Odoo Report Testing Documentation

Release 0.2

Pierre Verkest, Odoo Community Association

Aug 19, 2020

Contents

1 Resources **3**

2 Requirements **5**

3 Quickstart **7**

4 How it works **9**

5 Settings **11**

6 Contents **13**

6.1 Install 13

6.2 Contribute 14

6.3 API Documentation 15

Python Module Index **19**

Index **21**

This lib provide tools to test odoo reports from version 7 and higher.

CHAPTER 1

Resources

- Documentation
- Issue Tracker
- Code

CHAPTER 2

Requirements

- `imagemagick` is used to compare 2 (one page) pdf bit to bit.
- `pdftk` is used to split pdf page per page.
- Install this package (with pip `pip install odoo-report-testing`).

CHAPTER 3

Quickstart

Here an example to test sale order quotation report `test_so_report.py`:

```python
# -*- coding: utf-8 -*-
import os
from openerp.tests.common import TransactionCase
from odoo_report_testing.assertions import OdooAssertions



class TestSoReport(TransactionCase, OdooAssertions):

    def test_simple_so_report(self):
        self.assertOdooReport(
            os.path.join(
                os.path.dirname(__file__),
                'expected_reports',
                'test_so_report.pdf'
            ),
            'sale.order',
            'sale.report_saleorder',
            [self.ref('sale.sale_order_1')],
            data={},
            context=None
        )
```

Assuming your module looks like:

```
my_module
├── test
│   ├── expected_reports
│   │   └── test_so_report.pdf
│   ├── __init__.py
│   └── test_so_report.py
```

Warning: You may want to generate those report without expose any odoo port so that you can render report properly without http access.

You can have a look to one of the following PRs:

- odoo/odoo
- oca/ocb
- anybox/odoo

CHAPTER 4

How it works

When using `assertOdooReport` the library will:

- Ask odoo to generates the report to test.
- Save the generate pdf on the file system.
- The generated report and the reference pdf are split page per page using **pdftk**.
- Each pdf page are compared using **compare** program from **imagemagick**.
- When a generated pdf page is different from its reference 2 images are generated:
  - A `.png` image with red color for diff pixels.
  - A `.gif` image with the generated page blinking (so only addition are visible).

CHAPTER 5

Settings

Mainly thinks for CI availaible environement variable:

- **REPORT_TESTING_OUTPUT_DIR**: Directory where are saved all generated files (report, diff files, . . .), if not provide the directory of the reference file is used.

CHAPTER 6

Contents

6.1 Install

We are going to see in details what are dependencies then various way to install this library:

- with pip
- from code source
- using a buildout configuration

6.1.1 External requirements

This library call `compare` from `imagemagick` to compare 2 (one page) pdf bit to bit. `pdftk` is used to split multi page pdf file into multiple single page pdf files.

To install those dependecies on a debian based system:

```
$ sudo aptitude install imagemagick pdftk
```

6.1.2 Install Odoo report testing

Using pip

The latest release pushed on pypi can be installed using pip:

```
pip install odoo-report-testing
```

Refer to the pip user guide for an advanced usage of pip!

Using anybox recipe odoo

Here a simple example of odoo 8 configuration:

```
[buildout]
parts = odoo
versions = versions
# Un-comment following 2 lines if you want to hack odoo report testing
# in your current project
# extensions = gp.vcsdevelop
# vcs-extend-develop = git+https://github.com/anybox/odoo-report-testing@master
↪#egg=odoo-report-testing

[odoo]
recipe = anybox.recipe.odoo:server
version = git http://github.com/anybox/odoo.git ocb 8.0-render_report_offline
addons = local my_addons

openerp_scripts = nosetests=nosetests command-line-options=-d

eggs =
    odoo-report-testing
    anybox.recipe.odoo
    nose
    coverage
    soappy
    PyPDF
    pysftp

[versions]
psutil = 2.2.1
feedparser = 5.1.3
paramiko = 1.16.0
gevent = 1.0.2
pysftp = 0.2.8
wstools = 0.4.3
```

6.2 Contribute

You can suggest a change by creating a PR on github against *master* branch.

6.2.1 Setup development environment

To launch unittest you have to get an odoo instance running, however as far as I know from odoo launcher you can't run test from files that are not part of an odoo module.

So let's use nose with the anybox.recipe.odoo:

```
# create a python virtualenv with no pip/setuptools
virtualenv -p python2 odoo-sandbox --no-setuptools
# clone odoo-report-testing repo
git clone https://github.com/anybox/odoo-report-testing
cd odoo-report-testing
# setup buildout
```

(continues on next page)

(continued from previous page)

```
../odoo-sandbox/bin/python bootstrap.py
# setup project according the choosen buildout.cfg
bin/buildout -c buildout.cfg
# create an empty pg database
createdb ort
# setup ort database with odoo base module
bin/start_odoo -d ort --stop-after-int -i base
# launch unittest test of odoo report testing
bin/nosetests -d ort -- -s -v odoo_report_testing/tests/
```

Voila!

6.3 API Documentation

6.3.1 Odoo Assertions

High level Odoo assertions.

class `odoo_report_testing.assertions.`**`OdooAssertions`**

Mixin class providing assertion and helper methods to write tests.

`assertImage`(*ref*, *compared*, *msg=None*, *output_dir=None*)

Test if two images are equals, if not, this generate a diff file and a animated gif file to highlight differences.

It could be two single page pdf files.

This delegate the work to `compare` application installed with **imagemagick** package which is required.

Parameters

- **`ref`** – a `path` to the file used as reference, expected result
- **`compared`** – a `path` to the file to compare to the ref file.
- **`msg`** – A message to print in case of faillure

Output_dir Directory to generate diff files

`assertOdooReport`(*reference*, *model*, *report_service_name*, *ids*, *data=None*, *context=None*)

Generate report and compare to a reference file, test will failed if files are different, have a look close to the reference file you will find a diff picture that show you differences.

here an example to test sale order quotation report:

```python
# -*- coding: utf-8 -*-
import os
from openerp.tests.common import TransactionCase
from odoo_report_testing.assertions import OdooAssertions


class TestSoReport(TransactionCase, OdooAssertions):

    def test_simple_so_report(self):
        self.assertOdooReport(
            os.path.join(
                os.path.dirname(__file__),
                'expected_reports',
```

(continues on next page)

(continued from previous page)

```
                'test_so_report.pdf'
            ),
            'sale.order',
            'sale.report_saleorder',
            [self.ref('sale.sale_order_1')],
            data={},
            context=None
        )
```

Warning: You may want to generate those report without expose any odoo port so that you can render report properly without http access.

You can follow this PR:

- anybox/odoo

Parameters

- **`reference`** – Path to the report that the generated report should looks like
- **`model`** – fully qualified model name
- **`report_service_name`** – report name (without *report.*)
- **`ids`** – object used to generate the report
- **`data`** – extra data given to draw the report
- **`context`** – odoo context

`assertPdf`(*ref*, *compared*, *msg=None*, *output_dir=None*)

Test if two pdf are equals

This split the pdf and delegate the assertion to assertImageEquals.

To split the pdf `pdftk` application from **pdftk** package is required.

6.3.2 Pdf tools

`class` `odoo_report_testing.reports.`**`pdftools`**

Utility class to generate pdf file from odoo record, compare 2 pdf files

`static generateReport`(*cr*, *uid*, *model*, *report_service_name*, *ids*, *data=None*, *context=None*, *version7=False*)

Generate the report and return it as a tuple (result, format) where *result* is the report document and *format* is the file extension.

`static imagediff`(*ref*, *compared*, *output_dir=None*)

Test if two images are equals, if not, this generate a diff file and a animated gif file to highlight differences.

It can be two single page pdf file.

This delegate the work to `compare` application installed with **imagemagick** package which is required.

Parameters

- **`ref`** – a `path` to the file used as reference, expected result
- **`compared`** – a `path` to the file to compare to the ref file.

Output_dir Directory to generate diff files

return (Boolean equals, [String path to diff files,]): The result is a a tuple with bool value to tell if files are equals or not, and a list of diff files generated.

`static outputs_env`(*expected_file*, *output_dir=None*)
Return tuple with directory and base file name to use from expected file

`static pdfdiff`(*ref*, *compared*, *output_dir=None*)
Test if two pdf are equals

This split the pdf and delegate to imagediff to compare each pages.

To split the pdf `pdftk` application from **pdftk** package is required.

Python Module Index

o

odoo_report_testing.assertions, 15
odoo_report_testing.reports, 16

Index

A

`assertImage()` (*odoo_report_testing.assertions.OdooAssertions method*), 15

`assertOdooReport()` (*odoo_report_testing.assertions.OdooAssertions method*), 15

`assertPdf()` (*odoo_report_testing.assertions.OdooAssertions method*), 16

G

`generateReport()` (*odoo_report_testing.reports.pdftools static method*), 16

I

`imagediff()` (*odoo_report_testing.reports.pdftools static method*), 16

O

`odoo_report_testing.assertions` (*module*), 15

`odoo_report_testing.reports` (*module*), 16

`OdooAssertions` (*class in odoo_report_testing.assertions*), 15

`outputs_env()` (*odoo_report_testing.reports.pdftools static method*), 17

P

`pdfdiff()` (*odoo_report_testing.reports.pdftools static method*), 17

`pdftools` (*class in odoo_report_testing.reports*), 16