perplexity

Arrays in C++ vs JavaScript: Contiguous Memory Locations

C++ Arrays

- **Contiguous Memory**: Arrays in C++ are always stored in contiguous memory locations. This means each element sits right next to its neighbor in memory, with no gaps in between. For example, an array of five integers occupies a single, unbroken block of memory sized for five integers, regardless of values stored.[1] [2] [3]
- **Fixed Size & Data Type**: The size of a C++ array is fixed when declared, and all elements must be the same type.
- **Access**: Knowing the base address of the array allows quick (constant-time) indexing because element offsets are simply $ base address + index \times element size $.
- **Efficiency**: This contiguous layout supports efficient iteration, quick access, and data locality, leading to better performance in computations and cache usage.

JavaScript Arrays

- **Non-Contiguous Storage**: Arrays in JavaScript are fundamentally different. Under the hood, they are objects with integer keys and are **not guaranteed** to be stored in contiguous memory. The language standard does not specify any fixed memory layout, and actual storage is up to the engine.[4] [5] [6]
    - **Sparse Arrays**: JavaScript arrays can have "holes" (missing values) and can be gigantic in length but small in actual stored elements—memory is only allocated for explicit values.[6] [4]
    - **Dynamic and Heterogeneous**: They can grow or shrink in size and allow elements of different types within the same array.
    - **Engine Optimizations**: Some JavaScript engines (like V8 in Chrome) do try to optimize storage by using contiguous blocks (much like C++ backing arrays) if the array is dense and contains only numbers, but this is not required or reliable across all usage. As soon as the array becomes sparse or heterogeneous, this optimization falls away.[7]

Table: Key Differences

Feature	C++ Array	JavaScript Array
Memory Layout	Always contiguous[1] [2] [5]	Not guaranteed, typically not contiguous[5] [6]
Size	Fixed at compile-time	Dynamic, can grow or shrink

Feature	C++ Array	JavaScript Array
Data Type	Homogeneous (single type)	Heterogeneous (any types per index)
Indexing Efficiency	O(1), pointer arithmetic	O(1) for dense, may vary for sparse
Gaps ("holes")	Not possible	Possible (sparse arrays are allowed) [4] [6]
Storage Optimizations	None, always contiguous	Engine may optimize dense arrays [7], but not required

Special Case: JavaScript Typed Arrays

For **typed arrays** in JavaScript (like `Uint8Array`, `Float32Array`), the elements **are** stored in contiguous memory for performance reasons, mimicking the layout and speed of C++ arrays. Typed arrays are different from regular JavaScript arrays and are intended for efficient manipulation of binary data.[8]

Summary:

- In C++, arrays always occupy a contiguous block of memory.
- In JavaScript, normal arrays do not have a defined contiguous memory layout and are implemented as objects, though engines may optimize for some use cases. Use JS Typed Arrays for C++-like contiguous blocks.

⁂

1. https://www.geeksforgeeks.org/cpp/cpp-arrays/
2. https://www.geeksforgeeks.org/cpp/one-dimensional-arrays-in-cpp/
3. https://trainings.internshala.com/blog/cpp-array/
4. https://www.reddit.com/r/learnjavascript/comments/14mbm7o/arrays_in_javascript/
5. https://www.geeksforgeeks.org/dsa/how-array-is-stored-in-different-programming-languages/
6. https://stackoverflow.com/questions/20321047/how-are-javascript-arrays-represented-in-physical-memory
7. https://ryanpeden.com/how-do-javascript-arrays-work-under-the-hood/
8. https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Typed_arrays