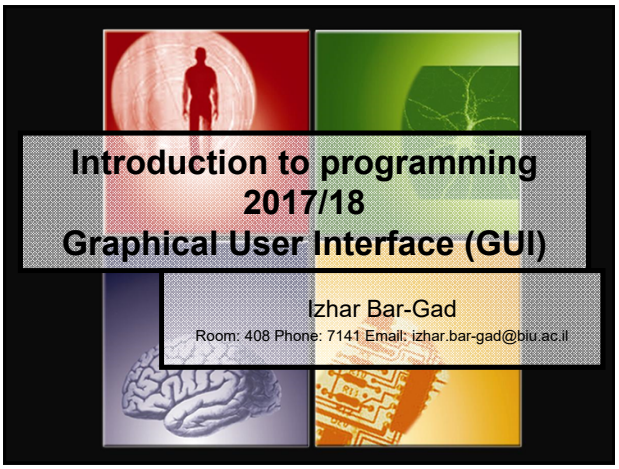
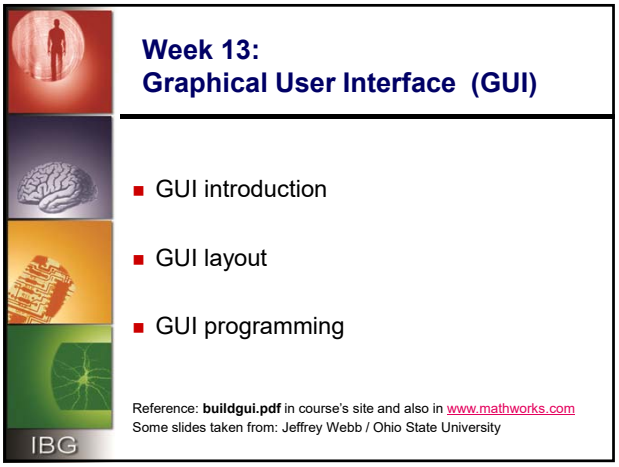
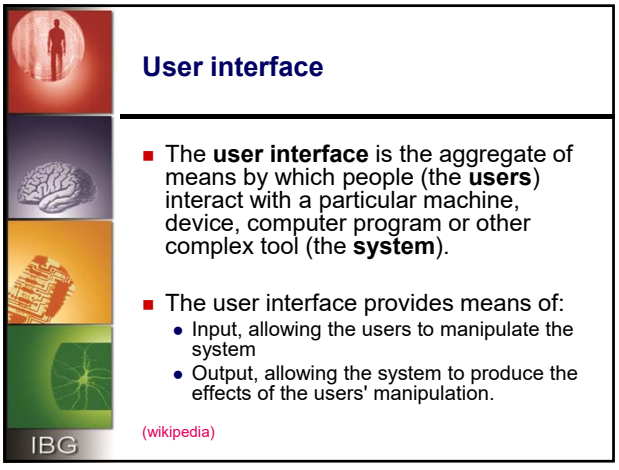
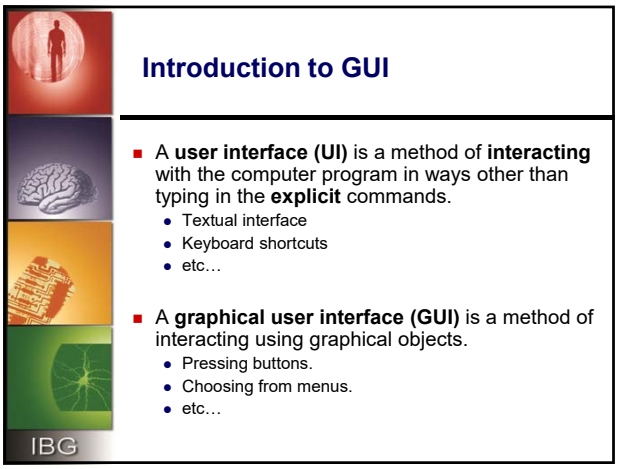
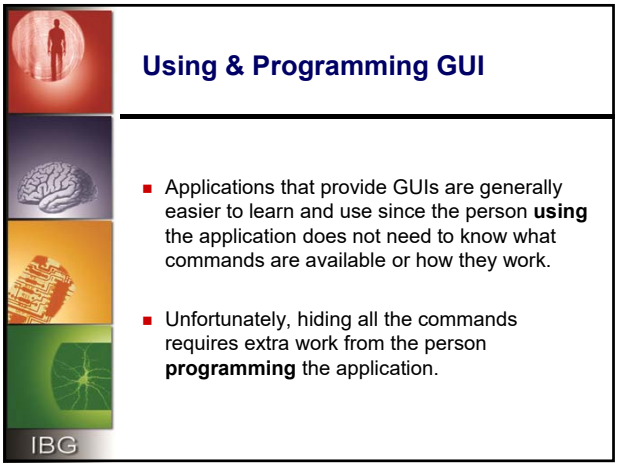
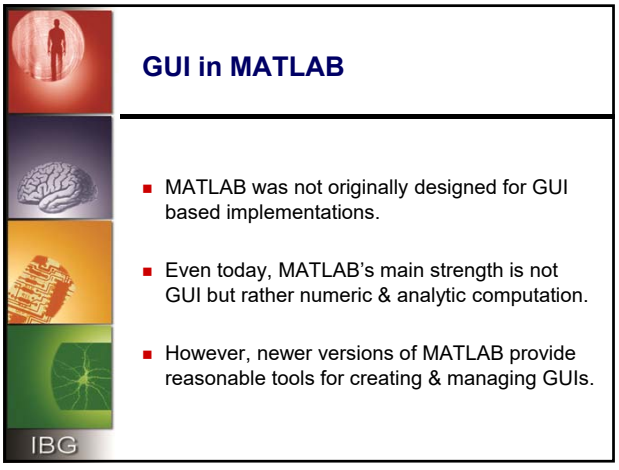

Introduction to programming
2017/18
Graphical User Interface (GUI)
Izhar Bar-Gad
Room: 408 Phone: 7141 Email: izhar.bar-gad@biu.ac.il


Week 13:
Graphical User Interface (GUI)
GUI introduction
GUI layout
GUI programming
Reference: buildgui.pdf in course's site and also in www.mathworks.com
Some slides taken from: Jeffrey Webb / Ohio State University
IBG


User interface
The user interface is the aggregate of means by which people (the users) interact with a particular machine, device, computer program or other complex tool (the system).
The user interface provides means of:
Input, allowing the users to manipulate the system
Output, allowing the system to produce the effects of the users' manipulation.
(wikipedia)
IBG


Introduction to GUI
A **user interface (UI)** is a method of **interacting** with the computer program in ways other than typing in the **explicit** commands.
Textual interface
Keyboard shortcuts
etc…
A **graphical user interface (GUI)** is a method of interacting using graphical objects.
Pressing buttons.
Choosing from menus.
etc…
IBG


Using & Programming GUI
Applications that provide GUIs are generally easier to learn and use since the person **using** the application does not need to know what commands are available or how they work.
Unfortunately, hiding all the commands requires extra work from the person **programming** the application.
IBG


GUI in MATLAB
MATLAB was not originally designed for GUI based implementations.
Even today, MATLAB's main strength is not GUI but rather numeric & analytic computation.
However, newer versions of MATLAB provide reasonable tools for creating & managing GUIs.
IBG

GUI implementation

- Multiple stages which are performed serially.
- GUI design
  - Planning the interaction with the user. Typically includes manual drawing of the UI.
- GUI components layout
  - Laying out figure windows containing various styles of user interface objects (uicontrol, uimenu, ...)
- GUI components programming
  - Programming each UI object to perform the intended action when activated by the user

IBG

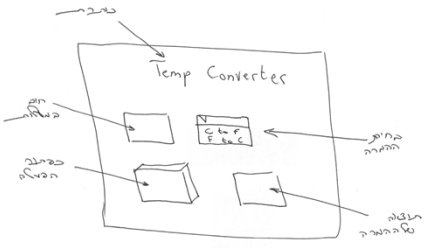
Example GUI

- We will go through the stages of GUI implementation using a simple example.
- The example that we will use is of a simple accessory for transforming the temperature from Celsius to Fahrenheit.

IBG

GUI design - layout

- The program will consist of a single figure with **approximately** the following layout.



IBG

GUI design – use case

- Use cases serve as a technique for capturing **functional requirements** of systems. Each use case provides one or more scenarios that convey how the system should interact with the users (also called actors) to achieve a specific goal. (wikipedia)
- Use cases are useful for describing many **interactive** or **data driven** processes

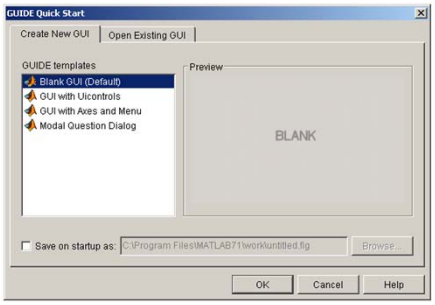
GUI design - functionality

- Design of the functionality should cover all of the use cases. However, this example will examine only the major cases (e.g. will not handle cases of non-numeric temperature).
- Use case I:
  - The user types in the temperature in Celsius in the input **text-box**, chooses "C to F" conversion from the **menu** and presses the convert **push-button**. As a result, the temperature in Fahrenheit is displayed in the output **text-box.**
- Use case II:
  - What happens in the "F to C" conversion?

Laying out the GUI

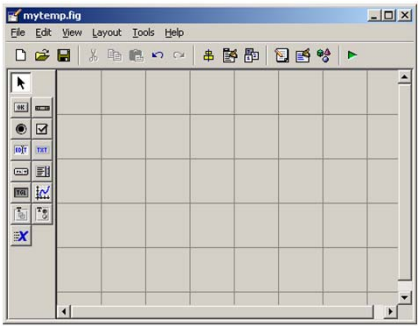
- **GUIDE** (Graphical User Interface Development Environment) is a **WYSIWYG** (What You See Is What You Get) tool used primarily for GUI layout.
- GUIDE generates a FIG-file and a M-file that contains code to handle the initialization and launching of the GUI.
- Layout can also be performed manually using functions for accessing UI objects, much like the graphical objects that we have previously seen. This is somewhat complex and will not be addressed in this lecture…

Starting GUIDE



IBG

GUIDE – Layout editor

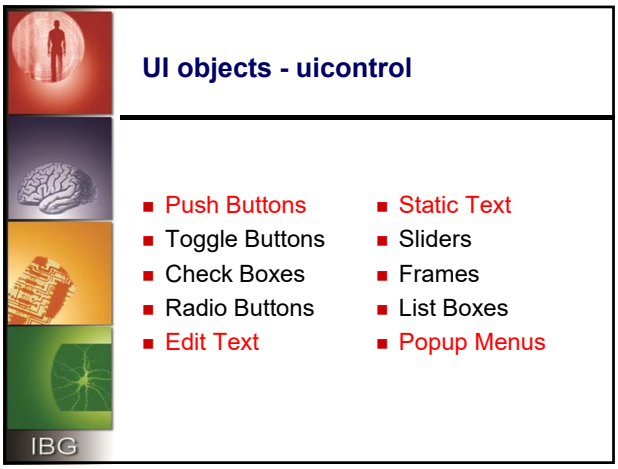


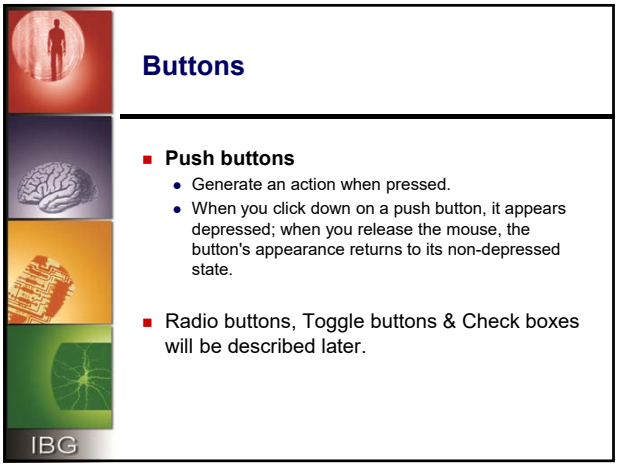
IBG

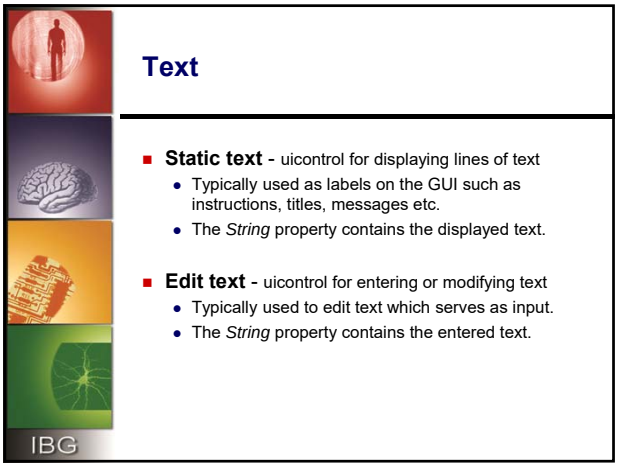
uicontrol

- Many of the GUI objects are **uicontrol**s.
- Usage:
  handle = uicontrol('PropertyName',PropertyValue,...);
- Example
  h = uicontrol('Style', 'pushbutton', 'String', 'Clear', …
  'Position', [20 150 100 70], 'Callback', 'cla');
- Other GUI objects include uimenu, uipanel, uicontextmenu, …

IBG


UI objects - uicontrol
Push Buttons
Toggle Buttons
Check Boxes
Radio Buttons
Edit Text
Static Text
Sliders
Frames
List Boxes
Popup Menus
IBG

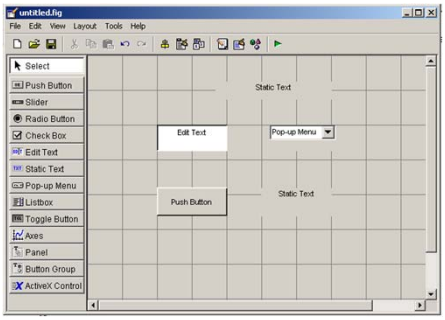

Buttons
Push buttons
Generate an action when pressed.
When you click down on a push button, it appears depressed; when you release the mouse, the button's appearance returns to its non-depressed state.
Radio buttons, Toggle buttons & Check boxes will be described later.
IBG


Text
Static text - uicontrol for displaying lines of text
Typically used as labels on the GUI such as instructions, titles, messages etc.
The String property contains the displayed text.
Edit text - uicontrol for entering or modifying text
Typically used to edit text which serves as input.
The String property contains the entered text.
IBG

Menus

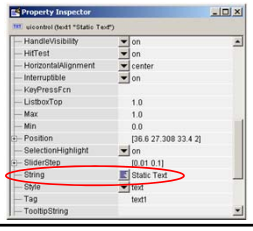
- **Popup Menus**
  - Open to display a list of choices when users press the arrow.
  - The list of choices is defined using the *String* property.
  - When not open, a popup menu displays the current choice.
  - The current choice is determined by the index contained in the *Value* property.

Initial GUI layout



Property Inspector

- Enables setting of component properties in the GUI layout.
- Provides a list of all the properties and displays their current values.

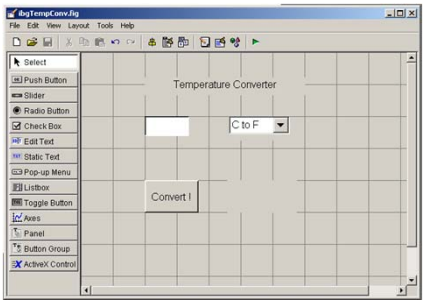


Common properties

- Properties vary depending on uicontrol type.
- Some useful properties:
  - **String** – the string displayed by the uicontrol.
  - **Tag** – name of the uicontrol used for program access to the ui element.
  - **Position** – location and size of the element.
  - **Visible** – Boolean indicator of visibility

IBG

Final GUI layout



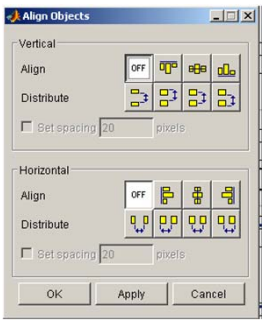
IBG

GUIDE - Additional Layout Tools

- **Alignment Tool**
  - Positioning of objects with respect to each other and to adjustment of spacing between selected objects
  - Specified alignment operations apply to all components that are selected
- **Object Browser**
  - Observing the hierarchical list of the Handle Graphics objects
- **Menu Editor**
  - Allows menus on figures to be interactively modified
  - The Label, Tag, and Callback properties can be modified directly on the tool

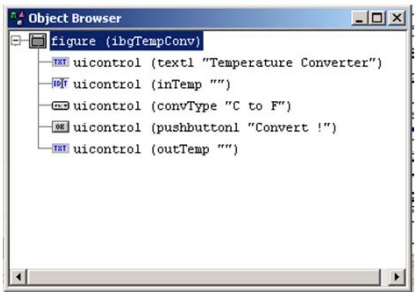
IBG

Alignment tool



IBG

Object browser



IBG

Callbacks

- The M-file generated by GUIDE also provides a framework for the implementation of the **callbacks** .
- Callbacks are functions that are execute when users activate a component in the GUI.
- The callback is a string that is a valid MATLAB expression or the name of an M-file.
  - Callbacks might also be scripts or worse: any MATLAB code. However, we will deal only with functions in this lecture.

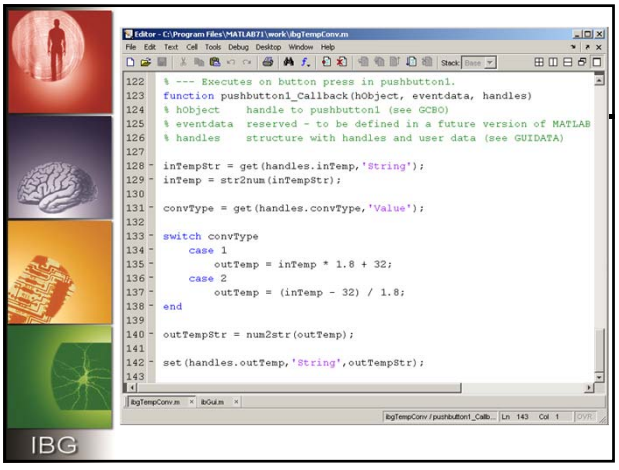
IBG

Callback structure

- A callback is usually made of:
  1. Getting the handle of the object initiating the action.
  2. Getting the handles of the objects being affected.
  3. Getting necessary information / values
  4. The actual processing
  5. Setting the relevant object properties.

GUI handles

- GUI handles provide similar access to the other graphical objects.
- Access to properties is performed through get/set functions.
- All callbacks have a *handles* structure which contains handles to all the GUI objects.
  - The GUI objects are referenced as fields within the structure.
  - The field name is set using the *tag* property of the object in the UI.

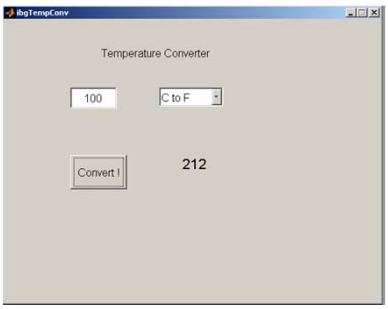


Saving the GUI

- The GUI is saved as two files
- **FIG-file**
  - Binary file containing a complete description of the GUI figure and all of its children (uicontrols and axes), as well as the values of all object properties
  - All of the objects property values are set to the values they were saved with when the figure is recreated
- **M-file**
  - Text file containing the functions that launch and control the GUI and the callbacks. All code, including the callbacks, is contained in the application M-file
  - Each callback is implemented as a **private** function in the M-file

IBG

Running the GUI



IBG

Combining graphics objects

- The GUI may include *axes* graphical objects which allow access to general graphics.
- Multiple axes may exist within the same figure and may be manipulated simultaneously.
- The axes object has callbacks too…

IBG

Additional uicontrol objects I

- **Toggle Buttons**
  - Generate an action and indicate a binary state.
  - The **Value** property is set to 1 when depressed and 0 when not depressed
- **Check boxes**
  - Generate an action when clicked and indicate their state as checked or not checked
  - Useful when providing the user with a number of independent choices that set a mode
  - The **Value** property indicates the state of the check box by taking on the value 1 or 0

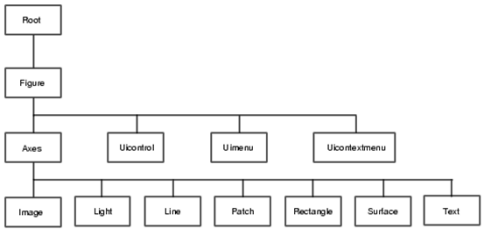
Additional uicontrol objects II

- **Radio buttons**
  - Similar to check boxes, but are intended to be mutually exclusive.
  - To make radio buttons mutually exclusive within a group, the callback for each radio button must set the **Value** property to 0 on all other radio buttons in the group
- **List boxes**
  - Display a list of items (defined using the **String** property) and enable users to select one or more items
  - By default, the first item in the list is highlighted when the list box is first displayed

Additional uicontrol objects III

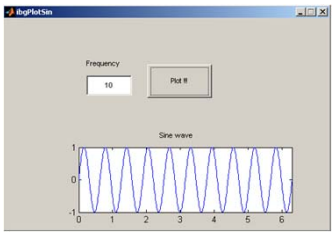
- **Sliders**
  - Accept numeric input within a specific range by enabling the user to move a sliding bar
  - The location of the bar indicates a numeric value
- **Frames**
  - Boxes that enclose regions of a figure window
  - Can make a user interface easier to understand by visually grouping related controls

UIcontrols are part of the hierarchy



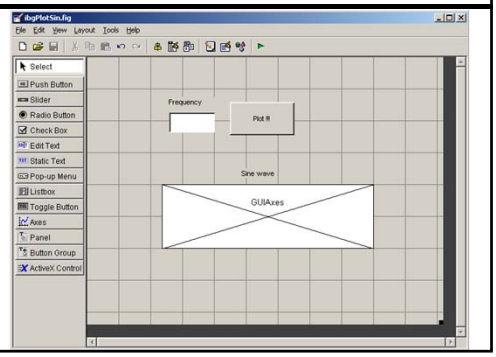
IBG

Combining GUI with graphics



IBG

Layout for graphics example



IBG

Callback for graphics example

```
% Callback of the button press
function Plot_Callback(hObject, eventdata,
    handles)

freqStr = get(handles.Freq,'String');
freq = str2num(freqStr);

axes(handles.GUIAxes);

a = [0:0.01:2*pi];
plot(a, sin(a*freq));
axis([0 2*pi -1 1]);
```

The ‘guidata’ function

- The GUI handles are local to the function. Thus, any change to them is not passed onwards.
- Saving the GUI handles changes is performed by calling the *guidata* function with the graphic object and handles as parameters.

Example:

```
handles.number_errors = 0;
guidata(hObject,handles)
```