Tutorial 14

- during this tutorial we will
  - understand the PGE API
  - create a small pinball game using PGE
  - understand how PGE integrates with PyGame

Using PGE to create a game

- full documentation about PGE is available ⟨`http://floppsie.comp.glam.ac.uk/Southwales/gaius/pge/homepage.html`⟩

Creating a pinball game using PGE

- start with the breakout game source code ⟨`http://floppsie.comp.glam.ac.uk/Southwales/gaius/pge/example_games.html`⟩ and adapt it
- throw away all the blue boxes and associated callbacks
- see that the modified game still runs to completion

Pinball

- add some extra circles and maybe modify the triangles (smaller possibly)
- add a score so that every time the ball hits a triangle you gain +20 points
  - every time it hits a particular circle it gains +10 points
- every time the points increase by a 100 an extra life should be granted

Pinball

- allow the user to fire a new pinball when the old pinball is lost
  - decrease the life count as a pinball is released

- add a firing shoot for the pinball
  - allow the user to aim the ball with the mouse

Pinball flippers

- PGE is not capable of rotating objects, so you will have to think creatively to implement a way of ‘‘interfering’’ with the pinball
- you could adjust the x movement of the ball slightly if the left/right mouse button was pressed
- or you might introduce a fixed polygon (representing the flipper) as a paddle (aka breakout)
  - the polygon could be deleted and moved upon mouse movement
  - be careful and ensure that the polygon resides along a, y = constant, line
- what simple changes would you make to the API to create a simple flipper?

Conclusion

- we have
  - understood more about the PGE API
  - created a small pinball game using PGE
  - understood how PGE integrates with PyGame

- you might want to read around the subject by reading this paper The Construction of a Predictive Collision 2D Game Engine ⟨`../../../Papers/paper21/ieee.pdf`⟩