

Looping and Branching in JavaScript

Understanding Iteration Thoroughly



Andrejs Doronins

Software Developer in Test

Looping and Branching in JavaScript

Version Check

Version Check



This course was created by using:

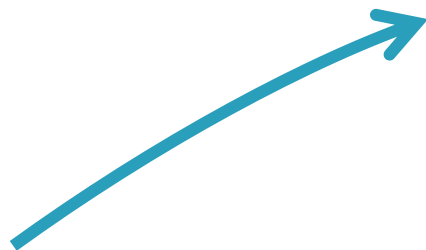
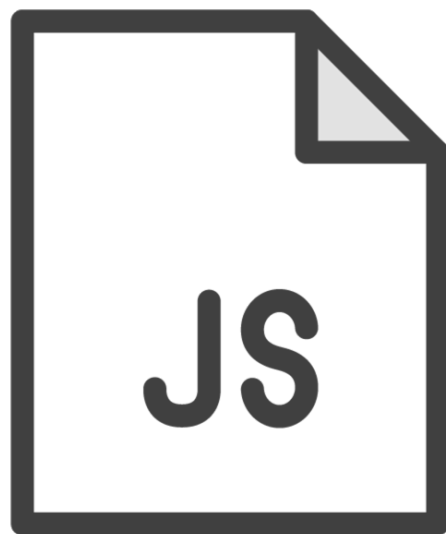
- ECMAScript 2022

Version Check



This course is 100% applicable to:

- ECMAScript 2022



Prerequisites



General programming concepts

- **Data types**
- **Functions or methods**

HTML basics

Code Editor

Setup



OS: Windows

IDE: VS Code

Node.js

Course Overview



Looping over iterables

- for, for-of, forEach(), while, do-while

Advanced looping concepts

- for-in, for-await-of, performance

Branching

- Nullish coalescing operator - ??

[Table of contents](#)

[Description](#)

[Transcript](#)

[Exercise files](#)



conservative

<

1

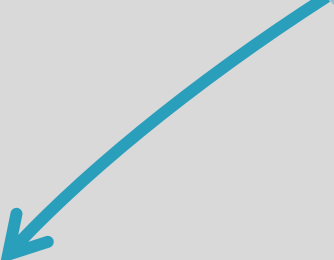
<



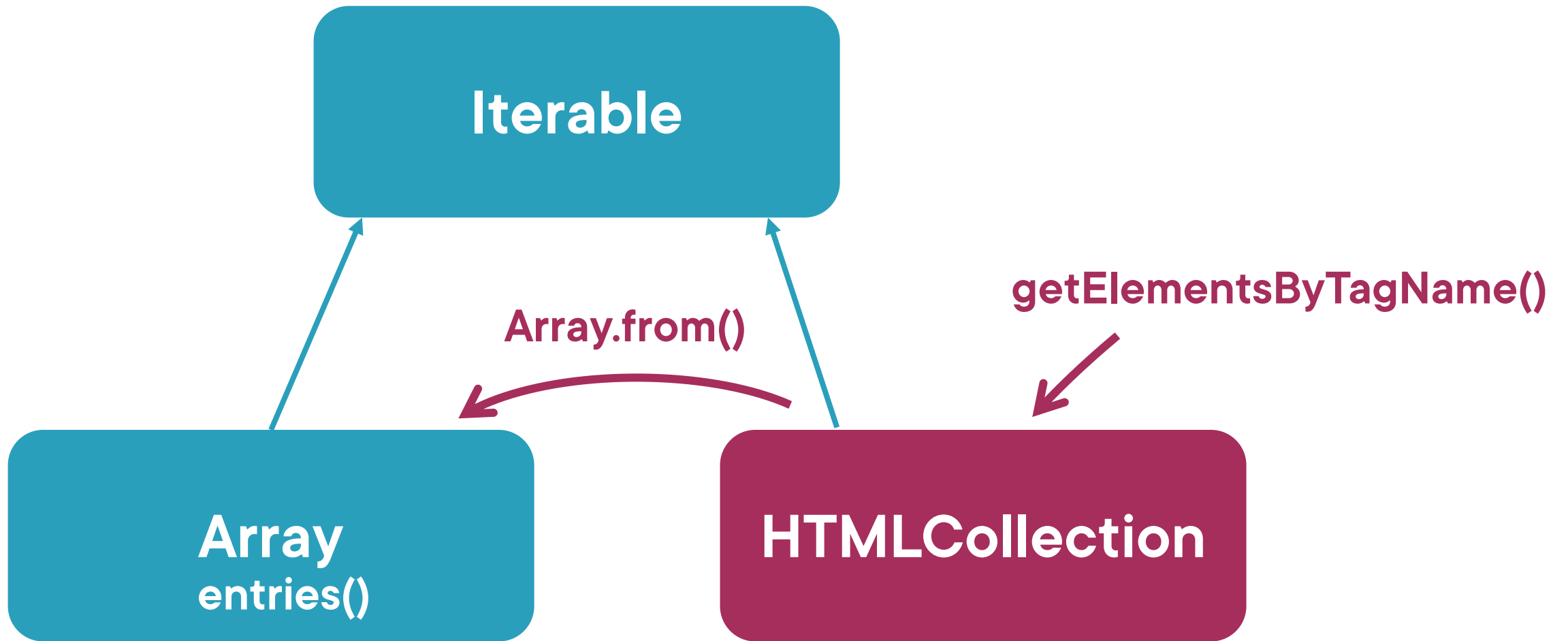
riskier

```
for («init»; «condition»; «post_iter») {  
    «statements»  
}
```

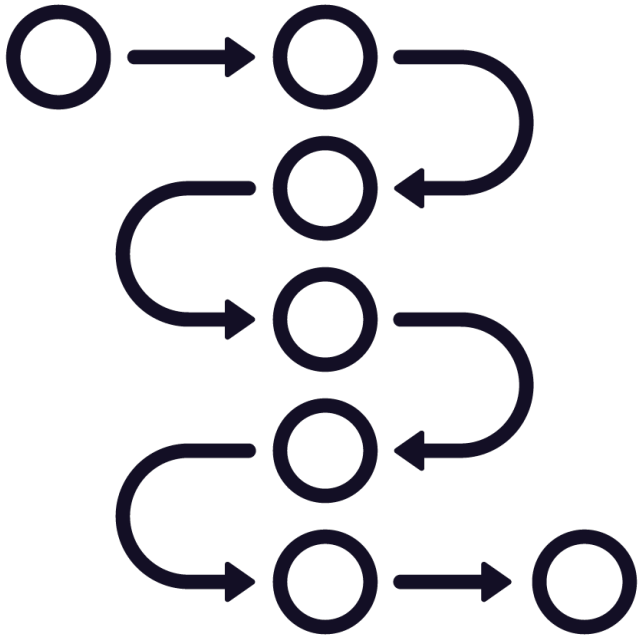
```
let strings = ["a", "b", "c"]; undefined
```



```
for (let i = 0; i <= strings.length; i++) {  
    console.log("Element num " + strings[i]);  
}
```



Iterables



Array

HTMLCollection


String

Sets (unique values only)

Maps (key-value pairs)



```
for (let value of array) { ... }
```



```
array.forEach(e => ...);
```

Sparse Arrays

```
// arrays may have null, undefined or empty slots  
let array = [,,,];  
array.length = 20;  
array[30] = 100;  
delete array[5];
```


for(init;evaluate;update) {

next iteration

continue;

// OR

break;

}

leave



Array Methods

`find();`

`every();`

`some();`

Summary



Different loops



Up Next:

Grasping Advanced Iteration Concepts
