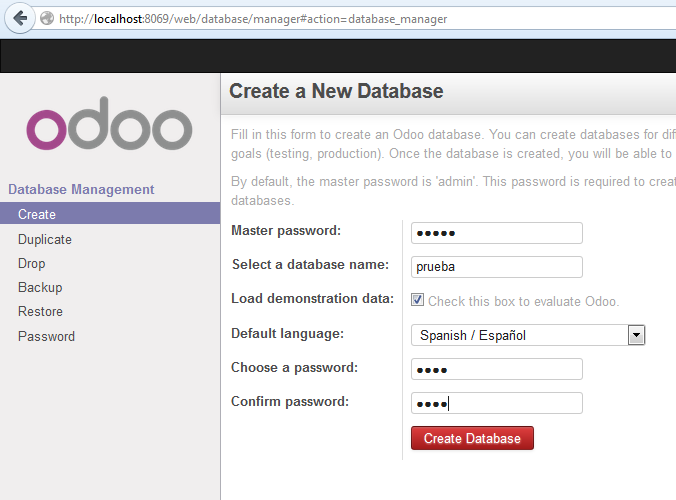


CON PORTABLE

Ejecutamos start.bat

localhost:8069



Base datos: prueba

Usuario: admin

Contraseña: popo

Correo electrónico: admin@yourcompany.example.com

**PARA INSTALAR PLUGINS PROPIOS**

**Modelo vista – controlador**

CONTROLADOR 🡪 Controler.py (lógica)

+

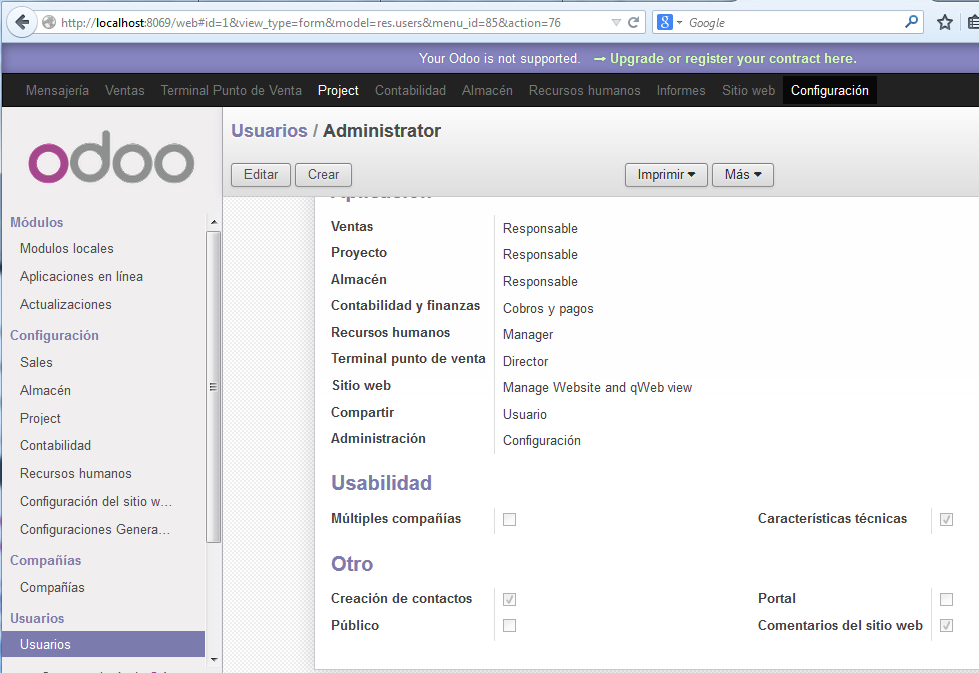
VISTA 🡪 template.xml (cómo nos lo muestra)

+

MODELO 🡪 model.py (datos)

**Tutorial:** [**https://www.odoo.com/documentation/8.0/howtos/website.html**](https://www.odoo.com/documentation/8.0/howtos/website.html)

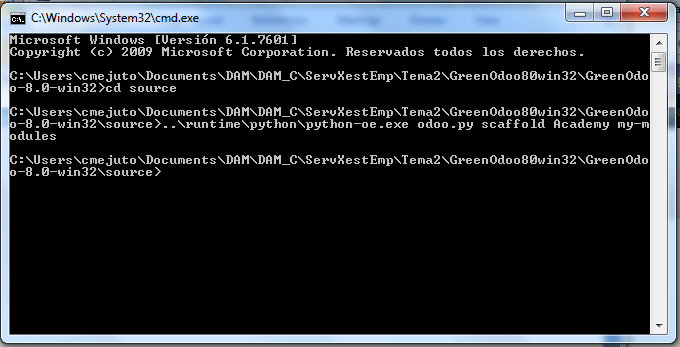
Marcar características técnicas en configuración /administrador /editar para poder instalar módulos propios



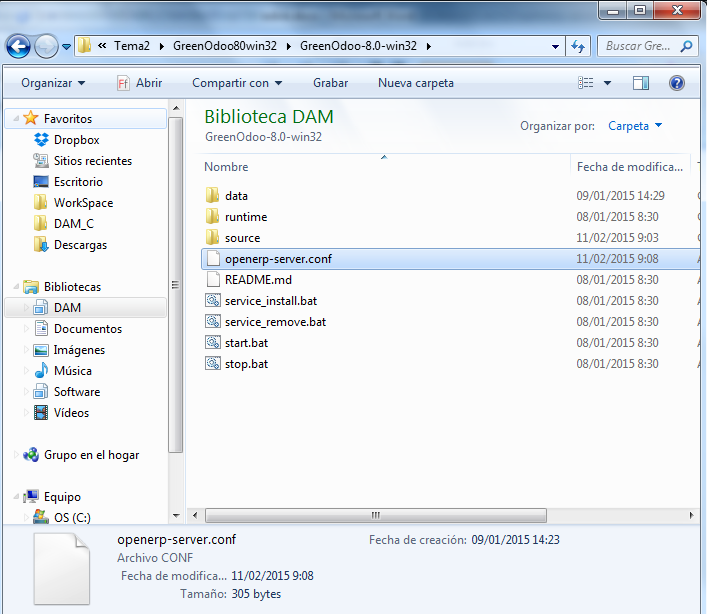
Ahroa tendremos habilitado el ACTUALIZAR LISTA DE MÓDULOS

Ahora aparecerán los módulos que hayamos creado

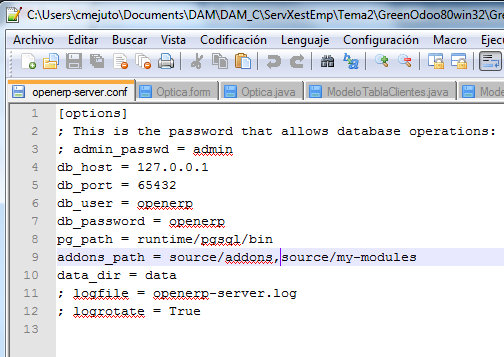
En un cmd o terminal



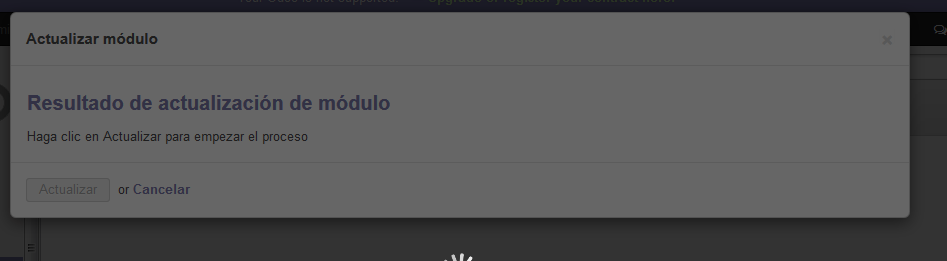
Editamos el fichero openerp-server.conf

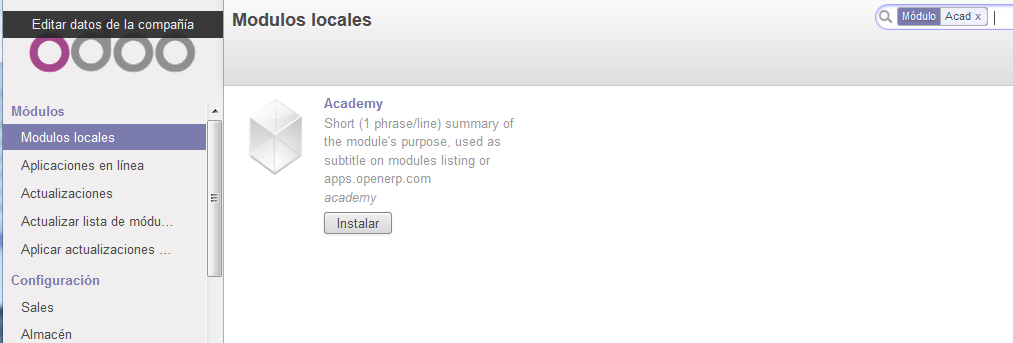


En la línea **addons\_path** añadimos **,source/my-modules**



Actualizamos la lista de módulos y buscamos el módulo que hemos instalado



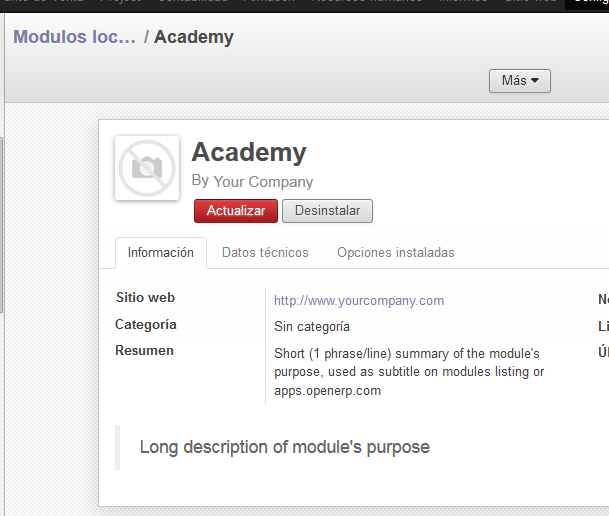


Seguir el siguiente tutorial

[**https://www.odoo.com/documentation/8.0/howtos/website.html**](https://www.odoo.com/documentation/8.0/howtos/website.html)

Cada vez que hagamos algún cambio tenemos que parar el servicio e iniciarlo de nuevo con los bat stop y start, y actualizar el módulo en el botón de Actualizar

Tenemos que abrir otra pestaña con <http://localhost:8069/academy/academy/>



CONFIGURACIÓN

Directorio instalación

Openerp-server.conf

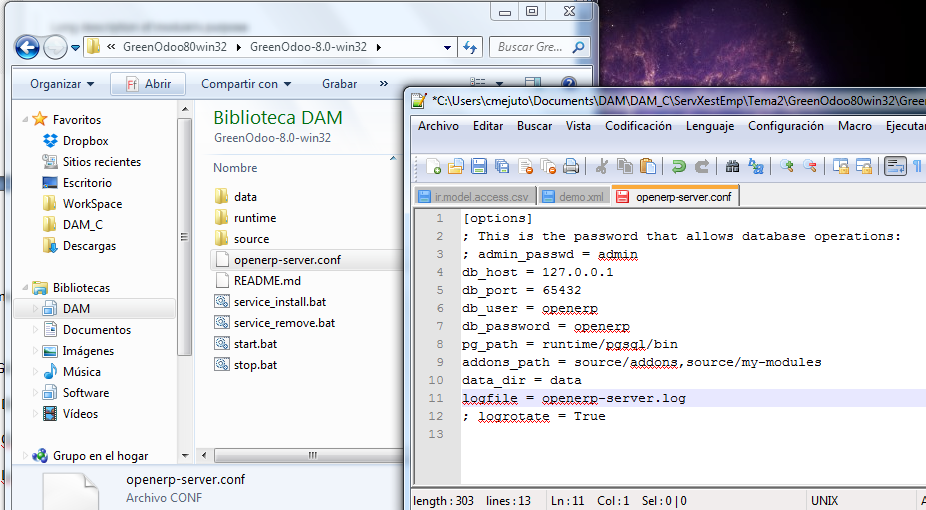
Loglife = openerp.server.log

LOGS ODOO

/var/log/odoo/odoo-server.log

En greenodoo

OJO!! Dicen que con esta línea descomentada puede dar fallo al arrancar el odoo, así que puede que haya que comentarlo de nuevo (ponerle antes un ;)



Descomentamos la línea logfile

**BUILDING THEMES**

<https://www.odoo.com/documentation/8.0/howtos/themes.html>

en linux

./odoo.py scaffold -t theme "Dummy Theme" addons

En windows también funcionaría si odoo está instalado

En greenodoo tenemos que ejecutar este comando desde python.

Abrir consola en carpeta

C:\Users\cmejuto\Documents\DAM\DAM\_C\ServXestEmp\Tema2\GreenOdoo80win32\GreenOdoo-8.0-win32\**source**

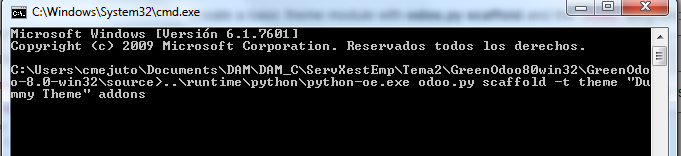
Para ejecutar esto hay que hacerlo añadiendo antes del comando esto:

**..\runtime\python\python-oe.exe**

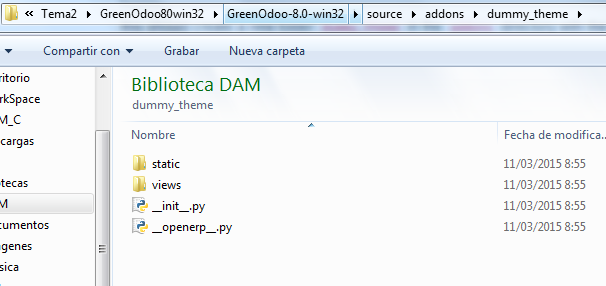
Por lo tanto el comando completo será

**..\runtime\python\python-oe.exe** .py scaffold -t theme "Dummy Theme" addons

Dummy Theme – es el nombre del módulo que vamos usar para este ejemplo.



Esto crea esta carpeta



* En pages.xml odoo/addons/theme\_dummy/views/pages.xml

<openerp>

<data>

<template id="website.hello" name="Homepage" page="True">

<t t-call="website.layout">

<div id="wrap" class="oe\_structure oe\_empty">

</div>

</t>

</template>

</data>

</openerp>

* En la parte Comment the lines about options.xml and pages.xml in \_\_openerp\_\_.py.

Comentamos solo option y snippets y descomentamos pages

# templates, pages, and snippets

'data': [

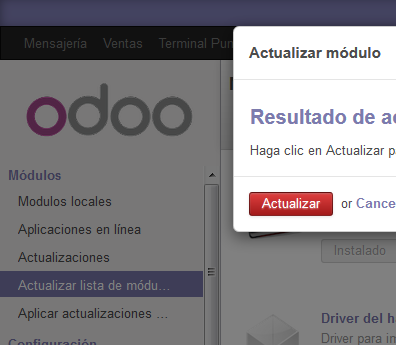
'views/pages.xml',

# 'views/options.xml',

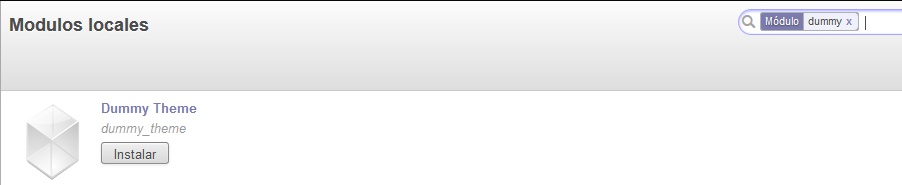
# 'views/snippets.xml',

],

* Actualizamos la lista de módulos



Buscamos el dummy-theme y lo instalamos



<http://localhost:8069/page/hello> -- que estará vacía

#### Editing content on a page

editamos el odoo/addons/theme\_dummy/views/pages.xml que quedará así:

<openerp>

<data>

<template id="website.hello" name="Homepage" page="True">

<t t-call="website.layout">

<div id="wrap" class="oe\_structure oe\_empty">

<section>

<div class="container">

<div class="row">

<h1>This is Your Content</h1>

<p>Isn't amazing to edit everything inline?</p>

<hr/>

</div>

</div>

</section>

</div>

</t>

</template>

<record id="hello\_menu" model="website.menu">

<field name="name">Hello</field>

<field name="url">/page/hello</field>

<field name="parent\_id" ref="website.main\_menu"/>

<field name="sequence" type="int">20</field>

</record>

</data>

</openerp>

### Pimp Your Theme

En C:\Users\cmejuto\Documents\DAM\DAM\_C\ServXestEmp\Tema2\GreenOdoo80win32\GreenOdoo-8.0-win32\source\addons\dummy\_theme\static\style

Editar custom.less

Less es un lenguaje que nos ayuda a compilar css. <http://lesscss.org/>

Con less hacemos unas instrucciones que al compilar nos devuelve css. Hay compiladores en línea que pasan de less a css.

<http://less2css.org/>

el código que hay que compilar es css puro, lo pego en un un custom.css que creo en la misma carpeta. Así quedará

**custom.css**

h1 {

color: #215487;

}

span {

border: 2px solid black;

background-color: #eee;

}

* Editamos de nuevo el pages.xml y añadimos un nuevo template. Quedará así:

<openerp>

<data>

<template id="website.hello" name="Homepage" page="True">

<t t-call="website.layout">

<div id="wrap" class="oe\_structure oe\_empty">

<section>

<div class="container">

<div class="row">

<h1>This is Your Content</h1>

<p>Isn't amazing to edit everything inline?</p>

<hr/>

</div>

</div>

</section>

</div>

</t>

</template>

<template id="dummy\_theme\_asset" name="website assete for Dummy theme" inherit\_id="website.theme">

<xpath expr="." position="inside">

<link rel="stylesheet" href="/dummy\_theme/static/style/custom.css"/>

</xpath>

</template>

<record id="hello\_menu" model="website.menu">

<field name="name">Hello</field>

<field name="url">/page/hello</field>

<field name="parent\_id" ref="website.main\_menu"/>

<field name="sequence" type="int">20</field>

</record>

</data>

</openerp>

Actualizamos el módulo y recargamos la página <http://localhost:8069/page/hello>

#### Get the most of the dom

#### Easy layout with bootstrap

Editamos de nuevo el pages.xml y añadimos una nueva section. Ojo!!!! La section va dentro del uno de los templates.

Quedará asi:

<openerp>

<data>

<template id="website.hello" name="Homepage" page="True">

<t t-call="website.layout">

<div id="wrap" class="oe\_structure oe\_empty">

<section>

<div class="container">

<div class="row">

<h1>This is Your Content</h1>

<p>Isn't amazing to edit everything inline?</p>

<hr/>

</div>

</div>

</section>

<section>

<div class="container">

<div class="row">

<div class="alert alert-primary" role="alert">

<a href="#" class="alert-link">...</a>

</div>

<div class="col-md-6 bg-blue">

<h2>BLUE it!</h2>

</div>

<div class="col-md-6 bg-green">

<h2>GREEN THAT!</h2>

</div>

</div>

</div>

</section>

</div>

</t>

</template>

<template id="dummy\_theme\_asset" name="website assete for Dummy theme" inherit\_id="website.theme">

<xpath expr="." position="inside">

<link rel="stylesheet" href="/dummy\_theme/static/style/custom.css"/>

</xpath>

</template>

<record id="hello\_menu" model="website.menu">

<field name="name">Hello</field>

<field name="url">/page/hello</field>

<field name="parent\_id" ref="website.main\_menu"/>

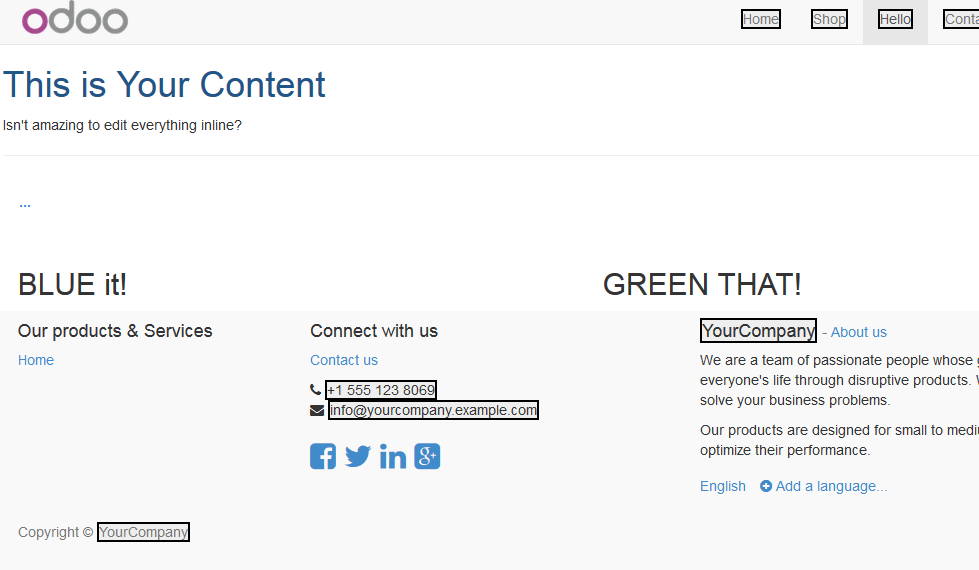
<field name="sequence" type="int">20</field>

</record>

</data>

</openerp>

Y el resultado es el siguiente:



* Editamos custom.less

C:\Users\cmejuto\Documents\DAM\DAM\_C\ServXestEmp\Tema2\GreenOdoo80win32\GreenOdoo-8.0-win32\source\addons\dummy\_theme\static\style

Añadimos el código que dice el manual y quedará así

h1 {

color: #215487;

}

span {

border: 2px solid black;

background-color: #eee;

}

@brand-primary: #1abc9c;

@color-blue: #3498db;

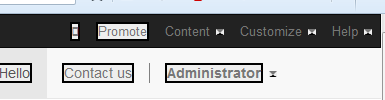
@color-green: #2ecc71;

.bg-blue { background: @color-blue; }

.bg-green { background: @color-green; }

.h2 { color: white; }

Al recargar cambiará el color de la barra y quedará así



ME QUEDO EN

### Build Your First Snippet

# Examen ODOO

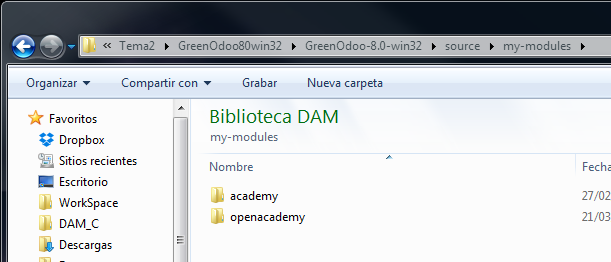
# Building a Module

<https://www.odoo.com/documentation/8.0/howtos/backend.html>

1. ejecutar start.bat

C:\Users\cmejuto\Documents\DAM\DAM\_C\ServXestEmp\Tema2\GreenOdoo80win32\GreenOdoo-8.0-win32

1. <http://localhost:8069/>
2. Puestos en esta carpeta



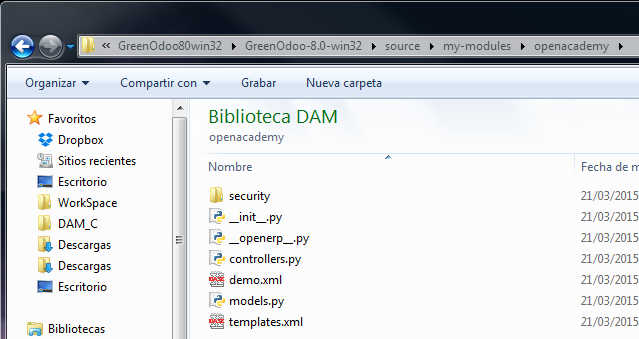
Ejecutamos un terminal y el siguiente commando



Según el manual

$ odoo.py scaffold <module name> <where to put it>

Esto nos creará la carpeta openacademy y el módulo.



Edito los siguientes archivos de my-modules/openacademy tal como indica el tutorial:

\_\_openerp\_\_.py

\_\_init\_\_.py

Controllers.py

Demo.xml

Models.py

Ir.model.access.csv

Templates.xml

Recargo los módulos e instalo el módulo openacademy

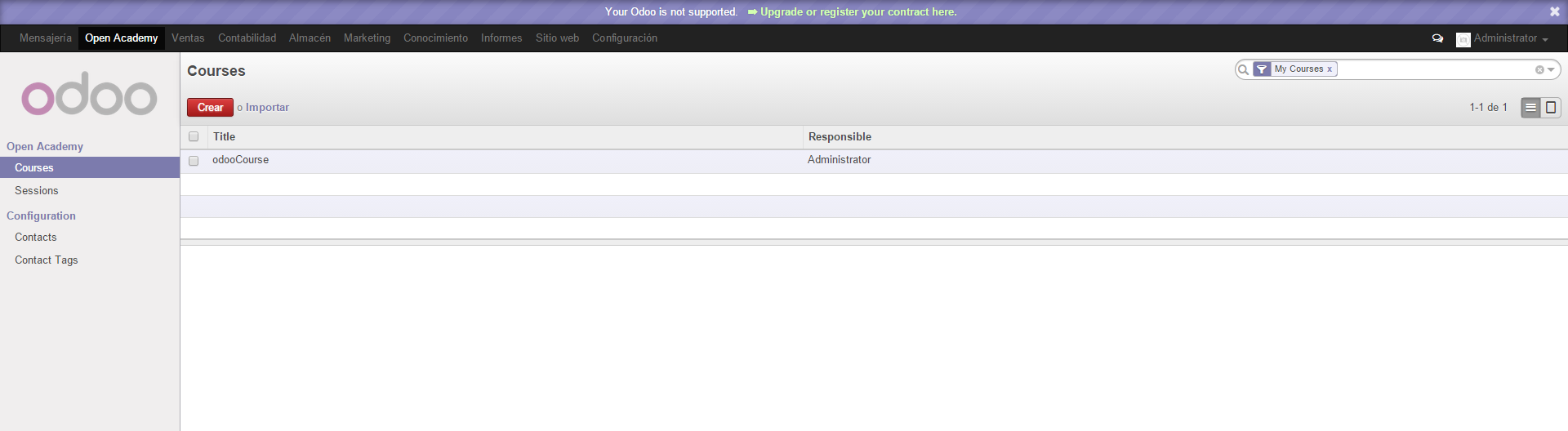
Voy a <http://localhost:8069/openacademy/openacademy/>

Pero



\*\*\* esto es porque no es un módulo del frontend, sino del **backend. No tenemos una vista web del módulo.**

Tenemos que tener esto al final del tutorial



### Continuo con

### Composition of a module

Un modulo tiene:

* Business objects
* Data files
* Web controllers

Seguimos los pasos del tutorial

### Data files

El fichero demo.xml queda así:

<openerp>

<data>

<record model="openacademy.course" id="course0">

<field name="name">Course 0</field>

<field name="description">Course 0's description</field>

</record>

<record model="openacademy.course" id="course1">

<field name="name">Course 1</field>

</record>

<record model="openacademy.course" id="course2">

<field name="name">Course 2</field>

<field name="description">Course 2's description</field>

</record>

</data>

</openerp>

### Actions and Menus

OJO!!!! El tutorial dice

**Danger**

The action must be declared before its corresponding menu in the XML file.

Data files are executed sequentially, the action’s id must be present in the database before the menu can be created.

**Tenemos que crear la carpta views y el archivo xml openacademy openacademy/views/openacademy.xml**

En \_\_openerp\_\_.py añadimos la siguiente línea (en rojo)

'data': [

# 'security/ir.model.access.csv',

'templates.xml',

**'views/openacademy.xml',**

],

# only loaded in demonstration mode

'demo': [

**Basic views**

Views / Openacademy.xml queda así

<?xml version="1.0" encoding="UTF-8"?>

<openerp>

<data>

<record model="ir.ui.view" id="course\_form\_view">

<field name="name">course.form</field>

<field name="model">openacademy.course</field>

<field name="arch" type="xml">

<form string="Course Form">

<sheet>

<group>

<field name="name"/>

</group>

**<notebook>**

**<page string="Description">**

**<field name="description"/>**

**</page>**

**<page string="About">This is an example of notebooks</page>**

**</notebook>**

</sheet>

</form>

</field>

</record>

<!-- window action -->

<!--

The following tag is an action definition for a "window action",

that is an action opening a view or a set of views

-->

<record model="ir.actions.act\_window" id="course\_list\_action">

<field name="name">Courses</field>

<field name="res\_model">openacademy.course</field>

<field name="view\_type">form</field>

<field name="view\_mode">tree,form</field>

<field name="help" type="html">

<p class="oe\_view\_nocontent\_create">Create the first course</p>

</field>

</record>

<!-- top level menu: no parent -->

<menuitem id="main\_openacademy\_menu" name="Open Academy"/>

<!-- A first level in the left side menu is needed

before using action= attribute -->

<menuitem id="openacademy\_menu" name="Open Academy" parent="main\_openacademy\_menu"/>

<!-- the following menuitem should appear \*after\*

its parent openacademy\_menu and \*after\* its

action course\_list\_action -->

<menuitem id="courses\_menu" name="Courses" parent="openacademy\_menu" action="course\_list\_action"/>

<!-- Full id location:

action="openacademy.course\_list\_action"

It is not required when it is the same module -->

</data>

</openerp>

### Search views

Añadimos el siguiente código a views/openacademy.xml

<?xml version="1.0" encoding="UTF-8"?>

<openerp>

<data>

<record model="ir.ui.view" id="course\_form\_view">

<field name="name">course.form</field>

<field name="model">openacademy.course</field>

<field name="arch" type="xml">

<form string="Course Form">

<sheet>

<group>

<field name="name"/>

</group>

<notebook>

<page string="Description">

<field name="description"/>

</page>

<page string="About">This is an example of notebooks</page>

</notebook>

</sheet>

</form>

</field>

</record>

**<record model="ir.ui.view" id="course\_search\_view">**

**<field name="name">course.search</field>**

**<field name="model">openacademy.course</field>**

**<field name="arch" type="xml">**

**<search>**

**<field name="name"/>**

**<field name="description"/>**

**</search>**

**</field>**

**</record>**

<!-- window action -->

<!--

The following tag is an action definition for a "window action",

that is an action opening a view or a set of views

-->

<record model="ir.actions.act\_window" id="course\_list\_action">

<field name="name">Courses</field>

<field name="res\_model">openacademy.course</field>

<field name="view\_type">form</field>

<field name="view\_mode">tree,form</field>

<field name="help" type="html">

<p class="oe\_view\_nocontent\_create">Create the first course</p>

</field>

</record>

<!-- top level menu: no parent -->

<menuitem id="main\_openacademy\_menu" name="Open Academy"/>

<!-- A first level in the left side menu is needed

before using action= attribute -->

<menuitem id="openacademy\_menu" name="Open Academy" parent="main\_openacademy\_menu"/>

<!-- the following menuitem should appear \*after\*

its parent openacademy\_menu and \*after\* its

action course\_list\_action -->

<menuitem id="courses\_menu" name="Courses" parent="openacademy\_menu" action="course\_list\_action"/>

<!-- Full id location:

action="openacademy.course\_list\_action"

It is not required when it is the same module -->

</data>

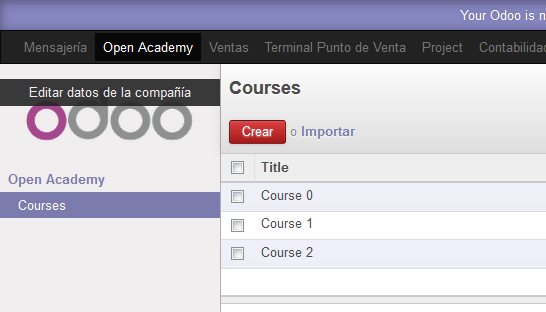
</openerp>

SI HAY ERRORES

Comprobar el log de errores

Openerp-server.log

El resultado es este



**Relations between models**

Using a many2one, modify the *Course* and *Session* models to reflect their relation with other models:

* A course has a *responsible* user; the value of that field is a record of the built-in model res.users.
* A session has an *instructor*; the value of that field is a record of the built-in model res.partner.
* A session is related to a *course*; the value of that field is a record of the model openacademy.course and is required.
* Adapt the views.

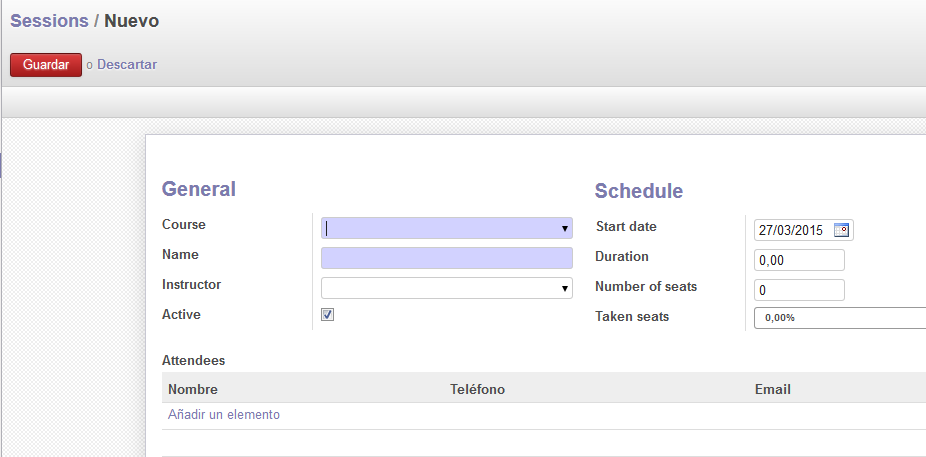
Nota: .users son como tablas de odoo

El resultado es





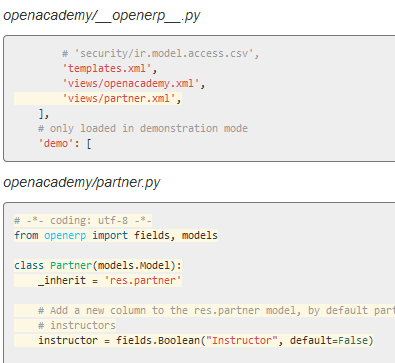




**Inheritance**

(herencia)

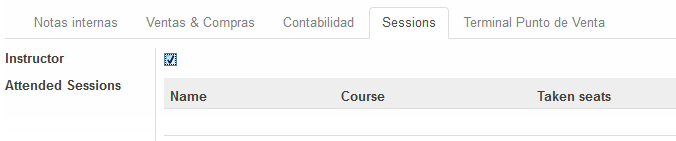
Es similar a java



Añadimos el campo partner (partner.xml)

Añadimos el campo instructor como falso en todos los partners.

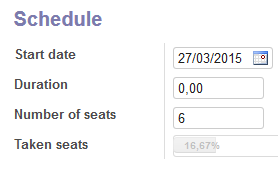
Resultado:



**Computed fields and default values**

En las vistas de cursos recogemos datos de la bd para presentarlos en la vista pero otros campos son calculados en tiempo real o tienen valor por defecto (asientos)

El computed field es por ejemplo el % de asientos ocupados

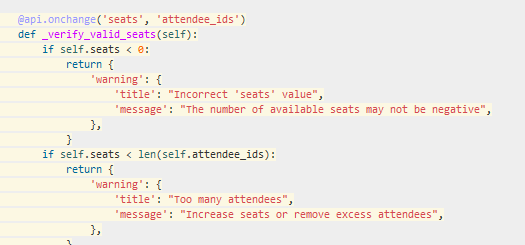


**Onchange**

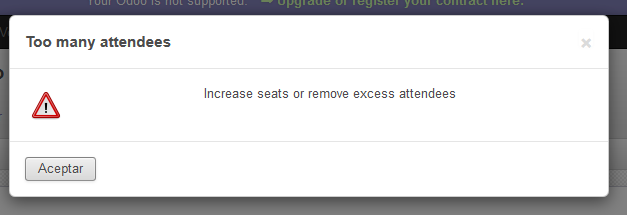
Evento que se lanza cuando hay un cambio en los datos del formulario

Ejemplo

Comprueba número de asientos



Resultado:

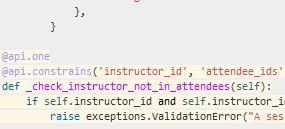


**Model constraints**

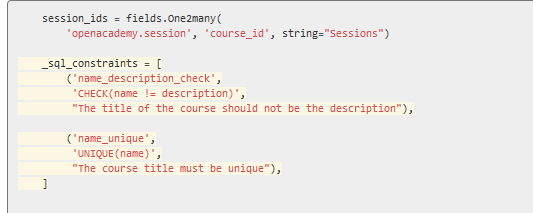
Restricciones

En los ejemplos lo hace con :

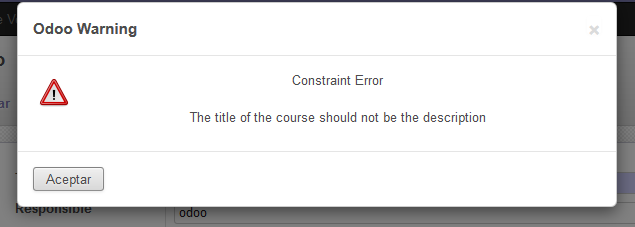
@apiconstraint



Y con sql



Resultado (de sql):



**Advanced Views**

Crea las vistas avanzadas de cada sesión

Resultado:



**Workflows**

Define el estado de la sesión cuando se crea

Add a state field to the *Session* model. It will be used to define a workflow-ish.

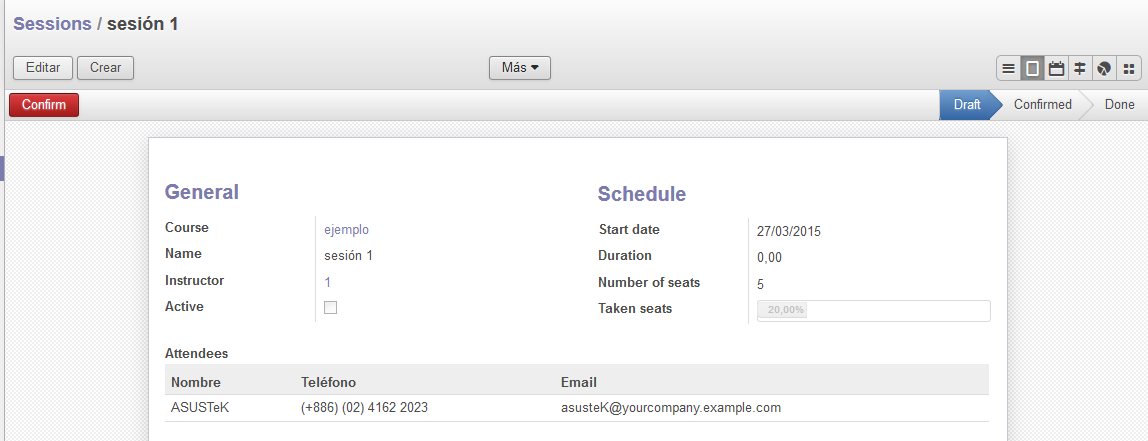
A sesion can have three possible states: Draft (default), Confirmed and Done.

In the session form, add a (read-only) field to visualize the state, and buttons to change it. The valid transitions are:

* Draft -> Confirmed
* Confirmed -> Draft
* Confirmed -> Done
* Done -> Draft

1. Add a new state field
2. Add state-transitioning methods, those can be called from view buttons to change the record’s state
3. And add the relevant buttons to the session’s form view

Resultado:



Si clicamos confirmado pasa a estado confirm.

En estado confirmado pasa a done

## Security

Limita el acceso a usuarios.