## NAME

**v8** — Google’s JavaScript engine

## SYNOPSIS

**shell** [*options*] [**−−shell**] [*file ...*]<br>
**d8** [*options*] [**−e** *string*] [**−−shell**] [[**−−module**] *file...*]

## OPTIONS

**−−** Capture all remaining arguments in JavaScript.

**−e** *string*
Execute *string* in V8.

**−−shell**
Run an interactive JavaScript shell.

**−−help**
Print flags and usage message to stdout, then exit.

**−−module**
Execute *file* as a JavaScript module.

Note: the **--module** option is implicitly enabled for `*.mjs` files.

## V8 FLAGS

Option names beginning with “**no-**” pertain to a boolean V8 flag enabled by default.

**−−abort-on-uncaught-exception**
Abort program (dump core) when an uncaught exception is thrown.

**−−no-allocation-site-pretenuring**
Don’t pretenure with allocation sites.

**−−allow-natives-syntax**
Allow natives syntax.

**−−allow-unsafe-function-constructor**
Allow invoking the function constructor without security checks.

**−−always-compact**
Perform compaction on every full GC.

**−−always-opt**
Always try to optimise functions.

**−−always-osr**
Always try to OSR functions.

**−−no-analyze-environment-liveness**
Don’t analyse liveness of environment slots and zap dead values.

**−−arm-arch**

**Type:** String
**Default:** `“armv8”`

Generate instructions for the selected ARM architecture if available: armv6, armv7, armv7+sudiv or armv8.

**−−asm-wasm-lazy-compilation**

Enable lazy compilation for asm-wasm modules.

**−−assert-types**

Generate runtime type assertions to test the typer.

**−−assume-asmjs-origin**

Force WASM decoder to assume input is internal asm-wasm format.

**−−no-async-stack-traces**

Exclude async stack traces in `Error.stack`.

**−−block-concurrent-recompilation**

Block queued jobs until released.

**−−budget-for-feedback-vector-allocation**

**Type:** Int
**Default:** `1024`

The budget in amount of bytecode executed by a function before we decide to allocate feedback vectors.

**−−builtins-in-stack-traces**

Show built-in functions in stack traces.

**−−no-cache-prototype-transitions**

Don’t cache prototype transitions.

**−−check-icache**

Check icache flushes in ARM and MIPS simulator.

**−−clear-exceptions-on-js-entry**

Clear pending exceptions when entering JavaScript.

**−−clear-free-memory**

Initialise free memory with 0.

**−−code-comments**

Emit comments in code disassembly; for more readable source positions you should add `--no-concurrent-recompilation`.

**−−no-compact-code-space**
Don’t compact code space on full collections.

**−−no-compilation-cache**
Disable compilation cache.

**−−compiler-dispatcher**
Enable compiler dispatcher.

**−−no-concurrent-array-buffer-freeing**
Don’t free array buffer allocations on a background thread.

**−−concurrent-inlining**
Run optimising compiler’s inlining phase on a separate thread.

**−−no-concurrent-marking**
Don’t use concurrent marking.

**−−no-concurrent-recompilation**
Force synchronous optimisation of hot functions.

**−−concurrent-recompilation-delay**
**Type:** Int
**Default:** `0`

Artificial compilation delay in ms.

**−−concurrent-recompilation-queue-length**
**Type:** Int
**Default:** `8`

The length of the concurrent compilation queue.

**−−no-concurrent-store-buffer**
Don’t use concurrent store buffer processing.

**−−no-concurrent-sweeping**
Don’t use concurrent sweeping.

**−−correctness-fuzzer-suppressions**
Suppress certain unspecified behaviours to ease correctness fuzzing: abort program when the stack overflows or a string exceeds maximum length (as opposed to throwing `RangeError`), and use a fixed suppression string for error messages.

**−−cpu-profiler-sampling-interval**
**Type:** Int
**Default:** `1000`

CPU profiler sampling interval in microseconds.

**−−csa-trap-on-node**

**Type:** String
**Default:** `NULL`

Trigger break point when a node with given id is created in given stub. The format is:`StubName,NodeId`.

**−−debug-code**

Generate extra code (assertions) for debugging.

**−−debug-sim**

Enable debugging the simulator.

**−−deopt-every-n-times**

**Type:** Int
**Default:** `0`

Deoptimise every n times a deopt point is passed.

**−−detailed-error-stack-trace**

Include arguments for each function call in the error stack frames array.

**−−detailed-line-info**

Always generate detailed line information for CPU profiling.

**−−no-detect-ineffective-gcs-near-heap-limit**

Don’t trigger out-of-memory failure to avoid GC storm near heap limit.

**−−disable-abortjs**

Disable AbortJS runtime function.

**−−disable-old-api-accessors**

Disable old-style API accessors whose setters trigger through the prototype chain.

**−−disallow-code-generation-from-strings**

Disallow `eval` and friends.

**−−dump-counters**

Dump counters on exit.

**−−dump-counters-nvp**

Dump counters as name-value pairs on exit.

**−−dump-wasm-module-path**

**Type:** String
**Default:** `NULL`

Directory to dump WASM modules to.

**−−embedded-src**

**Type:** String
**Default:** `NULL`

Path for the generated embedded data file. (`mksnapshot` only).

**−−embedded-variant**

**Type:** String
**Default:** `NULL`

Label to disambiguate symbols in embedded data file. (`mksnapshot` only).

**−−enable-32dregs**

**Type:** maybe_bool
**Default:** `unset`

Deprecated (use `--arm-arch` instead).

**−−enable-armv7**

**Type:** maybe_bool
**Default:** `unset`

Deprecated (use `--arm-arch` instead).

**−−enable-armv8**

**Type:** maybe_bool
**Default:** `unset`

Deprecated (use `--arm-arch` instead).

**−−no-enable-avx**

Disable use of AVX instructions if available.

**−−no-enable-bmi1**

Disable use of BMI1 instructions if available.

**−−no-enable-bmi2**

Disable use of BMI2 instructions if available.

**−−no-enable-fma3**

Disable use of FMA3 instructions if available.

**−−no-enable-lazy-source-positions**

Don’t skip generating source positions during initial compile, but regenerate when actually required.

**−−no-enable-lzcnt**

Disable use of LZCNT instruction if available.

**−−enable-neon**

**Type:** maybe_bool
**Default:** `unset`

Deprecated (use `--arm-arch` instead).

**−−no-enable-one-shot-optimization**

Disable size optimisations for the code that will only be executed once.

**−−no-enable-popcnt**

Disable use of POPCNT instruction if available.

**−−no-enable-regexp-unaligned-accesses**

Disable unaligned accesses for the regexp engine.

**−−no-enable-sahf**

Disable use of SAHF instruction if available (X64 only).

**−−enable-source-at-csa-bind**

Include source information in the binary at CSA bind locations.

**−−no-enable-sse3**

Disable use of SSE3 instructions if available.

**−−no-enable-sse4-1**

Disable use of SSE4.1 instructions if available.

**−−no-enable-sse4-2**

Disable use of SSE4.2 instructions if available.

**−−no-enable-ssse3**

Disable use of SSSE3 instructions if available.

**−−enable-sudiv**

**Type:** maybe_bool
**Default:** `unset`

Deprecated (use `--arm-arch` instead).

**−−enable-vfp3**

**Type:** maybe_bool
**Default:** `unset`

Deprecated (use `--arm-arch` instead).

**−−ephemeron-fixpoint-iterations**

**Type:** Int
**Default:** `10`

Number of fixpoint iterations it takes to switch to linear ephemeron algorithm.

**−−es-staging**

**Internal use only.**

Enable test-worthy Harmony features.

**−−no-experimental-inline-promise-constructor**

Don’t inline the `Promise` constructor in TurboFan.

**−−experimental-new-space-growth-heuristic**

Grow the new space based on the percentage of survivors instead of their absolute value.

**−−experimental-stack-trace-frames**

Enable experimental frames (API/Builtins) and stack trace layout.

**−−experimental-wasm-anyref**

Enable prototype anyref opcodes for WASM.

**−−experimental-wasm-bigint**

Enable prototype JS `BigInt` support for WASM.

**−−no-experimental-wasm-bulk-memory**

Disable prototype bulk memory opcodes for WASM.

**−−experimental-wasm-compilation-hints**

Enable prototype compilation hints section for WASM.

**−−experimental-wasm-eh**

Enable prototype exception handling opcodes for WASM.

**−−experimental-wasm-mv**

Enable prototype multi-value support for WASM.

**−−experimental-wasm-return-call**

Enable prototype return call opcodes for WASM.

**−−no-experimental-wasm-sat-f2i-conversions**

Disable prototype saturating float conversion opcodes for WASM.

**−−no-experimental-wasm-se**

Disable prototype sign extension opcodes for WASM.

**−−experimental-wasm-simd**

Enable prototype SIMD opcodes for WASM.

**−−experimental-wasm-threads**

Enable prototype thread opcodes for WASM.

**−−experimental-wasm-type-reflection**
Enable prototype WASM type reflection in JS for WASM.

**−−expose-async-hooks**
Expose `async_hooks` object.

**−−expose-cputracemark-as**
**Type:** String
**Default:** `NULL`

Expose `cputracemark` extension under the specified name.

**−−expose-externalize-string**
Expose `externalize` string extension.

**−−expose-free-buffer**
Expose `freeBuffer` extension.

**−−expose-gc**
Expose `gc` extension.

**−−expose-gc-as**
**Type:** String
**Default:** `NULL`

Expose `gc` extension under the specified name.

**−−expose-inspector-scripts**
Expose `injected-script-source.js` for debugging.

**−−expose-trigger-failure**
Expose trigger-failure extension.

**−−no-expose-wasm**
Don’t expose WASM interface to JavaScript.

**−−no-fast-math**
Don’t enable faster, potentially less accurate, math functions.

**−−fast-promotion-new-space**
Fast promote new space on high survival rates.

**−−feedback-normalization**
Feed back normalisation to constructors.

**−−no-flush-bytecode**
Don’t flush bytecode that hasn’t executed recently.

**−−force-long-branches**
Force all emitted branches to be in long mode (MIPS/PPC only).

**−−force-marking-deque-overflows**
Force overflows of marking deque by reducing its size to 64 words.

**−−force-slow-path**
Always take the slow path for builtins.

**−−frame-count**
**Type:** Int
**Default:** `1`

Number of stack frames inspected by the profiler.

**−−function-context-specialization**
Enable function context specialization in TurboFan.

**−−future**
Implies all staged features that we want to ship in the not-too-far future.

**−−fuzzer-gc-analysis**
Print number of allocations and enable analysis mode for GC fuzz-testing, e.g. `--stress-marking`, `--stress-scavenge`.

**−−fuzzer-random-seed**
**Type:** Int
**Default:** `0`

Default seed for initialising fuzzer random generator (0, the default, means to use v8's random number generator seed).

**−−gc-experiment-background-schedule**
Use new background GC schedule heuristics.

**−−gc-experiment-less-compaction**
Use less compaction in non-memory reducing mode.

**−−gc-fake-mmap**
**Type:** String
**Default:** `"/tmp/__v8_gc__"`

Specify the name of the file for fake gc mmap used in `ll_prof`.

**−−gc-freelist-strategy**
**Type:** Int
**Default:** `5`

Freelist strategy to use. Supported values and their meanings are:

| | |
|---|---|
| 0 | `FreeListLegacy` |
| 1 | `FreeListFastAlloc` |
| 2 | `FreeListMany` |
| 3 | `FreeListManyCached` |
| 4 | `FreeListManyCachedFastPath` |
| 5 | `FreeListManyCachedOrigin` |

**−−gc-global**

Always perform global GCs.

**−−gc-interval**

**Type:** Int<br>
**Default:** `-1`

Garbage collect after *N* allocations.

**−−gc-stats**

**Type:** Int<br>
**Default:** `0`

Used by tracing internally to enable gc statistics.

**−−gdbjit**

Enable GDBJIT interface.

**−−gdbjit-dump**

Dump ELF objects with debug info to disk.

**−−gdbjit-dump-filter**

Dump only objects containing this substring.

**−−gdbjit-full**

Enable GDBJIT interface for all code objects.

**−−no-global-gc-scheduling**

Disable GC scheduling based on global memory.

**−−no-hard-abort**

Don’t abort by crashing.

**−−harmony**

Enable all completed Harmony features.

**−−no-harmony-dynamic-import**

Disable dynamic import.

**−−no-harmony-import-meta**

Disable `import.meta` property.

**−−harmony-intl-add-calendar-numbering-system**
Add `calendar` and `numberingSystem` to `DateTimeFormat`.

**−−no-harmony-intl-bigint**
Disable `BigInt.prototype.toLocaleString`.

**−−no-harmony-intl-date-format-range**
Disable `DateTimeFormat.formatRange`.

**−−harmony-intl-dateformat-day-period**
Add `dayPeriod` option to `DateTimeFormat`.

**−−harmony-intl-dateformat-fractional-second-digits**
Add `fractionalSecondDigits` option to `DateTimeFormat`.

**−−harmony-intl-dateformat-quarter**
**In progress.**

Add `quarter` option to `DateTimeFormat`.

**−−no-harmony-intl-datetime-style**
Disable `dateStyle` and `timeStyle` for `DateTimeFormat`.

**−−no-harmony-intl-numberformat-unified**
Disable unified `Intl.NumberFormat` features.

**−−harmony-intl-segmenter**
Enable `Intl.Segmenter`.

**−−no-harmony-namespace-exports**
Disable namespace exports (`export * as foo from "bar"`).

**−−harmony-nullish**
**In progress.**

Enable nullish operator.

**−−harmony-optional-chaining**
**In progress.**

Enable optional chaining syntax.

**−−harmony-private-methods**
**In progress.**

Enable private methods in class literals.

**−−no-harmony-promise-all-settled**
Disable `Promise.allSettled`.

**−−harmony-regexp-sequence**
**In progress.**

Enable RegExp Unicode sequence properties.

**−−no-harmony-sharedarraybuffer**
Disable `SharedArrayBuffer`.

**−−no-harmony-shipping**
Disable all shipped Harmony features.

**−−harmony-weak-refs**
**In progress.**

Enable weak references.

**−−hash-seed**
**Type:** Uint64
**Default:** `0`

Fixed seed to use to hash property keys (0 means random). With snapshots this option cannot override the baked-in seed.

**−−heap-growing-percent**
**Type:** Int
**Default:** `0`

Specifies heap growing factor as ( 1 + heap_growing_percent / 100 ).

**−−heap-profiler-trace-objects**
Dump heap object allocations/movements/size_updates.

**−−no-heap-profiler-use-embedder-graph**
Don’t use the new EmbedderGraph API to get embedder nodes.

**−−heap-snapshot-string-limit**
**Type:** Int
**Default:** `1024`

Truncate strings to this length in the heap snapshot.

**−−histogram-interval**
**Type:** Int
**Default:** `600000`

Time interval in ms for aggregating memory histograms.

**−−huge-max-old-generation-size**

Increase maximum size of the old space to 4 GB for x64 systems with the physical memory bigger than 16 GB.

**−−no-icu-timezone-data**

Don’t get information about timezones from ICU.

**−−no-idle-time-scavenge**

Don’t perform scavenges in idle time.

**−−no-ignition-elide-noneffectful-bytecodes**

Don’t elide bytecodes which won’t have any external effect.

**−−no-ignition-filter-expression-positions**

Don’t filter expression positions before the bytecode pipeline.

**−−no-ignition-reo**

Don’t use ignition register equivalence optimiser.

**−−no-ignition-share-named-property-feedback**

Don’t share feedback slots when loading the same named property from the same object.

**−−ignore-asm-unimplemented-break**

Don’t break for `ASM_UNIMPLEMENTED_BREAK` macros.

**−−no-incremental-marking**

Don’t use incremental marking.

**−−no-incremental-marking-wrappers**

Don’t use incremental marking for marking wrappers.

**−−initial-heap-size**

**Type:** size_t
**Default:** `0`

Initial size of the heap (in MBytes).

**−−initial-old-space-size**

**Type:** size_t
**Default:** `0`

Initial old space size (in MBytes).

**−−no-inline-accessors**

Don’t inline JavaScript accessors.

**−−no-inline-new**

Don’t use fast inline allocation.

**−−interpreted-frames-native-stack**
Show interpreted frames on the native stack (useful for external profilers).

**−−interrupt-budget**
**Type:** Int
**Default:** `147456`

Interrupt budget which should be used for the profiler counter.

**−−jitless**
Disable runtime allocation of executable memory.

**−−no-lazy**
Don’t use lazy compilation.

**−−no-lazy-feedback-allocation**
Don’t allocate feedback vectors lazily.

**−−liftoff**
Enable Liftoff, the baseline compiler for WebAssembly.

**−−lite-mode**
Enable trade-off of performance for memory savings.

**−−ll-prof**
Enable low-level Linux profiler.

**−−log** Minimal logging (no API, code, GC, suspect, or handles samples).

**−−log-all**
Log all events to the log file.

**−−log-api**
Log API events to the log file.

**−−log-code**
Log code events to the log file without profiling.

**−−no-log-colour**
Don’t use coloured output when logging.

**−−log-function-events**
Log function events (parse, compile, execute) separately.

**−−log-handles**
Log global handle events.

**−−log-instruction-file**

**Type:** String
**Default:** `“arm64_inst.csv”`

AArch64 instruction statistics log file.

**−−log-instruction-period**

**Type:** Int
**Default:** `4194304`

AArch64 instruction statistics logging period.

**−−log-instruction-stats**

Log AArch64 instruction statistics.

**−−log-internal-timer-events**

Time internal events.

**−−log-source-code**

Log source code.

**−−log-suspect**

Log suspect operations.

**−−logfile**

**Type:** String
**Default:** `“v8.log”`

Specify the name of the log file.

**−−no-logfile-per-isolate**

Use a single log-file for each isolate.

**−−manual-evacuation-candidates-selection**

Test mode only flag. It allows a unit test to select evacuation candidates pages (requires `--stress-compaction`).

**−−map-counters**

Map counters to a file.

**−−max-heap-size**

**Type:** size_t
**Default:** `0`

Maximum size of the heap (in MBytes). Both `--max-semi-space-size` and `--max-old-space-size` take precedence. All three flags cannot be specified at the same time.

**−−max-inlined-bytecode-size**

**Type:** Int
**Default:** `500`

Maximum size of bytecode for a single inlining.

**−−max-inlined-bytecode-size-absolute**
**Type:** Int
**Default:** `5000`

Maximum cumulative size of bytecode considered for inlining.

**−−max-inlined-bytecode-size-cumulative**
**Type:** Int
**Default:** `1000`

Maximum cumulative size of bytecode considered for inlining.

**−−max-inlined-bytecode-size-small**
**Type:** Int
**Default:** `30`

Maximum size of bytecode considered for small function inlining.

**−−max-lazy**
Ignore eager compilation hints.

**−−max-old-space-size**
**Type:** size_t
**Default:** `0`

Maximum size of the old space (in MBytes).

**−−max-optimized-bytecode-size**
**Type:** Int
**Default:** `61440`

Maximum bytecode size to be considered for optimisation; too high values may cause the compiler to hit (release) assertions.

**−−max-polymorphic-map-count**
**Type:** Int
**Default:** `4`

Maximum number of maps to track in `POLYMORPHIC` state.

**−−max-semi-space-size**
**Type:** size_t
**Default:** `0`

Maximum size of a semi-space (in MBytes), the new space consists of two semi-spaces.

**−−max-stack-trace-source-length**
**Type:** Int
**Default:** `300`

Maximum length of function source code printed in a stack trace.

**−−mcpu**
**Type:** String
**Default:** “`auto`”

Enable optimisation for a specific CPU.

**−−no-memory-reducer**
Don’t use memory reducer.

**−−no-memory-reducer-for-small-heaps**
Don’t use memory reducer for small heaps.

**−−min-inlining-frequency**
**Type:** Float
**Default:** `0.15`

Minimum frequency for inlining.

**−−min-semi-space-size**
**Type:** size_t
**Default:** `0`

Minimum size of a semi-space (in MBytes), the new space consists of two semi-spaces.

**−−minor-mc**
Perform young generation mark compact GCs.

**−−no-minor-mc-parallel-marking**
Don’t use parallel marking for the young generation.

**−−mock-arraybuffer-allocator**
Use a mock `ArrayBuffer` allocator for testing.

**−−mock-arraybuffer-allocator-limit**
**Type:** size_t
**Default:** `0`

Memory limit for mock `ArrayBuffer` allocator used to simulate OOM for testing.

**−−no-modify-field-representation-inplace**
Disable in-place field representation updates.

**−−no-move-object-start**
Disable moving of object starts.

**−−native-code-counters**
Generate extra code for manipulating stats counters.

**−−never-compact**

**Testing only.**

Never perform compaction on full GC.

**−−no-opt**

Don’t use adaptive optimisations.

**−−optimize-for-size**

Enable optimisations which favour memory size over execution speed.

**−−no-page-promotion**

Don’t promote pages based on utilisation.

**−−page-promotion-threshold**

**Type:** Int
**Default:** `70`

Minimum percentage of live bytes on a page to enable fast evacuation.

**−−no-parallel-compaction**

Don’t use parallel compaction.

**−−parallel-compile-tasks**

Enable parallel compile tasks.

**−−no-parallel-marking**

Don’t use parallel marking in atomic pause.

**−−no-parallel-pointer-update**

Don’t use parallel pointer update during compaction.

**−−no-parallel-scavenge**

Disable parallel scavenging.

**−−parse-only**

Only parse the sources.

**−−no-partial-constant-pool**

Disable use of partial constant pools (X64 only).

**−−perf-basic-prof**

Enable perf Linux profiler (basic support).

**−−perf-basic-prof-only-functions**

Only report function code ranges to perf (i.e. no stubs).

**−−perf-prof**
Enable perf Linux profiler (experimental annotate support).

**−−perf-prof-annotate-wasm**
Load WASM source-map and provide annotate support when used with `--perf-prof` (experimental).

**−−perf-prof-unwinding-info**
Enable unwinding info for perf Linux profiler (experimental).

**−−no-polymorphic-inlining**
Disable polymorphic inlining.

**−−predictable**
Enable predictable mode.

**−−predictable-gc-schedule**
Predictable garbage collection schedule. Fixes heap growing, idle, and memory reducing behavior.

**−−prepare-always-opt**
Prepare for turning on always opt.

**−−print-all-exceptions**
Print exception object and stack trace on each thrown exception.

**−−print-bytecode**
Print bytecode generated by ignition interpreter.

**−−print-bytecode-filter**
**Type:** String
**Default:** `"*"`

Filter for selecting which functions to print bytecode.

**−−print-deopt-stress**
Print number of possible deopt points.

**−−print-opt-source**
Print source code of optimised and inlined functions.

**−−print-wasm-code**
Print WebAssembly code.

**−−print-wasm-stub-code**
Print WebAssembly stub code.

**−−prof**
Log statistical profiling information (implies `--log-code`).

**−−no-prof-browser-mode**
Turn off browser-compatible mode when profiling with `--prof`.

**−−prof-cpp**
Like `--prof`, but ignore generated code.

**−−prof-sampling-interval**
**Type:** Int
**Default:** `1000`

Interval for `--prof` samples (in microseconds).

**−−profile-deserialization**
Print the time it takes to deserialise the snapshot.

**−−random-gc-interval**
**Type:** Int
**Default:** `0`

Collect garbage after **`random`**`(0,` *X*) allocations. It overrides `--gc-interval`.

**−−random-seed**
**Type:** Int
**Default:** `0`

Default seed for initialising random generator (0, the default, means to use system random).

**−−no-randomize-hashes**
Don't randomise hashes to avoid predictable hash collisions (with snapshots this option cannot override the baked-in seed).

**−−redirect-code-traces**
Output deopt information and disassembly into file "code-*pid*-*isolate-id*.asm."

**−−redirect-code-traces-to**
**Type:** String
**Default:** `NULL`

Output deopt information and disassembly into the given file.

**−−regexp-interpret-all**
Interpret all regexp code.

**−−regexp-mode-modifiers**
Enable inline flags in regexp.

**−−no-regexp-optimization**
Don’t generate optimised regexp code.

**−−regexp-tier-up**
Enable regexp interpreter and tier up to the compiler.

**−−no-rehash-snapshot**
Don’t rehash strings from the snapshot to override the baked-in seed.

**−−reserve-inline-budget-scale-factor**
**Type:** Float
**Default:** `1.2`

Maximum cumulative size of bytecode considered for inlining.

**−−retain-maps-for-n-gc**
**Type:** Int
**Default:** `2`

Keeps maps alive for *N* old space garbage collections.

**−−runtime-call-stats**
Report runtime call counts and times.

**−−sampling-heap-profiler-suppress-randomness**
Use constant sample intervals to eliminate test flakiness.

**−−no-script-streaming**
Disable parsing on background.

**−−semi-space-growth-factor**
**Type:** Int
**Default:** `2`

Factor by which to grow the new space.

**−−serialization-chunk-size**
**Type:** Uint
**Default:** `4096`

Custom size for serialization chunks.

**−−serialization-statistics**
Collect statistics on serialized objects.

**−−sim-stack-alignment**
**Type:** Int
**Default:** `8`

Stack alignment in bytes in simulator (4 or 8, 8 is default).

**−−sim-stack-size**

**Type:** Int
**Default:** `2048`

Stack size of the ARM64, MIPS64 and PPC64 simulator in kBytes (default is 2 MB).

**−−single-threaded**

Disable the use of background tasks.

**−−stack-size**

**Type:** Int
**Default:** `984`

Default size of stack region v8 is allowed to use (in kBytes).

**−−stack-trace-limit**

**Type:** Int
**Default:** `10`

Number of stack frames to capture.

**−−stack-trace-on-illegal**

Print stack trace when an illegal exception is thrown.

**−−startup-blob**

**Type:** String
**Default:** `NULL`

Write V8 startup blob file. (`mksnapshot` only).

**−−startup-src**

**Type:** String
**Default:** `NULL`

Write V8 startup as C++ src. (`mksnapshot` only).

**−−stop-sim-at**

**Type:** Int
**Default:** `0`

Simulator stop after *X* number of instructions.

**−−stress-compaction**

Stress the GC compactor to flush out bugs (implies `--force-marking-deque-overflows`).

**−−stress-compaction-random**

Stress GC compaction by selecting random percent of pages as evacuation candidates. It overrides `--stress-compaction`.

**−−stress-flush-bytecode**
Stress bytecode flushing.

**−−stress-gc-during-compilation**
Simulate GC/compiler thread race related to https://crbug.com/v8/8520.

**−−stress-incremental-marking**
Force incremental marking for small heaps and run it more often.

**−−stress-inline**
Set high thresholds for inlining to inline as much as possible.

**−−stress-lazy-source-positions**
Collect lazy source positions immediately after lazy compilation.

**−−stress-marking**
**Type:** Int
**Default:** `0`

Force marking at random points between 0 and *X* (inclusive) percent of the regular marking start limit.

**−−stress-runs**
**Type:** Int
**Default:** `0`

Number of stress runs.

**−−stress-sampling-allocation-profiler**
**Type:** Int
**Default:** `0`

Enable sampling allocation profiler with *X* as a sample interval.

**−−stress-scavenge**
**Type:** Int
**Default:** `0`

Force scavenge at random points between 0 and *X* (inclusive) percent of the new space capacity.

**−−stress-validate-asm**
Try to validate everything as asm.js.

**−−stress-wasm-code-gc**
Stress-test garbage collection of WASM code.

**−−suppress-asm-messages**
Don’t emit asm.js related messages (for golden file testing).

**−−target-arch**

**Type:** String
**Default:** `NULL`

The `mksnapshot` target architecture. (`mksnapshot` only).

**−−target-os**

**Type:** String
**Default:** `NULL`

The `mksnapshot` target OS. (`mksnapshot` only).

**−−test-small-max-function-context-stub-size**

Enable testing the function context size overflow path by making the maximum size smaller.

**−−testing-float-flag**

**Type:** Float
**Default:** `2.5`

Float-flag.

**−−testing-int-flag**

**Type:** Int
**Default:** `13`

Testing_int_flag.

**−−testing-maybe-bool-flag**

**Type:** maybe_bool
**Default:** `unset`

Testing_maybe_bool_flag.

**−−testing-prng-seed**

**Type:** Int
**Default:** `42`

Seed used for threading test randomness.

**−−testing-string-flag**

**Type:** String
**Default:** `“Hello, world!”`

String-flag.

**−−no-thin-strings**

Disable ThinString support.

**−−trace**

Trace function calls.

**−−trace-all-uses**
Trace all use positions.

**−−trace-allocation-stack-interval**
**Type:** Int
**Default:** `-1`

Print stack trace after *N* free-list allocations.

**−−trace-allocations-origins**
Show statistics about the origins of allocations. Combine with `--no-inline-new` to track allocations from generated code.

**−−trace-asm-parser**
Verbose logging of asm.js parse failures.

**−−trace-asm-scanner**
Log tokens encountered by asm.js scanner.

**−−trace-asm-time**
Log asm.js timing info to the console.

**−−trace-block-coverage**
Trace collected block coverage information.

**−−trace-compiler-dispatcher**
Trace compiler dispatcher activity.

**−−trace-concurrent-marking**
Trace concurrent marking.

**−−trace-concurrent-recompilation**
Track concurrent recompilation.

**−−trace-deopt**
Trace optimise function deoptimisation.

**−−trace-detached-contexts**
Trace native contexts that are expected to be garbage collected.

**−−trace-duplicate-threshold-kb**
**Type:** Int
**Default:** `0`

Print duplicate objects in the heap if their size is more than given threshold.

**−−trace-environment-liveness**
Trace liveness of local variable slots.

**−−trace-evacuation**
Report evacuation statistics.

**−−trace-evacuation-candidates**
Show statistics about the pages evacuation by the compaction.

**−−trace-file-names**
Include file names in trace-opt/trace-deopt output.

**−−trace-for-in-enumerate**
Trace for-in enumerate slow-paths.

**−−trace-fragmentation**
Report fragmentation for old space.

**−−trace-fragmentation-verbose**
Report fragmentation for old space (detailed).

**−−trace-gc**
Print one trace line following each garbage collection.

**−−trace-gc-freelists**
Print details of each freelist before and after each major garbage collection.

**−−trace-gc-freelists-verbose**
Print details of freelists of each page before and after each major garbage collection.

**−−trace-gc-ignore-scavenger**
Do not print trace line after scavenger collection.

**−−trace-gc-nvp**
Print one detailed trace line in name=value format after each garbage collection.

**−−trace-gc-object-stats**
Trace object counts and memory usage.

**−−trace-gc-verbose**
Print more details following each garbage collection.

**−−trace-generalization**
Trace map generalisation.

**−−trace-heap-broker**
Trace the heap broker (reports on missing data only).

**−−trace-heap-broker-verbose**
Trace the heap broker verbosely (all reports).

**−−trace-ic**
Trace inline cache state transitions for tools/ic-processor.

**−−trace-idle-notification**
Print one trace line following each idle notification.

**−−trace-idle-notification-verbose**
Print the heap state used by the idle notification.

**−−trace-ignition-codegen**
Trace the codegen of ignition interpreter bytecode handlers.

**−−trace-ignition-dispatches**
Traces the dispatches to bytecode handlers by the ignition interpreter.

**−−trace-ignition-dispatches-output-file**
**Type:** String
**Default:** `NULL`

The file to which the bytecode handler dispatch table is written (by default, the table is not written to a file).

**−−trace-incremental-marking**
Trace progress of the incremental marking.

**−−trace-maps**
Trace map creation.

**−−no-trace-maps-details**
Don't log map details.

**−−trace-migration**
Trace object migration.

**−−trace-minor-mc-parallel-marking**
Trace parallel marking for the young generation.

**−−trace-mutator-utilization**
Print mutator utilisation, allocation speed, GC speed.

**−−trace-opt**
Trace lazy optimisation.

**−−trace-opt-stats**
Trace lazy optimisation statistics.

**−−trace-opt-verbose**
Extra verbose compilation tracing.

**−−trace-osr**
Trace on-stack replacement.

**−−trace-parallel-scavenge**
Trace parallel scavenge.

**−−trace-pretenuring**
Trace pretenuring decisions of HAllocate instructions.

**−−trace-pretenuring-statistics**
Trace allocation site pretenuring statistics.

**−−trace-protector-invalidation**
Trace protector cell invalidations.

**−−trace-prototype-users**
Trace updates to prototype user tracking.

**−−trace-rail**
Trace RAIL mode.

**−−trace-representation**
Trace representation types.

**−−trace-serializer**
Print code serializer trace.

**−−trace-side-effect-free-debug-evaluate**
Print debug messages for side-effect-free debug-evaluate for testing.

**−−trace-sim**
Trace simulator execution.

**−−trace-sim-messages**
Trace simulator debug messages. Implied by `--trace-sim`.

**−−trace-store-elimination**
Trace store elimination.

**−−trace-stress-marking**
Trace stress marking progress.

**−−trace-stress-scavenge**
Trace stress scavenge progress.

**−−trace-track-allocation-sites**
Trace the tracking of allocation sites.

**−−trace-turbo**
Trace generated TurboFan IR.

**−−trace-turbo-alloc**
Trace TurboFan's register allocator.

**−−trace-turbo-ceq**
Trace TurboFan's control equivalence.

**−−trace-turbo-cfg-file**
**Type:** String
**Default:** `NULL`

Trace turbo cfg graph (for C1 visualiser) to a given file name.

**−−trace-turbo-filter**
**Type:** String
**Default:** `"*"`

Filter for tracing turbofan compilation.

**−−trace-turbo-graph**
Trace generated TurboFan graphs.

**−−trace-turbo-inlining**
Trace TurboFan inlining.

**−−trace-turbo-jt**
Trace TurboFan's jump threading.

**−−trace-turbo-load-elimination**
Trace TurboFan load elimination.

**−−trace-turbo-loop**
Trace TurboFan's loop optimisations.

**−−trace-turbo-path**
**Type:** String
**Default:** `NULL`

Directory to dump generated TurboFan IR to.

**−−trace-turbo-reduction**
Trace TurboFan's various reducers.

**−−trace-turbo-scheduled**
Trace TurboFan IR with schedule.

**−−trace-turbo-scheduler**
Trace TurboFan’s scheduler.

**−−trace-turbo-trimming**
Trace TurboFan’s graph trimmer.

**−−no-trace-turbo-types**
Don’t trace TurboFan’s types.

**−−trace-unmapper**
Trace the unmapping.

**−−trace-verify-csa**
Trace code stubs verification.

**−−trace-wasm-ast-end**
**Type:** Int
**Default:** `0`

End function for WASM AST trace (exclusive).

**−−trace-wasm-ast-start**
**Type:** Int
**Default:** `0`

Start function for WASM AST trace (inclusive).

**−−trace-wasm-code-gc**
Trace garbage collection of WASM code.

**−−trace-wasm-memory**
Print all memory updates performed in WASM code.

**−−trace-wasm-serialization**
Trace serialization/deserialization.

**−−trace-zone-stats**
Trace zone memory usage.

**−−no-track-computed-fields**
Don’t track computed boilerplate fields.

**−−no-track-detached-contexts**
Don’t track native contexts that are expected to be garbage collected.

**−−no-track-double-fields**
Don’t track fields with double values.

**−−no-track-field-types**
Don’t track field types.

**−−no-track-fields**
Don’t track fields with only SMI values.

**−−track-gc-object-stats**
Track object counts and memory usage.

**−−no-track-heap-object-fields**
Don’t track fields with heap values.

**−−track-retaining-path**
Enable support for tracking retaining path.

**−−no-turbo-allocation-folding**
Disable TurboFan allocation folding.

**−−no-turbo-cf-optimization**
Don’t optimise control flow in TurboFan.

**−−turbo-control-flow-aware-allocation**
Consider control flow while allocating registers.

**−−no-turbo-escape**
Disable escape analysis.

**−−turbo-filter**
**Type:** String
**Default:** “*”

Optimisation filter for TurboFan compiler.

**−−no-turbo-inline-array-builtins**
Don’t inline array builtins in TurboFan code.

**−−no-turbo-inlining**
Disable inlining in TurboFan.

**−−turbo-instruction-scheduling**
Enable instruction scheduling in TurboFan.

**−−no-turbo-jt**
Disable jump threading in TurboFan.

**−−no-turbo-load-elimination**
Disable load elimination in TurboFan.

**−−no-turbo-loop-peeling**
Disable TurboFan loop peeling.

**−−no-turbo-loop-rotation**
Disable TurboFan loop rotation.

**−−no-turbo-loop-variable**
Disable TurboFan loop variable optimisation.

**−−no-turbo-move-optimization**
Don’t optimise gap moves in TurboFan.

**−−turbo-profiling**
Enable profiling in TurboFan.

**−−no-turbo-rewrite-far-jumps**
Don’t rewrite far to near jumps (ia32,x64).

**−−turbo-sp-frame-access**
Use stack pointer-relative access to frame wherever possible.

**−−no-turbo-splitting**
Don’t split nodes during scheduling in TurboFan.

**−−turbo-stats**
Print TurboFan statistics.

**−−turbo-stats-nvp**
Print TurboFan statistics in machine-readable format.

**−−turbo-stats-wasm**
Print TurboFan statistics of WASM compilations.

**−−no-turbo-store-elimination**
Disable store-store elimination in TurboFan.

**−−turbo-stress-instruction-scheduling**
Randomly schedule instructions to stress dependency tracking.

**−−turbo-verify**
Verify TurboFan graphs at each phase.

**−−turbo-verify-allocation**
Verify register allocation in TurboFan.

**−−turbo-verify-machine-graph**
**Type:** String
**Default:** `NULL`

Verify TurboFan machine graph before instruction selection.

**−−no-unbox-double-arrays**
Don’t automatically unbox arrays of doubles.

**−−untrusted-code-mitigations**
Enable mitigations for executing untrusted code.

**−−use-external-strings**
Use external strings for source code.

**−−no-use-ic**
Don’t use inline caching.

**−−no-use-idle-notification**
Don’t use idle notification to reduce memory footprint.

**−−no-use-marking-progress-bar**
Don’t use a progress bar to scan large objects in increments when incremental marking is active.

**−−no-use-osr**
Don’t use on-stack replacement.

**−−use-strict**
Enforce strict mode.

**−−no-use-verbose-printer**
Disable verbose printing.

**−−v8-os-page-size**
**Type:** Int
**Default:** `0`

Override OS page size (in KBytes).

**−−no-validate-asm**
Don’t validate asm.js modules before compiling.

**−−no-wasm-async-compilation**
Disable actual asynchronous compilation for `WebAssembly.compile`.

**−−no-wasm-code-gc**
Disable garbage collection of WASM code.

**−−wasm-disable-structured-cloning**
Disable WASM structured cloning.

**−−wasm-fuzzer-gen-test**
Generate a test case when running a WASM fuzzer.

**−−wasm-grow-shared-memory**
Allow growing shared WebAssembly memory objects.

**−−wasm-interpret-all**
Execute all WASM code in the WASM interpreter.

**−−wasm-lazy-compilation**
Enable lazy compilation for all WASM modules.

**−−wasm-lazy-validation**
Enable lazy validation for lazily-compiled WASM functions.

**−−no-wasm-math-intrinsics**
Don’t intrinsify some Math imports into WASM.

**−−wasm-max-code-space**
**Type:** Uint
**Default:** `1024`

Maximum committed code space for WASM (in MB).

**−−wasm-max-mem-pages**
**Type:** Uint
**Default:** `32767`

Maximum number of 64KiB memory pages of a WASM instance.

**−−wasm-max-table-size**
**Type:** Uint
**Default:** `10000000`

Maximum table size of a WASM instance.

**−−wasm-no-bounds-checks**
Disable bounds checks (performance testing only).

**−−wasm-no-stack-checks**
Disable stack checks (performance testing only).

**−−wasm-num-compilation-tasks**
**Type:** Int
**Default:** `10`

Number of parallel compilation tasks for WASM.

**−−wasm-opt**
Enable WASM optimisation.

**−−no-wasm-shared-code**
Don’t share code underlying a WASM module when it is transferred.

**−−no-wasm-shared-engine**
Don’t share one WASM engine between all isolates within a process.

**−−wasm-staging**
Enable staged WASM features.

**−−wasm-test-streaming**
Use streaming compilation instead of async compilation for tests.

**−−wasm-tier-mask-for-testing**
**Type:** Int
**Default:** `0`

Bitmask of functions to compile with TurboFan instead of Liftoff.

**−−no-wasm-tier-up**
Disable WASM baseline compilation and tier up to the optimising compiler.

**−−no-wasm-trap-handler**
Don’t use signal handlers to catch out of bounds memory access in WASM (currently Linux x86_64 only).

**−−wasm-write-protect-code-memory**
Write protect code memory on the WASM native heap.

**−−no-win64-unwinding-info**
Disable unwinding info for Windows/x64.

**−−no-write-protect-code-memory**
Don’t write protect code memory.

**−−no-young-generation-large-objects**
Don’t allocate large objects by default in the young generation large object space.

## SEE ALSO

**node**(1)

## AUTHORS