Programmatically Create a User Interface with a Graphical Table

This example shows how to create a graphical table in a user interface using `uitable`. It also shows how to modify the appearance of the table and how to restrict the changes users can make to the data in the table.

Try it in MATLAB

Create Graphical Table with Simple Numeric Data

The function `uitable` creates an empty graphical table. You can populate the table by setting the `Data` property. For example, set the data displayed to be a magic square.

```
f = figure('Position', [100 100 752 250]);
t = uitable('Parent', f, 'Position', [25 50 700 200], 'Data', magic(10))
```

```
t =

  Table with properties:

              Data: [10x10 double]
       ColumnWidth: 'auto'
    ColumnEditable: []
  CellEditCallback: ''
          Position: [25 50 700 200]
             Units: 'pixels'

  Use GET to show all properties
```

	1	2	3	4	5	6	7	8	9
1	92	99	1	8	15	67	74	51	
2	98	80	7	14	16	73	55	57	
3	4	81	88	20	22	54	56	63	
4	85	87	19	21	3	60	62	69	
5	86	93	25	2	9	61	68	75	
6	17	24	76	83	90	42	49	26	
7	23	5	82	89	91	48	30	32	
8	79	6	13	95	97	29	31	38	
9	10	12	94	96	78	35	37	44	

Create Graphical Table with Mixed Type Data

Display mixed type data by setting the `Data` property to a cell array.

```
load patients LastName Age Weight Height SelfAssessedHealthStatus                  % load data
PatientData = [LastName num2cell([Age Weight Height]) SelfAssessedHealthStatus];   % convert to cell array

t.Data = PatientData;
```

Customize the Display

You can customize the display of a table in several ways. Use the `ColumnName` property to add headings to the top of each column. To create multi-line headings, use the divider line symbol.

```
t.ColumnName = {'LastName', 'Age', 'Weight', 'Height', 'Self Assessed|Health Status'};
```

To adjust the widths of the columns, use the `ColumnWidth` property. The `ColumnWidth` property is a *1* by *N* cell array where *N* is the number of columns in the table. You can choose to set a specific width for columns or autofit the width based on the contents.

```
t.ColumnWidth = {100, 'auto', 'auto', 'auto', 150};
```

To completely remove the row names, set the `RowName` property to empty using `[]`.

```
t.RowName = [];
```

You can resize the table to remove any extra space using the `Position` property.

```
t.Position = [15 25 495 200];
```

By default, tables use row striping. To turn off row striping, set the `RowStriping` property to `'off'`. To control the colors of the row stripes, set two different colors for the `BackgroundColor` property. Use the `ForegroundColor` property to control the color of the text.

```
t.BackgroundColor = [.4 .4 .4; .4 .4 .8];
t.ForegroundColor = [1 1 1];
```

LastName	Age	Weight	Height	Self Assessed Health Status
Smith	38	176	71	Excellent
Johnson	43	163	69	Fair
Williams	38	131	64	Good
Jones	40	133	67	Fair
Brown	49	119	64	Good
Davis	46	142	68	Good
Miller	33	142	64	Good
Wilson	40	180	68	Good
Moore	28	183	68	Excellent

Restrict Editing of Cell Values

To restrict the ability for users to edit data in the columns of the table, set the `ColumnEditable` property. By default, data can not be edited. Setting the `ColumnEditable` property to `true` for a column allows the data in that column to be edited.

```
t.ColumnEditable = [false true true true true];
```

LastName	Age	Weight	Height	Self Assessed Health Status
Smith	38	176	71	Excellent
Johnson	43	163	69	Fair
Williams	38	131	64	Good
Jones	40	133	67	Fair
Brown	49	119	64	Good
Davis	46	142	68	Good
Miller	33	142	64	Good
Wilson	40	180	68	Good
Moore	28	183	68	Excellent

Change Column Format

The `ColumnFormat` property controls how data is displayed and edited for each column. To specify choices for a popup menu use a cell array of strings as the column format. In this example, the *Self Assessed Health Status* column uses a popup menu with four option - `Excellent`, `Fair`, `Good`, and `Poor`.

```
t.ColumnFormat = {[] [] [] [] {'Excellent', 'Fair', 'Good', 'Poor'}};
```

LastName	Age	Weight	Height	Self Assessed Health Status
Smith	38	176	71	Excellent
Johnson	43	163	69	Fair
Williams	38	131	64	Good
Jones	40	133	67	Fair
Brown	49	119	64	Good
Davis	46	142	68	Good
Miller	33	142	64	Good
Wilson	40	180	68	Good
Moore	28	183	68	Excellent

Create Callback

The table object has two commonly used callbacks. The `CellSelectionCallback` is called when the user changes the currently selected cell in the table. The `CellEditCallback` is called when the user changes a value in a cell.

```
t.CellEditCallback = @ageCheckCB;
```

For example, if you want the *Age* column to contain values that must be between `0` and `120`, set the `CellEditCallback` to a function with this format:

```
function ageCheckCB(src, eventdata)

if (eventdata.Indices(2) == 2 && ...                % check if column 2
      (eventdata.NewData < 0 || eventdata.NewData > 120))
   tableData = src.Data;
   tableData{eventdata.Indices(1), eventdata.Indices(2)} = eventdata.PreviousData;
   src.Data = tableData;                            % set the data back to its original value
   warning('Age must be between 0 and 120.')        % warn the user
end

end
```

If a value entered in the *Age* column is outside of the acceptable range, the callback function will issue an warning and set the cell contents back to the original value.

Get All Table Properties

Graphics objects in MATLAB have many properties. To see all the properties of a Table object, use the `get` command.

```
get(t)
```

```
          BackgroundColor: [2x3 double]
             BeingDeleted: 'off'
               BusyAction: 'queue'
            ButtonDownFcn: ''
         CellEditCallback: @ageCheckCB
    CellSelectionCallback: ''
                 Children: [0x0 handle]
           ColumnEditable: [0 1 1 1 1]
             ColumnFormat: {[]  []  []  []  {1x4 cell}}
               ColumnName: {5x1 cell}
              ColumnWidth: {[100]  'auto'  'auto'  'auto'  [150]}
                CreateFcn: ''
                     Data: {100x5 cell}
                DeleteFcn: ''
                   Enable: 'on'
                   Extent: [0 0 479 1940]
                FontAngle: 'normal'
                 FontName: 'Helvetica'
                 FontSize: 10
                FontUnits: 'points'
               FontWeight: 'normal'
          ForegroundColor: [1 1 1]
        HandleVisibility: 'on'
           InnerPosition: [15 25 495 200]
           Interruptible: 'on'
             KeyPressFcn: ''
           KeyReleaseFcn: ''
           OuterPosition: [15 25 495 200]
                  Parent: [1x1 Figure]
                Position: [15 25 495 200]
    RearrangeableColumns: 'off'
                 RowName: ''
             RowStriping: 'on'
                     Tag: ''
           TooltipString: ''
                    Type: 'uitable'
           UIContextMenu: [0x0 GraphicsPlaceholder]
                   Units: 'pixels'
                UserData: []
                 Visible: 'on'
```