Javascript – Course Outline

# Duration

4 days

# Objectives

Web client-side Programming in Javascript (along with Object oriented and functional programming constructs)

# Audience

This course is for Web developers

# Pre-requisite

* Prior experience developing web applications using HTML, CSS and Javascript

# Outline

## Day 1

**JavaScript Functions - Advanced**

* Invocation, Overloading, The Arguments Object, Recursion
* Different ways of declaring the functions
* Nested / inner functions
* Named functions
* Anonymous functions
* Function as singleton object
* Assigning functions to variables (named and anonymous)

**Functions as Object - Advanced**

* Changing the context of a function
* Different ways of changing the context – call, apply
* Implement looping with a callback

**Instantiating Objects - Advanced**

* Using arguments.callee
* Variable arguments – using arguments.length
* Creating (Array().slice) array objects and working with arrays methods
* Modifying Array.prototype, add methods to array (Array.method)
* Exception handling

**Closures**

* Basic Closure
* Callback functions, timers, event listeners using closures
* Private properties
* Self-executing, temporary function
* Closures and looping
* Anonymous wrapper functions for wrapping libraries

**Function Prototypes and OOPs**

* Augmenting Types (Function.prototype)
* Adding prototyped method to a function
* Overriding prototyped properties
* Chainable methods
* Using typeof, instanceof, constructor
* Writing default constructor
* Understand prototypal inheritance
* Modifying built-in prototypes
* Potential pitfalls that occur in extension
* Associate context to the original object
* Enforce context to all functions
* Use .bind method from prototype.js
* Determine function’s argument length
* Method overloading
* Javascript Object Notation: JSON
* Marshalling and Unmarshalling JSON Data

**Modular javascript**

* Use functions to create modules
* Remove the use of global variables
* Method cascade (returning same object by many methods)
* Method currying to manipulate function values using Function.prototype
* Automatically adding “on”, “fire” and event registry

**Accessing and manipulating HTML DOM and Browser Object Model (BOM)**

* BOM: Windows methods and events
* Window objects – navigator, frames, location, history, document, screen
* Document API
* Accessing nodes hierarchically, by type, name and value
* Attaching and Detaching events
* Event propagation: capturing and bubbling
* Accessing attribute nodes
* Creating, deleting nodes in the DOM
* Identifying the target of an event
* Javascript – cross browser support in a matrix

## Day 2

**Browser Caching**

* Understand caching in web applications – Browser caches and Proxy caches
* Controlling caches – meta tags, pragma http headers, Expires, Cache-control http headers, validators and validation
* Building a cache aware site
* Javascript versioning and caching

**Testing Javascript**

* QUnit
* Testing DOM interaction
* Behavior Driven Development (BDD)
* Using Jasmine to add BDD Domain Specific Language (DSL)

**Asynchronous Communication with Server**

* Making GET and POST calls
* Processing Text and HTML Response
* Processing XML Response
* Handling JSON in AJAX
* Handling JSON request/response in REST environment

**Cross-domain communication**

* Cross-domain communication options and issues
* Implementing JSONP on the client

**JavaScript in HTML 5 Environment**

* Storage API: LocalStorage and SessionStorage
* PostMessage API
* Using Web Workers

**Javascript in Production**

* Coding practices (specifically for production code)
* Javascript minimizers
* Minimize http requests
* Use Content Delivery Network
* Add Expires / Cache-control header
* Gzip components
* Remove duplicate scripts

**Enterprise Development**

* Code Reuse Patterns
* Browser Compatibility Issues
* DOM and Browser Patterns
* Logging and Debugging
* Testing JavaScript Applications
* Performance Issues

**History, Back-button, Deep-linking (and SEO)**

* What these features have in common: URLs
* URLs and JavaScript applications
* APIs and browser facilities for history and deep-links
* State-based application design for history/deep-link support

## Day 3

**Patterns for clean extensible design in larger JavaScript applications with Hands-on**

* Namespaces
* Service Locators
* Dependency Injection
* Inversion of Control
* Messaging and event buses
* Different types of events
* Design tradeoffs

**Overview of jQuery**

* Initialize jQuery
* Selections using CSS3
* Filters, multiple item selectors
* Changing styles and content
* Effects with jQuery and Animations
* Callback functions, event handlers
* AJAX with jQuery

**Hands-on using Require JS**

* Configure require JS
* Load nested dependencies
* Tool to use deployment

## Day 4

**Overview of NodeJS**

* Setting up and configuration of NodeJS
* Web application with request handler using NodeJS
* Using Node Package Manager (npm)
* Leveraging Express JS framework

**Overview of Grunt JS**

* Installing Grunt
* Configuring Tasks
* Creating Grunt tasks
* Writing custom Grunt task

# Hardware & Network Requirements

* All participants to have individual desktops with at least dual core or higher CPU with 2GB RAM
* All participants’ system to be connected to internet

# Software Requirements

* Windows XP/7 / Linux Operating System
* Text editor – SublimeText / Notepad++
* Web browser – Mozilla Firefox with firebug, Google Chrome
* Web server either Apache HTTP server or Apache Tomcat server