
WELCOME...

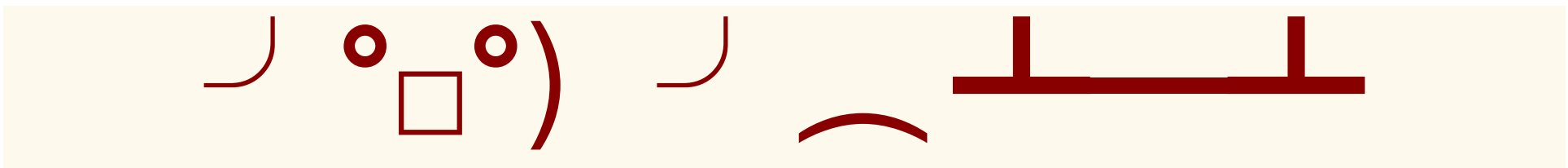
WHAT IS NEXT?

(A FRIENDLY GUIDE TO CHOOSING YOUR NEXT LANGUAGE)

DECEMBER 2017

@silverspoon

Disclaimer:



IRISH-ISMS AHEAD

CRAIC, EEJIT, ETC ARE TOTALLY OK WORDS

- Eejit – an idiot or a fool, but more often it's used in an affectionate (yet still mocking!) manner.
- Cop on - common sense
- Give out - complaint

THIS IS BACON

DELPHI

VB.NET

C# (SOME JS WHEN JQUERY WAS NEW, SOME JAVA)

F# / C#

SCALA

REFERENTIAL TRANSPARENCY

LESS MUTABLE STATE!!

NO EXCEPTIONS FOR FLOW CONTROL!!

.. AND MORE

BACON'S FRIENDS FELT AWKWARD

@silverspoon

BACON'S FP

- **Typed** FP
- FP everywhere
- Aspiring to purity / Total functions

@silverspoon

PROBLEMS WITH FP AS BACON UNDERSTANDS IT

- Dependency management
- Type tetris
- Complicated concepts

... is it worth it?... is it the best way?

@silverspoon

BACON DREAMS OF WELL STRUCTURED PROGRAMS

Well-structured software is easy to write and to debug, and provides a collection of modules that can be reused to reduce future programming costs. [Why FP matters. John Hughes]

@silverspoon

MEET OOOOO

- Works with Bacon
- Shipping is everything
- Curious about functional approach

"FUNCTIONAL PROGRAMMING HAS EMERGED SINCE THE MID-2000S AS AN ATTRACTIVE BASIS FOR SOFTWARE CONSTRUCTION. ONE REASON IS THE INCREASING IMPORTANCE OF PARALLELISM AND DISTRIBUTION IN COMPUTING." ODERSKY, ROMPF APRIL 2014

"...ESPECIALLY ITS (SCALA) FOCUS ON PRAGMATIC CHOICES THAT UNIFY TRADITIONALLY DISPARATE PROGRAMMING-LANGUAGE PHILOSOPHIES (SUCH AS OBJECT-ORIENTED AND FUNCTIONAL PROGRAMMING)

ODERSKY, ROMPF APRIL 2014


SOLID LOOKS A LOT LIKE FP WHEN YOU SQUINT

FROM THE PL DESIGNERS

Scala is very much about better component oriented programming for the Java platform. Although we do a good job of object oriented programming which is very nice in F#, we haven't thought to make fundamental improvements at the component level, in a sense. We are quite happy to say "You are making components? OK, make it a .NET component". — *Don Syme - March 2009*

"...[Scala] focus on pragmatic choices that unify traditionally disparate programming-language philosophies (such as object-oriented and functional programming). The key lesson is these philosophies need not be contradictory in practice. — *[Odersky, Rompf - April 2014]*

Regarding functional and object-oriented programming, one fundamental choice is where to define pieces of functionality (...) ...and Scala gives programmers the choice. — *[Odersky, Rompf - April 2014]*

Choice also involves responsibility, and in many cases novice Scala programmers need guidance to develop an intuitive sense of how to structure programs effectively. — *[Odersky, Rompf - April 2014]*


When Oooo and Bacon talk, they often disagree and call each other names

DOING + THINKING

WE BUILD SYSTEMS WITH:

LANGUAGE(S)

TOOLS: LIBRARIES, FRAMEWORKS

CONTEXT: USERS AND COMMUNITY

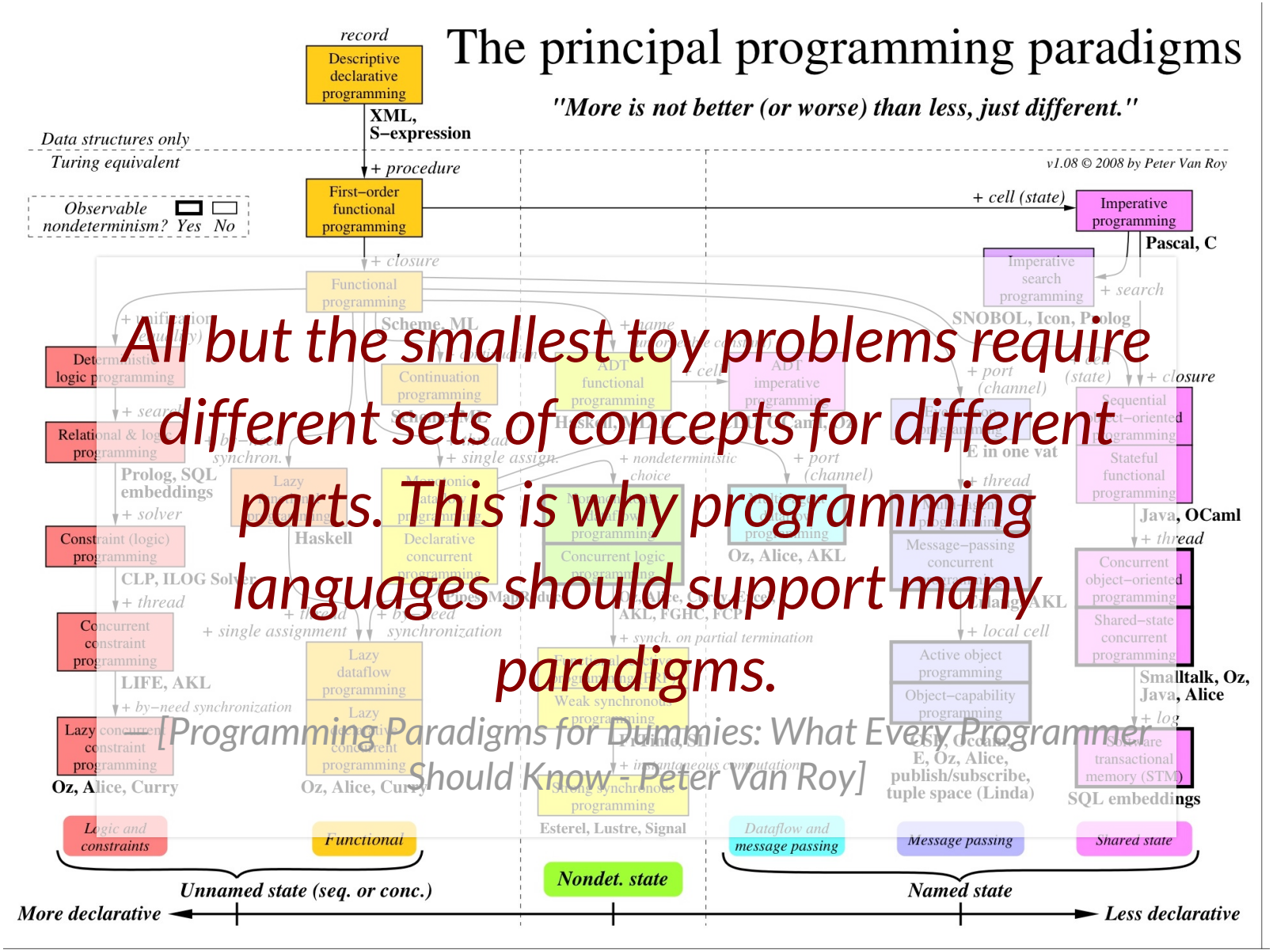
BELIEFS MATTER

PARADIGMS

A PROGRAMMING PARADIGM

...is an approach to programming a computer based on a mathematical theory or a coherent set of principles.

— [Programming Paradigms for Dummies: What Every Programmer Should Know - Peter Van Roy]



All but the smallest toy problems require different sets of concepts for different parts. This is why programming languages should support many paradigms.

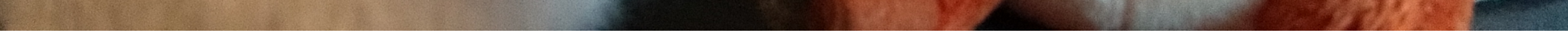
[Programming Paradigms for Dummies: What Every Programmer Should Know - Peter Van Roy]

A language should ideally support many concepts in a well-factored way, so that the programmer can choose the right concepts whenever they are needed without being encumbered by the others.

— [Programming Paradigms for Dummies: What Every Programmer Should Know - Peter Van Roy]

...it is certainly not true that there is one "best" paradigm

— [Programming Paradigms for Dummies: What Every Programmer Should Know - Peter Van Roy]



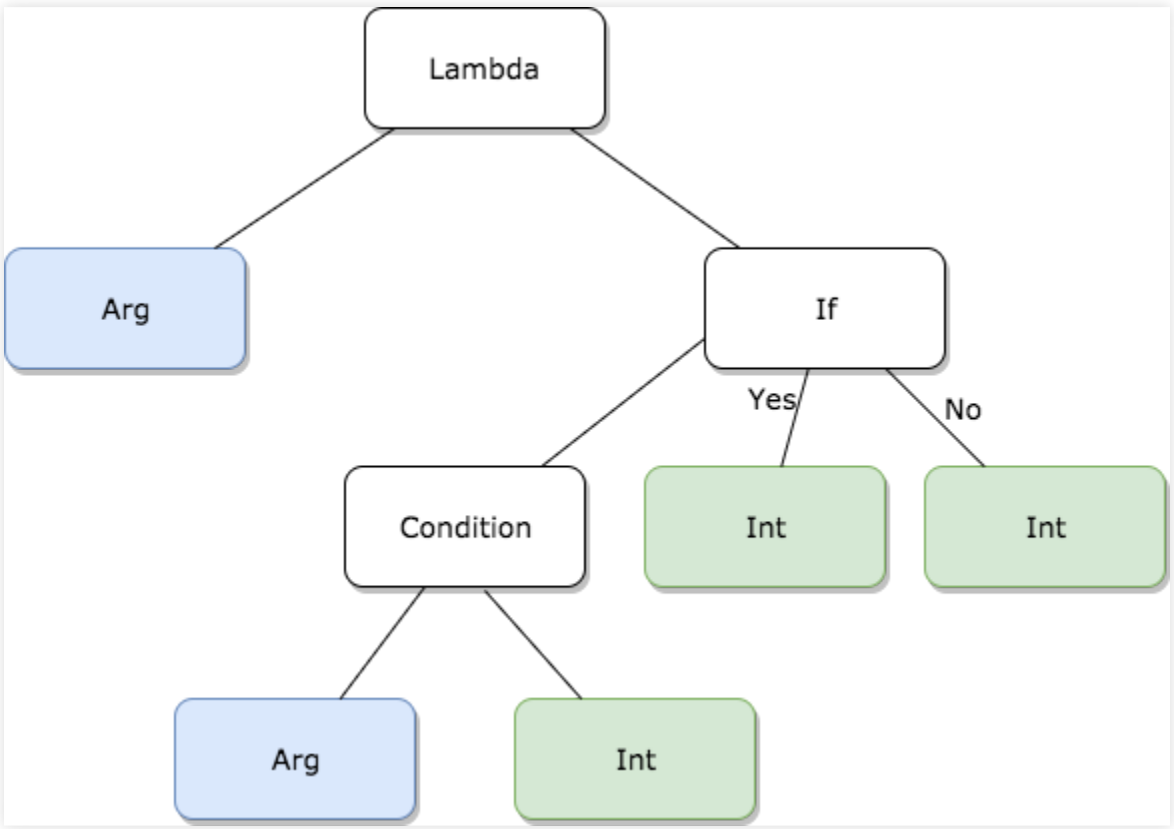
> *If the need for pervasive modifications manifests itself, we can take this as a sign that there is a new concept waiting to be discovered.*
>
> *— [Programming Paradigms for Dummies: What Every Programmer Should Know - Peter Van Roy]*

TYPE INFERENCE

```
1: let myfn bla =
2:         if bla = 0 then 0 else 42
```

```
1: // int -> int
2: let myfn bla =
3:         if bla = 0 then 0 else 42
```

```
let simpleFuncAst = LAMBDA(ARG "bla",
                        IF(EQUAL(
                            ARG("bla"), INT(0)),
                            INT(0),
                            INT(42)
                        ))
```


Lambda
Arg
If
Yes
No
Condition
Int
Int
Arg
Int

```
1:  (Unbound "🐱_1" === Unbound "🐱_2")
2:  (Unbound "🐱_2" === Bound "INT")
3:  (Unbound "🐱_3" === Bound "BOOL")
4:  (Unbound "🐱_4" === Bound "INT")
5:  (Unbound "🐱_4" === Unbound "🐱_6")
6:  (Unbound "🐱_5" === Bound "INT")
7:  (Unbound "🐱_5"=== Unbound "🐱_6")
8:  (Unbound "🐱_7" === Unbound "🐱_1" -> Unbound "🐱_6")
```

RESULTS

[("😺2", "INT"); ("😺3", "BOOL"); ("😺4", "INT"); ("😺5", "INT"); ("😺1", "INT"); ("😺6", "INT"); ("😺_7", "INT->INT")]

ASI64

Why write 6502 assembly when you can inline it in Racket?

https://github.com/pezipink/asi64

```
1: (define (clear-screen start character)
2:   {      ldx @0
3:          lda @character
4:  :loop   (for ([i '(0 1 2 3)])
5:             {sta (+ start (* i $100)) x})
6:          dex
7:          bne loop-   })
```

```
1000: A9 38       LDA #$38
1002: 8D 18 D0    STA $D018
1005: A9 00       LDA #$00
1007: 8D 21 D0    STA $D021
100A: A9 00       LDA #$00
100C: 8D 20 D0    STA $D020
100F: A2 00       LDX #$00
1011: A9 01       LDA #$01
1013: 9D 00 D8    STA $D800,X
1016: 9D 00 D9    STA $D900,X
1019: 9D 00 DA    STA $DA00,X
101C: 9D 00 DB    STA $DB00,X
101F: CA          DEX
1020: D0 F1       BNE $1013
```

A PARADIGM SHIFTS

"a proliferation of compelling articulations, the willingness to try anything, the expression of explicit discontent, the recourse to philosophy and to debate over fundamentals"

— Kuhn, Thomas S.. The Structure of Scientific Revolutions.

Many langauges adding features generally associated with functional programming:

- lambdas
- functional data structures
- pattern matching, etc

C++, Java, C#

The decision to reject one paradigm is always simultaneously the decision to accept another, and the judgment leading to that decision involves the comparison of both paradigms with nature and with each other.

— *Kuhn, Thomas S.. The Structure of Scientific Revolutions.*

It is, I think, particularly in periods of acknowledged crisis that scientists have turned to philosophical analysis as a device for unlocking the riddles of their field. Scientists have not generally needed or wanted to be philosophers.

— *Kuhn, Thomas S.. The Structure of Scientific Revolutions.*

"... two scientific schools disagree about what is a problem and what a solution, they will inevitably talk through each other when debating the relative merits of their respective paradigms."

— Kuhn, Thomas S.. The Structure of Scientific Revolutions.

"He argued that competing paradigms are "incommensurable": that is to say, there exists no objective way of assessing their relative merits."

— Kuhn, Thomas S.. The Structure of Scientific Revolutions.

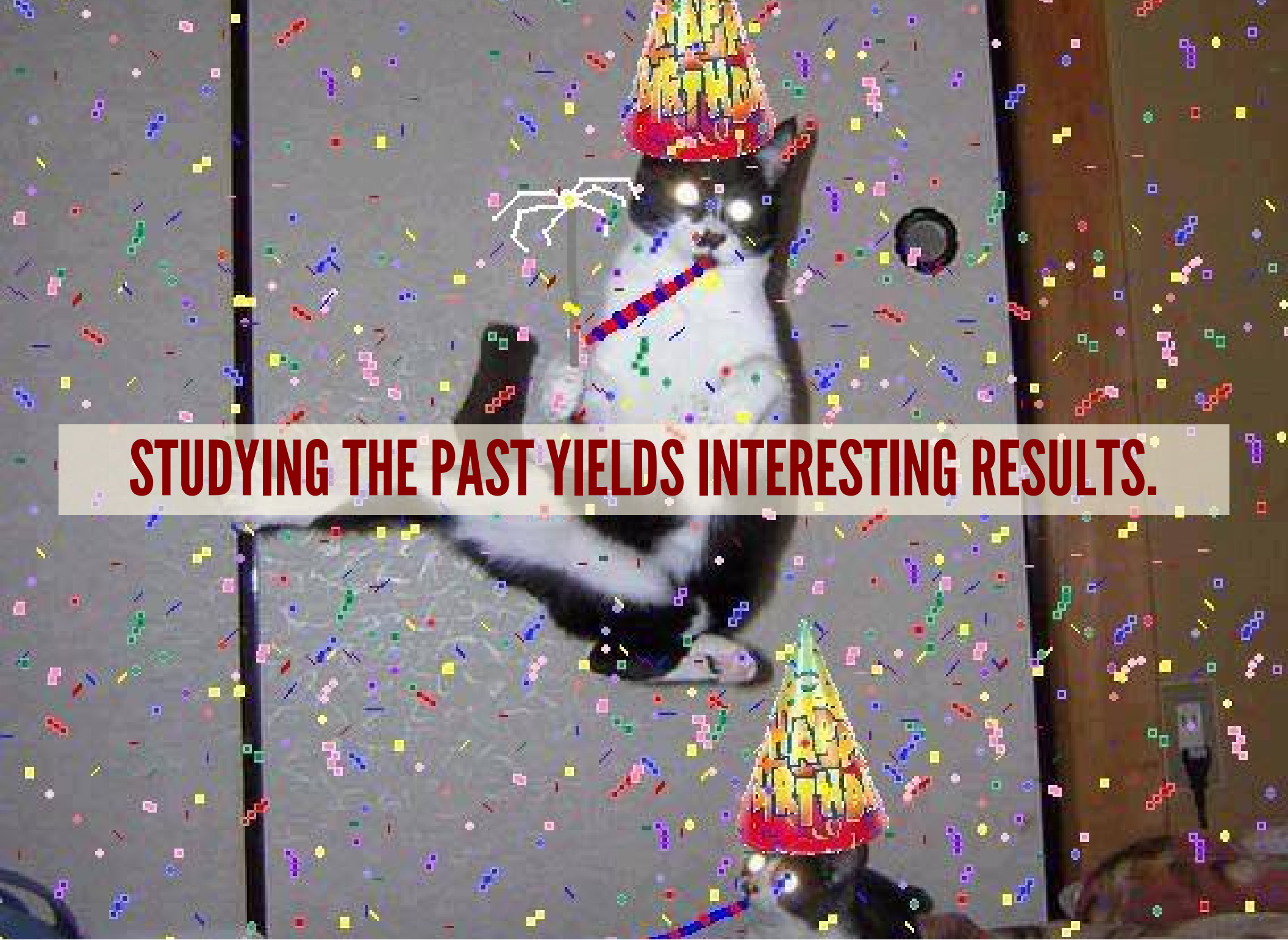
ARE WE SCIENTISTS?

Almost always the people who achieve these fundamental inventions of a new paradigm have been either very young or very new to the field whose paradigm they change .

— *Kuhn, Thomas S.. The Structure of Scientific Revolutions.*


ALL THIS HAS HAPPENED BEFORE AND IT
WILL HAPPEN AGAIN


PEOPLE ARE PART OF THE CONTEXT, MAKE THEM PART OF YOUR CONTEXT


STUDYING THE PAST YIELDS INTERESTING RESULTS.


CHANGING BELIEFS IS A PERSONAL JOURNEY.

THANKS TO:

ROSS MCKINLAY

CHRIS MEIKLEJOHN

EDWIN BRADY

JUAN MANUEL SERRANO

TOMAS PETRICEK

AND OTHERS

THANK YOU

ANDREA MAGNORSKY

@SILVERSPOON

SOURCES | REFERENCES

PAPERS

- Programming Paradigms for Dummies: What Every Programmer Should Know - Peter Van Roy
- The paradigms of programming
- The next 700 programming languages by Peter Landin
- Why Functional Programming Matters by John Hughes
- Joe Armstrong Thesis

ARTICLES, POSTS, VIDEOS

- A punchcard ate my programme by Walt Mankowski
- Clojure spec
- Lenses in F#
- F# Don Syme
- Programming paradigm
- The expression problem

IMAGES

- Cats with hats link