blinkdagger

an Engineering and MATLAB blog

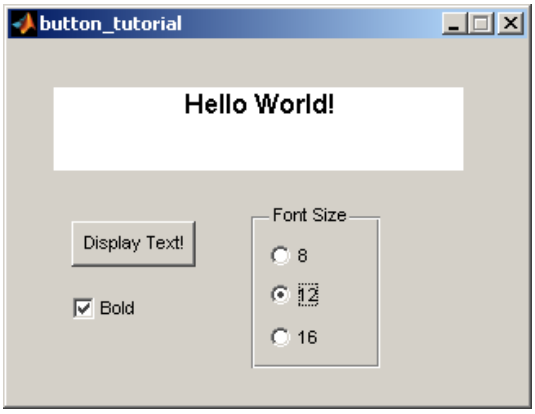
- Home
- Listchecker
- MATLAB
- Contact
- About

MATLAB GUI Tutorial - Button Types and Button Group

03 Nov 2007 *Quan Quach* 93 comments 25,002 views

Introduction

In this three-part Matlab GUI Tutorial, you will learn how to use the different types of buttons available within Matlab GUIs. These button types are: push button, radio button, check box, and toggle buttons. In addition, you will learn how to use the button panel to control a group of buttons.

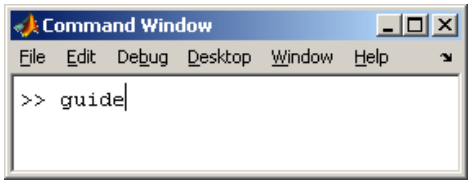


This tutorial is written for those with little or no experience creating a Matlab GUI (Graphical User Interface). If you're new to creating GUIs in Matlab, you should visit this tutorial first. Basic knowledge of Matlab is recommended. Matlab version 2007a is used in writing this tutorial. Both earlier versions and new versions should be compatible as well (as long as it isan't too outdated). Let's get started!

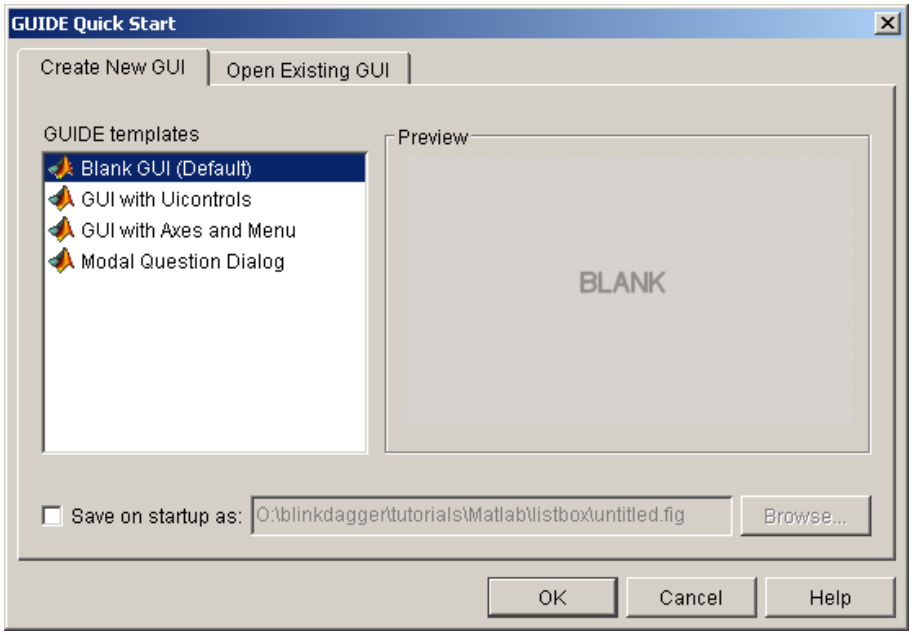
Part One: The Pushbutton

The push button is a very simple component. When the user clicks on a push button, it causes an action to occur. This action is dictated by the code that is programmed in the push button's callback function. In this part of the tutorial, we will program the push button to display some text when it is pressed.

1. First, we are going to create the visual aspect of the GUI. Open up Matlab. Go to the command window and type in guide.

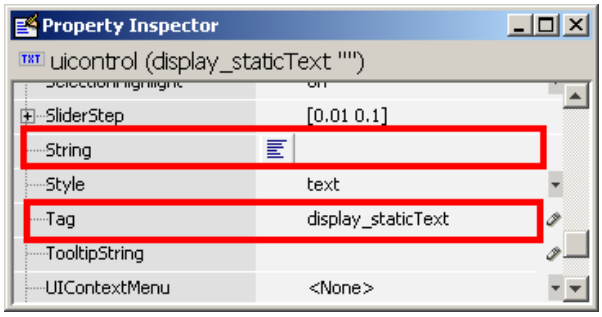


2. You should see the following screen appear. Choose the first option Blank GUI (Default).

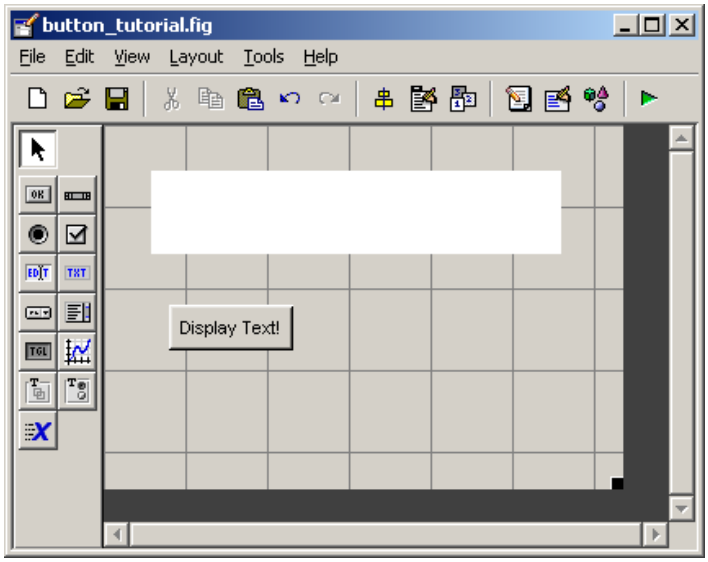


3. Click on and add one *Static Text* component to the GUI figure. Next, click on and add one *Push button* component onto the GUI figure.

4. Double click the *Static Text* component to bring up the Property Inspector. Change the *String* property so that there is nothing inside. Change the *Tag* property to display_staticText. Similarly, double click on the *Pushbutton* component and change the *String* property to Display Text! and change the *Tag* property to displayText_pushbutton.



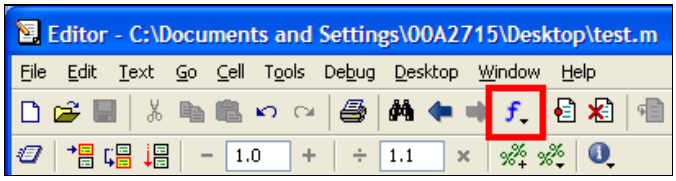
5. Here's what your figure should look like after you add the components and modify them.



6. Save your GUI wherever you please with your desired filename.

7. Now, we are going to write the code for the GUI. When you save your GUI, Matlab automatically generates an .m file to go along with the figure that you just put together. The .m file is where we attach the appropriate code to the callback of each component. For the purposes of this tutorial, we are primarily concerned only with the callback functions. You don't have to worry about any of the other function types.

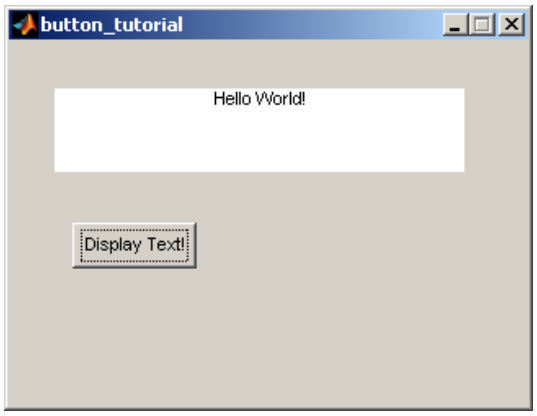
   Open up the .m file that was automatically generated when you saved your GUI. In the Matlab editor, click on the  icon, which will bring up a list of the functions within the .m file. Select *displayText_pushbutton_Callback*.



   Add the following code to the function:

```
%display "Hello Word!" in the static text component when the
%pushbutton is pressed
set(handles.display_staticText,'String','Hello World!');
```

8. Now that we've completed both the visual and code aspects of the GUI, its time to run the GUI to make sure it works before we move on. From the m-file editor, you can click on the  icon to save and run the GUI. Alternatively, from the GUIDE editor, you can click on the  to launch the GUI. The GUI should appear once you click the icon. Now try clicking on the button to make sure that Hello World! appears on the GUI.



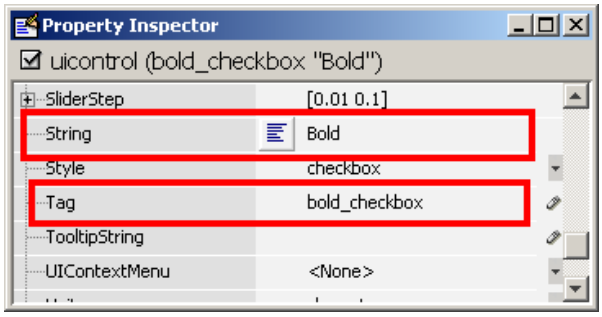
And that's it. Those are the basics of using the *Push button* component. Now we're ready to move onto the *Check box* component.

Part Two: The Check Box

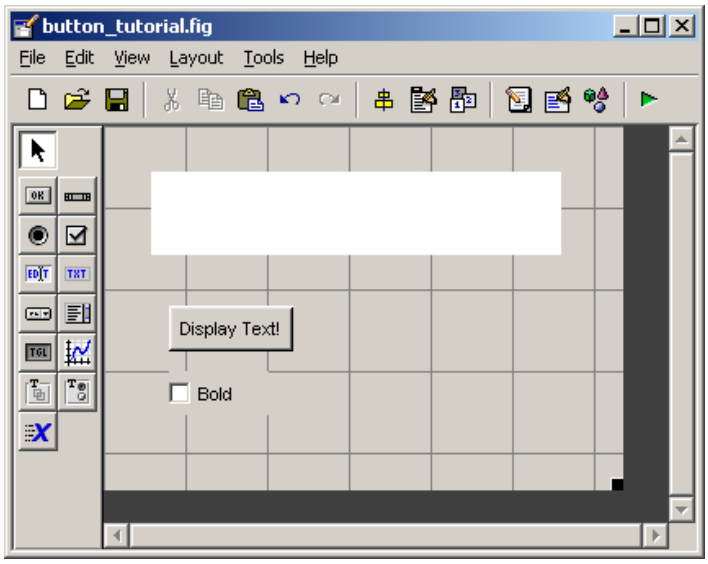
The *Check Box* component has two states, *checked* and *unchecked*. These components are usually used to control options that can be turned *on* and *off*. In this part of the tutorial, we will add the functionality of making the display text become bold or unbolded.

1. The first thing we need to do is to add a *Check Box* Component to the GUI figure that we were just working with. So if you closed GUIDE, reopen it again. Once you have GUIDE opened again, Click on  and add one *Check Box* component to the GUI figure.

2. Double click the *Check Box* component to bring up the Property Inspector. Change the *String* property to Bold. Change the *Tag* property to bold_checkbox.



3. Here's what your figure should look like after you add the *Check Box* component and modify it.

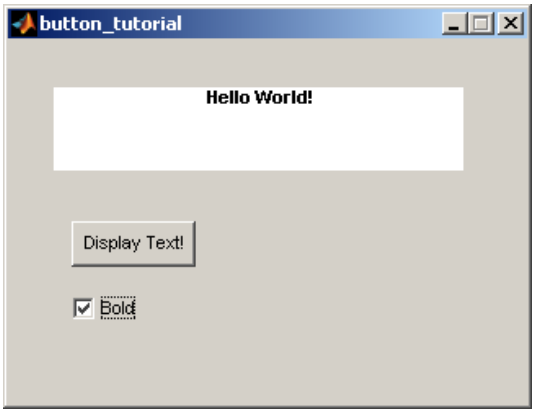


4. Add the following code to the *bold_checkbox_Callback* function:

```
%checkboxStatus = 0, if the box is unchecked,
%checkboxStatus = 1, if the box is checked
checkboxStatus = get(handles.bold_checkbox,'Value');
if(checkboxStatus)
   %if box is checked, text is set to bold
   set(handles.display_staticText,'FontWeight' , 'bold');
else
   %if box is unchecked, text is set to normal
   set(handles.display_staticText,'FontWeight', 'normal');
end
```

Note: The *bold_checkbox_Callback* function triggers when the user activates the check box **AND** when the user deactivates the check box.

5. Now that we've completed both the visual and code aspects of the GUI, its time to run the GUI to make sure it works before we move on. Try checking and unchecking the *Check Box* component to make sure that the text "Hello World!" is being bolded and unbolded.



And that's it. Those are the basics of using the *Check Box* component. Now we're ready to move onto the *Button Group*, which is the most challenging part of this tutorial.

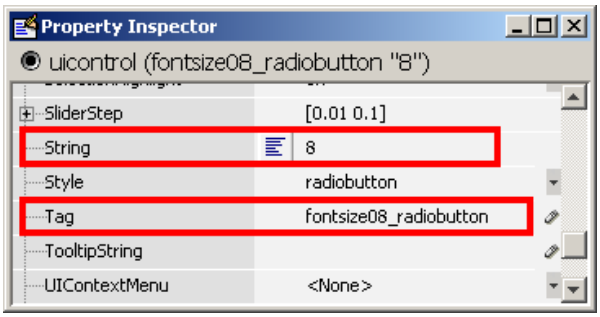
Part Three: Radio Buttons, Toggle Buttons, and Button Group Panel

Radio buttons and Toggle buttons are used exactly the same way that check boxes are used in Matlab GUIs, so we won't go over how to use them. But there is one special case that needs to be covered. When either radio buttons or toggle buttons are used in conjunction with the button group panel, they exhibit mutually exclusive behavior. Simply put, this means that only one radio button or one toggle button can be selected at a time. This behavior can come in very useful for some GUIs. Since radio buttons and toggle buttons are identical in their functionality, what is said about one, is true for the other. Thus, only radio buttions will be discussed from here on out.

In this part of the tutorial, we will create a button group that will allow you to choose between different font sizes for the display text.

1. The first thing we need to do is to add a *Button Panel* component to the GUI figure that we were just working with. So if you closed GUIDE, reopen it again. Once you have GUIDE opened again, click on  and add one *Button Panel* component to the GUI figure. Make sure it's large enough to fit in three radio buttons. Next, click on  and add three radio buttons onto the button group panel.

2. Double click on the first *Radio Button* component to bring up the Property Inspector. Change the *String* property to 8. Change the *Tag* property to fontsize08_radiobutton.

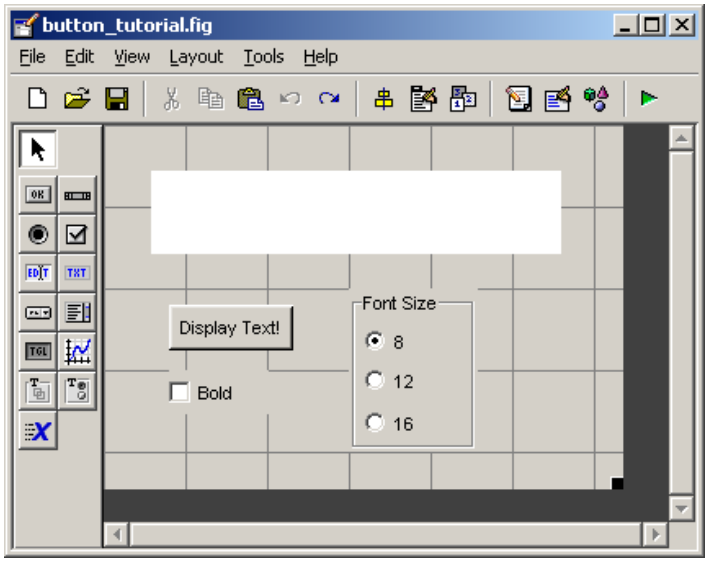


Next, double click on the second *Radio Button* component, and change the *String* property to 12, and change the *Tag* property to fontsize12_radiobutton.

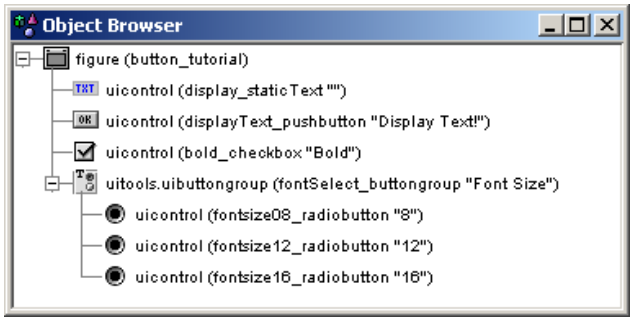
Next, double click on the third *Radio Button* component, and change the *String* property to 16, and change the *Tag* property to fontsize16_radiobutton.

Finally, double click on the button group panel and change the *Tag* property to fontSelect_buttongroup. You should also change the *String* property for the button group panel to Fontsize.

3. Here's what your figure should look like after you add the components and modify them.



4. Before we move on, we should check the hierarchical structure of the GUI figure. Click on the  icon and the followinging should appear:



Make sure that the three radio buttons are one hierarchy below the button group icon.

5. Add the following line of code to the opening function. In this tutorial example, it is named *button_tutorial_OpeningFcn* function. Yours will be the name of the file you saved it as, followed by "_OpeningFcn".

```
set(handles.fontSelect_buttongroup,'SelectionChangeFcn',@fontSelect_buttongroup_SelectionChangeFcn);
```

Make sure the previous line was added right before the line:

```
guidata(hObject, handles);
```

Next, add the following function at the very end of the .m file.

```
function fontSelect_buttongroup_SelectionChangeFcn(hObject, eventdata)

%retrieve GUI data, i.e. the handles structure
handles = guidata(hObject);

switch get(eventdata.NewValue,'Tag')   % Get Tag of selected object
   case 'fontsize08_radiobutton'
     %execute this code when fontsize08_radiobutton is selected
     set(handles.display_staticText,'FontSize',8);
```

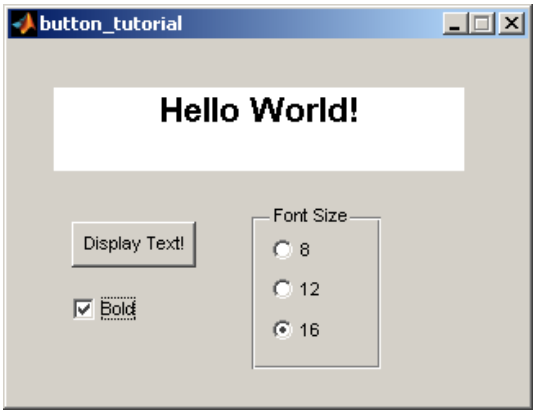
```
    case 'fontsize12_radiobutton'
      %execute this code when fontsize12_radiobutton is selected
      set(handles.display_staticText,'FontSize',12);

    case 'fontsize16_radiobutton'
      %execute this code when fontsize16_radiobutton is selected
      set(handles.display_staticText,'FontSize',16);
    otherwise
       % Code for when there is no match.

end
%updates the handles structure
guidata(hObject, handles);
```

6. Notice that the callback functions for the radio buttons were not automatically generated by Matlab. This is completely normal. Each time a button is selected within the *Button Group Panel* component, the function defined within the *SelectionChangeFcn* property of *Button Group Panel* component is called. The line of code that was added in the opening function specifies the callback function when a button within the button group is selcted. The selection change function is then defined at the end of the .m file.

7. Now that we've completed both the visual and code aspects of the GUI, its time to run the GUI again. Try clicking on all of the buttons to make sure they perform their function correctly. Specifically, make sure that the font size changes accordingly.



And that's it. Those are the basics of using the different buttons within the Matlab GUI.

Download Source Files

Source files can be downloaded here.

This is the end of the tutorial.

93 Responses to "MATLAB GUI Tutorial - Button Types and Button Group"

1. *on 14 Dec 2007 at 1:36 am 1Tongtong*

   A very good tutorial for the beginners. Thanks.

2. *on 14 Dec 2007 at 2:09 am 2Tongtong*