Day 1: Node Core
Kraken Forge at PayPal

Objectives

On completion of this course, students will be able to:

- Clearly understand the platform design choices that led to Node.js choosing an event loop and what this means for applications built on that foundation.
- Understand the unique trade-offs present in event-driven programming.
- Create Node.js modules and express code modularity in an application.
- Understand the core flow control patterns in Node.js and know when it is appropriate to use callbacks, event emitters or streams.
- Create and manipulate buffers.
- Understand how to manage error state and know when a process should exit due to an error.
- Build network applications with Node.js.
- Interact with the local file system.
- Understand and use the path module to effectively handle and transform file paths.
- Perform common file system operations.
- Create a low-level TCP server and manage connections to the server.
- Create HTTP servers in Node.js.
- Use Node.js to create APIs and communicate server-to-server.

Training Units

- Intro to Node.js
- Modules and npm
- The Callback Pattern
- Event Emitter
- Error Handling
- Buffers
- Streams
- File System
- TCP
- HTTP
- Making HTTP Requests

Detailed Syllabus

- Intro to Node.js
    - RAM vs. I/O latency
    - Blocking vs. Non-Blocking
    - Event-driven Programming
    - JavaScript Closures
    - Event Loop
    - Blocking The Event Loop
    - Node.js Philosophy (userland vs. core)
    - Dealing with docs

- Modules and npm
    - Anatomy of a module
    - Private code
    - Accessing and using modules
    - npm commands
    - package.json
    - versioning patterns in the node community
- The Callback Pattern
    - NOTE: closures covered extensively in Crockford's course
    - What are callbacks
    - Examples
    - Callback-last
    - Error-first
- Event Emitter
    - When to use Event Emitters
    - Binding Functions to Events
    - Examples
    - Creating an Event Emitter
- Error Handling
    - Callbacks: Error-first
    - Errors in Event Emitters
    - Uncaught Exceptions
    - Using Domains
- Buffers
    - Why Buffers exist
    - Creating Buffers
    - Reading Buffers
    - Writing Buffers
    - Manipulating Buffers

- Streams
    - What are streams
    - Read Stream API
    - Write Stream API
    - Flow Control
    - Piping
    - Duplex Stream
    - Transform Stream
- File System
    - Disk I/O
    - File descriptors
    - The path module
    - The fs module
- TCP
    - Networking with TCP
    - TCP servers
    - Creating connections
- HTTP
    - Creating an HTTP Server
    - HTTP server requests
    - Piping requests
    - HTTP server responses
    - Managing headers
- Making HTTP Requests
    - HTTP Client Requests