C++ as a service — rapid software development and dynamic interoperability with Python and beyond

Interactive C++: cling and clang-repl

Ioana Ifrim & Vassil Vassilev

07.07.2022

Status. Cling

- Continuing to rebase cling on top of llvm13, need to support unloading for CallFunc; working on a prototype for the JIT deadlock issue
- Improving the stability of cpt.py used to build and package cling

Status. Clang-Repl

- Error recovery for template instantiations completed
- Added support for weak symbols
- Fixed PPC exception issues for compiled and interpreted code
- Taught clang to parse statements on the global scope: D127284

The goal is to provide a more stable error recovery approach than the currently available one in cling

Clang-Repl and Xeus

- Within clang-repl, a python interactive session can now be invoked; here, python is aware of the variables declared in C++

```
(base) ioana@Ioanas-MacBook-Pro build % ./bin/clang-repl
clang-repl> python
>>> globals()
{'__builtins__': <module '__builtin__' (built-in)>, '__name__': '__main__', '__doc__': None, '_
_package__': None}
>>> new_var
Traceback (most recent call last):
  File "", line 1, in <module>

NameError: name 'new_var' is not defined
>>> ^D
clang-repl> int new_var = 0;
clang-repl> python
>>> globals()
{'__builtins__': <module '__builtin__' (built-in)>, '__name__': '__main__', 'new_var': 0, '__do
c__': None, '__package__': None}
>>> new_var
0
>>> import numpy as np
>>> a = np.asarray([new_var, new_var + 1, new_var +2])
>>> a
array([0, 1, 2])
```

Clang-Repl and Xeus

- We have Clang-Repl and Xeus integration for the C++ kernel

jupyter xeus & clang-repl Integration Demo Last Checkpoint: 23 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Help Trusted clang-repl

Run Code

In [1]:
```
#include <iostream>
#include <vector>
```

In [2]:
```
extern "C" int printf(const char*,...);
```

Declaring variables

In [3]:
```
int new_var = 0;
int extra_var = 11;
```

In [4]:
```
auto a = printf("new_var = %d\nextra_var = %d\n", new_var, extra_var);
```

```
new_var = 0
extra_var = 11
```

Working with a vector type

In [5]:
```
std::vector<int> fill_vector(std::vector<int> g) { for (int i = 1; i <= 5; i++){ g.push_back(i);} return g; }
```

In [6]:
```
int size_of_vector (std::vector<int> g) { std::cout << "Size : " << g.size(); return g.size(); }
```

In [7]:
```
int capacity_of_vector (std::vector<int> g) { std::cout << "Capacity : " << g.capacity(); return g.capacity(); }
```

In [8]:
```
std::vector<int> g1 = fill_vector(g1);
```

In [9]:
```
int s = size_of_vector(g1);
```

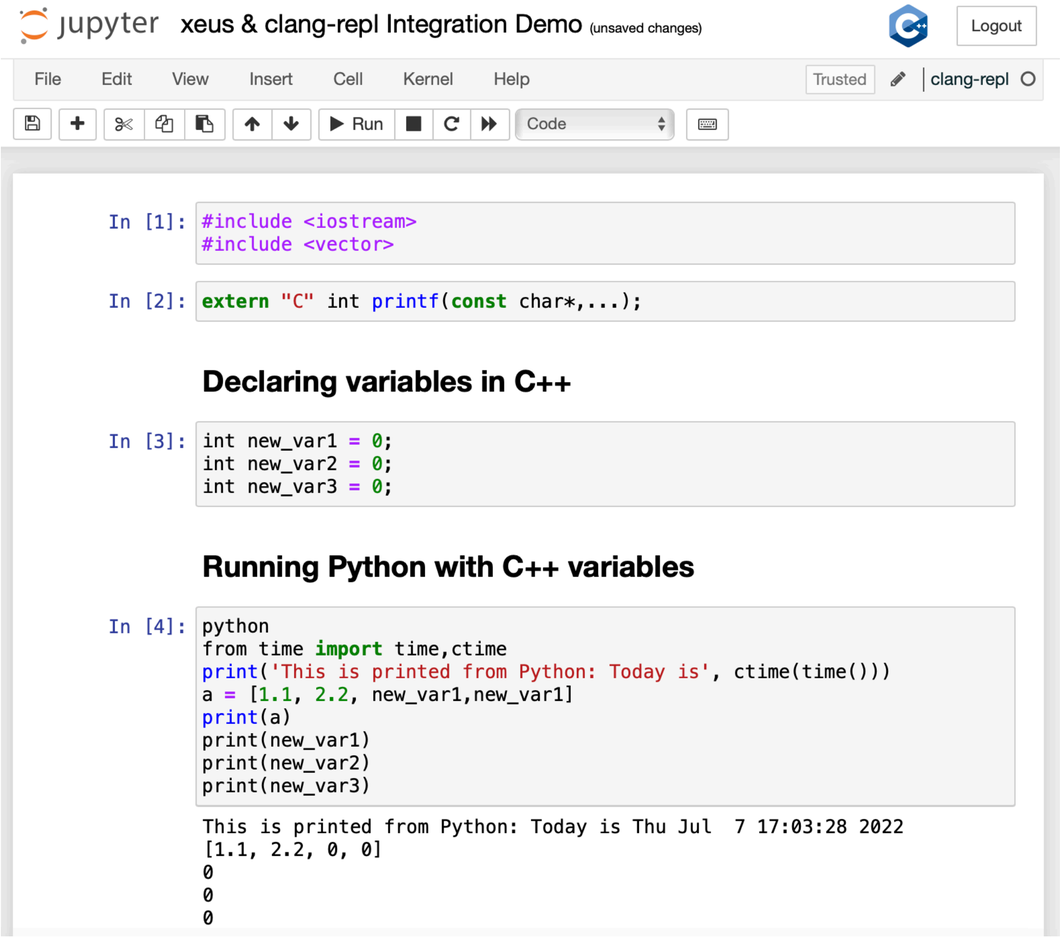
```
Size : 5
```

In [10]:
```
int c = capacity_of_vector(g1);
```

```
Capacity : 5
```

Clang-Repl and Xeus

- Working on bringing python features working in clang-repl to Jupyter



Status. InterOp

- Working on a full surgery of cppyy where we split it into libInterOp
- Working on simplifying CallFunc and moving it in libInterOp: PR10850

Status. Clad

- Added basic infrastructure for integration with Enzyme
- Ongoing integration of Clad in RooFit, now the hf001 example works.

Upstreaming Patches

- Spreadsheet tracking the progress here.
- Reduced 8 more patches.

CaaS Open Projects

- Open projects are tracked in our open projects page.

Next Meetings

- Monthly Meeting — 4th Aug, 1700 CET/0800 PDT

If you want to share your knowledge/experience with interactive C++ we can include presentations at an upcoming next meeting

Thank you!