SWEN20003 Object Oriented Software Development

Workshop 4 – Week 5

This week there will be no tutorial. If there are any past lab exercises you haven't finished, finish those before moving onto this week's exercises.

Credit to Kerri Coombs, Living Midnight Design, for the balloon sprite.

Lab Exercises

a) Create a project for this workshop. We're going to set up the Slick library for game development, which uses LWJGL for its graphics and input. These instructions are for Windows systems.
   i. Download the appropriate libraries for your system from the LMS (either 32 or 64-bit). Extract the archive, and store it in a folder somewhere on your computer. I'll call it **SLICK_HOME**.
   NOTE: On the lab computers, put this in your My Documents folder, otherwise you will have issues with permissions.
   ii. Create a new folder named **lib** in your project. To import the libraries, right click the folder and press **Import**. Select **General -> File System** from the dialog, and browse to your **SLICK_HOME\mulit_platform_slick_libraries** folder. From here, select all of the **.jar** and **.dll** files, and press **Finish**.
   iii. Right click each of the **.jar** files, and press **Build Path -> Add to build path**.
   iv. Right click **lwjgl.jar** from the **Referenced Libraries** folder of your project. Press **Properties**, then **Native Library -> External Folder…**. Navigate to your **SLICK_HOME\mulit_platform_slick_libraries** folder, and press **OK**.

b) Next, we'll look at a sample Slick program.
   i. Download **SimpleSlickGame.java** from the LMS, and put it in the **src** directory of your project.
   ii. Create a folder named **assets** in your project. Download and import **background.jpg** from the LMS to this **assets** folder.
   iii. Run the program and experiment with it. Try the up and down arrow keys to see what they do.
   iv. Read through the code, and have a look at how it works. Slick documentation is available from http://slick.ninjacave.com/. Focus on the **Image** and **Input** classes.
   v. Modify the program to change to a different background image when the C key is pressed, and to exit when Esc is pressed.

c) Finally, let's write a simple game using Slick. Create a new class named **SlickFlyingGame**, and select **public static void main** from the method stubs option. Download the plane, map, and balloon images from the LMS (or you can use your own). Use **SimpleSlickGame** as a basis for your program. Here are the specifications for the game:

i. The map should be a background. When the game starts, the plane should be at the centre. A balloon should be placed randomly on the screen.
ii. The player should be able to move left, right, up, and down, using the respective arrow keys. **Note:** the parameter **delta** in **update** is the number of frames between updates. The amount you move should be proportional to **delta** e.g. x += delta * SPEED; Try different values for SPEED, starting at 0.4f.
iii. The plane should not be able to move off the screen. It should be able to “slide” along the edge of the screen where possible.
iv. When the plane is within 50 pixels of the balloon, it should move to another random place on the screen. Hint: try Pythagoras to figure out the distance between objects.
v. The game should exit when Escape is pressed.

For extension, keep track of the number of times the plane has hit the balloon, and draw this on the screen. You could also try implementing rotation of the image, and movement in the direction of rotation. This is slightly more realistic for a plane.