Web Technologies

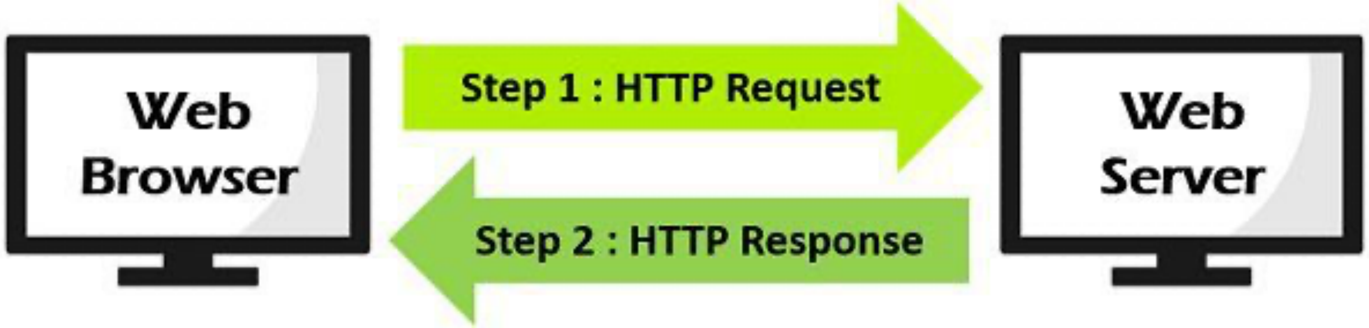
Dr. Angel J. Lopez

Web Documents

Static and Dynamic documents

Static Web Pages

- Static web pages are **simple**
- They are written in the HTML language and stored in web server
- If a server receives a request regarding a web page
  - It sends a response along with the requested web page to the client
  - It does not perform any additional processing
  - It just locates that page on its hard disk and add HTTP headers, and reply back an HTTP response



Static Web Pages

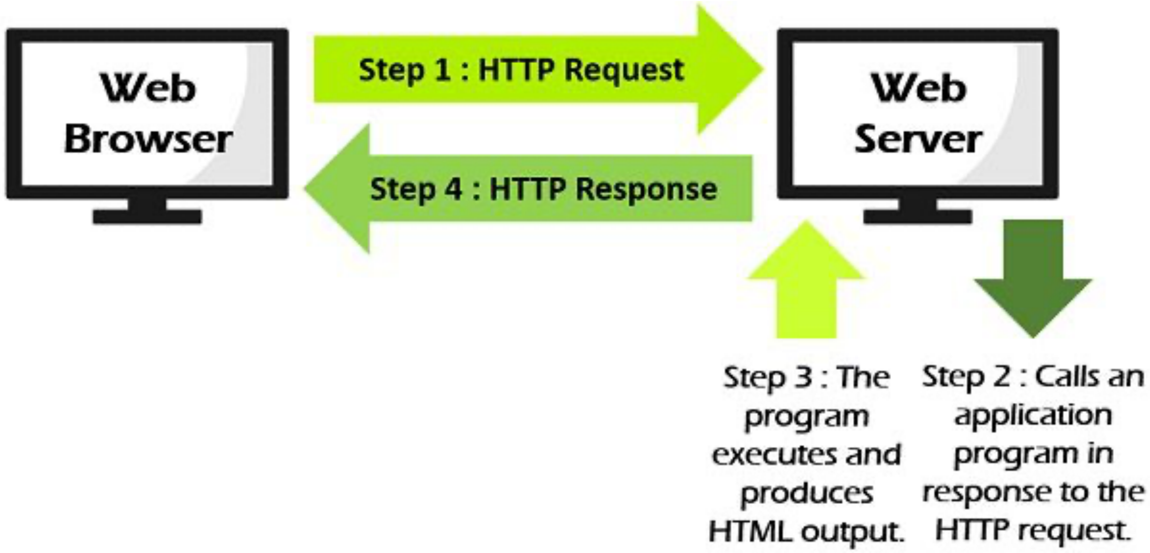
- The fact is, a static web page does not change depending on the request
- A static web page is always the same unless the content is changed physically on the server’s hard disk.
- That is the reason these web pages are known as static web pages.

Dynamic Web pages

- **Dynamic Web pages** provide a solution for the static web pages.
- The dynamic Web page content can vary depending on the number of parameters
- It not just simply send HTML page in response.
  - The web server calls a program located on the hard disk which can access a database, perform transaction procedure, ...
  - If the application program produces HTML output, the web server sends the HTTP response, back to the web browser.
- Dynamic web pages uses client-side scripting or server-side scripting, or both to generate dynamic content

Dynamic Web pages

- The dynamic web pages are employed where the information changes very often


Web Browser
Step 1 : HTTP Request
Step 4 : HTTP Response
Web Server
Step 3 : The program executes and produces HTML output.
Step 2 : Calls an application program in response to the HTTP request.

Static Web Pages vs Dynamic Web Pages

	STATIC WEB PAGES	**DYNAMIC WEB PAGES**
Basic	Static web pages will remain same unless someone changes it manually	Dynamic web pages are behavioral and have the capacity to produce distinctive content for different visitors
Complexity	Simple to design.	Complicated to construct.
Application and web languages used to create web pages	HTML, JavaScript, CSS, etc.	CGI, AJAX, ASP, ASP.NET, etc.
Information change	Occurs rarely	Frequently
Page loading time	Less comparatively	More
Use of Database	Doesn't use databases	A database is used

Client-side Web Programming

- Makes interactive web pages
- Make stuffs work dynamically
- Interact with temporary storage
- Works as an interface between user and server
- Sends requests to the server
- Retrieval of data from Server
- Interact with local storage
- Provides remote access for client server program

Client-side programming languages

- There are many client-side scripting languages, for example:
  - JavaScript
  - VBScript
  - HTML (Structure)
  - CSS (Designing)
  - AJAX
  - jQuery

DOM

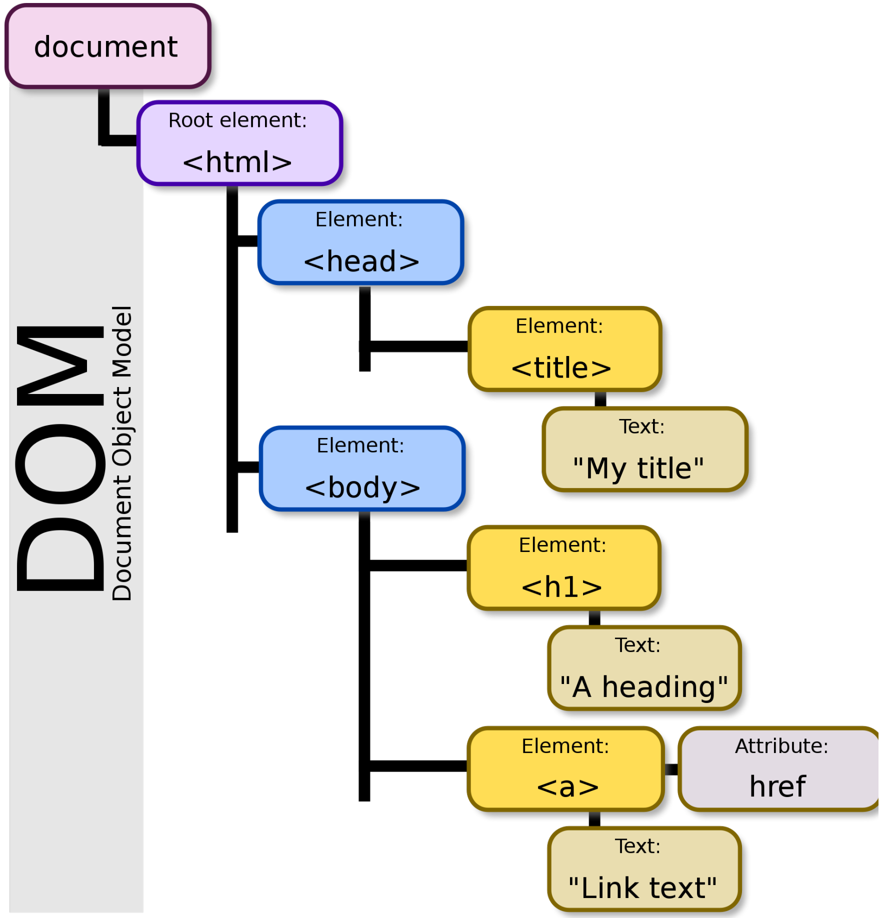
Document Object Model

Document Object Model (DOM)

- The **Document Object Model** (**DOM**) is the data representation of the objects that comprise the structure and content of a document on the web

DOM

- It is a programming interface for HTML and XML documents
- It represents the page so that programs can change the document structure, style, and content
- The DOM represents the document as nodes and objects
  - programming languages can connect to the page



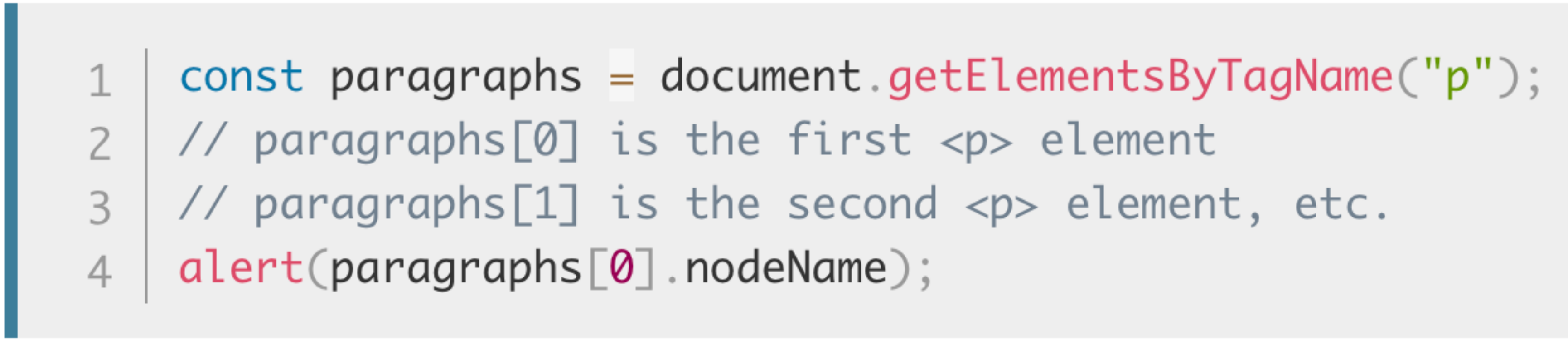
DOM

Let’s consider a Web page:

- It is a document.
- This document can be either displayed in the browser window or as the HTML source. But it is the same document in both cases.
- The Document Object Model (DOM) represents that same document so it can be manipulated.
- The DOM is an object-oriented representation of the web page, which can be modified with a scripting language such as JavaScript.

DOM

- For example, the standard DOM specifies that the `getElementsByTagName` method in the code below must return a list of all the `<p>` elements in the document:



```
const paragraphs = document.getElementsByTagName("p");
// paragraphs[0] is the first <p> element
// paragraphs[1] is the second <p> element, etc.
alert(paragraphs[0].nodeName);
```

DOM

- All of the properties, methods, and events available for manipulating and creating web pages are organized into objects
- For example:
  - The *document* object that represents the document itself
  - The *table* object that implements the special `HTMLTableElement` DOM interface for accessing HTML tables
- This documentation provides an object-by-object reference to the DOM

JavaScript

- JavaScript (or "JS") is a programming language
- Java Script is used most often for dynamic **client-side** scripts on webpages
- It is also often used on the **server-side**, using packages such as Node.js
- In the browser, it enables developers to manipulate webpage content through the DOM
- It also manipulates data with AJAX and IndexedDB, draw graphics with canvas, interact with the device running the browser through various APIs, and more
- JavaScript is one of the world's most commonly-used languages
  - Performance improvement

Java Script

- JavaScript *uses* the DOM to access the document and its elements
- The **DOM is not a programming language**, but without it, the JavaScript language wouldn't have any model or notion of:
  - web pages, HTML documents, XML documents, and their component parts (e.g. elements)
- Every element in a document is part of the document object model
  - the document as a whole, the head, tables within the document, table headers, text within the table cells
- They can all be **accessed and manipulated using the DOM and a scripting language** like JavaScript

Accessing the DOM

- When you create a script, you can immediately begin using the API for the document or window elements
  - to manipulate the document itself
  - to get at the children of that document

Accessing the DOM

- Your DOM programming may be something as simple as the following, which displays an alert message by using the `alert()`function from the window object

- See examples on the repository:

  - path: unit02/activity05/DOM

Data types

- `Document`
  - This object is the root document object itself
- `Node`
  - Every object located within a document is a `node`
- `Element`
  - The `element` type is based on node
  - It refers to an `element` or a node of type `element`

Data types

- `NodeList`
  - A `nodeList` is an array of elements
  - Items in a `nodeList` are accessed by index
    - list.item(1)
    - list[1]
- `Attribute`
  - Attributes are nodes in the DOM just like elements are
- `NamedNodeMap`
  - It is like an array, but the items are accessed by name or index

Activity

Client-side programming

- Create a Web page using client-side programming
- Examples:
  - https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Examples

- Deliverable:
  - Submit your Web page (html file) to Sidweb