
Mia! Accounting

Release 1.5.11

imacat

Dec 16, 2023

CONTENTS:

1	Introduction	1
1.1	Live Demonstration and Test Site	1
1.2	Installation	1
1.3	Prerequisites	1
1.4	Configuration	2
1.5	Database Initialization	2
1.6	Navigation Menu	2
2	accounting package	3
2.1	Subpackages	3
2.2	Submodules	82
2.3	accounting.commands module	82
2.4	accounting.forms module	82
2.5	accounting.locale module	82
2.6	accounting.models module	84
2.7	accounting.template_filters module	92
2.8	accounting.template_globals module	93
2.9	Module contents	93
3	Examples	95
3.1	An Example Configuration	95
4	History	97
5	Change Log	99
5.1	Version 1.5.11	99
5.2	Version 1.5.10	99
5.3	Version 1.5.9	99
5.4	Version 1.5.8	99
5.5	Version 1.5.7	100
5.6	Version 1.5.6	100
5.7	Version 1.5.5	100
5.8	Version 1.5.4	100
5.9	Version 1.5.3	101
5.10	Version 1.5.2	101
5.11	Version 1.5.1	101
5.12	Version 1.5.0	101
5.13	Version 1.4.1	101
5.14	Version 1.4.0	102
5.15	Version 1.3.3	102

5.16	Version 1.3.2	102
5.17	Version 1.3.1	102
5.18	Version 1.3.0	102
5.19	Version 1.2.1	103
5.20	Version 1.2.0	103
5.21	Version 1.1.0	103
5.22	Version 1.0.1	103
5.23	Version 1.0.0	103
5.24	Version 0.11.1 (Pre-release)	103
5.25	Version 0.11.0 (Pre-release)	104
5.26	Version 0.10.0 (Pre-release)	104
5.27	Version 0.9.1 (Pre-release)	104
5.28	Version 0.9.0 (Pre-release)	104
5.29	Version 0.8.0 (Pre-release)	105
5.30	Version 0.7.0 (Pre-release)	105
5.31	Version 0.6.0 (Pre-release)	105
5.32	Version 0.5.0 (Pre-release)	105
5.33	Version 0.4.0 (Pre-release)	105
5.34	Version 0.3.1 (Pre-release)	106
5.35	Version 0.3.0 (Pre-release)	106
5.36	Version 0.2.0 (Pre-release)	106
5.37	Version 0.1.1 (Pre-release)	106
5.38	Version 0.1.0 (Pre-release)	106
5.39	Version 0.0.0 (Pre-release)	106
6	Indices and tables	107
	Python Module Index	109
	Index	111

INTRODUCTION

Mia! Accounting is an accounting module for [Flask](#) applications. It is designed both for mobile and desktop environments. It implements [double-entry bookkeeping](#). It generates the following accounting reports:

- Trial balance
- Income statement
- Balance sheet

In addition, *Mia! Accounting* tracks offsets for unpaid payables and receivables.

1.1 Live Demonstration and Test Site

There is a [live demonstration](#) for *Mia! Accounting*. It runs the same code as the [test site](#) in the [source distribution](#). It is the simplest website that works with *Mia! Accounting*. It is also used in the automatic tests.

If you do not have a running Flask application or do not know how to start one, you may start with the test site.

1.2 Installation

Install *Mia! Accounting* with `pip`:

```
pip install mia-accounting
```

You may also download from the [PyPI project page](#) or the [release page](#) on the [Git repository](#).

1.3 Prerequisites

You need a running Flask application with database user login. The primary key of the user data model must be integer. You also need at least one user.

The following front-end JavaScript libraries must be loaded. You may download it locally or use [CDN](#).

- [Bootstrap](#) 5.2.3 or above
- [FontAwesome](#) 6.4.0 or above
- [decimal.js](#) 10.4.3 or above, or [decimal.js-light](#) 2.5.1 or above.
- [Tempus-Dominus](#) 6.7.7 or above

1.4 Configuration

You need to pass the Flask *app* and an implementation of `accounting.utils.user.UserUtilityInterface` to the `accounting.init_app()` function. `UserUtilityInterface` contains everything *Mia! Accounting* needs.

See an example in *An Example Configuration*.

1.5 Database Initialization

After the configuration, run the `accounting-init-db` console command to initialize the accounting database. You need to specify the username of a user as the data creator.

```
% flask --app myapp accounting-init-db -u username
```

1.6 Navigation Menu

Include the navigation menu in the [Bootstrap navigation bar](#) in your base template:

```
<nav class="navbar navbar-expand-lg bg-body-tertiary bg-dark navbar-dark">
  <div class="container-fluid">
    ...
    <div id="collapsible-navbar" class="collapse navbar-collapse">
      <ul class="navbar-nav me-auto mb-2 mb-lg-0">
        ...
        {% include "accounting/include/nav.html" %}
        ...
      </ul>
    </div>
  </div>
</nav>
```

Check your Flask application and see how it works.

ACCOUNTING PACKAGE

2.1 Subpackages

2.1.1 `accounting.account` package

Submodules

`accounting.account.commands` module

The console commands for the account management.

`accounting.account.commands.AccountData`

The format of the account data, as a list of (ID, base account code, number, English, Traditional Chinese, Simplified Chinese, is-need-offset) tuples.

alias of `tuple[int, str, int, str, str, str, bool]`

`accounting.account.commands.init_accounts_command(username: str) → None`

Initializes the accounts.

`accounting.account.converters` module

The path converters for the account management.

`class accounting.account.converters.AccountConverter(map: Map, *args: t.Any, **kwargs: t.Any)`

Bases: `BaseConverter`

The account converter to convert the account code and to the corresponding account in the routes.

`to_python(value: str) → Account`

Converts an account code to an account.

Parameters

value – The account code.

Returns

The corresponding account.

`to_url(value: Account) → str`

Converts an account to its code.

Parameters

value – The account.

Returns

The code.

accounting.account.forms module

The forms for the account management.

```
class accounting.account.forms.AccountForm(*args, **kwargs)
```

Bases: FlaskForm

The form to create or edit an account.

```
base_code = <UnboundField(StringField, (), {'filters': [<function strip_text>],
'validators': [<wtforms.validators.DataRequired object>,
<accounting.account.forms.BaseAccountExists object>,
<accounting.account.forms.BaseAccountAvailable object>]})>
```

The code of the base account.

```
property base_options: list[BaseAccount]
```

The selectable base accounts.

Returns

The selectable base accounts.

```
is_need_offset = <UnboundField(BooleanField, (), {'validators':
[<accounting.account.forms.NoOffsetNominalAccount object>]})>
```

Whether the the journal entry line items of this account need offset.

```
populate_obj(obj: Account) → None
```

Populates the form data into an account object.

Parameters

obj – The account object.

Returns

None.

```
post_update(obj: Account) → None
```

The post-processing after the update.

Parameters

obj – The account object.

Returns

None

```
property selected_base: BaseAccount | None
```

The selected base account in the form.

Returns

The selected base account in the form.

```
title = <UnboundField(StringField, (), {'filters': [<function strip_text>],
'validators': [<wtforms.validators.DataRequired object>]})>
```

The title.

```
class accounting.account.forms.AccountReorderForm(base: BaseAccount)
```

Bases: object

The form to reorder the accounts.

base: *BaseAccount*

The base account.

is_modified: **bool**

Whether the order is modified.

save_order() → None

Saves the order of the account.

Returns

class `accounting.account.forms.BaseAccountAvailable`

Bases: `object`

The validator to check if the base account is available.

class `accounting.account.forms.BaseAccountExists`

Bases: `object`

The validator to check if the base account exists.

class `accounting.account.forms.NoOffsetNominalAccount`

Bases: `object`

The validator to check nominal account is not to be offset.

`accounting.account.forms.sort_accounts_in(base_code: str, exclude: int)` → None

Sorts the accounts under a base account after changing the base account or deleting an account.

Parameters

- **base_code** – The code of the base account.
- **exclude** – The account ID to exclude.

Returns

None.

`accounting.account.queries` module

The queries for the account management.

`accounting.account.queries.get_account_query()` → list[*Account*]

Returns the accounts, optionally filtered by the query.

Returns

The accounts.

`accounting.account.views` module

The views for the account management.

`accounting.account.views.bp: Blueprint = <Blueprint 'account'>`

The view blueprint for the account management.

Module contents

The account management.

`accounting.account.init_app(app: Flask, bp: Blueprint) → None`

Initialize the application.

Parameters

- **app** – The Flask application.
- **bp** – The blueprint of the accounting application.

Returns

None.

2.1.2 accounting.base_account package

Submodules

accounting.base_account.commands module

The console commands for the base account management.

`accounting.base_account.commands.init_base_accounts_command() → None`

Initializes the base accounts.

accounting.base_account.converters module

The path converters for the base account management.

`class accounting.base_account.converters.BaseAccountConverter(map: Map, *args: t.Any, **kwargs: t.Any)`

Bases: `BaseConverter`

The account converter to convert the account code and to the corresponding base account in the routes.

`to_python(value: str) → BaseAccount`

Converts an account code to a base account.

Parameters

value – The account code.

Returns

The corresponding base account.

`to_url(value: BaseAccount) → str`

Converts a base account to its code.

Parameters

value – The base account.

Returns

The code.

accounting.base_account.queries module

The queries for the base account management.

`accounting.base_account.queries.get_base_account_query()` → list[*BaseAccount*]

Returns the base accounts, optionally filtered by the query.

Returns

The base accounts.

accounting.base_account.views module

The views for the base account management.

`accounting.base_account.views.bp`: **Blueprint** = <Blueprint 'base-account'>

The view blueprint for the base account management.

Module contents

The base account management.

`accounting.base_account.init_app(app: Flask, bp: Blueprint)` → None

Initialize the application.

Parameters

- **app** – The Flask application.
- **bp** – The blueprint of the accounting application.

Returns

None.

2.1.3 accounting.currency package

Submodules

accounting.currency.commands module

The console commands for the currency management.

`accounting.currency.commands.init_currencies_command(username: str)` → None

Initializes the currencies.

accounting.currency.converters module

The path converters for the currency management.

class `accounting.currency.converters.CurrencyConverter`(*map: Map, *args: t.Any, **kwargs: t.Any*)

Bases: `BaseConverter`

The currency converter to convert the currency code and to the corresponding currency in the routes.

to_python(*value: str*) → *Currency*

Converts a currency code to a currency.

Parameters

value – The currency code.

Returns

The corresponding currency.

to_url(*value: Currency*) → *str*

Converts a currency to its code.

Parameters

value – The currency.

Returns

The code.

accounting.currency.forms module

The forms for the currency management.

class `accounting.currency.forms.CodeUnique`

Bases: `object`

The validator to check if the code is unique.

class `accounting.currency.forms.CurrencyForm(*args, **kwargs)`

Bases: `FlaskForm`

The form to create or edit a currency.

CODE_BLOCKLIST: `list[str] = ['create', 'store', 'exists-code']`

The reserved codes that are not available.

```
code = <UnboundField(StringField, (), {'filters': [<function strip_text>],
'validators': [<wtforms.validators.DataRequired object>, <wtforms.validators.Regexp
object>, <wtforms.validators.NoneOf object>, <accounting.currency.forms.CodeUnique
object>]})>
```

The code. It may not conflict with another currency.

```
name = <UnboundField(StringField, (), {'filters': [<function strip_text>],
'validators': [<wtforms.validators.DataRequired object>]})>
```

The name.

obj_code: `str | None`

The current code of the currency, or None when adding a new currency.

populate_obj(*obj: Currency*) → *None*

Populates the form data into a currency object.

Parameters

obj – The currency object.

Returns

None.

accounting.currency.queries module

The queries for the currency management.

`accounting.currency.queries.get_currency_query()` → list[*Currency*]

Returns the base accounts, optionally filtered by the query.

Returns

The base accounts.

accounting.currency.views module

The views for the currency management.

`accounting.currency.views.api_bp: Blueprint = <Blueprint 'currency-api'>`

The view blueprint for the currency management API.

`accounting.currency.views.bp: Blueprint = <Blueprint 'currency'>`

The view blueprint for the currency management.

Module contents

The currency management.

`accounting.currency.init_app(app: Flask, bp: Blueprint)` → None

Initialize the application.

Parameters

- **app** – The Flask application.
- **bp** – The blueprint of the accounting application.

Returns

None.

2.1.4 accounting.journal_entry package

Subpackages

accounting.journal_entry.forms package

Submodules

accounting.journal_entry.forms.currency module

The currency sub-forms for the journal entry management.

`accounting.journal_entry.forms.currency.CURRENCY_REQUIRED: DataRequired = <wtforms.validators.DataRequired object>`

The validator to check if the currency code is empty.

```
class accounting.journal_entry.forms.currency.CashDisbursementCurrencyForm(*args, **kwargs)
```

Bases: *CurrencyForm*

The form to create or edit a currency in a cash disbursement journal entry.

```
code = <UnboundField(StringField, (), {'filters': [<function strip_text>],
'validators': [<wtforms.validators.DataRequired object>,
<accounting.forms.CurrencyExists object>,
<accounting.journal_entry.forms.currency.SameCurrencyAsOriginalLineItems object>,
<accounting.journal_entry.forms.currency.KeepCurrencyWhenHavingOffset object>]})>
```

The currency code.

```
debit = <UnboundField(FieldList, (<UnboundField(FormField, (<class
'accounting.journal_entry.forms.line_item.DebitLineItemForm'>), {}>),),
{'validators': [<accounting.journal_entry.forms.currency.NeedSomeLineItems
object>]})>
```

The debit line items.

```
property debit_errors: list[str | LazyString]
```

Returns the debit line item errors, without the errors in their sub-forms.

Returns

```
property debit_total: Decimal
```

Returns the total amount of the debit line items.

Returns

The total amount of the debit line items.

```
no = <UnboundField(IntegerField, (), {})>
```

The order in the journal entry.

```
whole_form = <UnboundField(BooleanField, (), {})>
```

The pseudo field for the whole form validators.

```
class accounting.journal_entry.forms.currency.CashReceiptCurrencyForm(*args, **kwargs)
```

Bases: *CurrencyForm*

The form to create or edit a currency in a cash receipt journal entry.

```
code = <UnboundField(StringField, (), {'filters': [<function strip_text>],
'validators': [<wtforms.validators.DataRequired object>,
<accounting.forms.CurrencyExists object>,
<accounting.journal_entry.forms.currency.SameCurrencyAsOriginalLineItems object>,
<accounting.journal_entry.forms.currency.KeepCurrencyWhenHavingOffset object>]})>
```

The currency code.

```
credit = <UnboundField(FieldList, (<UnboundField(FormField, (<class
'accounting.journal_entry.forms.line_item.CreditLineItemForm'>), {}>),),
{'validators': [<accounting.journal_entry.forms.currency.NeedSomeLineItems
object>]})>
```

The credit line items.

```
property credit_errors: list[str | LazyString]
```

Returns the credit line item errors, without the errors in their sub-forms.

Returns

property credit_total: Decimal

Returns the total amount of the credit line items.

Returns

The total amount of the credit line items.

no = <UnboundField(IntegerField, (), {})>

The order in the journal entry.

whole_form = <UnboundField(BooleanField, (), {})>

The pseudo field for the whole form validators.

class `accounting.journal_entry.forms.currency.CurrencyForm(*args, **kwargs)`

Bases: `FlaskForm`

The form to create or edit a currency in a journal entry.

code = <UnboundField(StringField, (), {})>

The currency code.

property is_code_locked: bool

Returns whether the currency code should not be changed.

Returns

True if the currency code should not be changed, or False otherwise

property line_items: list[LineItemForm]

Returns the line item sub-forms.

Returns

The line item sub-forms.

no = <UnboundField(IntegerField, (), {})>

The order in the journal entry.

whole_form = <UnboundField(BooleanField, (), {})>

The pseudo field for the whole form validators.

class `accounting.journal_entry.forms.currency.IsBalanced`

Bases: `object`

The validator to check that the total amount of the debit and credit line items are equal.

class `accounting.journal_entry.forms.currency.KeepCurrencyWhenHavingOffset`

Bases: `object`

The validator to check if the currency is the same when there is offset.

class `accounting.journal_entry.forms.currency.NeedSomeLineItems`

Bases: `object`

The validator to check if there is any line item sub-form.

class `accounting.journal_entry.forms.currency.SameCurrencyAsOriginalLineItems`

Bases: `object`

The validator to check if the currency is the same as the original line items.

class `accounting.journal_entry.forms.currency.TransferCurrencyForm(*args, **kwargs)`

Bases: `CurrencyForm`

The form to create or edit a currency in a transfer journal entry.

```
code = <UnboundField(StringField, (), {'filters': [<function strip_text>],
'validators': [<wtforms.validators.DataRequired object>,
<accounting.forms.CurrencyExists object>,
<accounting.journal_entry.forms.currency.SameCurrencyAsOriginalLineItems object>,
<accounting.journal_entry.forms.currency.KeepCurrencyWhenHavingOffset object>]})>
```

The currency code.

```
credit = <UnboundField(FieldList, (<UnboundField(FormField, (<class
'accounting.journal_entry.forms.line_item.CreditLineItemForm'>,), {}>),),
{'validators': [<accounting.journal_entry.forms.currency.NeedSomeLineItems
object>]})>
```

The credit line items.

```
property credit_errors: list[str | LazyString]
```

Returns the credit line item errors, without the errors in their sub-forms.

Returns

```
property credit_total: Decimal
```

Returns the total amount of the credit line items.

Returns

The total amount of the credit line items.

```
debit = <UnboundField(FieldList, (<UnboundField(FormField, (<class
'accounting.journal_entry.forms.line_item.DebitLineItemForm'>,), {}>),),
{'validators': [<accounting.journal_entry.forms.currency.NeedSomeLineItems
object>]})>
```

The debit line items.

```
property debit_errors: list[str | LazyString]
```

Returns the debit line item errors, without the errors in their sub-forms.

Returns

```
property debit_total: Decimal
```

Returns the total amount of the debit line items.

Returns

The total amount of the debit line items.

```
no = <UnboundField(IntegerField, (), {})>
```

The order in the journal entry.

```
whole_form = <UnboundField(BooleanField, (), {'validators':
[<accounting.journal_entry.forms.currency.IsBalanced object>]})>
```

The pseudo field for the whole form validators.

accounting.journal_entry.forms.journal_entry module

The journal entry forms for the journal entry management.

class

```
accounting.journal_entry.forms.journal_entry.CannotDeleteOriginalLineItemsWithOffset
```

Bases: object

The validator to check the original line items with offset.

```
class accounting.journal_entry.forms.journal_entry.CashDisbursementJournalEntryForm(*args,
                                                                                   **kwargs)
```

Bases: *JournalEntryForm*

The form to create or edit a cash disbursement journal entry.

```
currencies = <UnboundField(FieldList, (<UnboundField(FormField, (<class
'accounting.journal_entry.forms.currency.CashDisbursementCurrencyForm'>)), {}>)),
{'validators': [<accounting.journal_entry.forms.journal_entry.NeedSomeCurrencies
object>]})>
```

The line items categorized by their currencies.

```
date = <UnboundField(DateField, (), {'validators':
[<wtforms.validators.DataRequired object>,
<accounting.journal_entry.forms.journal_entry.NotBeforeOriginalLineItems object>,
<accounting.journal_entry.forms.journal_entry.NotAfterOffsetItems object>]})>
```

The date.

```
note = <UnboundField(TextAreaField, (), {'filters': [<function
strip_multiline_text>]})>
```

The note.

```
whole_form = <UnboundField(BooleanField, (), {'validators': [<accounting.
journal_entry.forms.journal_entry.CannotDeleteOriginalLineItemsWithOffset
object>]})>
```

The pseudo field for the whole form validators.

```
class accounting.journal_entry.forms.journal_entry.CashReceiptJournalEntryForm(*args,
                                                                                   **kwargs)
```

Bases: *JournalEntryForm*

The form to create or edit a cash receipt journal entry.

```
currencies = <UnboundField(FieldList, (<UnboundField(FormField, (<class
'accounting.journal_entry.forms.currency.CashReceiptCurrencyForm'>)), {}>)),
{'validators': [<accounting.journal_entry.forms.journal_entry.NeedSomeCurrencies
object>]})>
```

The line items categorized by their currencies.

```
date = <UnboundField(DateField, (), {'validators':
[<wtforms.validators.DataRequired object>,
<accounting.journal_entry.forms.journal_entry.NotBeforeOriginalLineItems object>,
<accounting.journal_entry.forms.journal_entry.NotAfterOffsetItems object>]})>
```

The date.

```
note = <UnboundField(TextAreaField, (), {'filters': [<function
strip_multiline_text>]})>
```

The note.

```
whole_form = <UnboundField(BooleanField, (), {'validators': [<accounting.
journal_entry.forms.journal_entry.CannotDeleteOriginalLineItemsWithOffset
object>]})>
```

The pseudo field for the whole form validators.

```
accounting.journal_entry.forms.journal_entry.DATE_REQUIRED: DataRequired =
<wtforms.validators.DataRequired object>
```

The validator to check if the date is empty.

```
class accounting.journal_entry.forms.journal_entry.JournalEntryForm(*args, **kwargs)
```

Bases: FlaskForm

The base form to create or edit a journal entry.

```
collector: Type[LineItemCollector]
```

The line item collector. The default is the base abstract collector only to provide the correct type. The subclass forms should provide their own collectors.

```
property credit_account_options: list[AccountOption]
```

The selectable credit accounts.

Returns

The selectable credit accounts.

```
currencies = <UnboundField(FieldList, (<UnboundField(FormField, (<class
'accounting.journal_entry.forms.currency.CurrencyForm'>,), {}>,), {}>
```

The line items categorized by their currencies.

```
property currencies_errors: list[str | LazyString]
```

Returns the currency errors, without the errors in their sub-forms.

Returns

The currency errors, without the errors in their sub-forms.

```
date = <UnboundField(DateField, (), {}>
```

The date.

```
property debit_account_options: list[AccountOption]
```

The selectable debit accounts.

Returns

The selectable debit accounts.

```
property description_editor: DescriptionEditor
```

Returns the description editor.

Returns

The description editor.

```
is_modified: bool
```

Whether the journal entry is modified during populate_obj().

```
property line_items: list[LineItemForm]
```

Collects and returns the line item sub-forms.

Returns

The line item sub-forms.

property max_date: `date | None`

Returns the maximum available date.

Returns

The maximum available date.

property min_date: `date | None`

Returns the minimal available date.

Returns

The minimal available date.

note = `<UnboundField(TextAreaField, (), {})>`

The note.

obj: `JournalEntry | None`

The journal entry, when editing an existing one.

property original_line_item_options: `list[JournalEntryLineItem]`

Returns the selectable original line items.

Returns

The selectable original line items.

populate_obj(*obj*: `JournalEntry`) → `None`

Populates the form data into a journal entry object.

Parameters

obj – The journal entry object.

Returns

`None`.

class `accounting.journal_entry.forms.journal_entry.LineItemCollector`(*form*: `T`, *obj*: `JournalEntry`)

Bases: `Generic[T], ABC`

The line item collector.

abstract collect() → `set[int]`

Collects the line items.

Returns

The ID of the line items to keep.

form: `T`

The journal entry form.

to_keep: `set[int]`

The ID of the existing line items to keep.

class `accounting.journal_entry.forms.journal_entry.NeedSomeCurrencies`

Bases: `object`

The validator to check if there is any currency sub-form.

class `accounting.journal_entry.forms.journal_entry.NotAfterOffsetItems`

Bases: `object`

The validator to check if the date is not after the offset items.

```
class accounting.journal_entry.forms.journal_entry.NotBeforeOriginalLineItems
```

Bases: object

The validator to check if the date is not before the original line items.

```
class accounting.journal_entry.forms.journal_entry.T
```

A journal entry form variant.

alias of TypeVar('T', bound=*JournalEntryForm*)

```
class accounting.journal_entry.forms.journal_entry.TransferJournalEntryForm(*args,  
                                                                           **kwargs)
```

Bases: *JournalEntryForm*

The form to create or edit a transfer journal entry.

```
currencies = <UnboundField(FieldList, (<UnboundField(FormField, (<class  
'accounting.journal_entry.forms.currency.TransferCurrencyForm'>,), {}>)),  
{'validators': [<accounting.journal_entry.forms.journal_entry.NeedSomeCurrencies  
object>]})>
```

The line items categorized by their currencies.

```
date = <UnboundField(DateField, (), {'validators':  
[<wtforms.validators.DataRequired object>,  
<accounting.journal_entry.forms.journal_entry.NotBeforeOriginalLineItems object>,  
<accounting.journal_entry.forms.journal_entry.NotAfterOffsetItems object>]})>
```

The date.

```
note = <UnboundField(TextAreaField, (), {'filters': [<function  
strip_multiline_text>]})>
```

The note.

```
whole_form = <UnboundField(BooleanField, (), {'validators': [<accounting.  
journal_entry.forms.journal_entry.CannotDeleteOriginalLineItemsWithOffset  
object>]})>
```

The pseudo field for the whole form validators.

accounting.journal_entry.forms.line_item module

The line item sub-forms for the journal entry management.

```
class accounting.journal_entry.forms.line_item.CreditLineItemForm(*args, **kwargs)
```

Bases: *LineItemForm*

The form to create or edit a credit line item.

```
account_code = <UnboundField(StringField, (), {'filters': [<function strip_text>],  
'validators': [<wtforms.validators.DataRequired object>,  
<accounting.forms.AccountExists object>,<accounting.forms.IsCreditAccount object>,  
<accounting.journal_entry.forms.line_item.SameAccountAsOriginalLineItem object>,  
<accounting.journal_entry.forms.line_item.KeepAccountWhenHavingOffset object>,  
<accounting.journal_entry.forms.line_item.NotStartReceivableFromCredit object>]})>
```

The account code.

```
amount = <UnboundField(DecimalField, (), {'validators':
[<accounting.journal_entry.forms.line_item.PositiveAmount object>,
<accounting.journal_entry.forms.line_item.NotExceedingOriginalLineItemNetBalance
object>, <accounting.journal_entry.forms.line_item.NotLessThanOffsetTotal
object>]})>
```

The amount.

```
description = <UnboundField(StringField, (), {'filters': [<function strip_text>]})>
```

The description.

```
id = <UnboundField(IntegerField, (), {})>
```

The existing line item ID.

```
no = <UnboundField(IntegerField, (), {})>
```

The order in the currency.

```
original_line_item_id = <UnboundField(IntegerField, (), {'validators':
[<wtforms.validators.Optional object>,
<accounting.journal_entry.forms.line_item.OriginalLineItemExists object>,
<accounting.journal_entry.forms.line_item.OriginalLineItemOppositeDebitCredit
object>, <accounting.journal_entry.forms.line_item.OriginalLineItemNeedOffset
object>, <accounting.journal_entry.forms.line_item.OriginalLineItemNotOffset
object>]})>
```

The ID of the original line item.

```
populate_obj(obj: JournalEntryLineItem) → None
```

Populates the form data into a line item object.

Parameters

obj – The line item object.

Returns

None.

```
class accounting.journal_entry.forms.line_item.DebitLineItemForm(*args, **kwargs)
```

Bases: [LineItemForm](#)

The form to create or edit a debit line item.

```
account_code = <UnboundField(StringField, (), {'filters': [<function strip_text>],
'validators': [<wtforms.validators.DataRequired object>,
<accounting.forms.AccountExists object>, <accounting.forms.IsDebitAccount object>,
<accounting.journal_entry.forms.line_item.SameAccountAsOriginalLineItem object>,
<accounting.journal_entry.forms.line_item.KeepAccountWhenHavingOffset object>,
<accounting.journal_entry.forms.line_item.NotStartPayableFromDebit object>]})>
```

The account code.

```
amount = <UnboundField(DecimalField, (), {'validators':
[<accounting.journal_entry.forms.line_item.PositiveAmount object>,
<accounting.journal_entry.forms.line_item.NotExceedingOriginalLineItemNetBalance
object>, <accounting.journal_entry.forms.line_item.NotLessThanOffsetTotal
object>]})>
```

The amount.

```
description = <UnboundField(StringField, (), {'filters': [<function strip_text>]})>
```

The description.

`id = <UnboundField(IntegerField, (), {})>`

The existing line item ID.

`no = <UnboundField(IntegerField, (), {})>`

The order in the currency.

`original_line_item_id = <UnboundField(IntegerField, (), {'validators':
[<wtforms.validators.Optional object>,
<accounting.journal_entry.forms.line_item.OriginalLineItemExists object>,
<accounting.journal_entry.forms.line_item.OriginalLineItemOppositeDebitCredit
object>, <accounting.journal_entry.forms.line_item.OriginalLineItemNeedOffset
object>, <accounting.journal_entry.forms.line_item.OriginalLineItemNotOffset
object>]})>`

The ID of the original line item.

`populate_obj(obj: JournalEntryLineItem) → None`

Populates the form data into a line item object.

Parameters

`obj` – The line item object.

Returns

None.

`class accounting.journal_entry.forms.line_item.KeepAccountWhenHavingOffset`

Bases: object

The validator to check if the account is the same when having offset.

`class accounting.journal_entry.forms.line_item.LineItemForm(*args, **kwargs)`

Bases: FlaskForm

The base form to create or edit a line item.

`account_code = <UnboundField(StringField, (), {})>`

The account code.

`property account_text: str`

Returns the text representation of the account.

Returns

The text representation of the account.

`property account_title: str`

Returns the title of the account.

Returns

The title of the account.

`property all_errors: list[str | LazyString]`

Returns all the errors of the form.

Returns

All the errors of the form.

`amount = <UnboundField(DecimalField, (), {})>`

The amount.

`description = <UnboundField(StringField, (), {})>`

The description.

id = <UnboundField(IntegerField, (), {})>

The existing line item ID.

property is_need_offset: bool

Returns whether the line item needs offset.

Returns

True if the line item needs offset, or False otherwise.

journal_entry_form: *JournalEntryForm* | None

The source journal entry form.

property net_balance: Decimal | None

Returns the net balance.

Returns

The net balance.

no = <UnboundField(IntegerField, (), {})>

The order in the currency.

property offset_total: Decimal | None

Returns the total amount of the offsets.

Returns

The total amount of the offsets.

property offsets: list[*JournalEntryLineItem*]

Returns the offsets.

Returns

The offsets.

property original_line_item_date: date | None

Returns the text representation of the original line item.

Returns

The text representation of the original line item.

original_line_item_id = <UnboundField(IntegerField, (), {})>

The Id of the original line item.

property original_line_item_text: str | None

Returns the text representation of the original line item.

Returns

The text representation of the original line item.

class accounting.journal_entry.forms.line_item.**NotExceedingOriginalLineItemNetBalance**

Bases: object

The validator to check if the amount exceeds the net balance of the original line item.

class accounting.journal_entry.forms.line_item.**NotLessThanOffsetTotal**

Bases: object

The validator to check if the amount is less than the offset total.

class `accounting.journal_entry.forms.line_item.NotStartPayableFromDebit`

Bases: object

The validator to check that a payable line item does not start from debit.

class `accounting.journal_entry.forms.line_item.NotStartReceivableFromCredit`

Bases: object

The validator to check that a receivable line item does not start from credit.

class `accounting.journal_entry.forms.line_item.OriginalLineItemExists`

Bases: object

The validator to check if the original line item exists.

class `accounting.journal_entry.forms.line_item.OriginalLineItemNeedOffset`

Bases: object

The validator to check if the original line item needs offset.

class `accounting.journal_entry.forms.line_item.OriginalLineItemNotOffset`

Bases: object

The validator to check if the original line item is not itself an offset item.

class `accounting.journal_entry.forms.line_item.OriginalLineItemOppositeDebitCredit`

Bases: object

The validator to check if the original line item is on the opposite debit or credit.

class `accounting.journal_entry.forms.line_item.PositiveAmount`

Bases: object

The validator to check if the amount is positive.

class `accounting.journal_entry.forms.line_item.SameAccountAsOriginalLineItem`

Bases: object

The validator to check if the account is the same as the original line item.

accounting.journal_entry.forms.reorder module

The reorder forms for the journal entry management.

class `accounting.journal_entry.forms.reorder.JournalEntryReorderForm`(*date: date*)

Bases: object

The form to reorder the journal entries.

date: date

The date.

is_modified: bool

Whether the order is modified.

save_order() → None

Saves the order of the account.

Returns

```
accounting.journal_entry.forms.reorder.sort_journal_entries_in(date: date, exclude: int | None =
                                                                None) → None
```

Sorts the journal entries under a date after changing the date or deleting a journal entry.

Parameters

- **date** – The date of the journal entry.
- **exclude** – The journal entry ID to exclude.

Returns

None.

Module contents

The forms for the journal entry management.

accounting.journal_entry.utils package

Submodules

accounting.journal_entry.utils.account_option module

The account option for the journal entry management.

```
class accounting.journal_entry.utils.account_option.AccountOption(account: Account)
```

Bases: object

An account option.

code: str

The account code.

id: str

The account ID.

is_in_use: bool

True if this account is in use, or False otherwise.

is_need_offset: bool

True if this account needs offset, or False otherwise.

query_values: list[str]

The values to be queried.

title: str

The account title.

accounting.journal_entry.utils.description_editor module

The description editor.

```
class accounting.journal_entry.utils.description_editor.DescriptionAccount(account: Account,  
                                                                           freq: int)
```

Bases: object

An account for a description tag.

```
add_freq(freq: int) → None
```

Adds the frequency of an account.

Parameters

freq – The frequency of the tag name with the account.

Returns

None.

```
code: str
```

The account code.

```
freq: int
```

The frequency of the tag with the account.

```
id: int
```

The account ID.

```
is_need_offset: bool
```

Whether the journal entry line items of this account need offset.

```
property title: str
```

Returns the account title.

Returns

The account title.

```
class accounting.journal_entry.utils.description_editor.DescriptionDebitCredit(debit_credit:  
                                                                               Literal['debit',  
                                                                               'credit'])
```

Bases: object

The description on debit or credit.

```
property accounts: list[DescriptionAccount]
```

Returns the suggested accounts of all tags in the description editor in debit or credit, in their frequency order.

Returns

The suggested accounts of all tags, in their frequency order.

```
add_tag(tag_type: Literal['general', 'travel', 'bus'], name: str, account: Account, freq: int) → None
```

Adds a tag.

Parameters

- **tag_type** – The tag type, either “general”, “travel”, or “bus”.
- **name** – The name.

- **account** – The associated account.
- **freq** – The frequency of the tag name with the account.

Returns

None.

bus: *DescriptionType*

The bus tags.

debit_credit: `Literal['debit', 'credit']`

Either debit or credit.

general: *DescriptionType*

The general tags.

recurring: `list[DescriptionRecurring]`

The recurring transactions.

travel: *DescriptionType*

The travel tags.

class `accounting.journal_entry.utils.description_editor.DescriptionEditor`

Bases: object

The description editor.

credit: *DescriptionDebitCredit*

The credit tags.

debit: *DescriptionDebitCredit*

The debit tags.

class `accounting.journal_entry.utils.description_editor.DescriptionRecurring`(*name: str*,
account: Account,
description_template: str)

Bases: object

A recurring transaction.

account: *DescriptionAccount*

The account.

property account_codes: `list[str]`

Returns the account codes by the order of their frequencies.

Returns

The account codes by the order of their frequencies.

description_template: `str`

The description template.

name: `str`

The name.

```
class accounting.journal_entry.utils.description_editor.DescriptionTag(name: str)
    Bases: object
    A description tag.
    property account_codes: list[str]
        Returns the account codes by the order of their frequencies.
        Returns
            The account codes by the order of their frequencies.
    property accounts: list[DescriptionAccount]
        Returns the accounts by the order of their frequencies.
        Returns
            The accounts by the order of their frequencies.
    add_account(account: Account, freq: int)
        Adds an account.
        Parameters
            • account – The associated account.
            • freq – The frequency of the tag name with the account.
        Returns
            None.
    freq: int
        The frequency of the tag.
    name: str
        The tag name.
class accounting.journal_entry.utils.description_editor.DescriptionType(type_id: Literal['general', 'travel', 'bus'])
    Bases: object
    A description type
    add_tag(name: str, account: Account, freq: int) → None
        Adds a tag.
        Parameters
            • name – The tag name.
            • account – The associated account.
            • freq – The frequency of the tag name with the account.
        Returns
            None.
    id: Literal['general', 'travel', 'bus']
        The type ID.
    property tags: list[DescriptionTag]
        Returns the tags by the order of their frequencies.
```

Returns

The tags by the order of their frequencies.

`accounting.journal_entry.utils.description_editor.get_position`(*string: str | Column, substring: str | Column*) → Function

Returns the SQL function to find the position of a substring.

Parameters

- **string** – The string.
- **substring** – The substring.

Returns

The position of the substring, starting from 1.

`accounting.journal_entry.utils.description_editor.get_prefix`(*string: str | Column, separator: str | Column*) → Function

Returns the SQL function to find the prefix of a string.

Parameters

- **string** – The string.
- **separator** – The separator.

Returns

The position of the substring, starting from 1.

accounting.journal_entry.utils.operators module

The operators for different journal entry types.

class `accounting.journal_entry.utils.operators.CashDisbursementJournalEntry`

Bases: *JournalEntryOperator*

A cash disbursement journal entry.

CHECK_ORDER: int = 1

The order when checking the journal entry operator.

property form: Type[JournalEntryForm]

Returns the form class.

Returns

The form class.

is_my_type(*journal_entry: JournalEntry*) → bool

Checks and returns whether the journal entry belongs to the type.

Parameters

journal_entry – The journal entry.

Returns

True if the journal entry belongs to the type, or False otherwise.

render_create_template(*form: CashDisbursementJournalEntryForm*) → str

Renders the template for the form to create a journal entry.

Parameters

form – The journal entry form.

Returns

the form to create a journal entry.

render_detail_template(*journal_entry*: `JournalEntry`) → str

Renders the template for the detail page.

Parameters

journal_entry – The journal entry.

Returns

the detail page.

render_edit_template(*journal_entry*: `JournalEntry`, *form*: `CashDisbursementJournalEntryForm`) → str

Renders the template for the form to edit a journal entry.

Parameters

- **journal_entry** – The journal entry.
- **form** – The form.

Returns

the form to edit a journal entry.

class `accounting.journal_entry.utils.operators.CashReceiptJournalEntry`

Bases: `JournalEntryOperator`

A cash receipt journal entry.

CHECK_ORDER: int = 2

The order when checking the journal entry operator.

property form: `Type[JournalEntryForm]`

Returns the form class.

Returns

The form class.

is_my_type(*journal_entry*: `JournalEntry`) → bool

Checks and returns whether the journal entry belongs to the type.

Parameters

journal_entry – The journal entry.

Returns

True if the journal entry belongs to the type, or False otherwise.

render_create_template(*form*: `CashReceiptJournalEntryForm`) → str

Renders the template for the form to create a journal entry.

Parameters

form – The journal entry form.

Returns

the form to create a journal entry.

render_detail_template(*journal_entry*: `JournalEntry`) → str

Renders the template for the detail page.

Parameters

journal_entry – The journal entry.

Returns

the detail page.

render_edit_template(*journal_entry*: `JournalEntry`, *form*: `CashReceiptJournalEntryForm`) → str

Renders the template for the form to edit a journal entry.

Parameters

- **journal_entry** – The journal entry.
- **form** – The form.

Returns

the form to edit a journal entry.

```
accounting.journal_entry.utils.operators.JOURNAL_ENTRY_TYPE_TO_OP: dict[JournalEntryType,
JournalEntryOperator] = {JournalEntryType.CASH_DISBURSEMENT:
<accounting.journal_entry.utils.operators.CashDisbursementJournalEntry object>,
JournalEntryType.CASH_RECEIPT:
<accounting.journal_entry.utils.operators.CashReceiptJournalEntry object>,
JournalEntryType.TRANSFER: <accounting.journal_entry.utils.operators.TransferJournalEntry
object>}
```

The map from the journal entry types to their operators.

class `accounting.journal_entry.utils.operators.JournalEntryOperator`

Bases: ABC

The base journal entry operator.

CHECK_ORDER: int = -1

The order when checking the journal entry operator.

abstract property form: `Type[JournalEntryForm]`

Returns the form class.

Returns

The form class.

abstract is_my_type(*journal_entry*: `JournalEntry`) → bool

Checks and returns whether the journal entry belongs to the type.

Parameters

journal_entry – The journal entry.

Returns

True if the journal entry belongs to the type, or False otherwise.

abstract render_create_template(*form*: `FlaskForm`) → str

Renders the template for the form to create a journal entry.

Parameters

form – The journal entry form.

Returns

the form to create a journal entry.

abstract render_detail_template(*journal_entry*: `JournalEntry`) → str

Renders the template for the detail page.

Parameters

journal_entry – The journal entry.

Returns

the detail page.

abstract render_edit_template(*journal_entry*: `JournalEntry`, *form*: `FlaskForm`) → str

Renders the template for the form to edit a journal entry.

Parameters

- **journal_entry** – The journal entry.
- **form** – The form.

Returns

the form to edit a journal entry.

class `accounting.journal_entry.utils.operators.TransferJournalEntry`

Bases: `JournalEntryOperator`

A transfer journal entry.

CHECK_ORDER: int = 3

The order when checking the journal entry operator.

property form: `Type[JournalEntryForm]`

Returns the form class.

Returns

The form class.

is_my_type(*journal_entry*: `JournalEntry`) → bool

Checks and returns whether the journal entry belongs to the type.

Parameters

journal_entry – The journal entry.

Returns

True if the journal entry belongs to the type, or False otherwise.

render_create_template(*form*: `TransferJournalEntryForm`) → str

Renders the template for the form to create a journal entry.

Parameters

form – The journal entry form.

Returns

the form to create a journal entry.

render_detail_template(*journal_entry*: `JournalEntry`) → str

Renders the template for the detail page.

Parameters

journal_entry – The journal entry.

Returns

the detail page.

render_edit_template(*journal_entry*: `JournalEntry`, *form*: `TransferJournalEntryForm`) → str

Renders the template for the form to edit a journal entry.

Parameters

- **journal_entry** – The journal entry.
- **form** – The form.

Returns

the form to edit a journal entry.

```
accounting.journal_entry.utils.operators.get_journal_entry_op(journal_entry: JournalEntry,
                                                             is_check_as: bool = False) →
                                                             JournalEntryOperator
```

Returns the journal entry operator that may be specified in the “as” query parameter. If it is not specified, check the journal entry type from the journal entry.

Parameters

- **journal_entry** – The journal entry.
- **is_check_as** – True to check the “as” parameter, or False otherwise.

Returns

None.

accounting.journal_entry.utils.original_line_items module

The selectable original line items.

```
accounting.journal_entry.utils.original_line_items.get_selectable_original_line_items(line_item_id_on_form:
                                                                                       set[int],
                                                                                       is_payable:
                                                                                       bool,
                                                                                       is_receivable:
                                                                                       bool)
                                                                                       →
                                                                                       list[JournalEntryLineItem]
```

Queries and returns the selectable original line items, with their net balances. The offset amounts of the form is excluded.

Parameters

- **line_item_id_on_form** – The ID of the line items on the form.
- **is_payable** – True to check the payable original line items, or False otherwise.
- **is_receivable** – True to check the receivable original line items, or False otherwise.

Returns

The selectable original line items, with their net balances.

Module contents

The utilities for the journal entry management.

Submodules

accounting.journal_entry.converters module

The path converters for the journal entry management.

class `accounting.journal_entry.converters.DateConverter`(*map: Map, *args: t.Any, **kwargs: t.Any*)

Bases: `BaseConverter`

The date converter to convert the ISO date from and to the corresponding date in the routes.

to_python(*value: str*) → `date`

Converts an ISO date to a date.

Parameters

value – The ISO date.

Returns

The corresponding date.

to_url(*value: date*) → `str`

Converts a date to its ISO date.

Parameters

value – The date.

Returns

The ISO date.

class `accounting.journal_entry.converters.JournalEntryConverter`(*map: Map, *args: t.Any, **kwargs: t.Any*)

Bases: `BaseConverter`

The journal entry converter to convert the journal entry ID from and to the corresponding journal entry in the routes.

to_python(*value: str*) → `JournalEntry`

Converts a journal entry ID to a journal entry.

Parameters

value – The journal entry ID.

Returns

The corresponding journal entry.

to_url(*value: JournalEntry*) → `str`

Converts a journal entry to its ID.

Parameters

value – The journal entry.

Returns

The ID.

class `accounting.journal_entry.converters.JournalEntryTypeConverter`(*map: Map, *args: t.Any, **kwargs: t.Any*)

Bases: `BaseConverter`

The journal entry converter to convert the journal entry type ID from and to the corresponding journal entry type in the routes.

to_python(*value: str*) → *JournalEntryType*

Converts a journal entry ID to a journal entry.

Parameters

value – The journal entry ID.

Returns

The corresponding journal entry type.

to_url(*value: JournalEntryType*) → str

Converts a journal entry type to its ID.

Parameters

value – The journal entry type.

Returns

The ID.

accounting.journal_entry.template_filters module

The template filters for the journal entry management.

`accounting.journal_entry.template_filters.format_amount_input`(*value: Decimal | None*) → str

Format an amount for an input value.

Parameters

value – The amount.

Returns

The formatted amount text for an input value.

`accounting.journal_entry.template_filters.text2html`(*value: str*) → str

Converts plain text into HTML.

Parameters

value – The plain text.

Returns

The HTML.

`accounting.journal_entry.template_filters.to_transfer`(*uri: str*) → str

Adds the transfer journal entry type to the URI.

Parameters

uri – The URI.

Returns

The result URL, with the transfer journal entry type added.

`accounting.journal_entry.template_filters.with_type`(*uri: str*) → str

Adds the journal entry type to the URI, if it is specified.

Parameters

uri – The URI.

Returns

The result URL, optionally with the journal entry type added.

accounting.journal_entry.views module

The views for the journal entry management.

`accounting.journal_entry.views.bp: Blueprint = <Blueprint 'journal-entry'>`

The view blueprint for the journal entry management.

Module contents

The journal entry management.

`accounting.journal_entry.init_app(app: Flask, bp: Blueprint) → None`

Initialize the application.

Parameters

- **app** – The Flask application.
- **bp** – The blueprint of the accounting application.

Returns

None.

2.1.5 accounting.option package

Submodules

accounting.option.forms module

The forms for the option management.

class `accounting.option.forms.AccountNotCurrent`

Bases: `object`

The validator to check that the account is a current account.

class `accounting.option.forms.CurrentAccountExists`

Bases: `object`

The validator to check that the current account exists.

class `accounting.option.forms.NotStartPayableFromExpense`

Bases: `object`

The validator to check that a payable line item does not start from expense.

class `accounting.option.forms.NotStartReceivableFromIncome`

Bases: `object`

The validator to check that a receivable line item does not start from income.

class `accounting.option.forms.OptionForm(*args, **kwargs)`

Bases: `FlaskForm`

The form to update the options.

property current_accounts: `list[CurrentAccount]`

Returns the current accounts.

Returns

The current accounts.

default_currency_code = `<UnboundField(StringField, (), {'filters': [<function strip_text>], 'validators': [<wtforms.validators.DataRequired object>, <accounting.forms.CurrencyExists object>]})>`

The default currency code.

default_ie_account_code = `<UnboundField(StringField, (), {'filters': [<function strip_text>], 'validators': [<wtforms.validators.DataRequired object>, <accounting.option.forms.CurrentAccountExists object>, <accounting.option.forms.AccountNotCurrent object>]})>`

The default account code for the income and expenses log.

populate_obj(*obj*: `Options`) → `None`

Populates the form data into a currency object.

Parameters

obj – The currency object.

Returns

`None`.

recurring = `<UnboundField(FormField, (<class 'accounting.option.forms.RecurringForm'>,), {})>`

The recurring expenses and incomes.

class `accounting.option.forms.RecurringExpenseForm(*args, **kwargs)`

Bases: `RecurringItemForm`

The sub-form to add or update the recurring expenses.

account_code = `<UnboundField(StringField, (), {'filters': [<function strip_text>], 'validators': [<wtforms.validators.DataRequired object>, <accounting.forms.AccountExists object>, <accounting.forms.IsDebitAccount object>, <accounting.option.forms.NotStartPayableFromExpense object>]})>`

The account code.

description_template = `<UnboundField(StringField, (), {'filters': [<function strip_text>], 'validators': [<wtforms.validators.DataRequired object>]})>`

The template for the line item description.

name = `<UnboundField(StringField, (), {'filters': [<function strip_text>], 'validators': [<wtforms.validators.DataRequired object>]})>`

The name of the recurring item.

no = `<UnboundField(IntegerField, (), {})>`

The order number of this recurring item.

class `accounting.option.forms.RecurringForm(*args, **kwargs)`

Bases: `RecurringItemForm`

The sub-form for the recurring expenses and incomes.

```
property as_data: dict[str, list[tuple[str, str, str]]]
```

Returns the form data.

Returns

The form data.

```
property expense_accounts: list[Account]
```

The expense accounts.

Returns

None.

```
expenses = <UnboundField(FieldList, (<UnboundField(FormField, (<class  
'accounting.option.forms.RecurringExpenseForm'>)), {}>)), {}>
```

The recurring expenses.

```
property income_accounts: list[Account]
```

The income accounts.

Returns

None.

```
incomes = <UnboundField(FieldList, (<UnboundField(FormField, (<class  
'accounting.option.forms.RecurringIncomeForm'>)), {}>)), {}>
```

The recurring incomes.

```
property item_template: str
```

Returns the template of a recurring item.

Returns

The template of a recurring item.

```
class accounting.option.forms.RecurringIncomeForm(*args, **kwargs)
```

Bases: *RecurringItemForm*

The sub-form to add or update the recurring incomes.

```
account_code = <UnboundField(StringField, (), {'filters': [<function strip_text>],  
'validators': [<wtforms.validators.DataRequired object>,  
<accounting.forms.AccountExists object>, <accounting.forms.IsCreditAccount object>,  
<accounting.option.forms.NotStartReceivableFromIncome object>]})>
```

The account code.

```
description_template = <UnboundField(StringField, (), {'filters': [<function  
strip_text>], 'validators': [<wtforms.validators.DataRequired object>]})>
```

The description template.

```
name = <UnboundField(StringField, (), {'filters': [<function strip_text>],  
'validators': [<wtforms.validators.DataRequired object>]})>
```

The name of the recurring item.

```
no = <UnboundField(IntegerField, (), {})>
```

The order number of this recurring item.

```
class accounting.option.forms.RecurringItemForm(*args, **kwargs)
```

Bases: *FlaskForm*

The base sub-form to add or update the recurring item.

account_code = <UnboundField(StringField, (), {})>

The account code.

property account_text: str | None

Returns the account text.

Returns

The account text.

property all_errors: list[str | LazyString]

Returns all the errors of the form.

Returns

All the errors of the form.

description_template = <UnboundField(StringField, (), {})>

The description template.

name = <UnboundField(StringField, (), {})>

The name of the recurring item.

no = <UnboundField(IntegerField, (), {})>

The order number of this recurring item.

accounting.option.views module

The views for the option management.

accounting.option.views.bp: Blueprint = <Blueprint 'option'>

The view blueprint for the currency management.

Module contents

The option management.

accounting.option.init_app(bp: Blueprint) → None

Initialize the application.

Parameters

bp – The blueprint of the accounting application.

Returns

None.

2.1.6 accounting.report package

Subpackages

accounting.report.period package

Submodules

accounting.report.period.chooser module

The period chooser.

This file is largely taken from the NanoParma ERP project, first written in 2021/9/16 by imacat (imacat@nanoparma.com).

```
class accounting.report.period.chooser.PeriodChooser(get_url: Callable[[Period], str])
```

Bases: object

The period chooser.

all_url: str

The URL for all period.

available_years: list[int]

The available years.

data_start: date | None

The start of the data.

has_data: bool

Whether there is any data.

has_last_month: bool

Where there is data in last month.

has_last_year: bool

Whether there is data in last year.

has_yesterday: bool

Whether there is data in yesterday.

last_month_url: str

The URL for last month.

last_year_url: str

The URL for last year.

since_last_month_url: str

The URL since last mint.

this_month_url: str

The URL for this month.

this_year_url: str

The URL for this year.

today_url: str

The URL for today.

url_template: str

The URL template.

year_url(*year: int*) → str

Returns the period URL of a year.

Parameters

year – The year

Returns

The period URL of the year.

yesterday_url: **str**

The URL for yesterday.

accounting.report.period.description module

The period description composer.

`accounting.report.period.description.get_desc(start: date | None, end: date | None) → str`

Returns the period description.

Parameters

- **start** – The start of the period.
- **end** – The end of the period.

Returns

The period description.

accounting.report.period.month_end module

The utility to return the end of a month.

`accounting.report.period.month_end.month_end(day: date) → date`

Returns the end day of month for a date.

Parameters

day – The date.

Returns

The end day of the month of that day.

accounting.report.period.parser module

The period specification parser.

`accounting.report.period.parser.DATE_SPEC_RE: str =`
`'(\\d{4})(?:-(\\d{2})(?:-(\\d{2}))?)?'`

The regular expression of a date specification.

`accounting.report.period.parser.get_period(spec: str | None = None) → Period`

Returns a period instance.

Parameters

spec – The period specification, or omit for the default.

Returns

The period instance.

Raises

ValueError – When the period specification is invalid.

accounting.report.period.period module

The date period.

This file is largely taken from the NanoParma ERP project, first written in 2021/9/16 by imacat (imacat@nanoparma.com).

class `accounting.report.period.period.Period`(*start: date | None, end: date | None*)

Bases: `object`

A date period.

property before: Self | None

Returns the period before this period.

Returns

The period before this period.

desc: str

The text description.

end: date | None

The end of the period.

is_a_day: bool

Whether the period is a single day.

is_a_month: bool

Whether the period is a whole month.

is_a_year: bool

Whether the period is a whole year.

is_all: bool

Whether the period is all time.

is_default: bool

Whether this is the default period.

is_last_month: bool

Whether the period is last month.

is_last_year: bool

Whether the period is last year.

is_since_last_month: bool

Whether the period is since last month.

is_this_month: bool

Whether the period is this month.

is_this_year: bool

Whether the period is this year.

is_today: bool

Whether the period is today.

property is_type_arbitrary: bool

Returns whether this period is an arbitrary period.

Returns

True if this is an arbitrary period, or False otherwise.

is_type_month: bool

Whether the period is for the month chooser.

is_year(*year: int*) → bool

Returns whether the period is the specific year period.

Parameters

year – The year.

Returns

True if the period is the year period, or False otherwise.

is_yesterday: bool

Whether the period is yesterday.

spec: str

The period specification.

start: date | None

The start of the period.

accounting.report.period.shortcuts module

The named shortcut periods.

class `accounting.report.period.shortcuts.AllTime`

Bases: *Period*

The period of all time.

class `accounting.report.period.shortcuts.LastMonth`

Bases: *Period*

The period of this month.

class `accounting.report.period.shortcuts.LastYear`

Bases: *Period*

The period of last year.

class `accounting.report.period.shortcuts.SinceLastMonth`

Bases: *Period*

The period of this month.

class `accounting.report.period.shortcuts.TemplatePeriod`

Bases: *Period*

The period template.

class `accounting.report.period.shortcuts.ThisMonth`

Bases: *Period*

The period of this month.

class `accounting.report.period.shortcuts.ThisYear`

Bases: *Period*

The period of this year.

class `accounting.report.period.shortcuts.Today`

Bases: *Period*

The period of today.

class `accounting.report.period.shortcuts.YearPeriod`(*year: int*)

Bases: *Period*

A year period.

class `accounting.report.period.shortcuts.Yesterday`

Bases: *Period*

The period of yesterday.

accounting.report.period.specification module

The period specification composer.

`accounting.report.period.specification.get_spec`(*start: date | None, end: date | None*) → str

Returns the period specification.

Parameters

- **start** – The start of the period.
- **end** – The end of the period.

Returns

The period specification.

Module contents

The period utility.

accounting.report.reports package

Submodules

accounting.report.reports.balance_sheet module

The balance sheet.

class `accounting.report.reports.balance_sheet.AccountCollector`(*currency: Currency, period: Period*)

Bases: object

The balance sheet account collector.

accounts: list[*ReportAccount*]

The balance sheet accounts.

class `accounting.report.reports.balance_sheet.BalanceSheet`(*currency*: `Currency`, *period*: `Period`)

Bases: `BaseReport`

The balance sheet.

csv() → `Response`

Returns the report as CSV for download.

Returns

The response of the report for download.

html() → `str`

Composes and returns the report as HTML.

Returns

The report as HTML.

class `accounting.report.reports.balance_sheet.CSVHalfRow`(*title*: `str` | `None`, *amount*: `Decimal` | `None`)

Bases: `object`

A half row in the CSV.

amount: `Decimal` | `None`

The amount.

title: `str` | `None`

The title.

class `accounting.report.reports.balance_sheet.CSVRow`

Bases: `BaseCSVRow`

A row in the CSV.

asset_amount: `Decimal` | `None`

The amount of the asset.

asset_title: `str` | `None`

The title of the asset.

liability_amount: `Decimal` | `None`

The amount of the liability.

liability_title: `str` | `None`

The title of the liability.

property values: `list[str | Decimal | None]`

Returns the values of the row.

Returns

The values of the row.

class `accounting.report.reports.balance_sheet.PageParams`(*currency*: `Currency`, *period*: `Period`,
has_data: `bool`, *assets*: `Section`,
liabilities: `Section`, *owner_s_equity*:
`Section`)

Bases: `BasePageParams`

The HTML page parameters.

assets: *Section*

The assets.

currency: *Currency*

The currency.

property currency_options: *list[OptionLink]*

Returns the currency options.

Returns

The currency options.

property has_data: *bool*

Returns whether there is any data on the page.

Returns

True if there is any data, or False otherwise.

liabilities: *Section*

The liabilities.

owner_s_equity: *Section*

The owner's equity.

period: *Period*

The period.

period_chooser: *PeriodChooser*

The period chooser.

property report_chooser: *ReportChooser*

Returns the report chooser.

Returns

The report chooser.

class `accounting.report.reports.balance_sheet.ReportAccount`(*account: Account, amount: Decimal, url: str*)

Bases: `object`

An account in the report.

account: *Account*

The account.

amount: *Decimal*

The amount of the account.

url: *str*

The URL to the ledger of the account.

class `accounting.report.reports.balance_sheet.Section`(*title: BaseAccount*)

Bases: `object`

A section.

subsections: *list[Subsection]*

The subsections in the section.

title: *BaseAccount*

The title account.

property total: *Decimal*

Returns the total of the section.

Returns

The total of the section.

class `accounting.report.reports.balance_sheet.Subsection`(*title: BaseAccount*)

Bases: *object*

A subsection.

accounts: *list[ReportAccount]*

The accounts in the subsection.

title: *BaseAccount*

The title account.

property total: *Decimal*

Returns the total of the subsection.

Returns

The total of the subsection.

accounting.report.reports.income_expenses module

The income and expenses log.

class `accounting.report.reports.income_expenses.CSVRow`(*date: date | str | None, account: str | None, description: str | None, income: str | Decimal | None, expense: str | Decimal | None, balance: str | Decimal | None, note: str | None*)

Bases: *BaseCSVRow*

A row in the CSV.

account: *str | None*

The account.

balance: *str | Decimal | None*

The balance.

date: *date | str | None*

The date.

description: *str | None*

The description.

expense: *str | Decimal | None*

The expense.

income: *str | Decimal | None*

The income.

note: `str | None`

The note.

property values: `list[str | Decimal | None]`

Returns the values of the row.

Returns

The values of the row.

class `accounting.report.reports.income_expenses.IncomeExpenses`(*currency: Currency, account: CurrentAccount, period: Period*)

Bases: `BaseReport`

The income and expenses log.

csv() → `Response`

Returns the report as CSV for download.

Returns

The response of the report for download.

html() → `str`

Composes and returns the report as HTML.

Returns

The report as HTML.

class `accounting.report.reports.income_expenses.LineItemCollector`(*currency: Currency, account: CurrentAccount, period: Period*)

Bases: `object`

The line item collector.

brought_forward: `ReportLineItem | None`

The brought-forward line item.

line_items: `list[ReportLineItem]`

The line items.

total: `ReportLineItem | None`

The total line item.

class `accounting.report.reports.income_expenses.PageParams`(*currency: Currency, account: CurrentAccount, period: Period, has_data: bool, pagination: Pagination[ReportLineItem], brought_forward: ReportLineItem | None, line_items: list[ReportLineItem], total: ReportLineItem | None*)

Bases: `BasePageParams`

The HTML page parameters.

account: `CurrentAccount`

The account.

property account_options: `list[OptionLink]`

Returns the account options.

Returns

The account options.

brought_forward: `ReportLineItem | None`

The brought-forward line item.

currency: `Currency`

The currency.

property currency_options: `list[OptionLink]`

Returns the currency options.

Returns

The currency options.

property has_data: `bool`

Returns whether there is any data on the page.

Returns

True if there is any data, or False otherwise.

line_items: `list[ReportLineItem]`

The line items.

pagination: `Pagination[ReportLineItem]`

The pagination.

period: `Period`

The period.

period_chooser: `PeriodChooser`

The period chooser.

property report_chooser: `ReportChooser`

Returns the report chooser.

Returns

The report chooser.

total: `ReportLineItem | None`

The total line item.

class `accounting.report.reports.income_expenses.ReportLineItem`(*line_item: JournalEntryLineItem | None = None*)

Bases: `object`

A line item in the report.

account: `Account | None`

The account.

balance: `Decimal | None`

The balance.

date: `date | None`

The date.

description: `str | None`

The description.

expense: `Decimal | None`

The expense amount.

income: `Decimal | None`

The income amount.

is_brought_forward: `bool`

Whether this is the brought-forward line item.

is_total: `bool`

Whether this is the total line item.

note: `str | None`

The note.

url: `str | None`

The URL to the journal entry line item.

accounting.report.reports.income_statement module

The income statement.

class `accounting.report.reports.income_statement.AccumulatedTotal`(*title: str*)

Bases: `object`

An accumulated total.

amount: `Decimal`

The amount of the account.

title: `str`

The account.

class `accounting.report.reports.income_statement.CSVRow`(*text: str | None, amount: str | Decimal | None*)

Bases: `BaseCSVRow`

A row in the CSV.

amount: `str | Decimal | None`

The amount.

text: `str | None`

The text.

property values: `list[str | Decimal | None]`

Returns the values of the row.

Returns

The values of the row.

class `accounting.report.reports.income_statement.IncomeStatement`(*currency: Currency, period: Period*)

Bases: `BaseReport`

The income statement.

csv() → Response

Returns the report as CSV for download.

Returns

The response of the report for download.

html() → str

Composes and returns the report as HTML.

Returns

The report as HTML.

class `accounting.report.reports.income_statement.PageParams`(*currency: Currency, period: Period, has_data: bool, sections: list[Section]*)

Bases: *BasePageParams*

The HTML page parameters.

currency: *Currency*

The currency.

property currency_options: *list[OptionLink]*

Returns the currency options.

Returns

The currency options.

property has_data: *bool*

Returns whether there is any data on the page.

Returns

True if there is any data, or False otherwise.

period: *Period*

The period.

period_chooser: *PeriodChooser*

The period chooser.

property report_chooser: *ReportChooser*

Returns the report chooser.

Returns

The report chooser.

sections: *list[Section]*

The sections in the income statement.

class `accounting.report.reports.income_statement.ReportAccount`(*account: Account, amount: Decimal, url: str*)

Bases: *object*

An account in the report.

account: *Account*

The account.

amount: *Decimal*

The amount of the account.

url: str

The URL to the ledger of the account.

class accounting.report.reports.income_statement.**Section**(*title: BaseAccount, accumulated_title: str*)

Bases: object

A section.

accumulated: AccumulatedTotal

The accumulated total.

subsections: list[Subsection]

The subsections in the section.

title: BaseAccount

The title account.

property total: Decimal

Returns the total of the section.

Returns

The total of the section.

class accounting.report.reports.income_statement.**Subsection**(*title: BaseAccount*)

Bases: object

A subsection.

accounts: list[ReportAccount]

The accounts in the subsection.

title: BaseAccount

The title account.

property total: Decimal

Returns the total of the subsection.

Returns

The total of the subsection.

accounting.report.reports.journal module

The journal.

class accounting.report.reports.journal.**CSVRow**(*journal_entry_date: str | date, currency: str, account: str, description: str | None, debit: str | Decimal | None, credit: str | Decimal | None, note: str | None*)

Bases: *BaseCSVRow*

A row in the CSV.

account: str

The account.

credit: str | Decimal | None

The credit amount.

currency: `str`

The currency.

date: `str | date`

The date.

debit: `str | Decimal | None`

The debit amount.

description: `str | None`

The description.

note: `str | None`

The note.

property values: `list[str | Decimal | None]`

Returns the values of the row.

Returns

The values of the row.

class `accounting.report.reports.journal.Journal`(*period*: `Period`)

Bases: `BaseReport`

The journal.

csv() → `Response`

Returns the report as CSV for download.

Returns

The response of the report for download.

html() → `str`

Composes and returns the report as HTML.

Returns

The report as HTML.

class `accounting.report.reports.journal.PageParams`(*period*: `Period`, *pagination*: `Pagination[JournalEntryLineItem]`, *line_items*: `list[JournalEntryLineItem]`)

Bases: `BasePageParams`

The HTML page parameters.

property has_data: `bool`

Returns whether there is any data on the page.

Returns

True if there is any data, or False otherwise.

line_items: `list[JournalEntryLineItem]`

The line items.

pagination: `Pagination[JournalEntryLineItem]`

The pagination.

period: `Period`

The period.

period_chooser: *PeriodChooser*

The period chooser.

property report_chooser: *ReportChooser*

Returns the report chooser.

Returns

The report chooser.

class `accounting.report.reports.journal.ReportLineItem`(*line_item: JournalEntryLineItem*)

Bases: object

A line item in the report.

account: *Account*

The account.

amount: *Decimal*

The amount.

credit: *Decimal | None*

The credit amount.

currency: *Currency*

The account.

debit: *Decimal | None*

The debit amount.

description: *str | None*

The description.

journal_entry: *JournalEntry*

The journal entry.

line_item: *JournalEntryLineItem*

The journal entry line item.

`accounting.report.reports.journal.get_csv_rows`(*line_items: list[JournalEntryLineItem]*) → *list[CSVRow]*

Composes and returns the CSV rows from the line items.

Parameters

line_items – The line items.

Returns

The CSV rows.

accounting.report.reports.ledger module

The ledger.

class `accounting.report.reports.ledger.CSVRow`(*date: date | str | None, description: str | None, debit: str | Decimal | None, credit: str | Decimal | None, balance: str | Decimal | None, note: str | None*)

Bases: *BaseCSVRow*

A row in the CSV.

balance: `str | Decimal | None`

The balance.

credit: `str | Decimal | None`

The credit amount.

date: `date | str | None`

The date.

debit: `str | Decimal | None`

The debit amount.

description: `str | None`

The description.

note: `str | None`

The note.

property values: `list[str | Decimal | None]`

Returns the values of the row.

Returns

The values of the row.

class `accounting.report.reports.ledger.Ledger`(*currency: Currency, account: Account, period: Period*)

Bases: `BaseReport`

The ledger.

csv() → Response

Returns the report as CSV for download.

Returns

The response of the report for download.

html() → str

Composes and returns the report as HTML.

Returns

The report as HTML.

class `accounting.report.reports.ledger.LineItemCollector`(*currency: Currency, account: Account, period: Period*)

Bases: `object`

The line item collector.

brought_forward: `ReportLineItem | None`

The brought-forward line item.

line_items: `list[ReportLineItem]`

The line items.

total: `ReportLineItem | None`

The total line item.

```
class accounting.report.reports.ledger.PageParams(currency: Currency, account: Account, period:
Period, has_data: bool, pagination:
Pagination[ReportLineItem], brought_forward:
ReportLineItem | None, line_items:
list[ReportLineItem], total: ReportLineItem |
None)
```

Bases: *BasePageParams*

The HTML page parameters.

account: *Account*

The account.

property account_options: *list[OptionLink]*

Returns the account options.

Returns

The account options.

brought_forward: *ReportLineItem | None*

The brought-forward line item.

currency: *Currency*

The currency.

property currency_options: *list[OptionLink]*

Returns the currency options.

Returns

The currency options.

property has_data: *bool*

Returns whether there is any data on the page.

Returns

True if there is any data, or False otherwise.

line_items: *list[ReportLineItem]*

The line items.

pagination: *Pagination[ReportLineItem]*

The pagination.

period: *Period*

The period.

period_chooser: *PeriodChooser*

The period chooser.

property report_chooser: *ReportChooser*

Returns the report chooser.

Returns

The report chooser.

total: *ReportLineItem | None*

The total line item.

```
class accounting.report.reports.ledger.ReportLineItem(line_item: JournalEntryLineItem | None = None)
```

Bases: object

A line item in the report.

balance: `Decimal` | `None`

The balance.

credit: `Decimal` | `None`

The credit amount.

date: `date` | `None`

The date.

debit: `Decimal` | `None`

The debit amount.

description: `str` | `None`

The description.

is_brought_forward: `bool`

Whether this is the brought-forward line item.

is_total: `bool`

Whether this is the total line item.

note: `str` | `None`

The note.

url: `str` | `None`

The URL to the journal entry line item.

accounting.report.reports.search module

The search.

```
class accounting.report.reports.search.LineItemCollector
```

Bases: object

The line item collector.

line_items: `list`[[JournalEntryLineItem](#)]

The line items.

```
class accounting.report.reports.search.PageParams(pagination: Pagination[JournalEntryLineItem],  
                                                line_items: list[JournalEntryLineItem])
```

Bases: [BasePageParams](#)

The HTML page parameters.

property has_data: `bool`

Returns whether there is any data on the page.

Returns

True if there is any data, or False otherwise.

line_items: `list[JournalEntryLineItem]`

The line items.

pagination: `Pagination[JournalEntryLineItem]`

The pagination.

property report_chooser: `ReportChooser`

Returns the report chooser.

Returns

The report chooser.

class `accounting.report.reports.search.Search`

Bases: `BaseReport`

The search.

csv() → Response

Returns the report as CSV for download.

Returns

The response of the report for download.

html() → str

Composes and returns the report as HTML.

Returns

The report as HTML.

accounting.report.reports.trial_balance module

The trial balance.

class `accounting.report.reports.trial_balance.CSVRow`(*text: str | None, debit: str | Decimal | None, credit: str | Decimal | None*)

Bases: `BaseCSVRow`

A row in the CSV.

credit: `str | Decimal | None`

The credit amount.

debit: `str | Decimal | None`

The debit amount.

text: `str | None`

The text.

property values: `list[str | Decimal | None]`

Returns the values of the row.

Returns

The values of the row.

class `accounting.report.reports.trial_balance.PageParams`(*currency: Currency, period: Period, accounts: list[ReportAccount], total: Total*)

Bases: *BasePageParams*

The HTML page parameters.

accounts: `list[ReportAccount]`

The accounts in the trial balance.

currency: *Currency*

The currency.

property currency_options: `list[OptionLink]`

Returns the currency options.

Returns

The currency options.

property has_data: `bool`

Returns whether there is any data on the page.

Returns

True if there is any data, or False otherwise.

period: *Period*

The period.

period_chooser: *PeriodChooser*

The period chooser.

property report_chooser: *ReportChooser*

Returns the report chooser.

Returns

The report chooser.

total: *Total*

The total of the trial balance.

```
class accounting.report.reports.trial_balance.ReportAccount(account: Account, amount: Decimal,
                                                            url: str)
```

Bases: object

An account in the report.

account: *Account*

The account.

credit: `Decimal | None`

The credit amount.

debit: `Decimal | None`

The debit amount.

url: `str`

The URL to the ledger of the account.

```
class accounting.report.reports.trial_balance.Total(debit: Decimal, credit: Decimal)
```

Bases: object

The totals.

credit: Decimal | None

The credit amount.

debit: Decimal | None

The debit amount.

class `accounting.report.reports.trial_balance.TrialBalance`(*currency: Currency, period: Period*)

Bases: *BaseReport*

The trial balance.

csv() → Response

Returns the report as CSV for download.

Returns

The response of the report for download.

html() → str

Composes and returns the report as HTML.

Returns

The report as HTML.

`accounting.report.reports.unapplied` module

The unapplied original line items.

class `accounting.report.reports.unapplied.CSVRow`(*journal_entry_date: str | date, currency: str, description: str | None, amount: str | Decimal, net_balance: str | Decimal*)

Bases: *BaseCSVRow*

A row in the CSV.

amount: str | Decimal

The amount.

currency: str

The currency.

date: str | date

The date.

description: str | None

The description.

net_balance: str | Decimal

The net balance.

property values: list[str | date | Decimal | None]

Returns the values of the row.

Returns

The values of the row.

class `accounting.report.reports.unapplied.PageParams`(*currency: Currency, account: Account, pagination: Pagination[JournalEntryLineItem], line_items: list[JournalEntryLineItem]*)

Bases: *BasePageParams*

The HTML page parameters.

account: *Account*

The account.

property account_options: *list[OptionLink]*

Returns the account options.

Returns

The account options.

currency: *Currency*

The currency.

property currency_options: *list[OptionLink]*

Returns the currency options.

Returns

The currency options.

property has_data: *bool*

Returns whether there is any data on the page.

Returns

True if there is any data, or False otherwise.

line_items: *list[JournalEntryLineItem]*

The line items.

pagination: *Pagination[JournalEntryLineItem]*

The pagination.

property report_chooser: *ReportChooser*

Returns the report chooser.

Returns

The report chooser.

class `accounting.report.reports.unapplied.UnappliedOriginalLineItems`(*currency: Currency,*
account: Account)

Bases: *BaseReport*

The unapplied original line items.

csv() → Response

Returns the report as CSV for download.

Returns

The response of the report for download.

html() → str

Composes and returns the report as HTML.

Returns

The report as HTML.

`accounting.report.reports.unapplied.get_csv_rows(line_items: list[JournalEntryLineItem]) → list[CSVRow]`

Composes and returns the CSV rows from the line items.

Parameters

line_items – The line items.

Returns

The CSV rows.

accounting.report.reports.unapplied_accounts module

The accounts with unapplied original line items.

`class accounting.report.reports.unapplied_accounts.AccountsWithUnappliedOriginalLineItems(currency: Currency)`

Bases: *BaseReport*

The accounts with unapplied original line items.

`csv()` → Response

Returns the report as CSV for download.

Returns

The response of the report for download.

`html()` → str

Composes and returns the report as HTML.

Returns

The report as HTML.

`class accounting.report.reports.unapplied_accounts.CSVRow(account: str, count: int | str)`

Bases: *BaseCSVRow*

A row in the CSV.

account: str

The currency.

count: int | str

The number of unapplied original line items.

property values: list[str | date | Decimal | None]

Returns the values of the row.

Returns

The values of the row.

`class accounting.report.reports.unapplied_accounts.PageParams(currency: Currency, accounts: list[Account])`

Bases: *BasePageParams*

The HTML page parameters.

property `account_options`: `list[OptionLink]`

Returns the account options.

Returns

The account options.

accounts: `list[Account]`

The accounts.

currency: `Currency`

The currency.

property `currency_options`: `list[OptionLink]`

Returns the currency options.

Returns

The currency options.

property `has_data`: `bool`

Returns whether there is any data on the page.

Returns

True if there is any data, or False otherwise.

property `report_chooser`: `ReportChooser`

Returns the report chooser.

Returns

The report chooser.

`accounting.report.reports.unapplied_accounts.get_csv_rows(accounts: list[Account]) → list[CSVRow]`

Composes and returns the CSV rows from the line items.

Parameters

`accounts` – The accounts.

Returns

The CSV rows.

`accounting.report.reports.unmatched` module

The unmatched offsets.

class `accounting.report.reports.unmatched.CSVRow`(`journal_entry_date`: `str | date`, `currency`: `str`, `description`: `str | None`, `debit`: `str | Decimal`, `credit`: `str | Decimal`, `balance`: `str | Decimal`)

Bases: `BaseCSVRow`

A row in the CSV.

balance: `str | Decimal`

The balance.

credit: `str | Decimal | None`

The credit amount.

currency: `str`

The currency.

date: `str` | `date`

The date.

debit: `str` | `Decimal` | `None`

The debit amount.

description: `str` | `None`

The description.

property values: `list[str | date | Decimal | None]`

Returns the values of the row.

Returns

The values of the row.

class `accounting.report.reports.unmatched.PageParams`(*currency: Currency, account: Account, match_status: str | LazyString, matched_pairs: list[OffsetPair], pagination: Pagination[JournalEntryLineItem], line_items: list[JournalEntryLineItem]*)

Bases: `BasePageParams`

The HTML page parameters.

account: `Account`

The account.

property account_options: `list[OptionLink]`

Returns the account options.

Returns

The account options.

currency: `Currency`

The currency.

property currency_options: `list[OptionLink]`

Returns the currency options.

Returns

The currency options.

property has_data: `bool`

Returns whether there is any data on the page.

Returns

True if there is any data, or False otherwise.

line_items: `list[JournalEntryLineItem]`

The line items.

match_status: `str` | `LazyString`

The match status message.

matched_pairs: `list[OffsetPair]`

A list of matched pairs.

pagination: *Pagination*[*JournalEntryLineItem*]

The pagination.

property report_chooser: *ReportChooser*

Returns the report chooser.

Returns

The report chooser.

class `accounting.report.reports.unmatched.UnmatchedOffsets`(*currency*: *Currency*, *account*: *Account*)

Bases: *BaseReport*

The unmatched offsets.

csv() → *Response*

Returns the report as CSV for download.

Returns

The response of the report for download.

html() → *str*

Composes and returns the report as HTML.

Returns

The report as HTML.

`accounting.report.reports.unmatched.get_csv_rows`(*line_items*: *list*[*JournalEntryLineItem*]) → *list*[*CSVRow*]

Composes and returns the CSV rows from the line items.

Parameters

line_items – The line items.

Returns

The CSV rows.

accounting.report.reports.unmatched_accounts module

The accounts with unmatched offsets.

class `accounting.report.reports.unmatched_accounts.AccountsWithUnmatchedOffsets`(*currency*: *Currency*)

Bases: *BaseReport*

The accounts with unmatched offsets.

csv() → *Response*

Returns the report as CSV for download.

Returns

The response of the report for download.

html() → *str*

Composes and returns the report as HTML.

Returns

The report as HTML.

class `accounting.report.reports.unmatched_accounts.CSVRow`(*account: str, count: int | str*)

Bases: `BaseCSVRow`

A row in the CSV.

account: str

The currency.

count: int | str

The number of unapplied original line items.

property values: list[str | date | Decimal | None]

Returns the values of the row.

Returns

The values of the row.

class `accounting.report.reports.unmatched_accounts.PageParams`(*currency: Currency, accounts: list[Account]*)

Bases: `BasePageParams`

The HTML page parameters.

property account_options: list[OptionLink]

Returns the account options.

Returns

The account options.

accounts: list[Account]

The accounts.

currency: Currency

The currency.

property currency_options: list[OptionLink]

Returns the currency options.

Returns

The currency options.

property has_data: bool

Returns whether there is any data on the page.

Returns

True if there is any data, or False otherwise.

property report_chooser: ReportChooser

Returns the report chooser.

Returns

The report chooser.

`accounting.report.reports.unmatched_accounts.get_csv_rows`(*accounts: list[Account]*) → `list[CSVRow]`

Composes and returns the CSV rows from the line items.

Parameters

accounts – The accounts.

Returns

The CSV rows.

Module contents

The reports.

accounting.report.utils package

Submodules

accounting.report.utils.base_page_params module

The page parameters of a report.

```
class accounting.report.utils.base_page_params.BasePageParams
```

Bases: ABC

The base HTML page parameters class.

```
property csv_uri: str
```

```
abstract property has_data: bool
```

Returns whether there is any data on the page.

Returns

True if there is any data, or False otherwise.

```
property journal_entry_types: Type[JournalEntryType]
```

Returns the journal entry types.

Returns

The journal entry types.

```
abstract property report_chooser: ReportChooser
```

Returns the report chooser.

Returns

The report chooser.

accounting.report.utils.base_report module

The base report.

```
class accounting.report.utils.base_report.BaseReport
```

Bases: ABC

The base report class.

```
abstract csv() → Response
```

Returns the report as CSV for download.

Returns

The response of the report for download.

```
abstract html() → str
```

Composes and returns the report as HTML.

Returns

The report as HTML.

accounting.report.utils.csv_export module

The utilities to export the report as CSV for download.

class `accounting.report.utils.csv_export.BaseCSVRow`

Bases: `ABC`

The base CSV row.

abstract property values: `list[str | Decimal | None]`

Returns the values of the row.

Returns

The values of the row.

`accounting.report.utils.csv_export.csv_download(filename: str, rows: list[BaseCSVRow])` → `Response`

Exports the data rows as a CSV file for download.

Parameters

- **filename** – The download file name.
- **rows** – The data rows.

Returns

The response for download the CSV file.

`accounting.report.utils.csv_export.period_spec(period: Period)` → `str`

Constructs the period specification to be used in the filename.

Parameters

period – The period.

Returns

The period specification to be used in the filename.

accounting.report.utils.offset_matcher module

The forms for the unmatched offset management.

class `accounting.report.utils.offset_matcher.OffsetMatcher(currency: Currency, account: Account)`

Bases: `object`

The offset matcher.

line_items: `list[JournalEntryLineItem]`

The unapplied debits or credits and unmatched offsets.

match() → `None`

Matches the original line items with offsets.

Returns

`None`.

matched_pairs: `list[OffsetPair]`

A list of matched pairs.

property status: `str | LazyString`

Returns the match status message.

Returns

The match status message.

unapplied: `list[JournalEntryLineItem]`

The unapplied debits or credits.

unmatched: `list[JournalEntryLineItem]`

The unmatched offsets.

class `accounting.report.utils.offset_matcher.OffsetPair`(*original_line_item: JournalEntryLineItem, offset: JournalEntryLineItem*)

Bases: object

A pair of an original line item and its offset.

offset: `JournalEntryLineItem`

The offset.

original_line_item: `JournalEntryLineItem`

The original line item.

`accounting.report.utils.option_link` module

The option link.

class `accounting.report.utils.option_link.OptionLink`(*title: str, url: str, is_active: bool, fa_icon: str | None = None*)

Bases: object

An option link.

fa_icon: `str | None`

The font-awesome icon, if any.

is_active: `bool`

True if active, or False otherwise.

title: `str`

The title.

url: `str`

The URL.

`accounting.report.utils.report_choser` module

The report chooser.

This file is largely taken from the NanoParma ERP project, first written in 2021/9/16 by imacat (imacat@nanoparma.com).

```
class accounting.report.utils.report_chooser.ReportChooser(active_report: ReportType, period:  
Period | None = None, currency:  
Currency | None = None, account:  
Account | None = None)
```

Bases: object

The report chooser.

```
current_report: str | LazyString
```

The title of the current report.

```
is_search: bool
```

Whether the current report is the search page.

accounting.report.utils.report_type module

The report types.

```
class accounting.report.utils.report_type.ReportType(value, names=None, *, module=None,  
qualname=None, type=None, start=1,  
boundary=None)
```

Bases: Enum

The report types.

```
BALANCE_SHEET: str = 'balance-sheet'
```

The balance sheet.

```
INCOME_EXPENSES: str = 'income-expenses'
```

The income and expenses log.

```
INCOME_STATEMENT: str = 'income-statement'
```

The income statement.

```
JOURNAL: str = 'journal'
```

The journal.

```
LEDGER: str = 'ledger'
```

The ledger.

```
SEARCH: str = 'search'
```

The search.

```
TRIAL_BALANCE: str = 'trial-balance'
```

The trial balance.

```
UNAPPLIED: str = 'unapplied'
```

The unapplied original line items.

```
UNMATCHED: str = 'unmatched'
```

The unmatched offsets.

accounting.report.utils.unapplied module

The unapplied original line item utilities.

`accounting.report.utils.unapplied.get_accounts_with_unapplied(currency: Currency) → list[Account]`

Returns the accounts with unapplied original line items.

Parameters

currency – The currency.

Returns

The accounts with unapplied original line items.

`accounting.report.utils.unapplied.get_net_balances(currency: Currency, account: Account) → dict[int, Decimal | None]`

Returns the net balances of the unapplied line items of the account.

Parameters

- **currency** – The currency.
- **account** – The account.

Returns

The net balances of the unapplied line items of the account.

accounting.report.utils.unmatched module

The unmatched offset utilities.

`accounting.report.utils.unmatched.get_accounts_with_unmatched(currency: Currency) → list[Account]`

Returns the accounts with unmatched offsets.

Parameters

currency – The currency.

Returns

The accounts with unmatched offsets, with the “count” property set to the number of unmatched offsets.

accounting.report.utils.urls module

The utilities to get the ledger URL.

`accounting.report.utils.urls.balance_sheet_url(currency: Currency, period: Period) → str`

Returns the URL of a balance sheet.

Parameters

- **currency** – The currency.
- **period** – The period.

Returns

The URL of the balance sheet.

`accounting.report.utils.urls.income_expenses_url`(*currency*: Currency, *account*: CurrentAccount, *period*: Period) → str

Returns the URL of an income and expenses log.

Parameters

- **currency** – The currency.
- **account** – The account.
- **period** – The period.

Returns

The URL of the income and expenses log.

`accounting.report.utils.urls.income_statement_url`(*currency*: Currency, *period*: Period) → str

Returns the URL of an income statement.

Parameters

- **currency** – The currency.
- **period** – The period.

Returns

The URL of the income statement.

`accounting.report.utils.urls.journal_url`(*period*: Period) → str

Returns the URL of a journal.

Parameters

period – The period.

Returns

The URL of the journal.

`accounting.report.utils.urls.ledger_url`(*currency*: Currency, *account*: Account, *period*: Period) → str

Returns the URL of a ledger.

Parameters

- **currency** – The currency.
- **account** – The account.
- **period** – The period.

Returns

The URL of the ledger.

`accounting.report.utils.urls.trial_balance_url`(*currency*: Currency, *period*: Period) → str

Returns the URL of a trial balance.

Parameters

- **currency** – The currency.
- **period** – The period.

Returns

The URL of the trial balance.

`accounting.report.utils.urls.unapplied_url(currency: Currency, account: Account | None) → str`
Returns the URL of the unapplied original line items.

Parameters

- **currency** – The currency.
- **account** – The account, or None to list the accounts with unapplied original line items.

Returns

The URL of the unapplied original line items.

`accounting.report.utils.urls.unmatched_url(currency: Currency, account: Account | None) → str`
Returns the URL of the unmatched offset line items.

Parameters

- **currency** – The currency.
- **account** – The account, or None to list the accounts with unmatched offset line items.

Returns

The URL of the unmatched offset line items.

Module contents

The utilities for the reports.

Submodules

accounting.report.converters module

The path converters for the report management.

class `accounting.report.converters.CurrentAccountConverter`(*map: Map, *args: t.Any, **kwargs: t.Any*)

Bases: `BaseConverter`

The converter to convert the current account code from and to the corresponding account in the routes.

to_python(*value: str*) → `CurrentAccount`

Converts an account code to an account.

Parameters

value – The account code.

Returns

The corresponding account.

to_url(*value: CurrentAccount*) → str

Converts an account to account code.

Parameters

value – The account.

Returns

Its code.

class `accounting.report.converters.NeedOffsetAccountConverter`(*map: Map, *args: t.Any, **kwargs: t.Any*)

Bases: `BaseConverter`

The converter to convert the unapplied original line item account code from and to the corresponding account in the routes.

to_python(*value: str*) → *Account*

Converts an account code to an account.

Parameters

value – The account code.

Returns

The corresponding account.

to_url(*value: Account*) → *str*

Converts an account to account code.

Parameters

value – The account.

Returns

Its code.

class `accounting.report.converters.PeriodConverter`(*map: Map, *args: t.Any, **kwargs: t.Any*)

Bases: `BaseConverter`

The converter to convert the period specification from and to the corresponding period in the routes.

to_python(*value: str*) → *Period*

Converts a period specification to a period.

Parameters

value – The period specification.

Returns

The corresponding period.

to_url(*value: Period*) → *str*

Converts a period to its specification.

Parameters

value – The period.

Returns

Its specification.

accounting.report.template_filters module

The template filters for the reports.

`accounting.report.template_filters.format_amount`(*value: Decimal | None*) → *str | None*

Formats an amount for the report.

Parameters

value – The amount.

Returns

The formatted amount text.

accounting.report.views module

The views for the report management.

`accounting.report.views.bp: Blueprint = <Blueprint 'accounting-report'>`

The view blueprint for the reports.

Module contents

The report management.

`accounting.report.init_app(app: Flask, url_prefix: str) → None`

Initialize the application.

Parameters

- **app** – The Flask application.
- **url_prefix** – The URL prefix of the accounting application.

Returns

None.

2.1.7 accounting.utils package

Submodules

accounting.utils.cast module

The utilities to cast values into desired types, to avoid IDE warnings.

This module should not import any other module from the application.

`accounting.utils.cast.s(message: Any) → str`

Casts the LazyString message to the string type.

Parameters

message – The message.

Returns

The binary expression itself.

accounting.utils.current_account module

The current assets and liabilities account.

`class accounting.utils.current_account.CurrentAccount(account: Account | None = None)`

Bases: object

A current assets and liabilities account.

`CURRENT_AL_CODE: str = '0000-000'`

The account code for all current assets and liabilities.

`account: Account | None`

The actual account.

classmethod `accounts()` → list[Self]

Returns the current assets and liabilities accounts.

Returns

The current assets and liabilities accounts.

code: str

The code.

classmethod `current_assets_and_liabilities()` → Self

Returns the pseudo account for all current assets and liabilities.

Returns

The pseudo account for all current assets and liabilities.

id: int

The ID.

classmethod `sql_condition()` → BinaryExpression

Returns the SQL condition for the current assets and liabilities accounts.

Returns

The SQL condition for the current assets and liabilities accounts.

str: str

The string representation of the account.

title: str

The title.

accounting.utils.flash_errors module

The utility to flash all errors from the forms.

This module should not import any other module from the application.

`accounting.utils.flash_errors.flash_form_errors(form: FlaskForm) → None`

Flash all errors from a form recursively.

Parameters

form – The form.

Returns

None.

accounting.utils.journal_entry_types module

The journal entry types.

class `accounting.utils.journal_entry_types.JournalEntryType`(*value, names=None, *, module=None, qualname=None, type=None, start=1, boundary=None*)

Bases: Enum

The journal entry types.

CASH_DISBURSEMENT: str = 'disbursement'

The cash disbursement journal entry.

CASH_RECEIPT: str = 'receipt'

The cash receipt journal entry.

TRANSFER: str = 'transfer'

The transfer journal entry.

accounting.utils.next_uri module

The utilities to handle the next URI.

This module should not import any other module from the application.

`accounting.utils.next_uri.append_next(uri: str) → str`

Appends the current URI as the next URI to the query argument.

Parameters

uri – The URI.

Returns

The URI with the current URI appended as the next URI.

`accounting.utils.next_uri.decode_next(uri: str) → str`

Decodes the encoded next URI.

Parameters

uri – The encoded next URI.

Returns

The next URI.

`accounting.utils.next_uri.encode_next(uri: str) → str`

Encodes the next URI.

Parameters

uri – The next URI.

Returns

The encoded next URI.

`accounting.utils.next_uri.inherit_next(uri: str) → str`

Inherits the current next URI to the query argument, if exists.

Parameters

uri – The URI.

Returns

The URI with the current next URI added at the query argument.

`accounting.utils.next_uri.init_app(bp: Blueprint) → None`

Initializes the application.

Parameters

bp – The blueprint of the accounting application.

Returns

None.

`accounting.utils.next_uri.or_next(uri: str) → str`

Returns the next URI, if exists, or the supplied URI.

Parameters

uri – The URI.

Returns

The next URI or the supplied URI.

accounting.utils.offset_alias module

The SQLAlchemy alias for the offset items.

`accounting.utils.offset_alias.offset_alias()` → Alias

Returns the SQLAlchemy alias for the offset items.

Returns

The SQLAlchemy alias for the offset items.

accounting.utils.options module

The getter and setter for the option management.

class `accounting.utils.options.Options`

Bases: object

The options.

`commit()` → None

Commits the options to the database.

Returns

None.

property `default_currency`: *Currency*

Returns the default currency.

Returns

The default currency.

property `default_currency_code`: **str**

Returns the default currency code.

Returns

The default currency code.

property `default_ie_account`: *CurrentAccount*

Returns the default account for the income and expenses log.

Returns

The default account for the income and expenses log.

property `default_ie_account_code`: **str**

Returns the default account code for the income and expenses log.

Returns

The default account code for the income and expenses log.

is_modified: **bool**

Whether the options were modified.

property recurring: *Recurring*

Returns the recurring expenses and incomes.

Returns

The recurring expenses and incomes.

property recurring_data: `dict[str, list[tuple[str, str, str]]]`

Returns the data of the recurring expenses and incomes.

Returns

The data of the recurring expenses and incomes.

class `accounting.utils.options.Recurring`(*data: dict[str, list[tuple[str, str, str]]]*)

Bases: object

The recurring expenses or incomes.

property codes: `set[str]`

Returns all the account codes.

Returns

All the account codes.

expenses: `list[RecurringItem]`

The recurring expenses.

incomes: `list[RecurringItem]`

The recurring incomes.

class `accounting.utils.options.RecurringItem`(*name: str, account_code: str, description_template: str*)

Bases: object

A recurring item.

account_code: `str`

The account code.

property account_text: `str`

Returns the account text.

Returns

The account text.

description_template: `str`

The description template.

name: `str`

The name.

`accounting.utils.options.options:` *Options* = <accounting.utils.options.Options object>

The options.

accounting.utils.pagination module

The pagination utilities.

This module should not import any other module from the application.

class `accounting.utils.pagination.AbstractPagination`

Bases: `Generic[T]`

An abstract pagination.

is_paged: `bool`

Whether there should be pagination.

list: `list[T]`

The items shown in the list

page_size: `int`

The number of items in a page.

page_size_options: `list[Link]`

The options to the number of items in a page.

pages: `list[Link]`

The pages.

`accounting.utils.pagination.DEFAULT_PAGE_SIZE:` `int = 10`

The default page size.

class `accounting.utils.pagination.EmptyPagination`

Bases: `AbstractPagination[T]`

The pagination from empty data.

class `accounting.utils.pagination.Link`(*text: str, uri: str | None = None, is_current: bool = False, is_for_mobile: bool = False*)

Bases: `object`

A link.

is_current: `bool`

Whether the link is the current page.

is_for_mobile: `bool`

Whether the link should be shown on mobile screens.

text: `str`

The link text

uri: `str | None`

The link URI, or None if there is no link.

class `accounting.utils.pagination.NonEmptyPagination`(*items: list[T], is_reversed: bool = False*)

Bases: `AbstractPagination[T]`

The pagination with real data.

PAGE_SIZE_OPTION_VALUES: `list[int] = [10, 100, 200]`

The page size options.

class `accounting.utils.pagination.Pagination`(*items: list[T], is_reversed: bool = False*)

Bases: `Generic[T]`

The pagination utility.

is_paged: `bool`

Whether there should be pagination.

list: `list[T]`

The items shown in the list

page_size: `int`

The number of items in a page.

page_size_options: `list[Link]`

The options to the number of items in a page.

pages: `list[Link]`

The pages.

exception `accounting.utils.pagination.Redirection`(*new_url: str*)

Bases: `RequestRedirect`

The redirection.

code: `int | None = 302`

The HTTP code.

class `accounting.utils.pagination.T`

The pagination item type.

alias of `TypeVar('T')`

accounting.utils.permission module

The permissions.

This module should not import any other module from the application.

`accounting.utils.permission.can_admin()` → `bool`

Returns whether the current user can administrate the accounting settings.

The user has to log in.

Returns

True if the current user can administrate the accounting settings, or False otherwise.

`accounting.utils.permission.can_edit()` → `bool`

Returns whether the current user can edit the accounting data.

The user has to log in.

Returns

True if the current user can edit the accounting data, or False otherwise.

`accounting.utils.permission.can_view()` → `bool`

Returns whether the current user can view the accounting data.

Returns

True if the current user can view the accounting data, or False otherwise.

`accounting.utils.permission.has_permission(rule: Callable[[], bool]) → Callable`

The permission decorator to check whether the current user is allowed.

Parameters

rule – The permission rule.

Returns

The view decorator.

`accounting.utils.permission.init_app(bp: Blueprint, user_utils: UserUtilityInterface) → None`

Initializes the application.

Parameters

- **bp** – The blueprint of the accounting application.
- **user_utils** – The user utilities.

Returns

None.

accounting.utils.query module

The query keyword parser.

This module should not import any other module from the application.

`accounting.utils.query.parse_query_keywords(q: str | None) → list[str]`

Returns the query keywords by the query parameter.

Parameters

q – The query parameter.

Returns

The query keywords.

accounting.utils.random_id module

The random ID utility for the data models.

This module should not import any other module from the application.

`accounting.utils.random_id.new_id(cls: Type[Model])`

Generates and returns a new, unused random ID for the data model.

Parameters

cls – The data model.

Returns

The newly-generated, unused random ID.

accounting.utils.strip_text module

The text stripper for the form fields.

This module should not import any other module from the application.

`accounting.utils.strip_text.strip_multiline_text(s: str | None) → str | None`

The filter to strip a piece of multi-line text.

Parameters

s – The text input string.

Returns

The filtered string.

`accounting.utils.strip_text.strip_text(s: str | None) → str | None`

The filter to strip the leading and trailing white spaces of text.

Parameters

s – The text input string.

Returns

The filtered string.

accounting.utils.title_case module

The title case capitalization for the base account titles. This follows the APA style title case capitalization. See <https://apastyle.apa.org/style-grammar-guidelines/capitalization/title-case>.

This module should not import any other module from the application.

`accounting.utils.title_case.ARTICLES: set[str] = {'a', 'an', 'the'}`

Articles.

`accounting.utils.title_case.CONJUNCTIONS: set[str] = {'and', 'as', 'but', 'for', 'if', 'nor', 'or', 'so', 'yet'}`

Short conjunctions.

`accounting.utils.title_case.MINOR_WORDS: set[str] = {'a', 'an', 'and', 'as', 'at', 'but', 'for', 'if', 'in', 'nor', 'of', 'on', 'or', 'per', 'so', 'the', 'to', 'up', 'via', 'yet'}`

Minor words that should be in lowercase.

`accounting.utils.title_case.PREPOSITIONS: set[str] = {'as', 'at', 'by', 'for', 'in', 'of', 'on', 'per', 'to', 'up', 'via'}`

Short prepositions.

`accounting.utils.title_case.title_case(s: str) → str`

Capitalize a title string for the base account titles. Do not use it in other places. This excludes “by” as in “1223 by-products”.

Parameters

s – The title string.

Returns

The capitalized title string.

accounting.utils.user module

The user utilities.

This module should not import any other module from the application.

class `accounting.utils.user.T`

The user data model data type.

alias of `TypeVar('T', bound=Model)`

class `accounting.utils.user.UserUtilityInterface`

Bases: `Generic[T], ABC`

The interface for the user utilities.

abstract `can_admin()` → bool

Returns whether the currently logged-in user can administrate the accounting settings.

Returns

True if the currently logged-in user can administrate the accounting settings, or False otherwise.

abstract `can_edit()` → bool

Returns whether the currently logged-in user can edit the accounting data.

Returns

True if the currently logged-in user can edit the accounting data, or False otherwise.

abstract `can_view()` → bool

Returns whether the currently logged-in user can view the accounting data.

Returns

True if the currently logged-in user can view the accounting data, or False otherwise.

abstract `property cls: Type[T]`

Returns the class of the user data model.

Returns

The class of the user data model.

abstract `property current_user: T | None`

Returns the currently logged-in user.

Returns

The currently logged-in user, or None if the user has not logged in

abstract `get_by_username(username: str) → T | None`

Returns the user by her username.

Returns

The user by her username, or None if the user was not found.

abstract `get_pk(user: T) → int`

Returns the primary key of the user, as an integer.

Returns

The primary key of the user, as an integer.

abstract property pk_column: `Column`

Returns the primary key column.

Returns

The primary key column.

abstract unauthorized() `→ Response | None`

Returns the response to require the user to log in.

This may be a redirection to the login page, or an HTTP 401 Unauthorized response for HTTP Authentication. If this returns None, an HTTP 403 Forbidden response is return to the user.

Returns

The response to require the user to log in.

`accounting.utils.user.get_current_user()` `→ Model | None`

Returns the currently logged-in user. The result is cached in the current request.

Returns

The currently logged-in user.

`accounting.utils.user.get_current_user_pk()` `→ int`

Returns the primary key value of the currently logged-in user.

Returns

The primary key value of the currently logged-in user.

`accounting.utils.user.get_user_pk(username: str)` `→ int`

Returns the primary key value of the user by the username.

Parameters

username – The username.

Returns

The primary key value of the user by the username.

`accounting.utils.user.has_user(username: str)` `→ bool`

Returns whether a user by the username exists.

Parameters

username – The username.

Returns

True if the user by the username exists, or False otherwise.

`accounting.utils.user.init_user_utils(utils: UserUtilityInterface)` `→ None`

Initializes the user utilities.

Parameters

utils – The user utilities.

Returns

None.

`accounting.utils.user.user_cls`

The user class.

`accounting.utils.user.user_pk_column:` `Column = Column(None, Integer(), table=None)`

The primary key column of the user class.

Module contents

The independent utilities.

This module should not import any other module from the application.

2.2 Submodules

2.3 accounting.commands module

The console commands.

2.4 accounting.forms module

The forms.

```
accounting.forms.ACCOUNT_REQUIRED: DataRequired = <wtforms.validators.DataRequired object>
```

The validator to check if the account code is empty.

```
class accounting.forms.AccountExists
```

Bases: object

The validator to check if the account exists.

```
class accounting.forms.CurrencyExists
```

Bases: object

The validator to check if the account exists.

```
class accounting.forms.IsCreditAccount(message: str | LazyString)
```

Bases: object

The validator to check if the account is for credit line items.

```
class accounting.forms.IsDebitAccount(message: str | LazyString)
```

Bases: object

The validator to check if the account is for debit line items.

2.5 accounting.locale module

The localization for the accounting application.

```
accounting.locale.domain: Domain = <Domain([PosixPath('/home/docs/checkouts/readthedocs.org/user_builds/mia-accounting/checkouts/latest/src/accounting/translations')], ['accounting'])>
```

The message domain.

`accounting.locale.gettext(string, **variables) → str`

A replacement of the Babel `gettext()` function..

Parameters

- **string** – The message to translate.
- **variables** – The variable substitution.

Returns

The translated message.

`accounting.locale.init_app(app: Flask, bp: Blueprint) → None`

Initializes the application.

Parameters

- **app** – The Flask application.
- **bp** – The blueprint of the accounting application.

Returns

None.

`accounting.locale.lazy_gettext(string, **variables) → LazyString`

A replacement of the Babel `lazy_gettext()` function..

Parameters

- **string** – The message to translate.
- **variables** – The variable substitution.

Returns

The translated message.

`accounting.locale.pgettext(context, string, **variables) → str`

A replacement of the Babel `gettext()` function..

Parameters

- **context** – The context.
- **string** – The message to translate.
- **variables** – The variable substitution.

Returns

The translated message.

`accounting.locale.translation_dir: Path = PosixPath('/home/docs/checkouts/readthedocs.org/user_builds/mia-accounting/checkouts/latest/src/accounting/translations')`

The directory of the translation files.

2.6 accounting.models module

The data models.

```
class accounting.models.Account(**kwargs)
    Bases: Model
    An account.
    ACCUMULATED_CHANGE_CODE: str = '3351-001'
        The code of the accumulated-change account,
    CASH_CODE: str = '1111-001'
        The code of the cash account,
    NET_CHANGE_CODE: str = '3353-001'
        The code of the net-change account,
    classmethod accumulated_change() → Self
        Returns the accumulated-change account.
        Returns
            The accumulated-change account
    base: Mapped[BaseAccount]
        The base account.
    base_code: Mapped[str]
        The code of the base account.
    property can_delete: bool
        Returns whether the account can be deleted.
        Returns
            True if the account can be deleted, or False otherwise.
    classmethod cash() → Self
        Returns the cash account.
        Returns
            The cash account
    property code: str
        Returns the code.
        Returns
            The code.
    property count: int
        Returns the number of items in the account.
        Returns
            The number of items in the account.
    created_at: Mapped[dt.datetime]
        The date and time when this record was created.
    created_by: Mapped[user_cls]
        The user who created the record.
```


created_by_id: `Mapped[int]`

The ID of the user who created the record.

delete() → `None`

Deletes this account.

Returns

`None`.

classmethod find_by_code(*code: str*) → `Self | None`

Finds an account by its code.

Parameters

code – The code.

Returns

The account, or `None` if this account does not exist.

id: `Mapped[int]`

The account ID.

property is_modified: `bool`

Returns whether a product account was modified.

Returns

True if modified, or False otherwise.

is_need_offset: `Mapped[bool]`

Whether the journal entry line items of this account need offset.

property is_nominal: `bool`

Returns whether the account is a nominal account.

Returns

True if the account is a nominal account, or False otherwise.

property is_real: `bool`

Returns whether the account is a real account.

Returns

True if the account is a real account, or False otherwise.

l10n: `Mapped[list[AccountL10n]]`

The localized titles.

line_items: `Mapped[list[JournalEntryLineItem]]`

The journal entry line items.

no: `Mapped[int]`

The account number under the base account.

property query_values: `list[str]`

Returns the values to be queried.

Returns

The values to be queried.

classmethod selectable_credit() → `list[Self]`

Returns the selectable debit accounts. Receivable line items can not start from credit.

Returns

The selectable debit accounts.

classmethod selectable_debit() → list[Self]

Returns the selectable debit accounts. Payable line items can not start from debit.

Returns

The selectable debit accounts.

property title: str

Returns the title in the current locale.

Returns

The title in the current locale.

title_l10n: Mapped[str]

The title.

updated_at: Mapped[dt.datetime]

The date and time when this record was last updated.

updated_by: Mapped[user_cls]

The last user who updated the record.

updated_by_id: Mapped[int]

The ID of the last user who updated the record.

class accounting.models.AccountL10n(**kwargs)

Bases: Model

A localized account title.

account: Mapped[Account]

The account.

account_id: Mapped[int]

The account ID.

locale: Mapped[str]

The locale.

title: Mapped[str]

The localized title.

class accounting.models.BaseAccount(**kwargs)

Bases: Model

A base account.

accounts: Mapped[list[Account]]

The descendant accounts under the base account.

code: Mapped[str]

The account code.

l10n: Mapped[list[BaseAccountL10n]]

The localized titles.

property query_values: list[str]

Returns the values to be queried.

Returns

The values to be queried.

property title: str

Returns the title in the current locale.

Returns

The title in the current locale.

title_l10n: Mapped[str]

The title.

class accounting.models.BaseAccountL10n(**kwargs)

Bases: Model

A localized base account title.

account: Mapped[BaseAccount]

The account.

account_code: Mapped[str]

The account code.

locale: Mapped[str]

The locale.

title: Mapped[str]

The localized title.

class accounting.models.Currency(**kwargs)

Bases: Model

A currency.

property can_delete: bool

Returns whether the currency can be deleted.

Returns

True if the currency can be deleted, or False otherwise.

code: Mapped[str]

The currency code.

created_at: Mapped[dt.datetime]

The date and time when this record was created.

created_by: Mapped[user_cls]

The user who created the record.

created_by_id: Mapped[int]

The ID of the user who created the record.

delete() → None

Deletes the currency.

Returns

None.

property is_modified: bool

Returns whether a product account was modified.

Returns

True if modified, or False otherwise.

l10n: Mapped[list[CurrencyL10n]]

The localized names.

line_items: Mapped[list[JournalEntryLineItem]]

The journal entry line items.

property name: str

Returns the name in the current locale.

Returns

The name in the current locale.

name_l10n: Mapped[str]

The currency name.

updated_at: Mapped[dt.datetime]

The date and time when this record was last updated.

updated_by: Mapped[user_cls]

The last user who updated the record.

updated_by_id: Mapped[int]

The ID of the last user who updated the record.

class accounting.models.CurrencyL10n(**kwargs)

Bases: Model

A localized currency name.

currency: Mapped[Currency]

The currency.

currency_code: Mapped[str]

The currency code.

locale: Mapped[str]

The locale.

name: Mapped[str]

The localized name.

class accounting.models.JournalEntry(**kwargs)

Bases: Model

A journal entry.

property can_delete: bool

Returns whether the journal entry can be deleted.

Returns

True if the journal entry can be deleted, or False otherwise.

created_at: Mapped[dt.datetime]

The date and time when this record was created.

created_by: `Mapped[user_cls]`

The user who created the record.

created_by_id: `Mapped[int]`

The ID of the user who created the record.

property currencies: `list[JournalEntryCurrency]`

Returns the line items categorized by their currencies.

Returns

The currency categories.

date: `Mapped[dt.date]`

The date.

delete() → None

Deletes the journal entry.

Returns

None.

id: `Mapped[int]`

The journal entry ID.

property is_cash_disbursement: `bool`

Returns whether this is a cash disbursement journal entry.

Returns

True if this is a cash disbursement journal entry, or False otherwise.

property is_cash_receipt: `bool`

Returns whether this is a cash receipt journal entry.

Returns

True if this is a cash receipt journal entry, or False otherwise.

line_items: `Mapped[list[JournalEntryLineItem]]`

The line items.

no: `Mapped[int]`

The journal entry number under the date.

note: `Mapped[str | None]`

The note.

updated_at: `Mapped[dt.datetime]`

The date and time when this record was last updated.

updated_by: `Mapped[user_cls]`

The last user who updated the record.

updated_by_id: `Mapped[int]`

The ID of the last user who updated the record.

class `accounting.models.JournalEntryCurrency`(*code: str, debit: list[JournalEntryLineItem], credit: list[JournalEntryLineItem]*)

Bases: `object`

A currency in a journal entry.

code: str

The currency code.

credit: list[JournalEntryLineItem]

The credit line items.

property credit_total: str

Returns the total amount of the credit line items.

Returns

The total amount of the credit line items.

debit: list[JournalEntryLineItem]

The debit line items.

property debit_total: Decimal

Returns the total amount of the debit line items.

Returns

The total amount of the debit line items.

property name: str

Returns the currency name.

Returns

The currency name.

class accounting.models.JournalEntryLineItem(**kwargs)

Bases: Model

A line item in the journal entry.

account: Mapped[Account]

The account.

property account_code: str

Returns the account code.

Returns

The account code.

account_id: Mapped[int]

The account ID.

amount: Mapped[Decimal]

The amount.

property balance: Decimal

Returns the balance.

Returns

The balance.

property credit: Decimal | None

Returns the credit amount.

Returns

The credit amount, or None if this is not a credit line item.

currency: `Mapped[Currency]`

The currency.

currency_code: `Mapped[str]`

The currency code.

property debit: `Decimal | None`

Returns the debit amount.

Returns

The debit amount, or None if this is not a debit line item.

description: `Mapped[str | None]`

The description.

id: `Mapped[int]`

The line item ID.

is_debit: `Mapped[bool]`

True for a debit line item, or False for a credit line item.

property is_need_offset: `bool`

Returns whether the line item needs offset.

Returns

True if the line item needs offset, or False otherwise.

property is_offset: `bool`

Returns whether the line item is an offset.

Returns

True if the line item is an offset, or False otherwise.

journal_entry: `Mapped[JournalEntry]`

The journal entry.

journal_entry_id: `Mapped[int]`

The journal entry ID.

property match: `Self | None`

Returns the match of the line item.

Returns

The match of the line item.

property net_balance: `Decimal`

Returns the net balance.

Returns

The net balance.

no: `Mapped[int]`

The line item number under the journal entry and debit or credit.

property offsets: `list[Self]`

Returns the offset items.

Returns

The offset items.

original_line_item: `Mapped[JournalEntryLineItem | None]`

The original line item.

original_line_item_id: `Mapped[int | None]`

The ID of the original line item.

property query_values: `list[str]`

Returns the values to be queried.

Returns

The values to be queried.

class `accounting.models.Option(**kwargs)`

Bases: `Model`

An option.

created_at: `Mapped[dt.datetime]`

The date and time when this record was created.

created_by: `Mapped[user_cls]`

The user who created the record.

created_by_id: `Mapped[int]`

The ID of the user who created the record.

name: `Mapped[str]`

The name.

updated_at: `Mapped[dt.datetime]`

The date and time when this record was last updated.

updated_by: `Mapped[user_cls]`

The last user who updated the record.

updated_by_id: `Mapped[int]`

The ID of the last user who updated the record.

value: `Mapped[str]`

The option value.

2.7 accounting.template_filters module

The template filters.

`accounting.template_filters.default(value: Any, default_value: Any = "") → Any`

Returns the default value if the given value is None.

Parameters

- **value** – The value.
- **default_value** – The default value when the given value is None.

Returns

The value, or the default value if the given value is None.

`accounting.template_filters.format_amount(value: Decimal | None) → str | None`

Formats an amount for readability.

Parameters

value – The amount.

Returns

The formatted amount text.

`accounting.template_filters.format_date(value: date) → str`

Formats a date to be human-friendly.

Parameters

value – The date.

Returns

The human-friendly date text.

2.8 accounting.template_globals module

The template globals.

`accounting.template_globals.currency_options() → list[Currency]`

Returns the currency options.

Returns

The currency options.

`accounting.template_globals.default_currency_code() → str`

Returns the default currency code.

Returns

The default currency code.

2.9 Module contents

The accounting application.

`accounting.VERSION: str = '1.5.11'`

The package version.

`accounting.data_dir: Path = PosixPath('/home/docs/checkouts/readthedocs.org/user_builds/mia-accounting/checkouts/latest/src/accounting/data')`

The data directory.

`accounting.db: SQLAlchemy = <SQLAlchemy>`

The database instance.

`accounting.init_app(app: Flask, user_utils: UserUtilityInterface, url_prefix: str = '/accounting') → None`

Initialize the application.

Parameters

- **app** – The Flask application.
- **user_utils** – The user utilities.

- **url_prefix** – The URL prefix of the accounting application.

Returns

None.

3.1 An Example Configuration

The following is an example configuration for *Mia! Accounting*.

```
from flask import Response, redirect
from .auth import current_user()
from .modules import User

def create_app(test_config=None) -> Flask:
    app: Flask = Flask(__name__)

    ... (Configuration of SQLAlchemy, CSRF, Babel_JS, ... etc) ...

    import accounting

    class UserUtils(accounting.UserUtilityInterface[User]):

        def can_view(self) -> bool:
            return True

        def can_edit(self) -> bool:
            return "editor" in current_user().roles

        def can_admin(self) -> bool:
            return current_user().is_admin

        def unauthorized(self) -> Response:
            return redirect("/login")

        @property
        def cls(self) -> t.Type[User]:
            return User

        @property
        def pk_column(self) -> Column:
            return User.id

        @property
        def current_user(self) -> User | None:
            return current_user()
```

(continues on next page)

(continued from previous page)

```
def get_by_username(self, username: str) -> User | None:
    return User.query.filter(User.username == username).first()

def get_pk(self, user: User) -> int:
    return user.id

accounting.init_app(app, UserUtils())

... (Any other configuration) ...

return app
```

HISTORY

I created my own private accounting application in [Perl/mod_perl](#) in 2007, as part of my personal website. The first revision was made using [Perl/Mojolicious](#) in 2019, with the aim of making it mobile-friendly using [Bootstrap](#), and with modern back-end and front-end technologies such as [jQuery](#).

The second revision was done in [Python/Django](#) in 2020, as I was looking to change my career from [PHP/Laravel](#) to Python, but lacked experience with large Python projects. I needed something in my portfolio and decided to work on the somewhat outdated [Mojolicious](#) project.

Despite having no prior experience with Django, I spent two months working late nights to create the [Mia! Accounting Django](#) application. It took me another 1.5 months to make it an independent module, which I later released as an open source project on PyPI.

The application worked nicely for my household bookkeeping for two years. However, new demands arose over time, especially with tracking payables and receivables. This was critical [during the pandemic](#) as more payments were made online with credit cards.

The biggest issue I encountered was with [Django's MTV architectural pattern](#). Django takes over the control flow. I had to override several parts of the [class-based views](#) for different but yet simple control flow logic. In the end, it became very difficult to track whether things went wrong because I overrode something or because it just wouldn't work with the basic assumption of the class-based views. By the time I realized it, it was too late for me to drop Django's MTV and rewrite everything from class-based views to function-based views.

Therefore, I decided to turn to [microframeworks](#) like [Flask](#). After working with modularized Flask and [FastAPI](#) applications for two years, I returned to the project and wrote its third revision using Flask in 2023.

CHANGE LOG

5.1 Version 1.5.11

Released 2023/12/26

Bug fix.

- Refined to enable the selection of the 3351-001 Accumulated Profit or Loss account.

5.2 Version 1.5.10

Released 2023/11/28

Bug fix.

- Fixed the form validator to enable the selection of Accumulated Profit or Loss accounts other than 3351-001.

5.3 Version 1.5.9

Released 2023/11/28

Bug fix.

- Refined to enable the selection of Accumulated Profit or Loss accounts other than 3351-001, facilitating the consolidation of existing balances.

5.4 Version 1.5.8

Released 2023/10/24

Bug fix.

- Fixed an icon in the detail of the cash receipt journal entry.

Released at Jaipur, India on vacation.

5.5 Version 1.5.7

Released 2023/7/29

Revised account title capitalization to capitalize account titles upon initialization of base accounts, rather than when displaying the accounts. This prevents the system from incorrectly capitalizing titles of user-added accounts.

For existing installation, run the `accounting-titleize` console command to capitalize the existing account titles that were already initialized.

Other fixes:

- Added missing documentation to the global variables, class properties, and object properties.
- Various minor fixes.

5.6 Version 1.5.6

Released 2023/5/23

Bug fixes.

- Fixed the return URI of the creation forms to decode the next URI.
- Fixed the unmatched offset list to use the encoded next URI.

5.7 Version 1.5.5

Released 2023/5/23

Security fixes.

- Revised the next URI utilities to encode and decode the next URI preventing tampering with the next URI.
- Added the integrity value of the CDN stylesheet links.
- Various fixes.

5.8 Version 1.5.4

Released 2023/5/18

Security fixes.

- Added safeguard to the next URI utilities, to prevent Cross-Site Scripting (XSS) attacks.
- Applied the safe next URI utilities to the test site.
- Added the `SameSite` and `Secure` flags to the session cookie of the test site.

5.9 Version 1.5.3

Released 2023/4/30

- Fixed the error of the net balance in the unmatched offset list.
- Revised the original line item editor not to override the existing amount when the existing amount is less or equal to the net balance.

5.10 Version 1.5.2

Released 2023/4/30

- Fixed the error of the net balance in the unmatched offset list.

5.11 Version 1.5.1

Released 2023/4/30

- Fixed the error calling the old `setEnabledDescriptionAccount` method in the `saveOriginalLineItem` method of the JavaScript `JournalEntryLineItemEditor` class.

5.12 Version 1.5.0

Released 2023/4/23

- Updated to require `SQLAlchemy >= 2`.
- Added the change log.
- Added the `VERSION` constant to the `accounting` module for the package version, and revised `pyproject.toml` and `conf.py` to read the version from it.

5.13 Version 1.4.1

Released 2023/4/22

- Updated to allow editing the description of the journal entry line item with offsets or are offsetting to original line items.
- Updated not to override the existing description of a journal entry line item after choosing the original line item to offset to.

5.14 Version 1.4.0

Released 2023/4/18

- Rewrote the unapplied original line items and unmatched offsets.
 - The unapplied original line items and unmatched offsets are both in the report submodule. They can be filtered with currency and period now.
 - Show the unapplied original line items and unmatched offsets together, and added the accumulated balance in the unmatched offset list, for ease of reference.
- Removed the account code from the journal entry detail and journal entry form for mobile devices.
- Made the account options in the reports to be scrollable.

5.15 Version 1.3.3

Released 2023/4/13

Changed the sample data generation in the test site live demonstration from pre-recorded data to real-time generation, to avoid the problem with the start of months and weeks changed with the date of the import.

5.16 Version 1.3.2

Released 2023/4/12

Added the sample data generation and database reset on the test site for live demonstration.

5.17 Version 1.3.1

Released 2023/4/11

- Fixed the permission of the navigation menu of the unmatched offsets.
- Revised the test site to be more accessible as the live demonstration.

5.18 Version 1.3.0

Released 2023/4/11

Added the `accounting-init-db` console command to replace all the other console commands to initialize the accounting database. The test site does not work with previous versions (<1.3.0).

5.19 Version 1.2.1

Released 2023/4/9

Fixed the search result to allow full year/month/day specification.

5.20 Version 1.2.0

Released 2023/4/9

- Simplified the URL of the default reports.
- Fixed the crash with malformed Chinese translation.
- Fixed the crash when downloading CSV data with non-US-ASCII filenames.

5.21 Version 1.1.0

Released 2023/4/9

- Added the unapplied original line item list, to track unpaid payables, unreceived receivables, assets, prepaids, refundable deposits, etc.
- Added the offset matcher to match unapplied original line items with unmatched offsets.

5.22 Version 1.0.1

Released 2023/4/6

Documentation fixes.

5.23 Version 1.0.0

Released 2023/4/6

The first formal release in Flask.

Added the documentation.

5.24 Version 0.11.1 (Pre-release)

Released 2023/4/5

Removed the zero balances from the trial balance, the income statement, and the balance sheet.

5.25 Version 0.11.0 (Pre-release)

Released 2023/4/5

- Renamed the project from `mia-accounting-flask` to `mia-accounting`.
- Updated the URL of the reports, as the default views of the accounting application.
- Updated README.
- Various fixes.

5.26 Version 0.10.0 (Pre-release)

Released 2023/4/3

- Added the `unauthorized` method to the `UserUtilityInterface` interface to allow fine control to how to handle the case when the user has not logged in.
- Revised the JavaScript description editor to respect the account that the user has confirmed or specifically selected.
- Various fixes.

5.27 Version 0.9.1 (Pre-release)

Released 2023/3/24

- A distinguishable look in the option detail than the option form.
- A better look in the new journal entry forms when there is no line item yet.
- Fixed the search in the original entry selector in the journal entry form to always do a partial match, to fix the problem that there is no match when typing is not finished yet.
- Fixed the search in the original entry selector to search the net balance correctly.
- Replaced the `editor` and `editor2` accounts with the `admin` and `editor` accounts.
- Various fixes.

5.28 Version 0.9.0 (Pre-release)

Released 2023/3/23

Moved the settings from the `.env` file to the option table in the database that can be set and updated on the web interface. Added the settings page to show and update the settings.

5.29 Version 0.8.0 (Pre-release)

Released 2023/3/22

- Added the recurring transactions to the description editor.
- Added prevention to delete database objects that are essential or referenced by others with foreign keys.
- Various fixes on the visual layout.

5.30 Version 0.7.0 (Pre-release)

Released 2023/3/21

- Renamed “transaction” to “journal entry”, and “journal entry” to “journal entry line item”.
- Renamed `summary` to `description`.
- Updated `tempus-dominus` from version 6.2.10 to 6.4.3.
- Fixed titles and capitalization.
- Fixed to search case-insensitively.
- Added favicon to the test site.
- Fixed the navigation menu when there is no matching endpoint.
- Various fixes.

5.31 Version 0.6.0 (Pre-release)

Released 2023/3/18

- Added offset tracking to the journal entries in the payable and receivable accounts.
- Renamed the `is_offset_needed` column to `is_need_offset` in the Account data model.

5.32 Version 0.5.0 (Pre-release)

Released 2023/3/10

Added the accounting reports.

5.33 Version 0.4.0 (Pre-release)

Released 2023/3/1

Added the transaction summary helper.

5.34 Version 0.3.1 (Pre-release)

Released 2023/2/28

- Fixed the error that cannot select any account when adding new transactions.
- Fixed the database error when adding new transactions.
- Added the button to convert a cash income or cash expense transaction to a transfer transaction.

5.35 Version 0.3.0 (Pre-release)

Released 2023/2/27

Added the transaction management.

5.36 Version 0.2.0 (Pre-release)

Released 2023/2/7

- Added the currency management.
- Changed the `can_edit` permission to at least require the user to log in first.
- Changed the type hint of the `current_user` pseudo property of the `AbstractUserUtils` class to return `None` when the user has not logged in.

5.37 Version 0.1.1 (Pre-release)

Released 2023/2/3

Finalized the account management, with tests and reordering.

5.38 Version 0.1.0 (Pre-release)

Released 2023/2/3

Added the account management, and updated the API to initialize the accounting application.

5.39 Version 0.0.0 (Pre-release)

Released 2023/2/3

Initial release with main account list, localization, pagination, query, permission, Sphinx documentation, and a test case based on a test demonstration site.

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

a

accounting, 93
accounting.account, 6
accounting.account.commands, 3
accounting.account.converters, 3
accounting.account.forms, 4
accounting.account.queries, 5
accounting.account.views, 5
accounting.base_account, 7
accounting.base_account.commands, 6
accounting.base_account.converters, 6
accounting.base_account.queries, 7
accounting.base_account.views, 7
accounting.commands, 82
accounting.currency, 9
accounting.currency.commands, 7
accounting.currency.converters, 7
accounting.currency.forms, 8
accounting.currency.queries, 9
accounting.currency.views, 9
accounting.forms, 82
accounting.journal_entry, 32
accounting.journal_entry.converters, 30
accounting.journal_entry.forms, 21
accounting.journal_entry.forms.currency, 9
accounting.journal_entry.forms.journal_entry, 13
accounting.journal_entry.forms.line_item, 16
accounting.journal_entry.forms.reorder, 20
accounting.journal_entry.template_filters, 31
accounting.journal_entry.utils, 29
accounting.journal_entry.utils.account_option, 21
accounting.journal_entry.utils.description_editor, 22
accounting.journal_entry.utils.operators, 25
accounting.journal_entry.utils.original_line_items, 29
accounting.journal_entry.views, 32
accounting.locale, 82
accounting.models, 84
accounting.option, 35
accounting.option.forms, 32
accounting.option.views, 35
accounting.report, 71
accounting.report.converters, 69
accounting.report.period, 40
accounting.report.period.chooser, 36
accounting.report.period.description, 37
accounting.report.period.month_end, 37
accounting.report.period.parser, 37
accounting.report.period.period, 38
accounting.report.period.shortcuts, 39
accounting.report.period.specification, 40
accounting.report.reports, 63
accounting.report.reports.balance_sheet, 40
accounting.report.reports.income_expenses, 43
accounting.report.reports.income_statement, 46
accounting.report.reports.journal, 48
accounting.report.reports.ledger, 50
accounting.report.reports.search, 53
accounting.report.reports.trial_balance, 54
accounting.report.reports.unapplied, 56
accounting.report.reports.unapplied_accounts, 58
accounting.report.reports.unmatched, 59
accounting.report.reports.unmatched_accounts, 61
accounting.report.template_filters, 70
accounting.report.utils, 69
accounting.report.utils.base_page_params, 63
accounting.report.utils.base_report, 63
accounting.report.utils.csv_export, 64
accounting.report.utils.offset_matcher, 64
accounting.report.utils.option_link, 65
accounting.report.utils.report_chooser, 65
accounting.report.utils.report_type, 66
accounting.report.utils.unapplied, 67
accounting.report.utils.unmatched, 67
accounting.report.utils.urls, 67
accounting.report.views, 71
accounting.template_filters, 92
accounting.template_globals, 93

accounting.utils, 82
accounting.utils.cast, 71
accounting.utils.current_account, 71
accounting.utils.flash_errors, 72
accounting.utils.journal_entry_types, 72
accounting.utils.next_uri, 73
accounting.utils.offset_alias, 74
accounting.utils.options, 74
accounting.utils.pagination, 76
accounting.utils.permission, 77
accounting.utils.query, 78
accounting.utils.random_id, 78
accounting.utils.strip_text, 79
accounting.utils.title_case, 79
accounting.utils.user, 80

INDEX

A

- AbstractPagination (class in accounting.utils.pagination), 76
- account (accounting.journal_entry.utils.description_editor.DescriptionRecurring attribute), 23
- account (accounting.models.AccountL10n attribute), 86
- account (accounting.models.BaseAccountL10n attribute), 87
- account (accounting.models.JournalEntryLineItem attribute), 90
- account (accounting.report.reports.balance_sheet.ReportAccount attribute), 42
- account (accounting.report.reports.income_expenses.CSVRow attribute), 43
- account (accounting.report.reports.income_expenses.PageParams attribute), 44
- account (accounting.report.reports.income_expenses.ReportLineItem attribute), 45
- account (accounting.report.reports.income_statement.ReportAccount attribute), 47
- account (accounting.report.reports.journal.CSVRow attribute), 48
- account (accounting.report.reports.journal.ReportLineItem attribute), 50
- account (accounting.report.reports.ledger.PageParams attribute), 52
- account (accounting.report.reports.trial_balance.ReportAccount attribute), 55
- account (accounting.report.reports.unapplied.PageParams attribute), 57
- account (accounting.report.reports.unapplied_accounts.CSVRow attribute), 58
- account (accounting.report.reports.unmatched.PageParams attribute), 60
- account (accounting.report.reports.unmatched_accounts.CSVRow attribute), 62
- account (accounting.utils.current_account.CurrentAccount attribute), 71
- Account (class in accounting.models), 84
- account_code (accounting.journal_entry.forms.line_item.CreditLineItemForm attribute), 16
- account_code (accounting.journal_entry.forms.line_item.DebitLineItemForm attribute), 17
- account_code (accounting.journal_entry.forms.line_item.LineItemForm attribute), 18
- account_code (accounting.models.BaseAccountL10n attribute), 87
- account_code (accounting.models.JournalEntryLineItem property), 90
- account_code (accounting.option.forms.RecurringExpenseForm attribute), 33
- account_code (accounting.option.forms.RecurringIncomeForm attribute), 34
- account_code (accounting.option.forms.RecurringItemForm attribute), 34
- account_code (accounting.utils.options.RecurringItem attribute), 75
- account_codes (accounting.journal_entry.utils.description_editor.DescriptionRecurring property), 23
- account_codes (accounting.journal_entry.utils.description_editor.DescriptionTag property), 24
- account_id (accounting.models.AccountL10n attribute), 86
- account_id (accounting.models.JournalEntryLineItem attribute), 90
- account_options (accounting.report.reports.income_expenses.PageParams property), 44
- account_options (accounting.report.reports.ledger.PageParams property), 52
- account_options (accounting.report.reports.unapplied.PageParams property), 57
- account_options (accounting.report.reports.unapplied_accounts.CSVRow attribute), 58

<i>ing.report.reports.unapplied_accounts.PageParameters</i> property), 58	<i>accounting.currency</i> module, 9
<i>account_options</i> (<i>account- ing.report.reports.unmatched.PageParameters</i> property), 60	<i>accounting.currency.commands</i> module, 7
<i>account_options</i> (<i>account- ing.report.reports.unmatched_accounts.PageParameters</i> property), 62	<i>accounting.currency.converters</i> module, 7
ACCOUNT_REQUIRED (<i>in module accounting.forms</i>), 82	<i>accounting.currency.forms</i> module, 8
<i>account_text</i> (<i>account- ing.journal_entry.forms.line_item.LineItemForm</i> property), 18	<i>accounting.currency.queries</i> module, 9
<i>account_text</i> (<i>account- ing.option.forms.RecurringItemForm</i> property), 35	<i>accounting.currency.views</i> module, 9
<i>account_text</i> (<i>accounting.utils.options.RecurringItem</i> property), 75	<i>accounting.forms</i> module, 82
<i>account_title</i> (<i>account- ing.journal_entry.forms.line_item.LineItemForm</i> property), 18	<i>accounting.journal_entry</i> module, 32
<i>AccountCollector</i> (<i>class in account- ing.report.reports.balance_sheet</i>), 40	<i>accounting.journal_entry.converters</i> module, 30
<i>AccountConverter</i> (<i>class in account- ing.account.converters</i>), 3	<i>accounting.journal_entry.forms</i> module, 21
<i>AccountData</i> (<i>in module account- ing.account.commands</i>), 3	<i>accounting.journal_entry.forms.currency</i> module, 9
<i>AccountExists</i> (<i>class in accounting.forms</i>), 82	<i>accounting.journal_entry.forms.journal_entry</i> module, 13
<i>AccountForm</i> (<i>class in accounting.account.forms</i>), 4	<i>accounting.journal_entry.forms.line_item</i> module, 16
<i>accounting</i> module, 93	<i>accounting.journal_entry.forms.reorder</i> module, 20
<i>accounting.account</i> module, 6	<i>accounting.journal_entry.template_filters</i> module, 31
<i>accounting.account.commands</i> module, 3	<i>accounting.journal_entry.utils</i> module, 29
<i>accounting.account.converters</i> module, 3	<i>accounting.journal_entry.utils.account_option</i> module, 21
<i>accounting.account.forms</i> module, 4	<i>accounting.journal_entry.utils.description_editor</i> module, 22
<i>accounting.account.queries</i> module, 5	<i>accounting.journal_entry.utils.operators</i> module, 25
<i>accounting.account.views</i> module, 5	<i>accounting.journal_entry.utils.original_line_items</i> module, 29
<i>accounting.base_account</i> module, 7	<i>accounting.journal_entry.views</i> module, 32
<i>accounting.base_account.commands</i> module, 6	<i>accounting.locale</i> module, 82
<i>accounting.base_account.converters</i> module, 6	<i>accounting.models</i> module, 84
<i>accounting.base_account.queries</i> module, 7	<i>accounting.option</i> module, 35
<i>accounting.base_account.views</i> module, 7	<i>accounting.option.forms</i> module, 32
<i>accounting.commands</i> module, 82	<i>accounting.option.views</i> module, 35
	<i>accounting.report</i> module, 71

accounting.report.converters
 module, 69
 accounting.report.period
 module, 40
 accounting.report.period.chooser
 module, 36
 accounting.report.period.description
 module, 37
 accounting.report.period.month_end
 module, 37
 accounting.report.period.parser
 module, 37
 accounting.report.period.period
 module, 38
 accounting.report.period.shortcuts
 module, 39
 accounting.report.period.specification
 module, 40
 accounting.report.reports
 module, 63
 accounting.report.reports.balance_sheet
 module, 40
 accounting.report.reports.income_expenses
 module, 43
 accounting.report.reports.income_statement
 module, 46
 accounting.report.reports.journal
 module, 48
 accounting.report.reports.ledger
 module, 50
 accounting.report.reports.search
 module, 53
 accounting.report.reports.trial_balance
 module, 54
 accounting.report.reports.unapplied
 module, 56
 accounting.report.reports.unapplied_accounts
 module, 58
 accounting.report.reports.unmatched
 module, 59
 accounting.report.reports.unmatched_accounts
 module, 61
 accounting.report.template_filters
 module, 70
 accounting.report.utils
 module, 69
 accounting.report.utils.base_page_params
 module, 63
 accounting.report.utils.base_report
 module, 63
 accounting.report.utils.csv_export
 module, 64
 accounting.report.utils.offset_matcher
 module, 64
 accounting.report.utils.option_link
 module, 65
 accounting.report.utils.report_chooser
 module, 65
 accounting.report.utils.report_type
 module, 66
 accounting.report.utils.unapplied
 module, 67
 accounting.report.utils.unmatched
 module, 67
 accounting.report.utils.urls
 module, 67
 accounting.report.views
 module, 71
 accounting.template_filters
 module, 92
 accounting.template_globals
 module, 93
 accounting.utils
 module, 82
 accounting.utils.cast
 module, 71
 accounting.utils.current_account
 module, 71
 accounting.utils.flash_errors
 module, 72
 accounting.utils.journal_entry_types
 module, 72
 accounting.utils.next_uri
 module, 73
 accounting.utils.offset_alias
 module, 74
 accounting.utils.options
 module, 74
 accounting.utils.pagination
 module, 76
 accounting.utils.permission
 module, 77
 accounting.utils.query
 module, 78
 accounting.utils.random_id
 module, 78
 accounting.utils.strip_text
 module, 79
 accounting.utils.title_case
 module, 79
 accounting.utils.user
 module, 80
 AccountL10n (*class in accounting.models*), 86
 AccountNotCurrent (*class in accounting.option.forms*),
 32
 AccountOption (*class in account-*
ing.journal_entry.utils.account_option),
 21

AccountReorderForm (class in accounting.journal_entry.forms.line_item.CreditLineItemForm attribute), 4

accounts (accounting.journal_entry.utils.description_editor.DescriptionTag attribute), 22

accounts (accounting.journal_entry.utils.description_editor.DescriptionTag attribute), 24

accounts (accounting.models.BaseAccount attribute), amount (accounting.models.JournalEntryLineItem attribute), 86

accounts (accounting.report.reports.balance_sheet.Account attribute), amount (accounting.report.reports.balance_sheet.CSVHalfRow attribute), 40

accounts (accounting.report.reports.balance_sheet.Subsection attribute), amount (accounting.report.reports.balance_sheet.ReportAccount attribute), 43

accounts (accounting.report.reports.income_statement.Subsection attribute), amount (accounting.report.reports.income_statement.AccumulatedTotal attribute), 48

accounts (accounting.report.reports.trial_balance.PageParam attribute), amount (accounting.report.reports.income_statement.CSVRow attribute), 55

accounts (accounting.report.reports.unapplied_accounts.PageParam attribute), amount (accounting.report.reports.income_statement.ReportAccount attribute), 59

accounts (accounting.report.reports.unmatched_accounts.PageParam attribute), amount (accounting.report.reports.journal.ReportLineItem attribute), 62

accounts() (accounting.utils.current_account.CurrentAccount class method), amount (accounting.report.reports.unapplied.CSVRow attribute), 71

AccountsWithUnappliedOriginalLineItems (class in accounting.report.reports.unapplied_accounts), 58

AccountsWithUnmatchedOffsets (class in accounting.report.reports.unmatched_accounts), 61

accumulated (accounting.report.reports.income_statement.Section attribute), 48

accumulated_change() (accounting.models.Account class method), 84

ACCUMULATED_CHANGE_CODE (accounting.models.Account attribute), 84

AccumulatedTotal (class in accounting.report.reports.income_statement), 46

add_account() (accounting.journal_entry.utils.description_editor.DescriptionTag method), 24

add_freq() (accounting.journal_entry.utils.description_editor.DescriptionAccount method), 22

add_tag() (accounting.journal_entry.utils.description_editor.DescriptionDebitCredit method), 22

add_tag() (accounting.journal_entry.utils.description_editor.DescriptionType method), 24

all_errors (accounting.journal_entry.forms.line_item.LineItemForm attribute), 18

all_errors (accounting.option.forms.RecurringItemForm attribute), 35

all_url (accounting.report.period.chooser.PeriodChooser attribute), 36

AllTime (class in accounting.report.period.shortcuts), 39

amount (accounting.journal_entry.forms.line_item.CreditLineItemForm attribute), 16

amount (accounting.journal_entry.forms.line_item.DebitLineItemForm attribute), 17

amount (accounting.journal_entry.forms.line_item.LineItemForm attribute), 18

amount (accounting.models.JournalEntryLineItem attribute), 90

amount (accounting.report.reports.balance_sheet.CSVHalfRow attribute), 41

amount (accounting.report.reports.balance_sheet.ReportAccount attribute), 42

amount (accounting.report.reports.income_statement.AccumulatedTotal attribute), 46

amount (accounting.report.reports.income_statement.CSVRow attribute), 46

amount (accounting.report.reports.income_statement.ReportAccount attribute), 47

amount (accounting.report.reports.journal.ReportLineItem attribute), 50

amount (accounting.report.reports.unapplied.CSVRow attribute), 56

api_bp (in module accounting.currency.views), 9

append_next() (in module accounting.utils.next_uri), 73

ARTICLES (in module accounting.utils.title_case), 79

as_data (accounting.option.forms.RecurringForm attribute), 33

asset_amount (accounting.report.reports.balance_sheet.CSVRow attribute), 41

asset_title (accounting.report.reports.balance_sheet.CSVRow attribute), 41

assets (accounting.report.reports.balance_sheet.PageParams attribute), 41

available_years (accounting.report.period.chooser.PeriodChooser attribute), 36

Dr.DescriptionAccount

balance (accounting.models.JournalEntryLineItem attribute), 90

balance (accounting.report.reports.income_expenses.CSVRow attribute), 43

balance (accounting.report.reports.income_expenses.ReportLineItem attribute), 45

balance (accounting.report.reports.ledger.CSVRow attribute), 50

balance (accounting.report.reports.ledger.ReportLineItem attribute), 53

balance (accounting.report.reports.unmatched.CSVRow attribute), 59

BALANCE_SHEET (*accounting.report.utils.report_type.ReportType* attribute), 66
balance_sheet_url() (in module *accounting.report.utils.urls*), 67
BalanceSheet (class in *accounting.report.reports.balance_sheet*), 40
base (*accounting.account.forms.AccountReorderForm* attribute), 4
base (*accounting.models.Account* attribute), 84
base_code (*accounting.account.forms.AccountForm* attribute), 4
base_code (*accounting.models.Account* attribute), 84
base_options (*accounting.account.forms.AccountForm* property), 4
BaseAccount (class in *accounting.models*), 86
BaseAccountAvailable (class in *accounting.account.forms*), 5
BaseAccountConverter (class in *accounting.base_account.converters*), 6
BaseAccountExists (class in *accounting.account.forms*), 5
BaseAccountL10n (class in *accounting.models*), 87
BaseCSVRow (class in *accounting.report.utils.csv_export*), 64
BasePageParams (class in *accounting.report.utils.base_page_params*), 63
BaseReport (class in *accounting.report.utils.base_report*), 63
before (*accounting.report.period.period.Period* property), 38
bp (in module *accounting.account.views*), 5
bp (in module *accounting.base_account.views*), 7
bp (in module *accounting.currency.views*), 9
bp (in module *accounting.journal_entry.views*), 32
bp (in module *accounting.option.views*), 35
bp (in module *accounting.report.views*), 71
brought_forward (*accounting.report.reports.income_expenses.LineItemCollector* attribute), 44
brought_forward (*accounting.report.reports.income_expenses.PageParams* attribute), 45
brought_forward (*accounting.report.reports.ledger.LineItemCollector* attribute), 51
brought_forward (*accounting.report.reports.ledger.PageParams* attribute), 52
bus (*accounting.journal_entry.utils.description_editor.DescriptionEditor* attribute), 23
C
can_admin() (*accounting.utils.user.UserUtilityInterface* method), 80
can_admin() (in module *accounting.utils.permission*), 77
can_delete (*accounting.models.Account* property), 84
can_delete (*accounting.models.Currency* property), 87
can_delete (*accounting.models.JournalEntry* property), 88
can_edit() (*accounting.utils.user.UserUtilityInterface* method), 80
can_edit() (in module *accounting.utils.permission*), 77
can_view() (*accounting.utils.user.UserUtilityInterface* method), 80
can_view() (in module *accounting.utils.permission*), 77
CannotDeleteOriginalLineItemsWithOffset (class in *accounting.journal_entry.forms.journal_entry*), 13
cash() (*accounting.models.Account* class method), 84
CASH_CODE (*accounting.models.Account* attribute), 84
CASH_DISBURSEMENT (*accounting.utils.journal_entry_types.JournalEntryType* attribute), 72
CASH_RECEIPT (*accounting.utils.journal_entry_types.JournalEntryType* attribute), 72
CashDisbursementCurrencyForm (class in *accounting.journal_entry.forms.currency*), 9
CashDisbursementJournalEntry (class in *accounting.journal_entry.utils.operators*), 25
CashDisbursementJournalEntryForm (class in *accounting.journal_entry.forms.journal_entry*), 13
CashReceiptCurrencyForm (class in *accounting.journal_entry.forms.currency*), 10
CashReceiptJournalEntry (class in *accounting.journal_entry.utils.operators*), 26
CashReceiptJournalEntryForm (class in *accounting.journal_entry.forms.journal_entry*), 13
CHECK_ORDER (*accounting.journal_entry.utils.operators.CashDisbursementJournalEntry* attribute), 25
CHECK_ORDER (*accounting.journal_entry.utils.operators.CashReceiptJournalEntry* attribute), 26
CHECK_ORDER (*accounting.journal_entry.utils.operators.JournalEntryOperator* attribute), 27
CHECK_ORDER (*accounting.journal_entry.utils.operators.TransferJournalEntry* attribute), 28
clear_credit() (*accounting.utils.user.UserUtilityInterface* property), 80
code (*accounting.currency.forms.CurrencyForm* attribute), 8
code (*accounting.journal_entry.forms.currency.CashDisbursementCurrencyForm* attribute), 8

attribute), 10
 code (accounting.journal_entry.forms.currency.CashReceiptCurrencyForm attribute), 10
 code (accounting.journal_entry.forms.currency.CurrencyForm attribute), 11
 code (accounting.journal_entry.forms.currency.TransferCurrencyForm attribute), 11
 code (accounting.journal_entry.utils.account_option.AccountOption attribute), 21
 code (accounting.journal_entry.utils.description_editor.DescriptionEditor attribute), 22
 code (accounting.models.Account property), 84
 code (accounting.models.BaseAccount attribute), 86
 code (accounting.models.Currency attribute), 87
 code (accounting.models.JournalEntryCurrency attribute), 89
 code (accounting.utils.current_account.CurrentAccount attribute), 72
 code (accounting.utils.pagination.Redirection attribute), 77
 CODE_BLOCKLIST (accounting.currency.forms.CurrencyForm attribute), 8
 codes (accounting.utils.options.Recurring property), 75
 CodeUnique (class in accounting.currency.forms), 8
 collect() (accounting.journal_entry.forms.journal_entry.LineItemCollection method), 15
 collector (accounting.journal_entry.forms.journal_entry.LineItemCollection attribute), 14
 commit() (accounting.utils.options.Options method), 74
 CONJUNCTIONS (in module accounting.utils.title_case), 79
 count (accounting.models.Account property), 84
 count (accounting.report.reports.unapplied_accounts.CSVRow attribute), 58
 count (accounting.report.reports.unmatched_accounts.CSVRow attribute), 62
 created_at (accounting.models.Account attribute), 84
 created_at (accounting.models.Currency attribute), 87
 created_at (accounting.models.JournalEntry attribute), 88
 created_at (accounting.models.Option attribute), 92
 created_by (accounting.models.Account attribute), 84
 created_by (accounting.models.Currency attribute), 87
 created_by (accounting.models.JournalEntry attribute), 88
 created_by (accounting.models.Option attribute), 92
 created_by_id (accounting.models.Account attribute), 84
 created_by_id (accounting.models.Currency attribute), 87
 created_by_id (accounting.models.JournalEntry attribute), 89
 created_by_id (accounting.models.Option attribute), 92
 credit (accounting.journal_entry.forms.currency.CashReceiptCurrencyForm attribute), 10
 credit (accounting.journal_entry.forms.currency.TransferCurrencyForm attribute), 12
 credit (accounting.journal_entry.utils.description_editor.DescriptionEditor attribute), 23
 credit (accounting.models.JournalEntryCurrency attribute), 90
 credit (accounting.models.JournalEntryLineItem property), 90
 credit (accounting.report.reports.journal.CSVRow attribute), 48
 credit (accounting.report.reports.journal.ReportLineItem attribute), 50
 credit (accounting.report.reports.ledger.CSVRow attribute), 51
 credit (accounting.report.reports.ledger.ReportLineItem attribute), 53
 credit (accounting.report.reports.trial_balance.CSVRow attribute), 54
 credit (accounting.report.reports.trial_balance.ReportAccount attribute), 55
 credit (accounting.report.reports.trial_balance.Total attribute), 55
 credit (accounting.report.reports.unmatched.CSVRow attribute), 59
 credit_errors (accounting.journal_entry.forms.currency.CashReceiptCurrencyForm property), 10
 credit_errors (accounting.journal_entry.forms.currency.TransferCurrencyForm property), 12
 credit_total (accounting.journal_entry.forms.currency.CashReceiptCurrencyForm property), 10
 credit_total (accounting.journal_entry.forms.currency.TransferCurrencyForm property), 12
 credit_total (accounting.models.JournalEntryCurrency property), 90
 CreditLineItemForm (class in accounting.journal_entry.forms.line_item), 16
 csv() (accounting.report.reports.balance_sheet.BalanceSheet method), 41
 csv() (accounting.report.reports.income_expenses.IncomeExpenses method), 44
 csv() (accounting.report.reports.income_statement.IncomeStatement method), 46
 csv() (accounting.report.reports.journal.Journal

- method), 49
- csv() (accounting.report.reports.ledger.Ledger method), 51
- csv() (accounting.report.reports.search.Search method), 54
- csv() (accounting.report.reports.trial_balance.TrialBalance method), 56
- csv() (accounting.report.reports.unapplied.UnappliedOriginalLineItems method), 57
- csv() (accounting.report.reports.unapplied_accounts.AccountsWithUnappliedOriginalLineItems method), 58
- csv() (accounting.report.reports.unmatched.UnmatchedOffsets method), 61
- csv() (accounting.report.reports.unmatched_accounts.AccountsWithUnmatchedOffsets method), 61
- csv() (accounting.report.utils.base_report.BaseReport method), 63
- csv_download() (in module accounting.report.utils.csv_export), 64
- csv_uri (accounting.report.utils.base_page_params.BasePageParams attribute), 63
- CSVHalfRow (class in accounting.report.reports.balance_sheet), 41
- CSVRow (class in accounting.report.reports.balance_sheet), 41
- CSVRow (class in accounting.report.reports.income_expenses), 43
- CSVRow (class in accounting.report.reports.income_statement), 46
- CSVRow (class in accounting.report.reports.journal), 48
- CSVRow (class in accounting.report.reports.ledger), 50
- CSVRow (class in accounting.report.reports.trial_balance), 54
- CSVRow (class in accounting.report.reports.unapplied), 56
- CSVRow (class in accounting.report.reports.unapplied_accounts), 58
- CSVRow (class in accounting.report.reports.unmatched), 59
- CSVRow (class in accounting.report.reports.unmatched_accounts), 61
- currencies (accounting.journal_entry.forms.journal_entry.CashDisbursements.JournalEntryForm attribute), 13
- currencies (accounting.journal_entry.forms.journal_entry.CashReceipts.JournalEntryForm attribute), 13
- currencies (accounting.journal_entry.forms.journal_entry.CurrentEntryForms attribute), 14
- currencies (accounting.journal_entry.forms.journal_entry.TransferEntryForm attribute), 16
- currencies (accounting.models.JournalEntry property), 89
- currencies_errors (accounting.journal_entry.forms.journal_entry.JournalEntryForm property), 14
- currency (accounting.models.CurrencyL10n attribute), 88
- currency (accounting.models.JournalEntryLineItem attribute), 90
- currency (accounting.report.reports.balance_sheet.PageParams attribute), 42
- currency (accounting.report.reports.income_expenses.PageParams attribute), 47
- currency (accounting.report.reports.income_statement.PageParams attribute), 47
- currency (accounting.report.reports.journal.CSVRow attribute), 50
- currency (accounting.report.reports.journal.ReportLineItem attribute), 50
- currency (accounting.report.reports.ledger.PageParams attribute), 52
- currency (accounting.report.reports.trial_balance.PageParams attribute), 55
- currency (accounting.report.reports.unapplied.CSVRow attribute), 56
- currency (accounting.report.reports.unapplied.PageParams attribute), 57
- currency (accounting.report.reports.unapplied_accounts.PageParams attribute), 59
- currency (accounting.report.reports.unmatched.CSVRow attribute), 59
- currency (accounting.report.reports.unmatched.PageParams attribute), 60
- currency (accounting.report.reports.unmatched_accounts.PageParams attribute), 62
- Currency (class in accounting.models), 87
- currency_code (accounting.models.CurrencyL10n attribute), 88
- currency_code (accounting.models.JournalEntryLineItem attribute), 91
- currency_options (accounting.report.reports.balance_sheet.PageParams property), 42
- currency_options (accounting.report.reports.income_expenses.PageParams property), 47
- currency_options (accounting.report.reports.ledger.PageParams property), 52
- currency_options (accounting.report.reports.trial_balance.PageParams property), 55
- currency_options (accounting.report.reports.unapplied_accounts.PageParams property), 59
- currency_options (accounting.report.reports.unmatched_accounts.PageParams property), 62

ing.report.reports.unapplied.PageParams (property), 57
currency_options (*accounting.report.reports.unapplied_accounts.PageParams* property), 59
currency_options (*accounting.report.reports.unmatched.PageParams* property), 60
currency_options (*accounting.report.reports.unmatched_accounts.PageParams* property), 62
currency_options() (in module *accounting.template_globals*), 93
CURRENCY_REQUIRED (in module *accounting.journal_entry.forms.currency*), 9
CurrencyConverter (class in *accounting.currency.converters*), 7
CurrencyExists (class in *accounting.forms*), 82
CurrencyForm (class in *accounting.currency.forms*), 8
CurrencyForm (class in *accounting.journal_entry.forms.currency*), 11
CurrencyL10n (class in *accounting.models*), 88
current_accounts (*accounting.option.forms.OptionForm* property), 32
CURRENT_AL_CODE (*accounting.utils.current_account.CurrentAccount* attribute), 71
current_assets_and_liabilities() (*accounting.utils.current_account.CurrentAccount* class method), 72
current_report (*accounting.report.utils.report_chooser.ReportChooser* attribute), 66
current_user (*accounting.utils.user.UserUtilityInterface* property), 80
CurrentAccount (class in *accounting.utils.current_account*), 71
CurrentAccountConverter (class in *accounting.report.converters*), 69
CurrentAccountExists (class in *accounting.option.forms*), 32

D

data_dir (in module *accounting*), 93
data_start (*accounting.report.period.chooser.PeriodChooser* attribute), 36
date (*accounting.journal_entry.forms.journal_entry.CashDisbursementJournalEntryForm* attribute), 13
date (*accounting.journal_entry.forms.journal_entry.CashReceiptJournalEntryForm* attribute), 13
date (*accounting.journal_entry.forms.journal_entry.JournalEntryForm* attribute), 14
date (*accounting.journal_entry.forms.journal_entry.TransferJournalEntryForm* attribute), 16
date (*accounting.journal_entry.forms.reorder.JournalEntryReorderForm* attribute), 20
date (*accounting.models.JournalEntry* attribute), 89
date (*accounting.report.reports.income_expenses.CSVRow* attribute), 43
date (*accounting.report.reports.income_expenses.ReportLineItem* attribute), 45
date (*accounting.report.reports.journal.CSVRow* attribute), 49
date (*accounting.report.reports.ledger.CSVRow* attribute), 51
date (*accounting.report.reports.ledger.ReportLineItem* attribute), 53
date (*accounting.report.reports.unapplied.CSVRow* attribute), 56
date (*accounting.report.reports.unmatched.CSVRow* attribute), 60
DATE_REQUIRED (in module *accounting.journal_entry.forms.journal_entry*), 14
DATE_SPEC_RE (in module *accounting.report.period.parser*), 37
DateConverter (class in *accounting.journal_entry.converters*), 30
db (in module *accounting*), 93
debit (*accounting.journal_entry.forms.currency.CashDisbursementCurrentAccountForm* attribute), 10
debit (*accounting.journal_entry.forms.currency.TransferCurrencyForm* attribute), 12
debit (*accounting.journal_entry.utils.description_editor.DescriptionEditor* attribute), 23
debit (*accounting.models.JournalEntryCurrency* attribute), 90
debit (*accounting.models.JournalEntryLineItem* property), 91
debit (*accounting.report.reports.journal.CSVRow* attribute), 49
debit (*accounting.report.reports.journal.ReportLineItem* attribute), 50
debit (*accounting.report.reports.ledger.CSVRow* attribute), 51
debit (*accounting.report.reports.ledger.ReportLineItem* attribute), 53
debit (*accounting.report.reports.trial_balance.CSVRow* attribute), 54
debit (*accounting.report.reports.trial_balance.ReportAccount* attribute), 55
debit (*accounting.report.reports.trial_balance.TotalJournalEntryForm* attribute), 56
debit (*accounting.report.reports.unmatched.CSVRow* attribute), 60
debit_account_options (*accounting.journal_entry.forms.journal_entry.JournalEntryForm* attribute), 14

23
 DescriptionRecurring (class in *accounting.journal_entry.utils.description_editor*),
 23
 DescriptionTag (class in *accounting.journal_entry.utils.description_editor*),
 23
 DescriptionType (class in *accounting.journal_entry.utils.description_editor*),
 24
 domain (in module *accounting.locale*), 82

E

EmptyPagination (class in *accounting.utils.pagination*), 76
 encode_next() (in module *accounting.utils.next_uri*),
 73
 end (*accounting.report.period.period.Period* attribute),
 38
 expense (*accounting.report.reports.income_expenses.CSVRow* attribute), 43
 expense (*accounting.report.reports.income_expenses.Report* attribute), 46
 expense_accounts (*accounting.option.forms.RecurringForm* property),
 34
 expenses (*accounting.option.forms.RecurringForm* attribute), 34
 expenses (*accounting.utils.options.Recurring* attribute),
 75

F

fa_icon (*accounting.report.utils.option_link.OptionLink* attribute), 65
 find_by_code() (*accounting.models.Account* class method), 85
 flash_form_errors() (in module *accounting.utils.flash_errors*), 72
 form (*accounting.journal_entry.forms.journal_entry.LineItemCollection* attribute), 15
 form (*accounting.journal_entry.utils.operators.CashDisbursementJournalEntry* property), 25
 form (*accounting.journal_entry.utils.operators.CashReceiptJournalEntry* property), 26
 form (*accounting.journal_entry.utils.operators.JournalEntryOperator* property), 27
 form (*accounting.journal_entry.utils.operators.TransferJournalEntry* property), 28
 format_amount() (in module *accounting.report.template_filters*), 70
 format_amount() (in module *accounting.template_filters*), 92
 format_amount_input() (in module *accounting.journal_entry.template_filters*), 31

format_date() (in module *accounting.template_filters*),
 93
 freq (*accounting.journal_entry.utils.description_editor.DescriptionAccount* attribute), 22
 freq (*accounting.journal_entry.utils.description_editor.DescriptionTag* attribute), 24

G

general (*accounting.journal_entry.utils.description_editor.DescriptionDel* attribute), 23
 get_account_query() (in module *accounting.account.queries*), 5
 get_accounts_with_unapplied() (in module *accounting.report.utils.unapplied*), 67
 get_accounts_with_unmatched() (in module *accounting.report.utils.unmatched*), 67
 get_base_account_query() (in module *accounting.base_account.queries*), 7
 get_by_username() (*accounting.utils.user.UserUtilityInterface* method),
 80
 get_csv_rows() (in module *accounting.report.reports.journal*), 50
 get_csv_rows() (in module *accounting.report.reports.unapplied*), 57
 get_csv_rows() (in module *accounting.report.reports.unapplied_accounts*),
 59
 get_csv_rows() (in module *accounting.report.reports.unmatched*), 61
 get_csv_rows() (in module *accounting.report.reports.unmatched_accounts*),
 62
 get_currency_query() (in module *accounting.currency.queries*), 9
 get_current_user() (in module *accounting.utils.user*), 81
 get_current_user_pk() (in module *accounting.utils.user*), 81
 get_desc() (in module *accounting.report.period.description*), 37
 get_journal_entry_op() (in module *accounting.journal_entry.utils.operators*), 29
 get_net_balances() (in module *accounting.report.utils.unapplied*), 67
 get_period() (in module *accounting.report.period.parser*), 37
 get_pk() (*accounting.utils.user.UserUtilityInterface* method), 80
 get_position() (in module *accounting.journal_entry.utils.description_editor*),
 25
 get_prefix() (in module *accounting.journal_entry.utils.description_editor*),

- 25
 get_selectable_original_line_items() (in module accounting.journal_entry.utils.original_line_items), 29
- get_spec() (in module accounting.report.period.specification), 40
- get_user_pk() (in module accounting.utils.user), 81
- gettext() (in module accounting.locale), 82
- ## H
- has_data(accounting.report.period.chooser.PeriodChooser attribute), 36
- has_data(accounting.report.reports.balance_sheet.PageParams property), 42
- has_data(accounting.report.reports.income_expenses.PageParams property), 45
- has_data(accounting.report.reports.income_statement.PageParams property), 47
- has_data(accounting.report.reports.journal.PageParams property), 49
- has_data(accounting.report.reports.ledger.PageParams property), 52
- has_data(accounting.report.reports.search.PageParams property), 53
- has_data(accounting.report.reports.trial_balance.PageParams property), 55
- has_data(accounting.report.reports.unapplied.PageParams property), 57
- has_data(accounting.report.reports.unapplied_accounts.PageParams property), 59
- has_data(accounting.report.reports.unmatched.PageParams property), 60
- has_data(accounting.report.reports.unmatched_accounts.PageParams property), 62
- has_data(accounting.report.utils.base_page_params.BasePageParams property), 63
- has_last_month (accounting.report.period.chooser.PeriodChooser attribute), 36
- has_last_year (accounting.report.period.chooser.PeriodChooser attribute), 36
- has_permission() (in module accounting.utils.permission), 77
- has_user() (in module accounting.utils.user), 81
- has_yesterday (accounting.report.period.chooser.PeriodChooser attribute), 36
- html() (accounting.report.reports.balance_sheet.BalanceSheet method), 41
- html() (accounting.report.reports.income_expenses.IncomeExpenses method), 44
- html() (accounting.report.reports.income_statement.IncomeStatement method), 47
- html() (accounting.report.reports.journal.Journal method), 49
- html() (accounting.report.reports.ledger.Ledger method), 51
- html() (accounting.report.reports.search.Search method), 54
- html() (accounting.report.reports.trial_balance.TrialBalance method), 56
- html() (accounting.report.reports.unapplied.UnappliedOriginalLineItems method), 57
- html() (accounting.report.reports.unapplied_accounts.AccountsWithUnapplied method), 58
- html() (accounting.report.reports.unmatched.UnmatchedOffsets method), 61
- html() (accounting.report.reports.unmatched_accounts.AccountsWithUnmatched method), 61
- html() (accounting.report.utils.base_report.BaseReport method), 63
- ## I
- id(accounting.journal_entry.forms.line_item.CreditLineItemForm attribute), 17
- id(accounting.journal_entry.forms.line_item.DebitLineItemForm attribute), 17
- id(accounting.journal_entry.forms.line_item.LineItemForm attribute), 18
- id(accounting.journal_entry.utils.account_option.AccountOption attribute), 21
- id(accounting.journal_entry.utils.description_editor.DescriptionAccount attribute), 22
- id(accounting.journal_entry.utils.description_editor.DescriptionType attribute), 24
- id(accounting.models.Account attribute), 85
- id(accounting.models.JournalEntry attribute), 89
- id(accounting.models.JournalEntryLineItem attribute), 91
- id(accounting.utils.current_account.CurrentAccount attribute), 72
- income(accounting.report.reports.income_expenses.CSVRow attribute), 43
- income(accounting.report.reports.income_expenses.ReportLineItem attribute), 46
- income_accounts (accounting.option.forms.RecurringForm property), 34
- INCOME_EXPENSES (accounting.report.utils.report_type.ReportType attribute), 66
- income_expenses_url() (in module accounting.report.utils.urls), 67
- INCOME_STATEMENT (accounting.report.utils.report_type.ReportType attribute), 67

- tribute), 66
- income_statement_url() (in module *accounting.report.utils.urls*), 68
- IncomeExpenses (class in *accounting.report.reports.income_expenses*), 44
- incomes (accounting.option.forms.RecurringForm attribute), 34
- incomes (accounting.utils.options.Recurring attribute), 75
- IncomeStatement (class in *accounting.report.reports.income_statement*), 46
- inherit_next() (in module *accounting.utils.next_uri*), 73
- init_accounts_command() (in module *accounting.account.commands*), 3
- init_app() (in module *accounting*), 93
- init_app() (in module *accounting.account*), 6
- init_app() (in module *accounting.base_account*), 7
- init_app() (in module *accounting.currency*), 9
- init_app() (in module *accounting.journal_entry*), 32
- init_app() (in module *accounting.locale*), 83
- init_app() (in module *accounting.option*), 35
- init_app() (in module *accounting.report*), 71
- init_app() (in module *accounting.utils.next_uri*), 73
- init_app() (in module *accounting.utils.permission*), 78
- init_base_accounts_command() (in module *accounting.base_account.commands*), 6
- init_currencies_command() (in module *accounting.currency.commands*), 7
- init_user_utils() (in module *accounting.utils.user*), 81
- is_a_day (accounting.report.period.period.Period attribute), 38
- is_a_month (accounting.report.period.period.Period attribute), 38
- is_a_year (accounting.report.period.period.Period attribute), 38
- is_active (accounting.report.utils.option_link.OptionLink attribute), 65
- is_all (accounting.report.period.period.Period attribute), 38
- is_brought_forward (accounting.report.reports.income_expenses.ReportLineItem attribute), 46
- is_brought_forward (accounting.report.reports.ledger.ReportLineItem attribute), 53
- is_cash_disbursement (accounting.models.JournalEntry property), 89
- is_cash_receipt (accounting.models.JournalEntry property), 89
- is_code_locked (accounting.journal_entry.forms.currency.CurrencyForm property), 11
- is_current (accounting.utils.pagination.Link attribute), 76
- is_debit (accounting.models.JournalEntryLineItem attribute), 91
- is_default (accounting.report.period.period.Period attribute), 38
- is_for_mobile (accounting.utils.pagination.Link attribute), 76
- is_in_use (accounting.journal_entry.utils.account_option.AccountOption attribute), 21
- is_last_month (accounting.report.period.period.Period attribute), 38
- is_last_year (accounting.report.period.period.Period attribute), 38
- is_modified (accounting.account.forms.AccountReorderForm attribute), 5
- is_modified (accounting.journal_entry.forms.journal_entry.JournalEntryForm attribute), 14
- is_modified (accounting.journal_entry.forms.reorder.JournalEntryReorderForm attribute), 20
- is_modified (accounting.models.Account property), 85
- is_modified (accounting.models.Currency property), 87
- is_modified (accounting.utils.options.Options attribute), 74
- is_my_type() (accounting.journal_entry.utils.operators.CashDisbursementJournalEntry method), 25
- is_my_type() (accounting.journal_entry.utils.operators.CashReceiptJournalEntry method), 26
- is_my_type() (accounting.journal_entry.utils.operators.JournalEntryOperator method), 27
- is_my_type() (accounting.journal_entry.utils.operators.TransferJournalEntry method), 28
- is_need_offset (accounting.account.forms.AccountForm attribute), 4
- is_need_offset (accounting.journal_entry.forms.line_item.LineItemForm property), 19
- is_need_offset (accounting.journal_entry.utils.account_option.AccountOption attribute), 21
- is_need_offset (accounting.journal_entry.utils.description_editor.DescriptionAccount attribute), 22
- is_need_offset (accounting.models.Account attribute), 22

- tribute), 85
- is_need_offset (accounting.models.JournalEntryLineItem property), 91
- is_nominal (accounting.models.Account property), 85
- is_offset (accounting.models.JournalEntryLineItem property), 91
- is_paged (accounting.utils.pagination.AbstractPagination attribute), 76
- is_paged (accounting.utils.pagination.Pagination attribute), 77
- is_real (accounting.models.Account property), 85
- is_search (accounting.report.utils.report_chooser.ReportChooser attribute), 66
- is_since_last_month (accounting.report.period.period.Period attribute), 38
- is_this_month (accounting.report.period.period.Period attribute), 38
- is_this_year (accounting.report.period.period.Period attribute), 38
- is_today (accounting.report.period.period.Period attribute), 38
- is_total (accounting.report.reports.income_expenses.ReportLineItem attribute), 46
- is_total (accounting.report.reports.ledger.ReportLineItem attribute), 53
- is_type_arbitrary (accounting.report.period.period.Period property), 38
- is_type_month (accounting.report.period.period.Period attribute), 39
- is_year() (accounting.report.period.period.Period method), 39
- is_yesterday (accounting.report.period.period.Period attribute), 39
- IsBalanced (class in accounting.journal_entry.forms.currency), 11
- IsCreditAccount (class in accounting.forms), 82
- IsDebitAccount (class in accounting.forms), 82
- item_template (accounting.option.forms.RecurringForm property), 34
- ## J
- JOURNAL (accounting.report.utils.report_type.ReportType attribute), 66
- Journal (class in accounting.report.reports.journal), 49
- journal_entry (accounting.models.JournalEntryLineItem attribute), 91
- journal_entry (accounting.report.reports.journal.ReportLineItem attribute), 50
- journal_entry_form (accounting.journal_entry.forms.line_item.LineItemForm attribute), 19
- journal_entry_id (accounting.models.JournalEntryLineItem attribute), 91
- JOURNAL_ENTRY_TYPE_TO_OP (in module accounting.journal_entry.utils.operators), 27
- journal_entry_types (accounting.report.utils.base_page_params.BasePageParams property), 63
- journal_url() (in module accounting.report.utils.urls), 68
- JournalEntry (class in accounting.models), 88
- JournalEntryConverter (class in accounting.journal_entry.converters), 30
- JournalEntryCurrency (class in accounting.models), 89
- JournalEntryForm (class in accounting.journal_entry.forms.journal_entry), 14
- JournalEntryLineItem (class in accounting.models), 91
- JournalEntryOperator (class in accounting.journal_entry.utils.operators), 27
- JournalEntryReorderForm (class in accounting.journal_entry.forms.reorder), 20
- JournalEntryType (class in accounting.utils.journal_entry_types), 72
- JournalEntryTypeConverter (class in accounting.journal_entry.converters), 30
- ## K
- KeepAccountWhenHavingOffset (class in accounting.journal_entry.forms.line_item), 18
- KeepCurrencyWhenHavingOffset (class in accounting.journal_entry.forms.currency), 11
- ## L
- 110n (accounting.models.Account attribute), 85
- 110n (accounting.models.BaseAccount attribute), 86
- 110n (accounting.models.Currency attribute), 88
- last_month_url (accounting.report.period.chooser.PeriodChooser attribute), 36
- last_year_url (accounting.report.period.chooser.PeriodChooser attribute), 36
- LastMonth (class in accounting.report.period.shortcuts), 39
- LastYear (class in accounting.report.period.shortcuts), 39

- lazy_gettext() (in module *accounting.locale*), 83
 LEDGER (*accounting.report.utils.report_type.ReportType* attribute), 66
 Ledger (class in *accounting.report.reports.ledger*), 51
 ledger_url() (in module *accounting.report.utils.urls*), 68
 liabilities (*accounting.report.reports.balance_sheet.PageParams* attribute), 42
 liability_amount (*accounting.report.reports.balance_sheet.CSVRow* attribute), 41
 liability_title (*accounting.report.reports.balance_sheet.CSVRow* attribute), 41
 line_item (*accounting.report.reports.journal.ReportLineItem* attribute), 50
 line_items (*accounting.journal_entry.forms.currency.CurrencyForm* property), 11
 line_items (*accounting.journal_entry.forms.journal_entry.JournalEntryForm* property), 14
 line_items (*accounting.models.Account* attribute), 85
 line_items (*accounting.models.Currency* attribute), 88
 line_items (*accounting.models.JournalEntry* attribute), 89
 line_items (*accounting.report.reports.income_expenses.LineItemCollector* attribute), 44
 line_items (*accounting.report.reports.income_expenses.PageParams* attribute), 45
 line_items (*accounting.report.reports.journal.PageParams* attribute), 49
 line_items (*accounting.report.reports.ledger.LineItemCollector* attribute), 51
 line_items (*accounting.report.reports.ledger.PageParams* attribute), 52
 line_items (*accounting.report.reports.search.LineItemCollector* attribute), 53
 line_items (*accounting.report.reports.search.PageParams* attribute), 53
 line_items (*accounting.report.reports.unapplied.PageParams* attribute), 57
 line_items (*accounting.report.reports.unmatched.PageParams* attribute), 60
 line_items (*accounting.report.utils.offset_matcher.OffsetMatcher* attribute), 64
 LineItemCollector (class in *accounting.journal_entry.forms.journal_entry*), 15
 LineItemCollector (class in *accounting.report.reports.income_expenses*), 44
 LineItemCollector (class in *accounting.report.reports.ledger*), 51
 LineItemCollector (class in *accounting.report.reports.search*), 53
 LineItemForm (class in *accounting.journal_entry.forms.journal_entry*), 18
 Link (class in *accounting.utils.pagination*), 76
 list (*accounting.utils.pagination.AbstractPagination* attribute), 76
 list (*accounting.utils.pagination.Pagination* attribute), 77
 locale (*accounting.models.AccountL10n* attribute), 86
 locale (*accounting.models.BaseAccountL10n* attribute), 87
 locale (*accounting.models.CurrencyL10n* attribute), 88
M
 match (*accounting.models.JournalEntryLineItem* property), 91
 match() (*accounting.report.utils.offset_matcher.OffsetMatcher* method), 64
 match_status (*accounting.report.reports.unmatched.PageParams* attribute), 60
 matched_pairs (*accounting.report.reports.unmatched.PageParams* attribute), 60
 matched_pairs (*accounting.report.utils.offset_matcher.OffsetMatcher* attribute), 64
 max_date (*accounting.journal_entry.forms.journal_entry.JournalEntryForm* property), 14
 min_date (*accounting.journal_entry.forms.journal_entry.JournalEntryForm* property), 15
 MINOR_WORDS (in module *accounting.utils.title_case*), 79
 module
 accounting, 93
 accounting.account, 6
 accounting.account.commands, 3
 accounting.account.converters, 3
 accounting.account.forms, 4
 accounting.account.queries, 5
 accounting.account.views, 5
 accounting.base_account, 7
 accounting.base_account.commands, 6
 accounting.base_account.converters, 6
 accounting.base_account.queries, 7
 accounting.base_account.views, 7
 accounting.commands, 82
 accounting.currency, 9
 accounting.currency.commands, 7
 accounting.currency.converters, 7
 accounting.currency.forms, 8
 accounting.currency.queries, 9
 accounting.currency.views, 9
 accounting.forms, 82
 accounting.journal_entry, 32
 accounting.journal_entry.converters, 30
 accounting.journal_entry.forms, 21

- accounting.journal_entry.forms.currency, 9
 accounting.journal_entry.forms.journal_entry, 13
 accounting.journal_entry.forms.line_item, 16
 accounting.journal_entry.forms.reorder, 20
 accounting.journal_entry.template_filters, 31
 accounting.journal_entry.utils, 29
 accounting.journal_entry.utils.account_option, 21
 accounting.journal_entry.utils.description_editor, 22
 accounting.journal_entry.utils.operators, 25
 accounting.journal_entry.utils.original_line_items, 29
 accounting.journal_entry.views, 32
 accounting.locale, 82
 accounting.models, 84
 accounting.option, 35
 accounting.option.forms, 32
 accounting.option.views, 35
 accounting.report, 71
 accounting.report.converters, 69
 accounting.report.period, 40
 accounting.report.period.chooser, 36
 accounting.report.period.description, 37
 accounting.report.period.month_end, 37
 accounting.report.period.parser, 37
 accounting.report.period.period, 38
 accounting.report.period.shortcuts, 39
 accounting.report.period.specification, 40
 accounting.report.reports, 63
 accounting.report.reports.balance_sheet, 40
 accounting.report.reports.income_expenses, 43
 accounting.report.reports.income_statement, 46
 accounting.report.reports.journal, 48
 accounting.report.reports.ledger, 50
 accounting.report.reports.search, 53
 accounting.report.reports.trial_balance, 54
 accounting.report.reports.unapplied, 56
 accounting.report.reports.unapplied_accounts, 58
 accounting.report.reports.unmatched, 59
 accounting.report.reports.unmatched_accounts, 61
 accounting.report.template_filters, 70
 accounting.report.utils, 69
 accounting.report.utils.base_page_params, 63
 accounting.report.utils.base_report, 63
 accounting.report.utils.csv_export, 64
 accounting.report.utils.offset_matcher, 64
 accounting.report.utils.option_link, 65
 accounting.report.utils.report_chooser, 65
 accounting.report.utils.report_type, 66
 accounting.report.utils.unapplied, 67
 accounting.report.utils.unmatched, 67
 accounting.report.utils.urls, 67
 accounting.report.views, 71
 accounting.template_filters, 92
 accounting.template_globals, 93
 accounting.utils, 82
 accounting.utils.cast, 71
 accounting.utils.current_account, 71
 accounting.utils.flash_errors, 72
 accounting.utils.journal_entry_types, 72
 accounting.utils.next_uri, 73
 accounting.utils.offset_alias, 74
 accounting.utils.options, 74
 accounting.utils.pagination, 76
 accounting.utils.permission, 77
 accounting.utils.query, 78
 accounting.utils.random_id, 78
 accounting.utils.strip_text, 79
 accounting.utils.title_case, 79
 accounting.utils.user, 80
 month_end() (in module *accounting.report.period.month_end*), 37
- ## N
- name (*accounting.currency.forms.CurrencyForm* attribute), 8
 name (*accounting.journal_entry.utils.description_editor.DescriptionRecurring* attribute), 23
 name (*accounting.journal_entry.utils.description_editor.DescriptionTag* attribute), 24
 name (*accounting.models.Currency* property), 88
 name (*accounting.models.CurrencyL10n* attribute), 88
 name (*accounting.models.JournalEntryCurrency* property), 90
 name (*accounting.models.Option* attribute), 92
 name (*accounting.option.forms.RecurringExpenseForm* attribute), 33
 name (*accounting.option.forms.RecurringIncomeForm* attribute), 34
 name (*accounting.option.forms.RecurringItemForm* attribute), 35

name (*accounting.utils.options.RecurringItem* attribute), 75
 name_line (*accounting.models.Currency* attribute), 88
 NeedOffsetAccountConverter (class in *accounting.report.converters*), 69
 NeedSomeCurrencies (class in *accounting.journal_entry.forms.journal_entry*), 15
 NeedSomeLineItems (class in *accounting.journal_entry.forms.currency*), 11
 net_balance (*accounting.journal_entry.forms.line_item.LineItemForm* property), 19
 net_balance (*accounting.models.JournalEntryLineItem* property), 91
 net_balance (*accounting.report.reports.unapplied.CSVRow* attribute), 56
 NET_CHANGE_CODE (*accounting.models.Account* attribute), 84
 new_id() (in module *accounting.utils.random_id*), 78
 no (*accounting.journal_entry.forms.currency.CashDisbursementForm* attribute), 10
 no (*accounting.journal_entry.forms.currency.CashReceiptForm* attribute), 11
 no (*accounting.journal_entry.forms.currency.CurrencyForm* attribute), 11
 no (*accounting.journal_entry.forms.currency.TransferCurrencyForm* attribute), 12
 no (*accounting.journal_entry.forms.line_item.CreditLineItemForm* attribute), 17
 no (*accounting.journal_entry.forms.line_item.DebitLineItemForm* attribute), 18
 no (*accounting.journal_entry.forms.line_item.LineItemForm* attribute), 19
 no (*accounting.models.Account* attribute), 85
 no (*accounting.models.JournalEntry* attribute), 89
 no (*accounting.models.JournalEntryLineItem* attribute), 91
 no (*accounting.option.forms.RecurringExpenseForm* attribute), 33
 no (*accounting.option.forms.RecurringIncomeForm* attribute), 34
 no (*accounting.option.forms.RecurringItemForm* attribute), 35
 NonEmptyPagination (class in *accounting.utils.pagination*), 76
 NoOffsetNominalAccount (class in *accounting.report.forms*), 5
 NotAfterOffsetItems (class in *accounting.journal_entry.forms.journal_entry*), 15
 NotBeforeOriginalLineItems (class in *accounting.journal_entry.forms.journal_entry*), 15
 note (*accounting.journal_entry.forms.journal_entry.CashDisbursementJournalEntryForm* attribute), 13
 note (*accounting.journal_entry.forms.journal_entry.CashReceiptJournalEntryForm* attribute), 13
 note (*accounting.journal_entry.forms.journal_entry.JournalEntryForm* attribute), 15
 note (*accounting.journal_entry.forms.journal_entry.TransferJournalEntryForm* attribute), 16
 note (*accounting.models.JournalEntry* attribute), 89
 note (*accounting.report.reports.income_expenses.CSVRow* attribute), 43
 note (*accounting.report.reports.income_expenses.ReportLineItem* attribute), 46
 note (*accounting.report.reports.journal.CSVRow* attribute), 49
 note (*accounting.report.reports.ledger.CSVRow* attribute), 51
 note (*accounting.report.reports.ledger.ReportLineItem* attribute), 53
 NotExceedingOriginalLineItemNetBalance (class in *accounting.journal_entry.forms.line_item*), 19
 NotLessThanOffsetTotal (class in *accounting.journal_entry.forms.line_item*), 19
 NotStartPayableFromDebit (class in *accounting.journal_entry.forms.line_item*), 19
 NotStartPayableFromExpense (class in *accounting.option.forms*), 32
 NotStartReceivableFromCredit (class in *accounting.journal_entry.forms.line_item*), 20
 NotStartReceivableFromIncome (class in *accounting.option.forms*), 32
 O
 obj (*accounting.journal_entry.forms.journal_entry.JournalEntryForm* attribute), 15
 obj_code (*accounting.currency.forms.CurrencyForm* attribute), 8
 offset (*accounting.report.utils.offset_matcher.OffsetPair* attribute), 65
 offset_alias() (in module *accounting.utils.offset_alias*), 74
 offset_total (*accounting.journal_entry.forms.line_item.LineItemForm* property), 19
 OffsetMatcher (class in *accounting.report.utils.offset_matcher*), 64
 OffsetPair (class in *accounting.report.utils.offset_matcher*), 65
 offsets (*accounting.journal_entry.forms.line_item.LineItemForm* property), 19
 offsets (*accounting.models.JournalEntryLineItem* property), 91
 Option (class in *accounting.models*), 92
 OptionForm (class in *accounting.option.forms*), 32

OptionLink (class in accounting.tribute), 76
 (ing.report.utils.option_link), 65
 Options (class in accounting.utils.options), 74
 options (in module accounting.utils.options), 75
 or_next() (in module accounting.utils.next_uri), 73
 original_line_item (accounting.models.JournalEntryLineItem attribute), 91
 original_line_item (accounting.report.utils.offset_matcher.OffsetPair attribute), 65
 original_line_item_date (accounting.journal_entry.forms.line_item.LineItemForm property), 19
 original_line_item_id (accounting.journal_entry.forms.line_item.CreditLineItemForm attribute), 17
 original_line_item_id (accounting.journal_entry.forms.line_item.DebitLineItemForm attribute), 18
 original_line_item_id (accounting.journal_entry.forms.line_item.LineItemForm attribute), 19
 original_line_item_id (accounting.models.JournalEntryLineItem attribute), 92
 original_line_item_options (accounting.journal_entry.forms.journal_entry.JournalEntryForm property), 15
 original_line_item_text (accounting.journal_entry.forms.line_item.LineItemForm property), 19
 OriginalLineItemExists (class in accounting.journal_entry.forms.line_item), 20
 OriginalLineItemNeedOffset (class in accounting.journal_entry.forms.line_item), 20
 OriginalLineItemNotOffset (class in accounting.journal_entry.forms.line_item), 20
 OriginalLineItemOppositeDebitCredit (class in accounting.journal_entry.forms.line_item), 20
 owner_s_equity (accounting.report.reports.balance_sheet.PageParams attribute), 42
P
 page_size (accounting.utils.pagination.AbstractPagination attribute), 76
 page_size (accounting.utils.pagination.Pagination attribute), 77
 PAGE_SIZE_OPTION_VALUES (accounting.utils.pagination.NonEmptyPagination attribute), 76
 page_size_options (accounting.utils.pagination.AbstractPagination attribute), 76
 page_size_options (accounting.utils.pagination.Pagination attribute), 77
 PageParams (class in accounting.tribute), 76
 PageParams (class in accounting.ing.report.reports.balance_sheet), 41
 PageParams (class in accounting.ing.report.reports.income_expenses), 44
 PageParams (class in accounting.ing.report.reports.income_statement), 47
 PageParams (class in accounting.report.reports.journal), 49
 PageParams (class in accounting.report.reports.ledger), 51
 PageParams (class in accounting.report.reports.search), 53
 PageParams (class in accounting.ing.report.reports.trial_balance), 54
 PageParams (class in accounting.ing.report.reports.unapplied), 56
 PageParams (class in accounting.ing.report.reports.unapplied_accounts), 58
 PageParams (class in accounting.ing.report.reports.unmatched), 60
 PageParams (class in accounting.ing.report.reports.unmatched_accounts), 62
 pages (accounting.utils.pagination.AbstractPagination attribute), 76
 pages (accounting.utils.pagination.Pagination attribute), 77
 pagination (accounting.report.reports.income_expenses.PageParams attribute), 45
 pagination (accounting.report.reports.journal.PageParams attribute), 49
 pagination (accounting.report.reports.ledger.PageParams attribute), 52
 pagination (accounting.report.reports.search.PageParams attribute), 54
 pagination (accounting.report.reports.unapplied.PageParams attribute), 57
 pagination (accounting.report.reports.unmatched.PageParams attribute), 60
 Pagination (class in accounting.utils.pagination), 76
 parse_query_keywords() (in module accounting.utils.query), 78
 period (accounting.report.reports.balance_sheet.PageParams attribute), 42
 period (accounting.report.reports.income_expenses.PageParams attribute), 45
 period (accounting.report.reports.income_statement.PageParams attribute), 47
 period (accounting.report.reports.journal.PageParams attribute), 47

- attribute), 49
 - period (accounting.report.reports.ledger.PageParams attribute), 52
 - period (accounting.report.reports.trial_balance.PageParams attribute), 55
 - Period (class in accounting.report.period.period), 38
 - period_chooser (accounting.report.reports.balance_sheet.PageParams attribute), 42
 - period_chooser (accounting.report.reports.income_expenses.PageParams attribute), 45
 - period_chooser (accounting.report.reports.income_statement.PageParams attribute), 47
 - period_chooser (accounting.report.reports.journal.PageParams attribute), 49
 - period_chooser (accounting.report.reports.ledger.PageParams attribute), 52
 - period_chooser (accounting.report.reports.trial_balance.PageParams attribute), 55
 - period_spec() (in module accounting.report.utils.csv_export), 64
 - PeriodChooser (class in accounting.report.period.chooser), 36
 - PeriodConverter (class in accounting.report.converters), 70
 - pgettext() (in module accounting.locale), 83
 - pk_column (accounting.utils.user.UserUtilityInterface property), 80
 - populate_obj() (accounting.account.forms.AccountForm method), 4
 - populate_obj() (accounting.currency.forms.CurrencyForm method), 8
 - populate_obj() (accounting.journal_entry.forms.journal_entry.JournalEntryForm method), 15
 - populate_obj() (accounting.journal_entry.forms.line_item.CreditLineItemForm method), 17
 - populate_obj() (accounting.journal_entry.forms.line_item.DebitLineItemForm method), 18
 - populate_obj() (accounting.option.forms.OptionForm method), 33
 - PositiveAmount (class in accounting.journal_entry.forms.line_item), 20
 - post_update() (accounting.account.forms.AccountForm method), 4
 - PREPOSITIONS (in module accounting.utils.title_case), 79
- ## Q
- query_values (accounting.journal_entry.utils.account_option.AccountOption attribute), 21
 - query_values (accounting.models.Account property), 85
 - query_values (accounting.models.BaseAccount property), 86
 - query_values (accounting.models.JournalEntryLineItem property), 92
- ## R
- recurring (accounting.journal_entry.utils.description_editor.DescriptionEditor attribute), 23
 - recurring (accounting.option.forms.OptionForm attribute), 33
 - recurring (accounting.utils.options.Options property), 74
 - Recurring (class in accounting.utils.options), 75
 - recurring_data (accounting.utils.options.Options property), 75
 - RecurringExpenseForm (class in accounting.option.forms), 33
 - RecurringForm (class in accounting.option.forms), 33
 - RecurringIncomeForm (class in accounting.option.forms), 34
 - RecurringItem (class in accounting.utils.options), 75
 - RecurringItemForm (class in accounting.option.forms), 34
 - Redirection, 77
 - render_create_template() (accounting.journal_entry.utils.operators.CashDisbursementJournalEntry method), 25
 - render_create_template() (accounting.journal_entry.utils.operators.CashReceiptJournalEntry method), 26
 - render_create_template() (accounting.journal_entry.utils.operators.JournalEntryOperator method), 27
 - render_create_template() (accounting.journal_entry.utils.operators.TransferJournalEntry method), 28
 - render_detail_template() (accounting.journal_entry.utils.operators.CashDisbursementJournalEntry method), 26
 - render_detail_template() (accounting.journal_entry.utils.operators.CashReceiptJournalEntry method), 26

`render_detail_template()` (*accounting.journal_entry.utils.operators.JournalEntryOperator* *ReportAccount* (class in *accounting.report.reports.balance_sheet*), 42
method), 27

`render_detail_template()` (*accounting.journal_entry.utils.operators.TransferJournalEntry* *ReportAccount* (class in *accounting.report.reports.income_statement*), 47
method), 28

`render_edit_template()` (*accounting.journal_entry.utils.operators.CashDisbursementJournalEntry* *ReportChooser* (class in *accounting.report.utils.report_chooser*), 65
method), 26

`render_edit_template()` (*accounting.journal_entry.utils.operators.CashReceiptJournalEntry* *ReportLineItem* (class in *accounting.report.reports.income_expenses*), 45
method), 27

`render_edit_template()` (*accounting.journal_entry.utils.operators.JournalEntryOperator* *ReportLineItem* (class in *accounting.report.reports.journal*), 50
method), 28

`render_edit_template()` (*accounting.journal_entry.utils.operators.JournalEntryOperator* *ReportType* (class in *accounting.report.reports.ledger*), 52
method), 28

`render_edit_template()` (*accounting.journal_entry.utils.operators.TransferJournalEntry* *ReportType* (class in *accounting.report.utils.report_type*), 66
method), 28

S

`report_chooser` (*accounting.report.reports.balance_sheet.PageParams* *s()* (in module *accounting.utils.cast*), 71
property), 42

`report_chooser` (*accounting.report.reports.income_expenses.PageParams* *SameAccountAsOriginalLineItem* (class in *accounting.journal_entry.forms.line_item*), 20
property), 45

`report_chooser` (*accounting.report.reports.income_statement.PageParams* *SameCurrencyAsOriginalLineItems* (class in *accounting.journal_entry.forms.currency*), 11
property), 47

`report_chooser` (*accounting.report.reports.journal.PageParams* *save_order()* (*accounting.account.forms.AccountReorderForm*
property), 50 *method*), 5

`report_chooser` (*accounting.report.reports.ledger.PageParams* *save_order()* (*accounting.journal_entry.forms.reorder.JournalEntryReorderForm*
property), 52 *method*), 20

`report_chooser` (*accounting.report.reports.search.PageParams* *SEARCH* (*accounting.report.utils.report_type.ReportType*
property), 54 *attribute*), 66

`report_chooser` (*accounting.report.reports.trial_balance.PageParams* *Search* (class in *accounting.report.reports.search*), 54
property), 55

`report_chooser` (*accounting.report.reports.unapplied.PageParams* *Section* (class in *accounting.report.reports.balance_sheet*), 42
property), 57

`report_chooser` (*accounting.report.reports.unapplied_accounts.PageParams* *Section* (class in *accounting.report.reports.income_statement*), 48
property), 59

`report_chooser` (*accounting.report.reports.unmatched.PageParams* *sections* (*accounting.report.reports.income_statement.PageParams*
property), 61 *attribute*), 47

`report_chooser` (*accounting.report.reports.unmatched_accounts.PageParams* *selectable_credit()* (*accounting.models.Account*
property), 62 *class method*), 85

`report_chooser` (*accounting.report.utils.base_page_params.BasePageParams* *selectable_debit()* (*accounting.models.Account*
property), 63 *class method*), 86

`report_chooser` (*accounting.report.reports.unmatched_accounts.PageParams* *selected_base* (*accounting.account.forms.AccountForm*
property), 62 *property*), 4

`report_chooser` (*accounting.report.reports.unmatched_accounts.PageParams* *since_last_month_url* (*accounting.report.period.chooser.PeriodChooser*
property), 62 *attribute*), 36

`report_chooser` (*accounting.report.reports.unmatched_accounts.PageParams* *SinceLastMonth* (class in *accounting.report.period.shortcuts*), 39
property), 62

`report_chooser` (*accounting.report.utils.base_page_params.BasePageParams* *sort_accounts_in()* (in module *accounting.account.forms*), 5
property), 63

`report_chooser` (*accounting.report.reports.unmatched_accounts.PageParams* *sort_journal_entries_in()* (in module *accounting.journal_entry.forms.reorder*), 20
property), 63

- spec (*accounting.report.period.period.Period* attribute), 39
- sql_condition() (*accounting.utils.current_account.CurrentAccount* class method), 72
- start (*accounting.report.period.period.Period* attribute), 39
- status (*accounting.report.utils.offset_matcher.OffsetMatcher* property), 64
- str (*accounting.utils.current_account.CurrentAccount* attribute), 72
- strip_multiline_text() (in module *accounting.utils.strip_text*), 79
- strip_text() (in module *accounting.utils.strip_text*), 79
- Subsection (class in *accounting.report.reports.balance_sheet*), 43
- Subsection (class in *accounting.report.reports.income_statement*), 48
- subsections (*accounting.report.reports.balance_sheet.Section* attribute), 42
- subsections (*accounting.report.reports.income_statement.Section* attribute), 48
- ## T
- T (class in *accounting.journal_entry.forms.journal_entry*), 16
- T (class in *accounting.utils.pagination*), 77
- T (class in *accounting.utils.user*), 80
- tags (*accounting.journal_entry.utils.description_editor.DescriptionType* property), 24
- TemplatePeriod (class in *accounting.report.period.shortcuts*), 39
- text (*accounting.report.reports.income_statement.CSVRow* attribute), 46
- text (*accounting.report.reports.trial_balance.CSVRow* attribute), 54
- text (*accounting.utils.pagination.Link* attribute), 76
- text2html() (in module *accounting.journal_entry.template_filters*), 31
- this_month_url (*accounting.report.period.chooser.PeriodChooser* attribute), 36
- this_year_url (*accounting.report.period.chooser.PeriodChooser* attribute), 36
- ThisMonth (class in *accounting.report.period.shortcuts*), 39
- ThisYear (class in *accounting.report.period.shortcuts*), 39
- title (*accounting.account.forms.AccountForm* attribute), 4
- title (*accounting.journal_entry.utils.account_option.AccountOption* attribute), 21
- title (*accounting.journal_entry.utils.description_editor.DescriptionAccount* property), 22
- title (*accounting.models.Account* property), 86
- title (*accounting.models.AccountL10n* attribute), 86
- title (*accounting.models.BaseAccount* property), 87
- title (*accounting.models.BaseAccountL10n* attribute), 87
- title (*accounting.report.reports.balance_sheet.CSVHalfRow* attribute), 41
- title (*accounting.report.reports.balance_sheet.Section* attribute), 42
- title (*accounting.report.reports.balance_sheet.Subsection* attribute), 43
- title (*accounting.report.reports.income_statement.AccumulatedTotal* attribute), 46
- title (*accounting.report.reports.income_statement.Section* attribute), 48
- title (*accounting.report.reports.income_statement.Subsection* attribute), 48
- title (*accounting.report.utils.option_link.OptionLink* attribute), 65
- title (*accounting.utils.current_account.CurrentAccount* attribute), 72
- title_case() (in module *accounting.utils.title_case*), 79
- title_l10n (*accounting.models.Account* attribute), 86
- title_l10n (*accounting.models.BaseAccount* attribute), 87
- to_keep (*accounting.journal_entry.forms.journal_entry.LineItemCollector* attribute), 15
- to_python() (*accounting.account.converters.AccountConverter* method), 3
- to_python() (*accounting.base_account.converters.BaseAccountConverter* method), 6
- to_python() (*accounting.currency.converters.CurrencyConverter* method), 7
- to_python() (*accounting.journal_entry.converters.DateConverter* method), 30
- to_python() (*accounting.journal_entry.converters.JournalEntryConverter* method), 30
- to_python() (*accounting.journal_entry.converters.JournalEntryTypeConverter* method), 30
- to_python() (*accounting.report.converters.CurrentAccountConverter* method), 69
- to_python() (*accounting*...)

[url \(accounting.report.reports.income_expenses.ReportLineItem attribute\), 46](#)
[url \(accounting.report.reports.income_statement.ReportAccounting attribute\), 47](#)
[url \(accounting.report.reports.ledger.ReportLineItem attribute\), 53](#)
[url \(accounting.report.reports.trial_balance.ReportAccounting attribute\), 55](#)
[url \(accounting.report.utils.option_link.OptionLink attribute\), 65](#)
[url_template \(accounting.report.period.chooser.PeriodChooser attribute\), 36](#)
[user_cls \(in module accounting.utils.user\), 81](#)
[user_pk_column \(in module accounting.utils.user\), 81](#)
[UserUtilityInterface \(class in accounting.utils.user\), 80](#)

V

[value \(accounting.models.Option attribute\), 92](#)
[values \(accounting.report.reports.balance_sheet.CSVRow property\), 41](#)
[values \(accounting.report.reports.income_expenses.CSVRow property\), 44](#)
[values \(accounting.report.reports.income_statement.CSVRow property\), 46](#)
[values \(accounting.report.reports.journal.CSVRow property\), 49](#)
[values \(accounting.report.reports.ledger.CSVRow property\), 51](#)
[values \(accounting.report.reports.trial_balance.CSVRow property\), 54](#)
[values \(accounting.report.reports.unapplied.CSVRow property\), 56](#)
[values \(accounting.report.reports.unapplied_accounts.CSVRow property\), 58](#)
[values \(accounting.report.reports.unmatched.CSVRow property\), 60](#)
[values \(accounting.report.reports.unmatched_accounts.CSVRow property\), 62](#)
[values \(accounting.report.utils.csv_export.BaseCSVRow property\), 64](#)
[VERSION \(in module accounting\), 93](#)

Y

[year_url\(\) \(accounting.report.period.chooser.PeriodChooser method\), 36](#)
[YearPeriod \(class in accounting.report.period.shortcuts\), 40](#)
[Yesterday \(class in accounting.report.period.shortcuts\), 40](#)
[yesterday_url \(accounting.report.period.chooser.PeriodChooser attribute\), 37](#)

W

[whole_form \(accounting.journal_entry.forms.currency.CashDisbursementCurrencyForm attribute\), 10](#)
[whole_form \(accounting.journal_entry.forms.currency.CashReceiptCurrencyForm attribute\), 11](#)
[whole_form \(accounting.journal_entry.forms.currency.CurrencyForm attribute\), 11](#)
[whole_form \(accounting.journal_entry.forms.currency.TransferCurrencyForm attribute\), 12](#)