**[|| (ИЛИ)](http://learn.javascript.ru/logical-ops" \l "%D0%B8%D0%BB%D0%B8)**

Оператор ИЛИ выглядит как двойной символ вертикальной черты:

result = a || b;

tсли хотя бы один из аргументов true, то возвращает true, иначе – false". В JavaScrip, как ми увидим далее, это не совсем так, но для начала рассмотрим только логические значения.

Получается следующая «таблица результатов»:

alert( true || true ); // true

alert( false || true ); // true

alert( true || false ); // true

alert( false || false ); // false

если значение не логичиское типа то оно к нему приводит в целях вичисления

Например, число 1 будет воспринято как true, а 0 – как false:

if (1 || 0) { // сработает как if( true || false )

alert( 'верно' );

}

Обычно оператор ИЛИ используется в if, чтобы проверить, выполняется ли хотя бы одно из условий, например:

var hour = 9;

if (hour < 10 || hour > 18) {

alert( 'Офис до 10 или после 18 закрыт' );

}

Можно передать и больше условий:

var hour = 12,

isWeekend = true;

if (hour < 10 || hour > 18 || isWeekend) {

alert( 'Офис до 10 или после 18 или в выходной закрыт' );

}

## [Короткий цикл вычислений](http://learn.javascript.ru/logical-ops" \l "%D0%BA%D0%BE%D1%80%D0%BE%D1%82%D0%BA%D0%B8%D0%B9-%D1%86%D0%B8%D0%BA%D0%BB-%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D0%B9)

JavaScript вычисляет несколько ИЛИ слева направо. При этом, чтобы экономить ресурсы, используется так называемый «короткий цикл вычисления».

Допустим, вычисляются несколько ИЛИ подряд: a || b || c || .... Если первый аргумент – true, то результат заведомо будет true (хотя бы одно из значений – true), и остальные значения игнорируются.

Это особенно заметно, когда выражение, переданное в качестве второго аргумента, имеет сторонний эффект – например, присваивает переменную.

При запуске примера ниже присвоение x не произойдёт:

var x;

true || (x = 1);

alert(x); // undefined, x не присвоен

…А в примере ниже первый аргумент – false, так что ИЛИ попытается вычислить второй, запустив тем самым присваивание:

var x;

false || (x = 1);

alert(x); // 1

## [Значение ИЛИ](http://learn.javascript.ru/logical-ops" \l "%D0%B7%D0%BD%D0%B0%D1%87%D0%B5%D0%BD%D0%B8%D0%B5-%D0%B8%D0%BB%D0%B8)

*|| запинается на «правде»,  
&& запинается на «лжи».*

Итак ми видим оператор ИЛИ вичесляють ровно столько сколько необходимо до первого true

При этом оператор ИЛИ возвращает то значение, на котором остановились вычисления. Причём, не преобразованное к логическому типу.

Например:

alert( 1 || 0 ); // 1

alert( true || 'неважно что' ); // true

alert( null || 1 ); // 1

alert( undefined || 0 ); // 0

Это используют, в частности, чтобы выбрать первое «истинное» значение из списка:

var undef; // переменная не присвоена, т.е. равна undefined

var zero = 0;

var emptyStr = "";

var msg = "Привет!";

var result = undef || zero || emptyStr || msg || 0;

alert( result ); // выведет "Привет!" - первое значение, которое является true

Если все значения «ложные», то || возвратит последнее из них:

alert( undefined || '' || false || 0 ); // 0

Итак, оператор || вычисляет операнды слева направо до первого «истинного» и возвращает его, а если все ложные – то последнее значение.

Иначе можно сказать, что "|| запинается на правде".

## [&& (И)](http://learn.javascript.ru/logical-ops" \l "%D0%B8)

Оператор И пишется как два амперсанда &&:

result = a && b;

в класичиском программирований и возращает тру если оба аргумента истина а иначе фолс

Пример с if:

var hour = 12,

minute = 30;

if (hour == 12 && minute == 30) {

alert( 'Время 12:30' );

}

Как и или в и допустим лубимие значение

if (1 && 0) { // вычислится как true && false

alert( 'не сработает, т.к. условие ложно' );

}

Если левый аргумент – false, оператор И возвращает его и заканчивает вычисления. Иначе – вычисляет и возвращает правый аргумент.

Например:

// Первый аргумент - true,

// Поэтому возвращается второй аргумент

alert( 1 && 0 ); // 0

alert( 1 && 5 ); // 5

// Первый аргумент - false,

// Он и возвращается, а второй аргумент игнорируется

alert( null && 5 ); // null

alert( 0 && "не важно" ); // 0

можно передать не сколько значений подряд при етом возратица первое ложное  а если его нет – то последнее:

alert( 1 && 2 && null && 3 ); // null

alert( 1 && 2 && 3 ); // 3