Control Flow and JavaScript Errors. 14th **July** 2020.

The control flow is the order in which the computer executes statements in a script.

Code is run in order from the first line in the file to the last line, unless the computer runs across the (extremely frequent) structures that change the control flow, such as conditionals and loops.

In JavaScript we have the following conditional statements:

1). **if statement** – to specify a block of code to be executed, if a specified condition is true.

2). **else statement** – to specify a block of code to be executed, if the same condition is false.

3). **if...else statement** – to specify a new condition to test, if the first condition is false.

4). **Switch statement** – to select one of many blocks of code to be executed

1). **if statement**.

The **if** statement is the fundamental control statement that allows JavaScript to make decisions and execute statements conditionally.

The syntax for a basic if statement is as follows –

```
if (expression) {
   Statement(s) to be executed if expression is true
}
```

When a JavaScript expression is evaluated, If the resulting value is true, the given statement(s) are executed. If the expression is false, then no statement would be not executed. It is usually recommended to you use comparison operators while making decisions.

Example:Create a file called index.html and add the following code.

```
 <html>
 <body>
      <script type = "text/javascript">
         <!--
            var age = 20;

            if( age > 18 ) {
               document.write("<b>Qualifies for driving</b>");
            }
         //-->
      </script>
      <p>Set the variable to different value and then try...</p>
   </body>
</html>
```

After running the above code in your browser it will output:

Does not qualify for driving

```
Sets the variable to different value and then try…
```

2) **else statement**.

This statement is used to specify that block of code should execute if a condition is wrong.

The syntax for a basic else statement is as follows –

```
if (condition) {
  // block of code to be executed if the condition is true
} else {
  // block of code to be executed if the condition is false
}
```

Example:

```
int a = 100;

int b = 100;

if (a < b){

  print("a is less than b") // This will run because 1 < 2

   }

else{

  //This only runs if the condition for the if statement is not met

  print("a is greater than or equal to b")

   }
```

3). **if…else statement**.

The **’if…else’** statement is the next form of control statement that allows JavaScript to execute statements in a more controlled way.

Following is an example of if…else statement –

```
var age=20;

if (age < 18) {

      console.log("underage");

} else {

      console.log("let em in!");

}
```

There is a **else if** statement used to specify a new condition if the first condition is false.

The syntax for a basic else statement is as follows –

```
if (condition1)                                                        {
  //    block    of    code    to    be    executed    if    condition1    is    true
} else if (condition2)                                                 {
  //  block of code to be executed if the condition1 is false and condition2 is
true
} else {
  //  block of code to be executed if the condition1 is false and condition2 is
false
}
```

Example:

If time is less than 10:00, create a "Good morning" greeting, if not, but time is less than 20:00, create a "Good day" greeting, otherwise a "Good evening":

Solution

```
if (time < 10) {
  greeting = "Good morning";
} else if (time < 20) {
  greeting = "Good day";
} else {
  greeting = "Good evening";
}
```

Discuss your output with others?

4). **switch statement –**

This statement is used to perform different actions based on different conditions. We will discuss switch more when discussing continue and break keywords.

The syntax for a basic switch statement is as follows –

```
 switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

JavaScript Errors.

When executing JavaScript code, different errors can occur. Errors can be coding errors made by the programmer, errors due to wrong input, and other unforeseeable things.

The **try** statement lets you test a block of code for errors.

The **catch** statement lets you handle the error.

The **throw** statement lets you create custom errors.

Technically you can **throw an exception** (**throw an error** which can be a JavaScript String, a Number, a Boolean or an Object.

This is all for today, we will discus more on errors next week when we are handling Input Validation, HTML validation, range and eval error.

Remember:

No amount of academic study or watching other people code can compare to breaking open an editor and start making mistakes.

Best Wishes.
Lux Tech Academy.
Email: luxtinc@gmail.com