

## Introduction

La Programmation Orientée Objet (POO) est un paradigme de programmation qui repose sur les concepts d'objets et de classes. En JavaScript, la POO est implémentée à l'aide de classes, d'objets, et de prototypes.

### 1. Concepts de base de la POO

- a. **Classe** : Une classe est une structure qui définit les propriétés (attributs) et les méthodes (comportements) d'un objet.
- b. **Objet** : Une instance d'une classe.
- c. **Encapsulation** : Regrouper les données (attributs) et les méthodes dans une classe.
- d. **Héritage** : Une classe peut hériter des propriétés et des méthodes d'une autre classe.
- e. **Polymorphisme** : Une méthode peut se comporter différemment selon le contexte.
- f. **Abstraction** : Cacher les détails complexes et n'exposer que l'essentiel.

### Création des objets en JS :

#### Littéral d'objet

```
const obj = {  
  name: "Alami",  
  age: 25,  
  display: function() {  
    console.log(`Bonjour, je suis ${this.name}`);  
  }  
};  
  
obj.greet();
```

La méthode la plus simple consiste à utiliser une notation littérale.  
javascript

#### Object.create

```
const prototype = {  
  greet: function() {  
    console.log(`Bonjour, je suis ${this.name}`);  
  }  
};  
  
const obj = Object.create(prototype);  
obj.name = "Alami";  
obj.greet();
```

La méthode `Object.create` permet de créer un objet en spécifiant un prototype existant.  
javascript

### Création des objets en JS :

#### Constructeur avec new Object()

```
const obj = new Object();  
obj.name = "RAMI";  
obj.age = 30;  
obj.greet = function() {  
  console.log(`Bonjour, je suis ${this.name}`);  
};  
  
obj.greet();
```

On peut créer un objet à l'aide  
du constructeur global Object.  
javascript

Fonction constructeur

### Création des objets en JS :

#### Fonction constructeur

```
function Personne(name, age) {  
  this.name = name;  
  this.age = age;  
  this.greet = function() {  
    console.log(`Bonjour, je suis ${this.name}`);  
  };  
}  
  
const person1 = new Personne("KABIRI", 40);  
const person2 = new Personne("RAMI", 35);  
  
person1.greet();  
person2.greet();
```

```
class Personne {  
  constructor(nom, age) {  
    this.nom = nom;  
    this.age = age;  
  }  
  
  greet() {  
    console.log(`Bonjour, je m'appelle ${this.nom} et j'ai ${this.age} ans.`);  
  }  
}  
  
// Créer une instance de la classe  
const person1 = new Personne('Alami', 25);  
person1.greet(); // Bonjour, je m'appelle Alami et j'ai 25 ans.
```

```
class Personne {  
  constructor(nom, age) {  
    this.nom = nom;  
    this.age = age;  
  }  
  
  greet() {  
    console.log(`Bonjour, je m'appelle ${this.nom} et j'ai ${this.age} ans.`);  
  }  
}  
  
// Créer une instance de la classe  
const person1 = new Personne('Alami', 25);  
person1.greet(); // Bonjour, je m'appelle Alami et j'ai 25 ans.
```

# Pratique

## La programmation Orienté Objet en JS

### Etude de cas (Gestion panier site e-commerce)

#### Objet Ligne panier

```
function lignepanier(produit,qtc){  
  this.produit=produit  
  this.qtc=qtc  
}
```

#### Objet Produit

```
{  
  id:1,  
  titre:"Smartphone 1",  
  description:"lorem1 lorem2",  
  prix:2500,  
  image:"ilg.png",  
  categorie:"smartphone"  
},
```

```
localStorage.setItem('panier')
```

**Panier**

**Lignepanier : []**

**lignePanier**

**produit: Produit**  
**qtc : int**

### Etude de cas (Gestion panier site e-commerce)


Liste des catégories

- smartphone
- tv

Liste des produits

Search


Search



Smartphone 1

card's content.


Ajouter



Smartphone 2

card's content.


Ajouter



TV 1

card's content.


Ajouter



Smartphone 3

card's content.


Ajouter



TV 2

card's content.


Ajouter



TV 3

card's content.

Ajouter

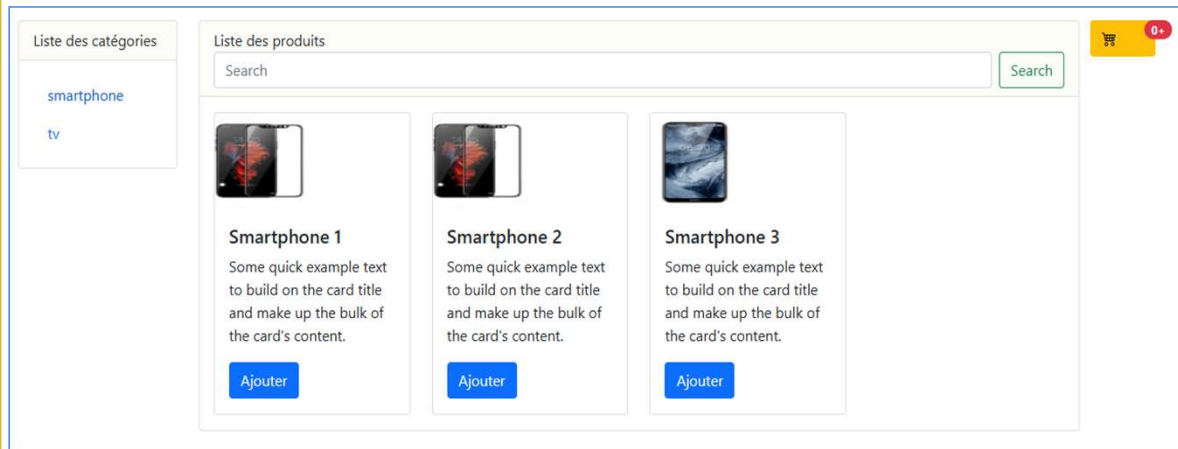


0+

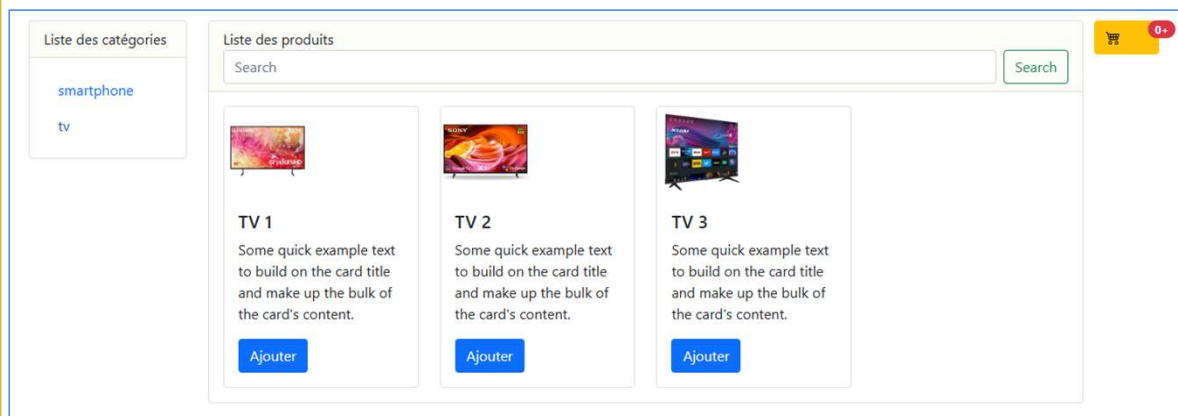
Feature 1 : Display All products



### Feature 2 : Display By catégorie

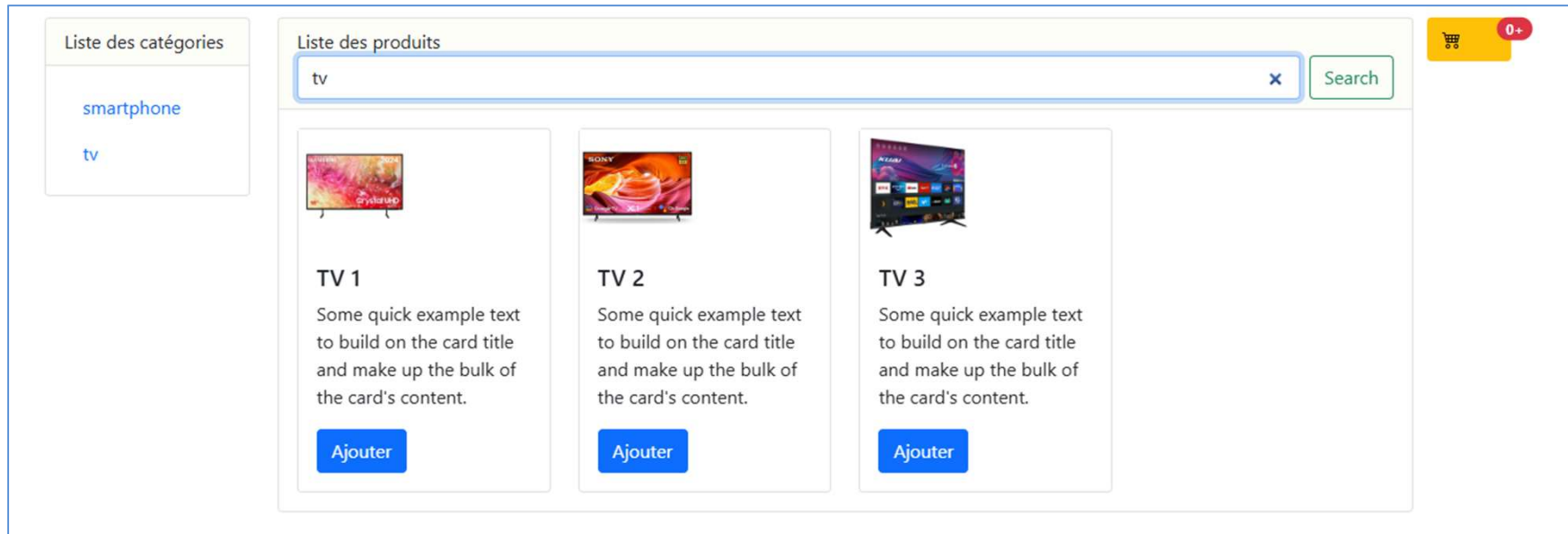


```
const  
produitsCat=produits.filter(produit=>  
produit.categorie===x)
```



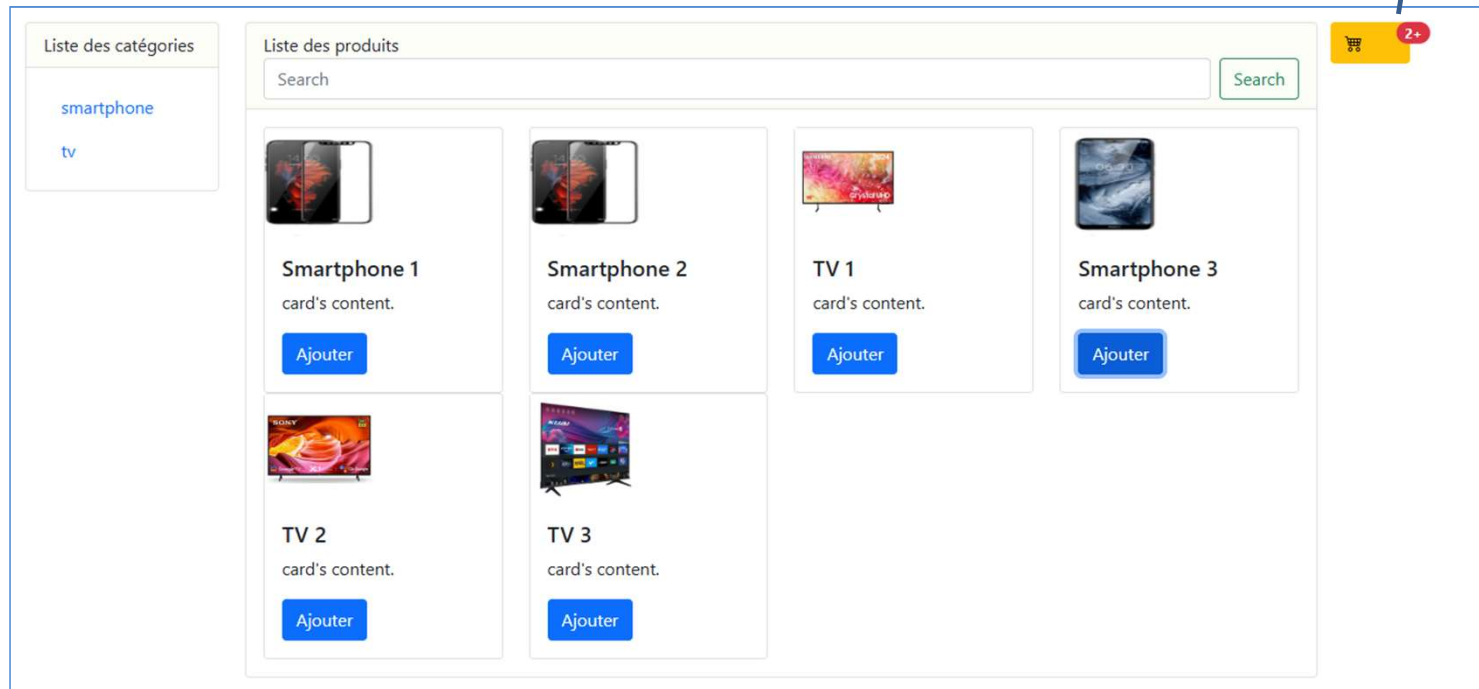
#### Feature 3 : Search by Title

```
const  
produits2=produits.filter(produit=>produit.titre.toUpperCase().startsWith(this.v  
alue.toUpperCase()))
```



#### Feature 4 : Ajouter dans le panier

Mise à jour de panier







# Pratique

## La programmation Orienté Objet en JS

### Etude de cas (Gestion panier site e-commerce)

#### Feature 5 : Afficher le contenu de panier





Panier				
photo	Titre	Prix	Quantité	Action
	TV 1	7000	<input type="text" value="1"/>	
	Smartphone 3	7000	<input type="text" value="1"/>	
Montant :				14000

# Pratique

## La programmation Orienté Objet en JS

### Etude de cas (Gestion panier site e-commerce)

#### Feature 6 : Mise à jour de panier et le montant total

Panier				
photo	Titre	Prix	Quantité	Action
	TV 1	7000	<input type="text" value="2"/>	
	Smartphone 3	7000	<input type="text" value="3"/>	
Montant :				35000