
pympress documentation

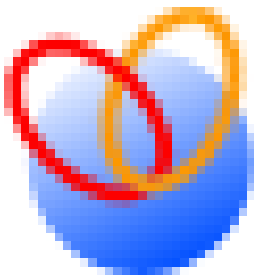
Release 1.6.4

Thomas Jost, Cimbali

27 August, 2021

Contents

1	Contents	1
2	Indices and tables	13
	Python Module Index	15
	Index	17



1.1 Pympress logo What is Pympress?

Pympress is a PDF presentation tool designed for dual-screen setups such as presentations and public talks. Highly configurable, fully-featured, and portable

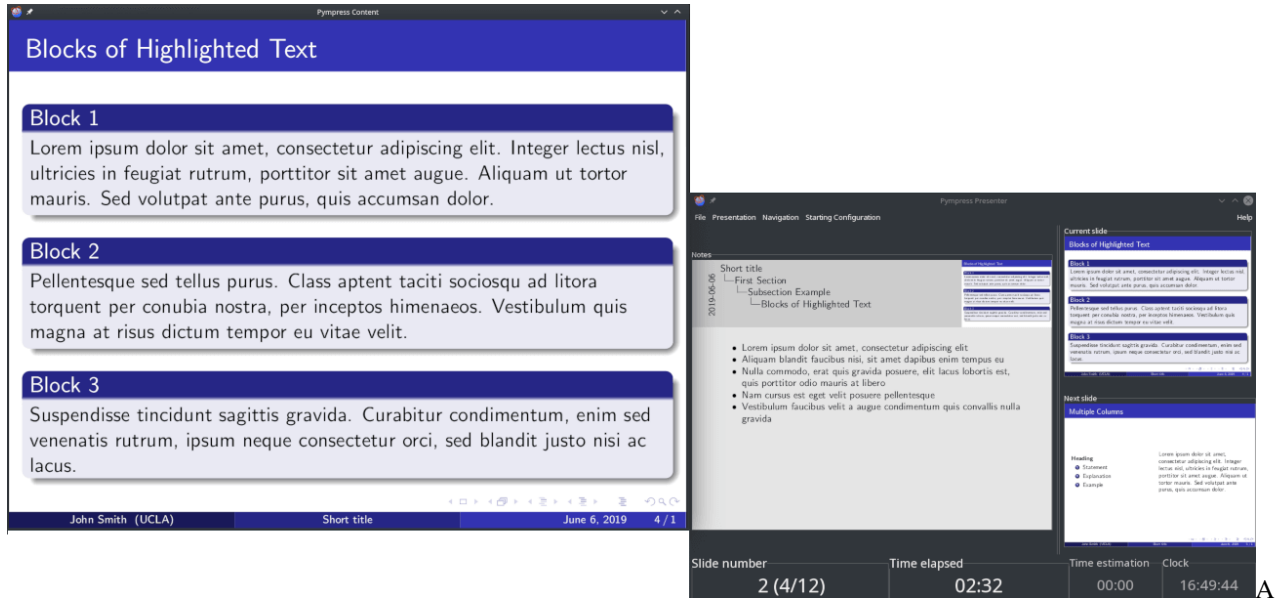
It comes with many great features (*more below*):

- supports embedded gifs (out of the box), videos, and audios (with VLC or Gstreamer integration)
- text annotations displayed in the presenter window
- natively supports beamer's *notes on second screen*, as well as Libreoffice notes pages!

Pympress is a free software, distributed under the terms of the GPL license (version 2 or, at your option, any later version).



Pympress was originally created and maintained by [Schnouki](#), on [his repo](#).

Here is what the 2 screen setup looks like, with a big notes slide next to 2 small slides (current and next) on the presenter





side:
screenshot with Pympress' 2 screens

1.2 Installing github version badge

- 
 ubuntu logo 20.04 focal or newer, Debian  debian logo 11 Bullseye or newer [ubuntu version badge](#) [debian version badge](#) (maintained by @mans0954)

```
apt-get install pympress libgtk-3-0 libpoppler-glib8 libcairo2 python3-gi python3-
↳gi-cairo gobject-introspection libgirepository-1.0-1 gir1.2-gtk-3.0 gir1.2-
↳poppler-0.18
```

- RPM-based Linux (Fedora  fedora logo CentOS  centos logo Mageia  mageia logo OpenSuse  suse logo RHEL) [Copr build version](#)

You can get pympress from the [pympress COPR repo](#) of your system. With yum or dnf, simply do:

```
dnf copr enable cimbali/pympress
dnf install python3-pympress

# optionally, required for VLC video support:
dnf install https://download1.rpmfusion.org/free/fedora/rpmfusion-free-release-
↳$(rpm -E %fedora).noarch.rpm
dnf install python3-vlc vlc
```

With zypper, fetch the link of the .repo in the table at the bottom of the COPR page and add it as a source.

```
zypper addrepo https://copr.fedorainfracloud.org/coprs/cimbali/pympress/repo/
↳opensuse-tumbleweed/cimbali-pympress-opensuse-tumbleweed.repo
zypper install python3-pympress python3-python-vlc vlc # both *vlc packages,
↳optional, required for VLC video support
```

If `python3-vlc` or `python3-python-vlc` are not available on your system, you can use `python3 -m pip install python-vlc`.

- Arch Linux  arch linux logo from AUR [AUR version badge](#) (maintained by @Jose1711)

```
git clone https://aur.archlinux.org/python-pympress.git
cd python-pympress
makepkg -si
pacman -S poppler-glib vlc # dependency temporarily missing from AUR package,
↔and VLC optional for video support
```

Or using any other tool to manage AUR packages (yay, pacaaur, etc.):

```
yay -S python-pympress
pacman -S poppler-glib vlc # dependency temporarily missing from AUR package,
↔and VLC optional for video support
```

- macOS  apple logo using Homebrew [homebrew version badge](#)

```
brew install pympress
brew cask install vlc # optional, to support playing videos with VLC integration
```

- Windows  windows logo with Chocolatey [chocolatey version badge](#) (maintained by @ComFreek)

```
choco install pympress vlc # vlc optional, to support playing videos with VLC
↔integration
```

Or download the latest installer from the [latest Github release](#).

- If you get an error message along the lines of “MSVCP100.dll is missing”, get the Visual C++ 2010 redistributables from Microsoft (x86 (32 bit) or x64 (64 bits)). Those libraries really should already be installed on your system.

- Other systems, directly from PyPI [pypi version badge](#) requires *python, gtk+3, poppler, and their python bindings*:

```
pip install "pympress[vlc_video]"
```

where `[vlc_video]` is optional and specifies the dependency on `python-vlc` for VLC video support.

- Make sure you have all *the dependencies*. (These are already included in binary packages or their dependencies.)
- Using pip, you may want to use `python3 -m pip` if pip points to the python 2.x pip. You may also want to install with the `--user` option, or install from github or downloaded sources. See [the python documentation on installing](#).
- If your python environment lacks the GObject Introspections module, try
 1. using `--system-site-packages` for [virtual environments](#),
 2. installing pygobject from pip (`pip install pygobject`, which requires the correct development/header packages. See [the PyPI installation instructions of PyGObject](#) for your system).

1.2.1 Notes

To support playing embedded videos in the PDFs, your system must have VLC installed (with the same bitness as pympress). VLC is not distributed with pympress, but it is certainly available in your system’s package manager and on their website.

1.3 Usage

1.3.1 Opening a file

Simply start Pympress and it will ask you what file you want to open. You can also start pympress from the command line with a file to open like so: `pympress slides.pdf` or `python3 -m pympress slides.pdf`

1.3.2 Functionalities

All functionalities are available from the menus of the window with slide previews. Don't be afraid to experiment with them!

Keyboard shortcuts are also listed in these menus. Some more usual shortcuts are often available, for example `Ctrl+L`, and `F11` also toggle fullscreen, though the main shortcut is just `F`.

A few of the fancier functionalities are listed here:

- **Two-screen display:** See on your laptop or tablet display the current slide, the next slide, the talk time and wall-clock time, and annotations (either PDF annotations, beamer notes on second slide, or Libreoffice notes pages). The position of the beamer or Libreoffice notes in the slide is detected automatically and can be overridden via a menu option.
- **Media support:** supports playing video, audio, and gif files embedded in (or linked from) the PDF file.
- **Highlight mode:** Allows one to draw freehand on the slide currently on screen.
- **Go To Slide:** To jump to a selected slide without flashing through the whole presentation on the projector, press `G` or click the “current slide” box. Using `J` or clicking the slide label will allow you to navigate slide labels instead of page numbers, useful e.g. for multi-page slides from beamer `\pause`.

A spin box will appear, and you will be able to navigate through your slides in the presenter window only by scrolling your mouse, with the `Home/Up/Down/End` keys, with the `+` and `-` buttons of the spin box, or simply by typing in the number of the slide. Press `Enter` to validate going to the new slide or `Esc` to cancel.
- **Software pointer:** Clicking on the slide (in either window) while holding `ctrl` down will display a software laser pointer on the slide. Or press `L` to permanently switch on the laser pointer.
- **Talk time breakdown:** The `Presentation > Timing Breakdown` menu item displays a breakdown of how much time was spent on each slide, with a hierarchical breakdown per chapters/sections/etc. if available in the PDF.
- **Automatic file reloading:** If the file is modified, pympress will reload it (and preserve the current slide, current time, etc.)
- **Big button mode:** Add big buttons (duh) for touch displays.
- **Swap screens:** If Pympress mixed up which screen is the projector and which is not, press `S`
- **Estimated talk time:** Click the `Time estimation` box and set your planned talk duration. The color will allow you to see at a glance how much time you have left.
- **Adjust screen centering:** If your slides' form factor doesn't fit the projectors' and you don't want the slide centered in the window, use the “Screen Center” option in the “Presentation” menu.
- **Resize Current/Next slide:** You can drag the bar between both slides on the Presenter window to adjust their relative sizes to your liking.
- **Preferences:** Some of your choices are saved in a configuration file, and more options are accessible there. See the [configuration file documentation](#) for more details.

- **Caching:** For efficiency, Pympress caches rendered pages (up to 200 by default). If this is too memory consuming for you, you can change this number in the configuration file.

1.3.3 Command line arguments

- `-h, --help`: Shows a list of all command line arguments.
- `-t mm[:ss], --talk-time=mm[:ss]`: The estimated (intended) talk time in minutes and optionally seconds.
- `-n position, --notes=position`: Set the position of notes on the pdf page (none, left, right, top, or bottom). Overrides the detection from the file.
- `--log=level`: Set level of verbosity in log file (DEBUG, INFO, WARNING, ERROR).

1.4 Dependencies

Pympress relies on:

- Python (version 3.x strongly recommended though 2.7 should still work fine).
- **Poppler**, the PDF rendering library.
- **Gtk+ 3**, a toolkit for creating graphical user interfaces, and [its dependencies](#), specifically:
 - **Cairo** (and python bindings for cairo), the graphics library which is used to pre-render and draw over PDF pages.
 - **Gdk**, a lower-level graphics library to handle icons.
- **PyGi**, the python bindings for Gtk+3. PyGi is also known as *pygobject3*, just *pygobject* or *python3-gi*.
 - Introspection bindings for poppler may be shipped separately, ensure you have those as well (`typelib-1_0-Poppler-0_18` on OpenSUSE, `gir1.2-poppler-0.18` on Ubuntu)
- optionally **VLC**, to play videos (with the same bitness as Python) and the `python-vlc` bindings.

1.4.1 On linux platforms

The dependencies are often installed by default, or easily available through your package or software manager. For example, on ubuntu, you can run the following as root to make sure you have all the prerequisites *assuming you use python3*:

```
apt-get install python3 python3-pip libgtk-3-0 libpoppler-glib8 libcairo2 python3-gi_
↪python3-cairo python3-gi-cairo gobject-introspection libgirepository-1.0-1_
↪libgirepository1.0-dev gir1.2-gtk-3.0 gir1.2-poppler-0.18
```

Different distributions might have different package naming conventions, for example the equivalent on OpenSUSE would be:

```
zypper install python3 python3-pip libgtk-3-0 libpoppler-glib8 libcairo2 python3-
↪gobject python3-gobject-Gdk python3-cairo python3-gobject-cairo typelib-1_0-
↪GdkPixbuf-2_0 typelib-1_0-Gtk-3_0 typelib-1_0-Poppler-0_18
```

On CentOS/RHEL/Fedora the dependencies would be:

```
yum install python36 python3-pip gtk3 poppler-glib cairo gdk-pixbuf2 python3-gobject_
↳python3-cairo
```

And on Arch Linux:

```
pacman -S --needed python python-pip gtk3 poppler cairo gobject-introspection poppler-
↳glib python-gobject
```

1.4.2 On macOS

Dependencies can be installed using [Homebrew](#):

```
brew install --only-dependencies pympress
```

1.4.3 On windows

The *binary installer for windows* comes with pympress and all its dependencies packaged.

Alternately, in order to install from pypi or from source on windows, there are two ways to get the dependencies:

1. using MSYS2 (replace x86_64 with i686 if you're using a 32 bit machine).

Warning: this can take a substantial amount of disk size as it requires a full software distribution and building platform.

```
pacman -S --needed mingw-w64-x86_64-gtk3 mingw-w64-x86_64-cairo mingw-w64-x86_64-
↳poppler mingw-w64-x86_64-python3 mingw-w64-x86_64-vlc python3-pip mingw-w64-x86_
↳64-python3-pip mingw-w64-x86_64-python3-gobject mingw-w64-x86_64-python3-cairo
```

This is also the strategy used to automate [builds on appveyor](#).

2. Using PyGobjectWin32. *Be sure to check the supported Python versions (up to 3.4 at the time of writing)*, they appear in the FEATURES list in the linked page.
 - Install native [python for windows](#)
 - Get GTK+3, Poppler and their python bindings by executing [the PyGi installer](#). Be sure to tick all the necessary dependencies in the installer (Poppler, Cairo, Gdk-Pixbuf).

Alternately, you can build your Gtk+3 stack from source using MSVC, see [the Gnome wiki](#) and [this python script that compiles the whole Gtk+3 stack](#). This strategy has not been used successfully yet, due to problems building Poppler with its introspection bindings (i.e. typelib) see [#109](#).

1.5 Contributing

Feel free to clone this repo and use it, modify it, redistribute it, etc, under the GPLv2+. A [number of contributors](#) have taken part in the development of pympress and submitted pull requests to improve it.

Be respectful of everyone and keep this community friendly, welcoming, and harrassment-free. Abusive behaviour will not be tolerated, and can be reported by email at me@cimba.li wrongdoers may be permanently banned.

Pympress has inline sphinx documentation ([Google style](#), contains rst syntax), and the docs generated from it are hosted on the [github pages](#) of this repo.

1.5.1 Translations

Czech French German Polish Spanish Italian

If you want to add or contribute to a translation, check [pympress' page on POEditor](#) and add your efforts to make pympress available in your own language to those of [@Vulpeculus](#), [@polaksta](#), [@susobaco](#), [Agnieszka](#), [FriedrichFröbel](#), [Jaroslav Svoboda](#), [Jeertmans](#), [Lorenzo](#), [pacchiardi](#), [Nico](#), [Saulpierotti](#), and [Cimbali](#).

1.5.2 Packages

Official releases are made to [PyPI](#) and with [github releases](#). The community maintains a number of other packages or recipes to install pympress (see *Install section*). Any additions welcome.

1.6 Configuration file

Pympress has a number of options available from its configuration file.

This file is usually located in:

- `~/.config/pympress` on Linux,
- `%APPDATA%/pympress.ini` on Windows,
- `~/Library/Preferences/pympress` on macOS,
- in the top-level of the pympress install directory for portable installations.

The path to the currently used configuration file can be checked in the `Help > About` information window.

1.6.1 Shortcuts

The shortcuts are parsed using `Gtk.accelerator_parse()`:

The format looks like “<Control>a” or “<Shift><Alt>F1” or “<Release>z” (the last one is for key release).

The parser is fairly liberal and allows lower or upper case, and also abbreviations such as “<Ctl>” and “<Ctrl>”. Key names are parsed using `Gdk.keyval_from_name()`. For character keys the name is not the symbol, but the lowercase name, e.g. one would use “<Ctrl>minus” instead of “<Ctrl>-”.

This means that any value in this [list of key constants](#) is valid (removing the initial `Gdk.KEY_` part). You can verify that this value is parsed correctly from the `Help > Shortcuts` information window.

1.6.2 Layouts

The panes (current slide, next slide, notes, annotations, etc.) can be rearranged arbitrarily by setting the entries of the `layout` section in the configuration file. Here are a couple examples of layouts, with `Cu` the current slide, `No` the notes half of the slide, `Nx` the next slide:

- All-horizontal layout:

```
+-----+-----+-----+
| Cu | No | Nx |
+-----+-----+-----+
```

Setting:

```
notes = {"children": ["current", "notes", "next"], "proportions": [0.33, 0.33, 0.
↪33], "orientation": "horizontal", "resizeable": true}
```

- All-vertical layout:

```
+-----+
| Cu |
+-----+
| No |
+-----+
| Nx |
+-----+
```

Setting:

```
notes = {"children": ["current", "notes", "next"], "proportions": [0.33, 0.33, 0.
↪33], "orientation": "vertical", "resizeable": true}
```

- Vertical layout with horizontally divided top pane:

```
+-----+-----+
| Cu | No |
+-----+-----+
|      Nx      |
+-----+-----+
```

Setting:

```
notes = {"children": [
    {"children": ["current", "notes"], "proportions": [0.5, 0.5],
↪"orientation": "horizontal", "resizeable": true},
    {"children": ["next", "notes"], "proportions": [0.5, 0.5],
↪"orientation": "vertical", "resizeable": true}
```

- Horizontal layout with horizontally divided right pane:

```
+-----+-----+
|      | Nx |
+ Cu +-----+
|      | No |
+-----+-----+
```

Setting:

```
notes = {"children": [
    "current",
    {"children": ["next", "notes"], "proportions": [0.5, 0.5],
↪"orientation": "vertical", "resizeable": true},
    {"children": ["next", "notes"], "proportions": [0.5, 0.5],
↪"orientation": "horizontal", "resizeable": true}
```

And so on. You can play with the items, their nesting, their order, and the orientation in which a set of widgets appears. For each entry the widgets (strings that are leaves of “children” nodes in this representation) must be:

- for notes: “current”, “notes”, “next”
- for plain: “current”, “next” and “annotations” (the annotations widget is toggled with the A key by default)

- for `highlight`: same as `plain` with “highlight” instead of “current”

A few further remarks:

- If you set “resizeable” to `false`, the panes won’t be resizeable dynamically with a handle in the middle
- “proportions” are normalized, and saved on exit if you resize panes during the execution. If you set them to 4 and 1, the panes will be $4 / (4 + 1) = 20\%$ and $1 / (4 + 1) = 100\%$, so the ini will contain something like 0.2 and 0.8 after executing `pympress`.

1.6.3 Themes on Windows

`Pympress` uses the default `Gtk` theme of your system, which makes it easy to change on many OSs either globally via your `Gtk` preferences or [per application](#). Here’s the way to do it on windows:

1. Install a theme

There are 2 locations, either install the theme for all your `gtk` apps, e.g. in `C:\Users\%USERNAME%\AppData\Local\themes`, or just for `pympress`, so in `%INSTALLDIR%\share\themes` (for me that’s `C:\Users\%USERNAME%\AppData\Local\Programs\pympress\share\themes`)

Basically pick a theme e.g. [from this list of dark themes](#) and make sure to unpack it in the selected directory, it needs at least `%THEMENAME%\gtk-3.0\gtk.css` and `%THEMENAME%\index.theme`, where `THEMENAME` is the name of the theme.

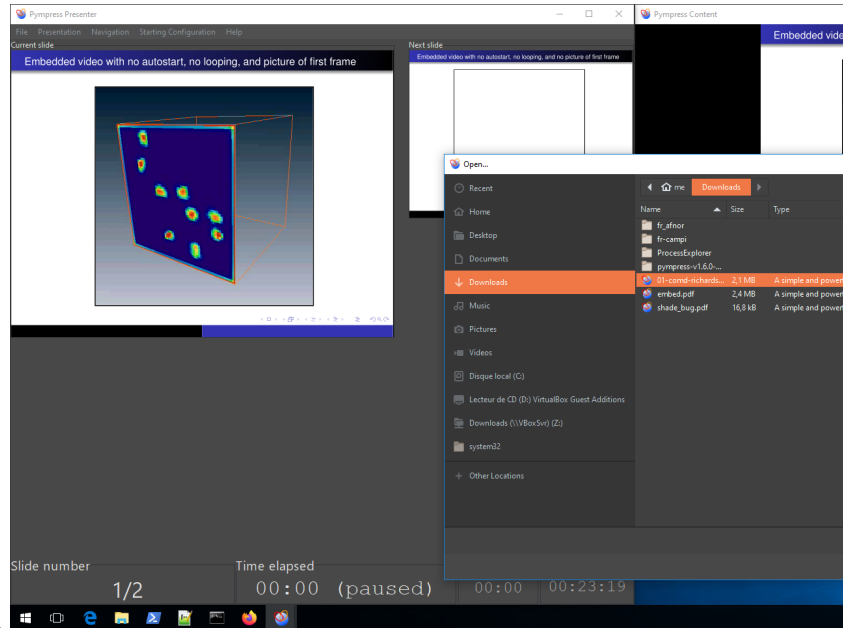
There are 2 pitfalls to be aware of, to properly install a theme:

- themes that are not self-contained (relying on re-using `css` from default linux themes that you might not have), and
- linux links (files under `gtk-3.0/` that point to a directory above and that need to be replaced by a directory containing the contents of the target directory that has the same name as the link file).

2. Set the theme as default

Create a `settings.ini` file, either under `C:\Users\%USERNAME%\AppData\Local\gtk-3.0` (global setting) or `%INSTALLDIR%\etc\gtk-3.0` (just `pympress`) and set the contents:

```
[Settings]
gtk-theme-name=THEMENAME
```



Here's what it looks like with the [Obscure-Orange](#) theme.

In testing this found these 2 stackoverflow questions useful:

- [Change GTK+3 look on Windows](#) which contains a list of all interesting directories
- [How to get native windows decorations on GTK3 on Windows 7+ and MSYS2](#) which details the process

1.7 Pympress package

This page contains the inline documentation, generated from the code using sphinx.

The code is documented in the source using the [Google style](#) for docstrings. Sphinx has gathered a [set of examples](#) which serves as a better crash course than the full style reference.

Restructured text (rst) can be used inside the comments and docstrings.

1.7.1 Modules

`pympress.util` – various utility functions

`pympress.util.fileopen(f)`

Call the right function to open files, based on the platform.

Parameters `f` (`str`) – path to the file to open

`pympress.util.get_default_config()`

Returns the path to the configuration file containing the defaults.

Returns The path to the portable configuration file.

Return type `str`

`pympress.util.get_icon_path(name)`

Load an image from pympress' resources in a Gdk Pixbuf.

Parameters `name` (`str`) – The name of the icon to load

Returns The loaded icon

Return type `Pixbuf`

`pympress.util.get_locale_dir()`
Returns the path to the locale directory.

Returns The path to the locale directory

Return type `str`

`pympress.util.get_log_path()`
Returns the appropriate path to the log file in the user app dirs.

Returns path to the log file.

Return type `str`

`pympress.util.get_portable_config()`
Returns the path to the configuration file for a portable install (i.e. in the install root).

Returns The path to the portable configuration file.

Return type `str`

`pympress.util.get_pympress_meta()`
Get metadata (version, etc) from pympress' `__init__.py` or `git describe`.

Returns metadata properties (version, contributors) mapped to their values

Return type `dict`

`pympress.util.get_ui_resource_file(name, ext='.glade')`
Load an UI definition file from pympress' resources.

Parameters

- **name** (`str`) – The name of the UI to load
- **ext** (`str`) – The extension of the file

Returns The full path to the glade file

Return type `str`

`pympress.util.get_user_config()`
Returns the path to the configuration file in the user config directory.

Returns path to the user configuration file.

Return type `str`

`pympress.util.hard_set_screensaver(disabled)`
Enable or disable the screensaver.

Parameters **disabled** (`bool`) – if `True`, indicates that the screensaver must be disabled; otherwise it will be enabled

`pympress.util.list_icons()`
List the icons from pympress' resources.

Returns The paths to the icons in the pixmaps directory

Return type `list of str`

`pympress.util.load_style_provider(style_provider)`
Load the css and in a style provider.

Parameters `style_provider` (`CssProvider`) – The style provider in which to load CSS

Returns The style provider with CSS loaded

Return type `CssProvider`

CHAPTER 2

Indices and tables

- genindex
- modindex

p

`pympress.util`, 10

F

`fileopen()` (in module *pympress.util*), 10

G

`get_default_config()` (in module *pympress.util*), 10

`get_icon_path()` (in module *pympress.util*), 10

`get_locale_dir()` (in module *pympress.util*), 11

`get_log_path()` (in module *pympress.util*), 11

`get_portable_config()` (in module *pympress.util*), 11

`get_pympress_meta()` (in module *pympress.util*), 11

`get_ui_resource_file()` (in module *pympress.util*), 11

`get_user_config()` (in module *pympress.util*), 11

H

`hard_set_screensaver()` (in module *pympress.util*), 11

L

`list_icons()` (in module *pympress.util*), 11

`load_style_provider()` (in module *pympress.util*), 11

P

`pympress.util` (module), 10