pyminitouch Documentation

Release latest

williamfzc

Oct 30, 2019

Contents:

1 CommandBuilder 1

2 MNTDevice 3

3 MNTConnection 7

4 Indices and tables 9

Index 11

CHAPTER 1

CommandBuilder

class pyminitouch.actions.**CommandBuilder**

Bases: object

Build command str for minitouch.

You can use this, to custom actions as you wish:

```
with safe_connection(_DEVICE_ID) as connection:
    builder = CommandBuilder()
    builder.down(0, 400, 400, 50)
    builder.commit()
    builder.move(0, 500, 500, 50)
    builder.commit()
    builder.move(0, 800, 400, 50)
    builder.commit()
    builder.up(0)
    builder.commit()
    builder.publish(connection)
```

use *d.connection* to get *connection* from device

append(*new_content*)

commit()

add minitouch command: 'c '

down(*contact_id*, *x*, *y*, *pressure*)

add minitouch command: 'd <contact_id> <x> <y> <pressure> '

move(*contact_id*, *x*, *y*, *pressure*)

add minitouch command: 'm <contact_id> <x> <y> <pressure> '

publish(*connection*)

apply current commands (_content), to your device

reset()

clear current commands (_content)

up(*contact_id*)
add minitouch command: ‘u <contact_id> ‘

wait(*ms*)
add minitouch command: ‘w <ms> ‘

CHAPTER 2

MNTDevice

class pyminitouch.actions.**MNTDevice**(*device_id*)

Bases: object

minitouch device object

Sample:

```
device = MNTDevice(_DEVICE_ID)

# It's also very important to note that the maximum X and Y coordinates may, but␣
↪usually do not, match the display size.
# so you need to calculate position by yourself, and you can get maximum X and Y␣
↪by this way:
print('max x: ', device.connection.max_x)
print('max y: ', device.connection.max_y)

# single-tap
device.tap([(400, 600)])
# multi-tap
device.tap([(400, 400), (600, 600)])
# set the pressure, default == 100
device.tap([(400, 600)], pressure=50)

# long-time-tap
# for long-click, you should control time delay by yourself
# because minitouch return nothing when actions done
# we will never know the time when it finished
device.tap([(400, 600)], duration=1000)
time.sleep(1)

# swipe
device.swipe([(100, 100), (500, 500)])
# of course, with duration and pressure
device.swipe([(100, 100), (400, 400), (200, 400)], duration=500, pressure=50)
```

(continues on next page)

(continued from previous page)

```
# extra functions ( their names start with 'ext_' )
device.ext_smooth_swipe([(100, 100), (400, 400), (200, 400)], duration=500,␣
↪pressure=50, part=20)

# stop minitouch
# when it was stopped, minitouch can do nothing for device, including release.
device.stop()
```

ext_smooth_swipe(*points*, *pressure=100*, *duration=None*, *part=None*, *no_down=None*, *no_up=None*)

smoothly swipe between points, one by one it will split distance between points into pieces

before:

```
points == [(100, 100), (500, 500)]
part == 8
```

after:

```
points == [(100, 100), (150, 150), (200, 200), ... , (500, 500)]
```

Parameters

- **points** –
- **pressure** –
- **duration** –
- **part** – default to 10
- **no_down** – will not 'down' at the beginning
- **no_up** – will not 'up' at the end

Returns

reset()

start()

stop()

swipe(*points*, *pressure=100*, *duration=None*, *no_down=None*, *no_up=None*)

swipe between points, one by one

Parameters

- **points** – [(400, 500), (500, 500)]
- **pressure** – default == 100
- **duration** –
- **no_down** – will not 'down' at the beginning
- **no_up** – will not 'up' at the end

Returns

tap(*points*, *pressure=100*, *duration=None*, *no_up=None*)

tap on screen, with pressure/duration

Parameters

- **points** – list, looks like [(x1, y1), (x2, y2)]
- **pressure** – default == 100
- **duration** –
- **no_up** – if true, do not append 'up' at the end

Returns

CHAPTER 3

MNTConnection

class pyminitouch.connection.**MNTConnection**(*port*)

Bases: object

manage socket connection between pc and android

disconnect()

send(*content*)

send message and get its response

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Index

A

append() (*pyminitouch.actions.CommandBuilder method*), 1

C

CommandBuilder (*class in pyminitouch.actions*), 1
commit() (*pyminitouch.actions.CommandBuilder method*), 1

D

disconnect() (*pyminitouch.connection.MNTConnection method*), 7
down() (*pyminitouch.actions.CommandBuilder method*), 1

E

ext_smooth_swipe() (*pyminitouch.actions.MNTDevice method*), 4

M

MNTConnection (*class in pyminitouch.connection*), 7
MNTDevice (*class in pyminitouch.actions*), 3
move() (*pyminitouch.actions.CommandBuilder method*), 1

P

publish() (*pyminitouch.actions.CommandBuilder method*), 1

R

reset() (*pyminitouch.actions.CommandBuilder method*), 1
reset() (*pyminitouch.actions.MNTDevice method*), 4

S

send() (*pyminitouch.connection.MNTConnection method*), 7
start() (*pyminitouch.actions.MNTDevice method*), 4
stop() (*pyminitouch.actions.MNTDevice method*), 4
swipe() (*pyminitouch.actions.MNTDevice method*), 4

T

tap() (*pyminitouch.actions.MNTDevice method*), 4

U

up() (*pyminitouch.actions.CommandBuilder method*), 1

W

wait() (*pyminitouch.actions.CommandBuilder method*), 2