UnrealVS Extension

Guide for installing, setting up, and using the UnrealVS extension for Visual Studio.

The **UnrealVS** extension for **Visual Studio (VS)** can be used for easy access to common actions when developing with **Unreal Engine (UE)**. This document will explain how to install the extension and how to use it with your project.

Content Covered:

- Extension installation and setup
- Setting the Startup Project
- Bindable command for building the Startup Project
- Setting the command line arguments
- Batch building projects
- Quick Build project menu

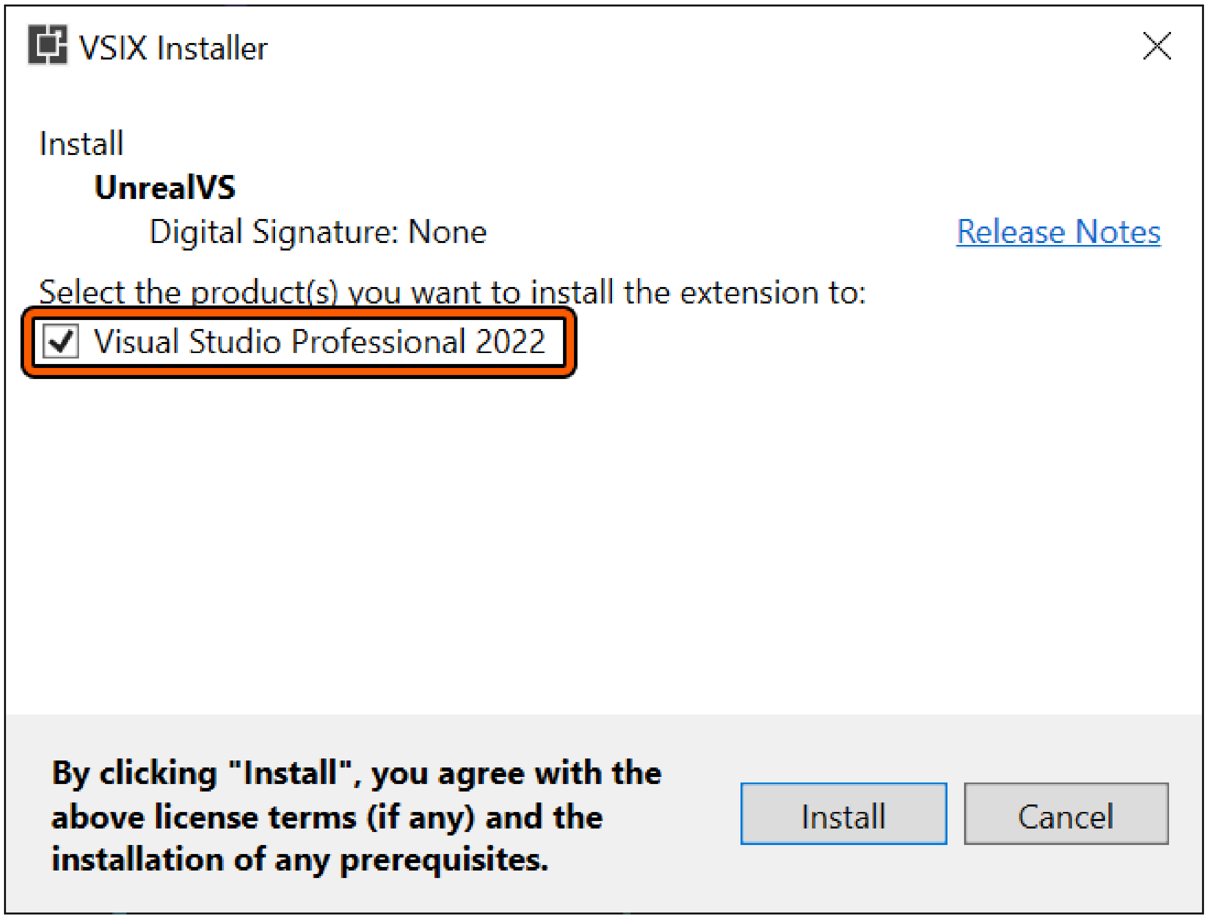
The UnrealVS extension does not work with Visual Studio Express. It is only compatible with Visual Studio Community and Professional editions.

Installation and Setup

The following installation and setup steps were drafted while using VS Community 2022 Version 17.4.1. Depending on which version of VS you are using, these steps might vary.

Install UnrealVS Visual Studio Extension

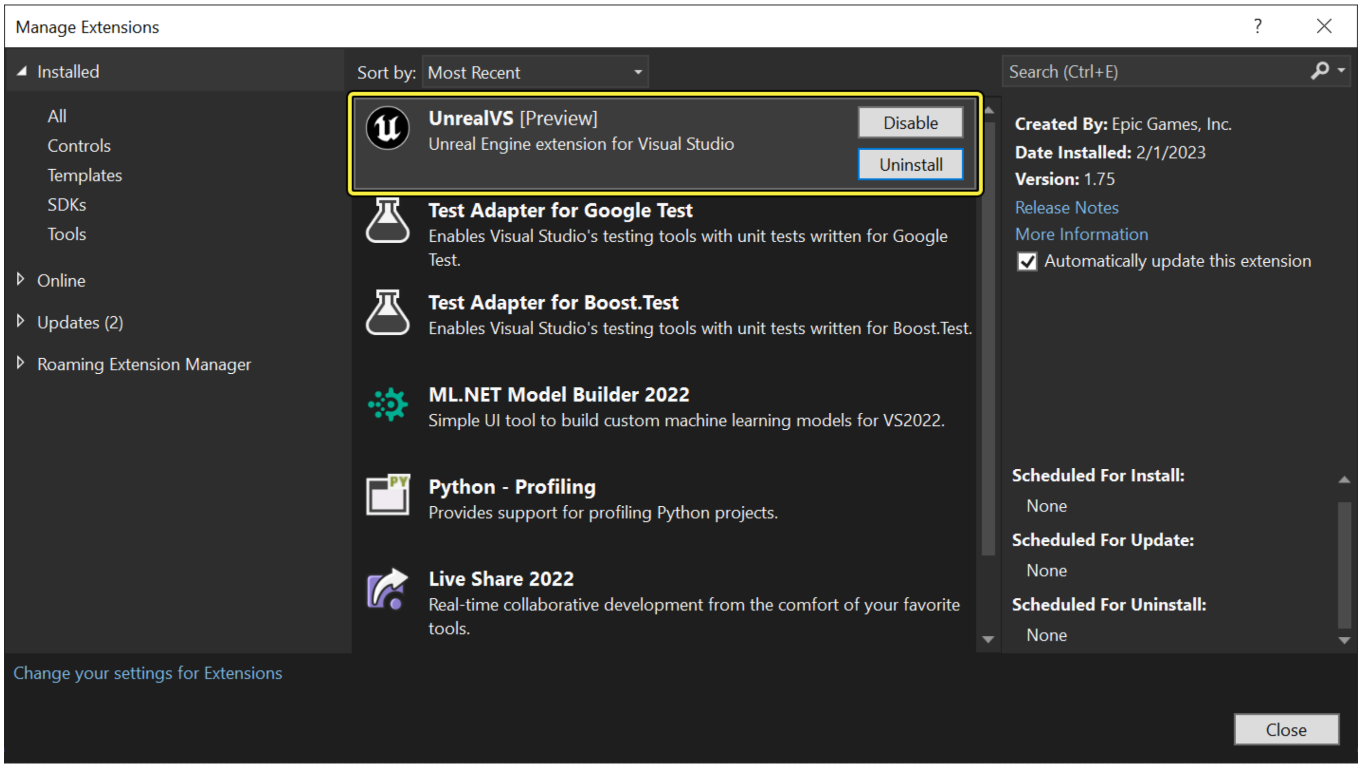
1. Find the extension for your version of Visual Studio located at `<UNREAL_ENGINE_ROOT>\Engine\Extras\UnrealVS\VS_VERSION\UnrealVS.vsix` For example, `C:\Program Files\Epic Games\UE5\Engine\Extras\UnrealVS\VS2019\UnrealVS.vsix`.
2. Run the `UnrealVS.vsix` file by double-clicking it to begin the installation.
3. The UnrealVS extension should detect and target your version of VS. Check that the installer is targeting the correct version of VS and that the box is ticked. If everything looks good, click **Install** to proceed.


VSIX Installer

Install
UnrealVS
Digital Signature: None
Select the product(s) you want to install the extension to:
[✓] Visual Studio Professional 2022

By clicking "Install", you agree with the above license terms (if any) and the installation of any prerequisites.

4. Click the **Close** button once the installation is complete.
5. Run VS and in **Extensions** > **Manage Extensions** > **Installed** you should see the extension.


Manage Extensions ? ✕

Installed
- All
- Controls
- Templates
- SDKs
- Tools

Online
Updates (2)
Roaming Extension Manager

Sort by: Most Recent

Search (Ctrl+E)

UnrealVS [Preview]
Unreal Engine extension for Visual Studio
Disable
Uninstall

Test Adapter for Google Test
Enables Visual Studio's testing tools with unit tests written for Google Test.

Test Adapter for Boost.Test
Enables Visual Studio's testing tools with unit tests written for Boost.Test.

ML.NET Model Builder 2022
Simple UI tool to build custom machine learning models for VS2022.

Python - Profiling
Provides support for profiling Python projects.

Live Share 2022
Real-time collaborative development from the comfort of your favorite tools.

Created By: Epic Games, Inc.
Date Installed: 2/1/2023
Version: 1.75
Release Notes
More Information
☑ Automatically update this extension

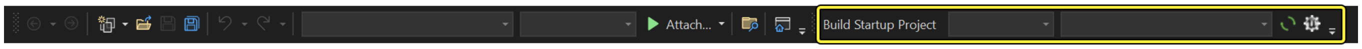
Scheduled For Install:
None
Scheduled For Update:
None
Scheduled For Uninstall:
None

Change your settings for Extensions

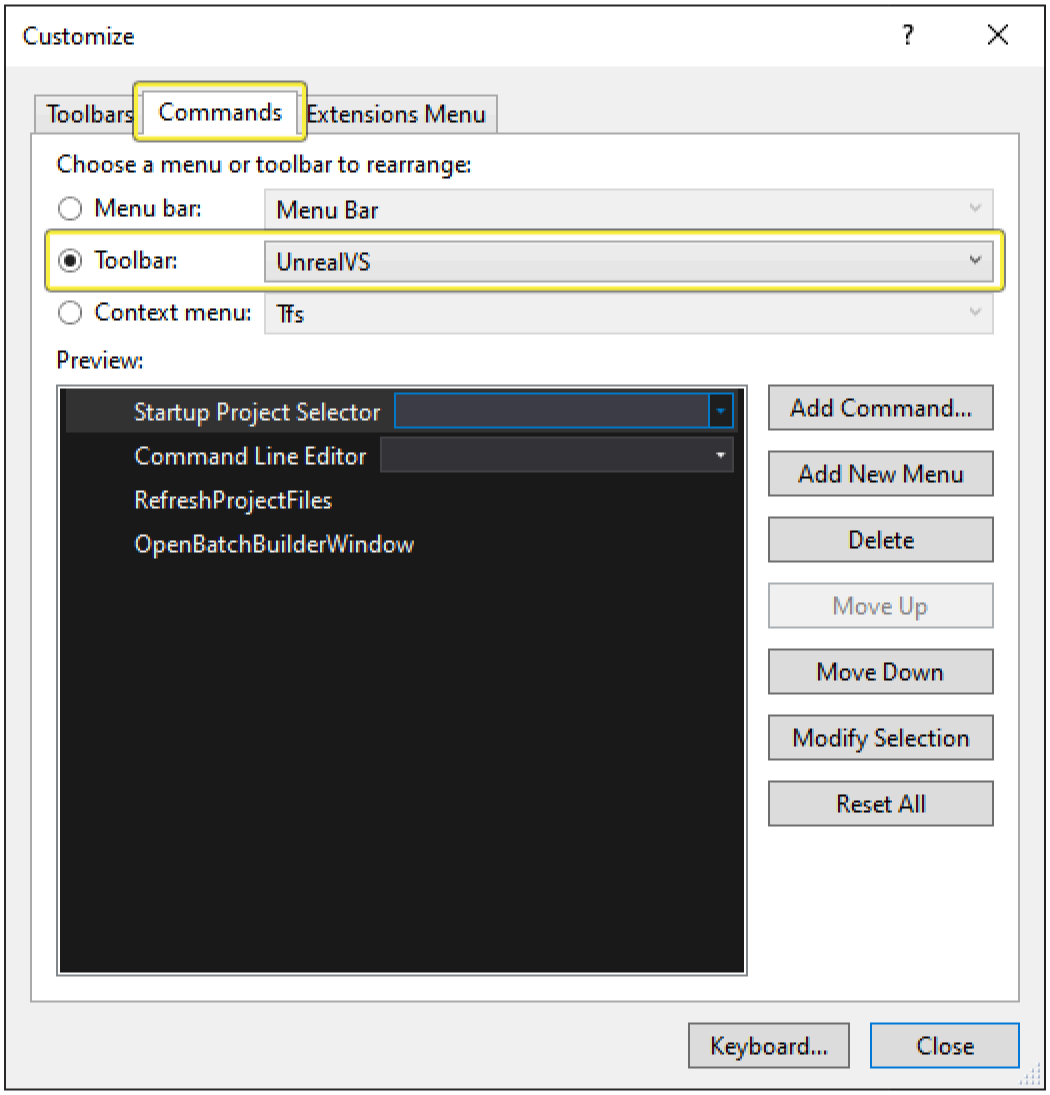
Close

If Visual Studio is already running, you will need to restart before the extension will be loaded and displayed. The extension is also shown in the Installed Products list in the **Help** > **About Microsoft Visual Studio** dialog.

6. Go to **View > Toolbars** (or **Right-click** the Visual Studio toolbar area) and select **UnrealVS** to display the extension's toolbar.



7. By default the toolbar appears as above. However, its contents can be customized by going to **Tools** > **Customize...** > **Commands tab** > **Toolbar radio button** > **UnrealVS.**



Choose a menu or toolbar to rearrange:

Menu bar: Menu Bar

[✓ Toolbar:] UnrealVS

Context menu: Tfs

Preview:

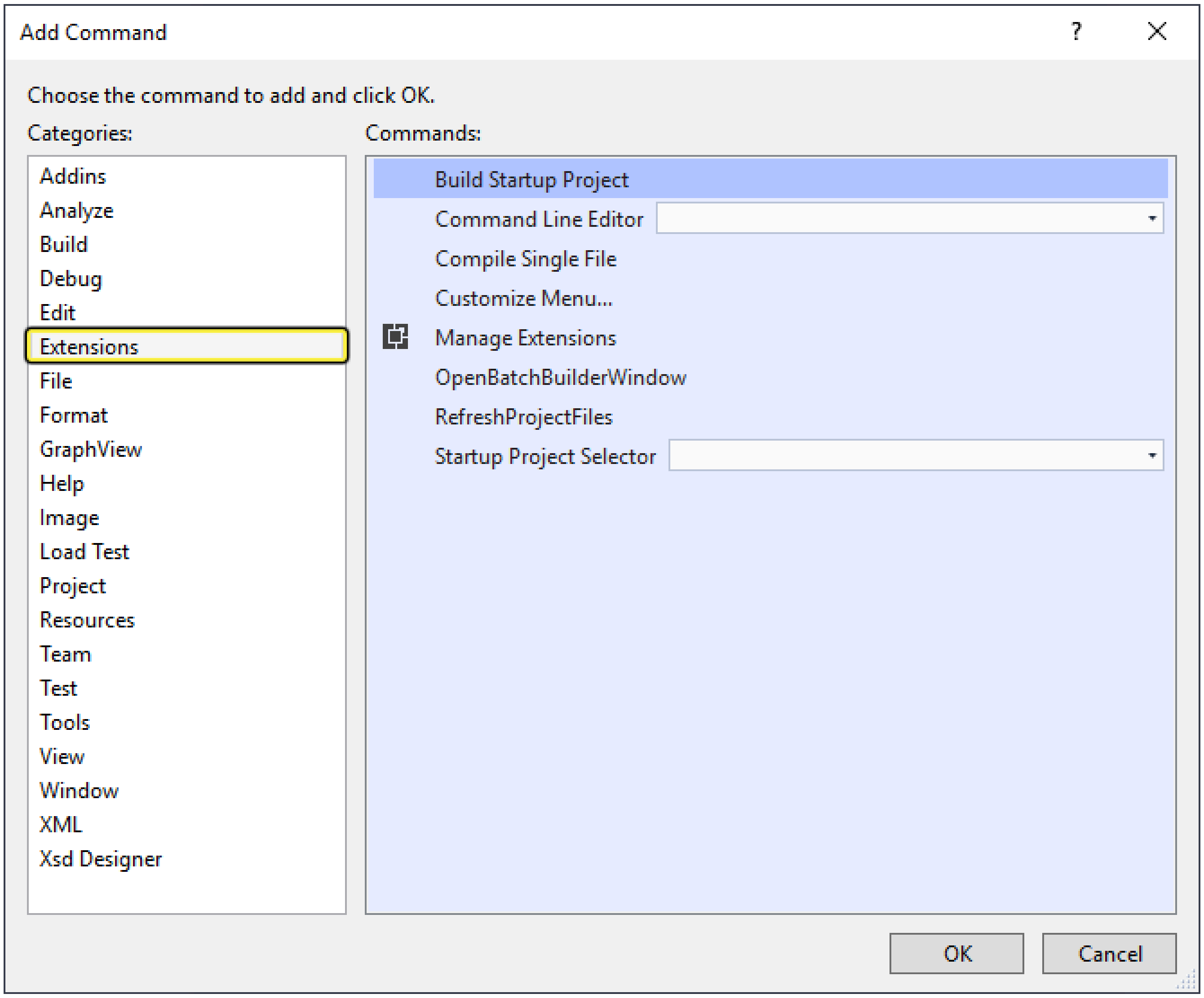
Startup Project Selector

Command Line Editor

RefreshProjectFiles

OpenBatchBuilderWindow

8. In the **Customize** box, choose **Add Command...** and select **Extensions** from the **Categories** list to see a list of UnrealVS commands that can be added to the toolbar. When you have finished adding commands, click **OK**.

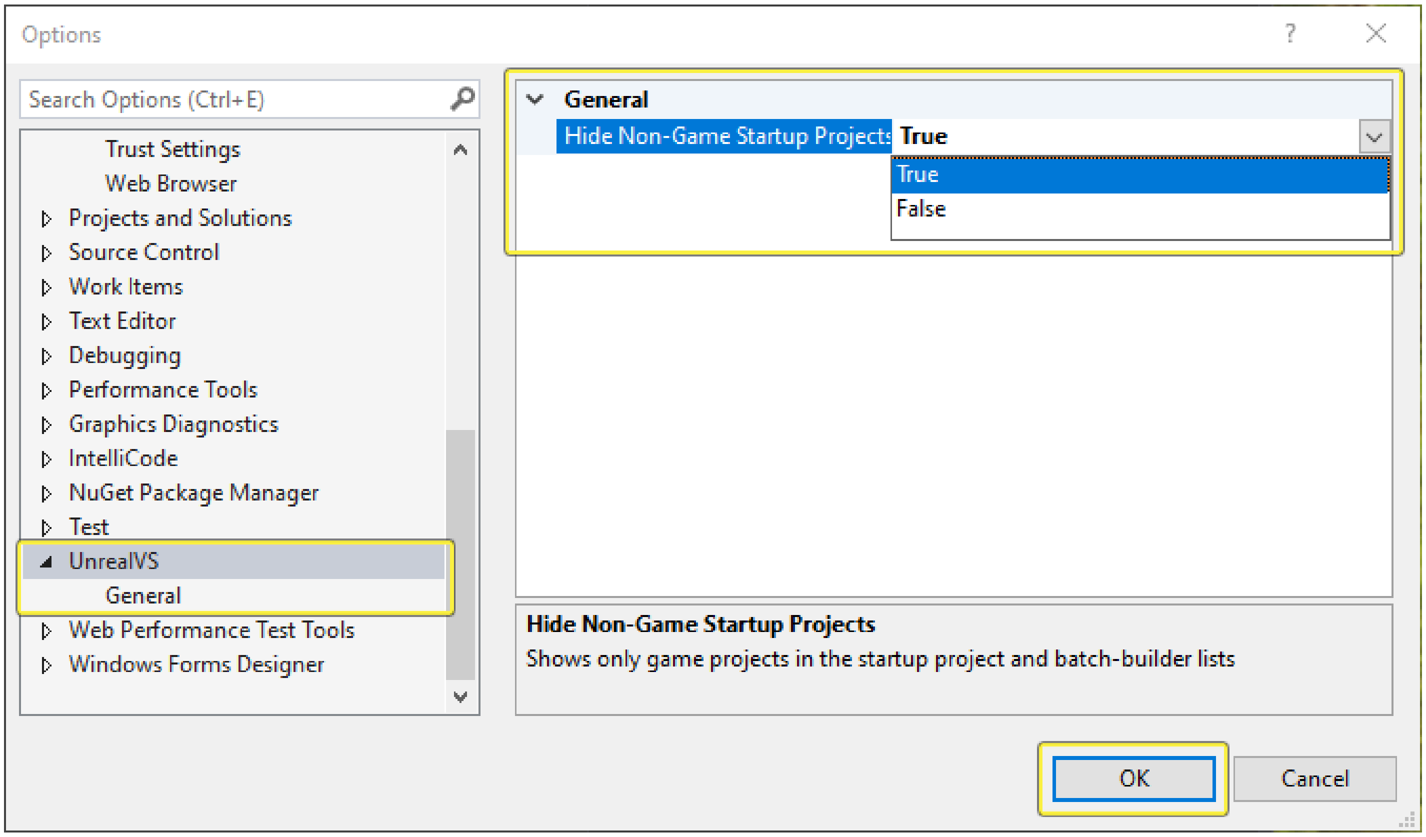


Now you can add various commands, such as the Startup Project, various command line arguments, Refresh Projects, Batch Builder, and the Quick Build Menu.

Setting the Startup Project

The **Startup Project** dropdown is a means of quickly setting or switching between startup projects. It automatically lists all available projects that build executables in the solution. Selecting a project from the list sets that as the current startup project.

You can change which projects are listed in the UnrealVS options. To show only Game projects, go to **Tools > Options...** and select **UnrealVS**. Click the dropdown next to **Hide Non-Game Startup Projects** and select **True**. Then click **OK**.



Hide Non-Game Startup Projects: True

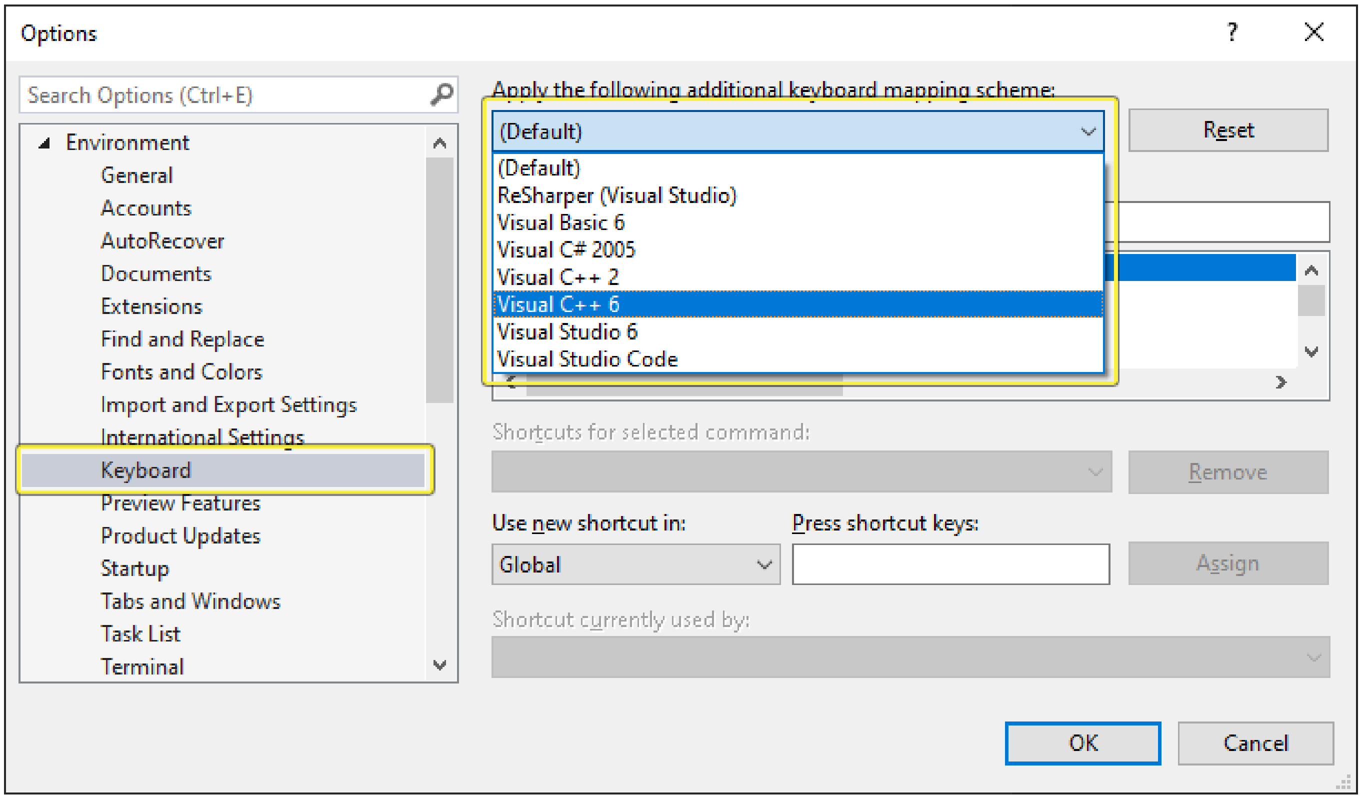
Hide Non-Game Startup Projects
Shows only game projects in the startup project and batch-builder lists

Building the Startup Project

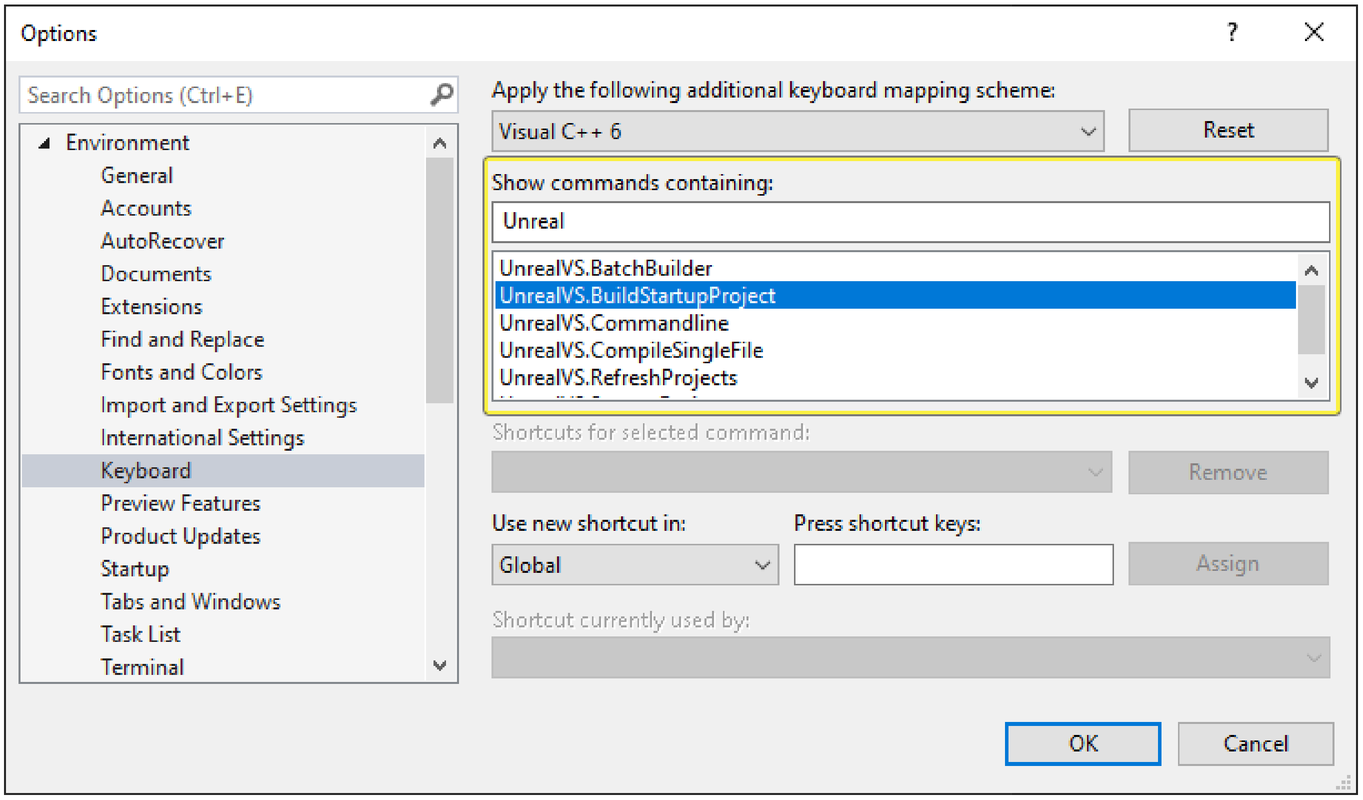
The extension also contains a command for building the current startup project which can be bound to a hotkey or other means of running custom commands.

Binding the Command to a Hotkey

1. Go to **Tools > Options...** and select **Keyboard** under the **Environment** tree. Under **Apply the following additional keyboard mapping scheme** click the dropdown and select **Visual C++ 6**.

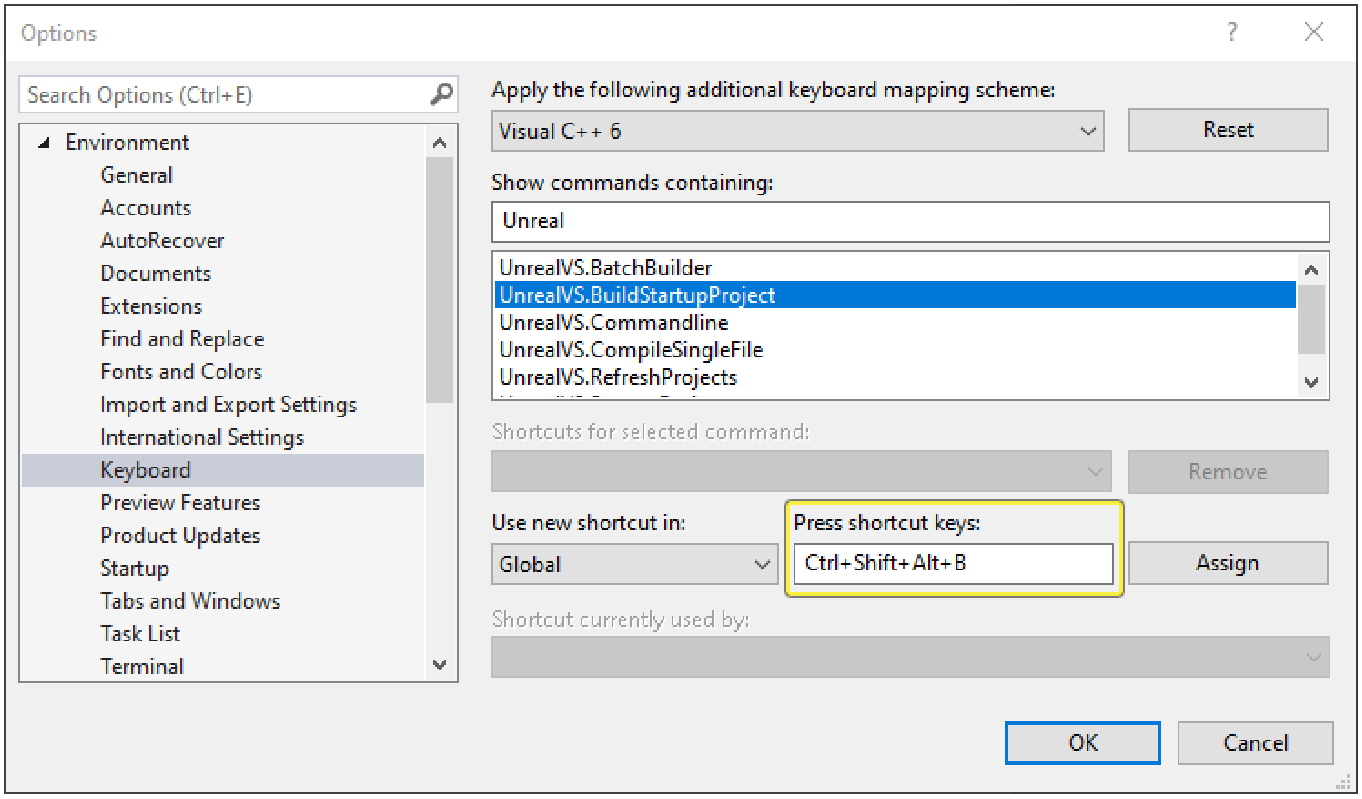


2. Select the **UnrealVS.BuildStartupProject** command in the commands list.



You can type "Unreal" in the search box to filter the list.

3. With the command selected, click in the **Press Shortcut Keys** box and then press the combination of keys you want to use to perform the command (for example, **Ctrl + Shift + Alt + B** is shown in the example below).



Apply the following additional keyboard mapping scheme: Visual C++ 6

Show commands containing: Unreal

UnrealVS.BatchBuilder
UnrealVS.BuildStartupProject
UnrealVS.Commandline
UnrealVS.CompileSingleFile
UnrealVS.RefreshProjects

Shortcuts for selected command:

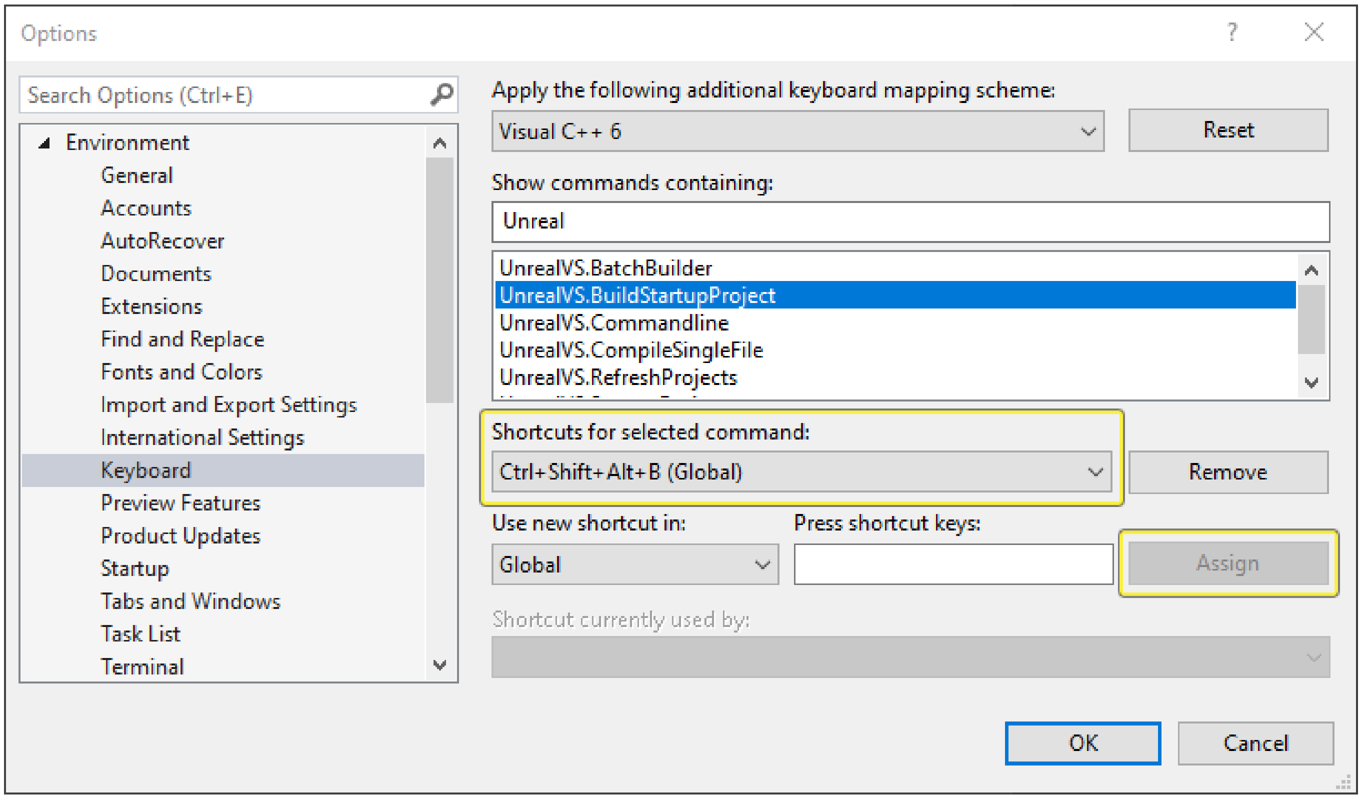
Use new shortcut in: Global

Press shortcut keys: Ctrl+Shift+Alt+B

Shortcut currently used by:

You should choose a combination of keys not already assigned to another command. The **Shortcut currently used by** box will alert you to the fact that the chosen combination is already in use.

4. Press the **Assign** button to bind the keys to the command. The key combination is displayed under **Shortcuts for selected command**.


Options

Search Options (Ctrl+E)

Environment
- General
- Accounts
- AutoRecover
- Documents
- Extensions
- Find and Replace
- Fonts and Colors
- Import and Export Settings
- International Settings
- Keyboard
- Preview Features
- Product Updates
- Startup
- Tabs and Windows
- Task List
- Terminal

Apply the following additional keyboard mapping scheme: Visual C++ 6

Show commands containing: Unreal

UnrealVS.BatchBuilder
UnrealVS.BuildStartupProject
UnrealVS.Commandline
UnrealVS.CompileSingleFile
UnrealVS.RefreshProjects

Shortcuts for selected command: Ctrl+Shift+Alt+B (Global)

Use new shortcut in: Global

Press shortcut keys:

Shortcut currently used by:

5. Press the **OK** button to save the changes. Now, when you use the shortcut, the project set as the startup project will automatically build.

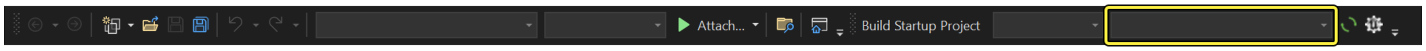
Iterating on a Single Source File

If you are used to iterating on updates to a single source (`.cpp`) file using Ctrl+F7 in Visual Studio, you can bind **Ctrl+F7** to the **UnrealVS.CompileSingleFile** command. This will build your active project with the current solution configuration, platform, and the `-singlefile=PATH/TO/OPEN/FILE` argument.

Command Line Arguments

The **Command-Line** control is used to set the **Command Arguments** for the current project when running a debugging session. You can use this shortcut rather than setting them through the project **Properties**. In fact, changes made here will be reflected automatically in the **Properties**.

Enter the arguments in the text box or choose from a list of recent arguments using the dropdown list. These arguments will be passed to the executable when a debugging session is started.



Project configurations for game projects that build Unreal Editor add the project name to the command line automatically, to tell the editor executable which project you are using.

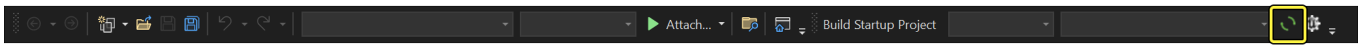
For example, if you build QAGame using the build config 'Debug Editor', the command line will have QAGame.uproject added to the command arguments without the need to type it in the box. To launch the editor for QAGame, you can leave the command-line control empty and the editor will still know which project you are running.

Refresh Projects

Project files can be generated inside of Visual Studio using the Automatic Project File Generation. This makes it much quicker to sync and update all the project files, since you do not have to navigate to the `.bat` file and run it manually.

To refresh project files:

1. Press the **Refresh Projects** button on the **UnrealVS** toolbar.



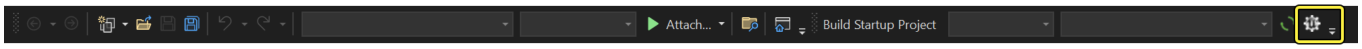
2. The project generation process will be displayed in the **Output** window as it occurs.
3. Reload any projects that require it, as you are prompted.

Batch Builder

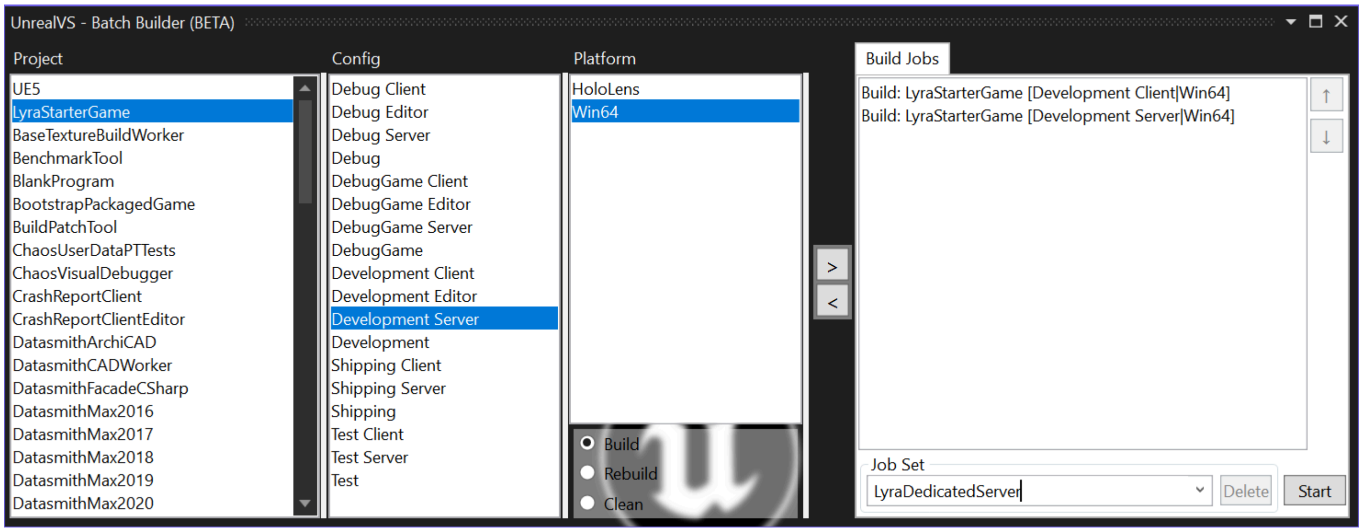
Create and run custom sets of build jobs with **UnrealVS Batch Builder**. It is more versatile than the **Build > Batch Build...** option in Visual Studio.

To open the Batch Builder window:

1. Press the Batch Builder button on the **UnrealVS** toolbar or use a keyboard shortcut that you have assigned to the command UnrealVS.BatchBuilder (see **Build Startup Project** above for instructions on setting up keyboard shortcuts with **UnrealVS** commands).



2. The **UnrealVS Batch Builder** window appears.

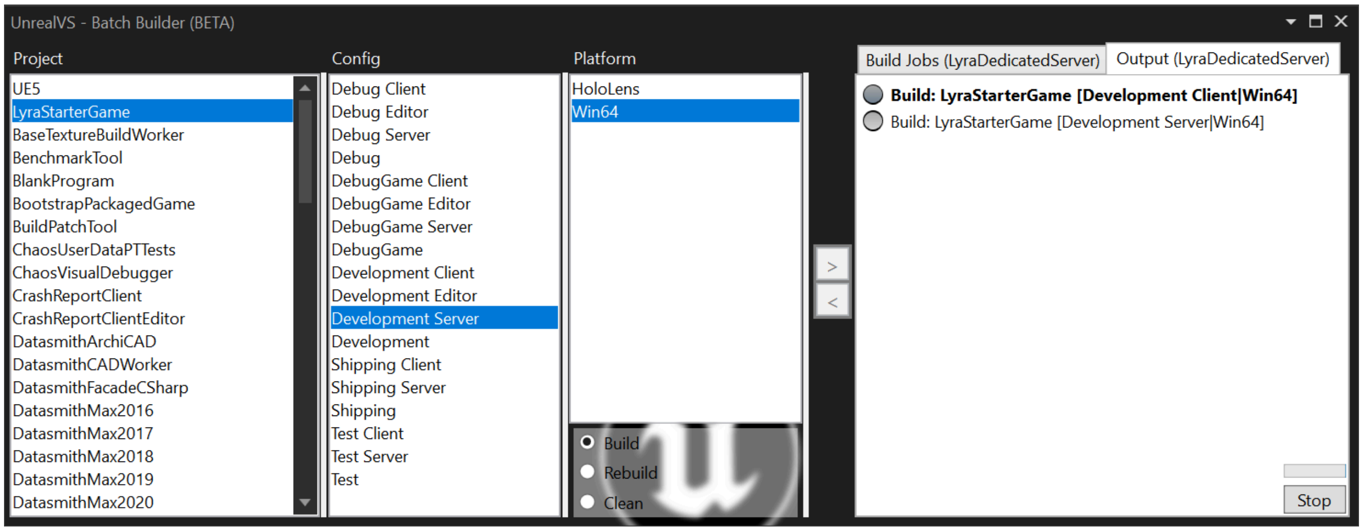


Build: LyraStarterGame [Development Client|Win64]
Build: LyraStarterGame [Development Server|Win64]

LyraDedicatedServer

- Create a **Build Job** by selecting from the **Project**, **Config**, **Platform**, and then clicking the radio button for the job type.
- Add/remove jobs with the **>** and **<** buttons.
- In the **Build Jobs** list, move selected jobs up or down the list with the arrow buttons on the right.
- Select the **Job Set** you want to edit in the dropdown under the **Build Jobs** list.
- To create a new **Job Set,** type a new name into the field.
- The **Delete** button deletes the selected **Job Set** from the list.
- **Start or Stop** the batch build task on the current Job Set with the **Start** button.
- Job Sets are stored for next time in the .suo file for the loaded solution.

3. When you click **Start**, the **Output** tab displays, showing the progress of the batch build.



Build: LyraStarterGame [Development Client|Win64]
Build: LyraStarterGame [Development Server|Win64]

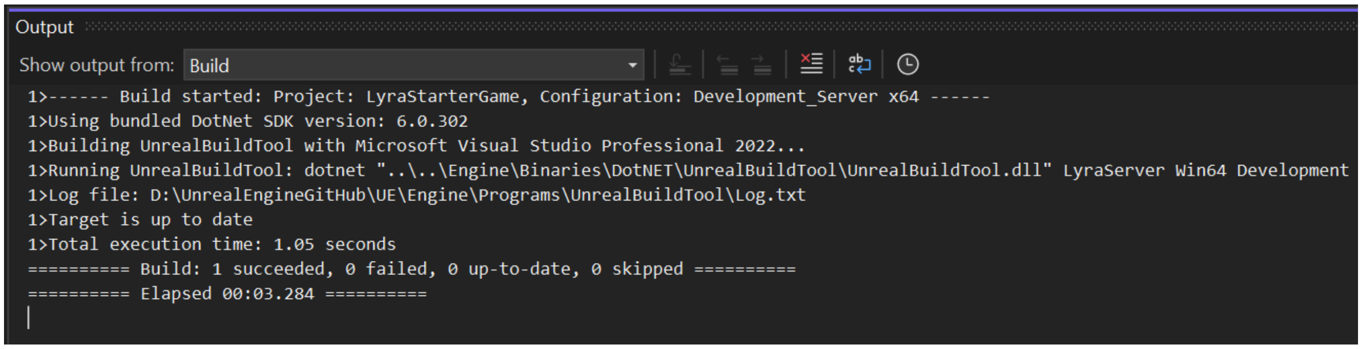
The **Build Jobs** in the running set are shown in the output list. The current executing build job is shown in bold.

- ● = Queued job

- 🟢 = Job complete, succeeded
- 🔴 = Failed job
- 🟠 = Cancelled job

A busy animation, and the **Stop** button, are shown when a batch build is running.

4. You can view the output from individual **Build Jobs** by double-clicking an item in the **Output tab**, or by right-clicking and selecting **Show Output** from the context menu.



```
1>------ Build started: Project: LyraStarterGame, Configuration: Development_Server x64 ------
1>Using bundled DotNet SDK version: 6.0.302
1>Building UnrealBuildTool with Microsoft Visual Studio Professional 2022...
1>Running UnrealBuildTool: dotnet "..\..\Engine\Binaries\DotNET\UnrealBuildTool\UnrealBuildTool.dll" LyraServer Win64 Development
1>Log file: D:\UnrealEngineGitHub\UE\Engine\Programs\UnrealBuildTool\Log.txt
1>Target is up to date
1>Total execution time: 1.05 seconds
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
========== Elapsed 00:03.284 ==========
```

Batch Builder output for individual items is shown in the Visual Studio **Output** window, in a pane named **UnrealVS - BatchBuild**. This is not to be confused with the Build pane that shows the output of the current/last build job.

Quick Build Menu

Use the **Quick Build** menu to build a project using any config and platform combination, without having to change the main solution build config.

1. Right-click a Project node in the **Solution Explorer**.
2. The **UnrealVS Quick Build** menu contains sub-menus for the available platforms, and build configs, in the Unreal Engine solution.
3. Select an item to build the selected project, platform, and config. This method is much quicker than changing the Solution Configuration and Solution Platform in the IDE, starting a build, then switching the config and platform back again.

This method is much quicker than changing the Solution Configuration and Solution Platform in the IDE, starting a build, then switching the config and platform back again.