Football Game Engine(Basic)

Andrew Manyore

INTRODUCTION

Thank you for purchasing Football Game Engine. Initially known as Soccer AI and now Football Game Engine, Football Game Engine (Basic) is simple implementation of a football(soccer) game. The package is ideal for learning purposes. That doesn't necessarily mean that it can't be used to make a game but the features inside this package are there to showcase the various concepts you can use to make your own game. You will definitely need to extend this package and make it fun.

I hope Football Game Engine (Basic) will serve your purpose and you will find it useful.

Please rate us on Unity Asset Store

CREDITS

INTRODUCTION

As I am basically a programmer with very little skills in other departments that are required to create this package, I had to rely on other assets. Like I mentioned before this is a basic implementation of a football game, thus most of the assets used in this package are merely Unity's primitive assets.

Don't worry about including the assets found in this project into your own project. There all covered by the appropriate licenses

GAME SOUNDS

- Football Ambience - https://freesound.org/s/424228/
- Football Kick - https://bigsoundbank.com/detail-1044-ball-kicked.html
- Goal Celebration - https://freesound.org/people/jordiroquer/sounds/148153/ cut it

INSTALLATION

INTRODUCTION

In this section you will learn how to install this package on your local machine

PREREQUISITES

For you to open and use this asset you will require a set of tools which are

- Game Engine – Unity3D
- C# IDE- Visual Studio (I used VS2017 Community Edition)

STEPS

Note: If you are going to import this package into an already existing project make sure you have your project backed up.

1. Simply import this project into a new or old project.
2. You are good to go

FEATURES

INTRODUCTION

In this section you will learn about the features in this package. Don't expect to get a fully blown out football game implementation will all the soccer rules. The core mechanics of a soccer match are implemented in this asset, you can easily learn how this asset has been built and you can easily add your own. That is how this asset has been designed.

FEATURE LIST

- Football match 90mins long with 2 halves, the first half and second half.
- Basic football artificial intelligence with the ability to attack, defend, press, pass, dribble, tackle and shoot
- Goal/Score implementation
- Match restart
- Game quit

MODEL LIST

- Goal model
- Pitch model
- The rest of the models you see in this package are primitive Unity3D assets

PLAYING FOOTBALL GAME ENGINE (BASIC)

INTRODUCTION

In this section you will learn about the features of this package. You will also learn how to open this package and interact with it.

STEPS

1. Go to the scenes folder and open the demo scene.
2. Hit play
3. The game will start. The interfaces are self-explanatory. You can navigate across them easily
4. In this game you control the yellow team, whilst the red team is controlled by the artificial intelligence

GAME CONTROLS

1. **W, A, S, D** to move around.
2. **N** to pass/press.
3. **Space** to shoot.

SETTING UP FOOTBALL GAME ENGINE (BASIC)

INTRODUCTION

In this section you will learn how to set up the Football Game Engine (Basic) as in the demo scene. If you have your own assets that you want to use this section will help you on how to substitute the package's assets with your own.

The package is designed in a way such that you only need to use the primitive Unity3D assets without using your assets. To then add your own models like the stadium, pitch and goal you only need to set it up on the side and place it such that it matches the pitch that you would have setup before. This makes swapping in and out of assets easier without messing up your scene.

SETTING THE MANAGERS

INTRODUCTION

In this section you will learn how to setup the different managers found in this application. In this package, all the managers are placed under one parent the Managers game object

THE GAME MANAGER

In this design the Game Manager is a simple class that displays the various in game menus depending on the state of the game

SETTING UP THE GAME MANAGER

1. Create an empty game object under the Managers game object and name it the GameManager.
2. Add the GameManager script to the GameManager gameobject.

CONFIGURING THE GAME MANAGER

The GameManager requires you to initialize the various menu panels with the appropriate data. In this example the RootUI game object holds the different menus needed be the Game Manager. You simply need to drag and drop the menu items into there appropriate fields as shown in the demo scene.

THE MATCH MANAGER

INTRODUCTION

The Match Manager manages the match proceedings. It listens to the pitch events and raised the appropriate events e.g. when a goal is scored, it stops the game and raises the match stopped event and gives the reason why, when a match starts it raises the take kick-off event so that the appropriate team can take the kick off etc.

SETTING UP THE MATCH MANAGER

1. Create an empty game object under the Managers game object and name it the Match Manager.
2. Add the Match Manager script to the Match Manager game object.

CONFIGURING THE GAME MANAGER

- Distance Pass Max – the maximum distance a player can pass the ball.
- Distance Pass Min – the minimum distance a player can pass the ball.
- Distance Threat Max – the closest distance to the player that a position will be considered to be threatening the player at his maximum
- Distance Threat Min – the furthest distance away from the player that a position will be considered to be threatening the player at its minimum
- Power – the maximum kicking power of the player
- Speed – the maximum speed the player can move with
- Team Away – a reference to the away team
- Team Home – a reference to the home team
- Root Team – a reference to the game object holding the teams
- Transform Centre Spot – A reference to the pitch's centre spot

THE SOUND MANAGER

INTRODUCTION

The Sound Manager handles all the game sounds you hear in this package. In this section you will learn how to set it up and configure it.

Setting up The Sound Manager

1. Create an empty game object under the Managers game object and name it the Sound Manager.
2. Add the Sound Manager script to the Sound Manager game object.
3. Create an empty game object and name it 'FootballAmbienceAS' and child it to the Sound Manager. Add the audio source component to it.
4. Create an empty game object and name it 'BallKickAS and child it to the Sound Manager. Add the audio source component to it.
5. Create an empty game object and name it 'GoalAS' and child it to the Sound Manager. Add the audio source component to it.

CONFIGURING THE SOUND MANAGER

1. Drag the 'FootballAmbienceAS' into it's appropriate field in the Sound Manager.
2. Drag the 'BallKickAS into it's appropriate field in the Sound Manager.
3. Drag the 'GoalAS into it's appropriate field in the Sound Manager.
4. Set the 'FootballAmbienceAS' to play on awake and loop.
5. Drag the appropriate sound clip into the 'FootballAmbienceAS'.
6. Drag the appropriate sound clip into the 'BallKickAS.
7. Drag the appropriate sound clip into the 'GoalAS.

THE TIME MANAGER

INTRODUCTION

The Time Manager handles the time update frequency which is utilized inside the game.

SETTING UP THE TIME MANAGER

1. Create an empty game object under the Managers game object and name it the TimeManager.
2. Add the TimeManager script to the TimeManager game object.

SETTING UP THE GAME ENTITIES

INTRODUCTION

The game entities are the numerous entities that the user interacts with during a match. There are numerous game entities in this package, and you will learn how to set them up.

THE BALL

INTRODUCTION

The ball is the game object that both teams will try to fight and get control of. The team controlling the ball will have to manoeuvre it and get it past the opponents' goal.

SETTING UP THE BALL

1. Create an empty game object and attach the ball script to it.
2. Attach the model to this game object. In this game object a sphere was used from Unity's primitive 3D objects with the sphere collider removed from it.
3. Adjust the sphere collider of your ball to match the dimensions of your ball.

THE GOAL

INTRODUCTION

The goal is an entity that will raise events when a team scores a goal. Each team has its own goal. The team will try to defend the goal during a game and attack the opponent's goal to score goals for itself.

The goal has a child game object called the Goal Trigger which is responsible for detecting those goals

SETTING UP THE GOAL

1. Create an empty game object and name Goal, attach the goal script to it
2. Create a cube child game object. Scale it such that it fits the dimensions of your goal area. Add the Goal Trigger to it. Set the layer to GoalTrigger. Drag it into the GoalTrigger slot of the goal in the inspector
3. Create a cube child game object. Remove the box collider. Attach it to the goal and place it behind the goal. The local x and z coordinate should be 0. In this example I placed it 5 units behind the goal. Drag the goal into the appropriate field in the Shot Target Reference Point slot in the inspector.
4. You can now save the goal as a prefab.

THE FORMATION

INTRODUCTION

Controls the positions of the players during various game scenarios depending on the position of the ball. There are three scenarios which are the kick-off scenario, the attack scenario and the defend scenario.

In the kick-off scenario the team will place every player on there kick-off position. The attack and the defend scenarios use a different approach.

The attack positions define the final position the player should be when the ball has been moved up the pitch when the team is attacking. Just like the attack positions, the defend positions define the positions for each player when the ball has been moved down the pitch when the team is defending. The team logic utilizes the players current home position to know where the player should be.

So, during an attack, the team tries to move the players ahead of the ball. The positions are moved from the defensive spots to attack spots depending on how far the ball has travelled up the pitch e.g. if the ball has travelled ratio 0.5 to the pitch's length the current home positions are moved by the same ratio up the pitch.

The same applies for when the team is defending. The only difference is the team will try to pack more players between the team's goal and the ball.

SETTING UP THE FORMATION

Setting the formation requires numerous steps. Check the formation prefab to see how it has been setup.

THE PLAYER SUPPORT SPOTS

INTRODUCTION

Player support spots are points on the pitch where players will query to find the point that allows the controlling player to pass to him. During an attack, players without the ball will try to find points on the pitch that are safe for the controlling player to pass to.

Setting up the Player Support Spots

1. Create a player support spot prefab by creating a cube, removing the box collider and adding the SupportSpot on this entity.
2. For your game to work you will need numerous of these, so create another empty game object, name it 'PlayerSupportSpots' and child this player support spot to it.
3. Duplicate the player support spots and place them around the pitch as you see fit. The more support spots you have the more dynamic your game will be.

THE PLAYER

INTRODUCTION

This is the game object that will represent the soccer player inside the game.

SETTING UP THE SOCCER PLAYER

1. Create an empty game and name it player.
2. Attach the player script to it.
3. Attach your model to the player. In this example Unity3D's primitive game object where used to create a model.
4. Adjust your player's capsule collider to match the dimensions of your model.

THE TEAM

INTRODUCTION

The team in general is a collection of the team's players. The package needs two teams to be setup, the home team and the away team. In this package, teams are placed under the team's game object which itself is placed under the entities game object

SETTING UP THE TEAM

1. Create a game object under the team's game object and name it 'TeamAway'.
2. Drag the Team script and attach it to this game object.
3. Attach the goal under the team game object. Note when you are placing the goal, the team object should be at the centre of the pitch and the z-offset should be at the end of the pitch. In this case I assumed that my pitch will be 100m long, so I placed the goal 50m relative to its parent.
4. Drag the team's goal into the appropriate field in the inspector.
5. Attach the formation to the team, make sure it's at position 0,0,0 relative to its parent
6. Drag the formation into its relative slot in the inspector.
7. Create an empty game object, name it KickOffRefDirection and make sure it pointing the direction that the player taking the kick-off should face during a kick-off. Drag it into the KickOffRefDirection in the team inspector.
8. Attach the player support spots onto the team. Drag the player support spots into the appropriate in the team inspector. Make sure the position is 0,0,0.
9. Create the RootPlayers game object and place it under the Team game object. Add as many players you need under this. Note if your formation has 11 positions make sure you put 11 players. Drag it into the RootPlayers in the team inspector.
10. Duplicate the team object, rotate it 180 degrees and name it 'TeamAway'.
11. Drag the 'TeamHome' into the opponent field of the 'TeamAway' inspector and do the same for the 'TeamAway'.
12. Drag the 'TeamAway' into the team TeamAway field in the MatchManager inspector.
13. Drag the 'TeamHome into the team TeamHome field in the MatchManager inspector.

CONCLUSION

At this stage your scene is ready for game play. Hit play and enjoy the game.

USING YOUR OWN MODELS

INTRODUCTION

Different ways of using your models have been explained on numerous occasions along the documentation. This chapter though will focus on showing how you can have one base game environment and easily swap stadium models in and out without breaking the game and having to reconfigure it.

SETTING IT UP

If you have played around with the scene, you would have seen that there is the game world root where everything from the goals to the teams is set. Then there is the models root where only the models are set. Simply set up your models so that they match the dimensions and locations of the goals of your pitch. Then place the items that you don’t want to be rendered by the camera under the helper’s layer. The camera will only render your models.

YOU MIGHT ALSO LIKE

- Intelligent Selector
- Robust FSM
- Soccer AI
- Super Goalie Basic