Custom Annotations for the text format

Proposal status update

Andreas Rossberg

Motivation

Binary format has custom sections

Want equivalent in text format

E.g. for hinting, debugging, custom use, etc.

…use cases keep popping up

Proposal Summary

Free-form format just like custom sections

Allow (@id …) to be placed anywhere in text

No prescribed or implied syntax or semantics,
neither regarding form nor placement

Can be ignored by tools,
but some may assign meaning to some ids

Only affects text format!

Details

Allow annotation brackets anywhere in text format

    annot ::= "(@"id annotelem* ")"

Headed by id that categorises extension

…by convention, corresponds to name of a custom section

Body is arbitrary token sequence, but must be well-bracketed

A tool may define more concrete structure for its annotations

Example: Names

```
(global (@name "äquidistanz") f64)

(func $sum (@name "∑")
    (local (@name "δ") i32)
    …
)
```

Example: Generic custom section

```
(module
  …
  (@custom "my-section" (after func)
    "contents-bytes"
  )
)
```

NB: placement “after func” only legal if a (non-empty) function section exists

Spec Appendix

Describes @name and @custom annotations

Analogous to “name” sections in binary format

Proposal Status

✅ Full spec (prose and formal)

✅ Interpreter implementation

✅ Test suite
(though no tests for Appendix)

✅ Implementation in wabt
(text-format only, does not affect engines)

Stage 3 (2020/03)

Discussion: Test suite

Custom sections and annotations are optional

…no implementation needs to recognise them

Thus, cannot put them into the regular test suite

Need infrastructure for optional tests

…fairly general problem, maybe solve separately?

Discussion

Move to stage 4?

Previous Discussion

Asked for lossless *generic* text-binary-text round-tripping and transformational tools that are generic over annotations

Asked for equally fine-grained annotations in binary format

But by design, custom sections and annotations are both uninterpreted; generic transformation cannot possibly work

No concrete suggestions have been made

Binary-text-binary round-tripping is the relevant direction, and (only) enabled by this proposal (@custom annotation)

Previous Discussion

Desire for lossless *generic* text-binary-text round-tripping with annotations

… but that direction of roundtrip never was lossless

… binary-text-binary roundtrip is the relevant direction,
and (only) enabled by this proposal (generic @custom annotation)

… custom binary sections and custom text annotations are both uninterpreted by design;
unclear how a generic bijection could possibly work in that setting

Concern about custom sections having coarser granularity

… e.g., no built-in way to annotate individual instructions

… but can be encoded in custom sections

… ultimately that's a complaint about the *binary* format

No concrete suggestions have been made