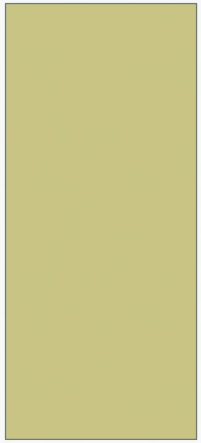
CLASSES ET OBJETS JAVASCRIPT

DR MAMADOU BOUSSO
UNIVERSITÉ DE THIÉS

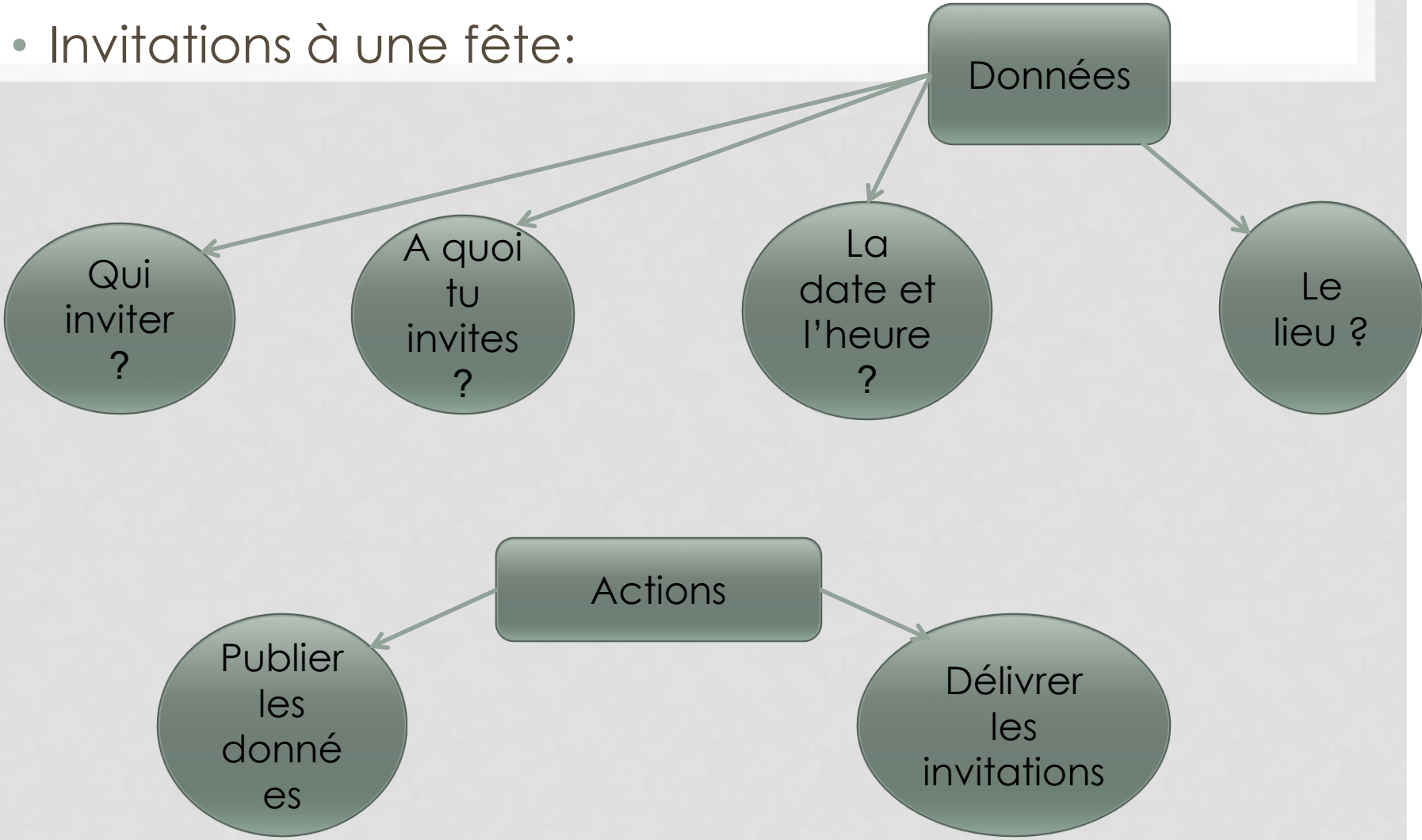


DEFINITIONS

- Données + actions = Objet
  - Les objets javascript combinent données et actions dans une structure de données unique qui garde les données et agit sur elles;

EXEMPLE

- Invitations à une fête:



EXEMPLE: UN OBJET INVITATION COMPORTERA

- Des données:

```
var invite;
var lieuInvitation;
var dateInvitation;
var natureInvitation;
```

- Des actions:

```
function afficher(){
..... }

function delivrer(){
.....
}
```

EXEMPLE: UN OBJET INVITATION

- A travers l'objet invitation:
  - Données et fonctions co-existent et sont liées;
  - Les fonctions de l'objet ont accés aux données de l'objet sans avoir besoin de les passer par arguments;
  - Les données sont visibles à l'intérieur de l'objet mais cachées de l'extérieure de l'objet;

PROPRIÉTÉS DES OBJETS

- Un objet posséde ses propres données:
  - Ses variables sont appelées **propriétés** et déterminent son **état**;
  - Ses fonctions sont appelées **méthodes** et déterminent ses **responsabilités**;
- Un objet est un **type de données composé**;

EXEMPLE: UN OBJET INVITATION EST UN TYPE DE DONNÉES COMPOSÉ:

- Des **propriétés**:

```
var invite;
var lieuInvitation;
var dateInvitation;
var natureInvitation;
```

- Des **methodes**:

```
function afficher(){
..... }

function delivrer(){
.....
}
```

ACCÉS DE L'OBJET À SES MEMBRES:

- Se fait par l'opérateur ".":

```
invitation.invite = "Mamadou";

invitation.lieu = "Universite de Thies";

invitation.afficher();
```

L'opérateur "." etablit une référence entre une propriété ou méthode et l'objet auquel il appartient

CONSTRUCTION D'UN OBJET

- Un objet est un type de données composé et requiert une fonction spéciale appélée constructeur pour:
  - Être crée
  - Être initialisé

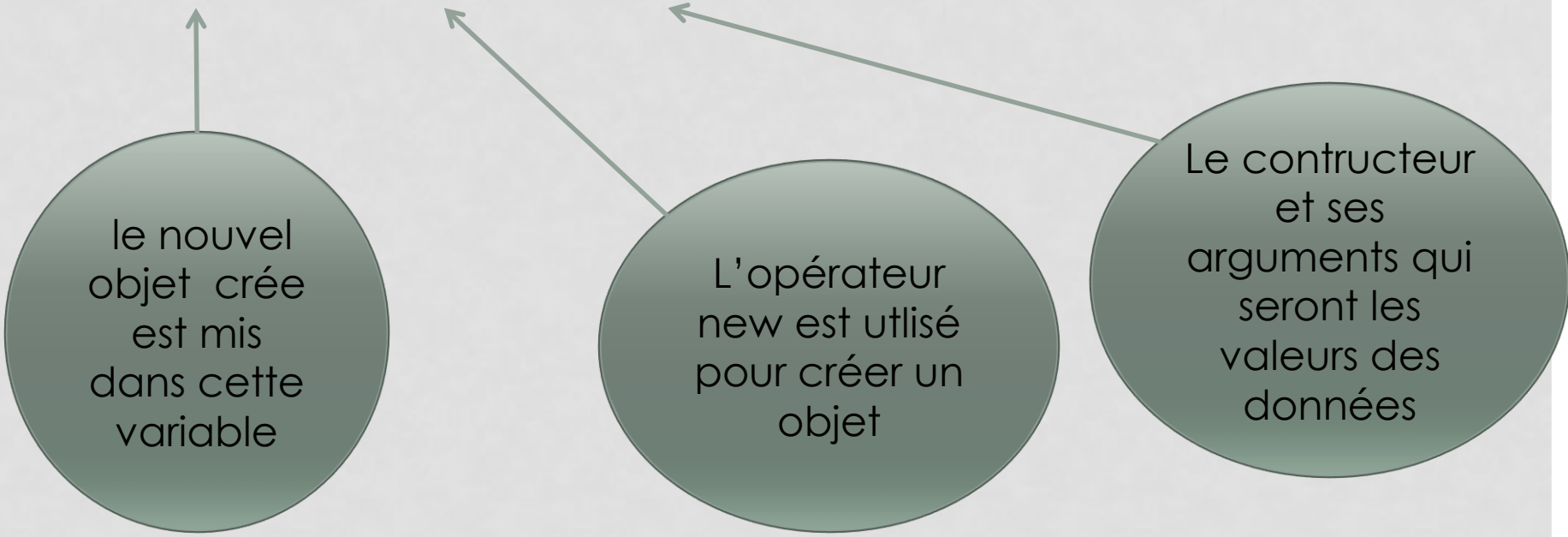
LE CONSTRUCTEUR

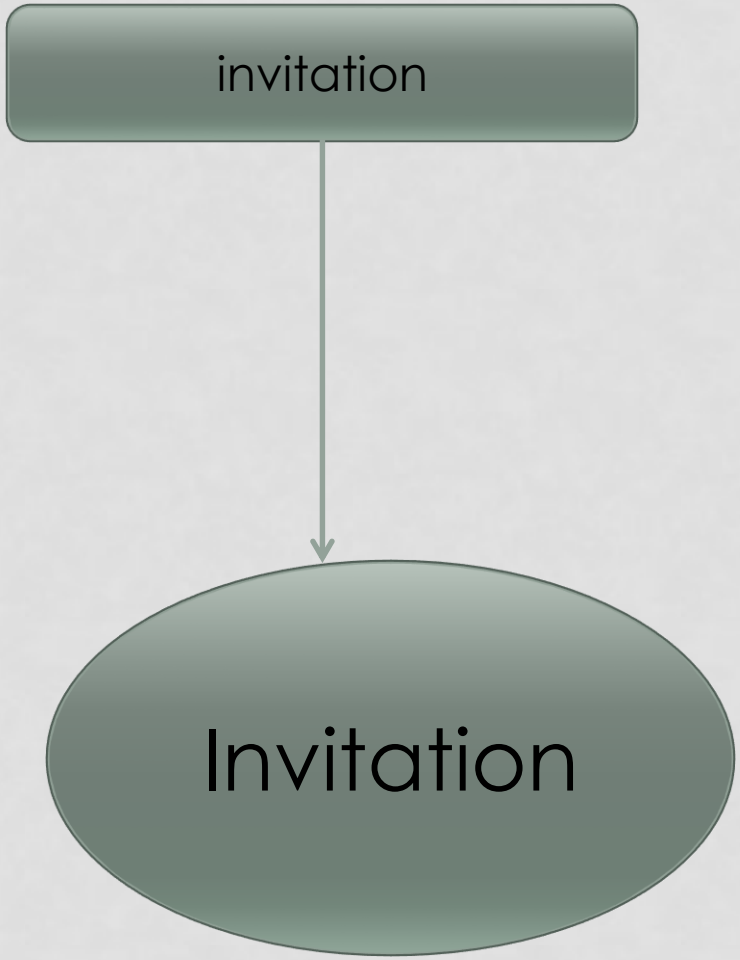
- Une fonction spéciale qui:
  - Porte le même nom que l'objet;
  - Qui initialise les données de l'objet;
  - Doit être écrit par le programmeur pour créer les objets;

COMMENT CONSTRUIRE UN OBJET?

- Utiliser l'opérateur new et un constructeur:

```
var invitation = new Invitation("Toi","fete","10h00","Dakar")
```




invitation
Invitation

QUE MET-ON DANS UN CONSTRUCTEUR?

- Le rôle important d'un constructeur est d'initialiser les propriétés de l'objet;
- Le mot-clé this est utilisé pour créer une propriété qui appartient à l'objet que nous construisons;

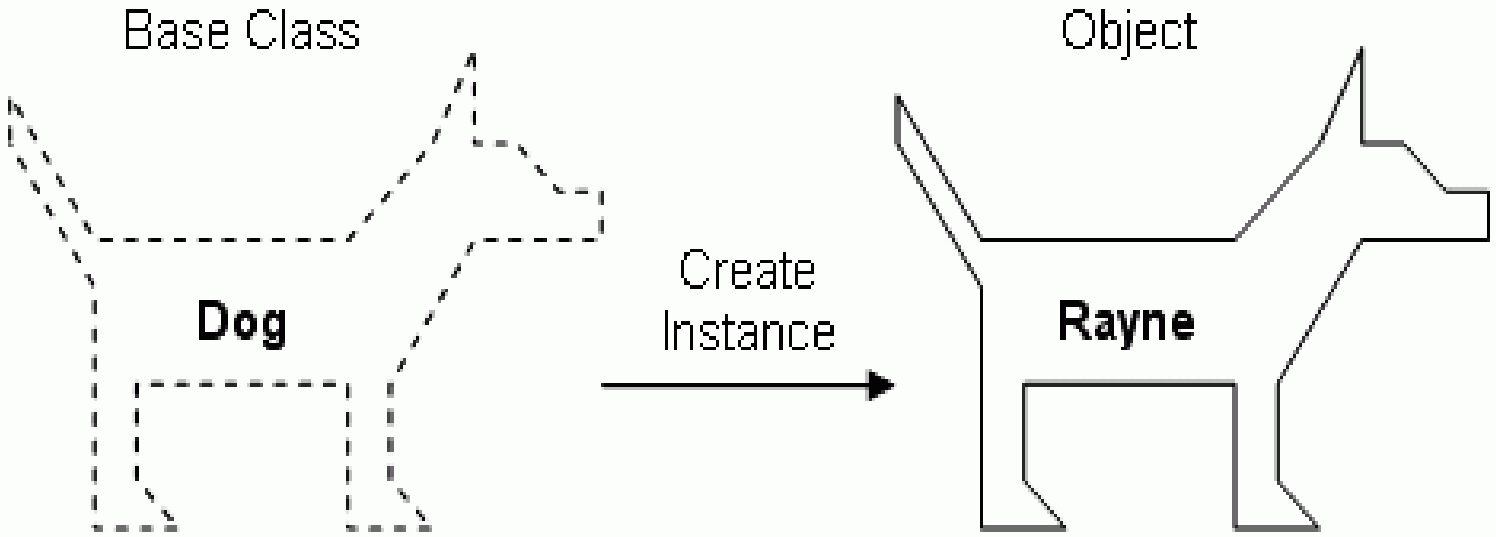
QUE MET-ON DANS UN CONSTRUCTEUR?

```
function Invitation(qui, quoi, quand,ou)
{
        this.invite = qui;
        this.natureInvitation = quoi;
        this.dateInvitation = quand;
        this.lieuInvitation = ou;
}
```

Creation d'un constructeur avec quatre propriétés à qui sont assignées les valeurs passées en paramétre

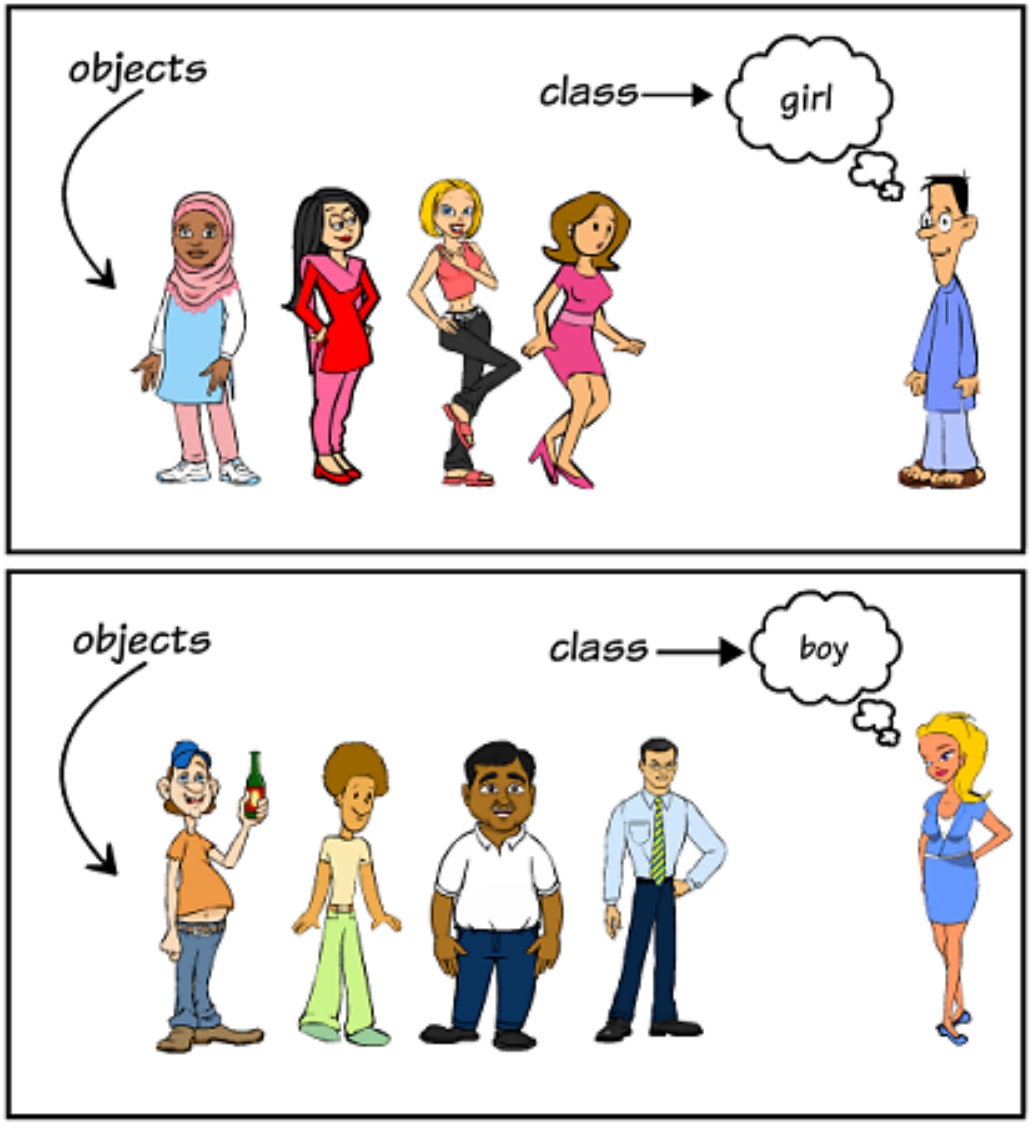
LES CLASSES

- Une classe est la description d'un objet;
- Un modéle qui décrit de quoi les objets sont faits



Properties	**Methods**
Color	Sit
Eye Color	Lay Down
Height	Shake
Length	Come
Weight	

Property values	**Methods**
Color: Gray, White, and Black	Sit
Eye Color: Blue and Brown	Lay Down
Height: 18 Inches	Shake
Length: 36 Inches	Come
Weight: 30 Pounds	


objects
class
girl
objects
class
boy

CLASSES VS OBJETS

Classes

- Elles décrivent les propriétés et les méthodes d'un objet;
- N'existe pas en mémoire,
- Elles sont des modéles

Objets

- Une instance améne les objets à la vie;
- Donne une valeur aux propriétés;
- Vit en mémoire
- Chaque objet a ses propres copies de ses propriétés
- Les méthodes sont dupliquées dans chaque instance;
- On accéde à une instance par le mot-clé this

CREATION D'UNE CLASSE EN JAVASCRIPT

- Le mot-clé prototype

```
//Constructeur de la classe
function maClasse() {
    this.attribut1;
    this.attribut2;
}
 //methodeA de la classe
maClasse.prototype.methodeA() = function() {
    // code
}
//methodeB de la classe
maClasse.prototype.methodeB() = function() {
    // code
}
```

EXERCICE 1

- Construire la classe Invitation en javascript

EXERCICE 2

- Un blog est composé de plusieurs entrées;
- Chaque entrée comprend une date, un corps de texte, un graphique(optionnelle)
- Creer un objet entrée de blog et écrire son constructeur;
- Construire un objet blog et écrire un constructeur(utiliser les tableaux)

EXERCICE 3

- Que vous suggére:

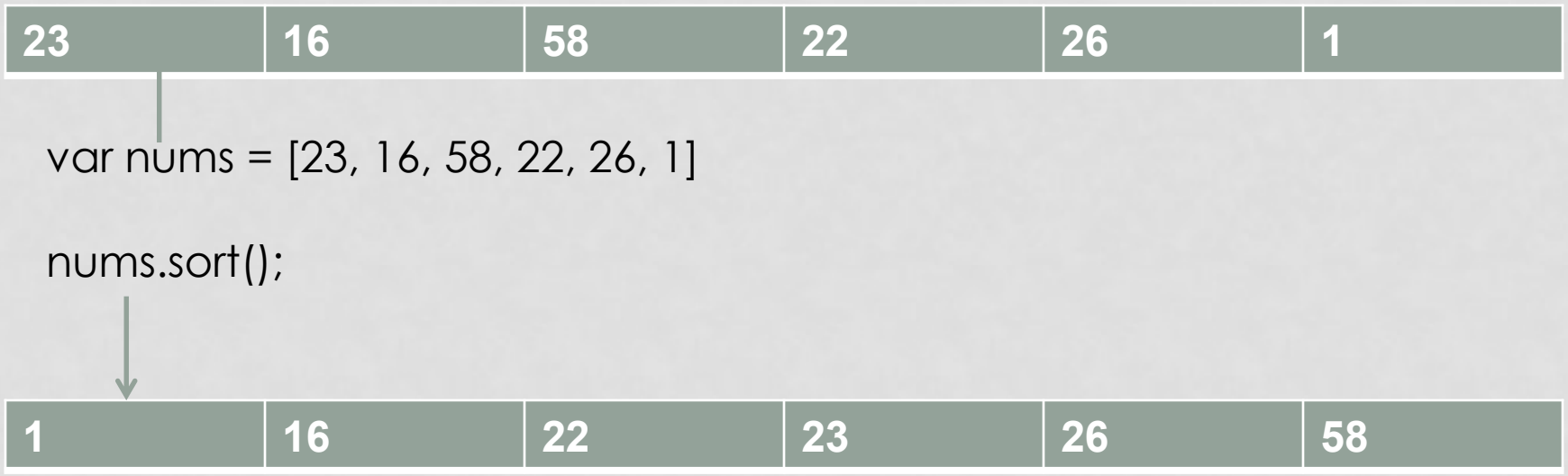
document.getElementById(“test”);

var tab = new Array();

LES TABLEAUX EN JAVASCRIPT

- Les tableaux sont des objets avec des propriétés et des méthodes:
  - length est une propriété qui retourne le nombre d'éléments du tableau;
  - Sort() est une méthode qui permet de trier les éléments d'un tableau suivant l'ordre ascendant;

EXEMPLE



FONCTION COMPARE()

- Le tri d'un tableau d'objet est déterminé par la fonction compare(x,y);
- Elle montre comment deux éléments d'un tableau sont comparés;
- Dans le cas d'entiers décimaux ou relatifs x et y elle retourne x-y;
- Pour d'autre type de données il faudra la réécrire et l'injecter dans la fonction sort()

EXEMPLE

```
function compare(){
return blog.date1 – blog.date2;
}

blogs.sort(compare);
```

LA MÉTHODE TOSTRING()

- Chaque objet javascript a une méthode appelée toString();
- Il fournit la représentation textuelle d'un objet;
- Il intervient automatiquement quand un objet est utilisé dans un contexte ou un String est attendu,
- Ses résultats n'ont pas forcément de sens mais on peut le réimplémenter;

GETTER

- Ce sont des méthodes qui permettent d'accéder aux données d'un objet;

Exemple:

```
function getInvite(){
        return this.invite;
}
```

SETTER

- Ce sont des méthodes qui permettent d'affecter une valeur aux données d'un objet;

Exemple:

```
function setInvite(qui){
        this.invite = qui;
}
```

L'OBJET STRING

- Les chaines de caractéres sont des objets String;
- Ils incluent beaucoup d'objets qui permettent de faire de la recherche sur des chaines de caractéres;

L'OBJET STRING

- length: propriété qui fournit le nombre de chaine de caractére d'un objet String;
- indexOf(): recherche si l'objet String contient une sous-chaine donnée;
- charAt(): trouve si un certain caractére est dans la chaine;
- toLowerCase(): convertit la chaine en minuscule;
- toUpperCase(): convertit la chaine en majuscule

EXEMPLE

- Testez les codes suivants:

```
var str = “je vais au marche”;

alert(str.toString());

alert(str.indexOf(“vais”));

alert(str.indexOf(“maman”));
```

L'OBJET MATH

- L'objet Math est un objet qui n'a pas de données variables;
- L'objet Math contient toutes les méthodes et constantes mathématiques;

L'OBJET MATH

- Math.PI = la constante mathématique pi;
- Math.round(a) arrondit un nombre flottant a en nombre entier;
- Math.floor(a) arrondit un nombre flottant a au nombre entier inférieur;
- Math.ceiling(a) arrondit un nombre flottant a au nombre entier supérieur;
- Math.random() génére un nombre aléatoire entre 0 et 1

L'OBJET DOCUMENT

- Chaque page web chargé a son propre objet document;
- C'est un point d'entrée au contenu de la page;
- Il fournit des fonctionalités globales pour tout le document;

L'OBJET DOCUMENT

- Quelques propriétés:
  - documentURI: URL du document;
  - Implementation: DOM du document
  - lastStyleSheetSet: le dernier CSS utilisé;
  - styleSheets: la feuille de style utilisée;
  - body: le body du document;
  - cookie: retourne la liste des cookies du document

L'OBJET DOCUMENT

- Quelques méthodes:
  - createElement(nom de balise en string)
  - getElementById(id de l'élément)
  - n,de balise en String)
  - createTextNode(texte)
  - getElementsByClassName(nom de la classe)
  - Pour tout infos complémentaires visitez le lien

  https://developer.mozilla.org/en/docs/Web/API/Document

AUTRE MANIÉRE DE CREATION D'OBJETS

Les initialisateurs d'objet:

- une liste contenant plusieurs propriétés(eventuellement 0) et leurs valeurs associées;
- La liste est entourée d'accolades ({})

LES INITIALISATEURS D'OBJET

- **Exemple:**

```
var obj = {
propriété_1:   valeur_1,   // propriété_# peut être un identifiant
propriété_2: valeur_2,   // ou un nombre
  // ...,
propriété_n: valeur_n
 }; // ou une chaîne
```

- Valeur_i est une expression dont la valeur sera affectée à propriété_i

LES INITIALISATEURS D'OBJET

- Chaque initialisateur est une expression;
- Il entraine la création d'un nouvel objet dans l'instruction pour laquelle il est executé;
- Ils sont créés de la même maniére qu'avec " new Object()"
- Ils sont des instances de la classe Object

LES INITIALISATEURS D'OBJET

Exemple:

- // L'objet x est crée si la condition cond est vraie

If (cond) var x = { emplacement: "le monde"};

// Un objet maHonda est créé avec trois propriétés

// La propriété moteur est un objet avec deux propriétés

var maHonda = { couleur: "rouge", roue: 4, moteur: {cylindres: 4, taille: 2.2}};

LES INITIALISATEURS D'OBJET

On peut y ajouter des méthodes:

```
var monObj = {

MaMethode: function(parametres) {
//.....corps de la methode
 }
};
```

LES INITIALISATEURS D'OBJET

On peut y ajouter des méthodes:

```
var maHonda  = { couleur: "rouge", roue: 4, moteur: {cylindres: 4, taille: 2.2},

afficherCouleur: function(){
Console.log(this.couleur);
}
};
```

LA MÉTHODE OBJECT.CREATE()

- Les objets peuvent être crées en utilisant la méthode Object.create()

- Elle permet de choisir le prototype pour l'objet que l'on souhaite créer sans définir un constructeur

LA MÉTHODE OBJECT.CREATE()

- // Propriétés pour animal et encapsulation des méthodes

```
var Animal = {
        type: "Invertébrés",        // Valeur par défaut  value of properties
        afficherType : function(){  // Une méthode pour afficher le type Animal
                console.log(this.type);
        }
}
```

- // On crée un nouveau type d'animal, animal1

```
        var animal1 = Object.create(Animal);
        animal1.afficherType(); // affichera Invertébrés
```

- // On crée un type d'animal "Poissons"

```
        var poisson = Object.create(Animal);
        poisson.type = "Poisson";
        poisson.afficherType(); // affichera Poissons
```

COMPARER DES OBJETS

- En javascript les objets fonctionnent par référence:
  - Deux objets distincts ne sont jamais égaux mêmes s'ils ont les mêmes valeurs pour les mêmes propriétés;
  - On aura une équivalence uniquement si on compare un objet à lui-même;

COMPARER DES OBJETS

Exemple:

- // Deux variables avec deux objets distincts qui ont les mêmes propriétés

```
var fruit = {nom: "pomme"};
var fruit2 = {nom: "pomme"};

fruit == fruit2  // return false
fruit === fruit2 // return false
```

COMPARER DES OBJETS

Exemple:

- // Deux variables avec un même objet

```
var fruit = {nom: "pomme"};
var fruit2 = fruit;  // On affecte la même référence
```

- // dans ce cas fruit et fruit2 pointent vers le même objet

```
fruit == fruit2  // return true
fruit === fruit2 // return true
```