Udemy JS notes

Section 9

# Lesson 101 - Intro

Use Snippets to test multiple lines of code and keep them.

Open developer tools, go to Console tab, click the >> next to File System and choose Snippets from the dropdown.

For correct syntax and proper format, follow this style guide:

<https://github.com/rwaldron/idiomatic.js/>

# Data types

## String

A string is a data type that has no function but is output at face value. It is defined with quotation marks. Doesn't matter if you use ' or ", but they should be consistent for clarity. Angela recommends double quotes since it's what is used most for proper syntax.

Note: Be careful not to copy and paste smart quotes: the indicators need to open and close with the same symbol – smart quotes use two different symbols and will mess up your code.

string

alert("Hello!");

function message end

## Number

Numbers need no quotes since they are not used as code functions and are immediately recognized

## Boolean

Boolean are simply true or false.

From MDN:

[Number](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Data_structures#number_type): used for all number values (integer and floating point) except for very big integers.

[BigInt](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Data_structures#bigint_type): used for arbitrarily large integers.

[String](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Data_structures#string_type): used to store text.

[Boolean](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Data_structures#boolean_type): true and false — usually used for conditional logic.

[Symbol](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Data_structures#symbol_type): used for creating unique identifiers that won't collide.

[Undefined](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Data_structures#undefined_type): indicating that a variable has not been assigned a value.

[Null](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Data_structures#null_type): indicating a deliberate non-value.

# Variables – Lesson 105

A variable is a keyword: When the computer sees this, it will know to construct a container. The box gets a name. When we get the string, it goes in the box.

keyword

var myName = "Lisa";

name string

## Naming variables

* Give meaningful names
* Can use letters, numbers, underscore, dollar sign (abc123\_$)
* Best to use camelCase
* Can't use:
* var (myVar is ok)
* begin with a number (but can contain a number)
* contain a space (will read only word before space)

## Variables with strings

Concatination:

Hello, Lisa

var message = "Hello";

var name = "Lisa";

alert(message + ", " + name);

var myName = "Lisa";

var yourName = prompt ("What is your name?"); //*Pickle//*

alert ("My name is " + myName + "! Welcome, " + yourName + "!");

After running and typing "Pickle" for the yourName prompt:

My name is Lisa! Welcome, Pickle!

length

var tweet = prompt("Write your tweet."); //*150 charaters*//

alert ("You have written "+tweet.length+" characters, you have "+(228-tweet.length)+" characters left.");

You have written 150 characters, you have 78 characters left.

## Slice

Dummy text

A variable is a keyword: When the computer sees this, it will know to construct a container. The box gets a name. When we get the string 140 When the computer sees this, it will know to construct a container just a few more until we get here and then well over until we get to 280

var myName = "Lisa";

alert (tweet.slice(0,1)); will result in L

With prompt and alert:

var tweet = prompt("Write your tweet.");

alert (tweet.slice(0,140));

Simpler:

alert prompt("Write your tweet.").slice(0,140));

## Change case

var name = ("lisa"); //lisa

alert("Hello, "+name.toUpperCase+"!"); //LISA!

Exercise

Make prompt input result in upper case first character followed by lower case for the remainder.

var name = prompt ("What is your name?");

var firstLetter = name.slice(0,1);

var remaining = name.slice(1,name.length);

alert("Hello, "+firstLetter.toUpperCase() + remaining.toLowerCase() +"!");

## Variables with numbers

Numbers mostly use basic math

var a = 2 + 3; //5

var b = 10 - 2; //8

var c = 3 \* 3; //9

var d = 6 + 2; //3

Modulo is unique. It gives the remainder of an equation and the operand is a %.

var e = 10 % 2; //0

var f = 9 % 2; //1

## Increments – Lesson 114

Long way:

var x = 5;

x = x + 1; //6

Short way:

var x = 5;

x++; //6

var x = 5;

x--; //4

var x = 5;

x += 2; //7

var x = 5;

var y = 3;

x += y; //8

x -= y; //2

x \*= y; //15

x /= y; //1.4

## Arrays – Lesson 131

An array is a variable containing multiple values.

var guestList = ["Angela", "Jack", "Pam", "James", "Lara", "Jason"];

var guestName = prompt ("What is your name?");

if (guestList.includes(guestName)) {

alert("Welcome!");

}else {

alert("Sorry!");

}

### Guestlist - sandbox.js line 270, 324

# Functions – Lesson 115

A function takes a bunch of instructions and bundles them into one executable instruction.

## Creating a function

function nameOfFunction() { //Creates the function.

alert ("action1");

alert ("action2");

alert ("action2");

}

## Calling a function

nameOfFunction(); //Calls the function.

## Basic (vanilla)

//performs instructions of actions

function getMilk() {

console.log ("moveLeft");

console.log ("buyMilk");

console.log ("moveLeft");

}

getMilk(); //Just gets the milk.

## Intermediate (chocolate)

//performs instructions and allows input

function getMilk(money) {

var numbBottles = Math.floor(money/1.5);

console.log ("moveLeft");

console.log ("buy "+numbBottles+" bottles of milk.");

console.log ("moveLeft");

console.log ("moveLeft");

}

getMilk(10); //$10 to buy as much milk as possible

//'buy 6 bottles of milk.'

## Complex (strawberry?)

// performs actions, allows input and returns output

function getMilk(money) {

var numBottles = Math.floor(money/1.5);

console.log ("moveLeft");

console.log ("buy "+numBottles+" bottles of milk.");

//buy 2 bottles of milk.

console.log ("moveLeft");

console.log ("moveLeft");

return money % 1.5;

}

getMilk(4); //returns 1($) as remainder after buying 2.

Now bind to a variable called change:

var change = getMilk(4);

//change = 1($) as remainder after buying 2.

## Function Exercises

### Love calculator - sandbox.js line 158

### Reverse string - sandbox.js line 169

## Side note: about functions

### 3 types of functions in JS

**1- Callback Functions :**  
**a)** Functions with no return  as in  function typeOut () { ...}  
**b)**Functions with return as in function fibonacci(n) { ... return output;}  
**c)**Anonymous Functions  as in document.querySelectorAll(".drum")[i].addEventListener("click",function () { ...});  
  
**2- Constructor Functions:**

As in  function teacher(name, subject){ this.name = name , this.subject = subject; }

**3- Higher Order Functions:**

As in  document.querySelectorAll(".drum")[i].addEventListener("click",function () { ...});

Make name Sentence case

var name = prompt("What is your name?"); //lIsA

var numLetters = name.length; //gets number of lettters in name

var firstLetter = name.slice(0,1); //slices out first letter

var restOfName = name.slice(1,numLetters); //slices out rest

var firstCap = firstLetter.toUpperCase(); //caps first letter

var restLower = restOfName.toLowerCase();//lc rest

alert("Hello, "+(firstCap)+(restLower)); //”Hello, Lisa”

# Control Statements – Lesson 126

## Comparitors

=== equal to and same data type

== equal to, data type irrelevant

!== not equal to

> greater than

< lesser than

> greater than

< lesser than

>= greater than & equal

<= lesser than & equal

&& and

|| or

! not

## if else

The **if...else** statement executes a statement if a specified condition is [truthy](https://developer.mozilla.org/en-US/docs/Glossary/Truthy). If the condition is [falsy](https://developer.mozilla.org/en-US/docs/Glossary/Falsy), another statement in the optional else clause will be executed.

function testNum(a) {

let result;

if (a > 0) {

result = 'positive';

} else {

result = 'NOT positive';

}

return result;

}

console.log(testNum(-5));

// expected output: "NOT positive"

### BMI calculator - sandbox.js line 176

### Leap year - sandbox.js line 211

### FizzBuzz - sandbox.js line 282

## while loop

The **while** statement creates a loop that executes a specified statement as long as the test condition evaluates to true. The condition is evaluated before executing the statement. Use this when you are determining a state.

let n = 0;

while (n < 3) {

n++;

}

console.log(n);

// expected output: 3

### Bottles of beer - sandbox.js line 418

### FizzBuzz - sandbox.js line 364

## for loop

The **while** statement creates a loop that executes a specified statement as long as the test condition evaluates to true. The condition is evaluated before executing the statement. You are trying to iterate.

let n = 0;

while (n < 3) {

n++;

}

console.log(n);

// expected output: 3

### FizzBuzz - sandbox.js line 363

### fibonacci - sandbox.js exercise line 484

I want: fibonacciGenerator();

to return:

[0,1,1,2,3] //an array

tasks:

* create a function that feeds an array with the sum of the last 2 numbers pushed
* start with 0
* return 5 values in array

n = array length

while: n > 0 ?

and: n <= 5 ?

add to array: n = [array length + (array length+1)]

0 = apples

1 = bananas

2 = cherries

3 = dates

4 = eggs

5 = figs

6 = guava

7 = ham

8 = ice cream

9 = jam

10 = kiwi

11 = leek

12 = mango

13 = nectarine

14 = orange

15 = papaya

16 = quince

17 = raspberry

18 = squash

19 = tamarind

20 = uvula

21 = velvet

[0] = apples (1)

[1] = bananas (0+1)

1++ = 2, 2-1 = bananas