Лабораторная работа 11. Основы программирования на JavaScript

Цель работы. Познакомиться с JavaScript, с его помощью изменить вёрстку на странице.
Результатом работы должен быть пройденный тренажер на htmlacademy.ru

Вспомогательные материалы
https://htmlacademy.ru/courses/343

Конспект

JavaScript: что это такое и как подключить его на страницу

Язык программирования JavaScript придумали специально для того, чтобы создавать интерактивные сайты.

Код на языке JavaScript называют скриптом. Его сохраняют в отдельный файл с расширением js, а чтобы запустить, подключают этот файл на страницу. В HTML для добавления JavaScript есть специальный тег:

```
<script src="адрес_файла"></script>
```

Подключают скрипт обычно в самом конце страницы, перед закрывающим тегом </body>.

Программа на JavaScript — это последовательность инструкций, то есть указаний браузеру выполнить какие-то действия. Инструкции выполняются последовательно, сверху вниз.

Чтобы сказать JavaScript, что инструкция закончена, нужно поставить точку с запятой или перейти на новую строку. Новая строка правильно работает в большинстве случаев, а точка с запятой — **всегда**. Поэтому лучше ставить точку с запятой в конце каждой инструкции.

JavaScript не меняет исходный файл с разметкой, но, выполняя инструкции, меняет страницу прямо в браузере пользователя.

Комментарии

Комментарий — это текст, поясняющий код. Он не выводится в браузер и никак не влияет на работу программы. Инструкции внутри комментария не выполняются, поэтому комментарии часто используют, если нужно временно отключить часть кода.

В JavaScript есть два вида комментариев:

```
// Однострочные комментарии.
/*
И многострочные.
Они могут отключить сразу несколько строк кода.
*/
```

Консоль

Консоль — инструмент разработчика, который помогает тестировать код. Если во время выполнения скрипта возникнет ошибка, в консоли появится сообщение о ней. А ещё в консоль можно выводить текстовые подсказки. Чтобы вывести сообщение в консоль, нужно использовать console.log:

```
console.log('Привет от JavaScript!');
// Выведет: Привет от JavaScript!

console.log(document.querySelector('.page'));
// Выведет в консоль найденный элемент
```

Типы данных

С разными типами данных можно производить разные действия, поэтому программисту важно знать, с чем он работает. В нашей консоли тип данных выводится в скобках, например (String) или (Number).

Существуют простые и сложные типы данных. Простые:

- number — числа: целые и с точкой;
- string — строки;
- boolean — логические, или булевы, значения: true — «истина» и false — «ложь»;
- undefined — «не определено», англ.

Строки нужно оборачивать в кавычки: одинарные или двойные.

Сложные, или составные, типы содержат не одно, а несколько значений. Массив, array, хранит последовательность значений, и порядок этих значений важен. Объект, object, состоит из множества пар «ключ-значение», порядок этих пар не важен.

// Массив

[1, 2, 3, 4, 5]

// Объект

{month: 'june', day: 15}

Переменные

Переменная — просто название для данных, которое можно делать понятным для людей. Переменные упрощают работу с памятью: они «приклеиваются» к ячейкам памяти, как наклейка с названием приклеивается к папке с документами.

В JavaScript переменные можно создавать командой let, за которой следует имя переменной:

```
let имяПеременной;
```

Имя переменной можно записать по-разному. Два самых популярных способа: camelCase (верблюжья нотация) и snake_case (змеиная нотация). В первом случае все слова пишутся слитно и каждое слово, за исключением первого,

начинается с большой буквы (myNumber, userName). Во втором случае все слова разделяются нижним подчёркиванием (my_number, my_name).

Имена переменных в JavaScript чувствительны к регистру: myname и myName — две разные переменные. Имя переменной может содержать буквы, цифры и знак подчёркивания, но оно не должно начинаться с цифры. Кроме того, в качестве имени переменной нельзя использовать ключевые слова, такие как let или if. Вот полный список этих ключевых слов.

После создания переменной её можно использовать в других командах, например, выводить в консоль:

```
// Обратите внимание, что кавычек нет!
console.log(имяПеременной);
```

Если обратиться к пустой переменной, то получим undefined — «не определено». Чтобы записать в переменную данные, ей их нужно присвоить. Для операции присваивания используется знак равенства:

```
let timeInHours;                // Объявляем переменную
console.log(timeInHours);       // Выведет: undefined
timeInHours = 2;                // Присваиваем одно значение
console.log(timeInHours);       // Выведет: 2
timeInHours = 'три часа';       // Присваиваем совершенно другое значение
console.log(timeInHours);       // Выведет: три часа
```

Команда let для создания каждой переменной используется всего один раз. Дальше мы обращаемся к переменной по её имени, без let. Если повторно задать значение переменной, то значение этой переменной изменится. Предыдущее значение при этом исчезнет. Это называется переопределением переменной.

Операции и операторы

Команды состоят из операций. 5 + 10; — это операция. Она состоит из оператора, +, и двух операндов, 5 и 10.

Оператор указывает, что произойдёт с операндами. Операции бывают унарными, бинарными и тернарными, в зависимости от количества операндов. Бинарные операции самые распространённые.

Над разными типами операндов можно производить разные операции, поэтому важно понимать, данные какого типа хранятся в переменных.

Порядок выполнения операций зависит от их приоритета. Если у операций одинаковый приоритет, они выполняются слева направо. Приоритет различных операторов можно посмотреть здесь.

Арифметические операции

Арифметические операции в JavaScript выполняются так же, как в математике: сначала умножение, потом сложение. Изменить порядок операций можно с помощью круглых скобок. Снова как в математике: выражение в скобках посчитается в первую очередь.

Сложение +

Вычитание -

Умножение *

Деление /

Конкатенация

Самая частая строковая операция — это «склеивание» строк, или конкатенация:

```
let name = 'Кекс';

'Инструктор' + 'Кекс';  // Результат: 'ИнструкторКекс'
'Инструктор ' + 'Кекс'; // Результат: 'Инструктор Кекс'
'Инструктор ' + name;   // Результат: 'Инструктор Кекс'
```

Конкатенация позволяет делать сообщения программ более информативными и «человечными».

Приведение типов

Что будет, если использовать операнды разного типа?

```
'Время, мин: ' + 50; // Результат: 'Время, мин: 50'
'2' * 50;            // Результат: 100
```

JavaScript попытается привести операнды к одному типу и выполнить операцию. Подходящий тип будет выбираться в зависимости от операции.

Плюс может быть знаком сложения или конкатенации, но так как один из операндов — строка, то сложение не подходит. Поэтому число 50 приводится к строке '50' и склеивается со строкой 'Время, мин: '.

Звёздочка — это знак умножения, со строками она не используется. Поэтому JavaScript пытается превратить строку '2' в число, и ему это удаётся. Затем числа 2 и 50 перемножаются, и получается 100.

Из-за того, что JavaScript умеет изменять тип операндов на лету, он называется языком со *слабой типизацией*.

Метод querySelector

Чтобы найти на странице элемент, нужно использовать метод querySelector, он ищет по селектору:

```
document.querySelector('селектор');
```

Эта инструкция состоит из двух частей. Первая часть — элемент, внутри которого будет искать JavaScript. Словом document обозначается веб-страница, к которой скрипт подключён. Неважно, как называется файл на самом деле, в JavaScript это всегда «документ». Он является элементом-родителем для любого другого элемента на странице.

Вторая часть инструкции — это то, что нужно сделать. Её называют методом.

Методы для изменения классов

Чтобы убрать у элемента класс, нужно использовать метод **`classList.remove`**. Он убирает с элемента тот класс, который указан в скобках:

```
элемент.classList.remove('класс');
```

Чтобы добавить элементу класс, нужно использовать метод **`classList.add`**:

```
элемент.classList.add('класс');
```

Метод-переключатель classList.toggle убирает у элемента указанный класс, если он есть, и добавляет, если этого класса нет:

```
элемент.classList.toggle('класс');
```

Свойство textContent

У каждого элемента имеется множество свойств: его размеры, цвет и так далее. Свойство textContent хранит в себе текстовое содержимое элемента. Свойствам можно присваивать новые значения:

```
let paragraph = document.querySelector('p');
paragraph.textContent = 'Здесь был Кекс. Мяу!';
```

Свойство value

У полей ввода есть особое свойство — value. Оно хранит данные, введённые в поле. Мы можем вывести их прямо на страницу:

```
let input = document.querySelector('input');
paragraph.textContent = input.value;
```

Конкатенация

Операция, когда мы «склеиваем» несколько значений, называется конкатенацией и в JavaScript выполняется с помощью знака плюс.

```
let name = 'Кекс';
paragraph.textContent = 'Вас зовут ' + name + '. Хорошего дня!';
console.log(paragraph.textContent);
// Выведет: Вас зовут Кекс. Хорошего дня!
```

Обработчики событий onclick и onsubmit

JavaScript следит за всем, что происходит на странице. Клик по кнопке или отправка формы — это событие. Мы можем сказать JavaScript, что сделать, когда некое событие произойдёт. Для этого используют обработчики событий. Инструкции, которые должны будут выполниться, когда событие произойдёт, располагают между фигурных скобок.

Свойство onclick означает «по клику»:

```
let button = document.querySelector('button');
button.onclick = function() {
```

```
  console.log('Кнопка нажата!');
};
```

При каждом клике по кнопке в консоли будет появляться новое сообщение Кнопка нажата!.

За обработку отправки формы отвечает свойство onsubmit:

```
let form = document.querySelector('form');
form.onsubmit = function() {
  console.log('Форма отправлена!');
};
```

После отправки формы в консоли появится сообщение `Форма отправлена!`.

Задание. Первая программа: KeksoFit v0.1

Выполнить упражнения в тренажере https://htmlacademy.ru/courses/343

И составить программу, разместив ее в одной из «песочниц»: codepen.io, jsfiddle.com, repl.it

Ваша задача — написать фитнес-калькулятор для Кекса. Вот техническое задание, написанное от его лапы:

«Мяу! Я решил вести здоровый образ жизни и больше двигаться. Для этого мне нужен свой особый фитнес-калькулятор.

Я ввожу, сколько времени в соцсетях я провёл накануне, а он строит программу тренировок и показывает сообщение-подсказку. Тренировка состоит из прыжков на шкаф. А сообщение выглядит так: «За <число> минут в соцсетях полагается <число> прыжков.»

Вот алгоритм работы калькулятора:

```
Время в соцсетях хранится в переменной timeInMinutes. Вводить ее можно командой
 prompt(title, [default]);
За каждые 10 минут, которые Кекс провёл в соцсетях, он получает 3 прыжка.
Прыжки храним в переменной jumps.
Сообщение склеиваем из трёх строк: 'За ', ' минут в соцсетях полагается ', ' прыжков.' – и двух переменных: timeInMinutes и jumps.
Например: 'За 50 минут в соцсетях полагается 15 прыжков.'
Сообщение записываем в переменную message.
```

Значения переменных желательно выводить в консоль.

Предположим, в соцсетях проведено 300 минут.

```
// Считаем, сколько отрезков времени по 10 минут
300 / 10 = 30;
// За каждые 10 минут добавляется 3 прыжка
30 * 3 = 90
// Склеиваем сообщение
'За ' + 300 + ' минут в соцсетях полагается ' + 90 + ' прыжков.'
```

Полезные ссылки и материалы

1. https://www.w3schools.com/js/default.asp
2. Основы JavaScript