Not so Common Desktop Environment (NsCDE) Manual

M. Z.

This manual describes version 2.0 of NsCDE.

This manual describes NsCDE: Not so Common Desktop Environment

1. Introduction

1.1. What is NsCDE?

In a nutshell, NsCDE is CDE visual clone built with more moden tools around FVWM window manager. Tehnically, it can also be considered a heavyweight FVWM theme enriched with additional free software tools and applications, combining all this components into something which can be called lightweight hybrid desktop environment. It can even be integrated into existing desktop environments with session management as a window manager wrapper for session handling and additional DE functionality.

NsCDE's main goal is to revive somewhat brutalist look and feel of the Common Desktop Environment found on many UNIX and unix-like systems during nineties and first decade of the 21 century, but with a more polished interface (XFT, unicode, dynamic changes, rich key and mouse bindings, workspace pages, rich menus etc) and a goal to produce comfortable "retro" environment which is not just a eye candy toy, but a real working environment for users who contrary to mainstream trends really like CDE, thus making semi-optimal blend of usability and compatibility with modern tools with look and feel which mainstream abadoned for some new fashion, and ... in a nutshell, giving to user the best of the both worlds.

Main driver behind NsCDE is the excellent FVWM window manager with it's endless options for customization, GUI Script engine, Colorsets, and modules. NsCDE is largely a wrapper around FVWM - something like a heavyweight theme, sort of. The other important pars are FvwmScript GUI in which CDE-like tools are written, GTK2 and GTK3 theme for unifying look and feel, which can also extend to Qt4 and Qt5 with gtk engine of Qt, X settings which are following the same theme rules, icon theme, python programs, converters and generators, korn shell scripts, CDE-compatible backdrops and palettes and misc pieces for integration like CSS theme for Firefox and Thunderbird, and stand alone integrated desktop-making parts like stalonetray, dunst, xsettingsd, xscreensaver etc.

1.2. Why NsCDE?

Since the 90-ties, I have always liked this environment and it's somewhat crude socrealistic look, described by some people as brutalism in achitecture. All that in a contrast to "modern" Windows and GNOME flat black/white interfaces and Google's "material" design - which are going in the opposite taste direction from what I always liked to see on my screen. I have created this environment for my own usage 8-10 years ago and it was a patchwork, chaotic and not well suited for sharing with someone. While it looked ok on the surface, behind it was a years of ad hoc hacks and senseless configurations and scripts, dysfunctional menus etc. Couple of months in a row I had a time and chance to rewrite this as a more consistent environment, first for myself, and during this process, idea came to do it even better, and put on the web for everyone else who may like this idea of CDE for 21st century. Year by year NsCDE has 3 years now. Many things are added, changed, hardened, debugged.

NsCDE is intended for a people which doesn't like "modern" hypes, interfaces that try to mimic Mac and Windows and reimplementing their (poor) ideas for non-technical user's desktops, and reimplementing them even more poorly. Older and mature system administrators, programmers and generally people from the Unix background are more likely to have attraction to NsCDE. It is probably not well suited for beginners.

Of course, question arises: why not simply use original original CDE now when it is open sourced?

Apart from desirable look, because it has it's own problems: it is a product from 90-ties, based on Motif and time has passed since then. In CDE there is no really smooth out of the box XFT font rendering, no immediate application dynamic changes, there are insecure RPC and obsolete tooltalk procedures even for starting local applications. Beside that, I have found dtwm, CDE's window manager inferior to FVWM and some 3rd party solutions which can be paired with it. So I wanted the best of the two worlds: good old retro look and feel from original CDE, but more flexible, modern and maintained "drivers" behind it, which will allow for individual customizations as one find's them fit for it's own amusement and usage. As it will be seen later, there are some small intentional differences between CDE and NsCDE - a middle line between trying to stay as close as possible to look of the CDE, but with more flexibility and functionality on the second and third look.

2. Components of the NsCDE

2.1. Components overview

NsCDE uses very extensive bunch of configurations, scripts and apps around FVWM. FVWM is in my opinion very good model of free choice for people who like to have things set up by their own wishes and who are aware what real freedom of choice is. A stunning contrast to policies forced on Linux an Unix users in the last decade from the mainstream desktop players.

NsCDE is not using FVWM default `$HOME/.fvwm` if exists, but sets `$FVWM_USERDIR` to `$HOME/.NsCDE`, and uses `$NSCDE_DATADIR/fvwm` and `$NSCDE_DATADIR/default` as a sources of configuration.

NsCDE configuration consists largely of files parsed by it's own scripts, and FVWM configuration which is grouped in separate thematic files. System FVWM configuration is initialized by `nscde` X startup wrapper which calls `Main.fvwmconf`. This file, as well as the rest of FVWM functions and code (except FvwmScript GUI programs) is located in `$NSCDE_DATADIR/fvwm`.

FVWM configuration sections are:

- Animate.fvwmconf - FvwmAnimate module configuration
- Backdrops.fvwmconf - Backdrops, read by FvwmBacker and FvwmPager
- Backer.fvwmconf - FvwmBacker module configuration
- Colorset.fvwmconf - FVWM Colorset definitions (generated for user in ~/.NsCDE)
- Event.fvwmconf - FvwmEvent module configuration
- FinishInit.fvwmconf - End of startap cleanup and safety last-second routines
- Font-XXXdpi.fvwmconf - Set of fonts ad InfoStore variables used across FVWM
- Form.fvwmconf - obsolete and not used by default, NsCDE does not use FvwmForm
- FrontPanel.fvwmconf - selector of FrontPanel configuration which is going to be read
- FrontPanel.fvwm2.fvwmconf - FrontPanel configuration for FVWM2
- FrontPanel.fvwm3.fvwmconf - FrontPanel configuration for FVWM3
- Functions.fvwmconf - almost all functions used by NsCDE FVWM setup, paired in logical sections
- IconMan.fvwmconf - Not used by default
- Ident.fvwmconf - FvwmIdent module configuration
- Init.fvwmconf - Fallback and example of session statup. Usually run from ~/.NsCDE
- Keybindings.fvwmconf - Static key bindings, without written menu hints
- LocalPager.fvwmconf - Single Workspace floating FvwmPager configuration
- Main.fvwmconf - Most important and early FVWM directives, environment and InfoStore variables, functions
- Menus.fvwmconf - All FVWM Menus used or generated for NsCDE
- Monitors.fvwmconf - Monitor and resolution static database as helper for some calculations
- Mousebindings.fvwmconf - All mouse actions with and without keyboard modifiers
- Sandbox.fvwmconf - not read by default, cleanup definitions for stripped down NsCDE sandbox mode
- Script.fvwmconf - FvwmScript module defaults
- Style.fvwmconf - All NsCDE FVWM Style directives, Menustyle, Cursorstyle, Decor addons etc
- Subpanels.fvwmconf - Default system configuration for subpanels of the FrontPanel, generated on install.

- WspLocPager.fvwmconf - Configuration of another local pager which uses FvwmPager, but for non-local workspaces
- NsCDE.conf - de-facto main user configuration file. All important variables are here.

It should be noted that FVWM configuration system is created in a very flexible manner. While there are only files of type "fvwmconf" in `$NSCDE_DATADIR/fvwm`, user's `$FVWM_USERDIR` or "~/.NsCDE" can contain this 3 types of FVWM configuration:

- Files with extension ".fvwmconf" are overriding defaults from the `$NSCDE_DATADIR/fvwm`. In other words, if `$NSCDE_USERDIR/Init.fvwmconf` exists, it will be read instead of `$NSCDE_DATADIR/fvwm/Init.fvwmconf`. For some parts of the FVWM configuration this makes a sense, for most it doesn't.
- Files with extension ".fvwmgen" are files which are generated by NsCDE tools and GUI programs. They can be managed by hand, but usage of tools will overwrite them. They counterparts in the `$NSCDE_DATADIR/fvwm` have extension ".fvwmconf", but if for example `$NSCDE_USERDIR/Colorset.fvwmgen` exits, it will be read instead of `$NSCDE_DATADIR/fvwm/Colorset.fvwmconf`. Putting static `$NSCDE_USERDIR/Colorset.fvwmconf` will be unusual, and will disable GUI tools and FVWM under NsCDE to read dynamically generated configurations.
- Third form of FVWM configuration in `$NSCDE_USERDIR` are ".fvwmlocal" files. This files will extend (append) configuration read from their ".fvwmconf" and/or ".fvwmgen" files. Good examples whould be Functions.fvwmlocal and Styles.fvwmlocal in `$NSCDE_USERDIR` to add local custom functions and per-application styles.

One notable exception to this system of configuration is the way FVWM Styles are read. In addition to reading `$FVWM_USERDIR/Style.fvwmlocal` "early" configuration in `$FVWM_USERDIR/NsCDE-Style.override` will be read from `$NSCDE_DATADIR/fvwm/Style.fvwmconf` just after applying general core style for all applications *(*)* but before reading custom application styles and prior to reading `$FVWM_USERDIR/Style.fvwmlocal`. This mechanism is intended for changing and overriding core style options which are not customizable by the Window Style Manager. Putting core *Style* options after NsCDE applications in the `$NSCDE_DATADIR/fvwm/Style.fvwmconf` are processed can nullify overrides for that particular applications, this is why this mechanism is provided. This file is not created in `$FVWM_USERDIR` by setup procedure or Style Managers, but it will be read if created manually.

In addition to FVWM configuration read by fvwm window manager and it's modules, there are some files parsed by NsCDE scripts and programs in `$NSCDE_DATADIR/defaults`.

This files are:

- AppMenus.conf - default per application addons for Window Operations menu on the 1st titlebar button
- FrontPanel.actions - read by FVWM functions called when FrontPanel controls and buttons are used
- Keymenu.actions - Part of the FVWM NsCDE keybindings which have menu hints written on menus
- Subpanels.actions - From this file, Subpanels.fvwmconf is generated.
- WSM.conf - Workspace Manager and Graphical Workspace Manager configuration file

All files above except WSM.conf if found in `$NSCDE_USERDIR` are read in addition to system defaults in `$NSCDE_DATADIR/defaults`. Local lines and statemets in user's copy are overriding system defaults. WSM.conf is read exclusively from `$NSCDE_USERDIR`. System file is only an example.

The rest of configuration is in `$NSCDE_USERDIR`. For example Dunst.conf, Stylonetray.conf, Xsettingsd.conf, NsCDE.radi (rofi), Xset.conf and Xdefaults. This files are either read by their programs which are called and configured/integrated by NsCDE, or by X server (Xset.conf, Xdefaults*). Various subdirectories contains additional data generated for the user or configured by the user. This parts of the configuration are described in more detail later in this document.

2.2. Applets and GUI Tools

NsCDE provides GUI tools which are built in FvwmScript(1) and their shell and python helpers. Also, some external applications that fit in the picture as recommended. This tools are mainly built for NsCDE from scratch, but some, such as mouse, keyboard and beep control are modified from the default FVWM scripts to look more CDEish and they implement some additional functionality.

Applets docks and panels are:

- Front Panel (FvwmButtons)
- Subpanels (FvwmButtons)
- Workspace Manager (WSM) - FvwmScript
- MonthDayApplet - FvwmScript
- Clock - External C applet pclock
- CheckMailApplet - FvwmScript
- FpLite - FvwmScript

GUI tools are:

- Style Manager (StlyeMgr) - FvwmScript
- Backdrop Style Manager (BackdropMgr) - FvwmScript + Korn Shell

- Beep Style Manager - FvwmScript
- Color Style Manager (ColorMgr) - FvwmScript
- ExecDialog - FvwmScript
- Font Style Manager (FontMgr) - FvwmScript + Korn Shell
- Keyboard Style Manager (KeyboardMgr) - FvwmScript
- Occupy Workspace (Occupy) - FvwmScript
- Occupy Page (Occupy) - FvwmScript
- Occupy Monitor (Occupy) - FvwmScript
- Graphical Workspace Manager (GWM) - FvwmScript
- Graphical Workspace Manager Options (GWMOptions) - FvwmScript
- Window Geometry Manager - FvwmScript
- Mouse Style Manager (PointerMgr) - FvwmScript
- Power Save Manager (PowerSaveMgr) - FvwmScript
- Subpanel Manager (SubpanelMgr) - FvwmScript
- Subpanel Settings (SubpanelSettings) - FvwmScript
- System Action Dialog (SysActionDialog) - FvwmScript, sudo
- Sysinfo - FvwmScript, python
- Window Style Manager (WindowMgr) - FvwmScript, sed, egrep
- Workspaces and Pages Style Manager (WsPgMgr) - FvwmScript
- NsCDE Process Manager (NProcMgr) - FvwmScript

Helper Dialogs:

- ActionForm - FvwmScript
- ChoiceForm - FvwmScript
- FilePicker - FvwmScript
- InputForm - FvwmScript
- WaitNotice - FvwmScript
- Splash - FvwmScript
- NColorsDialog (Color Style Manager part) - FvwmScript
- PaletteDialog - (Backdrop Style Manager part) - FvwmScript

External fit-in Programs:

- Xscreensaver (xscreensaver-demo called from StyleMgr) installed separately.

3. NsCDE Basics

3.1. Desktop Overview

NsCDE as desktop provides windows, workspaces, keybindings, controls, menus and the Front Panel.

- Windows contain software applications and are framed with controls so you can move them, resize them, or place them in additional workspaces.
- Keybindings, or keyboard shortuts enables you to call actions or manipulate windows without touching the mouse and immediately.
- Workspaces are the screen areas where application windows are placed when opened. We can consider workspaces as software or virtual screens.
- Controls enable you to manipulate objects, select choices, or type information.
- Menus provide access to commands used to manage windows and operate applications.
- The Front Panel is a collection of frequently used controls, available in every workspace.

Some of the windows, icons and objects you'll encounter as you use the desktop are:

- Window icons
- The Front Panel
- Style Manager
- File Manager
- Editor
- Tray

3.1.1. Window Icons

When you iconify a window, it becomes an icon on the workspace. To iconify a window, click the first of two buttons near the upper right corner of the window titlebar.

To deiconify the window, double-click mouse button 1 on the icon. You can also deiconify the window by clicking its icon to display the Window menu and then clicking on the menu's Deiconify item.

3.1.2. Front Panel

The Front Panel is a special window at the bottom of the display. It provides controls, indicators, and subpanels for quick overview and launch. The Front Panel also provides the workspace manager (WSM) for selecting a workspace. The clock, calendar, check mail, and FP Lite are examples of indicators. Arrow buttons over Front Panel controls are Subpanels launchers. Clicking an arrow button will open a Subpanel.

3.1.3. Style Manager

Use Style Manager to easily customize many elements of the desktop including:

- Colors, Backdrops
- Font size
- Keyboard, Mouse, Screensaver, Beep
- Screen Power, Window Behavior, Workspaces and Pages Behavior

To start Style Manager, click its icon on the Front Panel. Second right of the Workspace Manager.

3.1.4. File Manager and Editor

File Manager and Editor are the user's choice among great amount of them for X11 interface. While it is totally possible to run NsCDE without them, many people prefer to have them. Good choices for graphical File Manager are Krusader, PcmanFM, PcmanFM-qt, Spacefm and so on. PcmanFM and PcmanFM-qt can serve also as application managers, because they support *menu://applications/* pseudo-path which presents system catalog of installed GUI apps with their icons under their categories. Gvim is a good choice for editor, while some more lightweight include Mousepad, Nedit, Xfe ...

3.1.5. System Tray

System Tray functionality is provided by the stalonetray(1) application which is fully integrated with NsCDE and serves well for the purpose of hosting tray icons added by the user, user's XDG autostart files by dex(1), or by the programs. By default, stalonetray is located in lower right corner of the screen, and it is configured in `$FVWM_USERDIR/Stalonetray.conf`.

3.2. Working with Window controls

This section describes how to start applications in NsCDE, move them around, iconify, shade, resize, maximize, close and so on.

3.2.1. Window Frame Controls

A window's frame provides controls with which you can:

- Focus attention on a window, making it active for input from mouse or keyboard. When window is selected and it becomes active, its titlebar changes color into a active window color.
- Move a window.
- Turn the window into a icon (iconify).
- Size a window.
- Close a window, removing it from the desktop.
- Migrate open window into other workspaces.

The leftmost button on the window titlebar is menu button. Click the Window menu button to display the Window Operations Menu.

To close a window and remove it from the desktop, double-click the Window menu button or display the Window menu and click on Close. If this doesn't work for some reason, you can try to forcibly remove a window with Forcefully Close.

Right of the window title, there are iconify and maximize buttons. Iconify is the first one, while maximize is the rightmost at the corner. Click the iconify button to turn the window into a window icon. To restore a window from its icon, double-click the icon.

Click the maximize button to display the window in maximized state. To ignore area of the Front Panel, and really maximize a window to the whole screen, double-click the button. Window can be turned into its former size by clicking the maximize button again.

The title bar shows the name of the application that owns the window. Select a window by clicking its title bar. You can move a window by dragging its title bar. Clicking title bar with the third mouse button will lower or raise it below or above another overlapping window, second (middle) mouse button gives you a quick diagnostic and screenshot options menu.

Click and drag a window's resize corner or border to resize the window in any direction.

3.2.2. Opening Application Windows

There are couple ways to open applications in NsCDE. NsCDE offers a handy way to group often used applications and other applications in a more or less accessible desktop controls.

- Front Panel: big icons of the Front Panel offer easiest and most visible access to applications. By

default,it contains access to editor, office application, terminal, Style Manager, Printer Manager etc. This list can be customized to suit individual needs.

- Subpanels of the Front Panel are by category grouped popup menus of applications. Often used, important and popular applications are here, and can be put here by the user, to suit the needs.
- Workspace Menu: called with third mouse button on the root window (screen backdrop). It contains submenu Applications which is generated by gathering system GUI applications into categorized submenus. Almost every application is accessible from this menu. There is also a Quick Menu submenu of the Workspace Menu, where user can put it' picks of applications for ease of access.
- Exec dialog, called with Alt+F12, or rofi(1) launcher if it is installed and configured to work with NsCDE. Here, the name of the command which invokes application can be typed. Return will execute a command and window will appear.
- By using some file manager which supports *menu://application/* pseudo path, that is, it can serve as Application Manager. PcmanFM is a good choice here.
- By typing application invocation commands into terminal emulator.

3.3. Using Workspaces

Workspaces are like separate screens or windows. To help organize the desktop, user can have separate workspaces for different groups of related tasks. For example, a workspace named Reports could contain the tools you use to prepare reports, such as a spreadsheet, Libreoffice text, and graphics applications. You might set up other workspaces according to your projects. By default, NsCDE offers four workspaces named "One", "Two", "Three" and "Four".

This section describes how to display workspaces, place windows in specific workspaces, and rename workspaces.

3.3.1. To Display Another Workspace

3.3.1.1. Mouse

Click the workspace's button in the Workspace Manager on the Front Panel.

3.3.1.2. Keyboard

Press Ctrl+Escape twice: first time to focus Front Panel, then second time to get mouse pointer into the middle of Workspace Manager. Press the TAB key to move between WSM buttons. When pointer is above workspace you want to display. Press Return.

3.3.2. Relocating a Window into Other Workspaces

3.3.2.1. Mouse

Click the titlebar menu button to display Window Operations menu. Choose "Occupy Workspace" from the menu. In the Workspaces list of the "Occupy Workspace" dialog select the workspace in which you want the window to appear. To place the window in every workspace, click the All Workspaces check box. Click OK. Action for moving window between pages and monitors is the same. If Occupy dialog was called as "Occupy Workspace", mode of operation can be looped to "Occupy Page" or "Occupy Monitor" by pressing it's radio button left of the list of workspace, page or monitor objects.

3.3.2.2. Keyboard

To display the Window Operations Menu, press Meta+Alt+M. Press the Down Arrow key to select "Occupy Workspace", then press Return. The Occupy Workspace dialog box appears. With Up or Down Arrow, select the workspace in which you want the window to appear. Press Return.

3.3.3. Renaming Workspaces

3.3.3.1. Mouse

Click the Front Panel button of the workspace whose name you want to change with third mouse button to display button's contextual menu. Click on Rename .... The button becomes a text field. Edit the workspace's name in the text field. Once the workspace is renamed, press Ctrl+Return on the keyboard to convert text field back to button.

3.3.3.2. Keyboard

Press Ctrl+Escape twice: first time to focus Front Panel, then second time to get mouse pointer into the middle of Workspace Manager. Press the TAB key to position pointer above workspace intended for renaming. Press Return to select (and change) the workspace. Press the Press the Ctrl+Space. The button becomes a text field. Edit the workspace's name in the text field. Once the workspace is renamed, press Ctrl+Return to convert text field back into button.

4. Applets Docks and Panels

4.1. Front Panel

In NsCDE, CDE Front Panel is re-implemented with the help of FvwmButtons(1) Configuration is stored under alias ***FrontPanel*** in `FrontPanel.fvwm2.fvwmconf` and `FrontPanel.fvwm3.fvwmconf`, read and activated from the `Main.fvwmconf`. Visually, this remake of Front Panel matches almost in a pixel CDE Front Panel. The main differences are:

- Since FvwmButtons doesn't implement drag and drop protocol, there is no possibility to install icons by dragging them from file manager or from one to another position. However, icons (with their applications) on the Front Panel can be customized by clicking any item on the subpanel above it with the right mouse button. Dinamic contextual menu with the name of the application will appear and action *Copy to Main Panel* can be choosen. This will edit file `FrontPanel.actions` in user's `$FVWM_USERDIR` and put appropriate custom launcher. All Front Panel launchers can be customized, including the ones with applets (clock, calendar, mail) instead of static icons. One notable inconvinience here is that subpanel must be enabled and icon installed on that subpanel before it can be copied to the main panel. If one does want to customize icon launcher without having subpanel above it, subpanel can be enabled temporary, icon installed on it, and copied to the main panel, and subpanel above icon disabled afterwards. File `FrontPanel.actions` can also be edited manually as an alternative (caution should be taken) to achieve the same effect. Manual editing is mandatory when one wants to install new applets instead of static icons, because this cannot be done with gui menu actions. See FrontPanel.actions for examples. The other way (full control) is by copying `FrontPanel.actions` configuration file from `$NSCDE_ROOT/config` to `$FVWM_USERDIR`.
- In the original CDE, Front Panel is part of the dtwm Window Manager binary, while in NsCDE it is configuration of FvwmButtons(1) FVWM module. Workspace Manager in the middle of the Front Panel is a separate applet written in FvwmScript(1).
- On every icon, for the first two mouse buttons different action can be assigned. This is used for example 7th icon where mouse button 1 calls Style Manager, while mouse button 2 calls Backdrop Style Manager directly as a quick shortcut. Second (middle) mouse button action can be considered as hidden hack for advanced usage.
- Mouse button 3 on any of the default 10 Front Panel icons brings contextual pop-up menu titled by the main action from mouse button 1. Action on this menu are: a) call main application as if icon has been clicked with the 1st mouse button, b) "reset this button" which will remove user's customizations for that launcher button from the `$FVWM_USERDIR/FrontPanel.actions` and load default from the `$NSCDE_DATADIR/defaults/FrontPanel.actions`., c) "Reset Subpanel" will reset upper subpanel to it's default value (remove definition of the numbered subpanel from the `$FVWM_USERDIR/Subpanels.actions`), d) "Delete Subpanel" will disable subpanel for above this icon (without resetting user's configuration for it), and e) Help will call this documentation.
- In addition to iconification, Front Panel can be shaded to the bottom edge of the screen with Shift-Esc action, and put back in it's place with the same key binding when it is called again. Shade direction of the Front Panel is the opposite one from ordinary windows - they are rolled from the bottom up.
- Iconification is by default to bottom right screen edge, leaving last 96px space to the right for Stalonetray the while all other programs are by default iconified in the top left edge as in CDE.

- NsCDE Front Panel is flexible. It can be overlapped with programs, moves away (lower) for fully maximized windows and while pretty much thick, it is not in the user's way on the screen.
- Front Panel has it's own menu on the top left button and special context menu if this button is clicked with right mouse button. Middle mouse button behaves as if title bar of a any normal window is clicked - with special diagnostic tool menu. Right-clicked special menu has this important tasks:
  - Calls Workspace and Page Manager Manager
  - Restart WorkSpace Manager
  - Restart Page Manager
  - Number of Launchers ... (submenu)
  - Restart Panel Clock (pclock)
  - Restart Panel Mail Applet
  - Restart Panel Date (MonthDayApplet)
  - Restart Panel Lite (FpLite)
  - Restart the (whole) Front Panel
  - Help

  This menu can also be called from the customized Window Options Menu which appears if menu button is clicked with the left mouse button. Item *Front Panel Controls* will replace Window Options Menu with Front Panel Controls Menu.
- As FvwmButtons based dock, by default it swallows the following applets:
  - pclock (external standalone app with CDEish skin)
  - MonthDayApplet (FvwmScript)
  - CheckMailApplet (FvwmScript)
  - WSM / Workspace Manager (FvwmScript)
  - FpLite (FvwmScript)
- Third icon expects `$[infostore.filemgr]` to be defined or discovered and automatically filled if not defined.
- Fourth icon will call `$[infostore.terminal]` which must be defined or it is discovered.
- Sixth icon tries to discover system print manager. Currently, only system-config-printer and $[infostore.browser] http://localhost:631 are discovered, but this can be set manually by putting InfoStore variable `$[infostore.printmgr]` in `~/.NsCDE/NsCDE.conf`. Good place for personal custom icon and redefinition if printing really doesn't that much to the user. `FrontPanel.actions`.
- Seventh: NsCDE Style Manager from which all other Style Managers can be called.
- `$[infostore.xeditor]` - if defined in `$FVWM_USERDIR/NsCDE.conf`. If not defined, NsCDE will try to find some usable app and autofill this value on startup.
- Semi-empty. By default, it popups 9th subpanel if pressed. A nice idea is to call pavucontrol or some audio mixer on 3rd mouse click from `FrontPanel.actions`.
- Help, documentation.

- Front Panel Subpanels 2, 5, 6 and 8 are empty by default, but they can be activated with a triple middle pointer click on an empty place, or with a contextual menu on the icon below it on the Front Panel, and selecting a menu item "Add Subpanel". In a former case, dialog will ask user if he wants subpanel to be enabled, while in later case, subpanel will be enabled immediately. This is specially useful for subpanel 5 which will show `$[infostore.mailreader]` if it is defined or autodiscovered.
  - Subpanel 1: Applications
  - Subpanel 3: Libre Office Components and various office/productivity tools (PS/PDF reades etc ...)
  - Subpanel 4: Tools
  - Subpanel 7: NsCDE Style Manager and various Qt, Gtk and misc desktop management applications
  - Subpanel 9: Multimedia programs: audio, video, photo ...
  - Subpanel 10: Documentation
- In the center of the Front Panel there is a place without subpanel launchers and separated by vertical line. Inside this area, there are 4 small command icons: Left: Lock Screen (**xscreensaver -activate**), Page Manager Menu, Right are Front Panel Lite (system load indicator), and Exit button (SysActionDialog).

  Lock Screen icon (upper left) has a contextual menu invoked by the 3rd mouse button click, which allows user to suspend and resume background activity of the xscreensaver(1) in addition to the locking mechanism. Help menu entry is also present. Mouse button 2 brings Xscreensaver preferences dialog as if it was called from the Style Manager.

  Page Manager Menu icon has a contextual menu invoked by the 3rd mouse button click. In addition to returning back to main menu, it can call visual local FVWM pager LocalPager. Help menu entry is present too. Mouse button 2 brings Workspaces and Pages Style Manager configuration.

  Front Panel Lite (upper right) has a small contextual menu which calls main action and help.

  Exit button (lower right) has a small contextual menu which calls main action and help.

  In the middle of this area there is WSM - Work Space Manager with well known buttons for navigating NsCDE workspaces. By default, four workspaces are shown and configured, but this can be changed (see Section 4.3).

4.1.1. Customizing number of Front Panel launchers

NsCDE Front Panel can be customized to contain from zero to 20 Front Panel buttons or launchers. Default is 10 (5 launchers left and 5 launchers right from the Workspace Manager). There can be even or unequal number of launchers on any side of the Front Panel. In that way, it is possible to:

- Extend number of launchers (buttons)

- Move Workspace Manager to left or right
- Reduce number of launchers
- Extend or reduce number of subpanels on active launchers
- All extended (more than 5 on each side) launchers can be customized with icons, subpanels and swallowed applets.

This is very flexible scheme, and can scale from minimal two-workspaces configuration without any launcher (just left and right handlers, and Workspace Manager with Lock, FpLite, Exit and Page Manager in the middle), up to configuration with 20 launchers freely arranged left and right with up to eight workspaces. For this last one, Full HD resolution, or at least 1680x1050 is needed.

To change number of launchers left or right of the Workspace Manager, leftmost menu button of the Front Panel must be selected, which will bring Window Options Menu, then *Front Panel Controls* can be selected to replace Window Options Menu. Front Panel Controls can be directly called by clicking this same button with third (right) mouse button. Next, *Number of Launchers ...* submenu must be opened.

Actions are:

- Add Left Launcher
- Add Right Launcher
- Remove Left Launcher
- Remove Right Launcher
- Help

Once a new launcher is on the Front Panel, it can be customized by the means of enabling (temporary or permanently) subpanel on which icons can be installed, or by editing configuration `$FVWM_USERDIR/FrontPanel.actions` and restarting the Front Panel. Under the hood, adding or removing launchers on the Front Panel is actually editing FVWM InfoStore variables `FP.LeftLaunchersNum` and `FP.RightLaunchersNum` in the `$FVWM_USERDIR/NsCDE.conf` which are both by default 5. Front Panel geometry is automatically recalculated on restart.

4.2. Subpanels

NsCDE subpanels are simple transient FvwmButtons based docks. As the Front Panel itself, they are also as much as possible similar to original CDE forms of the same purpose. Some applications in them are predefined, and discovered if installed, and the rest is up to user to populate. Their purpose is not to show all possible GUI applications installed on the system as right-clicked Workspace Menu does. They are meant for favorite, important and often used programs.

There is one big difference between CDE and NsCDE subpanels: *Install Icon* is not (and cannot be due to FvwmButtons not implementing) drag-and-drop destionation. Click action on *Install Icon* calls special NsCDE tool for installing and defining items on the subpanel. There is no drag and drop off items from

the applicaton manager (the later also doesn't exist, but it can be used as menu://applications/ pseudo path in some file managers).

On every subpanel, last selected entry will remain hilighted next time subpanel is popped up again.

Each Subpanel's application item or entry has additional third mouse pointer button action which will pop up dinamically populated contextual menu named after item's title. Here, there are actions to move up or down item for one space on the subpanel, to move item to the beginning or the end of the subpanel's application list, as well as to delete item from the subpanel (warning message will appear before deletion is finally performed). First menu item is the same as menu name: title of the application from submenu. If clicked, it will perform default action as if item's title or icon was clicked on the submenu itself. This is a kind of a escape from the contextual menu, but to still use subpanel's default action without repeating opening of a submenu again. If move or delete action is silently not performed, this is most likely the situation where user's `$FVWM_USERDIR/Subpanels.actions` is newer than `$FVWM_USERDIR/Subpanels.fvwmconf`, and must be rebuilt for configuration actions to take a place properly. In that case, repeated operation must succeed, otherwise, see X session error log for details.

Subpanels, like in CDE has titlebars but as windows on screen they are without borders and handles. They have only left menu button like other windows, but with one exception: there are no actions for closing window and re-positioning it (no sense in this), but they have "Refresh Subpanel" and "Subpanel Settings" controls. First one kills process module, re-reads it's configuration and starts it on the next click on Front Panel subpanel launcher. Subpanel Settings is the small and simple FvwmScript tool which allows one to rename Subpanel, set it's width for application titles to fit if necessary, and to enable or disable that particular Subpanel.

On the first change with *Install Icon* or *Subpanel Settings*, non-FVWM configuration file *Subpanels.actions* will be created in the `$FVWM_USERDIR`, from there, all subpanels changed from default will be read by the **$NSCDE_TOOLSDIR/generate_subpanels**, while the rest will be generated and combined/mixed from the `$NSCDE_DATADIR/defaults/Subpanels.actions`. This file can also be edited by hand (ok, not by hand, but editor will suffice) and the result may be applied by calling **generate_subpanels** and then `f_ReadCfg Subpanels`. Generated files will be called `Subpanels.fvwm2.fvwmconf` and `Subpanels.fvwm3.fvwmconf` with the same content, but a slightly different syntax, one of this files will be read as subpanels configuration depending under which version of FVWM NsCDE is running. This files are expected to appear in the `$FVWM_USERDIR`. If not found there, `$NSCDE_ROOT/config/Subpanels.fvwmX.fvwmconf` will be read instead. Syntax of the Subpanels.actions is explained in the Configuration files explained section.

Presently, there is one workaround here under FVWM2: as much as FVWM, and specially FvwmButtons are very flexible and configurable, no title for the buttons app can be set apart from module alias, but module alias doesn't support names with spaces in them. Internal names as *"NsCDE-SubpanelX"* are for that reason referenced in `FrontPanel.fvwmX.fvwmconf`, and are internally mandatory names of their Subpanels. Because there is no configuration option in FVWM2 for subpanel to set Window Title, we are using tool **xdotool(1)** which is run on Subpanel initialization with a delay of 3,2 seconds (internal workaround for xdotool) and this then takes infostore variable `NsCDE-SubpanelX-Name` and sets literal, system default or user picked name of the subpanel. Presently, an alternative option is to apply

FvwmButtons patch present in the patches subdirectory of NsCDE tarball which introduces a *WindowName* option for FVWM2 to it's configuration, or even better, to use new FVWM3 which has this and many other handy options and capabilities already in place.

The rest about Subpanel, or to say their visible outfit, and their main function are the same as in CDE - a nice, heavyweight and elegant application launchers.

4.3. Workspace Manager

Workspace Manager is a Widget in the center of the Front Panel. Visually, it replicates in almost a pixel similarity with the center of the CDE's Workspace Manager's buttons, but with a few exceptions beneath the surface and further configuration:

- There is a limited number for possible of workspaces. There can be no odd number of workspaces, and combinations are: 2, 4, 6 and 8. Default is of course 4.
- If InfoStore variable wsm.eco is defined as *"InfoStoreAdd wsm.eco 1"* in user's profile `$FVWM_USERDIR/NsCDE.conf`, WSM buttons will not be fixed in width as in CDE. In that case, Workspace Manager will not extend a width of the Front Panel, rather buttons will be more wide in 2-buttons combination, and more narrow in 6 and specially 8 button combination. By default, buttons are all of the same width and they are extending or narrowing the cummulative Front Panel width size if changed from default 4 workspaces to 2, 6 or 8 workspaces.
- As in CDE, workspace names can be renamed. There is a keyboard combination in FvwmScript WSM which enters rename mode: Ctrl+Space on the current active workspace button, while pointer is above it. Then, Ctrl+Enter saves a new name. Names are synchronized with FVWM desktop names and used in the rest of the configuration. Names are saved in `$FVWM_USERDIR/WSM.conf`, and EWMH desktop names are applied immediately in a existing X11 NsCDE session. Workspaces can also enter rename mode if their workspace buttons are clicked with 3rd mouse button and Rename ... option choosen from the popup menu.
- If Sun type keyboard is in use, Help key above WSM will bring this help text in browser. On PC keyboard, F1 has the same function.
- Addition: right mouse button on workspace button brings contextual menu. From this menu, the following items can be selected: default acion (activate workspace), Rename ... to rename the workspace, Manage ... to call Workspace and Page Style Manager,Popup menu Go to Page ..., Local Pager (visual FvwmPager) (for a workspace, not necessarily the current one), then the option Windows ... to bring up Window List, with windows on that workspace, and last, a small submenu Go to Page ... for changing the active page on the that workspace, which will of course change active workspace to that where page was selected.
- Number of Workspaces and Pages can be configured with Workspace and Pages Style Manager tool.

State of the buttons is synchronized by FVWM function called from FvwmEvent(1) module whenever desks and pages are changed by other means, such as keyboard shortcut, mouse move, or by issuing direct commands to **FvwmCommand(1)**.

In 4-color palette mode, all WSM buttons are of the same color, while in 8-colors mode, there are four color variations from the given palette.

Workspaces in NsCDE are named and numbered from 1, while in FVWM (where they are called desks), they start from 0. This fact required additional effort while coding FvwmScripts and making core configurations.

4.4. Page Manager

Page Manager is entirely new thing. There is no concept of *Pages* in the original CDE, just workspaces (FVWM: desks). A nice FVWM feature of pages is too useful to be disabled and sacrificed just for the sake of even more CDE similarity, but really zealous user can configure NsCDE not to use pages, just workspaces. That is, 1 page per workspace. PGM is a dynamic FvwmButtons Icon southwest from the WSM, represented by the silver-gray icon of the workspace divided on *pagesX* x *pagesY*. Default is 2x2, that is, four pages per every workspace, which in default setup gives 4 x 4 = 16 screens for applications. Minimum for pages is 0, and maximum 16. For example, in maximal workspaces+pages configuration, one gets 8x16 desktop, that is 128 screens! While undoubtedly this is diversion from a even more flexible plain FVWM configuration, it covers really great number of possible preferences. Pages can be configured in any *XxY* combination. For example 1x4, 2x3, 3x3, 2x1, 2x2, 3x2 ...

Page Manager icon changes with the page to represent position of the current page on current workspace while user moves from page to page automatically with the help of the FvwmEvent(1) just like the Workspace Manager (WSM).

When clicked, popups menu with names of the pages. When option from the menu is selected, menu pops down and active pointer screen is changed to the selected page. Second pointer button calls Workspace and Page Manager, while the third pointer button calls pages popup menu.

4.5. MonthDayApplet

Familiar CDE icon with month and day of the month in it. Simple applet which calls empty, do-nothing (by default) function `f_Calendar`. This function can be overridden in `$FVWM_USERDIR/Functions.fvwmlocal` to call a program configured by the user.

If Sun type keyboard is in use, Help key above Month Day Applet will bring this help text in browser. On PC keyboard, F1 has the same function.

4.6. Front Panel Clock - fpclock

This is a small C program (GPL) written many years ago by Alexander Kourakos. It supports XPM skins and displays hours, minutes and seconds. It is well suited for window manager docks like

FvwmButtons(1). In NsCDE it is applied with a skin similar to original one, but slightly bigger and with more clear edges and colors. Default can be used from `$NSCDE_DATADIR/icons/NsCDE` or even replaced with a Solaris version with picture of the globe with red hands for hours and minutes and white for seconds. If clicked, it will try to execute firefox by default (which appears doesn't work if Num Lock is active on the keyboard).

4.7. Check Mail Applet

Fifth icon from the left on Front Panel is FvwmScript applet. It is calling `f_CheckMail` FVWM function if clicked. By default `f_CheckMail` is calling `$[infostore.mailreader]`, which if defined by user or discovery, will bring up mail application. Up to user is to redefine this variable in his `$FVWM_USERDIR/Functions.fvwmlocal` to suit the needs for mail checking. To be clear, by default, it is not functional as an new mail notification applet. External applications can use it for this by calling **nscde_fvwmclnt** to send it a signal. The syntax is:

```
nscde_fvwmclnt "SendToModule CheckMailApplet SendString 1 2 \
 LetterUp <accnum> <accname> <fontsize> <color>"
```

Where:

- *nscde_fvwmclnt* is NsCDE wrapper for FVWM2 or FVWM3 FvwmCommand sender
- "SendToModule CheckMailApplet SendString 1 2" will send directions to widget 1, subroutine 2 of the CheckMailApplet FvwmScript module
- "accnum" is a number identifying a separate mail account or entity, one per name, 10 max
- "accname" is 3 letters identifying account, for example "PTN" for proton
- "fontsize" can send font size hint to CheckMailApplet for displaying notification info
- "color" is a hex or X11 color name hint to CheckMailApplet for displaying notification info

This command can be used by mail programs to notify Front Panel Mail icon to change from closed letter to open letter when new mail arrives. For any new mail, counter for named and numbered account will increase by 1, until icon is clicked by the user. For example, Thunderbird has a "mailbox alert" extension that can use scripting way of sending notifications.

Above this applet, there is an empty space for subpanel launcher which can be activated with middle triple click, and it will present *Mail Reader* if thunderbird, claws-mail, mutt or some other popular and known mail application is installed. User can use *Install Icon* action to change or add entries on this menu. For example, to call **urxvt -e** mutt or something like that.

Key F1 will bring this help text in browser. If Sun type keyboard is in use, Help key above Check Mail Applet has the same function as F1 on PC.

4.8. FpLite

Load Indicator on the top right side of the center of the Front Panel contains a small applet called FpLite. In original CDE it was used to indicate desktop activity, but since on today's processors this tasks are short and almost immediate (specially with a good window manager such as FVWM), It appears to be better suited to server as system load indicator.

It has 10 micro-bars. When there is no load, all are yellow. Load grows from left to right. First 5 green bars, then 3 blue, 2 magenta, an after that it starts from the beginning with red bars. FpLite summarizes load of all CPUs on the system in a way that 1-minute load is divided with number of CPU cores, and then counted as such while displaying load with color micro-bars. Everything under 1 (internally 100) is yellow, green, blue and magenta, and after that it counts 10 red micro-bars. For example: on the system with 2 CPU cores, 1-minute load of 0.6 will be presented with 3 bars (0.6 / num-cores), load of 2.2 will be presented with one red bar etc ... on the system with four CPU cores load of 3 will be magenta on the two rightmost bars, and load of 4 or more will be red. Load of more than (numcpu * 10) will not be shown specially, but user gets an idea what is going on if FpLite is all red.

If clicked, it will call a function `f_FpLiteClickAction` which is by default set to `$[infostore.taskmgr]` or autodiscovery during starup. If nothing else, it will call *[default terminal app]* **-e top** on Linux and BSD systems, *[default terminal app]* **-e prstat** on Solaris and it's clones, and *[default terminal app]* **-e topas** on AIX. FpLite FvwmScript app uses little portable python script *getla1* from the `$NSCDE_TOOLSDIR/libexec` to obtain 1-minute load data.

Key F1 will bring this help text in browser. If Sun type keyboard is in use, Help key above FpLite has the same function as F1 on PC.

5. GUI Tools

5.1. Style Manager

This Window is a starting point for all other *Style Manager* applications to be called. It is called from 7th button on the Front Panel. It has big icons for calling:

- Color Style Manager
- Font Style Manager
- Backdrop Style Manager
- Keyboard Style Manager

- Mouse Style Manager
- Beep Style Manager
- Xscreensaver Demo (setup)
- Window Style Manager
- Workspaces and Pages Style Manager
- Power Style Manager
- Startup Style Manager

If NsCDE was started by some X Session Manager, Startup Style Manager icon will call setup tool for that session manager or DE if it is known to Style Manager. Otherwise, error message is displayed: either that NsCDE was not started under X Session Management, or X Session Manager is not recognized, and it's setup tool cannot be run. Currently, only MATE, LXDE, KDE and GNOME session managers are recognized and their respective tools called. See Section 12 about running NsCDE under X Session Manager for more information about this matter.

Key Bindings:

- Ctrl+Q: Quits Style Manager.
- Sun Help and F1: Displays this help text.
- C: Opens Color Style Manager
- F: Opens Font Style Manager
- B: Opens Backdrop Style Manager
- K: Opens Keyboard Style Manager
- M: Opens Mouse Style Manager
- E: Opens Beep Style Manager
- S: Opens Screen Style Manager
- W: Opens Window Style Manager
- P: Opens Power Style Manager
- P: Opens Workspaces and Pages Style Manager
- T: Opens Session Style Manager

5.2. Backdrop Style Manager

Part of the Style Managers which can be called from the main Style Manager (7th button on the Front Panel). This is the clone of the same-named CDE tool. It loads backdrops from the `$NSCDE_DATADIR/backdrops` and `$FVWM_USERDIR/backdrops` (if any). From XPM backdrop

templates with symbol names (with .pm extension) it will generate previews in user's `$FVWM_USERDIR/tmp` and if applied or OK'ed, will set permanent backdrop in `$FVWM_USERDIR/backer`. Backer is named after FvwmBacker(1) module which then loads this X Pixmap as numbered FVWM Colorset from the `$FVWM_USERDIR/Backdrops.fvwmgen` which will be written by Backdrop Style Manager (or by hand). Backdrops are generated in the colors of the current color theme from the active palette (*Broica* by default). It has different colors for a group of every four desktops in 8-colors mode and the same base color in 4-color mode. Generated backdrop in the `$FVWM_USERDIR/backer` are named `DeskN-<name-of-the-backdrop>.pm` where *N* is the workspace (desk) number from 1-8. In such a way it is possible to have the same backdrop pattern on more than one workspace in 8-colors mode.

In case of NsCDE setup with FVWM3 and *per-monitor* or *shared* X RandR monitor model (fvwm3 DesktopConfiguration), backdrops in `$FVWM_USERDIR/backer` will be used as a source for procedure which builds joined tiled backdrops for multiple monitors. Output of this backdrops, which is then loaded directly by FVWM function is placed in `$FVWM_USERDIR/backer/bgcache` subdirectory.

There is an option to use the same backdrop for all desks too. User can add and delete custom backdrops in `$FVWM_USERDIR/backdrops`. First action with Add button and file picker, and second action with Delete button when particular backdrop from the list on the right of the preview is selected. Delete action will fail for system-pathed backdrops with appropriate error message, while both actions will reload list of backdrops immediately. Apart from doing this, NsCDE Backdrop Style Manager has couple of features more than CDE original:

- In 8-color mode, user can select another color variant instead of default for the current workspace from the popup menu. There are four variants.
- Custom palette can be loaded instead of default one, and backdrops can be set with colors from that palette. In 8-colors mode, there is even more possibility because colors can freely change their associated workspace.
- Instead of backdrops, user can opt for a background image. If option "Use photo or picture" is selected, list of backdrops will dissapear and image backgrounds (so called "wallpapers") of PNG and XPM type will be loaded from `$NSCDE_DATADIR/photos` and from the `$FVWM_USERDIR/photos` (if any). Option to use one photo for all workspaces exists too. In this mode, backdrop-specific options will be hidden until Use photo or picture is not deselected. Add and Delete of photos/pictures is supported in a same way as for backdrops. Photos must be in PNG or XPM format to be loaded. This means that images of that type can be added to `$FVWM_USERDIR/photos` but if some other known format is added via Add button of Backdrop Style Manager, like JPG, TIFF or GIF, it will be converted to PNG on the fly. Pictures can be loaded as FVWM Colorset definition "Pixmap", "AspectPixmap" and "TiledPixmap". Default is "AspectPixmap" (in contrast to backdrops which are always TiledPixmap). This 3 options can be choosen on the popup menu.

Key Bindings:

- Ctrl+Q: Quits Backdrop Style Manager.
- Sun Help and F1: Displays this help text.
- P: Applies preview of the currently selected backdrop (or photo) on the root window.

- Up/Down: Selects previous or next element on the backdrop (or photo) list.

5.2.1. Load Custom Palette Dialog

A Backdrop Style Manager part

Helper dialog which provides a list of palettes to the Backdrop Style Manager when user wants to use color schemes from another palette from currently used in user's setup. This is NsCDE addon functionality, not present in original CDE. Additionally, background variants from custom palettes can be used too as from the default user's palette in 8-color mode, which is also NsCDE feature not present in original CDE.

Key Bindings:

- Escape: Quits Dialog.
- Sun Help and F1: Displays this help text.

5.3. Beep Style Manager

Simple tool to adjust system beep device - if it is present as device and if desired/enabled. This tool uses xset(1) *b* command to set volume, pitch and duration of the beep sound. Modified setting can be tested with additional Beep button which is not present in the original tool, and also dynamically applied with Apply button. Save button will save `$FVWM_USERDIR/Xset.conf` with other xset(1) options which are executed during NsCDE startup.

Changing Beep Options

I. Open Beep Style Manager Beep from the main Style Manager window.

II. Select the settings you prefer:

- Volume: Determines the volume of the beep (off at 0%)
- Pitch (Hz): Determines the pitch of the beep from 82 to 9000 Hz
- Duration: Determines the time length of the beep sound
- Beep Button (upper left) test the current settings

III. To save the changes, click Save. To return to the default settings, click Default button at upper right corner of the window.

Key Bindings:

- Escape: Quits Beep Style Manager.
- Ctrl+Q: Quits Beep Style Manager.
- Sun Help and F1: Displays this help text.

5.4. Color Style Manager

With Backdrop Style Manager, this is probably the most important theme tool in (Ns)CDE. This tool applies colors to the widgets, menus, applications and backdrops. As in CDE, it reads color information from the palette files in `$NSCDE_DATADIR/palettes` and `$FVWM_USERDIR/palettes`. Palettes are the 16bpp color definitions (8 of them). This colors and border bg/fg/sel colors calculated from them are the base of the look of pretty much all of the things on the screen. Colors can be applied in 4 or 8 colors mode. Most notable palettes are *Broica* in 8-colors mode and *Solyaris* (called *Default* on SunOS) in 4 colors mode. Color mode can be selected with the Number Of Colors dialog which opens when the same named button in the main dialog of Color Style Manager is pressed.

Color Buttons and the Number of Colors selection determine the number of colors that make up a palette. You will have four or eight colors in the Color dialog box. The different color buttons in the palette control the colors used for different areas of the screen.

If you have eight color buttons, they will be used as follows (buttons numbered from left to right, top to bottom):

I. Active window borders, corners and titlebars

II. Inactive window borders and non-application menus

III. Workspace Manager (WSM) Switch for workspace 1 and 5

IV. Text, text field, list areas and text areas

V. Main window background and Workspace 2 and 6 button background

VI. Dialog box background, menu bar, popup menu and background of workspace 3 and 7

VII. Workspace 4 and 8 background

VIII. Front Panel background when "Color 8 for Front Panel and Icons" is selected in the "Number of Colors ..." subdialog.

If you have four color buttons, they will be used as follows (buttons numbered from left to right):

I. Active window borders

II. Window bodies:

    a. Inactive window borders and non-application menus

b. Main window and dialog box backgrounds and menu bar

c. Front Panel background

III. Workspace switches and backgrounds

IV. Text, text field, list areas and text areas

Color Style Manager as most tools is written in FvwmScript with background shell helper and color calculation and generator routines. Visually it tries to be as much as possible similar to the original CDE, but since it has some new features, there are some new buttons and commands introduced. Tool has a list of the palettes (system + user), preview button which can temporary apply some palette on the current workspace backdrop and FVWM based applications (FrontPanel, other scripts ...)

As in Backdrop Style Manager there are Add and Delete button actions. System palettes cannot be deleted, while local can be added to `$FVWM_USERDIR/palettes` and applied immediately.

Importing a ready-made new custom Palette: you can import your own palette

I. Open the Color Style Manager.

II. Choose Add ... in the Color dialog box.

III. Browse files and directories to the new Palette file. Palette file must have *.dp* extension and conform to the simple format (8x16bpp hex colors).

IV. Select the new Palette file from the list of files

V. Click OK in the Color dialog box.

Custom Palettes from the `$FVWM_USERDIR/palettes` can also be deleted. Just pick the name from the list and click on the Delete ... button. You will be asked for confirmation, and then, if standard UNIX filesystem user and group ownership and permissions are allowing that, palette file will be permanently deleted.

Creating new palette by modifying an existing one makes a copy of the currently selected palette with a default name "Custom". When applying this new palette, Color Style Manager will ask you to rename this palette, which is a good idea, because the next modification will rewrite "Custom" generic name again, and previous work will be lost. You then create your custom palette by modifying the copy, so the original palette is not changed.

Button Modify will popup color editor if user selects one of the 8 (or 4) base colors. When selected, this color frames will get "Abc" written in them with automatic foreground choice for that RGB/HSV combination. Frames can be unseleced by simply clicking on them again. When one base color is selected Modify will present editor with controls for Red, Green and Blue values, as well as Hue, Saturation and Value. On the top left corner are preview squares with names "Old" and "New". When

changing color with RGB and/or HSV controls, this "New" square button will change it's colorset. Color can also be picked from any screen element with Grab Color button- If action is not Cancel but OK, selected color will be modified and new palette with generic name "Custom" created immediately. When finishing theme selection in Color Style Manager with modified colors, Color Style Manager will ask for a name of the new palette. The suggested default is "Custom" but on the subsequent modification, this is the palette which will be modified and past modifications will be effectively lost. For that reason, it is probably a good idea to save modified palette as new palette with some other name. In that way, it can be temporary changed for some other and turned back again later. This color modification dialog actually serves as palette creator (based on previous palettes) and editor.

There are 8 spaces with colors from the currently selected palette (4 spaces in 4-color mode) and generated XPM file with all 40 colors displayed. Button Number of Colors calls transient window where user can select 4 or 8 color mode. System default on modern desktop is 8.

What is most important new feature in Color Style Manager are integration options. This are:

- Own currently used backdrop synchronization (default)
- X resources in `$FVWM_USERDIR/Xdefaults` (default)
- GTK2 (default)
- GTK3 (default)
- Qt4 (default)
- Qt5 (default)
- User's `$FVWM_USERDIR/libexec/colormgr.local` script if exists, called with the path of the applied palette and number of colors.

The last integration is used to integrate what default widget integrations cannot reach. For example Gkrellm skin or some terminal preferences. Qt/Qt5 integration is easy, since this toolkits can use their *GTK* engine to integrate self with GTK theme. All that Color Style Manager has to do is to define GTK engine in `~/.config/Trolltech.conf` and `~/.config/qt5ct/qt5ct.conf` for colors from the new palette to be used.

GTK2 and GTK3 are heavy work part. Here, we are using work derived from one CDE theme for XFCE desktop and GTK2 + GTK3, purified and adapted for NsCDE (see Section 22). This is written in python. If turned on, this will produce `$HOME/.themes/NsCDE` directory with the theme for GTK2 and GTK3, and will edit `$HOME/.gtkrc-2.0` and `$HOME/.config/gtk-3.0/settings.ini` to put or change `gtk-theme-name` value. If NsCDE palette with dark background 4 color is choosen, Color Style Manager will put `gtk-application-prefer-dark-theme` into `$HOME/.gtkrc-2.0` and `$HOME/.config/gtk-3.0/settings.ini`. If palette with light color 4 (used for text areas and text fields usually) is choosen, that variable will be removed from the both configuration files. When switching off from NsCDE to some other environment, care must be taken manually to handle `gtk-application-prefer-dark-theme` in configuration files if last palette in NsCDE was using dark background.

If `nscde_use_xsettingsd` is set to 1 in the `$FVWM_USERDIR/NsCDE.conf` after applying new color theme, user's X Settings in `$FVWM_USERDIR/Xsettingsd.conf` will be adjusted and xsettingsd(1) daemon restarted for settings in GTK and Qt applications to be applied immediately. This option can be enabled by editing NsCDE.conf or during initial setup. NsCDE starts **xsettingsd** daemon with "-c $FVWM_USERDIR/Xsettingsd.conf" parameter. This file must be present if it was not installed by the initial setup procedure.

Key Bindings:

- Ctrl+Q: Quits Color Style Manager.
- P: Like Preview was pressed. Previews currently selected color scheme from the list.
- Up/Down: Goes one item on the color schemes list up or down.
- Sun Help and F1: Displays this help text.

Notice: In the preview mode under FVWM3 non-global monitor layout, only the backdrop of the currently focused monitor is previewed in the colors of the new palette.

5.4.1. Number Of Colors Dialog

A Color Style Manager part

Helper dialog to select 4, 8 or default color scheme in Color Style Manager. It changes number of colors while browsing, previewing or choosing a color theme, as well as two additional options for 8 colors scheme.

- Use 4 Colors Scheme: Uses only the first four colors of the choosen palette. With slightly modified palette "Crimson" this was probably default on all versions of Sun Solaris.
- Use 8 Colors Scheme: uses full palette with all 8 colors. This is NsCDE default on all modern displays.
- Default (4|8): Indicates default which can be changed or turned back.
- Color 8 for Front Panel and Icons: in 8 colors scheme, this will make icon part of the Front Panel and workspace icons background to use eighth color from the choosen palette instead of fifth color. This scheme was known to have been default on some HP-UX and AIX versions of CDE, and in NsCDE it is a user option. This option does not have effect in 4 colors scheme, and hence cannot be selected together with it.
- Color 6 for Workspace Manager: in 8 colors scheme, this will make Workspace Manager colored with color 6 which is usually used for menus, and tools background. This scheme was default on some versions of CDE on some UNIX systems, but in NsCDE it is a user option. This option does not have effect in 4 colors scheme, and hence cannot be selected together with it.

Key Bindings:

- Return: Selects OK.
- Escape: Quits Dialog.
- Sun Help and F1: Displays this help text.

5.5. Exec Dialog

By default key binding for Exec Dialog is Alt+F12, and an entry "Execute ..." located on main Workspace Menu above terminal item as "Exec" dialog. This is an input form for executing one-shot commands without terminal. It has options to run command in terminal (`$[infostore.terminal]`), and to remain open after executing commands for subsequent commands. It has it's own command history which can be turned back with cursor up and down keys. Escape key closes dialog, enter executes, Ctrl+Enter executes in default terminal application.

As an example, this dialog can be used if on the current page or workspace terminal application is not present, and only some simple command is needed to be quickly executed.

Key Bindings:

- Escape: Quits Exec dialog
- Return: Performs an action like if Exec is pressed.
- Ctrl+Return: Performs an action like if Exec is pressed, and Execute in terminal checkbox is checked.
- Shift+Return: Performs an action like if Exec is pressed, and Leave this dialog open checkbox is checked.
- Ctrl+Shift+Return: Performs an action like if Exec is pressed, Execute in terminal and Leave this dialog open checkboxes are both checked.
- Up/Down: Brings back and forth command history in text dialog box
- Sun Help and F1: Displays this help text.

If installed, NsCDE can use rofi(1) command and application launcher from the keybinding Alt+F12 instead of built in Exec Dialog. Workspace Menu item "Execute ..." remains the same, only keybinding will call rofi(1) instead of built-in simple Exec dialog. For this to work, InfoStore variable `nscde_use_rofi` must be set to "1" in the `$FVWM_USERDIR/NsCDE.conf`. Rofi is integrated with NsCDE color theme and will be set up on the first use, which lasts 3 seconds initially and after changing color theme with Color Style Manager.

5.6. Font Style Manager

Font management is the area where NsCDE and CDE are probably most different. Font Style Manager is completely NsCDE tool to set fonts for usage inside FVWM and external toolkits integration (X Resources/Motif, GTK2, GTK3, Qt4, Qt5 ...).

NsCDE defines 15 fonts. Five groups with three members:

- Normal Small
- Normal Medium
- Normal Large
- Bold Small
- Bold Medium
- Bold Large
- Italic Small
- Italic Medium
- Italic Large
- Monospaced Small
- Monospaced Medium
- Monospaced Large
- Monospaced Bold Small
- Monospaced Bold Medium
- Monospaced Bold Large

This fonts are defined as FVWM *infostore* variables in the file `$NSCDE_DATADIR/fvwm/Font-$NSCDE_FONT_DPI.fvwmconf` and/or in the file `$FVWM_USERDIR/NsCDE-Font-$NCDE_FONT_DPI.fvwmgen`. User's `$FVWM_USERDIR/NsCDE-Font-$NSCDE_FONT_DPI.fvwmgen` is a symlink to either `$NSCDE_DATADIR/fontsets/SomeName.fontset` or to `$FVWM_USERDIR/fontsets/SomeName.fontset`. Further, they are defined as CPP macros in `$FVWM_USERDIR/Xdefaults.fontdefs` which is included in `$FVWM_USERDIR/Xdefaults` where it is used. GTK2 and GTK3 are also getting default font (Normal Medium) in their configurations if integration option has been selected in Font Style Manager. X resources and GTK are not refreshed by default, their checkboxes can be unselected if some of this widget integrations is not desirable by the user.

The Font Style Manager itself consists of fontsets and fonts. Fontsets are named complete sets of five groups of three members of fonts defined above. Fontsets are stored in `$NSCDE_DATADIR/fontsets` and in `$FVWM_USERDIR/fontsets`. If font set is selected in Font Style Manager, 15 fonts from the set

are loaded into preview lists of the application and can be immediately applied or further customized by leaving Use Predefined Font Set mode before saving defined scheme as `$FVWM_USERDIR/fontsets/SomeName.fontset` and linking this name to `$FVWM_USERDIR/NsCDE-Font-$NSCDE_FONT_DPI.fvwmgen`. List of fontsets on the left GUI list is inactive until button Use Predefined Font Set is not turned on, then Font Style Manager operates with sets of fonts, and not in compose mode with individual fonts. In this mode, right GUI list contains information about fontset and previewed fonts when they are selected with mouse or keyboard. This information can contain additional bottom line *(Not available)* if font from the fontset does not exist under this name on the system. If such fontset is loaded and saved anyway, system's XFT subsystem will select nearest match or the default font instead of non-existing one.

By default, manual font selection contains list of XFT fonts found on the system in the left GUI list and their styles (regular, bold, italic ...) on the right GUI list. list.

DPI value for fonts in NsCDE is by default defined to be 96 in `$FVWM_USERDIR/Xdefaults.fontdefs` as "Xft.dpi". This is to accomplish reasonable defaults for all widgets and programs which use them, but if overriden, existing user custom fontset for 96 DPI will not be considered anymore. System default from that approximate DPI range will be used until new custom fontset is created, where `NSCDE_FONT_DPI` will not be 96 anymore.

Main font selectors are:

- Font Size Group
- Font Style Group
- Set Size (available only in manual selection mode)

First popup menu loads 5 fonts from one of the 3 sets: small, medium or large. Second popup determines on which font current selection is working: normal, bold, italic, mono or mono bold (clicking on font preview itself has the same function) and third popup menu sets font size. When Font Style Manager is started, current fontset is loaded and previews are populated with that fonts. Default mode of operation will be switched to manual font selection for customizations into the new fontset.

Button Default loads default `$NSCDE_DATADIR/fontsets/DejaVuSerif.fontset` which can then be saved as a choice or further customized into the new fontset. Bottom half of the Font Style Manager contains preview for all fonts from the one of the three selected size sets.

Checkboxes Refresh GTK2/GTK3, and Refresh X Resources" are integrating font selection with popular widgets by providing *normal medium* font and it's size to their configuration files. Qt4 and Qt5 should automatically pick Gtk fonts if Qt "GTK2" font engine is active in their configurations. If not, **qtconfig-qt4** and **qt5ct** applications can be started and some minor changes done and undone - enough for Apply/Save to take effect, and then font from Gtk will be loaded for sure. Checkbox Run User Script will attempt to run **$FVWM_USERDIR/libexec/fontmgr.local** if it exists, with argument of the new config file. This is intended for user's customizations which are currently beyond NsCDE's scope of program and widget integrations.

Save button will save fontset choice, or generated selection as a new fontset in manual selection mode, make link to it in `$FVWM_USERDIR`. If manual modification has taken place, user will be asked to name a new fontset with the popup dialog. Name "custom" is the default proposal in the dialog, but it should be changed, because this name is predestinated to be rewritten on the next Font Style Manager saving action. Further, user will be asked to immediately restart FVWM, for changes to be applied from all parts from the new configuration.

Key Bindings:

- Ctrl+Q: Quits Font Style Manager.
- Sun Help and F1: Displays this help text.
- Up (Arrow Up): Selects prior font or fontset on the list
- Down (Arrow Down): Selects next font or fontset on the list
- Prior (Page Up): Selects preview box from the bottom to the top and cycles back to the bottom again
- Next (Page Down): Selects preview box from the top to the botttom and cycles back to the top again
- Ctrl+S: Loads small group of fonts on the preview fileds
- Ctrl+M: Loads medium group of fonts on the preview fileds
- Ctrl+L: Loads large group of fonts on the preview fileds
- Ctrl+Plus: In manual selection mode, increases font size for selected font
- Ctrl+Minus: In manual selection mode, decreases font size for selected font
- Ctrl+Equal: Sets font size to default start point size for current font size group previewed
- Ctrl+F: Changes mode of operation between prefedined fontsets and manual font selection mode

5.7. Keyboard Style Manager

Keyboard Style Manager tool can be used to set (xset) 4 values:

- Auto Repeat on/off: Sets characters to repeat when you hold down their keys
- Start Delay (start of repeat delay): initial delay on pressed key to start with auto repetition
- Repeat Delay: Speed at which auto repeat works
- Click Volume: Determines the volume of key clicks (turned off at 0%)

This values are standard xset(1) *r* and *c* subcommands and their values, minimal and maximal allowed values are (or should be in most cases) the same in GUI as they are in command line tool.

Default button will set auto repeat to on, start rate to 512, repeat delay to 16 and click volume to 50.

Apply button applies setting in runtime, while Save button writes `$FVWM_USERDIR/Xset.conf` file which is a generated **xset** command batch executed during startup.

Key Bindings:

- Ctrl+Q: Quits Keyboard Style Manager.
- Escape: Quits Keyboard Style Manager.
- Sun Help and F1: Displays this help text.

5.8. Occupy Workspace, Occupy Page and Occupy Monitor

This tool dialog is called from the left titlebar button 1 Window Operations Menu as "Occupy Workspace", "Occupy Page" or "Occupy Monitor". Or from Meta+Space key for Workspace, Alt+Space for Page, and Meta+Alt+Space key combinations from the window context. Occupy tool is the extended (for FVWM pages and multiple monitors) version of the same CDE dialog and it sends selected window to a workspace, page or XRandR monitor selected from the list, or it can make it sticky across workspaces and/or pages with All Workspaces or All Pages checkbox pressed in. One addition here is the checkbox Go With the Window; when checked, makes NsCDE to change a current workspace, page and/or monitor and go with the window, or where window was sent. Radio buttons Workspaces:, Pages: and Monitors: are changing the current context of operation between workspaces, pages and monitors, which is visually also manifested in the main list as the names of either workspaces, pages or RandR names of the monitors. If called as "Occupy Workspace", pages or monitors can also be selected and changed by switching this context with this buttons or with the Space key shortcut in focused Occupy window. Radio button for monitors and "Occupy Monitor" functionality is only shown and available if multiple physical (or virtual) monitors are detected by FVWM and NsCDE. OK performs an action, Dismiss quits Occupy tool without taking action.

On the upper left part of the window there are three buttons with labels Workspaces, Pages and Monitors:. When either one or another selection is changed from the current state, that is, state of the workspace and page on which window context was when Occupy is called, there will appear an ASCII asterisk (*) symbol at the end of the label. This means selection is changed, and this is visible for workspaces or monitors even when pages are on the selection list and/or vice versa. If selection is changed back again to state of page, workspace and/or monitor, which was initial, asterisk will disappear, but will again appear when this selection is changed to some other choice again. That way, orientation of what was touched can be known.

Key Bindings:

- Escape: Quits Occupy Tool without performing an action.
- H, Sun Help and F1: Displays this help text, like Help was pressed.
- A: Checks All Workspaces or if context is pages, All Pages checkbox.

- Return: Performs move action like OK button was pressed, without going to the selected workspace and/or page.
- Ctrl+Return: Performs move action like OK button was pressed, and changes active workspace and/or page to the same destination where window has been sent.
- Up/Down: Selects workspace, page or monitor in the up or down direction on the list.
- Space: Changes between workspaces, pages and monitors mode of operation. Actions on workspaces, pages and monitors can be combined while switching modes of operation, no matter if it was initially called as "Occupy Workspace", "Occupy Page" or "Occupy Monitor".

5.9. Graphical Workspace Manager (GWM)

Same named graphical tool was historically present usually on Sun Solaris. Graphical Workspace Manager of NsCDE is FvwmPager and FvwmScript based version of GWM. It implements most of the functions of CDE GWM which are applicable in NsCDE and FVWM environment. It also implements some NsCDE and FVWM specific options.

This tool can be called from Root Menu, from Applications->NsCDE submenu of the Workspace Menu, Shift+Tab (second call) and Meta+Alt+Home keybindings, and by default from the 7th Subpanel of the Front Panel (Desktop Settings).

GWM enables user to see all of the NsCDE workspaces and pages, navigate pages and workspaces with mouse, select windows with middle mouse button, and taking actions on them with Window menu actions.

With actions on the Workspaces menu, it is possible to:

- Call Workspaces and Pages Style Manager
- Call Workspaces and Pages Style Manager in Rename Workspace mode active with the current workspace selected for rename.
- Rearrange all windows on the current page in a cascade
- Rearrange all windows on the current page to be tiled vertically
- Rearrange all windows on the current page to be tiled horizontally
- Open the GWM Options dialog
- Exit GWM

With actions on the Window menu, it is possible to:

- Iconify and deiconify Window selected with a mouse button 2
- Shade or deshade Window selected with a mouse button 2
- Close Window
- Forcefully Close Window (Terminate Application)
- Call Occupy Workspace on the window selected with a mouse button 2

Help menu contains information about GWM keybindings and GWM manual (this one).

Key Bindings:

- Ctrl+M: Calls Workspaces and Pages Style Manager
- Ctrl+R: Calls Workspaces and Pages Style Manager in rename workspace mode with the current workspace ready to be renamed
- Ctrl+O: Pops up Options window of the GWM
- Ctrl+W: Calls Occupy Workspace tool
- Escape: Quits GWM.
- Ctrl+Q: Quits GWM.
- Sun Help and F1: Displays this help text, like Help was pressed.

5.9.1. Graphical Workspace Manager Options

GWM Options dialog can configure some aspects of the GWM itself and GWM pager behaviour. By configuring the later, a stripped down version "GlobalPager" is also configured.

The following controls can be configured:

- Show Workspace Backdrops - if unselected, only a CDE palette colors associated to this workspace will be drawin in the workspace pager background. Default is selected.
- Highlight Current Workspace Page - current workspace background will be colored in active window color. By default this option is off, it is however used by LocalPager.
- Workspace Names Above Workspaces - workspace names are drawn as headers of the workspaces. Default.
- Workspace Names Below Workspaces - workspace names are drawn as footers of the workspaces. Not default.
- Number Of Visible Rows - when this number is smaller, workspaces are firstly drawn horizontally, then vertically. If this number is 1, workspaces are all in one row. Biggest number possible with

current number of workspaces used will produce vertical GWM application with all workspaces in one column.

- Desired Workspace Width (Ratio) - number between 10 and 20. By default a calculation is done to show bigger GWM on higher screen resolutions and more compact smaller on lower screen resolutions. If this number is too big, and in combination with one-row or one-column setup with 4 or more workspaces on lower resolutions can produce wider window that cannot fit on the screen. Experimenting is enocouraged however.
- Show Window Names In Balloons - weather to show small marker balloons on mouse over windows action in the pager. Default is on.
- Respect Window Skip List - do not draw in the pager windows which are part of the FVWM skiplist (like Front Panel, Gkrellm, transients etc ...)
- Show Small Icons Inside Windows - if selected (the default) it will draw mini icons of applications present on the workspaces.
- Show Names Inside Windows: icon names of the windows will be written in pager windows frames with very small font instead of mini icons. Not default.

Options Dialog Key Bindings:

- Escape: Closes GWM Options dialog without taking any action
- Tab: Cycles configurable item elements from top to bottom of the window
- Return: Takes the action as if OK button was pressed.
- Sun Help and F1: Displays this help text, like Help was pressed.

5.10. Window Geometry Manager

This tool dialog is called from the left titlebar button menu. Or from Meta+Alt+G key combination from the window context. This Manager offers convenience to save size and position of the non-transient window. Once window geometry is saved, next run of the same application window on FVWM page where the window of the same resource+class combination is not already present will cause window to be resized to that exact width, height, X position and Y position X11 coordinates as stored in the `$FVWM_USERDIR/GeoDB.ini`. This ini-style file is managed by this application, as well as from the background runner `$NSCDE_TOOLSDIR/confset` and read by the `$NSCDE_TOOLSDIR/confget`. Application automatically fills X11 Class and X11 Resource names on contextual call, as well as geometry parameters. Before applying this settings, user has a chance to fine tune this parameters in text fields and confirm (or cancel) the action on the end. Using Clear all geometry values, as well as those which may be saved from the past are cleared and deleted. After the Clear action, there are two options:

- Filling all fields manually and saving the result

- Saving empty fields: this action will cancel and/or remove any existing geometry mappings for the selected X11 program window.

Note that geometry savings are dependent on the current monitor resolution. In other words, size and position of the X11 window are saved for 1920x1080, 2560x1440, 1600x900 and any other possible monitor resolution. Multiple savings on different monitor resolutions are possible and saved under separate sections in `$FVWM_USERDIR/GeoDB.ini`.

Key Bindings:

- Escape: Quits Window Geometry Manager without performing an action.
- Sun Help and F1: Displays this help text, like Help was pressed.
- Return: Performs action like Save button was pressed.

5.11. Mouse Style Manager

Mouse Style Manager tool manages pointer - that is, mouse settings. It's duties are few more than only xset(1) *m* command. Namely:

- Mouse acceleration (xset)
- Threshold (xset)
- Double-Click
- Handedness (xmodmap)

Acceleration and Threshold are standard xset(1) *m* options and man page for xset(1) should be consulted.

Handedness can be set to flip left and right mouse button functions, while *Double Click* is in fact most complex: it has test area where user can test double click speed (pixmap will change on double-click success), and this setting is changing:

- FVWM MenuStyle DoubleClick value (`$FVWM_USERDIR/NsCDE.conf`)
- X Resources multiClickTime resource in `$FVWM_USERDIR/Xdefaults.mouse`
- Qt/KDE settings in `$HOME/.kde/share/config/kdeglobals` (or similar path) if found
- Gtk2 (`$HOME/.gtkrc-2.0`) if file exists
- Gtk3 (`$HOME/.config/gtk-3.0/settings.ini`) if file exists

Double-Click value is in milliseconds in all mentioned configurations. Apply button applies **xset** and **xmodmap** values set in runtime, but not double-click settings.

Save button saves changes in `$FVWM_USERDIR/Xset.conf` and all other optional files for widget and FVWM integration. Default button will set handedness for right-handed, double-click on 450, acceleration on 60, and threshold on 8, apply xset and xmodmap values, FVWM *MenuStyle DoubleClickTime* and will warp pointer to Save button for actions to be written in config files.

Key Bindings:

- Escape: Quits Mouse Style Manager.
- Ctrl+Q: Quits Mouse Style Manager.
- Sun Help and F1: Displays this help text.

5.12. Power Save Manager

This tool manages screen DPMS values. It uses standard **xset(1)** to set screen *standby*, *suspend* and *off* times. Values are from 0 to 65535. Standby time cannot be bigger than suspend and/or off time, and suspend time cannot be bigger than off time.

It has a checkbox which enables or disables DPMS management at all. Values are written in `$FVWM_USERDIR/Xset.conf`.

If `$HOME/.xscreensaver` is present and has DPMS options in it, they will be synchronized with xset *dpms* options written in `Xset.conf`. Apply button applies runtime (but not xscreensaver) while Save button writes configuration file and `$HOME/.xscreensaver` DPMS settings if this file exists. Default button will set the following defaults: DPMS *on*, standby *1200 seconds*, suspend *1800 seconds*, and off time *2400 seconds*, apply this settings on runtime, and point mouse to Save button for changes to be written.

Key Bindings:

- Escape: Quits Powersave Style Manager.
- Ctrl+Q: Quits Powersave Style Manager.
- Sun Help and F1: Displays this help text.

5.13. Subpanel Manager

Only in NsCDE. Tool written for managing FvwmButtons transient Subpanels which are opened from the Front Panel. It is called from *Install Icon* action on every Subpanel. This tool exists because FvwmButtons doesn't implement drag and drop, and there is no application manager present, since this part of CDE functions cannot be easy implemented even if some file manager is taken to act as application manager.

Subpanel Manager has a list of all applications which are providing system menu with it's presence (fvwm-menu-desktop is used in the background for generating application list), and the list of applications provided in the current Subpanel from which tool is called by "Install Icon". This lists are displayed on the top of the window side by side: system menu applications on the left, and current Subpanel items on the right list. There are 3 text fields: Name, Command and Icon File. This fields are automatically populated when some item from mentioned lists is clicked, but it can also be populated manually if user wishes to add a custom application, Fvwm Module, Fvwm Function or other entry manually on the Subpanel. For manually adding application entry to subpanel, initially at least one item from the list of available applications must be initially selected, and then totally or partially rewritten and overriden, otherwise, no item will be added to the subpanel. Special type of "Check for ..." is meant for entries which in command field are defining for the first command something common like shell, **env** etc. If we want to override a pointless check existance for this, and define other command string for checking, popdown menu option Check for ... can be selected, and small text field below popdown menu will appear, where this command can be specified. Subpanel Manager can also remove existing entries from the Subpanel.

Subpanel Manager implements this helper functions for modifying and applying settings on new items:

- Type (exec, module, function, other, check-for)
- Do not check for command existence
- Icon indicator

To place some new application from the left (all applications) list in exact place on the subpanel, select an existing entry on the subpanel (right) list. When left arrow button is clicked, new application will appear on the list exactly below that one which has been previously selected, otherwise, default is to place new entries as second entry on the subpanel's list of applications.

Click on the icon indicator will open simple file browser which can be used to find, see as preview, and set icon for manual entries which are not part of the applications list, or an alternative icon for program from applications list. Apply button regenerates subpanel, while Save button does this and also quits Subpanel Manager.

Subpanels configuration file `Subpanels.actions`, can be edited by hand in `$FVWM_USERDIR` if something needs to be changed on existing entries. If editing by hand, **$NSCDE_TOOLSDIR/generate_subpanels** must be used to generate FVWM configuration output

which must be redirected into `$FVWM_USERDIR/Subpanels.fvwmX.fvwmconf`, where "fvwmX" is either "fvwm2" or "fvwm3".

Key Bindings:

- Escape: Quits Subpanel Manager.
- Ctrl+Q: Quits Subpanel Manager.
- Sun Help and F1: Displays this help text.

5.14. Subpanel Settings

A small helper dialog with functions to change display name of the subpanel, width of the subpanel if titles require wider (or short ones narrower) panel frame, end enabled/disabled state of the subpanel. Button Reset will erase user configuration and load system default one for given subpanel. Name, width and enablement all have their own Default button right of the text field. If pressed, it will load system defaults for name, width and subpanel's enablement state. All buttons are doing in memory changes until OK is pressed, except Reset button which acts immediately. Close button closes dialog without changes except if Reset has been pressed. This dialog is called from subpanel menu which can be popped with the left (and only) button on subpanel's titlebar. It is called Subpanel Settings. It reloads configuration of the given subpanel after applying changes and exiting with OK.

Key Bindings:

- Escape: Quits Subpanel Settings.
- Ctrl+Q: Quits Subpanel Settings.
- Sun Help and F1: Displays this help text.

5.15. System Action Dialog

This is the login/logout form with the possibilities to reboot or shutdown the system, or change X session. It has also options for suspend/sleep (S3), hybrid suspend, and hibernation of the system. Of course, **reboot**, **shutdown**, **suspend**, **hybrid suspend** and **hibernate** will work for a ordinary user only if system is configured with authorizations for a user to perform such actions. Examples of this can be systemd(1) on Linux which recognizes local user, or system with **sudo(8)** entries are configured to do so. While System Action Dialog is active, root cursor changes to line-crossed cursor which is dismissed after the action is performed or dialog action dismissed. Not all of this actions are possible on all systems, but with the `$NSCDE_TOOLSDIR/acpimgr` wrapper, Linux with pm-utils, Linux without pm-utils but with systemd(1) and FreeBSD with acpiconf(8) are supported in this moment.

In `$NSCDE_ROOT/share/doc/nscde/examples/sudo`, one can find example which can be put in `/etc/sudoers.d` with little changes. Confirm button applies, Dismiss cancels and closes the dialog.

Key Bindings:

- Escape: Quits System Action Dialog without performing any action.
- R: Toggles Remember State of Action On/Off
- Ctrl+Q: Same as Escape
- Ctrl+Return: Performs an action as if Confirm is pressed.
- Up/Down: Changes between 7 possible options of System Action Dialog's popup menu.
- Sun Help and F1: Displays this help text.

5.16. Sysinfo

Well known *Workstation Information* info dialog. It doesn't have any functions apart close (Dismiss) button. It's role is informational. It displays current username, hostname, machine and CPU architecture type. IP address, hostid, network (NIS, NISPLUS, LDAP ...) domain name, internet domain name, size of the physical RAM, swap size, swap usage (as progressbar), operating system long name, and then version of the FVWM and version of the NsCDE. Last, it shows time when system was last booted. This dialog can be found on the *Tools* or fourth Subpanel of the Front Panel in default configuration, under the entry *Workstation Info*. In the context of it's window, keys Escape, Return and Q will close a window, while Sun Help and F1 displays this documentation.

5.17. Window Style Manager

This GUI tool is not just exact copy of the similary named CDE tool Window Style Manager. NsCDE Window Style Manager extends number of actions which can be configured regarding window behaviour, which is undestandible because FVWM window manager is a way more configurable and powerful than CDE's original dtwm.

On the other hand, writing a tool that will handle ALL fvwm options and write that in FvwmScript which not much a powerful language can easily contain tens of thousands lines of code, and yet be buggy and probably some things will be impossible to do. Even then, it will be burden for users to use and almost completely avoided.

A middle solution was provided: all functions from original CDE dialog, plus some similar extended FVWM options, on the first tab, and other important configurations on the rest three tabs. Tabs are implemented as popup menu on the top right side of the window - choosing an option from that menu changes displayed options - a poor man's tabs in a way. Some of options provided by Window Style

Manager are not full set of this options if configured manually in FVWM configuration, but for needs of CDE clone this is more than enough.

5.17.1. Configuration Section: Window Behavior

- Only Pointer Inside Window Makes Focus: this configures so called *MouseFocus* as catch-all FVWM Style (*). See fvwm(1) for MouseFocus.
- Point In Window To Make Active: this is FVWM *SloppyFocus* Style option. This is default focus style. If you want more CDE behavior, select MouseFocus option above. In SloppyFocus mode, pointer will make focus on window while entering it, but will not lose focus while leaving the window. See fvwm(1) for SloppyFocus.
- Click In Window To Make Active: self-explanatory. This is FVWM's ClickToFocus style.
- Raise Window When Made Active: self explanatory. If selected, focused window will be raised. This option will enable FvwmAuto module instance with `-mfocus` option. See fvwm(1) and FvwmAuto(1).
- Allow Primary Windows On Top: this will allow window to lower it's transient windows (popups and such). See fvwm(1) for *RaiseTransient* and *DontRaiseTransient* styles.
- Lower Transient With Primary Window: self explanatory. See fvwm(1) for *LowerTransient* and *DontLowerTransient*
- Raise/Lower Primaries With Transients: if transient windows are raised or lowered, primary windows goes with them. See fvwm(1) for *StackTransientParent* and *DontStackTransientParent* style options.
- Show Contents During Move: weather window contents is visible or not during window move. Default is a transparent frame with a grid. See fvwm(1) for *OpaqueMoveSize*.
- Time After Which Active Window Is Raised: if Raise Window When Made Active is turned on, this option will be enabled and time in milliseconds can be set here. See FvwmAuto(1).
- Manual Window Movement Threshold: see fvwm(1) for a *MoveThreshold* option. 5 pixels is the default.

5.17.2. Configuration Section: Window Icons

- Use Icon Box: if this option is selected, main FVWM configuration in `Main.fvwmconf` will spawn *FvwmIconBox* configured as close as possible as an alternative to icons of iconified windows. Classic icons will be disabled. The rest of options in this "tab" will be disabled because they do not apply anymore in this configuration. Note: FvwmIconBox has not exactly the same functionality as CDE's Icon Box.

- Place On Workspace: default. Icons of the windows will be placed on the screen. By default, from top left to the direction bottom left.
- Place Icons Left/Right from Bottom/Top to Top/Bottom: this four exclusive options will direct icon placement on the screen. See *IconBox* and *IconFill* options in fvwm(1) for more information.
- Default Icon Size: in pixels. See *IconSize* in fvwm(1).
- Maximum Icon Size: in pixels. See *IconSIze* in fvwm(1).

5.17.3. Configuration Section: Page Edges

- Raise Front Panel On Page Change: as is says, when current active page changes, Front Panel will be raised.
- Pager Preview On Page Change: on page change, spawn a small transient FVWM pager called *LocalPager* on the top center of the screen to indicate what is on the current page in the workspace - this is controlled by the FVWM infostore variable `pageshowrootpager` in the `$FVWM_USERDIR/NsCDE.conf`
- Disable Page Change On Screen Edge: if selected, current page will not change when pointer is longer time on the screen edge. In this mode, active page must be changed by other means (keyboard shortcuts, WSM submenus, PGM, left click on 1st titlebar button ...)
- Page Change On Screen Edge (1px): default internal detector of the screen edge. Do not change this if it is working.
- Page Change On Screen Edge (2px) If FVWM has a problem with X server and page change does not work smooth, use this option as a safe alternative. See fvwm(1).
- Page Edge Resistance: how many milliseconds FVWM waits on the screen edge area for an page change action to be taken. Default is 380.
- Edge Window Move Resistance: similar as Page Edge Resistance, but for move operation - how hard it should be to move a window between pages. Defaults to 80 pixels.
- Edge Window Move Delay: time to wait to consider moved window for page change in the first place (to start counting pixels of the Edge Window Move Resistance). Defaults to 320 pixels.

5.17.4. Configuration Section: Animation

This tab controls behavior of the FvwmAnimate. See FvwmAnimate(1).

- Animate Window Iconification: on/off of the FvwmAnimate module.
- Animation Effect: See FvwmAnimate(1)
- Animation Frame Delay: See FvwmAnimate(1)
- Animation Revolution Twist: See FvwmAnimate(1)

- Outline Width: See FvwmAnimate(1)
- Animation Iterations: See FvwmAnimate(1), be careful on virtual displays with a bad video driver. It can behave really slower than on host system with the same parameters.

5.17.5. Misc Window Style Manager Functions

Button Default on the top right edge of the window will read system defaults into options and they will be set in permanent configuration if OK button is pressed afterwards.

OK button applies changes to `$FVWM_USERDIR/NsCDE.conf`. Dismiss button will close the window without making any changes.

Key Bindings:

- Escape: Quits Window Style Manager.
- Ctrl+Q: Quits Window Style Manager.
- Sun Help and F1: Displays this help text.

5.18. Workspaces and Pages Style Manager (WsPgMgr)

This tool is specific to NsCDE. It configures the following things:

- Number of the workspaces
- Names of the workspaces
- Number of virtual pages in the workspaces
- Use / Do not use compact Workspace Manager on the Front Panel
- Desktop Model on multi head monitor setup (FVWM3 only)

This graphical tool writes files `$FVWM_USERDIR/WSM.conf` and `$FVWM_USERDIR/NsCDE.conf`. On the top of the window is a graphical representation of the Workspace Manager. When clicked, every button becomes editable field and it's name can be changed (pres return on the text field after writing new name). This change will be written immediately WITHOUT pressing OK button. Name of the workspace will be changed across all NsCDE and FVWM (WSM, Occupy Workspace/Page/Monitor, menus etc). In NsCDE, there are 4 options for workspaces: 2, 4, 6 and 8 model. FVWM itself however supports much larger number, but in NsCDE care must be taken about presentation of this workspaces in various applets and tools. Nevertheless, theoretically even in NsCDE with maximum number of

workspaces multiplied with maximum number of pages, user can get 128 working screens which is probably enough (too much) for 99% of the people.

Check Box Use Compact Workspace Manager when checked in will instruct NsCDE to use Workspace Manager widget in the Front Panel sized for as it is for four workspaces, but it will expand buttons in two workspaces setup and shrink accordingly in the case of six or eight workspaces setup. This option does not have any visual effect on four workspaces setups, because it is sized for such setup.

Number of vertical and horizontal pages can also be configured here. This change will affect Page Manager (PGM) icon on the Front Panel, and menus which are displaying pages. Page names are not configurable via GUI, their names are read from `$NSCDE_DATADIR/defaults/pages/`, but can be overriden in user's configuration. User can put into it's `$FVWM_USERDIR/WSM.conf` enties in the same format as files in the above mentioned directory.

Pop up menu prefixed as "Monitor Workspaces" allows users of NsCDE with FVWM3 to select between currently existing three modes of FVWM3 desktop configurations: global, per-monitor or shared. When used with FVWM2, this popup will not be shown, instead, text "Global Only (FVWM2)" will indicate non-applicability of this feature under FVWM2. Under FVWM3, new DesktopConfiguation model is used immediately, but FVWM3 is still restarted to get consistent and clear new behaviour of the window manager.

OK button saves configuration, restarts window manager and quits. Cancel button discards, except if new names of desks are set, because this is written immediately, but window manager is not restarted. Workspaces and Pages Style Manager is called from the Style Manager, NsCDE submenu of the Workspace Menu's Applications submenu, and Front Panel's small left button menu which is called when that button is clicked with the right pointer button.

5.19. NsCDE Process Manager (NProcMgr)

This tool is specific to NsCDE. It manages NsCDE FVWM and rest NsCDE processes, and re-reads given part of the FVWM configuration on demand. The following actions are provided and supported:

- Recheck - Check selected item processes running state
- Restart - Restarts selected item processes
- Stop - Stops selected item processes
- Start - Starts selected item processes
- Reread - Reads fresh FVWM configuration for the item selected on the Configurations popup menu.

There can be more processes (items) selected for a given action. Items are not unselected after action. NsCDE Process Manager can be used for diagnostic, debugging and developing functions and other parts of the NsCDE. It Provides GUI for "KillModule", "Module" and various NsCDE FVWM functions, and

for f_ReadCfg". Process list consists of check buttons with item/process names on the left, and their functional state, running state and PID on the right side. After every action, selected item is checked again and it's running state and PID are refreshed. Check All Now button when pressed will refresh status for all items, selected or not.

Key Bindings:

- Ctrl+Q: Quits Workspaces and Pages Style Manager.
- Sun Help and F1: Displays this help text.
- Ctrl+A: Hits Apply button.
- Ctrl+R: Hits Reread button.
- Ctrl+K: Hits Check All Now button.
- A: Cycles actions from Action PopUp menu.
- C: Cycles configurations from Configurations PopUp menu.

6. Helper Dialogs

6.1. ActionForm - FvwmScript

Dialog which uses custom text and asks user for action. Action is then executed (OK) or aborted (Dismiss). Example of usage is restart dialog. Application must provide in argument vector question text, title text, buttons text, and buttons actions when calling this dialog.

6.2. ChoiceForm - FvwmScript

Similar as ActionForm, but button actions are not provided in command line, but signal about chosen action is sent to the calling program ("father" FvwmScript usually). Used only in Font Style Manager for now.

6.3. FilePicker - FvwmScript

A simple file pick open/save dialog. Copy of FVWM file picker, but with added option to display a file if file is an icon. It is a simple file browser with up and home shortcuts, path view and show/hide button for hidden (files starting with a dot) files.

Used in Backdrop Style Manager, Subpanel Manager, and Color Style Manager for adding backdrops, photos, icons and palettes.

6.4. InputForm - FvwmScript

Form with text field which asks user to name something. If OK is pressed, string is sent to the parent script for further processing. Used in Font Style Manager.

6.5. WaitNotice - FvwmScript

Short lived simple FvwmScript form, butonless and with a 3 slots for text. This dialog serves as short information if some NsCDE action is started which is not immediately obvious in a 1-2 seconds. It will appear in the middle of the screen with a bigger font and relief text and live between half of the second till 5 seconds. Depending with which text and duration time it was called by some function or other FvwmScript program. If clicked or receives return or escape key, it will dissapear immediately.

Used by Color Style Manager, SysActionDialog and a documetation view function `f_DisplayURL`.

6.6. Splash - FvwmScript

Splash screen is started at the beggining of the NsCDE session. It covers all the screen(s) and displays NsCDE logo in upper left corner, and on the lower left corner there is a info about system on which NsCDE is running, FVWM version used, licence, and startup message. Pressing Ctrl+C or clicking at the text exits Splash screen before it's timeout, which is 7-8 seconds. Splash is wrapped in the f_Splash function and can be called interactively during existing session by calling this function.

7. Backdrops, Palettes and Fonts

Together with Workspace Manager, Backdrops and Palettes are probably most recognizable aspects of CDE, by which CDE is visually distinguished from other desktop environments.

CDE backdrops, are relatively simple XPM textures and pictures consisting usualy from 2-5 *base* colors: *background*, *foreground*, *selectColor*, *topShadowColor*, and *bottomShadowColor*. This colors are taken from the current (or custom) palette and applied to the symbolic definition of colors in XPM templates. Backdrop is then generated, tiled and applied to the root window. Every workspace can have it's own backdrop texture. In 4 colors mode of the palette theme, they are all colored in the same pattern, while in 8 colors mode, every workspace from 1-4 has it's own color variant from the current palette, in that case, workspaces from 5 to 8 are repeating colors from 1 to 4. In other words, if there are more that four desks defined, color of the fifth workspace is repeated color of the first, sixth of the second and so on.

Here, in addition to original CDE textures, there are some 100 new and custom textures created from (free and public) textures which were convenient for this customization. In other words, NsCDE implements more than 100 backdrops, and with Backdrop Style Manager user can import to it's `$FVWM_USERDIR/backdrops` it's own backdrops, or put them there with terminal or file manager.

Backdrops must have alternative extension for X Pixmaps: that is, not *.xpm*, but *.pm*.

If one wants their custom backdrop to be dynamic with palette/theme of NsCDE, one must edit them to set symbolic names of the colors described above. Examining existing backdrops is the fastest way to get idea what needs to be done. Apart from symbolic names, backdrops also have a real color defined to be compatible with XPM specification, but values of this colors can be arbitrary, since they are not used if symbolic name on the same line is set. Usually, they are in gray spectrum.

Default backdrops are set from the `$NSCDE_DATADIR/defaults/backer` until user does not redefine/set his own with Backdrop Style Manager. Default palette is *Broica* in 8 colors variant.

Thematically generated backdrops generated by Backdrop Style manager are installed in user's `$FVWM_USERDIR/backer/DeskN-<backdropname>.pm` and defined in `$FVWM_USERDIR/Backdrops.fvwmgen` as colorsets of *TiledPixmap* type. NsCDE reserved FVWM colorsets numbers for backdrops are from 31-38 for all eight possible workspaces. `$FVWM_USERDIR/Backdrops.fvwmgen` file is read by FvwmBacker(1) FVWM module. It is automatically generated when user makes first change with Backdrop Style Manager.

Until then, system wide file `$NSCDE_DATADIR/fvwm/Backdrops.fvwmconf` applies, which itself reads pre-generated and pre-defined backdrops from the `$NSCDE_DATADIR/defaults/backer` directory. Default color theme is *Broica*.

We can consider backdrops as the source form or template files, and when processed with color values from the palette, this backdrop's final form is, ready to be set by FvwmBacker(1).

When NsCDE is used with FVWM3 and *per-monitor* or *shared* DekstopConfiguration monitor modes, FvwmBacker(1) is not in use, because it cannot treat different physical (and virtual) monitors separately. Functions driven by the *new_desk* FvwmEvent(1) event are called in that case, which will be loaded from the cache directory `$FVWM_USERDIR/backer/bgcache` if found, or automatically generated with the help of ImageMagick's convert(1) tool if they do not exist, and then loaded. This mechanism automatically calculates screen width, height, X and Y coordinates, desks and backdrops to produce cjoined backdrops as one big PNG file which is then loaded to show correct background for desks on different monitors.

8. Configuration files explained

As pointed above, NsCDE has a number of configuration files. Approximately 75% of them are the FVWM configurations, where a line between using FVWM directives for it's core and modular functionality as a desktop-building material and tool, and merely configuring aspects for end use are wastly blurred and cannot be really distinguished and separated in that strict categories. This system of configurations is arranged in some logical and consistent way. For example, keyboard shortcuts in Keybindings.fvwmconf, FvwmBacker configuration in Backer.fvwmconf, (generated) colorsets in Colorset.fvwmgen etc.

All this configurations are included from the Main.fvwmconf. This is the starting FVWM configuration which sets core options and safe defaults, and reads the rest of the configuration files which are included there. It defines StartFunction which starts all additional modules and calls important things during start or restart of the Window Manager. System Wide configuration files are located in $NSCDE_DATADIR/fvwm and $NSCDE_DATADIR/defaults, while user local hooks or user complete overrides are stored in `$FVWM_USERDIR`.

This is default list of system-wide configurations:

8.1. FrontPanel.actions

A non-FVWM file. Lines in this file are default actions and icons for Front Panel. This file is parsed by the **fpexec** and **fpseticon** shell script tools. All or individual entries from this file can be overridden by creating `$FVWM_USERDIR/FrontPanel.actions` file. This is a CSV-like file (comma is a field separator), and it defines buttons of the Front Panel, their actions and icons.

File format is:

- Button Number (Btn1, Btn2, BtnN ...)
- Icon path (FVWM relative from ImagePath)
- Mouse Button (3 mouse buttons for 3 different actions if needed)
- Program executable to check for or *NOCHK* for check avoidance
- Actions (commands) with options and arguments to the end of the line

If Icon Path field is `__APPLET__` for mouse button 1, then in the check/nocheck field an applet program may be defined for FvwmButtons based Front Panel to be swallowed instead of an icon. In this field, direct FvwmButtons(1) syntax must be manually written with this exceptions:

- Comma (,) must be replaced with a pipe (|)

- Double quotes (") must be escaped by the two backslashed (\\)

If the swallowed applet has it's own action on mouse buttons clicks, then declaration of the applet must be prefixed with a `ActionIgnoresClientWindow`. In this case, last filed with the actions can be "Nop" because Front Panel will not accept this clicks (it may accept them if possible transparent area around applet exists and is clicked). If the applet is not clickable, a standard FVWM Action or command can be put in the last field to make a click on the applet useful.

Here is the example of the custom swallowed applet. This one was initially created for the Window Maker window manager:

```
Btn10,__APPLET__,M1,
"ActionIgnoresClientWindow| Swallow (Respawn) \\"WmstickynotesApplet\\"
\\"Exec exec wmstickynotes\\"",Nop

Btn10„M2,FVWM,Nop

Btn10„M3,FVWM,f_FrontPanelPropsMenu 10 APPLET
```

Here is the example of the custom swallowed applet which does not accept mouse clicks:

```
Btn10,__APPLET__,M1,
"Swallow (Respawn) \\"MyFavoriteApplet\\"
\\"Exec exec mfa -s\\"",Exec exec vlc

Btn10„M2,FVWM,Nop

Btn10„M3,FVWM,f_FrontPanelPropsMenu 10 VLC
```

This example is replacing standard Front Panel Clock with Solaris Globe Icon based Front Panel clock, using the same pclock program as standard one, but with a slightly different options:

```
Btn1,__APPLET__,M1,"Swallow (Respawn) \\"pclock\\"
\\"Exec exec $NSCDE_TOOLSDIR/$(uname -s)_$(uname -m)/fpclock
-H red -S white --hands-width=4
--hour-hand-length=15 --minute-hand-length=20
--second-hand-length=22 -w
-B $NSCDE_DATADIR/icons/NsCDE/SDtEarth.l.xpm\\",Exec exec firefox

Btn1„M2,FVWM,Nop
```

```
Btn1„M3,FVWM,f_FrontPanelPropsMenu 1 Browser
```

Notice that button definition for mouse M1 (first line) is split in 3 lines in this examples, but in the `FrontPanel.actions` must be written as one line.

After editing this file (system-wide or user's) nothing needs to be reloaded because file is read from the `f_FrontPanelAction` function on every click on every icon on the Front Panel. There is no GUI tool for fully managing this file yet, but currently, "Copy to Main Panel" action from the submenu will edit `FrontPanel.actions` and put choosen entry there, binding it to some button.

8.2. AppMenus.conf

This file defines which custom menu entries will be appended on the Window Options menu when this menu is called by titlebar button 1. This is for example used by all known terminal applications to implement Wide Terminal menu entry. By default, **Watch Errors** and **Fvwm Diagnostic Console** have appended appropriate entries for conveniently call each other. Also, some of the known File Managers has this entry for opening new window of the same type. Entries in this file are the following comma separated values (syntax):

- X11 Window Class
- X11 Window Resource
- Menu item title (and optionally keyboard shortcut after two TAB's)
- FVWM Exec or function command, module or action to be executed

This file exists as the `$NSCDE_DATADIR/defaults/AppMenus.conf`, but it can be extended by creating and writing `$FVWM_USERDIR/AppMenus.conf` file.

8.3. Animate.fvwmconf / Animate.fvwmgen

FVWM Animate Module configuration. Animate module is started by NsCDE by default automatically, but with *None* as a default effect. This can be reconfigured by the user in private `$FVWM_USERDIR/Animate.fvwmgen` with Window Style Manager, or in static manual configuration `$[FVWM_USERDIR]Animate.fvwmconf`. No effect (None) is choosen as default for increased CDE similarity, because CDE doesn't have iconification animation effects.

8.4. Backdrops.fvwmgen

This file defines 8 colorsets for all (maximal) 8 desktops as a *TiledPixmap* colorset type. In the system configuration, static non-generated configuration (.fvwmconf) defines pre-generated default backdrops of default *Broica* color scheme. When user makes the first change with Backdrop Style Manager, user's private copy of this file is created in `$FVWM_USERDIR`. In NsCDE, colorsets 31 - 38 are reserved for backdrops (or png, xpm photos).

8.5. Backer.fvwmconf

Rarely needed in `$FVWM_USERDIR`. FvwmBacker(1) configuration which defines 8 maximum desks and refers them to 8 colorsets from 31 - 38. Option `RetainPixmap` is defined in case user wants to use X compositor such as compton(1) with NsCDE.

8.6. Colorset.fvwmgen

Definition of all colorsets minus colorsets 31 - 38 which are reserved for the backdrops. System-wide file has predefined color values for default color scheme (Broica), while user's file in `$FVWM_USERDIR` is created on first change made with Color Style Manager. Apart from FVWM colorsets, this file exports in environment two variables: `NSCDE_PALETTE` with the name of the color palette used in generation of the file, and `NSCDE_PALETTE_NCOLORS` which is either 4 or 8, depending which color variant has been used in Color Style Manager.

8.7. NsCDE.conf

This file defines various FVWM and NsCDE defaults. NsCDE.conf can be considered as the main user's configuration file. System wide configuration are static defaults which can be loaded by Window Style Manager or by erasing user's copy of the file. User's copy of the `NsCDE.conf` contains all options (minus FvwmAnimate) from Window Style Manager's set of options, but it has some options such as FVWM *infostore* variables for default terminal and file manager applications, graphical editor, and such. Infostore variables `desknum`, `pagematrixX` and `pagematrixY` are managed by the Workspace and Pages Manager while `menudclicktm` infostore variable is managed by the Pointer Style Manager. In `NsCDE.conf`, defaults for page edges, focus, icons, and such are defined. See Section 5.17 and fvwm(1) for details. Since this is read by FVWM, user can set in this file local variables and additional configuration options if needful, which are not covered in other parts of the configuration. While applications are taking great care with long regexp lines to parse and write this file, if edited manually, user is advised to keep it clean: use proper capitalization as it is described in fvwm(1), without line breaks and if possible, surplus spaces and tabs. Comments are allowed as usual: as lines which begins with # sign.

Some of the important FVWM Infostore variables which can be set in the `NsCDE.conf` are:

- InfoStoreAdd filemgr `file manager of choice`
- InfoStoreAdd xeditor `GUI text editor of choice`
- InfoStoreAdd calculator `GUI calculator of choice`
- InfoStoreAdd taskmgr `Task manager of choice`
- InfoStoreAdd printmgr `Printer manager of choice`
- InfoStoreAdd browser `WWW Browser of choice`
- InfoStoreAdd mailreader `Mail application of choice`
- InfoStoreAdd terminal `terminal` - emulator app of choice by default commented out and figured out by the list of known terminals. It can be set here.
- InfoStoreAdd sandboxmode 0|1 - reduced NsCDE for embedded Xephyr X jails
- InfoStoreAdd desklastpage 0|1 - remember last visited page on desk
- InfoStoreAdd wsmcolored 0|1 - additional menu color around Workspace Manager like in some versions of CDE
- InfoStoreAdd nscde_use_xscreensaver 0|1
- InfoStoreAdd nscde_use_stalonetray 0|1
- InfoStoreAdd nscde_use_dunst 0|1
- InfoStoreAdd nscde_use_rofi 0|1
- InfoStoreAdd nscde_use_xsettingsd 0|1
- InfoStoreAdd wsm.eco 0|1
- InfoStoreAdd frontpanel.pos.placement `screen c 50-50w -0p ewmhiwa`
- InfoStoreAdd stalonetray.pos.placement `-0 -0 ewmhiwa`
- InfoStoreAdd gkrellm.pos.placement `-0 +0 ewmhiwa`
- InfoStoreAdd polkit.agent `Path of the choosen PolicyKit Agent`
- InfoStoreAdd loc_pg.desk_scale `Zoom factor of the Local Pager`
- InfoStoreAdd glob_pg.desk_scale `Zoom factor of the Global Pager`
- InfoStoreAdd windowlist.fontsize `small | medium | large`
- InfoStoreAdd windowlist.title.fontsize `small | medium | large`
- InfoStoreAdd fvwm3_default_logging `0|1` weather to log FVWM3 messages by default from the startup
- InfoStoreAdd xlogcmd `custom-command` if `$HOME/.xsession-errors` is not in use, for example `"journalctl -u gdm -n 300 -f"`
- InfoStoreAdd rootpagerposition `"screen c 50-50w +10p"` - where to put local pager on page change if enabled
- InfoStoreAdd desktopconfiguration global|per-monitor|shared - for FVWM3

Further, common system environment variables are provided already set, or for optional uncommenting if needed or desirable:

- `QT_QPA_PLATFORMTHEME` - set to "qt5ct"
- `HAS_WINDOWNAME 1` - uncomment and set it to 1 if FVWM is patched with additional FvwmButtons NsCDE patches. On FVWM3, this is default
- `GTK_OVERLAY_SCROLLING 0` - handy to turn off irritating blinking of scrollbar area in GTK3 applications
- `SetEnv GTK_CSD 0` - if you have misfortune to must use some of the GNOME 3 applications and have gtk3-nocsd installed
- `f_VarAppend : LD_PRELOAD /usr/local/lib/libgtk3-nocsd.so.0` - if you have misfortune to must use some of the GNOME 3 applications and have gtk3-nocsd installed
- `SetEnv NSCDE_REDRAW_WORKAROUND 1` - Uncomment this if you are having problems with FvwmScript PopupMenu widgets under compton, compton-ng or picom in the form of not refreshing their part of the screen.

8.8. Event.fvwmconf

FvwmEvent(1) module configuration. In this file a single instance of the FvwmEvent called `MainLoop` is defined. It passes ID (Window ID, desk etc ... depending on context) for window manager actions. `Cmd` option is empty: FVWM functions are used for all defined actions. The following events are observed and reacted by functions Currently:

- `new_desk`
- `new_page`
- `add_window`
- `destroy_window`
- `focus_change`
- `enter_window`
- `leave_window`
- `configure_window`
- `iconify`
- `deiconify`
- `res_class`
- `map`

Event Functions from `Functions.fvwmconf` are triggered on the events above. This serves Workspace Manager, Page Manager (PGM) and window placement functions in an important way. If redefined or disabled, things will start to break. It can be extended by the user to suit the needs in `$NSCDE_USERDIR/Event.fvwmlocal`, but here also care must be taken, because complex functions, or calling slow and/or resource hungry commands from that functions can make FVWM (and hence NsCDE) dramatically slow and even unstable.

8.9. Font-<NSCDE_FONT_DPI>dpi.fvwmgen

... where "DPI" ($NSCDE_FONT_DPI) is by default hardcoded to 96.

This files are regular files in the system-wide configuration, but may be symbolic links to some system wide or local fontset in `$FVWM_USERDIR` when generated by Font Style Manager.

Font sizes in this config are defined as infostore variables and used across FVWM config files, they are provided to FvwmScript programs with **getfont** wrapper. Font sizes are in points. While defining them in pixels (pixelsize=) will be easier, and all this care about DPI will not be needed, integration with GTK2 and GTK3 in best of my knowledge and research does not provide a way to define fonts in pixel sizes, so either font sizes in points or unsure recalculation (again based on DPI) will be needed while writing gtk settings.

8.10. Form.fvwmconf

Defaults for fonts, colorsets, cursors etc of FVWM form module. FvwmForm is reading this. FvwmForm is not used actively by NsCDE anymore, This file and definitions in it are provided only to enforce colors and fonts in accordance with the current NsCDE theme if any custom installation decides to use forms again.

8.11. FrontPanel.fvwmX.fvwmconf

Main NsCDE Front Panel configuration file. Here, FvwmButtons is configured under the alias `*FrontPanel`. Special care is taken to place most of configurable parts out of this file, so it doesn't have to be forked into `$FVWM_USERDIR`, but this option nevertheless exists. Here, all geometry, buttons, subpanels, default icons, frames and widgets are written and put in place. This configuration, together with swallowed WSM (Workspace Manager) is probably the most recognizable part of the setup which provides us with familiar and desired CDE look - a Front Panel. FvwButtons FrontPanel configuration is non-trivial, but it is very trustworthy mimicking the original. Icon actions which user wants to change here can be overridden with `FrontPanel.actions` file and Subpanels which are also described here. Swallowed apps and "widgets" are in most part already described in sections above.

In NsCDE 2.0, FrontPanel has double configuration: in two separate configuration files. One is for fvwm2 (`FrontPanel.fvwm2.fvwmconf`) and the second for fvwm3 (`FrontPanel.fvwm3.fvwmconf`) a "front" file `FrontPanel.fvwmconf` contains the code for auto detection of FVWM (patched fvwm2 will use fvwm3 file if detected).

8.12. Functions.fvwmconf

Another important part of the configuration. Almost all FVWM functions are defined here, except 5-6 of

core functions in `Main.fvwmconf` which are reading the rest of the configuration. They are sorted in logical groups and are used widely in almost every part of the configuration, and particularly from the FvwmScript scripts. Main groups of NsCDE FVWM functions are:

- Core Window Operation Functions
- Front Panel functions
- Misc core functions
- Functions called from FvwmEvent `MainLoop`
- Functions for generating menus
- Placeholders for functions aimed for user to override
- Functions used in NsCDE FvwmScripts
- Upgrade Functions

For a FVWM function description see fvwm(1), in this file there is a plethora of examples, and for user usage is the most interesting part placeholders for functions which are here merely for programs to not complain about missing them and which should be overridden in user's local extension `$FVWM_USERDIR/Functions.fvwmlocal` - this extension file will be read by the main configuration immediately after processing `Functions.fvwmconf`. This functions are:

- `f_CheckMail`: called by CheckMailApplet on the FrontPanel on click and periodically. This is the place where some script can be called and with SendToModule to "1 1" (widget 1, routine 1) icon of empty mailbox will be changed to the icon of the full mailbox.
- `f_Calendar`: called by MonthDayApplet on click. Can be used to call external calendar application, to focus Thunderbird with lightning extension or whatever user finds useful.
- `f_Mixer`: unused currently.
- `f_AddCustomToRootMenu`: add custom entries in a convenient point of the root menu which is called by the right mouse button on the root window.
- `f_UserChangeDesk`: called when current active workspace changes
- `f_UserChangePage`: called when current active page changes
- `f_UserRestartFunction`: hook called when NsCDE restarts
- `f_UserEnterWindow`: hook called when window is entered
- `f_UserLeaveWindow`: hook called when focus leaves window

Another useful function is conditional execution function `f_WarpOrExec`. It takes 3+ arguments. First is the window name or class (or icon, resource) name, second is the binary to check in `$PATH`>, and 3rd to the rest of the command line is what to execute with all arguments included. If window with name from arg1 is already present on `$DISPLAY`, it will not be executed, but pointer will be simply pointed to that window. If window was iconified, or function called from another workspace or page, window will be deiconified, and workspace and/or current page changed to one where existing window is residing.

It is pointless and bad idea for this file to be overridden by the local copy of the *conf* (fvwmconf, not fvwmlocal) file, because a lot of things depends on this functions. Regular update of NsCDE software version will almost certain make setup with missing or incompatible functions.

8.13. IconMan.fvwmconf

If Use Icon Box option is selected in the Window Style Manager, infostore variable `iconbox` will be defined as non-zero, and FvwmIconMan(1) module will be started on login from the `Main.fvwmconf`. This file, `IconMan.fvwmconf` contains default configuration of that module.

8.14. Ident.fvwmconf

Module FvwmIdent(1) is called either from a small menu which can be popped up with middle pointer click on a titlebar, or from the root window version of the Window Options menu. This is FvwmIdent's configuration file. It simply defines colorset and font for the FvwmIdent's module window.

8.15. Init.fvwmconf

Most probable candidate for copying to `$FVWM_USERDIR`. Here are defined start, quit and restart function (sessionless and session-managed) which are internally recognized by FVWM during certain important actions. `InitFunction` or `SessionInitFunction` is the place to put all programs and actions user wants to be executed during NsCDE startup. In system-wide default configuration there are already conditionally defined some probable applications and there are hints and examples for user to customize this further.

8.16. Keybindings.fvwmconf

According to the old CDE documentation, there are couple of key bindings known under CDE, but that documentation also assumes Motif application bindings and X resource configuration, which is far from universally applicable in today's world. That said, while GTK and Qt applicatios have it's own key bindings, NsCDE tries to combine a bit of CDE legacy with a huge number of new functions, originally not present in CDE. Hence, we have a rich set of keybindings for almost any action in NsCDE.

NsCDE is using 4 modifier keys in single and multi-modifier combinations with other keys to implement key bindings. This modifiers are:

- Control
- Shift
- Meta (often called Super)
- Alt

In combinations with escape, function keys, tab, backspace, home, menu, letters, numbers and other keys, NsCDE builds rich set of keyboard shortcuts also called keybindings which tries to achieve some golden middle between ergonomics position on the keyboard for often used ones, and logical grouping for easy memorizing actions.

Keybindings can be partially or complitely overriden by the user's writing `$FVWM_USERDIR/Keybindings.fvwmconf` (override) or `$FVWM_USERDIR/Keybindings.fvwmlocal` (extend, paritally override). Key bindings which are described with their *Mod+Key* items on menus can be extended or overwritten in `$FVWM_USERDIR/Keymenu.actions`.

Notice when there is a reference to the "Menu" key this means also "Compose" on some keyboards.

If some unwanted or clashing keybindings needs to be cleared, for example Shift+Tab from the any context, use the following FVWM syntax:

```
Silent Key Tab A S -
```

For explanation what is the context, and what modifier, see FVWM explanation (copied from original default FVWM config and extended a bit). Namely:

- cursor keys up, down, left and right with ctrl modifier are moving viewport from page to page in any context.
- cursor keys up, down, with Alt modifier are cycling window focus and raises or lowers them on the current page in all contexts except icon
- cursor keys up, down, left and right, with meta (mod4) are moving viewport by 4% of the screen. (Ctrl moves 100%)
- the same cursor keys as above, but with shift modifier moves pointer by 1% of the screen
- Meta+Alt+I in any context apart from icon context will move focus and pointer to icons on the currect page (if any). Here, Up and Down keys can be used to browse icons. In the context of icon itself, this keybinding will choose last non-iconified window and move off focus from icons.
- Up/Down in the context of a icon focuses first icon above or below of the current selected and focused icon. Key Space will bring main icon menu in the context of the focused icon.
- I in the context of the icon deiconifies that icon.
- Menu (Compose) key, if pressed twice in a time window of two seconds pops up root menu in any context. On keyboards without Menu (Compose) key, combination Alt+ISO_Level3_Shift does the same thing
- Meta+Menu combination pops up root Window Operations Menu On keyboards without Menu (Compose) key, combination Meta+ISO_Level3_Shift does the same thing

- Meta+Alt+Menu pressed twice, brings up page menu on the Front Panel and Local Pager in the middle of the screen if enabled. On keyboards without Menu (Compose), combination Meta+Alt+U does the same thing
- Meta+Alt+Home will spawn or raise and focus if already active - a Graphical Workspace Manager or GWM.
- Meta+Alt+N Cycles window focus between all windows on the current page, raises them, and moves pointer to them
- Meta+Alt+B Same as Meta+Alt+N but in backward direction
- Meta+Shift+N Cycles window focus between all windows on the current workspace, raises them, and moves pointer to them
- Meta+Shift+B Same as Meta+Shift+N but in backward direction
- Meta+Alt+J Moves pointer to the focused window if pointer is not already there
- Space in the icon frame context pops up icon-specific contextual menu
- Key Meta+Alt+Insert will give a focus to the last opened window
- Key Meta+Alt+BackSpace will give a focus to the previously focused window
- Shift+BackSpace If pressed twice in a time frame of two seconds will call LocalPager in any context. This Pager will disappear soon as it looses a pointer focus, or if keybinding is pressed again once, while pager is still visible.
- Shift+TabCalls visual GlobalPager, a stripped down version of GWM (Graphical Workspace Manager) which shows matrix of pages and workspaces. This keybinding works in any context, but it must be called twice to avoid accidental invocation. When Global Pager is visible, calling it second time will replace it's window with full version of standard GWM window tool until it is not closed with a third invocation of this same keybinding. In stripped down semi-transient mode, this pager, if not called as full GWM, disappears shortly after losing mouse focus to free up space on the screen.
- Alt+Tab is cycling trough pages of the active page of the current workspace from up to down and then right up to down
- Meta+Tab is cycling trough the all workspaces (desks)
- Meta+Shift+Tab is reverse cycling trough the all workspaces (desks)
- Meta+Alt+L activates screensaver, that is, locks the screen
- Meta+Alt+X Executes "xrandr --auto"
- Meta+Alt+W will call f_CleanRestoreWorkspace function. This will iconify and put into invisible group all iconified windows on the active page. Repeating this action once again, restores all iconified windows back in place quickly and without animation. Windows which were iconified before calling this action or from windows started and iconified after this action are not affected by this function. This is NsCDE smart version of the "show desktop" functionality. Individual windows from the group can be deiconified too, which will sign off this windows from the group.
- Key Meta+Alt+Min the context of the window, frame corners, frame sides and a title bar pops up Window Options context menu
- Key Meta+Alt+O - same as Meta+Space if called in the window context.

- Key Meta+Alt+R - Vertically tiles windows of the current page in a grid, making them "grow" maximized state (See Alt+F7 and Alt+F8).
- Key Meta+Alt+H - Horizontally tiles windows of the current page in a grid, making them "grow" maximized state (See Alt+F7 and Alt+F8).
- Meta+Alt+E Again executes last selected item from the last used subpanel. From the Sun keyboard, this can also be accomplished with Meta+Sun Again (Redo).
- Meta+Escape Will cycle trough focus-accepting windows on the current page of the current desk, avoiding CirculateSkip windows and the FrontPanel.
- Alt+Escape Will open WindowList in the middle of the screen for the current workspace (desk) if pressed twice, local WindowList will be replaced with global WindowList
- XF86PowerOff on Sun keyboards (most upper right) calls System Action Dialog with system suspend (S3) option selected. For this to work, ACPI action needs to be configured on the system. Otherwise, this will likely initiate direct system shutdown.
- Ctrl+XF86PowerOff on Sun keyboards (most upper right) calls System Action Dialog with system shutdown option selected. For this to work, ACPI action needs to be configured on the system. Otherwise, this will likely initiate direct system shutdown.
- Alt+XF86PowerOff on Sun keyboards (most upper right) calls System Action Dialog with system reboot option selected. For this to work, ACPI action needs to be configured on the system. Otherwise, this will likely initiate direct system shutdown.
- Meta+XF86PowerOff on Sun keyboards (most upper right) calls System Action Dialog with X Session logout option selected. For this to work, ACPI action needs to be configured on the system. Otherwise, this will likely initiate direct system shutdown.
- Help key on Sun keyboards if pressed twice in a second, in the context of the root window will call PDF viewer (if any) with complete (this) NsCDE documentation.
- Sun Front key on Sun keyboards acts as a Alt+F6 - Raise or Lower the window.
- Sun Find key on Sun keyboards calls f_Find NsCDE FVWM function which has to be user defined to be usefull.
- Sun Props key on Sun keyboards will call Style Manager window when pointer is on the root window.
- Sun Meta+Props key on Sun keyboards will call Style Manager window.
- Sun Open same as Alt+F12 invokes Exec dialog or Rofi launcher if configured with nscde_use_rofi infostore variable in the `$FVWM_USERDIR/NsCDE.conf`
- XF86AudioLowerVolume and XF86AudioRaiseVolume on Sun type 6 or 7 keyboards will lower and raise sound volume by 1 percent, or by 10 percent if combined with Ctrl modifier. This works by default on systems on which pactl(1) PulseAudio tool is installed. On other systems, f_Mixer function has to be redefined. See `$NSCDE_DATADIR/fvwm/Keybindings.fvwmconf` for f_Mixer options and interaction.
- XF86AudioMute key on Sun keyboards will mute the sound. Function f_Mixer expects PulseAudio system. See previous item for possible alternatives.
- Shift+BackSpace if pressed twice in a short time frame will spawn Local Pager under the pointer. Pressend second time just once, it will dismiss Local Pager

- Key Meta+Space in the context of the window, frame corners, frame sides, title bar and icon (ovoids root window context!) calls Occupy Workspace dialog for window moving between the desks.

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Key Alt+Space in the context of the window, frame corners, frame sides, title bar and icon (ovoids root window context!) calls Occupy Page dialog for window moving between the desks.

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Key Meta+Alt+Space in the context of the window, frame corners, frame sides, title bar and icon (ovoids root window context!) calls Occupy Monitor dialog for window moving between the monitors. This key binding is available only when multiple monitors are detected by NsCDE and FVWM.

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Alt+F1 regenerates and refreshes the window

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+F1 Calls xrefresh(1) command to refresh the X11 display

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Alt+F2 iconifies (deiconifies if in icon context)

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+F2 "shades" or rolls up the window to titlebar only view

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Alt+F3 Repositions the window according to saved geometry (see Geometry Manager), or if there is no record for a window in `GeoDB.ini`, places again window with FVWM **PlaceAgain** command

Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+F3 Centers the window in the middle of the screen where mouse pointer currently resides, together with frame

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Alt+F4 enters resize mode which can be finished with cursors keys and enter

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+F4 will resize and enlarge any resizable focused window by 25 pixels in all directions and place it again on page

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+Shift+F4 will resize and shrink any resizable focused window by 25 pixels in all directions and place it again on page

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Alt+F5 enters move mode which can be finished with cursors keys and enter

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+F5 In the context of the window and it's frames places that window in a full sticky state. That is, it occupies all workspaces and pages. Pressing this key combination again, or pressing Meta+Alt+F5 plus Meta+Shift+F5 combinations will put window back from sticky state

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+Alt+F5 In the context of the window and it's frames places that window in a state sticky across workspaces. That is, it occupies all workspaces, but not pages Pressing again the same combination toggles this state back

Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+Shift+F5 In the context of the window and it's frames places that window in a state sticky across pages. That is, it occupies all pages, but not workspaces Pressing again the same combination toggles this state back

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Alt+F6 raises or lowers the window

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Alt+F7 maximizes window 100% (whole screen + decorations); when pressed again it maximizes to 86% (stretch), third press will put window in the grow state (Maximized on page up to the first obstacle), while third press will put window into normal state. We can say how window is cycling between maximized, stretched, grow and maximized states. This is a cyclic keybinding in a "shrink" direction.

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+F7 Works like Alt+F7, but selects every second operation. In practice, from restored state goes to stretched and back, and from grow state goes to fully maximized and back.
- Alt+F8 grows window up to the first obstacle on page, when pressed again, it maximizes window ~ 86% - avoids Front Panel; when pressed further, it maximizes to 100% (whole screen + decorations); third call to Alt+F8 restores window into normal state. We can say how window is cycling between normal, grow, stretched and maximized state. This is a cyclic keybinding in a "grow" direction.

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+F8 is the same as Meta+F7. This variant is presented on Window Operations menu for actions when going towards bigger maximized state, while Meta+F7 is presented for actions which are shrinking back.
- Alt+F9 is empty

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+F9 in the context of the known terminal application windows scratches the window to some 75%x72% of the screen, which is also a menu option in this windows called *Wide Terminal*
- Alt+F10 deletes a window (see fvwm(1))

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+F10 closes a window

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+Alt+F10 forcefully destroys a window

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Alt+F11 is empty

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+F11 is empty

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Alt+F12 invokes Exec dialog or Rofi launcher if configured with nscde_use_rofi infostore variable in the `$FVWM_USERDIR/NsCDE.conf`

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+F12 invokes default terminal app ($[infostore.terminal])

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+Alt+G Calls Window Geometry Manager dialog which saves geometry information for a current window in the GeoDB.ini

Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+Alt+F in the context of a window, frame, or corner, puts a current window into a real fullscreen mode. Calling this keybinding again, restores a window.
- Meta+Alt+DIn the context of a window calls FVWM Ident module which presents dialog with various X11 and window manager attributes on the screen
- Meta+Shift+Print takes a screenshot of the root window with 3 seconds delay. Screenshots in PNG format are saved in `XDG_PICTURES_DIR` or `$HOME` if `XDG_PICTURES_DIR` is not defined.

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Meta+Control+Print pops cross cursor to pick selected screen area for screenshot. Screenshots in PNG format are saved in `XDG_PICTURES_DIR` or `$HOME` if `XDG_PICTURES_DIR` is not defined.

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Shift+Print takes a screenshot of the root window with 3 seconds delay. Screenshots in PNG format are saved into clipboard for pasting into compatible X11 applications.

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Control+Print pops cross cursor to pick selected screen area for screenshot. Screenshots in PNG format are saved into clipboard for pasting into compatible X11 applications.

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Shift+Escape will do the same as Ctrl+Escape, but Front Panel is after repositioning shaded to bottom border of the screen. Invoking this key sequence again will unshade the Front Panel. Middle mouse button on the borders of the frame has the same effect

  Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- Ctrl+Escape will raise Front Panel and reposition it to it's default place on the screen. If key combination is pressed for the second time, focus and pointer focus will be transferred to the Front

Panel, enabling Meta+[0-9] keys to function on subpanels and Ctrl+Return on icons of the Front Panel itself.

Dynamic: defined in `$NSCDE_DATADIR/defaults/Keymenu.actions` for automatic description on menus.

- In the context of the Front Panel, Meta+ 1-10 will pop up subpanels 1 to 10 and warp pointer to it
- In the context of the any subpanel, Meta + 1-10 will pop down this subpanel and warp pointer to Front Panel
- In the context of the Front Panel and any subpanel, Meta+Return will activate button under the pointer and execute FvwmButton action defined in configuration of that button.
- In the context of the Front Panel and any Subpanel, Sun Help key will display Front Panel or Subpanels documentation.

8.17. Keymenu.actions

This file is not FVWM configuration file. It is written manually (with editor), and read by the **$NSCDE_TOOLSDIR/keymenu** command which will generate infostore variables with descriptive keyboard shortcut names which are used in NsCDE menus from `Menus.fvwmconf` and `Functions.fvwmconf`, and additionaly, use the whole line after the keyword in the first column with "Silent Key" prefix to dinamically generate this part of keybindings which are separated from `Keybindings.fvwmconf` and processed specially because they definition must automatically match keyboard binding description in various menus.

The syntax of this file is simple: first column is the keyword which becomes infostore variable (km_xrefresh as `$[infostore.km_xrefresh]`), and the rest columns of every line is FVWM "Key" sintax which is paired with "Silent Key" prefix during initialization with `f_KeyMenu -a` from `Main.fvwmconf`.

Every line, or dynamic keybinding can be overriden here if (re)defined in user's `$FVWM_USERDIR/Keymenu.actions`: whole file or just choosen keybindings. The goal is to get their descriptions (after functions and menus reload) automatically in place on menus. For example, "F3 A M" configuration for FVWM "Key" will become Alt+F3 description right of the (De)Shade item on titlebar left button popdown menu.

File Keymenu.actions is processed by **$NSCDE_TOOLSDIR/keymenu** script. This script generates FVWM infostore variables used in menus, and executes FVWM "Key" commands producing dynamic keybindings which are described in menus.

For a list of keybindings that are "dynamic", that is, not defined in `Keybindings.fvwmconf`, but in `Keymenu.actions`, see notices in section about Keybindings.fvwmconf.

8.18. Main.fvwmconf

Core configuration. This file is read as first initialization upon starting FVWM Window Manager. In fact, FVWM is invoked with `-f /path/to/Main.fvwmconf` to read it instead of default FVWM system configuration or user's `~/.fvwm/config`. This invocation and configuration in `Main.fvwmconf` changes everything: it avoids `~/.fvwm` if user has a plain (normal) FVWM setup, defines and redefines FVWM internal variables and sets `NSCDE_ROOT`, configures some core FVWM options (like `DefaultIcon`), defines main FVWM **Read** command wrapper `f_ReadCfg`, sets desktop names, reads `NsCDE.conf` depending if user has it, or system-wide one, defines `StartFunction`, `DesktopSize`, and then reads most of the files described in this documents, composing NsCDE FVWM configuration. Care must be taken NOT to read this file with `f_ReadCfg`, because it will end up in the endless CPU hogging loop because `f_ReadCfg` will be destroyed and recreated during it's own execution. This file should never be overridden in `$FVWM_USERDIR`. It is the init(8) of the NsCDE system.

8.19. Menus.fvwmconf

In NsCDE, there is a bunch of the menus. Root menus, contextual menus, and even menus called or dinamically generated from the FvwmScript(1) scripts.

- MenuFvwmRoot built-in root menu of FVWM. As in CDE, menu of such type, it is called with a right click on the root window.
- m_Applications - main and dynamically generated menu with a FVWM python script **fvwm-menu-desktop** (contains flat list of apps and icons for Front Panel's subpanels). It reads `/etc/xdg/menus/<desk>-applications.menu`
- m_QuickMenu beneath m_Applications. Empty by default. Intended to be destroyed and recreated by the user (`Menus.fvwmlocal`) with own favorites.
- m_NsCDEHelpMenu beneath Quick Menu. Contains links to this documentation in PDF and HTML forms.
- m_NsCDEMenu on the top of generated Applications menu. Contains entries to various the NsCDE internal tools. in PDF and HTML forms.
- m_MoveToPage: invoked from the 1st titlebar button. Moves window to the specified page on desk
- m_MoveToWorkspace: as m_MoveToPage, but moves across desks. Invoked with middle pointer on the 1st titlebar button
- m_ControlFrontPanelMenu: Front Panel specific functions. Invoked from the left top control panel menu button on right pointer click
- m_DeiconifyOnPage: submenu of the main icon menu invoked with the 1st pointer click on desktop icon, deiconify window on another page
- m_FrontPanelWinMenu: Front Panel flavor of the *Window Operations* menu. Invoked from the top left menu button of the Front Panel
- m_IconM2: calls small menu with FvwmIdent, xwininfo and xprop if icon is clicked with a middle pointer button

- m_IconOps: main icon menu invoked with a click on the icon. Contains Icon flavor of the "Window Operations" menu, submenus (see above) and deiconify action
- m_SubpanelWindowOps: a Subpanel flavor of the Window Operations menu. Invoked from the Subpanel's titlebar left (and only) button contains also contextual actions Refresh Subpanel and Subpanel Settings
- m_TitleBarM2: Middle pointer click on titlebar. Invokes a small menu which gives FvwmIdent, xwininfo, xprop, and two to four types of screenshot of the window.
- m_WindowOpsStandard: main menu of *Window Operations* invoked with a mouse click on left titlebar button. It has extended (More ...) and reduced (Less ...) version.
- m_WindowOpsExtended: changes m_WindowOpsStandard from default reduced, to extended view (More ... option on Window Operation menu)
- m_RootMenu: Small root menu with options to call Invoked with a click on the root window. It has menu items for invoking workspace-local window list, global window list, local pager, global pager, Window Options menu, and to refresh all windows on the screen.
- m_WindowOpsRootWin: Standalone root window version of the Window Operations menu intended for a general and tricky circumstances. It is invoked from the m_RootMenu, and keybindings Ctrl+Meta+Space and Meta+Menu.

8.20. Mousebindings.fvwmconf

File `Mousebindings.fvwmconf` is done in similar manner as the `Keybindings.fvwmconf`. Can be overridden (.fvwmconf) or extended (.fvwmlocal) just like (almost) any NsCDE conf file. See fvwm(1) for "Mouse" commands. Here commands invoked with pointer are defined. As it is the case with key bindings, mouse actions are too far more in NsCDE than in original CDE. The syntax is described at the top of the file.

Actions are:

- Titlebar 1st (left) button left click: invokes Window Operations menu, double click closes (Delete) a window
- Titlebar 1st (left) button middle click: Shortcut for calling Occupy dialog with Go with the window selected by default.
- Titlebar 1st (left) button left click: invokes extended or full version of the Window Operations menu, double click closes (Delete) a window
- Titlebar 2nd (right) button left click: Iconify Window
- Titlebar 2nd (right) button middle click: No Operation
- Titlebar 2nd (right) button right click: Shade (Roll up/down) Window
- Titlebar 3rd (rightmost) button left click: (Un)Maximize Window 86% or up to the Front Panel on the bottom of the screen. Double click (un)maximizes 100%, covering Front Panel and it's EWMH panel area

- Titlebar 3rd (rightmost) button middle click: (Un)Maximizes 86% and makes window sticky or unsticks it depending on window's initial state
- Titlebar 3rd (rightmost) button right click: (Un)Maximizes 86% and makes window raise or lower depending on window's initial state
- Titlebar left click: Moves window on move, raises/lowers on click
- Titlebar middle click: pops up menu with functions to take a screenshot of the window, identify with info (FvwmIdent), xprop(1), and xwininfo(1)
- Titlebar right mouse button quickly raises or lowers a window
- Pointer actions 4 and 5 (mouse wheel) are shading and unshading (rollup, rolldown) a window
- Left pointer click on border or corner raises or lowers a window while move action will interactively resize the window
- Middle pointer click on border or corner will only do lower/raise action, without resize on pointer movement
- Right pointer click on border or corner also does lower/raise action, but on pointer movement moves the window
- Left pointer click in combination with control on border or corner calls a root window version of the Window Operations menu - this rare and border functionality is aimed for difficult situations where there is no other option easily available
- Middle pointer click in combination with control on border or corner refreshes the window
- Right pointer click in combination with control on border or corner calls root applications menu (MenuFvwmRoot) - this rare and border functionality is aimed for difficult situations where there is no other option easily available
- Left pointer click on icon calls m_IconOps menu, while double click action deiconifies a window
- Middle pointer click on icon calls m_IconM2 menu
- Right pointer click on icon directly deiconifies a window
- Left click on the root window calls Root Menu whith handy shortcuts for calling visual pagers and window lists. Submenu "Window Operations" will show Window Options actions.
- Left double click on the root window calls Window Operations.
- Middle click calls FVWM `WindowList` super-menu see fvwm(1) for `WindowList`
- Right click calls main root menu MenuFvwmRoot
- Pointer actions 4 and 5 (mouse wheel) will scroll between current up and down pages of the current workspace by 2% if pointer is in the context of the root window
- Control+ left mouse click will call f_CleanRestoreWorkspace function. This will iconify and put into invisible group all iconified windows on the active page. Repeating this action once again, restores all iconified windows back in place quickly and without animation. Windows which were iconified before calling this action or from windows started and iconified after this action are not affected by this function. This is NsCDE smart version of the "show desktop" functionality.
- Control+ middle mouse click will call extended version of the WindowList with additional window info such as page number and window geometry.

- Left mouse button on border of the shaded Front Panel context will de-shade and reposition Front Panel
- Right mouse button on border of Front Panel will shade or de-shade Front Panel

8.21. WspLocPager.fvwmconf

Configures FvwmPager(1) type which is called as Local Pager from the right-click popup menu on Workspace Manager buttons. This pager is transient and will disappear after being used with a pointer click.

8.22. LocalPager.fvwmconf

Pager which is showing only a current workspace. It's windows and pages on thish windows reside. This pager is meant to be primary of informational nature. Infostore variable `pageraisefp` in `NsCDE.conf` is by default 0. If enabled (1), when active page or workspace changes, visual FvwmPager(1) will be shown in the center of the screen near the top of it by default. On Ctrl+Compose (Ctrl+Menu) and/or Shift+ISO_Level3_Shift (Right Alt), pager will move to the position of the pointer, it will eventually disappear from the screen after 1 second, 3 seconds, 5 seconds, 8 seconds, or 10 seconds if it looses the focus, or it can be dismissed by pressing the same key combination once more while pointer is above pager. Local pager can also be called with Shift+Backspace keys. Local pager's initial appearing position on the active screen can be controled with `rootpagerposition` FVWM InfoStore variable in the `$FVWM_USERDIR/NsCDE.conf`. There, a timeout can also be set with `rootpagertimeout` variable; rootpagertimeout is in milliseconds.

8.23. Script.fvwmconf

Some bare defaults for all FvwmScript(1) Module based applications and widgets: script Path, default font and colorset. All this values are usually redefined outside of this file, so in FvwmScript source body, nothing really should be changed here.

8.24. WSM.conf

Non-FVWM config file. Here, the names of the NsCDE Workspaces are read and written by the NsCDE and can be changed manually or with the Workspaces and Pages Style Manager. Options for Graphical Workspace Manager (GWM) are also saved here. Names of the workspace pages (for the menus) can be redefined here manually (there is no tool for this one).

WSM file is not read from the system's central location. It is installed there only for a reference. `WSM.conf` will be created in `$FVWM_USERDIR` when workspaces are renamed, or GWM settings are

changed. Overrides for names of the pages are user's care if there is really need to name pages differently from default scheme.

8.25. Style.fvwmconf

Main decoration configuration. `Style` '*' is applied globally. This is the main source of CDE and Motif-like look and behavior. If user wants to preserve CDE-like look and feel, this options should not be changed too much. Otherwise, a plain FVWM configuration can be done which can drastically differ from NsCDE, since FVWM has much more options and variants for a huge number of tastes. `Style` '*' options are partially overridden or extended in `NsCDE.conf` which can be generated with Window Style Manager or simply copied from `$NSCDE_DATADIR/fvwm` to `$FVWM_USERDIR` and edited to suit.

Options in Style.fvwmconf are grouped in 7 categories:

- Default, or '*' styles
- Fvwm modules and FvwmScript(1) script specific
- Some basic sane defaults for common applications
- Menu styles (not a style commands, but styles anyway)
- Cursor styles (not a style commands, but styles anyway)
- Decors "SubpanelsDecor" and "FpDecor" which are slightly modifiying Front Panel and it's Subpanels.
- GeometryWindow directives (FVWM3 only)

This styles can be extended and/or overridden by the user's own `$FVWM_USERDIR/Style.fvwmlocal`, and core style for all windows *(*)* can be overriden by creating `$FVWM_USERDIR/Style.override` which extends or changes directives for core style from `$NSCDE_DATADIR/fvwm/Style.fvwmconf` prior to applying particular application styles and prior to reading `$FVWM_USERDIR/Style.fvwmlocal`.

Colorsets and fonts used in this configuration are generated and stored in `Font-$DPIdpi.fvwmgen` and `Colorset.fvwmgen`.

Man page fvwm(1) has a rich and extended description of what can be done with a huge set of `Style` commands.

8.26. Subpanels.actions

This file is not FVWM configuration file. It is written by the Subpanels Manager, Subpanel Settings, or manually with editor, and read by the **$NSCDE_TOOLSDIR/generate_subpanels** command which will generate `Subpanels.fvwmX.fvwmconf` file in user's `$FVWM_USERDIR` directory, where "fvwmX"

is either fvwm2 or fvwm3 depending under which FVWM NsCDE is running. The syntax of this file is simple. It is CSV-like file where values are delimited with a comma ",". Every line belongs to one of the ten subpanels. Comma and "'" characters cannot be part of the field values. This values are:

- S<X>: where <X> is a number from 1 to 10 indicates which subpanel's line is this
- NAME, WIDTH, ENABLED, ENTRY: second line indicates subpanel's display name, subpanel's width regarding font and long application names on the menu, state of enablement, and entries defined for this subpanel. ENTRY lines can be multiple (as much as screen resolution allows), other values must be unique for every subpanel.
- For NAME, WIDTH and ENABLED, there is only a third parameter: for a NAME the name of the subpanel, WIDTH is an integer (reasonable values: 120 - 260), and enabled is boolean 1 or 0.
- For ENTRY lines, there are fields application title, check type, icon path and name, and command with arguments fields that must be defined. Title is name of the entry. For example "Firefox" or "Workstation Info". Check type can be one of "FVWM-M" for FVWM module, "FVWM-F" for FVWM function, "OTHER" (currently unused), CHECK:<appname> where <appname> is the command which should be checked for existance instead of the first string of the command field, empty space (nothing between commas: „), and NOCHK which indicates that no check for a command existance should be done in `Subpanels.fvwmconf`. Most of the entries will default to *empty* which will prepend `Test (x <appname>)` to the entry specification in the resulting FvwmButtons(8) config. Icon is full path of the icon file (32x32) which should belong to the application. and the rest of the line is application's calling command, possibly with options and arguments.

8.27. Subpanels.fvwmX.fvwmconf

The files `Subpanels.fvwm2.fvwmconf` and `Subpanels.fvwm3.fvwmconf` are generated by the **$NSCDE_TOOLSDIR/generate_subpanels**. It's configuration is static in the system directory after NsCDE installation, but customizable and easly generated in the user's `$FVWM_USERDIR` where it is then called `$FVWM_USERDIR/Subpanels.fvwmX.fvwmgen`, where "fvwmX" is either FVWM2 or FVWM3, depending on which FVWM is used by the NsCDE. It contains FvwmButtons(8) definitions of all 20 possible subpanels which can be popped up from the Front Panel. There are 3 ways to regenerate this file: Subpanels Manager tool called from the *Install Icon*, Subpanel Settings tool called from the titlebar popdown menu on every subpanel as *Subpanel Settings* or manually by calling **$NSCDE_TOOLSDIR/generate_subpanels** which will read user's or system `Subpanels.actions` for every subpanel and if it exists, user's one; take this one while generating Subpanels.fvwmX.fvwmconf. In system default, subpanels 2, 5, 6 and 8 are disabled by default, but can be activated with a very quick tripple middle pointer click on the empty launcher without up arrow: a Subpanel Settings application will appear on the screen which has a checkbox "This Subpanel is Enabled" which will be checked out by for disabled subpanels by default, and can be checked in, to apply OK will enable subpanel with initial system defaults for name, width and application entries.

9. System and User NsCDE Tree Layout

This section describes in detail what is stored where in NsCDE system-wide installation hierarchy, and user's home directory `.NsCDE` or `$FVWM_USERDIR`.

9.1. System Tree Layout

From the major version 2, NsCDE is following Filesystem Hierarchy Standard of Linux and other surviving Unix and BSD systems. The installation procedure will put main calling wrapper *nscde* and **nscde_fvwmclnt** into PREFIX/bin, NsCDE helper programs and tools into PREFIX/libexec/NsCDE, python api code and FvwmScript scripts, fpclock and colorpicker binaries into PREFIX/lib/NsCDE, while rest of the data goes into PREFIX/share/NsCDE, except icon theme, .desktop files, which are following XDG specification and are installed in /usr/share/icons/NsCDE, /usr/share/applications, and so on. NsCDE app menu will be positioned in PREFIX/etc/xdg/menus (or /etc/xdg/menus if installed from package), while X session file read by the graphical display manager usually goes into /usr/share/xsessions, regardless of the PREFIX, or /usr/local/share/xsessions on BSD-based systems.

Variable meanings for further paths:

- `$NSCDE_ROOT`

  Main top directory of NsCDE installation; usually /usr, /usr/local, /opt/local, /opt/sfw, /usr/pkg etc ... most parts of the installation are under this hierarcy referenced also as "$PREFIX" in documentation

- `$NSCDE_DATADIR`

  This points to $PREFIX/share/NsCDE

- `$NSCDE_TOOLSDIR`

  This is sometimes $PREFIX/libexec/NsCDE and some systems, which doesn't like libexec will put this as $PREFIX/lib/NsCDE, $PREFIX/lib64/NsCDE and so on

- `$NSCDE_LIBDIR`

  This is usually $PREFIX/lib/NsCDE with variations as $PREFIX/lib64/NsCDE, $PREFIX/lib/x86_64-linux-gnu/NsCDE and so on.

This are the main directory components of the NsCDE with description what is what, and what is where:

- `bin`

  nscde start wrapper called from `/usr/share/xsessions/nscde.desktop` nscde_fvwmclnt caller of the right (FVWM2 or FVWM3) FvwmCommand.

- `$NSCDE_DATADIR/fvwm`

  NsCDE private FVWM settings directory. All .fvwmconf files described in documentation are here. They are read from `Main.fvwmconf` which is called from PREFIX/bin/nscde by fvwm binary with `-f` directly.

- `$NSCDE_DATADIR/config_templates/progbits`

  Template X pixmap files used by Color Style Manager for producing user's copy in the `$FVWM_USERDIR/icons/NsCDE/` this pixmaps are invalid as pictures in their source form since they contain internal macros for replacement with real colors. System starting theme is using their copied in `$NSCDE_DATADIR/icons/NsCDE`.

- `$NSCDE_LIBDIR/python`

  Python libraries used by **themegen**: part of the integration suite for GTK and Qt theme.

- `$NSCDE_TOOLSDIR`

  The rest of the scripts (korn shell and python) are located here. In normal circumstances this scripts should not be run directly, but they are used by numerous NsCDE FvwmScript apps and FVWM functions as helpers and background program workers.

- `$NSCDE_DATADIR/backdrops`

  Backdrop files. CDE and new, additional. Source for generation of active user's backdrop depending on theme, that is color scheme. They have `.pm` extension instead of `.xpm`. Bitmap files `.bm` (`.xbm`) are not supported by style managers and hence some of CDE's original backdrops of that type are in NsCDE converted to X pixmaps.

- `$NSCDE_DATADIR/icons/cursors`

  Custom cursors which are missing on plain X server installations but can be found in CDE. Referenced in `$NSCDE_DATADIR/fvwm/Style.fvwmconf`.

- `$NSCDE_DATADIR/defaults/backer`

Default generated backdrops for first start (Broica, 8 colors) Referenced in system's `$NSCDE_DATADIR/fvwm/Backdrops.fvwmconf`.

- `$NSCDE_DATADIR/defaults/pages`

  Default page names for every possible combination supported by NsCDE

- `$PREFIX/share/doc/nscde`

  Documentation

- `$PREFIX/share/doc/nscde/examples`

  Examples for X display manager and DE integrations, **sudo** for **shutdown reboot**, **pm-suspend** or **pm-hibernate**, **Gkrellm** NsCDE skin.

- `$NSCDE_DATADIR/fontsets`

  Default font sets used by the Font Style Manager

- `$NSCDE_DATADIR/icons/NsCDE`

  CDE and custom NsCDE icons of which many are part of FvwmScript programs and applets

- `$NSCDE_DATADIR/palettes`

  CDE palettes plus a bunch of new custom palettes. Used by Color Style Manager and Backdrop Style Manager, as well as **$NSCDE_TOOLSDIR/themegen**, **$NSCDE_TOOLSDIR/backdropmgr**, **$NSCDE_TOOLSDIR/colormgr**, **$NSCDE_TOOLSDIR/palette_colorgen**

- `$NSCDE_DATADIR/photos`

  A couple of nice free photos collected and resized for various screen resolutions. Can be used instead of backdrops (selectable from Backdrop Style Manager) or in `$HOME/.xscreensaver` for some screensavers which are loading photos.

- `$NSCDE_DATADIR/config_templates/app-defaults`

X resources for a particular X applications (like **XTerm**) which are processed by the Color Style Manager for user to be put into `$FVWM_USERDIR/app-defaults` (if enabled). Referenced by the usual `XAPPLRESDIR` environment variable.

- `$NSCDE_DATADIR/integration/gtk2_gtk3_qt`

  Part of the CDE theme which are used by **$NSCDE_TOOLSDIR/themegen** and the rest of $NSCDE_LIBDIRY/python/*.py to generate `$HOME/.themes/NsCDE` with a selected palette and color depth.

  $NSCDE_DATADIR/config_templates also contains `Xdefaults` and some include files for it, as well as configuration for **stalonetray** which is installed on initial setup, and `BGdefault`, which is a monochrome pixmap loaded as bare default early on start, before FvwmBacker(1) sets up backdrops on each user's workspace. From there, on initial setup, default `Xsettingsd.conf`, `Stalonetray.conf`, `Dunst.conf`, `colormgr.local` and possibly other files are installed into user's `$FVWM_USERDIR`.

- `$NSCDE_DATADIR/integration/firefox`

  Mozilla Firefox CSS which integrates Firefox Web Browser with the current NsCDE theme.

- `$NSCDE_DATADIR/integration/thunderbird`

  Mozilla Thunderbird CSS which integrates Thunderbird Mail Reader with the current NsCDE theme.

- `$NSCDE_DATADIR/integration/rofi`

  Template theme for rofi(1) window switcher, launcher, ssh dialog and dmenu replacement

9.2. User Tree Layout

User part of configuration is located in `$HOME/.NSCDE` - this place is what is referred as `$FVWM_USERDIR` in this documentation. If user has a plain FVWM configuration in `$HOME/.fvwm` it will not be used in any way and can co-exist with NsCDE. Here is the simple layout of things in `$FVWM_USERDIR`:

- `app-defaults/` directory:

  X resources referenced by the usual `XAPPLRESDIR` environment variable. Files inside are (will be) generated by the Color Style Manager

- `backdrops/` directory:

  If created, user can put custom backdrop sources here, and they can than be selected by the Backdrop Style Manager and processed with current or custom color scheme.

- `photos/` directory:

  User's photos which can be used instead of backdrops if selected in Backdrop Style Manager or configured in `Backdrops.fvwmgen` manually.

- `backer/` directory:

  Generated backdrops referenced by Colorsets 31-38 for FvwmBacker. This is what is displayed on the screen background, different for every workspace.

- `backer/bgcache/` subdirectory:

  If FVWM3 is used by NsCDE in non-global monitor *DesktopConfiguration* mode, Large PNG files are generated from the existing backdrops in `$NSCDE_DATADIR/defaults/backer`. This directory contains automatically generated hierachy structure of background cache for multiple different X RandR monitors when NsCDE is used with FVWM3 in non-global `DesktopConfiguration` mode, which are matching various combinations of desks across monitors. Referenced in function f_SetFvwm3NonGlobalBackground of `fvwm/Functions.fvwmconf` and in `$NSCDE_TOOLSDIR/xrandr_backer` backdrop autocreate script.

- `fontsets/` directory:

  If created, user can put or generate with Font Style Manager own fontsets here.

- `icons/` directory:

  Populated by dynamic menu action **fvwm-menu-desktop**. If directory does not exist, script will create it.

- `icons/NsCDE/` directory:

  NsCDE custom icons. Put here by Color Style Manager and the rest of the tools. Since icons from here are referenced with a relative path, whatever is missing here, will be loaded from system's `$NSCDE_DATADIR/icons/NsCDE` automatically.

- `libexec/` directory:

  If created, `colormgr.local` script can be written and put here, as well as `fontmgr.local` and other user's hooks.

- `palettes/` directory:

  User can put custom palette files here, and they can than be selected by the Color Style Manager and processed for a preview or applied as new theme.

- `templates/` directory:

  Here, local subdirectory of `app-defaults` with tmpl files can be optionally created. Also, it is a good choice for Gkrellm or other files processed by the `libexec/colormgr.local`

- `tmp/` directory:

  Place used by parts of the NsCDE and in particular NsCDE's FvwmScript programs for temporary generated files for previews, or as scratch and work directory. Tools are usually taking care to cleanup their garbage from tmp/ on exit.

- `XYZ.fvwmconf` files:

  Absolute overrides of `$NSCDE_DATADIR/fvwm/XYZ.fvwmconf` files. If in existence, they will be read instead of system defaults. *XYZ* is here placeholder/example for Style, Functions, Keybindings, Init, Menus etc ... For some groups of FVWM configuration points this makes a sense, for most it doesn't see next description for such examples.

- `XYZ.fvwmlocal` files:

  Extensions, added values of `$NSCDE_DATADIR/fvwm/XYZ.fvwmconf` files. If in existence, they will be read right as the continuation of their .fvwmconf main configurations from system (or local) directory. This is preferred way to extend functionality or override something not big enough for a complete "fork" of the config file. Colorset, Backdrops, Animate, Font-$DPIdpi, Init, and Subpanel are exception of this, that is, it is preferred (if not only thing possible) to have it as *.fvwmconf* files only, and not *.fvwmlocal* files.

  `Style.override` file:

  If created, it will be read by the `$NSCDE_DATADIR/fvwm/Style.fvwmconf`. Here, core style for all windows (*) can be overriden prior to reading particular NsCDE application style overrides and

addons, and prior to reading `$FVWM_USERDIR/Style.fvwmlocal`. Here, advanced users can override or extend core styles which are not customizable by the Window Style Manager. See fvwm(1) for a huge list of *Style* options. Putting core *Style* options after NsCDE applications in the `$NSCDE_DATADIR/fvwm/Style.fvwmconf` are processed can nullify overrides for that particular applications, this is why this mechanism is provided. This file is not created in `$FVWM_USERDIR` by setup procedure or Style Managers, but if created (manually) it will be read and processed.

XYZ is here placeholder/example for Style, Functions, Keybindings, Init, Menus etc ...

- `NsCDE.conf` file:

  managed by Window Style Manager, Workspaces and Pages Style Manager, Pointer Style Manager and users own editor manually. See the rest of the documentation. `NsCDE.conf` may be considered the most important user configuration NsCDE file.

- `WSM.conf` file:

  read/written by Workspaces and Pages Style Manager, and Graphical Workspace Manager (GWM) Options dialog, and user's favorite editor. Not an FVWM config file. System default of this file is `$NSCDE_DATADIR/defaults/WSM.conf`, and it is not actively used, because defaults are hardcoded in WSM and GWM.

- `FrontPanel.actions` file:

  user's overrides/addons for Front Panel icons and actions. Written by editor manually, and by Copy to Main Panel option of the context submenu of subpanel's items. See section about Front Panel for more information.

- `GeoDB.ini` file:

  part of the Geometry Manager functionality. Written and read by the **$NSCDE_TOOLSDIR/confset** and **$NSCDE_TOOLSDIR/confget** on Save Geometry" and Reposition from Extended and Standard Window Operations menus. Windows-like *ini* files are WAY nicer than dconf and such binary registry-like facilities. See detals about Geometry Manager for more information.

- `Xdefaults` file:

  Read on startup by xrdb(1).

- `Xdefaults.local`, `Xdefaults.fontdefs`, `Xdefaults.mouse` files:

Included with preprocessor directives from `Xdefaults`

- `Xset.conf` file:

  Configuration (a batch file or shell script basically) with xset(1) parameters for system beep, pointer, keyboard, and DPMS settings which are managed by their respective Style Managers. User can put here **setxkbmap**, **xgamma** and such additional X server configuration commands (or whatever one likes). Care must be taken not to mess lines beggining with *#XYZMgr,xxxx* till *#end*, since this is internal marker of FvwmScript's buggy `WriteToFile` function.

- `NsCDE-Sandbox.conf` file:

  If exists, used only in bare sandbox mode, where basic functionality of the NsCDE is needed, and not full DE-like environment.

10. Installation Dependencies

For NsCDE to work, essential software is FVWM Window Manager. Almost all is based on it. Since NsCDE is heavy user of `infostore` internal variables and other new features of FVWM, development has been done on FVWM versions 2.6.7 and 2.6.8. At this time, this are recommended, if not mandatory versions of FVWM for NsCDE. Other dependencies, that is, software used by NsCDE is:

- Original Korn Shell 93 or newer. All shell script routines inside configuration, helper scripts and FvwmScript helpers are written with ksh. It is known for sure that pdksh cannot be drop in replacement, and in tests on Arch Linux with mksh it became clear that mksh cannot replace Korn Shell. Korn Shell is available and it is free.
- *Xorg utils* (Fedora/CentOS RPM xorg-x11-utils) - **xdpyinfo**, **xprop** ...
- Util *xdotool* - used by FVWM2 which is not patched with WindowName patch for the FvwmButtons. Not used by FVWM3.
- *ImageMagick* - convert(1), display(1), resize(1) ... really needed.
- *Xscreensaver* - optional, but Screen Style Manager functionality will not work without it. Something needs to be installed for locking the screen.
- *cpp* - C preprocessor for **xrdb** functionality - for X resources integration. Used by xrdb(1).
- *xorg-x11-server-utils* (CentOS, Fedora name) - xrdb, xset, xrefresh mandatory for startup, some style managers and menus.
- *python-yaml* - needed for python part of the color theme management and for Gtk+Qt integration.
- *PyQt4* or *PyQt5* (or possibly python-qt4, python-qt5 ...) This is unfortunate dependency which is further dependent on Qt libraries. NsCDE tries to have as less as possible dependencies, specially indirect (dependencies of dependent dependencies of dependencies ...). Gtk/Qt integration is borrowed

from CDEtheme Motif/CDE theme project and adapted for use with FVWM (instead of heavy Xfce dependency) or standalone engine. In part of the `Theme.py` code, some png pixmaps are cut and colored with functions from this API. With present job and lack of time, there was no time to do this without PyQt4 or PyQt5 for the first public release.

- *Gtk2, Gtk3, Qt4, Qt5, qtconfig-qt4, qt5ct, qt5-qtstyleplugins* (optional) There is a great chance this libraries and some useful programs using them are already installed on user's system. If Gtk and Qt integration is activated in Color Style Manager, there is no point not to have it installed.

  Notice about Qt4 and Qt5: **qt4-config** (or **qt-config**) and **qt5ct**: Although colors will be applied, for font setting to take effect, qtconfig-qt4 (or qtconfig) must be run, something changed back and forth, and then applied/saved - no matter that you will see fonts of your choice already selected. This can be considered a bug. Same goes for Qt5.

  Notice about Qt5: `QT_QPA_PLATFORMTHEME` environment variable must be set, and be set to `qt5ct` value in order to run **qt5ct** configurator. This variable is set by default from the `$FVWM_USERDIR/NsCDE.conf`.

- Recommended fonts for as close as possible CDE look are *DejaVu Serif* for variable, and *DejaVu Sans Mono* for monospaced fonts. Check should be made if this fonts are installed on the system. For Solaris CDE look, *Lucida Sans* and monospaced *Lucida Sans Typewriter* should be installed, selected and used instead. (optional)
- **Stalonetray** for "tray" facility (optional)
- **Dunst** for notification daemon (optional)
- **xterm**
- *python3*
- *python36-pyxdg* or **python3-pyxdg**
- *libstroke* (FVWM2)

11. Installation

In it's original non-packaged form, NsCDE release or gut clone directory result is in unconfigured source form. NsCDE uses autoconf and automake to preprocess it's scripts, programs and configurations, and compile 3 helper binaries. Dependencies are checked during **./configure** phase, but errors are only produced if something needed for final product is missing. That is, it is assumed that this will be built on some build machines which doesn't have installed all runtime dependencies of the NsCDE.

Much fine grained control over what is needed in build time, and what in run time is acomplished when NsCDE is packaged as RPM, DEB or Arch package. Fully functional example RPM *spec*, Debian DEB *control* and Arch *PKGBUILD* files are provided in `pkg` of the NsCDE source distribution. Thanks to the "-tb" feature of RPM, RPM package can be built straight from tarball as **rpmbuild -tb NsCDE-2.X.tar.gz**.

A generic installation instruction should look familiar:

```
$ su - || sudo -i
# umask 0022
# cd /tmp
# wget https://github.com/NsCDE/NsCDE/archive/NsCDE-<version>.tar.gz
# tar xpzf NsCDE-<version>.tar.gz
# cd NsCDE-<version>
# ./configure
# make
# make install
```

Example above will install NsCDE into /usr/local. However, it is possible to install to some other place. For example, this will install NsCDE in /opt/sfw, but put XDG menu file in /etc/xdg/menus.

```
# ./configure --prefix=/opt/sfw --sysconfdir=/etc
# make
# make install
```

For a latest master tree from development page on Github, NsCDE can be fetched with git(1) and upgraded with fetching changes in the master or some other branch. An example of this would be:

```
# git clone https://github.com/NsCDE/NsCDE.git
# cd NsCDE
# ./configure
# make
# make install
```

Upgrades, either from tarball or from git, once new version is downloaded can be made by uninstalling and installing NsCDE with make(1) tool. Example of upgrade:

```
# cd NsCDE
# make uninstall
# git fetch
# git reset --hard @{u}
# ./configure --sysconfdir=/etc
# make
# make install
```

Running only **make uninstall** will uninstall NsCDE.

12. NsCDE Startup

Session can be started from the `$HOME/.xsession` in last command line as **exec nscde** or **ssh-agent nscde** or with **gpg-agent**, **lxsession** or whatever.

If supported by the X Display Manager which is in use on the target system, an xsession file `nscde.desktop` will be put during the installation into `/usr/share/xsessions` (or in whatever place your system and your X Display Manager reads this files) and afterwards NsCDE can be selected from the display manager's menu or similar session selector. See the rest of the X Session Manager integration examples in the directory `$PREFIX/share/doc/nscde/examples/` for MATE, KDE, LXDE and similar DE integrations and play with this if you like.

13. NsCDE Localization

NsCDE implements localization capabilities provided by FVWM and system C library. Directive `LocalePath` in `$NSCDE_DATADIR/fvwm/Main.fvwmconf` is set to look at `$[NSCDE_ROOT]/share/locale;NsCDE:$[NSCDE_ROOT]/share/locale;NsCDE-Subpanels:+`. In other words, whole NsCDE reads it's localization from ".mo" files from `$[NSCDE_ROOT]/share/locale/$LANG/LC_MESSAGES/NsCDE.mo` except subpanels which have their localization in separate mo files in the same directory.

All FvwmScript based applications are reading their own "mo" file from `$[NSCDE_ROOT]/share/locale/$LANG/LC_MESSAGES/NsCDE.mo` which is defined in every program's header with `UseGettext` directive.

All parts of visible text provided by NsCDE have localization ability.

LANGUAGE and LC_MESSAGES environment variables must be set for localization to work correctly. Best way to do this are user's login and profile files, but it can also work with parameters choosen with display manager's popdown menu or similar pre-login mechanism, or as a fallback, `$FVWM_USERDIR/NsCDE.conf` can be used to assign value of the LC_MESSAGES (or LC_ALL) and LANGUAGE commands.

First translated language from original English is Croatian. Other translations are welcome and expected. There are some pitfalls and things which must be observed when translating to the new languages. Existing .po files in "hr" can serve as a template for new translations since just msgstr lines must be replaced and msgid's are already there.

For more information about localization see $PREFIX/share/doc/nscde/README.localization README file which is also provided with software in the root of the unpacked tarball.

14. Initial Configuration

Upon the first (successful) start, `~/.NsCDE`, that is `$FVWM_USERDIR` is created, and only icons subdirectory is created as **fvwm-menu-desktop** is run. User will be presented with a first run NsCDE setup wizard in plain default xterm terminal emulator and with default color theme *Broica* in 8 colors. If Gkrellm and pnmixer programs are installed, on the system and found, they will be run too. Stalonetray and Dunst will be run if installed and enabled by the user during the wizard process. Stalonetray and Dunst colors and fonts will be handled internally by Color Style Manager and Font Style Manager when theme or font is changed and the same goes for Gkrellm, rofi, Firefox, Thunderbird or any 3rd party program integrated with NsCDE color and font theme.

Initial setup wizard is a simple script (**$NSCDE_DATADIR/bootstrap**) from the terminal which will run automatically and will set up the following:

- X resources in `~/.NsCDE`
- Default background color (pre-FvwmBacker) from default theme
- Default `~/.NsCDE/NsCDE.conf`
- Default `~/.NsCDE/GeoDB.ini`
- `~/.icons/default/index.theme` (default X cursor scheme)
- `~/.gtkrc-2.0`
- `~/.config/gtk-3.0/settings.ini`
- `~/.themes/NsCDE`
- `~/.config/Trolltech.conf`
- `~/.config/qt5ct/qt5ct.conf`
- `~/NsCDE/Stalonetray.conf if it does not exist yet`
- `~/NsCDE/Dunst.conf if it does not exist yet`
- Visual Pager NsCDE addon Option
- Disable or enable XscreenSaver
- Disable or enable use of the XSETTINGS xsettingsd daemon
- Disable or enable use of Rofi Launcher on Meta+F1 if installed
- Enable NsCDE to remember last used page on workspace if selected
- Enable FVWM3 diagnostic logging by default or not
- Configure favorite X file manager, if desired
- Configure favorite X text editor, if desired

There are more options in `$FVWM_USERDIR/NsCDE.conf` which can be edited after first setup finishes, and during the desktop operation.

Note that no file from the above list will be overwritten if it already exists in it's place. It will be skipped, but GTK and Qt theme integration files will be edited with **$NSCDE_TOOLSDIR/confset** to change theme and settings. This settings can be further edited with Color Style Manager, or with the tools from other desktop environments. After **bootstrap** script finishes setup, Color Style Manager will be run and user asked to confirm default theme or change it. *Do not* avoid this step, because some program bits are not fully setup if local user configuration uses fallback bare defaults, (like a clock background) and must be generated in the `~/.NsCDE/icons/NsCDE` directory.

After Color Style Manager's OK button is pressed, theme will be regenerated. Gtk and Qt themes will be regenerated only if their checkboxes in Color Style Manager are checked in. This is default since NsCDE 1.0. Setup script after the finish will ask user to press RETURN to exit. This is for user's convenience to read output of the setup for informative and/or diagnostic reasons. It is advised that after this setup `$FVWM_USERDIR/NsCDE.conf` is opened and further edited, for example InfoStoreAdd internal FVWM variables for `terminal`, `browser`, `taskmgr`, `printmgr` to user's favorite programs for mentioned functions.

Layout of the `$FVWM_USERDIR` after the initial setup should look like this:

- app-defaults/
- backdrops/
- palettes/
- fontsets/
- templates/
- photos/
- backer/
- backer/bgcache
- GeoDB.ini
- icons/
- icons/NsCDE/
- Backdrops.fvwmgen
- Colorset.fvwmgen
- NsCDE.conf
- NsCDE.rasi
- tmp/
- Xdefaults
- Xdefaults.fontdefs
- Xdefaults.local
- Xdefaults.mouse

It is advised to logout and login from the X session after finishing initial setup and check if everything looks ok. Also, it is a good idea to start using programs from the menu and examine environment around, before customizing Subpanels, and Front Panel, running Style Manager (2nd button right of the Workspace Manager on the Front Panel) to customize other aspects of the interface. NsCDE is now ready for everyday work.

15. Diagnostic: X11, FVWM and Watch Errors

Most of the X11 Display Managers, such as XDM, sddm, gdm etc, are redirecting diagnostic output (standard error, stderr, file descriptor 2) into a file or some logging facility. This file is usually `$HOME/.xsession-errors`. When "Watch Errors" item is launched from default subpanel 7 (Desktop Settings), it will execute **xterm -e $[infostore.xlogcmd]**. If not redefined in `$FVWM_USERDIR/NsCDE.conf`, contents of the FVWM infostore variable `$[infostore.xlogcmd]` will be *"tail -300f ~/.xsession-errors"*.

If your X Display Manager is using different path or file name, this can be redefined in a way mentioned above. Beware that some X Display Managers (namely lightdm) are sending diagnostic output into `/dev/null`. In this case, no variable redefinition can get this output visible to the user.

Reading X, FVWM and NsCDE log output can be informative and helpful while solving possible unexpected or unexplainable problems with desktop setup.

Window Options menu (called by titlebar button 1, leftmost) of the "Watch Errors" window has a custom menu entry Fvwm Diagnostic Console which starts FVWM module FvwmConsole. FvwmConsole on the other hand has custom menu entry Watch Errors which calls Watch errors. When both windows are on the screen, this menu entries will simply transfer focus to each other. Both of this applications have custom Window Options Menu "NsCDE Process Manager" which will run the same named application on demand.

16. Integration with X resources and widgets

16.1. Integration of X resources

NsCDE is using it's own copies of `Xdefaults` and includes files for X resources integration in `$FVWM_USERDIR`. X resources are filled with this from **$NSCDE_ROOT/bin/nscde** main wrapper during startup as the part of session assembling. Variable `XAPPLRESDIR` is also adjusted to `$FVWM_USERDIR/app-defaults`. There can be problems while using certain X session managers or DE which are clearing environment on a startup, and in this cases user must take care to put environment from **nscde** wrapper in place after startup. Probably autostart job in `$HOME/.config/autostart` and select from Session Manager's app will do the job.

Special private paths for X resources are used in order not to mess with user's maybe existing resources and files. If wanted, custom app-defaults files can be placed in `$FVWM_USERDIR/app-defaults` or even better, `$FVWM_USERDIR/templates/app-defaults` and reworked for Color Style Manager integration, because if find in that directory, and with *.tmpl* extension, it will be processed in the same way as system files from `$NSCDE_DATADIR/config_templates/app-defaults/` and put in `$FVWM_USERDIR/app-defaults`.

Plain custom X resources can be put in `$FVWM_USERDIR/Xdefaults.local`. This file will not be overwritten by Style Managers. X resources integration is turned on by default in Color Style Manager.

16.2. Gtk2, Gtk3, Qt4 and Qt5

$NSCDE_TOOLSDIR/themegen with `$NSCDE_LIBDIR/NsCDE/python` and with `$NSCDE_DATADIR/integration/gtk2_gtk3_qt` are parts of the optional Gtk2, Gtk3, Qt4 and Qt5 integration suite. When run from the Color Style Manager or manually with the **$NSCDE_TOOLSDIR/themegen**, with proper options, this will produce `$HOME/.themes/NsCDE` directory with either or both Gtk2 and Gtk3 themes. `$HOME/.gtkrc-2.0` and `$HOME/.config/gtk-3.0/settings.ini` will be edited to point to this directory with `gtk-theme-name` option. Excessive button images on menus and buttons will be turned off of course.

If Qt4 and/or Qt5 integration is also selected in Color Style Manager, files `$HOME/.config/Trolltech.conf` and `$HOME/.config/qt5ct/qt5ct.conf` will be edited to use "GTK2" Qt theme engine. This means, there is no Qt4 and/or Qt5 integration without at least Gtk2 integration because Gtk2 theme in use is deciding what GTK2 Qt4 and Qt5 engine will display. For Qt5 integration, make sure *qt5-qtstyleplugins* (or something like that name) is installed: `platformthemes/libqgtk2.so` is needed.

All integrations of this type: GTK2, GTK3, QT4 and QT5 are turned on in the Color Style Manager.

16.3. Mozilla: Firefox and Thunderbird

From version 1.3 and further, NsCDE contains CSS code and icons which can be used from `$FVWM_USERDIR/libexec/colormgr.local` while applying new color style and theme, to generate colorsets also for either or both Firefox and Thunderbird.

For this to work, appropriate code or the whole `colormgr.local` from `$NSCDE_DATADIR/config_templates/colormgr.local` must be adapted into `$FVWM_USERDIR/libexec/colormgr.local`. Shell variable `FIREFOX_CHROMEDIR` should contain path to the `~/.mozilla/firefox/__PROFILE__/chrome.`, `FIREFOX_MAJVER_PFX` should be either "7x" or "9x" depending of major version of the Firefox is installed and used on the system. Firefox 8x belongs to 9x code base. The same as for `FIREFOX_CHROMEDIR`, variable `THUNDERBIRD_CHROMEDIR` should point to chrome directory of user's thunderbird profile. If chrome subdirectory in the user's

profile directory does not exist, it will be created by the `colormgr.local`. This will make appropriate entiries in `userChrome.css` and `userContent.css` to import `nscdeChrome.css`, `nscdeIcons.css` and `nscdeContent.css` from the `$NSCDE_DATADIR/integration/firefox` and/or from `$NSCDE_DATADIR/integration/thunderbird`. File `nscdeColordefs.css` will be generated in the chrome dir with color mappings to the current NsCDE theme. This wile is also imported in userChrome.css and userContent.css.

If the option "Run $FVWM_USERDIR/libexec/colormgr.local (if exists)" is not turned off in the Color Style Manager while applying a theme, CSS which makes Firefox and Thunderbird as to a high degree similar to Motif/CDE application, and in the right colors will be generated. Firefox and Thunderbird must be restarted for this changes to take a place.

In Thunderbird, because of the nature of mail, it is not 100% possible to use color 4 from the NsCDE theme as background/foreground combination, because this can give ugly and bad results with some amount of structured HTML mails. Hence, color for text is black, while background is from Solyaris theme (#fff7e9) which is yellowish, almost white.

Here is how chrome directory layout and files must look for thunderbird after `colormgr.local` makes a changes:

```
/home/user/.thunderbird/profile_test_009/chrome
|- nscdeColordefs.css
|- userChrome.css
|- userContent.css
```

```
$ cat /home/user/.thunderbird/Profile_1/chrome/userContent.css
@import url("nscdeColordefs.css");
@import url("/usr/share/NsCDE/integration/thunderbird/nscdeContent.css");

$ cat /home/user/.thunderbird/Profile_1/chrome/userChrome.css
@import url("nscdeColordefs.css");
@import url("/usr/share/NsCDE/integration/thunderbird/nscdeChrome.css");
@import url("/usr/share/NsCDE/integration/thunderbird/nscdeIcons.css");
```

This is how Firefox chromedir should look like, assuming NsCDE was installed in /usr as prefix and root of installation dir:

```
/home/user/.mozilla/firefox/testtest.default-release/chrome
|- nscdeColordefs.css
|- nscdeContent.css -> /usr/share/NsCDE/integration/firefox/ff9x_nscdeContent.css
|- userChrome.css
|- userContent.css

$ cat /home/user/.mozilla/firefox/testtest.default-release/chrome/userChrome.css
```

```
@import url("nscdeColordefs.css");
@import url("/usr/share/NsCDE/integration/firefox/ff9x_nscdeChrome.css");
@import url("/usr/share/NsCDE/integration/firefox/nscdeIcons.css");

$ cat /home/user/.mozilla/firefox/testtest.default-release/chrome/userContent.css
@import url("nscdeColordefs.css");
@import url("nscdeContent.css");
```

Great care has been taken to get working CSS which modifies Mozilla applications not to display light text on light background, or dark text on dark background, but this ever changing CSS mess should be considered of beta quality, so issue reports, specially those with ready made fixes are more than welcome.

16.4. Custom application integration

If **$FVWM_USERDIR/libexec/colormgr.local** exists, Color Style Manager will run it if it's checkbox is selected. This script program will be run with a full path of CDE palette file, followed by the number of colors selected in the interface (4 or 8). This can be useful for regenerating settings of applications which do not use X resources, and neither GTK nor Qt, but have support for some level of customization of this resources. Also "skins" for programs like **smplayer, audacious** and **Gkrellm** can be processed from custom **colormgr.local**.

In the directory `$NSCDE_DATADIR/config_templates/Gkrellm` is the complete NsCDE theme for the Gkrellm. File `$NSCDE_ROOT/share/examples/colormgr.local.example` can be used for this integration. There are also examples for **Gkrellm** and **mate-terminal**. Local script **colormgr.local** will most likely use **$NSCDE_TOOLSDIR/palette_colorgen** in some way.

17. Additional recommended software

NsCDE is basically a collection of configurations, themes and tools around FVWM and is a kind od "hybrid" desktop environment as unofficial definition and attempt to define it, user must choose some favorite and default applications such as X terminal emulator file manager, and X editor, which will then be provided to him in occations where programs of that type must be called.

Apart from this, since *system* tray concept has been introduced on X11 and is here to stay, user will need some standalone tray application. For this purpose, a logical and really great **stalonetray** (Stand Alone Tray) is more than adequate. When NsCDE configuration for stalonetray `$FVWM_USERDIR/Stalonetray.conf` is used, it will have this defaults: grid 3x3 and it's place will be in the bottom right corner of the screen. Stalonetray is not integrated into Front Panel because it's size cannot be known in all times: is it one button size, two, ten? It is growing and shrinking depending on

number of widgets or tray icons, and apart from that, this can significantly alter the precious CDE look of the Front Panel. A window with traditional mwm/dtwm borders and without title in corner of the screen is default in NsCDE. Alternative ideas are welcome.

There are some programs which needs to be run under escalated privileges. Usually as root. Some examples are firewall-applet(1) for managing firewalld on Linux systems, Wireshark etc ... For this purpose on Linux (and most probably BSD systems), so called *PolicyKit* or "polkitd" is used as a authenticator component, while on the client side, PolicyKit agent must be used. This will then prompt user for a password. Since there is no equivalent of this agent as standalone DE-indepentent program analogously to stalonetray(1) or dunst(1), probably the closest match and most often used is polkit-gnome-authentication-agent-1 which is usually installed in `/usr/libexec` and relatively without huge dependencies on some desktop environment. It can be started from profile, for example from InitFunction in `$FVWM_USERDIR/Init.fvwmconf` or by using dex autostarter from this very file (default Init.fvwmconf has an example for this).

Small python program *dex(1)* can be installed and used from InitFunction in `$FVWM_USERDIR/Init.fvwmconf`. This program can be configured to read system autostart files in `/etc/xdg/autostart` and/or user's from `$HOME/.config/autostart`. From here, PolicyKit agent, NetworkManager applet, Nextcloud agent, Firewall agent, pnmixer and similar can be started. Programs such as stalonetray(1), dunst(1), xsettingsd(1), xscreensaver(1) are not candidates for this, since they are integrated directly with NsCDE and managed by it's configuration and autodetenction procedures.

If found, and if configured in `$FVWM_USERDIR/NsCDE.conf` with InfoStore variable `nscde_use_dunst` set to 1, NsCDE will make fresh copy of `$FVWM_USERDIR/Dunst.conf` if it doesn't already exist, and start *dunst(1)* notification daemon. This standalone notification daemon is highly configurable and is Window Manager and Desktop Environment agnostic. Usage of dunst(1) is highly recommended.

X Terminal program? **Urxvt**, **xterm**, **mate-terminal**, **terminus** ... user's choice as always. As a slight recommendation, **mate-terminal** from MATE DE can be set to look almost as Dtterm, but with richer menu and better UTF-8 handling, the bad thing is that configuration if not done via GUI or configuration file but is stored in binary DCONF registry, and registry editor like **dconf-editor** or or dconf **gsettings** must be used for non-interactive or CLI editing. See the example in `$NSCDE_ROOT/share/examples/colormgr.local.example` on how to integrate **mate-terminal** with a Color Style Manager. Second (if not first) best choice is **Urxvt**, but since it does not have a menu nor a real tabs, tmux(1), screen(1) or possibly tabbed(1) can be used for the same functionality. Suggestions for more dtterm-like alternative are welcome.

File manager? Since author does not use them very much, there is no some strong suggestion. Maybe **Krusader** from KDE is a best choice because it has a lot of features and functions plus two pane mode for work. It looks like a total contrast to GNOME way of doing things, so it must be good, although it is not at all similar to CDE's original dtfile(1), but dtfile(1) is a bad and poor file manager anyway. Another reasonable choice can be **pcmanfm** or **pcmanfm-qt**. For something more *original*, Xplore file manager is written with Motif widget. It looks nice, but it is unfinished (lacks real actions for many things, and instead input dialogs are popped up for copy/paste ...) it is not maintained and developed, and if someone

does not brings it up from the past it can serve only for overview of directories, simple actions and nice Motif decoration.

Editor? Gvim, Emacs, Xemacs, Nedit ... user's choice.

Another nice and useful app is **Gkrellm** for which NsCDE has a ready drop-in theme called (of course) NsCDE in `share/config_templates` and it can be put in user's `~/.gkrellm2/themes` and integrated with Color Style Manager with the `$NSCDE_DATADIR/config_templates/colormgr.local` which is by default installed as `$FVWM_USERDIR/libexec/colormgr.local`.

If standalone freedesktop autostarter **dex-autostart** (sometimes called "dex") is installed, it will be used by default `Init.fvwmconf` function `CommonInitFunction` in local mode: it will read and start ".desktop" files in the `$HOME/.config/autostart` directory.

X Compositor: if user likes visual effects with tinting, transparency, shadows, 3D, smooth changes and so on, *compton(1)*, *compton-ng(1)* or *picom(1)* standalone compositors are excellent programs and tools for such users, who want to combine retro and modern style. In the time of writing this, it looks like *picom(1)* is the most maintained of this three X compositors. Personally, I feel it like some kind of *lag*, no matter how powerful GPU, CPU and RAM I have. I turn it on occasionally, more as an amusement of xsnow, xsanta or xeyes type, but when I have serious work to do, I simply turn it off in some moment. Maybe it can be better if it is configured more conservative than example. See `/opt/NsCDE/share/examples/compton-integration` for a starting point. Syntax of this file is largely compatible for all three above mentioned X compositors.

Notice: when X Compositor is active on X display, FVWM geometry indicator which is shown at the center of the screen during window move or resize actions will not be visible. Also, consider uncommenting `NSCDE_REDRAW_WORKAROUND` and set it to 1 in your `$FVWM_USERDIR/NsCDe.conf`.

18. Single Logical Screen, Xinerama - multiscreen support

NsCDE has a basic support for the multiscreen setup which is basic as it is FVWM2 multiscreen support, with couple of menus added and functions dealing with move and resize operations which are aware of the multiple logical screens. Single logical screen is referred as "SLS" in FVWM and NsCDE documentation. Screens are implemented (and this cannot be changed in FVWM2) on the sub-page level. In other words, as workspaces (desks) contains pages, pages are split to two or more monitors inside one single page. This can be a bit confusing in the combination with edge scrolling and window positioning and it takes some practice to become comfortable with such third, non-trivial space on the screen which is already divided logically in two levels. Better solution here will be to use FVWM3 which is slowly becoming of non-beta quality at the time of writing this. FVWM3 has native XRandR support.

Monitor handling by the Xrandr X extension is out of scope for NsCDE under FVWM2 and is dealed with xrandr(1) command and other such tools. Nevertheless, when other monitor is added to the system in SLS mode, FVWM/NsCDE must be restarted (restart session simply) to recompute spaces, screen sizes and so on. After restart, two new menus are available: One on Root menu on which there are entries to move all windows on current screen, or to pick a window for moving to some of the (xrandr identified names) logical screens connected to the system. Second menu will appears on the "Window Options" menu called from the first titlebar button of the window or from the root version of the "Window Options" menu. This menu allows moving current window to other logical screen.

When logical monitor is disconnected from the X setup, FVWM NsCDE must be restarted again to get things right.

Front Panel will appear on the primary screen, but can be moved to other screens by Ctrl+Escape pressed while pointer is on the desired screen. This does not work always well when logical screens are of different resolution and it is specially visible when making third mouse click on the Workspace Manager buttons which can be popped down below the screen instead of up to be visible. On the monitors of the same resolution in SLS configuration, no such problem exists.

All other functions and window positioning managed directly by the NsCDE will handle windows and transient windows correctly, so no windows centered between two screens are expected, but some barely visible flickering and quick moving can be observed by some parts of the NsCDE in some cases. For example, PGM - page manager left down from the Workspace Manager on the Front Panel when clicked will popup "Go to Page ..." menu in a more free floating form, and not directly above PGM dynamic icon and such things ...

NsCDE under FVWM3 is a different story. FVWM3 implements couple different multimonitor modes of operation which are configured with DesktopConfiguration command in FVWM3 config. NsCDE under FVWM3 can manage and change this modes of operation from the Workspaces and Pages Style Manager. Currently, supported workspace modes of operation are: global, per-monitor and shared. See fvwm3(1) for explanation and detailed documentation how this multi-monitor models are behaving and how they are implemented. In non-global mode, Local Pager, Global Pager and GWM will be showing only workspaces and pages from the current monitor on which they are started (and possibly updated in the future versions of FVWM3).

19. Similarities and differences in usage and look between CDE and NsCDE

NsCDE is not a mere clone of CDE. Under the first visual impression, there are some unintentional and intentional differences.

First of all, it is not a standalone Window Manager or Desktop Environment written in some language(s), tightly integrated as uniform army of tools (like lxqt, xfce4 ...). It is a patchwork which owns 80% of it’s

functionality to wonderful and powerful Window Manager of FVWM. Other parts are standalone desktop tools and implementations of desktop concepts which are integrated, various configurations, scripts and programs which are making the whole thing to function like one entirety: the combination of the CDE experience and modern powerful X Window Manager ... a hybrid desktop environment. Here are some things that I can recall to be different - for the worse or for the better, user's opinion may vary.

What is similar or the same:

- There is a recognizable titlebar and buttons
- Titlebar buttons have the same basic (left click) actions as CDE
- Color themes and theming
- Front Panel and subpanels
- Workspace Manager
- Workspace Menu / Root Menu (right click on the root window)
- Workspaces (desks)
- Many of the icons are reused for XDG compatible icon theme
- Backdrops
- Style Manager launcher and most of the Style Managers
- Occupy Workspace/Page/Monitor dialog
- Workstation Info window
- FpLite (not with the same function)
- Front Panel clock, calendar and check mail
- Icon positioning
- Look and feel via FVWM Styles
- Nice vintage wait cursor in the sand clock shape
- Various misc small imitations ...

Differencies exist: for worse or better. They are described here in detail with complete explanations:

- Workspace Manager has a four default choices for workspaces (desks). As in CDE four is a default, but combinations with 2, 6 and 8 are possible. Workspace Manger can be of dynamic width when number of workspaces is changed, or it can be of fixed width if InfoStore variable `wsm.eco` is set to 1 in `$FVWM_USERDIR/NsCDE.conf`.
- No drag and drop. This is specially visible in *Install Icon* action which actually calls custom tool Subpanel Manager for this actions. Subpanel Manager itself will be rewritten in a nicer and less buggy way on the first good occasion.
- No Dt Actions builder, and never will be. Write FvwmScript scripts or use some toolkit in combination with python, perl ...

- No Application Manager. If integration with *Install Icon* and possibly menus will be possible with some file manager, it may be (re)invented in the future. The good candidates are pcmanfm and spacefm, but this has it's problems still.
- Keybindings are 90% custom made, and user have a choice to use it or - partially or totally rewrite it. There are more functions and actions in NsCDE than in CDE, and hence there are a lot of key bindings.
- Mouse bindings - some actions like Workspace Menu in CDE are mimicked in NsCDE, titlebar and titlebar buttons too, but since there is no much of them in original CDE anyway, there is a plenty of custom mouse bindings and mouse bindings in combination with modifier keys. As for keybindings apply: use it or write your own.
- Color Style Manager has numerous new functions: Gtk and Qt integration, X resources integration is optional, and it has even a possibility to run a custom script with required parameters of current palette and number of colors for external and marginal color scheme integrations - like **Gkrellm** for example.
- Font Style Manager is totally NsCDE oriented and doesn't work the same as font management in CDE. NsCDE supports XFT fonts (disable antialiasing if you want *extreme* original look) and it combines 5 groups of fonts in 3 sizes described in this documentation.
- Keyboard Style Manager implements all options supported by the xset(1) on PC. CDE original in default installation at least, seems to have only auto-repeat and click volume controls.
- Mouse Style Manager does not have configurable middle mouse (button 2) action since this is not applicable very much on today's GUI widgets.
- Beep Style Manager has a additional Beep button for testing during setup.
- Screensaver Style Manager is in fact Xscreensaver setup. Perfect drop in replacement and much fancier than original.
- Window Style Manager manages much more of window, icon, pages and animation behavior than original program in CDE, and even this is a small subset of options in FVWM. See it's documentation and fvwm(1) man page.
- Power Style Manager is actually very rare in Style Manager across old CDE setups. It manages DPMS setting of the monitor with xset(1).
- Workspaces and Pager Style Manager is NsCDE custom tool for managing Workspaces and Pages. It is not present in any version of the CDE.
- Startup Style Manager is available only if NsCDE is started under some X Session Manager. It detects supported DE's and starts appropriate settings tool for that desktop environment if it is found.
- Pages: not present in CDE in best of my knowledge. Only workspaces (desks) in original. Page Manager (PGM) is a custom FrontPanel icon which is using place left bottom of the Workspace Manager. It popups menu with the list of pages and can change current page.
- Custom keyboard and mouse actions on titlebars, buttons and root window.
- GeometryMgr - Manages custom X11 window starting size and position. NsCDE invention.
- FpLite is measuring system load, not desktop activity. It has much more fine grained indication of activity with colors, and it's height is 3x of the original for a better visibility. On click it is calling FVWM function which will run terminal program with top or similar program (`$[infostore.taskmgr]`), or anything else if user overrides that function in local configuration (NsCDE.conf) with InfoStore variable.

- Calendar and Mail widgets are placeholders and simple indicators which are expected to be extended with already named functions to do what user wants. Mail widget will call either default user's mailer program, or it will try to guess from the names of some popular programs if they are installed on the system.
- Probably there are more small differences.

20. Patches for FVWM

Optional but recommended patches for FVWM from 2.6.7 to 2.6.9 are in `patches/fvwm2` top directory of the NsCDE tarball for installation.

This patches will add:

- corrections for cursor icon under buttons of the FvwmScript(1) it is really not a nice thing to have `XC_hand2` which is usually used for hyperlinks as a pointer icon when mouse is above buttons. Implemented as an option in FVWM3, not to disturb old default, no matter how bad is that default probably is.
- FvwmButtons(1) WindowName support - an native alternative to xdotool(1) workaround. It will set name and icon name of single subpanels, that is, every FvwmButtons object which has titlebar enabled with FVWM styles. Implemented by default in FVWM3.
- FvwmButtons triangle-in (sunken) support. Provides a 3rd argument for `indicator` parameter of the FvwmButtons(1) button. It can be "in" (default for NsCDE in `fvwm/FrontPanel.fvwmconf)` or "out" to confirm the FVWM default. If omitted, "out" is default, since it was that way before this patch. Implemented by default in FVWM3.

In order to have patched FVWM2, apply this patch or patches against FVWM 2.6.7, 2.6.8 or 2.6.9 source and (re)compile FVWM2. You can even make your own RPM DEB, Arch, BSD, SunOS or similar package from that and install it.

21. FVWM3 Support

NsCDE from version 1.3 has late beta quality of support for FVWM3. MsCDE is and will stay compatible for FVWM2 as long as possible. At the time of this writing, FVWM3 was in it's fourth release and much of the previous problems are solved, altrough it still has some problems with FvwmPager in multi-monitor setup. Nevertheless, NsCDE will search for "fvwm3" and then "fvwm2" and "fvwm" in `$PATH`. If both, FVWM3 and FVWM2 are installed on the same system, from NsCDE 2.0 and up, FVWM3 has higher priority. This can be overriden in user's environment (for example `$HOME/.bashrc`) by setting environment variable `FVWM_BIN` with the value which points to name of the FVWM binary, or full path to the binary if for example, FVWM3 was installed from source in some non-standard place out of the path.

For example:

```
export FVWM_BIN=/opt/fvwm3/bin/fvwm3
```

or ...

```
export FVWM_BIN=fvwm3
```

or ...

```
export FVWM_BIN=fvwm
```

When running with FVWM3, NsCDE will behave almost identical as in FVWM2, with benefit of RandR support in multiple monitors setup. This allows dynamic addition and removal of physical monitors and better management of such configurations.

Probably the most notable feature of FVWM3 is `DesktopConfiguration`. FVWM3 configuration parameter `DesktopConfiguration` decides of workspaces layout in case of multiple monitors. For now, there are three options: "global" (default), "per-monitor" and "shared". First option is very similar in layout to old Xinerama support in FVWM2, while second and third model are splitting workspaces, so different parts of different workspaces can be shown on different monitors. NsCDE implements dynamic finding and displaying, as well as creation of joint backdrops combinations for multiple monitors layout out of the box, so old hack described in `$NSCDE_ROOT/share/doc/nscde/examples/fvwm3-per-monitor/README` is not needed anymore. As in the future more desktop layouts are planned by FVWM3 developers, this setting has it's GUI control usable on FVWM3 in the Workspaces and Pages Style Manager. NsCDE provides `InfoStoreAdd desktopconfiguration` directive in the `$FVWM_USERDIR/NsCDE.conf` when FVWM3 is used. This value can be edited manually to contain "global", "per-monitor" or "shared" value, or Workspaces and Pages Manager can be used for that task.

Third difference between NsCDE under FVWM2 and FVWM3 is logging. While FVWM2 is logging all it's actions on X server's standard output and standard error, which usually ended up in `$HOME/.xsession-errors`, FVWM3 logs into default or configured log file. In NsCDE this file is `$FVWM_USERDIR/tmp/fvwm.log`. By default, FVWM3 does not log anything there if not invoked with "-v" option, but logging can be toggled by sending SIGUSR2 to FVWM3 process. When Watch Errors menu item is called, it has one new option on personalized window menu which is called from the first (left) titlebar button: Togle FVWM3 Logging. When opening this log window, FVWM3 logging will be enabled almost immediately. To have logging enabled as soon as possible when FVWM3 is started or restarted, infostore variable `$[infostore.fvwm3_default_logging]` should be set in `$FVWM_USERDIR/NsCDE.conf`.

Caution: using NsCDE under FVWM3 is still a bit of beta quality. Example of the current unsolved FVWM3 bug: Local Pager shown when changing pages and workspaces will contain primary monitor content, which will be wrong on the secondary monitor(s). Any misbehaviour should be carefully distinguished if it is NsCDE or FVWM3 bug while deciding where to report bugs. Watching logs, trying

to trigger the same error under FVWM2 will be a good starting point for diagnostic of such problems which may manifest itself.

22. Credits

Apart from FVWM, GTK integration framework was forked from one advanced theme, clock is old standalone widget which I have found in the old X11 software archives while searching for something which can act as a Front Panel clock. Pclock fits here perfectly. Xscreensaver seems to me as a logical choice for screensaver facility.

- For forked CDEtheme: Jos van Riswick
- For pclock on a Front Panel: Alexander Kourakos
- For using Xscreensaver: Jamie Zawinski

23. Missing parts and existing problems

- Application Manager: Maybe with the help of some extensible (but sane and standalone, with normal titlebar) file manager, but question remains how to send enumerated apps from such a file manager view to Front Panel subpanels or as submenu. Probably external drag and drop applet which can be swallowed in subpanels to accept drop with middle mouse move and edit subpanel configuration? This will than replace Subpanels Manager app, but it must also have functions for editing, deleting etc ...
- Action builder (dtaction) - not likely ever. Use FvwmScript or maybe some Python gui bindings.
- Session Management (dtsession) - NsCDE can use the external custom Session Managers from various DE's. See examples in `$NSCDE_ROOT/share/doc/nscde/examples`. Since there are similar programs in existence, plus FVWM's own functions for automatic start of programs, NsCDE is more or less covered here.