Activités JavaScript

Fonctions & évènements

Les fonctions

L'utilisation des fonctions permet de réduire la complexité d'un problème en le décomposant en des sous-problèmes moins compliqués.

Les fonctions en JavaScript sont des **citoyens de première classe**.

Une fonction peut être déclarée de plusieurs façons. Elle peut être déclarée comme une variable et passée en tant que paramètre à une autre fonction.

Une fonction est déclarée :

- À l'aide du mot clé : `function`.
- Comme une variable ou constante à l'aide de `const`, `let` ou `var`.
- Comme une fonction anonyme.

Forme générale

```
/* fonction ordinaire */
function nom_fonction(par1, par2, ..., parN) {
  // traitement

  /* La fonction peut retourner, ou non, un résultat à son appelant */
  [return résultat;]
}

/* fonctions anonymes */
const nom_fct2 = function (par1, par2, ..., parM) {
  // traitement
  [return resultat;]
};

const nom_fct3 = (par1, par2, ..., parM) => résultat;
```

Exemples

On veut calculer la distance entre deux points A(x_a, y_a) et B(x_b, y_b) dans un plan munit d'un repère orthonormé. Il existe plusieurs méthodes de calculs :

- Distance de Manhattan,
- Distance euclidienne.

Le script suivant définit trois fonctions permettant de calculer la distance entre deus points en utilisant les deux méthodes.

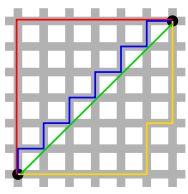

Figure 1, Le chemin en vert indique la distance euclidienne, les autres chemins (rouge, bleu et jaune) indiquent la distance de Manhattan

```
/* fonction ordinaire */
function dist_euclidienne(xa, ya, xb, yb) {
  return Math.sqrt((xb - xa)**2 + (yb - ya)**2);
}

/* fonctions anonymes */
const dist_manhattan = function (xa, ya, xb, yb) {
  return Math.abs(xb - xa) + Math.abs(yb - ya);
};

const dist = (fn, xa, ya, xb, yb) => fn(xa, ya, xb, yb);

/* Afficher la distance entre les deux points A(1, 2) et B(7, 8) */
console.log(dist_euclidienne(1, 2, 7, 8));       // => 8.48528137423857
console.log(dist_manhattan(1, 2, 7, 8));         // => 12

console.log(dist(dist_euclidienne, 1, 2, 7, 8)); // => 8.48528137423857
console.log(dist(dist_manhattan, 1, 2, 7, 8));   // => 12
```

Les évènements

JavaScript est un langage évènementiel, chaque nœud du DOM génère des évènements. Un évènement est déclenché, automatiquement, par le navigateur lorsque :

- Le contenu de la page est chargé.
- L'utilisateur clique sur un bouton dans la page.
- L'utilisateur modifie le contenu d'un champ dans un formulaire.
- L'utilisateur glisse un objet au dessus d'un élément.
- etc.

JavaScript permet d'**écouter un évènement** et d'y attacher, par conséquence, **un gestionnaire d'évènement**.

Réponse à un évènement

Il est possible de répondre aux évènements de plusieurs manières.

Par exemple, pour répondre au clic d'un bouton on peut écrire :

Exemple 1

Définir l'attribut `onclick` de la balise `<button>`. Le clic sur ce bouton exécute le code JavaScript.

```
<button type="button" onclick="alert('You\'re welcome!')">Cliquez-moi</button>
```

Exemple 2

Définir l'attribut `onclick` de la balise `<button>`. Le clic sur ce bouton appelle la fonction `clic01()`.

```
<button type="button" onclick="clic01()">Cliquez-moi</button>
<script>
  function clic01() {
    alert("Good luck!");
  }
</script>
```

Exemple 3

Récupérer la référence du bouton à l'aide de `document.getElementById`, puis attacher l'évènement `onclick` dans JavaScript.

```
<button id="bouton" type="button">Cliquez-moi</button>
<script>
  const bouton = document.getElementById("bouton");
  bouton.onclick = () => {
    alert("You are the best of the best!");
  };
</script>
```

Exemple 4

Récupérer la référence du bouton à l'aide de `document.getElementById`, puis utiliser la méthode `addEventListener` pour attacher une fonction à l'évènement `click`.

```
<button id="btn-2" type="button">Cliquez-moi</button>
<script>
  const btn2 = document.getElementById("btn-2");
  btn2.addEventListener("click", () => {
    alert("Wish you success and happiness!");
  });
</script>
```

Exemple avec les évènements click et submit

Fichier HTML "click_n_submit.html"

```
<!DOCTYPE html>
<html>
<head><title>TP sur les fonctions et les évènements</title></head>
<body>
  <button id="button1">Afficher un message</button>
  <button id="button2">Afficher un autre message</button>
  <p id="message"></p>
  <form>
    <label for="input">Entrez votre nom :</label><br>
    <input type="text" id="input"><br>
    <input type="submit" value="Envoyer">
  </form>
  <script src="events.js"></script>
</body>
</html>
```

Fichier JavaScript "events.js"

```
// Récupérer les champs du formulaire
const msgDiv = document.getElementById('message');
const nameInput = document.getElementById('input');
const btn1 = document.getElementById('button1');
const btn2 = document.getElementById('button2');
const form = document.querySelector('form');

// Afficher un message dans l'élément #message
function showMessage(message) {
  msgDiv.innerHTML = message;
}

// Les fonctions qui gèrent l'évènement click sur les deux boutons
function handleButton1Click() {
  showMessage('Vous avez cliqué sur le premier bouton');
}

function handleButton2Click() {
  showMessage('Vous avez cliqué sur le deuxième bouton');
}

// La fonction qui gère l'évènement submit sur le formulaire
function handleFormSubmit(event) {
  event.preventDefault();              // Annuler l'envoi du formulaire
  const inputValue = nameInput.value;  // Récupérer la valeur du champ #input

  // Affichage de la valeur du champ de formulaire dans le paragraphe
  showMessage('Vous vous appelez ' + inputValue);
}

// Ajouts des écouteurs des évènements
btn1.addEventListener('click', handleButton1Click);
btn2.addEventListener('click', handleButton2Click);
form.addEventListener('submit', handleFormSubmit);
```