pympris Documentation

Release 1.5dev

wistful

February 25, 2014

CHAPTER 1

pympris Package

1.1 pympris Package

pympris is a Python library used to control media players using MPRIS2 interfaces.

Usage:

```
import gobject
import dbus
from dbus.mainloop.glib import DBusGMainLoop

import pympris

dbus_loop = DBusGMainLoop()
bus = dbus.SessionBus(mainloop=dbus_loop)

# get unique ids for all available players
players_ids = list(pympris.available_players())
mp = pympris.MediaPlayer(players_ids[0], bus)

# mp.root implements org.mpris.MediaPlayer2 interface
# mp.player implements org.mpris.MediaPlayer2.Player
# mp.track_list implements org.mpris.MediaPlayer2.TrackList
# mp.playlists implements org.mpris.MediaPlayer2.Playlists

# print player Identity
print mp.root.Identity

if mp.root.CanRaise:
    mp.root.Raise()

if mp.player.CanPlay and mp.player.CanPause:
    mp.player.PlayPause()

mp.player.Volume = mp.player.Volume * 2

if mp.player.CanGoNext:
    mp.player.Next()

tracks = mp.track_list.Tracks
for track_id in tracks:
    print track_id
```

```
if len(tracks) > 1:
    mp.track_list.RemoveTrack(tracks[-1])
    mp.track_list.GoTo(tracks[0])

n = mp.playlists.PlaylistCount
ordering = pympris.PlaylistOrdering.LastPlayDate
playlists = mp.playlists.GetPlaylists(0, n, ordering, reversed=False)
pl_id, pl_name, pl_icon = playlists[-2]
mp.playlists.ActivatePlaylist(pl_id)

# setup signal handlers


def handle_properties_changes(changed_props, invalidated_props):
    for name, value in changed_props.items():
        print('Property %s was change value to %s.' % (name, value))


def seeked(x):
    print("Positin was seeded to %s" % x)


def PlaylistChanged(arg):
    print("PlaylistChanged", arg)


def TrackMetadataChanged(track_id, metadata):
    print("TrackMetadataChanged", track_id, metadata)


def TrackListReplaced(tracks, current_track):
    print("TrackListReplaced", tracks, current_track)


def TrackAdded(metadata, after_track):
    print("TrackAdded", metadata, after_track)


def TrackRemoved(track_id):
    print "TrackRemoved", track_id


mp.player.register_properties_handler(handle_properties_changes)
mp.playlists.register_properties_handler(handle_properties_changes)

mp.player.register_signal_handler('Seeked', seeked)
mp.playlists.register_signal_handler('PlaylistChanged', PlaylistChanged)
mp.track_list.register_signal_handler('TrackMetadataChanged',
                                      TrackMetadataChanged)
mp.track_list.register_signal_handler('TrackListReplaced', TrackListReplaced)
mp.track_list.register_signal_handler('TrackAdded', TrackAdded)
mp.track_list.register_signal_handler('TrackRemoved', TrackRemoved)

loop = gobject.MainLoop()
loop.run()
```

1.2 Base Module

This module provides a *Base* class used as a base class for implementing MPRIS2 interfaces.

class pympris.Base.**Base**(**args*, ***kwds*)
Bases: pympris.common.BaseVersionFix

Base class provides common functionality for other classes which implement MPRIS2 interfaces.

OBJ_PATH = '/org/mpris/MediaPlayer2'

bus = None
Bus object from the functions argument or SessionBus()

get = None
function to receive property's value

iface = None
DBUS interface (uses self.IFACE path to create it)

name = None
objects name from the functions argument

properties = None
DBUS interface to work with object's properties

proxy = None
DBUS proxy object

register_properties_handler(**args*, ***kwds*)
register *handler_function* to receive *signal_name*.

Uses dbus interface IPROPERTIES and objects path self.OBJ_PATH to match 'PropertiesChanged' signal.

Parameters **handler_function** (*function*) – The function to be called.

register_signal_handler(**args*, ***kwds*)
register *handler_function* to receive *signal_name*.

Uses class's dbus interface self.IFACE, objects name self.name and objects path self.OBJ_PATH to match signal.

Parameters

- **signal_name** (*str*) – The signal name; None(default) matches all names.
- **handler_function** (*function*) – The function to be called.

set = None
function to set property's value

1.3 MediaPlayer Module

Module provides a *MediaPlayer* class wich contains instances of all implementations of MPRIS2 interfaces.

Usage:

```
mp = MediaPlayer('org.mpris.MediaPlayer2.rhythmbox')
print(mp.root.Identity)
if mp.root.CanRaise:
    mp.root.Raise()
```

```
if mp.player.CanPause and mp.player.CanPlay:
    mp.player.PlayPause()
if mp.player.CanGoNext:
    mp.player.Next()

print(mp.track_list.Tracks)
print(mp.playlists.PlaylistCount)

if mp.root.CanQuit:
    mp.root.Quit()
```

class pympris.MediaPlayer.**MediaPlayer**(*dbus_name*, *bus=None*, *private=False*)
Bases: object

Class implements all MPRIS2 interfaces.

player = None
Instance of pympris.Player class

playlists = None
Instance of pympris.PlayLists class

root = None
Instance of pympris.Root class

track_list = None
Instance of pympris.TrackList class

1.4 PlayLists Module

Module provides a *PlayLists* class wich implemented MPRIS2 PlayLists interface: http://specifications.freedesktop.org/mpris-spec/latest/Playlists_Interface.html

Class *PlaylistOrdering* uses as an enum for Ordering type.

Usage:: from pympris import PlayLists

pl = PlayLists('org.mpris.MediaPlayer2.rhythmbox') print(pl.PlaylistCount) print(pl.ActivePlaylist)

items = pl.GetPlaylists(0, 100, PlaylistOrdering.Alphabetical, reversed=False) for uri, name, icon_uri in items:

print(uri, name, icon_uri)

class pympris.PlayLists.**PlayLists**(**args*, ***kwds*)
Bases: pympris.Base.Base

Class implements methods and properties to work with MPRIS2 Playlists interface.

ActivatePlaylist(**args*, ***kwds*)
Starts playing the given playlist.

Param str playlist_id: The id of the playlist to activate.

ActivePlaylist
The currently-active playlist.

GetPlaylists(**args*, ***kwds*)
Gets a set of playlists.

Parameters

- **start** (*int*) – The index of the first playlist to be fetched (according to the ordering).

- **max_count** (*int*) – The maximum number of playlists to fetch.
- **order** (*str*) – The ordering that should be used.
- **reversed** (*bool*) – Whether the order should be reversed.

IFACE = 'org.mpris.MediaPlayer2.Playlists'
The D-Bus MediaPlayer2.Playlists interface name

Orderings
The available orderings. At least one must be offered.

PlaylistCount
The number of playlists available.

class pympris.PlayLists.**PlaylistOrdering**
Bases: object

Alphabetical = 'Alphabetical'

CreationDate = 'Created'

LastPlayDate = 'Played'

ModifiedDate = 'Modified'

UserDefined = 'User'

1.5 Player Module

Module provides a *Player* class wich implemented MPRIS2 Player interface: http://specifications.freedesktop.org/mpris-spec/latest/Player_Interface.html

Usage:

```
from pympris import Player

player = Player('org.mpris.MediaPlayer2.vlc')
if player.CanPause:
    player.PlayPause()

player.Volume = player.Volume * 2
player.Play()
if player.CanGoNext:
    player.Next()

if player.CanSeek:
    player.Seek = 15000000
```

class pympris.Player.**Player**(**args*, ***kwds*)
Bases: pympris.Base.Base

Class implements methods and properties to work with MPRIS2 Player interface.

CanControl
Whether the media player may be controlled over this interface.

CanGoNext
Whether the client can call the Next method on this interface and expect the current track to change.

CanGoPrevious
Whether the client can call the Previous method on this interface and expect the current track to change.

CanPause
Whether playback can be paused using Pause or PlayPause.

CanPlay
Whether playback can be started using Play or PlayPause.

CanSeek
Whether the client can control the playback position using Seek and SetPosition. This may be different for different tracks.

IFACE = 'org.mpris.MediaPlayer2.Player'
The D-Bus MediaPlayer2.Player interface name

LoopStatus
The current loop / repeat status

Getter Returns loop status

Setter Sets loop status

Type str

May be: - "None" if the playback will stop

when there are no more tracks to play

- **"Track" if the current track will start again from** the begining once it has finished playing
- "Playlist" if the playback loops through a list of tracks

MaximumRate
The maximum value which the Rate property can take. This value should always be 1.0 or greater.

Metadata
The metadata of the current element.

MinimumRate
The minimum value which the Rate property can take. This value should always be 1.0 or less.

Next(**args*, ***kwds*)
Skips to the next track in the tracklist.

OpenUri(**args*, ***kwds*)
Opens the Uri given as an argument

Parameters **uri** (*str*) – Uri of the track to load.

Its uri scheme should be an element of the org.mpris.MediaPlayer2.SupportedUriSchemes property and the mime-type should match one of the elements of the org.mpris.MediaPlayer2.SupportedMimeTypes.

If the playback is stopped, starts playing If the uri scheme or the mime-type of the uri to open is not supported, this method does nothing and may raise an error.

Pause(**args*, ***kwds*)
Pauses playback.

Play(**args*, ***kwds*)
Starts or resumes playback.

PlayPause(**args*, ***kwds*)
Pauses playback.

PlaybackStatus
The current playback status. May be "Playing", "Paused" or "Stopped".

Position
The current track position in microseconds, between 0 and the 'mpris:length' metadata entry (see Metadata).

Previous(**args*, ***kwds*)
Skips to the previous track in the tracklist.

Rate
The current playback rate.

Getter Returns current playback rate.

Setter Sets playback rate.

Type float

Seek(**args*, ***kwds*)
Seeks forward in the current track

Parameters **offset** (*int*) – The number of microseconds to seek forward. A negative value seeks back.

SetPosition(**args*, ***kwds*)
Sets the current track position in microseconds.

Parameters

- **track_id** (*str*) – The currently playing track's identifier.
- **position** (*int*) – Track position in microseconds. This must be between 0 and <track_length>.

If the Position argument is less than 0, do nothing. If the Position argument is greater than the track length, do nothing. If the CanSeek property is false, this has no effect.

Shuffle
A value of false indicates that playback is progressing linearly through a playlist, while true means playback is progressing through a playlist in some other order.

Getter Returns a value of false indicates that playback.

Setter Sets a value of false indicates that playback.

Type bool

Stop(**args*, ***kwds*)
Stops playback.

Volume
The volume level

Getter Returns a volume level.

Setter Sets a volume level.

Type float

1.6 Root Module

Module provides a *Root* class wich implements MPRIS2 Root interface: http://specifications.freedesktop.org/mpris-spec/latest/Media_Player.html

Usage:

```
from pympris import Root

root = Root('org.mpris.MediaPlayer2.vlc')
print(root.Identity)  # VLC media player

if root.CanRaise:
    root.Raise()

print("Supported Mime Types", root.SupportedMimeTypes)
print("Supported Uri Schemes", root.SupportedUriSchemes)

if root.CanQuit:
    root.Quit()
```

class pympris.Root.**Root**(**args*, ***kwds*)

Bases: pympris.Base.Base

Class implements methods and properties to work with MPRIS2 MediaPlayer2 interface.

CanQuit

If false, calling Quit will have no effect, and may raise a NotSupported error. If true, calling Quit will cause the media application to attempt to quit (although it may still be prevented from quitting by the user, for example).

CanRaise

If false, calling Raise will have no effect, and may raise a NotSupported error. If true, calling Raise will cause the media application to attempt to bring its user interface to the front, although it may be prevented from doing so (by the window manager, for example).

CanSetFullscreen

If false, attempting to set Fullscreen will have no effect, and may raise an error. If true, attempting to set Fullscreen will not raise an error, and (if it is different from the current value) will cause the media player to attempt to enter or exit fullscreen mode. Note that the media player may be unable to fulfil the request. In this case, the value will not change. If the media player knows in advance that it will not be able to fulfil the request, however, this property should be false.

DesktopEntry

The basename of an installed .desktop file which complies with the Desktop entry specification, with the ".desktop" extension stripped.

Example: The desktop entry file is "/usr/share/applications/vlc.desktop", and this property contains "vlc"

Fullscreen

This property is optional. Clients should handle its absence gracefully. Whether the media player is occupying the fullscreen. This is typically used for videos. A value of true indicates that the media player is taking up the full screen. Media centre software may well have this value fixed to true If CanSetFullscreen is true, clients may set this property to true to tell the media player to enter fullscreen mode, or to false to return to windowed mode. If CanSetFullscreen is false, then attempting to set this property should have no effect, and may raise an error. However, even if it is true, the media player may still be unable to fulfil the request, in which case attempting to set this property will have no effect (but should not raise an error).

HasTrackList

Indicates whether the /org/mpris/MediaPlayer2 object implements the org.mpris.MediaPlayer2.TrackList interface.

IFACE = 'org.mpris.MediaPlayer2'

The D-Bus MediaPlayer2 interface name

Identity

A friendly name to identify the media player to users. This should usually match the name found in

.desktop files (eg: "VLC media player").

Quit (**args*, ***kwds*)
Causes the media player to stop running. The media player may refuse to allow clients to shut it down. In this case, the CanQuit property is false and this method does nothing. Note: Media players which can be D-Bus activated, or for which there is no sensibly easy way to terminate a running instance (via the main interface or a notification area icon for example) should allow clients to use this method. Otherwise, it should not be needed. If the media player does not have a UI, this should be implemented.

Raise (**args*, ***kwds*)
Brings the media player's user interface to the front using any appropriate mechanism available. The media player may be unable to control how its user interface is displayed, or it may not have a graphical user interface at all. In this case, the CanRaise property is false and this method does nothing.

SupportedMimeTypes
The mime-types supported by the media player. Mime-types should be in the standard format (eg: audio/mpeg or application/ogg).

SupportedUriSchemes
The URI schemes supported by the media player. This can be viewed as protocols supported by the player in almost all cases. Almost every media player will include support for the "file" scheme. Other common schemes are "http" and "rtsp". Note that URI schemes should be lower-case.

1.7 TrackList Module

Module provides a *TrackList* class wich implemented MPRIS2 TrackList interface: http://specifications.freedesktop.org/mpris-spec/latest/Track_List_Interface.html

Usage:

```
from pympris import TrackList

tl = TrackList('org.mpris.MediaPlayer2.vlc')
print(tl.Tracks)
tl.RemoveTrack(tl.Tracks[2])
```

class pympris.TrackList.**TrackList** (**args*, ***kwds*)
Bases: pympris.Base.Base

Class implements methods and properties to work with MPRIS2 TrackList interface

AddTrack (**args*, ***kwds*)
Adds a URI in the TrackList.

Parameters

- **uri** (*str*) – The uri of the item to add.
- **after_track** (*str*) – The identifier of the track after which the new item should be inserted.
- **set_as_current** (*bool*) – Whether the newly inserted track should be considered as the current track.

CanEditTracks
If false, calling AddTrack or RemoveTrack will have no effect, and may raise a NotSupported error.

Type bool

GetTracksMetadata (**args*, ***kwds*)
Gets all the metadata available for a set of tracks.

Parameters **track_ids** – list of track ids

Returns Metadata of the set of tracks given as input.

GoTo(**args*, ***kwds*)
Skip to the specified TrackId.

Parameters **track_id** (*str*) – Identifier of the track to skip to.

IFACE = 'org.mpris.MediaPlayer2.TrackList'
The D-Bus MediaPlayer2.Player.TrackList interface name

RemoveTrack(**args*, ***kwds*)
Removes an item from the TrackList.

Parameters **track_id** (*str*) – Identifier of the track to be removed.

Tracks

Returns A list which contains the identifier of each track in the tracklist, in order.

1.8 common Module

Module provides helper functions.

pympris.common.**signal_wrapper**(*f*)
Decorator converts function's arguments from dbus types to python.

pympris.common.**filter_properties_signals**(*f*, *signal_iface_name*)
Filters signals by iface name.

Parameters

- **f** (*function*) – function to wrap.
- **signal_iface_name** (*str*) – interface name.

pympris.common.**convert2dbus**(*value*, *signature*)
Converts *value* type from python to dbus according signature.

Parameters

- **value** – value to convert to dbus object
- **signature** (*str*) – dbus type signature.

Returns value in dbus type.

class pympris.common.**ExceptionMeta**
Bases: type

Metaclass to wrap all class methods and properties using *exception_wrapper* decorator to avoid raising dbus exceptions.

class pympris.common.**ConverterMeta**
Bases: type

Metaclass to wrap all class methods and properties using *converter* decorator to avoid returning dbus types.

CHAPTER 2

Indices and tables

- *genindex*
- *modindex*
- *search*

p