CS 50 : C Programming

Jinan Darwiche

Topics

- Welcome to class
- Computer languages
- The compilation process·
- The history of C
- Developing programs with C
- Helloworld.Cpp

Welcome to Class

– Each unit contains a slides file, reading materials, an assignment or a quiz, and Discussions
– The slides are your starting point of the learning process
– Check the due dates of the assignments, quizzes and or Discussion using our course site
– Ask questions in the Discussion if you have them; you may also answer others' questions − The Discussion is our "meeting" place. Check it daily

Programming: Why do it?

- Processor based devices such as desktops/ laptops, “smart” phones, electronic pads, even some cars, TV’s, etc execute instructions, called machine language software.
- Instructions are written in a programming language of your choice. This is called Source Code.
- Another software, called Compiler, changes the source code into Assembly/Machine Code.

Computer languages

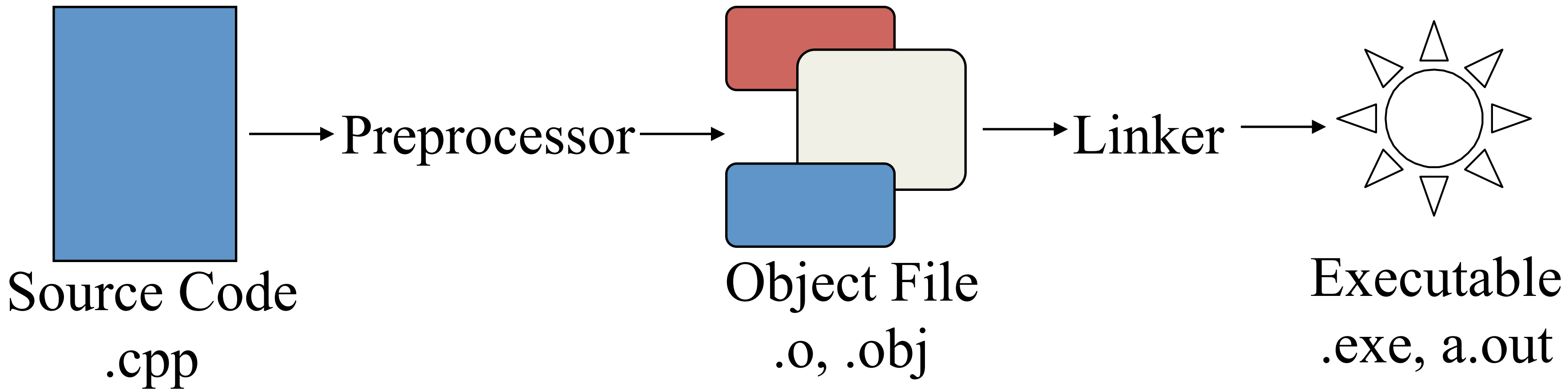
- Computer languages have evolved over time
- Initially, programmers coded in machine language : 01010110 0001000
- Eventually, assemblers were made to hide machine language behind mnemonic instruction: ADD R1, 8

High-Level Languages

- C offers convenient “high-level” language features with access to low-level hardware primitives
- Languages are interpreted or compiled
- C is a compiled language

Compiled Languages

- Compiled languages must be turned into executable computer instructions



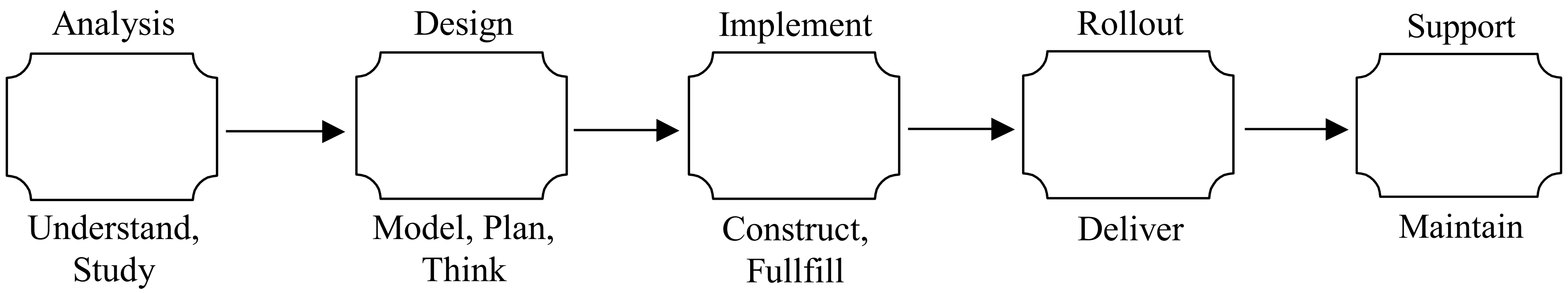
- Errors can occur at each step!
  - Compile-time, linkage, run-time

The History of C

- Authored by Dennis Ritchie, AT&T bell labs
- Originally, C was a programming tool to help in creating the UNIX operating system for the DEC PDP-11 computer
- Language is now an international standard
- Read the **book sections 1 and 2.**

Developing Programs With C

- Program Development Methodology



- Our initial programs won't require much analysis or design

Visual Studio .NET 2012

- We will use the Visual Studio compiler
- Visual Studio .NET 2012, let’s call it VS contains several compilers, including C++.
- C++ is a language that supersedes C. We can use it to write and compile C code.
- If you have not done so already, follow the “Setup Slides” to install VS on your machine. Other compiler versions are NOT accepted.

Purpose of Software

- Most of the software/programs we will learn and write will follow the same pattern:
  - Get input
  - Process the input to product some result
  - Show output − usually the result produced in the previous step

Learning Programming

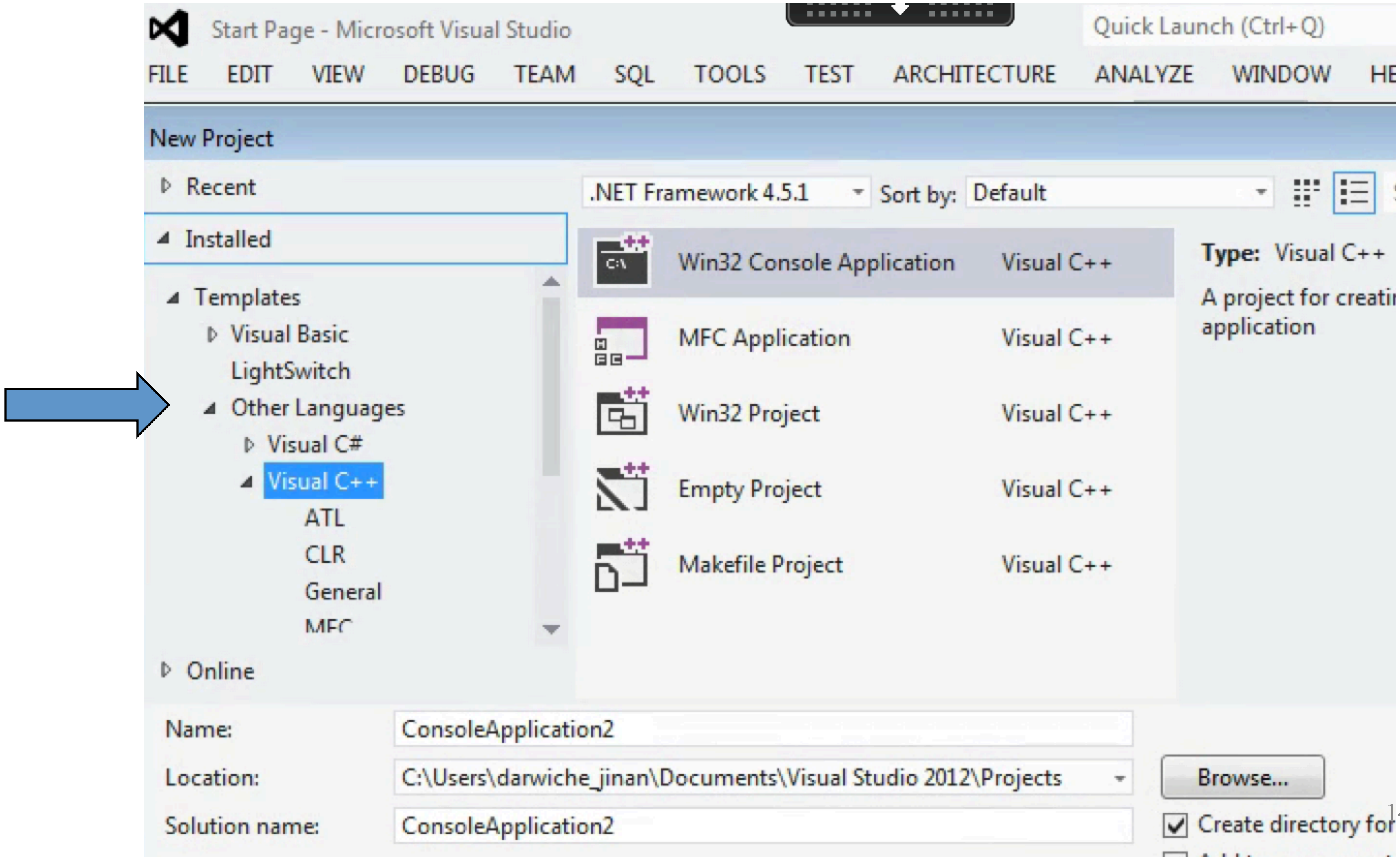
- Analyzing existing code helps in understanding how to write code
- In the early stages, you may not understand every line − focus on the main concepts
- Syntax errors are very common − be patient

To Learn Programming You Learn

- The **syntax**: What words and how to put them together to form sentences
  - Words are referred to as **keywords**, and sentences are referred to as **statements**
  - **Each Programming language has its own syntax**
- The **semantics:** How to form statements to solve a particular problem.
  - Semantics are universal − not language dependent

Time For Our First Demo: Hello World

- Start VS 2012 then click New Project. You will get:


Start Page - Microsoft Visual Studio

Quick Launch (Ctrl+Q)

FILE EDIT VIEW DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDOW HE

New Project

- Recent
- Installed
  - Templates
    - Visual Basic
    - LightSwitch
    - Other Languages
      - Visual C#
      - Visual C++
        - ATL
        - CLR
        - General
        - MFC
- Online

.NET Framework 4.5.1 Sort by: Default

Win32 Console Application	Visual C++
MFC Application	Visual C++
Win32 Project	Visual C++
Empty Project	Visual C++
Makefile Project	Visual C++

Type: Visual C++

A project for creati
application

Name: ConsoleApplication2

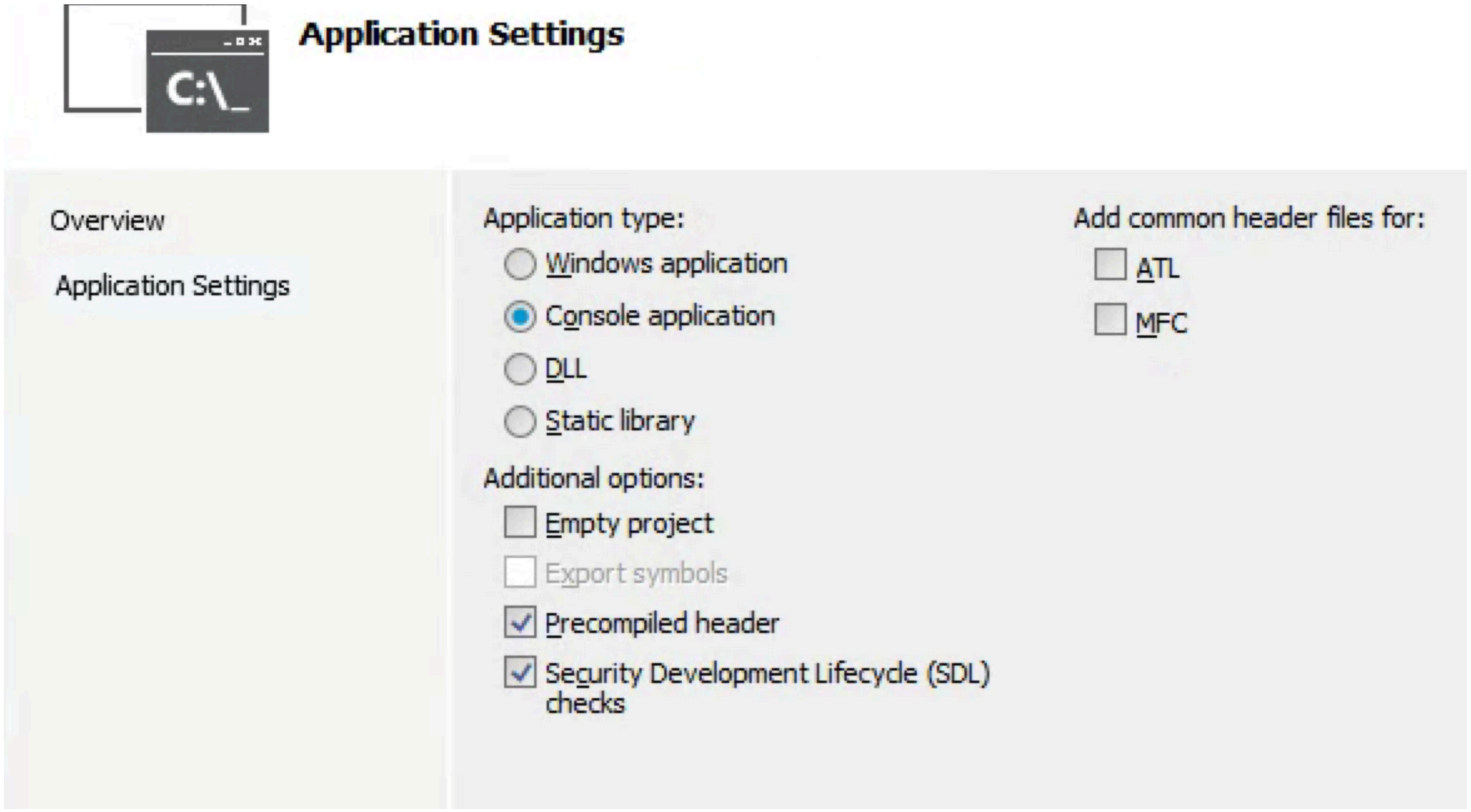
Location: C:\Users\darwiche_jinan\Documents\Visual Studio 2012\Projects

Browse...

Solution name: ConsoleApplication2

☑ Create directory for

First time you get the setup wizard - Choose Console Application


Application Settings

Overview

Application Settings

Application type:

- ○ Windows application
- ● Console application
- ○ DLL
- ○ Static library

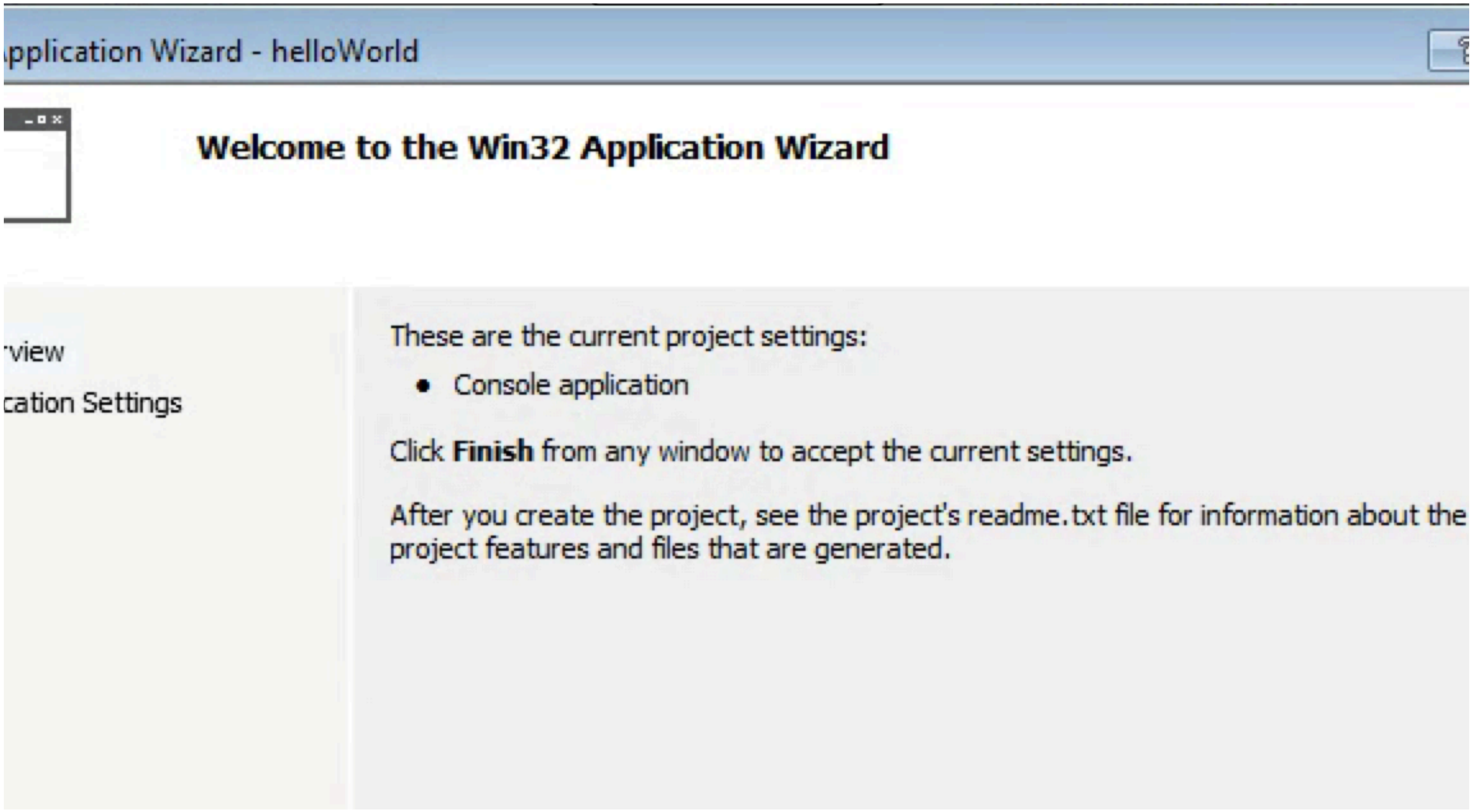
Additional options:

- ☐ Empty project
- ☐ Export symbols
- ☑ Precompiled header
- ☑ Security Development Lifecycle (SDL) checks

Add common header files for:

- ☐ ATL
- ☐ MFC

Click Finish



Welcome to the Win32 Application Wizard

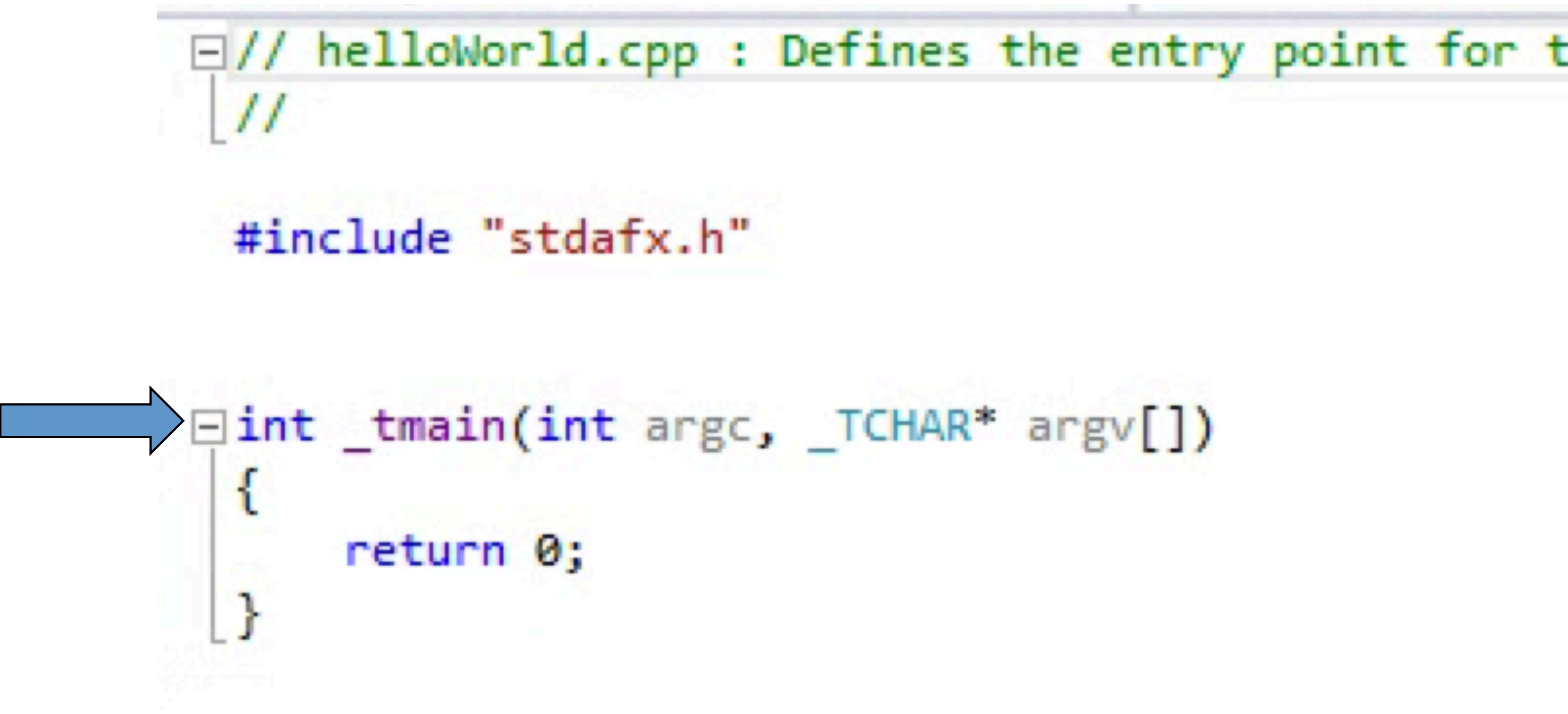
These are the current project settings:

- Console application

Click **Finish** from any window to accept the current settings.

After you create the project, see the project's readme.txt file for information about the project features and files that are generated.

You will get a “Blank Application”
Minimize the existing code
Click -



```
// helloWorld.cpp : Defines the entry point for t
//

#include "stdafx.h"


int _tmain(int argc, _TCHAR* argv[])
{
	return 0;
}
```

What is Hello World

- A starting point to learn how to show output.
- Code is typically made of one statement to show the words “hello world” as output.

Write the Code

#include "stdafx.h" ← this line already exists

#include "stdio.h"

int main(void)

{

printf("Hello World!");

return 0;

}

Understanding the Code

#include "stdafx.h” ← this line already exists

#include "stdio.h” ← this line is needed so we can show output and get input - Lines with # go together at the top

← You can leave blank lines to make the code more readable. This is called white space

int main(void) ← this is the starting point of executing the code

{ ← after main, you must enclose your code inside curled brackets. This is called code block

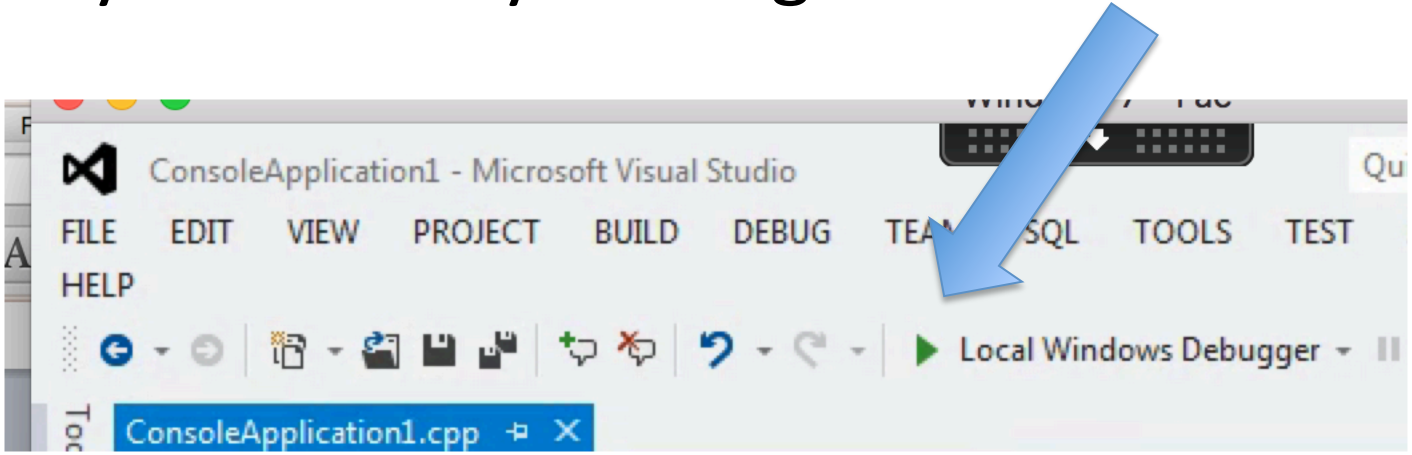
printf("Hello World!"); ← this is a statement – every statement must end with a semi-colon

return 0; ← every function must "return" a value. We will discuss this in more details

} ← every opened curled bracket must be closed with a closing curled bracket.

After you type the code

- Click the Debug button (green play icon)
- You must always test if your code is syntax error free and executes without errors.
- If you get an error, you must fix it
- Run your code by clicking:



When you run….

- The code we covered for Hello World, runs, but you don’t get to “see” it
- The runtime Console screen appears then quickly disappears
- To make the Console stay, we need an input statement
- After the printf line, add: scanf_s("press any key");

C Code components

- Include statements
- main function
- Statements inside functions
- Comments: are lines that explain the code but the compiler does not translate to machine language.
- Comments can be single-line: // or extend over multiple lines using /* */
- Comments help programmers understand the code later when they read it

Output in C

- C output statement: printf(“Hello World\n”);
  - sends information from program to terminal screen - this is what VS2012 refers to as the Console(standard output)
  - double quotes "..." delimit a string
  - \n sends a new-line-character − this is optional

- See book section 5 – note the differences between the book and our code-
- Skip to section 6 (The classical hello world program).

Overview of C and Programming

- Most programs get input from the user
- Input in code is saved into Variables
- Input is processed to produce some result
- Results are also saved in vartiables
- A variable is a location in RAM, marked by its name, and what type of information it can contain.

- Read the book sections 8, 9 and 10.
- You may ignore sections marked with links.

Next Unit

- We learn how to get input, process it then show output
- This requires the use of variables, input and output statements
- Review the slides after reading the book
- Post questions in the Discussion
- Complete the quiz/assignment