JavaScript

Ашрафи Арифа
Мохначев Виктор Сергеевич

Что такое JavaScript? // What is JavaScript?

JavaScript — это скриптовый язык программирования, который позволяет нам реализовывать сложные функции на веб-страницах. // JavaScript is a scripting or programming language that allows us to implement complex features on web pages.

JavaScript — это текстовый язык программирования, используемый как на стороне клиента, так и на стороне сервера, который позволяет делать веб-страницы интерактивными. // JavaScript is a text-based programming language used both on the client-side and server-side that allows us to make web pages interactive.

Он позволяет нам создавать динамически обновляемый контент, управлять мультимедиа, анимировать изображения и так далее. // It enables us to create dynamically updating content, control multimedia, animate images, and so on.

Что можно делать на JavaScript? // What we can do with JavaScript?

Основной клиентский язык JavaScript позволяет нам делать такие вещи, как:

- Хранить полезные значения внутри переменных.
- Выполнять операции над фрагментами текста (известными в программировании как «строки»).
- Запускать код в ответ на определенные события, происходящие на веб-странице.

The core client-side JavaScript language allow us to do things like:

- Store useful values inside variables.
- Operations on pieces of text (known as "strings" in programming).
- Running code in response to certain events occurring on a web page.

Что можно делать на JavaScript? // What we can do with JavaScript?

Web/Mobile Apps

Веб и мобильные приложения

Real-Time Networking Apps

Сетевые приложения в реальном времени

Command-line Tools

Инструменты командной строки

Games

Игры

Где запускается JavaScript-код? // Where does JavaScript code run?

JavaScript работает в каждом веб-браузере “из коробки”. Приложение, написанное на JavaScript, работает на любом устройстве, тогда как настольное или мобильное приложение работает только на той платформе, для которой оно предназначено (Windows, Mac OS X, Linux, iPhone, Android). Это позволяет нам очень легко писать кроссплатформенные приложения.

Мы можем запустить код в браузере, создав файл HTML, в котором есть ссылка на скрипт. В этом случае мы использовали возможность отложенного запуска, которая позволяет запустить JS после завершения загрузки файла HTML.

JavaScript runs in every web browser, out of the box. A JavaScript application runs on every device, whereas a desktop or mobile application runs only on the platform it is targeted to (Windows, Mac OS X, Linux, iPhone, Android). This allows us to write cross-platform apps in a really easy way.

We can run code in the browser by creating an HTML file that references the script. In this case, we used the defer option, which will execute the JS after the HTML file is finished loading.

Как связаны JavaScript и ECMAScript? // How are JavaScript and ECMAScript related?

Спецификация ECMAScript — это стандартизированная спецификация скриптовых языков, разработанная Бренданом Эйхом из Netscape; первоначально назывался Mocha, затем LiveScript и, наконец, JavaScript.

Проще говоря, JavaScript — это скриптовый язык общего назначения, соответствующий спецификации ECMAScript. Спецификация ECMAScript — это план создания скриптового языка. JavaScript является реализацией этого плана. В целом JavaScript реализует спецификацию ECMAScript, как описано в ECMA-262.

The ECMAScript specification is a standardized specification of a scripting language developed by Brendan Eich of Netscape; initially named Mocha, then LiveScript, and finally JavaScript.

Simply speaking, JavaScript is a general-purpose scripting language that conforms to the ECMAScript specification. The ECMAScript specification is a blueprint for creating a scripting language. JavaScript is an implementation of that blueprint. On the whole, JavaScript implements the ECMAScript specification as described in ECMA-262.

Вывод JavaScript // JavaScript Output

JavaScript может “отображать” данные разными способами // JavaScript can "display" data in different ways:
JavaScript can "display" data in different ways:

- Запись в элемент HTML с использованием innerHTML // Writing into an HTML element, using innerHTML.
- Запись в вывод HTML с помощью document.write(). // Writing into the HTML output using document.write().

```
Использование document.write() после загрузки HTML-документа удалит весь существующий HTML. Метод document.write() следует использовать только для тестирования. // Using document.write() after an HTML document is loaded, will delete all existing HTML. The document.write() method should only be used for testing.
```

- Запись в окно предупреждения с помощью window.alert(). // Writing into an alert box, using window.alert().
- Запись в консоль браузера с помощью console.log(). // Writing into the browser console, using console.log().

Примеры JavaScript // JavaScript Example

```
<!DOCTYPE html>
<html>
<body>

<h2>Javascript Example 1</h2>
<p>Using innerHTML.</p>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = 221 + 328;
</script>

</body>
</html>
```

Примеры JavaScript // JavaScript Example

```
<!DOCTYPE html>
<html>
<body>

<h2>Javascript Example 2</h2>
<p>Using document.write()</p>

<p>Never call document.write after the document has finished loading.</p>
<p>It will overwrite the whole document.</p>

<script>
document.write(221 + 328);
</script>

</body>
</html>
```

Примеры JavaScript // JavaScript Example

```
<!DOCTYPE html>
<html>
<body>

<h2>Javascript Example 3</h2>
<p>document.write().</p>

<button type="button"
onclick="document.write(221 + 328)">Try it</button>

</body>
</html>
```

Примеры JavaScript // JavaScript Example

```
<!DOCTYPE html>
<html>
<body>

<h2>Javascript Example 4</h2>
<p>window.alert.</p>

<script>
window.alert(221 + 328);
</script>

</body>
</html>
```

Примеры JavaScript // JavaScript Example

console.log():

В целях отладки мы можем вызвать метод console.log() в браузере для отображения данных. // For debugging purposes, we can call the console.log() method in the browser to display data.

Print:

В JavaScript нет объекта print или методов print(). Мы не можем получить доступ к устройствам вывода из JavaScript. Единственным исключением является то, что мы можем вызвать метод window.print() в браузере для печати содержимого текущего окна. // JavaScript does not have any print object or print methods. We cannot access output devices from JavaScript. The only exception is that we can call the window.print() method in the browser to print the content of the current window.

Операторы JavaScript // JavaScript Statements

На языке программирования компьютерная программа представляет собой список «инструкций», которые должны быть «выполнены» компьютером. Эти инструкции по программированию называются операторами. // In a programming language, a computer program is a list of "instructions" to be "executed" by a computer. These programming instructions are called statements.

Программа JavaScript представляет собой набор операторов. // A JavaScript program is a list of programming statements.

Операторы JavaScript состоят из: значений, операций, выражений, ключевых слов и комментариев. // JavaScript statements are composed of: Values, Operators, Expressions, Keywords, and Comments.

The statements are executed, one by one, in the same order as they are written. // Операторы выполняются один за другим в том же порядке, в котором они написаны.

Важные замечания об операторах JavaScript // Important Notes about JavaScript Statements

1. Операторы JavaScript разделяются точками с запятой (‘;’). // Semicolons separate JavaScript statements (‘;’).
2. При разделении точкой с запятой допускается несколько операторов в одной строке. // When separated by semicolons, multiple statements on one line are allowed.
3. JavaScript игнорирует множественные пробелы. // JavaScript ignores multiple spaces.
4. Если оператор JavaScript не помещается на одной строке, лучше всего его разбить после символа операции. // If a JavaScript statement does not fit on one line, the best place to break it is after an operator.
5. Операторы JavaScript могут быть сгруппированы в блоки кода внутри фигурных скобок {...}. // JavaScript statements can be grouped together in code blocks, inside curly brackets {...}.
6. Ключевые слова JavaScript являются зарезервированными словами. Зарезервированные слова нельзя использовать в качестве имен переменных. // JavaScript keywords are reserved words. Reserved words cannot be used as names for variables.

Примеры JavaScript // JavaScript Example

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Statements</h2>

<p>Multiple statements on one line are allowed.</p>

<p id="demo1"></p>

<script>
let a, b, c;
a = 221; b = 328; c = a + b;
document.getElementById("demo1").innerHTML = c;
</script>

</body>
</html>
```

Значения в JavaScript // JavaScript Values

Синтаксис JavaScript определяет два типа значений // The JavaScript syntax defines two types of values:

- Фиксированные значения // Fixed values
- Переменные значения // Variable values

Фиксированные значения называются Литералами. // Fixed values are called Literals.
Переменные значения называются Переменными. // Variable values are called Variables.

Литералы в JavaScript // JavaScript Literals

Литералы в JavaScript — это постоянные значения, которые могут быть присвоены переменным, называемым литералами или константами. // JavaScript Literals are constant values that can be assigned to the variables that are called literals or constants.

Литералы JavaScript — это синтаксические представления для различных типов данных, таких как числовые, строковые, логические значения, массивы и т. д. // JavaScript Literals are syntactic representations for different types of data like numeric, string, Boolean, array, etc data.

Литералы в JavaScript предоставляют средства для представления определенных значений в нашей программе. // Literals in JavaScript provide a means of representing particular or some specific values in our program.

Рассмотрим пример: `var name = "arifa"`, объявляется строковая переменная с именем name, которой присваивается строковое значение «arifa». Литерал «arifa» представляет собой значение для переменной name. // Consider an example, `var name = "arifa"`, a string variable named name is declared and assigned a string value "arifa". The literal "arifa" represents, the value for the variable name.

Типы литералов в JavaScript // JavaScript Literal Types

- Массивы // Array literals.
- Логические // Boolean literals.
- Числовые // Numeric literals.
- Объекты // Object literals.
- Регулярные выражения // RegExp literals.
- Строки // String literals.

Переменные в JavaScript // JavaScript Variables

Переменные - это контейнеры для хранения данных (значений). // Variables are containers for storing data (storing data values).

В JavaScript существует 4 способа объявления переменной // 4 Ways to Declare a JavaScript Variable:

- Используя `var` // Using `var`
- Используя `let` // Using `let`
- Используя `const` // Using `const`
- Используя `nothing` // Using `nothing`

Переменные в JavaScript // JavaScript Variables

Чтобы объявить переменную в JavaScript, нужно использовать ключевое слово `let`. // To create a variable in JavaScript, it is necessary to use the `let` keyword.

Оператор ниже создаёт (другими словами, объявляет) переменную с именем “message”. // The statement below creates (in other words: declares) a variable with the name “message”:

```
let message;
```

Теперь мы можем поместить в неё некоторые данные используя оператор присваивания = // Now, we can put some data into it by using the assignment operator =:

```
let message;
message = 'Hello'; // store the string 'Hello' in the variable named
message
```

Переменные в JavaScript // JavaScript Variables

Когда использовать ключевое слово `var`? // When to Use JavaScript `var`?

- Необходимо всегда объявлять переменные JavaScript с помощью `var`, `let` или `const`. // It is necessary to always declare the JavaScript variables with `var`, `let`, or `const`.
- Ключевое слово `var` используется во всем коде JavaScript с 1995 по 2015 год. // The `var` keyword is used in all JavaScript code from 1995 to 2015.
- Ключевые слова `let` и `const` были добавлены в JavaScript в 2015 году. // The `let` and `const` keywords were added to JavaScript in 2015.
- Если мы хотим, чтобы наш код работал в старых браузерах, мы должны использовать `var`. // If we want our code to run in older browsers, we must use `var`.

Когда использовать ключевое слово `const`? // When to Use JavaScript `const`?

- Общее правило: всегда объявлять переменные при помощи `const`. // If we want a general rule: always necessary to declare the variables with `const`.
- Если мы думаем, что значение переменной может измениться, тогда нужно использовать `let`. // If we think the value of the variable can change, then we should use `let`.

Темы следующего занятия // Topics for Next Class

- Идентификаторы // JavaScript Identifiers
- Типы данных // JavaScript Data Types
- Объявление переменных // Declaring JavaScript Variables
- Арифметика // JavaScript Arithmetic
- Функции и т.д. // JavaScript Functions, etc.