Steam Integration Into Your Unreal Project

Creating, Finding, and Joining Sessions

By Marc Cohen

Project

- Look/follow along with the project from GitHub

https://github.com/Ironthighs/NGDASteamIntegration

Prerequisites

- Have Steam

  http://store.steampowered.com/about/

- Make a new Unreal C++ Project

Configuring the Project

- Open the <ProjectName>.Build.cs (In this example, the project name is NGDA)
  - Add the following to the constructor:

```
PrivateDependencyModuleNames.Add("OnlineSubsystem");
```

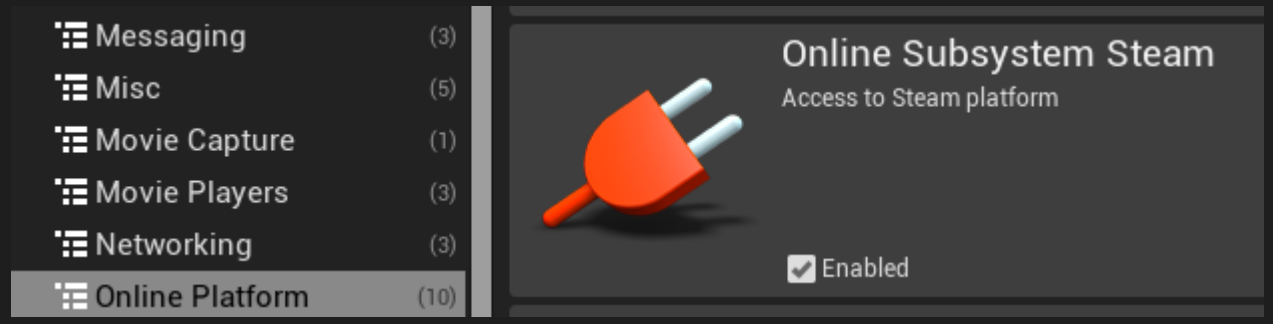
Configuring the Project

- Open Config/DefaultEngine.ini
  - Add the following to the file:

```
[/Script/Engine.GameEngine]
!NetDriverDefinitions=ClearArray
+NetDriverDefinitions=(DefName="GameNetDriver",DriverClassName="/Script/OnlineSubsystemSteam.SteamNetDriver",DriverClassNameFallback="/Script/OnlineSubsystemUtils.IpNetDriver")

[OnlineSubsystem]
bEnabled=true
DefaultPlatformService=Steam
PollingIntervalInMs=20

[OnlineSubsystemSteam]
bEnabled=true
SteamDevAppId=480
GameServerQueryPort=27015
bRelaunchInSteam=false
GameVersion=1.0.0.0
bVACEnabled=1
bAllowP2PPacketRelay=true
P2PConnectionTimeout=90

[/Script/OnlineSubsystemSteam.SteamNetDriver]
NetConnectionClassName="/Script/OnlineSubsystemSteam.SteamNetConnection"
```

Configuring the Project

- Run the editor and go to Editor>Plugins>Online Platform and enable Online Subsystem Steam. Restart the editor.



- https://answers.unrealengine.com/questions/484873/413-steam-setup-not-working.html

Code - Overview

- The goal is to hook up session creation, finding, joining, and destroying through the Online Subsystem's Online Session interface.
- Methods to be called:

```
bool IOnlineSession::CreateSession(
        const FUniqueNetId & HostingPlayerId,
        FName SessionName,
        const FOnlineSessionSettings & NewSessionSettings)

bool IOnlineSession::FindSessions(
        const FUniqueNetId & SearchingPlayerId,
        const TSharedRef < FOnlineSessionSearch > & SearchSettings)
```

Code - Overview

- The goal is to hook up session creation, finding, joining, and destroying through the Online Subsystem's Online Session interface.
- Methods to be called:

```
bool IOnlineSession::JoinSession(
        const FUniqueNetId & LocalUserId,
        FName SessionName,
        const FOnlineSessionSearchResult & DesiredSession)

bool IOnlineSession::DestroySession(
        FName SessionName,
        const FOnDestroySessionCompleteDelegate & CompletionDelegate)
```

Code - Overview

- The goal is to hook up session creation, finding, joining, and destroying through the Online Subsystem's Online Session interface.
- Methods to be called:

```
bool IOnlineSession::StartSession(
        FName SessionName)
```

Code - Overview

- Must bind to callbacks for when the asynchronous call finishes. These delegates are (there are more, but these are the bare minimum):

```
DECLARE_MULTICAST_DELEGATE_TwoParams(FOnCreateSessionComplete, FName, bool);
typedef FOnCreateSessionComplete::FDelegate FOnCreateSessionCompleteDelegate;

DECLARE_MULTICAST_DELEGATE_TwoParams(FOnStartSessionComplete, FName, bool);
typedef FOnStartSessionComplete::FDelegate FOnStartSessionCompleteDelegate;

DECLARE_MULTICAST_DELEGATE_OneParam(FOnFindSessionsComplete, bool);
typedef FOnFindSessionsComplete::FDelegate FOnFindSessionsCompleteDelegate;
```

Code - Overview

- Must bind to callbacks for when the asynchronous call finishes. These delegates are (there are more, but these are the bare minimum):

```
DECLARE_MULTICAST_DELEGATE_TwoParams(FOnJoinSessionComplete, FName,
EOnJoinSessionCompleteResult::Type);

typedef FOnJoinSessionComplete::FDelegate FOnJoinSessionCompleteDelegate;


DECLARE_MULTICAST_DELEGATE_TwoParams(FOnDestroySessionComplete, FName, bool);

typedef FOnDestroySessionComplete::FDelegate FOnDestroySessionCompleteDelegate;
```

Code - GameSession

- Create a new GameSession class for all your steam code (per Shooter Game example)
- To make your new game session class the default class, must override your GameMode:: GetGameSessionClass() method to return:

YourGameSessionClass::StaticClass()

```
virtual TSubclassOf<AGameSession> GetGameSessionClass() const override;


TSubclassOf<class AGameSession> ANGDAGameMode::GetGameSessionClass() const
{
    return ANGDAGameSession::StaticClass();
}
```

Code - Session Settings

- Creating visible matches requires advertising...

```
sessionSettings->bIsLANMatch = bIsLAN;
sessionSettings->bUsesPresence = bIsPresence;
sessionSettings->NumPublicConnections = MaxNumPlayers;
sessionSettings->Set(SETTING_MAPNAME, MapName, EOnlineDataAdvertisementType::ViaOnlineService);

// Again, you MUST have bShouldAdvertise = true if you want others to see your game session.
sessionSettings->bShouldAdvertise = true;
sessionSettings->bAllowJoinViaPresence = true;
sessionSettings->bAllowJoinInProgress = true;
```

Code - Travel URL and Travelling

- Once a session is created, usually you want to perform a travel to another game scene.
- Use a class member variable to build the travel url just before creating the session then ServerTravel/ClientTravel to the url after Start Session.
- Use a class member variable for specifying whether the user is creating a game or not. StartSession is used for both server and client. If creating a session, use ServerTravel, otherwise use ClientTravel.

Useful Links

- UE4 API
  - Link to searchable API
    - https://docs.unrealengine.com/latest/INT/Search/index.html
  - IOnlineSession
    - https://docs.unrealengine.com/latest/INT/API/Runtime/OnlineSubsystem/Interfaces/IOnlineSession/index.html
- UE4 AnswerHub
  - Answer for Steam setups not working from 4.12.5 to 4.13
    - https://answers.unrealengine.com/questions/484873/413-steam-setup-not-working.html