# Конструкция "switch"

Конструкция switch заменяет собой сразу несколько if.

Она представляет собой более наглядный способ сравнить выражение сразу с несколькими вариантами.

# Синтаксис

Конструкция switch имеет один или более блок case и необязательный блок default.

Выглядит она так:

switch(x) {

case 'value1': // if (x === 'value1')

...

[break]

case 'value2': // if (x === 'value2')

...

[break]

default:

...

[break]

}

* Переменная x проверяется на строгое равенство первому значению value1, затем второму value2 и так далее.
* Если соответствие установлено – switch начинает выполняться от соответствующей директивы case и далее, до ближайшего break (или до конца switch).
* Если ни один case не совпал – выполняется (если есть) вариант default.

# Пример работы

Пример использования switch (сработавший код выделен):

let a = 2 + 2;

switch (a) {

case 3:

alert( 'Маловато' );

break;

case 4:

alert( 'В точку!' );

break;

case 5:

alert( 'Перебор' );

break;

default:

alert( "Нет таких значений" );

}

Здесь оператор switch последовательно сравнит a со всеми вариантами из case.

Сначала 3, затем – так как нет совпадения – 4. Совпадение найдено, будет выполнен этот вариант, со строки alert( 'В точку!' ) и далее, до ближайшего break, который прервёт выполнение.

**Если break нет, то выполнение пойдёт ниже по следующим case, при этом остальные проверки игнорируются.**

Пример без break:

let a = 2 + 2;

switch (a) {

case 3:

alert( 'Маловато' );

case 4:

alert( 'В точку!' );

case 5:

alert( 'Перебор' );

default:

alert( "Нет таких значений" );

}

В примере выше последовательно выполнятся три alert:

alert( 'В точку!' );

alert( 'Перебор' );

alert( "Нет таких значений" );

Любое выражение может быть аргументом для switch/case

И switch и case допускают любое выражение в качестве аргумента.

Например:

let a = "1";

let b = 0;

switch (+a) {

case b + 1:

alert("Выполнится, т.к. значением +a будет 1, что в точности равно b+1");

break;

default:

alert("Это не выполнится");

}

В этом примере выражение +a вычисляется в 1, что совпадает с выражением b + 1 в case, и следовательно, код в этом блоке будет выполнен.

# Группировка «case»

Несколько вариантов case, использующих один код, можно группировать.

Для примера, выполним один и тот же код для case 3 и case 5, сгруппировав их:

let a = 2 + 2;

switch (a) {

case 4:

alert('Правильно!');

break;

case 3: // (\*) группируем оба case

case 5:

alert('Неправильно!');

alert("Может вам посетить урок математики?");

break;

default:

alert('Результат выглядит странновато. Честно.');

}

Теперь оба варианта 3 и 5 выводят одно сообщение.

Возможность группировать case – это побочный эффект того, как switch/case работает без break. Здесь выполнение case 3 начинается со строки (\*) и продолжается в case 5, потому что отсутствует break.

# Тип имеет значение

Нужно отметить, что проверка на равенство всегда строгая. Значения должны быть одного типа, чтобы выполнялось равенство.

Для примера, давайте рассмотрим следующий код:

let arg = prompt("Введите число?");

switch (arg) {

case '0':

case '1':

alert( 'Один или ноль' );

break;

case '2':

alert( 'Два' );

break;

case 3:

alert( 'Никогда не выполнится!' );

break;

default:

alert( 'Неизвестное значение' );

}