JAVASCRIPT

- Pas Java
- Pour ajouter de l'interactivité aux pages Web

ORGANISATION

- Faire afficher les extensions de fichiers
- Isoler l'exercice du reste
- Travailler dans un seul dossier à la fois

OUTILS

- Travailler avec l’inspecteur Web
- Vérifier la console à chaque changement
- Désactiver le cache

DEBRIEF EXERCICES

Pour insérer du **JavaScript** directement dans une page :

```
<script type="text/javascript">
    alert('Hello INSEEC!');
</script>
```

Ça s'appelle *JavaScript en ligne* ou *inlined JS*

Mais inconvénients…

Si le même script est utilisé sur toutes les pages du site (extrêmement fréquent), il est rechargé par le réseau à chaque chargement de page. Pas bon…

AU PASSAGE...

```
<script type="text/javascript">
```

`text/javascript` est le type MIME de JavaScript

- `text/css` pour du CSS
- `text/plain` pour du texte simple
- `text/html` pour du HTML
- `image/jpg` pour une image JPEG
- `text/png` pour une image PNG

Pour en savoir plus, voir Type MIME

Pour appeler un fichier **JavaScript** externe :

```
<script type="text/javascript" src="index.js"></script>
```

- Mis en cache !
- Chargé une seule fois

Contenu du fichier `index.js` :

```
alert('Hello INSEEC!');
```

(Ce qui était entre les balises <script>)

On place cette balise
juste avant la balise </body>

Permet de ne charger / exécuter le JavaScript qu'après chargement complet du DOM par le navigateur.

TYPES DE DONNÉES JS

Chaîne de caractère

```
var toto = 'Une chaîne de caractère';
var tutu = "Une autre chaîne de caractère";
```

Les chaînes de caractère en JavaScript peuvent être délimités
soit par des guillemets simples (') soit par des guillemets doubles (").

Nombre entier

```
var toto = 123;
var tutu = parseInt('123');
```

La fonction `parseInt()` permet de convertir une chaîne de caractère en nombre entier

Nombre à virgule

```
var toto = 12.5;
var tutu = parseFloat('12.5');

// Affichera 12 !
console.log(parseInt('12.5'));
```

Tableau

```
// un tableau vide
var toto = [];

// un tableau à quatre éléments
var tutu = ['A', 'B', 'C', 'D'];
```

Accéder aux éléments d’un tableau

```
var tutu = ['A', 'B', 'C', 'D'];

// accéder au premier élément (A)
tutu[0]

// accéder au quatrième élément d’un tableau (D)
tutu[3]

// erreur ! le tableau n’a que quatre éléments
// donc pas d’index 4
tutu[4]
```

Connaître la longueur d'un tableau

```
var tutu = ['A', 'B', 'C', 'D'];

// affiche 4
console.log(tutu.length);

// on peut donc boucler sur tout un tableau
for (var i = 0; i < tutu.length; i++) {
    console.log(tutu[i]);
}
```

Modifier un tableau

```
var tutu = ['A', 'B', 'C', 'D'];

// ajoute à la fin
tutu.push('E', 'Z');

// ajoute au début
tutu.unshift('W');

// retire le dernier item et le retourne
var item = tutu.pop();

// affiche 'Z'
console.log(item);
```

Ça n’est qu’un aperçu…

Voir la documentation sur MDN

Objet

```
// un objet vide
var titi = {};

// un objet avec deux propriétés
var coords = {
    lat: 45.769245,
    lng: 4.826735
};

// affiche la latitude
console.log(coords.lat);

// affiche la longitude (moyen d'accès alternatif)
console.log(coords['lng']);
```

Un objet avec des propriétés et des méthodes

```
var voiture = {
    vitesse: 0,
    direction: 0,
    accelerer: function (n) {
        // …
    },
    ralentir: function (n) {
        // …
    },
    tourner: function (angle) {
        // …
    }
};
```

undefined

```
if (typeof toto === 'undefined') {
    // quand toto n'existe absolument pas
}
```

OPÉRATEURS

`var` pour définir une variable

Modulo (%) pour obtenir un reste

```
// Affichera 5 (le reste de la division de 23 par 9)
console.log(23 % 9);
```

$$23 = 9 \times 2 + 5$$

++ et - -

```
var i = 0;

// ajoute 1 à i
i++

// retire 1 à i
i--
```

$$23 = 9 \times 2 + 5$$

Comparaisons

==	égalité simple
!=	inégalité simple
===	égalité de valeur **et** de type
!==	inégalité de valeur **et** de type
>	plus grand que
<	plus petit que
<=	plus petit ou égal

EXERCICE 1

Pour créer une boîte de dialogue :

```
alert('Hello INSEEC!');
```

- `alert` est le nom d'une fonction
- les parenthèses servent à l'appeler
- entre les parenthèses se trouvent les paramètres de la fonction
- ici une *chaîne de caractère* : du texte entre guillemets

La fonction `alert` est pré-définie par le navigateur.

Mais on peut définir nos propres fonctions…

```
function doSomething(param) {
    // Ici on fait quelque chose, par exemple…
    console.log(param);
}
```

Et les appeler

```
doSomething();
```

Et à propos…

```
// Ça, c'est un commentaire, sur une ligne

/* Et ça aussi,
   mais attention à bien le refermer */
```

EXERCICE 2

Pour afficher un message dans la console de l'inspecteur Web :

```
console.log('Hello INSEEC!');
```

Voir la doc de `console.log`

`console` est un objet JavaScript fourni par le navigateur.
`log` est une `méthode` de l'objet console (une fonction spécifique à un objet).
On accède à la méthode d'un objet grâce à l'opérateur `.`

EXERCICE 3

Utiliser un dialogue de confirmation
et n'afficher un message dans la console
que si l'utilisateur clique sur OK.

```
var choix = confirm('Afficher un message ?');
if (choix) {
    console.log('En v'la un message !');
}
```

`var` est un mot clé JS.
`choix` est le nom de la variable (une boîte).
La fonction `confirm` retourne vrai ou faux.
`if` est une structure de contrôle

Un bloc de code est contenu entre des accolades.

```
if (test) {
    // Ce block est exécuté si test est vrai
}
else {
    // Ce block est exécuté si test est faux
}
```

EXERCICE 5

Boucle `for`

```
for (var i = 1; i <= 1000; i = i + 1) {
    console.log(i);
}
```

- Initialisation
- Condition
- Incrémentation

`for` est une autre structure de contrôle JS.

Boucle `while`

```
var i = 1;
while (i <= 1000) {
    console.log(i);
    i++;
}
```

EXERCICE 6

Fibonacci

```
function fibonacci(max) {
    var fib = [0, 1];
    for (var i = 2; i < max; i++) {
        fib[i] = fib[i - 2] + fib[i - 1];
    }
    return fib;
}

var resultat = fibonacci(20);
console.log(resultat);
```

`fib` est un tableau. L'index des tableaux commence à 0.
`fib[0]` pour accéder au premier élément.
`return` pour renvoyer la valeur.

Permet d’approcher le nombre d’or.

```
var fib = fibonacci(50);
console.log('Phi ~ '+ (fib[49] / fib[48]));
```

EXERCICE 7

Récupération d'un élément du DOM et pose d'un écouteur

```
function showMessage() {
    console.log('Bouton cliqué !');
}

var button = document.querySelector('.navToggle');
button.addEventListener('click', showMessage);
```

`document` est l'élément racine de la page, fourni par le navigateur.
`querySelector` est une méthode de `document`.
Sélecteur CSS en paramètre.
`addEventListener` est une méthode de l'élément `button`.
Au clic, on exécute la fonction `showMessage`

Pourrait aussi s'écrire

```
var button = document.querySelector('.navToggle');
button.addEventListener('click', function () {
    console.log('Bouton cliqué !');
});
```

Avec une fonction anonyme ou *closure*.

EXERCICE 8

Ajout / modification d'une classe

```
var button = document.querySelector('.navToggle');
button.addEventListener('click', function () {
    var body = document.querySelector('body');
    body.classList.toggle('nav-on');
});
```

`classList` est une propriété de l'élément `body`.
`toggle()` est une méthode de la propriété `classList`.
On accède à la propriété d'un objet grâce à l'opérateur `.`
`toggle()` ajoute ou retire la classe selon qu'elle est présente ou pas.

EXERCICE 9

Masquage et affichage du menu

```
nav {
    display: none;
}

body.nav-on nav {
    display: block;
}
```

On ne gère que l'interactivité (événement click) en JS. Le changement d'affichage est laissé à CSS grâce à la modification d'une classe.

Et en JS, comme prédemment (ou presque)

```
var button = document.querySelector('.navToggle');

button.addEventListener('click', function () {
    var body = document.querySelector('body');
    body.classList.toggle('nav-on');
});
```

Pour des présentations HTML / CSS / JS

- Reveal.js utilisé pour cette présentation
- Impress.js

Choses à voir pour aller plus loin

- NodeJS, JavaScript hors du navigateur
- Markdown, mise en forme texte
- Sass, préprocesseur CSS (à l'instar de Less)
- Yarn, gestionnaire de dépendances pour le web
- Gulp / Grunt / Webpack