

# 1 Desenvolvemento de componentes

Índice

## 1.1 Sumario

- 1 Introducción a Python
- 2 Creación de módulos Odo

## 1.2 Introducción a Python

Para crear novos módulos en Odo utilizamos a linguaxe de programación Python.

Introdución a la programación con Python - Andrés Marzal e Isabel Gracia

The Python Language Reference

The Python Standard Library

- Practicamos creando un pequeno programa, *par.py*, que recolle números do teclado e responde se son pares ou impares:

```
GNU nano 2.2.6 File: par.py
#!/usr/bin/python
#coding: latin-1

print "Insire '0' para finalizar"
print "Insire un número:",
numero = int(raw_input())

while numero != 0:

    print "0 número", numero, "é",
    if (numero % 2 == 0):
        print "par."
    else:
        print "impar."

    print "Insire un número:",
    numero = int(raw_input())
```

```
root@BB-U3-0:/home/administrador# python par.py
Insire "0" para finalizar
Insire un número: 7
0 número 7 é impar.

Insire un número: 4
0 número 4 é par.

Insire un número: 0
root@BB-U3-0:/home/administrador#
```

- Creamos unha segunda versión cunha función propia:

```
GNU nano 2.2.6 File: par_f.py
#!/usr/bin/python
#coding: latin-1

def par (n):
    if (n % 2 == 0):
        resposta = 'par'
    else:
        resposta = 'impar'
    return resposta

print "Insire '0' para finalizar"
print "Insire un número:",
numero = int(raw_input())

while numero != 0:
    print "0 número", numero, "é", par(numero)
    print "Insire un número:",
    numero = int(raw_input())
```

```
root@BB-U3-0:~/home/administrador# python par_f.py
Insire "0" para finalizar
Insire un número: 99
0 número 99 é impar
Insire un número: 88
0 número 88 é par
Insire un número: 0
root@BB-U3-0:~/home/administrador#
```

- Para finalizar, creamos unha terceira versión formada por:
  - ◆ Un módulo de funcións propias, chamado *funcions\_novagalaxia.py* onde gardamos a función creada no punto anterior.
  - ◆ O programa *par\_final.py* que importa e utiliza esa función.

```
GNU nano 2.2.6 File: funcions_novagalaxia.py
#f:/usr/bin/python
#coding: latin-1

def par (n):
    if (n % 2 == 0):
        resposta = 'par'
    else:
        resposta = 'impar'
    return resposta
```

```
GNU nano 2.2.6 File: par_final.py
#f:/usr/bin/python
#coding: latin-1

from funcions_novagalaxia import par

print "Insire '0' para finalizar"
print "Insire un número:"
numero = int(raw_input())

while numero != 0:
    print "0 número", numero, "é", par(numero)
    print "Insire un número:"
    numero = int(raw_input())
```

### 1.3 Creación de módulos Odoo

Vamos crear un módulo novo: unha axenda telefónica.

- Movémonos ao directorio */usr/lib/python2.7/dist-packages/openerp/addons*, onde están todos os módulos da aplicación.
- Creamos o directorio *axenda*.
- Dentro del, creamos o ficheiro *\_\_init\_\_.py* que chama ao ficheiro principal do noso paquete: *axenda.py*. Grazas a *\_\_init\_\_.py*, o módulo é recoñecido como tal por Odoo.

```
GNU nano 2.2.6 File: __init__.py
import axenda
```

- Creamos o ficheiro *\_\_openerp\_\_.py* coa descrición do módulo (diccionario):

```
GNU nano 2.2.6 File: openerp__.py Modified
#f:/usr/bin/python
#coding: utf-8

{
    'name': 'axenda',
    'author': 'Barcos',
    'category': 'Uncategorized',
    'version': '1.0',
    'description': 'Módulo personalizado para xestionar unha axenda telefónica',
    'depends': ['base'],
    'init_xml': [],
    'update_xml': ['axenda_views.xml'],
    'active': False,
    'installable': True,
}
```

- Creamos o ficheiro *axenda.py* (modelo e controlador). Define a clase *axenda* que é unha táboa na BD con varias columnas:

```

GNU nano 2.2.6 File: axenda.py
# -*- coding: utf-8 -*-

from openerp.osv import osv, fields

class axenda(osv.osv):
    _name = 'axenda'
    _columns = {
        'name': fields.char('Nome', size=64, required=True),
        'telefono': fields.char('Teléfono', size=16, required=True),
        'sexo': fields.selection(('M', 'Home'), ('F', 'Muller')), 'Sexo'),
        'facturacion': fields.float('Facturación anual media'),
        'disponible': fields.boolean('Disponible')
    }

axenda()

```

- Unha forma de comprobar que non haxa erros nos ficheiros *\*.py*, é executándoos con *python*:

```

python __init__.py
python __openerp__.py
python axenda.py

```

- Creamos a vista en *axenda\_view.xml* (dunha forma parecida a como fixeramos en [Creación de vistas en Odoo](#)), formada por tres elementos:

- ♦ Unha vista de formulario e outra vista de árbore:

```

GNU nano 2.2.6 File: axenda_view.xml
<?xml version="1.0" encoding="utf-8" ?>
<openerp>
<data>

<record model="ir.ui.view" id="axenda_vista_formulario">
<field name="name">axenda_vista_formulario</field>
<field name="model">axenda</field>
<field name="type">form</field>
<field name="priority" eval="5" />
<field name="arch" type="xml">
<form string="Axenda - Vista Formulario">
<field name="name" select="1" string="Nome" />
<field name="telefono" select="1" string="Teléfono" />
<field name="sexo" select="1" string="Sexo" />
<field name="facturacion" select="1" string="Facturación anual" />
<field name="disponible" select="1" string="Disponible" />
</form>
</field>
</record>

<record model="ir.ui.view" id="axenda_vista_arbore">
<field name="name">axenda_vista_arbore</field>
<field name="model">axenda</field>
<field name="type">tree</field>
<field name="priority" eval="5" />
<field name="arch" type="xml">
<tree string="Axenda - Vista Arbore">
<field name="name" select="1" string="Nome" />
<field name="telefono" select="1" string="Teléfono" />
<field name="sexo" select="1" string="Sexo" />
<field name="facturacion" select="1" string="Facturación anual" />
<field name="disponible" select="1" string="Disponible" />
</tree>
</field>
</record>

```

- ♦ Accións. Ao facer clic nunha opción do menú, a acción vai abrir a vista correspondente):

```

GNU nano 2.2.6 File: axenda_view.xml
<record model="ir.actions.act_window" id="action_axenda_form">
<field name="name">Información Axenda (Formulario)</field>
<field name="type">ir.actions.act_window</field>
<field name="res_model">axenda</field>
<field name="view_type">form</field>
<field name="view_mode">tree,form</field>
</record>

<record model="ir.actions.act_window" id="action_axenda_tree">
<field name="name">Información Axenda (Arbore)</field>
<field name="type">ir.actions.act_window</field>
<field name="res_model">axenda</field>
<field name="view_type">tree</field>
<field name="view_mode">tree,form</field>
</record>

```

- ◆ Menú, submenú e opcións:

```

GNU nano 2.2.6 File: axenda.view.xml
<menutitem name="NovasGalaxia" id="menu_novasgalaxia" />
<menutitem name="Axenda" id="menu_axenda" parent="axenda.menu_novasgalaxia" />

<ver axenda formulario" id="ver_axenda_formulario"
parent="axenda.menu_axenda"
action="action_axenda_form" />

<ver axenda arbore" id="ver_axenda_arbore"
parent="axenda.menu_axenda"
action="action_axenda_tree" />

</data>
</openERP>

```

- En Odo: actualizamos a lista de módulos, facemos clic en *Módulos locais*, procuramos o módulo *axenda* e o instalamos.



- Se houber erros no ficheiro *xml*, saberémolo neste momento (a mensaxe de erro é moi longa pero temos que fixarnos na última liña). Neste exemplo especificouse mal a codificación *utf-8*:

```

OpenERP
File "/usr/lib/python2.7/openERP/modules/loading.py", line 114, in load_module
loader, processed = load_module_graph(ogr, graph, progressidiot, report=report, skip_module="ic
File "/usr/lib/python2.7/openERP/modules/loading.py", line 107, in load_module_graph
load_update_xml(module_name, idref, mode)
File "/usr/lib/python2.7/openERP/modules/loading.py", line 74, in <lambda>
load_update_xml = lambda *args: load_data(ogr, *args, kind='update_xml')
File "/usr/lib/python2.7/openERP/modules/loading.py", line 124, in load_data
tools.convert_xml_report(ogr, module_name, fp, idref, mode, soupdate, report)
File "/usr/lib/python2.7/openERP/tools/convert.py", line 94, in convert_xml_report
doc = etree.parse(xmlfile)
File "lxml.etree.pyx", line 2933, in lxml.etree.parse (src/lxml/lxml.etree.c:54204)
File "parser.pyx", line 1555, in lxml.etree._parseDocument (src/lxml/lxml.etree.c:82511)
File "parser.pyx", line 1359, in lxml.etree._parseFilelikeDocument (src/lxml/lxml.etree.c:18232)
File "parser.pyx", line 1469, in lxml.etree._parseDocFromFilelike (src/lxml/lxml.etree.c:16481)
File "parser.pyx", line 1024, in lxml.etree._BaseParser._parseDocFromFilelike (src/lxml/lxml.et
File "parser.pyx", line 949, in lxml.etree._ParserContext._handleParseResultDoc (src/lxml/lxml
File "parser.pyx", line 450, in lxml.etree._handleParseResult (src/lxml/lxml.etree.c:75263)
File "parser.pyx", line 390, in lxml.etree._raiseParseError (src/lxml/lxml.etree.c:74694)
XMLSyntaxError: Unsupported encoding utf-8, line 1, column 34

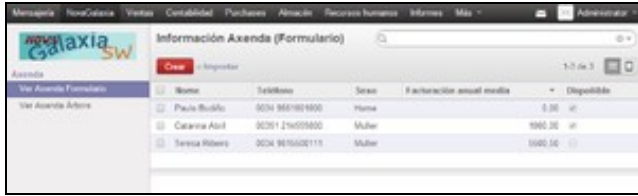
```

- Outro erro típico: "Ha ocurrido un error mientras se validaban los campos arch: Invalid XML for View Architecture!". A causa é esquecer inserir espazos en branco antes das "/" de cerce.

- Se todo vai ben, deber aparecer o novo menú *Axenda* coas dúas vistas deseñadas. Inserimos datos na axenda:



- E para finalizar, comprobamos o resultado:



Galaxia SW

Información Axenda (Formulario)

Crear Imprimir 1 de 3

	Nome	Telefono	Sexo	Factorización anual media	Disponibilidade
Ver Axenda Francisco	Paulo Budo	0034 967401400	Home	0.00	✖
Ver Axenda Añore	Catarina Alari	00351 216050400	Mulher	1000.00	✖
	Teresa Ribeiro	0034 967600111	Mulher	1000.00	✖

Fontes: [1] [2] [3]