JavaScript notes

**SUMMARY OF TYPES & CONCEPTS**

| **Concept** | **Type** | **Example** |
| --- | --- | --- |
| document.querySelector(...) | Element or null | Selects one matching element |
| document.querySelectorAll() | NodeList | Collection of elements (array-like) |
| window.innerHeight | Number | Current viewport height |
| Element.getBoundingClientRect() | DOMRect object | Contains .top, .left, etc. for position |
| classList.add() | Method | Adds class to element |
| classList.toggle() | Method | Adds or removes a class based on a condition |
| forEach() | Method on NodeList/Array | Loops through list of elements |

Summary of Code

**Concept Description**

querySelector Selects the first element that matches a CSS selector

querySelectorAll Selects all matching elements and returns a NodeList

addEventListener Attaches a function to run when a specific event occurs

scrollTop Tells you how far the user has scrolled vertically

classList.add() Adds a CSS class to an element

classList.toggle() Adds or removes a class based on a condition

getBoundingClientRect().top Gets distance of an element from the top of the viewport

NodeList.forEach() Runs a function for every element in a list

window.innerHeight Returns the height of the viewport in pixels

**1. Which is an Array, Function, or Fat Arrow Function?**

| **Code** | **Type** | **Description** |
| --- | --- | --- |
| [] | **Array literal** | Used to create an array (e.g., const arr = [1, 2, 3]) |
| function animateOnScroll() {} | **Function declaration** | A traditional function that can be hoisted |
| el => { ... } | **Fat arrow function** (aka Arrow Function) | A more concise way to write functions |
| document.querySelectorAll(...) | **Returns a NodeList**, not an Array | It's *array-like* and can be looped with .forEach() but lacks some Array methods unless converted using Array.from() |
| window.addEventListener(...) | **Function call** | Calling a built-in method that expects a function or arrow function as argument |

**Arrow functions** (el => {...}) are especially common in short callbacks, like .forEach().

**2. What does document.body.scrollTop || document.documentElement.scrollTop do?**

**Explanation:**

This line returns how far the user has scrolled down vertically in the browser.

const scrollTop = document.body.scrollTop || document.documentElement.scrollTop;

**What's going on:**

* document.body.scrollTop: works in **some browsers** (like Safari).
* document.documentElement.scrollTop: works in **others** (like Chrome, Firefox, Edge).

**|| means: “use the first truthy value.”**

So this line ensures cross-browser compatibility. Whichever is valid (not 0 or undefined) will be used.

**3. What does the dollar sign (${}) do?**

This is part of **template literals** (backtick strings) in JavaScript:

console.log(`Window height is: ${windowHeight}`);

**It lets you:**

* **Insert a variable or expression** directly inside a string
* Without breaking it up with +

Equivalent to:

console.log("Window height is: " + windowHeight);

But using `backticks` and ${} is **cleaner and more modern**.

**4. What does const elementTop = el.getBoundingClientRect().top; mean?**

**Step-by-step:**

* el: is the element in the loop (like a <p> or <h1>).
* getBoundingClientRect(): is a method that gives you the element’s **position & size** relative to the **viewport**.
* .top: gives the **number of pixels** from the **top of the viewport** to the top of the element.

const elementTop = el.getBoundingClientRect().top;

**If elementTop is less than window.innerHeight**, the element is on screen (or about to enter).

**5. const windowHeight = window.innerHeight;**

**What it does:**

**This line gets the height of the visible area of the browser window (the *viewport*), in pixels — *not* including any scrollbars, dev tools, or off-screen content.**

* **window: refers to the global browser window object.**
* **.innerHeight: returns a number representing the pixel height of the viewport.**

**This is useful because:**

**Typically use this in scroll-based animations or layout logic to compare where an element is in relation to what's visible on screen.**

**TL;DR Cheat Sheet**

| **Code** | **Type** | **Purpose** |
| --- | --- | --- |
| [] | Array | List of values |
| function name() {} | Function | Reusable block of code |
| () => {} | Fat arrow function | Short syntax function |
| `scrollTop = body.scrollTop |  | docEl.scrollTop` |
| `${x}` | Template literal | String with embedded variables |
| .getBoundingClientRect().top | Number | Distance from element to top of viewport |