**Grain Quest**

**By Alex Wilber**

**This is the controller where the code is run. It also contains methods that switch the panel and a method to update the information. This class links all of the panels together so that they can be added to the frame. This class also acts as the Frame and contains a method that sets the size of the frame and makes the frame visible.**

import javax.swing.JFrame;

import javax.swing.JPanel;

public class KingdomController extends JFrame

{

//Defines objects of all of the panels.

private IntroPanel introPanel;

private BuySellPanel buySellPanel;

private CharacterCreatePanel characterCreatePanel;

private HarvestPanel harvestPanel;

private GranaryPanel granaryPanel;

private TaxPanel taxPanel;

private MapPanel mapPanel;

private InvestmentPanel investmentPanel;

private SerfPanel serfPanel;

public static void main(String [] args)

{

//runs constructor method

new KingdomController();

}

//constructor method, creates the objects of the panels and adds the intro panel to the frame.

public KingdomController()

{

introPanel = new IntroPanel(this);

characterCreatePanel = new CharacterCreatePanel(this);

buySellPanel = new BuySellPanel(this);

harvestPanel = new HarvestPanel(this);

granaryPanel = new GranaryPanel(this);

taxPanel = new TaxPanel(this);

mapPanel = new MapPanel(this);

serfPanel = new SerfPanel(this);

investmentPanel = new InvestmentPanel(this);

add(introPanel);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

setTitle("Grain Quest");

setSize(615,555);

}

//Starts the game by removing the intro panel and adding the character creation panel.

public void startGame()

{

remove(introPanel);

add(characterCreatePanel);

//refreshes the ContentPane

getContentPane().invalidate();

getContentPane().validate();

}

//Starts the harvest phase by removing the character creation panel and adding the harvest panel.

public void havestPhase()

{

remove(characterCreatePanel);

characterCreatePanel.setVisible(false);

add(harvestPanel);

harvestPanel.setVisible(true);

//refreshes the ContentPane

getContentPane().invalidate();

getContentPane().validate();

}

//Starts Buy/Sell Phase buy adding the BuySell Panel and removing the Harvest panel

public void buySellPhase()

{

remove(harvestPanel);

harvestPanel.setVisible(false);

add(buySellPanel);

buySellPanel.setVisible(true);

//refreshes the ContentPane

getContentPane().invalidate();

getContentPane().validate();

}

public void sellGrainPhase()

{

remove(buySellPanel);

buySellPanel.setVisible(false);

add(granaryPanel);

granaryPanel.setVisible(true);

//refreshes the ContentPane

getContentPane().invalidate();

getContentPane().validate();

}

public void serfReport()

{

remove(granaryPanel);

granaryPanel.setVisible(false);

add(serfPanel);

serfPanel.setVisible(true);

//refreshes the ContentPane

getContentPane().invalidate();

getContentPane().validate();

}

public void taxPhase()

{

remove(serfPanel);

serfPanel.setVisible(false);

add(taxPanel);

taxPanel.setVisible(true);

//refreshes the ContentPane

getContentPane().invalidate();

getContentPane().validate();

}

public void mapPhase()

{

remove(taxPanel);

taxPanel.setVisible(false);

add(mapPanel);

mapPanel.setVisible(true);

//refreshes the ContentPane

getContentPane().invalidate();

getContentPane().validate();

}

public void investmentPhase()

{

remove(mapPanel);

mapPanel.setVisible(false);

add(investmentPanel);

investmentPanel.setVisible(true);

//refreshes the ContentPane

getContentPane().invalidate();

getContentPane().validate();

}

public void loopPhase()

{

remove(investmentPanel);

investmentPanel.setVisible(false);

add(harvestPanel);

harvestPanel.setVisible(true);

//refreshes the ContentPane

getContentPane().invalidate();

getContentPane().validate();

}

//Updates information so it can be displayed properly.

public void updateInformation()

{

harvestPanel.updateInformation();

mapPanel.updateInformation();

taxPanel.updateInformation();

investmentPanel.updateInformation();

granaryPanel.updateInformation();

buySellPanel.updateInformation();

serfPanel.updateInformation();

}

}

**The intro panel displays an image as the splash screen and prompts the user with a start button. The purpose of this class is to greet the user with the name of the game and the option to start the game. Once the user clicks the "Start Game!" button, the panels switch from the Intro Panel to the Character Create Panel.**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class IntroPanel extends JPanel implements ActionListener

{

private JButton startButton;

private JLabel backgroundPicture;

private KingdomController Parent;

public IntroPanel(KingdomController parent)

{

setLayout(null);

Parent = parent;

startButton = new JButton("Start!");

backgroundPicture = new JLabel();

backgroundPicture.setIcon(new ImageIcon("intro.png"));

backgroundPicture.setBounds(0,-13,615,555);

startButton.setBounds(360, 420,200,50);

add(startButton);

add(backgroundPicture);

startButton.addActionListener(this);

}

public void actionPerformed(ActionEvent e)

{

if(e.getSource() == startButton)

{

//runs the method that swtiches the panels

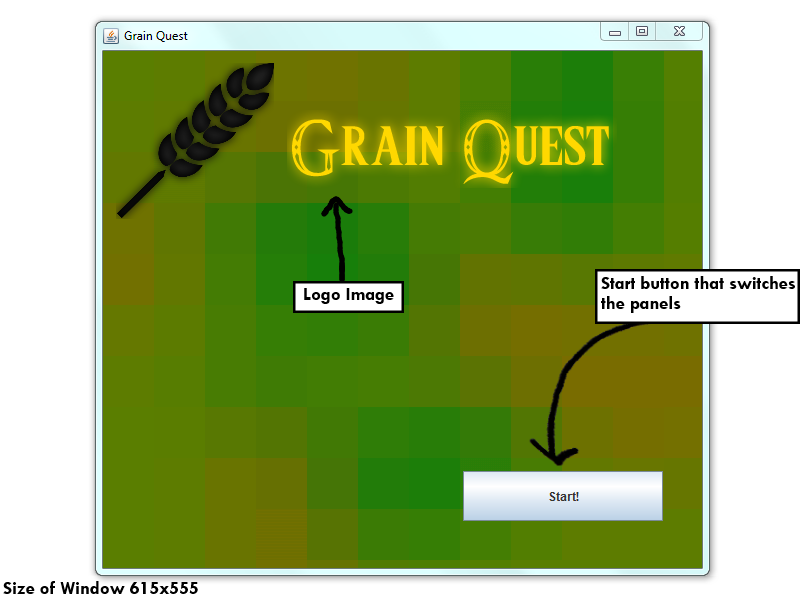
Parent.startGame();

}

}

}

**Intro Panel Diagram**



*The “Start!” button switches to the* ***Intro Panel*** *to the* ***Character Create Panel****.*

**This class displays the character create panel. You are prompted with a text box where you can enter your name, a drop down menu where you can choose our city-state and a radio button option where you can choose "Boy" or "Girl to decide your gender. This information is seen on the playerLabel which can be seen on every panel after this one. Once this information is entered it cannot be changed. There is one button on this panel Labeled "Next" which you click to advance to the next panel.**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class CharacterCreatePanel extends JPanel implements ActionListener

{

private KingdomController Parent;

//Array of Strings for the city state JComboBox

private String[] cStrings = { "Monaco", "Singapore", "Vatican City", "Hong Kong", "Macau" };

private ButtonGroup genderButtons = new ButtonGroup();

private JRadioButton jBoy = new JRadioButton("Boy");

private JRadioButton jGirl = new JRadioButton("Girl");

private JLabel logo = new JLabel();

private JLabel name = new JLabel("Enter your rulers name:");

private JLabel cState = new JLabel("Choose your City-State:");

private JLabel genderLabel = new JLabel("Choose your Gender:");

private JButton nextButton = new JButton("Next");

private JTextField ruler = new JTextField();

private JComboBox cStateList = new JComboBox(cStrings);

public CharacterCreatePanel(KingdomController parent)

{

setLayout(null);

Parent = parent;

logo.setIcon(new ImageIcon("logo.png"));

genderLabel.setBounds(25,325,200,25);

jBoy.setBounds(25,350,200,25);

jGirl.setBounds(25,375,200,25);

cStateList.setBounds(300,225,200,22);

cState.setBounds(300,200,200,20);

logo.setBounds(25,0,600,173);

ruler.setBounds(25,225,200,25);

name.setBounds(25,200,200,25);

nextButton.setBounds(360,420,200,50);

cState.setFont(new Font("Serif", Font.BOLD, 16));

cState.setForeground(Color.BLACK);

name.setFont(new Font("Serif", Font.BOLD, 16));

name.setForeground(Color.BLACK);

genderLabel.setFont(new Font("Serif", Font.BOLD, 16));

genderLabel.setForeground(Color.BLACK);

genderButtons.add(jBoy);

genderButtons.add(jGirl);

add(nextButton);

add(jBoy);

add(jGirl);

add(name);

add(cState);

add(genderLabel);

add(cStateList);

add(logo);

add(ruler);

nextButton.addActionListener(this);

}

public void actionPerformed(ActionEvent e)

{

if(e.getSource() == nextButton)

{

//Sets the player name, Gender and Place ruled and sets them in the Player class

Player.setName(ruler.getText());

if(jBoy.isSelected())

{

Player.setGender(true);

}

else

{

Player.setGender(false);

}

Player.setRuledLocation((String)cStateList.getSelectedItem());

Parent.updateInformation();

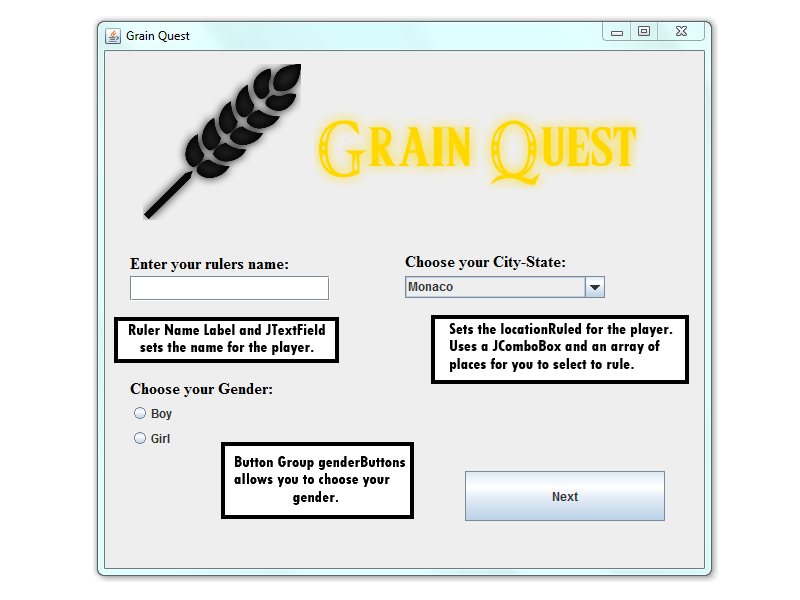
Parent.havestPhase();

}

}

}

**Character Create Diagram**



**This class sets grain destroyed by the rats with a random number from 1 - 50 and takes away that percentage amount from your grain reserve. It then displays the Grain Percentage destroyed by rats and your grain reserve. This uses static variables and methods from the Harvest class and displays that information to the label. The Grain Demand is displayed here which has not been fully coded into the game. Grain Price is also displayed in this class using a static variable from the Harvest class. Currently Grain Price is set as a random double type variable which generates a number from 1-2. This feature isn't fully completed but it would be based off of the amount of serfs you have and how much Land you had. The land variable is also displayed in this class, each time you start off with 10000 hectares of land. The land price is displayed in the panel. Treasury is also displayed and each time you start with 1000 florins.**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.util.Random;

import java.text.DecimalFormat;

public class HarvestPanel extends JPanel implements ActionListener

{

Random r = new Random();

Random landPriceNum = new Random();

//Creates a random number for the rats to destroy grain from 1 - 50%

private int grainDestroyed = r.nextInt(49)+1;

private KingdomController Parent;

private JLabel leaves = new JLabel();

private JLabel harvestPhaseLogo = new JLabel();

private JLabel playerLabel = new JLabel();

private JLabel ratLabel = new JLabel();

private JLabel landPrice = new JLabel();

private JLabel grainReserve = new JLabel();

private JLabel grainDemand = new JLabel();

private JLabel grainPrice = new JLabel();

private JLabel land = new JLabel();

private JLabel treasury = new JLabel();

private JButton nextButton = new JButton("Next");

DecimalFormat df = new DecimalFormat("#.##");

public HarvestPanel(KingdomController parent)

{

setLayout(null);

Parent = parent;

setBackground(Color.BLACK);

leaves.setIcon(new ImageIcon("leaves.gif"));

harvestPhaseLogo.setIcon(new ImageIcon("HarvestPhaseLogo.png"));

leaves.setBounds(45,-15,615,555);

harvestPhaseLogo.setBounds(25,0,600,173);

nextButton.setBounds(360,420,200,50);

ratLabel.setBounds(10,-20,600,400);

grainReserve.setBounds(10,30,600,400);

grainDemand.setBounds(10,80,600,400);

grainPrice.setBounds(10,130,600,400);

land.setBounds(10,180,600,400);

landPrice.setBounds(10,230,600,400);

treasury.setBounds(10,280,600,400);

playerLabel.setBounds(370,-60,400,400);

ratLabel.setFont(new Font("Vivaldi", Font.BOLD, 20));

ratLabel.setForeground(Color.WHITE);

grainReserve.setFont(new Font("Vivaldi", Font.BOLD, 18));

grainReserve.setForeground(Color.WHITE);

grainDemand.setFont(new Font("Vivaldi", Font.BOLD, 18));

grainDemand.setForeground(Color.WHITE);

grainPrice.setFont(new Font("Vivaldi", Font.BOLD, 18));

grainPrice.setForeground(Color.WHITE);

land.setFont(new Font("Vivaldi", Font.BOLD, 18));

land.setForeground(Color.WHITE);

landPrice.setFont(new Font("Vivaldi", Font.BOLD, 18));

landPrice.setForeground(Color.WHITE);

treasury.setFont(new Font("Vivaldi", Font.BOLD, 20));

treasury.setForeground(Color.WHITE);

playerLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

playerLabel.setForeground(Color.WHITE);

Harvest.setDestoryedGrain(grainDestroyed);

//sets a random land price between 2-3

Harvest.setLandPrice(landPriceNum.nextDouble()\*(1.0)+2);

//Destroys grain based on the rats percentage.

Harvest.setGrainReserve(Harvest.getGrainReserve() \* grainDestroyed/100);

ratLabel.setText("Grain destroyed by rats: "+ Harvest.getDestoryedGrain()+ "%");

add(playerLabel);

add(ratLabel);

add(landPrice);

add(grainReserve);

add(grainDemand);

add(grainPrice);

add(land);

add(landPrice);

add(treasury);

add(nextButton);

add(harvestPhaseLogo);

add(leaves);

nextButton.addActionListener(this);

System.out.println(Player.getName());

}

public void actionPerformed(ActionEvent e)

{

if(e.getSource() == nextButton)

{

//runs the method that swtiches the panels

Parent.buySellPhase();

}

}

//updates Information so it can be displayed properly.

public void updateInformation()

{

playerLabel.setText(Player.getGender() + Player.getName() + " of " + Player.getRuledLocation());

grainReserve.setText("Grain Reserve: " + Harvest.getGrainReserve());

grainDemand.setText("Grain Demand: " + Harvest.getGrainDemand());

grainPrice.setText("Grain Price: " + Harvest.getGrainPrice() + "ƒ");

land.setText("Land: " + Harvest.getLand());

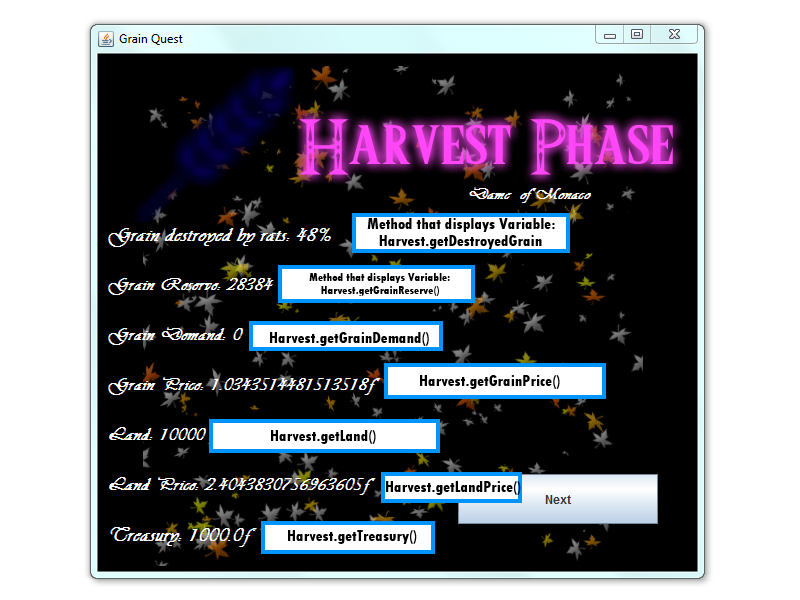
landPrice.setText("Land Price: " + Harvest.getLandPrice() + "ƒ");

treasury.setText("Treasury: " + Harvest.getTreasury() + "ƒ");

}

}

**Harvest Phase Diagram**

****

**This class handles all of the buying and selling of the grain logic. This currently works, although the prices for the land and grain are random numbers and not based on the amount of serfs and land like it should be. The amount of land that can be bought or sold is currently no limit so it is possible to go into negative money. Buying and selling does work though by taking money from your funds using a formula. It displays all the variables that are displayed in your haarvest panel to the right of the screen except % of destroyed grain and displays two more variables than the harevst phase does which are grain price and land price.**

**What needs to be Added:**

**A proper way of generating the right grain and land price with a formula thats based on the number of serfs you have**

**A Limit on how much you can sell/buy (currently no limit on anything)**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class BuySellPanel extends JPanel implements ActionListener

{

private JButton nextButton;

private JButton buyGrainButton = new JButton("Buy Grain");

private JButton sellGrainButton = new JButton("Sell Grain");

private JButton buyLandButton = new JButton("Buy Land");

private JButton sellLandButton = new JButton("Sell Land");

private JLabel backgroundPicture;

//Creates a new instance of kingdom controller and names it Parent

private KingdomController Parent;

private JLabel playerLabel = new JLabel();

private JLabel yearLabel = new JLabel();

private JLabel buyGrainLabel = new JLabel("Grain (steres): ");

private JLabel sellGrainLabel = new JLabel("Grain (steres): ");

private JLabel buyLandLabel = new JLabel("Land (hectares): ");

private JLabel sellLandLabel = new JLabel("Land (hectares): ");

private JLabel grainReserve = new JLabel();

private JLabel grainDemand = new JLabel();

private JLabel grainPrice = new JLabel();

private JLabel land = new JLabel();

private JLabel landPrice = new JLabel();

private JLabel treasury = new JLabel();

private JTextField buyGrain = new JTextField();

private JTextField sellGrain = new JTextField();

private JTextField buyLand = new JTextField();

private JTextField sellLand = new JTextField();

public BuySellPanel(KingdomController parent)

{

setLayout(null);

Parent = parent;

nextButton = new JButton("Next");

backgroundPicture = new JLabel();

backgroundPicture.setIcon(new ImageIcon("marketbg.png"));

backgroundPicture.setBounds(0,-13,615,555);

nextButton.setBounds(360,420,200,50);

buyGrainLabel.setBounds(20,180,200,50);

sellGrainLabel.setBounds(20,210,200,50);;

buyLandLabel.setBounds(20,240,200,50);

sellLandLabel.setBounds(20,270,200,50);

buyGrain.setBounds(140,198,100,18);

sellGrain.setBounds(140,228,100,18);

buyLand.setBounds(149,258,100,18);

sellLand.setBounds(149,288,100,18);

grainReserve.setBounds(374,170,200,50);

grainDemand.setBounds(374,200,200,50);

grainPrice.setBounds(374,230,200,50);

land.setBounds(374,260,200,50);

landPrice.setBounds(374,290,200,50);

treasury.setBounds(374,320,200,50);

buyGrainButton.setBounds(20,350,100,20);

sellGrainButton.setBounds(140,350,100,20);

buyLandButton.setBounds(20,390,100,20);

sellLandButton.setBounds(140,390,100,20);

playerLabel.setBounds(200,-157,400,400);

yearLabel.setBounds(200,-175,400,400);

buyGrainLabel.setFont(new Font("Vivaldi", Font.BOLD, 18));

buyGrainLabel.setForeground(Color.BLACK);

sellGrainLabel.setFont(new Font("Vivaldi", Font.BOLD, 18));

sellGrainLabel.setForeground(Color.BLACK);

buyLandLabel.setFont(new Font("Vivaldi", Font.BOLD, 18));

buyLandLabel.setForeground(Color.BLACK);

sellLandLabel.setFont(new Font("Vivaldi", Font.BOLD, 18));

sellLandLabel.setForeground(Color.BLACK);

playerLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

playerLabel.setForeground(Color.WHITE);

yearLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

yearLabel.setForeground(Color.WHITE);

grainReserve.setFont(new Font("Vivaldi", Font.BOLD, 18));

grainReserve.setForeground(Color.BLACK);

grainDemand.setFont(new Font("Vivaldi", Font.BOLD, 18));

grainDemand.setForeground(Color.BLACK);

grainPrice.setFont(new Font("Vivaldi", Font.BOLD, 18));

grainPrice.setForeground(Color.BLACK);

land.setFont(new Font("Vivaldi", Font.BOLD, 18));

land.setForeground(Color.BLACK);

landPrice.setFont(new Font("Vivaldi", Font.BOLD, 18));

landPrice.setForeground(Color.BLACK);

treasury.setFont(new Font("Vivaldi", Font.BOLD, 18));

treasury.setForeground(Color.BLACK);

add(nextButton);

add(buyGrainButton);

add(sellGrainButton);

add(buyLandButton);

add(sellLandButton);

add(grainReserve);

add(grainDemand);

add(grainPrice);

add(land);

add(landPrice);

add(treasury);

add(playerLabel);

add(buyGrainLabel);

add(sellGrainLabel);

add(buyLandLabel);

add(sellLandLabel);

add(buyGrain);

add(sellGrain);

add(buyLand);

add(sellLand);

add(yearLabel);

add(backgroundPicture);

nextButton.addActionListener(this);

buyGrainButton.addActionListener(this);

sellGrainButton.addActionListener(this);

buyLandButton.addActionListener(this);

sellLandButton.addActionListener(this);

}

public void actionPerformed(ActionEvent e)

{

if(e.getSource() == nextButton)

{

//runs panel switching method

Parent.sellGrainPhase();

}

if(e.getSource() == buyGrainButton)

{

//Runs a formula to buy the grain. Treasury minus the amount of grain you want to by multiplied by the grain price.

int bGrain = Integer.parseInt(buyGrain.getText());

double bGrainTransaction = Harvest.getTreasury() - (bGrain\*Harvest.getGrainPrice());

//adds Grain to reserve

Harvest.addGrainReserve(bGrain);

grainReserve.setText("Grain Reserve: " + Harvest.getGrainReserve());

Harvest.setTreasury(bGrainTransaction);

treasury.setText("Treasury: " + Harvest.getTreasury() + "ƒ");

}

if(e.getSource() == sellGrainButton)

{

//Runs a formula to sell the grain. Treasury plus the amount of grain you want to by multiplied by the grain price.

int sGrain = Integer.parseInt(sellGrain.getText());

double sGrainTransaction = Harvest.getTreasury() + (sGrain\*Harvest.getGrainPrice());

//subtracts Grain from reserve

Harvest.subtractGrainReserve(sGrain);

Harvest.setTreasury(sGrainTransaction);

grainReserve.setText("Grain Reserve: " + Harvest.getGrainReserve());

treasury.setText("Treasury: " + Harvest.getTreasury() + "ƒ");

}

if(e.getSource() == buyLandButton)

{

//Runs a formula to buy the land. Treasury minus the amount of land you want to by multiplied by the land price.

int bLand = Integer.parseInt(buyLand.getText());

double buyLandTransaction = Harvest.getTreasury() - (bLand\*Harvest.getLandPrice());

Harvest.setTreasury(buyLandTransaction);

//adds Land your bought to your land

Harvest.setLand(Harvest.getLand() + bLand);

land.setText("Land: " + Harvest.getLand());

treasury.setText("Treasury: " + Harvest.getTreasury() + "ƒ");

}

if(e.getSource() == sellLandButton)

{

//Runs a formula to sell the land. Treasury plus the amount of land you want to by multiplied by the land price.

int sLand = Integer.parseInt(sellLand.getText());

double transaction = Harvest.getTreasury() + (sLand\*Harvest.getLandPrice());

//subtracts the Land you sold from your land

Harvest.setLand(Harvest.getLand() - sLand);

Harvest.setTreasury(transaction);

land.setText("Land: " + Harvest.getLand());

treasury.setText("Treasury: " + Harvest.getTreasury() + "ƒ");

}

}

//updates Information so it can be displayed properly.

public void updateInformation()

{

playerLabel.setText(Player.getGender() + Player.getName() + " of " + Player.getRuledLocation());

grainReserve.setText("Grain Reserve: " + Harvest.getGrainReserve());

grainDemand.setText("Grain Demand: " + Harvest.getGrainDemand());

grainPrice.setText("Grain Price: " + Harvest.getGrainPrice() + "ƒ");

land.setText("Land: " + Harvest.getLand());

landPrice.setText("Land Price: " + Harvest.getLandPrice() + "ƒ");

treasury.setText("Treasury: " + Harvest.getTreasury() + "ƒ");

yearLabel.setText("Year " + Harvest.getYear());

}

}

**Buy/Sell Diagram**



**This class is responsible for releasing grain to your serfs. Currently serfs are not coded into the game yet. Currently what is working is when you enter an amount of grain in the text field and click "Release!" It will subtract the number of grain from your reserve. Doing this will also enable the next button which will allow you to continue to the next panel.**

**What needs to be added:**

**A formula for the serfs to be born / die based on the amount of grain you release**

**A limit on how much grain you can use (You can currently go into negative grain)**

**A limit on the amount you can keep(at least 20%)**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class GranaryPanel extends JPanel implements ActionListener

{

private JButton nextButton;

private JButton releaseButton = new JButton("Release!");

private JLabel backgroundPicture;

private JLabel playerLabel = new JLabel();

private KingdomController Parent;

private JLabel grainReserve = new JLabel();

private JLabel grainDemand = new JLabel();

private JLabel yearLabel = new JLabel();

private JLabel grainReleasedLabel = new JLabel();

private JTextField releaseGrain = new JTextField();

public GranaryPanel(KingdomController parent)

{

setLayout(null);

Parent = parent;

nextButton = new JButton("Next");

nextButton.setEnabled(false);

backgroundPicture = new JLabel();

backgroundPicture.setIcon(new ImageIcon("Granarybg.png"));

backgroundPicture.setBounds(0,-13,615,555);

nextButton.setBounds(360,420,200,50);

releaseButton.setBounds(238,250,100,30);

grainReserve.setBounds(50,350,200,50);

grainDemand.setBounds(50,400,200,50);

releaseGrain.setBounds(190,210,200,30);

grainReleasedLabel.setBounds(150,145,400,300);

playerLabel.setBounds(200,-160,400,400);

yearLabel.setBounds(200,-175,400,400);

playerLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

yearLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

yearLabel.setForeground(Color.BLACK);

playerLabel.setForeground(Color.BLACK);

grainReserve.setFont(new Font("Vivaldi", Font.BOLD, 18));

grainReserve.setForeground(Color.BLACK);

grainDemand.setFont(new Font("Vivaldi", Font.BOLD, 18));

grainDemand.setForeground(Color.BLACK);

grainReleasedLabel.setFont(new Font("Vivaldi", Font.BOLD, 18));

grainReleasedLabel.setForeground(Color.BLACK);

add(playerLabel);

add(grainDemand);

add(grainReserve);

add(releaseGrain);

add(grainReleasedLabel);

add(nextButton);

add(releaseButton);

add(yearLabel);

add(backgroundPicture);

nextButton.addActionListener(this);

releaseButton.addActionListener(this);

}

public void actionPerformed(ActionEvent e)

{

if(e.getSource() == nextButton)

{

//runs the method that swtiches the panels

Parent.serfReport();

}

if(e.getSource() == releaseButton)

{

//Releases the amount of grain you input.

int rGrain = Integer.parseInt(releaseGrain.getText());

Harvest.subtractGrainReserve(rGrain);

grainReleasedLabel.setText("You have released " + rGrain + " grain to your serfs!");

grainReserve.setText("Grain Reserve: " + Harvest.getGrainReserve());

nextButton.setEnabled(true);

}

}

//Updates information so it can be displayed properly.

public void updateInformation()

{

playerLabel.setText(Player.getGender() + Player.getName() + " of " + Player.getRuledLocation());

grainReserve.setText("Grain Reserve: " + Harvest.getGrainReserve());

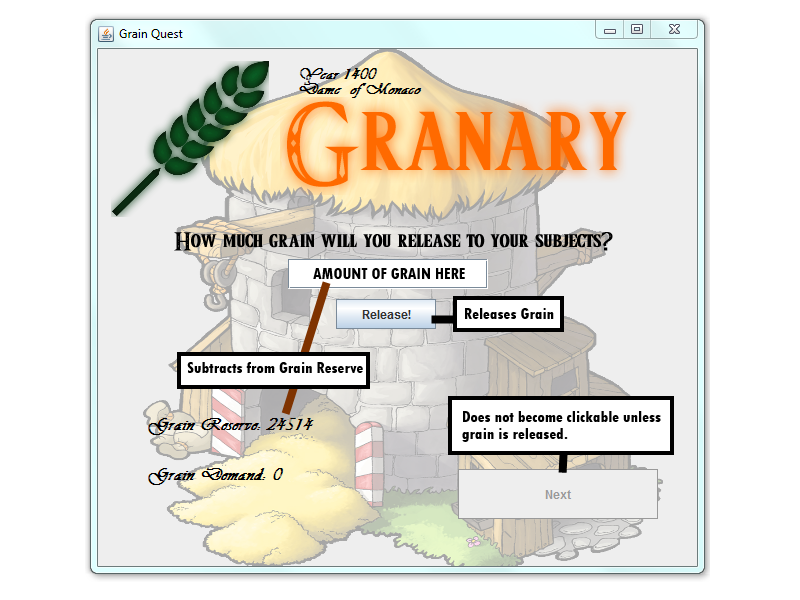
grainDemand.setText("Grain Demand: " + Harvest.getGrainDemand());

yearLabel.setText("Year " + Harvest.getYear());

}

}

**Granary Diagram**



**The serf report panel is meant to show the number of serfs you gained, the number of serfs that haved died and the number of clergymen and merchants you have gained. Currently serfs are not coded into the game but if they were they would be used to determine land price, grain price, and the amount of money you make from them.**

**Currently not working:**

**Serfs**

**Clergymen and Merchants**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class SerfPanel extends JPanel implements ActionListener

{

private int numSerfsDead = 0;

private int numSerfsGained = 0;

private JButton nextButton;

private JLabel backgroundPicture;

private JLabel playerLabel = new JLabel();

private JLabel yearLabel = new JLabel();

private JLabel serfLabel = new JLabel();

private JLabel serfsDead = new JLabel();

private JLabel serfsGained = new JLabel();

private KingdomController Parent;

public SerfPanel(KingdomController parent)

{

setLayout(null);

Parent = parent;

nextButton = new JButton("Next");

backgroundPicture = new JLabel();

backgroundPicture.setIcon(new ImageIcon("SerfReport.png"));

backgroundPicture.setBounds(0,-13,615,555);

nextButton.setBounds(360,420,200,50);

playerLabel.setBounds(168,-155,400,400);

serfLabel.setBounds(130,150,400,400);

yearLabel.setBounds(200,-175,400,400);

serfsDead.setBounds(130,50,400,400);

serfsGained.setBounds(330,50,400,400);

playerLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

playerLabel.setForeground(Color.BLACK);

serfLabel.setFont(new Font("Vivaldi", Font.BOLD, 20));

serfLabel.setForeground(Color.BLACK);

serfsDead.setFont(new Font("Vivaldi", Font.BOLD, 20));

serfsDead.setForeground(Color.BLACK);

serfsGained.setFont(new Font("Vivaldi", Font.BOLD, 20));

serfsGained.setForeground(Color.BLACK);

yearLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

yearLabel.setForeground(Color.BLACK);

serfsDead.setText("Serfs Dead: " + numSerfsDead);

serfsGained.setText("Serfs Gained: " + numSerfsDead);

add(nextButton);

add(serfLabel);

add(serfsDead);

add(serfsGained);

add(playerLabel);

add(yearLabel);

add(backgroundPicture);

nextButton.addActionListener(this);

}

public void actionPerformed(ActionEvent e)

{

if(e.getSource() == nextButton)

{

//runs the method that swtiches the panels

Parent.taxPhase();

}

}

//Updates Information so it can be displayed properly.

public void updateInformation()

{

playerLabel.setText(Player.getGender() + Player.getName() + " of " + Player.getRuledLocation());

serfLabel.setText("Total Serfs: " + Harvest.getSerfs());

yearLabel.setText("Year " + Harvest.getYear());

}

}

**Serf Report Diagram**



**This panel is meant to display the Customs Duty, Sales Tax, Wealth Tax, and set the rate for each, which will increase your florins in each phase accordingly. You can also set the Customs Duty, Sales Tax, Wealth Tax to 1 of 4 options using a combo box to; Very fair which will give you -200f each stage or Moderate which will give you 100f each phase or HARSH will give you 400f each phase. It will then display the tax revenue of each phase. the fairer the tax, the more serfs that will migrate to your land.**

**Not Finished:**

**Customs Duty, Sales Tax, Wealth Tax**

**Justice level**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class TaxPanel extends JPanel implements ActionListener

{

private JButton nextButton;

private JLabel backgroundPicture;

private JLabel playerLabel = new JLabel();

private JLabel yearLabel = new JLabel();

private KingdomController Parent;

public TaxPanel(KingdomController parent)

{

setLayout(null);

Parent = parent;

nextButton = new JButton("Next");

backgroundPicture = new JLabel();

backgroundPicture.setIcon(new ImageIcon("Taxbg.png"));

backgroundPicture.setBounds(0,-13,615,555);

nextButton.setBounds(360,420,200,50);

playerLabel.setBounds(200,-156,400,400);

yearLabel.setBounds(200,-175,400,400);

playerLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

playerLabel.setForeground(Color.BLACK);

yearLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

yearLabel.setForeground(Color.BLACK);

add(nextButton);

add(playerLabel);

add(yearLabel);

add(backgroundPicture);

nextButton.addActionListener(this);

}

public void actionPerformed(ActionEvent e)

{

if(e.getSource() == nextButton)

{

//runs the method that swtiches the panels

Parent.mapPhase();

}

}

//updates Information so it can be displayed properly.

public void updateInformation()

{

playerLabel.setText(Player.getGender() + Player.getName() + " of " + Player.getRuledLocation());

yearLabel.setText("Year " + Harvest.getYear());

}

}

**This map panel is responsible for handling everything involving the map. It supposed to set images of markets, wooden mills, Cathedrals and palaces to be visible when you buy them. Each time you buy one of these you gain points. If you gain enough points then your Name will be changed from 'Sir' to a higher ranking until you become king and win the game. Points are set in the Player class.**

**Not Done:**

**Currently the Player class does not have the appropriate variables and methods to handle the points.**

**No name upgrades coded in yet.**

**You cannot buy or sell markets yet.**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class MapPanel extends JPanel implements ActionListener

{

private JButton nextButton;

private JLabel backgroundPicture;

private JLabel playerLabel = new JLabel();

private JLabel yearLabel = new JLabel();

private KingdomController Parent;

public MapPanel(KingdomController parent)

{

setLayout(null);

Parent = parent;

nextButton = new JButton("Next");

backgroundPicture = new JLabel();

backgroundPicture.setIcon(new ImageIcon("Map.png"));

backgroundPicture.setBounds(0,-13,615,555);

nextButton.setBounds(360,420,200,50);

yearLabel.setBounds(200,-175,400,400);

playerLabel.setBounds(360,-130,400,400);

playerLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

playerLabel.setForeground(Color.WHITE);

yearLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

yearLabel.setForeground(Color.WHITE);

add(nextButton);

add(playerLabel);

add(yearLabel);

add(backgroundPicture);

nextButton.addActionListener(this);

}

public void actionPerformed(ActionEvent e)

{

if(e.getSource() == nextButton)

{

//runs the method that swtiches the panels

Parent.investmentPhase();

}

}

public void updateInformation()

{

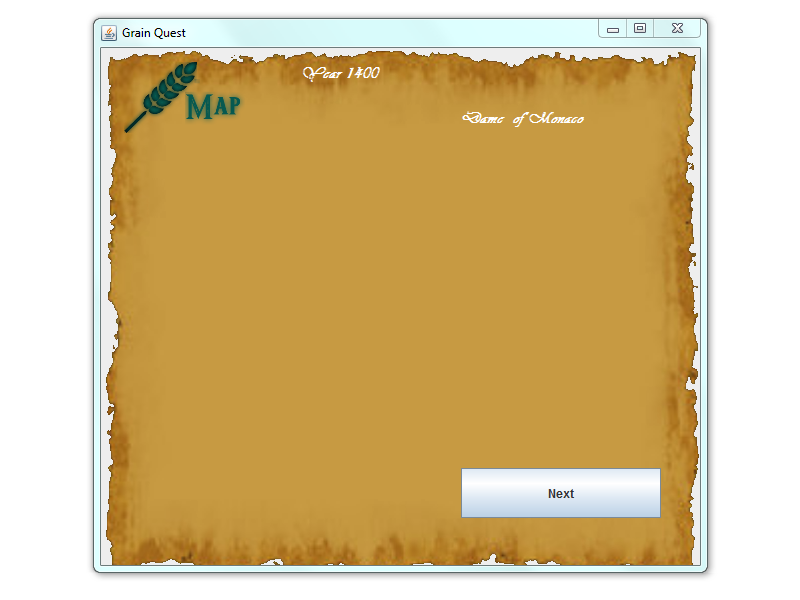
playerLabel.setText(Player.getGender() + Player.getName() + " of " + Player.getRuledLocation());

yearLabel.setText("Year " + Harvest.getYear());

}

}

**Map Diagram**



**This class handles all of the Cathedrals, Palaces, Wooden Mills, Markets and Soldiers owned.**

**It displays a price for each one with a button to purchase that item. It also displays your treasury amount. Currently none of this is implemented but this is what the class is supposed to do the next button will also loop back to the Harvest Phase which does work currently.**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class InvestmentPanel extends JPanel implements ActionListener

{

private JButton nextButton;

private JLabel backgroundPicture;

private JLabel playerLabel = new JLabel();

private JLabel yearLabel = new JLabel();

private KingdomController Parent;

public InvestmentPanel(KingdomController parent)

{

setLayout(null);

Parent = parent;

nextButton = new JButton("Next");

backgroundPicture = new JLabel();

backgroundPicture.setIcon(new ImageIcon("Investments.png"));

backgroundPicture.setBounds(0,-13,615,555);

nextButton.setBounds(360,420,200,50);

yearLabel.setBounds(200,-175,400,400);

playerLabel.setBounds(200,-60,400,400);

playerLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

playerLabel.setForeground(Color.BLACK);

yearLabel.setFont(new Font("Vivaldi", Font.BOLD, 16));

yearLabel.setForeground(Color.BLACK);

add(nextButton);

add(playerLabel);

add(yearLabel);

add(backgroundPicture);

nextButton.addActionListener(this);

}

public void actionPerformed(ActionEvent e)

{

if(e.getSource() == nextButton)

{

//runs the method that swtiches the panels

Parent.loopPhase();

Harvest.addYear();

yearLabel.setText("Year " + Harvest.getYear());

}

}

//updates Information so it can be displayed properly.

public void updateInformation()

{

playerLabel.setText(Player.getGender() + Player.getName() + " of " + Player.getRuledLocation());

yearLabel.setText("Year " + Harvest.getYear());

}

}

**This public class contains gets and sets methods for the Kingdom Project. The get and set are used to set values to variables and then retrieve them. The gets and sets in this class are used for the different phases of the game. For example, it is here where the price and reserve of grain are determined. Also other values are determined here like land, serfs, treasury, and year. For the most part it’s within this class that the different values for the phase come from.**

import java.util.Random;

public class Harvest{

// The static Random grainRand is used to determine a random percentage of the grainPrice

// The static Random r is used to determine a random percentage of the grainReserve

// The varibles are set as so they can run on all instances

private static Random r = new Random();

private static Random grainRand = new Random();

private static int grain = 0;

private static int land = 10000;

private static int serfs = 2000;

private static int steres = 0;

private static double landPrice = 0;

private static double grainPrice = grainRand.nextDouble()\*(0.1)+1;

private static int grainDemand = 0;

private static int grainReserve = r.nextInt(100000-50000 + 1) + 50000;

private static double treasury = 1000;

private static int sellGrain = 0;

private static int buyGrain = 0;

private static int sellLand = 0;

private static int buyLand= 0;

private static int destoryedGrain = 0;

private static int soldiers = 0;

private static int convertSerfs = 0;

private static int markets = 0;

private static int marketInvest = 0;

private static int taxRevenue = 0;

private static int produceIncome = 0;

private static int woolenMills = 0;

private static int woolenInvest = 0;

private static int jobs = 0;

private static int produceProfits = 0;

private static int palaces = 0;

private static int palaceInvest = 0;

private static int nobility = 0;

private static int catherals = 0;

private static int clergy = 0;

private static int year = 1400;

private static String lowtaxRate = "";

private static String midtaxRate = "";

private static String hightaxRate = "";

private static String unjustLaw = "";

private static String justLaw = "";

private static String fairLaw = "";

public static int getGrain(){

return grain;

}

public static int getLand(){

return land;

}

public static int getSerfs(){

return serfs;

}

public static int getSteres(){

return steres;

}

public static double getLandPrice(){

return landPrice;

}

public static double getGrainPrice(){

return grainPrice;

}

public static int getGrainDemand(){

return grainDemand;

}

public static int getGrainReserve(){

return grainReserve;

}

public static double getTreasury(){

return treasury;

}

public static int getSellGrain(){

return sellGrain;

}

public static int getBuyGrain(){

return buyGrain;

}

public static int getSellLand(){

return sellLand;

}

public static int getBuyLand(){

return buyLand;

}

public static int getDestoryedGrain(){

return destoryedGrain;

}

public static int getSoldiers(){

return soldiers;

}

public static int getConvertSerfs(){

return convertSerfs;

}

public static int getMarkets(){

return markets;

}

public static int getMarketInvest(){

return marketInvest;

}

public static int getTaxRevenue(){

return taxRevenue;

}

public static int getProduceIncome(){

return produceIncome;

}

public static int getWoolenMills(){

return woolenMills;

}

public static int getWoolenInvest(){

return woolenInvest;

}

public static int getJobs(){

return jobs;

}

public static int getProduceProfits(){

return produceProfits;

}

public static int getPalaces(){

return palaces;

}

public static int getPalaceInvest(){

return palaceInvest;

}

public static int getNobility(){

return nobility;

}

public static int getCatherals(){

return catherals;

}

public static int getClergy(){

return clergy;

}

public static int getYear(){

return year;

}

public static String getLowtaxRate(){

return lowtaxRate;

}

public static String getMidtaxRate(){

return midtaxRate;

}

public static String getHightaxRate(){

return hightaxRate;

}

public static String getUnjustLaw(){

return unjustLaw;

}

public static String getJustLaw(){

return justLaw;

}

public static String getFairLaw(){

return fairLaw;

}

public static void setGrain(int aGrain){

grain = aGrain;

}

public static void setLand(int aLand){

land = aLand;

}

public static void setSerfs(int aSerfs){

serfs = aSerfs;

}

public static void setSteres(int aSteres){

steres = aSteres;

}

public static void setLandPrice(double aLandPrice){

landPrice = aLandPrice;

}

public static void setGrainPrice(double aGrainPrice){

grainPrice = aGrainPrice;

}

public static void setGrainDemand(int aGrainDemand){

grainDemand = aGrainDemand;

}

public static void setGrainReserve(int aGrainReserve){

grainReserve = aGrainReserve;

}

//The subtractGrainReserve and addGrainReserve are used to either add or subtract to the current grainReserve

public static void subtractGrainReserve(int aGrain){

int newGrain;

newGrain = grainReserve - aGrain;

grainReserve = newGrain;

}

public static void addGrainReserve(int aGrain){

int newGrain;

newGrain = grainReserve + aGrain;

grainReserve = newGrain;

}

public static void setTreasury(double aTreasury){

treasury = aTreasury;

}

public static void setSellGrain(int aSellGrain){

sellGrain = aSellGrain;

}

public static void setBuyGrain(int aBuyGrain){

buyGrain = aBuyGrain;

}

public static void setSellLand(int aSellLand){

sellLand = aSellLand;

}

public static void setBuyLand(int aBuyLand){

buyLand = aBuyLand;

}

public static void setDestoryedGrain(int aDestoryedGrain){

destoryedGrain = aDestoryedGrain;

}

public static void setSoldiers(int aSoldiers){

soldiers = aSoldiers;

}

public static void setConvertSerfs(int aConvertSerfs){

soldiers = aConvertSerfs;

}

public static void setMarkets(int aMarkets){

markets = aMarkets;

}

public static void setMarketInvest(int aMarketInvest){

marketInvest = aMarketInvest;

}

public static void setTaxRevenue(int aTaxRevenue){

taxRevenue = aTaxRevenue;

}

public static void setProduceIncome(int aProduceIncome){

produceIncome = aProduceIncome;

}

public static void setWoolenMills(int aWoolenMills){

woolenMills = aWoolenMills;

}

public static void setWoolenInvest(int aWoolenInvest){

woolenInvest = aWoolenInvest;

}

public static void setJobs(int aJobs){

jobs = aJobs;

}

public static void setProduceProfits(int aProduceProfits){

produceProfits = aProduceProfits;

}

public static void setPalaces(int aPalaces){

palaces = aPalaces;

}

public static void setPalaceInvest(int aPalaceInvest){

palaceInvest = aPalaceInvest;

}

public static void setNobility(int aNobility){

nobility = aNobility;

}

public static void setCatherals(int aCatherals){

catherals = aCatherals;

}

public static void setClergy(int aClergy){

clergy = aClergy;

}

public static void addYear(){

year++;

}

public static void setLowtaxRate(String aLowtaxRate){

lowtaxRate = aLowtaxRate;

}

public static void setMidtaxRate(String aMidtaxRate){

midtaxRate = aMidtaxRate;

}

public static void setHightaxRate(String aHightaxRate){

hightaxRate = aHightaxRate;

}

public static void setUnjustLaw(String aUnjustLaw){

unjustLaw = aUnjustLaw;

}

public static void setJustLaw(String aJustLaw){

unjustLaw = aJustLaw;

}

public static void setFairLaw(String aFairLaw){

fairLaw = aFairLaw;

}

public String toString(){

String aString = "";

aString = aString + "Grain " + this.getGrain() + "\n";

aString = aString + "Land " + this.getLand() + "\n";

aString = aString + "Serfs " + this.getSerfs() + "\n";

aString = aString + "Steres " + this.getSteres() + "\n";

aString = aString + "Land Price " + this.getLandPrice() + "\n";

aString = aString + "Grain Price " + this.getGrainPrice() + "\n";

aString = aString + "Grain Demand " + this.getGrainDemand() + "\n";

aString = aString + "Grain Reserve " + this.getGrainReserve() + "\n";

aString = aString + "Treasury " + this.getTreasury() + "\n";

aString = aString + "Sell Grain " + this.getSellGrain() + "\n";

aString = aString + "Buy Grain " + this.getBuyGrain() + "\n";

aString = aString + "Sell Land " + this.getSellLand() + "\n";

aString = aString + "Buy Land " + this.getBuyLand() + "\n";

aString = aString + "Destoryed Grain " + this.getDestoryedGrain() + "\n";

aString = aString + "Soldiers " + this.getSoldiers() + "\n";

aString = aString + "Convert Serfs " + this.getConvertSerfs() + "\n";

aString = aString + "Markets " + this.getMarkets() + "\n";

aString = aString + "Market Invest " + this.getMarketInvest() + "\n";

aString = aString + "Tax Revenue " + this.getTaxRevenue() + "\n";

aString = aString + "Produce Income " + this.getProduceIncome() + "\n";

aString = aString + "Woolen Mills " + this.getWoolenMills() + "\n";

aString = aString + "Woolen Invest " + this.getWoolenInvest() + "\n";

aString = aString + "Jobs " + this.getJobs() + "\n";

aString = aString + "Produce Profits " + this.getProduceProfits() + "\n";

aString = aString + "Palaces " + this.getPalaces() + "\n";

aString = aString + "Palace Invest " + this.getPalaceInvest() + "\n";

aString = aString + "Nobility " + this.getNobility() + "\n";

aString = aString + "Catherals " + this.getCatherals() + "\n";

aString = aString + "Clergy " + this.getClergy() + "\n";

aString = aString + "Low Tax Rate " + this.getLowtaxRate() + "\n";

aString = aString + "Mid Tax Rate " + this.getMidtaxRate() + "\n";

aString = aString + "High Tax Rate " + this.getHightaxRate() + "\n";

aString = aString + "Unjust Law " + this.getUnjustLaw() + "\n";

aString = aString + "Just Law " + this.getJustLaw() + "\n";

aString = aString + "Fair Law " + this.getFairLaw() + "\n";

return aString;

}

}

**Contains the gets and sets for the gender, locataion and gender name. This information is globaly displayed through each panel.**

public class Player

{

static Boolean playerGender = true;

static String genderName = "";

static String playerLocation = "";

static String playerName = "";

static int points = 0;

public static void setGender(Boolean gender)

{

playerGender = gender;

}

public static void setRuledLocation(String location)

{

playerLocation = location;

}

public static void setName(String name)

{

playerName = name;

}

public static String getGender()

{

//picks your gender based on a boolean which is set in the CharacterCreatePanel and sets it as you gender

if(playerGender == true)

{

genderName = "Sir ";

}

else

{

genderName = "Dame ";

}

return genderName;

}

public static String getRuledLocation()

{

return playerLocation;

}

public static String getName()

{

return playerName;

}

}