



ES6+



План

- оголошення змінних
- деструктуризація
- строки
- стрілочні функції
- об'єкти і прототипи
- класи
- promise

let

- для оголошення змінних
- область видимості змінної let - блок {...}
- змінна let видно тільки після оголошення
- при використанні в циклі, для кожної ітерації створюється своя змінна

```
let apples = 5; // (*)
if (true) {
    let apples = 10;

    alert(apples); // 10 (в середині блока)
}
alert(apples); // 5 (зовні блока значення не змінилось)
```

```
alert(a); // помилка, немає такої змінної
let a = 5;
```

const

- Оголошення `const` задає константу, тобто змінну, яку не можна змінювати

```
const apple = 5;  
apple = 10; // помилка
```

<https://learn.javascript.ru/let-const>

Деструктуризація

- особливий синтаксис присвоювання, при якому можна присвоїти масив або об'єкт відразу декільком змінним, розбивши його на частини

```
let [firstName, lastName] = ["Іван", "Пупкін"];  
alert(firstName); // Іван  
alert(lastName);  // Пупкін
```

```
let user = {  
  name: "Петро",  
  age: 25,  
};  
let {name, age} = user;  
alert(name);    // Петро  
alert(age);     // 25
```

- Якщо ми хочемо отримати і наступні значення масиву, але не впевнені в їх числі - можна додати ще один параметр, який отримає «все інше», за допомогою оператора "..." («spread», три крапки):

```
let [firstName, lastName, ...info] = ["Іван", "Пупкін", "front-end", "junior"];  
alert(firstName); // Іван  
alert(lastName);  // Пупкін  
alert(info);      // ["front-end", "junior"]
```

<https://learn.javascript.ru/destructuring>

Оператори Spread і Rest

- оператор ... інтерпретується по-різному, в залежності від контексту застосування. **Spread** використовується для поділу колекцій на окремі елементи
- розділяє масив на значення

```
var log = function(a, b, c) {
```

```
  console.log(a, b, c);
```

```
};
```

```
log(...['Spread', 'Rest', 'Operator']); // Spread Rest Operator
```


- КОПІЮВАННЯ ВЛАСТИВОСТЕЙ МАСИВІВ

```
var arr = ['will', 'love'];  
var data = ['You', ...arr, 'spread', 'operator'];  
console.log(data); // ['You', 'will', 'love', 'spread', 'operator']
```

- КОПІЮВАННЯ ВСІХ ВЛАСТИВОСТЕЙ, А НЕ ПОСИЛАННЯ НА МАСИВ

```
var arr = [1, 2, 3, 4, 5];  
var data = [...arr];  
var copy = arr;
```

```
arr === data; // false - посилання відрізняються - два різних масиву  
arr === copy; // true - дві змінні посилаються на один масив
```

- **rest**, навпаки, для з'єднання окремих значень в масив

```
var log = function(a, b, ...rest) {  
  console.log(a, b, rest);  
};  
log('Basic', 'rest', 'operator', 'usage'); // Basic rest ['operator', usage]
```

зberi всі залишилися елементи в масив з ім'ям rest

Функції

- параметри по замовчуванні

```
function calcPrice(price, sale = 20) {  
    return price - price * sale /100;  
}
```

```
calcPrice(500, 40); // 500-500*40/100
```

```
calcPrice(500); // 500-500*20/100
```

<https://learn.javascript.ru/es-function>

Стрілочні функції

```
let inc = x => x+1;
```

```
let inc = function(x) { return x + 1; };
```

```
let sum = (a,b) => a + b;
```

```
// аналог function
```

```
// let sum = function(a, b) { return a + b; };
```

```
alert( sum(1, 2) ); // 3
```

- Усередині функцій-стрілок - той же this, що і зовні.

```
let group = {  
  title: "Наш курс",  
  students: ["Вася", "Петя", "Даша"],  
  
  showList: function() {  
    this.students.forEach(  
      student => alert(this.title + ': ' + student)  
    )  
  }  
}
```

```
group.showList();  
// Наш курс: Вася  
// Наш курс: Петя  
// Наш курс: Даша
```

Практика

- написати стрілочну функцію яка буде повертати остачу від ділення
функція приймає два параметри, другий з яких по замовчуванні дорівнює
2

Строки

- Можна вставляти вирази/змінні за допомогою ``$ {...}``

```
let [firstName, lastName] = ["Іван", "Пупкін"];  
console.log(`Мое ім'я ${firstName} ${lastName}`);
```

```
console.log(`Мое ім\`я` + firstName + ` ` + lastName);
```

<https://learn.javascript.ru/es-string>

Map

Map - колекція для зберігання записів виду ключ: значення.

На відміну від об'єктів, в яких ключами можуть бути тільки рядки, в Map ключем може бути довільне значення, наприклад:

```
let map = new Map();  
  
map.set(1, 'num1');      // число  
  
alert( map.get(1) ); // 'num1'  
alert( map.size ); // 3
```

<https://learn.javascript.ru/set-map>

`new Map ()` - створює колекцію.

`map.set (key, value)` - записує по ключу `key` значення `value`.

`map.get (key)` - повертає значення по ключу або `undefined`, якщо ключ `key` відсутня.

`map.has (key)` - повертає `true`, якщо ключ `key` присутній в колекції, інакше `false`.

`map.delete (key)` - видаляє елемент по ключу `key`.

`map.clear ()` - очищає колекцію від всіх елементів.

`map.size` - повертає поточну кількість елементів.

```
let recipeMap = new Map ([  
  ["Огірок", 500],  
  ["Помідор", 350],  
  ["Цибуля", 50]  
]);
```

```
// перебір по ключам (овочі)  
for (let vegetable of recipeMap.keys ()) {  
  alert (vegetable); // огірок, помідор, цибуля  
}  
  
// перебір за значеннями (числа)  
for (let amount of recipeMap.values ()) {  
  alert (amount); // 500, 350, 50  
}  
  
// перебір за елементами в форматі [ключ, значення]  
for (let entry of recipeMap) { // те ж саме, що і recipeMap.entries ()  
  alert (entry); // огірок, 500 (і так далі)  
}
```

Set

Set - колекція для зберігання безлічі значень, причому кожне значення може зустрічатися лише один раз.

<https://learn.javascript.ru/set-map#set>

тип Symbol

- це унікальний і незмінний тип даних. Метою Symbol є створення унікального ідентифікатора, до якого не можна отримати доступ

```
let user = {  
  name: "Вася"  
};
```

```
let id = Symbol("id");
```

```
user[id] = 1;
```

```
alert( user[id] ); // ми можемо отримати доступ до даних по ключу-символу
```

Symbol ("id") vs рядок "id"

Так як об'єкт user належить сторонньому коду, і цей код також працює з ним, то нам не слід додавати до нього будь-які поля. Це не безпечно. Але до символу складно ненавмисно звернутися, сторонній код навряд чи його взагалі побачить, і, швидше за все, додавання поля до об'єкту не викличе жодних проблем.

Крім того, припустимо, що інший скрипт для якихось своїх цілей хоче записати власний ідентифікатор в об'єкт user. Цей скрипт може бути якийсь JavaScript-бібліотекою, абсолютно не пов'язаної з нашим скриптом.

Нові функції

- `Object.fromEntries()` - приймає об'єкт в якості аргументу і повертає масив пар властивостей строкового ключа об'єкта у вигляді `[key, value]`

```
const obj = {one: 1, two: 2, three: 3};  
console.log(Object.entries(obj));  
// => [ ["one", 1], ["two", 2], ["three", 3] ]
```

```
const myArray = [['one', 1], ['two', 2], ['three', 3]];  
const obj = Object.fromEntries(myArray);  
console.log(obj);    // => {one: 1, two: 2, three: 3}
```


- `flat()` дозволяє вам легко об'єднувати всі елементи підмасива в масив.

```
const arr = ['a', 'b', ['c', 'd']];  
const flattened = arr.flat();  
console.log(flattened);    // => ["a", "b", "c", "d"]
```

- `flatMap()` об'єднує `map()` і `flat()` в один метод

```
const arr = [4.25, 19.99, 25.5];  
console.log(arr.map(value => [Math.round(value)]));  
// => [[4], [20], [26]]  
console.log(arr.flatMap(value => [Math.round(value)]));  
// => [4, 20, 26]
```

- BigInt це вбудований об'єкт, який надає спосіб представляти цілі числа більше 2^{53} , найбільшого числа, яке JavaScript може надійно уявити з Number примітивом

```
const theBiggestInt = 9007199254740991n;  
const alsoHuge = BigInt(9007199254740991);  
//  9007199254740991n
```

```
typeof 1n === 'bigint'; // true  
typeof BigInt('1') === 'bigint'; // true
```


Посилання

<https://habr.com/post/305900/#4>

<https://learn.javascript.ru/es-modern>

<https://habr.com/company/ruvds/blog/353174/>

<https://habr.com/ru/users/alexzfort/posts/>

[tps://webdevblog.ru/5-funkcij-es2019-kotorye-vy-mozhete-ispolzovat-segodnya/](https://webdevblog.ru/5-funkcij-es2019-kotorye-vy-mozhete-ispolzovat-segodnya/)

<https://medium.com/web-standards/es2019-417d8b406346>

<https://blog.logrocket.com/5-es2019-features-you-can-use-today/>

Відео

<https://www.youtube.com/watch?v=4YfsAz-sNAo&list=PLqHIAwsJRxAOpWPtj2T6HhSzX-IKmKV2q>