skillup

ООП

План

- Що таке ООП
- Класи на прототипах
- Класи в JS

ООП

Об'єктно-орієнтоване програмування (ООП) - це шаблон проектування програмного забезпечення, який дозволяє вирішувати завдання з точки зору об'єктів і їх взаємодій. JavaScript реалізує ООП через прототипне наслідування

Свої класи на прототипах

```
// конструктор
function Animal(name) {
  this.name = name;
  this.speed = 0;
}

// методи в прототипі
Animal.prototype.run = function(speed) {
  this.speed += speed;
  alert( this.name + ' біжить, швидкість ' + this.speed );
};

let animal = new Animal('Звір');

alert( animal.speed ); // 0, властивість взято з прототипа
animal.run(5); // Звір біжить, швидкість 5
```

Наслідування класів

```
// 1. Конструктор Animal
function Animal(name) {
  this.name = name;
  this.speed = 0;
}

// 1.1. Методи -- в прототип

Animal.prototype.stop = function() {
  this.speed = 0;
  alert( this.name + ' стоит' );
}

Animal.prototype.run = function(speed) {
  this.speed += speed;
  alert( this.name + ' бежит, скорость ' + this.speed );
};
```

```
// 2. Конструктор Rabbit
function Rabbit(name) {
  this.name = name;
  this.speed = 0;
}

// 2.1. Наслідування
Rabbit.prototype = Object.create(Animal.prototype);
Rabbit.prototype.constructor = Rabbit;

// 2.2. Методи Rabbit
Rabbit.prototype.jump = function() {
  this.speed++;
  alert( this.name + ' прыгает, скорость ' + this.speed );
}

let rabbit = new Rabbit('Кроль');
rabbit.run();
```

Додавання своїх методів для примітивів

```
String.prototype.repeat = function(times) {
  return new Array(times + 1).join(this);
};

alert( "ля".repeat(3) ); // ляляля
```

Класи

```
class Название [extends Родитель]   {
  constructor
  методы
}
```

конструктор використовується для створення екземпляра класу з заданими властивостями.

https://learn.javascript.ru/es-class

```
class User {
  constructor(name) {
    this.name = name;
  }
  sayHi() {
    alert(this.name);
  }
}
let user = new User("Вася");  // екземпляр класу
user.sayHi(); // Вася


function User(name) {
  this.name = name;
}
User.prototype.sayHi = function() {
  alert(this.name);
};
```

Наслідування

```
class Child extends Parent {
   ...
}
```

```
class Animal {
  constructor(name) {
    this.name = name;
  }

  walk() {
    alert("I walk: " + this.name);
  }
}

class Rabbit extends Animal {
  walk() {
    super.walk();
    alert("...and jump!");
  }
}

new Rabbit("Вася").walk();
// I walk: Вася
// and jump!
```

Геттери / сеттери

```
class User {
  constructor(name) {
    this.name = name;  // викликаєм сеттер
  }
  get name() {
    return this._name;
  }
  set name(value) {
    if (value.length < 4) {
      alert("Ім’я коротке");
      return;
    }
    this._name = value;
  }
}

let user = new User("John");
alert(user.name); // John

user = new User(""); // Ім’я коротке
```

Практика

- Написати таймер Timer з методами start(), stop(), reset()

Посилання

https://tproger.ru/translations/oop-js-fundamentals/

https://learn.javascript.ru/class-inheritance

https://developer.mozilla.org/ru/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#Object-oriented_programming

Відео

https://www.youtube.com/watch?v=uLY9GXGMXaA

https://www.youtube.com/watch?v=cS6nTVNzOPw