Online PVP 3D football Game Template

Requires Unity3D v2020.1.15f1+
Works on Android, iOS, WebGL and Standalone

Dear Customer,

Thank you so much for purchasing our "finger soccer 3D" game kit. Here you can learn about the important things you need to know in order to use this asset with maximum proficiency.

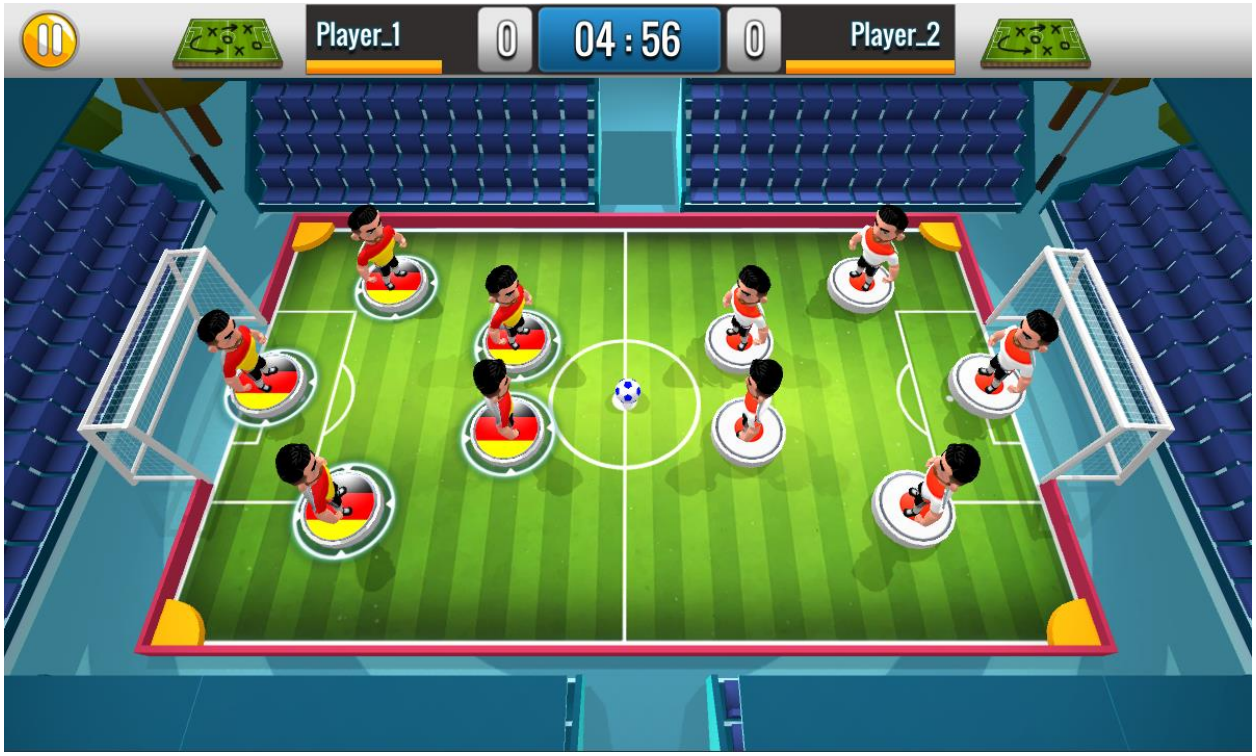
All classes are already fully commented, but if you ever had a question about anything, feel free to contact us at http://www.finalbossgame.com .

We'll try our best to help you with your questions ASAP, till you are fully satisfied.

Overview

Finger Soccer 3D is a flexible football game kit that can be played as a single player (player vs AI), 2-Players (two players on a same device) or online match where different players from all over the world can join and compete with each other in real-time. Players can simply select their units and adjust the power and direction of the shoot. There is also plenty of defensive and offensive formations available which players can unlock using the coins they receive as win prizes.

The finger soccer 3D game template is so flexible that you can use it as a strong base to prototype many other similar games. You can easily add new character models, stadiums, animations to offer a unique gameplay experience to your audience.



The finger soccer 3D asset uses real world physics to simulate the interaction with ball & units, and uses PUN (photon unity networking) to manage the online system. The online multiplayer module is able to handle the connections gracefully and offer a smooth and bug free online gaming experience.

This game kit features a complete game flow with menu, shop, configuration and game scenes. The game is also comes integrated with the Unity IAB and accepts all sorts of micro-transactions. You are minutes away from having a complete game that can generate huge money for you!

This project accepts both touch and mouse inputs, and can be played on **Android**, **iOS**, **WebGL** and **Stand-Alone** platforms simultaneously, i.e. an Android player is able to be paired against an iOS player easily!

Game Play

The goal of this game, like every other soccer game is to get the ball into the opponent's gate. The game-manager grants each player their turns to play. When you have the turn, you can select one of your units that you think has the best chance to score a goal. Then you can tap on the selected unit, hold your finger and drag it away to adjust the power and direction of the shoot.

The underlying processor works similar to billiard or any other similar games. It processes everything with real world physics calculation to deliver a rich and rational response. When testing the game inside editor, you can see all the debug lines that are rendered to help you see what is happening.

The game uses a simple formation system that positions the units on the field. You have full control over this system and can add/remove/modify unlimited number of formations to the game. You can make some of these special formations non-free and give them to your players as rewards or have them buy it from the game's shop with their money.

The game's time is also flexible and can be easily set on any value you desire. You can also provide additional options to your players and let them choose the time of the game on their own.

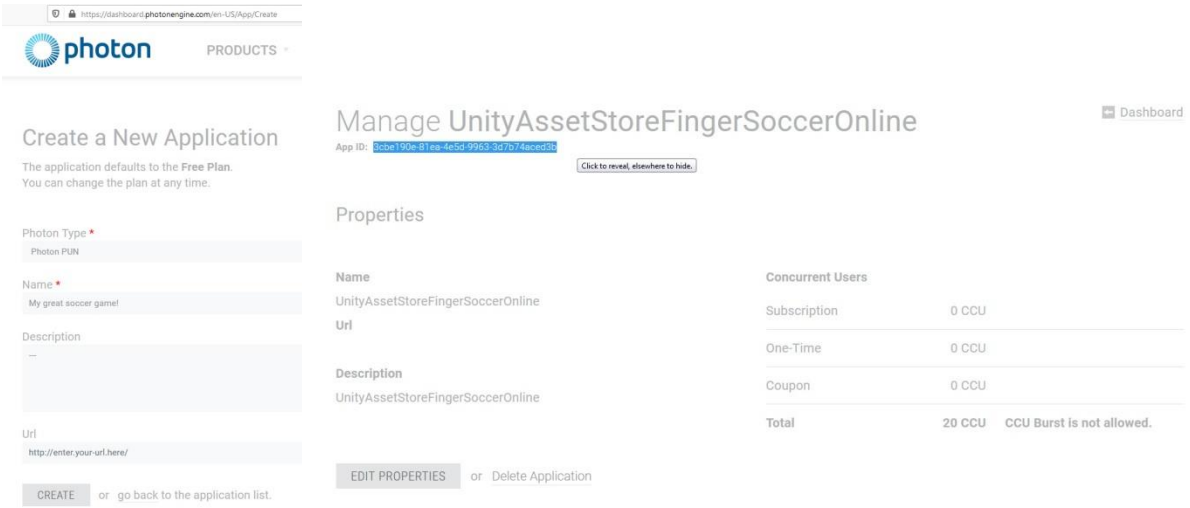
By default, the game is finished when the time runs out. So, the player that lands more goals is considered the winner. But you can also configure the game to finish on special occasions like when a player reaches a certain number of goals, or after a certain amount of time is passed.

Importing the Asset - Initial setup

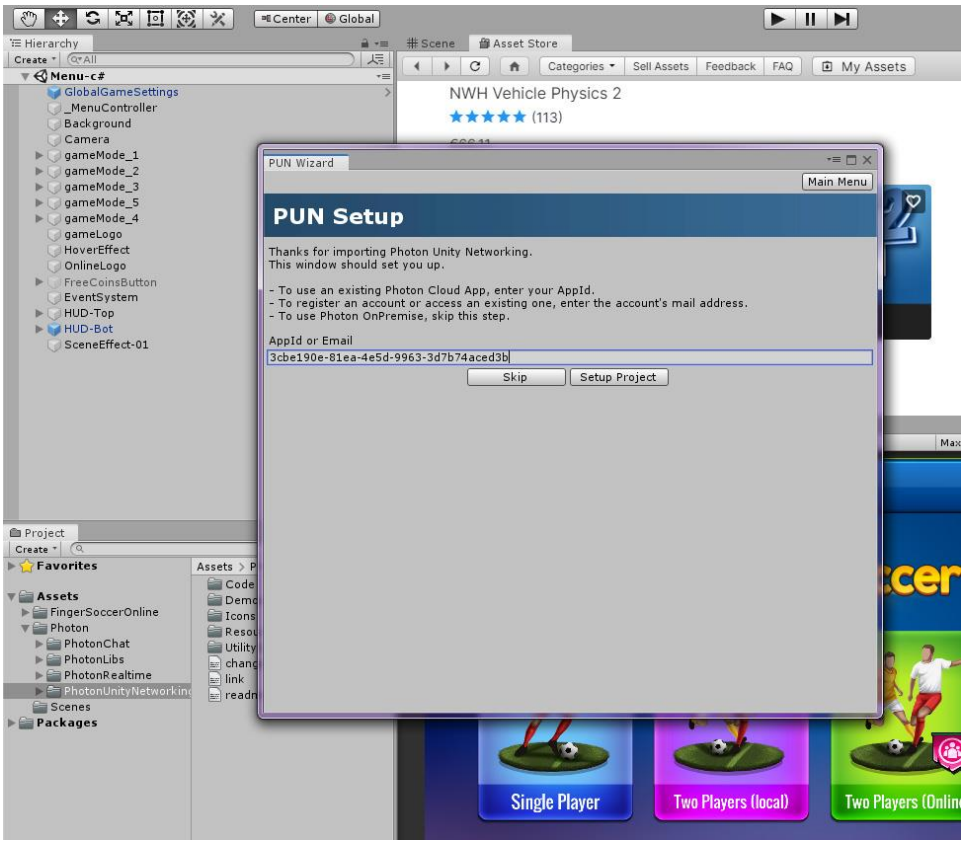
After you've downloaded the asset and imported it into a new blank project, you will receive a lot of error messages on the console. Don't panic! It's perfectly normal. The game is looking for missing Photon (PUN) components and we are going to add PUN package right away.

Search for "Photon pun 2 free" on Unity Asset Store or open ***this URL*** in your browser in order to download and install Photon unity networking package. Once the installation of photon is complete, you can see that all errors disappear and you can actually run the game. But before doing that, there is another important step that you need to take before playing the game. You need to register a free Photon account and setup a demo project in order to get an App-ID. We need this app-id to configure and finish our photon setup in a proper way.

Head over to ***Photon Signup page*** and create your free account. Then go to the ***dashboard*** and create a new app like this:



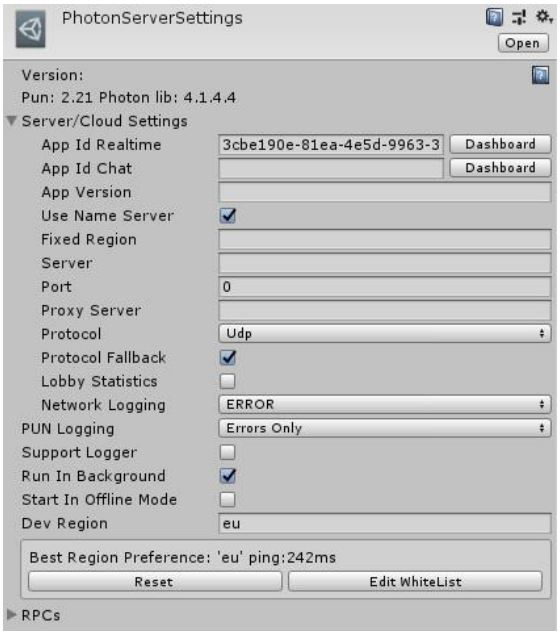
You need to copy the entire App-ID and enter it in Photon setup window that opens up in your Unity editor like this:



After that, just hit "Setup Project". Photon will create the required configuration files for you and your game should be ready for live testing.

Important: You can always change your PUN settings by going to "Window->Photon Unity Networking->

Highlight server settings". In the panel that opens, you can edit all your PUN related settings, including the server region and App-ID.

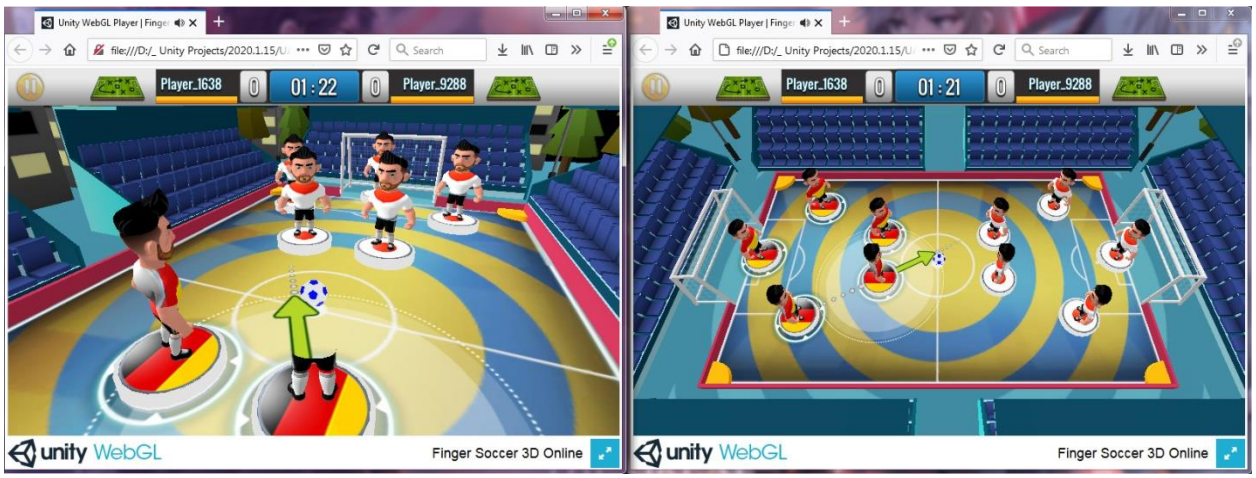


Testing the game

Once the above steps are done, you are ready to test your new online soccer game. Since we need two instances of a game in order to be able to test the online functionality, its best to proceed with WebGL builds, since we can open multiple instances of our game in different windows (of a browser). You just need to notice that a few parts of the game (cup amount, player money, etc) might not work correctly, since both webGL instances are accessing the same PlayerPrefs data stored on the browser. But this should not be an issue at all once you are publishing your final game on Android, iOS or any other platform, since they will not use a shared PlayerPrefs storage.

You can run the game by opening "menu-c#" or "***Init***" scene.

You can test most parts of the game in your editor. But to check the online mode, you need to build the game for "WebGL" and then run two instances of your build (or one instance of WebGL build and using your editor as the second player) in order to create a lobby, join a match and start playing.

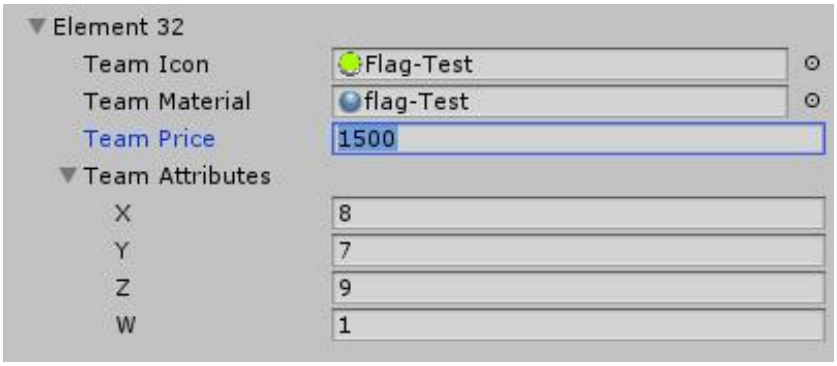


Adding new Teams

Each new team needs to have an icon (flag), a material and attributes (power, time, aim, price, locked/unlocked state). To add a new team to the game, you need to prepare all the above items beforehand. Now follow these steps to add a new team:

1. Select and open "Prefabs->General-> GlobalGameSettings" prefab object. It's already populated with 32 default teams. Change the available teams array size to 33 and open the 33th element inside the array.

2. Set your desired values for different items (Use this image for a better understanding) and you are done!



x: power (1 to 10) / y: time (1 to 10) / z: aim (1 to 10) / w: lock state (0:locked/1:unlocked)

Editing existing teams

Like the steps above, you just need to open the team's respective field inside the main array (on main prefab object) and change the values as you see fit. You can easily change the flag, price and attribute values of all existing teams inside the game.

Adding new Fields

Each new field requires an icon, a material and a big image to be used as the main texture for the field object. Once the above items are prepared, please follow these steps:

1. Select and open "Prefabs->General-> GlobalGameSettings" prefab object. It's already populated with 7 default fields. Change the available teams array size to 8 and open the 8th element inside the array.

2. Set the required items for each field and save the prefab. You are done!

In the next update, we will add the ability to add new 3d stadiums to the game. So, you can change both stadium 3d model and the field texture.

Adding new Formations

Each new formation requires a string (i.e. 1-2-2), an icon, a material, price, locked/unlocked state and position of each unit inside that formation. Once the above items are prepared, please follow these steps:

1. Select and open "Prefabs->General-> GlobalGameSettings" prefab object. It's already populated with 5 default formations. Change the available teams array size to 6 and open the 6th element inside the array.

2. Set the required items for each field and save the prefab.

3. Open " FormationManager" class and add a new case (6th element) to the " getPositionInFormation" method. Then edit the X & Y parameter of each unit as you see fit. Save the class . Your new formation is ready to be selected in game-configuration scene or be purchased using the in-game shop.

```
64              case 3:
65                  if(_UnitIndex == 0) output = new Vector3(-14, 0 + yFixer, fixedZ);
66                  if(_UnitIndex == 1) output = new Vector3(-11, 5.5f + yFixer, fixedZ);
67                  if(_UnitIndex == 2) output = new Vector3(-11, 2 + yFixer, fixedZ);
68                  if(_UnitIndex == 3) output = new Vector3(-11, -2 + yFixer, fixedZ);
69                  if(_UnitIndex == 4) output = new Vector3(-11, -5.5f + yFixer, fixedZ);
70                  break;
71
72              case 4:
73                  if(_UnitIndex == 0) output = new Vector3(-14, 0 + yFixer, fixedZ);
74                  if(_UnitIndex == 1) output = new Vector3(-12.5f, 2.5f + yFixer, fixedZ);
75                  if(_UnitIndex == 2) output = new Vector3(-9, 4.5f + yFixer, fixedZ);
76                  if(_UnitIndex == 3) output = new Vector3(-5, 5.5f + yFixer, fixedZ);
77                  if(_UnitIndex == 4) output = new Vector3(-1.5f, 5.5f + yFixer, fixedZ);
78                  break;
79
80              case 5:
81                  if (_UnitIndex == 0) output = new Vector3(-14, 0 + yFixer, fixedZ);
82                  if (_UnitIndex == 1) output = new Vector3(-12.5f, 2.5f + yFixer, fixedZ);
83                  if (_UnitIndex == 2) output = new Vector3(-9, 4.5f + yFixer, fixedZ);
84                  if (_UnitIndex == 3) output = new Vector3(-5, 5.5f + yFixer, fixedZ);
85                  if (_UnitIndex == 4) output = new Vector3(-1.5f, 5.5f + yFixer, fixedZ);
86                  break;
87          }
88
89          return output;
90      }
91
```

Using IAB/IAP

This template is featuring a shop system, including two different scenes that lets you purchase and unlock new teams & formations. However, it’s not using Unity's built-in IAP service by default, so you need to activate that service on your project and define new products to let your players to initiate a purchase request, if you want to connect the game to real products available on your dashboard.

To enable Unity’s IAP service, please follow these steps:

1) Go to "Window->General->Services" and enable "In app purchasing" service on your project. Then follow the official “Instruction Manual” on how to configure & add/edit new products to the service. Please make sure that you are using the exact same naming pattern for the items you are defining in your Google Play/AppStore dashboard.

2) Here are some useful instructions on how to setup a working IAB system:

https://docs.unity3d.com/Packages/com.unity.purchasing@4.1/manual/Overview.html

https://docs.unity3d.com/Manual/UnityIAPSettingUp.html

https://learn.unity.com/tutorial/unity-iap

Re-skinning the game

This template is using 2D png textures [Sprites] for UI and game objects. It is also using 3d models with animations for the units and soccer stadium. To re-skin the game, you need to edit/replace the images/models that are available in asset folders. Please notice that to ease the re-skinning, we already presented a PSD (open-layered Photoshop file) version of important sprites, so you can easily edit them the way you see fit. For the rest of the images, you can simply replace them with a new replacement that is using the exact same width/height as the original texture. Failing to do so might result in blurred or stretched result.

Support

If you have any questions, feel free to write me a message at http://www.finalbossgame.com and I will get back to you ASAP ;)