



Objectif du module

À la fin de ce module, l'étudiant sera capable de :

- Comprendre ce qu'est une variable
- Stocker une information temporaire
- Utiliser des conditions (if / else)
- Adapter une interface en fonction d'une règle
- Créer un comportement dynamique simple

Table des matières

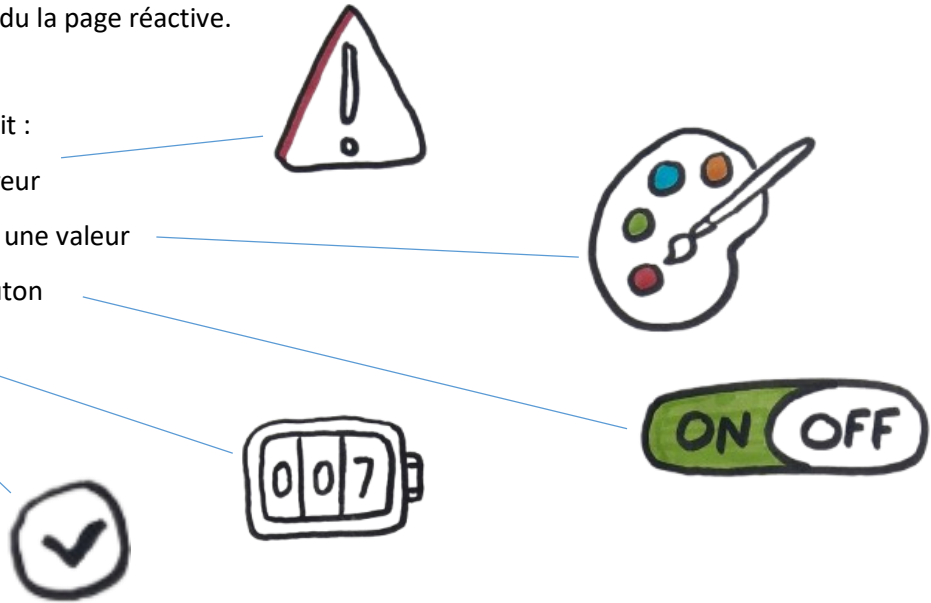
1. Pourquoi avons-nous besoin de logique ?	2
2. Les variables : mémoriser une information	2
Qu'est-ce qu'une variable ?	2
let vs const	2
Exemple concret : Compteur de clics.....	3
3. Les conditions : décider.....	3
Structure de base	3
Avec alternative	3
Exemple : message selon nombre de clics.....	4
Comparaisons importantes.....	4
Exemple : validation simple	5
Exemple : Mode clair / sombre	5
Exercice – panier.html	6
Exercice – verifpromo.html.....	7
Exercice - tva.html.....	7
Exercice - moyenne.html	8
Exercice - like.html (Like/Dislike)	8

1. Pourquoi avons-nous besoin de logique ?

Dans le module 2, nous avons rendu la page réactive.

Mais une interface intelligente doit :

- Afficher un message d'erreur
- Changer de couleur selon une valeur
- Activer/désactiver un bouton
- Compter des clics
- Valider un champ



Pour cela, il faut mémoriser et décider.

2. Les variables : mémoriser une information

Qu'est-ce qu'une variable ?

Une variable est un espace mémoire qui stocke une valeur.

Exemple :

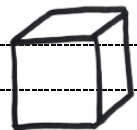
```
let score = 0;
```

Ici, score contient la valeur 0.

let vs const

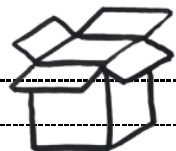
const : Valeur qui ne change pas.

```
const pi = 3.141592653589793;
```



let : Valeur qui peut évoluer.

```
let compteur = 0;
```



Exemple concret : Compteur de clics

- La variable compteur est déclarée en dehors de l'événement.
- Elle augmente à chaque clic.
- L'affichage se met à jour dynamiquement.

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Introduction au javascript</title>
  <!-- Inclure Bootstrap CSS -->
  <link rel="stylesheet" href="css/bootstrap.css">
</head>
<body class="p-4">
  <button id="btnCompteur" class="btn btn-primary">
    Cliquez-moi
  </button>
  <p id="resultat">0 clic</p>
<script>
  let compteur = 0;
  const bouton = document.querySelector("#btnCompteur");
  const resultat = document.querySelector("#resultat");
  bouton.addEventListener("click", function() {
    compteur = compteur + 1;
    resultat.textContent = compteur + " clic";
  });
</script>
</body>
</html>
```

3. Les conditions : décider

Structure de base

```
if (condition) {
  // action
}
```

Avec alternative

```
if (condition) {
  // si vrai
} else {
  // sinon
}
```

Exemple : message selon nombre de clics

Si le compteur atteint 10 → le texte devient vert.

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Introduction au javascript</title>
  <!-- Inclure Bootstrap CSS -->
<link rel="stylesheet" href="css/bootstrap.css">
</head>
  <body class="p-4">
    <button id="btnCompteur" class="btn btn-primary">
      Cliquez-moi
    </button>
    <p id="resultat">0 clic</p>
  <script>
    let compteur = 0;
    const bouton = document.querySelector("#btnCompteur");
    const resultat = document.querySelector("#resultat");
    bouton.addEventListener("click", function() {
      compteur = compteur + 1;
      resultat.textContent = compteur + " clic";
      if (compteur >= 10) {
        resultat.classList.add("text-success");
      }
    });
  </script>
</body>
</html>
```

Comparaisons importantes

Opérateur	Signification
===	strictement égal
!==	différent
>	supérieur
<	inférieur
>=	supérieur ou égal
<=	inférieur ou égal

Exemple : validation simple

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Introduction au javascript</title>
  <!-- Inclure Bootstrap CSS -->
<link rel="stylesheet" href="css/bootstrap.css">
</head>
<body class="p-4">
  Quel est ton age?
  <input type="number" id="age" class="form-control mb-2">
  <p id="message"></p>
<script>
  const age = document.querySelector("#age");
  const message = document.querySelector("#message");
  age.addEventListener("input", function() {
    if (age.value >= 18) {
      message.textContent = "Accès autorisé";
      message.classList.add("text-success");
      message.classList.remove("text-danger");
    } else {
      message.textContent = "Accès refusé, dégage de là!";
      message.classList.add("text-danger");
      message.classList.remove("text-success");
    }
  });
</script>
</body>
</html>
```

Exemple : Mode clair / sombre

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Introduction au javascript</title>
  <!-- Inclure Bootstrap CSS -->
<link rel="stylesheet" href="css/bootstrap.css">
</head>
<body class="p-4">
  <button id="toggleTheme" class="btn btn-secondary">
    Changer le thème
  </button>
<script>
  let modeSombre = false;
  const bouton = document.querySelector("#toggleTheme");
```

```
bouton.addEventListener("click", function() {  
    if (modeSombre === false) {  
        document.body.classList.add("bg-dark", "text-white");  
        modeSombre = true;  
    } else {  
        document.body.classList.remove("bg-dark", "text-white");  
        modeSombre = false;  
    }  
});  
</script>  
</body>  
</html>
```

Exercice – panier.html

Créer une page contenant :

1. Un bouton “Ajouter au panier”
2. Un compteur affichant le nombre d’articles

Comportement :

- Chaque clic augmente le compteur.
- Si le nombre dépasse 5 :
 - Afficher un message spécial.
- Si inférieur ou égal à 5 :
 - Aucun message.

Contraintes :

- Utiliser let
- Utiliser une condition if
- Utiliser classList pour le style du message

Objectif :

Comprendre le lien entre variable, condition et interface.



Variantes possibles

- Afficher “Plus que X articles avant livraison gratuite”
- Désactiver le bouton après 10 articles
- Ajouter un bouton “Vider le panier”

Exercice – verifpromo.html

L'utilisateur entre une phrase et un mot, et on vérifie si le mot est présent dans la phrase.

Créer :

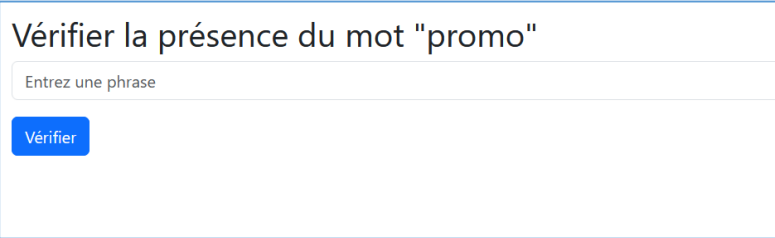
- Un champ texte
- Un bouton "Vérifier"
- Une zone message

Comportement :

- Si le texte contient "promo"
→ afficher message vert
- Sinon
→ message rouge

Contraintes :

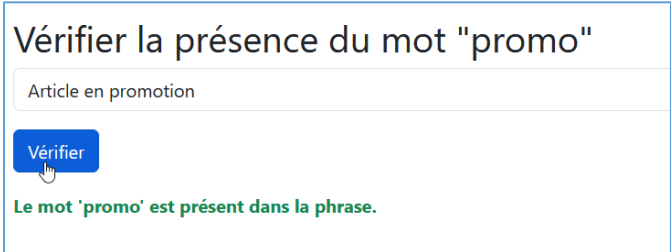
- Utiliser `includes()`
- Utiliser `classList`



Vérifier la présence du mot "promo"

Entrez une phrase

Vérifier

Vérifier la présence du mot "promo"

Article en promotion

Vérifier

Le mot 'promo' est présent dans la phrase.

Exercice - TVA.html

Calculer la TVA sur un prix

Créer :

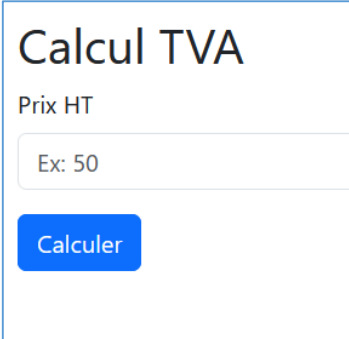
- Un champ prix HT
- Un bouton "Calculer"
- Une zone résultat

Comportement :

- Calculer prix TTC (TVA 21%)
- Afficher le résultat
- Si prix > 100 → afficher en vert
- Sinon → en noir

Contraintes :

- Utiliser `let`
- Utiliser `if`
- Utiliser `classList`



Calcul TVA

Prix HT

Ex: 50

Calculer




Calcul TVA

Prix HT

123

Calculer

Prix TTC : 148.83 €

Exercice - moyenne.html

Afficher réussi ou raté en fonction d'une moyenne

Créer :

- Champ moyenne
- Bouton "Vérifier"

Comportement :

- Si $\geq 10 \rightarrow$ badge "Réussi"
- Sinon $\rightarrow$ badge "Échec"

Contraintes :

- Utiliser condition
- Utiliser classes Bootstrap

Exercice - like.html (Like/Dislike)

Créer une page contenant :

- Un bouton `<button id="btnLike">Like</button>`
- Une zone message (facultatif) : `<p id="etat"></p>`

Comportement :

- Au premier clic : le bouton devient "Dislike" et prend un style "activé" (ex : vert).
- Au second clic : il redevient "Like" et reprend un style "normal".
- Le comportement alterne à chaque clic.

Contraintes :

- Utiliser une variable `let liked = false;`
- Utiliser `if/else`
- Utiliser `classList.add()` / `classList.remove()` (ex : `btn-success` / `btn-outline-secondary`)

Livable attendu :

- Le bouton change de texte et de style à chaque clic.