Dianne Kristel D. Castillo
BSIT BA – 3101

Web Systems & Technologies Reviewer
Module 1

- HTML
    - Hyper Text Markup Language
    - The standard markup language for documents designed to be displayed in a web browser.
    - It can be assisted by technologies such as Cascading Style Sheets and scripting languages such as JavaScript or PHP.
    - The standard markup language for creating Web pages.
- HTML TAGS
    - Used to hold the HTML element
    - Starts with < and ends with >
    - Almost like keywords where every single tag has unique meaning.
    - Used to mark up the start of an HTML element and they are usually enclosed in angle brackets.
    - Most tags must be opened <h1> and closed </h1> in order to function.
- HTML ELEMENTS
    - It holds the content.
    - Whatever written within a HTML tag.
    - It specifies the general content.
- Semantic Element
    - It clearly describes its meaning to both the browser and the developer.
- <!DOCTYPE html>
    - Declaration defines that this document is an HTML5 document.
    - It must only appear once, at the top of the page (before any HTML tags).
- <html>
    - element is the root element of an HTML page
- <head>
    - element contains meta information about the HTML page
- <title>
    - element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
- <body>
    - element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- <style>
    - Tag is used to define style information (CSS) for a document.
    - Inside the <style> element you specify how HTML elements should render in a browser.
    - <style> tag should be inside the <head> tag.
- <div>
    - Tag defines a division or a section in an HTML document.
    - Used as a container for HTML elements - this is then styled

with CSS or manipulated with JavaScript.
    - Tag is easily styled by using the class or id attribute.
- <h1> to <h6>
    - Tags are used to define HTML headings.
    - <h1> defines the most important heading. <h6> defines the least important heading.
    - Only use one <h1> per page - this should represent the main heading/subject for the whole page. Also, do not skip heading levels - start with <h1>, then use <h2>, and so on.
- <p>
    - Tag defines a paragraph.
- <span>
    - Tag is an inline container used to mark up a part of a text, or a part of a document.
    - The <span> tag is easily styled by CSS or manipulated with JavaScript using the class or id attribute.
    - The <span> tag is much like the <div> element, but <div> is a block-level element and <span> is an inline element.
- HTML lists
    - Allow web developers to group a set of related items in lists.
    - The list items will be marked with numbers by default.
    - An ***unordered list*** starts with the <ul> tag. Each list item starts with the <li> tag.
    - An ***ordered list*** starts with the <ol> tag. Each list item starts with the <li> tag.
- HTML Description Lists
    - A description list is a list of terms, with a description of each term.
    - The <dl> tag defines the description list, the <dt> tag defines the term (name), and the <dd> tag describes each term.
- HTML <img>
    - Tag is used to embed an image in a web page.
    - Images are not technically inserted into a web page; images are linked to web pages.
    - The <img> tag creates a holding space for the referenced image.
    - The <img> tag is empty, it contains attributes only, and does not have a closing tag.
    - The <img> tag has two required attributes:
        - src - Specifies the path to the image
        - alt - Specifies an alternate text for the images

Module 2

CLIENT SIDE VS. SERVER SIDE

- Client Side
    - Client-side computers request information while server side serves this information.
- Server Side
    - The server computer serves this information, allowing for a seamless browsing experience.
- Scripting
    - Scripting simply refers to programming languages that automate tasks a human would normally complete.

- Client side scripting
    - runs code like JavaScript on your phone or desktop.
    - refers to scripts that run within your web browser
- Server-side scripting
    - runs on the back-end server
    - It can deliver dynamic content to web pages in response to the client device.
    - Server-side scripting makes use of hosting platforms like GoDaddy, Siteground, and Bluehost, alongside databases like Oracle.
- Client side languages
    - CSS, HTML, JavaScript
    - It also sends requests via web browsers like Firefox, Google Chrome, Internet Explorer, and Safari.
    - This allows you to receive server information within seconds.
- Server side languages
    - PHP, Java, C#
    - Note that JavaScript can be used for client-side and server-side scripting.

- To find out if a language is client or server side, you have to look at where the language is compiled. If this process takes place on your computer, it's a client-side language. And if it's on the server, it's a server-side language.

- Let's say you're loading a standard web page and your browser asks for an HTML page. The server will also give you any associated CSS and JavaScript files. We receive this on the client side, but since CSS and HTML are markup languages, they aren't compiled. JavaScript is a scripting language, so it is gathered on our computer.

JAVASCRIPT

- Server Side Scripting Language
- It is untyped, multi paradigm, functional, event driven.
- It is interpreted by browser's js engine.
- the most popular and widely used client-side scripting language
- designed to add interactivity and dynamic effects to the web pages by manipulating the content returned from a web server.
- is an object-oriented language, and it also has some similarities in syntax to Java programming language.

APPLICATIONS OF JAVASCRIPT PROGRAMMING

- Client side validation
    - This is really important to verify any user input before submitting it to the server and Javascript plays an important role in validting those inputs at front-end itself.
- Manipulating HTML Pages
    - Javascript helps in manipulating HTML page on the fly. This helps in adding and deleting any HTML tag very easily using javascript and modify your HTML to change its look and feel based on different devices and requirements.
- User Notifications
    - You can use Javascript to raise dynamic pop-ups on the webpages to give different types of notifications to your website visitors.

- Back-end Data Loading
    - Javascript provides Ajax library which helps in loading back-end data while you are doing some other processing. This really gives an amazing experience to your website visitors.
- Presentations
    - JavaScript also provides the facility of creating presentations which gives website look and feel. JavaScript provides RevealJS and BespokeJS libraries to build web-based slide presentations.
- Server Applications
    - Node JS is built on Chrome's Javascript runtime for building fast and scalable network applications. This is an event based library which helps in developing very sophisticated server applications including Web Servers.

WHAT YOU CAN DO WITH JAVASCRIPT

- You can modify the content of a web page by adding or removing elements.

- You can change the style and position of the elements on a web page.

- You can monitor events like mouse click, hover, etc. and react to it.

- You can perform and control transitions and animations.

- You can create alert pop-ups to display info or warning messages to the user.

- You can perform operations based on user inputs and display the results.

- You can validate user inputs before submitting it to the server.

JAVASCRIPT ADVANTAGES AND DISADVANTAGES

JAVASCRIPT ADVANTAGES

- ***Fast speed***: JavaScript is executed on the client side that’s why it is very fast.
- ***Easy to learn***: JavaScript is easy to learn. Any one which have basic knowledge of programming can easily lean JavaScript.
- ***Versatilit***y: It refers to lots of skills. It can be used in a wide range of applications.
- ***Browser Compatible***: JavaScript supports all modern browsers. It can execute on any browser and produce same result.
- ***Server Load***: JavaScript reduce the server load as it executes on the client side.
- ***Rich interfaces***: JavaScript provides the drag and drop functionalities which can provide the rich look to the web pages.
- ***Popularity***: JavaScript is a very popular web language because it is used everywhere on the web.
- ***Regular Updates***: JavaScript updated annually by ECMA.

JAVASCRIPT DISADVANTAGES

- ***Code Visibility***: JavaScript code is visible to everyone and this is the biggest disadvantage of JavaScript.
- ***Stop Render***: One error in JavaScript code can stop whole website to render.
- ***No Multiple Inheritances***: JavaScript only support single inheritance.

HOW TO USE JAVASCRIPT IN HTML

- Inline JavaScript
- Internal JavaScript w/ the script tag
- External JavaScript

JAVASCRIPT DISPLAY POSSIBILITIES

JavaScript can "display" data in different ways:

- Writing into an HTML element, using innerHTML.
- Writing into the HTML output using document.write().
- Writing into an alert box, using window.alert().
- Writing into the browser console, using console.log ().

Using innerHTML

- To access an HTML element, JavaScript can use the document.getElementById (id) method.
- The id attribute defines the HTML element. The innerHTML property defines the HTML content:

Using document.write()

- For testing purposes, it is convenient to use document.write():

- Using document.write() after an HTML document is loaded, will delete all existing HTML

Using window.alert()

- You can use an alert box to display data

Using console.log()

- For debugging purposes, you can call the console.log() method in the browser to display data. Putting it all together

1. Create a document in your text editor.
2. Type the code as shown below and save it as js1.html
3. Preview your HTML work using your browser

JavaScript Coding Conventions

Coding conventions are style guidelines for programming. They typically cover:

- Naming and declaration rules for variables and functions.
- Rules for the use of white space, indentation, and comments.

Coding conventions secure quality:

- Improves code readability
- Make code maintenance easier

Always use the same naming convention for all your code. For example:

- Do use camelCasing for variables, function names and function argument names;
- Do use PascalCasing for global variables;
  - Do use UPPERCASE for constants (like PI);
- Do not use under_scores in variable, constants, function arguments or function names;
- Do not use hyphens in JavaScript names.

Naming Conventions

- Do use camelCasing for function names

Spaces Around Operators

- Always put spaces around operators (= + / *), and after commas

Code Indentation

- Always use 4 spaces for indentation of code blocks.
- Do not use tabs (tabulators) for indentation. Text editors interpret tabs differently.

Variables

- Variables are Containers for Storing Data

JavaScript Variables can be declared in 4 ways

- Using var
- Using let
- Using const
- Automatically

When to Use var, let, or const?

- Always declare variables

- Always use const if the value should not be changed
- Always use const if the type should not be changed (Arrays and Objects)
- Only use let if you can't use const
- Only use var if you MUST support old browsers.

JavaScript Identifiers

- An identifier is a name that is given to entities like variables, functions, class, etc.

Rules for Naming JavaScript Identifiers

- Identifier names must start with either a letter, an underscore _, or **the dollar sign $.**
- Identifier names cannot start with numbers.
- JavaScript is case-sensitive. So y and Y are different identifiers.
- Keywords cannot be used as identifier names.

Though you can name identifiers in any way you want, it's a good practice to give a descriptive identifier name.
If you are using an identifier for a variable to store the number of students, it is better to use students or numberOfStudents rather than x or n.

JavaScript Keyword

- Keywords are reserved words that are part of the syntax in the programming language. For example,
  const a = 'hello';
- const is a keyword that denotes that a is a constant.
- Keywords cannot be used to name identifiers.

JS DATA TYPES

- In programming, data types are an important concept. To be able to operate on variables, it is important to know something about the type

Types of JavaScript Operators

- Arithmetic Operator
- Assignment Operators
- Comparison Operators
- String Operators
- Logical Operators
- Bitwise Operators
- Ternary Operators
- Type Operators

Arithmetic Operators

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
**	Exponentiation (ES2016)
	Division
%	Modulus (Division Remainder)
++	Increment
--	Decrement

JavaScript Assignment Operators

- Assignment operators assign values to JavaScript variables.
- The Addition Assignment Operator (+=) adds a value to a variable.

JavaScript Comparison Operators

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 String Addition

- The + can also be used to add (concatenate) strings.
- Adding two numbers, will return the sum, but adding a number and a string will return a string:

JavaScript Date Objects

- By default, JavaScript will use the browser's time zone and display a date as a full text string.

JavaScript Conditional Statements

- Very often when you write code, you want to perform different actions for different decisions.
- You can use conditional statements in your code to do this.
- In JavaScript we have the following conditional statements:
- Use if to specify a block of code to be executed, if a specified condition is true
- Use else to specify a block of code to be executed, if the same condition is false
- Use else if to specify a new condition to test, if the first condition is false
- Use switch to specify many alternative blocks of code to be executed.

The if Statement

- Use the if statement to specify a block of JavaScript code to be executed if a condition is true.

The else Statement

- Use the else statement to specify a block of code to be executed if the condition is false.

The JavaScript Switch Statement

- Use the switch statement to select one of many code blocks to be executed.

The break Keyword

- When JavaScript reaches a break keyword, it breaks out of the switch block.
- This will stop the execution inside the switch block.
- It is not necessary to break the last case in a switch block. The block breaks (ends) there anyway.

The default Keyword

- The default keyword specifies the code to run if there is no case match

JavaScript Loops

- Loops are handy, if you want to run the same code over and over again, each time with a different value. Often this is the case when working with arrays.

Different Kinds of Loops

- for - loops through a block of code a number of times
- for/in - loops through the properties of an object
- for/of - loops through the values of an iterable object
- while - loops through a block of code
- while a specified condition is true
- do/while - also loops through a block of code while a specified condition is true

For Loop

- The for statement creates a loop with 3 optional expressions;

  Expression 1 is executed (one time) before the execution of the code block.

  Expression 2 defines the condition for executing the code block.

  Expression 3 is executed (every time) after the code block has been executed.

The For In Loop

- The JavaScript for in statement loops through the properties of an Object

The For Of Loop

- The JavaScript for of statement loops through the values of an iterable object.
- It lets you loop over iterable data structures such as Arrays, Strings, Maps, NodeLists, and more.

- variable - For every iteration the value of the next property is assigned to the variable. Variable can be declared with const, let, or var.
- iterable - An object that has iterable properties.

The While Loop

- The while loop loops through a block of code as long as a specified condition is true.

The Do While Loop

- The do while loop is a variant of the while loop. This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.

JavaScript Functions

- JavaScript function is a block of code designed to perform a particular task.
- JavaScript function is executed when "something" invokes it (calls it).
- JavaScript function is defined with the function keyword, followed by a name, followed by parentheses ().
- Function names can contain letters, digits, underscores, and dollar signs (same rules as variables).
- The parentheses may include parameter names separated by commas: (parameter1, parameter2, ...)
- The code to be executed, by the function, is placed inside curly brackets: {}
- Function parameters are listed inside the parentheses () in the function definition.
- Function arguments are the values received by the function when it is invoked.
- Inside the function, the arguments (the parameters) behave as local variables.

MODULE 3: JavaScript DOM

Document Object Model (DOM)

- is a programming interface for HTML(HyperText Markup Language) and XML(Extensible markup language) documents
- It defines the logical structure of documents and the way a document is accessed and manipulated.
- It is called a Logical structure because DOM doesn't specify any relationship between objects.
- DOM is a way to represent the webpage in a structured hierarchical way so that it will become easier for programmers and users to glide through the document.
- With DOM, we can easily access and manipulate tags, IDs, classes, Attributes, or Elements of HTML using commands or methods provided by the Document object.
- Using DOM, the JavaScript gets access to HTML as well as CSS of the web page and can also add behavior to the HTML elements.
- Document Object Model is an API that represents and interacts with HTML or XML documents.

Why DOM is required?

- DOM is basically the representation of the same HTML document but in a different format with the use of objects.
- Javascript interprets DOM easily i.e javascript cannot understand the tags(<h1>H</h1>) in HTML document but can understand object h1 in DOM.

Structure of DOM

- DOM can be thought of as a Tree or Forest(more than one tree).
- The term structure model is sometimes used to describe the tree-like representation of a document.
- Each branch of the tree ends in a node, and each node contains objects
- Event listeners can be added to nodes and triggered on an occurrence of a given event.
- One important property of DOM structure models is structural isomorphism:
- if any two DOM implementations are used to create a representation of the same document, they will create the

same structure model, with precisely the same objects and relationships.

Why called an Object Model?

- Documents are modeled using objects, and the model includes not only the structure of a document but also the behavior of a document and the objects of which it is composed like tag elements with attributes in HTML.

Properties of DOM

- **Window Object**
    - Window Object is object of the browser which is always at top of the hierarchy.
    - It is like an API that is used to set and access all the properties and methods of the browser. It is automatically created by the browser.
- **Document object**
    - When an HTML document is loaded into a window, it becomes a document object.
    - The 'document' object has various properties that refer to other objects which allow access to and modification of the content of the web page.
    - If there is a need to access any element in an HTML page, we always start with accessing the 'document' object.
    - Document object is property of window object.
- Form Object
    - It is represented by form tags
- Link Object
    - It is represented by link tags.
- Anchor Object
    - It is represented by a href tags.
- Form Control Elements
    - Form can have many control elements such as text fields, buttons, radio buttons checkboxes, etc.

Methods of Document Object

- write("string")
  Writes the given string on the document.
- getElementById()
  Returns the element having the given id value.
- getElementsByName()
  Returns all the elements having the given name value.
- getElementsByTagName()
  Returns all the elements having the given tag name.
- getElementsByClassName()
  Returns all the elements having the given class name.

What DOM is not?

- The Document Object Model is not a binary description where it does not define any binary source code in its interfaces.
- The Document Object Model is not used to describe objects in XML or HTML whereas the DOM describes XML and HTML documents as objects.
- The Document Object Model is not represented by a set of data structures; it is an interface that specifies object representation.
- The Document Object Model does not show the criticality of objects in documents i.e it doesn't have information about which object in the document is appropriate to the context and which is not.

Levels of DOM

- **Level 0:** Provides a low-level set of interfaces.

- **Level 1:** DOM level 1 can be described in two parts: CORE and HTML.
  CORE provides low-level interfaces that can be used to represent any structured document.

HTML provides high-level interfaces that can be used to represent HTML documents.

- **Level 2:** consists of six specifications: CORE2, VIEWS, EVENTS, STYLE, TRAVERSAL, and RANGE.
  CORE2: extends the functionality of CORE specified by DOM level 1.
  VIEWS: views allows programs to dynamically access and manipulate the content of the document.
  EVENTS: Events are scripts that are either executed by the browser when the user reacts to the web page.
  STYLE: allows programs to dynamically access and manipulate the content of style sheets.
  TRAVERSAL: This allows programs to dynamically traverse the document.
  RANGE: This allows programs to dynamically identify a range of content in the document.

- **Level 3:** consists of five different specifications: CORE3, LOAD and SAVE, VALIDATION, EVENTS, and XPATH.
  CORE3: extends the functionality of CORE specified by DOM level 2.
  LOAD and SAVE: This allows the program to dynamically load the content of the XML document into the DOM document and save the DOM Document into an XML document by serialization.

Fundamentals of PHP

- PHP stands for Hypertext Preprocessor. PHP is a very popular and widely-used open source server- side scripting language to write dynamically generated web pages.
- PHP was originally created by Rasmus Lerdorf in 1994. It was initially known as Personal Home Page.
- In PHP, variable and constant names are case sensitive, while function names are not