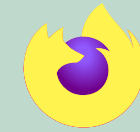




WEBASSEMBLY

[Overview](#)[Getting Started](#)[Specs](#)[Feature Extensions](#)[Community](#)[FAQ](#)

WebAssembly 1.0 has shipped in 4 major browser engines.

[Learn more](#)

DOCUMENTATION

[FAQ](#)[WebAssembly High-Level Goals](#)[Use Cases](#)[Portability](#)[Security](#)[Web Embedding](#)[Non-Web Embeddings](#)[Tooling support](#)

WebAssembly High-Level Goals

1. Define a [portable](#), size- and load-time-efficient [binary format](#) to serve as a compilation target which can be compiled to execute at native speed by taking advantage of common hardware capabilities available on a wide range of platforms, including [mobile](#) and [IoT](#).
2. Specify and implement incrementally:
 - a [Minimum Viable Product \(MVP\)](#) for the standard with roughly the same functionality as [asm.js](#), primarily aimed at [C/C++](#);
 - [additional features](#) 🦄, initially focused on key features like [threads](#), [zero cost exceptions](#), and [SIMD](#), followed by additional features prioritized by feedback and experience, including support for languages other than C/C++.
3. Design to execute within and integrate well with the *existing* [Web platform](#):
 - maintain the versionless, [feature-tested](#) and backwards-compatible evolution story of the Web;
 - execute in the same semantic universe as JavaScript;
 - allow synchronous calls to and from JavaScript;
 - enforce the same-origin and permissions security policies;
 - access browser functionality through the same Web APIs that are accessible to JavaScript; and
 - define a human-editable text format that is convertible to and from the binary format, supporting View Source functionality.
4. Design to support [non-browser embeddings](#) as well.
5. Make a great platform:
 - build a new LLVM backend for WebAssembly and an accompanying clang port ([why LLVM first?](#));
 - promote other compilers and tools targeting WebAssembly; and
 - enable other useful [tooling](#).