Therapeutic beam particle generator based on IAEA phase space characteristic

HOW TO SETUP C/C++ BUILD PIPELINE TO CREATE PYTHON LIBRARIES

GOALS

- Simplify calibration proces of equipment used in radio therapy
- Reduce data volume from 1 TB -> suitable size for storing and rerun simulation (PHSP files)
- Simplify computer simulation of the therapeuetic beam
- Modernization of IAEA standard
- Reduce redundant computation (include metadata in files)

Get right tool for the job



CPP

- compiled
- fast execution
- better at complex projects
- lack of standard in development tools



PYTHON

- interpreted
- fast prototyping
- better comunity support(libraries, documentation, forums)
- greate for integrating other technologies
- official tools (packet manager, deplyment system)

ROAD MAP



Better in terms of code quality (OOP, ...)

ANALYSIS

Take advantage of a high-quality data analysis ecosystem (jupyter,matpololib,...)



 \mathbf{C}

Fast, binary files, lack of documentation, hard to maintain

PYTHON

Expose Cpp library as python module

Create generator as a particle gun tool

Use modern Cpp for performance, quality control, easy mentaince

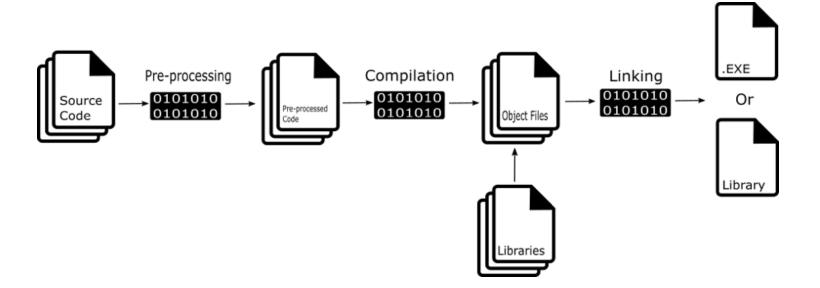
Build systems

CPP:

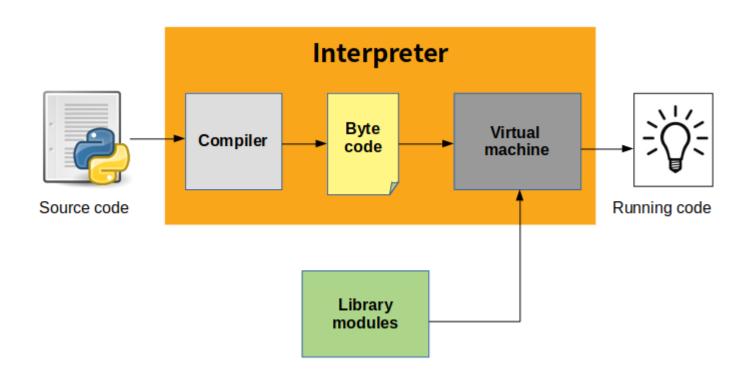
- Make
- Ninja
- MSBuild
- And dozens more

Meta Build Systems:

CMake



Python interptreter



Python bindings

CYTHON

The approach Cython takes to creating Python bindings uses a Python-like language to define the bindings and then generates C or C++ code that can be compiled into the module.

Low-level, code written with intetion to run only in python enviroment

BOOST.PYTHON

Library from huge(a lot of dependencies) project BOOST.

Compatibile with all C++ versions. (backward compatibilty)

PyBind11

Written to mimic BOOST.PYTHON, self contained support only C++ 11 and newer.

Packet managers and others deployment tools



CPP

- Conan
- Vcpkg
- Conda
- and dozens more



PYTHON

- Pip
- Conda
- Venv

Build pipeline



VENV

Activate virtual env

SETUP

Manage modules required by setup.py script

SETUP

Rerun setup.py to download all dependencies and start build process **CMake**

Run CMake with flags from received from setup.py Make

Use any build system compatible with CMake

SETUP

Add all .py and .pyc files to import path

Minimal project

https://github.com/ZychuDev/pybind11-CMake-Setup-Project-Structure