

# JS. Работа с переменными, условный оператор



1

# ОСНОВНОЙ БЛОК

# Что такое JavaScript

JavaScript (JS) - это мультипарадигмальный язык программирования, который широко используется в веб-разработке для создания интерактивных пользовательских интерфейсов, динамических веб-страниц и многих других веб-приложений.

JavaScript выполняется на стороне клиента (т.е. в браузере) и на стороне сервера (с использованием платформы Node.js).

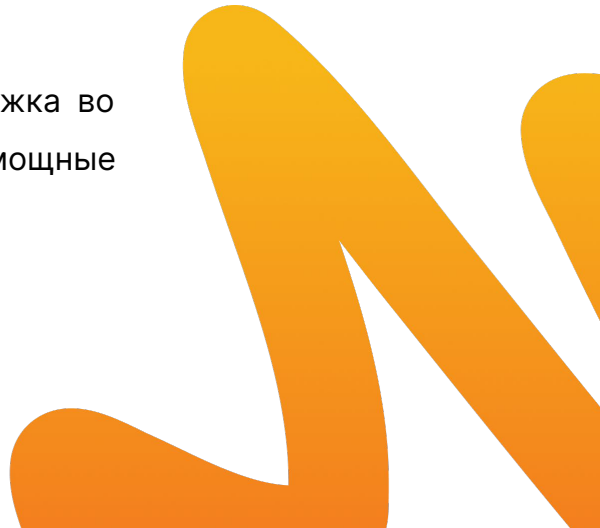


# Что такое JavaScript

JavaScript используется для добавления динамического поведения веб-страницам, таким как обработка событий, анимация, проверка форм, валидация ввода и многие другие функции.

Он также может использоваться для создания веб-приложений с помощью различных фреймворков и библиотек, таких как Angular, React и Vue.js.

Одним из главных преимуществ JavaScript является его широкая поддержка во всех современных веб-браузерах, что позволяет разработчикам создавать мощные и эффективные веб-приложения для любой аудитории.



# Переменные

Ключевые слова в JavaScript, которые используются для объявления переменных:

`let`

`const`

`var`

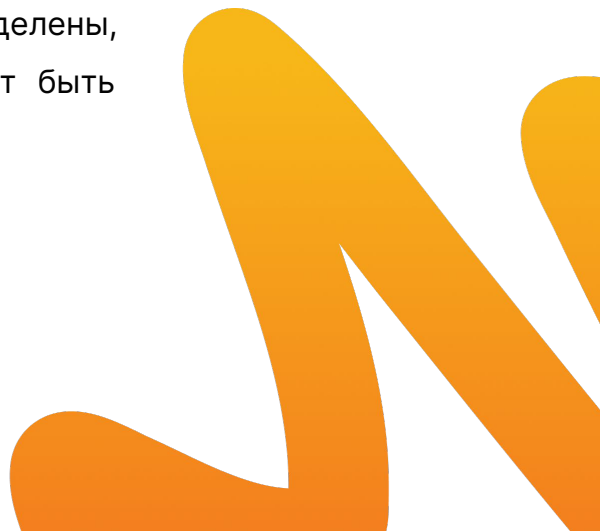


# var

- var было первым ключевым словом, используемым для объявления переменных в JavaScript.
- Недостаток var заключается в том, что он не имеет блочной области видимости, что может привести к нежелательным эффектам при использовании вложенных блоков кода.
- Сейчас var считается устаревшим и не используется при написании кода.

# let и const

- let и const были добавлены в JavaScript в ECMAScript 6 (ES6) для решения проблемы области видимости переменных.
- Они также имеют блочную область видимости, что означает, что переменные, объявленные с помощью let и const, доступны только в блоке кода, в котором они были объявлены, и в любых вложенных блоках кода.
- Переменные, объявленные с помощью "let", могут быть переопределены, тогда как переменные, объявленные с помощью "const", не могут быть переопределены.



# Правильное именование переменных

Хорошее именование переменных является важной практикой в программировании. Хорошо именованные переменные делают код более читаемым и понятным, что в свою очередь облегчает его сопровождение и отладку.





# Правильное именование переменных

Используйте осмысленные имена переменных, которые описывают их назначение или значение. Например, "age", "username", "totalAmount".

Используйте английский язык для именования переменных, если это возможно, чтобы обеспечить совместимость и понимание кода для других разработчиков.

Используйте camelCase или snake\_case для составных слов.

Избегайте использования ключевых слов языка программирования (например, "if", "for", "while" и т.д.) в качестве имен переменных.

Начинайте имена переменных с буквы. Нельзя начинать их с цифры или специального символа.


Используйте ясные и понятные имена переменных, даже если это может привести к более длинным именам.

Старайтесь использовать существительные в качестве имен переменных, чтобы обозначать объекты или значения.

# Типы данных `string` и `number`

"String" и "number" - это два основных типа данных в JavaScript.


"String" (строка) - это последовательность символов, заключенных в кавычки (одинарные или двойные). Он может содержать любые символы, включая буквы, цифры, специальные символы и пробелы. Например:



```
1 let greeting = "Hello, World!";  
2 let message = 'Welcome to our website.';
```

# Типы данных `string` и `number`

"Number" (число) - это значение, представляющее числовое значение. Оно может быть целым или дробным, положительным или отрицательным. Например:



```
1 let age = 30;  
2 let price = 9.99;
```

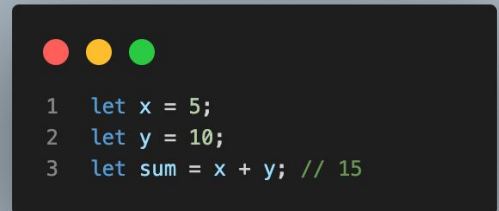
# Типы данных `string` и `number`

Операции, которые можно выполнять с `"string"` и `"number"`, могут различаться. Например, если сложить две строки с помощью оператора `+` (конкатенации), они будут объединены в одну строку:



```
1 let firstName = "John";  
2 let lastName = "Doe";  
3 let fullName = firstName + " " + lastName; // "John Doe"
```

Если сложить два числа, они будут сложены вместе:



```
1 let x = 5;  
2 let y = 10;  
3 let sum = x + y; // 15
```




# Конкатенация и интерполяция

"Конкатенация" и "интерполяция" - это два способа объединения строк в JavaScript.

"Конкатенация" - это процесс объединения двух или более строк в одну строку с использованием оператора "+" (плюс).

Например:



```
1 let firstName = "John";  
2 let lastName = "Doe";  
3 let fullName = firstName + " " + lastName; // "John Doe"
```

# Конкатенация и интерполяция

"Интерполяция" - это способ вставки значений переменных внутри строки, используя специальный синтаксис "\${ }".

Например:



```
1  et firstName = "John";
2  let lastName = "Doe";
3  let fullName = `${firstName} ${lastName}`; // "John Doe"
```

# Интерполяция

- Использование интерполяции может сделать код более читаемым и понятным, особенно если вам нужно вставлять много значений переменных внутри строки.
- Он также облегчает изменение значения переменной, так как вы можете просто изменить значение переменной, а не переписывать всю строку.
- Важно помнить, что при использовании интерполяции вы должны заключать строку в обратные кавычки (`), а не в одинарные или двойные кавычки.



2

# ВОПРОСЫ ПО ОСНОВНОМУ БЛОКУ





TEL-RAN  
by Starta Institute

# 3

## ЗАДАНИЕ ДЛЯ ЗАКРЕПЛЕНИЯ

#### ЗАДАНИЕ

Есть переменные `r`, `g`, `b` с числовыми значениями.  
Вывести в консоль строку `rgb(12, 34, 14)` используя  
конкатенацию и интерполяцию.

# Неявное преобразование типов данных

- Иногда JavaScript может автоматически преобразовывать "number" в "string" и наоборот.

Например, если вы используете оператор "+" для объединения "string" и "number", JavaScript автоматически преобразует "number" в "string".

- В случае преобразования "string" в "number", JavaScript пытается преобразовать строку в число, если это возможно.

Если строка не может быть преобразована в число, результатом будет значение "NaN" (Not a Number).





TEL-RAN  
by Starta Institute

# 4

## ЗАДАНИЕ ДЛЯ ЗАКРЕПЛЕНИЯ

## ЗАДАНИЕ

1. Предположите, что получится в результате 'b' + 'a' + '+c'

## ЗАДАНИЕ

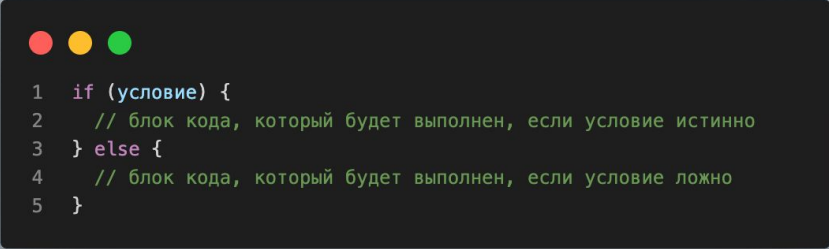
1. Предположите, что получится в результате 'b' + 'a' + '+c'
2. Совместно с преподавателем:  
Написать программу, которая считывает через prompt число и выводит в консоль ее квадрат

## ЗАДАНИЕ

1. Предположите, что получится в результате 'b' + 'a' + '+c'
- Совместно с преподавателем:
2. Написать программу, которая считывает через prompt число и выводит в консоль ее квадрат
3. Написать программу, которая считывает два числа (объявляем две переменные и записываем туда результат работы двух вызовов prompt) и выводит их сумму. Не забыть преобразовать полученные значения в число.

# Условный оператор if-else

"if-else" - это условный оператор в JavaScript, который позволяет выполнить определенный блок кода, если определенное условие истинно, и выполнить другой блок кода, если условие ложно. Синтаксис оператора "if-else":

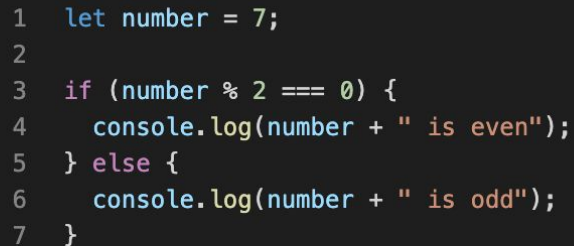


```
1  if (условие) {  
2    // блок кода, который будет выполнен, если условие истинно  
3  } else {  
4    // блок кода, который будет выполнен, если условие ложно  
5  }
```



# Условный оператор if-else

Пример использования оператора "if-else" для проверки, является ли число четным или нечетным:



```
1 let number = 7;  
2  
3 if (number % 2 === 0) {  
4   console.log(number + " is even");  
5 } else {  
6   console.log(number + " is odd");  
7 }
```

5

# ВОПРОСЫ ПО ОСНОВНОМУ БЛОКУ



TEL-RAN  
by Starta Institute

# 6

## ЗАДАНИЕ ДЛЯ ЗАКРЕПЛЕНИЯ

## ЗАДАНИЕ

1. Написать программу, которая получает два числа и выводит наибольшее.

## ЗАДАНИЕ

1. Написать программу, которая получает два числа и выводит наибольшее.
2. Написать программу, которая считывает через prompt одно число и выводит одну из строк “число положительное”, “число отрицательное”, “число равно нулю”.

# isNaN

`isNaN()` - это функция в JavaScript, которая позволяет проверить, является ли переданное значение "Not-a-Number" (NaN).

Функция возвращает логическое значение "true", если переданное значение является NaN, и "false", если значение может быть преобразовано в число.

Синтаксис функции `isNaN()`:



`value` - значение, которое необходимо проверить на NaN.



# isNaN

Примеры использования функции isNaN():



```
1  isNaN("Hello World") // true
2  isNaN(123) // false
3  isNaN(NaN) // true
4  isNaN("123") // false
5  isNaN("1e10000") // true
```



7

# ОСТАВШИЕСЯ ВОПРОСЫ



# Полезные ссылки

- [Что такое JavaScript? - Изучение веб-разработки | MDN](#)
- [Основы JavaScript](#)