

Un peu de détente avant de commencer : <https://youtu.be/Pms4a0Gih4U>

Projet Calendrier

Objectif :

L'objectif de ce projet est de mettre en place notre calendrier.

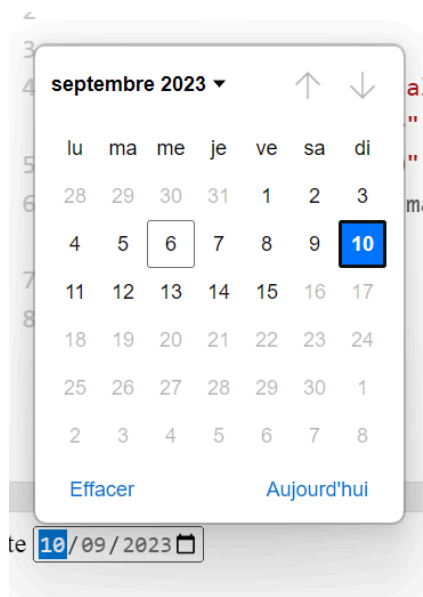
HTML

L'affichage d'un calendrier  dans une page HTML ne pose aucun problème.

La balise `<input>` permet d'afficher un calendrier, il faut pour cela ajouter l'attribut `type="date"`

1. `<label for="calendrier">Date</label>`
- 2.
3. `<input type="date" id="calendrier" name="exemple-calendrier"`
4. `value="2023-09-10"`
5. `min="2023-09-01" max="2023-09-15">`

code : <https://codepen.io/dupontcodepen/pen/MWZbXKL>



Je reste très impressionné par la qualité de l'affichage.

En gras apparaissent les jours de la période indiquée dans les attributs.

En bleu apparaît le jour indiqué dans l'attribut `value` (dans la limite de `min,max`). *Autrement dit si vous donnez la valeur `value="2023-09-16"` qui est en dehors des contraintes, le jour 15 sera affiché !*

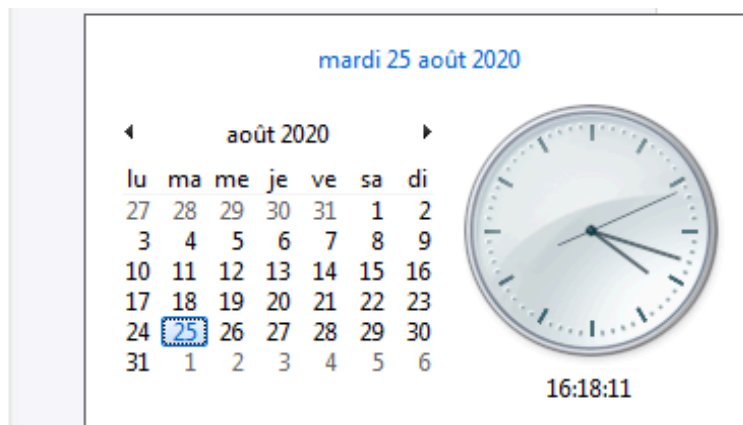
Nous voudrions coder à titre d'exercice notre propre calendrier.

En action

Lors du confinement, j'ai intégré le code de mon calendrier dans un blog :
<https://dupontl3alt.blogspot.com/2020/04/suivez-votre-agenda-virtuel.html>

Analyse d'un code existant

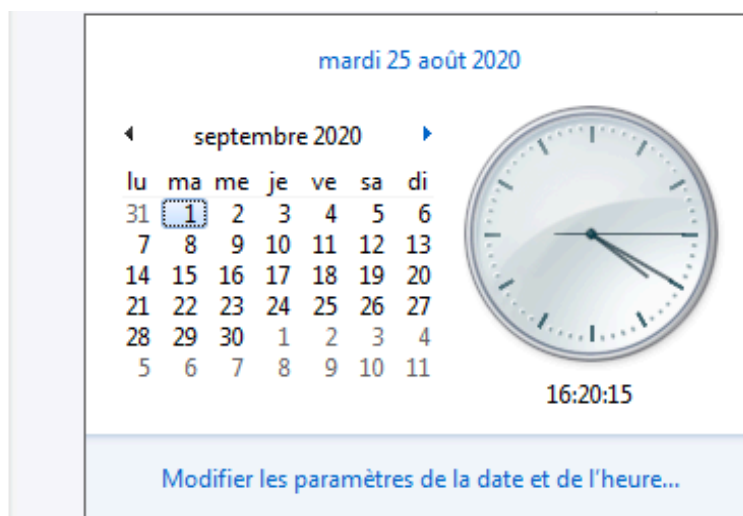
Observons le calendrier de Windows pour le mois d'août 2020 !



Le calendrier affiche 7*6 valeurs.

Il y a 6 semaines affichées. Les semaines 31-36 de l'année courante sont affichées.

Le mois de septembre 2020 est particulièrement intéressant à observer. En effet, le calendrier affiche toujours 7*6 valeurs, alors que le mois de septembre pourrait ne s'afficher que sur 7*5 valeurs (semaines 36-40).



Ainsi, quelque soit le mois, le calendrier affiche toujours 7*6 valeurs.

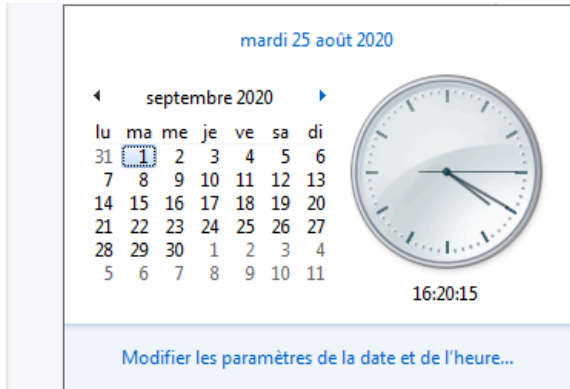
L'idée est d'harmoniser sur 6 semaines la présentation d'un mois.

Contraintes

Nous allons afficher uniquement les jours du mois courant.

Ainsi pour le mois de septembre 2020, notre calendrier devra s'afficher sans faire apparaître le 31 août et les jours du mois d'octobre.

Le tableau suivant permet de comparer les affichages.

Notre calendrier	Windows																																			
<p>Septembre</p> <p>Lu Ma Me Je Ve Sa Di</p> <table><tbody><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td></td></tr><tr><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td></tr><tr><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td></tr><tr><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td></tr><tr><td>28</td><td>29</td><td>30</td><td></td><td></td><td></td><td></td></tr></tbody></table>	1	2	3	4	5	6		7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30					
1	2	3	4	5	6																															
7	8	9	10	11	12	13																														
14	15	16	17	18	19	20																														
21	22	23	24	25	26	27																														
28	29	30																																		

Séparation des contraintes

Une des clefs de la réussite de notre projet est de **séparer** l'aspect génération du code de l'aspect affichage.

Le code qui permet de générer les jours du mois est indépendant de l'affichage.

L'affichage pourrait se résumer aux choix des balises [des](#), [des <Div>](#).

Nous allons donc nous attarder sur le code JS¹ permettant de générer les valeurs du calendrier. Nous reviendrons dans un autre document à l'affichage.

¹ L'idée de ce projet est de se familiariser avec les types de base de JS

Javascript

Nous allons nous intéresser à un objet natif de javascript "Date". Car, l'objet Date dispose d'un grand nombre de propriétés et de méthodes qui faciliteront l'écriture de notre programme.

L'objet Date

L'utilisation de l'objet [Date](#) va nous faciliter grandement notre code².

Je vous laisse prendre *quelques jours* à l'étude de cet objet. 

Le kit de survie³

Pour l'instant, nous allons découvrir quelques méthodes de l'objet Date.

Il nous suffit de répondre à ces questions :

1. Quel jour sommes-nous ?
2. Afficher le jour du 01/09/2023
3. Afficher le jour de la semaine du 01/09/2023 sous forme d'indice. Les indices correspondent à Dimanche - Samedi : 0 - 6

 Voici le code à étudier :

1. Quel jour sommes-nous ?

Tapez dans la console du navigateur :

1. `let today = new Date();`
2. `const options = { year: 'numeric', month: 'long', day: 'numeric', weekday: 'long' };`
3. `console.log(today.toLocaleDateString('fr-CA', options));`

lundi 4 septembre 2023

2. Afficher le jour du 01/09/2023

Tapez le code dans une console :

1. `let firstOfSeptember = new Date(2023, 8, 1);`

² La manipulation dans un programme des dates est en réalité loin d'être simple ; la gestion des différentes zones s'avère bien plus complexe que l'on pense.

³ Découvrez un ensemble de méthodes supplémentaires : [code](#)

2. `const options = { weekday: 'long' };`
3. `console.log(firstOfSeptember.toLocaleDateString('fr-CA', options));`

vendredi

Dans la suite de notre programme, nous allons volontairement nous passer de la méthode `toLocaleDateString`

3. Afficher l'indice du 01/09/2023

1. `let date = new Date(2023, 8, 1),`
2. `indice = date.getDay();`
3. `console.log(indice);`

Bilans

Vous le saviez sans doute déjà, les valeurs numériques des mois et des jours correspondent à :

```
mois = new Array ("Janvier", "Février", "Mars", "Avril", "Mai", "Juin", "Juillet", "Août",  
"Septembre", "Octobre", "Novembre", "Décembre");
```

```
jours4 = new Array("Di", "Lu", "Ma", "Me", "Je", "Ve", "Sa");
```

Ainsi, pour les exemples précédents du kit de survie :

```
mois[8] == "Septembre" et jours[5] == "vendredi"
```

Revenons au calendrier et proposons un code de base pour remplir le tableau de `dim = [6,7]`.

Version 0⁵

Nous allons partir d'un code existant qui permet de générer les valeurs d'un mois.

Analysez le code  suivant :

1. `const calendrier = new Array(),`
2. `mois = 9,`
3. `annee = 2023,`

⁴ Il nous semble étrange de commencer la semaine par le Dimanche !

⁵ Nous appelons cette version "Version 0", et nous n'aurons de cesse d'améliorer le code.

```
4.     temps = new Date(2023, mois-1, 1);
5.
6.     let Start = temps.getDay();
7.         if (Start > 0) {
8.             Start--;
9.         }
10.        else {
11.            Start = 6;
12.        }
13.        let Stop = 31;
14.        if (mois == 4 || mois == 6 || mois == 9 || mois == 11)--Stop;
15.        if (mois == 2) {
16.            Stop = Stop - 3;
17.            if (an % 4 == 0) Stop++;
18.            if (an % 100 == 0) Stop--;
19.            if (an % 400 == 0) Stop++;
20.        }
21.
22.
23.        let nombre_jours = 1;
24.        for (let i = 0; i <= 5; i++) {
25.            for (let j = 0; j <= 6; j++) {
26.                if ((i == 0) && (j < Start)) {
27.                    calendrier.push(" ");
28.                }
29.                else {
30.                    if (nombre_jours > Stop) {
31.                        calendrier.push(" ");
32.                    }
33.                    else {
34.                        calendrier.push(`${nombre_jours}`);
35.                        nombre_jours++;
36.                    }
37.                }
38.            }
39.        }
40.        console.log(calendrier)
```

[Pythontutor](#)

[Github](#)

En particulier, vous devez comprendre l'idée du code :

```
7.      if (Start > 0) {
8.          Start--;
9.      }
10.     else {
11.         Start = 6;
12.     }
```

 Remplacez le test précédent par un test ternaire.

Quelle est la fonctionnalité de ces lignes de code ?

```
13. let Stop = 31;
14.     if (mois == 4 || mois == 6 || mois == 9 || mois == 11)--Stop;
15.     if (mois == 2) {
16.         Stop = Stop - 3;
17.         if (an % 4 == 0) Stop++;
18.         if (an % 100 == 0) Stop--;
19.         if (an % 400 == 0) Stop++;
20.     }
```

Nous pourrions renommer Stop par un identifiant plus explicite, lequel ?

Remplacer la lig.14 par l'utilisation d'un ensemble [Set\(\)](#)⁶.

Remplacer dans le code suivant, les valeurs 5,6,i et j par des variables plus explicites :

```
24.     for (let i = 0; i <= 5; i++) {
25.         for (let j = 0; j <= 6; j++) {
26.             if ((i == 0) && (j < Start)) {
27.                 calendrier.push(" ");
28.             }
29.             else {
30.                 if (nombre_jours > Stop) {
31.                     calendrier.push(" ");
32.                 }
33.                 else {
```

⁶ Ensemble exclusif !

Calendrier : partie code 1 / 9

```
34.         calendrier.push(`${nombre_jours}`);
35.         nombre_jours++;
36.     }
37. }
38. }
39. }
```

Nous pouvons réécrire notre version 0.

Cette réécriture ne modifie pas fondamentalement *le côté fonctionnel* de notre code.