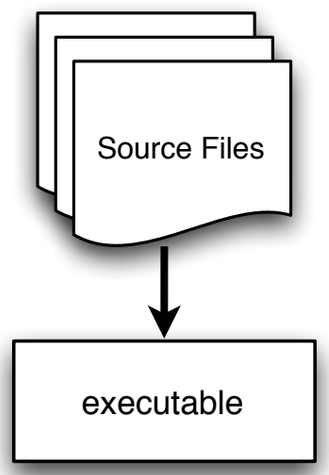
An Introduction to CMake

Scratching the Surface...

THE UNIVERSITY OF UTAH

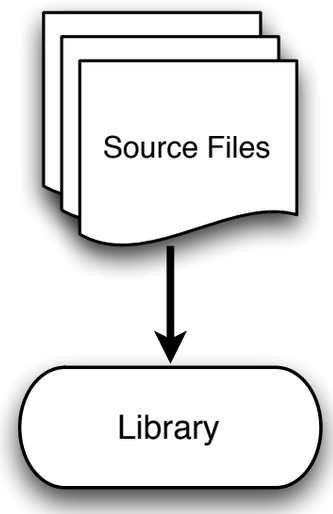
Common Tasks when Building

Compile one or more files into an executable



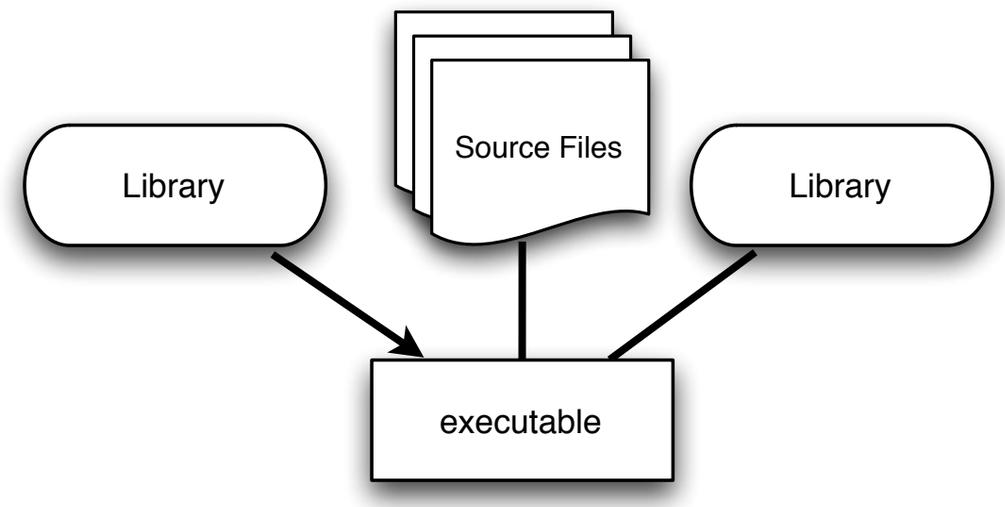
binary code that can be run on your system

Compile files to form a *library*



binary code

Compile files and *link* with libraries to get an executable



Options for "build systems"

- Maintain a full build system directly
  - Works okay for small projects that don't do too much fancy stuff if you are willing to dig into regularly
  - If you want to use different build systems (e.g. XCode, MS Visual Studio, etc.) then you must maintain a build system for each platform. YUCK!
- Auto-tools
  - Scripting level that generates makefiles - only useable on Unix/Linux like platforms
  - Quite awkward to use and a steep learning curve.
- CMake
  - High-level scripting language. Outputs build systems for various platforms.
  - Integrated support for testing and installing/packaging
  - very powerful - lots of capabilities!
  - Large and small projects alike are turning to CMake.
  - Download from http://www.cmake.org/cmake/resources/software.html

THE UNIVERSITY OF UTAH

Using CMake

- Create a "CMakeLists.txt" file in the directory containing your source files.
  - In this, tell CMake what you want to do
- Create a "build" directory where you will build the code.
  - Do NOT do this in the source directory.
- Run CMake in the build directory, and point to the source directory.
- Run "make" in the build directory.

THE UNIVERSITY OF UTAH

A Simple Example

```
#include <iostream>
using namespace std;
int main()
{
  cout << "Hello" << endl;
}
```

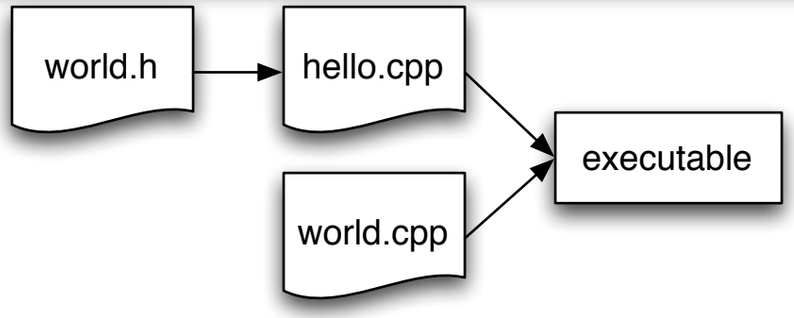
hello.cpp

```
# required: set the minumum version of
# CMake (for backward compatibility)
cmake_minimum_required( VERSION 2.8 )

# required: set a name for this project
project( Hello )

# optional: set the language(s) used in
# this project. CXX is the default.
# Also supports C and Fortran.
enable_language( CXX )

# create an executable.  First argument
# is the name. Remaining arguments are
# the source files that are to be
# compiled to form the executable.
add_executable( hello hello.cpp )
```

CMakeLists.txt

1. Create `hello.cpp`
2. Create `CMakeLists.txt`
3. Create a build directory anywhere.
4. Run "`cmake [path to src]`"
5. Run "`make`"

Multiple Source Files



hello.cpp

```
#include <iostream>
#include "world.h"
using namespace std;
int main()
{
  cout << "Hello" << endl;
  world(cout);
}
```

world.h

```
#include <iostream>
void world( std::ostream& );
```

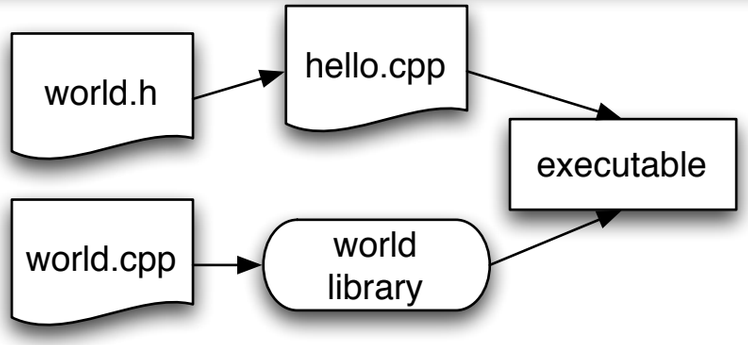
world.cpp

```
#include <iostream>
void world( std::ostream& os )
{
  os << " world!" << std::endl;
}
```

```
cmake_minimum_required( VERSION 2.8 )
project( HelloWorld )

# create an executable.  First argument
# is the name. Remaining arguments are
# the source files that are to be
# compiled to form the executable.
add_executable(
 hello
 hello.cpp
 world.cpp
 )
```

CMakeLists.txt

Creating & Linking Libraries



hello.cpp

```
#include <iostream>
#include "world.h"
using namespace std;
int main()
{
  cout << "Hello" << endl;
  world(cout);
}
```

world.h

```
#include <iostream>
void world( std::ostream& );
```

world.cpp

```
#include <iostream>
void world( std::ostream& os )
{
  os << " world!" << std::endl;
}
```

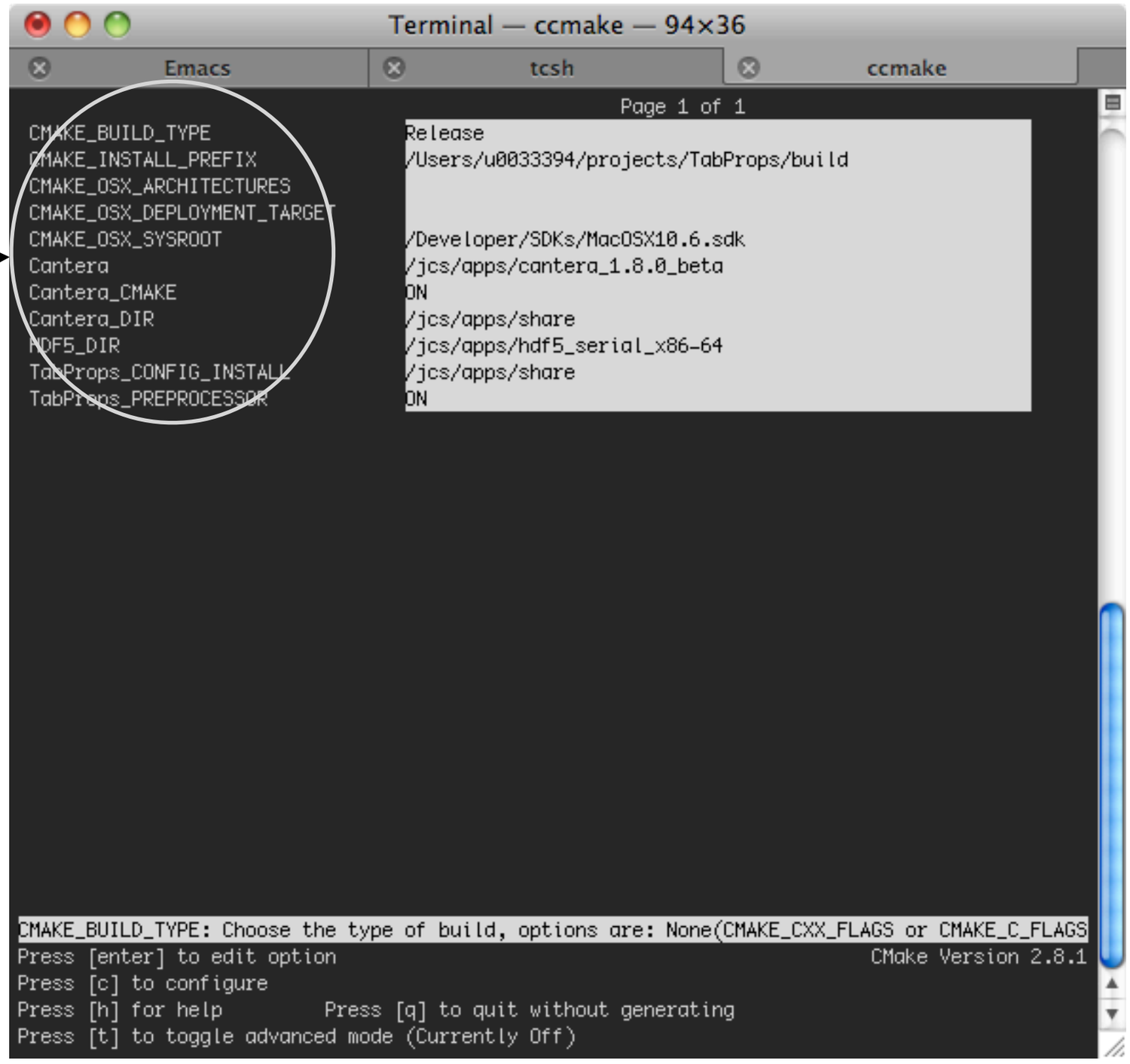
```
cmake_minimum_required( VERSION 2.8 )
project( HelloWorld )

add_library( world world.cpp )
add_executable(
 hello
 hello.cpp
 )

# link the “hello” executable with the
“world” library.
target_link_libraries(
  hello
  world
 )
```

CMakeLists.txt

CCMake

Things to pay attention to.

For simple cases (like our previous examples) there will not be very much here.



The CMake GUI

CMake 2.8.1 - /Users/u0033394/projects/TabProps/build

Where is the source code: /Users/u0033394/projects/TabProps [Browse Source...]

Where to build the binaries: /Users/u0033394/projects/TabProps/build [Browse Build...]

Search: [ ] Grouped View [Add Entry] [Remove Entry]

Name	Value
Ungrouped Entries	
CMAKE	
CPACK	
Cantera	
HDF5	
TabProps	

Press Configure to update and display new values in red, then press Generate to generate selected build files.

[Configure] [Generate] Current Generator: Unix Makefiles

```
Files will be installed to: /Users/u0033394/projects/TabProps/build/tabprops
Configuring done
```

THE UNIVERSITY OF UTAH

CMake Resources

- CMake wiki
- Online documentation
- The book (I have a copy)
- Me - I have done a lot of work with CMake and can help answer many of your questions.

THE UNIVERSITY OF UTAH