linkedIssues()

This function returns the **list of issues linked** to the specified issue(s), including **Epic-Task** links.

Syntax
`linkedIssues() #Output: Issue list`

Examples

Parser expression	Description
`linkedIssues()`	This example returns an **issue list** with all issue **linked** to the **current issue**.
`distinct(linkedIssues() EXCEPT linkedIssues("has Epic, is Epic of"))`	This example returns an **issue list** of all the issues linked to current issue, excluding **Epic** issue links. To achieve this, the following functions are used: • distinct() • EXCEPT

Output

This function returns an ISSUE LIST

An issue appears in the output as many times as it is linked to the current issue.

Function distinct() can be used to remove duplicated issues.

Variant where you additionally can define **issue link types**.

Syntax
`linkedIssues(linkTypes) #Output: Issue list`

Examples

Parser expression	Description

`linkedIssues("blocks, clones")`	This example returns an **issue list** of all the issues linked with the **current issue** using issue link types **blocks** or **clones**.

Additional information

Parameters used in this function

Parameter	Input (data type)	Description
`linkTypes`	TEXT	Comma-separated list of issue link type names, or an empty text (**`""`**) for representing all issue link types, i.e., **`linkedIssues("")`** is equivalent to **`linkedIssues()`** .

Output

This function returns an ISSUE LIST

Variant where you additionally define **issues**.

Syntax
`linkedIssues(linkTypes, issueList) #Output: Issue list`

Examples

Parser expression	Description
`linkedIssues("", subtasks())`	This example returns an **issue list** of all issues linked to the current **issue's sub-tasks** using any issue link type. To achieve this, the following functions are used: • subtasks()
`linkedIssues("clones, duplicates", subtasks())`	This example returns an issue list of all issues linked to the current issue's sub-tasks using the link type **"clones"** or **"duplicates"**.

Additional information

Parameters used in this function

Parameter	Input (data type)	Description
`linkTypes`	TEXT	Comma-separated list of issue link type names, or an empty text (**`""`**) for representing all issue link types, i.e., **`linkedIssues("")`** is equivalent to **`linkedIssues()`** .

`issueList`	ISSUE LIST	Any given issue list. Usually this value is retrieved from a function (e.g. linkedIssues() or subtasks()). Duplicated issues are ignored.

Output

This function returns an ISSUE LIST

Variant where you define **issue keys** instead of **issues**.

Syntax

```
linkedIssues(linkTypes, issueKeys) #Output: Issue list
```

Examples

Parser expression	Description
`linkedIssues("", %{parent.key})`	This example returns an **issue list** of all issues linked to the current **issue's parent** using any issue link type.

Additional information

Parameters used in this function

Parameter	Input (data type)	Description
`linkTypes`	TEXT	Comma-separated list of issue link type names, or an empty text (**`""`**) for representing all issue link types, i.e., **`linkedIssues("")`** is equivalent to **`linkedIssues()`** .
`issueKeys`	TEXT	Comma-separated list of issue keys. Duplicated issue keys are discarded. Optionally you can use field codes returning issue keys.

Output

This function returns an ISSUE LIST

Use cases and examples

Use case	JWT feature	Workflow function	Field type	Automated action	Parser functions

Number of unresolved blocking issues		Number	count() filterByResolution() linkedIssues()
Highest ranked custom field value among all linked issues		Text	first() sort() fieldValue() linkedIssues()
Creation date of the linked epic		Date-time	first() fieldValue() linkedIssues()
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 depending on linked issues status and due date	Logical validator Logical condition		count() filterByPredicate() linkedIssues()