JavaScript Syntax and Operators

ALL ABOUT THE SWITCH STATEMENT

Paul D. Sheriff

BUSINESS/TECHNOLOGY CONSULTANT

paul.d.sheriff@gmail.com

Course Goals

Learn basics of JavaScript syntax and operators

Switch statement

For/in and for/of

Math, comparison and logical operators

Truthy and falsy

Exception handling

Data types

'this' keyword

The spread operator

I assume you are…

- A beginning JavaScript programmer
- Familiar with the basics of HMTL, CSS

You want to…

- Understand more about JavaScript syntax

Related Pluralsight Courses

JavaScript Variables and Types

Barry Luijbregts

JavaScript Fundamentals

Mark Zamoyta

Modules in This Course

Modules

All About the Switch Statement

- Simplify multiple if else statements
- Block level scope issue/resolution

The Difference Between for/in and for/of

- Using the appropriate for loop
- Break, continue and labels

Modules

Using Math and Comparison Operators

- Demos of operators
- 'use strict'

Working with Logical Operators and Short-circuit Evaluation

- Truthy and falsy
- How short-circuit evaluation works

Modules

Utilizing JavaScript Exception Handling

- try...catch
- finally

How to Determine JavaScript Variable Data Types

- typeof operator
- instanceof operator

Modules

Understanding 'this' in JavaScript

- Use of 'this' in different scopes
- Call() and apply() methods

Using the Powerful Spread Operator

- Manipulating arrays
- Passing arrays to functions

Switch

Switch

Use instead of multiple if...else statements

'case' statements compare to expression in switch(exp)

'break' statements exit out of each case

The 'default' statement is for no match

switch Statement

```
switch(<expression>) {
  case <expression 1>:
     // Statement(s)
     break;

  case <expression 2>:
    // Statement(s)
    break;

  default:   // If no other case is matched
     // Statement(s)
    break;
}
```

Exits out of switch

Demo

Simple switch statement

'default' statement can be anywhere

Multiple Case Statements

```
switch(<expression>) {
  case <expression 1>:
  case <expression 2>:
  case <expression 3>:
    // Statement(s)
    break;

  default:
    break;
}
```

If expression matches any case, then the statement(s) are executed.

Demo

Multiple case statements

What happens when you forget a break

Demo

Switch does a strict comparison

- Type and value must match

Block Level Scope

```
switch(<expression>) {
  case <expression 1>:
    // Statement(s)
    break;

}
```

Each case statement is NOT a block.

Block Level Scope

```
switch(<expression>) {
  case <expression 1>: {
    // Statement(s)
    break;
  }
}
```

Make statements a block by wrapping in braces.

Demo

Block level scope demo

Summary

Use switch statement for readability

More efficient than multiple if...else statements

Be careful with block level scope

Coming up in the next module…

Using the appropriate for loop

Break, continue, and labels