InvoiceGenerator Documentation

Release

See CONTRIBUTORS.rst

Sep 11, 2017

Contents

1 Modules **1**
- 1.1 InvoiceGenerator.api . . . 1
- 1.2 InvoiceGenerator.pdf . . . 4
- 1.3 InvoiceGenerator.pohoda . . . 4

2 History **5**
- 2.1 1.0.0 - 2017-09-10 . . . 5
- 2.2 0.5.4 - 2017-03-22 . . . 5
- 2.3 0.5.3 - 2017-01-09 . . . 5
- 2.4 0.5.2 - 2014-12-04 . . . 5
- 2.5 0.5.1 - 2014-10-28 . . . 6
- 2.6 0.5.0 - 2014-09-21 . . . 6
- 2.7 0.4.9 - 2014-07-3 . . . 6
- 2.8 0.4.8 - 2014-07-3 . . . 6
- 2.9 0.4.7 - 2014-07-1 . . . 6
- 2.10 0.4.6 - 2014-05-14 . . . 6
- 2.11 0.4.5 - 2014-04-21 . . . 6

3 Contributors **7**

4 InvoiceGenerator **9**
- 4.1 Installation . . . 9
- 4.2 Example . . . 9
- 4.3 Hacking . . . 10
- 4.4 Indices and tables . . . 11

Python Module Index **13**

CHAPTER 1

Modules

InvoiceGenerator is made of four submodules:

- *InvoiceGenerator.api*
- *InvoiceGenerator.pdf*
- *InvoiceGenerator.pohoda*

InvoiceGenerator.api

class `InvoiceGenerator.api.`**`Address`**(*summary*, *address=''*, *city=''*, *zip_code=''*, *phone=''*, *email=''*, *bank_name=''*, *bank_account=''*, *bank_code=''*, *note=''*, *vat_id=''*, *ir=''*, *logo_filename=''*, *vat_note=''*)

Bases: `InvoiceGenerator.api.UnicodeProperty`

Abstract address definition

Parameters

- **`summary`** – address header line - name of addressee or company name
- **`address`** – line of the address with street and house number
- **`city`** – city or part of the city
- **`zip_code`** – zip code (PSČ in Czech)
- **`phone`** –
- **`email`** –
- **`bank_name`** –
- **`bank_account`** – bank account number
- **`bank_code`** –
- **`note`** – note that will be written on the invoice

- **vat_id** – value added tax identification number (DIČ in czech)
- **ir** – Taxpayer identification Number (IČO in czech)
- **logo_filename** – path to the image of logo of the company

bank_account_str()
Returns bank account identifier with bank code after slash

class InvoiceGenerator.api.**Client**(*summary*, *address=''*, *city=''*, *zip_code=''*, *phone=''*, *email=''*, *bank_name=''*, *bank_account=''*, *bank_code=''*, *note=''*, *vat_id=''*, *ir=''*, *logo_filename=''*, *vat_note=''*)
Bases: *InvoiceGenerator.api.Address*

Definition of client (recipient of the invoice) address.

bank_account_str()
Returns bank account identifier with bank code after slash

class InvoiceGenerator.api.**Provider**(*summary*, *address=''*, *city=''*, *zip_code=''*, *phone=''*, *email=''*, *bank_name=''*, *bank_account=''*, *bank_code=''*, *note=''*, *vat_id=''*, *ir=''*, *logo_filename=''*, *vat_note=''*)
Bases: *InvoiceGenerator.api.Address*

Definition of prvider (subject, that issued the invoice) address.

bank_account_str()
Returns bank account identifier with bank code after slash

class InvoiceGenerator.api.**Creator**(*name*, *stamp_filename=''*)
Bases: InvoiceGenerator.api.UnicodeProperty

Definition of creator of the invoice (ussually an accountant).

Parameters

- **name** – name of the issuer
- **stamp_filename** – path to file with stamp (or subscription)

class InvoiceGenerator.api.**Item**(*count*, *price*, *description=''*, *unit=''*, *tax=Decimal('0')*)
Bases: object

Item on the invoice.

Parameters

- **count** – number of items or quantity associated with unit
- **price** – price for unit
- **unit** – unit in which it is measured (pieces, Kg, l)
- **tax** – the tax rate under which the item falls (in percent)

count
Count or amount of the items.

count_tax()
Value of only tax that will be payed for the items.

description
Short description of the item.

price
Price for unit.

tax
Tax rate.

total
Total price for the items without tax.

total_tax
Total price for the items with tax.

unit
Unit.

class InvoiceGenerator.api.**Invoice**(*client*, *provider*, *creator*)
Bases: InvoiceGenerator.api.UnicodeProperty

Invoice definition

Parameters

- **client** (Client) – client of the invoice
- **creator** (Creator) – creator of the invoice
- **provider** (Provider) – provider of the invoice

add_item(*item*)
Add item to the invoice.

Parameters **item** (*Item class*) – the new item

currency = u'K\u010d'
currency identifier (e.g. "$" or "Kč")

currency_locale = 'cs_CZ.UTF-8'
currency_locale: locale according to which will be the written currency representations

date = None
date of exposure

difference_in_rounding
Difference between rounded price and real price.

generate_breakdown_vat()

generate_breakdown_vat_table()

iban = None
iban

items
Items on the invoice.

number = None
number or string used as the invoice identifier

payback = None
due date

paytype = None
textual description of type of payment

price
Total sum price without taxes.

price_tax
Total sum price including taxes.

rounding_result = False
round result to integers?

rounding_strategy = 'ROUND_HALF_EVEN'
Result rounding strategy (identifiers from *decimal* module). Default strategy for rounding in Python is bankers' rounding, which means that half of the X.5 numbers are rounded down and half up. Use this parameter to set different rounding strategy.

specific_symbol = None
specific_symbol

swift = None
swift

taxable_date = None

taxable date

title = ''
title on the invoice

use_tax = False

variable_symbol = None
variable symbol associated with the payment

InvoiceGenerator.pdf

class InvoiceGenerator.pdf.**SimpleInvoice**(*invoice*)
Bases: InvoiceGenerator.pdf.BaseInvoice

Generator of simple invoice in PDF format

Parameters **invoice** (Invoice) – the invoice

gen(*filename*, *generate_qr_code=False*)
Generate the invoice into file

Parameters

- **filename** (*string or File*) – file in which the PDF simple invoice will be written
- **generate_qr_code** (*boolean*) – should be QR code included in the PDF?

class InvoiceGenerator.pdf.**ProformaInvoice**(*invoice*)
Bases: *InvoiceGenerator.pdf.SimpleInvoice*

class InvoiceGenerator.pdf.**CorrectingInvoice**(*invoice*)
Bases: *InvoiceGenerator.pdf.SimpleInvoice*

drawCorretion(*TOP*, *LEFT*)

gen(*filename*)
Generate the invoice into file

Parameters **filename** (*string or File*) – file in which the PDF correcting invoice will be written

InvoiceGenerator.pohoda

CHAPTER 2

History

1.0.0 - 2017-09-10

- Add support for Pohoda XML format
- Added parameter Address.bank_code. If present, the bank code will be written after dash to the account number, otherwise whole Address.bank_account will be used.
- Added much more complex documentation
- Parameter `Address.zip` was renamed to `Address.zip_code`
- Code style fixes
- Fixes for rounding: usage of `decimal.Decimal` and added parameter `Invoice.rounding_strategy`

0.5.4 - 2017-03-22

- Fix locale in build package

0.5.3 - 2017-01-09

- Use Babel for currency formating; fix and improve tests

0.5.2 - 2014-12-04

- Stop mentionning python2.6 support
- Make invoice.variable_symbol optional

0.5.1 - 2014-10-28

- Fix conf relative import
- Use python native function splitlines for notes

0.5.0 - 2014-09-21

- Add property number to object Invoice
- Replaced variable symbol for invoice number in invoice header
- Update Czech translations

0.4.9 - 2014-07-3

- Bug fix previous commit

0.4.8 - 2014-07-3

- Create proforma invoice

0.4.7 - 2014-07-1

- Change date format for qr code generator
- Disable converting datetime to string on Invoice
- Disable rendering empty values

0.4.6 - 2014-05-14

- The displayed number of pages only when there is more than one
- Rename Date to Date of exposure
- Use pillow instead of PIL

0.4.5 - 2014-04-21

- Support for multipage items printout
- Support for multiline item description
- Use locale to print currency strings and values
- Adding logo to provider header

No notes on earlier releases.

CHAPTER 3

Contributors

- Adam Strauch (@creckx)
- Martin Voldrich (@rbas)
- Petr Dlouhy (@PetrDlouhy)
- Antoine Musso (@hashar)

CHAPTER 4

InvoiceGenerator

This is library to generate a simple invoices. Currently supported formats are PDF and XML for Pohoda accounting system. PDF invoice is based on ReportLab.

Installation

Run this command as root:

```
pip install InvoiceGenerator
```

If you want upgrade to new version, add `--upgrade` flag:

```
pip install InvoiceGenerator --upgrade
```

You can use setup.py from GitHub repository too:

```
python setup.py install
```

Documentation

Complete documentation is available on Read The Docs.

Example

Basic API

Define invoice data first:

```
import os

from tempfile import NamedTemporaryFile

from InvoiceGenerator.api import Invoice, Item, Client, Provider, Creator

# choose english as language
os.environ["INVOICE_LANG"] = "en"

client = Client('Client company')
provider = Provider('My company', bank_account='2600420569', bank_code='2010')
creator = Creator('John Doe')

invoice = Invoice(client, provider, creator)
invoice.currency_locale = 'en_US.UTF-8'
invoice.add_item(Item(32, 600, description="Item 1"))
invoice.add_item(Item(60, 50, description="Item 2", tax=21))
invoice.add_item(Item(50, 60, description="Item 3", tax=0))
invoice.add_item(Item(5, 600, description="Item 4", tax=15))
```

Note: Due to Python's representational error, write numbers as integer `tax=10`, Decimal `tax=Decimal('10.1')` or string `tax='1.2'` to avoid getting results with lot of decimal places.

PDF

Generate PDF invoice file:

```
from InvoiceGenerator.pdf import SimpleInvoice

pdf = SimpleInvoice(invoice)
pdf.gen("invoice.pdf", generate_qr_code=True)
```

Pohoda XML

Generate XML invoice file:

```
from InvoiceGenerator.pohoda import SimpleInvoice

pdf = SimpleInvoice(invoice)
pdf.gen("invoice.xml")
```

Note: Pohoda uses three tax rates: none: 0%, low: 15%, high: 21%. If any item doesn't meet those percentage, the rateVat parameter will not be set for those items resulting in 0% tax rate.

Only SimpleInvoice is currently supported for Pohoda XML format.

Hacking

Fork the repository on github and write code. Make sure to add tests covering your code under */tests/*. You can run tests using:

```
python setup.py test
```

Then propose your patch via a pull request.

Documentation is generated from *doc/source/* using Sphinx:

```
python setup.py build_sphinx
```

Then head to *doc/build/html/index.html*.

Indices and tables

- genindex
- modindex
- search

Python Module Index

i

InvoiceGenerator.api, 1
InvoiceGenerator.pdf, 4

Index

A

add_item() (InvoiceGenerator.api.Invoice method), 3
Address (class in InvoiceGenerator.api), 1

B

bank_account_str() (InvoiceGenerator.api.Address method), 2
bank_account_str() (InvoiceGenerator.api.Client method), 2
bank_account_str() (InvoiceGenerator.api.Provider method), 2

C

Client (class in InvoiceGenerator.api), 2
CorrectingInvoice (class in InvoiceGenerator.pdf), 4
count (InvoiceGenerator.api.Item attribute), 2
count_tax() (InvoiceGenerator.api.Item method), 2
Creator (class in InvoiceGenerator.api), 2
currency (InvoiceGenerator.api.Invoice attribute), 3
currency_locale (InvoiceGenerator.api.Invoice attribute), 3

D

date (InvoiceGenerator.api.Invoice attribute), 3
description (InvoiceGenerator.api.Item attribute), 2
difference_in_rounding (InvoiceGenerator.api.Invoice attribute), 3
drawCorretion() (InvoiceGenerator.pdf.CorrectingInvoice method), 4

G

gen() (InvoiceGenerator.pdf.CorrectingInvoice method), 4
gen() (InvoiceGenerator.pdf.SimpleInvoice method), 4
generate_breakdown_vat() (InvoiceGenerator.api.Invoice method), 3
generate_breakdown_vat_table() (InvoiceGenerator.api.Invoice method), 3

I

iban (InvoiceGenerator.api.Invoice attribute), 3
Invoice (class in InvoiceGenerator.api), 3
InvoiceGenerator.api (module), 1
InvoiceGenerator.pdf (module), 4
Item (class in InvoiceGenerator.api), 2
items (InvoiceGenerator.api.Invoice attribute), 3

N

number (InvoiceGenerator.api.Invoice attribute), 3

P

payback (InvoiceGenerator.api.Invoice attribute), 3
paytype (InvoiceGenerator.api.Invoice attribute), 3
price (InvoiceGenerator.api.Invoice attribute), 3
price (InvoiceGenerator.api.Item attribute), 2
price_tax (InvoiceGenerator.api.Invoice attribute), 3
ProformaInvoice (class in InvoiceGenerator.pdf), 4
Provider (class in InvoiceGenerator.api), 2

R

rounding_result (InvoiceGenerator.api.Invoice attribute), 3
rounding_strategy (InvoiceGenerator.api.Invoice attribute), 4

S

SimpleInvoice (class in InvoiceGenerator.pdf), 4
specific_symbol (InvoiceGenerator.api.Invoice attribute), 4
swift (InvoiceGenerator.api.Invoice attribute), 4

T

tax (InvoiceGenerator.api.Item attribute), 2
taxable_date (InvoiceGenerator.api.Invoice attribute), 4
title (InvoiceGenerator.api.Invoice attribute), 4
total (InvoiceGenerator.api.Item attribute), 3
total_tax (InvoiceGenerator.api.Item attribute), 3

U

unit (InvoiceGenerator.api.Item attribute), 3
use_tax (InvoiceGenerator.api.Invoice attribute), 4

V

variable_symbol (InvoiceGenerator.api.Invoice attribute), 4