**Setting Up a New Plugin Project in VS17**

Prerequisites:

1.      <https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=Community&rel=15>

2.      <https://github.com/Michidu/ARK-Server-API/archive/master.zip>

Setting up the project:

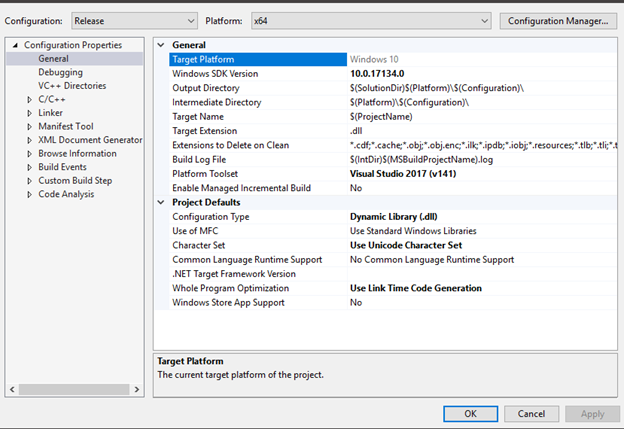
1. Install Visual Studio by the link given on the first prerequisite, when the install gets to a certain point it will ask you which addons you want to install, here you need to chose C++ as it contains all the necessary stuff for DLL compiling
2. Configuration: after installing visual studio and the required addons for C++ now you need to create a new project for this click **File** in the top left corner of the program, hover your cursor over **New**  and then click project
3. Expand the C++ tab and then select Dynamic-Link Library (this option only shows up if you have the C++ addon, so make sure you installed it!), now give your project a name and click OK
4. After visual studio has created the necessary environment for coding DLL files you should see a snippet of code which are the precompiled header files, now we need to prepare visual studio for plugin development!

Setting up the Project Configuration:

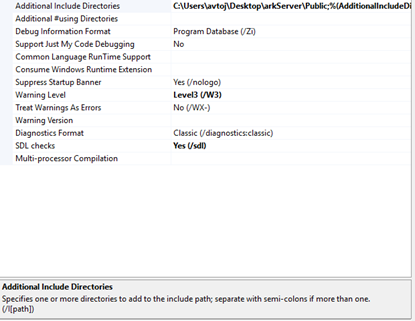
1.       Make sure to set visual studio to Release and X64

https://lh6.googleusercontent.com/_W6Q8b3WHzjnZTmIJENfwyYGzwuHcH04ykyQloTtw4OZPcAqdqfusaYeb6nW6r1BmifwnElyvckp_q76lHlW2b6A0sM2SqI9OrctPG5T9A8adR7XkyoJHJunabUGu0GlwI1M7_VZ

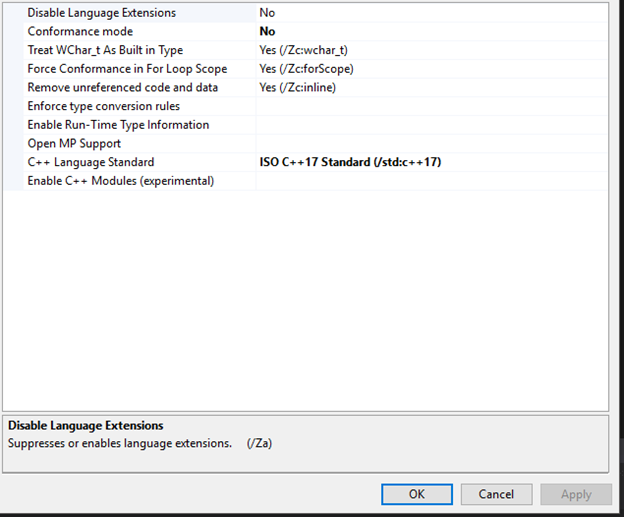
2.       Click view and open Property manager and then click the wrench icon of the property manager to open project specific options!



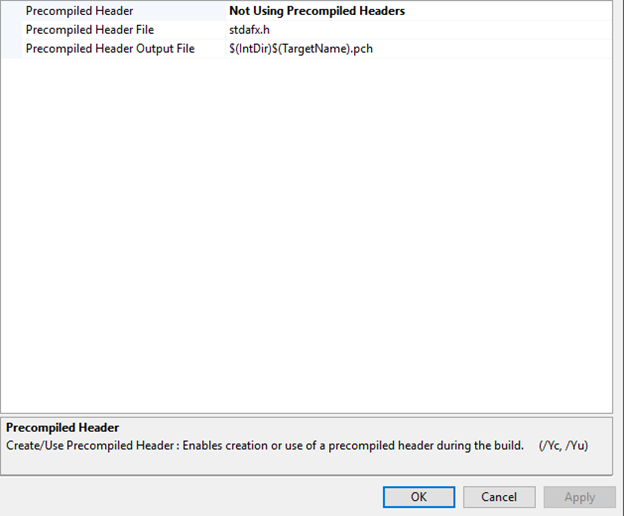
3.       Click on the C/C++ tab first, then select additional include directories, here you need to point Visual studio to the path where you downloaded the second resource (it needs to be pointed to the Public folder not the root of the API folder!)



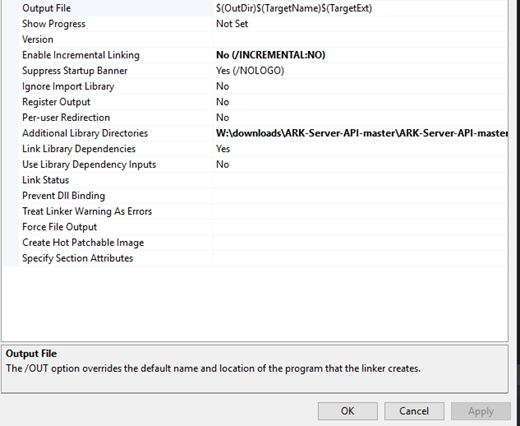
4.       Click The language tab in C/C++ tab and set the language standard to ISO C++ 17



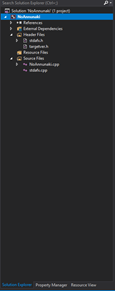
5.       Click precompiled headers in C/C++ and select not using precompiled headers



6.       Now Expand the linker tab and click general, after select Additional Library Directories, here you need to set the path to the second resource but instead of selecting the Public folder this time you select the Lib folder



7.       Sometimes you will have precompiled files that stay in the project that are leftover from the first creation of the environment, sometimes those headers will not let you compile plugins correctly usually a situation like this is accompanied by an error message in the compiler, now since we aren’t using any precompiled headers we don’t need them if you’re having a problem of a plugin not compiling due to a precompiled header click **View** and then select solution explorer here find the precompiled header that is causing issues and remove it !



You should now be able to compile basic plugins !