L'objectif est de determiner la difference entre une Arrow Function et une function normale

# 02 - Fonctions fléchées

## 2.1

À faire dans [02-functions.html](02-functions.html)

Convertissez les 3 fonctions en fonctions fléchées et vérifiez que le code fonctionne comme avant la conversion.

syntax :fname : function() { bla bla...bla..}  //fonction normal

const fname : () => { bla bla...bla..}  //fonction Arrow

Code Avant :  
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <meta http-equiv="X-UA-Compatible" content="ie=edge" />

    <title>Fonctions!</title>

  </head>

  <body>

    <script>

      //2.1 Fonctions simples

      function sayHello() {

        console.log("Hello");

      }

      function sayMyName(first, last) {

        console.log(first, last);

      }

      function sayMyAge(age) {

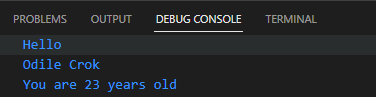
        console.log("You are " + age + " years old");

      }

      sayHello();

      sayMyName("Odile", "Crok");

      sayMyAge(23);

Résultat avant :  


**Code après la conversion des 3 fonctions en fonctions fléchées**

!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <meta http-equiv="X-UA-Compatible" content="ie=edge" />

    <title>Fonctions!</title>

  </head>

  <body>

    <script>

      //2.1 Fonctions simples

      //Arrow fucntion

      let sayHello = () => console.log("Hello");

      //Arrow fucntion

      let sayMyName = (first, last) => console.log(first, last);

      //Arrow fucntion

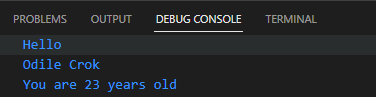
      let sayMyAge = (age) => console.log("You are " + age + " years old");

      sayHello();

      sayMyName("Odile", "Crok");

      sayMyAge(23);

**Résultat après:**

le résultat reste le même

## 2.2

1. Convertissez la fonction `showThis` en notation "Arrow function".

2. Observez le résultat : que se passe-t-il ?

Avant :  
//----------------------//

      //2.2 this

      const object = {

        color: "red",

        shape: "circle",

        threeDimensions: false,

        showThis: function () {

          console.log(this);

        },

      };

      object.showThis();

Résultat :   
{color: 'red', shape: 'circle', threeDimensions: false, showThis: ƒ}

Après :

     //----------------------//

      //2.2 this

      const object = {

        color: "red",

        shape: "circle",

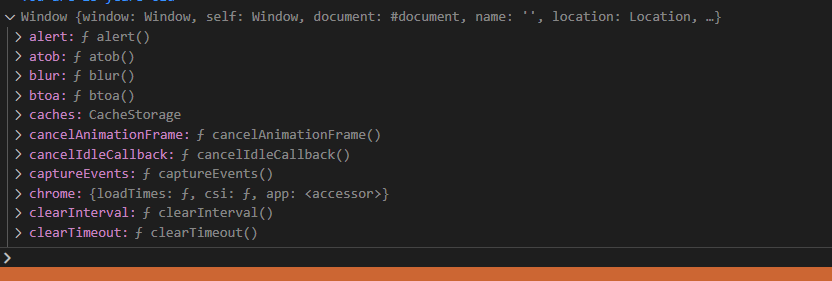
        threeDimensions: false,

        showThis:  () => {

          console.log(this);

        },

      };

Résultat :   


Les fonctions fléchées ne fonctionne pas avec ‘this’. Si cela est accessible, il est pris de l'extérieur.

## 2.3

Complétez le code pour reproduire le comportement de la section 1 mais cette fois-ci grâce à l'objet `odile`. Vous utiliserez dès que possible des "Arrow functions".

Avant :   
 //----------------------//

      //2.3 Application

      const odile = {

        name: {

          first: "Odile",

          last: "Crok",

        },

        age: 23,

        sayHello: function () {

          console.log("Hello");

        },

        sayMyName: function () {

          console.log(this.name.first + " " + this.name.last);

        },

        sayMyAge: function () {

          console.log(this.age);

        },

      };

      odile.sayHello();

      odile.sayMyName();

      odile.sayMyAge();

    </script>

Résultat :   


Code après :  
const odile = {

        name: {

          first: "Odile",

          last: "Crok",

        },

        age: 23,

        sayHello: () =>

          console.log("Hello")

        ,

        sayMyName: () =>

          console.log(this.name.first + " " + this.name.last)

        ,

        sayMyAge: () =>

          console.log(this.age)

        ,

      };

      odile.sayHello();

      odile.sayMyName();

      odile.sayMyAge();

    </script>

  </body>

</html>

Résultat :

