Generación de informes profesionales desde Python

REPORTLAB

Hoy en día se hace imprescindible disponer de herramientas que permitan generar informes en PDF de alta calidad rápida y dinámicamente. Existen diferentes herramientas para esta finalidad, entre ellas cabe destacar ReportLab, biblioteca gratuita que permite crear documentos PDF empleando como lenguaje de programación Python. **POR ANA M. FERREIRO Y JOSE A. GARCÍA**

La biblioteca ReportLab crea directamente documentos PDF basándose en comandos gráficos y sin pasos intermedios, generando informes en un tiempo extremadamente rápido y siendo de gran utilidad en los siguientes contextos: generación dinámica de PDFs en aplicaciones web (empleado con Zope), generación de informes y publicación de datos almacenados en bases de datos, embebiendo el motor de impresión en aplicaciones para conseguir la generación de informes a medida, etc.

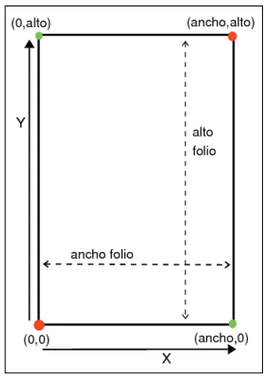

Figura 1: Coordenadas cartesianas de una hoja.

Primeros pasos

Lo primero es tener instalados Python y ReportLab para realizar todas las pruebas que van surgiendo y las que se nos ocurran. En [1] se detallan los pasos que hay que seguir para instalar y configurar ReportLab.

El paquete *pdfgen* es el nivel más bajo para generar documentos PDF, que se basa esencialmente en una secuencia de instrucciones para "dibujar" cada página del documento. El objeto que proporciona las operaciones de dibujo es el *Canvas*. El *Canvas* mide igual que una hoja de papel blanco, con puntos sobre la misma identificados mediante coordenadas cartesianas *(X,Y)*, que por defecto tienen el origen (0,0) en la esquina inferior izquierda de la página. La coordenada *X* va hacia la derecha y la coordenada *Y* avanza hacia arriba (ver Figura 1).

Para crear nuestro primer PDF basta escribir en un fichero, que podemos llamar *ejemplo1.py*, las siguientes líneas de código:

```
from reportlab.pdfgen➥
import canvas
c=canvas.Canvas("primer.pdf")
c.drawString(50,500, " Mi➥
PRIMER PDF")
c.drawString(250,300,➥
"Coordenada=(250,300) ")
c.drawString(350,200,➥
"(350, 10)")
c.drawString(150,400,➥
"Aprendiendo REPORTLAB")
c.showPage()
c.save()
```

Probamos el programa y vemos que en el mismo directorio ya se ha creado un fichero llamado *primer.pdf*, análogo al que se muestra en la Figura 2, sin necesidad de realizar ningún otro paso intermedio. Mediante la línea *from reportlab.pdfgen import canvas* importamos *Canvas*, utilizado para dibujar en el PDF. El comando

canvas.Canvas(path_fichero) permite indicar el nombre con el que se guardará el PDF. El método *drawString(x,y,cadena_texto)* empieza a escribir el texto en la coordenada (x,y) (se puede probar a cambiar las diferentes coordenadas). El método *showPage()* crea la página actual del documento. Finalmente, *save()* guarda el fichero en el path indicado.

En el ejemplo previo hemos creado un PDF sin especificar el tamaño del documento, si queremos fijar el tamaño de la hoja (A4, letter, A5, etc.) bastaría indicarlo en el Canvas mediante:

```
from reportlab.lib.➥
pagesizes import letter,A4,A5,➥
A3
c=canvas.Canvas("primer.pdf",➥
pagesize=letter)
```

Listado 1: ejemplo2.py

```
01 from reportlab.pdfgen import
   canvas
02
   c=canvas.Canvas("canvas_draw.p
   df")
03 c.setFont("Helvetica",24)
04 c.line(50,50,50,350)
05 c.line(50,50,350,50)
06 c.setStrokeColorRGB(1,1,0.0)
07 c.setFillColorRGB(0,0.0,0.5)
08
   c.roundRect(75,75,275,275,20,s
   troke=0,
   fill=1)
09 c.setFillColorRGB(0.8,0.,0.2)
10 c.circle(205,205,100,stro-
   ke=1,fill=1)
11
   c.setFillColorRGB(0.75,0.75,0.
   )
12
   c.drawString(125,80,"Cuadrado"
   )
13 c.setFillColorRGB(0,1,0.2)
14
   c.drawString(155,200,"Circulo"
   )
15 c.setStrokeColorRGB(1,0,0.0)
16
   c.ellipse(75,450,350,335,fill=
   0)
17 c.setFillColorRGB(0,0,0.5)
18 c.drawString(150,375,"Elipse")
19 c.showPage()
20 c.save()
```

El tamaño de las hojas se importa mediante *from reportlab.lib.pagesizes import letter,A4,A5,A*, y se especifica en el *Canvas* con la propiedad *pagesize*. Muchas veces querremos adaptar el dibujo a las dimensiones de la hoja, por lo que necesitamos conocer el ancho y el alto de la misma: *ancho = tipo_hoja[0]* y *alto = tipo_hoja[1]*; donde *tipo_hoja* puede ser *letter, A4, A5* , etc.

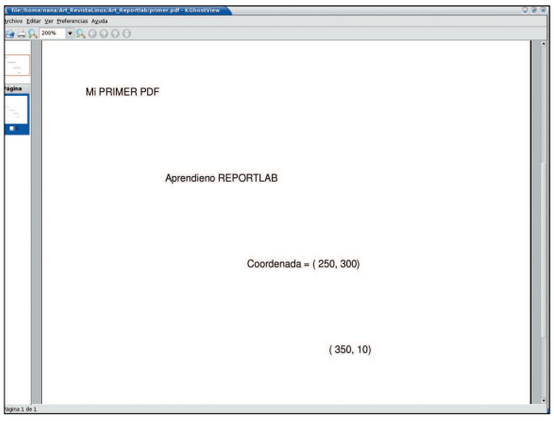

Figura 2: Primer documento generado. El resultado impreso refleja cómo controlar las coordenadas de una hoja.

La clase *Canvas* dispone de diferentes herramientas para dibujar líneas, circunferencias, rectángulos, arcos, etc. Además permite modificar el color de los objetos, rotar, trasladar, indicar tipo y tamaño de fuente, etc. En el código del Listado 1 podemos ver cómo se dibujan, por ejemplo líneas, mediante *canvas.line(x1,y1,x2,y2)*; círculos, empleando el método *canvas.circle(x_centro,y_centro,radio,stroke = 1,fill = 1)*; y rectángulos con esquinas redondeadas, *canvas.roundRect(x,y,ancho,alto,angulo,stroke = 1,fill = 0)*. Nótese que cada vez que se quiera emplear un color nuevo hay que indicarlo mediante *canvas.setFillColorRGB(r,g,b)*, para el color de relleno, o *canvas.setStrokeColorRGB(r,g,b)*, para fijar el color de las líneas. La elección del tipo de fuente se realiza usando *canvas.SetFont(tipo_fuente,tamaño)*. En la Figura 3 se muestra el PDF que creamos con el simple código del Listado 1. Podemos probar a cambiar propiedades (en el manual de ReportLab encontraremos muchas otras).

Añadiendo imágenes

En este instante ya podemos demostrar nuestra creatividad en dibujo "artístico", aunque de un modo bastante laborioso. Seguro que más de uno preferimos incluir en nuestros ficheros imágenes ya creadas. Pues esto es posible: en el área de descarga de Linux Magazine tenemos la imagen *Tux2.png* para las diferentes pruebas.

A la hora de incluir imágenes podemos optar por las siguientes opciones. La pri-

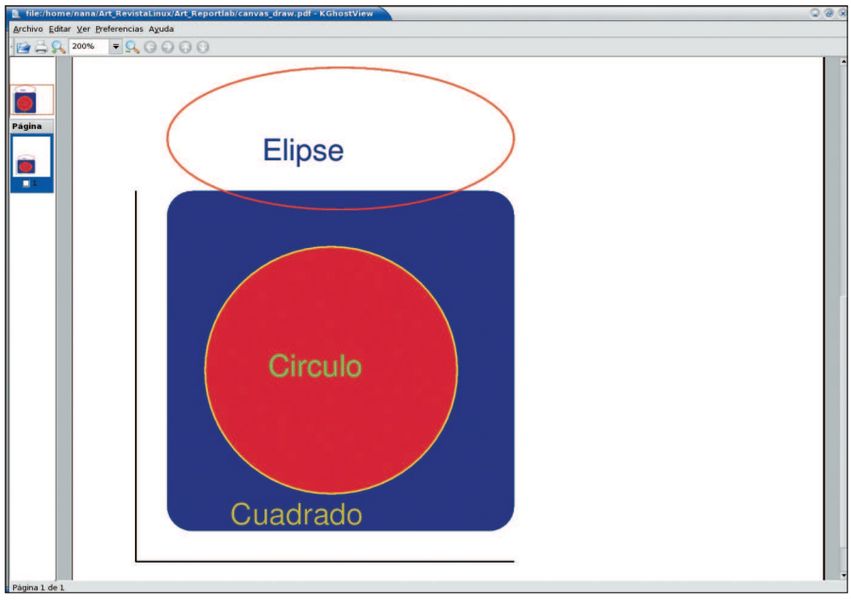

Figura 3: Objetos que se pueden dibujar con un Canvas.

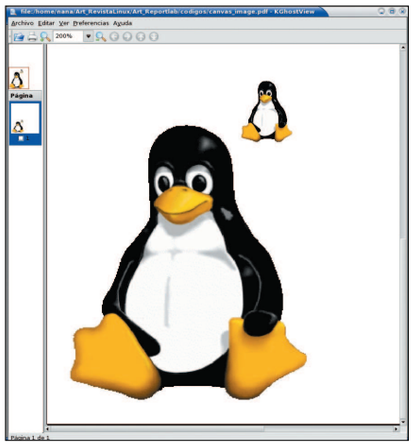

Figura 4: Colocando imágenes con *drawImage*.

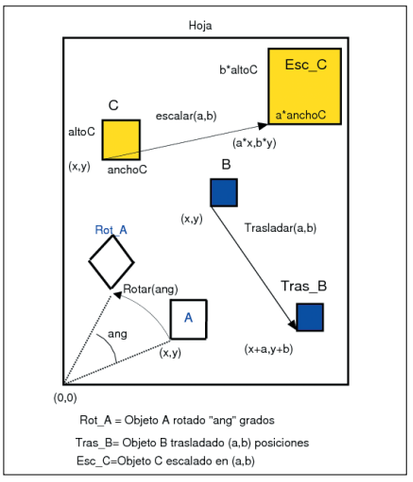

Figura 5: Modo en que se rota, traslada y escala un objeto *Drawing*.

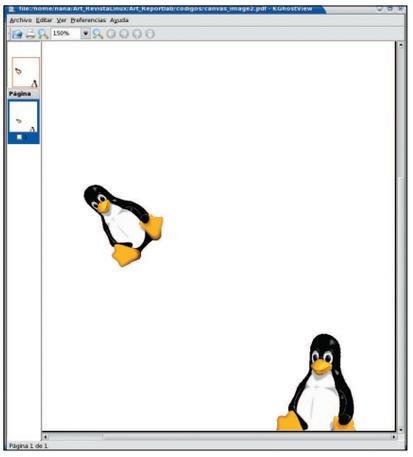

Figura 6: Ejemplo de rotación, traslación y escalado de imágenes.

mera y más sencilla, pero con la que no nos es posible rotar, trasladar, ni redimensionar es mediante el método *drawImage(image,x,y,width = None,height = None)* de la clase *Canvas* (si no se especifica el alto y el ancho, coloca la figura con sus dimensiones originales). Mediante las siguientes líneas podemos crear un fichero similar al de la Figura 4.

```
c.drawImage("Tux2.png",0,0)
```

Listado 2: Jugando con imagenes (ejemplo3_2.py)

```
01 from reportlab.graphics.shapes
      import Image, Drawing
02 from reportlab.graphics import
   renderPDF
03 from reportlab.lib.pagesizes
   import A4
04 inpath="Tux2.png"
05 IMAGES=[]
06 d=Drawing(80,100)
07 img=Image(200,0,80,100,inpath)
08 d.add(img)
09 d.rotate(45)
10 d.scale(1.5,1.5)
11 IMAGES.append(d)
12 d=Drawing(80,100)
13 d.add(img)
14 d.translate(10,0)
15 d.scale(2,2)
16 d.rotate(-5)
17 IMAGES.append(d)
18 d=Drawing(A4[0],A4[1])
19 for img in IMAGES:
20     d.add(img)
21 renderPDF.drawToFile(d,"can-
   vas_image2.pdf")
```

```
c.drawImage("Tux2,png",200,300,
 width=30,height=60)
```

Si lo que pretendemos es rotar imágenes o escalarlas, debemos emplear los objetos *Image(x,y,ancho,alto,path_imagen)* y *Drawing(ancho,alto)* que se importan mediante *from reportlab.graphics.shapes import Image, Drawing*. El objeto *Drawing* puede escalarse, rotarse y trasladarse; pero hay que tener en cuenta que todas estas operaciones son acumulativas (ver Figura 5). En el Listado 2 podemos ver cómo emplear correctamente estos objetos (Figura 6). Obsérvese que ahora el PDF no se genera a partir de un *Canvas*, sino que se genera mediante *renderPDF.drawToFile(d,"canvas_image2.pdf")*, donde *d = Drawing(A4[0],A4[1])*. Podemos probar a modificar los valores de los distintos métodos *scale*, *rotate*, *translate*; observaremos que a veces la imagen puede desaparecer del folio, eso es debido a que los valores que se dan hacen que nos salgamos de las dimensiones de la página.

Creación de párrafos y tablas

La clase *reportlab.platypus.Paragraph* permite escribir texto formateado (justificado, alineado a la derecha o izquierda, centrado) en un aspecto elegante, además de modificar el estilo y color de trozos de la línea a través de XML. Mediante *Paragraph(texto,style,bulletText = None)* se instancia la clase; *texto* contiene el texto del párrafo, en el que se eliminan los espacios en blanco innecesarios; *bulletText* indica si el párrafo se escribe con un punto al inicio del mismo; la fuente y otras propiedades del párrafo y el punto se indican mediante el argumento *style*. Veamos cómo añadir un párrafo:

```
01 from reportlab.lib.styles
   import getSampleStyleSheet
02
   styleSheet=getSampleStyleShee
   t()
03 story=[]
04 h1=styleSheet['Heading1']
05 h1.pageBreakBefore=0
06 h1.keepWithNext=1
07 h1.backColor=colors.red
08 P1=Paragraph("Estilo Cabecera
   - h1 ",h1)
09 story.append(P)
10 style=styleSheet['BodyText']
11 P2=Paragraph("Estilo
   BodyText"
   ,style)
12 story.append(P2)
```

El paquete *reportlab.lib.styles* contiene estilos predefinidos. Con *getSampleStyleSheet* obtenemos un estilo ejemplo. Tenemos un estilo para la cabecera y otro para el texto normal. Mediante *h1.pageBreakBefore = 0* decimos que no queremos un salto de página cada vez que se escriba una cabecera *h1*, en caso contrario basta escribir 1. Los diferentes párrafos se van almacenando en la lista *story* porque posteriormente se añaden al pdf a través el paquete *SimpleDocTemplate* de *reportlab.platypus*:

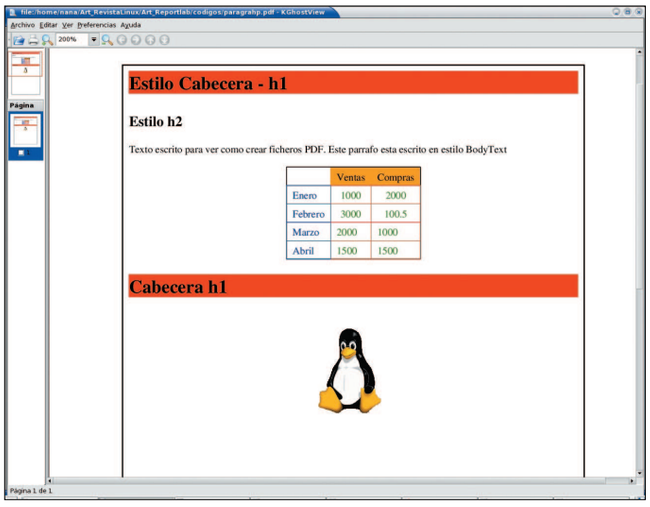

Figura 7: Ejemplo de párrafo, tabla e imagen.

```
doc=SimpleDocTemplate(➥
"paragrahp.pdf", pagesize=A4,➥
showBoundary=1)
doc.build(story)
```

En este caso, se genera un PDF con tantas páginas como sea necesario. Comprobar esto es tan sencillo como tener un texto muy largo. Si añadimos un párrafo cuyo texto sea *"Hola"*300*, seguro que se generan más de una hoja.

Si lo que queremos es añadir una tabla, es necesario importar *from reportlab.platypus import Table,TablsStyle*. Una tabla se crea añadiendo una lista de listas, donde cada componente de la lista guarda la información de cada fila. Si queremos construir una tabla de 5 filas y 3 columnas hacemos

```
t=Table([['','Ventas',➥
'Compras'],➥
['Enero',1000, 2000],➥
['Febrero',3000,100.5],➥
['Marzo',2000,1000],➥
['Abril',1500,1500]]
```

En una tabla se puede fijar el estilo de cada miembro de la misma. Si por ejemplo, se quiere que el texto de la primera columna sea azul, y que los números sean todos verdes, haremos

```
t.setStyle([➥
 ('TEXTCOLOR',(0,1),(0,-1),➥
colors.blue), ('TEXTCOLOR',➥
(1,1), (2,-1),colors.green)])
```

En el código del Listado 3 mostramos un ejemplo en él se ilustra cómo adaptar el estilo según se quiera (Figura 7). Podemos ver que para incluir un nuevo elemento en el PDF es suficiente con ir añadiendo cada objeto a la lista *story*.

Ahora ya sabemos todo lo necesario para crear nuestros propios carteles, informes, catálogos, presentaciones, etc. ■

RECURSOS

[1] Reportlab: *http://www.reportlab.org*

Listado 3: Crear Tablas y Párrafos (ejemplo4.py)

```
01 from reportlab.lib.pagesizes
   import A4
02 from reportlab.lib.styles
   import
   getSampleStyleSheet,ParagraphS
   tyle
03 from reportlab.platypus import
   Spacer,
     SimpleDocTemplate, Table,
   TableStyle
04 from reportlab.platypus import
   Paragraph, Image
05 from reportlab.lib import
   colors
06
07
   styleSheet=getSampleStyleSheet
   ()
08 story=[]
09 h1=styleSheet['Heading1']
10 h1.pageBreakBefore=0
11 h1.keepWithNext=1
12 h1.backColor=colors.red
13 h2=styleSheet['Heading2']
14 h2.pageBreakBefore=0
15 h2.keepWithNext=1
16 P=Paragraph("Estilo Cabecera -
   h1 ",h1)
17 story.append(P)
18 P=Paragraph("Estilo h2 ",h2)
19 story.append(P)
20 style=styleSheet['BodyText']
21 texto=" Texto escrito para ver
   como
      crear ficheros PDF."+\
22    "Este parrafo esta escrito
       en estilo BodyText"
23 texto_largo=texto
24 #texto_largo=texto*100
25 P=Paragraph(texto_largo,style)
26 story.append(P)
27 story.append(Spacer(0,12))
28
29
   t=Table([['','Ventas','Compras
   '],
30         ['Enero',1000, 2000],
31
   ['Febrero',3000,100.5],
32         ['Marzo',2000,1000],
33         ['Abril',1500,1500]]
34     )
35
36 story.append(t)
37
38 story.append(Spacer(0,15))
39 P=Paragraph("Cabecera h1",h1)
40 story.append(P)
41
42 cadena=''' Mediante ReportLab
   es posible
43     generar ficheros PDF de
   gran
44     calidad. Es posible
45     incluir graficos, image-
   nes,
46     tablas; creando informes
47     de gran calidad '''
48 P=Paragraph(cadena,style)
49
50 story.append(Spacer(0,15))
51
52
   img=Image("Tux2.png",width=80,
   height=100)
53 story.append(img)
54 doc=SimpleDocTemplate("para-
   grahp.pdf",pagesize=A4,showBou
   ndary=1)
55 doc.build(story)
```