GETTING STARTED WITH

Steve Jones
www.gameinstitute.com

TABLE OF CONTENTS

TABLE OF FIGURES

INTRODUCTION

This document describes, step-by-step, how to install and setup Visual C++ Community 2019 which is one of the free versions of Microsoft Visual Studio. After showing how to install the software you will then learn how to create your first C++ program. The sample program is also found in *C++ Programming for Games Module I* course textbook or you can type or copy it directly from this document. Let us get started first by downloading and installing Microsoft Visual Studio 2019 Community.

INSTALLING VISUAL STUDIO COMMUNITY 2019

Download your copy of Visual Studio 2019 Community from Microsoft's website.
https://www.visualstudio.microsoft.com/vs/community/

Click the blue download button to download the installer file **vs_community__<big number>.exe.**

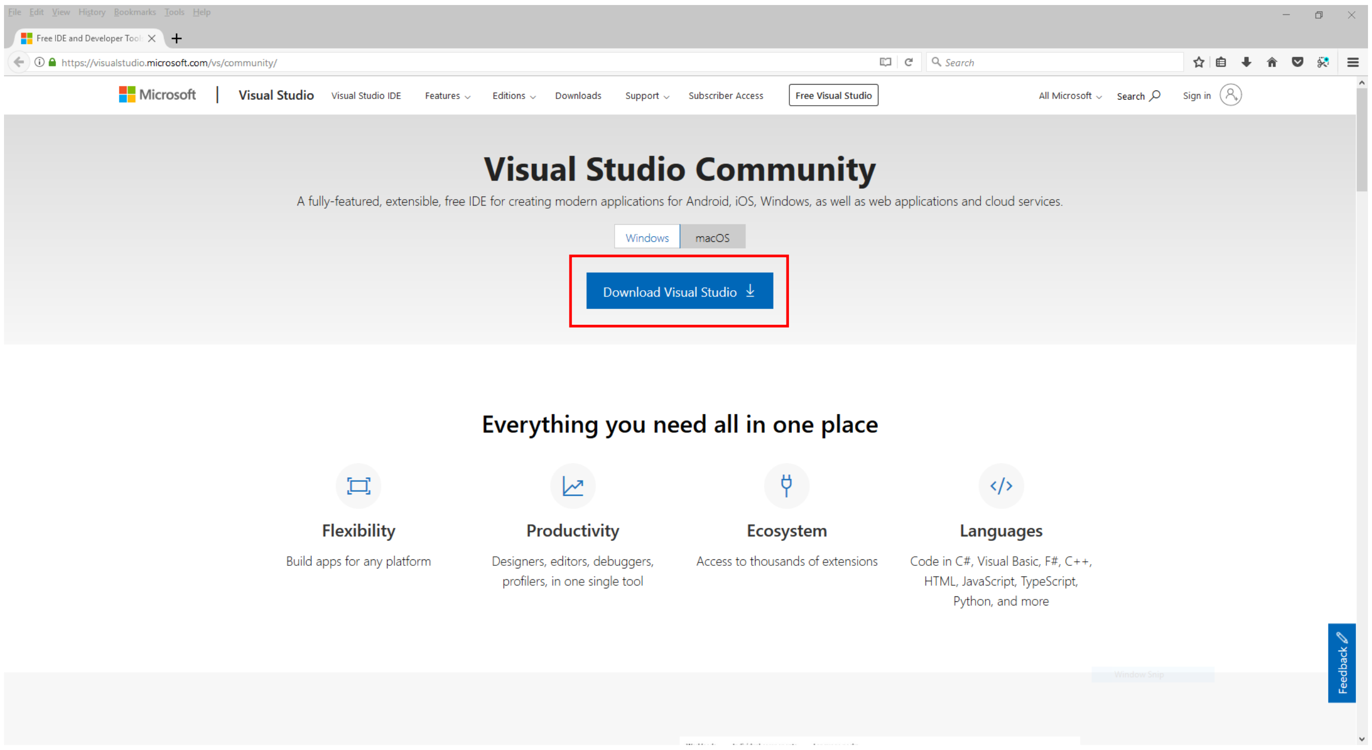
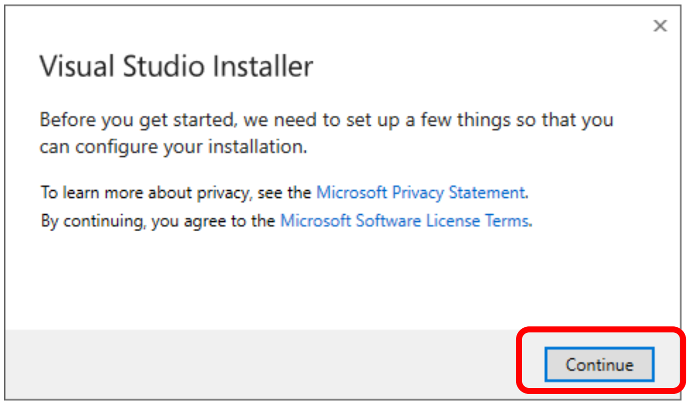

Visual Studio Community
A fully-featured, extensible, free IDE for creating modern applications for Android, iOS, Windows, as well as web applications and cloud services.
Windows
macOS
Download Visual Studio
Everything you need all in one place
Flexibility
Build apps for any platform
Productivity
Designers, editors, debuggers, profilers, in one single tool
Ecosystem
Access to thousands of extensions
Languages
Code in C#, Visual Basic, F#, C++, HTML, JavaScript, TypeScript, Python, and more

Figure 1. Visual Studio Community 2019 download web page.

Let's get started installing Visual Studio Community 2019!

INSTALLATION PROCEDURES

Launch the executable **vs_community__<big number>.exe** that was downloaded from the Microsoft website. The first screen you will see is the installer loading dialog. Click **Continue**.

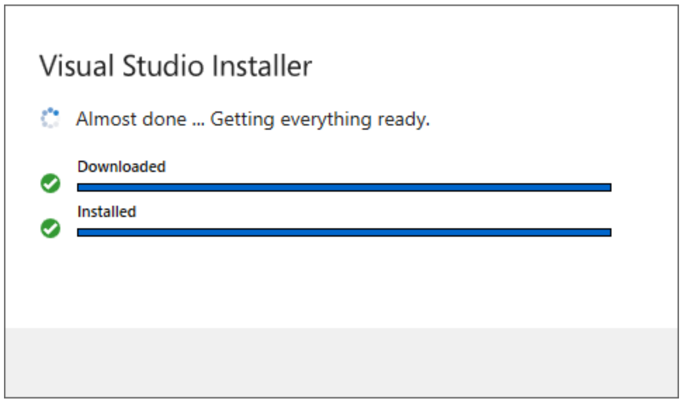


Visual Studio Installer

Before you get started, we need to set up a few things so that you can configure your installation.

To learn more about privacy, see the Microsoft Privacy Statement.
By continuing, you agree to the Microsoft Software License Terms.

Figure 2. Visual Studio Community 2019 installer loading dialog



Visual Studio Installer

Almost done ... Getting everything ready.

Downloaded

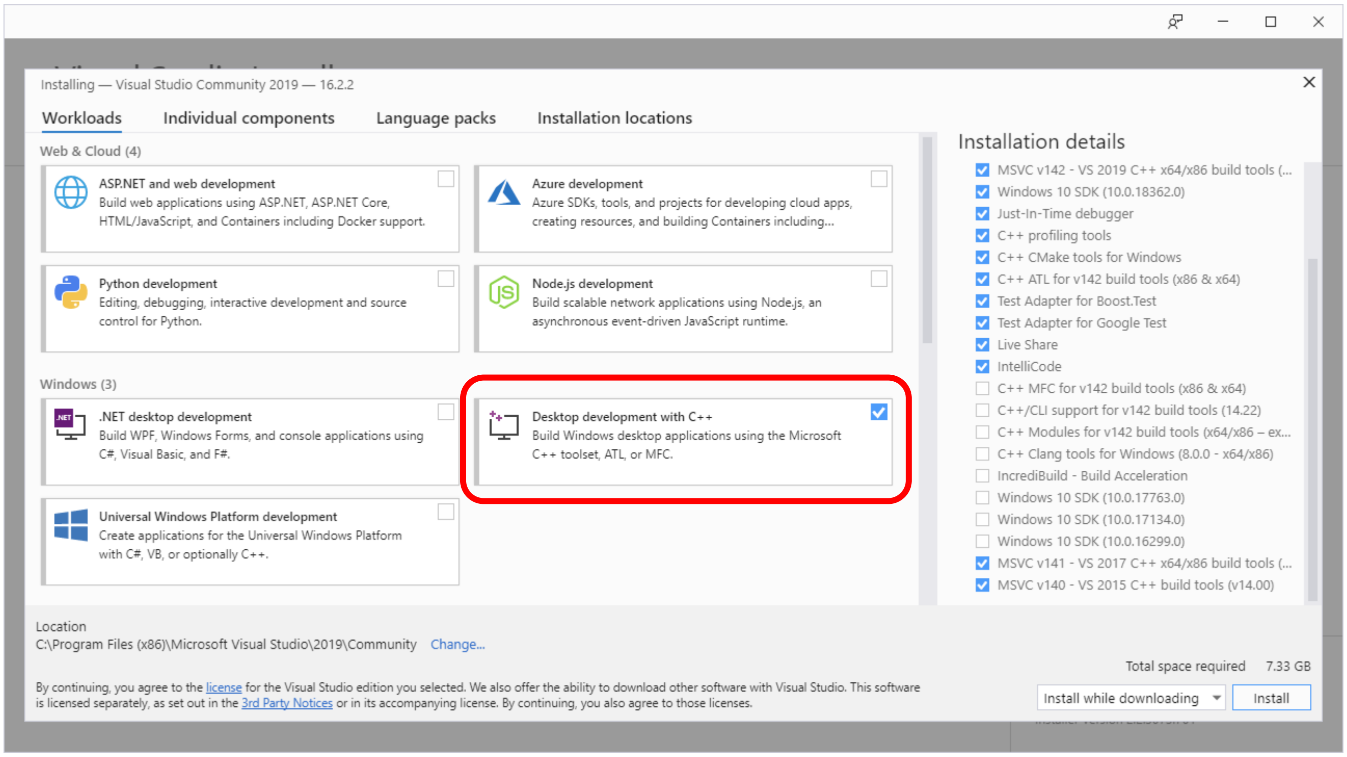
Installed

Figure 3. Installation process loading screen.

After the Visual Studio Installer window loads there will be many types of Visual Studio installations and options to select. The packaged ones which include whole languages like C++ and C# are called Workloads. For this course you will need the **Desktop development with C++** Workload type. You can select others if you like but they will take up more space on your hard drive and this course will not need them.

Select **Desktop development with C++** checkbox.

There are more options to select for the type Workload that was chosen. The scroll bar can show more options that you may want to install. Note that you can always run the installer again to *modify* your installation by adding or removing options and workloads later on if you choose.



Installing — Visual Studio Community 2019 — 16.2.2

Web & Cloud (4)

- ☐ **ASP.NET and web development** — Build web applications using ASP.NET, ASP.NET Core, HTML/JavaScript, and Containers including Docker support.
- ☐ **Azure development** — Azure SDKs, tools, and projects for developing cloud apps, creating resources, and building Containers including...
- ☐ **Python development** — Editing, debugging, interactive development and source control for Python.
- ☐ **Node.js development** — Build scalable network applications using Node.js, an asynchronous event-driven JavaScript runtime.

Windows (3)

- ☐ **.NET desktop development** — Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F#.
- ✓ **Desktop development with C++** — Build Windows desktop applications using the Microsoft C++ toolset, ATL, or MFC.
- ☐ **Universal Windows Platform development** — Create applications for the Universal Windows Platform with C#, VB, or optionally C++.

Installation details

- ✓ MSVC v142 - VS 2019 C++ x64/x86 build tools (...
- ✓ Windows 10 SDK (10.0.18362.0)
- ✓ Just-In-Time debugger
- ✓ C++ profiling tools
- ✓ C++ CMake tools for Windows
- ✓ C++ ATL for v142 build tools (x86 & x64)
- ✓ Test Adapter for Boost.Test
- ✓ Test Adapter for Google Test
- ✓ Live Share
- ✓ IntelliCode
- ☐ C++ MFC for v142 build tools (x86 & x64)
- ☐ C++/CLI support for v142 build tools (14.22)
- ☐ C++ Modules for v142 build tools (x64/x86 – ex...
- ☐ C++ Clang tools for Windows (8.0.0 - x64/x86)
- ☐ IncrediBuild - Build Acceleration
- ☐ Windows 10 SDK (10.0.17763.0)
- ☐ Windows 10 SDK (10.0.17134.0)
- ☐ Windows 10 SDK (10.0.16299.0)
- ✓ MSVC v141 - VS 2017 C++ x64/x86 build tools (...
- ✓ MSVC v140 - VS 2015 C++ build tools (v14.00)

Location
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community Change...

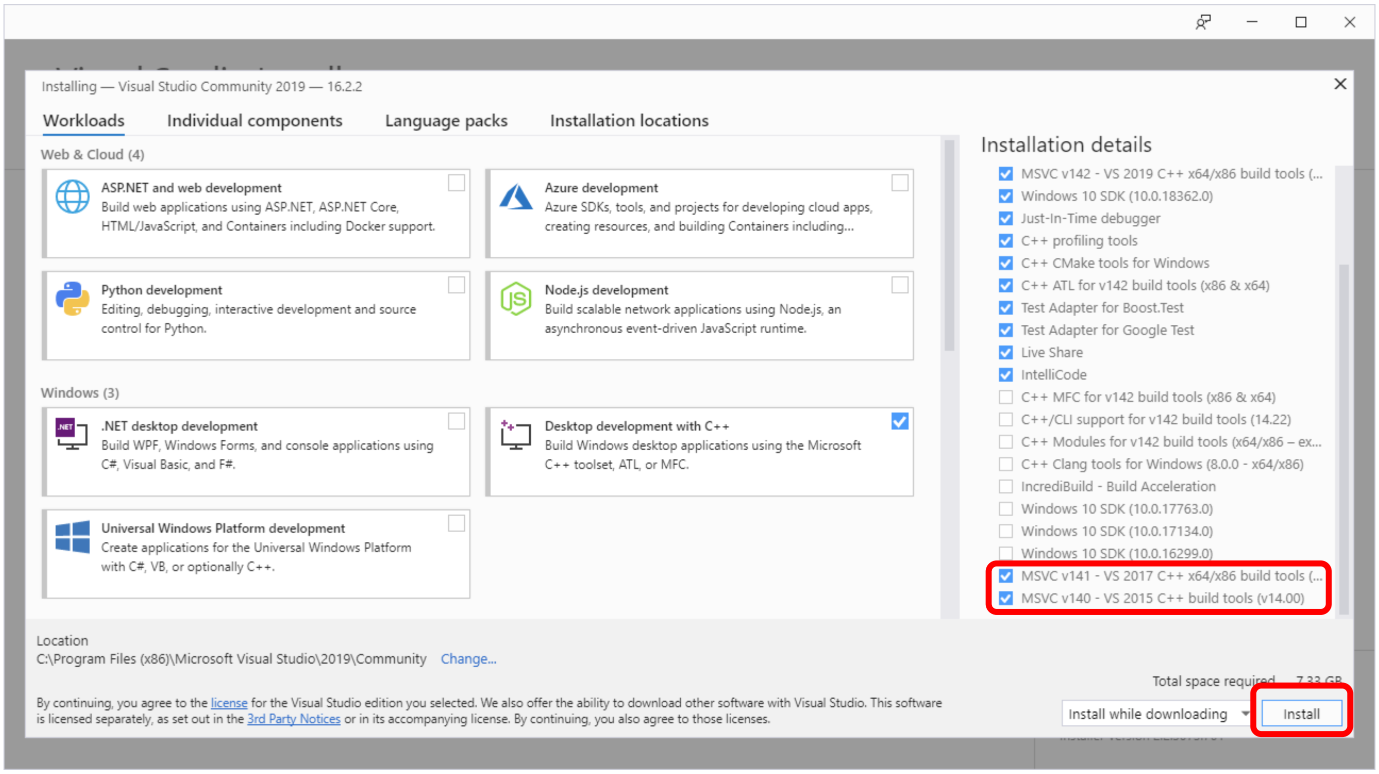
By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Total space required 7.33 GB

Figure 4. Select Desktop development with C++ for this course.

A summary panel on the left will show the options selected for installation.

You need to select option **MSVC v141 – VS 2017 C++ x64/x86 build tools** and **MSVC v140 – VS 2015 C++ build tools (v14.00)** options for building the C++ Module II Windows applications. This option is not required for C++ Module I Console-based applications, however, but it is fine to install it anyway if you only plan to take C++ Module I. You may need to scroll down to see them.



Installing — Visual Studio Community 2019 — 16.2.2

Workloads | Individual components | Language packs | Installation locations

Web & Cloud (4)

- [ ] **ASP.NET and web development** — Build web applications using ASP.NET, ASP.NET Core, HTML/JavaScript, and Containers including Docker support.
- [ ] **Azure development** — Azure SDKs, tools, and projects for developing cloud apps, creating resources, and building Containers including...
- [ ] **Python development** — Editing, debugging, interactive development and source control for Python.
- [ ] **Node.js development** — Build scalable network applications using Node.js, an asynchronous event-driven JavaScript runtime.

Windows (3)

- [ ] **.NET desktop development** — Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F#.
- [✓] **Desktop development with C++** — Build Windows desktop applications using the Microsoft C++ toolset, ATL, or MFC.
- [ ] **Universal Windows Platform development** — Create applications for the Universal Windows Platform with C#, VB, or optionally C++.

Installation details

- [✓] MSVC v142 - VS 2019 C++ x64/x86 build tools (...
- [✓] Windows 10 SDK (10.0.18362.0)
- [✓] Just-In-Time debugger
- [✓] C++ profiling tools
- [✓] C++ CMake tools for Windows
- [✓] C++ ATL for v142 build tools (x86 & x64)
- [✓] Test Adapter for Boost.Test
- [✓] Test Adapter for Google Test
- [✓] Live Share
- [✓] IntelliCode
- [ ] C++ MFC for v142 build tools (x86 & x64)
- [ ] C++/CLI support for v142 build tools (14.22)
- [ ] C++ Modules for v142 build tools (x64/x86 – ex...
- [ ] C++ Clang tools for Windows (8.0.0 - x64/x86)
- [ ] IncrediBuild - Build Acceleration
- [ ] Windows 10 SDK (10.0.17763.0)
- [ ] Windows 10 SDK (10.0.17134.0)
- [ ] Windows 10 SDK (10.0.16299.0)
- [✓] MSVC v141 - VS 2017 C++ x64/x86 build tools (...
- [✓] MSVC v140 - VS 2015 C++ build tools (v14.00)

Location
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community Change...

By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Total space required 7.33 GB

Install while downloading | Install

Figure 5. Select options included in the installation.

The software that will be installed is shown in the Summary. Click **Install** to start the installation.

Figure 6 shows the progress of the installation. The installation will take many minutes so be patient!

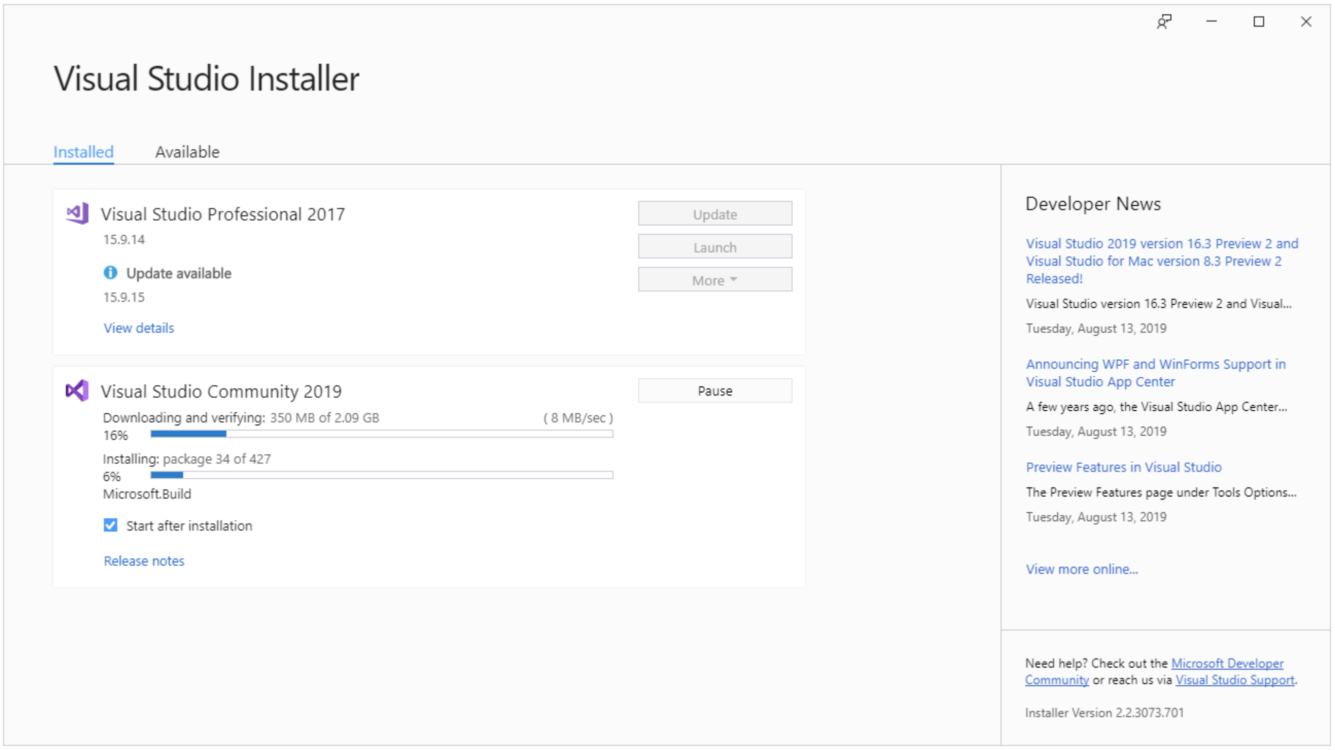
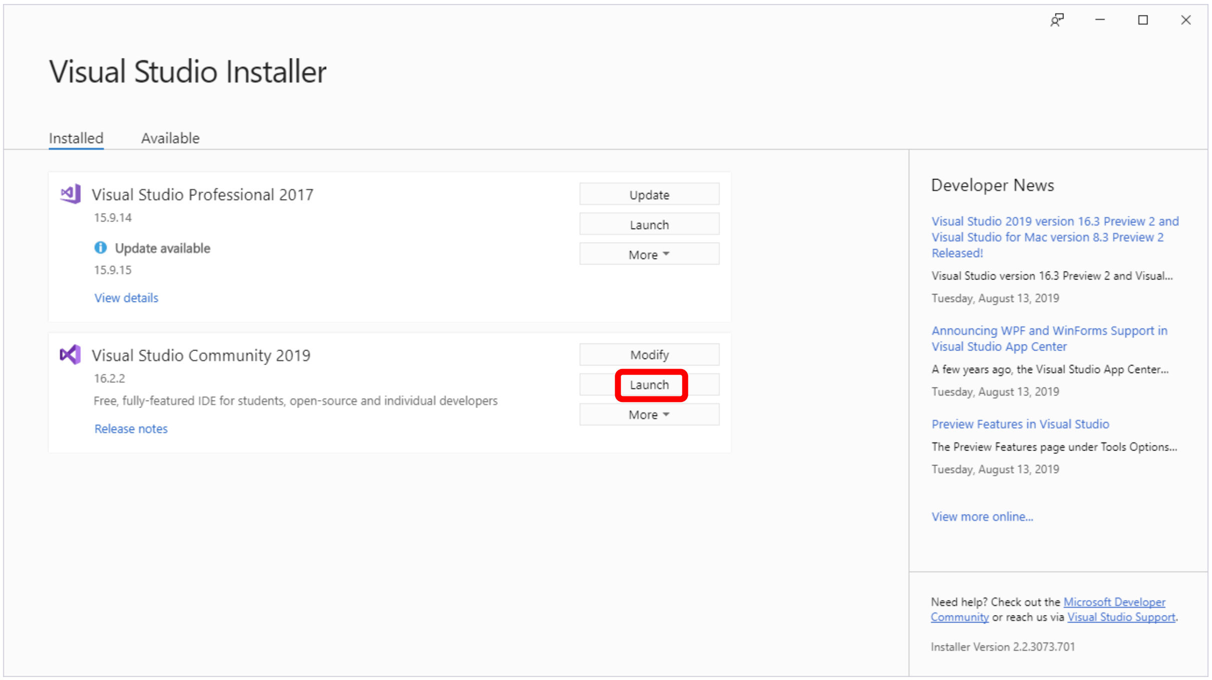

Visual Studio Installer
Installed
Available
Visual Studio Professional 2017
15.9.14
Update available
15.9.15
View details
Update
Launch
More
Visual Studio Community 2019
Downloading and verifying: 350 MB of 2.09 GB
(8 MB/sec)
16%
Installing: package 34 of 427
6%
Microsoft.Build
Start after installation
Release notes
Pause
Developer News
Visual Studio 2019 version 16.3 Preview 2 and Visual Studio for Mac version 8.3 Preview 2 Released!
Visual Studio version 16.3 Preview 2 and Visual...
Tuesday, August 13, 2019
Announcing WPF and WinForms Support in Visual Studio App Center
A few years ago, the Visual Studio App Center...
Tuesday, August 13, 2019
Preview Features in Visual Studio
The Preview Features page under Tools Options...
Tuesday, August 13, 2019
View more online...
Need help? Check out the Microsoft Developer Community or reach us via Visual Studio Support.
Installer Version 2.2.3073.701

Figure 6. Installing Visual Studio Community 2019.

When finished you can choose to launch Visual Studio 2019 from this installer window or from the menu short cut that will be created. Going forward you do not have to run the VS installer in order to run Visual Studio. The Launch button is presented here for convenience.

There will be an application menu item added. You can start Visual Studio here by clicking **LAUNCH** or find the icon in the Start menu.



Visual Studio Professional 2017
15.9.14
Update available
15.9.15

Visual Studio Community 2019
16.2.2
Free, fully-featured IDE for students, open-source and individual developers

Developer News

Visual Studio 2019 version 16.3 Preview 2 and Visual Studio for Mac version 8.3 Preview 2 Released!
Visual Studio version 16.3 Preview 2 and Visual...
Tuesday, August 13, 2019

Announcing WPF and WinForms Support in Visual Studio App Center
A few years ago, the Visual Studio App Center...
Tuesday, August 13, 2019

Preview Features in Visual Studio
The Preview Features page under Tools Options...
Tuesday, August 13, 2019

Need help? Check out the Microsoft Developer Community or reach us via Visual Studio Support.

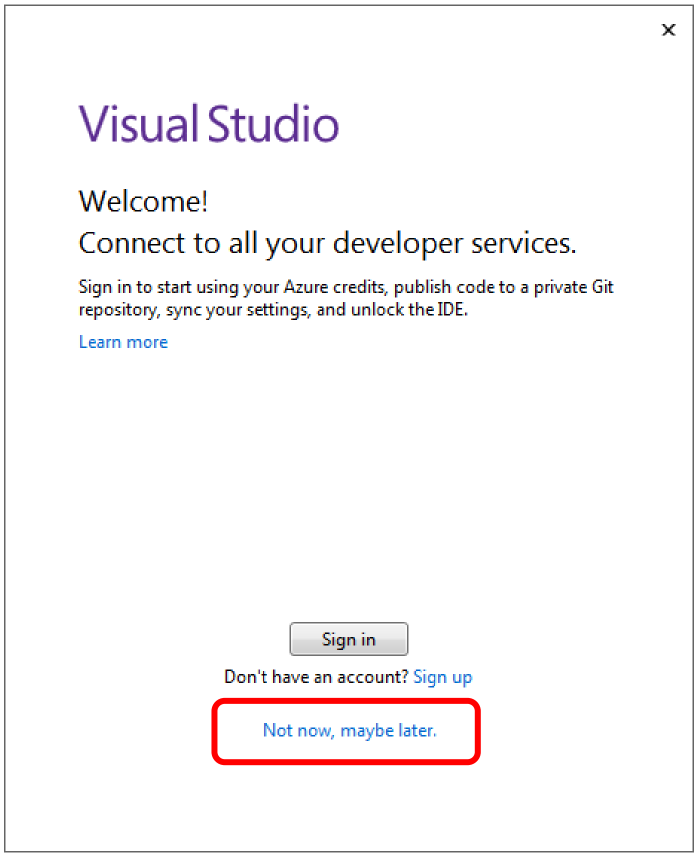
Installer Version 2.2.3073.701

Figure 7. Visual Studio Community 2019 installation complete.

That's it! We've just installed Visual Studio 2019 Community Edition which is all that is necessary for the C++ Module I and Module II courses.

Click the **Launch** button on the Visual Studio Installer window.

When you run for the first time you will be presented with a Welcome login screen (Figure 8). If you already have a Microsoft account, you can sign in using that account or you can click Sign up to create an account. You will need an account for the free Community license. We will assume you have an account already so click **"Not now, maybe later"** link to continue.


Visual Studio

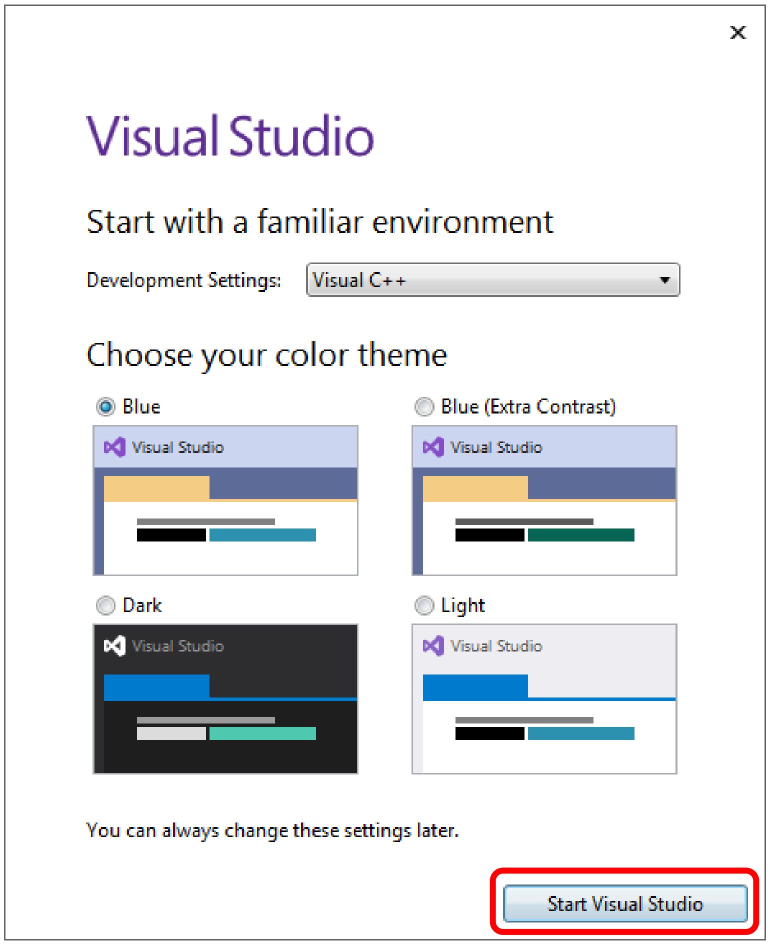
Welcome!

Connect to all your developer services.

Sign in to start using your Azure credits, publish code to a private Git repository, sync your settings, and unlock the IDE.

Figure 8. Login screen.

The next screen shown in Figure 9 is will allow you to select environment settings such as the color theme and development settings. The Dark theme is particularly easy on the eyes so that is what was chosen for this guide. Note that you can always go back and change these settings after Visual Studio has been installed. This dialog is for convenience to set up a few common styles ahead of time.


Visual Studio

Start with a familiar environment

Development Settings: Visual C++

Choose your color theme

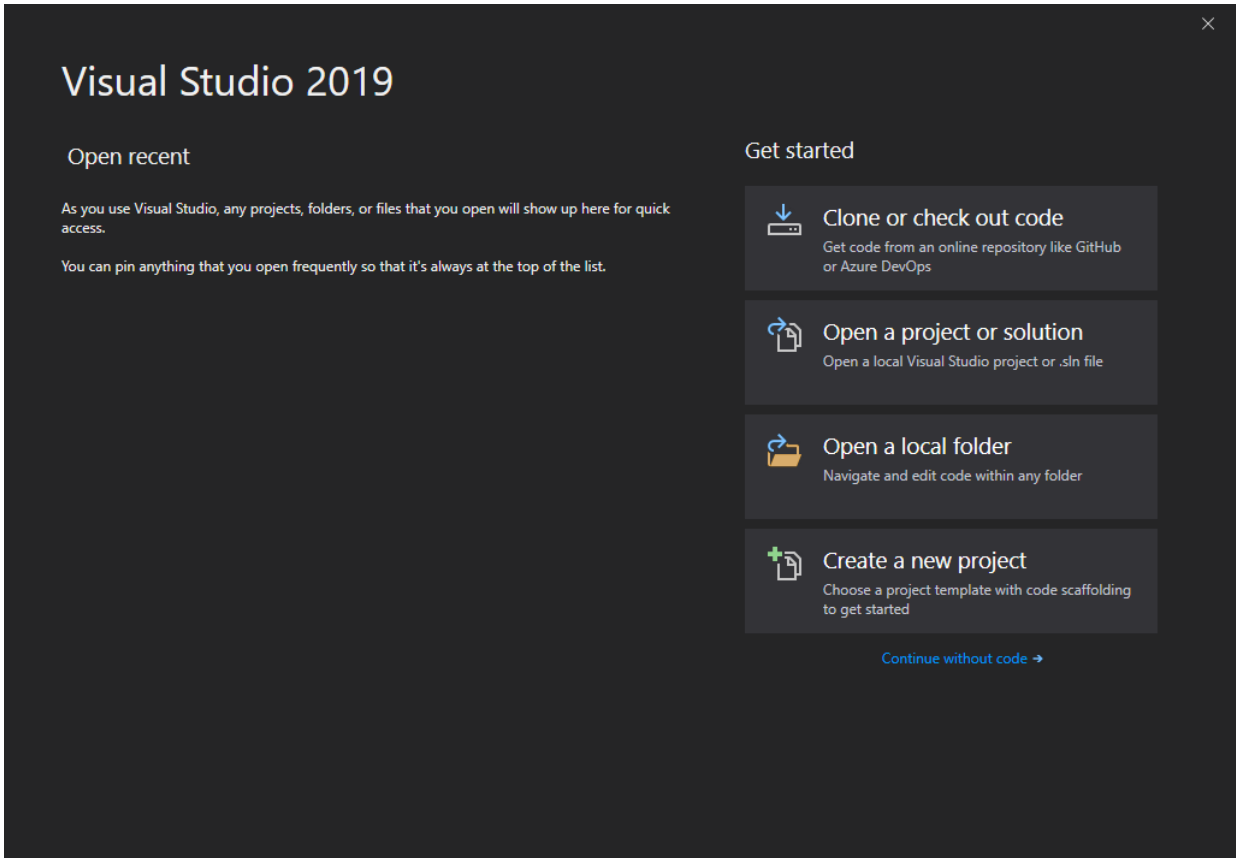
[✓ Blue] Blue (Extra Contrast) Dark Light

You can always change these settings later.

Figure 9. Select Visual C++ for the Development Settings.

Select **Visual C++** from the **Development Settings** drop down list. This will set the common keyboard keys for building and other common settings used for C++ development. Note that both the color theme and **Development Settings** can be changed at any time later on under **Tools - Options**.

Click **Start Visual Studio** button.


Visual Studio 2019

Open recent

As you use Visual Studio, any projects, folders, or files that you open will show up here for quick access.

You can pin anything that you open frequently so that it's always at the top of the list.

Get started

Clone or check out code
Get code from an online repository like GitHub or Azure DevOps

Open a project or solution
Open a local Visual Studio project or .sln file

Open a local folder
Navigate and edit code within any folder

Create a new project
Choose a project template with code scaffolding to get started

Continue without code →

Figure 10. Start page dialog.

This is the Start window for Visual Studio 2019. In this window you can choose to open a recent project or solution, Open a project or solution by browsing to it or Create a new project. You can also continue opening Visual Studio 2019 without opening or creating a new project or solution.

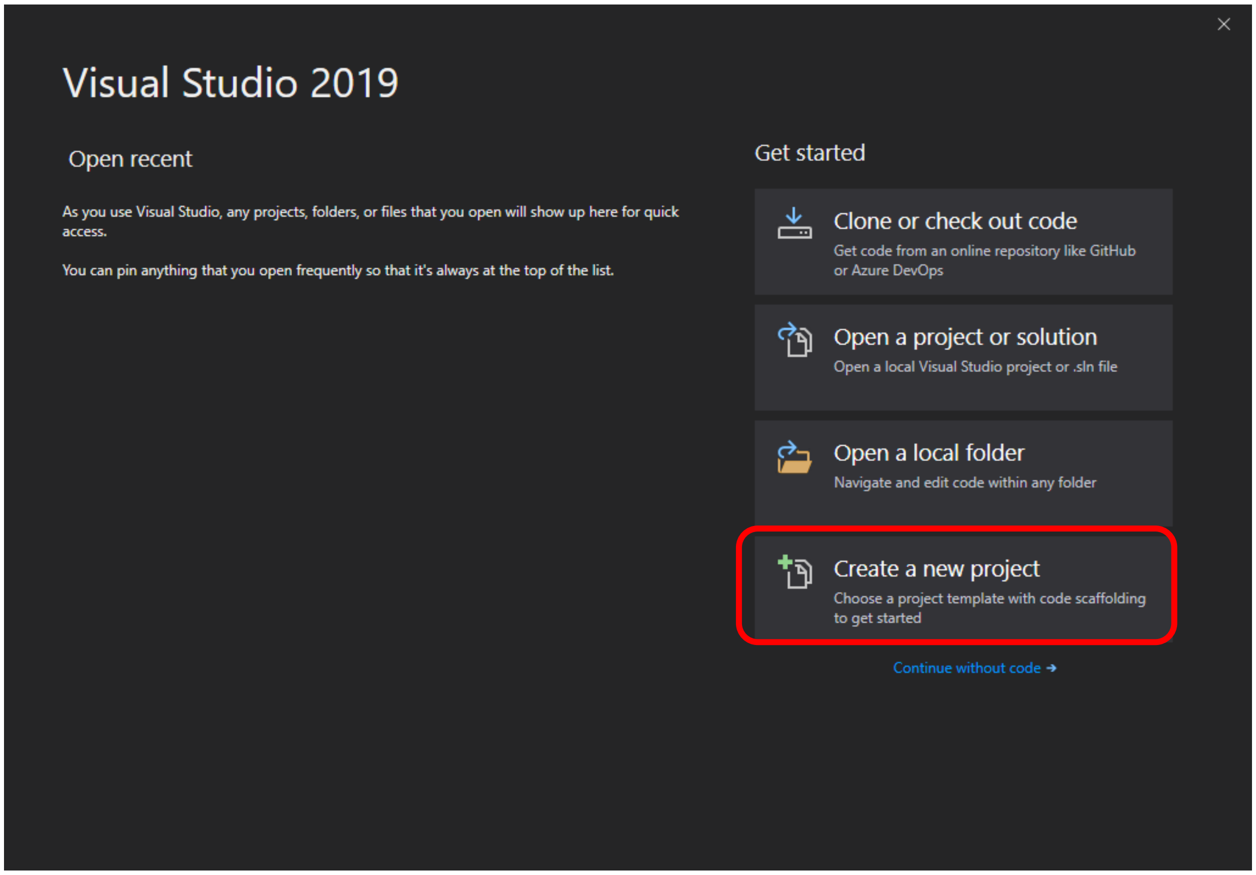
In the next section we will create our first program step by step.

GETTING STARTED - CREATING YOUR FIRST C++ PROJECT

ADDING A NEW PROJECT

After launching Visual Studio 2019 Community Edition we will create a new Project. When you launch Visual Studio 2019 you will get the Start window shown in Figure 11 by default. Let's create a simple project to get started with.

Click **Create a new project** button.



Visual Studio 2019

Open recent

As you use Visual Studio, any projects, folders, or files that you open will show up here for quick access.

You can pin anything that you open frequently so that it's always at the top of the list.

Get started

Clone or check out code
Get code from an online repository like GitHub or Azure DevOps

Open a project or solution
Open a local Visual Studio project or .sln file

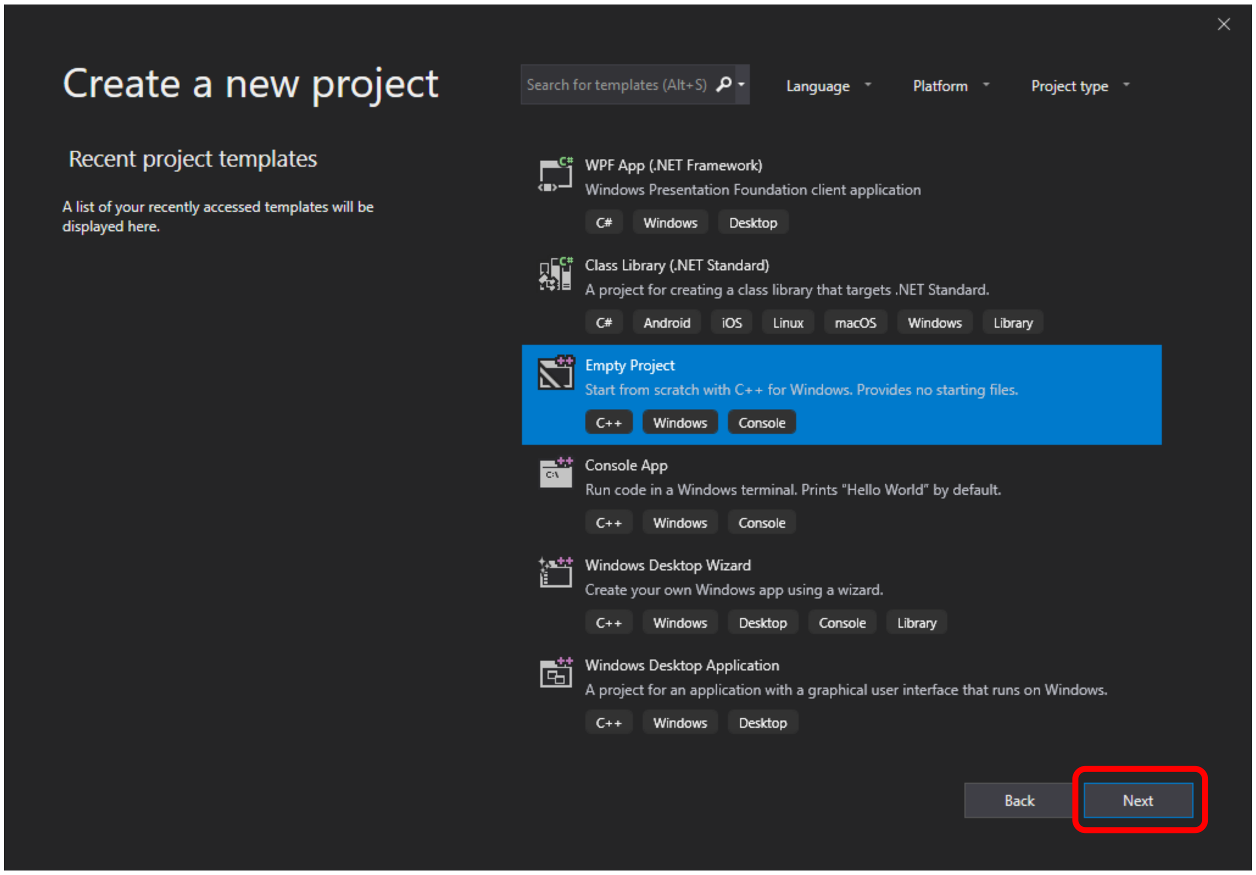
Open a local folder
Navigate and edit code within any folder

Create a new project
Choose a project template with code scaffolding to get started

Continue without code →

Figure 11. Default Start dialog.

This project along with all projects for the C++ Module I course will be console applications. This type of application does not have the typical Windows GUI. They run in a command console window instead.


Create a new project

Recent project templates

A list of your recently accessed templates will be displayed here.

WPF App (.NET Framework)
Windows Presentation Foundation client application
C# Windows Desktop

Class Library (.NET Standard)
A project for creating a class library that targets .NET Standard.
C# Android iOS Linux macOS Windows Library

Empty Project
Start from scratch with C++ for Windows. Provides no starting files.
C++ Windows Console

Console App
Run code in a Windows terminal. Prints "Hello World" by default.
C++ Windows Console

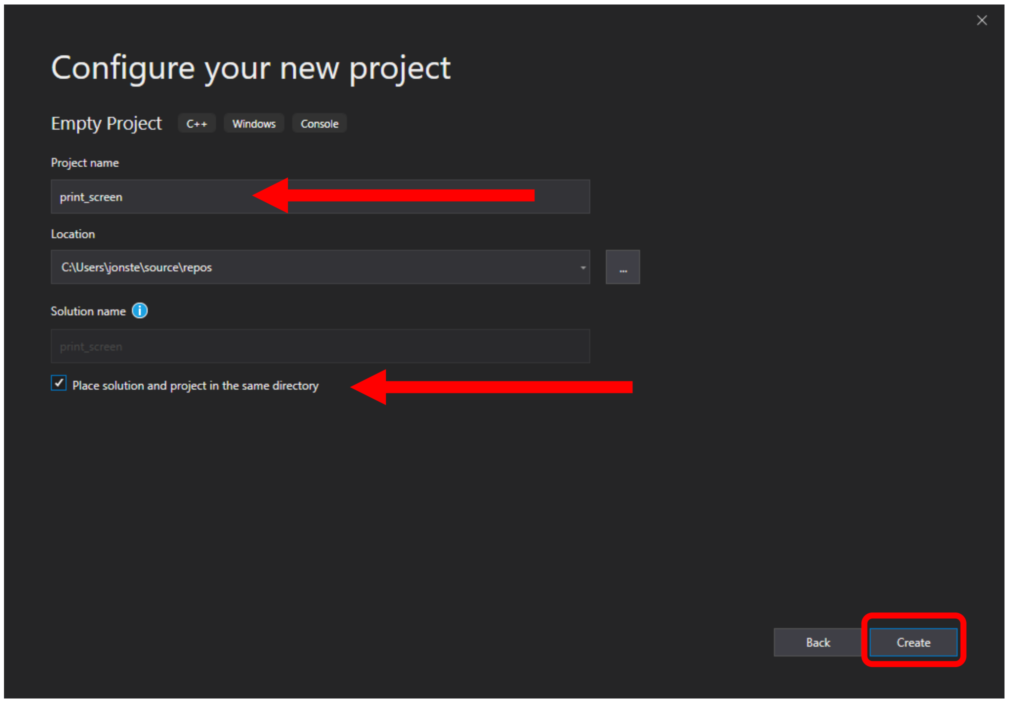
Windows Desktop Wizard
Create your own Windows app using a wizard.
C++ Windows Desktop Console Library

Windows Desktop Application
A project for an application with a graphical user interface that runs on Windows.
C++ Windows Desktop

Figure 12. Create project page

The following dialog should appear in Figure 12. You can choose many types of applicate templates to start with but for our purposes for this guide we will choose an **Empty Project**.

Select **Empty Project** and click the **Next** button.


Configure your new project

Empty Project C++ Windows Console

Project name: print_screen

Location: C:\Users\jonste\source\repos

Solution name: print_screen

✓ Place solution and project in the same directory

Figure 13. Empty project configuration page.

Enter a name for the project in the **Name** field. The name we are using is "print_screen". Enter or browse to the location where you want to save it on your hard drive. We are not going to create a separate directory for the Solution so check the box to place the solution and project files in the same folder. Creating a new Solution directory is most beneficial when you have multiple Projects in the Solution. The Solution becomes the parent directory with all Projects under it in subdirectories. This adds more complexity than what we need. Checking this box will create the Solution in the same directory as the Project.

Click **Create** to create the project.

At this point, we have successfully created a C++ project. The next step is to add some source code files.

As a side note, you might have noticed in the project selection dialog shown in Figure 12 there was a project type called **Console App**. We did not choose that type because we would not have been given any options to allow us to make an empty project. This type will create the console application project and create a single file with some starter code. This option can save a few steps but for this guide we want to show you how to manually add a file to the project so we chose **Empty Project**.

Let's take a look at what we have so far. In Figure 14 the window on the left side is called the Solution Explorer. It shows the layout of the files for the Project. It is important to note that the structure shown here *does not necessarily match* the physical directory structure as found on your hard drive. There are currently no files in the Project called **print_string**. That was intentional, in this case, since we chose an **Empty project**. You will not necessary always want to create empty projects but in this case we are so you can learn how to manually add you own files to the project. The Solution name is right above the Project name in the Solution tree. The Solution name will be the same as the name you gave the Project by default.

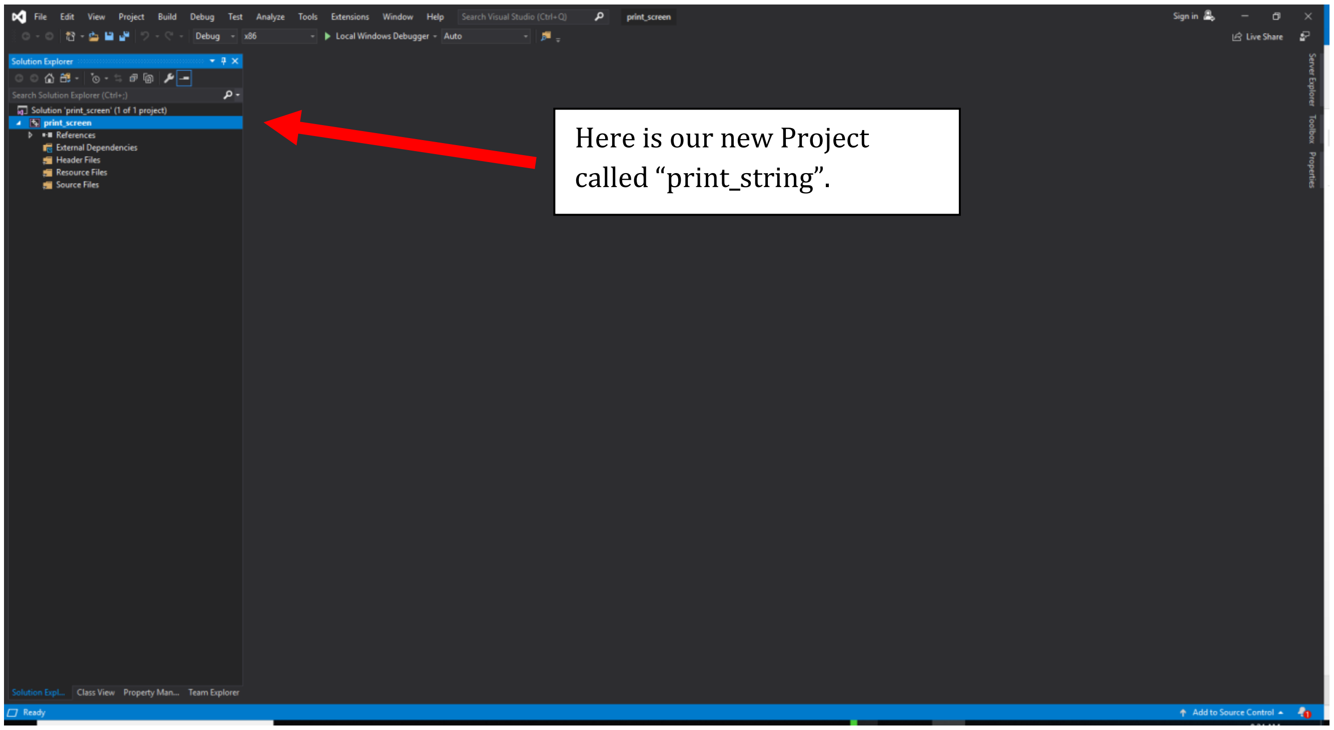

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help
Search Visual Studio (Ctrl+Q)
print_screen
Sign in
Live Share
Debug
x86
Local Windows Debugger
Auto
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'print_screen' (1 of 1 project)
print_screen
References
External Dependencies
Header Files
Resource Files
Source Files
Here is our new Project called "print_string".
Solution Expl... Class View Property Man... Team Explorer
Ready
Add to Source Control

Figure 14. Visual Studio 2019 Community Edition – Highlighted in the Solution Explorer

Adding a .CPP File to the Project

Ok so now let's add some code. We're going to add a .CPP file to the project. From the top menu select **Project – Add new item**.

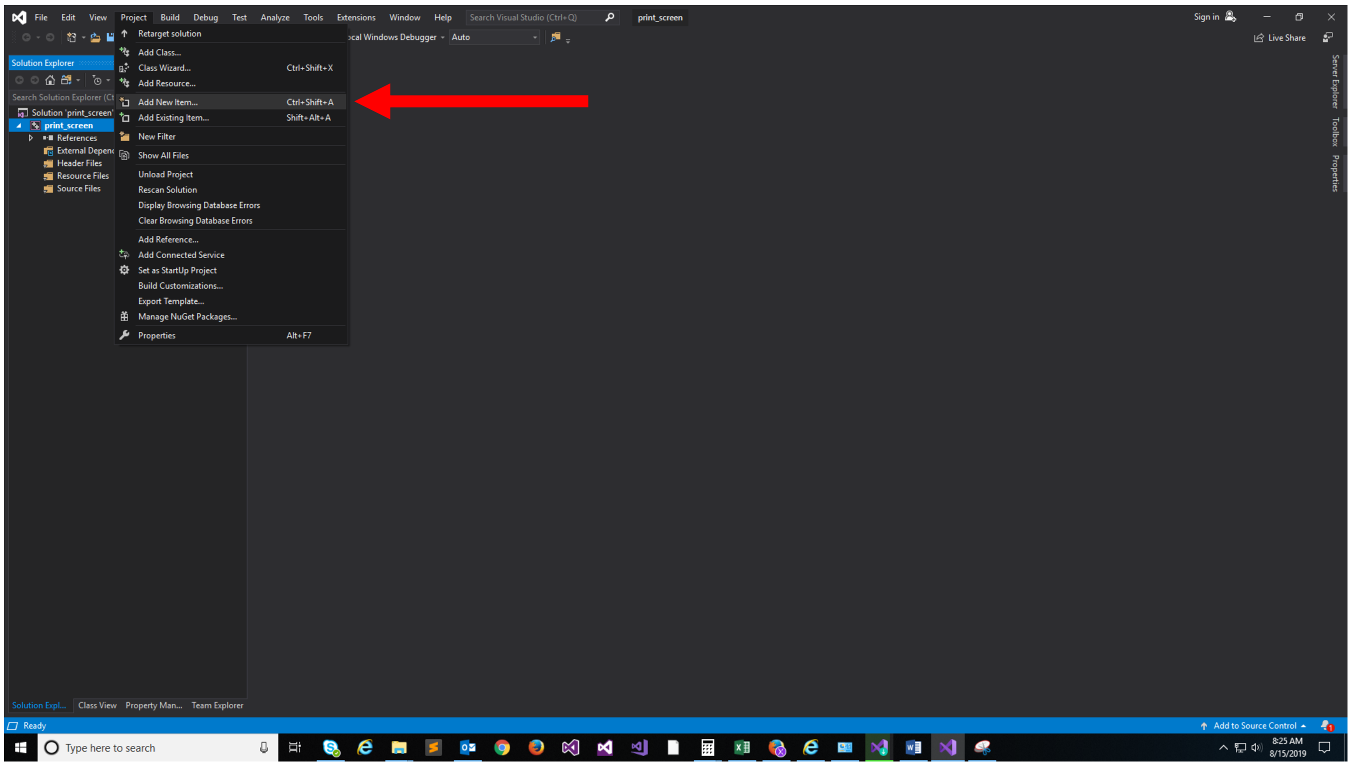
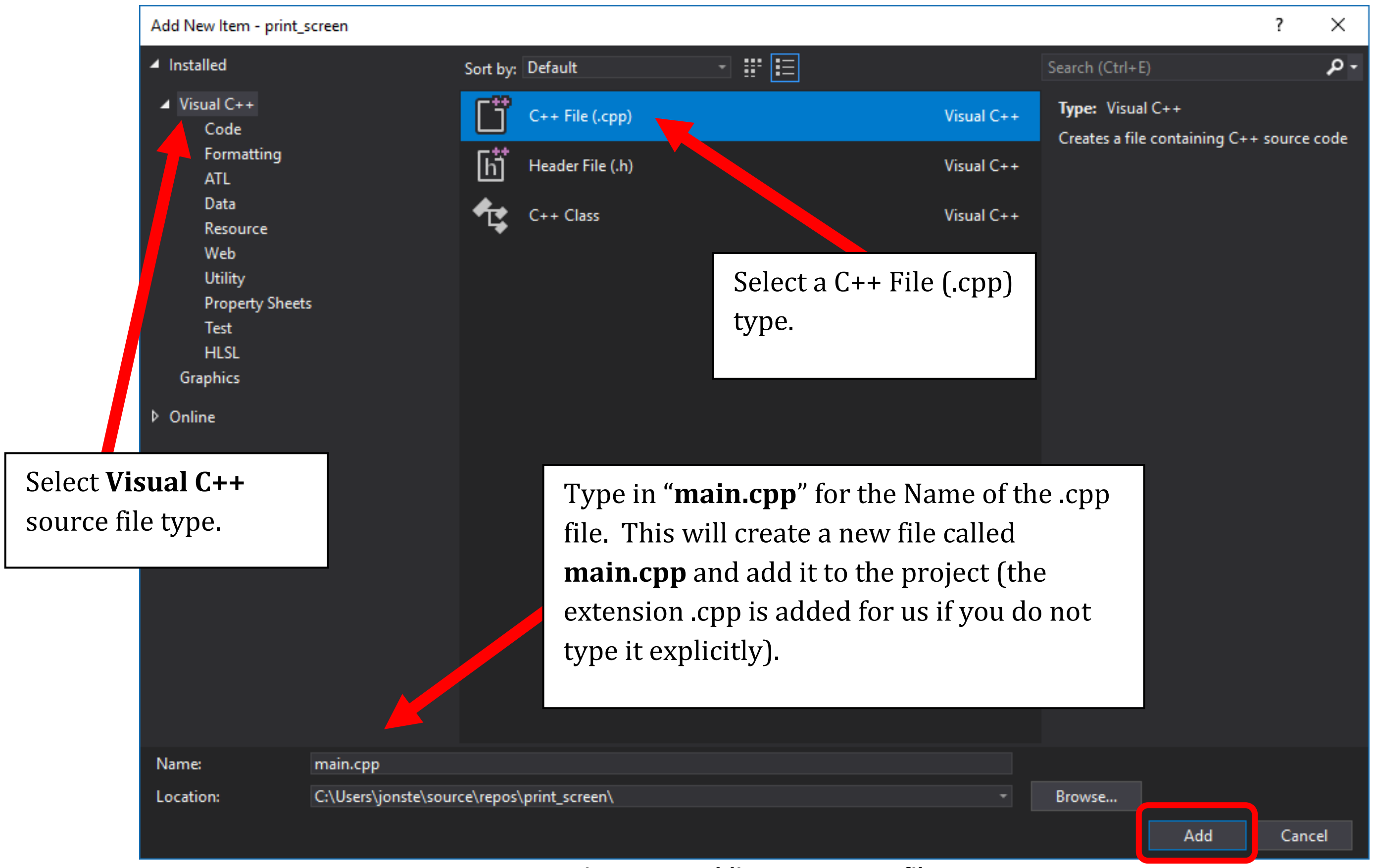


Figure 15. Using the menus to create a new Item (source code file) that will be added to the Project.

You can also right mouse click on the Project name in the Solution Explore and select **Add – New Item** from the context menu. Either way it is important that you do not select menu **File – New – File** otherwise the newly created file will not be part of the Project, therefore it will not get compiled into your final executable.

You will be presented with a dialog in Figure 16 to add a new file to the project. Since this is a C++ programming course we want to create a C++ source code file.



Type: Visual C++
Creates a file containing C++ source code

Name: main.cpp
Location: C:\Users\jonste\source\repos\print_screen\

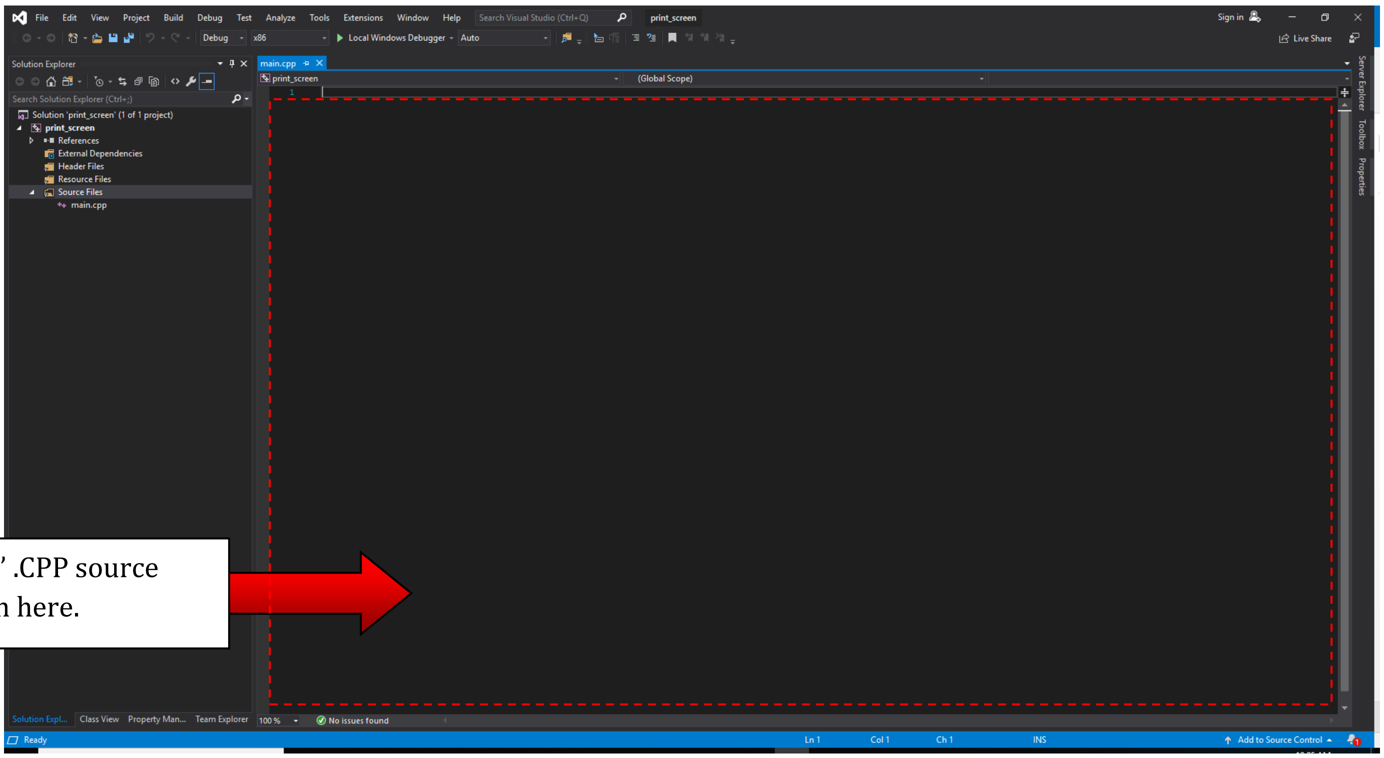
Select a C++ File (.cpp) type.

Select **Visual C++** source file type.

Type in "**main.cpp**" for the Name of the .cpp file. This will create a new file called **main.cpp** and add it to the project (the extension .cpp is added for us if you do not type it explicitly).

Figure 16. Adding a new .CPP file

Click on **Visual C++** in the left pane box. In the middle pane select a **C++ file (.cpp)**. Give the .CPP file a name and leave the Location as it shows. For console applications it is typical to call the source file "**main**" in a single file console application. Click **Add** to create the new file. You should see a blank .CPP source file showing in the main window of the IDE (Integrated Development Environment) as well as the .CPP file under the source folder in the Solution Explorer.



“main.cpp” .CPP source file. Edit in here.

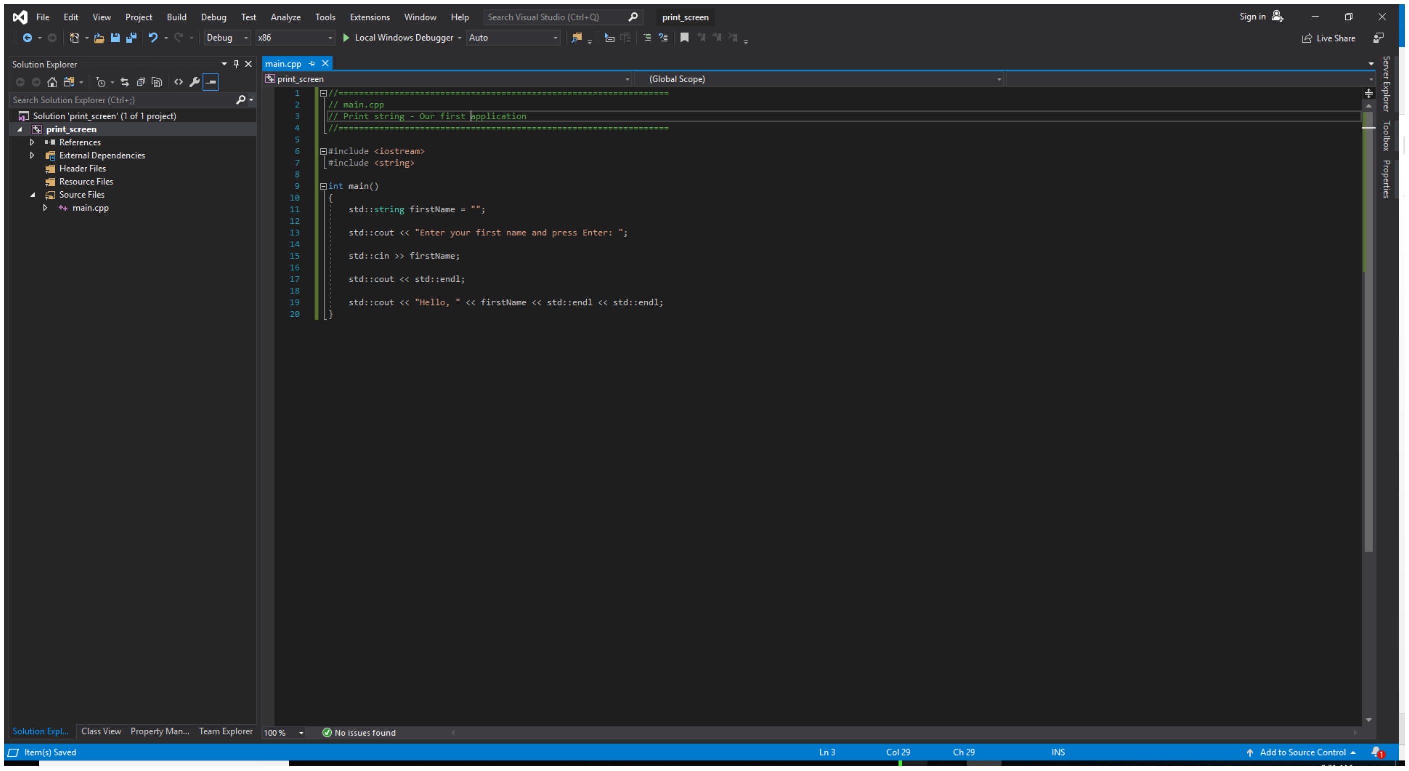
Figure 17. Visual Studio 2019 Community Edition window showing the newly added file.

WRITING THE CODE

In the blank .CPP file in the editor window, type the following C++ code, exactly as it is, into your .CPP file outlined by the red dashed line in Figure 17. You may have trouble copy-pasting the code from this PDF file as it may change the double quotes to a different character that will cause the code to fail to compile. It is good practice to just type in the code anyway!

```
//================================================================
// main.cpp
// Print string - Our first application
//================================================================
#include <iostream>
#include <string>
int main()
{
       std::string firstName = "";

       std::cout << "Enter your first name and press Enter: ";

       std::cin >> firstName;

       std::cout << std::endl;

       std::cout << "Hello, " << firstName << std::endl << std::endl;
}
```

We will not discuss the source code and what it means because this has already been done in the *C++ Programming for Games Module I* course textbook. Refer to Section 1.2 **The "Print String" program Explained** to have the specifics of the source code explained.



```
//==========================================================================
// main.cpp
// Print string - Our first application
//==========================================================================

#include <iostream>
#include <string>

int main()
{
    std::string firstName = "";

    std::cout << "Enter your first name and press Enter: ";

    std::cin >> firstName;

    std::cout << std::endl;

    std::cout << "Hello, " << firstName << std::endl << std::endl;
}
```

Figure 18. Showing the final code edited in the source file "main.cpp".

BUILDING THE PROGRAM

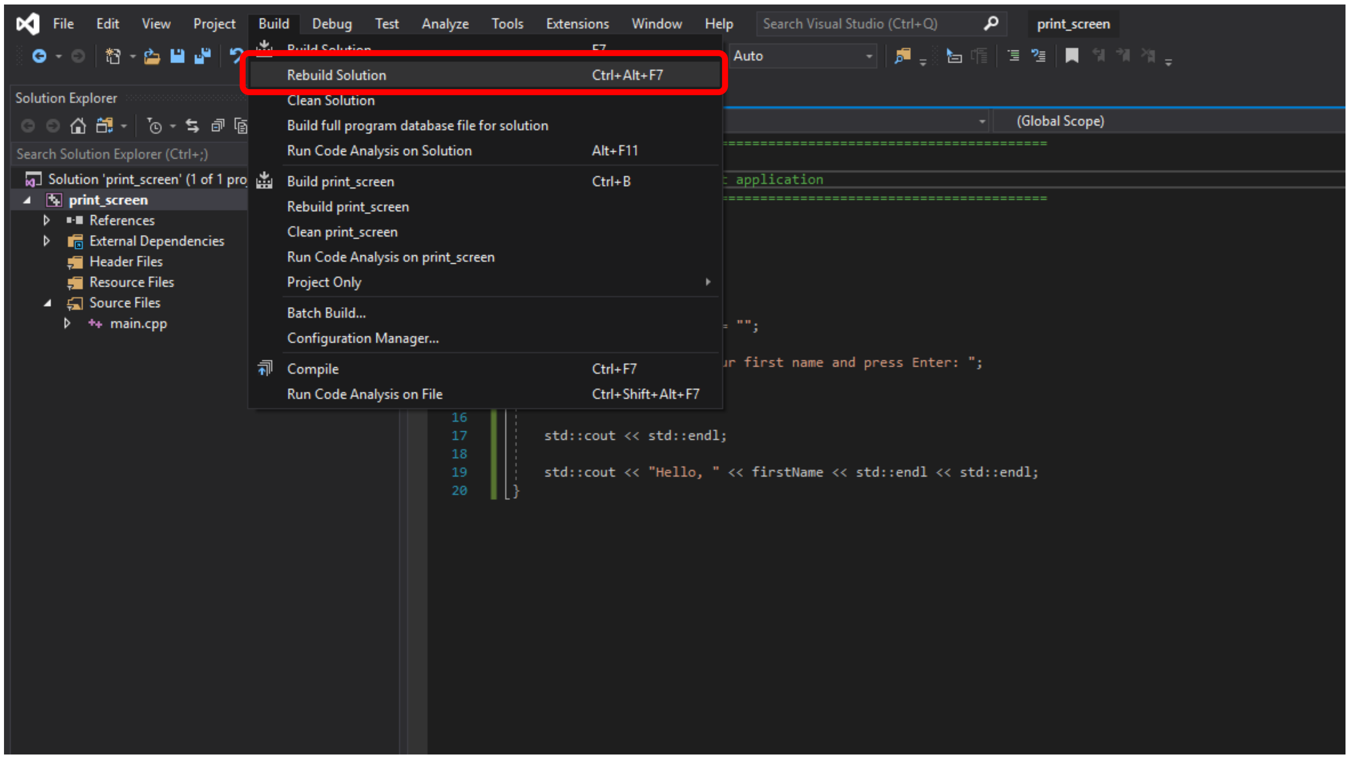
After the C++ source code has been entered, it must be translated into a language that computer understands. There are two steps in the translation process.

1. Compilation
2. Linking

Solutions in Visual Studio Community 2019, as in previous editions, contain one or more Projects. Throughout the *C++ Programming For Games Modules I and II* only a single project is ever discussed and used in a Solution.

Don't worry if you do not fully understand what compiling and linking means, these steps are explained in more detail in the course. Now let's build our project. From the main menu select **Build – Rebuild**

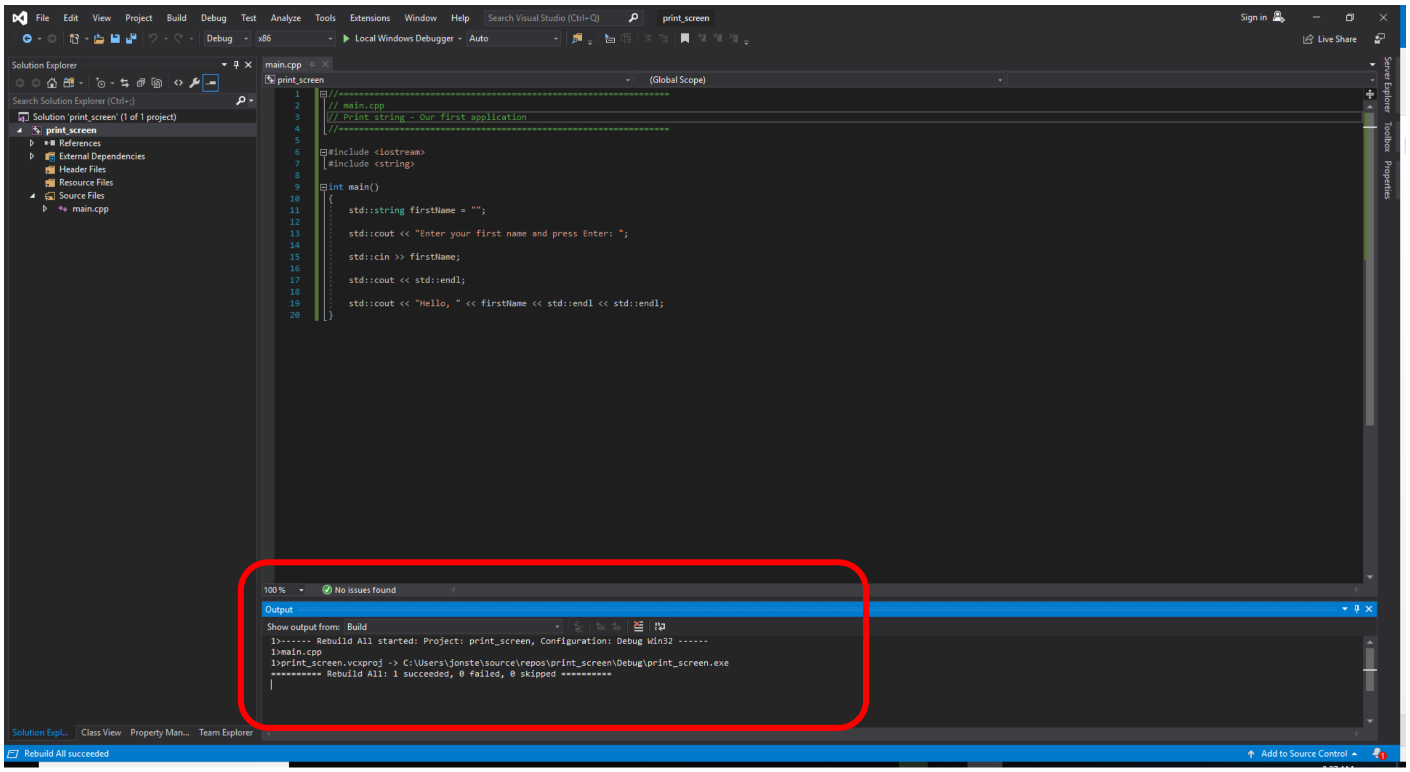
Solution (see Figure 19) or Press F7 which is the short cut key. Rebuild Solution versus Build Solution will perform a Clean Solution first before building which cleans up any existing files that were created for the build. It does not erase source files, however.



```
    std::cout << std::endl;

    std::cout << "Hello, " << firstName << std::endl << std::endl;
}
```

Figure 19. Build the program.

While it is compiling and linking you will see status activity in the Output window. If you do not see the Output window or any other window for that matter you can always go to the menu View and show and or hide them. You will see all the compiler complaints and other information in this window when compiling and linking projects.



```
//=================================================================
// main.cpp
// Print string - Our first application
//=================================================================

#include <iostream>
#include <string>

int main()
{
    std::string firstName = "";

    std::cout << "Enter your first name and press Enter: ";

    std::cin >> firstName;

    std::cout << std::endl;

    std::cout << "Hello, " << firstName << std::endl << std::endl;
}
```

Show output from: Build

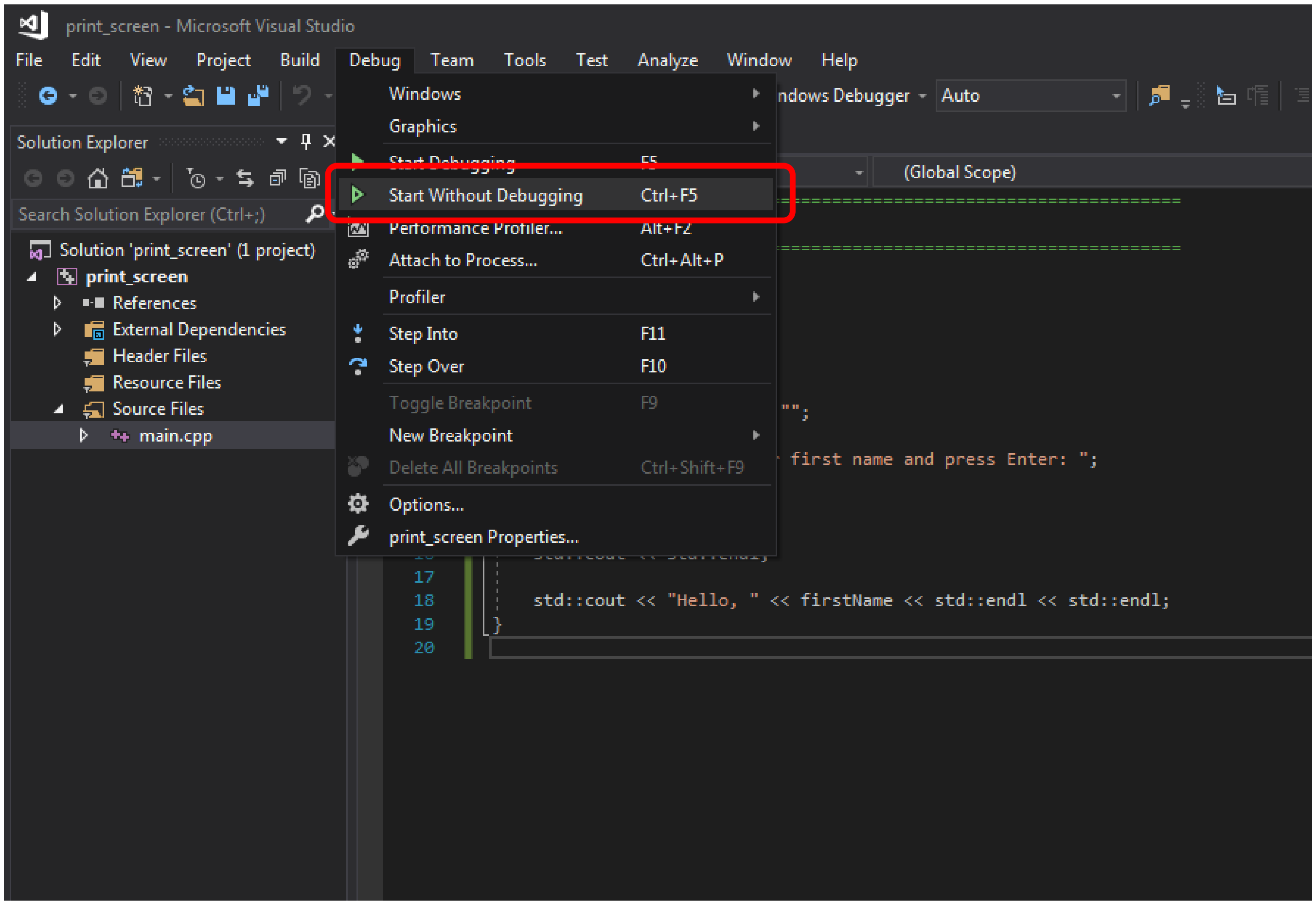
```
1>------ Rebuild All started: Project: print_screen, Configuration: Debug Win32 ------
1>main.cpp
1>print_screen.vcxproj -> C:\Users\jonste\source\repos\print_screen\Debug\print_screen.exe
========== Rebuild All: 1 succeeded, 0 failed, 0 skipped ==========
```

Figure 20. The Output Window after compiling and linking

In Figure 20 look at the last two lines. This compilation had 0 errors and 0 warnings which is the best we can ever hope to achieve. That means the code was written with legal C++ code and therefore the program has linked successfully. The textbook goes into more detail on what exactly it means to compile and link a program.

RUNNING THE PROGRAM

At this point an executable has been created for our program. We can now run the executable and make sure it works. From the Debug menu, select **Start without Debugging** or use CTRL+F5 keyboard keys. This will launch the console application from within the Visual C++ IDE.



```
std::cout << "Hello, " << firstName << std::endl << std::endl;
}
```

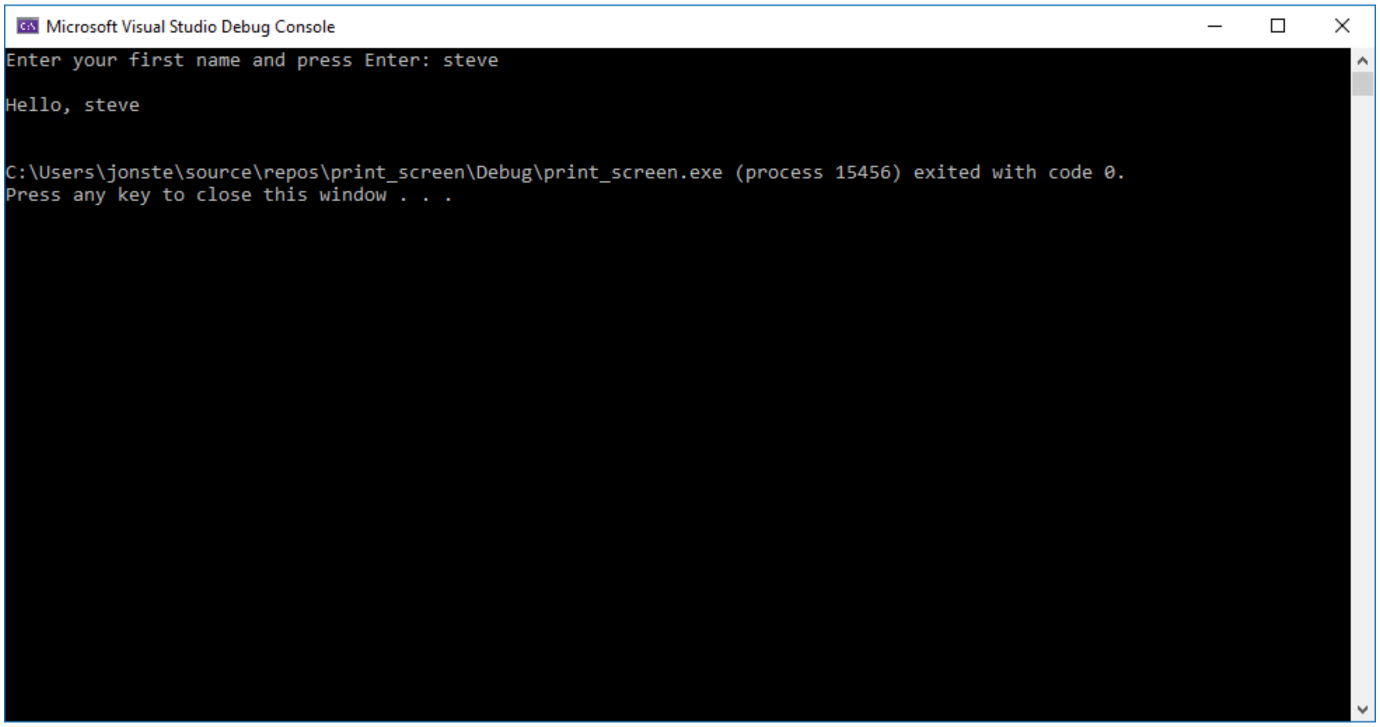
Figure 21. Run your program!

You should first see the following output:

```
Enter your first name and press Enter:
```

In this example the text “steve” was typed and the Enter key was pressed. The program then displays the following:

Program 1: Output



```
Enter your first name and press Enter: steve

Hello, steve


C:\Users\jonste\source\repos\print_screen\Debug\print_screen.exe (process 15456) exited with code 0.
Press any key to close this window . . .
```

Congratulations!! You have just run your first C++ program! Note that the textbook for C++ Programming for Games Module I course goes through each line in the source code explaining exactly what is going on in the code.

CONCLUSION

We began by installing Visual Studio 2019 Community Edition software then we moved on to writing our first simple C++ program. Finally, we built then ran the program resulting in some output in a console window. The basic steps to creating a console application written in C++ have been demonstrated. With this knowledge you can develop other types as well such as Win32 Windows applications. The line by line detailed explanation of the source code we used to build the executable is found in the textbook for the first course on C++ at the Game Institute, *C++ Programming for Games Module I*. Have fun coding!