blinkdagger

an Engineering and MATLAB blog

- Home
- Listchecker
- MATLAB
- Contact
- About

MATLAB GUI Tutorial - A Brief Introduction to handles

13 Nov 2007 *Quan Quach* 58 comments 12,689 views

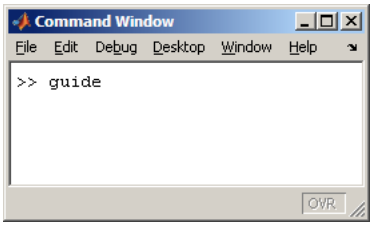
Introduction

In this tutorial, we will discuss what *handles* are and how to use the get and set commands. When dealing with a Matlab GUI, you have probably noticed the variable *handles* being used in the accompanying .m file. The *handles* structure contains all the information for each object in the .fig file of the GUI. But it can also store other information. Let's explore a little bit more about *handles*.

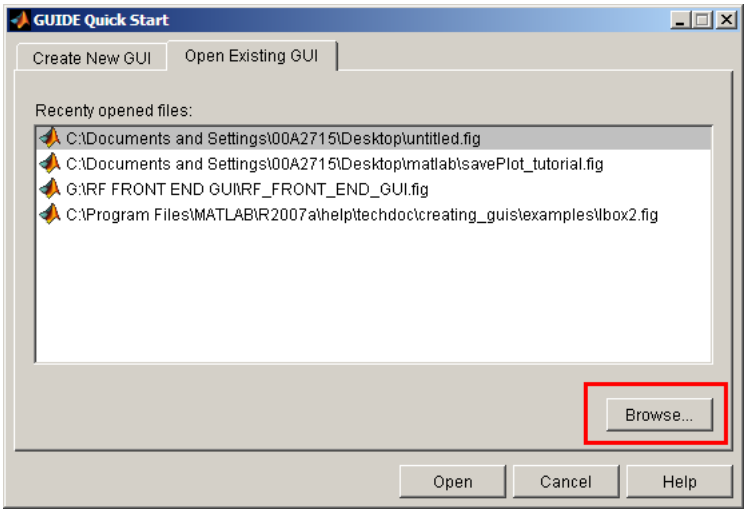
This tutorial is written for those with some experience creating a Matlab GUI. If you're new to creating GUIs in Matlab, you should visit this tutorial first. Basic knowledge of Matlab is recommended. Matlab version 2007a is used in writing this tutorial. Both earlier versions and new versions should be compatible as well (as long as it isan't too outdated).

Handles, get, and set

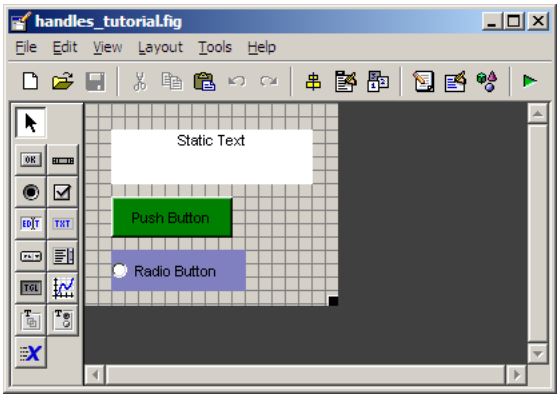
1. First, download the sample GUI here. Unzip the files and place them wherever you please.
2. Now, type guide at the command prompt.



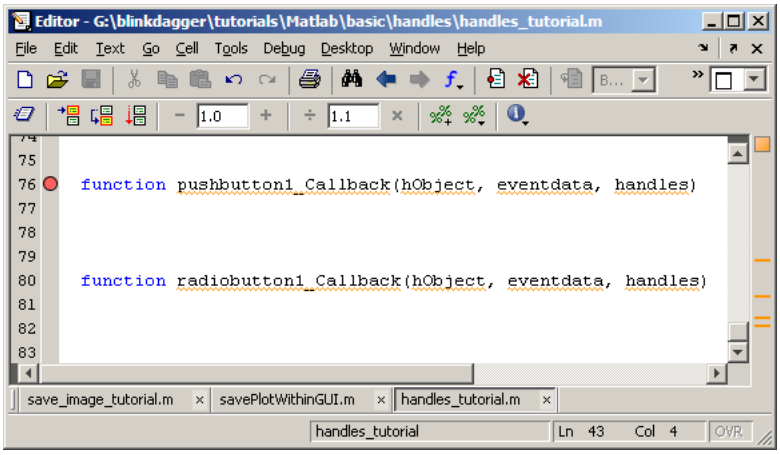
3. Choose to open the sample GUI by clicking on "Open Existing GUI". Click on "Browse" to locate where you saved the GUI files.



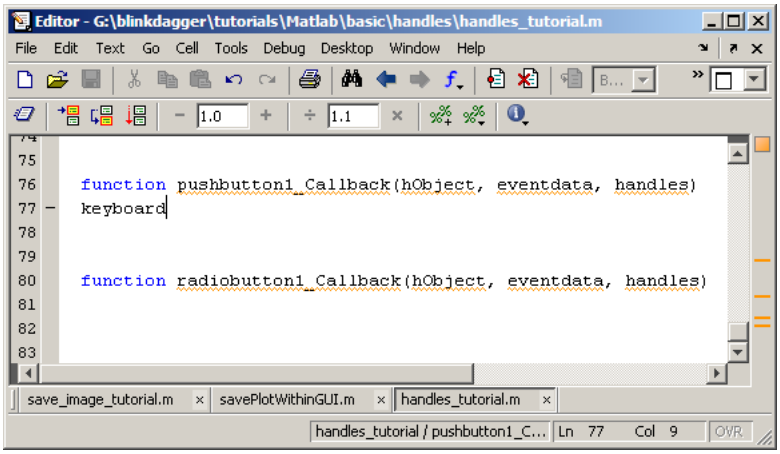
4. Here is what the GUI should look like when you open it:



5. The handles structure contains all the information for the push button, radio button, static text, the figure itself, as well as output. So how exactly do we view and access this information? There are two ways to do this. Click on the  icon on the GUI figure to bring up the accompanying .m file. You can insert in a breakpoint by left clicking on the left side of the .m file as shown.



Or you can type in keyboard into the .m file, as shown below.

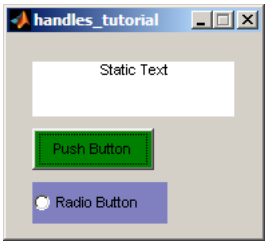


Both methods result in the same thing. This causes the GUI to go into command line mode, allowing you to examine and change the function's local workspace. (Incidentally, this is a great way to debug your code!)

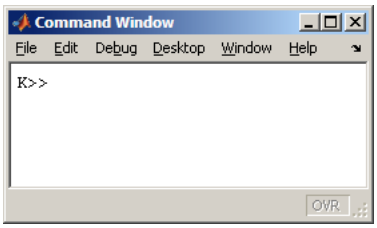
6. Let's use the *keyboard* method for this tutorial.

   Type keyboard right below pushbutton1_Callback, as shown in the figure above.

7. Now, save and run the GUI. Press the pushbutton.

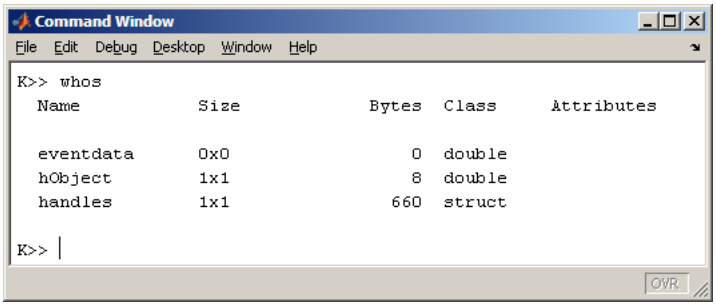


8. The command window should pop up and you should see the following at the command window. Notice the normal command line has been replaced with "K>>" . This just means that you're in *keyboard* mode.

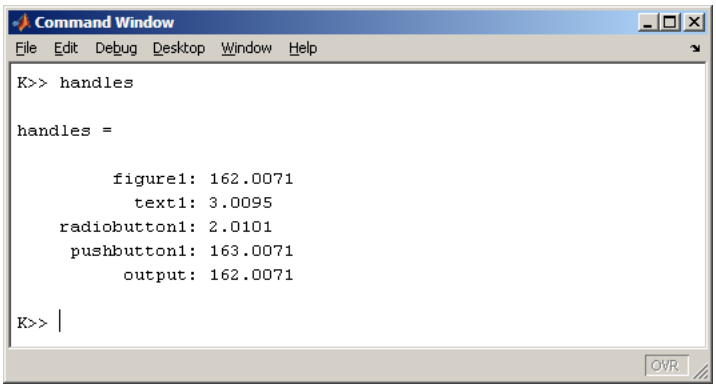


Handles, get, and set (cont)

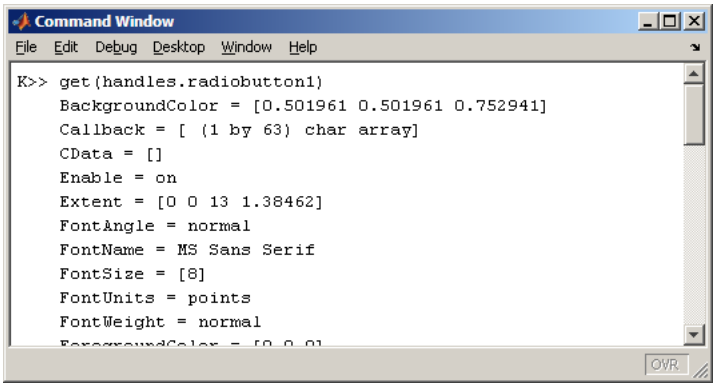
1. Now, let's take a look at the variables within the function's workspace. Type whos at the command prompt. This command tells you what variables are in the local workspace. Nothing too interesting.



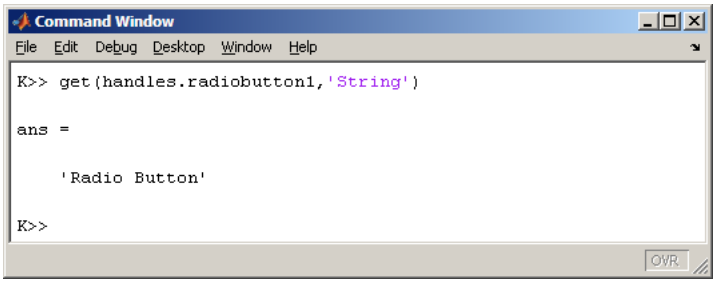
2. Now, let's take a look at the handles. Type handles at the command prompt. This command gives you more details on the *handles* structures. You'll notice that each object on the GUI figure is accounted for. (figure1 is the background image that your components are placed on)



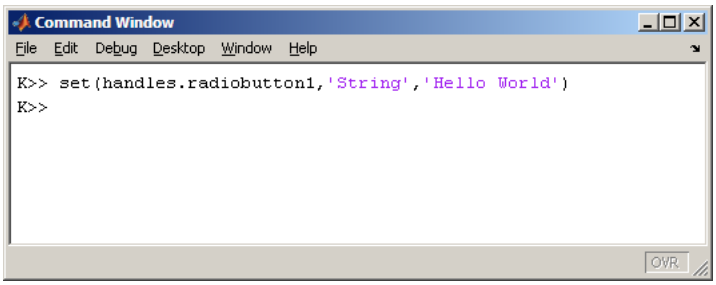
3. Lets say you wanted more details on the properties of *radiobutton1*. You can type get(handles.radiobutton1) at the command prompt to get a list of all the properties of this object. This command will display all the properties of that component, similar to what you would see in the Property Inspector when you double click on this component in the GUIDE figure.



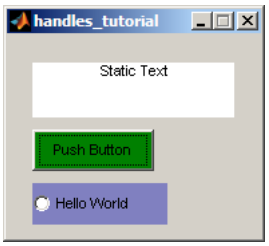
4. Let's say you only wanted details on the *String* property for *radiobutton1*; you can type get(handles.radiobutton1,'String') at the command prompt. Additionally, you can store this value into a variable for later use. The get command is probably used most often with *Edit Text* components to extract user inputs.



5. Lets say you wanted to change the *String* property on radiobutton1. You can do this by using set(handles.radiobutton1,'String', 'hello world') at the command prompt.

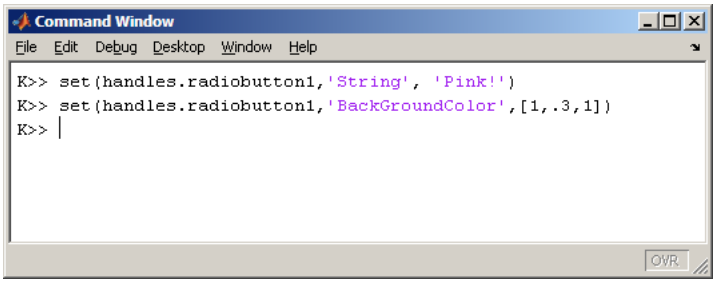


Notice that any changes you make using the set command are instantly reflected on the GUI program (not the GUIDE figure, but the actual GUI that is running).

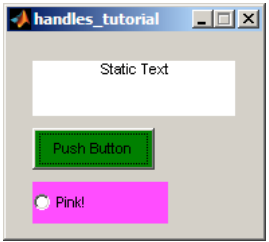


6. Try inserting these commands into the command line:

set(handles.radiobutton1,'String', 'The button is changed!')
set(handles.radiobutton1,'BackGroundColor',[1,.3,1])



The GUI that is running should now look like this:



7. After you're done playing around, type return at the command prompt to exit keyboard mode. Next, you should erase the keyboard command that you placed in the .m file and save it. Otherwise, the GUI will keep going into keyboard mode when you push that button.