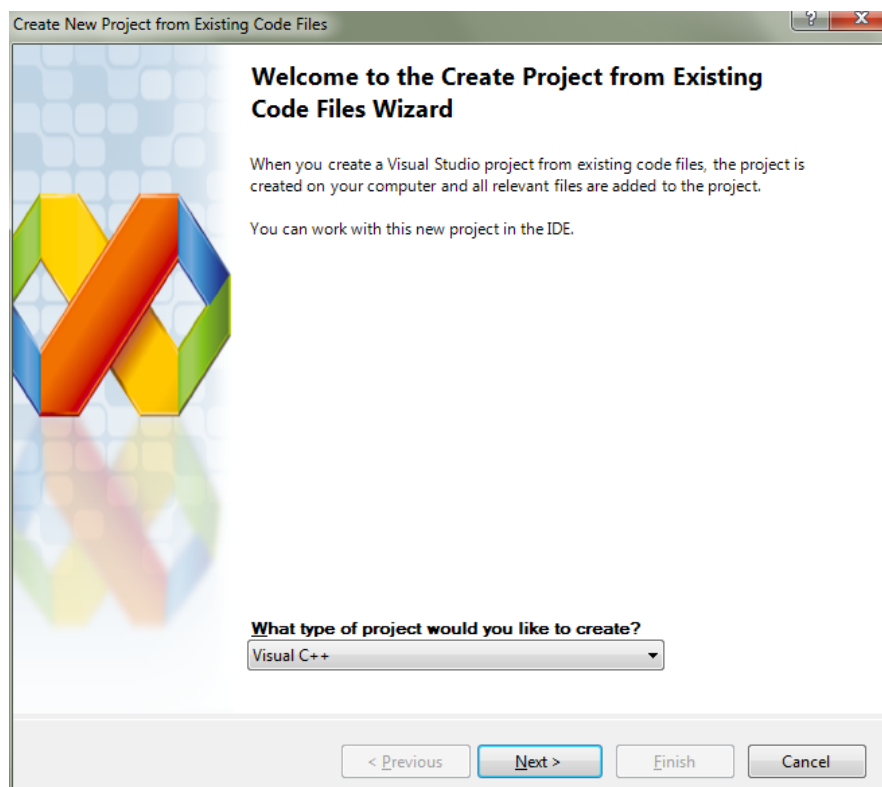


## How-to Create a Visual Studio Solution from UDK2010 code

Browsing UDK2010 code and finding function and variable definitions is a tedious task when the right tool isn't used. Visual Studio\* 2008 and later versions have the ability to create a solution from an existing code tree, which makes project browsing easier. Visual Studio also provides right-click support on variables and functions to go to the definition of the item.

1. Start Visual Studio.
2. Select **File → New → Project From Existing Code**.
3. Select project type as Visual C++.



4. Click Next.

5. Click the **Browse** button next to the **Project file Location** field and select the folder where the UDK2010 code is located.
6. Enter a project name in the **Project Name** field.
7. Specify *"File types to add to this project"* as:  
\*.c;\*.h;\*.asm;\*.asm16;\*.s;\*.inc;\*.asl;\*.aslc;\*.fdf;\*.dsc;\*.dec;\*.inf;  
\*.uni;\*.hfr;\*.vfr;\*.dxs  
**Note:** If you want to include the python files, then append ;\*.py to the file type list.

Create New Project from Existing Code Files

### Specify Project Location and Source Files

You can choose the files from one or more folders.

**Project file location:**

**Project name:**

☒ Add files to the project from these folders

Folders:

Add subfolders	Folder

File types to add to the project:

☒ Show all files in Solution Explorer

8. Click **Next**.

9. Select the **Use external build system** option.

The screenshot shows a Windows-style dialog box titled "Create New Project from Existing Code Files". Inside, the "Specify Project Settings" section explains that these details determine how the project is built. Under the heading "How do you want to build the project?", there are two radio button options. The first option, "Use Visual Studio", is currently selected. It includes a "Project type:" dropdown menu set to "Windows application project", and three unchecked checkboxes for "Add support for ATL", "Add support for MFC", and "Add support for the Common Language Runtime". Below these is a "Common Language Runtime Support:" dropdown menu set to "Common Language Runtime Support". The second option, "Use external build system", is also visible but not selected. It includes a note: "To specify build command lines, click Next to set the settings on the 'Specify Debug Configuration Settings' and 'Specify Release Configuration Settings' pages." At the bottom of the dialog are four buttons: "< Previous", "Next >" (which is highlighted with a blue border), "Finish", and "Cancel".

Create New Project from Existing Code Files

**Specify Project Settings**

These details determine how the project is built and the type of the project created.

**How do you want to build the project?**

☐ Use **V**isual Studio

Project type:  
Windows application project

☐ Add support for **A**TL

☐ Add support for **M**FC

☐ Add support for the **C**ommon Language Runtime

Common Language Runtime Support:  
Common Language Runtime Support

☒ Use external build system

To specify build command lines, click Next to set the settings on the "Specify Debug Configuration Settings" and "Specify Release Configuration Settings" pages.

< Previous   Next >   Finish   Cancel

10. Click **Next**.

11. Create a batch file B.BAT in the *UDK2010* folder, and add the following lines (change options to match your build environment)
 

```
@call edksetup.bat
@rem if you are using a platform that uses EDK components
@rem or you need the EdkCompatibilityPkg libraries
@set EFI_SOURCE=%CD%\MyPlatformPkg
build %1 -p MyPlatformPkg\MyPlatformPkg.dsc
--log=build.log
```
12. Input "B.BAT" as the "Build command line"
13. Input "B.BAT all" as the "Rebuild command line"
14. Input "B.BAT clean" as the "Clean command line"

**Create New Project from Existing Code Files**

**Specify Debug Configuration Settings**  
Set the debug configuration's settings.

**What settings do you want for the Debug configuration?**

Build command line:	Preprocessor definitions (/D):
<input type="text" value="b.bat"/>	<input type="text"/>
Rebuild command line:	Include search paths (/I):
<input type="text" value="b.bat all"/>	<input type="text"/>
Clean command line:	Forced included files (/FI):
<input type="text" value="b.bat clean"/>	<input type="text"/>
Output (for debugging):	.NET assembly search paths (/AI):
<input type="text"/>	<input type="text"/>
	Forced using .NET assemblies (/FU):
	<input type="text"/>

When using an external build system, the command lines are the commands that will be executed when a build action occurs and the output specifies the name of the executable to debug.

The preprocessor definitions, include search paths, forced included files, assembly search paths, and forced using assemblies are used for IntelliSense. These settings also control how Visual Studio builds your project when not using an external build system.

< Previous   Next >   Finish   Cancel

15. Click **Finish** to complete creating a new project. The folder you selected will be scanned for the specified file extensions.  
You will now have a Visual Studio solution for your UDK2010 code tree.
16. Right-click on functions, variables, etc. and select **Go To Definition** to look up their definition if desired.
17. Click **Build → Build Solution** or the **Build Solution** button on the menu bar to build the code.