🚀 Advanced JavaScript & CSS: A Developer's Guide

✅ Table of Contents

□ ADVANCED JAVASCRIPT

1⃣ Closures

A closure is when a function "remembers" its outer scope even after the outer function has finished executing.

Example:

javascript

CopyEdit

```
function outer() {
```

```
  let count = 0;
  return function inner() {
    count++;
    console.log(count);
  };
}
const counter = outer();
counter(); // 1
counter(); // 2
```

Use closures for data privacy and function factories.

2️⃣ Asynchronous JavaScript

- **Callbacks:**

```
javascript
CopyEdit
function fetchData(callback) {
  setTimeout(() => {
    callback('Data loaded!');
  }, 1000);
}
fetchData(data => console.log(data));
```

- **Promises:**

```
javascript
CopyEdit
const promise = new Promise((resolve, reject) => {
  resolve('Success!');
});
promise.then(data => console.log(data));
```

- **Async/Await:**

```
javascript
CopyEdit
```

```
async function getData() {
  let res = await fetch('https://api.example.com');
  let data = await res.json();
  console.log(data);
}
getData();
```

3️⃣ ES6+ Features

- let & const
- Arrow functions: const add = (a,b) => a+b
- Destructuring: const {name, age} = user
- Spread/Rest: [...arr] or (...args) => {}
- Default parameters: function(x=1){}
- Template literals: `Hello ${name}`

4️⃣ Modules & Imports

Modularize your code:

javascript

CopyEdit

```
// math.js
export function add(x, y) { return x + y; }

// app.js
import { add } from './math.js';
```

5️⃣ The Event Loop

JavaScript is single-threaded; the **Event Loop** handles async tasks (microtasks, macrotasks).
Understand setTimeout, Promises, and how the call stack & callback queue work.

6️⃣ Prototypes & Inheritance

JavaScript uses prototypal inheritance.

javascript

CopyEdit

```
function Person(name) {
  this.name = name;
}
Person.prototype.greet = function() {
  console.log(`Hello, ${this.name}`);
};
const john = new Person('John');
john.greet();
```

7️⃣ Error Handling

- try...catch blocks:

javascript

CopyEdit

```
try {
  throw new Error('Oops!');
} catch(e) {
  console.error(e.message);
}
```

- Use finally for cleanup.

8️⃣ Design Patterns Basics

Understand patterns like:

- Module pattern
- Singleton pattern
- Observer pattern
- Factory pattern

9️⃣ Testing & Debugging Tips

- Use console.log() or debugger

- Jest for unit testing
- Chrome DevTools for step debugging & network inspection

🔟 Best Practices & Resources

✅ Keep code modular
✅ Use ESLint or Prettier
✅ Handle errors gracefully
✅ MDN Web Docs, JavaScript.info

🎨 ADVANCED CSS

1️⃣ Flexbox Deep Dive

Master alignment, ordering, and distribution.

```
css

CopyEdit

.container {

  display: flex;

  justify-content: space-between;

  align-items: center;

}
```

Properties: flex-direction, flex-wrap, flex-basis, flex-grow.

2️⃣ CSS Grid Layout

Powerful 2D layout system.

```
css

CopyEdit

.container {

  display: grid;

  grid-template-columns: repeat(3, 1fr);

  grid-gap: 20px;

}
```

Combine Grid & Flexbox for advanced layouts.

3️⃣ CSS Variables

Reusable values!

```
css
CopyEdit
:root {
  --main-color: #3498db;
}
.button {
  color: var(--main-color);
}
```

4️⃣ Advanced Selectors

- :nth-child(), :not(), [attribute^="val"]
- Combinators: >, +, ~

5️⃣ Animations & Transitions

```
css
CopyEdit
.box {
  transition: transform 0.3s ease;
}
.box:hover {
  transform: scale(1.1);
}

@keyframes fadeIn {
  from { opacity: 0; }
  to { opacity: 1; }
}
```

6️⃣ Responsive Design & Media Queries

css

CopyEdit

@media (max-width: 768px) {

  body { font-size: 14px; }

}

Use relative units (em, rem, %).

7️⃣ CSS Architecture

Organize code:

- **BEM:** Block__Element--Modifier
- **SMACSS, OOCSS:** Write reusable, modular styles.

8️⃣ CSS Preprocessors

SASS/SCSS: Variables, nesting, mixins.

scss

CopyEdit

$main-color: #333;

.nav {

  background: $main-color;

}

9️⃣ Performance Tips

✅ Minify CSS
✅ Use critical CSS
✅ Avoid excessive specificity
✅ Combine and compress assets

🔟 Helpful Tools & Resources

✅ MDN CSS, CSS-Tricks
✅ Can I Use (browser support)

✅ Flexbox Froggy & Grid Garden (games!)
✅ PostCSS, Autoprefixer

Final Tip

Advanced front-end devs write **clean, maintainable, scalable code** — practice often and stay updated! 🚀