Fadi ALOUANI

BTS SIO 1
groupe 1
Atelier
03/12/25

Atelier 11 : Javascript 1/2

Sommaire

Introduction

Dans le cadre de mes premiers travaux pratiques en JavaScript, j'ai découvert les bases du langage et la manière d'intégrer des scripts dans une page HTML. J'ai appris à utiliser la balise <script>.
Ce TP marque le début de mon apprentissage du fonctionnement et des règles fondamentales de JavaScript.

Fadi ALOUANI

Fondamentaux 1

Exercice : Hello word

Créez une page qui affiche un message “Je suis JavaScript!”.

Assurez-vous simplement que cela fonctionne.

Pour afficher une alertes j'ai Crée un fichier nommé index.html, puis j'ai utiliser ce code :

```html
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Test JavaScript</title>
</head>
<body>

<script>
    alert("Je suis JavaScript!");
</script>

</body>
</html>
```

voici ce que fait ce code sur le navigateur :

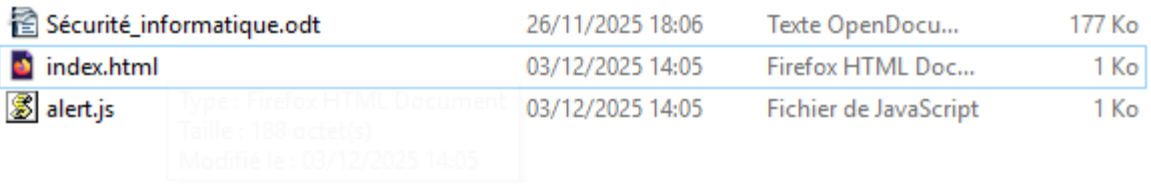
Fadi ALOUANI

Prendre la solution de l'exercice précédent Afficher une alerte. Modifiez-le en extrayant le contenu du script dans un fichier externe `alert.js`, résidant dans le même dossier.

Ouvrez la page, assurez-vous que l'alerte fonctionne.

J'ai donc crée un ficher externe"alert.js" et je l'ai placer dans le meme dossier que mon index.html

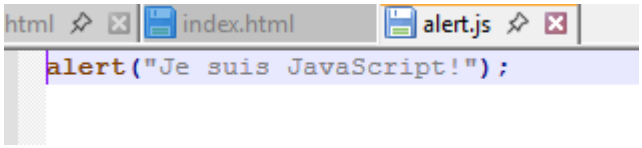
2 -



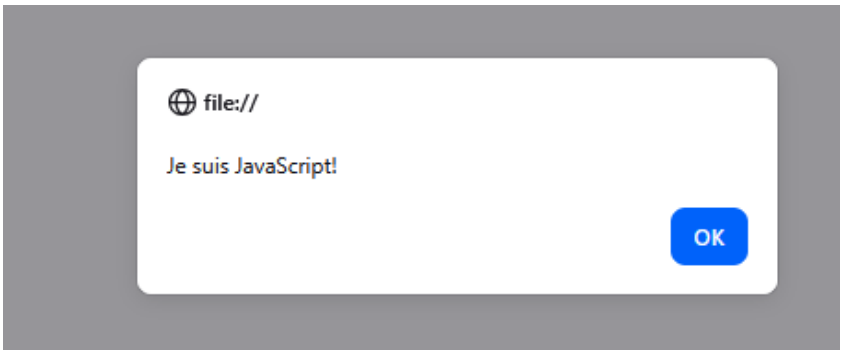
La commande que que j'ai utiliser dans mon HTML pour appeler le fichier externe alert.js

```
<body>

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

</body>
```

J'ai ensuite rempli le fichier alert.js avec "Je suis JavaScript!".



Le pop up s'affiche comme sur l'exercice précédant cela a donc bien fonctionner.



Fadi ALOUANI

Fondamentaux 2

Exercice : Variable

Déclarez deux variables : `admin` and `name`.

Assignez la valeur `"John"` à `name`.

Copiez la valeur de `name` à `admin`.

Afficher la valeur de `admin` en utilisant `alert` (devrait afficher “John”).

Pour cet exercice, j'ai réutilisé le fichier alert.js voici comment j'ai déclaré les variables et comment je leur ai assigné les valeurs.

```
let admin;
let name = "John";

admin = name;

alert(admin); // Affiche "John"
```

Voici le résultat quand j'ouvre l' HTML :

Comme decrit sur le tp le fichier **alert.js** affiche John.

Fadi ALOUANI

Exercice : Les type de données

Créez la variable avec le nom de notre planète.

Comment nommeriez-vous une telle variable ?

Créez la variable pour stocker le nom du visiteur actuel.

Comment nommeriez-vous cette variable ?

Voici l' HTML qui appelle le pop-up alert.js

lex.html alert.js

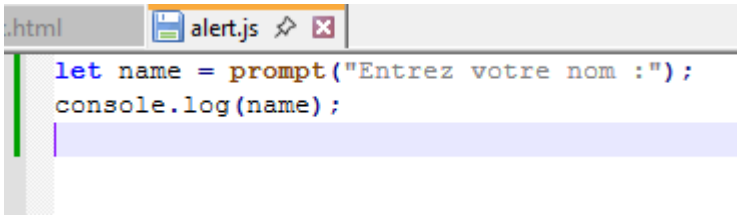
```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Test JavaScript externe</title>
</head>
<body>

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

</body>
</html>
```

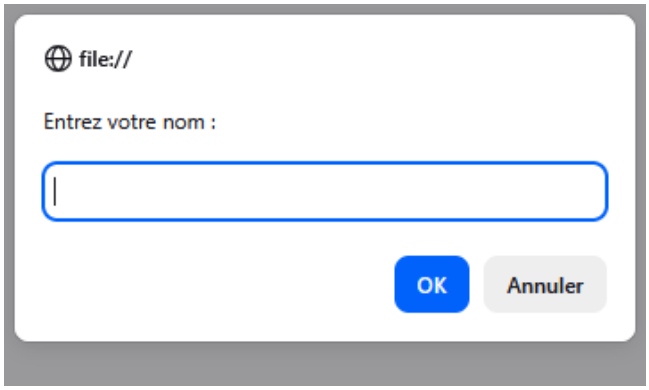
tml alert.js

```
let planetName = "Terre";
let visitorName = "Karim";

let admin = planetName;

alert(admin);
```

Exercice interaction : alert,prompt,confirm

Une simple page
Créez une page Web qui demande un nom et l'affiche





Exercice : opérateur de base , mathématique

Les formes postfixes et préfixes

Quelles sont les valeurs finales de toutes les variables `a`, `b`, `c` et `d` après le code ci-dessous ?

```
let a = 1, b = 1;

let c = ++a; // ?
let d = b++; // ?
```

Explication :

- ++a est préfixe : on incrémente d'abord, puis on retourne la nouvelle valeur.
- b++ est postfixe : on retourne la valeur avant l'incrémentation.

Résultats :

- a devient 2
- c reçoit 2
- d reçoit 1
- b devient 2

Résultat d'affectation

Quelles sont les valeurs de `a` et `x` après le code ci-dessous ?

```
let a = 2;

let x = 1 + (a *= 2);
```

Explication :

a *= 2 est équivalent à a = a * 2.
Donc :

- d'abord : a = 2 * 2 = 4
- ensuite : x = 1 + 4 = 5

Réponse :

- a = 4
- x = 5

Les conversions de type

Quels sont les résultats de ces expressions ?

```
"" + 1 + 0
"" - 1 + 0
true + false
6 / "3"
"2" * "3"
4 + 5 + "px"
"$" + 4 + 5
"4" - 2
"4px" - 2
"  -9  " + 5
"  -9  " - 5
null + 1
undefined + 1
" \t \n" - 2
```

Corrigez l'addition

Voici un code qui demande à l'utilisateur deux nombres et affiche leur somme.

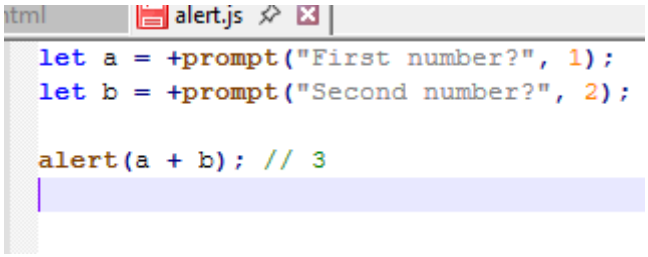
Cela ne fonctionne pas correctement. La sortie dans l'exemple ci-dessous est `12` (pour les valeurs d'invite par défaut).

Pourquoi ? Réparez-le. Le résultat doit être `3`.

```
let a = prompt("First number?", 1);
let b = prompt("Second number?", 2);

alert(a + b); // 12
```

Le problème est le prompt() renvoie une chaîne, donc "1" + "2" = "12".

Correction : convertir en nombre



Exercice : Comparaison

Comparaisons

Quel sera le résultat pour les expressions suivantes :

```
5 > 4
"apple" > "pineapple"
"2" > "12"
undefined == null
undefined === null
null == "\n0\n"
null === +"\n0\n"
```

1.5 > 4
true
Car 5 est plus grand que 4.

2."apple" > "pineapple"
false
Comparaison lexicographique (alphabetique) :

- on compare lettre par lettre
- 'a' vient avant 'p'

Donc "apple" est considéré plus petit.

3. "2" > "12"
✔ true
Comparaison alphabétique (car ce sont des chaînes) :

- compare le premier caractère
- "2" vs "1" → "2" est plus grand

Donc "2" > "12".

4. undefined == null
true
La règle spéciale de JavaScript :
null et undefined sont égaux avec ==
Mais à rien d'autre.

5. undefined === null
false
=== compare le type et la valeur :

- undefined : type undefined
- null : type object

Types différents → false.

6. null == "\n0\n"
false
== ne convertit null qu'envers undefined.
Le string "\n0\n" devient "0" → puis 0.
Mais :
null == 0 est false
Donc résultat : false.

7. null === +"\n0\n"
D'abord calculons +"\n0\n" :

- "\n0\n" devient "0" → converti en nombre = 0

Donc comparaison :
null === 0
false
Car types différents.

Exercice : conditionnelle if

if (une chaîne de caractères avec zéro)

Est-ce que `alert` sera affiché ?

```
if ("0") {
  alert( 'Hello' );
}
```

Fondamentaux 3

Exercice : BOUCLES

Dernière valeur de boucle

Quelle est la dernière valeur affichée par ce code ? Pourquoi ?

```
let i = 3;

while (i) {
  alert( i-- );
}
```

i est initialisé à 3 la boucle while(i) continue tant que i est vrai (en JavaScript, tant que i est différent de 0).

alert(i--) affiche la valeur de i avant de la soustraire .

La dernière valeur affichée est donc 1.

Parce que alert(i--) affiche la valeur avant la soustraction. Quand i vaut 1, il affiche 1, puis i devient 0 et la boucle s'arrête

Quelles valeurs affiche la boucle while ?

A votre avis, quelles sont les valeurs affichées pour chaque boucle ? Notez-les puis comparer avec la réponse.

Les deux boucles affichent-elles les mêmes valeurs dans l'`alert` ou pas ?

1.

Le préfixe sous forme `++i` :

```
let i = 0;
while (++i < 5) alert( i );
```

2.

Le postfixe sous forme `i++` :

```
let i = 0;
while (i++ < 5) alert( i );
```

1)**. Boucle avec pré-incrément ++i :**

```
let i = 0;
while (++i < 5) alert(i);
```

Explication :

- ++i : pré-incrément, donc i est incrémenté avant la comparaison.
- La condition ++i < 5 signifie : incrémenter i puis vérifier si i est inférieur à 5.

Étapes :

- i = 0 → ++i → i = 1 ; 1 < 5 ? Oui → alert(1)
- i = 1 → ++i → i = 2 ; 2 < 5 ? Oui → alert(2)
- i = 2 → ++i → i = 3 ; 3 < 5 ? Oui → alert(3)
- i = 3 → ++i → i = 4 ; 4 < 5 ? Oui → alert(4)
- i = 4 → ++i → i = 5 ; 5 < 5 ? Non → boucle s'arrête

Valeurs affichées :
1, 2, 3, 4

2). **Boucle avec post-incrément i++ :**

```
let i = 0;
while (i++ < 5) alert(i);
```

Explication :

- i++ : post-incrément, donc la comparaison se fait avec la valeur avant l'incrément, mais i est incrémenté après.
- La condition i++ < 5 signifie : vérifier si i est inférieur à 5, puis incrémenter i.

Étapes :

- i = 0 ; 0 < 5 ? Oui → alert(i) → i est maintenant 1, donc alert(1)
- i = 1 ; 1 < 5 ? Oui → alert(2)
- i = 2 ; 2 < 5 ? Oui → alert(3)
- i = 3 ; 3 < 5 ? Oui → alert(4)
- i = 4 ; 4 < 5 ? Oui → alert(5)
- i = 5 ; 5 < 5 ? Non → boucle s'arrête

Valeurs affichées :
1, 2, 3, 4, 5

Résumé :

Boucle	Valeurs affichées
while (++i < 5)	1, 2, 3, 4
while (i++ < 5)	1, 2, 3, 4, 5

Pourquoi la différence ?

- Le pré-incrément (++i) incrémente avant la comparaison, donc la première valeur testée est 1, et la boucle s'arrête avant d'atteindre 5.
- Le post-incrément (i++) compare d'abord, puis incrémente, donc la boucle va afficher

jusqu’à i = 5.

Quelles valeurs sont affichées par la boucle "for" ?

1. Boucle for avec post-incrément i++ :
for (let i = 0; i < 5; i++) alert(i);
Fonctionnement :

- Initialisation : i = 0
- Condition : i < 5
- Incrément : i++ (post-incrément, mais dans une boucle for, l’incrément est effectué après l’exécution du corps)
- Le corps : alert(i)

Valeurs affichées :

- i = 0 → alert(0), puis i devient 1
- i = 1 → alert(1), puis i devient 2
- i = 2 → alert(2), puis i devient 3
- i = 3 → alert(3), puis i devient 4
- i = 4 → alert(4), puis i devient 5
- i = 5 → condition i < 5 est fausse, boucle s'arrête

Donc la boucle affiche :
0, 1, 2, 3, 4

2. Boucle for avec pré-incrément ++i :
for (let i = 0; i < 5; ++i) alert(i);
Fonctionnement :

- La seule différence est que l’incrément est ++i (pré-incrément).
- Dans une boucle for, l’incrément est effectué après le corps de la boucle, donc que ce soit pré- ou post-incrément, ça ne change rien pour la valeur affichée dans le corps.

Valeurs affichées :

- i = 0 → alert(0), puis i devient 1
- i = 1 → alert(1), puis i devient 2
- i = 2 → alert(2), puis i devient 3
- i = 3 → alert(3), puis i devient 4
- i = 4 → alert(4), puis i devient 5
- i = 5 → condition i < 5 fausse, boucle s'arrête

Donc la boucle affiche :
0, 1, 2, 3, 4

Conclusion :
Les deux boucles affichent les mêmes valeurs :
0, 1, 2, 3, 4

Dans une boucle for, l'incrément se fait après l'exécution du corps de la boucle. La différence entre pré-incrément et post-incrément est donc insignifiante ici.

Remplacer "for" par "while"

Réécrivez le code en modifiant la boucle `for` en `while` sans modifier son comportement (la sortie doit rester la même).

```
for (let i = 0; i < 3; i++) {
  alert( `number ${i}!` );
}
```

Répéter jusqu'à ce que l'entrée soit correcte

Ecrivez une boucle qui demande un nombre supérieur à `100`. Si le visiteur saisit un autre numéro, demandez-lui de le saisir à nouveau.

La boucle doit demander un numéro jusqu'à ce que le visiteur saisisse un nombre supérieur à `100` ou annule l'entrée/entre une ligne vide.

Ici, nous pouvons supposer que le visiteur ne saisit que des chiffres. Il n'est pas nécessaire de mettre en œuvre un traitement spécial pour une entrée non numérique dans cette tâche.

Extraire des nombres premiers

Un nombre entier supérieur à 1 est appelé un Nombre premier s'il ne peut être divisé sans reste par rien d'autre que 1 et lui-même.

En d'autres termes, `n > 1` est un nombre premier s'il ne peut être divisé de manière égale par autre chose que `1` et `n`.

Par exemple, `5` est un nombre premier, car il ne peut pas être divisé sans reste par `2`, `3` et `4`.

Écrivez un code qui produit les nombres premiers dans l'intervall e 2 à n.

Pour `n = 10`, le résultat sera `2`, `3`, `5`, `7`.

P.S. Le code devrait fonctionner pour n'importe quel `n` et aucune valeur fixe ne doit être codé en dur.

1. Remplacer for par while

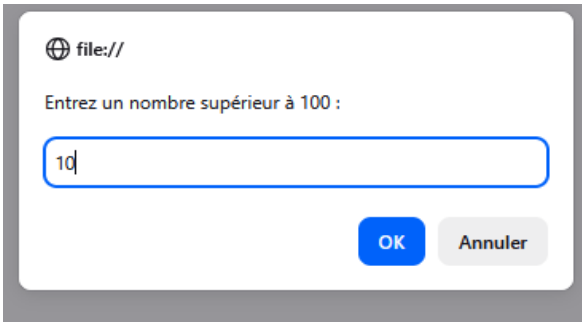
```
let i = 0;

while (i < 3) {
  alert(`number ${i}!`);
  i++;
}
```

2.Répéter jusqu'à ce que l'entrée soit correcte

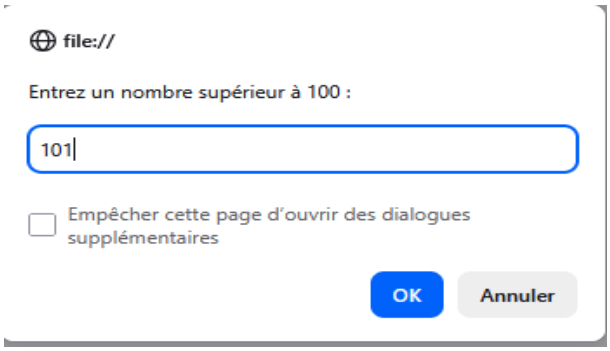
La commande utilisée :

```
let number;

do {
  number = prompt("Entrez un nombre supérieur à 100 :", "");
} while (number !== null && number <= 100);

alert("Entrée acceptée !");
```

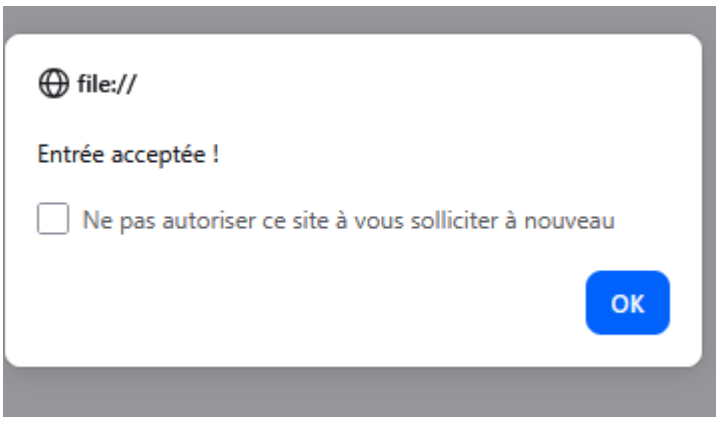
j'ai fait un test avec un nombre inférieur à 100 , quand j'appuie sur la touche entrer la page s'actualise et me redemande un nombre supérieur à 100, la commande fonctionne donc correctement.



test avec la bonne entrée



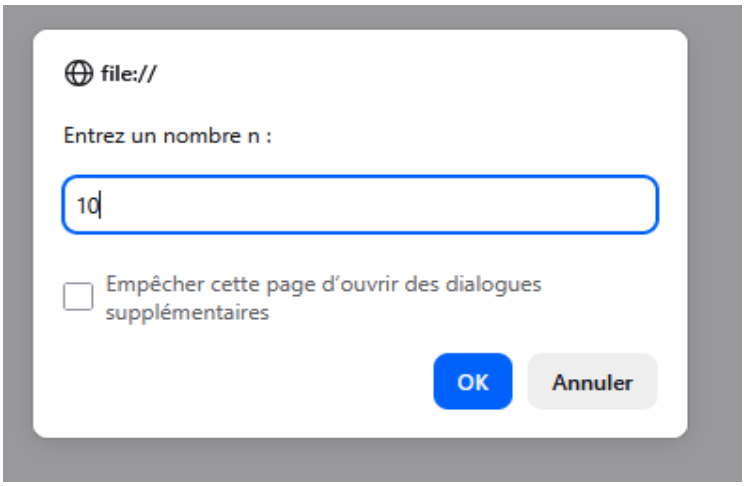
La boucle tant que l'entrer n'est pas correct fonctionnent, j'ai effectué le test avec une bonne entrée voici le résultat,j'ai également le même résultat quand j'annule l'opération quand j'utilise la touche échap.



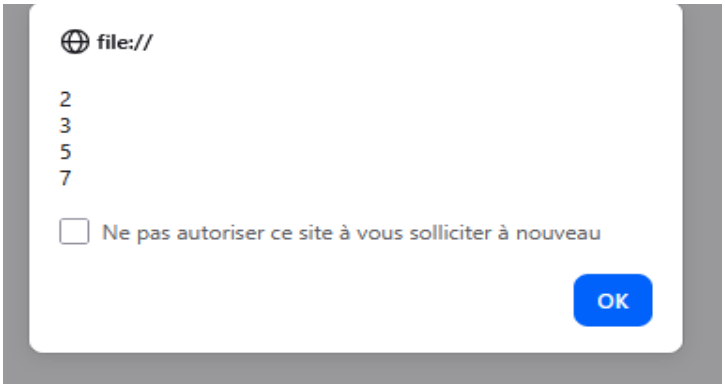
3.Extraire des nombres premiers

```
let n = +prompt("Entrez un nombre n :");
let result = "";

for (let i = 2; i <= n; i++) {
  let isPrime = true;

  for (let j = 2; j < i; j++) {
    if (i % j === 0) {
      isPrime = false;
      break;
    }
  }

  if (isPrime) {
    result += i + "\n";
  }
}

alert(result);
```

test avec 10



résultat



comme décrit sur le TP Pour
`n = 10`, le résultat sera `2,3,5,7`.

P.S. Le code devrait fonctionner pour n'importe quel `n` et aucune valeur fixe ne doit être codé en dur.

Exercice : Fonction fléchees

Résultat de OR

```
alert( null || 2 || undefined );
```

→ **Affiche : `2`**

Parce que `null || 2` renvoie `2`, et on ne regarde pas la suite.

Résultat des alertes OR

```
alert( alert(1) || alert(3) );
```

Déroulement :

1. `alert(1)` → affiche **1** et retourne `undefined`
2. OR continue donc avec `alert(3)` → affiche **3**, retourne `undefined`
3. L'expression entière vaut `undefined`
4. Donc le dernier `alert` affiche **undefined**

Résultat visible :

1
3
undefined

Résultat de AND

```
alert( 1 && null && 2 );
```

→ **Affiche : `null`**

Car AND s'arrête au premier falsy → `null`.

Résultat des alertes AND

```
alert( alert(1) && alert(2) );
```

Déroulement :

1. `alert(1)` → affiche **1**, retourne `undefined`
2. AND s'arrête immédiatement (undefined est falsy)
3. Expression vaut `undefined`
4. Dernier alert → affiche **undefined**

Résultat visible :

Fadi ALOUANI

1
undefined

Résultat de OR AND OR

```
alert( null || 2 && 3 || 4 );
```

Priorité :

`2 && 3` → **3**

Expression → `null || 3 || 4` → **3**

→ **Affiche : `3`**

Vérifiez la plage entre

Écrire une condition pour vérifier si `age` est entre 14 et 90 inclus.

```
age >= 14 && age <= 90
```

Conclusion

Ce TP m'a permis de comprendre les premiers éléments essentiels de JavaScript et d'apprendre à exécuter du code dans une page web. J'ai découvert les instructions, les points-virgules, les commentaires et le mode strict. Ces notions me serviront de base pour progresser vers des scripts plus complexes et développer mes compétences en programmation web.