Validation based on the value of a date type project property

On this page

Configuration | Screenshots | Related use cases

Use case

Let us suppose you have a valid **project property** within the given context named **endDate**.

In the following use case the execution of the transition is only allowed, if the value in a **custom date or date-time field** is **greater or equal** to the **reference value** stored in the **project property**.

Configuration

1. Add the Logical validator to the desired workflow transition.

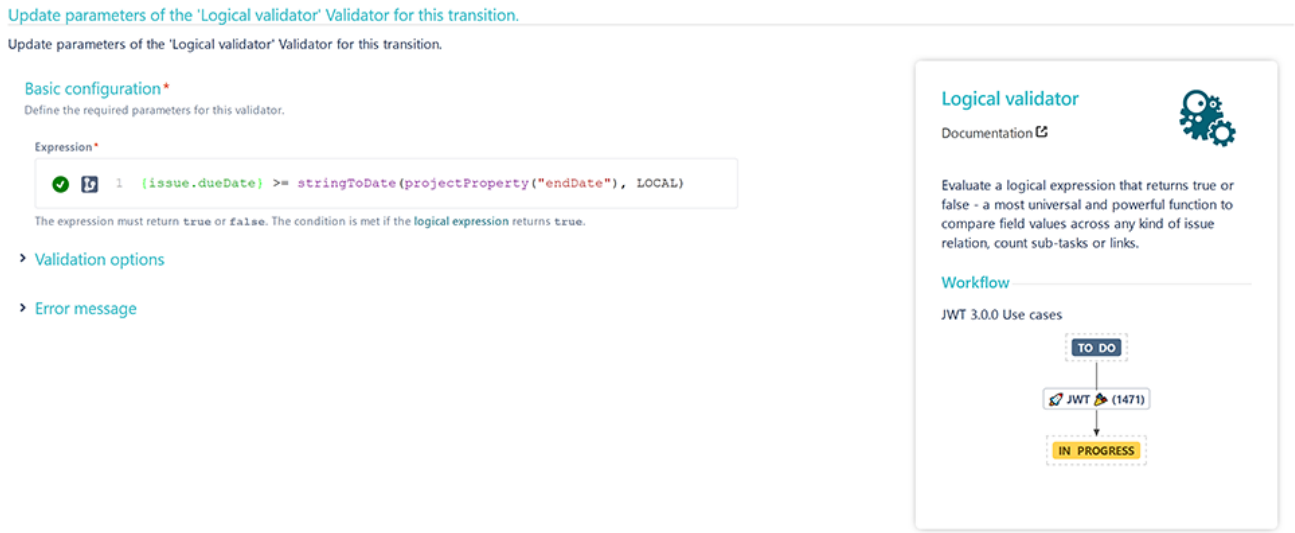
2. **Expression**

   Add the following expression:

   ```
   {issue.cfnnnnn} >= stringToDate(projectProperty("endDate"), LOCAL)
   ```

   Replace **cfnnnnn** with the actual field code of the custom date or date-time field.

Screenshots



Related use cases

Use case	JWT feature	Workflow function	Parser functions	Label
Block an Epic's transition depending on linked issues status and due date		Logical validator Logical condition	count() filterByPredicate() linkedIssues()	
Block an Epic's transition until all the issues under that Epic are resolved		Logical validator Logical condition	count() issuesUnderEpic() filterByResolution()	
Block a transition until all sub-tasks have certain fields populated		Logical validator Logical condition	count() filterByPredicate() subtasks()	STAFF PICK
Ensure that all issues linked with a certain issue link type have "Due Date" field set		Logical validator Logical condition	count() linkedIssues() fieldValue()	
Ensure that an issue has at least one attachment		Logical validator Logical condition		STAFF PICK
Evaluate Assets objects		Logical validator Logical condition	findPattern() findReplaceAll() replaceAll() toStringList()	

Halt a transition if an element is not contained in a list		Logical validator Logical condition	toStringList()
Hide transition to a previous status		Logical condition	
Make "Time spent" field required		Logical validator	
Prevent a closed issue from being reopened after resting 1 week closed		Logical validator Logical condition	isInRole()
Prevent a transition depending of the number of components		Logical validator Logical condition	numberOfSelectedItems()
Prevent external users from creating issues		Logical validator	matches() toString() textOnStringList() toStringList() userDisplayName()
Prevent issue creation if description contains less than 100 characters		Logical validator	replaceAll() length()
Prevent issue creation with the same field value (Boolean)		Logical validator Validation based on JQL query	count() issuesFromJQL()
Prevent setting due dates outside business hours		Logical validator Logical condition	withinCalendar()
Prevent the creation of sub-tasks unless the parent issue is in a certain status		Logical validator	
Reject duplicated file names in attachments		Logical validator Logical condition	count() toStringList() distinct()
Restrict issue creation per issue type and project role		Logical validator	isInRole()
Set a condition in a global transition which only applies in a certain status		Logical condition	
Validate a Select List (cascading) custom field		Logical validator Logical condition	
Validate only issue links created in transition screen		Logical validator	count() transitionLinkedIssues() filterByProject() filterByIssueType()

STAFF PICK

Validation based on the value of a date type project property		Logical validator	stringToDate()
		Logical condition	projectProperty()

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