Program Flow: Conditional Statements

Deciding When Code Will Run

We'll cover the following

Conditional statements are another way we can control the flow of our program. They allow us to execute code only under certain *conditions* that we define.

`if` statements

Let’s start with an example. Given the variable `temperature`, we want to inform a user about weather conditions.

```
var weather = function(temperature){
  console.log("The temperature outside is", temperature, "degrees farenheight.");
}

weather(60);
```

Let’s say we want to go beyond just giving the `temperature` value and want to inform users on whether or not they should wear a jacket. How could this be done programmatically?

We can add a **statement** that will only execute **if** the `temperature` meets a certain **condition**.

```
var weather = function(temperature){
  console.log("The temperature outside is", temperature, "degrees farenheight.");

  if(temperature <= 65) {
    console.log("It's getting cold outside. Better wear a jacket!");
  }
}


weather(60);
```

An 'if' statement will only execute if the condition in parenthesis is met.

Conditional statements rely on providing a **Boolean value** that can determine whether a **code block** (the statements between the `{}` curly braces) should execute.

In the above example, `temperature <= 65` will return `true` if the temperature is less than or equal to 65. Otherwise, it will return `false`. Therefore, the code within the `if` statement will only run if `temperature` is less than or equal to 65.

The below example illustrates how conditional execution works by directly providing `true` or `false` values:

```
if(true){
  console.log("This line will execute.");
}

if(false){
  console.log("This line will not execute.");
}
```

`if`/`else` statements

Let’s take another look at our weather example. What if we want to include a statement about warmer weather, if the `temperature` is greater than 65?

One way we could do this is by creating another `if` condition:

```
var weather = function(temperature){
  console.log("The temperature outside is", temperature, "degrees farenheight.");

  if(temperature <= 65) {
    console.log("It's getting cold outside. Better wear a jacket!");
  }
  if(temperature > 65) {
  console.log("It's pleasant outside!");
  }
}

weather(70);
```

Though the above code is perfectly valid, there is a better way to write this. Since we are trying to execute certain code if a condition is true, and execute another set of code *otherwise*, we can use an `if`/`else` statement like so:

```
var weather = function(temperature){
  console.log("The temperature outside is", temperature, "degrees farenheight.");

  if(temperature <= 65) {
    console.log("It's getting cold outside. Better wear a jacket!");
  } else {
    console.log("It's pleasant outside!");
  }
}

weather(70);
```

Although both examples result in the same output, the second version is better practice because it more clearly indicates the *intent* of the code found in the conditional statements.

Exercise

Write a function that indicates whether or not to sell a stock. Use conditional statements to return a **Boolean value** that:

- returns `true` if `stockPrice` is *greater than or equal to* `sellPrice`
- returns `false` if `stockPrice` is *less than* `sellPrice`

```
var sellStock = function(stockPrice, sellPrice){
  //write your code here.
}
```

`if`/`else if`/`else` statements

Looking again to the weather example, what if you wanted multiple statements for

Looking again to the weather example, what if you wanted multiple statements for various temperatures?

Adding an `else if` allows us to make additional checks when the `if` condition is false. Let’s take a look at how this works:

```
var weather = function(temperature){
  console.log("The temperature outside is", temperature, "degrees farenheight.");

  if(temperature <= 65) {
    console.log("It's getting cold outside. Better wear a jacket!");
  } else if (temperature > 65 && temperature <= 80){
    console.log("It's pleasant outside!");
  } else {
    console.log("It's getting hot outside!");
  }
}

weather(90);
```

The condition in the `else if` statement will only be evaluated after the first `if` condition is false. If the `else if` condition is also false, the code in the `else` block will execute.

Multiple `else if` statements can be chained together, like so:

```
var weather = function(temperature){
  console.log("The temperature outside is", temperature, "degrees farenheight.");

  if(temperature <= 30) {
    console.log("It's freezing outside! It'll be best to bundle up.");
  } else if(temperature <= 55) {
    console.log("It's getting cold outside. Better wear a jacket!");
  } else if (temperature <= 75){
    console.log("It's pleasant outside!");
  } else {
    console.log("It's getting hot outside!");
  }
}

weather(20);
weather(40);
weather(60);
weather(90);
```

The second `else if` will only be evaluated if the first `else if` condition is false, and so on and so forth.

and so on and so forth.

Conditional statements are incredibly useful for defining exactly *under what circumstances* a block of code will run.

Now that you have learned about conditional statements, let’s learn how to iterate over the same piece of code through the use of loops. See you in the next lesson.