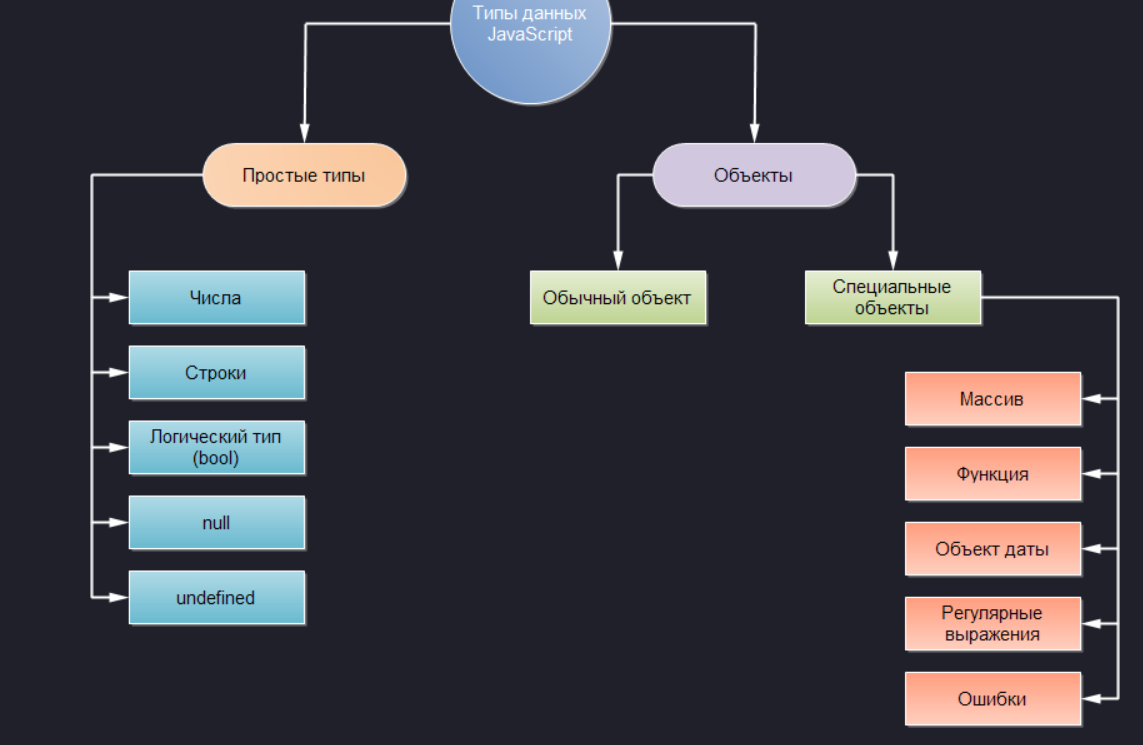
1. Как объявляются переменные в языке JavaScript?

Тип данных Название переменной

1. В чем разница между ключевыми словами var и let?

**let** позволяет объявить локальную переменную с областью видимости, ограниченной текущим блоком кода . В отличие от **ключевого слова var**, которое объявляет переменную глобально или локально во всей функции, независимо от области блока.

1. Какие элементарные типы данных присутствуют в языке JavaScript?



Все используемые данные в javascript имеют определенный тип. В JavaScript имеется восемь типов данных:

* **String**: представляет строку
* **Number**: представляет числовое значение
* **BigInt**: предназначен для представления очень больших целых чисел
* **Boolean**: представляет логическое значение **true** или **false**
* **Undefined**: представляет одно специальное значение - **undefined** и указывает, что значение не установлено
* **Null**: представляет одно специальное значение - **null** и указывает на отсутствие значения
* **Symbol**: представляет уникальное значение, которое часто применяется для обращения к свойствам сложных объектов
* **Object**: представляет комплексный объект

1. Как определить тип переменной?

С помощью оператора typeof можно получить тип переменной. Стоит отметить, что для значения null оператор typeof возвращает значение "object", несмотря на то, что согласно спецификации JavaScript значение null представляет отдельный тип.

1. Как происходит приведение типов?

Чаще всего операторы и функции автоматически приводят переданные им значения к нужному типу.

Всего есть три преобразования:

1. Строковое преобразование.

String(value)

1. Численное преобразование.

Number(value)

Изображение выглядит как текст

Автоматически созданное описание

1. Преобразование к логическому значению.

Boolean(value)

1. Можно ли объявить анонимную функцию?

Запомни! Если после **function** есть имя — функция именованная, во всех остальных случая анонимная.

Вызвать ее можно

function sayAfter(text,delay){

setTimeout(function(){

alert(text);

},delay);

}

sayAfter("lol",1000);

1. Как создать массив в языке JavaScript?

Существует два варианта синтаксиса для создания пустого массива:

letarr=newArray();

letarr=[];

1. Назовите назначение и способы использования JavaScript-кода на HTML-страницах. Что такое бумарклет?

JavaScript-код может позволяет вам применять сложные вещи на web странице.Отображение периодически обновляемого контента, или интерактивных карт, или анимация 2D/3D графики, или прокрутка видео в проигрывателе, и т.д.

Клиентский JavaScript-код может встраиваться в HTML-документы четырьмя способами:

* встроенные сценарии между парой тегов <script> и </script>;
* в обработчик события, заданный в качестве значения HTML-атрибута, такого как onclick или onmouseover;
* как тело URL-адреса, использующего специальный спецификатор псевдопротоколаJavaScript:.

Букмарклет — это специальная закладка. От обычной закладки букмарклет отличается тем, что содержит не ссылку на документ, а специальный JavaScript-код.

1. Атрибуты тега <script>.

**Атрибуты**

[async](http://htmlbook.ru/html/script/async) -- Загружает скрипт асинхронно.

[defer](http://htmlbook.ru/html/script/defer) -- Откладывает выполнение скрипта до тех пор, пока вся страница не будет загружена полностью.

[language](http://htmlbook.ru/html/script/language) -- Устанавливает язык программирования на котором написан скрипт.

[src](http://htmlbook.ru/html/script/src) -- Адрес скрипта из внешнего файла для импорта в текущий документ.

[type](http://htmlbook.ru/html/script/type) -- Определяет тип содержимого тега **<script>**.

1. Как избежать преждевременного выполнения скрипта?

Необходимо помещать код в самый конец тела документа (т.е. перед </body>) в данном случае интерпретатор JavaScript встроенный в браузер начнет разбирать код только **после загрузки** документа.

1. Что делает метод document.write?

Метод document.write вставляет любой текст на страницу «как есть», в то время как innerHTML может вписать лишь валидный HTML (при попытке подсунуть невалидный – браузер скорректирует его).

1. Методы для работы с датой и временем. Как получить текущий месяц?

Для работы с датой и временем в JavaScript используются объекты [Date](https://developer.mozilla.org/en/JavaScript/Reference/Global_Objects/Date/).

Для создания нового объекта типа Date используется один из синтаксисов:

**new Date()**

Пример

var now =newDate();

Методы:

**getFullYear()**

Получить год (из 4 цифр)

**getMonth()**

Получить месяц, **от 0 до 11**.

**getDate()**

Получить число месяца, от 1 до 31.

**getHours(), getMinutes(), getSeconds(), getMilliseconds()**

Получитьсоответствующиекомпоненты.

Получениятекущегомесяца

varm\_names=['January','February','March',

'April','May','June','July',

'August','September','October','November','December'];

d =newDate();

var n =m\_names[d.getMonth()];

1. Что такое объектная модель документа?

**DOM** (от [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *DocumentObjectModel* — «объектная модель документа») — это независящий от платформы и языка [программный интерфейс](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B9), позволяющий [программам](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0) и [скриптам](https://ru.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B8%D0%BF%D1%82) получить доступ к содержимому [HTML](https://ru.wikipedia.org/wiki/HTML)-, [XHTML](https://ru.wikipedia.org/wiki/XHTML)- и [XML](https://ru.wikipedia.org/wiki/XML)-документов, а также изменять содержимое, структуру и оформление таких документов.

Объектная Модель Документа (DOM) – это программный интерфейс (API) для HTML и XML документов. DOM предоставляет структурированное представление документа и определяет то, как эта структура может быть доступна из программ, которые могут изменять содержимое, стиль и структуру документа. Представление DOM состоит из структурированной группы узлов и объектов, которые имеют свойства и методы. По существу, DOM соединяет веб-страницу с языками описания сценариев либо языками программирования.

1. Свойства и методы объекта window.

# Свойства объекта Window

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| [closed](http://www.wisdomweb.ru/JSd/wiclosed.php) | Возвращает логическое значение (true или false) в зависимости от того, закрыто указанное окно или отрыто. |
| [frames](http://www.wisdomweb.ru/JSd/wiframes.php) | Возвращает массив всех фреймов на странице (включая iframe). |
| [document](http://www.wisdomweb.ru/HDOMd/document.php) | Возвращает объект Document данного окна. |
| [history](http://www.wisdomweb.ru/JSd/history.php) | Возвращает объект History данного окна. |
| [length](http://www.wisdomweb.ru/JSd/wilength.php) | Возвращает количество фреймов (включая iframe), которые находятся в данном окне. |
| [location](http://www.wisdomweb.ru/JSd/location.php) | Возвращает объект Location данного окна. |
| [name](http://www.wisdomweb.ru/JSd/winame.php) | Устанавливает или возвращает имя данного окна. |
| [navigator](http://www.wisdomweb.ru/JSd/navigator.php) | Возвращает объект Navigator данного окна. |
| [opener](http://www.wisdomweb.ru/JSd/wiopener.php) | Возвращает ссылку на окно, которое открыло данное. |
| [parent](http://www.wisdomweb.ru/JSd/wiparent.php) | Возвращает родительское окно данного окна. |
| [screen](http://www.wisdomweb.ru/JSd/screen.php) | Возвращает объект Screen данного окна. |
| [self](http://www.wisdomweb.ru/JSd/wiself.php) | Возвращает текущее окно. |
| [top](http://www.wisdomweb.ru/JSd/witop.php) | Возвращает верхнее браузерное окно для данного окна. |

# Методы объекта Window

|  |  |
| --- | --- |
| **Метод** | **Описание** |
| [alert()](http://www.wisdomweb.ru/JSd/wialert.php) | Вызывает окно оповещения, которое содержит текст сообщения и клавишу ОК. |
| [blur()](http://www.wisdomweb.ru/JSd/wiblur.php) | Делает окно неактивным. |
| [clearInterval()](http://www.wisdomweb.ru/JSd/wiclinterval.php) | Прекращает повторное выполнение кода заданного setInterval(). |
| [clearTimeout()](http://www.wisdomweb.ru/JSd/wicltimeout.php) | Отменяет запланированное методом setTimeout() выполнение кода. |
| [close()](http://www.wisdomweb.ru/JSd/wiclose.php) | Закрывает окно. |
| [confirm()](http://www.wisdomweb.ru/JSd/wiconf.php) | Вызывает окно подтверждения содержащее текст сообщения и клавиши ОК и Отмена. |
| [focus()](http://www.wisdomweb.ru/JSd/wifocus.php) | Делает окно активным. |
| [moveBy()](http://www.wisdomweb.ru/JSd/wimoveby.php) | Смещает окно относительно его текущей позиции. |
| [moveTo()](http://www.wisdomweb.ru/JSd/wimoveto.php) | Перемещает окно на указанную позицию. |
| [open()](http://www.wisdomweb.ru/JSd/wiopen.php) | Открывает новое окно браузера. |
| [print()](http://www.wisdomweb.ru/JSd/wiprint.php) | Распечатывает содержимое текущего окна. |
| [prompt()](http://www.wisdomweb.ru/JSd/wiprompt.php) | Вызывает окно запроса, побуждающее посетителя ввести в него определенные данные. |
| [scrollBy()](http://www.wisdomweb.ru/JSd/wiscrollby.php) | Прокручивает содержимое окна на указанное количество пикселей. |
| [scrollTo()](http://www.wisdomweb.ru/JSd/wiscrollto.php) | Прокручивает содержимое окна до указанных координат. |
| [setInterval()](http://www.wisdomweb.ru/JSd/wisetint.php) | Вызывает функцию или выполняет код через определенные промежутки времени (указанные в миллисекундах). |
| [setTimeout()](http://www.wisdomweb.ru/JSd/wisetimeout.php) | Вызывает функцию или выполняет код после указанного количества миллисекунд один раз. |

1. Каким образом можно использовать объект Location?

Объект **location** содержит информацию о расположении текущей веб-страницы: URL, информацию о сервере, номер порта, протокол. С помощью свойств объекта мы можем получить эту информацию:

* **href**: полная строка запроса к ресурсу
* **pathname**: путь к ресурсу
* **origin**: общая схема запроса
* **protocol**: протокол
* **port**: порт, используемый ресурсом
* **host**: хост
* **hostname**: название хоста
* **hash**: если строка запроса содержит символ решетки (#), то данное свойство возвращает ту часть строки, которая идет после этого символа
* **search**: если строка запроса содержит знак вопроса (?), например, то данное свойство возвращает ту часть строки, которая идет после знака вопроса

Изображение выглядит как текст

Автоматически созданное описание

1. Какой BOM-объект содержит информацию о браузере пользователя?

**объектная модель браузера (BOM – Browser Object Model)**

**глобальный объект window**

1. Чем отличается setTimeout от setInterval?

Мы можем вызвать функцию не в данный момент, а позже, через заданный интервал времени. Этоназывается «планированиевызова».

Дляэтогосуществуютдваметода:

* **setTimeout** позволяет вызвать функцию **один раз** через определённый интервал времени.
* **setInterval** позволяет вызывать функцию **регулярно**, повторяя вызов через определённый интервал времени.

1. Методы для получения объектов страницы.

* циклы [for...in (en-US)](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/for...in" \o "Currently only available in English (US))  
  Этот метод перебирает все перечисляемые свойства объекта и его цепочку прототипов
* [Object.keys(o) (en-US)](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/keys)  
  Этот метод возвращает массив со всеми собственными (те, что в цепочке прототипов, не войдут в массив) именами перечисляемых свойств объекта o.
* [Object.getOwnPropertyNames(o) (en-US)](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/getOwnPropertyNames)  
  Этот метод возвращает массив содержащий все имена своих свойств (перечисляемых и неперечисляемых) объекта o.

1. Для чего используется библиотека jQuery?

данная библиотека позволяет изменять содержимое HTML-документов путем манипулирования объектами модели, создаваемой браузерами в процессе обработки HTML-кода (так называемые DOM-манипуляции)

1. Какие элементы на странице будут выбраны jQuery селектором $("p#test")?

Будут выбраны все элементы p на странице сid =”test”

1. В чем разница между методами .after() и .insertAfter() библиотеки jQuery?

**.insertAfter( target )**

Новый контент будет вставлен после указанного в **target**

#### .after( content [, content] )

Cразу после выбранных элементов будет добавлен **content**, который может быть задан html

"**after**" вставляет аргумент после селектора.

'**insertAfter**' вставляет селектор после аргумента.

1. Возможно ли чтение из файла, расположенного локально на стороне клиента?

Да можно

Существует такая вещь, как **HTML5 File API** для доступа к локальным файлам, выбранным пользователем, без их загрузки в любом месте.

Есть одна проблема с этим, вы не **можете читать большие файлы** (~400 MB и больше), потому что прямые функции FileAPI пытаются загрузить весь файл в память.

1. Способы создания объектов в языке JavaScript.

JavaScript предоставляет разработчикам возможность создавать объекты и работать с ними. Дляэтогосуществуютследующиеприёмы:

* Оператор new
* Литеральнаянотация
* Конструкторыобъектов
* Ассоциативныемассивы

**оператор new**

Это, наверное, самый легкий способ создания объекта. Вы просто создаете имя объекта и приравниваете его к новому объекту Javascript.

Изображение выглядит как текст

Автоматически созданное описание

## **Литеральнаянотация**

Изображение выглядит как текст

Автоматически созданное описание

## **Конструкторы объектов**

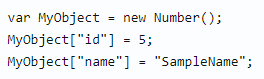
Конструктор объекта — это, по сути, обычная функция Javascript, которой так же можно передавать различные параметры

.Изображение выглядит как текст

Автоматически созданное описание

**Ассоциативные массивы**

Подобный метод будет полезен упорядочивания большого числа однотипных объектов.



1. Что такое JSON?

**JSON**(англ. JavaScript Object Notation) — текстовый формат обмена данными, основанный на JavaScript. Но при этом формат независим от JS и может использоваться в любом языке программирования.

JSON используется в REST API.

Он обычно используется для передачи данных в веб-приложениях (например, отправка некоторых данных с сервера клиенту,таким образом чтобы это могло отображаться на веб-странице или наоборот).

1. Как происходит добавление свойств и методов в объект?

Чтобы добавить новое свойство существующему объекту, нужно просто присвоить ему значение.

Предположим, что объект person уже существует. Вы можете добавить ему новое свойство:

person.nationality="English";

1. Поясните понятие замыкания.

## **Замыкания**

Всякий раз, когда вы вызываете функцию внутри другой функции, вы создаете замыкание. Говорят, что внутренняя функция является замыканием. Результатом замыкания обычно является то, что в дальнейшем становятся доступными переменные внешней функции.

**functionouterFunction** () {

**const** outer = `I see the outer variable!`;

**returnfunctioninnerFunction**() {

console.log(outer);

}

}

outerFunction()(); // I see the outer variable!

1. Что обозначает ключевое слово this, и как оно соотносится с областью видимости переменной?

This – означает, что мы обращаемся с его помощью к некоему объекту.

Если вы попытаетесь обратиться к ключевому слову this в глобальной области видимости, оно будет привязано к глобальному контексту, то есть — к объекту window в браузере.

Пример:console.log(this);

## **Использование this внутри объекта**

Когда this используется внутри объекта, это ключевое слово ссылается на сам объект.

**var** dog = {

 name: 'Chester',

 breed: 'beagle',

 intro: **function**(){

   console.log(**this**);}};

this в стрелочных функциях относится к глобальному контексту (к объекту window).

Пример

**var**objArrow = {

   hello: () =>**this**

};objArrow.hello(); // возвращает объект Window!

## Ключевоеслово this

* this - это объект, свойством которого является функция;
* this - дает функциям доступ к своему объекту и его свойствам;
* this - помогает выполнить один и тот же код для нескольких объектов;
* this - можно рассматривать как **кто меня вызвал?**; т.е. то, что находится слева от точки. Например, window.a();
* this - имеет **динамическую область**, т. е. не важно, где он был написан, важно, где он был вызван.

1. Реализация принципов ООП в языке JavaScript.

### Инкапсуляция

Инкапсуляция включает в себя идею о том, что данные объекта не должны быть напрямую доступны. Нужно вызывать методы вместо прямого доступа к данным. Инкапсуляция позволяет нам скрывать/показывать свойства функций.

**Абстракция** - это способ создания простой модели, которая содержит только важные свойства с точки зрения контекста приложения, из более сложной модели. Иными словами - это способ скрыть детали реализации и показать пользователям только функциональность.

### Полиморфизм

Полиморфизмом является одним из принципов объектно-ориентированного программирования (ООП). Это помогает проектировать объекты таким образом, чтобы они могли совместно использовать или переопределять любое поведение с конкретными предоставленными объектами.

**Наследование** - это механизмбазирования **объекта** или **class** на другом объекте (наследование на основе прототипа) или **class** (наследование на основе класса). Мы избегаем необходимости переписывать один и тот же код, а также экономим пространство памяти, используя общие методы.

1. Как реализуется наследование?

В JavaScript выделяют:

1. функциональное наследование (через функции-конструкторы и классы class - с особенностями);
2. прототипное наследование. (выполняется через процесс повторного использования существующих объектов посредством **делегирования**, которые служат как **prototypes**.)

Функциональное наследование использует вызов родительской функции-конструктора внутри функции-конструктора дочернего класса с одновременной передачей ей в качестве контекста **this** текущего объекта:

functionUser(firstname,lastName){

  this.firstname = firstname;

  this.lastName = lastName;

  this.getFullName = function(){

    return`${firstname}${lastName}`;

  };

}

Наследование в JavaScript - динамическое и не включает в себя реализацию понятия class

В плане наследования JavaScript работает лишь с одной сущностью: объектами.

1. Что такое инкапсуляция? Как она реализуется в языке JavaScript?

Инкапсуляция – это объединение функций и данных в рамках одной структуры, внутреннее состояние которой (данные) скрыто от внешнего мира (этот аспект мы разберем позже). Такие функции называют методами.

*// Вызов метода*

user.getName();

*// Вызов функции*

getName(user);

1. Как использовать элемент canvas?

Создает область, в которой при помощи JavaScript можно рисовать разные объекты, выводить изображения, трансформировать их и менять свойства. При помощи тега **<canvas>** можно создавать рисунки, анимацию, игры и др.

### Синтаксис

**<canvas** id="идентификатор"**>**

**</canvas>**

**Атрибуты**

HeightЗадает высоту холста. По умолчанию 300 пикселов.

WidthЗадает ширину холста. По умолчанию 150 пикселов.

1. Для чего служит объект Notification?

Интерфейс Notification из [Notifications API](https://developer.mozilla.org/en-US/docs/Web/API/Notifications_API) используется для настройки и отображения уведомлений на рабочий стол пользователя.

Notifications API позволяют web-страницам контролировать отображение системных уведомлений для конечного пользователя — они находятся вне контекста окна браузера верхнего уровня, поэтому могут отображаться даже если пользователь сменил вкладку или перешёл к другому приложению.