CS106 Project Report

By Nikola Jocic

Instructor: Dr. Gregory Baglavas
American College of Thessaloniki
Date: 31/03/2021

OUTLINE

- Variables
- Part A
    - Constructor
    - Interface
    - Action Listener
        - Check variables and how they work
- Window Quitter and Main Method

Variables

Class PROJECTMORE
Variables

Name	Description	Type	Inital Value
random1	Stores the value of the first random generator	int	N/A
random2;	Stores the value of the second random generator	Int	N/A
check;	Checks the number of the round so that the hidden dice in border panel can appear	Int	N/A
sum;	Stores the sum of radom1 and random2	Int	N/A
check2;	Checks the number of the round so that the first turn win/loss can be determined	Int	N/A
diceNumber1;	Stores the path of the image and the converted to String value of radnom1	String	N/A
diceNumber2;	Stores the path of the image and the converted to String value of radnom2	String	N/A
point;	Stores the sum of the first roll	Int	N/A
roundCount;	Counts the number of the round so that it can be subtracted from 10 to determine the score later in the game	Int	N/A
score;	Stores the score value	Int	N/A
textScoreLabel;	Stores the text to fill the Score label later	String	N/A

JElements

Name	Description	Type
play	The play button	JButton
roll;	The roll button	JButton
quit;	The quit button	JButton
rules;	The rules button	JButton
showResult;	Text field that shows whether the player has won or lost	JTextField
insertPlayerCount;	Text field where the player inserts the amount of additional players they want to play with	JTextField
Score;	The label which displays the score	JLabel
result;	The label above the showResult textbox saying 'Result: '	JLabel
dice1;	The label for dice1 which will store the die Icon	JLabel
dice2;	The label for dice2 which will store the die Icon	JLabel
diceHidden1;	The label for diceHidden1 which will store the die Icon and appear from the 2nd round on showing the previous dice	JLabel
diceHidden2;	The label for diceHidden1 which will store the die Icon and appear from the 2nd round on showing the previous dice	JLabel
radioButton;	Label that is located next to the actual radio buttons describing that the player can chose beteen playing with or without the computer	JLabel
playerCount;	Label located next to the insertPlayrCount field that tells the player that they can insert the amount of additional players they want to play with	JLabel

	into the field adjecent to this layer	
diceImage1;	Stores the Image Icon	ImageIcon
diceImage2;	Stores the Image Icon	ImageIcon
yes;	Raddio button yes that indicates the player wishes to play with the computer	JRadioButton
no;	Raddio button yes that indicates the player does not wish to play with the computer	JRadioButton

JPanels

Name	Description
panel_Play;	Panel of the play button
panel_Roll;	Panel of the roll button
panel_Controls;	Panel of the play and roll button
panel_Score;	Panel of the score label
panel_quit;	Panel of the quit button
panel_rules;	Panel of the rules button
panel_result;	Panel of the result label
panel_titleBorder;	Panel of the hiddenDice1/2 labels
panel_left;	Panel of the whole left side of the GUI (panel_titleBorder + panel_controlsAndDice)
panel_right;	Panel of the whole right side of the GUI (panel_quitRulesResult + panel_Score)
panel_dice;	Panel of the dice labels
panel_quitRulesResult;	Panel that contains the panel_quit, panel_rules, panel_result and panel_showResult (the bottom right side of the interface)
panel_showResult;	Panel of the showResult text box
panel_players;	Panel of the playerCount label and insertPlayerCount text box
panel_computerButton;	Panel of the radioButton label and yes and no radio buttons
panel_playerCount;	Panel that containts panel_players and panel_computerButton
panel_upperControls;	Panel that contains all the left upper elements of the GUI into one panel (panel_dice + panel_Controls + panel_playerCount
panel_controlsAndDice;	Panel that puts the panel_uppperControls in oreder with the box layour manager

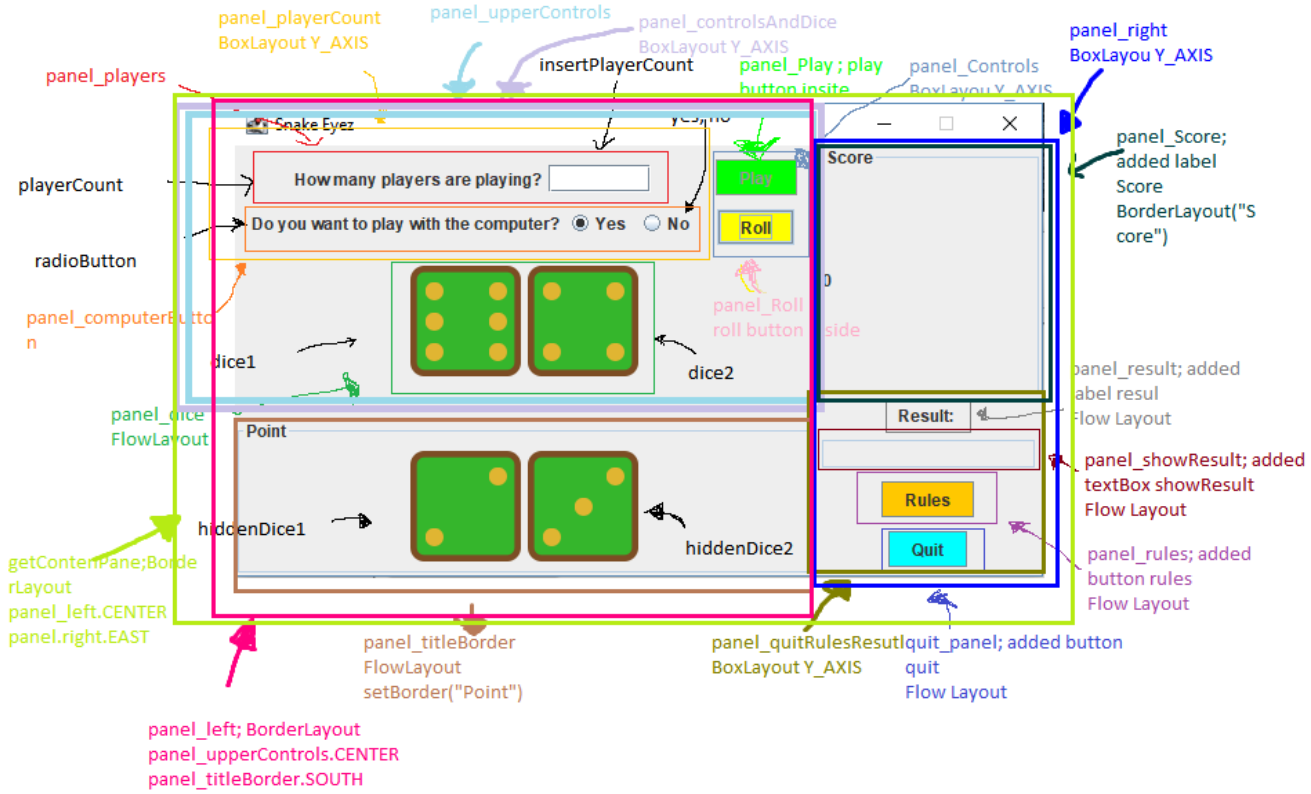
Part A

Constructor

In the constructor, the variables are being assigned their initial values. I have chosen the appropriate sizes for the textboxes(showResult, insertPlayerCount) corresponding to the size of the GUI. The text box showResult, is there to represent the result which the player get after rolling the dice (win or lose). The textbox, insertPlayerCount, is there so that the player can type in the amount of additional players that he wants to play with. The labels are filled with the appropriate, initial text which explains what they are used for and can be recognized on the interface. The labels playerCount and radioButton are the two labels which are being displayed at the top left of the screen. The label playerCount is located right next to the insertPlayerCount text box indicating to the player what the field is used for. The label radioButton, is there to let the player know that they have a choice of playing with or without the computer. At the beginning of the constructor, the title is being set as well. Some of the radio buttons were arranged to look as in the picture below and certain text boxes are left with being editable while others are not (and are left in that way for the duration of the game). In the beginning the roll

button (roll) is the only button which is disabled so that the player first has to click the play button (play), more of this in the Action Listener section. The buttons ‘yes’ and ‘no’ are radio buttons which are located right next to the label radioButton, which are used to indicate whether a player wishes to play with a computer or not. Labels dice1/2 and diceHidden1/2 are having their visibility set to hidden so that the player sees them on the corresponding turn. These labels also have initial values of diceImage1/2 set to them. Later in the action listeners, those values will change. The colours of the buttons are also set after defining all the JPanels. The naming and what each JPanel does is explained in the following section, “Interface”.

Interface



The above schematic and legend is the one of the Interface. The coloured letters are for the panels while the black ones are for ordinary elements (labels, buttons etc.). For quicker interpretation, panel_players (label playerCount, textBox insertPlayerCount) is being put into the box layout Y_AXIS panel_playerCount with panel_ComputerButton. Next to them, ‘play’ and ‘roll’ panels are being put into individual panels (panel_Play, panel_Roll, respectively), before being put into panel_Controls with a box layout Y_AXIS. The two dice underneath (dice1, dice2) are being put into panel_dice with a flow layout. All of these 3 panels are being put into panel_upperControls, and then another panel, panel_controlsAndDice with the box layout Y_AXIS. Moving on to the right side of the interface, we have, at the bottom, label result in the

flow layout panel_result, textBox showResult in flow layout panel_ showResult, buttons rules and quit in their own flow layout panels, panel_rules and panel_quit respectively. These four panels are being put into one bigger panel_quitRulesResult with a box layout Y_AXIS. Above them we have a simple panel_Score with a title property and a border layout inside of which is a label Score. Finally the panel_right assembles these 2 major panels together into one, putting panel_Score on top of panel_quitRulesResult with box layout Y_AXIS. At the left bottom of the interface another bordered panel is located panel_titleBorder with a flow layout manager, inside of which are the two hiddenDice labels and a border property. The left side of the interface is put together into panel_left that has a border layout manager. Inside, panel_upperControls.CENTER and titleBorder.SOUTH locations. Finally both sides (left and right) are put onto the ContentPane together with another border style layout manager (panel_left.CENTER ; panel_right.EAST)

Action Listener

Before going into the Action Listener, it is worth noting that the play button is the only available button at the start of the application. Once it is clicked, the action listener is triggered for the first time of the game, at which point it becomes disabled but the roll button becomes enabled. Moving on to the 'roll' Action Listener before explaining the logic behind the two variables 'check' and 'check2' the explanation will start from the random generator. There are two different random generators which are storing the value into the random1 and random2. These two values are being added together into one sum to later be checked in the if clauses for loses and wins. The variable diceNumber1 and diceNumber2 are used to store the path later to be used to put the images into icons and then into their respective labels. The variables diceNumber1 and 2 are being stored with the String version of the random1 and 2 respectively, before they are being stored with the 'path' to the images. The way that the program works is that whatever number random generators come up with, that number is going to be added into the path "Dice " + diceNumber1/2 (number being added into here) + ".png" so that the numbers correspond automatically with the pictures. After storing the icons into labels dice1 and dice2 the same are being set as visible now. The roundCount++ is used to count the number of rounds that is later to be subtracted, in part B, from number 10 in order to calculate the points which should be awarded to the player.

Check Variables and How They Work

Since there are certain commands which only happen on the first turn and some only after the second move and forward on, the variables 'check' and 'check2' are used to regulate those events. First and foremost, it is important to point out that the values which are being assigned to these variables are completely arbitrary. The values do not represent the number of rounds or anything else for that matter and are a simple way of controlling when something happens and when it does not. Starting from the variable check, it is assigned the value 1 in the middle of the 'roll' action event. Looking at the beginning of the event, there is an if clause inspecting whether

the ‘check’ is equal to one, however on the press of the first button the check starts with the value of 0 and is only being assigned the value of 1 after that if clause. The reason for this is that the hidden dice which are located in the Point title border, are supposed to appear on once the roll button has been pressed for the second time and those said dice are supposed to take the the icons and pictures of the regular labels dice1 and dice2 to show the players what they had rolled in their previous turn. So at this moment, the variable check always has the value 1 and therefore, starting from the second press of the roll button, the if clause will always inspect for true and be fulfilled, setting the labels diceHidden1 and diceHidden2 with the icons of the dice1 and dice2 from the previous turn as well as setting the said labels visibility to be true. Moving on to the variable ‘check2’. At the beginning of the ‘roll’ action, the ‘check2’ is being set to 1 (Note at this point, since this is the first press of the roll button, the if clause (check==1) is still not true). Then, at the end of the ‘roll’ action, there are a series of if statements, checking whether the player has rolled a winning or losing sum on their first roll. If one of these two statements is true, the game stops with the player winning/losing, the roll button disables and the points are being awarded in response to the result. On the next press of the roll button, the action starts out by setting the check2=1, however, because this is already the second press onto the roll button now, the if statement (check==1) is true, and at the end of the statement the check2 is set to the value of 2, and since, from now on, the if statement (check==1) will always be true, the value of check2 will always be set to 2 before the if clause (check2==1), meaning that from the 2nd press on the button roll and going forward, the if statement (check2==1) will never be true again. The purpose of this is to check if the sum on the first, and first roll only is 7, 11 (wining) or 2,3,12 (losing). If none of those things is true in the first role, there is an else statement which records and stores the value of the sum of the first role, into the point. Later on, the else if statements corresponding to the if statement (check2==1), are being put to check where the point variable is compared to the current sum to check if the player is to win, or to check if that same current sum is equal to 7, in which case the player loses. In any case, at the end of the if statements that check whether the player has won or lost, at the end, the text of the play button will be changed to ‘Again’ and set to enabled, indicating that the player may start another round immediately should they wish to do so. In such case, the ‘play’ action is called upon again which not only sets the roll button to enabled again, but also makes all the labels with dice images invisible as well as resets the score, the ‘check’ variables, the sum and the random variables, the roundCount variable as well as the point variable and additionally, resetes the shoResult text box.
The last action event is “quit” which, as its name implies, quits the game.

Window Quitter and Main Method

Window Quitter is an additional class attached to the main class that is PROJECTMORE. The way it works is that it allows the user to exit the application with the click on the red X button in the top right corner without the application running in the background when the frame is invisible. Besides the Window Quitter being attached to the main method, the frame is also being defined in the main method with the size of it also being determined there. The size of the frame was picked out so that all the elements would fit onto it as displaced above in the picture of the interface.