Thriving in a polyglot world
Genesis 11:1—9 (KJV)

11 And the whole earth was of one language, and of one speech.

2 And it came to pass, as they journeyed from the east, that they found a plain in the land of Shinar; and they dwelt there.

3 And they said one to another, Go to, let us make brick, and burn them thoroughly. And they had brick for stone, and slime had they for mortar.

4 And they said, Go to, let us build us a city and a tower, whose top may reach unto heaven; and let us make us a name, lest we be scattered abroad upon the face of the whole earth.

5 And the Lord came down to see the city and the tower, which the children of men builded.

6 And the Lord said, Behold, the people is one, and they have all one language; and this they begin to do: and now nothing will be restrained from them, which they have imagined to do.

7 Go to, let us go down, and there confound their language, that they may not understand one another's speech.

8 So the Lord scattered them abroad from thence upon the face of all the earth: and they left off to build the city.

9 Therefore is the name of it called Babel; because the Lord did there confound the language of all the earth: and from thence did the Lord scatter them abroad upon the face of all the earth.
1. የጉት የር ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ ግ }
And the whole earth was of one language, and of one speech.
And they said,
Go to, let us build us a city, and a tower, whose top may reach unto heaven;
and let us make us a name, lest we be scattered abroad upon the face of the whole earth.
And the LORD said, Behold, the people is one, and they have all one language; and this they begin to do: and now nothing will be restrained from them, which they have imagined to do.
Go to, let us go down, and there confound their language, that they may not understand one another's speech.
So the LORD scattered them abroad from thence upon the face of all the earth: and they left off to build the city.
Therefore is the name of it called Babel; because the LORD did there confound the language of all the earth: and from thence did the LORD scatter them abroad upon the face of all the earth.
Hiro Protagonist
Y.T.
Enki / Asherah

Sumerian/Glossolalia

machine code for the brain
I speak Spanish to God,
I speak Spanish to God,
Italian to Women,
I speak Spanish to God,
Italian to Women,
French to Men,
I speak Spanish to God, Italian to Women, French to Men, and German to my Horse.
This is, basically, the Sapir-Whorf hypothesis
Sapir-Whorf Hypothesis
“Linguistic Relativity”
Linguistic relativity

“The principle of linguistic relativity holds that the structure of a language affects the ways in which its respective speakers conceptualize their world, i.e. their world view, or otherwise influences their cognitive processes”

http://en.wikipedia.org/wiki/Linguistic_relativity
"The purpose of Newspeak was not only to provide a medium of expression for the world-view and mental habits proper to the ..., but to make all other modes of thought impossible. Its vocabulary was so constructed as to give exact and often very subtle expression to every meaning that a Party member could properly wish to express, while excluding all other meaning and also the possibility of arriving at them by indirect methods."
I speak Spanish to God, Italian to Women, French to Men, and German to my Horse.
ORLANDO LASSO FIAMINGO COMPOSIT. NASCU L'ANNO 1520. E, MORI NEL 1594 FATO CAVALIERE DA MASSIMIL. IL IMPER.
Orlando de Lassos:
Born in Belgium
Moved to Italy aged 12
Traveled to France & England
Back to Holland
Court Composer in Bavaria
Wrote music in Italian, French, German, Dutch, Latin
Being monolingual is not the norm in human history
Being monolingual is not the norm in modern society
[citation, please!]
174,000,000 people
521 languages
Benefits of multilingualism
Delving the onset of Alzheimer disease
Bilingualism as a form of cognitive reserve

Fergus I.M. Craik, PhD, Ellen Bialystok, PhD, and Morris Freedman, MD

Author information ► Copyright and License information ►

This article has been cited by other articles in PMC.

Abstract

Objectives: There is strong epidemiologic evidence to suggest that older adults who maintain an active lifestyle in terms of social, mental, and physical engagement are protected to some degree against the onset of dementia. Such factors are said to contribute to cognitive reserve, which acts to compensate for the accumulation of amyloid and other brain pathologies. We present evidence that lifelong bilingualism is a further factor contributing to cognitive reserve.

Methods: Data were collected from 211 consecutive patients diagnosed with probable Alzheimer disease (AD) and 211 age-matched healthy controls. Language abilities were assessed using five language tests.

http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3033609/
“There is strong epidemiologic evidence to suggest that older adults who maintain an active lifestyle in terms of social, mental, and physical engagement are protected to some degree against the onset of dementia. Such factors are said to contribute to cognitive reserve, which acts to compensate for the accumulation of amyloid and other brain pathologies. **We present evidence that lifelong bilingualism is a further factor contributing to cognitive reserve.**”
Learn More Languages!
It is practically impossible to teach good programming to students that have had a prior exposure to BASIC: as potential programmers they are mentally mutilated beyond hope of regeneration.
Languages affect cognition
Languages affect cognition

Across the board, object gender influenced the participants’ judgments. For example, the word “key” is masculine in German and feminine in Spanish. German speakers in the study tended to describe keys as hard, heavy, jagged, metal, and useful. Spanish speakers, on the other hand, used words such as golden, intricate, little, lovely, and tiny when describing keys.
Languages affect cognition

Small, unnoticed features of language can influence our thoughts, sometimes in big ways. Knowing that, imagine how else language, culture, and society might affect our thoughts, feelings, and behaviors.
"A programming language is a tool that has a profound influence on our thinking habits"

- Edsger Dijkstra
org 100h

; set video mode
mov ax, 3 ; text mode 80x25, 16 colors, 8 pages (ah=0, al=3)
int 10h ; do it!

; cancel blinking and enable all 16 colors:
mov ax, 1003h
mov bx, 0
int 10h

; set segment register:
mov ax, 0b8000h
mov ds, ax

; print "hello world"
; first byte is ascii code, second byte is color code.
mov [02h], 'H'
mov [04h], 'e'
mov [06h], 'l'
mov [08h], 'o'
mov [0ah], 'C'
mov [0ch], ','
mov [0eh], 'W'
mov [10h], 'o'
mov [12h], 'r'
mov [14h], 'l'
mov [16h], 'd'
mov [18h], '!'

; color all characters:
mov cx, 12 ; number of characters.
mov di, 03h ; start from byte after 'h'

; c: mov [di], 11101100b ; light red(110) on yellow(1110)
add di, 2 ; skip over next ascii code in vga memory.
loop c

; wait for any key press:
mov ah, 0
int 16h

ret
/**
 * The HelloWorldApp class implements an application that
 * simply prints "Hello World!" to standard output.
 */

class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!"); // Display the string.
    }
}
Python 2.7.6 (default, Mar 22 2014, 22:59:56)
Type "copyright", "credits" or "license" for more information.

IPython 1.2.1 -- An enhanced Interactive Python.
?
   -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help     -> Python's own help system.
object?  -> Details about 'object', use 'object??' for extra details

In [1]: print "Hello, World!"
Hello, World!
You can't be a polyglot and make code that looks the same in all your languages. Approach each lang idiomatically, and embrace its strengths

Bryan Liles (@bryanel)
from abc import ABCMeta, abstractmethod, abstractproperty

class IWidget(object):
    __metaclass__ = ABCMeta

    @abstractproperty
def thing(self):
        '''get / set thing'''

    @abstractmethod
def foobar(self):
        '''is it better to foo, or to bar?'''

class AbstractWidgetFactoryManager(object):
    ....
from abc import ABCMeta, abstractmethod, abstractproperty

class IWidget(object):
    __metaclass__ = ABCMeta

    @abstractproperty
    def thing(self):
        '''get / set thing'''

    @abstractmethod
    def foobar(self):
        '''is it better to foo, or to bar?'''

class AbstractWidgetFactoryManager(object):
    ....
When in Rome...

Write C++ in C++,
Java in Java,
Python in Python.
The Start of Polyglot Programming
(okay, Unix has always been slightly polyglot, with shells, sed, awk) – but even on Unix it didn't really kick off until Perl(yuck) came along
In the Windows World, we were stuck with C. Or C++. And huge great switch statements.
Of course, in theory, you could create dlls in different languages
- calling conventions
- object lifetimes
- even between compiler versions it could be tricky
Then came
Visual Basic
And COM, and OCXes, and Excel Addins
Don Box Book:
COM: A Really Elegant Concept
COM: Horrid in Practice!
Java
SQL
shell
Alternate Hard And Soft Layers

PatternForm Version:

Aliases: Scripting System, Interpreter, Extensible Middle Layer

Problem: The mapping between the primitives available and the driver that controls them is too static. Combinations of the primitives are

Context: The primitive elements of the system are used in a limited but effective way. The applications serve existing needs using well-defined interfaces, but the new needs into the system depends in part on the changes to the workflow that supporting the new features produces.

Forces: A context for rapid experimentation and novel functionality requires new infrastructure. New data types must be created to accommodate the new requirements while markedly reducing efficiency of the existing system functions.

Solution: Build an extensible interpreter environment that makes all the primitive elements of the existing system available via universal data type adaptive interfaces.

Resulting Context: A single generic driver system provides an entry point for a data type that relates the combinations and relationships between the primitive elements, providing a much larger context has been created at the same time.
<table>
<thead>
<tr>
<th>Hard</th>
<th>Difficult to change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Soft</td>
<td>Flexible, but can be slow</td>
</tr>
<tr>
<td>Hard</td>
<td>Optimised for speed</td>
</tr>
<tr>
<td>UI</td>
<td>Java</td>
</tr>
<tr>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>Business logic</td>
<td>Python</td>
</tr>
<tr>
<td>Numerics</td>
<td>C</td>
</tr>
<tr>
<td>User Programs</td>
<td>Excel Functions</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Excel</td>
<td>Excel</td>
</tr>
<tr>
<td>Business logic</td>
<td>Python</td>
</tr>
<tr>
<td>Numerics</td>
<td>C/Numpy</td>
</tr>
</tbody>
</table>
Python C API example

#include <Python.h>
#include <math.h>
Python C API example

```c
#include <Python.h>
#include <math.h>

static PyObject* cos_func(PyObject* self, PyObject* args)
{
    double value;
    double answer;
```
#include <Python.h>
#include <math.h>

static PyObject* cos_func(PyObject* self, PyObject* args)
{
    double value;
    double answer;
    /* parse the input, from python float to c double */
    if (!PyArg_ParseTuple(args, "d", &value))
        return NULL;
    /* if the above function returns -1, an appropriate Python exception will
    * have been set, and the function simply returns NULL
    */
#include <Python.h>
#include <math.h>

static PyObject* cos_func(PyObject* self, PyObject* args)
{
    double value;
    double answer;
    /* parse the input, from python float to c double */
    if (!PyArg_ParseTuple(args, "d", &value))
        return NULL;
    /* if the above function returns -1, an appropriate Python exception will
    * have been set, and the function simply returns NULL
    */

    /* construct the output from cos, from c double to python float */
    return Py_BuildValue("f", answer);
}
/* define functions in module */
static PyMethodDef CosMethods[] =
{
    {"cos_func", cos_func, METH_VARARGS, "evaluate the cosine"},
    {NULL, NULL, 0, NULL}
};

/* module initialization */
PyMODINIT_FUNC
initcos_module(void)
{
    (void) Py_InitModule("cos_module", CosMethods);
}
```c
#include <Python.h>
#include <math.h>

static PyObject* cos_func(PyObject* self, PyObject* args)
{
    double value;
    double answer;
    /* parse the input, from python float to c double */
    if (!PyArg_ParseTuple(args, "d", &value))
        return NULL;
    /* if the above function returns -1, an appropriate Python exception will */
    /* have been set, and the function simply returns NULL */
    
    /* call cos from libm */
    answer = cos(value);

    /* construct the output from cos, from c double to python float */
    return Py_BuildValue("f", answer);
}
```
Python ctypes

```python
>>> from ctypes import *
>>> print windll.kernel32
<WinDLL 'kernel32', handle ... at ...>
>>> print cdll.msvcrt
<CDLL 'msvcrt', handle ... at ...>
>>> libc = cdll.msvcrt

>>> print libc.time(None)
1150640792
>>> print hex(windll.kernel32.GetModuleHandleA(None))
0x1d000000
>>>```
Swig python example

/* example.i */
#include example
{%
/* Put header files here or function declarations like below */
extern double My_variable;
extern int fact(int n);
extern int my_mod(int x, int y);
extern char *get_time();
%

extern double My_variable;
extern int fact(int n);
extern int my_mod(int x, int y);
extern char *get_time();
Swig python example

/* example.i */
module example
{
    /* Put header files here or function declarations like below */
    extern double My_variable;
    extern int fact(int n);
    extern int my_mod(int x, int y);
    extern char *get_time();
}

extern double My_variable;
extern int fact(int n);
extern int my_mod(int x, int y);
extern char *get_time();
Boost python example

#include <iostream>

using namespace std;

void say_hello(const char* name) {
    cout << "Hello " << name << "!\n";
}

#include <boost/python/module.hpp>
#include <boost/python/def.hpp>
using namespace boost::python;

BOOST_PYTHON_MODULE(hello) {
    def("say_hello", say_hello);
}
def fibonacci(n):
    """ calculates the first n numbers in the fibonacci series""
    assert n > 0, "don't be silly, n must be > 0"
    assert n > 2, "surely you can work out the first two yourself?"

    fibo = [1, 1]
    for _ in range(2, n):
        fibo.append(fibo[-2] + fibo[-1])

    return fibo
def test_fibo_function(f):
    def check_equal(lhs, rhs):
        assert lhs == rhs, "%s != %s" % (lhs, rhs)

    check_equal(f(3), [1, 1, 2])
    check_equal(f(8), [1, 1, 2, 3, 5, 8, 13, 21])
    print f.__name__, "passed"
def fibonacci(n):
    ''' calculates the first n numbers in the fibonacci series'''
    assert n > 0, "don't be silly, n must be > 0"
    assert n > 2, "surely you can work out the first two yourself?"

    fibo = [1, 1]
    for _ in range(2, n):
        fibo.append(fibo[-2] + fibo[-1])

    return fibo
from distutils.core import setup
from Cython.Build import cythonize

setup(
    ext_modules = cythonize("cython_fibo.py")
)

python setup.py build_ext --inplace
Cython python example

In [11]: import python_fibo, cython_fibo

In [12]: %timeit python_fibo.fibonacci(100)
10000 loops, best of 3: 23.4 µs per loop

In [13]: %timeit cython_fibo.fibonacci(100)
1000000 loops, best of 3: 9.41 µs per loop
/* Generated by Cython 0.20.1post0 (Debian 0.20.1+git90-g0e6e38e-1ubuntu2) on Thu Apr 16 08:30:31 2015 */

define PY_SSIZE_T_CLEAN

ifndef CYTHON_USE_PYLONG_INTERNALS
  ifdef PYLONG_BITS_IN_DIGIT
    define CYTHON_USE_PYLONG_INTERNALS 0
  else
    include "pyconfig.h"
  endif
endif
endif
endif

#include "Python.h"

ifndef Py_PYTHON_H
  #error Python headers needed to compile C extensions, please install development version of Python.
endif

elif PY_VERSION_HEX < 0x02040000
  #error Cython requires Python 2.4+

"cython_fibo.c" 3380L, 134148C
def fibonacci(int n):
    ''' calculates the first n numbers in the fibonacci series'''
    assert n > 0, "don't be silly, n must be > 0"
    assert n > 2, "surely you can work out the first two yourself?"

    cdef int i
    fibo = [1, 1]
    for i in range(2, n):
        fibo.append(fibo[-2] + fibo[-1])

    return fibo
Cython python example

In [14]: import cython_fibo2

In [15]: %timeit python_fibo.fibonacci(100)
   10000 loops, best of 3: 24.6 µs per loop

In [16]: %timeit cython_fibo.fibonacci(100)
   1000000 loops, best of 3: 8.16 µs per loop

In [17]: %timeit cython_fibo2.fibonacci(100)
   1000000 loops, best of 3: 6.97 µs per loop
Good artists copy; great artists steal.
The easiest way to build an interface...
... is to use one that’s already built.
```
In [12]: import numpy as np

In [13]: np.log(np.arange(1, 360, 5))
Out[13]:
array([  0.        , 1.79175947, 2.39789527, 2.77258872, 3.04452244,
        3.25809654, 3.4339872 , 3.58351894, 3.71357207, 3.8286414 ,
        3.93182563, 4.02535169, 4.11087386, 4.18965474, 4.26267988,
        4.33073334, 4.39444915, 4.4543473 , 4.51085951, 4.56434819,
        4.61512052, 4.66343909, 4.7095302 , 4.7539019 , 4.79579055,
        4.83628191, 4.87519732, 4.91265489, 4.94875989, 4.98360662,
        5.01727984, 5.04985601, 5.08140436, 5.11198779, 5.14166356,
        5.170484 ,  5.19849703, 5.22574667, 5.25227343, 5.27811466,
        5.30330491, 5.32787617, 5.35185813, 5.37527841, 5.39816227])
```
from pyxll import xl_func

@xl_func("string name: string", thread_safe=True)
def hello(name):
    """return a familiar greeting""
    return "Hello, %s" % name

@xl_func("int n: int", category="fibonacci", thread_safe=True)
def fibonacci(n):
    """naive iterative implementation of fibonacci""
    a, b = 0, 1
    for i in xrange(n):
        a, b = b, a + b
    return a
Rcpp / Rinline examples
```cpp
#include <Rcpp.h>
using namespace Rcpp;
#include <algorithm>

// [[Rcpp::export]]
double vectorSum(NumericVector x) {
    return std::accumulate(x.begin(), x.end(), 0.0);
}
```
require (inline)

src <- "
int n = as<int>(ns);
int fact = 1;
    for (int i = 1; i < n; ++i)
        fact *= i;
    return wrap(fact);
"

fact <- cxxfunction(signature(ns = "integer"),
    plugin = "Rcpp",
    body = src)
Great!
Impedance Mismatch
Another term we've stolen from engineering
<table>
<thead>
<tr>
<th></th>
<th>Difficult to change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hard</td>
<td></td>
</tr>
<tr>
<td>Soft</td>
<td>Flexible, but can be slow</td>
</tr>
<tr>
<td>Hard</td>
<td>Optimised for speed</td>
</tr>
<tr>
<td>Hard</td>
<td>Difficult to change</td>
</tr>
<tr>
<td>----------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Soft</td>
<td>Flexible, but can be slow</td>
</tr>
<tr>
<td>Hard</td>
<td>Optimised for speed</td>
</tr>
</tbody>
</table>
Impedance Mismatch

Different Skillsets
Different Ways of Thinking
Different Levels
Context Switching / Debugging
Julia is a high-level, high-performance dynamic programming language for technical computing, with syntax that is familiar to users of other technical computing environments. It provides a sophisticated compiler, distributed parallel execution, numerical accuracy, and an extensive mathematical function library. The library, largely written in Julia itself, also integrates mature, best-of-breed C and Fortran libraries for linear algebra, random number generation, signal processing, and string processing. In addition, the Julia developer community is contributing a number of external packages through Julia’s built-in package manager at a rapid pace. IJulia, a collaboration between the IPython and Julia communities, provides a powerful browser-based graphical notebook interface to Julia.
function mandel(z)
    c = z
    maxiter = 80
    for n = 1:maxiter
        if abs(z) > 2
            return n-1
        end
        z = z^2 + c
    end
    return maxiter
end

function randmatstat(t)
    n = 5
    v = zeros(t)
    w = zeros(t)
    for i = 1:t
        a = randn(n,n)
        b = randn(n,n)
        c = randn(n,n)
        d = randn(n,n)
        P = [a b c d]
        Q = [a b; c d]
        v[i] = trace((P.'*P)^4)
        w[i] = trace((Q.'*Q)^4)
    end
    std(v)/mean(v), std(w)/mean(w)
end
bk@xk7.com
@georgebernhard (play)
@numericalR (work)