About

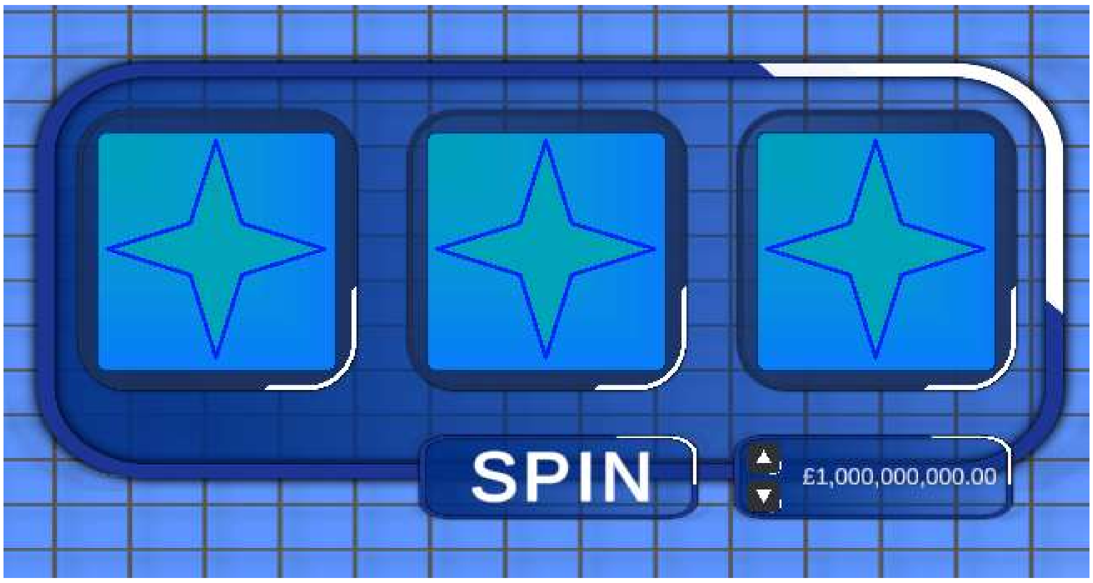
The Tegridy Bandit controller provides a simple solution to implement your own pot and odd based multi wheeled bandits / fruit machines with simple configuration and adaption.

The main component is the TegridyBanditController class which handles all computations, you pass it a *wheels* and *prizes* class when the StartUp function is called, then call Spin(*stake*) with your players stake and it will return the winnings as a *float* along with the position / image each wheel should end on to replicate the results for the user. The system takes a user defined house cut on each spin and then adds the remainder to the prize pots for each prize. Wins are only presented when the pot has sufficient funds to payout and a spin record is also kept if required.

The Controller comes with an example of a 2d and 3d implementation that be can be used as a base for your own systems that build upon the controller.

Usage

The package comes with two example scenes and scripts for both a 2D GUI and a 3D model based system that demonstrate how the TegridyBanditController can easily be utilised to give reliable results with simple integration into your projects.



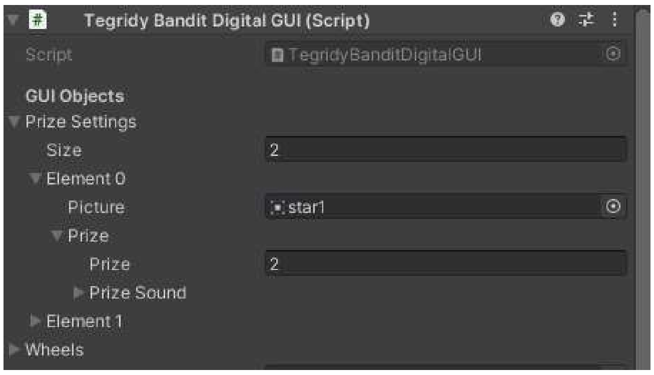
Support

support@tegridygames.co.uk

Configuring 2D Scene

- Create an empty GameObject in your scene and add the TegridyBanditDigital component to the object.
- Add any default audio you would like use, this is used when no audio clips have been set on the GUI. These can be left null if no audio is required.
- You will require at least two images for your prize symbols imported into the scene to configure the GUI Later.
- You will want to build your desired GUI in the editor then add 4 buttons to spin the machine, change the stake and close down plus an ImageObject for each of the slots you want on the machine. Also add TextMeshPro text object for the stake display Additional fields can be completed for debug output
- Add the TegridyBanditDigitalGUI to the GUI you just created and configure the component as described below.
- Once you have all your GUI's configured, set the size of *machines* on the TegridyBanditDigital component and for each element drag your GUI's into the *gui* variable.

Configuring 2D GUI Example



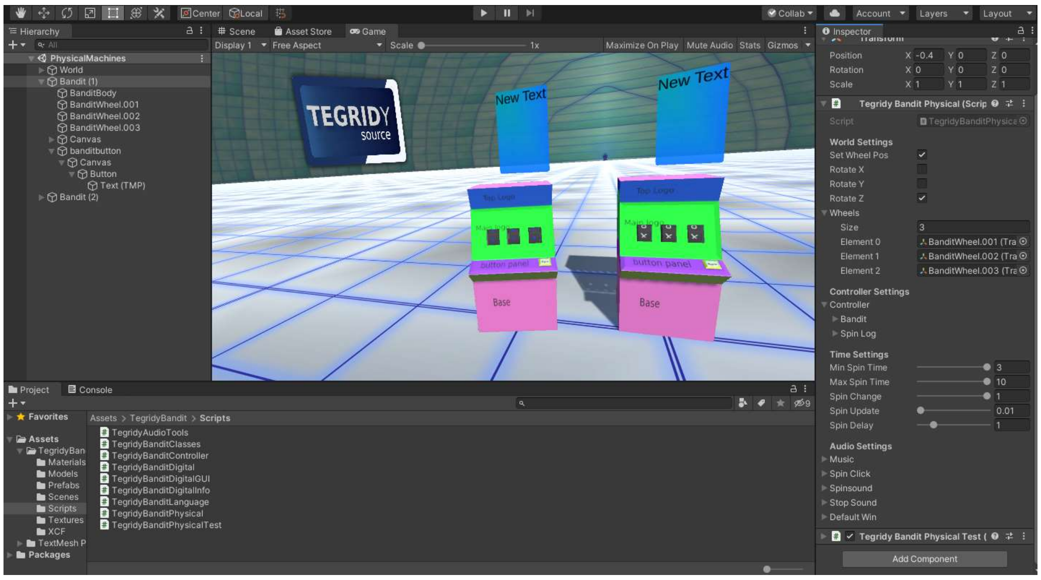
Tegridy Bandit Digital GUI (Script)
Script: TegridyBanditDigitalGUI
GUI Objects
Prize Settings
Size: 2
Element 0
Picture: star1
Prize
Prize: 2
Prize Sound
Element 1
Wheels

GUI Objects

- **Prize Settings -** Set the size to match the Prize Symbols you imported earlier, then for each element set the *picture* variable to the desired image and *prize* to what the players stake should be multiplied by if they win. If this prize should have its own audio add the clips to *prizeSound*
- **Wheels -** Add the ImageObjects you created in the scene earlier here these will be changed as the machine "spins".
- **Other UI** - You will also want to configure the values for *close, stake, spin, stakeUp, stakeDown* along with the increments you would like the stake modified.
- **BanditSettings** – HouseRake is the amount the house will take out of the pot for each spin, and the win chance is the chance of winning if a prize pot has enough to pay out.
- **AudioSettings** – If these values are left empty the default audio set in the controller will be used.

- **TimeSettings** – Min/Max ImageSwaps is used to decide how many times the machine should spin the wheels and this will be a random number between the two values, *swap delay* is the time between swaps and *spinDelay* is the time between the spin ending and then being able to spin again.

Configuring 3D Scene



```
PhysicalMachines
  World
  Bandit (1)
    BanditBody
    BanditWheel.001
    BanditWheel.002
    BanditWheel.003
    Canvas
    banditbutton
      Canvas
        Button
          Text (TMP)
  Bandit (2)
```

```
Favorites
Assets
  TegridyBan
    Materials
    Models
    Prefabs
    Scenes
    Scripts
    Textures
    XCF
  TextMesh P
Packages
```

Assets > TegridyBandit > Scripts

```
TegridyAudioTools
TegridyBanditClasses
TegridyBanditController
TegridyBanditDigital
TegridyBanditDigitalGUI
TegridyBanditDigitalInfo
TegridyBanditLanguage
TegridyBanditPhysical
TegridyBanditPhysicalTest
```

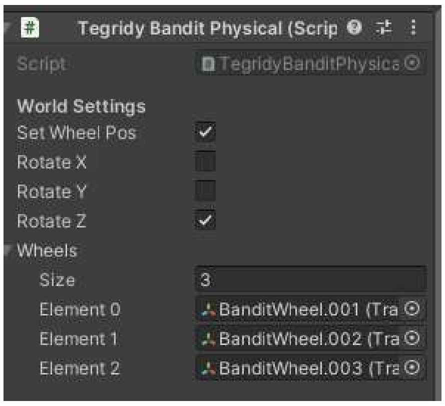
```
Transform
Position    X -0.4   Y 0   Z 0
Rotation    X 0      Y 0   Z 0
Scale       X 1      Y 1   Z 1

Tegridy Bandit Physical (Scrip
Script              TegridyBanditPhysica
World Settings
Set Wheel Pos       ✓
Rotate X
Rotate Y
Rotate Z            ✓
Wheels
  Size              3
  Element 0         BanditWheel.001 (Tra
  Element 1         BanditWheel.002 (Tra
  Element 2         BanditWheel.003 (Tra
Controller Settings
Controller
  Bandit
  Spin Log
Time Settings
Min Spin Time       3
Max Spin Time       10
Spin Change         1
Spin Update         0.01
Spin Delay          1
Audio Settings
Music
Spin Click
Spinsound
Stop Sound
Default Win

✓ Tegridy Bandit Physical Test (
```

- For this scene you will require 3D models that will represent the wheels of the machine, once you have these add them to your scene and then add the TegridyBanditPhysical component. These models should be skinned with your prize symbols and can contain different amounts of symbols on each wheel.

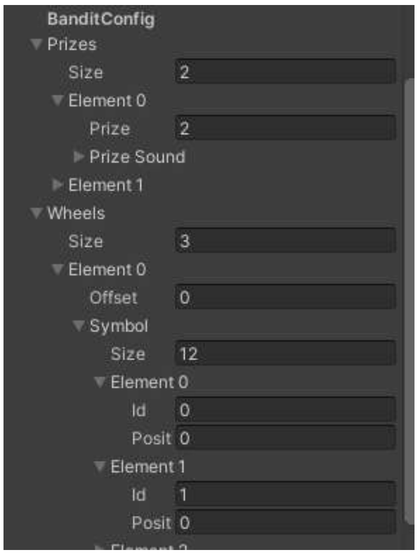
TegridyBanditPhysical Configuration



```
Tegridy Bandit Physical (Scrip
Script              TegridyBanditPhysica
World Settings
Set Wheel Pos       ✓
Rotate X
Rotate Y
Rotate Z            ✓
Wheels
  Size              3
  Element 0         BanditWheel.001 (Tra
  Element 1         BanditWheel.002 (Tra
  Element 2         BanditWheel.003 (Tra
```

- World Settings – If *SetWheelPos* is true the program will presume your wheel models have equal spacing for the prize symbols and will work out the rotations based of the offset for the wheel automatically, if not set you will need to configure the rotations for each wheel manually. Rotation will happen on any axis set to true.

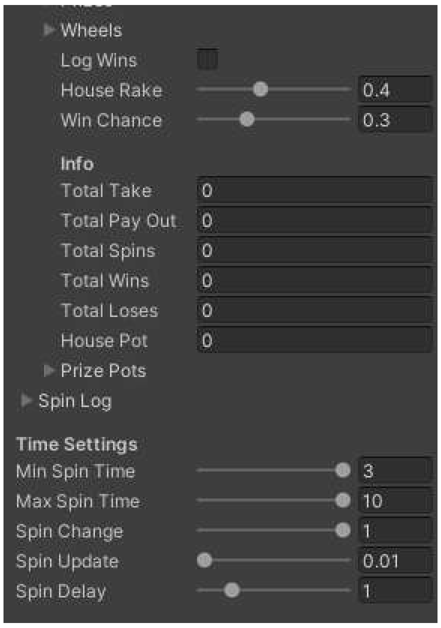
- Wheels – Add the transforms of your wheel objects here.



```
BanditConfig
▼ Prizes
    Size            2
  ▼ Element 0
      Prize         2
    ▶ Prize Sound
  ▶ Element 1
▼ Wheels
    Size            3
  ▼ Element 0
      Offset        0
    ▼ Symbol
        Size        12
      ▼ Element 0
          Id        0
          Posit     0
      ▼ Element 1
          Id        1
          Posit     0
```

•

- Prizes - The size should be set to the number of unique prizes that are on the wheel. If a prize is won the payers stake will be multiplied by *prize.* If this prize should have a unique sound it can be set in *prizeSound.*
- Wheels – *Offset* should be set to the rotation your first prize lines up on, *symbols* should be set to the amount of symbols on that wheel. Each symbols *ID* should be set to the index of the *Prize* that it matches. If you have set *setWheelPos* to true above you can leave the position field as default, if not you will need to enter the rotation for each symbol on all your prize wheels.



```
  ▶ Wheels
    Log Wins        ☐
    House Rake      0.4
    Win Chance      0.3

    Info
    Total Take      0
    Total Pay Out   0
    Total Spins     0
    Total Wins      0
    Total Loses     0
    House Pot       0
  ▶ Prize Pots
▶ Spin Log

Time Settings
Min Spin Time       3
Max Spin Time       10
Spin Change         1
Spin Update         0.01
Spin Delay          1
```

- If you’d like to keep a full record of each spin you will need to set *logWins* to true, *houseRake* defines the amount the house will keep from each spin and *winChance* is the chance of a payout if the one of the prize pots is full.

- TimeSettings – *Min/MaxSpinTime* is the time that the machine should spin for calculated from a random number between the two settings. *spinChange* is the amount in degrees the rotation should move when the machine is updated, *spinUpdate* is the time between updating the wheels rotation and *spinDelay* is the time between pressing the button and spinning the machine again.

Usage - TegridyBanditController Class

- **BanditConfig(*bandit*)**
  This should be called before using any other functions, and when you would like to reset the machine to its starting values. When called it also rebuilds the prizes rows so a new wheel configuration can be applied.
- **SpinWheels(*stake*)**
  Once BanditConfig has been called pass this function a float representing the players stake and it will return a *BanditResults* class that can be used to display the results. If logging is enabled the results will be added to *spinLog.*
- **Bandit Class**
  Provides the settings required for the main system to operate and requires that the *prizes* and *wheels* arrays have been defined. The machines house rake, and chance of winning are defined in *houseRake* & *winChance* it also provides info on the machines takings, payouts and pot totals.
- **BanditResults Class**
  Contains 4 variables, a bool *winner* represents if this spin was a winning one. A float *winnings*, to be added to the players money. Integer that provides an index to the prize that was won called *prizeID* and also an array of integers that represents the bandits symbols.