Tutorial: Setting up your development environment

This tutorial walks you through the steps to prepare your development environment for building the Lumberyard SDK, including the engine, editor and sample project.

You will learn how to do the following:

- Set up Lumberyard in source control
- Run SetupAssistant
- Identify and install required 3rd party software
- Configure your development environment
- Generate a Visual Studio solution

Prerequisites

This tutorial assumes you have downloaded and installed the Lumberyard SDK.

You must have the following before starting this tutorial:

- Lumberyard installed
- Visual C++ Redistributable Packages for Visual Studio 2013
- Visual Studio 2013 with update 5 or higher

Step 1: Set up Lumberyard in source control

Before you being development with Lumberyard, it is highly recommended that you add the pristine version of Lumberyard to source control. Using source control is a best practice for game development and is essential for versioning and team collaboration.

When selecting a source control solution keep in mind that Lumberyard includes tools and plugins for Perforce, but you may use any source control system you want.

To set up Lumberyard in source control for the first time

1. Check in the pristine, unmodified Lumberyard to source control.
2. Update your source control to ignore the following folders and file types:

   dev/Solutions
   dev/BinTemp
   dev/Cache
   dev/Code/SDKs
   *.ilk
Create a new branch for development off of the pristine Lumberyard branch. When creating your game you will be making changes to the development branch only.

To upgrade Lumberyard with an existing version in source control

1. Before beginning the upgrade process, make sure to have the previous pristine version of Lumberyard checked in to source control.
2. In Windows Explorer, navigate to the folder where the previous pristine version of Lumberyard is located and delete the contents of the folder. 
   **NOTE:** If you don’t follow this step, old files that should be removed after upgrading won’t be removed from source control.
3. Install or copy the new version of Lumberyard to the empty folder. Make sure the general folder structure is the same as the previous version before continuing.
4. Using source control, reconcile the files in the folder with those in the pristine Lumberyard branch. For example, if you’re using Perforce, use the “Reconcile offline work” feature.
5. Commit the reconciled version to the pristine Lumberyard branch as the new version of Lumberyard.
6. Integrate the updated pristine Lumberyard branch into your development branch.

To upgrade Lumberyard *without* an existing version in source control
1. Commit the pristine version of Lumberyard that you based your game project on to source control.
2. Update your source control to ignore the following folders and file types:
   
   dev/Solutions  
   dev/BinTemp  
   dev/Cache  
   dev/Code/SDKs  
   *.ilk  
   *.suo  
   *.user  
   *.o  
   *.temp  
   *.bootstrap.digests  
   *.log  
   *.exp  
   *.vssettings  
   *.exportlog  
   *.mayaSwatches  
   *.ma.swatches  
   *.dds  
   *.bak  
   *.bak2  
   *.options  
   *.pyc  
   *.db

3. Create a new branch for development off of the pristine Lumberyard branch.
4. Check out the new development branch.
5. In Windows Explorer, navigate to the folder for the new development branch and delete the contents.
6. Copy all the files from your previous version of Lumberyard to this folder and make sure the general folder structure matches.
7. Use source control to reconcile the files in the new development branch folder with those in source control and accept your changes. For example, if you’re using Perforce, use the “Reconcile offline work” feature.
8. Your source control is now in a state to upgrade Lumberyard. Follow the instructions to upgrade Lumberyard with an existing version in source control.

You are now ready to run SetupAssistant from your development branch and configure your development environment.
**Step 2: Run SetupAssistant**

Before beginning development, you need to have the software and SDKs required for the kind of game or plugin you want to make. Lumberyard comes with a tool called SetupAssistant (formerly known as LumberyardLauncher in V1.1 and under) to make this process simple and painless.

**To run SetupAssistant**

1. Run **SetupAssistant** from the Start Menu or navigate to the folder where you installed Lumberyard and run **SetupAssistant.bat**

![SetupAssistant Interface](image)

**Note:** if the program fails to open or an error dialog appears, make sure you have the Visual C++ 2013 Redistributable for Visual Studio installed.
2. Verify the Lumberyard install path is correct and then select the first five options under *What do you want to do with Lumberyard*, if you intend to make a game for pc. If you want to build a game for mobile devices, select the additional checkboxes that correspond to the mobile devices you will be targeting.

- Run your game project
- Run the Lumberyard Editor and tools
- Compile the game code
- Compile the engine and asset pipeline
- Compile the Lumberyard Editor and tools
3. Press **Next** to proceed to the **Install software** page.

Install all required software by clicking the **Install it** links if you see a red x next to the software title. Once all required software is installed you should see green check marks next to each software title.
4. Press **Next** to proceed to the **Required SDKs** page. Install any missing required SDKs by following the instructions provided.
5. Press **Next** to continue to the **Install plugins** page. Install any plugins you intend to use to create assets for your game.
6. Press **Next** to proceed to the **Summary** page which shows buttons to configure a project or launch the editor.

![Summary Page](image)

You now have all the software and SDKs installed that you need and can run the editor and the Samples Project or create a new game.

**Step 3: Run Imbr_waf configure**

After making any changes to a project or changing installed 3rd party software you should configure your build environment. Lumberyard uses Imbr_waf to configure your environment, generate Visual Studio solutions and create builds.

**To run Imbr_waf**

1. Open a command prompt and navigate to the dev folder inside the Lumberyard root folder. For example, if you installed Lumberyard in C:\Amazon\Lumberyard\1.1.0.0, navigate to C:\Amazon\Lumberyard\1.1.0.0\dev
2. Type `lmbr_waf configure` at the command prompt and press ENTER.

The configure command instructs the Waf build system to do some essential tasks like running SetupAssistant in batch mode to ensure the correct 3rdParty libraries are accessible from dev/Code/SDKs. It also generates the LumberyardSDK.sln Visual Studio solution in the dev/Solutions folder.

3. After the command completes, check the output for any errors and address those before continuing. Errors in the configure command output are usually from missing 3rd party libraries. Run SetupAssistant again if you see any of these errors and install those missing libraries.

Congratulations! You have successfully installed the necessary software to develop and build a game with Lumberyard.
Related tasks and tutorials

Now that you have set up your development environment you’re ready to continue on to the following tutorials:

- Tutorial: Compiling the engine, editor and game
- Tutorial: Creating a new game project

We’d love to hear from you! Head to our Tutorial Discussion forum to share any feedback you have, including what you do or don’t like about our tutorials or new content you’d like to see in the near future.