ETGG3802 **Lab4: ECS**

1. Get a copy of the start files (component_camera.h, component_camera.cpp, etc.). Add these files to our project.
2. (**5 points**) Get rid of that temporary attachObject method in GameObject (the purpose of this lab is to shift that role to our ECS).
3. (**35 points**) Make the ECS system described in lecture
4. (**15 points**) Add a playAnimation method to MeshComponent that starts an animation (have a looping parameter)
5. (**5 points**) Add an update method to the GOM that updates all game objects (which in turn updates all their components)
6. (**20 points**) Replace all Entity's, Cameras, and Lights from the main program. Replace them with calls to create a component. Make sure if you didn't get the animation working in Lab3, do it now (using our component). Don't store the component* in Application (so you're forced to test the getComponent methods of game object to get it in frameStarted).
7. (**5 points**) Add a method to the Application that takes a camera component and makes that the active camera. Make the Application a friend to it can access the protected Ogre::Camera* inside the component.

[This part is in preparation for the next lab, but I want it done here so we're ready to hit the ground running]

8. (**15 points**) Find and build development dependencies for an xml parser (using cmake in pugixml's case), or by making a dll project manually (in tinyxml2's case). Two I've used in the past are pugixml and tinyxml2. Make sure they follow ogre's pattern where you have a folder to put in the dependencies folder and link to it in the project properties.
9. (**20 points**) Get the sample scene I linked on blackboard, add a folder for it in resources.cfg and then…(the real work)…make your GOM recursively visit (in a depth-first traversal fashion), each node in the xml file. You don't have to do anything with that data in this lab…just log the tag you're visiting.