

Gnumeric: la hoja de cálculo Python

GNUMEROS



Gnumeric es un software bastante desconocido, que en un mundo de suites ofimáticas se establece como una hoja de cálculo en solitario. Pero aún así esconde muchos secretos que no dejarán de sorprendernos. Entre ellos, la posibilidad de usar Python como lenguaje de creación de macros y funciones.

POR JOSÉ MARÍA RUÍZ

Si existe un programa que aleja al mundo Linux de la pequeña y mediana empresa es sin duda Excel. Excel es un estándar inamovible. Miles, si no cientos de miles, de personas lo usan diariamente en su trabajo o aprenden a usarlo en las universidades más prestigiosas. Y si bien el proyecto OpenOffice ha logrado hasta cierto punto compatibilidad con Excel, el proyecto Gnumeric decidió hace muchos años tomar otros derroteros.

Gnumeric es *mejor* que Excel para determinadas actividades (ver [1]). Créetelo: el destino del mundo financiero descansa en una hoja de cálculo, Excel, que ¡ni siquiera implementa bien la mayoría de las funciones estadísticas que trae de serie!

Aunque Gnumeric mantiene compatibilidad con Excel, ha decidido no seguir el camino de Excel. Una hoja de cálculo puede resultar muy útil, pero con VisualBasic es un arma muy poderosa. El proyecto Gnumeric ha decidido no emplear VisualBasic... ¡sino Python y Perl!

Python y Gnumeric

Gnumeric incorpora un mecanismo que permite el uso de extensiones, que ellos llaman *complementos*. Entre estas extensiones está Python. Para activarla deberemos ir al menú *Herramientas, Complemento* y buscar el complemento *Cargador de complementos de Python* (también veremos una opción parecida para Perl).

Una vez tengamos activado el complemento aparecerá una nueva opción en el menú *Herramientas: Python Console*, (ver

Figura 1) que esperemos algún día ponga *Consola Python*. Lo importante es que una vez que hagamos click con el ratón en esa entrada del menú aparecerá la consola, ver Figura 2.

La consola Python es básicamente un intérprete Python normal y corriente, con la ventaja de tener acceso a la librería Gnumeric. Una vez abierta podemos comenzar a trabajar con ella:

```
>>> import Gnumeric
>>> dir(Gnumeric)
['Boolean', 'CellPos', 'FALSE', 'GnmStyle', 'GnumericError',...]
```

El comando *dir()* nos da una lista de todos los métodos a los que responde un objeto. En este caso aparecen tantos métodos que no es conveniente reflejarlos

Listado 1: linuxmagazine.py

```
01 import Gnumeric
02
03 def hola (col,filas,texto):
04     w =
05     Gnumeric.workbooks()[0]
06     s = w.sheets()[0]
07     c = s.cell_fetch(col,filas)
08     c.set_text(texto)
09
10 funciones_functions = {
11     'hola':
12     (('iis','col,filas,texto',hola)
13 }
```

Listado 2: plugin.xml 1

```
01 <?xml version="1.0"?>
02 <plugin
03     id="Gnumeric_linuxmagazine"
04     <information>
05         <name>Linux Magazine</name>
06         <description>Ejemplos.</description>
07         </information>
08         <loader
09             type="Gnumeric_PythonLoader:python"
10             <attribute
11                 name="module_name"
12                 value="linuxmagazine"/>
13         </loader>
14         <services>
15             <service
16                 type="function_group"
17                 id="funciones">
18                 <description>Ejemplos.</description>
19                 <category>Local
20                 Python</category>
21                 <functions>
22                 <function name="hola" />
23                 </functions>
24             </service>
25         </services>
26     </plugin>
```

debido al espacio. Nos interesa el método `workbooks()`, el cual, como otros similares, devuelve una lista de objetos. En este caso, estos objetos serán los libros que Gnumeric tiene abiertos:

```
>>> w = Gnumeric.
workbooks()[0]
>>> dir(w)
['gui_add', 'sheet_add',
'sheets']
```

volvemos a hacer lo mismo con el `workbook`, `sheets()` nos da una lista de hojas del libro:

```
>>> s = w.sheets()[0]
>>> dir(s)
['cell_fetch', 'get_extent',
'get_name_unquoted', 'rename',
'style_apply_range',
'style_get', 'style_set_pos',
'style_set_range']
```

Bien, ya hemos abierto la muñeca matroska hasta un nivel en el que podremos trabajar. De todos estos métodos nos interesa `cell_fetch()`. Con él podemos acceder a la celda (0,0) pero ¿no se direccionaban las celdas en base a letras y números? Por desgracia no es así desde la API de Python. El primer número se refiere a la columna; como no se usan letras, entonces se usan números, empezando en el 0, y por tanto hemos accedido a la columna A. Las filas se indican con el segundo número, pero que comienza en 0 que indica la fila 1.

```
>>> c = s.cell_fetch(0,0)
>>> dir(c)
['get_entered_text',
'get_rendered_text',
'get_style', 'get_value',
'get_value_as_string',
'set_text']
```

Muy bien, tenemos una celda y podemos asignarle un valor:

```
>>> c.set_text("Hola Linux
Magazine")
```

La cadena de texto aparecerá en la celda correspondiente a la primera columna y la primera celda. Ya tenemos nuestro *Hola Mundo*. La consola Python puede ser útil, pero en cuanto la cerremos perdemos todo nuestro trabajo. Resultaría más interesante poder cargar funciones Python o, y esto sería genial, hacer que éstas se comportasen como el resto de las funciones de Gnumeric, integrándose con ellas.

Pues Gnumeric cumple este deseo. Veamos cómo podemos crear nuestras propias funciones Gnumeric con Python.

Funciones Python

Gnumeric, dentro de esa moda que algunos esperamos que pase pronto, emplea XML para la definición de plugins. Se crea un fichero XML donde se da a conocer a Gnumeric información sobre el complemento. Un complemento puede ofrecer un número ilimitado de funciones, y estas funciones pueden cambiar con el idioma. Esta es una costumbre de

las hojas de cálculo, puesto que desde muy pronto casi todas decidieron ofrecer las mismas funciones, pero con nombres acordes a cada idioma. También es posible incluir documentación que aparecerá en la descripción de la función. Tanto los ficheros Python como los ficheros XML se deben almacenar en el directorio:

```
~/gnumeric/<versión de
Gnumeric>/plugins/
```

Donde *versión de Gnumeric* sería del tipo 1.7.12. Este directorio es individual para cada usuario, pero es posible hacer que las funciones estén disponibles para todos los usuarios empleando el directorio general en el que Gnumeric almacena sus recursos. Dependiendo de la distribución de Linux que empleemos este directorio se encontrará en diferentes sitios, pero normalmente estará en algún lugar dentro de `/usr/share` o `/usr/lib`.

Vamos a comenzar con una función simple. Por ejemplo, podríamos encapsular una función a la que demos una columna, una fila y un texto, y ponga el texto en la celda referenciada por la columna y la fila (ver Listado 1).

Dentro del directorio de plugins debemos crear un directorio para nuestro plugin que llamaremos *linuxmagazine*. En ese directorio deben ir todos los ficheros necesarios.

En nuestro caso vamos a hacer el mínimo trabajo posible y sólo crearemos dos ficheros: uno llamado *plugin.xml*, y otro con el contenido del Listado 1 llamado *linuxmagazine.py*.

Listado 3: estadisticas_log.py

```
01 import re
02 import Gnumeric
03
04 def estadistica_log (fichero):
05     f = open(fichero)
06
07     datos = {}
08
09     for linea in f:
10         encontrado = re.findall("(^[^
11         ip,fecha = encontrado[0]
12
13         if datos.has_key(fecha) :
14             d_ip = datos[fecha]
15             if d_ip.has_key(ip):
16                 d_ip[ip] += 1
17             else :
18                 d_ip[ip] = 1
19
20         datos[fecha] = d_ip
21     else :
22         datos[fecha]=dict([(ip,1)])
23
24     col = 0
25     fila = 0
26
27     print datos
28
29     llaves = datos.keys()
30
31     w = Gnumeric.workbooks()[0]
32     s = w.sheets()[0]
33
34     for fecha in llaves:
35
36         c = s.cell_fetch(col,fila)
37         c.set_text(str(fecha))
38
39         fila += 1
40         for ip in
41             datos[fecha].keys():
42             c = s.cell_fetch(col,fila)
43             c.set_text( str(ip) )
44             c = s.cell_fetch(col+1,fila)
45             c.set_text(str
46                 (datos[fecha][ip]))
47             fila += 1
48
49     funciones_functions = {
50         'estadistica_log':
51             estadistica_log
52     }
```

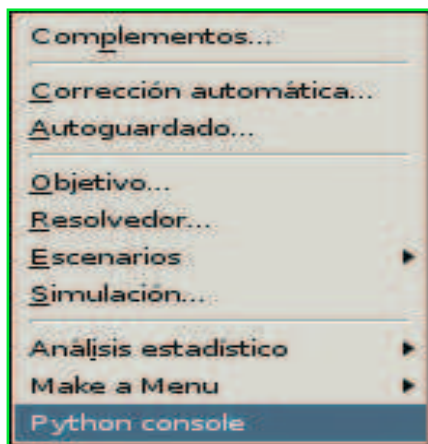


Figura 1: Menú.

El fichero *plugin.xml* describe el contenido de *linuxmagazine.py*, y podemos verlo en el Listado 2. Es bastante complejo, se compone de diferentes secciones: *information*, *loader* y *services*. Las diferentes funciones del Listado 1 se agrupan en un módulo. Este módulo aparecerá en la ventana de carga de complementos que vimos anteriormente. En la sección *information* es donde indicamos el nombre del módulo y damos una breve descripción de qué hace el mismo.

loader indica a Gnumeric qué tipo de ficheros vamos a cargar. Existen diferentes cargadores, es posible escribir plugins en Perl o en C, por lo que debemos indicarle a Gnumeric qué es lo que queremos cargar y cómo tiene que hacerlo. Aquí también asignamos un nombre de módulo, pero éste es para uso interno de Gnumeric.

Por último, *services* le indica a Gnumeric qué funciones debe cargar. Normalmente, el fichero Python contendrá muchas funciones, y no queremos que todas sean visibles desde Gnumeric. Para evitar el acceso

a las que queramos que permanezcan ocultas sólo tenemos que crear un diccionario, en nuestro caso *funciones*, en el que indicamos el nombre de las funciones que

estarán disponibles en Gnumeric y con qué nombre queremos registrarlas (ver final de Listado 1). En *services* indicamos el nombre del diccionario de funciones a exportar dentro del campo *id* de la etiqueta *services* y posteriormente indicamos, de nuevo, las funciones de las que vamos a disponer, sólo que ahora podremos documentarlas.

Extrayendo Estadísticas

Como ejemplo práctico de la creación y empleo de funciones Python vamos a crear una función que analice el log de Apache y nos diga cuántas veces nos ha visitado cada IP en un mismo día, ver código de Listado 3. En el Listado 4 está el código del fichero *plugin.xml* que acompaña al fichero Python.

Para ello vamos a crear una función que acepta 3 parámetros. El primero será la columna y el segundo la fila, a partir de las cuales se escribirán los datos. El tercer parámetro será la ruta del fichero que vamos a analizar. Generalmente será */var/log/httpd-access.log*, pero como ese suele ser muy grande y nuestra hoja de cálculo sólo tiene 65535 filas, he supuesto que lo mismo sería interesante trocear el fichero en otros más pequeños, y por ello necesitamos el tercer parámetro.

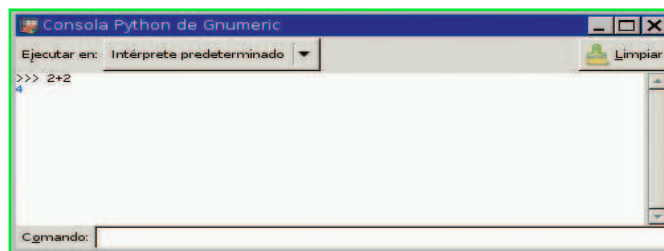


Figura 2: Consola Python.

El formato será muy simple: por cada día se escribirá la fecha en particular y debajo de ella las IPs en una columna y el número de páginas pedidas la siguiente.

¿Es muy simple? No tiene que ser más complicado para que sea útil, una vez que los datos estén en Gnumeric será muy sencillo analizarlos y crear gráficas con ellos. Mi experiencia me dice que lo más útil en el mundo de las hojas de cálculo es la capacidad de importar datos a partir de casi cualquier cosa. Es algo que hago constantemente. Lo mismo un cliente te manda una relación en formato TXT de las compras que ha realizado o de clientes a los que hay que llamar. Con Python podemos analizar el fichero, extraer los datos e integrarlos en una hoja de cálculo de Gnumeric sin problemas.

Conclusión

Las hojas de cálculo son la sangre de las empresas, y negarlo puede alejar a Linux del mundo de las pequeñas, medianas y grandes empresas. La hoja de cálculo es tan ubicua que para muchos sería impensable su trabajo sin ella. Por eso es tan importante impulsar Gnumeric, ya que apenas tiene apoyos... excepto de los propios interesados. OpenOffice tiene a Sun como respaldo, pero Gnumeric sólo tiene un grupo de hackers que tiene claro lo que es una buena hoja de cálculo.

Python puede ser una herramienta clave para que Gnumeric se haga cada vez más conocido. Trabajar con OpenOffice y Python es muy engorroso, como ya vimos en otros artículos, así que personalmente prefiero trabajar con Gnumeric, que me permite ser productivo y obtener resultados increíbles con muy poco esfuerzo. ■

Listado 4: plugin.xml 2

```

01 <?xml version="1.0"?>
02 <plugin value="estadistica_log"/>
03 <information id="Gnumeric_linuxmagazine">
04 <name>Linux Magazine
05 </name>
06 <description>Ejemplos.</description>
07 <loader type="Gnumeric_PythonLoader:python">
08 <attribute name="module_name"
09 </loader>
10 <services>
11 <service
12 <category>Local
13 <functions>
14 <function
15 </functions>
16 </service>
17 </services>
18 </plugin>

```

RECURSOS

[1] Proyecto Gnumeric: <http://www.gnome.org/projects/gnumeric/>