Cheatography

Getting Started with Programming-Javascript Cheat Sheet

by AEAE00 via cheatography.com/30526/cs/9106/

Using console.log

You may have noticed that the interpreter doesn't print out every single thing it does. So if we want to know what it's thinking, we sometimes have to ask it to speak to us.
console.log() will take whatever is inside the parentheses and log it to the console below your code—that's why it's called console.log()!
This is commonly called printing out

Comparisons

So far we've learned about three data types:
strings (e.g. "dogs go woof!")
numbers (e.g. 4, 10)
booleans (e.g. false, 5 > 4)
Now let's learn more about comparison operators.
List of comparison operators:
> Greater than
< Less than
<= Less than or equal to
>= Greater than or equal to
=== Equal to
!== Not equal to

Decisions, decisions

Nice work on comparisons! Now let's see how we can use comparisons to ask yes or no questions.

Decisions, decisions (cont)

Say we want to write a program that asks whether your name is longer than 7 letters. If the answer is yes, we can respond with "You have a long name!" We can do this with an if statement:

```
if( "myName".length >= 7 )
{
    console.log("You have a long name!");
}
```

An if statement is made up of the if keyword, a condition like we've seen before, and a pair of curly braces { }. If the answer to the condition is yes, the code inside the curly braces will run.

Computers are smart

Great! We used an if statement to do something if the answer to the condition was yes, or true as we say in JavaScript. In addition to doing something when the condition is true, we can do something else if the condition is false. For example, if your name is shorter than 7 letters, we can respond with "You have a short name!" We can do this using an if / else statement:

```
if( "myName".length >= 7 )
{
    console.log("You have a long name!");
}
else {
    console.log("You have a short name!");
}
```

Just like before, if the condition is true, then only the code inside the first pair of curly braces will run. Otherwise, the condition is false, so only the code inside the second pair of curly braces after the else keyword will run.
In the example above the condition "myName".length >= 7 evaluates to false since "myName" only has 6 letters. Since the condition is false, only the code inside the curly braces after the else keyword runs, and prints out You have a short name!.

Mid-lesson breather

1. Confirm and prompt
We can make pop-up boxes appear!

```
confirm("I am ok");
prompt("Are you ok?");
```

2. Data types
a. numbers (e.g. 4.3, 134)
b. strings (e.g. "dogs go woof!", "JavaScript expert")
c. booleans (e.g. false, 5 > 4)
3. Conditionals
If the first condition is met, execute the first code block. If it is not met, execute the code in the else block. See the code on the right for another example.

Math and the modulo

Let's meet an interesting symbol called modulo. When % is placed between two numbers, the computer will divide the first number by the second, and then return the remainder of that division.
So if we do 23 % 10, we divide 23 by 10 which equals 2 with 3 left over. So 23 % 10 evaluates to 3.

Substrings

Sometimes you don't want to display the entire string, just a part of it. For example, in your Gmail inbox, you can set it to display the first 50 or so characters of each message so you can preview them. This preview is a substring of the original string (the entire message).
Code:

Substrings (cont)

`"some word".substring(x, y)` where x is where you start chopping and y is where you finish chopping the original string.
The number part is a little strange. To select for the "he" in "hello", you would write this:

```
"hello". substring(0, 2);
```

Each character in a string is numbered starting from 0, like this:
`
0 1 2 3 4
| | | | |
h e l l o`
The letter h is in position 0, the letter e is in position 1, and so on. Therefore if you start at position 0, and slice right up till position 2, you are left with just he

More examples:

1. First 3 letters of "Batman"

```
"Batman".substring(0,3);
```

2. From 4th to 6th letter of "laptop"

```
"laptop".substring(3,6);
```

Variables

Variables
We have learned how to do a few things now: make strings, find the length of strings, find what character is in the nth position, do basic math. Not bad for a day's work!

Variables (cont)

To do more complex coding, we need a way to 'save' the values from our coding. We do this by defining a variable with a specific, case-sensitive name. Once you create (or declare) a variable as having a particular name, you can then call up that value by typing the variable name.
Code:

```
var varName = data;
```

Example:

a. `var myName = "Leng";`

b. `var myAge = 30;`

c. `var isOdd = true;`

More Variable Practice

We have seen how to create a variable. But how do we use it? It is useful to think that any time you type the variable's name, you are asking the computer to swap out the variable name and swap in the value of the variable.
For example:

```
var myName = "Steve Jobs";
myName.substring(0,5)
```

Look at the second line above. You have asked the computer to swap out `myName` and swap in Steve Jobs, so

```
myName.substring(0,5)
```

becomes

```
"Steve Jobs".substring(0,5)
```

More Variable Practice (cont)

which evaluates to Steve.
Another example

```
var myAge = 120;
```

What is

```
myAge % 12
```

?

So the variable stores the value of the variable, whether that is a number or a string. As you will see soon, this makes writing long programs much easier!

Conclusion: Part 1

Let's do a quick review!
Data types
strings (e.g. "dogs go woof!")
numbers (e.g. 4, 10)
booleans (e.g. false, 5 > 4)
Variables
We store data values in variables. We can bring back the values of these variables by typing the variable name.
Manipulating numbers & strings
comparisons (e.g. >, <=)
modulo (e.g. %)
string length (e.g. "Emily".length;)
substrings (e.g. "hi".substring(0, 1);)
console.log( )
Prints into the console whatever we put in the parentheses.