# Programación Hipermedia I

Práctica 15: PHP 8 (PDF)

## 1. Objetivos

- Aprender a crear un documento PDF.

## 2. Recursos

¿Cómo se genera un documento PDF en PHP?

- **FPDF**[1]: librería gratuita que genera documentos PDF.
- **How to Generate a PDF With PHP**[2]: pequeño tutorial que explica cómo usar FPDF.

## 3. ¿Qué tengo que hacer?

En esta práctica tienes que añadir a la parte privada de la aplicación una opción para que un usuario se pueda descargar un documento PDF con las imágenes que contiene uno de sus álbumes. El propósito es poder tener una copia estática de un álbum y poderla compartir con otros usuarios.

El álbum tiene que cumplir el siguiente formato:

1. Primera página: tiene que aparecer el título del álbum, la fecha del álbum, el nombre del usuario y la fecha de creación del documento PDF.
2. Segunda página: tiene que aparecer el título, la descripción, la fecha y el país del álbum, junto con un mensaje que indique el número de fotografías que componen el álbum.
3. Siguientes páginas: tiene que aparecer la fotografía, el título, la descripción y la fecha de la fotografía. Pueden aparecer varias fotografías por página con diferentes diseños (diferentes tamaños, diferentes alineaciones).
4. Última página: tiene que aparecer una lista con el título de todas las fotografías del álbum.

## 4. ¿Cómo lo hago?

Existen múltiples APIs para la creación de documentos PDF "al vuelo" (*on the fly*). PHP trae soporte para dos APIs, haru[3] y PDF[4], pero ambas requieren la instalación de bibliotecas adicionales, por lo que su uso no es fácil. En aquellas situaciones en las que se emplea un alojamiento (*hosting*) compartido, su uso puede no estar permitido.

Una de las APIs más potente es PDFlib[5], pero es de pago.

Una API que es muy sencilla de utilizar, instalar y que es gratuita es FPDF[6] (*Free PDF*). FPDF está liberado bajo una licencia permisiva: no hay restricción alguna de uso. Esto significa que se puede integrar libremente en cualquier aplicación (comercial o no), con o sin modificaciones.

La velocidad de generación de un documento con FPDF es menor que con otros sistemas, ya que es una API 100 % PHP. Sin embargo, esto sólo supone un problema cuando el documento a generar sea complejo y extenso.

[1] http://fpdf.org/
[2] http://answers.oreilly.com/topic/1414-how-to-generate-a-pdf-with-php/
[3] http://www.php.net/manual/es/book.haru.php
[4] http://www.php.net/manual/es/book.pdf.php
[5] http://www.pdflib.com/
[6] http://fpdf.org/

En principio, FPDF no tiene ningún límite respecto al tamaño de los ficheros PDF que puede generar, pero existen ciertas limitaciones de funcionamiento que pueden influir en su rendimiento:

- El máximo de memoria reservada para los scripts en PHP. En versiones anteriores a PHP 5.2 el límite estaba fijado en 8 MB, ahora son 128 MB. Para documentos muy extensos, especialmente si contienen imágenes, este límite puede alcanzarse (ya que FPDF construye el fichero en memoria). El límite se define con el valor `memory_limit`[7] en el fichero `php.ini`.
- El tiempo máximo de ejecución de un script PHP es por defecto 30 segundos. Este límite puede ser fácilmente sobrepasado cuando se genera un documento complejo y extenso. Este tiempo máximo también se define con el valor `max_execution_time`[8] en el fichero `php.ini` y puede ser modificado dinámicamente mediante la función de PHP `set_time_limit()`[9].
- Los navegadores tienen por lo general un límite máximo de 5 minutos de inactividad. Si se envía directamente el PDF al navegador y se sobrepasa este límite, se perderá el documento, ya que el navegador cerrará la conexión. Por lo tanto, en el caso de documentos muy grandes, se aconseja generarlos en un fichero y, cuando el documento esté terminado, redireccionar al usuario al documento generado con JavaScript o con un enlace.

## 4.1. Instalación y configuración

FPDF no necesita ningún tipo de instalación o configuración. Sólo hay que descargar el fichero comprimido con la última versión, descomprimirlo en un directorio que sea accesible desde PHP e incluir el fichero principal cuando se vaya a utilizar:

```
require("fpdf17/fpdf.php");
```

Del contenido de FPDF sólo son necesarios el fichero `fpdf.php` y el directorio `font`. El resto de ficheros y directorios no son necesarios para el funcionamiento de esta API.

## 4.2. Creación un documento sencillo

Un documento PDF se crea a partir de la clase `FPDF`. El constructor de esta clase tiene tres parámetros opcionales:

- La orientación de la página:
  - `P` o `Portrait`: vertical (orientación por defecto).
  - `L` o `Landscape`: horizontal (apaisado).
- La unidad de medida usada:
  - `pt`: punto.
  - `mm`: milímetro (unidad por defecto).
  - `cm`: centímetro.
  - `in`: pulgada.
- El tamaño de la página:
  - `A3`.
  - `A4` (tamaño por defecto).
  - `A5`.
  - `Letter`.
  - `Legal`.
  - Un array, con el ancho y el alto (expresado en la unidad definida).

[7] http://www.php.net/manual/es/ini.core.php#ini.memory-limit
[8] http://www.php.net/manual/es/info.configuration.php#ini.max-execution-time
[9] http://php.net/manual/es/function.set-time-limit.php

Para fijar los márgenes de una página se pueden usar los siguientes métodos:

- `SetMargins(float left, float top [, float right])`: fija el margen izquierdo, superior y derecho. El valor por defecto es 1cm.
- `SetLeftMargin(float margin)`: fija el margen izquierdo.
- `SetTopMargin(float margin)`: fija el margen superior.
- `SetRightMargin(float margin)`: fija el margen derecho.

Para definir el estilo del texto se emplea el método `SetFont()` que tiene tres parámetros:

- La familia (el tipo de letra o fuente). Puede ser una de las familias que están definidas en FPDF, o se puede añadir una familia nueva con el método `AddFont()`. Las familias predefinidas son:
  - `Arial`.
  - `Courier`.
  - `Helvetica`.
  - `Times`.
  - `Symbol`.
  - `ZapfDingbats`.
- El estilo de la fuente. Los estilos se pueden combinar entre sí. Los valores posibles son:
  - Cadena vacía: normal.
  - `B`: negrita (*bold*).
  - `I`: cursiva (*italic*).
  - `U`: subrayado (*underline*).
- El tamaño de la fuente en puntos. El valor por defecto es el tamaño actual. Si ningún tamaño ha sido especificado desde el comienzo del documento, toma el valor de 12.

Para añadir una página al documento se emplea el método `AddPage()`. Este método tiene dos parámetros opcionales que permiten indicar la orientación y el tamaño de la página, si no se indican, se emplean los valores definidos en el constructor.

Para escribir texto existe cuatro métodos:

- `Cell()`: muestra una celda (un área rectangular) con texto y con bordes opcionales. El texto puede estar alineado (izquierda o derecha) o centrado.
- `MultiCell()`: similar a `Cell()`, pero permite escribir texto con saltos de línea. Los saltos de línea pueden ser automáticos (al alcanzar el borde derecho de la celda) o explícitos (señalados con el carácter de salto de línea `\n`). El texto puede estar alineado (izquierda o derecha), centrado o justificado.
- `Text()`: escribe una cadena de texto en la posición indicada.
- `Write()`: escribe una cadena de texto con saltos de línea en la posición actual.

El método `Ln(altura)` genera un salto de línea de la altura indicada.

Finalmente, para generar el documento PDF y enviarlo al navegador se debe invocar el método `Output()` que tiene dos parámetros opcionales:

- El nombre del fichero que se genera.
- El destino del documento, puede tomar uno de los siguientes valores:
  - `I`: envía el documento en forma de fichero al navegador para que éste lo intente visualizar.
  - `D`: envía el documento en forma de fichero al navegador y fuerza su descarga.
  - `F`: guarda el documento como un fichero en el servidor.

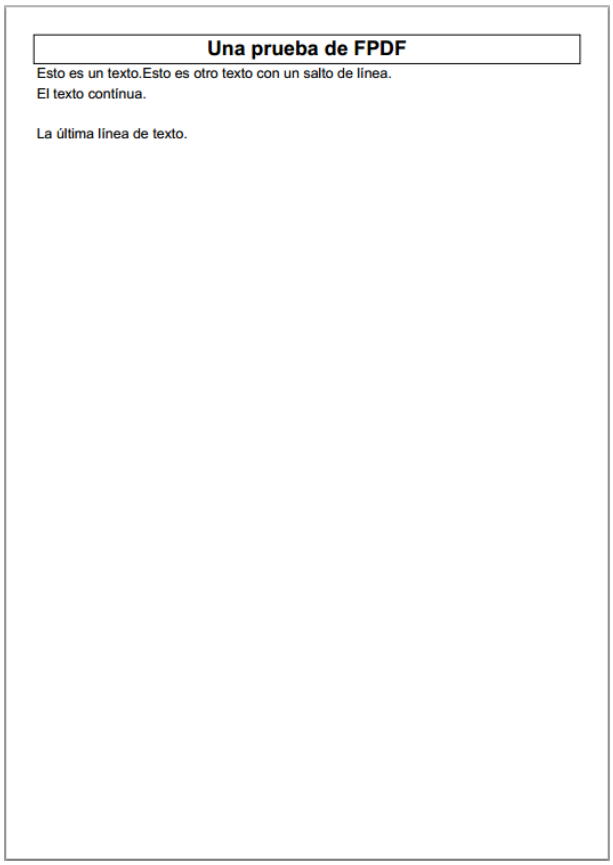


Figura 1: Ejemplo de un documento sencillo

- `S`: devuelve el documento como una cadena.

El ejemplo siguiente crea un documento PDF sencillo compuesto de una página con una celda y con un fragmento de texto; para escribir el texto se utilizan los métodos `Cell()` y `Write()`:

```
<?php
require("fpdf17/fpdf.php");

$pdf = new FPDF();

$pdf->AddPage();

$pdf->SetFont("Arial", "B", 20);
$pdf->Cell(0, 10, "Una prueba de FPDF", 1, 1, "C");

$pdf->SetFont("Arial", "", 14);
$pdf->Write(7, "Esto es un texto.");
$pdf->Write(7, "Esto es otro texto con un salto de línea.\nEl texto contínua.");
$pdf->Ln(7);
$pdf->Ln(7);
$pdf->Write(7, "La última línea de texto.");

$pdf->Output();
?>
```

En la Figura 1 se puede ver el resultado de este ejemplo.

## 4.3. Metadatos

En FPDF existen cinco métodos para definir los metadatos de un documento PDF:

- `SetTitle()`: define el título del documento.
- `SetAuthor()`: define el autor del documento.
- `SetSubject()`: define el asunto del documento.
- `SetKeywords()`: define las palabras clave del documento.
- `SetCreator()`: define la aplicación de creación del documento.

El siguiente ejemplo crea un documento con todos sus metadatos; en este ejemplo, al generar el documento PDF se ha optado por forzar la descarga del documento con el nombre `metadatos.pdf`:

```
<?php
require("fpdf17/fpdf.php");

$pdf = new FPDF();

$pdf->SetTitle("Un documento de prueba");
$pdf->SetAuthor("Sergio Luján Mora");
$pdf->SetCreator("FPDF y PHP");
$pdf->SetSubject("Un documento de prueba creado con FPDF desde PHP");
$pdf->SetKeywords("prueba FPDF PHP");

$pdf->AddPage();
$pdf->SetFont('Arial', 'B', 16);
$pdf->Cell(0, 10, 'Esto es una prueba...');
$pdf->Output("metadatos.pdf", "D");
?>
```

En la Figura 2 se puede ver cómo se visualizan los metadatos a través de la opción Propiedades en Adobe Acrobat:

## 4.4. Dibujar una línea

FPDF es un poco limitado a la hora de dibujar figuras geométricas, ya que sólo dispone del método `Line()` para dibujar líneas y el método `Rect()` para dibujar rectángulos:

- `SetDrawColor(rojo, verde, azul)`: define el color usado en las operaciones de dibujado, como por ejemplo, el dibujado de una línea.
- `SetLineWidth(ancho)`: define el ancho de la línea.
- `Line(x1, y1, x2, y2)`: dibuja una línea entre dos puntos.
- `SetFillColor(rojo, verde, azul)`: define el color usado en las operaciones de relleno, como por ejemplo, el dibujado de un rectángulo.
- `Rect(x, y, w, h, estilo)`: dibuja un rectángulo con borde, relleno (sin borde) o con ambos.

En el siguiente ejemplo, se dibujan dos líneas de color rojo en forma de aspa que unen las esquinas de la página. En las coordenadas para dibujar las líneas se ha respetado el margen por defecto de 1 cm que posee la página:

```
<?php
require("fpdf17/fpdf.php");

$pdf = new FPDF();
```

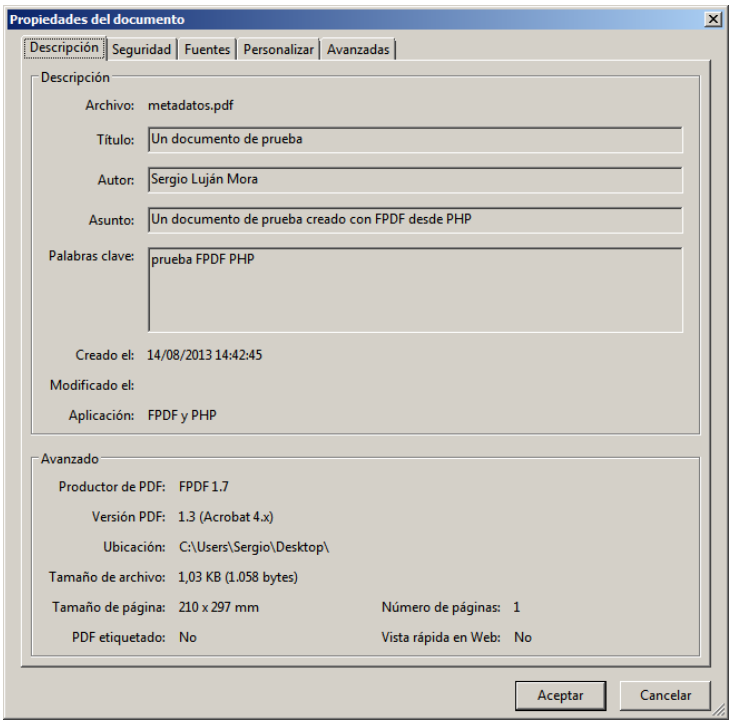


Figura 2: Visualización de metadatos en Adobe Acrobat

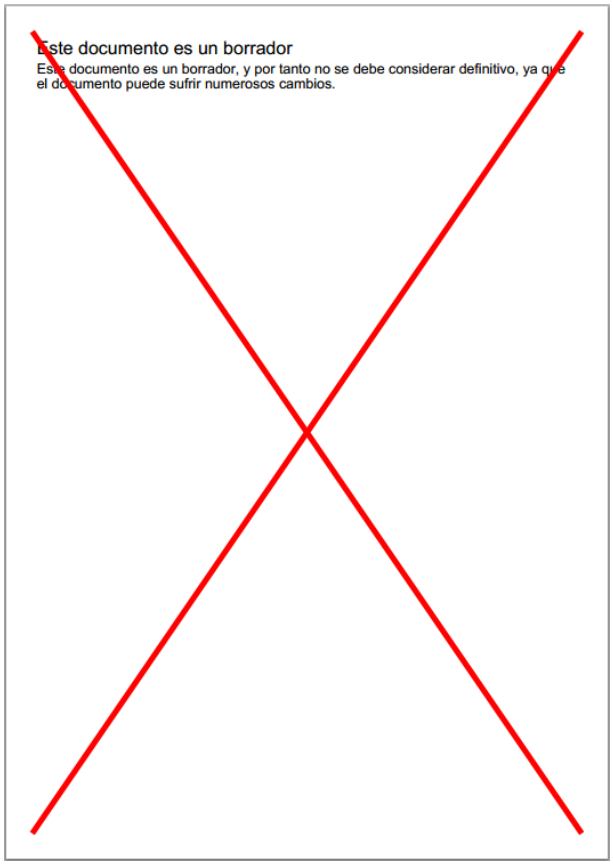


Figura 3: Ejemplo de un documento con líneas

```
$pdf->AddPage();

$pdf->SetFont('Arial', '', 18);
$pdf->Write(10, "Este documento es un borrador\n");
$pdf->SetFont('Arial', '', 14);
$pdf->Write(5, "Este documento es un borrador, y por tanto no se debe considerar
            definitivo, ya que el documento puede sufrir numerosos cambios.");

$pdf->SetDrawColor(255, 0, 0);
$pdf->SetLineWidth(2);
$pdf->Line(10, 10, 210 - 10, 297 - 10);
$pdf->Line(210 - 10, 10, 10, 297 - 10);

$pdf->Output();
?>
```

En la Figura 3 se puede ver el resultado de este ejemplo.

## 4.5. Cabecera y pie de página

FPDF permite generar de forma automática una cabecera y un pie de página. Para ello se tienen que sobreescribir los métodos `Header()` y `Footer()` que tiene definida la clase `FPDF`. Estos métodos están declarados, pero están vacíos.

Si se quiere una cabecera o pie de página distintos para la primera página, para las páginas pares (o impares), o para otro tipo de diseño, se puede invocar el método `PageNo()` para saber en qué página se está mostrando la cabecera o el pie de página.

Si se quiere mostrar el número total de páginas, se tiene que invocar el método `AliasNbPages(alias)` que genera un alias para el número total de páginas que será reemplazado por su valor real cuando el documento sea cerrado.

Por ejemplo, el siguiente código genera un documento PDF con dos páginas, con su cabecera y su pie de página:

```
<?php
require("fpdf17/fpdf.php");

class MyFPDF extends FPDF
{
  function Header()
  {
    $this->SetFont("Arial", 'B', 15);
    // Se desplaza a la derecha 6 cm para que la cabecera
    // aparezca centrada
    $this->Cell(60);
    $this->Cell(70, 10, "El título del documento", 1, 0, 'C');
    $this->Ln(20);
  }

  function Footer()
  {
    // El pie está situado a 1 cm del final de la página,
    // es decir, dentro del margen inferior
    $this->SetY(-10);
    $this->SetFont("Arial", "I", 8);
    $this->Cell(0, 10, "Página " . $this->PageNo() . "/{nb}", 0, 0, "C");
  }
}

$pdf = new MyFPDF();
$pdf->AliasNbPages();
```

```
$pdf->SetFont("Arial", "B", 16);

$pdf->AddPage();
$pdf->Write(8, "Esto es una prueba...");

$pdf->AddPage();
$pdf->Write(8, "Esto es una prueba...");

$pdf->Output();
?>
```

### 4.6. Inserción de una imagen

FPDF permite inserta una imagen en formato JPEG, PNG o GIF en un documento PDF. Para ello se emplea el método `Image()`.

En el siguiente ejemplo, se insertan dos imágenes con unos textos en un documento PDF:

```
<?php
require("fpdf17/fpdf.php");

$pdf = new FPDF();


$pdf->AddPage();
$pdf->SetFont('Arial', 'B', 24);
$pdf->Write(12, "Fotografías de montañas");
$pdf->Ln();
$pdf->SetFont('Arial', 'B', 16);
$pdf->Image("cervino.jpg", NULL, NULL, 100);
$pdf->Write(10, "El Cervino");
$pdf->Ln(20);
$pdf->Image("benasque.jpg", NULL, NULL, 100);
$pdf->Write(10, "Benasque");
$pdf->Output();
?>
```

En la Figura 4 se puede ver el resultado de este ejemplo.

## 5. Recomendaciones

Intenta que el álbum no sea simplemente una sucesión de fotografías y textos, intenta que tenga un estilo visual atractivo.

Si quieres modificar un fichero PDF ya existente, se puede hacer con FPDF, pero necesitas la extensión FPDI (Import existing PDF documents into FPDF)[10].

[10] http://www.setasign.de/products/pdf-php-solutions/fpdi/

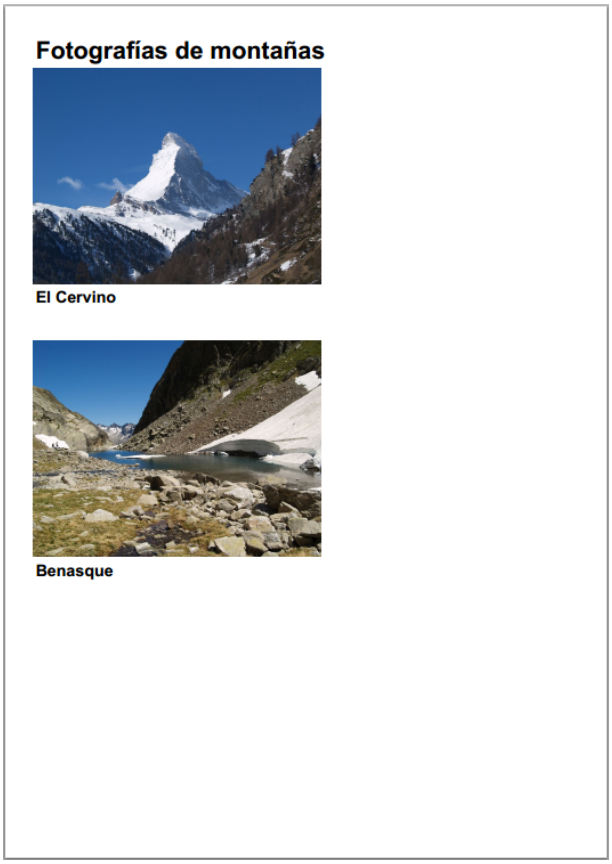


Figura 4: Ejemplo de un documento con dos imágenes