More About Matlab GUI's (v. 1.1)

Dr. C. S. Tritt
with some slides from
Dr. J. A. LaMack
January 11, 2012

What's Coming

- In this show I plan to fill in many of the details regarding GUI programming in Matlab
- Specifically I will cover:
  - Some important *figure* and *uicontrol* properties.
  - The general usage of some specific uicontrols.
  - Some examples of these uses.

2

Specific uicontrol Uses

- GUI controls are designed for specific uses. Use them correctly.
- Static Text – Labels and output.
- Edit Boxes – User input and output.
- Pushbuttons – Cause events to occur.
- Toggle Button – User mode specification.
- Check Boxes – Option specification.

3

Specific Uses (continued)

- Radio Button & Button Groups – Specification of mutually exclusive options.
- Programmatically control possible:
  - Popup Menu – Specification of one item from a list.
  - List Boxes – Specification of one or more items from a list.
- Sliders – Specification of a numeric value within a range.

4

Getting and Setting

- You will use the `get` function to obtain the specifics of the user's input and
- The `set` to change an attribute of the same or a different component (possible because we pass around component handles in in the GUIDE created *handles* structure).

5

Changing Control Properties

- It is generally possible to get or change the value of any properties for any control for which you have a handle (either using *hObject* or the *handles* structure and the control's *Tag*).
- Particularly useful properties include *Enable*, *ForegroundColor*, *String* and *Value*.

6

Some Figure Properties

Table 10.2 Important `figure` Properties

Property	Description
`Color`	Specifies the color of the figure. The value is either a predefined color such as `'r'`, `'g'`, or `'b'`, or else a 3-element vector specifying the red, green, and blue components of the color on a 0–1 scale. For example, the color magenta would be specified by `[1 0 1]`.
`CurrentCharacter`	Contains the character corresponding to the last key pressed in this figure.
`CurrentPoint`	Location of the last button click in this figure, measured from the lower left-hand corner of the figure in units specified in the `Units` property.
`Dockable`	Specifies whether or not the figure can be docked to the desktop. Possible values are `'on'` or `'off'`.
`MenuBar`	Specifies whether or not the default set of menus appears on the figure. Possible values are `'figure'` to display the default menus or `'none'` to delete them.
`Name`	A string containing the name that appears in the title bar of a figure.
`NumberTitle`	Specifies whether or not the figure number appears in the title bar. Possible values are `'on'` or `'off'`.

(continued)

7

Figure Properties (continued)

Table 10.2 *(continued)*

Property	Description
`Position`	Specifies the position of a figure on the screen, in the units specified by the `'units'` property. This value accepts a 4-element vector in which the first two elements are the x and y positions of the lower left-hand corner of the figure and the next two elements are the width and height of the figure.
`SelectionType`	Specifies the type of selection for the last mouse click on this figure. A single click returns type `'normal'`, while a double click returns type `'open'`. There are additional options; see the MATLAB on-line documentation.
`Tag`	The "name" of the figure, which can be used to locate it.
`Units`	The units used to describe the position of the figure. Possible choices are `'inches'`, `'centimeters'`, `'normalized'`, `'points'`, `'pixels'`, or `'characters'`. The default units are `'pixels'`.
`Visible`	Specifies whether or not this figure is visible. Possible values are `'on'` or `'off'`.
`WindowStyle`	Specifies whether this figure is normal or modal (see discussion of Dialog Boxes). Possible values are `'normal'` or `'modal'`.

8

Some uicontrol Properties

Table 10.3 Important `uicontrol` Properties

Property	Description
`BackgroundColor`	Specifies the background color of the object. The value is either a predefined color such as `'r'`, `'g'`, or `'b'`, or else a 3-element vector specifying the red, green, and blue components of the color on a 0–1 scale. For example, the color magenta would be specified by `[1 0 1]`.
`Callback`	Specifies the name and parameters of the function to be called when the object is activated by a keyboard or text input.
`Enable`	Specifies whether or not this object is selectable. If it is not enabled, it will not respond to mouse or keyboard input. Possible values are `'on'` or `'off'`.
`FontAngle`	A string containing the font angle for text displayed on the object. Possible values are `'normal'`, `'italic'`, and `'oblique'`.
`FontName`	A string containing the font name for text displayed on the object.
`FontSize`	A number specifying the font size for text displayed on the object.
`FontUnits`	The units in which the font size is defined. Possible choices are `'inches'`, `'centimeters'`, `'normalized'`, `'points'`, and `'pixels'`. The default font units are `'points'`.
`FontWeight`	A string containing the font weight for text displayed on the object. Possible values are `'light'`, `'normal'`, `'demi'`, and `'bold'`. The default font weight is `'normal'`.
`ForegroundColor`	Specifies the foreground color of the object.

(continued)

9

uicontrol Properties (cont.)

Table 10.3 (continued)

Property	Description
`HorizontalAlignment`	Specifies the horizontal alignment of a text string within the object. Possible values are `'left'`, `'center'`, and `'right'`.
`Max`	The maximum size of the `value` property for this object.
`Min`	The minimum size of the `value` property for this object.
`Parent`	The handle of the figure containing this object.
`Position`	Specifies the position of the object on the screen, in the units specified by the `'units'` property. This value accepts a 4-element vector in which the first two elements are the *x* and *y* positions of the lower left-hand corner of the object *relative the figure containing it*, and the next two elements are the width and height of the object.
`Tag`	The "name" of the object, which can be used to locate it.
`TooltipString`	Specifies the help text to be displayed when a user places the mouse pointer over an object.
`Units`	The units used to describe the position of the figure. Possible choices are `'inches'`, `'centimeters'`, `'normalized'`, `'points'`, `'pixels'`, or `'characters'`. The default units are `'pixels'`.
`Value`	The current value of the `uicontrol`. For toggle buttons, check boxes, and radio buttons, the value is `max` when the button is on and `min` when the button is off. Other controls have different meanings for this term.
`Visible`	Specifies whether or not this object is visible. Possible values are `'on'` or `'off'`.

10

Regarding Color Properties

- The uicontrols to have two color related properties. These are:
  - *ForegroundColor* – The color of any text displayed in the control.
  - *BackgroundColor* – The color around and behind the text.
- In addition to foreground and background colors, Panels have *HighlightColor* and *ShadowColor* properties.
- The figure in which controls and panels reside have a single color property:
  - *Color* – The background color of the figure.

11

Specifying Colors

Color values can generally be specified as a row vectors containing red, green and blue values (between 0 and 1), a short string or a long string (as shown below).

RGB Value	[0 0 0]	[1 0 0]	[0 1 0]	[0 0 1]	[1 1 0]	[1 0 1]	[0 1 1]	[1 1 1]
Short String	k	r	g	b	y	m	c	w
Long String	black	red	green	blue	yellow	magenta	cyan	white

12

Callbacks

- Callbacks are functions that are executed when a GUI control changes state.
- The specific conditions under which callbacks are executed depends on the type of control involved.
- For example, edit boxes execute their callbacks when they lose focus, while sliders call theirs when the mouse button is release after the "slider bar" has been moved.

13

Focus

- Focus is an important concept with respect to GUI programming.
- When the user takes an action, like pressing a key, the program needs to know what control it should be routed it to. The control to which input is sent is said to have focus.
- Having focus is generally indicated by the presence of a flashing cursor or some type of highlighting.

14

Callbacks (continued)

- Callbacks can have any name and are "registered" with controls using their handles.
- GUIDE allows specification of callback functions by name as a property of a particular control.
- By default, GUIDE creates callback names by pre-pending the control's Tag.
- For example, calcButton_Callback.

15

Other Control Functions

- Most `uicontrols` can be configured to call other functions in response to certain conditions.
- These include `SelectionChangeFcn`, `ButtonDownFcn`, `CreateFcn`, `DeleteFcn`, and `KeyPressFcn`.
- These functions are useful in special cases, but generally need not be used in simple GUI programs.

16

SelectionChangeFcn

- Button Group panels call particular functions when buttons they contain change state.
- These functions are can only be named *name_SelectionChangeFcn* where *name* is the name of the button group.
- Note that the *hObject* passed to the *SelectionChangeFcn* is the handle of the selected control (not the group).

17

GUIDE's *OpeningFcn*

- GUIDE creates a special function called *myname*_OpeningFcn where *myname* is the name of the GUI.
- This function is called just before the GUI is made visible to the user. Place code to be executed once at the start of the program in it.
- Its arguments are always: *hObject*, *eventdata*, *handles*, *varargin*.

18

GUI Function Arguments

- GUI functions (like callbacks) are always called with the same set of arguments. These are:
  - *hObject* – The object that caused the function to be called.
  - *eventdata* – key pressed and reserved for use in a future version of MATLAB.
  - *handles* – a structure containing the handles of all the objects in a figure.

19

Changing Gears

- In the following slides, I'll cover some of the details of using Matlab's graphical user interface controls (*uicontrols*) and panels (*uipanels*).
- Static text, edit boxes, push buttons, button groups, on-off buttons (like check boxes, radio buttons and toggle buttons) and list boxes will be discussed and demonstrated.

20

Static Text Fields

- We've already used this in an example.
- Generic way to display info
- Most common property to change is `'String'`
- The String Property can be set to a single string (for one line of output) or a cell array of strings (for multiple lines).

21

Edit Boxes

- Allows user to enter one or more lines of text.
- The primary property of interest is `'String'`, which is generally gotten but it can be *set*.
- If numbers are expected, remember to use `str2double` to convert the returned string to a number.

22

Single Line Input

- Set both the `min` and `max` properties to 1.
- As soon as the user hits *Enter* or *Esc* after typing, the callback is invoked.

23

Multiple Line Input

- Set the `max` and `min` properties such that `max` – `min` is greater than 1.
- A scroll bar will be included.
- If the user hits *Enter*, cursor will move down to next line. As soon as the user hits *Esc* after typing (or selects another control), the callback is invoked.
- The *String* Property will be a cell array of strings.

24

Practice with Edit Boxes

- Create a GUI called 'MySecondGUI2'
- Include a static text field called 'StaticText' which initially contains the text 'Initial Text' (make the box relatively tall).
- Include an edit box called 'EditBox' initially containing the text (don't forget to change `min` and `max`):

  Enter
  Data
  Here

25

Practice with Edit Boxes

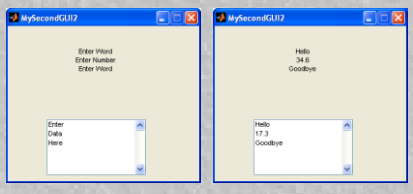
- Modify the m-file so that before it opens, the static text field becomes:

  Enter Word
  Enter Number
  Enter Word

- When the user enters data, echo the input to the static text field, except double the number.

26

Edit Box Example Results



27

'N Sync

- It is important to keep displayed data synchronized.
- See my "MAPCalculator" program handout for a demonstration of how to deactivate and clear Edit Boxes so what's in the boxes always matches what's elsewhere on the screen.

28

Pushbuttons

- We've already used these.
- Uncommon to get or set any properties associated with it during program (except perhaps the `'String'`).

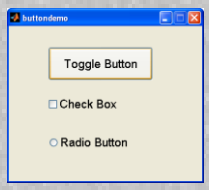
29

On and Off Buttons

- Include toggle buttons, checkboxes, and radio buttons
- All do the same thing and operate the same way
- Commonly associated with different tasks
- Most common property to get is `'Value'`
  - Button on: `'Value'` = `Max` (usually 1)
  - Button off: `'Value'` = `Min` (usually 0).
- Activate their callbacks when they change state.

30

On and Off Button Appearances



31

Single Check Box Demo

- Create a GUI program that changes the color of a Static Text message (tagged *theText* and containing the string *Full blood test* from black to red when a Check Box (tagged rushIt and labeled *Rush*) is selected.

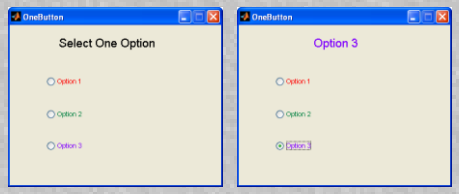
32

Mutually Exclusive Radio Buttons Example

- Create a GUI called ColorRadio that has three radio buttons given different colors (specified using the Property Inspector during figure creation), labeled *Option 1*, *Option 2*, and *Option 3*.
- Allow only one button to be selected at any time, and display which in a static text field in the same color as the selected option.
- Note this is most efficiently done using a button group (See my "Button Group Demo" handout).

33

Button Example Output



34

List Boxes

- User selects one item from a mutually exclusive list of options
- The `'String'` property is typically the only property that is set.
  - Set during creation using the "matrix" editor.
  - Set in functions with the `set` command; must be a cell array of strings (each string is an option).
- Get the `'Value'` property, which is a number representing the option choice.

35

Populating Lists

- The choices displayed in popup menus and list boxes consist of a cell array of strings stored as the controls' String property.
- These strings can be entered in GUIDE by doubling clicking on the "matrix" icon next to the String property in the property inspector.

36

List Boxes

- I had some problems drawing these in GUIDE, so I had to manually set size using *Position* property.
- The first option is always selected by default.
- Specific values of *max* and *min* are irrelevant as long as max – min > 1 to enable multiple selections.
- Note that Matlab Pop-up Menus work much like List Boxes but only allow a single item to be selected.

37

List Box Exercise

- Create a new GUI project using GUIDE.
- Add a List Box that contains the four color names:
  Red, Green, Blue, Black
- When one of these is selected, display the name of the color in its appropriate color.

38

Altering the Availability of Components

- Not covered well in Chapman's book.
- Very often in GUI design, the ability to use certain components (like buttons) often should change as the user proceeds.
- The 'enable' property of components controls availability.
  - Set to 'on' to enable, 'off' to disable.

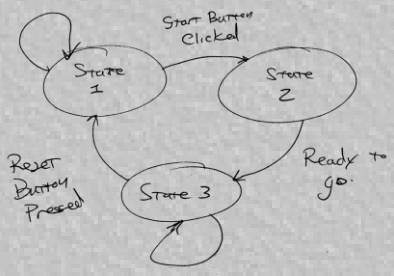
39

State Diagram Approach

- Using state diagrams often makes it easier to think about GUI software designs.
- Define states to describe which components are enabled and disabled
- Connect the states with all actions that the user could possibly take
- States can be used to define any program status (not just availability of components)

40

Example of a State Diagram



41

Exercise

- Modify the ColorRadio GUI to now contain a reset push button (called "ResetButton")
- Save it as ColorRadioPlus.fig
  - The original code from OneButton should transfer to the new .m file (make sure it does).

42

Add the Following Features

- When the program starts, only the radio buttons should be selectable, and they should all be cleared. "Select one option" should appear in black.
- When a user selects a radio button, in addition to the textbox changing as before, the popup menu should become available and the radio buttons should become unavailable

43

More Added Features

- When the user then selects something from the popup menu, in addition to the text box changing as before, the radio buttons should become available again, and the reset should become available.
- The GUI should work as before at this point.
- If the Reset button is pressed, the program should revert back to its condition when the program started.

44

Sources

- Some material (including the tables of properties) in the show were obtained from Stephen Chapman's "Matlab Programming for Engineers, 3rd ed."

45