Project 1 - Rube Goldberg Machine

Name: Jack Moyen

- **Repo Link:** https://github.com/IGME-202-2241/202-work-MilkDiver/commit/86d0fdd4c399abfc0c57ae544f0fa3f0d96760ee
- **Build Link:** https://github.com/IGME-202-2241/202-work-MilkDiver/blob/main/builds/Project_01/index.html
- **Assignment Details:** Project 01 - Rube Goldberg Machine

Machine Overview

*Give an overview of the design of your machine. This is *not* a detailed level design. Instead, give us 2-3 sentences describing the overall theme and major machine parts.*

Machine Part Prefabs

Add more subsections as needed, but you must have at least 4 unique machine part prefabs.

Seesaw Wedge

This diamond shaped seesaw element operates from a hinged fulcrum point at its bottom point, which can be adjusted. This machine part allows for left and right pathing decisions, or simply as an interactive decorative element.

Door

When working with bouncy objects, it’s best to keep confined to one specific area. By initially not having collision, then enabling its collision once a specified gameObject meets a specific condition, in this case, having a lower Y value, this machine part can dynamically change the play area.

Conveyor

This machine part applies force to an object while said object is in contact with it. This value can be adjusted based on direction, acceleration, and variance.

Variant - SingleBoostConveyor: Uses SingleBoost Script to give one quick boost of speed in a specified direction to the colliding object.

BouncePad

The generic, bouncy surface needed for fun movement of the ball.

Variant - EvilBouncer: Similar to the BouncePad, except enhances the ball’s velocity.

Rotating Piece

Able to rotate using a custom script which incrementally changes angle, this prefab can be used to simulate a milling or blending motion in a machine. Can be a bit… unpredictable. Subsequent runs using this piece do not follow the same pattern. This is a runtime bug.

Camera System

In an effort to reuse and not worry too much about “following the action”, I decided to have the camera simply follow the ball at all times and have all the “action” centered around the ball.

User Interface

I utilized the Canvas object in Unity’s toolkit and made a custom script to change the Text component of it every frame to the RealTimeSinceStartup.

Scripts

BoingColor

Every time the attached gameObject has a collision, it's color changes from a specified array of colors.

Collision100Leave

After experiencing a specified number of collisions, the object's collision is disabled.

Display Time

Changes a Canvas object's text component to the real time in seconds since startup.

EvilSpring

Allows for a specified bounce force to be inflicted upon a colliding gameObject.

Follow

Allows the attached gameObject to follow another specific gameObject. In this case, the camera is tracking the ball.

Obliterate

Indiscriminately deletes any offending gameObject that dares collide with this gameObject.

On Collision Disable Collision

On collision with another gameObject, this object's collision is disabled.

On Collision Move

On collision with another gameObject, this object's RigidBody state is changed from static to dynamic.

RBRotate

Allows for incremental rotation of an object at a specified amount in degrees and speed.

ShutBehindYou

Takes a specific gameObject, and enables this object's collision when the specified object has a lower Y value than this one.

SingleBoost

Changes the color of and effectively disables an effector object once a collision has occurred.

Release Notes

Overview

Due to a runtime bug being passed off as an intentional feature, the rotating pieces are technically completely random, and obey no one. This means that every run, once the ball makes contact with the rotating pieces, is completely random. There should be no errors, but it's possible that an error may occur as a result of this.

Rubric Notes

Annotate this copy of the rubric with quick notes about where/how your project meets each criterion. Yes, you are effectively pre-grading yourself. More importantly, it demonstrates to us that you understand how the pieces of your project fit together.

Item	Points	Notes
Machine Requirements	**10**	
Machine runs for at least 30 seconds	5	Runs for 30 seconds. May be a bit "slow" at times.
Clearly marked start and end	5	Start is the starting point of the ball, end is clearly marked.
Machine Parts	**16**	
4 unique machine part prefabs (2pts each)	8	There's a bunch of them. I probably have all the points for this one.
2 Effector components used (2pts each)	4	Both the Conveyor and Single Boost Conveyor are technically variants of the same effector, but each serves a different purpose. The conveyor moves objects around, whereas the single boost conveyor is used more as a "brakepad" of sorts.
2 Joint components used (2pts each)	4	The Rotating piece and Seesaw Piece are both hinge pieces. Couldn't figure out how to make motor or wheel Joint work, so I scripted rotation myself.
Scripting Requirements	**26**	
2 machine parts have a script attached (3pts each)	6	Most machine parts have a script attached to them that does something. I had a lot of fun making small, silly scripts.
Script to control an Effector component	5	SingleBoost script, or Collision100Leave
Script to control a Joint component	5	OnCollisionDisableCollision script, or RBRotate
Script to control the Effector implements one of the Physics event methods	5	SingleBoost Script, or Collision100Leave

(OnTriggerEnter2D, etc.)		
Script to control the Joint implements one of the Physics event methods (OnTriggerEnter2D, etc.)	5	OnCollisionDisableCollision
Camera System	**10**	
Camera follows the action of the machine	10	Follow Script
UI	**10**	
Elapsed time displayed on the screen	10	Display Time Script + Canvas Object
Documentation	**28**	
Machine Design	4	:D
Machine Part details	8	:D
Camera System details	8	:D
UI System details	8	:D
Total	**100**	