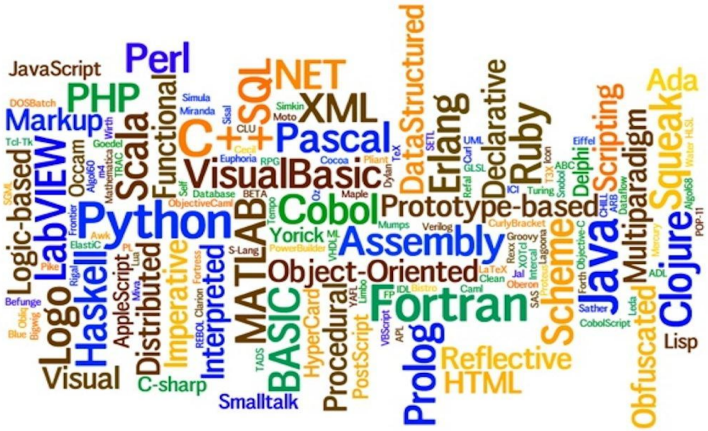
«Переменные, типы, операции»

JavaScript Courses

www.courses.dp.ua

JavaScript – язык программирования

JavaScript – язык программирования



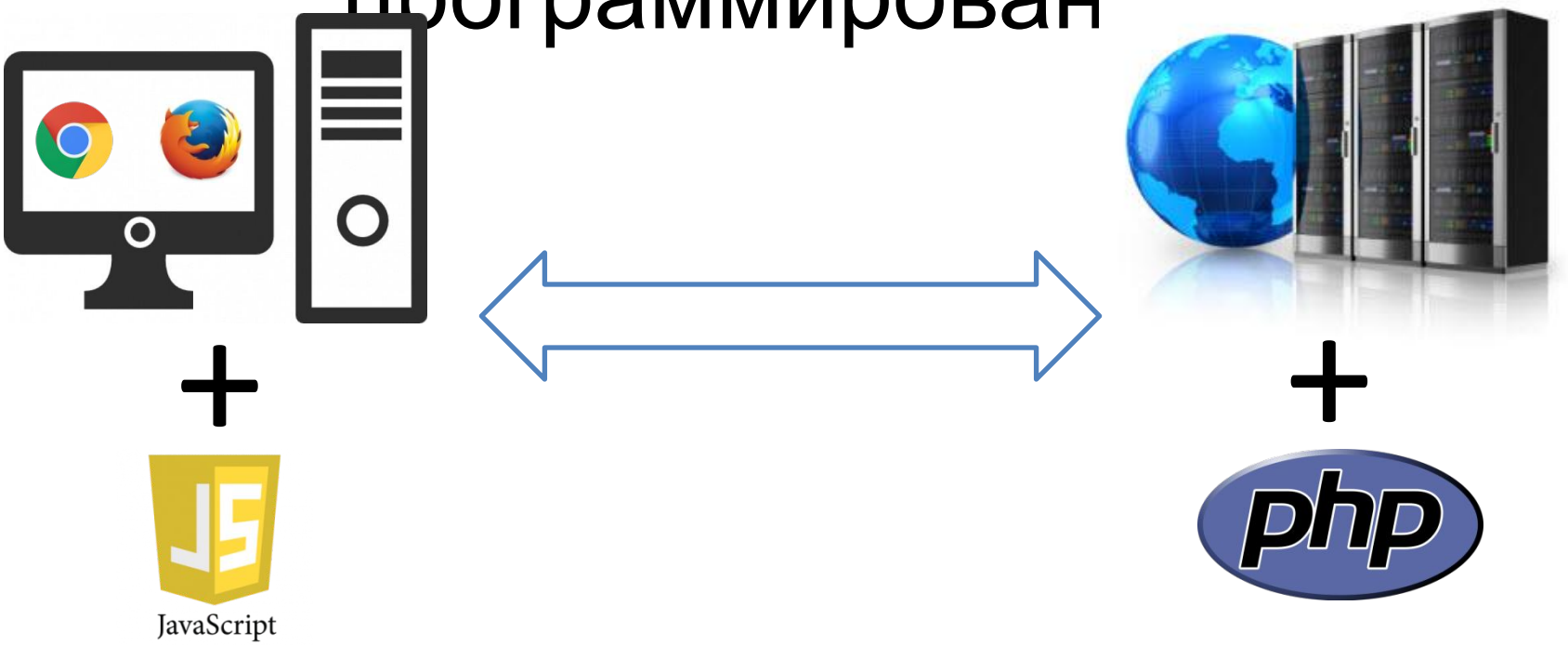
1. Компьютеры не понимают русский язык (пока), они понимают языки программирования;

2. Чтобы компьютер (и браузер как его часть) что-то сделал нужно ему сказать что нужно делать (описать последовательность действий) на языке программирования;

3. Как правило, задача любой программы заключается в манипулировании информацией (данными), например: текстом и картинками;

4. JavaScript тоже занимается манипуляцией данными (тегами и их содержимым). При помощи JS мы можем манипулировать HTML-документом: изменять теги, добавлять и удалять их.

JavaScript – язык программирован



JavaScript предназначен, чтобы уговорить компьютер что-то сделать ***на стороне пользователя*** *(на вашем компьютере, в вашем браузере), в отличии от других языков которые работают* ***на стороне сервера****.*

Основы программирования на базе JavaScript

JavaScript как язык программирования

его концепции

Переменные / Типы / Операции

Ветвления (условные операторы)

Циклы / Массивы (структуры данных)

Функции

И

Объекты

JavaScript

1. Интерпретируемый.

```
var new_paragraph           = document.createElement("p");
new_paragraph.innerHTML     = paragraph_text;
new_paragraph.ondblclick    = remove_paragraph;
new_paragraph.onclick       = set_class;
document.body.appendChild(new_paragraph);
```

2. Чувствительный к регистру.

~~GETELEMENTBYID();~~ ~~GetElementById();~~

getElementById(); ~~getelementbyid();~~

JavaScript варианты подключения

```
<script>
...
</script>
```

HTML5

```
<script type="text/javascript">
...
</script>
```

HTML < 5

```
<script type="text/javascript" src="./js_files/my_script.js"></script>
```

HTML < 5, внешний файл сценария.

*Тег **<script>** может присутствовать в любом месте документа. Но чаще всего его размещают в блоке **<head>**.*

Однако JavaScript код можно писать и в атрибутах тегов

```
<html>
<body>
    <p onclick = " this.style.color = 'red'; this.style.fontSize = '24pt';
    this.style.background='yellow'; "
    ondblclick = " this.style.color = 'black'; this.style.fontSize = 'medium';
    this.style.background='white';" >
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin lacus elit,
    commodo vitae dui in, consectetur vulputate nisl. Mauris lacinia enim non
    rhoncus tristique.
    </p>
</body>
</html>
```

Но это привод к «распылению» кода по странице.

«Допустимый» синтаксис

```
4  <script>
5
6      var message = "Text, text, text.";
7
8      console.log(message);
9
10 </script>
```

В процессе обучения мы можем ограничиваться только тегами <script></script> для написания кода, и опускать полную разметку документа.

Алгоритм

Алгоритмы

Задача: Написать скрипт, который рассчитает сколько гривен в день приносит депозит размещенный на полтора года под 22% годовых?

Проблемы:

✔*Дан недостаточный объём данных или часть данных задана неявно, нужно уточнять;*

✔*Часть данных избыточна (но отвлекает);*

✔*Есть сторонние факторы, не известные заранее, влияющие на результат.*

Алгоритмы

Задача: Написать скрипт, который рассчитает сколько гривен в день приносит депозит размещенный на полтора года под 22% годовых?

Алгоритм:

1. Уточняем сумму депозита.
2. Рассчитываем сколько будет дохода за целый год:

```
Доход = Сумма*(22%/100);
```

3. Считаем доход за 1 день:

```
Доход_день = Доход / Количество_дней_в_году;
```

4. Расчитываем налоги:

```
Сумма_налога = Доход_день*((18% + 1,5%)/100);
```

5. Учитываем налог:

```
Доход_день_после_налога = Доход_день – Сумма налога;
```

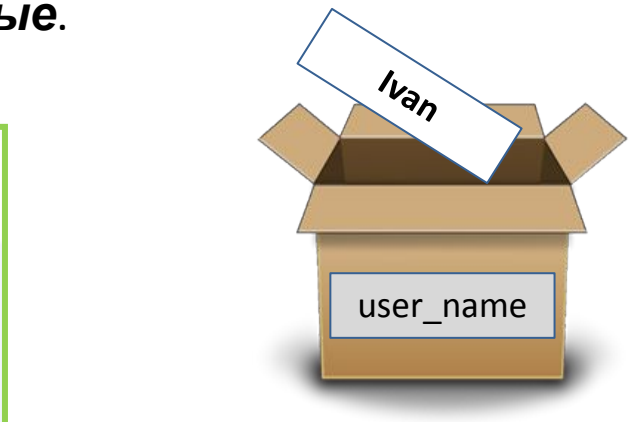
6. Выводим результаты.

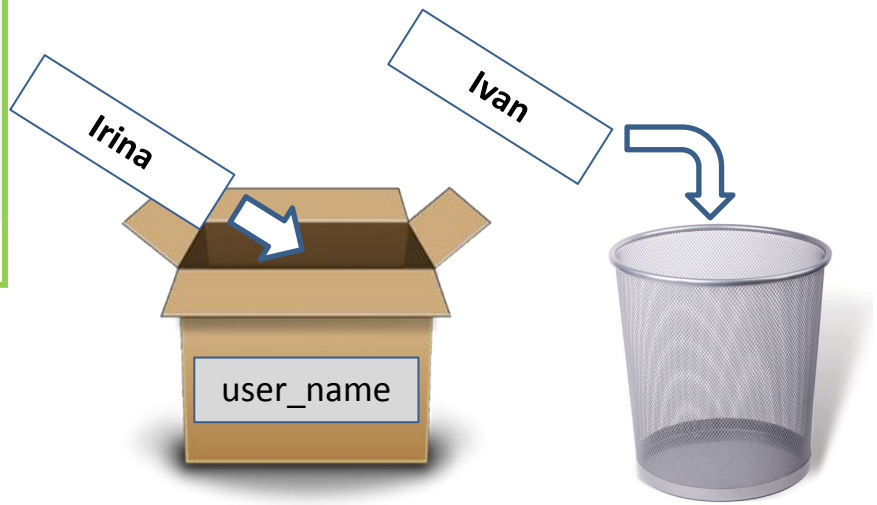
Переменные, типы, операции

Вне зависимости от того, для чего делается скрипт, понадобится работать с информацией

Для хранения информации, используются ***переменные***.

```
1  <script>
2
3      var user_name = "Ivan";
4
5      alert(user_name);
6
7      user_name = "Irina";
8
9      alert(user_name);
10
11 </script>
```





Переменные

Для хранения информации, используются ***переменные***.


```
var user_name = "Ivan";
```

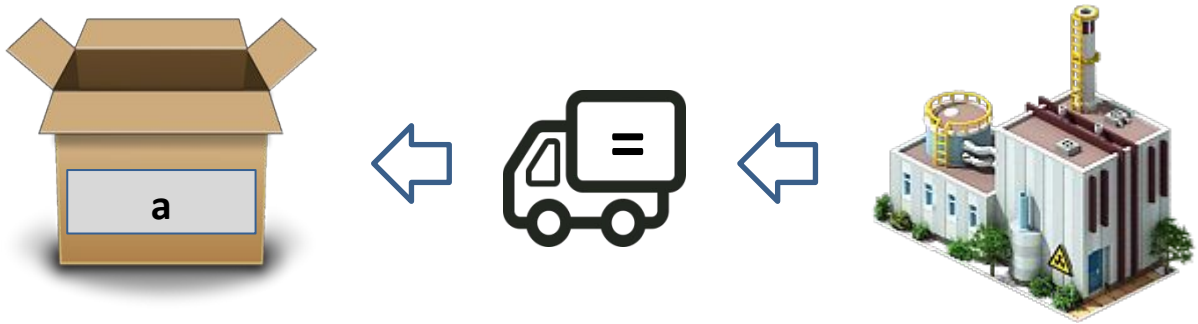
*Перед использованием переменной мы должны попросить выделить под неё место с памяти. Для этого используется ключевое слово **var**. С его помощью происходит т.н. определение переменной. Определение переменной нужно делать только одни раз. В дальнейшем можно использовать переменную по имени, без слова **var**.*

*В **ECMAScript-2015** добавилось ключевое слово **let**, основное отличие в области видимости переменной обвяленной с его помощью, и **const** - позволяющий объявлять константы.*

Оператор присвоения

Чтобы сказать компьютеру, что именно нужно записать в перемененную используется оператор присвоения =

```
a = 2 + 3 * 4;
```



Оператор присвоения берёт то что справа от него и записывает в переменную имя которой расположено слева от него.

Операторы, операнды и операции…

Для выполнения действий (операций) над переменными (или значениями) используются операторы, операторов существует много. С некоторыми из них все знакомы, например с арифметические операторами.

Унарный оператор – тот который взаимодействует только с одной переменной (операндом).

```
6       ++a;
7       b--;
```

Бинарный оператор – тот который взаимодействует с двумя переменными (операндами).

```
6       var c = (a + b) * 4 - (++b);
```

У операторов есть приоритеты, какой приоритет выше, какой ниже запомнить непросто. Поэтому в случае сомнений какая операция будет первой а какая второй – смело используйте скобки. Принцип их применения такой же как и в математике – скобки повышают приоритет операции в них записанной.

«Скобками программу не испортишь» (с)

```
var c = (a + b) * 4 - (++b);
```

Операторы, операнды и операции…

Унарный оператор – тот который взаимодействует только с одной переменной (операндом).

```
<script>

    var a = 5;
    var b = -a;

    ++a;
    b--;

</script>
```

```
<script>
    var x = 5;

    alert(++x);
    alert(x);
</script>
```

⇨ [6, 6]

```
<script>
    var x = 5;

    alert(x++);
    alert(x);
</script>
```

⇨ [5, 6]

Бинарный оператор – тот который взаимодействует с двумя переменными (операндами).

```
<script>

    var a = 5;
    var b = 6;

    var c = (a + b) * 4 - (++b);

</script>
```

Операторы, операнды и операции…

Что получится?

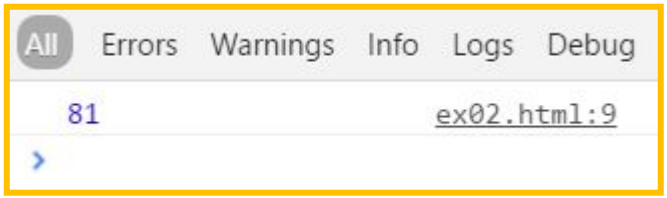
```
1  <script>
2
3      var x = 5;
4
5      var y = x++ + ++x;
6
7      alert(y);
8
9  </script>
```

?!?

Выражения

По правую сторону от оператора присвоения может быть как конкретное значение (5 или 9 или “Ivan”), а также может быть выражение – формула рассчитав которую компьютер получит результат который будет записан в переменную имя которой стоит слева от знака присвоения. В выражении могут участвовать как и конкретные значения так и другие переменные.

```
1  <script>
2
3      var a = 5;
4      var b = 7;
5      var c = 3;
6
7      var c = (a + b) * c + 45;
8
9      console.log(c);
10
11 </script>
```



Операторы и операции (их приоритеты)

operator	Описание
. [ ] ( )	Доступ к полям, индексация массивов, вызовы функций и группировка выражений
++ -- - ~ ! delete new typeof void	Унарные операторы, тип возвращаемых данных, создание объектов, неопределенные значения
* / %	Умножение, деление, деление по модулю
+ - +	Сложение, вычитание, объединение строк
<< >> >>>	Сдвиг битов
< <= > >= instanceof	Меньше, меньше или равно, больше, больше или равно, instanceof
== != === !==	Равенство, неравенство, строгое равенство, строгое неравенство
&	Побитовое И
^	Побитовое исключающее ИЛИ
|	Побитовое ИЛИ
&&	Логическое И
||	Логическое ИЛИ
?:	Условный оператор
= *OP*=	Присваивание, присваивание с операцией (например += и &=)
,	Вычисление нескольких выражений

https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Operators/Operator_Precedence

«Скобками программу не испортишь» (с)

У операторов есть приоритеты, уровней приоритета полтора десятка и помнить затруднительно. Поэтому в случае сомнений какая операция будет первой а какая второй – смело используйте скобки. Принцип их применения такой же как и в математике – скобки повышают приоритет операции в них записанной.

```
var c = (a + b) * 4 - (++b);
```

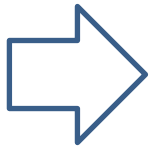
Алгоритмы

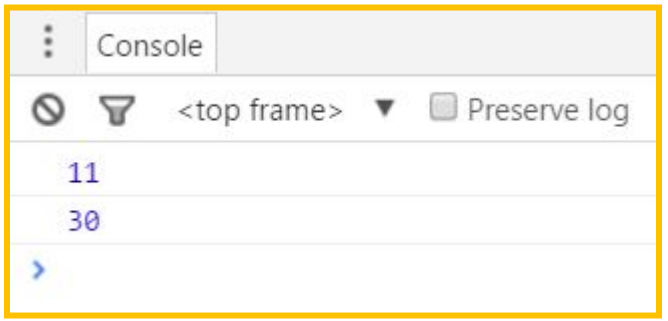
Задача: Написать скрипт, который рассчитает сколько гривен в день приносит депозит размещенный на полтора года под 22% годовых?

Пора писать код….

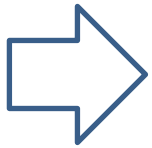
Операторы, операнды и операции и…

```
1  <script>
2
3      var a = 5;
4      var b = 6;
5
6      console.log(a + b);
7      console.log(a * b);
8
9  </script>
```






```
1  <script>
2
3      var a = "5";
4      var b = "6";
5
6      console.log(a + b);
7      console.log(a * b);
8
9  </script>
```



?!?

Типы данных (переменных)

Тип данных – пометка для компьютера как относиться к тем или иным данным (переменным) и какие операции с ними возможно проводить.

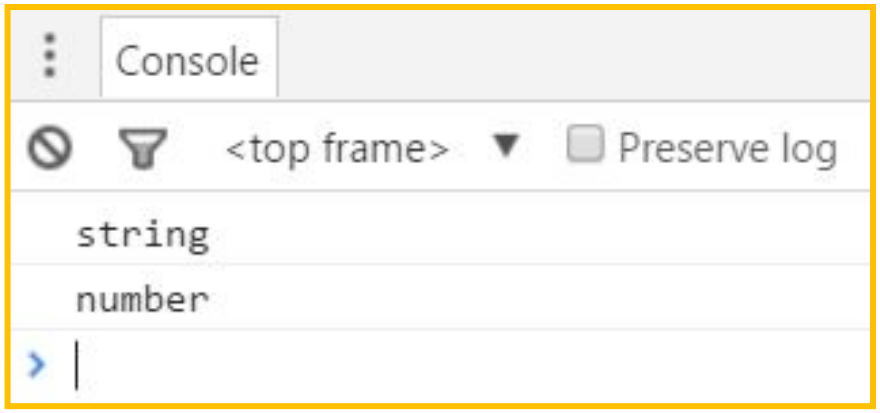
*Тип определяет возможные значения и их смысл, а также **операции** которое возможны над этим типом данных.*

Разные типы требуют разного подхода.

Тип переменной

*В **JavaScript** отсутствует жёсткая типизация данных, при которой тип переменной определяется при её объявлении. В **JavaScript** тип переменной определяется при присвоении ей значения. И может меняться при каждом новом присвоении. Мы можем узнать тип переменной воспользовавшись функцией **typeof**.*

```
1  <script>
2
3      var a = "5";
4      console.log(typeof(a));
5
6      a = 5;
7      console.log(typeof(a));
8
9  </script>
```



Типы/ types

Тип данных – пометка для компьютера как относиться к тем или иным данным и какие операции с ними возможно проводить.

*Тип определяет возможные значения и их смысл, а также **операции** которое возможны над этим типом данных.*

5 типов: number, string, boolean, function, object.

1 «служебный» тип: undefined.

+1 тип добавлен в ECMAScript-2015: symbol.

Javascript не типизированный язык. Тип переменной не указывается при объявлении и может меняться по ходу выполнения программы.

Pascal/Delphi

```
1  var
2  a: integer;
3  b: real;
4  c: string;
```

C/C++/C#/Java

```
1  int a;
2  double b;
3  char c;
```

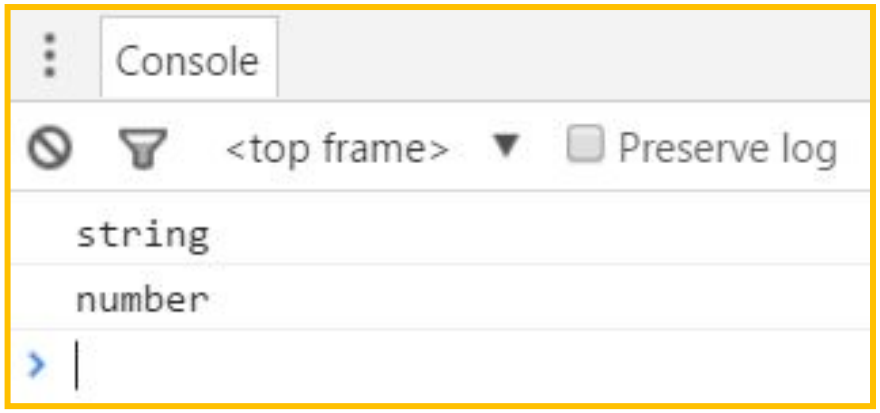
JavaScript

```
1  var a = 123;
2  var b = "text";
3  var c = 234.55;
4  var d;
```

Тип переменной

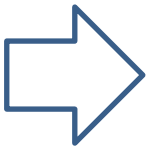
В JavaScript отсутствует жёсткая типизация данных, при которой тип переменной определяется при её объявлении. В JavaScript тип переменной определяется при присвоении ей значения. И может меняться при каждом новом присвоении.

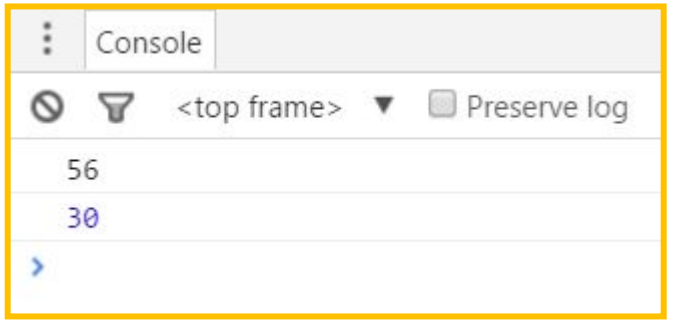
```
1  <script>
2
3      var a = "5";
4      console.log(typeof(a));
5
6      a = 5;
7      console.log(typeof(a));
8
9  </script>
```



Преобразование типов в JavaScript

```
1  <script>
2
3      var a = "5";
4      var b = 6;
5
6      console.log(a + b);
7      console.log(a * b);
8
9  </script>
```

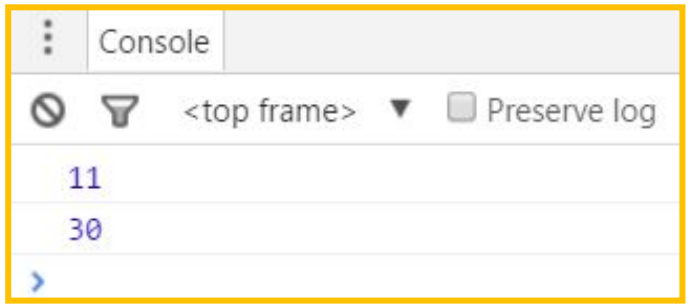




Подробнее:
https://learn.javascript.ru/types-conversion

Преобразование типов в JavaScript

```
1  <script>
2
3      var a = "5";
4      var b = 6;
5
6      console.log(Number(a) + b);
7      console.log(Number(a) * b);
8
9  </script>
```



Подробнее:
https://learn.javascript.ru/types-conversion

Преобразование типов в JavaScript

Где зарыта собака?

```
<script>

    var a = prompt("Введие любое число от 1 до 100");

    console.log(a);

    console.log(typeof(a));

</script>
```

Подробнее:
https://learn.javascript.ru/types-conversion

Преобразование строк в числа

```
1  <script>
2      var data = prompt("Enter you age: ");
3      data = Number(data);
4      console.log(data, typeof(data));
5  </script>
```

*Используя функцию **Number()** мы явно указываем системе, что хотим преобразовать значение к числовому типу.*

```
1  <script>
2      var data = +prompt("Enter you age: ");
3
4      console.log(data, typeof(data));
5  </script>
```

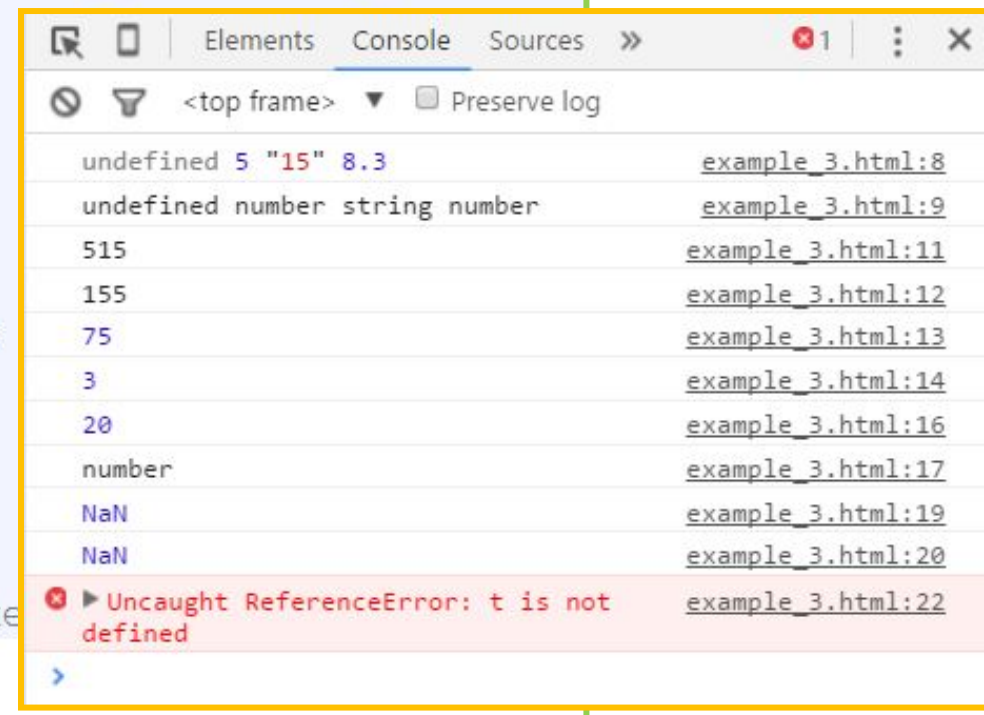
*Добавив унарный оператор **+** мы заставляем систему неявно преобразовать значение в числовое.*

```
1  <script>
2      var data = prompt("Enter you age: ");
3      data = parseInt(data);
4      console.log(data, typeof(data));
5  </script>
```

Функция parseInt () – позволяет преобразовать строку в число, при этом спокойно относиться к «лишним» символам в строке.

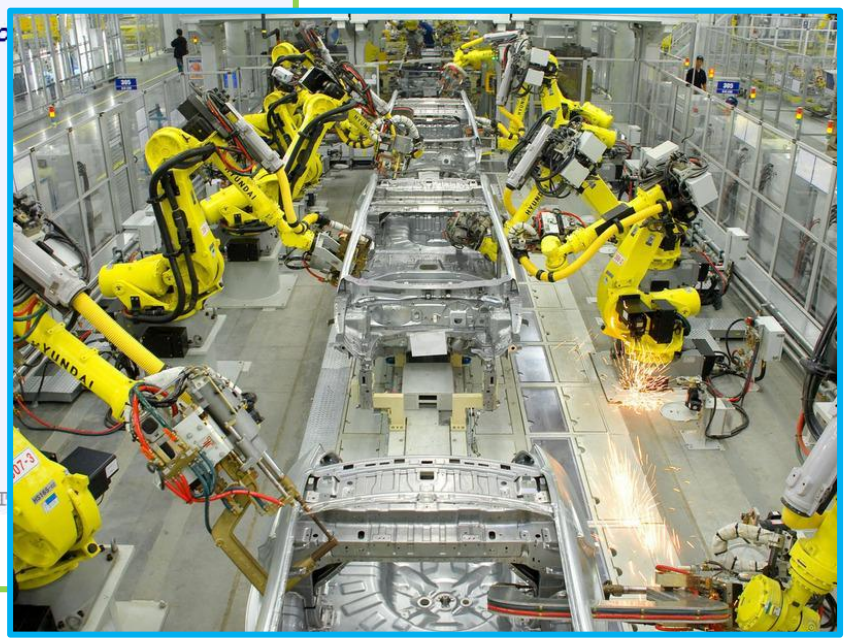
Операции и типы

```
<!DOCTYPE html><html><head>
<script>
    var a;
    var b = 5;
    var c = "15";
    var d = 8.3;

    console.log(a, b, c, d);
    console.log(typeof(a), typeof(b), typeof(c), typeof(d));

    console.log(b + c);
    console.log(c + b);
    console.log(b * c);
    console.log(c / b);

    console.log(b + Number(c));
    console.log(typeof(Number(c)));

    console.log(a + b);
    console.log("text" * b);

    console.log(b + t);
    console.log("Это сообщение вы не
</script>
</head><body></body></html>
```



Ход выполнения программы

```
<!DOCTYPE html><html><head>
<script>
    var a;
    var b = 5;
    var c = "15";
    var d = 8.3;

    console.log(a, b, c, d);
    console.log(typeof(a), typeof(b), typeo

    console.log(b + c);
    console.log(c + b);
    console.log(b * c);
    console.log(c / b);

    console.log(b + Number(c));
    console.log(typeof(Number(c)));

    console.log(a + b);
    console.log("text" * b);

    console.log(b + t);
    console.log("Это сообщение вы не увидит
</script>
</head><body></body></html>
```



Выполнение программы подобно сборке автомобиля на конвеере, каждое выражение (каждый оператор) вносит свои изменения в данные (переменные), чтобы на выходе (в итоге) получился готовый результат.

Немного практики

Немного практики №1

Задача: *Написать скрипт для сети пунктов обмена валют. Скрипт должен сообщать сколько гривен нужно для совершения покупки той или иной суммы долларов.*

1. *Недостаток информации;*
2. *Избыточная информация;*
3. *Косвенно влияющие аспекты.*

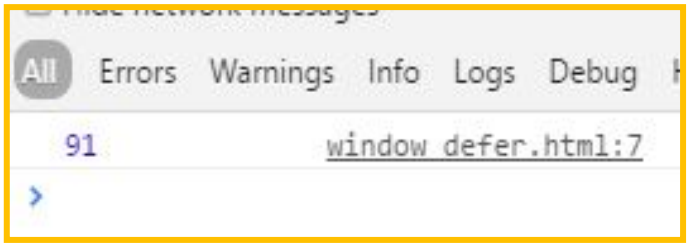
Немного практики №2

Пользователь вводит количество секунд, определить сколько часов минут секунд в указанном количестве секунд.

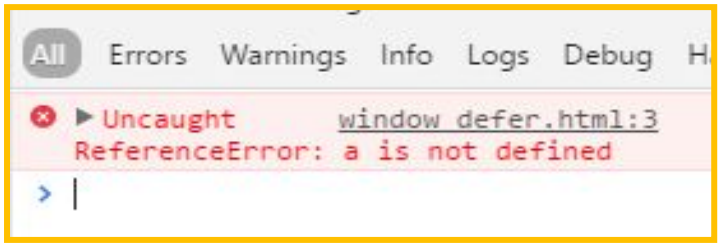
Нововведения

Директива “use strict”

```
1  <script>
2
3      a = 56;
4      b = 35;
5      c = a + b;
6
7      console.log(c);
8
9  </script>
```




```
1  <script>
2      "use strict";
3      a = 56;
4      b = 35;
5      c = a + b;
6
7      console.log(c);
8
9  </script>
```



*Директива “**use strict**” говорит браузеру, что следует относиться к JavaScript коду строго по стандарту **ECMAScript 5**. Все «попустительства» поддерживаемые для совместимости со старыми стандартами перестают действовать.*

Что перестанет работать: https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Strict_mode

let и область видимости (ECMAScript-2015)

```
1  <script>
2
3      var x = true;
4
5      if(x){
6          let y = "Test";
7      };
8
9      alert(y);
10
11 </script>
```

?!?

```
1  <script>
2
3      var x = true;
4
5      if(x){
6          var y = "Test";
7      };
8
9      alert(y);
10
11 </script>
```

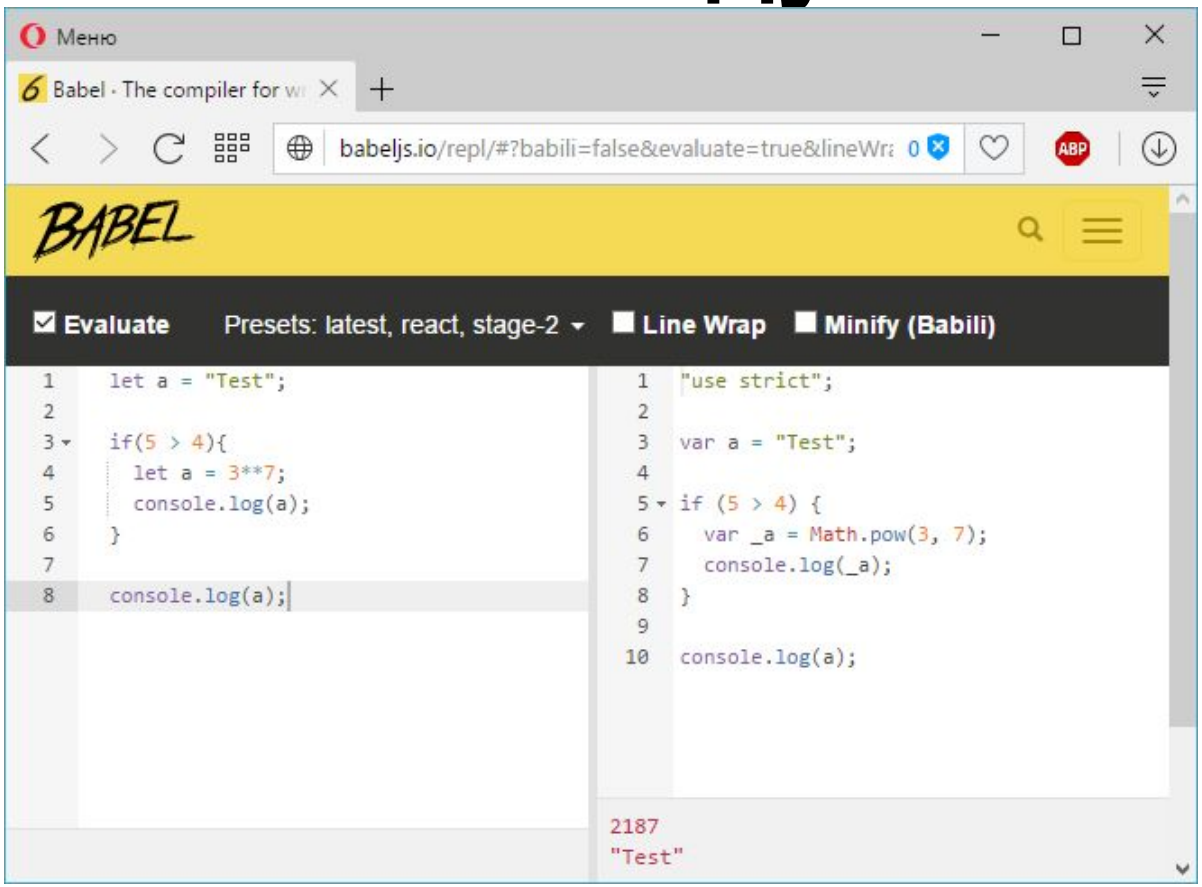
*Оператор **let** объявляет переменную, но такие переменные существуют только в той области видимости (тех операторных скобках) в которой они объявлены, и не видны снаружи, в отличии от переменных объявленных через **var**.*

Если нельзя, но очень хочется ECMAScript-2015

http://babeljs.io/

Babel.JS *– это транспайлер, переписывающий код на ES-2015 в код на предыдущем стандарте ES5.* ***Не забываем использовать “use strict”.***

Babel.js – ECMASCript-2015 повсюду



Babel.JS *– это транспайлер, переписывающий код на ES-2015 в код на ES5.* ***Не забываем использовать “use strict”.***

Babel.js – ECMAScript-2015 повсюду

```
<script src="
https://cdnjs.cloudflare.com/ajax/libs/babel-standalone/6.18
.1/babel.min.js"></script>
<script type="text/babel">

    let a = "Test";

    if(5 > 4){
        let a = 3**7;
        console.log(a);
    }

    console.log(a);

</script>
```

<script src="https://cdnjs.cloudflare.com/ajax/libs/babel-standalone/6.18.1/babel.min.js"></script>

Babel.JS *можно использовать по другому – включить специальный скрипт, который перепишет весь* ***ES-2015*** *код в* ***ECMAScript 5.*** *Подробности:* http://babeljs.io/docs/setup/#installation

Домашнее задание

Домашнее задание №1

Задача: *Разработать скрипт который на основе роста и веса пользователя рассчитывает его индекс массы тела (для ввода использовать функцию* ***prompt()****, для вывода –* ***alert()****).*

Домашнее задание №2

***Задача**: Необходимо написать скрипт который получает температуру в градусах по Цельсию, а выводит её эквивалент в градусах по Фаренгейту и по Кельвину.*

Домашнее задание №3

«Задача банкомата» Написать скрипт, который спрашивает у пользователя сумму, а в ответ сообщает купюры каких номинала, и в каком количестве необходимо выдать, а также суммарное количество купюр. При этом суммарное количество купюр было минимально возможным. Помните, что у нас в стране купюры номинала 1, 2, 5, 10, 20, 50, 100, 200, 500 гривен.

W3Schools.com

http://www.w3schools.com/js/

Майкл Моррисон
«Изучаем JavaScript»