SUMMARY OF UDACITY COURSE

INTRO TO JAVASCRIPT

RATINGS: 5/5

LESSON 1 – WHAT IS JAVASCRIPT

UNIT 1: Intro to JavaScript

- JS is a programming language made for the web
- Opportunities with JS are endless

UNIT 2: History of JS

- It was created in 10days
- It is not related to Java; it was initially called “LiveScript” but later called “JavaScript”
- One of the 3 foundational pillars of Front-end development
- HTML and CSS are markup languages, JavaScript is a programming language

UNIT 3: The JavaScript Console

- We will be using the console to run most of the JS code in this course, you can use the Chrome Dev Tool

UNIT 4: Developer Tools on Different Browsers

- Every modern browser has its own developer tools

UNIT 5: console.log

- “console.log” is used to display content to the JS console
- Dev Tool can be used to debug code

UNIT 6: JavaScript Demo

UNIT 7: Summary

- All major browser come with built in JS Engines

LESSON 2 – DATA TYPES AND VARIABLES

UNIT 1: Intro to Data Types

- Data is important, data and data type are the basic building blocks of any programming language
- It important to know the type of data you are using and when to use each type

UNIT 2: Numbers

- Number data type includes any positive or negative integer, as well as decimals
- Comparison operators: "<, >, <=, >=, ==, !="
- "true" and "false" are called Boolean Values

UNIT 3: Comments

- Comments help to clarify the meaning of your code, they are not executed by the JS engine
- It makes code more readable
- Use "//" to comment in JS
- Snapshot

UNIT 4: Quiz – First Expression (2-1)

UNIT 5: Strings

- It is correct to use double " or single ' quotes with strings as long as you are consistent

UNIT 6: String Concatenation

- String are a collection of characters enclosed inside double or single quotes
- Concatenating simply means adding two strings together
- JS has a peculiar behavior called "implicit type coercion" (snap shot)

UNIT 7: Variables

- With variables you no longer need to work with one-time-use data
- Variable store data in JS
- "var variable_name = ***"
- When creating a variable you write the name in "camelCase"

UNIT 8: Quiz – Converting Temperatures (2-2)

UNIT 9: String Indexing

- Index help us access individual characters in the string
- Character within a String start indexing from "0"

UNIT 10: Escaping Strings

- The backslash (\) character helps us to use quotes within a String
- We use "\" to escape characters in JS
- SNAPSHOT
- Special Characters: snapshot

UNIT 11: Comparing Strings

- When comparing String the “case” matters
- When comparing Strings the comparison happens character by character for the ASCII values

UNIT 12: Quiz – Favorite Food (2-3)

UNIT 13: Quiz – String Equality for All (2-4)

UNIT 14: Quiz – All Tied Up (2-5)

UNIT 15: Quiz – Yosa Buson (2-6)

UNIT 16: Booleans

- Booleans can take either of two values: “true” or “false”
- A Boolean variable is mainly essential in evaluating the outcome of conditional (comparison), the result of a comparison is always a Boolean variable

UNIT 17: Quiz – Facebook Post (2-7)

UNIT 18: Null, Undefined, and NaN

- “null”: refers to the “value of nothing”
- “undefined”: refers to “absence of value”, will be returned if we don’t assign a value to a variable
- “NaN”: stands for “Not-A-Number”, usually indicates an error with number operations

UNIT 19: Equality

- Using “==” as a measure of equality will implement “Type Conversion” if we are comparing different data types
- If we don’t want this “Type Conversion” we need to use the “Strict Comparison/Equality ===”
- Snapshot: implicit type coercion

UNIT 20: Quiz – Semicolons (2-8)

- Not adding semicolons to the end of each line can cause errors and bugs in your program

UNIT 21: Quiz – What’s my Name? (2-9)

UNIT 22: Quiz – Out to Dinner (2-10)

UNIT 23: Quiz – Mad Libs (2-11)

UNIT 24: Quiz – One Awesome Message (2-12)

UNIT 25: Quiz – Lesson 2 Summary

- Covered the basic JS data types

LESSON 3 – CONDITIONALS

UNIT 1: Intro to Conditionals

- The main goal of writing code is to solve problem

UNIT 2: Flowcharts (3-1)

- A flowchart is a visual diagram that outlines the solution to a problem through a series of logical statements. The order in which statements are evaluated and executed is called the "control flow"

UNIT 3: Flowchart to Code

- "if/ else statement"

UNIT 4: If…Else Statements

- If…else statements allow you to execute certain pieces of code based on a condition, or a set of conditions, being met
- The values inside the "if" statement is always converted to "true" or "false", either the code in the "if" statement or the one in the "else" statement is run, but not both
- You cannot not use only an "else" statement

UNIT 5: Else if Statements

- Sometimes two conditionals are not enough
- "else if" statement is used to represent a secondary check, it adds an extra conditional statement

UNIT 6: Quiz – Even or Odd (3-2)

UNIT 7: Quiz – Musical Groups (3-3)

UNIT 8: Quiz – Murder Mystery (3-4)

UNIT 9: More Complex Problems

- We can combine logical expressions with logical operators to form more complex expressions

UNIT 10: Logical Operators

- "&&": means the AND logical operation
- "||": means the OR logical operation
- "!": means the NOT logical operation

- By combining two Boolean values together with a logical operator, we create a logical expression that returns another Boolean value

UNIT 11: Logical AND and OR

- Truth tables: they show all the possible combination of inputs in a logical expression
- Snap shots
- Short-Circuiting; describes the event when later arguments in a logical expression are not considered because the first argument already satisfies the condition

UNIT 12: Quiz – Checking your balance (3-5)

UNIT 13: Quiz – Ice Cream (3-6)

UNIT 14: Quiz – What do I wear (3-7)

UNIT 15: Advanced Conditionals

- Some advanced conditionals

UNIT 16: Truthy and Falsy

- Every value in JS has an inherent Boolean value. When that value is evaluated in the context of a Boolean expression, the value will be transformed into that inherent Boolean value
- A value is "FALSY/TRUTHY" if it converts to "false/true" when evaluated in a Boolean context, snap shot
- Essentially if it is FALSY it is TRUTHY

UNIT 17: Ternary Operator

- The "ternary operator" provides us with a shortcut alternative for writing lengthy if…else statements
- "conditional ? (if condition is true) : (if condition is false)", snap shots

UNIT 18: Quiz – Navigating the Food Chain (3-8)

UNIT 19: Switch Statement

- If you find yourself repeating "else if" statements in your code, where each condition is based on the same values, then it might be time to use a switch statement
- "Falling through", sometimes you will want to leverage this attribute of Switch statements
- Adding a "break" statement to "case" statement
- A "switch statement" is another way to chain multiple "else if " statements bases on the same value without using conditional statements.
- "switch (option) { case **: ; case **: ; …}"Snap Shot

- The “break statement” can be used to terminate a switch statement and transfer control to the code following the terminated statement, it fixes the issue of switch statement “falling through” to other case clauses
- Snap Shot

UNIT 20: Falling-through

- Sometimes we might want to leverage the “falling-through” behavior of switch statements to our advantage
- For example, when the code follows a hierarchical-type structure
- The “default” case will be executed when none of the values match the values of the switch expression

UNIT 21: Quiz – Back to School (3-9)

UNIT 22: Lesson 3 Summary

- Learn to break down problems before trying to solve them

LESSON 4 – LOOPS

UNIT 1: Intro to Loops

- Control flow of program with control statements like Conditional
- How to repeatedly execute a block of code

UNIT 2: While Loops

- Repeating the same code over and over again is not the best choice sometimes look for a better way
- Loops let you iterate over values and repeatedly run a block of code
- “while (condition) {block of code to repeat}”

UNIT 3: Parts of a While Loop

- Different kinds of loop but they essentially do the same thing, they repeat an action some number of times
- 3 main piece of information any loop should have: SNAPSHOT
- If a loop is missing any of these three things, we might find ourselves in trouble
- “An Infinite Loop will run forever”

UNIT 4: Quiz – JuliaJames (4-1)

UNIT 5: Quiz – 99 Bottles of Juice (4-2)

UNIT 6: Quiz – Countdown, Liftoff (4-3)

UNIT 7: For Loops

- It is easy to forget the 3 main pieces of a loop when using a “while” loop

UNIT 8: Parts of a For Loop

- For loops are the most common type of loop in JS
- For loops explicitly forces you to define the start point, stop point and each step of the loop
- “for (start; stop; step) { block of code}”

UNIT 9: Nested Loops

- You can add loops in a loop
- Nested Loops add another layer of complexity to our code

UNIT 10: Increment and Decrement

- Increment/ Decrement Operators provide shortcut/ short hand ways for implementing increment/ decrement
- Snap shot

UNIT 11: Quiz – Changing the Loop (4-4)

UNIT 12: Quiz – Fix the Error (4-5)

UNIT 13: Quiz – Fix the Error (4-6)

UNIT 14: Quiz – Factorials! (4-7)

- We can declare a variable using: “let”, “const” and “var”
- Scope: is defined as a specific portion of the code, there are three types of scope in JS: “Global”, “Function” and “Block”
- Snap shot
- “var” keyword cannot have “Block Scope”
- “const” keyword used to declare constants, the value of this variable CANNOT be changed or reassigned throughout the code

UNIT 15: Quiz – Find my Seat (4-8)

UNIT 16: Lesson 4 Summary

- Loops are very fundamental concepts for any programming language
- Functions are also very important

LESSON 5 – FUNCTIONS

UNIT 1: Intro to Functions

- Functions are reusable block of code packaged together
- It is like a shortcut (it abstract some code)

UNIT 2: Function Example

- Reverse Function Example

UNIT 3: Declaring Functions

- Functions allow you to package up lines of code that you can use (and often use) in your programs, sometimes they take parameters
- Snapshot
- "return" statement, snapshot
- To gey your function to do something, you have to "invoke" or "call" the function using the function name, followed by parentheses with any "arguments":
- A "parameter" is always going to be a variable name and appears in the "function declaration" while an "argument" is always going to be a value and will always appear in the code when the function is "called or invoked"

UNIT 4: Function Recap

- Snapshot
- "return" statement explicitly make your function return a value

UNIT 5: Quiz – Laugh it Off (5-1)

UNIT 6: Quiz – Laugh it Off (5-2)

UNIT 7: Return Values

- A function is always going to return some value back to the caller. If a return value is not specified, then the function will just return back "undefined"
- "return", use to stop execution of a function and returns a value back to the caller
- "console.log" is not the same as "return"
- "console.log" should only be used to test your code in the JS console

UNIT 8: Using Return Values

- A function's return value can be stored in a variable or reused throughout your program as a function argument

UNIT 9: Scope

- Scope will be at the heart of a lot of coding bugs

UNIT 10: Scope Example

- Global Scope: identifiers can be accessed everywhere within your program

UNIT 11: Shadowing

- Scope overriding/ overshadowing
- Declaring our variable again in the function will prevent it from override/ overshadowing the one in the Global Scope

UNIT 12: Global Variables

- Global variables should only be used when really needed
- They call conflict with other global variables of the same name, as program gets larger it will get harder to keep track
- Work on minimizing the use of global variables as much as possible

UNIT 13: Scope Recap

- Snap shot

UNIT 14: Hoisting

- "Hoisting": before any JS is executed, all function declarations are "hoisted" to the top of their current scope
- Hoisting also works with "variable declaration", note only the declaration is hoisted
- So declare variable and function at the top before implementing them

UNIT 15: Hoisting Recap

- Declare functions and variables at the top of your script, so the syntax and behavior are consistent with each other
- Snapshot

UNIT 16: Quiz: Build a Triangle (5-3)

- Professionals plan out their code before writing anything

UNIT 17: Function Expressions

- You can store functions in variable, it is called "Function Expression"
- Two ways to define a function in JS: "Function Declaration" and "Function Expression"
- Snap shot
- Function Expressions are not hoisted only the "variable declarations" are hoisted

UNIT 18: Functions as parameters

- Function that is passed into another function is called a "callback"
- "Named function expression"
- "Inline Function Expression": when you pass a function inline as an argument to another function, snap shot
- "Anonymous inline function expression" is often used with function callbacks that are probably not going to be reused elsewhere

UNIT 19: Function Expression Recap

- "Function Expression": when a function is assigned to a variable. The function can be names, or anonymous. Use the variable name to call a function defined in a function expression
- Snap shot

UNIT 20: Quiz – Laugh (5-4)

UNIT 21: Quiz – Cry (5-5)

UNIT 22: Quiz – Inline (5-6)

UNIT 23: Lesson 5 Summary

- They are very fundamental in programming
- Next lesson is on Arrays

LESSON 6 – ARRAYS

UNIT 1: Intro to Arrays

- Array data structure that can store multiple data structures

UNT 2: Donuts to Code

- An "Array" is a data structure that you can use to store multiple values and Arrays are also organized
- "[ ]": signifies an empty array, separate each element of an Array with ","

UNIT 3: Creating an Array

- You can define a new array by listing values separated with commas between square brackets "[ ]"
- You can store any data structure in an Array

UNIT 4: Accessing Array Elements

- Elements in an Array refer to individual elements in an Array
- "index": references the location, or position, of an element in an array
- Indexing in Arrays start from "0"

UNIT 5: Array Index

- If you try to access an element at an index that does not exist, a value of "undefined" will be returned back
- You can change the value of an element in array, by setting it equal to a new value

UNIT 6: Quiz – UdaciFamily (6-1)

UNIT 7: Quiz – Building the Crew (6-2)

UNIT 8: Quiz – The Price is Right (6-3)

UNIT 9: Array Properties and Methods

- Arrays have a number of properties and methods
- Properties: special pieces of information about a data structure
- Methods: special functions that an Array can call
- Check out https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array
- Type "[ ]" into the JS console to get a list of available Array methods

UNIT 10: Length

- Can find the "length" of an array by using its "length" property

UNIT 11: Push

- Quite a few built-in methods for adding and removing elements from an array. The two most common methods for modifying an array are "push( )" and "pop( )"
- Use "push( )" to add elements to the "end of an array", you need to pass the value of the element you want to add to the end of the array as an argument to the function
- The "push( )" also returns the length of Array after an element has been added

UNIT 12: Pop

- You can "pop( )" method to remove elements from the "end of an array"
- You don't need to pass an argument to the "pop( )" method, it will always remove the last element from the end of the array, it also returns the element that has been removed in case you need it

UNIT 13: Splice

- "splice( )" is another handy method that allows you to add and remove elements from anywhere within an array
- Snap shot
- "arrayName.splice(arg1, arg2, item1, …, itemX)":
- "arg1": mandatory argument. Specifies the starting index position to add/remove items
- "arg2": optional argument, specifies the count of elements to be removed. If set to 0, no items will be removed
- "item1, …, itemX": are the items to be added at index position arg 1
- "splice( )" is an incredibly powerful method that allows you to manipulate your arrays in a variety of ways
- Check out https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/splice

UNIT 14: Quiz – Colors of the Rainbow (6-4)

UNIT 15: Quiz – Quidditch Cup (6-5)

UNIT 16: Quiz – Joining the Crew (6-6)

UNIT 17: Quiz – Quiz – Checking out the docs (6-7)

- Check out https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array
- "reverse( )", "sort( )", "shift( )", "unshift( )", "join( )"

UNIT 18: Arrays Loops

- Snap shot

UNIT 19: The forEach Loop

- Arrays have a set of specials methods to help you iterate over and perform operations on collections of data
- You can use a for loop or use array methods: "forEach( )", "map( )"
- Snap shot

UNIT 20: Quiz – Another Type of Loop (6-8)

UNIT 21: Map

- The various methods for looping through an Array have their Pros and Cons
- The "for" loop offers a lot of versatility
- "map( )": it returns a new array, snap shot

UNIT 22: Quiz – I Got Bills (6-9)

UNIT 23: Arrays in Arrays

- You can store an array in a arrays, it resembles a Grid Structure
- You can use a loop in a loop to iterate through the Grid

UNIT 24: 2D Donut Arrays

UNIT 25: Quiz – Nested Numbers (6-10)

UNIT 26: Lesson 6 Summary

- We will be talking about "Objects" in the next lesson

LESSON 7 – OBJECTS

UNIT 1: Intro to Objects

- Object help us to wrap a lot of functionality and data into a data structure

UNIT 2: Objects in Code

- "{ }": to create an Object
- Objects can have properties and methods
- A "method" is a function associated with an "Object"
- "typeof variable_name": returns the name of "variable_name" data type

UNIT 3: Quiz – Umbrella (7-1)

UNIT 4: Objects

- "Objects": are data structures in JS that let you store data about a particular thing and helps you keep track of that data by using a "key"
- "each key – value pair on its own line to improve readability"

UNIT 5: Object-literal notation

- Snap shot
- To retrieve info using keys from an Object: 1) sister["key"] 2) sister.key

UNIT 6: Naming Conventions

- Quote around property names are not necessary but they help prevent some bugs
- "don't use numbers as your first notation (to start) for property names"
- "don't use space and hyphens in your property names, you can use camelCases instead"
- They also apply to variable names

UNIT 7: Summary of Objects

- Objects are one of the most important data structures in JS
- They are extremely powerful data type and are all over the place in JS or any object oriented programming language

UNIT 8: Menu Items (7-2)

UNIT 9: Bank Accounts 1 (7-3)

UNIT 10: Bank Accounts 2 (7-4)

UNIT 11: Facebook Friends (7-5)

UNIT 12: Donuts Revisited (7-6)

UNIT 13: Lesson 7 Summary

- Try to create something with this knowledge