Изучение JavaScript

План

**1.Основы JavaScript**

*1.1 Типы данных*

1.1.1 Number

1.1.2 String

1.1.3 Boolean

1.1.4 Null

1.1.5 Undefined

1.1.6 Object

1.1.7 Symbol

1.1.8 BigInt

*1.2 Операторы*

*1.3 Условия*

*1.4 Циклы*

*1.5 Функции*

1.5.1 Классификация функции

1.5.2 Замыкание функции

1.5.3 callback-функции

1.6 ООП

1.7 Динамическая типизация

**2. Работа со страницей**

*2.1 Получение элементов*

*2.2 События*

2.2.1 Всплытие события

2.2.2 Делегирование события

2.2.3 Объект события

2.2.4 Цикл события

*2.3 Обработчик события*

*2.4 Отмена стандартных событий браузера*

*2.5 Параметры глобальных объектов*

2.5.1 window

2.5.2 screen

2.5.3 document

*2.6 JS параметры элементов*

*2.7 Работа скриптов с течением времени*

**3.Регулярные выражения**

**Введение**

*Основы*

*Прямых const в JS не бывает*

*Стилистические правила*

**

« ; » ставится после объявления переменной и когда закончился какой то логический блок

**Название и типы названия переменных**

*(Дополнительно)*

Называть переменные чтоб мы знали, что там лежит

Кемелкейс – когда каждое новое слово начинается с большой буквы кроме первого

UPPER\_SNAKE\_CASE – переменные которые программисты хотят видеть в виде констант ( const ) их никогда нельзя переопределять, классический пример это цвета const COLOR\_RED = ‘#f00’; НЕЛЬЗЯ ИЗМЕНЯТЬ ЗНАЧЕНИЕ ПЕРЕМЕННОЙ

В начале « \_ »так же не изменять пример const \_apiBase = ……..

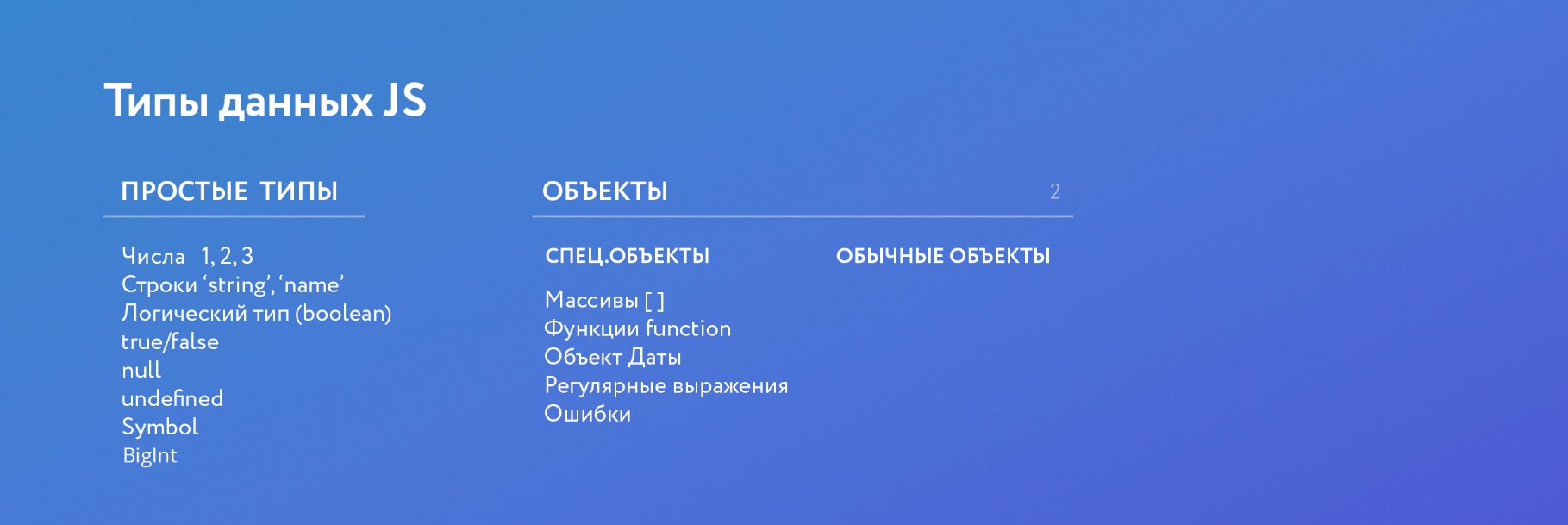
Kebab-case каждое слово начинается с маленькой буквы через дефиз

PascalCase – для названия класссов в JS , первая буква всегда должна быть большой

**1.Основы JS**

**1.1 Типы данных**

Существует 8 типов данных см.таблицу



Типы данных можно разделить на 2 большие категории, простые типы (примитивы) и комплексные (объекты)

* + 1. Числа

Числа могут быть как целыми так и дробными 4 или 4.6 (ставиться точка а не запятая)

Infinity – числовой тип данных который обозначает бесконечность пример 4/0 – это бесконечность

NaN – числовой тип данных ( not a number «не число») пример console.log(‘string’ \* 9); это абсолютно не математическая операция, мы пытаемся набор букв умножить на число

* + 1. Строка

Обязательным атрибутом является кавычки, любая строка пишется в них пример const persone = “Anton”; , строки могут содержать огромное количество какого либо текста, строки могут содержать в себе и цифры “5” но это будет все равно строка !!!

В JS абсолютно нет ни какой разницы какие кавычки будем использовать, главное их не комбинировать, обычно вопрос кавычек решается стилистикой проекта или компанией

`` бэктики еще одни ковычки,

* + 1. Boolean либо да true либо нет false
    2. Null когда чего не существует
    3. Undefined когда что то существует но у его нет ни какого значения пример let und; console.log(und); есть холодильник а в нем не чего нет, он как бы существует но внутри пустой
    4. Symbol на практике он практически не используется
    5. Bigint
    6. Object это коллекция данных, структура которая используется для хранения абсолютно любых данных, такой тип данных не зря называется комплексным, ведь в нем могут содержаться все эти типы данных выше, для того чтобы создать объекты мы будем пользоваться синтаксисом фигурных скобок

Const obj = {

name: “Anton”,

age: 25,

isMarried: false

};

Если мы хотим добраться до свойства объекта, мы можем сделать это 2 способами, через:

Console.log(obj.name); через точку

Console.log(obj[name]); и получаем ошибку !!!

А вот почему Console.log(obj[“name”]); кавычки

Объект - это ключевая единица в JS в них будет храниться очень много информации, есть частный случай объектов его разновидность это массив [], Является ли массив отдельным типом данных? Нет он является частным случаем объекта

**Массив** - используется для хранения данных которые идут строго по порядку, каждый элемент массива имеет свой порядковый номер и можно поместить абсолютно любой тип данных

Let arr = [‘plum.png’, 6, {}, []];

Мы можем обратиться к каждому элементу массива по порядку, например меня интересует 2 элемент console.log(arr[1]);

**В программирование нумерация начинается не с 1 а с 0 !!!!!!!!!!!**

Let arr = [‘plum.png’, 6, {}, []]; Пример

**0 1 2 3**

И если я хочу получить 2 элемент тоесть «6» console.log(arr[1]);

**Разница между объектами и массивами**

**(Дополнительно)**

Массив – это перечень информации по порядку, у каждой сущности (информации которая записана в массиве) есть номер по порядку и порядок имеет важную роль, по этим номерам мы можем обращаться к этим сущностям

Объект – это структура хранения данных в парном формате ( ключ и значение) пример a: 1; a это ключ 1 это значение

В массиве и объекте есть свои методы и свойства

Добавление в объект новые данные

Пример

Arr.b = ‘1234’;

Console.log(arr[‘b’]; в скобках ключ должен быть

Можно создать с квадратными скобками

Arr[‘b’] = ‘1234’;

Квадратные скобки использовать приоритетнее !!!!!!!!!!

**Простое общение с пользователем**

Команда alert(‘Hello world’); используется чтоб пользователя о чем то важном предупредить, на сайтах можно увидеть когда закрываем сайт, приходит сообщение вы действительно хотите покинуть сайт

Typeof оператор при помощи которого можно проверить на тип данных

Вся информация которая приходит от пользователя приходит в виде строк !!!!!!!!!!!!!!!! запомнить

* 1. **Операторы в JS**

Унарный + называется + который ставиться перед определенным аргументом пример

Console.log(4 + +’5’); перед строкой 5 и делает 5 не строкой а числом

Инкремент и декремент они существуют для укорочения нашего кода и увеличивает или уменьшает значение на единицу пример

Let incr = 10,

Decr = 10;

Incr++;

Decr--:

Если мы поставим их впереди форма записи называется префиксной а после посфиксной

Посфиксная форма возвращает сначало старое значение а потом уже увеличивает или уменьшает а префиксной то сразу новое значение !!!!!!!!

% возвращает остаток от деления 2 чисел

Console.log(5%2); результат будет 1 т.к остаток (разделили 5 на 2 на сколько это возможно и остался остаток 1) и именно этот остаток нам возвращается

== сравнить (булиновое значение)

=== строгое равенство по типу данных сравнивает по числу строке и тд

&& и работает только тогда когда

|| или

! отрицание

**Вложенность циклов**

Перенос строки пример

Result += ‘+\n’;

**Функции, стрелочные функции**

**Смотреть в блокноте “ Функции”**



*Алгоритм кода для себя*

Функция

Условие проверяем (если оно выполянется идем дальше)

Используем цикл (для повторения действий однотипных)

И в цыкле можно писать условии которые нам будут делать конечный результат

Так же не забывать про переменную в функции в которую мы будем присваивать результат, а потом return что вернуть во внешний код для дальнейшего использования

**Методы и св-ва строк и чисел**

**Объекты диструктаризация объектов**

***См. текстовый файл объекты диструктаризация объектов***

***Массивы и псевдлмассивы***

*См. текстовый файл массивы и псевдомассивы*

***Передача по ссылке или по значению, Spread оператор***

*См. текстовый файл*