Program Cover Sheet

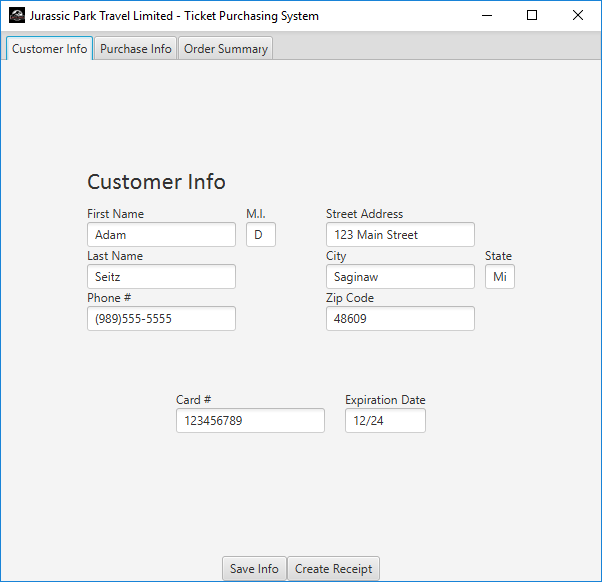
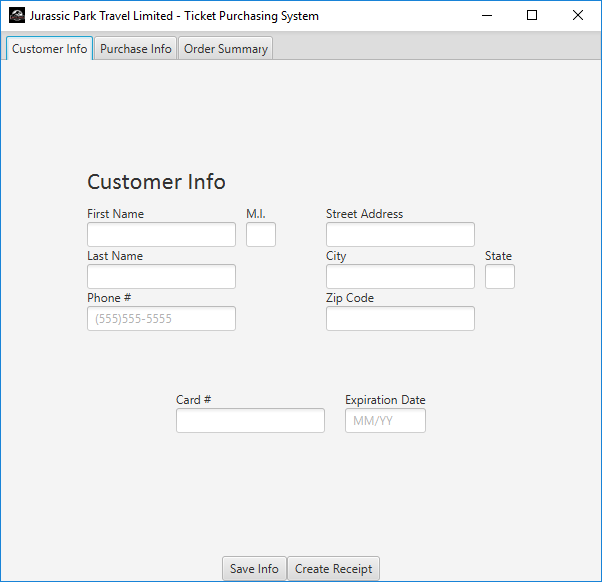
|  |
| --- |
| Name: Adam D. Seitz |
| Assignment: Assignment 1 - Jurassic Park Travel Limited: Ticket Purchasing System (9-5-19) |
| Program is completely functional to my knowledge and testing.  Before compiling and running my program, navigate to the “application” folder containing the .java files within the command prompt. You then need to set the path to your java jdk bin.  (Ex: set “path=%path%;C:\Program Files\Java\jdk1.8.0\_202\bin”)  To compile, type in the command prompt “javac -d . \*.java”. This will create compile your files and put them into an “application” subfolder.  To execute the program, type in the command prompt “java application.Main” to run the application. |

|  |
| --- |
| Instructor’s Comments: |
| Grade: |

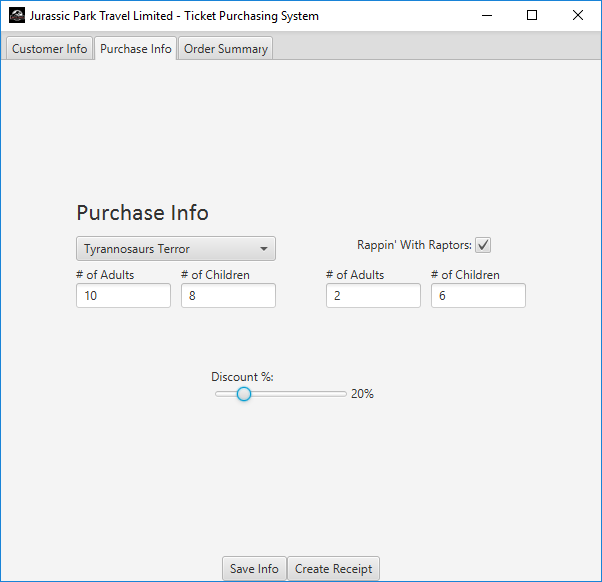
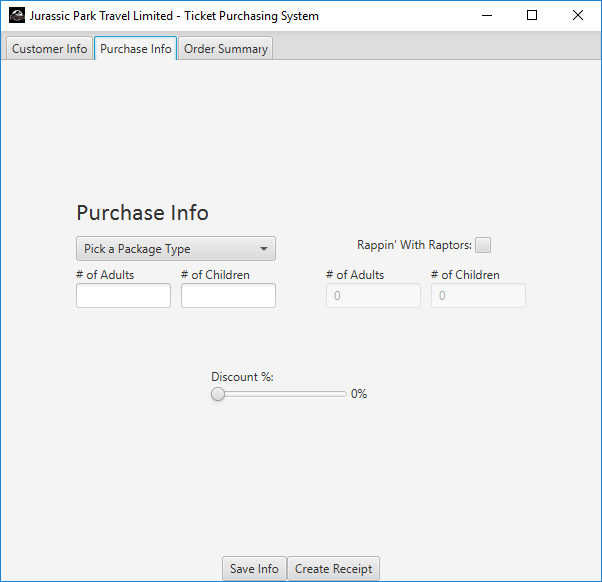
Program Submission Requirements: (1) all files, zipped and uploaded to Canvas and (2) a completed cover sheet, program execution screenshots and source code printed, **stapled** and turned in during class. Failure to follow the submission requirements will result in points lost on that particular assignment.

Screen Shots

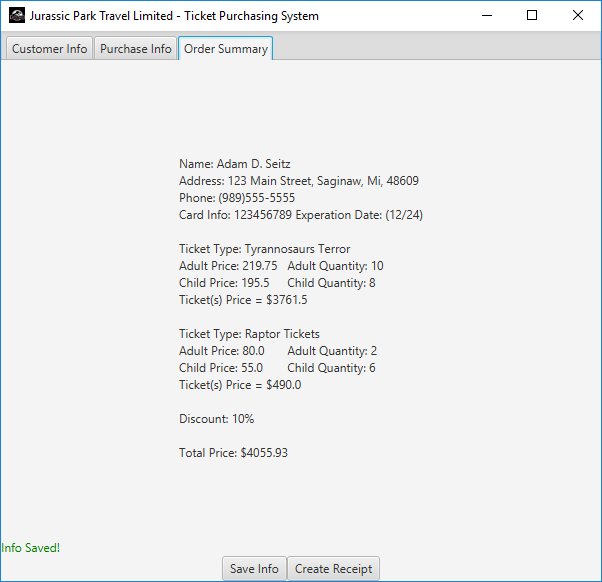
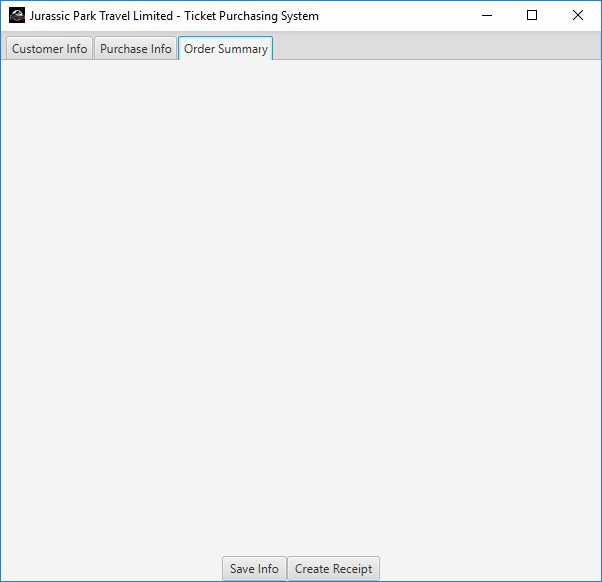
Customer Info



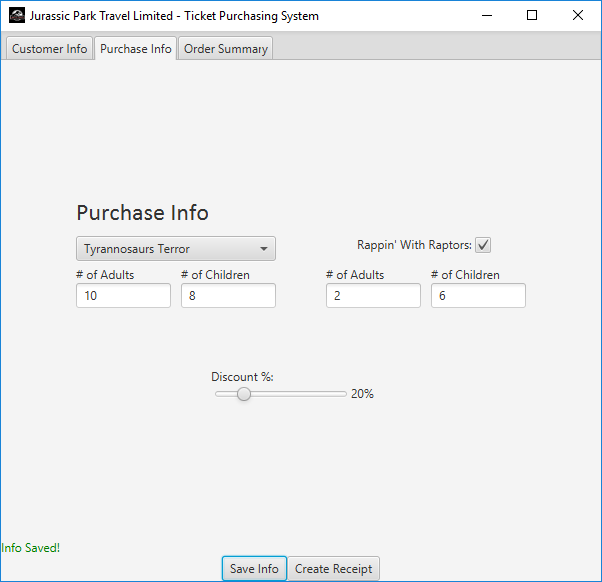
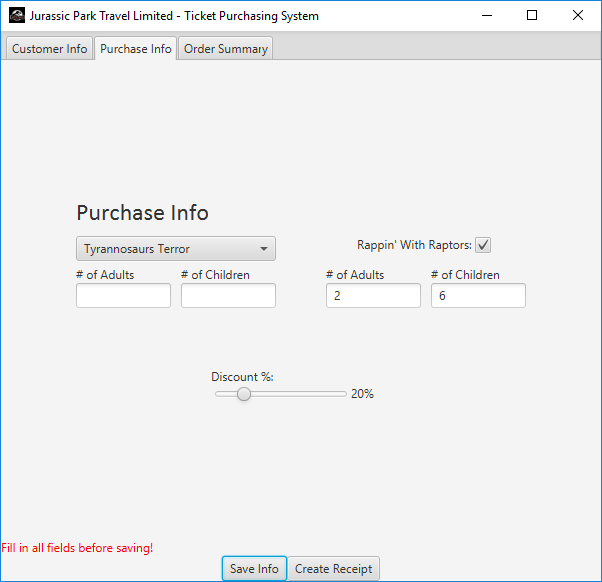
Purchase Info



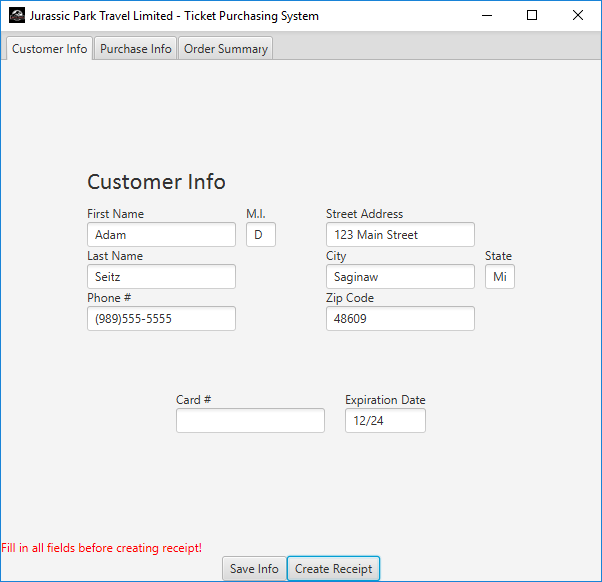
Order Summary



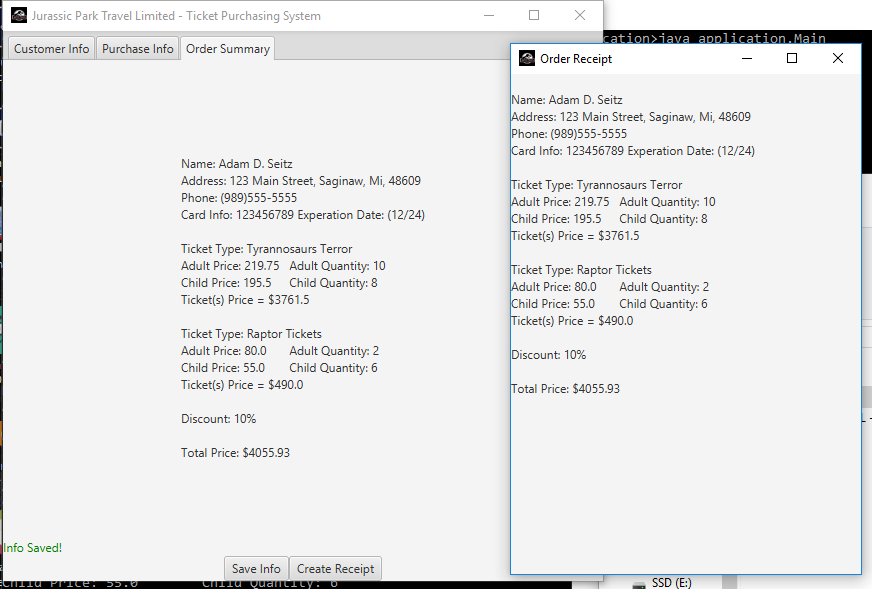
Field Completion Check



Check Fields Before Creating Receipt



Receipt Generated in New Tab



Source Code

/\*

------------------------------------------------------------

- File Name: Customer.java -

- Written By: Adam D. Seitz -

- Written On: 9/4/2019 -

------------------------------------------------------------

- File Purpose: -

- -

- This file contains the Customer class -

------------------------------------------------------------

- Program Purpose: -

- -

- This program takes customer information as -

- input from an employee and will -

- print a receipt when the form is completely filled out. -

------------------------------------------------------------

- Local Variable Dictionary (alphabetically): -

- -

- cardNum - Card number for purchasing -

- city - City of the customer -

- expDate - Expiration date of purchasing card -

- firstName - First name of the customer -

- lastName - Last name of the customer -

- middleInitial - Middle initial of the customer -

- phone - Phone number of the customer -

- state - State of the customer -

- streetAdd - Street address of the customer -

- zip - Zip code of the customer -

------------------------------------------------------------

\*/

package application;

public class Customer {

//Customer info variables

public String firstName;

public String middleInitial;

public String lastName;

public String streetAdd;

public String city;

public String state;

public int zip;

public String phone;

public int cardNum;

public String expDate;

//Constructor with all variables entered

public Customer(String firstName, String middleInitial, String lastName, String streetAdd, String city, String state, int zip, String phone, int cardNum, String expDate){

this.firstName = firstName;

this.middleInitial = middleInitial;

this.lastName = lastName;

this.streetAdd = streetAdd;

this.city = city;

this.state = state;

this.zip = zip;

this.phone = phone;

this.cardNum = cardNum;

this.expDate = expDate;

}

//Empty Constructor

public Customer() {

this.firstName = "temp";

this.middleInitial = "temp";

this.lastName = "temp";

this.streetAdd = "temp";

this.city = "temp";

this.state = "temp";

this.zip = 0;

this.phone = "temp";

this.cardNum = 0;

this.expDate = "temp";

}

//String output of Customer object values

public String toString() {

String output = "Name: " + this.firstName + " " + this.middleInitial + ". " + this.lastName + "\n";

output += ("Address: " + this.streetAdd + ", " + this.city + ", " + this.state + ", " + this.zip + "\n");

output += ("Phone: " + this.phone) + "\n";

output += ("Card Info: " + this.cardNum + " Experation Date: (" + this.expDate + ")");

return output;

}

}

/\*

------------------------------------------------------------

- File Name: Ticket.java -

- Written By: Adam D. Seitz -

- Written On: 9/4/2019 -

------------------------------------------------------------

- File Purpose: -

- -

- This file contains the Ticket class -

------------------------------------------------------------

- Program Purpose: -

- -

- This program takes customer information as -

- input from an employee and will -

- print a receipt when the form is completely filled out. -

------------------------------------------------------------

- Local Variable Dictionary (alphabetically): -

- -

- adultPrice - Price of adult tickets -

- adultQty - Quantity of adult tickets -

- childPrice - Price of child tickets -

- childQty - Quantity of child tickets -

- name - Name of ticket category -

------------------------------------------------------------

\*/

package application;

public class Ticket {

//Class variables

public String name;

public int adultQty;

public int childQty;

public double adultPrice;

public double childPrice;

//Constructor with all variables inputed

public Ticket(String name, int adultQty, int childQty, double adultPrice, double childPrice) {

this.name = name;

this.adultQty = adultQty;

this.childQty = childQty;

this.adultPrice = adultPrice;

this.childPrice = childPrice;

}

//Constructor with 'essentials'

public Ticket(String name, double adultPrice, double childPrice) {

this.name = name;

this.adultQty = 0;

this.childQty = 0;

this.adultPrice = adultPrice;

this.childPrice = childPrice;

}

//Empty constructor

public Ticket() {

this.name = "temp";

this.adultQty = 0;

this.childQty = 0;

this.adultPrice = 0;

this.childPrice = 0;

}

//Calculate the ticket(s) price based on price per ticket and bundle

public double calcPrice() {

return (this.adultQty\*this.adultPrice) + (this.childQty\*this.childPrice);

}

//String output of Ticket object values

public String toString() {

String output = "Ticket Type: " + this.name + "\n";

output += ("Adult Price: " + this.adultPrice + "\t Adult Quantity: " + this.adultQty + "\n");

output += ("Child Price: " + this.childPrice + "\t Child Quantity: " + this.childQty + "\n");

output += ("Ticket(s) Price = $" + this.calcPrice());

return output;

}

}

/\*

------------------------------------------------------------

- File Name: Main.java -

- Written By: Adam D. Seitz -

- Written On: 9/4/2019 -

------------------------------------------------------------

- File Purpose: -

- -

- This file contains the GUI information and calculation -

------------------------------------------------------------

- Program Purpose: -

- -

- This program takes customer information as -

- input from an employee and will -

- print a receipt when the form is completely filled out. -

------------------------------------------------------------

\*/

package application;

import java.text.DecimalFormat;

import javafx.animation.PathTransition;

import javafx.animation.Timeline;

import javafx.application.Application;

import javafx.beans.value.ChangeListener;

import javafx.beans.value.ObservableValue;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.event.EventHandler;

import javafx.stage.Stage;

import javafx.util.Duration;

import javafx.geometry.Insets;

import javafx.geometry.Pos;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.CheckBox;

import javafx.scene.control.ComboBox;

import javafx.scene.control.Label;

import javafx.scene.control.Slider;

import javafx.scene.control.Tab;

import javafx.scene.control.TextField;

import javafx.scene.image.Image;

import javafx.scene.control.TabPane;

import javafx.scene.control.TabPane.TabClosingPolicy;

import javafx.scene.layout.BorderPane;

import javafx.scene.layout.GridPane;

import javafx.scene.layout.HBox;

import javafx.scene.layout.Pane;

import javafx.scene.paint.Color;

import javafx.scene.shape.Circle;

public class Main extends Application {

//Declare Variables

static DecimalFormat priceFormat = new DecimalFormat("#.##");

double miTax = 0.06;

boolean allowSave = false;

Ticket bundle = new Ticket();

Ticket raptor = new Ticket("Raptor Tickets", 80, 55);

Customer newCustomer = new Customer();

//Fields for Customer Info

Label lbCustInfo = new Label("Customer Info");

TextField fName = new TextField();

TextField mInitial = new TextField();

TextField lName = new TextField();

TextField phone = new TextField();

TextField street = new TextField();

TextField city = new TextField();

TextField state = new TextField();

TextField zip = new TextField();

TextField cardNum = new TextField();

TextField expDate = new TextField();

//Fields for Purchase Info

Label lbPurchInfo = new Label("Purchase Info");

ComboBox cbPackage = new ComboBox();

TextField packageAdultQty = new TextField();

TextField packageChildQty = new TextField();

CheckBox raptorSelect = new CheckBox();

TextField raptorAdultQty = new TextField();

TextField raptorChildQty = new TextField();

Label discountPercent = new Label("0%");

Slider discount = new Slider();

Label lbErrorNotifier = new Label();

Button btSaveInfo = new Button("Save Info");

Button btPrintReceipt = new Button("Create Receipt");

//build textfield array for exception testing

TextField[] infoArray = {fName, mInitial, lName, phone, street, city, state, zip, cardNum, expDate, packageAdultQty, packageChildQty, raptorAdultQty, raptorChildQty};

//List for combobox

private String[] packageTypes = {"Tyrannosaurs Terror","Stegosaurus Plates","Pterodactyl Droppings"};

//set up panes

GridPane mainPane = new GridPane();

TabPane tabs = new TabPane();

Tab customer = new Tab("Customer Info");

GridPane customerPane = new GridPane();

GridPane customerDetail = new GridPane();

GridPane contactInput = new GridPane();

GridPane addressInput = new GridPane();

GridPane paymentInput = new GridPane();

Tab purchase = new Tab("Purchase Info");

GridPane purchasePane = new GridPane();

GridPane ticketDetail = new GridPane();

GridPane packageDetail = new GridPane();

GridPane packageInput = new GridPane();

GridPane raptorDetail = new GridPane();

GridPane raptorInput = new GridPane();

GridPane discountInput = new GridPane();

GridPane raptorSelectPane = new GridPane();

Tab summary = new Tab("Order Summary");

GridPane summaryPane = new GridPane();

GridPane footerPane = new GridPane();

GridPane receiptPane = new GridPane();

@Override // Override the start method in the Application class

public void start(Stage primaryStage) {

//don't allow editing of raptor ticket quantities until checkbox is checked

raptorAdultQty.setDisable(true);

raptorAdultQty.setText("0");

raptorChildQty.setDisable(true);

raptorChildQty.setText("0");

//Build GUI panes and Tabs

tabs.setPrefWidth(600);

tabs.setPrefHeight(550);

//CustomerInfo Tab

lbCustInfo.setStyle("-fx-font: 24 calibri;");

customerPane.add(lbCustInfo, 0, 0);

customerPane.setVgap(10);

//Contact Info

Label lbContact = new Label("Contact");

contactInput.setHgap(10);

contactInput.add(new Label("First Name"), 0, 0);

contactInput.add(fName, 0, 1);

mInitial.setPrefWidth(30);

contactInput.add(new Label("M.I."), 1, 0);

contactInput.add(mInitial, 1, 1);

contactInput.add(new Label("Last Name"), 0, 2);

contactInput.add(lName, 0, 3);

contactInput.add(new Label("Phone #"), 0, 4);

phone.setPromptText("(555)555-5555");

contactInput.add(phone, 0, 5);

//Address Info

addressInput.setHgap(10);

addressInput.add(new Label("Street Address"), 0, 0);

addressInput.add(street, 0, 1);

addressInput.add(new Label("City"), 0, 2);

addressInput.add(city, 0, 3);

state.setPrefWidth(30);

addressInput.add(new Label("State"), 1, 2);

addressInput.add(state, 1, 3);

addressInput.add(new Label("Zip Code"), 0, 4);

addressInput.add(zip, 0, 5);

customerDetail.setHgap(50);

customerDetail.add(contactInput, 0, 0);

customerDetail.add(addressInput, 1, 0);

customerPane.setPadding(new Insets(20, 20, 20, 20));

customerPane.add(customerDetail, 0, 1);

//Payment Info

paymentInput.setHgap(20);

paymentInput.setPadding(new Insets(50, 0, 0, 0));

paymentInput.add(new Label("Card #"), 0, 0);

paymentInput.add(cardNum, 0, 1);

expDate.setPrefWidth(60);

paymentInput.add(new Label("Expiration Date"), 1, 0);

expDate.setPromptText("MM/YY");

paymentInput.add(expDate, 1, 1);

paymentInput.setAlignment(Pos.CENTER);

customerPane.add(paymentInput, 0, 2);

customer.setContent(customerPane);

customerPane.setAlignment(Pos.CENTER);

tabs.getTabs().add(customer);

//PurchaseInfo Tab

lbPurchInfo.setStyle("-fx-font: 24 calibri;");

purchasePane.add(lbPurchInfo, 0, 0);

purchasePane.setVgap(10);

//Combo Box with package select

packageInput.setHgap(20);

packageInput.setVgap(5);

cbPackage.setPrefWidth(200);

cbPackage.setValue("Pick a Package Type");

//Add options to Combo Box

ObservableList<String> items =

FXCollections.observableArrayList(packageTypes);

cbPackage.getItems().addAll(items);

//Package Info

packageInput.add(cbPackage, 0, 0);

packageDetail.setHgap(10);

packageDetail.add(new Label("# of Adults"), 0, 0);

packageAdultQty.setPrefWidth(95);

packageDetail.add(packageAdultQty, 0, 1);

packageDetail.add(new Label("# of Children"), 1, 0);

packageChildQty.setPrefWidth(95);

packageDetail.add(packageChildQty, 1, 1);

packageInput.add(packageDetail, 0, 1);

purchasePane.setPadding(new Insets(20, 20, 20, 20));

ticketDetail.setHgap(50);

ticketDetail.add(packageInput, 0, 0);

//Raptor Info

raptorDetail.setHgap(10);

raptorInput.setVgap(13);

raptorSelectPane.add(new Label("Rappin' With Raptors: "), 0, 0);

//only allow typing in qty boxes when checkbox is selected

raptorSelectPane.add(raptorSelect, 1, 0);

raptorInput.add(raptorSelectPane, 0, 0);

raptorSelectPane.setAlignment(Pos.CENTER);

raptorDetail.add(new Label("# of Adults"), 0, 0);

raptorAdultQty.setPrefWidth(95);

raptorDetail.add(raptorAdultQty, 0, 1);

raptorDetail.add(new Label("# of Children"), 1, 0);

raptorChildQty.setPrefWidth(95);

raptorDetail.add(raptorChildQty, 1, 1);

raptorInput.add(raptorDetail, 0, 1);

ticketDetail.add(raptorInput, 1, 0);

purchasePane.add(ticketDetail, 0, 1);

//discount slider and textbox work interchangeably

discountInput.setPadding(new Insets(50, 0, 0, 0));

discountInput.add(new Label("Discount %: "), 0, 0);

discountInput.add(discount, 0, 1);

discountPercent.setPrefWidth(40);

discountInput.add(discountPercent, 1, 1);

discountInput.setAlignment(Pos.CENTER);

purchasePane.add(discountInput, 0, 2);

purchase.setContent(purchasePane);

purchasePane.setAlignment(Pos.CENTER);

tabs.getTabs().add(purchase);

//Summary Tab

tabs.getTabs().add(summary);

summaryPane.setAlignment(Pos.CENTER);

tabs.setTabClosingPolicy(TabPane.TabClosingPolicy.UNAVAILABLE);

mainPane.add(tabs, 0, 0);

mainPane.add(lbErrorNotifier, 0, 1);

//Footer with save and receipt buttons

footerPane.add(btSaveInfo, 0, 0);

footerPane.add(btPrintReceipt, 1, 0);

mainPane.add(footerPane, 0, 2);

footerPane.setAlignment(Pos.CENTER);

// Create a scene and place it in the stage

Scene appScene = new Scene(mainPane, 600, 550);

mainPane.setAlignment(Pos.CENTER);

primaryStage.setTitle("Jurassic Park Travel Limited - Ticket Purchasing System");

primaryStage.getIcons().add(new Image(Main.class.getResourceAsStream( "/JPlimitedTravelico.png" )));

primaryStage.setScene(appScene);

// Display the stage

primaryStage.show();

//Set up 'secondaryStage' for receipt window

Stage secondaryStage = new Stage();

Scene receiptScene = new Scene(receiptPane, 350, 500);

secondaryStage.setTitle("Order Receipt");

secondaryStage.getIcons().add(new Image(Main.class.getResourceAsStream( "/JPlimitedTravelico.png" )));

//enable and/or disable Raptor tickets with checkbox

raptorSelect.setOnMouseClicked(e -> {

if(raptorSelect.isSelected()) {

raptorAdultQty.setDisable(false);

raptorChildQty.setDisable(false);

}

else {

raptorAdultQty.setDisable(true);

raptorAdultQty.setText("0");

raptorChildQty.setDisable(true);

raptorChildQty.setText("0");

}

});

//connect discount textFeild and slider and save to variable

discount.valueProperty().addListener(new ChangeListener<Object>() {

@Override

public void changed(ObservableValue arg0, Object arg1, Object arg2) {

discountPercent.textProperty().setValue(String.valueOf((int)discount.getValue()) + "%");

}

});

//After clicking a tab, all entered info is saved to variables

btSaveInfo.setOnMouseClicked(e -> {

//check all textfields before storing data

for(int i=0; i < infoArray.length; i++) {

if(infoArray[i].getText().isEmpty()) {

allowSave = false;

break;

}

else {

allowSave = true;

}

}

//Detect which package was chose and set ticket price

if(cbPackage.getValue() == "Pick a Package Type") {

allowSave = false;

}

else {

switch (cbPackage.getValue().toString()) {

case "Tyrannosaurs Terror":

bundle.name = "Tyrannosaurs Terror";

bundle.adultPrice = 219.75;

bundle.childPrice = 195.50;

break;

case "Stegosaurus Plates":

bundle.name = "Stegosaurus Plates";

bundle.adultPrice = 155.00;

bundle.childPrice = 120.25;

break;

case "Pterodactyl Droppings":

bundle.name = "Pterodactyl Droppings";

bundle.adultPrice = 53.00;

bundle.childPrice = 26.50;

break;

}

}

//Only allow save of data when all fields are filled in

if(allowSave == true) {

System.out.println("Allow save. All fields are filled in.");

lbErrorNotifier.setText("");

//Store all info fields into objects

newCustomer.firstName= fName.getText();

newCustomer.middleInitial = mInitial.getText();

newCustomer.lastName = lName.getText();

newCustomer.streetAdd = street.getText();

newCustomer.city = city.getText();

newCustomer.state = state.getText();

newCustomer.zip = Integer.parseInt(zip.getText());

newCustomer.phone = phone.getText();

newCustomer.cardNum = Integer.parseInt(cardNum.getText());

newCustomer.expDate = expDate.getText();

bundle.adultQty = Integer.parseInt(packageAdultQty.getText());

bundle.childQty = Integer.parseInt(packageChildQty.getText());

raptor.adultQty = Integer.parseInt(raptorAdultQty.getText());

raptor.childQty = Integer.parseInt(raptorChildQty.getText());

//print info to console

System.out.println("\n" + newCustomer.toString());

System.out.println("\n" + bundle.toString());

System.out.println("\n" + raptor.toString());

System.out.println("\nTotal Price: $" + priceFormat.format(calcTotal(bundle, raptor, miTax, ((int)discount.getValue()))));

//All current info is printed onto the Summary Tab

saveToSummary(summary, summaryPane, newCustomer, bundle, raptor, miTax, ((int)discount.getValue()));

lbErrorNotifier.setText("");

System.out.println("Info Saved!");

lbErrorNotifier.setTextFill(Color.GREEN);

lbErrorNotifier.setText("Info Saved!");

}

else {

lbErrorNotifier.setTextFill(Color.RED);

lbErrorNotifier.setText("Fill in all fields before saving!");

}

});

//After Clicking on "Print Receipt" a new window is opened with all customer and purchase info shown

btPrintReceipt.setOnMouseClicked(e -> {

receiptPane.getChildren().clear();

if(allowSave == true) {

receiptPane.add(new Label("\n" + newCustomer.toString()), 0, 0);

receiptPane.add(new Label("\n" + bundle.toString()), 0, 1);

receiptPane.add(new Label("\n" + raptor.toString()), 0, 2);

receiptPane.add(new Label("\nDiscount: " + (int)discount.getValue() + "%"), 0, 3);

receiptPane.add(new Label("\nTotal Price: $" + priceFormat.format(calcTotal(bundle, raptor, miTax, (int)discount.getValue()))), 0, 5);

secondaryStage.setScene(receiptScene);

secondaryStage.show(); //show only when print receipt is clicked

}

else {

lbErrorNotifier.setTextFill(Color.RED);

lbErrorNotifier.setText("Fill in all fields before creating receipt!");

}

});

}

//Calculate the total price after discount and taxes

public static double calcTotal(Ticket bundle, Ticket raptor, double tax, double discount) {

return (1+tax)\*(1-discount/100)\*(bundle.calcPrice() + raptor.calcPrice());

}

//print info to summary tab

public static void saveToSummary(Tab summary, GridPane summaryPane, Customer newCustomer, Ticket bundle, Ticket raptor, double miTax, int discount) {

summaryPane.getChildren().clear();

summaryPane.add(new Label("\n" + newCustomer.toString()), 0, 0);

summaryPane.add(new Label("\n" + bundle.toString()), 0, 1);

summaryPane.add(new Label("\n" + raptor.toString()), 0, 2);

summaryPane.add(new Label("\nDiscount: " + discount + "%"), 0, 3);

summaryPane.add(new Label("\nTotal Price: $" + priceFormat.format(calcTotal(bundle, raptor, miTax, discount))), 0, 5);

summary.setContent(summaryPane);

}

//Main method to start the GUI application

public static void main(String[] args) {

launch(args);

}

}