

JavaScript

NEXT



План

Оператори

Вивід даних/ Ввід даних

Вирішення логічних задач та побудова алгоритмів

Умови if/else

switch/case



JavaScript

Оператори

Оператор typeof

Для отримання типу значення змінної використовується оператор типу, який повертає на місце свого виклику тип значення змінної вказаного після нього. Значення, що повертається, це просто рядок в якому вказаний тип.

```
const b = null;  
console.log(typeof null); // "object"
```

```
const c = 5;  
console.log(typeof c); // "number"
```

```
const d = 'JavaScript is awesome!';  
console.log(typeof d); // "string"
```

```
const e = false;  
console.log(typeof e); // "boolean"
```

Вивід даних/ Ввід даних

Взаємодія: alert, prompt, confirm

Оскільки основним середовищем для демонстрації можливостей JavaScript буде браузер, давайте розглянемо декілька функцій для взаємодії з користувачем: alert, prompt та confirm.

alert

Функція показує повідомлення та чекає, доки користувач не натисне кнопку “ОК”.
`alert("Привіт");`

prompt

Функція prompt показує модальне вікно з текстовим повідомленням, полем, куди відвідувач може ввести текст, та кнопками ОК/Скасувати.

```
let age = prompt('Скільки вам років?');
```

confirm

```
result = confirm(question);
```

Функція `confirm` показує модальне вікно з питанням `question` та двома кнопками: ОК та Скасувати.

Результат: `true`, якщо натиснути кнопку ОК, інакше — `false`.

Для виведення даних будемо використовувати два методи: `console.log()` і `alert()`. `console` та `alert` є частиною інтерфейсу `window` – глобального об'єкта, доступного при виконанні скрипту на веб-сторінці. Запис `window.alert()` надлишковий, пишемо просто `alert()` або `console.log()`.

```
console.log("Hello").
```

Вирішення логічних задач та побудова алгоритмів

Математичні оператори

Нічим не відрізняються від шкільного курсу математики. Оператори повертають значення місце операції. Порядок обчислення математичних виразів тощо це всім звична алгебра.

Важливо запам'ятати правильне найменування складових висловлювання. $+$ $-$ $*$ $/$ $\%$ називаються операторами, бо на чому вони застосовуються операндами.

```
const x = 10;  
const y = 5;
```

```
console.log(x + y); // 15  
console.log(x - y); // 5  
console.log(x * y); // 50
```

```
console.log(x / y); // 2  
console.log(x % y); // 0
```

Оператори порівняння

Використовуються для порівняння значень. Результатом свого виконання повертають булі, true або false.

$a > b$ та $a < b$ - більше/менше

```
console.log('5 > 10:', 5 > 10); // false
```

$a \geq b$ і $a \leq b$ - більше/менше чи одно

```
console.log('5 < 10', 5 < 10); // true
```

$a == b$ - рівність

$a != b$ - нерівність

$a === b$ - сувора рівність

$a !== b$ - сувора нерівність

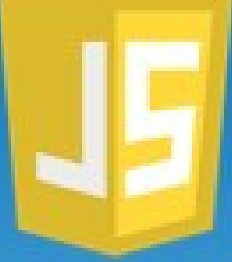
$==$ або $===$

Завжди використовуйте строгу рівність $===$ і строгу нерівність $!==$.

Оператори $==$ і $!=$ виконують перетворення типів порівнюваних значень, що може призвести до помилок, особливо у початківців.

Логічні оператори

В JavaScript існує чотири логічні оператори: || (АБО), && (І), ! (НЕ), ?? (оператор null-об'єднання).



Logical Operators in JavaScript

| Operator | Name |
|----------|-------------|
| (!) | Logical NOT |
| (&&) | Logical AND |
| () | Logical OR |
| | |

|| (АБО)

Оператор “АБО” представлений двома символами вертикальної лінії:

```
result = a || b;
```

Логічний оператор АБО призначений для маніпулювання лише булевими значеннями. Якщо будь-який з його аргументів означає true, повертається true, інакше повертається false.

&& (І)

Оператор І представлений двома амперсандами &&:

```
result = a && b;
```

Повертає true, якщо обидва оператори є правдивими, і false в іншому випадку

! (НЕ)

Булевий оператор НЕ представлений знаком оклику !.

```
result = !value;
```

Оператор приймає один аргумент і виконує наступне:

Перетворює операнд на булевий тип: true/false.

Повертає зворотне значення.

```
alert( !true ); // false
```

```
alert( !0 ); // true
```

Подвійний НЕ !! іноді використовується для перетворення значення на булевий тип:

```
alert( !! "не пустий рядок" ); // true
```

```
alert( !!null ); // false
```

Тобто, перший НЕ перетворює значення на булеве і повертає зворотне, а другий НЕ інвертує його знову.

Умовні розгалуження: if- else

Інструкція “if”

Інструкція if(...) обчислює умову в дужках і, якщо результат умови true, виконує блок коду.

```
if (5 * 5 === 25) {  
  console.log("Виведе в консоль цей рядок")}
```

Блок “else”

Вираз if може містити необов'язковий блок “else” (“інакше”). Він виконується, коли умова є хибною.

```
// if (5 * 5 === 25) {  
//   console.log("1");  
// } else {  
//   console.log("2");  
// }
```

Декілька умов: “else if”

Іноді ми хотіли б перевірити кілька варіантів умови. Блок else if дозволяє нам це зробити.

```
// if (5 + 5 === 10) {  
//   console.log("5+5 ===10");  
// } else if (2 + 2 === 4) {  
//   console.log("2+2 ===4");  
// } else {  
//   console.log("no");  
// }
```

Умовний оператор ‘?’

Так званий “умовний” оператор або оператор “знак питання” дає нам зробити це в більш короткій і простій формі.

Оператор представлений знаком питання ?. Іноді його називають “тернарним”, оскільки оператор має три операнди.

```
let result = умова ? значення1 : значення2;
```

```
/ 2 + 2 === 4 ? console.log("1") : console.log("2")
```

```
// 5 * 5 === 24 ? console.log(true) : console.log(false)
```

Конструкція "switch"

Конструкція switch може замінити кілька if.

Вона дає можливість більш наочного способу порівняння значення відразу з кількома варіантами.

Конструкція switch має один або більше case блоків та необов'язковий блок default.

```
switch(x) {  
  case 'value1': // if (x === 'value1')  
    ...  
    [break]  
  
  case 'value2': // if (x === 'value2')  
    ...  
    [break]  
  
  default:  
    ...  
    [break]  
}
```

Значення змінної `x` перевіряється на строгу рівність (`===`) значенню із першого блоку `case` (яке дорівнює `value1`), потім значенню із другого блоку (`value2`) і так далі.

Якщо строго рівне значення знайдено, то `switch` починає виконання коду із відповідного `case` до найближчого `break` або до кінця всієї конструкції `switch`. Якщо жодне `case`-значення не збігається – виконується код із блоку `default` (якщо він присутній).

Якщо `break` відсутній, то буде продовжено виконання коду по наступним блокам `case` без перевірок.

Додаткові матеріали

- <https://uk.javascript.info/comparison>
- <https://uk.javascript.info/ifelse>
- <https://uk.javascript.info/logical-operators>
- <https://uk.javascript.info/switch>
- <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators>
- <https://www.youtube.com/watch?v=UvI-AMAttrvE&feature=youtu.be&themeRefresh=1>
- <https://medium.com/@nicolasmarcora/mastering-javascripts-and-logical-operators-fd619b905c8f>

Домашнє завдання.

// Якщо змінна більше нуля - виведіть true, менше - false

//Перевірте це на варіантах 1, 0, -3.

// Якщо змінна ="test" - виведіть true,

//Перевірте це на варіантах 'test', "qwerty", true

// Якщо змінна більше 10 - відніміть 5,

//менше - додайте 5, результат виведіть в консоль

//Перевірте це на варіантах 1, 10, 13.

//Зробіть сервіс який отримує число від 1 до 12

// виведіть місяць який дорівнює числу

//Зробіть сервіс який отримує тризначне число

//Поверніть користувачу сумму цих чисел