Les objets en javascript

Introduction

- Le JavaScript est un langage qui possède un fort potentiel pour la programmation orientée objet.

- En effet, vous devez savoir que le JavaScript est un langage qui intègre l'orienté objet dans sa définition même, ce qui fait que tous les éléments du JavaScript vont soit être des objets, soit pouvoir être convertis et traités comme des objets.

Création d'un objet javascript

- Syntaxe de déclaration d'un objet littérale:

```
Let monObjet={

      propriete1   :   valeur 1  ,
      propriete2   :   valeur 2  ,
      propriete3   :   valeur 3  ,
      Methode1(..) {....}  ,
      Methode2(..){....}

};
```

- Pour accéder à la valeur d'une propriété ou méthode :
- nomObjet.Nompropriete
- nomObjet.NomMéthode

Création d'un objet javascript

- ***Exemple*** :

```
let eleve={

        nom: "Tahri",
        prenom: "Salim" ,
        age: 19 ,
        NomPrenom: function()
                {

        alert("Bonjour, je suis " + this.nom+ "  "+ this.prenom+" agé de :" +this.age+ "
        ans") ;
                }
};

Document.getElementById("d").innerHTML=eleve . nom
Document.getElementById("d").innerHTML=eleve . prenom
Eleve.NomPrenom()
```

L'utilisation du mot clef **this**

- Le mot clef this, sert à faire référence à **l'objet** qui est **couramment manipulé. (équivalent de self en python)**

- autrement dit: lorsque on écrit **this.nom** est équivalent à écrire **eleve.nom**

Création d'un objet javascript

- On peut créer un objet d'une autre manière:

```
var eleve=new Object();

eleve . nom="Tahri";
eleve . prenom="Salim";
eleve . age=19;
eleve . NomPrenom=function(){

        alert("Bonjour, je suis " + this. nom+ "  "+ this. prenom) ;
            }
```

Création d'un objet javascript

- On peut aussi créer un objet d'une autre manière:

```
var eleve={};

eleve.nom="Tahri";
eleve.prenom="Salim";
eleve.age=19;
eleve.NomPrenom=function(){

        alert("Bonjour, je suis " + this.nom+ "   "+ this.prenom) ;
            }
```

Appel des propriété d'un objet

- Pour appeler ou utiliser la propriété d'un objet javascript, il suffit d'utiliser le point (.) de la manière suivante:

- ***Syntaxe:***

```
let x= monObjet . Propriété 1 ;
```

- ***Exemple :***

```
let x= eleve . nom ;
```

Appel de la propriété d'un objet

- Il existe une autre syntaxe pour accéder à la valeur d'une propriété :

```
let x=eleve["nom"]
```

Appel d'une méthode d'un objet

- Pour appeler ou utiliser la propriété d'un objet javascript, il suffit d'utiliser le point (.) de la manière suivante:

- ***Syntaxe***:

```
let x= monObjet . Méthode1() ;
```

- ***Exemple*** :

```
let x= eleve . NomPrenom() ;
```

Appel d'une méthode d'un objet

- On eut faire l'appel d'une méthode avec la syntaxe suivante:

```
Let x=eleve["NomPrenom"]()
```

Ajouter ou modifier des propriétés ou des méthodes

- On peut ajouter ou modifier une propriété ou une méthode la manière suivante:

- ***Syntaxe***:

```
Objet . Propriété =valeur
```

- ***Exemple***:

```
Eleve . nom="saadi"
Eleve . moyenne =12.5
```

La boucle for … in

- L'instruction for … in permet d'itérer sur l'ensemble des propriétés énumérables d'un objet. Pour chaque propriété, JavaScript exécutera l'instruction indiquée. Cette instruction s'utilise de la façon suivante :

```
for (variable in objet) {
  instruction
}
```

- Exemple:

```
for (var cle in eleve) {
  alert(cle)
  alert(eleve[cle])
}
```

Récupérer la liste des clés dans un objet js

- Pour connaitre la liste des clés dans un objet js, on utilise la méthode suivante:

```
Object.keys(VotreObjet)
```

- Exemple:

```
Let t=Object.keys(eleve)

// T=["nom","prenom","age","NomPrenom"]
```

Récupérer les valeurs de chaque propriété d'un objet

- **Object.values** crée un tableau contenant les valeurs de chaque propriété d'un objet.

```
let languages = {
  JavaScript: 50,
  Python: 45,
  Java: 30,
  PHP: 10,
}

let values = Object.values(languages)
console.log(values)
```

- Résultat: [50, 45, 30, 10]

Récupérer la liste des clés et leurs valeurs

- Object.entries crée un tableau de tableaux. Chaque tableau interne a deux éléments. Le premier est la clé et le deuxième est la valeur.

```
let languages = {
  JavaScript: 50,
  Python: 45,
  Java: 30,
  PHP: 10,
}

const tab = Object.entries(languages)
console.log(tab)
```

```
[
    [JavaScript, 50],
    [Python, 45],
    [Java, 30],
    [PHP, 10]
 ]
```

Création d'un tableau d'objets

- On a même la possibilité de définir un tableau d'objets:

```
Let ListePersonne=[

{nom:"salmi",prenom:"karim"},
{nom:"badaoui",prenom:"nissrine"},....
]
```

Supprimer une propriété

- On supprime une propriété en utilisant le mot clé **delete**

- **Exemple:**

```
delete eleve . age
```

Création d'objet avec constructeur

- Syntaxe:

```
function monObjet  (param1, param2, ...){
        this.propriete1  = param1,
        this.propriete2 =param2,
        ....
        this.methode1 = function () {
                //....
                 } ,
         ...

};
```

Création d'un objet avec constructeur

- ***Exemple :***

```
function Telephone(n, p, s, r) {
    this.nom = n;
    this.prix = p;
    this.stock = s;
    this.ref = r;
    this.VerifierStock = function() {
        if (this.stock > 0) {return true;}
        else {return false;}
        }
    }

var t1 = new Telephone("SmartF22", 400, 200, "pro1");
var t2= new Telephone("t2", 200, 0, "Mi Max");
console.log(t1.nom); //SmartF22
console.log(t2.VerifierStock()); //False
```