



# How to create website front-end controllers



Basics of server-side rendering



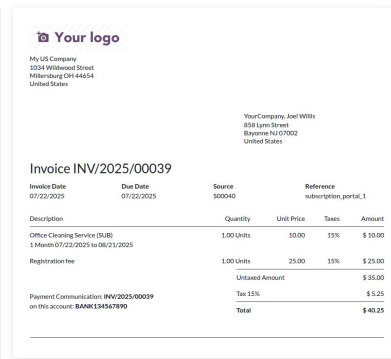
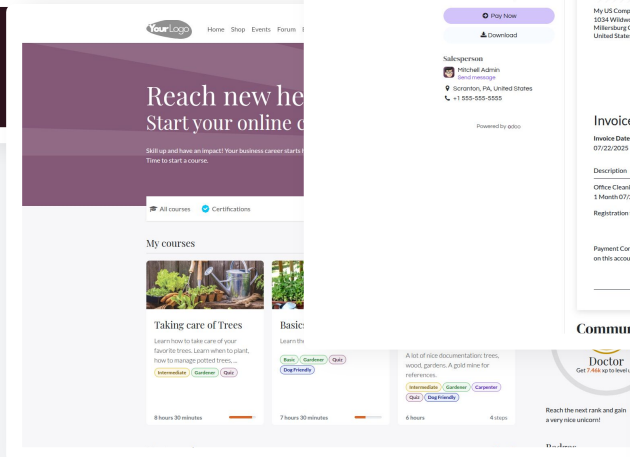
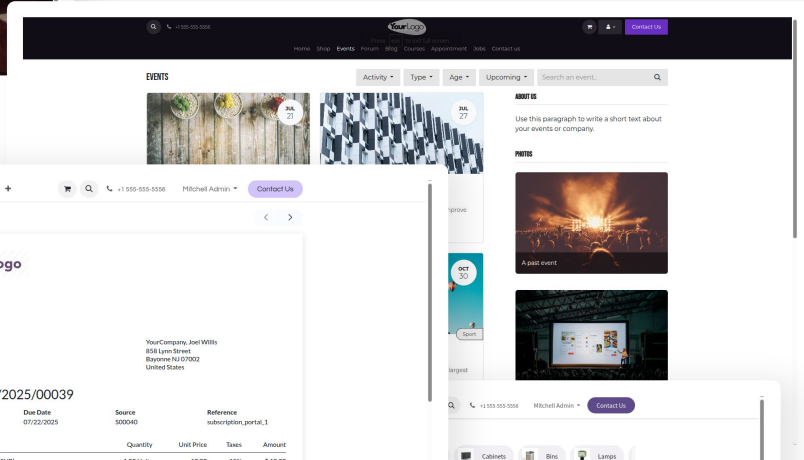
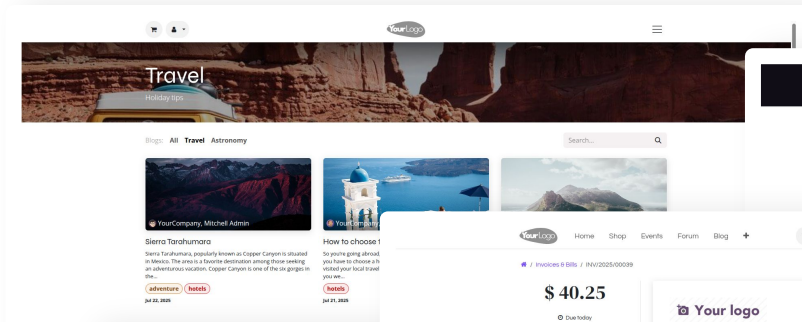
*This is a **basic introduction**  
Refer to the actual documentation  
for full details*

*Scan to ask your  
questions*

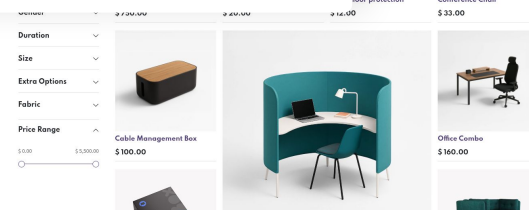


**Benoit Socias**, Research & Development

<https://pad.odoo.com/p/oxp25-frontend-controllers>



#### Communication history



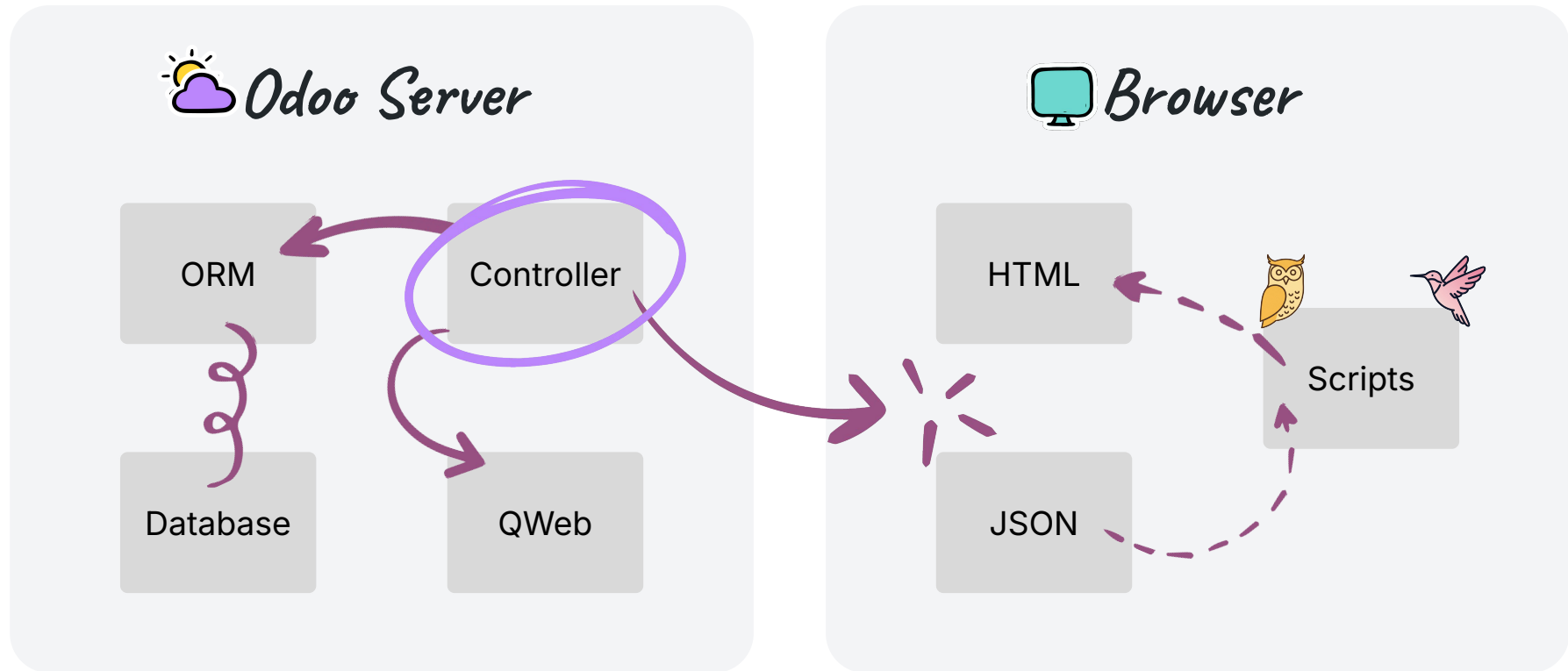
*Dynamic Content from Server*



*Being called through*

# *Controllers*

# What are Controllers?



# Controllers

Class



## Note

A single addon can have several controllers

```
# In /myapp/controllers/main.py
```

```
from odoo import http
```

```
class MyApp(http.Controller):
```

```
$ ls -l addons/portal/controllers
__init__.py
mail.py
message_reaction.py
portal.py
portal_thread.py
thread.py
web.py
```

# Routes

## End points

- Describe **URLs** to match
- Feeds **typed** parameters to method

## Syntax

```
<type:name>
```

## Supports Odoo models

```
<model("res.user"):user>
```

## Slugs examples found on odoo.com

odoo-experience-2025-**6601**

how-to-create-website-front-end-controllers-**8769**

software-developer-**1**

```
# In /myapp/controllers/main.py
```

```
from odoo import http
from odoo.http import request
```

```
class MyApp(http.Controller):
```

```
    @http.route([
        '/myapp/mystuff',
        '/myapp/mystuff/<int:page>',
        '/myapp/mystuff/<model("my.stuff"):stuff>',
    ], type='http', auth='public', website=True)
    def mystuff(self, stuff=None, page=1):
        # Use ORM
        records = request.env['some.model'].search(...)
        # Render with QWeb
        values = {
            'records': records,
            'stuff': stuff,
        }
        return request.render('myapp.mytemplate', values)
```

# Types

## Protocols

### http

- Reads further parameters from **encoded form**
- Wraps result into a **text/html** response

### jsonrpc

- Reads further parameters from **XmlHttpRequest's params**
- Wraps result into an **application/json** response

```
# In /myapp/controllers/main.py
```

```
from odoo import http
from odoo.http import request
```

```
class MyApp(http.Controller):
```

```
    @http.route([
        '/myapp/mystuff',
        '/myapp/mystuff/<int:page>',
        '/myapp/mystuff/<model("my.stuff"):stuff>',
    ], type='http', auth='public', website=True)
    def mystuff(self, stuff=None, page=1):
        # Use ORM
        records = request.env['some.model'].search(...)
        # Render with QWeb
        values = {
            'records': records,
            'stuff': stuff,
        }
        return request.render('myapp.mytemplate', values)
```



# Authorizations

## Users

### *user*

- Requires the **user** to be authenticated
- Runs with the access rights of that **user**

### *public*

- **User** may or may not be authenticated
- If not: **Public User** is used  
(base.public\_user)

```
# In /myapp/controllers/main.py
```

```
from odoo import http
from odoo.http import request
```

```
class MyApp(http.Controller):
```

```
    @http.route([
        '/myapp/mystuff',
        '/myapp/mystuff/<int:page>',
        '/myapp/mystuff/<model("my.stuff"):stuff>',
    ], type='http', auth='public', website=True)
    def mystuff(self, stuff=None, page=1):
        # Use ORM
        records = request.env['some.model'].search(...)
        # Render with QWeb
        values = {
            'records': records,
            'stuff': stuff,
        }
        return request.render('myapp.mytemplate', values)
```

# Access the ORM

## Models & Fields

- Use any **ORM** method through the env

```
# In /myapp/controllers/main.py
```

```
from odoo import http
from odoo.http import request
```

```
class MyApp(http.Controller):
```

```
    @http.route([
        '/myapp/mystuff',
        '/myapp/mystuff/<int:page>',
        '/myapp/mystuff/<model("my.stuff"):stuff>',
    ], type='http', auth='public', website=True)
    def mystuff(self, stuff=None, page=1):
        # Use ORM
        records = request.env['some.model'].search(...)
        # Render with QWeb
        values = {
            'records': records,
            'stuff': stuff,
        }
        return request.render('myapp.mytemplate', values)
```

# Render a template

## QWeb

- Prepare the rendering context
- Render a **QWeb template** from the request

For a JSONRPC route, return a dictionary

```
return { 'key': value }
```

```
# In /myapp/controllers/main.py
```

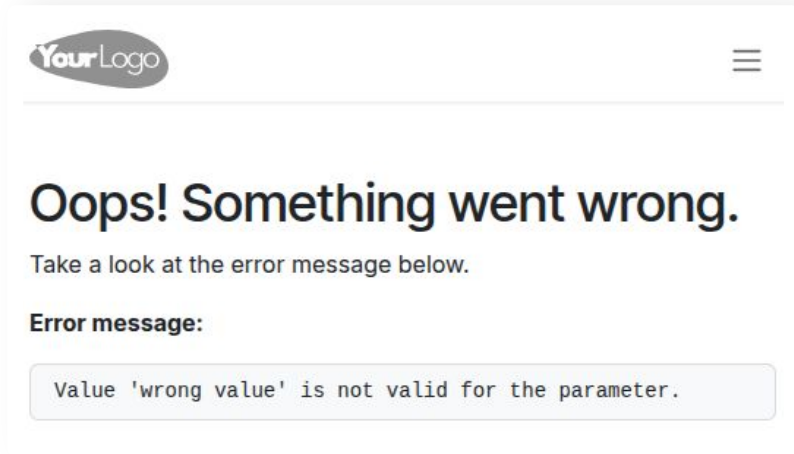
```
from odoo import http
from odoo.http import request
```

```
class MyApp(http.Controller):
```

```
    @http.route([
        '/myapp/mystuff',
        '/myapp/mystuff/<int:page>',
        '/myapp/mystuff/<model("my.stuff"):stuff>',
    ], type='http', auth='public', website=True)
    def mystuff(self, stuff=None, page=1):
        # Use ORM
        records = request.env['some.model'].search(...)
        # Render with QWeb
        values = {
            'records': records,
            'stuff': stuff,
        }
        return request.render('myapp.mytemplate', values)
```

# Errors

## Handling problems



Note: Runtime errors are wrapped as **500** http status

*# When called through a route*

```
import werkzeug.exceptions
from odoo.exceptions import UserError
from odoo import _, http
...
@http.route(...)
def morestuff(self, some_param):
    ...
    # Misc error types are available in odoo.exceptions
    raise UserError(_(
        "Value %(param)r is not valid for the parameter.",
        param=some_param))
    ...
    # The network layer library provides predefined errors
    raise werkzeug.exceptions.NotFound()
    ...
    # You can return any kind of response
    return request.make_response(content, headers=[
        ('Content-Type', 'application/pdf'), ...
    ], status=200)
```

# JSON RPC call

(Too) Simple Example



Do not do this at home !

Component or Interaction might have been  
destroyed before async call returns

```
// Inside Owl Component or Colibri Interaction
```

```
import { rpc } from "@web/core/network/rpc";
```

```
async someMethod() {  
  const result = await rpc("/myapp/myjsonstuff", {  
    someParam: "Some value",  
    otherParam: 123,  
  });  
  return result.something;  
}
```

# POST requests

Be safe

- Restrict **HTTP** method types
- Rely on defaults

*# Example from /website/controllers/form.py*

```
@http.route('/website/form/<string:model_name>',
            type='http',
            auth='public',
            methods=['POST'],
            website=True,
            csrf=False,
            captcha='website_form'
)
def website_form(self, model_name, **kwargs):
    ...
    # Manual handling of the CSRF token
```

*Using QWeb templates for*  
**Rendering**

Free Returns and Standard Shipping

4 My Cart

Mitchell Admin ▾

[Home](#) [Shop](#) [Events](#) [Forum](#) [Blog](#) [Courses](#) [Appointment](#) [Jobs](#) +

YourLogo

Search...



Contact Us

Order > Address > Payment

## Order summary



Conference Chair - Steel

[Remove](#) [Save for Later](#)

\$ 66.00

- 2 +



Office Lamp

[Remove](#) [Save for Later](#)

\$ 40.00

- 1 +



Large Desk

Minimalist wooden desk for executive use

[Remove](#) [Save for Later](#)

\$ 1,799.00

- 1 +

## Suggested accessories



Storage Box

Blue plastic bin for flexible office storage

\$ 15.80

Add to cart

Condition

Delivery	-
Subtotal	\$ 1,905.00
Taxes	\$ 285.75
<b>Total</b>	<b>\$ 2,190.75</b>

Gift card or discount code...

Apply

**Loyalty Points** 21907.5

**Free Product - Simple Pen**

5 Loyalty Points

Claim

Pay with Demo

Checkout >

or

< Continue shopping

QWeb Quick Reminder



# QWeb quick reminder

## Cheat sheet

- Conditions

t-if

t-else

- Loops

t-foreach

t-as

- Access variables and **ORM** records

t-att-\*

t-attf-\*

t-out

t-field

t-set

```
<!-- In /myapp/views/template.xml -->
```

```
<template id="myapp.mytemplate" name="My records">
  <t t-if="records">
    <h1>My records</h1>
    <div t-foreach="records" t-as="record">
      <a
        t-field="record.name"
        t-att-href="record.website_url"
        t-attf-class="myapp_{{record.type}}"
      />
      <div t-out="record.get_description()"/>
    </div>
  </t>
  <t t-else="">
    <div class="alert-info">
      There are no records.
    </div>
  </t>
</template>
```

# QWeb quick reminder

Rendered

## Example with records

record 1:

name: "First"

record 2:

name: "Second"

```
<!-- Rendered with records -->
```

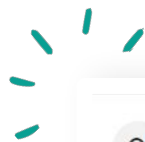
```
<h1>My records</h1>
<div>
  <a href="/myapp/mystuff/first-1"
    class="myapp_abc">First</a>
  <div>Method result for First</div>
  <a href="/myapp/mystuff/second-2"
    class="myapp_xyz">Second</a>
  <div>Method result for Second</div>
</div>
```

*Demo*



*Fitting inside*

*Website pages*



Menu Item

## Choose your appointment

Search... 

Header



### Dental Care

📍 Doctor's Office

Schedule your visit today and experience expert dental care brought right to your doorstep.

Book now



### Tennis Court

📍 Tennis Club

Easily reserve tennis courts for your next match, anytime and anywhere.

Book now



### Online Cooking Lesson

📺 Online \$ 150.00

Discover the secrets kept in high-end kitchens with one of our starred chefs, from the comfort of your own home.

Book now



### Table

📍 Bistr-Odoo

Reserve your table at our bistro and enjoy a delightful dining experience at your convenience.

Book now

# Fitting in Website

## Layout & Menu

- If the page displays a specific record, set it in **main\_object** in the rendering context

```
<!-- In /myapp/views/template.xml -->
```

```
<template id="myapp.mytemplate" name="My records">  
  <t-call="website.layout">  
    <div id="wrap" class="myapp_stuff">  
      <!-- Actual content -->  
      <div class="oe_structure oe_empty"  
        id="oe_structure_example"/>  
    </div>  
  </t>  
</template>
```

# Fitting in Website

## Layout & Menu

- If the page displays a specific record, set it in **main\_object** in the rendering context
- Hint: Specifying a **class** is an easy way to find out about this context from JS/CSS

```
<!-- In /myapp/views/template.xml -->
```

```
<template id="myapp.mytemplate" name="My records">  
  <t t-call="website.layout">  
    <div id="wrap" class="myapp_stuff">  
      <!-- Actual content -->  
      <div class="oe_structure oe_empty"  
        id="oe_structure_example"/>  
    </div>  
  </t>  
</template>
```

# Fitting in Website

## Layout & Menu

- If the page displays a specific record, set it in **main\_object** in the rendering context
- Hint: Specifying a **class** is an easy way to find out about this context from JS/CSS
- You can add editable drop-zones for the website builder with **oe\_structure**

```
<!-- In /myapp/views/template.xml -->
```

```
<template id="myapp.mytemplate" name="My records">
  <t t-call="website.layout">
    <div id="wrap" class="myapp_stuff">
      <!-- Actual content -->
      <div class="oe_structure oe_empty"
        id="oe_structure_example"/>
    </div>
  </t>
</template>
```



# Fitting in Website

## Layout & Menu

- If the page displays a specific record, set it in **main\_object** in the rendering context
- Hint: Specifying a **class** is an easy way to find out about this context from JS/CSS
- You can add editable drop-zones for the website builder with **oe\_structure**

```
<!-- In /myapp/views/template.xml -->
```

```
<template id="myapp.mytemplate" name="My records">
  <t t-call="website.layout">
    <div id="wrap" class="myapp_stuff">
      <!-- Actual content -->
      <div class="oe_structure oe_empty"
        id="oe_structure_example"/>
    </div>
  </t>
</template>
```

```
<!-- In /myapp/data/menu.xml -->
```

```
<record id="menu_stuff" model="website.menu">
  <field name="name" model="Stuff"/>
  <field name="url">/myapp/mystuff</field>
  <field name="parent_id" model="website.main_menu"/>
  <field name="sequence" type="int">25</field>
</record>
```



### Individual Workplace

Private pod with desk and chair

\$ 885.00



Add to Cart



### Desk Stand with Screen

\$ 2,100.00



Add to Cart



### Drawer Black

\$ 25.00



Add to Cart



### Large Meeting Table

Big table seats 10-12 for team discussions

\$ 4,000.00



Add to Cart

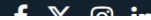
Pager

## Designed for companies

We are a team of passionate people whose goal is to improve everyone's life through disruptive products. We build great products to solve your business problems. Our products are designed for small to medium size companies willing to optimize their performance.

**My Company**  
8000 Marina Blvd, Suite 300  
Brisbane CA 94005  
United States

+1 555-555-5556  
hello@mycompany.com



*Fitting inside Website*

# Pager

## Browsing records

- Fetch records and compute count using the same **domain**

*# In a route method of /myapp/controllers/main.py*

```
records = request.env['some.model'].search(domain,  
    offset=(page - 1) * self._records_per_page,  
    limit=self._records_per_page)  
total = request.env['some.model'].search_count(domain)
```

# Pager

## Browsing records

- Fetch records and compute count using the same **domain**
- Put the **pager** in the rendering context

*# In a route method of /myapp/controllers/main.py*

```
records = request.env['some.model'].search(domain,
    offset=(page - 1) * self._records_per_page,
    limit=self._records_per_page)
total = request.env['some.model'].search_count(domain)
pager = odoo.tools.lazy(lambda: request.website.pager(
    url=request.httprequest.path.partition('/page/')[0],
    total=total,
    page=page,
    step=self._records_per_page,
    url_args=url_args,
))
```

# Pager

## Browsing records

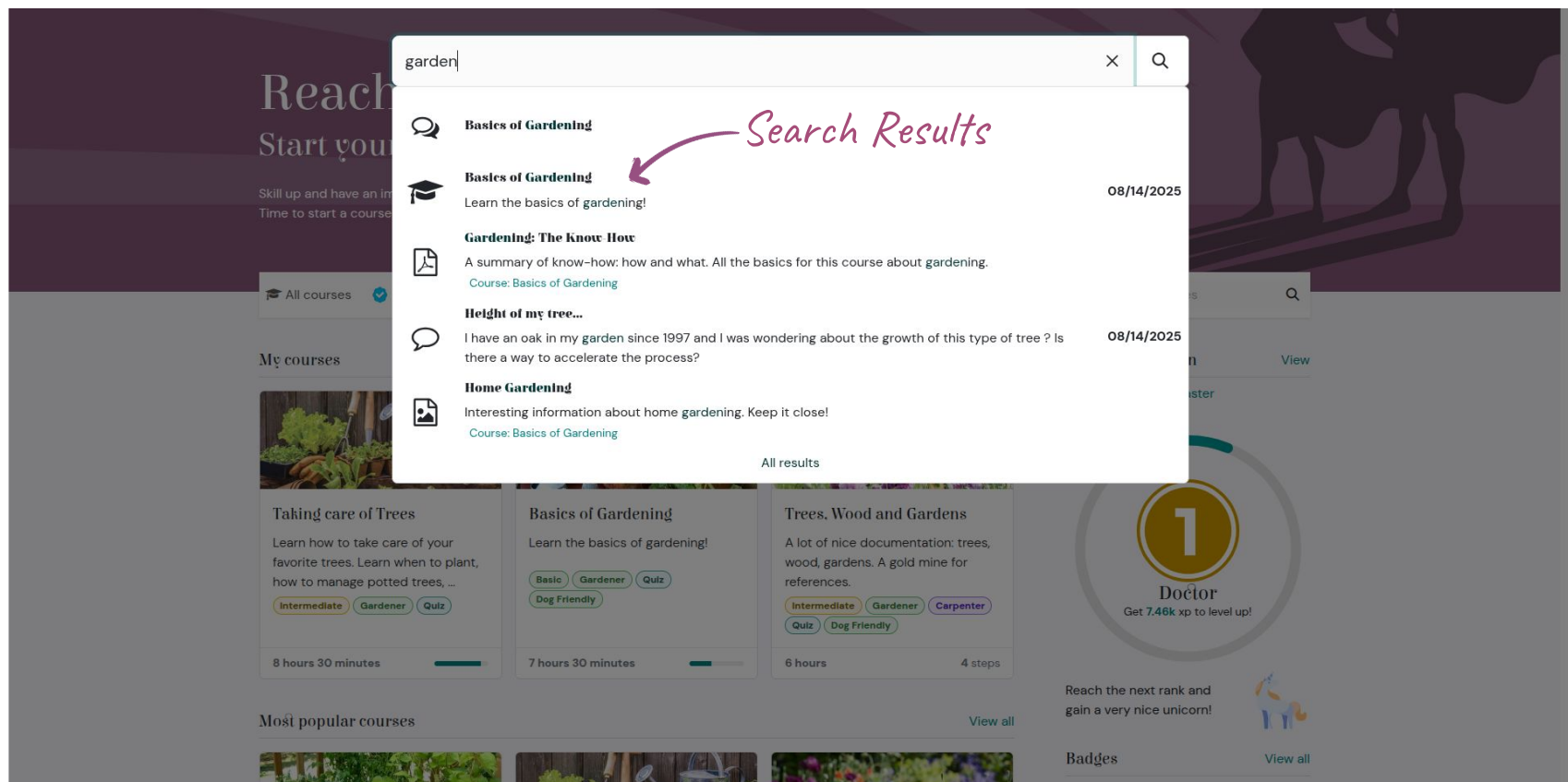
- Fetch records and compute count using the same **domain**
- Put the **pager** in the rendering context
- Simply **t-call** the **website.pager** template inside your template

*# In a route method of /myapp/controllers/main.py*

```
records = request.env['some.model'].search(domain,
    offset=(page - 1) * self._records_per_page,
    limit=self._records_per_page)
total = request.env['some.model'].search_count(domain)
pager = odoo.tools.lazy(lambda: request.website.pager(
    url=request.httprequest.path.partition('/page/')[0],
    total=total,
    page=page,
    step=self._records_per_page,
    url_args=url_args,
))
```

*<!-- In a template of /myapp/views/template.xml -->*

```
<t t-call="website.pager">
  <t t-set="classname"
    t-valueef="justify-content-center"/>
  <t t-set="extraLinkClass" t-valueef="extra_class"/>
</t>
```



*Fitting inside Website*

# Search

## Types of searches

- **Dispatch** the search according to the **search\_type**
- Use **all** for the **Everything** search type

```
# In /myapp/models/website.py
```

```
from odoo import models
```

```
class Website(models.Model):
```

```
    _inherit = 'website'
```

```
    def _search_get_details(
```

```
        self, search_type, order, options
```

```
):
```

```
    result = super()._search_get_details(
```

```
        search_type, order, options)
```

```
    if search_type in ['mystuff', 'all']:
```

```
        result.append(self.env['some.model']
```

```
            ._search_get_detail(self, order, options))
```

```
    return result
```

# Search

## Types of searches

- **Dispatch** the search according to the **search\_type**
- Use **all** for the **Everything** search type

```
# In /myapp/models/website.py
```

```
from odoo import models
```

```
class Website(models.Model):  
    _inherit = 'website'
```

```
    def _search_get_details(  
        self, search_type, order, options  
    ):  
        result = super()._search_get_details(  
            search_type, order, options)  
        if search_type in ['mystuff', 'all']:  
            result.append(self.env['some.model']  
                ._search_get_detail(self, order, options))  
        return result
```



# Search

## Results mapping

```
# In /myapp/controllers/portal.py
```

```
_inherit = [  
    'website.published.multi.mixin',  
    'website.searchable.mixin',  
]  
  
@api.model  
def _search_get_detail(self, website, order, options):  
    return {  
        'model': 'some.stuff',  
        'base_domain': [website.website_domain()],  
        'search_fields': ['name', 'description'],  
        'fetch_fields': ['name', 'description',  
                          'website_url'],  
        'mapping': {  
            'name': {'name': 'name', 'type': 'text',  
                     'match': True},  
            'description': {'name': 'description',  
                             'type': 'text', 'match': True},  
            'website_url': {'name': 'website_url',  
                             'type': 'text', 'truncate': False},  
        },  
        'icon': 'fa-question',  
        'order': 'create_date desc, id desc',  
    }
```

# Search

## Results mapping

- Put the target URL of the result in

`website_url`



```
# In /myapp/controllers/portal.py
```

```
_inherit = [  
    'website.published.multi.mixin',  
    'website.searchable.mixin',  
]  
  
@api.model  
def _search_get_detail(self, website, order, options):  
    return {  
        'model': 'some.stuff',  
        'base_domain': [website.website_domain()],  
        'search_fields': ['name', 'description'],  
        'fetch_fields': ['name', 'description',  
                          'website_url'],  
        'mapping': {  
            'name': {'name': 'name', 'type': 'text',  
                     'match': True},  
            'description': {'name': 'description',  
                             'type': 'text', 'match': True},  
            'website_url': {'name': 'website_url',  
                             'type': 'text', 'truncate': False},  
        },  
        'icon': 'fa-question',  
        'order': 'create_date desc, id desc',  
    }  
}
```

## Our latest content

Check out what's new in our company !

See all →



Water can

\$ 12.00



Flower pot

\$ 4.50



Basics of Furniture Creation

\$ 200.00



Taking care of Trees Course

\$ 150.00

Dynamic Snippets

## Our latest content

Check out what's new in our company !

See all →



Fitting inside Website

# Dynamic Snippet

## Filters

- Define **which** records to search with a classic `ir.filters`
- Make filter **available** for dynamic snippets with a `website.snippet.filter`

```
<!-- In /myapp/data/filter.xml -->
```

```
<record id="dynamic_snippet_new_stuff_filter"
  model="ir.filters">
  <field name="name">New Stuff</field>
  <field name="model_id">some.model</field>
  <field name="user_ids" eval="False" />
  <field name="domain">[]</field>
  <field name="sort">
    ["write_date desc, id desc"]</field>
  <field name="action_id" ref="website.action_website"/>
</record>
```

```
<record id="dynamic_filter_new_stuff"
  model="website.snippet.filter">
  <field name="name">New Stuff</field>
  <field name="filter_id"
    ref="myapp.dynamic_snippet_new_stuff_filter"/>
  <field name="field_names">name,description</field>
  <field name="limit" eval="16"/>
</record>
```

# Dynamic Snippet

## Display

Each record contains:

- Formatted values for each **field** specified in **field\_names**
- The actual **ORM** record in **\_record**
- The **record**'s **website\_url** in **call\_to\_action\_url**



Template is located by matching its **id** with **dynamic\_filter\_template\_<type>\_\***

```
<!-- In /myapp/views/snippets/snippets.xml -->
```

```
<template id="dynamic_filter_template_some_model_list"
  name="List">
  <div t-foreach="records" t-as="data"
    class="s_mystuff"
    data-column-classes="col-12 col-sm-6 col-lg-4">
    <t t-set="record" t-value="data['_record']"/>
    <a class="text-decoration-none text-reset"
      t-att-href="data['call_to_action_url']">
      <h4 class="mt-2">
        <span t-if="is_sample" class="badge">
          Sample
        </span>
        <span t-field="record.name"/>
      </h4>
      <span t-field="record.description"/>
    </a>
  </div>
</template>
```

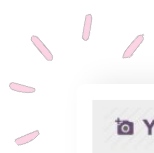
*Demo*

 *Fitting inside*




# Portal pages





Breadcrumbs

 Your logo

Joel Willis ▾

 / Sales Orders / Sales Order S00020

\$ 3,389.63

## Sales Order - S00020

 View Details

### Sale Information

Order Date: 06/24/2025

### Invoicing and Shipping Address

YourCompany, Joel Willis  
858 Lvn Street

Layout

Sales Order - S00020  
Communication history

Your contact

 Mitchell Admin  
Send message

Connect with your software!

Powered by 

 Your logo

Joel Willis ▾

## My account

1 Quotations to review

Alert



### Your Orders

Follow, view or pay your orders



### Your Invoices

Follow, download or pay your invoices



### Addresses

Add, remove or modify your addresses



### Connection & Security

Configure your connection parameters



Joel Willis  
YourCompany

858 Lynn Street  
Bayonne NJ 07002  
United States  
(683)-556-5104  
joel.willis63@example.com

 Edit information

Links

Fitting inside Portal



# Fitting in Portal

Layout

```
<!-- In /myapp/views/template.xml -->
```

```
<template id="myapp.mytemplate" name="My records">  
  <t t-call="portal.portal_layout">  
    <div id="wrap" class="myapp_stuff">  
      <!-- Actual content -->  
    </div>  
  </t>  
</template>
```

# Fitting in Portal

## Alert & Links

- In **placeholder\_count** specify the context variable name inside which you will store the record count
- For alerts, replace **client** by **alert** and specify classes in **bg-color**

```
<!-- In /myapp/views/template.xml -->
```

```
<template id="myapp.portal_my_home"
  name="Portal home for myapp"
  customize_show="True"
  inherit_id="portal.portal_my_home">
  <xpath expr="//div[class='o_portal_docs']"
    position="before"/>
  <t t-set="portal_client_category_enable"
    t-value="True"/>
</xpath>
<div id="portal_client_category" position="inside">
  <t t-call="portal.portal_docs_entry">
    <t t-set="icon" t-value="'/myapp/static/...'/>
    <t t-set="title">Your Stuff</t>
    <t t-set="url" t-value="'/my/mystuff'"/>
    <t t-set="text">Manage Your Stuff</t>
    <t t-set="placeholder_count"
      t-value="'myapp_stuff_count'"/>
  </t>
</div>
</template>
```

# Fitting in Portal

## Alert & Links - preparing

- Prepare the variable you specified in **placeholder\_count**

```
# In /myapp/controllers/portal.py
```

```
from odoo.addons.portal.controllers import portal
```

```
class CustomerPortal(portal.CustomerPortal):
```

```
    def _prepare_home_portal_values(self, counters):
        values = super()._prepare_home_portal_values(
            counters)
        if 'myapp_stuff_count' in counters:
            values['myapp_stuff_count'] =
                request.env['myapp_stuff_count']
                    .search_count([])
        return values
```

# Fitting in Portal

## Breadcrumbs

Have information in context to determine

- Should breadcrumb **appear** on page
- Is breadcrumb the **active** one

```
<!-- In /myapp/views/template.xml -->
```

```
<template id="myapp.portal_breadcrumbs"
  name="Portal breadcrumbs for myapp"
  inherit_id="portal.portal_breadcrumbs">
  <xpath expr="//ol[hasclass('o_portal_submenu')]"
    position="inside"/>
    <li t-if="is_mine and not myrecord"
      class="breadcrumb-item active">My Stuff</li>
    <t t-if="myrecord">
      <li class="breadcrumb-item">
        <a href="/my/mystuff">My Stuff</a>
      </li>
      <li class="breadcrumb-item active"
        t-out="myrecord.name"
      />
    </t>
  </xpath>
</template>
```

*Demo*

# Summary

- Accept requests with **routes** inside **Controllers**
- Use **ORM** to build context
- Produce content through **QWeb** templates

Q&A



<https://pad.odoo.com/p/oxp25-frontend-controllers>

# Useful links



*Controllers documentation*

<https://www.odoo.com/documentation/master/developer/reference/backend/http.html>



*Example github repository*

<https://github.com/bs0-odoo/exp25-frontend-controllers>



*For client-side dynamic content :*

*Owl documentation*

[https://www.odoo.com/documentation/master/developer/reference/frontend/owl\\_components.html](https://www.odoo.com/documentation/master/developer/reference/frontend/owl_components.html)

*Owl tutorials*

[https://www.odoo.com/documentation/master/developer/tutorials/discover\\_js\\_framework.html](https://www.odoo.com/documentation/master/developer/tutorials/discover_js_framework.html)



*Interactions*

<https://github.com/odoo/odoo/blob/master/addons/web/static/src/public/interaction.js>



# Thank You!



#OdooExperience



/odoo



@odoo.official



@odoo



# odoo<sup>2025</sup>

EXPERIENCE

