GUIDE’s GUI Callbacks

- GUIDE’s Callbacks are simply the MATLAB Script files that will be executed when a GUI’s UI is activated. (UIs are activated with a left button mouse click.)

Lesson Objectives
- Handle Graphic Basics
- Writing Callbacks
- The Callback Editor
- Linking UIs w/ code
- The Object Browser
- Edit & Active Modes
- GUI Execution

UIs that can be activated
- Push Button
- Toggle Button
- Radio Button
- Check Box
- Edit Box
- Slider
- List Box
- Popup Menu
The Basics of Handle Graphics

Every graphic object in MATLAB has a what is called a handle. Handles are used to identify any one of these objects. Once the handle is known it will then be possible to determine or change properties of that object using the ‘set’ or ‘get’ function. Handles are assigned/determined a couple of different ways. One way is to use the very popular Utility Functions; gcf, gca and gco for Get the Current Figure, Axis or Object. These handles can be used for any active figure, axis or object. What is it that makes an object active? It’s the last one created or selected.

Example: Change a figure’s color from its default of light gray to blue.

```
>> figure
>> set(gcf,’Color’, [0 0 1])
```

Notice that the handle ‘gcf’ is the 1st input argument, that the property ‘Color’ is the 2nd and the RGB matrix (the new color) is the 3rd.
Two other figure properties that can be changed with the ‘set’ function are Units and Position. One convenient use of these two properties is place 4 figures on the screen so they can all be viewed without using the mouse. Change the Units property from its default to ‘Normalized’ so that problems don’t occur when screen sizes change. Positions can be set with a 1x4 row vector. Each element represents the following: [Left Bottom Length Height] Remember that the units are now normalized so each element here needs to be between 0 and 1.

```matlab
>> figure(1)
>> set(gcf,'Units','normalized')
>> set(gcf,'Position',[0.01 0.53 0.48 0.35])
>> figure(2)
>> set(gcf,'Units','normalized')
>> set(gcf,'Position',[0.5 0.53 0.48 0.35])
>> figure(3)
>> set(gcf,'Units','normalized')
>> set(gcf,'Position',[0.01 0.05 0.48 0.35])
>> figure(4)
>> set(gcf,'Units','normalized')
>> set(gcf,'Position',[0.5 0.05 0.48 0.35])
```
To obtain a list of the properties that are available for each graphics object simply go to the Property Inspector and scroll the list. Don’t feel that you have to master all of these properties. There are 10-20 that are used often but most are rarely used. Notice that the names of the properties are listed along the left side. Be sure to use these names verbatim (case insensitive) in the code.
Using the ‘get’ function.

Any UI that can be activated has a property called ‘Value’. This property allows the user to determine if, as an example, a radio button (RB) has been turned on or off. (RBs have values of either 0 or 1.) Assume the handle of a RB is called ‘h’. The code would then be: `>> RBValue = get(h,’Value’);`

Obtaining values is often done when using callbacks.

UIs that have ‘Values’.

- Push Button
- Toggle Button
- Radio Button
- Check Box
- Edit Box
- Slider
- List Box
- Popup Menu
The ‘Tag’ property and the function ‘findobj’.

Another property available to the user is the ‘Tag’ property. The ‘Tag’ is often used to obtain a handle for UI by using the function ‘findobj’. The UI’s Tag has a default. Normally this default is changed so that the user has an easier time selecting the correct UI when using the Callback Editor. (The Callback Editor will be discussed later on in this lesson plan.)

Notice in this example that the Tags for the two Radio Buttons are called TimeDomainPlot and FreqDomainPlot. The handles are the two variables that are assigned.
Callbacks

Callbacks are simply ordinary script files. Any script *.m file in MATLAB can be used as a callback. Once the script file is completed, all that is needed to make it a callback is to call it in the Callback Editor by typing the filename in the correct location. (To be explained later in this lesson plan.)

Notice how the Tags are used here to obtain the handles.

In this example (which is one of the callback files used in the GUI built in the previous lesson plan) the first 12 executable lines of code assign handles to each of the 12 touch tone pushbuttons.
Callbacks – cont.

Understand that when writing callbacks, the MATLAB syntax and use of the MATLAB editor remains the same as if there wasn’t a GUI involved. The same is true when using Handle Graphics.

From the callback used as an example in the previous slide, the next 12 lines of executable lines of code are to determine the value of each push button. What helps is that only one can be ‘on’ at a time.
The Touch Tone Keypad

A brief respite from callbacks to explain the Touch Tone Keypad. The keypad is comprised of 4 rows and 3 columns. Each row represents a different frequency under 1kHz. Each column represents a different frequency over 1kHz. Each button’s tone, then, is comprised of the sum of two sinusoids each containing a single frequency corresponding to a particular row and column.

- Row 1: 697 Hz
- Row 2: 770 Hz
- Row 3: 852 Hz
- Row 4: 941 Hz
- Column 1: 1209 Hz
- Column 2: 1336 Hz
- Column 3: 1477 Hz
Determining which touch tone is touched

With the handles and values of the relevant UIs determined, code needs to be written to generate the proper signal that will correspond to the activated keypad button.

- Notice that the code here is quite simple and straightforward.

- Remember that once a “true” is found, the keypad button activated has been determined and the variable ‘Tone’ is assigned.
Determine the handles and values of the radio buttons.
The GUI that was built in the last lesson plan offered the option of time-domain or frequency-domain plots. Their radio button values will need to be determined to decide which data to plot.

- Again, the code is easy
- Four lines and they’re fairly straightforward.
- First use the Tags to get handles…
- then use handles to obtain the Values.
Generating and Plotting the data
Once all of the relevant handles and values are determined, the data can then be generated and put on a plot to display the output.

The code in this example generates and plots the output data as well as ensuring that if the Time Domain RB is clicked ‘on’ then the Frequency Domain RB will be turned off and vice-versa.
The Option to Hear the Tone.

The GUI that was built in the last lesson plan provided a radio button asking the user if the tone wanted to be heard. The code for this takes just 5 lines of code as shown. (Lines 96-100)

The callback used here will be executed whenever any one of the 12 buttons in the keypads is activated. This means that the filename will have to be mentioned 12 times in the Callback Editor.
Making Radio Buttons Mutually Exclusive

Radio Buttons (RBs) are used most often in batches or groups in such a way that only one of them can be ‘on’ at a time. If this is the case, the RBs are then know to be mutually exclusive. The current GUI has a very good example of this with the Time Domain Plot RB and the Frequency Domain Plot RB. Obviously there can only be one of these ‘on’ at a time as this is an either or situation.

The code here represents the script file for the Time Domain RB. This code will be executed whenever this RB is activated. Notice that the code makes sure that whenever one is on, the other is off.
The Other Radio Button Callback

In the previous slide the callback for the Time Domain Plot radio button was discussed. Below is the code for the Frequency Domain Plot. (Different filenames, of course.) Notice that the code is very similar between the two.

So the entire file is comprised of 11 executable lines of code following the usual pattern; get the handles using the Tag, get the values using the handles then compare the values to execute the proper code.
The Clear Button Callback

The callback below represents the script *.file for the Clear Button on the current GUI. Yes, the 3 lines of code could have been written directly into the Callback Editor. Either way works just as well, it’s simply a matter of choice.

- The first line of code clears the axis.
- The second line sets the ‘XTickLabel’ to an empty set.
- The third line sets the ‘YTickLabel’ to an empty set.

FYI - That’s the end of the callback code.
The Callback Editor
Once all of the callback files have been written and saved, the Callback Editor can be used to link the code with the GUI. Right click on the “1” button and select the Edit Callback option.

- The GUI’s *.m file will be returned as shown. The blue highlighted part will correspond to whichever UI has been chosen with the right mouse click.
- A very specific Tag name helps out here.
The Call Editor – cont.

Comment out the ‘disp’ line and type in the filename KeyPadCallbackECE201 in the subfunction called ‘Number1_callback’ as shown. Instead of right mouse clicking on every UI, scroll down to repeat the process for the 11 other ‘Number*’ subfunctions. Ctrl-S to save.

- Notice how the Tag name is used in the Callback’s subfunction name.
- Understand that the order of the subfunctions really isn’t relevant.
More on the Callback Editor...

Now that the callback file called ‘KeyPadCallbackECE201’ has been entered into all 12 subfunctions whose names begin with ‘Number’, enter in the callbacks for the other UIs.

- Be sure to comment out all of the code that begins with the ‘disp’ function as shown.
- Be sure to save the file once you are done. (Ctrl-S)
Finishing up with the Callback Editor

Scroll down within GUI1.m and enter in the 3 remaining callback filenames in their appropriate places as shown.

- Do not enter in a callback for the subfunction SoundRadioButton* as it won’t need code to be executed when it is activated.
- Save using Ctrl-S and close the editor.
The Object Browser & Handle Visibility

The Object Browser can be opened by clicking on the 2nd button from the right on the option bar. Once opened, double-click on the word ‘figure’ and its Property Inspector will open. Change the HandleVisibility property from ‘off’ to ‘callback’. Doing so will tell MATLAB to plot data on the axis in the GUI. If not, when callback code containing the plot function is executed, MATLAB will open a figure outside of the GUI and plot the data there. Go back to edit version of this GUI (the one in GUIDE) and click on the Save button (3rd from left on the option bar) to save.
Activate Figure

The Activate Figure button is the last button on the right on the option bar. Clicking on this button will open an “active” or executable version of the GUI in another window. This will be the GUI that the user will use.

GUI1.fig is the GUI in the Edit Mode. To open this GUI in the Active Mode (outside of GUIDE), simply type

```
>> GUI1
```

at the MATLAB prompt.
GUI Execution

Once in the Active Mode, the user can use the GUI as it was designed. Any edits, corrections or debugging can be done back in the Edit Mode by going back to GUIDE. Consider GUI1, both Layout and Callbacks, now complete.

- When plotting frequency domain data, notice that each signal is comprised of two frequencies, one always being less than 1000 Hz and the other always more than 1000 Hz.
- Experiment as you wish.