Front-end

JavaScript

Преподаватель: Стельмашок Евгений Олегович

JS. Intro

- Как работает браузер
- О JavaScript
- Синтаксис
- Линтер
- Типы данных и переменные
- Ввод и вывод данных (alert, confirm, prompt, console)
- Операторы typeof

JS. Браузер и рендеринг

- DOM
- CSSOM
- Render Tree
- Layout
- Painting

https://habr.com/ru/articles/224187/

https://developer.mozilla.org/ru/docs/Web/Performance/How_browsers_work

https://webdevblog.ru/kak-brauzer-renderit-veb-stranicu/

https://habr.com/ru/articles/484900/

JS. История

В 1995 году Брендан Эйх получил задачу внедрить язык программирования в браузер Netscape. Изначально язык назывался Mocha, затем LiveScript. Наконец, он получил свое современное имя – JavaScript. Здесь разработчики пошли на хитрость. В то время, когда они занимались улучшением LiveScript, довольно большой популярностью пользовался язык Java. Для того, чтобы привлечь больше разработчиков для работы с новым языком, было решено использовать в его названии Java. В итоге получился JavaScript.

Выполнение JS-кода — однопоточное. Это значит, что в конкретный момент времени движок может выполнять не более одной строки кода. То есть вторая строка не будет выполнена, пока не выполнится первая. Такое выполнение кода (строка за строкой) называется синхронным.

JS. Особенности

Динамическая типизация

JavaScript является слабо типизированным или динамическим языком. Это значит, что вам не нужно определять тип переменной заранее. Тип определится автоматически во время выполнения программы. Также это значит, что вы можете использовать одну переменную для хранения данных различных типов.

Код JavaScript выполняется JavaScript-движком браузера, после того как код HTML и CSS был обработан и сформирован в веб-страницу.

JS. Особенности

К основным особенностям этого языка программирования относятся:

- динамическая типизация. То есть тип данных будет определяться только тогда, когда переменной или const будет присваиваться ее значение.
- гибкая работа с функциями. В JS функции можно не только выполнять, но еще и возвращать функции из функций, передавать функции в качестве параметров другим функциями и присваивать функции в качестве значения переменных.
- JavaScript поддерживается всеми современным браузерами.
- объектно-ориентированное программирование. То есть это такая методология программирования, в которой вся программа представляется в виде совокупности объектов.

JS. Особенности

Возможности

- добавлять различные эффекты анимации
- реагировать на события - обрабатывать перемещения указателя мыши, нажатие клавиш с клавиатуры
- осуществлять проверку ввода данных в поля формы до отправки на сервер, что в свою очередь снимает дополнительную нагрузку с сервера
- создавать и считывать cookie, извлекать данные о компьютере посетителя
- определять браузер
- изменять содержимое HTML-элементов, добавлять новые теги, изменять стили

Этим конечно же список не ограничивается, так как помимо перечисленного JavaScript позволяет делать и многое другое.

JS. Особенности

Ограничения

- не может закрывать окна и вкладки, которые не были открыты с его помощью
- не может защитить исходный код страницы и запретить копирование текста или изображений со страницы
- не может осуществлять кроссдоменные запросы, получать доступ к веб-страницам, расположенным на другом домене. Даже когда страницы из разных доменов отображаются в одно и тоже время в разных вкладках браузера, то код JavaScript принадлежащий одному домену не будет иметь доступа к информации о веб-странице из другого домена. Это гарантирует безопасность частной информации, которая может быть известна владельцу домена, страница которого открыта в соседней вкладке
- не имеет доступа к файлам, расположенным на компьютере пользователя, и доступа за пределы самой веб-страницы, единственным исключением являются файлы cookie, это небольшие текстовые файлы, которые JavaScript может записывать и считывать

JS. Синтаксис

Комментарии:

- **однострочные комментарии**

Однострочные комментарии начинаются с //. После этих двух символов может следовать любой текст, вся строчка не будет анализироваться и исполняться.

- **многострочные комментарии**

Многострочные комментарии начинаются с /* и заканчиваются на */

JS. Синтаксис

Кавычки:

Любой одиночный символ в кавычках — это строка. Пустая строка '' — это тоже строка. То есть строкой мы считаем всё, что находится внутри кавычек, даже если это пробел, один символ или вообще отсутствие символов.

Можно использовать одинарные, двойные и обратные кавычки - "Dracarys!" или 'Dragon's mother', `Spider man`

Экранирование:

\ - символ экранирования

JS. Синтаксис

Точка с запятой

В JavaScript точки с запятой являются необязательными. Однако есть ситуации, в которых пропуск точки с запятой может привести к нежелательным последствиям.

Регистр

JavaScript - это язык, чувствительный к регистру символов. Это значит, что ключевые слова, имена переменных и функций и любые другие идентификаторы языка должны всегда содержать одинаковые наборы прописных и строчных букв.

JS. Синтаксис

Нейминг

- имя переменной должно максимально чётко соответствовать хранимым в ней данным
- никакого транслита. Только английский
- переменные из нескольких слов пишутся слитно

camelCase - стиль написания составных слов, при котором несколько слов пишутся слитно без пробелов, при этом каждое слово внутри фразы пишется с прописной буквы

JS. Линтер

Код программы следует оформлять определенным образом, чтобы он был достаточно понятным и простым в поддержке. Специальные наборы правил — стандарты — описывают различные аспекты написания кода.

Конкретно в JavaScript самым распространенным стандартом является стандарт от AirBnb.

В любом языке программирования существуют утилиты — так называемые **линтеры**. Они проверяют код на соответствие стандартам. В JavaScript это eslint.

JS. Types

Стандарт ECMAScript определяет 8 типов:

6 типов данных являющихся примитивами

- Number
- String
- Boolean
- Undefined
- BigInt
- Symbol

JS. Types

Примитивные значения

Все типы данных в JavaScript, кроме объектов, являются иммутабельными (значения не могут быть модифицированы, а только перезаписаны новым полным значением). Например, в отличии от C, где строку можно посимвольно корректировать, в JavaScript строки пересоздаются только полностью. Значения таких типов называются «примитивными значениями».

JS. Types

- Number
- String
- Boolean
- Null
- Undefined
- BigInt
- Symbol
- Object

Тип данных	Пример	Описание
Number	15	Целое число или число с плавающей запятой
String	“Hello”	Последовательность символов в кавычках
Boolean	true	Значение true (1) или false (0)
Null	null	Отсутствие какого-либо объектного значения
Undefined	undefined	Пустое значение
BigInt	123n	Большие числа
Symbol	Symbol	Уникальный идентификатор
Object	{}	Пользовательский или встроенный объект

JS. Types

Для представления значения в JavaScript можно использовать литералы. Они являются фиксированными значениями, не переменными, которые Вы литерально\буквально предоставляете в Вашем скрипте.

Литерал массива это заключённый в квадратные скобки ([]) список из нуль или более выражений, каждое из которых представляет элемент массива. Если Вы создаёте массив с использованием литерала массива, этот массив инициализируется специфицированными значениями в качестве элементов, и его размер равен количеству специфицированных аргументов.

JS. Types

Тип Boolean имеет два литеральных значения: true и false.

Литерал объекта это заключённый в фигурные скобки ({}) список из 0 или более пар свойств объекта и ассоциированных с ними значений.

Строковой литерал это 0 или более символов, заключённых в двойные (") или одинарные (') кавычки. Строка обязана быть ограничена кавычками одного вида; то есть, оба знака – двойные, или оба знака – одинарные кавычки.

JS. Variables

- **let**
- **const**
- **var**

Общие правила построения для имен переменных (уникальных идентификаторов):

- имена переменных могут содержать буквы, цифры, символы подчеркивания и знаки доллара
- имена переменных должны начинаться с буквы
- имена переменных, также могут начинаться с $ и _
- имена переменных чувствительны к регистру (y и Y - разные переменные)
- зарезервированные слова (например, ключевые слова JavaScript) нельзя использовать в качестве переменных имен

JS. Input and Output. Console

console - объект console служит для доступа к средствам отладки браузера. Чаще всего Консоль используется для вывода строк текста и других типов данных.

console.log() — это метод, предназначенный для печати в консоль браузера. При написании скриптов иногда нужно увидеть промежуточный результат прямо в консоли браузера — это просто, удобно и не требует никакой дополнительной логики для отображения.

JS. Input and Output. Alert

alert - функция **alert()** предназначена для вывода в браузере предупреждающего модального диалогового окна с некоторым сообщением и кнопкой **«OK»**. При его появлении дальнейшее выполнение кода страницы прекращается до тех пор, пока пользователь не закроет это окно. Кроме этого, оно также блокирует возможность взаимодействия пользователя с остальной частью страницы. Применение этого окна в основном используется для вывода некоторых данных при изучении языка JavaScript, в реальных проектах команда **alert()** не используется.

Метод **alert()** имеет один аргумент **(message)** - текст сообщения, которое необходимо вывести в модальном диалоговом окне. В качестве результата **alert()** ничего не возвращает.

alert() позволяет вывести любое сообщение, но необходимо помнить, что аргумент будет приведён к строке.

JS. Input and Output. Prompt

prompt - метод **prompt()** предназначен для вывода диалогового окна с сообщением, текстовым полем для ввода данных и кнопками **«OK»** и **«Отмена»**. Это окно предназначено для запроса данных, которые пользователю нужно ввести в текстовое поле.

Синтаксис:

message - текст сообщения (является не обязательным), предназначено для информирования пользователя о том, какие данные у него запрашиваются

default - начальное значение для поля ввода, которое будет по умолчанию в нём отображаться (является не обязательным)

```
const result = prompt(message, default);
```

В переменную **result** возвращается значение введённое пользователем или **null**.

Если пользователь не ввёл данные (поле ввода пустое) и нажал на **«OK»**, то в **result** будет находиться пустая строка.

JS. Input and Output. Confirm

confirm - метод **confirm()** применяется для вывода модального диалогового окна с сообщением и кнопками **«OK»** и **«Отмена»**. Оно обычно используется для запроса у пользователя разрешения на выполнение того или иного действия.

Синтаксис метода confirm():

```
const result = confirm(question)
```

question - текст сообщения (вопроса)

В переменную **result** возвращается:

- **true** - если пользователь нажал на кнопку **«OK»**
- **false** - в остальных случаях

JS. typeof

Оператор typeof

Оператор typeof возвращает тип аргумента.

У него есть два синтаксиса: со скобками и без:

Синтаксис оператора: **typeof x**

Синтаксис функции: **typeof(x)**

Работают они одинаково, но первый синтаксис короче.

Результатом typeof является строка, содержащая тип

JS. Понятия и определения

- **Рендеринг** - визуализация
- **Интерпретатор** — программа, выполняющая код на JavaScript.
- **Инструкция (statement)** — команда для компьютера, написанная на языке программирования. Код на JavaScript — это набор инструкций, разделенных (чаще всего) символом ;
- **Комментарий** — текст в коде программы, который не влияет на функциональность и добавляется программистами для себя и своих коллег.
- **Синтаксическая ошибка** — нарушение грамматических правил языка программирования.
- **SyntaxError (ошибка парсинга)** — тип ошибок в JavaScript, возникающих при наличии синтаксических ошибок в коде.
- **Инструкция** — наименьшая автономная часть языка программирования; команда или набор команд. Программа обычно представляет собой последовательность инструкций.
- **Арифметическая операция** — сложение, вычитание, умножение и деление.
- **Унарная операция** — операция с одним операндом. Например, -3 — унарная операция для получения числа, противоположного числу три.
- **Бинарная операция** — операция с двумя операндами. Например, 3 + 9.

JS. Понятия и определения

- **Композиция** — метод объединения нескольких простых операций в одну сложную.
- **Выражение** — последовательность действий над данными, приводящая к каком-то результату, который можно использовать.
- **Экранирующая последовательность** — специальная комбинация символов в тексте. Например, \n — это перевод строки.
- **Конкатенация** — операция соединения двух строк. Например, console.log("King's " + ' Landing');
- **Переменная** — способ сохранить информацию и дать ей имя для последующего использования в коде.
- **Стандарт кодирования** — Набор синтаксических и стилистических правил написания кода.
- **Интерполяция** — способ соединения строк через вставку значений переменных в строку-шаблон с помощью фигурных скобок. Например, `Hi, ${name}!`.
- **Индекс** — позиция символа внутри строки.

JS. Понятия и определения

- **Примитивные типы данных** — простые типы, встроенные в сам язык программирования.
- **undefined** — аналог отсутствия значения; указывает, что переменной не присвоено значение или она вообще не объявлена.
- **Слабая типизация** — это типизация, при которой язык программирования выполняет множество неявных преобразований типов автоматически, даже если может произойти потеря точности или преобразование неоднозначно.
- **Функция** — операция, способная принимать данные и возвращать результат; функция вызывается так: foo().
- **Аргумент** — информация, которую функция получает при вызове. Например, foo(42) — передача аргумента 42 функции foo()
- **Параметр по умолчанию** — необязательный параметр функции
- **Выражение** — последовательность действий над данными, приводящая к каком-то результату, который можно использовать.
- **Побочный эффект** — действие, которое изменяет внешнее окружение (среду выполнения).
- **Метод** — это функция или процедура, принадлежащая какому-то классу или объекту.