Switch

Justin Besteman

December 30, 2016

Synopsis

Switch statements are a powerful tool for a coder to test multiple conditions.

Why

If a coder is testing only a few conditions then an if, else, or else if are perfect tools to use. But what if there are a large number of things to test? You could still use these tools but you better get some ice to cool off your fingers when you are done! A switch statement is a way to reduce the amount code that needs to be typed in such cases, and as a bonus the code is easy to understand at a glance.

Syntax

The switch statement is structured like a train track system where the rails switch (see what I did there?) based on particular cases. With this tool, you have a switch and cases that will be executed based on the condition of the switch. After each case, you must use the keywork break because if you don't then bad things happen to good people. A switch statement is block level code so without the break it would just continue and excute all the further conditional code for other cases after the first met case code was read.

Advanced

Switch statements have the ability to “Stack cases on top of each other.” (My mentor, Sean, taught me that!) This means you can have more then one condition or case that can be met for that code to excute. At the end of the switch statement, a coder may put the keyword “default” which is triggered if none of the cases are met.

Examples

```

// ------------------ Example 1 -------------------

var traffic_light = "green";

switch (traffic_light) {

  case "green":
    console.log("Go");
    break;
  case "yellow":
    console.log("Speed up or slow down");
    break;
  case "red":
    console.log("Stop, Stop, Stoopppp!!!");
    break;

} // End of Switch Statement

// ------------------ Example 2 -------------------

var user_pick = 3;

switch (user_pick) {

  case 1:
    console.log("User 1");
    break;
  case 2:
    console.log("User 2");
    break;
  case 3:
    console.log("User 3");
    break;
  case 4:
    console.log("User 4");
    break;
  case 5:
    console.log("User 5");
    break;

} // End of Switch Statement

// ----------------- Example 3 ------------------

// This shows how cases can be stacked

var user_pick = 3;

switch (user_pick) {

  case "one":
  case 1:
    console.log("User 1");
    break;
  case "two":
  case 2:
    console.log("User 2");
    break;
  case "three":
  case 3:
    console.log("User 3");
    break;
  case "four":
  case 4:
    console.log("User 4");
    break;
  case "five":
  case 5:
    console.log("User 5");
    break;

} // End of Switch Statement

// This is powerful because if a user entered an
// integer or a string for their choice, this code
// would still work. We are able to stack both the
// integer and string cases.

// ----------------- Example 4 ------------------

// This shows the default keyword

var user_pick = 3;

switch (user_pick) {

  case "one":
  case 1:
    console.log("User 1");
    break;
  case "two":
  case 2:
    console.log("User 2");
    break;
  case "three":
  case 3:
    console.log("User 3");
    break;
  case "four":
  case 4:
    console.log("User 4");
    break;
  case "five":
  case 5:
    console.log("User 5");
    break;
  default:
    console.log("Invalid input");
    break;

} // End of Switch Statement

// Very awesome here. With this default
// statement, if the user's input doesn't
// match any case you can say it's not valid
```