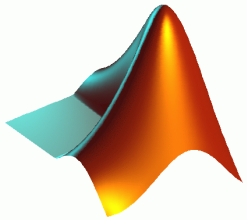
CS@CU

COMS W3101-2

Programming Languages: MATLAB



Lecture 6

Spring 2010

Instructor: Michele Merler

http://www1.cs.columbia.edu/~mmerler/comsw3101-2.html

Graphical User Interface (GUI)

- Basic GUI: a menu

- `menu()`
  - If graphics are enabled, displays a menu with options

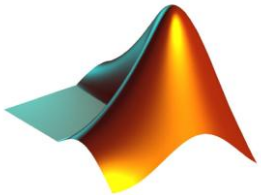
```
k = menu('Choose a Color',...
         'Red', 'Green', ...
         'Blue' ,'Cyan');
```

Graphical User Interface (GUI)

- Basic GUI: a menu

- `menu()`

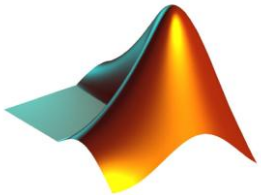
  - `colors = {'Red','Green','Blue','Cyan'};`
  - `k = menu('Choose a color',colors);`
  - `fprintf('you picked %s\n',colors{k});`



GUI

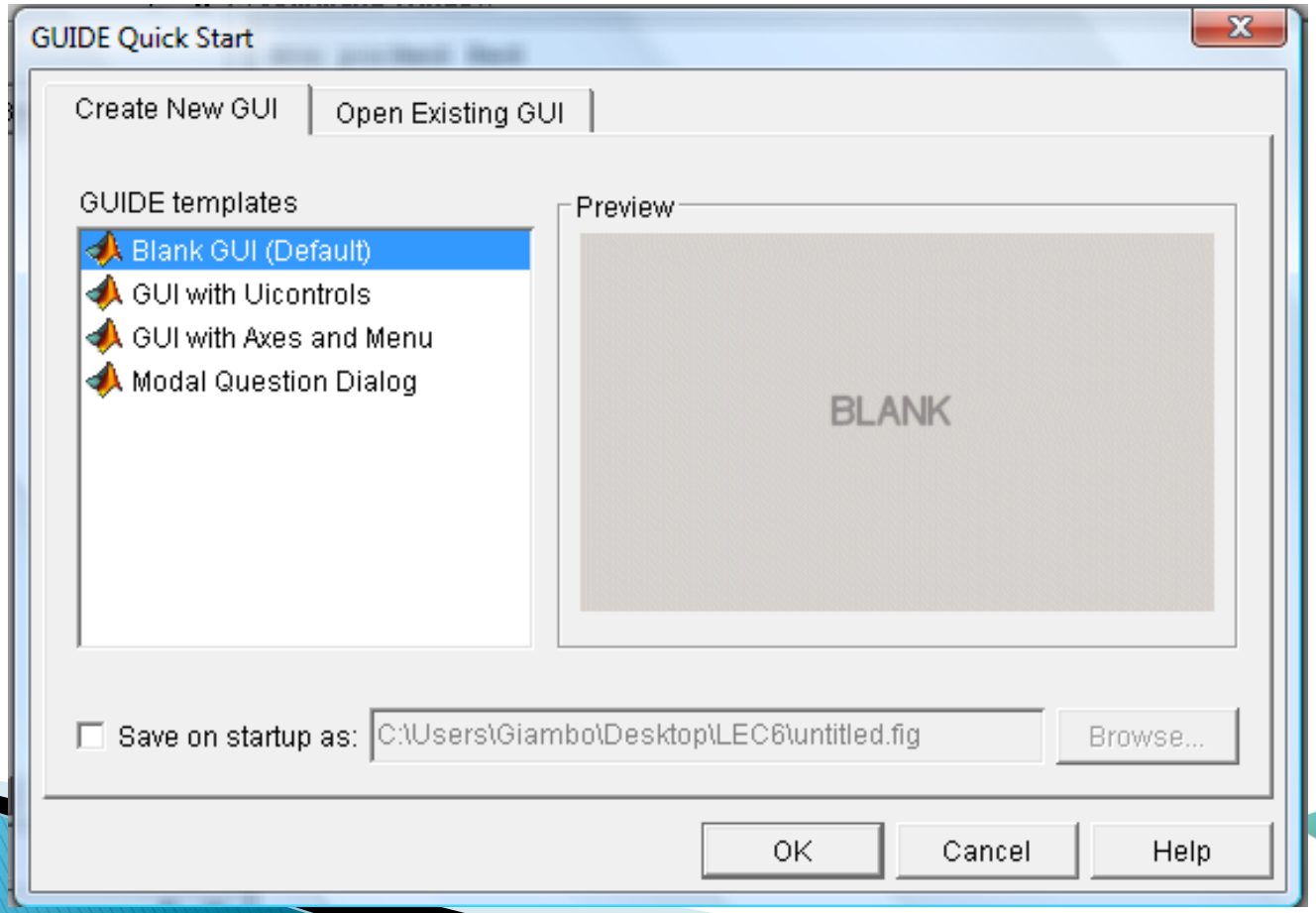
- There are 4 steps in creating a GUI:

1. Designing the GUI
2. Laying out the GUI (with the layout editor)
3. Programming the GUI (write the callbacks in the .m file)
4. Saving and Running the GUI



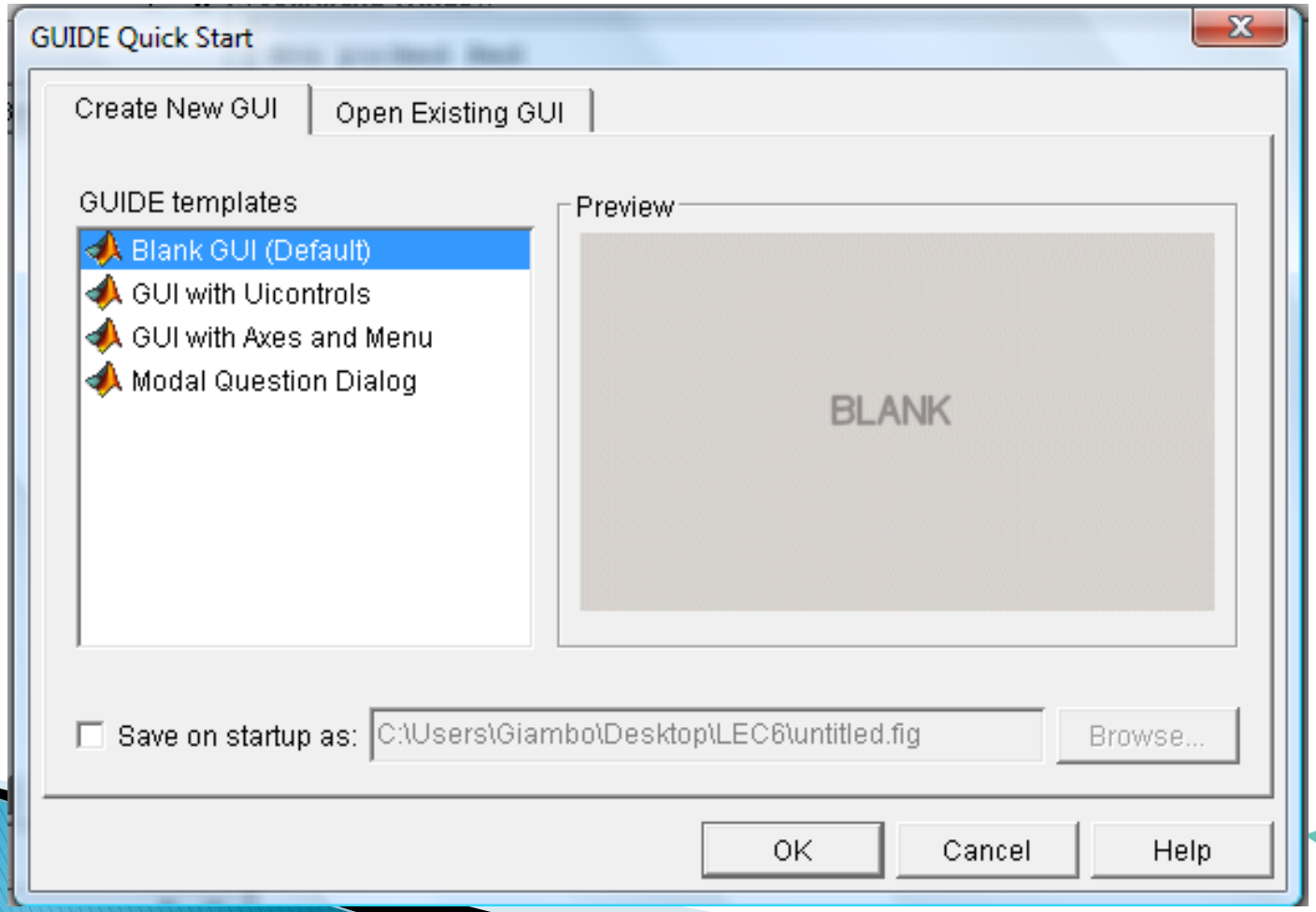
GUI – Laying out the GUI

- To start the MATLAB GUI Layout Editor creation environment, type ‘guide’ in the command window



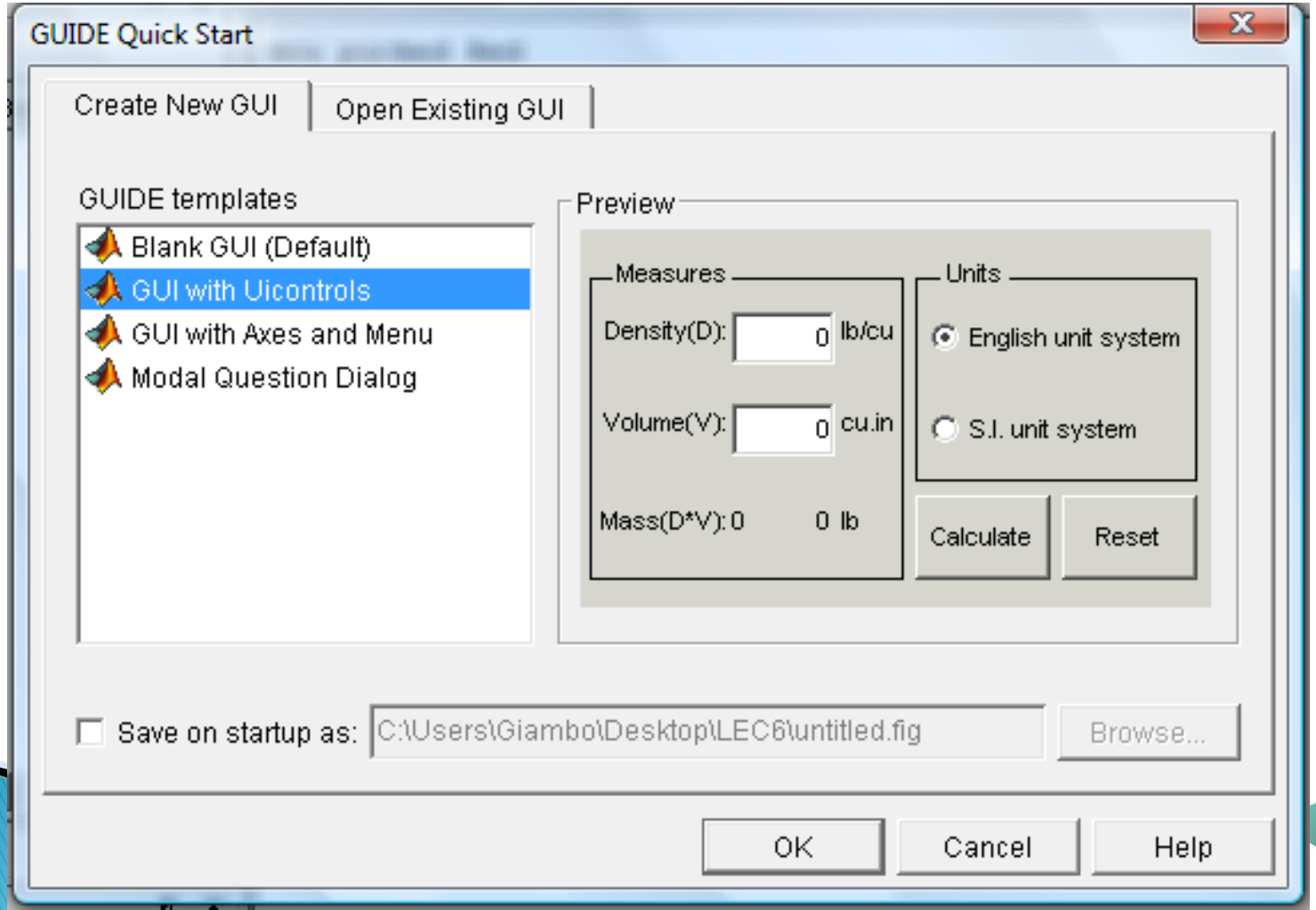
GUI Types

- Blank GUI, we can add anything we want



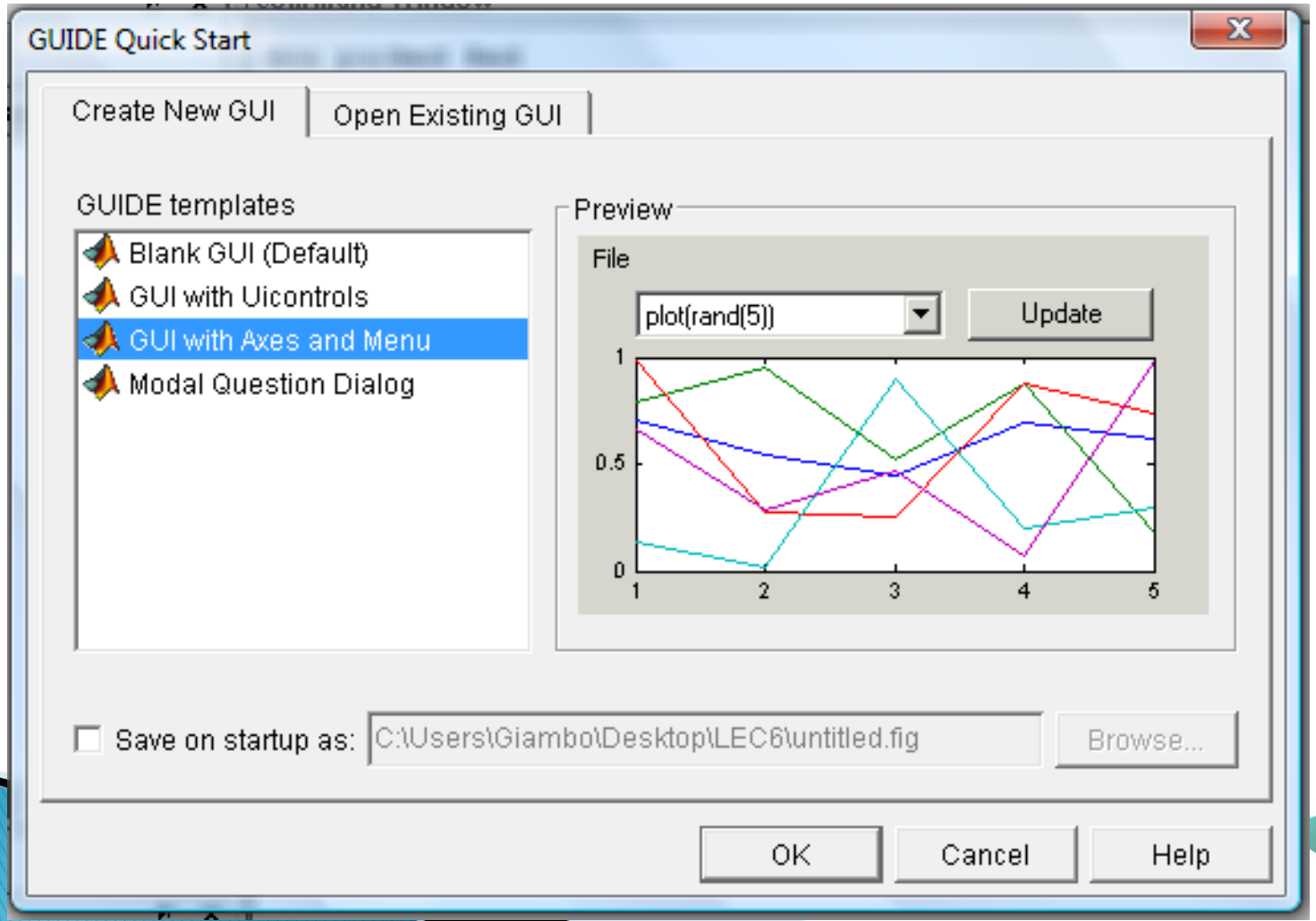
GUI Types

- GUI with UIcontrols



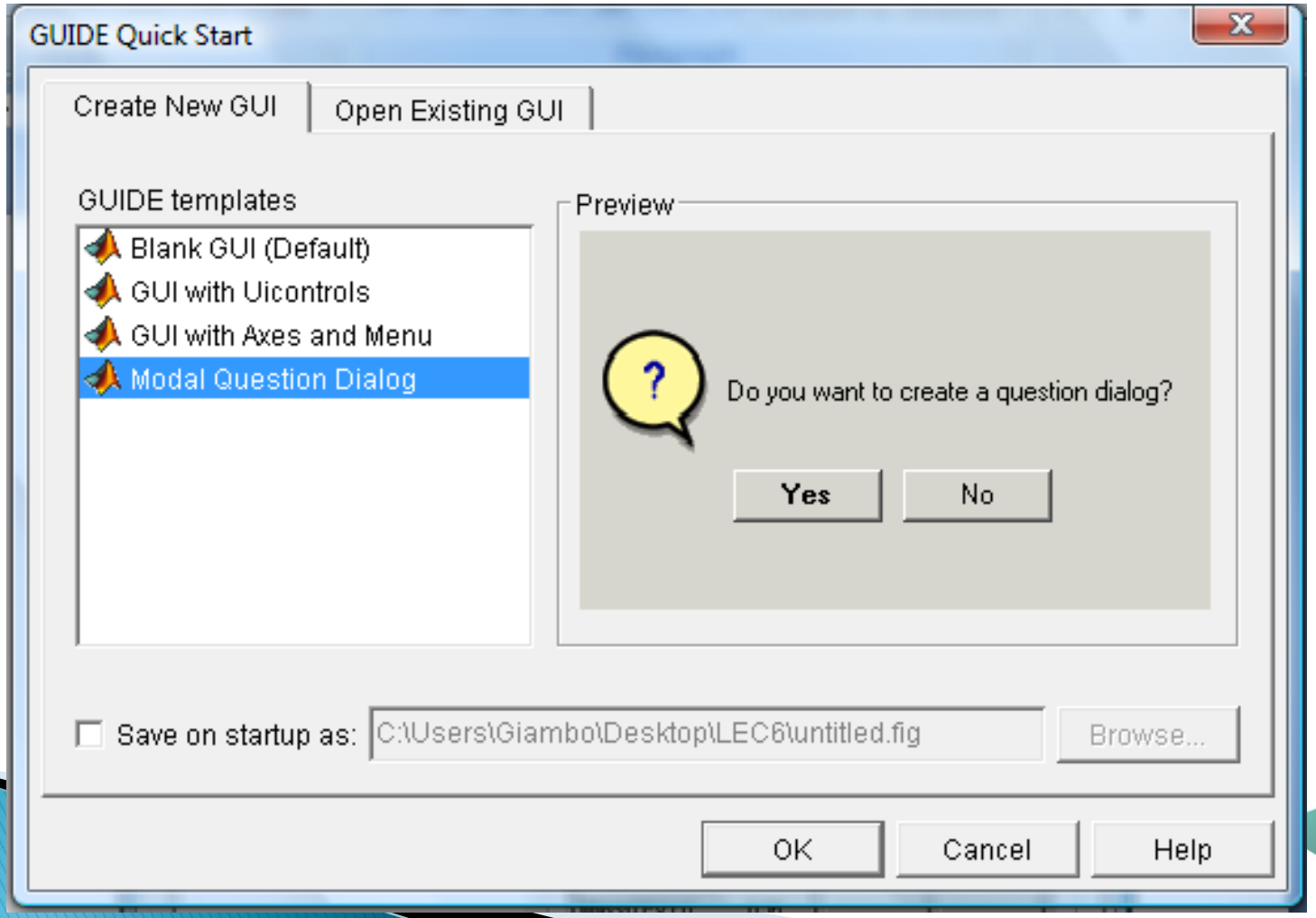
GUI Types

- GUI with Axes and Menu

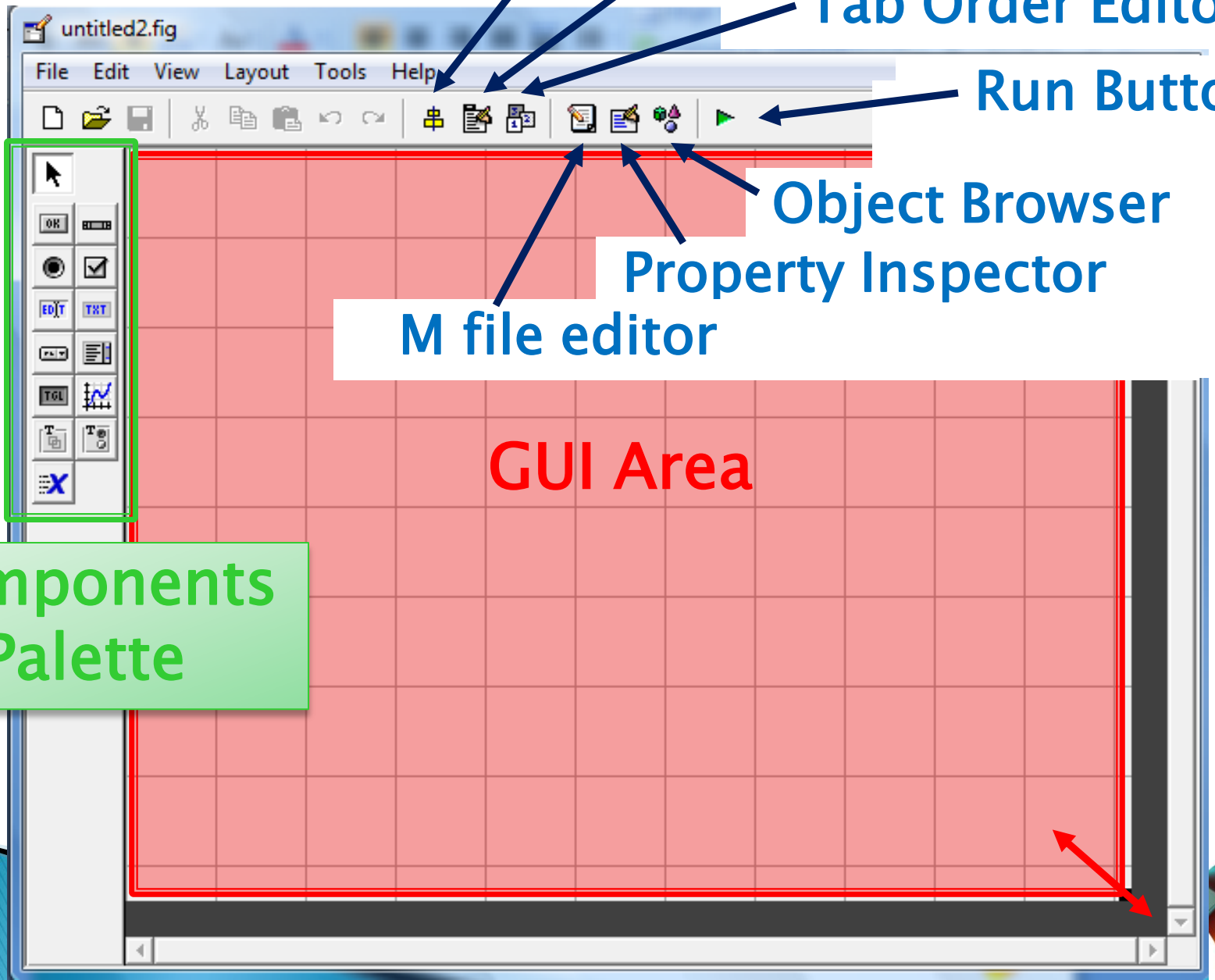


GUI Types

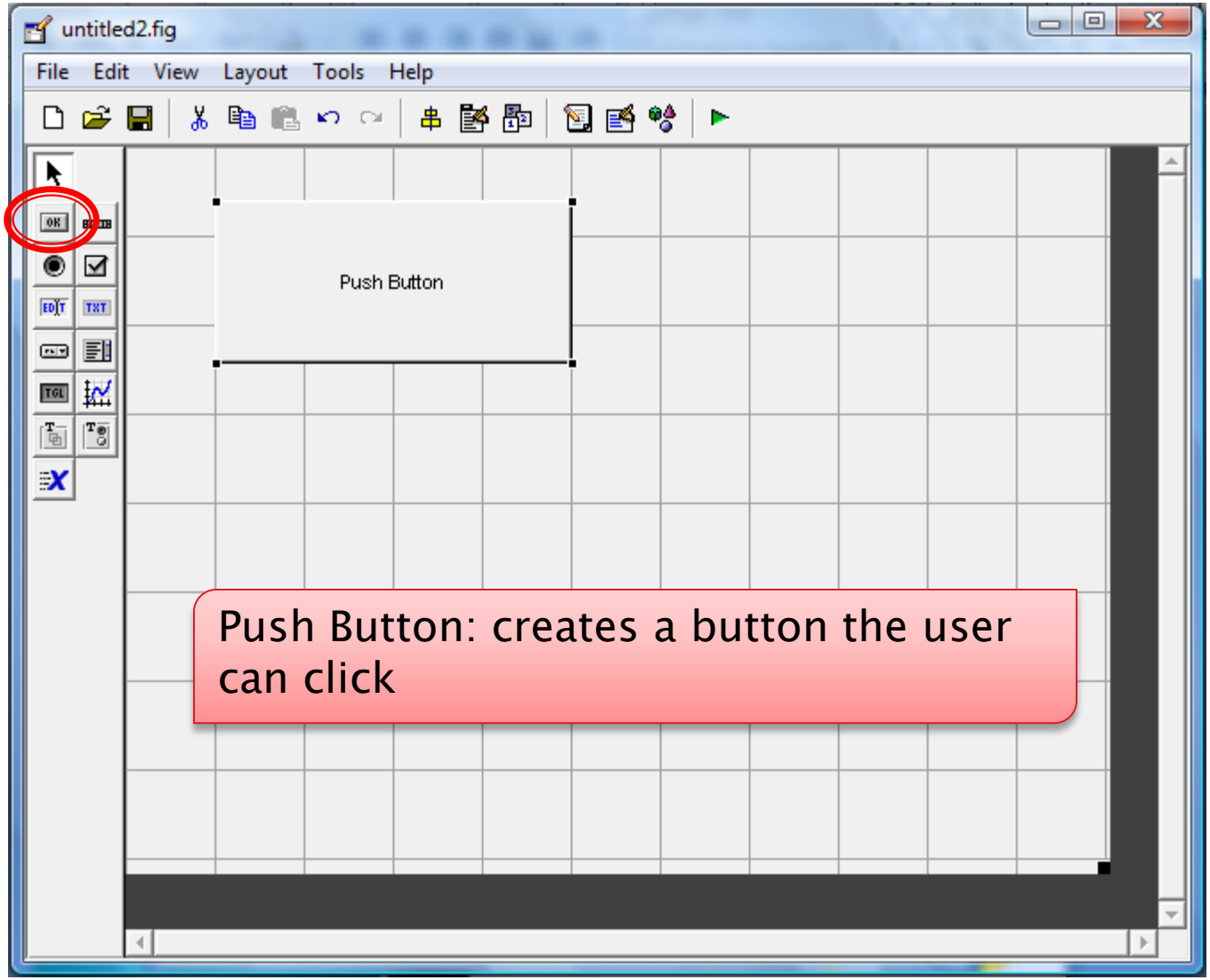
- Modal Question Dialog



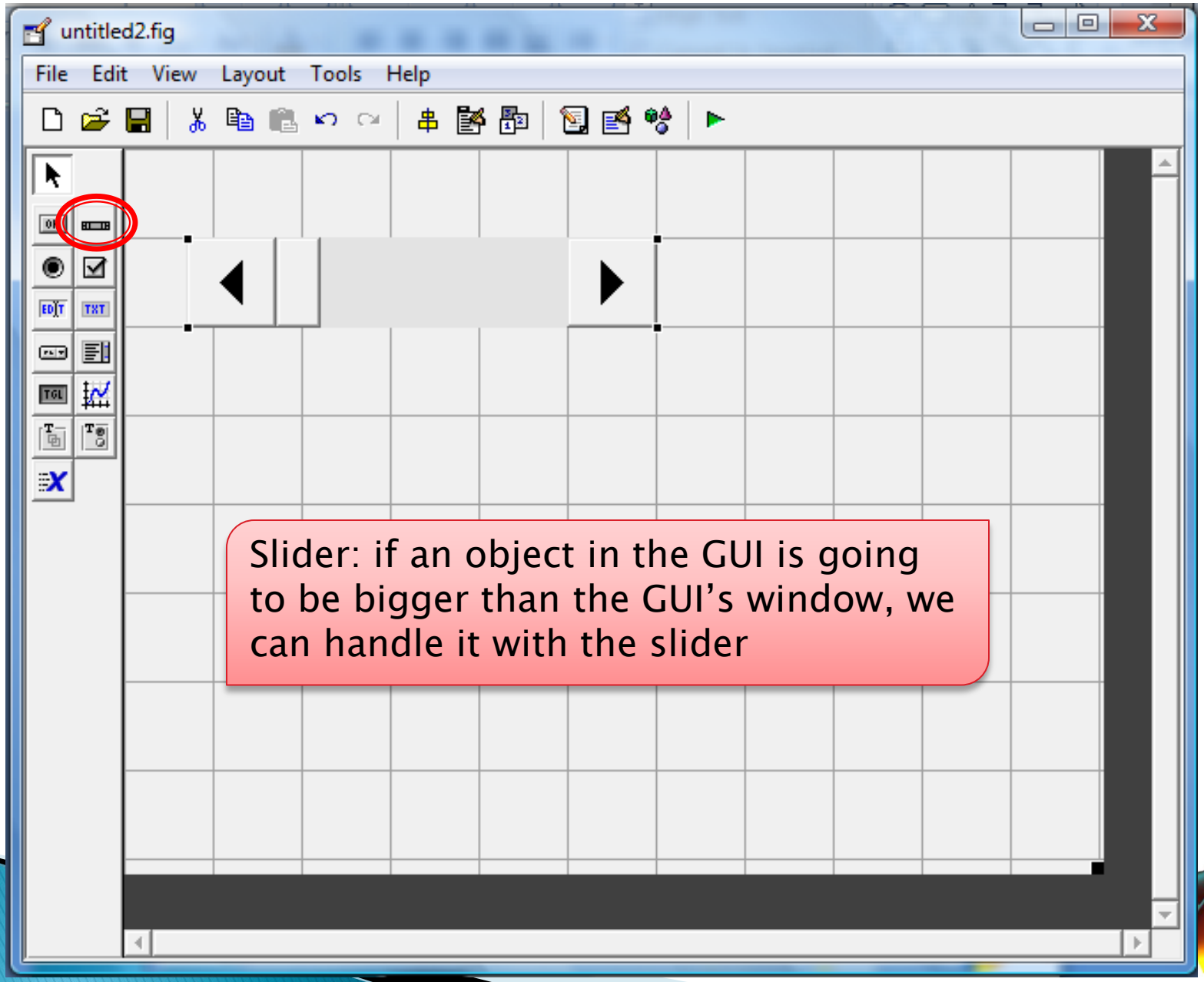
Layout Editor



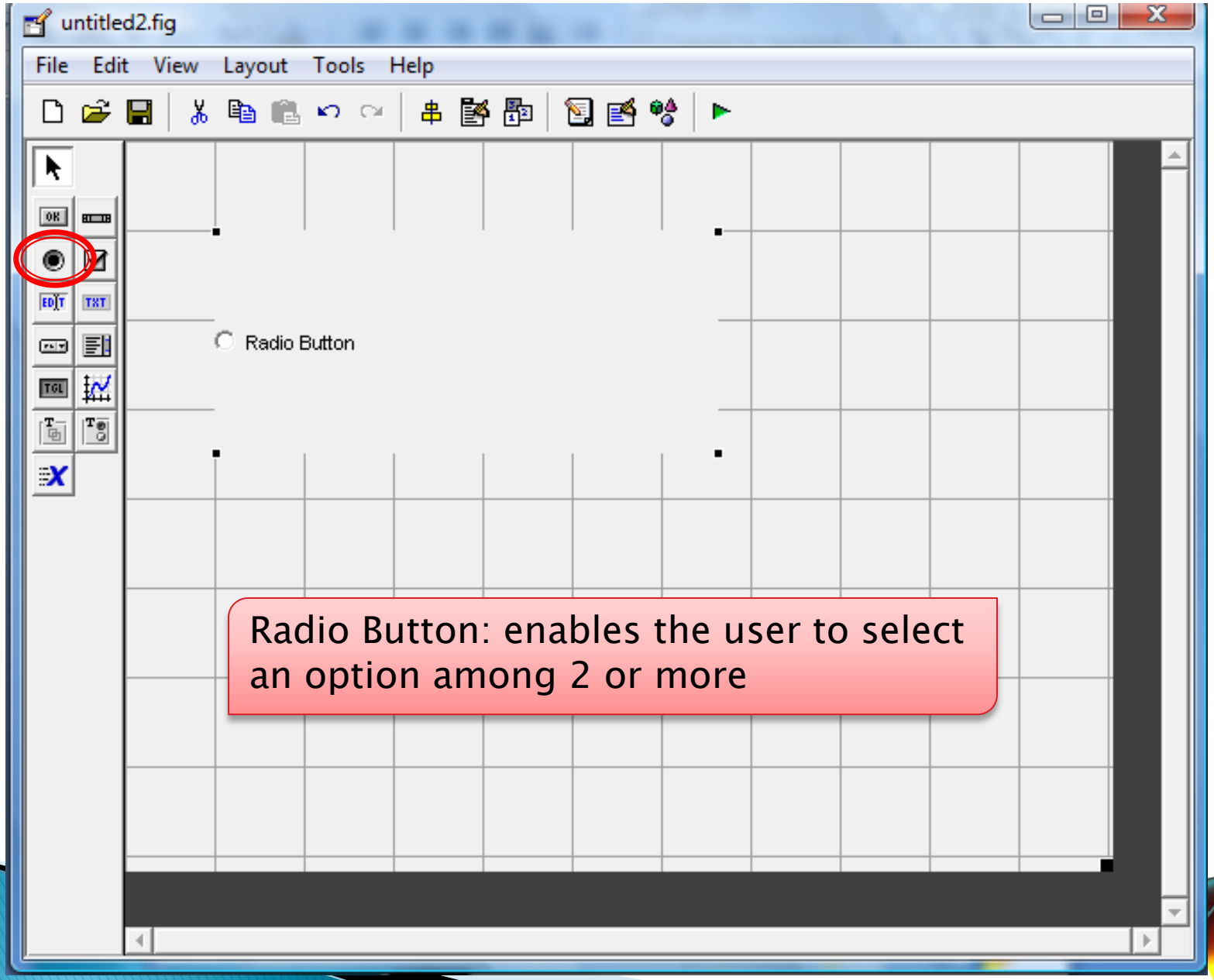
Components


untitled2.fig
File Edit View Layout Tools Help
Push Button
Push Button: creates a button the user can click

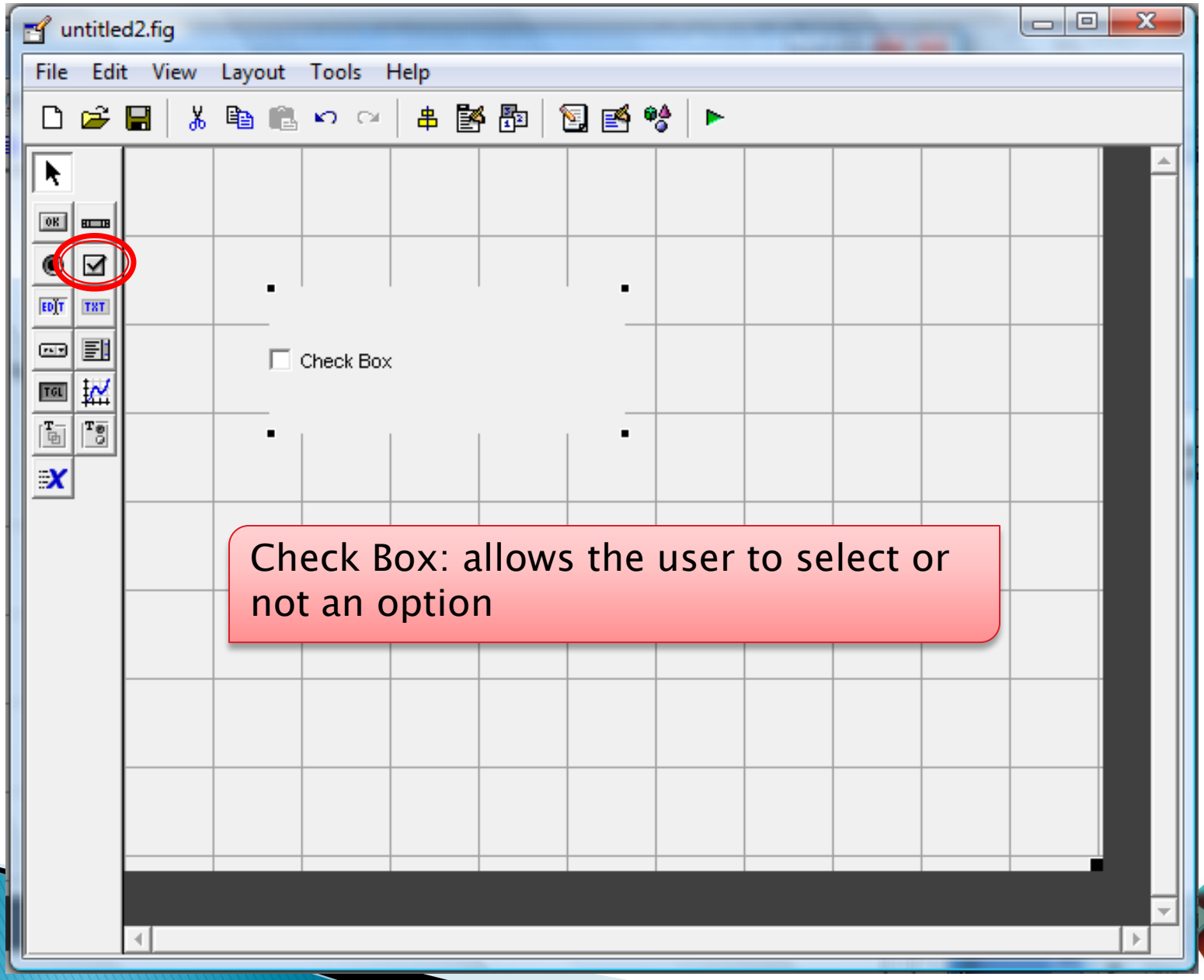
Components


untitled2.fig
File Edit View Layout Tools Help
Slider: if an object in the GUI is going to be bigger than the GUI's window, we can handle it with the slider

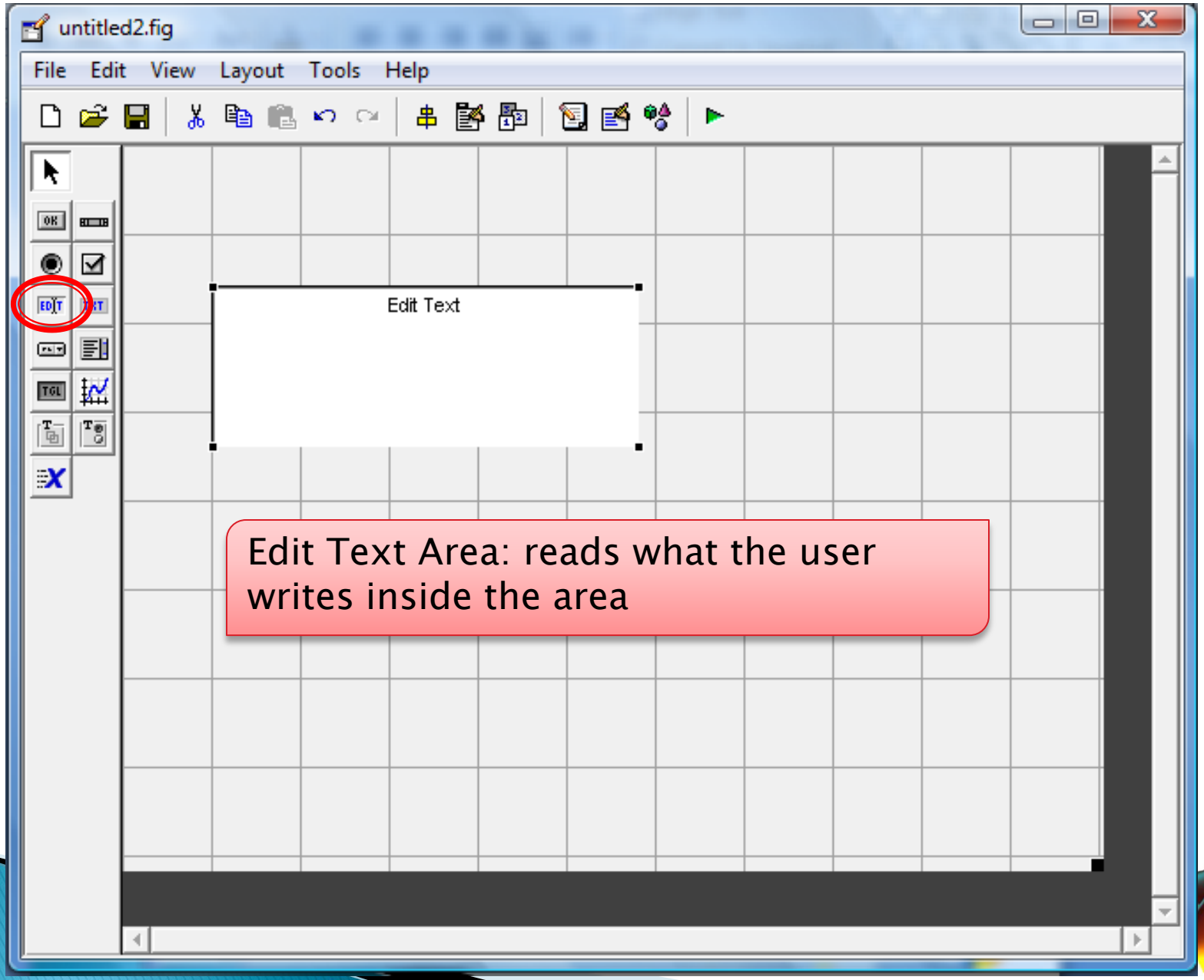
Components


untitled2.fig
File Edit View Layout Tools Help
Radio Button
Radio Button: enables the user to select an option among 2 or more

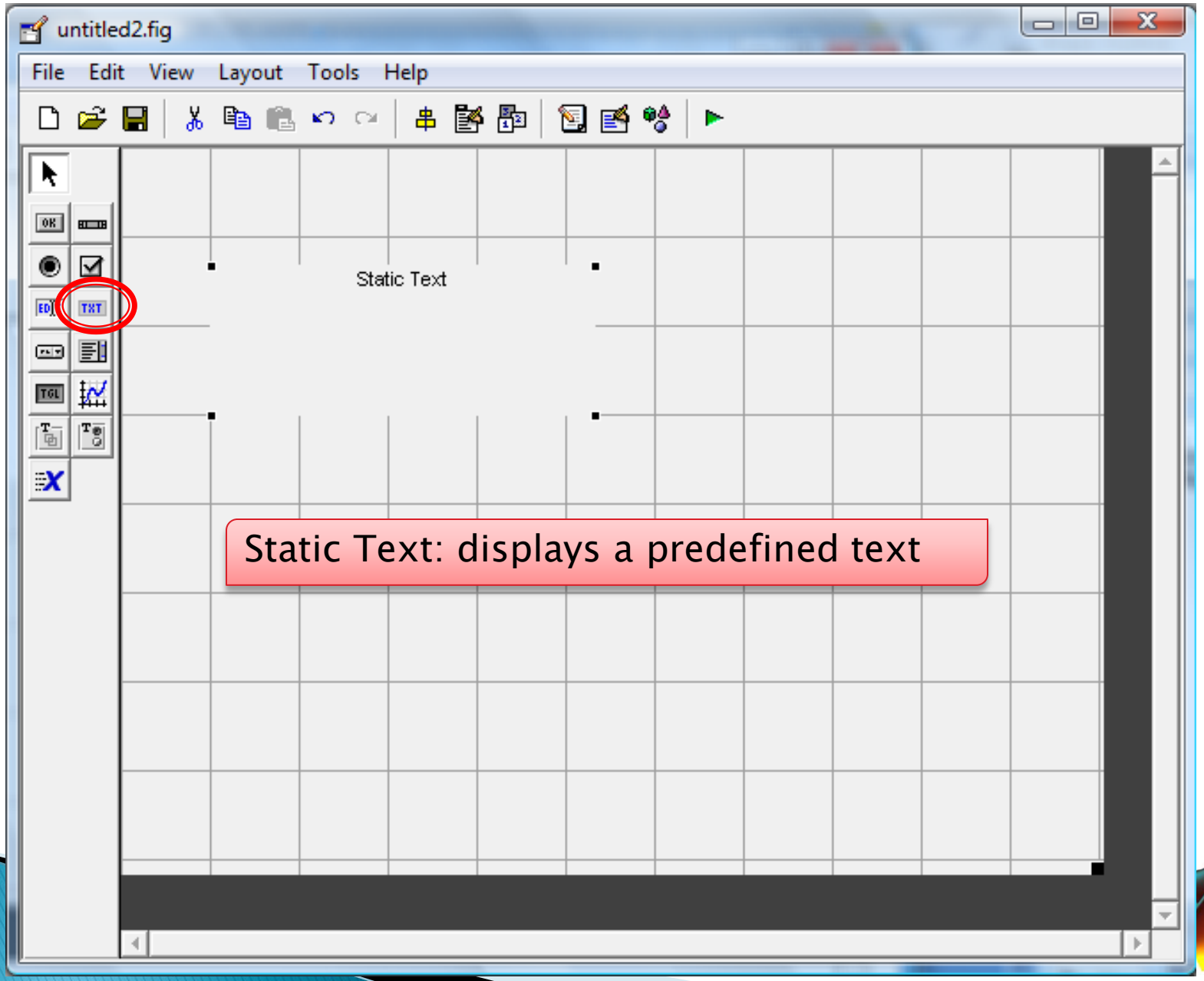
Components


untitled2.fig
File Edit View Layout Tools Help
Check Box
Check Box: allows the user to select or not an option

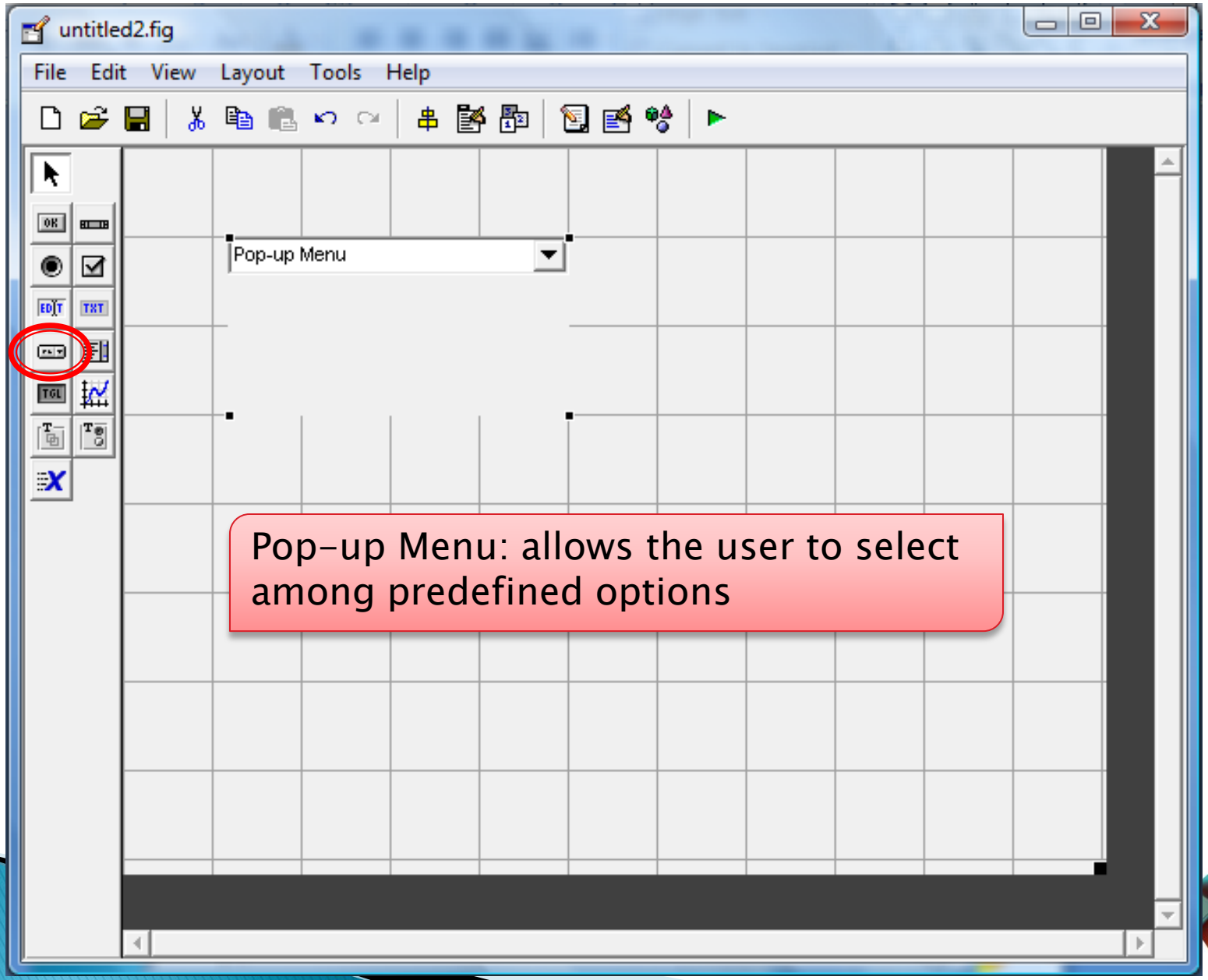
Components


untitled2.fig
File Edit View Layout Tools Help
Edit Text
Edit Text Area: reads what the user writes inside the area

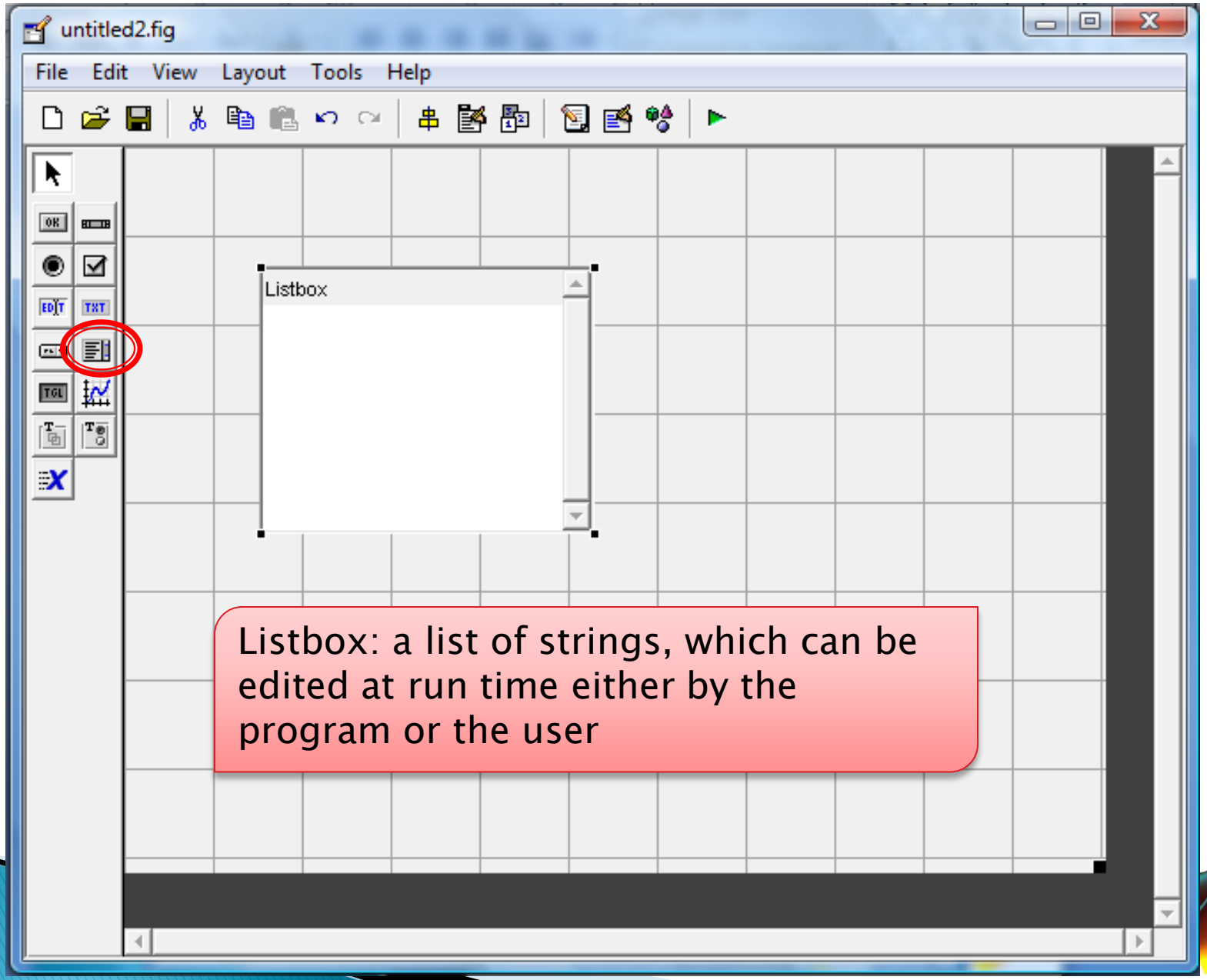
Components


untitled2.fig
File Edit View Layout Tools Help
Static Text
Static Text: displays a predefined text

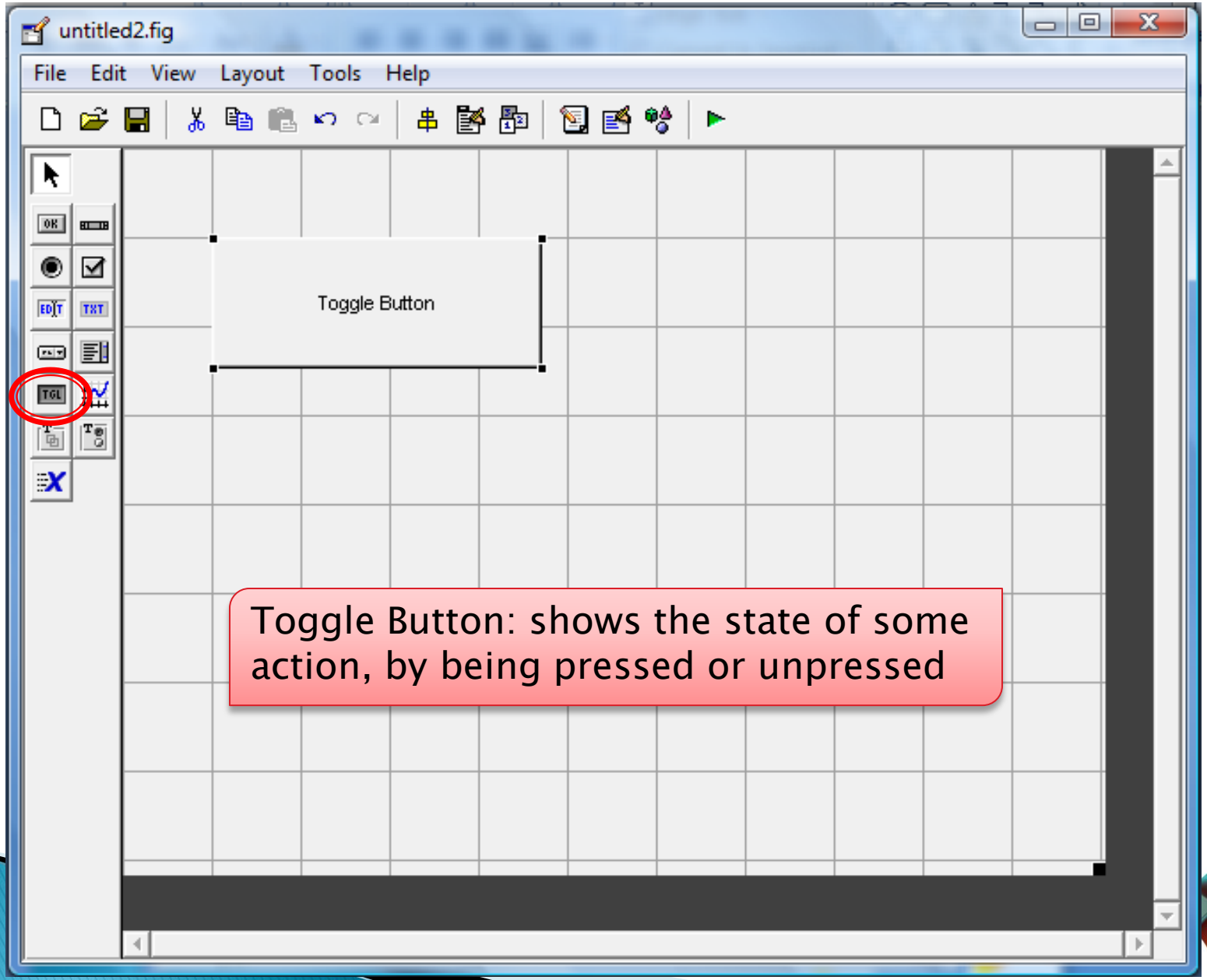
Components


untitled2.fig
File Edit View Layout Tools Help
Pop-up Menu
Pop-up Menu: allows the user to select among predefined options

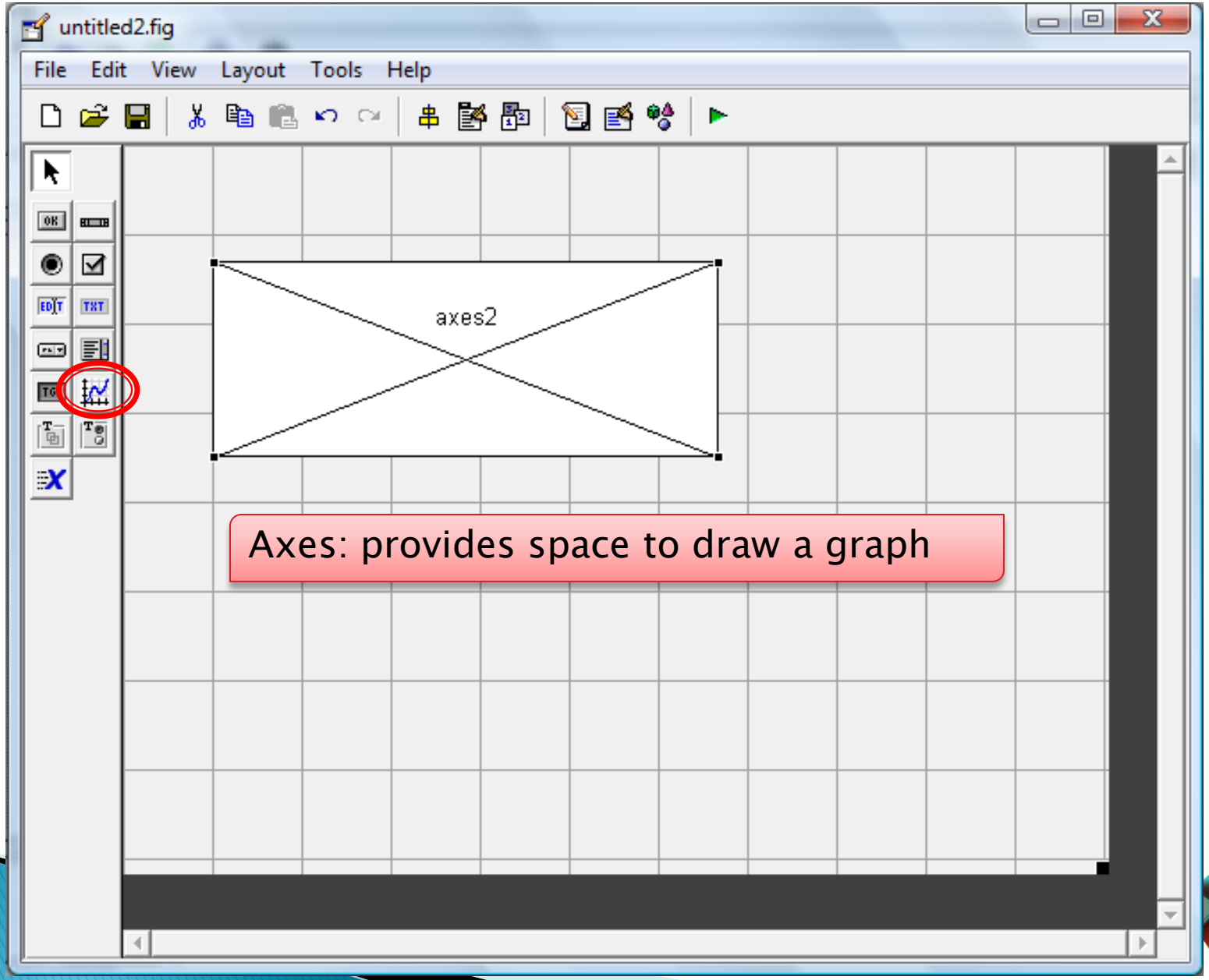
Components


untitled2.fig
File Edit View Layout Tools Help
Listbox
Listbox: a list of strings, which can be edited at run time either by the program or the user

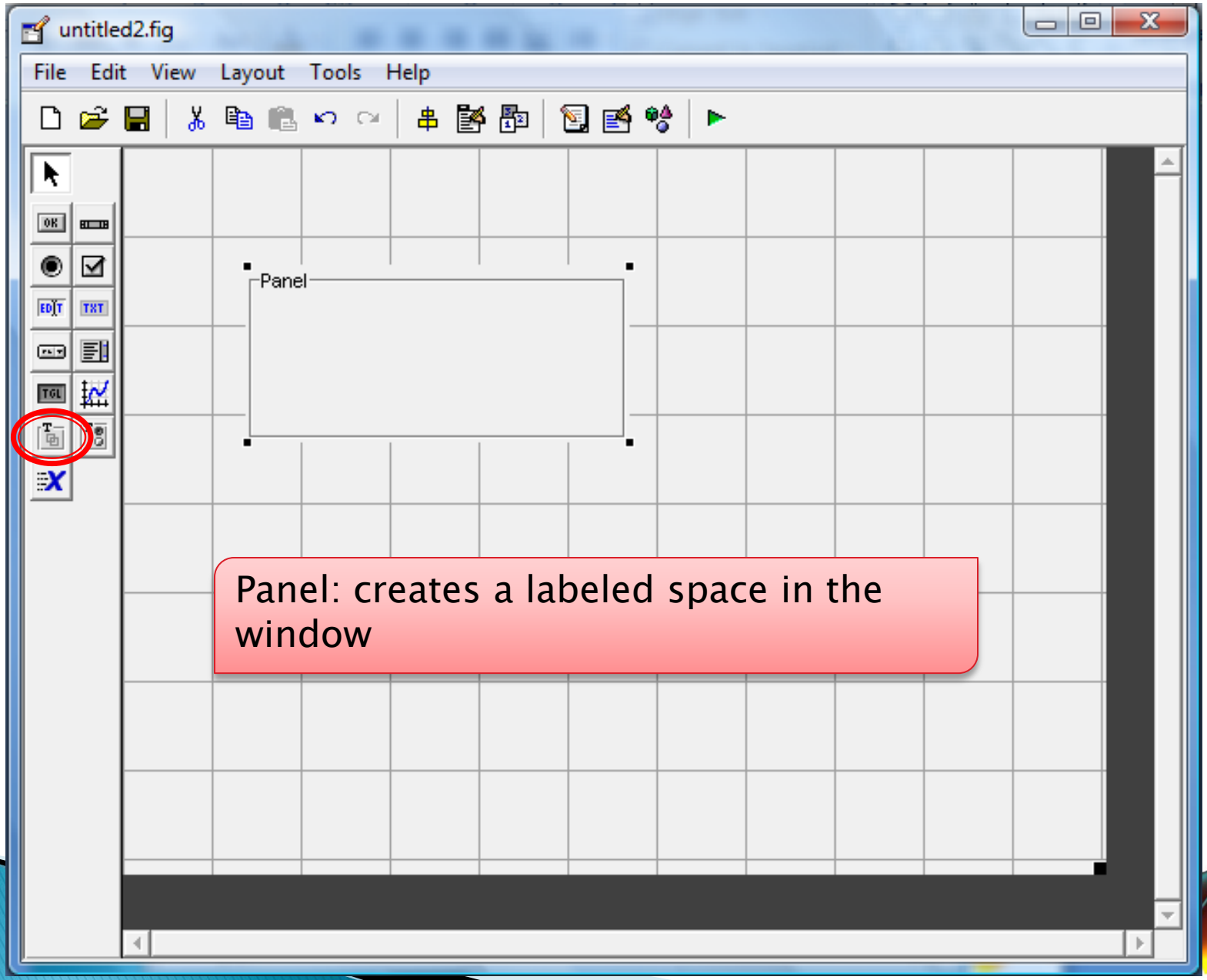
Components


untitled2.fig
File Edit View Layout Tools Help
Toggle Button
Toggle Button: shows the state of some action, by being pressed or unpressed

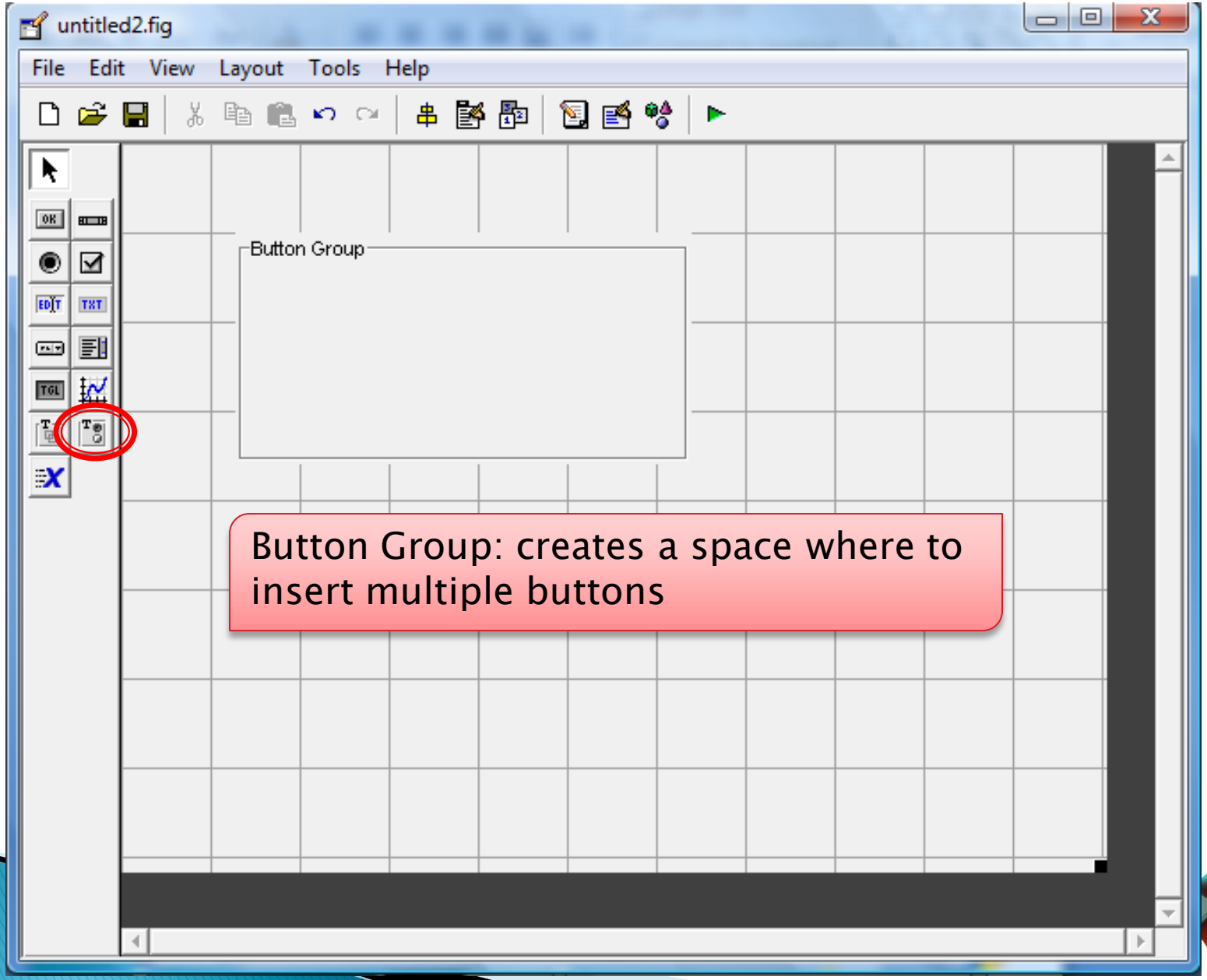
Components


untitled2.fig
File Edit View Layout Tools Help
axes2
Axes: provides space to draw a graph

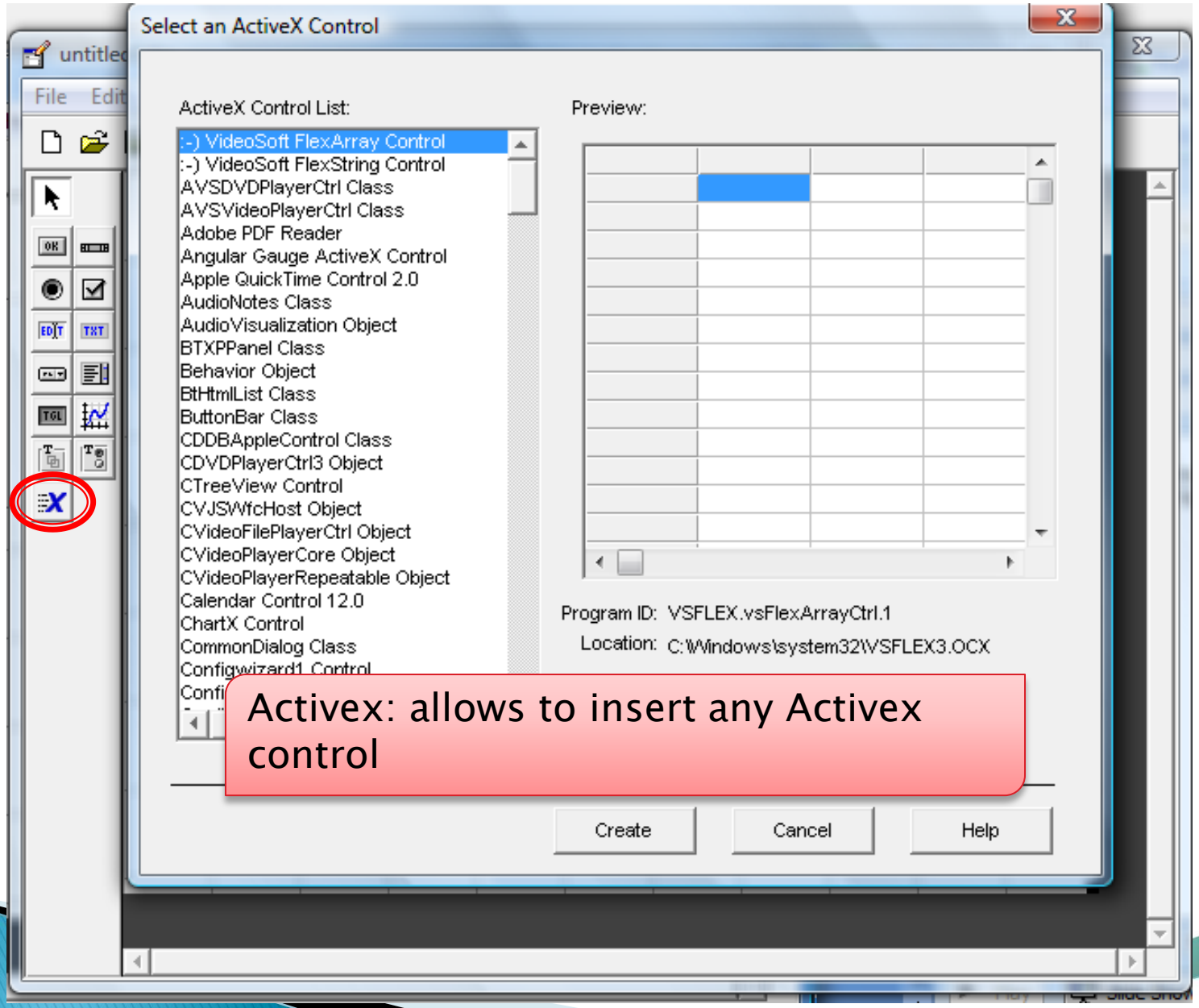
Components


untitled2.fig
File Edit View Layout Tools Help
Panel
Panel: creates a labeled space in the window

Components


untitled2.fig
File Edit View Layout Tools Help
Button Group
Button Group: creates a space where to insert multiple buttons

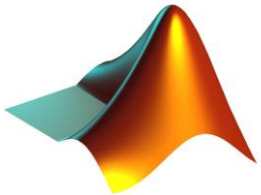
Components


Select an ActiveX Control
ActiveX Control List:
:-) VideoSoft FlexArray Control
:-) VideoSoft FlexString Control
AVSDVDPlayerCtrl Class
AVSVideoPlayerCtrl Class
Adobe PDF Reader
Angular Gauge ActiveX Control
Apple QuickTime Control 2.0
AudioNotes Class
AudioVisualization Object
BTXPPanel Class
Behavior Object
BtHtmlList Class
ButtonBar Class
CDDBAppleControl Class
CDVDPlayerCtrl3 Object
CTreeView Control
CVJSWfcHost Object
CVideoFilePlayerCtrl Object
CVideoPlayerCore Object
CVideoPlayerRepeatable Object
Calendar Control 12.0
ChartX Control
CommonDialog Class
Configwizard1 Control
Preview:
Program ID: VSFLEX.vsFlexArrayCtrl.1
Location: C:\Windows\system32\VSFLEX3.OCX
Activex: allows to insert any Activex control
Create
Cancel
Help

GUI – Example: Adder

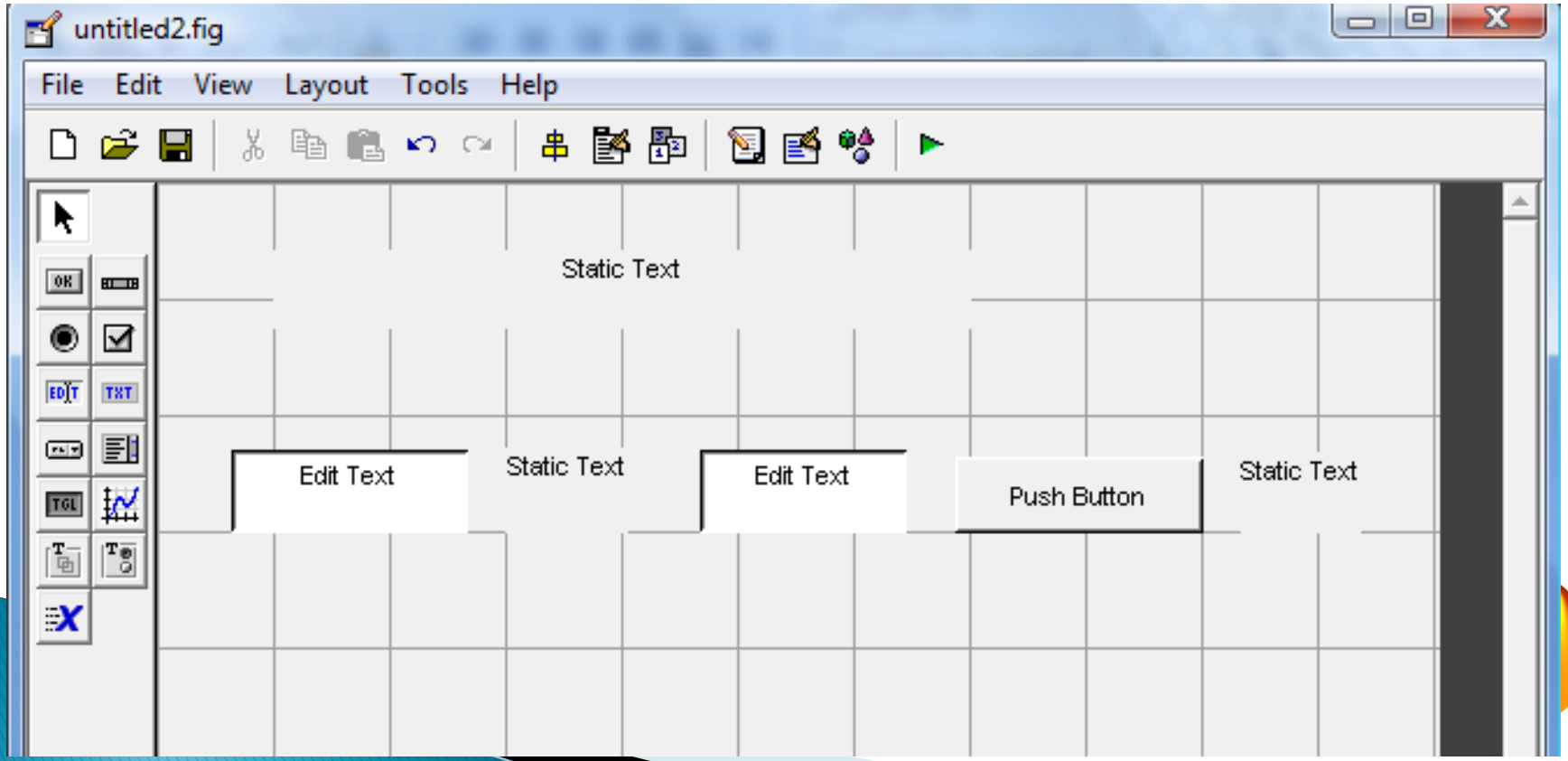
- Create a GUI where the user can insert 2 numbers and we compute their sum

Input1 + Input2 = Result



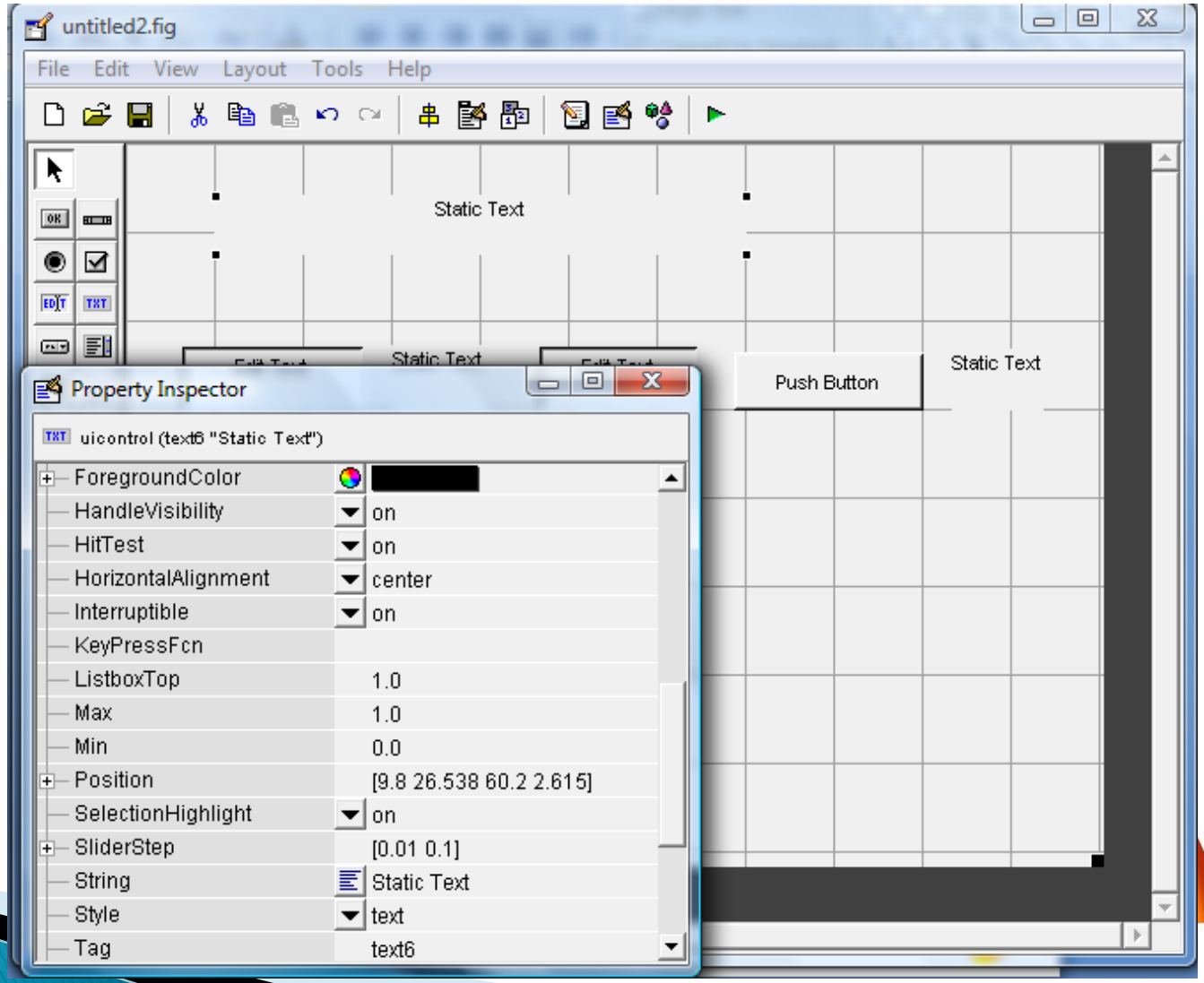
GUI – Example: Adder

- 3 Static Texts
- 2 Edit Texts
- 1 Push Button



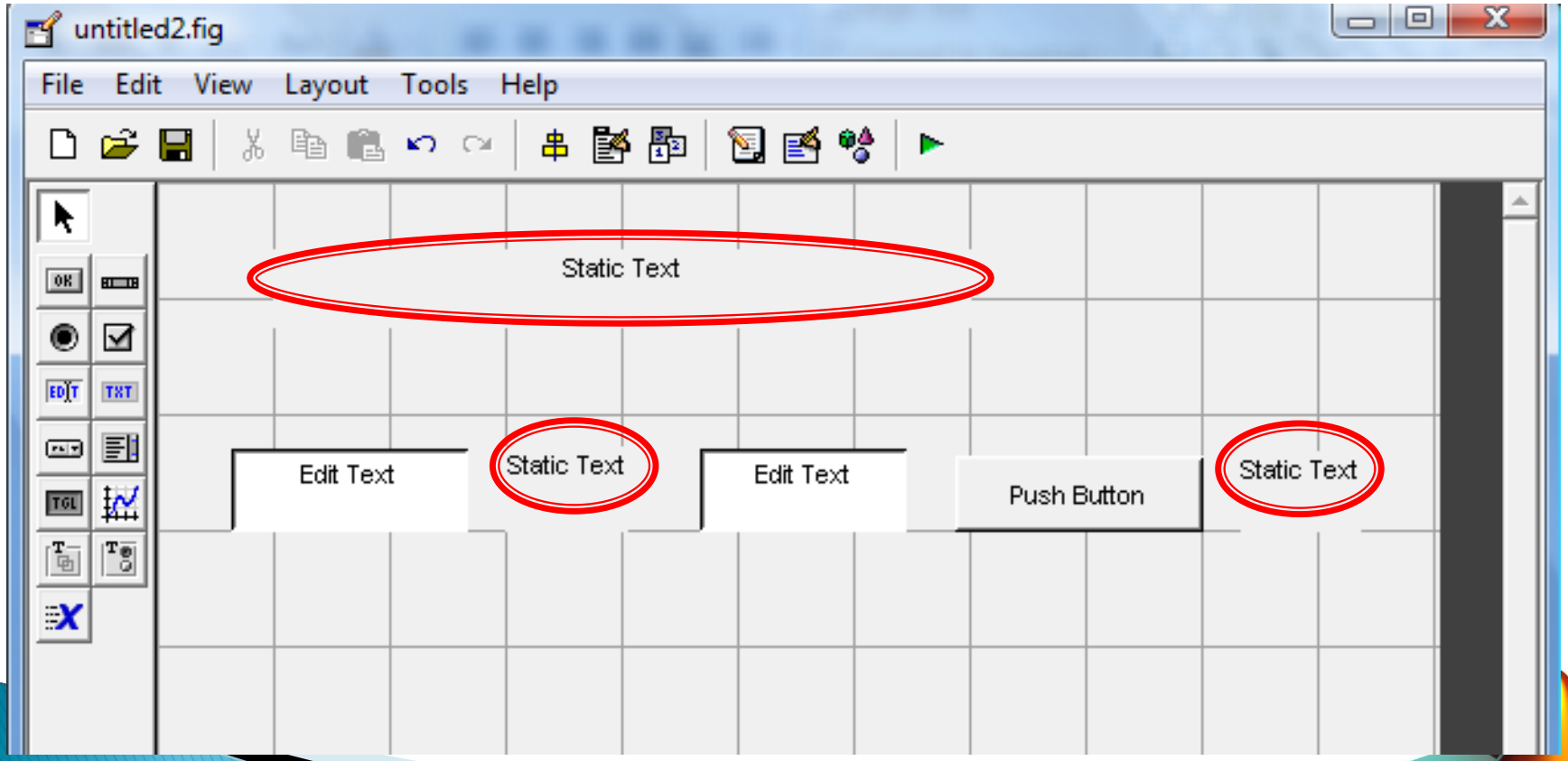
GUI – Example: Adder

- Properties Inspector can be opened by double clicking on item or using the menu bar button



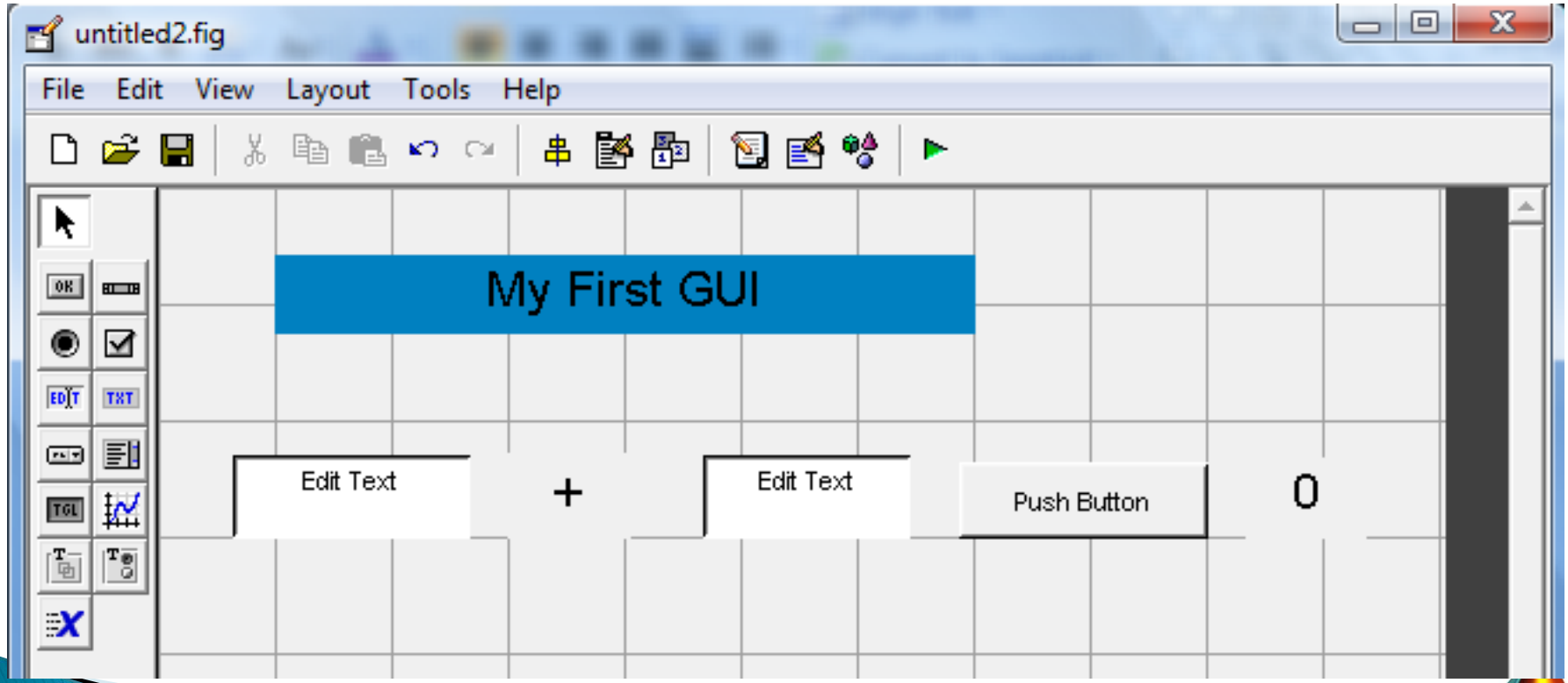
GUI – Example: Adder

- Set the text ‘String’ and ‘FontSize’ properties in the Static Text items



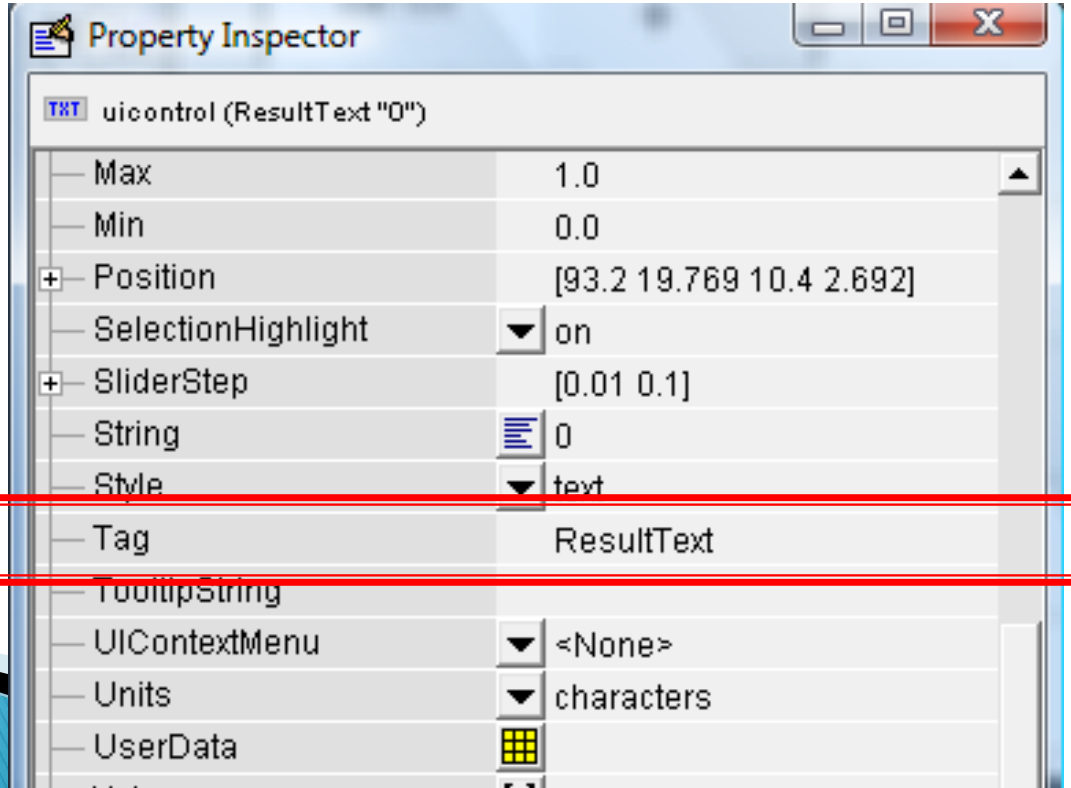
GUI – Example: Adder

- Set the text ‘String’ and ‘FontSize’ properties in the Static Text items



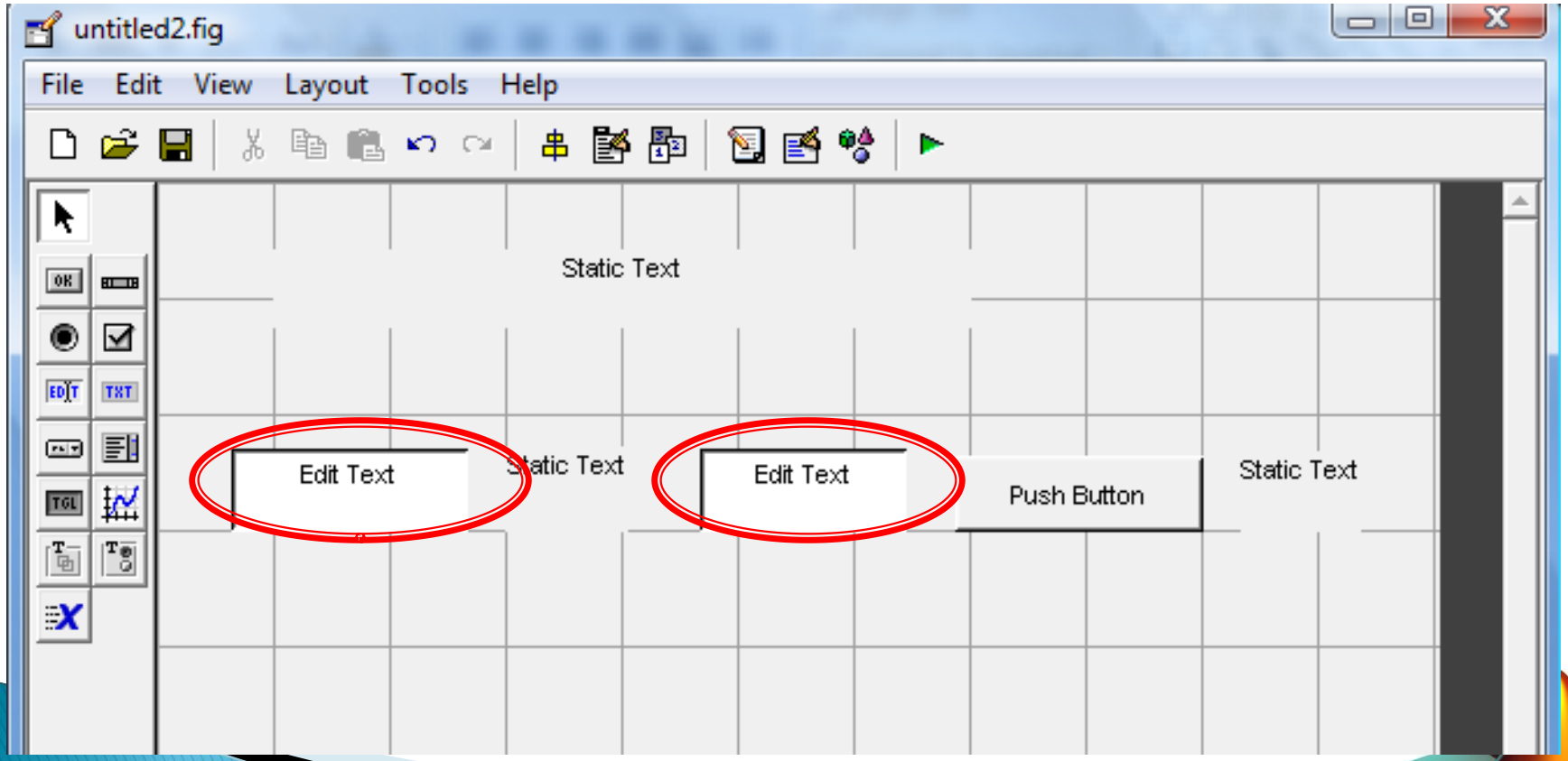
Tagging Elements

- It is very important to assign a unique name to each element, so that we can refer to it in the programming stage
- We can do it by using the ‘tag’ property in the property inspector



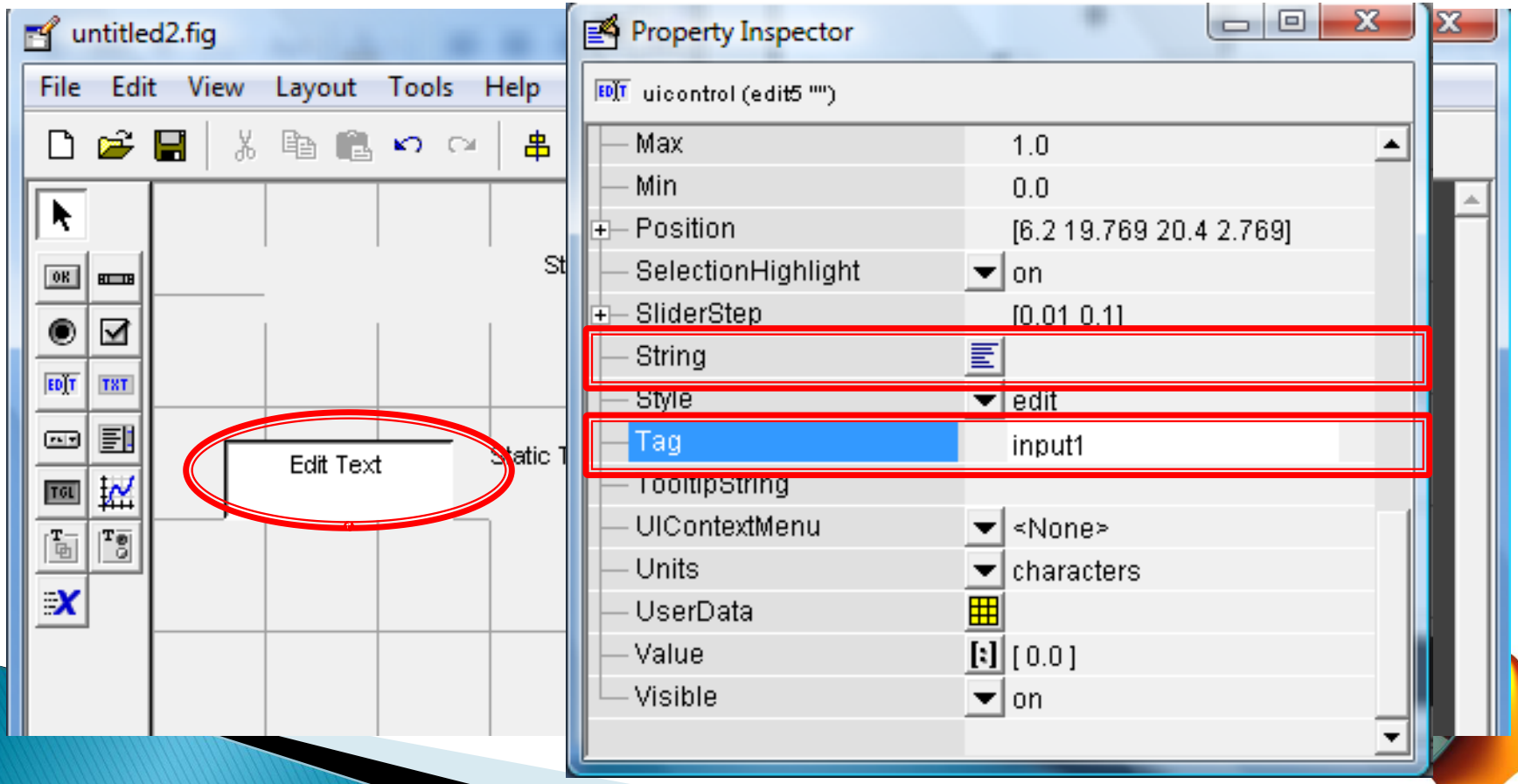
GUI – Example: Adder

- Set the ‘Tag,’ ‘String’ and ‘FontSize’ properties in the Edit Text items



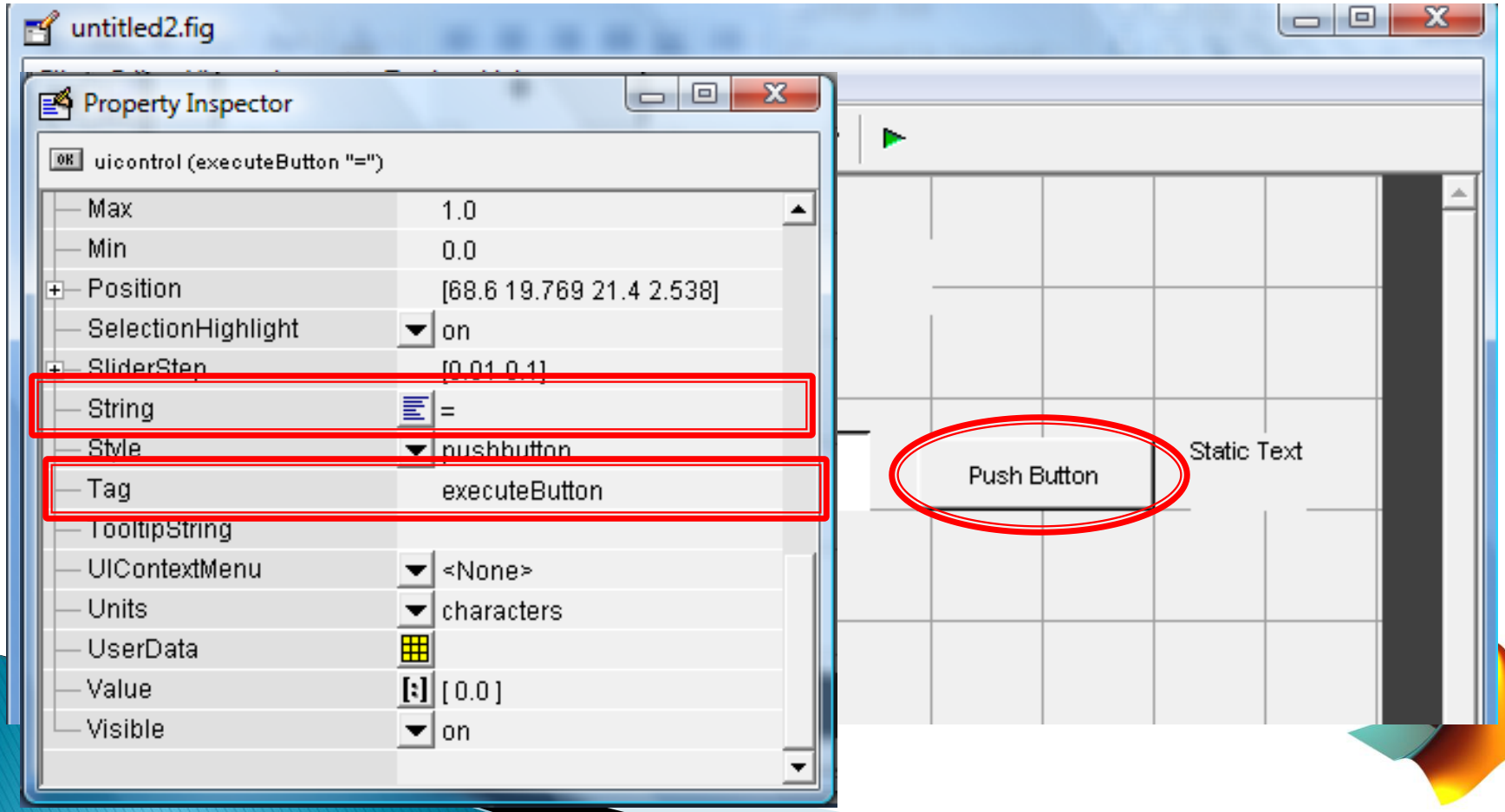
GUI – Example: Adder

- Set the ‘Tag,’ ‘String’ and ‘FontSize’ properties in the Edit Text items

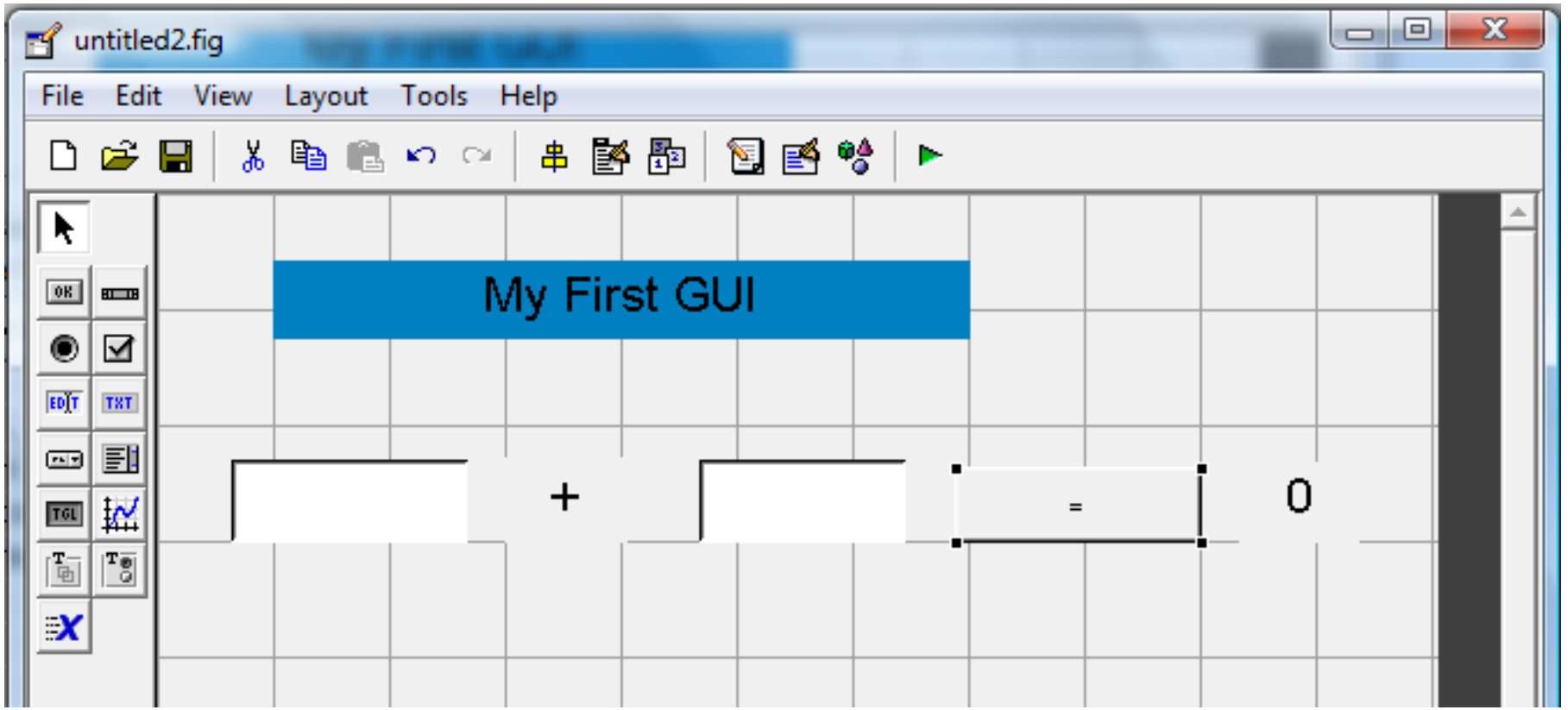


GUI – Example: Adder

- Set the ‘Tag,’ ‘String’ and ‘FontSize’ properties in the PushButton item



GUI – Example: Adder

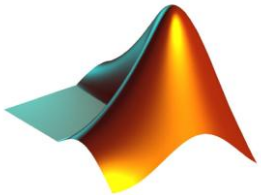


Saving the GUI

- MATLAB saves the GUI in 2 files:

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

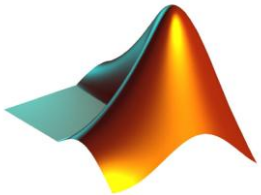
- .m file – containing the code that controls the GUI
  - You can program the callbacks in this file using the M-file Editor



MATLAB Handle Graphics

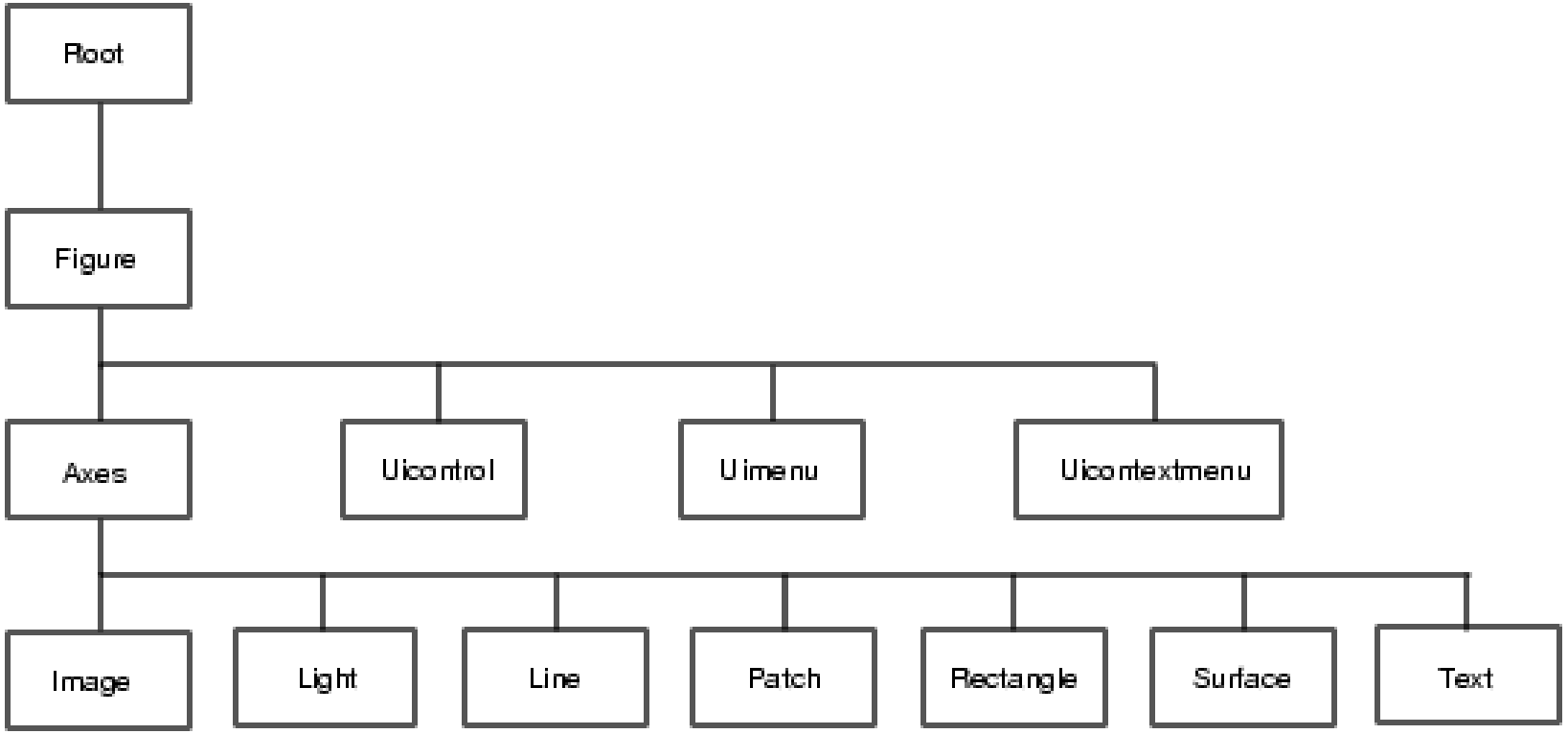
- Handle Graphics refers to the system of graphics objects that MATLAB uses to implement graphing and visualization functions.

- Each graphics objects is defined by:
  - A unique identifier, called handle
  - A set of properties, defining its appearance



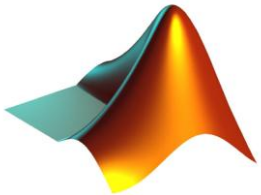
MATLAB Handle Graphics

- Handle Graphics objects are organized in a hierarchy



MATLAB Handle Graphics

- There are 2 ways to obtain an object's handle:
  - Upon creation:
    - `h = plot(x,y);`
  - Using dedicated functions:
    - 0 – root object handle (the screen)
    - gcf – returns the handle for current figure
    - gca – returns the handle for current axes
    - gco – returns the handle for current object
    - gcbf – returns the handle for callback figure
    - gcbo – returns the handle for callback object



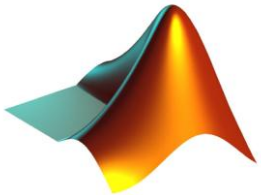
MATLAB Handle Graphics

- Modifying Properties
  - Return a list of all object properties and their current values:
    - `get(handle)`
  - Return current value of an object property:
    - `get(handle, 'PropertyName')`
  - Return a list of all user-settable object properties and their current values:
    - `set(handle)`
    - `set(handle, 'PropertyName', 'NewValue')`
  - Example
    - `x = [-pi:0.01:pi];`
    - `plot(x,sin(x));`
    - `set(gca, 'XDir', 'Reverse');`
  - All the above can also be modified using the Property Editor

Programming the GUI

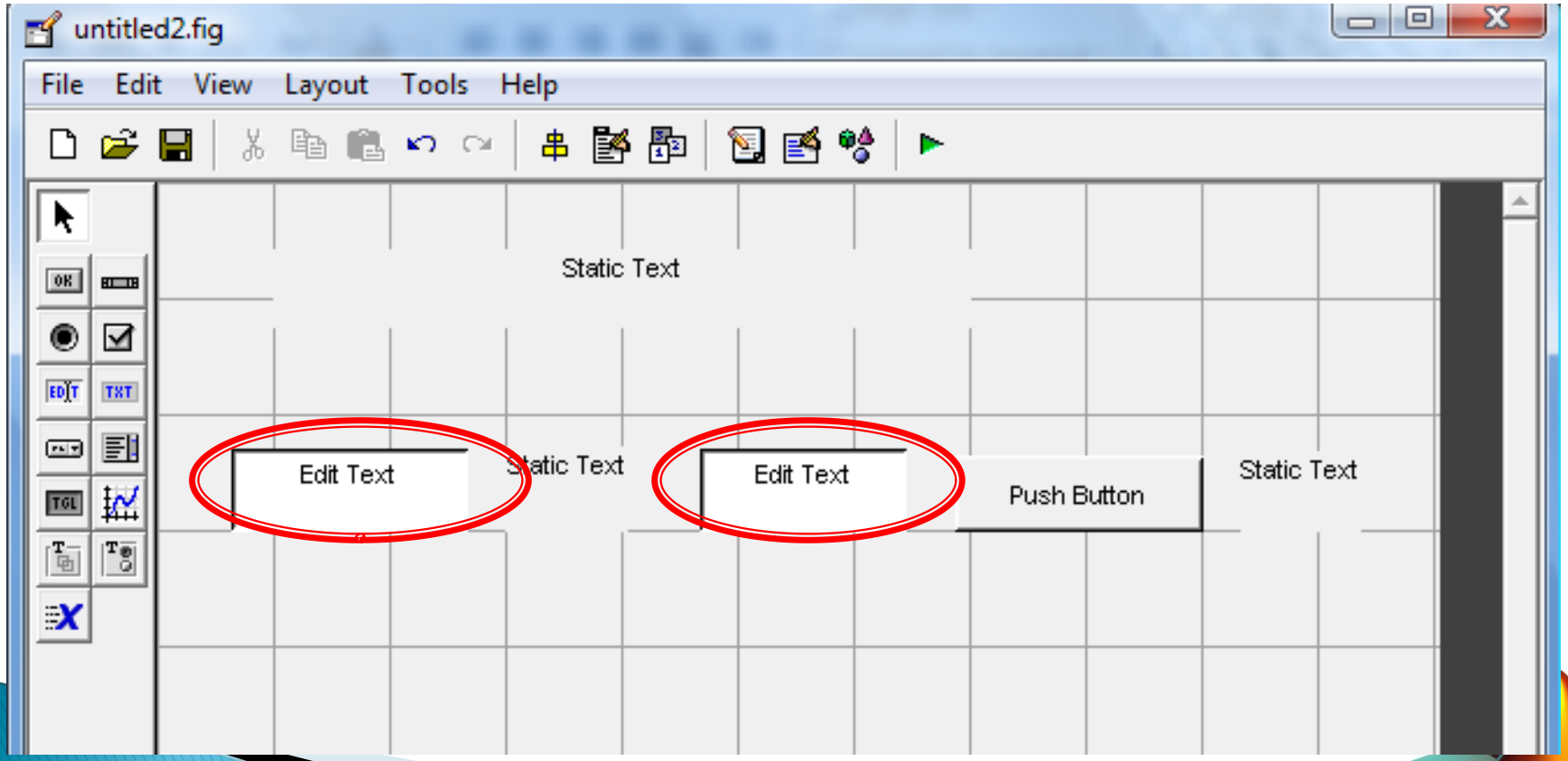
- A **callback** is a sequence of commands that are executed when a graphics object is activated
- Stored in the GUI's M-file
- Is a property of a graphics object (e.g. CreateFnc, ButtonDwnFnc, Callback, DeleteFnc)
- A callback is usually made of the following stages:

1. Getting the handle of the object initiating the action (the object provides event / information / values)
2. Getting the handles of the objects being affected (the object 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



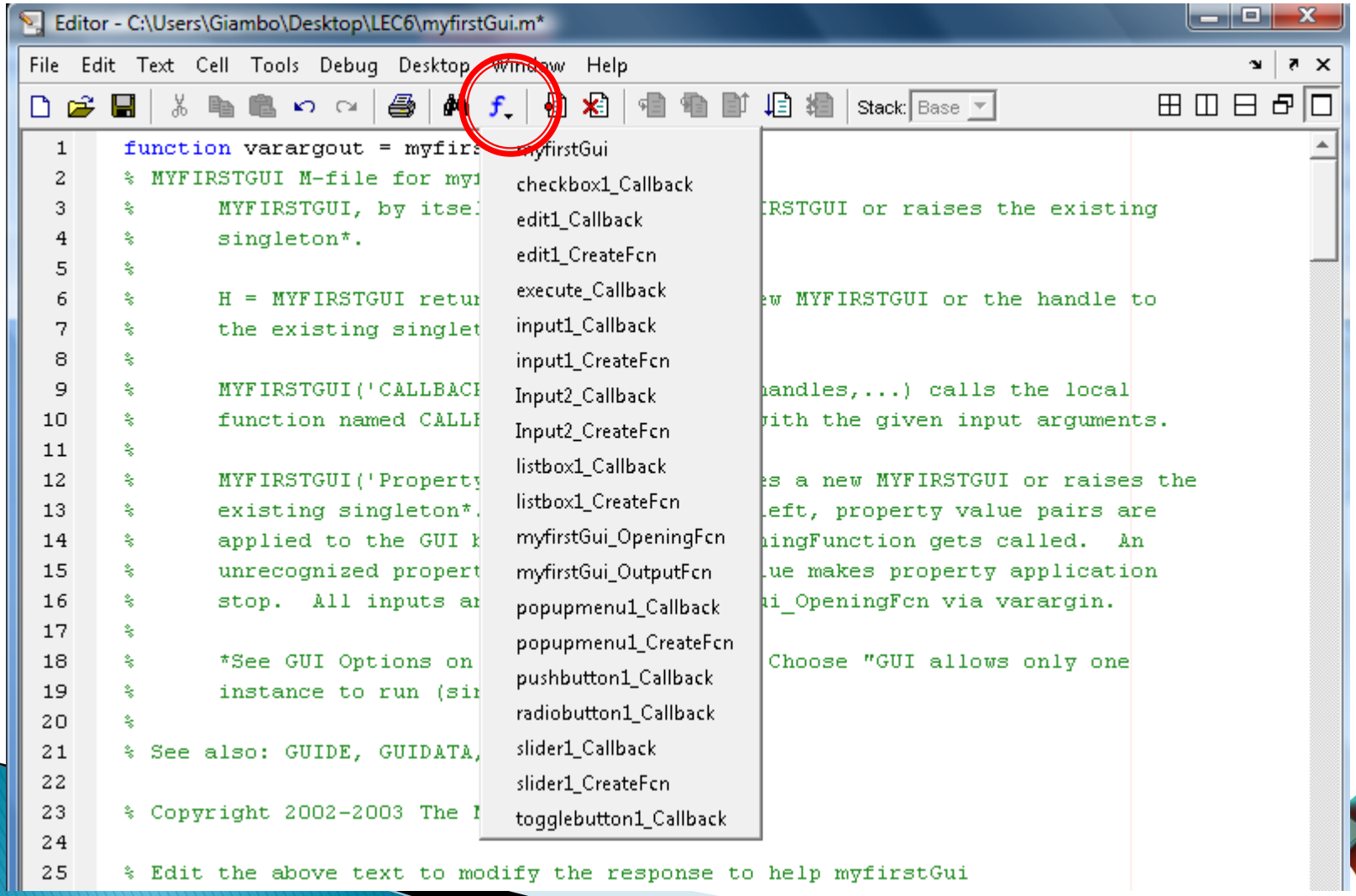
Programming the GUI

- Edit the callbacks for the Edit Text boxes



Programming the GUI

With the f_x menu we can jump to any function in the .m file



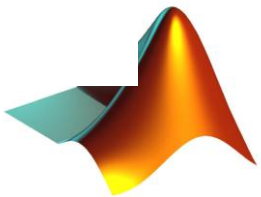
Programming the GUI

```
function Input1_Callback(hObject, eventdata, handles)
% hObject    handle to Input1 (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 Input1 as text
in1 = str2double(get(hObject,'String')); % returns contents of Input1 as a double
if (isempty(in1))
     set(hObject,'String','0')
end
guidata(hObject, handles);

function Input2_Callback(hObject, eventdata, handles)
% hObject    handle to Input2 (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 Input2 as text
in2 = str2double(get(hObject,'String')); % returns contents of Input2 as a double
if (isempty(in2))
     set(hObject,'String','0')
end
guidata(hObject, handles);
```



Programming the GUI

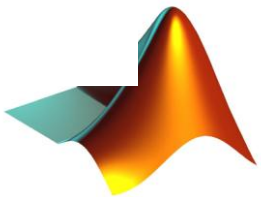
```
function Input1_Callback(hObject, eventdata, handles)
% hObject    handle to Input1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Input1 as text
in1 = str2double(get(hObject,'String')); % returns contents of Input1 as a double
if (isempty(in1))
     set(hObject,'String','0')
end
guidata(hObject, handles);
```

handles is a **struct** containing all the graphics objects in the GUI

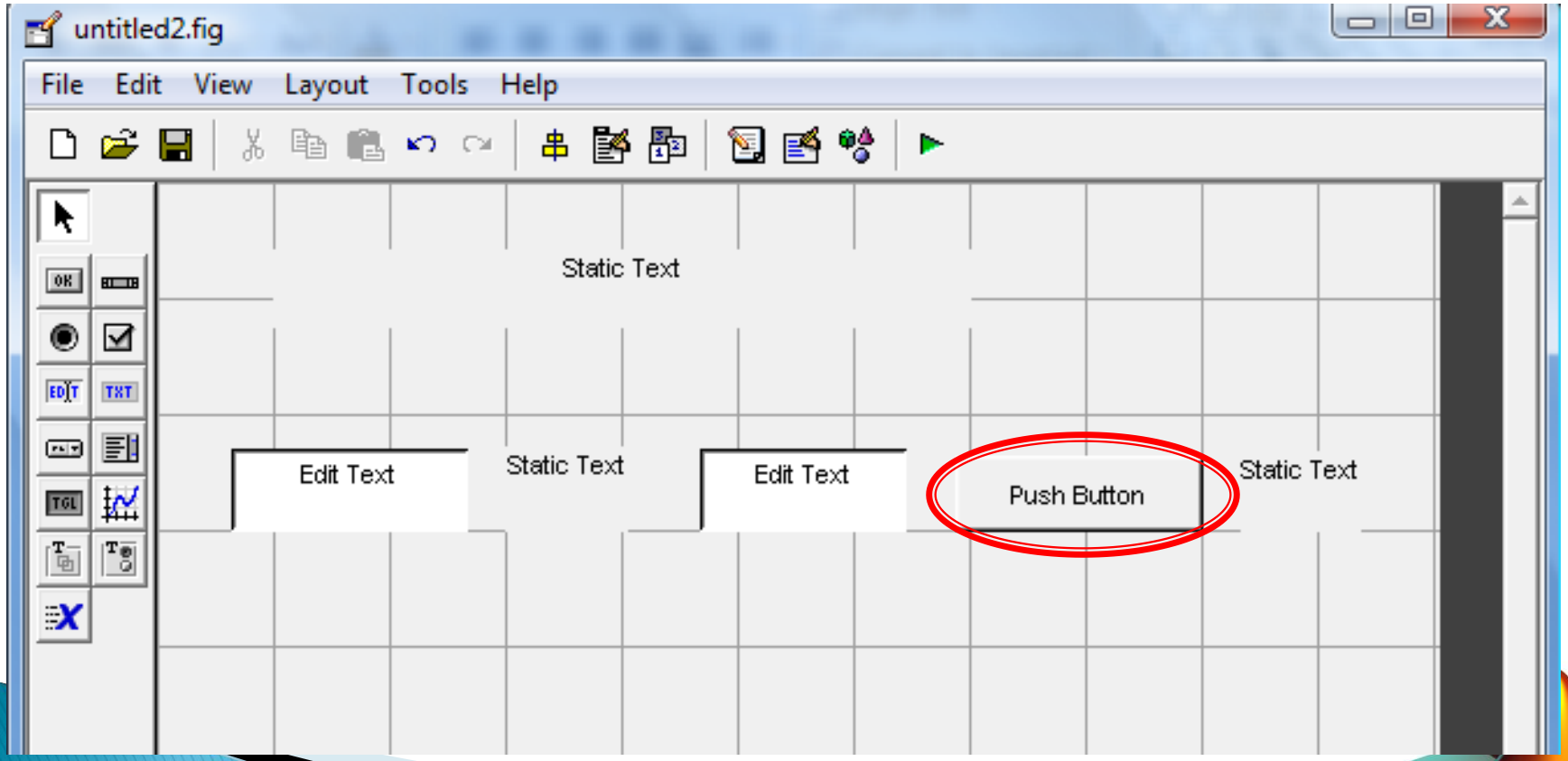
```
function Input2_Callback(hObject, eventdata, handles)
% hObject    handle to Input2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Input2 as text
in2 = str2double(get(hObject,'String')); % returns contents of Input2 as a double
if (isempty(in2))
     set(hObject,'String','0')
end
guidata(hObject, handles);
```



Programming the GUI

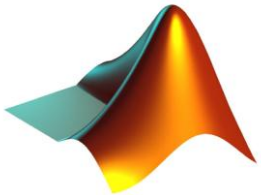
- Edit the callback for the Push Button



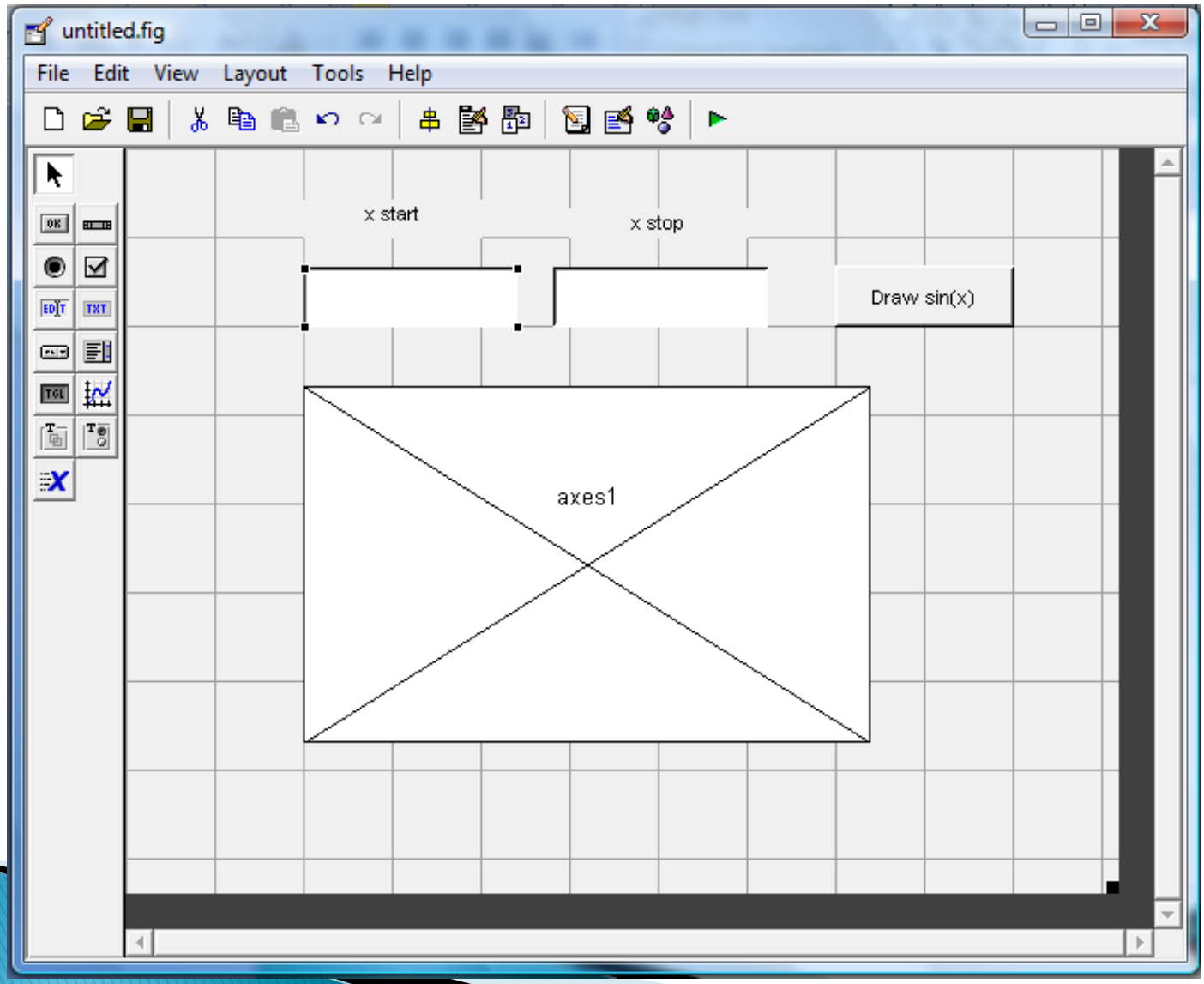
Programming the GUI

```
% --- Executes on button press in Execute.
function Execute_Callback(hObject, eventdata, handles)
% hObject    handle to Execute (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
in1 = get(handles.Input1,'String');
in2 = get(handles.Input2,'String');
% a and b are variables of Strings type, and need to be converted
% to variables of Number type before they can be added together

sum12 = str2num(in1) + str2num(in2);
sum12str = num2str(sum12);
% need to convert the answer back into String type to display it
set(handles.ResultText,'String',sum12str);
guidata(hObject, handles);
```



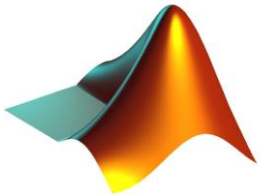
GUI – Example: plot



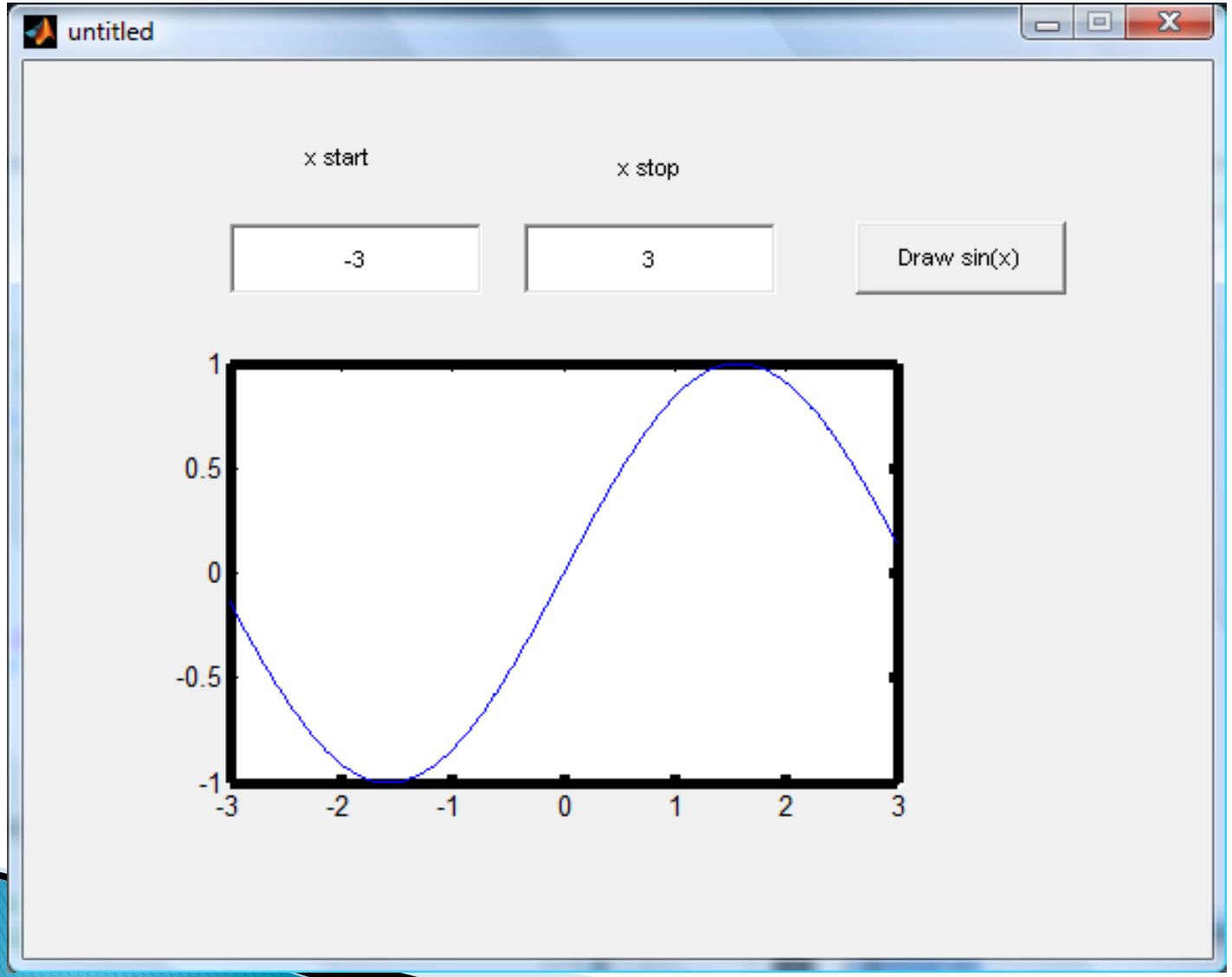
GUI – Example: plot

```
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
xLow = str2num(get(handles.edit1,'String'));
xHigh = str2num(get(handles.edit2,'String'));
x = [xLow : 0.01 : xHigh];

axes(handles.axes1); % select the axes where to draw
plot(x,sin(x));
set(handles.axes1,'LineWidth',4);
guidata(hObject,handles);
```



GUI – Example: plot



GUI – Example: plot

- `im = imread('img.jpg');`
- `image(im,'parent',handles.axes1);`