2 minutes

L'API des objets Date

Lire des dates

Il existe de très nombreuses méthodes permettant d'obtenir des informations depuis un objet `Date`.

Toutes ces instances sont sur le prototype des objets `Date` dont ils héritent (nous étudierons plus tard en détails les prototypes et l'héritage).

Voici toutes les méthodes, celles qui ne sont pas listées sont dépréciées ou non standardisées et ne doivent plus être utilisées.

Faites attention car vous pourriez avoir des bugs en utilisant des méthodes non standardisées (qui ont des implémentations différentes suivant les navigateurs).

Les méthodes utilisant l'heure du système local

Nous allons commencer par voir toutes les méthodes qui utilisent le fuseau horaire local de votre système pour renvoyer leur valeur :

`getDate()` permet d'obtenir le jour du mois (1 à 31).

`getDay()` permet d'obtenir le jour de la semaine (0 à 6). Mais attention ! Cela utilise le système américain où le premier jour de la semaine est le dimanche et non le lundi comme en Europe, donc 0 est dimanche :

```
const date = new Date(2020, 0, 12); // Sun Jan 12 2020 00:00:00 GMT+0100
date.getDay(); // 0
```

`getFullYear()` permet d'obtenir les quatre chiffres de l'année.

`getMonth()` permet d'obtenir l'index du mois (**entre 0 et 11**).

`getHours()` permet d'obtenir l'heure (entre 0 et 23).

`getMinutes()` permet d'obtenir les minutes (entre 0 et 59).

`getSeconds()` permet d'obtenir les secondes (entre 0 et 59).

`getMilliseconds()` permet d'obtenir les millisecondes (entre 0 et 999).

Les méthodes utilisant `UTC`

Toutes les méthodes précédentes ont un équivalent pour obtenir des informations mais plus à partir de l'heure locale mais de `UTC`.

Il suffit d'ajouter `UTC` : `getUTCDate()`, `getUTCDay()`, `getUTCFullYear()`, `getUTCMonth()`, `getUTCHours()`, `getUTCMinutes()`, `getUTCSeconds()` et `getUTCMilliseconds()`.

Obtenir le décalage par rapport à `UTC` du système local

Vous pouvez obtenir le décalage en minutes du système local par rapport à `UTC`.

Par exemple si nous sommes en heure française d'hiver, donc en `UTC+1` nous avons un décalage de 60 minutes, et il faut soustraire 60 minutes à l'heure française pour obtenir une heure `UTC`, nous obtenons donc sans surprise :

```
const date = new Date(2020, 0, 2);
date.getTimezoneOffset(); // -60
```

Modifier des dates

Il existe également une grande variété de méthodes permettant de modifier un objet `Date`. Par exemple pour changer l'heure ou le jour.

Les méthodes utilisant l'heure du système local

Nous allons commencer par voir toutes les méthodes qui utilisent le fuseau horaire local de votre système et permettent de modifier un objet date.

Toutes ces méthodes s'appliquent sur un objet date, par exemple :

```
const date = new Date();
date.setFullYear(1998);
```

`setDate()` : permet de définir le jour du mois (1 à 31) pour la date spécifiée.

`setFullYear()` : permet de définir l'année en utilisant 4 chiffres. Vous pouvez également optionnellement passer un deuxième argument pour définir l'index du mois (0 à 11) et un troisième argument pour définir le jour (1 à 31).

`setMonth()` : permet de définir l'index du mois (0 à 11), et optionnellement le jour en deuxième argument, en passant un nombre entre 1 et 31.

`setHours()` : permet de définir l'heure (0 à 23). Vous pouvez également optionnellement passer un deuxième argument pour les minutes, un troisième pour les secondes et un quatrième pour les millisecondes.

`setMinutes()` : permet de définir les minutes (0 à 59). Vous pouvez également optionnellement passer un deuxième argument pour les secondes et un troisième pour les millisecondes.

`setSeconds()` : permet de définir les secondes (0 à 59). Vous pouvez également optionnellement passer un deuxième argument pour les millisecondes.

`setMilliseconds()` : permet de définir le nombre de millisecondes (0 à 999).

`setTime()` permet de définir une date en utilisant un `timestamp` :

```
const timestamp = Date.now();
const date = new Date();
date.setTime(timestamp); // 1578735391202
date; // Sat Jan 11 2020 10:36:31 GMT+0100
```

Les méthodes utilisant `UTC`

Toutes les méthodes précédentes ont un équivalent pour définir des dates mais en utilisant `UTC`.

Il suffit d'ajouter `UTC` : `setUTCDate()`, `setUTCFullYear()`, `setUTCMonth()`, `setUTCHours()`, `setUTCMinutes()`, `setUTCSeconds()` et `setUTCMilliseconds()`.