Javascript. Уровень 1

Урок 1

Основы языка Javascript

Общее знакомство с JavaScript, создание первого кода и его запуск.

План курса

- Урок 1. Основы языка JavaScript.
- Урок 2. Основные операторы Javascript.
- Урок 3. Циклы, массивы, структуры данных.
- Урок 4. Объекты в JavaScript.
- Урок 5. Введение в DOM.
- Урок 6. Обработка событий в JavaScript.
- Урок 7. Практикум.
- Урок 8. Анонимные функции. Замыкания.

План урока

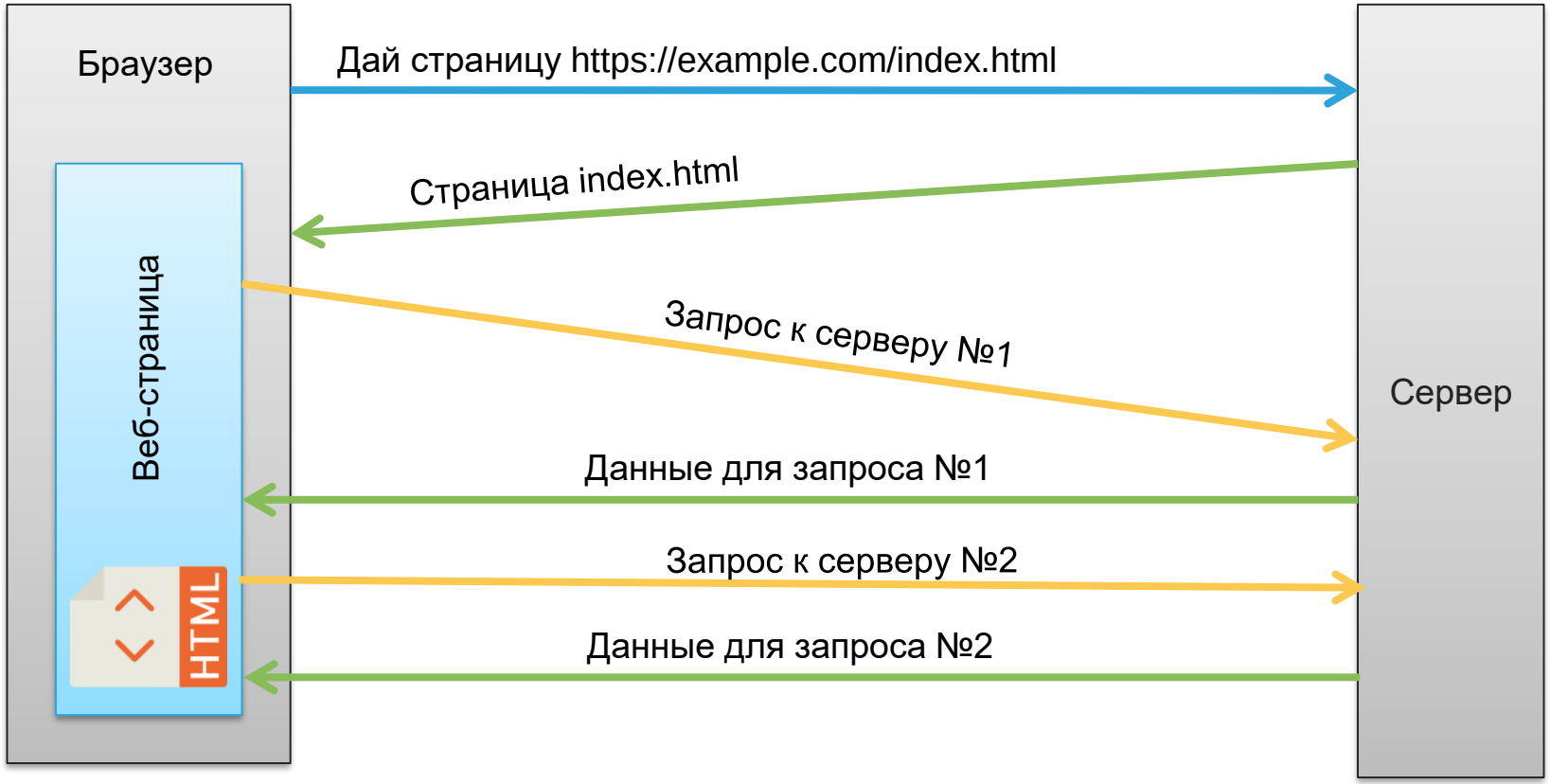
- Способы исполнения программ
- Как исполняется JavaScript-программа в браузере
- Принципы написания кода на JavaScript
- Структура кода
- Типы данных

Где используется Javascript

- На веб-страницах в браузере
- Для серверного программирования с помощью node.js
- Мобильные и настольные приложения
- Где угодно, если есть интерпретатор или компилятор Javascript кода

Для чего нужен Javascript на веб-странице

Схема работы AJAX на странице



Исполнение программы. Компиляция в машинный код.

```go
package main

import "fmt"

func main() {
    fmt.Println("Hello, world")
}
```

Исходный код(здесь язык Go), который пишет человек.

Процесс компиляции

Исполняемы файл, который может выполнить процессор компьютера.

Компилируемые языки

- C
- C++
- C#
- Go
- Fortran
- Delphi
- …

Исполнение программы. Компиляция в байт-код.

```java
class Hello {
    public static void main(String[] args) {
        System.out.println("Hello Java!");
    }
}
```

Исходный код(здесь язык Java).

Процесс компиляции в байт-код

Исполняемы файл, который может выполнить процессор виртуальной машины.

Виртуальная машина превращает байт-код в машинный код, который выполняет физический процессор компьютера.

Языки компилируемые в байт-код

- Java
- C#
- Python
- Erlang
- …

Исполнение программы. Интерпретация.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        console.log("Hello, world!");
    </script>
</body>
</html>
```

Код на Javascript, встроенный в html страницу.

Интерпретатор читает код. Компилирует на лету, скомпилированная версия выполняется процессором. Интерпретация происходит каждый раз при открытии файла.

V8 – это движок (интерпретатор), который встроен в браузер Chrome.

Интерпретируемые языки

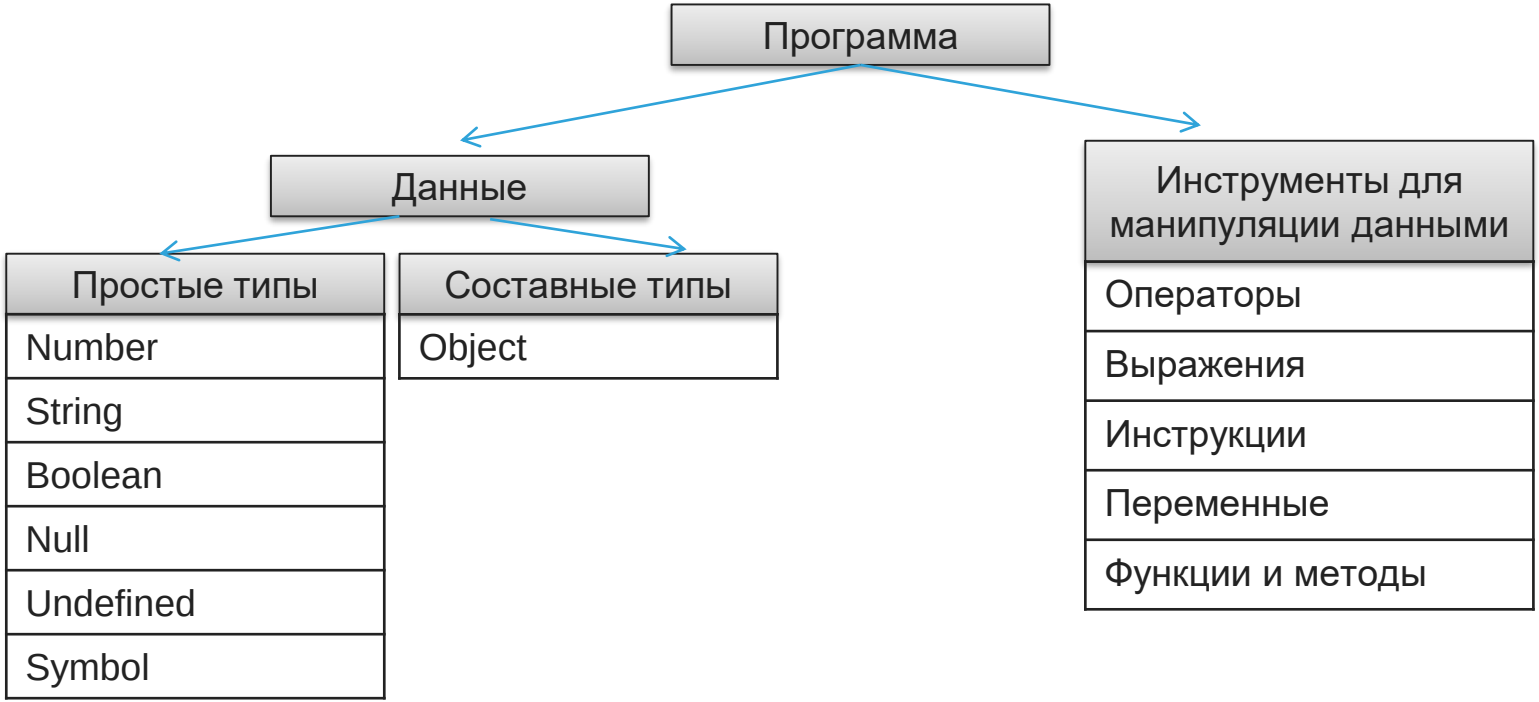
- Javascript
- PHP
- Python
- …

Схема работы программы

Примеры обработки данных

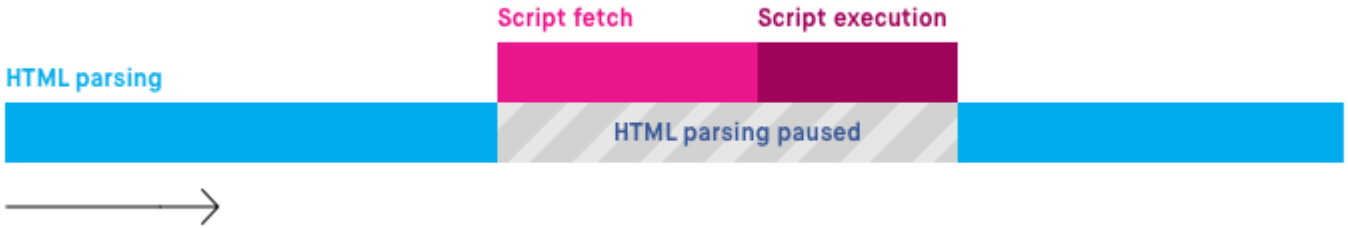
Вид программы	Входящие данные	Исходящие данные
3d игра	Нажатия на клавиши, перемещение мыши	Выстрелы из оружия, перемещение персонажа
Веб-страница	Заполненная форма регистрации на сайте	Письмо на почте пользователя о подтверждении email-адреса
Сервис онлайн-трансляций youtube	Видеопоток на сервер youtube	Видеопоток у пользователей на веб-странице в браузере

Из чего состоит программа на Javascript



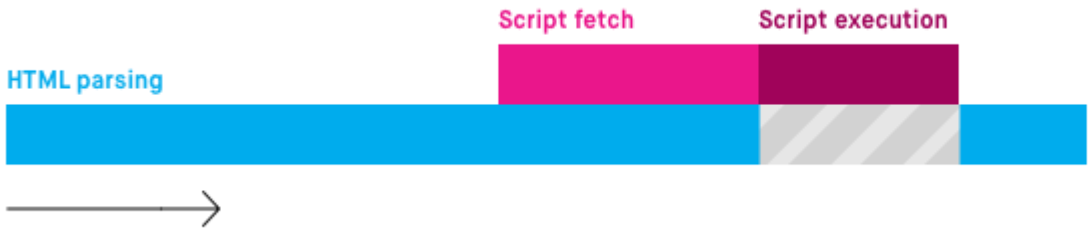
Обычный порядок выполнения скрипта

```
<!DOCTYPE html>
<html>
    <head> ... </head>
    <body>
        ...
        <script src="script.js"></script>
        ....
    </body>
</html>
```



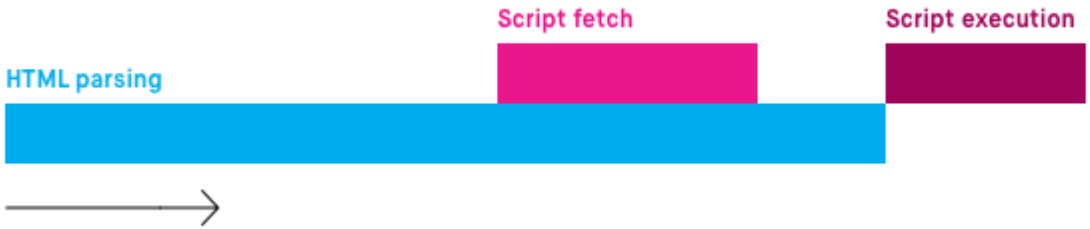
Порядок выполнения скрипта с атрибутом async

```
<!DOCTYPE html>
<html>
    <head> ... </head>
    <body>
        ...
        <script async src="script.js"></script>
        ....
    </body>
</html>
```



Порядок выполнения скрипта с атрибутом defer

```
<!DOCTYPE html>
<html>
    <head> ... </head>
    <body>
        ...
        <script defer src="script.js"></script>
        ....
    </body>
</html>
```



Функция – блок кода, содержащий в себе некоторую логику и действия.

Функции предназначены для многократного использования.

```
alert("hello, world!");

let greetings = "Всем привет!";
alert(greetings);
```

Строка – последовательность символов, используемых для представления текста.

```
alert(2 + 3);
alert("hello" + " " + "world");
```

Выражение – это фраза языка, которую может вычислить интерпретатор для получения значения.

Литерал – значение, указанное непосредственно в тексте программы.

```
//строковый литерал
alert("hello");

//числовой литерал
alert(10);

//логический(boolean) литерал
alert(true);
```

Комментарии в коде.

Переменные

			size (25)	

```
let size = 25;
```

Объявление переменных с помощью var и let

```
let myVar = "hello";
var num = 45;
```

Предпочитайте let.

Имена переменных

Неправильно

```
let 10years = 10;
let @myMail = "hello@example.com";
let всемПривет = "как дела?";
let function = "do something";
```

Корректно

```
let years10 = 10;
let $myMail = "hello@example.com";
let hi_all = "как дела?";
let _doSome = "do something";
```

Константы – не изменяемые значения

Операторы

- Арифметические
- Присваивание
- Конкатенация
- Побитовые
- Доступ к значениям
- Сравнения
- Логические

Приоритеты операторов

Приоритет	Тип оператора	Ассоциативность	Конкретные операторы
20	Группировка	не определено	(…)
19	Вызов функции	слева направо	… (…)
17	Постфиксный инкремент	не определено	… ++
16	Логическое отрицание	справа налево	! …
	Унарный плюс		+ …
	typeof		typeof …
14	Умножение	слева направо	… * …
	Деление		… / …
13	Сложение	слева направо	… + …
	Вычитание		… - …
11	Меньше	слева направо	… < …
	Меньше или равно		… <= …
	Больше		… > …
	Больше или равно		… >= …
10	Равно	слева направо	… == …
	Не равно		… != …
	Строго равно		… === …
	Строго не равно		… !== …
6	Логическое «И»	слева направо	… && …
5	Логическое «ИЛИ»	слева направо	… || …
3	Присваивание	справа налево	… = …

Ассоциативность – определяет в каком порядке будут вычисляться части выражения.

Тип string

```
let size = "middle";
let greetings = 'hello';
```

Тип number

```
let integerValue = 125;
let floatingPoint = 10.367;
let negative = -34;
```

Тип number

```
let val1 = 10 / 0;        //Infinity
let val2 = 3 * "hello"; //NaN
```

Тип boolean

```
let money = 100;
let price = 20;

//100 > 20 = true
if (money > price) {
    console.log("покупаем");
} else {
    console.log("отказываемя");
}
```

Тип null

```
let message = null;
```

null – это явное указание на отсутствие значения.

Тип undefined

```
let a;
console.log(a); //undefined
```

Переменная имеет значение undefined, если ей не присвоили значение. Явно не пишется.

Отличие null от undefined

null – используется для явного присваивания, чтобы указать что в переменной ничего нет.
undefined – это значение переменной, которую создали, но ничего не присвоили. В явном виде присваивать undefined не надо.

Оператор typeof

Типизация

В Javascript	В другом языке (для сравнения)

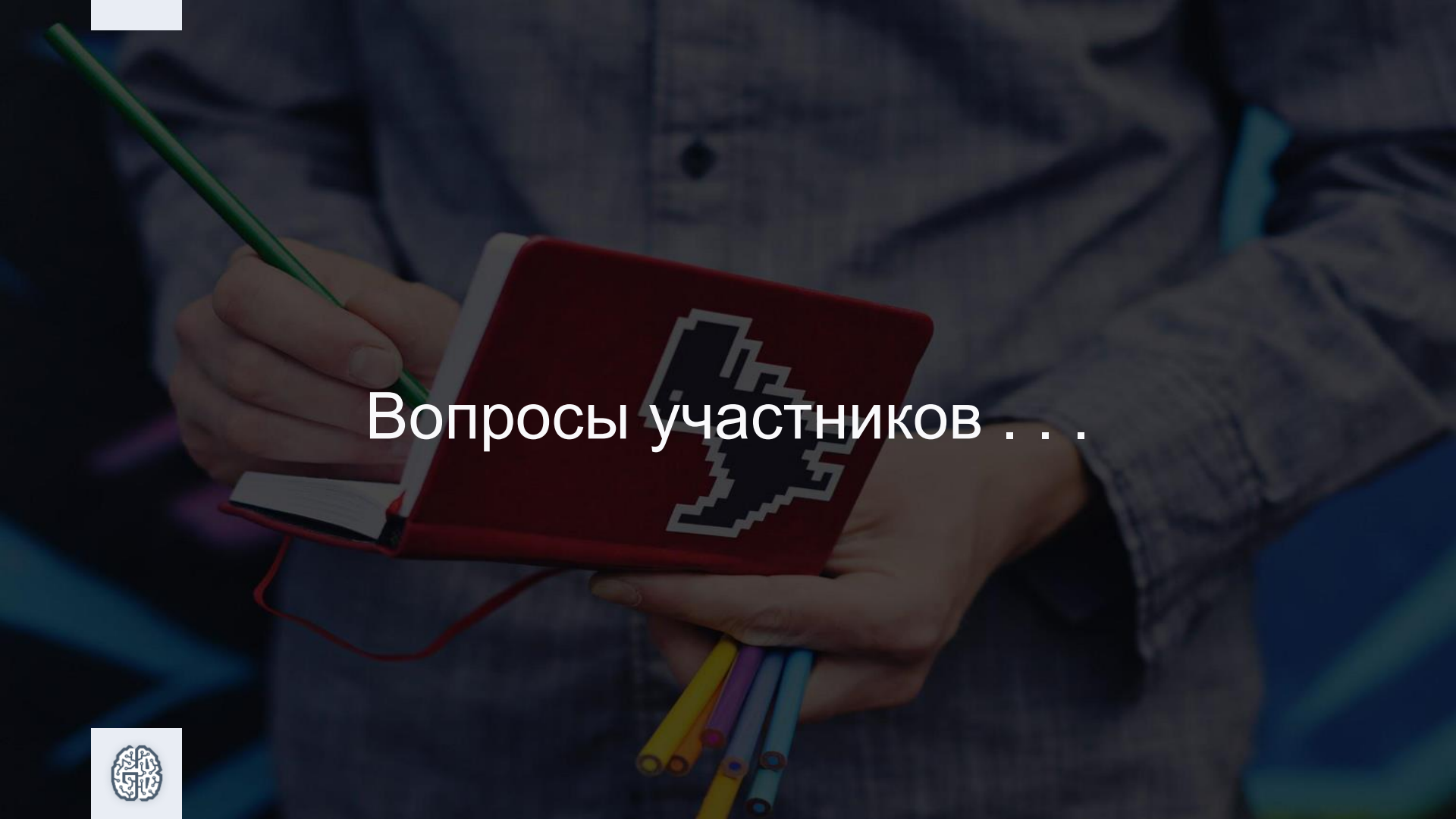
`let a = 44; //типизация не явная, т.к. тип переменной не указан явно`	`int x = 10; //тип указан явно`
	Здесь язык Java

`let value = prompt("введите значение");`
`value = "i'am string";`
`value = 50;`
Код корректно отработает. Тип может меняться на лету, когда приходят данные. Типизация динамическая.
	`value := 10`
	`value = "hi"`
	Ошибка: cannot use "hi" (type string) as type int in assignment.
	Типизация статическая, мы не можем поменять тип переменной.
	Здесь язык Go.

`console.log(3 == "3");` Этот код выведет true. Произойдет приведение к одному типу, а затем сравнение. Типизация слабая (не строгая). Значения разных типов можно смешивать.	`fmt.Println(3 == "3")` Ошибка: prog.go:6:16: cannot convert "3" (type untyped string) to type int prog.go:6:16: invalid operation: 3 == "3" (mismatched types int and string) Типизация сильная(строгая). Язык Go. Значения разных типов смешивать нельзя.

Домашнее задание

Актуальное домашнее задание находится в материалах урока в отдельном pdf-файле.


Вопросы участников . . .