2016

odoo

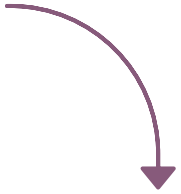
EXPERIENCE

How to adapt Odoo Accounting to your own country

Quentin De Paoli • developer

Country-adapted accounting is

super important



- Comfort zone for end-users
- Usable out of the box
- Avoid complex config/reports

1. How it works
2. The different objects
3. Country-adapted demo data
4. How to easily create a financial report

1 How it works

How it works

- Done in account/__init__.py: *_auto_install_l10n()*
- If no localization module found:
  installs l10n_generic_coa (US)
- May install other localized modules (SEPA, …)
- l10n_xx module must contain a .yml file calling *try_loading_for_current_company()*

The different objects

Chart template

- Bank and cash prefixes
- Properties: default accounts for
  - Receivable, payable, expense, income, stock operations, multi-currencies use cases
- Number of digits in your COA
- Anglo saxon or continental accounting
- Account for inter-bank transfers
- Currency
- Languages in case of l10n_multilang dependency

Chart template

```xml
<?xml version="1.0" encoding="utf-8"?>
<odoo>
        <record id="trans" model="account.account.template">
            <field name="name">Transferts</field>
            <field name="code">580</field>
            <field name="reconcile" eval="True"/>
            <field name="user_type_id" ref="account.data_account_type_current_assets"/>
        </record>
        <!-- Chart template -->
        <record id="l10nbe_chart_template" model="account.chart.template">
            <field name="name">Belgian PCMN</field>
            <field name="transfer_account_id" ref="trans"/>
            <field name="currency_id" ref="base.EUR"/>
            <field name="spoken_languages" eval="'nl_BE'"/>
        </record>
        <record id="trans" model="account.account.template">
            <field name="chart_template_id" ref="l10nbe_chart_template"/>
        </record>
</odoo>
```

Chart template

```xml
<?xml version="1.0" encoding="utf-8"?>
<odoo>
        <record id="l10nbe_chart_template" model="account.chart.template">
            <field name="name">Belgian PCMN</field>
            <field name="bank_account_code_prefix">550</field>
            <field name="cash_account_code_prefix">570</field>
            <field name="code_digits">6</field>
            <field name="transfer_account_id" ref="trans"/>
            <field name="property_account_receivable_id" ref="a4000"/>
            <field name="property_account_payable_id" ref="a440"/>
            <field name="property_account_expense_categ_id" ref="a600"/>
            <field name="property_account_income_categ_id" ref="a7010"/>
            <field name="property_stock_account_input_categ_id" ref="..."/>
            <field name="property_stock_account_output_categ_id" ref="..."/>
            <field name="property_stock_valuation_account_id" ref="..."/>
            <field name="expense_currency_exchange_account_id" ref="a654"/>
            <field name="income_currency_exchange_account_id" ref="a754"/>
            <!--
            <field name="complete_tax_set" eval="True"/>
            <field name="use_anglo_saxon" eval="False"/>
            -->
        </record>
</odoo>
```

Chart of Accounts

- Not too much accounts (200-300)
- No liquidity accounts
- Only 1 payable/receivable account
- Reuse account types from account module
- Tags on accounts for
  - an accurate cash flow statement
  - custom reports

Taxes

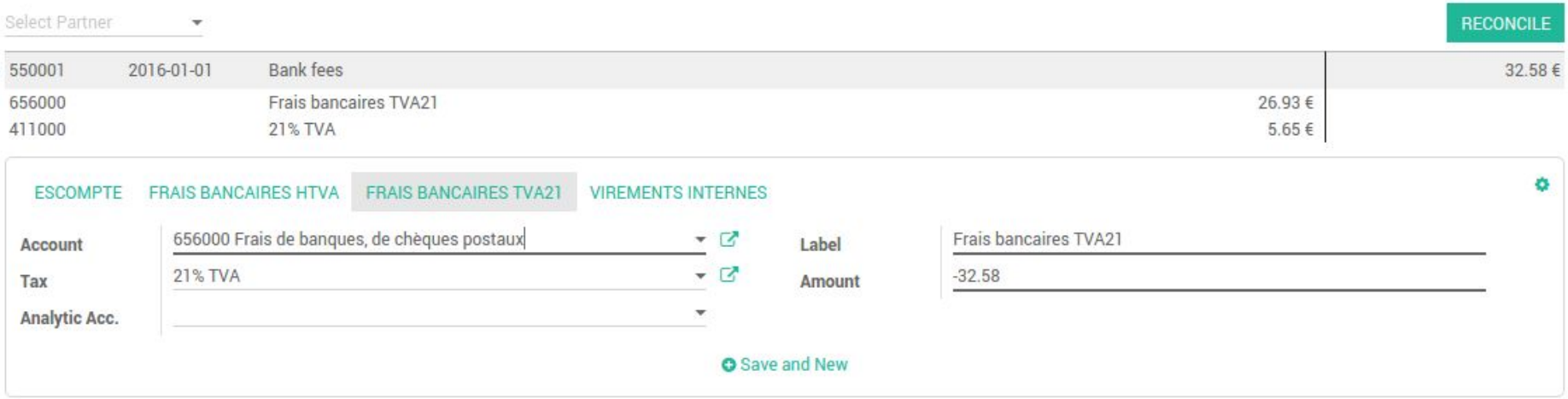
- Chart of Taxes in enterprise module
  - One tag per line in chart of taxe statement
  - Export in XML available
  - Tax adjustment wizard for manual input
- Could be
  - Simple rates set on the chart template
  - Complete set of tax template

Fiscal positions

- Optional
- Map accounts
- Map taxes
- Replace onchanges' defaults in SO, PO and invoices

Bank Operations

- New in v10
- Define common operations to use in bank statement reconciliation widget



Bank Operations

```
<record id="frais_bancaires_tva21_template" model="account.reconcile.model.template">
    <field name="name">Frais bancaires TVA21</field>
    <field name="account_id" ref="a656"/>
    <field name="amount_type">percentage</field>
    <field name="tax_id" ref="attn_TVA-21-inclus-dans-prix"/>
    <field name="amount">100</field>
    <field name="label">Frais bancaires TVA21</field>
</record>
```

3

Country-adapted demo data

Demo Data

- Meaningful demo data is important
- Example: l10n_be/demo/l10n_be_demo.yml

```
-
  Set the demo tags on account templates and on their respective accounts (already generated during the loading
of data)
-
  !python {model: account.account.template, id: False}: |
      mapping_list = [
          ('a1000', 'account.demo_capital_account'),
          ('a300', 'account.demo_stock_account'),
          ('a7600', 'account.demo_sale_of_land_account'),
          ('a6201', 'account.demo_ceo_wages_account'),
          ('a24011', 'account.demo_coffee_machine_account'),
      ]
      for xml_id, tag in mapping_list:
          account_template = self.browse(ref(xml_id))
          account_template.write({'tag_ids': [(4, ref(tag))]})
          accounts = self.env['account.account'].search([('code', 'like', account_template.code)])
          if accounts:
              accounts.write({'tag_ids': [(4,ref(tag))]})
```

4

How to easily create a financial report

Accounting Reports

Financial

- Very easy to create (XML)
- Ready to use
- Only works with amls
- Only with sums over a period

Custom

- More advanced (Python)
- Possibilities are much more open
- Talk by Cédric "*How to create custom accounting reports*"

Report Object

```
<record id="account_financial_report_profitandloss0" model="account.financial.html.report">
    <field name="name">Profit and Loss</field>
    <field name="debit_credit" eval="False"/>
    <field name="report_type">date_range_analytic</field>
    <field name='parent_id' ref='account_reports_legal_statements_menu'/>
</record>
```

- One *account.financial.html.report* object per report
- *name*: Name (title) of the report
- *debit_credit*: Toggles debit and credit columns (default = False)
- *report_type*:
  - *date_range* (Profit & Loss)
  - *no_date_range* (Balance Sheet)
  - … + *_analytic* (Toggles analytic filter)
  - *date_range_cash* (Cash Method by default)
- *tax_report*: set to True for tax statements
- *parent_id*: menuitem under which this report should appear

Report Line Object

```
<record id="account_financial_report_gross_profit0" model="account.financial.html.report.line">
    <field name="name">Gross Profit</field>
    <field name="code">GRP</field>
    <field name="formulas">balance = OPINC.balance - COS.balance</field>
    <field name="parent_id" ref='account_financial_report_totalincome0'/>
    <field name="sequence" eval="1"/>
    <field name="level" eval="2" />
</record>
```

- *name*: what's displayed on the report
- *code*: can be reused in other report lines
- *parent_id*: another account.financial.html.report.line
- *sequence*: sets their display order
- *level*: determines the layout

Report Line Object

```
<record id="account_financial_report_income0" model="account.financial.html.report.line">
    <field name="name">Operating Income</field>
    <field name="code">OPINC</field>
    <field name="formulas">balance = -sum.balance</field>
    <field name="parent_id" ref='account_financial_report_gross_profit0'/>
    <field name="domain" eval="[('account_id.user_type_id', '=', ref('account.data_account_type_revenue'))]" />
    <field name="groupby">account_id</field>
    <field name="sequence" eval="1"/>
    <field name="level" eval="3" />
</record>
```

- *formulas*: Assigns a value to the columns (*balance= X [;debit=Y; credit=Z]*)
- Available objects in '*formulas*':
  - *Ndays*: number of days in selected period (date_range reports)
  - another report line: *<code>.balance*, *<code>.credit*, *<code>.debit*, *<code>.amount_residual*
  - *sum*, *sum_if_pos*, *sum_if_neg*
- *domain*: An Odoo domain on the account move line object
- *groupby*: Group the account move lines by one of their columns

Other useful fields

- *financial_report_id*: For the root financial report lines. Must link to the related *accout.financial.html.report* object
- *figure_type*: how to format the columns of the line. Can be *'float'* (default, for monetary values), *'percents'* or *'no_unit'*.
- *hide_if_zero*: False by default. If True and the line result is falsy, hides the line.
- *show_domain*: How the domain of a line is displayed. Can be *'foldable'* (default, hidden at the start but can be unfolded), *'always'* (always displayed) or *'never'* (never shown).
- *green_on_positive*: Used when computing the comparison column. True (default) if growth is good (displayed in green) or not.

Other useful fields

- *special_date_changer*: If a specific line in a report should not use the same dates as the rest of the report. Can be
  - *'normal'*: default,
  - *'strict_range'*: force the accounts with include_initial_balance set to True to consider only the date range,
  - *'from_beginning'*: start date is *epoch*, eg. *unnaffected earnings line* in a balance sheet,
  - *'to_beginning_of_period'*: start date is *epoch* and end date is the start date of the selected period,
- *action_id*: Many2one linking to an action that will be executed when clicking on the line name in the report.

2016

odoo

EXPERIENCE

Thank you.

#odooexperience