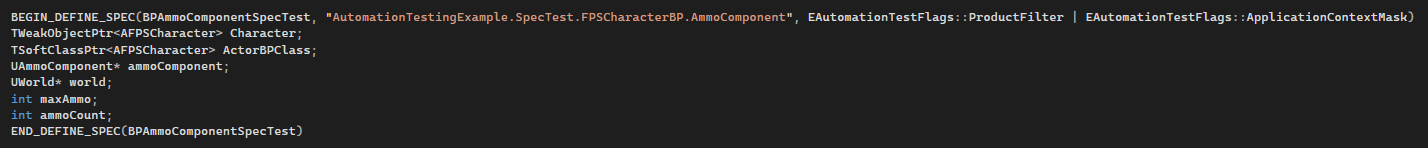
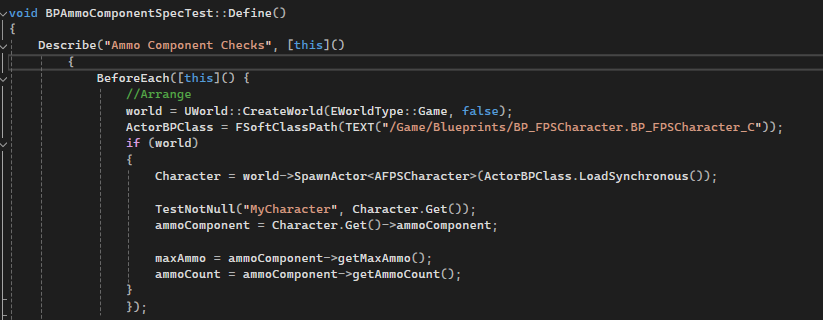
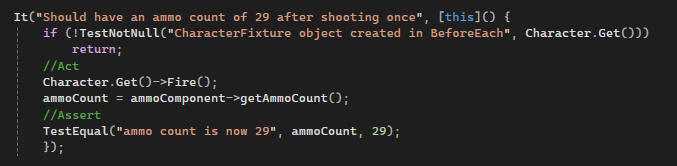
Unreal Engine Spec Automation Test

# Creating a Spec Test Class

1. Open Unreal Engine.
2. Select "C++ Classes" folder in the Content Browser.
3. Right Click and add a new C++ class.
4. Select None for the parent class.
5. Select "Private".
6. Input the class name.
7. Add "Tests/" to the end of the file path.
8. Once compiling has finished close Unreal Engine.
9. Open the projects source folder and navigate to the new class.
10. Delete the header file and rename the cpp file to "*FileName*.spec.cpp".
11. Return to the project folder and delete the following folders/file:
    1. .vs
    2. Binaries
    3. Intermediate
    4. Saved
    5. "*ProjectName*.sln"
12. Right click the "*ProjectName*.uproject" file and select "Generate Visual Studio Project Files".
13. Once completed open the "*ProjectName*.sln" file.
14. Build the vs project by pressing F5.
15. Open the "*FileName*.spec.cpp" file in visual studio.

# Writing a Spec Test

1. Delete all the code that’s in the "*FileName*.spec.cpp" file.
2. At the top of the file you will need to add any includes that are required for the test to run.
   1. The Class you are testing, eg: #include “AmmoComponent.h”
   2. If a character is required for testing, in this case for the fire function.
3. Start with the Define Spec Macro, this can be the Begin\_Define\_Spec or just Define\_Spec depending on your requirements.
   1. The first section of the Macro is the test class name.
   2. The second section is where the test will be located in the session front end.
   3. The third section is for the [EAutomationTestFlags](https://docs.unrealengine.com/4.26/en-US/API/Runtime/Core/Misc/EAutomationTestFlags__Type/)
4. If you used the Begin\_Define\_Spec Macro add any global variables then use the End\_Define\_Spec.  
   
5. Write the "Define" function for the test class name that was assigned in the Macro.
6. Inside the "Define" function write a "Describe" function with two parameters, the first one is a string that describes the tests being preformed, eg "Ammo Component Checks" and the second is a lamba function with "this" as a capture clause.
7. If required a "BeforeEach" and "AfterEach" function can be added to the Describe function which will run before and after each "It" function. These functions take on parameter which is a Lamba function with "this" as a capture clause.   
     
   For this ammo component test we will need a "BeforeEach" where we will create a world using "Uworld::CreateWorld(EWorldType::Game, false);"  
   and then using the world object spawn in an actor of the required character class. If spawning a blueprint character, you will also have to find the blueprint class in this function before spawning it.  
   
8. Inside the "Describe" lamba function is where we add the "It" functions which preform the tests. The "It" function takes two parameters, the first one is a string describing the test and should start with "Should" eg: It("Should start with a max ammo count of 30". And the second being a lamba function again with "this" as a capture clause.
9. Inside the "It" lamba function you can preform whatever actions are needed for the test and then evaluate the results by using an assert such as "TestEqual".  
   
10. Finally, as we created a world in the "BeforeEach" function we need to destroy that in an "AfterEach" function.

