0.1 Les Variables

Définition du Javascript Le Javascript est un langage de programmation créé en 1995 par la société Sun. Il s'agit d'un langage haut niveau à typage dynamique. Il autorise notamment la réaffectation quelque que soit le type de la valeur stockée initialement dans la variable. Il est compris par les navigateurs car ils sont pourvu d'un moteur JS (pour Chrome c'est le moteur V8).

Définition d'une variable Une variable est une entité à qui on alloue un espace mémoire pour stocker une valeur. En Javascript on utilise un mot clé (keyword) pour assigner

```
<html>
    <head></head>
    <body>
        <form>
            <section id="form-inputs"></section>
            <section id="form-buttons">
                <button id="submit-button">Valider
                    </button>
            </section>
        </form>
    </body>

</html>
```

```
const existingHtmlSubmitButton = document.
    getElementById('submit-button');
// alternative document.querySelector("#submit-
    button");
```

Explication Ci-dessus, je définis une variable nommée existingHtmlSubmitButton. Je stocke dans cette variable assignée en constante (en l'occurrence non modifiable dans son type d'objet et sa valeur) l'objet qui représente l'élément HTML button identifié par l'identifiant(id) "submit-button".

0.2 Les fonctions

Définition d'une fonction Une fonction représente un morceau de code réutilisable. Elle peut accepter des paramètres et on peut lui passer des argu-

ments quand on l'exécute. Une fonction peut être définie et appelée ensuite.

```
    <html>
        <head></head>
        <body>
            <form>
                <section id="form-inputs"></section>
                <section id="form-buttons">

                </section>
            </form>
        </body>

    </html>
```

```
function notExistingElementErrorMessage(id){
    console.log('l\'element ayant l\'id ' + elementId +
        ' n\'existe pas')
}

function createHtmlElement(elementType, id){

    const element = document.createElement(elementType)
    element.id = id
    return element

}

function giveClassesToElement(elementId, elementClasses
   ){

    const element = document.getElementById(elementId)
    if(element !== null){
        for(const cl of classes){
            element.classList.add(cl)
        }
    } else {
        notExistingElementErrorMessage(id)
    }
```

```
}

function appendNewHtmlElementToExistingParent(
   futureChild, parentId){
    const parent = document.getElementById(parentId)
    if(parent !== null){
        parent.appendChild(futureChild)
    } else {
        notExistingElementErrorMessage(id)
    }
}

function init(){

    createHtmlElement("button", "submit-button")
    giveClassesToElement("submit-button", ["btn", "form
       -btn", "form-btn--submit"])
    const newButton = document.getElementById("submit-
       button")
    appendNewHtmlElementToExistingParent(newButton, "
       form-buttons")
}

init()
```

Explication Dans la définition de la fonction init() vous pouvez voir 4 appels de fonction l'ordre est important :

1. La fonction de création qui va donc créer le bouton et lui donner l'identifiant "submit-button"
2. La fonction giveClassesToElement() qui donne des classes à l'élément nouvellement créé
3. La fonction getElementById() stocker l'élément nouvellement créé dans une variable newButton assignée en constante
4. La fonction appendNewHtmlElementToExistingParent() ajoute à un élément existant le bouton

0.3 Les tableaux

```
giveClassesToElement("submit-button", ["btn", "form-btn
    ", "form-btn--submit"])
```

Signature de la fonction Dans l'exemple ci-dessous vous pouvez voir que j'appelle la fonction giveClassesToElement(). Je lui passe deux arguments car dans la signature de cette fonction il y a deux paramètres le premier est elementId et elementClasses.

Les arguments Les arguments sont les valeurs que vous donnez lors de l'appel de la fonction aux paramètres qu'elle attend

Les tableaux Comme vous pouvez le voir l'argument que j'ai passé en deuxième position est un tableau. En effet vous pouvez voir des crochets [].

```
["btn", "form-btn", "form-btn--submit"]
```

Les éléments d'un tableau Pour accéder aux éléments d'un tableau vous utilisez l'indexation car chaque valeur stockée dans le tableau est récupérable via sa position dans le tableau. Le premier indice est 0 donc la valeur "btn" est accessible ainsi :

```
const arr = ["btn", "form-btn", "form-btn--submit"]
console.log(arr[0]) // "btn"
```