Logical condition

The logical condition is one of the **most powerful** and **versatile** conditions that can be used in JWT since it uses the **full potential** of the JWT expression parser functions.

This condition evaluates a logical expression that returns **`true`** or **`false`**. The transition will only be available, if the result is **`true`**.

Configuration

Expression

Since you only have a single parameter, an **expression**, you need to familiarize yourself with the Logical mode, which explains how to write logical expressions.

Additional examples

Parser expression	Description
`false`	Simply returns **false**. You can use this expression for "switching off" a specific post function.
`{parent.votes} > 5`	A numerical comparison which returns **true** if the parent issue has more than **5** votes.

`%{issue.assignee} = %{issue.project.lead} and %{issue.issueType} = "Bug"`
 	
`(%{issue.assignee} = %{issue.project.lead}) and (%{issue.issueType} = "Bug")`
	A logical conjunction which takes two comparisons as operands.
	
	It returns **true** when the **assignee** of the issue is the **project lead** and if it's a **Bug**.
	
	The second expression has the same meaning but due to use of brackets may be more readable.

`%{issue.assignee} = null`	Returns **true** if the issue does **not** have an assignee. This expression uses the **null** value as an operator.

`%{issue.priority} IN ["Blocker", "Critical"]`
 	
`%{issue.priority} = "Blocker" OR %{issue.priority} = "Critical"`
	Returns **true** if the Priority has the value "**Blocker**" or "**Critical**". The first expressions uses a list whereas the second one uses single comparisons connected via the logical operator **OR**.

`%{issue.issueType} = "Bug" IMPLIES %{issue.versions} != null`	Returns **true** if **Affects version/s** is set whenever the issue type equals "**Bug** ".

`%{issue.priority} IN ["Blocker", "Critical", "Major"] IMPLIES (%{issue.assignee} != null AND %{issue.duedate} != null)`	Returns **true** if **Priority** is "**Blocker**", "**Critical**" or "**Major**", the issue is assigned and **Due date** is set.
`%{issue.labels} ~ ["Blocker", "Critical", "Major"]`	Returns **true** if **Labels** (which is a field holding a TEXT LIST) contains "**Blocker**", "**Critical**" or "**Major**".

If you want to use this functionality in a **validator** instead, have a look at the Logical validator.

Use cases and examples

Use case	JWT feature	Workflow function	Parser functions	Label
Prevent a transition depending of the number of components		Logical validator Logical condition	numberOfSelectedItems()	
Prevent a closed issue from being reopened after resting 1 week closed		Logical validator Logical condition	isInRole()	
Prevent issue creation if description contains less than 100 characters		Logical validator	replaceAll() length()	
Evaluate Assets objects		Logical validator Logical condition	findPattern() findReplaceAll() replaceAll() toStringList()	
Block a transition until all sub-tasks have certain fields populated		Logical validator Logical condition	count() filterByPredicate() subtasks()	STAFF PICK
Validate a Select List (cascading) custom field		Logical validator Logical condition		
Prevent external users from creating issues		Logical validator	matches() toString() textOnStringList() toStringList() userDisplayName()	
Validate only issue links created in transition screen		Logical validator	count() transitionLinkedIssues() filterByProject() filterByIssueType()	

Hide transition to a previous status	Logical condition		
Prevent setting due dates outside business hours	Logical validator Logical condition	withinCalendar()	STAFF PICK
Set a condition in a global transition which only applies in a certain status	Logical condition		
Prevent the creation of sub-tasks unless the parent issue is in a certain status	Logical validator		
Halt a transition if an element is not contained in a list	Logical validator Logical condition	toStringList()	
Validation based on the value of a date type project property	Logical validator Logical condition	stringToDate() projectProperty()	
Restrict issue creation per issue type and project role	Logical validator	isInRole()	
Reject duplicated file names in attachments	Logical validator Logical condition	count() toStringList() distinct()	
Prevent issue creation with the same field value (Boolean)	Logical validator Validation based on JQL query	count() issuesFromJQL()	
Ensure that all issues linked with a certain issue link type have "Due Date" field set	Logical validator Logical condition	count() linkedIssues() fieldValue()	
Block an Epic's transition until all the issues under that Epic are resolved	Logical validator Logical condition	count() issuesUnderEpic() filterByResolution()	
Block an Epic's transition depending on linked issues status and due date	Logical validator Logical condition	count() filterByPredicate() linkedIssues()	
Make "Time spent" field required	Logical validator		
Ensure that an issue has at least one attachment	Logical validator Logical condition		STAFF PICK

If you still have questions, feel free to refer to our support team.