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

Interactive C++: cling and clang-repl

Vassil Vassilev

05.05.2022
Status. Cling

- Continuing to rebase cling on top of llvm13, working on some related to ROOT jit issues
Status. Clang-Repl

- Working on error recovery for template instantiations
- Working on teaching clang to parse statements on the global scope.

The goal is to provide a more stable error recovery approach than the currently available on in cling
Status. InterOp

- Working towards supporting the initial cppyy `import cppyy`.
- Progress on Numba/cppyy integration - we now can call a C++ function pointer provided by Cling in Numba.

The goal is to rework the python-to-C++ automatic binding generator cppyy to use LLVM interfaces which can help improving speed and accuracy.
Status. Clad

- Added support for reference return types.
- Enabled differentiation of std::accumulate, std::vector and std::map
- Floating-point error estimation talk went well. More info here.
- Working on various prototypes and have a concrete approach for getting Clad derivatives from RooFit functions. Status presentation available here.
Tapasweni Pathak
Principal Product Manager at Microsoft Azure Core Engineering
Improving performance of C++ modules in Clang. Upstream some of our cling patches (May 2022-Sep 2022)
CaaS Open Projects

- Patches against clang.git - we track them here:
  - Implement FileManager uncaching
  - Adapt the user of invalidateCache to its new signature
  - Mark the file entry invalid, until reread
  - Propagate cache flags from LookupFile() to FileManager::getFile()
  - Pass the OpenFile flag also to DirectoryLookup
  - Do not load the source file just to get an irrelevant SourceLoc (ROOT-7111)
  - Allow interfaces to operate on in-memory buffers with no source location info [Pratyush Das]

- Open projects are tracked in our open projects page.
Next Meetings

- Monthly Meeting — 9th Jun, 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!