CS 4730: Computer Game Design
Lab 3: Display Tree

Overview

This week, you will be updating your engine to support a display tree. You will do this by implementing the new DisplayObjectContainer class and slightly modifying your other code as well.

PART 1: Create the DisplayObjectContainer class

Begin by creating a new class called DisplayObjectContainer. This class should extend DisplayObject (Sprite, Game, etc. should now extend DisplayObjectContainer instead).

Your class should have a single field, a list of DisplayObject objects that are the children of this container. Thus, containers are essentially nodes in a tree and this list is the list of child objects (which might be DisplayObject type which represents a leaf node OR DisplayObjectContainer types representing another internal node).

PART 2: Functionality

Your DisplayObjectContainer needs to support the following functionality:

- Your ***DisplayObject*** class should now contain a private reference to the ***parent DisplayObject***. Notice that this makes sense to put in *DisplayObject* (not container) because even objects that are not containers themselves may be in the display tree as leaf nodes and thus need to potentially have a parent.
- Your ***DisplayObjectContainer*** class should contain a ***list of DisplayObject objects*** that are the children of this container.
- It should be possible for a ***DisplayObject*** (and/or a DOC) to *not have an image* and thus not be drawn to the screen. If this is the case though, the children should still be drawn. Only if the DOC is marked as "invisible" will the children also not be drawn.
- Children (Display Objects) can be ***added and removed*** from this container. You might want to provide several add and remove methods (*addChild, addChildAtIndex, removeChild, removeByIndex, removeAll, etc.*).
- Make sure there is a method for removing all children at once.
- A ***contains(DO)***method returning true iff a display object is already a child of this container.
- Methods for getting a child (returns the DO itself) by ***string id*** or by ***index***.
- update() and draw() methods overridden. This is a bit tricky because you will need to draw yourself, then draw your children one at a time while making sure your transformations are applied to your children. See the slides for a reference for the draw method (update is similar).
- A getChildren() getter that returns all of this object's children.

Test Demo: Solar System Simulator

Create a simple solar system that shows off your engine's new features:

- Make a sun sprite that is the root node of the whole system.
- Planets should be children of the sun.

- Moons of planets should be children of the planets
- The whole system should move continuously.
- The planets and moons should rotate around their parent at different rates. Think carefully about how to do this.
- Planets and moons should have elliptical orbits. Again, try to think of an easy way to accomplish this in code.

Your demo should also support the following commands:

- Q&W: Zooms in and out towards center of sun OR center of current screen position.
- Arrows: Pans up/down/left/right around the system.
- A&S: Rotates the system around the sun's center point.

Turn In

You will submit your working code as a zip file on Collab.