7. Циклы for и оператор break ✅

Привет, сегодня мы изучим циклы for и оператор break

Введение

При написании скриптов зачастую встаёт задача сделать однотипное действие много раз.

На прошлом уроке ты познакомился с циклом while и вообще узнал что такое циклы. Если нужно многократно выполнить одну и ту же последовательность действий, можно воспользоваться **циклом**, чтобы повторить определенные код.

На этом же уроке, мы изучим более сложный, но при этом самый распространённый цикл — цикл `for`.

Цикл for

Цикл for выглядит примерно так:

```
for (начало; условие; шаг) {
  // ... тело цикла ...
}
```

Давай разберем как всё работает на конкретном примере

```
for (let i = 0; i < 3; i++) { // выведет 0, затем 1, затем 2
  alert(i);
}
```

В это примере цикл выполняет `alert(i)`
для `i`
от `0`
до `3` (но не включая)

Рассмотрим конструкцию `for` подробней:

часть		
начало	`let i = 0`	Выполняется один раз при входе в цикл
условие	`i < 3`	Проверяется *перед* каждой итерацией цикла. Если оно вычислится в `false`, цикл остановится.
тело	`alert(i)`	Выполняется снова и снова, пока условие вычисляется в `true`.
шаг	`i++`	Выполняется *после* тела цикла на каждой итерации *перед* проверкой условия.

```
Выполнить *начало*
→ (Если *условие* == true → Выполнить *тело*, Выполнить *шаг*)
→ (Если *условие* == true → Выполнить *тело*, Выполнить *шаг*)
→ (Если *условие* == true → Выполнить *тело*, Выполнить *шаг*)
→ ...
```

То есть, *начало* выполняется один раз, а затем каждая итерация заключается в проверке *условия*, после которой выполняется *тело* и *шаг*.

Вот в точности то, что происходит в нашем случае:

```
// for (let i = 0; i < 3; i++) alert(i)

// Выполнить начало
let i = 0;
```

```
// Если условие == true → Выполнить тело, Выполнить шаг
if (i < 3) { alert(i); i++ }
// Если условие == true → Выполнить тело, Выполнить шаг
if (i < 3) { alert(i); i++ }
// Если условие == true → Выполнить тело, Выполнить шаг
if (i < 3) { alert(i); i++ }
// ...конец, потому что теперь i == 3
```

Встроенное объявление переменной

В примере переменная счётчика `i` была объявлена прямо в цикле. Это так называемое «встроенное» объявление переменной. Такие переменные существуют только внутри цикла.

```
for ( let  i = 0; i < 3; i++) {
  alert(i); // 0, 1, 2
}
alert(i); // ошибка, нет такой переменной
```

Вместо объявления новой переменной мы можем использовать уже существующую:

```
let i = 0;

for (i = 0; i < 3; i++) { // используем существующую переменную
  alert(i); // 0, 1, 2
}

alert(i); // 3, переменная доступна, т.к. была объявлена снаружи цикла
```

Пропуск частей «for»

Любая часть `for` может быть пропущена.

Для примера, мы можем пропустить `начало` если нам ничего не нужно делать перед стартом цикла.

Вот так:

```
let i = 0; // мы уже имеем объявленную i с присвоенным значением

for (; i < 3; i++) { // нет необходимости в "начале"
  alert( i ); // 0, 1, 2
}
```

Можно убрать и `шаг`:

```
let i = 0;

for (; i < 3;) {
  alert( i++ );
}
```

Это сделает цикл аналогичным `while (i < 3)`.

А можно и вообще убрать всё, получив бесконечный цикл:

```
for (;;) {
  // будет выполняться вечно
}
```

При этом сами точки с запятой `;` обязательно должны присутствовать, иначе будет ошибка синтаксиса.

Прерывание цикла: «break»

Обычно цикл завершается при вычислении *условия* в `false`.

Но мы можем выйти из цикла в любой момент с помощью специальной директивы `break`.

Например, следующий код подсчитывает сумму вводимых чисел до тех пор, пока посетитель их вводит, а затем – выдаёт:

```
let sum = 0

// запускаем бесконечный цикл
for (;;) {
  let value = +prompt("Введите число", ''); // знак + перез promt преобразует
                                            // ответ из string в number
  if{!value}{
    break // если пользователь ничего не ввел, цикл останавливается
  }

  sum += value; // суммируем ответы
}

alert(sum) // выводим сумму после окончания цикла
```

Директива `break` в строке `(*)` полностью прекращает выполнение цикла и передаёт управление на строку за его телом, то есть на `alert`.

Задание 1

Начнем с простого задания

Нужно вывести alert числа от 1 до 5

То есть вот так:

1,2,3,4,5

Задание 2

Теперь тебе необходимо сделать тоже самое, только наоброт

Вывести в alert числа от 5 до 1

То есть вот так:

5,4,3,2,1

Задание 3

Необходимо вывести на экран таблицу умножения на 3:

3*1=3

3*2=6

3*3=9

3*4=12

3*5=15

3*6=18

3*7=21

3*8=24

3*9=27

3*10=30