Carp

A Language for the 21st Century

Veit Heller

September 15, 2018

Port Zero

man carp

```
man carp
```

- a Lisp-1
- type-inferred
- borrow-checked
- compiles to C
- for realtime applications

- a Lisp-1
- type-inferred $\Rightarrow$ statically typed, at no extra charge
- borrow-checked $\Rightarrow$ no GC, at no extra charge
- compiles to C
- for realtime applications

- Haskell implements a Hindley-Milner type system and inference
  $\Rightarrow$ You don't have to spell types out!
- Rust implements borrow checking
  $\Rightarrow$ You don't have to manually manage memory, even without a GC!

Let's put those things together (after simplifying) and rejoice!

$\Rightarrow$ Also add some Lisp macro goodness and a near-seamless C FFI for good measure!

source carp

```
; (type f)
; f : (Fn [(Ref (Array a)), Int, Int] a)
(defn f [x y z]
  @(Array.nth x (* y z)))
```

Listing 1: A silly Carp function

```
(deftype (AssocArray a b) [lst (Array (Pair a b))])
```

Listing 2: An associative array type, simplified.

open demo.live

cmp clojure carp

Carp	Clojure
Statically typed	Dynamically typed
Compiled	Kind of compiled?
Works with C	Works with Java (and JS)
Modules	Namespaces
Compile-time metaprogramming	full metaprogramming
Interfaces	Protocols

exit

```
trap
```

Carp is early stage software.

⇒ Small community, few packages

⇒ We're less than a handful of maintainers

⇒ Insufficient documentation

⇒ May change under your feet

⇒ May blow up in your face!

We're approaching the first stable release (0.3)

Thank you!

Questions?

Slides at `https://github.com/hellerve/carp_talks`