Manual for Pi Presents
2/1/2013, Ken Thompson

READING THE MANUAL

The manual is issued as a PDF because it was too large and unwieldy to be a text file. Raspbian has a pdf reader. However there are a couple of issues with this:

- The currently installed pdf reader mupdf is somewhat limited. Install xpdf using

```
sudo apt-get install xpdf
```

You cannot cut and paste from xpdf. I have therefore provided a text file install_instructions.txt with the necessary information.

1 Introduction

Pi Presents is a presentation application intended for Museums and Visitor Centres. I am involved with a couple of charity organisations that are museums or have visitor centres and have wanted a cheap way to provide audio interpretation and slide/videoshow displays. Until the Raspberry Pi arrived buying or constructing even sound players was expensive. The Raspberry Pi with its combination of Linux, GPIO and a powerful GPU is ideal for black box multi-media applications; all it needs is a program to harness its power in a way that could be used by non-programmers.

This initial issue of Pi Presents offers some basic features but under the bonnet it is, I hope, a flexible modular application with simple to use editing facilities to configure it to your exact display requirements. The initial features include:

- Animation or interpretation of exhibits by triggering a sound, video, or slideshow from a PIR or button.

- A repeating media show for a visitor centre. Images, videos, audio tracks, and messages can be displayed.

- Allow media shows to be interrupted by the visitor and a menu of videos to be presented.

- Showing 'Powerpoint' like presentations where progress is controlled by buttons or keyboard. The presentation may include images, text, audio tracks and videos.

I have immediate plans to develop an audio player based on pulseaudio which should replace omxplayer for audio tracks and allow use of the 3.5mm jack without popping. Also the ability to asynchronously trigger a number of tracks from GPIO pins. Other ideas are animation of exhibits using GPIO outputs, synthetic speech, and display of animations using OpenGLES.

There are potentially many applications of Pi Presents and your input on real world experiences would be invaluable to me, both minor tweaks to the existing functionality and major improvements.

Once set up, use of Pi Presents is simple and does not require a network:

- Applications can be prepared using a simple to use editor

- Will operate on a Model A Raspberry Pi.

- No need to modify the Pi's SD card after initial installation. All media and configuration options can be kept on a removable usb stick and can edited on a Linux machine or Windows PC (tbd). Installing a new application is as simple as inserting the USB stick.

- Control can be by buttons or PIR switched to 0 volts. It is straightforward to make your own controls in a box containing the Pi, or perhaps a wired remote. Alternatively a keyboard can be used.

- Video is selectable between HDMI or composite using the normal Pi setup procedure. Audio output is selectable on a per track basis. Amplification and volume control can use readily available PC loudspeakers.

- If a LAN is available configuration data and media can easily be FTP'ed to Pi Presents.

PI Presents, out of the box, runs as a desktop application on the Raspberry Pi using the keyboard for control. However with a little bit of Linux magic it can be made to run as a black box application with control from GPIO; full instructions are given below. Black box features include:

- Disabling screen blanking
- Full screen operation without window decorations
- Black screen behind videos
- Completely headless operation without a keyboard, mouse or buttons (except perhaps a shutdown button if you are not brave)
- Optional operation with pushbuttons or PIR through GPIO
- Automatic start up when power is applied to the Pi
- All media can be on a USB stick or on the SD card
- Safe shutdown without using keyboard or mouse

While Pi Presents is easy to set up for the majority of tasks using the editor application; it also very flexible allowing complex combinations of media shows to be produced and triggered.

2 Installation

Install required applications (PIL and X Server utils) as described in the install_instructions.txt file.

Download and install p-expect as specified here http://www.noah.org/wiki/pexpect#Download_and_Installation and in the install_instructions.txt file.

Download Pi Presents from the pipresents github repository as described in the install_instructions.txt file.

Now run Pi Presents to check the installation is successful. From a terminal window opened in the pipresents directory type:

```
python pipresents.py
```

You will see a welcome message followed by an error message which is because you have no profiles. Exit Pi Presents using CTRL-BREAK or close the window.

3 Try Some Examples

Download the examples from the pipresents-examples github repository as described in the install_instructions.txt file.

From a terminal window opened in the pipresents directory type:

```
python pipresents.py –p pp_mediashow
```

You will see a continuous looping show.

The pipresents-examples repository has examples of how to use Pi Presents:

- pp_mediashow
  The profile you have just run. The show will start immediately, progress automatically, and repeat after 70 seconds. Use the left and right cursor keys to skip.

- pp_menu
  Use the Up and Down cursor keys to traverse the menu Return to Start the track, and Escape to terminate it.

- pp_exhibit
  This demonstrates how to trigger a mediashow from a PIR or button. When started the screen will be blank. Pressing the Play button or triggering the PIR will start a one shot mediashow and then wait for another trigger. Use the Return key instead of the PIR to see the effect.

- pp_interactive
  This combines a mediashow and a menu. The mediashow is run continuously but every track has a hint showing that a menu can be triggered by the Return key.

- pp_presentation
  The mediashow is configured as a presentation. Use the Return key to start the presentation and the Left and Right cursor keys to traverse through it.

All examples use a selection of media tracks which are in pp_home/media. The profiles are in /pp_home/pp_profiles. The pp_showlist.json file specifies the look and feel of the show; other files specify the media to use. The files can be viewed in a text editor.

The keyboard commands are:

- Up, Down Cursor- move through a menu
- Left Right Cursor - move through a presentation or skip to the next track in a mediashow
- Return – Start a presentation or play the selected menu entry.
- Escape - Back a level
- In a video or audio track p or spacebar- pause/resume, Escape to stop.
- CTRL-BREAK always exits Pi Presents

4 pp_editor - The Pi Presents Profile Editor

Configuration uses a separate program which can be run on the Pi, on a Linux machine, or on a Windows PC.

TBD

5 Black Box Operation

There are a number of things to set up to make Pi Presents into a full screen, auto starting, GPIO controlled application. You do not need to use them all.

5.1 Command Line Options

`python pipresents.py -h` will show the command line options

Options	
-p --profile <arg>	Base name of the profile to be used. e.g. pp_mediashow If this is not specified then it defaults to pp_`profile`.
-g --gpio	Use the GPIO for buttons and PIR. To use this Pi Presents must be run as root: `sudo python pipresents.py -g` The keyboard can still be used if gpio is enabled.
-b --noblank	Disable screen blanking. For this to function x11-server-utils must have been installed.
-f --fullscreen <arg>	Run Pi Presents in full screen mode. For this to function it must be set up as described below. <arg> is vertical or horizontal, the orientation of your taskbar
-o --home <arg>	Location of the Pi Presents 'home' directory' e.g. /media/USBSTICK or /home/pi/my_data If this option is not specified it defaults to /home/pi
-d --debug	Run time errors give alerts and are reported to the terminal window if open. Turning on debugging will in addition provide a trace of the operation of Pi Presents in the terminal window. Errors and the trace are also recorded in the /pipresents/pp_log.log file.
-c --code <arg>	Specifies the location of Pi Presents. If not used then the default is /home/pi/pipresents.

5.2 Specify a Profile

The --profile (-p) command line option specifies the profile to use. This is the name of the profile directory in the pp_profiles directory. If the option is omitted it will default to pp_profile.

The prefix pp_ means nothing special other than denoting files provided by the developer.

5.3 Specify a Home Directory

The --home (-o) command line option specifies the location of Pi Presents data home. If the option is omitted it will default to /home/pi.

Hence if both --profile and --home options are omitted the profile directory will be:

```
/home/pi/pp_home/pp_profiles/pp_profile
```

which should contain at least a pp_showlist.json file and optionally xxx.json medialist files.

Media files should ideally be stored under:

```
/home/pi/pp_home
```

sub-directories are permissible.

USB sticks can be used to hold media. They will be assigned mount points in the /media directory.

e.g. /media/KINGSTON

Raspbian will auto mount USB sticks if they are present at power on, or plugged in afterwards. It takes up to 10 seconds for the drive to be mounted after power on.

Raspbian will compute the mount point from the name of the drive on the stick. If preparing the USB Stick on Windows machines it appears Windoze converts all drive names to upper case.

5.4 Using GPIO to Control Pi Presents

GPIO control is enabled using the --gpio command option. If GPIO is used then the command starting Pi Presents must be preceded by sudo.

The following Pins of the GPIO Connector 1 correspond to the keyboard keys.

Pin	Button	Equivalent Key	
P1-12	Shutdown	-	Press for 5 seconds to exit Pi Presents and shut down the Pi.
P1-15	Down	Cursor Down for Menus Cursor Left for Mediashows	
P1-16	Up	Cursor Up for Menus Cursor Right for Mediashows	
P1-18	Play	Return	
P1-22	Pause	p or spacebar	toggle pause for videos.
P1-7	Stop	Escape	Stop playing a track or return to the higher level show.

P1-11	PIR	Return	Currently acts the same as Play button; debounce can be configured differently.
-	-	CTRL-BREAK	Abort Pi Presents. Focus must be on a Pi Presents window.

The GPIO ports are set up as leading edge triggered inputs with internal pull-up resistors. Push buttons should be mechanical and connected to Ground. A 330 ohm resistor is series with the button is recommended to protect the Pi should the inputs inadvertently be used as outputs. PIR's should have relay contacts and be connected in a similar manner.

Pi Input ----- 330R ----- contact ---- 0 volts

There is software contact de-bouncing which is set with a small hysteresis. If you have problems with contact bounce increase the THRESHOLD of the appropriate pin by modifying pp_buttons.py

5.5 Disable Screen Blanking

To disable screen blanking you must first install xset which is part of the x server utilities package.

```
sudo apt-get install x11-xserver-utils
```

You can than use the --noblank command option to disable screen blanking.

5.6 Make Pi Presents Occupy the Whole Screen

Move the Taskbar and minimize it:

- Right click on the Taskbar and select Panel setting
- Select the Appearance Tab. Select Solid colour and, choose black with 100% opacity.
- Select the Advanced Tab
  Set 'minimise panel when not in use' to On
  Set 'Size when minimized' to 2 pixels

Change the Openbox configuration so it removes window decorations for Pi Presents (only when --fullscreen option used):

- Edit the file /home/pi/.config/openbox/lxde-rc.xml to add the following after the line `<applications>`

```
<application name="fspipresents">
     <decor>no</decor>
</application>
```

  which can be achieved as user Pi using Leafpad. I suggest you make a backup first.

- Use the --fullscreen command option to switch full screen on or off and to specify the orientation of the taskbar.

5.7 *Start Pi Presents when Power is Applied to the Pi*

This will work only if you have set 'boot to desktop' using raspi-config.

- Create the folder home/pi/.config/lxsession/LXDE/

  Note: The directory .config is already present in the image but you will need to select 'Show Hidden Files' in the File Manager to see it.

- In this folder put a file named `autostart` containing one line specifying the full path:

  e.g. `python /home/pi/pipresents/pipresents.py <options>`

- Make the autostart file executable ?????.

5.8 *Shutdown the Raspberry Pi from the GPIO*

Press the Shutdown button for more than 5 seconds. Pi Presents will exit and shut down the Pi.

6 Anatomy of a Profile

Configuration data and media

A profile directory contains the information needed to define a single application of Pi Presents. It contains the pp_showlist.json file and one or more medialist (.json) files. All Pi Presents profiles should be kept in the directory pp_profiles which is in the directory pp_home.

For portability all media should also be kept in the directory pp_home, but this is not essential. Media can be kept anywhere. If you have more than one profile it may best to keep the media for each profile in a separate directory.

There is a pp_home is in the pipresents directory, but this should not be used as it contains internal data. Using command line options it can be situated elsewhere.

Medialist (.json) files define the content each show and some per track configuration options. Shows of the pp_showlist.json file define the look and feel of the shows in an application and how they link together.

Currently there are two types of show, mediashow and menu.

The pp_showlist.json file contains a number of shows, a 'start' show and a number of sections each defining a show

6.1 Shows

6.1.1 Start Show

This must be present and serves just to initiate the first user defined show.

Field	Examples	Values
type	start	A special type
title	First Show	Text describing the show displayed in the editor.
show-ref	start	must be 'start'
start-show	myfirstshow	Any text without spaces.

6.1.2 Mediashow

A 'Media show' is a sequence of one or more images, videos, messages, menus, mediashows or soundtracks. The content is defined by a medialist (.json) file. A show in the pp_showlist.json defines the look and feel of the show.

Field	Examples	Values
		Essential information
type	menu	
title	My First Show	Text describing the show displayed in the editor or menus.
show-ref	myfirstshow	A 'label' by which the show is referenced by other shows. Any text without spaces
medialist	mymedia.json	Filename of the medialist file containing the tracks for the menu.
		How the show is to be run
trigger	start	How the media show is started: • start - start when Pi Presents starts • PIR - wait for a trigger from a PIR, • button - wait for the Play button or Return key
repeat	interval	How the media show is repeated: • oneshot - mediashow is run once • interval - run at intervals
repeat-interval	10	The time between the start of one mediashow and the start of the next in seconds. If the value is less than the length of the show play will be continuous. A value of 0 will always give a continuous repeat of the show
progress	auto	How the show progresses between tracks: • manual - Pi presents will wait for next/previous keys/buttons between tracks. • auto - tracks play continuously.
sequence		sequence of tracks: • ordered - played in the order of the medialist
		Mediashows can optionally provide an entry point for a menu. If the Return key or Play button is pressed the menu will be started.

		The mediashow will continue when the menu terminates.
has-child	yes	yes/no. If yes the child is enabled. This is usually a menu but could be a mediashow. The child is defined in the medialist with the track-ref of pp-child-show.
		A hint is a line of text displayed near the bottom of the screen when has-child is yes. Used as a hint on the existence of a menu and how to enter it.
hint-text	Press Play to..	
hint-font	Helvetica 30 bold	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
hint-colour	white	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
hint-y	100	distance of the text from the bottom of the screen (pixels)
		Tracks played in the show need some configuration if not supplied in the medialist
transition	cut	cut, fade, slide, crop. Type of transition between tracks. Only cut works until Pi Presents gets GPU acceleration. Others are implemented as an experiment, but incorrectly.
duration	10	How long an image or message is displayed in seconds. A value of 0 displays continuously.
omx-audio	hdmi	hdmi/local/<blank>. Sound output channel for any track played by omxplayer from the show.
omx-other-options		other options for omxplayer (care required to avoid have a nice day!)

6.1.3 Menu

A 'Menu' is a collection of one or more images, videos, messages, menus, mediashows or soundtracks. The content is defined by a medialist (.json) file. A show in the pp_showlist.json defines the look and feel of the menu.

Field	Examples	Values
		Essential information
type	menu	
title		
show-ref	mymenu	A ‘label’ by which the show is referenced by other shows. Any text without spaces
medialist	themenu.json	Filename of the medialist file contaning the tracks for the menu.
		The look and feel of the menu
menu-x	300	integer, x position of the start of the first line of the menu (pixels from left of screen)
menu-y	250	integer, y position of the start of the first line of the menu (pixels from top of screen)
menu-spacing	70	integer, gap between menu entries (top to top)
entry-font	Helvetica 30 bold	Tkinter font name see: http://effbot.org/tkinterbook/tkinter-

		widget-styling.htm
entry-colour	black	Tkinter colour see: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
entry-select-colour	red	colour when the entry selected
timeout	60	seconds, If there is no activity on the menu Pi presents automatically returns to the previous display. 0 for no timeout.
has-background	yes	yes/no. If yes an image is displayed behind the menu. The image is defined in the medialist with the track-ref of pp-menu-background.
		Menus cannot have children; the hint is used for track playing instructions.
hint-text	To Play..	
hint-font	Helvetica 30 bold	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
hint-colour	white	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
hint-y	100	distance of the text from the bottom of the screen (pixels)
		Tracks played from a menu need some configuration if not supplied in the medialist
transition	cut	see mediashow for values
duration	10	see mediashow for values
omx-audio	hdmi	hdmi/local/<blank> Sound output channel for any track played by omxplayer from the show.
omx-other-options	-t 1	other options for omxplayer (care required)

6.2 *Medialists*

Medialists define the content of a show. Each entry is a 'track'. Tracks can be of various types:

- image - a still image. Image types are currently those that can be rendered by the Python Imaging Library. Image size should be kept to the 1 megapixel region.
- video - a track played by omxplayer. Track types depend on the codec licences purchased from the Foundation.
- audio - an audio track played by aplayer on some such. This is currently not implemented; the aim is to enable a number of parallel asynchronous tracks to be played from one Pi. Currently audio tracks are played by omxplayer.
- message - displays lines of text against a black background. Can also used to display a blank screen.
- show - shows can be used as tracks allowing nesting to any depth.

- child show - a special type of track to specify the child show of a mediashow.

- menu background – a special type of track to specify the image file to be used as the background for a menu.

Media Track Path References
The leading + sign in file paths allows tracks to be specified relative to the pp_home directory. If a plus sign is not present then the path must be absolute. It is recommended that media files be stored under pp_home if the profiles are to be portable.

Track Reference Labels
Some medialist entries will have labels defined by the key 'track-ref'. If the label is blank then the track is included in the menu or medialist. If the label is not blank then the track is used for a special purpose. Currently there is two special purposes;

- The background image for a menu which has the label pp-menu-background

- The child show of a mediashow which has the label pp-child-show

Labels will be used in the future to respond to asynchronous events.

The field definitions for each type of medialist track follow.

6.2.1 Image

Field	Example	Values
track-ref		A label for the track - blank so that the track is included in a menu or mediashow.
type	image	
title		Displayed on a menu and in the editor
location	+/media/sarename.gif	The filename of the track. (relative)
duration	5	Seconds. . If 0 then image is displayed until terminated by user input. If omitted then the value in the referencing show is used.
transition	cut	cut/fade/rotate/slide/. If omitted then the value in the referencing show is used. (Only cut currently works)

6.2.2 Video

A track to be played with omxplayer, currently audio tracks use this player.

Field	Example	Values
track-ref		a label for the track - blank so that the track is included in a menu or mediashow.

type	video	
title	The Film	Displayed on a menu and in the editor
location	/home/pi/film.mp4	The filename of the track. (absolute)
omx-audio	local	hdmi/local. If omitted then the value in the referencing show is used.

6.2.3 Message

Field	Example	Values
track-ref		A label for the track - blank so that the track is included in a menu or mediashow.
type	message	
title	A Message	Displayed on a menu and in the editor
text	Welcome	The text to be displayed. Newlines can be inserted using \n
duration	5	Seconds. If 0 then message is displayed until terminated by user input. If omitted then the value in the referencing show is used.
message-font	Helvetica 30 bold	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
message-colour	white	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm

6.2.4 Show

Menus or mediashows can be a track in another show or menu. Uses might be:

- A mediashow made up of smaller mediashows
- A menu of mediashows, particularly presentations
- Menus with sub-menus

Field	Example	Values
track-ref		A label for the track - blank so that the track is included in a menu or mediashow.
type	show	
title	My Other Show	Displayed on a menu and in the editor
sub-show	myothershow	Reference to the show to be run

6.2.5 Child Show

This is a special medialist entry which provides a referencing mechanism for the child show of a mediashow.

track-ref	pp-child-show	The label indicates this contains a reference to a child show. The track will not be included in the menu or mediashow.
type	show	

title	Child Show	Displayed on a menu and in the editor
sub-show	mychildshow	Reference to the show to be run

6.2.6 Menu Background

Field	Example	Values
track-ref	pp-menu-background	The label indicates this contains a reference to a menu background. The track will not be included in the menu or mediashow.
type	image	
title	Menu Background	Displayed on a menu and in the editor
location	+/media/sarename.gif	The filename of the track. (relative)

7 Hardware Requirements

Pi Presents can be used with either Rev 1 or Rev 2 Pi's. The GPIO pins have been chosen such that they are version agnostic. 256MB Pi's can be used provided image size kept to the 1 megapixel region.

Display of images is slow. A one megapixel images takes a couple of seconds to display. The one 10 megapixel image I tried took 10 seconds to display and crashed a 256MB RPi.

Use the HDMI or 3.5mm jack output for audio. Amplification and volume control will need to be provided in external hardware to suit the application.

8 Updating Pi Presents

Download pipresents from github as described in the install_scraps.txt file. Data stored in /home/pi/pp_home will not be affected. Do not store your data in the pipresents/pp_home directory as it may be overwritten

9 Bug Reports and Feature Requests

Please use Github Issues or the Pi Presents thread on the RPi Forum to report bugs.

I am keen to improve Pi Presents to ?????????

10 Known Problems

10.1 Permission to write to pp_log.log is denied.

The log file pp_log.log is created with the user permissions currently employed. If sudo was used in the command that created the file it is owned by root. Delete the file and then create with user Pi or run pipresents.py without sudo.