ECMAScript 2020

Key	Value
Name	ECMAScript 2020
Short name	ES2020
Was finalized in	June 2020

Feature list

#	Name	Desc
1	BigInt	New primitive type. Big integer can represent number larger than 2^53 - 1
2	Dynamic Import	"function-like" import() module loading syntactic
3	**Nullish Coalescing**	`x ?? y` means return y if x is nullish
4	**Optional Chaining**	access deep property without checking each ref in the chain is valid
5	Promise.allSettled	wait until all promises finished, either fulfilled or rejected
6	String.prototype.matchAll	leave it for now, get back later in regular expression section
7	**globalThis**	Get the global object of current environment.
8	**Module Namespace Exports**	export * as something from 'module'
9	Well defined for-in order	standardlize the enumeration order of for-in
10	import.meta	access meta information about the module

1. BigInt

```
Number.MAX_SAFE_INTEGER; // 2^53 - 1 = 9007199254740991
Number.MAX_SAFE_INTEGER + 2; // 9007199254740992 ??? incorrect calculation

const balance = BigInt('9007199254740991');
balance + 2n; // 9007199254740993n
```

```
> JSON.parse('{ "balance": 9007199254740993 }')
<- {balance: 9007199254740992}
```

```
JSON.parse('{ "balance": 9007199254740993 }'); // { balance:
9007199254740992 }

// correct way to handle it
const json = '{ "balance": "9007199254740993" }';
const { balance } = JSON.parse(json);
BigInt(balance); // 9007199254740993n
```

> Tham khảo: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/BigInt

2. Dynamic Import

"function-like" import() module loading syntactic. source

```
// math.js
const add = (num1, num2) => num1 + num2;

export { add };
```

```
// index.js
const doMath = async (num1, num2) => {
  if (num1 && num2) {
    const math = await import('./math.js');
    console.log(math.add(5, 10));
  };
};

doMath(4, 2);
```

> Tham khảo: https://www.digitalocean.com/community/tutorials/js-es2020

3. Nullish Coalescing

```
const value = x ?? y;
// - return y if x is nullish (null or undefined)
// - otherwise return x
```

```
null ?? 'Easy Frontend'; // 'Easy Frontend'
undefined ?? 'Easy Frontend'; // 'Easy Frontend'

'' ?? 'Easy Frontend'; // ''
0 ?? 'Easy Frontend'; // 0
Number.NaN ?? 'Easy Frontend'; // NaN
false ?? 'Easy Frontend'; // false
```

```
null || 'Easy Frontend'; // 'Easy Frontend'
undefined || 'Easy Frontend'; // 'Easy Frontend'

'' || 'Easy Frontend'; // 'Easy Frontend'
0 || 'Easy Frontend'; // 'Easy Frontend'
Number.NaN || 'Easy Frontend'; // 'Easy Frontend'
false || 'Easy Frontend'; // 'Easy Frontend'
```

> Tham khảo: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Nullish_coalescing_operator

4. Optional Chaining

- it returns **undefined** if some ref in the chain is nullish (null or undefined)

```
obj?.prop       // optional static property access
obj?.[expr]     // optional dynamic property access

arr?.[index]    // optional array item access
func?.(args)    // optional function or method call
```

```
const student = {
  id: 1,
  address: {
    city: 'HCMC',
  },
  hobbies: ['music'],
  sayHello() {
    console.log('Hello');
  },
}

student.address.city; // 'HCMC'
student.hobbies[0]; // 'music'
student.sayHello(); // 'Hello'

// but what if you're trying to access non-existing props in the chain
student.profile.name;
// TypeError: Cannot read property 'name' of undefined

student.colors[0];
// TypeError: Cannot read property '0' of undefined

student.learnJavascript();
// TypeError: student.learnJavascript is not a function
```

```
if (student.profile) {
  console.log(student.profile.name);
}
```

```
// Optional chaining
student.profile?.name; // undefined
student.colors?.[0]; // undefined
student.learnJavascript?.(); // undefined
```

5. Promise.allSettled

```
const p1 = new Promise((res, rej) => setTimeout(res, 1000));

const p2 = new Promise((res, rej) => setTimeout(rej, 1000));

Promise.allSettled([p1, p2]).then(data => console.log(data));
```

```
// Output
[
   Object { status: "fulfilled", value: undefined},
   Object { status: "rejected", reason: undefined}
]
```

> Tham khảo: https://www.digitalocean.com/community/tutorials/js-es2020

6. String.prototype.matchAll

> Tham khảo: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/matchAll

7. globalThis

Get the global object of current environment.

Environment	globalThis value
browser	window
nodejs	global
web workers	self

```
// before ES2020
var getGlobal = function () {
  if (typeof self !== 'undefined') { return self; }
  if (typeof window !== 'undefined') { return window; }
  if (typeof global !== 'undefined') { return global; }
  throw new Error('unable to locate global object');
};

var globals = getGlobal();

if (typeof globals.setTimeout !== 'function') {
  // no setTimeout in this environment!
}
```

```
// when working with server side rendering,
// you may see this code block to check whether it's client or server
if (typeof window !== 'undefined') {
  // it's on client side
} else {
  // it's on server side
}
```

8. Module Namespace Exports

```
// before ES2020
import * as utils from './utils.mjs'
export { utils }

// ES2020
export * as utils from './utils.mjs'
```

9. Well defined for-in order

The ECMA specification did not specify in which order for (x in y) should run. Even though browsers implemented a consistent order on their own before now, this has been officially standardized in ES2020.

> Tham khảo: https://www.freecodecamp.org/news/javascript-new-features-es2020/

10. import.meta

You can access meta information about the module using the import.meta object:

```
console.log(import.meta); // { url: "file:///home/user/module.js" }
```

> Tham khảo: https://www.freecodecamp.org/news/javascript-new-features-es2020/

Tham khảo

- https://h3manth.com/ES2020/
- https://www.freecodecamp.org/news/javascript-new-features-es2020/
- https://www.digitalocean.com/community/tutorials/js-es2020

Khoá học Javascript cho người mới bắt đầu 2021 🎉

- Tác giả: **Hậu Nguyễn** - Founder Easy Frontend
- Khoá học chỉ được published trên Udemy, không thông qua trung gian.
- Khoá học không bán dạng videos upload trên Google Drive hay bất cứ hình thức nào tương tự.
- Khoá học có nhóm discord để hỗ trợ trong quá trình học tập.

☎️ Liên hệ tác giả để được hỗ trợ:

- ✅ Facebook: https://www.facebook.com/nvhauesmn/
- ✅ Fanpage: https://www.facebook.com/learn.easyfrontend
- ✅ Youtube Channel: https://www.youtube.com/easyfrontend