

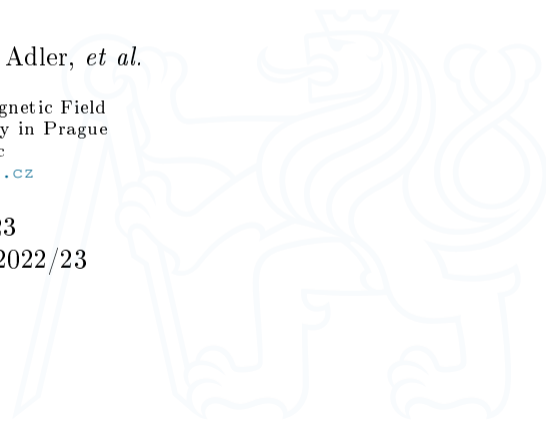
Lecture 8: Static Graphical User Interface

B0B17MTB, BE0B17MTB – MATLAB

Miloslav Čapek, Viktor Adler, *et al.*

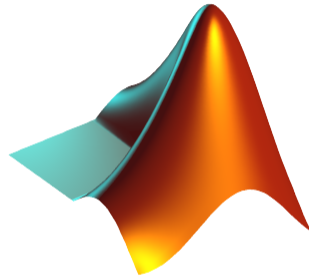
Department of Electromagnetic Field
Czech Technical University in Prague
Czech Republic
matlab@fel.cvut.cz

April 19, 2023
Summer semester 2022/23



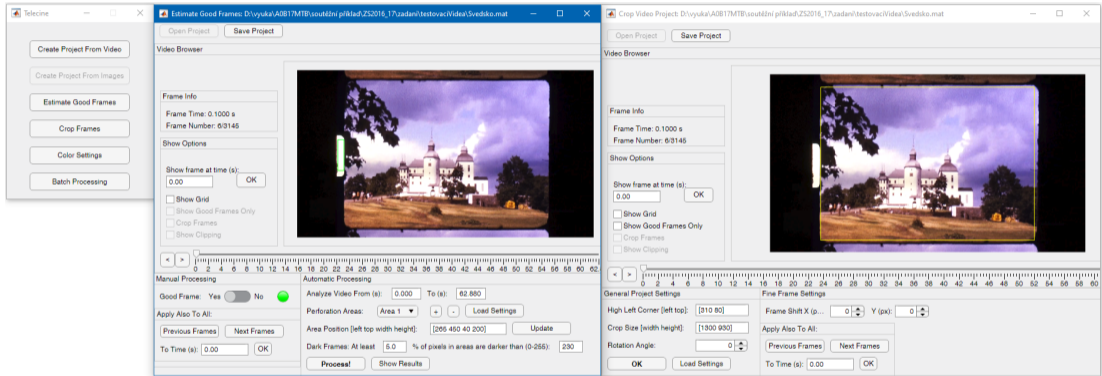


1. Static GUI
2. Exercises





Structure of GUI I.

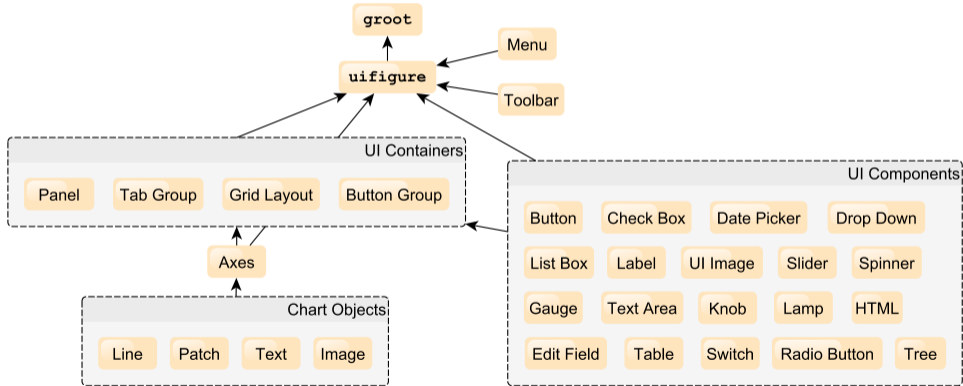


GUI



Structure of GUI II.

► Object Hierarchy (simplified):





Structure of GUI III.

► Basic UI components:

The screenshot displays a window titled "UI Components" containing a grid of 18 different UI components:

- uiaxes:** A 2D plot with x-axis from 0 to 1 and y-axis from 0 to 1.
- uidatepicker:** A date selection interface showing "November 2019" with a calendar grid where the 2nd is selected.
- uiimage:** A placeholder for an image, showing a blue outline of the lion logo.
- uitable:** A table with columns "Name", "Age", and "Authorized".

	Name	Age	Authorized
1	Smith	38	<input checked="" type="checkbox"/>
2	Johnson	43	<input type="checkbox"/>
3	Williams	38	<input type="checkbox"/>
- uitree:** A tree view with two expandable categories: "Runners" (Joe, Linda) and "Cyclists" (Rajeev, Anne).
- uigauge:** A circular gauge with a scale from 0 to 100 and a red needle pointing to approximately 15.
- uibutton:** A simple rectangular button labeled "button".
- uicheckbox:** A checkbox labeled "Check Box" which is currently unchecked.
- uidropdown:** A dropdown menu showing "First Item".
- uiditfield:** A text input field containing the value "3.14".
- uilabel:** A text label with the text "UI Label".
- uistbox:** A list box containing three items: "First Item", "Second Item", and "Third Item".
- uislider:** A horizontal slider with a scale from 0 to 100.
- uispinner:** A spinner box containing the value "20".
- uitextarea:** A text area containing the text "The S-Parameters are a linear model of the DUT."
- uiknob:** A circular knob with a scale from 0 to 100.
- uilamp:** A lamp indicator represented by a green circle.
- uiswitch:** A toggle switch labeled "Off" and "On", currently in the "Off" position.



Screen Properties – root

- ▶ Corresponds to computer screen in Matlab.
- ▶ All other graphical objects are children (descendants) of Root object.
- ▶ In workspace is handle object of class `matlab.ui.Root`.
- ▶ Usually used to find out the number of screens and their resolution.

```
hG = root();
nScreens = size(hG.MonitorPositions, 1);
mainScreenRes = hG.ScreenSize(3:4);
```

```
mainScreenRes = root().ScreenSize(3:4);
```

```
>> root

ans =

  Show all properties
  CallbackObject: [0x0 GraphicsPlaceholder]
  Children: [0x0 GraphicsPlaceholder]
  CurrentFigure: [0x0 GraphicsPlaceholder]
  FixedWidthFontName: 'Courier New'
  HandleVisibility: 'on'
  MonitorPositions: [2x4 double]
  Parent: [0x0 GraphicsPlaceholder]
  PointerLocation: [311 870]
  ScreenDepth: 32
  ScreenPixelsPerInch: 96
  ScreenSize: [1 1 1920 1200]
  ShowHiddenHandles: off
  Tag: ''
  Type: 'root'
  Units: 'pixels'
  UserData: []
```



Graphical Window – uifigure

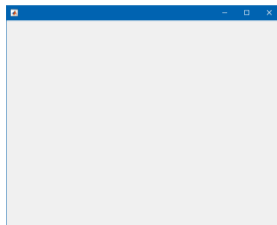
- ▶ Function `uifigure` creates standalone graphical window:

```
hFig = uifigure()
```

- ▶ All figures are descendants of the object `groot`.
- ▶ All secondary graphic objects are descendants of the object `figure` and are drawn in the figure.
- ▶ Object `uifigure` has many properties (see `get(uifigure)`).
- ▶ Figure can be closed using:
 - ▶ `close(hFig)` which call the `CloseRequestFcn` callback:

```
hFig = uifigure('CloseRequestFcn', 'disp(''Your figure will be closed!'')');
```

- ▶ `hFig.delete()`, or `delete(hFig)` directly deletes the figure.
- ▶ `close all force` is for debugging purposes!.





Position of Components

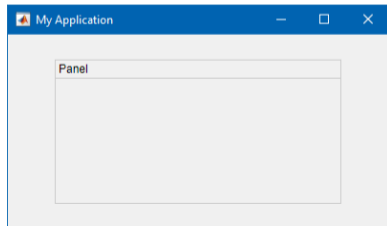
- ▶ MATLAB combines the size of UI components and their position in one vector (property `Position`):

```
uifigure('Position', [left, bottom, width, height]);
```

- ▶ Position is absolute in pixels relative to the parent container by default.
 - ▶ Property Units of UI components set to `'pixels'`.
 - ▶ `uifigure`, `uitab` and `uipanel` supports also `'normalized'`, `'centimeters'`, `'inches'`, `'characters'`, ...
- ▶ Creation of the figure in the center of the screen and a panel in the center of the figure:

```
figSize = [400, 200];
panelSize = [300 150];
screenSize = get(groot, 'ScreenSize');
hFig = uifigure('Position', ...
    [(screenSize(3:4) - figSize)/2, figSize], ...
    'Name', 'My Application');
hPanel = uipanel(hFig, 'Position', ...
    [(figSize - panelSize)/2, panelSize], ...
    'Title', 'Panel');
```

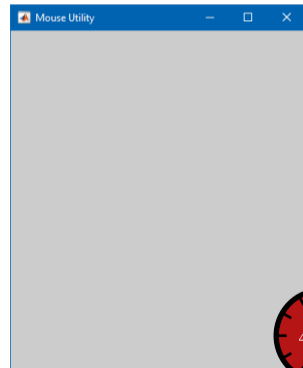
```
hFig = uifigure('Units', 'normalized', ...
    'Position', [0.4 0.4 0.2 0.2]);
```





GUI Window Creation

- ▶ In a new script that we will be extending throughout today's lecture create a figure window that opens in the center of the screen having a width of 350 pixels and a height of 400 pixels.
 - ▶ Make sure the figure's Name is "Mouse Utility".
 - ▶ Change window's Color (up to you).

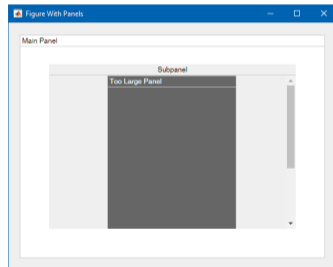




Container Components – uipanel

- ▶ Create panel as a parent to other objects.
- ▶ Objects inside are positioned related to the panel.
- ▶ Many features available (see >> doc `uipanel`)
- ▶ Generally, string class can be used to define properties of UI components.

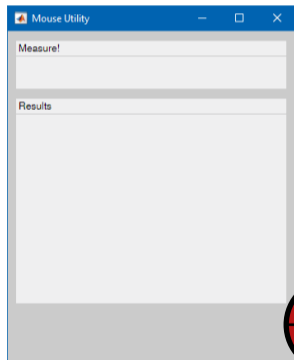
```
hFig = uifigure("Name", "Figure With Panels");
hPanel1 = uipanel(hFig, "Title", "Main Panel", ...
    "FontSize", 12, "BackgroundColor", "white", ...
    "Position", [20, 20, 520, 380]);
hPanel2 = uipanel(hPanel1, "Title", "Subpanel", ...
    "TitlePosition", "centertop", ...
    "Scrollable", true, "BorderType", "none", ...
    "Position", [50 50 420 280]);
hPanel3 = uipanel(hPanel2, ...
    "Title", "Too Large Panel", ...
    "ForegroundColor", "white", ...
    "BackgroundColor", 0.4*ones(1, 3), ...
    "Position", [100 20 220 400]);
```





Panel Creation

- ▶ Into previously created figure create two panels with Position and Title:
 - ▶ [10, 330, 330, 60] px, 'Measure!'
 - ▶ [10, 70, 330, 250] px, 'Results'.
- ▶ Set proper Backgroundcolor, FontWeight, ... (up to you).



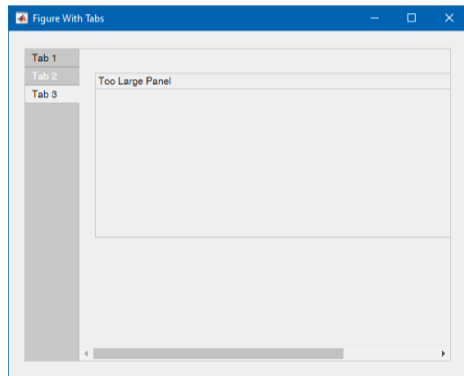


Container Components – uitab

- ▶ Creates a tab that will be a parent for other objects (same as with panel).
- ▶ uitab object has to be children of uitabgroup object.
 - ▶ Manage the tab switching, title positioning, ...

```

hFig = uifigure('Name', 'Figure With Tabs');
hTabGroup = uitabgroup(hFig, ...
    'Position', [20, 20, 520, 380], ...
    'TabLocation', 'left');
hTab1 = uitab(hTabGroup, 'Title', 'Tab 1');
hTab2 = uitab(hTabGroup, 'Title', 'Tab 2', ...
    'BackgroundColor', 'k', ...
    'ForegroundColor', 'w');
hTab3 = uitab(hTabGroup, 'Title', 'Tab 3', ...
    'Scrollable', true);
hPanel = uipanel(hTab3, ...
    'Title', 'Too Large Panel', ...
    'Position', [20, 150, 600, 200]);
hTabGroup.SelectedTab = hTab3;
  
```





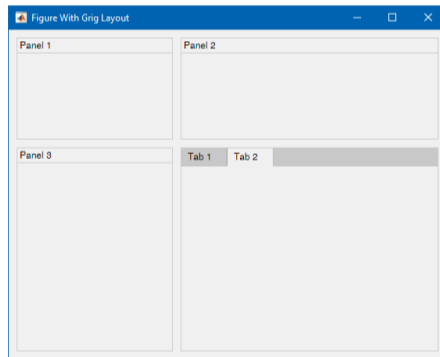
Container Components – uigridlayout

- ▶ Grid layout manages position UI components along the rows and columns of an invisible grid that spans the entire figure or a container within the figure.
- ▶ Gradually added components are placed in column-wise order (as with subplot).
- ▶ It is always possible to edit Layout property of children component to place it to arbitrary cell.

```
hFig = uifigure('Name', 'Figure With Grig Layout');
hGrid = uigridlayout(hFig, ...
    'ColumnWidth', {200, '1x'}, ...
    'RowHeight', {'1x', '2x'});
hPanel1 = uipanel(hGrid, 'Title', 'Panel 1');
hPanel2 = uipanel(hGrid, 'Title', 'Panel 2');

hTabGrop = uitabgroup(hGrid);
hTabGrop.Layout.Column = 2;
hTabGrop.Layout.Row = 2;
uitab(hTabGrop, 'Title', 'Tab 1');
uitab(hTabGrop, 'Title', 'Tab 2');

hPanel3 = uipanel(hGrid, 'Title', 'Panel 3');
hPanel3.Layout.Row = 2;
hPanel3.Layout.Column = 1;
```





Container Components – uibuttongroup

- ▶ Create a button group to manage radio buttons and toggle buttons.
 - ▶ It is not possible to mix these objects in one button group.
- ▶ The children of a button group object can be, beside that, any UI component.

```
hFig = uifigure('Name', 'Figure With Button Group', ...
  'Position', [200 200 250 170]);
hGroup = uibuttongroup(hFig, 'Position', [10, 10, 230, 150], ...
  'Title', 'Toggle Button Group');
hButt1 = uitogglebutton(hGroup, 'Text', 'Option 1', ...
  'Position', [10 90 100 30]);
hButt2 = uitogglebutton(hGroup, 'Text', 'Option 2', ...
  'Position', [10 50 100 30]);
hButt3 = uitogglebutton(hGroup, 'Text', 'Option 3', ...
  'Position', [10 10 100 30]);
hPanel = uipanel(hGroup, 'Title', 'Panel', ...
  'Position', [120, 10, 100, 110]);
```

```
%% Selection via button group
hGroup.SelectedObject = hButt2;
% or use button Value property
hButt2.Value = true;
% get values
values = [hGroup.Buttons.Value]
```

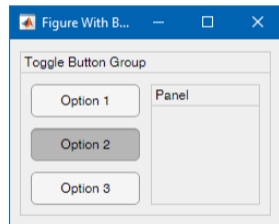


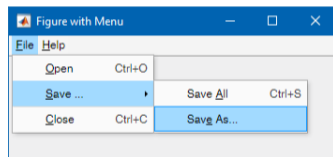


Figure Menu – uimenu

- ▶ Create figure menu items with arbitrary hierarchy.
- ▶ Mnemonic keyboard shortcut (Alt+*mnemonic*) by using the ampersand (&) character in the text for the label.
 - ▶ To use mnemonics, you must specify a mnemonic for all menus and menu items.
- ▶ Accelerator define a keyboard shortcut for selecting a menu item (Ctrl+*acc.*).

```

hFig = uifigure('Name', 'Figure with Menu', ...
    'Position', [200, 200, 350, 110]);
hMenuFile = uimenu(hFig, 'Text', '&File');
hMenuOpen = uimenu(hMenuFile, 'Text', '&Open', ...
    'Accelerator', 'O');
hMenuSave = uimenu(hMenuFile, 'Text', '&Save ...');
hMenuSaveAll = uimenu(hMenuSave, 'Text', 'Save &All', ...
    'Accelerator', 'S');
hMenuSaveAs = uimenu(hMenuSave, 'Text', 'Sav&e As...');
hMenuClose = uimenu(hMenuFile, 'Text', '&Close', ...
    'Accelerator', 'C', 'Separator', true);
hMenuHelp = uimenu(hFig, 'Text', '&Help');
hMenuShowHelp = uimenu(hMenuHelp, 'Text', 'Show &Help');
hMenuWhisper = uimenu(hMenuHelp, 'Text', 'Whis&per', ...
    'Checked', true);
  
```





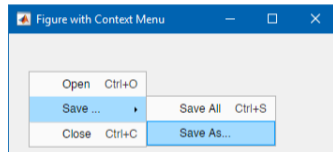
Context Menu – uicontextmenu

- ▶ Context menu is invoked by right-click on an assigned UI object.
- ▶ To assign the context menu to the UI object set its ContextMenu property.
- ▶ Individual items in context menu are uimenu objects.
- ▶ Accelerator shortcuts works even when the context menu is not shown.

```

hFig = uifigure('Name', 'Figure with Context Menu', ...
    'Position', [200, 200, 350, 130]);
hContMenu = uicontextmenu(hFig);
hMenuOpen = uimenu(hContMenu, 'Text', 'Open', ...
    'Accelerator', 'O');
hMenuSave = uimenu(hContMenu, 'Text', 'Save ...');
hMenuSaveAll = uimenu(hMenuSave, 'Text', 'Save All', ...
    'Accelerator', 'S');
hMenuSaveAs = uimenu(hMenuSave, 'Text', 'Save As...');
hMenuClose = uimenu(hContMenu, 'Text', 'Close', ...
    'Accelerator', 'C', 'Separator', true);
%% Assign menu to proper object
hFig.ContextMenu = hContMenu;
%% Open menu programmatically
% at specified position of low left corner
open(hContMenu, 10, 10);

```





Graph Area – uiaxes

- ▶ Defines area where descendants of object `uiaxes` are placed.
- ▶ Axes is created using `uiaxes` command even without actual existence of parental `uifigure` object.

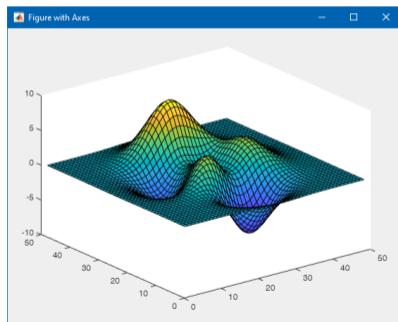
```
hFig = uifigure('Name', 'Figure with Axes');
hAx = uiaxes(hFig, ...
    'Position', [20, 20, 520, 380]);
surf(hAx, peaks);
```

- ▶ Turn off interactivity:

```
hAx.Toolbar.Visible = false;
% disableDefaultInteractivity(hAx)
```

- ▶ Usual plotting functions as `plot`, `surf`, `pcolor`, `stem`, ... support `uiaxes` objects but its handle is needed:

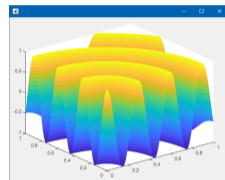
```
plot(hAx, rand(10, 5));
```





Fast Graphics Update

- ▶ Graphics made from user interface (UI) components is significantly slower than graphics made within "old" figure.
- ▶ drawnow and pause works in the same way as with figure.



```
f0 = 1e9;
x = linspace(0, 1, 201);
y = x.';
c0 = 3e8;
lambda = c0/f0;
t = linspace(0, 1/f0);
k = 2*pi/lambda;
R = sqrt(x.^2 + y.^2);
```

```
hFig = uifigure;
hAx = uiaxes('Parent', hFig, 'Position', [10 10 540 400]);
hSurf = surf(hAx, 'Parent', hAx, 'LineStyle', 'none', ...
    'XData', x, 'YData', y, 'ZData', nan(size(R)));
drawnow();
tic;
for thisTime = t
    E = exp(1j*(-k*R + 2*pi*f0*thisTime));
    hSurf.ZData = real(E);
    drawnow();
end
tEnd = toc;
fprintf('Reached FPS: %.2f Hz.\n', length(t)/tEnd);
```



UI Components – uilabel

- ▶ Labels contain static text for labelling parts of an application.
- ▶ Property Interpreter can be `'tex'`, `'latex'` and `'html'` for additional formatting.
- ▶ Labels can be children of all UI containers.
- ▶ All UI component labels and titles can contain unicode characters (`char(uniNumber)`).
 - ▶ Can be found, *e.g.*, here: <http://unicode-table.com/>.

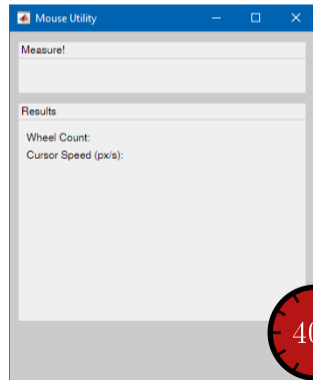
```
hFig = uifigure('Name', 'Figure with Label', ...
  'Position', [200, 200, 400, 100]);
hLabel = uilabel(hFig, 'Text', ...
  ['Fancy Label with ', char(960)], ...
  'HorizontalAlignment', 'Center', ...
  'Position', [10, 10, 380, 80], ...
  'FontName', 'Times New Roman', ...
  'FontSize', 40, ...
  'FontWeight', 'bold', ...
  'FontAngle', 'italic', ...
  'FontColor', 'm', ...
  'BackgroundColor', 0.3*ones(1, 3));
```





Label Creation

- ▶ Create two labels into panel called 'Results' with Position and Text properties as:
 - ▶ [10, 200, 90, 20], 'Wheel Count:',
 - ▶ [10, 180, 120, 20], 'Cursor Speed (px/s):'.
- ▶ Set proper Backgroundcolor, FontWeight, ... (up to you).

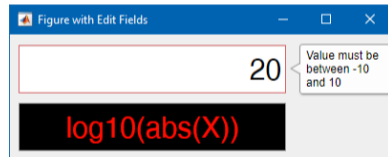




UI Components – uieditfield

- ▶ Create text or numeric edit field component.
- ▶ Function `uieditfield` or `uieditfield('text')` creates edit field handling strings.
 - ▶ See `>> doc EditField` for formatting possibilities.
- ▶ Function `uieditfield('numeric')` creates edit field handling numerical values.
 - ▶ Moreover offers rounding, limits and format (see `>> doc NumericEditField`).

```
hFig = uifigure('Name', 'Figure with Edit Fields', ...
    'Position', [200, 200, 400, 130]);
hEdit1 = uieditfield(hFig, 'numeric', 'Value', pi, ...
    'Position', [10, 70, 280, 50], ...
    'Limits', [-10, 10], ...
    'ValueDisplayFormat', '%.2e V/m', ...
    'FontSize', 30);
hEdit2 = uieditfield(hFig, 'text', ...
    'Value', 'log10(abs(X))', ...
    'HorizontalAlignment', 'center', ...
    'FontSize', 30, 'Position', [10, 10, 280, 50], ...
    'FontColor', 'r', 'BackgroundColor', 'k');
% value regardless of display format:
setValue = hEdit1.Value
```





Edit Field Creation

- ▶ In panel titled **'Results'** create numeric edit box with Position property as [120, 200, 90, 22] px.
- ▶ Set proper Backgroundcolor, FontWeight, ... (up to you).



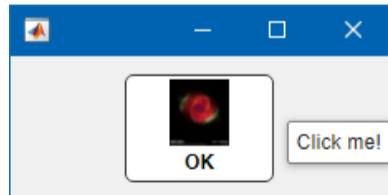


UI Components – uibutton

- ▶ Create push button or state button component.
- ▶ The state button can be done by style argument as `uibutton('state')`.
 - ▶ It is not the same as `uitogglebutton`, can't be managed by `uibuttongroup`!
- ▶ Icon can be defined also as m-by-n-by-3 truecolor image array.
- ▶ Properties can be aggregated into a cell array and used as comma-separated list by `{:}` as function inputs (holds generally).

```
buttProperties = {'Text', 'OK', ...
  'Position', [70, 10, 90, 65], ...
  'Icon', 'ngc6543a.jpg', ...
  'IconAlignment', 'top', ...
  'FontWeight', 'bold', ...
  'ToolTip', 'Click me!'};
hFig = uifigure('Position', [200, 200, 230, 85]);
hButt = uibutton(hFig, buttProperties{:});
```

```
% Icon as a random color image
hButt.Icon = uint8(rand(20, 20, 3)*255);
```





Push Button Creation

- ▶ Create push button labelled 'Close' positioned in figure with Position property [130, 20, 90, 30] px.
- ▶ Set proper Backgroundcolor, FontWeight, ... (up to you).





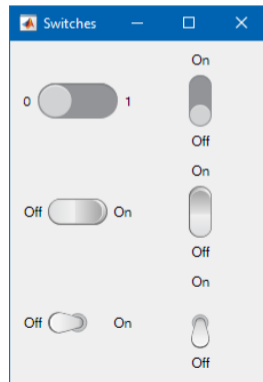
UI Components – uiswitch

- ▶ Create slider switch, rocker switch, or toggle switch component.
- ▶ All switches can be oriented vertically or horizontally (Orientation property).
- ▶ Labels can be modified (Items property).

```
hFig = uifigure('Name', 'Switches', ...
    'Position', [200, 200, 225, 300]);
hGrid = uigridlayout(hFig, ...
    'ColumnWidth', {'1x', '1x'}, ...
    'RowHeight', {'1x', '1x', '1x'});
hSwitch1 = uiswitch(hGrid, 'slider', ...
    'Orientation', 'horizontal', ...
    'Items', {'0', '1'}, 'ItemsData', {false, true});
uiswitch(hGrid, 'slider', 'Orientation', 'vertical');

uiswitch(hGrid, 'rocker', 'Orientation', 'horizontal');
uiswitch(hGrid, 'rocker', 'Orientation', 'vertical');

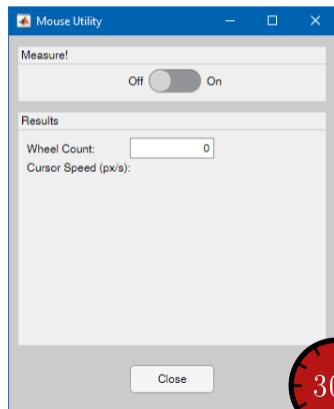
uiswitch(hGrid, 'toggle', 'Orientation', 'horizontal');
uiswitch(hGrid, 'toggle', 'Orientation', 'vertical');
% Value as corresponding 'ItemsData' element
logValue = hSwitch1.Value
```





Switch Creation

- ▶ Create switch positioned in panel titled 'Measure!' with Position property [140, 10, 100, 25] px, Items as {'Off', 'On'} and ItemsData as [false, true].
- ▶ Set proper Style, Items, FontWeight, ... (up to you).



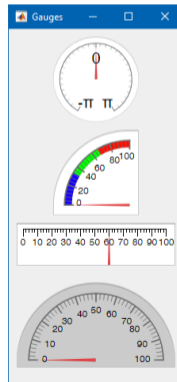


UI Components – uigauge

- ▶ Create gauge component for indicating numerical value.
- ▶ Several *style* options `uigauge(style)`:
 - ▶ 'circular', 'linear', 'ninetydegree', 'semicircular'.
 - ▶ Different orientation possibilities ('counterclockwise', 'north', 'vertical', ...).

```

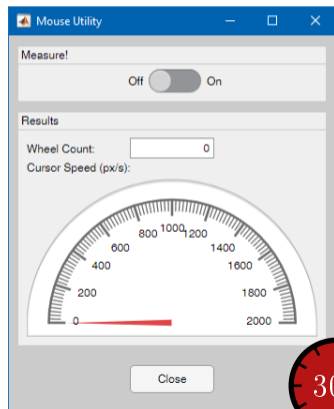
hFig = uifigure('Name', 'Gauges', ...
    'Position', [200, 200, 220, 450]);
hGrid = uigridlayout(hFig, ...
    'ColumnWidth', {'1x'}, ...
    'RowHeight', {'1x', '1x', '0.5x', '1.2x'});
hGauge1 = uigauge(hGrid, 'circular', ...
    'Limits', [-pi, pi], ...
    'ScaleDirection', 'counterclockwise', ...
    'MajorTicks', -pi:pi:pi, ...
    'MajorTickLabels', ...
    {char(960), '0', ['-'], char(960)}}, ...
    'FontSize', 20);
hGauge2 = uigauge(hGrid, 'ninetydegree', ...
    'ScaleColors', {'b', 'g', 'r'});
hGauge3 = uigauge(hGrid, 'linear', 'Value', 60);
hGauge4 = uigauge(hGrid, 'semicircular', ...
    'BackgroundColor', 0.8*ones(1, 3));
  
```





Gauge Creation

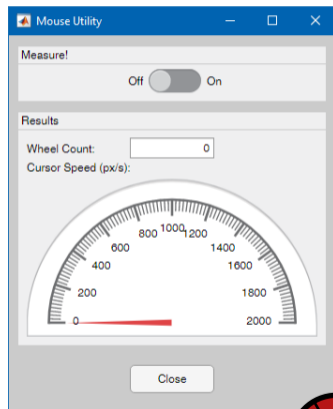
- ▶ Create gauge with proper style, Limits [0 2e3] and Position [10, 10, 310, 170] px in panel titled 'Results'.
- ▶ You can freely change display settings.





Save Static GUI

- ▶ Save the creation of static GUI as a script or as a function for later use (the next lecture).

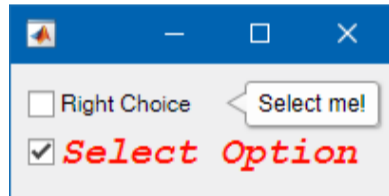




UI Components – uicontrol

- ▶ UI component for indicating the state of a preference or option.
- ▶ Can be children of any container component.
- ▶ Always has the same background color as BackgroundColor property of parent container.

```
hFig = uifigure('Position', ...
    [200, 200, 200, 70]);
hCheck1 = uicontrol(hFig, ...
    'Text', 'Right Choice', ...
    'Position', [10, 40, 100, 20], ...
    'Tooltip', 'Select me!');
hCheck2 = uicontrol(hFig, ...
    'Value', true, ...
    'Text', 'Select Option', ...
    'FontName', 'Courier New', ...
    'FontSize', 20, ...
    'FontWeight', 'bold', ...
    'FontAngle', 'italic', ...
    'FontColor', 'r', ...
    'Position', [10, 10, 180, 30]);
```



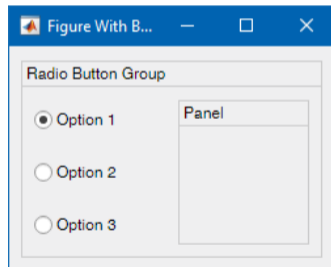


UI Components – uiradiobutton

- ▶ Create radio button within a button group (uibuttongroup).
- ▶ Only one radio button can be selected.
- ▶ Always has the same background color as BackgroundColor property of parent container.

```
hFig = uifigure('Name', 'Figure with Button Group', ...
    'Position', [200 200 250 170]);
hGroup = uibuttongroup(hFig, 'Position', [10, 10, 230, 150], ...
    'Title', 'Radio Button Group');
hButt1 = uiradiobutton(hGroup, 'Text', 'Option 1', ...
    'Position', [10 90 100 30]);
hButt2 = uiradiobutton(hGroup, 'Text', 'Option 2', ...
    'Position', [10 50 100 30]);
hButt3 = uiradiobutton(hGroup, 'Text', 'Option 3', ...
    'Position', [10 10 100 30]);
hPanel = uipanel(hGroup, 'Title', 'Panel', ...
    'Position', [120, 10, 100, 110]);
```

```
% Button selection programmatically:
hButt2.Value = true;
% or
hGroup.SelectedObject = hButt2;
```



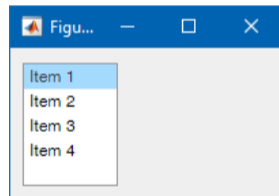


UI Components – uilistbox

- ▶ UI component for displaying items in a list.
- ▶ It is possible to select more options using Ctrl and Shift keys.
- ▶ It is possible to get the selected items via Value property.
 - ▶ Values are elements from ItemsData vector, if defined.

```
hFig = uifigure('Name', 'Figure with List Box', ...
    'Position', [200 200 200 110]);
hList = uilistbox(hFig, 'Position', [10, 10, 70, 90], ...
    'Items', {'Item 1', 'Item 2', 'Item 3', 'Item 4'}, ...
    'Multiselect', true);
```

```
% get selected items from Items vector
selItems = hList.Value
% get selected items from ItemsData vector
hList.ItemsData = 1:4;
selItems = hList.Value
```



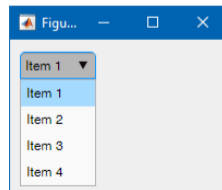
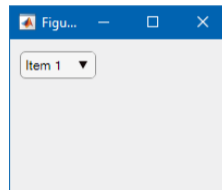


UI Components – uiddropdown

- ▶ UI component that enables the user to select an option.
- ▶ Only one option can be selected.

```
hFig = uifigure('Name', 'Figure with Drop Down', ...
    'Position', [200 200 200 140]);
hList = uiddropdown(hFig, 'Position', [10, 105, 70, 25], ...
    'Items', {'Item 1', 'Item 2', 'Item 3', 'Item 4'});
```

```
% get selected item from Items vector
selItems = hList.Value
% get selected item from ItemsData vector
hList.ItemsData = 1:4;
selItems = hList.Value
```



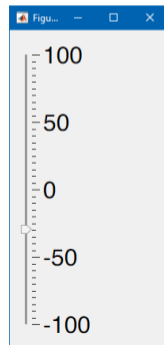


UI Components – uislider

- ▶ UI component that allows the user to select a value along a continuum.
- ▶ Orientation can be 'horizontal' or 'vertical'.
 - ▶ Height, or width, respectively, can't be changed and is always 3 px.
 - ▶ Position property defines the slider's position excluding tick marks and tick labels.
 - ▶ OuterPosition property (read-only) is a position of the whole slider.

```
hFig = uifigure('Name', 'Figure with Slider', ...
    'Position', [200, 200, 200, 400], ...
    'Scrollable', true, 'AutoSizeChildren', false);
hSlider = uislider(hFig, 'Limits', [-100, 100], ...
    'Orientation', 'vertical', ...
    'Position', [20, 20, 3, 360], ...
    'FontSize', 30, ...
    'MajorTicks', -100:50:100, ...
    'Value', -30);
```

```
hSlider.OuterPosition
```

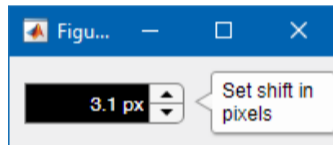




UI Components – uispinner

- ▶ UI component for selecting a numeric value from a finite set.
- ▶ Offers rounding, limits, step and format (see >> doc [Spinner](#)).
- ▶ Value property contains a set value regardless of display format.

```
hFig = uifigure('Name', 'Figure with Spinner', ...
    'Position', [200, 200, 200, 55]);
hSpinner = uispinner(hFig, 'Value', pi, ...
    'Limits', [-10, 10], ...
    'Step', 5, ...
    'ValueDisplayFormat', '%.1f px', ...
    'FontColor', 'white', ...
    'BackgroundColor', [0 0 0], ...
    'ToolTip', 'Set shift in pixels', ...
    'Position', [10, 15, 100, 25]);
value = hSpinner.Value
```

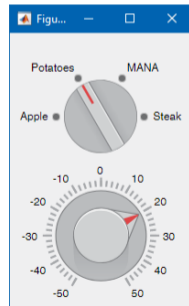




UI Components – uiknob

- ▶ UI components representing instrument control knobs that the user can adjust to control a value.
- ▶ `uiknob()` and `uiknob('continuous')` creates knob with continuous values.
- ▶ `uiknob('discrete')` creates knob with discrete values.
 - ▶ Value, Items and ItemsData properties works the same way as for, *e.g.*, `uidropdown`.

```
hFig = uifigure('Name', 'Figure with Knobs', ...
    'Position', [200, 200, 200, 300]);
hKnob1 = uiknob(hFig, 'discrete', 'Items', ...
    {'Apple', 'Potatoes', 'MANA', 'Steak'}, ...
    'ItemsData', 1:4, ...
    'Value', 2, ...
    'Position', [60, 170, 80, 140]);
hKnob2 = uiknob(hFig, ...
    'Position', [50, 30, 100, 140], ...
    'Value', 20, ...
    'Limits', [-50, 50]);
```





UI Components – uitable

- ▶ Create table user interface component.
- ▶ Table offers many of formatting and interactive options (see >> doc `uitable`).

```

hFig = uifigure('Name', 'Figure with Table', ...
    'Position', [200, 200, 330, 330]);
% simple table:
hTab1 = uitable(hFig, 'Data', magic(10), ...
    'Position', [10 110 310 210]);
% advanced table:
hTab2 = uitable(hFig, 'Data', ...
    {'Smith', 38, 'Choose', true;
    'Johnson', 43, 'Choose', false;
    'Williams', 38, 'Choose', false}, ...
    'ColumnName', ...
    {'Name', 'Age', 'Personality', ...
    'Authorized'}, ...
    'ColumnFormat', ({[], [], {'Analyst', ...
    'Diplomat', 'Explorer', 'Idiot'}, []}), ...
    'ColumnWidth', {50, 50, 90, 70}, ...
    'Position', [10 10 310 90], ...
    'ColumnEditable', true(1, 4));
  
```

	1	2	3	4
1	92	99	1	
2	98	80	7	
3	4	81	88	
4	85	87	19	
5	86	93	25	
6	17	24	76	
7	23	5	82	
8	79	6	13	
9	10	12	94	
10	

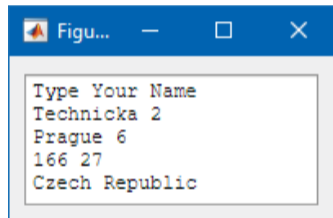
	Name	Age	Personality	Authorized
1	Smith	38	Choose	<input checked="" type="checkbox"/>
2	Johnson	43	Choose	<input type="checkbox"/>
3	Williams	38	Choose	<input type="checkbox"/>



UI Components – uitextarea

- ▶ UI component for entering multiple lines of text.
- ▶ If the text does not fit into the width of the text area, the text is wrapped.

```
hFig = uifigure('Name', 'Figure with Text Area', ...  
    'Position', [200, 200, 200, 100]);  
hText = uitextarea(hFig, 'Value', ...  
    ["Type Your Name"; "Technicka 2"; "Prague 6";  
    "166 27"; "Czech Republic"], ...  
    'FontName', 'Courier New', ...  
    'FontColor', [0.1, 0.1, 1], ...  
    'Position', [10, 10, 180, 80]);  
% get entered address  
yourAddress = hText.Value
```





UI Components – uilamp

- ▶ Lamp is UI component that indicates state using color.

```
hFig = uifigure('Name', 'Figure with Lamps', ...  
    'Position', [200, 200, 200, 560], ...  
    'Color', 'k');  
hLamp1 = uilamp(hFig, 'Color', [1 0 0], ...  
    'Position', [20, 380, 160, 160]);  
hLamp2 = uilamp(hFig, 'Color', [1 0.5 0.25], ...  
    'Position', [20, 200, 160, 160]);  
hLamp3 = uilamp(hFig, 'Color', [0 1 0], ...  
    'Position', [20, 20, 160, 160]);  
% Turn the light off  
hLamp1.Enable = false;
```





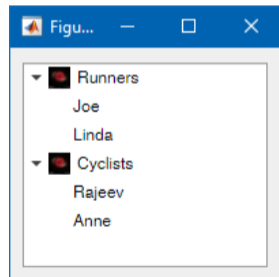
UI Components – uitree

- ▶ Create tree component capable of showing data hierarchy.
- ▶ Individual items in tree are objects created by uitreenode.
 - ▶ The nodes has to be children of uitree object.
 - ▶ Items can have Icon defined as path to a figure.

```
hFig = uifigure('Name', 'Figure with Tree', ...
    'Position', [200, 200, 200, 170]);
hTree = uitree(hFig, 'Position', [10 10 180 150]);

category1 = uitreenode(hTree, 'Text', 'Runners', ...
    'Icon', 'ngc6543a.jpg');
p1 = uitreenode(category1, 'Text', 'Joe');
p2 = uitreenode(category1, 'Text', 'Linda');

category2 = uitreenode(hTree, 'Text', 'Cyclists', ...
    'Icon', 'ngc6543a.jpg');
p3 = uitreenode(category2, 'Text', 'Rajeev');
p4 = uitreenode(category2, 'Text', 'Anne');
% Expand the tree
expand(hTree);
```

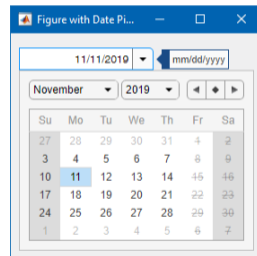
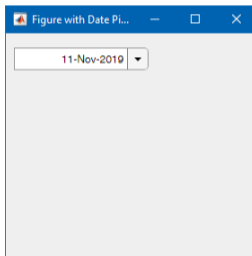




UI Components – uideatepicker

- ▶ Date pickers allow users to select dates from an interactive calendar.
- ▶ Accepts and returns datetime objects (see >> doc `datetime`).
- ▶ Some dates and days in week can be disabled.

```
hFig = uifigure('Name', ...
    'Figure with Date Picker', ...
    'Position', [200, 200, 280, 250]);
hDate = uideatepicker(hFig, ...
    'Value', datetime('today'), ...
    'Position', [10, 210, 150, 25], ...
    'DisabledDaysOfWeek', [6, 7]);
% get selected date
setDate = hDate.Value
```

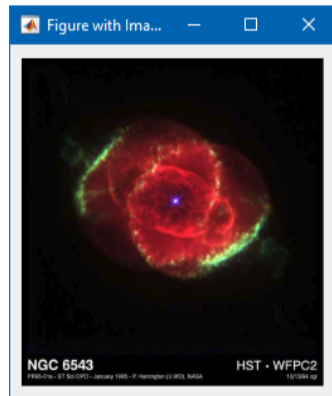




UI Components – uimage

- ▶ UI component that allow you to display an picture in your app.
- ▶ Can be children of any container component.

```
hFig = ufigure('Name', ...  
    'Figure with Image', ...  
    'Position', [200, 200, 260, 280]);  
hImage = uimage(hFig, ...  
    'ImageSource', 'ngc6543a.jpg', ...  
    'Position', [10, 10, 240, 260]);
```



GUI

UI Components – uitoolbar, uipushtool, uitoggletool



- ▶ A toolbar is a container for a horizontal list of buttons at the top of a figure window.
- ▶ Can contain push buttons and toggle buttons.
- ▶ Does not provide management of the toggle tools.



```
iconFolder = fullfile(matlabroot, 'toolbox', 'matlab', 'icons');
hFig = uifigure('Position', [200, 200, 180, 40]);
hTool = uitoolbar(hFig);
hPushNew = uipushtool(hTool, 'Icon', fullfile(iconFolder, 'file_new.png'), ...
    'ToolTip', 'Open File');
hPushSave = uipushtool(hTool, 'Icon', fullfile(iconFolder, 'file_save.png'), ...
    'ToolTip', 'Save File');
hPushOpen = uipushtool(hTool, 'Icon', fullfile(iconFolder, 'file_open.png'), ...
    'ToolTip', 'Open File');
hToggPlay = uitoggletool(hTool, 'Icon', fullfile(iconFolder, 'greenarrowicon.gif'), ...
    'State', true, 'Separator', true, 'Tooltip', 'Play');
hToggZoom = uitoggletool(hTool, 'Icon', fullfile(iconFolder, 'view_zoom_in.gif'), ...
    'ToolTip', 'Zoom');
```

```
% Set state of toggletool programmatically
hToggZoom.State = true;
```



Enabling UI Components

- ▶ Majority of UI components have Enable property which defines its operational state.
- ▶ Possible values are 'on' and 'off', or true and false.
- ▶ When the component is disabled it doesn't respond to events.
- ▶ It is also possible to let the component disappear using Visible property.

The screenshot displays a window titled "UI Components" containing a grid of 18 different MATLAB GUI components:

- uiaxes:** A 2D plot with axes ranging from 0 to 1.
- uidatepicker:** A date selection field showing "mm/dd/yyyy".
- uimage:** A display area showing the same heraldic crest seen in the top right of the slide.
- uitable:** A table with columns "Name", "Age", and "Authorized".

	Name	Age	Authorized
1	Smith	38	<input checked="" type="checkbox"/>
2	Johnson	43	<input type="checkbox"/>
3	Williams	36	<input type="checkbox"/>
- uitree:** A tree view with expandable nodes: "Runners" (Joe, Linda) and "Cyclists" (Rajeev, Anne).
- uigauge:** A circular gauge with a needle pointing to approximately 18 on a scale from 0 to 100.
- uibutton:** A simple rectangular button labeled "ubutton".
- uicheckbox:** A checkbox labeled "Check Box".
- uidropdown:** A dropdown menu currently showing "First Item".
- uiditfield:** A text input field containing the number "3.14".
- uilabel:** A text label that reads "UI Label".
- uistack:** A list box containing three items: "First item", "Second item", and "Third item".
- uislider:** A horizontal slider with a range from 0 to 100.
- uispinner:** A spinner control showing the value "20".
- uitextarea:** A text area containing the text: "The S-Parameters are a linear model of the DUT."
- uiknob:** A rotary knob with a scale from 0 to 100.
- uilamp:** A lamp indicator consisting of a green circle.
- uiswitch:** A toggle switch currently in the "Off" position.



Preallocation of UI Components

- ▶ Function `gobjects` initialize array of graphical objects.
 - ▶ It has the same usage as function `zeros`, `ones`, ... for numeric values.

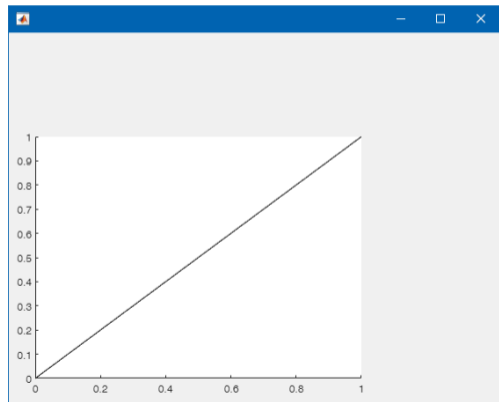
```
>> hObjs = gobjects(3, 1)
```

```
hObjs =
```

```
3x1 GraphicsPlaceholder array:
```

```
GraphicsPlaceholder  
GraphicsPlaceholder  
GraphicsPlaceholder
```

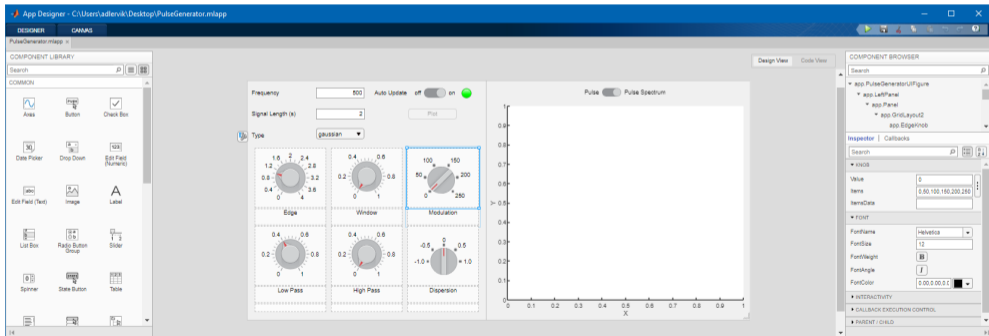
```
hObjs(1) = uifigure();  
hObjs(2) = uiaxes(hObjs(1));  
hObjs(3) = line(hObjs(2));
```





App Designer

- ▶ App Designer is a development environment that provides an interactive design of applications using UI components.
 - ▶ appdesigner
 - ▶ Designed application using object-oriented programming (OOP).
 - ▶ It is possible to use App Designer for semester projects.



Exercises



Exercise I.a

- ▶ Create a function with two inputs and one output:

```
function logicState = createButtons(nRows, nColumns)
% function generating GUI with state buttons
```

- ▶ Function creates figure with state buttons arranged in matrix $nRows \times nColumns$.
- ▶ After clicking on state buttons and close window function returns a matrix of logical values representing state of state buttons.

```
>> logicState = createButtons(3, 5)
```

```
logicState =
```

```
3x5 logical array
```

```
1 0 1 0 1
0 1 0 1 0
1 0 1 0 1
```





Exercise I.b

```
function logicState = createButtons(nRows, nColumns)
% function generating GUI with state buttons
screenSize = get(getroot, 'ScreenSize'); % size of screen

figSize = [500, 500]; % size of generated figure
figPos = [(screenSize(3)-figSize(1))/2, ...
          (screenSize(4)-figSize(2))/2]; % position of centered figure

hFig = uifigure('Name', 'Figure', 'Position', [figPos, figSize]);
hGrid = uigridlayout(hFig, 'RowHeight', repmat({'lx'}, 1, nRows), ...
                    'ColumnWidth', repmat({'lx'}, 1, nColumns));
% allocatinon of state buttons
hStateButt = gobjects(nRows, nColumns);
% creating state buttons (automatically placed column-wise into grid)
for iRow = 1:nRows
    for iCol = 1:nColumns
        hStateButt(iRow, iCol) = uibutton(hGrid, 'state', ...
            'Text', sprintf('%d, %d', iRow, iCol));
    end
end
% set CloseRequestFcn to figure
hFig.CloseRequestFcn = @getLogicalState;
uiwait(hFig); % wait until figure exists

function getLogicalState(hFig, ~)
    valuesInRow = [hStateButt.Value]; % states of buttons
    logicState = reshape(valuesInRow, size(hStateButt));
    hFig.delete();
end
end
```

Questions?

B0B17MTB, BE0B17MTB – MATLAB
matlab@fel.cvut.cz

April 19, 2023
Summer semester 2022/23

This document has been created as a part of B(E)0B17MTB course.
Apart from educational purposes at CTU in Prague, this document may be reproduced, stored, or transmitted only with the prior permission of the authors.

Acknowledgement: Filip Kozák, Pavel Valtr, Michal Mašek, and Vít Losenický.