

J is for JavaScript

Olivier Danvy (Aarhus University)

Ken Shan (Rutgers)

Ian Zerny (Aarhus University)

DSL

Oxford, 15 July 2009

1

Peter J. Landin

Founding father of

- DSL's and
- everything λ in programming languages.

`http://www.brics.dk/~danvy/
tmp/tribute-peter-landin.pdf`

2

Then and now

Then: the λ -calculus (applicative expressions).

Now: JavaScript.

3

Meta-language vs. target language

interpreter vs. compiler

4

Meta-language vs. target language

interpreter vs. compiler

The first Futamura projection.

5

General linguistic device

Compositional embedding into a meta-language.

6

General linguistic device

Compositional embedding into a meta-language.

But how about handling jumps?

7

Specific linguistic devices

Handling jumps:

- The J operator (Landin).
- Continuations and CPS (Wadsworth and Strachey).

8

CPS

- Rabbit (Steele).
- SML/NJ (Appel).

9

CPS: too much?

- Monadic style / A-normal forms.
- Making the CPS transformation selective.

10

Back to JavaScript

- CPS and the issue of tail calls.
- Selective CPS: a partial solution.
- No CPS at all: use J.

11

Our paper

Retarget Landin's embedding of ALGOL 60
from the λ -calculus to JavaScript with J.

12

The translation (1/2)

- declarations translate to declarations
- blocks translate to blocks
- commands translate to commands
- function calls translate to function calls
- etc.

13

The translation (2/2)

- label declarations translate to J,
to capture a continuation
- jumps translate to calls
to captured continuations

14

A middle ground

Not just program templates but language idioms:

- Compilers are geared
for programs people write.
- Gearing JavaScript processors
for programs that are generated.

15

Practical conclusion

Towards “Common JavaScript”
(by idealized analogy with Common Lisp).
as translation target for DSL's.

16

Practical conclusion

Towards “Common JavaScript”
(by idealized analogy with Common Lisp).
as translation target for DSL's.

Peter Landin lives on.