**String Methods**

* **charAt**: Returns the character at a specified index.
* **charCodeAt**: Provides the Unicode of the character at a specified index.
* **concat**: Combines two or more strings.
* **endsWith**: Checks if a string ends with specified characters.
* **fromCharCode**: Creates a string from Unicode values.
* **includes**: Checks if a string contains specified text.
* **indexOf**: Finds the first occurrence of a specified value in a string.
* **lastIndexOf**: Finds the last occurrence of a specified value in a string.
* **localeCompare**: Compares two strings in the current locale.
* **match**: Searches a string for a match using a regular expression.
* **padEnd**: Pads a string with a specified character from the end.
* **padStart**: Pads a string with a specified character from the start.
* **repeat**: Repeats a string a specified number of times.
* **replace**: Replaces a substring with another value.
* **replaceAll**: Replaces all matches of a substring.
* **search**: Finds the position of a match using a regular expression.
* **slice**: Extracts a section of a string.
* **split**: Splits a string into an array.
* **startsWith**: Checks if a string starts with specified characters.
* **substring**: Extracts characters between two indices.
* **toLocaleLowerCase**: Converts a string to lowercase, respecting locale.
* **toLocaleUpperCase**: Converts a string to uppercase, respecting locale.
* **toLowerCase**: Converts a string to lowercase.
* **toString**: Converts an object to a string.
* **toUpperCase**: Converts a string to uppercase.
* **trim**: Removes whitespace from both ends of a string.
* **trimEnd**: Removes trailing whitespace.
* **trimStart**: Removes leading whitespace.
* **valueOf**: Returns the primitive value of a string object.

**Array Methods**

* **at**: Returns the element at a specified index.
* **concat**: Merges arrays into one.
* **copyWithin**: Copies elements within an array.
* **entries**: Returns an array iterator object with key/value pairs.
* **every**: Checks if all elements satisfy a condition.
* **fill**: Replaces array elements with a static value.
* **filter**: Creates an array with elements that satisfy a condition.
* **find**: Returns the first element matching a condition.
* **findIndex**: Returns the index of the first matching element.
* **flat**: Flattens nested arrays into a single array.
* **flatMap**: Maps and flattens the results into a new array.
* **forEach**: Executes a function for each array element.
* **includes**: Checks if an array contains a value.
* **indexOf**: Finds the first index of a value.
* **isArray**: Checks if an object is an array.
* **join**: Combines array elements into a string.
* **keys**: Returns an iterator for the array's keys.
* **lastIndexOf**: Finds the last index of a value.
* **map**: Creates a new array with transformed elements.
* **pop**: Removes and returns the last element.
* **push**: Adds elements to the end of an array.
* **reduce**: Reduces an array to a single value.
* **reduceRight**: Reduces an array from right to left.
* **reverse**: Reverses the order of elements.
* **shift**: Removes and returns the first element.
* **slice**: Extracts a section of an array.
* **some**: Checks if any elements satisfy a condition.
* **sort**: Sorts array elements.
* **splice**: Adds or removes elements from an array.
* **toLocaleString**: Converts an array to a localized string.
* **toString**: Converts an array to a string.
* **unshift**: Adds elements to the beginning of an array.
* **values**: Returns an iterator for the array's values.

**Number Methods**

* **toExponential**: Converts a number to exponential notation.
* **toFixed**: Formats a number with fixed decimal points.
* **toLocaleString**: Converts a number to a localized string.
* **toPrecision**: Formats a number to a specified precision.
* **toString**: Converts a number to a string.
* **valueOf**: Returns the primitive value of a number.

**Date Methods**

* **getDate**: Returns the day of the month (1–31).
* **getDay**: Returns the day of the week (0–6).
* **getFullYear**: Returns the year in four digits.
* **getHours**: Returns the hour (0–23).
* **getMilliseconds**: Returns milliseconds (0–999).
* **getMinutes**: Returns the minutes (0–59).
* **getMonth**: Returns the month (0–11).
* **getSeconds**: Returns the seconds (0–59).
* **getTime**: Returns the number of milliseconds since January 1, 1970.
* **getTimezoneOffset**: Returns the time zone difference in minutes.
* **getUTCDate**: Returns the UTC day of the month.
* **getUTCDay**: Returns the UTC day of the week.
* **getUTCFullYear**: Returns the UTC year.
* **getUTCHours**: Returns the UTC hour.
* **getUTCMilliseconds**: Returns the UTC milliseconds.
* **getUTCMinutes**: Returns the UTC minutes.
* **getUTCMonth**: Returns the UTC month.
* **getUTCSeconds**: Returns the UTC seconds.
* **now**: Returns the current time in milliseconds since January 1, 1970.
* **parse**: Parses a date string and returns milliseconds since January 1, 1970.
* **setDate**: Sets the day of the month.
* **setFullYear**: Sets the year (with optional month and day).
* **setHours**: Sets the hour (and optionally minutes, seconds, milliseconds).
* **setMilliseconds**: Sets the milliseconds.
* **setMinutes**: Sets the minutes (and optionally seconds, milliseconds).
* **setMonth**: Sets the month (0–11).
* **setSeconds**: Sets the seconds (and optionally milliseconds).
* **setTime**: Sets the time in milliseconds since January 1, 1970.
* **setUTCDate**: Sets the UTC day of the month.
* **setUTCFullYear**: Sets the UTC year.
* **setUTCHours**: Sets the UTC hour.
* **setUTCMilliseconds**: Sets the UTC milliseconds.
* **setUTCMinutes**: Sets the UTC minutes.
* **setUTCMonth**: Sets the UTC month.
* **setUTCSeconds**: Sets the UTC seconds.
* **toDateString**: Returns the date as a readable string.
* **toISOString**: Converts a date to ISO 8601 format.
* **toJSON**: Returns a JSON representation of the date.
* **toLocaleDateString**: Returns the date as a locale-specific string.
* **toLocaleTimeString**: Returns the time as a locale-specific string.
* **toTimeString**: Returns the time as a readable string.
* **toUTCString**: Converts a date to a string in UTC format.
* **valueOf**: Returns the primitive value of the date object.

**Math Methods**

* **abs**: Returns the absolute value of a number.
* **acos**: Returns the arccosine of a number.
* **acosh**: Returns the hyperbolic arccosine of a number.
* **asin**: Returns the arcsine of a number.
* **asinh**: Returns the hyperbolic arcsine of a number.
* **atan**: Returns the arctangent of a number.
* **atan2**: Returns the angle of a coordinate in radians.
* **atanh**: Returns the hyperbolic arctangent of a number.
* **cbrt**: Returns the cube root of a number.
* **ceil**: Rounds up to the nearest integer.
* **clz32**: Counts leading zeroes in a 32-bit binary representation.
* **cos**: Returns the cosine of a number.
* **cosh**: Returns the hyperbolic cosine of a number.
* **exp**: Returns e^x, where x is the argument.
* **expm1**: Returns e^x - 1.
* **floor**: Rounds down to the nearest integer.
* **fround**: Returns the nearest 32-bit float representation.
* **hypot**: Returns the square root of the sum of squares of arguments.
* **imul**: Performs 32-bit integer multiplication.
* **log**: Returns the natural logarithm of a number.
* **log1p**: Returns the natural logarithm of 1 + x.
* **log2**: Returns the base-2 logarithm of a number.
* **log10**: Returns the base-10 logarithm of a number.
* **max**: Returns the maximum of the given numbers.
* **min**: Returns the minimum of the given numbers.
* **pow**: Raises a number to a specified power.
* **random**: Returns a pseudo-random number between 0 and 1.
* **round**: Rounds a number to the nearest integer.
* **sign**: Returns the sign of a number (positive, negative, or zero).
* **sin**: Returns the sine of a number.
* **sinh**: Returns the hyperbolic sine of a number.
* **sqrt**: Returns the square root of a number.
* **tan**: Returns the tangent of a number.
* **tanh**: Returns the hyperbolic tangent of a number.
* **trunc**: Returns the integer part of a number.

**DOM Methods**

**Manipulation**

* **getElementById**: Selects an element by its ID.
* **getElementsByClassName**: Selects all elements with the specified class name.
* **getElementsByTagName**: Selects all elements with the specified tag name.
* **querySelector**: Selects the first element matching the CSS selector.
* **querySelectorAll**: Selects all elements matching the CSS selector.
* **createElement**: Creates a new HTML element.
* **createTextNode**: Creates a text node.
* **appendChild**: Adds a child node to a parent node.
* **removeChild**: Removes a child node from a parent node.
* **replaceChild**: Replaces an existing child node with a new one.
* **cloneNode**: Creates a duplicate of a node.
* **insertBefore**: Inserts a node before another child node.
* **setAttribute**: Sets a specific attribute on an element.
* **getAttribute**: Retrieves the value of a specific attribute on an element.
* **removeAttribute**: Removes a specific attribute from an element.

**Event Handling**

* **addEventListener**: Attaches an event handler to an element.
* **removeEventListener**: Removes an event handler from an element.
* **dispatchEvent**: Manually triggers an event on an element.

**Traversal and Nodes**

* **parentNode**: Returns the parent node of an element.
* **childNodes**: Returns a collection of child nodes.
* **firstChild**: Returns the first child node.
* **lastChild**: Returns the last child node.
* **nextSibling**: Returns the next sibling node.
* **previousSibling**: Returns the previous sibling node.

**Object Methods**

* **assign**: Copies properties from one or more objects to a target object.
* **create**: Creates a new object with the specified prototype.
* **defineProperty**: Adds or modifies a property on an object.
* **defineProperties**: Adds or modifies multiple properties on an object.
* **entries**: Returns an array of key-value pairs for an object.
* **freeze**: Freezes an object, preventing modifications.
* **fromEntries**: Creates an object from an array of key-value pairs.
* **getOwnPropertyDescriptor**: Retrieves a descriptor for a property on an object.
* **getOwnPropertyDescriptors**: Retrieves descriptors for all properties on an object.
* **getOwnPropertyNames**: Returns an array of property names of an object.
* **getOwnPropertySymbols**: Returns an array of symbol properties of an object.
* **getPrototypeOf**: Retrieves the prototype of an object.
* **hasOwnProperty**: Checks if an object has a specific property.
* **is**: Determines if two values are the same.
* **isExtensible**: Checks if an object is extensible.
* **isFrozen**: Checks if an object is frozen.
* **isSealed**: Checks if an object is sealed.
* **keys**: Returns an array of an object's property names.
* **preventExtensions**: Prevents new properties from being added to an object.
* **seal**: Seals an object, making properties non-configurable.
* **setPrototypeOf**: Sets the prototype of an object.
* **values**: Returns an array of an object's property values.

**ES6 and Beyond**

**Set Methods**

* **add**: Adds a value to the set.
* **clear**: Removes all elements from the set.
* **delete**: Removes a specific value from the set.
* **entries**: Returns an iterator with [value, value] pairs.
* **forEach**: Executes a function for each set value.
* **has**: Checks if a value exists in the set.
* **keys**: Same as values for sets.
* **values**: Returns an iterator for all values in the set.

**Map Methods**

* **set**: Adds a key-value pair to the map.
* **get**: Retrieves the value associated with a key.
* **has**: Checks if a key exists in the map.
* **delete**: Removes a key-value pair by key.
* **clear**: Removes all key-value pairs from the map.
* **keys**: Returns an iterator for all keys.
* **values**: Returns an iterator for all values.
* **entries**: Returns an iterator for key-value pairs.

**Promise Methods**

* **all**: Resolves when all promises resolve or rejects if any promise rejects.
* **allSettled**: Resolves when all promises settle (fulfilled or rejected).
* **any**: Resolves when any promise resolves or rejects if all promises reject.
* **catch**: Handles promise rejections.
* **finally**: Executes a callback after promise resolution or rejection.
* **race**: Resolves or rejects as soon as one promise settles.
* **reject**: Returns a rejected promise.
* **resolve**: Returns a resolved promise.
* **then**: Attaches callbacks for promise fulfillment and rejection.

**JSON and AJAX**

**JSON**

* **parse**: Converts a JSON string into a JavaScript object.
* **stringify**: Converts a JavaScript object into a JSON string.

**AJAX**

* **open**: Initializes a request for an AJAX call.
* **send**: Sends the AJAX request.
* **abort**: Cancels an ongoing AJAX request.
* **setRequestHeader**: Sets a request header for the AJAX call.
* **onreadystatechange**: Defines a function to execute when the ready state changes.