cxxpods Documentation

Jonathan Glanz

Jun 30, 2019

Contents

logo

CXXPods is a command line tool that allows for both simple c/c++ and very complex multi-platform, cross-compiling dependency managerment.

Think of it as a highly customizable (obviously because it needs to support complex native code) NPM or Maven like package management solution that deals in source artifacts instead of compiled or binary artifacts.

It comes as an amalgamation of other attempts including awesome projects like mason, hunter & conan.

In order to be as comprehensive as possible, there is a single enforced guideline, your project must you CMake.

CHAPTER 1

Install

1.1 Linux, macOS

On linux or macOS you can run the following snippet to install a binary version of cxxpods, no need to install node, npm or anything else. It will be installed to `/usr/local/bin`

```
curl -s https://raw.githubusercontent.com/cxxpods/cxxpods/master/scripts/client-
↪install.sh | bash
```

1.2 NPM (Windows, Linux, macOS)

You can also install via npm if you have node and npm installed.

```
npm i -g cxxpods
```

CHAPTER 2

Quickstart

It takes a few simple steps.

- Create a cxxpods.yml file in the root of your project

```yaml
name: cxxpods-example

dependencies:
  opencv: 3.4.1
```

- Then run configure

```
cxxpods project configure
```

- In your root `CMakeLists.txt` *BEFORE* your `project` declaration add something like the following:

```cmake
cmake_minimum_required(VERSION 3.10)

# INSERT THIS LINE
include(${CMAKE_CURRENT_LIST_DIR}/.cxxpods/cxxpods.cmake)

project(cxxpods_example)
```

CHAPTER 3

Repos and Recipes

CXXPODS works with `recipes` in repositories. The global default repository is here: github.com/cxxpods/cxxpods-registry

3.1 Dependency/Recipe

Each dependency/recipe in a repo must contain a `cxxpods.yml` file at a minimum. Assuming it needs to be findable as a library within `CMake`, it also must have a finder template.

3.1.1 Recipe (cxxpods.yml)

```
name: opencv
repository:
  url: https://github.com/opencv/opencv.git

cmake:
  flags:
    CMAKE_BUILD_TYPE: Release
    BUILD_SHARED_LIBS: OFF
    BUILD_JPEG: OFF
    BUILD_JASPER: OFF
    BUILD_PNG: ON
    BUILD_ZLIB: ON
    BUILD_IPP_IW: OFF
    BUILD_ITT: OFF
    BUILD_JAVA: OFF
    BUILD_PROTOBUF: OFF
    WITH_PROTOBUF: OFF
    WITH_CAROTENE: OFF
    WITH_CUBLAS: OFF
    WITH_CUDA: OFF
```

(continues on next page)

(continued from previous page)

```
    WITH_CUFFT: OFF
    WITH_FFMPEG: OFF
    WITH_GPHOTO2: OFF
    WITH_GSTREAMER: OFF
    WITH_GTK: OFF
    WITH_ITT: OFF
    WITH_IPP: OFF
    WITH_JASPER: OFF
    WITH_LAPACK: OFF
    WITH_MATLAB: OFF
    WITH_NVCUVID: OFF
    WITH_OPENCL: OFF
    WITH_OPENCLAMDBLAS: OFF
    WITH_OPENCLAMDFFT: OFF
    WITH_OPENEXR: OFF
    WITH_PTHREADS_PF: OFF
    WITH_V4L: OFF
    WITH_WEBP: OFF
  findTemplate: cmake/FindOpenCV.cmake.hbs

dependencies:
  libtiff: Release-v4-0-9
  zlib: v1.2.11
  libpng: v1.6.33
  libjpeg: 8.4.0
```

3.1.2 Finder Template

A finder template (as well as all templates used within CXXPODS) is a `Handlebars` template, i.e. `cmake/FindOpenCV.cmake.hbs`.

```
if(NOT OpenCV_FOUND)
    set(_OpenCV_LIBS
        calib3d features2d flann highgui imgcodecs
        imgproc ml objdetect photo shape stitching superres
        video videoio videostab core
        )


    foreach(_lib ${_OpenCV_LIBS})
        set(_target OpenCV::${_lib})
        set(_libPath {{cxxpodsLibDir}}/${CMAKE_STATIC_LIBRARY_PREFIX}opencv_${_lib}$
↪{CMAKE_STATIC_LIBRARY_SUFFIX})

        list(APPEND OpenCV_LIBRARIES ${_libPath})
        list(APPEND OpenCV_TARGETS ${_target})
        if (NOT TARGET ${_target})
            add_library(${_target} STATIC IMPORTED)
            set_target_properties(${_target} PROPERTIES
                IMPORTED_LOCATION ${_libPath}
            )
        endif()
    endforeach()

    set(OpenCV_FOUND true)
```

(continues on next page)

(continued from previous page)

```
endif()
```

Template Variables

- `cxxpodsLibDir` the install lib dir
- `cxxpodsIncludeDir` the install include dir

3.2 Add your Own Repo

There are a large number of reasons to add your own recipe repos

- Custom configuration of recipes, examples include
  - Recipes configured for a Raspberry Pi specifically.
  - A different FFMPEG configuration that enables CUDA
  - OpenCV with Java/Python support
- Private recipes
  - **Yes we are happy for you to use CXXPODS commercially**
- Offline recipes

3.2.1 Structure of a Repo

A repo is really in simplest terms, a folder with child folders that are each named respective to a given dependency, i.e. "opencv".

```
TOP OF REPO
| -> opencv
| | -> cmake
| | | -> FindOpenCV.cmake.hbs
| | -> cxxpods.yml
```

3.2.2 Example Commands

note you can add your own repos *public or private* both locally and git based as follows

```
# GITHUB EXAMPLE PUBLIC OR PRIVATE
cxxpods repo add https://github.com/myorg/my-cxxpods.git
# OR A LOCAL DIR
cxxpods repo add file:///var/cxxpods-local-on-disk
```

3.3

CHAPTER 4

Tools

First and foremost - this is not a page that is dedicated to David Hasselhoff.

More info will follow soon enough.

CHAPTER 5

Android

CXXPODS supports Android out of the box. Simply add `android: true` to your `cxxpods.yml`, also you will likely want to exclude the host toolchain as well. Below is a very brief example.

5.1 Android vs Cross-Compilation

The primary difference, and it is significant, is that the dependencies are created/built when you run `Sync` in Android Studio as opposed to when you run `configure` normally. Tools are still built during `configure`.

5.2 Config

The `cxxpods.yml` should be in the module folder of your project, not the root, i.e. `<root>/app/cxxpods.yml`.

```
name: my-android-project
android: true
toolchainExcludeHost: true

dependencies:
        opencv: 3.4.1
```

5.3 Configure

After creating your config file, you need to run `configure` before adding to your `CMakeLists.txt`.

```
# Get to your app modules
cd <root>/app

# Configure
cxxpods configure
```

5.4 CMakeLists.txt

Just as you do with a regular project, add the `cxxpods.cmake` that was generated to your project.

```
cmake_minimum_required(VERSION 3.10)

# INSERT THIS LINE
include(${CMAKE_CURRENT_LIST_DIR}/.cxxpods/cxxpods.cmake)

project(cxxpods_example)
```

CHAPTER 6

Cross-Compiling & Toolchains

Create your standard cmake toolchain file and use it as follows:

```
name: cxxpods-example
profiles: [Debug,Release]

toolchains:
  "aarch64-linux-gnu": cmake/aarch64.cmake
  # file would be at this relative location from the project root

dependencies:
  protobuf: 3.1.0
  opencv: 3.4.1
```

In order to use with `non-cmake` dependencies and scripts add the following to the top of your toolchain file:

```
include(${CMAKE_CURRENT_LIST_DIR}/.cxxpods/cxxpods.toolchain.cmake)
```

and add the following to the bottom of your toolchain file

```
cxxpods_toolchain_export()
```

CHAPTER 7

Other Commands

TBD