Lucky Wheel
(Physics Based)

Overview

Create customizable lucky wheels for your mobile game with this and let the user win some fantastic prices.

Key Features:

Fully Configurable Wheel:
Configure the spin force, angular drag, spin time and many more options for your wheel.

Customizable Reward Sections:
Easily create your customized wheel for as many rewards as you like

Smooth 2D Physics Based Animation:
Experience realistic physics driven spinning mechanics with adjustable spin force, angular drag and more.

Ready to use out of the box:
Just open one of the demo Scene and click on spin.No need for

Mobile optimized:
No matter which orientation your game is (landscape or portrait mode), this asset comes with demo scenes that fit both variants.

Extensive Online Documentation & Intuitive Integration:
Integrate this asset into your Unity project with user friendly scripts and detailed documentation and make the most of this powerful tool. No need to be a coding expert - this asset is designed for easy implementation.

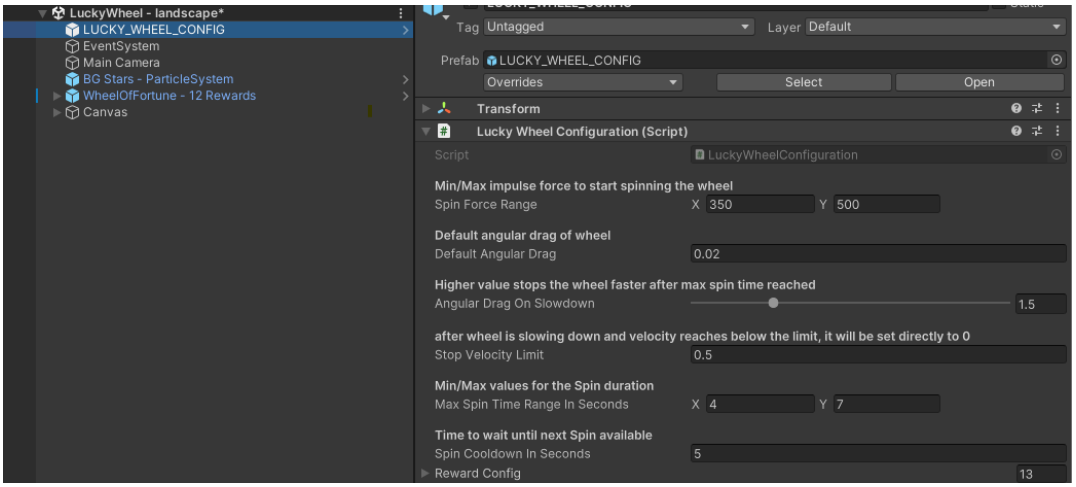
Configuring the Wheel

Demo Scene

There are 2 demo scenes included, one in portrait and one in landscape mode.
Simply start one demo scene and click on the spin button to start the wheel.
Just make sure you have the proper game resolution set in your editor to display it correctly.
(Portrait scene displayed in landscape wont look as it is intended to)

Lucky Wheel Configuration

You can find the centralized wheel configurator as the first GameObject in the scene

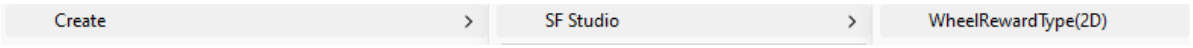


Spin Force	Define a range for the spin force. Spin force is applied to the Rigidbody of the wheel and defines how strongly you rotate the wheel.
Default Angular Drag	This value describes the standard angular drag of the wheel. This can be also considered as wheel friction. The higher the value, the more “friction” the wheel has and therefore will faster come to a stop
Angular Drag On Slowdown	After the configured max spin time (if the wheel hasn't stopped yet due to the default angular drag and spin force) the angular drag will increase so the wheel will come to a halt soon.
Stop Velocity Limit	Define the min velocity of the wheel that is considered as “still spinning”. If after you spin the wheel, the velocity gets below the defined limit, the velocity will be set to zero immediately and the reward claim will be triggered.
Max Spin Time	After the time is reached, the angular drag on slowdown will overwrite the default angular drag, so the wheel will come to a stop faster.
Spin Cooldown	Defines a cooldown that the user needs to wait after spinning the wheel until he can spin it again.
Reward Config	You can define the rewardType and the rewardAmount per entry. When creating your own wheel with your own rewards, please make sure that there are as many rewards configured as there are reward sections on the wheel, otherwise you will get an error on wheel initialization.

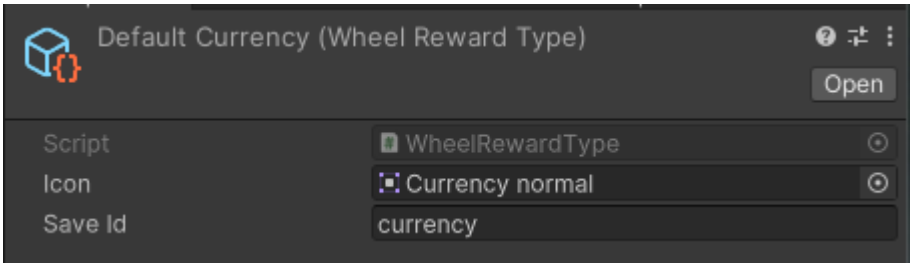
Adding new RewardTypes

This asset comes with a default and premium currency reward type with the option to add an arbitrary amount of new reward types to the wheel.

The reward types are built with scriptable objects. So If you want to create your own reward type, you can do it pretty easy by right clicking somewhere in your project folder and selecting Create → SF Studio → WheelRewardType(2D)

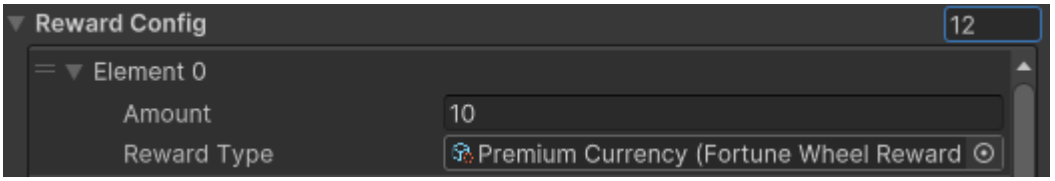


Rename the created scriptable to fit your rewardType name and assign an icon and a saveId to your type:



The saveId is used as an identifier to save the given reward amount to the correct reward, via *PlayerPrefs*. If you want to *change how and where the rewards are saved*, please adapt the *AddReward method in the WheelRewardType* scriptable.
This is located under Scripts → Scriptables:

Then simply go to your Wheel in the demo scene and add (or change) a reward type in the Reward Config list.



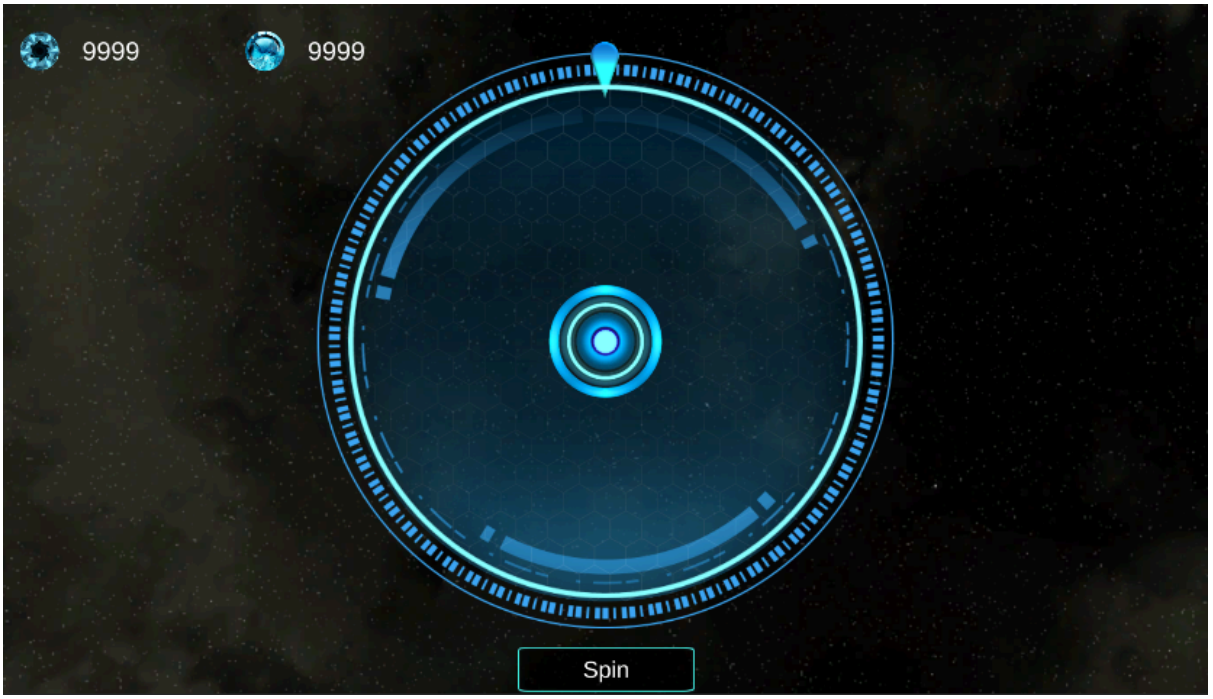
Creating your own wheel

Empty Wheel

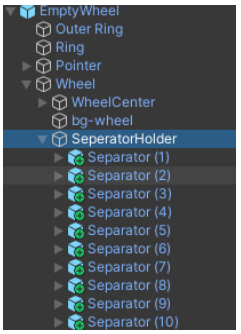
In the prefabs folder you can find a subfolder called *WheelComponents*
Drag following prefabs to your scene:

- EmptyWheel
- LUCKY_WHEEL_CONFIG
- Wheel UI (optional if you want to reuse it, otherwise you can also create your own UI)

Your scene should look something like this

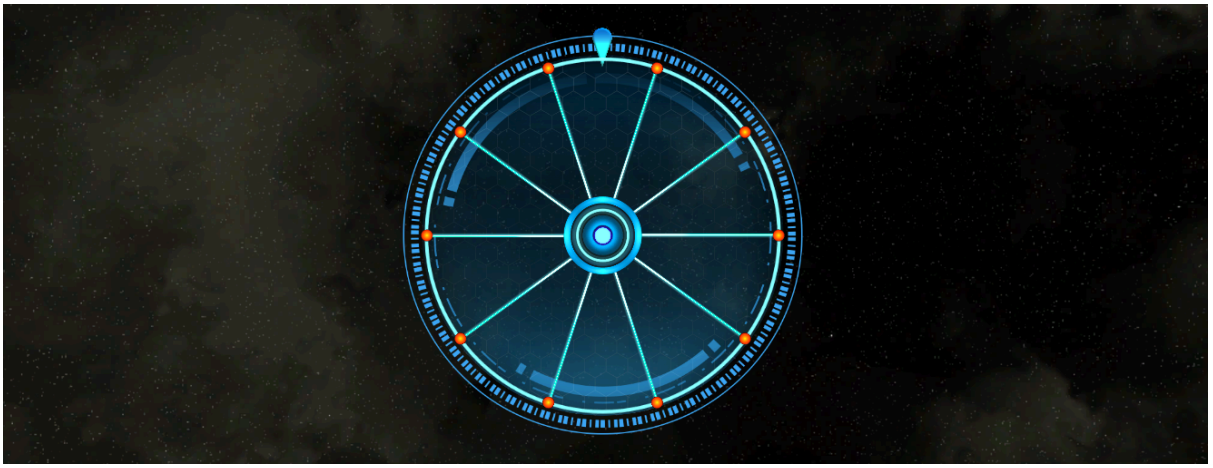


Separators



In the same prefab folder where you found the other wheel components there is a *Separator (1)* prefab.
Depending on how much reward sections you want to create, add X of them to EmptyWheel→Wheel→SeparatorHolder
Then rotate each separator along the Z axis until you get your desired outcome

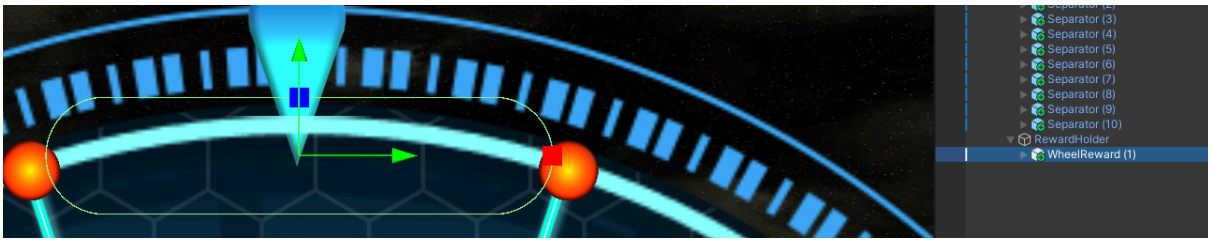
Adding 10 separators, rotating the SeparatorHolderby 18 and rotating each separator by 36 degrees more than the previous one results in following wheel.



Basically what I do is I calculate the rotation by 360 divided by the amount of separators. In our case it equals 36. So I set up the separator rotation like 36, 72, 108 etc.
To offset the separators so no separator is directly under the Pin, i divide the 36 by 2 and use this as the rotation for the SeparatorHolder (which in our case is 18)

Rewards

Now it’s time to add some rewards to it. In the same prefab folder you can find the *WheelReward (1)* prefab drag it to the RewardHolder.
Click on the added object and in scene view there should appear some handlers for the 2d collider

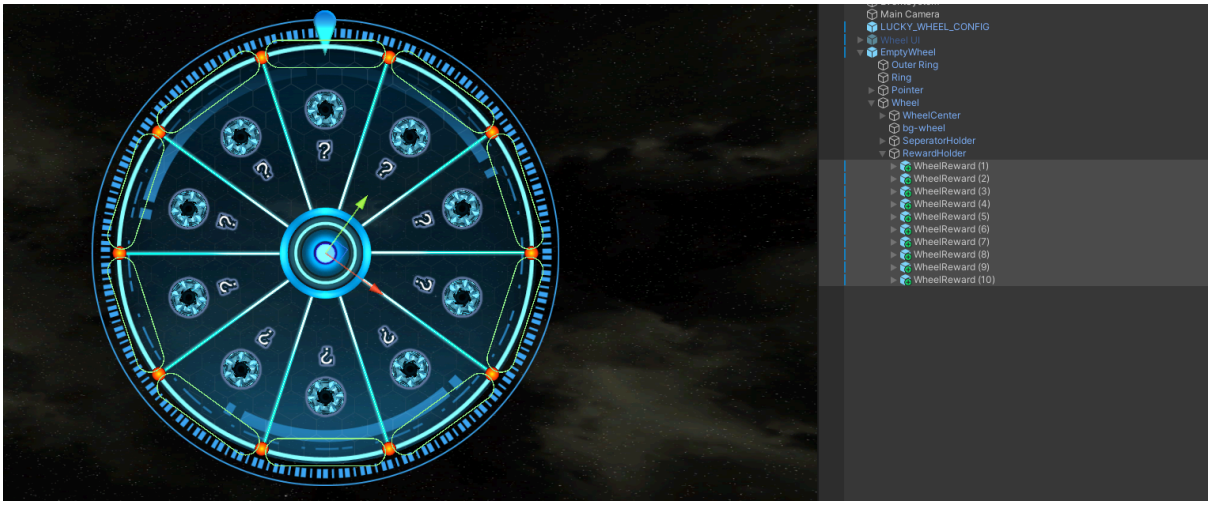


With the green arrows you can move the collider position and with the blue and red cubes you can resize it. Make sure that the collider is between the 2 orange separator knobs and that it is going at least outside of the inner blue circle. So we can make sure that the pin collider detects the proper reward when the wheel stops.

Now duplicate your WheelReward X times. Duplicating in unity will ensure that the next reward number at the end of the name is automatically incremented, so you will have objects ranging from WheelReward (1) until WheelReward (X).

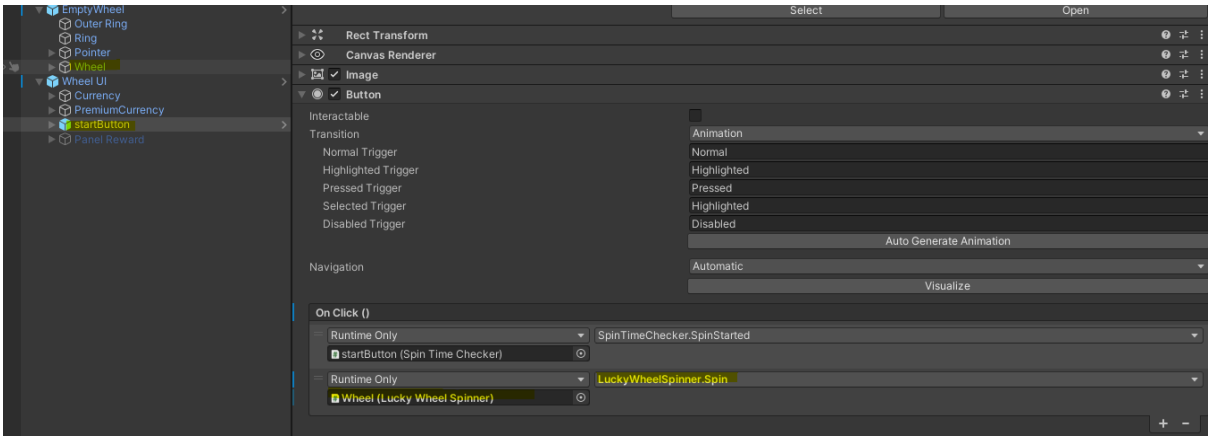
Please do not delete the number at the end of the reward name.
The wheel component initializer will automatically parse the name, detect this number and assign the proper reward amount and icon, configured in your LUCKY_WHEEL_CONFIG.

After setting up all rewards it should look something like that:



Connect Wheel with the UI

Go to the startButton and drag the Wheel component from the empty wheel to the OnClick handler of the button, then select LuckyWheelSpinner.Spin



And that’s it. Congratulations, you have just created your own customized lucky wheel!

I hope you enjoyed this asset and had a smooth integration into your project!

If you liked this asset I would really appreciate it if you could take a few minutes of your time and help me out by leaving a review on the Unity asset store for this package!

Also if you have any questions/complaints/suggestions please contact me under:
gamedevibk@gmail.com