BusinessHours Class

Use the `BusinessHours` methods to set the business hours at which your customer support team operates.

Namespace

System

BusinessHours Methods

The following are methods for `BusinessHours`. All methods are static.

- **add(businessHoursId, startDate, intervalMilliseconds)**
  Adds an interval of time from a start Datetime traversing business hours only. Returns the result Datetime in the local time zone.
- **addGmt(businessHoursId, startDate, intervalMilliseconds)**
  Adds an interval of milliseconds from a start Datetime traversing business hours only. Returns the result Datetime in GMT.
- **diff(businessHoursId, startDate, endDate)**
  Returns the difference in milliseconds between a start and end Datetime based on a specific set of business hours.
- **isWithin(businessHoursId, targetDate)**
  Returns `true` if the specified target date occurs within business hours. Holidays are included in the calculation.
- **nextStartDate(businessHoursId, targetDate)**
  Starting from the specified target date, returns the next date when business hours are open. If the specified target date falls within business hours, this target date is returned.

add(businessHoursId, startDate, intervalMilliseconds)

Adds an interval of time from a start Datetime traversing business hours only. Returns the result Datetime in the local time zone.

Signature

```
public static Datetime add(String businessHoursId, Datetime startDate, Long
intervalMilliseconds)
```

Parameters

businessHoursId

Type: String

startDate

Type: Datetime

intervalMilliseconds

Type: Long

Interval value should be provided in milliseconds, however time precision smaller than one minute is ignored.

addGmt(businessHoursId, startDate, intervalMilliseconds)

Adds an interval of milliseconds from a start Datetime traversing business hours only. Returns the result Datetime in GMT.

Signature

```
public static Datetime addGmt(String businessHoursId, Datetime startDate, Long intervalMilliseconds)
```

Parameters

businessHoursId

Type: String

startDate

Type: Datetime

intervalMilliseconds

Type: Long

Return Value

Type: Datetime

diff(businessHoursId, startDate, endDate)

Returns the difference in milliseconds between a start and end Datetime based on a specific set of business hours.

Signature

```
public static Long diff(String businessHoursId, Datetime startDate, Datetime endDate)
```

Parameters

businessHoursId

Type: String

startDate

Type: Datetime

endDate

Type: Datetime

Return Value

Type: Long

isWithin(businessHoursId, targetDate)

Returns `true` if the specified target date occurs within business hours. Holidays are included in the calculation.

Signature

```
public static Boolean isWithin(String businessHoursId, Datetime targetDate)
```

Parameters

businessHoursId

Type: String

The date to verify.

Return Value

Type: Boolean

Example

The following example finds whether a given time is within the default business hours.

```
// Get the default business hours
BusinessHours bh = [SELECT Id FROM BusinessHours WHERE IsDefault=true];

// Create Datetime on May 28, 2013 at 1:06:08 AM in the local timezone.
Datetime targetTime = Datetime.newInstance(2013, 5, 28, 1, 6, 8);

// Find whether the time is within the default business hours
Boolean isWithin= BusinessHours.isWithin(bh.id, targetTime);
```

nextStartDate(businessHoursId, targetDate)

Starting from the specified target date, returns the next date when business hours are open. If the specified target date falls within business hours, this target date is returned.

Signature

```
public static Datetime nextStartDate(String businessHoursId, Datetime targetDate)
```

Parameters

businessHoursId

Type: String

The business hours ID.

targetDate

Type: Datetime

The date used as a start date to obtain the next date.

Return Value

Type: Datetime

Example

The following example finds the next date starting from the target date when business hours reopens. If the target date is within the given business hours, the target date is returned. The returned time is in the local time zone.

```
// Get the default business hours
BusinessHours bh = [SELECT Id FROM BusinessHours WHERE IsDefault=true];

// Create Datetime on May 28, 2013 at 1:06:08 AM in the local timezone.
Datetime targetTime = Datetime.newInstance(2013, 5, 28, 1, 6, 8);
// Starting from the targetTime, find the next date when business hours reopens. Return th

// if it is within the business hours. The returned time will be in the local time zone
Datetime nextStart = BusinessHours.nextStartDate(bh.id, targetTime);
```

DEVELOPER CENTERS

- Heroku
- MuleSoft
- Tableau
- Commerce Cloud
- Lightning Design System
- Einstein
- Quip

POPULAR RESOURCES

- Documentation
- Component Library
- APIs
- Trailhead
- Sample Apps
- Podcasts
- AppExchange

COMMUNITY

- Trailblazer Community
- Events and Calendar
- Partner Community
- Blog
- Salesforce Admins
- Salesforce Architects