# Language-oriented programming in Racket

## —A cultural anthropology—

Jesse Alama

**jesse@lisp.sh**

# WTF is “language-oriented programming”?!

# WTF is “language-oriented programming”?!

I kinda-sorta know.

# WTF is “language-oriented programming”?!

I kinda-sorta know.

There are some canonical resources.

# WTF is “language-oriented programming”?!

I kinda-sorta know.

There are some canonical resources.

But I’m just one Racket programmer.

# WTF is “language-oriented programming”?!

I kinda-sorta know.

There are some canonical resources.

But I’m just one Racket programmer.

How does the idea play out in practice among other Racketeers?

# Survey: Question 1

Can you point to an example or two of your own work that best exemplifies what you consider to be LOP? Even if you haven't made your own language, but work within (or teach) something other than "full-on" Racket, feel free to mention it.

# Survey: Question 2

If you've made your own language, what was your motivation? Did you just want to experiment? Was there some pain that you faced that a new language could alleviate?

# Survey: Question 3

What about an example or two of LOP in the work of others that you find impressive or inspiring (not even necessarily from the Racket world)?

# Survey: Question 4

Can you identify an “aha!” moment or two when your understanding of LOP evolved?

# Survey: Question 5

Is there a book, paper, video, course, or any other material, that you would recommend to Racket newcomers to help unlock LOP? How did you yourself get started with that idea?

# Survey: Question 6

If you've made you own language, do you recall whether there were moments where you hesitated before going down that path? Or, once you were on that path, were there stumbling blocks hindered your progress? If you've not made your own language but have considered it, what uncertainties do you face?

# Survey: Question 7

The technicalities and mechanics that go into making your own language can be difficult compared to other "everyday" ideas in computer science: lexing, parsing, macros, phase separation, evaluation, semantics. Do you find some of these ideas easier to appreciate (and execute on) than others?

# Survey: Question 8

What's your experience talking with others outside the Racket world about Racket and its language-oriented philosophy? Do you find that there are frequently recurring sticking points, misunderstandings, or other obstacles in discussions about "make your own language"?

# Survey: Question 9

Can you of situations where “make your own language” is definitely not an appropriate way to tackle a problem? Have you yourself made your own language only to realize later that it adds little value, or was over-engineering? When do you decide to make a language as opposed to just making a library?

# The Responses

- Surveyed people by looking through the official PLT list, list of people participating in previous RacketCons, etc.
- Responses came in from 30 Racket developers.
- I made an ebook out of it. It weighs 315 pages.

# LOP Groups

- enthusiastic embracers

# LOP Groups

- enthusiastic embracers
- jaded professionals

# LOP Groups

- enthusiastic embracers
- jaded professionals
- fellow travelers

# LOP Groups

- enthusiastic embracers
- jaded professionals
- fellow travelers
- “mere” users

# LOP Groups

- enthusiastic embracers
- jaded professionals
- fellow travelers
- “mere” users
- would-like-to-ers

# LOP inside Racket

- Racket itself is a large-scale example
- A new `#lang`
- Using a metalanguage (e.g., `s-exp`)
- Reader hacking
- Racket + macros (no `#lang`)
- Interpreter that evaluates S-expressions

# LOP outside Racket

- Ward's language-oriented programming (1994)
- Spoofax
- MPS (JetBrains)
- Macros & macro-like features in other Lisps & non-Lisp languages
- Embedded DSLs in Haskell, JavaScript, etc.

# Our Problems

# Rough path ahead

Racket smoothes the path for making a language.

# Rough path ahead

Racket smoothes the path for making a language.

BUT:

# Rough path ahead

Racket smoothes the path for making a language.

BUT:

The “API” is still a a bit hard, even for experienced Racketeers.

# Evangelism

Promoting Racket is  hard

# Surface syntax!

- Even some Racketeers don’t especially like the parentheses.
- Others work as long as possible with S-expressions.
- When selling Racket to others, surface syntax might be crucial.

# Recurring themes

- *Beautiful Racket*
- Racket Summer School
- Teaching languages
- Racket itself as an examplar of LOP
- Exemplary languages: Scribble, Typed Racket, Hackett, Turnstile, Redex

# The stars of the show

Eli Barzilay

Annaia Berry

Jörgen Brandt

Matthew Butterick

Ngugyen Linh Chi

Christos Dimoulas

Joel Dueck

# The stars of the show

Kathi Fisler

Matthew Flatt

Spencer Florence

Stephen Foster

Tony Garnock-Jones

Panicz Godek

Ben Greenman

# More stars!

Eric Griffis

Andrew Gwozdziewycz

William Hatch

Shriram Krishnamurthi

Jay McCarthy

Darren Newton

Pavel Panchekha

# More stars!!

Daniel Prager

Praghakar Ragde

Michael Sperber

Vincent St-Amour

Asumu Takikawa

Éric Tanter

Emina Torlak

# Final stars

Jesse Tov

Jon Zeppieri

# Final stars

Jesse Tov

Jon Zeppieri

***YOU?***