TreeTable revisited

Posted by aephyr [1] on October 10, 2010 at 8:24 PM PDT

It might be scary to think about starting an implementation by extending JComponent, but I wanted to try something new and that just happens to be where I started. Ironically it doesn't reinvent the wheel as much as Netbean's Outline component but that's only because I used a dedicated JTable and JTree under the hood. Right now everything is implemented in the JComponent as opposed to a BasicTreeTableUI. I'd like to move the UI stuff so that creating a SwingX or other JTable/JTree extension compatible TreeTable would be possible (for example, in SwingX, the dedicated tree/tables would be JXTree/JXTable). Also, a TreeTableUI from scratch should be able to start its implementation without prerequisites; though I don't know if any L&F's actually do start from javax.swing.plaf. instead of javax.swing.plaf.basic.

Anyway onto some of the implementations that have been changed:

There is one thing that I really liked with Outline's implementation and that is the TreeModel, RowModel separation. So that is more of a change from JXTreeTable.

Variable row heights are supported. The JTable's row heights are kept in sync with the JTree's row heights. The JTree's calculated row height takes into account all cell renderer's preferred size in the row.

No more clumsy TableModelEvents that utilize fireTableDataChanged or delayedFireTableChanged when all the data has in fact not changed. The most minimalistic TableModelEvent that can be determined is produced for changes to the TableModel. There is a public getTableModel() method so that changes to the TableModel can be heard in terms of a TableModelListener. It differs from listening to the TreeModel/RowModel as rows inserted/deleted due to expansion/collapse can be listened to, which provides more information that TreeExpansionEvents (you get the number of rows that have came into/fallen out of view).

The JTable's ListSelectionModel is implemented by wrapping the JTree's TreeSelectionModel, literally. It doesn't duplicate the data into a DefaultListSelectionModel. This is also queryable via getRowSelectionModel() so that the selection can be listened to with a ListSelectionListener if desired.

Editing has been butchered. Or rather, nothing custom in the way of editing has been implemented yet.

The source is in TreeTable.java:

http://code.google.com/p/aephyr/source/browse/#svn/trunk/src/aephyr/swing [2]

With dependencies in the treetable and event folders.

Any opinions? Is the painting mechanism bad (aside from not being in a UI delegate)?

Edit: Moved UI details to BasicTreeTableUI:

http://code.google.com/p/aephyr/source/browse/trunk/src/aephyr/swing/ui/... [3]

Special Sections [4] [5] Using ttf font runtime in an applet

Swing & AWT [6]

Source URL: http://www.java.net/node/706971

Links:
[1] http://www.java.net/blog/531606
[2] http://code.google.com/p/aephyr/source/browse/#svn/trunk/src/aephyr/swing
[3] http://code.google.com/p/aephyr/source/browse/trunk/src/aephyr/swing/ui/BasicTreeTableUI.java
[4] http://www.java.net/forum/topic/javadesktop/java-desktop-technologies/swing-awt/special-sections
[5] http://www.java.net/node/709392
[6] http://www.java.net/forums/read-only-archived-forums/javadesktop/swing-awt