Screen subareas module for StumpWM

Michael Raskin[*]
MIPT and IUM
Moscow, Russia
raskin@mccme.ru

ABSTRACT

This paper presents a module for StumpWM tiling window manager that allows user to manage parts of X11 screen (which includes one or several physical displays) independently.

A window manager is a program in X11-based systems (X Window system is currently the most used graphical system for UNIX-like systems except MacOS X) that gets requests whenever any other application wants to create a window. This program further manages window positions according to user commands. It can also draw decorations (like window titles).

Tiling window managers are a class of window managers that allow to separately choose a way to split screen into non-overlapping areas and then distribute windows into these predefined tiles. Traditionally, most of the tiling window managers are primarily keyboard-driven and do not draw window titles by default.

StumpWM [**??**] is a tiling window manager written in Common Lisp by Shawn Betts (with later code contributions from StumpWM users). Its current maintainer is David Bjergaard.

Default StumpWM implementation of virtual desktops supposes that each window belongs to one virtual desktop and switching between virtual desktops affects all the physical displays at once. The module presented is a relatively non-intrusive way to divide a virtual desktop into multiple parts and independently choose windows for each of them.

Categories and Subject Descriptors

H.5.2 [**Information interfaces and presentation**]: User Interfaces

[*]The work was partially supported by RFBR grant 12-01-00864-■ř

General Terms

Window management

1. INTRODUCTION

StumpWM is a tiling window manager for X11 systems implemented in Common Lisp. It uses manual tiling: by default there is at most one window visible on each physical display, but user can always split the single fullscreen frame into smaller ones (manually or by a script) and see different windows in different frames.

Like many other window managers, StumpWM provides the virtual desktop (workspace) functionality, i.e. it allows user to divide windows into multiple groups and manage window layout for each group separately. When user switches to some group, only the windows belonging to this group are shown.

In StumpWM different virtual desktops also have independent layout settings. Switching virtual desktops switches the layout on every physical display at once.

Sometimes it is more convenient to switch virtual desktops on different physical displays independently (e.g. have a display dedicated to email and IM, and switch between a browser and a text editor on another display). By default, StumpWM doesn't support such workflow well.

2. DESIGN GOAL

The goal was to specify a split of a virtual desktop into areas and to choose windows placed there separately. These areas may coincide with physical displays but may also be parts of a single display.

For example, in the bottom-most area of my notebook screen I keep a terminal emulator window with various system information; I want it to be persistent while I switch the windows in any of the other frames.

3. SOLUTION DESIGN

To unify window specification, each window can have an arbitrary number of window tags. Tags are implemented as character strings not containing space. Tags are case-insensitive.

Frames can also have tags. Some of the tags (ones beginning with TG/) define the tagged group containing the frame. The commands are implemented to perform actions not on the current frame or on all frames on the current virtual desktop, but on all the frames in the same tagged group on the virtual desktop.

4. IMPLEMENTATION NOTES

In some configurations X11 server and application windows can be left in place while the window manager is shut down and started again. As window tags are tied to windows, keeping them across StumpWM cold-restarts is a well-defined goal. To achieve it, window tags are kept as X11 string window properties independently of StumpWM. This functionality allows even to temporarily run another window manager and still have all the window tags ready after the return to StumpWM.

Frames are internal StumpWM objects, and their tags are currently stored inside StumpWM. Most known use patterns can be implemented with a fixed set of tagged groups, so the frames can be created and frame tags automatically set on StumpWM launch/restart.

While it would be possible to update the frame class definition and store tags in a special slot, current implementation uses a separate hash table to store frame tags.

5. FURTHER RELATED WORK

Some fixes are needed so that splitting a frame never tries to steal a window from another tagged group.

Current StumpWM behavior on split removal usually leads to resizing of all the frames on the physical display. While the most common case of removal of all splits inside the tagged group with a single command is implemented to preserve borders between tagged groups, there is still work to be done to handle all events correctly.

Full use of frame tagging and tagged groups in StumpWM requires overloading some StumpWM settings. It is likely that some reasonable default overloading function can be shipped with the frame-tagging implementation.

6. SOURCE LINKS

Window tags (without frame tags) in StumpWM contrib: `https://github.com/stumpwm/stumpwm-contrib/tree/master/util/windowtags`

Frame tags code and an example of configuration using frame tags: `http://mtn-host.prjek.net/viewmtn/stumpwm-tagging/branch/changes/com.ignorelist.401a0bf1.raskin.stumpwm-config`

7. REFERENCES

[1] StumpWM home page at GitHub,
`https://github.com/stumpwm/stumpwm`