# Mastering V8 Javascript Engine

### Part I: Introduction to JavaScript Engines

1. **Understanding JavaScript Engines**
   * What is a JavaScript Engine?
   * Role of JavaScript Engines in Modern Applications
   * Overview of Popular JavaScript Engines: V8, SpiderMonkey, JavaScriptCore, Chakra

* **Getting Acquainted with V8**
  + History and Evolution of V8
  + Key Features and Design Goals
  + The Role of V8 in Chrome and Node.js
  + Recent Updates and Version Overview (up to V8 v11)

### Part II: Setting Up the Development Environment

1. **Installing V8**
   * System Requirements
   * Cloning the V8 Repository
   * Building V8 from Source
   * Understanding Build Configurations: Debug vs. Release
   * Setting Up Development Tools and IDEs

* **Hello World with V8**
  + Writing Your First Embedded V8 Application
  + Compiling and Running the Application
  + Understanding Basic V8 Concepts: Isolates and Contexts

### Part III: Core Concepts and APIs

1. **V8 Architecture Overview**
   * Execution Pipeline: Parsing, Bytecode Generation, Optimization
   * Ignition Interpreter and TurboFan Compiler
   * Hidden Classes and Inline Caches
   * Memory Management and the Heap

* **Working with V8 Isolates and Contexts**
  + What are Isolates?
  + Creating and Managing Isolates
  + Understanding Contexts
  + Sharing Data Between Contexts
* **V8 Data Types and Handles**
  + JavaScript Values in C++: v8::Value, v8::Primitive, v8::Object, etc.
  + Working with Handles: Local, Persistent, and Weak Handles
  + Converting Between JavaScript and C++ Types
  + Managing Memory with Garbage Collection
* **Executing JavaScript Code**
  + Compiling Scripts: v8::Script, v8::ScriptCompiler
  + Running Scripts and Retrieving Results
  + Handling Exceptions and Errors
  + Working with Script Origins and Source Maps

### Part IV: JavaScript and C++ Interoperability

1. **Embedding Functions and Objects**
   * Exposing C++ Functions to JavaScript
   * Creating and Manipulating JavaScript Objects in C++
   * Using Templates and Function Callbacks
   * Handling Asynchronous Operations

* **Accessing JavaScript from C++**
  + Calling JavaScript Functions from C++
  + Passing Arguments and Handling Return Values
  + Working with Promises and Async Functions
  + Implementing Custom JavaScript Classes in C++

### Part V: Advanced V8 Concepts

1. **Understanding V8's Optimization Techniques**
   * Just-In-Time (JIT) Compilation
   * Deoptimization and Bailouts
   * Inline Caching Mechanisms
   * Code Caching and Startup Snapshots

* **Deep Dive into Garbage Collection**
  + Generational Garbage Collection
  + The Scavenger and Mark-Compact Collectors
  + Incremental and Concurrent Garbage Collection
  + Tuning Garbage Collection for Performance
* **Profiling and Performance Tuning**
  + Using V8's Built-in Profiling Tools
  + Analyzing Heap Snapshots
  + Identifying and Fixing Memory Leaks
  + Optimizing Code for Better Performance
* **Debugging Techniques**
  + Setting Up V8 Inspector
  + Integrating with Debugging Tools (LLDB, GDB)
  + Using Breakpoints and Watch Expressions
  + Debugging Optimized Code

### Part VI: V8 Internals and Contributions

1. **Exploring V8 Internals**
   * Bytecode and the Ignition Interpreter
   * TurboFan Optimizing Compiler Pipeline
   * Understanding the V8 Heap Layout
   * Hidden Classes and Property Access

* **Contributing to V8**
  + Navigating the V8 Codebase
  + Understanding the Development Workflow
  + Writing and Running Tests
  + Submitting Patches and Code Reviews
  + V8's Release Cycle and Versioning

### Part VII: Specialized Topics

1. **WebAssembly and V8**
   * Introduction to WebAssembly
   * V8's Support for WebAssembly
   * Compiling and Executing WebAssembly Modules
   * Interoperability Between JavaScript and WebAssembly

* **Security Best Practices**
  + Common Security Vulnerabilities in V8
  + Sandboxing and Isolate Isolation
  + Mitigating Spectre and Meltdown Attacks
  + Secure Coding Guidelines
* **Embedding V8 in Applications**
  + Case Studies of V8 Embedding
  + Building High-Performance Applications with V8
  + Multi-threading and Concurrent Execution
  + Customizing V8 for Specific Use Cases
* **V8 and Node.js**
  + The Role of V8 in Node.js
  + Node.js Native Addons Using V8 APIs
  + Async Hooks and Performance Hooks
  + Optimizing Node.js Applications with V8 Insights

### Part VIII: Latest Features and Future Directions

1. **New ECMAScript Features in V8**
   * Overview of ECMAScript 2023 Features
   * Private Fields and Methods
   * Top-Level await
   * Temporal API Integration

* **Recent Performance Enhancements**
  + Optimizations in String and Array Handling
  + Improvements in Garbage Collection Algorithms
  + Enhanced WebAssembly Performance
  + Startup and Memory Usage Optimizations
* **Future Roadmap and Upcoming Features**
  + Planned ECMAScript Feature Support
  + Proposed Changes in V8 APIs
  + Community Proposals and RFCs
  + How to Stay Updated with V8 Developments

### Appendices

A. **Glossary of Terms**

* Definitions of Common V8 and JavaScript Engine Terminology

B. **V8 API Reference**

* Quick Reference Guide to Essential V8 Classes and Functions

C. **Build Configuration Options**

* Detailed Explanation of Build Flags and Options

D. **Resources and Further Reading**

* Official Documentation and Guides
* Community Forums and Support Channels
* Recommended Books and Tutorials

#software/tools/webdev/v8