# V8 release v9.2

Published 16 July 2021 · Tagged with [release](https://v8.dev/blog/tags/release)

Every six weeks, we create a new branch of V8 as part of our [release process](https://v8.dev/docs/release-process). Each version is **branched** from V8’s Git master immediately before **a Chrome Beta milestone**. Today we’re pleased to announce our newest branch, [V8 version 9.2](https://chromium.googlesource.com/v8/v8.git/+log/branch-heads/9.2), which is in beta until its release in coordination with Chrome 92 Stable in several weeks. V8 v9.2 is filled with all sorts of developer-facing goodies. This post provides a preview of some of the highlights in anticipation of the release.

## JavaScript

### at method

The new at method is now available on Arrays, TypedArrays, and Strings. When passed a negative value, it performs relative indexing from the end of the indexable. When passed a positive value, it behaves identically to property access. For example, [1,2,3].at(-1) is 3. See more at [our explainer](https://v8.dev/features/at-method).

## Shared Pointer Compression Cage

V8 supports [pointer compression](https://v8.dev/blog/pointer-compression) on 64-bit platforms including x64 and arm64. This is achieved by splitting a 64-bit pointer into two halves. The upper 32-bits can be thought of as a base while the lower 32-bits can be thought of as an index into that base.

|----- 32 bits -----|----- 32 bits -----|

Pointer: |\_\_\_\_\_\_\_\_base\_\_\_\_\_\_\_|\_\_\_\_\_\_\_index\_\_\_\_\_\_\_|

Currently, an Isolate performs all allocations in the GC heap within a 4GB virtual memory "cage", which ensures that all pointers have the same upper 32-bit base address. With the base address held constant, 64-bit pointers can be passed around only using the 32-bit index, since the full pointer can be reconstructed.

With v9.2, the default is changed such that all Isolates within a process share the same 4GB virtual memory cage. This was done in anticipation of prototyping experimental shared memory features in JS. With each worker thread having its own Isolate and therefore its own 4GB virtual memory cage, pointers could not be passed between Isolates with a per-Isolate cage as they did not share the same base address. This change has the additional benefit of reducing virtual memory pressure when spinning up workers.

The tradeoff of the change is that the total V8 heap size across all threads in a process is capped at a maximum 4GB. This limitation may be undesirable for server workloads that spawn many threads per process, as doing so will run out of virtual memory faster than before. Embedders may turn off sharing of the pointer compression cage with the GN argument v8\_enable\_pointer\_compression\_shared\_cage = false.

## V8 API

Please use git log branch-heads/9.1..branch-heads/9.2 include/v8.h to get a list of the API changes.

Developers with an active V8 checkout can use git checkout -b 9.2 -t branch-heads/9.2 to experiment with the new features in V8 v9.2. Alternatively you can [subscribe to Chrome’s Beta channel](https://www.google.com/chrome/browser/beta.html) and try the new features out yourself soon.



Posted by Ingvar Stepanyan ([@RReverser](https://twitter.com/RReverser)).