國立臺灣大學
計算機及資訊網路中心
電子報
第0015期 · 2010.12.20 發行
ISSN 2077-8813

技術論壇

使用ASP .NET (C#) 產生PDF檔的好幫手—iTextSharp library (上)

作者：唐瑤瑤 / 臺灣大學計算機及資訊網路中心程式設計組程式設計師

由於工作內容需要使用ASP .NET C#產生PDF檔，但是微軟的.NET framework 並沒有內建產生PDF 的功能，所以只能上網找Third-Party 提供的函式庫。請出Google大神幫忙，搜尋出來的結果有上萬筆，在沒有頭緒的情況下真是大海撈針。搜尋結果中有很多PDF 函式庫是要付費的，就先將它排除，我們將搜尋範圍縮小至free又能支援中文後，終於讓我找到一個很實用且容易上手的library — iTextSharp。

本函式庫原名是iText，主要是支援Java程式語言。之後針對Microsoft .NET C Sharp做了一個版本，也就是我們今天要介紹的iTextSharp。針對PDF檔案的製作與修改，支援的功能如下：

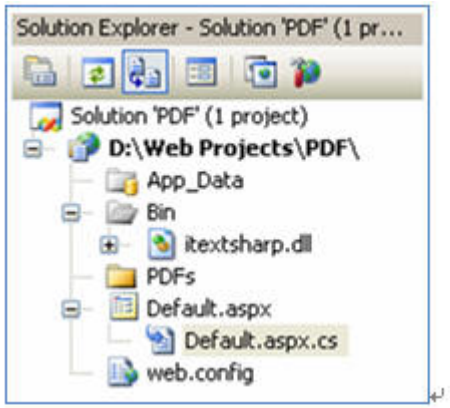
1. Create：Automate、Convert、Sign、Encrypt。
2. Read：Extract。
3. Update：Stamp、Fill out、Split/Merge、Convert、Sign、Encrypt。

接下來就讓我帶領大家一步一步用程式碼產生PDF 檔。

開發環境

1. 我的ASP .NET開發環境是Microsoft Visual Studio 2010 版，使用的程式語言是C#。
2. iTextSharp 目前版本是5.0.4，下載位置：http://sourceforge.net/projects/itextsharp/ 。請先下載zip，解壓縮後只有一個dll檔，利用Add Reference方式將itextsharp.dll 加入您微軟專案的Bin目錄中，就這麼簡單而且可以開始coding了。

Part 1：首先建立一個Web Application



所有的範例程式碼都會建立在這個 Default.aspx.cs 中。

先加入以下 references，因爲會使用到 memorystream，所以必須引用 System.IO：

```
using System.IO;
using iTextSharp.text;
using iTextSharp.text.pdf;
```

在這個範例中，我們利用web application在Server Memory產生pdf 檔後，使用者可以自行下載瀏覽或存檔，採用的是PdfWriter類別：

```
var doc1 = new Document(PageSize.A4, 50, 50, 80, 50);//設定Page size 及 Margin
left, right, top, bottom
MemoryStream Memory = new MemoryStream();
PdfWriter PdfWriter = PdfWriter.GetInstance(doc1, Memory);
```

若是要在Server FileSystem 先產生pdf檔，可以使用下列程式碼：

```
string path = Server.MapPath("pdf");
PdfWriter PdfWriter = PdfWriter.GetInstance(doc1, new FileStream(path +
"/pdfexample.pdf", FileMode.Create));
```

在PDF檔案內容中要顯示中文，最重要的是字型設定，如果沒有正確設定中文字型，會造成中文無法顯示的問題。首先設定基本字型：kaiu.ttf 是作業系統系統提供的標楷體字型，IDENTITY_H 是指編碼(The Unicode encoding with horizontal writing)，及是否要將字型嵌入 PDF 檔中。再來針對基本字型做變化，例如Font Size、粗體斜體以及顏色等。當然你也可以採用其他中文字體字型。

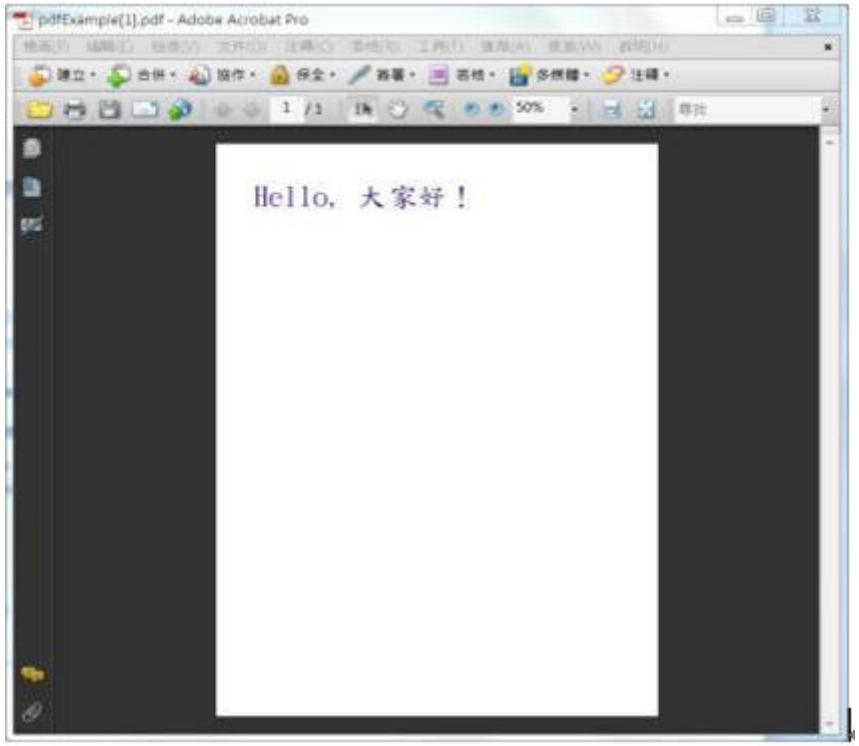
```
//字型設定
BaseFont bfChinese = BaseFont.CreateFont(@"C:\WINDOWS\Fonts\kaiu.ttf",
BaseFont.IDENTITY_H, BaseFont.NOT_EMBEDDED);
Font ChFont = new Font(bfChinese, 12);
Font ChFont_blue = new Font(bfChinese, 40, Font.NORMAL, new BaseColor(51,
0, 153));
Font ChFont_msg = new Font(bfChinese, 12, Font.ITALIC, BaseColor.RED);
```

開啟檔案寫入內容後，將檔案關閉。

```
doc1.Open();
// your PDF content
doc1.Add(new Paragraph(10f,"Hello, 大家好！",ChFont_blue));
doc1.Close();
// 若是要在client 端顯示PDF 檔，並讓 user下載，必須加上下面這一段程式碼，將檔案輸出至瀏
覽器端：
Response.Clear();
Response.AddHeader("Content-Disposition", "attachment;
filename=pdfExample.pdf");
Response.ContentType = "application/octet-stream";
Response.OutputStream.Write(Memory.GetBuffer(), 0,
Memory.GetBuffer().Length);
Response.OutputStream.Flush();
Response.OutputStream.Close();
Response.Flush();
Response.End();
```

到此階段，您已經產生了一個可以在瀏覽器端下載的檔案（檔名為pdfExample.pdf）。執行程式結果如下：

選擇開啟舊檔，顯示如下圖：

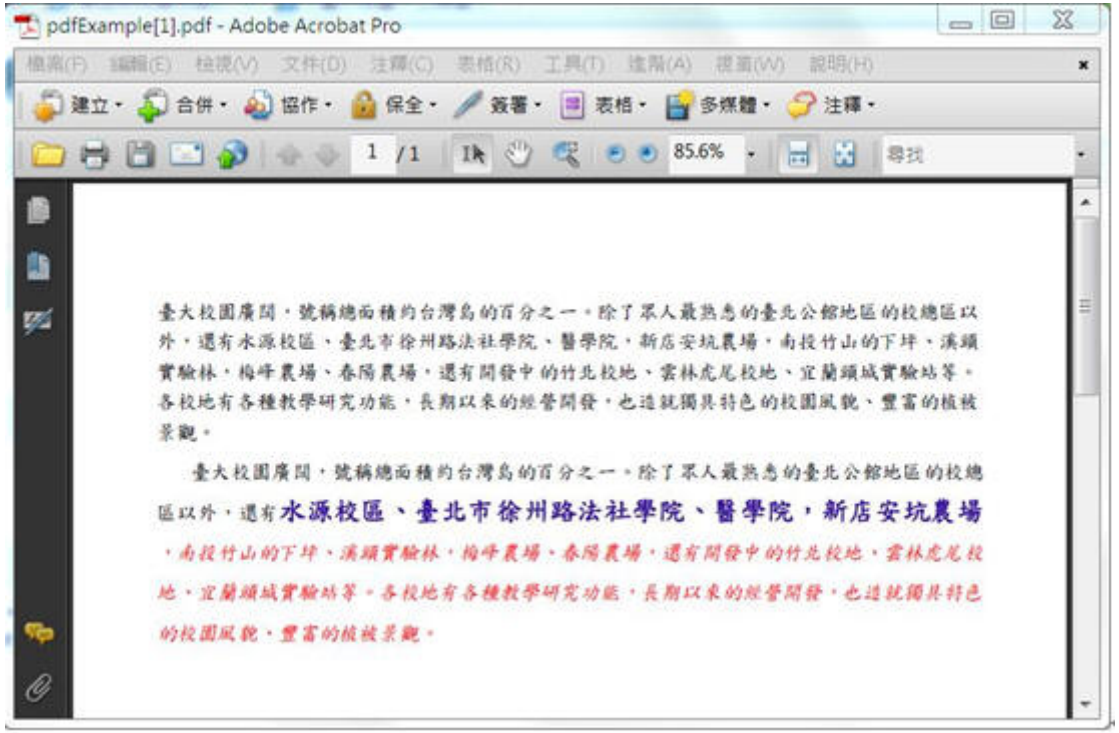


Part 2：使用**chunk**、**phrase**及**paragraph**建立文字段落內容

Paragraph是文章段落，可由phrases (句子)組成，然後phrase又可以由chunks(文字片段)所組成。這樣的組成方式可以很方便的在文字段落中穿插不同的字型樣式。但要注意的是chunk不會自動換行，必須自行插入換行符號”\n” 或是使用`Environment.NewLine`。至於Paragraph有許多樣式可以設定，像是Alignment、indentation、leading及spacing 等。

```
//Chunk, Phrase, Paragraph
Chunk c = new Chunk("臺大校園廣闊，號稱總面積約台灣島的百分之一。除了眾人最熟悉的臺北公館地區的校總區以外，還有水源校區、臺北市徐州路法社學院、醫學院，新店安坑農場，南投竹山的下坪、溪頭實驗林，梅峰農場、春陽農場，還有開發中的竹北校地、雲林虎尾校地、宜蘭頭城實驗站等。各校地有各種教學研究功能，長期以來的經營開發，也造就獨具特色的校園風貌、豐富的植被景觀。",
ChFont);
Phrase p1 = new Phrase(c);
doc1.Add(p1);

Chunk c1 = new Chunk("臺大校園廣闊，號稱總面積約台灣島的百分之一。除了眾人最熟悉的臺北公館地區的校總區以外，還有", ChFont);
Chunk c2 = new Chunk("水源校區、臺北市徐州路法社學院、醫學院，新店安坑農場",
ChFont_bold_blue);
Chunk c3 = new Chunk("，南投竹山的下坪、溪頭實驗林，梅峰農場、春陽農場，還有開發中的竹北校地、雲林虎尾校地、宜蘭頭城實驗站等。各校地有各種教學研究功能，長期以來的經營開發，也造就獨具特色的校園風貌、豐富的植被景觀。", ChFont_msg);
Phrase p2 = new Phrase();
p2.Add(c1);
p2.Add(c2);
p2.Add(c3);
Paragraph pg = new Paragraph(p2);
pg.SetAlignment("Justify");            //左右對齊
pg.FirstLineIndent = 20f;              //段落句首縮排
pg.SetLeading(0.0f,2.0f);              //設定行距
doc1.Add(pg);
```

執行結果如下：



Part 3：產生表格

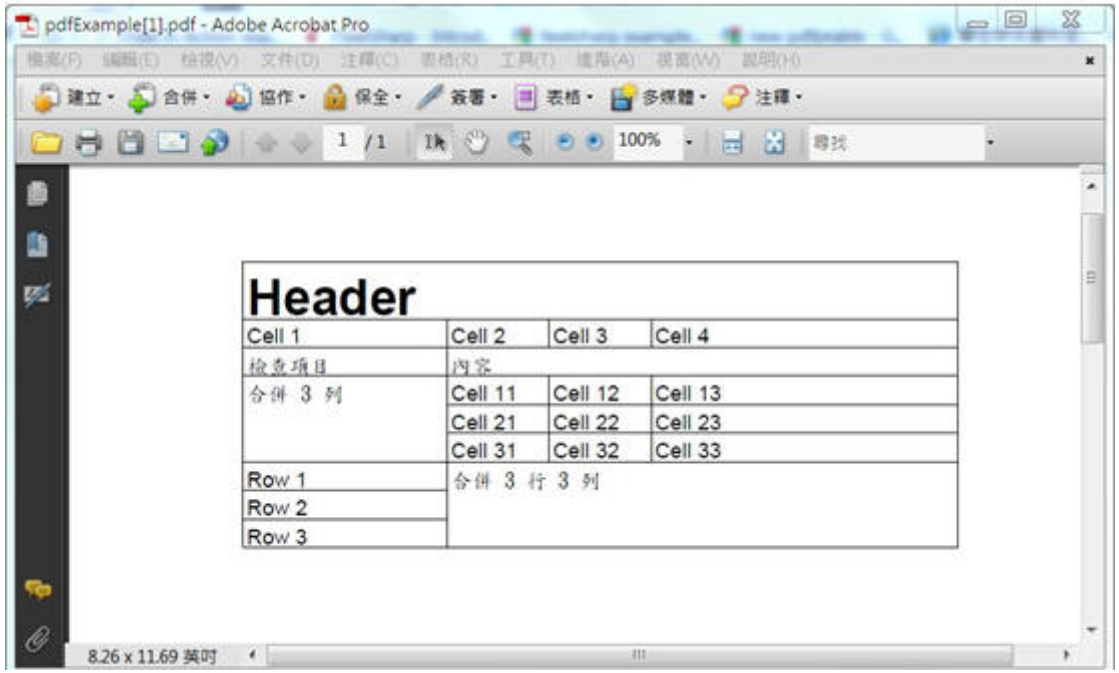
使用 iTextSharp 產生表格是十分直覺且容易的，類似CSS的寫法。在建立table 時，可以很輕易用欄位相對寬度做設定，也可以給絕對寬度，或者單純的給予欄位數做平均等分，ex: `PdfPTable table = new PdfPTable(4);`。

比較值得注意的是，由於PdfPTable表格裡面，每一格叫做cell，因此在塞資料時，必須注意填寫方式是由左而右、由上而下。此外，PdfpCell 有合併欄位的功能Colspan，也有合併列的功能Rowspan，我們可以利用這兩項特性將平淡的表格做些變化。請參考下面的範例：

```
//create table with relative width of 4 columns
PdfPTable table = new PdfPTable(new float[] { 2, 1, 1, 3 });
//actual width of table in points
table.TotalWidth = 400f;
//fix the absolute width of the table
table.LockedWidth = true;
PdfPCell header = new PdfPCell(new Phrase("Header", new
Font(Font.FontFamily.HELVETICA, 28f, Font.BOLD)));
header.Colspan = 4;
table.AddCell(header);
table.AddCell("Cell 1");
table.AddCell("Cell 2");
table.AddCell("Cell 3");
table.AddCell("Cell 4");
PdfPCell itemname = new PdfPCell(new Phrase("檢查項目", ChFont));
itemname.Colspan = 1;
table.AddCell(itemname);
PdfPCell content = new PdfPCell(new Phrase("內容", ChFont));
content.Colspan = 3;
table.AddCell(content);
PdfPCell rows = new PdfPCell(new Phrase("合併  3 列", ChFont));
rows.Rowspan = 3;
table.AddCell(rows);
for (int i = 1; i <= 3; i++)
          {
              table.AddCell("Cell "+i.ToString()+"1");
              table.AddCell("Cell " + i.ToString() + "2");
              table.AddCell("Cell " + i.ToString() + "3");
          }
table.AddCell("Row 1");
```

```
PdfPCell row = new PdfPCell(new Phrase("合併 3 行 3 列", ChFont));
row.Rowspan = 3;
row.Colspan = 3;
table.AddCell(row);
table.AddCell("Row 2");
table.AddCell("Row 3");
doc1.Add(table);
```

執行結果如下圖：



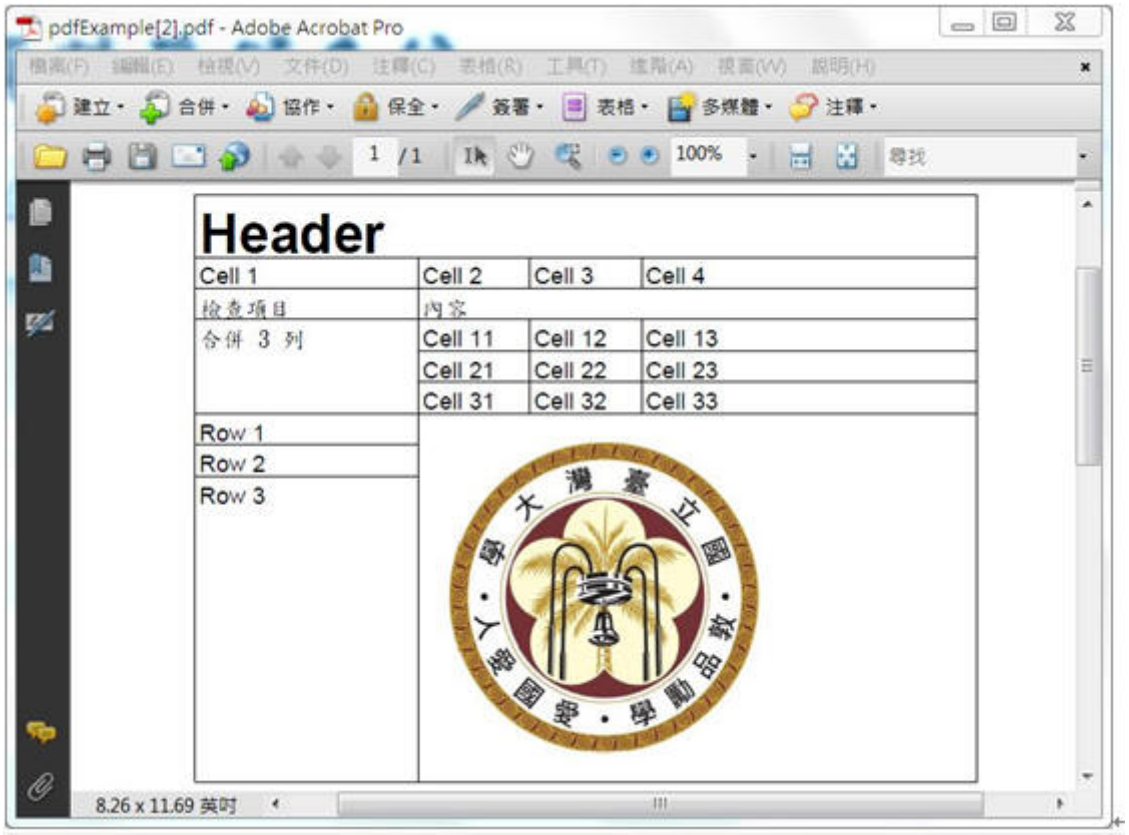
Part 4：插入圖片

iTextSharp 支援的image type包括：JPEG, JPEG2000, GIF, PNG, BMP, WMF, TIFF, and JBIG2。除了可以讀取伺服器本地端圖檔外，也可以直接指定圖檔的URL位置。我們延續表格

的範例實作如下：

```
//取得圖檔的URL
string image_url = "http://www.ntu.edu.tw/about/Emblem72.jpg";
iTextSharp.text.Image jpg = iTextSharp.text.Image.GetInstance(new
Uri(image_url));
//設定圖檔的縮放大小
jpg.ScaleToFit(160f, 160f);

PdfPCell jpgcell = new PdfPCell(jpg);
jpgcell.Colspan = 3;
jpgcell.Rowspan = 3;
jpgcell.BackgroundColor = new BaseColor(255, 255, 255);
jpgcell.Padding = 15f;
table.AddCell(jpgcell);
```

執行結果如下圖：



Part 5：書籤

iTextSharp 是透過Chapter類別及Section類別來產生樹狀結構的書籤功能。最上層的書籤必須是Chapter，且每個Chapter之開始必定是一個New Page，就是會自動換頁的意思。Section則須在Chapter 物件中，或是在另一個Section物件中，無法單獨存在。

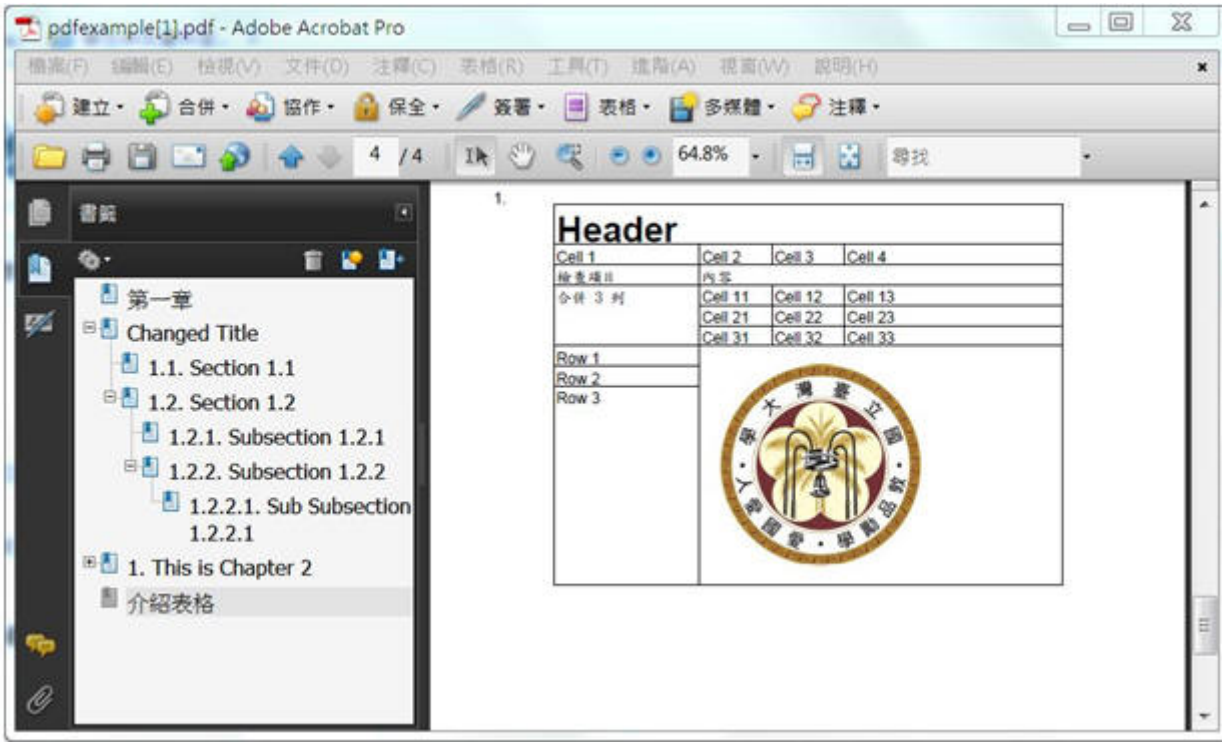
```
Chapter chapter1 = new Chapter(new Paragraph("This is Chapter 1"), 1);
Section section1 = chapter1.AddSection(20f, "Section 1.1", 2);
Section section2 = chapter1.AddSection(20f, "Section 1.2", 2);
Section subsection1 = section2.AddSection(20f, "Subsection 1.2.1", 3);
Section subsection2 = section2.AddSection(20f, "Subsection 1.2.2", 3);
Section subsubsection = subsection2.AddSection(20f, "Sub Subsection
1.2.2.1", 4);
```

```
Chapter chapter2 = new Chapter(new Paragraph("This is Chapter 2"), 1);
Section section3 = chapter2.AddSection("Section 2.1", 2);
Section subsection3 = section3.AddSection("Subsection 2.1.1", 3);
Section section4 = chapter2.AddSection("Section 2.2", 2);

chapter0.BookmarkTitle = "第一章";
chapter1.BookmarkTitle = "Changed Title";
chapter1.BookmarkOpen = true;
chapter2.BookmarkOpen = false;
doc1.Add(chapter0);
doc1.Add(chapter1);
doc1.Add(chapter2);

Chapter chapter3 = new Chapter(new Paragraph("介紹表格"), 1);
chapter3.BookmarkTitle = "介紹表格";

//省略建立表格內容

chapter3.Add(table);
doc1.Add(chapter3);
```

若要使用者一開啟PDF檔案時，書籤列就自動展開，請加上這一行程式碼：

```
PdfWriter.ViewerPreferences =  PdfWriter.PageModeUseOutlines;
```

執行結果如下圖：



Part 6：浮水印

iTextSharp 產生浮水印(Watermark)的方式是使用PdfStamper 類別，且必須是針對已經產生之pdf檔做加工。在我們的範例中，請先產生一個儲存在server 端的檔案，再經由PdfReader

物件讀進來處理。

```
PdfReader reader = new PdfReader(path + "/" + fname + ".pdf");
using (MemoryStream memoryStream = new MemoryStream())
{
PdfStamper pdfStamper = new PdfStamper(reader, memoryStream);
for (int i = 1; i <= reader.NumberOfPages; i++) // Must start at 1 because
0 is not an actual page.
    {
    Rectangle pageSize = reader.GetPageSizeWithRotation(i);
    PdfContentByte pdfPageContents = pdfStamper.GetOverContent(i);//or
GetUnderContent(i);
    pdfPageContents.BeginText(); // Start working with text.
    pdfPageContents.SetFontAndSize(bfChinese, 80); // 80 point font
    pdfPageContents.SetRGBColorFill(192, 192, 192); // Sets the color of
the font, RED in this instance
    float textAngle = 45.0f;
    pdfPageContents.ShowTextAligned(PdfContentByte.ALIGN_CENTER, "機密資
料", pageSize.Width / 2, pageSize.Height / 2 + 120f, textAngle);
    pdfPageContents.ShowTextAligned(PdfContentByte.ALIGN_CENTER, "僅供"
+ text + "使用", pageSize.Width / 2, pageSize.Height / 2, textAngle);
    pdfPageContents.EndText(); // Done working with text
    }
pdfStamper.FormFlattening = true; // enable this if you want the PDF
flattened.
pdfStamper.Close(); // Always close the stamper or you'll have a 0 byte
stream.
reader.Close();
}
```

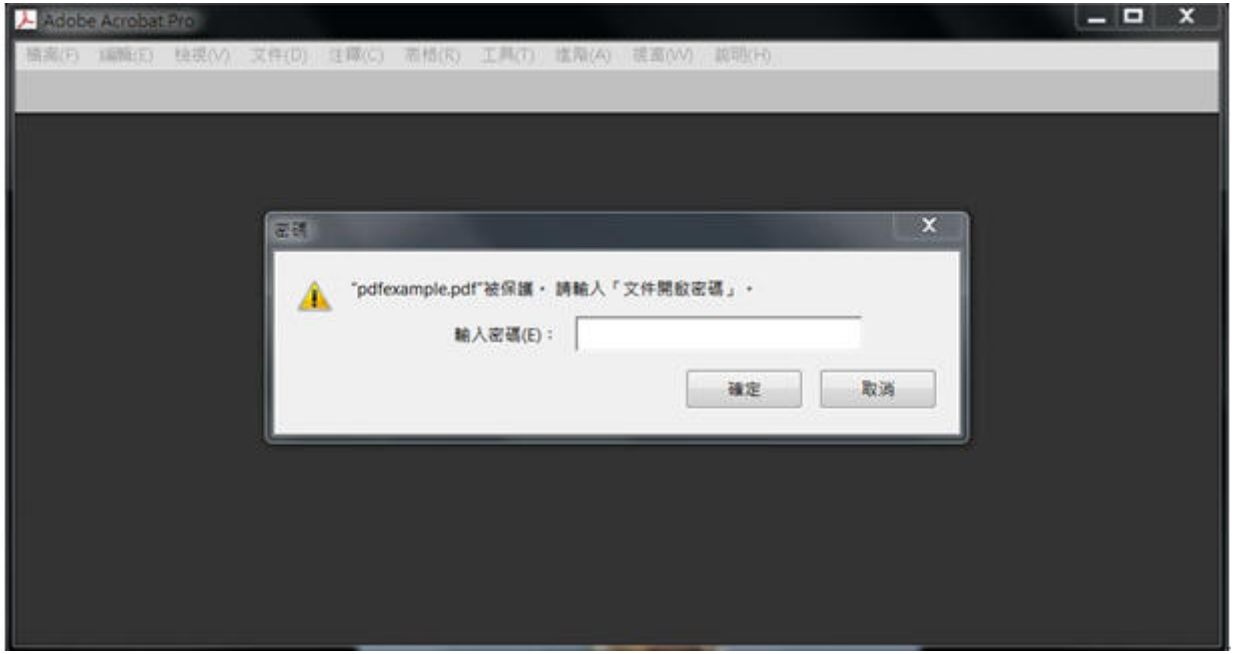
執行結果如下圖：

Part 7：文章加密及保護

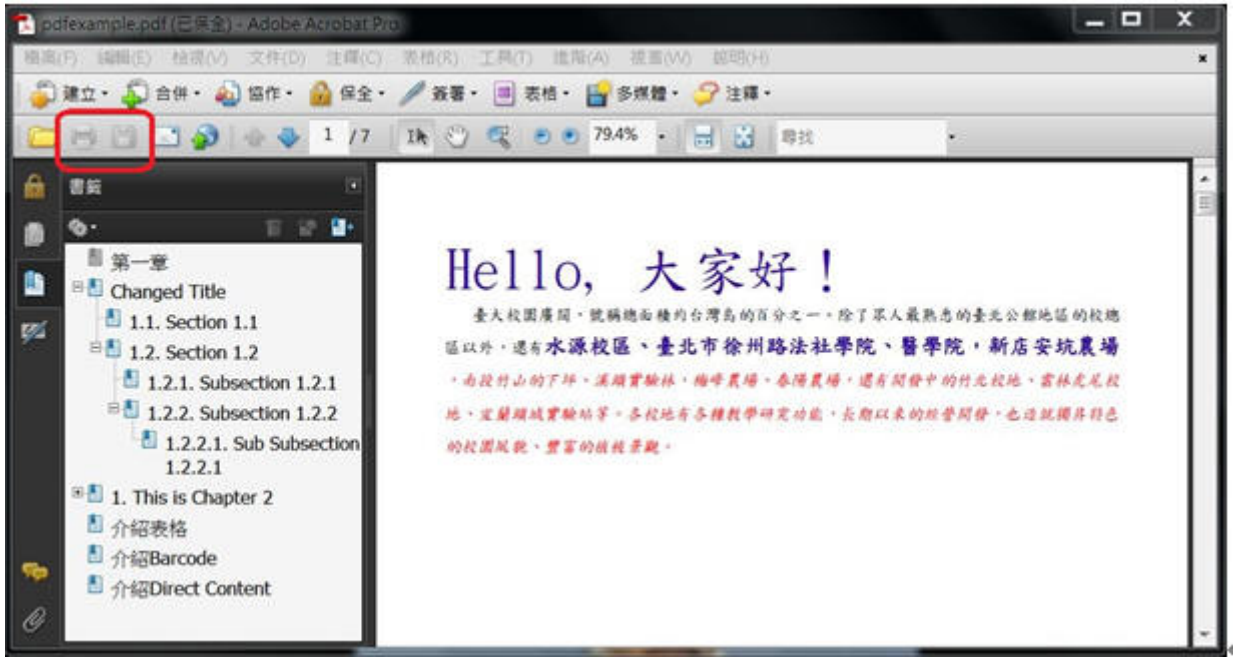
若要使用者必須輸入密碼才能開啟pdf檔，iTextSharp的做法是針對已經產生之pdf檔做加工。在我們的範例中，請先產生一個儲存在server 端的檔案，再經由PdfReader物件讀進來處理。一切安全性設定都是使用PdfEncryptor 物件。

```
PdfEncryptor .Encrypt (PdfReader reader, Stream os, bool strength, string
userPassword, string ownerPassword, int permissions);
```

上述主要函式中，可以設定user password 及owner password。一旦設定後，每次打開PDF 檔時都必須輸入密碼。User password 可以讀檔，但不能列印及修改儲存；owner password 則具有全部權限。



下圖可以看到列印及存檔的圖示都被disable了


pdfexample.pdf (已保全) - Adobe Acrobat Pro
書籤
第一章
Changed Title
1.1. Section 1.1
1.2. Section 1.2
1.2.1. Subsection 1.2.1
1.2.2. Subsection 1.2.2
1.2.2.1. Sub Subsection 1.2.2.1
1. This is Chapter 2
介紹表格
介紹Barcode
介紹Direct Content
Hello, 大家好！

```
void EncryptPDF(string fname, string pwd)
{
    string path = Server.MapPath("App_Data");
    PdfReader reader = new PdfReader(path + "/" + fname + ".pdf");

    using (MemoryStream memoryStream = new MemoryStream())
    {
        PdfEncryptor.Encrypt(reader, memoryStream, true, pwd,
"ownerpwd", 1);
        reader.Close();
        Response.Clear();
        Response.AddHeader("Content-Disposition", "attachment;
filename=" + fname + ".pdf");
        Response.ContentType = "application/octet-stream";
        Response.OutputStream.Write(memoryStream.GetBuffer(), 0,
memoryStream.GetBuffer().Length);
        Response.OutputStream.Flush();
        Response.OutputStream.Close();
        Response.Flush();
        Response.End();
    }
}
```

如果想要隱藏Menu Bar 及工具列，只要簡單的設定PdfWriter.ViewerPreferences，這在之前介紹自動開啟書籤的做法時有使用過。

```
PdfWriter.ViewerPreferences =  PdfWriter.HideMenubar |
PdfWriter.HideToolbar;
```

針對被保護的文件，如果想開放某些權限，例如讓使用者能列印，做法是在PdfEncryptor 物件中設定如下：

```
PdfEncryptor .Encrypt(reader,  memoryStream, null,null,
PdfWriter .AllowAssembly  |PdfWriter .AllowFillIn|
PdfWriter .AllowScreenReaders| PdfWriter .AllowPrinting, false);
```

Part 8：Metadata

我們可以在PDF 文件的metadata 中寫入一些值，但為防止被修改，建議搭配PdfEncryptor 物件使用。

```
doc1.AddTitle( "介紹DPDF");
doc1.AddAuthor( "Dana Tang" );
doc1.AddSubject( "This example  shows how to add metadata" );
doc1.AddKeywords( "Metadata,  iTextSharp,  PDF");
doc1.AddCreator( "Using  iTextSharp" );
doc1.Open();
```

開啟PDF檔，選擇工具列【檔案】→【內容】，即會看到如下圖所顯示：


文件內容
描述
保全
字型
初始視圖
自訂
進階
描述
檔案： pdfexample[2]
標題(T)： 介紹PDF
作者(A)： Dana Tang
主題(S)： This example shows how to add metadata
關鍵字(K)： "Metadata, iTextSharp, PDF"
建立日期： 2010/11/1 下午 03:44:50
其它元資料(M)...
修改日期： 2010/11/1 下午 03:44:51
應用程式： Using iTextSharp
進階
PDF 製作程式： iTextSharp 5.0.4 (c) 1T3XT BVBA
PDF 版本： 1.4 (Acrobat 5.x)
位置： E:\IE Temp\Temporary Internet Files\Low\Content.IE5\MIOJSY89\
檔案大小： 1.61 MB (1,692,260 位元組)
頁面大小： 8.26 x 11.69 英吋
頁數： 7
標籤化 PDF： 否
快速 Web 檢視： 否
說明
確定
取消

大功告成！是不是覺得iTextSharp 功能很強大？我們現在已經可以產生一個專業級的PDF檔。事實上，iTextSharp還有許多進階功能，像是Header、Footer、Barcode、Direct content及JavaScript等互動功能，就留待下次再介紹給各位。

參考資料

1.主要教學網站
http://itextpdf.com/index.php http://www.mikesdotnetting.com/Article/80/Create-PDFs-in-ASP.NET-getting-started-with-iTextSharp
2.浮水印Watermark製作
http://footheory.com/blogs/donnfelker/archive/2008/05/11/using-itextsharp-to-watermark-write-text-to-existing-pdf-s.aspx
3.書籤Bookmark製作
http://www.mazsoft.com/blog/post/2008/04/30/Code-sample-for-using-iTextSharp-PDF-library.aspx
4.Footer 製作
http://stackoverflow.com/questions/1032614/itextsharp-creating-a-footer-page-of

電話：02-33665022 或 3366-5023 傳真： 02-23637204
讀者意見信箱：ntuccepaper@ntu.edu.tw
地址：10617 臺北市羅斯福路四段一號
建議最佳螢幕解析度 1024*768