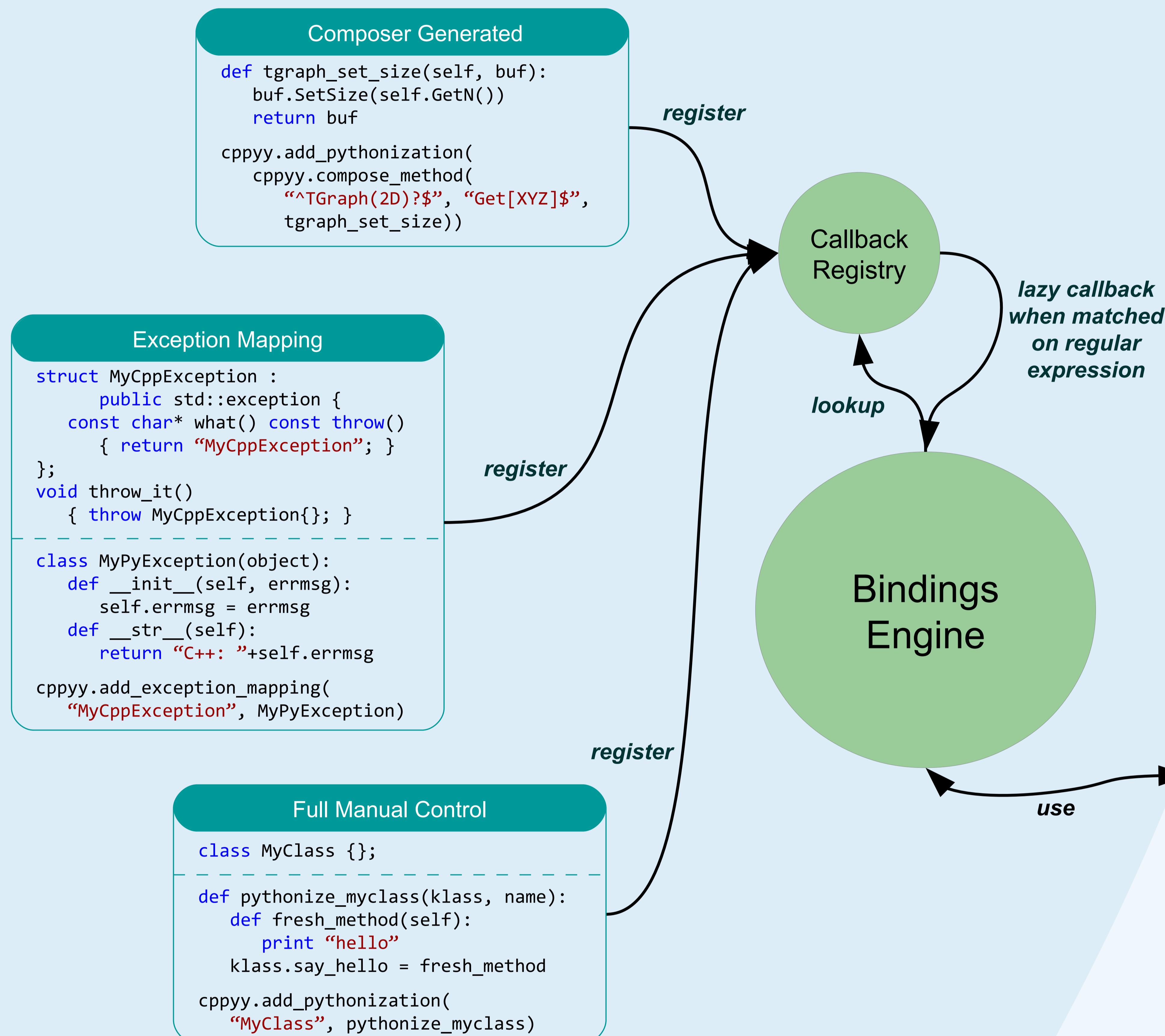


Pythonization API for CPPYY

A New Level of Integration between Python and C++

cppyy is a fully automated, run-time, language bridge between C++ and Python. Full automation is very effective and covers the vast majority of use cases, but strictly bound C++ code sometimes lacks a Python “feel” to it, making it harder to understand and use than necessary. We provide a pythonization API to easily cover common cases and allow further custom detailing where desired.



Pythonizations Supported

- Method decoration/adaptation
- Getter/setters → properties
- Overload additions
- Method/variable renaming
- Memory ownership control
- GIL management
- Transparent smart pointers
- Iteration on STL-like containers
- Exception remapping
- Return type pinning

Or, take full manual control using reflection and JIT!

Python Interpreter

```

>>> from cppyy.gbl import TGraph, MyClass, throw_it
>>> N = 5
>>> vals = array.array('d', xrange(N))
>>> g = TGraph(N, vals, vals)
>>> print len(g.GetX()) == N # naked pointer return
True # has its size set
>>> m = MyClass()
>>> m.say_hello() # added by custom pythonization
hello
>>> throw_it() # throws C++ MyCppMethod
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
__main__.MyPyException: C++: void ::throw_it() =>
    MyCppMethod
>>> # is translated to MyPyException

```

Pythonization API

Python classes are easily augmented, but only once they exist, which can lead to unnecessary instantiations and loading of modules and libraries. Our API design provides pre-registration of callbacks and lazy invocation just after class creation.

The API provides composers covering a range of common cases. Callbacks are matched using regular expressions, and scoped for code/project organization. A project's coding conventions can be leveraged to capture and pythonize application-specific patterns.

Full manual control is available by introspecting classes in the callback. C++ helper functions can be generated, JITed, bound and used to further augment a Python class.

Pythonic ROOT: rootpy

Provides pythonic interfaces on top of PyROOT and integration with packages such as SciPy, scikit-learn, matplotlib, etc.

See: <http://rootpy.org>

References

- <http://doc.pypy.org/en/latest/cppyy.html>
- <http://root.cern.ch/drupal/content/pyroot>
- <http://root.cern.ch/drupal/content/pypyroot>