🎓 LESSON 06 Goal Collision

Start

Video Link

LESSON GOALS

🎯 Goal Summary:

- Create goal trigger zones
- Detect ball-goal collisions
- Validate scoring team
- Implement ScoreController
- Add score UI to GameHUD
- Reset ball when scoring occurs

Next

Learning Objectives + Deliverables

Learning Objectives

Reset ball when scoring occurs

Detect collisions only when ball is carried by the scorer

Identify scoring team correctly

Implement networked scoring using UdonSynced

Update scores on HUD

Deliverables

✓ Goal zones detecting valid scoring events

✓ Networked scoring system

✓ Ball respawn after each goal

✓ GameHUD showing scores

GENERAL DESCRIPTION OF THE LESSON

High-Level Description

🎯 **Goal:** In this lesson we are going to complete the game loop of scoring, increasing the score and respawning the ball to start over.

🔴 LEVEL 1: Advanced Challenge

In this lesson we will:

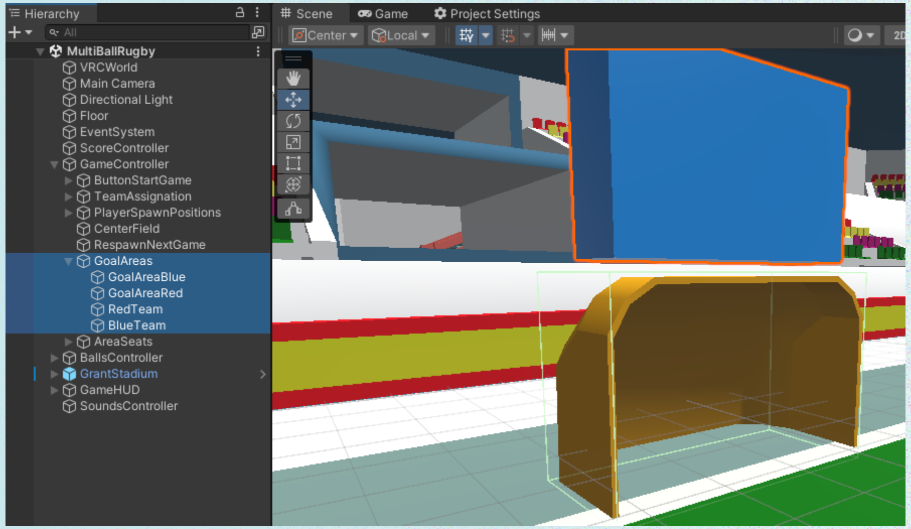
- Create invisible trigger areas for Red and Blue goals
- Add GameGoalArea script detecting ball collisions
- Validate conditions for scoring:
  - Only count if local player carries the ball
  - Only count if ball enters the rival's goal
- Implement ScoreController:
  - Track TeamRed and TeamBlue scores
  - Sync changes with UdonSynced
  - Update GameHUD
- Reset ball position after scoring
- Test scoring with multiple VRChat clients

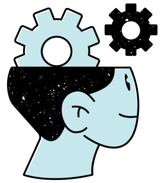
🧠AI Lesson Prompt: Goal Collision

Next

Exercise 1: Create the goal areas using trigger colliders.

- Actions:
  - Add two cube primitives for each goal: GoalBlue, GoalRed
  - Disable MeshRenders and ensure BoxColliders' isTrigger is true
  - Add colored cube visuals above them





Exercise 2: Create the **GameGoalArea** script that will handle the collision of the GameBall with the Goal.

- Actions (Part 1/2):

1) Before start, in (**GameController**), implement this public getter:

2) Create the Udon script (**GameGoalArea**) in the path (**/Game/Scripts/View/**) and define this member variables. Next attach the script to the GameObject created in the scene and initialize the fields:

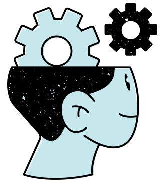
3) We will override (**OnTriggerEnter**) and when we detect a collision with a GameBall we will:

- Check if the ball is being carried by the local player. Use the previously implemented (**BallController::GetBallOwnedByLocalPlayer**)
- Check to what team belongs the local player. Use (**GameController::GetPlayerTeam**)

```
public class GameGoalArea : UdonSharpBehaviour
{
  [SerializeField]
  private GameController gameController;

  [SerializeField]
  private Team goalTeam = Team.TEAM_BLUE; // To which team belongs
private void OnTriggerEnter(Collider other)
{
   if (gameController == null) return;

   GameBall ballCollided = other.GetComponent<GameBall>();
   if (ballCollided != null)
   {
     GameBall ballOwned = gameController.BallController.GetBallOwnedByLocalPlayer();
     if (ballOwned == ballCollided)
     {
       Team ownerTeam = gameController.GetPlayerTeam(Networking.LocalPlayer.playerId);
       if (ownerTeam != goalTeam)
       {
          // RESET THE BALL
       }
     }
   }
}
```

```
public class GameController : UdonSharpBehaviour
{
   [SerializeField] BallController ballController;

   public BallController BallController
   {
      get { return ballController; }
   }
```

Exercise 2: Create the **GameGoalArea** script that will handle the collision of the GameBall with the Goal.



- Actions (Part 2/2):

4) If all the previous conditions meet, means that we have scored in the right goal and we need to respawn the ball in a random position of (BallController::**spawnPoints**).

- In the script (GameBall) we need to implement (**void ResetBall(Vector3 position)**) which will release clear the ownership and place the ball in the defined position.
- In the script (BallController) we need to implement (**void ResetPlayerBall()**) which will get a random position from (**spawnPoints**)
- Now back in the script (GameGoalArea) we are going to call (BallController::**ResetPlayerBall**) when a goal is scored.

5) Test it in the Unity Editor. The ball should be respawned when you collide with the right goal.

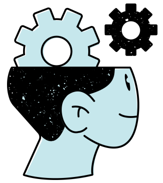
6) Test it with multiple VRChat instances (2 and 4 instances)

`Code Checkpoint: Goal Collision`

```
public void ResetBall(Vector3 position)
{
    impulseVector = Vector3.zero;
    ownerPlayerId = -1;
    this.transform.position = position;
    _hasBeenRespawned = true;
    RequestSerialization();
    OnDeserialization();
}
```

```
public void ResetPlayerBall()
{
    int idLocalPlayer = Networking.LocalPlayer.playerId;
    foreach (GameBall ball in balls)
    {
       if (ball.ownerPlayerId == idLocalPlayer)
       {
          ball.ResetBall(GetRandomRespawnPosition());
       }
    }
}
```

```
private void OnTriggerEnter(Collider other)
{
   if (gameController == null) return;

   GameBall ballCollided = other.GetComponent<GameBall>();
   if (ballCollided != null)
   {
     GameBall ballOwned = gameController.BallController.GetBallOwnedByLocalPlayer();
     if (ballOwned == ballCollided)
     {
       Team ownerTeam = gameController.GetPlayerTeam(Networking.LocalPlayer.playerId);
       if (ownerTeam != goalTeam)
       {
          gameController.BallController.ResetPlayerBall();
       }
     }
   }
}
```



Exercise 3: Create a **ScoreController** script that will keep the score of the game.

- Actions (Part 1/2):

1) Create an empty GameObject in the scene named ScoreController.

2) Create an Udon script (**ScoreController**) in the path (**/Game/Scripts/Controller/**) with these variable members. Add the script to the previous GameObject and initialize the gameHUD:

3) Still in the script (**ScoreController**), define these methods:

void IncreaseScoreTeamBlue() // Increase score and RequestSerialization

void IncreaseScoreTeamRed() // Increase score and RequestSerialization

void UpdateScore() // Will use gameHUD to update the score visually. You will need to create a new public method "SetScore(...)" in GameHUD and the visual elements (textfields, icons) to display the current score

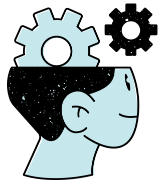
override void OnDeserialization() // Score updated

```
public class ScoreController : UdonSharpBehaviour
{
    public override void OnDeserialization()
    {
        UpdateScore();
    }

    // OTHER CODE ...
}
```

```
public void IncreaseScoreTeamRed()
{
   scoreTeamRed++;
   RequestSerialization();
   UpdateScore();
}
```

```
public class ScoreController : UdonSharpBehaviour
{
  private void UpdateScore()
  {
     gameHUD.SetScore(scoreTeamBlue, scoreTeamRed);
  }

   // OTHER CODE ...
}


public class GameHUD : UdonSharpBehaviour
{
  [SerializeField]
  private TMP_Text scoreBlue;
  [SerializeField]
  private TMP_Text scoreRed;

  public void SetScore(int goalsBlue, int goalRed) {
      scoreBlue.text = goalsBlue.ToString();
      scoreRed.text = goalRed.ToString();
  }

  // OTHER CODE ...
```

```
public class ScoreController : UdonSharpBehaviour
{
    [SerializeField]
    private GameHUD gameHUD;

    [UdonSynced, NonSerialized]
    public int scoreTeamBlue;

    [UdonSynced, NonSerialized]
    public int scoreTeamRed;
public void IncreaseScoreTeamBlue()
{
   scoreTeamBlue++;
   RequestSerialization();
   UpdateScore();
}
```



Exercise 3: Create a **ScoreController** script that will keep the score of the game

- Actions (Part 2/2):

4) Back to the (**GameController**), you will create a new public method:

void OnGoalScored(Team teamToScore) // Reset the ball and increase the score (only by the Networking.isMaster, use SendCustomNetworkEvent to ask the master to increase the score)

5) In the script (**GameGoalArea**), instead of reseting the ball, call OnGoalScored

6) Test in the Unity editor to verify the GameHUD is properly update with the rigth score.

7) Test with multiple VRChat instances (2 and 4) to verify the GameHUD is properly updated for all the players.

`Code Checkpoint: Score Controller`

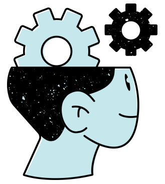
```
public class GameGoalArea : UdonSharpBehaviour
{
 // OTHER CODE ...

 private void OnTriggerEnter(Collider other)
 {
     if (gameController == null) return;
     GameBall ballCollided = other.GetComponent<GameBall>();
     if (ballCollided != null)
     {
         GameBall ballOwned = gameController.BallController.GetBallOwnedByLocalPlayer();
         if (ballOwned == ballCollided)
         {
             Team ownerTeam = gameController.GetPlayerTeam(Networking.LocalPlayer.playerId);
             if (ownerTeam != goalTeam)
             {
                 gameController.OnGoalScored(ownerTeam);
             }
         }
     }
 }
```

```
public class GameController : UdonSharpBehaviour
{
   // OTHER CODE ...

   [SerializeField]
   private ScoreController scoreController;

   public void OnGoalScored(Team teamToScore) {
      ballController.ResetPlayerBall();
      switch (teamToScore) {
          case Team.TEAM_BLUE:
               SendCustomNetworkEvent(VRC.Udon.Common.Interfaces.NetworkEventTarget.All, nameof(ScoreForTeamBlue));
               break;
          case Team.TEAM_RED:
               SendCustomNetworkEvent(VRC.Udon.Common.Interfaces.NetworkEventTarget.All, nameof(ScoreForTeamRed));
               break;
          }
   }
   public void ScoreForTeamBlue() {
      if (Networking.IsMaster) {
         scoreController.IncreaseScoreTeamBlue();
      }
   }
   public void ScoreForTeamRed() {
      if (Networking.IsMaster) {
          scoreController.IncreaseScoreTeamRed();
      }
   }

   // OTHER CODE ...
```

LESSON 06 COMPLETED

You now have:

- Full goal-scoring logic
- Networked score tracking
- Automatic ball reset
- HUD scoreboard
- Validation that only legitimate scoring attempts count



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

Question 4

Why are goal areas implemented using trigger colliders instead of normal colliders?

To detect when the ball enters the goal without blocking its movement	To apply physical force to the ball when it enters the goal	To make goals visible only to the Master player

Why should a goal only be counted when the local player is carrying the ball?

Because remote players cannot interact with triggers

Because balls without an owner do not have colliders

To ensure the scoring logic is validated locally and avoids duplicate scoring

What is the main responsibility of the ScoreController?

- To detect collisions between players and the ball
- To control the game timer and state transitions
- To store, synchronize, and update team scores across all players

Why is the ball reset immediately after a goal is scored?

- To increase game difficulty
- To reduce physics calculations
- To return the game to a fair and consistent state before play continues

Help us to improve

Was the goal trigger system easy to understand?

Write your answer here.

Send

Did you understand how ScoreController synchronizes values?

Write your answer here.

Send

Did you struggle with trigger detection or scoring conditions?

Write your answer here.

Send

What part of the scoring system felt unclear or too fast?

Write your answer here.

Send