

# Javascript : quelques précisions

## Ressources

\* <https://developer.mozilla.org/fr/docs/Web/JavaScript>

## Présentation

La programmation orientée objet permet d'encapsuler dans des objets :

- des variables que l'on appelle **attributs**,
- des fonctions que l'on appelle **méthodes**.

Cela permet de contrôler et de gérer de manière spécifique et centralisée les informations (données) et les fonctionnalités (actions) liées :

- à un **objet métier**,
- ou un **objet technique**.

De cette manière ce ne sont pas des **fonctionnalités externes** (fonctions du programme) qui vont manipuler l'objet mais l'objet lui-même qui contient **ses propres fonctionnalités** (méthodes de l'objet) pour les gérer. Pour cela :

- on crée un modèle d'objet avec ses attributs et ses méthodes : on créer une **classe d'objet**,
- on **instancie** la classe pour générer autant d'objet que l'on a besoin dans le programme.

## Création simple d'un objet (pas encore une classe d'objet)

La création d'un objet peut être intéressante pour gérer un seul objet qui a uniquement avec des attributs(données sans méthodes particulières supplémentaires).

- On utilise les accolades pour créer un objet :

```
Var moyenneClasse = {  
};
```

- on ajoute des attributs qui peuvent avoir ou nom une valeur par défaut , être un tableau, etc.

```
var moyenneClasse = {  
    nomClasse : "BTSSIO",  
    moyennes : [],  
    derniereRecuperation : Date.now()  
};
```

- il est possible, après la création d'un objet et **à la volée** :
  - d'ajouter, modifier ou supprimer attribut ou une méthode <code javascript> ajout d'un

`atttribut moyenneClasse.annee = 2017;` ===== Utiliser un objet ===== \* l'afficher dans la console Javascript `console.log(moyenneClasse);` ===== \* l'affichage d'un attribut peut se faire de 2 manières car javascript s'appuie sur les tableaux pour gérer les objets : `utilisation de la notation pointée console.log(moyenneClasse.nomClasse); utilisation de la notation tableau associatif console.log(moyenneClasse["nomClasse"]);` ===== Création d'une classe d'objet (modèle d'objet) ===== La création d'une classe d'objet permet d'avoir un modèle qui encapsule : \* des **attributs**, \* des **méthodes** (fonctionnalités) pour manipuler l'objet lui même.

Une classe d'objet n'est **pas manipulable directement**. Il faudra **instancier** cette classe pour avoir un objet manipulable dans le programme.

On pourra ensuite instancier autant d'objet que l'on a besoin sur le même modèle. \* pour **créer** une classe d'objet : \* la classe est référencée par une **variable**, \* par convention la première lettre du nom de la classe est en **majuscule**, \* utilisation du mot clé **function** pour permettre de mettre le code des méthodes \* remplacer le : par = et , par ; \* ajouter **this** devant les attributs pour préciser qu'il s'agit des attribut de l'objet lui même. `var MoyenneClasse = function() { les attributs this.nomClasse = "BTSSIO"; this.moyennes = []; contiendra autant d'objets que de moyennes this.derniereRecuperation = Date.now(); une méthode gérée par l'objet pour afficher des informations this.afficherConsole = function() { console.log("Nom classe : " + this.nomClasse); console.log("Moyennes : " + JSON.stringify(this.moyennes)); }; };`

## Instancier un objet à partir de sa classe

- on instancie un objet et on met la référence à cet objet dans une variable: `var moyenneClasse = new MoyenneClasse(); afficher les informations de l'objet moyenneClasse.afficherConsole();` ===== Ajout dynamique d'une méthode ===== `MoyenneClasse.resetDerniereRecuperation = function() { this.derniereRecuperation = Date.now(); };` ===== Retour projet 2017 =====

◦ PROJET 2017

From:

/ - APs et stages du BTS SIO du lycée Suzanne Valadon

Permanent link:

</doku.php/hackathon/pronote/javascript>



Last update: **2017/06/01 11:39**