Annotations for the Wasm text format

Proposal update

Andreas Rossberg

Dfinity

Motivation

Binary format has custom sections

Want equivalent in text format

E.g. for interface types, debugging, etc.

Proposal Summary

Similar free-form format as 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

```
annotelem ::=
    keyword | reserved | uN | sN | fN | string | "$"id |
    "(" annotelem* ")" |
    "(@"id annotelem* ")"
```

A tool may define more concrete structure for its annotations

Example: Interface types

```
(module
  (memory (export "mem") 1)
  (func (import "" "log_") (param i32 i32))
  ...
  (@interface func $log (import "" "log") (param $arg string))
  (@interface implement (import "" "log_") (param $ptr i32) (param $len i32)
    arg.get $ptr
    arg.get $len
    memory-to-string "mem"
    call-import $log
  )
)
```

Example: Represent generic custom section

```
(module
  ...
  (@custom "my-section" (after function)
    "contents-bytes"
  )
)
```

Appendix

Describes @name and @custom annotations

Analogous to “name” sections

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)

Proposal Status

prose spec : ✓

formal spec : ✓

interpreter : ✓

tests : ✓

Stage 0

Note: text-format only, does not affect engines

Poll

Move to stage 2?