Outline:

Level Loading basics

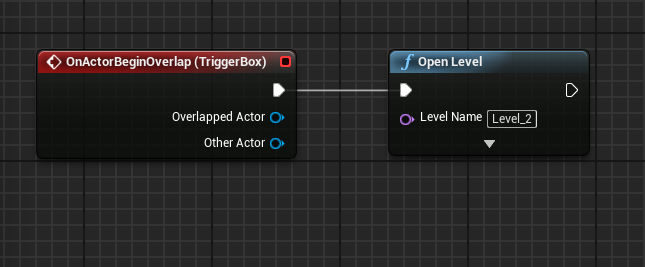
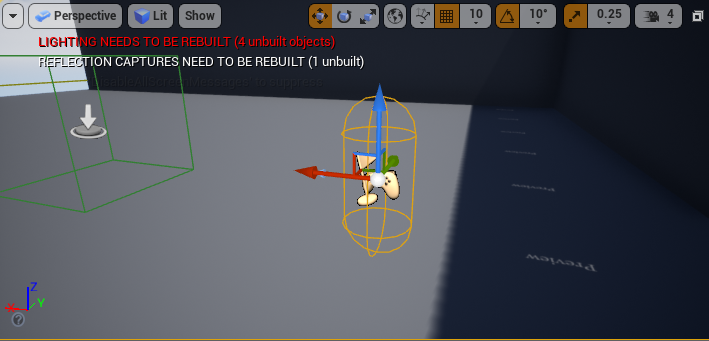
Level loading Open level

Level Streaming Volumes

Level Streaming blueprints

Level loading is a key feature in any game and if done properly can enhance the experience of the player or if done poorly can become a huge burden on the players enjoyment. There are three basic methods of loading and unloading levels in unreal. These methods are Open Level, Level Streaming volumes, and Level streaming blueprints.

Starting off with the first method you can use to load levels is the simplistic Open Level function that unreal provides for you. All the function does is unloads the previous persistent level and loads in a new persistent level and places you where the player start is located in the level loaded.

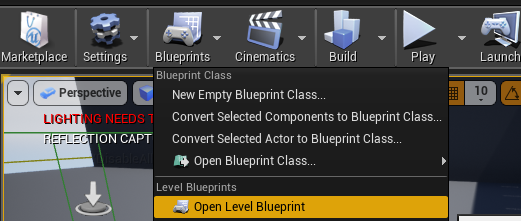
 

One thing to note about the open level function or any function that uses the level name is that level names are case sensitive. The down sides to Open level are it loads the entire level you chose to load in so it can be a very expensive function to use. Lastly Open Level doesn’t just load in a new level it creates a new instance of the level. What this means is if you don’t have some form of saving system or use the game instance to store data so you can access it through all the levels then data will be lost across levels and anything previously destroyed in the level will reappear as if it was never destroyed.

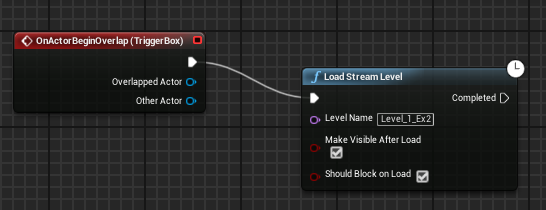
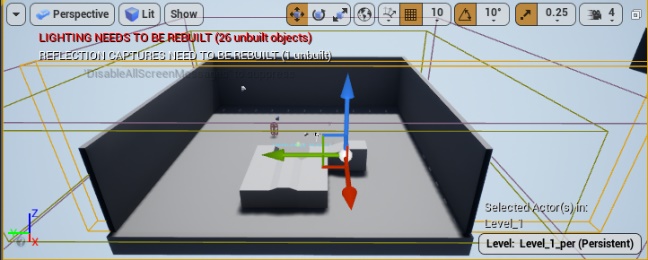
The Second method of loading and unloading levels is level streaming volumes. Level streaming volumes are very simple to use and is less expensive than open levels, but still has its own issues. From the unreal documentation it says, “Each frame, the engine iterates over each Level and checks to see if the player's viewpoint is inside any of the Level Streaming Volumes associated with that Level.” (“Level Streaming Volumes,”). This shows that each frame or in other words every tick the level checks every single volume to see what should and shouldn’t be loaded. The reason this could be a problem is if you had a larger level with tons of level streaming volumes iterating over all of them could get expensive. The upsides to using volumes are it makes multiplayer games easier to load and its less expensive than using open level to load all of the level at once.

The final method which in theory would be the most efficient method of level loading is level streaming through blueprints. To have a level streaming system first you need to make the persistent level. This level will be the base of the entire level. The main features of the persistent level are the player, the player start and any lighting that can be seen through the whole level. The persistent level is always loaded and can’t be removed from the stream. The reason the player is included in the persistent level is because if the level the player started in gets unloaded then the player gets unloaded.

After you have your persistent level made you can move on to actually implementing level streaming. All of the level streaming functions should be set in the level blueprint.



Basically, all that’s left to do is to go into the level blueprint put an event based on overlapping with a trigger box and when this happens load the level you wish to load.

**Load Stream function: in Level Blueprint** **Trigger Box: Used for level streaming**

Lastly some theory about level loading. Most ideas for leveling loading are to have hidden loading screens so the player doesn’t see the level loading. In my example I have the loading zones well before the player can see the level beginning to load in. Other ideas that are used are loading through a transition level or a long hallway and using doorways and corners to trick the player into thinking this was all loaded when it wasn’t.

In conclusion Level loading has many takes and approaches that all have some merit. Open Level is useful for getting the player into the next level but is expensive because by default it loads the whole level. Level volumes is extremely easy to implement, but constantly checks if the player is in the volume being loaded. Level streaming is a very effective solution , because you can control exactly when things should be loaded and unloaded, but takes longer to implement and it’s possible to unload part of the level and got back to where the level no longer is so, its easy to have more bugs and headache with this method. From my finding I can conclude that the best strategy is to use a mix of the methods and not hold strong to any particular method. If you have level streaming or level streaming volumes implemented and load that level with open level, then you lose the downside of open level just loading the whole level in at once. Also, if the level isn’t too large than having well placed level streaming volumes is infinitely better than using level streaming.

Another noteworthy piece that the engine gives you to edit how levels load is the levels window.

References:

“Level Streaming Overview,” *Unreal Engine Documentation*. [Online]. Available: https://docs.unrealengine.com/en-US/Engine/LevelStreaming/Overview/index.html.

“Load and Unload Levels with Blueprints,” *Unreal Engine Documentation*. [Online]. Available:

https://docs.unrealengine.com/en-US/Engine/LevelStreaming/HowTo/StreamWithBlueprints/index.html.

“Level Streaming Volumes,” *Unreal Engine Documentation*. [Online]. Available: https://docs.unrealengine.com/en-US/Engine/LevelStreaming/StreamingVolumes/index.html.

“Stream Sublevels with Level Streaming Volumes,” *Unreal Engine Documentation*. [Online]. Available: https://docs.unrealengine.com/en-US/Engine/LevelStreaming/HowTo/StreamWithVolumes/index.html.