From References to GC

Proposal updates

Andreas Rossberg

Dfinity

Dependencies

Reference Types

Typed (Function) References

Reference Equality

Type Imports

GC MVP

GC Post-MVP

Dependencies

Reference Types

Typed (Function) References

Reference Equality

Type Imports

GC MVP

GC Post-MVP

Dependencies

Reference Types

Typed (Function) References

Type Imports

Reference Equality

GC MVP

GC Post-MVP

Reference Types

Introduces basic notion of reference types

Only funcref and externref

Multiple tables

Reference Types: Status

Phase 3

Ready for Phase 4,
pending 2nd engine to adapt to latest changes

No more open questions

Dependencies

Reference Types

Typed (Function) References

Type Imports

Reference Equality

GC MVP

GC Post-MVP

Dependencies

Reference Types

Typed (Function) References

Type Imports

Reference Equality

GC MVP

GC Post-MVP

Typed References

Allows references to be typed: (ref null? $t)

Currently only function types exist

Instructions for checking against null,
and to bind locals with non-nullable type

call_ref, closures (func.bind)

Subtyping: (ref $t) <: (ref null $t) <: funcref

Typed References: Status

Phase 2, ready for phase 3

Ready to be implemented in engines

Open questions:

1. Tables of non-nullable element type
2. Amount of type annotation on ref instructions
3. JS API?

Dependencies

Reference Types

Typed (Function) References

Reference Equality

Type Imports

GC MVP

GC Post-MVP

Dependencies

Reference Types

Typed (Function) References

Reference Equality

Type Imports

GC MVP

GC Post-MVP

Type Imports

Allows type definitions (as for reference types) to be imported and exported

Imports can specify upper type bound to constrain allowable types (e.g. funcref)

New requirement: include a mechanism to define abstract types

Type Imports: Status

Phase 1, next major work item

Proposal sketch exists

Open questions:

1. Reconcile with lack of anyref
2. Type abstraction mechanism
3. JS interop, especially for abstract types

Dependencies

Reference Types

Typed (Function) References

Reference Equality

Type Imports

GC MVP

GC Post-MVP

Dependencies

Reference Types

Typed (Function) References

Type Imports

Reference Equality

GC MVP

GC Post-MVP

Reference Equality

New ref.eq instruction

New eqref type of comparable references (e.g., funcref <: eqref)

Reference Equality: Status

No proposal yet
(originally was part of reference types)

Open questions:

1. Eq attribute in ref type or in type definition?
   (choose at allocation or definition time)
2. If there is anyref then eqref would be natural

Dependencies

Reference Types

Typed (Function) References

Reference Equality

Type Imports

GC MVP

GC Post-MVP

Dependencies

Reference Types

Typed (Function) References

Type Imports

GC MVP

GC Post-MVP

Reference Equality

GC MVP

Almost bare minimum of features

structs, arrays, scalars, casts

Simplicity first, avoid language bias

But remain forward-compatible to extensions

GC MVP: Status

Still Phase 1, adapted to upstream changes

Open questions (not exclusively):

1. Details of run time types
2. Nominal types, esp nominal subtyping
3. Function subtyping?
4. Other features to include?

outtakes