**Web Page and Website**

The difference between a website and a web page is that a website is a collection of web pages which are grouped together, and a web page is a smaller part of a larger website usually containing more specific information.

A website is under 1 domain (such as coolwebsite.com). For example if there is a company that owns abccompany.com then this website will have several Webpages like Home, About Us, Contact Us, Testimonials, Products, Services, FAQ’s, and others. All of these pages together make up a Website.

**Static Page and Dynamic Page**

There are basically two main types of website - static and dynamic.

A static site is one that is usually written in plain HTML and what is in the code of the page is what is displayed to the user.

Static Web pages display the exact same information whenever anyone visits it. Static Web pages do not have to be simple plain text. They can feature detailed multimedia design and even videos. However, every visitor to that page will be greeted by the exact same text, multimedia design or video every time he visits the page until you alter that page's source code.

A dynamic site is one that is written using a server-side scripting language such as PHP, ASP, JSP, or Coldfusion.

Dynamic Web pages are capable of producing different content for different visitors from the same source code file. The website can display different content based on what operating system or browser the visitor is using, whether she is using a PC or a mobile device, or even the source that referred the visitor. A dynamic Web page is not necessarily better than a static Web page. The two simply serve different purposes.

|  |  |
| --- | --- |
| Static sites - advantages  Flexibility is the main advantage of a static site - every page can be different if desired, to match the layout to different content, and the designer is free to put in any special effects that a client may ask for in a unique way on different pages. This allows theming - for instance an author may want a different theme for a different book and associated pages or perhaps for a series of books, in order to match the cover designs or the context of the stories.  Cost is generally lower up-front than a dynamic site. | Dynamic sites - advantages  The main advantages of dynamic sites are that by connecting them to databases you can easily pull in information in an organized and structured way to create product pages or categories of related products sorted in a variety of different ways depending on how the user wants to view them.  This ability to connect to a database means that you can also create a content management system - an interface which allows the client to input and manage data via a web-based series of administration pages. That content can be text for their pages and images to go along with the text, or items in their product range with categories, specifications, short and long descriptions, images, etc. In both these cases it can be as simple or as complex as the client requires.  There are little or no ongoing costs unless there is a change in the basic design or an extra capability added. |
| Static sites - disadvantages  The main problem with any static site appears when you wish to update the content. Unless you are conversant with HTML and the design methods used in the site then you have to go back to the designer to have any content changes made. This may be perfectly ok when a new page is required which needs design input, but if all you want to do is change some text then it can be a nuisance for both client and designer.  The second main problem is scalability. If you wish to sell products on your site and you have a lot of them then you may have to construct individual pages for each one, which can take considerable time, effort and cost.  Costs - there are ongoing costs for updating the content. | Dynamic sites - disadvantages  The design of a dynamic site is more fixed than a static one because many of the pages are essentially a template into which data and content is poured to create multiple pages of a similar type. So for instance all your product pages will be essentially the same page layout with different data being displayed. While some customization capability can be built in it is usually quite limited, such a selecting from a set of pre-defined options. Individual layout changes to particular pages are not usually possible.  Costs are higher initially than for a static site, and additional functionality may also cost more, particularly if it's something that wasn't envisaged originally and requires re-writing of the core code or database. |

**Script**

A script is a set of instructions. For Web pages they are instructions either to the Web browser (client-side scripting) or to the server (server-side scripting). Scripts provide change to a Web page.

Most websites make use of both a client side and a server side language. Although there are things both can do, there are some things which can only be done server side, and there are some things which can only be done client side.

Front-end scripting is good for anything that requires user interaction, such as a making the page interactive, displaying or sorting data.

Back-end scripting is good for anything that requires dynamic data to be loaded, such as handling log in, personal information and preferences and provides the specific data which the user wants.

**Client Side and Server Side scripting**

|  |  |
| --- | --- |
| Client Side  Client-side scripts run on the client side, and is done in languages that can be executed by the browser, such as JavaScript  The reason JavaScript is called a client side language is because it runs scripts on your computer after you’ve loaded a web page.  <script>  document.getElementById('hello').innerHTML = 'Hello';  </script> | Server Side scripting  Server-side scripts run on the server, using languages supported by the server such as Java, PHP, C#, etc.  A server side or back-end language runs its scripts before the HTML is loaded, not after.  <h1 id="hello"><?php echo 'Hello'; ?></h1> |
| Uses  •Make interactive webpages.  •Make stuff happen dynamically on the web page.  •Interact with temporary storage, and local storage (Cookies, localStorage).  •Send requests to the server, and retrieve data from it.  •Provide a remote service for client-side applications, such as software registration, content delivery, or remote multi-player gaming. | Uses  •Process user input.  •Display pages.  •Structure web applications.  •Interaction with SQL, files, etc. |

**Javascript and JQuery**

Javascript

JavaScript is a scripting language that is used inside your web browser.

Unfortunately, JavaScript still has some issues with cross-browser compatibility due to poor JavaScript implementation practices on the part of web browser developers.

function changeBackground(color) {

document.body.style.background = color;

}

onload="changeBackground('red');"

jQuery

jQuery is a library/ framework written in Javascript.

It simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development.

You can code most common JS actions using jQuery with fewer lines of code. jQuery is a Cross-browser, it means that it is compatible with multiple Web browsers.

$('body').css('background', '#ccc');

**AJAX**

AJAX stands for Asynchronous JavaScript and XML. We use AJ to do asynchronous things such updating a page, making actions, etc.

Ajax is the concept of the client calling the server directly to interact with server objects like a database, without a postback involved.

In simpler meaning, AJAX is all about updating parts of a webpage without having to reload the entire thing.

Note: “When you execute something synchronously, you wait for it to finish before moving on to another task. When you execute something asynchronously, you can move on to another task before it finishes.”

**Pros of Using AJAX**

•Improved User Experience — the enriched user experience provided by AJAX is the foremost benefit. AJAX allows webpages to update serially by exchanging a small amount of data with the server. This way it is possible to update parts of a webpage, without reloading the whole page. Classic webpages must reload the entire page and are cumbersome. AJAX increases the browser’s performance and facilitates faster browsing speed thereby providing a responsive user experience.

•Enhanced User Productivity — The AJAX library provides object-oriented helper functions that dramatically increase the productivity while decreasing frustration. In addition, a well-configured ASP.NET application has its own data access layer and business layer. Finally, the “robust” ASP.NET application includes a UI layer where server side operations are performed. If you already have included these features, AJAX only needs an extra layer of AJAX-specific services and some enrichment on client features. This way the deployment cost is reduced and the productivity of the user can be enhanced. Popular websites like Amazon, Google, Yahoo, etc. also incorporate AJAX in their development.

•Reduced Bandwidth Usage and Increased speed — AJAX uses client-side scripting to communicate with the web server and exchange data using JavaScript. Using AJAX, you can cut down on network load and bandwidth usage and retrieve only the data that is required to give you faster interfaces and better responsive times. Response time is faster, hence performance and speed are increased.

•Increased Compatibility — AJAX can be compatible with ASP.NET, J2EE, PHP, or any languages. It almost supports all popular browsers such as Internet Explorer 5 and above, Mozilla Firefox 1.0 and above, Apple Safari 1.2 and above, Opera 7.6 and above, and RockMelt.

•Supports Asynchronous Processing — Asynchronous data retrieval can be done by using XmlHttpRequest, the backbone of AJAX applications. Hence, requests are handled effectively and dynamic content loading is brought to higher heights by improving the performance considerably.

•Reduced server hits and network load — Atlas, an older form of Microsoft AJAX library, is a framework that integrates the Client-side JavaScript library and is easily available and can be used with ASP.NET to develop Ajax applications. It has cross-browser support and exposes object-oriented APIs, which can be used to develop web applications that minimize server hit/network load and perform asynchronous processing.

•Easier Navigation — AJAX applications can be built to allow easy transition between Webpages to the users instead of using conventional back and forward buttons on a browser.

**Cons of Using AJAX**

•Browser Incompatibility — AJAX highly depends on JavaScript which is implemented differently for various browsers. This turns out to be a hurdle especially when AJAX has to work across many browsers. Browsers which do not support JavaScript or have the JavaScript option disabled will not be able to use its functionality. Due to the AJAX’s dependency on JavaScript, it is not suitable for designing mobile applications. The Back button of your web browser does not work as expected.

•Insecurity — the webpage can be difficult to debug, increases the code size of your webpage, and makes your webpage prone to severe security threats.

•Increased load on Web Server — the load can be increased depending on the user if you are adding an auto-update type that hits the server every few seconds.

**Angular js, Node js, React js**

All the three JavaScript frameworks mentioned above are high performing, advanced and widely used worldwide. It all depends on your business needs and custom app goals which framework you want to choose.

AngularJS is fully featured framework than ReactJS, which is not really a framework but a library. With ReactJS, you write less code and do more. In fact, React is better than AngularJS when it comes to their performance. It is because of React’s implementation of virtual DOM. However, AngularJS has a large support community and following while React is just getting started.

On the other hand, Nodejs is simply a JavaScript runtime which is lightweight and fast. It is mainly created to build fast and scalable network applications.

|  |  |  |
| --- | --- | --- |
| [Image result for angular JS](https://www.google.com/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwimhsDrxtfaAhXB2LwKHYVcDsAQjRx6BAgAEAU&url=https%3A%2F%2Fwww.w3schools.com%2Fangular%2Fdefault.asp&psig=AOvVaw3Km5mqEKDv5Nk0gfMDsrvV&ust=1524818777582489) |  |  |
| Angular Js | **Node Js** | **React Js** |
| Uses MVVC pattern | Event driven architecture | Uses Controller View pattern |
| Designed for single page application development | For highly concurrent server side design | Best for building scalable network applications |
| Open source javascript framework | Run time environment | Open source javascript library |
| Used for front end tasks | It’s a javascript runtime | Can create view in client side and server side |

**Angular Js**

In 2009, Google launched its open source client side web framework called as AngularJS. It was developed for angular developers aiming at resolving issues in single page application creation.

It extends HTML vocabulary for your web application. Due to its extensibility feature, it can work well with other libraries. Moreover, it has a large support community.

Top 3 reasons to choose AngularJS:

|  |  |  |
| --- | --- | --- |
| 1) User interface  AngularJS uses HTML to define a web app’s user interface because HTML is a declarative language and less brittle to recognize. The overall game is of attributes you use in your HTML where these attributes define which controller will be used for which element.  It simplifies your web development process and you just define what you want. However, AngularJS takes care of all the dependencies. | 2) Flexibility  The use of filters and directives makes it a flexible for web app development. Directives bring functionality to HTML rather than manipulating DOM. In fact, if you put DOM manipulation code into directives, you can easily separate them out of your MVC app.  Always remember: Every DOM manipulations should be performed by directives.  On the other hand, Filters are designed as standalone functions which are separate from your app but take care of data transformations. Filters are used to create a sortable HTML without writing any JavaScript. | 3) Testing  AngularJS is completely linked to Dependency Injection and all your controllers depend on the DI. AngularJS unit testing is done by injecting mock data into your controller and then measuring the output with behavior.  This provides a new way of testing web apps in which individual test pages are created which further calls one component and interacts with it to see if it works. |

Pros and Cons

|  |  |
| --- | --- |
| Advantages  – Data synchronization is automatically done between model view and components  – Easily testable framework  – Inbuilt dependency injection subsystem  – Customized Document Object Model(DOM) can be created easily.  – Simple routing  – Amazing UI design  – Angular Data binding  – It is an extension to HTML syntax and also used to create reusable components by directives.  – It gives robust template building solutions  – Great Angular libraries | Drawbacks  – DOM elements claim to performance issues  – Scopes are hard to debug  – Limited Routing  – Angular becomes slow with pages embedding interactive elements  – Third party integration is complex  – Steep learning curve |

**Node Js**

ReactJS is considered as an open-source JavaScript library rather than a framework. It is mainly used for building amazing user interfaces with a great focus on rendering performance. In MVC(Model View Controller) architecture, React is more dependent on “V”.

It quickly gained a good reputation when it was launched first. It was built with the aim of resolving issues in JavaScript frameworks related to efficient rendering of large datasets.

Top 3 reasons to choose ReactJS:

|  |  |  |
| --- | --- | --- |
| 1) SEO Effective  Search engines find it difficult to read JavaScript heavy applications even after having improvements in that area. So this is one of the big issues that come with JavaScript frameworks.  But ReactJS has beaten up this. You can easily run ReactJS on the server and the virtual DOM will be rendered then which further return to the browser as a regular web page. No hard tricks are required for this. | 2) Components in ReactJS  PolymerJS and Shadow DOM has already created a lot of buzzes which are typically used to create customizable elements, self-contained elements that you can easily import into your project.  But RecatJS makes you able to create your own components which you can later combine, reuse or nest your core content. So it doesn’t make use of Shadow DOM or PolymerJS. | 3) Great efficiency  ReactJS creates its own virtual DOM where your components actually live. It takes care of all the changes to made in the DOM and updates in the DOM tree also. This makes it a flexible approach to gain a good performance. Hence, it discards costly DOM operations and regularly makes updates efficiently. |

Pros and Cons

|  |  |
| --- | --- |
| Advantages  – It has smooth interface designs and easy learning API  – It consists of React native library  – You can reuse code with React JS  – JavaScript debugging is easy  – High performance as compared to other JS frameworks  – It is fully component based architecture  – JSX(JavaScript extension syntax) allows quoting of HTML to render sub components  – Importing components is easy  – You can even create isomorphic/universal web apps with server side rendering  – Faster updates | Drawbacks  – Steep learning curve  – JSX is not appreciated by all web developers  – It consists of a very sophisticated view layer  – It’s a library only not a framework  – Flux architectures  – Some configurations would be required if you integrate React into an MVC framework |

**React Js**

NodeJS can be considered as a server itself rather than a framework which is powered by Google chrome V8 JavaScript engine. It comes with a callback concept to achieve the all new idea of event driven single threaded server programming and also executes JavaScript in the server side. Moreover, Node.js is mainly used to simplify the development of complex applications.

Top 3 reasons to choose NodeJS:

|  |  |  |
| --- | --- | --- |
| 1) Server side proxy  NodeJS can be easily used as a server-side proxy. That’s how it can handle a number of simultaneous connections in a non-blocking manner. It is mainly used when you want to collect data from multiple source points or want to proxying different services with different response times.  Moreover, you can build a client side app for assets and proxying/stubbing API requests with NodeJS development server. | 2) NPM  NPM(Node Package Manager) play an important role which should not be neglected at all. It comes by default with your Node.js installation and provides support for package management. The idea of NPM is similar to the concept of Ruby Gems. Some of the most popularly used NPM modules are given here:  – mongojs and mongodb  – connect  – bluebird  – lodash  – pug  – socket.io and sockjs  – moment | 3) Data Streaming  HTTP requests and responses are considered as isolated events but they are actual data streams . These are used in NodeJS to build some amazing features. This helps in proxying between different data resources and also used for real time video and audio encoding. |

Pros and Cons

|  |  |
| --- | --- |
| Advantages  – It shares the same piece of code with both client and server side  – Concurrent request handling due to asynchronous event driven IO  – You can easily stream big files  – NPM(Node-packaged modules) has already become wide and growing at fast rate  – Easy to learn  – Large and vibrant support community | Drawbacks  – It is not scalable because one CPU is not enough to take advantage of multiple tasks  – One should have a deep understanding of JavaScript if he wants to work with NodeJS  – It is not meant for CPU-intensive tasks and only suited for web servers  – Nested callbacks  – Relational database issues |