Pi Presents

A Multi-media Display and Animation Toolkit
for Museums, Visitor Centres and more.....

Running on the Raspberry Pi

Ken Thompson

http://pipresents.wordpress.com

Contents

Copyright Notice

1 Introduction

Pi Presents is a multi-media presentation toolkit with animation control facilities. It was originally intended for Museums and Visitor Centres but has already found uses in hospitals, shops, art installations, and libraries.

I am involved with a couple of charities that have museums or visitor centres. For several years I searched for a cost effective way to provide audio interpretation of exhibits and slide/video show displays for the reception area. Until the Raspberry Pi arrived buying or constructing something even as simple as an interactive audio player 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 needed was a program to harness its power in a way that could be used by non-programmers.

The Pi Presents toolkit currently supports five types of show commonly seen in museums and each of these can be configured to meet the your individual requirements. The toolkit includes a simple to use show editor running on the Pi or a Windows PC.

The toolkit allows construction of:

- A show for the animation or interpretation of exhibits by triggering a sound, video, or slideshow from a button, keyboard or other GPIO input.
- While playing media, GPIO outputs can be used to control external devices such as lights, moving machinery, animatronics etc.
- An automatic show for a visitor centre. Images, videos, audio tracks, and messages can be displayed. Repeats can be at intervals or at specified times of day.
- Automatic shows to be interrupted by the visitor and a menu of further content presented.
- A show for kiosks where content can be initiated by pressing one of a number of buttons.
- Facilities to construct shows commonly seen on touch screens in museums.
- It can be used to construct a 'powerpoint' like multi-media presentation where progress through slides is manually controlled by buttons or keyboard.
- It has facilities for basic remotely controlled digital signage with the ability upload image, video or audio tracks over a network for display in a repeating show.
- Run multiple shows simultaneously so that more than one exhibit can be controlled from one Pi or complex interactive exhibits can be controlled.
- Foreign language support.

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 Pi Presents is easy to use and does not require a network:

- Shows can be prepared using a simple to use editor on a Windows PC, the Pi, or another Linux box.
- Will operate on a Model A Raspberry Pi (but see hardware requirements).
- 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. Installing a new application is as simple as inserting the USB stick.
- Video output can use HDMI or composite selected using the normal Pi setup procedure. As Pi Presents can be started and shutdown without a keyboard a monitor is not required.
- Audio track output is selectable between HDMI and analogue and can be output from left, right or stereo speakers on a per track basis. A number of tracks can be played simultaneously to different speakers.

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, which is explained in this manual, it can be made to run as a black box application with control from GPIO. Black box features include:

- Disabling screen blanking and the mouse pointer.
- Full screen operation without window decorations
- Completely headless operation without a keyboard, mouse or buttons (except perhaps a shutdown button if you do not want to risk SD Card corruption)
- Black box control can be by buttons, PIR, or any source of digital inputs. It is straightforward to make your own controls.
- Mouse compatible touchscreens are supported if drivers are available for the Raspberry Pi. Alternatively a keyboard and mouse can be used. Pi Presents has been designed such that it is possible to add drivers written in Python for other forms of input and output such as RS232 and I2C.
- Automatic start up when power is applied to the Pi
- Safe shutdown without using keyboard or mouse
- Automatic shutdown of the Pi at a specified time of day.

1.1 Acknowledgements

Johannes Baiter (jbaiter) for pyomxplayer based on Noah's pexpect

Numerous people on the Raspberry Pi forum and websites, particularly StackOverflow, who have unwittingly provided me with solutions to many technical problems and taught me Python.

ActiveState for a free Python systems for Windows

The Raspberry Pi Foundation for the first affordable machine that plays video, audio and has GPIO.

Having been introduced to open source software through the Raspberry Pi I was amazed at the time and effort which so many people put in to produce software for others, so I thought it worthwhile to do the same with Pi Presents. I am glad I did because the feedback I have had from users and potential users has given me so many good ideas for extensions to Pi Presents. Needless to say I have been so busy producing Pi Presents that I have not had time to use it for my own intended projects, however one day I am sure I will ☺.

2 Installation

The installation instructions are in the README.md file. Pi Presents is aimed at the latest Foundation Raspbian release and must be installed and run from the LXDE desktop.

3 Try Some Examples

Download the examples from the pipresents-examples github repository as described in the README.md 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. The examples are designed for a 1920*1080 display:

- pp_mediashow
  The profile you have just run. The show will start immediately, progress automatically, and repeat after 200 seconds. Use the Up and Down cursor keys to skip tracks.

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

- pp_exhibit
  This demonstrates how to trigger a mediashow from a PIR. When started, the screen will be blank. Triggering the PIR input (opening a contact connected to

0 volts on P1-11) or pressing a button (closing a contact connected to 0 volts on P1-18) will start a one shot mediashow and then wait for another trigger.

It uses a gpio.cfg in the profile which has pins P1-18 and P1-11 bound to the symbolic name PIR. This name is then associated with Trigger for Start in the mediashow.

The example uses the Return key as a trigger in addition to the GPIO inputs so you can see the effect without using the GPIO pins. If you wish to disable this then set Disable Controls to yes.

BEWARE: Do not use the GPIO pins until you have read Section 9.4

- 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. Pressing Escape from the menu returns to the mediashow.

- pp_presentation
  The mediashow is configured as a presentation. Use the Return or Down Cursor keys to start the presentation and the Up and Down cursor keys to traverse through it. Escape stops a running video and waits.

- pp_liveshow
  While the Liveshow is running place some video files, audio files, or images into the directory /pp_home/pp_live_tracks and watch them miraculously appear.

- pp_radiobuttonshow
  A Radiobuttonshow controlled either from the keyboard or GPIO pins.

  Links defined in the Radiobutttonshow profile allow for both keyboard keys and buttons to be used to select a track. In addition a gpio.cfg file is included in the profile to bind four GPIO pins to entries and one to the Stop Internal Operation.

- pp_hyperlinkshow
  This demonstrates the features of the hyperlinkshow. The show is controlled by on screen buttons which you can click with a mouse. It should ideally be used with a touchcreen but I do not have one to try. The file screen.cfg defines the buttons and the symbolic names of their input events. The show could be controlled by GPIO pins by binding pins to the symbolic names in a gpio.cfg file.

- pp_audio
  This mediashow demonstrates the capabilities of the audioplayer. You will need speakers connected to hdmi and analogue ports to fully appreciate it. It plays tracks to different speakers; it also demonstrates the use of Run-Time controls to control volume from keyboard or GPIO pins. For this reason it has controls.cfg and gpio.cfg files in the profile,

- pp_concurrent
  This application demonstrates concurrent show playing capabilities. There are two mediashows running simultaneously. The mediashow showing images

must be controlled from the keyboard while the concurrent mediashow containing audio tracks has its controls disabled and is played continuously.

- pp_subshow
  Demonstrates how to use subshows to segment a larger show.

- pp_timeofday
  Triggers two mediashows at different times of day. To make this demonstration work I have replaced the times of day with delays (+xx). You can replace these delays with real local times e.g. 10:30.

- pp_showcontrol
  Demonstrates the use of Show Control to start and stop one concurrent show from another.

- pp_animate
  Demonstrates the use of Animation Control by setting P1-11 to On at the start of a track and to Off at the end. To achieve this it has its own gpio.cfg file. BEWARE: Ensure P1-11 can safely be used as an output before running this.

- pp_shutdown

  A mediashow that demonstrates how to use Show Control to shut down the Pi It displays a message track and then when the Down Cursor is pressed starts a track that shuts down the Pi immediately.

You can use the bash script examples.sh to execute the examples, just type ./examples.sh from a terminal window open in the /pipresents directory.

All examples use a selection of media tracks which are in /home/pi/pp_home/media. The profiles are in /home/pi/pp_home/pp_profiles. In each profile the pp_showlist.json file specifies the look and feel of each show; other files specify the media to use. The files can be viewed in a text editor but it is much better to edit them using Pi Present's own editor described in Section 4.

The 'out of the box' keyboard commands are:

- Up or Down Cursor- move through a menu.
- Up or Down Cursor - Move through a mediashow show. Can also be used to skip to the next or previous track in an automatic mediashow. The movement circles around at the start and end of a show.
- Return – Start a presentation or play the selected menu entry.
- Escape - In a track, Stop and return to the show. In a show, back to the calling show.
- Spacebar - in an image, video, or audio track, toggle pause.
- CTRL-BREAK always exits Pi Presents.

3.1 Terminating Pi Presents and Shutdown the Pi

Out of the box pressing CTRL-BREAK or clicking on the 'Close' icon always exits Pi Presents. Pressing the GPIO pin bound to the shutdown function for 5 seconds will shutdown the Pi, See Section 9.7

There are other ways to exit Pi Presents:

- Some keyboards do not have a BREAK key; it is possible to configure another key to exit Pi Presents by editing the keys.cfg file to bind another key to pp-exit as described in Section 10.2.3.

- Bind a different key, GPIO pin, or click area to the special symbolic name pp-exit as described in Section 10.2

- Use the Show control 'exit' command to exit Pi Presents from within a show, See Section 6.4.3

There are other ways to shut down the Pi from Pi Presents:

- The default operation of pressing a button for 5 seconds is by binding a GPIO pin to the special symbolic name pp-shutdown. See Section 10.2.1. Only a pin can be bound to pp-shutdown as it needs to be asserted for 5 seconds as an anti-tamper measure.

- Bind a key, pin or click area to the special symbolic name pp-shutdownnow as described in Section 10.2. An input event with this symbolic name will cause the Rasperry PI to shutdown immediately.

- Use the Show control 'shutdownnow' command to shut down the Pi immediately from within a show, See Section 6.4.3

4 The Pi Presents Profile Editor

Profiles can be created and edited using the Pi Presents editor.

```
python pp_editor.py
```

When using the editor you will need to supply show references, track references and file names. It is advisable not to create names beginning with pp- or pp_ to avoid clashes with names used by Pi Presents.

4.1 Using the Editor

4.1.1 The Displays

Select one of the example profiles using the Profile>Open menu. Gotcha: On a Linux system remember to open the selected profile directory before pressing OK.

- The top left panel displays the shows in the profile
- The bottom left panel shows the medialists in the profile, click on one of the entries to select it.
- The right-hand panel shows the tracks in the selected medialist.

The selected entries are shown in red. Click the 'Edit Show' button adjacent to the left-hand panel to edit the selected show and the 'Edit' button adjacent to the right-hand panel to edit the selected track.

Edits are saved to disc when OK is pressed, use Cancel if you want to duck out.

You can run pipresents.py and pp_editor.py concurrently from two terminal windows so the effect of any edits can quickly be seen. You will need to restart pipresents.py to see the updates made by the editor. If you are going to edit the examples make a copy of them first by using the File Manager to access /home/pi/pp_home/pp_profiles and to copy a profile directory.

4.1.2 Profile Menu

Profile>Open - displays a directory viewer to select a profile for editing. The profiles displayed are those in the home directory defined in the Options>Edit menu.

Profile>Validate - Validate the Profile. Font and Colour fields are currently not validated and if you edit the .json files with a text editor you are on your own!

Profile>New from Template - lists templates for all types of show. The templates have example tracks so they will run un-edited.

4.1.3 Show Menu

Show>Add - Add a new show. The show type of an existing show cannot be edited.

Show> Edit - Duplicates the Edit Show button

Show>Delete - Deletes the show from the profile, no going back!

4.1.4 Medialist Menu

Medialist>Add - Add a new medialist. The .json extension is added if not included.

Medialist>Delete - Deletes the medialist from the profile, no going back!

4.1.5 Track Menu

Track>Add from File

Adds one or more track entries containing media files to the end of the medialist. These may be images, videos or audio tracks. The editor automatically calculates the file location and track type. If either of these is unacceptable or the media file extension is rejected then use Track>New to add a blank track of the required type.

The list of extensions that are used to select track type is in the first few lines of the pp_definitions.py source. Please raise a bug report if these are inadequate.

Track>Add from Dir

Similar to Add from File, except that all eligible tracks in the directory are added to the medialist.

The list of eligible extensions that is used to select tracks and their types is in the first few lines of the pp_definitions.py source. Please raise a bug report if these are inadequate.

Track>New

Adds a blank track of the selected type to the end of the medialist.

Track>Edit
Duplicates the Edit Button

Track>Delete
Deletes the track from the medialist, no going back!

4.1.6 Tools Menu

Tools>Update All
Update the version of all profiles in the current data home directory, see Section 12 . If you wish to see which profiles are being been updated then run the editor from a terminal window using the -d option.

4.1.7 Options Menu

Options>Edit
Edit the editor options:

- **Pi Presents Home Directory**
  This is an important option which must be set to the directory in which any profile directories and relative media is to be assembled. Out of the box it is set to /home/pi/pp_home. If you choose an alternative setting for this then ensure that the directory /pp_profiles is created inside the chosen /pp_home.

- **Initial Media Directory**
  This is just a helper setting to define where 'Add Track' and 'Add from Dir' start their browsing.

4.1.8 Editor Command Line Options

-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 the editorin the terminal window. Errors and the trace are also recorded in the /pipresents/pp_log.log file.
--forceupdate	Update a profile even if the editor and profile issues are the same (for beta testers only)

4.2 Making Profiles

Application data is kept is a directory called /pp_home. /pp_home must contain a directory /pp_profiles in which profiles are stored.

A profile is a directory which contains the information needed to configure a single application of Pi Presents. It must contain a pp_showlist.json file and one or more medialist (.json) files. The pp_showlist.json file contains a number of sections, a 'start' show section and a number of sections each defining a user generated show

Each section in the pp_showlist.json file defines the look and feel of a show and how they link together. Medialist (.json) files in a profile define the content each show and some per track look and feel information.

For portability reasons all media is best kept under the directory /pp_home, but this is not essential. Media can be stored anywhere under the /pp_home directory using sub-directories if necessary.

The default location of /pp_home is /home/pi; this will need to be overridden by command line options if you are using a USB stick. There is a /pp_home directory in the /pipresents directory, but this should not be used as it contains the templates for the editor and 'out of the box' configuration files.

4.2.1 Making Profiles Portable

I wanted to produce a system where profiles and their media can be moved between different drives without breaking the references between shows and their media tracks. This is necessary if a profile is prepared on a Pi on a SD card and moved to a USB Stick for operational use; also if the profiles are to be prepared on a Windows machine.

To achieve portability media tracks should be stored under the /pp_home directory, in sub-directories if desired. If tracks are copied to this location before the profile is prepared then the Profile Editor will automatically compute the appropriate track reference:

- If the file is below the home directory (…../pp_home), as defined in the Options>Edit menu then the file reference will look like '+/track_to_play.mp4'

- Otherwise the reference will look like '/home/pi/mymedia/track_to_play.mp4'; an absolute reference.

Absolute references do have their uses, for example in specifying internet url's e.g. http://www.mysite.com/ track_to_play.mp4

4.2.2 Using the SD Card for Profiles

This is very useful for developing applications on a Pi. To use the SD card for developing and running profiles

- Create a directory /pp_home in the User Pi's home directory and inside that create a directory called pp_profiles. You can also create a directory in pp_home called say, media, to hold media files.

- In the Editor Options set the Pi Presents Data Home directory to /home/pi/pp_home and the Initial Directory for Media to /home/pi/pp_home/media

- Copy media files to the /media directory

- Using the editor create a new profile call it myprofile and edit it.

- To run the profile type:

  python pipresents.py -p myprofile

  from a terminal window open in the pipresents directory.

When running Pi Presents using sudo, from a desktop shortcut, or from an autostart file it is best to specify the full path of pipresents and also the full path of the data home directory e.g.

sudo python /home/pi/pipresents.py -o /home/pi/pp_home -p myprofile

4.2.3 Using the Editor on a Windows PC

Use on a PC requires a Python system to be installed. I use the free Activestate Python http://www.activestate.com/activepython. There is a good installation guide here http://www.richarddooling.com/index.php/2006/03/14/python-on-xp-7-minutes-to-hello-world/.

To install the editor on a Windows PC download the Zip File from https://github.com/KenT2/pipresents (just click on the Zip button). Extract the contents of the folder pipresents-master to a folder called pipresents.

Either download pipresents_examples and extract the folder to your top level folder (c:\users\ken on my Windows 7 machine) or create a 'pp_home' folder here with a folder 'pp_profiles' inside.

Open pp_editor.py in the Idle editor and run it. Alternatively you can double click on pp_editor.py or create a desktop shortcut to pp_editor.py; you may need associate '.py' files with python.exe.

Use of the editor is the same as on the Pi, it is obviously important that all track references are to media files under pp_home.

Profiles created on a PC can be transferred to the Pi using a USB Stick, alternatively you can ftp them across a network using a PC ftp program such as Filezilla.

4.2.4 Using a USB Stick for Profiles

To run profiles from a USB stick copy /pp_home to the top level of a USB stick from its location either on a Pi or on the PC.

Insert the USB stick into a Pi and run Pi Presents with the -o option set to /media/STICKNAME. STICKNAME is the drive name. Wiindows seems to convert lowercase drive names to upper case.

The editor will probably run on any Linux PC. If you are using such a beast you probably do not need your hand held when transferring profiles.

5 The Components of Pi Presents

5.1 Introduction

Shows and Tracks

The Pi Presents toolkit has two building blocks - shows and tracks.

A show plays tracks; the list of tracks to be played is contained in a medialist which is associated with the show, think of the medialist as an enhanced playlist.

The toolkit currently has five types of show each optimised for a different purpose:

- Mediashow - plays a sequence of tracks, usually automatically, but progress can be manually controlled.
- Hyperlinkshow - Implements the touchscreen functionality that you see in many museums.
- Radiobuttonshow - A simple kiosk show showing a navigation screen - press one of many buttons to play a track.
- Menu - Similar to a Radiobuttonshow but the track selection is by traversing a menu using 'cursor' keys or a single button.
- Liveshow - A Mediashow like show with dynamic remotely sourced content.

The toolkit currently has 'players' for four types of track each playing a different type of media. All 'players' allow the main media to be displayed in a window with an optional background of plain colours, images, and text annotations. They allow animation to be controlled and for other concurrent shows to be started and stopped.

- video - plays videos using omxplayer
- audio - plays audio tracks using mplayer
- image - displays images in many different formats (.jpg etc.)
- message - a quick way to display text.

Subshows and Child Shows

In Pi Presents a show can be a track of another parent show; this is called a subshow. Subshows have a number of uses which include:

- Dividing a long mediashow into segments. Each segment is a show and a parent mediashow contains the segment shows in its medialist.
- A menu or radiobutton show might have mediashows as entries rather than single tracks
- Multi-level menus

Subshows can be nested to any depth, the limit is probably the confusion that it will cause the customer when using them. Sub-shows are tracks so appear in the medialist.

Child shows are subshows used in a special way by mediashows and liveshows. If child shows are enabled (Has Child) the child show can be initiated while running any track in the parent show returning to the next track in the parent when finished.

Concurrent Shows

Pi Presents can run two or more shows concurrently. The concurrent shows appear to run in parallel. This has many uses which include:

- Providing a background audio track to a slideshow.
  Use two mediashows, one with the slideshow and the other with the audio tracks. The latter will need the controls disabled if there is customer interaction with the former.

- Running shows at specific times of day
  Use two or more liveshows or mediashows each triggered at time of day. The liveshow and mediashow can also be stopped at a specific time of day or after a period of time.

- Being really thrifty and doing two completely different tasks with the same Pi, perhaps a slideshow in the Reception and a dummy talking in a museum exhibit triggered by a PIR

Each concurrent show can have subshows.

Pi Presents can be configured so that each concurrent show/subshow stack has its own set of controls.

5.2 Types of Show

Shows have fields which control the sequencing and look of the show. They also have a number of fields which provide default values for all the tracks in the show e.g. Duration, Transition, OMXPlayer Audio; tracks will use the values in the show if their corresponding fields are left blank.

All shows have the following fields:

- Title - Text displayed in the editor and in the entries of a menu show

- Show Reference - A label which allows other shows to reference this show. Can be any alpha-numeric string without spaces.

- Medialist - this is the name of a file which appears in the medialist panel. It must have the extension .json. Every show must have a medialist to define the tracks in the show. The same medialist can be used by more than one show.

5.2.1 Mediashow

Think of a mediashow as a slideshow that can play tracks of different types - videos, audio tracks, images, and even animation control sequences.

Mediashows have a number of fields to define the control of the show, e.g. Trigger for Start, Progress, Repeat and Repeat Interval.

A show can be associated with a mediashow such that the show is accessible from any track in the show. These are termed 'child shows'. The Has Child field tells Pi Presents that a show is associated with the show; the child show itself is referenced in the medialist for the show. Associated with the use of a child show is Hint Text that is displayed only when Has Child is yes.

Mediashows have a number of options in the Repeat field which control how they run:

- interval - the mediashow repeats until it is stopped.
- oneshot - The mediashow waits for Trigger For Start runs once and then waits for Trigger For Start again. It has to be stopped by some other means.
- single-run - the mediashow runs once and then stops by itself. It does not need to be stopped although it can be. Show Control Stop commands sent to a show that is already stopped are ignored. Mainly for use with Show Control

Immediately after starting all a mediashow consider conditions for continuing:

- Mediashows with Trigger For Start = start will run immediately. Mediashows with Trigger For Start = input, or time will proceed when the trigger condition is met.

There is more information on the control of mediashows in Section 6.2.1

Field	Examples	Values
		Essential information
Type	mediashow	Cannot be edited
Title	My First Show	Text describing the show displayed in the editor and on menus
Show Reference	show1	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 mediashow.
		How the show is to be run
Disable Controls	no	yes/no If 'yes' Internal Operations (play, pause, up, down, stop) and Run Time controls are disabled. This is a quick way to inhibit all controls for a concurrent show. For more selective control use the Controls field (See Section 6.4)

Field	Examples	Values
Trigger For Start	start	How the media show proceeds after it is started: • start - continue without a wait • input/input-quiet - wait for an input event. 'input-quiet' suppresses on screen messages. • time/time-quiet - wait for a time of day. 'time-quiet' suppresses the next show message.
Trigger Input	PIR	If Trigger For Start is 'input' or 'input-quiet': A symbolic name that is bound to an input event as described in Section 10.2 If trigger is 'time': A list of times of day h:m:s in 24 hour format separated by spaces. (See Section 6.2.1)
Progress	auto	How the show progresses between tracks: • manual - Pi presents will wait for up/down internal operations between tracks. • auto - tracks play continually, up/down internal operations can be used to skip.
Trigger For Next	continue	If Progress is manual how to trigger the next track • input - wait for an input event • continue - do not wait
Next Input		If Trigger For Next is 'input': A symbolic name that is bound to an input event as described in Section 10.2
Sequence		sequence of tracks: • ordered - played in the order of the medialist • shuffle - play tracks in a random order There is no natural end to a shuffled set of tracks so it will be necessary to use Trigger For End if the show is to be terminated.
Repeat	interval	How the media show is repeated: • oneshot - tracks are sequenced once and then the show waits for a start trigger. • interval - run at intervals • single-run - mediahow is run once and then stops. Reaching the repeat interval will not interrupt playing tracks so all tracks in a repeating show must have non-zero duration or end naturally. To have a blank screen at the end of a mediashow include a 1 second long blank message track.

Field	Examples	Values
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
Trigger For End		How the end of a show can be triggered • none - the end trigger is not operative • time -end the show at a time of day • duration - end the show after a period of time.
End Times	5:00	If Trigger For End = time: • A list of times of day h:m:s in 24 hour format separated by spaces. (See Section 6.2.1) If Trigger For End = duration • a period of time in h:m:s format.
		Tracks of a mediashow can optionally provide an entry point for a show or track. If the play operation is executed the show will be started. The mediashow will continue when the subshow terminates. **If Has Child = yes the hint is displayed**
Has Child	yes	yes/no. If yes the child show facility is enabled. The child show/track is defined in the medialist with the Track Reference of pp-child-show.
Hint Text	Press Play to..	A single line of text
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 From Bottom	100	distance of the text from the bottom of the screen (pixels)
		Show text is overlayed on all tracks in the show other than message tracks
Show Text	Picture of Taj Mahal	If not blank the text to be displayed.
Show Text Font	Helvetica 30 bold	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Show Text Colour	white	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Show Text x Position	100	distance of the start of the text from the left of the screen (pixels)
Show Text y Position	100	distance of the top of the text from the top of the screen (pixels)
Controls		If this show is at the top level of the show/sub-show stack then bindings for Internal and Run-time Operations overriding those in controls.cfg See Section 10.2.4

Field	Examples	Values
		Tracks played in the show need some configuration if not supplied in the individual tracks
Background Image	+/media/image.jpg	If not blank The file name of an image which is used as the background for any tracks in the show.
Background Colour		See http://effbot.org/tkinterbook/tkinter-widget-styling.htm The background colour is set to black when Pi Presents starts. Shows do not change the background colour when they run. The value in this field is used by a track in the show if the value in the track's field is blank.
Transition	cut	cut. Type of transition between tracks. Currently not used.
Duration	10	seconds. How long a track having no natural end is displayed. A value of 0 displays continuously.
Image Window	centred 10 100 10 100 200 700 10 100 200 700 ANTIALIAS	For any image track in the show a viewport in which to show the image: • the value 'centred' • one x,y, pair • two x,y, pairs • two x,y, pairs and a filter value Each field separated by spaces. centred - image is displayed full size centred on the screen. one x,y, coordinate - image is displayed full size with the top left corner at x,y two x,y, coordinates (top left, bottom right) - image is re-sized such that it fits within the rectangle preserving the aspect ratio. Image is displayed centred in the rectangle. An optional filter can be specified, see http://effbot.org/imagingbook/image.htm thumbnail section. If blank NEAREST is used
OMX Audio	hdmi	hdmi/local/<blank>. Sound output channel for any video track played by omxplayer from the show. If blank then the channel is set automatically by the presence of a hdmi monitor.
OMX Volume	0	Volume of audio for the video track (-60 -> 0 dB).
OMX Window	centred 10 100 200 700	For any video track in the show a viewport in which to show the video. • centred - use OMXPlayer default behaviour • Two x,y, pairs (top left, bottom right) - the

Field	Examples	Values
		viewport window. The video is scaled to this size without preserving the aspect ratio.
OMX Other Options		Other options for omxplayer (care required to avoid have a nice day!)
MPlayer Audio	local	hdmi/local. Sound output channel for any audio track played by MPlayer from the show.
Audio Speaker	stereo	left/right/stereo. Speaker for any audio track played by MPlayer in the show.
MPlayer Volume	0	Volume of audio track (-60 -> 0dB) played by MPlayer in the show.
MPlayer Other Options		Other options for MPlayer (care required to avoid rejection!)

5.2.2 Menu

A menu show uses the Title field of tracks and shows in its medialist to automatically generate a text menu on the screen. The Up and Down internal operations can then be used to scroll down the menu and the Play operation to play the track or show.

A menu has a number of fields to define the look and feel of the menu e.g Menu x Position, Menu y Position, Entry Spacing, Entry Font etc.

A menu needs some instructions for the user this is included in the Hint Text field.

If a menu is not used for a period of time it might be useful to return to the parent show, the Timeout field, if non-zero, provides this function.

A menu can have a background image, The Has Background field enables this. The reference to the image file is in the medialist associated with the show.

Field	Examples	Values
		Essential information
Type	menu	Cannot be edited
Title		Text describing the show, displayed in the editor and for menus.
Show Reference	mymenu	A ‘label’ by which the show is referenced by other shows. Any text without spaces
Medialist	themenu.json	Filename of the medialist file containing the tracks for the menu.
		The look and feel of the menu
Disable Controls	no	yes/no If 'yes' Internal Operations (play, pause, up, down, stop) and Run Time controls are disabled. This is a quick way to inhibit all controls for a concurrent show. For more selective control use the Controls field (See Section 6.4)

Field	Examples	Values
		Essential information
Type	menu	Cannot be edited
Title		Text describing the show, displayed in the editor and for menus.
Show Reference	mymenu	A 'label' by which the show is referenced by other shows. Any text without spaces
Medialist	themenu.json	Filename of the medialist file containing the tracks for the menu.
		The look and feel of the menu
Menu x Position	300	x position of the start of the first line of the menu (pixels from left of screen)
Menu y Position	250	y position of the start of the first line of the menu (pixels from top of screen)
Entry Spacing	70	pixels, 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	white	Tkinter colour see: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Selected Entry 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 Image	yes	yes/no. If yes an image is displayed behind the menu. The image is defined in the medialist with the Track Reference of pp-menu-background.
		Menus do not 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 From Bottom	100	distance of the text from the bottom of the screen (pixels)
		Show text is overlayed on all tracks other than message tracks opened from the menu.
Show Text	Picture of Taj Mahal	If not blank the text displayed in all but message tracks played from the menu.
Show Text Font	Helvetica 30 bold	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Show Text Colour	white	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm

Field	Examples	Values
		Essential information
Type	menu	Cannot be edited
Title		Text describing the show, displayed in the editor and for menus.
Show Reference	mymenu	A 'label' by which the show is referenced by other shows. Any text without spaces
Medialist	themenu.json	Filename of the medialist file containing the tracks for the menu.
		The look and feel of the menu
Show Text x Position	100	distance of the start of the text from the left of the screen (pixels)
Show Text y Position	100	distance of the top of the text from the top of the screen (pixels)
Controls		see mediashow for values
		Tracks played from a menu need some configuration if not supplied in the individual tracks
Background Image	+/media/image.jpg	see mediashow for values
Background Colour		see mediashow for values
Transition	cut	see mediashow for values
Duration	10	see mediashow for values
Image Window		see mediashow for values
OMXPlayer audio	hdmi	see mediashow for values
OMXPlayer volume	0	see mediashow for values
OMXPlayer window		see mediashow for values
Other OMX Options	-t 1	see mediashow for values
MPlayer audio	local	see mediashow for values
MPlayer speaker	stereo	see mediashow for values
MPlayer volume	0	see mediashow for values
Other MPlayer options		see mediashow for values

5.2.3 Liveshow

A Liveshow is a sequence of one or more image, video and audio tracks. The content is dynamically supplied.

Liveshows repeatedly display tracks in alphabetical sequence of filenames.

A show or track can be associated with a liveshow such that the show is accessible from any track in the show. These are termed child shows. The Has Child field tells Pi Presents that a child show is associated with the show; the show itself is

referenced in the medialist for the liveshow. Associated with the use of a child show is Hint Text, this is displayed only when Has Child = yes.

A medialist associated with the liveshow is used to specify the child show and anonymous tracks in it will be ignored. A medialist must be associated with a Liveshow even if there is no child show.

Liveshows continue until they are stopped. Immediately after starting liveshows consider conditions for continuing:

- Liveshows with Trigger For Start = start will run immediately. Liveshows with Trigger For Start = time will proceed when the trigger condition is met. Liveshow triggering is explained in more detail in Section 6.2.1

Field	Examples	Values
		Essential information
Type	liveshow	Cannot be edited
Title	My Live Show	Text describing the show displayed in the editor and menu
Show Reference	myliveshow	A 'label' by which the show is referenced by other shows. Any text without spaces
Disable Controls	no	yes/no If 'yes' Internal Operations (play, pause, up, down, stop) and Run Time controls are disabled. This is a quick way to inhibit all controls for a concurrent show. For more selective control use the Controls field (See Section 6.4)
Trigger For Start	time	start/time/time-quiet How the liveshow proceeds after it is started: • start - continue without a wait • time/time-quiet - wait for a time of day. time-quiet suppresses the next show message.
Start Times	9:30 2:30	A list of times of day h:m:s in 24 hour format separated by spaces. (See section 6.2.1)
Trigger For End	none	none/time Trigger the end of the show • none - continue forever • time - terminate at a time of day
End Times	12:00 17:30 5	If Trigger For End = time: • A list of times of day h:m:s in 24 hour format separated by spaces. (See Section 6.2.1) If Trigger For End is duration • a period of time in h:m:s format.
Medialist	mymedia.json	Filename of the medialist file containing the child show track for the liveshow. Must be present even if there is no child show.

Field	Examples	Values
		Liveshows can optionally provide an entry point for a show. If the play internal operation is executed the show will be started. The liveshow will continue when the menu terminates.
Has Child	yes	yes/no. If yes the child show facility is enabled. A child show is defined in the medialist with the Track Reference of pp-child-show.
		A hint is a line of text displayed near the bottom of the screen when Has Child = 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 From Bottom	100	distance of the text from the bottom of the screen (pixels)
		Show text is overlaid on all images in the show.
Show Text	Picture of Taj Mahal	If not blank the text displayed in all image, video and audio tracks in the show
Show Text Font	Helvetica 30 bold	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Show Text Colour	white	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Show Text x Position	100	distance of the start of the text from the left of the screen (pixels)
Show Text y Position	100	distance of the top of the text from the top of the screen (pixels)
Controls		see mediashow for values
		Tracks played in the show need some configuration.
Background Image	+/media/image.jpg	see mediashow for values
Background Colour		see mediashow for values
Transition	cut	see mediashow for values
Duration	10	see mediashow for values
Image Window		see mediashow for values
OMXPlayer Audio	hdmi	see mediashow for values
OMXPlayer Volume	0	see mediashow for values
OMXPlayer Window		see mediashow for values
Other OMX	-t 1	see mediashow for values

Field	Examples	Values
Options		
MPlayer Audio	local	see mediashow for values
MPlayer Speaker	stereo	see mediashow for values
MPlayer Volume	0	see mediashow for values
Other MPlayer Options		see mediashow for values

5.2.4 Radiobuttonshow

A Radiobuttonshow provides the sort of show facilities that are in many kiosks namely:

> 'Start with an initial display with maybe some text inviting the user to press buttons to initiate a track. Press one of the buttons and play a track that is attached to that button. While playing the track pressing another button starts the attached track. At the end of a track or when Stop is pressed revert to the initial display.'

In Pi Presents the initial display could be an image, message, video, audio track, or show; all of these allow some form of text to be displayed. Playing of other tracks is by means of play commands in the Links field of the e.g.

```
but1 play myimagetrack
but2 play myvideotrack
```

Each command has three fields separated by spaces:

- symbolic name - the symbolic name of the input event that will trigger playing of the show or track.
- command - always play for a Radiobuttonshow. This is not the same as the play Internal Operation.
- track - the Track Reference of the track to be played.

While in the Radiobuttonshow the Internal Operations are not used. However when playing a track or show started by the Radiobuttonshow the Internal Operations may be required.

Field	Examples	Values
		Essential information
Type	radiobuttonshow	Cannot be edited
Title	My Show	Text describing the show displayed in the Editor and menu
Show Reference	myradioshow	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 Radiobuttonshow. All tracks should have a Track Reference.
Links	myname play mytrack	See above

Field	Examples	Values
First Track	myfirsttrack	The Track Reference of the track that will form the initial display for the show.
Disable Controls	no	yes/no If 'yes' Internal Operations (play, pause, up, down, stop) and Run Time controls are disabled. This is a quick way to inhibit all controls for a concurrent show. For more selective control use the Controls field (See Section 6.4)
Timeout	30	seconds. When playing a track if no user input is received or the track does not naturally finish before the timeout then return to the initial track.
		Show text is overlaid on all images in the show.
Show Text	Picture of Taj Mahal	If not blank the text displayed in all image, video and audio tracks in the show
Show Text Font	Helvetica 30 bold	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Show Text Colour	white	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Show Text x Position	100	distance of the start of the text from the left of the screen (pixels)
Show Text y Position	100	distance of the top of the text from the top of the screen (pixels)
Controls		see mediashow for values
		Tracks played in the show need some configuration.
Background Image	+/media/image.jpg	see mediashow for values
Background Colour		see mediashow for values
Transition	cut	see mediashow for values
Duration	10	see mediashow for values
Image Window		see mediashow for values
OMXPlayer audio	hdmi	see mediashow for values
OMXPlayer volume	0	see mediashow for values
OMXPlayer Window		see mediashow for values
Other OMX Options	-t 1	see mediashow for values
MPlayer Audio	local	see mediashow for values
MPlayer Speaker	stereo	see mediashow for values
MPlayer Volume	0	see mediashow for values
Other		see mediashow for values

Field	Examples	Values
MPlayer options		

5.2.5 Hyperlinkshow

A Hyperlinkshow provides the sort of show facilities that are used in touchscreen displays in museums:

> 'Start with an initial page, the home page, with some introductory text and or images and a selection of buttons which allowing the user to move to another page. Each page can have a different selection of buttons to initiate other pages. When in any page other than the home page additional buttons allow the user to go back to the previous page or to return to the home page.

Every page in a Hyperlinkshow is a track. Each has a Links field which contains commands implementing the movement between tracks and back.

e.g.track story1b might contain

> next-name call story1c
> alternative-name call alternative1
> back-name return
> home-name home

This says either go forward to story1c or to alternative1, return to the previous track (which was probably story1b), or return to the home track which probably means going back through story1b and story1a without displaying them.

When executing the call command Pi Presents remembers where it has come from so the return command can go back one and home can go back until it arrives at the home track. Think of nested call and return of subroutines in a programming language.

Each command has three fields separated by spaces:

- symbolic name - the symbolic name of the input event that will trigger the command
- command - call, return, goto, jump, exit, null
- track - the Track Reference of the track or show to be played

Command	Argument	Effect
call	Track Reference	play Track Reference and add it to the path
return	<blank>	return 1 back up the path removing the track from the path, stops at Home Track.
return	number	return n tracks back up the path removing the track from the path, stops at Home Track.
return	<Track Reference>	return to Track Reference removing tracks from the path, goes through Home Track if necessary.

home		return to Home Track removing tracks from the path
jump	<Track Reference>	play Track Reference forgetting the path back to Home Track
goto	<Track Reference>	play Track Reference, forget the path
null		inhibits the Link with the same symbolic name that has been defined in the show.
exit		end the Hyperlinkshow

Commands can be placed in the Links field of the Hyperlinkshow in addition to the Links field of tracks. This is a convenience to save typing because the commands from the show are used in every track and the commands from the track are merged with them. Sometimes this is not desirable and using the null command in the track will cancel the command with the same symbolic name in the show.

The goto command does not remember where it has come from. It is an alternative way to call/return for implementing a back button; every back has to be a goto. It is not a good idea to mix call/return and goto in the same part of a Hyperlinkshow.

If a track, such as a video or a timed image, ends naturally there needs to be a way for Pi Presents to execute a Link command in order to determine what to do next. This is achieved by Pi Presents generating an internal input event with the symbolic name pp-onend; the Links field can then have a command to service it e.g.

```
pp-onend goto mynextpage
```

While in the Hyperlinkshow the Internal Operations are not used, however when playing a track or show started by the Hyperlinkshow the Internal Operations may be required.

When developing a hyperlinkshow application it is sometimes useful to see the path of pages that Pi Presents has remembered. To enable this edit pp_pathmanager.py to remove the # sign from self.debug = True at around line 12.

5.2.5.1 Click Areas

In the pilot application of the Hyperlinkshow that I am developing with a museum we envisage using either 'soft buttons' or a touchscreen to control the Hyperlinkshow, both of these require click areas.

In its screen.cfg file Pi Presents allows the definition of click areas. These are polygonal areas of the screen which are touch or mouse click sensitive. A touch or click will produce an input event identified by a symbolic name.

Click areas can have text, coloured backgrounds, and outlines. The click areas to be displayed on each page is determined by the symbolic names in the list of Links from the merged show and track Links.

A Hyperlinkshow can also be controlled from the keyboard or from buttons. In our pilot we intend to use 'soft buttons' a technique I last used on a computer system for

the Navy where mice where not liked because they ran away in rough weather. We intend to have a row of buttons either at the side or bottom of the panel holding the screen, the legends for the buttons are click areas which will appear on the display in the appropriate position; they will have symbolic names which will not be used.

Field	Examples	Values
		Essential information
Type	radiobuttonshow	Cannot be edited
Title	My Live Show	Text describing the show displayed in the editor and menu.
Show Reference	myliveshow	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 Hyperlinkshow. All tracks should have a Track Reference
Links	name play mytrack	See above
First Track	myfirsttrack	The Track Reference of the track that will form the initial display for the show.
Home Track		The Track Reference of the track that will form the initial display for the show.
Timeout Track		The track displayed when a timeout occurs. Pi Presents does not return directly to the Home track as there may be a need to reset animation or stop concurrent shows. Usually this track will do whatever is required and 'goto' the Home Track after a short time. If the track is and audio track it can have no sound and a zero duration.
Disable Controls	no	yes/no If 'yes' Internal Operations (play, pause, up, down, stop) and Run Time controls are disabled. This is a quick way to inhibit all controls for a concurrent show. For more selective control use the Controls field (See Section 6.4)
Timeout	30	seconds. When playing a track if no user input is received or the track does not naturally finish before the timeout then go to the Timeout Track.
		Show text is overlaid on all images in the show.
Show Text	Picture of Taj Mahal	If not blank the text displayed in all image, video and audio tracks in the show
Show Text Font	Helvetica 30 bold	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Show Text Colour	white	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Show Text x Position	100	distance of the start of the text from the left of the screen (pixels)

Field	Examples	Values
Show Text y Position	100	distance of the top of the text from the top of the screen (pixels)
Controls		see mediashow for values
		Tracks played in the show need some configuration.
Background Image	+/media/image.jpg	see mediashow for values
Background Colour		see mediashow for values
Transition	cut	see mediashow for values
Duration	10	see mediashow for values
Image Window		see mediashow for values
OMXPlayer Audio	hdmi	see mediashow for values
OMXPlayer volume	0	see mediashow for values
OMXPlayer window		see mediashow for values
Other OMX Options	-t 1	see mediashow for values
MPlayer Audio	local	see mediashow for values
MPlayer Speaker	stereo	see mediashow for values
MPlayer Volume	0	see mediashow for values
Other MPlayer options		see mediashow for values

5.2.6 Start Show

The Start show specifies the shows to be run in the Start Shows field. All shows specified in this field will be run when Pi Presents starts. Each show will run concurrently with other shows. Input events will be passed to all concurrent shows.

There are limits to combinations of resources that can be used with concurrent shows. These are described in Section 6.4.4

The Start Show must be present and serves just to initiate the first user defined shows.

Field	Examples	Values
type	start	Must always be start, not editable.
title	First Show	Text describing the show, displayed in the editor.
Show Reference	start	must be 'start'
Start	show1 show2	A list of Show References separated by spaces.

Shows		

5.3 Medialists

Medialists are similar to playlists in a media player in that they define the tracks to be played. How they are played is defined by the show that they are associated with. Each entry in a medialist is a 'track'. Tracks can be of various types:

- image - a still image
- video - a track played by OMXPlayer.
- audio - an audio track played by MPlayer
- message - displays lines of text against a coloured background with optional background image.
- show - shows can be used as tracks allowing nesting to any depth.
- child-show - to specify the child show of a mediashow or liveshow.
- menu-background - to specify the image file to be used as the background for a menu.

A medialist is generally associated with a single show. In Pi Presents they have been kept separate from the remainder of the show specification so that the same medialist can be used in two different shows.

5.4 Tracks

Each type of track has fields describing how to display the track, some of these override the corresponding fields in the parent show.

- image - a still image. Image types are currently those that can be rendered by the Python Imaging Library. Image size is best limited to the 1 megapixel region.

- video - a track played by OMXPlayer. Playable video formats depend on the codec licences purchased from the Foundation.

- audio - an audio track played by MPlayer. Any track type playable by MPlayer should be playable. The audio track type is very flexible as it has extended duration handling features.

- message - displays lines of text against a coloured background with optional background image. Can also used to display a blank screen. It provides a simple slide preparation facility. If you want something better use Powerpoint or similar and export as .jpg displaying as an image.

- show - shows can be used as tracks allowing nesting to any depth.

- child-show - a track used to specify the child show of a mediashow or liveshow.

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

5.4.1 Track File Names

Track file names can be relative or absolute. Relative file names allow profiles to be portable. See Section 4.2.1

- 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.

- Absolute references do have their uses, for example in specifying internet url's e.g. http://www.mysite.com/ track_to_play.mp4

5.4.2 Anonymous and Labelled tracks

Some medialist entries will have labels defined by the key Track Reference. If the Track Reference is blank then the track is included in a Menu or Mediashow. If the label is not blank then the track is used in Radiobuttonshows, Hyperlinkshows or for a special purpose. Currently there are two special purposes:

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

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

The field definitions for each type of medialist track follow in the next section.

5.4.3 Show Track

The tracks in a show can themselves be shows. These are called sub-shows. Show tracks allow sub-shows to be included in a medialist. The Show To Run field specifies the Show Reference of the show.

5.4.4 Image Track

Image tracks are rendered by the Python Imaging Library. (See hardware requirements for recommended image size). Image tracks can be paused with the Pause Internal Operation. Images can appear in a window, can have an image or colour as a background, can be overlaid with text; Track Text is overlaid on a per track basis, Show Text per show.

Field	Example	Values
Type	image	
Title		Displayed on a menu and in the editor
Track Reference		A label for the track - blank if the track is to be included in a menu or mediashow. Non-blank if the track is to be triggered from a symbolic name or is special.
Location	+/media/sarename.gif	The filename of the track which may be blank.
Duration	5	Seconds. If 0 then image is displayed until

Field	Example	Values
		terminated by user input. If blank then the value in the parent show is used.
Transition	cut	cut.
Image Window	centred 10 100 10 100 200 700 10 100 200 700 ANTIALIAS	A viewport in which to show the image. If blank then the value in the parent show is used otherwise: • the value 'centred' • one x,y, pair • two x,y, pairs • two x,y, pairs and a filter value Each field separated by spaces. centred - image is displayed full size centred on the screen. one x,y, coordinate - image is displayed full size with the top left corner at x,y two x,y, coordinates (top left, bottom right) - image is re-sized such that it fits within the rectangle. Image is displayed centred in the rectangle. An optional filter can be specified, see http://effbot.org/imagingbook/image.htm thumbnail section. If blank NEAREST is used.
Background Image	+/media/image.jpg	The file name of an image which is used as the background for the main image. If blank then the value in the parent show is used.
Background Colour		See http://effbot.org/tkinterbook/tkinter-widget-styling.htm If blank then the value in the parent show is used.
Track Text	Picture of Taj Mahal	If not blank the text displayed on the image.
Track Text Font	Helvetica 30 bold	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Track Text Colour	white	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Track Text x Position	100	distance of the left end of the text from the left of the screen (pixels)
Track Text y Position	100	distance of the top of the text from the top of the screen (pixels)
Links	myname play track34	Link commands for Radiobuttonshow and Hyperlinkshow
Show Control at Beginning	start audio1	See Section 6.4
Show Control at End	stop audio1	See Section 6.4
Animation at Beginning	out1 on 1 out1 off 2 out2 on 1 out3 on 6	See Section 8

Field	Example	Values
Clear Animation	no	yes/no See Section 8
Animation at End	out2 off	See Section 8

5.4.5 Video Track

A track played by omxplayer. Pi Presents should play any track that OMXPlayer can play (but see hardware requirements Section 11). Video tracks can be paused with the Pause Internal Operation. Videos can appear in a window, can have an image or colour as a background which can be overlaid with text; Track Text is overlaid on a per track basis, Show Text per show. The video itself cannot be overlaid with text because of OMXPlayer limitations.

Field	Example	Values
Type	video	
Title	The Film	Displayed on a menu and in the editor
Track Reference		A label for the track - blank if the track is included in a menu or mediashow. Non-blank if the track is to be triggered from a symbolic name or is special.
Location	+/myvideos/film.mp4	The filename of the track.
OMXPlayer Audio	local	hdmi/local. If blank then the value in the parent show is used.
OMXPlayer Volume	0	Volume of video track (-60 -> 0 dB). If blank then the value in the parent show is used.
OMXPlayer Window	centred 10 10 500 1000	A viewport in which to show the video. If blank then the values are taken from the parent show. • centred - use OMXPlayer default behaviour • Two x,y, pairs (top left, bottom right) - the viewport window. The video is scaled to this size without preserving the aspect ratio..
Background Colour	red	If not blank the colour of the background. See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm. If blank then the value in the parent show is used.
Background Image	+/images/back.jpg	The filename of an optional image that is displayed in the background, can be blank. The video, Show Text and Track text is displayed above the image. If blank then the value in the parent show is used.
Track Text	Picture of Taj Mahal	If not blank the text to be displayed.
Track Text Font	Helvetica 30 bold	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm

Track Text Colour	white	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Track Text x Position	100	distance of the left end of the text from the left of the screen (pixels)
Track Text y Position	100	distance of the top of the text from the top of the screen (pixels)
Links	myname play track34	Link commands for Radiobuttonshow and Hyperlinkshow
Show Control at Beginning	start audio1	See Section 6.4
Show Control at End	stop audio1	See Section 6.4
Animation at Beginning	out1 on 1 out1 off 2 out2 on 1 out3 on 6	See Section 8
Clear Animation	no	yes/no See Section 8
Animation at End	out2 off	See Section 8

5.4.6 Audio Track

A track played by MPlayer. Pi Presents should play any audio track that MPlayer can play. Sound can be directed to HDMI or analogue and to either of the analogue speakers. Audio tracks can be paused with the Pause Internal Operation

Audio tracks can have an associated display. The display can have a coloured or image background; Track Text is overlaid on a per track basis, Show Text per show.

The audio track has enhanced duration and screen control making it suitable for a 'dummy' track for controlling animation or concurrent shows with zero duration and without playing any media.

Field	Example	Values
Type	audio	
Title	The Music	Displayed on a menu and in the editor
Track Reference		A label for the track - blank if the track is included in a menu or mediashow. Non-blank if the track is to be triggered from a symbolic name or is special.
Location	+/tracks/music.mp3	The filename of the track. The location is optional. If blank the timing of the track is taken from the Duration field which must not be blank.
Duration	5	Seconds. If duration is blank then the track ends when the audio file ends or is aborted, or must be terminated by the user if there is no audio track. If zero then the duration is always zero.

		The Location field should be left blank so no audio track is played. Zero duration is aimed at use of the track for show or animation control. If greater than zero the 'track' ends at the stated time. If a file has been specified in Location the playing of the track may be cut short or there may be period of silence after the audio file.
MPlayer Speaker	left	left/right/stereo. If blank then the value in the parent show is used.
MPlayer Audio	local	hdmi/local. If blank then the value in the parent show is used.
MPlayer Volume	0	Volume of audio track (-60 -> 0 dB). If blank then the value in the parent show is used.
Clear Screen	no	yes/no The screen is normally not cleared at the start of an audio track as this would clear any concurrent displays from other shows. If you are using the audio track as part of a main show you may want to clear the screen of previous content.
Background Colour	red	The colour of the background. See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm.). If blank then the value in the parent show is used.
Background Image	+/images/back.jpg	The filename of an optional image that is displayed in the background. The video, Show Text and Track text is displayed above the image. If blank then the value in the parent show is used.
Track Text	Picture of Taj Mahal	If not blank the text to be displayed.
Track Text Font	Helvetica 30 bold	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Track Text Colour	white	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Track Text x Position	100	distance of the left end of the text from the left of the screen (pixels)
Track Text y Position	100	distance of the top of the text from the top of the screen (pixels)
Links	myname play track34	Hyperlinks for Radiobuttonshow and Hyperlinkshow
Show Control at Beginning	start audio1	See Section 6.4
Show Control at End	stop audio1	See Section 6.4
Animation at Beginning	out1 on 1 out1 off 2 out2 on 1 out3 on 6	See Section 8
Clear Animation	no	yes/no See Section 8

Animation at End	out2 off	See Section 8

5.4.7 Message Track

Message tracks display text against a coloured background or optional image. They do not need a media file to be specified as the text is contained in the Message Text field.

Field	Example	Values
Type	message	
Title	A Message	Displayed on a menu and in the editor
Track Reference		A label for the track - blank if the track is included in a menu or mediashow. Non-blank if the track is to be triggered from a symbolic name or is special.
Message Text	Welcome	The text to be displayed.
Duration	5	Seconds. If 0 then message is displayed until terminated by user input. If blank then the value in the parent show is used.
Text Font	Helvetica 30 bold	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Text Colour	white	See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm
Justification	left	left/center/right. Text justification.
Message x Position	100	If blank then the message is centred in the screen. If non-blank the distance of the left end of the text from the left of the screen (pixels)
Message y Position	500	If Message x Position is specified then distance of the top of the text from the top of the screen (pixels)
Background Colour	red	The colour of the background. See: http://effbot.org/tkinterbook/tkinter-widget-styling.htm If blank then the value in the parent show is used.
Background Image	+/images/back.jpg	The filename of an optional image that is displayed instead of a plain background. The message is displayed on top of the image. If blank then the value in the parent show is used.
Links	myname play track34	Link commands for the Radiobuttonshow and Hyperlinkshow
Show Control at Beginning	start audio1	See Section 6.4
Show Control at End	stop audio1	See Section 6.4
Animation at Beginning	out1 on 1 out1 off 2	See Section 8

	out2 on 1 out3 on 6	
Clear Animation	no	yes/no See Section 8
Animation at End	out2 off	See Section 8

5.4.8 Show Track

A show can be a track in another show. Uses might be:

- A mediashow made up of smaller mediashows
- A menu of mediashows, particularly presentations
- Menus with sub-menus
- A Liveshow run from a Mediashow which provides an initial screen.

Field	Example	Values
Type	show	
Title	My Other Show	Displayed on a menu and in the editor
Track Reference		A label for the track - blank if the track is included in a menu or mediashow. Non-blank if the track is to be triggered from a symbolic name or is special.
Show to Run	myothershow	Show Reference of the show to be run

5.4.9 Child Show Track

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

Field	Example	Values
Type	show	
Title	Child Show	Displayed on a menu and in the editor
Track Reference	pp-child-show	The label indicates the track contains a reference to a child show. The track will not be included in the menu or mediashow.
Show to Run	mychildshow	Reference to the show to be run

5.4.10 Menu Background Track

Field	Example	Values
Type	menu-background	
Title	Menu Background	Displayed in the editor
Track Reference	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.

Location	+/media/sarename.gif	The filename of the track.

6 Controlling Shows

6.1 Introduction

Show control in Pi Presents is very flexible to allow advanced applications to be built. Fortunately for most uses the out of the box configuration will suffice.

There are a number of control techniques for shows and their tracks, each having a specific use:

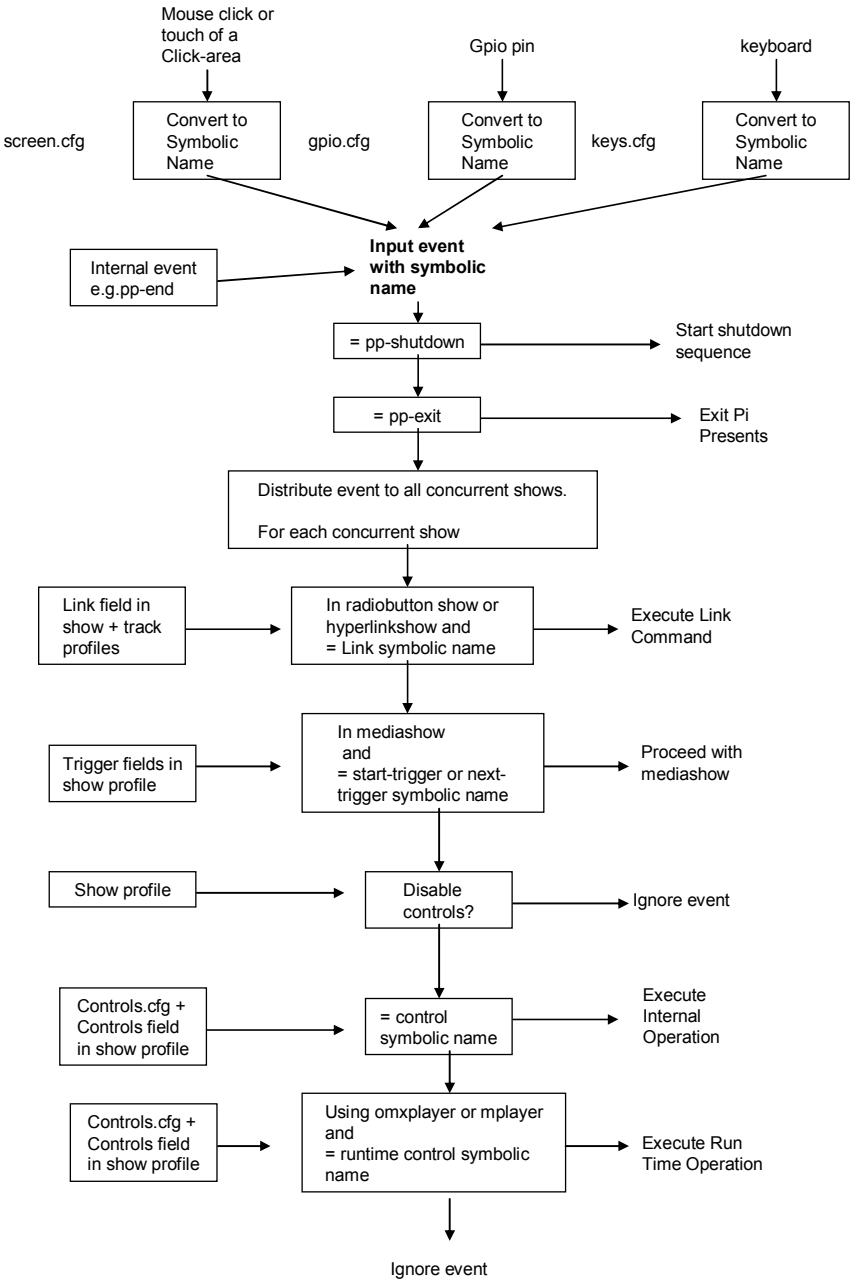
- Internal Operations - play, pause, stop, up and down Internal Operations are used to control mediashows, menus and liveshows
- Triggers - Triggers are used by mediashows as an alternative to Internal Operations. They are primarily used with GPIO inputs.
- Links - Links are used by Radiobuttonshows and Hyperlinkshows to move between tracks.
- Run-time Controls - OMXPlayer and MPlayer allow control of some aspects of their operation while a track is running.

Every control in Pi Presents; Internal Operation, Trigger, Link or Run-time control; is associated with a symbolic name. For example, out of the box, the up Internal Operation has been bound to the symbolic name pp-up in the file controls.cfg. Also you might set up a command in a Radiobuttonshow to play a track by inserting

```
myname play mytrack
```

in the Links field of a track. myname and pp-up are symbolic names.

Real world events such as a GPIO edge detection, a key press, or a mouse click are converted by Pi Presents into an 'input event' which is identified by a symbolic name. Thes symbolic name in the input event is compared with the symbolic names of the controls to determine the action to take. This comparison uses the procedure in the Figure below.



Processing Input Events

Remember that a show can have sub-shows and tracks; input events are generally passed down so they affect the currently lowest level component.

When editing configuration (.cfg) files you will need to supply symbolic names. It is advisable not to create names beginning with pp- to avoid clashes with names used by Pi Presents.

6.2 Controlling Menus, Mediashows and Liveshows

Control of menus, mediashows and liveshows uses a set of 'Internal Operations'. Out of the box these operations are bound to a set of symbolic names and these in turn are bound to an arbitrary set of keyboard keys and GPIO inputs:

Internal Operation	**'Out of the box'** For GPIO Pin see Section 9.4			**Effect**
	Symbolic Name	**Key**	**GPIO Pin.**	
up	pp-up	Cursor Up		Previous track in mediashow or up for menu
down	pp-down	Cursor Down		Next track in show or down for menu
play	pp-play	Return		Start an entry in a menu, or start a child show.
pause	pp-pause	Spacebar		toggle pause for tracks that support pause.
stop	pp-stop	Escape		Stop playing a track or return to a parent show.

The 'out of the box' key for each operation is set in keys.cfg and the GPIO pin in gpio.cfg. The bindings can be modified as described in Section 10.2. Binding the Internal Operation to the symbolic name is in controls.cfg. It is unlikely you will want to modify these but you may wish to override them for an individual show as described below.

Remember that if more than one show is running concurrently the input event is passed to and potentially executed by each concurrent show. This may be undesirable, for example if a mediashow of audio tracks is running as a background to a manually advanced slideshow, we do not want the up and down operations to change audio track, so Disable Controls is used with the audio show.

If the Disable Controls = yes then the show must run automatically or be controlled by triggers as described in the next section.

For more advanced situations individual Operations can be inhibited or bound to alternative symbolic names for a specific show by reconfiguring the control using the Controls field of a show. This is described in Section 10.2.4.

6.2.1 Triggering Mediashows and Liveshows

Mediashows have triggering facilities; liveshows have a subset of these and references to mediashow in the paragraphs below include liveshow where appropriate. The facilities are designed for use when the show is played without user interaction; however if Internal Operations are not disabled they can also be used to control the show (with care!)

In theory triggers can be applied to sub-shows and child shows, I have not tested these extensively.

6.2.1.1 Trigger for Start

Immediately after a mediashow is run it uses the Trigger For Start field to decide what to do next, also after each track Trigger For Next can be employed to control movement to the next track:

Trigger For Start can take the following values:

- start
  The mediashow continues to the first track.

- input/input-quiet
  The mediashow waits for an input event before running the first track of the show. The symbolic name of the input event must be included in the Trigger Input field.

  When an input event, such as a GPIO line changing state, is received Pi Presents generates and input event which is identified by a symbolic name. Symbolic names are assigned to GPIO pins in the gpio.cfg file (see Section 10.2.1). Out of the box P1-11 is configured as an input and assigned to the symbolic name PIR to provide continuity with the previous version of Pi Presents. So if PIR is specified in the Trigger Input field the show will start when P1-11 goes to zero.

  There is an unexpected feature in Pi Presents; a waiting show is also triggered by the Play and Down operations. If you want to use just these choose input or input-quiet and leave the Trigger Input field blank.

- time/time-quiet

  To be able to use time of day either keep the Pi connected to the internet or invest in a hardware real time clock add-on.

  The mediashow waits for a time of day before proceeding. A list of times of day must be included in the Trigger Input field.

  Each time of day can specify hour, minute and optionally second as a 24 hour clock and should be separated by a space. Entries may be in any order. e.g.

  21:2 21:03 9:30 9:30:30 0:0 23:59:59

  Pi Presents will proceed with the show when the local time reaches one of the times specified in the list. If a show is being played when a time in the list is reached the show will not be interrupted or the need for a show to proceed remembered.

  If there are no shows left in the list for today then the first show in the list will be scheduled and started tomorrow.

  If the option is time and not time-quiet an admin message is displayed while waiting for the show to commence. It is possible to show the time of the next show in the admin messages, see Section 10.3. There is an alternative message for shows starting tomorrow.

  Pi Presents may not cope with changes to daylight saving time or any other non-linear changes in local time.

There is an alternative way to specify the time of day. If +10 +40 is entered instead of, say, 09:30 then the show will start in 10 seconds time then in 40 seconds time from when the mediashow was started. This is aimed at testing only.

6.2.1.2 Trigger for End

Some mediashows that repeat, or whose Sequence = shuffle, and all liveshows will not end naturally; they require their end to be forced.

This can be useful in some circumstances, for example:

- Running a liveshow throughout the day and then a 'we are closing' mediashow at a particular time of day. Run the two shows concurrently and terminate the liveshow with a time based Trigger For End and use a time based Trigger for Start to continue the 'we are closing' mediashow a little later.

- Terminating a repeating mediashow or a mediashow that has Sequence=shuffle. The latter cannot be naturally terminated even if it is a one-shot show because, where is the end of a random sequence? In this case the Duration option might be preferred over the time of day.

Trigger for End can take the following values:

- none
  The end trigger is not operative.

- time
  To be able to use time of day either keep the Pi connected to the internet or invest in a hardware real time clock add-on.

  The mediashow continues until the time of day for ending. A list of times of day must be included in the Trigger End Times field.

  Each time of day can specify hour, minute and optionally second as a 24 hour clock and should be separated by a space. Entries may be in any order. e.g.

  21:2 21:03 9:30 9:30:30 0:0 23:59:59

  Pi Presents will terminate the show when the local time reaches the appropriate time specified in the list.

  The system will allow shows to proceed over midnight so a show starting at 23:55 can be terminated at 00:05 the next day.

  Pi Presents may not cope with changes to daylight saving time or any other non-linear changes in local time.

  There is an alternative way to specify the time of day. If +20 + 50 is entered instead of, say, 09:30 then the show will end in 20 seconds time then in 50

seconds time from when the mediashow was started. This is aimed at testing only.

- duration
  The mediashow is terminated after a specified period of time. The time period should be entered in the Trigger End Time field as hour, minute, second.

  The format is h:m:s hour and minute are optional. e.g

  5 - 5 seconds
  6:0 - 6 minutes
  10:0:0 - 10 hours

6.2.1.3 Trigger for Next

The Trigger For Next field allows individual tracks in a mediashow to be triggered by an input event. For this to function correctly Progress should be set to manual:

- none
  The trigger is not operative.

- input
  The show moves to the next track when it is triggered by the input event having the symbolic name specified in Next Input.

6.3 Controlling RadiobuttonShows and Hyperlinkshows

These shows are controlled by commands in their Links field and in the Links field of their tracks. This is described in Section 5.2.5 and 5.2.4. For example the command in the LInks field of a Radiobuttonshow

myname play mytrack

will play a track with the track reference mytrack when an input event with the symbolic name myname is received. Links are not disabled by Disable Controls

Radiobuttonshows and Hyperlinkshows do not use Internal Operations, however any tracks and shows run by them might.

6.4 Controlling Concurrent Shows

6.4.1 Introduction

Pi Presents can run two or more shows concurrently. The concurrent shows appear to run in parallel and Input Events are passed to all concurrent shows. This is essential but if not managed carefully can lead to some undesirable effects. Most of the common situations can however be addressed by Disable Controls

- Providing a background audio track to a slideshow.

Use two mediashows, one with a manually controlled slideshow and the other with the audio tracks. The latter will need the controls disabled using Disable Controls if there is any customer interaction with the former.

- Running shows at specific times of day
  Use two or more liveshows or mediashows each triggered at time of day such that they do not overlap. The liveshow and mediashow can also be stopped at a specific time of day or after a period of time.

- Being really thrifty and doing two completely different tasks with the same Pi, perhaps a slideshow in the Reception with child show facilities, and a dummy talking in a museum exhibit triggered by a PIR using a single shot mediashow. In this case disable controls for the exhibit and rely on the Trigger For Start field which is not affected by Disable Controls.

More complicated scenarios can be addressed by overriding the bindings of the Controls for a specific show. The default bindings of symbolic name to Internal Operation are in controls.cfg. Using the Controls field of a show an Internal Operation can be bound to a different symbolic name for the show and hence be triggered by a different input. Alternatively the override can disable individual operations for that show.

There is more discussion of this in Section 10.2.4

6.4.2 Starting and Stopping Shows

Concurrent shows run in parallel. There are two methods to start and stop concurrent shows:

- Start one or more shows by including their show references in the Start Shows field of the Start Show. All shows specified in the Start Shows field of the Start Show will be run when Pi Presents starts.

- Use a Show Control command in Show Control field of a track to start or stop a show e.g.

  myshow start
  myothershow stop

Shows started in the Start Show can be stopped by a Show Control command.

Only one instance of a show can be running at a time. Attempts to start a show that is already running will be ignored.

6.4.3 Exiting Pi Presents and Shutdown of the Pi

In addition to starting and stopping shows Show Control commands can be used to exit Pi Presents or to shut down the Raspberry Pi e.g.

pipresents exit
gobbledegook shutdownnow

Any word can be used instead of the word pipresents

6.4.4 Limitations on Concurrency

Concurrent shows have some limitations due to limits built into Linux, MPlayer and OMXPlayer:

- Two shows cannot in general use the screen at the same time.
- If audio tracks to be played with MPlayer might overlap then they should all use the same output, either HDMI or Analogue. If not, switching will occur mid-track and I find the Pi crashes. If you want to play two audio tracks that might overlap you can use Videoplayer to play one of them or just direct each of the tracks to left and right outputs.
- HD videos may stutter while an audio track is being played.
- Animation outputs are delayed if they are coincident with the start of an audio track or image. The output event will not be missed but will be delayed.

7 Remote Operation

Pi Presents was not intended for remote operation however by popular demand I have included a facility where a 'Liveshow' can play tracks dynamically ftp'ed from another computer.

The New> Liveshow template is a working Liveshow. The tracks to be played should be placed in /pp_home/pp_live_tracks.

Tracks can be ftp'ed into the Pi using Filezilla or some such. Using the -l command line option of Pi Presents it is possible to have a second location containing live tracks, one in the data home as above and another in another directory specified by the -l option. The directory could be on a remote fileserver (I have not tried this, Samba required?).

Alternatively the complete profile for a show could possibly be held on a remote fileserver.

You may need to set your Pi to use static IP and take network security measures if you use this, as a minimum change the password for user Pi.

8 Animation Control

All types of track have facilities to control animation. Using commands included in the 'Animation at Beginning' and 'Animation at End' fields a GPIO output can be turned on or off synchronised with the start or end of tracks, with optional delay.

An example of animation commands is shown below.

animate at beginning	out1 on 1 out1 off 2 out2 on 1 out3 on 6
Clear Animation	no
Animation at End	out2 off

A command has three fields separated by spaces and terminated with a newline:

- Symbolic Name - The name of the output as defined in gpio.cfg (Section 10.2.1). There can be different gpio.cfg files for each profile.
- State to go to - on or off
- Delay - seconds as a positive integer or 0. If the field is omitted 0 is assumed.

The state of GPIO outputs before Pi Present starts is not well documented and seems to vary between pins and versions of the Pi so be careful if there is anything of concern attached to an output.

All outputs are set to the off state when Pi Presents starts. Commands in Animation at Beginning are executed at the beginning of a track and Animation at End at the end of a track. When the commands are executed the required changes to the GPIO are put in a queue for firing at the appropriate time. The time is not affected by pausing a track.

Animation commands in the queue are not forgotten at the end of a track so animation can be extended over a show. A side effect of this is that it is possible for a GPIO change to happen after a track or show has finished. If you want to avoid this and ensure outputs are in a defined state at the end of a track set Animate Clear to yes. If yes then, before Animation at End is executed, the queue will be cleared of those events that were commanded by the track but not fired.

9 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.

9.1 Command Line Options

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

Options	
-p --profile <arg>	Name of the profile to be used. e.g. pp_mediashow If this is not specified then it defaults to pp_profile.
-g --gpio	Enable the Pi's GPIO system. To use this Pi Presents must be run as root:

	`sudo python pipresents.py -g`
-b --noblank	Disable screen blanking. For this to function x11-server-utils must have been installed.
-f --fullscreen	Run Pi Presents in full screen mode.
-o --home <arg>	Location of the Pi Presents 'data home' directory e.g. /media/USBSTICK or /home/pi/my_data If this option is not specified it defaults to the users home directory. **NOTE**: If sudo is used then the user's home directory is /root so the option must be specified with a full path. Furthermore, to be safe it is best to specify the full path whenever executing pipresent.py from anywhere other than the pipresents directory.
-d --debug	Run time errors give alerts and are reported to the terminal window if open. Turning on debugging will provide an additional log of the operation of Pi Presents in the terminal window. Errors and the log are also recorded in the /pipresents/pp_log.log file.
-v --validate	Validate the profile when Pi Presents is started.
-l --liveshow <arg>	Liveshow tracks are always played from the directory pp_live_tracks in the data home specified in the -o option defaulting to /home/pi/pp_home/pp_live_tracks If the -l option contains a directory path this is used as a second source for liveshow tracks. e.g. /home/pi/mylivetracks

9.2 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.

9.3 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 the users home directory. **NOTE**: If sudo is used then the user's home directory is /root so the option must be specified. .

Furthermore, to be safe it is best to specify the full path whenever executing pipresent.py from anywhere other than the pipresents directory.

USB sticks can be used to hold profiles and 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. If Pi Presents is started at power on it takes up to 10 seconds for the drive to be mounted; Pi Presents allows for this.

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

9.4 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.

Out of the box the following pins of the GPIO connector 1 are bound to the following symbolic names in gpio.cfg. The binding can be modified as described in Section 10.2.1

Bound Pin	Symbolic Name	Bound Key	Function
Internal Operations			
P1-15	pp-down	Cursor Down	Next in show or down for menu
P1-16	pp-up	Cursor Up	Previous in show or up for menu
P1-18	pp-play	Return	Start a mediashow or play the standard child show.
P1-22	pp-pause	Spacebar	toggle pause for videos.
P1-7	pp-stop	Escape	Stop playing a track or return to the higher level show.
User Binding			
P1-11	PIR	-	Not an internal operation
Special Bindings			
P1-12	pp-shutdown	-	Press for 5 seconds to exit Pi Presents and shut down the Pi.
-	-	CTRL-BREAK	Abort Pi Presents. Focus must be on a Pi Presents window.

Out of the box the GPIO ports are configured as edge triggered inputs with internal pull-up resistors and for the following device characteristics:

- Push buttons should be mechanical, push to make (normally open), and be connected between the GPIO pin and 0 volts.

- PIR's should have Normally Closed relay contacts and be connected between the GPIO pin and 0 volts.

Inputs can be changed from normally open to normally closed and vice versa by changing the edge which is used for triggering as described in Section 10.2.1

A 330 ohm resistor is series with the button or PIR is recommended to protect the Pi should the inputs inadvertently be used as outputs.

GPIO Pin ----- 330R ----- contact ---- 0 volts

Be very careful not to connect a GPIO pin to the +5volt pin; it is likely to fry your Pi

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 gpio.cfg (See Section 10.2.1).

9.5 Disable Screen Blanking

To disable screen blanking you must first install xset which is part of the x server utilities package, this should have been done if you followed the install instructions..

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

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

9.6 Start Pi Presents when Power is applied to the Pi

This will function 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 of pipresents and of the home directory:

  e.g. `sudo python /home/pi/pipresents/pipresents.py -o /home/pi/pp_home -p myprofile`

- Make the autostart file executable by using the properties menu in the File Manager to alter the permissions on the file.

9.7 Shutdown the Raspberry Pi from the GPIO

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

The PI can also be shut down by using a Show Control command as described in Section 6.4.3

10 Configuring Pi Presents

10.1 Location of Configuration Files

The out of the box configuration of Pi Presents is defined in a set of files in /pipresents/pp_home:

- gpio.cfg - gpio inputs and animation outputs
- keys.cfg - keyboard
- screen.cfg - click areas on the monitor
- controls.cfg - internal operations and run-time operations
- resources.cfg - administrative messages

The .cfg files are all text files which can be edited by Leafpad. Do not edit the files inside /pipresents/pp_home; if you wish to change their content then copy the file to either:

- Inside a profile e.g. inside /home/pi/pp_home/myprofile
- The data home directory, e.g. /home/pi/pp_home

Pi Presents looks for a .cfg file first in a profile, then in the data home directory, then in /pipresents/pp_home.

If changing these files beware:

- The file in /pipresents/pp_home may be overwritten by updates to Pi Presents.

- There is little checking of the content of these files by Pi Presents. If you modify the file run pipresents.py from a terminal window first so that any Python error messages can be displayed.

10.2 Configuring Inputs and Outputs

Section 6.1 describes the input system of Pi Presents and how external events are converted to input events. Most of the responses are configurable using the files described in this section.

When editing configuration files you will need to supply symbolic names. It is advisable not to create names beginning with pp- to avoid clashes with names used by Pi Presents.

10.2.1 Configuring GPIO Pins

BEWARE: Accidentally using a pin as an output with the output shorted to ground or +3.3 volts might fry your Pi, use a series resistor on every input and output for protection.

The configuration of the GPIO used by Pi Presents is defined in the file gpio.cfg. The file in /pipresents/pp_home/gpio.cfg configures Pi Presents for the buttons and the PIR described in this manual and used by the examples.

gpio.cfg maps physical Pi GPIO input and output pins to the symbolic names of inputs and outputs used by Pi Presents. It also configures the input pins.

A section for every pin must be present in the file. A pin with direction=none is ignored .

Inputs

Each pin can generate an event having the specified symbolic name in any of four ways:

- rising edge - An event with the symbolic name specified in 'rising-name' is generated when the input changes from 0 to 1 (0 volts to 3.3 volts)
- falling edge - An event with the symbolic name specified in 'falling-name' is generated when the input changes from 1 to 0 (3.3 volts to 0 volts)
- one state - An event with the symbolic name specified in 'one-name' is generated at 'repeat' intervals while the input state is '1' (3.3 volts). The first event happens after 'repeat' interval. If you want the input to respond immediately set the rising edge event to the same symbolic name.
- zero state - An event with the symbolic name specified in 'zero-name' is generated at 'repeat' intervals while the input state is '0' (0 volts). The first event happens after 'repeat' interval. If you want the input to respond immediately set the falling edge event to the same symbolic name.

If you do not want the event to be generated leave the symbolic name blank

For the purposes of this manual and the examples gpio.cfg is set up to allow normally open push buttons connected to ground (0 volts) and a PIR with a normally closed contact connected to ground.

Outputs

The logical 'ON' state produces +3.3 volts. The logical 'OFF' state produces 0 volts

Pi Presents initialises GPIO outputs to 0 volts so it is best to design relays etc. for positive logic.

10.2.2 Configuring Click Areas

The file screen.cfg defines the areas of the screen that will become mouse click or touch sensitive.

The file consists of a number of sections each with a unique name. The name can be anything but must be unique within the file.

All fields in each section must be present. The fields of each section are used as follows:

- name - The symbolic name of the click area. Each command in the Links field of a track has a symbolic name. When the track in a hyperlink show is played the click-areas in the Links field are displayed on the screen and when an area is clicked the symbolic name will identify an input event.

- points - this is a set of x y pairs that defines the points of a polygon bounding the area. The polygon is automatically closed so a quadrilateral will have 4 (not 5) x,y pairs. There must be at least three pairs of points. For details of this and other attributes see http://effbot.org/tkinterbook/canvas.htm create_polygon

- fill-colour, outline-colour

  Specifies the look of the polygon. Use a blank field for transparent

- text, text-font, text-colour

  If text is not blank then the text is written centred in the polygon.

10.2.3 Configuring Keyboard Keys

Keyboard keys are bound to symbolic names in the file keys.cfg. The file has a number of lines with the format

    condition = symbolicname

The conditions are defined in effbot.org/tkinterbook/tkinter-events-and-bindings.htm in the <Return>, a, and <Shift-Up> sections. The conditions and the symbolic names are case sensitive.

In addition to these bindings the printing characters on the keyboard, (the ones obeying the <Key> condition in the reference), are automatically bound to the symbolic name pp-key-x

    e.g the 'a' key produces pp-key-a

Automatic binding of a printing key can be overridden by a line such as a = pp-pause.

Note: <any control> = pp-shutdown will not work as pp-shutdown needs to be asserted for 5 seconds. It is useful for gpio only. However you can shutdown the Pi

from Pi Presents using a show control command or by binding a key to the special symbolic name pp-shutdownnow.

10.2.4 Configuring Internal and Run-Time Operations

Internal Operations

The file controls.cfg binds symbolic names to the Internal Operations of Pi Presents. This is different to the other configuration files which bind external event detections to symbolic names.

It is unlikely that you will want to modify this file, more likely you will want to use the Controls field in a Show to change the binding for a specific show. The symbolic name that is bound to an internal operation can be changed for a top level show and all its child/subshows by specifying a new symbolic name for the function in the Controls field of the top level show.

The operation can also be disabled for the show stack by binding the symbolic name to null. e.g.

In the Controls field of a show:

```
mypause pause
pp-stop null
```

will make an input event matching mypause toggle the pause for tracks of this show instead of pp-pause and inhibit any action for pp-stop when received by this show, its sub-shows, or tracks.

Run-Time Controls

The file controls.cfg may be used to add run-time control operations. OMXPlayer and MPlayer have a number of run-time controls which are specified here and here. The operations must be a single character from the list of operations and be preceded by omx- or mplay- respectively. Multi-character sequences are not currently supported.

For example:

```
videovolup = omx-+
videovoldown = omx--
```

adds volume control to omxplayer. videovolup can be bound to a key, gpio pin etc. Keyboard keys naturally have repeats; for GPIO pins gpio.cfg can enable repeating for buttons.

Run-time operations may also be defined in the Controls field of a show.

10.3 Modifying Admin Messages

A few administrative messages seen by customers cannot be controlled using profiles.

resources.cfg is a text file which can be edited using Leafpad. It has a section for each show or player which has messages which are displayed to customers. e.g.

```
[mediashow]
# waiting for user to start a show with progress = manual
m01 = To start the show press 'Play'

# waiting for trigger from PIR
m02 =
```

There is a comment (#.....) which describes how the message is used and a number, e.g. m01, which is used to refer to the message in the Python code.

You can edit the text after the = sign to change a message or, if blank, to not display the message.

Make sure to try out your edits by running Pi Presents from the terminal window so that any Python exceptions due to mistakes in edits are displayed.

Two of the messages allow the time of the next show to be displayed. This is achieved by inserting %tt in the message string.

11 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.

The RAM memory split is determined by OMXPlayer and on 256MB machines should be the minimum that OMXPlayer requires so as to allow the maximum RAM memory for the handling of images. OMXPlayer plays some videos using 64MB of RAM; others need 128MB, especially if you want sub-titles.

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 Pi. Larger images, greater than the screen pixel dimensions, will do nothing to improve the picture and will take longer to display even on 512MB machines; I use the brilliant Faststone Photo Resizer http://www.faststone.org/FSResizerDetail.htm to reduce the size of images on a Windows PC.

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.

12 Updating Pi Presents

For safety take a copy of the pipresents folder and any data before doing updates.

Download Pi Presents from github and install it as described in the README.md 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.

As Pi Presents develops new fields will be added to the profile definition and others deleted. To control this, the three elements - Pi Presents, the editor and the profiles - must have the same version. Pi Presents will object if a profile with the wrong version is used. To correct this open the profile in the editor, it will automatically update the version of the profile and its fields. There will be a few residual update tasks that cannot be automated; read the release notes to identify these.

If you have many profiles to update in the same /pp_home then you can use the 'tools>update all' menu option of the editor to update them.

The profiles in pipresents-examples will be kept compatible with the latest version of Pi Presents. Beware, re-installing these might overwrite profiles you have made.

When updating Pi Presents read the release notes. You may need to update the configuration files and carry out a few tasks that cannot be done automatically.

13 Bug Reports and Feature Requests

Please use the Github Issues Tab https://github.com/KenT2/pipresents/issues or the Pi Presents thread http://www.raspberrypi.org/phpBB3/viewtopic.php?f=38&t=29397on the Raspberry Pi forum to report bugs.

I am keen to improve Pi Presents and your input on real world experiences and requirements would be invaluable to me, both minor tweaks to the existing functionality and major improvements.

14 Gotchas and Known Problems

When using sudo, autostart, or running from a desktop shortcut my profile is not found.
In any of these situations it is best to use the full path of pipresents and the data home in commands e.g.

```
sudo python /home/pi/pipresents.py -o /home/pi -p myprofile
```

Pi Presents locks up in full screen, how do I escape.
This is usually caused by Python reporting an exception due to incorrect configuration data.

To avoid this use validation in the editor and try the show, not full screen, and running from a terminal window so you can CTRL-C out if CTRL-BREAK fails. If this fails, try closing the terminal window and if necessary reboot. Pulling the power has potential for corruption, so the ideal solution is to SSH into the Pi from another machine, run top (top -upi) and kill the python process with the k xxxx command. You may need to kill an omxplayer process as well.

My video/audio track does not play with Pi Presents
Try playing it using OMXPlayer or MPlayer from the command line.

I have built a gpio input or output device and it is not working with Pi Presents.

There are two stand alone GPIO test programs in the pipresents directoryr input_test.py and output_test.py so you can test you I/O before blaming Pi Presents!

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 with user root and then create with user Pi or run pipresents.py without sudo.

Pi Presents sometimes crashes when two audio tracks are being played

I have found that if two audio tracks are played with MPlayer and one is set to HDMI and the other to Local then as expected the sound output channel switches. It also causes MPlayer to crash.

Pi Presents crashes when a video and audio track is played simultaneously.

When the audio track played by MPlayer finishes the Raspberry Pi crashes. Need to check whether this is present in the latest firmware.

When using the editor on Windows I get the message "failed to save medialist, trying again"

When using Windows the medialist is intermittently not saved the first time. However if you try again it will probably succeed. Ideas on why this happens are welcome. Code is in pp_medialist.py at around line 266.

Pi Presents crashes after playing videos for a few hours

There seems to be some timing problem in the interface with OMXPlayer or with OMXPlayer itself. If a crash happens its usually best to restart the Raspberry Pi as OMXPlayer seems to be left in a funny state.