Not so Common Desktop Environment (NsCDE) Manual

M. Z.

This manual describes version 0.9.b70 of NsCDE.

This manual describes NsCDE: Not so Common Desktop Environment

Table of Contents

Introduction

What is NsCDE?

In a nutshell, NsCDE is the CDE clone. Tehnically, it can be considered a heavy 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 as a window manager wrapper for session handling and additional DE functionality.

NsCDE's main goal is to revive 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 slightly polished interface (XFT, unicode, dynamic changes, rich key and mouse bindings, desk 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.

Other main components are GTK2, GTK3, Qt4 and Qt5 theme for unifying look and feel for the most Unix/Linux applications, custom scripts which are helpers and backend workers for GUI parts and some data from the original CDE, as icons, palettes, and backdrops.

Why NsCDE?

Since the 90-ties, I have always liked this environment and it's somewhat crude socrealistic look in a contrast to "modern" Windows and GNOME approach which is going in the opposite taste 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 modern CDE.

NsCDE is intended for a people which doesn't like "modern" hypes, interfaces that try to mimic Mac and Windows and reimplementing their ideas for non-technical user's desktops, and reimplementing them 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 XFT font rendering, no immediate application dynamic changes. 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 "driver" 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 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.

Components of the NsCDE

Components overview

NsCDE is a wrapper and a bunch of configurations, scripts and apps around FVWM. FVWM is in my opinion a 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 users in the last decade from the mainstream desktop players.

NsCDE is by default rooted in `/opt/NsCDE` ($NSCDE_ROOT), but it can be relocated with only one variable changed in main wrapper `bin/nscde` and `NsCDE-Main.conf`.

It is not using your existing `$HOME/.fvwm` but sets `$FVWM_USERDIR` to `$HOME/.NsCDE`, and uses `/opt/NsCDE/config` as a sources of configuration.

Configuration model is a bit complex, but very flexible: configuration options are grouped in logical order. Configuration files are names `NsCDE-<group>.conf`. For example, `NsCDE-Functions.conf` for FVWM functions. Each configuration file can have two exclusive sources, and one additional. For example, if user doesn't have `$FVWM_USERDIR/NsCDE-Functions.conf`, then `$NSCDE_ROOT/config/NsCDE-Functions.conf` is read as default. Additionally, if `$FVWM_USERDIR/NsCDE-Functions.local` exists, it will be read in addition to conf file, from wherever it was read. This is intended as a primary mechanism for customization: If user doesn't need to override and change a lot of "system" configuration, but just add it's own in addition to existing, local file is place for such customization (of course, most parts of the existing FVWM configuration can be overridden or destroyed and recreated even in local files.

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 by me, 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
- Page Manager (PGM) - 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 (OccupyWorkspace) - 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 Manager (WsPgMgr) - FvwmScript

Helper Dialogs:

- ActionForm - FvwmScript
- ChoiceForm - FvwmScript
- FilePicker - FvwmScript
- InputForm - FvwmScript
- WaitNotice - 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.

Applets Docks and Panels

Front Panel

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

- Icons are static in configuration. FvwmButtons doesn't implement drag and drop protocol. It can be changed and customized in a two ways: first one, 8 of 10 icons (minus swallowed applets clock, calendar and mail that is) can be dinamically changed by writing button actions and icon paths in `FrontPanel.actions` in user's `$FVWM_USERDIR` and this will be applied *after* reading static configuration and running Front Panel for a piece of a second. The other way (full control) is by copying configuration file from `$NSCDE_ROOT/config` to `$FVWM_USERDIR`.
- In the original CDE, Front Panel is part of the dtwm Window Manager binary, while here 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 all 3 main mouse buttons different action can be assigned. This is used for example 8th icon where mouse button 1 calls editor, while mouse button 3 is calling file manager (if defined)
- In addition to iconification, Front Panel can be shaded to the bottom edge of the screen with Shift-Esc action, and put in a place with the same shortcut again.
- Iconification is by default to bottom right screen edge, while all other programs are by default iconified in the top left edge as in CDE.
- It is flexible, can be overlapped with programs, moves away for fully maximized windows and while pretty much thick, not in the way while not needed.
- - It has it's own menu on top left button and special context menu if this button is clicked with right mouse button. Middle mouse button behaves as if title bar 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
  - Restart Panel Clock (pclock)
  - Restart Panel Mail Applet
  - Restart Panel Date (MonthDayApplet)
  - Restart Panel Lite (FpLite)
  - Restart the whole Front Panel
- As FvwmButtons based dock, it swallows the following applets:
  - pclock (external standalone app with CDEish skin)
  - MonthDayApplet (FvwmScript)
  - CheckMailApplet (FvwmScript)
  - PGM (FvwmScript)
  - WSM / Workspace Manager (FvwmScript)
  - FpLite (FvwmScript)
- Third icon expects `$[infostore.filemgr]` to be defined.
- Fourth icon will call `$[infostore.terminal]` which must be defined or it is discovered.
- Sixth or Print icon is not really usable. It should call some predefined printer application in the future or to be replaced with something more useful. Good place for personal custom icon and redefinition with `FrontPanel.actions`.
- Seventh: Style Manager - clone of the well known CDE Launcher of "Style" tools.
- `$[infostore.xeditor]` - if defined in `$FVWM_USERDIR/NsCDE.conf`.
- 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 and 6 are empty, but they can be activated with middle pointer click on an empty place. Dialog will ask user if he wants it enabled. This is specially useful for subpanel 5 which will show Thunderbird if activated and if Thunderbird is installed.
  - Subpanel 1: Applications
  - Subpanel 3: Libre Office Components and various office/productivity tools
  - Subpanel 4: System Tools

  - Subpanel 7: NsCDE Style Manager and various Qt, Gtk and misc management applications
  - Subpanel 8: Tools
  - 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 (PGM), Right are Front Panel Lite (system load indicator), and **Exit** button (SysActionDialog).

  In the middle of this area there is WSM - Work Space Manager with well known four buttons for four virtual desks. By default, four desks are shown and configured, but this can be changed (see the Section called *Workspace Manager*).

Subpanels

NsCDE subpanels are simple transient FvwmButton docks. As Front Panel, 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-click root menu. They are meant for favorite, important and often used apps.

There is one main difference between CDE and NsCDE subpanels: *Install Icon* action calls special NsCDE tool for actions defining. There is no drag and drop from the app manager (which also doesn't exist).

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 really 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/NsCDE-Subpanels.conf`, 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 `$FVWM_USERDIR`, from where all subpanels changed from default will be read by the **$NSCDE_ROOT/libexec/generate_subpanels**, while the rest will be generated from the `$NSCDE_ROOT/config/Subpanels.actions`. This file can also be edited by hand (ok, not by hand, but editor will suffice) and the result may be applied by calling **generate_subpanels** and then `f_ReadCfg Subpanels`. Generated file is called `NsCDE-Subpanels.conf` and it is expected in the `$FVWM_USERDIR`. If not found there, `$NSCDE_ROOT/config/NsCDE-Subpanels.conf` will be read instead. Syntax of the Subpanels.actions is explained in the Configuration files explained section.

Presently, there is one workaround here: 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 `NsCDE-FrontPanel.conf`, and are internally mandatory names of their Subpanels. Since there is no configuration option for subpanel to set Window Title, we are using here 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 which introduces a *WindowName* option to it's configuration. Of course, patch will be proposed to the upstream FVWM.

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.

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 desks. There can be no odd number of desks, and combinations are: 2, 4, 6 and 8. Default is of course 4.
- Buttons are not fixed in width size as in CDE. They are not extending a width of the Front Panel, rather they are more wide in 2-buttons combination, and narrow in 6 and 8 button combination.
- As in CDE, desk names can be renamed. There is a keyboard combination in FvwmScript WSM which enters rename mode: Ctrl+Space on the current active desk, while pointer is inside WSM. Ctrl+Enter saves new name. Names are synchronized with FVWM desktop names and used in the rest of the configuration. Names are saved in `$FVWM_USERDIR/WSM.conf`
- Addition: right mouse button on desk button brings contextual menu. From this menu, global (all desks and their pages), or local (pages of the one desk) FvwmPager - visual pager can be called. Below this menu options, there is an option to bring up Window List, and small submenu for changing the active page on the current desk.
- If `$FVWM_USERDIR/WSM.conf` value *WSPPG* is 1, then in 2 or 4 desks/buttons mode a small current page indicator is show on the right side of the button. In addition to Page Manager (PGM) southwest of the WSM. Disabled (0) by default. `WSM.conf` (system wide: `$NSCDE_ROOT/config/WSM.conf`) is a separate configuration file read by WSM, PGM, WsPgMgr and such. Not a FVWM configuration file.
- Number of Desks and Pages can be configured with Workspace and Page Manager tool which is called with right pointer click on Front Panel's left button and selected from the menu.

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 **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.

Desks in NsCDE are named and numbered from 1, while in FVWM, they start from 0. This fact required additional effort while coding FvwmScripts and making default and core configuration.

Page Manager

Page Manager is entirely new thing. There is no concept of *Pages* in the original CDE, just desks (workspaces). This nice feature of pages is too useful to be disabled and sacrificed just to get even more CDE similarity, but really zealous user can configure NsCDE not to include pages, just desks. PGM is a small FvwmScript applet southwest from the WSM, represented by the silver-gray icon of the desk divided on *pagesX* x *pagesY*. Default is 2x2, that is, four pages per every desk, which in default first run gives 4 x 4 = 16 screens for applications. Minimum for pages is 0, and maximum 16. For example, in maximal desks+pages configuration, one gets 8x16 desktop, that is 128 screens! While undoubtedly this is diversion from a 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 ...

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

When clicked, PGM transforms in a closed popup menu with names of the pages and special "..." option on the top which calls local visual FvwmPager. When option from the menu is selected, menu again transforms to the Icon. If called, but not clicked, menu automatically transforms back into Icon after 20 seconds of inactivity. Middle pointer button calls Workspace and Page Manager, while the right pointer button calls visual Local Pager.

MonthDayApplet

Usual 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/NsCDE-Functions.local` to call a program which user wishes.

Front Panel Clock - pclock

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 `$NSCDE_ROOT/share/icons/CDE` or even replaced with a Solaris version with picture of the globe with red hands for hours and minutes and white for seconds. If clicked, it will try to execute firefox by default (which appears doesn't work if FVWM is started under some desktop environment like MATE). Pclock C source is provided for user's convenience if it needs to be recompiled on another system or architecture. Source is in `$NSCDE_ROOT/src/pclock-0.13.1`. With NsCDE binary for Linux is provided by default in `$NSCDE_ROOT/bin`.

Check Mail Applet

Fifth icon from the left on Front Panel is FvwmScript applet. It is calling `f_CheckMail` FVWM function every minute. If clicked, function is called immediately. By default `f_CheckMail` is an empty function. Up to user is to redefine it in his `$FVWM_USERDIR/NsCDE-Functions.local` to suit the needs for mail checking. To be clear, by default, it is not functional as an applet. Above this applet, there is an empty space for subpanel launcher which can be activated with middle click, and it will present Thunderbird entry if thunderbird 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.

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), I find it better suited to show system load. 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 safe defaults (*[default terminal app]* **-e top**). FpLite FvwmScript app uses little portable python script *getla1.py* from the `$NSCDE_ROOT/libexec` to obtain 1-minute load data.

GUI Tools

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
- Power Style Manager
- Startup Style Manager

If NsCDE was started under X Session Manager, Startup Style Manager icon will call setup tool for that session manager or DE. 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 the Section called *NsCDE Startup* about running NsCDE under X Session Manager for more information about this matter.

Key Bindings:

- Ctrl+Q: Quits Style Manager.

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 `$NSCDE_ROOT/share/backdrops` and `$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 `$FVWM_USERDIR/NsCDE-Backdrops.conf` which will be written by Backdrop Style Manager (or by hand). Backdrops are generated in the colors of the current color theme from the active palette (*Broica* by default). It has different colors for a group of every four desktops in 8-colors mode and the same base color in 4-color mode. Generated backdrop in the `$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 desk in 8-colors mode.

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 desk 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 of course.
- 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") will be loaded from `$NSCDE_ROOT/share/photos` and from the `$FVWM_USERDIR/photos` (if any). Options to use one photo for all desks exists. 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.

Key Bindings:

- Ctrl+Q: Quits Backdrop Style Manager.
- 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.

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.

Key Bindings:

- Escape: Quits Beep Style Manager.

- Ctrl+Q: Quits Beep Style Manager.

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 `$NSCDE_ROOT/share/palettes` and `$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 *Solaris* (called *Default* on SunOS) in 4 colors mode.

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 desk 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.

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 `$FVWM_USERDIR/Xdefaults` (default)
- GTK2
- GTK3
- Qt4
- Qt5
- User's `$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 the Section called *Credits*). 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.

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.

Exec Dialog

By default key binding for Exec Dialog is Meta (mod4) + F2, and located on main root menu before terminal i a "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 desk 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

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 `$NSCDE_ROOT/config/NsCDE-Font-$DPI.conf` and/or in `$FVWM_USERDIR/NsCDE-Font-$DPI.conf`. 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 must be selected 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 `$NSCDE_ROOT/share/fontsets` and in `$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 before saving as `$FVWM_USERDIR/NsCDE-Font-$DPI.conf`. Fontsets are on the leftmost GUI list and inactive until button *Use Selected Font Set* is not clicked. Manual font selection contains list of XFT fonts found on the system in the middle list and their styles (regular, bold, italic ...) on the rightmost list.

Fonts can be set for the current DPI, or custom predefined DPI can be set. If custom DPI is set, fonts configuration file will change in `$DPI` part of the name.

Main font selectors are:

- Preview Set For Size
- Font Group
- Font Size

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, mono bold), and third popup menu sets font size. When Font Style Manager is started, current configuration is loaded and fonts selected for a preview.

Button **Default** loads `$NSCDE_ROOT/share/fontsets/Default_$DPI.fontset` which can then be saved or further customized. 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 variable normal medium font and it's size to their configuration files. Qt4 and Qt5 should automatically pick Gtk fonts if 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 exists, with arguments of the new config file and current DPI. This is intended for user's customizations which are currently beyond NsCDE's scope of integrations.

Save button will save configuration of fonts and ask user to immediately restart FVWM, while **Save Font Set** will first save configuration as a new fontset and then as active configuration. In that way, further experiments with fonts can be done by the user, but saved fontset from some good point can be loaded to replace current configuration. When **Use Selected Font Set** is clicked, list of the fontsets becomes active, and lists of fonts and their styles inactive. Once font set is clicked, it will load preview of all the fonts in the fontset and the same button which is now called **Select Fonts Manually** can be clicked for further customizations. Mentioned button will change back it's title to **Use Selected Font Set**. Button **Close** discards actions, and **Help** button will call help text (if implemented).

Configuration of fonts can be totally reset and discarded by removing file `$FVWM_USERDIR/NsCDE-Font-$DPI.conf` and FVWM restarted.

Key Bindings:

- Ctrl+Q: Quits Font Style Manager.

Keyboard Style Manager

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

- Auto Repeat on/off
- Start Delay (start of repeat delay)
- Repeat Delay
- Click Volume

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.

Occupy Workspace

This tool dialog is called from the left titlebar button menu. Or from Alt+Space key combination from the window context. It is exact copy of the same CDE dialog and it sends selected window to workspace selected from the list, or it can make it sticky with **All Workspaces** checkbox pressed in. One addition here is the checkbox **Go With the Window**; when checked, makes NsCDE to change a current workspace and go where window was sent. **OK** performs an action, **Dismiss** quits Occupy Workspace tool.

Key Bindings:

- Escape: Quits Occupy Workspace without performing an action.
- H: Displays this help text, like **Help** was pressed.
- A: Checks *All Workspaces* checkbox.
- Return: Performs move action like OK button was pressed, without going to the selected workspace.
- Ctrl+Return: Performs move action like OK button was pressed, and changes active workspace to the same destination.
- Up/Down: Selects workspace in the up or down direction on the list.

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.

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.

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 (patched version of fvwm-menu-desktop is used in the background), 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. 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_ROOT/libexec/generate_subpanels** must be used to generate FVWM configuration output which must be redirected into `$FVWM_USERDIR/NsCDE-Subpanels.conf`.

Key Bindings:

- Escape: Quits Subpanel Manager.
- Ctrl+Q: Quits Subpanel Manager.

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.

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**, **pm-suspend**, **pm-hybrid-suspend** and **pm-hibernate** will work only if **sudo** entries are configured properly. 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. In `$NSCDE_ROOT/share/doc/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.
- 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.

Sysinfo

Copy of the Sun Solaris *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 *System* 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.

Window Style Manager

In the beginning, this GUI tool has been exact copy of the CDE Window Style Manager. In a way, this was bad. While providing exact copy of the same named CDE tool. It was ignoring the fact we are using FVWM as a WM and framework for NsCDE, and FVWM has couple hundred more options, and it is tens times more powerful than CDE's dtwm, and probably the main reason for combining CDE look and (partially) feel of CDE with powerful new functionalities which are provided by FVWM and some 3rd party programs.

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 from CDE original 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.

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.

Configuration Section: Window Icons

- *Use Icon Box*: if this option is selected, main FVWM configuration in `NsCDE-Main.conf` 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).

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 desk - this is controlled by the infostore variable 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.

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.

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.

Workspaces and Pages Manager (WsPgMgr)

This tool is specific to NsCDE. It configures three things:

- Number of virtual desktops
- Names of virtual desktops
- Number of virtual pages

This graphical tool writes files `$FVWM_USERDIR/WSM.conf` and `$FVWM_USERDIR/NsCDE.conf`. On the top of the window is graphical representation of the Workspace Manager. Every button when clicked becomes editable 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 desk will be changed across all NsCDE and FVWM (WSM, Occupy Workspace, menus etc) after window manager restart is triggered with **OK** button. In NsCDE, there are 4 options for desks: 2, 4, 6 and 8 desks. FVWM itself supports infinite number, but in NsCDE care must be taken about presentation of this desks in various applets and tools. Nevertheless, theoretically even in NsCDE with maximum number of desks multiplied with maximum number of pages, user can get 128 work spaces which is probably enough (too much) for 99% of the people.

Number of vertical and horizontal pages can also be configured here. This change will affect Page Manager (PGM), and menus which are displaying pages. Page names are not configurable via GUI. User is encouraged to open `$FVWM_USERDIR/WSM.conf` and edit to suit.

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 Manager is called from the Front Panel's small left button menu which is called when that button is clicked with the right pointer button.

Key Bindings:

- Escape: Quits Workspaces and Pages Manager.
- Ctrl+Q: Quits Workspaces and Pages Manager.

Helper Dialogs

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.

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.

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.

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.

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`.

NColorsDialog - FvwmScript

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.

Key Bindings:

- Escape: Quits Dialog.

PaletteDialog - FvwmScript

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.

Backdrops, Palettes and Fonts

Together with Workspace Manager, Backdrops and Palettes are probably most known things in CDE by which it is visually recognized.

Backdrops, as many probably know 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 desk 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 desk from 1-4 has it's own color variant from the current palette. If there are more that four desks defined, fifth 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. See existing backdrops to get an idea. 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.

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

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

Until then, file `$NSCDE_ROOT/config/NsCDE-Backdrops.conf` is read, which itself reads pre-generated and pre-defined backdrops from the `$NSCDE_ROOT/share/defaults/backer` directory.

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

Configuration files explained

As pointed above, NsCDE has a quite complex set of configuration files. ~ 90% of them are the FVWM configurations. But, this system of configurations is arranged in some logical and consistent way. For example, keyboard shortcuts in NsCDE-Keybindings.conf, FvwmBacker configuration in NsCDE-Backer.conf, (generated) colorsets in NsCDE-Colorset.conf etc.

All this configurations are included from the NsCDE-Main.conf. 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_ROOT/config, while user local hooks or user complete overrides are in $FVWM_USERDIR.

This is default list of system-wide configurations:

FrontPanel.actions

This is not a FVWM file. Lines in this file are default actions and icons for Front Panel. This file is parsed from the **fpexec** and **fpseticon** shell scripts. 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 main 10 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

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 managing this file yet.

NsCDE-Animate.conf

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/NsCDE-Animate.conf` (or `.local`) with editor or with Window Style Manager. None effect is choosen as default for increased CDE similarity, because CDE doesn't have iconification animation effects.

NsCDE-Auto.conf

Configuration of the *FvwmAuto* module and provided is a private set of FVWM functions from which only `f_AutoHide` (and parent functions) are used currently. FvwmAuto is started by NsCDE by default i `-menter` mode from `NsCDE-Main.conf`. Currently used to autoraise and autohide Front Panel, no matter which focus style is selected by default. In that way, Front Panel is not in the way of applications, yet on mousover action it is raised and lowered when it is not in focus and not use. FvwmAuto can be used to autoshade or autoiconify Front Panel and much more. See FvwmAuto(1).

NsCDE-Backdrops.conf

This file defines 8 colorsets for all (maximal) 8 desktops as a *TiledPixmap* colorset type. In the system configuration, static non-generated configuration 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).

NsCDE-Backer.conf

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.

NsCDE-Banner.conf

FvwmBanner(1) configuration. Displays logo during login.

NsCDE-Colorset.conf

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.

NsCDE.conf

This file defines various FVWM and NsCDE defaults. 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 the Section called *Window Style Manager* 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.

NsCDE-Event.conf

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. Currently, actions `new_desk`, `new_page`, `add_window` and `focus_change` are observed and their respective functions from `NsCDE-Functions.conf` are triggered. 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, 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.

NsCDE-Font-$DPIdpi.conf

... where $DPI is either 120, 144, 75 or 96.

This files are static when system wide, but generated by Font Style Manager in `$FVWM_USERDIR`. Every X server's *DPI state* selects and reads one of this files - the one which exactly or approximately matches current DPI in this order:

- if DPI is <= 85, $DPI is 75
- if DPI is > 85 and < 108, $DPI is 96
- if DPI is >= 108 and < 132, $DPI is 120
- if DPI is >= 132, $DPI is 144

Font sizes in configs are defined as infostore variables and used across FVWM config files, an 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.

NsCDE-Form.conf

Definitions of few FVWM forms. FvwmForm is reading this. This are simple pop-ups and dialogs used in FvwmScripts and in part of the configuration.

NsCDE-FrontPanel.conf

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 so wanted 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.

NsCDE-Functions.conf

Another important part of the configuration. Almost all FVWM functions are defined here, except 2-3 of core functions in NsCDE-Main.conf 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

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/NsCDE-Functions.local` - this extension file will be read by the main configuration immediately after processing `NSCDE-Functions.conf`. 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 desk changes
- `f_UserChangePage`: called when current active page changes

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 desk or page, window will be deiconified, and desk and/or current page changed to one where existing window is residing.

Care must be taken if this file is overridden by the local copy of the *conf* (not local) file, because a lot of things depends on this functions.

NsCDE-IconMan.conf

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 `NsCDE-Main.conf`. This file, `NsCDE-IconMan.conf` contains default configuration of that module.

NsCDE-Ident.conf

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 window.

NsCDE-Init.conf

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 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.

NsCDE-Keybindings.conf

For CDE clone and CDE compatibility, key bindings are a bit problematic area. In my local installation of open sourced CDE in virtual machine with CentOS 7, (didn't bothered to install Solaris 10 to examine their flavor again), key bindings are almost non-existent. Alt+F4 usually closes the window and that's it. Because of this fact, and because NsCDE is not only a visual clone of the CDE, but it aims to provide more functionality and to be useful *daily driver* for user to enjoy in familiar look and feel (hated by a lot of people) and to be seriously productive in the same time, comfortable and quick, to have good graphic (such as anti-aliased fonts, but this also can be turned off), a set of useful key bindings is defined by default. This default keybindings set can be extended, partially overridden, or completely replaced (copy the `NsCDE-Keybinding.conf` in `$FVWM_USERDIR` and edit, or even write from scratch). Defaults are author's *daily driver*. 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.
- the same combination, but with meta (mod4) is moving viewport by 4% of the screen. (Ctrl moves 100%)
- the same combination, but with shift modifier moves mouse by 1% of the screen
- Ctrl+Meta+cursor keys are changing first 4 desks
- Meta+Shift+cursor keys are moving currently focused window between first four desks
- Menu (Compose) key if pressed twice in a time window of two seconds pops up root menu in any context
- Meta+Menu combination pops up window operations menu
- Ctrl+Menu moves Root Pager (if enabled) beneath the pointer or iconifies Local Pager if LocalPager already has the pointer - toggle action
- Ctrl+ISO_Level3_Shift (right alt) moves Root Pager (if enabled) beneath the pointer or iconifies Local Pager if LocalPager already has the pointer - a toggle action
- Shift+Menu pops up Window List of all windows on the current desk
- Space under icon frame pops up icon-specific contextual menu
- Key Alt+Space in the context of the window, frame corners, frame sides, title bar and icon (ovoids root window context!) calls Occupy Space dialog for window moving between the desks.

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.
- Key Meta+Space in the context of the window, frame corners, frame sides and a title bar pops up Window Options context menu
- Key Meta+Space in the context of the icon, pops up Icon Options context menu
- Key Alt+Insert will give a focus to the last opened window
- Key Meta+Insert will warp a pointer and give a focus to the last opened window
- Key Alt+BackSpace will give a focus to the previously focused window
- Key Meta+BackSpace will warp a pointer and give a focus to the previously focused window
- Alt+F1 regenerates and refreshes the window

Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.

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

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.

- Alt+F3 "shades" or rolls up the window to titlebar only view

  Dynamic: defined in `$NSCDE_ROOT/config/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_ROOT/config/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_ROOT/config/Keymenu.actions` for automatic description on menus.

- Alt+F6 raises or lowers the window

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.

- Alt+F7 maximizes window 100% (whole screen + decorations)

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.

- Alt+F8 maximizes window ~ 80% - avoids Front Panel

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.

- Alt+F9 is empty

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.

- Alt+F10 deletes a window (see fvwm(1))

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.

- Meta+F10 closes a window

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.

- Meta+Alt+F10 forcefully destroys a window

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.

- Alt+F11 is empty

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.

- Alt+F12 calls xrefresh

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.

- Shift+Menu Will open WindowList in the middle of the screen for a current workspace (desk) if pressed twice in a time frame of two seconds
- Shift+Tab Will open WindowList in the middle of the screen for all workspaces (desks) if pressed twice in a time frame of two seconds
- Shift+BackSpace If pressed twice in a time frame of two seconds will do the same as Shift+Tab
- Alt+Tab is cycling trough pages of the active desk from up to down and then right up to down
- Meta+Tab is cycling trough the all workspaces (desks)

- Meta+Alt+L activates screensaver, that is, locks the screen
- Meta+Alt+Insert takes a screenshot of the root window with 200 ms delay

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.
- Meta+F1 invokes default terminal app ($[infostore.terminal])
- Meta+F2 invokes Exec dialog
- Meta+F3 toggles visibility of the titlebar

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.
- Meta+F4 toggles window's sticky state

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.
- Meta+F6 saves geometry information for a current window in GeoDB.ini

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.
- Meta+F7 centers the window in the middle of the screen

  Dynamic: defined in `$NSCDE_ROOT/config/Keymenu.actions` for automatic description on menus.
- Meta+F8 reads `GeoDB.ini` and if it finds an entry for the current resolution and `$[w.class]`, it resizes and moves a window according to the specification in `$FVWM_USERDIR/GeoDB.ini`

  Dynamic: defined in `$NSCDE_ROOT/config/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*
- Ctrl+Escape will raise Front Panel and reposition it to it's default place on the screen

  Dynamic: defined in `$NSCDE_ROOT/config/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_ROOT/config/Keymenu.actions` for automatic description on menus.
- Alt+Escape outside of the Front Panel focus will give Front Panel a focus and warp pointer in the middle of the Front Panel. If focus plus pointer is already on the Front Panel previous window will be selected for the focus and pointer warped in the center of that window.
- Meta+Escape will call transient GlobalPager in any context. When clicked, this Pager will close itself.
- In the context of the Front Panel, Alt + 1-10 will pop up subpanels 1 to 10 and warp pointer to it
- In the context of the any subpanel, Alt + 1-10 will pop down this subpanel and warp pointer to Front Panel

Keymenu.actions

This file is not FVWM configuration file. It is written Subpanel Settings, or manually with editor, and read by the **$NSCDE_ROOT/libexec/keymenu** command which will generate infostore variables with descriptive keyboard shortcut names which are used in NsCDE menus from `NsCDE-Menus.conf` and

`NsCDE-Functions.conf`, 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 `NsCDE-Keybindings.conf` 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 `NsCDE-Main.conf`.

Every line, or dynamic keybinding can be overriden here if (re)defined in user's `$FVWM_USERDIR/Keymeu.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 pop-down menu.

File Keymenu.actions is processed by **$NSCDE_ROOT/libexec/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 `NsCDE-Keybindings.conf`, but in `Keymenu.actions`, see notices in section about NsCDE-Keybindings.conf.

NsCDE-Main.conf

Core configuration. This file is read the first upon starting FVWM Window Manager. In fact, FVWM is invoked with `-f /path/to/NsCDE-Main.conf` to read it instead of default FVWM system configuration or user's `~/.fvwm/config`. This invocation and configuration in `NsCDE-Main.conf` 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 probably never be overridden in `$FVWM_USERDIR`. It is the init(8) of the NsCDE system.

NsCDE-Menus.conf

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

- **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 python script **nscde-fvwm-menu-desktop** which is a modified version of the **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 (`NsCDE-Menus.local`) 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 types of screenshot of the window.
- **m_WindowOps**: main menu of *Window Operations* invoked with a mouse click on left titlebar button. It has extended (**More ...**) and reduced (**Less ...**) version.
- **m_WindowOpsMore**: changes **m_WindowOps** from default reduced, to extended view (**More ...** option on Window Operation menu)
- **m_WindowOpsRootWin**: Standalone root window version of the Window Operations menu. Invoked with a click on the root window.
- **m_WindowOpsXterm**: extensions for the *Window Operations* menu activated on (recognized) terminal application windows.

NsCDE-Mousebindings.conf

File `NsCDE-Mousebindings.conf` is done in similar manner as the `NsCDE-Keybindings.conf`. Can be overridden (.conf) or extended (.local) 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: Move to Workspace menu double click closes (Close) a window
- Titlebar 1st (left) button right click: Move to Page menu double click closes (Destroy) 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 82% double click (un)maximizes 100%, covering Front Panel and it's area
- Titlebar 3rd (rightmost) button middle click: (Un)Maximizes 82% and makes window sticky or unsticks it depending on window's initial state

- Titlebar 3rd (rightmost) button right click: (Un)Maximizes 82% 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 **Window Options** menu
- 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 desk pages by 10%
- Control+ middle mouse click will call extended version of the WindowList with additional window info such as page number and window geometry.
- Pointer actions 4 and 5 (mouse wheel) will scroll between current up and down desk pages by 1% if Shift is simultaneously pressed
- Pointer actions 4 and 5 (mouse wheel) will scroll between current up and down desk pages by 1% if Meta (mod4) is simultaneously pressed, but in opposite direction then scroll with shift
- 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

NsCDE-Pager-WspLocPager.conf

Configures FvwmPager(1) type which is called from the Page Manager (PGM) or 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.

NsCDE-Pager-GlobalPager.conf

Similar, but bigger FvwmPager(1), invoked as *Global Pager* from right-click popup menu on Workspace Manager buttons, or (by default) with Meta+Escape key-binding this pager is transient and will disappear when used with a pointer click.

NsCDE-Pager-LocalPager.conf

Infostore variable `pageraisefp` in `NsCDE.conf` is by default 0. If enabled, when active page changes, visual FvwmPager(1) will be shown in the center of the screen near the top of it. 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.

NsCDE-Script.conf

Default for all FvwmScript(1) Module based applications and widgets. Script Path, default font and colorset. All this values are defined outside of this file, so nothing really should be changed here.

WSM.conf

Non-FVWM config file. Here, names of the desks are provided and can be changed manually of with Workspace Manager. Names of the desk pages (for the menus) are defined here too, and manager manually only for now. Option `WSPPG` can have value 0 or 1. If 1, small dynamic icon similar to the Page Manager (PGM) will be shown right of the desk name on the Workspace Manager active desk button, but only in 2 and 4 button configuration. Off by default, edit manually.

NsCDE-Style.conf

Main decoration configuration. `Style` '*' is applied globally. This is the main source of CDE and Motif-like 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 `$NSCDE_ROOT/config` to `$FVWM_USERDIR` and edited to suit.

Options are grouped in 5 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)

This styles can be extended and/or overridden by the user's own `$FVWM_USERDIR/NsCDE-Style.local`.

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

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

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 **$NSCDE_ROOT/libexec/generate_subpanels** command which will generate `NsCDE-Subpanels.conf` file in user's `$FVWM_USERDIR`. 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 `NsCDE-Subpanels.conf`. 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.

NsCDE-Subpanels.conf

This file is generated by the **$NSCDE_ROOT/libexec/generate_subpanels**. It is static in system directory, but changable and easly generated in the `$FVWM_USERDIR` It contains FvwmButtons(8) definitions of all 10 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_ROOT/libexec/generate_subpanels** which will read user's or system `Subpanels.actions` for every subpanel and if it is defined in user's one, take this one while generating NsCDE-Subpanels.conf. In system default, subpanels 2, 5 and 6 are disabled by default, but can be activated with a very quick double middle pointer click on the empty launcher without arrow: a Subpanel Settings application will appear on the screen which has a checkbox *"This Subpanel is Enabled"* which will be checked out by default, and can be checked in, and **OK** will enable subpanel with initial system defaults for name, width and application entries.

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`.

System Tree Layout

Everything from NsCDE is for now located as one compact place for easier portability between Linux and Unix systems in `/opt/NsCDE`. Only symlink to main starting wrapper `/opt/NsCDE/bin/nscde` is feasible to be put into `/usr/bin` or `/usr/local/bin`, since `/opt/NsCDE/bin` doesn't really need tobe put in user's `$PATH`.

This are subdirectories of `/opt/NsCDE` with short description what is what, and what is where:

- `/opt/NsCDE`

  main top directory of NsCDE installation all further descriptions will be written as relative to this directory

- `bin`

  nscde start wrapper called from `.xsession` or integrated as .desktop file in `/usr/share/xsession` is located here, as well as some helper scripts of the NsCDE which are suitable for general use. Front Panel **pclock** is also here. Pclock is the only binary part of the NsCDE

- config

  Configuration directory. All .conf files described in documentation are here. They are read from `config/NsCDE-Main.conf` which is called from bin/nscde by fvwm binary with `-f` directly.

- `lib/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 `share/icons/NsCDE`.

- `lib/python`

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

- `libexec`

  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.

- `share/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.

- `share/cursors`

  Custom cursors which are missing on plain X server installations but can be found in CDE. Referenced in `config/NsCDE-Style.conf`.

- `share/defaults/backer`

  Default generated backdrops for first start (Broica, 8 colors) Referenced in system's `config/NsCDE-Backdrops.conf`.

- `share/defaults/pages`

  Default page names for every possible combination supported by NsCDE

- `share/doc`

  Documentation

- `share/doc/examples`

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

- `share/fontsets`

  Default font sets used by the Font Style Manager

- `share/icons/CDE`

  Original CDE icons

- `share/icons/NsCDE`

  Custom NsCDE icons of which many are part of FvwmScript programs and applets

- `share/palettes`

  CDE palettes plus a bunch of new custom palettes. Used by Color Style Manager and Backdrop Style Manager, as well as **libexec/themegen.py**, **libexec/backdropmgr**, **libexec/colormgr**, **libexec/nscde_palette_colorgen.p**y

- `share/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.

- `share/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.

- `share/templates/integration/gtk2_gtk3_qt`

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

  share/templates also contains `Xdefaults` and some include files for it, as well as configuration for **stalonetray** if it is detected 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 desk.

- `src`

  Here is the patch for fvwm 2.6.7 and 2.6.8 which adds additional small features to fvwm, so we can achieve even more similarity between NsCDE and CDE. See the Section called *Patches for FVWM* for the details.

- `src/pclock-0.13.1`

  Latest version of pclock. Provided as C source (plus FreeBSD 12 binary) for non-Linux systems and Linux distributions where default binary cannot work (no matter how small and modest dependencies it has)

User Tree Layout

User's configuration is located in `$HOME/.NSCDE` - this place is what is referred as `$FVWM_USERDIR` in this documentation. There was no need to redefine this variable, since it server well for NsCDE. If user has a plain FVWM configuration in `$HOME/.fvwm` it will not be touched and can co-exist with NsCDE in that way. Here is the simple layout of things in `$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 `NsCDE-Backdrops.conf` manually.
- `backer/` directory: Generated backdrops referenced by Colorsets 31-38 for FvwmBacker
- `fontsets/` directory: If created, user can put or generate with Font Style Manager own fontsets here.
- `icons/` directory: Populated by dynamic menu action **libexec/nscde-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 `$NSCDE_ROOT/share/icons/NsCDE` automatically.
- `libexec/` directory: If created, `colormgr.local` script can be written and put here, as well as `fontmgr.local` and other user's hooks.
- `palettes/` directory: User can put custom palette files here, and they can than be selected by the Color Style Manager and processed for a preview or applied as new theme.
- `templates/` directory: Here, local subdirectory of `app-defaults` with tmpl files can be optionally created. Also, it is a good choice for Gkrellm or other files processed by the `libexec/colormgr.local`
- `tmp/` directory: Place used by parts of the NsCDE and in particular NsCDE's FvwmScript programs for temporary generated files for previews, or as scratch and work directory. Tools are usually taking care to cleanup their garbage from tmp/ on exit.
- `NsCDE-XYZ.conf` files: Absolute overrides of `$NSCDE_ROOT/config/NsCDE-XYZ.conf` files. If in existence, they will be read instead of system defaults. *XYZ* is here placeholder/example for Style, Functions, Keybindings, Init, Menus etc ...
- `NsCDE-XYZ.local` files: Extensions, added values of `$NSCDE_ROOT/config/NsCDE-XYZ.conf` files. If in existence, they will be read in right after their .conf main configurations from system (or local) directory. This is preferred way to extend functionality or override something not big enough for a complete "fork" of the config file. Colorset, Backdrops, Animate, Font-$DPIdpi, Init, and Subpanel are exception of this, that is, it is preferred (if not only thing possible) to have it as *.conf* files only and not *.local* files.

  XYZ is here placeholder/example for Style, Functions, Keybindings, Init, Menus etc ...
- `NsCDE.conf`: managed by Window Style Manager, Workspaces and Pages Manager, Pointer Style Manager and users own editor manually. See the rest of the documentation.
- `WSM.conf`: read/written by Workspace and Page Manager, WSM and user's favorite editor. Not an FVWM config file. System default of this file is `$NSCDE_ROOT/config/WSM.conf`.
- `FrontPanel.actions`: user's overrides/addons for Front Panel icons and actions. Written by editor, that is, manually only.
- `GeoDB.ini`: part of the Geometry Manager functionality. Written and read by the **bin/confset.py** and **bin/confget.py** on **Save Geometry**" and **Reposition Window** from Window Operations menu. Windows-like *ini* files are WAY nicer than dconf and such binary registry-like facilities.
- `Xdefaults`: Read on startup by xrdb(1).
- `Xdefaults.local`, `Xdefaults.fontdefs`, `Xdefaults.mouse`: Included with preprocessor directives from `Xdefaults`
- `Xset.conf`: 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`: If exists, used only in bare sandbox mode, where basic functionality of the NsCDE is needed, and not full DE-like environment.

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:

- Korn Shell 93. All shell script routines inside configuration, helper scripts and FvwmScript helpers are written with ksh. It is not sure if pdksh can be drop in replacement, but 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* - only if FVWM is not patched with WindowName patch for the FvwmButtons
- *ImageMagick* - really needed.
- *Xscreensaver* - optional, but Screen Style Manager 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 possibly python-qt4, or python2-qt4. 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 for the first public release.

  On CentOS, Fedora, Ubuntu and probably Debian, also on FreeBSD, there should be no problem to install this with package manager, but in tests on Arch Linux, PyQt4 was obsoleted and user will have a hard time getting source from AUR and compiling huge bunch of Qt4 libraries. There is possibility to replace PyQt4 with *PyQt5* imports in code and it will work, no matter that it will segfault and core dump on the exit.
- *Gtk2, Gtk3, Qt4, Qt5, qtconfig-qt4, qt5ct, qt5-qtstyleplugins* (optional) There is a great chance this libraries and some usefull 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.

- 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)
- Open Motif 2 libraries for some programs, although it is not used in any part of the NsCDE. (optional)
- **Stalonetray** for "tray" facility (optional)
- **xterm**
- *python3* (FVWM)
- *python34-pyxdg* or **python3-pyxdg** or ... (FVWM)
- *libstroke* (FVWM)
- *perl-File-MimeInfo* (on some platforms and some fvwm packages)

Installation

When FVWM and all/most above mentioned dependencies are met, NsCDE can be used. Present installation is very simple:

```
$ su - || sudo -i
# umask 0022
# cd /tmp
# tar xpzf /whatever/you/put/NsCDE-<version>.tar.gz
# cp -rp NsCDE-<version>/NsCDE /opt
```

... and optionally:

```
# cd /usr/local/bin || cd /usr/bin
ln -s /opt/NsCDE/bin/nscde
```

That's it. NsCDE is installed.

NsCDE Startup

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

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

Initial Configuration

Upon the first (successful) start, `~/.NsCDE`, that is `$FVWM_USERDIR` is created, and only icons subdirectory is created as **nscde-fvwm-menu-desktop** is run. User will be presented with a default system setup and with default color theme *Broica* in 8

colors, and `f_FindTerm` function will try hard to find some usable terminal application and run it with setup. If Gkrellm, pnmixer and/or stalonetray programs are installed, on the system and found, they will be run too.

Initial setup is a simple script (**$NSCDE_ROOT/libexec/nscde_setup**) 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`
- New empty `~/.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`
- `~/.stalonetrayrc`

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 can be still written with Color Style Manager. After **nscde_setup** script finishes setup, Color Style Manager will be run and user asked to confirm default theme or change it. *Do not* avoid this step, since some program bits are not fully setup on 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. 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 after this setup to open `$FVWM_USERDIR/NsCDE.conf` and set up InfoStoreAdd internal FVWM variables for `terminal`, `filemgr` and `xeditor` to user's favorite programs for functions.

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

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

- Xdefaults.mouse

It is advised to logout and login from the X session after this, and check if everything looks ok. Also, it is a good idea to start using programs from the menu and examine environment around for a half an hour or so, before 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 usage.

Integration with X resources and widgets

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 places 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 `$NSCDE_ROOT/share/templates/app-defaults/` and put in `$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.

Gtk2, Gtk3, Qt4 and Qt5

$NSCDE_ROOT/libexec/themegen.py with `$NSCDE_ROOT/lib/python` and with `$NSCDE_ROOT/share/templates/integration/gtk2_gtk3_qt` are parts of the optional Gtk2, Gtk3, Qt4 and Qt5 integration suite. When run from the Color Style Manager or manually with the **$NSCDE_ROOT/libexec/themegen.py**, 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.

Custom application integration

If **$FVWM_USERDIR/libexec/colormgr.local** exists, Color Style Manager will run it if it's checkbox is selected. This script or program will be run with a full path of CDE palette file followed by the number of colors selected in 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 cus-

tomization of this resources. Also "skins" for programs like **smplayer, audacious** and **Gkrellm** can be processed from custom **colormgr.local**.

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

Additional recommended software

Since NsCDE is basically a collection of configurations, themes and tools around FVWM and not desktop environment in official definition, 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. If NsCDE example configuration for stalonetray is used, it will have 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. Ideas are welcome.

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/doc/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/doc/examples` and it can be put in user's `~/.gkrellm2/themes` and integrated with Color Style Manager with the `$NSCDE_ROOT/share/doc/examples/colormgr.local.example` which can be installed as `$FVWM_USERDIR/libexec/colormgr.local`.

X Compositor: if user likes visual effects with tinting, transparency, shadows, 3D, smooth changes and so on, *compton(1)* standalone compositor is an excellent program and tool for such users - who want to combine retro and modern style. 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/doc/examples/compton-integration` for a starting point.

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 unintentional and intentional differences.

First of all, it is not a standalone Window Manager or Desktop Environment written in some language(s). It is a patchwork which owns 80% of it's functionality to wonderful and powerful Window Manager of FVWM. Other parts are configurations, scripts and programs which are making the whole thing to function like the combination of the CDE experience and modern powerful X Window Manager. 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)
- Most of the icons reused
- Backdrops
- Style Manager launcher and most of the Style Managers
- Occupy Workspace dialog
- Workstation Info window (as found in Solaris CDE)
- FpLite (not with the same function)
- Front Panel clock, calendar and check mail
- Icon positioning
- Look and feel via FVWM Styles
- Nice vintage but somewhat irritating 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 fixed choices for workspaces (desks). As in CDE four is a default, but combinations with 2, 6 and 8 are possible. If reduced to 2 desks, WSM buttons will grow in width to fill fixed space of the WSM FvwmScript applet. If 6 or 8 desks are configured, buttons will shrink and will be narrower. For that reason, in 6 and 8 desks mode it is not possible to have page indicator (`WSM.conf`: WSPPG) on the right side of the desk buttons, but this doesn't exist in CDE anyway, since CDE doesn't have concept of pages (in my best knowledge). This means that Front Panel has always a fixed width which is ok for a distinctive look and for screens with smaller resolutions - it will not grow or shrink depending of number of desks and their buttons shown in WSM.

- 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.
- 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. Palette color editor is missing. It should be possible to write it in the FvwmScript in some future version update.
- Font Style Manager is totally NsCDE oriented and doesn't work much 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).
- 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 widget which is using place left bottom of the Workspace Manager. It can change current page or invoke ("...") graphical pager - FvwmPager(1) for a current workspace (desk).
- Custom keyboard and mouse actions on titlebars, buttons and root window.
- WsPgMgr - Manage Workspaces and Pages. 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, or anything else if use overrides that function in local configuration.
- Calendar and Mail widgets are placeholders and simple indicators which are expected to be extended with already named functions to do what user wants.
- Probably some more small differences.

Patches for FVWM

Optional but recommended patches for FVWM 2.6.7 and 2.6.8 are in `/opt/NsCDE/src` directory.

This patches will add:

- Three underlines Menu Style (used in **MenuFvwmRoot** for NsCDE)
- 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. Planned to be implemented as an option, not to disturb old default, no matter how bad is probably that default
- 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
- 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 `config/NsCDE-FrontPanel.conf)` or "out" to confirm the FVWM default. If omitted, "out" is default, since it was that way before this patch.

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

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

Missing parts and existing problems

- Application Manager: Maybe with the help of some extensible (but sane and standalone, with 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 `share/doc/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.

Ideas and Tasks for future improvements

- Some default editor to be used as a Application Manager
- Rewrite a bad Subpanel Manager and awkward subpanel handling
- Geometry manager: introduce step that will ask for Window to be saved in `GeoDB.ini` with Name, Icon Name, Class or Resource in order to be more flexible with applications which have multiple windows with the same class name.
- Palette color editor in Color Style Manager is missing. It should be possible to write it in the FvwmScript(1).
- Find or write the replacement for PyQt4/PyQt5 API in Gtk/Qt theme integration
- FvwmScript for setting default X terminal emulator, X editor, file manager?
- Modern (in a sense of specs and completness) icon theme with the rest of available CDE icons (as much as possible)? Easy, but a lot of long time boring work.
- Localization. It will be hard and a lot of work. Specially FvwmScript(1) elements which are dependent on text length. Localization for Script elements, menus and documentation is the scope.
- ...