Conclusions

Use time.clock()

|  |  |  |
| --- | --- | --- |
| EStatusDialling | 2 | dialing is taking place on a phone line |
| EStatusRinging | 3 | the phone is ringing |

Checking what functions a module contains:

import time  
dir(time) # to get info what the functions are within the module time

start = time.time()

... do something

elapsed = (time.time() - start)

start = time.time()

e

elapsed = (time.time()-start)

print elapsed

13.0

elapsed = (time.time()-start)

print elapsed

20.0

elapsed = (time.time()-start)

print elapsed

22.0

start = time.clock()

... do something

elapsed = (time.clock() - start)

start = time.clock()

elapsed = (time.clock() - start)

print elapsed

8.34375

elapsed = (time.clock() - start)

print elapsed

19.53125

elapsed = (time.clock() - start)

print elapsed

22.25

How to give miss call?

import telephone

def my\_call\_back(param):

if param[0] == telephone.EStatusConnecting:

telephone.hang\_up()

telephone.call\_state(my\_call\_back)

# my\_call\_back fuction will be called whenever there is state change

telephone.dial(str(number))

Enter telephone number and duration in seconds.

**import** telephone, e32, appuifw

number = appuifw.query(u"number", "number")

duration = appuifw.query(u"duration", "number")

telephone.dial(str(number))

**def** handle\_hang\_up(status):

**if** status[0] == telephone.EStatusConnected:

e32.ao\_sleep(float(duration), telephone.hang\_up)

telephone.call\_state(handle\_hang\_up)

Recording calls

**import** appuifw

**import** audio

**import** e32

**import** telephone

**import** time

**class** CallRecorder(object):

**def** \_\_init\_\_(self, save\_path):

self.save\_path = save\_path

self.sound\_file = None

*# Bind the state\_change method to any call state change events*

telephone.call\_state(self.state\_change)

**def** state\_change(self, args):

state = args[0]

**if** state == telephone.EStatusConnected **and** self.sound\_file **is** None:

*# Get the number of the remote caller*

self.number = args[1]

**if** **not** self.number:

self.number = 'Unknown'

*# Create an audio file and start recording*

self.filename = '%s%s\_%s.wav' % (self.save\_path,

time.strftime('%Y%m%d\_%H-%M-%S'),

self.number.lower())

self.sound\_file = audio.Sound.open(self.filename)

self.sound\_file.record()

*# Get a timestamp of when the call was started*

self.start\_time = time.time()

**elif** state != telephone.EStatusConnected **and** self.sound\_file **is** **not** None:

*# Stop recording and close the sound file*

self.sound\_file.stop()

self.sound\_file.close()

self.sound\_file = None

*# "Log" the call details*

self.log\_call(self.number,

self.start\_time,

time.time(),

self.filename)

**def** log\_call(self, number, start\_time, end\_time, filename):

**print** 'Recorded call:'

**print** 'Number: %s' % self.number

**print** 'Duration: %d seconds' % (end\_time - start\_time)

**print** 'Filename: %s**\n**' % self.filename

**if** \_\_name\_\_ == '\_\_main\_\_':

*# Set the application window size/style & title*

appuifw.app.screen = 'normal'

appuifw.app.title = u'Call Recorder'

*# Create an instance of the CallRecorder class*

recorder = CallRecorder(save\_path='e:**\\**call\_recordings**\\**')

*# Create an app lock*

app\_lock = e32.Ao\_lock()

*# Bind the exit key to the app lock signal*

appuifw.app.exit\_key\_handler = app\_lock.signal

*# Block until the app lock signal is received*

app\_lock.wait()