GUI Input and Output

Greg Reese, Ph.D
Research Computing Support Group
Academic Technology Services
Miami University
GUI Input and Output
User I/O

Terminology
– GUI: Graphical User Interface
– I/O: Input or Output

Two ways to do user I/O, i.e., get input from the user or show output to the user)

1. In the command-line window
2. Using a GUI
Command-line I/O

Command line advantages

• Easy
• Can use if MATLAB graphics not available
• Automatically converts text input to number
Graphical User Interface (GUI) advantages

- Fairly easy
- Modern (current users familiar with it)
- Clear, convenient displays
- Fun!
Pre-defined dialog boxes

- `dialog` Create and display empty dialog box
- `errordlg` Create and open error dialog box
- `export2wsdlg` Export variables to workspace
- `helpdlg` Create and open help dialog box
- `inputdlg` Create and open input dialog box
- `listdlg` Create and open list-selection dialog box
- `msgbox` Create and open message box
- `printdlg` Print dialog box
- `printpreview` Preview figure to print
- `questdlg` Create and open question dialog box
- `uigetdir` Open standard dialog box for selecting directory
- `uigetfile` Open standard dialog box for retrieving files
- `uigetpref` Specify and conditionally open dialog box according to user preference
- `uiopen` Interactively select file to open and load data
- `uiputfile` Open standard dialog box for saving files
- `uisave` Interactively save workspace variables to MAT-file
- `uisetcolor` Open standard dialog box for setting object's ColorSpec
- `uisetfont` Open standard dialog box for setting object's font characteristics
- `waitbar` Open or update wait bar dialog box
- `warndlg` Open warning dialog box

(From MATLAB documentation)
Pre-defined dialog boxes

Most pre-defined dialog boxes are for getting input from the user

- **dialog**
  Create and display empty dialog box

- **export2wsdlg**
  Export variables to workspace

- **inputdlg**
  Create and open input dialog box

- **listdlg**
  Create and open list-selection dialog box

- **printdlg**
  Print dialog box

- **questdlg**
  Create and open question dialog box

- **uigetdir**
  Open standard dialog box for selecting directory

- **uigetfile**
  Open standard dialog box for retrieving files

- **uigetpref**
  Specify and conditionally open dialog box according to user preference

- **uiopen**
  Interactively select file to open and load data

- **uiputfile**
  Open standard dialog box for saving files

- **uisave**
  Interactively save workspace variables to MAT-file

- **uisetcolor**
  Open standard dialog box for setting object’s ColorSpec

- **uisetfont**
  Open standard dialog box for setting object’s font characteristics
Pre-defined dialog boxes

Some pre-defined dialog boxes are for showing output to the user

- `errordlg`: Create and open error dialog box
- `helpdlg`: Create and open help dialog box
- `msgbox`: Create and open message box
- `printpreview`: Preview figure to print
- `waitbar`: Open or update wait bar dialog box
- `warndlg`: Open warning dialog box
Pre-defined dialog boxes are either *modal* or *non-modal* (*modeless*)

- **Modal dialog box**
  - Prevents user from going to any other window in program (including command line) until user closes box
  - If used in script or function, code keeps running!
    - Use `uiwait()` to pause code
  - Examples: open file dialog, get user input dialog
Pre-defined dialog boxes

• Non-modal (modeless) dialog box
  – Permits user to go to any other window in program (including command line)
  – If used in script or function, code keeps running!
    • Use `uiwait()` to pause code
  – Examples: error dialog, message dialog
GUI input

Input dialog box

\[ \text{answer} = \text{inputdlg}(\text{prompt}) \]

- \textit{prompt} is a text string or cell array containing prompts
- \textit{answer} is cell array, same size as prompt
- Dialog box is modal

(Text strings and cell arrays discussed in other lectures)
GUI input

Example

>> name=inputdlg( 'Full name' )
name = 'Greg Reese'

>> whos name
    Name      Size  Bytes  Class    Attributes
    name      1x1     132  cell

>> n = name{1}
n = Greg Reese

>> whos n
    Name      Size  Bytes  Class    Attributes
    n         1x10     20  char
GUI input

Canceling returns empty cell array

>> name=inputdlg('Full name')
name = {}
>> isempty(name)
an
ans = 1
GUI input

Numerals returned as text, i.e., not converted to numbers

- Use `str2double` or `str2num` to convert

```matlab
>> weight=inputdlg('Weight')
weight = '212.5'
>> w = weight{1};
>> whos w
    Name      Size  Bytes  Class    Attributes
    w         1x5      10  char     
>> w = str2double( weight{1} );
>> whos w
    Name      Size  Bytes  Class     Attributes
    w         1x1       8  double   
```
GUI input

Try it

Put up an input box that asks for the user's car manufacturer and display the string "Your car was made by xxx" where "xxx" is the name the user entered.

Hint: concatenate the first part of the sentence with the user's response and display the result with \texttt{disp()}

\begin{verbatim}
>> name = inputdlg( 'Your car's manufacturer' )
name = 'Toyota'
>> disp( [ 'Your car was made by ' name{1} ] );
Your car was made by Toyota
\end{verbatim}
GUI input

str2double()

Call as  \[ x = \text{str2double}('\text{str}') \]

- \text{str} is a string that is text representation of one real or complex number
  - \( x \) is a double-precision number
  - if \text{str} doesn't represent a number, \( x \) is NaN

Call as  \[ x = \text{str2double}(\ C \) \]

- \( C \) is a cell array of strings
  - \( x \) is an array of double-precision numbers
  - \( x \) is same size as \( C \)
str2num()  

Call as \( x = \text{str2num('str')} \)

\textit{str} is a string that is text representation of a scalar or matrix of real or complex numbers
- \( x \) is same size as matrix in \textit{str}  
- \( x \) is empty matrix if \textit{str} not correct  
- \textit{str} can contain one or more numbers separated by spaces, commas, or semicolons

### Examples from MATLAB help

<table>
<thead>
<tr>
<th>String Input</th>
<th>Numeric Output</th>
<th>Output Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>'500'</td>
<td>500</td>
<td>1-by-1 scalar double</td>
</tr>
<tr>
<td>'500 250 125 67'</td>
<td>500, 250, 125, 67</td>
<td>1-by-4 row vector of double</td>
</tr>
<tr>
<td>'500; 250; 125; 62.5'</td>
<td>500.0000, 250.0000, 125.0000, 62.5000</td>
<td>4-by-1 column vector of double</td>
</tr>
<tr>
<td>'1 23 6 21; 53.56'</td>
<td>1.23 6 21 53 54 55 56</td>
<td>2-by-5 matrix of double</td>
</tr>
<tr>
<td>'12e-3 5.9e-3'</td>
<td>0.0120 0.0059</td>
<td>vector of double</td>
</tr>
<tr>
<td>'uint16(500)'</td>
<td>500</td>
<td>16-bit unsigned integer</td>
</tr>
</tbody>
</table>
GUI input

Variation 1 - add title to box

answer = inputdlg(prompt,dlg_title)

Dlg_title – title of dialog box

>> name = inputdlg( 'Name', 'Character Information' )
name = 'Suleiman the Magnificent'

Title doesn't appear fully in Windows 7 (MATLAB bug!)
GUI input

Variation 2 - number of entered lines

\[
\text{answer} = \text{inputdlg(} \text{prompt,dlg\_title,} \text{num\_lines})
\]

\text{num\_lines} – number of lines user can enter

– See Help if have multiple prompts

\[
>> \text{name} = \text{inputdlg(} \text{'}\text{Names}', \text{'}\text{Character Information}', 3) \)
\]
\text{name} = [3x18 char]

\[
>> \text{name}\{1\}
\]
\text{ans} =

Voldemort
Dark Lord
Tom Marvolo Riddle
Variation 3 - default value

answer = inputdlg(prompt,dlg_title,...
    num_lines,defAns)

defAns – default answer to display
    – See Help if have multiple prompts

>> name = inputdlg('Favorite character', ...'
    'Character Information', 1, 'Hermione')

??? Error using ==> inputdlg at 113
Default Answer must be a cell array of strings.

>> name = inputdlg('Favorite character',...'
    'Character Information', 1, {'Hermione'})
name = 'Hermione'
Variation 4 - miscellaneous options

```matlab
answer = inputdlg(prompt,dlg_title,...
    num_lines,defAns,options)
```

**options** – if 'on', **can resize box horizontally**

– See Help for other **options** options!

```matlab
>> name = inputdlg( 'Favorite character',...
    'Character Information', 1,{'Hermione'} );
```
Try it

Make this dialog box and print the user's input as "John Lennon's bride is xxx" where "xxx" is what the user entered.

```matlab
>> name=inputdlg( 'John Lennon''s?', 'Brides' );
>> disp( [ 'John Lennon''s bride is ' name{1} ] )
John Lennon's bride is Yoko Ono
GUI input

Question dialog box

```matlab
button = questdlg( question, title )
```

Lets user answer by pressing buttons

- `question` is a text string or cell array
- `title` is text string displayed in box's title bar
- by default, there are three buttons: 'Yes', 'No', or 'Cancel'
- `button` is text string set to one of above three values, or empty if user closed box
- dialog box is modal
Example

```matlab
>> button = questdlg( 'Save in default folder?', ...
     'Location of results' );
>> if strcmp( button, 'Yes' )
disp( 'Results stored in default folder' );
elseif strcmp( button, 'No' )
folder = uigetdir;
else
disp( 'No results stored' );
end
Results stored in default folder
```

Pressing Yes button makes this output
GUI input

Variations

- Can specify default button
- Can specify text and defaults for two buttons
- Can specify text and defaults for three buttons
- Can specify miscellaneous options

See Help for more information
Variation - specify two buttons

```python
button = questdlg( question, title, str1, str2, default )

- str1 and str2 are a text strings for the two buttons
- default is str1 or str2 and specifies which button is initially selected
```

Try it

Make this dialog box and display the label of the chosen button

```python
>> questdlg( 'Which dessert do you like better?',...
'Food survey', 'Pie', 'Ice cream', 'Ice cream' )
ans = Pie
```
GUI input

Some predefined dialog boxes used to get specific types of input

• Examples: select directory, select file, specify color
GUI input

Standard dialog box for getting file name

```matlab
fileName = uigetfile
```

Displays list of files in current folder for user to select

- If file name valid and file exists, returns file name as character array. Otherwise, displays error message and returns to list
- If user clicks "Cancel" or closes box, returns 0 (not empty cell!)
- Dialog box is modal
Example

```matlab
>> fileName = uigetfile;
>> if ischar( fileName )
    disp( ['File: ' fileName] );
else
    disp( 'User canceled' );
end
```

File: uniqueId_assignment2.m
Variation 1

```matlab
fileName = uigetfile( FilterSpec )
fileName = uigetfile( FilterSpec, ... DialogTitle )
```

- **FilterSpec** - one or more file specifications, usually with wildcard (*)
- **DialogTitle** - title of dialog box

See help on `uigetfile()` for details of FilterSpec
Try it
Make a dialog box whose title is "MATLAB m-files" and whose filter specification is "*.m". Call it and display the result

```matlab
>> fileName = uigetfile( '*.m', ... 'MATLAB m-files' )
fileName = assignment2_solution.m
```
Variation 2

\[ [ \text{fileName pathName} ] = \text{uigetfile()} \]
\[ [ \text{fileName pathName} ] = \text{uigetfile( FilterSpec )} \]
\[ [ \text{fileName pathName} ] = \text{uigetfile( FilterSpec,... DialogTitle )} \]

- **fileName** - name and extension of chosen file
- **pathName** - path
Try it

Make a dialog box whose title is "MATLAB m-files" and whose filter specification is "*.m". Maneuver to a different folder, select an m-file, and display the result.
Try it

```matlab
>> [ fileName pathName ] = uigetfile(...'
'* .m', 'MATLAB m-files' )

fileName = cell_char_compare.m
pathName = C:\Greg\CSA 441\2009\n```
There are a few pre-defined dialog boxes to get special kinds of input. One example is selecting a color. Use `uisetcolor()` to do this.

**Try it**

```matlab
>> plot(1:10)
```
GUI input

Try it

>> c = uisetcolor; \%pick obnoxious color

>>
GUI input

Try it

```matlab
>> axes('color', c)
```
GUI Input

Questions?
GUI output

Message dialog box

\texttt{msgbox}( \texttt{message} )

Displays text to user

- \texttt{message} is a text string, text matrix or string array
- Text is automatically wrapped to fit in box
- User presses "Okay" button to close box
  - Program does \textbf{not} stop executing while waiting for user to close box
- Dialog box is non-modal (code keeps on running)
  - Use \texttt{uiwait()} to pause code
Try it
Call `msgbox` to display "Passed!", type some other MATLAB commands, then close the message box

```matlab
>> msgbox( 'Passed!' )
>> pwd
ans = c:\greg\csa 441\2010\lectures
>> date
ans = 09-Jul-2010
>>
```
uiwait() makes program stop until message box is closed. Two ways to use:

```matlab
>> h = uiwait( msgbox('Hello World') );
>> uiwait( h );
```

or

```matlab
>> uiwait( msgbox('Hello World') );
```

Try it
Call `msgbox within uiwait()` and notice that don't get prompt in command window

```matlab
>> uiwait( msgbox( 'Passed!' ) )
```

Doesn't appear until message box closed
Example

```matlab
keepLooping = true;
while keepLooping
    answer = inputdlg( 'Enter two characters' );
    if length( answer{1} ) ~= 2
        h = errordlg( 'Enter only two characters' );
        uiwait( h );
    else
        keepLooping = false;
    end
end
```

GUI output

42
Variation 1

msgbox( message, title )

title is title of message box

Try it

Display this message box and then close it

>> msgbox( 'You passed', 'RESULT' )
Variation 2

```
msgbox( message, title, icon )
```

**icon** specifies an icon to display

- 'error'
- 'help'
- 'warn'
- 'custom' (specify your own)
- 'none' (no icon. This is the default)
Variation 2
Try it
Display this message box and then close it

>> msgbox('You failed the written part of the exam', 'RESULT', 'warn')
GUI output

Some predefined dialog boxes for output are:

- **errordlg** - Display errors
- **printpreview** - Show current figure as it will print
- **waitbar** - Open or update a wait bar (progress bar)
- **warndlg** - Display warning

All are non-modal
GUI Input and Output

Questions?
The End