🌐
EN

🗼 JS

EPUB/PDF   👤 🔍

🏠 → The JavaScript language → JavaScript Fundamentals

📅 29th July 2019

The "switch" statement

A `switch` statement can replace multiple `if` checks.

It gives a more descriptive way to compare a value with multiple variants.

The syntax

The `switch` has one or more `case` blocks and an optional default.

It looks like this:

```
1   switch(x) {
2     case 'value1':  // if (x === 'value1')
3       ...
4       [break]
5
6     case 'value2':  // if (x === 'value2')
7       ...
8       [break]
9
10    default:
11      ...
12      [break]
13  }
```

- The value of `x` is checked for a strict equality to the value from the first `case` (that is, `value1`) then to the second (`value2`) and so on.
- If the equality is found, `switch` starts to execute the code starting from the corresponding `case`, until the nearest `break` (or until the end of `switch`).
- If no case is matched then the `default` code is executed (if it exists).

An example

An example of `switch` (the executed code is highlighted):

▶ 🖉

```
1   let a = 2 + 2;
2
3   switch (a) {
4     case 3:
5       alert( 'Too small' );
6       break;
7     case 4:
```

```
 8      alert( 'Exactly!' );
 9      break;
10    case 5:
11      alert( 'Too large' );
12      break;
13    default:
14      alert( "I don't know such values" );
15  }
```

Here the `switch` starts to compare `a` from the first `case` variant that is `3`. The match fails.

Then `4`. That's a match, so the execution starts from `case 4` until the nearest `break`.

If there is no `break` then the execution continues with the next `case` without any checks.

An example without `break`:

```
 1  let a = 2 + 2;
 2
 3  switch (a) {
 4    case 3:
 5      alert( 'Too small' );
 6    case 4:
 7      alert( 'Exactly!' );
 8    case 5:
 9      alert( 'Too big' );
10    default:
11      alert( "I don't know such values" );
12  }
```

In the example above we'll see sequential execution of three `alert`s:

```
1  alert( 'Exactly!' );
2  alert( 'Too big' );
3  alert( "I don't know such values" );
```

> ℹ️ **Any expression can be a `switch/case` argument**
>
> Both `switch` and `case` allow arbitrary expressions.
>
> For example:
>
> ```
>  1  let a = "1";
>  2  let b = 0;
>  3
>  4  switch (+a) {
>  5    case b + 1:
>  6      alert("this runs, because +a is 1, exactly equals b+1");
>  7      break;
>  8
>  9    default:
> 10      alert("this doesn't run");
> 11  }
> ```
>
> Here `+a` gives `1`, that's compared with `b + 1` in `case`, and the corresponding code is executed.

Grouping of "case"

Several variants of `case` which share the same code can be grouped.

For example, if we want the same code to run for `case 3` and `case 5`:

```
 1  let a = 2 + 2;
 2
 3  switch (a) {
 4    case 4:
 5      alert('Right!');
 6      break;
 7
 8    case 3: // (*) grouped two cases
 9    case 5:
10      alert('Wrong!');
11      alert("Why don't you take a math class?");
12      break;
13
14    default:
15      alert('The result is strange. Really.');
16  }
```

Now both `3` and `5` show the same message.

The ability to "group" cases is a side-effect of how `switch/case` works without `break`. Here the execution of `case 3` starts from the line `(*)` and goes through `case 5`, because there's no `break`.

Type matters

Let's emphasize that the equality check is always strict. The values must be of the same type to match.

For example, let's consider the code:

```
 1  let arg = prompt("Enter a value?");
 2  switch (arg) {
 3    case '0':
 4    case '1':
 5      alert( 'One or zero' );
 6      break;
 7
 8    case '2':
 9      alert( 'Two' );
10      break;
11
12    case 3:
13      alert( 'Never executes!' );
14      break;
15    default:
16      alert( 'An unknown value' );
17  }
```

1. For `0` , `1` , the first `alert` runs.

2. For `2` the second `alert` runs.

3. But for `3` , the result of the `prompt` is a string `"3"` , which is not strictly equal `===` to the number `3` . So we've got a dead code in `case 3` ! The `default` variant will execute.

✅ Tasks

Rewrite the "switch" into an "if" ↗

importance: 5

Write the code using `if..else` which would correspond to the following `switch` :

```
 1  switch (browser) {
 2    case 'Edge':
 3      alert( "You've got the Edge!" );
 4      break;
 5
 6    case 'Chrome':
 7    case 'Firefox':
 8    case 'Safari':
 9    case 'Opera':
10      alert( 'Okay we support these browsers too' );
11      break;
12
13    default:
14      alert( 'We hope that this page looks ok!' );
15  }
```

(solution)

Rewrite "if" into "switch" ↗

importance: 4

Rewrite the code below using a single `switch` statement:

```javascript
let a = +prompt('a?', '');

if (a == 0) {
  alert( 0 );
}
if (a == 1) {
  alert( 1 );
}

if (a == 2 || a == 3) {
  alert( '2,3' );
}
```

solution

‹ Previous lesson	Next lesson ›

Share 🐦 f   🔗 Tutorial map

💬 Comments

- If you have suggestions what to improve - please submit a GitHub issue or a pull request instead of commenting.
- If you can't understand something in the article – please elaborate.
- To insert a few words of code, use the `<code>` tag, for several lines – use `<pre>`, for more than 10 lines – use a sandbox (plnkr, JSBin, codepen…)