magneto Documentation

Release 0.1

EverythingMe Automation

October 14, 2015

Contents

CHAPTER 1

Installation

it is recommended to have a virtualenv activated

```
pip install magneto
```

Running tests

If a virtualenv is available, activate it before run

```
$ magneto run <TEST_DIR> --apk-path <PATH_TO_APK> --apk-package <APK_PACKAGE_NAME> --app-activity <AP
```

Required:

Option	Description
`<TEST_DIR>`	Test file directory path. Usually `ui_tests` (Required)

Additional params:

Option	Description
`--app-package <APK_PACKAGE_NAME>`	Tested app package name. e.g. `--app-package com.facebook.katana`
`--app-activity <APK_ACTIVITY>`	Tested app activity string. e.g. `--app-activity com.facebook.katana.LoginActivity`
`-k <TEST_NAME / TAG>`	Run only matching tests by test name (i.e. `test_my_feature`) or @tag decorator (i.e. `search_results`)
`--clean-install`	Uninstalls and then installs the tested apk specifies in the parameter `--apk-path`.
`--apk-path <PATH_TO_APK>`	Path to tested app apk file. e.g `--app-apk /Users/username/Downloads/facebook.apk`. Required when using `--clean-install`
`--device-id`	Points Magneto to a specific device available in `adb devices`
`--clean-install`	Uninstalls and then installs the tested apk specifies in the parameter `--apk-path`.
`--save-data-on-failure`	Saves failed test data such as adb logcat (logcat.log), element hierarchy (hierarchy.xml) and corresponding screen capture (screenshot.png).
`--include-video-on-failure`	Instructs Magneto to save video capture (video.mp4) alongside the above default assets. Requires parameter `--save-data-on-failure`
`--magneto-failed-data-dir`	Allows overriding the default `/tmp/magneto_test_data/` directory for failed test data.
`--wait-for-element-timeout`	Allows overriding the default 5000ms (5 seconds) element find timeout.

CHAPTER 3

Run the demo

Install magento:

```
$ pip install magneto
```

Enter a new folder:

```
$ mkdir calc
$ cd calc
```

Init the calc demo:

```
$ magneto init calc
```

Run the tests:

```
$ magneto run tests/
```

This will install the app on your device, run the tests and display a final report.

Here's a quick video showcasing the demo

CHAPTER 4

Writing tests

Tests are defined within a test case file. e.g `login_test.py`. It is recommended to store these files under a `ui_tests` folder:

```
- ui_tests
  - boarding_test.py
  - login_test.py
  - mainpage_test.py
  - feature_test.py
  - other_feature_test.py
```

Test cases inherit from *BaseTestCase class* and utilize features such as setup, teardown and assertions.

4.1 Defining a test

Example:

```
from magento.base import BaseTestCase

class ExampleTest(BaseTestCase):
    """
    Tests example functionality
    """

    def test_example_1(self):
        ...

    def test_example_2(self):
        ...

    def test_example_3(self):
        ...
```

Tests are defined as functions prefixed with *test_* and usually include some kind of assertion.

Example:

```
def test_example(self):
   # scroll fast to bottom
   self.magneto(scrollable=True).fling()
   # get element by id
   el = self.magneto(text=ids.foo)
```

```
    # assert that the element exists in the current view
    Assert.true(el.exists)
```

4.2 Results

Test results are available via Magneto logs:

```
ui_tests/ftu_test.py .s
ui_tests/cards_test.py F
ui_tests/discovery_test.py .
ui_tests/folders_test.py ..s..
ui_tests/homescreen_test.py .
ui_tests/magneto_test.py ....
ui_tests/search_test.py .F....
```

In the log above, each line represents a test case and its tests results.

- `.` = passed
- `s` = skipped
- `F` = failed

A summary at the end:

```
=============== 2 failed, 16 passed, 2 skipped in 348.66 seconds ===============
```

Failed tests log more information, pointing to where an error or assertion failure occurred:

```
self = <magneto.test.cards_test.CardsTestCase testMethod=test_browse_news_cards>

def test_browse_news_cards(self):
    """
        Test browse through cards in News smart folder
        """
        cards = self.magneto(resourceId=ids.card)

        folder = Folder(folders.news)
        folder.click()
        Assert.true(cards.exists)

        folder.menu.click()
        self.magneto(text=names.hide_cards).click()
        self.magneto.press.home()
        folder.click()
        Assert.false(cards.exists)

        folder.menu.click()
        self.magneto(text=names.show_cards).click()
        self.magneto.press.home()
        folder.click()
>       Assert.true(cards.exists)
E       AssertionError: False is not true

magneto/test/cards_test.py:29: AssertionError
```

4.3 Even more data about failed tests

Magneto can be instructed to capture adb logcat logs, element hierarchy and screen image at the moment the fail was determined. When all test runs are over, these files are made available in the dedicated folder (usually `tmp/magneto_test_data` unless specified differently with the `--magneto_failed_data_dir` parameter) as one bundled file. This file could be made available in CI systems as a build artifact.

Example:

```
- Nexus4-01acd7ef4c3d12d4 4.53.24 PM
  - 7-test_example_1-201503081428-1.video.mp4
  - 7-test_example_1-201503081428.hierarchy.uix
  - 7-test_example_1-201503081428.logcat.log
  - 7-test_example_1-201503081428.screenshot.png
```

CHAPTER 5

Common usages

5.1 Tag

Magneto allows attaching tags to test cases and tests alike, using `@pytest.mark.TAG_NAME`:

```
@pytest.mark.cards
class CardsTestCase(BaseTestCase):
    """
    Tests example functionality
    """

    @pytest.mark.toggle
    def test_cards_toggle(self):
        ...
```

Now I can choose to run only `CardsTestCase` using `-k cards`:

```
$ magneto run ui_tests/ -k cards
```

5.2 Skip

Magneto allows conditional skipping for test cases and tests, using `@pytest.mark.skipif(CONDITION, reason=REASON)`:

```
@pytest.mark.skipif(settings.param == False, reason='Skipped cause param is False')
class ExampleTestCase(BaseTestCase):
    """
    Tests example functionality
    """

    def test_example_1(self):
        ...
```

5.3 Blockers

Some tests can be defined as blockers so that if they fail, all the rest would be skipped. The boarding test, for instance, is considered a blocker as there's no point in continuing with following tests if the boarding process failed. We do this by using a pytest plugin called pytest-blocker.

Example:

```
class ExampleTestCase(BaseTestCase):
    """
    Tests example functionality
    """

    @pytest.mark.blocker
    def test_example_1(self):
        ...
```

CHAPTER 6

API Reference

6.1 The Magneto class

6.2 BaseTestCase class

6.3 Utilities