Garratt's JavaScript Cheatsheet

A practical JavaScript reference for modern web projects. Focused on patterns you'll use in blogs, static sites, and apps: data, async, DOM, and modules.

Basics and Modern Syntax

Concept	Use	Example
const / let	Prefer const, use let when reassigned	const site = "gcampton"; let views = 0; views += 1;
Template strings	Readable interpolation	const url = `/blog/${slug}`;
Destructuring	Pick fields cleanly	const { title, date } = post;
Default params	Safe function defaults	function greet(name = "friend") { … }
Optional chaining	Avoid crashes on null	const city = user?.profile?.city;

Types and Equality

Topic	Rule of thumb	Example
typeof	Quick type checks	typeof x === "string";
Arrays	Use Array.isArray	Array.isArray(items);
Strict equality	Use === and !==	0 === false; // false
Nullish coalescing	Fallback only for null/undefined	const name = input ?? "Untitled";

Functions

Pattern	Use	Example
Arrow fn	Short callbacks	const add = (a, b) => a + b;
Named fn	Better stack traces	function slugify(s) { return s.toLowerCase(); }
Rest params	Variadic args	function sum(...nums) { return nums.reduce((a,n)=>a+n,0); }
IIFE	Local scope	(() => { /* setup */ })();

Arrays

Method	Use	Example
map	Transform	posts.map(p => p.title);
filter	Keep matching	posts.filter(p => p.published);
find	First match	posts.find(p => p.slug === s);
reduce	Accumulate	posts.reduce((acc,p)=>acc+p.views,0);
sort	Custom ordering	posts.sort((a,b)=>b.date.localeCompare(a.date));

Objects

Pattern	Use	Example
Shallow copy	Clone with spread	const copy = { ...post };
Merge	Combine objects	const merged = { ...a, ...b };
Computed key	Dynamic property names	const obj = { [key]: value };
Object.entries	Iterate key/value	for (const [k,v] of Object.entries(meta)) { … }

Async and Promises

Tool	Use	Example
async/await	Readable async flow	const res = await fetch(url); const data = await res.json();
try/catch	Handle failures	try { … } catch (err) { console.error(err); }
Promise.all	Run in parallel	const [a,b] = await Promise.all([fa(), fb()]);
Timeout	Abort slow work	const ctl = new AbortController(); setTimeout(()=>ctl.abort(), 8000);

DOM Essentials

Task	Use	Example
Select	Grab elements safely	`const el = document.querySelector(".card");`
Create	Build nodes	`const li = document.createElement("li");` `li.textContent = title;`
Insert	Append/prepend/replace	`list.append(li);` `el.replaceChildren(list);`
Events	Click, input, submit	`btn.addEventListener("click", () => { … });`

fetch and JSON

Case	Use	Example
GET JSON	Load data	`const res = await fetch("/api/posts");` `const posts = await res.json();`

POST JSON	Send data	`await fetch("/api/subscribe", {`
		` method: "POST",`
		` headers: { "Content-Type": "application/json" },`
		` body: JSON.stringify({ email })`
		`});`

Check status	Avoid silent failures	`if (!res.ok) throw new Error(`HTTP ${res.status}`);`

Modules and Tooling

Concept	Use	Example
Export	Expose functions	`export function slugify(s) { … }`
Default export	Single main export	`export default function Page() { … }`
Import	Use modules	`import { slugify } from "./slugify.js";`
npm scripts	Common commands	`"scripts": { "dev": "vite", "build": "vite build" }`

Debug and Quality

Tip	Why	Example
console.*	Inspect quickly	`console.log({ x, y });` `console.table(posts);`
Guard clauses	Reduce nesting	`if (!user) return;` `if (!user.email) return;`
Prefer early returns	Readable logic	`if (res.ok) return data;` `throw new Error("Bad response");`
Lint + format	Consistent style	eslint + prettier (or biome) in CI