**Table of Contents**

[i](#_Toc514106013)

[Acknowledgement ii](#_Toc514106016)

[Table of Figures iv](#_Toc514106017)

[1. INTRODUCTION 1](#_Toc514106018)

[2. OBJECTIVE 3](#_Toc514106019)

[3.01 DESIGN AND FLOW ROUTINES 4](#_Toc514106020)

[3.02 FLOWCHART DESIGN 6](#_Toc514106022)

[EXPLANATION 11](#_Toc514106026)

[4.0 FORMAT AND STYLE ROUTINE 13](#_Toc514106027)

[4.01 Inbuilt properties. 13](#_Toc514106028)

[4.02 Initializing the input variables. 14](#_Toc514106029)

[4.03 PLAYNOTE (main) method. 15](#_Toc514106030)

[4.04 Play Reset and Notes method 18](#_Toc514106032)

[4.05 Duration of note method 19](#_Toc514106033)

[4.06 SAVEAS method 20](#_Toc514106034)

[5. RESULT AND ANALYSIS 20](#_Toc514106035)

[7. DISCUSSION 24](#_Toc514106039)

[7. CONCLUSION 25](#_Toc514106040)

[REFRENCES 25](#_Toc514106041)

[APPENDIX 27](#_Toc514106042)

# Table of Figures

[Figure 1 ANALYZING A NODE 1](#_Toc514105644)

[Figure 2 Sound processing MATLAB. 2](#_Toc514105645)

[Figure 3 a) Square wave b) Sine wave c) Sawtooth wave 3](#_Toc514105646)

[Figure 4 Main design 4](#_Toc514105647)

[Figure 5 Flowchart Informal 1 6](#_Toc514105648)

[Figure 6 Flowchart Informal 2 7](#_Toc514105649)

[Figure 7 Flowchart Informal 3 8](#_Toc514105650)

[Figure 8 Program formal Flowchart 1 9](#_Toc514105651)

[Figure 9 Program formal Flowchart 2 10](#_Toc514105652)

[Figure 10 Program formal Flowchart 3 11](#_Toc514105653)

[Figure 11Type of waveform 12](#_Toc514105654)

[Figure 12 Duration of a note 12](#_Toc514105655)

[Figure 13 Notes selection 12](#_Toc514105656)

[Figure 14 Record function 13](#_Toc514105657)

[Figure 15 Auto update code 13](#_Toc514105658)

[Figure 16 User defined Variable initialization 14](#_Toc514105659)

[Figure 17 Play note function variable initialization 15](#_Toc514105660)

[Figure 18 PLAYNOTE method waveform function 16](#_Toc514105661)

[Figure 19 PLAYNOTE method plot function 16](#_Toc514105662)

[Figure 20 PLAYNOTE method record 17](#_Toc514105663)

[Figure 21 play reset and note methods 18](#_Toc514105664)

[Figure 22 Duration of note method 19](#_Toc514105665)

[Figure 23 SAVEAS method. 20](#_Toc514105666)

[Figure 24 MAIN PANEL 21](#_Toc514105667)

[Figure 25 SINE waveform 21](#_Toc514105668)

[Figure 26 B4 note pressed in sine waveform 22](#_Toc514105669)

[Figure 27 B4 note of square waveform 22](#_Toc514105670)

[Figure 28 B4 note of sawtooth waveform 23](#_Toc514105671)

[Figure 29 Filename input 23](#_Toc514105672)

[Figure 30 Output record 23](#_Toc514105673)

# 1. INTRODUCTION

Music is as important as any other activities for any human daily life for someone it relax and for someone it may even heal and even make someone more focused and energetic towards their work. Music is such a big branch that there is so many respective fields of research in philosophy and engineering which deals with music. Oxford dictionary defined music as “the art of combining vocal or instrumental sounds (or both) to produce beauty of form, harmony, and expression of emotion”. Which is true but this is good for common understanding of music but is there any definition of music in engineering or in science is still a debate but the most famous and detailed definition of music is Music is an art form and cultural activity whose medium is sound organized in time. The common elements of music are pitch (which governs melody and harmony), rhythm (and its associated concepts tempo, meter, and articulation), and dynamics (loudness and softness).

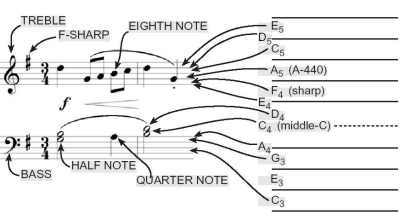


Figure 1 ANALYZING A NODE

The famous scientist Pythagoras was often considered as the father of geometry and music too because it was him who first realized that dividing a vibrating string in half will result in creating an octave of the original note. Sine or Cosine wave is often considered as the purest and most beautiful of all the oscillations or vibrations.

Analyzing a sound or music is quite fascinating and complex thing in the field of physics and engineering there are different tools available to be to research in the field of sound but among the best and greatest tool ever made for analyzing signals and sound is MATLAB.

MATLAB is programming environment which runs MATLAB language which is a matrix-based language designed specifically for engineers and scientist. MATLAB is used in numerical computation, programming and visualizations whereby a user can analyze data, develop algorithms and also to create models and applications.

MATLAB is used in wide variety of field including sound and signal processing (Figure 2), image and video processing, control systems, and many other computational tasks.

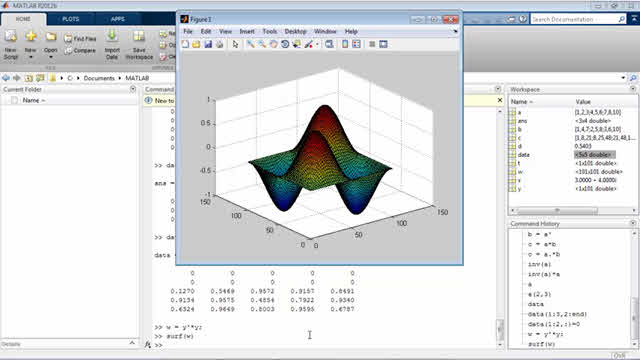


Figure 2 Sound processing MATLAB.

MATLAB has evolved a lot and one of the most important function of the MATLAB is its graphical user interface features which allows developers to analyze and develop complex task using GUIDE (Graphical user interface development guide) and recently developed App Designer which is much simpler and features packed than GUIDE.

MATLAB GUI (graphical user interface) allows developers to interact and communicate with electronic devices through graphical icons and indicators. MATLAB GUI have one function which is used most times by any MATLAB developer which is GRAPH(plot) function.

In this assignment MATLAB have been used to develop a Graphical user interface program which is able to compose a music melody. Some of the concept used in making of the design are: -

1. Notes:-In music world notes is what we play and what we hear or it is the duration and pitch of a sound. There are seven or eight notes in western world there are seven notes starting from A to G where as in Indian music and many other there are eight notes but this assignment follows the worldwide acceptance of notes which is from A to G. Notes itself is a field which is very key and basic for music related filed researcher but in basic every type of notes from A to G has its own frequency which results in difference type of sound and Oscillations.
2. Waveform:-In Sound’s world everything that comes out result in vibration and that vibration is what we hear. The vibration keeps on disturbing the upcoming particle which In results makes a wave known as sound wave in which SIN WAVE(Figure 3b) is the purest wave which contains only fundamental frequency , SQUARE WAVE(Figure 3a)is a type of wave in which the waveform rises to a particular level and then stay there and then instantly drops to another level and then rises to its original level results in completion of one wave cycle, whereas a SAWTOOTH WAVEFORM(Figure 3c) rises from zero value to a peak value and then quickly drops to a zero value for each cycle.

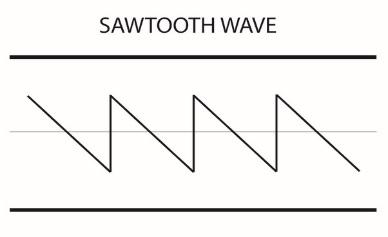
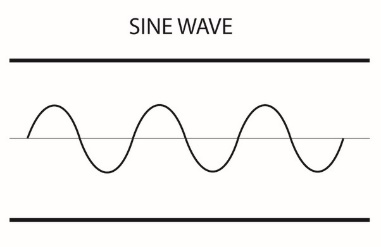
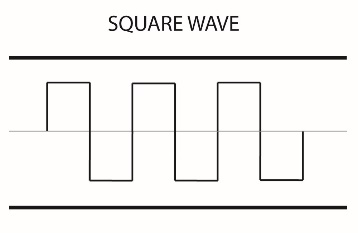


Figure 3 a) Square wave b) Sine wave c) Sawtooth wave

1. Duration of a note: - Duration of a note is just a time span for how long every note must need to be played, as seen in music each note is played for different time span depends upon the music composition which makes it more satisfactory so keeping this in mind this feature been packed in the program.
2. Graph functionality: -For engineers or scientist a graph is the most important data that they can retrieve from any information so this program also uses three different type of graph one for plotting the frequency which is played one for Fourier transformation of the note one is the noise which is generated when a frequency is played.
3. Basic functions: -Record and Output to .wav file functionality been added to let user listen to the music they created.

# 2. OBJECTIVE

To develop a MATLAB Graphic User Interface which is able to compose a simple music melody.

# 3.01 DESIGN AND FLOW ROUTINES

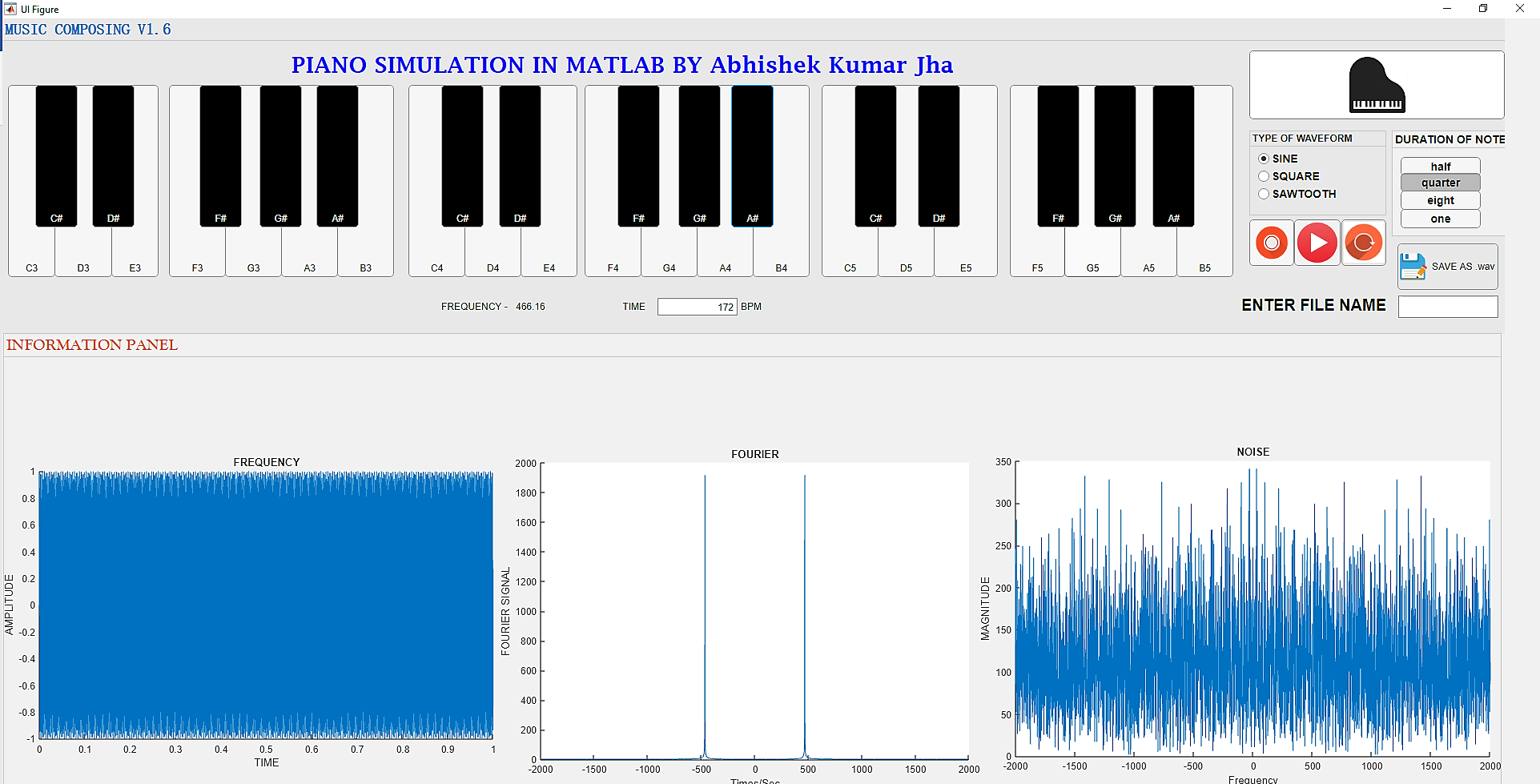


Figure 4 Main design

# The design or GUI has been done in APP DESIGNER using all the below functionality: -

1. Buttons: - All the 36 notes available in the program were designed using Buttons. Buttons are the user interface components of MATLAB that respond’s the value when the user presses or releases them
2. Statebutton: - Three State buttons been used for record play and reset function. Statebutton is same like button it’s just indicates the logical state for eg:- when the user selected the record button it will keep recording unless the user clicked it again holding the state until the next state.
3. ToogleButton:-For the Duration of the note design ToogleButton been used which are a set of options inside a button group where user chooses one from that set of button group.
4. UIAxes: -All the three types of graph use UIAxes which works similar as axes in GUIDE which represents a y and x axis and helps to plot the data required by the program it can also plot 3d data.
5. NumericEditField: - For the Tempo or BPM NumericEditField properties been used in the design. This property allows users to type numeric value in the required field. And as BPM will be in numeric format so this property been used.
6. EditField: -For the filename which will be provided by the user EditField property been used. EditField property works as same as the NumericEditField just this accepts the values in string format.
7. Label: - Label are used for displaying a static text in the design or labelling the desired parts in an app and it’s use in the design are overly. wherever there was the requirement of labeling this AppDesigner property been used.
8. Button Group: - For type of waveform in the design this property been used. Button Group are containers for managing exclusive of radio buttons and toggle buttons.

# 3.02 FLOWCHART DESIGN

# 

Figure 5 Flowchart Informal 1

# 

Figure 6 Flowchart Informal 2

# 

Figure 7 Flowchart Informal 3

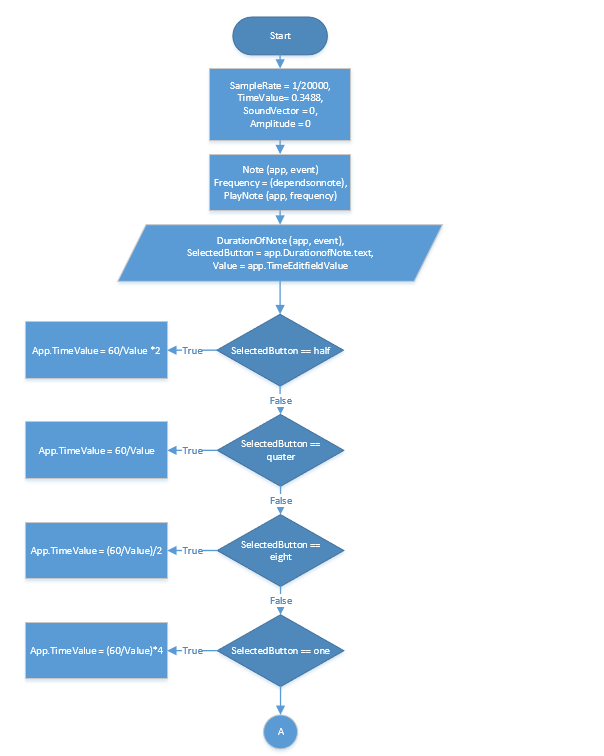


Figure 8 Program formal Flowchart 1



Figure 9 Program formal Flowchart 2

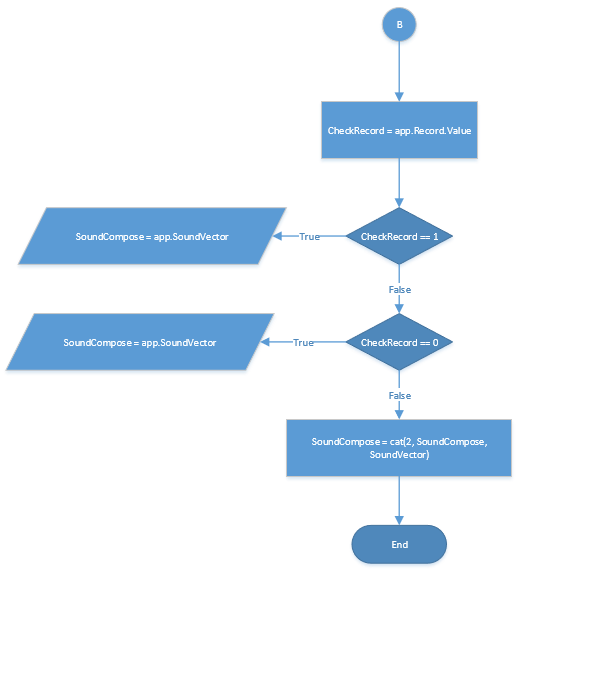


Figure 10 Program formal Flowchart 3

# EXPLANATION

This program runs on the basic of realization of how a musician makes a music through piano, which if rolls down to some basic steps will be:

1. User select the type of sound(waveform) he wants to play.

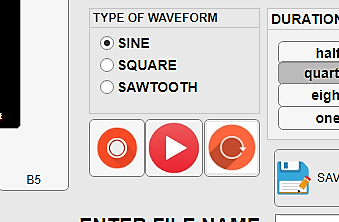


Figure 11Type of waveform

User selects among the three different type of waveforms provided in the program using Radio button group in App designer depends on the type and unless they select different waveform the same waveform will be sound for every note and by default SINE waveform is set.

1. User select the duration of the note (or the amount of time he wants to play the particular note).

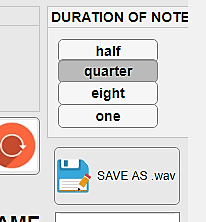


Figure 12 Duration of a note

This works same as the waveform its just use the Toggle Button group function to let user select the amount of duration of note they want and by default it’s set to quarter

1. User select the note and then sound been produced.

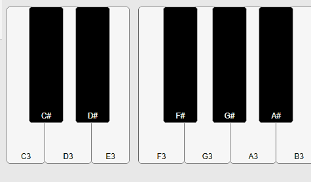


Figure 13 Notes selection

This is basically like the input of the program where user select the notes which made by Button’s in App designer and this works same like real piano instrument user just press the note, there are total 36 notes available for user to play.

1. User can record the whole music he wants to play and save the file.

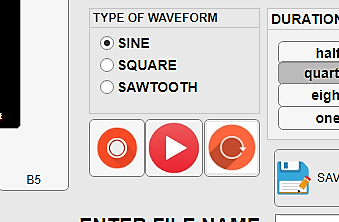


Figure 14 Record function

# 4.0 FORMAT AND STYLE ROUTINE

Format and style routine refers to the way in which the MATLAB code is arranged or set out (Palm, 2011). For this assignment, different styles and format were applied to enable the MATLAB program to run efficiently and up to standard.

### 4.01 Inbuilt properties.

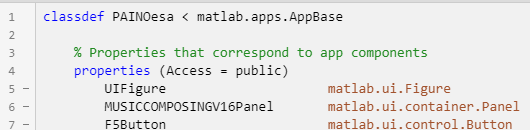


Figure 15 Auto update code

App designer automatically generate code that corresponds to the app components which been selected by the programmer to develop and all the codes are under one properties which automatically set the Access specifier to public and the all the component’s used in the app which be inside the structure of this properties and any change in the component will results in the auto update of the code.

Classdef is basically the definition of the class of the program but it defines the same properties for all the Object in the program whereas every Object can have unique data values. Properties helps to allocate the control of which methods can access the property.

### 4.02 Initializing the input variables.

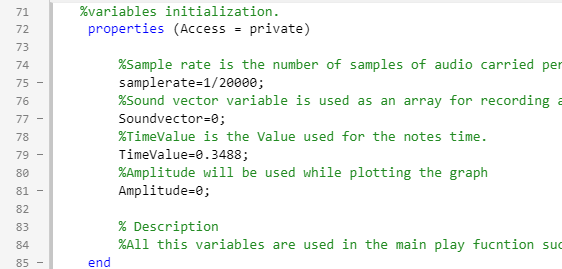


Figure 16 User defined Variable initialization

The last property which automatically update by the compiler is called as in-built variable, in (Figure 10) all the variables are user defined and all are enclosed in the Access Specifier type of private which lets the variables to be used in just the same class of the program providing a security feature.

samplerate variable is a double type basically is used for sampling the frequency which means number of samples of audio carried per second, Formula for calculating the sample rate is **(1/Δt)** where Δt is the sampling period in this program samplerate variable is initialized to 1/20000;

Soundvector is an array of integer type which is used in making an array of frequency when the recording button is selected.

TimeValue variable is used for the timesampling which is same as the frequency sample just in this case wrt to time.

Amplitude variable is used for frequency over amplitude graph for every notes selected by the user.

# 4.03 PLAYNOTE (main) method.

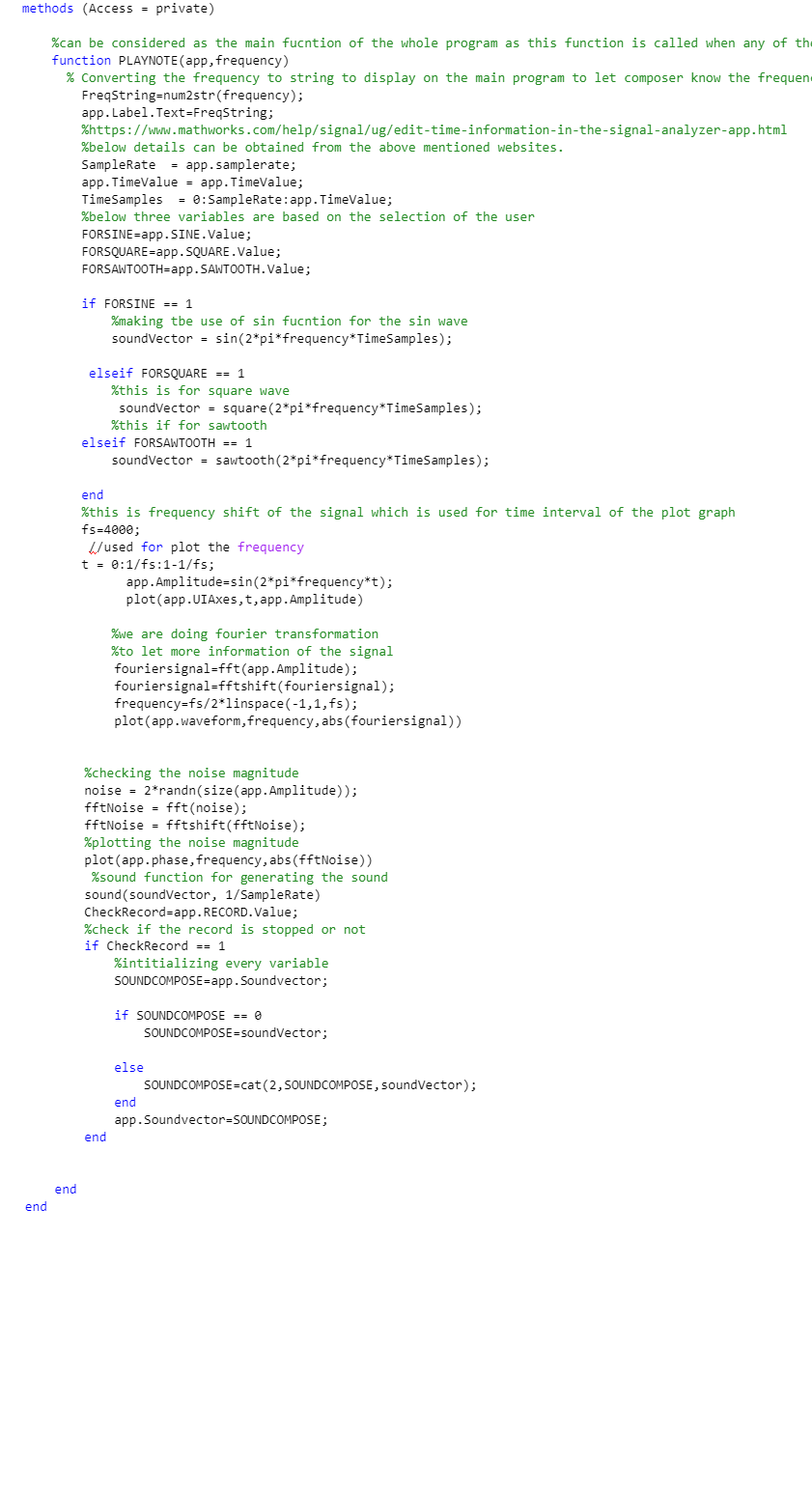


Figure 17 Play note function variable initialization

PLAYNOTE function is the function which produces sound plot, graphs and also record that’s why it can be considered the main function(method). PLAYNOTE function accepts app and frequency as the parameters where app is just like handle of GUIDE telling the program which event is occurred at the time. Next parameter is frequency which is basically the frequency passed as the arguments from all the note’s function.

To make the frequency appear on the UI num2str() been used to convert the frequency of Integer type to String type and then display on the dedicated label.

SampleRate and TimeValue variable are just used for accessing the value of main samplerate and TimeValue variable from the properties. TimeSample variable is an array variable for the time sampling where it uses SampleRate and TimeValue for sampling the time period.

FORSINE, FORSQUARE, FORSAWTOOTH are the variables used for inspecting that which type of user select’s, for eg:- if the user selected the SINE waveform FORSINE variable will receive a value of 1. Same goes for other two variable.

# 

Figure 18 PLAYNOTE method waveform function

After the program receives the input from the user for the waveform type it then uses if else conditional statement in which three different function (sin (), square (), sawtooth ()) respective of their types uses the frequency and TimeSamples variable for generating an array for the sound which is stored in soundVector variable

sin (), square (), sawtooth () are the inbuilt function in MATLAB which generates wave of their respective type. This function operates on array’s element wise basis and this function accept both real and complex inputs.

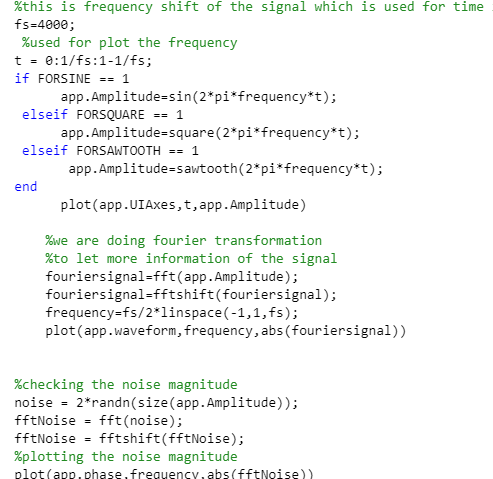


Figure 19 PLAYNOTE method plot function

There are three different types of graph used in this program 1) Amplitude vs Frequency 2) Fourier transformation 3) Noise, all types use Plot function is one of most useful inbuilt function of MATLAB used for plotting the graph of desired values.

For Amplitude vs Frequency graph Amplitude been initialized depends on which type of waveform using the same concept used earlier, and for the frequency shift fs variable is used making a default variable of 400 is used basically this variable determine the time interval in the graph and then variable t is used for making an array of time sample using the frequency value which is then used in initialization of the Amplitude variable, and then plot function is used for making the graph.

For Fourier transformation graph fft and fftshift function is used which is inbuilt functions in MATLAB for generating Fourier transform of a signal and the argument passed to this function is just the signal which required to be transformed and then later use plot function for the graph.

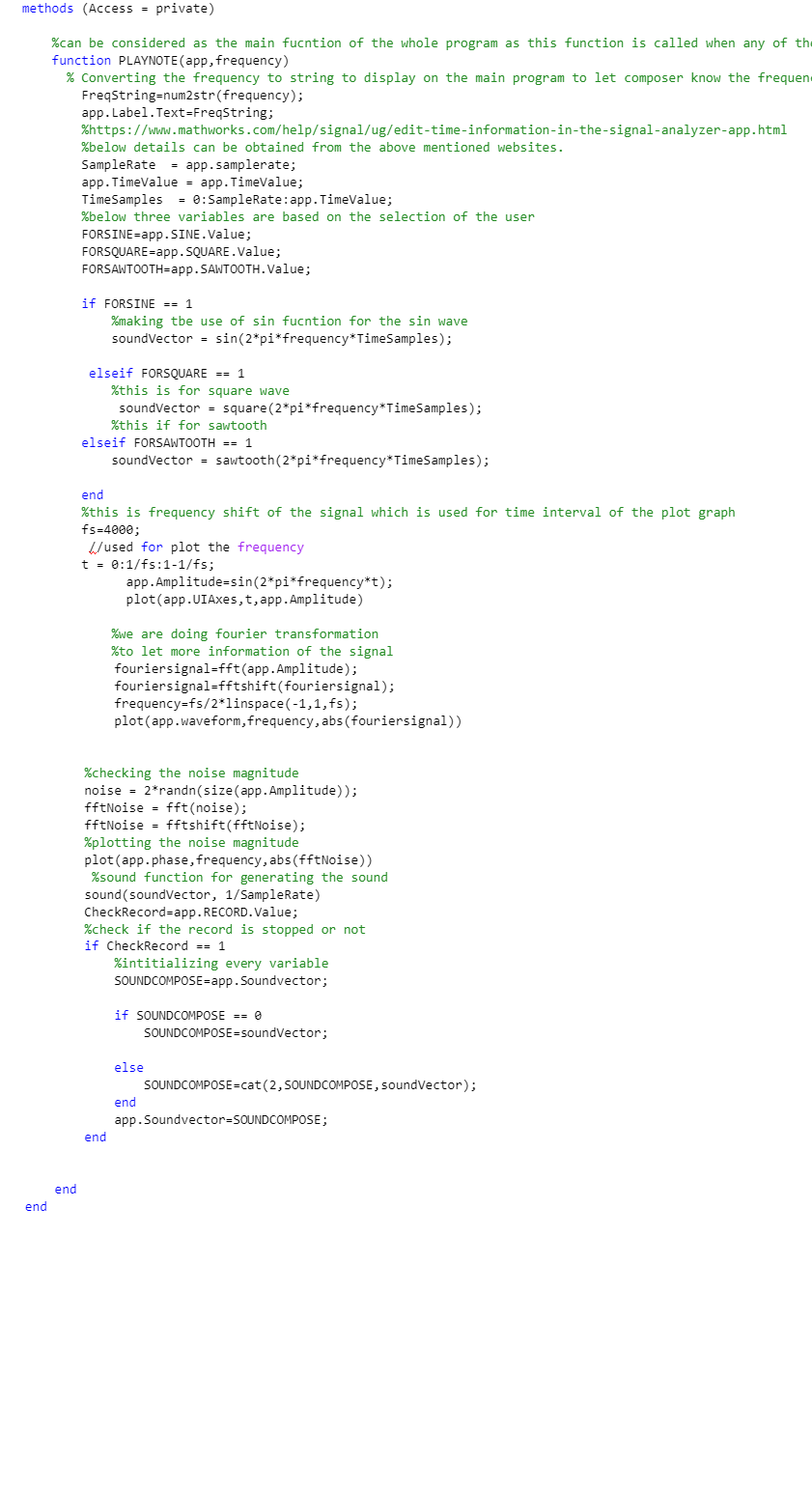


Figure 20 PLAYNOTE method record

For producing sound inbuilt function sound () is used which basically convert matrix of signal data to sound, sound() can have up to three parameters but in this program just two arguments been passed to sound function first one is the array and the other one is sample rate.

For record function program first check whether the user selects the record button using if else case then if program receives value of 1 then it initializes the SoundVector array which hold the current note selected by the user to SOUNDCOMPOSE variable and then using the cat function which is inbuilt function if MATLAB it concatenates or merges the array passing 2 as the dimension size of the array and SOUNDCOMPOSE having the past notes with SoundVector having the current note and then at the end it initialize the SOUNDCOMPOSE variable to SoundVector because in program sound() function uses SoundVector as the argument .

# 4.04 Play Reset and Notes method

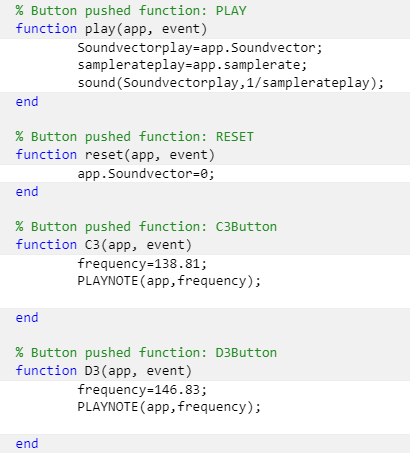


Figure 21 play reset and note methods

The play method uses it to play the current recording which is done by the user for which it just access to the SoundVector and samplerate variable then using sound method it just passes the two variables as the argument and hence producing sound of the current recording.

The reset method just initializes the SoundVector variable to 0 such that there is nothing left in the array to produce the sound.

Each note has its own frequency there are total 36 different frequency value set and all the notes function works on same principle in which it first initializes the frequency variable and then call PLAYNOTE function passing the frequency as the argument.

# 4.05 Duration of note method

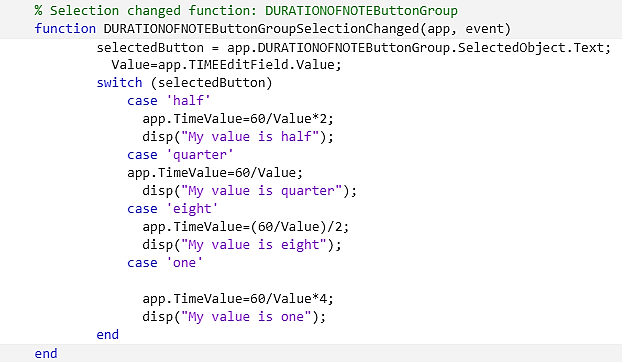


Figure 22 Duration of note method

This method initializes variable selectedButton which contains the value of the duration selected by the user from the button group, a Value variable been initialized with the Time value which user passed in the main design that denotes bpm (beats per minute) and then uses Switch case statement is also like if else is a conditional statement where it evaluates an expression and then chooses to executes one of the several cases listed.

Different cases been provided as per in the button group and every case just initialize the global variable TimeValue as per the case which is for half the 60/Value multiplies by 2 for quarter numerator remains same denominator changes and same for rest cases, and in every case disp () function is used to print out the statement provided such as to debug and trace for the flow of program.

# 4.06 SAVEAS method

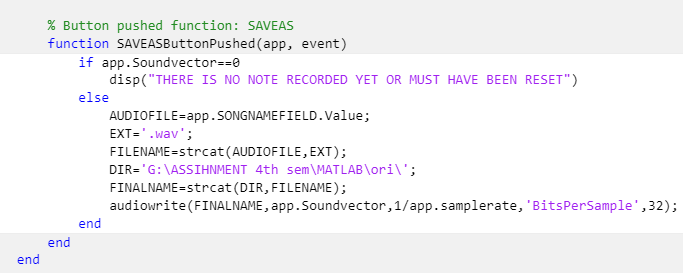


Figure 23 SAVEAS method.

SAVEAS method save the recorded file provided by the user, it first runs a if conditional statement checking the condition that whether the variable which have the recorded array variable have any value by comparing the SoundVector to 0 and then if the condition is true it display the message stating there is no note recorded or must have been reset.

If the condition is false then in else block a Variable name AUDIOFILE been initialized to get the value from the Field where user enter the filename, as the extension of the saved file is .wav another variable with .wav extension been initialized as a String. Using strcat () function which is a inbuilt function in MATLAB which merges the two different strings or concatenates them, it merges the filename variable with the extension variable. DIR variable is a string variable holding the address of the file path where the file will be saved.

audiowrite () function is an inbuilt function in MATLAB which writes a matrix of audio data with sample rate to a file given by the user.so using this function a 32-bit audio file been generated by providing the 32 as the argument to the audiowrite function.

# 5. RESULT.

Upon running the .mlapp extension file, a GUI of the main program is shown

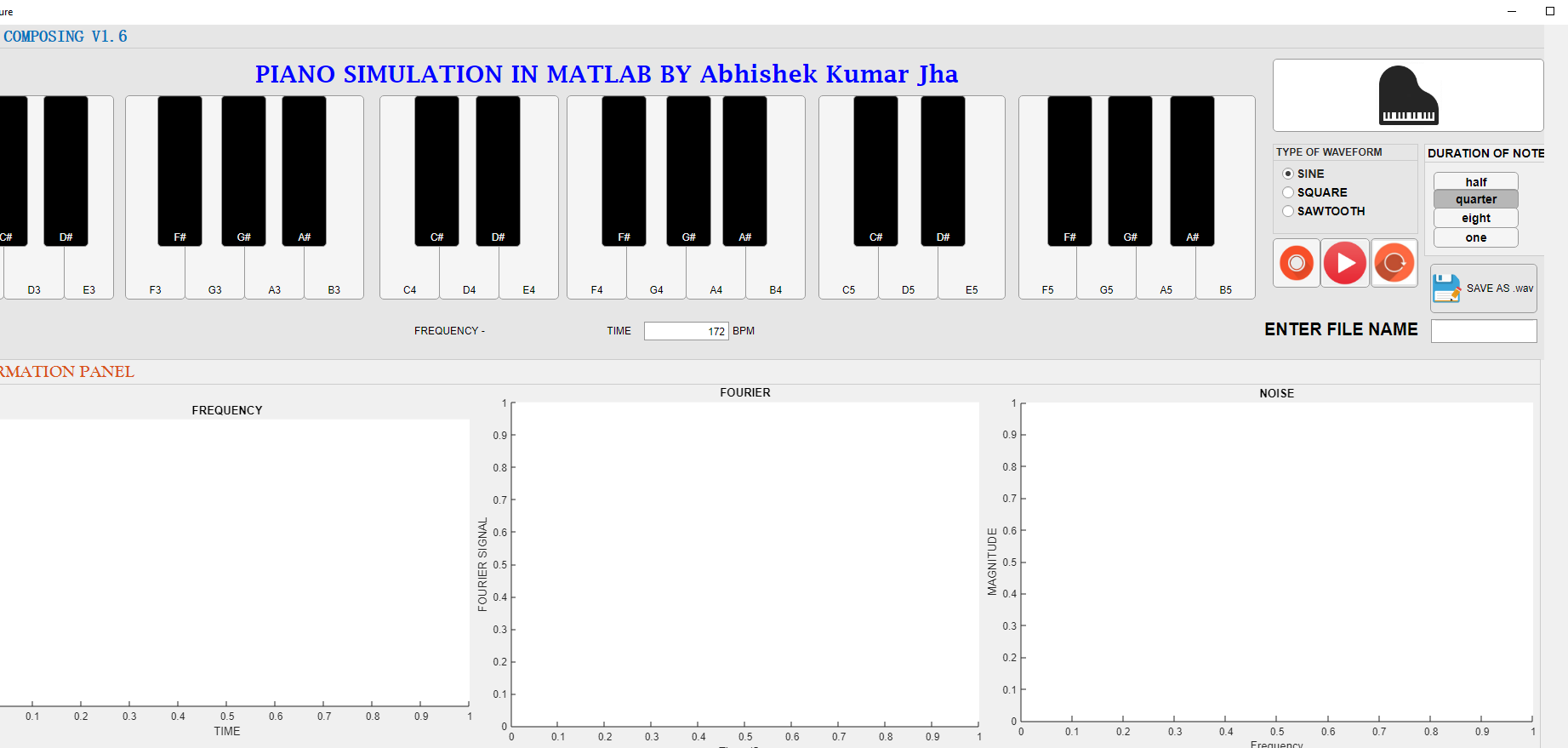


Figure 24 MAIN PANEL

After the user selects a note the graph been displayed and the sound can be heard from the speaker. This waveform is by default the sine waveform. Duration of note just affect the NOISE graph as the more sound will be produced the more noise will be generated but the Frequency and the Fourier will be same.

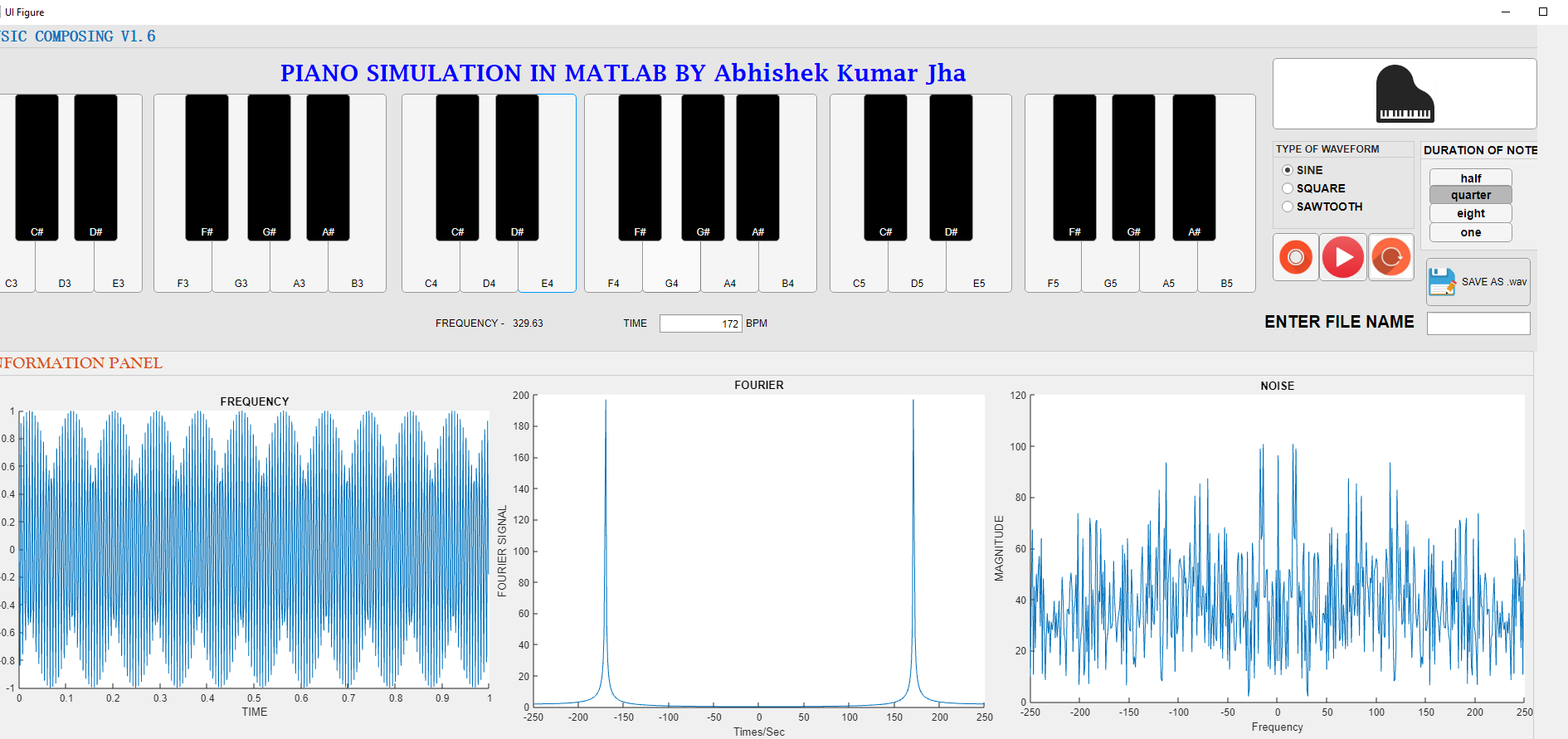


Figure 25 SINE waveform

All the three different types of graph show their respective graph when user selects different types of waveform.

# 

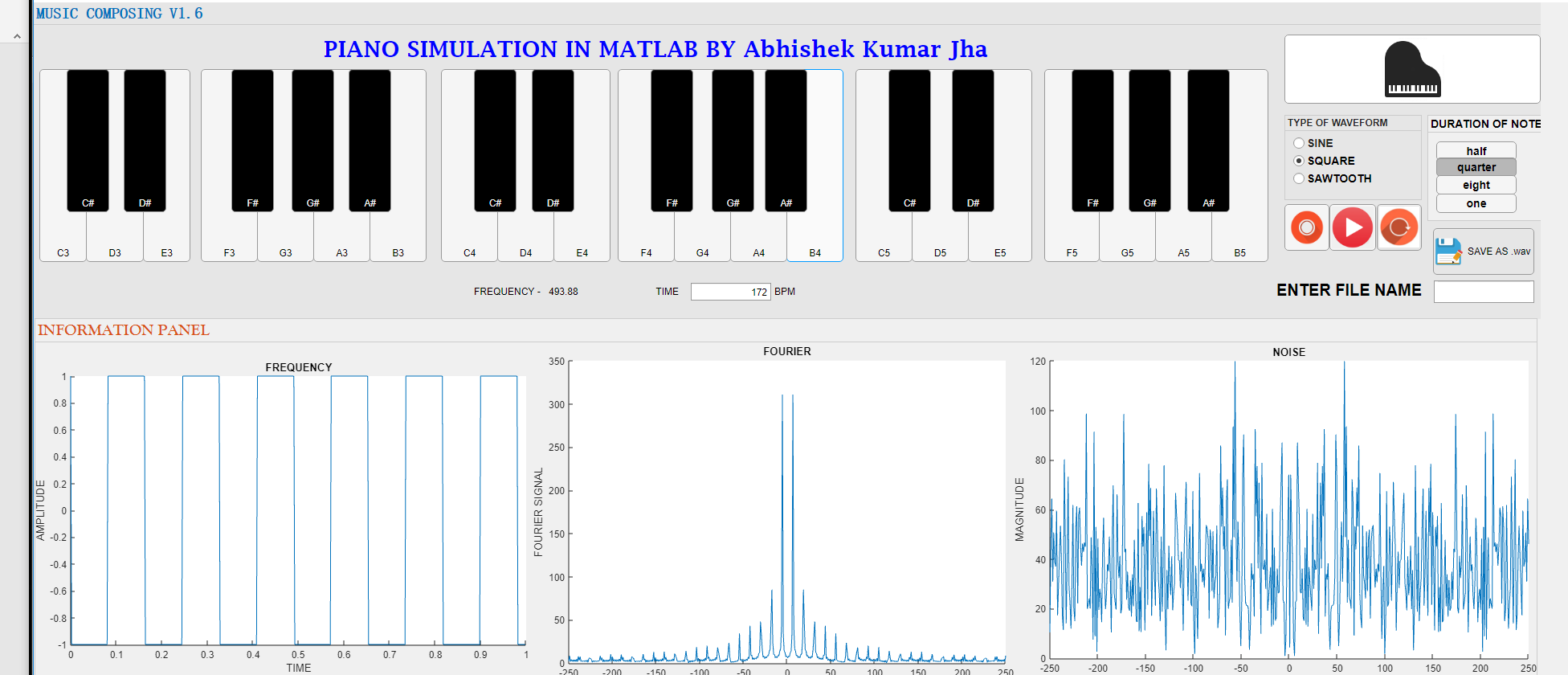
Figure 26 B4 note pressed in sine waveform

Figure 27 B4 note of square waveform

# 

Figure 28 B4 note of sawtooth waveform

(Figure 29) Shows the input of the filename for saving the audio file and (Figure 30) shows the output of the recorded file

# 

Figure 29 Filename input

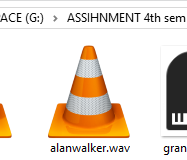


Figure 30 Output record

# 7. DISCUSSION

The most important finding while researching for this program has been App designer there are many similarities and differences between the GUIDE and App designer basically in terms of similarities the way of approach towards a task is similar but App designer on the other hand is more efficient and faster in terms of handling the code. App designer codes are also better in terms for handing the components and at the same time making it more secure and simple.

For making the design of the GUI the idea has been taken from other available music composing apps to make it more user friendly and realistic. So, for that the help library of MATLAB turns out to be one of the most significant and powerful tool of the MATLAB. Use of different GUI components of Matlab came out to be fruitful in many scenarios for the use of different functions or methods.

The function record came out to be just a bit challenging because of the use of the function in its callback which later overcome by applying the logic in the play or main function rather than calling it in its callback.

Some of the inbuilt function of the MATLAB shows the versatility and the importance of the MATLAB in engineering environment like Plot audiowrite and sound. Plot is one of the best features of MATLAB which has been used for three different implementations for graph in the program. Exporting the file to an audio format at first sounds so tough task to do which with the help of the function audiowrite came out to be just a single line process.

Frequency set for every note can be considered as the most time consuming task as to set the frequency to reflect the corresponding sound is not easily achievable thing but after many trials and playing with the value of both sample variable and frequency variable the sound produced has been acceptable when compared with original sound of the note, Initializing the sound function for every frequency also very lengthy and long process which was later updated and overcome with by applying a separate function.

Though the designed GUI has many important features which makes it a preferable music composing application but still there are some further improvements which can be applied in the future to make it more User friendly and features packed.

# 7. CONCLUSION

The task allocated for this assignment has been successfully achieved, whereby a Music composing app that let user to compose the music as per required and also graph the different factors of the generated sound also facilitating the user to record their music to listen and to save the file to audio format have been developed by App Designer in Matlab and the program has been executed with no errors. However, multiple errors were experienced during coding and were solved through extensive research and consultations. One of the most important features of the MATLAB interface was the “help”. It has been very helpful and rigorous and a lot have been learnt from it. Because MATLAB is an interpreted language, it can be slow, and poor programming practices can make it unacceptably slow especially when it comes to executing blocks of code with loops which are not terminated correctly. A GUI application design should begin with a set of specifications and parameters that serve as baseline requirements that the program must accomplish to meet the needs and demands of the user. Some of the advantages learnt about the graphical user interface include being easy to use, easy accessibility, increased productivity and being attractive to non-technical users since it enables users to accomplish work on a computer using images instead of text commands.

# REFRENCES

Mathworks.com. (2018). *MATLABDocumentation*. [online] Available at: https://www.mathworks.com/help/matlab/index.html [Accessed 12 May 2018].

Fon.hum.uva.nl. (2018). *Get sampling period*. [online] Available at: http://www.fon.hum.uva.nl/praat/manual/Get\_sampling\_period.html [Accessed 12 May 2018].

Ian Cavers (2008). *Numerical computing with Matlab*. 3rd ed. New jersey: McGraw-Hill Education. P23-24.

ELSOC Wiki. (2018). *Music in Matlab*. [online] Available at: http://elsoc.wikia.com/wiki/Music\_in\_Matlab [Accessed 12 May 2018].

Higham, D. and Higham, N. (2005). *MATLAB guide*. Philadelphia: Society for Industrial and Applied Mathematics.

Abebooks.com. (2018). *9780495244493 - Matlab Programming for Engineers by Stephen J Chapman - AbeBooks*. [online] Available at: https://www.abebooks.com/book-search/isbn/9780495244493/ [Accessed 12 May 2018].

# APPENDIX

classdef PAINOesa < matlab.apps.AppBase

% Properties that correspond to app components

properties (Access = public)

UIFigure matlab.ui.Figure

MUSICCOMPOSINGV16Panel matlab.ui.container.Panel

F5Button matlab.ui.control.Button

G5Button matlab.ui.control.Button

A5Button matlab.ui.control.Button

B5Button matlab.ui.control.Button

Asharp5 matlab.ui.control.Button

TYPEOFWAVEFORMButtonGroup matlab.ui.container.ButtonGroup

SINE matlab.ui.control.RadioButton

SQUARE matlab.ui.control.RadioButton

SAWTOOTH matlab.ui.control.RadioButton

PIANOSIMULATIONINMATLABBYAbhishekKumarJhaLabel matlab.ui.control.Label

PLAY matlab.ui.control.Button

RESET matlab.ui.control.Button

HELP matlab.ui.control.Button

DURATIONOFNOTEButtonGroup matlab.ui.container.ButtonGroup

half matlab.ui.control.ToggleButton

quarter matlab.ui.control.ToggleButton

eight matlab.ui.control.ToggleButton

one matlab.ui.control.ToggleButton

SAVEAS matlab.ui.control.Button

ENTERFILENAMEEditFieldLabel matlab.ui.control.Label

SONGNAMEFIELD matlab.ui.control.EditField

E5Button matlab.ui.control.Button

D5Button matlab.ui.control.Button

C5Button matlab.ui.control.Button

B4Button matlab.ui.control.Button

A4Button matlab.ui.control.Button

G4Button matlab.ui.control.Button

F4Button matlab.ui.control.Button

E4Button matlab.ui.control.Button

D4Button matlab.ui.control.Button

C4Button matlab.ui.control.Button

B3Button matlab.ui.control.Button

A3Button matlab.ui.control.Button

G3Button matlab.ui.control.Button

F3Button matlab.ui.control.Button

E3Button matlab.ui.control.Button

D3Button matlab.ui.control.Button

C3Button matlab.ui.control.Button

Gsharp5 matlab.ui.control.Button

Fsharp5 matlab.ui.control.Button

Dsharp5 matlab.ui.control.Button

Csharp5 matlab.ui.control.Button

ASharp4 matlab.ui.control.Button

Gsharp4 matlab.ui.control.Button

Fsharp4 matlab.ui.control.Button

Dsharp4 matlab.ui.control.Button

Csharp4 matlab.ui.control.Button

Asharp3 matlab.ui.control.Button

Gsharp3 matlab.ui.control.Button

Fsharp3 matlab.ui.control.Button

Dsharp3 matlab.ui.control.Button

Csharp3 matlab.ui.control.Button

RECORD matlab.ui.control.StateButton

Label matlab.ui.control.Label

FREQUENCYLabel matlab.ui.control.Label

TIMEEditFieldLabel matlab.ui.control.Label

TIMEEditField matlab.ui.control.NumericEditField

BPMLabel matlab.ui.control.Label

INFORMATIONPANELPanel matlab.ui.container.Panel

phase matlab.ui.control.UIAxes

waveform matlab.ui.control.UIAxes

UIAxes matlab.ui.control.UIAxes

end

%variables initialization.

properties (Access = private)

%Sample rate is the number of samples of audio carried per second.

samplerate=1/20000;

%Sound vector variable is used as an array for recording and inserting the frequency one by one as per pressed

Soundvector=0;

%TimeValue is the Value used for the notes time.

TimeValue=0.3488;

%Amplitude will be used while plotting the graph

Amplitude=0;

% Description

%All this variables are used in the main play fucntion such as to make it easier for other function to access this variable if requied

end

methods (Access = private)

%can be considered as the main fucntion of the whole program as this function is called when any of the note is selected

function PLAYNOTE(app,frequency)

% Converting the frequency to string to display on the main program to let composer know the frequency of the note

FreqString=num2str(frequency);

app.Label.Text=FreqString;

%https://www.mathworks.com/help/signal/ug/edit-time-information-in-the-signal-analyzer-app.html

%below details can be obtained from the above mentioned websites.

SampleRate = app.samplerate;

app.TimeValue = app.TimeValue;

TimeSamples = 0:SampleRate:app.TimeValue;

%below three variables are based on the selection of the user

FORSINE=app.SINE.Value;

FORSQUARE=app.SQUARE.Value;

FORSAWTOOTH=app.SAWTOOTH.Value;

if FORSINE == 1

%making tbe use of sin fucntion for the sin wave

soundVector = sin(2\*pi\*frequency\*TimeSamples);

elseif FORSQUARE == 1

%this is for square wave

soundVector = square(2\*pi\*frequency\*TimeSamples);

%this if for sawtooth

elseif FORSAWTOOTH == 1

soundVector = sawtooth(2\*pi\*frequency\*TimeSamples);

end

%this is frequency shift of the signal which is used for time interval of the plot graph

fs=500;

%used for plot the frequency

t = 0:1/fs:1-1/fs;

if FORSINE == 1

app.Amplitude=sin(2\*pi\*frequency\*t);

elseif FORSQUARE == 1

app.Amplitude=square(2\*pi\*frequency\*t);

elseif FORSAWTOOTH == 1

app.Amplitude=sawtooth(2\*pi\*frequency\*t);

end

plot(app.UIAxes,t,app.Amplitude)

%we are doing fourier transformation

%to let more information of the signal

fouriersignal=fft(app.Amplitude);

fouriersignal=fftshift(fouriersignal);

frequency=fs/2\*linspace(-1,1,fs);

plot(app.waveform,frequency,abs(fouriersignal))

%checking the noise magnitude

noise = 2\*randn(size(app.Amplitude));

fftNoise = fft(noise);

fftNoise = fftshift(fftNoise);

%plotting the noise magnitude

plot(app.phase,frequency,abs(fftNoise))

%sound function for generating the sound

sound(soundVector, 1/SampleRate)

CheckRecord=app.RECORD.Value;

%check if the record is stopped or not

if CheckRecord == 1

%intitializing every variable

SOUNDCOMPOSE=app.Soundvector;

if SOUNDCOMPOSE == 0

SOUNDCOMPOSE=soundVector;

else

SOUNDCOMPOSE=cat(2,SOUNDCOMPOSE,soundVector);

end

app.Soundvector=SOUNDCOMPOSE;

end

end

end

methods (Access = private)

% Size changed function: MUSICCOMPOSINGV16Panel

function MUSICCOMPOSINGV16PanelSizeChanged(app, event)

position = app.MUSICCOMPOSINGV16Panel.Position

end

% Button pushed function: PLAY

function play(app, event)

Soundvectorplay=app.Soundvector;

samplerateplay=app.samplerate;

sound(Soundvectorplay,1/samplerateplay);

end

% Button pushed function: RESET

function reset(app, event)

app.Soundvector=0;

end

% Button pushed function: C3Button

function C3(app, event)

frequency=138.81;

PLAYNOTE(app,frequency);

end

% Button pushed function: D3Button

function D3(app, event)

frequency=146.83;

PLAYNOTE(app,frequency);

end

% Button pushed function: E3Button

function E3(app, event)

frequency=164.81;

PLAYNOTE(app,frequency);

end

% Callback function

function size\_change(app, event)

position = app.UIFigure.Position;

end

% Button pushed function: HELP

function help\_pushed(app, event)

message = sprintf('Matlab paino version V1.6\nFor any enquiry Contact-\nEmail-abhi83166@gmail.com\nWhatsapp- +601128544017');

uiwait(msgbox(message));

end

% Button pushed function: Csharp3

function Csharp3ButtonPushed(app, event)

frequency=138.59;

PLAYNOTE(app,frequency);

end

% Button pushed function: F5Button

function F5ButtonPushed(app, event)

frequency=698.46;

PLAYNOTE(app,frequency);

end

% Button pushed function: G5Button

function G5ButtonPushed(app, event)

frequency=783.99;

PLAYNOTE(app,frequency);

end

% Button pushed function: A5Button

function A5ButtonPushed(app, event)

frequency=880;

PLAYNOTE(app,frequency);

end

% Button pushed function: B5Button

function B5ButtonPushed(app, event)

frequency=987.77;

PLAYNOTE(app,frequency);

end

% Button pushed function: Asharp5

function Asharp5ButtonPushed(app, event)

frequency=932.33;

PLAYNOTE(app,frequency);

end

% Button pushed function: E5Button

function E5ButtonPushed(app, event)

frequency=659.26;

PLAYNOTE(app,frequency);

end

% Button pushed function: D5Button

function D5ButtonPushed(app, event)

frequency=587.33;

PLAYNOTE(app,frequency);

end

% Button pushed function: C5Button

function C5ButtonPushed(app, event)

frequency=523.25;

PLAYNOTE(app,frequency);

end

% Button pushed function: B4Button

function B4ButtonPushed(app, event)

frequency=493.88;

PLAYNOTE(app,frequency);

end

% Button pushed function: A4Button

function A4ButtonPushed(app, event)

frequency=440;

PLAYNOTE(app,frequency);

end

% Button pushed function: G4Button

function G4ButtonPushed(app, event)

frequency=392;

PLAYNOTE(app,frequency);

end

% Button pushed function: F4Button

function F4ButtonPushed(app, event)

frequency=349.23;

PLAYNOTE(app,frequency);

end

% Button pushed function: E4Button

function E4ButtonPushed(app, event)

frequency=329.63;

PLAYNOTE(app,frequency);

end

% Button pushed function: D4Button

function D4ButtonPushed(app, event)

frequency=293.66;

PLAYNOTE(app,frequency);

end

% Button pushed function: C4Button

function C4ButtonPushed(app, event)

frequency=261.625;

PLAYNOTE(app,frequency);

end

% Button pushed function: B3Button

function B3ButtonPushed(app, event)

frequency=246.94;

PLAYNOTE(app,frequency);

end

% Button pushed function: A3Button

function A3ButtonPushed(app, event)

frequency=220;

PLAYNOTE(app,frequency);

end

% Button pushed function: G3Button

function G3ButtonPushed(app, event)

frequency=196;

PLAYNOTE(app,frequency);

end

% Button pushed function: F3Button

function F3ButtonPushed(app, event)

frequency=174.61;

PLAYNOTE(app,frequency);

end

% Button pushed function: Gsharp5

function Gsharp5ButtonPushed(app, event)

frequency=830.61;

PLAYNOTE(app,frequency);

end

% Button pushed function: Fsharp5

function Fsharp5ButtonPushed(app, event)

frequency=739.99;

PLAYNOTE(app,frequency);

end

% Button pushed function: Dsharp5

function Dsharp5ButtonPushed(app, event)

frequency=622;

PLAYNOTE(app,frequency);

end

% Button pushed function: Csharp5

function Csharp5ButtonPushed(app, event)

frequency=554.37;

PLAYNOTE(app,frequency);

end

% Button pushed function: ASharp4

function ASharp4ButtonPushed(app, event)

frequency=466.16;

PLAYNOTE(app,frequency);

end

% Button pushed function: Gsharp4

function Gsharp4ButtonPushed(app, event)

frequency=415.3;

PLAYNOTE(app,frequency);

end

% Button pushed function: Fsharp4

function Fsharp4ButtonPushed(app, event)

frequency=369.99;

PLAYNOTE(app,frequency);

end

% Button pushed function: Dsharp4

function Dsharp4ButtonPushed(app, event)

frequency=311.13;

PLAYNOTE(app,frequency);

end

% Button pushed function: Csharp4

function Csharp4ButtonPushed(app, event)

frequency=277.18;

PLAYNOTE(app,frequency);

end

% Button pushed function: Asharp3

function Asharp3ButtonPushed(app, event)

frequency=233.08;

PLAYNOTE(app,frequency);

end

% Button pushed function: Gsharp3

function Gsharp3ButtonPushed(app, event)

frequency=207.65;

PLAYNOTE(app,frequency);

end

% Button pushed function: Fsharp3

function Fsharp3ButtonPushed(app, event)

frequency=185.00;

PLAYNOTE(app,frequency);

end

% Button pushed function: Dsharp3

function Dsharp3ButtonPushed(app, event)

frequency=155.56;

PLAYNOTE(app,frequency);

end

% Selection changed function: DURATIONOFNOTEButtonGroup

function DURATIONOFNOTEButtonGroupSelectionChanged(app, event)

selectedButton = app.DURATIONOFNOTEButtonGroup.SelectedObject.Text;

Value=app.TIMEEditField.Value;

switch (selectedButton)

case 'half'

app.TimeValue=60/Value\*2;

disp("My value is half");

case 'quarter'

app.TimeValue=60/Value;

disp("My value is quarter");

case 'eight'

app.TimeValue=(60/Value)/2;

disp("My value is eight");

case 'one'

app.TimeValue=60/Value\*4;

disp("My value is one");

end

end

% Selection changed function: TYPEOFWAVEFORMButtonGroup

function TYPEOFWAVEFORMButtonGroupSelectionChanged(app, event)

selectedButton = app.TYPEOFWAVEFORMButtonGroup.SelectedObject;

end

% Value changed function: RECORD

function RECORDValueChanged(app, event)

value = app.RECORD.Value;

end

% Value changed function: TIMEEditField

function TIMEEditFieldValueChanged(app, event)

value = app.TIMEEditField.Value;

app.TimeValue=60/value;

selectedButton = app.DURATIONOFNOTEButtonGroup.SelectedObject.Text;

switch(selectedButton)

case 'half'

app.TimeValue=app.TimeValue\*2;

case 'one'

app.TimeValue=app.TimeValue\*4;

case 'eight'

app.TimeValue=app.TimeValue/2;

end

end

% Button pushed function: SAVEAS

function SAVEASButtonPushed(app, event)

if app.Soundvector==0

disp("THERE IS NO NOTE RECORDED YET OR MUST HAVE BEEN RESET")

else

AUDIOFILE=app.SONGNAMEFIELD.Value;

EXT='.wav';

FILENAME=strcat(AUDIOFILE,EXT);

DIR='G:\ASSIHNMENT 4th sem\MATLAB\ori\';

FINALNAME=strcat(DIR,FILENAME);

audiowrite(FINALNAME,app.Soundvector,1/app.samplerate,'BitsPerSample',32);

end

end

end

% App initialization and construction

methods (Access = private)

% Create UIFigure and components

function createComponents(app)

% Create UIFigure

app.UIFigure = uifigure;

app.UIFigure.Position = [100 100 1952 862];

app.UIFigure.Name = 'UI Figure';

% Create MUSICCOMPOSINGV16Panel

app.MUSICCOMPOSINGV16Panel = uipanel(app.UIFigure);

app.MUSICCOMPOSINGV16Panel.ForegroundColor = [0 0.451 0.7412];

app.MUSICCOMPOSINGV16Panel.BorderType = 'none';

app.MUSICCOMPOSINGV16Panel.Title = 'MUSIC COMPOSING V1.6';

app.MUSICCOMPOSINGV16Panel.BackgroundColor = [0.902 0.902 0.902];

app.MUSICCOMPOSINGV16Panel.SizeChangedFcn = createCallbackFcn(app, @MUSICCOMPOSINGV16PanelSizeChanged, true);

app.MUSICCOMPOSINGV16Panel.FontName = 'SimSun';

app.MUSICCOMPOSINGV16Panel.FontWeight = 'bold';

app.MUSICCOMPOSINGV16Panel.FontSize = 20;

app.MUSICCOMPOSINGV16Panel.Position = [1 469 1876 394];

% Create F5Button

app.F5Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.F5Button.ButtonPushedFcn = createCallbackFcn(app, @F5ButtonPushed, true);

app.F5Button.VerticalAlignment = 'bottom';

app.F5Button.Position = [1259 72 69 240];

app.F5Button.Text = 'F5';

% Create G5Button

app.G5Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.G5Button.ButtonPushedFcn = createCallbackFcn(app, @G5ButtonPushed, true);

app.G5Button.VerticalAlignment = 'bottom';

app.G5Button.Position = [1327 72 71 240];

app.G5Button.Text = 'G5';

% Create A5Button

app.A5Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.A5Button.ButtonPushedFcn = createCallbackFcn(app, @A5ButtonPushed, true);

app.A5Button.VerticalAlignment = 'bottom';

app.A5Button.Position = [1397 72 71 240];

app.A5Button.Text = 'A5';

% Create B5Button

app.B5Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.B5Button.ButtonPushedFcn = createCallbackFcn(app, @B5ButtonPushed, true);

app.B5Button.VerticalAlignment = 'bottom';

app.B5Button.Position = [1467 72 71 240];

app.B5Button.Text = 'B5';

% Create Asharp5

app.Asharp5 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Asharp5.ButtonPushedFcn = createCallbackFcn(app, @Asharp5ButtonPushed, true);

app.Asharp5.VerticalAlignment = 'bottom';

app.Asharp5.BackgroundColor = [0 0 0];

app.Asharp5.FontColor = [0.9412 0.9412 0.9412];

app.Asharp5.Position = [1437 134 53 178];

app.Asharp5.Text = 'A#';

% Create TYPEOFWAVEFORMButtonGroup

app.TYPEOFWAVEFORMButtonGroup = uibuttongroup(app.MUSICCOMPOSINGV16Panel);

app.TYPEOFWAVEFORMButtonGroup.SelectionChangedFcn = createCallbackFcn(app, @TYPEOFWAVEFORMButtonGroupSelectionChanged, true);

app.TYPEOFWAVEFORMButtonGroup.ForegroundColor = [0.149 0.149 0.149];

app.TYPEOFWAVEFORMButtonGroup.Title = 'TYPE OF WAVEFORM';

app.TYPEOFWAVEFORMButtonGroup.BackgroundColor = [0.902 0.902 0.902];

app.TYPEOFWAVEFORMButtonGroup.FontName = 'Arial';

app.TYPEOFWAVEFORMButtonGroup.FontWeight = 'bold';

app.TYPEOFWAVEFORMButtonGroup.Position = [1558 149 171 106];

% Create SINE

app.SINE = uiradiobutton(app.TYPEOFWAVEFORMButtonGroup);

app.SINE.Text = 'SINE';

app.SINE.FontSize = 14;

app.SINE.FontWeight = 'bold';

app.SINE.FontColor = [0.149 0.149 0.149];

app.SINE.Position = [11 60 58 22];

app.SINE.Value = true;

% Create SQUARE

app.SQUARE = uiradiobutton(app.TYPEOFWAVEFORMButtonGroup);

app.SQUARE.Text = 'SQUARE';

app.SQUARE.FontSize = 14;

app.SQUARE.FontWeight = 'bold';

app.SQUARE.Position = [11 38 82 22];

% Create SAWTOOTH

app.SAWTOOTH = uiradiobutton(app.TYPEOFWAVEFORMButtonGroup);

app.SAWTOOTH.Text = 'SAWTOOTH';

app.SAWTOOTH.FontSize = 14;

app.SAWTOOTH.FontWeight = 'bold';

app.SAWTOOTH.Position = [11 16 103 22];

% Create PIANOSIMULATIONINMATLABBYAbhishekKumarJhaLabel

app.PIANOSIMULATIONINMATLABBYAbhishekKumarJhaLabel = uilabel(app.MUSICCOMPOSINGV16Panel);

app.PIANOSIMULATIONINMATLABBYAbhishekKumarJhaLabel.BackgroundColor = [0.902 0.902 0.902];

app.PIANOSIMULATIONINMATLABBYAbhishekKumarJhaLabel.HorizontalAlignment = 'center';

app.PIANOSIMULATIONINMATLABBYAbhishekKumarJhaLabel.FontName = 'Lucida Fax';

app.PIANOSIMULATIONINMATLABBYAbhishekKumarJhaLabel.FontSize = 28;

app.PIANOSIMULATIONINMATLABBYAbhishekKumarJhaLabel.FontWeight = 'bold';

app.PIANOSIMULATIONINMATLABBYAbhishekKumarJhaLabel.FontColor = [0 0 1];

app.PIANOSIMULATIONINMATLABBYAbhishekKumarJhaLabel.Position = [17 320 1526 35];

app.PIANOSIMULATIONINMATLABBYAbhishekKumarJhaLabel.Text = 'PIANO SIMULATION IN MATLAB BY Abhishek Kumar Jha ';

% Create PLAY

app.PLAY = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.PLAY.ButtonPushedFcn = createCallbackFcn(app, @play, true);

app.PLAY.Icon = 'Play.png';

app.PLAY.Position = [1614 86 58 58];

app.PLAY.Text = '';

% Create RESET

app.RESET = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.RESET.ButtonPushedFcn = createCallbackFcn(app, @reset, true);

app.RESET.Icon = 'reset.jpg';

app.RESET.BackgroundColor = [0.902 0.902 0.902];

app.RESET.Position = [1673 86 56 58];

app.RESET.Text = '';

% Create HELP

app.HELP = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.HELP.ButtonPushedFcn = createCallbackFcn(app, @help\_pushed, true);

app.HELP.Icon = 'grand\_piano.png';

app.HELP.BackgroundColor = [1 1 1];

app.HELP.Position = [1558 269 319 86];

app.HELP.Text = '';

% Create DURATIONOFNOTEButtonGroup

app.DURATIONOFNOTEButtonGroup = uibuttongroup(app.MUSICCOMPOSINGV16Panel);

app.DURATIONOFNOTEButtonGroup.SelectionChangedFcn = createCallbackFcn(app, @DURATIONOFNOTEButtonGroupSelectionChanged, true);

app.DURATIONOFNOTEButtonGroup.Title = 'DURATION OF NOTE';

app.DURATIONOFNOTEButtonGroup.FontName = 'Yu Gothic UI Semilight';

app.DURATIONOFNOTEButtonGroup.FontWeight = 'bold';

app.DURATIONOFNOTEButtonGroup.FontSize = 14;

app.DURATIONOFNOTEButtonGroup.Position = [1736 123 157 132];

% Create half

app.half = uitogglebutton(app.DURATIONOFNOTEButtonGroup);

app.half.Text = 'half';

app.half.FontSize = 14;

app.half.FontWeight = 'bold';

app.half.Position = [11 75 100 24];

% Create quarter

app.quarter = uitogglebutton(app.DURATIONOFNOTEButtonGroup);

app.quarter.Text = 'quarter';

app.quarter.FontSize = 14;

app.quarter.FontWeight = 'bold';

app.quarter.Position = [11 55 100 24];

app.quarter.Value = true;

% Create eight

app.eight = uitogglebutton(app.DURATIONOFNOTEButtonGroup);

app.eight.Text = 'eight';

app.eight.FontSize = 14;

app.eight.FontWeight = 'bold';

app.eight.Position = [11 33 100 24];

% Create one

app.one = uitogglebutton(app.DURATIONOFNOTEButtonGroup);

app.one.Text = 'one';

app.one.FontSize = 14;

app.one.FontWeight = 'bold';

app.one.Position = [11 10 100 24];

% Create SAVEAS

app.SAVEAS = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.SAVEAS.ButtonPushedFcn = createCallbackFcn(app, @SAVEASButtonPushed, true);

app.SAVEAS.Icon = 'Save-as-icon.png';

app.SAVEAS.HorizontalAlignment = 'right';

app.SAVEAS.BackgroundColor = [0.902 0.902 0.902];

app.SAVEAS.Position = [1743 56 126 58];

app.SAVEAS.Text = 'SAVE AS .wav';

% Create ENTERFILENAMEEditFieldLabel

app.ENTERFILENAMEEditFieldLabel = uilabel(app.MUSICCOMPOSINGV16Panel);

app.ENTERFILENAMEEditFieldLabel.BackgroundColor = [0.902 0.902 0.902];

app.ENTERFILENAMEEditFieldLabel.HorizontalAlignment = 'right';

app.ENTERFILENAMEEditFieldLabel.FontSize = 20;

app.ENTERFILENAMEEditFieldLabel.FontWeight = 'bold';

app.ENTERFILENAMEEditFieldLabel.Position = [1542 25 187 24];

app.ENTERFILENAMEEditFieldLabel.Text = 'ENTER FILE NAME';

% Create SONGNAMEFIELD

app.SONGNAMEFIELD = uieditfield(app.MUSICCOMPOSINGV16Panel, 'text');

app.SONGNAMEFIELD.FontSize = 20;

app.SONGNAMEFIELD.Position = [1744 21 125 28];

% Create E5Button

app.E5Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.E5Button.ButtonPushedFcn = createCallbackFcn(app, @E5ButtonPushed, true);

app.E5Button.VerticalAlignment = 'bottom';

app.E5Button.Position = [1164 72 80 240];

app.E5Button.Text = 'E5';

% Create D5Button

app.D5Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.D5Button.ButtonPushedFcn = createCallbackFcn(app, @D5ButtonPushed, true);

app.D5Button.VerticalAlignment = 'bottom';

app.D5Button.Position = [1094 72 71 240];

app.D5Button.Text = 'D5';

% Create C5Button

app.C5Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.C5Button.ButtonPushedFcn = createCallbackFcn(app, @C5ButtonPushed, true);

app.C5Button.VerticalAlignment = 'bottom';

app.C5Button.Position = [1024 72 71 240];

app.C5Button.Text = 'C5';

% Create B4Button

app.B4Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.B4Button.ButtonPushedFcn = createCallbackFcn(app, @B4ButtonPushed, true);

app.B4Button.VerticalAlignment = 'bottom';

app.B4Button.Position = [938 72 71 240];

app.B4Button.Text = 'B4';

% Create A4Button

app.A4Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.A4Button.ButtonPushedFcn = createCallbackFcn(app, @A4ButtonPushed, true);

app.A4Button.VerticalAlignment = 'bottom';

app.A4Button.Position = [868 72 71 240];

app.A4Button.Text = 'A4';

% Create G4Button

app.G4Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.G4Button.ButtonPushedFcn = createCallbackFcn(app, @G4ButtonPushed, true);

app.G4Button.VerticalAlignment = 'bottom';

app.G4Button.Position = [798 72 71 240];

app.G4Button.Text = 'G4';

% Create F4Button

app.F4Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.F4Button.ButtonPushedFcn = createCallbackFcn(app, @F4ButtonPushed, true);

app.F4Button.VerticalAlignment = 'bottom';

app.F4Button.Position = [728 72 71 240];

app.F4Button.Text = 'F4';

% Create E4Button

app.E4Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.E4Button.ButtonPushedFcn = createCallbackFcn(app, @E4ButtonPushed, true);

app.E4Button.VerticalAlignment = 'bottom';

app.E4Button.Position = [648 72 71 240];

app.E4Button.Text = 'E4';

% Create D4Button

app.D4Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.D4Button.ButtonPushedFcn = createCallbackFcn(app, @D4ButtonPushed, true);

app.D4Button.VerticalAlignment = 'bottom';

app.D4Button.Position = [578 72 71 240];

app.D4Button.Text = 'D4';

% Create C4Button

app.C4Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.C4Button.ButtonPushedFcn = createCallbackFcn(app, @C4ButtonPushed, true);

app.C4Button.VerticalAlignment = 'bottom';

app.C4Button.Position = [508 72 71 240];

app.C4Button.Text = 'C4';

% Create B3Button

app.B3Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.B3Button.ButtonPushedFcn = createCallbackFcn(app, @B3ButtonPushed, true);

app.B3Button.VerticalAlignment = 'bottom';

app.B3Button.Position = [419 72 71 240];

app.B3Button.Text = 'B3';

% Create A3Button

app.A3Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.A3Button.ButtonPushedFcn = createCallbackFcn(app, @A3ButtonPushed, true);

app.A3Button.VerticalAlignment = 'bottom';

app.A3Button.Position = [349 72 71 240];

app.A3Button.Text = 'A3';

% Create G3Button

app.G3Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.G3Button.ButtonPushedFcn = createCallbackFcn(app, @G3ButtonPushed, true);

app.G3Button.VerticalAlignment = 'bottom';

app.G3Button.Position = [279 72 71 240];

app.G3Button.Text = 'G3';

% Create F3Button

app.F3Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.F3Button.ButtonPushedFcn = createCallbackFcn(app, @F3ButtonPushed, true);

app.F3Button.VerticalAlignment = 'bottom';

app.F3Button.Position = [209 72 71 240];

app.F3Button.Text = 'F3';

% Create E3Button

app.E3Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.E3Button.ButtonPushedFcn = createCallbackFcn(app, @E3, true);

app.E3Button.VerticalAlignment = 'bottom';

app.E3Button.Position = [137 72 59 240];

app.E3Button.Text = 'E3';

% Create D3Button

app.D3Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.D3Button.ButtonPushedFcn = createCallbackFcn(app, @D3, true);

app.D3Button.VerticalAlignment = 'bottom';

app.D3Button.Position = [66 72 72 240];

app.D3Button.Text = 'D3';

% Create C3Button

app.C3Button = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.C3Button.ButtonPushedFcn = createCallbackFcn(app, @C3, true);

app.C3Button.VerticalAlignment = 'bottom';

app.C3Button.Position = [8 72 59 240];

app.C3Button.Text = 'C3';

% Create Gsharp5

app.Gsharp5 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Gsharp5.ButtonPushedFcn = createCallbackFcn(app, @Gsharp5ButtonPushed, true);

app.Gsharp5.VerticalAlignment = 'bottom';

app.Gsharp5.BackgroundColor = [0 0 0];

app.Gsharp5.FontColor = [0.9412 0.9412 0.9412];

app.Gsharp5.Position = [1364 134 53 178];

app.Gsharp5.Text = 'G#';

% Create Fsharp5

app.Fsharp5 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Fsharp5.ButtonPushedFcn = createCallbackFcn(app, @Fsharp5ButtonPushed, true);

app.Fsharp5.VerticalAlignment = 'bottom';

app.Fsharp5.BackgroundColor = [0 0 0];

app.Fsharp5.FontColor = [0.9412 0.9412 0.9412];

app.Fsharp5.Position = [1293 134 53 178];

app.Fsharp5.Text = 'F#';

% Create Dsharp5

app.Dsharp5 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Dsharp5.ButtonPushedFcn = createCallbackFcn(app, @Dsharp5ButtonPushed, true);

app.Dsharp5.VerticalAlignment = 'bottom';

app.Dsharp5.BackgroundColor = [0 0 0];

app.Dsharp5.FontColor = [0.9412 0.9412 0.9412];

app.Dsharp5.Position = [1144 134 53 178];

app.Dsharp5.Text = 'D#';

% Create Csharp5

app.Csharp5 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Csharp5.ButtonPushedFcn = createCallbackFcn(app, @Csharp5ButtonPushed, true);

app.Csharp5.VerticalAlignment = 'bottom';

app.Csharp5.BackgroundColor = [0 0 0];

app.Csharp5.FontColor = [0.9412 0.9412 0.9412];

app.Csharp5.Position = [1065 134 53 178];

app.Csharp5.Text = 'C#';

% Create ASharp4

app.ASharp4 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.ASharp4.ButtonPushedFcn = createCallbackFcn(app, @ASharp4ButtonPushed, true);

app.ASharp4.VerticalAlignment = 'bottom';

app.ASharp4.BackgroundColor = [0 0 0];

app.ASharp4.FontColor = [0.9412 0.9412 0.9412];

app.ASharp4.Position = [911 134 53 178];

app.ASharp4.Text = 'A#';

% Create Gsharp4

app.Gsharp4 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Gsharp4.ButtonPushedFcn = createCallbackFcn(app, @Gsharp4ButtonPushed, true);

app.Gsharp4.VerticalAlignment = 'bottom';

app.Gsharp4.BackgroundColor = [0 0 0];

app.Gsharp4.FontColor = [0.9412 0.9412 0.9412];

app.Gsharp4.Position = [845 134 53 178];

app.Gsharp4.Text = 'G#';

% Create Fsharp4

app.Fsharp4 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Fsharp4.ButtonPushedFcn = createCallbackFcn(app, @Fsharp4ButtonPushed, true);

app.Fsharp4.VerticalAlignment = 'bottom';

app.Fsharp4.BackgroundColor = [0 0 0];

app.Fsharp4.FontColor = [0.9412 0.9412 0.9412];

app.Fsharp4.Position = [769 134 53 178];

app.Fsharp4.Text = 'F#';

% Create Dsharp4

app.Dsharp4 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Dsharp4.ButtonPushedFcn = createCallbackFcn(app, @Dsharp4ButtonPushed, true);

app.Dsharp4.VerticalAlignment = 'bottom';

app.Dsharp4.BackgroundColor = [0 0 0];

app.Dsharp4.FontColor = [0.9412 0.9412 0.9412];

app.Dsharp4.Position = [621 134 53 178];

app.Dsharp4.Text = 'D#';

% Create Csharp4

app.Csharp4 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Csharp4.ButtonPushedFcn = createCallbackFcn(app, @Csharp4ButtonPushed, true);

app.Csharp4.VerticalAlignment = 'bottom';

app.Csharp4.BackgroundColor = [0 0 0];

app.Csharp4.FontColor = [0.9412 0.9412 0.9412];

app.Csharp4.Position = [549 134 53 178];

app.Csharp4.Text = 'C#';

% Create Asharp3

app.Asharp3 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Asharp3.ButtonPushedFcn = createCallbackFcn(app, @Asharp3ButtonPushed, true);

app.Asharp3.VerticalAlignment = 'bottom';

app.Asharp3.BackgroundColor = [0 0 0];

app.Asharp3.FontColor = [0.9412 0.9412 0.9412];

app.Asharp3.Position = [393 134 53 178];

app.Asharp3.Text = 'A#';

% Create Gsharp3

app.Gsharp3 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Gsharp3.ButtonPushedFcn = createCallbackFcn(app, @Gsharp3ButtonPushed, true);

app.Gsharp3.VerticalAlignment = 'bottom';

app.Gsharp3.BackgroundColor = [0 0 0];

app.Gsharp3.FontColor = [0.9412 0.9412 0.9412];

app.Gsharp3.Position = [322 134 53 178];

app.Gsharp3.Text = 'G#';

% Create Fsharp3

app.Fsharp3 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Fsharp3.ButtonPushedFcn = createCallbackFcn(app, @Fsharp3ButtonPushed, true);

app.Fsharp3.VerticalAlignment = 'bottom';

app.Fsharp3.BackgroundColor = [0 0 0];

app.Fsharp3.FontColor = [0.9412 0.9412 0.9412];

app.Fsharp3.Position = [247 134 53 178];

app.Fsharp3.Text = 'F#';

% Create Dsharp3

app.Dsharp3 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Dsharp3.ButtonPushedFcn = createCallbackFcn(app, @Dsharp3ButtonPushed, true);

app.Dsharp3.VerticalAlignment = 'bottom';

app.Dsharp3.BackgroundColor = [0 0 0];

app.Dsharp3.FontColor = [0.9412 0.9412 0.9412];

app.Dsharp3.Position = [113 134 53 178];

app.Dsharp3.Text = 'D#';

% Create Csharp3

app.Csharp3 = uibutton(app.MUSICCOMPOSINGV16Panel, 'push');

app.Csharp3.ButtonPushedFcn = createCallbackFcn(app, @Csharp3ButtonPushed, true);

app.Csharp3.VerticalAlignment = 'bottom';

app.Csharp3.BackgroundColor = [0 0 0];

app.Csharp3.FontColor = [0.9412 0.9412 0.9412];

app.Csharp3.Position = [42 134 53 178];

app.Csharp3.Text = 'C#';

% Create RECORD

app.RECORD = uibutton(app.MUSICCOMPOSINGV16Panel, 'state');

app.RECORD.ValueChangedFcn = createCallbackFcn(app, @RECORDValueChanged, true);

app.RECORD.Icon = 'record.png';

app.RECORD.Text = '';

app.RECORD.Position = [1558 86 56 58];

% Create Label

app.Label = uilabel(app.MUSICCOMPOSINGV16Panel);

app.Label.Position = [642 24 84 22];

app.Label.Text = '';

% Create FREQUENCYLabel

app.FREQUENCYLabel = uilabel(app.MUSICCOMPOSINGV16Panel);

app.FREQUENCYLabel.Position = [549 24 116 22];

app.FREQUENCYLabel.Text = 'FREQUENCY -';

% Create TIMEEditFieldLabel

app.TIMEEditFieldLabel = uilabel(app.MUSICCOMPOSINGV16Panel);

app.TIMEEditFieldLabel.HorizontalAlignment = 'right';

app.TIMEEditFieldLabel.Position = [770 24 34 22];

app.TIMEEditFieldLabel.Text = 'TIME';

% Create TIMEEditField

app.TIMEEditField = uieditfield(app.MUSICCOMPOSINGV16Panel, 'numeric');

app.TIMEEditField.ValueChangedFcn = createCallbackFcn(app, @TIMEEditFieldValueChanged, true);

app.TIMEEditField.Position = [819 24 100 22];

app.TIMEEditField.Value = 172;

% Create BPMLabel

app.BPMLabel = uilabel(app.MUSICCOMPOSINGV16Panel);

app.BPMLabel.Position = [923 24 32 22];

app.BPMLabel.Text = 'BPM';

% Create INFORMATIONPANELPanel

app.INFORMATIONPANELPanel = uipanel(app.UIFigure);

app.INFORMATIONPANELPanel.ForegroundColor = [0.851 0.3294 0.102];

app.INFORMATIONPANELPanel.Title = 'INFORMATION PANEL';

app.INFORMATIONPANELPanel.FontName = 'Goudy Old Style';

app.INFORMATIONPANELPanel.FontWeight = 'bold';

app.INFORMATIONPANELPanel.FontSize = 20;

app.INFORMATIONPANELPanel.Position = [2 1 1871 469];

% Create phase

app.phase = uiaxes(app.INFORMATIONPANELPanel);

title(app.phase, 'NOISE')

xlabel(app.phase, 'Frequency')

ylabel(app.phase, 'MAGNITUDE')

app.phase.PlotBoxAspectRatio = [1 0.621262458471761 0.621262458471761];

app.phase.Position = [1218 7 652 431];

% Create waveform

app.waveform = uiaxes(app.INFORMATIONPANELPanel);

title(app.waveform, 'FOURIER')

xlabel(app.waveform, 'Times/Sec')

ylabel(app.waveform, 'FOURIER SIGNAL')

app.waveform.PlotBoxAspectRatio = [1 0.692167577413479 0.692167577413479];

app.waveform.Position = [619 1 600 437];

% Create UIAxes

app.UIAxes = uiaxes(app.INFORMATIONPANELPanel);

title(app.UIAxes, 'FREQUENCY')

xlabel(app.UIAxes, 'TIME')

ylabel(app.UIAxes, 'AMPLITUDE')

app.UIAxes.PlotBoxAspectRatio = [1 0.589065255731922 0.589065255731922];

app.UIAxes.Position = [-1 16 621 422];

end

end

methods (Access = public)

% Construct app

function app = PAINOesa

% Create and configure components

createComponents(app)

% Register the app with App Designer

registerApp(app, app.UIFigure)

if nargout == 0

clear app

end

end

% Code that executes before app deletion

function delete(app)

% Delete UIFigure when app is deleted

delete(app.UIFigure)

end

end

end