Examples of Issue List expressions

This page presents a collection of issue selection expressions valid for the **Expression Parser**. All these expressions return an **Issue List** type.

Linked Issues

Epic Link is also a kind of issue link. It's represented by the following 2 issue link types **`has Epic`** and,**`is Epic`** of which are used like this:

- **Epic issue `is Epic` of Story issue**
- **Story issue `has Epic` Epic issue**

On this page

Sub-tasks

All sub-tasks have one and only one parent issue and may have sibling sub-tasks, i.e., those issues sharing the same parent issue. Relation between **Epic** and **Stories** is not implemented through parent-child relation, but using issue links "**is Epic of**" and "**has Epic**", as explained above.

Filtering Issues Lists

Once we have an issue list, we can filter it by **issue type, status, status category, resolution, project, field values, cardinality** (i,e., number of appearances in the list), or using a **boolean predicate**, which is the most powerful method of issue filtering.

Obtaining Issue Lists using JQL Queries

Issue lists with big numbers of issues are temporarily stored in server's memory. For this reason, it's recommended not to build up big lists in your expressions, like retrieving all the issues in a project using function **`getIssuesFromProjects("PKEY")`**. Instead, it's better to use function **`issuesFromJQL("JQL_Query")`** using a **`JQL_Query`** that returns a small number of issues to work with.

Parameter **`JQL_Query`** is a **string** that represents a valid JQL Query. We typically build dynamic JQL queries inserting field values that we concatenate to string literals using + operator.

`linkedIssues()`	Issues linked to current issue through any issue link type, including **Epic Link**.
`linkedIssues("is blocked by")`	Issues linked to current one through **is blocked by** issue link type, i.e., **current issue `is blocked by` linked issue**.
`linkedIssues("is blocked by, is duplicated by, clones")`	Issues linked to current issue through **is blocked by, is duplicated by** and **blocks** issue link types.
`linkedIssues("has Epic")`	An issue list containing only the Epic of current issue. The returned list will contain 0 or 1 element, depending on whether current issue has an epic issue.
`linkedIssues("is Epic of")`	Issues current issue is epic of.
`linkedIssues("is Epic of", linkedIssues("has Epic")))`	Issues with the same epic as current issue. Current issue is also included in the issue list returned.

`linkedIssues("is Epic of", linkedIssues("has Epic")) EXCEPT issueKeysToIssueList (%{...issuekey})`	Issues with the same epic as current issue, excluding current issue. Current issue is not included in the issue list returned.
`linkedIssues() EXCEPT linkedIssues("is Epic of, has Epic")`	All the issues linked to current issue, except those linked through **has Epic** or **is Epic of** issue link types.
`transitionLinkedIssues("")`	Issues that have been linked to current issue in transition screen.
`transitionLinkedIssues ("blocks")`	Issues that have been linked to current issue in transition screen through **blocks** issue link type.
`transitivelyLinkedIssues ("is blocked by")`	Issues which are directly or indirectly blocking current issue.
`linkedIssues("", %{... parentIssueKey})`	Issues linked to parent of current issue. This expression only makes sense when current issue is a sub-task.
`linkedIssues("blocks", % {...parentIssueKey})`	Issues blocked byparentofcurrentissue. This expression only makes sense when current issue is a sub-task.
`subtasks()`	Sub-tasks of current issue.
`subtasks(%{... parentIssueKey})`	Sub-tasks of current sub-task's parent, including current sub-task.
`subtasks(linkedIssues("is blocked by"))`	Sub-tasks of all the issues linked to current issue using **is blocked by** issue link type.
`siblingSubtasks()`	Sub-tasks of current sub-task's parent, excluding current sub-task.
`filterByIssueType (linkedIssues(), "Improvement, New Feature")`	Issue types "**Improvement**" and "**New Feature**" linked to current issue.
`filterByStatus (filterByIssueType (linkedIssues(), "Improvement, New Feature"), "Open, In Progress")`	Issue types "**Improvement**" and "**New Feature**" linked to current issue, which are in statuses "**Open**" or "**In Progress**". In this example we are applying 2 filters, one after another, using function composition.
`filterByResolution (subtasks(), "Cannot Reproduce, Incomplete")`	Sub-tasks with resolutions "**Cannot Reproduce**" or "**Incomplete**".
`filterByResolution (subtasks(), "")`	Unresolved subtasks.
`filterByProject (linkedIssues(), "CRM, HR")`	Issue that belong to projects with keys "**CRM**" or "**HR**".
`filterByPredicate (linkedIssues(), ^%{... status} not in ["Closed", "Resolved"])`	**`%{...status}` = Issue status** Linked issues in statuses different from "**Closed**" and "**Resolved**". **`^%{...status} not in ["Closed", "Resolved"]`** is a boolean expression which should be satisfied in order to pass the filter. We add suffix ^ to field codes in order to reference the values of issues being filtered (i.e., linked issues), instead of current issues values.

On this page

`filterByPredicate (linkedIssues(), ^%{... status} = %{...status} AND toUpperCase(^%{... summary}) ~ toUpperCase ("important"))`
 	
`filterByPredicate (linkedIssues(), ^%{... status} = %{...status} AND ^%{...summary} ~~ "important")`
	Linked issues with the same status as current issue, which also contain the word "**important**" in their summary.
	
	We use function **`toUpperCase()`** in order to ignore the case when looking for the word "**important**" in issue summaries.

`issuesFromJQL("project = " + %{...project} + " AND issuetype in (Bug, Incident)")`	Issues with types "**Bug**" and "**Incident**" in the same project of current issue.
`issuesFromJQL("project = " + %{...project} + " AND issuetype = '" + %{... issuetype} + "'")`	Issues with same issue type and project as current issue. Note that we have written issue type in simple quotation marks. The reason is that issue type name may contain spaces.

On this page