



dedVitalik Adds Lesson 16

15 hours ago



169 lines (103 loc) · 4.3 KB

Preview

Code

Blame

Raw



Появление классов

С выходом ECMAScript 6 появился целый набор ключевых слов, реализующих классы. Они могут показаться знакомыми людям, изучавшим языки, основанные на классах, но есть существенные отличия. JavaScript был и остаётся прототипно-ориентированным языком. Новые ключевые слова: "class", "constructor", "static", "extends" и "super".

Классы в JavaScript представляют собой шаблоны для создания объектов. Они предоставляют удобный способ определения объектов с общими свойствами и методами.

```
class Wizard {  
  constructor(name, house) {  
    this.name = name;  
    this.house = house;  
  }  
  
  introduce() {  
    console.log(`I am ${this.name} from ${this.house} house.`);  
  }  
}  
  
// Создание экземпляра класса  
const harry = new Wizard('Harry Potter', 'Gryffindor');  
harry.introduce(); // "I am Harry Potter from Gryffindor house."
```

Методы класса в JavaScript:

Методы класса - это функции, определенные внутри класса и используемые для выполнения определенных действий.

```
class Wizard {  
  constructor(name, house) {  
    this.name = name;  
    this.house = house;  
  }  
  
  introduce() {  
    console.log(`I am ${this.name} from ${this.house} house.`);  
  }  
  
  castSpell(spell) {  
    console.log(`${this.name} casts ${spell}!`);  
  }  
}  
  
const hermione = new Wizard('Hermione Granger', 'Gryffindor');  
hermione.castSpell('Lumos'); // "Hermione Granger casts Lumos!"
```



Наследование в JavaScript:

Наследование позволяет создавать новые классы, используя свойства и методы существующего класса.

```
class DarkWizard extends Wizard {  
  constructor(name, house, darkPower) {  
    super(name, house);  
    this.darkPower = darkPower;  
  }  
  
  useDarkPower() {  
    console.log(`${this.name} uses dark power: ${this.darkPower}`);  
  }  
}  
  
const voldemort = new DarkWizard('Lord Voldemort', 'Slytherin', 'Avada Keda  
voldemort.introduce(); // "I am Lord Voldemort from Slytherin house."  
voldemort.useDarkPower(); // "Lord Voldemort uses dark power: Avada Kedavra"
```



Геттеры и Сеттеры в JavaScript:

Геттеры используются для получения значения свойства, а сеттеры - для установки его значения.

```
class Wizard {  
  constructor(name, house) {  
    this.#name = name; // Приватное поле  
    this.house = house;  
  }  
  
  get name() {  
    return this.#name;  
  }  
  
  set name(newName) {  
    this.#name = newName;  
  }  
}  
  
const ron = new Wizard('Ron Weasley', 'Gryffindor');  
console.log(ron.name); // "Ron Weasley"  
ron.name = 'Ronald Weasley';  
console.log(ron.name); // "Ronald Weasley"
```



Приватные Поля и Методы в JavaScript:

Приватные поля и методы могут быть созданы с использованием предлагаемого синтаксиса # .

```
class Wizard {  
  #privateField;  
  
  constructor(name, house) {  
    this.name = name;  
    this.house = house;  
    this.#privateField = 'Secret';  
  }  
  
  #privateMethod() {  
    console.log('This is a private method.');  }  
  
  revealSecret() {  
    console.log(`My secret is ${this.#privateField}.`);  
  }  
}
```



```
    this.#privateMethod();  
  }  
}
```

```
const ginny = new Wizard('Ginny Weasley', 'Gryffindor');  
ginny.revealSecret(); // "My secret is Secret." / "This is a private method"
```

Статические Поля и Методы в JavaScript:

Статические поля и методы принадлежат самому классу, а не его экземплярам.

```
class Hogwarts {  
  static schoolMotto = 'Draco Dormiens Nunquam Titillandus';  
  
  static displayMotto() {  
    console.log(`Hogwarts School Motto: ${Hogwarts.schoolMotto}`);  
  }  
}
```



```
Hogwarts.displayMotto(); // "Hogwarts School Motto: Draco Dormiens Nunquam"
```

