

Carp

A Language for the 21st Century

Veit Heller

August 25, 2018

Port Zero

whoami

PL nerd

CTO @ Port Zero

Carp standard library maintainer

Secretly a turtle

man carp

a Lisp-1

type-inferred

borrow-checked

compiles to C

for realtime applications

a Lisp-1

~~type-inferred~~ statically typed, at no extra charge

~~borrow-checked~~ no GC, at not extra charge

compiles to C

for realtime applications

Haskell implements a Hindley-Milner type system and inference

⇒ You don't have to spell types out anymore!

Rust implements borrow checking

⇒ You don't have to manually manage memory, even without a GC!

Let's put those things together and rejoice!

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

source carp

```
(defn f [x y z]  
  (Array.nth x (* y z)))
```

Listing 1: A silly zepto function

Carp has a typed (but generic) hashmap/dictionary type.

look hashmap

It is not a builtin type.

Let's briefly look at a simple hashmap implementation

A hash function determines the placement of an element in an array of arrays.

We append the element to the array inside the other to deal with hash collisions.

Lookup combines hashing and a linear search.

Insert table here

```
(deftype (Map a b) [buckets (Array (Array (Pair a b)))])
```

Listing 2: The hashmap type, simplified.

look hashmap

```
(defmodule Map
  (def dflt-len 256)

  (defn create []
    (init (Array.repeat dflt-len Array.zero)))

  (defn put [map key value]
    ; ...
  )
)
```

Listing 3: The hashmap module, with omissions.

look hashmap

```
(defn put [map key value]
  (let [buckets (buckets map)
        len (Array.length buckets)
        idx (Int.mod (hash key) len)
        bucket @(Array.nth buckets idx)
        pair (Pair.init @key @value)
        new-bucket (Array.push-back bucket pair)
        new-buckets (Array.aset @b
                                  idx
                                  new-bucket)]
    (set-buckets @map new-buckets)))
```

Listing 4: Defining put.

open demo.live

exit

Carp is early stage software.

- ⇒ Small community, few packages
- ⇒ Insufficient documentation
- ⇒ May change under your feet
- ⇒ May blow up in your face!

We're approaching the first stable release (0.2)

Thank you!

Questions?