

1 Desenvolvimento de compoñentes

Anterior: [Implantación de sistemas ERP/CRM](#)

1.1 Sumario

- [1 Introducción a Python](#)
- [2 Creación de módulos OpenERP](#)

1.2 Introducción a Python

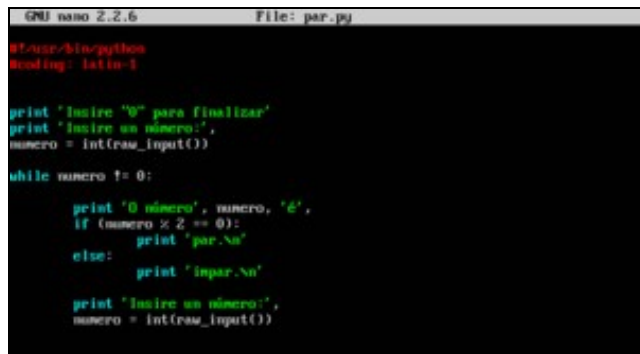
Para crear novos módulos en OpenERP utilizamos a linguaxe de programación [Python](#).

[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 "O número", numero, "é",
    if (numero % 2 == 0):
        print "par.\n"
    else:
        print "impar.\n"
    print "Insire un número:",
    numero = int(raw_input())
```

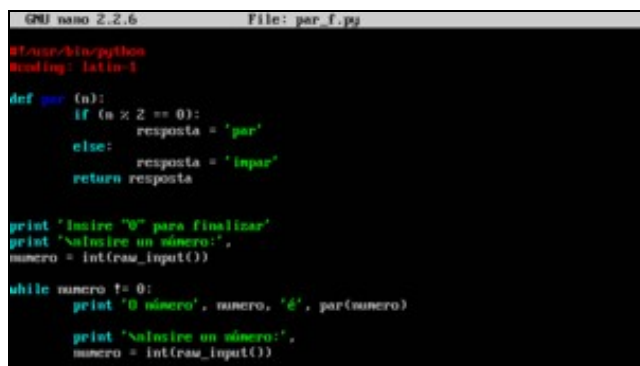


```
root@B-U3-0:/home/administrador# python par.py
Insire "0" para finalizar
Insire un número: 7
O número 7 é impar.

Insire un número: 4
O número 4 é par.

Insire un número: 0
root@B-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 "O número", numero, "é", par(numero)
    print "Insire un número:",
    numero = int(raw_input())
```

```
root@08-0:/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@08-0:/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())
```

1.3 Creación de módulos OpenERP

Vamos crear un módulo novo: unha axenda telefónica.

- Movémonos ao directorio */usr/lib/pymodules/python2.7/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 OpenERP.

```
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': 'Marcos',
    '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),
        'sexu': 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 OpenERP](#)), 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="sexu" select="1" string="Sexo" />
            <field name="facturacion" select="1" string="Facturación anual media" />
            <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="sexu" select="1" string="Sexo" />
            <field name="facturacion" select="1" string="Facturación anual media" />
            <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.action.act_window" id="action_axenda_form">
  <field name="name">Información 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 model="ir.action.act_window" id="action_axenda_tree">
  <field name="name">Información 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>
```

- ◆ Menú, submenú e opcións:

```
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>
</openrp>
```

- En OpenERP: actualizamos a lista de módulos, facemos clic en *Módulos locales*, 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/openrp/modules/loading.py", line 254, in load_marked_module
  loaded, processed = load_module_graph(cursor, graph, progressdot, report=report, skip_modules=1)
File "/usr/lib/python2.7/openrp/modules/loading.py", line 187, in load_module_graph
  load_update_xml(module_name, idref, mode)
File "/usr/lib/python2.7/openrp/modules/loading.py", line 74, in <lambda>
  load_update_xml = lambda *args: load_data(cursor, *args, kind='update_xml')
File "/usr/lib/python2.7/openrp/modules/loading.py", line 124, in _load_data
  tools.convert_xml_import(cursor, module_name, ty, idref, mode, loadupdate, report)
File "/usr/lib/python2.7/openrp/tools/convert.py", line 941, in convert_xml_import
  doc = etree.parse(xmlfile)
File "lxml.etree.pyx", line 2953, in lxml.etree.parse (src/lxml/lxml.etree.c:56204)
File "parser.pyx", line 1355, in lxml.etree._parseDocument (src/lxml/lxml.etree.c:52511)
File "parser.pyx", line 1355, in lxml.etree._parseFilelikeDocument (src/lxml/lxml.etree.c:52532)
File "parser.pyx", line 1468, in lxml.etree._parseDocFromFilelike (src/lxml/lxml.etree.c:51688)
File "parser.pyx", line 1024, in lxml.etree._BaseParser._parseDocFromFilelike (src/lxml/lxml.etree.c:4949)
File "parser.pyx", line 949, in lxml.etree._ParserContext._handleParseResultDoc (src/lxml/lxml.etree.c:485)
File "parser.pyx", line 936, in lxml.etree._handleParseResult (src/lxml/lxml.etree.c:475343)
File "parser.pyx", line 936, in lxml.etree._raiseParseError (src/lxml/lxml.etree.c:47494)
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:

Formulario de creación de novo elemento na Axenda:

Nome	Catalina Abel	Telefone	90361 214000000
Sexo	Mulher	Fatorización anual media	1000
Disponíbel	☒		

- E para finalizar, comprobamos o resultado:

Resultado da consulta:

	Nome	Telefone	Sexo	Fatorización anual media	Disponíbel
☒	Paula Budillo	90361 9851001600	Femea	0.00	☒
☒	Catalina Abel	90361 214000000	Mulher	1000.00	☒
☒	Teresa Ribeiro	90361 9816002111	Mulher	1000.00	☒

Fontes: [1] [2] [3]