## [Оператор нулевого слияния (??)](https://learn.javascript.ru/nullish-operators" \l "operator-nulevogo-sliyaniya)

Результат выражения a ?? b будет следующим:

* если a не  null или undefined, то a,
* если a null или undefined, то b.

Иначе говоря, оператор ?? возвращает первый аргумент, если он не null/undefined, иначе второй.

Вот как можно переписать выражение result = a ?? b, используя уже знакомые нам операторы:

result = (a !== null && a !== undefined) ? a : b;

Как правило, оператор ?? нужен для того, чтобы задать значение по умолчанию для потенциально неопределённой переменной.

Например, отобразим user, если её значение не null/undefined, в противном случае Аноним:

let user;

alert(user ?? "Аноним"); // Аноним (user не существует)

А вот пример, когда user присвоено значение:

let user = "Иван";

alert(user ?? "Аноним"); // Иван (user существует)

можно записать последовательность из операторов ??, чтобы получить первое значение из списка, которое не является null/undefined.

Допустим, у нас есть данные пользователя в переменных firstName, lastName или nickName. Все они могут не существовать, если пользователь решил не вводить соответствующие значение.

Мы хотели бы отобразить имя пользователя, используя одну из этих переменных, или показать «Аноним», если все они null/undefined.

Для этого воспользуемся оператором ??:

let firstName = null;

let lastName = null;

let nickName = "Суперкодер";

// показывает первое значение, которое определено:

alert(firstName ?? lastName ?? nickName ?? "Аноним"); // Суперкодер

### [Сравнение с ||](https://learn.javascript.ru/nullish-operators" \l "sravnenie-s)

Оператор || можно использовать для тех же целей

let firstName = null;

let lastName = null;

let nickName = "Суперкодер";

// показывает первое значение, которое определено:

alert(firstName ?? lastName ?? nickName ?? "Аноним"); // Суперкодер

// показывает первое истинное значение:

alert(firstName || lastName || nickName || "Аноним"); // Суперкодер

С другой стороны, сравнительно недавно в язык был добавлен оператор нулевого слияния ?? – как раз потому, что многие были недовольны оператором ||.

Важное различие между ними заключается в том, что:

* || возвращает первое *истинное* значение.
* ?? возвращает первое *определённое* значение.

оператор || не различает false, 0, пустую строку "" и null/undefined. Для него они все одинаковы, т.е. являются ложными значениями. Если первым аргументом для оператора || будет любое из перечисленных значений, то в качестве результата мы получим второй аргумент.

Однако на практике часто требуется использовать значение по умолчанию только тогда, когда переменная является null/undefined. Ведь именно тогда значение действительно неизвестно/не определено.

Рассмотрим следующий пример:

let height = 0;

alert(height || 100); // 100

alert(height ?? 100); // 0

* height || 100 проверяет height на ложное значение, оно равно 0, да, ложное.
  + поэтому результатом || является второй аргумент, т.е. 100.
* height ?? 100 проверяет, что переменная height содержит null/undefined, а поскольку это не так,
  + то результатом является сама переменная height, т.е. 0.

На практике нулевая высота часто является вполне нормальным значением, которое не следует заменять значением по умолчанию. Таким образом, ?? здесь как раз работает так, как нужно.

### [Приоритет](https://learn.javascript.ru/nullish-operators" \l "prioritet)

Приоритет оператора ?? такой же, как и у ||. Они оба равны 3 в [таблице на MDN](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Operator_Precedence#Table).

Это означает, что, как и ||, оператор нулевого слияния ?? вычисляется до = и ?, но после большинства других операций, таких как +, \*.

Так что, в выражениях такого вида понадобятся скобки:

let height = null;

let width = null;

// важно: используйте круглые скобки

let area = (height ?? 100) \* (width ?? 50);

alert(area); // 5000

### [Использование ?? вместе с && или ||](https://learn.javascript.ru/nullish-operators" \l "ispolzovanie-vmeste-s-ili)

По соображениям безопасности JavaScript запрещает использование оператора ?? вместе с && и ||, если приоритет явно не указан при помощи круглых скобок.

Выполнение следующего кода приведёт к синтаксической ошибке:

let x = 1 && 2 ?? 3; // Синтаксическая ошибка

Используйте скобки, чтобы обойти это ограничение:

let x = (1 && 2) ?? 3; // Работает без ошибок

alert(x); // 2

## [Оператор нулевого присваивания (??=)](https://learn.javascript.ru/nullish-operators" \l "operator-nulevogo-prisvaivaniya)

Предположим, нам необходимо проверить, равна ли переменная null или undefined, и если это так — присвоить этой переменной какое-либо другое значение.

Вот как мы сделали бы это сейчас:

let userAge = null;

if (userAge === null || userAge === undefined) {

userAge = 18;

}

Выглядит громоздко, правда? Существует оператор, более подходящий для подобных задач. Вот его синтаксис:

x ??= y

Оператор ??= присвоит x значение y только в том случае, если x не определено (null/undefined).

Теперь попробуем переписать уже знакомый нам фрагмент кода используя новый оператор:

let userAge = null;

userAge ??= 18;

alert(userAge) // 18

Обратите внимание: если бы userAge не был равен null/undefined, то выражение справа от ??= никогда бы не выполнилось:

let userAge = 18;

userAge ??= alert("не сработает");

userAge ??= 21;

userAge ??= null;

alert(userAge) // по-прежнему 18