Scripted JQL Quick Reference Guide

This is a reference guide for common scripted JQL functions in ScriptRunner for Jira Server and Data Center, as well as several use cases to show examples of working with scripted JQL functions.

JQL function	Example	Desciption
epicsOf	issueFunction in epicsOf (Subquery) Example: issueFunction in epicsOf ("resolution = unresolved")	This function allows you to query on epic links, such as finding all epics that have unresolved stories.
issuesinEpics	issueFunction in issuesInEpics (Subquery) Example: issueFunction in issuesInEpics ("project=JRA and status = 'in progress'")	This function lets you find all stories for open epics in a project, and then look specifically at the status of issues, such as *in progress.*
linkedIssuesOf	linkedIssuesOf(Subquery, [link name]) Example: linkedIssuesOf ("status="Open", "blocks") and resolution is empty	This function allows you to return linked issues, such as all unresolved issues that are blocked by open issues.
linkedIssues OfRecursive	issueFunction in linkedIssuesOfRecursive (Subquery) Example: issueFunction in linkedIssuesOfRecursive("issue =Jira-1")	This function allows you to return linked issues recursively, that is, to return all issues that are linked both directly and indirectly to the results of the initial query. For example, the example query returns all issues linked to *Jira-1*. Check out the scenarios to get a more detailed explanation of this function.
linkedIssuesOf AllRecursive	issueFunction in linkedIssuesOf AllRecursive Example: issueFunction in linkedIssues OfAllRecursive("issue =ADLEARN-711")	This function allows you to include the sub-task and epic links when you don't specify a link type. Otherwise, it functions identically to *linkedissuesofRecursive*
substasksOf	issueFunction in subtasksOf (Subquery) Example: issuefunction in subtasksOf ("issuetype = story and status != closed")	This function allows you to query on epic links, such as finding all epics that have unresolved stories.
parentsOf	issueFunction in parentsOf (Subquery) Example: issueFunction in parentsOf ("resolution is empty and assignee = currentUser()")	This function allows you to return the parents of issues that you specify in the subquery. For example, you can find the parent issues of all issues where you are the assignee of an open sub-task using the *resolution is empty* and *currentUser* function.
expression	issueFunction in expression (Subquery, expression) Example: issueFunction in expression (", "timespent > originalestimate")	This function lets you compare attributes of fields including system estimate and date fields, as well as any numeric, date, or *datetime* custom field. For example, you can find issues where work logged on the issue was more than originally estimated.
commented	issueFunction in commented (comment query) Example: issueFunction in commented ("visibility internal")	This function allows you to return issues based on a comment query. Your comment query can include a date query, username, roles, as well as comment types such as internal. For example, you can search for issues with only internally visible comments.
lastComment	issueFunction in lastComment (comment query) Example: issueFunction not in lastComment ("inRole Developers")	This function allows you to see the last comment for an issue based on your query. For example, you can find issues where the person last commenting is not in the *Developers* role. Or, remove the "not" and search for the last comment of someone in the *Developers* role.

Scenarios

Refer to this page to view the scripted JQL Functions in examples for context on their use. In the following scenarios, query values that you can swap for other items (such as project, assignee, etc.) shown in purple text. Values that we used for the specific examples are labeled in orange.

project = "Mobile"

Function and scenario	Syntax

epicsOf As a project manager, I need to keep up on the opened epics in my main project called Mobile App (Mobile). So, I want to create a filter using epicsOf that finds all of the epics with unresolved issues in a specific project.
 	
With the example function, I can look at all of the epics with unresolved issues in the Mobile project.
	`issueFunction in`
	`epicsOf("resolution=empty`
	`and project=Mobile")`

issuesinEpics Instead of looking at only unresolved epics, I decide to look at open epics and then issues in specific statuses to see how many open, in progress, or other issues statues are in the current project. Specifically, I want to see issues in the open epics that are also open, so we can gauge the work we have yet to start.
 	
Now, I can see all of my "to do" issues within epics in the specific Mobile project.
	`issueFunction in`
	`issuesInEpics("project=`
	`Mobile and status='to`
	`do'")`

linkedIssuesOf In another scenario, I may be a developer who wants to find unresolved issues that are blocked by issues in a "to do" status.
 	
This function helps me see issues that I have waiting on other issues that the team hasn't started yet. From this information, I can take action on those blockers.
	`issueFunction in`
	`linkedIssuesOf ("status=`
	`'to do'","blocks") AND`
	`resolution = empty`

linkedIssuesOfRecursive Perhaps as a developer, I also need to see how a bug issue affects a product in multiple ways, so I need all of the links, both direct and indirect, that trace back to that main bug issue. In this case, I need to use a recursive link.
 	
With this function, I can see all issues that link back to the main issue I selected, including blocks links, duplicates, clones, and more. This information gives me a good idea of the scope of a particular issue.
	`issueFunction in`
	`linkedIssuesOfRecursive`
	`("issue =adlearn-711")`

linkedIssuesOfAllRecursive And if I want to get detailed in this search, I can use a function to look at all recursive links that also include all epics and all subtasks for an issue as well.
 	
Including all items connected to the issue expands the search and gives an even greater scope of where an issue is linked in Jira.
	`issueFunction in`
	`linkedIssuesOfAllRecursive`
	`("issue =adlearn-711")`

substasksOf In another scenario, I want to see subtasks that are still open, but their parent issue is resolved, in the Mobile project. In this case, I can use subtasksOf to find all of the sub-tasks of issues that match my subquery.
 	
With this function, I get a list of all sub-tasks with parent issues that are marked resolved, but the sub-tasks are still open.
	`issueFunction in`
	`subtasksOf("project=Mobile`
	`and resolution is not`
	`empty") and status="to do"`

parentsOf If you work more in sub-tasks and want to see the parents of sub-task issues you specify in a subquery, use the ParentsOf scripted function. For example, maybe you want to see all of the parent issues where you are the assignee of an open sub-task.
 	
This function lets you see what issues may be waiting on you to finish up a sub-task!
	`issueFunction in`
	`parentsOf("resolution is`
	`empty and assignee =`
	`currentUser()")`

expression As a project manager, I want to find out how what issues we estimated poorly on—specifically those issues where we logged more work on the issue than was originally estimated. For that case, I can use the expression JQL function.
 	
With this information, I'll see issues where the team logged more work than we estimated, so we can investigate and improve estimation.
	`issueFunction in`
	`expression (","timespent"`
	`> originalestimate")`

commented As a service desk manager, I want to find all of the internal comments made during the last three weeks. To achieve this, I can use the commented JQL function.
 	
With this information, I can see a list of issues that contain internal comments from the last three weeks.
	`issuefunction in`
	`commented ("visibility`
	`internal after -3w")`

lastComment As a project manager, I want to find issues with comments where the last comment was made within the last hour. Perhaps I'm tracking down a recent correspondence and need to see what the latest is on the issues in my Mobile project within the last hour. For that, I can use lastComment along with specifying the project in the query.
 	
This JQL Function gives me all the issues in the mobile project that had comments made within the last hour.
	`project=mobile AND`
	`issueFunction in`
	`lastComment ("after -1h")`

![Adaptavist logo]