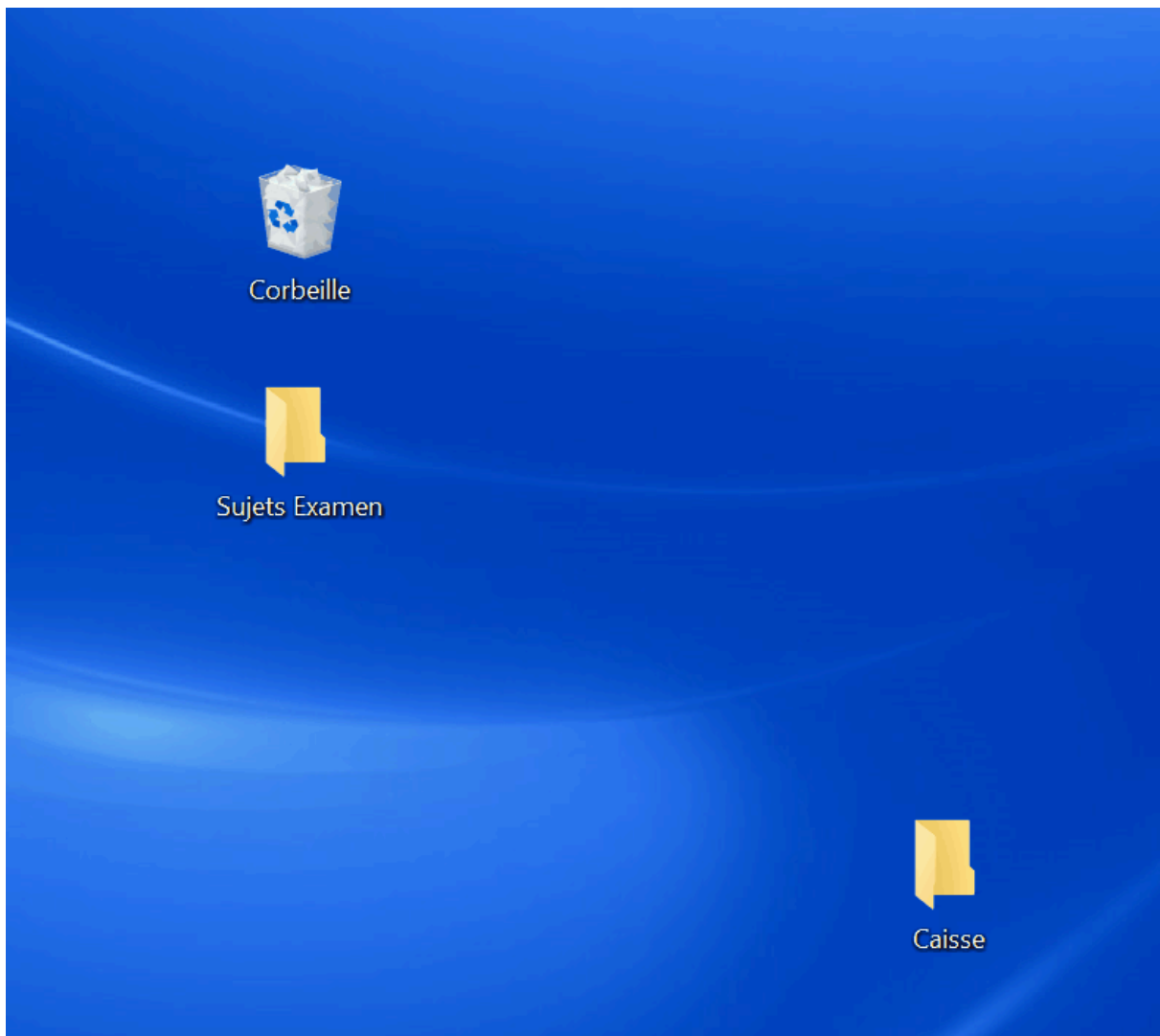


Nous allons écrire notre code avec un éditeur visual Studio.

Rappels

VisualStudio Code

- Créer sur votre bureau (ou dossier favori) un dossier 📁 Caisse.
- Ouvrir Code (📁 bouton droit sur le dossier Caisse).

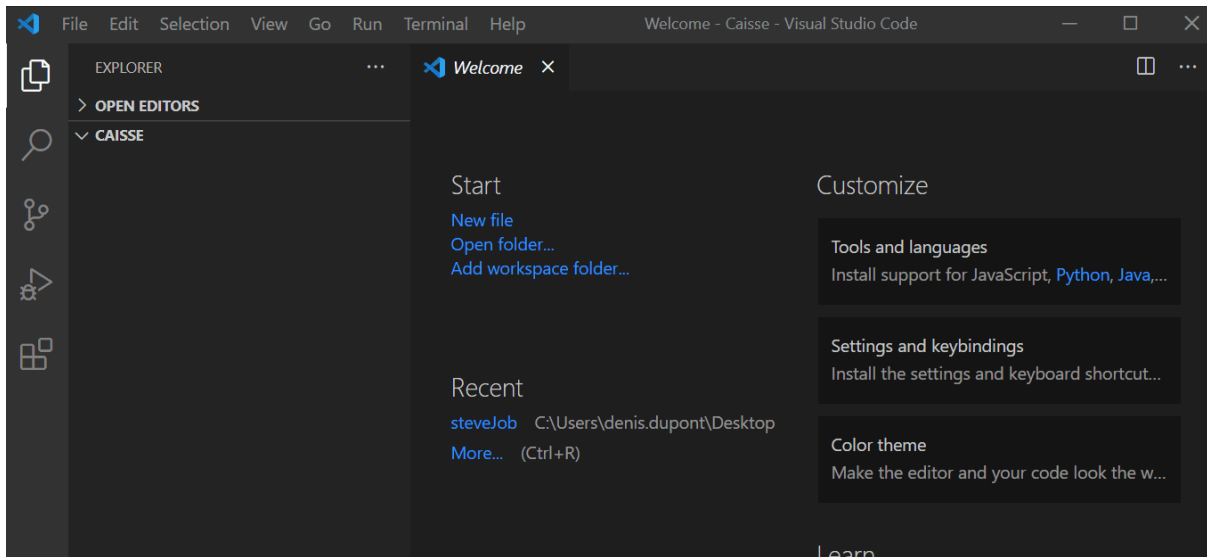


Dans l'application, créer un fichier caisse.html. Dans le code du fichier, tapez ! puis TAB↔

lig.5 : Insérez dans <head> un lien vers un fichier code.js

1. <head>

2. `<meta charset="UTF-8">`
3. `<meta name="viewport" content="width=device-width, initial-scale=1.0">`
4. `<title>Document</title>`
5. `<script src="code.js"></script>`
6. `</head>`



Créer un fichier code.js et insérez le code js suivant :

```
1. class Caisse {
2.   constructor() {
3.     this.clients = 0;
4.     this.somme = 0;
5.   }
6.
7.   _add(element) {
8.     this.somme += element;
9.   }
10.
11.   encaisser(...tableau) {
12.     ++this.clients;
13.     tableau.forEach(this._add, this);
14.   }
15. }
16.
```


```

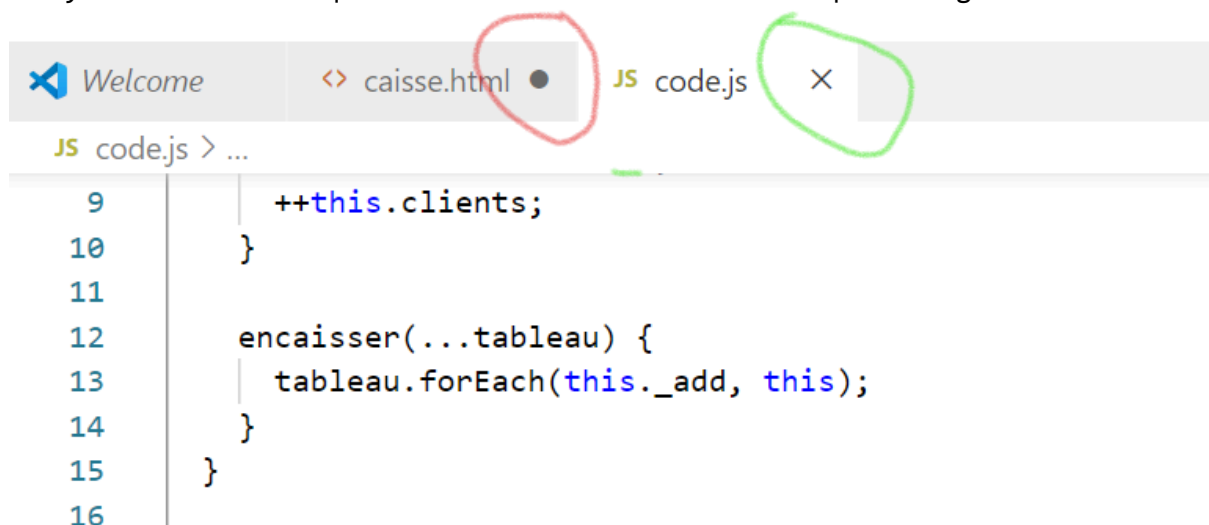
17. let soldes = new Caisse();
18.
19. soldes.encaisser(49.99);
20. soldes.encaisser(239.99, 229.99);
21. soldes.encaisser(9.9, 120.99);
22. soldes.encaisser(239.99, 19.99, 19.99, 19.99, 59.99);
23.
24. for (let [k, v] of Object.entries(soldes)) {
25.     document.body.insertAdjacentHTML(
26.         "beforeend",
27.         `

${JSON.stringify(k)} : ${
28.             k == "somme"
29.                 ? new Intl.NumberFormat("de-DE", {
30.                     style: "currency",
31.                     currency: "EUR"
32.                 }).format(v)
33.                 : JSON.stringify(v)
34.             } </p>`
35.     );
36. }
37.
38. console.log(`Bilan : nous avons ${soldes.compte} clients`);
39. console.log(JSON.stringify(Object.entries(soldes), null,
    "\t"));


```

Pensez à enregistrer vos fichiers.

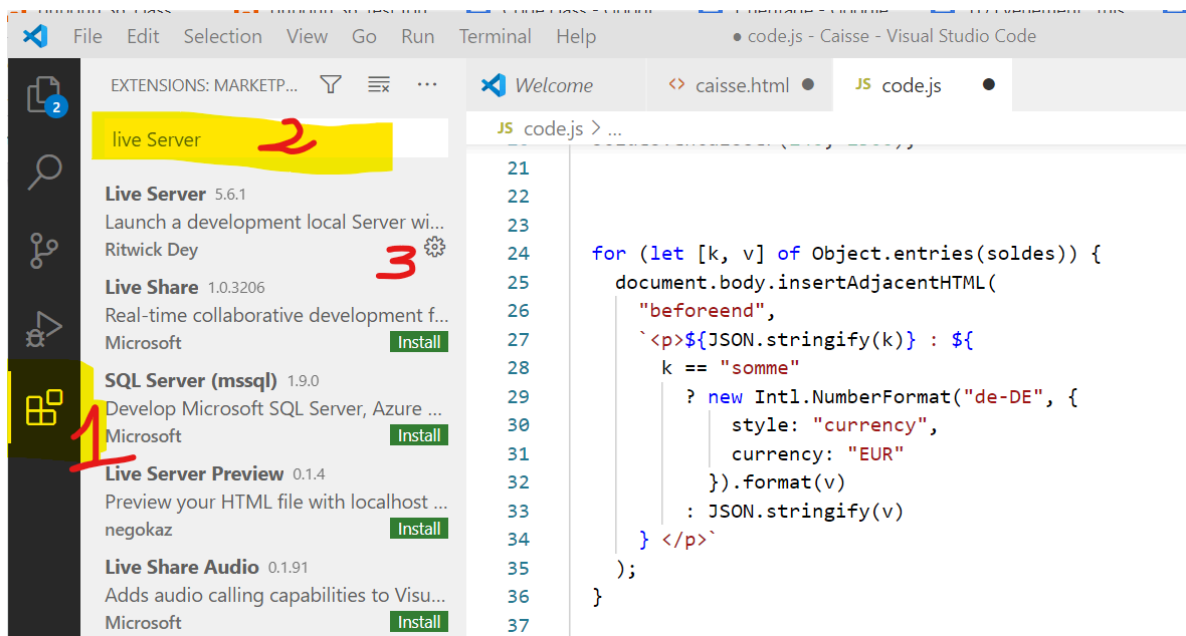
Le symbole  montre que le fichier caisse.html n'est pas enregistré



La figure suivante montre que tous les fichiers sont enregistrés.

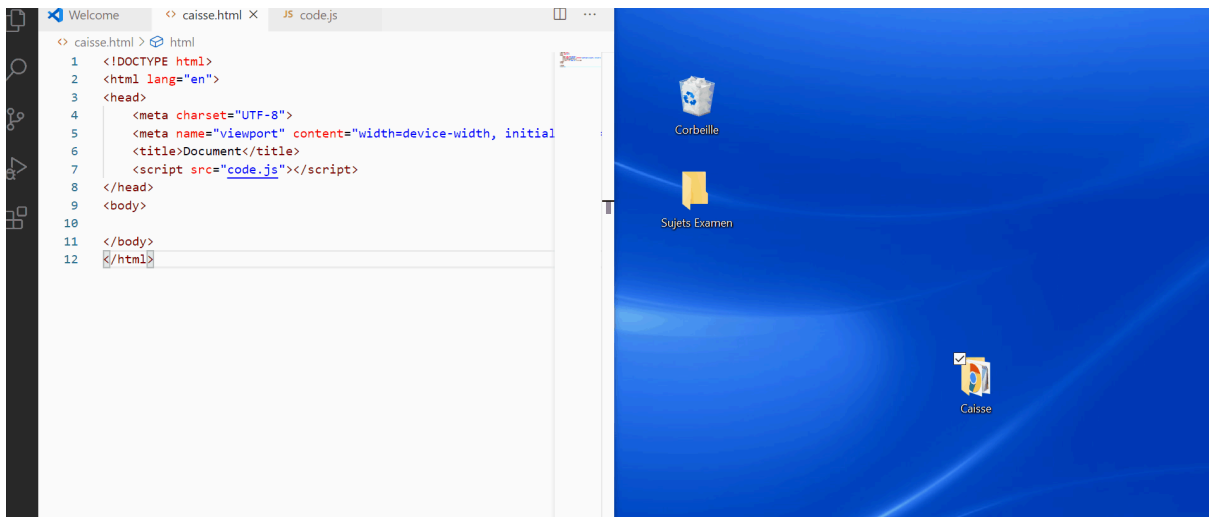
```
Welcome  <> caisse.html X  JS code.js
<> caisse.html > html > body
1  <!DOCTYPE html>
2  <html lang="en">
3  > <head> ...
```

Installez (ou vérifiez) que l'extension **Live Server** est installée
La figure suivante résume les actions en trois étapes.



Lancez l'extension Live Server :

Dans le fichier `caisse.html`, bouton de droite lancer "open with Live Server".



Une page s'affiche mais rien ne s'affiche alors que nous voudrions voir apparaître notre résultat de la journée de soldes.

```
"clients" : 4
```

```
"somme" : 1.010,81 €
```

L'inspection dans la console montre que la méthode `insertAdjacent` ne répond pas :

```
code.js:25 Uncaught TypeError: Cannot read property  
'insertAdjacentHTML' of null
```

On nous indique clairement que l'objet est `null`.

En fait, `body` n'existe pas. En effet, le script est lancé alors que le DOM (la structure en arbre de notre fichier n'est pas en mémoire).

Correction.

Une solution consiste à déplacer le script juste avant la fin du `body`.

1. `<!DOCTYPE html>`
2. `<html lang="en">`
- 3.
4. `<head>`
5. `<meta charset="UTF-8">`

```
6.     <meta name="viewport" content="width=device-width,
      initial-scale=1.0">
7.     <title>Document</title>
8.
9. </head>
10.
11. <body>
12.     <script src="code.js"></script>
13. </body>
14.
15. </html>
```

Relancez LiveServer. Le résultat ne s'affiche toujours pas, c'est l'enfer.

L'inspection de la console indique cette fois

```
1. code.js:8 Uncaught TypeError: Cannot read property 'somme' of
  undefined
2.     at _add (code.js:8)
3.     at Array.forEach (<anonymous>)
4.     at Caisse.encaisser (code.js:13)
5.     at code.js:19
6. _add @ code.js:8
7. encaisser @ code.js:13
8. (anonymous) @ code.js:19
```

Lig. 2 cliquez sur le lien (code.js:8)

Dans la console, nous voyons que *this* est undefined.

```
1 class Caisse {
2   constructor() {
3     this.clients = 0;
4     this.somme = 0;
5   }
6
7   _add(element) {
8     this.somme += element;
9     ++this.clients;
10  }
11
12  encaisser(...tableau) {
13    tableau.forEach(this._add);
14  }
15 }
16
```

Nous sommes face à l'enfer du this¹ !

Evidemment c'est l'occasion d'apprendre à déboguer le code dans visual studio

Déboguer

Commencez par étudier la vidéo "le débogueur".

https://youtu.be/_imeD5H-yCw?t=556

Revenez ici dans un jour² !

Voici la version de débogage dans chrome

<https://youtu.be/N9F6u8JXjiE>

Voici la version de débogage dans Visual Studio

<https://youtu.be/E0vThAWmDsU>

1

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Op%C3%A9rateurs/L_op%C3%A9rateur_this

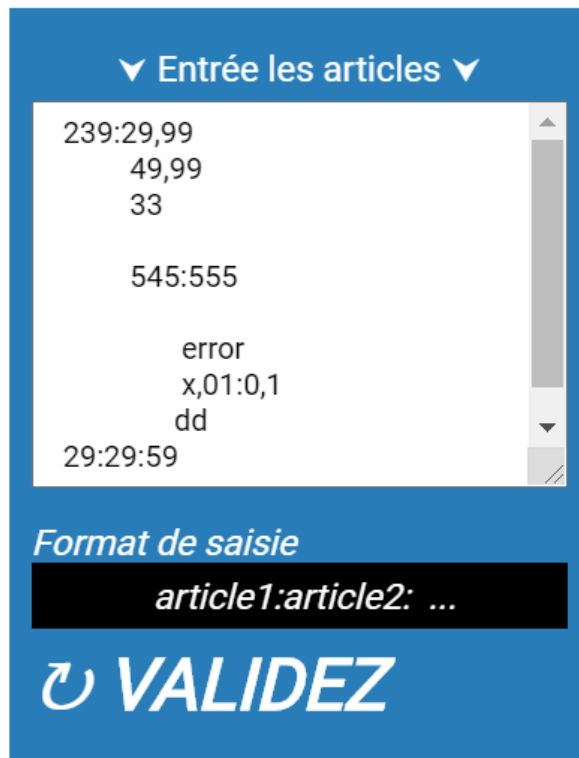
² Un jour minimum pour assimiler les pratiques.

Solution

Une solution (autre que celle donnée dans les vidéos) est de faire appel au fonction fléchée et leur *this* syntaxique.

<https://codepen.io/dupontcodepen/pen/yLJrqdY>

Cette classe peut être reprise dans un projet avec une saisie utilisateur



projet final

[Code](#)