|  |
| --- |
| Advanced JavaScript |
| **Overview** |
| This workshop teaches the participants how to take advantage of some of the advanced features offered by JavaScript for building better maintainable applications large scale applications. It also introduces some of the frameworks that can assist in this process. |
| **Past occurances of this program** |
| * Amdocs, Pune * Honeywell, Bangalore (3 occurrences) * Pega Systems, Bangalore * 12 other instances |
| **OBJECTIVES** |
| At the end of this training course, the participants will:   * Understand and appreciate the dynamic typed nature of JavaScript language * Be able to understand the exploit some of the functional programming techniques in JavaScript * Understand and resolve scope issues in JavaScript * Techniques for modularizing JavaScript code * Know how to modularize and loadJavaScript files using Require.js * Be aware of some of the quirks in the JS language * Using JavaScript Design Patterns |
| **suggested audience** |
| Web developer who wants to build best-of-breed web UIs with the simplicity and elegance of JavaScript. |
| **DURATION** |
| 2 Days |
| **PARTICIPANT PREREQUISITES** |
| **Must Have**   * Web development experience with HTML, CSS, JavaScript   **Good To Have**   * Exposure to any server technology (J2EE, .NET, Ruby etc) |
| **infrastructure requirements** |
| * Any Web Server (IIS or Tomcat) * Chrome * Internet Connection (Mandatory) * Node.js * Any Text Editor |
| **DAY WISE SYLLABUS** |
| * Expressive JavaScript   + The Flexibility of JavaScript   + JavaScript as a Loosely Typed Language   + Functions as First-Class Objects   + Object Mutability * Functions In Depth   + Function Objects   + Function Literal   + Function Invocation Patterns   + Augmenting Types   + Recursion & Closures   + Functions as Callbacks   + Function Currying & Memorization * Closures In Depth   + How Closures Work   + Private Variables, Callbacks and Timers   + Binding Function Contexts   + Overriding Function Behavior * Object Orientation With Prototypes   + Object Instantiation   + Constructors   + Inheritance and Prototype Chain   + Extending Object   + Extending Number   + Instantiation Issues   + Writing class-like code * JavaScript Design Patterns   + Common Design Pattern Implementations   + Module Pattern   + Revealing Module Pattern   + Observer Pattern   + Factory Pattern   + Strategy Pattern   + PubSub Implementation  ES6  * New in ES6   + Classes   + let and const   + Arrow functions   + Enhanced Object Literals   + Destructuring   + Spread Operator   + Generators and Iterators   + Symbols   + Comprehensions   + Async Programming using Promises * Using Babel.js |