

Data Analysis in Geophysics

ESCI 7205

Bob Smalley

Room 103 in 3892 (long building), x-4929

Tu/Th - 13:00-14:30

CERI MAC (or STUDENT) LAB

Lab - 17, 10/24/13

Intro writing GUI's

MATLAB

MATLAB

What is a GUI?

Graphical User Interface

(Aside - what is “wysiwyg”?)

MatLab provides a tool called the
Graphical User Interface Development
Environment

(GUIDE)

A GUI used to create GUI's.

You can also be a masochist and write the code
from scratch.

A GUI should be consistent and easily understood.

(if you need the manual, there's a bug in the program or a flaw in the gui. Non-UNIX philosophy!)

Provide the user with the ability to use a program without having to worry about commands to run the actual program.

Possible components of a GUI -

Pushbuttons

Sliders

List boxes

Menus

Interactive Graphics

....etc

3 Essential Parts of a GUI –

1

Graphical Components

pushbuttons, edit boxes, sliders, labels, menus,
etc...

Static Components

Frames, text strings,...

Both created using the MATLAB function
uicontrol.

3 Essential Parts –

2

Figures – components are contained in figures.

3

Callbacks – The functions which perform the required action when a component is “pushed”.

GUIDE Properties

Allows the user to drag and drop components that he/she wants in the “layout” area of the GUI.

All “guide” GUI’s start with an opening function.

Callback is performed before user has access to GUI.

GUIDE stores GUIs in two files, which are generated the first time you save or run the GUI:

- .fig file - contains a complete description of the GUI figure layout and the components of the GUI. Changes to this file are made in the Layout Editor

- .m file - contains the code that controls the GUI.

You program callbacks in this file using the M-file Editor.

Creating a GUI

Typical stages of creating a GUI are:

1. Designing the GUI

2. Laying out the GUI
Using the Layout Editor

3. Programming the GUI
Writing callbacks in the M-file Editor

4. Saving and Running the GUI

Assessing the Value of Your GUI

Ask yourself two basic questions when designing your GUI.

- Do the users always know where they are?
- Do they always know where to go next?

Constantly answering these two questions will help you keep in perspective the goal of your GUI.

Callback function

The “meat” of the GUI process.

Opening function is first callback in every “guide” generated GUI.

Usually used to generate data used in GUI.

Callbacks define what will happen when a figure component is selected.

You must write the callback code!!!!

Summary

At command prompt type “guide”.

Lay out your GUI in the layout editor.

Define data in Opening Function.

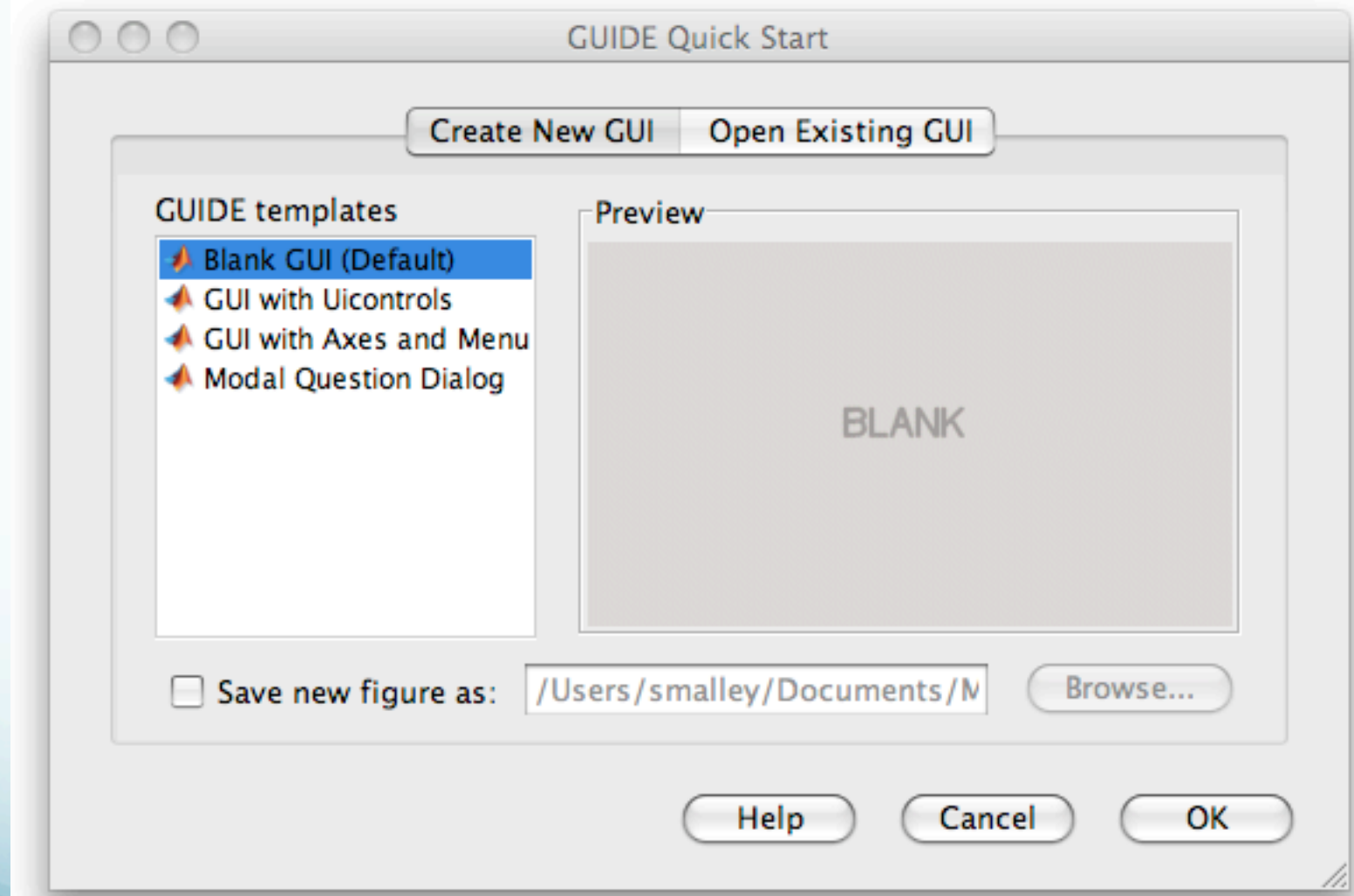
Edit/Align your components using

- Tools Menu
- Align
- View menu
- Property Inspector

Write the Callbacks

(This is the most difficult aspect when creating GUI's)

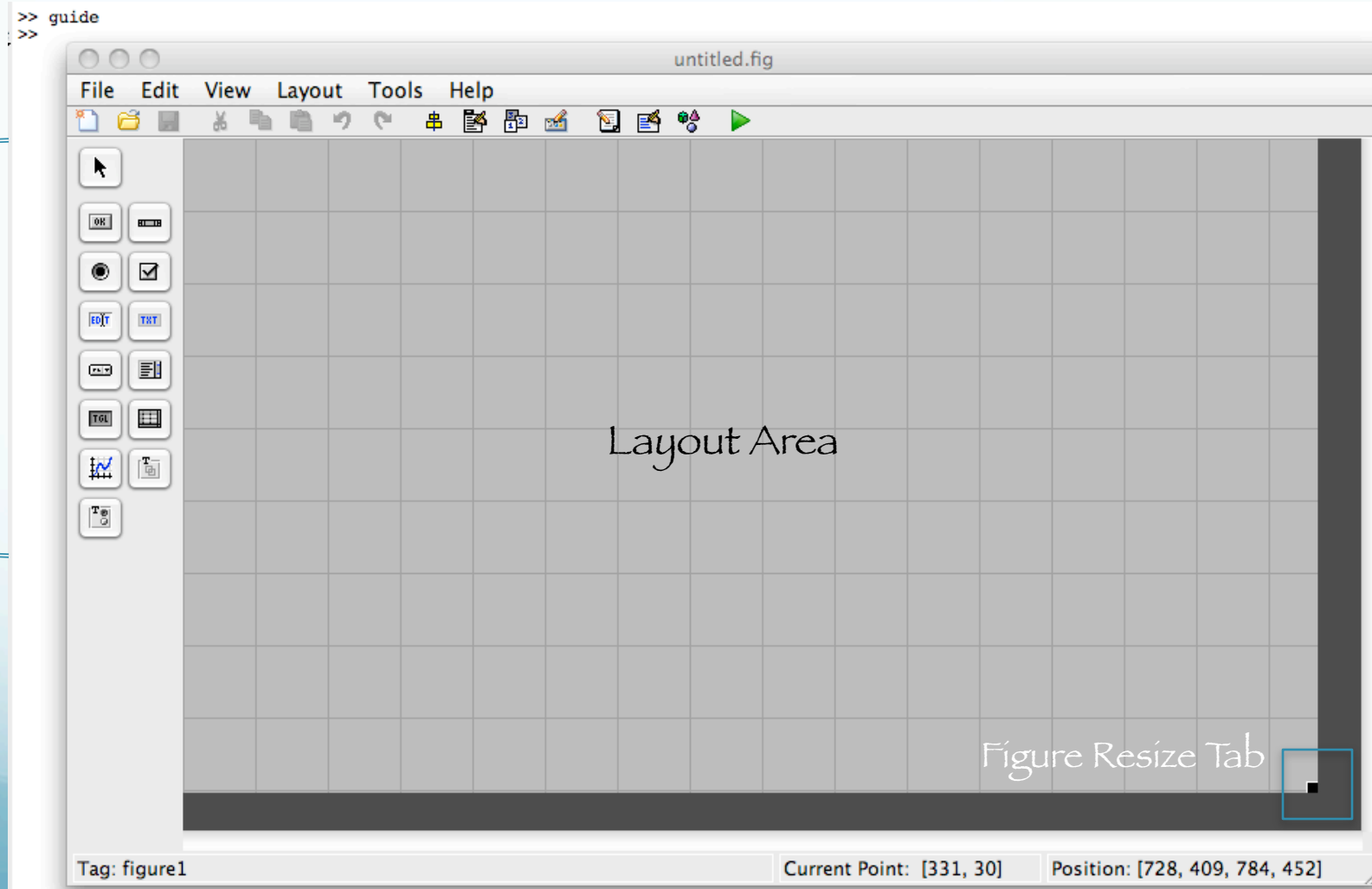
```
>> guide
>>
```



Components of GUIDE GUI interface

Alignment Tool
Menu Editor
Tab Order Editor
Toolbar Editor
M-File Editor
Property Inspector
Object Browser
Run Button

Component
Palette



Writing Callbacks (the hard part).

A callback is a sequence of commands (function) that are execute when a graphics object is activated.

Callbacks are stored in the GUI's m-file.

Callbacks are a property of a graphics object
(e.g. CreateFcn, ButtonDownFcn,
Callback, DeleteFcn).

(Also called an “event handler” in some programming languages.)

A callback is usually made of the following stages:

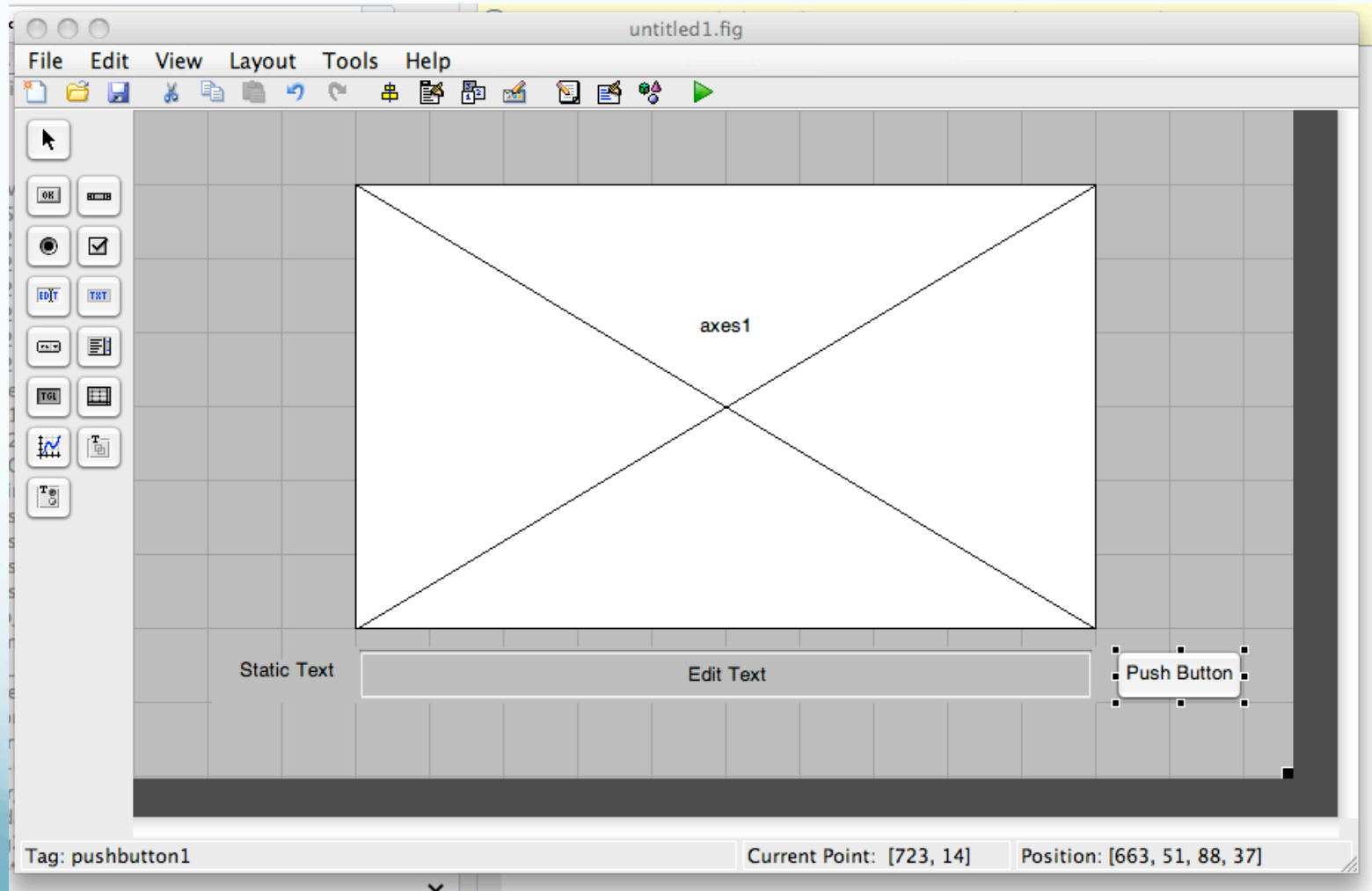
1. Get handle of object initiating the action
(the object provides event / information / values).
2. Get handles of objects being affected
(the object that whose properties are to be changed).
3. Getting necessary information / values.
4. Doing some calculations and processing.
5. Setting relevant object properties to effect action.

Let's create a GUI that plots a function that we can interactively specify.

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.

Define and place the axis, static text (will have the prompt for the function), edit text (to interactively enter the function), and a button to do the plot.



Basic Elements of our GUI-

axes: a place to draw.

static text: text that is stuck/fixed/static on the screen, the user can't edit it.

edit box: a white box that the user can type input 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

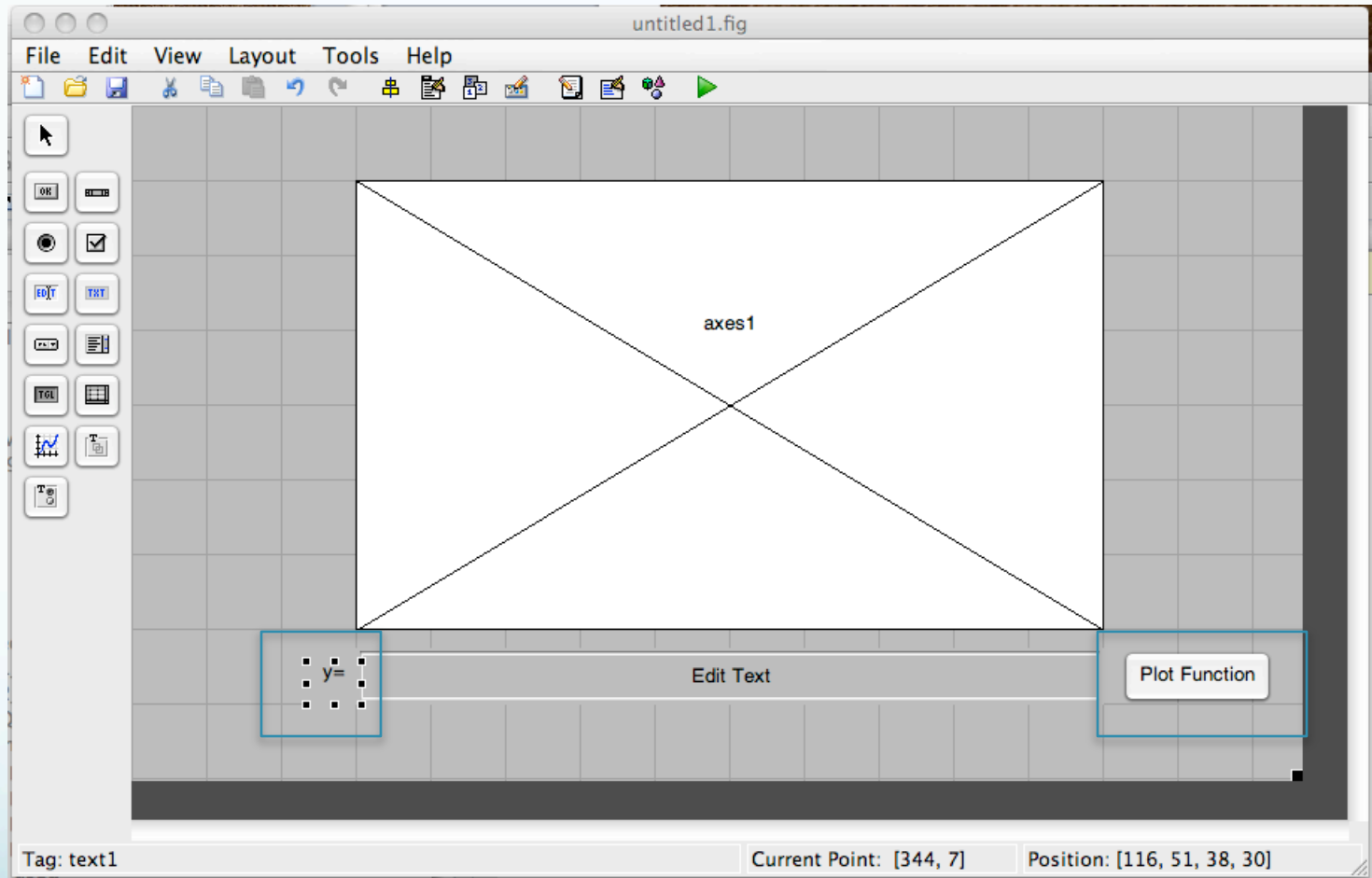
ForegroundColor - color of the text

BackgroundColor - color of the control

Inspector: uicontrol (PlotFunction "Plot Function")

BackgroundColor	
BeingDeleted	off
BusyAction	queue
ButtonDownFcn	
CData	[0x0 double array]
Callback	%automatic
Clipping	on
CreateFcn	
DeleteFcn	
Enable	on
Extent	[0 0 11.571 1.214]
FontAngle	normal
FontName	Helvetica
FontSize	10.0
FontUnits	points
FontWeight	normal
ForegroundColor	
HandleVisibility	on
HitTest	on
HorizontalAlignment	center
Interruptible	on
KeyPressFcn	
ListboxTop	1.0
Max	1.0
Min	0.0
Position	[94.571 3.571 12.571 2.643]
SelectionHighlight	on
SliderStep	[0.01 0.1]
String	Plot Function
Style	pushbutton
Tag	PlotFunction
TooltipString	
UIContextMenu	<None>
Units	characters
UserData	[0x0 double array]
Value	[0.0]
Visible	on

Enter text string for
pushbutton
Enter tag for
pushbutton



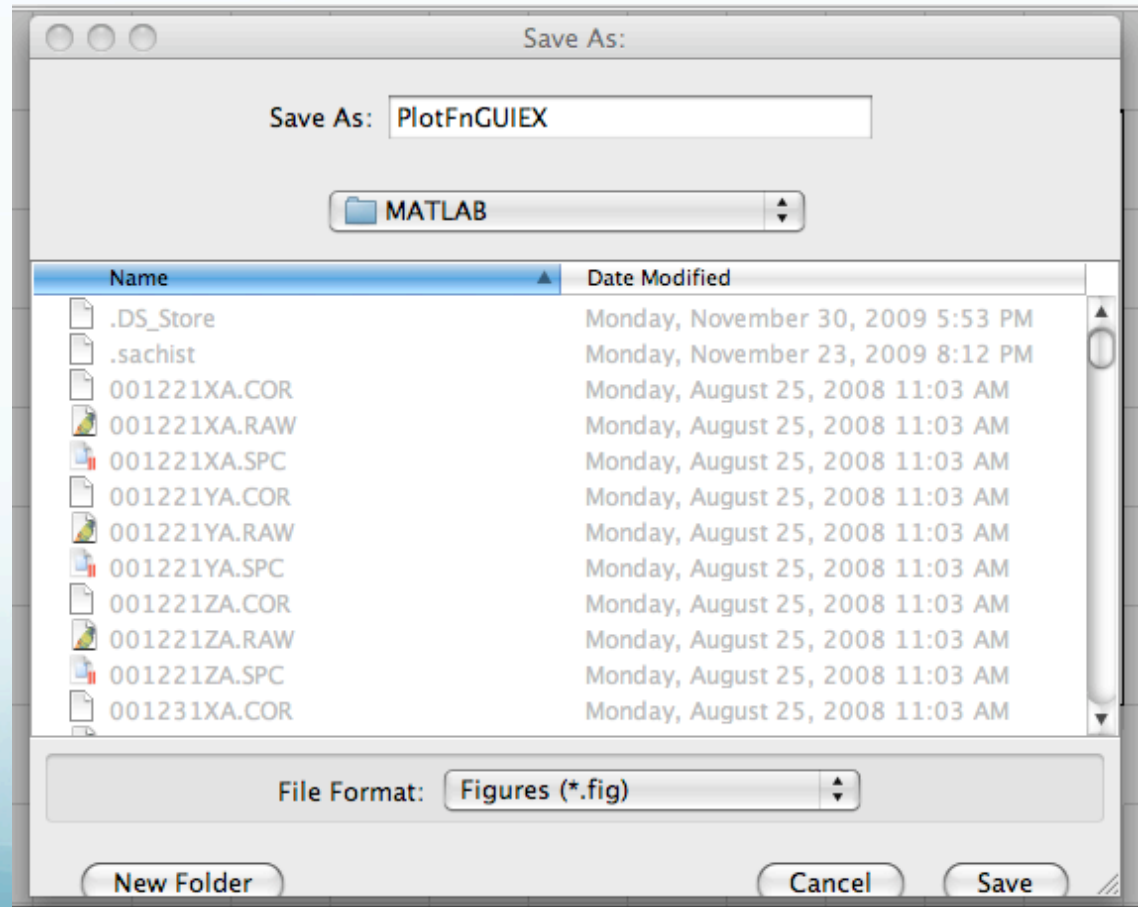
Running

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.



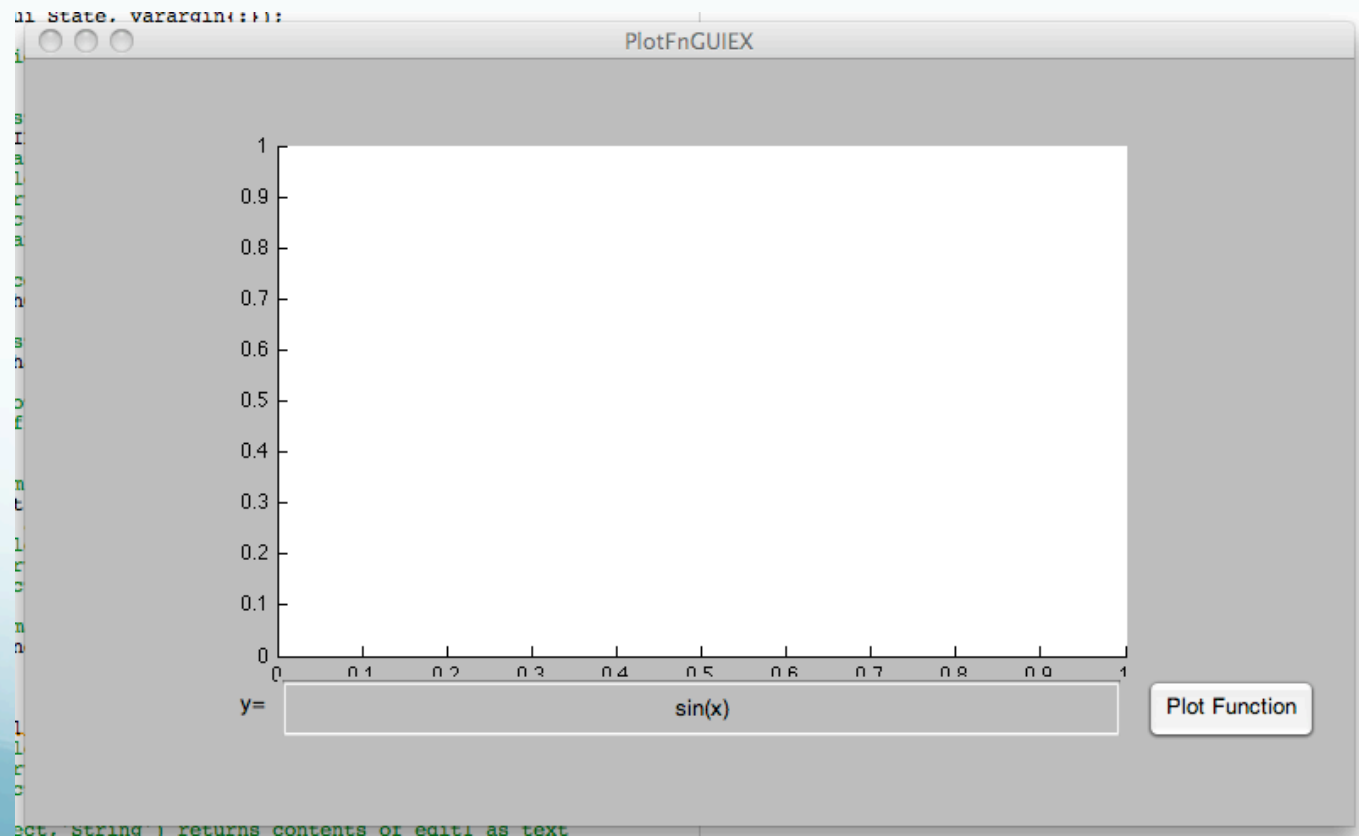
Pile of windows - GUIDE design window, m file with code for GUI, window with running GUI.

The image displays three overlapping MATLAB windows illustrating the GUI development process:

- GUIDE Design Window (Left):** Shows the graphical user interface design environment. It features a toolbar with various widget icons (buttons, text boxes, plots) and a central workspace where a simple plot window is being designed. The plot window has a white background and a black border.
- Editor Window (Top Right):** Shows the MATLAB source code for the GUI, titled "PlotFnGUIEX.m". The code includes:
 - Initialization of `gui_State.gui_Callback` to `str2func(varargin{1})`.
 - A `if nargin` block that sets `gui_mainfcn` to `gui_mainfcn(gui_State, varargin{:})`.
 - Comments for initialization and execution.
 - A `function PlotFnGUI` block with comments: "Executes just this function handles", "Choose default handles.output = handles.output", "Update handles guidata(hObject, handles)", and "UIWAIT makes PlotFnGUI wait for the GUI to finish".
 - A `function varargout = edit1_Callback(hObject, eventdata, handles)` block with comments: "Outputs from edit1_Callback", "Get default command string", and "Hints: get(hObject, 'String') returns contents of edit1 as text; str2double(get(hObject, 'String')) returns contents of edit1 as a double".
 - A `function edit1_CreateFcn(hObject, eventdata, handles)` block with comments: "Executes during object creation, after setting all properties", "handles to edit1 (see GCBO)", and "Hint: edit controls usually have a white background on Windows. See ISPC and COMPUTER." It includes a `if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))` block to set the background color to white.
 - A `function PlotFunction_Callback(hObject, eventdata, handles)` block with comments: "Executes on button press in PlotFunction", "handles to PlotFunction (see GCBO)", and "handles structure with handles and user data (see GUIDATA)".
- Running GUI Window (Bottom Right):** Shows the final running GUI. It features a plot window with a white background and a black border. The plot area is empty. Below the plot, there is an "Edit Text" input field and a "Plot Function" button.

Buttons “work” (respond when click in them), can enter text.
But nothing happens.

Have to write callback routine to specify what happens.



Writing Callbacks

As noted, 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.

Writing Callbacks

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 PlotFunction.  
function PlotFunction_Callback(hObject, eventdata, handles)  
% hObject      handle to PlotFunction (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 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.

This is where things get complicated.

Writing Callbacks

We need two callbacks.

1) We want to get the `String` typed into the edit box

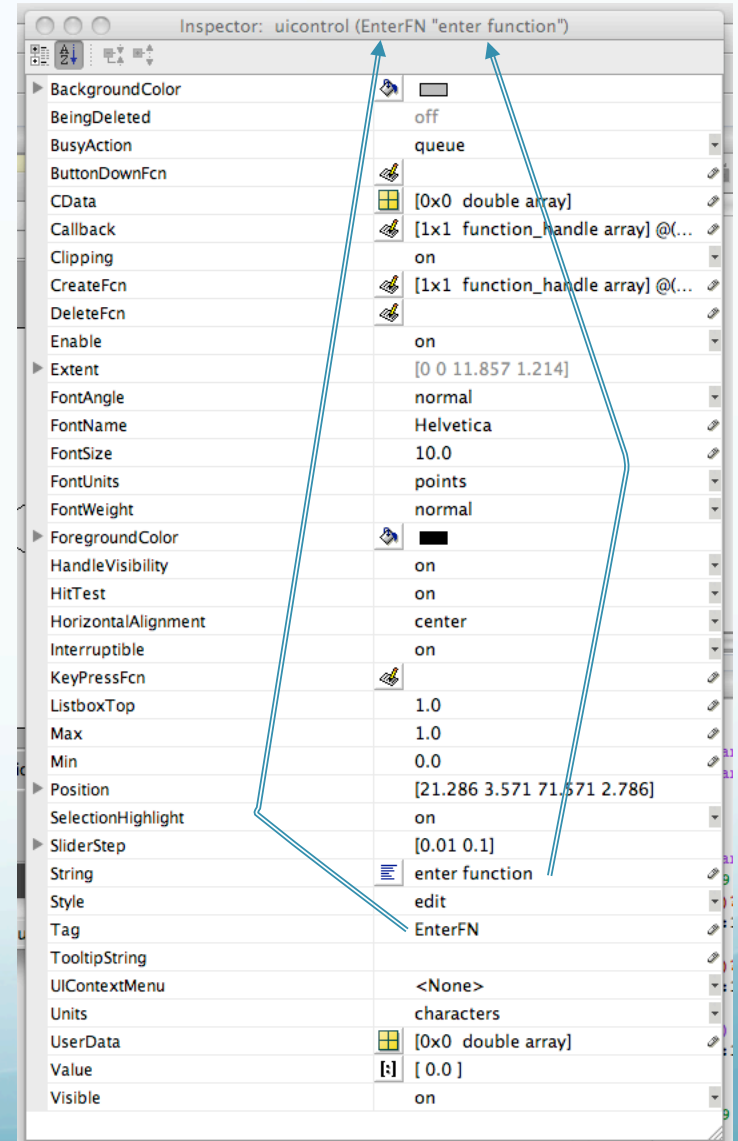
2) and plot it.

```
function EnterFN_Callback(hObject, eventdata, handles)
```

• • •

```
function EnterFN_CreateFcn(hObject, eventdata, handles)
```

Look at properties inspector and m file to see how things match up.



1) We want to get the string typed into the edit box

Blue produced by guide, have to add the black (one line). Variable handles.EnterFn created here.

```
function EnterFn_Callback(hObject, eventdata, handles)
% hObject      handle to EnterFn (see GCBO)
% eventdata    reserved - to be defined in a future version of
MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of EnterFn as
text
% str2double(get(hObject, 'String')) returns contents of
EnterFn as a double
handles.EnterFn=get(hObject, 'String');
```

2) and plot it.

Blue produced by guide, have to add the stuff in black (a couple of lines). Variable `handles.EnterFn` created by us, while `handles.axes1` created by guide.

```
% --- Executes on button press in PlotFunction.  
function PlotFunction_Callback(hObject, eventdata, handles)  
% hObject      handle to PlotFunction (see GCBO)  
% eventdata    reserved - to be defined in a future version of  
MATLAB  
% handles      structure with handles and user data (see GUIDATA)  
x=-10:.01:10  
s = get(handles.EnterFN, 'String');  
y = eval(s); %eval just evaluates the given string  
handles.axes1; %Subsequent commands draw on axes1.  
plot(x, y);  
grid;
```

Final result.

```
>> PlotFnGUIEX  
>>
```

