**W1D1-:**

I learn about :

Layer of the web:

The difference of internet and web browser.

More Advance URLs.

How Browser display a page.

Structure of a page and web programming

* Structure of an HTML5 page.
* Block and Inline level elements.
* HTML Validation checking code it follows HTML syntax.

**W1D2-:**

I learn about:

Cascading Style Sheet

The Basic of CSS and relativity to see with content vs. Presentation.

And the use of CSS property.

Override Rule and class vs id

W3C CSS Validator: Check your CSS to make sure it meets the official CSS specifications

**W1D3-:**

I learn about:

CSS for page layaout

Box model is a fundamental knowledge in this step to work in CSS. Eg. Content border padding and margin.

Detail about Block boxes: •By default block elements take the entire width space of the page unless we specify.

Detail about Inline boxes • size properties (width, height, min-width, etc.) are ignored for inline • margin-top and margin-bottom are ignored

There is four type of Positioning i.e static, relative, absolute and fixed

Differentiating Alignment vs. float vs. position.

**W1D4-:**

I learn about:

HTML5 Forms:

Responsiveness guidelines: •Mobile First means designing for mobile before designing for desktop or any other device.

Responsive is the strategy of making a site that responds to the browser and device width.

Forms are created with the <form> tag, and can be submitted with either an HTTP GET or POST method.

The use of Form controls, Text fields, check box, Radio buttons, text area and labels, and buttons.

**W1D5-:**

I learn about:

JavaScript for Modern Web Apps:

The use of client side programming.

How code running between web browser and web server.

JavaScript variables and type in ECMAS6.

The use and difference of Function Declaration and Function expression.

JavaScript allows you to declare anonymous functions.

The use of Window.onload in javaScript.

Event-driven programming • JS programs have no main; they respond to user actions called events

Document Object Model: html -> head -> title and

: html -> body-> h1-> div -> p

**W1D6-:**

I learn about:

JavaScript Programming Environment:

The hierarch of the six global DOM/BOM objects and their properties.

Unobtrusive styling allows separation of web site into 3 major categories:

• content (HTML) - what is it?

• presentation (CSS) - how does it look?

• behavior (JavaScript) - how does it respond to user interaction?

Timer event (setTimerOut and serTimerInterval)

Testing code using mocha and chai.

**W1D7-:**

I learn about:

JavaScript Functional Programming:

How to use functional programming in javaScript most of the we learned in MPP I see the only difference is notation. And new futures such as ‘for of’ vs ‘for in’.

And Another good thing to know Rest and Spread.

• No function overloading in JavaScript

● You can create a regular expression in two ways – Either using the RegExp constructor – Or with the a regular expression literal

**W1D8-:**

I learn about:

Scope closures, and encapsulation:

I learned the basic concept of JavaScript has global scope and local scope within functions when variables are declared with var, and now has block scope with const and let.

variables defined with var are hoisted and have value undefined until it is assigned a value in code ¬ Do not use var assignments in new code.

In first phase, JS engine looks through all global code for function declarations and global variables (hoisting).

In second phase, JS engine • executes code line-by-line • for every function call adds new context and to execution context stack.

closure: A first-class function that binds to free variables that are defined in its execution environment

Function factory with closures.

**W1D9-:**

I learn about:

Module and Objects:

Unlike java in java script you can create objects without creating a class.

objects in JavasScript are like associative arrays. Eg const x = { ‘a’ :79 }

this = window //out side function

this = object inside method

implementation of revealing module

the use of IIFE module and syntax.

How to access the private variable and function.

**W1D10-:**

We learn about:

Inheritance :

The use of inheritance comparing with module.

inheritance is one important language feature that avoids duplicating code

the meaning of [[Prototype]].

methods are often shared, but the object state generally is not.

Creating objects using constructor.

Constructor vs object literal

The deference beteween java syntax and javaScript

**SCI Part**

1. **HTML:**

● The basic technologies that make up the Internet are the World Wide Web and the Hyper Text Markup Language (HTML). Many technologies are built on top of other technologies.

● Life is found in layers and the TM Technique gives us access to the full range of our

awareness and thoughts.

1. **CSS:**

The basics of CSS give different visual styles to HTML elements, changing their preset default appearance.

● Separating style rules from HTML enables the same underlying content to have multiple appearances. The same underlying pure consciousness (or unified field) takes form as the infinite diversity of the manifest universe.

**3) JavaScript:**

JavaScript is a powerful lightweight loosely typed interpreted functional OO language that makes web pages highly dynamic and responsive. It runs in all browsers and is an increasingly important programming language.

● If our awareness is grounded in pure consciousness then our actions will be dynamic and responsive.

**W3D1-:**

We learn about:

Classes:

JavaScript classes are mainly a helpful syntax over JavaScript constructor functions. often need to create many objects of the same kind. When you use ‘new’ ¬ A new object is created. ¬ The constructor runs with the given argument and assigns it to this ¬ …Then we can call object methods.

JavaScript classes, introduced in ECMAScript 2015, are primarily syntactical sugar over JavaScript's existing prototype-based inheritance. Classes are in fact "special [functions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions)", and just as you can define [function expressions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/function) and [function declarations](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/function), the class syntax has two components: [class expressions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/class) and [class declarations](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/class).

The body of a class is executed in [strict mode](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Strict_mode). Getter/Setter is different from ordinary oriented taker of - latter will throw error if misspell - former will create a new property with the misspelling .

JavaScript classes add keywords extend and super that are abstractions over the details of inheritance with function constructors and prototype chains. As a developer you should implementing the encapsulation, Encapsulation is hides internal component implementation details and provides a well delimited external interface for external components.

**W3D2-:**

We learn about:

DOM, JQuery:

Working in the DOM can be clunky to use, the same code doesn't always work the same way in every browser due to this problem JQuery framework adds useful futures to java Script.

Let say • many useful extensions to the DOM • adds utility functions for built-in types String, Array, Object, Function • improves event-driven programming • many cross-browser compatibility fixes • makes Ajax programming easier.

jQuery is powerful because of these design principles • an expressive method for defining a set of elements, a superset of CSS selectors • useful and commonly needed methods for navigating the DOM tree • heavily overloaded APIs • functional programming techniques that apply operations to sets of elements at a time • method chaining for succinct operations

**W3D3-:**

We learn about:

Event Handling:

Event handling is a call back function which executed later when the event occurs. Such as mouse, keyboard, windows, forms events. And Event objects may have properties and methods such as the following: function handler(evt) { // an event handler function ... }.

JavaScript code runs inside of an object and the ‘this’ keyword refers to that object. Meanwhile ‘this’ is used that enables handlers to be reused across different kinds of elements.

Events bubble from the bottom of the DOM tree to the top. The jQuery stopPropagation method prevents bubbling up the element tree. jQuery’s stopImmediatePropagation method prevents any other handlers that might be attached to the current element from being executed.

JavaScript event handling use in the stack event loop to execute events.

**W3D4-:**

We learn about:

Introduction to Servlets and Web Containers:

Interaction between a browser (client) and a web server, the Client requests a resource (file, picture, etc) and Server returns the resource, or declares it’s unavailable.

The knowledge of server is a very amazing and Servers know how to translate an HTTP request into an action of locating a resource. They also know how to produce an HTTP response that contains HTML and gives information about the requested resource.

A request-response programming model by setting up Java JDK • IntelliJ IDE • Tomcat (Container). Meanwhile Every platform for web applications has a mechanism to dynamically generate web pages containing information from the server.

**W3D5-:**

We learn about:

Managing state in servlet:

HTTP request messages send input parameters as name/value pairs. Input parameters are text that must be accessed and converted by a servlet. This is the main mechanism web apps use to send information from the browser to the server.

The most important thing in this topic is understanding Redirecting and forwarding requests. Every request is new request to the server. From the server's perspective, the request begins when the client opens a socket to the server, and it ends when the server sends the last packet back to the client and closes the connection.

And the other important understanding unlike IP and TCP/IP, Application often cannot function correctly being totally stateless.

Web containers use cookies to create sessions. A web conversation, holds information across multiple requests. Before container responds, saves the session info to some data store.

**W3D6-:**

We learn about:

Java Server Pages:

The web container generates a servlet from a JSP file the first time the JSP is requested from a web application. Using java server page has a benefit of separate display from processing, i.e., separate html from java.

Four types of JSP elements: Script: embedded Java statements … • Directive: page level operations for the translation phase. • EL Expression: more convenient and powerful than a JSP expression ${ } vs • Action: JSP “functions” that encapsulate some commonly used functionality

In a well-structured, cleanly coded application, the presentation layer is separated from the business logic, which is likewise separated from the data persistence layer. • It's actually possible to create JSPs that display dynamic content without single line of Java inside the JSP. That's our mission!!!

When you use JSP pages according to a Model 2 architecture, there is a servlet that acts as a controller (process of knowing). JSP is a technology that was designed for the presentation layer, also known as the view. In most organizations, user interface developers are responsible for creating presentation layer. These developers rarely have experience writing Java code, and providing them with the ability can be dangerous.

**W3D7-:**

We learn about:

Custom Tags:

Developers can create custom tags for JSP that use a Java tag handler to provide any desired functionality in a simple tag definition for JSP authors.

main purpose of most JSP custom tags is to provide an easy mechanism to dynamically “generate markup” for common processing tasks • Current best practice with JSP is to use tags on pages and avoid any JSP scripting elements • JSTL and many other libraries of custom tags have been created.

Reuse through software components is an important general principle of software engineering. As web applications gain sophistication and complexity, component reuse becomes increasingly important.

**W3D7-:**

We learn about:

Ajax and JSON:

Ajax requests require a url for the target on the server, a designation of whether to send a Get or Post request, one or more callback functions to be called with the result, and optionally a set of request parameters. jQuery provides convenient wrapper methods: $.ajax(), $.get(), and $.post.

Ajax must be run on a web page stored on a web server • cannot be run from a web page stored on your hard drive • can open an html page from hard drive and run normal JavaScript • will not work if there is an Ajax call on that same page because browsers only allow Ajax calls to go to a ‘domain’ from which the page is served.

Ajax can only fetch files from the same server that the page is on.