abcFinance Documentation

Release 0.1

Davoud Toghawi-Nejad / Christoph Siebenbrunner / Maarten Scho

Apr 30, 2018

Contents:

1 accounting module 1

2 contractAPI module 3

3 Indices and tables 5

Python Module Index 7

CHAPTER 1

accounting module

abcFinance is an implementation of an double entry book keeping system

Initialize the accounting system with, the name of the residual_account:

```
accounts = AccountingSystem('equity')
```

Create stock and flow account:

accounts.make_stock_account([‘cash’, ‘claims’]) accounts.make_flow_account([‘expenditure’])

In order to book give a list of credit and debit tuples. Each tuple should be an account and a value:

```
accounts.book(
    debit=[('cash', 50), ('claims', 50)],
    credit=[('equity', 100)])
```

get balance gives you the balance of an account:

assert accounts[‘cash’].get_balance() == (s.DEBIT, 50)

Balance sheet

accounts.book(debit=[(‘expenditure’, 20)], credit=[(‘cash’, 20)])

assert accounts.get_total_assets() == 80, accounts.get_total_assets()

accounts.print_profit_and_loss() print(‘–’) accounts.make_end_of_period()

accounts.print_profit_and_loss()

accounts.print_balance_sheet()

assert accounts[‘equity’].get_balance() == (s.CREDIT, 80)

class accounting.**Account**

Bases: object

get_balance()

print_balance()

class accounting.**AccountingSystem**(*residual_account_name='equity'*)

Bases: object

The main class to be initialized

book(*debit*, *credit*, *text=''*)

Book a transaction.

Arguments: debit, list of tuples ('account', amount)

credit, list of tuples ('account', amount)

text, for booking history

Example:

```
accounts.book(debit=[('inventory',20)], credit=[('cash',20)], text="Purchase␣
↪of equipment")
```

get_total_assets()

Return total assets.

make_end_of_period()

Close flow accounts and credit/debit residual (equity) account

make_flow_account(*names*)

Create flow accounts.

Args: names, list of names for the accounts

make_stock_account(*names*)

Create stock accounts.

Args: names, list of names for the accounts

print_balance_sheet()

Print a balance sheets

print_profit_and_loss()

Print profit and loss statement

class accounting.**s**

Bases: enum.Enum

An enumeration.

CREDIT = 1

DEBIT = 0

CHAPTER 2

contractAPI module

abcFinance specifies a contract API. This allows the user to use the full python language to specify contracts.

Both / all contract parties get a copy of the contract.

class contractAPI.**Action**(*action*, *more_info=None*)

Bases: object

An action has to properties. The action to be executed and further information. This could be for example whether the action is obligatory or not.

Example:

```
Action(('pay', 50, 'UBS'), 'compulsory')
```

The action property and more_info, can have any content the tuple ('pay', 50, 'UBV') is just and example.

action = None

self.action, action to be executed,

more_info = None

self.more_info, further information

class contractAPI.**Contract**(*issuing_party*, *_, **__)

Bases: object

A contract is a tree or sequence of `Action`s. A contract must implement :method:`get_action`, which gives the next action given the current time and relevant state. Further it must implement **:method:'action_executed'**, which is lets the contract know that an action has been is executed.

Optionally is_terminanted should return whether the contract is terminated.

In order to use a contract with a book keeping system **:method:'valuation'** needs to return the valuation

action_executed(*obligation*)

Let the contract know that a particular obligation has been executed. This is used by the agent that execute the obligation on his copy and by the receiving agent on his copy of the contract.

Example:

```
self.num_excecuted_payments += 1
```

get_actions(*party*, *time*, *_, **__)
Returns a list of actions that have to or could be executed now arguments are the current time and any state informations that are necessary to decide which actions have to could be taken.

Example:

```
if self.issuing_party == party:
    if self.num_excecuted_payments < self.number_of_payments_required:
        return [Action(('pay', self.amount), 'at_least_3_times')]
else:
    return []
```

is_terminanted()
Whether the contract is terminated.

Example:

```
if.num_excecuted_payments >= self.number_of_payments_required:
    return True
else:
    return False
```

last_valuation()

sign(*party*)
Signs the contract and sets the self.counter_party property

valuation(*party*, *time*, *_, **__)
returns a valuation of the contract. Arguments:

party, the party for whom the valuation is made time, current time other state variables necessary for the valuation

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

accounting, 1

c

contractAPI, 3

Index

A

Account (class in accounting), 1
accounting (module), 1
AccountingSystem (class in accounting), 1
Action (class in contractAPI), 3
action (contractAPI.Action attribute), 3
action_executed() (contractAPI.Contract method), 3

B

book() (accounting.AccountingSystem method), 2

C

Contract (class in contractAPI), 3
contractAPI (module), 3
CREDIT (accounting.s attribute), 2

D

DEBIT (accounting.s attribute), 2

G

get_actions() (contractAPI.Contract method), 4
get_balance() (accounting.Account method), 1
get_total_assets() (accounting.AccountingSystem method), 2

I

is_terminanted() (contractAPI.Contract method), 4

L

last_valuation() (contractAPI.Contract method), 4

M

make_end_of_period() (accounting.AccountingSystem method), 2
make_flow_account() (accounting.AccountingSystem method), 2
make_stock_account() (accounting.AccountingSystem method), 2
more_info (contractAPI.Action attribute), 3

P

print_balance() (accounting.Account method), 1
print_balance_sheet() (accounting.AccountingSystem method), 2
print_profit_and_loss() (accounting.AccountingSystem method), 2

S

s (class in accounting), 2
sign() (contractAPI.Contract method), 4

V

valuation() (contractAPI.Contract method), 4