**Added the x for loop and also the getTile method to the drawGameState method.**

**package** sonar.gamestates;

**import** java.awt.Graphics;

**import** sonar.gamestates.states.levels.Screen;

**import** sonar.gamestates.states.levels.stages.entities.SpriteManager;

**import** sonar.gamestates.states.levels.stages.entities.animations.tiles.TileManager;

**public** **abstract** **class** GameState

{

//The base class Template for all the gamestates in the game.

**private** StateBuilder buildState;

**private** GSM gsm;

**private** Keyboard key;

**private** SpriteManager smanage;

**private** TileManager tmanage;

**protected** GameState(StateBuilder buildState, GSM gsm)

{

**this**.buildState = buildState;

**this**.gsm = gsm;

**if**(buildState.stateType().equals("Single"))

{

key = **new** Keyboard(gsm);

smanage = **new** SpriteManager();

tmanage = **new** TileManager(smanage);

}

}

**protected** **void** update()

{

}

**protected** **void** render(**int** xScroll, **int** yScroll, Screen screen, Graphics g)

{

screen.setOffset(xScroll, yScroll);

**int** x0 = xScroll / lm.getTmanager().voidTile.getWidth(); //divided by 16

**int** x1 = (xScroll + screen.getWidth() + lm.getTmanager().voidTile.getWidth()) / lm.getTmanager().voidTile.getWidth();

**int** y0 = yScroll / lm.getTmanager().voidTile.getHeight();

**int** y1 = (yScroll + screen.getHeight() + lm.getTmanager().voidTile.getHeight()) / lm.getTmanager().voidTile.getHeight();

drawGameState(x0, x1, y0, y1, screen);

}

**private** **void** drawGameState(**int** x0, **int** x1, **int** y0, **int** y1, Screen screen)

{

**for**(**int** y = y0; y < y1; y++)

{

**for**(**int** x = x0; x < x1; x++)

{

getTile(x, y).render(x, y, screen);

}

}

}

StateBuilder getBuildState(){**return** buildState;}

**protected** GSM getGsm(){**return** gsm;}

**protected** Keyboard getKey(){**return** key;}

**protected** **void** resetKeyboard(){key = **null**;}

**protected** **void** initKey(){key = **new** Keyboard(gsm);}

**protected** SpriteManager getSmanage(){**return** smanage;}

**protected** TileManager getTmanage(){**return** tmanage;}

}