Java Code Generator

## Purpose:

The Basic Purpose of this Applet is to Generate Basic Structural Code of Components of Java. As this Applet is in its initial Stage and is not a deployed Applet so it has only limited number of Concepts and Components, of which it creates the Source Code.

The Backbone of this Applet is File-Handling and the Template Files that the Developer has already created. To keep the Applet simple, we created The Template Files using File-Handling, so that it is Easy and Fast to fetch and Store data on runtime.

## Technology:

**JavaFX**

JavaFX is Used in this Applet as JavaFX is more vast, flexible, Compatible, Reliable and Supports some major functions that Swing or AWT lack. It also gives to so many options of Customization.

JavaFX is so far the Best Framework of Java to Create Real-Time and Real-Life Applications.

## How to Use?

This Applet Works in a very User-Friendly way. After Starting the Application the Applet Ask the user to Enter the Name of the File to be Generated and ask to set a Location to store the file. Following this we Reach to a Java Component Page. Where we have a Tabbed Pane to Categorize some of the Features of the Java Components/Concepts. After Checking the Desired Concepts to be put into the File, we reach a Final Screen which ask us for confirmation. After Confirmation, the Applet Generate the .JAVA file at the given Location.

## How it Works?

When you start the Applet, the main is executed and Applet is initialized by the initializer function, the initializer then calls the Controller and in that it looks for the Screen to make visible, after that it waits and listen to the Actions to be performed by the buttons and if buttons are clicked then it gets the actions and validate them, if action is invalid then it calls the alert box from alert class. If the action is valid then it checks for the next controller to be called, after that it loads that controller and again waits and start listening and this is where we may select the checkboxes and other options, after that it doesn’t validate them as it did before, because on the first two screens there were no default values set for those actions, but at this java Concept/Component selection screen, there are already Default Templates generated. So, if no option is selected, then from the controller the function of file-handling is called at default and a simple file with the given name {Class} is created. But when any options are selected the Applet listen and call their respective functions from File-Handling class to Execute them. So for every action there is a respective function in File-Handling class which gets the selected parameters from the controller of the screen and store them into a buffer.

There is already a Buffer which had been initialized when file-handling function were called, this Applet had a check which selected the respective Template from the Template Folder and this Buffer stored all of the data from that file to itself, Now when the selections from the Controller reach the function, the new Buffer initialized contact the already running buffer and make changes to it according to controller needs.

This process happens in every respective function which were called by Controller and at the end all the Streams and Buffers are closed and the file is saved on the Target Location by the Given Name with a .JAVA extension.