odoo

How to use the reporting framework to create custom accounting reports

PIERRE VYNCKE, R&D DEVELOPER

SUMMARY

1. Financial Reports vs Custom Reports
2. Demo
3. Steps to create your report
4. Further customization
5. Q&A

odoo

odoo

FINANCIAL VS CUSTOM REPORTS

FEATURES COMPARISON

FEATURES COMPARISON I

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

FEATURES COMPARISON II

Most account reports are financial reports :
Profit and Loss
Balance Sheet
Cash Summary
Executive Summary
Aged Partner Balances

Most localizations reports are also financial reports :
Belgian Profit & Loss
Belgian Balance Sheet
Belgian VAT Statement

Custom reports are few :
Bank Reconciliation (works with bank statement lines not move lines)
General Ledger (more columns, initial balance line)
Generic Tax Report (columns not sum of domain)
Belgian Partner VAT Listing (columns not sum of domain)
Followups (more options, more columns, actions, ...)

odoo

odoo

DEMO

odoo

STEPS TO CREATE YOUR REPORT

REPORT MODEL INTERFACE I

Your report should be an abstract model that implements the following interface :

```
@api.model
def get_title(self):

@api.model
def get_name(self):

@api.model
def get_template(self):

@api.model
def get_report_type(self):

@api.model
def get_lines(self, context_id, line_id=None)
```

REPORT MODEL INTERFACE I

```
@api.model
def get_report_type(self):
```

Available report types : date_range (Profit & Loss), no_date_range (Balance Sheet), date_range_extended (Aged Partner Balances), date_range_cash (Cash Method by default), no_comparison (General Ledger), or *add your own* !

REPORT MODEL INTERFACE II

```
@api.model
def get_lines(self, context_id, line_id=None):
```

The line_id parameter is used when unfolding a line. If line_id is set, this method should return only the line corresponding to line_id and its domain, nothing else.

This method should return the lines as a list of dictionnaries. Each dictionnary corresponds to one line. The lines should be in the same order as they need to be when displayed.

REPORT MODEL INTERFACE III

Dictionnaries for each line should contain the following :

id : So it can be refered to for footnotes, unfolding, as the active_id if an action is defined etc.

name : What will be displayed

type : The type of the line. This will determine what action is available when clicking on the line. Types supported until now : account_id, line, tax_id, unreconciled_aml, bank_statement_id, partner_id, move_line_id, too_many, payment. Or again, *add your own* !

footnotes : If footnotes have been created for one of the columns or for the line name (fetched from context). A dictionnary {column_number : footnote_id}

unfoldable : If the line can be unfolded

unfolded : If the line has already been unfolded (fetched from context) – only relevant if unfoldable = True

columns : A list with the column values.

level : Determines the layout.

action_id : An action that will be triggered when clicking on the line.

colspan : If you want the name to span over many columns, use this.

CREATE A CONTEXT

Report contexts are transient models that are instanciated once per report and per user. You need to create a new context model per custom report. Its interface should be :

```
_inherit = "account.report.context.common"

def get_report_obj(self):

def get_columns_names(self):

@api.multi
def get_columns_types(self):
```

Column types can be : text, date or number

LAST STEPS

A few last steps before it is in full working order :

Change the dictionnary methods of the account.report.context.common object.

_report_name_to_report_model() :

Add an item name_of_your_report: name_of_the_model

_report_model_to_report_context() :

Add an item name_of_the_model: name_of_the_context_model

Create the client action and the menuitem for your report.

```
<record id="action_account_report_general_ledger" model="ir.actions.client">
    <field name="name">General Ledger</field>
    <field name="tag">account_report_generic</field>
    <field name="context" eval="{'url': '/account/general_ledger/1', 'model': 'account.general.ledger'}" />
</record>
<menuitem id="menu_action_account_report_general_ledger"
    name="General Ledger" action="action_account_report_general_ledger" parent="account.menu_finance_reports"/>
```

odoo

FURTHER CUSTOMISATION

TEMPLATES

SEARCH VIEW

You can customise the search view of your report and adapt it to your needs.

- Create a new report_type to fit your needs, eg. 'journal_filter'.
- Add fields to your context object that correspond to the filters you want to add.
  eg. journal_id = fields.Many2one('account.journal')
- Modify the template 'accountReports.searchView' by adding a dropdown for your filter.

```
<div class="btn-group btn-group-sm oe-account-journal-filter" t-if="report_type == 'journal_filter'">
    <a type="button" class="btn btn-default dropdown-toggle" data-toggle="dropdown">
        <span class="fa fa-bar-chart"/> Journal Filter: <span class="caret"/>
    </a>
    <ul class="dropdown-menu filters-menu" role="menu">
        <li title="Bank" data-value="bank" class="oe-account-one-journal-filter"><a>Bank Journal</a></li>
        ...
    </ul>
</div>
```

- Bind events to your dropdown in the render_searchview_buttons() function of the account_report_generic widget

```
this.$searchview_buttons.find('.oe-account-one-journal-filter').bind('click', function (event) {
    var url = self.base_url + '?journal_filter=' + $(event.target).parents('li').data('value');
    self.$el.attr({src: url});
});
```

odoo

LINE TYPE

You can add your own line type and change the way it will appear on the report and what actions will be available. For that, inherit from and modify the 'account.report_financial_line' template.

To trigger an action, use the 'oe-account-web-action' class and then :

- Use data-res-model and data-active-id to open the form view of the object.
- Use data-action-name to trigger that action. You've got the option to add the data-force-context attribute to force the current context to be passed on with the action (only relevent if the action goes to another accounting report).
- Use data-action-id to trigger that action.

odoo

Thank You

For any question : pvy@odoo.com

Odoo
sales@odoo.com
+32 (0) 2 290 34 90
www.odoo.com

R&D and services office
Chaussée de Namur 40
B-1367 Grand Rosière

Sales office
Avenue Van Nieuwenhuyse 5
B-1160 Brussels