timeLogged()

This function returns the **sum** of all the **time logged** on **issues** in **milliseconds**.

Syntax
`timeLogged(issueList) #Output : Number`

Examples

Parser expression	Description
`timeLogged(subtasks())`	This example returns the **sum** of time logged in the current issue's **sub-tasks** in **milliseconds**.
`timeLogged(subtasks()) / {HOUR}`	This example returns the sum of time logged in the current issue's sub-tasks in **hours**. **`{HOUR}`** is a time macro used to convert the output to hours. subtasks() is a function to return all subtasks.
`round(timeLogged(subtasks()) / {HOUR})`	This example returns the sum of time logged in the current issue's sub-tasks in **full hours**. Learn more about numbers and mathematical functions.

Additional information

Parameters used in this function

Parameter	Input (data type)	Description
`issueList`	ISSUE LIST	A list of issue to be evaluated. Learn more about Lists.

Output

This function returns a NUMBER

Variant where you can additionally define the **time interval** when work was logged.

If any of the fields defined for the interval returns **null** the parameter will be ignored.

Syntax
`timeLogged(issueList, startDate, endDate) #Output: Number`

Examples

Parser expression	Description
`timeLogged(subtasks(), 2020/01/01, 2020/01/10) / {HOUR}`	This example returns the hours **logged** on **all subtasks** of the current issue between **January 1st** and January **10th of 2020**. **`{HOUR}`** is a time macro used to convert the output to hours. subtasks() is a function to return all subtasks.
`timeLogged(issuesUnderEpic(), datePart({system.currentDateTime}, LOCAL), addDays({system.currentDateTime}, LOCAL), 1, LOCAL))`	This example returns the sum of time logged **today** in issues under the current issue's Epic. The extract the date part of the current date (represented by its field code) the function datePart() is used.

Additional information

Parameters used in this function

Parameter	Input (data type)	Description
`issueList`	ISSUE LIST	A list of issues to be evaluated. Learn more about working with lists.
`startDate`	NUMBER	The parameter must be valid **timestamp**. Usually this value is retrieved from a field (e.g. due date, created date).
`endDate`	NUMBER	The parameter must be valid **timestamp**. Usually this value is retrieved from a field (e.g. due date, created date).

Output

This function returns a NUMBER

Variant where you can additionally limit the results to **certain users**.

Syntax
`timeLogged(issueList, user) #Output: Number`

Examples

Parser expression	Description
`timeLogged(linkedIssues(), %{issue.assignee}) / {HOUR}`	This example returns the sum of time logged on all **linked issues** by the current **assignee**. **{HOUR}** is a time macro used to convert the output to hours. linkedIssues() is a function to return all linked issues.
`timeLogged(subtasks(), %{issue.assignee}) / {HOUR}`	This example returns the sum of time logged on all **subtasks** of the current issue by the current **assignee**. **{HOUR}** is a time macro used to convert the output to hours. subtasks() is a function to return all subtasks.

Additional information

Parameters used in this function

Parameter	Input (data type)	Description
`issueList`	ISSUE LIST	A list of issues to be evaluated. Learn more about working with lists.
`user`	TEXT	Limits the results to certain users. Users can also be retrieved by using field codes. This parameter can contain a **single user name** (not to be confused with user's full name), or a comma separated **list of usernames**, **group** names or **project role** names.

Output

This function returns a NUMBER

Variant where you can additionally define the **time interval** when work was logged by certain **users**.

If any of the fields defined for the interval returns **null** the parameter will be ignored.

Syntax
`timeLogged(issueList, startDate, endDate, user) #Output: Number`

Examples

Parser expression	Description
`timeLogged(subtasks(), 2019/01/01, 2020/01/01, %{issue.assignee}) / {HOUR}`	This example returns the hours **logged** on **all subtasks** of the current issue during the year 2019 by the current **assignee**. **`{HOUR}`** is a time macro used to convert the output to hours. subtasks() is a function to return all subtasks.

Additional information

Parameters used in this function

Parameter	Input (data type)	Description
`issueList`	ISSUE LIST	A list of issues to be evaluated. Learn more about working with lists.
`startDate`	NUMBER	The parameter must be valid **timestamp**. Usually this value is retrieved from a field (e.g. due date, created date).
`endDate`	NUMBER	The parameter must be valid **timestamp**. Usually this value is retrieved from a field (e.g. due date, created date).
`user`	TEXT	Limits the results to certain users. Users can also be retrieved by using field codes. This parameter can contain a **single user name** (not to be confused with user's full name), or a comma separated **list of usernames**, **group** names or **project role** names.

Output

This function returns a NUMBER

Use cases and examples

Use case
No content found.