Todo

Año 1 • Número 13 • 6,50 euros

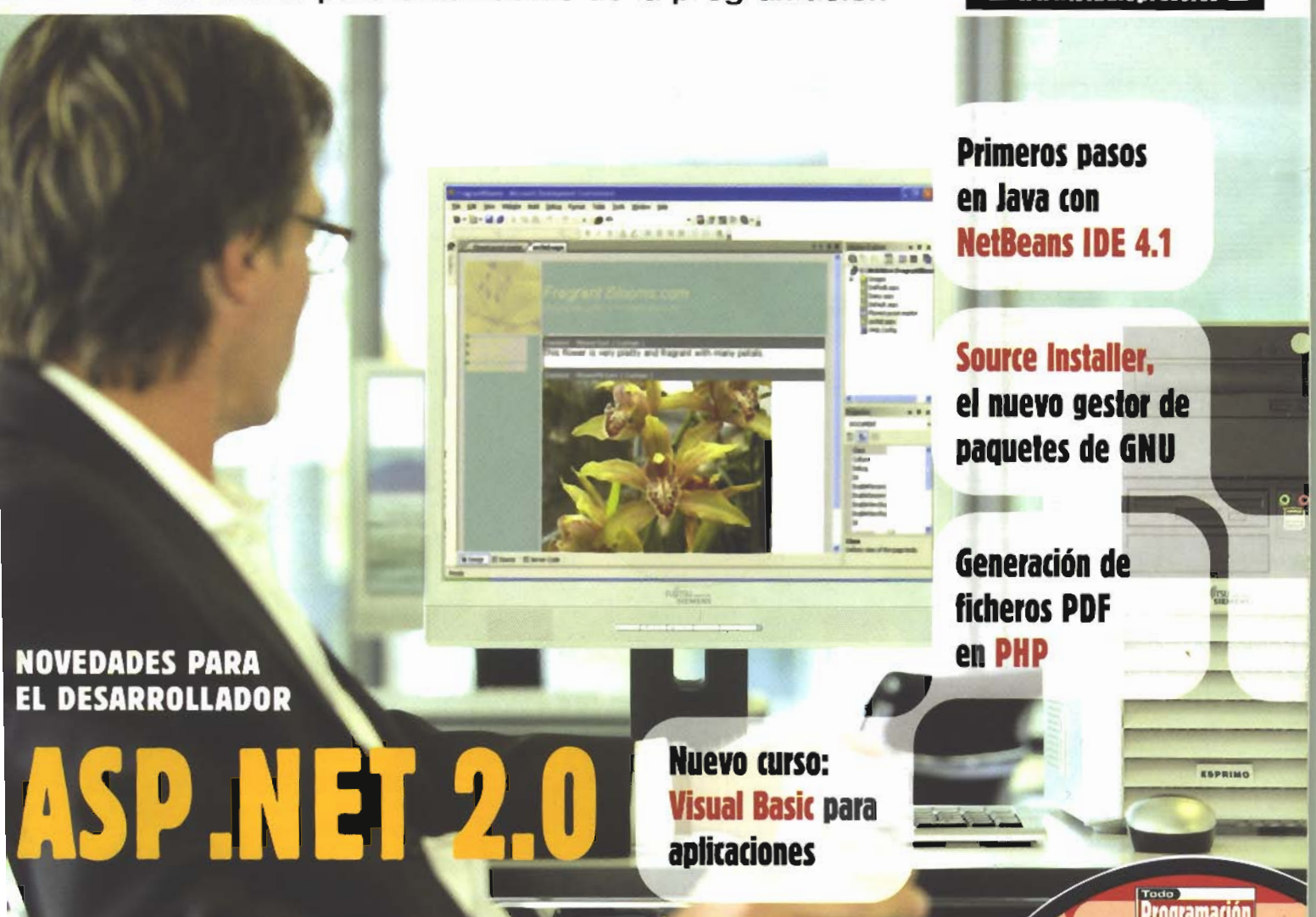
Studio Press

8 413042 751977 00013

Programación

La Revista bimestral para entusiastas de la programación

www.studiopress.es



Primeros pasos en Java con NetBeans IDE 4.1

Source Installer, el nuevo gestor de paquetes de GNU

Generación de ficheros PDF en PHP

NOVEDADES PARA EL DESARROLLADOR

ASP.NET 2.0

Nuevo curso: Visual Basic para aplicaciones

■ CD-ROM: ESPECIAL DESARROLLO JAVA (NETBEANS, ECLIPSE, J2EE, J2ME, ETC.)

Zona Linux

- C#: Las expresiones regulares
- Grids: Programación práctica con Globus

Zona Windows

- F#: Programación funcional para .NET
- Control de fuentes con FreeVCS

EL API DE NOKIA PARA LA PROGRAMACIÓN DE JUEGOS CON JAVA MOBILE

Todo Programación

La Revista bimestral para entusiastas de la programación

DIRECTOR
Eduardo Toribio
etoribio@iberprensa.com

REDACCIÓN
Yenifer Trabadela
yenifer@iberprensa.com

COLABORADORES
Antonio M. Zugaldía
(azugaldia@iberprensa.com)
David Santo Orcero
(orcero@iberprensa.com)
Manuel Domínguez
(mdominguez@iberprensacom)
Fernando Escudero
(fescudero@iberprensa.com)
José Manuel Navarro
(jnavarro@iberprensa.com)
Marcos Prieto
(mprieto@iberprensa.com)
Guillermo "el Guille" Som
(elguille@iberprensa.com)
Santiago Márquez
(smarquez@iberprensa.com)
José Rivera
(jrivera@iberprensa.com)
Alejandro Serrano
(aserrano@iberprensa.com)
Jordi Massaguer
(jmassague@iberprensa.com)
Pedro Agulló
(pagullo@iberprensa.com)
Moisés Díaz
(mdiaz@iberprensa.com)

DISEÑO PORTADA
Antonio Gª Tomé

MAQUETACIÓN
Antonio Gª Tomé

DIRECTOR DE PRODUCCIÓN
Carlos Peropadre
cperopadre@iberprensa.com

ADMINISTRACIÓN
Marisa Cogorro

SUSCRIPCIONES
Tel: 91 628 02 03
suscripciones@iberprensa.com

FILMACIÓN: Fotpreim Duvial
IMPRESIÓN: I. G. Printone
DUPLICACIÓN CD-ROM: M.P.O

DISTRIBUCIÓN
S.G.E.L.
Avda. Valdelaparra 29 (Pol. Ind.)
28108 Alcobendas (Madrid)
Tel.: 91 657 69 00

EDITA: Studio Press
www.studiopress.es

REDACCIÓN, PUBLICIDAD Y ADMINISTRACIÓN
C/ del Río Ter, Nave 13.
Polígono "El Nogal"
28110 Algete (Madrid)
Tel.: 91 628 02 03*
Fax: 91 628 09 35
(Añada 34 si llama desde fuera de España.)

DEPÓSITO LEGAL: M-13679-2004
Número 13 • Año 1

EDITORIAL

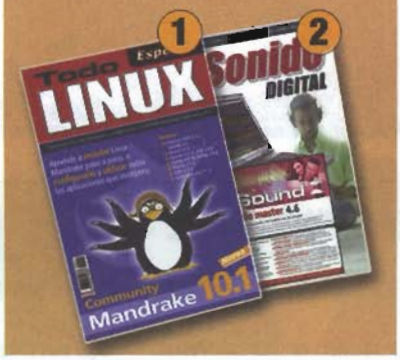
Eduardo Toribio

Nueva temporada

En este nuevo número de **Todo Programación** hemos tratado de reunir una oferta que satisfaga a todos nuestros lectores, lo cual no es fácil teniendo en cuenta lo heterogéneo de sus conocimientos y aficiones. Por un lado, los interesados en la programación con la plataforma .NET podrán encontrar varias secciones destacadas, como el reportaje sobre ASP. NET 2, el curso de C# o la sección dedicada al lenguaje funcional F#. Si, en cambio, el lector se decanta por Java, podrá acceder a un review sobre el nuevo NetBeans IDE 4.1, el curso de J2ME y podrá encontrar una recopilación de herramientas para desarrollo Java en el CD-ROM.
Para los principiantes incluimos una nueva sección orientada a enseñar a programar con Visual Basic para aplicaciones, y después, "apto para todos los públicos", una serie de talleres prácticos que entendemos pueden interesar a la mayoría de nuestros lectores, como los dedicados a la herramienta de control de versiones FreeVCS, otro a la realización de PDFs con la utilidad FPDF, y un tutorial práctico sobre programación de redes con Qt. Como siempre, nuestro objetivo es formar y mantener actualizado al lector en los principales lenguajes de programación y en las aplicaciones más demandadas por parte de las empresas para las plataformas Windows y GNU/Linux.

SUSCRIPCIONES

Como oferta de lanzamiento existe la posibilidad de suscribirse durante 12 números a **Todo Programación** por solo 61 euros lo que significa un ahorro del 20% respecto el precio de portada. Además de regalo puedes elegir entre una de las dos guías que aparecen abajo.
Más información en: www.studiopress.es



SERVICIO TÉCNICO

Todo Programación dispone de una dirección de correo electrónica y un número de Fax para formular preguntas relativas al funcionamiento del CD-ROM de la revista.
e-mail: todoprogramacion@iberprensa.com
Fax: 91 628 09 35

LECTORES

Comparte con nosotros tu opinión sobre la revista, envíanos tus comentarios, sugerencias, ideas o críticas.

Studio Press
(Todo Programación)
C/ Del Río Ter, Nave 13
Pol. "El Nogal"
28110 Algete. Madrid

DEPARTAMENTO DE PUBLICIDAD

Si le interesa conocer nuestras tarifas de publicidad no dude en ponerse en contacto con nuestro departamento comercial:
- **Tel. 91 628 02 03**
- **e-mail: publicidad@iberprensa.com**

Sumario

Número 13

En portada

ASP. NET 2.0

10 La tecnología ASP.NET se ha impuesto más rápido que la propia .NET, de hecho, es hoy en día una forma para crear sitios web muy aceptada por parte de la comunidad de desarrolladores, ya que ofrece muchas ventajas y facilidades en la fase de programación. Ahora, la versión 2.0 de ASP.NET trae bastantes cambios, la mayor parte de ellos convertidos en ventajas para el programador como vamos a poder ver en nuestro reportaje de portada.

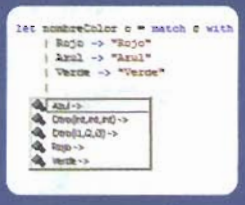
A quién vamos dirigidos

Todo Programación (TP) es una revista para programadores, escrita por programadores y con un enfoque eminentemente práctico. Trataremos de ser útiles al programador, tanto al profesional como al estudiante. Si hay algo cierto en este sector es que nunca podemos parar, vivimos en un continuo proceso de reciclaje. Ahí es donde tratará de encajarse TP: información, actualidad, cursos y prácticas de los lenguajes más demandados y formación en sistemas.

.net ZONA WINDOWS

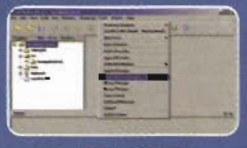
F#: Avanzando en la programación funcional >>

28 En nuestro número anterior vimos una introducción a la programación funcional para la plataforma .NET. Ahora rematamos, esta miniserie centrándonos en las características y la sintáxis del lenguaje F#.



Análisis de software: NetBeans IDE 4.1 >>

32 Lo que se nos presentaba como un simple incremento de versión menor en este popular IDE Java, parece que es, en realidad, un cúmulo de importantes mejoras, como veremos en nuestro reportaje: soporte mejorado para EJB, múltiples árboles de fuentes, etc.



REPORTAJE: Control de versiones con FreeVCS >>

16 Dedicamos un taller a estudiar el funcionamiento y las ventajas de los programas llamados de control de fuentes o de versiones. Realizaremos un ejemplo práctico utilizando la aplicación FreeVCS, disponible para Windows bajo licencia open source.

Ejemplos y código fuente

Cada CD-ROM de la revista incluye una carpeta denominada *fuentes* en la que se encuentra el material complementario para seguir cada uno de los cursos: por ejemplo, los listados completos, los ejemplos desarrollados en diversos lenguajes, compiladores, editores, utilidades y, en general, cualquier herramienta que se mencione o cite en la respectiva sección.
Todo con la finalidad de completar la formación y facilitar el seguimiento de cada artículo por parte del lector.

CONTENIDO DEL CD-ROM

Herramientas y recursos para el programador

64 En el CD de **Todo Programación** puedes encontrar en cada numero las herramientas de programación más útiles y que conseguirán simplificar tu trabajo. Además, tendrás a tu disposición todas las aplicaciones y los ejemplos que se requieran para seguir correctamente los artículos y cursos que presentamos en la revista. Este número tiene su punto fuerte en el desarrollo Java.

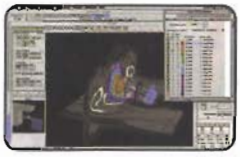
TALLER PRÁCTICO

Programación de Grids: un ejemplo práctico >>

40 En los artículos anteriores de nuestra serie hemos revisado los conceptos fundamentales de Globus. Ahora vamos a despiezar una aplicación "Hola Mundo", para poder entender cómo funciona un programa en Globus internamente.

Generación de PDFs en PHP con FPDF >>

43 FPDF es un proyecto desarrollado bajo licencia libre, escrito en PHP, y cuyo objetivo es simplificar al máximo la generación de documentos en formato PDF, sin que pierdan las propiedades que los hacen tan característicos.



Gráficos tridimensionales fotorealistas con OpenGL >>

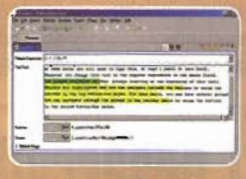
48 El estándar OpenGL es uno de los más antiguos para programación de gráficos en 3D de los que existen en el mercado, y también, sin lugar a dudas, el más empleado en todo tipo de aplicaciones industriales.

ZONA LINUX

Actualidad: Novedades en lenguajes y compiladores >>

21 GNU Source Installer es un proyecto anunciado recientemente, que tiene como objetivo administrar y facilitar la instalación de los clásicos paquetes TAR.GZ desde entorno gráfico. Veremos cómo funciona esta aplicación.

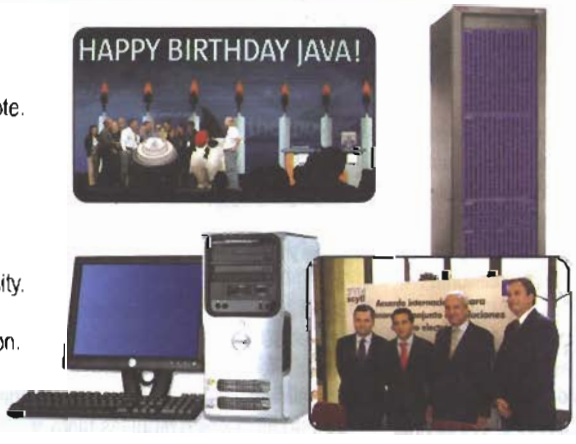
C#: Expresiones regulares en C# >>



24 Las expresiones regulares son prácticamente un lenguaje de programación independiente, pero son además una poderosa herramienta heredada del lenguaje Perl que permite la manipulación avanzada de cadenas de texto.

NOTICIAS

6. **JavaOne** 2005.
6. **HP** y Scytl desarrollarán soluciones de e-vote.
7. **Bea,** HP, INFODESA y Oracle desarrollan el "Proyecto Pináculo".
7. **Proyecto** ecológico de Canon y RSA.
8. **Nueva** versión de Qt.
8. **Bea** anuncia nuevos desarrollos para Java.
8. **Sun** lanza el programa Share the Opportunity.
8. **Vodafone** 3G utilizará Java.
9. **Ordenadores** de sobremesa Dell Dimension.
9. **Nuevos** proyectores digitales HP vp6300.
9. **Sun** Storedge 9985.



Y ADEMÁS...

PROGRAMACIÓN DE REDES MULTIPLATAFORMA CON QT: Un Messenger>>

52 En este taller práctico nos introduciremos en la programación de redes con Qt. Para ello veremos un ejemplo sobre cómo programar una aplicación de tipo Messenger con Qt para que pueda utilizarse en Windows, MacOS y Linux.

JUEGOS JAVA: El API de Nokia >>

60 Última entrega de nuestro curso sobre programación de juegos con J2ME, donde abordamos dos temas importantes: el API de Nokia y el entorno de desarrollo para J2ME de Eclipse.

DESARROLLO WEB: PHP desde la línea de comando >>

56 Las posibilidades del lenguaje PHP van mucho más allá de la creación de páginas dinámicas, es decir, podemos emplear PHP como un lenguaje estándar al estilo de C, Pascal, Visual Basic, etc. Pero eso sí, un lenguaje interpretado, no compilado. Veamos algunos ejemplos prácticos.

CUADERNOS DE PRINCIPIANTES

Visual Basic para Aplicaciones: macros en word >>

35 Comenzamos una serie dedicada a principiantes, y centrada en el uso del lenguaje Visual Basic para aplicaciones, que podemos emplear para extender las funcionalidades de la suite Microsoft Office.

NIVEL DE DIFICULTAD:

MATERIAL INCLUIDO EN EL CD-ROM:
fpdf 1.5.3 (ZIP y TGZ) ■ scite-1.5.7. exe

Generación de PDFs en PHP con FPDF

MANUEL DOMÍNGUEZ (mdominguez@iberprensa.com)
SARA B. BARRIO

En la sociedad global actual, la información es el bien más preciado. Para asegurar el éxito de una aplicación web o un portal no basta con presentar información al usuario, sino que cada vez más éste exige poder descargarla para leerla cómodamente o imprimirla.

El estándar de facto para documentos electrónicos es PDF (Portable Document Format) de Adobe y para la generación de documentos en este formato hay muchas librerías, entre ellas, FPDF, una solución libre y gratuita para ser usada en aplicaciones PHP

INTRODUCCIÓN

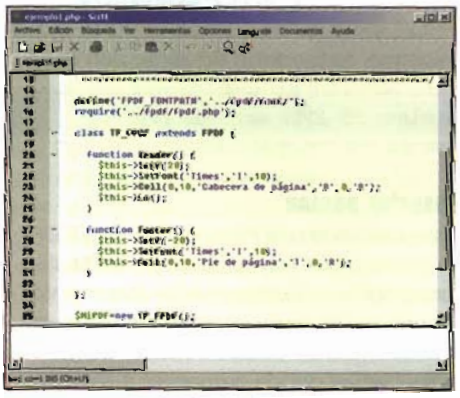
FPDF es una clase escrita completamente en PHP cuyo objetivo es simplificar lo máximo posible la generación de documentos en PDF sin que pierdan las propiedades que los hacen tan característicos. Otra ventaja que nos aporta FPDF son las funciones de alto nivel, por ejemplo, elegir centímetros como unidades de medida, el formato del papel y el formato de página También permite crear saltos de página automáticos, colores, insertar imágenes, etc

FPDF, EL PROYECTO

FPDF es un proyecto de software libre creado por Olivier Plathey, con licencia freeware y con permiso expreso del autor (visible en el propio código fuente) para modificar y usar el código como nos venga en gana, sea para un proyecto comercial o no, de código cerrado o abierto. Es decir, el proyecto está liberado con una licencia nada restrictiva.

REQUERIMIENTOS

Para hacer uso de la clase FPDF, dado que está implementada en PHP, es muy conveniente utilizar un buen editor de código fuente PHP, como PHP Editor o Scite. Este último se entrega en el CD de la revista. También es necesario un navegador para poder probar que lo que realmente estamos haciendo funciona, aunque PHP en



Edición de código PHP con Scite.

las últimas versiones puede actuar en línea de comandos sin necesidad de explorador Y por supuesto un visor de PDF, bien Acrobat Reader de Adobe, Foxit PDF Reader o Ghost View, que se puede encontrar fácilmente en la Web.

Durante todo este taller, para no salirnos del tema que nos ocupa, vamos a suponer que el lector se encuentra familiarizado con PHP y que tiene instalado y configurado Apache/PHP correctamente, apuntando a un directorio concreto del disco duro y un visor de PDF. De hecho, actualmente en **TP** existe una sección paralela centrada en dicho lenguaje.

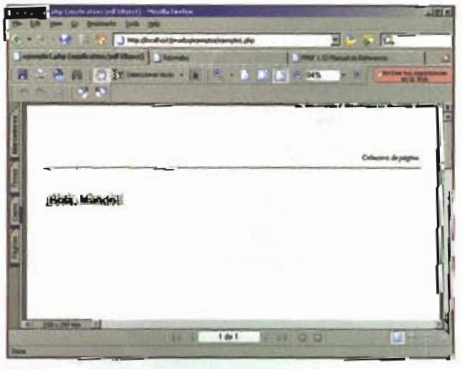
DESCARGA E INSTALACIÓN

En la sección de descargas de la página del proyecto (*http://fpdf.org/*) disponemos de la versión 1.53, que es la que utilizaremos en los ejemplos, que también incluimos en nuestro CD-ROM. Guardamos y descomprimimos el archivo y aparecerá una estructura de carpetas/directorios. El archivo principal, *fpdf.php*, contiene la clase que debemos usar durante todo el transcurso de generación del fichero PDF, por lo cual debe estar accesible en el sitio web donde deseemos hacer uso de la librería, sus métodos nos permitirán realizar todas las operaciones necesarias. La carpeta *Font*, contiene la definición de las fuentes de letras que se pueden utilizar. Estos ficheros son necesarios para la salida de texto en el documento y ya explicaremos más adelante cómo se utiliza. El directorio *Doc* contiene las páginas HTML con el manual de referencia al completo. Se detalla perfectamente el uso de cada uno de los métodos de la clase *FPDF*. Por último, en el directorio *Tutorial* se incluyen varios ejemplos completos comentados, desde un simple ¡Hola mundo!, hasta otros más elaborados que nos permitirán comprender la filosofía de funcionamiento de la librería FPDF.

EL API DE FPDF

Pero pongámonos manos a la obra empezando paso a paso a exprimir toda la funcionalidad de FPDF. Para ello comenzaremos generando un documento PDF básico, y le iremos añadiendo nuevas funcionalidades en el transcurso de este taller.

Lo primero que hay que hacer para poder utilizar FPDF en un programa PHP, es proporcionar a la librería accesibilidad al



PDF generado por ejemplo1.php.

FPDF es una clase escrita en PHP cuyo objetivo es simplificar al máximo la generación de PDFs

directorio donde se encuentran las definiciones de fuentes de letras y posteriormente incluir la librería para su uso. Es un proceso muy metódico, algo como:

```
define('FPDF_FONTPATH',
'/home/www/font/');
require('fpdf.php');
```

En el primero de los casos, se debe especificar la ruta del directorio *font*. Ésta puede ser absoluta o relativa al directorio actual, pero no debe faltar. En el segundo caso ocurre exactamente igual, hay que especificar la ruta completa hasta el fichero *fpdf.php*, absoluta o relativa.

Una vez realizada esta operación, ya tenemos completo acceso al API de FPDF y estamos en disposición de comenzar con la generación de documentos PDF.

La clase FPDF

FPDF es la clase principal de la librería. Contiene 45 métodos que nos permitirán personalizar nuestros documentos PDF, y que serán más que suficiente para la mayor parte de las necesidades habituales. Algunos de los métodos implementan toda la funcionalidad necesaria, por ejemplo, los de selección de la letra, tamaños etcétera. Otros, sin embargo, son métodos vacíos (abstractos) que no implementan nada y que se deberán redefinir para su uso. Éste es el caso de lo métodos encargados de crear automáticamente la cabecera y los pies de página de nuestro documento. Así pues, existen varios modos de hacer uso de la clase *FPDF*. El primero consiste en crear una instancia de la clase directamente:

```
$MiPDF = new FPDF();
```

Y a partir de ese momento podemos usar los métodos que deseemos de la clase. Por supuesto, los métodos que no implementan nada, seguirán sin hacer nada tras la creación del objeto *MiPDF*. También existe la segunda opción, más extendida y al final la que se acaba utilizando siempre, que consiste en heredar de la clase *FPDF*.

```
class MiClasePDF extends FPDF {
  ...
}
$MiPDF = new MiClasePDF();
```

donde creamos una clase *MiClasePDF*, que hereda de *FPDF*, y dentro de la cual podemos redefinir los métodos que no tienen implementación para que hagan lo que nosotros queramos. Posteriormente solo necesitamos crear objetos de esta nueva clase para generar documentos personalizados a nuestro antojo, en los que podremos utilizar todos los métodos de *FPDF*, además de los que le hayamos querido añadir. Éste será el camino que seguiremos en este taller.

Creando una clase propia

Durante toda la práctica vamos a trabajar sobre una clase propia heredada de *FPDF*, a la que llamaremos *TP_FPDF* y que inicialmente tendrá la siguiente definición:

```
class TP_FPDF extends FPDF {}
```

Insertar página

Tras creamos una instancia de *TP_FPDF*, lo primero que hay que hacer es añadir una página a nuestro nuevo documento donde poder alojar las tablas, el texto, las cabeceras... es decir, necesitamos un papel virtual donde comenzar a escribir. Para ello se utiliza el método *AddPage(orientacion)* de *FPDF*, que tenemos disponible en nuestro objeto por heredar de dicha clase. Este método acepta como parámetro un carácter entrecomillado; bien 'L' (*landscape*) si deseamos que la nueva página sea apaisada, bien 'P' (*portrait*) si queremos que ésta sea vertical.

Tras la invocación de dicho método, la posición actual de inserción de texto o cualquier otro objeto pasa a ser la esquina superior izquierda de la página, respetándose cualquier atributo que estuviese activado, o sea, colores, tipo de letra, etcétera. Así que es sencillo ir añadiendo páginas como vayamos necesitando.

En el ejemplo que estamos siguiendo, insertaríamos una página con la siguiente línea de código:

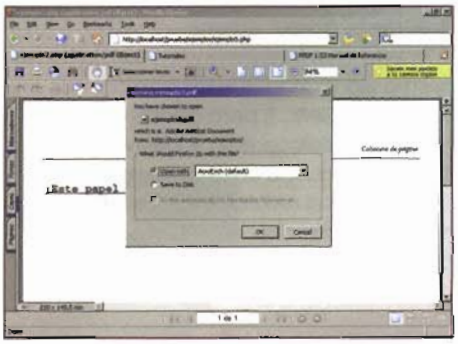
```
$MiPDF->AddPage();
```

Insertar cabecera

Una vez que tenemos el documento y páginas en él, fácilmente podremos necesitar la inclusión de una cabecera de documento en cada página. Este proceso se realiza automáticamente cuando se invoca al método *AddPage(...)* comentado en las líneas anteriores, el cual llama al método *Header()* encargado de crear la cabecera de la página. Por defecto este método *Header()* está vacío en *FPDF*, no está implementado, de modo que no aparecen cabeceras del documento. Es por ello que debemos redefinir este método en nuestra clase *TP_FPDF* e implementar en él todo lo que queremos que aparezca en la cabecera. Si no queremos cabecera, no se redefine este método en la clase hija y ya está.

Listado 1

```
<?php
  define('FPDF_FONTPATH','../fpdf/font/');
  require('../fpdf/fpdf.php');
  class TP_FPDF extends FPDF {
    function Header() {
      $this->SetY(20);
      $this->SetFont('Times','I',10);
      $this->Cell(0,10,'Cabecera de página','B',0,'R');
      $this->Ln();
    }
    function Footer() {
      $this->SetY(-20);
      $this->SetFont('Times','I',10);
      $this->Cell(0,10,'Pie de página','T',0,'R');
    }
  };
  $MiPDF=new TP_FPDF();
  $MiPDF->AddPage('P');
  $MiPDF->SetFont('Arial','BU',16);
  $MiPDF->SetY(40);
  $MiPDF->Cell(40,10,'¡Hola, Mundo!');
  $MiPDF->Ln();
  $MiPDF->Output();
?>
```



El navegador no abre el fichero sino que muestra el diálogo para guardar.

Insertar pie

Con el pie de página ocurre exactamente igual que con la cabecera. Se crea automáticamente cuando se invoca al método *AddPage(...)*, y en este caso, también cuando se finaliza el documento con *Close()*, que veremos posteriormente. Ambos métodos invocan al método *Footer()*, encargado de dibujar el pie de cada página. También, al igual que *Header()*, es un método sin implementación que debe ser redefinido e implementado en la clase que herede de *FPDF*, en nuestro caso *TP_FPDF*. Si no deseamos tener un pie de página, basta con no redefinir este método.

En el Listado 1 de la página anterior podemos observar el uso real de los métodos vistos hasta ahora en este artículo, para irnos haciendo una idea de las posibilidades de esta librería. Se utilizan un par de métodos todavía no explicados, pero aun así se entiende fácilmente.

Tipo y estilo de letra

Como con todo lo anterior ya tenemos un documento en blanco para escribir, ahora ya podemos empezar a seleccionar la fuente de texto a utilizar. Para ello tenemos el método *SetFont(Tipo, Estilo, Tamaño)*, de la clase *FPDF*, donde *Tipo* puede ser Couner, Times, Arial, Symbol o zapfDingbats, definidas en la carpeta *fonts*, de la cual hablamos en líneas atrás. *Estilo* es una cadena de caracteres formada por uno, dos o tres caracteres que pueden ser "B", "I" o "U" o una combinación de ellos, significando negrita, cursiva y subrayado, respectivamente. Una cadena vacía en el atributo *Estilo* indicará un tipo de letra regular, sin estilo. Por último, *Tamaño* indica el tamaño (valga la redundancia) de la fuente en puntos. En nuestro caso, la siguiente línea:

```
$MiPDF->SetFont('Arial', 'BI', 14);
```

seleccionaría el tipo de letra Arial (o Helvética, es lo mismo) en negrita, cursiva y tamaño 14.

Estilo de página

Hasta ahora hemos utilizado un formato de página por defecto, el que está definido en el constructor de la clase *FPDF* por si se invoca sin pasarle ningún parámetro. Realmente el constructor de la clase *FPDF* tiene el siguiente aspecto: *FPDF(orientacion, unidades, formato)*, donde *orientación* especifica si el papel se considerará en vertical ('P'), o apaisado ('L'). *Unidades* sirve para especificar al constructor cuáles serán las unidades de medida que se utilizarán: punto ('pt'), milímetros ('mm'), centímetros ('cm') o pulgadas ('in'). Por último, con el parámetro *formato* diremos a la clase el tamaño del papel, que puede ser unas de las siguientes constantes de texto: *A3*, *A4*, *A5*, *Letter* y *Legal*, que representan a los formatos estándares con idéntico nombre. Con estos formatos tenemos resueltas la mayoría de las necesidades. Además, usando otros parámetros se pueden especificar formatos personalizados.

Este constructor se llamaría, por ejemplo, con una línea como la siguiente:

```
$MiPDF=FPDF('P', 'mm', 'A3');
```

Y si hemos heredado una clase de la clase *FPDF*, habría que sobrescribir el constructor y llamar al constructor de la clase padre. En el Listado 2 se observa este uso, que en principio podría parecer un poco complicado, pero no lo es en absoluto.

Insertar texto

A estas alturas el lector ya habrá apreciado en los ejemplos 1 y 2 la forma común de escribir texto con FPDF. En efecto lo más usual, por la facilidad que ofrece para la maquetación de las páginas, es crear una celda e incluir dentro de ella el texto que deseamos escribir. Esto ofrece innumerables ventajas como, por ejemplo, que se puede alinear el texto dentro de esa celda, situarlo en el lugar que se desee, mostrar todos o algunos de los bordes de la celda y el relleno de la misma, etcétera.

El método usado es *Cell(ancho, alto, texto, borde, salto, alineacion, relleno, enlace)*, aunque no es necesario utilizar y/o especificar todos los parámetros. *Ancho* y *alto* especifican las dimensiones de la celda en las unidades que se hayan definido en el constructor. *Borde* indica qué aristas de la celda se deben mostrar, a saber: 0 para ninguno, 1 para todas, 'L' para izquierda, 'T' para arriba, 'R' para la de la derecha y 'B' para la inferior. De estos cuatro últimos valores se puede

Listado 2

```
<?php
  define('FPDF_FONTPATH','../fpdf/font/');
  require('../fpdf/fpdf.php');
  class TP_FPDF extends FPDF {
    function TP_FPDF($orientacion='L', $unidades='mm', $formato='A4') {
      $this->FPDF($orientacion, $unidades, $formato);
    }
    function Header() {
      $this->SetY(20);
      $this->SetFont('Times','I',10);
      $this->Cell(0,10,'Cabecera de página','B',0,'R');
      $this->Ln();
    }
    function Footer() {
      $this->SetY(-20);
      $this->SetFont('Times','I',10);
      $this->Cell(0,10,'Pie de página','T',0,'R');
    }
  };
  $MiPDF=new TP_FPDF('L', 'mm', 'A5');
  $MiPDF->AddPage();
  $MiPDF->SetFont('Courier','BU',16);
  $MiPDF->SetY(40);
  $MiPDF->Cell(40,10,'¡Este papel está apaisado y es un A5!');
  $MiPDF->Ln();
  $MiPDF->Output();
?>
```

especificar más de uno, por ejemplo 'RTB'. *Salto* indica qué se debe hacer antes de comenzar a crear la celda; 0 indica ir al final de la línea actual; 1 un salto de línea y retorno de carro y 2 implica el desplazamiento del cursor hacia abajo (sin desplazarse horizontalmente). *Alineacion* permite decidir cómo se ajustará el texto a la celda. 'L' significa a la izquierda, 'R' a la derecha y 'C' centrado en la celda. *Relleno=0* o *relleno=1* significa que la celda debe ser transparente o debe mostrar color de fondo, respectivamente. *Enlace* es un identificador que se puede definir, hará que esta celda sea un destino para un enlace que se cree hacia la misma dentro del documento.

Mucha funcionalidad para un solo método. Generalmente no se utilizan todos los parámetros sino que solo se suelen usar los tres primeros. Pero no está de más saber todo de este método, porque nos permitirá realizar más operaciones de las que en principio aparenta.

Salto de línea

Cuando se está escribiendo, es más que probable que se desee pasar a la línea siguiente. Esto es sencillo en FPDF, puesto que solo hay que realizar una llamada al método *Ln()* y éste se encarga automáticamente de calcular la posición del cursor de escritura, teniendo en cuenta el tamaño de la fuente seleccionada y otros aspectos. En nuestro ejemplo de todo el artículo se realizaría con la siguiente línea:

```
$MiPDF->Ln();
```

Insertar texto en varias líneas

Lo normal es que cuando vayamos escribiendo, deseemos que automáticamente se vayan añadiendo saltos de línea al llegar al final de cada una. Con *Cell(...)* esto no es posible y el texto desborda la celda por el límite derecho. Sin embargo, FPDF cuenta con el método *MultiCell(ancho, alto, texto, borde, alineacion, relleno)* donde los parámetros indicados tienen el mismo significado que los homónimos en el método *Cell(...)*, pero donde el texto a mostrar se colocará en sucesivas líneas cuando no quepa en una sola celda.

Para ver el resultado de los últimos métodos utilizados, en el Listado 3 se desarrollan con más detalle los ejemplos y métodos vistos hasta ahora

Insertar imagen

Tras aprender a diseñar un documento, insertar texto y colocar encabezados y pies de página, el lector se preguntará cómo puede añadirle imágenes, puesto que hoy en día es un requerimiento generalizado de casi cualquier documento impreso. FPDF contiene un método para ello, *Image(fichero, x, y, ancho, alto, tipo, enlace)*. El parámetro *Fichero* especifica el nombre del fichero que contiene la imagen, con su ruta relativa o absoluta. *X* e *Y* representan el punto de la esquina superior izquierda de la imagen, o lo que es lo mismo, dónde se empezará a mostrar la imagen en el documento. *Ancho* y *alto* sirven para definir el tamaño de la imagen; *tipo* indica a FPDF el formato de la imagen, que solo puede ser, por ahora, JPG y PNG. Y por último, *enlace* permite definir un destino por si se hace clic en la imagen. Realmente solo es obligatorio especificar los tres primeros parámetros y los demás son extraídos por FPDF automáticamente por el contexto (extensión del fichero, datos internos de la imagen, etcétera) y suele ser la forma de uso mas habitual de este método. Por ejemplo:

```
$MiPDF->Image('pez.pgn', 40, 40);
```

Colocaría la imagen *pez.png* en la posición (40,40) de la página actual. Si una imagen se usa más de una vez dentro del mismo documento, FPDF solo incorporará los datos de la imagen al documento PDF una vez, reduciendo así el tamaño final del fichero generado.

Generación del documento

Pues bien, con todo esto hemos visto lo principal para poder generar un documento PDF que resuelva rápidamente las necesidades de casi cualquier aplicación web sin necesidad de usar un API complicado, a costa de no contar con opciones extremadamente avanzadas. Sin embargo, estas opciones avanzadas no suelen hacer excesiva falta en el contexto para el cual está pensado FPDF.

Listado 3

```
<?php
  define('FPDF_FONTPATH','../fpdf/font/');
  require('../fpdf/fpdf.php');
  class TP_FPDF extends FPDF {
    function TP_FPDF($orientacion='L', $unidades='mm', $formato='A4') {
      $this->FPDF($orientacion, $unidades, $formato);
    }
    function Header() {
      $this->SetY(20);
      $this->SetFont('Times','I',10);
     $this->Cell(0,10,'Taller: generación de PDF en PHP con FPDF','B',0,'R');
      $this->Ln();
    }
    function Footer() {
      $this->SetY(-20);
      $this->SetFont('Times','I',10);
      $this->Cell(0,10,'Manuel Domínguez Dorado','T',0,'R');
    }
  };

  $fichero=fopen('fichero.txt','r');
  $texto=fread($fichero,filesize('fichero.txt'));
  fclose($fichero);
  $MiPDF=new TP_FPDF('P', 'mm', 'A4');
  $MiPDF->AddPage();
  $MiPDF->SetFont('Times','',30);
  $MiPDF->SetY(40);
  $MiPDF->Cell(0,20,'Ejemplo número 3', 0, 0, 'C');
  $MiPDF->Ln();
  $MiPDF->SetFont('Times','',12);
  $MiPDF->MultiCell(0,5,$texto);
  $MiPDF->Ln();
  $MiPDF->Output('ejemplo3.pdf', 'D');
?>
```

Imagen y texto alrededor.

Una vez terminado el documento, dinámica o estáticamente, solo queda proceder a su generación. Para ello contamos con un método muy versátil de FPDF: *Output(Nombre, Destino)*. Como el documento PDF lo estamos generando al vuelo con FPDF, no existe aún y por tanto no tiene nombre, así que el primer parámetro, *Nombre*, permite especificar cómo queremos que se llame al documento generado. En cuanto a *Destino*, permite decir hacia dónde irá el documento que se generará, es un carácter y puede tener distintos valores. 'I' indica a FPDF que envíe el documento al navegador directamente sin que éste muestre la opción de guardarlo, sino que lo abra directamente con el plugin correspondiente; cuando el usuario seleccione la opción de *guardar*, aparecerá por defecto el nombre que se haya especificado en el parámetro *Nombre*. 'D' indica a FPDF que envíe el PDF generado al navegador pero sin que se abra, es decir, obliga al navegador a mostrar el cuadro de diálogo para guardar el fichero, con el nombre especificado. 'F' hace que FPDF almacene en local el fichero PDF, con el nombre especificado. Esto es especialmente bueno si no se está usando PHP para una aplicación web sino para realizar scripts en línea de comandos. Y por último 'S', que hace que FPDF ignore el nombre especificado y devuelva el fichero PDF como una cadena de caracteres.

EJEMPLO COMPLETO

Para ver cómo funciona todo el conjunto, el Listado 4 realiza un recopilatorio de todos los métodos vistos y usa, además, alguno que no se han explicado en este artículo. No obstante, el lector con ayuda del manual en línea de FPDF, que viene incorporado en el fichero que se puede descargar de la web del proyecto, y con el código fuente del ejemplo, que se encuentra en el CD-ROM de la revista, entenderá sin problemas el funcionamiento de esos métodos.

Lo curioso de este ejemplo está en ver cómo se conjuntan todos los métodos que hemos visto para que se pueda insertar una imagen y que quede el texto rodeándola, o por ejemplo, cómo insertar números de página, tanto actuales como totales.

CONCLUSIÓN

Hay más librerías para generación de PDFs desde PHP. Sin embargo, suelen ser complejas de utilizar, aunque ofrezcan muchas más posibilidades, o bien no son libres. Para casi todos los casos en que debemos generar un PDF con un listado, o por ejemplo con datos de una base de datos, FPDF es una opción estupenda puesto que su aprendizaje es sencillo, es muy ligera y permite realizar casi todo con pocas líneas de código. Una vez más, el software libre nos ofrece herramientas para crear aplicaciones rápidamente, y con la comodidad de que disponemos del código para saber en todo momento qué estamos haciendo y si es necesario, mejorarlo o adaptarlo.

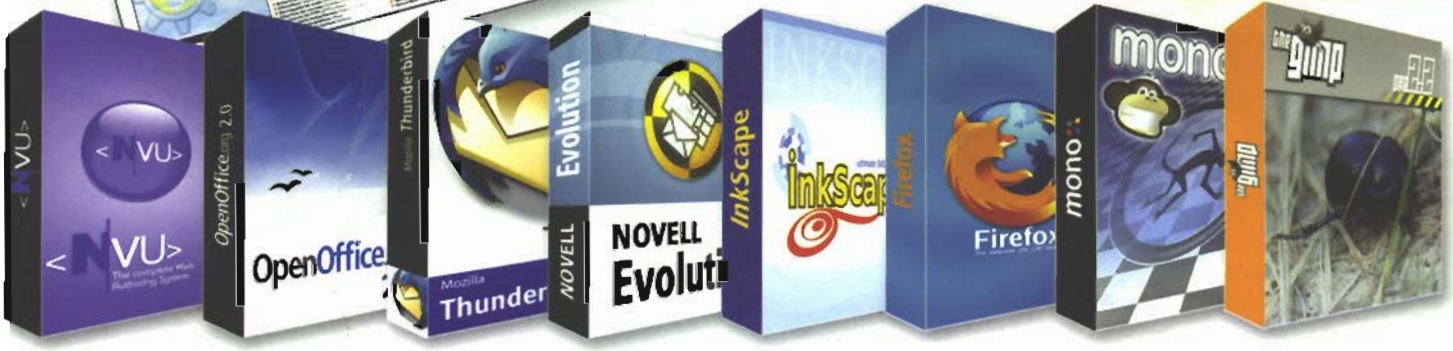
Listado 4

```
<?php
  define('FPDF_FONTPATH','../fpdf/font/');
  require('../fpdf/fpdf.php');
  class TP_FPDF extends FPDF {
    function TP_FPDF($orientacion='L', $unidades='mm', $formato='A4') {
      $this->FPDF($orientacion, $unidades, $formato);
    }
    function Header() {
      $this->SetY(20);
      $this->SetFont('Times','I',10);
     $this->Cell(0,10,'Taller: generación de PDF en PHP con FPDF. Ejemplo final',
'B',0,'R');
      $this->Ln();
    }
    function Footer() {
      $this->SetY(-20);
      $this->SetFont('Times','I',10);
     $this->Cell(0,10,'Página '.$this->PageNo().' de {PagTotales}','T',0,'R');
    }
  };
  $MiPDF=new TP_FPDF('P', 'mm', 'A4');
  $MiPDF->AliasNbPages('{PagTotales}');
  $MiPDF->AddPage();
  $MiPDF->SetFont('Times','',30);
  $MiPDF->SetY(40);
  $MiPDF->Cell(0,20,'Homo Antecesor', 0, 0, 'C');
  $MiPDF->Ln();
  $MiPDF->Image('homo_antecesor.png', 30, 60);
  $MiPDF->SetY(65);
  $MiPDF->SetX(120);
  $MiPDF->SetFont('Times','BI',12);
  $fichero=fopen('entradilla.txt','r');
  $texto=fread($fichero,filesize('fichero.txt'));
  fclose($fichero);
  $MiPDF->MultiCell(60,5,$texto);
  $MiPDF->Ln();
  $MiPDF->SetFont('Times','',12);
  $fichero=fopen('fichero.txt','r');
  $texto=fread($fichero,filesize('fichero.txt'));
  fclose($fichero);
  $MiPDF->MultiCell(0,5,$texto);
  $MiPDF->Ln();
  $MiPDF->Output();
?>
```

1ª entrega, DVD-ROM y Presentación de la obra por solo 5,99 euros

USUARIO LINUX

- *Usuario Linux es una obra compuesta por doce entregas de periodicidad semanal, cuyo objetivo es formar al lector en el manejo del sistema Linux y sus principales aplicaciones*
- *Aprende a trabajar de manera práctica, con tutoriales guiados y explicaciones sencillas, con el sistema operativo de mayor futuro, la plataforma que ya utilizan las empresas*
- *Linux es el sistema más moderno y eficiente para PC, con él olvídate de cuelgues, virus, spyware y demás problemas*
- *Con el número 1 se incluye un DVD con Debian Linux Sarge, un sistema Linux completo y en español que incluye más de 7000 aplicaciones.*



NVU	OpenOffice.org	Thunderbird	Evolution	Inkscape	Firefox	Mono	Gimp
Diseño Web	Suite de Ofimática	Gestión de Correo	Suite de Comunicaciones	Autoedición de Textos	Navegador Web	Desarrollo .NET	Tratamiento y y Retoque Digital

Composición de la obra

12 coleccionables divididos en:

Teoría
Conoce el sistema Linux a nivel profesional, cómo se instala en un PC, cómo se administra y cómo se trabaja con los principales entornos gráficos.

Software
Descubre todas las aplicaciones libres que hay para Linux, desde la suite de ofimática OpenOffice.org a las herramientas de desarrollo compatibles con .NET.

Tutorial Práctico
Aprende lo que realmente se necesita hoy en día: Cómo montar un servidor web, cómo configurar una red local, cómo programar una web dinámica o cómo instalar un cortafuegos profesional.

Oferta de suscripción

Usuario Linux está disponible en tu quiosco, pero también puedes suscribirte directamente llamando al **91 628 02 03** y recibir la obra en tres entregas mensuales por solo **66,°°** euros.

Además conseguirás de **regalo** la **tapa para encuadernar** la obra y una minisuscripción de tres números a la revista **Todo Linux.**

Studio PRESS
C/ del Río Ter, Nave 13 • Polígono Industrial "El Nogal" • 28110 Algete (Madrid) • Tel.: 916280203 • Fax: 916280935
e-mail: usuario@liberprensa.com • www.studiopress.es

ya a la venta en quioscos y centros comerciales