WARNING!!!
All Rigidbody objects in a scene that runs with the PhysicsPreview script need to have the PreviewedObject script attached!
WARNING!!!

Physics Prediciton

by Leo Traub

Quick Start

1. Import the PhysicsPreview object into the scene:
2. Attach the PreviewedObject script to ANY object in the scene, that has a rigidbody.
3. Adjust all the displayed values in the editor, as you like. (If you enable line rendering, you also have to add a LineRenderer Component to the object)
4. Set all materials of Objects with the PreviewedObject script to a shader, that renders the material with an alpha channel, or else the ghost objects won't be transparent. (If you are using Unity default shader, change Rendering Mode "Opaque" to "Fade" or "Transparent")
5. Call one or some of the following functions in any script, to enable the physics prediction:
    - PhysicsPreview.Preview(): New ghost frames are rendered
    - PhysicsPreview.RemoveAllGhosts(): All ghost objects in the scene are removed. It is recommended to call this function always right before you call PhysicsPreview.Preview().
    - PhysicsPreview.FreezePhysics(): All objects with a rigidbody are freezed, but the velocities are kept.
    - PhysicsPreview.StartPhysics(): All objects with a rigidbody start to move again
6. If the scene changes to another scene, set Physics.autosimulation = true; so that the physics work like usual again.

Notes

- Keep in mind, that the Pysics Prediction can be a quite **performance expensive** feature, so try to adjust all values to the best fps level, you can reach.
-**Collision Tests** can be done like usually, but make sure to write the following line at the begin of every OnTriggerEnter/OnCollisionEnter, etc:

```
if (PhysicsPreview.timeIsFreezed) { return; }
```

- This asset uses the default Physics System of Unity. It works with any kind of Colliders and Physics Materials. Manipulating a rigidbodies velocity via script will also work.
- It has only been tested with Unity default shaders, that support alpha channels. It will only work with MeshRenderer Components (no SpriteRenderer) and with a MeshFilter Component, that has a mesh assigned.
- Any data garbage that is created, which can be quite a lot, is removed frequently via System.GC.Collect(), so only remove the IEnumerator cleanGarbage() in PhysicsPreview, if you know what you are doing.
- Children of objects with the PreviewedObject script, will not be rendered as ghosts, but feel free, to create custom ghost objects, that can be handled with your particular children.
- If you want a player object to move inside the scene, while the time is freezed, you will have to model the movement without a rigidbody and use direct transformations on the players transform.
- You can use LineRenderers in any way you want, so feel free to set the width or color of your lines and of course you can use the included trail material in the ExampleScene folder.

Adjustable Values Description

PhysicsPreview:

-Iterations: How many ghosts frames are drawn? (Don't set this value too high)

-TimeStep: How long is the time between the rendered ghost frames? (Don't set this value to zero)

-InBetweenFrameSteps: How many times are the Physics rendered in between the time steps. Set this value high if your objects fly slow. Setting this value too high might lower your fps rate. (Don't set this value to zero)

Note: The value (TimeStep/InBetweenFrames) corresponds to the Physics time step, as it determines at what rate the Physics will be refreshed. Therefore you can set these two values to act exactly like the default physics. But setting the timestep higher will of course enhance your performance.

PreviewedObject:

-GhostObject: It is recommended to use the default GhostObject Prefab, but of course you can modify it or create new ones with attached ParticleSystems or whatever. If changing the GhostObject constantly increases your RAM usage, there is some garbage created by your custom GhostObject, so you should either not use it or find out, what reference creates that garbage.

-drawLine: Is there a line renderer atatched to the object? / Should the line be drawn?

-drawGhosts: Should the object draw GhostObjects on PhysicsPreview.Preview()?

-startSpeed: What ist the objects start velocity? (Set this to zero if you dont want to assign a start velocity)

Example Scene

In the example scene you can fly around using both the horizontal and vertical Input Axis and you can turn around using your mouse. Player movement is done wihtout a rigidbody, because a rigidbody would not work as long as the time is used Every time you move, the ghost objects will be re-rendered. It is recommended to set some kind of flags in your own scene, so that the ghosts will not be rendered in every frame.

By clicking, you can place one or multiple balls in the scene, that are initialized with a speed factor, you can adjust by scrolling.

Once your are ready and want to make the Physics run again, you can hit space and see the things move right through the ghosts shapes. To make the Physics stop again, hit space again.

The ExamplePlayer script is commented quite well and easy to understand, so have a look at it to see how the player movement can be modeled and how the PhysicsPreview Functions are called.

Have fun with my asset! I am really looking forward to see it in your games! ;D Please give me credit somewhere in your game and let me see your creations, as soon as they are ready!!

For questions and feedback contact me: toastarve@gmail.com