Profiles

Andreas Rossberg

Wasm CG 2023/10/11

Profiles

Precise specifications of sublanguages

Restrictions on syntax or semantics

Goal: minimising(!) fragmentation in the face of eco-system diversity and platform constraints

Chosen by eco-systems, not applications!

Profiles

Two separable aspects to the discussion:

1. Framework for specifying profiles

   …entirely editorial

2. Concrete definitions of profiles

   …separate proposals

Goals

Well-specified subsets

Compatibility across similar ecosystems

Stable and durable choices

Few and coarse

Non-Goals

Producer-side choice

Versioning

Feature detection

Alternate semantics

Intended Properties

Profiles need to be mutually compatible and composable

Producers should never assume *absence* of features

…targeted consumer may extend their profile over time

Deploy-time choice, avoid runtime conditionals on profiles

reftypes											
SIMD											
threads											
determinism											

	GraalWasm	Wabt	WAMR	Wasm3	WasmEdge	Wasmer	Wasmi	Wasmtime	WAVM	wazero	Wizard
reftypes											
SIMD											
threads											
determinism											

	GraalWasm	Wabt	WAMR	Wasm3	WasmEdge	Wasmer	Wasmi	Wasmtime	WAVM	wazero	Wizard
reftypes	✅	✅	✅	❌	✅	✅	✅	✅	✅	✅	✅
SIMD	❌	✅	✅	❌	✅	✅	❌❌	✅	✅	✅	✅
threads	✅	✅	✅	❌	✅	✅	❌	✅	✅	❌	✅
determinism	❌	❌	❌	❌	❌	❌	✅?	✅	❌	❌	❌

	GraalWasm	Wabt	WAMR	Wasm3	WasmEdge	Wasmer	Wasmi	Wasmtime	WAVM	wazero	Wizard
reftypes	⏯️	⏯️	✅	❌	⏯️	⏯️	✅	⏯️	⏯️	⏯️	✅
SIMD	❌	⏯️	✅	❌	⏯️	✅	❌❌	⏯️	⏯️	⏯️	✅
threads	⏯️	⏯️	✅	❌	⏯️	⏯️	❌	⏯️	⏯️	❌	⏯️
determinism	❌	❌	❌	❌	❌	❌	✅?	⏯️⏯️	❌	❌	❌

Process?

New *feature* proposals need to define interaction with existing *profiles*

…should be mentioned in process doc

New profiles are essentially feature proposals

…though some points from the process are N/A

All comes down to the usual proposal ordering problem

Outtakes

Risks

Profile inflation

False assumptions

Language design shortcutting and other abuses

Candidates

Non-determinism

Threads

SIMD

GC

Stack switching

Discussion

Candidates

…non-determinism, GC, SIMD, threads, stacks

Process

…as part of regular proposal evaluation?

…separately, retroactively? (breaking change?)

…case-by-case mixture?

Risk of Retroactivity

Technically, a breaking change

…future version of language allows feature removal

Possibly too late for respective customers

…need to make up a subset themselves

…incompatible choices may already have been made

Unclear if customers are willing to go through separate proposal process for this

…too much hassle for too little benefit at that point

…takes too long

…consequence may be maximal fragmentation and semantic proliferation after all

Risk of Proactivity

Hard to predict requirements

…may introduce unneeded profiles

…may specify the wrong thing

Though for concrete proposals, it may be easy to predict

Approach

For each profile X,

annotate syntactic and semantic rules that belong to X

define Wasm\X = Wasm minus all rules marked X

Intersections are automatically well-defined

There exists a full profile containing all features

memtype ::= *limits share*

share ::= unshared

| shared (T)

instr ::= …

| struct.new (G)

| struct.get_*sx*? (G)

| struct.set (G)

| …

(T = threads G = GC)