JavaScript: Logic & Loops

© New York Code + Design Academy 2014

What is logic (as it pertains to programming)?

- The control flow of your program
- Think of logic as a river that branches off in a few different ways
- It allows you to make the computer do the thinking for you!

© New York Code + Design Academy 2014

First stop: testing

- Any test returns a boolean true or false
- To test if two strings are equal:

```
"stringone" === "string two";
```

```
›false
```

- Using three equals signs instead of two also checks the object type
- If you don't check type, these are both true:
  - (10-5) == 5;
  - (10-5) == "5";
- Can cause bugs down the road!

© New York Code + Design Academy 2014

More testing examples

To test if two strings are NOT equal:
"stringone" !== "string two";

true

To test if one number is greater than another:
5 > 10;

false

<, >, <=, >= are also valid comparison operators

© New York Code + Design Academy 2014

The `if` statement

The `if` statement allows us to run code only if a certain test evaluates to `true`

```
if(5>10){
  console.log("You'll never see this in the console because 5 is not greater than 10");
}

if(5<10){
  console.log("But you'll definitely see this");
}
```

© New York Code + Design Academy 2014

The `else` statement

The `else` statement runs only if the statement in the `if` statement is false

```
if(5>10){
  console.log("You'll never see this because 5 is not greater than 10");
} else{
  console.log("You will see this though, since 5 < 10 evaluates to false");
}
```

© New York Code + Design Academy 2014

Exercises

- Write a program that checks if a variable is less than 10. If it is, alert the user that their variable is less than 10. If it is not, let the user know what the variable was and that it was greater than 10.
- Try running the script and then changing the variable's value to see how this affects the program output
- If you have extra time, write a similar program to check if a string stored in a variable is the same as another string

© New York Code + Design Academy 2014

The `else if` statement

If you want to run another test before getting to `else`, you'll want to use `else if`

```
if(5>10){
  console.log("You'll never see this because 5 is not greater than 10");
}else if(5===5){
  console.log("Yes, 5 really is equal to 5, so this will show up in the console")
}else{
  console.log("We won't get here because our else if evaluates to true");
}
```

© New York Code + Design Academy 2014

Functions

- A `function` is a way to encapsulate code for later use
- It can take arguments, which are used as variables inside the function
- It usually returns a value, which can be used later on or displayed immediately

© New York Code + Design Academy 2014

```
// Declare a function called someName that takes
// two arguments: numberOne and otherNumber
function someName(numberOne, otherNumber){
  // return the sum of numberOne, 10, and otherNumber
  return numberOne + 10 + otherNumber;
}
// call your new function, giving it 2 argument values
// numberOne = 4, otherNumber = 14
// log the result to the console
console.log(someName(4, 14));
// Calling a function is also known as "invoking" it
>28
```

© New York Code + Design Academy 2014

Another simple function example

```
// declare a function with the name alertName
// that takes one argument, somePersonsName
function alertName(somePersonsName){
  // the function returns an alert with their name
  return alert(somePersonsName);
}
// invoke the function
alertName("Zach");
```

© New York Code + Design Academy 2014

Exercises

- Declare a function that takes a name as an argument and tells the user what name they've entered, try invoking it after it has been declared
- Declare a function that takes no arguments but prints something to the console, try invoking it after it has been declared
- Declare a function that, depending upon which virtual "door" was entered, tells the user they've received a different "prize" in an alert. Try running it after it has been declared a few times with each door option

© New York Code + Design Academy 2014

Identifying data types

- JavaScript has plenty of useful built in functions
- Assuming you have a variable with some data stored inside of it, and are unaware of its data type…
- You can ask (or query) the variable for its datatype using the `typeof()` function

```
var yourData = "This is my data.";
typeof(yourData);
>string
```

© New York Code + Design Academy 2014

Accessing Array items

- If you're unsure of an items index number inside of an array, When unsure of an index number

```
var snoopyPosition = myArr.indexOf("Snoopy");
console.log(myArr[snoopyPosition]);
>"Snoopy"
```

© New York Code + Design Academy 2014

Loops

- A loop is a block of code that gets repeated for a defined amount of iterations (for loop) or until a certain condition is met (while loop)
- Typically one variable or condition in the loop changes each time it is run

© New York Code + Design Academy 2014

Loops - for

```
for(var i = 0; i<10; i++){
  console.log(i)
}

>>0
1
2
3
...
9
```

© New York Code + Design Academy 2014

Loops - for

```
beers = ["Lagunitas", "Peak"]
for(var i = 0; i< beers.length; i++){
  console.log(beers[i])
}

>>"Lagunitas"
"Peak"
```

© New York Code + Design Academy 2014

Exercise - `for` loop

- Create an array filled with two arrays
- Inside of each of these arrays should be several strings
- Use two loops to display the information (strings) inside of each of these arrays in the console

© New York Code + Design Academy 2014

Loops - while

```
x = 6
while(x < 10){
  console.log("On number " + x)
  x++;
}
>>6
7
8
9
```

© New York Code + Design Academy 2014

Exercise - `while` loop

- Create a while loop that "sings" the classic song "99 Bottles of Root Beer on the Wall". The code it outputs to the console should look similar to this:
- "99 bottles of root beer on the wall, 99 bottles of root beer...take one down, pass it around 98 bottles of root beer on the wall, 98 bottles of root beer on the wall, 98 bottles of root beer...take one down, pass it around 97 bottles of root beer on the wall," etc., all the way to 0 bottles.

© New York Code + Design Academy 2014