subtractDatesSkippingWeekends()

This functions **subtracts** two **timestamps** (the second date will be substracted from the first date) ignoring the weekends.

This way you can easily calculate the **elapsed working time** since **weekends** will be ignored.

Syntax
`subtractDatesSkippingWeekends(firstDate, dateToSubstract, timeZone) #Output: Number`

Examples

Parser expression	Description
`subtractDatesSkippingWeekends({issue.resolutionDate}, {issue.created}, LOCAL)`	This example calculates the **elapsed working time** from **issue creation** to **resolution** The result will return **milliseconds**.
`round(subtractDatesSkippingWeekends(({issue.resolutionDate}, {issue.created}, LOCAL)/{HOUR})`	This example returns the **hours** instead of **milliseconds.** Additional useful functions can be found here: Numbers

Additional information

Parameters used in this function

Parameter	Input (data type)	Description
`firstDate`	NUMBER	The parameter must be valid **timestamp**. Usually this value is retrieved from a field (e.g. due date, created date).
`dateToSubstract`	NUMBER	The parameter must be valid **timestamp**. Usually this value is retrieved from a field (e.g. due date, created date).
`timeZone`	TIMEZONE	The **time zone** used for the conversion.

Output

This function returns a NUMBER representing a timestamp.

Work days might depend on the **time zone** - it might be Sunday on the west coast of the US while at the same time it's already Monday in Australia.

Variant of the function where you can additionally define the **start** and the **end** of the **weekend**.

This function is useful when the non-working days **differ from the standard** (Saturday/Sunday).

Syntax
`subtractDatesSkippingWeekends(firstDate, dateToSubtract, timeZone, startOfWeekend, endOfWeekend) #Output: number`

Examples

Parser expression	Description
`subtractDatesSkippingWeekends({issue.resolutionDate}, {issue.created}, LOCAL, {FRIDAY}, {SATURDAY})`	This example calculates the **elapsed working time** from **issue creation** to **resolution** using an Arabic weekend. The result will return **milliseconds**.
`round(subtractDatesSkippingWeekends({issue.resolutionDate}, {issue.created}, LOCAL, {FRIDAY}, {SATURDAY})/{DAY})`	This example returns the **full** days instead of **milliseconds.** To achieve this the **`{DAY}`** time macro and the **`round()`** function are used**.** Additional useful functions can be found here: Numbers

Additional information

Parameters used in this function

Parameter	Input (data type)	Description
`firstDate`	NUMBER	The parameter must be valid **timestamp**. Usually this value is retrieved from a field (e.g. due date, created date).
`dateToSubtract`	NUMBER	The parameter must be valid **timestamp**. Usually this value is retrieved from a field (e.g. due date, created date).
`timeZone`	TIMEZONE	The **time zone** used for the conversion.
`startOfWeekend`	NUMBER	The parameter will take values in the format of **`{MONDAY},{TUESDAY}...{SUNDAY}`**
`endOfWeekend`	NUMBER	The parameter will take values in the format of **`{MONDAY},{TUESDAY}...{SUNDAY}`**

Output

This function returns a NUMBER representing a timestamp.

The output can be written into any number field.

Another very common use case is to use this function in a JWT calculated date-time fields. If you don't want to work with milliseconds check out all available time macros.

If you want to simply add days to a timestamp you might want to have a look at the function addDaysSkippingWeekends().

Not sure what this function returns? Simply test your expression on the expression parser test page.

Use cases and examples

Use case
No content found.