Rich Web Application Revision Week 1

# What is the Web?

## **What is the Web**

The informal definition behind this question is the web is the biggest application on the public internet and the private intranet. In one view as the consumer, the web is a portal to a vast amount of knowledge, opinion, services and a way to connect with others. In another as the developer, the web is a suite of technologies for building the interfaces that the users or machines can interact to get access to all the information.

Think about this question. What technologies make up the web and how do we use them.

## **What is a Web Application?**

Loosely, this is described as the client-side machine system provisioned over the network using the standard modern tools. This is used by the end-user to interact with a server-side interface on a remote device, which a browser or a graphical engine is the standard method. There are many different architecture designs used to build the business and presentation logic responsibilities between the client and server machines. This has evolved over time as devices has improved in the systems power combined with the users demand in improved UX.

## **What is a Rich Web Application?**

This is where the Client has the responsibilities to implement the presentation layer logic, which is the user interface and the user experience. This is the latest evolution in a series of iterative designs changes the web applications have undergone since the first websites. Rich Web Systems employ heavy use of JavaScript to offer a lean data transfer interface with the server, such as Bootstrap.

## **When is it a Web Application?**

The main difference between a website and web applications is the interaction with a server machine that makes the application more dynamic. Websites are designed to display information only. Usually the communication involves a client – server using a protocol like HTTP, combined with a certain type of message standard serialised formats like JSON. Once these are met, the last criteria would be an agreed design pattern using suitable APIs with the JavaScript as part of the application

Some of the technologies used in the modern architecture design are HTML5, CSS3, ECMAScript, Web Sockets, XML, JSON and design patterns like REST

# Websites vs Web Applications

## **What is the difference?**

### Websites

The purpose of the website is to convey information, such as a picture document repository or a gallery. These are statically created to being served to the browser server. Usually HTML and CSS handle most of the UI features. JavaScript was used but CSS is taking over some of its tasks more easily. There is little interaction with the server engine.

### Web Applications

This is heavily involved with the server, mainly interaction with Data Processing functions. For example, a message being sent through a social media app platform. The content is usually generated more dynamically from the server. HTML and CSS are currently being used but may become obsolete soon. There is heavier use of JavaScript to interact with the server interfaces.

## **Data Synchronisation**

Because of the complexity of building web application over websites, data synchronisation becomes an issue that needs to be handled. This is due to multiple sources from servers combined with ensuring the views contain the same information for all potential users.

## **User Experience**

The goal of the interface is to be as intuitively and functional as possible for the user. To rely on this, designers often use idiomatic usage of graphical elements as input nodes. The easier and more quality the UI is for the user, the higher level of complexity of the Web Application. This can be completed through the Dom, Google Metrics tools and AB testing

## **Security**

The issue of the user, their identity and identity verification in the application. When a change happens to the server, it must have been authorised by the application combined with the users’ authentication in the system. The client must also guard against the process and be ready for success and failure scenarios.

# Bootstrapping a Web Application

## **How does the Web Application get started?**

* User navigates the URL in the browser
* The browser makes the TCP/IP connection to the server IP address and the designated port associated
* The application server or HTTP proxy, which is constant state of listening, accepts the connection from the browser.
* The browser sends a HTTP request over the connection.
* The server parses the request and responds to the browser over the same open connection.

## **Loading the Web App**

The http request will start loading the initial assets (HTML, CSS etc) from the server. The browser parses the files and run any commands contained within, including commands back to the server to fetch more data. This continues until all the assets have been fetched.

## **The Document Object Model**

This is an internal, memory representation of the content in a browser window, which is both written to and read from. Displaying any content is achieved through the DOM, which when changed changes the outlook of the web app. The DOM is modelled logically as a tree structure.

## **Application State and the DOM**

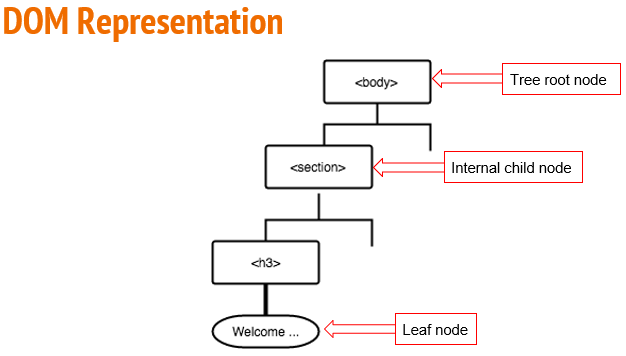
The application state is the values on the data objects on disk or in-memory on the server. We consider the DOM to be part of the App state, which must have the must accurately state from the server and therefore must be synchronised properly, which is in itself is a complex task.

## **Scenarios – HTML Nested Elements**

<body>

<section id = “intro”>

<h3>Welcome to Web Programming</h3>

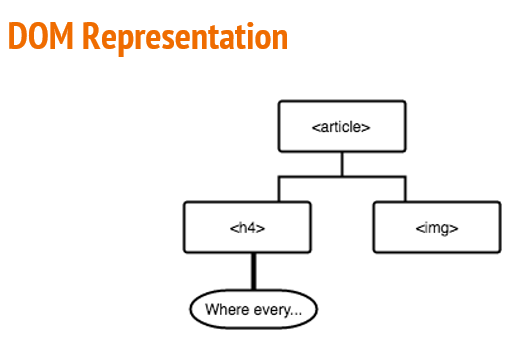
 </section>

</body>

<article class = “serious”

<h4>Where everyday is Caturday</h4>

<img src = “<website link to a image from another server http protocol”>

</article>

Other examples include calls to APIs, external Script Element or inline Script Element

# A brief history of Web Apps

## **How did we get here?**

The application is different from any other type in any coders’ perspective, from its mix of languages and design patterns to the different practises used on it. The continuous evolvement suggest things are still not settled just yet.

## **Origins**

A Researcher / Developer at CERN named Tim Berners-lee proposed the idea of a networked, hyperlinked documents in 1989. In 1991 the prototypes of HTML, HTTP and URL specifications were developed. The most innovative development at the time was one-way text over the network allowing users to reference documents within documentation. Later in the 90s, further developments and standardisations lead to the World Wide Web consortium (W3C). These early sites can be found at <http://info.cern.ch/>

## **Mosaic**

The first sites with GUI technology mixed in the WWW was developed by a team led by Marc Andreesen in 1993 at the National Center for Supercomputing Applications in the university of Chicago. Funding for the organisation came from the provisions of the US High Performance Computing and Communications Act of 1991, sponsored by the Senator AL Gore. Mosaic was free, which gave rapid growth to the Web. Domain name registration exploded in response to this.

## **Web Servers**

This is a network listener that accepts incoming HTTP requests from the browser and responses with data. Early web servers were simple document loading and transferring contents of HTML files from disk to the requesting browser using GET and POST protocols. HTML contained <form> to facilitate this and was very primitive.

## **Netscape and the Browser Wars**

Andreessen founded the Netscape Navigator, a dominant browser engine, when he left University of Chicago. The dominance lasted until Microsoft released its own Internet Explorer bundled in its OS package. Subsequently the web standards dropped from that. Microsoft aftermath was being sued after being found abusing its monopoly position to the US Dept. Of Justice antitrust suit.

## **CSS, Java, JavaScript**

CSS was initially combined with html to style the content. Very quickly it was found the necessity to separate the pages from each other. CSS also works well with JavaScript in CSS3. Java was used briefly to load applications over the network for application containers (known as Applets) in isolation. It was clunky and eventually dropped. JavaScript was developed as the functional way to get access to the DOM using APIs by an engineer in 1995 called Brendan Eich while working at Netscape. While it did not get immediate gratification due to the flaws and speed, Google released a JS runtime that sped the process for JavaScript. JavaScript is now regarded as the implementation of ECMAScript and became standardised.

# HTML5

## **What is HTML5**

Html5 is the latest technology suite that determines how the page is layered and rendered. It was created in 2004 and supersedes HTML4 from 1997. It removes the need for plugins like Silverlight. It also created sematic elements as part of the application, creating more readable code.

* Article
* Section
* Figure
* Audio
* Summary
* Header
* Address
* Figcaption
* Video
* Details
* Footer
* Aside
* Nav
* Canvas

Inline Elements were also created:

* Mark
* Meter – gives visual representation of fractions / decimal value with alternative text
* Progress
* Output
* Time

Input types have expanded in their functionality:

* Color
* Email
* Number
* Date, datetime, week, month
* Range
* Search
* Tel
* url

Input Qualifiers have been added

* height, width
* min, max
* pattern (regexp)
* placeholder
* step