Iterables e Iterators

São convenções implementadas por Arrays, Maps, Sets e Strings que os tornam iteráveis por meio de um protocolo de iteração

iterables_iterators_1.js — javascriptmasterclass

iterables_iterators_1.js

```
1  const languages = ["Fortran", "Lisp", "COBOL"];
2  for (let i = 0; i < languages.length; i++) {
3      console.log(languages[i]);
4  }
5
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_1.js
Fortran
Lisp
COBOL
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_2.js — javascriptmasterclass

JS iterables_iterators_2.js

```
1  const languages = ["Fortran", "Lisp", "COBOL"];
2  for (let i in languages) {
3      console.log(languages[i]);
4  }
5
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_2.js

Fortran
Lisp
COBOL
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_3.js — javascriptmasterclass

JS iterables_iterators_3.js

```
1  const languages = ["Fortran", "Lisp", "COBOL"];
2  languages.forEach(function (language) {
3      console.log(language);
4  });
5
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_3.js

Fortran
Lisp
COBOL
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_4.js — javascriptmasterclass

JS iterables_iterators_4.js

```
1  const languages = ["Fortran", "Lisp", "COBOL"];
2  languages.forEach((language) => {
3      console.log(language);
4  });
5
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_4.js

Fortran
Lisp
COBOL
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_5.js — javascriptmasterclass

JS iterables_iterators_5.js

```
1   const languages = ["Fortran", "Lisp", "COBOL"];
2   for (let language of languages) {
3       console.log(language);
4   }
5
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_5.js

Fortran
Lisp
COBOL
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_6.js — javascriptmasterclass

JS iterables_iterators_6.js

```
1   const classicLanguages = ["Fortran", "Lisp", "COBOL"];
2   const modernLanguages = ["Python", "Ruby", "JavaScript"]
3   const languages = classicLanguages.concat(modernLanguages);
4   console.log(languages);
5
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_6.js

[ 'Fortran', 'Lisp', 'COBOL', 'Python', 'Rub
y', 'JavaScript' ]
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_7.js — javascriptmasterclass

JS iterables_iterators_7.js

```
1  const classicLanguages = ["Fortran", "Lisp", "COBOL"];
2  const modernLanguages = ["Python", "Ruby", "JavaScript"]
3  const languages = [...classicLanguages, ...modernLanguages];
4  console.log(languages);
5
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_7.js

[ 'Fortran', 'Lisp', 'COBOL', 'Python', 'Rub
y', 'JavaScript' ]
rodrigobranas:javascriptmasterclass $
```

Além do Array é possível utilizar o protocolo de iteração dos objetos Map, Set e String

iterables_iterators_8.js — javascriptmasterclass

JS iterables_iterators_8.js

```
1  const languages = new Map([["Fortran", 1957], ["Lisp", 1958]
2  for (let language of languages) {
3      console.log(language);
4  }
5
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_8.js

[ 'Fortran', 1957 ]
[ 'Lisp', 1958 ]
[ 'COBOL', 1959 ]
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_9.js — javascriptmasterclass

JS iterables_iterators_9.js

```
1  const languages = new Map([["Fortran", 1957], ["Lisp", 1958]
2  for (let [language, year] of languages) {
3      console.log(language, year);
4  }
5
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_9.js

Fortran 1957
Lisp 1958
COBOL 1959
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_10.js — javascriptmasterclass

JS iterables_iterators_10.js

```
1   const languages = new Map([["Fortran", 1957], ["Lisp", 1958]
2   console.log([...languages]);
3
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_10.js

[ [ 'Fortran', 1957 ], [ 'Lisp', 1958 ], [ '
COBOL', 1959 ] ]
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_11.js — javascriptmasterclass

JS iterables_iterators_11.js

```
1  let languages = new Set(["Fortran", "Lisp", "COBOL"]);
2  for (let language of languages) {
3      console.log(language);
4  }
5
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_11.js

Fortran
Lisp
COBOL
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_12.js — javascriptmasterclass

JS iterables_iterators_12.js

```
1  let languages = new Set(["Fortran", "Lisp", "COBOL"]);
2  console.log([...languages]);
3
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_12.js

[ 'Fortran', 'Lisp', 'COBOL' ]
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_13.js — javascriptmasterclass

JS iterables_iterators_13.js

```
1  let language = "COBOL";
2  for (let char of language) {
3      console.log(char);
4  }
5
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_13.js

C
O
B
O
L
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_14.js — javascriptmasterclass

JS iterables_iterators_14.js

```
1  let language = "COBOL";
2  console.log([...language]);
3
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_14.js

[ 'C', 'O', 'B', 'O', 'L' ]
rodrigobranas:javascriptmasterclass $
```

Todo Iterable tem um propriedade de chave Symbol.iterator que define o protocolo de iteração para o objeto

iterables_iterators_15.js — javascriptmasterclass

JS iterables_iterators_15.js

```
1  const languages = ["Fortran", "Lisp", "COBOL"];
2  const iterator = languages[Symbol.iterator]();
3  console.log(iterator.next());
4  console.log(iterator.next());
5  console.log(iterator.next());
6  console.log(iterator.next());
7
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_15.js

{ value: 'Fortran', done: false }
{ value: 'Lisp', done: false }
{ value: 'COBOL', done: false }
{ value: undefined, done: true }
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_16.js — javascriptmasterclass

JS iterables_iterators_16.js

```
1  const languages = new Map([["Fortran", 1957], ["Lisp", 1958]
2  const iterator = languages[Symbol.iterator]();
3  console.log(iterator.next());
4  console.log(iterator.next());
5  console.log(iterator.next());
6  console.log(iterator.next());
7
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_16.js

{ value: [ 'Fortran', 1957 ], done: false }
{ value: [ 'Lisp', 1958 ], done: false }
{ value: [ 'COBOL', 1959 ], done: false }
{ value: undefined, done: true }
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_17.js — javascriptmasterclass

JS iterables_iterators_17.js

```
1  const languages = new Map([["Fortran", 1957], ["Lisp", 1958]
2  const iterator = languages.entries();
3  console.log(iterator.next());
4  console.log(iterator.next());
5  console.log(iterator.next());
6  console.log(iterator.next());
7
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_17.js

{ value: [ 'Fortran', 1957 ], done: false }
{ value: [ 'Lisp', 1958 ], done: false }
{ value: [ 'COBOL', 1959 ], done: false }
{ value: undefined, done: true }
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_18.js — javascriptmasterclass

JS iterables_iterators_18.js

```
1  const languages = new Map([["Fortran", 1957], ["Lisp", 1958]
2  const iterator = languages.keys();
3  console.log(iterator.next());
4  console.log(iterator.next());
5  console.log(iterator.next());
6  console.log(iterator.next());
7
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_18.js

{ value: 'Fortran', done: false }
{ value: 'Lisp', done: false }
{ value: 'COBOL', done: false }
{ value: undefined, done: true }
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_19.js — javascriptmasterclass

JS iterables_iterators_19.js

```
1   const languages = new Map([["Fortran", 1957], ["Lisp", 1958]
2   const iterator = languages.values();
3   console.log(iterator.next());
4   console.log(iterator.next());
5   console.log(iterator.next());
6   console.log(iterator.next());
7
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_19.js

{ value: 1957, done: false }
{ value: 1958, done: false }
{ value: 1959, done: false }
{ value: undefined, done: true }
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_20.js — javascriptmasterclass

JS iterables_iterators_20.js

```
1   const languages = new Set(["Fortran", "Lisp", "COBOL"]);
2   const iterator = languages[Symbol.iterator]();
3   console.log(iterator.next());
4   console.log(iterator.next());
5   console.log(iterator.next());
6   console.log(iterator.next());
7
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_20.js

{ value: 'Fortran', done: false }
{ value: 'Lisp', done: false }
{ value: 'COBOL', done: false }
{ value: undefined, done: true }
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_21.js — javascriptmasterclass

JS iterables_iterators_21.js

```
1  const languages = new Set(["Fortran", "Lisp", "COBOL"]);
2  const iterator = languages.entries();
3  console.log(iterator.next());
4  console.log(iterator.next());
5  console.log(iterator.next());
6  console.log(iterator.next());
7
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_21.js

{ value: [ 'Fortran', 'Fortran' ], done: fal
se }
{ value: [ 'Lisp', 'Lisp' ], done: false }
{ value: [ 'COBOL', 'COBOL' ], done: false }
{ value: undefined, done: true }
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_22.js — javascriptmasterclass

JS iterables_iterators_22.js

```
1  const languages = new Set(["Fortran", "Lisp", "COBOL"]);
2  const iterator = languages.keys();
3  console.log(iterator.next());
4  console.log(iterator.next());
5  console.log(iterator.next());
6  console.log(iterator.next());
7
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_22.js

{ value: 'Fortran', done: false }
{ value: 'Lisp', done: false }
{ value: 'COBOL', done: false }
{ value: undefined, done: true }
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_23.js — javascriptmasterclass

JS iterables_iterators_23.js

```
1  const languages = new Set(["Fortran", "Lisp", "COBOL"]);
2  const iterator = languages.values();
3  console.log(iterator.next());
4  console.log(iterator.next());
5  console.log(iterator.next());
6  console.log(iterator.next());
7
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_23.js

{ value: 'Fortran', done: false }
{ value: 'Lisp', done: false }
{ value: 'COBOL', done: false }
{ value: undefined, done: true }
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_24.js — javascriptmasterclass

JS iterables_iterators_24.js

```
1  const language = "COBOL";
2  const iterator = language[Symbol.iterator]();
3  console.log(iterator.next());
4  console.log(iterator.next());
5  console.log(iterator.next());
6  console.log(iterator.next());
7  console.log(iterator.next());
8  console.log(iterator.next());
9
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_24.js

{ value: 'C', done: false }
{ value: 'O', done: false }
{ value: 'B', done: false }
{ value: 'O', done: false }
{ value: 'L', done: false }
{ value: undefined, done: true }
rodrigobranas:javascriptmasterclass $
```

Como criar um Iterable?

iterables_iterators_25.js — javascriptmasterclass

JS iterables_iterators_25.js

```
function createIterator(...array) {
    let i = 0;
    return {
        next() {
            if (i < array.length) {
                return {
                    value: array[i++],
                    done: false
                }
            } else {
                return {
                    value: undefined,
                    done: true
                }
            }
        }
    }
}
const iterator = createIterator("Fortran", "Lisp", "COBOL");
console.log(iterator.next());
console.log(iterator.next());
console.log(iterator.next());
console.log(iterator.next());
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_25.js

{ value: 'Fortran', done: false }
{ value: 'Lisp', done: false }
{ value: 'COBOL', done: false }
{ value: undefined, done: true }
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_26.js — javascriptmasterclass

JS iterables_iterators_26.js

```
1   function createIterator(...array) {
2       let i = 0;
3       return {
4           next() {
5               if (i < array.length) {
6                   return {
7                       value: array[i++],
8                       done: false
9                   }
10              } else {
11                  return {
12                      value: undefined,
13                      done: true
14                  }
15              }
16          }
17      }
18  }
19  const iterator = createIterator("Fortran", "Lisp", "COBOL");
20  for (let language of iterator) {
21      console.log(language);
22  }
23
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_26.js

/Users/rodrigobranas/development/workspace/j
avascriptmasterclass/iterables_iterators/ite
rables_iterators_26.js:20
for (let language of iterator) {
                     ^


TypeError: iterator is not iterable
    at Object.<anonymous> (/Users/rodrigobra
nas/development/workspace/javascriptmastercl
ass/iterables_iterators/iterables_iterators_
26.js:20:22)
    at Module._compile (module.js:643:30)
    at Object.Module._extensions..js (module
.js:654:10)
    at Module.load (module.js:556:32)
    at tryModuleLoad (module.js:499:12)
    at Function.Module._load (module.js:491:
3)
    at Function.Module.runMain (module.js:68
4:10)
    at startup (bootstrap_node.js:187:16)
    at bootstrap_node.js:608:3
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_27.js — javascriptmasterclass

JS iterables_iterators_27.js

```
1   function createIterator(...array) {
2       let i = 0;
3       return {
4           next() {
5               if (i < array.length) {
6                   return {
7                       value: array[i++],
8                       done: false
9                   }
10              } else {
11                  return {
12                      value: undefined,
13                      done: true
14                  }
15              }
16          }
17      }
18  }
19  const iterator = createIterator("Fortran", "Lisp", "COBOL");
20  console.log([...iterator]);
21
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_27.js

/Users/rodrigobranas/development/workspace/j
avascriptmasterclass/iterables_iterators/ite
rables_iterators_27.js:20
console.log([...iterator]);
                ^


TypeError: iterator is not iterable
    at Object.<anonymous> (/Users/rodrigobra
nas/development/workspace/javascriptmastercl
ass/iterables_iterators/iterables_iterators_
27.js:20:17)
    at Module._compile (module.js:643:30)
    at Object.Module._extensions..js (module
.js:654:10)
    at Module.load (module.js:556:32)
    at tryModuleLoad (module.js:499:12)
    at Function.Module._load (module.js:491:
3)
    at Function.Module.runMain (module.js:68
4:10)
    at startup (bootstrap_node.js:187:16)
    at bootstrap_node.js:608:3
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_28.js — javascriptmasterclass

JS iterables_iterators_28.js

```
function createIterable(...array) {
    return {
        [Symbol.iterator]() {
            let i = 0;
            return {
                next() {
                    if (i < array.length) {
                        return {
                            value: array[i++],
                            done: false
                        }
                    } else {
                        return {
                            value: undefined,
                            done: true
                        }
                    }
                }
            }
        }
    }
}
const iterable = createIterable("Fortran", "Lisp", "COBOL");
for (let language of iterable) {
    console.log(language);
}
```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_28.js

Fortran
Lisp
COBOL
rodrigobranas:javascriptmasterclass $
```

iterables_iterators_29.js — javascriptmasterclass

JS iterables_iterators_29.js

```
function createIterable(...array) {
    return {
        [Symbol.iterator]() {
            let i = 0;
            return {
                next() {
                    if (i < array.length) {
                        return {
                            value: array[i++],
                            done: false
                        }
                    } else {
                        return {
                            value: undefined,
                            done: true
                        }
                    }
                }
            }
        }
    }
}
const iterable = createIterable("Fortran", "Lisp", "COBOL");
console.log([...iterable]);

```

TERMINAL 1: bash

```
rodrigobranas:javascriptmasterclass $ node i
terables_iterators/iterables_iterators_29.js

[ 'Fortran', 'Lisp', 'COBOL' ]
rodrigobranas:javascriptmasterclass $
```