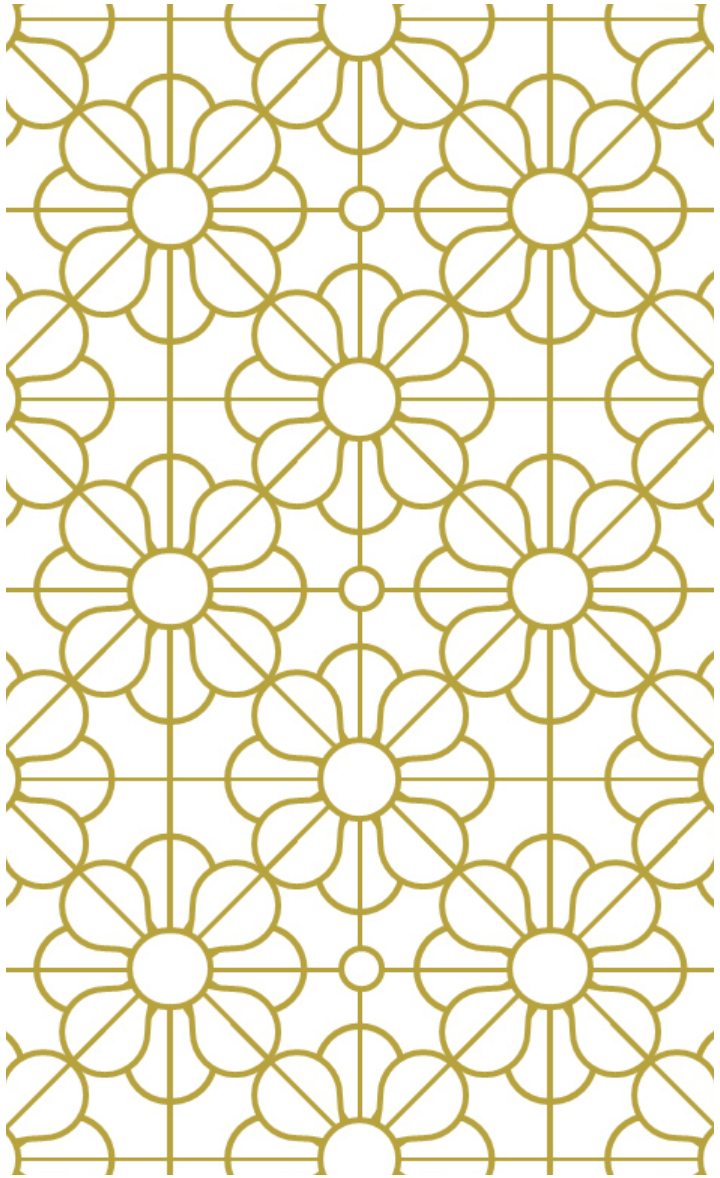
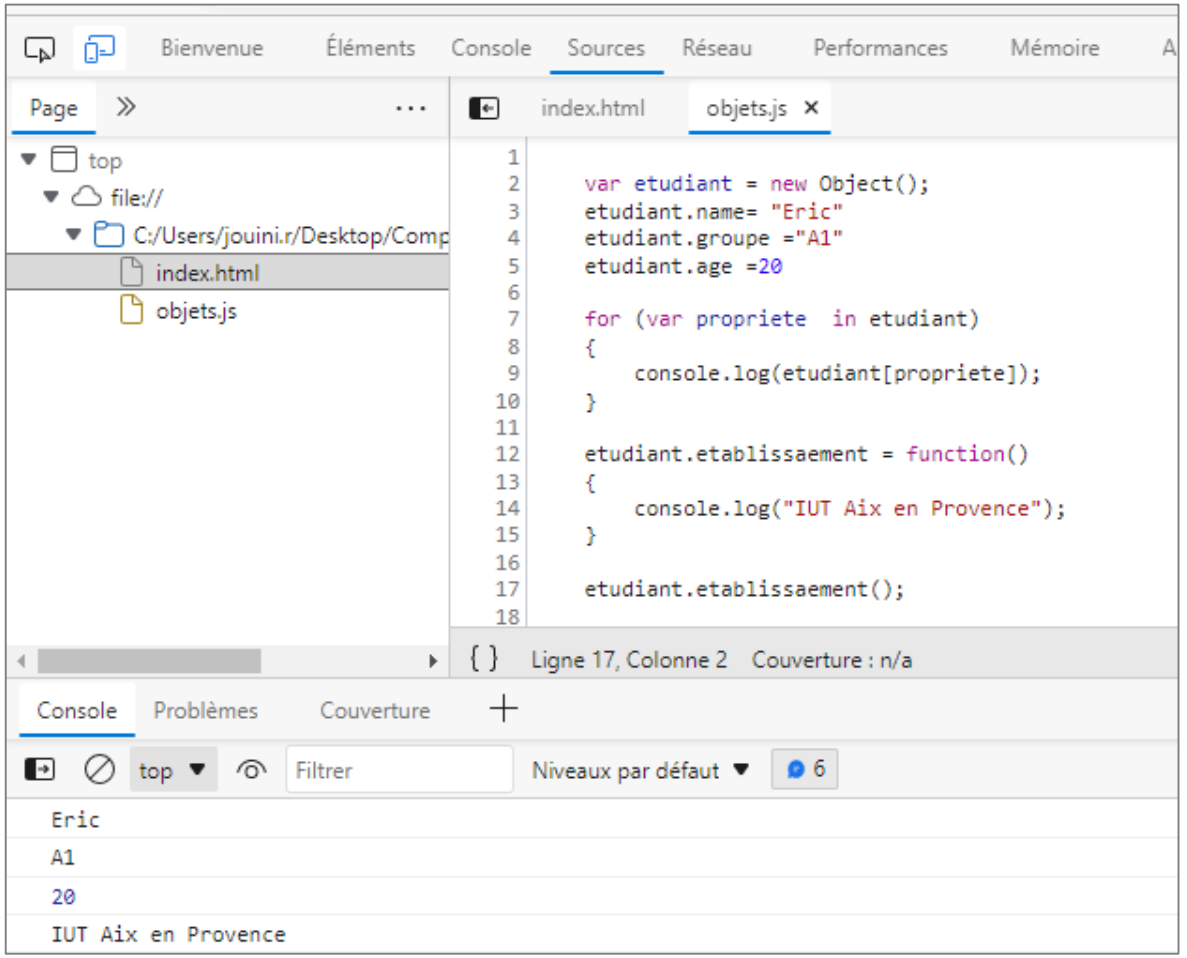


APPROCHE OBJECT EN JS

https://developer.mozilla.org/fr/docs/Learn/JavaScript/Objects/Basics

VARIABLE DE TYPE OBJET (CLASSE OBJECT)

```
//Une variable de type Objet
var nom_objet_1 : new Object ()
nom_objet_1. nom_propriété_1 : valeur_propriete1,
nom_objet_1. nom_propriété_2 : valeur_propriete2,
nom_objet_1. nom_propriété_3 : valeur_propriete3,
nom_objet_1. nom_méthode = function()
{
        //Déclaration complété de la fonction

}
```



VARIABLE DE TYPE OBJET

```
//Une variable de type Objet
 var nom_objet_1 = new Object ()
 nom_objet_1. nom_propriété_1 = valeur_propriete1,
 nom_objet_1. nom_propriété_2 = valeur_propriete2,
 nom_objet_1. nom_propriété_3 = valeur_propriete3,
 nom_objet_1. nom_méthode = function()
 {
          //Déclaration complété de la fonction

 }
```

```
//Une variable de type Objet
 var nom_objet_2 = new Object ()
 nom_objet_2. nom_propriété_1 = valeur_propriete1,
 nom_objet_2. nom_propriété_2 = valeur_propriete2,
 nom_objet_2. nom_propriété_3 = valeur_propriete3,
 nom_objet_2. nom_méthode = function()
 {
          //Déclaration complété de la fonction

 }
```

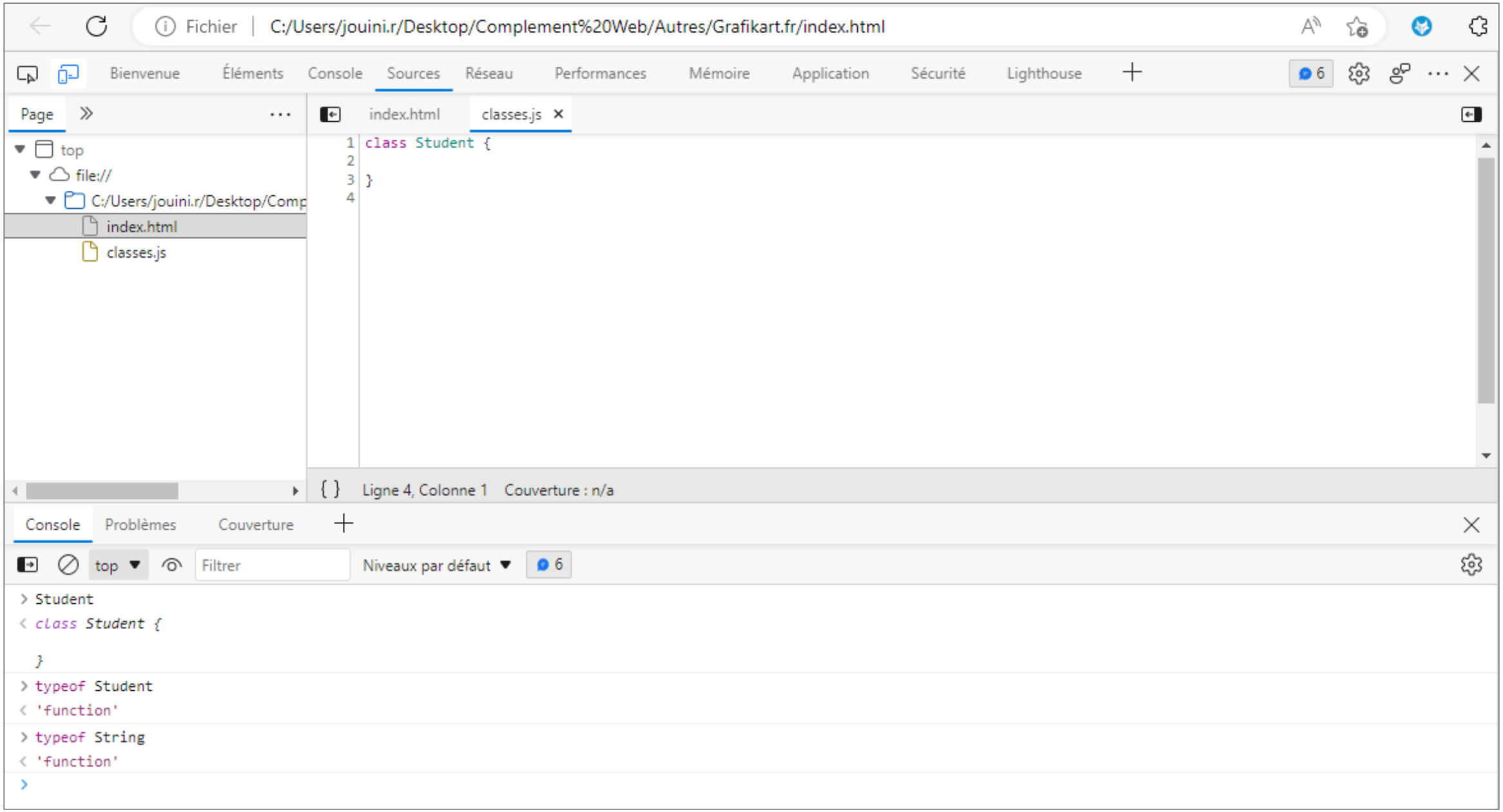
Pour créer des objets qu'ils ont les même attributs → **Fonction constructrice**

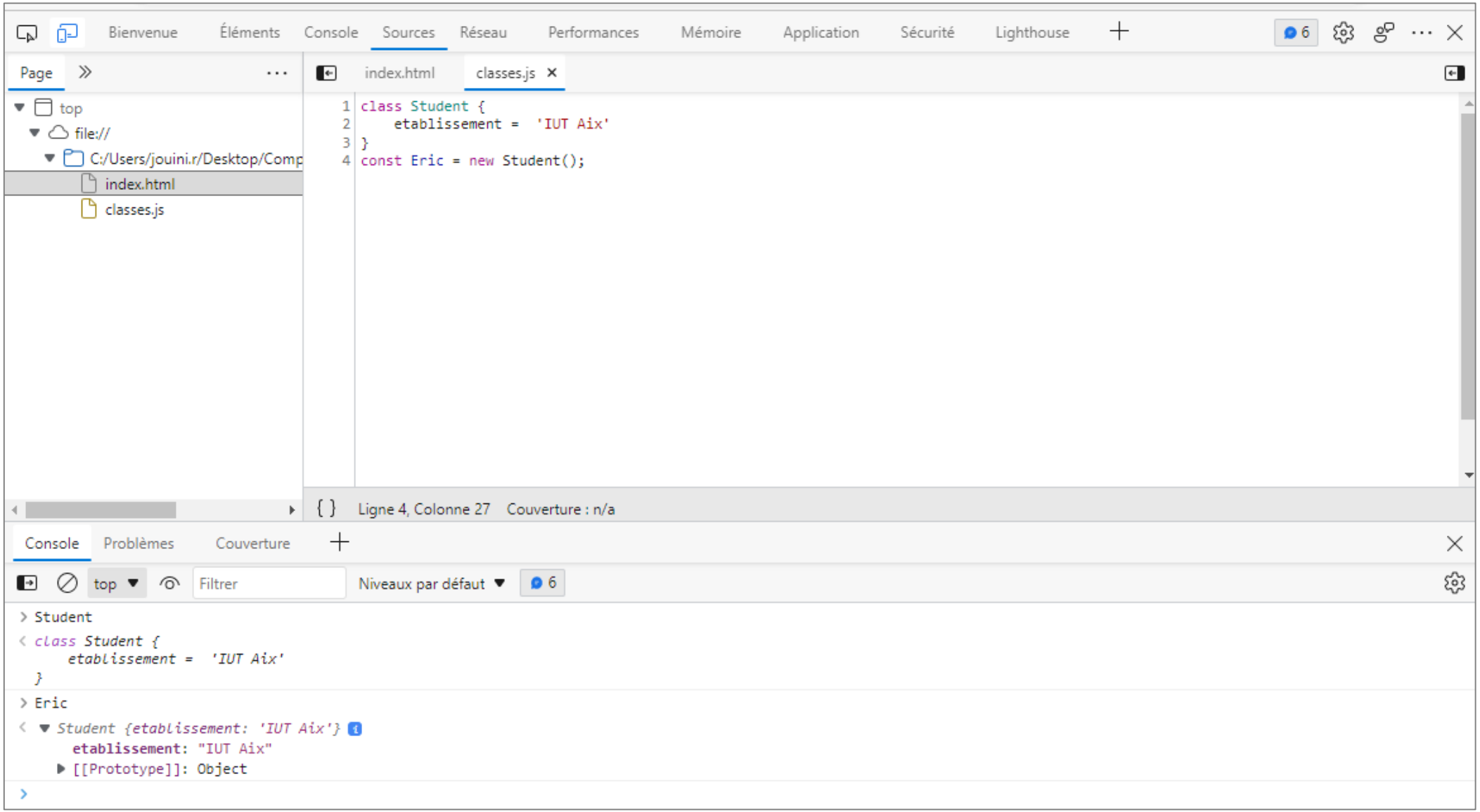
FONCTION CONSTRUCTRICE

```
function Nomfunction (valeur_propriete1, valeur_propriete2 )
{
        this.nom_propriété_1= valeur_propriete1;
        this.nom_propriété_2= valeur_propriete2;
         this. nom_méthode = function()
        {
                //Déclaration complété de la fonction
         }

}
```

```
var nom_objet_1 = new Object ()
 nom_objet_1. nom_propriété_1 = valeur_propriete1,
 nom_objet_1. nom_propriété_2 = valeur_propriete2,
 nom_objet_1. nom_propriété_3 = valeur_propriete3,
 nom_objet_1. nom_méthode = function()
 {
        //Déclaration complété de la fonction

 }
```

```
var nom_objet_1 = new Nomfunction (valeur_propriete1, valeur_propriete2 )

var nom_objet_2 = new Nomfunction (valeur_propriete1, valeur_propriete2 )
```


Fichier | C:/Users/jouini.r/Desktop/Complement%20Web/Autres/Grafikart.fr/index.html
Bienvenue
Éléments
Console
Sources
Réseau
Performances
Mémoire
Application
Sécurité
Lighthouse
6
Page
top
file://
C:/Users/jouini.r/Desktop/Comp
index.html
classes.js
index.html
classes.js
1 class Student {
2
3 }
4
{ } Ligne 4, Colonne 1 Couverture : n/a
Console
Problèmes
Couverture
top
Filtrer
Niveaux par défaut
6
> Student
< class Student {
}
> typeof Student
< 'function'
> typeof String
< 'function'
>


Bienvenue
Éléments
Console
Sources
Réseau
Performances
Mémoire
Application
Sécurité
Lighthouse
6
Page
top
file://
C:/Users/jouini.r/Desktop/Comp
index.html
classes.js
index.html
classes.js
1 class Student {
2 etablissement = 'IUT Aix'
3 }
4 const Eric = new Student();
{ } Ligne 4, Colonne 27 Couverture : n/a
Console
Problèmes
Couverture
top
Filtrer
Niveaux par défaut
6
> Student
< class Student {
etablissement = 'IUT Aix'
}
> Eric
< Student {etablissement: 'IUT Aix'}
etablissement: "IUT Aix"
[[Prototype]]: Object

DÉCLARATION DES OBJETS JAVASCRIPT EN MÉTHODE « INLINE »

```
<!-- Début script JavaScript -->
<script>
/*Séquence 1 – Déclaration des objets JavaScript en méthode "Inline" */
/* Déclaration inline d'un objet JavaScript*/
/* NB: Cette technique ne permet pas l'hériatge à partir de l'objet par la suite*/
    var Addition={
        x:10,
        y:15,
        calcul : function()
        {
            return this.x + this.y;
        }
    };
/* Utilisation de l'objet Addition */
document.write(" Somme : " + Addition.calcul());
</script>
```

CRÉATION DES OBJETS JAVASCRIPT PAR CONSTRUCTEUR

```
<!-- Début script JavaScript -->
<script>
/*Définition d'une fonction constructeur de nom Voiture */
    var Voiture = function()
    {
        /* Attributs de l'objet */
        this.possedeMoteur=true;
        /* Méthodes de l'objet */
        this.avancer = function()
        {
            document.write("avance");
        }

    }
/* Instanciation d'un objet  peugeot206 via le constructeur Voiture */
var peugeot206 = new Voiture();

/* Affichage de l'attribut possedeMoteur de l'objet peugeot206 */
if (peugeot206.possedeMoteur)
{
    document.write("La voiture peugeot206 possède bien un moteur <br />");
}
else
{
    document.write("La voiture peugeot206 ne possède pas de moteur <br />");
}

/* Appel de la méthode avancer de l'objet peugeot206 */
document.write("La voiture peugeot206 ");
peugeot206.avancer();
</script>
```

La voiture peugeot206 possède bien un moteur
La voiture peugeot206 avance

VARIABLES PRIVÉES DANS UNE INSTANCE D'OBJET

```
<!-- Début script JavaScript -->
<script>
/*Définition d'une fonction constructeur de nom Voiture */
    var Voiture = function()
    {
        /* Variable(s) locale(s) non accessible(s) depuis l'extérieur de l'objet */
        var nombreRoues=4;
        /* Méthodes de l'objet */
        this.avancer = function()
        {
            document.write("avance");
        }

    }
/* Instanciation d'un objet  peugeot206 via le constructeur Voiture */
var peugeot206 = new Voiture();

/* Appel de la méthode avancer de l'objet peugeot206 */
document.write("La voiture peugeot206 ");
peugeot206.avancer();

/* Tentative d'afficahge de la varaible locale du constructeur Voiture*/
document.write("<br /> ");
document.write("La voiture peugeot206 a " + peugeot206.nombreRoues + " roues");
</script>
```

La voiture peugeot206 avance
La voiture peugeot206 a undefined roues

PASSAGE DE PARAMÈTRE(S) À UN CONSTRUCTEUR

```
<!-- Début script JavaScript -->
<script>
/*Définition d'une fonction constructeur de nom Voiture */
var Voiture = function(modele)
{
    /* Attribut(s) de l'objet renseigné à l'instanciation*/
    this.modele = modele;
}
/* Instanciation d'un objet  peugeot206 via le constructeur Voiture avec passage de paramètre*/
var maVoiture = new Voiture("peugeot206");

/* Affichage de l'attribut modele de l'objet maVoiture*/
document.write("Je possède une voiture " + maVoiture.modele);
</script>
```

NON PARTAGE DES MÉTHODES PAR LES INSTANCES D'OBJETS

```
var Voiture = function()
{
    /* Attribut(s) de l'objet */
    this.possedeMoteur= true;

    /* Méthode(s) de l'objet*/
    this.avancer=function()
    {
        document.write("Avance");
    }

    this.reculer=function()
    {
        document.write("Recule");
    }

}
/* Instanciation d'un objet  peugeot206 via le constructeur Voiture*/
var peugeot206 = new Voiture();

/* Instanciation d'un objet  clioCapture via le constructeur Voiture*/
var clioCapture = new Voiture();

/* Test d'égalité des méthodes avancer des objets peugeot206 et clioCapture  */
if (peugeot206.avancer == clioCapture.avancer)
{
    document.write("Méthode avancer partagée par les objets peugeot206 et clioCapture <br />");
}
else
{
    document.write("Méthode avancer non partagée par les objets peugeot206 et clioCapture <br />");
}
```

Méthode avancer non partagée par les objets peugeot206 et clioCapture

NOTION DE PROTOTYPE

```
/*Définition d'une fonction constructeur de nom Voiture */
/* NB : Le constructeur est ici VIDE*/
var Voiture = function() {}

/* Test de l'existence d'un prototype par défaut pour tout constructeur */
document.write("Prototype du constructeur : " + Voiture.prototype + "<br />");

/* Ajout d'une méthode zigzaguer au prototype du constructeur Voiture */
Voiture.prototype.zigzaguer = function ()
{
    document.write("Zigzague dangereusement");
};

/* Instanciation d'un objet  peugeot206 via le constructeur Voiture*/
var peugeot206 = new Voiture();

/* Appel de la méthode zigzaguer de l'objet peugeot206 accessible via le prototype du constructeur Voiture*/
document.write("la voiture peugeot206 ");
peugeot206.zigzaguer();

/* Instanciation d'un objet  clioCapture via le constructeur Voiture*/
var clioCapture = new Voiture();

/* Test méthode zigzaguer partagée ou pas entre les objets peugeot206 et clioCapture  */
if (peugeot206.zigzaguer == clioCapture.zigzaguer)
{
    document.write("<br /> Méthode avancer partagée par les objets peugeot206 et clioCapture <br />");
}
else
{
    document.write("Méthode avancer non partagée par les objets peugeot206 et clioCapture <br />");
}
</script>
```

Prototype du constructeur : [object Object]
la voiture peugeot206 Zigzague dangereusement
Méthode avancer partagée par les objets peugeot206 et clioCapture

LE PROTOTYPE EN JAVASCRIPT

Le prototype en JavaScript est un mécanisme qui permet aux objets JavaScript d'hériter des propriétés d'autres objets.

Les prototypes implémentent un héritage différent de celui rencontré dans les langages de programmation objets habituels.

JavaScript est souvent décrit comme un langage basé sur les prototypes, chaque objet pouvant avoir un **prototype objet** d'où il hérite des méthodes et des attributs.

Un prototype peut lui aussi avoir son prototype objet duquel il héritera des méthodes et des attributs et ainsi de suite. On parle alors de chaîne de prototypage (ou *prototype chain* en anglais).

Cela permet d'expliquer pourquoi différents objets possèdent des attributs et des méthodes définis à partir d'autres objets.

POO CLASSIQUE VS JAVASCRIPT

En programmation orientée objet classique, les classes sont définies, puis lorsque des instances sont créées, l'ensemble des attributs et des méthodes sont copiés dans l'instance.

En JavaScript en revanche, tout n'est pas copié : on établit un lien entre l'objet instancié et son constructeur (c'est un lien dans la chaîne de prototypage).

→ On détermine alors les méthodes et les attributs en remontant la chaîne.

SURCHARGE D'UNE MÉTHODE

```
/*Définition d'une fonction constructeur de nom Voiture */
/* NB : Le constructeur est ici VIDE*/
var Voiture = function() {};

/* Ajout d'une méthode piler au prototype du constructeur Voiture */
Voiture.prototype.piler = function ()
{
    document.write("pile <br />");
};

/* Instanciation d'un objet  peugeot206 via le constructeur Voiture*/
var peugeot206 = new Voiture();

/* Appel de la méthode pilerer de l'objet peugeot206 accessible via le prototype du constructeur Voiture*/
document.write("la voiture peugeot206 ");
peugeot206.piler();

/* Instanciation d'un objet  clioCapture via le constructeur Voiture*/
var clioCapture = new Voiture();

/* Modification (surcharge) de la méthode piler pour l'objet peugeot206 */
peugeot206.piler= function()
{
    document.write("pile brutalement <br />");
};

/* Appel de la méthode piler surchargée de l'objet peugeot206*/
document.write("la voiture peugeot206 ");
peugeot206.piler();

/* Appel de la méthode piler non surchargée de l'objet clioCapture*/
document.write("la voiture clioCapture ");
clioCapture.piler();
```

la voiture peugeot206 pile
la voiture peugeot206 pile brutalement
la voiture clioCapture pile

EXTENSION D'UN PROTOTYPE

```
/*Définition d'une fonction constructeur de nom Voiture */
/* NB : Le constructeur est ici VIDE*/
var Voiture = function() {};

/* Ajout d'une méthode accelerer au prototype du constructeur Voiture */
Voiture.prototype.accelerer = function ()
{
    document.write("accélère <br />");
};

/* Instanciation d'un objet  peugeot206 via le constructeur Voiture*/
var peugeot206 = new Voiture();

/* Appel de la méthode accelerer  de l'objet peugeot206 accessible via le prototype du constructeur Voiture*/
document.write("la voiture peugeot206 ");
peugeot206.accelerer();

/* Ajout (surcharge) de la méthode ralentir accessible via le prototype du constructeur Voiture*/
Voiture.prototype.ralentir= function()
{
    document.write(" ralentir <br />");
};

/* Appel de la méthode ralentir  ajoutée dans le prototype du constructeur Voiture à partir de l'objet peugeot206*/
document.write("la voiture peugeot206 ");
peugeot206.ralentir();
```

la voiture peugeot206 accélère
la voiture peugeot206 ralentir