addTimeSkippingWeekends()

This function **adds** or **subtracts time** to or from **any given date**.

Weekends will **not be counted** as working days.

Syntax

```
addTimeSkippingWeekends(timestamp, timeToBeAdded, timeZone) #Output: Number
```

Examples

Parser expression	Description
`addTimeSkippingWeekends({issue.dueDate}, 1 * {MONTH} + 2 * {HOUR}, LOCAL)`	This example **adds 1 month and 2 hours** to the issue's due date. **Weekends** won't be taken into account.
`addTimeSkippingWeekends({issue.created}, 3 * {DAY} + 1 * {HOUR}, LOCAL)`	This example **adds 3 days and 1 hour** to the issue's creation date. **Weekends** won't be taken into account.
`addTimeSkippingWeekends({issue.dueDate}, - 1 * {YEAR}, LOCAL)`	This example **subtracts 1 year** from the issue's due date. **Weekends** won't be taken into account.

Additional information

Parameters used in this function

Parameter	Input (data type)	Description
`timestamp`	**NUMBER**	The parameter must be valid **timestamp**. Usually this value is retrieved from a field (e.g. due date, created date).
`timeToBeAdded`	**NUMBER**	The **offset**, supporting the usage of **time macros**. Learn more about time macros. **Negative** values are used to **subtract** time.
`timeZone`	**TIMEZONE**	The **time zone** used for the calculation.

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.

Output

This function returns a **NUMBER** representing a timestamp.

The output can be written into any Jira field of type **Date Picker** or **Date Time Picker**.

Another very common use case is to use this function in a [JWT calculated date-time field](#).

If you want to **convert** the number into a **text**, you might want to take a look at the [dateTimeToString()](#) function.

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

```
addTimeSkippingWeekends(timestamp, timeToBeAdded, timeZone, startOfWeekend, endOfWeekend) #Output: Number
```

Examples

Parser expression	Description
`addTimeSkippingWeekends({system.currentDateTime}, 12 * {HOUR}, LOCAL, {FRIDAY}, {SATURDAY})`	This example **adds 12 hours** to the current date and time. **Fridays** and **Saturdays** are not counted and will be skipped.
`addTimeSkippingWeekends({system.currentDateTime}, -2 * {DAY}, LOCAL, {SUNDAY}, {TUESDAY})`	This example **subtracts 2 days** from the current date and time. **Sundays**, **Mondays** and **Tuesdays** are not counted and will be skipped.

Additional information

Parameters used in this function

Parameter	Input (data type)	Description
`timestamp`	NUMBER	The parameter must be valid **timestamp**. Usually this value is retrieved from a [field](#) (e.g. due date, created date).
`timeToBeAdded`	NUMBER	The **offset**, supporting the usage of **time macros**. [Learn more about time macros.](#) **Negative** values are used to **subtract** time.
`timeZone`	TIMEZONE	The **time zone** used for the calculation.
`startOfWeekend`	NUMBER	Valid values are {**MONDAY**}, {**TUESDAY**} ... {**SUNDAY**}.
`endOfWeekend`	NUMBER	Valid values are {**MONDAY**}, {**TUESDAY**} ... {**SUNDAY**}.

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.

Output

This function returns a [NUMBER] representing a timestamp

The output can be written into any Jira field of type **Date Picker** or **Date Time Picker**.

Another very common use case is to use this function in a JWT calculated date-time field.

If you want to **convert** the number into a **text**, you might want to take a look at the dateTimeToString() function.

📁 Use cases and examples

Use case	JWT feature	Workflow function	Field type	Automated action	Parser functions
Predicted completion date of an epic	🕐		Date-time		addTimeSkippingWeekends()