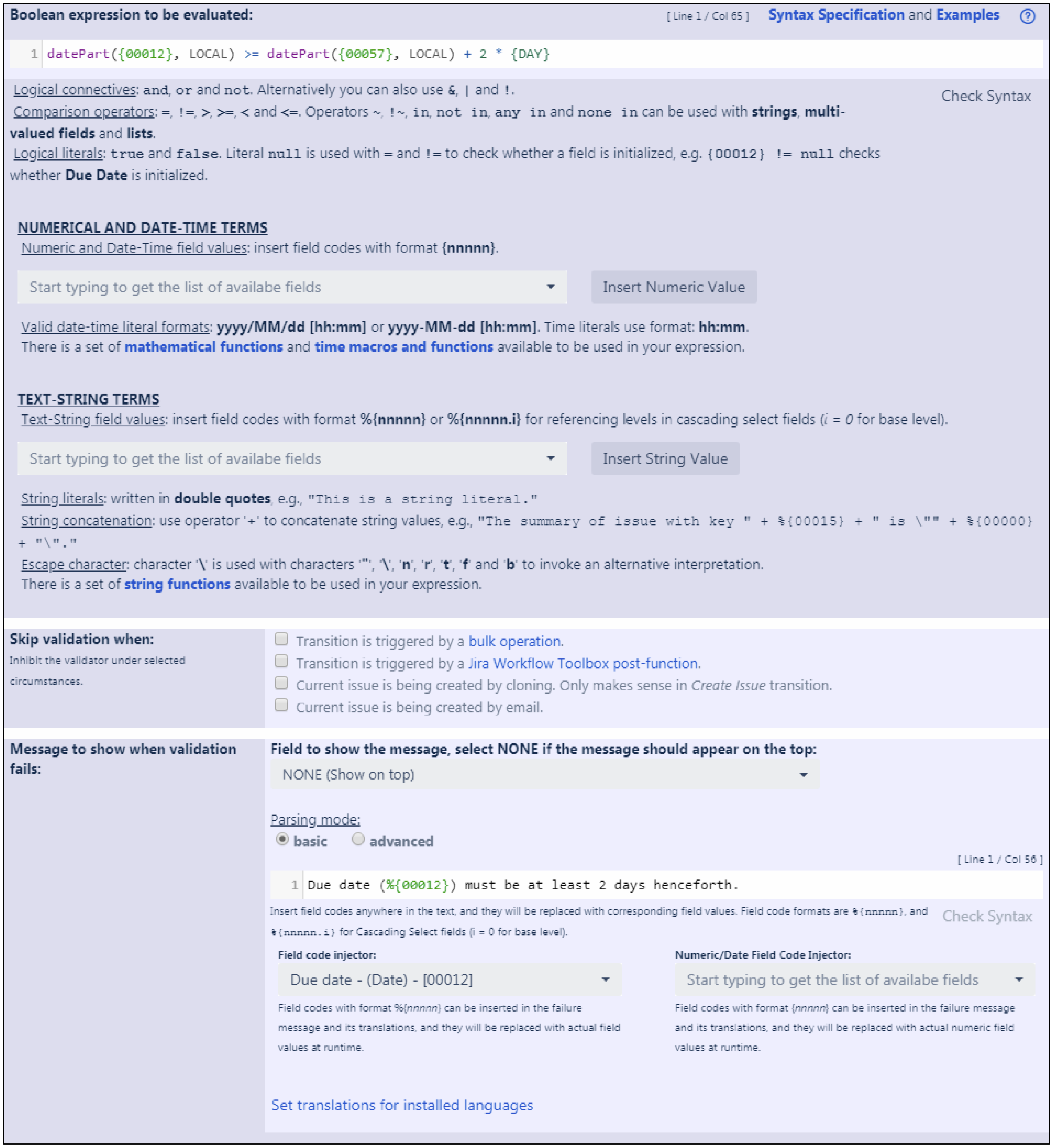
Logical expressions can contain **Numeric**, **Date-Time** and **Text-String** terms. Field values of most types can be referenced, including **Cascading Select** and **Multi-Cascading Select**. You can also introduce literal values (numbers, text strings, Dates-Time, logical values (true, false) and null).

A comprehensive set of **functions** and **macros** is available to operate on Date-Time, Number and Text-String terms.

The only difference between **Boolean condition with math, date-time or text-string terms** and **Boolean validator with math, date-time or text-string terms**, is that condition **hides the transition** where is inserted when the logical expression returns "false", and validator **shows a custom message**. In most cases the use of the validator is preferable than condition, since you can inform the user about the reason why he cannot execute the transition.

Example: Validate that there are still 2 days left to due date

In this example we implement a validation for checking that "**Due date**" has value at least **2 days greater than current date and time**.



Boolean expression to be evaluated:

[Line 1 / Col 65] **Syntax Specification** and **Examples**

```
datePart({00012}, LOCAL) >= datePart({00057}, LOCAL) + 2 * {DAY}
```

Logical connectives: `and`, `or` 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.

Check Syntax

NUMERICAL AND DATE-TIME TERMS

Numeric and Date-Time field values: insert field codes with format **{nnnnn}**.

Start typing to get the list of availabe fields ▾ | Insert Numeric Value

Valid date-time literal formats: **yyyy/MM/dd [hh:mm]** or **yyyy-MM-dd [hh:mm]**. Time literals use format: **hh:mm**.
There is a set of **mathematical functions** and **time macros and functions** available to be used in your expression.

TEXT-STRING TERMS

Text-String field values: insert field codes with format **%{nnnnn}** or **%{nnnnn.i}** for referencing levels in cascading select fields (*i* = *0* for base level).

Start typing to get the list of availabe fields ▾ | Insert String Value

String literals: written in **double quotes**, e.g., `"This is a string literal."`
String concatenation: use operator '+' to concatenate string values, e.g., `"The summary of issue with key " + %{00015} + " is \"" + %{00000} + "\"."`
Escape character: character '\' is used with characters '"', '\', '**n**', '**r**', '**t**', '**f**' and '**b**' to invoke an alternative interpretation.
There is a set of **string functions** available to be used in your expression.

Skip validation when:
Inhibit the validator under selected circumstances.

- □ Transition is triggered by a bulk operation.
- □ Transition is triggered by a Jira Workflow Toolbox post-function.
- □ Current issue is being created by cloning. Only makes sense in *Create Issue* transition.
- □ Current issue is being created by email.

Message to show when validation fails:

Field to show the message, select NONE if the message should appear on the top:
NONE (Show on top) ▾

Parsing mode:
● **basic** ○ **advanced**

[Line 1 / Col 56]

```
Due date (%{00012}) must be at least 2 days henceforth.
```

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).

Check Syntax

Field code injector:
Due date - (Date) - [00012] ▾
Field codes with format %{*nnnnn*} can be inserted in the failure message and its translations, and they will be replaced with actual field values at runtime.

Numeric/Date Field Code Injector:
Start typing to get the list of availabe fields ▾
Field codes with format {*nnnnn*} can be inserted in the failure message and its translations, and they will be replaced with actual numeric field values at runtime.

Set translations for installed languages

Note that:

- **{00012}** is code for numeric value of "**Due date**"
- **{00057}** is code for numeric value of "**Current date and time**"

Boolean condition with math, date-time or text-string terms and **Boolean validator with math, date-time or text-string terms** make use of the plugin's **Expression Parser**.

Collection of Boolean Expressions Examples

Usage Examples

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

Related Features

- **Condition and validation based on regular expression**