# Research Assignment

## Abstract

### **Downloading data into PDF Format**

I was assigned by our Team leader to do some research on how to download your viewable data that is displayed on the screen from the database into PDF\_ format within Java using servlets. I discovered the use of IText (theoretically) to be quite useful as it allows you to manipulate your Table information by setting the headings of content and adding sizes, fonts and colours to the information you are saving in the PDF format.

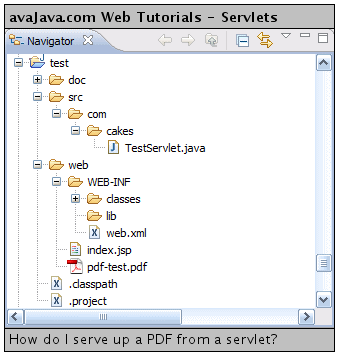
### **Uploading a document onto your Website**

I was also assigned to research uploading a document such as a copy of an ID document onto a Website . I discovered that HTML has an upload function that looks quite easy to use.

Below are code examples that demonstrate both requirement functions.

## **Example using serving a PDF Document in a Servlet**

“It's possible to have a servlet serve up PDF content by specifying the content type of the servlet response to be the 'application/pdf' MIME type via response.setContentType("application/pdf")”



“The TestServlet class is mapped to /test. When the TestServlet is hit by a browser request, it locates the pdf-test.pdf file in the web directory. It sets the response content type to be 'application/pdf', specifies that the response is an attachment, and sets the response content length. Following that, it writes the contents of the PDF file to the response output stream.”

**package** com.cakes;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.IOException;

**import** java.io.OutputStream;

**import** javax.servlet.ServletException;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

**public** **class** TestServlet **extends** javax.servlet.http.HttpServlet **implements** javax.servlet.Servlet {

**private** **static** **final** **long** serialVersionUID = 1L;

**protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

performTask(request, response);

}

**protected** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException,

IOException {

performTask(request, response);

}

**private** **void** performTask(HttpServletRequest request, HttpServletResponse response) **throws** ServletException,

IOException {

String pdfFileName = **"pdf-test.pdf"**;

String contextPath = getServletContext().getRealPath(File.separator);

File pdfFile = **new** File(contextPath + pdfFileName);

response.setContentType(**"application/pdf"**);

response.addHeader(**"Content-Disposition"**, **"attachment; filename="** + pdfFileName);

response.setContentLength((**int**) pdfFile.length());

FileInputStream fileInputStream = **new** FileInputStream(pdfFile);

OutputStream responseOutputStream = response.getOutputStream();

**int** bytes;

**while** ((bytes = fileInputStream.read()) != -1) {

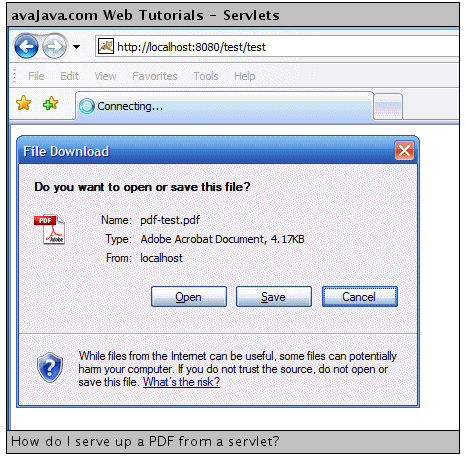
responseOutputStream.write(bytes);

}

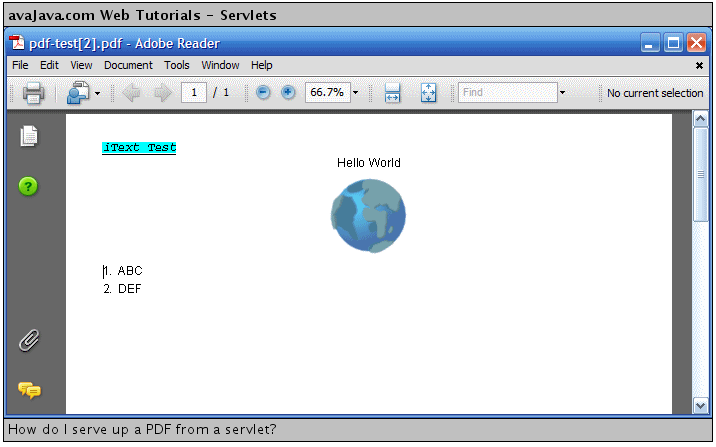
}

}

“If we run the TestServlet.java, the browser asks us if we'd like to open or save the pdf-test.pdf file.”



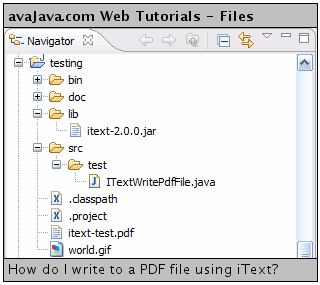
If we open the file, we can see that the servlet[W](http://en.wikipedia.org/wiki/Java_Servlet) did indeed serve up the PDF content via the response.



## **Example code Using iText to write to PDF Format**

“The iText project, located at [www.lowagie.com/iText/](http://www.lowagie.com/iText/), is a Java SW library that lets you generate PDF documents.”

“The iText jar file can be downloaded from the iText website mentioned above and placed in a project, as shown below.”



“The ITextWritePdfFile class creates a file called 'itext-test.pdf' and creates an output stream to write to this file. It creates an itext document object and associates this with the output stream to the file. It adds an author ("Me") and a title ("My iText Test") to the document metadata.”

**package** test;

**import** java.awt.Color;

**import** java.io.File;

**import** java.io.FileNotFoundException;

**import** java.io.FileOutputStream;

**import** java.io.IOException;

**import** java.net.MalformedURLException;

**import** com.lowagie.text.Chunk;

**import** com.lowagie.text.Document;

**import** com.lowagie.text.DocumentException;

**import** com.lowagie.text.Element;

**import** com.lowagie.text.Font;

**import** com.lowagie.text.Image;

**import** com.lowagie.text.List;

**import** com.lowagie.text.Paragraph;

**import** com.lowagie.text.pdf.PdfWriter;

**public** **class** ITextWritePdfFile {

**public** **static** **void** main(String[] args) {

**try** {

File file = **new** File(**"itext-test.pdf"**);

FileOutputStream fileout = **new** FileOutputStream(file);

Document document = **new** Document();

PdfWriter.getInstance(document, fileout);

document.addAuthor(**"Me"**);

document.addTitle(**"My iText Test"**);

document.open();

Chunk chunk = **new** Chunk(**"iText Test"**);

Font font = **new** Font(Font.COURIER);

font.setStyle(Font.UNDERLINE);

font.setStyle(Font.ITALIC);

chunk.setFont(font);

chunk.setBackground(Color.CYAN);

document.add(chunk);

Paragraph paragraph = **new** Paragraph();

paragraph.add(**"Hello World"**);

paragraph.setAlignment(Element.ALIGN\_CENTER);

document.add(paragraph);

Image image;

**try** {

image = Image.getInstance(**"world.gif"**);

image.setAlignment(Image.MIDDLE);

document.add(image);

} **catch** (MalformedURLException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

}

List list = **new** List(true, 15);

list.add(**"ABC"**);

list.add(**"DEF"**);

document.add(list);

document.close();

} **catch** (FileNotFoundException e) {

e.printStackTrace();

} **catch** (DocumentException e) {

e.printStackTrace();

}

}

}

## **Upload HTML Markup**

<form action="http://www.cs.tut.fi/cgi-bin/run/~jkorpela/echo.cgi"

enctype="multipart/form-data" method="post">

<p>

Type some text (if you like):<br>

<input type="text" name="textline" size="30">

</p>

<p>

Please specify a file, or a set of files:<br>

<input type="file" name="datafile" size="40">

</p>

<div>

<input type="submit" value="Send">

</div>

</form>

It simply echoes back the data it gets, but presented so that your browser will display it nicely; for a file field, only 40 first octets (bytes) are shown

