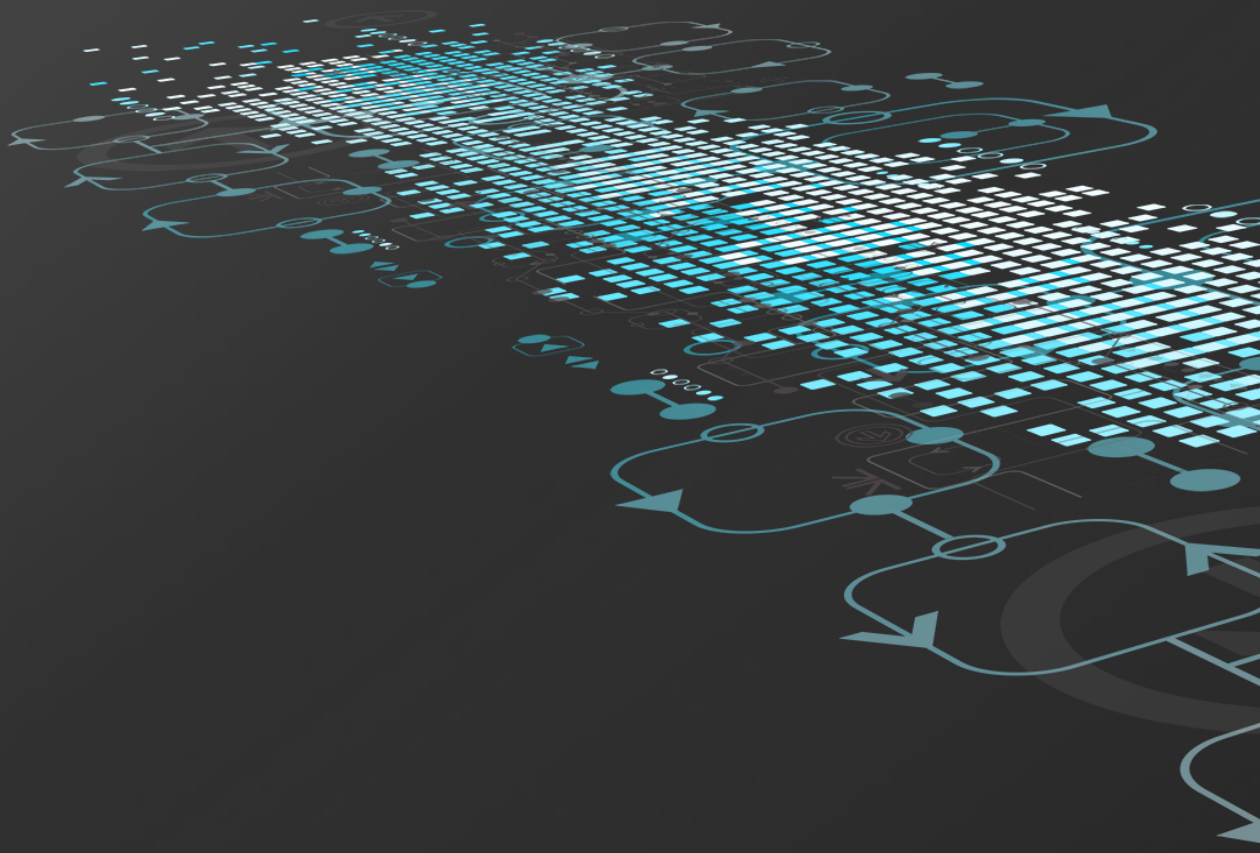


ccTools

Custom components tools

ccTools

In 2020, with the Covid pandemic and lockdown measures, I returned to working with Matlab after a 12-year hiatus. I had last used Matlab in 2008 when I completed my master's degree.

I must admit that I was impressed by how App Designer simplified the process of building apps. Since then, I have been developing engineering applications, particularly in the field of sensor control (such as spectrum analyzers, receivers, and antenna control units), as well as data processing generated by these sensors.

Over the past three years, the Custom Components Tools, or simply ccTools, emerged as a way to overcome some limitations of the MATLAB built-in components. For example, it is still not possible to edit the font of the header of the uitable, and the modal windows - uialert and uiprogressdlg - have minimal customization options, limited to the icon.

I have managed to organize the bits and pieces of code I have developed here and there. As a result, I am now releasing ccTools with the following features:

(a) A filterable table class.

(b) A push button class.

(c) Functions that create modal windows - MessageBox and ProgressDialog.

(d) Functions that allow customization of certain aspects of the MATLAB built-in components.

i

The use of ccTools requires adding its root folder to the MATLAB path.

Metadata files have been generated for the two new classes - the filterable table and the push button - enabling drag and drop of components, as well as the configuration of their public properties, directly in App Designer.

The ccTools was developed on the Windows platform and tested on three MATLAB releases - R2021b, R2022b, and R2023a. It is likely to work on R2022a as well, but there are no guarantees that all features will be operational in older versions.

Lastly, this work would not have been possible without Robyn Jackey's contribution in writing the “Widgets Toolbox - MATLAB App Designer Components”. I have learned a lot from Robyn’s code!

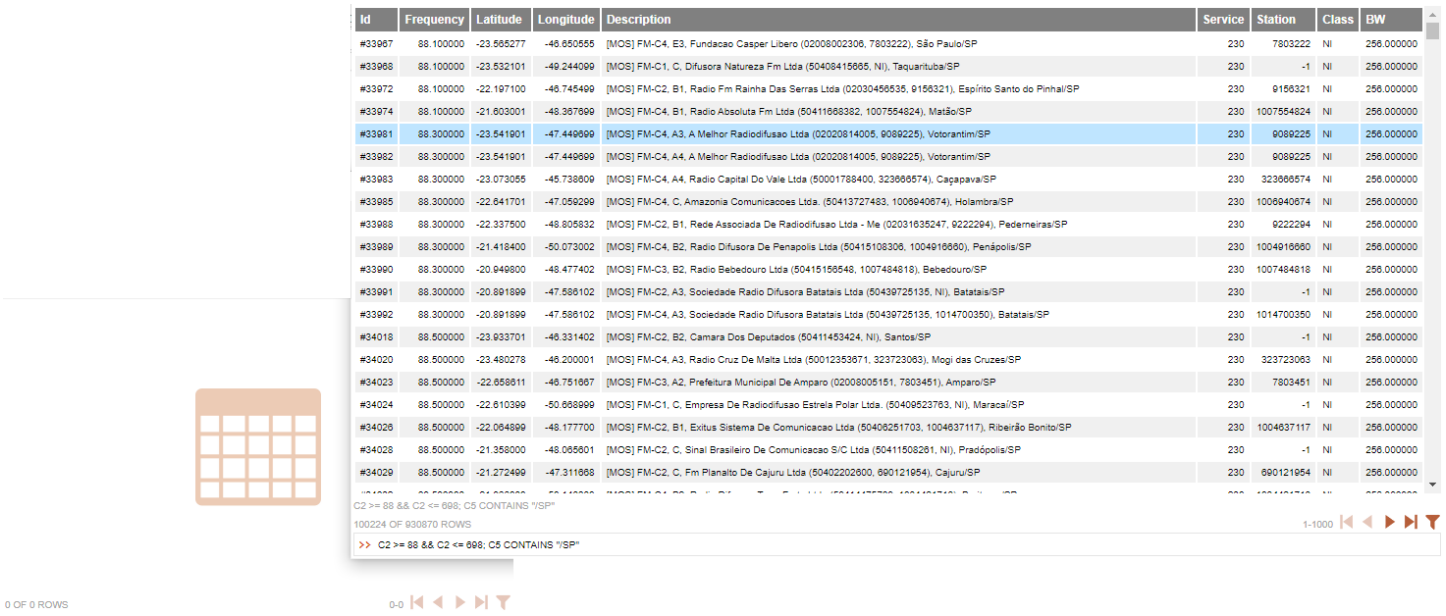
ccTools.Table

Create filterable table.

When loading a table into the object, through its "Data" property, this table can be composed only for textual values ("cell", "string", and "categorical"), "datetime" values, "logical" values, and numerical values, but the following conversions are applied before storage:

- "cell", "categorical" and "datetime" values are converted to "string".
- "logical" values are converted to "double".
- "&&", "||", and ";" are protected keywords and cannot be part of the textual values in the table. For this reason, these values are replaced by "&", "|", and ",", respectively.
- "\n" (newline) are replaced by"
", creating breakline effect.

In other words, all textual values are stored as "string", and numerical values retain their data type, except for "logical" values, which are converted to "double". Therefore, programmatically changes to this table after its loading must respect these new data types.



Syntax

```
tb = ccTools.Table
tb = ccTools.Table(parent)
tb = ccTools.Table(__, Name, Value)
```

Main properties

(argument name, type, and possible values or default value)

- **Data** - *table*
- **Selection** - *double*

Others properties

- **ColumnName** - *cell array* - {'auto'}
- **ColumnEditable** - *numerical array* - 0
- **ColumnWidth** - *cell array* - if not default value, each cell must be 'auto' or '%dpx' pattern (such as '60px' or '110px') - {'auto'}
- **ColumnAlign** - cell array - if not default value, each cell must be member of {'left', 'center', 'right'} - {'auto'}
- **ColumnPrecision** - *cell array* - if not default value, each cell must be member of {'%s', '%d', '%i', '%.0f', '%.1f' ... '%.99f'} - {'auto'}
- **FilterToolbar** - *logical* - true | false
- **FontFamily** - *enumeration* - Helvetica
- **hFontSize** - *numeric* - 12
- **hFontWeight** - *enumeration* - "normal" | "bold"
- **hFontAlign** - *enumeration* - "left" | "center" | "right"
- **hFontColor** - *scalar text (char | string)* - valid hexa color, or CSS color name - "#FF0000" | "red" and so on...
- **hCapitalLetter** - *logical* - true | false
- **hClickable** - *logical* - true | false
- **bFontFamily** - *enumeration* - Helvetica
- **bFontSize** - *numeric* - 10
- **bFontWeight** - *enumeration* - "normal" | "bold"
- **bFontColor** - *scalar text (char | string)* - valid hexa color, or CSS color name
- **bStripingColor** - *scalar text (char | string)* - valid hexa color, or CSS color name
- **bHoverColor** - *scalar text (char | string)* - valid hexa color, or CSS color name
- **bSelectedColor** - *scalar text (char | string)* - valid hexa color, or CSS color name
- **BackgroundColor** - *numeric array or scalar text (char | string)* - valid color - [1 0 0] | "#FF0000" | "red" and so on...

"ColumnName" property determines the names of the table's columns. The characters "|" and "
", or a cell array of characters, can be used to create a breakline effect. "hFontSize" and "bFontSize" in *pixels*. "hFontFamily", "hFontSize", "hFontWeight", "hFontAlign" and "hFontColor" are properties related to the header of the table. Finally, "bFontFamily", "bFontSize", "bFontWeight" and "bFontColor" are properties related to the body of the table.

Events

- SelectionChanged
- CellEdited
- DataFiltered

Filtering data

Filtering, sorting, and adjusting data precision operations can be applied to the table interactively.

The general model for a filtering request is: '1st Sentence; 2nd Sentence; ...; n-th Sentence'. Logically, the "&&" (AND) operator is applied to the request, resulting in the intersection of individual filters.

On the other hand, the general model for a specific sentence is '1st Operation && 2nd Operation && ... && n-th Operation' or '1st Operation || 2nd Operation || ... || n-th Operation'. In other words, within a specific sentence, the "&&" (AND) or "||" (OR) operator can be applied.

Below are the allowed operations.

Numeric Columns	Cn < numericValue Cn <= numericValue Cn == numericValue Cn != numericValue Cn >= numericValue Cn > numericValue Cn sorted Cn sorted up Cn sorted down Cn precision "%.1f"
Textual Columns	Cn contains "stringValue" Cn !contains "stringValue" Cn isequal "stringValue" Cn !isequal "stringValue" Cn sorted Cn sorted up Cn sorted down
Logical Operators	&& ||
Sentences Separators	;

The term Cn refers to the n-th column of the table. So, C1 corresponds to the first column, C2 to the second column, and so on. Finnaly, when applying a filter to a textual column, it is important to enclose the string being evaluated within double quotation marks.

Examples of valid sentences:

- C1 >= 100 && C1 <= 1000; C2 contains "stringValue"; C3 sorted
- C1 == 87.9 || C1 == 104.9; C2 !isequal "stringValue"
- C1 precision "%.0f"; C2 precision "%.3f"

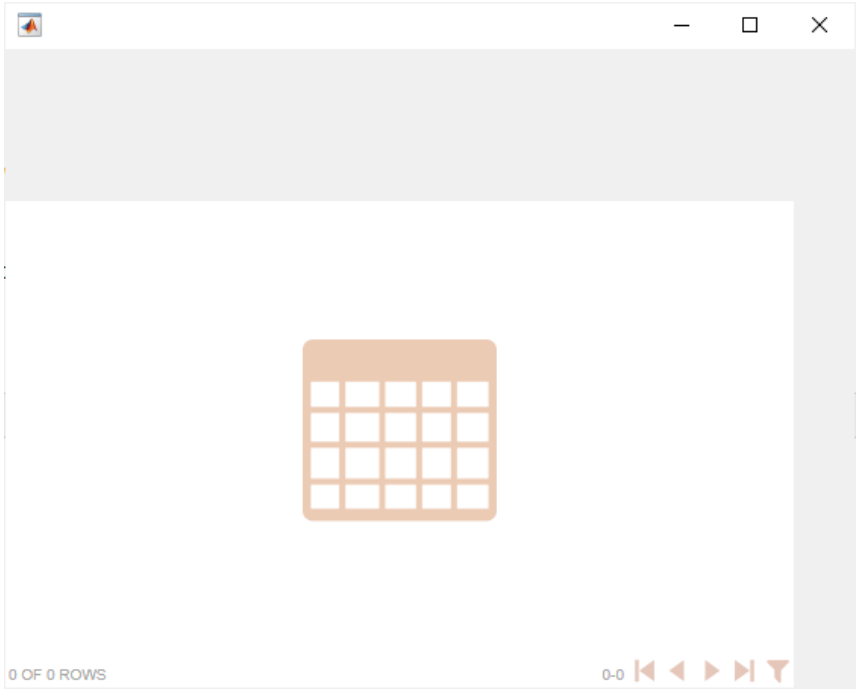
Example

```
tb = ccTools.Table
```

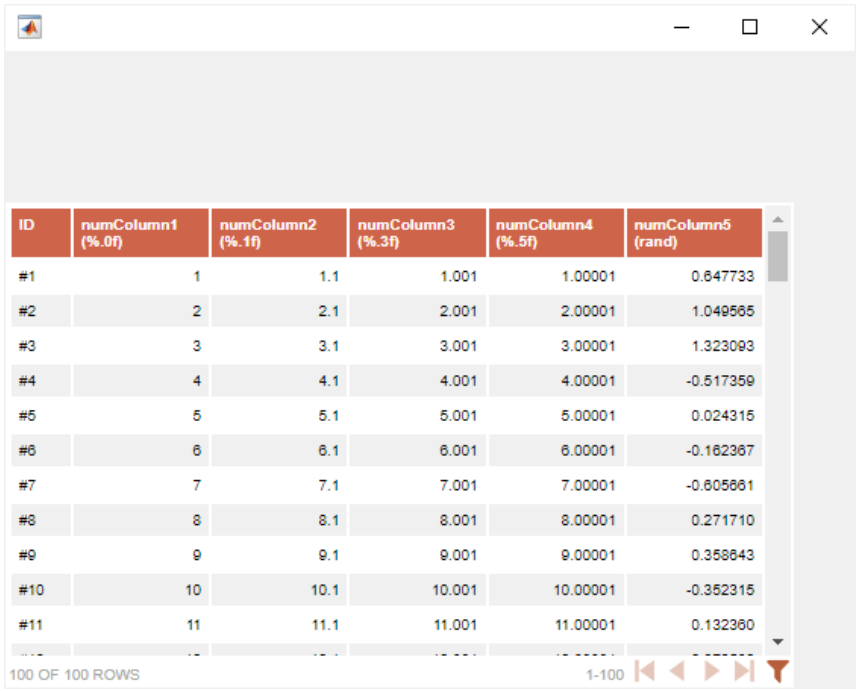
```
  tb =
    Table with properties:

                    Data: [0×0 table]
               Selection: 0
                    Cell: [0×0 struct]
              ColumnName: {'auto'}
          ColumnEditable: 0
             ColumnWidth: {'auto'}
             ColumnAlign: {'auto'}
        ColumnPrecision: {'auto'}
           FilterToolbar: 1
           FilteredIndex: [0×1 double]
             hFontFamily: Helvetica
               hFontSize: 12
             hFontWeight: bold
              hFontAlign: left
              hFontColor: 'white'
          hCapitalLetter: 0
              hClickable: 1
             bFontFamily: Helvetica
               bFontSize: 10
             bFontWeight: normal
```

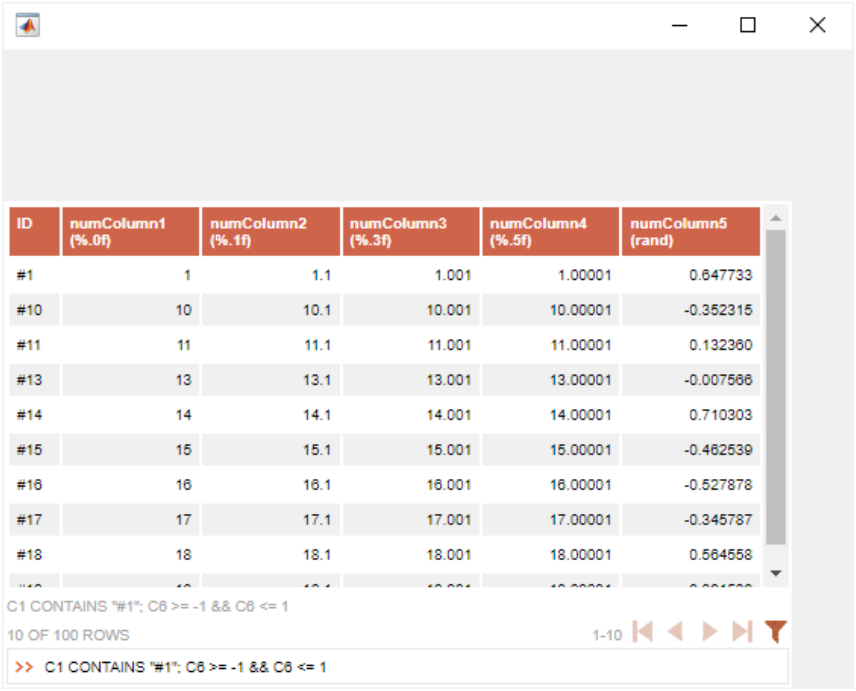
```
          bFontColor: 'black'
     bStripingColor: '#f0f0f0'
        bHoverColor: '#bfe5ff'
     bSelectedColor: '#bfe5ff'
SelectionChangedFcn: ''
       CellEditedFcn: ''
    DataFilteredFcn: ''
           Position: [1 1 520 320]
```




```
set(tb, Data = table("#"+string((1:100)'), (1:100)', (1:100)'+.1, (1:100)'+.001, (1:100)'+.00001, randn(100,1)), ...
        ColumnName = {'ID', 'numColumn1<br>(%.0f)', {'numColumn2' '(%.1f)'}, {'numColumn3' '(%.3f)'},            ...
                      {'numColumn4' '(%.5f)'}, {'numColumn5' '(rand)'}},                                          ...
        ColumnEditable = [1 1 1 1 1 1], BackgroundColor = "#ce644a", hFontSize = 10);
```



Filtering data...



ID	numColumn1 (%.0f)	numColumn2 (%.1f)	numColumn3 (%.3f)	numColumn4 (%.5f)	numColumn5 (rand)
#1	1	1.1	1.001	1.00001	0.647733
#10	10	10.1	10.001	10.00001	-0.352315
#11	11	11.1	11.001	11.00001	0.132360
#13	13	13.1	13.001	13.00001	-0.007566
#14	14	14.1	14.001	14.00001	0.710303
#15	15	15.1	15.001	15.00001	-0.462539
#16	16	16.1	16.001	16.00001	-0.527878
#17	17	17.1	17.001	17.00001	-0.345787
#18	18	18.1	18.001	18.00001	0.564558

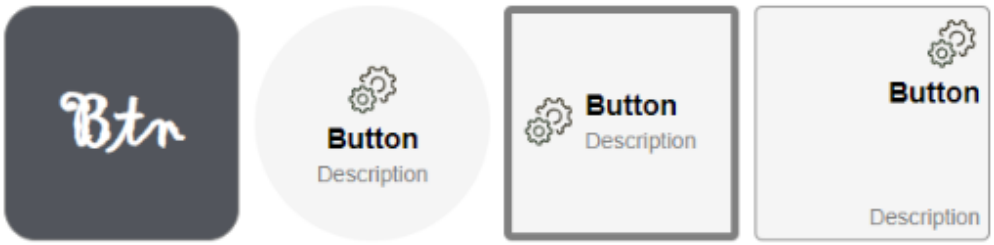
C1 CONTAINS "#1"; C6 >= -1 && C6 <= 1

10 OF 100 ROWS 1-10

>> C1 CONTAINS "#1"; C6 >= -1 && C6 <= 1

ccTools.Button

Create push button.



Syntax

```
btn = ccTools.Button
btn = ccTools.Button(parent)
btn = ccTools.Button(__, Name, Value)
```

Main properties

(argument name, type, and possible values or default value)

- **Text** - *scalar text (char | string)* - "Button"
- **Description** - *scalar text (char | string)* - "Description"
- **Icon** - *scalar text (char | string)* - ""

Others properties

- **Model** - *enumeration* - "IconPlusText" | "Icon" | "Text"
- **IconAlignment**- *enumeration* - "top" | "left" | "right"
- **HorizontalAlign** - *enumeration* - "left" | "center" | "right"
- **VerticalAlign** - *enumeration* - "top" | "bottom" | "left" | "right"
- **ColumnSpacing** - *numeric* - 5
- **RowSpacing** - *numeric* - 0
- **RowTextSpacing** - *numeric* - 0
- **BorderWidth** - *numeric* - 1
- **BorderColor** - *scalar text (char | string)* - valid hexa color, or CSS color name - "#FF0000" | "red" and so on...
- **BorderRadius** - *scalar text (char | string)* - "5px", "10px", "25%", "50%" (rounded button) and so on...
- **BorderPadding** - *numeric* - 5
- **IconWidth** - *numeric* - 18
- **IconHeight** - *numeric* - 18
- **FontFamily** - *enumeration* - Helvetica
- **tFontWeight** - *enumeration* - "normal" | "bold"
- **tFontSize** - *numeric* - 12
- **tFontColor** - *scalar text (char | string)* - valid hexa color, or CSS color name
- **dFontSize** - *numeric* - 10
- **dFontColor** - *scalar text (char | string)* - valid hexa color, or CSS color name
- **Enable** - *logical* - true | false
- **DisabledOpacity**: *numeric* - float between .1 and .9 - .35
- **BackgroundColor** - *numeric array or scalar text (char | string)* - valid color - [1 0 0] | "#FF0000" | "red" and so on...

"Icon" is the fullpath of the image, or just its name if the image is in the MATLAB search path. "ColumnSpacing", "RowSpacing", "RowTextSpacing", "BorderWidth", "BorderPadding", "IconWidth", "IconHeight", "tFontSize" and "dFontSize" in *pixels*. "RowTextSpacing" defines the vertical space between "Text" and "Description". "tFontWeight", "tFontSize" and "tFontColor" are properties related to "Text". Finally, "dFontSize" and "dFontColor" are properties related to "Description".

Event

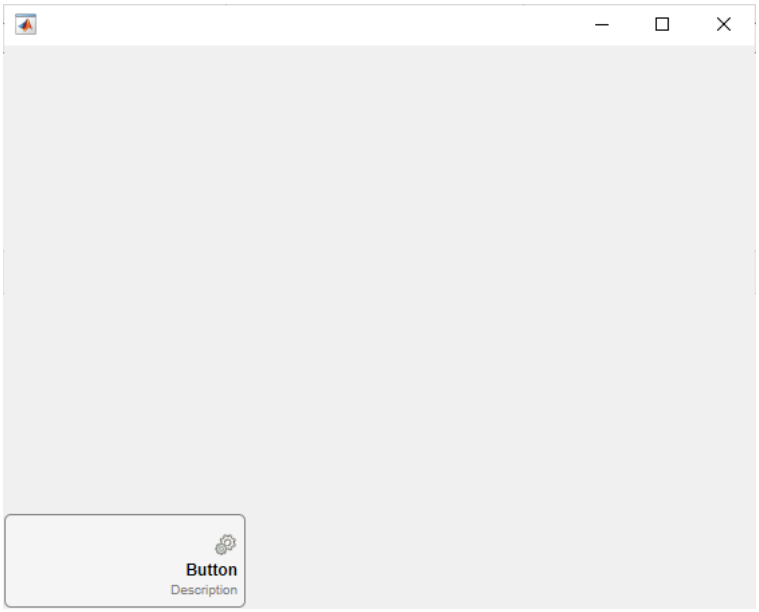
- ButtonPushed

Example

```
btn = ccTools.Button
```

```
btn =
  Button with properties:

              Model: IconPlusText
               Text: 'Button'
        Description: 'Description'
               Icon: [1×0 char]
      IconAlignment: top
    HorizontalAlign: right
      VerticalAlign: center
      ColumnSpacing: 5
         RowSpacing: 0
     RowTextSpacing: 0
        BorderWidth: 1
        BorderColor: '#808080'
       BorderRadius: '5px'
      BorderPadding: 5
          IconWidth: 18
         IconHeight: 18
         FontFamily: Helvetica
        tFontWeight: bold
          tFontSize: 12
         tFontColor: 'black'
          dFontSize: 10
         dFontColor: '#808080'
             Enable: 1
    DisabledOpacity: 0.3500
    ButtonPushedFcn: ''
           Position: [1 1 180 70]
```



ccTools.MessageBox

Display a modal message box in front of the specified uifigure window.

Syntax

```
ccTools.MessageBox(parent, message)
ccTools.MessageBox(parent, message, Name, Value)
```

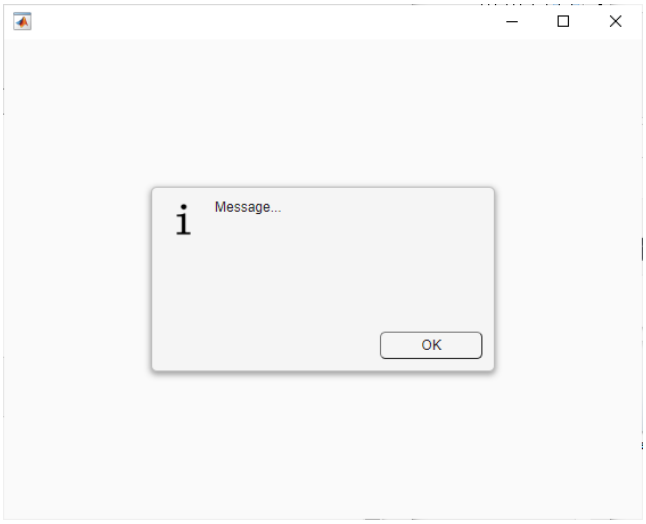
Name-Value Optional Arguments

(argument name, type, and possible values or default value)

- **winWidth** - *scalar text (char | string)* - "302px"
- **winHeight** - *scalar text (char | string)* - "162px"
- **winBackgroundColor** - *numeric array or scalar text (char | string)* - "#f5f5f5"
- **iconFullFile** - *scalar text (char | string)* - ""
- **iconWidth** - *scalar text (char | string)* - "35px"
- **iconHeight** - *scalar text (char | string)* - "35px"
- **msgFontFamily** - *scalar text (char | string)* - "Helvetica"
- **msgFontSize** - *scalar text (char | string)* - "12px"
- **msgFontColor** - *numeric array or scalar text (char | string)* - valid color
- **msgTextAlign** - *scalar text (char | string)* - "'left" | "center" | "right" | "justify"
- **buttonWidth** - *scalar text (char | string)* - "90px"
- **buttonHeight** - *scalar text (char | string)* - "24px"
- **buttonBackgroundColor** - *numeric array or scalar text (char | string)* - valid color
- **buttonBorderRadius** - *scalar text (char | string)* - "5px"
- **buttonBorderWidth** - *scalar text (char | string)* - "1px"
- **buttonBorderColor** - *numeric array or scalar text (char | string)* - valid color
- **buttonFontFamily** - *scalar text (char | string)* - "Helvetica"
- **buttonFontSize** - *scalar text (char | string)* - "12px"
- **buttonFontColor** - *numeric array or scalar text (char | string)* - "#212121"
- **buttonTextAlign** - *scalar text (char | string)* - "'left" | "center" | "right" | "justify"

Example

```
f = uifigure;
ccTools.MessageBox(f, 'Message...')
```



ccTools.ProgressDialog

Display a modal progress dialog element in front of the specified component (uifigure, uihtml, uitable and so on).

Syntax

```
d = ccTools.ProgressDialog(parent)
d = ccTools.ProgressDialog(parent, Name, Value)
```

Name-Value Optional Arguments

(argument name, type, and possible values or default value)

- **size** - *scalar text (char | string)* - "40px"
- **color** - *numeric array or scalar text (char | string)* - "#d95319"

Example

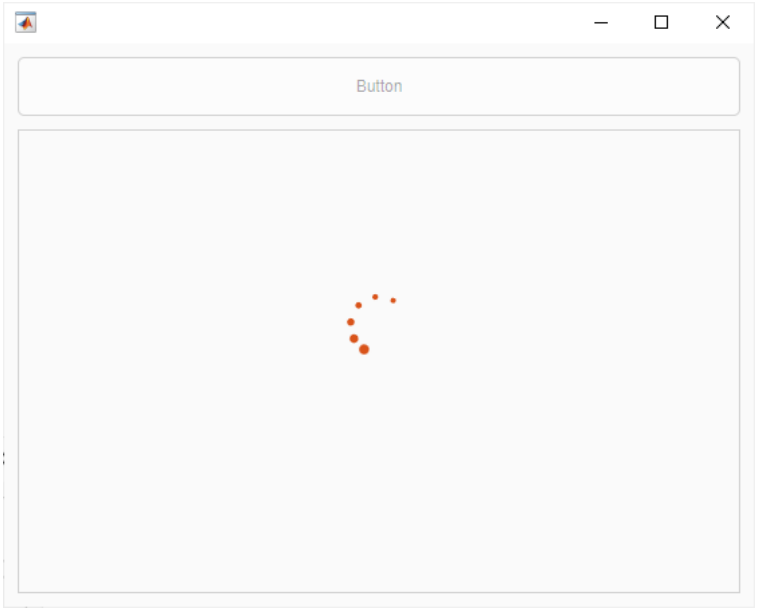
```
f = uifigure;
g = uigridlayout(f, 'RowHeight', {44, '1x'}, 'ColumnWidth', {'1x'});
b = uibutton(g);
p = uipanel(g);
drawnow

d = ccTools.ProgressDialog(f)
```

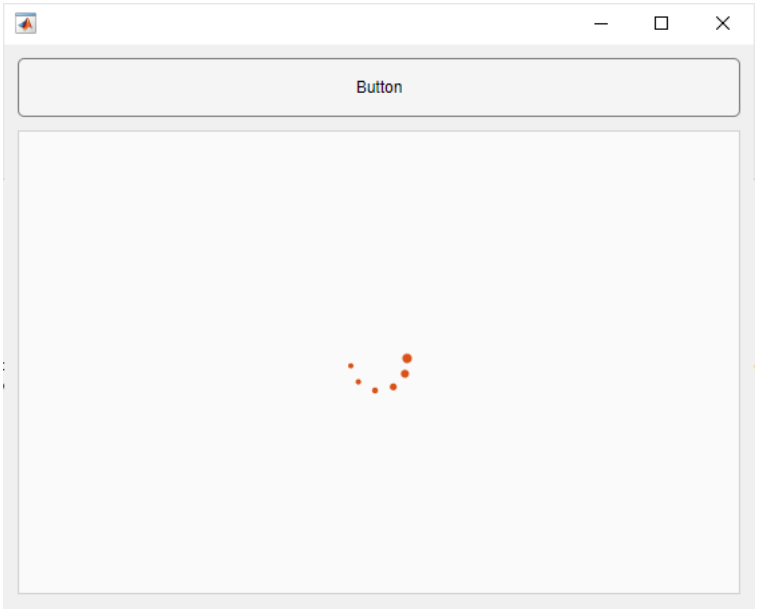
```
d =
  modalDialog with properties:

          type: 'ProgressDialog'
        webWin: [1×1 matlab.internal.webwindow]
     parentTag: 'd1b249f0-c1d4-4891-b193-edccc059fc5a'
       dataTag: 'fba63cfd-7b78-4daf-a4cb-67f409af6912'
    onCreation: '// zIndex↵var zIndex = 1000;↵var objList =
document.querySelectorAll('div[role="dialog"]');↵objList.forEach(element => {↵    let idx =
parseInt(element.style.zIndex);↵    zIndex = (zIndex < idx ? idx : zIndex);↵});↵objList = undefined;↵↵//
Elements creation↵let s_T005309784 = document.createElement("style");↵let u_T005309784 =
document.createElement("div");↵let w_T005309784 =
document.createElement("div");↵↵document.head.appendChild(s_T005309784);↵document.body.appendChild(u_T0053097
84);↵document.body.appendChild(w_T005309784);↵↵s_T005309784.setAttribute("data-type",
"ccTools.ProgressDialog");↵u_T005309784.setAttribute("data-type",
"ccTools.ProgressDialog");↵w_T005309784.setAttribute("data-type",
"ccTools.ProgressDialog");↵↵s_T005309784.setAttribute("data-tag", "fba63cfd-7b78-4daf-a4cb-
67f409af6912");↵u_T005309784.setAttribute("data-tag", "fba63cfd-7b78-4daf-a4cb-
67f409af6912");↵w_T005309784.setAttribute("data-tag", "fba63cfd-7b78-4daf-a4cb-67f409af6912");↵↵//
CSS↵s_T005309784.innerHTML = `↵    :root {↵      --sk-size: 40px;↵      --sk-color: #d95319;↵    }↵    ↵
.sk-chase {↵      width: var(--sk-size);↵      height: var(--sk-size);↵      position: relative;↵
animation: sk-chase 2.5s infinite linear both; ↵    }↵    ↵    .sk-chase-dot {↵      width: 100%;↵
height: 100%;↵      position: absolute;↵      left: 0;↵      top: 0; ↵      animation: sk-chase-dot 2.0s
infinite ease-in-out both; ↵    }↵    ↵    .sk-chase-dot:before {↵      content: '';↵      display: block;↵
width: 25%;↵      height: 25%;↵      background-color: var(--sk-color);↵      border-radius: 100%;↵
animation: sk-chase-dot-before 2.0s infinite ease-in-out both; ↵    }↵    ↵    .sk-chase-dot:nth-child(1) {
animation-delay: -1.1s; }↵    .sk-chase-dot:nth-child(2) { animation-delay: -1.0s; }↵    .sk-chase-dot:nth-
child(3) { animation-delay: -0.9s; }↵    .sk-chase-dot:nth-child(4) { animation-delay: -0.8s; }↵    .sk-
chase-dot:nth-child(5) { animation-delay: -0.7s; }↵    .sk-chase-dot:nth-child(6) { animation-delay: -0.6s;
}↵    .sk-chase-dot:nth-child(1):before { animation-delay: -1.1s; }↵    .sk-chase-dot:nth-child(2):before {
animation-delay: -1.0s; }↵    .sk-chase-dot:nth-child(3):before { animation-delay: -0.9s; }↵    .sk-chase-
dot:nth-child(4):before { animation-delay: -0.8s; }↵    .sk-chase-dot:nth-child(5):before { animation-delay:
```

```
-0.7s; }↵    .sk-chase-dot:nth-child(6):before { animation-delay: -0.6s; }↵    ↵    @keyframes sk-chase {↵
100% { transform: rotate(360deg); } ↵    }↵    ↵    @keyframes sk-chase-dot {↵      80%, 100% { transform:
rotate(360deg); } ↵    }↵    ↵    @keyframes sk-chase-dot-before {↵      50% {↵        transform:
scale(0.4); ↵      } 100%, 0% {↵        transform: scale(1.0); ↵      } ↵    }↵`;↵↵// Background
layer↵u_T005309784.setAttribute("class", "backgroundLayer");↵u_T005309784.style.position  =
"absolute";↵u_T005309784.style.left      = "0%";↵u_T005309784.style.top       =
"0%";↵u_T005309784.style.width     = "100%";↵u_T005309784.style.height    = "100%";↵u_T005309784.style.zIndex
= zIndex+1;↵↵// Progress dialog↵w_T005309784.setAttribute("role", "dialog");↵w_T005309784.style.position  =
"absolute";↵w_T005309784.style.left      = "50%";↵w_T005309784.style.top       =
"50%";↵w_T005309784.style.transform = "translate(-50%, -50%)";↵w_T005309784.style.zIndex    =
zIndex+2;↵w_T005309784.innerHTML       = `↵    <div class="sk-chase">↵      <div class="sk-chase-dot"></div>↵
<div class="sk-chase-dot"></div>↵      <div class="sk-chase-dot"></div>↵      <div class="sk-chase-
dot"></div>↵      <div class="sk-chase-dot"></div>↵      <div class="sk-chase-dot"></div>↵    </div>↵`;'
     onCleanup: 'zIndex =
undefined;↵↵s_T005309784.remove();↵u_T005309784.remove();↵w_T005309784.remove();↵↵delete
s_T005309784;↵delete u_T005309784;↵delete w_T005309784;'
      rootPath: 'C:\P&D\ccTools\+ccTools'
```




```
d = ccTools.ProgressDialog(p);
```



ccTools.compCustomization

Allow customization of some properties of the built-in ui Matlab components, such as "windowMinSize", "backgroundColor", "borderRadius" and so on.

Syntax

```
[status, errorMsg] = ccTools.compCustomization(comp, Name, Value)
```

Class-editable properties

- **matlab.ui.Figure**: 'windowMinSize'
- **matlab.ui.container.ButtonGroup**: 'backgroundColor', 'borderRadius', 'borderWidth' and 'borderColor'
- **matlab.ui.container.CheckBoxTree**: 'backgroundColor', 'borderRadius', 'borderWidth' and 'borderColor'
- **matlab.ui.container.GridLayout**: 'backgroundColor'
- **matlab.ui.container.Panel**: 'backgroundColor', 'borderRadius', 'borderWidth' and 'borderColor'
- **matlab.ui.container.Tab**: 'backgroundColor'
- **matlab.ui.container.TabGroup**: 'backgroundColor', 'backgroundHeaderColor', 'borderRadius', 'borderWidth', 'borderColor', 'fontFamily', 'fontStyle', 'fontWeight', 'fontSize' and 'color
- **matlab.ui.container.Tree**: 'backgroundColor', 'borderRadius', 'borderWidth' and 'borderColor'
- **matlab.ui.control.Button**: 'borderRadius', 'borderWidth' and 'borderColor'
- **matlab.ui.control.CheckBox**: 'backgroundColor', 'borderRadius', 'borderWidth' and 'borderColor'
- **matlab.ui.control.DropDown**: 'borderRadius', 'borderWidth' and 'borderColor'
- **matlab.ui.control.EditField**: 'borderRadius', 'borderWidth' and 'borderColor'
- **matlab.ui.control.ListBox**: 'borderRadius', 'borderWidth' and 'borderColor'
- **matlab.ui.control.NumericEditField**: 'borderRadius', 'borderWidth' and 'borderColor'
- **matlab.ui.control.StateButton**: 'borderRadius', 'borderWidth' and 'borderColor'
- **matlab.ui.control.Table**: 'backgroundColor', 'backgroundHeaderColor', 'borderRadius', 'borderWidth', 'borderColor', 'fontFamily', 'fontStyle', 'fontWeight', 'fontSize' and 'color'
- **matlab.ui.control.TextArea**: 'backgroundColor', 'borderRadius', 'borderWidth', 'borderColor' and 'textAlign'

Output Arguments

- **status** - logical - true | false
- **errorMsg** - char

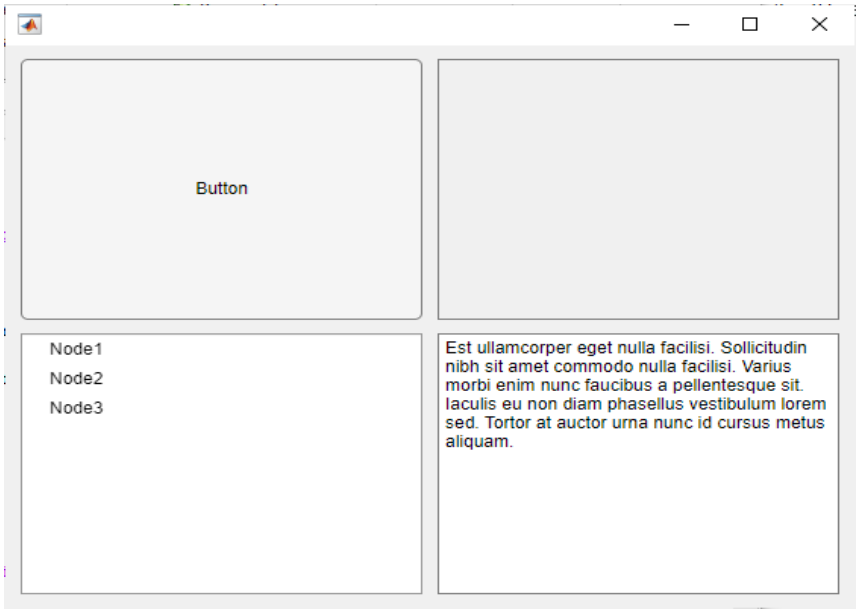
Input Name-Value Arguments

(argument name, type, and possible values or example value)

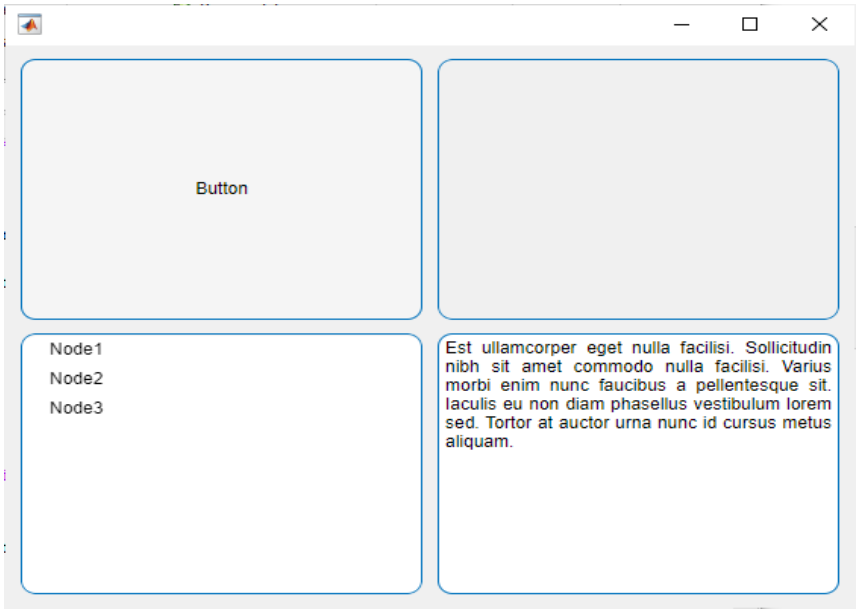
- **windowMinSize** - *scalar text (char | string)* - "640px"
- **backgroundColor** - *numeric array or scalar text (char | string)* - "#f5f5f5"
- **backgroundHeaderColor** - *numeric array or scalar text (char | string)* - "#f5f5f5"
- **borderRadius** - *scalar text (char | string)* - "10px"
- **borderWidth** - *scalar text (char | string)* - "2px"
- **borderColor** - *numeric array or scalar text (char | string)* - "blue"
- **textAlign** - *scalar text (char | string)* - "'left" | "center" | "right" | "justify"
- **fontFamily** - *scalar text (char | string)* - "Helvetica"
- **fontStyle** - *scalar text (char | string)* - "italic" | "normal"
- **fontWeight** - *scalar text (char | string)* - "bold" | "normal"
- **fontSize** - *scalar text (char | string)* - "12px"
- **color** - *numeric array or scalar text (char | string)* - "#212121"

Examples

```
f = uifigure;
g = uigridlayout(f, 'RowHeight', {'1x', '1x'}, 'ColumnWidth', {'1x', '1x'});
b = uibutton(g);
p = uipanel(g);
t = uitree(g); uitreenode(t, "Text", 'Node1'); uitreenode(t, "Text", 'Node2'); uitreenode(t, "Text", 'Node3');
a = uitextarea(g, "Value", ['Est ullamcorper eget nulla facilisi. Sollicitudin nibh sit amet commodo nulla '     ...
                            'facilisi. Varius morbi enim nunc faucibus a pellentesque sit. Iaculis eu non diam ' ...
                            'phasellus vestibulum lorem sed. Tortor at auctor urna nunc id cursus metus aliquam.']);
drawnow
```

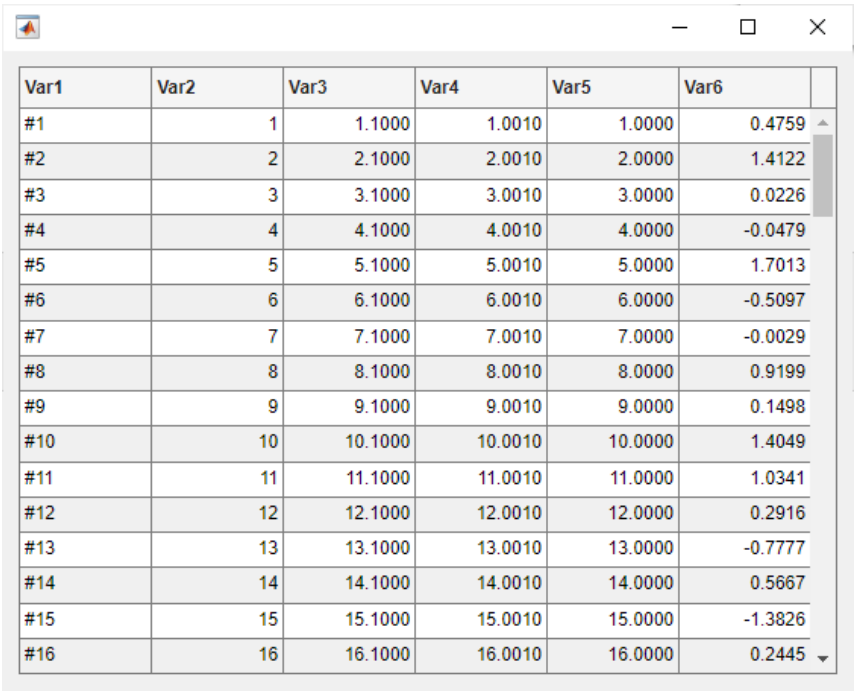



```
ccTools.compCustomization(b, 'borderRadius', '10px', 'borderColor', '#0072bd');
ccTools.compCustomization(p, 'borderRadius', '10px', 'borderColor', '#0072bd');
ccTools.compCustomization(t, 'borderRadius', '10px', 'borderColor', '#0072bd');
ccTools.compCustomization(a, 'borderRadius', '10px', 'borderColor', '#0072bd', 'textAlign', 'justify');
```



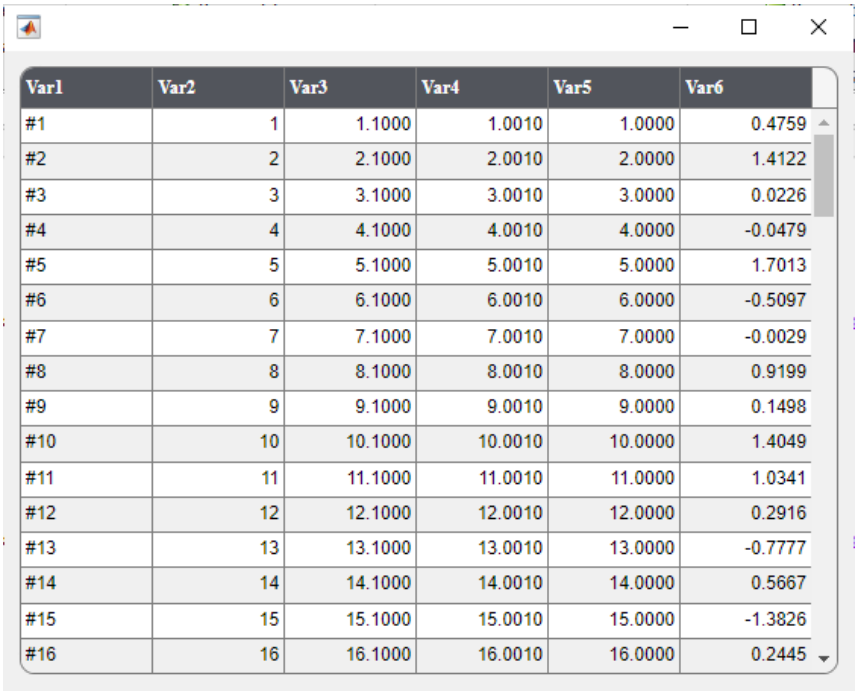
uitable customization...

```
f = uifigure;
g = uigridlayout(f, 'RowHeight', {'1x'}, 'ColumnWidth', {'1x'});
t = uitable(g, 'Data', table("#"+string((1:100)'), (1:100)', (1:100)'+.1, (1:100)'+.001, (1:100)'+.00001,randn(100,1)));
drawnow
```



Var1	Var2	Var3	Var4	Var5	Var6
#1	1	1.1000	1.0010	1.0000	0.4759
#2	2	2.1000	2.0010	2.0000	1.4122
#3	3	3.1000	3.0010	3.0000	0.0226
#4	4	4.1000	4.0010	4.0000	-0.0479
#5	5	5.1000	5.0010	5.0000	1.7013
#6	6	6.1000	6.0010	6.0000	-0.5097
#7	7	7.1000	7.0010	7.0000	-0.0029
#8	8	8.1000	8.0010	8.0000	0.9199
#9	9	9.1000	9.0010	9.0000	0.1498
#10	10	10.1000	10.0010	10.0000	1.4049
#11	11	11.1000	11.0010	11.0000	1.0341
#12	12	12.1000	12.0010	12.0000	0.2916
#13	13	13.1000	13.0010	13.0000	-0.7777
#14	14	14.1000	14.0010	14.0000	0.5667
#15	15	15.1000	15.0010	15.0000	-1.3826
#16	16	16.1000	16.0010	16.0000	0.2445

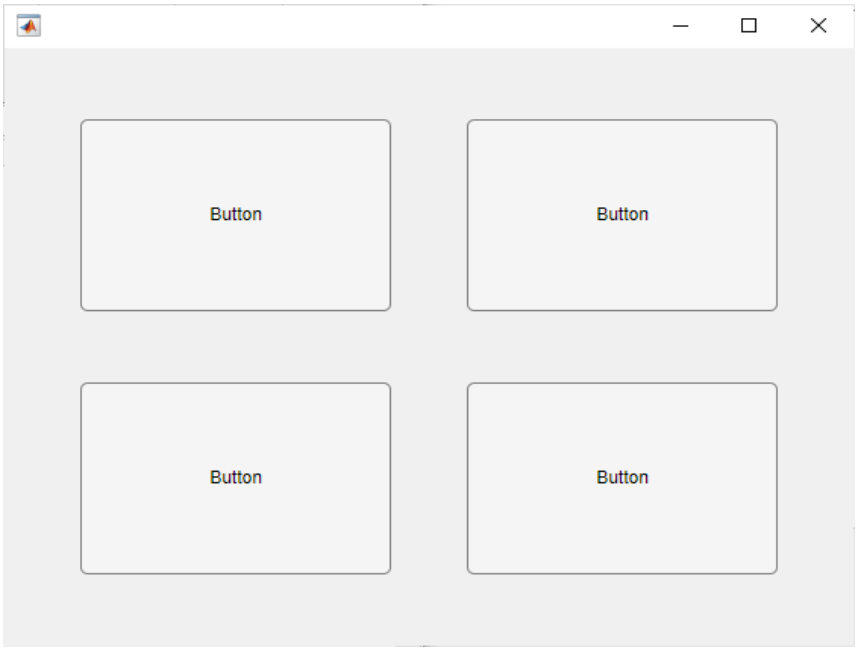
```
ccTools.compCustomization(t, 'backgroundHeaderColor', '#52555c', ...
                             'borderRadius', '10px',              ...
                             'fontFamily', 'Times New Roman',     ...
                             'color', 'white');
```



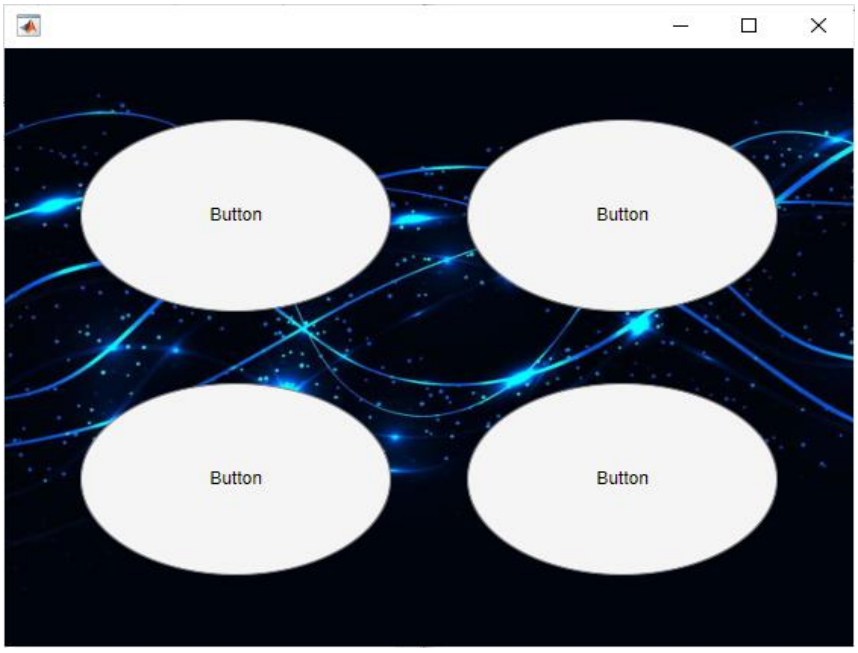
Var1	Var2	Var3	Var4	Var5	Var6
#1	1	1.1000	1.0010	1.0000	0.4759
#2	2	2.1000	2.0010	2.0000	1.4122
#3	3	3.1000	3.0010	3.0000	0.0226
#4	4	4.1000	4.0010	4.0000	-0.0479
#5	5	5.1000	5.0010	5.0000	1.7013
#6	6	6.1000	6.0010	6.0000	-0.5097
#7	7	7.1000	7.0010	7.0000	-0.0029
#8	8	8.1000	8.0010	8.0000	0.9199
#9	9	9.1000	9.0010	9.0000	0.1498
#10	10	10.1000	10.0010	10.0000	1.4049
#11	11	11.1000	11.0010	11.0000	1.0341
#12	12	12.1000	12.0010	12.0000	0.2916
#13	13	13.1000	13.0010	13.0000	-0.7777
#14	14	14.1000	14.0010	14.0000	0.5667
#15	15	15.1000	15.0010	15.0000	-1.3826
#16	16	16.1000	16.0010	16.0000	0.2445

A transparent backgroundColor...

```
f = uifigure;
g1 = uigridlayout(f, 'RowHeight', {'1x'}, 'ColumnWidth', {'1x'}, 'Padding', [0,0,0,0]);
uiimage(g1, "ImageSource", 'myImage.jpg', 'ScaleMethod', 'fill');
g2 = uigridlayout(f, 'ColumnSpacing', 50, 'RowSpacing', 50, 'Padding', 50);
b1 = uibutton(g2); b2 = uibutton(g2); b3 = uibutton(g2); b4 = uibutton(g2);
drawnow
```




```
ccTools.compCustomization(g2, 'backgroundColor', 'transparent')
ccTools.compCustomization(b1, 'borderRadius', '50%')
ccTools.compCustomization(b2, 'borderRadius', '50%')
ccTools.compCustomization(b3, 'borderRadius', '50%')
ccTools.compCustomization(b4, 'borderRadius', '50%')
```



Developed by Eric Magalhães Delgado
June 17, 2023