

Desenvolvimento de componentes

Índice

Sumario

- 1 Introducción a Python
- 2 Creación de módulos Odo

Introducción a Python

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

Introducción 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
#?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
#?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())
```

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
#?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 = {
        'nome': fields.char('Nome', size=64, required=True),
        'telefono': fields.char('Telefono', 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="nome" select="1" string="Nome" />
<field name="telefono" select="1" string="Telefono" />
<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="nome" select="1" string="Nome" />
<field name="telefono" select="1" string="Telefono" />
<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>

```

- ◆ Acci3ns. Ao facer clic nunha opci3n do men3, a acci3n vai abrir a vista correspondente):

```

GNU nano 2.2.6 File: axenda.view.xml
<record name="ir.action.act_window" id="action_axenda_form">
  <field name="name">Informaci3n Axenda (Formulario)</field>
  <field name="type">ir.action.act_window</field>
  <field name="res_model">axenda</field>
  <field name="view_type">form</field>
  <field name="view_mode">tree,form</field>
</record>

<record name="ir.action.act_window" id="action_axenda_tree">
  <field name="name">Informaci3n Axenda (Arbore)</field>
  <field name="type">ir.action.act_window</field>
  <field name="res_model">axenda</field>
  <field name="view_type">tree</field>
  <field name="view_mode">tree,form</field>
</record>

```

- ◆ Men3, submen3 e opci3ns:

```

GNU nano 2.2.6 File: axenda.view.xml
<menuitem name="NovaGalaxia" id="menu_novagalaxia" />
<menuitem name="axenda" id="menu_axenda" parent="axenda.menu_novagalaxia" />

<menuitem name="Ver Axenda Formulario" id="ver_axenda_formulario"
  parent="axenda.menu_axenda"
  action="action_axenda_form" />

<menuitem name="Ver Axenda Arbore" id="ver_axenda_arbore"
  parent="axenda.menu_axenda"
  action="action_axenda_tree" />

</data>
</serverp>

```

- En Odo: actualizamos a lista de m3dulos, facemos clic en *M3dulos locais*, procuramos o m3dulo *axenda* e o instalamos.



- Se houber erros no ficheiro *xml*, saber3molo neste momento (a mensaxe de erro 3 moi longa pero temos que fixarnos na 3ltima li3a). Neste exemplo especificouse mal a codificaci3n *utf-8*:

```

OpenERP
File "/usr/lib/python2.7/openerp/modules/loading.py", line 214, in load_marked_module
  loaded, processed = load_module_graph(ctx, graph, progressioct, report=report, skip_modules=sk
File "/usr/lib/python2.7/openerp/modules/loading.py", line 187, 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(ctx, *args, kind='update_xml')
File "/usr/lib/python2.7/openerp/modules/loading.py", line 124, in _load_data
  tools.convert_xml_report(ctx, module_name, fp, idref, mode, load_update, report)
File "/usr/lib/python2.7/openerp/tools/convert.py", line 941, in convert_xml_import
  doc = etree.parse(xmlfile)
File "lxml.etree.pyx", line 2955, in lxml.etree.parse (src/lxml/lxml.etree.c:56204)
File "parser.pxi", line 1355, in lxml.etree._parseDocument (src/lxml/lxml.etree.c:82511)
File "parser.pxi", line 1385, in lxml.etree._parseFilelikeDocument (src/lxml/lxml.etree.c:82832)
File "parser.pxi", line 1468, in lxml.etree._parseDocFromFilelike (src/lxml/lxml.etree.c:81688)
File "parser.pxi", line 1024, in lxml.etree._BaseParser._parseDocFromFilelike (src/lxml/lxml.et
File "parser.pxi", line 349, in lxml.etree._ParserContext._handleParseResultDoc (src/lxml/lxml
File "parser.pxi", line 450, in lxml.etree._handleParseResult (src/lxml/lxml.etree.c:78343)
File "parser.pxi", line 936, 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 cierre.

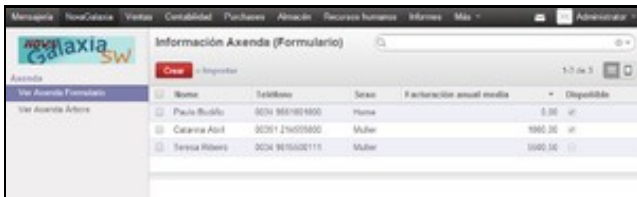
- Se todo vai ben, deber aparecer o novo menú *Axenda* coas dúas vistas deseñadas. Inserimos datos na axenda:



The screenshot shows a web browser window with the Galaxia SW logo. The page title is 'Información... / Nuevo'. There are two tabs: 'Ver Axenda Formulario' (selected) and 'Ver Axenda Actual'. The form contains the following fields:

Nome	Catalina Abel	Telefono	9851 214558802
Sexo	Mulher	Factorización anual media	1000
Disponíble	<input checked="" type="checkbox"/>		

- E para finalizar, comprobamos o resultado:



The screenshot shows the 'Información Axenda (Formulario)' view. It features a table with the following data:

Nome	Telefono	Sexo	Factorización anual media	Disponíble
Paula Budo	9034 961401600	Home	6.00	<input checked="" type="checkbox"/>
Catalina Abel	9034 214558802	Mulher	1000.00	<input checked="" type="checkbox"/>
Teresa Ribera	9034 9816402111	Mulher	1000.00	<input checked="" type="checkbox"/>

Fontes: [\[1\]](#) [\[2\]](#) [\[3\]](#)