Tutorial: Packaging your server build

This tutorial walks you through the steps to prepare a game server folder or “package” containing all the files necessary for your game server to run in Amazon GameLift. At the end of the tutorial you will have a server build package ready to upload to Amazon GameLift.

You will learn how to do the following:

- Enable the multiplayer sample
- Build the multiplayer sample dedicated server
- Compile the assets for the dedicated server
- Create a server build package
- Verify your server runs locally

Prerequisites

This tutorial uses the Multiplayer Sample, which is a sample game that has been configured to work with Amazon GameLift. The Multiplayer Sample has minimal assets to make it quick to package and upload to Amazon GameLift.

You must have completed the following tutorial before starting this tutorial:

- Setting up your development environment

Step 1: Enable the multiplayer sample

The first step is to enable the Multiplayer Sample as the default project.

To enable the Multiplayer Sample

1. In Windows Explorer, navigate to the folder where you installed Lumberyard and run SetupAssistant.bat
2. Select all the checkboxes shown in the following image:

3. Click on **Summary** to skip to the Summary Page
4. Click the **Configure project** button in the header to open the Project Configurator.

5. Select **MultiplayerSample** from the list of projects.
6. Press **Set as default** and then close the Project Configurator and Launcher.

When you activate the Multiplayer Sample, the bootstrap.cfg file is updated so the `sys_game_folder` points to the `MultiplayerSample` game folder. Now the game, server and editor applications will look in the `MultiplayerSample` game folder for assets.

**Step 2: Build the multiplayer sample dedicated server**

In this step you will compile the binaries for the Multiplayer Sample dedicated server. If you have already built the Multiplayer Sample dedicated server, you may skip this step.

**To build the MultiplayerSample dedicated server**

1. Open a command prompt and navigate to the dev folder inside the Lumberyard SDK root folder. For example, if you installed Lumberyard in `C:\LumberyardSDK`, navigate to `C:\LumberyardSDK\dev`

2. Type the following command in command prompt and press ENTER:

```
lmbr_waf configure build_win_x64_vs2013_release_dedicated -p game_and_engine --enabled-game-projects MultiplayerSample --progress
```
3. The configure command instructs the Waf build system to do some essential tasks like running the SetupAssistant (or LumberyardLauncher) in batch mode to ensure the correct 3rdParty libraries are accessible from dev/Code/SDKs.

   a. The “build_win_x64_vs2013_release_dedicated –p game_and_engine” command starts a build of the Windows x64 release_dedicated game and engine specification under Visual Studio 2013.

4. 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 (or LumberyardLauncher) again if you see any of these errors and install those missing libraries.

**Note:** If you build the dedicated server in any mode other than release_dedicated then please make sure to disable the Asset Processor from launching by modifying the file located in dev\MultiplayerSample_pc_Paks_Dedicated\bootstrap.cfg and setting connect_to_remote=0. The reason we disable the Asset Processor from launching is because we will be compiling the assets for the dedicated server in the next steps and will not require the Asset Processor to process them.

At this point you should have the MultiplayerSampleLauncher_server.exe in your dev/Bin64vc120.Release.Dedicated folder, if you used “vs2013” in the build command.

**Step 3: Compile the assets for the dedicated server**

In this step you will compile the assets for the Multiplayer Sample dedicated server using a .bat script that invokes the Asset Processor to convert source assets into compiled assets, and then invokes the Resource Compiler to package the compiled assets into .pak files.
To compile the assets for the Multiplayer Sample dedicated server

1. Open a command prompt and navigate to the dev folder inside the Lumberyard SDK root folder. For example, if you installed Lumberyard in C:\LumberyardSDK, navigate to C:\LumberyardSDK\dev

2. Type ‘BuildMultiplayerSample_Paks_PC_dedicated.bat’ at the command prompt and press ENTER.

The .bat file will invoke the Asset Processor, which converts the source assets in dev/MultiplayerSample into compiled assets. It then invokes the Resource Compiler with the dev/Bin64/rc/RCJob_Build_MultiplayerSample_paks_dedicated.xml resource build file. This .xml file has entries in it that instruct the Resource Compiler to put the compiled assets into .pak files and copy them and various other files into dev/MultiplayerSample_pc_Paks_Dedicated.

At this point you should have a dev/MultiplayerSample_pc_Paks_Dedicated folder with .pak files and configurations inside it.

Step 4: Create a server build package

Your build package must include all of the components required to run your game server. For this tutorial, we’ll use the dev/MultiplayerSample_pc_Paks_Dedicated folder as our build folder. In addition to the assets that are already there, the folder must contain the following:

- **install.bat** – The installer batch file, which must be in the folder root.
  - When a game session is started on a new server, the install.bat will be automatically run and will perform any necessary game server setup, including installing dependencies, configuring registry settings, etc. The commands invoked in install.bat must not require any user input.
- **Game server binaries** – The binaries necessary to run your game server.
- Any dependencies required by your game server executable to run.
To create the server build package for the Multiplayer Sample dedicated server

1. Copy the dev/Bin64vc120.Release.Dedicated folder into
dev/MultiplayerSample_pc_Paks_Dedicated

   NOTE: if you previously compiled a debug or profile version of the dedicated server you should remove the .pdb symbol files from the copy of dev/Bin64vc120.Release.Dedicated because they’re large and will make the upload and deployment process slower.

2. Copy the vcredist_x64.exe file from dev/Tools/Redistributables/Visual Studio 2013 or Visual Studio 2015 into dev/MultiplayerSample_pc_Paks_Dedicated.

3. Create a text file named install.bat in dev/MultiplayerSample_pc_Paks_Dedicated with the following text in it:
   
   ```vcredist_x64.exe /q```

4. **Step 5: Verify your server runs locally**

   Before uploading your build to Amazon GameLift you should test it locally to make sure you aren’t missing any required files and that the server performs as expected.

To verify the Multiplayer Sample dedicated server runs locally

1. In Windows Explorer, navigate to the
dev/MultiplayerSample_pc_Paks_Dedicated/Bin64vc120.Release.Dedicated folder.

2. Run **MultiplayerSampleLauncher_Server.exe** and the server CLI will appear.
3. Load a map by typing ‘map multiplayergame’ in the server CLI console and then press ENTER.

![Server CLI console output]

The server console messages should indicate that the map loaded with `mission: map:multiplayergame`

4. Close the server CLI by typing ‘quit’ in the server CLI console and then press ENTER.

5. Remove the servers log files by deleting the dev/ MultiplayerSample_pc_Paks_Dedicated/user folder because you want to avoid a situation where you could confuse the real server logs with your local test server logs.

Congratulations! You have successfully created a server build package, ready to upload to Amazon GameLift.

**Related tasks and tutorials**

Now that you have created your server build package, continue on to the next tutorial to learn how to upload your server build to Amazon GameLift.

- GameLift Getting Started – Uploading your server build
- GameLift Getting Started – Creating a fleet
- GameLift Getting Started – Creating and connecting to a game session

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.