Extending and Wrapping C and C++ with Python

Diego Rodriguez-Losada

@diegorlosada

accu
professionalism in programming

23-April-2016

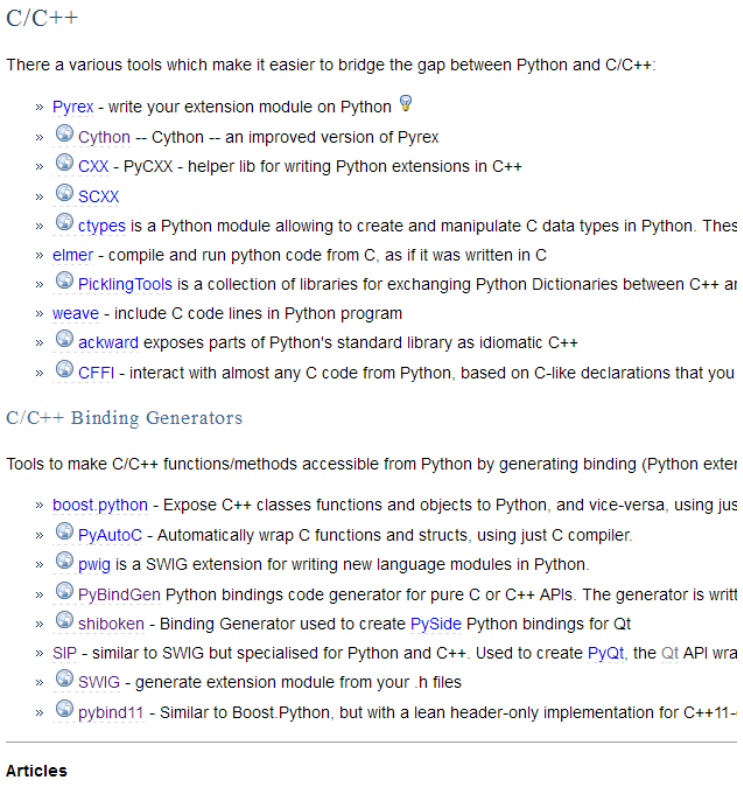
Intro

- Extending python with C/C++ extensions
  - Performance
  - Wrapping existing libraries
  - Integrations
- Embeding python in C/C++ apps
  - Scripting in your app
  - User extensions, config

* Will not cover Cython (Python & Cython => C)

Outline

- Extending python with C/C++
  - Bindings:
    - Python/C API
    - Pybind11
  - DLL
    - Ctypes
    - Cffi
  - Code generation:
    - SWIG
- Embedding python in C/C++ apps
  - Python/C API



C/C++

There a various tools which make it easier to bridge the gap between Python and C/C++:

- Pyrex - write your extension module on Python
- Cython -- Cython -- an improved version of Pyrex
- CXX - PyCXX - helper lib for writing Python extensions in C++
- SCXX
- ctypes is a Python module allowing to create and manipulate C data types in Python. Thes
- elmer - compile and run python code from C, as if it was written in C
- PicklingTools is a collection of libraries for exchanging Python Dictionaries between C++ ar
- weave - include C code lines in Python program
- ackward exposes parts of Python's standard library as idiomatic C++
- CFFI - interact with almost any C code from Python, based on C-like declarations that you

C/C++ Binding Generators

Tools to make C/C++ functions/methods accessible from Python by generating binding (Python exter

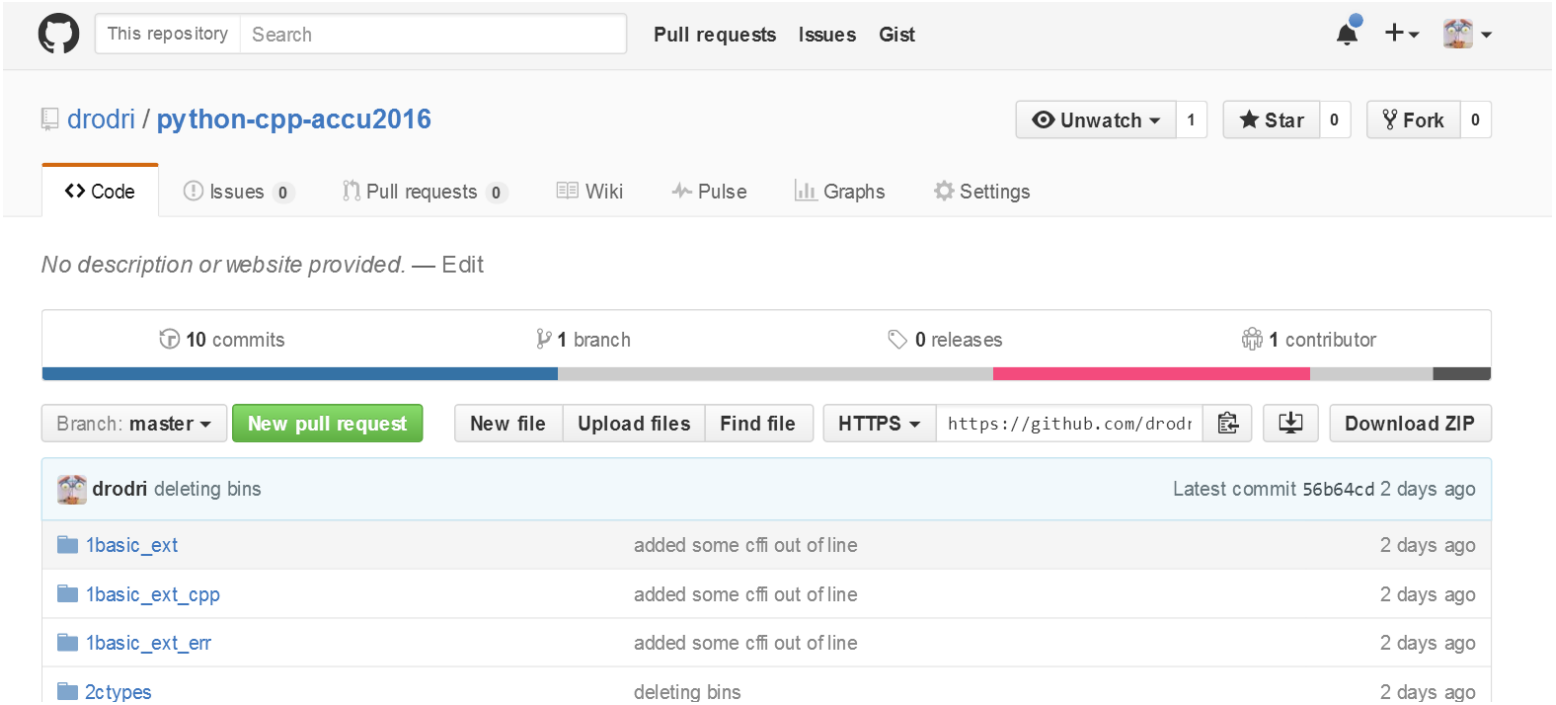
- boost.python - Expose C++ classes functions and objects to Python, and vice-versa, using jus
- PyAutoC - Automatically wrap C functions and structs, using just C compiler.
- pwig is a SWIG extension for writing new language modules in Python.
- PyBindGen Python bindings code generator for pure C or C++ APIs. The generator is writt
- shiboken - Binding Generator used to create PySide Python bindings for Qt
- SIP - similar to SWIG but specialised for Python and C++. Used to create PyQt, the Qt API wra
- SWIG - generate extension module from your .h files
- pybind11 - Similar to Boost.Python, but with a lean header-only implementation for C++11-

Articles

[1] https://wiki.python.org/moin/IntegratingPythonWithOtherLanguages

Enough slides!

https://github.com/drodri/python-cpp-accu2016

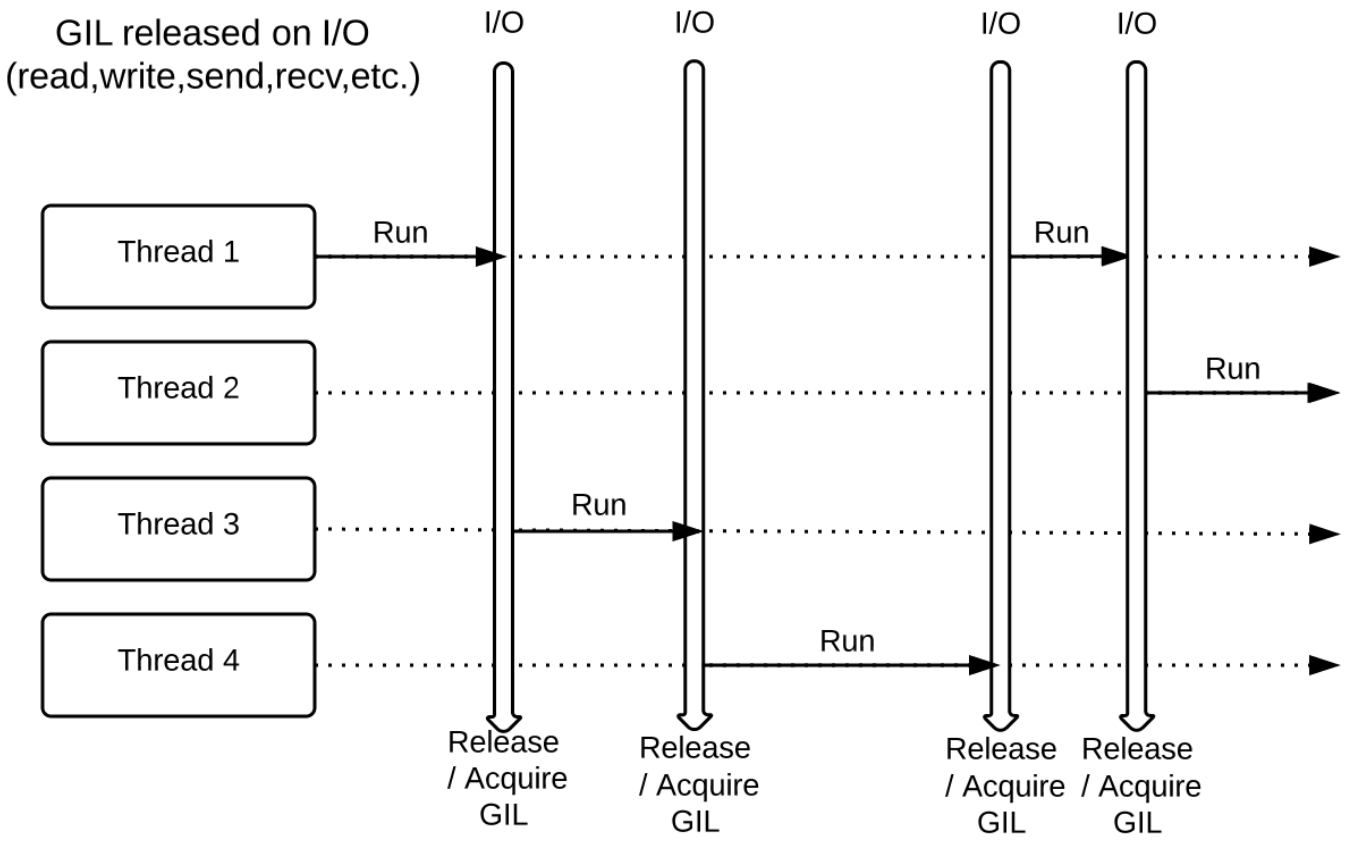


GIL

- As David Beazley writes in The Unwritten Rules of Python:
  - *1. You do not talk about the GIL.*
    2. You do NOT talk about the GIL.
    3. Don't even mention the GIL. No seriously.

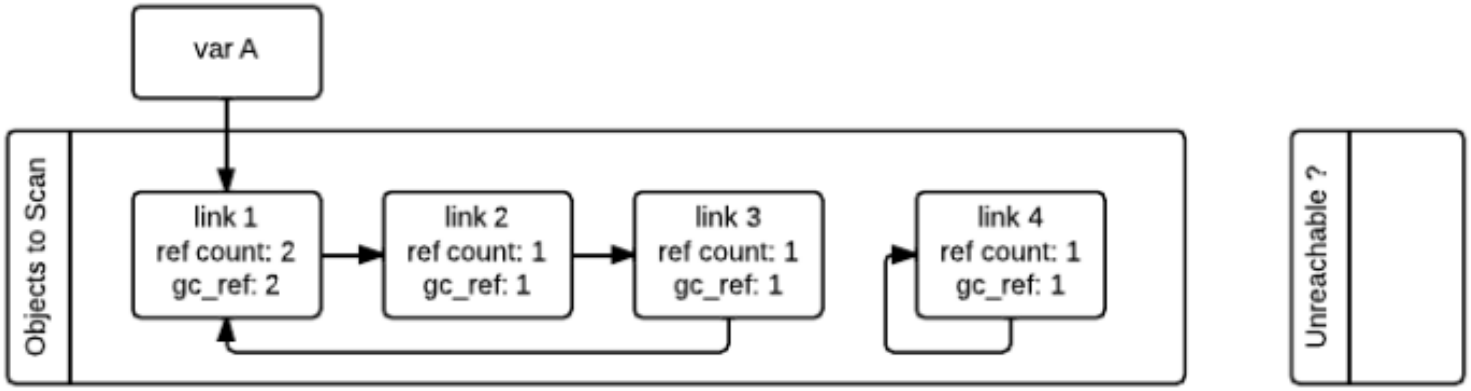
[1] http://www.slideshare.net/dabeaz/understanding-the-python-gil

GIL



[1] http://www.tivix.com/blog/lets-go-python/

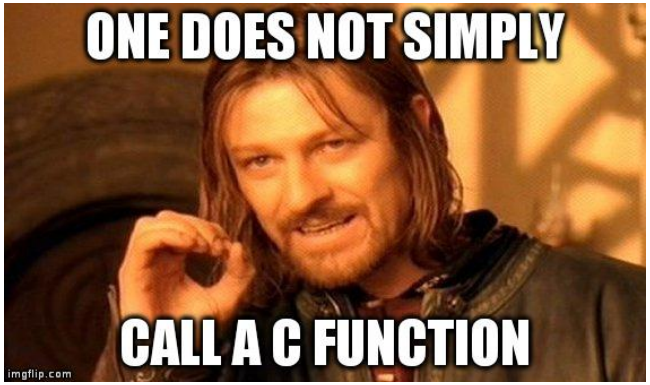
Python Auto GC



[1] http://9gag.com/gag/anB2KzE/this-is-how-your-multi-core-cpu-works
[2] https://pythoninternal.wordpress.com/2014/08/04/the-garbage-collector/

Conclusion

- Extending python with C/C++
  - Python/C API: Best
    - Improve python performance
    - Use existing libs (partially)
  - Pybind11
    - Very C++ full lib bindings
  - ctypes, CFFI
    - Full C libraries bindings
  - Swig:
    - Binding to other languages
- Embedding python in C/C++ apps
  - Python/C API: Don’t invent your IDL



Extending and Wrapping C and C++ with Python

Diego Rodriguez-Losada

@diegorlosada

accu
professionalism in programming

23-April-2016