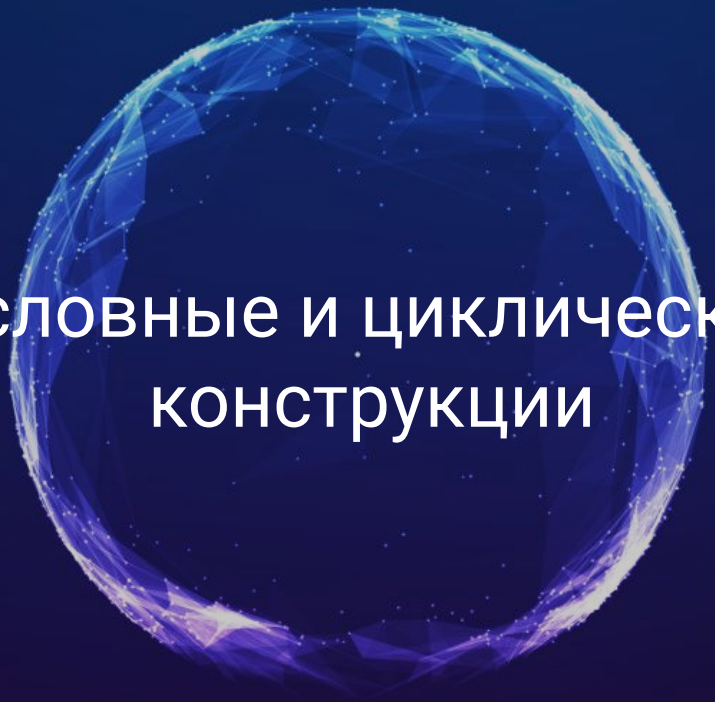


Условные и циклические конструкции

IT Education Academy

WWW.ITEA.UA

I T E A

www.itea.ua

Егор Маркевич

- Инструктор **IT Education Academy**
- Frontend developer **Matrixian Group**

Урок 2. Условные и циклические конструкции

- Условные конструкции
- Логические операторы
- Циклические конструкции
- Директивы break и continue

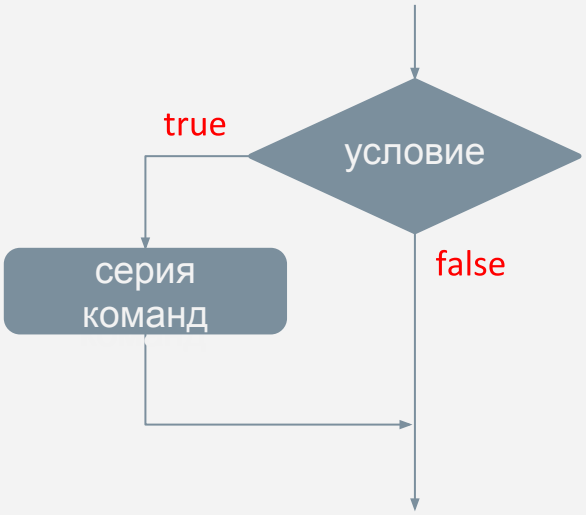
Понятие "условная конструкция"

Оператор ветвления (условная инструкция, условный оператор) —оператор, конструкция языка программирования, обеспечивающая выполнение определённой команды (набора команд) только при условии истинности некоторого логического выражения, либо выполнение одной из нескольких команд (наборов команд) в зависимости от значения некоторого выражения.

Разновидности условных конструкций:

- Условный оператор: if … else
- Тернарный оператор: … ? … : …
- Оператор многозначного выбора: switch

Условный оператор “if”

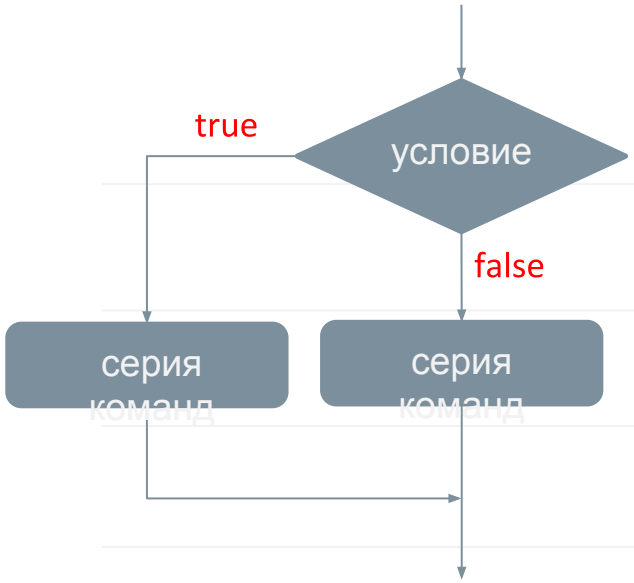


Оператор **if** проверяет переданное ему условие: если результат — **true**, то выполняется команда (серия команд) в операторных скобках.

```
var price = 150;
    If (price < 200) {
      console.log(“покупаю”);
    }
```

В логическом контексте число 0, пустая строка "", null и undefined, а также NaN являются false, остальные значения — true.

Условный оператор “if..else”



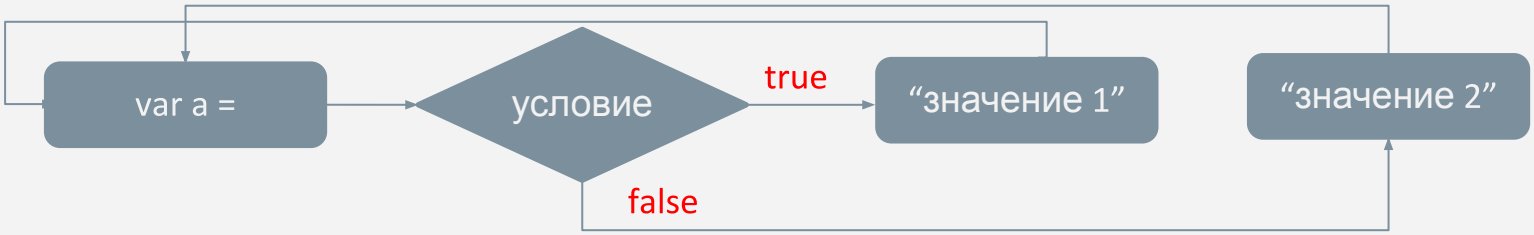
Необязательный блок **else** оператора **if** выполняется в том случае, если условие ложное (**false**).

```
var price = 150;
    If (price < 200) {
      console.log(“покупаю”);
    }
    else {
      console.log(“не покупаю”);
    }
```

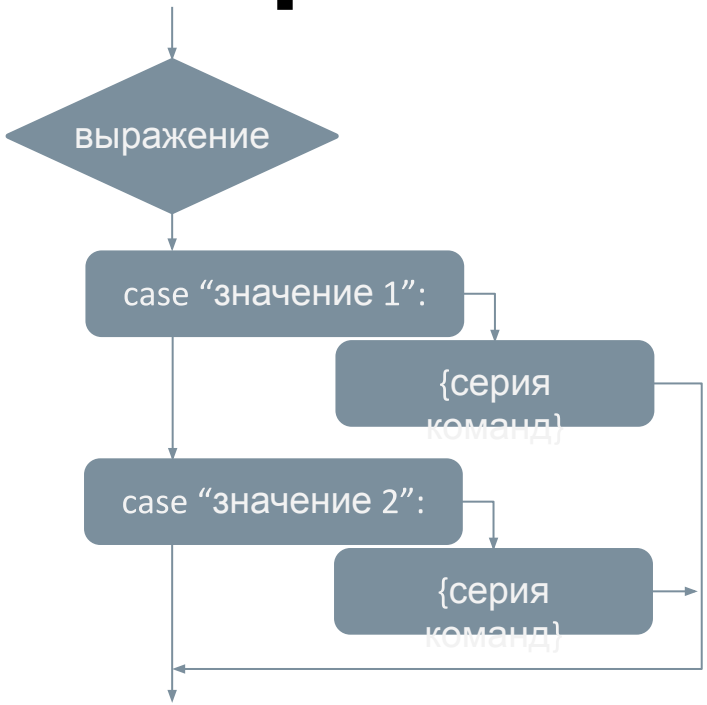
Тернарный оператор

Тернарный оператор имеет следующий вид: **условие ? значение1 : значение2**. Он проверяет условие, затем, если оно истинно, возвращает «значение1» , если ложно — «значение2».

```
var answer = (price < 200) ? “покупаю” : “не покупаю”;
console.log(answer);
```



Оператор многозначного выбора “switch”



Конструкция **switch** заменяет собой сразу несколько if. Применяется в том случае, если нужно сравнить выражение сразу с несколькими вариантами.

```
switch (wordToTranslate) {
        case “cat”: { console.log(“кошка”); break; }
        case “dog”: { console.log(“собака”); break; }
        case “mouse”: { console.log(“мышь”); break; }

        default: { console.log(“нет слова в словаре”); }
}
```

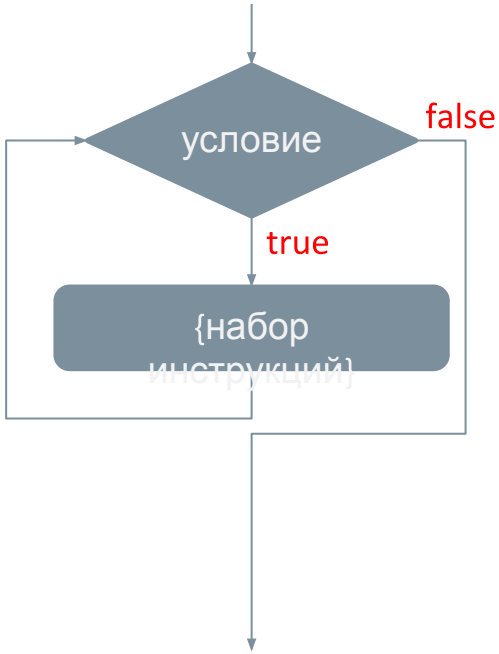
Понятие “циклическая конструкция”

Цикл — разновидность управляющей конструкции в высокоуровневых языках программирования, предназначенная для организации многократного исполнения набора инструкций. Также циклом может называться любая многократно исполняемая последовательность инструкций, организованная любым способом (например, с помощью условного перехода).

Разновидности циклических конструкций:

- Цикл с предусловием: while
- Цикл с постусловием: do…while
- Цикл со счётчиком: for

Цикл с предусловием “while”

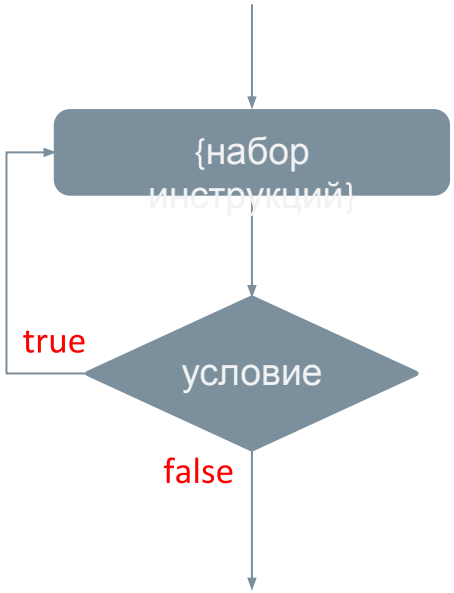


Цикл **while** выполняет набор инструкций до тех пор, пока условие удовлетворяет истинности.

Итерация — одно повторение цикла.

```
var a = 0;
while (a < 10) {
   console.log(a);
   a++;
}
```

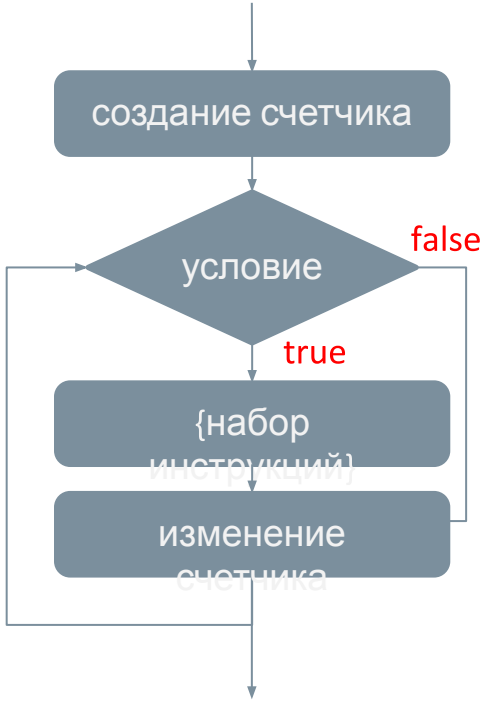
Цикл с постусловием “do…while”



Цикл **do…while** вначале выполняет набор инструкций, а затем проверяет истинность условия.

```
var a = 0;
do {
    console.log(a);
    a++;
}
while (a < 10);
```

Цикл со счетчиком “for”



Цикл **for** имеет вид: **for (счетчик; условие; шаг) { тело цикла }**, и выполняется с определенным шагом до тех пор, пока счетчик удовлетворяет условию.

Любая часть for может быть пропущена, однако сами точки с запятой обязательно должны присутствовать, иначе будет ошибка синтаксиса.

```
for (var i = 0; i < 10; i++) {
    console.log(i);
}
```

Операторы “break” и ”continue”

Операторы **break** и **continue** используются для более гибкого управления циклом:

- **break** — выход из цикла при необходимых условиях.
- **continue** — прекращение выполнения текущей итерации цикла и переход к следующей.

```
16      for (var i = 0; i < iteration_count; i++) {
17          if (i == exit) {
18              console.log("Работа цикла завершена принудительно");
19              console.log("***");
20              break;
21          }
22          else {
23              console.log(i);
24          }
25      }
```

Логические операторы

```
x=6
y=3
x==5 || y==5 Return False
```

```
x=6
y=3
x<10 && y>1 Return True
```

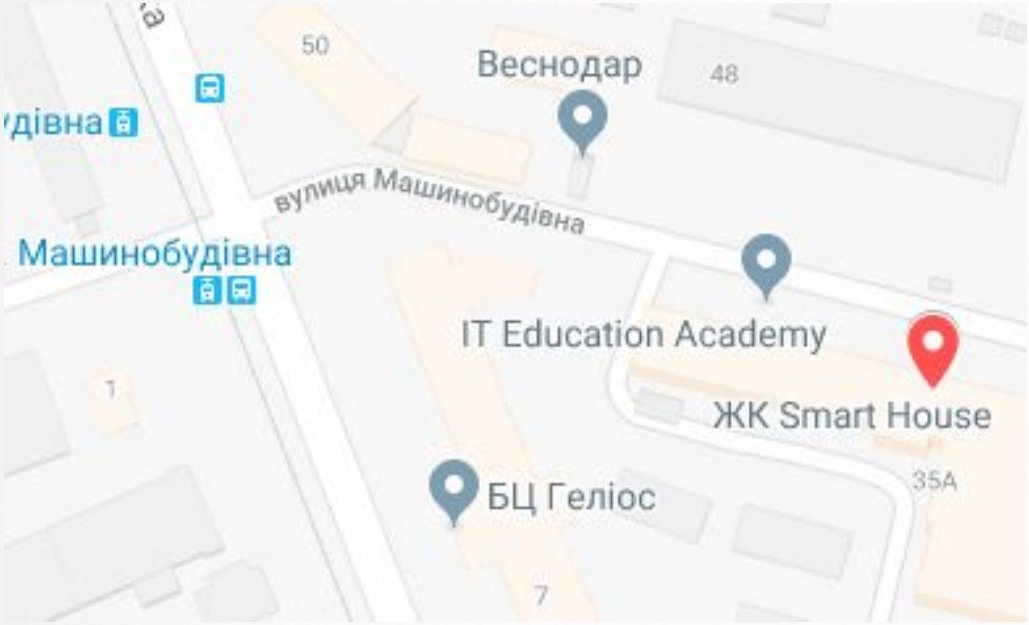
```
x=6
y=3
!(x==y) Return True
```

В JavaScript поддерживаются операторы **|| (ИЛИ)**, **&& (И)** и **! (НЕ)**.

Логическое ИЛИ в классическом программировании работает следующим образом: «если хотя бы один из аргументов true, то возвращает true, иначе — false.

Логическое И возвращает true, если оба аргумента истинны, а иначе — false.

Логическое НЕ приводит аргумент к логическому типу и возвращает противоположное значение.



I T E A

КОНТАКТНЫЕ ДАННЫЕ

ITEA

ЖК “Smart House”, ул. Машиностроительная, 41 (м. Берестейская)

бул. Дарницкий, 8В

+38 (044) 599-01-79
facebook.com/Itea
info@itea.ua
itea.ua