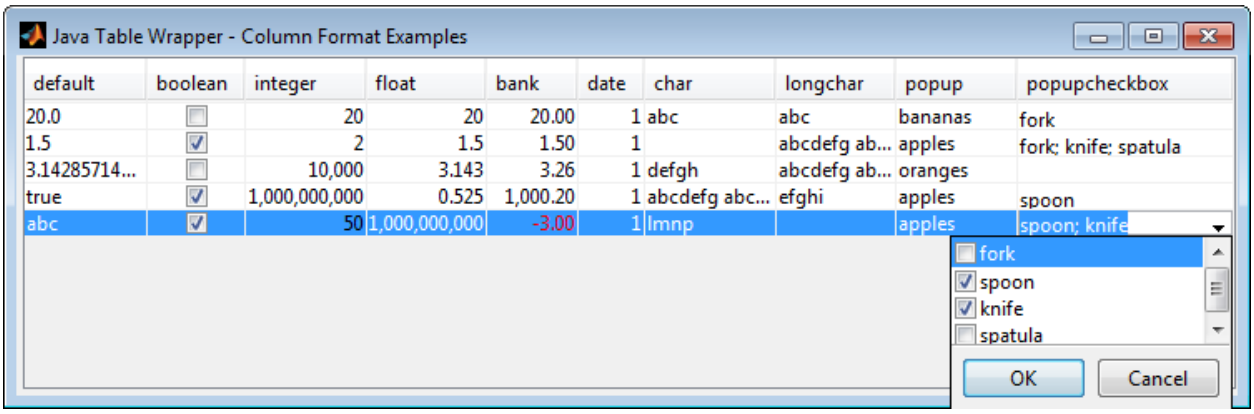
Java Table Wrapper for User Interfaces

This package provides the capability to easily create Java-based table controls for an application or user interface using a simple MATLAB® object-oriented interface. This package includes and wraps the Java® code necessary to implement a Java® JTable, enabling you to create the tree by writing only MATLAB® code, for a set of included features. This is useful for situations where you need to overcome a limitation in MATLAB's uitable.



A rich set of added features and capabilities are included:

- Additional column formats: multil-line text, popup list with checkboxes.
- Additional column resize policies (e.g. size the columns to fit the table).
- Method to set column widths to best fit the included data.
- Row based selection model.
- Get/Set individual cells' data.
    - Table scrollpanel "sticks" when updating data programmatically.
- Settable callbacks for your user interface for each event, including mouse events.

Limitations, known issues, and potential future features

- This table is set up for row-based selection only. Cell-based selection is not currently supported.
- Row headers are not implemented.
- Tooltips that are customized individually to columns, cells, etc. are not implemented.
- Date column format is not fully implemented.
- Column sorting is not fully implemented, but is possible in the future with additional work.
- Column filtering is not fully implemented, but is possible in the future with additional work.
- Drag and drop support is not implemented.
- Positioning of the table is sometimes offset by several pixels. This is a known issue, and the best workaround is to add padding as needed.
- In some instances, resizing a UI's height from the top of the figure window may incorrectly size the table.
- This table is not currently configured for large datasets. Performance improvements may be possible. Currently, we recommend no more than 10,000 cells.

- If you are using multiple Java® wrappers (such as in Tree Controls for User Interfaces), you may need to change the way the Java® JAR files are added to MATLAB's Java class path. In both widgets, they attempt to add these to the path dynamically, but this can cause warnings when both are used in the same application. It is recommended you either add both to MATLAB's static Java path or add both to MATLAB's dynamic Java path initially on the application startup before the widgets are constructed.

Please contact MathWorks Consulting for assistance with improvements to this table widget, or for advice on user interface planning, architecture, and development.