Advanced Programming Concepts with C++ CSI2372 – Fall 2016

Jochen Lang
EECS, University of Ottawa
Canada

Université d'Ottawa | University of Ottawa

uOttawa

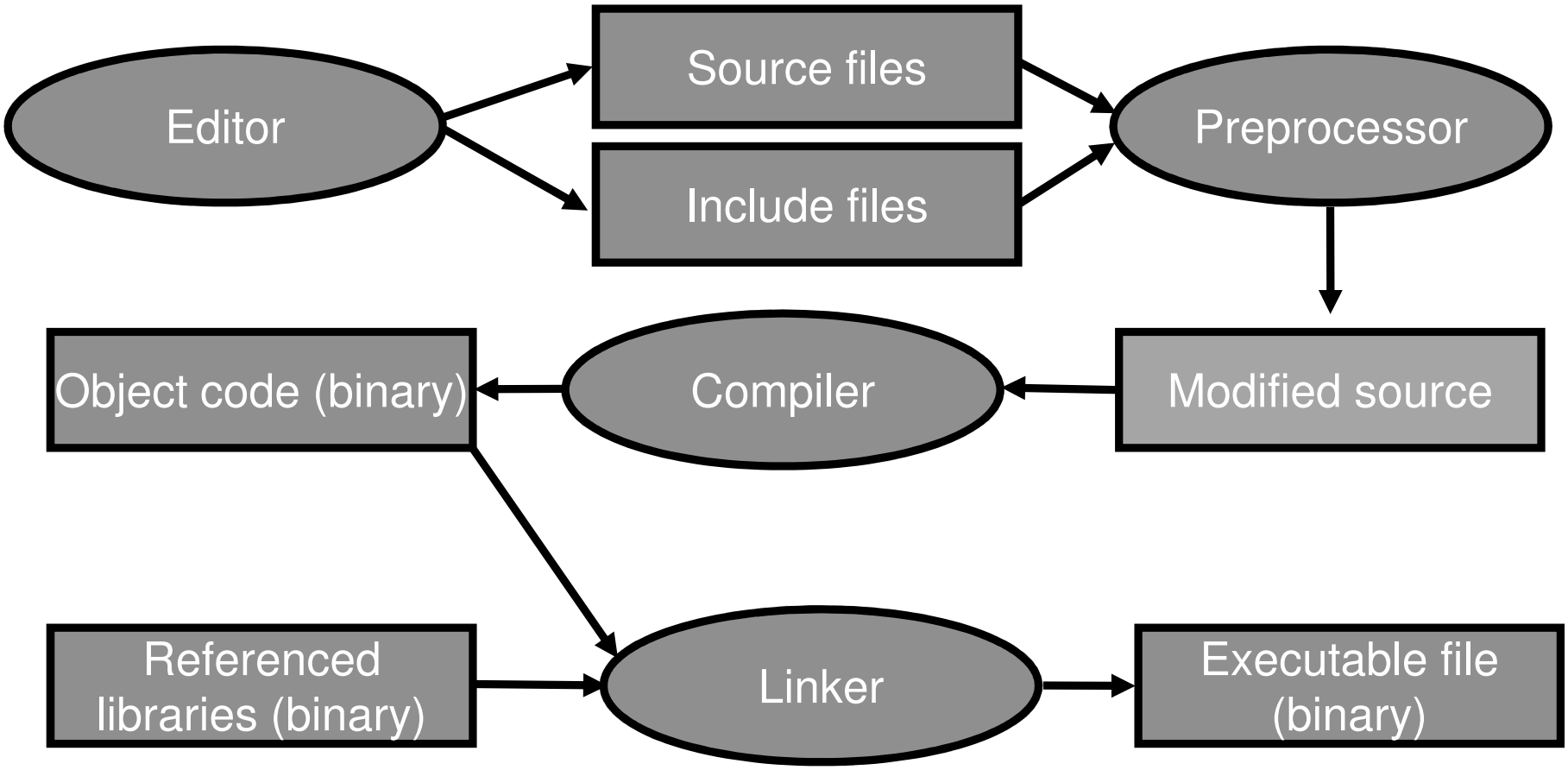
L'Université canadienne
Canada's university

uOttawa.ca

Introduction to C++ Laboratory

- **Use of development environment**
  - Microsoft Visual Studio 2013
  - But DevC++ (frontend to gcc) is also installed
- **A first C++ program**
  - Welcome to CSI2372
- **Porting of a Java program to C++**

Executable from C++



Development Environments

- **C++ Programming with an IDE**
  - Editor for creating the files
  - Preprocessor to expand macros, include files etc before compilation
  - Compiler to generate object code
  - Linker to bind object code and libraries resulting in an executable
  - Typically some kind of process automation, e.g., nmake
  - Debugger to analyze program executable

Visual C++ in Microsoft Visual Studio

- **Integrated Development Environment**
  - Combines all tools to create an executable into one tool
    - ✓ Seamless program creation
    - ✓ Provides extra tools for efficient program creation
    - × Obstructs portability and makes individual steps opaque

Welcome to CSI2372

- Please start Microsoft Visual Studio
- Select new project
- Select win32 console project
- Enter name and location; notice where the project will be created
- Wizard pops up
  - Look at application settings
  - Select empty project and deselect SDL

Welcome to CSI2372

- **Once inside the IDE**
  - Use “Hello World” example from Virtual Campus
  - Change the main function to print “Welcome to CSI2372” to cout
  - Compile and link the program
  - Run the program (hint: use debug start and debug start without debugging)
- **Outside the IDE**
  - Locate the program, e.g., *welcome.exe*, in windows explorer
  - Start a console (command prompt)
  - Run *welcome.exe*

Create a Release Version

- Go to solution explorer
- Right click on your project
- Select properties
- Study the options, note C/C++ and Linker configuration options
  - C/C++ controls how the compiler (and preprocessor) works
  - Linker controls the linking process
  - look at the command line options for both
- Change active configuration to release
- Compile
- Run from the command line

Visual C++ IDE Concepts

- **Solution**
  - Groups one or several related projects together
- **Project**
  - Groups all files related to one program together
- **Standard C/C++ filename extensions**
  - *.cpp are C++ file (also in use *.cxx, *.cc and .C); *.hh are header files (also *.hpp, *.hxx, *.h)
  - *.c and *.h are C files (Note that case sensitivity is sketchy in Windows which makes *.C also a C file but *.C is a C++ file for others).

More Visual C++ IDE Concepts

- **Standard object code, executable location**
  - Debug directory in directory *projName*
  - Release directory in directory *projName*
- **Project options can be set separately for debug and release**
  - Debug code works usually better for debugging
  - Release code can be significantly faster
- **Precompiled headers**
  - Allows you to not recompile headers in order to reduce recompilation time. (Extension *.pch)
  - Feature is automated in IDE; may fail

Files in a VC++ Project

- **projname.cpp (other names possible)**
  - Entry point if main is present.
- **projname.h (other names possible)**
  - Declarations, global symbols, and #include directives for all headers
- **projname.sdf**
  - Browsing database file used by Class View
- **projname.sln, projname.suo**
  - Solution file for IDE and solution options file, respectively
- **projname.vcxproj**
  - Project file for IDE
- **Readme.txt**
  - Generated text file about the project
- **Many others**
  - Related to different OS interfaces of the project

Towards a Standard C++ Project

- Make sure no
  - stdafx.h, stdafx.cpp

  are included in your project
- In main cpp file
  - Make sure program entry point is `int main(void)`
  - Include necessary library header file
- Select
  - not using precompiled headers

Java Program to C/C++

  - Download Java program `InputOut.java` from Virtual Campus
  - Create a new project “InputOut” (same as before for welcome)
  - Create a new source file
  - Compile and run
- **Instructions**
  - Use only global functions for now
  - Use only a single file
  - Keep java and cpp sources similar

Java Program to C/C++

- **Hints**
  - There is no built-in String type in C++ but the standard template library type string (Java and C++ are case sensitive)
    - You will need to include the library header `<string>`
  - Global methods in C++ (best wrapped into a namespace) closely match the functionality of static methods in Java (Note: There are also static member functions in C++).
  - C++ uses the input and output stream classes from the standard template library to input/output to/from console. `std::coutout` and `std::cin` in the library header `<iostream>`)

Change to Multiple Files

- **Task: Split InputOut.cpp in three separate files, each with the corresponding function(s):**
  - main.cpp
  - read_input.cpp
  - show.cpp
- **What do you need to do for the program to build?**

Hints

- **Each cpp file by itself forms a compilation unit**
  - All names need to be known to the compiler
- **Standard Solution: Create separate header files**
  - read_input.h
  - show.h
- **Include the header file wherever the function is needed**

Next Laboratory

– Lab assignment 1