Module 1 CSS107 Javascript Basics : Introduction, Syntax and Statements, Comments, Operators, Variables, Assignment, Loops, If and Switch conditions, break & Continue.

Notes

JavaScript Basics

- JavaScript is the world's most popular Web scripting language. When JavaScript was created, it was initially called “LiveScript”, as it evolved, LiveScript came to be known ECMAScript. JavaScript is one of the 3 languages all web developers must learn:
    - HTML to define the content of web pages
    - CSS to specify the layout of web pages
    - JavaScript to program the behaviour of web pages

Javascript : Syntax and Statements

- JavaScript syntax is the set of rules, that define how JavaScript programs are constructed. The JavaScript syntax defines two types of values:
    - Fixed values called Literals.
        - Numeric Literals eg. 10.6, 512
        - String Literals eg. “Good Day”, ‘Thank You’

- Variable values called Variables.
    - Variables are used to store data values.
    - JavaScript uses the var keyword to declare variables.
    - An equal sign is used to assign values to variables.
    - eg. var x; x = 6;

Javascript : Syntax and Statements

- An expression is a combination of values, variables, and operators, which computes to a value.
  JavaScript keywords are used to identify actions to be performed. In JavaScript, identifiers are used to name variables (and keywords, and functions, and labels).
- The rules for legal names are much the same in most programming languages.In JavaScript, the first character must be a letter, or an underscore (_), or a dollar sign ($).Subsequent characters may be letters, digits, underscores, or dollar signs.

Javascript : Comments

- JavaScript comments can also be used to prevent execution, when testing alternative code. Single line comments start with //. Any text between // and the

end of the line will be ignored by JavaScript (will not be executed). Multi-line comments start with /* and end with */. Any text between /* and */ will be ignored by JavaScript.

- eg. // Single Line comment

  /*
  Multiline

  comment*/

Javascript : Arithmetic Operators

- Arithmetic operators are used to perform arithmetic on numbers:

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
**	Exponentiation
/	Division
%	Modulus
++	Increment
--	Decrement

Javascript : Assignment Operators

- Assignment operators assign values to JavaScript variables.

Operator	Example	Same As
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
**=	x **= y	x = x ** y

Javascript : Relational Operators

- Relational operators are used to compare values.

Operator	Description
==	equal to
===	equal value and equal type
!=	not equal
!==	not equal value or not equal type
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
?	ternary operator

Javascript : Logical Operators

- Logical operators are used to evlaute expression to TRUE or FALSE states.

Operator	Description
&&	logical and
||	logical or
!	logical not

Javascript : Bitwise Operators

- Bit operators work on 32 bits numbers.

Operator	Description
&	AND
|	OR
~	NOT
^	XOR
<<	Zero fill left shift
>>	Signed right shift
>>>	Zero fill right shift

Javascript : Variables

- All JavaScript variables must be identified with unique names.
- The general rules for constructing names for variables are:

- Names can contain letters, digits, underscores, and dollar signs.
- Names must begin with a letter
- Names can also begin with $ and _
- Names are case sensitive (y and Y are different variables)
- Reserved words (like JavaScript keywords) cannot be used as names

Javascript : Loops

- JavaScript supports loops where you need to perform an action over and over again. JavaScript supports all the necessary loops to ease down the pressure of programming.
    - The while Loop
    - The do...while Loop
    - The for loop

Javascript : while Loop

- The purpose of a while loop is to execute a statement or code block repeatedly as long as an expression is true. Once the expression becomes false, the loop terminates.
- Syntax

```
while (expression) {
  Statement(s) to be executed if expression is true
```

}

```
eg. while (count < 10) {
        document.write("Current Count : " + count + "<br />");
        count++;
    }
```

Javascript : do..while Loop

- The do...while loop is similar to the while loop except that the condition check happens at the end of the loop. This means that the loop will always be executed at least once, even if the condition is false.
- Syntax

```
do {
  Statement(s) to be executed;
} while (expression);
```

```
eg. do {
        document.write("Current Count : " + count + "<br />");
        count++;
    }
    while (count < 5);
```

Javascript : for Loop

- The 'for' loop is the most compact form of looping. It includes the following three important parts −

    - The loop initialization where we initialize our counter to a starting value. The initialization statement is executed before the loop begins.
    - The test statement which will test if a given condition is true or not. If the condition is true, then the code given inside the loop will be executed, otherwise the control will come out of the loop.
    - The iteration statement where you can increase or decrease your counter.
- You can put all the three parts in a single line separated by semicolons.
- Syntax

```
for (initialization; test condition; iteration statement) {
  Statement(s) to be executed if test condition is true
}
```

```
eg. for(count = 0; count < 10; count++) {
        document.write("Current Count : " + count );
        document.write("<br />");       }
```

- JavaScript supports conditional statements which are used to perform different actions based on different conditions.
    - The if statement is the fundamental control statement that allows JavaScript to make decisions and execute statements conditionally.

- Syntax :

```
if (expression) {
  Statement(s) to be executed if expression is true
}
```

eg.

```
if( age > 18 ) {
        document.write("<b>Qualifies for driving</b>");
      }
```

Javascript : If and Switch conditions

- The 'if...else' statement is the next form of control statement that allows JavaScript to execute statements in a more controlled way.Here JavaScript expression is evaluated. If the resulting value is true, the given statement(s) in the ‘if’ block, are executed. If the expression is false, then the given statement(s) in the else block are executed.
- Syntax

```
if (expression) {
  Statement(s) to be executed if expression is true
} else {
  Statement(s) to be executed if expression is false
}
```

Javascript : If and Switch conditions

- The objective of a switch statement is to give an expression to evaluate and several different statements to execute based on the value of the expression. The interpreter checks each case against the value of the expression until a match is found. If nothing matches, a default condition will be used.
- eg

```
while (x < 20) {
        if (x == 5) {
          break;   // breaks out of loop completely
        }
        x = x + 1;
        document.write( x + "<br />");
      }
```

Javascript : break & Continue

- The continue statement tells the interpreter to immediately start the next iteration of the loop and skip the remaining code block. When a continue statement is encountered, the program flow moves to the loop check expression immediately and if the condition remains true, then it starts the next iteration, otherwise the control comes out of the loop.
- eg.

```
while (x < 10) {
          x = x + 1;
```

```
            if (x == 5) {
      continue;   // skip rest of the loop body
    }
    document.write( x + "<br />");
}
```