# Ring Game Poker Blueprint Documentation

## Game Maps

Due to the replicated nature of the template, you'll need 2 separate maps for this project. First map will serve as the lobby, the second serving as the game session map.

## Lobby Map Setup

Inside the lobby map, you need to place an actor of class "BP_RingGamePokerSessionCreationDesk_Actor". It has a public variable called DataAsset which you need to populate with your version of ring game data asset which will be explained later on. Upon interaction with the lobby actor, an instance of "WBP_SessionFinder_Blueprint" widget will be displayed to player, in which they'll be able to either create a session or find and join a session that has room for more players to join. After creating or joining a session players will be moved to the map created for sessions.

If you intend to edit the session creator actor, simply open up the actor and you'll be able to change the mesh, it's materials or anything you want.

## Game Map Setup

Inside game map you'll need to place an actor of type "BP_RingGamePokerGameTable_Actor". This is the actor that handles all poker and replication logic. There are some components in this actor and each serve a specific role.

### CameraTargetTransform

This component is a basic cube which will be used to determine the camera destination for the game view. If you change its location and rotation, the camera will have its transform to display the game table when players are playing the game.

### Camera3dViewTargetTransform

This component is a basic cube which will be used to determine the camera destination for the 3D game view in case player decides to go to 3D view. If you change its location and rotation, the camera will have its transform to display the game table when players are playing the game.

### CardSpawnTargetTransform

This cube is used for determining for defining the location from which cards will be spawned from and later moved to deck location

**CardDeckTargetTransform**

The cube for defining the location of deck of cards. All cards will be moved to this location after the are spawned.

**BurnedCardTargetTransform**

Durin the gameplay, after each round a card is burned from deck. This is the location burned cards will be moved to.

**CommunityCardsTargetTransform**

This is for defining the location of community cards placed on table during different rounds of the game.

**PotAmountText**

A text component used to display accumulated pot amount.

There are five components per each player that go like this

**PlayerHandTargetTransform**

Defines the location of each players 2 cards.

**PlayerChipsText**

Text component used to display amount of chips each player has placed on table as bet.

**DealerIconPlane**

A plane to be displayed only if the player is the dealer in that game.

**PlayerEmojiPlane**

A plane to display player's emoji reaction

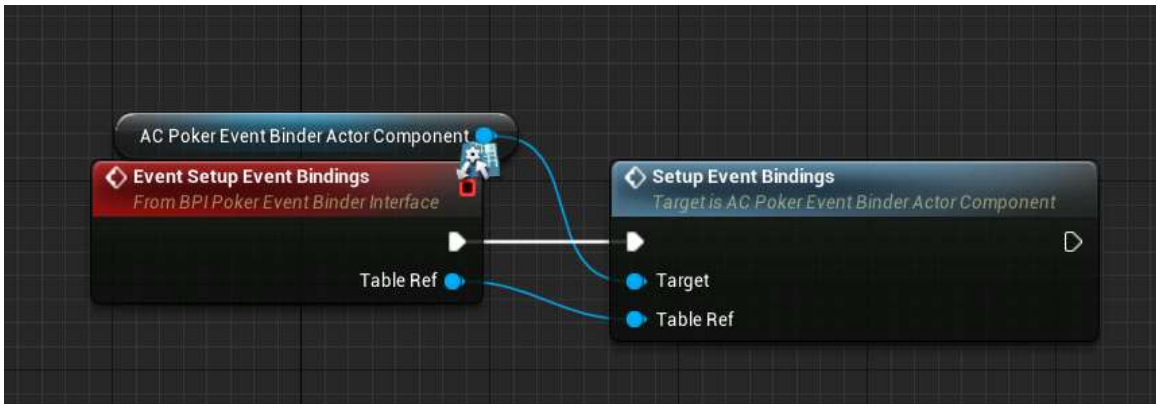
**PlayerInfoWidget**

This is a widget that will hold each player's balance and username.

You can change the transform and style of each of these components as you desire.

## Player Character Setup

If you have a custom player character or want to use any player character other than the one in the project, you need to go through a few simple steps. First add a an instance of "AC_PokerEventBinder_ActorComponent" component to your character. Then in character's class settings add "BPI_PokerEventBinder_Interface" implementation to list of implemented interfaces. Lastly, you'll need to have this small function call in your event graph


AC Poker Event Binder Actor Component
Event Setup Event Bindings
From BPI Poker Event Binder Interface
Table Ref
Setup Event Bindings
Target is AC Poker Event Binder Actor Component
Target
Table Ref

## Data Asset Setup

In order to have a customized version of the ring game, you need to setup an instance of "DA_PokerVersion_DataAsset". Inside it you can set some values related to the game which are pretty self-explanatory and will have an extra explanation if you hover over them. Some of them that will need some more explanation are as follow.

Lobby Level Name: Name of the map you created for the lobby management.

Game Level Name: Name of the map you created for game session management.

Card Textures: For each suit of cards, you need to assign 13 textures starting from Ace to King.

## Poker Variation Preset

You can setup new presets by adding new rows to the "DT_PokerVariationPresets" which receives 3 variables. First variable defines how many cards each player gets when being dealt cards, the second and third variables define how many of the players cards should be inside the combo that is formed at the end of the game, aka showdown, and if that value has to be absolute or not. So if it's set to 2 and absolute, exactly 2 of players cards have to be included in the combo. But if it's set to 2 and not absolute any number >= 2 can be in the combo from player's cards. Finally in the data asset, you need to specify which preset you're going to use by providing the name of its row.

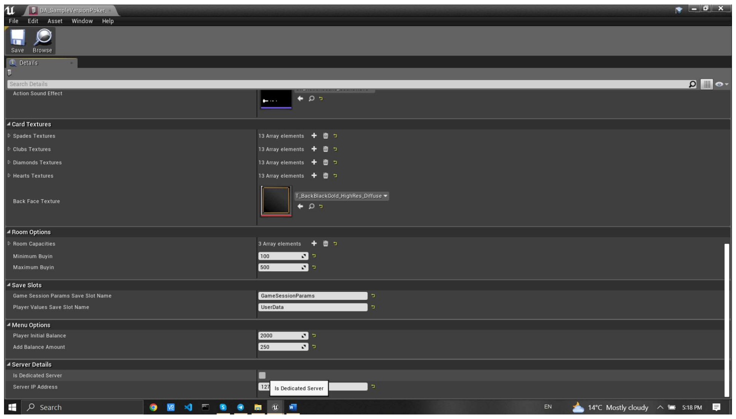
## Server Setup

The poker creator blueprint supports both dedicated server and listen server approaches.

You need to setup some configurations in order to achieve each of these two approaches.

## Listen servers

Listen sever setup is pretty straightforward. All you need to do is make sure you set the number of players in play mode to something more than 1, and in the data asset you created in the previous step, set “is dedicated server” to false. You can also leave server IP address field to empty as it’s only used for dedicated server. You can test this setup playing with PIE option. You also need to set net mode to Play As Listen Server.

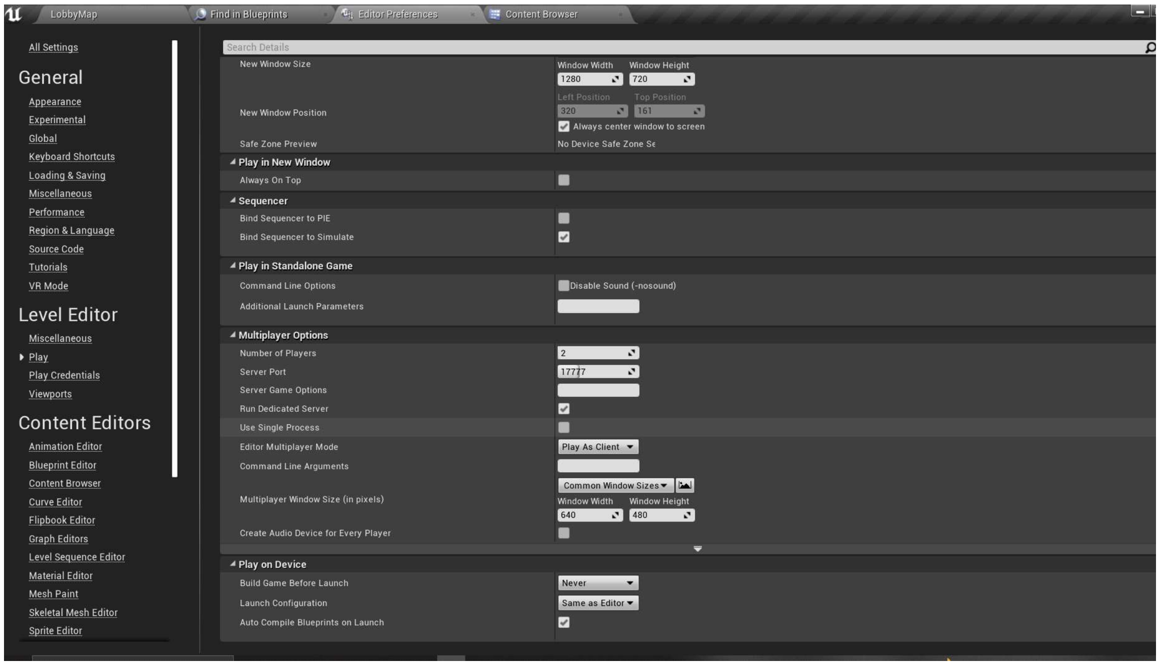


| | |
|---|---|
| Action Sound Effect | |
| **Card Textures** | |
| Spades Textures | 13 Array elements |
| Clubs Textures | 13 Array elements |
| Diamonds Textures | 13 Array elements |
| Hearts Textures | 13 Array elements |
| Back Face Texture | T_BackBlackGold_HighRes_Diffuse |
| **Room Options** | |
| Room Capacities | 3 Array elements |
| Minimum Buyin | 100 |
| Maximum Buyin | 500 |
| **Save Slots** | |
| Game Session Params Save Slot Name | GameSessionParams |
| Player Values Save Slot Name | UserData |
| **Menu Options** | |
| Player Initial Balance | 2000 |
| Add Balance Amount | 250 |
| **Server Details** | |
| Is Dedicated Server | ☐ |
| Server IP Address | 127 |

## Dedicated server

In order to setup dedicated server setup you need to go through a few steps.

If you want to have the entire game in 1 map, you can test in PIE. But if you want to have a lobby map and a game map like you would do with listen server approach, you can only do that in standalone. In order to have multiple players in standalone, you need to open editor preferences, go to **Level Editor > Play**, scroll down till you find an option called “Use Single Process”. By default that should be checked. Set that to unchecked and some new options will be available to you right there. Set Editor Multiplayer Mode to Play As Client, and check the run dedicated server option and you should be good.



| | |
|---|---|
| New Window Size | Window Width 1280, Window Height 720 |
| New Window Position | Left Position 320, Top Position 161; ✓ Always center window to screen |
| Safe Zone Preview | No Device Safe Zone Se |
| **Play in New Window** | |
| Always On Top | ☐ |
| **Sequencer** | |
| Bind Sequencer to PIE | ☐ |
| Bind Sequencer to Simulate | ✓ |
| **Play in Standalone Game** | |
| Command Line Options | ☐ Disable Sound (-nosound) |
| Additional Launch Parameters | |
| **Multiplayer Options** | |
| Number of Players | 2 |
| Server Port | 17777 |
| Server Game Options | |
| Run Dedicated Server | ✓ |
| Use Single Process | ☐ |
| Editor Multiplayer Mode | Play As Client |
| Command Line Arguments | |
| Multiplayer Window Size (in pixels) | Common Window Sizes; Window Width 640, Window Height 480 |
| Create Audio Device for Every Player | ☐ |
| **Play on Device** | |
| Build Game Before Launch | Never |
| Launch Configuration | Same as Editor |
| Auto Compile Blueprints on Launch | ✓ |

In the data asset you have created scroll all the way down and check the is dedicated server check box. Also set the server IP address to your server's remote IP, or if you're working on localhost set it to 127.0.0.1 . Then you need to set play mode to standalone. All this is because opening levels with single processor option turned off is not supported in PIE mode prior to UE5.2. You should be able to test the game with dedicated server in editor now.

## Dedicated server build

In order to launch your game with dedicated server, you need to build a server build. For that you need to have the source version of engine installed from Unreal github and go through a few steps. This video provides a good explanation of what you need to do.
https://www.youtube.com/watch?v=zNUxzl8Dcb4

## Server Manager setup

Unreal engine dedicated server approach by default supports a single server/session. So multiple sessions cannot be played at the same time if you launch your server build to a server. In order to achieve that you either need to use services like Photon, or write your own custom server manager script. Basically what that server manager does is running a new instance of your server build each time a new user requests a new session to be initiated. This video provides a good overall description of how a basic server manager should look like.
https://www.youtube.com/watch?v=QHPP48Y6Gu0

## Single Map In Dedicated Server Approach

In listen server approach, a level transition has to happen for both creating and joining a session. The same is not true for dedicated servers though, because it doesn’t use the same concept of sessions as used in listen server. What you need to do to achieve single map for the game, is having the clients and the server in a map. If you already have clients in a map along with the server, you can use the node “Execute Console Command” with the command being “ServerTravel <MapName>”. What this does is moving server to <MapName> and moving clients along too. If you do not have clients in a single map already, you can do “Travel <MapName>”. This will move server to <MapName> but won’t move clients. Later for each player you can use node Open Level so they can move to the map the server is in. You also need a way of setting player buyin and username before they interact with the game table. The buyin and username are stored in local storage of type “BP_GameSessionSaveGame_SaveGameObject”. A simple implementation can be found in lobby widget.