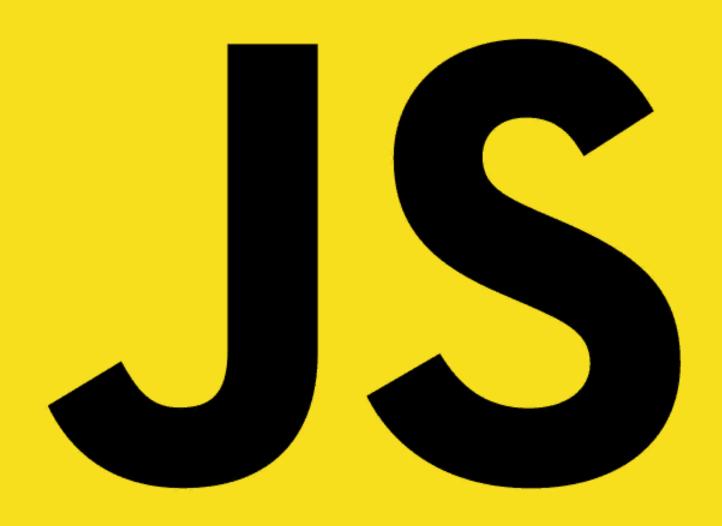
# Язык JavaScript (итераторы и генераторы)



## ИТЕРАТОРЫ (ES-2015)

В JavaScript добавлена концепция «итерируемых» (iterable) объектов.

Итерируемые или, иными словами, «перебираемые» объекты – это те, содержимое которых можно перебрать в цикле.

Для перебора таких объектов добавлен новый синтаксис цикла: for..of

Например, перебираемыми является массив и строка.

```
let arr = [1, 2, 3]; // массив — пример итерируемого объекта
for (let value of arr) {
   alert(value); // 1, затем 2, затем 3
}
for (let char of "Привет") {
   console.log(char); // Выведет по одной букве: П,р,и,в,е,т
}
```

# ИТЕРАТОРЫ (ES-2015)

Объект является итератором, если он умеет обращаться к элементам коллекции по одному за раз, при этом отслеживая свое текущее положение внутри этой последовательности. В JavaScript итератор - это объект, который предоставляет метод next(), возвращающий следующий элемент последовательности. Этот метод возвращает объект с двумя свойствами: done и value.

## ИТЕРАТОРЫ (ES-2015)

Для возможности использовать объект в for..of нужно создать в нём свойство с названием Symbol.iterator (системный символ). При вызове метода Symbol.iterator перебираемый объект должен возвращать другой объект («итератор»), который умеет осуществлять перебор.

```
function makeIterator(){
    let arr = this.arr;
    let nextIndex = 0;
    return {
       next: function(){
           return nextIndex < array.length ?</pre>
               {value: array[nextIndex++], done: false} :
               {done: true};
let myObj = { arr:['yo', 'ya'], [Symbol.iterator]:makeIterator}
```

В то время как пользовательские итераторы могут быть весьма полезны, при их программировании требуется уделять серьезное внимание поддержке внутреннего состояния.

Генераторы предоставляют мощную альтернативу: они позволяют определить алгоритм перебора, написав единственную функцию, которая умеет поддерживать собственное состояние.

Генераторы - это специальный тип функции, который работает как фабрика итераторов. Функция становится генератором, если содержит один или более yield операторов и использует function\* синтаксис.

Функция-генератор может быть объявлена путем добавления символа \* (звездочка) после ключевого слова function:

```
function* makeGenerator() { //тело }
```

Выполнение функции makeGenerator() можно приостановить с помощью ключевого слова yield и вернуть вызвавшему объекту некоторое значение:

```
function* makeGenerator() {
    yield 'Hello World';
    console.log('Re-entered');
}
```

При вызове функция генератор возвращает новый объект итератора:

```
const gen = makeGenerator();
```

Запускает/возобновления работы генератора осуществляется с помощью вызова функции next():

#### Пример:

```
function* idMaker(){
  let index = 0;
  while(true)
    yield index++;
let gen = idMaker();
console.log(gen.next().value); // 0
console.log(gen.next().value); // 1
console.log(gen.next().value); // 2
```

#### Пример:

```
let myIterable = {};
myIterable[Symbol.iterator] = function* () {
    yield 1;
    yield 2;
    yield 3;
};
let arr = [...myIterable]; // [1, 2, 3]
```

#### ПОЛЕЗНЫЕ ССЫЛКИ

https://learn.javascript.ru/iterator - Итераторы

https://learn.javascript.ru/generator - Генераторы

https://developer.mozilla.org/ru/docs/Web/JavaScript/Guide
/Iterators and Generators - Итераторы и генераторы