

---

# **InvoiceGenerator Documentation**

**See CONTRIBUTORS.rst**

**May 17, 2018**



---

# Contents

---

<b>1</b>	<b>Modules</b>	<b>1</b>
1.1	InvoiceGenerator.api . . . . .	1
1.2	InvoiceGenerator.pdf . . . . .	4
1.3	InvoiceGenerator.pohoda . . . . .	4
<b>2</b>	<b>History</b>	<b>5</b>
2.1	0.6.0 - 2017-09-07 . . . . .	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
<b>3</b>	<b>Contributors</b>	<b>7</b>
<b>4</b>	<b>InvoiceGenerator</b>	<b>9</b>
4.1	Installation . . . . .	9
4.2	Example . . . . .	9
4.3	Hacking . . . . .	10
4.4	Indices and tables . . . . .	11
	<b>Python Module Index</b>	<b>13</b>



InvoiceGenerator is made of four submodules:

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

## 1.1 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 provider (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 (usually 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 paid 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

## 1.2 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

## 1.3 InvoiceGenerator.pohoda



### 2.1 0.6.0 - 2017-09-07

- 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`

### 2.2 0.5.4 - 2017-03-22

- Fix locale in build package

### 2.3 0.5.3 - 2017-01-09

- Use Babel for currency formating; fix and improve tests

### 2.4 0.5.2 - 2014-12-04

- Stop mentioning python2.6 support
- Make `invoice.variable_symbol` optional

### 2.5 0.5.1 - 2014-10-28

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

### 2.6 0.5.0 - 2014-09-21

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

### 2.7 0.4.9 - 2014-07-3

- Bug fix previous commit

### 2.8 0.4.8 - 2014-07-3

- Create proforma invoice

### 2.9 0.4.7 - 2014-07-1

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

### 2.10 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

### 2.11 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)



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.

### 4.1 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
```

#### 4.1.1 Documentation

Complete documentation is available on [Read The Docs](#).

### 4.2 Example

#### 4.2.1 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.

### 4.2.2 PDF

Generate PDF invoice file:

```
from InvoiceGenerator.pdf import SimpleInvoice

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

### 4.2.3 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.

## 4.3 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*.

## 4.4 Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)





i

InvoiceGenerator.api, 1

InvoiceGenerator.pdf, 4



**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