Greenstone tutorial exercise

Back to wiki
Back to index
Sample files: Word_and_PDF.zip
Devised for Greenstone version: 2.70|3.06
Modified for Greenstone version: 2.87|3.08

Enhanced PDF handling

Greenstone converts PDF files to HTML using third-party software: *pdftohtml.pl*. This lets users view these documents even if they don't have the PDF software installed. Unfortunately, sometimes the formatting of the resulting HTML files is not so good. This exercise explores some extra options to the PDF plugin which may produce a nicer version for display.

1. In the Librarian Interface, start a new collection called "PDF collection" and base it on **-- New Collection --**.

   In the **Gather** panel, drag just the PDF documents from *sample_files* → *Word_and_PDF* → *Documents* into the new collection. Also drag in the PDF documents from *sample_files* → *Word_and_PDF* → *difficult_pdf*.

   Go to the **Create** panel and build the collection. Examine the output from the build process. You will notice that one of the documents could not be processed. The following messages are shown: "The file pdf05-notext.pdf was recognised but could not be processed by any plugin.", and "3 documents were processed and included in the collection. 1 was rejected".

2. Preview the collection and view the documents. *pdf05-notext.pdf* does not appear as it could not be processed. *pdf06-weirdchars.pdf* was processed but looks very strange. The other PDF documents appear as one long document, with no sections.

Modes in the Librarian Interface

The Librarian Interface can operate in different modes. The default mode is ***Librarian*** *mode. We can use* ***Expert*** *mode to work out why the pdf file could not be processed.*

3. Use the **Preferences...** item on the **File** menu, **Mode** tab, to switch to **Expert** mode and then build the collection again. The **Create** panel looks different in **Expert** mode because it gives more options: locate the **<Build Collection>** button, near the bottom of the window, and click it. Now a message appears saying that the file could not be processed, and why. Amongst all the output, we get the following message: "Error: PDF contains no extractable text. Could not convert pdf05-notext.pdf to HTML format". pdftohtml.pl cannot convert a PDF file to HTML if the PDF file has no extractable text.

4. We recommend that you switch back to **Librarian** mode for subsequent exercises, to avoid confusion.

Splitting PDFs into sections

5. In the **Document Plugins** section of the **Design** panel, configure **PDFPlugin**. Switch on the **use_sections** option.

   In the **Search Indexes** section, ensure that both the **section** and **document** boxes are checked. This will build the indexes on both the section level and the document level.

   Build and **preview** the collection. View the text versions of some of the PDF documents. Note that these are now split into a series of pages, and a "go to page" box is provided. The format is still a bit ugly though, and pdf05-notext.pdf is still not processed.

Using image format

6. If conversion to HTML doesn't produce the result you'd like, PDF documents can be converted to a series of images, one per page. This requires ImageMagick and Ghostscript to be installed.

7. In the **Document Plugins** section, configure **PDFPlugin**. Set the **convert_to** option to one of the image types, e.g. **pagedimg_jpg**. Switch off the **use_sections** option, as it is not used with image conversion.

8. **Build** the collection and **preview**.All PDF documents (including pdf05-notext.pdf) have been processed and divided into sections, but each section displays "This document has no text.". For the conversion to images for PDF documents, no text is extracted.

9. In order to view the documents properly, you will need to modify the format statement. In the **Format Features** section on the **Format** panel, select the **DocumentText** format statement. Replace

   ```
   [Text]
   ```

   with

   ```
   [srcicon]
   ```

10. Preview the collection. Images from the document are now displayed instead of the extracted text. Both *pdf05-notext.pdf* and *pdf06-weirdchars.pdf* display nicely now.

    In this collection, we only have PDF documents and they have all been converted to images. If we had other document types in the collection, we should use a different format statement, such as:

    ```
    {If}{[parent:FileFormat] eq PDF,[srcicon],[Text]}
    ```

    FileFormat *is an extracted metadata item which shows the format of the source document. We can use this to test whether the documents are PDF or not: for PDF documents, display [srcicon], for other documents, display [Text].*

Using process_exp to control document processing (advanced)

11. Processing all of the PDF documents using an image type may not give the best result for your collection. The images will look nice, but as no text is extracted, searching the full text will not be available for these documents. The best solution would be to process most of the PDF files as HTML, and only use the image format where HTML doesn't work.

12. We achieve this by putting the problem files into a separate folder, and adding another **PDFPlugin** plugin with different options.

13. Go to the **Gather** panel. Make a new folder called "notext": right click in the collection panel and select **New folder** from the menu. Change the **Folder Name** to "notext", and click **<OK>**.

    Move the two pdf files that have problems with html (*pdf05-notext.pdf* and *pdf06-weirdchars*.pdf) into this folder by drag and drop. We will set up the plugins so that PDF files in this *notext* folder are processed differently to the other PDF files.

14. Switch to the **Document Plugins** section of the **Design** panel. Add a second PDF plugin by selecting **PDFPlugin** from the **Select plugin to add:** drop-down list, and clicking **<Add Plugin...>**. This plugin will come after the first PDF plugin, so we configure it to process PDF documents as HTML. Set the **convert_to** option to **html**, and switch on the **use_sections** option. Click **<OK>**.

15. Configure the first PDF plugin, and set the **process_exp** option to **"notext.*\.pdf"**.

16. The two PDF plugins should have options like the following:

    ```
    plugin PDFPlugin -convert_to pagedimg_jpg -process_exp "notext.*\.pdf"
    plugin PDFPlugin -convert_to html -use_sections
    ```

    The *paged_img* version must come earlier in the list than the *html* version. The **process_exp** for the first **PDFPlugin** will process any PDF files in the *notext* directory. The second **PDFPlugin** will process any PDF files that are not processed by the first one.

    Note that all plugins have the **process_exp** option, and this can be used to customize which documents are processed by which plugin.

17. Edit the **DocumentText** format statement. PDF files processed as HTML will not have images to display, so we need to make sure they get text displayed instead. Change `[srcicon]` to `{If}{[NoText] eq "1",[srcicon],[Text]}`.

18. Build and preview the collection. All PDF documents should look relatively nice. Try searching this collection. You will be able to search for the PDFs that were converted to HTML (try e.g. "bibliography"), but not the ones that were converted to images (try searching for "FAO" or "METS").

Opening PDF files with query terms highlighted

19. Next we'll customize the **SearchVList** format statement to highlight the query terms in a PDF file when it is opened from the search result list. This requires Acrobat Reader 7.0 version or higher, and currently only works on a Microsoft Windows platform.

20. The search terms are kept in the macro variable **_queryterms_**, and we append **#search="_queryterms_"** to the end of a PDF file link to pass the query terms to the PDF.

PDFPlugin saves each PDF file in a unique directory. You can use

```
_httpcollection_/index/assoc/[archivedir]/[srclinkFile]
```

to refer to these files.

21. Add **SearchVList** by selecting **Search** from the **Choose Feature** drop down list, and **VList** from the **Affected Component** list. Click **<Add Format>** to add the **SearchVList** format statement into the list of assigned formats. We need to test whether the file is a PDF file before linking to it, using `{If}{[ex.FileFormat] eq 'PDF',,}`. For PDF files, we use the above path format instead of the `[ex.srclink]` and `[ex./srclink]` variables to link to the file.

    The resulting format statement is:

```
<td valign="top">[link][icon][/link]</td>
<td valign="top">{If}{[ex.FileFormat] eq 'PDF', <a href=\"_httpcollection_/index/assoc/[archivedir]/[srclinkFile]#search=&quot;_queryterms_&quot;\">
{Or}{[ex.thumbicon],[ex.srcicon]}</a>,
[ex.srclink]{Or}{[ex.thumbicon],[ex.srcicon]}[ex./srclink]}</td>
<td valign="top">[highlight]
{Or}{[dc.Title],[ex.Title],Untitled}
[/highlight]{If}{[ex.Source],<br><i>([ex.Source])</i>}</td>
```

When the PDF icons are clicked in the search results, Acrobat will open the file with the search window open with the query terms highlighted.

Copyright © 2005-2016 by the New Zealand Digital Library Project at the University of Waikato, New Zealand
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License."