Reports are written in HTML/QWeb, like all regular views in Odoo. You can use the usual [QWeb control flow tools](https://www.odoo.com/documentation/10.0/reference/qweb.html" \l "reference-qweb). The PDF rendering itself is performed by [wkhtmltopdf](http://wkhtmltopdf.org).

If you want to create a report on a certain model, you will need to define this [Report](https://www.odoo.com/documentation/10.0/reference/reports.html#reference-reports-report) and the [Report template](https://www.odoo.com/documentation/10.0/reference/reports.html#reference-reports-templates) it will use. If you wish, you can also specify a specific [Paper Format](https://www.odoo.com/documentation/10.0/reference/reports.html#reference-reports-paper-formats) for this report. Finally, if you need access to more than your model, you can define a [Custom Reports](https://www.odoo.com/documentation/10.0/reference/reports.html#reference-reports-custom-reports) class that gives you access to more models and records in the template.

**Report**

Every report must be declared by a [report action](https://www.odoo.com/documentation/10.0/reference/actions.html#reference-actions-report).

For simplicity, a shortcut <report> element is available to define a report, rather than have to set up [the action](https://www.odoo.com/documentation/10.0/reference/actions.html" \l "reference-actions-report) and its surroundings manually. That <report> can take the following attributes:

id

the generated record's [external id](https://www.odoo.com/documentation/10.0/glossary.html" \l "term-external-id)

name (mandatory)

only useful as a mnemonic/description of the report when looking for one in a list of some sort

model (mandatory)

the model your report will be about

report\_type (mandatory)

either qweb-pdf for PDF reports or qweb-html for HTML

report\_name

the name of your report (which will be the name of the PDF output)

groups

[Many2many](https://www.odoo.com/documentation/10.0/reference/orm.html" \l "odoo.fields.Many2many" \o "odoo.fields.Many2many) field to the groups allowed to view/use the current report

attachment\_use

if set to True, the report will be stored as an attachment of the record using the name generated by the attachment expression; you can use this if you need your report to be generated only once (for legal reasons, for example)

attachment

python expression that defines the name of the report; the record is acessible as the variable object

paperformat

external id of the paperformat you wish to use (defaults to the company's paperformat if not specified)

Example:

<report

id="account\_invoices"

model="account.invoice"

string="Invoices"

report\_type="qweb-pdf"

name="account.report\_invoice"

file="account.report\_invoice"

attachment\_use="True"

attachment="(object.state in ('open','paid')) and

('INV'+(object.number or '').replace('/','')+'.pdf')"

/>

**Report template**

**Minimal viable template**

A minimal template would look like:

<template id="report\_invoice">

<t t-call="report.html\_container">

<t t-foreach="docs" t-as="o">

<t t-call="report.external\_layout">

<div class="page">

<h2>Report title</h2>

<p>This object's name is <span t-field="o.name"/></p>

</div>

</t>

</t>

</t>

</template>

Calling external\_layout will add the default header and footer on your report. The PDF body will be the content inside the <div class="page">. The template's id must be the name specified in the report declaration; for example account.report\_invoice for the above report. Since this is a QWeb template, you can access all the fields of the docs objects received by the template.

There are some specific variables accessible in reports, mainly:

docs

records for the current report

doc\_ids

list of ids for the docs records

doc\_model

model for the docs records

time

a reference to [time](https://docs.python.org/2/library/time.html" \l "module-time" \o "(in Python v2.7)) from the Python standard library

user

res.user record for the user printing the report

res\_company

record for the current user's company

If you wish to access other records/models in the template, you will need [a custom report](https://www.odoo.com/documentation/10.0/reference/reports.html" \l "reference-reports-custom-reports).

**Translatable Templates**

If you wish to translate reports (to the language of a partner, for example), you need to define two templates:

* The main report template
* The translatable document

You can then call the translatable document from your main template with the attribute t-lang set to a language code (for example fr or en\_US) or to a record field. You will also need to re-browse the related records with the proper context if you use fields that are translatable (like country names, sales conditions, etc.)

Warning

If your report template does not use translatable record fields, re-browsing the record in another language is *not* necessary and will impact performances.

For example, let's look at the Sale Order report from the Sale module:

<!-- Main template -->

<template id="report\_saleorder">

<t t-call="report.html\_container">

<t t-foreach="docs" t-as="doc">

<t t-call="sale.report\_saleorder\_document" t-lang="doc.partner\_id.lang"/>

</t>

</t>

</template>

<!-- Translatable template -->

<template id="report\_saleorder\_document">

<!-- Re-browse of the record with the partner lang -->

<t t-set="doc" t-value="doc.with\_context({'lang':doc.partner\_id.lang})" />

<t t-call="report.external\_layout">

<div class="page">

<div class="oe\_structure"/>

<div class="row">

<div class="col-xs-6">

<strong t-if="doc.partner\_shipping\_id == doc.partner\_invoice\_id">Invoice and shipping address:</strong>

<strong t-if="doc.partner\_shipping\_id != doc.partner\_invoice\_id">Invoice address:</strong>

<div t-field="doc.partner\_invoice\_id" t-options="{&quot;no\_marker&quot;: True}"/>

<...>

<div class="oe\_structure"/>

</div>

</t>

</template>

The main template calls the translatable template with doc.partner\_id.lang as a t-lang parameter, so it will be rendered in the language of the partner. This way, each Sale Order will be printed in the language of the corresponding customer. If you wish to translate only the body of the document, but keep the header and footer in a default language, you could call the report's external layout this way:

<t t-call="report.external\_layout" t-lang="en\_US">

Tip

Please take note that this works only when calling external templates, you will not be able to translate part of a document by setting a t-lang attribute on an xml node other than t-call. If you wish to translate part of a template, you can create an external template with this partial template and call it from the main one with the t-lang attribute.

**Barcodes**

Barcodes are images returned by a controller and can easily be embedded in reports thanks to the QWeb syntax:

<img t-att-src="'/report/barcode/QR/%s' % 'My text in qr code'"/>

More parameters can be passed as a query string

<img t-att-src="'/report/barcode/?

type=%s&value=%s&width=%s&height=%s'%('QR', 'text', 200, 200)"/>

**Useful Remarks**

* Twitter Bootstrap and FontAwesome classes can be used in your report template
* Local CSS can be put directly in the template
* Global CSS can be inserted in the main report layout by inheriting its template and inserting your CSS:
* <template id="report\_saleorder\_style" inherit\_id="report.style">
* <xpath expr=".">
* <t>
* .example-css-class {
* background-color: red;
* }
* </t>
* </xpath>
* </template>
* If it appears that your PDF report is missing the styles, please check [these instructions](https://www.odoo.com/documentation/10.0/howtos/backend.html" \l "reference-backend-reporting-printed-reports-pdf-without-styles).

**Paper Format**

Paper formats are records of report.paperformat and can contain the following attributes:

name (mandatory)

only useful as a mnemonic/description of the report when looking for one in a list of some sort

description

a small description of your format

format

either a predefined format (A0 to A9, B0 to B10, Legal, Letter, Tabloid,...) or custom; A4 by default. You cannot use a non-custom format if you define the page dimensions.

dpi

output DPI; 90 by default

margin\_top, margin\_bottom, margin\_left, margin\_right

margin sizes in mm

page\_height, page\_width

page dimensions in mm

orientation

Landscape or Portrait

header\_line

boolean to display a header line

header\_spacing

header spacing in mm

Example:

<record id="paperformat\_frenchcheck" model="report.paperformat">

<field name="name">French Bank Check</field>

<field name="default" eval="True"/>

<field name="format">custom</field>

<field name="page\_height">80</field>

<field name="page\_width">175</field>

<field name="orientation">Portrait</field>

<field name="margin\_top">3</field>

<field name="margin\_bottom">3</field>

<field name="margin\_left">3</field>

<field name="margin\_right">3</field>

<field name="header\_line" eval="False"/>

<field name="header\_spacing">3</field>

<field name="dpi">80</field>

</record>

**Custom Reports**

The report model has a default get\_html function that looks for a model named report.*module.report\_name*. If it exists, it will use it to call the QWeb engine; otherwise a generic function will be used. If you wish to customize your reports by including more things in the template (like records of others models, for example), you can define this model, overwrite the function render\_html and pass objects in the docargs dictionary:

from odoo import api, models

class ParticularReport(models.AbstractModel):

\_name = 'report.module.report\_name'

@api.model

def render\_html(self, docids, data=None):

report\_obj = self.env['report']

report = report\_obj.\_get\_report\_from\_name('module.report\_name')

docargs = {

'doc\_ids': docids,

'doc\_model': report.model,

'docs': self,

}

return report\_obj.render('module.report\_name', docargs)

**Reports are web pages**

Reports are dynamically generated by the report module and can be accessed directly via URL:

For example, you can access a Sale Order report in html mode by going to http://<server-address>/report/html/sale.report\_saleorder/38

Or you can access the pdf version at http://<server-address>/report/pdf/sale.report\_saleorder/38