

On the page above, I have displayed the basic prototype for my frame. It contains the labels and button that I included in my frame as well as the layout I decided to use. The layout is done with a box layout which makes everything added to the panel appear beneath one another. I made the logo in the Competition Frame class. I made the jpeg file into an image icon and then initialised it into a label I called imageLabel. My title is made using a simple JLabel. My rules button is made from a JButton that has its own class in order to function the way it does. I have a simple label for the line of text stating where the competition will take place. I have 3 separate button for registering a school, a group, and withdrawing a school, all of which have their own functions to work. I have a text field which takes in any input as applies a 10% discount to the total price owed by the school when the ‘Apply voucher’ button is pressed. I have 4 different labels which are affected by the 3 button to register and withdraw. Finally, I have an exit button which has its own listen and action listener that closes the application when clicked.

The first even that takes place in my frame is the Rules button. The button displays the rules when clicked and hides them when clicked again. I achieved this by having a variable called countRules. When the button is pressed, the value of countRules is increased by 1. I have an if-statement for if the modulus of countRules is not equal to 0, then the visibility of the lvbRules variable is set to true and the rules can be seen. Else, the visibility is set to false. Next, my register school button has its own function. When the button is pressed, it increases the amount of schools entered in the competition by 1, it increased the price owed by the school by 20, and it enables the ‘register group’ and ‘withdraw’ buttons, which are disabled by default. Next is my ‘register group’ button. This button also works off of a separate ‘count’ variable that increases by 1 when the button is pressed. I have an if-statement where if the count variable is equal to 1, then the overall group count and the ‘groups entered by your school’ increases by 1. If the count is greater than or equal to 2, then the same changes are applied, except it increases the price owed by the school by 5. This is how I made the first group entered to be free, and every subsequent group costs 5 euros. The ‘withdraw’ button is quite simple. When pressed, it sets the group and school count back to 0 as well as the price owed by the school. It also disables the withdraw button and the ‘register group’ button so the user is essentially back to square one. Next Is my voucher function. When the ‘apply voucher’ button is pressed by itself, nothing happens. When text is entered into the text field below it, the text in the text field changes to "Code Enter Successfully! 10% Discount Applied", and the price owed by the school is reduced by 10%. This then disables the voucher button. The final event in my frame is my exit button. This button is unique because it has its own separate files for the listen and action listener. When the button is pressed, the listener reads the line of code “System.exit(0);” which simply closes the application.