**Technologies Internet 1**

**Le langage JavaScript**

Table des matières

[1. Qu'est-ce que le JavaScript ? 2](#_Toc147692485)

[1.1. Description 2](#_Toc147692486)

[1.2. Intégration 2](#_Toc147692487)

[2. Eléments de langage 3](#_Toc147692488)

[2.1. Les variables : typage dynamique 3](#_Toc147692489)

[2.2. Les conditions 3](#_Toc147692490)

[2.3. Les boucles 3](#_Toc147692491)

[3. Comment récupérer en javascript … 3](#_Toc147692492)

[4. Exercices 5](#_Toc147692493)

[5. Evénements et fonctions 6](#_Toc147692494)

[5.1. Fonctions classiques 6](#_Toc147692495)

[5.2. Fonctions fléchées 7](#_Toc147692496)

[5.3. Boucle dans une fonction forEach() 9](#_Toc147692497)

[6. Exercices (2) 10](#_Toc147692498)

# Qu'est-ce que le JavaScript ?

## Description

Le JavaScript appartient à la 3è couche du web, relative au comportement des objets de la page web et à l'interactivité.

Le langage est

* Principalement interprété côté client
* Événementiel : il écoute tout événement pouvant survenir dans la page web, de manière à déclencher un traitement : clic, changement dans une liste, chargement terminé, etc.
* Fonctionnel
* Orienté prototypes : les objets ne sont pas créés à partir d'une classe statique, mais d'un objet de base, le prototype, qui est modifiable.

Le moteur JS du navigateur démarre lorsque la page web, composée de HTML et de CSS, est prête à être déployée, c'est-à-dire lorsque les éléments sont en place pour que le JS puisse agir sur ces derniers.

Par sécurité, le JS ne s'applique le plus souvent qu'à l'onglet courant.

## Intégration

Bonne pratique : le code javascript s'écrit dans des fichiers externes aux pages html et le fichier le contenant sont chargés par l'index, entre les balises <head> et par une balise <script></script>. En ajoutant l'attribut defer, on assure que le chargement des éléments html continuera lorsque le navigateur rencontrera la balise <script>. Il n'y aura pas de blocage momentané de la page :

<script src="./lib/js/fonctions1.js" defer></script>

L'attribut defer assure que le navigateur n'interrompra pas le chargement du DOM (objets html) au moment où il rencontrera la balise <script>. Les chargements s'exécuteront parallèlement et le javascript ne s'exécutera qu'une fois le html entièrement chargé. Si plusieurs fichiers javascript doivent être chargés, defer permettra de les charger dans l'ordre où ils apparaissent sur la page.

<script src="./lib/js/fonctions1.js" async></script>

Avec l'attribut async, l'ordre de chargement des scripts n'est pas respecté.

# Eléments de langage

## Les variables : typage dynamique

On les déclare par let ou const, avant leur utilisation. La portée de ces variables est limitée au bloc où elles sont déclarées.

Une variable peut être de différents types :

* Nombre : let n = 4;
* Chaîne : let ch = "une chaîne de caractères";
* Booléen : let b = 5 < 10; b = true; b = false;
* Tableau : let tab = ["Aa", "Bb", "Cc"]; tab = ["Aa", 10, true, 5.0];

Afficher le contenu d'un tableau par forEach ou une boucle for classique :

**let *tab*** = [**"Aa"**, 10, **true**, 5.0];  
  
***tab*.forEach**(**function**(elt){  
 ***console***.log(elt);  
});  
  
**for**(**let** i = 0;i<***tab***.**length**;i++){  
 ***console***.log(***tab***[i]);  
}

## Les conditions

Comme en C, y compris switch case et l'opérateur conditionnel ternaire (exemple ci-dessous).

Noter le signe "triple égal", qui retourne true si aussi bien la valeur que le type correspondent.

**let *tab*** = [**"Aa"**, 10, **true**, 5.0];

***console***.log(***tab***[1]===**"10"** ? **true** : **false**); //false

***console***.log(***tab***[1]===**10** ? **true** : **false**); // true

## Les boucles

Comme en C. La boucle forEach peut s'utiliser sur les objets qui en disposent (voir leur prototype).

# Comment récupérer en javascript …

1. **D'abord cibler l'objet :**

let variable = document.getElementById('#id');

ou document.getElementsByClassName('classe\_css');

ou document.getElementsByName('attribut\_name');

ou document.getElementsByTagName('p');

ou document.querySelector('p ');

ou document.querySelectorAll('p');

1. **Puis récupérer/ajouter la valeur**
2. *Récupérer le contenu texte d'une balise*

variable.textContent; //*variable est une balise unique*

variable[i].textContent; //*variable existe en plusieurs exemplaires*

1. *Récupérer la valeur d'un attribut*

*D'abord sélectionner l'attribut de la balise, par exemple <p> :*

attribut = variable.querySelector('p[nom\_attribut]');

*Ensuite récupérer la valeur par getAttribute :*

valeur\_attribut = attribut.getAttribute('<nom\_attribut>');

1. *Récupérer la valeur d'un input*

variable.value;

1. *Ajouter du texte avec HTML : propriété innerHTML*

let variable = document.getElementById('#id');

variable.innerHTML = '<span class="">' + valeur +'</span>'

1. *Ajouter du texte simple : propriété innerText*

let variable = document.getElementById('#id');

variable.innerText = 'Texte sans formatage';

# Exercices

Copiez / composez le code HTML de chaque exercice dans un fichier séparé du dossier pages. Pour chacun de ces fichiers, créez un nouveau fichier lib/js/fonctions<numero>.js.

1. **Soit le fichier pages/js1.php**

<**p id="p1"**>pomme</**p**>  
<**p id="p2"**>fraise</**p**><**div id="div\_p2"**></**div**>  
<**p id="p3"**>orange</**p**>

<**input type="text" name="fruit" id="nom" value="cerise"**>

* Récupérez la valeur du premier paragraphe et affichez-la dans la console.
* Récupérez la valeur du second paragraphe et affichez dans la div div\_p2 la phrase "Fruit d'été", avec du HTML
* Récupérer d'un seul coup les 3 paragraphes et affichez-en les valeurs dans la console, à l'aide d'une boucle.
* Affichez dans la console le contenu de la balise input, "cerise".

1. **Remplir une table HTML par le javascript.**

Fichiers pages/js2.php et lib/js/fonctions2.js

Dans le fichier pages/js2.php, copiez le code suivant :

<**table**>  
 <**tr**>  
 <**td**></**td**>  
 <**td**></**td**>  
 <**td**></**td**>  
 <**td**></**td**>  
 <**td**></**td**>  
 </**tr**>  
</**table**>

En javascript, créez un tableau js et initialisez-le avec des noms de pays.

Rappel : un tableau simple se crée et s'initialise ainsi :

***let tab*** = [**"Aa"**, 10, **true**, 5.0];

L'objectif est d'obtenir dans le navigateur une ligne de tableau remplie de valeurs, par exemple :

Une image contenant texte, Police, capture d’écran, ligne

Description générée automatiquement

1. Soit le code HTML suivant (fichiers pages/js3.php et lib/js/fonctions3.js) :

<**div class="cards"**>  
 <**div class="card"**>  
 Pommes <**span data-suppl="apple" class="prix"**>1.54 €</**span**>  
 </**div**>  
 <**div class="card"**>  
 Bananes <**span data-suppl="banana" class="prix"**>1.15 €</**span**>  
 </**div**>  
 <**div class="card"**>  
 Poires <**span data-suppl="pear" class="prix"**>1.95 €</**span**>  
 </**div**>  
 <**div class="card"**>  
 Oranges <**span data-suppl="orange" class="prix"**>3.48 €</**span**>  
 </**div**>  
</**div**>

Affichez dans la console chaque fruit et son prix.

Pour chaque fruit, récupérez l'attribut data-suppl puis affichez la valeur de cet attribut dans la console, par exemple de la manière suivante :

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

# Evénements et fonctions

## Fonctions classiques

Lorsqu’un objet de la page htm a été récupéré, par exemple, à l’aide de document.querySelector(), on peut lui associer un écouteur d’événement. Cette fonction écoutera l’interaction spécifiée dans la page web et appliquera la fonction de traitement également précisée.

Exemple

<**p id="paragraphe1"**>Cliquez</**p**>

**let *recup\_p*** = ***document***.querySelector(**'#paragraphe1'**)  
**if**(***recup\_p***){  
 ***recup\_p***.addEventListener(**'click'**, **function**(){  
 ***console***.log(**"Ce paragraphe contient "**,***recup\_p***.**textContent**);  
 });  
}

Le code de la fonction ci-dessus peut être dissocié des paramètres de l’objet écouteur :

**let *recup\_p*** = ***document***.querySelector(**'#paragraphe1'**)  
**if**(***recup\_p***){  
***recup\_p***.addEventListener(**'click'**,***afficher***);  
  
 **function** *afficher*(){  
 ***console***.log(**"Texte affiché dans une fonction callback : "**,***recup\_p***.**textContent**);  
 }  
}

La fonction afficher s’appelle **callback** (fonction de rappel). Un callback est simplement une fonction fournie en paramètre d’une autre fonction. La variable recup\_p sera automatiquement communiquée à la fonction séparée. Le résultat est identique.

On utilisera de préférence une fonction séparée lorsque le code de la fonction pourra être réutilisé dans d’autres contextes, ou bien lorsque la séparation permettra de clarifier le code.

**Remarque** : lorsque le traitement fonctionnel n’est pas séparé, la fonction est dite **anonyme**.

Autre manière : la fonction écouteur appelle une fonction séparée :  
  
***recup\_p***.addEventListener(**'click'**,**function**(){  
 *afficher*();  
});

**function** *afficher*(){  
 ***console***.log(**"Texte affiché dans une fonction appelée à l'intérieur d'une autre fonction : "**,***recup\_p***.**textContent**);  
}

## Fonctions fléchées

Une fonction fléchée est un raccourci d’une fonction classique. Cette syntaxe est très utilisée depuis son introduction dans la version ES6 (2016+).

<**div**>  
 <**p id="paragraphe2"**>Cliquez pour une fonction fléchée</**p**>  
</**div**>

**let *recup\_p2*** = ***document***.querySelector(**'#paragraphe2'**);  
**if**(***recup\_p2***){  
 ***recup\_p2***.addEventListener(**'click'**, () => {  
 ***console***.log(**"Clic pour une fonction fléchée"**);  
 });  
}

Dans cet exemple, le mot clé *function* a disparu, les parenthèses subsistent et une flèche => précède les accolades.

Lorsque la fonction ne contient qu’une seule instruction, on peut encore la simplifier en omettant les accolades :

**let *recup\_p2*** = ***document***.querySelector(**'#paragraphe2'**);  
**if**(***recup\_p2***){***recup\_p2***.addEventListener(**'click'**, () => ***console***.log(**"Clic pour une fonction fléchée"**));  
}

Une fonction fléchée peut communiquer contenir des paramètres :

<**div**>  
 <**p id="paragraphe3"**>Cliquez pour une fonction fléchée avec paramètre </p>  
</**div**>

**let *tous\_les\_parag*** = ***document***.querySelectorAll(**'.classe\_parag'**);  
**let *cliquer\_ici*** = ***document***.querySelector(**'#cliquer\_ici'**);  
**if**(***tous\_les\_parag*** && ***cliquer\_ici***){  
 ***cliquer\_ici***.addEventListener(**'click'**, () => {  
 ***tous\_les\_parag***.forEach((elt) => {  
 ***console***.log(elt.**textContent**);  
 });  
 });  
}

## Boucle dans une fonction forEach()

Pour itérer dans un ensemble d’objet, on peut utiliser une des boucles classiques ou bien une boucle forEach(<variable>).

La variable de la boucle forEach(<variable>) représente chaque objet itéré : un élément de tableau, de chaîne, de map ou de set (ces deux objets seront vus plus tard).

<**div**>  
 <**p class="txtGras" id="cliquer\_ici"**>Cliquez ici pour une boucle</**p**>  
 <**p class="classe\_parag"**>Paragr 1</**p**>  
 <**p class="classe\_parag"**>Paragr 2</**p**>  
 <**p class="classe\_parag"**>Paragr 3</**p**>  
</**div**>

**let *tous\_les\_parag*** = ***document***.querySelectorAll(**'.classe\_parag'**);  
**let *cliquer\_ici*** = ***document***.querySelector(**'#cliquer\_ici'**);  
**if**(***tous\_les\_parag*** && ***cliquer\_ici***){  
 ***cliquer\_ici***.addEventListener(**'click'**, () => { //fonction fléchée  
 ***tous\_les\_parag***.forEach((elt) => {   
 ***console***.log(elt.**textContent**);  
 });  
 });  
}

On peut itérer sur tous\_les\_parag puisque l’ensemble est un tableau et dispose de la fonction forEach, ainsi que la console permet de le constater :

**let *tous\_les\_parag*** = ***document***.querySelectorAll(**'.classe\_parag'**);  
***console***.log(**"liste : "**,***tous\_les\_parag***);

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

# Exercices (2)

**HTML**

<h2>Evénements et fonctions</h2>  
  
<div id="clic1"><p>Premier clic, fonction anonyme</p></div>  
<div id="clic2"><p>Deuxième clic, callback</p></div>  
<div id="clic3"><p>Troisième clic, fonction fléchée</p></div>  
<div id="clic4"><p>Quatrième clic, fonction fléchée sans accolades</p></div>  
<div id="clic5" data-valeur="5"><p>Cinquième clic, fonction fléchée appelant une autre fonction fléchée paramétrée qui calcule le triple de la valeur récupérée</p></div

*Exercice 1*

Dans la console, affichez "Premier clic à partir d’une fonction anonyme", lorsqu’on clique sur <div id="clic1"></div>. Utilisez une fonction classique (non fléchée) anonyme.

*Exercice 2*

Dans la console, affichez "Deuxième clic à partir d’une callback", lorsqu’on clique sur <div id="clic2"></div>. Utilisez une fonction callback.

*Exercice 3*

Dans la console, affichez "Troisième clic à partir d’une fonction fléchée", lorsqu’on clique sur <div id="clic3"></div>. Utilisez une fonction fléchée.

*Exercice 4*

Dans la console, affichez "Troisième clic à partir d’une fonction fléchée courte", lorsqu’on clique sur <div id="clic4"></div>. Utilisez une fonction fléchée sans accolades.

*Exercice 5*

<**div id="clic5" data-valeur="5"**><**p**>

Récupérez l'objet clic5 ainsi que la valeur de l'attribut data-valeur.

Lors d'un clic sur la div, utilisez une fonction fléchée (ou classique) qui appelle une fonction tripler en lui communiquant le nombre 3. La fonction cliquer doit retourner le triple de la valeur de data-valeur.

Affichez dans la console " le triple de 5 est 15".

# Exercices (3)

*Exercice 1*

<**h2**>Afficher un ensemble de paragraphes</**h2**>  
<**h2**>Evénements et fonctions</**h2**>  
  
<**div id="clic1"**><**p**>Premier paragraphe</**p**></**div**>  
<**div id="clic2"**><**p**>Deuxième paragraphe</**p**></**div**>  
<**div id="clic3"**><**p**>Troisième paragraphe</**p**></**div**>  
<**div id="clic4"**><**p**>Quatrième paragraphe</**p**></**div**>  
<**div id="clic6"**><**p**>Cliquer ici pour afficher d'un coup dans la console le contenu de tous les autres paragraphes</**p**></**div**>

* Afficher dans la console l'ensemble des paragraphes lorsqu'on clique sur <div id="clic6"></div>

*Exercice 2 : événement* change

<**h2**>Jus de fruits - boutons radio et événement change</**h2**>  
<**div id="b\_radios"**>  
 Orange <**input type="radio" name="jus" id="orange" data-boissons="jus"**><**br**>  
 Pomme <**input type="radio" name="jus" id="pomme" data-boissons="jus"**><**br**>  
 Ananas <**input type="radio" name="jus" id="ananas" data-boissons="jus"**><**br**>  
 Multifruit <**input type="radio" name="jus" id="multifruit" data-boissons="jus"**><**br**>  
 Fraise <**input type="radio" name="jus" id="fraise" data-boissons="jus"**><**br**>  
</**div**>

Afficher dans la console le choix effectué.