esprit
Se former autrement

Chapitre 3: JavaScript

Année universitaire
2020-2021

CONFÉRENCE DES GRANDES ÉCOLES

cdio
CONCEIVE DESIGN IMPLEMENT OPERATE

AUF Agence UNIVERSITAIRE DE LA FRANCOPHONIE
IFGU INSTITUT DE LA FRANCOPHONIE POUR LA GOUVERNANCE UNIVERSITAIRE

United Nations Educational, Scientific and Cultural Organization
UNESCO Chair "Project-based learning" ESPRIT School of engineering, Tunisia

Plan

- Introduction
- HTML & JavaScript
- Syntaxe
- Objets
- Tableaux
- Fonctions
- DOM

Objectifs

- Manipuler le DOM
- Ecrire un script JS en utilisant les fonctions prédéfinis, événement…

Prérequis

✓ HTML

Introduction

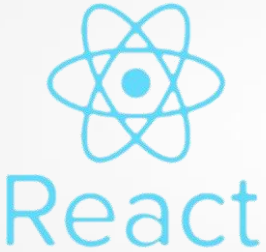
- **Javascript** permet de rendre **interactif** un site internet développé en HTML.
- **Javascript** est standardisé par un comité spécialisé, l'ECMA (European Computer Manufactures Association).
- **JavaScript** est un langage de programmation:
  - **scripté** (interprété) - pas de compilateur à proprement parler.
  - **côté client** - s'exécute dans un navigateur en général (il existe des environnements côté serveur : NodeJS).
  - **asynchrone** - plusieurs « morceaux » peuvent s'exécuter en parallèle.

Introduction

- JavaScript, permet :
  - de spécifier des changements sur le document :
    - sur le contenu: la structure, le style…
    - en interceptant des évènements: souris, clavier, …
  - échanger avec un serveur (AJAX)
  - dessiner (canvas - bitmap - ou svg - vectoriel)
  - se géolocaliser
  - enregistrer localement du contenu
  - jouer des fichiers audio ou vidéo

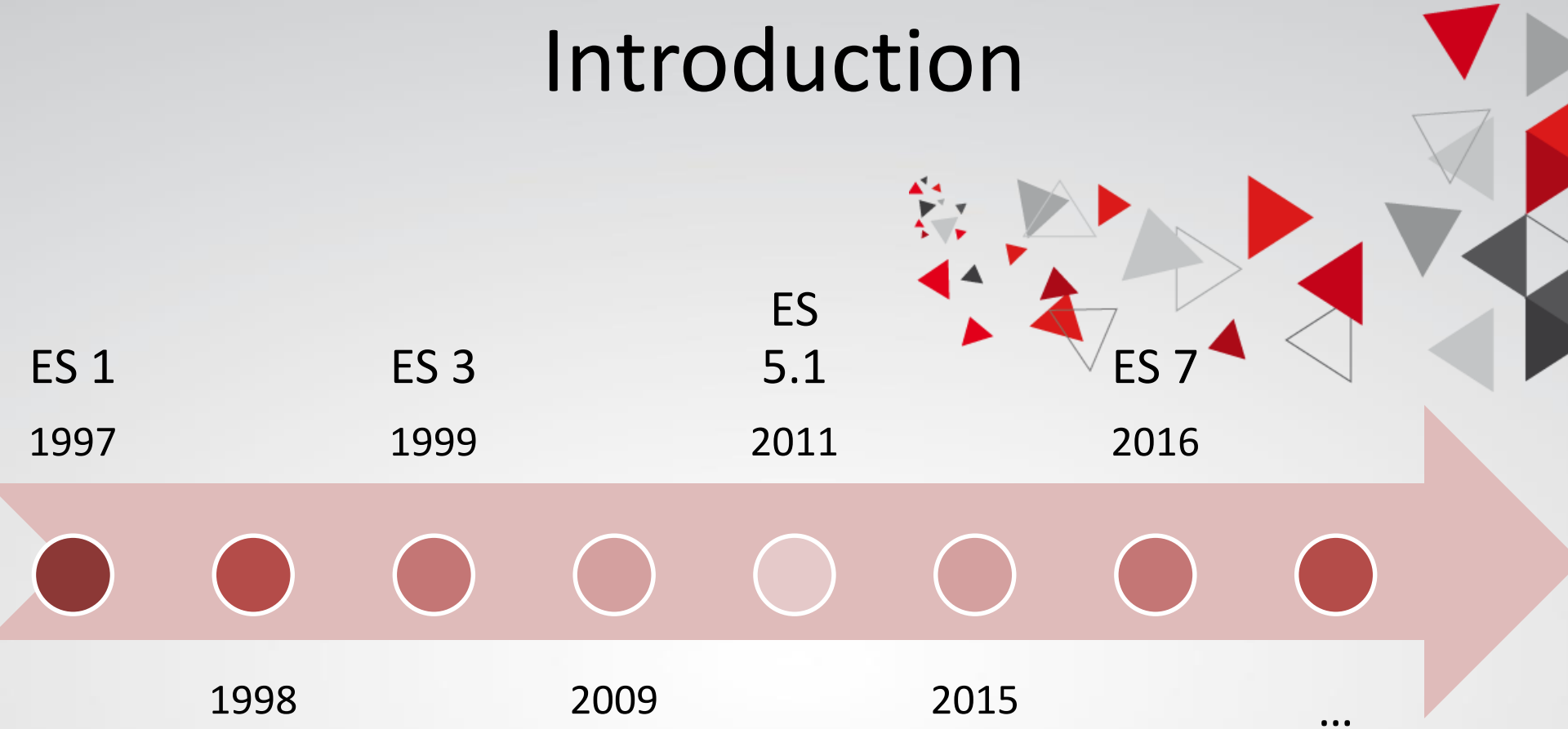
Introduction

- Utilitaires JavaScript



- On peut créer une base logicielle entièrement codée en JavaScript qui peut tourner sur **MEAN** → **M**ongoDB, **E**xpress.js, **A**ngular.js, et **N**ode.js au lieu de **LAMP** → **L**inux, **A**pache, **M**ySQL, **P**HP.

Introduction



A partir de 2015, la mise à jour est annuelle.

- ✓ ES6 est le standard supporté par la plupart des nouvelles versions de navigateurs. https://kangax.github.io/compat-table/es6/

HTML & JavaScript

Console développeur

- Firefox: Ctrl+Shift+K
- Chrome / Edge: Ctrl+Shift+I

Balise HTML

- Cliquez-moi !

Code HTML (interne)

- <script> … </script>

Fichier séparé (externe)

- <script src="script.js"></script>

Exemple

```html
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Document</title>
    </head>
    <body>
        <p id="item">Mon premier paragraphe</p>
        <p>Mon deuxième paragraphe</p>

        <script>
            // Instructions
        </script>
</body></html>
```

- Tester le code suivant:

```
alert("Bonjour tout le monde!!!");
```

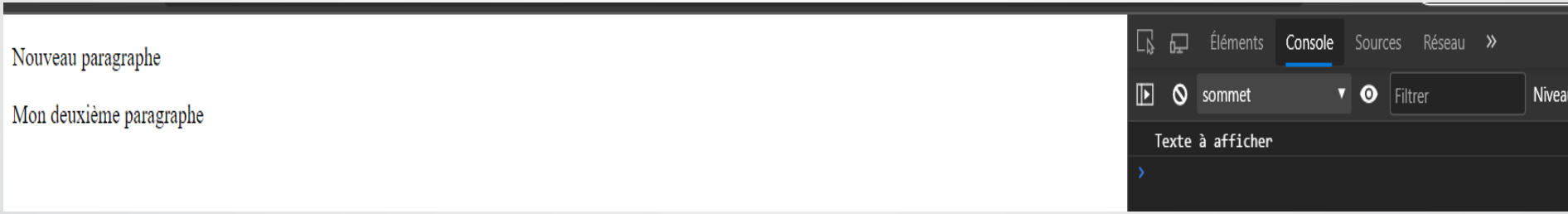
```
console.log("Texte à afficher");
```

```
document.getElementById('item').innerHTML = "<p> Nouveau paragraphe</p>";
```

Exemple

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Document</title>
    </head>
    <body>
        <p id="item">Mon premier paragraphe</p>
        <p>Mon deuxième paragraphe</p>

        <script>
            // Instructions
        </script>
</body></html>
```

- Résultat:



Syntaxe

Les commentaires

- Par ligne:

```
// un commentaire sur une ligne
```

- Par Bloc:

```
/* un commentaire plus
long sur plusieurs lignes
*/
```

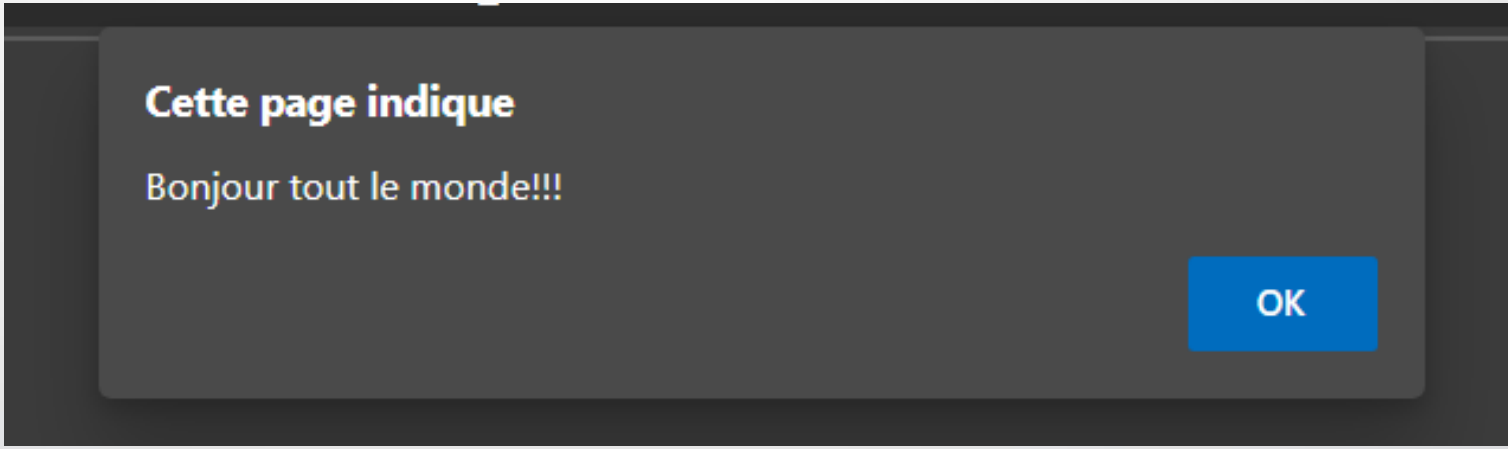
- Remarque:

```
/* Par contre on ne peut pas /* imbriquer des commentaires */ SyntaxError */
```

Syntaxe

Les boîtes de dialogue

```
<script>

    alert("Bonjour tout le monde!!!");

</script>
```

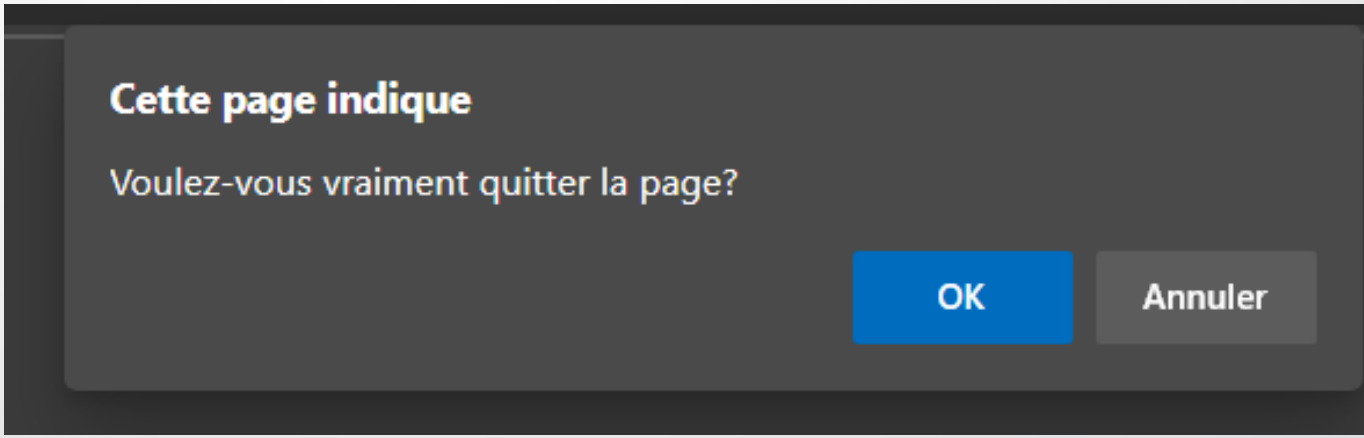


Syntaxe

Les boîtes de dialogue

```
<script>

    confirm("Voulez-vous vraiment quitter la page?");

</script>
```

confirm: renvoie true ou false

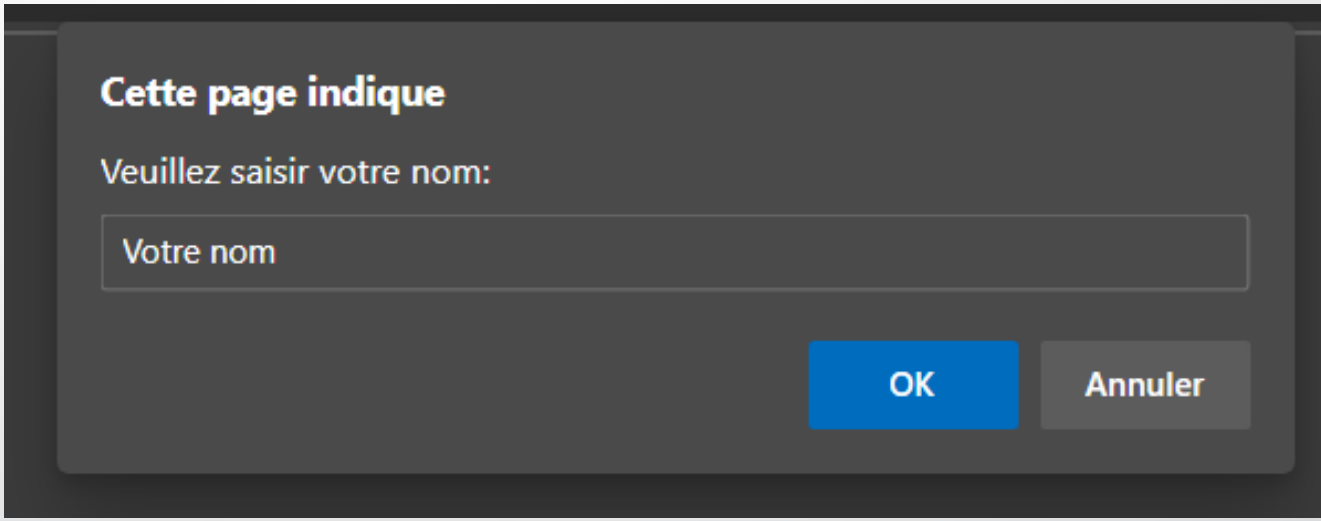


Syntaxe

Les boîtes de dialogue

```
<script>

    prompt("Veuillez saisir votre nom:", "Votre nom");

</script>
```

prompt: renvoie la valeur saisie ou Null



Syntaxe

Variable

- **JavaScript** est un langage pauvrement typé, il n'est pas indispensable de déclarer préalablement le type de variable.

- Il existe trois types de déclarations de variable en JavaScript.
  - **var:** déclare une variable, en initialisant sa valeur éventuellement.
  - **let:** déclare une variable dont la portée se limite au bloc courant.
  - **const:** déclare une constante, dont la portée se limite au bloc courant et accessible en lecture seule.

Syntaxe

Variable

- Le nom d'une variable doit commencer par:
  - Lettre
  - Tiret bas (_)
  - Symbole dollar ($)
- Les caractères qui suivent peuvent inclure les lettres minuscules et/ou majuscules et les chiffres.
- **Par convention:**
  - Noms de variables et fonctions écrits en CamelCase
  - Noms de constantes écrits en majuscule
- **Remarque:**
  - JavaScript est sensible à la casse: maVariable est différente de MaVariable.

Syntaxe

Variable

- Typage dynamique

```
var maVariable = 2020;     // maVariable est un nombre
    maVariable = "hello"; // maVariable est une chaîne de caractères
    maVariable = true;     // maVariable est un booléen
```

- Types de données
  - Primitifs:
    - Booléen
    - Null
    - Undefined
    - Nombre
    - String
  - Objet

Syntaxe

Variable

- Evaluation

```
var a;
console.log("La valeur de a est " + a); // La valeur de a est undefined

console.log("La valeur de b est " + b); // La valeur de b est undefined
var b; // La déclaration de la variable est "remontée" (hoisting)

console.log("La valeur de x est " + x); // signale une exception ReferenceError
let x, y;
console.log("La valeur de y est " + y); // La valeur de y est undefined
```

Syntaxe

Variable

- Portée

```
if (true) {
    var z = 2020;
}
  console.log(z); // z vaut 2020

if (true) {
    let y = 'Hello!!';
}
console.log(y); // Uncaught ReferenceError: y is not defined
```

Syntaxe

Variable

- L'opérateur

typeof renvoie le type d'une variable.

```
var X, nom = prompt("Veuillez saisir votre nom:", "Votre nom");

console.log(typeof X);
// expected output: "undefined"

X = 2020;
console.log(typeof X);
// expected output: "number"

console.log(typeof nom);
// expected output: "string"

console.log(typeof true);
// expected output: "boolean"

console.log(typeof Symbol('test'));
// expected output: "symbol"

console.log(typeof null);
// expected output: "object"
```

Syntaxe

Opérateurs

Opérateur	Explication	Symbole
Addition	• Additionner des nombres (1+5;) • Concaténer des chaînes (“Hello ” + “World! “;)	+
Arithmétique	• Les opérateurs mathématiques de base: soustraction, division et multiplication • Opérateur de puissance (**) • Reste de la division (%)	-, /, * ** %
Assignation	• Affecte une valeur à une variable • Affectation après addition, soustraction, division, multiplication • Affectation du reste (x %= y ➔ x = x % y)	= +=, -=, /=, *= %=
Négation	• Non Logique: Renvoie la valeur opposé (false devient true) • Non Unaire: Renvoie l’opposé de l’opérande • Non binaire: Inverse les bits de l’opérande (~1 ➔ 0)	! - ~

Syntaxe

Opérateurs

Opérateur	Explication	Symbole
Incrémentation Décrémentation	Ajoute / soustrait une unité à son opérande • **Suffixe:** renvoie la valeur avant l'incrémentation / décrémentation • **Préfixe:** renvoie la valeur après l'incrémentation / décrémentation	 X++, X-- ++X, --X
Relationnel	• Permet de comparer deux opérandes et de renvoyer une valeur booléenne	<, >, <=, >=
Binaire	• ET binaire (AND) • OU binaire (OR)	& |
Logique	• ET logique (AND) • OU logique (OR)	&& ||
Egalité	• (in)égalité faible • (in)égalité stricte	!==, === !=, ==

Structures Conditionnelles

- if (condition) {
  // instructions
  }
  else {
  // instructions
  }

```
let X = 'WEB'

if (X === 'web') {
    alert('Web Development');
}
else {
    alert('Others');
}
```

Structures Itératives

- for ([exp. Initiale]; [Condition]; [incrément]) {
  // instructions
  }

```
for (var compteur = 0; compteur < 5; compteur++){
    console.log("Compteur = " + compteur);
}
```

- do {
  // instructions
  } while (condition);

- while (condition) {
  // instructions
  }

```
Compteur = 0
Compteur = 1
Compteur = 2
Compteur = 3
Compteur = 4
>
```

Objets

Définition

- Un objet est une entité à part entière qui possède des propriétés.
- Une propriété est une association entre un nom (clé) et une valeur.
- On accède à une propriété en utilisant les notations suivantes:

  nom_Objet.nom_Propriété

  nom_Objet[nom_Propriété]

Objets

Définition

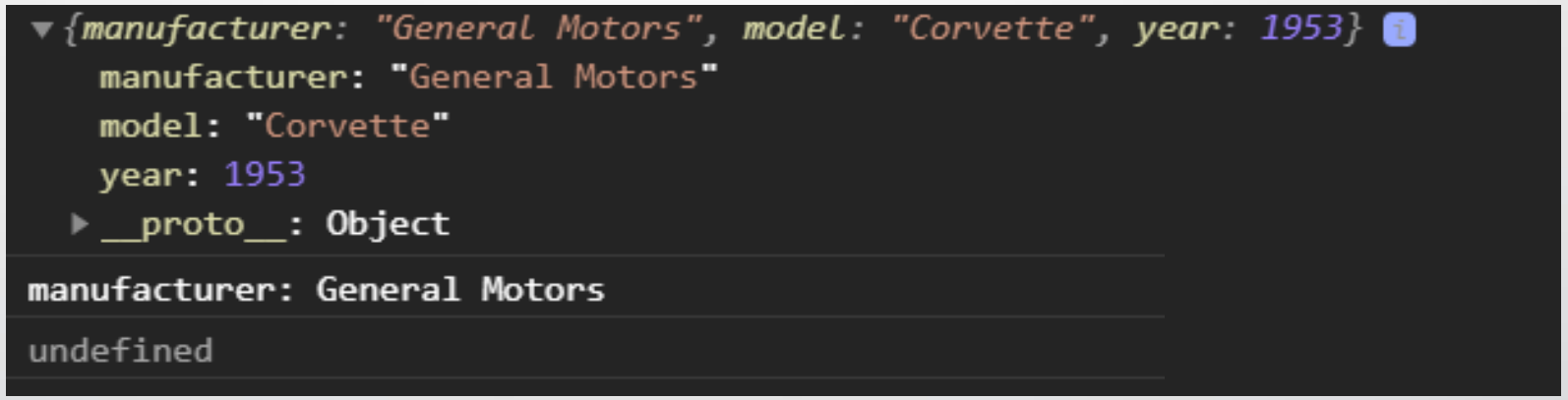
- La création d'un objet peut se faire :

```
var nom_Objet = new Object();
var nom_Objet = { prop_1: valeur_1,
                  'prop_2': valeur_2,
                   prop_3: function() { …},
                  …,
                  10: valeur_10
                 };

Object.create(nom_Objet);
```

Objets

Exemple 1

```
var car = new Object();
car.manufacturer = "General Motors";
car.model = "Corvette";
car.year = 1953;

console.log(car);
console.log('manufacturer: ' + car['manufacturer']);
console.log(car[2]);
```



Objets

Exemple 2

```
var transportation = {
    type: "car",
    afficherType : function() {
        console.log('transportation type: ' + this.type);
    }
}

var t1 = Object.create(transportation);
t1.afficherType();
```

```
transportation type: car
```

Tableau

- JavaScript ne possède pas de type particulier pour représenter un tableau de données.
- Utiliser l'objet natif **Array** ainsi que ses méthodes pour manipuler des tableaux.
- Pour créer un tableau:
  - var arr = new Array(élément0, élément1, ..., élémentN);
  - var arr = Array(élément0, élément1, ..., élémentN);
  - var arr = [élément0, élément1, ..., élémentN];

Tableau

- Pour créer un tableau sans aucun élément initial et de longueur non nulle (**l**):
  - var arr = new Array (**l**);
  - var arr = Array (**l**);
  - var arr = [];
    arr.length = **l**;

  Rq: **l** doit être un nombre

Tableau

Exercice 1

- Tester les instructions suivantes:

```
var arr1 = [5];
var arr2 = Array (5);
var arr3 = Array (5.2);
var arr4 = Array.of(5);

console.log(arr1);
console.log(arr2);
console.log(arr3);
console.log(arr4);
```

Tableau

Exercice 2

- Tester les instructions suivantes:

```
var arr = [];
arr[0] = "Un";
arr[1] = 2;
arr[2] = 3.14;

console.log(arr);
console.log(arr[0]);
console.log(arr["length"]);
arr.length = 0;
console.log(arr);
arr.length = 3;
console.log(arr);
```

Tableau

Exercice 3

- Tester les instructions suivantes:

```
var arr = [];
arr[0] = "Un";
arr[1] = 2;
arr[2] = 3.14;

for (var i = 0; i < arr.length; i++) {
    console.log(arr[i]);
}

arr.forEach(function(nb) {
    console.log('nb: ' + nb);
});

arr.forEach(nb => console.log('nb: ' + nb));
```

Fonctions

Syntaxe

- function nom_fonction (arg1, arg2, …) {

  // instructions

  }

```
var factorielle = function fac(n) { return n < 2 ? 1 : n * fac(n - 1) };

console.log(factorielle(3)); // expected output: 6
```

Fonctions

Fonctions prédéfinis

```
var t = [1, 2, 1, 2, 1]
var ch = 'Esprit'
```

Fonction	Explication	Exemple
concat	Permet de fusionner deux ou plusieurs tableaux et renvoie le nouveau tableau.	t1 = t.concat(3, 4); // [1, 2, 1, 2, 1, 3, 4]
join	Permet de fusionner les éléments du tableau en une chaîne de caractères.	ch = t.join(' - '); // 1 - 2 - 1 - 2 - 1
sort	Trie les éléments du tableau dans le même tableau.	t.sort(); // [1, 1, 1, 2, 2]
reverse	Transpose les éléments du tableau dans le même tableau.	t.reverse(); // [2, 2, 1, 1, 1]
pop	Permet de retirer le dernier élément du tableau et renvoie cet élément.	x = t.pop(); // x = 1
slice	extrait une partie du tableau et renvoie un nouveau tableau avec le reste	t.slice(1, 3); // [2, 1, 1] ch.slice(3, 5);
indexOf	Recherche la valeur et renvoie l'indice du premier élément correspondant.	pos = t.indexOf(2); ch.indexOf('rit') // 3
lastIndexOf	Recherche la valeur à partir de la fin du tableau.	pos = t.lastIndexOf(2);

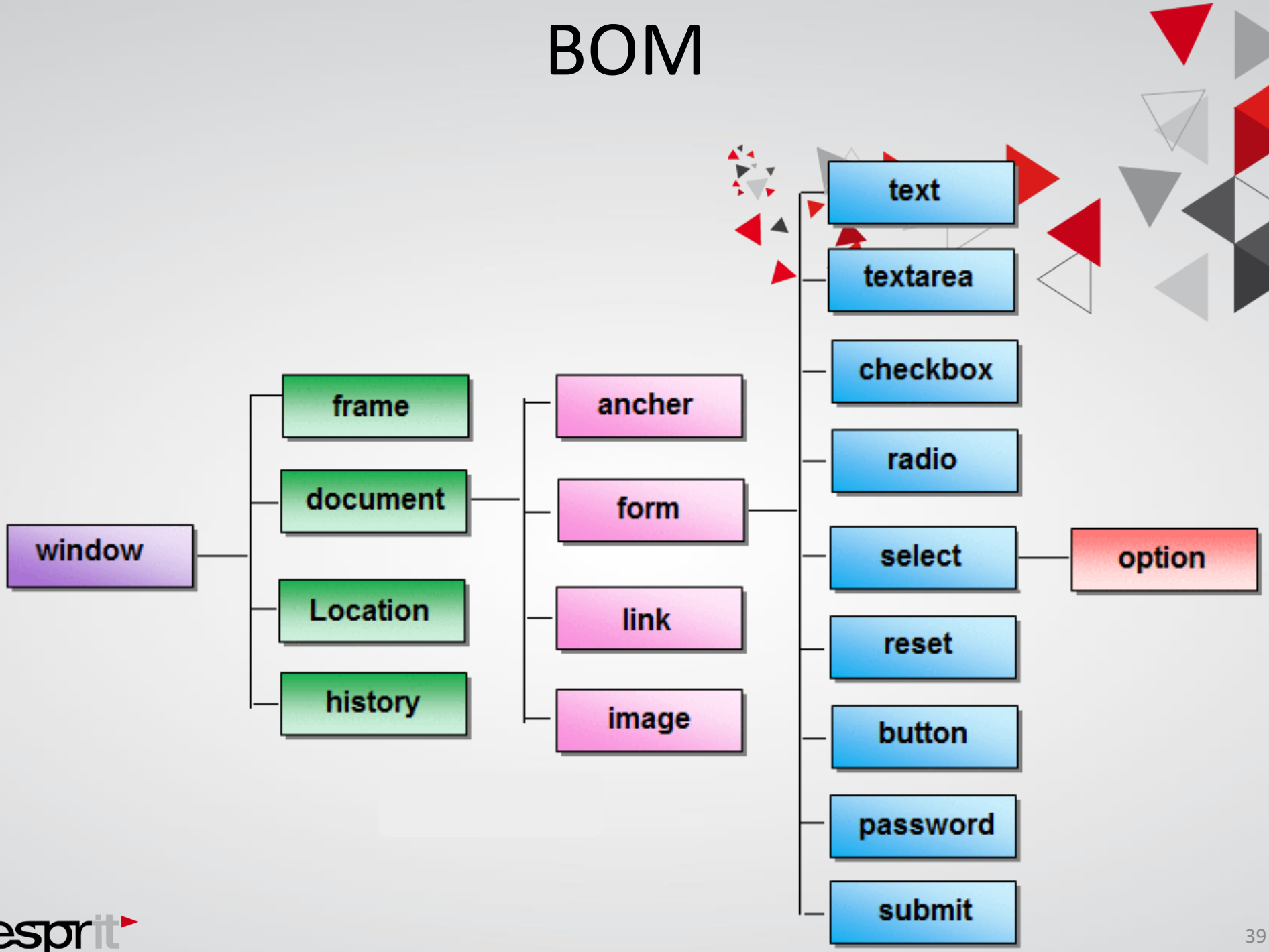
Fonctions

Fonctions prédéfinis

var ch = ‘ Hello world ’

Fonction	Explication	Exemple
trim	Retire les blancs en début et en fin de chaîne.	ch = ch.trim() // ‘Hello world’
toLowerCase toUpperCase	Convertissent tous les caractères respectivement en minuscules ou en majuscules.	ch = ch.toLowerCase(); ch = ch.toUpperCase();
charAt	Renvoie une nouvelle chaîne contenant le caractère à la position indiquée en paramètre.	ch.charAt(0); // H
endsWith	Renvoie un booléen indiquant si la chaîne se termine par la chaîne fournie en paramètre.	ch.endsWith(‘!’); // false
Includes	Détermine si une chaîne est contenue dans une autre chaîne et renvoie un booléen.	ch.includes(‘world’); // true
startsWith	Renvoie un booléen indiquant si la chaîne commence par la chaîne fournie en paramètre.	ch.startsWith(‘Hello’); // true
substring	extrait une partie de la chaîne à partir du premier indice jusqu’au second indice (non compris).	ch.substring(0, 5); // Hello

BOM


window
frame
document
Location
history
ancher
form
link
image
text
textarea
checkbox
radio
select
option
reset
button
password
submit

DOM

- Document Object Model
  - Structure arborescente créée par le navigateur.
  - Facilite l'accès à la structure HTML.

- Le navigateur utilise le DOM pour appliquer le style et corriger les éléments.
- Le développeur utilise le DOM pour manipuler la page.

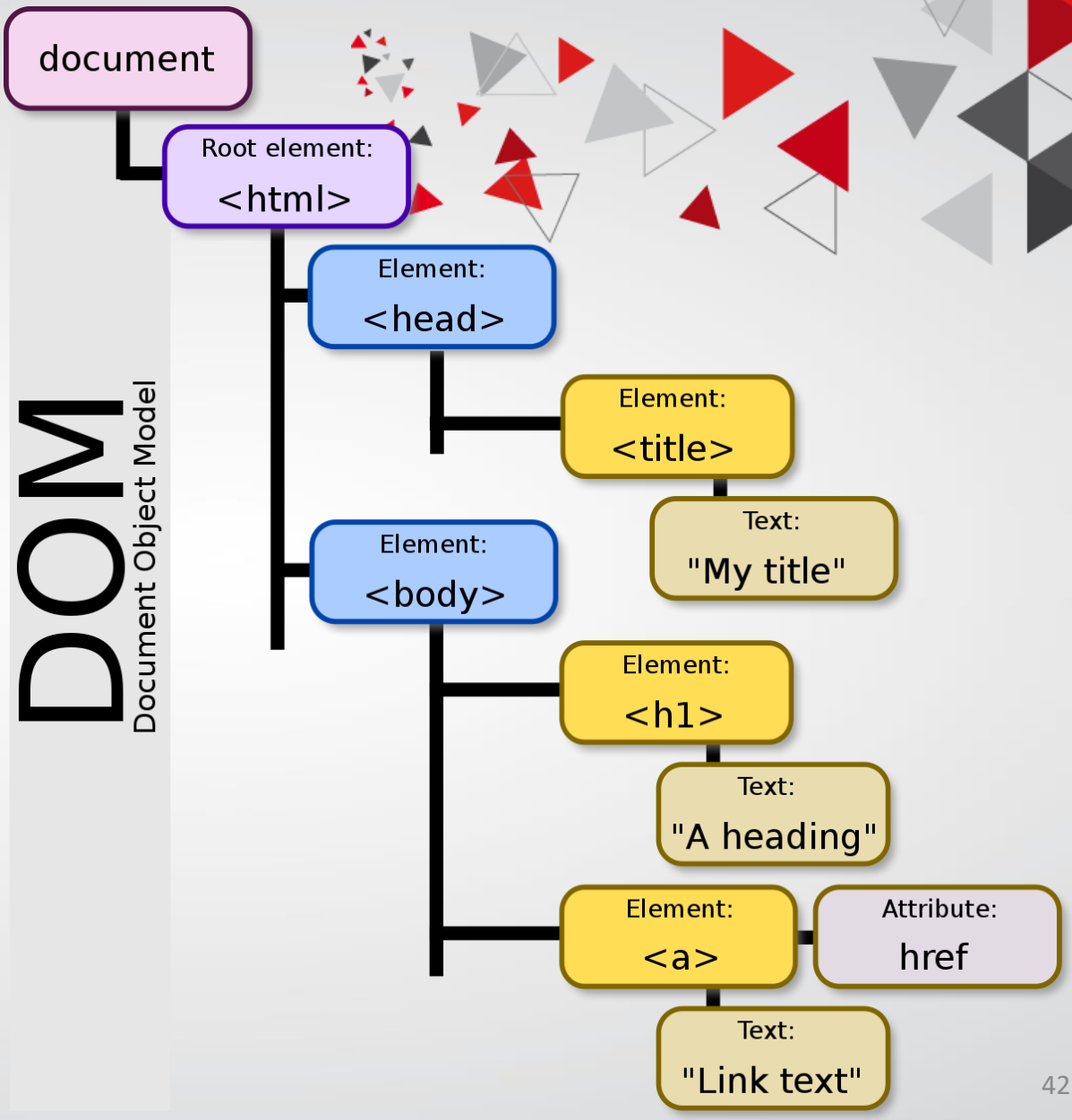
DOM

- Exemple

```
<html Lang="en">
    <head>
        <title>My title</title>
    </head>
    <body>
        <h1>A heading</h1>
        <a href="">Link text</a>
    </body>
</html>
```

DOM

- Le DOM correspondant au code précédent ressemble à ça ➔
- Chaque élément et donnée texte forme un nœud (element node) de l'arbre.



DOM

- Propriétés d'un nœud

Propriétés	Explication
childNodes	La liste des nœuds enfants
firstChild	Le premier nœud enfant
lastChild	Le dernier nœud enfant
nextSibiling	Le prochain nœud du même niveau
parentNode	Le nœud parent
previousSibiling	Le nœud précédent du même niveau
nodeName	Le nom du nœud
nodeValue	La valeur ou contenu du nœud
nodeType	Le type du nœud
innerHTML	Le contenu littéral du nœud

DOM

- Méthodes d'un nœud

Méthodes	Explication
createElement()	Permet de créer un nouvel élément HTML dans le document
createTextNode()	Permet de créer un nœud texte
appendChild()	Permet d'ajouter l'élément créé au document comme le dernier nœud enfant de l'élément parent.
removeChild()	Permet de supprimer un nœud.

DOM

- Manipuler le DOM

Méthodes	Explication	Syntaxe
querySelector()	Retourne le premier élément dans le document correspondant au sélecteur(s) spécifié(s) ou null	element = document.querySelector (sélecteur);
querySelectorAll()	Retourne uncontenant la liste des éléments du document qui correspondent au sélecteur(s) spécifié(s)	element = document.querySelectorAll (sélecteur);
getElementById()	Renvoie un objet (élément) qui représente l'élément portant l'id spécifié.	element = document.getElementById (id);

DOM

- Manipuler le DOM

Méthodes	Explication	Syntaxe
getElementsByTagName()	Renvoie un tableau qui contient tous les éléments ayant un type donné.	elements = document.getElementsByTagName (name);
getElementsByClassName()	Renvoie un tableau qui contient tous les éléments portant le nom de classe spécifié.	elements = document.getElementsByClassName (className);
getElementsByName()	Renvoie un tableau qui contient tous les éléments portant le « name » spécifié.	elements = document.getElementsByName (name);