How to iterate (keys, values) in javascript?

Asked 3 years, 9 months ago Active 2 months ago Viewed 323k times

243

I have a dictionary that has the format of

```
dictionary = {0: {object}, 1:{object}, 2:{object}}
```

How can I iterate through this dictionary by doing something like

44

```
for((key,value) in dictionary){
  //Do stuff where key would be 0 and value would be the object
}
```

javascript object iteration

edited Jan 21 '16 at 1:23 thefourtheye 177k 30 331 396

asked Jan 21 '16 at 1:11 nbroeking 2,162 3 13 33

3 `for (let [key, value] of Object.entries(obj))` , need Babel. – elclanrs Jan 21 '16 at 1:14

3 possible duplicate – Tholle Jan 21 '16 at 1:14

1 @elclanrs Its in ES2016 and it is not standardized yet :-) – thefourtheye Jan 21 '16 at 1:16

2 Possible duplicate of For-each over an array in JavaScript? – sdc Jan 21 '16 at 1:20

@dooagain, it is not an array in this question. – zangw Jan 21 '16 at 1:23

9 Answers

356

1. In ECMAScript 5, it is not possible.
2. In ECMAScript 2015, it is possible with `Map` s.
3. In ECMAScript 2017, it would be readily available.

ECMAScript 5:

No, its not possible with objects.

You should either iterate with `for..in` , or `Object.keys` , like this

```
for (var key in dictionary) {
    // check if the property/key is defined in the object itself, not in parent
    if (dictionary.hasOwnProperty(key)) {
        console.log(key, dictionary[key]);
    }
}
```

Note: The `if` condition above is necessary, only if you want to iterate the properties which are `dictionary` object's very own. Because `for..in` will iterate through all the inherited enumerable properties.

Or

```
Object.keys(dictionary).forEach(function(key) {
    console.log(key, dictionary[key]);
});
```

ECMAScript 2015

In ECMAScript 2015, you can use `Map` objects and iterate them with `Map.prototype.entries` . Quoting example from that page,

```
var myMap = new Map();
myMap.set("0", "foo");
myMap.set(1, "bar");
myMap.set({}, "baz");
```

```
console.log(mapIter.next().value); // [1, "bar"]
console.log(mapIter.next().value); // [Object, "baz"]
```

Or iterate with `for..of` , like this

```
'use strict';

var myMap = new Map();
myMap.set("0", "foo");
myMap.set(1, "bar");
myMap.set({}, "baz");

for (const entry of myMap.entries()) {
  console.log(entry);
}
```

Output

```
[ '0', 'foo' ]
[ 1, 'bar' ]
[ {}, 'baz' ]
```

Or

```
for (const [key, value] of myMap.entries()) {
  console.log(key, value);
}
```

Output

```
0 foo
1 bar
{} baz
```

ECMAScript 2017

```
'use strict';

const object = {'a': 1, 'b': 2, 'c' : 3};
for (const [key, value] of Object.entries(object)) {
  console.log(key, value);
}
```

Output

```
a 1
b 2
c 3
```

edited Dec 28 '18 at 10:36

ctrl-alt-delor
4,446 3 26 44

answered Jan 21 '16 at 1:14

thefourtheye
177k 30 331 396

1 A basic doubt here. I landed here looking for how to do this in node.js, which is javascript on server side. How do I know which ES version applies in my case. Also, in case of regular javascript users, what is the proper way to support as I understand that ES version depends on the client's browser? – Sandeepan Nath Aug 28 '18 at 13:43

2 @SandeepanNath You can use websites like node.green to know if a particular ES feature is supported in your Node.js. As far as browsers are concerned, people generally target the version which is widely supported, in this case, ES5. Apart from this, transpilers (like Babel) help convert ES2015+ code to ES5. – thefourtheye Aug 29 '18 at 7:51

1 I like `Object.keys(dictionary).forEach(function(key) {…` very readable, and compatible. – ctrl-alt-delor Dec 28 '18 at 10:50

3 `Object.entries(object).forEach(([key, val]) => {...});` – Krimson Feb 9 at 7:01

ECMAScript 2015 solution above threw "TypeScript and Iterator: Type 'IterableIterator<T>' is not an array type" but plain ol myMap().foreach() worked well. – ttugates Apr 10 at 18:13

42

Try this:

```
dict = {0:{1:'a'}, 1:{2:'b'}, 2:{3:'c'}}
for (var key in dict){
```

```
0 Object { 1="a"}
1 Object { 2="b"}
2 Object { 3="c"}
```

edited Jul 14 '16 at 23:13

Alexander Irbis
83 8

answered Jan 21 '16 at 1:18

alex10
1,180 2 13 22

32

The `Object.entries()` method has been specified in ES2017 (and is supported in all modern browsers):

```
for (const [ key, value ] of Object.entries(dictionary)) {
    // do something with `key` and `value`
}
```

Explanation:

- `Object.entries()` takes an object like `{ a: 1, b: 2, c: 3 }` and turns it into an array of key-value pairs: `[ [ 'a', 1 ], [ 'b', 2 ], [ 'c', 3 ] ]`.
- With `for ... of` we can loop over every entry of the so created array.
- Since we are *guaranteed* that each of the so iterated array items is another two-entry-array, we can use destructuring to directly assign variables `key` and `value` to its first and second item.

edited May 26 at 19:20

answered Aug 17 '17 at 9:25

Loilo
4,745 3 27 37

12

Try this:

```
var value;
for (var key in dictionary) {
    value = dictionary[key];
    // your code here...
}
```

AMACB
1,009 1 15 23

You can do something like this :

9

```
dictionary = {'ab': {object}, 'cd':{object}, 'ef':{object}}
var keys = Object.keys(dictionary);

for(var i = 0; i < keys.length;i++){
   //keys[i] for key
   //dictionary[keys[i]] for the value
}
```

answered Aug 17 '17 at 8:48

Dhaval Chaudhary
2,021 1 12 27

1 Beautiful! I love how your answer works in ECMAscript 5 despite the accepted and most upvoted answer saying it's not possible. You deserve all a lot more upvotes. – liljoshu Jan 31 at 1:31

I think the fast and easy way is

2

```
Object.entries(event).forEach(k => {
    console.log("properties ... ", k[0], k[1]); });
```

just check the documentation https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/entries

answered Sep 26 '18 at 11:38

Jonathan
121 3

even better: Object.entries(obj).forEach(([key, value]) => { console.log(`${key} ${value}`); }); – Hani Nov 22 '18 at 1:50

using swagger-ui.js

1

you can do this -

```
_.forEach({ 'a': 1, 'b': 2 }, function(n, key) {
    console.log(n, key);
 });
```

answered Nov 29 '17 at 23:42

deeshank

1,575 4 19 26

You can use below script.

1

```
var obj={1:"a",2:"b",c:"3"};
for (var x=Object.keys(obj),i=0;i<x.length,key=x[i],value=obj[key];i++){
    console.log(key,value);
}
```

outputs
1 a
2 b
c 3

edited Aug 19 at 10:37

answered Aug 19 at 8:43

Michael Piper

11 2

#will output #c 3 #1 a #2 b – Michael Piper Aug 19 at 8:44

3 Consider adding an explanation to your answer, code alone is less helpful. Also, you can edit your answer, comments are not meant to be an extension to the answer in the way you are using them – Виктор Aug 19 at 9:10

0

```
for (var key in dictionary) {
    if (!dictionary.hasOwnProperty(key)) {
        continue;
    }
    console.log(key, dictionary[key]);
}
```

answered Dec 21 '18 at 17:30

kloddant

471 2 12