
airtest Documentation

Release 1.x

codeskyblue

Jan 31, 2018

Contents

1	Documentation	3
1.1	Public class	3
1.2	Common methods	3
1.3	Method only for android	5
1.4	Method only for windows	8
2	Indices and tables	9

Release you from the boring testing work.

1.1 Public class

class `atx.drivers.Pattern` (*name*, *image=None*, *offset=None*, *anchor=0*, *rsl=None*, *resolution=None*, *th=None*, *threshold=None*)

save (*path*)
save image to path

1.2 Common methods

class `atx.drivers.mixin.DeviceMixin`

add_listener (*fn*, *event_flags*)

Listen event Args:

- *fn*: function call when event happens
- **event_flags: for example** `EVENT_UIAUTO_CLICK | EVENT_UIAUTO_SWIPE`

Returns: None

click (*x*, *y*)

Args: *x*, *y* (float): position to tap

Example: if *x*, *y* both less than 1.0. then *x*, *y* means percentage position

`d.click(0.5, 0.5)` # click center of screen `d.click(20, 10)` # click position(20, 10)

click_exists (**args*, ***kwargs*)

Click when target exists Example usage:

- `click_exists("button.png")`

- `click_exists(text="Update")`

click_image (**args, **kwargs*)

Simulate click according image position

Args:

- `pattern` (str or Pattern): filename or an opencv image object.
- `timeout` (float): if image not found during this time, `ImageNotFoundError` will raise.
- `action` (str): click or long_click
- `safe` (bool): if safe is True, Exception will not raise and return None instead.
- `method` (str): image match method, choice of <templatsift>
- `delay` (float): wait for a moment then perform click

Returns: None

Raises: `ImageNotFoundError`: An error occurred when img not found in current screen.

click_nowait (**args, **kwargs*)

Return immediately if no image found

Args:

- `pattern` (str or Pattern): filename or an opencv image object.
- `action` (str): click or long_click

Returns: Click point or None

delay (*secs*)

Delay some seconds

`secs`: float seconds

Returns: self

exists (*pattern, **match_kwargs*)

Check if image exists in screen

Returns: If exists, return `FindPoint`, or return None if `result.confidence < self.image_match_threshold`

free_screen ()

Unlock `keep_screen()`

keep_screen ()

Freeze screenshot, so all image functions will not take images, until call `free_screen()`

match (*pattern, screen=None, rect=None, offset=None, threshold=None, method=None*)

Check if image position in screen

Args:

- `pattern`: Image file name or opencv image object
- `screen` (PIL.Image): optional, if not None, screenshot method will be called
- `threshold` (float): it depends on the image match method
- `method` (string): choices on <template | sift>

Returns: None or FindPoint, For example:

FindPoint(pos=(20, 30), method='tpl', confidence=0.801, matched=True)

Only when confidence > self.image_match_threshold, matched will be True

Raises: TypeError: when image_match_method is invalid

match_all (*pattern*)

Test method, not suggested to use

region (*bounds*)

Set region of the screen area Args:

bounds: Bounds object

Returns: A new AndroidDevice object

Raises: TypeError

region_screenshot (*filename=None*)

Deprecated Take part of the screenshot

screenshot (**args, **kwargs*)

Take screen snapshot

Args:

- filename: filename where save to, optional

Returns: PIL.Image object

Raises: TypeError, IOError

touch (*x, y*)

Alias for click

touch_image (**args, **kwargs*)

Alias for click_image

wait (*pattern, timeout=10.0, safe=False, **match_kwargs*)

Wait till pattern is found or time is out (default: 10s).

1.3 Method only for android

class atx.drivers.android.**AndroidDevice** (*serial=None, **kwargs*)

serial

Android device serial number. **Optional**

adb_cmd (*command, **kwargs*)

Run adb command, for example: adb(['pull', '/data/local/tmp/a.png'])

Args: command: string or list of string

Returns: command output

adb_shell (*command, **kwargs*)

Run adb shell command

Args: command: string or list of string

Returns: command output

clear_text (*count=100*)

Clear text Args:

- count (int): send KEY_DEL count

current_app ()

Get current app (package, activity) Returns:

namedtuple ['package', 'activity', 'pid'] activity, pid maybe None

Raises: RuntimeError

current_ime ()

Get current input method

display

Virtual keyboard may get small d.info['displayHeight']

do_tap (*x, y*)

Touch specify position

Args: x, y: int

Returns: None

dump_nodes ()

Dump current screen UI to list Returns:

List of UINode object, For example:

```
[UINode( bounds=Bounds(left=0,      top=0,      right=480,      bottom=168),      check-
         able=False,      class_name='android.view.View',      text='',      resource_id='',      pack-
         age='com.sonyericsson.advancedwidget.clock')]
```

dump_view ()

Current Page XML

enable_ime (*ime*)

Enable input methods

Args:

- ime(string): for example "android.unicode.ime/.Utf7ImeService"

Raises: RuntimeError

forward (*device_port, local_port=None*)

Forward device port to local Args:

device_port: port inside device local_port: port on PC, if this value is None, a port will random pick one.

Returns: tuple, (host, local_port)

input_methods ()

Get all input methods

Return example: ['com.sohu.inputmethod.sogou/.SogouIME', 'android.unicode.ime/.Utf7ImeService']

is_app_alive (*package_name*)

Deprecated: use current_package_name instead. Check if app is running in foreground

keyevent (*keycode*)

call adb shell input keyevent \${keycode}

Args:

- keycode(string): for example, KEYCODE_ENTER

keycode need reference: <http://developer.android.com/reference/android/view/KeyEvent.html>

prepare_ime ()

Change current method to adb-keyboard

Raises: RuntimeError

properties

Android Properties, extracted from *adb shell getprop*

Returns: dict of props, for example:

```
{ 'ro.bluetooth.dun': 'true' }
```

raw_cmd (**args, **kwargs*)

Return subprocess.Process instance

rotation

Rotation of the phone

0: normal 1: home key on the right 2: home key on the top 3: home key on the left

serial

Android Device Serial Number

sleep (*secs=None*)

Deprecated. use delay instead.

source (**args, **kwargs*)

Dump page xml

start_app (*package_name, activity=None, stop=False*)

Start application

Args:

- package_name (string): like com.example.app1
- activity (string): optional, activity name

Returns time used (unit second), if activity is not None

Document: usage: adb shell am start -D: enable debugging -W: wait for launch to complete --start-profiler <FILE>: start profiler and send results to <FILE> --sampling INTERVAL: use sample profiling with INTERVAL microseconds

between samples (use with --start-profiler)

-P <FILE>: like above, but profiling stops when app goes idle -R: repeat the activity launch <COUNT> times. Prior to each repeat,

the top activity will be finished.

-S: force stop the target app before starting the activity --opengl-trace: enable tracing of OpenGL functions --user <USER_ID> | current: Specify which user to run as; if not

specified then run as the current user.

stop_app (*package_name, clear=False*)

Stop application

Args: package_name: string like com.example.app1 clear: bool, remove user data

Returns: None

takeSnapshot (*filename*)

Deprecated, use screenshot instead.

type (*s*, *enter=False*, *next=False*, *clear=False*, ***ui_select_kwargs*)

Input some text, this method has been tested not very stable on some device. “Hi world” maybe spell into “H iworld”

Args:

- *s*: string (text to input), better to be unicode
- *enter*(bool): input enter at last
- *next*(bool): perform editor action Next
- *clear*(bool): clear text before type
- *ui_select_kwargs*(**): tap then type

The android source code show that space need to change to %s interesting thing is that if want to input %s, it is really unconvinient. android source code can be found here. https://android.googlesource.com/platform/frameworks/base/+android-4.4.2_r1/cmds/input/src/com/android/commands/input/Input.java#159 app source see here: <https://github.com/openatx/android-unicode>

uiautomator

Returns: uiautomator: Device object describes in <https://github.com/openatx/atx-uiautomator>

wlan_ip

Wlan IP

1.4 Method only for windows

CHAPTER 2

Indices and tables

- `genindex`
- `modindex`
- `search`

A

adb_cmd() (atx.drivers.android.AndroidDevice method), 5
adb_shell() (atx.drivers.android.AndroidDevice method), 5
add_listener() (atx.drivers.mixin.DeviceMixin method), 3
AndroidDevice (class in atx.drivers.android), 5

C

clear_text() (atx.drivers.android.AndroidDevice method), 6
click() (atx.drivers.mixin.DeviceMixin method), 3
click_exists() (atx.drivers.mixin.DeviceMixin method), 3
click_image() (atx.drivers.mixin.DeviceMixin method), 4
click_nowait() (atx.drivers.mixin.DeviceMixin method), 4
current_app() (atx.drivers.android.AndroidDevice method), 6
current_ime() (atx.drivers.android.AndroidDevice method), 6

D

delay() (atx.drivers.mixin.DeviceMixin method), 4
DeviceMixin (class in atx.drivers.mixin), 3
display (atx.drivers.android.AndroidDevice attribute), 6
do_tap() (atx.drivers.android.AndroidDevice method), 6
dump_nodes() (atx.drivers.android.AndroidDevice method), 6
dump_view() (atx.drivers.android.AndroidDevice method), 6

E

enable_ime() (atx.drivers.android.AndroidDevice method), 6
exists() (atx.drivers.mixin.DeviceMixin method), 4

F

forward() (atx.drivers.android.AndroidDevice method), 6
free_screen() (atx.drivers.mixin.DeviceMixin method), 4

I

input_methods() (atx.drivers.android.AndroidDevice method), 6
is_app_alive() (atx.drivers.android.AndroidDevice method), 6

K

keep_screen() (atx.drivers.mixin.DeviceMixin method), 4
keyevent() (atx.drivers.android.AndroidDevice method), 6

M

match() (atx.drivers.mixin.DeviceMixin method), 4
match_all() (atx.drivers.mixin.DeviceMixin method), 5

P

Pattern (class in atx.drivers), 3
prepare_ime() (atx.drivers.android.AndroidDevice method), 7
properties (atx.drivers.android.AndroidDevice attribute), 7

R

raw_cmd() (atx.drivers.android.AndroidDevice method), 7
region() (atx.drivers.mixin.DeviceMixin method), 5
region_screenshot() (atx.drivers.mixin.DeviceMixin method), 5
rotation (atx.drivers.android.AndroidDevice attribute), 7

S

save() (atx.drivers.Pattern method), 3
screenshot() (atx.drivers.mixin.DeviceMixin method), 5
serial (atx.drivers.android.AndroidDevice attribute), 5, 7
sleep() (atx.drivers.android.AndroidDevice method), 7
source() (atx.drivers.android.AndroidDevice method), 7
start_app() (atx.drivers.android.AndroidDevice method), 7

`stop_app()` (`atx.drivers.android.AndroidDevice` method),
7

T

`takeSnapshot()` (`atx.drivers.android.AndroidDevice`
method), 8

`touch()` (`atx.drivers.mixin.DeviceMixin` method), 5

`touch_image()` (`atx.drivers.mixin.DeviceMixin` method),
5

`type()` (`atx.drivers.android.AndroidDevice` method), 8

U

`uiautomator` (`atx.drivers.android.AndroidDevice` at-
tribute), 8

W

`wait()` (`atx.drivers.mixin.DeviceMixin` method), 5

`wlan_ip` (`atx.drivers.android.AndroidDevice` attribute), 8