how do i do research

matthias felleisen,
racketeer,
plt, northeastern

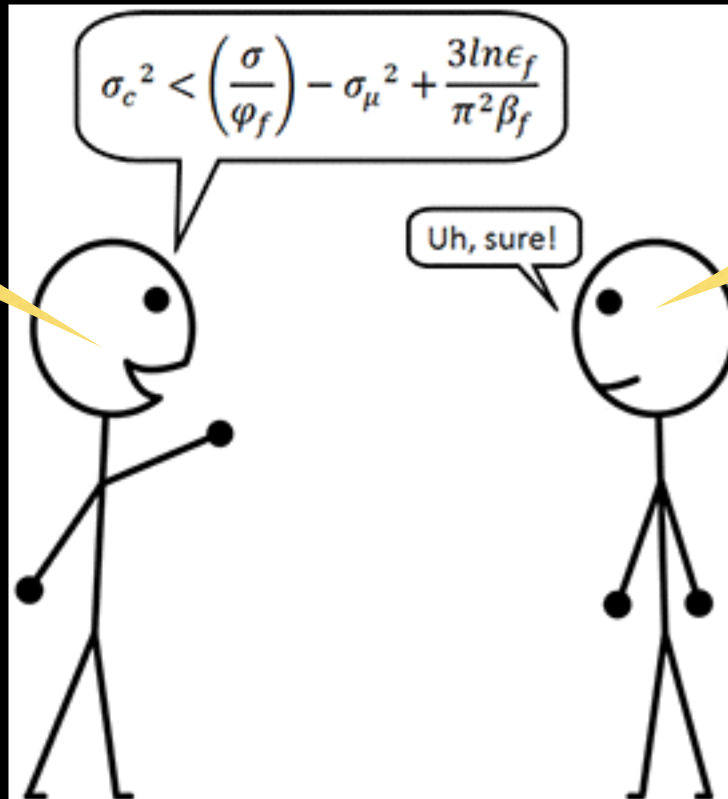
how do *I* do research?

how ~~do~~ I do research?

would

if I were you

your advisor



& you

how do *I* work with my
my PhD students?

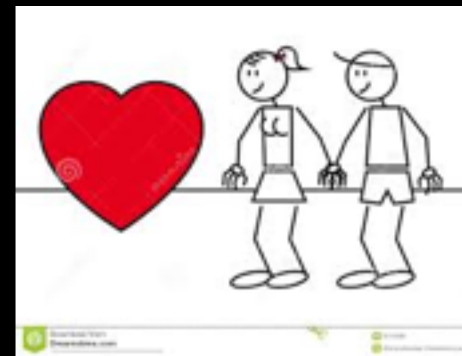
how would I do
research if I were you?

how do *I* do research?

how do i relate to my PhD students

I have *never, ever* hired a PhD student. Period.

Instead my students and I find a topic we both love.

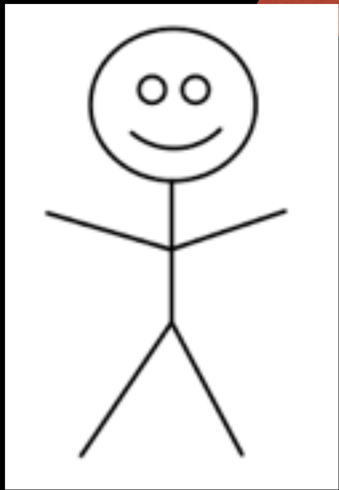


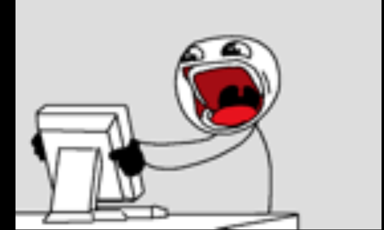
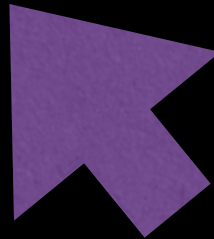
Functional I/O (ICFP ’09)

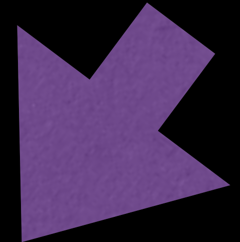
Typed Racket
(ICFP ’10)

Compiler
Coaching (OOPSLA ‘12)

Laziness, what is it (good for)?
(JFP 1996)





And that's what's called 'doing research.'

how would I do research if I were you

Two Case Studies

Asumu
Takikawa

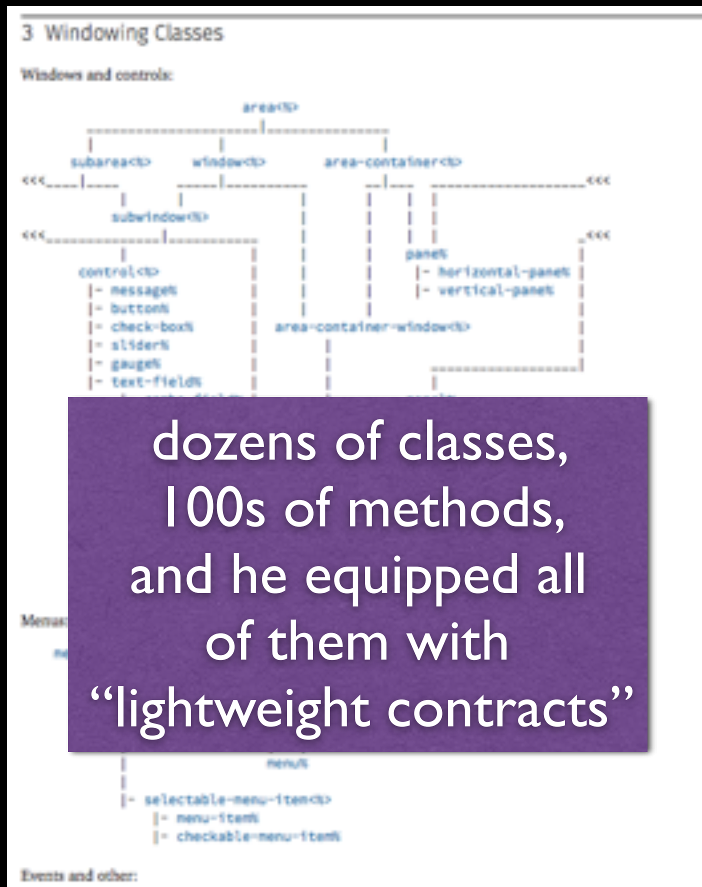
Tony
Garnock-Jones

Kuhn, *The Structure of Scientific Revolution*

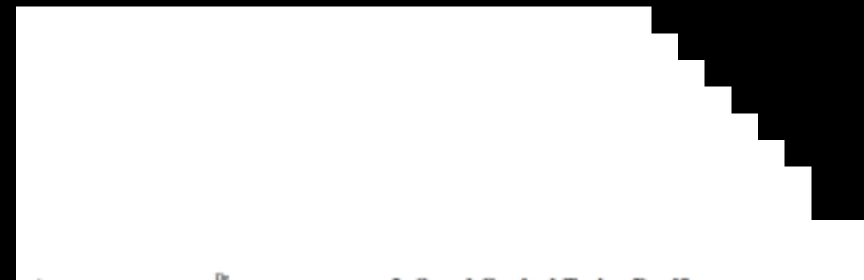
Types for Classes

Typed Racket
(ICFP '10)

Topic: *Gradual* Types for *First-class* Classes



dozens of classes,
100s of methods,
and he equipped all
of them with
"lightweight contracts"



Is Sound Gradual Typing Dead?

Programming Objects with ML-ART
An extension to ML with Abstract and Record Types

Didier Rémy

INFORMATION AND COMPUTATION 93, 1-15 (1991)

Type Inference for Record Concatenation and Multiple Inheritance*

MITCHELL WAND

Complete Type Inference for Simple Objects

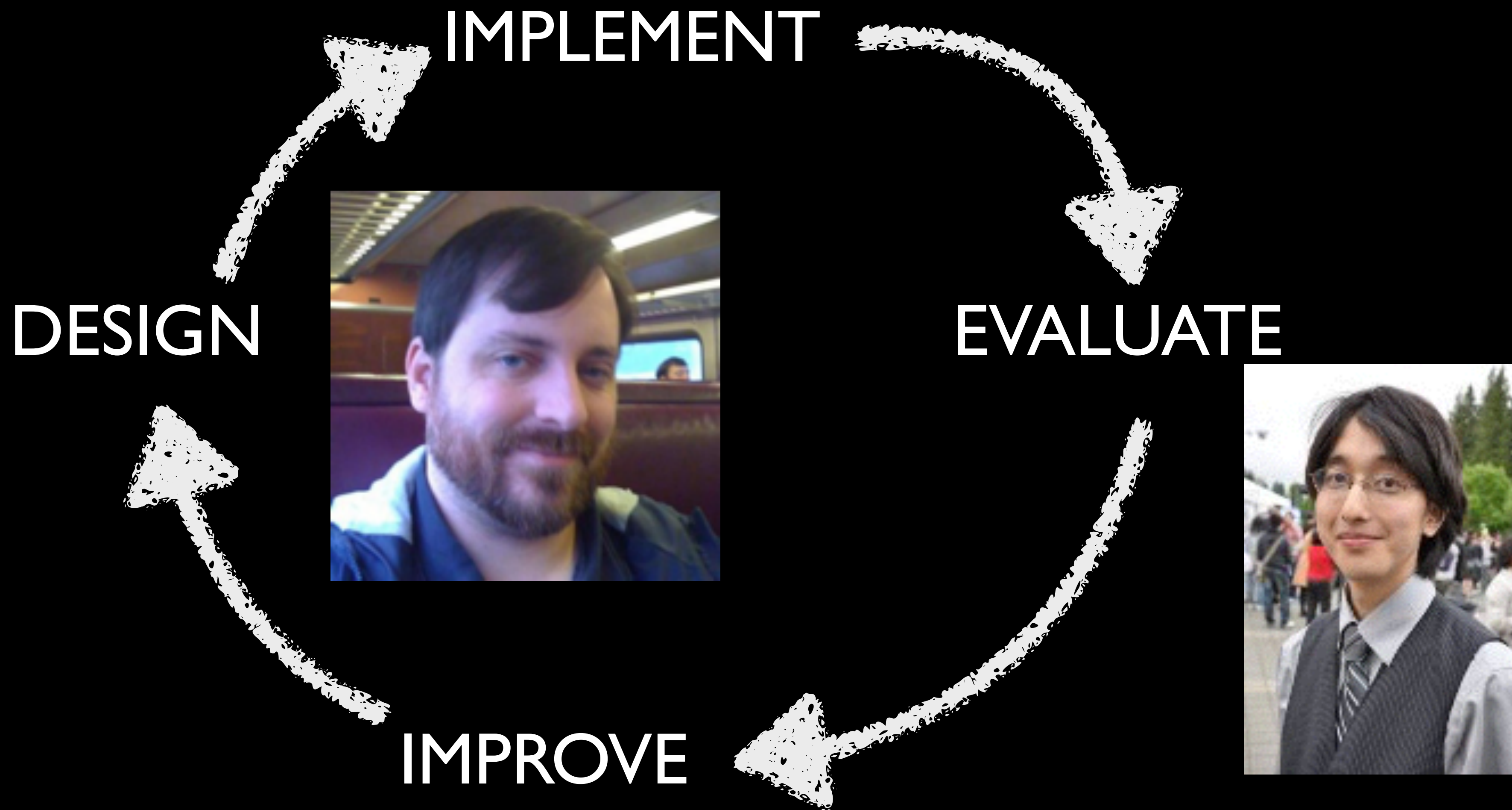
Mitchell Wand

College of Computer Science
Northeastern University

Abstract

1. Introduction

Previous Topic: Contracts for Classes & Objects



Takikawa & Greenman '15

Is Sound Gradual Typing Dead?

Dr. Double B. Reviewing, I — In Famous University — turing@award.com
Dr. Double B. Reviewing, II — In Famous University — turing@award.com
Dr. Double B. Reviewing, III — Somewhat Famous University — turing@award.com
Dr. Double B. Reviewing, VI — Less Famous University — turing@award.com

EVALUATE

Towards Practical Gradual Typing

Asumu Takikawa, Daniel Feltey, Earl Dean, Robert Bruce Findler, Sam Tobin-Hochstadt, Felleisen

IMPLEMENT

Takikawa '15

DESIGN

Takikawa & Strickland '13

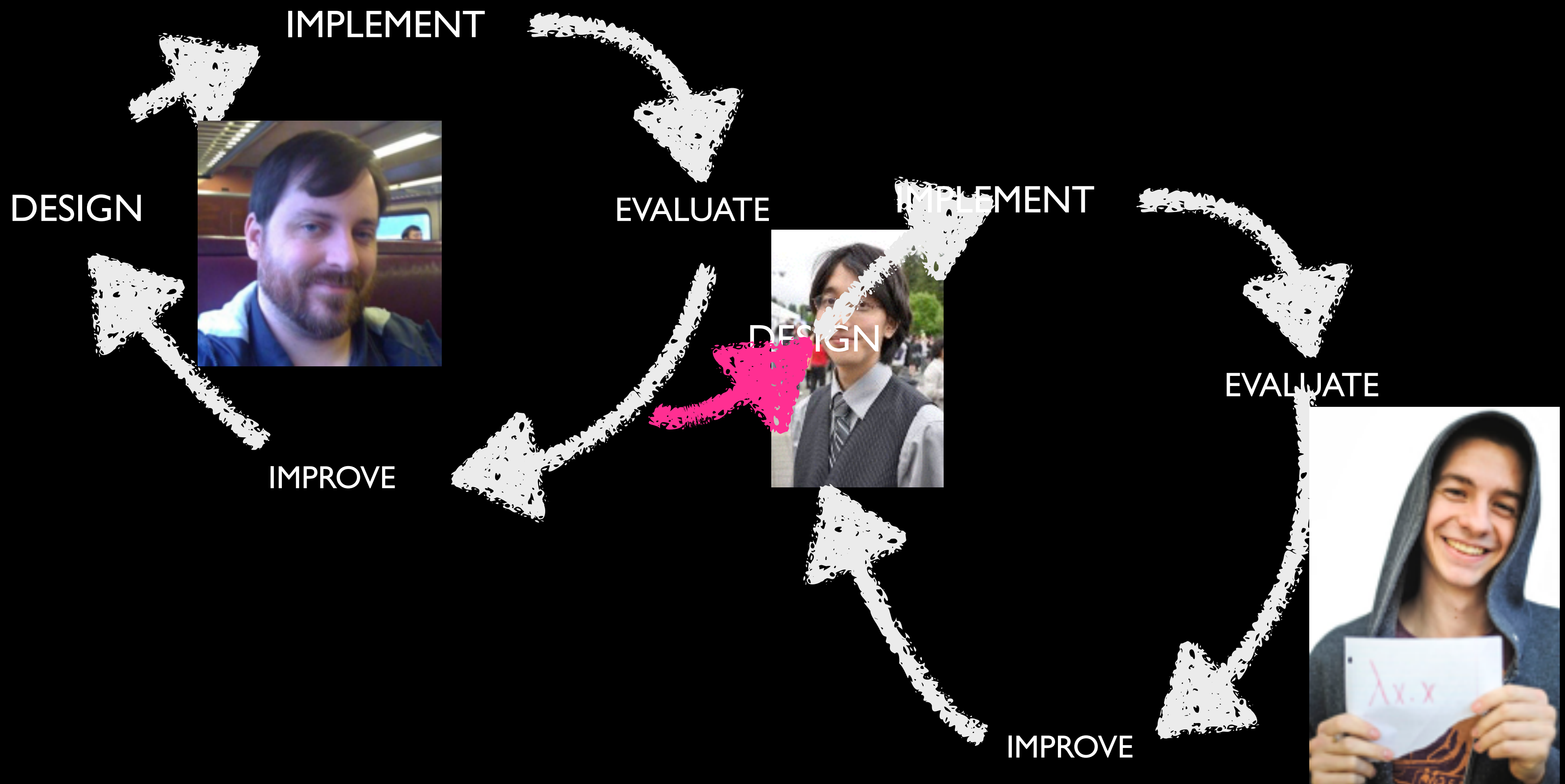
Contracts for First-Class Classes

T. STEPHEN STRICKLAND, MATTHIAS FELLEISEN

EVALUATE

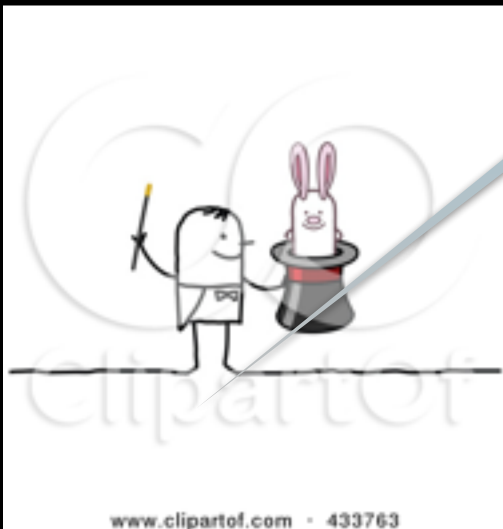
Strickland & Takikawa '12

A Positive (Self-perpetuating) Feedback Loop



Functional I/O (ICFP '09)

RabbitMQ

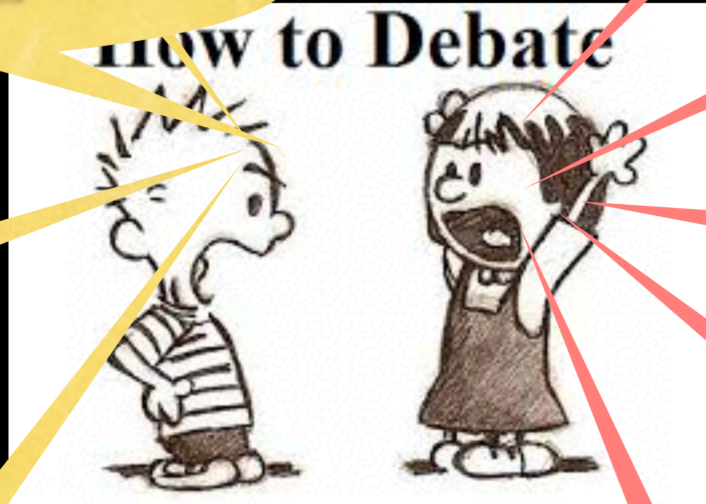


truly functional GUIs

messages as events

communicating worlds

from freshman programs to systems



actors

networks

publish subscribe

failures!

message brokers

Functional I/O &
Communicating Worlds

networking systems

DNS Proxy

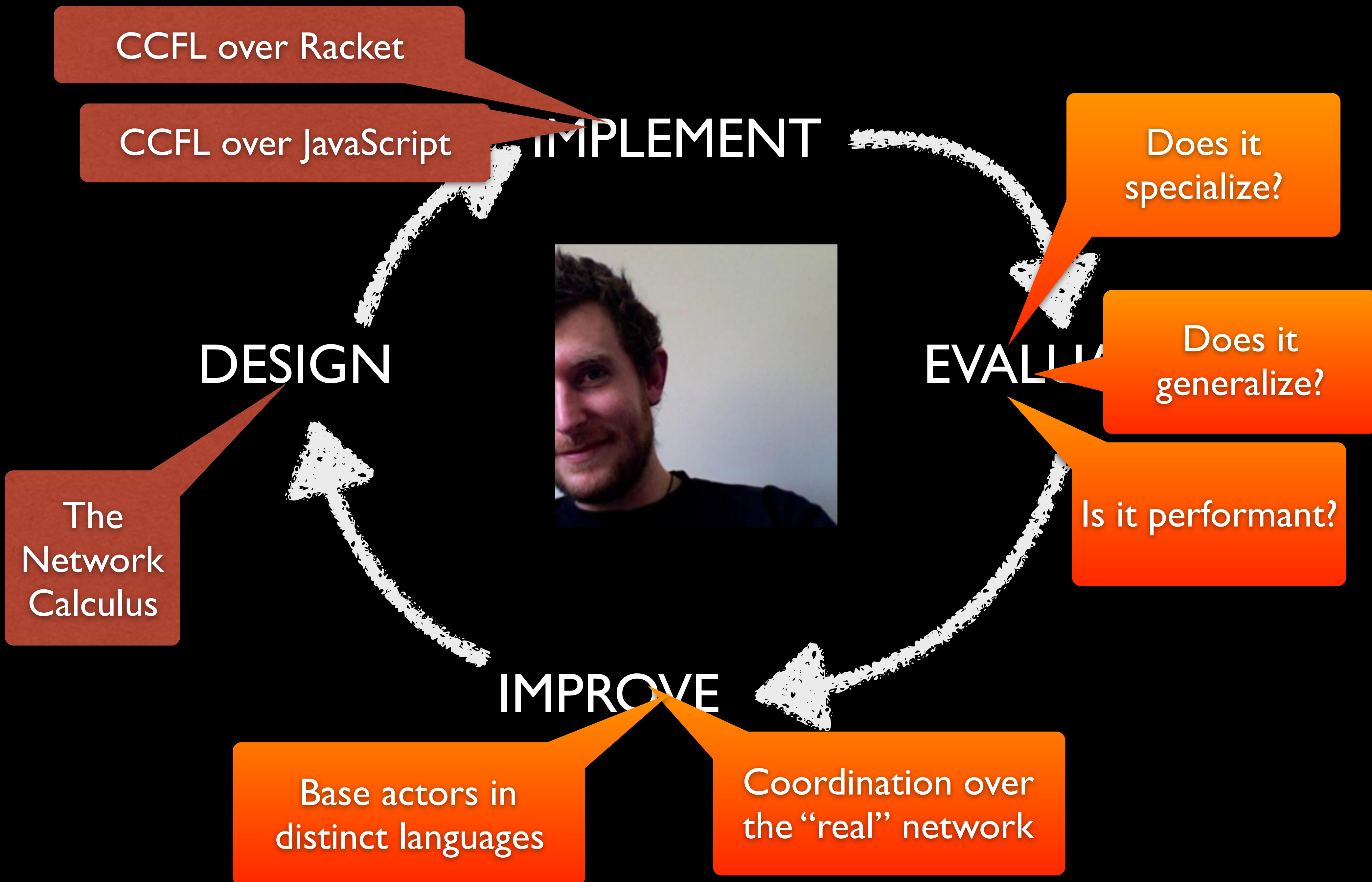
SSH Server

Chat Room

TCP Stack

DSL for comm. actors

Topic: *Coordinated Concurrent Functional Language*



What is the cost of breaking open a new field?

5 years

6.5 years

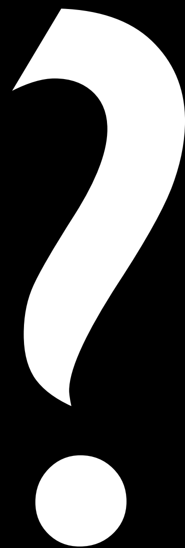
how did I do research as a PhD student

My Story

Dan Friedman

```
(f (g (call/cc k))
        =
(k (λ (x) (f (g x))))
```

Go, implement it. See what happens.



What does it mean to *implement equations*

Theoretical Computer Science 1 (1975) 125–159. © North-Holland Publishing Company

CALL-BY-NAME, CALL-BY-VALUE AND THE λ-CALCULUS

G. D. PLOTKIN

Department of Machine Intelligence, School of Artificial Intelligence, University of Edinburgh, Edinburgh, United Kingdom

Communicated by R. Milner
Received 1 August 197

Abstract. This paper examines the old question of the relation between ISWIM and the λ-calculus, using the distinction between call-by-value and call-by-name. It is held that the relationship should be mediated by a standardisation ... a new λ-calculus is introduced ...

I had read that paper.

… in two hours.

I read it again.

NOT 4 hours

NOT 4 days

I spent 4 MONTHS studying this paper.

What does it mean to *implement equations*

Theoretical Computer Science 1 (1975) 125–159. © North-Holland Publishing Company

CALL-BY-NAME, CALL-BY-VALUE AND THE λ-CALCULUS

G. D. PLOTKIN

Department of Machine Intelligence, School of Artificial Intelligence, University of Edinburgh, Edinburgh, United Kingdom

Communicated by R. Milner
Received 1 August 1974

Abstract. This paper examines the old question of the relationship between ISWIM and the λ-calculus, using the distinction between call-by-value and call-by-name. It is held that the relationship should be mediated by a standardisation theorem. Since this leads to difficulties, a new λ-calculus is introduced whose standardisation theorem gives a good correspondence

$+$

$\not\subset$

STUDIES IN LOGIC
AND
THE FOUNDATIONS OF MATHEMATICS

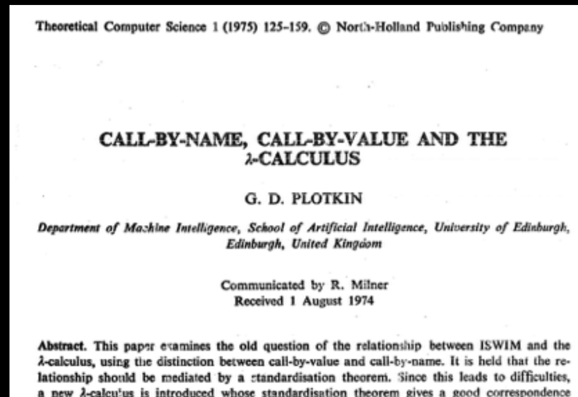
The Lambda Calculus
Its Syntax and Semantics

REVISED EDITION

H.P. BARENDREGT

NORTH-HOLLAND
AMSTERDAM • NEW YORK • OXFORD

What did four months of reading yield


Theoretical Computer Science 1 (1975) 125–159. © North-Holland Publishing Company

CALL-BY-NAME, CALL-BY-VALUE AND THE λ-CALCULUS

G. D. PLOTKIN

Department of Machine Intelligence, School of Artificial Intelligence, University of Edinburgh, Edinburgh, United Kingdom

Communicated by R. Milner
Received 1 August 1974

Abstract. This paper examines the old question of the relationship between ISWIM and the λ-calculus, using the distinction between call-by-value and call-by-name. It is held that the relationship should be mediated by a standardisation theorem. Since this leads to difficulties, a new λ-calculus is introduced whose standardisation theorem gives a good correspondence

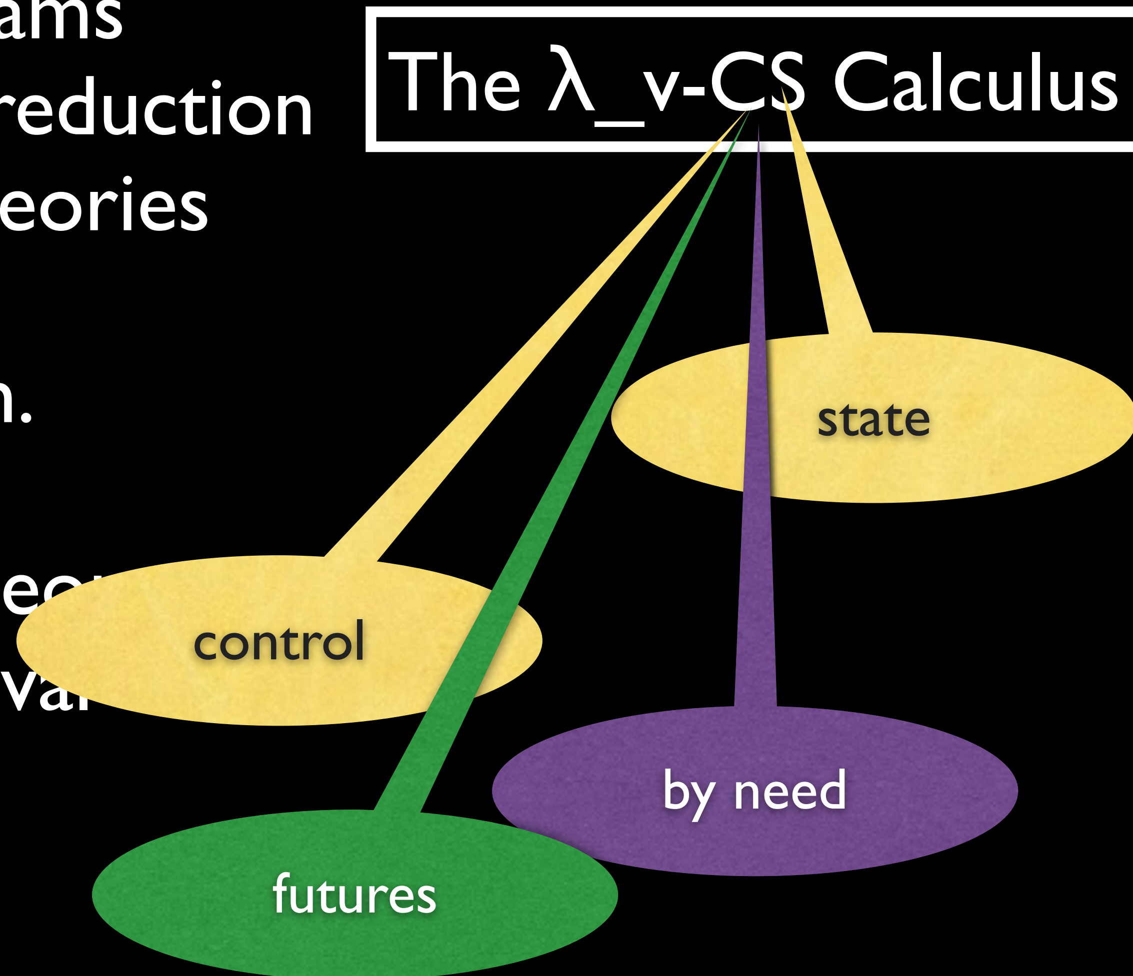
How do calculi correspond to eval?

- start from an abstract syntax
- identify values & programs
- define basic notion of reduction
- inductively generate theories
- eval-> and eval=
- Church & Rossser Thm.
- Thm. eval-> = eval=
- Standard Reduction Theorem
- Thm: eval-standard = eval->

My dissertation: “This” works for imperative features, too.

How do calculi correspond to eval?

- start from an abstract syntax
- identify values & programs
- define basic notion of reduction
- inductively generate theories
- eval-> and eval=
- Church & Rossser Thm.
- Thm. eval-> = eval=
- Standard Reduction Theo
- Thm: eval-standard = eva



Lessons

Know to distinguish the *good* from the *bad*
in your advisor’s suggestions.

Good paper require ‘deep study’
not just a ‘reading.’

Really good paper are ‘research programs’
not just results.

how do I do research now

problem I can solve

problem I can solve

problem I can solve

problem I can solve

problem I can solve

problem I can solve

problem I can solve

problem I can solve

problem I can solve

problem I can solve

problem I can solve

problem I can solve

problem I can solve

paper I can write

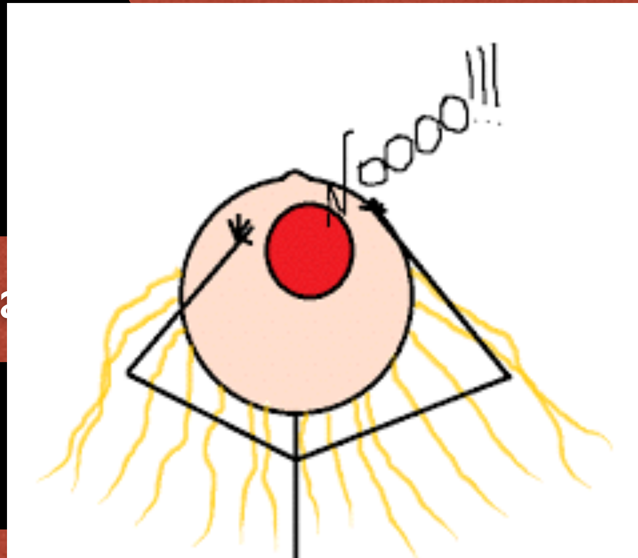
paper I can write

paper I can write

paper I can write

paper I can write

er I can write



pa

paper I can write

paper I can write

paper I can write

paper I can write

paper I can write

paper I can write

paper I can write

NOOOOOOO

More papers does **not** mean better researcher.

Think *big*, think long-term.

Lesson

Good researchers say “no” to many problems.
They focus on those that they care about.

My Long-term Projects

How can programmers design programs *systematically*?
(1985)

How do types fit into untyped languages?
(1988)

How do you teach 12, 14, 16 year olds programming and what benefit does this have?
(1995, last day of POPL)

What is linguistic power and why is a DSL better than an algorithm?
(1985)

What do such long-term projects look like?

How do you launch long-term projects?

What do such long-term projects look like?

How do types fit into untyped languages? (1988)

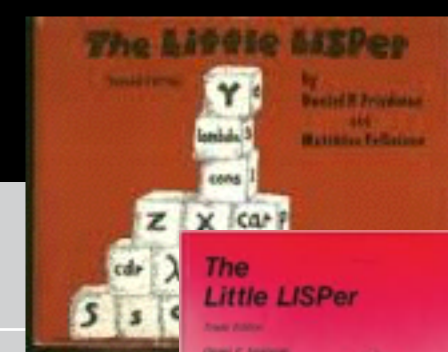
The "Gradual Typing" Dissertations

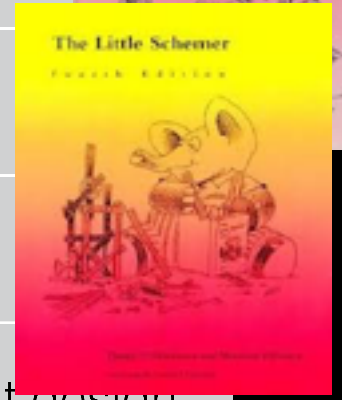
1990	Mike Fagan	Soft Typing (***)
1994	Andrew Wright	Practical Soft Typing
1998	Cormac Flanagan	Componential SBA
2002	Robby Findler	Higher-order Contracts
2005	Philippe Meunier	Modular SBA from Contracts
2006	Sam Tobin-H. (2010)	From Scripts to Programs
2012	Stevie Strickland	Contracts for First-class Classes
2015	Asumu Takikawa	Types for First-class Classes

How can programmers design programs *systematically*?

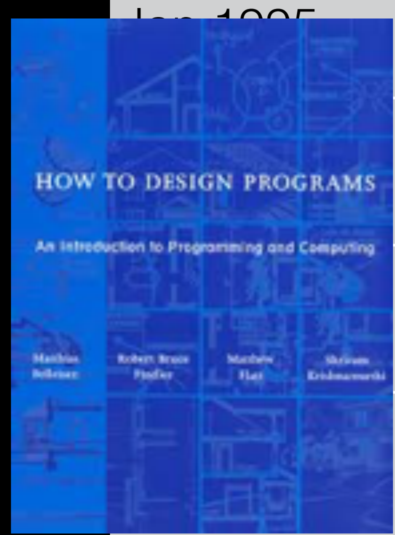
How do you teach 12,

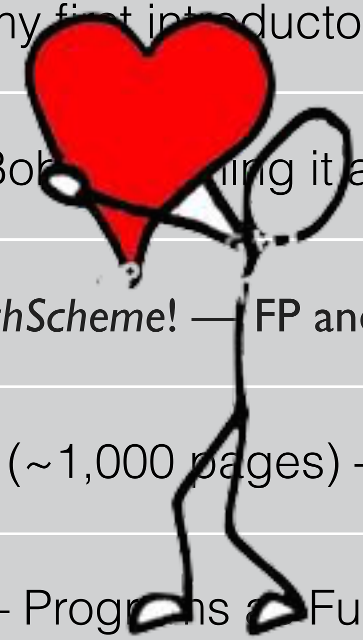
How to Design





1985	The Litte Lisper, 2nd ed.
F'1992	Teaching my first introductory programming course
1993/94	@ CMU, "Bob ... ing it all wrong"
	Launch *TeachScheme*! — FP and algebra in high schools
	The Dump (~1,000 pages) — re-focusing on explicit design
	HtDP/1e — Progr...
	Designing, implementing, eva...
2007-2015	HtDP, 2nd ed. — Programs are *n*...





What is linguistic power and why is a DSL better than an algorithm?
(1985)

"Macros"

1985	with Kohlbecker et al	Hygienic Maros
1986	with Bruce Duba	Macros in Phases
88/89	John Greiner, Steve Weeks	Programming Abstract Syntax
1991	Todd Yonkers	Extensible Syntax
1994	Matthew Flatt	Connecting DSLs into Applications
95/97	PLT	Teaching languages
95/99	Shriram Krishnamurthi	Parameterizing over Language
2002	Matthew Flatt	You want it when?
03/08	Ryan Culpepper	Protecting Macros
08/09	Ryan Culpepper	Debugging Macros

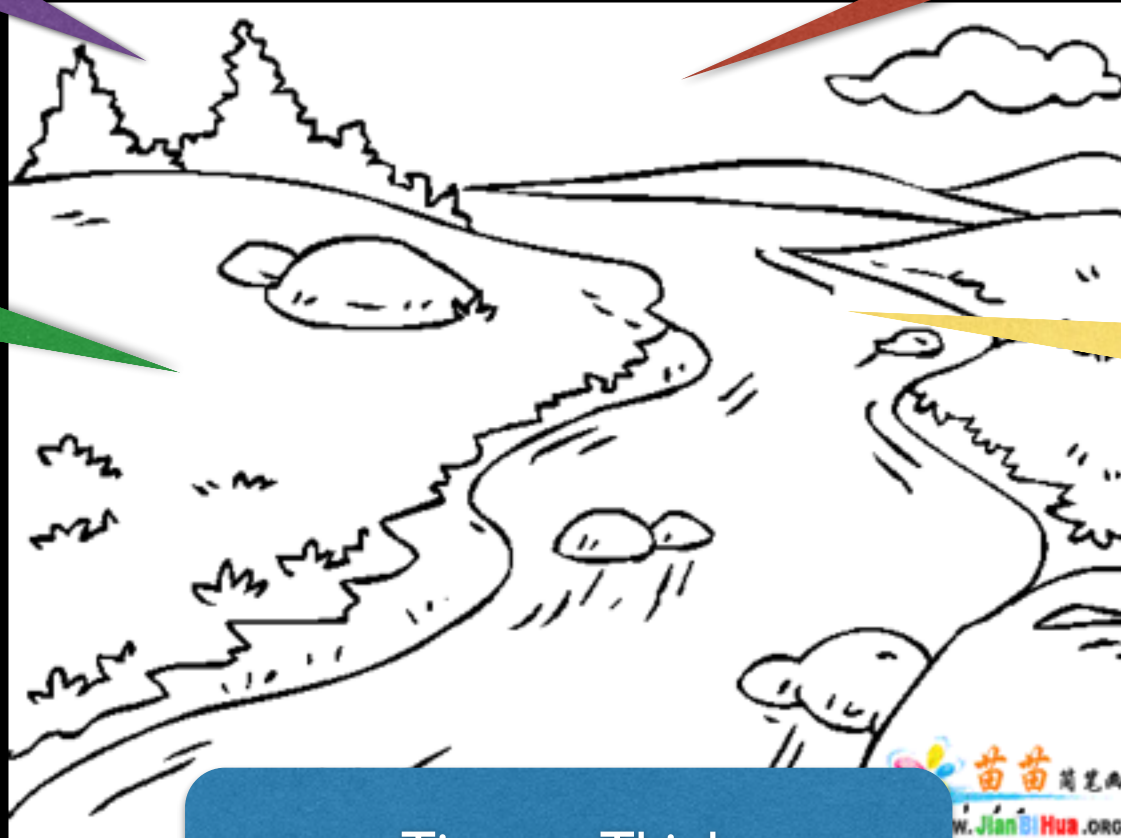
How do you launch
long-term projects?

People

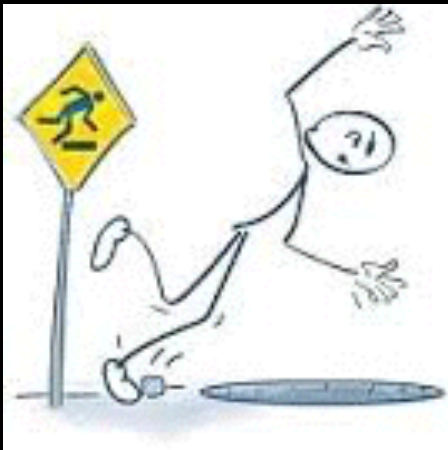
Readings

Teaching

“Reality”



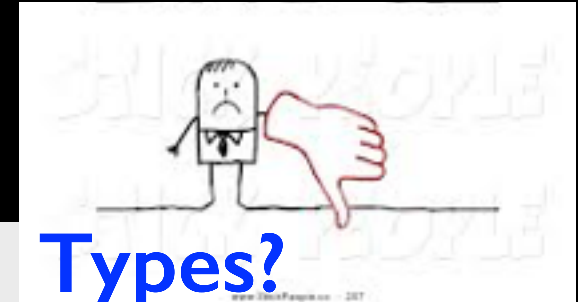
Time to Think



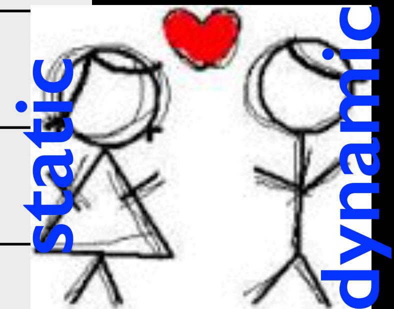
Sometimes you stumble into a topic.

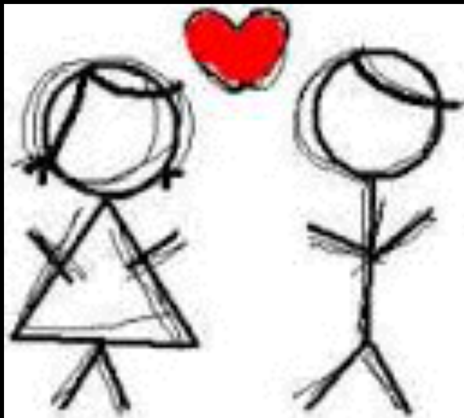
How do types fit into untyped languages? (1988)

The “Gradual Typing” Dissertations



1990	Mike Fagan	Soft Typing (***)
1994	Andrew Wright	Practical Soft Typing
1998	Cormac Flanagan	Componential SBA
2001	Robby Findler	Higher-order Contracts
2005	Philippe Meunier	lar SBA from Contracts
2006	Sam Tobin-H. (2	ts to Programs
2012	Stevie Stri	rst-class Classes
2015	Asum	Classes





Sometimes it is love at first sight.

How can programmers design programs *systematically*?

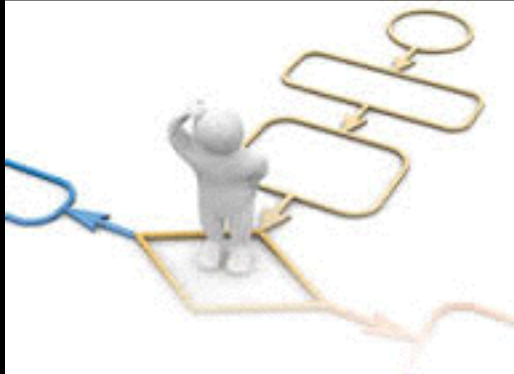
How do you teach 12, 14, 16 year olds programming What benefit does it have? (1995, las

1985	The Litte Lisper, 2nd ed.
F'1992	Teaching my first introductory programming course
1993/94	@ CMU, "Bob's teaching it all wrong"
Jan 1995	Launch *TeachScheme*! — FP and algebra in high schools
c 1995	The Dump (~1,000 pages) — re-focusing on explicit design
96-2001	HtDP/1e — Programs
2002-2005	Designing, implemen /O"
2007-2015	HtDP, 2nd ed. — Progr

An "entertaining" thought

Cormac asked the one critical question

We knew what we had to do: software, curriculum, teaching



Sometimes it develops as a necessity.

What is linguistic power and why is a DSL better than an algorithm? (1985)

"Macros"

1985	with Kohlbecker et al	Hygienic Maros
1986	with Bruce Duba	Macros in Phases
88/89	John Greiner, Steve Weeks	Programming Abstract Syntax
1991	Todd Yonkers	Extensible Syntax
1994	Matthew Flatt	Connecting DSLs into Applications
95/97	PLT	Teaching languages
95/99	Shriram Krishnamurthi	Parameterizing over Language
2002	Matthew Flatt	You want it when?
03/08	Ryan Culpepper	Protecting Macros
08/09	Ryan Culpepper	Debugging Macros

what to remember?

As a student, you need to
— develop a sense of the landscape
— follow your heart
— plan out design, implementation, evaluation.

No matter what, keep in mind that the number of your papers is ***unrelated*** to the quality of your work.

As a researcher, I
— look for long-term projects
— follow my heart
— use teaching (for the 99%) for inspiration
— develop dissertation-size goals
— plan for hand-over
— and have my eyes open for new ideas.

The End