Extending Python for Speed

2010-05-27

Martin Renold

MyPaint

- Code
  - 80% Python, 20% C++
  - 12K lines of code
- Project
  - Started in 2004
  - Developement slow but steady
  - Popular since David Revoy's work on Durian

Why Python?

C++

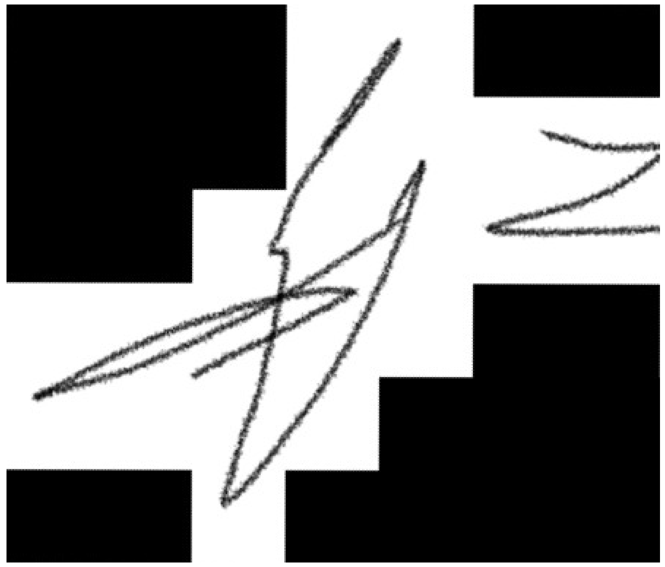
```
for(std::vector<std::string>::const_iterator
    i = items.begin(); i != items.end(); ++i) {
    ...
```

Python

```
for i in items:
    ...
```

Fast Enough?

- Python:
  - GUI
  - „for each tile“
  - „for each motion event“



- C/C++:
  - „for each pixel“
  - low-level algorithms (eg. interpolation)

Tools to Extend Python

• SWIG	-- C/C++
• Cython (Pyrex)	-- Python-like language
• h2defs.py	-- C (GObject), PyGTK

• SIP	-- C++, PyQt
• Boost.Python	-- C++
• ctypes	-- Load .so/.dll in Python
• ...	

SWIG: Code

hello.hpp

```
int answer() {
    return 42;
}
```

hello.i

```
%module hello
%{
#include "hello.hpp"
%}
%include "hello.hpp"
```

SWIG: Compiling

setup.py

```
from distutils.core import setup, Extension

setup(ext_modules=[
  Extension("_hello", ["hello.i"])
])
```

```
$ python setup.py build_ext -i
$ python
>> import hello
>> hello.answer()
42
```

SWIG: The End.

- Do not learn more SWIG!
  - People have died while trying to figure out SWIG Typemaps
- Use the Python/C API
  - SWIG supports this

Python/C API

- Reference Counting
  - Py_DECREF, Py_INCREF macros

```
PyObject * func(PyObject * arg);
```

New Reference

Borrowed Reference

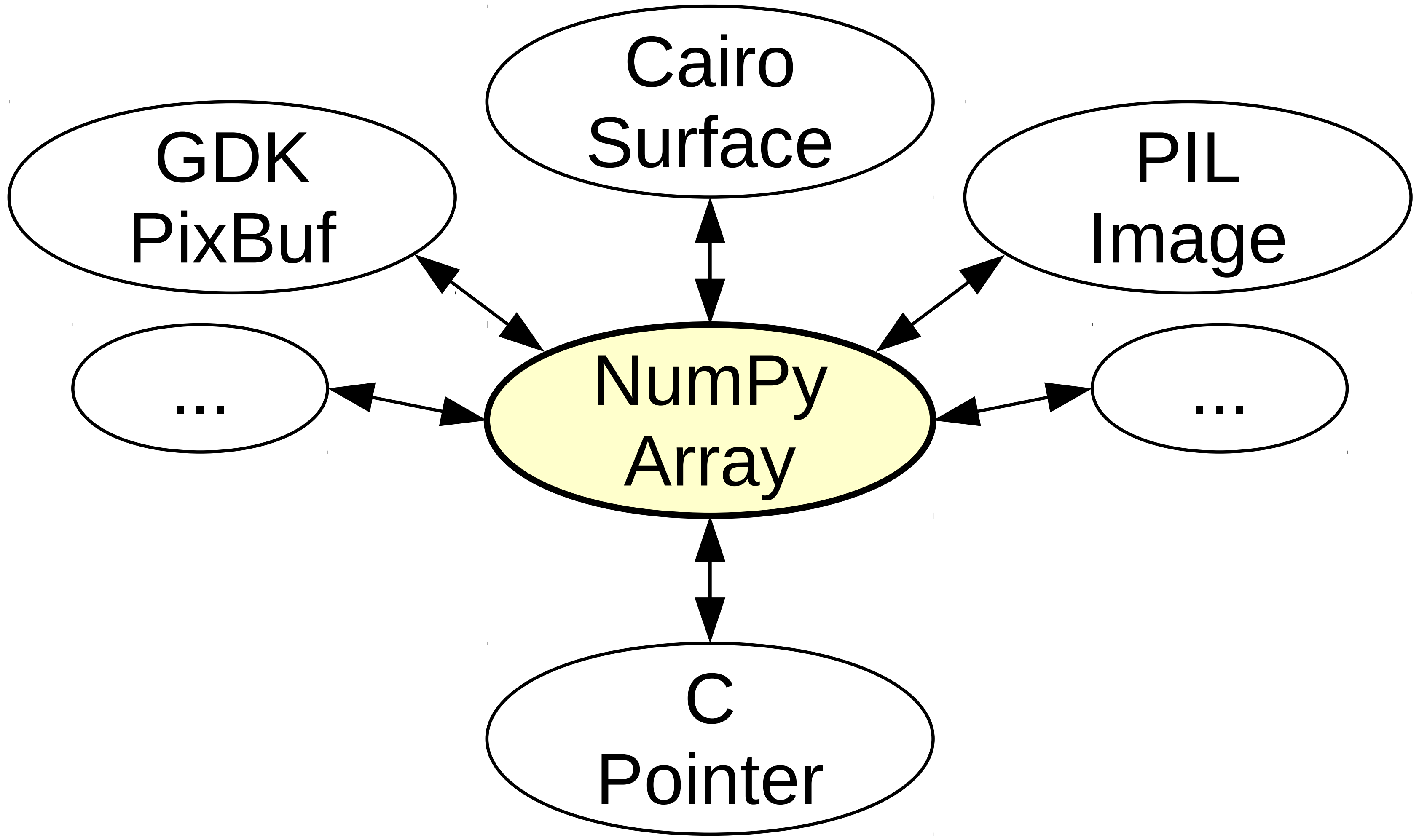
Example

```
class Gradient {
  public:
  float parm1;

  PyObject * get_color(float x, float y) {
    int r, g, b;
    // ...
    return Py_BuildValue("ddd", r, g, b);
  }
};
```

```
>> g = hello.Gradient()
>> g.parm1 = 2.8
>> r, g, b = g.get_color_at(0, 0)
```

Sharing Pixel Memory (without copy)

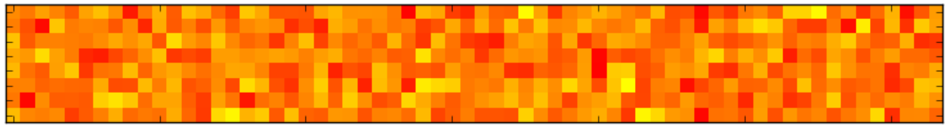


NumPy (and SciPy)

```
from pylab import *

pix = zeros((64, 8, 3), 'uint8')
pix[:,:,0] = 255
pix[:,:,1] = 128 + 60 * randn(64,8)
pix[:,:,2] = 0

imshow(pix)
```



Interfacing with NumPy

hello.hpp

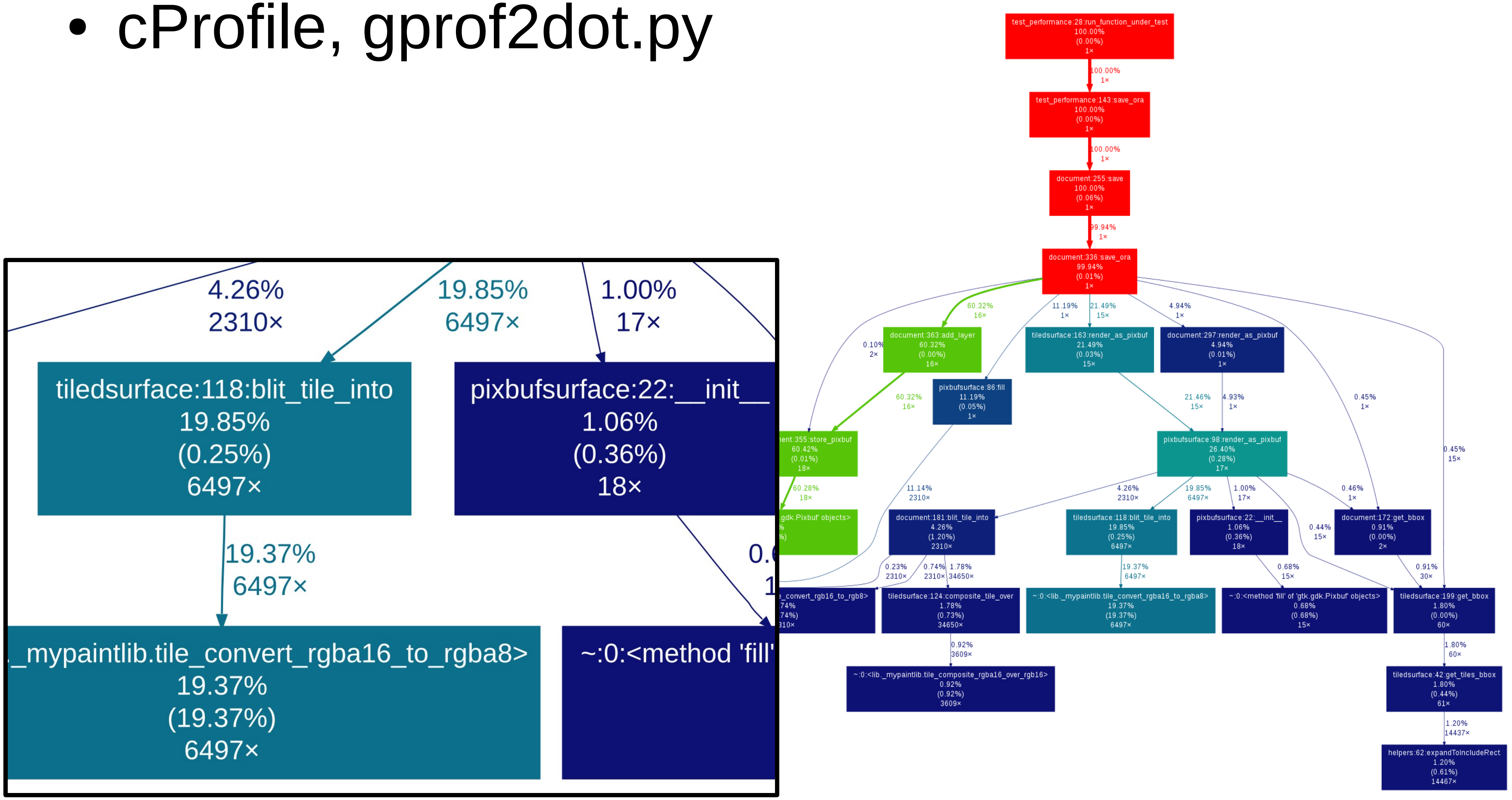
```cpp
void render(PyObject * arr, int radius) {
  int h, w;
  uint8_t * p;

  h = PyArray_DIM(arr, 0);
  w = PyArray_DIM(arr, 1);

  p = (uint8_t*)((PyArrayObject*)arr)->data;
```

Profiling Python

- cProfile, gprof2dot.py



test_performance:28:run_function_under_test 100.00% (0.00%) 1×
test_performance:143:save_ora 100.00% (0.00%) 1×
document:255:save 100.00% (0.06%) 1×
document:336:save_ora 99.94% (0.01%) 1×
document:363:add_layer 60.32% (0.00%) 16×
tiledsurface:163:render_as_pixbuf 21.49% (0.03%) 15×
document:297:render_as_pixbuf 4.94% (0.01%) 1×
pixbufsurface:86:fill 11.19% (0.05%) 1×
pixbufsurface:98:render_as_pixbuf 26.40% (0.28%) 17×
document:181:blit_tile_into 4.26% (1.20%) 2310×
tiledsurface:118:blit_tile_into 19.85% (0.25%) 6497×
pixbufsurface:22:__init__ 1.06% (0.36%) 18×
document:172:get_bbox 0.91% (0.00%) 2×
tiledsurface:124:composite_tile_over 1.78% (0.73%) 34650×
~:0:<lib._mypaintlib.tile_convert_rgba16_to_rgba8> 19.37% (19.37%) 6497×
~:0:<method 'fill' of 'gtk.gdk.Pixbuf' objects> 0.68% (0.68%) 15×
tiledsurface:199:get_bbox 1.80% (0.00%) 60×
~:0:<lib._mypaintlib.tile_composite_rgba16_over_rgb16> 0.92% (0.92%) 3609×
tiledsurface:42:get_tiles_bbox 1.80% (0.44%) 61×
helpers:62:expandToIncludeRect 1.20% (0.61%) 14467×
4.26% 2310×
19.85% 6497×
1.00% 17×
tiledsurface:118:blit_tile_into 19.85% (0.25%) 6497×
pixbufsurface:22:__init__ 1.06% (0.36%) 18×
19.37% 6497×
_mypaintlib.tile_convert_rgba16_to_rgba8> 19.37% (19.37%) 6497×
~:0:<method 'fill'

OProfile: System Profiler

```
$ opreport
...
389232 57.0081 Xorg
   ------------------
      360088 92.5124 libpixman-1.so.0.
       15630  4.0156 libxaa.so
       10684  2.7449 libc-2.10.2.so
       ...
 146698 21.4858 python2.5
   ------------------
       85310 58.1535 python2.5
       31580 21.5272 _mypaintlib.so
       15333 10.4521 libc-2.10.2.so
```

Debugging

- Same as debugging any C/C++ library

```
$ gdb /usr/bin/python
(gdb) run program.py
```

Memory Leaks

- Unused References (common)
  - Hard to find, no tools (?)
- Reference Cycles with __del__
  - check gc.garbage
  - SWIG generates empty __del__ (disable it)
- Missing Py_DECREF (rare)

Thanks

- Code Samples:

`http://github.com/martinxyz/python`

BACKUP

Why Not Python?

- Personal taste
  - Dislike syntax: self, __init__, whitespace
- Invested into C/C++
  - Existing codebase, expertise
- Performance
  - Not willing to use the Python/C API
- Contributors
  - C/C++ coders tend to be more experienced

Not Extending Python

- Use libraries from Python
  - eg. GDK-PixBuf, Cairo, NumPy
  - for standard tasks (eg. compositing)

```
cr.set_line_width(3.0)
cr.set_line_join(cairo.LINE_JOIN_ROUND)
cr.rectangle(10, 10, 90, 90)
cr.stroke()
```

NumPy Array

- PIL:
  - fromarray
- Cairo:
  - create_for_data
- GDK-PixBuf:
  - new_from_array
  - get_pixels_array

Interfacing with NumPy

hello.i

```
%module hello
%{
#include <numpy/arrayobject.h>
#include "hello.hpp"
%}
%include "hello.hpp"

%init %{
import_array();
%}
```

Interfacing with NumPy

hello.hpp

```
void render(PyObject * arr, int radius) {
  int h, w;
  uint8_t * p;

  assert (PyArray_ISCARRAY(arr));
  assert (PyArray_NDIM(arr) == 3);
  assert (PyArray_DIM(arr, 2) == 3);

  h = PyArray_DIM(arr, 0);
  w = PyArray_DIM(arr, 1);

  p = (uint8_t*)((PyArrayObject*)arr)->data;
```

OProfile: System Profiler

```
$ opreport -l /usr/bin/python2.5

37.8 libpng12.so
 8.7 libz.so
 7.8 _mypaintlib.so tile_convert_rgba16_to_
 4.4 multiarray.so
 4.1 python2.5      PyEval_EvalFrameEx
 3.3 _mypaintlib.so tile_convert_rgba8_to_r
 2.6 umath.so
 2.3 libc-2.10.2.so random
 2.3 libc-2.10.2.so memcpy
```