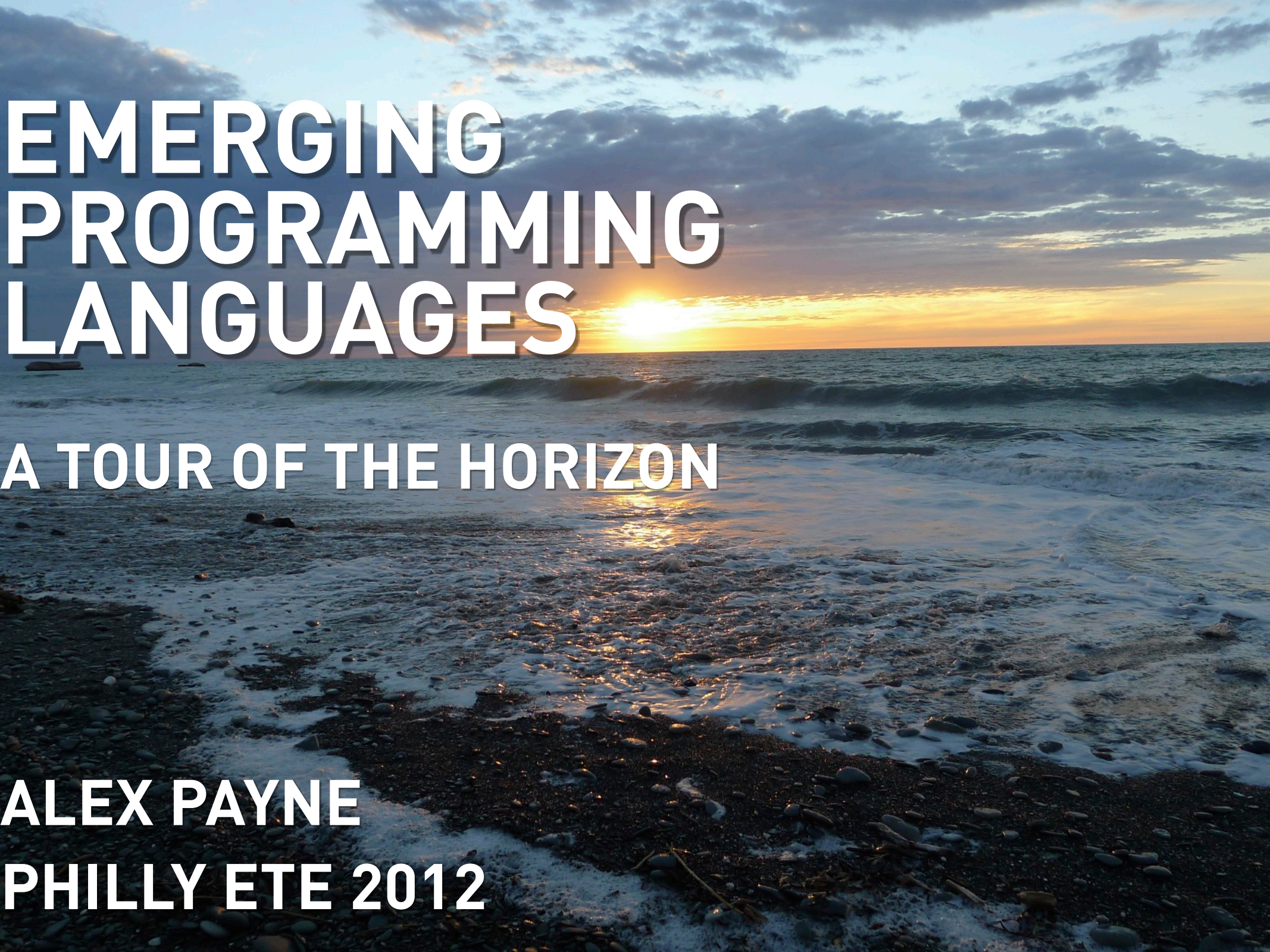
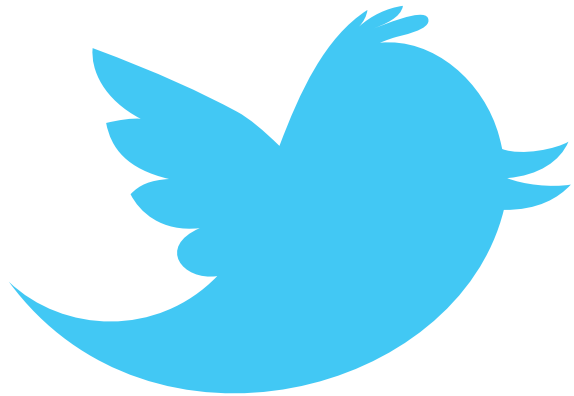
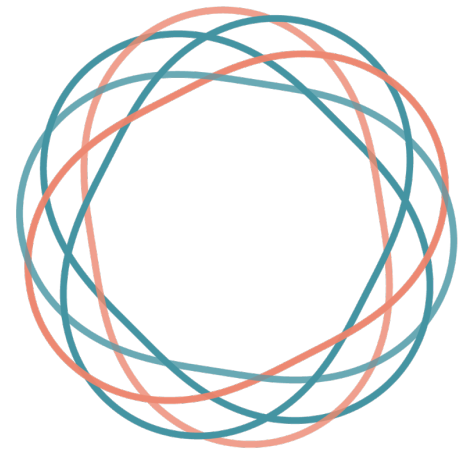

EMERGING
PROGRAMMING
LANGUAGES
A TOUR OF THE HORIZON
ALEX PAYNE
PHILLY ETE 2012



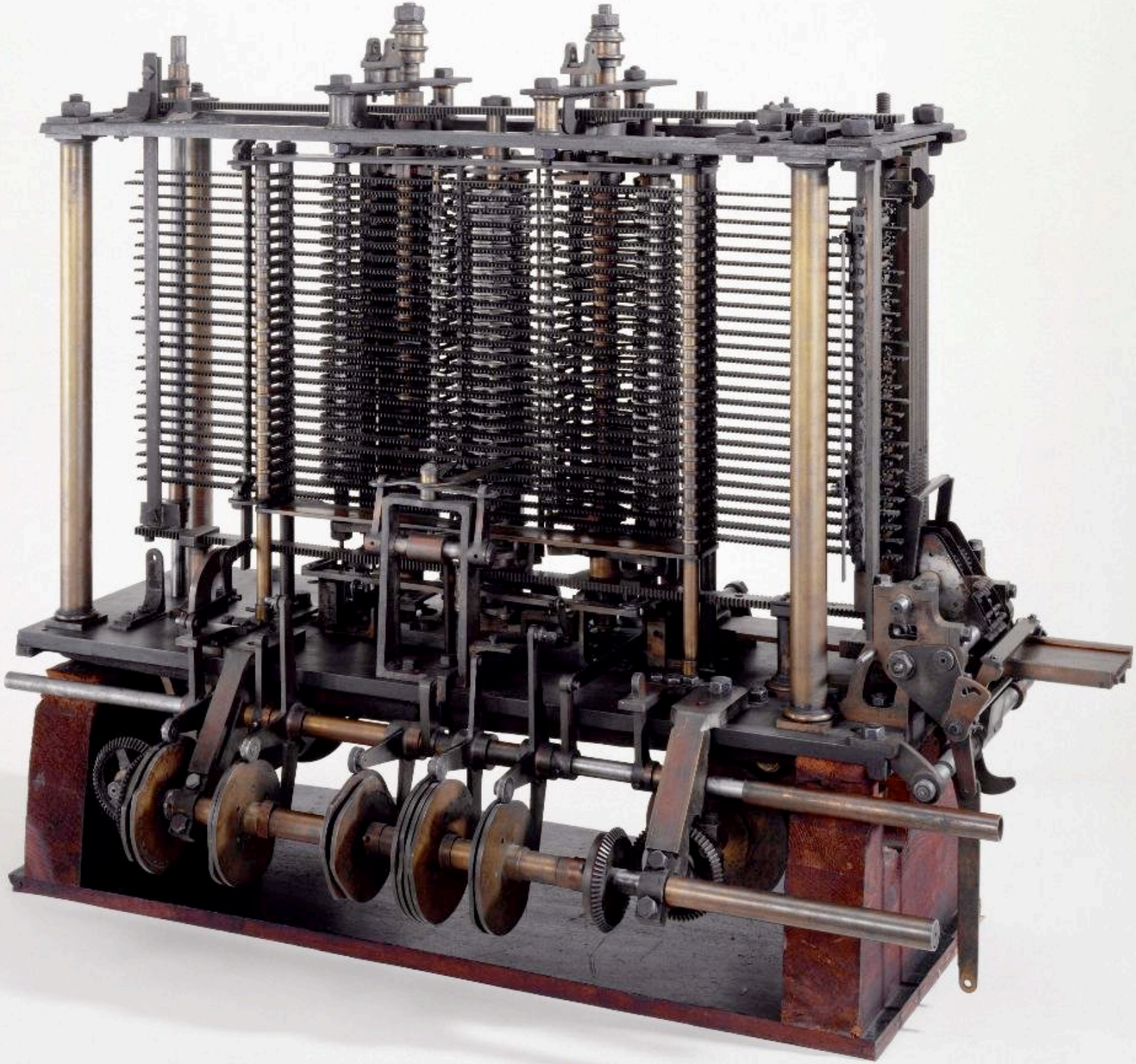

SCALA

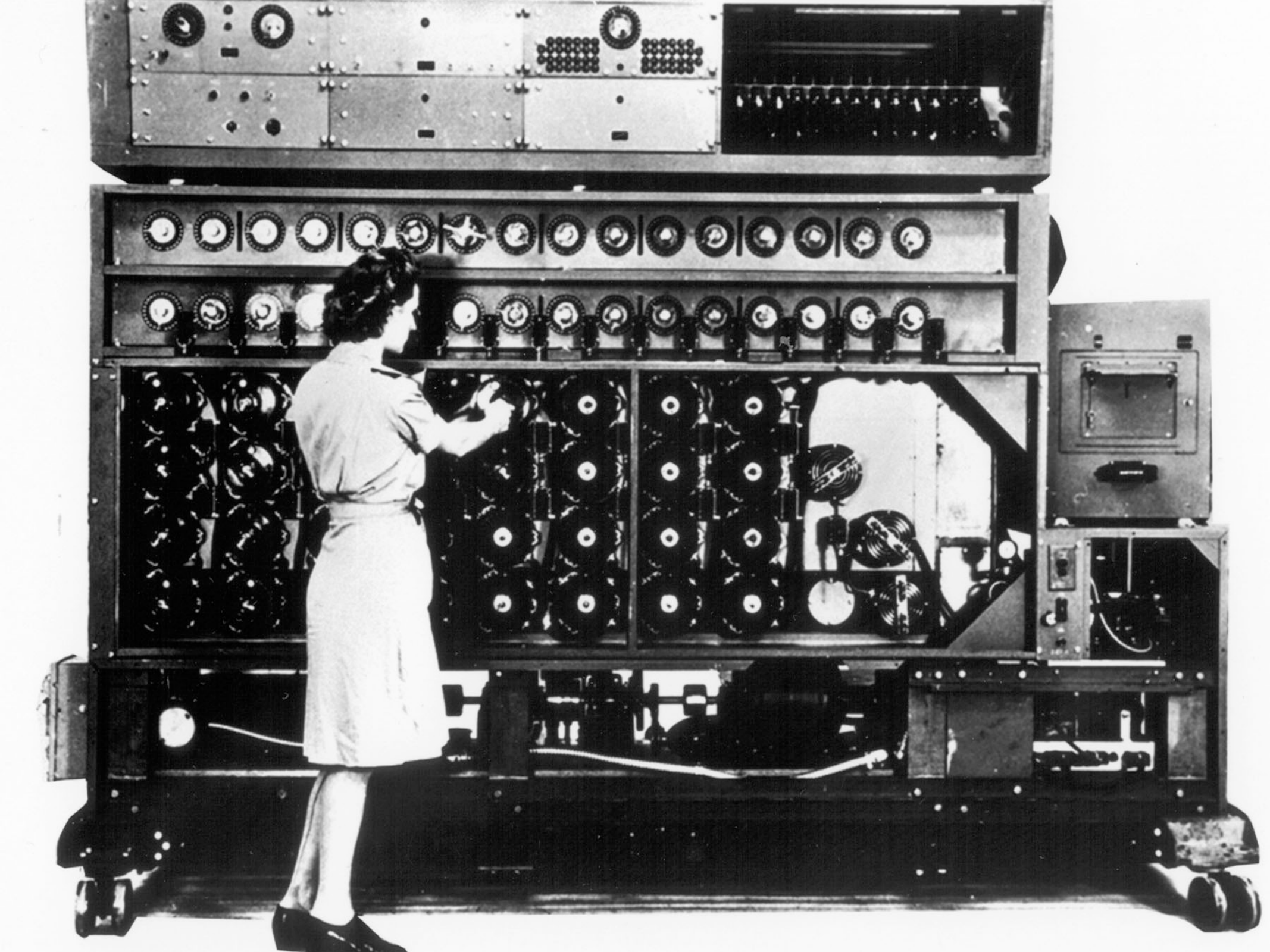


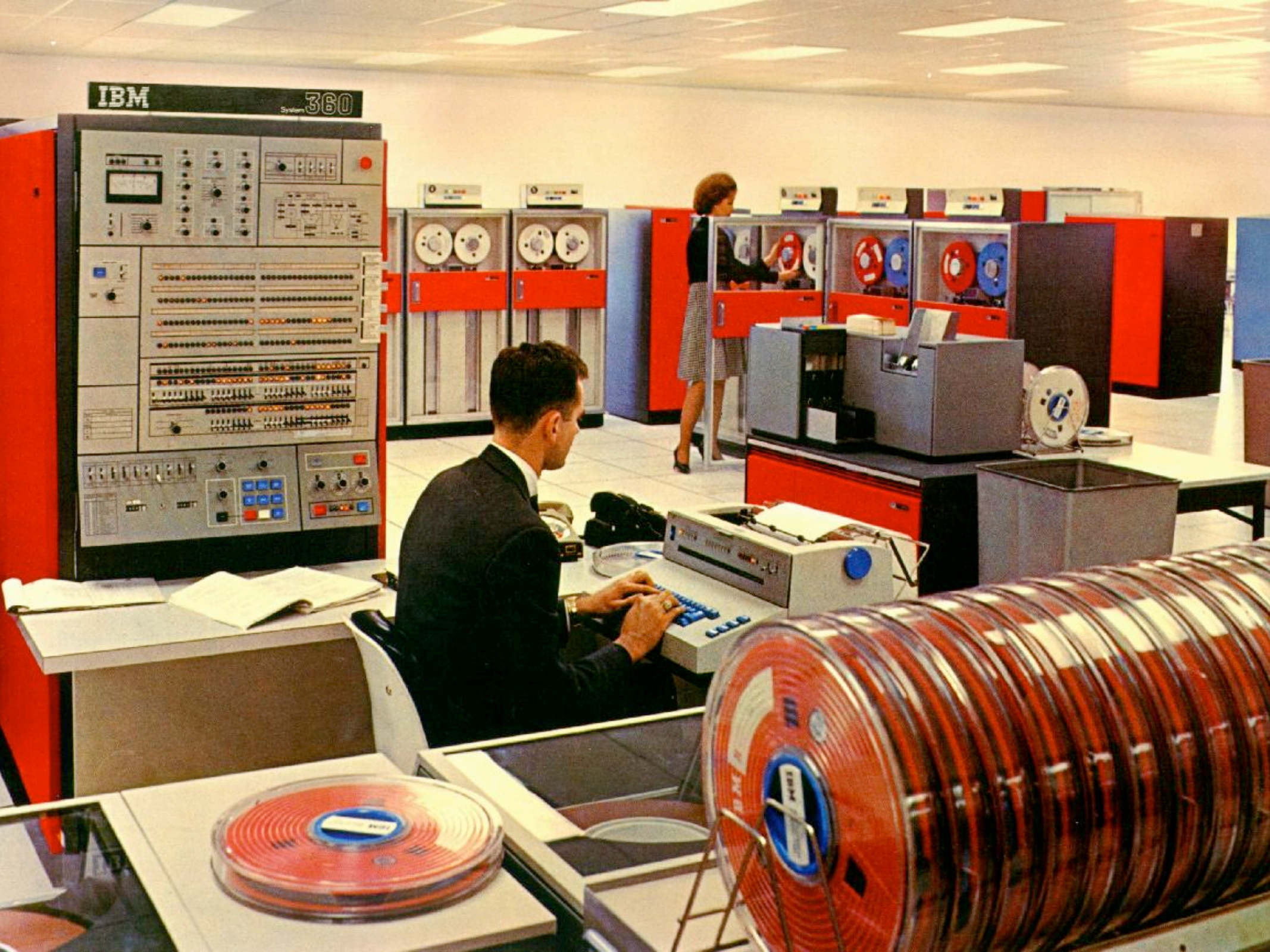

WHY
NEW
LANGUAGES

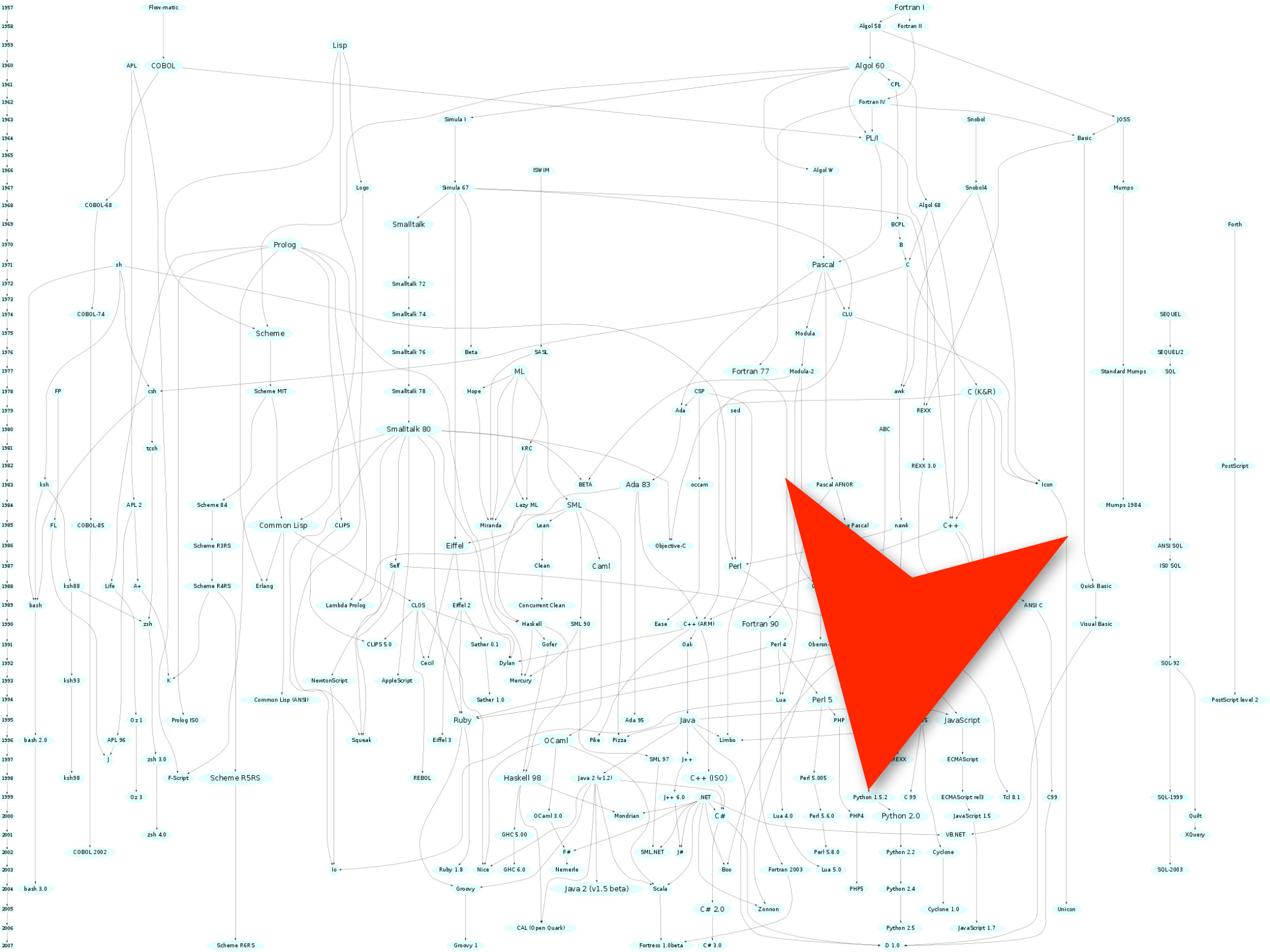

I KNOW THAT WE ALL DECIDED TO GO WITH SCALA LAST WEEK, BUT WE'VE CHANGED OUR MIND AND WE'RE GOING TO CONVERT EVERYTHING TO RUN ON ERLANG INSTEAD.
Emergent Languages - Just Say NO


Why do we need another language?
This is gonna be the
almost feels like Google is so confident on their "moving the
and they kind of ready to loose some of their good old friends. This might sound
political, but here is the truth. If you think of a website with some life on it (dynamism), it is most likely
provided by JavaScript. Obviously, there are other things like Flash, Applets, Silverlight etc. But, they are
much lesser in the grand scheme of "web" in general.
Anyway, I am giving a Stinky to the entire computer industry for its poor performance in standardizing the various languages it uses.
Yet Another Programming Language?
By Kurt Cagle, January 18, 2002
Post a Comment
Could this be the next Java?
Does the world need yet another programming language? After reviewing Curl, I'm still not
sure. Most languages start out as an attempt to simplify overly complex and cruft-filled older
languages. But as they develop, these new kids on the block are beset by many of the same
ills that plague their older siblings.
No thanks. Who needs yet another programming language, when they have LISP?
Can someone please explain to me why this is better than we have now, and what will require it? There is no move whatsoever to move from C++ in the
console world, and there is no mention of Visual Studio integration. In fact, there is no mention of Java–the most powerful and widely used language on
the planet.
Google could have used all this effort to actually integrate it's existing product line with each other and won many, many more hearts & minds. Doing
that would have actually been useful. This, unfortunately, is not.
So again, "Why Do We Need Another Programming Language?"
Does the world need another programming language?
BOO! A new programming language
Why do we need another programming language? Especially if it brings
nothing new, or it's not revolutionary in any way. Maybe because Rodrigo B.
de Oliveira like to play God and believes he's created something while actually
The number of developer years wasted coming up with yet another programming language or syntax is staggering.
I don't really need another programming language, let alone a closed-source once.
Oh great - just what we need - yet another programming language.






IBM
System 360


1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
Flow-matic
Fortran I
Algol 58
Fortran II
Lisp
APL
COBOL
Algol 60
CPL
Fortran IV
Simula I
PL/I
Snobol
JOSS
Basic
Algol W
Snobol4
Mumps
ISWIM
Logo
Simula 67
Algol 68
COBOL-68
Smalltalk
BCPL
B
Forth
Prolog
C
Pascal
sh
Smalltalk 72
Smalltalk 74
CLU
SEQUEL
Scheme
Modula
Smalltalk 76
Beta
SASL
SEQUEL/2
ML
Fortran 77
Modula-2
Standard Mumps
SQL
FP
csh
Scheme MIT
Smalltalk 78
Hope
CSP
awk
C (K&R)
Ada
sed
REXX
ABC
Smalltalk 80
PostScript
tcsh
KRC
REXX 3.0
ksh
BETA
Ada 83
occam
Pascal AFNOR
Icon
APL 2
Scheme 84
Lazy ML
SML
Mumps 1984
FL
COBOL-85
Common Lisp
CLIPS
Miranda
Lean
nawk
C++
Scheme R3RS
Eiffel
Objective-C
ANSI SQL
Self
Clean
Caml
Perl
ISO SQL
ksh88
Life
A+
Scheme R4RS
Erlang
Quick Basic
bash
Lambda Prolog
CLOS
Eiffel 2
Concurrent Clean
ANSI C
zsh
Haskell
SML 90
Ease
C++ (ARM)
Fortran 90
Visual Basic
CLIPS 5.0
Sather 0.1
Gofer
Oak
Perl 4
Oberon-
Cecil
Dylan
SQL-92
ksh93
K
NewtonScript
AppleScript
Mercury
Common Lisp (ANSI)
Sather 1.0
Lua
Perl 5
PostScript level 2
0 z 1
Prolog ISO
Ruby
Ada 95
Java
PHP
JavaScript
bash 2.0
APL 96
Squeak
Eiffel 3
0Caml
Pike
Pizza
Limbo
J
SML 97
J++
REXX
ECMAScript
zsh 3.0
ksh98
F-Script
Scheme R5RS
REBOL
Haskell 98
Java 2 (v1.2)
C++ (ISO)
Perl 5.005
Tcl 8.1
C99
0 z 3
J++ 6.0
.NET
Python 1.5.2
C 99
ECMAScript rel3
SQL-1999
OCaml 3.0
Mondrian
C#
Lua 4.0
Perl 5.6.0
PHP4
Python 2.0
JavaScript 1.5
Quilt
zsh 4.0
GHC 5.00
VB.NET
XQuery
COBOL 2002
F#
Perl 5.8.0
Python 2.2
Cyclone
SML.NET
J#
Io
Ruby 1.8
Nice
GHC 6.0
Nemerle
Boo
Fortran 2003
Lua 5.0
SQL-2003
bash 3.0
Groovy
Java 2 (v1.5 beta)
Scala
PHP5
Python 2.4
C# 2.0
Zonnon
Cyclone 1.0
Unicon
CAL (Open Quark)
Python 2.5
JavaScript 1.7
Scheme R6RS
Groovy 1
Fortress 1.0beta
C# 3.0
D 1.0


WHAT'S
NEXT

~~SCALA~~

~~CLOJURE~~

~~F#~~

~~HASKELL~~

~~ERLANG~~

...

~~R~~

~~GROOVY~~

~~D~~

~~FANTOM~~

~~LUA~~

...

DIFFERENT LANGUAGES FOR DIFFERENT JOBS

JOB:
BETTER
JAVA

KOTLIN

JAVA++ (OR SCALA--?) FROM JETBRAINS

```
fun main(args: Array<String>) {
  for (name in args)
    println("Hello, $name!")
}
```

STATIC • OOP • GENERICS • CLOSURES

GOSU

"A PRAGMATIC LANGUAGE FOR THE JVM"

```
strings = {"this", "otter", "other"}

bigStrings = strings
  .where( \ s -> s.length() > 4 )
  .map( \ s -> s.toUpperCase() )
  .orderBy( \ s -> s)

bigStrings // {"OTHER", "OTTER"}
```

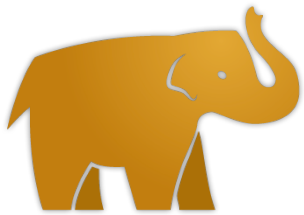
GOSU

STATIC • OOP • GENERICS • CLOSURES

CEYLON

REDHAT'S UPDATED JAVA

```
class Parent(String name) {
    shared String name = name;
    shared class Child(String name) {
        shared String name = outer.name + "/" + name;
        shared Parent parent { return outer; }
    }
}
```



STATIC • OOP • GENERICS • INTERCEPTORS

JOB:
BETTER
JAVASCRIPT

STRATIFIEDJS

"JAVASCRIPT + STRUCTURED CONCURRENCY"

```
var response;

waitfor {
  response = http.get('http://bbc.com');
} or {
  response http.get('http://cnn.com');
}

display(response);
```

CONCURRENT • ASYNCHRONOUS

COFFEESCRIPT

JAVASCRIPT, THE GOOD PARTS

```
Account = (customer, cart) ->
  @customer = customer
  @cart = cart

  $('.shopping_cart').bind('click', (e) =>
    @customer.purchase @cart
  )
```

SOURCE-TO-SOURCE TRANSLATION

OBJECTIVE-J

OBJECTIVE-C → JAVASCRIPT

```
@import <Foundation/CPString.j>

@implementation CPString (Reversing)

- (CPString)reverse
{
    var reversedString = "",
        index = [self length];

    while(index--)
        reversedString += [self characterAtIndex:index];

    return reversedString;
}

@end
```

DYNAMIC • OOP • CATEGORIES

CLOJURESCRIPT

CLOJURE → JAVASCRIPT

```
(defmethod effect :swipe [element m]
  (let [{:keys [start end time accel]} (standardize element m)]
    (goog.fx.dom.Swipe. element
                        (apply array start)
                        (apply array end)
                        time
                        accel)))
```

DYNAMIC • FUNCTIONAL • LISP

DART

GOOGLE'S JAVASCRIPT REPLACEMENT

```
class Point {
  num x, y;
  Point(num this.x, num this.y);
  Point scale(num factor) => new Point(x*factor, y*factor);
  num distance() => Math.sqrt(x*x + y*y);
}

void main() {
  Point a = new Point(2,3).scale(10);
  print(a.distance());
}
```

CLASSES • GENERICS • OPTIONAL TYPING

ROY

FUNCTIONAL CODE INTO JAVASCRIPT

```
let traceMonad = {
  return: \x ->
    console.log "Return:" x
    x
  bind: \x f ->
    console.log "Binding:" x
    f x
}

console.log (do traceMonad
  w <- 1
  let x = 2
  y <- 3
  z <- 4
  return w + x + y + z
)
```

MONADS • TYPE INFERENCE • PATTERN MATCHING

JOB:
WEB
DEVELOPMENT

OPA

"A UNIFIED PLATFORM FOR WEB APPS"

```
function user_update(message x) {
    line = <div class="row line">
              <div class="span1 columns userpic" />
              <div class="span2 columns user">{x.author}:</div>
              <div class="span13 columns message">{x.text}</div>
           </div>;
    #conversation =+ line;
    Dom.scroll_to_bottom(#conversation);
}
```

SOURCE-TO-SOURCE • OOP • METACLASSES

UR/WEB

"A DSL FOR WEB APPLICATIONS"

```
and imHere () =
    user0 <- getCookie username;
    case user0 of
        None => return <xml>You don't have a cookie set!</xml>
      | Some user =>
        dml (DELETE FROM lastVisit WHERE User = {[user]});
        dml (INSERT INTO lastVisit (User, When) VALUES ({[user]}, CURRENT_TIMESTAMP));
        main ()
```

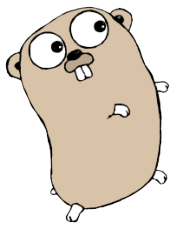
FUNCTIONAL • STATIC • METAPROGRAMMING

JOB:
SYSTEMS
PROGRAMMING

GO

REVENGE OF THE 1970s!

```
var sem = make(chan int, MaxOutstanding)

func handle(r *Request) {
    sem <- 1    // Wait for active queue to drain.
    process(r)  // May take a long time.
    <-sem       // Done; enable next request to run.
}

func Serve(queue chan *Request) {
    for {
        req := <-queue
        go handle(req)  // Don't wait for handle to finish.
    }
}
```



COMPILED • CONCURRENT • GARBAGE COLLECTED

RUST

"SAFE, CONCURRENT, PRACTICAL"

```
fn stringifier(from_parent: comm::port<uint>,
               to_parent: comm::chan<str>) {
    let mut value: uint;

    do {
        value = comm::recv(from_parent);
        comm::send(to_parent, uint::to_str(value, 10u));
    } while value != 0u;
}
```

COMPILED • OOP • FUNCTIONAL • STATIC

OOC

C + OBJECTS + MORE, COMPILING TO C99

```
main: func {
   number := 42 // alloc an int on the stack
   printf("number is %d\n", number)
   add(number&, 3)
   printf("number is now %d\n", number)
}

add: func (ptr: Int@, value: Int) {
   ptr += value
}
```

SOURCE-TO-SOURCE • OOP • METACLASSES

JOB:
DYNAMIC PROGRAMMING

FANCY

A DYNAMIC LANGUAGE ON RUBINIUS VM

```
require: "sinatra.fy"

configure: 'production with: {
  disable: 'show_errors
  enable: 'logging
}

before: {
  "incoming request: #{request inspect}" println
}

def page: text {
  """
  <h1>#{text}</h1>
  """
}

get: "/:p" do: |param| {
  page: "Fancy web page: #{param}"
}
```

DYNAMIC • OOP • ACTORS

SLATE

A MODERN SMALLTALK

```
s@(Sequence traits) isPalindrome
[
  s isEmpty
    ifTrue: [True]
    ifFalse: [(s first = s last) /\ [(s sliceFrom: 1 to: s indexLast - 1) isPalindrome]]
].
```

DYNAMIC • PROTOTYPES • STREAMS • MACROS

ELIXIR

"MODERN PROGRAMMING FOR THE ERLANG VM"

```
defmodule Hygiene do
  defmacro interference do
    quote do: var!(a) = 1
  end
end

defmodule HygieneTest do
  def go do
    require Hygiene
    a = 13
    Hygiene.interference
    a
  end
end
```

DYNAMIC • PROTOCOLS • RECORDS • MACROS

JOB:
TECHNICAL COMPUTING

FRINK

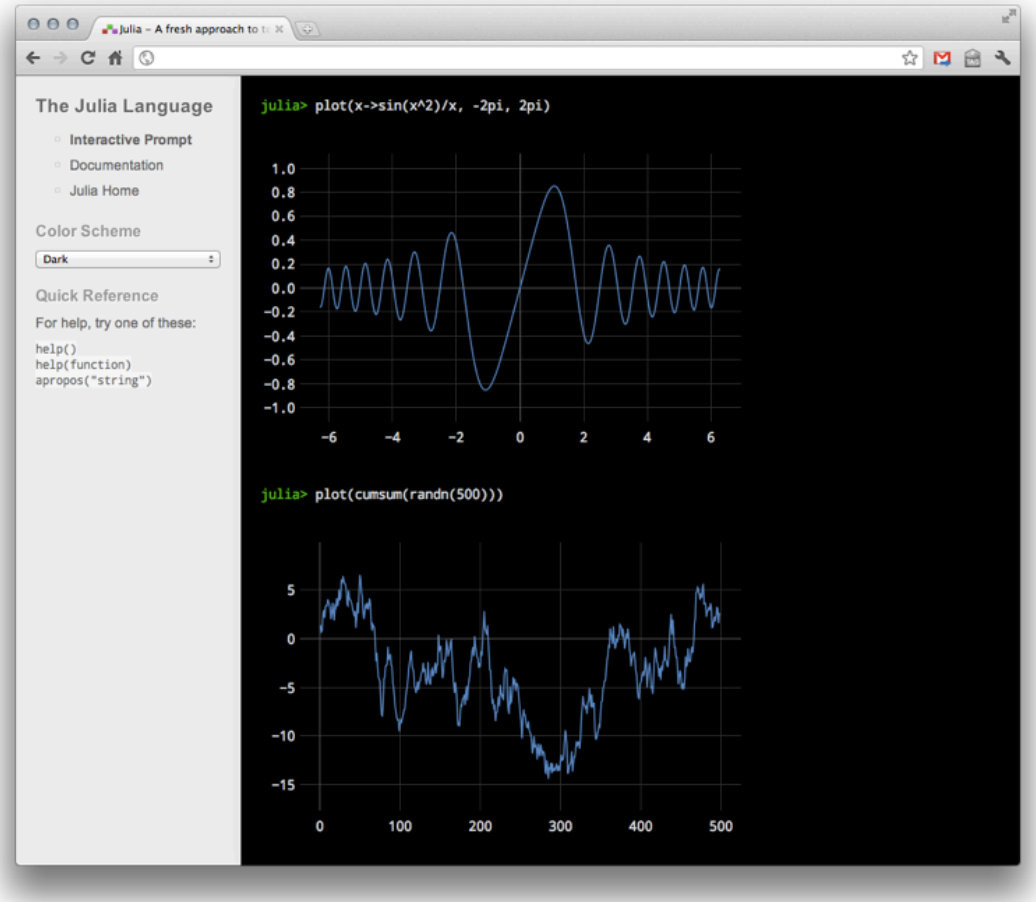
"MAKE PHYSICAL CALCULATIONS SIMPLE"

```
earthpower = sunpower / (4 pi sundist^2)

chargerate = earthpower 12 ft^2
chargerate -> watts
1530.1602

2 ton 7 feet gravity / chargerate -> sec
24.80975

(225 + 135) pounds 15000 feet gravity / chargerate -> minutes
59.809235
```

EMBEDDABLE • OOP • UNICODE

JULIA

"HIGH-LEVEL, HIGH-PERFORMANCE TECHNICAL COMPUTING"



DYNAMIC • COMPILED • PARALLEL

FAUST

A LANGUAGE FOR DSP AND SYNTHESIS

```
import["math.lib"];

delay[n,d,x] = x@(int(d)&(n=1));
msec = SR/1000.0;

duration = hslider("millisecond", 0, 0, 1000, 0.10) * msec : int;
feedback = [hslider("feedback", 0, 0, 100, 0.1) / 100.0];

echo = vgroup("echo", +-(delay(SR, duration) * feedback));

process = vgroup["stereo echo", [echo, echo]);
```

FUNCTIONAL • SOURCE-TO-SOURCE COMPILED

JOB: QUERYING DATA

BANDICOOT

A LANGUAGE FOR SET ALGEBRA

```
# selects fiction books from the input
fn Fiction(b: Books): Books
{
    return b select(genre == "Fiction");
}

# calculates an average price of fiction books
fn FictionPrice(b: Books): rel {avgPrice: real}
{
    # use of a temporary variable and a chained statement
    res := b select(genre == "Fiction")
             summary(avgPrice = avg(price, 0.0));

    return res;
}
```

[/Band|c00t]

RELATIONAL • PERSISTENCY • DISTRIBUTED

QUIRREL

A LANGUAGE FOR ANALYTICS

```
clicks := load(//clicks)
views  := load(//views)
clickthroughRate('page) :=
  {page: 'page, ctr: count(clicks where clicks.pageId = 'page) /
                     count(views where views.pageId = 'page)}
clickthroughRate
```

```
[{"page":"page-4","ctr":0.55555555555555555555555555555556},
 {"page":"page-1","ctr":2.07692307692307692307692307692307}, ...]
```

DECLARATIVE • FUNCTIONAL • COMPOSABLE

JOB: MAKE YOU THINK

WHEELER

"DIFFERENT"

```
transition (pattern print (string)) (action STDOUT)
print "Hello, world!"
// Hello World
"Hello, world!" print
// Hello World

transition (pattern fast car) (action print "ZOOM ZOOM")
fast car
// ZOOM ZOOM
car fast
// ZOOM ZOOM
```

NO VARIABLES • NO FUNCTIONS • NO OBJECTS

KODU

PROGRAMMING FOR KIDS, ON XBOX

VISUAL • INTERACTIVE • ITERATIVE

WHEW!

JOBS:

BETTER JAVA

BETTER JAVASCRIPT

WEB DEVELOPMENT

SYSTEMS PROGRAMMING

DYNAMIC PROGRAMMING

TECHNICAL COMPUTING

QUERYING DATA

MAKING YOU THINK

...

PLATFORMS:

JVM

CLR

JAVASCRIPT (V8, ETC.)

RUBINIUS

LLVM

ERLANG VM

PHP

XBOX

...

EXPLORE.
EXPERIMENT.
COMMIT.
LOOP.


FIN.
EMERGINGLANGS.COM
@EMERGINGLANGS