

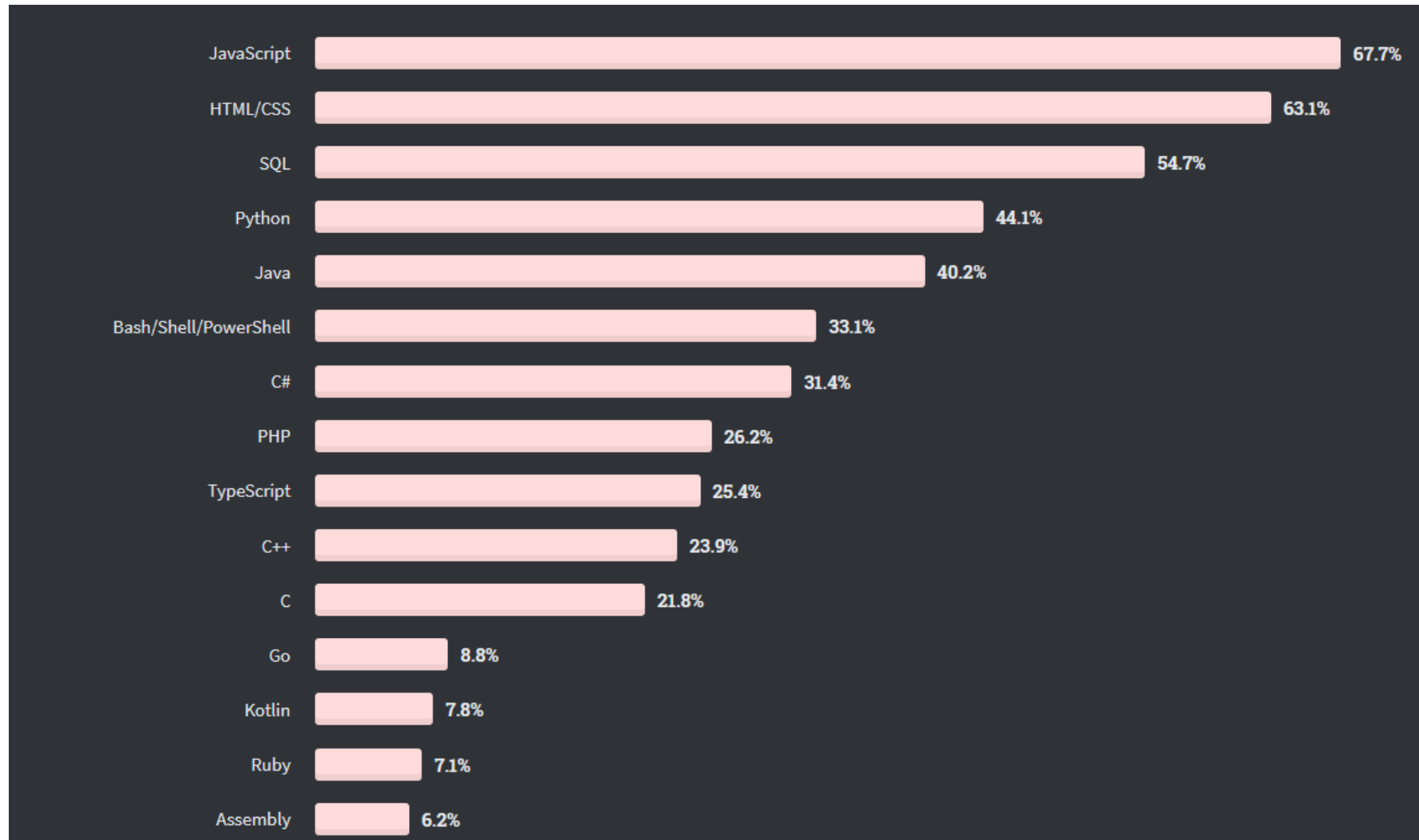
Разработка кросс-платформенных приложений

Лекция 1

Course Outline

- JavaScript
- React / Svelte*
- React Native
- D3.js

Почему JavaScript?



Что такое JavaScript?

- LiveScript, JavaScript, EcmaScript

JavaScript is **high-level**, often **just-in-time compiled**, and **multi-paradigm**. It has **curly-bracket syntax**, **dynamic typing**, **prototype-based object-orientation**, and **first-class functions**.

Прототипное наследование

```
let foo = {name: "foo", one: 1, two: 2};
```

```
let bar = {two: "two", three: 3};
```

```
bar.__proto__ = foo;
```

```
bar.one;
```

```
bar.three ;
```

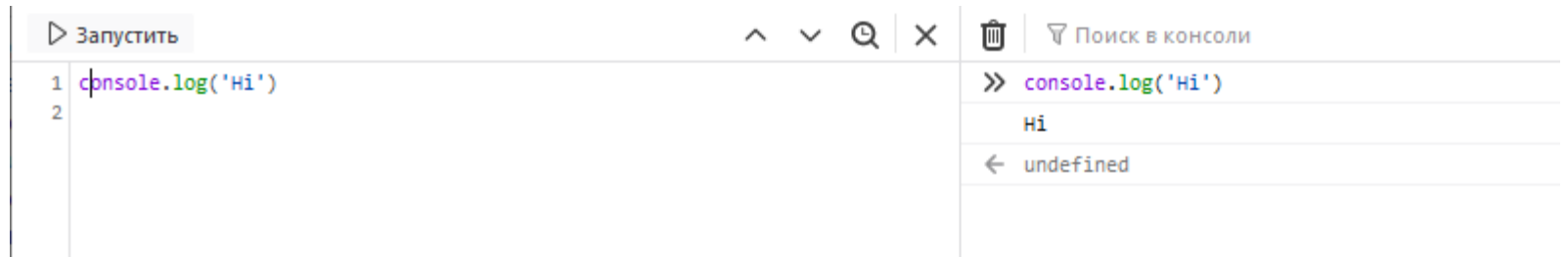
```
bar.two;
```

Альтернативы?

- JavaScript
- TypeScript
- CoffeeScript
- Dart
- Elm

Где запускать JS?

- Браузеры
- Web-движки (V8, WebView...)
- Платформы (Node.js)



```
▶ Запустить ^ v 🔍 × 🗑️ 🔍 Поиск в консоли
1 console.log('Hi')
2
>> console.log('Hi')
Hi
← undefined
```

Где писать JS?

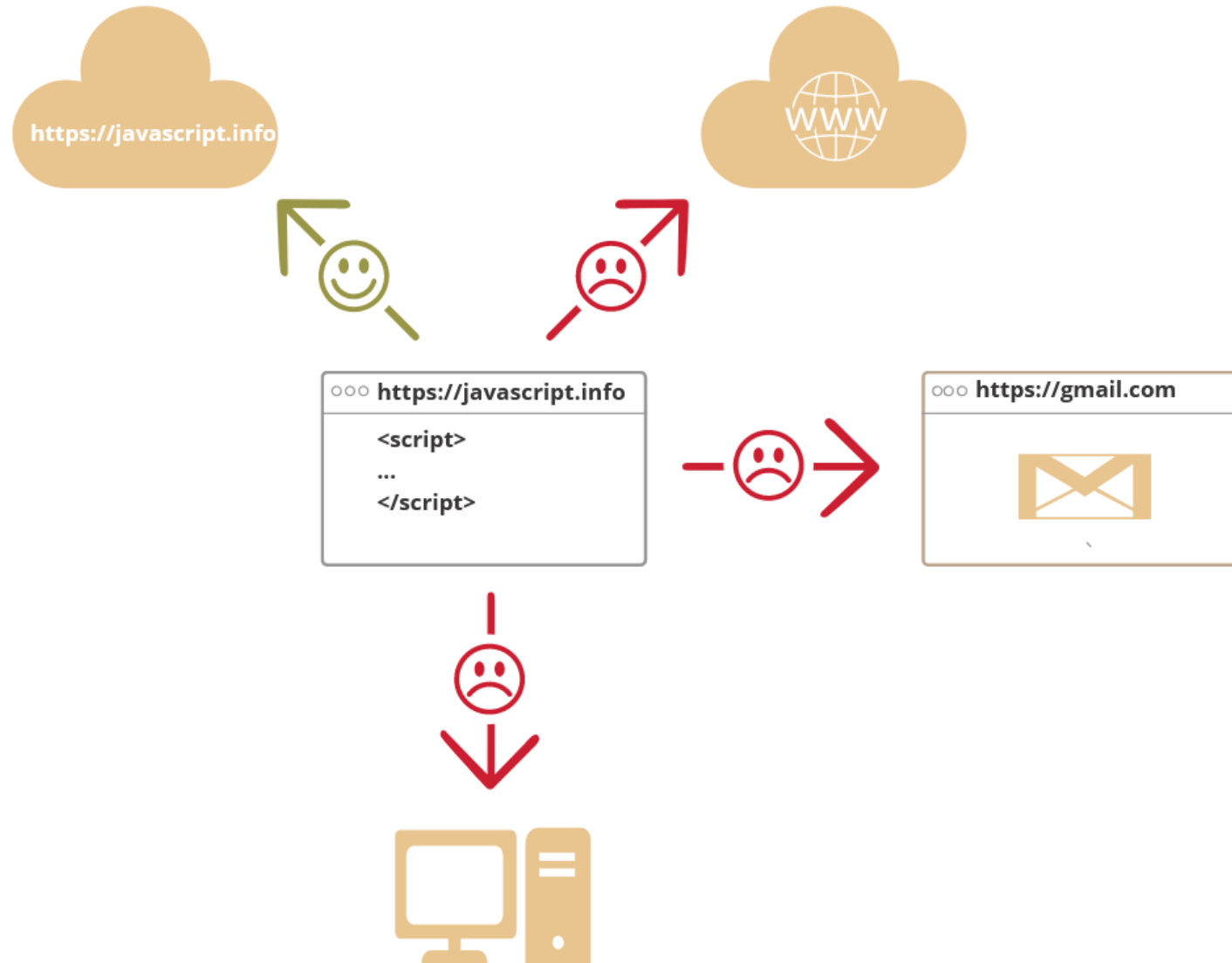
- WebStorm (Any JetBrains IDE)
- VS Code
- Any Editor

- Github Account

Что может делать JS в браузере?

- Добавлять, удалять и редактировать HTML и стили, изменять содержимое страницы
- Реагировать на действия пользователя.
- Отправлять запросы по сети
- Устанавливать сессии и куки, показывать диалоговые окна.
- Запоминать данные на клиенте

Чего не может?



Hello, World!

```
<html>
```

```
<body>
```

```
<p>Before the script...</p>
```

```
<script>
```

```
  alert( 'Hello, world!' );
```

```
</script>
```

```
<p>...After the script.</p>
```

```
</body>
```

```
</html>
```

Не весь код должен быть в HTML

```
<script src="/js/script1.js"></script>
```

```
<script src="https://cdn.com/ajax/libs/somelib.js"></script>
```

```
<script src="file.js">  
  alert(1);  
</script>
```

Структура кода

- Semicolons are reason for holy wars
- Parsers are smart, but not all-knowing

Comments

//Single line comment

*/*Multiline
comment */*

Переменные

- 1.The name must contain only letters, digits, or the symbols \$ and _.
- 2.The first character must not be a digit.

```
let message;  
message = 'Hello!';  
alert(message);
```

```
let id = 1; //Rec!
```

```
let user = 'Admin', age = 20; //Not Rec!
```

```
var oldVar = 'not so good'; //Not Rec!
```

Константы

```
const importantDate = '01.01.2021';
```

```
importantDate = '01.01.2022'; // error
```

```
const COLOR_RED = "#F00";
```


var vs let vs const

- Scopes
 - Redeclaration
 - Hoisting (Raising) for declarations (not assignments!)
-
- Что писать – **const**
 - Если очень надо – **let**
 - **var** обычно не нужно

Типы данных

- Number

```
let n = 123;    Infinity, NaN  
n = 12.345;
```

- String

```
let singleQuotedString = "String";  
let doubleQuotedString = 'I am a string too';  
let interpolatedString = `can embed another ${singleQuotedString}`;
```

Data Types

- Boolean

`true false`

- null, undefined

- object

- symbol

Wild dynamic typing

```
let a = 1;  
typeof a;  
typeof(a);
```

```
typeof undefined // "undefined"
```

```
typeof 0 // "number"
```

```
typeof 10n // "bigint"
```

```
typeof true // "boolean"
```

```
typeof "foo" // "string"
```

```
typeof Symbol("id") // "symbol"
```

```
typeof Math // "object" (1)
```

```
typeof null // "object" (2)
```

```
typeof alert // "function" (3)
```

Type Conversion

- Explicit

```
let x = 4;  
let s = String(x);  
alert(typeof(s));
```

- Implicit

```
alert( "4" / "2" );  
alert('2'*4);  
alert(true+'IDK'+7);
```

```
alert(2 + 2 + '1' );
```

Numeric Conversion

Value	Becomes...
<code>undefined</code>	<code>NaN</code>
<code>null</code>	<code>0</code>
<code>true</code> and <code>false</code>	<code>1</code> and <code>0</code>
<code>string</code>	Whitespaces from the start and end are removed. If the remaining string is empty, the result is <code>0</code> . Otherwise, the number is "read" from the string. An error gives <code>NaN</code> .

Математика

Сложение	+
Вычитание	-
Умножение	*
Деление	/
Остаток	%
Степень	**
In place	*=, /=, -=, +=
Инкремент	++

Сравнения

```
let a = 0;  
alert( Boolean(a) ); // false
```

```
let b = "0";  
alert( Boolean(b) ); // true
```

```
alert(a == b); // true!
```

== // неявное преобразование
=== // строго

Почему, зачем...

```
alert( null > 0 ); // false  
alert( null == 0 ); // false  
alert( null >= 0 ); // true
```

Ветвление

```
let condition = true;
```

```
if (condition) alert('ok');
```

```
if (condition){  
    alert('ok')  
}
```

```
if (condition){  
    alert('ok')
```

```
} else //may have some else if branches
```

```
{
```

```
    alert('not ok');
```

```
}
```

```
let output = condition ? 'ok' : 'not ok';
```

Логические операции

- Or `||` (short-circuit)
- And `&&`
- Not `!`
- First defined value `??`

ЦИКЛЫ

```
while (condition) {  
  // loop body  
}
```

```
do {  
  // loop body  
} while (condition);
```

```
for (let i = 0; i < 3; i++) {  
  alert(i);  
}
```

break
continue

```
outer: for (let i = 0; i < 3; i++) {  
  for (let j = 0; j < 3; j++) {  
    let input = prompt(`Value at coords (${i},${j})`, "");  
    if (!input) break outer;  
  }  
}  
alert('Done!');
```

Switch (строгое сравнение!)

```
switch(x) {  
  case 'value1':  
    ...  
    [break]  
  case 'value2':  
    ...  
    [break]  
  default:  
    ...  
    [break]  
}
```

```
switch (a) {  
  case 4:  
    alert('Right!');  
  case 3:  
  case 5:  
    alert('Left');  
    break;  
  default:  
    alert('The result is strange. Really.');
```

ФУНКЦИИ I

```
function showMessage() {  
    alert( 'Hello everyone!' );  
}
```

```
function showMessage(message, sender= 'Unknown') {  
    alert( `From ${sender}: ${message}` );  
}
```

```
function showMessage(message, sender) {  
    alert( `From ${sender ?? 'Unknown'}: ${message}` );  
}
```

Функции II

```
function sum(a, b) {  
  return a + b;  
}
```

Можно сделать **return** без значения
Функции без **return** вернут ***undefined***

Функции – граждане первого класса

Функциональные выражения

```
let sayHi = function() {  
    alert( "Hello" );  
};  
alert( sayHi );  
alert( sayHi() );
```


Callbacks

```
function check(condition, onSuccess, onFailure)
{
  if (condition)
  {
    onSuccess();
  }
  else
  {
    onFailure();
  }
}
```

Стрелочные Функции

```
let sum = (a, b) => a + b;
```

```
let sayHi = () => alert("Hello!");
```

```
let sum = (a, b) => {  
    let result = a + b;  
    return result;  
};
```

References

- <https://javascript.info/>
- <https://developer.mozilla.org/>