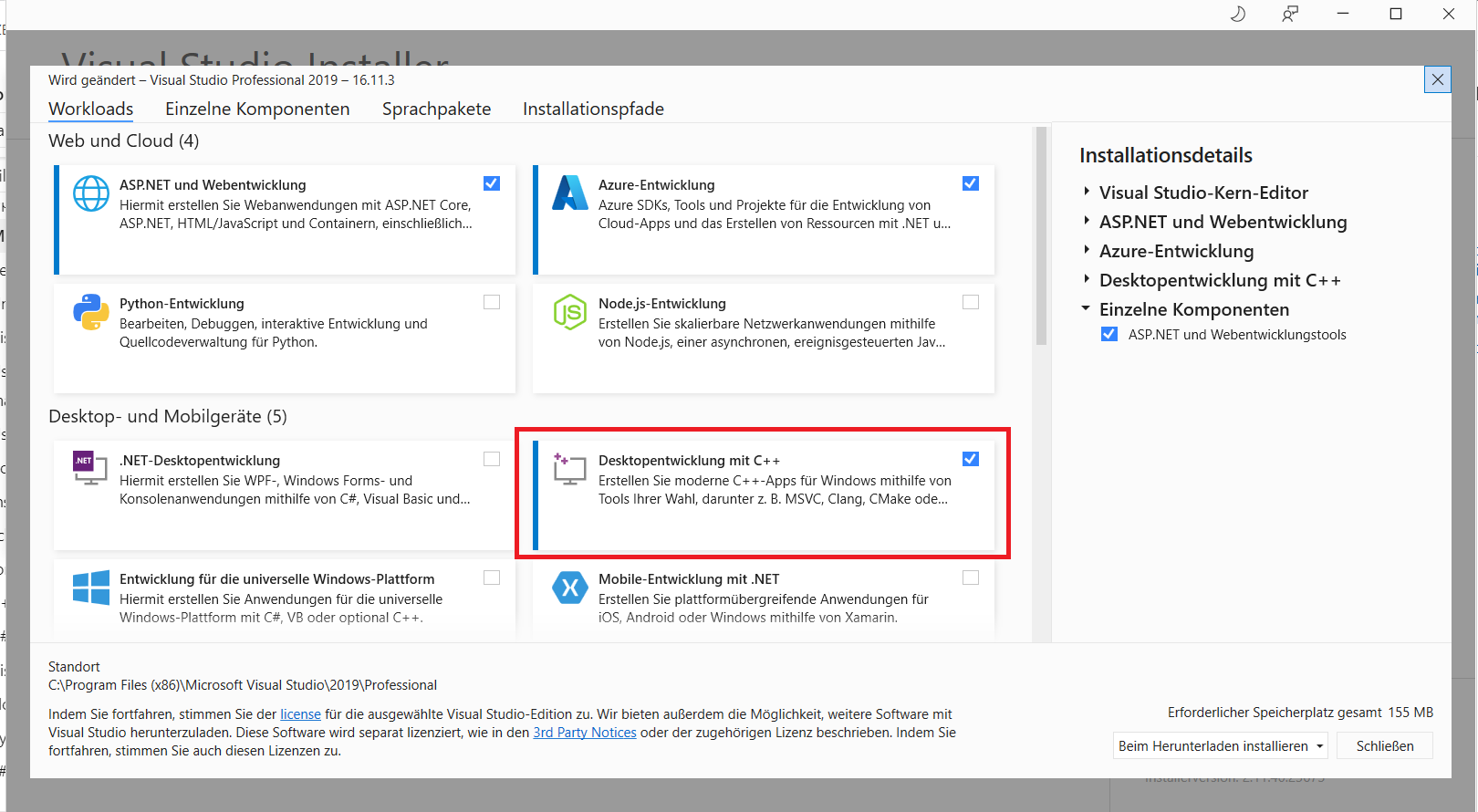
C++ Interoperabilität mit C#

In diesem Tutorial zeigen wir euch, wie ihr ein C++-Projekt in einer C#-Anwendung jeglicher Art verwenden. Ein Repository mit dem Code für dieses Tutorial wird bereitgestellt.

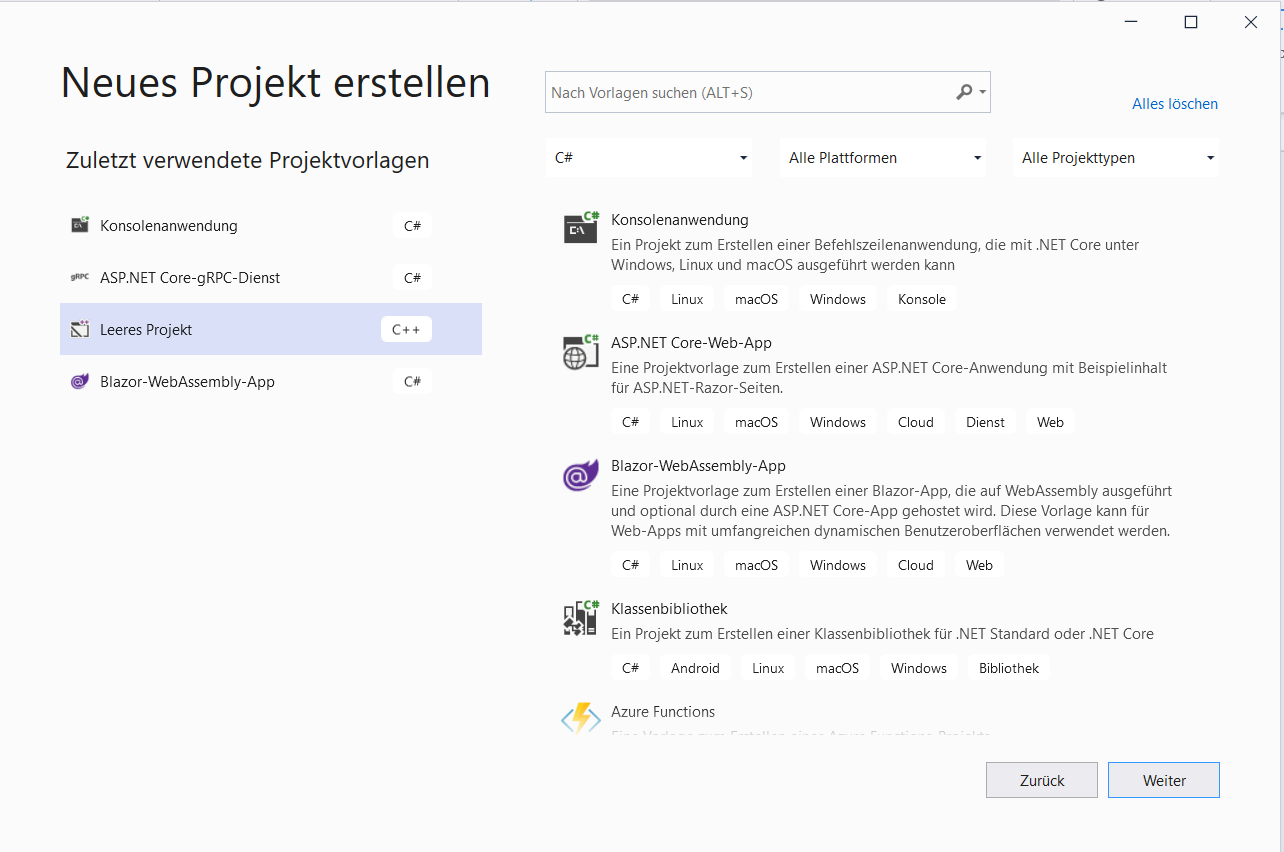
Voraussetzungen

1. Visual Studio 2019
2. .NET Core 5
3. Desktopentwicklung mit C++ (seht unter)



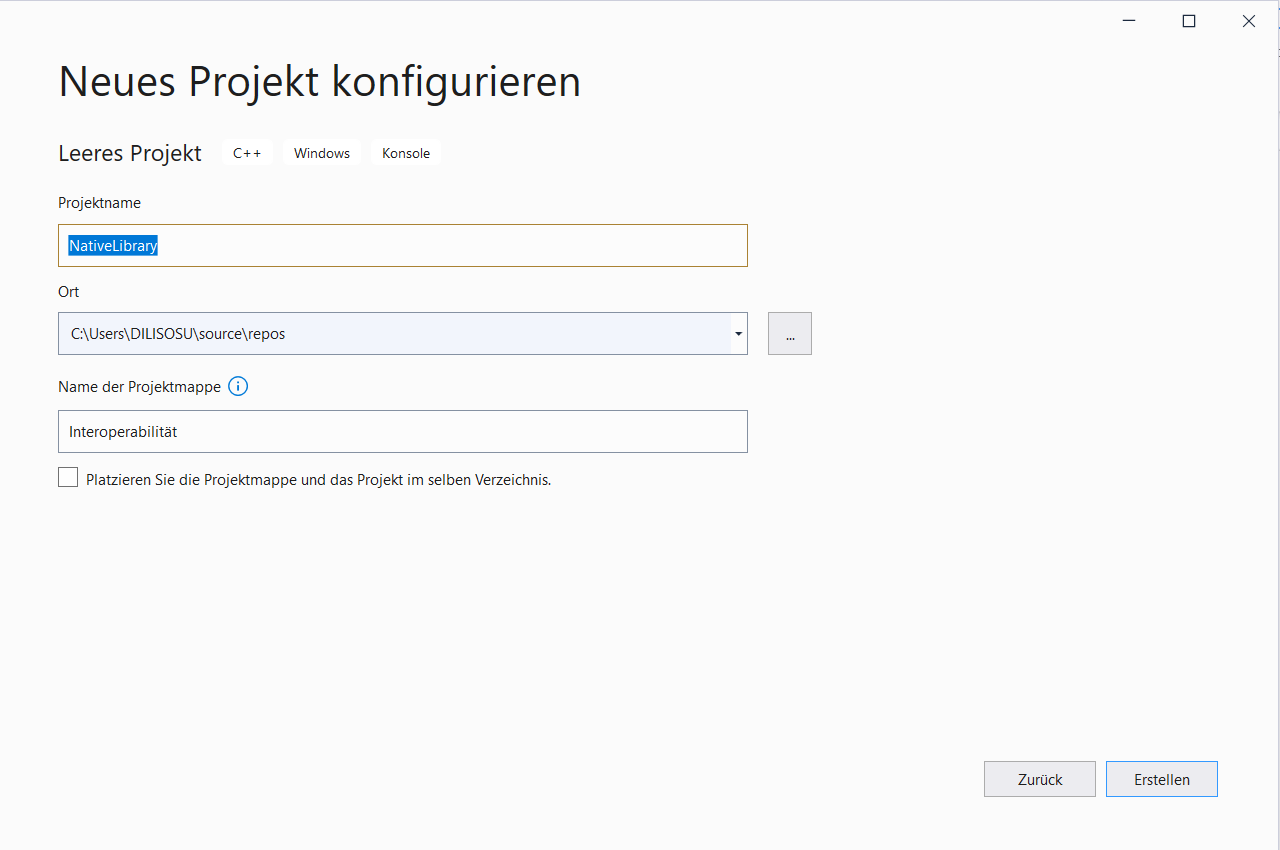
Tutorial

Schritt 1.

Erstellt ihr ein neues leeres C++ Projekt. 

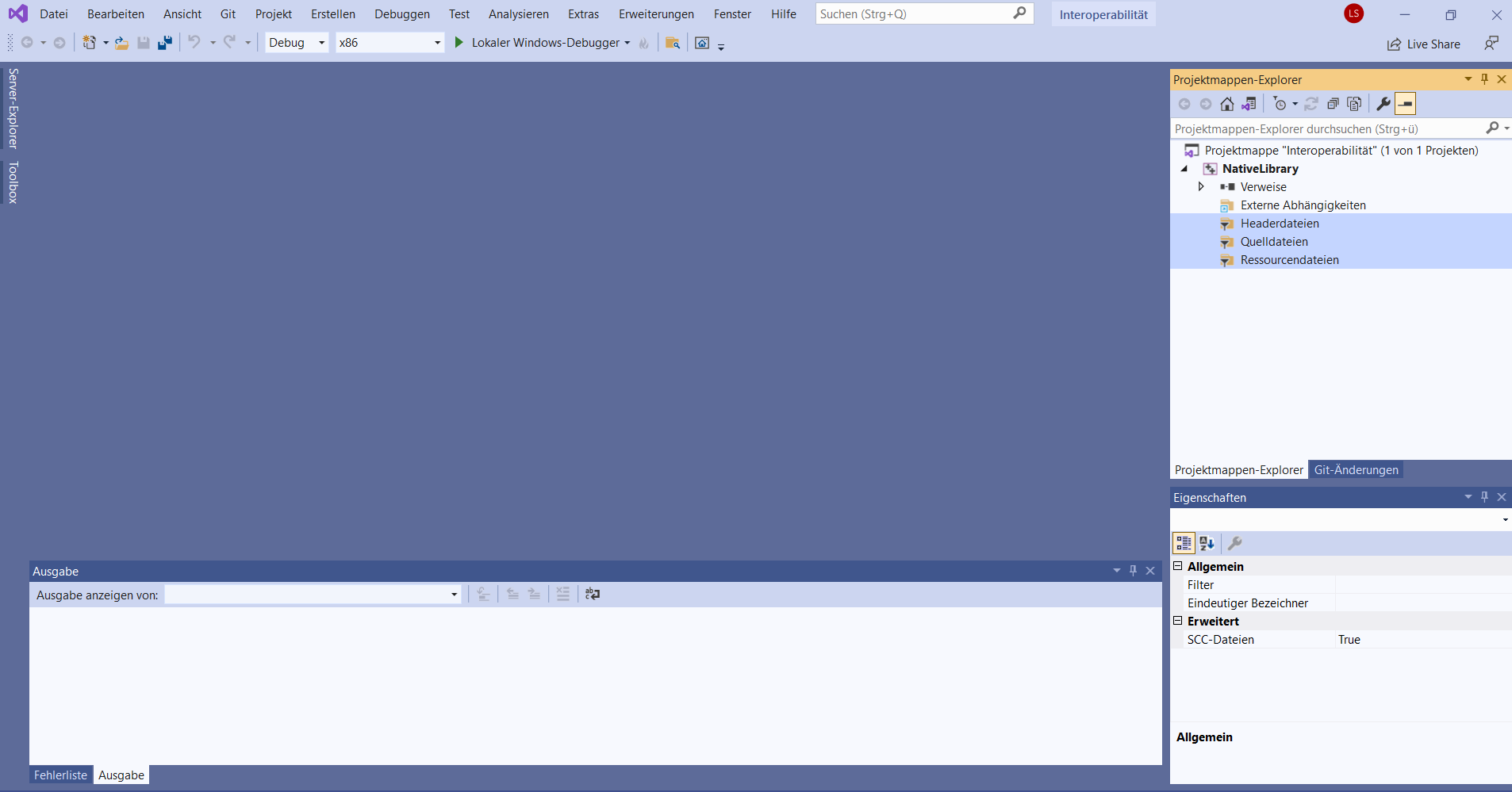
Schritt 2.

Benennet ihr das Projekt nach Belieben, aber wir nennen es "NativeLibrary"



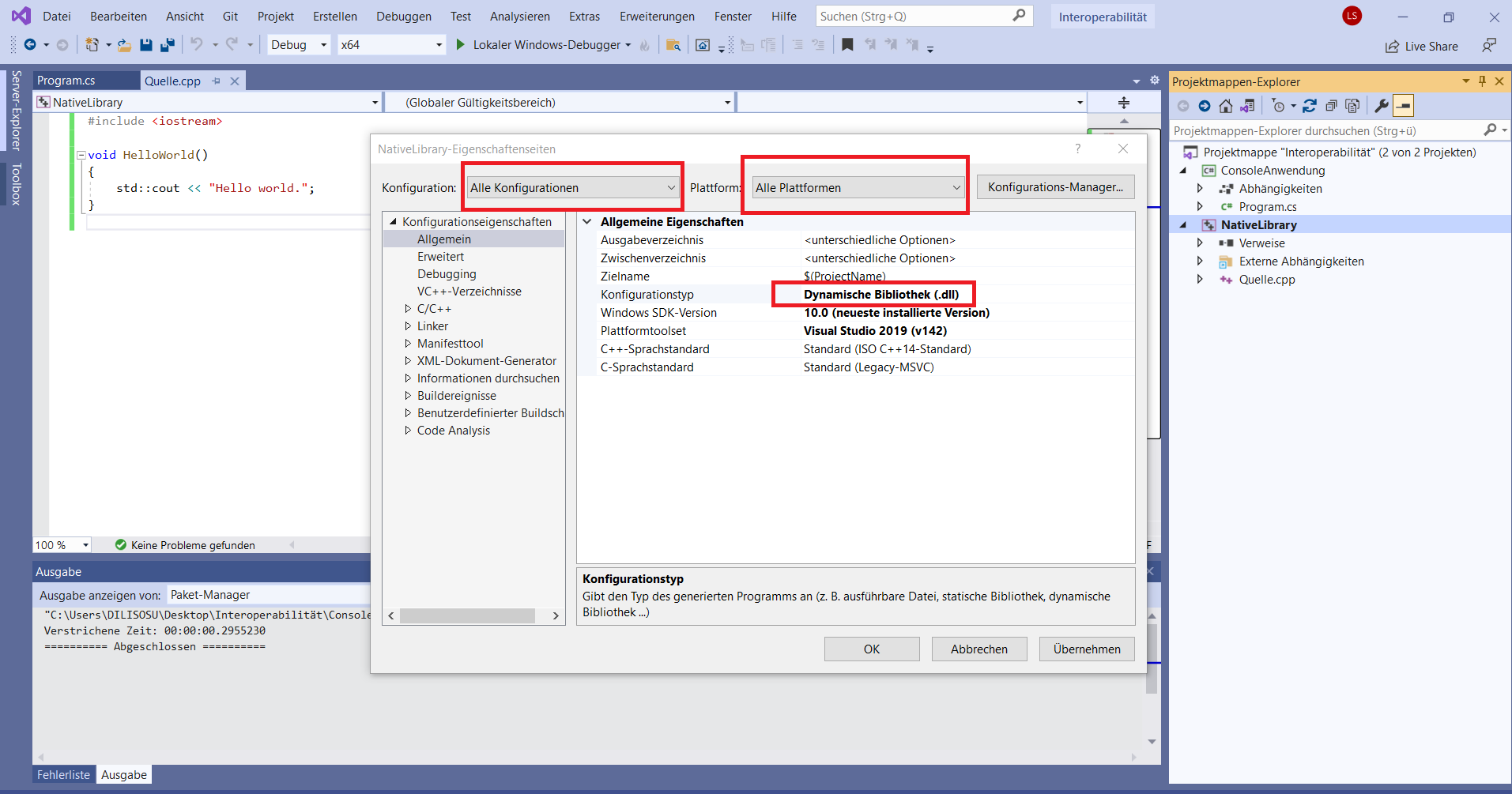
Schritt 3.

Entfernt ihr die markierten Ordner.



Schritt 4.

Bitte klickt ihr mit der rechten Maustaste auf das C++-Projekt und wählt ihr Eigenschaften. Wählt ihr die Dropdowns wie im Bild gezeigt aus.

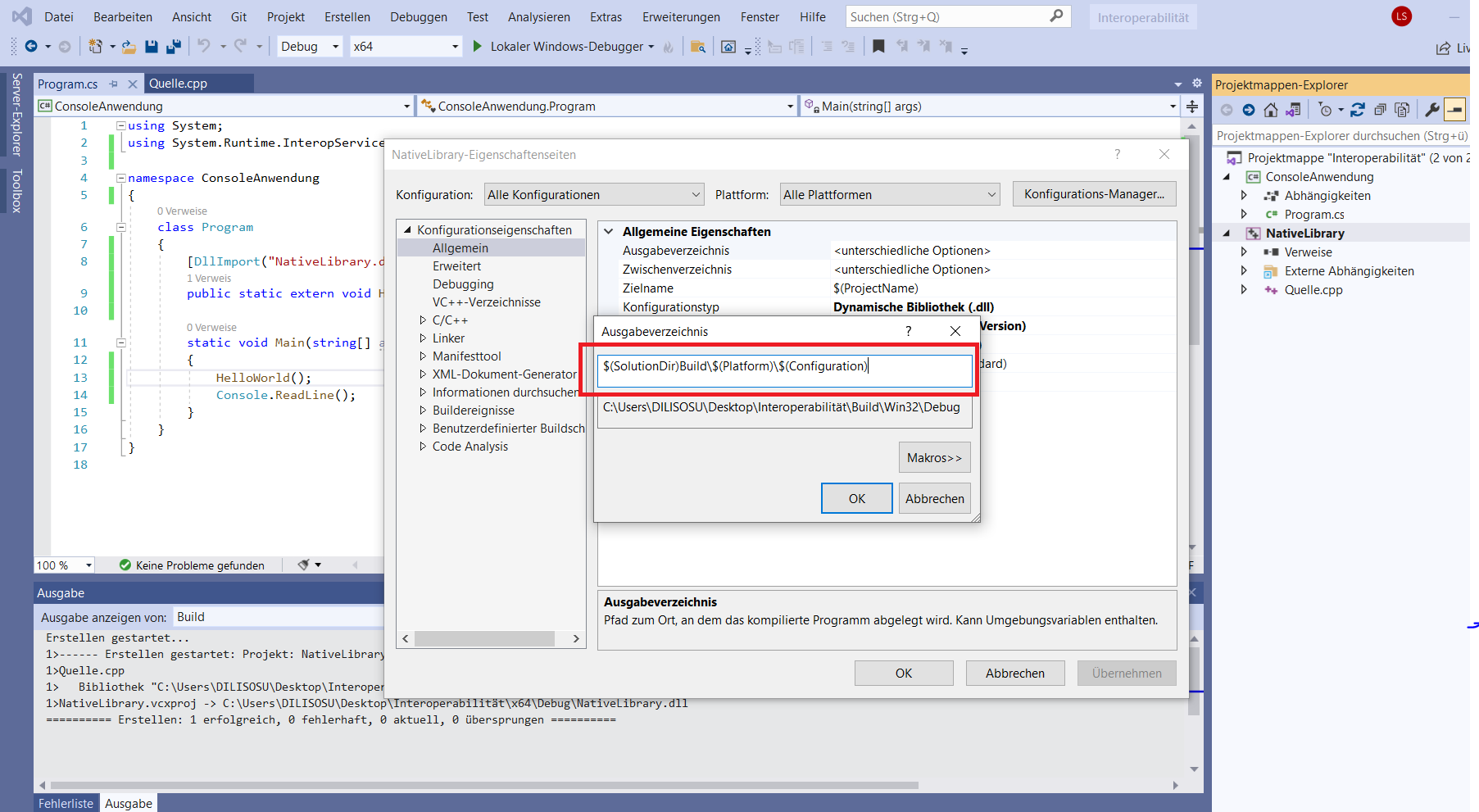


Schritt 5.

Hinzufügt bitte auch die folgende Pfad

*$(SolutionDir)Build\$(Platform)\$(Configuration)*

wie im Bild gezeigt aus.



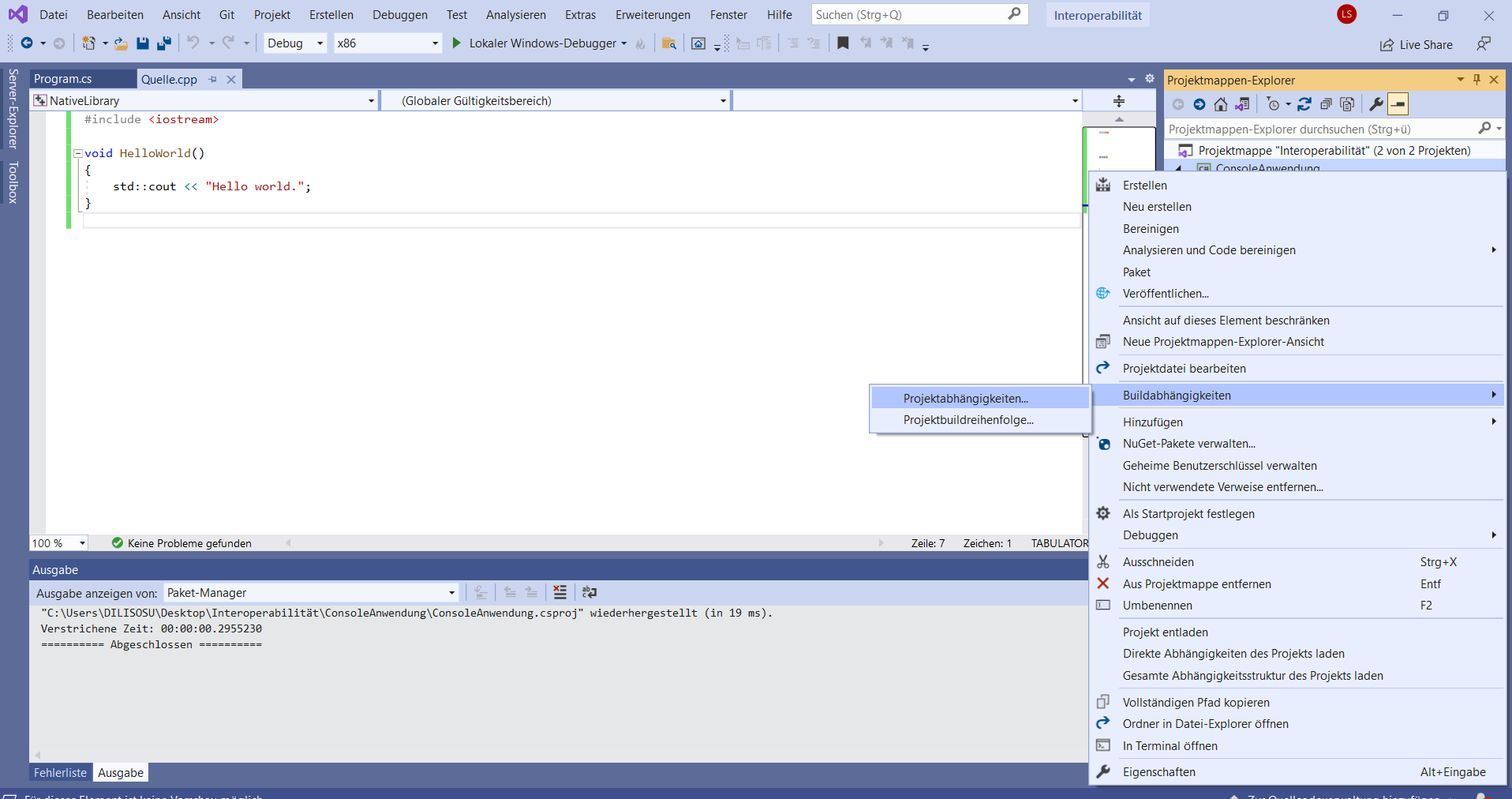
Das sagt, wohin der C++ .dll gelegt wird.

Schritt 6.

Hinzufügt ihr bitte eine C# Konsole Anwendung, und liegt ihr es als Startprojekt fest.

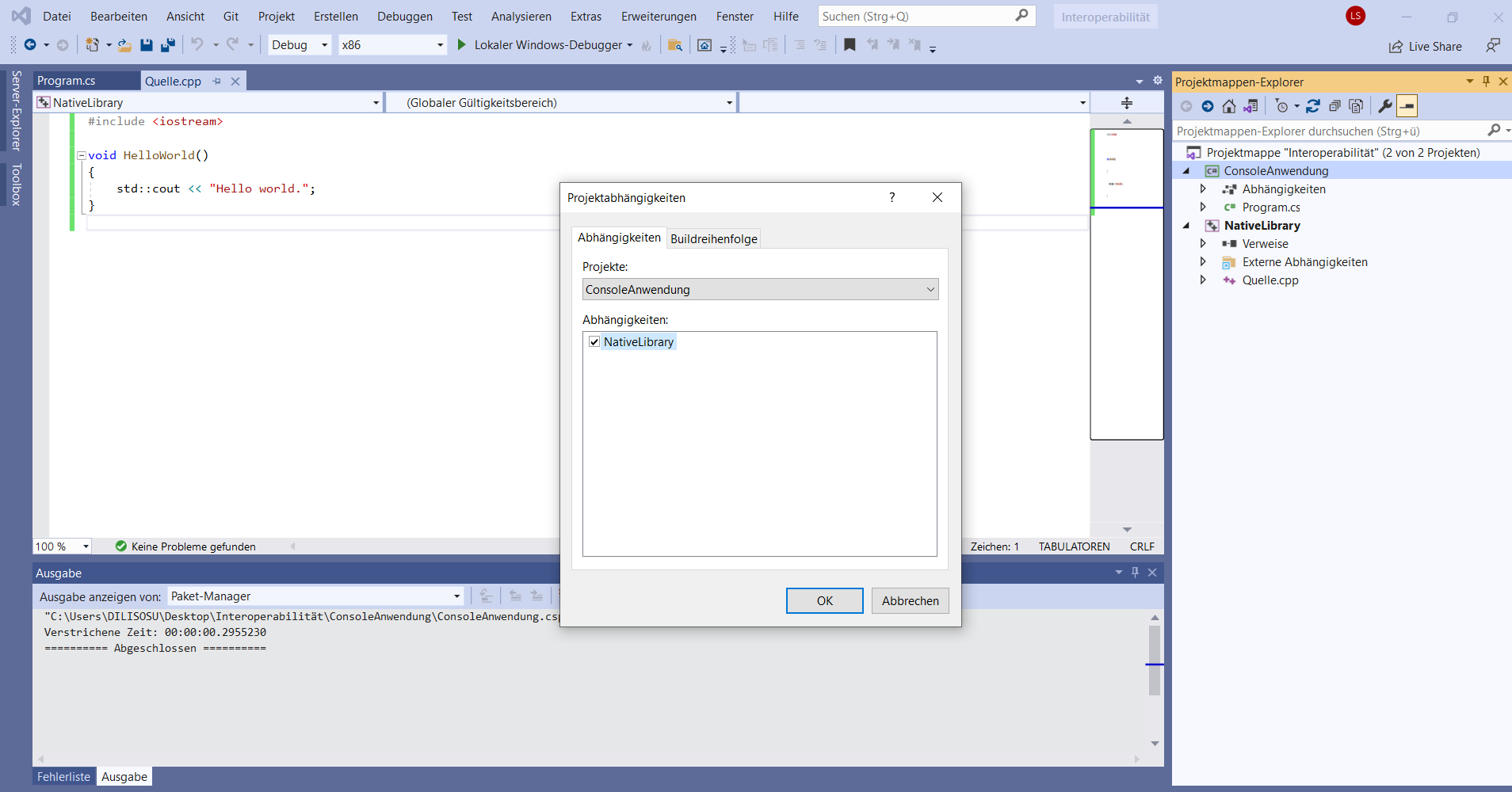
Schritt 7.

Bitte klickt ihr mit der rechten Maustaste auf das neues Projekt, und wählt bitte „Projektabhängigkeiten“.



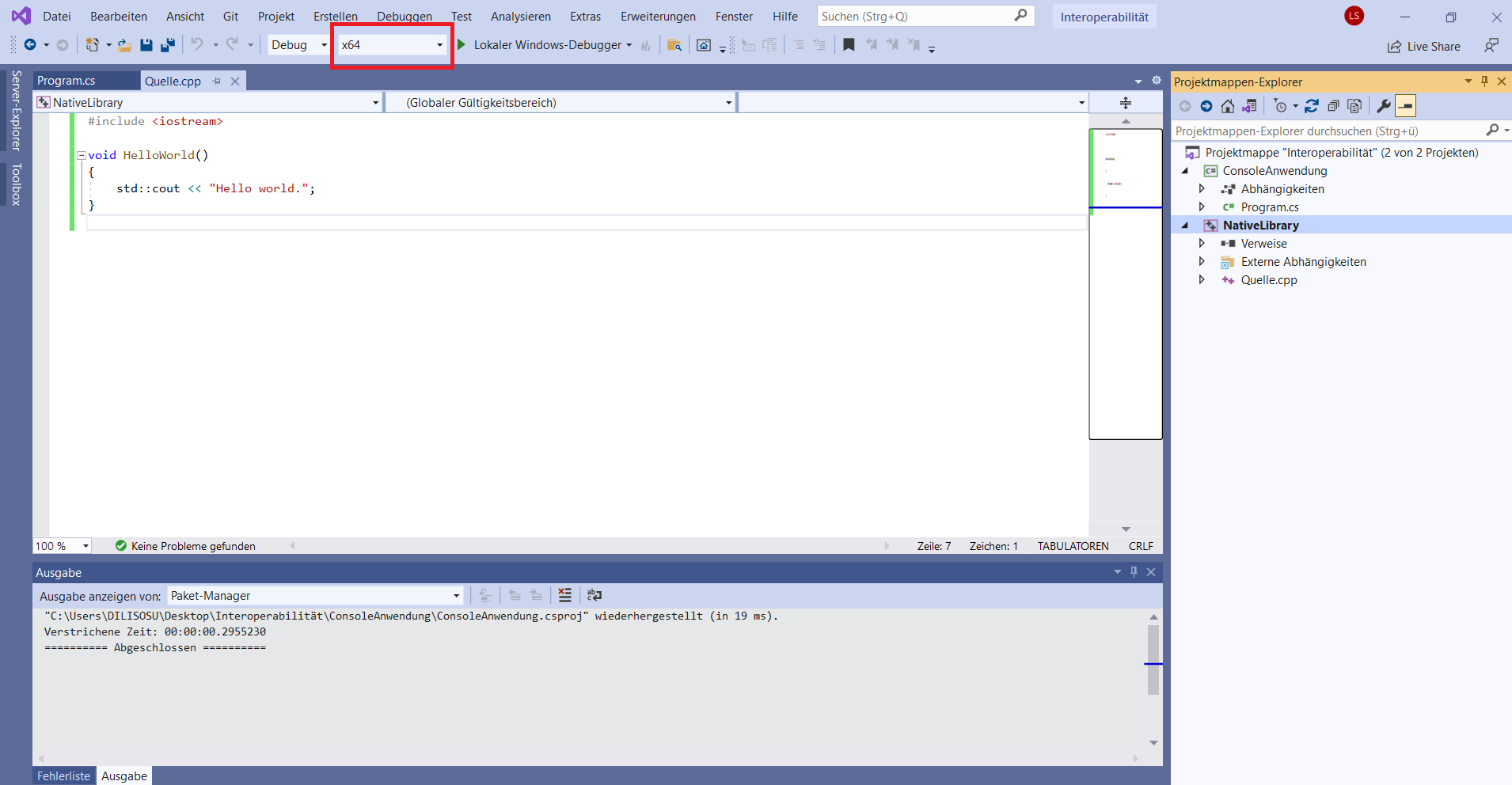
Schritt 8.

Bitte wählt ihr die C++-Bibliothek wie im Bild unten gezeigt aus. Wie bereits erwähnt, sollte der Code für diese Tutorials verfügbar sein, um einen Copy-Paste-Job zur Verfügung zu stellen.



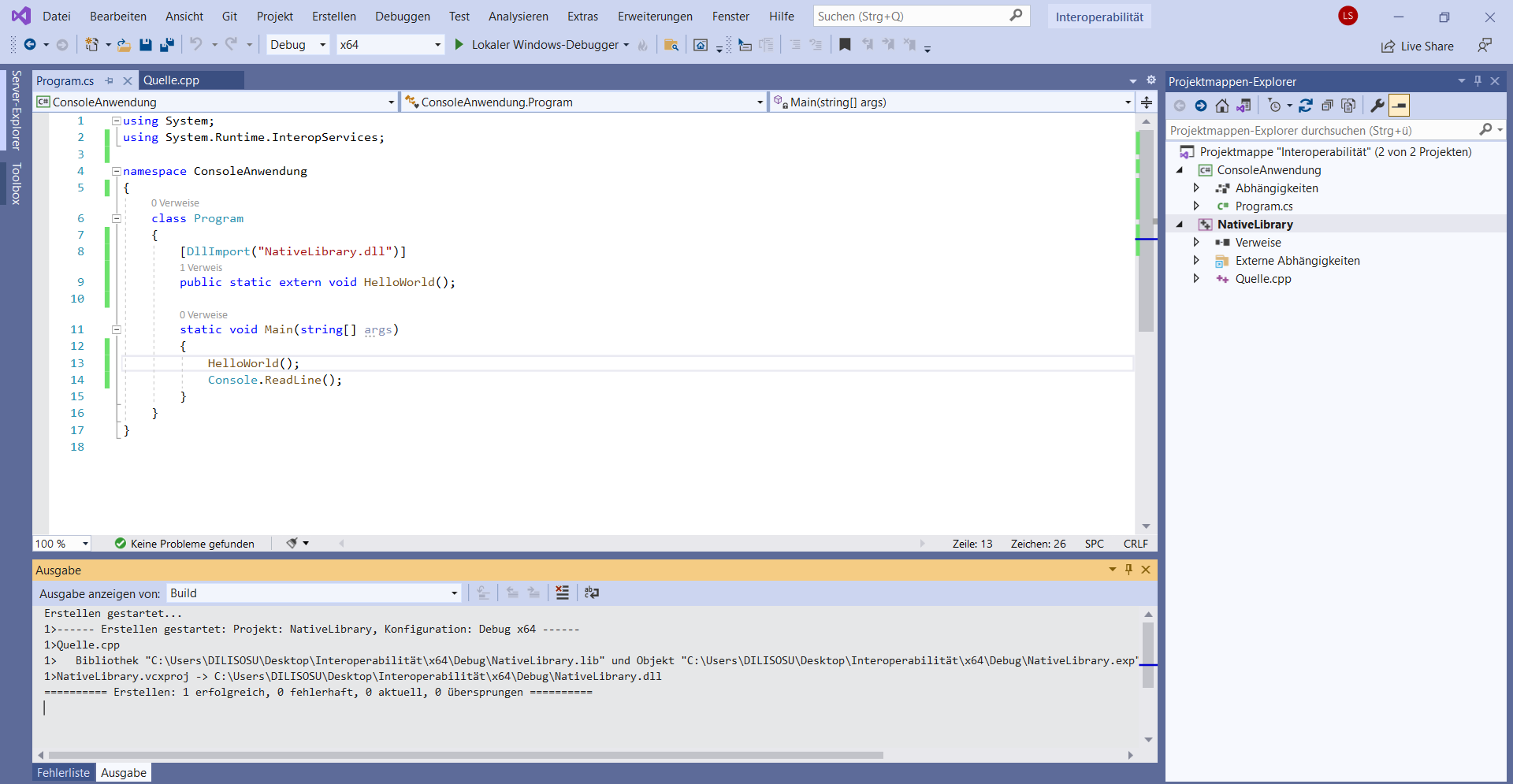
Schritt 9.

Wählt ihr bitte die Konfiguration wie im Bild unten gezeigt aus. Hinzufügt ihr bitte auch eine Quelle.cpp Date im eure C++ Projekt. Das soll wie unter aussehen.



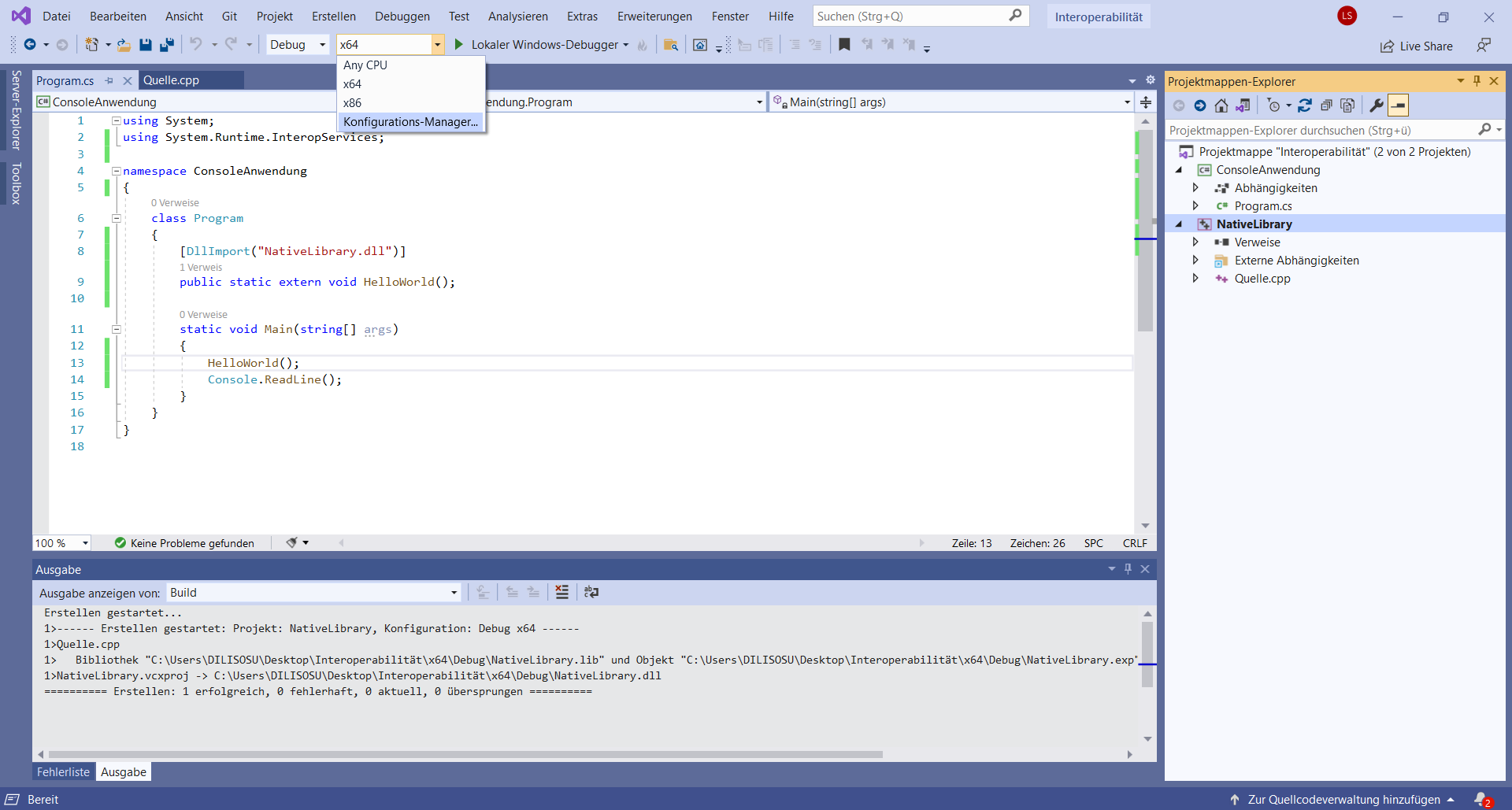
Schritt 10.

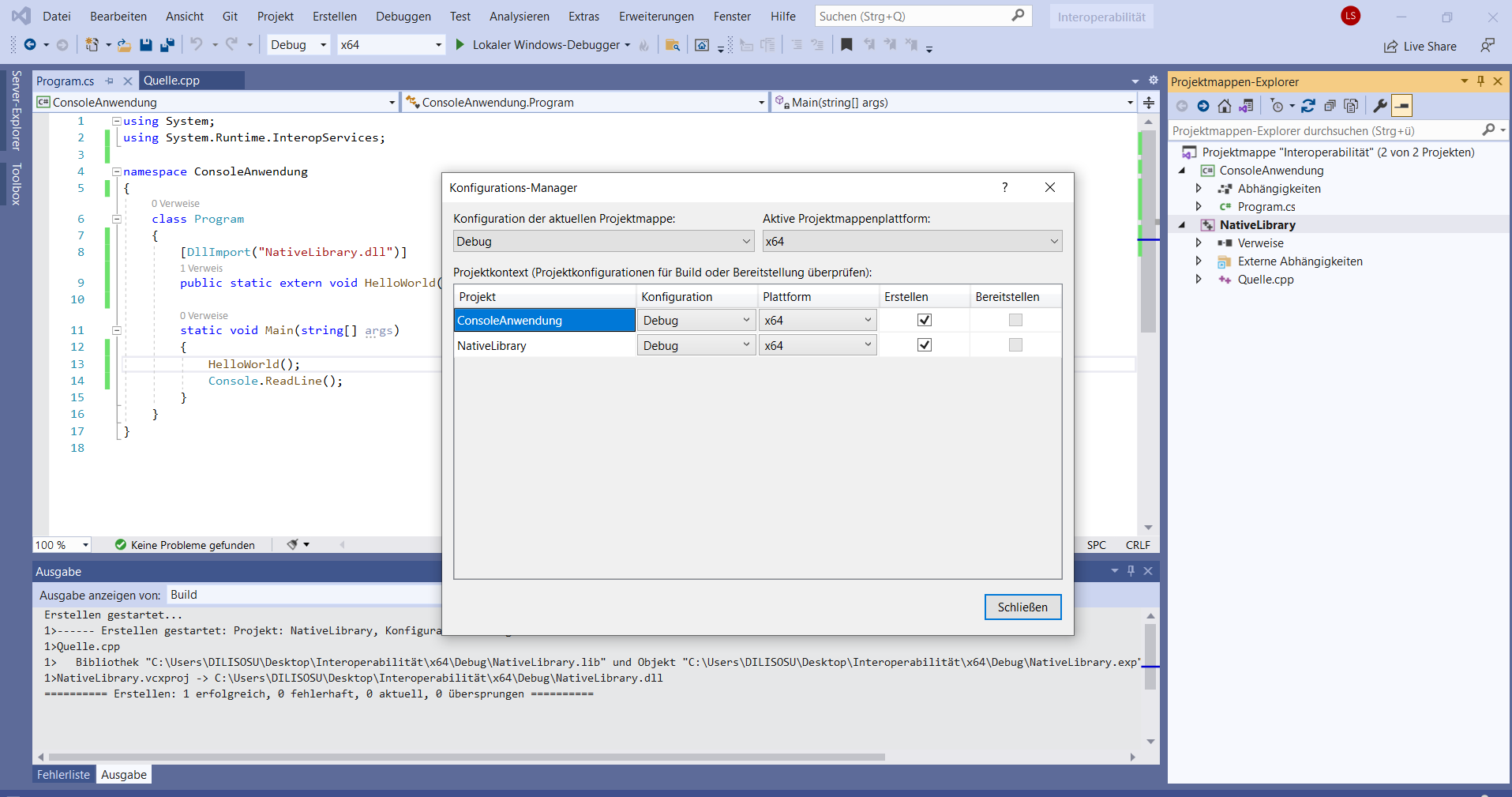
Bitte hinzufügt ihr In der Program.cs Klasse den C# Kode wie im Bild unten gezeigt aus.



Schritt 11.

Hinzufügt ihr bitte die nötige Konfiguration, wie die unten Bilden gezeigt aus.



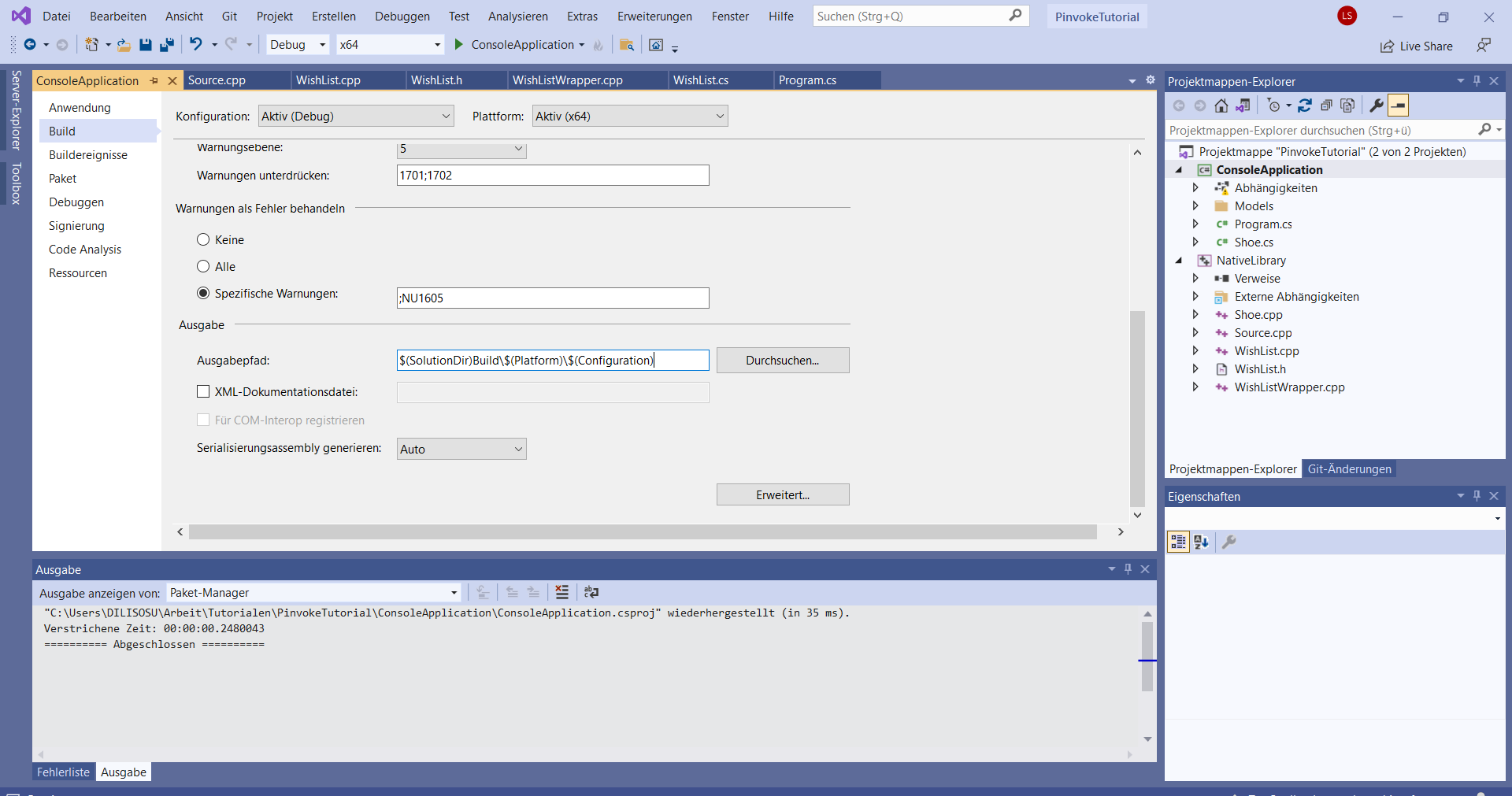


Schritt 12.

Bitte klickt ihr mit der rechten Maustaste auf das C# Projekt, und wählt bitte „Eingeschaften“. Bitte wählt das „Build“ Tab aus. Hinzufügt bitte auch die folgende Pfad

*$(SolutionDir)Build\$(Platform)\$(Configuration)*

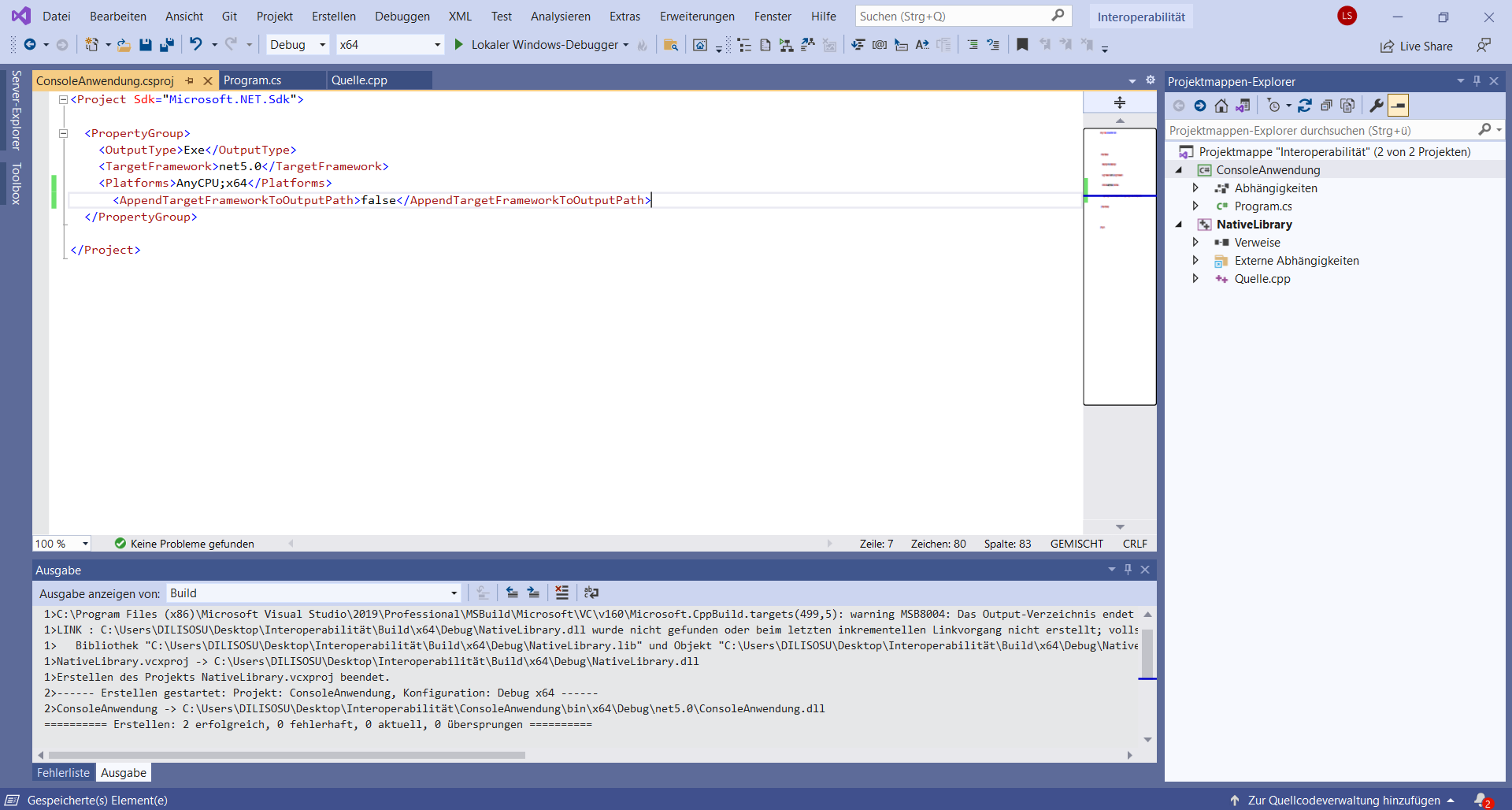
wie im Bild gezeigt aus.



Schritt 13.

Bitte doppelklickt ihr auf das C# Project, um den .csproj Datai zu bearbeiten, und hinzufügt ihr bitte die folge Zeile:

<AppendTargetFrameworkToOutputPath>false</AppendTargetFrameworkToOutputPath>

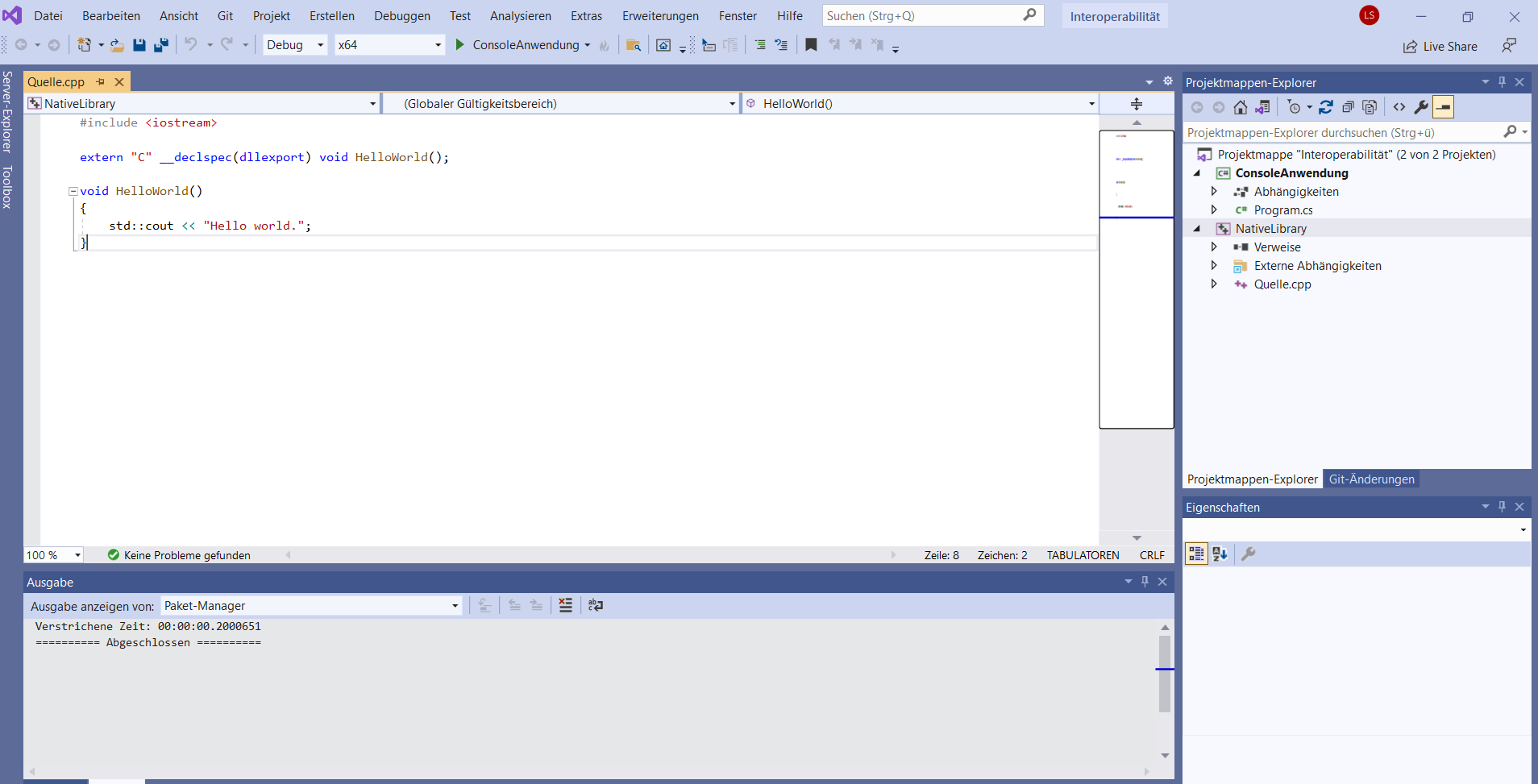
Das soll wie unten aussehen

Schritt 14.

Im Quelle.cpp von dem C++ Project (in unserem Fall - NativeLibrary) hinzufügt ihr bitte die folge Zeile:

extern "C" \_\_declspec(dllexport) void HelloWorld();

Es gibt unten einen Bild für eine besseren Verständnis:



Für Fragen stehen ich gerne zu eurer Verfügung. Bitte meldet ihr bei liviu.sosu@zeiss.com