Plyer Documentation

Release 2.2.0.dev0

Mathieu Virbel, Akshay Aurora, Gabriel Petier, Ben Rousch

Jul 14, 2023

Contents

Plyer is a Python library for accessing features of your hardware / platforms.

CHAPTER 1

Plyer

plyer.**accelerometer = <plyer.platforms.linux.accelerometer.LinuxAccelerometer object>**
Accelerometer proxy to *plyer.facades.Accelerometer*

plyer.**audio = <plyer.facades.audio.Audio object>**
Audio proxy to *plyer.facades.Audio*

plyer.**barometer = <plyer.facades.barometer.Barometer object>**
Barometer proxy to *plyer.facades.Barometer*

plyer.**battery = <plyer.facades.battery.Battery object>**
Battery proxy to *plyer.facades.Battery*

plyer.**bluetooth = <plyer.facades.bluetooth.Bluetooth object>**
Bluetooth proxy to *plyer.facades.Bluetooth*

plyer.**brightness = <plyer.facades.brightness.Brightness object>**
Brightness proxy to *plyer.facades.Brightness*

plyer.**call = <plyer.facades.call.Call object>**
Call proxy to :class *plyer.facades.Call*

plyer.**camera = <plyer.facades.camera.Camera object>**
Camera proxy to *plyer.facades.Camera*

plyer.**compass = <plyer.facades.compass.Compass object>**
Compass proxy to *plyer.facades.Compass*

plyer.**cpu = <plyer.platforms.linux.cpu.LinuxCPU object>**
Processors proxy to *plyer.facades.CPU*

plyer.**email = <plyer.platforms.linux.email.LinuxEmail object>**
Email proxy to *plyer.facades.Email*

plyer.**filechooser = <plyer.platforms.linux.filechooser.LinuxFileChooser object>**
FileChooser proxy to *plyer.facades.FileChooser*

plyer.**flash = <plyer.facades.flash.Flash object>**
Flash proxy to *plyer.facades.Flash*

plyer.**gps = <plyer.facades.gps.GPS object>**
GPS proxy to *plyer.facades.GPS*

plyer.**gravity = <plyer.facades.gravity.Gravity object>**
Gravity proxy to *plyer.facades.Gravity*

plyer.**gyroscope = <plyer.facades.gyroscope.Gyroscope object>**
Gyroscope proxy to *plyer.facades.Gyroscope*

plyer.**humidity = <plyer.facades.humidity.Humidity object>**
Humidity proxy to *plyer.facades.Humidity*

plyer.**irblaster = <plyer.facades.irblaster.IrBlaster object>**
IrBlaster proxy to *plyer.facades.IrBlaster*

plyer.**keystore = <plyer.facades.keystore.Keystore object>**
Keyring proxy to :class::*plyer.facades.Keystore*

plyer.**light = <plyer.facades.light.Light object>**
Light proxy to *plyer.facades.Light*

plyer.**maps = <plyer.facades.maps.Maps object>**
Maps proxy to *plyer.facades.Maps*

plyer.**notification = <plyer.facades.notification.Notification object>**
Notification proxy to *plyer.facades.Notification*

plyer.**orientation = <plyer.platforms.linux.orientation.LinuxOrientation object>**
Orientation proxy to *plyer.facades.Orientation*

plyer.**processors = <plyer.platforms.linux.processors.LinuxProcessors object>**
Processors proxy to *plyer.facades.Processors*

plyer.**proximity = <plyer.facades.proximity.Proximity object>**
Proximity proxy to *plyer.facades.Proximity*

plyer.**screenshot = <plyer.facades.screenshot.Screenshot object>**
Screenshot proxy to *plyer.facades.Screenshot*

plyer.**sms = <plyer.facades.sms.Sms object>**
Sms proxy to *plyer.facades.Sms*

plyer.**spatialorientation = <plyer.facades.spatialorientation.SpatialOrientation object>**
SpatialOrientation proxy to *plyer.facades.SpatialOrientation*

plyer.**storagepath = <plyer.platforms.linux.storagepath.LinuxStoragePath object>**
StoragePath proxy to *plyer.facades.StoragePath*

plyer.**stt = <plyer.facades.stt.STT object>**
Speech proxy to *plyer.facades.STT*

plyer.**temperature = <plyer.facades.temperature.Temperature object>**
Temperature proxy to *plyer.facades.Temperature*

plyer.**tts = <plyer.facades.tts.TTS object>**
TTS proxy to *plyer.facades.TTS*

plyer.**uniqueid = <plyer.facades.uniqueid.UniqueID object>**
UniqueID proxy to *plyer.facades.UniqueID*

plyer.**vibrator = <plyer.facades.vibrator.Vibrator object>**
Vibrator proxy to *plyer.facades.Vibrator*

plyer.**wifi = <plyer.facades.wifi.Wifi object>**
Wifi proxy to *plyer.facades.Wifi*

plyer.**devicename = <plyer.platforms.linux.devicename.LinuxDeviceName object>**
devicename proxy to *plyer.facades.DeviceName*

CHAPTER 2

Facades

Interface of all the features available.

class plyer.facades.**Accelerometer**
Accelerometer facade.

acceleration
Property that returns values of the current acceleration sensors, as a (x, y, z) tuple. Returns (None, None, None) if no data is currently available.

disable()
Disable the accelerometer sensor.

enable()
Activate the accelerometer sensor. Throws an error if the hardware is not available or not implemented on.

class plyer.facades.**Audio**(*file_path=None*)
Audio facade.

play()
Play current recording.

start()
Start record.

stop()
Stop record.

class plyer.facades.**Barometer**
Barometer facade.

Barometer sensor is used to measure the ambient air pressure in hPa.

With method *enable* you can turn on pressure sensor and 'disable' method stops the sensor.

Use property *pressure* to get current air pressure in hPa.

New in version 1.2.5.

Supported Platforms:: Android, iOS

disable()
Disable barometer sensor.

enable()
Enable barometer sensor.

pressure
Current air pressure in hPa.

class plyer.facades.**Battery**
Battery info facade.

get_state()
Public method for filling battery.status via platform-specific API in plyer.platforms.

status

Property that contains a dict with the following fields:

- **isCharging** *(bool)*: Battery is charging
- **percentage** *(float)*: Battery charge remaining

> **Warning:** If any of the fields is not readable, it is set as None.

class plyer.facades.**Call**
Call facade.

dialcall()
Opens dialing interface.

makecall(*tel*)
Make calls using your device.

Parameters **tel** (*number*) – The reciever

class plyer.facades.**Camera**
Camera facade.

take_picture(*filename*, *on_complete*)
Ask the OS to capture a picture, and store it at filename.

When the capture is done, on_complete will be called with the filename as an argument. If the callback returns True, the filename will be unlinked.

Parameters

- **filename** (*str*) – Name of the image file
- **on_complete** (*callable*) – Callback that will be called when the operation is done

take_video(*filename*, *on_complete*)
Ask the OS to capture a video, and store it at filename.

When the capture is done, on_complete will be called with the filename as an argument. If the callback returns True, the filename will be unlinked.

Parameters

- **filename** (*str*) – Name of the video file
- **on_complete** (*callable*) – Callback that will be called when the operation is done

class plyer.facades.**Compass**
Compass facade.

New in version 1.2.0.

disable()
Disable the compass sensor.

enable()
Activate the compass sensor.

field
New in version 1.3.1.

Property that returns values of the current compass (magnetic field) sensors, as a (x, y, z) tuple. Returns (None, None, None) if no data is currently available.

field_uncalib
New in version 1.3.1.

Property that returns the current value of Uncalibrated Magnetic Field (without hard iron calibration) along with the iron bias estimation along the three axes.

get_field_uncalib()
New in version 1.3.1.

orientation
WARNING:: This property is deprecated after API level 8. Use *compass.field* instead.

Property that returns values of the current compass (magnetic field) sensors, as a (x, y, z) tuple. Returns (None, None, None) if no data is currently available.

class plyer.facades.**Email**
Email facade.

send(*recipient=None*, *subject=None*, *text=None*, *create_chooser=None*)
Open an email client message send window, prepopulated with the given arguments.

Parameters

- **recipient** – Recipient of the message (str)
- **subject** – Subject of the message (str)
- **text** – Main body of the message (str)
- **create_chooser** – Whether to display a program chooser to handle the message (bool)

Note: create_chooser is only supported on Android

---

class plyer.facades.**FileChooser**
File Chooser facade.

choose_dir(**args*, ***kwargs*)
Open the directory chooser. Note that on Windows this is very limited. Consider writing your own chooser if you target that platform and are planning on using unsupported features.

open_file(**args*, ***kwargs*)
Open the file chooser in “open” mode.

save_file(**args*, ***kwargs*)
Open the file chooser in “save” mode. Confirmation will be asked when a file with the same name already exists.

class plyer.facades.**GPS**
GPS facade.

configure(*on_location*, *on_status=None*)
Configure the GPS object. This method should be called before *start()*.

Parameters

- **on_location**(*callable, multiples keys/value will be passed.*) – Function to call when receiving a new location
- **on_status** (*callable, args are "message-type", "status"*) – Function to call when a status message is received

Warning: The *on_location* and *on_status* callables might be called from another thread than the thread used for creating the GPS object.

start(*minTime=1000*, *minDistance=1*)
Start the GPS location updates. Expects 2 parameters:

minTime: milliseconds. (float) minDistance: meters. (float)

stop()
Stop the GPS location updates.

class plyer.facades.**Gravity**
Gravity facade.

New in version 1.2.5.

Supported Platforms:: Android

disable()
Disable the gravity sensor.

enable()
Activate the gravity sensor. Throws an error if the hardware is not available or not implemented on.

gravity
Property that returns values of the current gravity force as a (x, y, z) tuple. Returns (None, None, None) if no data is currently available.

class plyer.facades.**Gyroscope**
Gyroscope facade.

New in version 1.3.1.

disable()
Disable the Gyroscope sensor.

enable()
Activate the Gyroscope sensor.

orientation
WARNING:: This property is deprecated after API Level 8. Use *gyroscope.rotation* instead.

Property that returns values of the current Gyroscope sensors, as a (x, y, z) tuple. Returns (None, None, None) if no data is currently available.

rotation
Property that returns the rate of rotation around the device's local X, Y and Z axis.

Along x-axis: angular speed around the X axis Along y-axis: angular speed around the Y axis Along z-axis: angular speed around the Z axis

Returns (None, None, None) if no data is currently available.

rotation_uncalib
Property that returns the current rate of rotation around the X, Y and Z axis. An estimation of the drift on each axis is reported as well.

Along x-axis: angular speed (w/o drift compensation) around the X axis Along y-axis: angular speed (w/o drift compensation) around the Y axis Along z-axis: angular speed (w/o drift compensation) around the Z axis

Along x-axis: estimated drift around X axis Along y-axis: estimated drift around Y axis Along z-axis: estimated drift around Z axis

Returns (None, None, None, None, None, None) if no data is currently available.

class plyer.facades.**IrBlaster**
Infrared blaster facade.

exists()
Check if the device has an infrared emitter.

frequencies

Property which contains a list of frequency ranges supported by the device in the form:

[(from1, to1), (from2, to2), ... (fromN, toN)]

static microseconds_to_periods(*frequency*, *pattern*)
Convert a pattern from microseconds to period counts.

static periods_to_microseconds(*frequency*, *pattern*)
Convert a pattern from period counts to microseconds.

transmit(*frequency*, *pattern*, *mode='period'*)
Transmit an IR sequence.

Parameters

***frequency*: int** Carrier frequency for the IR transmission.

***pattern*: list[int]** Burst pair pattern to transmit.

***mode*: str, defaults to 'period'** Specifies the format of the pattern values. Can be 'period' or 'microseconds'.

class plyer.facades.**Light**
Light facade.

Light sensor measures the ambient light level(illumination) in lx. Common uses include controlling screen brightness.

With method *enable* you can turn on the sensor and *disable* method stops the sensor.

Use property *illumination* to get current illumination in lx.

New in version 1.2.5.

Supported Platforms:: Android

disable()
Disable light sensor.

enable()
Enable light sensor.

illumination
Current illumination in lx.

class plyer.facades.**Orientation**
Orientation facade.

set_landscape(*reverse=False*)
Rotate the app to a landscape orientation.

Parameters **reverse** – If True, uses the opposite of the natural orientation.

set_portrait(*reverse=False*)
Rotate the app to a portrait orientation.

Parameters **reverse** – If True, uses the opposite of the natural orientation.

set_sensor(*mode='any'*)
Rotate freely following sensor information from the device.

Parameters **mode** – The rotation mode, should be one of 'any' (rotate to any orientation), 'landscape' (choose nearest landscape mode) or 'portrait' (choose nearest portrait mode). Defaults to 'any'.

class plyer.facades.**Notification**
Notification facade.

notify(*title=''*, *message=''*, *app_name=''*, *app_icon=''*, *timeout=10*, *ticker=''*, *toast=False*, *hints={}*)
Send a notification.

Parameters

- **title** (*str*) – Title of the notification
- **message** (*str*) – Message of the notification
- **app_name** (*str*) – Name of the app launching this notification
- **app_icon** (*str*) – Icon to be displayed along with the message
- **timeout** (*int*) – time to display the message for, defaults to 10
- **ticker** (*str*) – text to display on status bar as the notification arrives
- **toast** (*bool*) – simple Android message instead of full notification
- **hints** (*dict*) – Optional hints that can be used to pass along extra instructions on Linux. (See https://specifications.freedesktop.org/notification-spec/latest/ar01s08.html) # noqa: E501

Note: When called on Windows, `app_icon` has to be a path to a file in .ICO format.

New in version 1.0.0.

Changed in version 1.4.0: Add 'toast' keyword argument

class plyer.facades.**Proximity**
Proximity facade.

The proximity sensor is commonly used to determine distance whether phone is close to your head. Commonly is used when you have a call and you stick your phone with your head. Then screen of phone turns off.

Use method *enable* to turn on proximity sensor and method *disable* for turn off.

To check if some object (or your head) is near sensor check values from property *proximity*. It returns *True* when object is close.

New in version 1.2.5.

Supported Platforms::Android

disable()
Disable the proximity sensor.

enable()
Enable the proximity sensor.

proximity
Return True or False depending if there is an object or not.

Returns True if there is an object. Otherwise False.

class plyer.facades.**Sms**
Sms facade.

send(*recipient*, *message*, *mode=None*, ***kwargs*)
Send SMS or open SMS interface. Includes optional *mode* parameter for macOS that can be set to *'SMS'* if carrier-activated device is correctly paired and configured to macOS.

Parameters

- **recipient** – The receiver
- **message** – the message
- **mode** – (optional, macOS only), can be set to 'iMessage'

(default) or 'SMS'

class plyer.facades.**TTS**
TextToSpeech facade.

speak(*message=''*)
Use text to speech capabilities to speak the message.

Parameters **message** (*str*) – What to speak

class plyer.facades.**UniqueID**
UniqueID facade.

get_uid()
Public method for receiving unique ID via platform-specific API in plyer.platforms.

id
Property that returns the unique id of the platform.

class plyer.facades.**Vibrator**
Vibration facade.

cancel()
Cancels any current vibration, and stops the vibrator.

exists()
Check if the device has a vibrator. Returns True or False.

pattern(*pattern=(0, 1)*, *repeat=-1*)
Ask the vibrator to vibrate with the given pattern, with an optional repeat.

Parameters

- **pattern** – Pattern to vibrate with. Should be a list of times in seconds. The first number is how long to wait before vibrating, and subsequent numbers are times to vibrate and not vibrate alternately. Defaults to `[0, 1]`.
- **repeat** – Index at which to repeat the pattern. When the vibration pattern reaches this index, it will start again from the beginning. Defaults to `-1`, which means no repeat.

vibrate(*time=1*)
Ask the vibrator to vibrate for the given period.

Parameters **time** – Time to vibrate for, in seconds. Default is 1.

class plyer.facades.**Wifi**
Wifi Facade.

connect(*network*, *parameters*, *interface=None*)
Method to connect to some network.

disable()
Wifi interface power state is set to "OFF".

disconnect(*interface=None*)
To disconnect from some network.

enable()
Wifi interface power state is set to "ON".

get_available_wifi()
Returns a list of all the available wifi.

get_network_info(*name*)
Return a dictionary of specified network.

interfaces
List all available WiFi interfaces.

New in version 1.4.0.

is_connected(*interface=None*)
Return connection state of WiFi interface.

New in version 1.4.0.

is_enabled()
Return enabled status of WiFi hardware.

start_scanning(*interface=None*)
Turn on scanning.

class plyer.facades.**Flash**
Flash facade.

off()
Deactiavte the flash

on()
Activate the flash

release()
Release any access to the Flash / Camera. Call this when you're done using the Flash. This will release the Camera, and stop any process.

Next call to *_on* will reactivate it.

class plyer.facades.**CPU**
Facade providing info about sockets, physical and logical number of processors.

cache
Property that contains the count of L1, L2, L3 caches in the system as a dictionary *{'L1': int, 'L2': int, 'L3': int}*.

logical
Property that contains the total number of logical cores (max thread count) in the system.

Note: *sockets * cores per socket * threads per core*

---

numa
Property that contains the count of NUMA nodes in the system.

Note: https://en.wikipedia.org/wiki/Non-uniform_memory_access

---

physical
Property that contains the total number of physical cores (max core count) in the system.

Note: *sockets * cores per socket*

---

sockets
Property that contains the count of CPU sockets.

class plyer.facades.**Temperature**
Temperature facade.

Temperature sensor is used to measure the ambient room temperature in degrees Celsius (°C) With method *enable* you can turn on temperature sensor and 'disable' method stops the sensor. Use property *temperature* to get ambient air temperature in degree C.

New in version 1.2.5.

Supported Platforms:: Android

disable()
Disable temperature sensor.

enable()
Enable temperature sensor.

temperature
Current air temperature in degree C.

class plyer.facades.**Humidity**
Humidity facade. Humidity sensor returns value of humidity. With method *enable* you can turn on Humidity sensor and 'disable' method stops the sensor. Use property *tell* to get humidity value.

Android

disable()
Disable Humidity sensor.

enable()
Enable Humidity sensor.

tell
Current humidity

class plyer.facades.**SpatialOrientation**
Spatial Orientation facade.

Computes the device's orientation based on the rotation matrix.

New in version 1.3.1.

disable_listener()
Disable the orientation sensor.

enable_listener()
Enable the orientation sensor.

orientation
Property that returns values of the current device orientation as a (azimuth, pitch, roll) tuple.

Azimuth, angle of rotation about the -z axis. This value represents the angle between the device's y axis and the magnetic north pole. The range of values is -π to π.

Pitch, angle of rotation about the x axis. This value represents the angle between a plane parallel to the device's screen and a plane parallel to the ground. The range of values is -π to π.

Roll, angle of rotation about the y axis. This value represents the angle between a plane perpendicular to the device's screen and a plane perpendicular to the ground. The range of values is -π/2 to π/2.

Returns (None, None, None) if no data is currently available.

Supported Platforms:: Android

class plyer.facades.**Brightness**
Brightness facade.

current_level()
Know the current level of device's brightness.

set_level(*level*)
Adjust the brightness of the screen. Minimum brightness level:: 1 Maximum brightness level:: 100

Parameters **level** (*int*) – New level of brightness between 1 and 100

class plyer.facades.**Processors**
Number of Processors info facade.

status

Property that contains a dict with the following fields:

- **Number_of_Processors** *(int)*: Number of Processors in

the system

Warning: If any of the fields is not readable, it is set as None.

class plyer.facades.**StoragePath**
StoragePath facade.

get_application_dir()
Get the path of the directory holding application files.

get_documents_dir()
Get the path of standard directory in which to place documents that have been created by the user.

get_downloads_dir()
Get the path of standard directory in which to place files that have been downloaded by the user.

get_external_storage_dir()
Get the path of primary shared or external storage directory.

get_home_dir()
Get the path of home directory of current user.

get_music_dir()
Get the path of standard directory in which to place any audio files that should be in the regular list of music for the user.

get_pictures_dir()
Standard directory in which to place pictures that are available to the user.

get_root_dir()
Get the path of root of the "system" partition holding the core OS.

get_sdcard_dir()
Get the path of external SD card.

New in version 1.4.0.

get_videos_dir()
Get the path of standard directory in which to place videos that are available to the user.

class plyer.facades.**Keystore**
Keystore facade

class plyer.facades.**Bluetooth**
Bluetooth facade.

info
Property that returns the info (currently status) of the bluetooth.

class plyer.facades.**Screenshot**(*file_path=None*)
Screenshot facade.

class plyer.facades.**STT**
Speech to text facade.

errors = []
List of errors found while listening.

exist()
Returns a boolean for speech recognition availability.

language
Return current language.

listening = False
Current state of listening.

partial_results = []
List of results found while the listener is still being active.

prefer_offline = True
Preference whether to use offline language package necessary for each platform dependant implementation or online API.

results = []
List of sentences found while listening. It may consist of many similar and possible sentences that was recognition program.

start()
Start listening.

stop()
Stop listening.

supported_languages
Return list of supported languages used in recognition.

class plyer.facades.**DeviceName**
DeviceName facade.

device_name
Property that returns the device name of the platform.

class plyer.facades.**Maps**
Maps facade.

open_by_address(*address*, ***kwargs*)
Open the specificed location by address in the default Maps API

open_by_lat_long(*latitude*, *longitude*, ***kwargs*)
Open the specificed location by latitude & longitude coordinates in the default Maps API

route(*saddr*, *daddr*, ***kwargs*)
To provide navigation directions from one location to another.

Parameters **saddr** – The source address to be used as the starting

point for directions.

Parameters **daddr** – The destination address to be used as the

destination point for directions.

search(*query*, ***kwargs*)
The query. This parameter is treated as if its value had been typed into the Maps search field by the user.

Note that query=* is not supported

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

Index

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W