Cheat Sheet for comprehensive JavaScript

Variables and Data Types

Variables

- **`var`**: Function-scoped, can be redeclared.

```
var x = 10;
```

- **`let`**: Block-scoped, can be reassigned.

```
let y = 20;
```

- **`const`**: Block-scoped, cannot be reassigned.

```
const z = 30;
```

Data Types

- **Primitive Types**:
  - **Number**: `let num = 42;`
  - **String**: `let str = "Hello";`
  - **Boolean**: `let bool = true;`
  - **Null**: `let n = null;`
  - **Undefined**: `let u = undefined;`
  - **Symbol**: `let sym = Symbol('foo');`
- **Complex Types**:
  - **Object**: `let obj = { key: 'value' };`
  - **Array**: `let arr = [1, 2, 3];`
  - **Function**: `let func = function() { return 'Hello'; };`

Operators

Arithmetic Operators

- `+` (Addition)
- `-` (Subtraction)
- `*` (Multiplication)
- `/` (Division)
- `%` (Modulus)

- `**` (Exponentiation)

Comparison Operators

- `==` (Equality)
- `===` (Strict Equality)
- `!=` (Inequality)
- `!==` (Strict Inequality)
- `>` (Greater than)
- `<` (Less than)
- `>=` (Greater than or equal to)
- `<=` (Less than or equal to)

Logical Operators

- `&&` (Logical AND)
- `||` (Logical OR)
- `!` (Logical NOT)

Control Structures

Conditional Statements

- **`if` Statement**:

```
if (condition) {
  // code
}
```

- **`else if` Statement**:

```
if (condition1) {
  // code
} else if (condition2) {
  // code
}
```

- **`else` Statement**:

```
if (condition) {
  // code
} else {
  // code
}
```

- **`switch` Statement**:

```
switch (expression) {
  case value1:
    // code
    break;
  case value2:
    // code
    break;
  default:
    // code
}
```

Loops

- **`for` Loop**:

```
for (let i = 0; i < 10; i++) {
  // code
}
```

- **`while` Loop**:

```
while (condition) {
  // code
}
```

- **`do...while` Loop**:

```
do {
  // code
} while (condition);
```

- **`for...in` Loop**:

```
  for (let key in object) {
    // code
  }
```

- **`for...of` Loop**:

```
  for (let value of array) {
    // code
  }
```

Functions

Function Declaration

```
function functionName(parameters) {
  // code
  return result;
}
```

Function Expression

```
const functionName = function(parameters) {
  // code
  return result;
};
```

Arrow Functions

```
const functionName = (parameters) => {
  // code
  return result;
};
```

Immediately Invoked Function Expression (IIFE)

```
(function() {
  // code
})();
```

Objects and Arrays

Objects

- **Creating Objects**:

```
let obj = { key1: 'value1', key2: 'value2' };
```

- **Accessing Properties**:

```
obj.key1; // Dot notation
obj['key2']; // Bracket notation
```

- **Adding/Updating Properties**:

```
obj.key3 = 'value3';
```

- **Deleting Properties**:

```
delete obj.key1;
```

Arrays

- **Creating Arrays**:

```
let arr = [1, 2, 3];
```

- **Accessing Elements**:

```
arr[0]; // First element
```

- **Adding Elements**:

```
arr.push(4); // Adds to the end
arr.unshift(0); // Adds to the beginning
```

- **Removing Elements**:

```
arr.pop(); // Removes from the end
arr.shift(); // Removes from the beginning
```

- **Slicing and Splicing**:

```
arr.slice(1, 3); // Returns a new array
arr.splice(1, 2); // Modifies the original array
```

ES6+ Features

Template Literals

```
let name = 'John';
let greeting = `Hello, ${name}!`;
```

Destructuring

- **Array Destructuring**:

```
let [a, b] = [1, 2];
```

- **Object Destructuring**:

```
let { key1, key2 } = { key1: 'value1', key2: 'value2' };
```

Spread/Rest Operators

- **Spread Operator**:

```
let arr1 = [1, 2];
let arr2 = [...arr1, 3, 4]; // [1, 2, 3, 4]
```

- **Rest Operator**:

```
function sum(...numbers) {
  return numbers.reduce((a, b) => a + b, 0);
}
```

Promises and Async/Await

- **Promises**:

```
let promise = new Promise((resolve, reject) => {
  // code
});
```

- **Async/Await**:

```
async function fetchData() {
  let response = await fetch('url');
  let data = await response.json();
  return data;
}
```

Error Handling

`try...catch` Statement

```
try {
  // code
} catch (error) {
  // error handling
} finally {
  // optional final block
}
```

DOM Manipulation

Selecting Elements

- **`getElementById`**:

```
document.getElementById('elementId');
```

- **`querySelector`**:

```
document.querySelector('.className');
```

- **`querySelectorAll`**:

```
document.querySelectorAll('p');
```

Modifying Elements

- **Changing Text**:

```
element.textContent = 'New Text';
```

- **Changing HTML**:

```
element.innerHTML = '<p>New HTML</p>';
```

- **Changing Attributes**:

```
element.setAttribute('attributeName', 'attributeValue');
```

Event Handling

- **Adding Event Listeners**:

```
element.addEventListener('click', function() {
  // code
});
```

- **Removing Event Listeners**:

```
element.removeEventListener('click', functionName);
```

Best Practices

Code Optimization

- **Minimize DOM Access**: Cache DOM elements.
- **Use Efficient Loops**: Prefer `for` loops over `forEach` for large arrays.
- **Avoid Global Variables**: Use closures and modules.

Debugging

- **`console.log`**:

```
console.log('Debugging message');
```

- **`debugger`**:

```
debugger;
```

Performance Tips

- **Avoid Inline Scripts**: Use external scripts.

- **Optimize Images**: Compress and use appropriate formats.

- **Use Web Workers**: For CPU-intensive tasks.

Libraries and Frameworks

Common Libraries

- **jQuery**: Simplifies DOM manipulation and AJAX.

- **Lodash**: Utility library for common tasks.

- **Moment.js**: Date manipulation library.

Popular Frameworks

- **React**: For building user interfaces.

- **Angular**: Full-featured framework for web apps.

- **Vue.js**: Progressive framework for building user interfaces.

Additional Resources

Documentation

- **MDN Web Docs**: https://developer.mozilla.org/

- **W3Schools**: https://www.w3schools.com/

Online Editors

- **JSFiddle**: https://jsfiddle.net/

- **CodePen**: https://codepen.io/

Communities

- **Stack Overflow**: https://stackoverflow.com/

- **Reddit**: https://www.reddit.com/r/javascript/

This cheat sheet covers the essential aspects of JavaScript, from basic syntax to advanced features and best practices. Use it as a quick reference to enhance your JavaScript skills.

By Ahmed Baheeg Khorshid

ver 1.0