Introduction TO Dynamic WEB PAGE

Web Application Development (SRM Institute of Science and Technology)

INTRODUCTION TO DYNAMIC WEB PAGE

A server-side dynamic web page is a web page whose construction is controlled by an application server processing server-side scripts. In server-side scripting, parameters determine how the assembly of every new web page proceeds, including the setting up of more client-side processing.

A client-side dynamic web page processes the web page using HTML scripting running in the browser as it loads. JavaScript and other scripting languages determine the way the HTML in the received page is parsed into the Document Object Model, or DOM, that represents the loaded web page. The same client-side techniques can then dynamically update or change the DOM in the same way. Even though a web page can be dynamic on the client-side, it can still be hosted on a static hosting service such as GitHub Pages or Amazon S3 as long as there isn't any server-side code included.

A dynamic web page is then reloaded by the user or by a computer program to change some variable content. The updating information could come from the server, or from changes made to that page's DOM. This may or may not truncate the browsing history or create a saved version to go back to, but a dynamic web page update using AJAX technologies will neither create a page to go back to, nor truncate the web browsing history forward of the displayed page. Using AJAX, the end user gets one dynamic page managed as a single page in the web browser while the actual web content rendered on that page can vary. The AJAX engine sits only on the browser requesting parts of its DOM, the DOM, for its client, from an application server.

DHTML is the umbrella term for technologies and methods used to create web pages that are not static web pages, though it has fallen out of common use since the popularization of AJAX, a term which is now itself rarely used. Client-side-scripting, server-side scripting, or a combination of these make for the dynamic web experience in a browser.

Classical hypertext navigation, with HTML or XHTML alone, provides "static" content, meaning that the user requests a web page and simply views the page and the information on that page.

However, a web page can also provide a "live", "dynamic", or "interactive" user experience. Content (text, images, form fields, etc.) on a web page can change, in response to different contexts or conditions.

There are two ways to create this kind of effect:

- Using client-side scripting to change interface behaviors within a specific web page, in response to mouse or keyboard actions or at specified timing events. In this case the dynamic behavior occurs within the presentation.
- Using server-side scripting to change the supplied page source between pages, adjusting the sequence or reload of the web pages or web content supplied to the browser. Server responses may be determined by such conditions as data in a posted HTML form, parameters in the URL, the type of browser being used, the passage of time, or a database or server state.

Web pages that use client-side scripting must use presentation technology broadly called rich interfaced pages. Client-side scripting languages like JavaScript or ActionScript, used for Dynamic HTML (DHTML) and Flash technologies respectively, are frequently used to orchestrate media types (sound, animations, changing text, etc.) of the presentation. The scripting also allows use of remote scripting, a technique by which the DHTML page requests additional information from a server, using a hidden Frame, XMLHttpRequests, or a web service.

Web pages that use server-side scripting are often created with the help of server-side languages such as PHP, Perl, ASP, ASP.NET, JSP, ColdFusion and other languages. These server-side languages typically use the Common Gateway Interface (CGI) to produce dynamic web pages. These kinds of pages can also use, on the client-side, the first kind (DHTML, etc.).

A program running on a web server (server-side scripting) is used to generate the web content on various web pages, manage user sessions, and control workflow. Server responses may be determined by such conditions as data in a posted HTML form, parameters in the URL, the type of browser being used, the passage of time, or a database or server state.

Such web pages are often created with the help of server-side languages such as ASP, ColdFusion, Go, JavaScript, Perl, PHP, Ruby, Python, WebDNA and other languages, by a support server that can run on the same hardware as the web server. These server-side languages often use the Common Gateway Interface (CGI) to produce dynamic web pages. Two notable exceptions are ASP.NET, and JSP, which reuse CGI concepts in their APIs but actually dispatch all web requests into a shared virtual machine.

The server-side languages are used to embed tags or markers within the source file of the web page on the web server. When a user on a client computer requests that web page, the web server interprets these tags or markers to perform actions on the server. For example, the server may be instructed to insert information from a database or information such as the current date.

Dynamic web pages are often cached when there are few or no changes expected and the page is anticipated to receive considerable amount of web traffic that would create slow load times for the server if it had to generate the pages on the fly for each request.

Client-side scripting is changing interface behaviors within a specific web page in response to mouse or keyboard actions, or at specified timing events. In this case, the dynamic behavior occurs within the presentation. The client-side content is generated on the user's local computer system.

Such web pages use presentation technology called rich interfaced pages. Client-side scripting languages like JavaScript or ActionScript, used for Dynamic HTML (DHTML) and Flash technologies respectively, are frequently used to orchestrate media types (sound, animations, changing text, etc.) of the presentation. Client-side scripting also allows the use of remote scripting, a technique by which the DHTML page requests additional information from a server, using a hidden frame, XMLHttpRequests, or a Web service.

The first widespread use of JavaScript was in 1997, when the language was standardized as ECMAScript and implemented in Netscape 3.

Example

The client-side content is generated on the client's computer. The web browser retrieves a page from the server, then processes the code embedded in the page (typically written in JavaScript) and displays the retrieved page's content to the user.

The innerHTML property (or write command) can illustrate the client-side dynamic page generation: two distinct pages, A and B, can be regenerated (by an "event response dynamic") as document.innerHTML =
A and document.innerHTML = B; or "on load dynamic"
by document.write(A) and document.write(B).

All of the client and server components that collectively build a dynamic web page are called a web application. Web applications manage user interactions, state, security, and performance.

Ajax uses a combination of both client-side scripting and server-side requests. It is a web application development technique for dynamically interchanging content, and it sends requests to the server for data in order to do so. The server returns the requested data which is then processed by a client-side script. This technique can reduce server load time because the client does not request the entire webpage to be regenerated by the server's language parser; only the content that will change is transmitted. Google Maps is an example of a web application that uses Ajax techniques.

A web client, such as a web browser, can act as its own server, accessing data from many different servers, such as Gopher, FTP, NNTP (Usenet) and HTTP, to build a page. HTTP supports uploading documents from the client back to the server. There are several HTTP methods for doing this.

The above is a brief about Dynamic Web Page. Watch this space for more updates on the latest trends in Technology.

A server-side dynamic web page is a web page whose construction is controlled by an application server processing server-side scripts. In server-side scripting, parameters determine how the assembly of every new web page proceeds, including the setting up of more client-side processing.

A client-side dynamic web page processes the web page using HTML scripting running in the browser as it loads. JavaScript and other scripting languages determine the way the HTML in the received page is parsed into the Document Object Model, or DOM, that represents the loaded web page. The same client-side techniques can then dynamically update or change the DOM in the same way. Even though a web page can be dynamic on the client-side, it can still be hosted on a static hosting service such as GitHub Pages or Amazon S3 as long as there isn't any server-side code included.

A dynamic web page is then reloaded by the user or by a computer program to change some variable content. The updating information could come from the server, or from changes made to that page's DOM. This may or may not truncate the browsing history or create a saved version to go back to, but a dynamic web page update using AJAX technologies will neither create a page to go back to, nor truncate the web browsing history forward of the displayed page. Using AJAX, the end user gets one dynamic page managed as a single page in the web browser while the actual web content rendered on that page can vary. The AJAX engine sits only on the browser requesting parts of its DOM, the DOM, for its client, from an application server.

DHTML is the umbrella term for technologies and methods used to create web pages that are not static web pages, though it has fallen out of common use since the popularization of AJAX, a term which is now itself rarely used. Client-side-scripting, server-side scripting, or a combination of these make for the dynamic web experience in a browser.

Classical hypertext navigation, with HTML or XHTML alone, provides "static" content, meaning that the user requests a web page and simply views the page and the information on that page.

However, a web page can also provide a "live", "dynamic", or "interactive" user experience. Content (text, images, form fields, etc.) on a web page can change, in response to different contexts or conditions.

There are two ways to create this kind of effect:

- Using client-side scripting to change interface behaviors within a specific web page, in response to mouse or keyboard actions or at specified timing events. In this case the dynamic behavior occurs within the presentation.
- Using server-side scripting to change the supplied page source between pages, adjusting the sequence or reload of the web pages or web content supplied to the browser. Server responses may be determined by such conditions as data in a posted HTML form, parameters in the URL, the type of browser being used, the passage of time, or a database or server state.

Web pages that use client-side scripting must use presentation technology broadly called rich interfaced pages. Client-side scripting languages like JavaScript or ActionScript, used for Dynamic HTML (DHTML) and Flash technologies respectively, are frequently used to orchestrate media types (sound, animations, changing text, etc.) of the presentation. The scripting also allows use of remote scripting, a technique by which the DHTML page requests additional information from a server, using a hidden Frame, XMLHttpRequests, or a web service.

Web pages that use server-side scripting are often created with the help of server-side languages such as PHP, Perl, ASP, ASP.NET, JSP, ColdFusion and other languages. These server-side languages typically use the Common Gateway Interface (CGI) to produce dynamic web pages. These kinds of pages can also use, on the client-side, the first kind (DHTML, etc.).

A program running on a web server (server-side scripting) is used to generate the web content on various web pages, manage user sessions, and control workflow. Server responses may be determined by such conditions as data in a posted HTML form, parameters in the URL, the type of browser being used, the passage of time, or a database or server state.

Such web pages are often created with the help of server-side languages such as ASP, ColdFusion, Go, JavaScript, Perl, PHP, Ruby, Python, WebDNA and other languages, by a support server that can run on the same hardware as the web server. These server-side languages often use the Common Gateway Interface (CGI) to produce dynamic web pages. Two notable exceptions are ASP.NET, and JSP, which reuse CGI concepts in their APIs but actually dispatch all web requests into a shared virtual machine.

The server-side languages are used to embed tags or markers within the source file of the web page on the web server. When a user on a client computer requests that web page, the web server interprets these tags or markers to perform actions on the server. For example, the server may be instructed to insert information from a database or information such as the current date.

Dynamic web pages are often cached when there are few or no changes expected and the page is anticipated to receive considerable amount of web traffic that would create slow load times for the server if it had to generate the pages on the fly for each request.

Client-side scripting is changing interface behaviors within a specific web page in response to mouse or keyboard actions, or at specified timing events. In this case, the dynamic behavior occurs within the presentation. The client-side content is generated on the user’s local computer system.

Such web pages use presentation technology called rich interfaced pages. Client-side scripting languages like JavaScript or ActionScript, used for Dynamic HTML (DHTML) and Flash technologies respectively, are frequently used to orchestrate media types (sound, animations, changing text, etc.) of the presentation. Client-side scripting also allows the use of remote scripting, a technique by which the DHTML page requests additional information from a server, using a hidden frame, XMLHttpRequests, or a Web service.

The first widespread use of JavaScript was in 1997, when the language was standardized as ECMAScript and implemented in Netscape 3.

Example

The client-side content is generated on the client's computer. The web browser retrieves a page from the server, then processes the code embedded in the page (typically written in JavaScript) and displays the retrieved page's content to the user.

The innerHTML property (or write command) can illustrate the client-side dynamic page generation: two distinct pages, A and B, can be regenerated (by an "event response dynamic") as document.innerHTML =
A and document.innerHTML = B; or "on load dynamic"
by document.write(A) and document.write(B).

All of the client and server components that collectively build a dynamic web page are called a web application. Web applications manage user interactions, state, security, and performance.

Ajax uses a combination of both client-side scripting and server-side requests. It is a web application development technique for dynamically interchanging content, and it sends requests to the server for data in order to do so. The server returns the requested data which is then processed by a client-side script. This technique can reduce server load time because the client does not request the entire webpage to be regenerated by the server's language parser; only the content that will change is transmitted. Google Maps is an example of a web application that uses Ajax techniques.

A web client, such as a web browser, can act as its own server, accessing data from many different servers, such as Gopher, FTP, NNTP (Usenet) and HTTP, to build a page. HTTP supports uploading documents from the client back to the server. There are several HTTP methods for doing this.

The above is a brief about Dynamic Web Page. Watch this space for more updates on the latest trends in Technology.

SERVER-SIDE WEBSITE PROGRAMMING

Web browsers communicate with web servers using
the **H**yper**T**ext **T**ransfer **P**rotocol (HTTP). When you click a link on a web page, submit a form, or run a search, an **HTTP request** is sent from your browser to the target server.

The request includes a URL identifying the affected resource, a method that defines the required action (for example to get, delete, or post the resource), and may include additional information encoded in URL parameters (the field-value pairs sent via a query string), as POST data (data sent by the HTTP POST method), or in associated cookies.

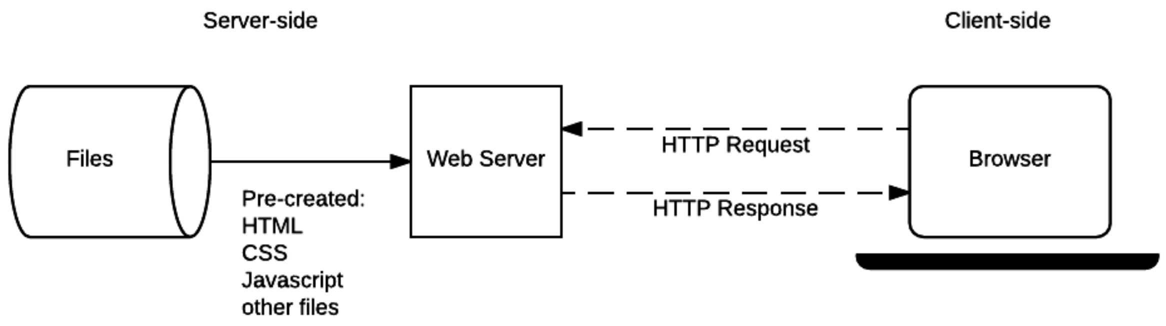
Web servers wait for client request messages, process them when they arrive, and reply to the web browser with an **HTTP response** message. The response contains a status line indicating whether or not the request succeeded (e.g. "HTTP/1.1 200 OK" for success).

The body of a successful response to a request would contain the requested resource (e.g. a new HTML page, or an image, etc...), which could then be displayed by the web browser.

Static sites

The diagram below shows a basic web server architecture for a static site (a static site is one that returns the same hard-coded content from the server whenever a particular resource is requested). When a user wants to navigate to a page, the browser sends an HTTP "GET" request specifying its URL.

The server retrieves the requested document from its file system and returns an HTTP response containing the document and a success status (usually 200 OK). If the file cannot be retrieved for some reason, an error status is returned (see client error responses and server error responses).



Dynamic sites

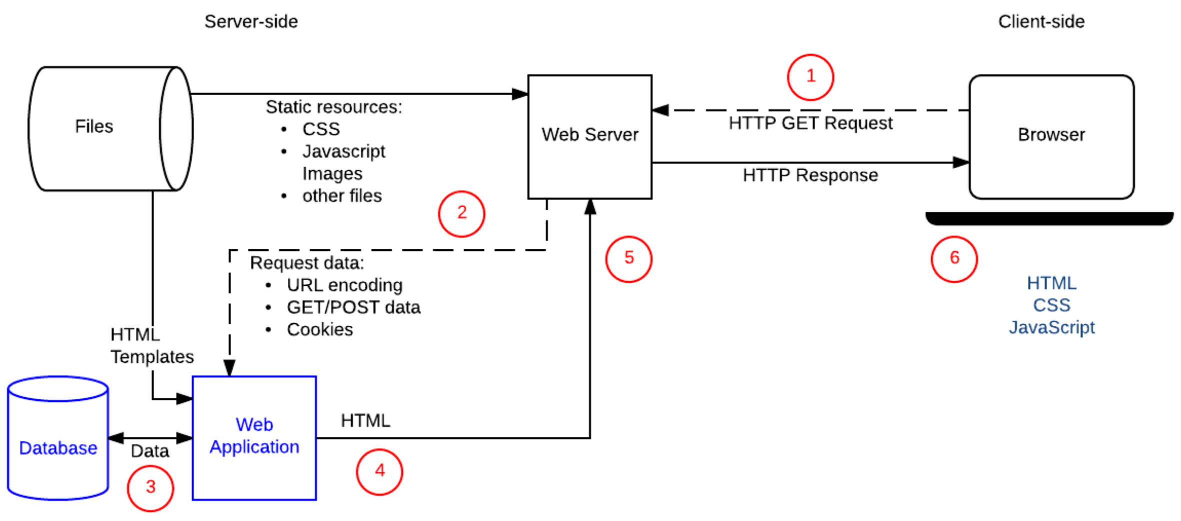
A dynamic website is one where some of the response content is generated dynamically, only when needed. On a dynamic website HTML pages are normally created by inserting data from a database into placeholders in HTML templates (this is a much more efficient way of storing large amounts of content than using static websites).

A dynamic site can return different data for a URL based on information provided by the user or stored preferences and can perform other operations as part of returning a response (e.g. sending notifications).

Most of the code to support a dynamic website must run on the server. Creating this code is known as "**server-side programming**" (or sometimes "**back-end scripting**").

The diagram below shows a simple architecture for a dynamic website. As in the previous diagram, browsers send HTTP requests to the server, then the server processes the requests and returns appropriate HTTP responses.

Requests for static resources are handled in the same way as for static sites (static resources are any files that don't change —typically: CSS, JavaScript, Images, pre-created PDF files etc).



Requests for dynamic resources are instead forwarded (2) to server-side code (shown in the diagram as a Web Application). For "dynamic requests" the server interprets the request, reads required information from the database (3), combines the retrieved data with HTML templates (4), and sends back a response containing the generated HTML (5,6).

server-side and client-side programming the same?

Let's now turn our attention to the code involved in server-side and client-side programming. In each case, the code is significantly different:

- They have different purposes and concerns.

- They generally don't use the same programming languages (the exception being JavaScript, which can be used on the server- and client-side).
- They run inside different operating system environments.

Code running in the browser is known as **client-side code** and is primarily concerned with improving the appearance and behavior of a rendered web page. This includes selecting and styling UI components, creating layouts, navigation, form validation, etc. By contrast, server-side website programming mostly involves choosing which content is returned to the browser in response to requests. The server-side code handles tasks like validating submitted data and requests, using databases to store and retrieve data and sending the correct data to the client as required.

Client-side code is written using HTML, CSS, and JavaScript — it is run inside a web browser and has little or no access to the underlying operating system (including limited access to the file system).

Web developers can't control what browser every user might be using to view a website — browsers provide inconsistent levels of compatibility with client-side code features, and part of the challenge of client-side programming is handling differences in browser support gracefully.

Server-side code can be written in any number of programming languages — examples of popular server-side web languages include PHP, Python, Ruby, C#, and JavaScript (NodeJS). The server-side code has full access to the server operating system and the developer can choose what programming language (and specific version) they wish to use.

Developers typically write their code using **web frameworks**. Web frameworks are collections of functions, objects, rules and other code constructs designed to solve common problems, speed up development, and simplify the different types of tasks faced in a particular domain.

Again, while both client and server-side code use frameworks, the domains are very different, and hence so are the frameworks. Client-side web frameworks simplify layout and presentation tasks while server-side web frameworks provide a lot of "common" web server functionality that you might otherwise have to implement yourself (e.g. support for sessions, support for users and authentication, easy database access, templating libraries, etc.).

Server-side Programming :

It is the program that runs on server dealing with the generation of content of web page.
1) Querying the database
2) Operations over databases
3) Access/Write a file on server.
4) Interact with other servers.
5) Structure web applications.
6) Process user input. For example if user input is a text in search box, run a search algorithm on data stored on server and send the results.

Examples :
The Programming languages for server-side programming are :
1) PHP
2) C++
3) Java and JSP
4) Python
5) Ruby on Rails

Client-side Programming :

It is the program that runs on the client machine (browser) and deals with the user interface/display and any other processing that can happen on client machine like reading/writing cookies.

1) Interact with temporary storage
2) Make interactive web pages
3) Interact with local storage
4) Sending request for data to server
5) Send request to server
6) work as an interface between server and user

The Programming languages for client-side programming are :
1) Javascript
2) VBScript
3) HTML
4) CSS
5) AJAX