Football Game Engine(INDIE)

Andrew Manyore

CONTENTS

INTRODUCTION

Thank you for purchasing Football Game Engine. Initially known as Soccer AI and now Football Game Engine, Football Game Engine (Indie) is a basic implementation of a football(soccer) game. The package is ideal for learning purposes and for publishing. You can also integrate it into your ow project or use it as the base project for your game. Though Football Game Engine (Indie) is game ready, it doesn't implement all the rules for a football game.

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

Please rate us on Unity Asset Store

CREDITS

INTRODUCTION

To produce this asset, I had to rely on some other assets. I am very grateful to the guys who made their assets available from free. Making this asset wouldn't have been possible without you guys

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

ANIMATIONS

- Soccer Game Pack - https://www.mixamo.com/#/
- Mecanim Animation Pack – decapreated from Unity Asset Store
- Imtrobin - http://envisagereality.com/friend/soccer/SoccerModelAndAnimation.zip

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 VS2019 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.

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
- Throw In
- Goal Kick
- Corner Kick
- Match Pause
- Math Resume
- Match restart
- Game quit

MODEL LIST

- Goal model
- Pitch model
- Low Poly Football player
- High Poly Football Player
- Soccer Ball

PLAYING FOOTBALL GAME ENGINE (INDIE)

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 short pass/press.
3. **M** to long pass.
4. **K** to sprint.
5. **Space** to shoot.

SETTING UP FOOTBALL GAME ENGINE (INDIE)

INTRODUCTION

In this section you will learn how to set up the Football Game Engine (INDIE) 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 SCENES

INTRODUCTION

In this section you will learn how to setup the scenes in this asset. Football Game Manager requires two scenes. The MainScene and the GameScene. Remember to add these scenes to your build hierarchy with the MainScene as the first scene.

THE MAIN SCENE

This is the first scene to be loaded. This is where the various menu panels show. The main purpose of this scene is to setup the game and prepare for a match. For the actual match, a different scene is used.

SETTING UP THE MAIN SCENE

First of all, you have to note everything setup in this scene requires to be persistent in other scenes. To achieve this every other game object is placed under the 'PersistentGameObject'. Various Managers need to be setup in this scene. The list of the Managers that need to be setup includes:

- DataManager - handles saving and retrieving of data.
- GameManager - controls every aspect of the game including other managers.
- GraphicsManager – controls the GUI of the game
- SoundManager – controls the game sounds

Besides setting up the managers, you also need to setup the game ui. There is already a UI game object setup for you. Make sure this UI component is also placed under the 'PersistentGameObject' as we need it in all our scenes

THE GAME ON SCENE

This is where you play the match. In this scene that's where you find the MatchManager, TimeManager, teams and players

SETTING UP THE GAME ON SCENE

What you have to take note of in this scene is setting up the MatchManager. The Manager depends on the teams, which depends on the players to be set properly. Here is how the different managers work.

- MatchManager – Controls the match. It receives data from the GameManager when the scene is loaded. It then initializes the teams with their data.
- TimeManager – is a simple class that controls the time in the game.

Besides the managers, there are also some entities that need to be setup in the game scene. These entities include:

- Ball – a sphere that we are all fighting for lol.
- Teams – we have two of them the home team and away team
- Players – we have 22 of them, 11 for each team
- Pitch – the turf was the match is played.

You also need to take note of one thing. The stadium with all its textures is a whole different mesh. You can remove it without disturbing the logic of the game. This makes it very easy to swap in your own stadium model. It also helps in cases where you want to load the stadium model dynamically. You can simply replace the stadium model.

SETTING THE MANAGERS

INTRODUCTION

Football Game Engine (Indie) relies on numerous managers for its functionality. Each manager handles a specific aspect of the asset. At the top of all is the ‘GameManager’ which controls every other manager.

THE GAME MANAGER

INTRODUCTION

As explained before the GameManager controls every aspect of your game. It should be set-up in the MainScene under the ‘PersistentGameObject. Check the MainScene for more info.

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

- Settings Save File Name – File name to be used when saving our settings.
- Teams Data Save File Name – File name to be used for saving teams (not in use)
- Default Settings – Settings that come bundled with asset before the user uses his
    - Match Settings – Settings for the match
        - Half Length – The length of each half in minutes
        - Match Difficulty – The difficulty level for each match
    - Sound Settings – Settings for the sound
        - Is Sound On -Disable or enable all sounds
        - Music Volume – Volume for music sounds
        - Sound Volume – Volume for sfx sounds
- Match Difficulties – This is a list of various difficulty levels that for this game. The asset comes with 3 difficulty levels predefined for you. Tests were done using the normal difficulty level, the other levels are there to show you how it works. Data specified in the difficulty level is the one that will initialize similar data in the team manager, team and players. Here is how to configure a difficulty level:
    - Name – name of difficulty level used for display.
    - Match Difficulty Type – a unique field used to identify this difficulty level.
    - Team Params – Parameters to be used by a specific team. For each level you have to define such parameters for the cpu and user-controlled team.
        - Global Control Variables
            - AI Update Frequency – wait in I seconds to update player ai.
            - Tight Press Frequency – Frequency for tightly marking the player with the ball.
            - 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 Shot Valid Max – This is the distance from opposition goal that we should start considering making shots at goal.
            - Distance Tend Goal – The distance from the goal the keeper should be when protecting the goal.
            - 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.
            - Distance Wonder Max – Maximum distance the player can move from its home position.

        - Velocity Long Pass Arrive – Ball arrival velocity when long pass is made.
        - Velocity Short Pass Arrive – Ball arrival velocity when short pass is made.
        - Velocity Shot Arrive – Ball arrival velocity when shot is made.
    - Player Control Variables
        - Dive Distance – Maximum distance the goalkeeper can dive.
        - Dive Speed – Maximum dive speed for the goalkeeper.
        - Jog Speed – Normal move speed for the entities.
        - Jump Height – Maximum jump height for the goalkeeper.
        - Power – the maximum kicking power of the player.
        - Reach – the maximum distance the goalkeeper can reach from its position if it extends its hand.
        - Speed – the maximum speed the player can move with.

THE DATA MANAGER

INTRODUCTION

This a utility class for saving and retrieving of data for this asset. It holds various data objects that are utilised inside the game. This should be placed under the 'PersistentGameObject' in the main scene.

SETTING UP THE DATA MANAGER

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

- Attack Tactics – A list of numerous attack tactics used in the game. For each attack tactic, you have to specify various constraints for it. These are the constraints:
    - Name – Name of tactic for display.
    - Forward Run Probability – Frequency to make runs towards the opponent goal by the players.
    - Long Ball Probability – Frequency for making long passes forward by the players.
    - Push Ahead Ratio – Ratio to stay ahead of ball during attack.
    - Push Forward Frequency – Frequency of team to push player home positions up the field during an attack.
    - Transit into Attack Speed – How fast should the team transit from the defend state into the attack when possession is regained.
    - Attack Type – A unique key to identify this tactic.
- Defend Tactics – This is a list of the various defend tactics that are used inside the game/asset. You must specify numerous constrains for each tactic. These are the constraints:
    - Name - Name of tactic for display.
    - Push back ratio - Ratio to stay behind the ball during defending.
    - Tight press frequency – Refers to how frequently we should tightly press the opponents.
    - Transit into Defend Speed – How fast should the team transit from the attack state into the defend state when possession is lost.
    - Defend Type – A unique key to identify this tactic.
- Teams – A list of the teams that are found in this. You must specify certain attributes for each team. These attributes are:
    - Left Side Corner Kick Taker – The id of the player to take the corner kick on the team's left side.
    - Left Side Throw-In Taker – The id of the player to take the throw-in on the team's left side.
    - Right Side Corner Kick Taker – The id of the player to take the corner kick on the team's left side.
    - Right Side Throw-In Taker – The id of the player to take the throw-in on the team's left side.
    - Formation Name – The name of the formation being used.
    - Short Name – The team's short name.
    - Icon – Icon for this team.
    - Attack Type – The attack tactic for this team.
    - Defend Type – The defend tactic for this team.
    - Formation Type – The formation used by this team

- Kits – A list of the kits for this team. You must specify these attributes for each team:
  - Name – The name of the kit
  - Icon -Icon of this kit used for display.
  - Goalkeeper Kit – Kit for the goalkeeper
  - In Field Player Kit – Texture for the in-field player
- Players – Each team has a set of players. You don't necessarily need to specify all the team's players. Just specify the attributes of the first 11. The attributes of the players are specified as ratios. They're ratio of the corresponding variables that are specified under the match difficulty in the Game Manager. For example, if we have specified the player to be 0.8 and, in the tactic, we have specified the power to be 20, the actual power of the player is calculated as actual power = 0.8 * 20 = 16. For each player you need to specify these attributes:
  - Can Join Corner Kick – A reference to whether this player should join the attack during a corner kick.
  - Accuracy – Ratio of the accuracy of this player in hitting its targets
  - Dive Speed – Ratio of the dive speed for this player.
  - Goal Keeping – Ratio of the goalkeeping ability of this player.
  - Jump Distance – Ratio of the jump distance of this player.
  - Jump Height – Ratio of the jump height of this player.
  - Power – Ratio of the power of this player.
  - Reach – Ratio of the reach of this player.
  - Speed – Ratio of the speed for this player.
  - Tackling Ration of the tackling of this player.

THE MATCH MANAGER

INTRODUCTION

This controls the match. It should be set-up in the GameOnScene. You don't need to specify all of the Match Manager constraints as some are initialized by the Game Manager using data from the team's tactics and the selected match difficulty.

CONFIGURING THE MATCH MANAGER

These are the variables that you need to specify:

- Left Throw-In Trigger – Reference to the left throw-in trigger.
- Right Throw-In Trigger – Reference to the right throw-in trigger.
- 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. The Sound Manager should be set-up under the in the Main Scene under the 'PersistentGameObject'.

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.

CONFIGURING THE SOUND MANAGER

You don't need to configure all the variables as some are initialized by the Game Manager from the settings object. These are the variables you need to configure:

- Audio Source Components – A list of the audio sources used in this asset. The asset was made to use 5 game sounds with predefined Id's. It will be wise to only change the audio clip unless you know what you are doing. Here is what you need to specify for each audio source component:
    - Can Loop – A reference to whether this component should loop.
    - Id – The unique id of this instance, please don't change unless if you know what you are doing.
    - Sound type – The group were this component belongs to.
    - Audio Clip – The audio clip of this component.

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.

Special THANKS

I would like to offer special thanks to Imtrobin, for supporting my asset in it's earliest stages. I'm sure he is the first person to buy my asset after donating some models to me. After that he wrote a review and suggested some features for me. That was a good example for someone who had just started on game development.

YOU MIGHT ALSO LIKE

- Intelligent Selector
- Robust FSM
- Football Game Engine (Basic)
- Super Goalie Basic

WHAT'S NEXT

I fee l there is a lot of ways that I can improve this package. I still have the dream of making this asset a complete template for creating a football game. Development has been slow for various reasons and one of the main reasons is that my skills in game development were limited. I have learned a lot of stuff during these past years and I'm now confident that I can now push this package forward.

Over the course of the months I will be fixing bugs. I won't add new features yet but will work on the current features to make the work properly. There's room for improvement in the corner-kick, throw-in, goal kick and punt kick. Theirs is also room for improvement in the asset in general. My goal is not to make something at FIFA or PES standard, but is to make a simple, fun football game asset.

After this stage is done I will move into adding other features that are lacking. These features include:

- Free-kick
- Penalty
- Extra – Time
- Referee
- Substitutions
- Team Management

After this stage has been completed the next stage is to add this set of features

- Injury
- Competitions

At the very end

- Add ads and in-app purchases.

At this stage my dream will be realised.

CONTRIBUTIONS

If you feel you can contribute to this asset feel free to contact me. The asset lacks in user UI etc. Any form of input will be greatly appreciated.

THANK YOU

ENJOY