Added the final modifier to the update and render methods in the Game class.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*Name: SonarBat

\*\*\*Version: 1.0(Alpha)

\*\*\*Description: SonarBat a young bat decides to take on the fox

\*\*\* in order to save the world. To do this he must defeat other

\*\*\* bats who have been programmed to take over the world. Sonar's

\*\*\* main weapon is his sonarWave that creates a projectile of

\*\*\* sound to defeat his enemies similar to the sound a bat emits

\*\*\* to track an insect.

\*\*\*Features: Single Level, State switching, Mob, Weapons,

\*\*\* and Energy.

\*\*\*Bugs: Keys might be a bit too sensitive.

\*\*\*Programmed by: Eric Beecroft

\*\*\*Date: 4/17/2015

\*\*\*Skills: C++, and Ruby on Rails.

\*\*\*Learning: Java

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**package** sonar;

**import** java.awt.Canvas;

**import** java.awt.Graphics;

**import** java.awt.image.BufferStrategy;

**import** javax.swing.JFrame;

**import** sonar.gamestates.GSM;

**public** **class** Game **extends** Canvas

{

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**private** **static** **short**[] *gameWindow*;

**private** **static** JFrame *frame*;

**private** **static** **boolean** *running*;

**private** **static** GSM *gsm*;

**private** Game(){} //Prevents multiple games from being created

**final** **static** Game createGame(**final** **short** width, **final** **short** height)

{

*gameWindow* = **new** **short**[2];

*gameWindow*[0] = width;

*gameWindow*[1] = height;

*frame* = **new** JFrame();

Game game = **new** Game();

**return** game;

}

**final** **void** start(**final** Game game)

{

**if**(*running*) **return**;

*running* = **true**;

*gsm* = createGSM(game);

run("SonarBat");

}

**private** **final** **void** run(String title)

{

//Initialize the variables necessary for the gameWorld

**double** delta = 0;

**byte** updates = 0;

**short** frames = 0;

**byte** ticks = 60;

**double** ns = 1000000000 / ticks;

BufferStrategy bs = **null**;

**long** renderTime = System.*currentTimeMillis*();

**long** updateTime = System.*nanoTime*();

requestFocus();

**do**

{

//Perform the gameWorld world functions

**long** now = System.*nanoTime*();

delta += (now - updateTime) / ns;

updateTime = now;

**if**(delta >= 1)

{

update();

delta--;

updates++;

}

render(bs);

frames++;

//Display the game's title every 1 second.

**if**(System.*currentTimeMillis*() - renderTime >= 1000)

{

*frame*.setTitle(title + " | ups: " + updates + " fps: " + frames);

updates = 0;

frames = 0;

renderTime += 1000;

}

**if**(bs == **null**) createBufferStrategy(2);

bs = getBufferStrategy();

}

**while**(*running*);

}

**private** **final** **void** update()

{

*gsm*.update();

}

**private** **final** **void** render(**final** BufferStrategy bs)

{

**if**(bs == **null**) **return**;

Graphics g = bs.getDrawGraphics();

*gsm*.render(g);

g.dispose();

bs.show();

}

//Observers

**final** **static** JFrame getFrame(){**return** *frame*;}

**public** **final** **static** **short** getWindowWidth(){**return** *gameWindow*[0];}

**public** **final** **static** **short** getWindowHeight(){**return** *gameWindow*[1];}

}