JavaScript Arrays: Detailed Notes

Definition and Creation

- Arrays in JavaScript are ordered collections of values that can store multiple data types
- Arrays are objects with numerical indices starting from 0
- Arrays can be created using array literals or the Array constructor

Key Characteristics

- Arrays are zero-indexed (first element is at position 0)
- Arrays are dynamic and can grow or shrink in size
- Arrays can store any data type (numbers, strings, objects, functions, other arrays)
- Arrays maintain insertion order of elements
- Array length is automatically updated when elements are added or removed

Array Creation Methods

- Array literal notation: `const fruits = ['apple', 'banana', 'orange']`
- Array constructor: `const numbers = new Array(1, 2, 3, 4)`
- Empty array: `const empty = []`
- Array.of() method: `const items = Array.of(1, 'two', {three: 3})`
- Array.from() method: Converts array-like objects to arrays

Array Properties

- length: Returns the number of elements in an array
- constructor: Returns the function that created the array
- prototype: Allows adding methods and properties to arrays

Basic Array Operations

- Accessing elements: Use square bracket notation with index (`array[index]`)
- Modifying elements: Assign new values using index (`array[index] = newValue`)
- Adding elements: Use push() method or direct index assignment
- Removing elements: Use pop(), shift(), or splice() methods
- Iterating: Use loops (for, for...of, forEach) to access each element

Array Methods for Adding/Removing Elements

- push(): Adds elements to the end of an array, returns new length
- pop(): Removes the last element, returns the removed element
- unshift(): Adds elements to the beginning, returns new length
- shift(): Removes the first element, returns the removed element
- splice(): Adds/removes elements at specified position

Array Methods for Searching Elements

- indexOf(): Returns the first index of an element, or -1 if not found
- lastIndexOf(): Returns the last index of an element, or -1 if not found
- includes(): Returns true if element exists in array, false otherwise
- find(): Returns the first element that passes a test function
- findIndex(): Returns the index of the first element that passes a test function

Array Methods for Transformation

- map(): Creates a new array by transforming each element
- filter(): Creates a new array with elements that pass a test
- reduce(): Reduces array to a single value by applying a function
- flat(): Creates a new array with sub-array elements concatenated
- flatMap(): Maps each element and flattens the result

Array Methods for Iteration

- forEach(): Executes a function on each array element
- every(): Tests if all elements pass a test function
- some(): Tests if at least one element passes a test function
- entries(), keys(), values(): Return iterator objects

Array Methods for Combining/Splitting

- concat(): Merges arrays and returns a new array
- slice(): Returns a portion of an array as a new array
- join(): Converts array to string with specified separator
- split(): (String method) Converts string to array based on separator

Array Sorting and Manipulation

- sort(): Sorts array elements in place (alphabetically by default)
- reverse(): Reverses the order of elements in place
- fill(): Fills array elements with a static value
- copyWithin(): Copies array elements to another position

Advanced Array Concepts

- Sparse arrays: Arrays with "empty slots" (undefined elements)
- Multidimensional arrays: Arrays containing other arrays
- Array destructuring: Extracting values into variables
- Spread operator: Expanding arrays into elements (`...array`)
- Rest parameters: Collecting multiple elements into an array

Array-like Objects

- Objects with a length property and indexed elements
- Converting to true arrays using Array.from() or spread operator
- Examples: DOM NodeLists, function arguments object, strings

Performance Considerations

- Arrays are optimized for numerical indices
- Large arrays may cause memory issues
- Adding/removing from beginning is slower than end (shifts all elements)
- Prefer array methods over manual iteration when possible

Array Subclassing (ES6+)

- Creating custom array types with extends
- Inherited methods and properties
- Customizing array behavior