6-Month JavaScript Framework Prerequisites Course

Master JavaScript Skills for Modern Frameworks (React, Vue, Angular)

Course Overview

This comprehensive 6-month JavaScript course builds the essential skills needed to succeed with modern JavaScript frameworks. You'll master every JavaScript concept required for React, Vue, Angular, and other modern frameworks, ensuring a smooth transition when you choose your framework path.

Course Duration: 6 months (24 weeks)
Schedule: 2 sessions per week, 2 hours each (4 hours/week)
Prerequisites: Basic HTML and CSS knowledge
Final Goal: Complete JavaScript mastery for framework success
Methodology: GitHub-first collaborative learning from day one

---

Month 1: JavaScript Fundamentals + Version Control

Week 1: Development Environment and Git Mastery

Session 1: Git & GitHub Foundations

- Git installation and configuration
- Repository creation, cloning, and management
- Basic workflow: add, commit, push, pull, status
- Branch creation and switching
- GitHub organization membership and collaboration

Session 2: JavaScript Environment Setup

- VS Code setup with essential extensions
- Browser DevTools introduction
- Node.js installation for local development
- First JavaScript program and debugging
- Console methods and basic output

Framework Connection: All modern frameworks require Git for version control and collaborative development. Professional developers use Git daily.

Week 2: Modern Variables and Data Types

Session 1: Variables and Scope

- `let` and `const` (never use `var`)
- Block scope vs function scope
- Hoisting behavior and temporal dead zone
- Global vs local scope understanding
- Variable naming conventions

Session 2: Data Types and Template Literals

- Primitive types: string, number, boolean, null, undefined, symbol
- Template literals with expressions `${}`
- String methods essential for frameworks
- Type checking with `typeof`
- Type coercion awareness

Framework Connection: Modern frameworks rely heavily on `const` and `let`. Template literals are essential for dynamic content and JSX-like syntax.

Week 3: Functions - The Framework Foundation

Session 1: Function Fundamentals

- Function declarations vs expressions
- Arrow functions syntax and behavior
- Arrow functions vs regular functions (`this` binding)
- Function parameters and default values
- Rest parameters `(...args)`

Session 2: Return Values and Pure Functions

- Return statements and implicit returns
- Pure functions concept (no side effects)
- Function composition basics

- Higher-order function introduction
- Functions as first-class citizens

Framework Connection: Arrow functions are everywhere in frameworks. Pure functions are crucial for predictable components and state management.

Week 4: Control Flow for Dynamic UIs

Session 1: Conditional Rendering Patterns

- If/else statements and ternary operators
- Logical operators: `&&`, `||`, `??`
- Short-circuit evaluation
- Switch statements and use cases
- Guard clauses and early returns

Session 2: Iteration Patterns

- `for...of` loops for arrays
- `for...in` loops for objects
- Traditional for loops when needed
- While and do-while loops
- Breaking and continuing loops

Framework Connection: Conditional rendering and list rendering are core framework concepts. Ternary operators and logical operators are used constantly in templates.

Month 2: Arrays and Objects - Data Management Masters

Week 5: Array Methods - The Framework Workhorses

Session 1: Essential Array Methods

- `map()` - Transform every element (component lists)
- `filter()` - Conditional element display
- `find()` and `findIndex()` - Single element search
- `includes()` - Check for element existence
- Method chaining patterns

Session 2: Advanced Array Methods

- `reduce()` - Complex data transformations
- `some()` and `every()` - Boolean testing
- `sort()` - Ordering with custom comparators
- `forEach()` vs other methods
- Performance considerations

Framework Connection: `map()` is used for rendering lists, `filter()` for conditional display, `reduce()` for state calculations. These are daily-use methods in frameworks.

Week 6: Object Mastery

Session 1: Object Fundamentals

- Object creation and property access
- Bracket vs dot notation
- Dynamic property names
- Property existence checking
- Object method definitions

Session 2: Object Manipulation

- Object.keys(), Object.values(), Object.entries()
- Object.assign() and object merging
- Nested objects and property access
- Object comparison challenges
- Reference vs value concepts

Framework Connection: Objects represent component props, state, and configuration. Understanding object manipulation is crucial for framework data management.

Week 7: Destructuring - The Modern Way

Session 1: Array Destructuring

- Basic array destructuring syntax
- Skipping elements and default values
- Rest patterns in destructuring

- Nested array destructuring
- Swapping variables

Session 2: Object Destructuring

- Basic object destructuring
- Renaming variables during destructuring
- Default values in object destructuring
- Nested object destructuring
- Destructuring function parameters

Framework Connection: Destructuring is used everywhere in frameworks - extracting props, state values, API responses, and event data.

Week 8: Spread and Rest - Modern JavaScript Patterns

Session 1: Spread Operator

- Array spreading and concatenation
- Object spreading and merging
- Spreading in function calls
- Shallow copying with spread
- Immutable update patterns

Session 2: Rest Patterns

- Rest in function parameters
- Rest in destructuring
- Collecting remaining elements
- Flexible function signatures
- Rest vs spread distinctions

Framework Connection: Spread operator is essential for immutable state updates. Rest patterns help with flexible component APIs.

Month 3: Asynchronous JavaScript - The API Connection

Week 9: Understanding Asynchronous JavaScript

Session 1: Synchronous vs Asynchronous

- JavaScript event loop understanding
- Call stack, callback queue, event loop
- Blocking vs non-blocking operations
- setTimeout and setInterval
- Understanding asynchronous behavior

Session 2: Callbacks and Callback Patterns

- Callback functions basics
- Error-first callback conventions
- Callback hell and pyramid of doom
- Event-driven programming with callbacks
- Callback composition patterns

Framework Connection: Frameworks handle many asynchronous operations - API calls, user events, lifecycle methods. Understanding async is crucial.

Week 10: Promises - Modern Async Handling

Session 1: Promise Fundamentals

- Promise creation and states
- `.then()` and `.catch()` methods
- Promise chaining
- Error propagation in chains
- `.finally()` method

Session 2: Advanced Promise Patterns

- Promise.all() for parallel operations
- Promise.race() for timeouts
- Promise.allSettled() for handling failures
- Creating custom promises

- Converting callbacks to promises

Framework Connection: API calls in frameworks return promises. Understanding promise chaining is essential for data fetching and error handling.

Week 11: Async/Await - The Modern Standard

Session 1: Async/Await Basics

- `async` function declaration
- `await` keyword usage
- Converting promises to async/await
- Sequential vs parallel execution
- Async function return values

Session 2: Error Handling and Best Practices

- Try/catch with async/await
- Error handling strategies
- Combining async/await with Promise methods
- Async/await in different contexts
- Performance considerations

Framework Connection: Modern frameworks use async/await for API calls, data fetching, and asynchronous operations in components.

Week 12: HTTP and API Communication

Session 1: Fetch API Mastery

- Fetch API basics and syntax
- Request configuration and headers
- GET, POST, PUT, DELETE requests
- Request body and data sending
- Response object handling

Session 2: Data Handling and Error Management

- JSON parsing and serialization
- Error handling for network requests

- Status codes and response validation
- Loading states and user feedback
- API response transformation

Framework Connection: Frameworks constantly communicate with APIs. Fetch API is the standard way to get data for framework applications.

Month 4: Advanced Functions and Patterns

Week 13: Advanced Function Concepts

Session 1: Closures and Lexical Scope

- Lexical scoping rules
- Closure creation and behavior
- Practical closure applications
- Memory implications of closures
- Module pattern with closures

Session 2: Function Context and Binding

- Understanding `this` in different contexts
- `this` in arrow functions vs regular functions
- Call, apply, and bind methods
- Method borrowing and context switching
- Avoiding `this` pitfalls

Framework Connection: Closures are used in hooks, event handlers, and component logic. Understanding `this` prevents common framework bugs.

Week 14: Higher-Order Functions and Composition

Session 1: Higher-Order Functions

- Functions that return functions
- Functions that accept functions as parameters
- Callback patterns and customization
- Function factories and generators

- Partial application concepts

Session 2: Function Composition

- Composing functions for complex operations
- Pipe and compose patterns
- Currying and its applications
- Function composition in practice
- Reusable function libraries

Framework Connection: Frameworks use higher-order components, render props, and function composition. These patterns are fundamental to advanced framework usage.

Week 15: Error Handling and Debugging

Session 1: Comprehensive Error Handling

- Try/catch/finally blocks
- Error object properties and types
- Throwing custom errors
- Error boundaries concepts
- Graceful degradation strategies

Session 2: Debugging Techniques

- Browser DevTools mastery
- Console methods beyond log()
- Breakpoints and step debugging
- Network tab for API debugging
- Performance profiling basics

Framework Connection: Frameworks need robust error handling. Component error boundaries and debugging techniques are essential skills.

Week 16: Modern JavaScript Features

Session 1: ES6+ Essential Features

- Default parameters and rest/spread
- Enhanced object literals

- Computed property names
- Method definitions and getters/setters
- Symbol primitive type

Session 2: Advanced ES Features

- Optional chaining (`?.`)
- Nullish coalescing (`??`)
- Logical assignment operators
- Array.from() and Array.of()
- Object.freeze() and immutability

Framework Connection: Modern frameworks expect knowledge of latest JavaScript features. These features make framework code cleaner and more robust.

Month 5: Modules and Architecture Patterns

Week 17: Module System Mastery

Session 1: ES6 Modules

- Import and export syntax
- Named vs default exports
- Re-exporting modules
- Dynamic imports
- Module loading behavior

Session 2: Module Organization

- File naming and organization
- Module dependencies management
- Circular dependency avoidance
- Module design patterns
- Creating reusable modules

Framework Connection: Frameworks are built on module systems. Component imports/exports and project organization rely on module understanding.

Week 18: Object-Oriented JavaScript

Session 1: Prototypes and Classes

- Prototype chain understanding
- ES6 class syntax
- Constructor methods and properties
- Static methods and properties
- Class inheritance with extends

Session 2: Advanced OOP Concepts

- Super keyword and method overriding
- Private fields and methods
- Getters and setters
- Class composition patterns
- When to use classes vs functions

Framework Connection: While frameworks favor functional patterns, understanding classes helps with component lifecycle, inheritance patterns, and library integration.

Week 19: Design Patterns for Frameworks

Session 1: Essential Design Patterns

- Observer pattern (event systems)
- Factory pattern (component creation)
- Module pattern (encapsulation)
- Singleton pattern (global state)
- Decorator pattern (higher-order components)

Session 2: Functional Patterns

- Pure functions and side effects
- Immutability principles
- Function composition patterns
- Memoization for performance
- Functional programming concepts

Framework Connection: Frameworks heavily use these patterns. Understanding them helps you write better framework code and understand framework internals.

Week 20: State Management Concepts

Session 1: State Management Principles

- State vs props concepts
- Immutable state updates
- State normalization techniques
- Local vs global state
- State lifting patterns

Session 2: Event Systems and Communication

- Custom event creation and dispatching
- Event delegation and bubbling
- Observer pattern implementation
- Pub/sub pattern for communication
- Event-driven architecture

Framework Connection: State management is crucial in frameworks. These concepts prepare you for Redux, Vuex, and built-in state solutions.

Month 6: Advanced Topics and Framework Preparation

Week 21: Performance and Optimization

Session 1: JavaScript Performance

- Performance measurement and profiling
- Memory management and garbage collection
- Avoiding memory leaks
- Efficient algorithm choices
- Performance monitoring tools

Session 2: DOM Performance and Optimization

- Efficient DOM manipulation

- Batch DOM operations
- Debouncing and throttling
- Lazy loading concepts
- Virtual DOM concepts (theory only)

Framework Connection: Framework performance depends on JavaScript optimization. Understanding these concepts helps you write efficient framework code.

Week 22: Testing JavaScript Code

Session 1: Testing Fundamentals

- Unit testing concepts and benefits
- Test-driven development basics
- Jest testing framework setup
- Writing testable functions
- Test organization and structure

Session 2: Advanced Testing Patterns

- Mocking functions and modules
- Testing asynchronous code
- Testing error conditions
- Test coverage and quality
- Integration testing concepts

Framework Connection: Framework code needs testing. Understanding JavaScript testing prepares you for component testing and framework-specific testing tools.

Week 23: Build Tools and Development Workflow

Session 1: NPM and Package Management

- Package.json understanding
- Installing and managing dependencies
- NPM scripts and task automation
- Version management and updates
- Local vs global installations

Session 2: Modern Build Tools

- Vite setup and configuration
- Development vs production builds
- Environment variables
- Hot module replacement
- Asset handling and optimization

Framework Connection: All frameworks use build tools and NPM. Understanding these tools is essential for framework development.

Week 24: Framework Preparation and Next Steps

Session 1: Framework Readiness Assessment

- Review of all framework-essential skills
- Common framework patterns in vanilla JavaScript
- Component-like thinking exercises
- State management pattern practice
- API integration patterns

Session 2: Framework Selection and Learning Path

- Overview of popular frameworks (React, Vue, Angular)
- Framework comparison and selection criteria
- Learning resources and next steps
- Portfolio project planning
- Interview preparation for framework roles

Framework Connection: This session ties everything together and prepares students for their framework learning journey.

Key Projects (Progressive Complexity)

Month 1: Interactive Calculator with Git workflow
Month 2: Data Dashboard with Array Methods and API calls
Month 3: Task Manager with Async Operations
Month 4: Component-like Widget Library

Month 5: Module-based Application Architecture

Month 6: Full-featured SPA (Single Page Application) in Vanilla JavaScript

---

Framework Skills Mastered

React Readiness:

✅ Arrow functions and proper `this` binding
✅ Array methods (map, filter, reduce) for rendering
✅ Destructuring for props and state
✅ Spread operator for immutable updates
✅ Template literals (JSX preparation)
✅ Async/await for data fetching
✅ Higher-order functions (HOCs)
✅ Closures (hooks understanding)
✅ Module system (component imports)
✅ Event handling patterns

Vue.js Readiness:

✅ Object reactivity concepts
✅ Template syntax preparation
✅ Component communication patterns
✅ Event handling and modifiers
✅ Computed property concepts
✅ Watchers and observers
✅ Lifecycle understanding

Angular Readiness:

✅ TypeScript-adjacent JavaScript features
✅ Class-based component understanding
✅ Dependency injection concepts
✅ Observable patterns
✅ Service and module patterns
✅ Decorator pattern understanding

Universal Framework Skills:

✅ Component-based architecture thinking
✅ State management principles

✅ API integration and data flow
✅ Error handling and debugging
✅ Testing and code quality
✅ Build tools and development workflow
✅ Git/GitHub collaboration
✅ Performance optimization

Success Guarantee

After completing this course, you'll be able to:

- Jump into any JavaScript framework with confidence
- Understand framework documentation without struggling with JavaScript concepts
- Write clean, modern JavaScript code
- Handle complex state management scenarios
- Build full-featured applications
- Collaborate effectively using Git/GitHub
- Debug and optimize JavaScript applications
- Test your JavaScript code effectively

This course ensures you're not just ready for one framework, but prepared for the entire modern JavaScript ecosystem!