CMake 101

- Build systems exist to automate the compiling process.

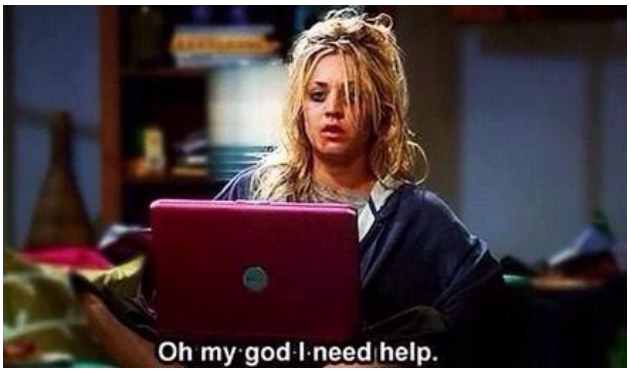
- Build automation involves scripting or automating the process of compiling computer source code into binary code.

1976

- Make was created in 1976

- Stuart Feldman (Bell Labs) was inspired to write Make by the experience of a coworker in futilely debugging a program of his where the executable was accidentally not being updated with changes. (wikipedia.org/wiki/Make_(software))

- “Make” figures out automatically which files it needs to update, based on which source files have changed.



- But Make became one of those projects that people love to hate. Developers were tired of writing Makefiles.

- Makefile was platform specific.

- Image: http://rebloggy.com/post/season-2-the-big-bang-theory-penny-same/40848778387

2000

- CMake is a scripting language that generates the Makefiles - that is where make will find the instructions on how to build your code.

- CMake development began in 1999 in response to the need for a cross-platform build environment for the Insight Segmentation and Registration Toolkit (ITK).[5] The project is funded by the United States National Library of Medicine as part of the Visible Human Project. (wikipedia.org/wiki/CMake)

The magic of CMake

- CMake recognizes which compiler to use for a given kind of source.
- CMake makes reasonable default choices depending on your system configuration.
- The language used to write CMakeLists.txt files is readable and easier to understand.

- Image: http://buffy.wikia.com/wiki/File:Willow_fire.png

Cross Platform

- Cross platform discovery of system libraries.

- Easier to compile your files into a shared library in a platform agnostic way, and in general easier to use than make. e.g.you could ship your code with 3rd party libraries and use CmakeLists.txt to build it and link it to your software.

- It supports multiple generators like Xcode, Eclipse, Visual Studio, etc. (A CMake Generator is responsible for writing the input files for a native build system. Exactly one of the CMake Generators must be selected for a build tree to determine what native build system is to be used.) e.g. one CmakeLists.txt that also runs on Linux can generate also Visual Studio files to be able to build in Windows.

C and C++

- CMake is for C and C++ specifically.

- “Make” is not limited to any particular language. For each non-source file in the program, the makefile specifies the shell commands to compute it. (can run a compiler to produce an object file, the linker to produce an executable, ar to update a library, or.. run npm install...)

- Now for some code...

- Image: http://www.oystermag.com/2015/09/looking-for-love-in-witchcraft-by-hazel-cills-for-oyster-107/

```
$ cd <source-folder>
$ vim CMakeLists.txt
```

Create a CMakeLists.txt file

```
$ g++ example_encode.cpp lodepng.cpp -std=c++11 -o example
```

- That was the command line to compile the lodepng example de from the last meetup – how to we do it with CMake?

```
cmake_minimum_required(VERSION 3.10)
project(loadpng)

set (LOADPNG_VERSION_MAJOR 0)
set (LOADPNG_VERSION_MINOR 1)

set(CMAKE_BUILD_TYPE Debug)

set(CMAKE_INSTALL_PREFIX ../install)

set(loadpng_src
   ${PROJECT_SOURCE_DIR}/lodepng.cpp
    ${PROJECT_SOURCE_DIR}/example_encode.cpp)

add_executable(loadpng ${loadpng_src})
```

- The basic to be able to compile a project.

```
# pass settings in the command line
cmake .. -DCMAKE_INSTALL_PREFIX=install

# detects the system – options: WIN32, APPLE and UNIX
if(WIN32)
	set(MY_SYSTEM "Windows detected :(")
endif(WIN32)

# finds packages/libs
find_package(Qt5 5.6 Core REQUIRED)

# check lib version
if (Qt5Core_VERSION VERSION_LESS 5.9.0)
	message(STATUS "For HTTP/2 support, compile with Qt 5.9 or higher.")
endif()
```

- Extras possibilities.

```
$ cd <source-folder>
$ mkdir build
$ cd build
$ cmake ..
$ ls
Makefile CMakeCache.txt cmake_install.cmake
$ make
```

- Setup your build environment.

- Cmake generated a bunch of files... but the main ones are in the root of the build folder:
Makefile, CmakeCache.txt and cmake_install.cmake

- then run make to compile and install

github.com/dynamic-cast/cmake-101

Links

CMake for Dummies:
https://www.youtube.com/watch?v=7W4Q-XLnMaA

CMake vs Make:
https://prateekvjoshi.com/2014/02/01/cmake-vs-make/