



# Building a Matlab GUI

Todd Wittman  
July 2, 2008

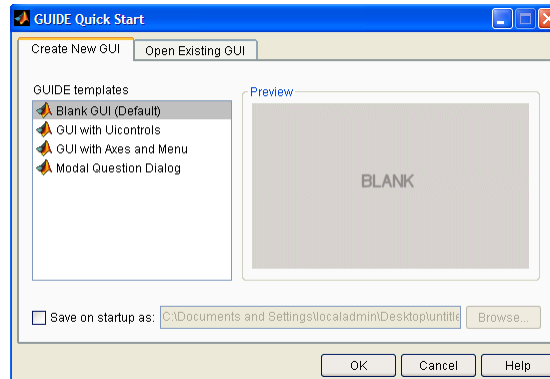


## GUIs

- A GUI (Graphical User Interface) is useful for presenting your final software.
- It also makes it easier to adjust parameters and visualize your programs.
- Your mentor may ask you to make a GUI. (My image processing students are required to present their final software in a GUI.)

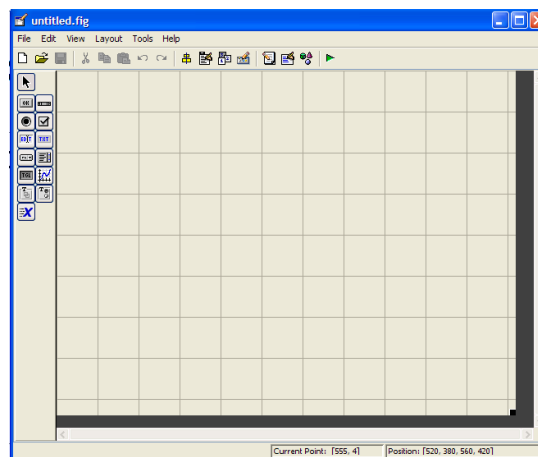
# GUIDE

- To get started, type "guide" in Matlab. Let's start with a blank GUI.



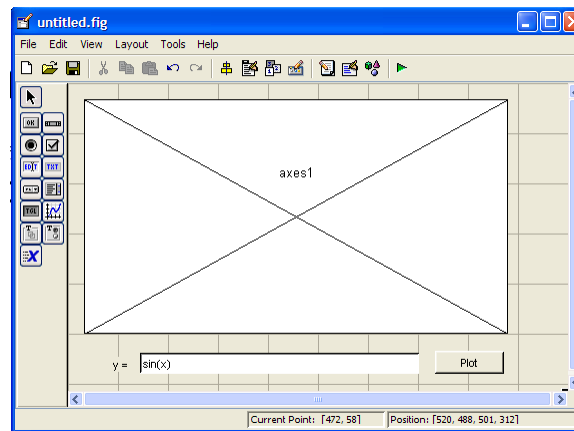
# GUIDE

- You get a blank form that you can place *controls* on.
- Before you jump in, you should have a plan of how you want to lay out your GUI.



## Example: Plotter

- Let's create a GUI that plots a given function.
- We first lay out the basic controls for our program, selected from the menu along the left side: axes, static text, edit box, and a button.

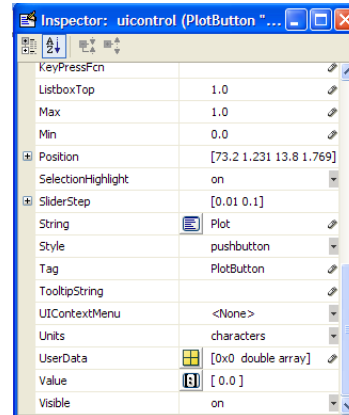


## Basic Controls

- axes: something to draw upon
- static text: text that is stuck on the screen, the user can't edit it
- edit box: a white box that the user can type into
- button: performs an action when user clicks on it

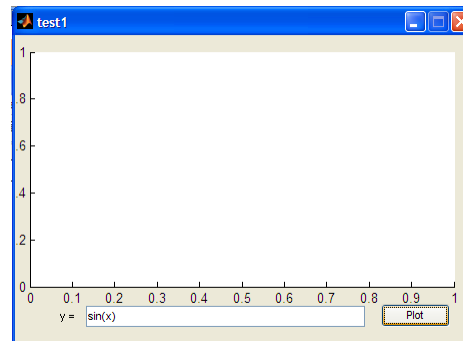
# The Property Inspector

- When you double-click on a control, it brings up a window listing all the properties of that control (font, position, size, etc.)
- **Tag** - the name of the control in the code. best to rename it to something identifiable ("PlotButton" vs "button1")
- **String** - the text that appears on the control
- **ForegroundColor** - color of the text
- **BackgroundColor** - color of the control



# Running

- If you press the green arrow at the top of the GUI editor, it will save your current version and run the program. The first time you run it, it will ask you to name the program.
- Our figure looks about right, but it doesn't do anything yet.
- We have to define a *callback* for the button so it will plot the function when we press it.



## Writing Callbacks

- When you run the program, it creates two files.
  - your\_gui.fig -- contains the layout of your controls
  - your\_gui.m -- contains code that defines a callback function for each of your controls
- We generally don't mess with the initialization code in the m-file.
- We will probably leave many of the control callbacks blank.
- In our example, we just need to locate the function for the button. This is why it is important to have a good Tag so we can keep our controls straight.
- You can also right-click on the control and select View Callback.

## Writing Callbacks

- Initially the button callback looks like this.

```
% --- Executes on button press in PlotButton.  
function PlotButton_Callback(hObject, eventdata, handles)  
% hObject    handle to PlotButton (see GCBO)  
% eventdata  reserved - to be defined in a future version of MATLAB  
% handles    structure with handles and user data (see GUIDATA)
```

- We can delete the comments and type in some code.
- Note every function has the parameter handles. This contains all the controls: handles.PlotButton, handles.edit1, handles.axes1, ...
- We can add variables to handles to make them available to all functions: handles.x = 42;

## Writing Callbacks

- We can look up any property of a control with the `get` function. Similarly, we can change any property with the `set` function.
- In this case, we want to get the String typed into the edit box and plot it.

```
function PlotButton_Callback(hObject, eventdata, handles)
x = -10 : 0.1 : 10;
s = get(handles.functionEdit, 'String');
y = eval(s); %eval just evaluates the given string
handles.axes1; %Subsequent commands draw on axes1.
plot(x, y);
```

## Running Your Program

- When you modify your m-file code, you don't have to re-run your GUI. The new code will run as a callback, so you can quickly test changes.
- To run your GUI, in the workspace you just type in the name you gave your GUI like it's a command:

```
>> my_gui
```

## Advanced Controls

- slider bar: the user can slide back and forth. the current position is given by Value, which is in between Min and Max. the callback is triggered whenever the slider is moved.
- check box: the user can toggle on or off
- radio button: like a check box, except only radio button in a group can be selected
- pop-up menu: user can select from a list of items. in the String property, you can type in multiple lines. The currently selected choice number is given by Value.
- panel: a rectangle to place controls upon. useful for organizing your GUI

## Working with the Workspace

- The workspace where we usually type in commands is called the base workspace.
- The `evalin` command allows you to read variables from the workspace.
- For example, to read the variable name typed into an edit box into handles.x:

```
s = get(variableEdit, 'String');
```

```
handles.x=evalin('base',s);
```

- You can write a value to the workspace with `assignin`.
- To create a variable x in the workspace:

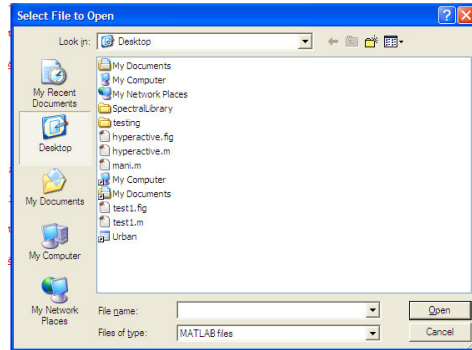
```
assignin ('base', 'x', handles.x);
```

## Loading & Saving Data

- The command `uigetfile` opens a browser so you can open a selected file.

`[filename, pathname] = uigetfile;`

- Similarly, `uiputfile` opens a browser to select a location to save a file.
- It's a little tricky if you want to specify file formats. Read the help on these functions for examples.



## More Information

- The Matlab help documentation on GUIs is quite good.
- You can Google many websites about building Matlab GUIs. A big list is at:  
[http://www.tech.plym.ac.uk/spmc/links/matlab/matlab\\_gui.html](http://www.tech.plym.ac.uk/spmc/links/matlab/matlab_gui.html)
- Mathworks has a 5-minute video about GUI building at:  
<http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=7598&ref=rssfeed&id=mostDownloadedFiles>