

# Классы ES2015/2019



JavaScript  
Courses

[www.courses.dp.ua](http://www.courses.dp.ua)

# Классы в ECMAScript 2015-2019

**Классы** пришли в JavaScript из других (типизированных) языков программирования. В которых классы применяли для описание структуры объектов которые на основе класса создаются. **Класс** выступают своего рода «чертежом» по которому будут создаваться объекты.

Подробнее: <https://learn.javascript.ru/class>

```
class Parcel{
  #code;
  #width;
  #length;
  #height;

  constructor(code, w, l, h){
    this.#code = code;
    this.#width = w;
    this.#length = l;
    this.#height = h;
  }

  getVolume(){
    return this.#width * this.#length * this.#height;
  }

  getReport(){
    return `Parcel ${this.code}: ${this.getVolume()}`;
  }
}

let box = new Parcel(100, 20, 45);

console.log(box.getReport());
```

# Классы в ECMAScript 2015-2019

**Классы** в JavaScript'е являются лишь надстройкой («маскировкой», «синтаксическим сахаром») **прототипной** модели построения объектов. И не являются её заменой.

Подробнее: <https://learn.javascript.ru/class>

# Классы в ECMAScript 2015-2019

По сути описывая **класс** мы создаём функцию **конструктор** в которой идёт перечисление свойств и методом будущего объекта. А далее эта функция вызывается через оператор **new**.

*В ES2019 была добавлена возможность создавать **приватные** (закрытые) свойства и методы. К этим методам есть возможность обратиться только из методов объекта. Из вне они недоступны. Их легко отличить по символу # в начале имени.*

Подробнее: <https://learn.javascript.ru/private-protected-properties-methods>

# Наследование

Класс может расширять функционал (наследовать) другого (родительского) класса, а по сути добавлять в него дополнительные методы и свойства. Для указания этого применяется ключевое слово **extends**. В конструкторе дочернего класса (и в его методах) можно обращаться к конструктору (и методам) класса родителя, для этого применяется ключевое слово **super**.

Подробнее:

<https://learn.javascript.ru/extend-natives>

```
class Person{
  #name;
  #lastName;

  constructor(name, lastName){
    this.#name = name;
    this.#lastName = lastName;
  }
  getInfo(){
    return `${this.#name} ${this.#lastName}`;
  }
}

class Driver extends Person{
  #driverLicense;

  constructor(name, lastName, driverLicense){
    super(name, lastName);
    this.#driverLicense = driverLicense;
  }

  getInfo(){
    return super.getInfo() + `, Driver License: ${this.#driverLicense}`;
  }
}

let driver = new Driver('Jhon', 'Smith', 'AA324356');

console.log(driver.getInfo());
```

# Статические свойства и методы

```
2
3   class Demo{
4       static #counter = 0;
5
6       static getNext(){
7           return ++this.#counter;
8       }
9   }
10
11   console.log( Demo.getNext() );
12   console.log( Demo.getNext() );
13   console.log( Demo.getNext() );
14
```

Статические свойства и методы помечаются ключевым словом **static** к ним можно обращаться без создания экземпляра класса (объекта). Статические свойства и методы хороши для данных которые являются общими для всех объектов класса.

Подробнее: <https://learn.javascript.ru/static-properties-methods>

глобальный объект  
window

# Глобальный объект **window**

Браузер добавляет в JavaScript всего один объект – **window**. Но этот объект содержит все необходимые инструменты для манипуляции HTML-документом.

Подробнее: <https://learn.javascript.ru/global-object>



# Глобальный объект **window**

Объект **window** можно использовать неявно, т.е. опускать его имя при написании кода.

## Свойства и методы window

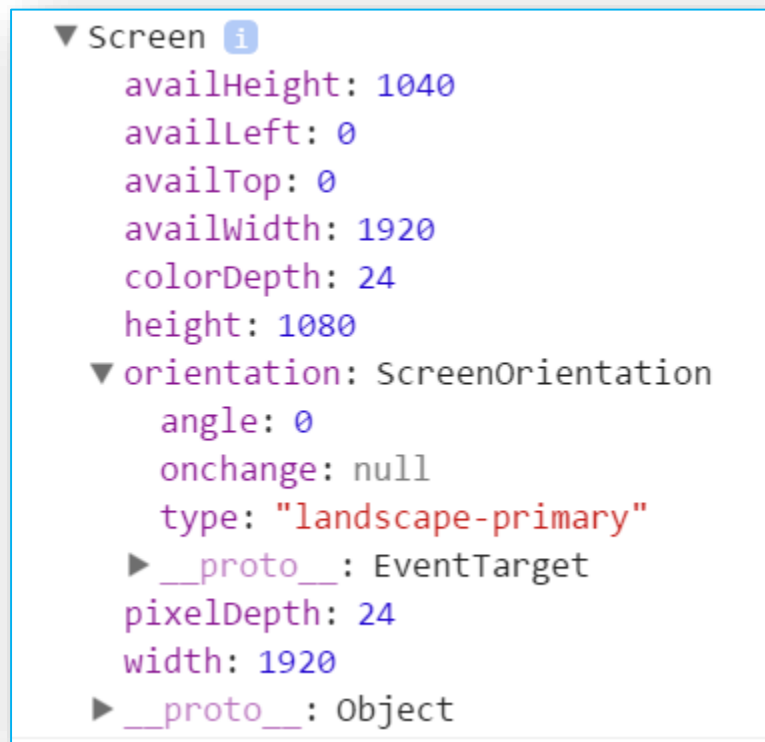
```
.setInterval() ;  
.setTimeout() ;  
.alert() ;  
.prompt() ;  
.confirm() ;  
...
```

# Глобальный объект **window**

```
▼ Location ⓘ
  ▶ ancestorOrigins: DOMStringList {length: 0}
  ▶ origin: "http://localhost:5000"
  ▶ protocol: "http:"
  ▶ host: "localhost:5000"
  ▶ hostname: "localhost"
  ▶ port: "5000"
  ▶ pathname: "/"
  ▶ search: ""
  ▶ hash: ""
  ▶ href: "http://localhost:5000/"
  ▶ assign: f assign()
  ▶ reload: f reload()
  ▶ toString: f toString()
  ▶ replace: f replace()
  ▶ valueOf: f valueOf()
  ▶ Symbol(Symbol.toPrimitive): undefined
  ▶ __proto__: Location
```

**window.location** –  
свойство определяющее  
какую страницу содержит  
окно браузера.

# Глобальный объект **window**



**window.screen** – информация об экране, размерах, ориентации и т.д.

# Глобальный объект **window**

```
▼ Navigator ⓘ
  appCodeName: "Mozilla"
  appName: "Netscape"
  appVersion: "5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/49.0.2623.87 Sa
  cookieEnabled: true
  doNotTrack: null
  ▶ geolocation: Geolocation
  hardwareConcurrency: 4
  language: "ru"
  ▶ languages: Array[4]
  maxTouchPoints: 0
  ▶ mediaDevices: MediaDevices
  ▶ mimeType: MimeTypeArray
  online: true
  ▶ permissions: Permissions
  platform: "Win32"
  ▶ plugins: PluginArray
  ▶ presentation: Presentation
  product: "Gecko"
  productSub: "20030107"
  ▶ serviceWorker: ServiceWorkerContainer
  userAgent: "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/49.0.262
  vendor: "Google Inc."
  vendorSub: ""
  ▶ webkitPersistentStorage: DeprecatedStorageQuota
  ▶ webkitTemporaryStorage: DeprecatedStorageQuota
  __proto__: Object
```

**window.navigator** – информация о браузере.

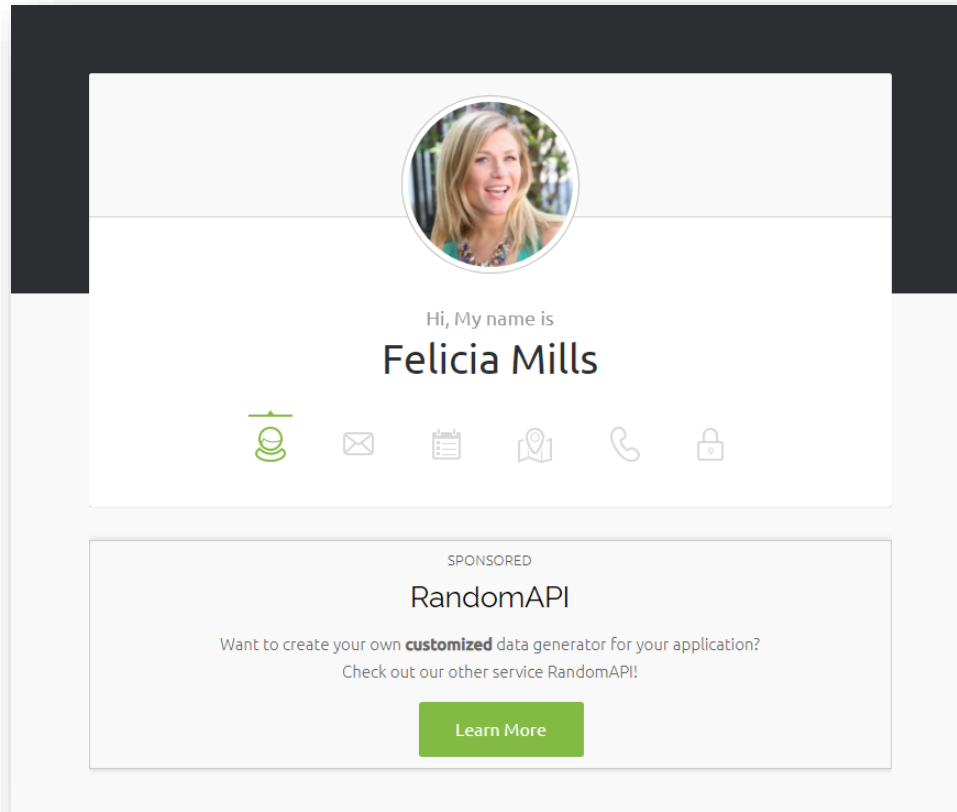
# **window.document** (корень DOM-дерева) хранилище HTML-документа

Подробнее: <https://learn.javascript.ru/dom-nodes>

Воспользуйтесь шаблоном /source/document.html из репозитория этого занятия

# Полезный API-сервис Генерация данных пользователей

# RANDOM USER GENERATOR



Сервис генерирует (фейковые) данные пользователей. Что может быть крайне полезным для наполнения и тестирования страниц наших приложений.

<https://randomuser.me/>

<https://randomuser.me/documentation#multiple>

Домашнее задание  
/узнать



# Современный учебник Javascript

Перед вами учебник по JavaScript, начиная с основ, включающий в себя много тонкостей и фишек JavaScript/DOM.

[смотреть на Github](#)

Поделиться:

[НАЙТИ](#)

## Содержание

Первые две части посвящены JavaScript и его использованию в браузере. Затем идут дополнительные циклы статей на разные темы.

**Предварительные знания – лучший помощник** в обучении, поэтому к следующему занятию жду, что **пройдёте разделы 2-й части 1.1-1.11**

<http://learn.javascript.ru/>