**📌 JavaScript Features – Granular Breakdown**

**1. Core Language**

**1.1 Data Types**

* Primitive Types
  + string (text)
  + number (integer, float, NaN, Infinity, -Infinity)
  + bigint
  + boolean (true, false)
  + undefined
  + null
  + symbol (unique identifiers)
* Objects
  + Plain objects {}
  + Arrays []
  + Functions function() {}
  + Dates
  + RegExp
  + Maps
  + Sets
  + WeakMaps
  + WeakSets
  + Typed Arrays (Uint8Array, Float32Array, etc.)

**1.2 Variables & Scope**

* var (function scoped, hoisting)
* let (block scoped)
* const (block scoped, immutable reference)
* Hoisting behavior (variables vs functions)
* Temporal Dead Zone (TDZ)

**1.3 Operators**

* Arithmetic: + - \* / % \*\*
* Assignment: = += -= \*= /= %= \*\*=
* Comparison: == != === !== > < >= <=
* Logical: && || ! ??
* Bitwise: & | ^ ~ << >> >>>
* Spread ...
* Rest ...
* Ternary ? :
* Comma ,
* typeof, instanceof, in, delete, void

**1.4 Control Structures**

* Conditionals: if, else if, else
* Switch/case (with fallthrough)
* Loops:
  + for
  + for…in
  + for…of
  + while
  + do…while
* Loop modifiers: break, continue
* Labels for breaking nested loops

**2. Functions**

* Function declaration
* Function expression
* Arrow functions () => {}
* Default parameters
* Rest parameters
* Spread arguments
* arguments object
* Higher-order functions
* Closures
* First-class functions
* Immediately Invoked Function Expression (IIFE)
* Pure vs Impure functions
* Function hoisting

**3. Objects & OOP**

* Object literals { key: value }
* Property shorthand
* Computed property names [expr]: value
* Object destructuring
* Nested destructuring
* Object methods
* Property attributes (writable, enumerable, configurable)
* Getters and setters
* Prototypes & prototype chain
* Object.create()
* Object.assign(), spread operator { ...obj }
* Object.freeze(), Object.seal(), Object.preventExtensions()
* ES6 Classes
  + Class declaration
  + Constructor
  + Methods
  + Getters/Setters in class
  + Static methods
  + Inheritance (extends)
  + super keyword
  + Private fields #field
  + Public class fields
  + Abstract-like patterns (not built-in but simulated)

**4. Advanced Concepts**

* Closures
* Scope chain
* Execution context (creation, execution)
* Call stack
* Event loop (macro/microtasks)
* Hoisting (variables, functions)
* this binding rules
  + Global context
  + Inside function
  + Inside object method
  + Arrow functions
  + Explicit binding (call, apply, bind)
* Currying
* Memoization
* Functional composition
* Recursion
* Tail call optimization (in strict mode)

**5. Asynchronous JavaScript**

* Callbacks
* Promises
  + resolve, reject
  + then, catch, finally
  + Promise.all, Promise.race, Promise.allSettled, Promise.any
* async/await
* Microtasks vs macrotasks
* Event loop phases
* Fetch API
* XMLHttpRequest
* Web Workers
* Service Workers
* Timers
  + setTimeout, setInterval, clearTimeout, clearInterval
  + queueMicrotask

**6. Modules**

* ES6 Modules
  + export (named, default, aggregated)
  + import (named, default, alias, namespace \* as)
* Dynamic imports import()
* CommonJS (require, module.exports)
* UMD

**7. Error Handling**

* try…catch…finally
* Error objects
  + Error
  + TypeError
  + ReferenceError
  + SyntaxError
  + RangeError
  + EvalError
  + URIError
* Custom errors (class extends Error)
* Throwing errors

**8. Built-in Objects & APIs**

* Math
* Date
* RegExp
* JSON (stringify, parse)
* Intl (internationalization)
* URL, URLSearchParams
* Console API (console.log, console.error, console.table, etc.)
* Performance API (performance.now())

**9. Browser-Specific Features**

* DOM manipulation (document.querySelector, getElementById, etc.)
* Events (addEventListener, removeEventListener)
* Event bubbling, capturing
* Custom events
* Storage
  + localStorage
  + sessionStorage
  + cookies
* Geolocation API
* WebSockets
* Canvas API
* Web Audio API
* File API
* Clipboard API
* Notifications API
* Fullscreen API
* Fetch + CORS

**10. Meta-Programming**

* eval()
* with statement (deprecated, but exists)
* Proxy
  + traps: get, set, has, deleteProperty, ownKeys, etc.
* Reflect
* Symbols (Symbol.iterator, Symbol.asyncIterator, Symbol.toStringTag)
* Iterators & Generators
  + Generator functions function\*
  + yield, yield\*
  + Async generators

**11. Memory Management**

* Garbage collection (reference counting, mark-and-sweep)
* WeakRefs (ES2021)
* FinalizationRegistry

**12. ECMAScript (ES) Features by Version**

* ES5: Strict mode, JSON, getters/setters
* ES6 (2015): let, const, arrow functions, classes, template literals, destructuring, promises, modules
* ES7 (2016): \*\* operator, Array.prototype.includes
* ES8 (2017): async/await, Object.entries, Object.values
* ES9 (2018): Promise.finally, rest/spread in objects, async iteration
* ES10 (2019): Array.flat, Array.flatMap, Object.fromEntries, optional catch binding
* ES11 (2020): BigInt, Nullish Coalescing, Optional Chaining, globalThis, Promise.allSettled
* ES12 (2021): Logical assignment operators (&&=, ||=, ??=), WeakRefs, numeric separators
* ES13+ : Top-level await, private class fields, Array.prototype.at, temporal proposals