launcher-menus

Release 1!1.0.0

Pradyumna Paranjape

May 07, 2021

CONTENTS:

CHAPTER

ONE

README

1.1 launcher_menus

1.1.1 Gist

Source Code Repository

 Repository

Badges

1.1.2 Description

Launcher menu wrapper.

Provides an API for launcher menus such as:

- dmenu
- bemenu

Can be extended to other menus: see section configuration

What does it do

- Runs a subprocess for the selected `<menu>` and returns its standard output or `None`

CHAPTER

TWO

INSTALLATION

2.1 Prerequisites

At least one of the following menu launchers need to be installed

- dmenu
- bemenu

Else, a *custom* menu may be used by supplying flags as described

2.2 Install

2.2.1 pip

Preferred method

Install

```
pip install launcher_menus
```

Update

```
pip install -U launcher_menus
```

Uninstall

```
pip uninstall -y launcher_menus
```

2.2.2 pspman

(Linux only)

For automated management: updates, etc

Install

```
pspman -s -i https://github.com/pradyparanjpe/launcher_menus.git
```

Update

```
pspman
```

That's all.

Uninstall

Remove installation:

```
pspman -s -d launcher_menus
```

CHAPTER

THREE

USER-CONFIGURATION

Configuration file is in yaml format.

<menu>.yml files bear flags corresponding to actions for <menu>, where <menu> may be dmenu, bemenu, etc

Any file named `_template.yml` is ignored.

3.1 Location of configuration files

3.1.1 Default:

`<installation path>/site-packages/launcher_menus/`menu-cfgs

Custom configuration may be specified at the following locations:

3.1.2 User (XDG_CONFIG_HOME):

This variable is generally set to `$HOME/.config` on unix-like systems. Even if unset, we will still try the `$HOME/.config` directory.

`$XFG_CONFIG_HOME/launcher_menus/<menu>.yml`

3.1.3 Local:

``.launcher_menus/<menu>.yml**

Note:

- Configuration is loaded in the same order as described above.

Warning:

- A later loaded configuration **SHALL** overwrite a previously loaded configuration if defined for the same <menu>.

3.2 Configuration format

Copy _template to menu-cfgs/<menu>.yml

Edit fields to provide flags:

- Example:

```
bottom: -b
prompt: --prompt
```

3.2.1 Example:

_template.yml

```
bool:
  bottom: null
  grab: null
  wrap: null
  ifne: null
  ignorecase: null
  nooverlap: null

input:
  version: null
  lines: null
  monitor: null
  height: null
  prompt: null
  prefix: null
  index: null
  scrollbar: null
  font: null
  title_background: null
  title_foreground: null
  normal_background: null
  normal_foreground: null
  filter_background: null
  filter_foreground: null
  high_background: null
  high_foreground: null
  scroll_background: null
  scroll_foreground: null
  selected_background: null
  selected_foreground: null
  windowid: null
```

CHAPTER

FOUR

USAGE

4.1 Instructions

4.1.1 Call <menu>

Call `menu` [`dmenu`, `bemenu`, `<others>`] from python script as a replacement for input popups.

Basic usage

- Import in script:

```
# import
from launcher_menus import menu

user_letter = menu(command='bemenu', opts=['a', 'b', 'c', 'd'])
if user_letter is not None:
    # user did not hit <Esc>
    print(user_letter)
else:
    print("Aborted...")
```

Results:

```
a
```

Fancy usage

- User-defined styles

```
# import
from launcher_menus import LauncherMenu

mask_color = "#000000"
password_menu = LauncherMenu(command='bemenu', filter_background=mask_color,
                             filter_foreground=mask_color)
password = password_menu()
if password is None:
    # user hit <Esc>
    print("Can't go ahead without password")
else:
    print(password)  # A bad idea
```

Results:

```
Can't go ahead without password
```

- Pre-defined themes

```
# import
from launcher_menus.themes import emergency_prompt, password_prompt
```

4.2 Recommendation

- Use user-defined configurations

CHAPTER

FIVE

SOURCE CODE DOC

5.1 Entry Point

5.1.1 Package import

launcher_menus

Python API for Launcher menus

class launcher_menus.**LauncherMenu**(*opts=None*, *command=None*, *flag_names=None*, *fail='warn'*, ***kwargs*)

Launcher Menu wrapper object with pre-defined menu options.

Parameters

- **opts** (Optional[List[str]]) – list: options to be offerred by menu.
- **command** (Optional[str]) – command to use {dmenu,bemenu,<custom>}
- **flag_names** (Union[PathLike, dict, None]) – dict providing action: flags or path to cognate yaml.
- **fail** (str) – 'warn': warn, 'fail': error, 'guess': try creating, else warn
- ****kwargs** – default values for kwargs of menu

opts
default options to be offerred

command
default menu command to run

flag_names
dictionary of {actions: flag_names}

fail
default failure behaviour

Raises

- **TypeError** –
- ***FlagNameNotFoundError*** –

menu(*opts=None*, *command=None*, *flag_names=None*, *fail='warn'*, ***kwargs*)

Call menu

Return type Optional[str]

exception launcher_menus.**MenuError**
<MENU> errors Base.

menu() call

launcher_menus.LauncherMenu.**__call__**(*self*, *opts=None*, *command=None*, *flag_names=None*, *fail='warn'*, ***kwargs*)
Call <command> menu to collect interactive information.

Parameters

- **opts** (Optional[List[str]]) – list: options to be offerred by menu.
- **command** (Optional[str]) – command to use {dmenu,bemenu,<custom>}
- **flag_names** (Union[PathLike, dict, None]) – dict providing action: flags or path to cognate yaml.
- **fail** (str) – 'warn': warn, 'fail': error, 'guess': try creating, else warn
- **kwargs** – flag to be called at command line:
  - bottom = bool: show bar at bottom
  - grab = bool: show menu before reading stdin (faster)
  - ignorecase = bool: match items ignoring case
  - wrap = bool: wrap cursor selection
  - ifne = bool: display only if opts
  - nooverlap = bool: do not overlap panels
  - lines = int: list opts on vertical 'lines'
  - monitor = int: show menu on (bemenu w/ wayland: -1: all)
  - height = int: height of each menu line
  - index = int: select index automatically
  - prompt = str: prompt string of menu
  - prefix = str: prefix added highlighted item
  - scrollbar = str: display scrollbar {none,always,autohide}
  - font = str: font to be used format: "FONT-NAME [SIZE]"
  - title_background = str: title background color
  - title_foreground = str: title foreground color
  - normal_background = str: normal background color
  - normal_foreground = str: normal foreground color
  - filter_background = str: filter background color
  - filter_foreground = str: filter foreground color
  - high_background = str: highlight background color
  - high_foreground = str: highlight foreground color
  - scroll_background = str: scrollbar background color

- scroll_foreground = `str`: scrollbar foreground color
- selected_background = `str`: selected background color
- selected_foreground = `str`: selected foreground color
- windowid = `str`: embed into windowid

Raises

- ***CommandError*** –
- ***UsageError*** –
- ***FlagNameNotFoundError*** –
- **ValueError** – bad scrollbar options

Return type `Optional[str]`

Returns User's selected opt from `opts` or overridden-entered choice else `None` [Esc]

5.2 Errors

5.2.1 Error/Warnings

Menu errors

exception `launcher_menus.errors.`**CommandError**(*args*, *err*)
<MENU> command failed.

Parameters

- **args** (`list`) – args called with <menu> command.
- **err** (`str`) – error raised by <menu> command.

exception `launcher_menus.errors.`**FlagNameNotFoundError**(*command*, *flag*)
Flag not found for <menu> in menu-cfgs/<menu>.yml, nor provided via `**flags` or `config_yml`.

Parameters

- **command** (`str`) – command that was unsed as <menu>.
- **flag** (`str`) – flag that was not identified from yml file.

exception `launcher_menus.errors.`**MenuError**
<MENU> errors Base.

exception `launcher_menus.errors.`**UsageError**(*args*, *err*)
Usage error described by <menu> command.

Parameters

- **args** (`list`) – args called with <menu> command.
- **err** (`str`) – error raised by <menu> command.

5.3 Structure

5.3.1 Themes

Launcher Menu Themes

launcher_menus.themes.**custom_themes**(*custom_config=None*)
Read configuration file `themes.yml` from standard configuration locations and generate custom themes

Return type Dict[str, LauncherMenu]

launcher_menus.themes.**emergency_prompt = <launcher_menus.functions.LauncherMenu object>**
Emergency prompt menu

launcher_menus.themes.**menu = <launcher_menus.functions.LauncherMenu object>**
Plain `menu` object.

`command` defaults to the first one found to be installed.

launcher_menus.themes.**password_prompt = <launcher_menus.functions.LauncherMenu object>**
Password prompt menu.

CHAPTER

SIX

TODO

- Configure rofi
- Configure wofi: Is it dead?
- Write launcher-menu in python and integrate.
- Configure others as issues arise
- If Configuration is not found, try to create a flag using standard posix and action keywords.

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

l

INDEX

Symbols

C

E

F

L

M

O

P

U