🎓 LESSON 09 Late Joiners

Start

LESSON GOALS

Video Link

🎯 Goal Summary:

- Prevent late joiners from interfering with GAME state
- Teleport late joiners to a spectator area
- Track which players are “in-game”
- Re-integrate spectators after:

  GAME_OVER → RELOAD → ORGANIZATION

- Maintain clean multiplayer flow

Next

Learning Objectives + Deliverables

Learning Objectives

Create a spectator zone

Add a boolean _isInGame status per player

Detect late joins during GAME state

Teleport spectator players on join

Restore full participation after round end

Deliverables

✓ Dedicated spectator spawn zone

✓ State-aware player join handling

✓ Local-only _isInGame logic

✓ Automatic reintegration into next round

✓ Clean user flow that protects GAME state

GENERAL DESCRIPTION OF THE LESSON

High-Level Description (Challenge Level: Max)

🎯 **Goal:** We are going to place any new player that joins the world during an on-going match in an area outside the field so they can see the game and join when it finishes.

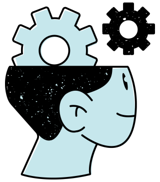
🔴 LEVEL 1: Advanced Challenge

In this lesson we will:

- ◆ If a player joins during ORGANIZATION → normal assignment
- ◈ If a player joins during GAME → they go to spectator mode
- ◈ Spectators can watch but cannot participate
- ◈ When the match resets → everyone becomes an eligible player again
- ◈ Create a spectator spawn point
- ◈ Modify OnPlayerJoined() to check current state
- ◈ Add teleport logic

🧠AI Lesson Prompt: Late Joiners

Next



Exercise 1: The code will handle the late joiners to make them spawn outside the field until the next game is organized.

- Actions in **GameController** script:

1) Create a member variable to connect with the GameObject where the later players joining will respawn and initialize it:

2) Create a member variable (**bool _isInGame**) to differentiate between players in the match and players outside.

3) Set the previous variable to true in the right position when the current players of the game start the match.

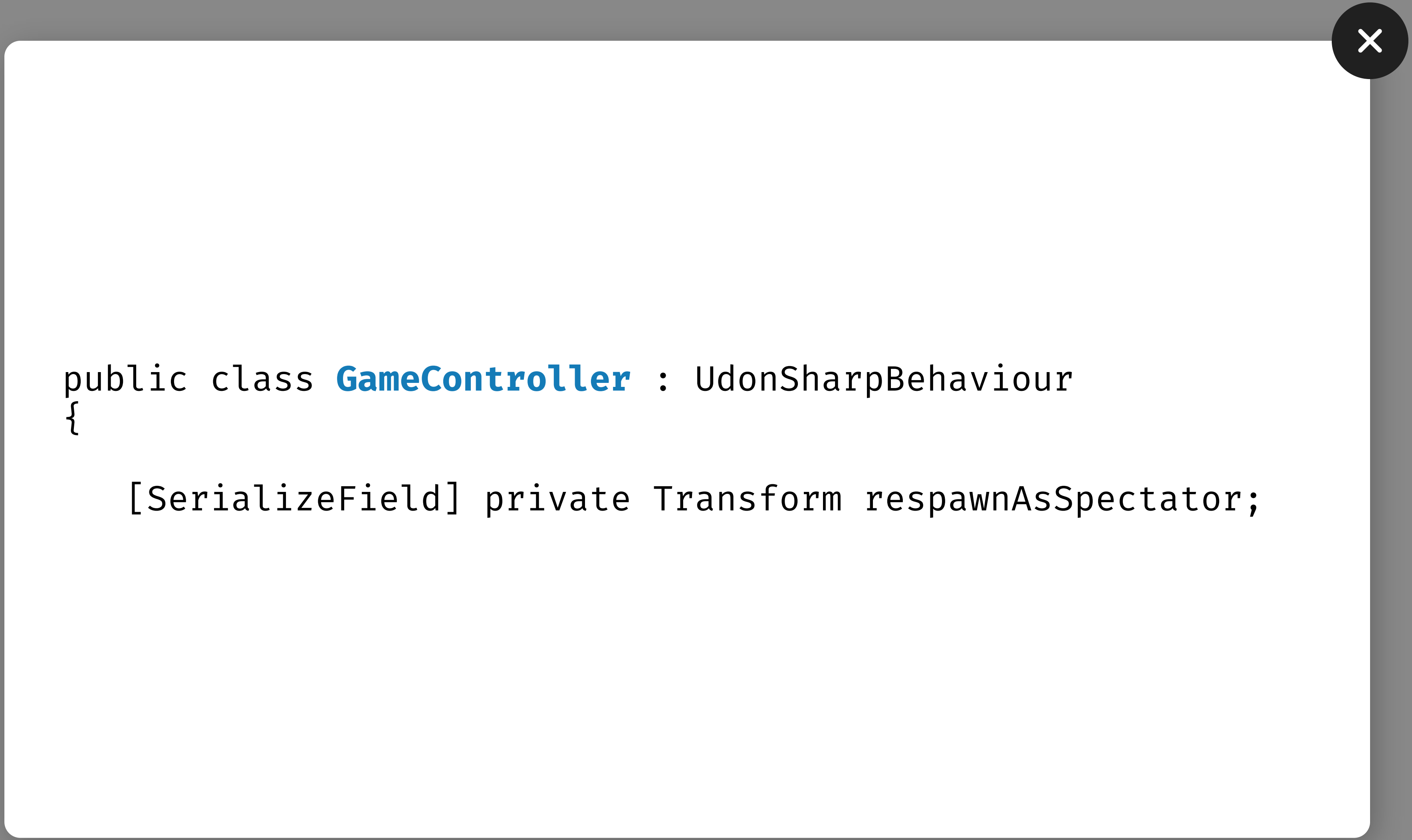
4) Set (**_isInGame=false**) in the right place.

5) When changing to the GAME state the players who aren't in a game should be teleported to the position outside the field.

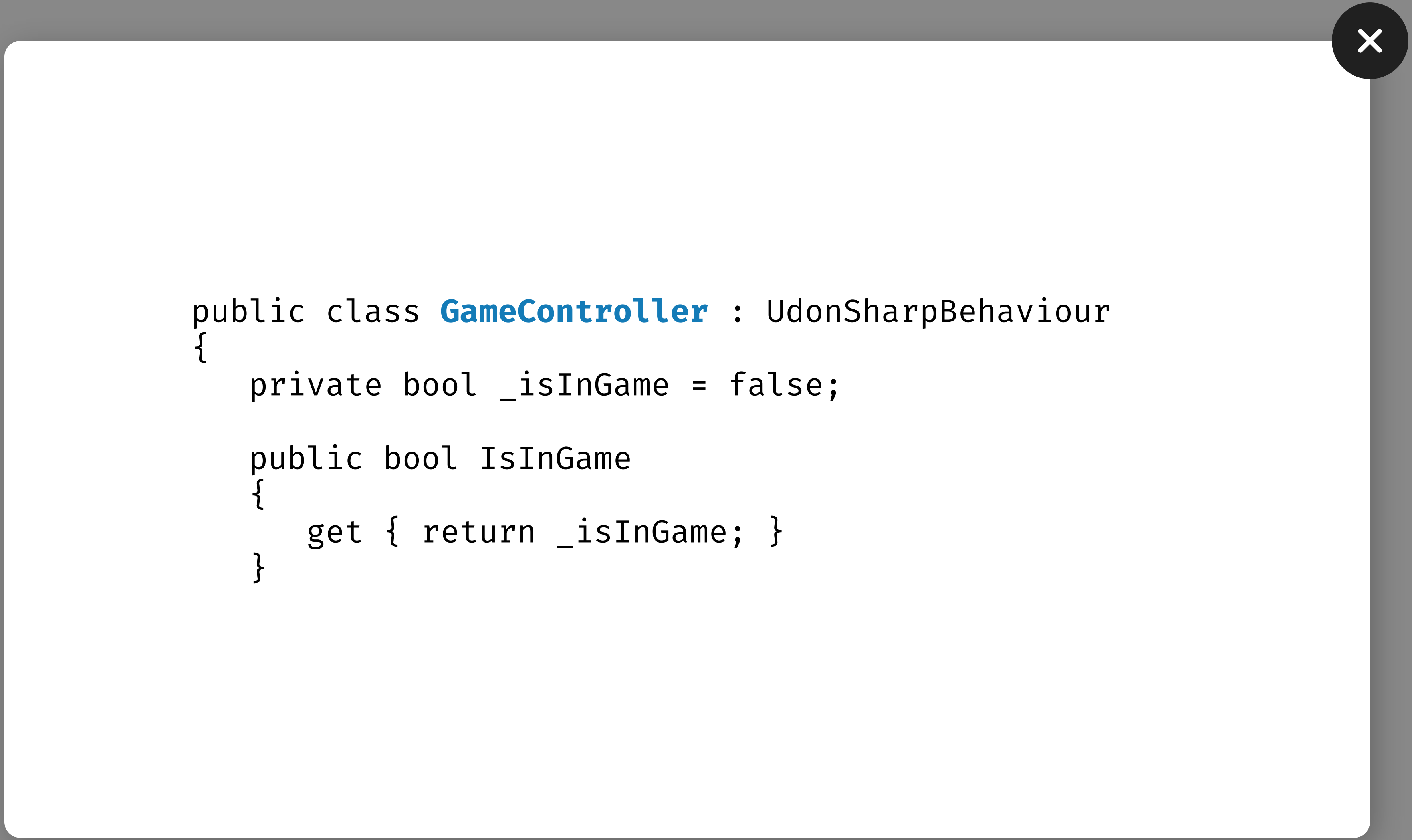
6) You will need to test it with multiple VRChat instances (2 VRChat instances playing + 1 VRChat instance joining with a match in progress)

7) Verify that when the game is over and we enter again to the ORGANIZATION state the players who weren't in the match are now considered.

Code Checkpoint: Late Joiners ✓

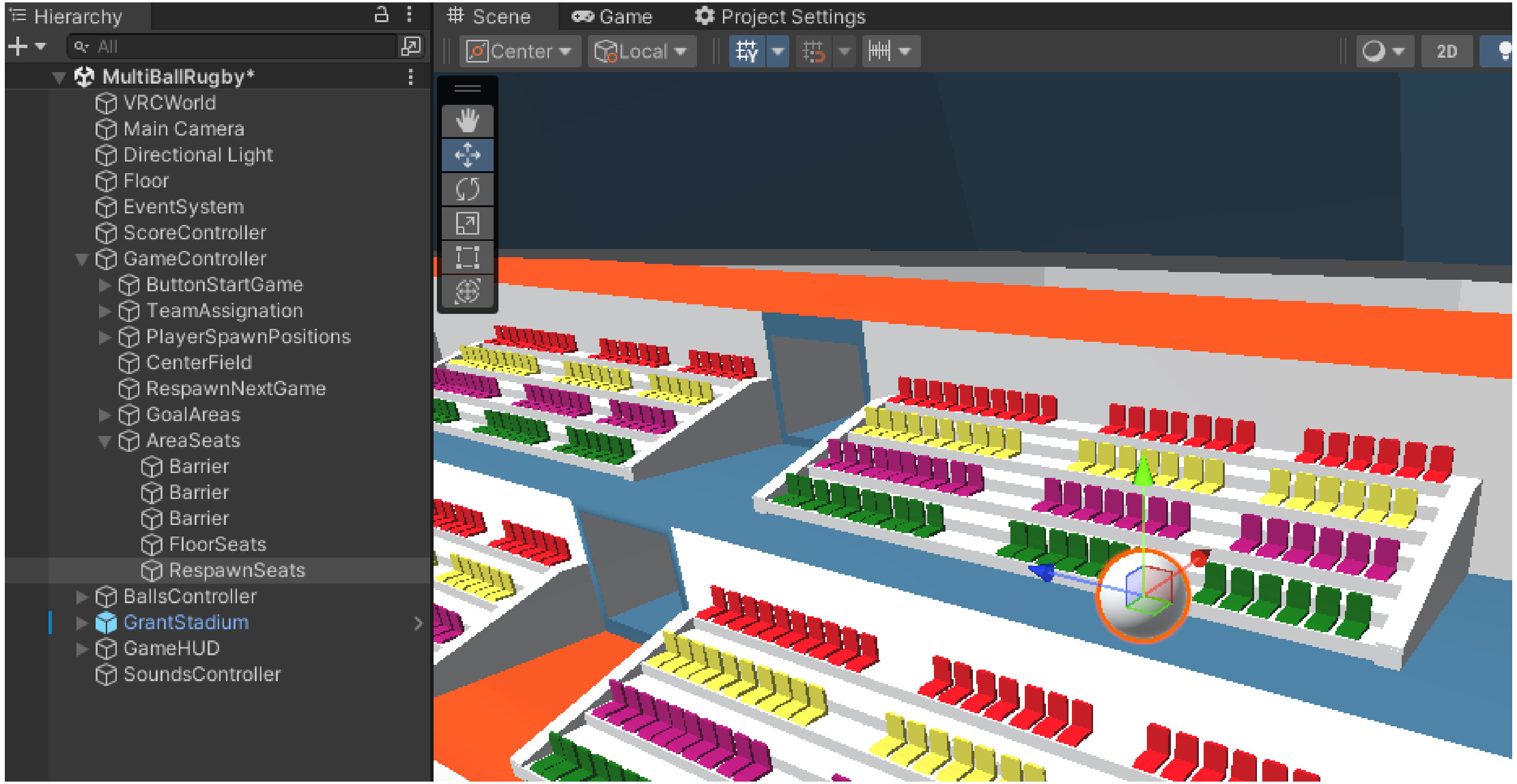


```
public class GameController : UdonSharpBehaviour
{

    [SerializeField] private Transform respawnAsSpectator;
```

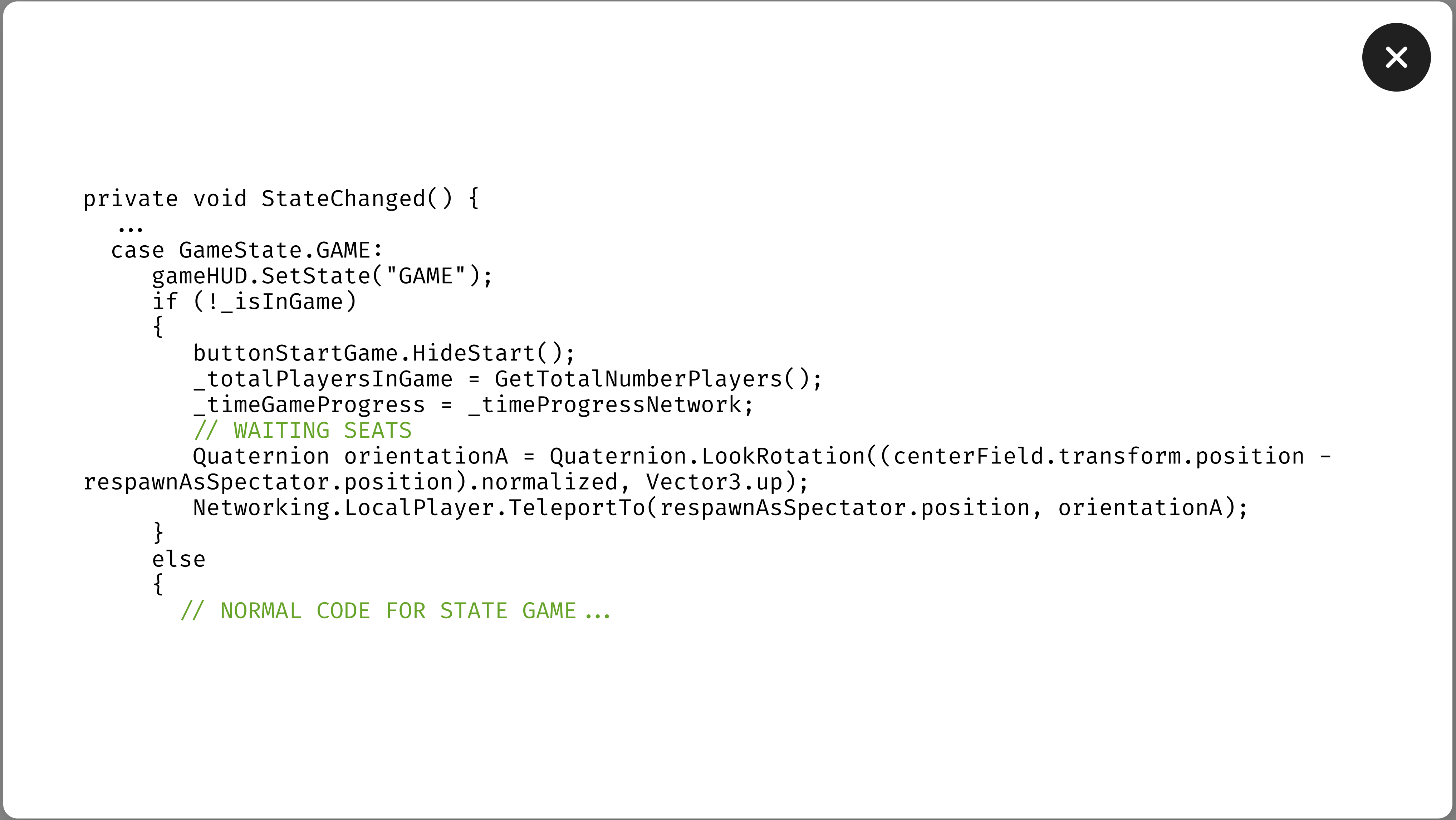


```
public class GameController : UdonSharpBehaviour
{
   private bool _isInGame = false;

   public bool IsInGame
   {
      get { return _isInGame; }
   }
```

```
public void RequestMasterStartGame()
{
     _isInGame = true;
     if (Networking.IsMaster)
     {
        SetState(GameState.GAME);
     }
}
```



```
MultiBallRugby*
  VRCWorld
  Main Camera
  Directional Light
  Floor
  EventSystem
  ScoreController
  GameController
    ButtonStartGame
    TeamAssignation
    PlayerSpawnPositions
    CenterField
    RespawnNextGame
    GoalAreas
    AreaSeats
      Barrier
      Barrier
      Barrier
      FloorSeats
      RespawnSeats
  BallsController
  GrantStadium
  GameHUD
  SoundsController
```



```
private void StateChanged() {
  ...
  case GameState.GAME:
     gameHUD.SetState("GAME");
     if (!_isInGame)
     {
        buttonStartGame.HideStart();
        _totalPlayersInGame = GetTotalNumberPlayers();
        _timeGameProgress = _timeProgressNetwork;
        // WAITING SEATS
        Quaternion orientationA = Quaternion.LookRotation((centerField.transform.position -
respawnAsSpectator.position).normalized, Vector3.up);
        Networking.LocalPlayer.TeleportTo(respawnAsSpectator.position, orientationA);
     }
     else
     {
       // NORMAL CODE FOR STATE GAME ...
```

```
private void StateChanged()
{  …
      case GameState.GAME_OVER:
        …
              gameHUD.ShowGameOver(gameResult,
                                   scoreController.GetScoreTeamRed(),
                                   scoreController.GetScoreTeamBlue());
              _isInGame = false;
              break;
```

LESSON 09 COMPLETED

You now have:

- Full support for late joining
- Safe spectator handling
- Clean reintegration to matches
- A professional multiplayer flow

☆ EXTRA FEATURES

If you want to evolve this project you could consider the next features:

- **Difficulty Easy:** Include sound effects
- **Difficulty Easy:** Display a congratulations message for the player who scores a goal
- **Difficulty Hard:** Implement a system so you can pass the ball to another player of your team.
- **Difficulty Hard:** Organize an alternative game organization system where the players have to confirm if they want to participate or not in a game by enabling a toogle.

🧠AI Lesson Prompt: Final Game

</> `Code Checkpoint: Final Game`

Self-Evaluation

It’s time to put what we’ve learned into practice! Here are 4 questions to check by yourself what you have learnt in this lesson.

Question 1

Question 2

Question 3

Why should players who join during an active game be placed in spectator mode?

To avoid disrupting the ongoing match and ensure fair gameplay

To prevent them from seeing the game

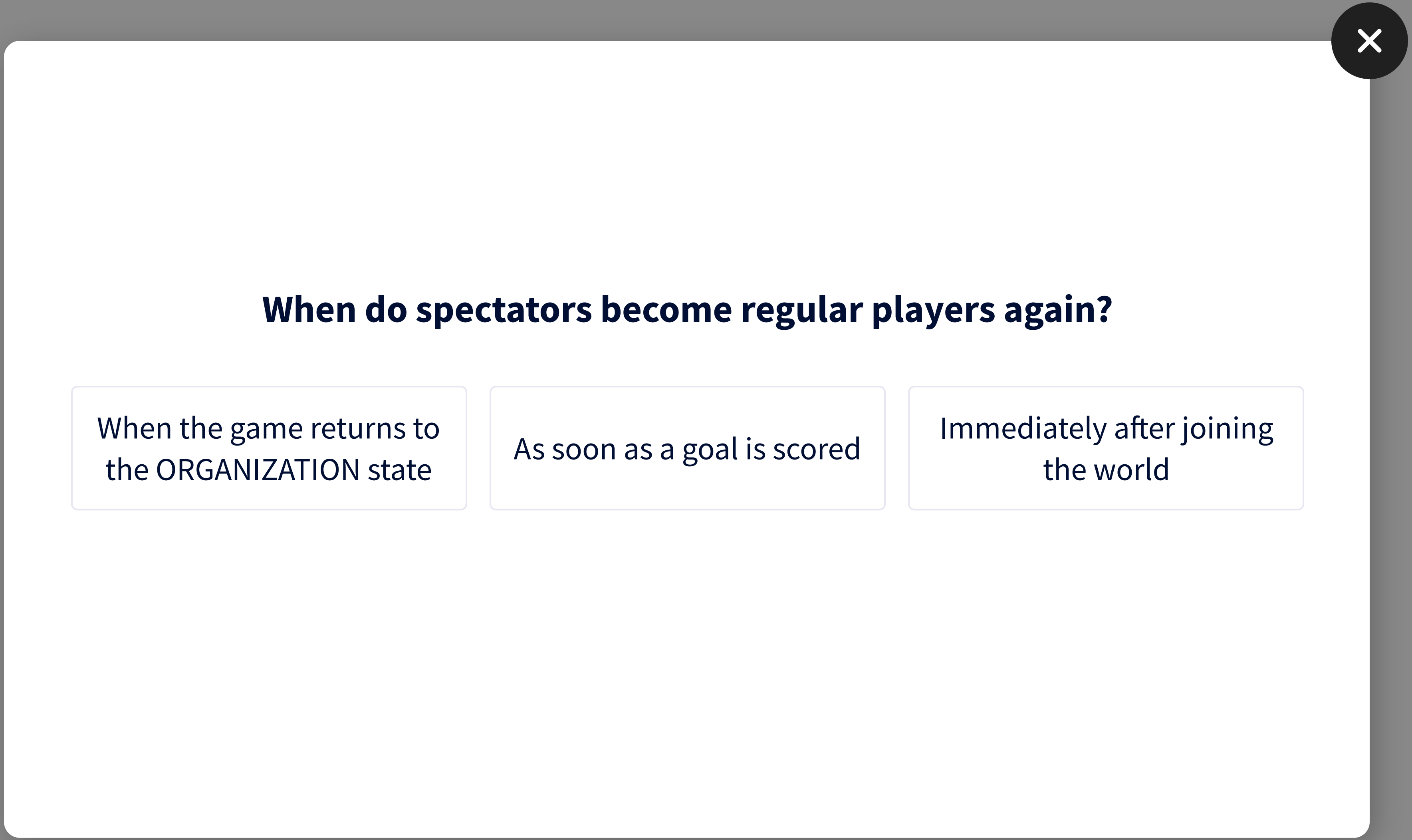
To reduce the number of players in the world

What is the purpose of the _isInGame flag?

To control whether the player can move or not

To check if the player is connected to the internet

To determine whether the local player is an active participant or a spectator



When do spectators become regular players again?

When the game returns to the ORGANIZATION state

As soon as a goal is scored

Immediately after joining the world

Help us to improve

Did you understand how _isInGame determines spectator state?

Write your answer here.

Send

Was teleporting late joiners clear in concept and implementation?

Write your answer here.

Send

Did you struggle with OnPlayerJoined logic?

Write your answer here.

Send

How can we improve the spectator experience or clarity of logic?

Write your answer here.

Send