Air Hockey Game Kit

by ETI

v 2.0

Change Log

Always backup you project before updating !

V 2.0 :

This is a major update - Let me know if anything broke in the process.

- Deprecated UnityScript (in line with Unity's updated UAS publishing rules)
- (finally !) switched game UI from custom one to **Unity**'s **uGUI**
- Set minimum compatible version to **Unity Editor 5.4.5f1**
- Debug GUI is now disabled by default - you can enable it by checking 'Show Debug GUI' in **MainScript** inspector.
- There is some undocumented code in **MainScript.cs** regarding player levelling as example

Known issues :
- There is a know Unity bug regarding Touch Screen handling according to Unity versions and build platforms. However it is hard to track, i'll do my best to help if you encounter it and need support.

v 1.4 : Optimization for ***Unity 5*** compatibility

v 1.3 : Fixed a physic bug in the *C Sharp* version

v 1.2 :

- Air Hockey Table low poly 3D model added
- **ColliderBounds** object mesh updated
- **Puck** Sphere collider replaced with capsule collider
- **PuckScript** : puck bound limits deactivated in **Update()** function
- **MalletPlayerScript**, **MalletCPUScript** : Mallets move limits slightly modified
- In demo mode (**menu** game State) puck is now replaced after scoring.
  (changes made in **MainScript UpdateScore()** function)
- Game preferences : difficulty, score to win, camera view, are now stored in Player Preferences
  (changes made in **MainScript** and **UIButtonScript**)

v 1.1 : Pause, camera selection, Level Up system example.

v 1.0 : first release

Requirements

This asset requires **Unity 5.4.5f1** or later version

Overview

General Description

'Air Hockey Game Kit' is a Unity complete project package aimed to give you hints on creation of your own physic based air hockey or similar game.

The project was designed to be cross-platform, and optimized to run smoothly on mobile devices. You'll find an example of how to handle 2 players game on touch screen devices.

Scripts are thoroughly commented and nothing fancy is added to the template, in order to keep it clear and efficient.

Key Features

- Real world scale and physics, local space based code : this is meant to facilitate possible integration and placement into another project

- Well organized, fully commented scripts

- Tight to essential : Even though an object-oriented User Interface example is provided, you will not find unnecessary assets or code

Use cases

- Help beginner developer to learn how to create his own game, or have a working template to build upon

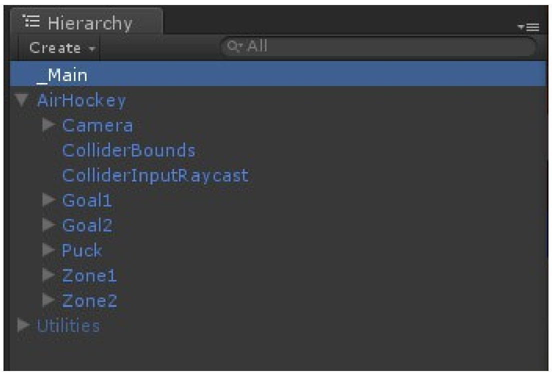
- Give intermediate developer another point of view on workflow, tips, and optimization

- Give a starting point to integrate a playing air hockey table into another project, for example a FPS or sandbox game. 'Air Hockey Game Kit' could be a handy way to do that (you should look at the main script for reference)

System Overview

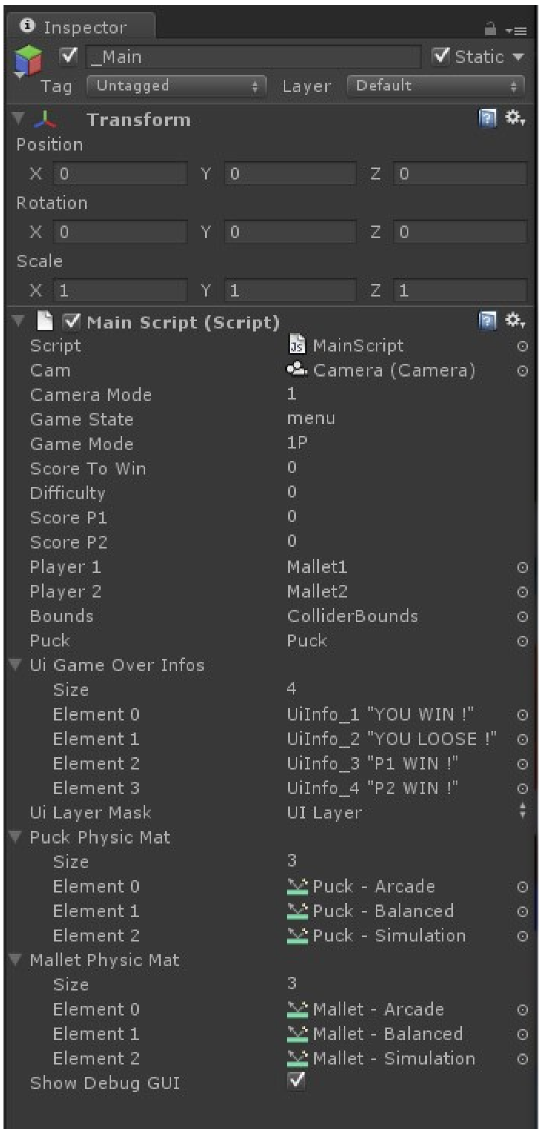
Main script

MainScript controls the main functions of the game. It receives and manages the different game and UI events. This script is attached to **_Main** gameObject.



```
_Main
▼ AirHockey
    ► Camera
      ColliderBounds
      ColliderInputRaycast
    ► Goal1
    ► Goal2
    ► Puck
    ► Zone1
    ► Zone2
► Utilities
```

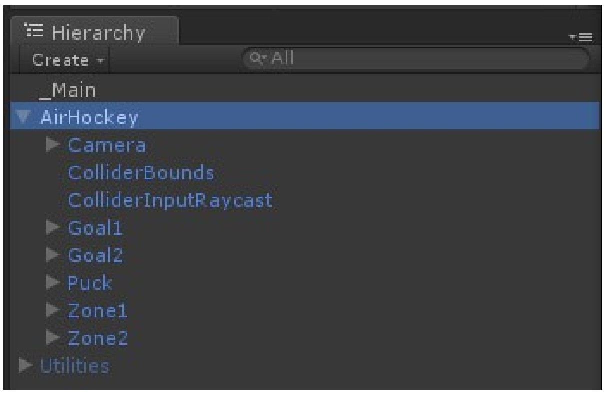
This asset use a single scene for the various game states (menu, game sessions, gameover...), this avoids loading times and optimizes the project, especially if you target low-end mobile devices.



```
_Main
Tag  Untagged        Layer  Default
Transform
Position
  X 0        Y 0        Z 0
Rotation
  X 0        Y 0        Z 0
Scale
  X 1        Y 1        Z 1
Main Script (Script)
Script                  MainScript
Cam                     Camera (Camera)
Camera Mode             1
Game State              menu
Game Mode               1P
Score To Win            0
Difficulty              0
Score P1                0
Score P2                0
Player 1                Mallet1
Player 2                Mallet2
Bounds                  ColliderBounds
Puck                    Puck
Ui Game Over Infos
  Size                  4
  Element 0             UiInfo_1 "YOU WIN !"
  Element 1             UiInfo_2 "YOU LOOSE !"
  Element 2             UiInfo_3 "P1 WIN !"
  Element 3             UiInfo_4 "P2 WIN !"
Ui Layer Mask           UI Layer
Puck Physic Mat
  Size                  3
  Element 0             Puck - Arcade
  Element 1             Puck - Balanced
  Element 2             Puck - Simulation
Mallet Physic Mat
  Size                  3
  Element 0             Mallet - Arcade
  Element 1             Mallet - Balanced
  Element 2             Mallet - Simulation
Show Debug GUI          ✓
```

« AirHockey » container

AirHockey gameObject contains all the other assets of the project, including the camera. All chidrens' behaviors are constrained to **AirHockey** position and rotation.



_Main
AirHockey
Camera
ColliderBounds
ColliderInputRaycast
Goal1
Goal2
Puck
Zone1
Zone2
Utilities

User Interface

A simple Unity UI interface is included in the example scene. It basically send Event messages to the main script when used.

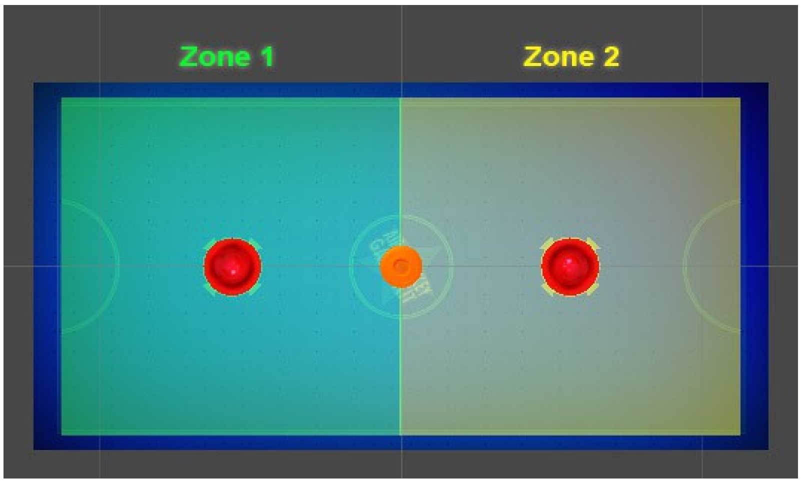
The UI elements are parented to the main UI canvas located inside the **Camera** gameObject.

Zones

Zones depicts the two sides of the playing surface.

Each zone contains several elements relative to it : **Mallet**, **MalletTarget**, and **PuckTarget**. These elements are described bellow.

Zone objects are displayed in **Debug Layer**.
Zones and their elements' display are differentiated by color.



Mallets

Each mallet contains two scripts to act as a CPU or an user.

The CPU (AI) control script is named **MalletCpuScript,** and the user control script **MalletPlayerScript**. These scripts use “target objects” described bellow.

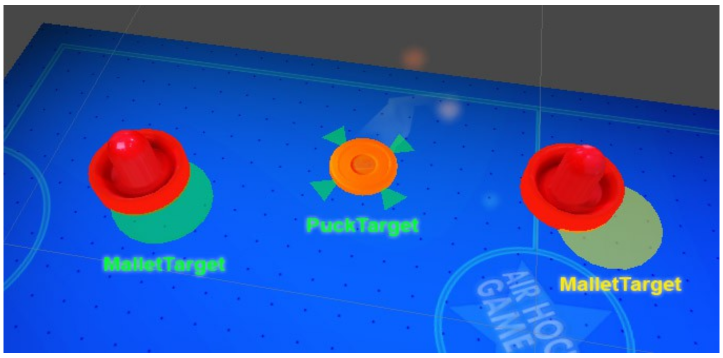
The CPU script uses **PuckTarget** to follow the puck, whereas the player script uses **MalletTarget** to compute mallet input position.

Target objects

In each **zone**, beside the **mallet** gameObject, you'll find two target dummy objects : **MalletTarget** and **PuckTarget**.

These objects are used to reference mallet and puck positions instead of directly using them. This is useful for debugging and tweaking.

Target objects are displayed in **Debug Layer**.

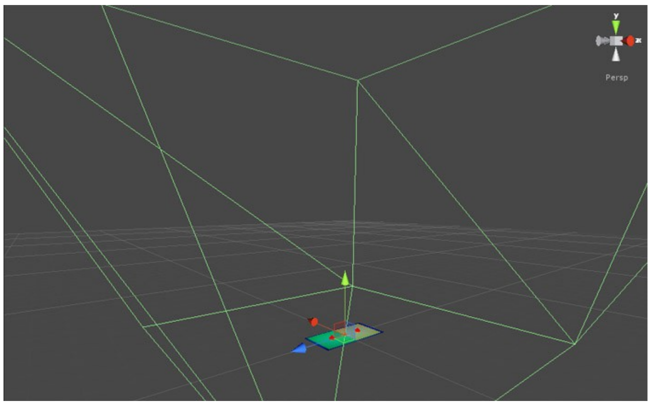


Bounds collider

The gameObject **ColliderBounds** contains the collider wich defines the playing surface's boundaries.

Input Raycast collider

The gameObject **ColliderInputRaycast** contains the collider which receives the mouse/touch inputs. It is assigned to the **InputRaycast Layer**.



Puck

Attached to the **Puck** gameObject, **PuckScript** controls the rigidbody's velocity limit and prevents the object to go out of the playing surface.

Goals

Each goal is controlled by is **GoalScript** and contains a **GoalAnimator** children object.

GoalAnimator object is a plane with a fading texture animation attached to it.

GoalScript simply triggers collision between puck and goal's box collider, then send the goal event to **MainScript**. It also plays **GoalAnimator** fading texture animation.

Additional notes

Included in the Main script, you'll find an example of gameplay presets, displayed on screen by Unity native "OnGUI" UI. This is aimed to give an example on how to customize the gameplay. **Debug GUI** is disabled by default - you can enable it by checking '**Show Debug GUI**' in **MainScript** inspector.

Also, the "DIFFICULTY" option button show you a quick implementation of a game option, and actually just change the mallet cpu script's speed.

The assets are arranged in layers for convenience. All the debug assets are linked to **Debug Layer**. The UI assets are linked to **UI Layer**.

Zones debug elements, including raycasts are differentiated by color. **Zone1** elements are displayed in green, **Zone2** ones in yellow.

You should find all the remaining needed informations in the scripts, where nearly each line is commented.

Support

If you have any question you can send an e-mail at support@eti-software.com

Other products

ETI on Unity Asset Store : http://bit.ly/2yFzKAq

Follow ETI

ETI on facebook : http://www.facebook.com/ETISoftware
ETI on Twitter : http://twitter.com/ETISoftware