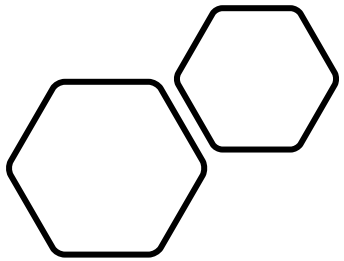
Why to Remove Subtyping



Ross Tate

1. No Established Use Cases

2. Counter to Primary Purpose

External References

anyref

Primary Purpose

- to avoid making assumptions about what a specific reference type in situations where you can't or don't need to know
- i.e. external references

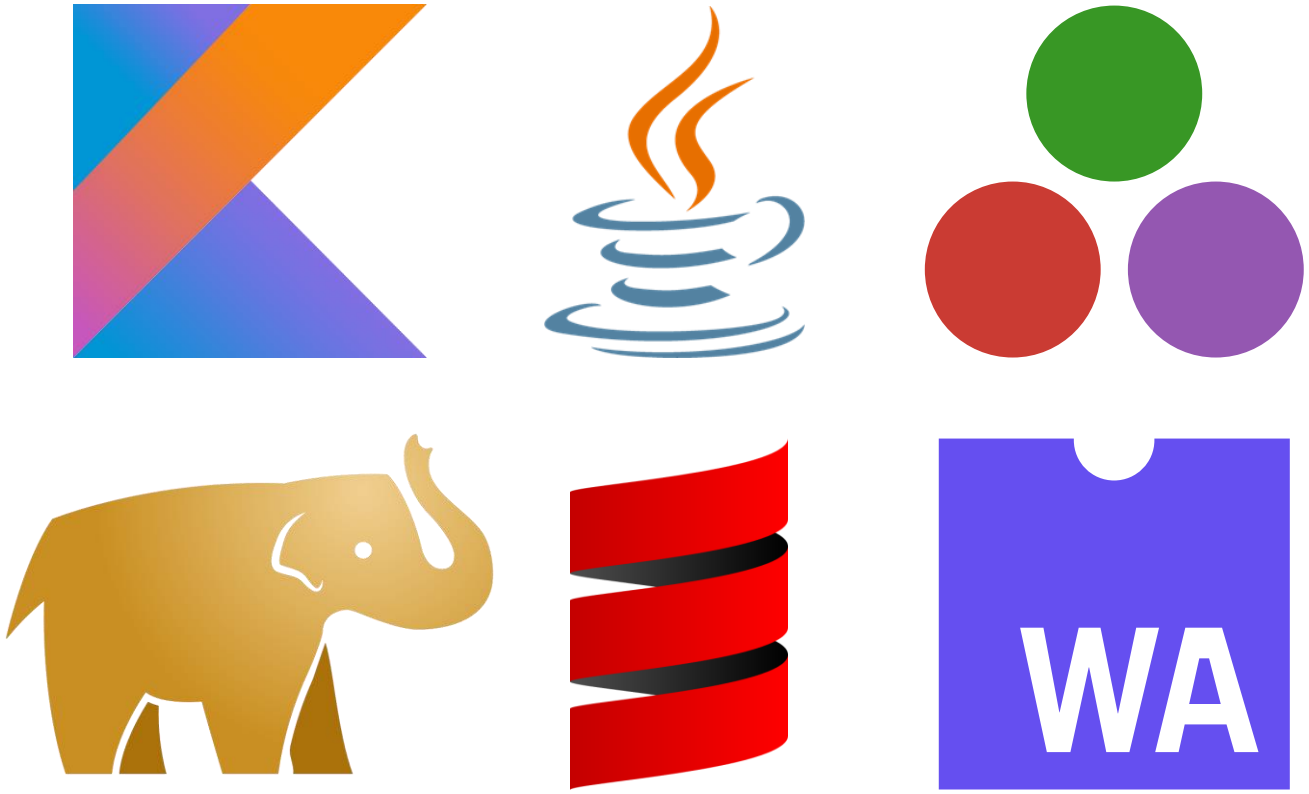
Subtyping

- Makes assumptions about the specific reference type
- WASI cannot assume all anyrefs are capabilities (even if those are the only external references)
- Browsers cannot assume all anyrefs are JS/DOM values (even if they are the only external refs)

3. Subtyping is notorious

It complicates *everything* if not done right

Subtyping in Industry



4. Imposes Casting Burden

Superfluous Casts

- GC proposal(s) require superfluous* casts
  - *only necessary to make wasm type-check
- Virtual methods and closures often have to cast “this”
- DaCapo Benchmark Suite for Java (2009)
  - Over 100 million superfluous casts due to virtual methods
  - 4 casts per microsecond (on basic desktop in 2017)

More Casts

- Java (Double[] doubles)
  - double sum = 0.0; for (Double d : doubles) {sum += d.doubleValue();}
  - 1 superfluous cast to Double per iteration (assuming doubleValue is inlined)
- C# (double[] doubles)
  - double sum = 0.0; foreach (double d in doubles) { sum += d; }
  - 1 superfluous cast per iteration
- OCaml (doubles : float list)
  - List.fold_left (+.) 0.0 doubles;;
  - 2 superfluous casts per iteration (only 1 if everything inlined)

Only existing practical technique
for eliminating superfluous casts
is incompatible with proposed subtyping

5. Restricts Implementations

Needlessly Unifies Representations

- Should code pointers be same as WASI capabilities?
  - funcrefs may or may not be closures
  - WASI capabilities might just be (abstract) integer handles
- Should anyrefs always have a run-time (type) tag?
  - Distinguishing own refs from other modules' refs is important for correctness (no unintended coincidences)
  - Should funcrefs be tagged? If so, how?

6. No One Else Does This

For a good reason?

How to Implement

Main Required Change in Code Bases

```
bool is_subtype_of(type t1, type t2) {
        /*
        …
        */ return is_equal_to(t1, t2);
}
```

That’s All It Takes To

1. Not ship a feature with no established uses
2. Better serve external references
3. Avoid many complications for WebAssembly
4. Keep path open for making wasm more efficient
5. Permit more flexible wasm implementations
6. Heed advice of related systems and experts