24th April 2009

Creating GtkTreeView with Glade-3, part 2

In first part of this Glade tutorial, we created tree view and added some data to it. Today, wi'll expand our sample application a bit and add some controls that will enable us to modify tree view content - moving items up and down.

Contents:

Manipulating data

First thing we need to do it to create widgets that trigger the modifications of tree view data. Let's load builder project from first part of this tutorial and fill the bottom compartment of vbox.

I'll just quickly describe what needs to be done in order to create GUI like the one on the image below:

1. Add GtkFrame in bottom compartment of vbox (the one that we left empty in previous post), change it's label to "Controls" and it's expand property to NO.
2. Add GtkTable with 2 columns and 1 row into the frame and set column spacing to 6 px.
3. Add two buttons to the table: left one should be "gtk-go-up" stock button and right one "gtk-go-down". Connect their "clicked" signal to "cb_move" function.

The final result should look something like this:

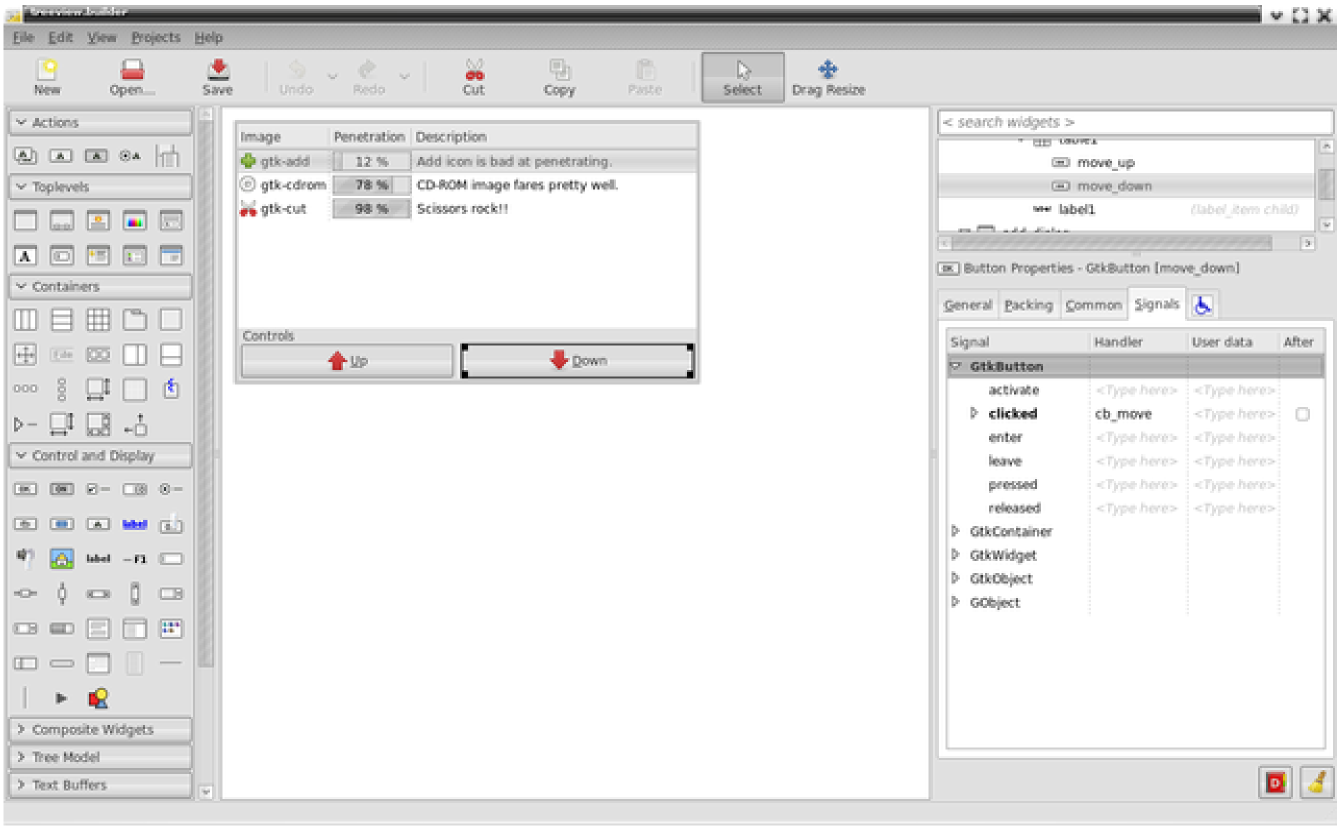

Image	Penetration	Description
gtk-add	12 %	Add icon is bad at penetrating.
gtk-cdrom	78 %	CD-ROM image fares pretty well.
gtk-cut	98 %	Scissors rock!!

Signal	Handler	User data	After
GtkButton			
activate	<Type here>	<Type here>	
clicked	cb_move	<Type here>	☐
enter	<Type here>	<Type here>	
leave	<Type here>	<Type here>	
pressed	<Type here>	<Type here>	
released	<Type here>	<Type here>	
GtkContainer			
GtkWidget			
GtkObject			
GObject			

[http://img105.imageshack.us/img105/5417/20090424094457z.png] You can download builder file from here [http://tadeboro.googlepages.com/treeview.builder] , just to be sure that we're working with the same GUI;) Now let's start coding. And for the first time, I'll provide sample application for C and Python coders.

First, we need to load our interface. If you listened to my advice and read Micah Carrick's tutorial first, there is nothing new in my code snippets.

All that we need to do now is to write cb_move function that will take care of moving our tree view items. The way we'll be doing it is to check which button has been pressed and based on this decide if we're moving item up or down. Actual move will be done using gtk_list_store_swap function. And that's it! Now dissect this code and have fun.

```
/* treeview.c */

#include <gtk/gtk.h>

typedef struct _Data Data;
struct _Data
{
    GtkWidget *up;   /* Up button */
    GtkWidget *down; /* Down button */
    GtkWidget *tree; /* Tree view */
};



G_MODULE_EXPORT void
cb_move( GtkWidget *button,
         Data      *data )
{
    gboolean          direction = TRUE; /* TRUE means "move up", FALSE "move down" */
    GList            *rows, *tmp;       /* List of selected row */
    GtkTreeModel     *model;            /* Model with data, which we'll be moving. */
    GtkTreeSelection *sel = gtk_tree_view_get_selection( GTK_TREE_VIEW( data->tree ) );

    /* If nothing is selected, return. */
    if( ! gtk_tree_selection_count_selected_rows( sel ) )
        return;

    /* In which direction we'll be moving? */
    if( button == data->down )
        direction = FALSE;

    /* Get selected rows */
    rows = gtk_tree_selection_get_selected_rows( sel, &model );

    /* Get new path for each selected row and swap items. */
    for( tmp = rows; tmp; tmp = g_list_next( tmp ) )
    {
        GtkTreePath *path1, *path2; /* Paths. */
        GtkTreeIter  iter1, iter2;  /* Iters for swapping items. */

        /* Copy path */
        path1 = (GtkTreePath *)tmp->data;
        path2 = gtk_tree_path_copy( path1 );
        /* Move path2 in right direction */
        if( direction )
            gtk_tree_path_prev( path2 );
        else
            gtk_tree_path_next( path2 );

        /* Compare paths and skip one iteration if the paths are equal, which means we're
         * trying to move first path up. */
        if( ! gtk_tree_path_compare( path1, path2 ) )
        {
            gtk_tree_path_free( path2 );
            continue;
        }

        /* Now finally obtain iters and swap items. If the second iter is invalid, we're
         * trying to move the last item down. */
        gtk_tree_model_get_iter( model, &iter1, path1 );
        if( ! gtk_tree_model_get_iter( model, &iter2, path2 ) )
        {
            gtk_tree_path_free( path2 );
            continue;
        }
        gtk_list_store_swap( GTK_LIST_STORE( model ), &iter1, &iter2 );
        gtk_tree_path_free( path2 );
    }

    /* Free our paths */
    g_list_foreach( rows, (GFunc)gtk_tree_path_free, NULL );
    g_list_free( rows );
}

int
main( int    argc,
      char **argv )
{
    GtkBuilder *builder;
    GtkWidget  *window;
    Data        data;

    gtk_init( &argc, &argv );

    /* Create builder */
    builder = gtk_builder_new();
    gtk_builder_add_from_file( builder, "treeview.builder", NULL );

    window    = GTK_WIDGET( gtk_builder_get_object( builder, "window1" ) );
    data.up   = GTK_WIDGET( gtk_builder_get_object( builder, "move_up" ) );
    data.down = GTK_WIDGET( gtk_builder_get_object( builder, "move_down" ) );
    data.tree = GTK_WIDGET( gtk_builder_get_object( builder, "treeview" ) );

    gtk_builder_connect_signals( builder, &data );
    g_object_unref( G_OBJECT( builder ) );

    gtk_widget_show( window );
    gtk_main();

    return( 0 );
}
```

```
#!/usr/bin/env python
# treeview.py

import pygtk
pygtk.require( "2.0" )
import gtk

class SampleApp( object ):
    def __init__( self ):
        builder = gtk.Builder()
        builder.add_from_file( "treeview.builder" )

        self.win   = builder.get_object( "window1" )
        self.up    = builder.get_object( "move_up" )
        self.down  = builder.get_object( "move_down" )
        self.tree  = builder.get_object( "treeview" )

        builder.connect_signals( self )

    # Callbacks
    def cb_move( self, button ):
        # Obtain selection
        sel = self.tree.get_selection()

        # If nothing is selected, return.
        if sel.count_selected_rows == 0:
            return

        # In which diection we'll be moving?
        if button == self.up:
            direction = True
        else:
            direction = False

        # Get selected path
        ( model, rows ) = sel.get_selected_rows()

        # Get new path for each selected row and swap items. */
        for path1 in rows:
            # Move path2 in right direction
             if direction:
                path2 = ( path1[0] - 1, )
            else:
                path2 = ( path1[0] + 1, )

            # If path2 is negative, we're trying to move first path up. Skip
            # one loop iteration.
            if path2[0] < 0:
                continue

            # Obtain iters and swap items. If the second iter is invalid, we're
            # trying to move the last item down. */
            iter1 = model.get_iter( path1 )
            try:
                iter2 = model.get_iter( path2 )
            except ValueError:
                continue
            model.swap( iter1, iter2 )

    def gtk_main_quit( self, object ):
        gtk.main_quit()

if __name__ == "__main__":
    app = SampleApp()
    app.win.show()
    gtk.main()
```

I hope you enjoyed this tutorial. Come back soon when I'll show you how to deal with GtkDialog and it's derivates in conjunction with glade. Bye

Posted 24th April 2009 by Tadej Borovšak

Labels: GTK+, GtkTreeView, tutorial

3 View comments

Anonymous 01 October, 2009 21:58

Tadej,
Great tutorial. Could you please make another posting regarding how you can re-order the columns (drag&drop) and add/remove the columns.

Thanks
Mo

Reply

Pierrick 10 October, 2009 16:59

Excellent!!! Your tutorials are all very helpful.

And thank your for the python code example.

Reply

Anonymous 04 April, 2010 11:55

Good work tadej.

There is a missing signal connection, the button 'move_up' and 'move_down' should correspond to the name specified in the gtk builder command, just change the button name accordingly to make it work.

Reply

Enter your comment...

Comment as: Google Accou ▼

Publish Preview