@zohodesk/datetimejs

Please check out this docs to know more about time zones and daylight savings.
Time zones supported in this library are in this link.

Prerequisite configurations for the library:

1. In order to copy the timezone files to the build-system of the project, add the below script in package.json.

```
/* Copies the timezone files (from the library) to the build path, so that the timezone file can be server from the static server */
react-cli copyTimezones <build-path>
```

Check out the sample here.

2. We have to fetch the timezone data before the app renders.

```
import dt from '@zohodesk/datetimejs';
/* sets the domain to serve the timezone file and the default timezone. Once this is set, other methods will work on this timezone by default */
dt.tz.set({
   domain: <static-domain-path>,
 _default: <default-timezone>
});

/* fetches the timezone file */
let promise = dt.tz.get(<timezone>);
promise.then(() => {
   /* Since the entire apps runs on that timezone, we should only render app, once the timezone file is fetched */
})
```

Check out the sample here.

Timezone Conversion methods:

➤ For the below methods, if the timezone is not passed to the function, it will work

on the '_default' timezone provided in the dt.tz.set() method.

- ➤ The timezone data should be fetched (using dt.tz.get() method) before we pass a particular timezone to the methods.
- ➤ The date-time passed to these methods should only be in one of the three formats:
  - UTC timestamp / UTC milliseconds
  - ISO format
  - Dates should be in format 'YYYY-MM-DD'

data(timezone)

returns the timezone data (the data is in the same format from moment. Refer to moment docs for how the data is represented). The DateTime, DateWidget component makes use of moment's timezone data to display the date time in the corresponding timezone. Check data for 'Asia/Kolkata' timezone here.

```
1  import React from 'react';
2  import ReactDOM from 'react-dom';
3  import dt from '@zohodesk/datetimejs';
4  import {DateWidget} from '@zohodesk/components';
5
6  function Widget(){
7  /* gets the timezone data for 'Asia/Kolkata' */
8          return(
9          <DateWidget
10                 timeZone={dt.tz.data()}
11                 value="2020-07-07T09:19:09.629Z"
12         />
13  );
14 }
15 const timezone='Asia/Kolkata';
16 dt.tz.set({
17         domain: 'https://js.zohostatic.com/js/timezones',
18         _default: timezone
19 });
20
21 dt.tz.get(timezone).then(() => {
22         ReactDOM.render(
23                 <Widget />,
24                 document.getElementById('react')
25         )
```

```
26 });
```

utcToTz(date, timezone)
Converts the UTC date to another timezone and returns the new date in milliseconds.

```
1  import dt from '@zohodesk/datetimejs';
2  const duedate = dt.toDate(dt.tz.utcToTz("2020-07-07T09:35:35.229Z"));
3  /* output:
4  Tue Jul 07 2020 15:05:35 GMT+0530 (India Standard Time)
5  */
```

tzToUtc(date, timezone)
Converts a date from a different timezone to UTC and returns the new date in milliseconds

```
1  import dt from '@zohodesk/datetimejs';
2  const year = 2019, month = 11, day = 10;
3  const dueInUTC = dt.toDate(dt.tz.tzToUtc(Date.UTC(year, month, day)));
4  /* output:
5  Tue Dec 10 2019 18:30:00 GMT+0530 (India Standard Time)
6  */
```

offset(date, timezone)
returns the corresponding offset for the date (corrected with daylight savings)

```
1  import dt from '@zohodesk/datetimejs';
2  dt.tz.set({
3          domain: 'https://js.zohostatic.com/js/timezones',
4          _default: 'Asia/Kolkata'
5  });
6  dt.tz.offset("2020-07-07T09:35:35.229Z") // -330 for 'Asia/Kolkata'
7  dt.tz.get('Asia/Shanghai').then(() => {
8          dt.tz.offset() // -480 for 'Asia/Shanghai' for current time
9  });
```

Please note that offsets are subject to daylight savings and change with different dates

for some countries. So, we need to pass the date for which the offset is computed. If not passed, the offset is computed for current date.

Other Methods:

browserOffset
returns the browser/system timezone offset of the current time.

```
1  import dt from '@zohodesk/datetimejs';
2  const offset = dt.browserOffset(); // -330 for IST
```

millis(date)
returns the UNIX timestamp of date in milliseconds (with system timezone)

```
1  import dt from '@zohodesk/datetimejs';
2  const millis = dt.millis('2019/11/20'); //1574188200000
3  const currMills = dt.millis() // millis of current time
```

The argument date is any string supported by the browser Date object.
Please note that the millis always represents time in +0:00 offset.

ISO(date)
returns the ISOString of the date (with system timezone)

```
1  import dt from '@zohodesk/datetimejs';
2  const ISO = dt.ISO('2019/11/20'); // "2019-11-19T18:30:00.000Z"
3  const currISO = dt.ISO(); // current time in ISO format
```

The argument date is any string supported by the browser Date object.
Please note that the ISO string always represents time in +0:00 offset.

toDate(date)
returns a new Date in GMT+0:00 of the argument date (with system timezone) (ie.) the function removes the browserOffset.

```
import dt from '@zohodesk/datetimejs';

const due = dt.toDate('2019/11/20');
/*
output:
Tue Nov 19 2019 18:30:00 GMT+0530 (India Standard Time)
*/

const date = due.getDay(); //19
/* can chain as a regular Date browser method */
```
