Création d'objets avec JavaScript

Introduction

JavaScript est un langage de programmation conçu autour d'un **paradigme** simple, basé sur les **objets**.

Mais, c'est QUOI un objet?

On peut **comparer les objets JavaScript aux objets du monde réel**. Un objet est une **entité à part entière** qui possède des **propriétés**. Si on effectue cette comparaison avec une **tasse** par exemple, on pourra dire qu'une tasse est un objet avec des propriétés. Ces **propriétés** pourront être la **couleur**, la **forme**, le **poids**, le **matériau** qui la constitue, **etc**.

Alors, mais c'est quoi un OBJET informatique?

Revenons à nos programmes : qu'est-ce qu'un **objet informatique**? Tout comme cette tasse, un **objet informatique** est une **entité (représentée par une variable) qui possède des propriétés**. Chaque **propriété** définit une **caractéristique** de l'objet. Une **propriété** est une **information** associée à l'objet (exemple : la couleur de la tasse). Un objet JavaScript possède donc plusieurs propriétés qui lui sont associées. ***Une propriété peut être vue comme une variable attachée à l'objet***. Donc, on peut considérer qu'un **objet** est une **variable** qui **contient plusieurs variables**.

Nous avons déjà utilisé les objets JavaScript, par exemples:

imgSourire = new Image();
imgSourire est un **objet** et *imgSourire*.**width** et *imgSourire*.**height** sont ses propriétés.

Création d'un objet avec JavaScript

Le code suivant montre comment créer un objet en JavaScript :

```
let unePersonne = {
  prenom : "Brendan",
  nom : "Eich",
  age  : 59
}
```

En JavaScript, on peut créer un objet en déclarant une variable et en **définissant ses propriétés** à l'intérieur d'une paire d'accolades : **{ ... }**. Cette manière de créer des objets est appelée **syntaxe littérale** ou **notation littérale d'objet**.

Ainsi, un **objet littéral:**

- commence par un **{** (accolade à gauche) et se termine par un **}** (accolade à droite);
- entre les accolades, chaque propriété est suivi de **:** (deux-points) et les **couples nom/valeur** sont séparés par des **,** (virgule). Les valeurs d'une propriété peuvent être de n'importe quel **type valide** : chaîne(String), nombre (Number), booléen (Boolean), etc.

Le code **ci-dessus** définit une **variable nommée unePersonne** dont la valeur est un objet : on dit aussi que **unePersonne est un objet**. Cet objet possède **trois** (*3*) propriétés : *prenom*, *nom* et *age*. Chaque propriété possède un identifiant (nom) et une valeur et, est séparée des autres **par une virgule**, *sauf la dernière*.

ATTENTION! Il s'agit ici d'une différence notable entre la déclaration d'objet JavaScript et de sélecteurs CSS pour la déclaration de chaque propriété… **le (;) ou la (,)**...

Accéder aux propriétés d'un objet

Une fois l'objet créé, on peut accéder à ses propriétés en utilisant la **notation pointée** (ou *notation à point*), du genre: **monObjet.maPropriete**.

```
let unePersonne = {
  prenom : "Brendan",
  nom : "Eich",
  age  : 59
}

console.log(unePersonne.prenom);//Retourne “Brendan”
console.log(unePersonne.age);//Retourne 59
```

Modifier les propriétés d'un objet

Une fois un objet créé, on peut modifier les valeurs de ses propriétés avec la syntaxe suivante: **monObjet.maPropriete = nouvelleValeur**.

```
let unePersonne = {
  prenom : "Brendan",
  nom : "Eich",
  age  : 59
}

//Modification de la propriété age de cet objet - l'âge de Brendan l’année prochaine...
unePersonne.age  = 60;

console.log(unePersonne.age);//Retourne 60
```

Exemple d'un objet JavaScript dans le contexte d'une petite animation

La déclaration d'objet en JavaScript, permet notamment d'**optimiser notre code** et de **minimiser** le **nombre de variables déclarées**, tout en rendant notre code plus explicite. Le tableau qui suit met en parallèle l'animation d'une balle dans le canvas **sans** la déclaration d'un objet balle (à gauche du tableau) et **avec** (à droite du tableau).

Déclaration de plusieurs variables pour gérer le déplacement d'une **image**: **imgBalle**	Déclaration d'un **objet balle** avec plusieurs propriétés pour gérer son déplacement : **balle**

//Déclaration des variables pour contrôler la balle
 	
let imgBalle= new Image();
let posXBalle = 0;
let posYBalle = 0;
let vitesseBalle = 3;
let largeurBalle = 40;
 	
// Charger l'image de la balle
imgBalle.src = "*images/balle.png*" ;
……
 	
// Dessiner la balle
ctx.drawImage(**imgBalle, posXBalle, posYBalle**);
…
 	
//Incrémenter la position X de la balle
posXBalle += vitesseBalle;
……
 	
//Si la balle arrive du côté droit ou gauche on inverse sa vitesse
if (**posXBalle > leCanvas.width -imgBalle.width** || **posXBalle < 0**){
vitesseBalle = -vitesseBalle;
}
	//Déclaration de la variable balle de type Objet
	
	let balle = {
	img : new Image(),
	posX : 0,
	posY : 0,
	vitesse : 3,
	largeur : 40,
	}
	
	// Charger l'image de la balle
	balle.img.src = "*images/balle.png*" ;
	……
	
	// Dessiner la balle
	ctx.drawImage(**balle.img, balle.posX, balle.posY**);
	…
	
	//Incrémenter la position X de la balle
	balle.posX += balle.vitesse;
	……
	
	//Si la balle arrive du côté droit ou gauche on inverse sa vitesse
	if (**balle.posX > leCanvas.width - balle.largeur** || **balle.posX < 0**){
	balle.vitesse = - balle.vitesse;
	}