Miscellaneous

On this page

Overview

The expression parser offers multiple functions that **cannot easily be categorized**.

A comprehensive list can be found below.

Available functions

Function	Input	Returned value
projectProperty(string **property_name**)	STRING	Returns a STRING with the value of project property with name **property_name** in current issue's project. `null` is returned if project property doesn't exist. Example: `projectProperty("maxNumberOfReopenings")` returns `"3"` , provided there is a string `{maxNumberOfReopenings=3}` in the description of current issue's project.
projectProperty(string **property_name**, string **project_key**)	STRING	Returns a STRING with the value of project property with name **property_name** in project with key **project_key**. `null` is returned if project property doesn't exist. Example: `projectProperty("maxNumberOfReopenings", "CRM")` returns `"3"` , provided there is a string `{maxNumberOfReopenings=3}` in the description of project with key **CRM**.
projectPropertyExists(string **property_name**)	STRING	Returns BOOLEAN **true** only if there is a project property with name **property_name** in current issue's project, i.e., if project's description contains a string like **{property_name=value}**. Example: `projectPropertyExists("maxNumberOfReopenings")` returns `true` only if there is a string like `{maxNumberOfReopenings=x}` in the description of current issue's project.
projectPropertyExists(string **property_name**, string **project_key**)	STRING	Returns BOOLEAN **true** only if there is a project property with name **property_name** in project with key **project_key**. **Example** `projectPropertyExists("maxNumberOfReopenings", "CRM")` returns `true` only if there is a string like `{maxNumberOfReopenings=x}` in the description of project with key `CRM`.
isAClone()		Returns BOOLEAN **true** only if current issue is a clone of another issue. An issue is a clone of another issue if it's being created by **Jira "Clone" operation**, or has **issue links of type "clones"**. This function if useful for bypassing validations in transition **Create Issue** when the issue is being created by a clone operation.
allComments()		Returns a STRING [] with all the comments in current issue in ascension order by creation date.

allComments(string **issue_keys**)	STRING	Returns a STRING [] with all the comments in issues with keys in **issue_keys**, in order of appearance in **issue_keys**, and by creation date in ascension order. Argument **issue_keys** is a comma separated list of issue keys. **Example** `allComments(%{...parentIssuekey})` returns parent issue's comments.
allComments(issue list **l**)	ISSUE []	Returns a STRING [] with all the comments in issues in **l**, in order of appearance in **l**, and by creation date in ascension order. Example: `allComments(subtasks())` returns all the comments in all the sub-tasks of current issue.
allCommenters()		Returns a STRING [] with the user names of comment authors and updaters in current issue, in ascension order by commenter's actuation time. The same user appears in the output as many times as the comments the user created and updated.
allCommentCreators()		Returns a STRING [] with the user names of comment creators in current issue, in ascension order by commenter's actuation time. A same user appears in the output as many times as comments has created. For anonimous comments an empty string (`""`) is returned.
allCommentCreators(string **issue_keys**)	STRING	Returns a STRING [] with the user names of comment creators in issues with keys in **issue_keys**, in order of appearance in **issue_keys**, and in ascension order by commenter's actuation time. A same user appears in the output as many times as comments has created. For anonymous comments an empty string (`""`) is returned.
allCommentCreators(string list **l**)	STRING []	Returns a STRING [] with the user names of comment creators of issues in **l**, in order of appearance in **l** , and in ascension order by commenter's actuation time. A same user appears in the output as many times as comments has created. For anonimous comments an empty string (`""`) is returned.
allCommenters(string **issue_keys**)	STRING	Returns a STRING [] with the user names of comment authors and updaters of issues with keys in **issue_keys**, in order of appearance in **issue_keys**, and in ascension order by commenter's actuation time. Argument **issue_keys** is a comma separated list of issue keys. **Example** `allComments(%{...parentIssuekey})` returns a string list with the user names of comment authors of parent issue.
allCommenters(issue list **l**)		Returns a STRING [] with the user names of comment authors and updaters of issues in **l** in ascension order by actuation time, in order of appearance in **l**, and in ascension order by commenter's actuation time. Example: `allCommenters(linkedIssues("is blocked by"))` returns a list with all the commenters and comment updaters for linked issues blocking current issue.

usersWhoTransitioned(string **origin_status**, string **destination_status**)	STRING	Returns a STRING [] with the names of the users who transitioned current issue from **origin_status** to **destination_status**, order ascending by time. An empty string as argument is interpreted as **any status**. **Example** `last(usersWhoTransitioned("Open", "In Progress"))` returns the name of the user who executed transition **"Start Progress"** more recently.
usersWhoTransitioned(string **origin_status**, string **destination_status**, string **issue_key**)	STRING	Returns a STRING [] with the names of the users who transitioned current issue from **origin_status** to **destination_status**, order ascending by time. An empty string as argument is interpreted as **any status**. **Example** `count(usersWhoTransitioned("Open", "In Progress", %{...parentIssuekey}))` returns the number of times transition **"Start Progress"** has been executed in parent issue.
timesOfTransition(string **origin_status**, string **destination_status**)	STRING	Returns a NUMBER [] with the times when current issue was transitioned from **origin_status** to **destination_status**, order ascending by time. An empty string as argument is interpreted as **any status**. **Example** `last(timesOfTransition("", "Resolved"))` returns the most recent time when the issue was resolved.
timesOfTransition(string **origin_status**, string **destination_status**, string **issue_key**)	STRING	Returns a NUMBER [] with the times when issue with key **issue_key** was transitioned from **origin_status** to **destination_status**, order ascending by time. An empty string as argument is interpreted as **any status**. **Example** `first(usersWhoTransitioned("Closed", "", %{...parentIssuekey}))` returns the first time when parent issue was reopened.
componentLeader(string **component_name**)	STRING	Returns the **user name** of the component lead with name **component_name** in current issue's project as STRING. This function also admits a comma separated list of components, and returns a comma separated list of **user names**. Output will contain repeated user names if a same user is leader of more than one component. **Example** `componentLeader(%{...components})` returns a comma separated list with the user names of the leaders of current issue's components.

componentLeader(string **component_name**, string **project_key**)	STRING	Returns a the **user name** of the component lead with name **component_name** in project with key **project_key** **as** STRING. This function also admits a comma separated list of components, and returns a comma separated list of **user names**. Output will contain repeated user names if a same user is leader of more than one component. **Example** `componentLeader("Web Portal", "CRM")` returns the user name of the leader of the component with name **Web Portal** in project with key **CRM**.
issueIDFromKey(string **issue_key**)	STRING	Returns a STRING of the **internal ID** of issue with key **issue_key**. This function also admits a comma separated list of issue keys, and returns a comma separated list of **internal IDs**. **Example** `issueIDFromKey("CRM-1")` returns `"10001"` .
issueKeyFromID(string **issue_ID**)	STRING	Returns a STRING of the **issue key** of issue with **internal ID issue_ID**. This function also admits a comma separated list of **issue IDs**, and returns a comma separated list of **issue keys**. **Example** `issueIDFromKey("10001")` returns `"CRM-1"` .
projectKeys()		Returns a STRING [] with all the *project keys* in the JIRA instance.
projectKeys(string **category)**	STRING	Returns a STRING [] with the *project keys* of those projects that belong to project category with name **category.**
projectName(string **project_key)**	STRING	Returns a STRING with the **name** of the project with key **project_key**.
projectCategory(string **project_key)**	STRING	Returns a STRING with the **category** of the project with key **project_key**.