The KDE Buildsystem

- R. Harish Navnit
- Software Developer
  - KDE, JoyStream, FOSS@Amrita

CMake ?

Introduction

- **`make`** is a utility/tool that controls the generation of executables and/or libraries from source files.
- **`Makefiles`** dictate how **`make`** should function for a certain project.
- Writing a **`Makefile`** is often quite tedious and tricky.
- We hence rely on tools that generate **`Makefile`**’s for us.

Why bother ?

- Manually compiling individual source files in a large project probably won’t scale!
- **`Makefiles`** allow linking against target libraries from external projects.
- Allows better dependency management.

Why CMake ?

- Cross Platform
  - Can even generate Visual Studio projects!
- Only dependency is a C++ compiler
- Has it’s own testing framework
- Supports compiling of apps/libs
- Easily extendable to dependency management tools.
  Eg: Conan

Why KDE chose CMake ?

- https://lwn.net/Articles/188693/
- https://dot.kde.org/2007/02/22/road-kde-4-cmake-new-build-system-kde

Examples : CMake

- kde:kdeexamples/CMakeLists.txt

Finding Dependencies

- **`find_package`**`(package ${PACKAGE_VERSION} REQUIRED MODE)`
  - `MODE =` **`CONFIG`** `or` **`MODULE`**
- To locate packages, **`find_package()`** command looks
  - cmake project config files (**`CONFIG`** mode) = default
  - cmake find modules

Project config files : CMake

- Most projects ship package config files(.pc)
  - Info can be obtained via the pkg-config command
- CMake project config files allow cmake to identify an installed package in your system
- CMake package config files can be manually written or be generated from within a project.

Generating config file : CMake

```
add_library(projectLib STATIC ${projectLib_SRCS})
target_link_libraries(projectLib
    Qt5::Core
    Qt5::Network
)
install(TARGETS projectLib EXPORT projectExport DESTINATION ${LIB_DIR})

install(EXPORT projectExport DESTINATION ${CMAKECONFIG_DIR} FILE
projectTargets.cmake)
configure_file(projectConfig.cmake.in ${CMAKE_OUT_DIR}/projectConfig.cmake)
install(
    FILES ${CMAKE_OUT_DIR}/projectConfig.cmake
    DESTINATION ${CMAKECONFIG_DIR}
)
```

Find modules : CMake

Live demo

~/Scratch/devops/cmake/talks/calculator/cmake/modules/

Cool features

- CLI for `find_pacakge()`
  - Can be invoked from external scripts or buildsystems

- ```
  cmake --find-package -DNAME=Boost
  -DCOMPILER_ID=GNU -DLANGUAGE=C
  -DMODE=EXIST
  ```
  ```

Thanks!

R. Harish Navnit

harishnavnit@gmail.com