GREENSTONE DIGITAL LIBRARY

INSIDE GREENSTONE COLLECTIONS

**Ian H. Witten, David Bainbridge, Stefan Boddie,
Kathy J. Don, John R. McPherson**

New Zealand Digital Library
Department of Computer Science
University of Waikato, New Zealand

One of the trickier parts of using Greenstone is coming up with a configuration file for your new collection. It seems like a black art! To help learn how to do it, we present, and explain, the configuration files for a few actual Greenstone collections:

- Greenstone demo
- MSWord and PDF demonstration
- Greenstone Archives collection (email)
- Simple image collection
- Bibliography collection (with fielded searching)
- OAI demonstration collection
- MARC record collection

We also give an extensive example of how Greenstone's appearance can be customized.

These descriptions are intended to be used with Greenstone version 2.40 and higher.

We want to ensure that this software works well for you. Please report any problems to *greenstone@cs.waikato.ac.nz*

Greenstone gsdl-2.40 June 2003

1. The Greenstone demo collection

The Greenstone demo collection is supplied with the software, and is used extensively as an example in the documentation. Although we've put it first because it is the the standard demo that comes with the software, it's a fairly complex example to start off with; you might prefer to skip ahead and look at some other collections first (e.g. MSWord and PDF, or Greenstone Archives, or the Simple image collection).

The collection configuration file is shown at the end of this section. All configuration files begin with a line ("creator") that gives the email address of the collection's creator, and another ("public") that determines whether the collection will appear on the home page of the Greenstone installation.

Collection-level metadata. The *collectionmeta* lines in the configuration file are also standard in all Greenstone collections. They give general information about the collection, defining its name, a brief description that appears on its home page, and two versions of the collection's icon. The brief description (in *collectionextra*) can be seen on the demo collection's home page in Figure 1. The *iconcollection* item gives the image proclaiming "greenstone demo" that appears at the upper left of Figure 1: if it is absent, the collection's name appears instead. This image is placed in the *images* subdirectory of the collection's directory (typically, on Windows configurations, in *C:\Program files\gsdl\collect\demo\images*). The *iconcollectionsmall* is a smaller version of the icon that is used on the Greenstone home page.

collect\demo\images\demo.gif

collect\demo\images\demosm.gif

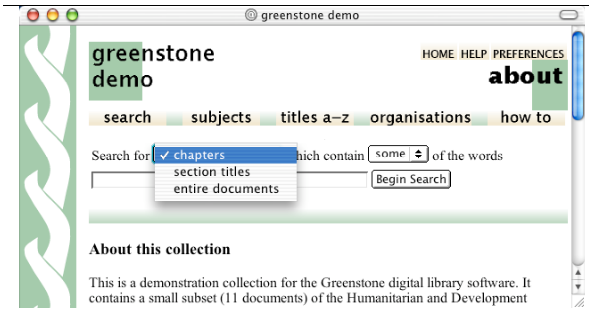
Plugins. The third block of lines in the configuration file gives the plugins used by the collection. The documents in the demo collection are in HTML, so *HTMLPlug* must be included. The *description_tags* option processes tags in the text that define sections and section titles as described below. The *cover_image* flag specifies that each document has a cover picture whose name is the same as the document's but with a *.jpg* extension. *WordPlug* and *PDFPlug* also appear in the configuration file, but are not used for the documents in the demo collection. Extra plugins do no harm. In general the ordering of plugins is not significant, unless there are two different plugins that can process the same type of document.

The other plugins, *GAPlug*, *ArcPlug*, and *RecPlug*, are used by Greenstone for internal purposes and are standard in almost all collections. The *use_metadata_files* flag on *RecPlug* directs Greenstone to look for *metadata.xml* files that specify metadata for the documents in XML format (see below).

Searchable indexes. The block of lines starting with *indexes* specifies what searchable indexes will be available. In this collection there are three: you can see them in Figure 1 because the "Search for" menu has been pulled down. The first index is called "chapters," the second "section titles," and the third "entire documents." The names of these three indexes are given by three *collectionmeta* statements.

The contents of the indexes—that is, the specification of what it is that will be searched—are defined by the *indexes* line at the beginning of this block. This specifies three indexes, two at the section level (beginning with *section:*) and one at the document level (beginning with *document:*). The difference is that a multi-word query will only match a section-level index if all query terms appear in the same section, whereas it will match a document-level index if the terms appear anywhere within the document (which typically comprises several sections). The first and third indexes are *section:text* and *document:text*, and the "*:text*" means that the full text of sections and documents respectively will be searched. The second is *section:Title*, which means that *Title* metadata will be searched—in this case, section titles (rather than document titles). The three indexes appear in the order in which they are specified on the *indexes* line.

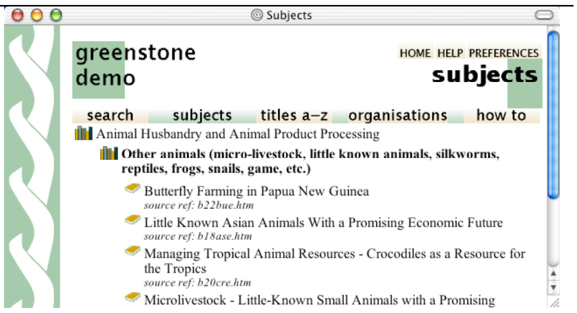
Classifiers. The block of lines labeled *classify* define the browsing indexes, called "classifiers" in Greenstone. There are four of them, corresponding to four buttons on the navigation bar in Figure 1: *subjects*, *titles a–z*, *organisations*, and *how to*. The *search* button comes first, then come the four classifiers, in order. The first classifier provides access by subject. It is a *Hierarchy* classifier whose hierarchy is defined in *sub.txt* (the *hfile* argument); this file is discussed below. This classifier is based on *Subject* metadata, and when several books appear at a leaf of the hierarchy they are sorted by *Title* metadata (as you can see in Figure 2). The second provides access by title: it is an *AZList* classifier based on *Title* metadata. The third provides access by organization: it is a *Hierarchy* classifier based on *Organization* metadata whose hierarchy is defined in *org.txt*; this file is given below. Again, the leaves of the hierarchy are sorted by *Title* metadata. The fourth provides access by *Keyword* metadata: it also is a *Hierarchy* classifer (see below).



About this collection

This is a demonstration collection for the Greenstone digital library software. It contains a small subset (11 documents) of the Humanitarian and Development

Figure 1. The greenstone demo collection



Animal Husbandry and Animal Product Processing

Other animals (micro-livestock, little known animals, silkworms, reptiles, frogs, snails, game, etc.)

Butterfly Farming in Papua New Guinea
source ref: b22bue.htm

Little Known Asian Animals With a Promising Economic Future
source ref: b18ase.htm

Managing Tropical Animal Resources - Crocodiles as a Resource for the Tropics
source ref: b20cre.htm

Microlivestock - Little-Known Small Animals with a Promising

Figure 2. The *subjects* hierarchy browser

Format statements. The next block contains five format statements. The first applies to *Vlists*. These are lists of items displayed vertically down the page, like the the lists displayed by the *titles a-z* browser, those at the leaves of the *subject* and *organisation* hierarchies, and the tables of contents of the target documents themselves. However, for the search results page it is overridden by the second format statement (*SearchVList*). The third governs how the document text is formatted, with *Title* metadata ([*Title*]) in HTML <*h3*> format followed by the text of the document [*Text*]. The fourth ensures that cover images are shown with each document. The fifth calls for the *Expand Text*, *Expand Contents*, *Detach* and *Highlight* buttons to be shown with each document.

Most format statements contain a string specified in an augmented form of HTML. Metadata names in square brackets (e.g. [*Title*], [*Creator*]) give the value of that metadata; [*Text*] gives the document text. A hyperlink to the document can be made using [*link*] … [/*link*]; an appropriate icon is produced by [*icon*]. Format strings can include {*If*}{… , …} and {*Or*}{… , …}; the first two give examples. These two are fairly complex format statements; we will not explain them here. In Greenstone, changes in format strings take effect immediately unless you are using the local library server, in which case the server needs to be restarted. This makes it easy to experiment with different versions of a format statement, and see what happens.

Language translations. The last part of the collection configuration file gives the collection-level metadata in French and Spanish respectively. The languages are indicated by square brackets: [*fr*] and [*es*]. If there is no language specification, English is assumed by default. The configuration file shows accented characters (e.g. French *é*). This file is in UTF-8, and these characters are represented by multi-byte sequences (<C3><A9> in this case). Alternatively they could be represented by their HTML entity names (like &*eacute*;). It makes no difference: they look the same on the screen. However, if the text was *searchable* it would make a difference; Greenstone uses Unicode internally to ensure that searching works as expected for non-English languages.

Description tags

The description tags recognized by *HTMLPlug* are inserted into the HTML source text of the documents to define where sections begin and end, and to specify section titles. They look like this:

```
<!--
<Section>
 <Description>
    <Metadata name="Title"> Realizing human rights for poor people: Strategies for
    achieving the international development targets </Metadata>
  </Description>
-->
```

(text of section goes here)

```
<!--
</Section>
-->
```

The <!-- … --> markers are used to ensure that these tags are marked as comments in HTML and therefore do not affect document formatting. In the *Description* part other kinds of metadata can be specified, but this is not done for the style of collection we are describing here. Exactly the same specification (including the <!-- … --> markers) can be used in Word documents too.

Metadata files

Metadata for all documents in the demo collection is provided in a single *metadata.xml* file in the import directory. It looks like this:

collect\demo\import\metadata.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE DirectoryMetadata SYSTEM
    "http://greenstone.org/dtd/DirectoryMetadata/1.0/DirectoryMetadata.dtd">
<DirectoryMetadata>
  <FileSet>
    <FileName>ec160e</FileName>
    <Description>
      <Metadata name="Subject" mode="accumulate">Settlements and housing: general
        works incl. low- cost housing, planning techniques, surveying, etc.</Metadata>
      <Metadata name="Subject" mode="accumulate">The Courier ACP 1990 - 1996 Africa
        -Caribbean-Pacific - European Union</Metadata>
      <Metadata name="Organization">EC Courier</Metadata>
    </Description>
  </FileSet>
```

(ten more FileSet specifications go here, one for each document in the collection)

```
<DirectoryMetadata>
```

This excerpt shows the beginning and end of the file, and the block of metadata for one of the eleven items in the demo collection—an issue of *The Courier*. It defines *Subject* and *Organization* metadata. Two values for *Subject* are specified, both of which are stored as metadata values for this particular document (because *mode=accumulate* is specified; the alternative, and the default, is *mode=override*).

Title metadata is often specified in the metadata file too. However, in this collection it is given in the text of each document instead, using description tags. (If it appeared in both places, Greenstone would use the version defined in the documents in preference to that in the *metadata.xml* file.)

Hierarchy files

The subject hierarchy file *sub.txt* is shown below. The actual file is much larger, but most of it is not needed because it involves subjects that don't occur in the books in the demo collection. Each line has three items. The first and last items are text strings, and they are the same. The middle item is a number that defines the position in the hierarchy. The first string is matched against the metadata that occurs in the *metadata.xml* file described above; the last one is the string that describes that node of the hierarchy on the web pages that Greenstone generates.

collect\demo\etc\sub.txt

```
"Society, Culture, Community, Woman, Youth, Population" 10 "Society, Culture,
   Community, Woman, Youth, Population"
"Social sciences, sociology (works comprising several subgroups) incl. participatory
   research and training" 10.6 "Social sciences, sociology (works comprising several
   subgroups) incl. participatory research and training"
"Communication, Information and Documentation" 12 "Communication, Information and
   Documentation"
"Communication, telecommunication, mass communication, mass media, film-making" 12.2
   "Communication, telecommunication, mass communication, mass media, film-making"
"Agriculture and Food Processing" 13 "Agriculture and Food Processing"
"Better Farming series of FAO and INADES - 46 booklets" 13.8 "Better Farming series of
   FAO and INADES - 46 booklets"
"Animal Husbandry and Animal Product Processing" 14 "Animal Husbandry and Animal
   Product Processing"
"Cattle" 14.5 "Cattle"
"Other animals (micro-livestock, little known animals, silkworms, reptiles, frogs,
   snails, game, etc.)" 14.6 "Other animals (micro-livestock, little known animals,
   silkworms, reptiles, frogs, snails, game, etc.)"
"Settlements, Housing, Building - Infrastructure Construction (Roads etc)" 15
   "Settlements, Housing, Building - Infrastructure Construction (Roads etc)"
"Settlements and housing: general works incl. low- cost housing, planning techniques,
   surveying, etc." 15.3 "Settlements and housing: general works incl. low- cost
   housing, planning techniques, surveying, etc."
"Development Periodicals and Magazines" 16 "Development Periodicals and Magazines"
"The Courier ACP 1990 - 1996 Africa-Caribbean-Pacific - European Union" 16.2 "The
   Courier ACP 1990 - 1996 Africa-Caribbean-Pacific - European Union"
```

The organization hierarchy file *org.txt* is shown below. Again, the actual file is much larger, but only this excerpt is needed for the demo collection. Again, the first and last text strings on each line are the same because the metadata values in *metadata.xml* are exactly what should be shown on the Greenstone web pages. The number between defines the position in the hierarchy: in this case the hierarchy is flat and the position is simply an integer that determines the order of the list.

collect\demo\etc\org.txt

```
"BOSTID"       4       "BOSTID"
"EC Courier"   7       "EC Courier"
"FAO Better Farming series" 9 "FAO Better Farming series"
"World Bank"   16      "World Bank"
```

The *How to* classifier is also a hierarchy classifier, in this case based on *Keyword* metadata. This is to allow for the possibility that two different documents have the same *Keyword*.

collect\demo\etc\keyword.txt

```
"introduce small animals and micro-livestock in your farm" 1 "introduce small animals
   and micro-livestock in your farm"
"introduce little-known Asian farm animals with a promising future" 2 "introduce
   little-known Asian farm animals with a promising future"
"utilize the Water Buffalo more effectively" 3 "utilize the Water Buffalo more
   effectively"
"start a butterfly farm" 4 "start a butterfly farm"
"farm snails" 5 "farm snails"
"achieve gender equality" 6 "achieve gender equality"
```

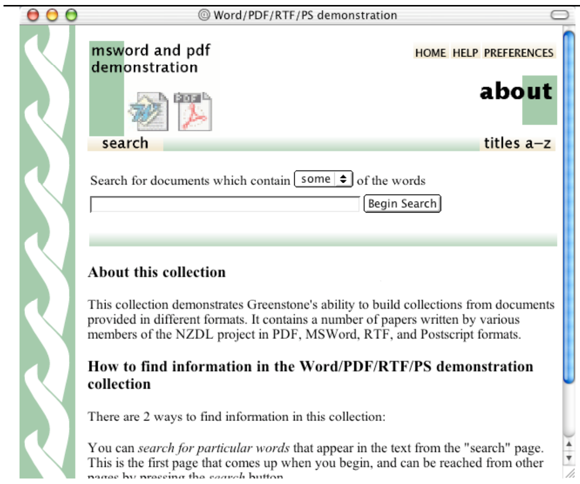
Configuration file

collect\demo\etc\collect.cfg

```
creator         greenstone@cs.waikato.ac.nz
public          true

collectionmeta collectionname        "greenstone demo"
collectionmeta collectionextra       "This is a demonstration collection for the
    Greenstone digital library software. It contains a small subset (11 documents) of
    the Humanity Development Library"
collectionmeta iconcollectionsmall    "_httpprefix_/collect/demo/images/demosm.gif"
collectionmeta iconcollection        "_httpprefix_/collect/demo/images/demo.gif"

plugin         HTMLPlug -description_tags -cover_image
plugin         WordPlug -description_tags
plugin         PDFPlug
plugin         GAPlug
plugin         ArcPlug
plugin         RecPlug -use_metadata_files

indexes        section:text section:Title document:text
collectionmeta .section:text  "chapters"
collectionmeta .section:Title "section titles"
collectionmeta .document:text "entire documents"

classify       Hierarchy -hfile sub.txt -metadata Subject -sort Title
classify       AZList -metadata Title
classify       Hierarchy -hfile org.txt -metadata Organization -sort Title
classify       Hierarchy -hfile keyword.txt -metadata Keyword -sort Title -buttonname
    Howto

format VList "<td valign=top>[link][icon][/link]</td>
    <td valign=top>[highlight]{Or}{[Title],Untitled}[/highlight]
    <i><small>{If}{[Date],<br>_textdate_[Date]}{If}{[NumPages],
    <br>_textnumpages_[NumPages]}{If}{[Source],<br>_textsource_[Source]}</small></i>
    </td>"
format SearchVList "<td valign=top>[link][icon][/link]</td>
    <td>{If}{[parent(All': '):Title],
    [parent(All':'):Title]:}
    [link][Title][/link]</td>"
format         DocumentText   "<h3>[Title]</h3>\\n<p>[Text]"
format         DocumentImages true
format         DocumentButtons "Expand Text|Expand Contents|Detach|Highlight"

collectionmeta collectionextra [l=fr] "C'est une collection pour démonstration du
    logiciel Greenstone. Elle contient une petite partie du projet de bibliothèques
    humanitaires et de développement (11 documents)."
collectionmeta .section:text [l=fr]  "chapitres"
collectionmeta .section:Title [l=fr] "titres des sections"
collectionmeta .document:text [l=fr] "documents entiers"

collectionmeta collectionextra [l=es] "Esto es una colección de demostración para el
    software de biblioteca digital Greenstone. Contiene un pequeño subconjunto (11
    documentos) de la biblioteca del desarrollo para la humanidad."
collectionmeta .section:text [l=es]  "capítulos"
collectionmeta .section:Title [l=es] "títulos de las secciones"
collectionmeta .document:text [l=es] "documentos enteros"
```

2. The MSWord and PDF demonstration

The MSWord and PDF demonstration is a small collection that includes a few Word, RTF, PDF, and PostScript documents (Figure 3). Its configuration file contains these four plugins (along with the standard three, *GAPlug*, *ArcPlug* and *RecPlug*). These four plugins all extract *Title* and *Source* (i.e. filename) metadata. Greenstone contains third-party software that is used to convert Word, RTF, PDF and PostScript files into HTML. The Greenstone team does not maintain these modules, although we do include the latest versions with each Greenstone release. Bugs arise with unusual Word documents (e.g. from older Macintosh systems), and sometimes the text is badly extracted. Some PDF files have no machine-readable text at all, comprising instead a sequence of page *images* from which text can only be extracted by optical character recognition (OCR), which Greenstone does not attempt. If you encounter these problems, here is nothing much you (or we) can do other than omit the rogue documents from the collection, or try to obtain different versions of them.



Search for documents which contain some of the words

About this collection

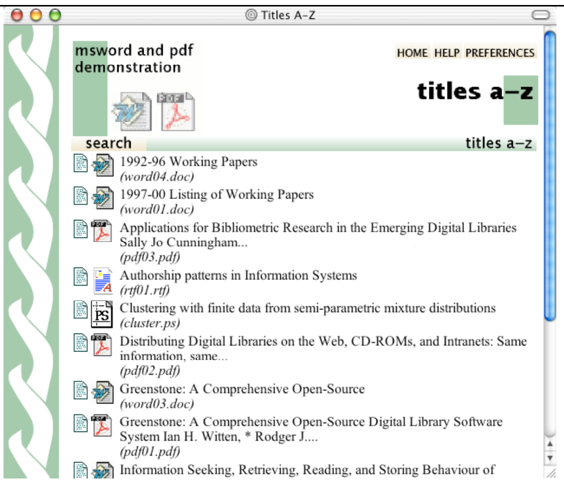
This collection demonstrates Greenstone's ability to build collections from documents provided in different formats. It contains a number of papers written by various members of the NZDL project in PDF, MSWord, RTF, and Postscript formats.

How to find information in the Word/PDF/RTF/PS demonstration collection

There are 2 ways to find information in this collection:

You can *search for particular words* that appear in the text from the "search" page. This is the first page that comes up when you begin, and can be reached from other pages by pressing the *search* button.

Figure 3. The MSWord and PDF collection



1992-96 Working Papers
(word04.doc)
1997-00 Listing of Working Papers
(word01.doc)
Applications for Bibliometric Research in the Emerging Digital Libraries Sally Jo Cunningham...
(pdf03.pdf)
Authorship patterns in Information Systems
(rtf01.rtf)
Clustering with finite data from semi-parametric mixture distributions
(cluster.ps)
Distributing Digital Libraries on the Web, CD-ROMs, and Intranets: Same information, same...
(pdf02.pdf)
Greenstone: A Comprehensive Open-Source
(word03.doc)
Greenstone: A Comprehensive Open-Source Digital Library Software System Ian H. Witten, * Rodger J....
(pdf01.pdf)
Information Seeking, Retrieving, Reading, and Storing Behaviour of

Figure 4. Browsing the collection

The configuration file includes a single index, based on document text, and one classifier, an *AZList* based on *Title* metadata, shown in Figure 4 (the alphabetic selector is suppressed automatically because the collection contains only a few documents). However, no format statement is specified. In the absence of explicit information, Greenstone supplies sensible defaults. In this case, the default format for the classifier gives:

- an icon for the HTML version of the document (the text that is actually indexed, essentially the same as the Greenstone Archive format)
- an icon for the original version of the document (clicking it opens the document in its original form)
- *Title* metadata, extracted from the document
- *Source* (i.e. filename) metadata, extracted from the document.

Here is a format statement that achieves exactly the same effect explicitly. It applies to all *Vlists*, and so controls both search results list and the alphabetic title browser.

```
format VList "<td>[link][icon][/link]</td>
              <td>[srclink][srcicon][/srclink]</td>
              <td>[Title]<br><i>([Source])</i></td>"
```

Configuration file

collect\wordpdf\etc\collect.cfg

```
creator        greenstone@cs.waikato.ac.nz
public         true

collectionmeta collectionname "Word/PDF/RTF/PS demonstration"
collectionmeta iconcollection "_httpprefix_/collect/wordpdf/images/wordpdf.gif"
collectionmeta collectionextra "This collection demonstrates Greenstone's ability to
    build collections from documents provided in different formats. It contains a
    number of papers written by various members of the NZDL project in PDF, MSWord,
    RTF, and Postscript formats."

plugin         WordPlug
plugin         RTFPlug
plugin         PDFPlug
plugin         PSPlug
plugin         GAPlug
plugin         ArcPlug
plugin         RecPlug

indexes        document:text
collectionmeta .document:text "documents"

classify       AZList -metadata Title

format DocumentHeading ""
format DocumentButtons ""

collectionmeta collectionname [l=fr] "Word/PDF/RTF/PS démonstration"
collectionmeta .document:text [l=fr] "documents"
collectionmeta collectionextra [l=fr] "Cette collection d&eacute;montre les
    capacit&eacute;s de Greenstone pour rassembler des collections &agrave; partir de
    documents existants en diff&eacute;rents formats. Elle contient plusieurs articles
    &eacute;crits par diff&eacute;rents membres du projet NZDL en format PDF, MS WORD,
    RTF, et Postscript."

collectionmeta .document:text [l=es] "documentos"
collectionmeta collectionextra [l=es] "Esta colecci&oacute;n demuestra la capacidad
    del programa Greenstone para construir colecciones con documentos en diferentes
    formatos. Contiene art&iacute;culos escritos por varios de los miembros del
    proyecto NZDL en formato PDF, MSWord, RTF y Postscript."
```

3. The Greenstone Archives collection

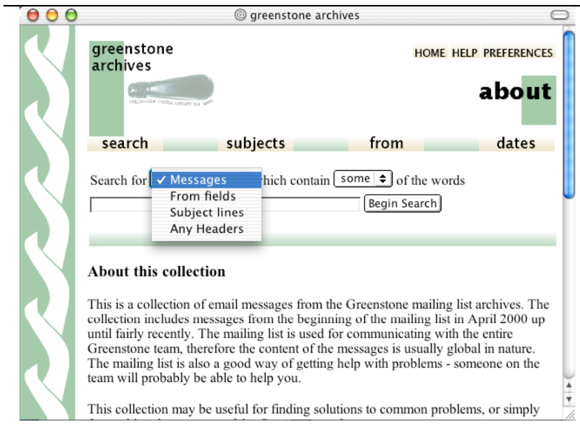
The Greenstone Archives collection contains email messages from the Greenstone mailing list, dating from when the list began in April 2000.

It uses the Email plugin, which parses files in email formats. There is one file for each year, and each file contains many email messages. The Email plugin splits these into individual documents, and produces *Title*, *Subject*, *Headers*, *From*, *FromName*, *FromAddr*, *Date*, and *DateText* metadata.

The collection configuration file begins with the specification *groupsize 200*. This groups documents together into groups of 200. Email collections typically have many small documents, and grouping them together prevents Greenstone's internal file structures from becoming bloated and occupying more disk space than necessary. Notice that first the Email plugin splits the input files up into individual Emails, then *groupsize* groups them together again. This allows the collection designer to control what is going on.

The *indexes* line specifies four searchable indexes, which can be seen on the menu in Figure 5. The first (called *Messages*) is created from the document text, while the others are formed from *From*, *Subject*, and *Headers* metadata.

There are three classifiers, based on *Subject*, *FromName*, and *Date* metadata. The *AZCompactList* classifier used for the first two is like *AZList* but generates a bookshelf for duplicate items, as illustrated in Figure 6. This is represented by a tree structure whose nodes are either leaf nodes, representing documents, or internal nodes. A metadata item called *numleafdocs* gives the total number of documents below an internal node. The format statements for the first classifier, called *CL1Vlist*, checks whether this item exists. If so the node must be an internal one, in which case it is labeled by its *Title*. Otherwise the node's label starts with the *Subject*, then gives *From* metadata (both name and email address, suitably hyperlinked), followed by the *DateText*.



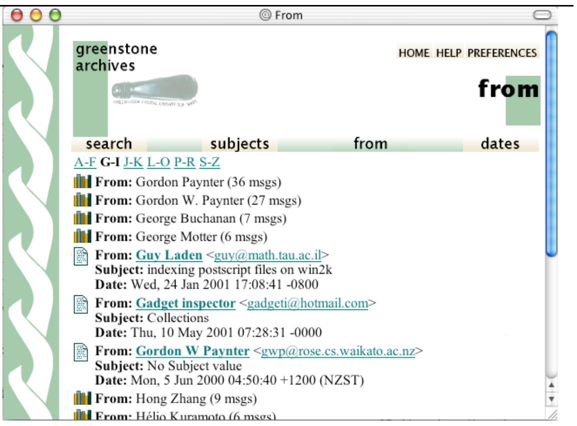
Search for Messages which contain some of the words

About this collection

This is a collection of email messages from the Greenstone mailing list archives. The collection includes messages from the beginning of the mailing list in April 2000 up until fairly recently. The mailing list is used for communicating with the entire Greenstone team, therefore the content of the messages is usually global in nature. The mailing list is also a good way of getting help with problems - someone on the team will probably be able to help you.

This collection may be useful for finding solutions to common problems, or simply

Figure 5. The Greenstone Archives collection



A-F **G-I** J-K L-O P-R S-Z

From: Gordon Paynter (36 msgs)
From: Gordon W. Paynter (27 msgs)
From: George Buchanan (7 msgs)
From: George Motter (6 msgs)
From: Guy Laden <guy@math.tau.ac.il>
Subject: indexing postscript files on win2k
Date: Wed, 24 Jan 2001 17:08:41 -0800
From: Gadget inspector <gadgeti@hotmail.com>
Subject: Collections
Date: Thu, 10 May 2001 07:28:31 -0000
From: Gordon W Paynter <gwp@rose.cs.waikato.ac.nz>
Subject: No Subject value
Date: Mon, 5 Jun 2000 04:50:40 +1200 (NZST)
From: Hong Zhang (9 msgs)
From: Hélio Kuramoto (6 msgs)

Figure 6. Browsing by subject

The second classifier (*CL2Vlist*) is similar, but shows slightly different information—the result can be seen in Figure 6. For internal nodes, the actual number of leaf documents (*numleafdocs*) is given in parentheses after the *Title*; for document nodes the *From*, *Subject*, and *Date* metadata is shown.

The third classifier is a *DateList*, which allows selection by month and year.

Finally, the document text is formatted to show the header fields followed by the message text (written as [*Text*] in the format statement). However, there is a subtle twist, and to see what it is you should look at a document in the collection. At the end of the document is a “show all headers” hyperlink, which, when clicked, shows a long list of email headers and changes the hyperlink at the end of the document to “hide headers.” The faint of heart should skip the following explanation! The *If* in the format statement tests *cgiargheaders*, which in fact determines whether the URL contains a CGI argument called “headers”. If so, the *Headers* metadata is displayed, otherwise it is not. After the the message text has been shown (by [*Text*]), the *cgiargheaders* variable is tested again to determine whether to put the “hide headers” or the “show all headers” hyperlink.

Configuration file

collect\gsarch\etc\collect.cfg

```
creator         greenstone@cs.waikato.ac.nz
public          true

collectionmeta collectionname "greenstone archives"
collectionmeta iconcollection _httpprefix_/collect/gsarch/images/gsarch.gif
collectionmeta collectionextra 'This is a collection of email messages from the
    Greenstone mailing list archives. The collection includes messages from the
    beginning of the mailing list in April 2000 up until fairly recently. The mailing
    list is used for communicating with the entire Greenstone team, therefore the
    content of the messages is usually global in nature. The mailing list is also a
    good way of getting help with problems - someone on the team will probably be able
    to help you. <p> This collection may be useful for finding solutions to common
    problems, or simply for tracking the progress of the Greenstone software. ...”

groupsize       200

plugin          EMAILPlug -process_exp "greenstone.*"
plugin          GAPlug
plugin          ArcPlug
plugin          RecPlug

indexes         document:text document:From document:Subject document:Headers
collectionmeta .document:text         "Messages"
collectionmeta .document:From         "From fields"
collectionmeta .document:Subject      "Subject lines"
collectionmeta .document:Headers      "Any Headers"

classify        AZCompactList metadata=Subject removeprefix=(Re|re|Fwd|fwd):\\s*
classify        AZCompactList metadata=FromName buttonname=From
classify        DateList bymonth=1

format CL1VList '<td valign=top>[link][icon][/link]</td>
    <td> {If}{[numleafdocs], <b>[Title]</b>,
    <b>[Subject]</b>
    <br>From: <a href="_httpquery_&q=[cgisafe:FromName]&h=dfr">[FromName]</a>
    &lt;<a href="mailto:[FromAddr]">[FromAddr]</a>&gt;
    <br>Date: [DateText]}</td>'

format CL2VList '<td valign=top>[link][icon][/link]</td>
    <td> {If}{[numleafdocs], <b>From:</b> [Title] ([numleafdocs] msgs),
    <b>From: <a href="_httpquery_&q=[cgisafe:FromName]&h=dfr">[FromName]</a></b>
    &lt;<a href="mailto:[FromAddr]">[FromAddr]</a>&gt;
    <br><b>Subject:</b> [Subject]<br><b>Date:</b> [DateText]</td>}'

format CL3DateList '<td valign=top>[link][icon][/link]</td>
    <td><b>[Subject]</b>
    <br>From: <a href="_httpquery_&q=[cgisafe:FromName]&h=dfr">[FromName]</a>
     &lt;<a href="mailto:[FromAddr]">[FromAddr]</a>&gt;
    <br>Date: [DateText]</td>'

format SearchVList '<td valign=top>[link][icon][/link]</td>
    <td><b>[Subject]</b>
    <br>From: <a href="_httpquery_&q=[cgisafe:FromName]&h=dfr">[FromName]</a>
    &lt;<a href="mailto:[FromAddr]">[FromAddr]</a>&gt;
    <br>Date: [DateText]</td>'

format DocumentHeading ""
format DocumentButtons ""

format DocumentText '<center> <h2>[Subject]</h2> <p> <table width=90%>
    <tr bgcolor=#DDDDEE><td align=right>From</td>
    <td><b><a href="_httpquery_&q=[cgisafe:FromName]&h=dfr">[FromName]</a>
    &lt;<a href="mailto:[FromAddr]">[FromAddr]</a>&gt;</b></td></tr>
    <tr bgcolor=#DDDDEE><td align=right>Date</td><td><b>[DateText]</b></td></tr>
    <tr bgcolor=#DDDDEE><td align=right>Subject</td><td><b>[Subject]</b></td></tr>
    {If}{_cgiargheaders_,<tr bgcolor=#DDDDEE valign=top><td align=right>Headers</td>
    <td><b>[Headers]</b></td></tr>,}
    <tr><td colspan=2>[Text]</td></tr>
    <tr bgcolor=#DDDDEE><td colspan=2 align=right> {If}{_cgiargheaders_,
    <a href="_httpcurrentdocument_">hide headers</a>,
    <a href="_httpcurrentdocument_&headers=show">show all headers</a>}
    </td></tr> </table> </center>'
```

4. A simple image collection

Figures 7 and 8 are from a basic image collection which contains no text and no explicit metadata. Several JPEG files are placed in the *import* directory prior to importing and building the collection, that is all.

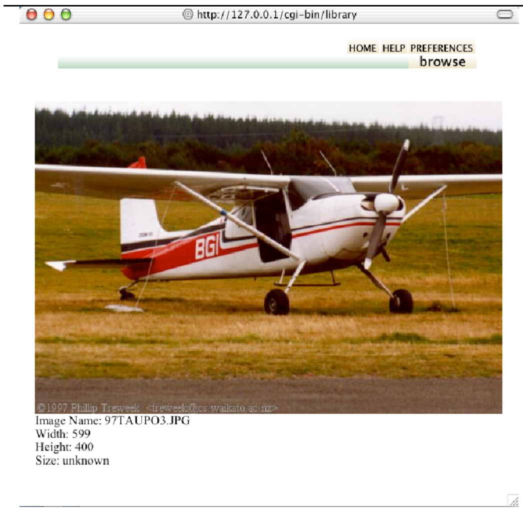

http://127.0.0.1/cgi-bin/library
HOME HELP PREFERENCES
browse
©1997 Phillip Treweek <treweek@cs.waikato.ac.nz>

Image Name: 97TAUPO3.JPG
Width: 599
Height: 400
Size: unknown

Figure 7. Document in a simple image collection

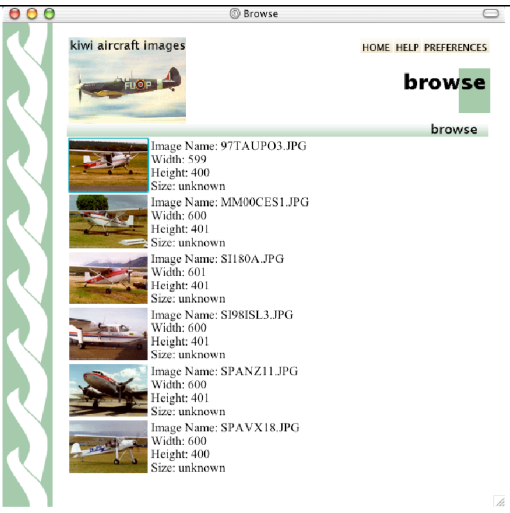


Image Name: 97TAUPO3.JPG
Width: 599
Height: 400
Size: unknown
Image Name: MM00CES1.JPG
Width: 600
Height: 401
Size: unknown
Image Name: SI180A.JPG
Width: 601
Height: 401
Size: unknown
Image Name: SI98ISL3.JPG
Width: 600
Height: 401
Size: unknown
Image Name: SPANZ11.JPG
Width: 600
Height: 401
Size: unknown
Image Name: SPAVX18.JPG
Width: 600
Height: 400
Size: unknown

Figure 8. Browsing the collection

The configuration file specifies no indexes, so the *search* button is suppressed.

There is only one plugin, *ImagePlug*, aside from the three that are always present (*GAPlug*, *ArcPlug*, *RecPlug*). *ImagePlug* relies on the existence of two programs from the ImageMagick suite (http://www.imagemagick.org): *convert* and *identify*. Greenstone will not be able to build the collection correctly unless ImageMagick is installed on your computer.

ImagePlug automatically creates a thumbnail and generates this metadata for each image in the collection:

Image	Name of file containing the image
ImageWidth	Width of image (in pixels)
ImageHeight	Height of image (in pixels)
Thumb	Name of gif file containing thumbnail of image
ThumbWidth	Width of thumbnail image (in pixels)
ThumbHeight	Height of thumbnail image (in pixels)
thumbicon	Full pathname specification of thumbnail image
assocfilepath	Pathname of image directory in the collection's *assoc* directory

The image is stored as an "associated file" in the *assoc* subdirectory of the collection's *index* directory. (*Index* is where all files necessary to serve the collection are placed, to make it self-contained.) The pathname *_httpcollimg_*, which is the same as *_httpcollection_/index/assoc*, refers to this directory. For any document, its thumbnail and image are both in a subdirectory whose filename is given by *assocfilepath*. The metadata element *thumbicon* is set to the full pathname specification of the thumbnail image, and can be used in the same way as *srcicon* (see the MSWord and PDF demonstration collection).

The second format statement in the configuration file, *DocumentText*, dictates how the document will appear, and Figure 7 shows the result. There is no document text (if there were, it would be producible by [*Text*]). What is shown is the image itself, along with some metadata extracted from it.

The configuration file specifies one classifier, an *AZList* based on *Image* metadata, shown in Figure 8 (Greenstone has suppressed the alphabetic selector because this collection has only six images). The format statement shows the thumbnail image along with some metadata. (Any other classifiers would have the same format, since this statement does not name the classifier.)

You may wonder why the thumbnail image is generated and stored explicitly, when the same effect would be obtained by using the original image and scaling it:

```
<td>[link]<img src='_httpcollimg_/[assocfilepath]
    /[Image]' width=[ThumbWidth] height=[ThumbHeight]>
    [/link]</td><td valign=middle><i>[Title]</i></td>
```

The reason is to save communication bandwidth by not sending large images when small ones would do.

For a more comprehensive image collection, see the kiwi aircraft images in the New Zealand Digital Library (the images in Figure 7 and 8 were taken from this collection). The structure of this collection is quite different, however: it is a collection of web pages that include many images along with the text. The HTML plugin *HTMLPlug* also processes image files, but it does so in a different way from *ImagePlug* (for example, it does not produce the metadata described above). In fact, this is one of the few situations where the ordering of plugins in the collection configuration files makes a difference. If both plugins were included, images would be processed by whichever came first in the configuration file.

Configuration file

```
creator         greenstone@cs.waikato.ac.nz
public          true

collectionmeta collectionname "img_demo"
collectionmeta iconcollection _httpprefix_/collect/img_demo/images/logo.gif

plugin          ImagePlug
plugin          GAPlug
plugin          ArcPlug
plugin          RecPlug

classify        AZList -metadata Image -buttonname Browse

format VList '<td valign="top">[srclink][thumbicon]/srclink]</td>
    <td valign="top">Image Name: [Image]<br>Width: [ImageWidth]
    <br>Height: [ImageHeight]<br>Size: [ImageSize]</td>'

format DocumentText '<center><table width="_pagewidth_">
    <tr><td><img src="_httpcollimg_/[assocfilepath]/[Image]">
    <br>Image Name: [Image]<br>Width: [ImageWidth]<br>Height: [ImageHeight]
    <br>Size: [ImageSize]</td></tr></table></center>'

format DocumentHeading ''
format DocumentButtons ''
```

5. Bibliography collection

The Colt bibliography is a bibliographic collection that incorporates a form-based search interface, which allows fielded searching. To do this the collection must use an enhanced search engine (called *mgpp*), rather than Greenstone's default search engine (called *mg*). There is an online help document for *mgpp*. Figure 9a shows the form search interface, Figure 9b shows the "advanced" version of form search, and Figure 9c shows the plain single-field search page that is also available in this collection. These two variants can be selected from the collection's *Preferences* page.

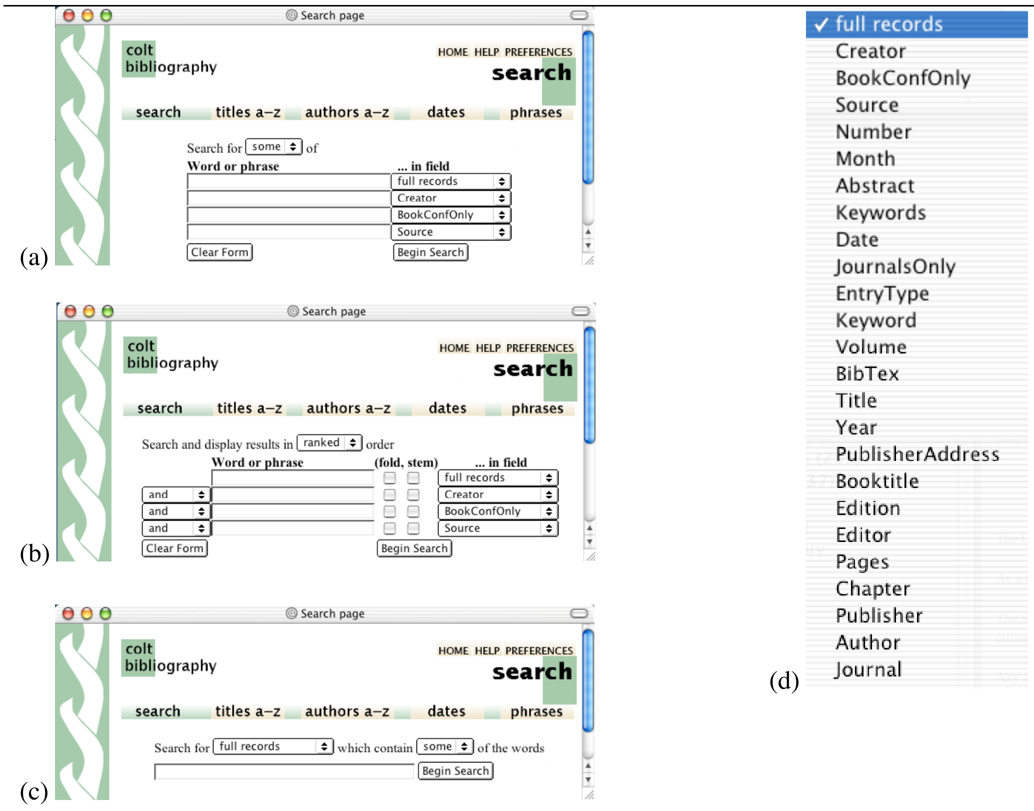


Figure 9. Searching a bibliographic collection

The collection configuration file begins with the specification *groupsize 200.* This groups documents together into groups of 200. Bibliography collections typically have many small documents, and grouping them together prevents Greenstone's internal file structures from becoming bloated and occupying more disk space than necessary.

Apart for the standard ones, the plugins specified for this collection are *ZIPPlug*, which unzips compressed documents and archives, and *BibTexPlug*, which processes references in the BibTeX format (well known to computer scientists).

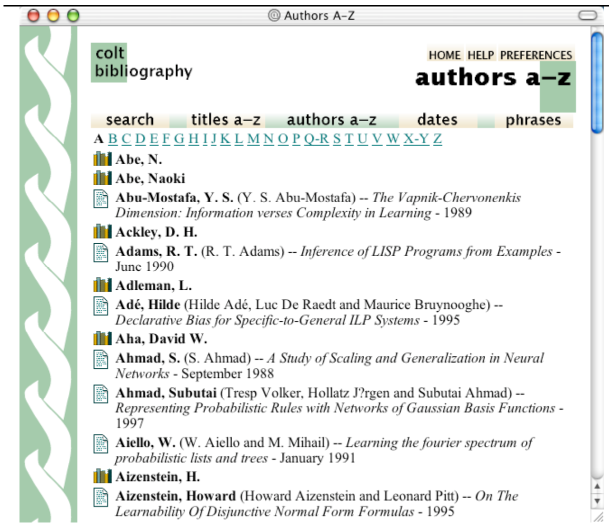
Fielded searching, with a form-based interface, is selected by *searchtype form* in the configuration file. In fact, this collection uses *searchtype form plain*, which includes a plain textual full-text search index as well (since *form* comes first, it is the default interface; you reach the *plain* search through the *Preferences* page).

The inclusion of *searchtype* means that the search engine *mgpp* is used, and for this indexes are specified in a slightly different way. Whereas with

Greenstone's default search engine *mg* the various indexes can be at different "levels" (*document*, *section*, *paragraph*), with *mgpp* they are all at the same level—*document*, by default (as in this case). The level can be changed using a *levels* statement. Also, whereas in other collections indexes can be specified on text or on any metadata; here there are additional possibilities: you can specify indexes on *every* metadata field by using the single word *metadata*, and an index for all the metadata fields together by using the word *allfields*.

In this case the *indexes* line specifies searchable indexes on the full text and on every metadata field. Thus when the "field" menus in Figure 9 are pulled down as shown in Figure 9d, they show *full record* followed by an entry for each metadata element. Collection-level metadata *collectionmeta* can be specified for any index to determine what it is called in the menu (except for *metadata*, which produces many menu items). In this case, the configuration file specifies that the text index should be named "full record" because it contains the original bibliographic record.

This collection contains *Title*, *Author*, and *Date* browsers, and a special kind of phrase index called "Phind." The *AZCompactList* classifier used for the *Author* browser is like *AZList* but generates a bookshelf for duplicate items as shown in Figure 10. The BibTeX plugin records each author as *Author* metadata; it also puts a list containing all authors into the *Creator* metadata element. Consequently the *AZCompactList* classifier is based on *Author*. However, Greenstone has a standard button reading *authors a-z* whose name is (confusingly) "Creator", so this button name is specified for the classifier.



Abe, N.

Abe, Naoki

Abu-Mostafa, Y. S. (Y. S. Abu-Mostafa) -- *The Vapnik-Chervonenkis Dimension: Information verses Complexity in Learning* - 1989

Ackley, D. H.

Adams, R. T. (R. T. Adams) -- *Inference of LISP Programs from Examples* - June 1990

Adleman, L.

Adé, Hilde (Hilde Adé, Luc De Raedt and Maurice Bruynooghe) -- *Declarative Bias for Specific-to-General ILP Systems* - 1995

Aha, David W.

Ahmad, S. (S. Ahmad) -- *A Study of Scaling and Generalization in Neural Networks* - September 1988

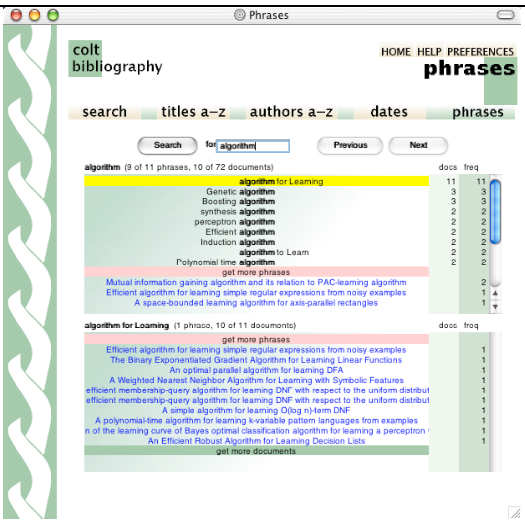
Ahmad, Subutai (Tresp Volker, Hollatz J?rgen and Subutai Ahmad) -- *Representing Probabilistic Rules with Networks of Gaussian Basis Functions* - 1997

Aiello, W. (W. Aiello and M. Mihail) -- *Learning the fourier spectrum of probabilistic lists and trees* - January 1991

Aizenstein, H.

Aizenstein, Howard (Howard Aizenstein and Leonard Pitt) -- *On The Learnability Of Disjunctive Normal Form Formulas* - 1995

Figure 10. The compacted version of the *AZList*



algorithm (9 of 11 phrases, 10 of 72 documents)	docs	freq
algorithm for Learning	11	11
Genetic **algorithm**	3	3
Boosting **algorithm**	3	3
synthesis **algorithm**	2	2
perceptron **algorithm**	2	2
Efficient **algorithm**	2	2
Induction **algorithm**	2	2
algorithm to Learn	2	2
Polynomial time **algorithm**	2	2
get more phrases		
Mutual information gaining algorithm and its relation to PAC-learning algorithm		2
Efficient algorithm for learning simple regular expressions from noisy examples		1
A space-bounded learning algorithm for axis-parallel rectangles		1

algorithm for Learning (1 phrase, 10 of 11 documents)	docs	freq
get more phrases		
Efficient algorithm for learning simple regular expressions from noisy examples		1
The Binary Exponentiated Gradient Algorithm for Learning Linear Functions		1
An optimal parallel algorithm for learning DFA		1
A Weighted Nearest Neighbor Algorithm for Learning with Symbolic Features		1
efficient membership-query algorithm for learning DNF with respect to the uniform distribut		1
efficient membership-query algorithm for learning DNF with respect to the uniform distribut		1
A simple algorithm for learning O(log n)-term DNF		1
A polynomial-time algorithm for learning k-variable pattern languages from examples		1
n of the learning curve of Bayes optimal classification algorithm for learning a perceptron		1
An Efficient Robust Algorithm for Learning Decision Lists		1
get more documents		

Figure 11. The phrase browser

The "Phind" classifier creates the phrase index seen in Figure 11. This is a browsable list of phrases extracted from the material specified in the *text* argument of the *classify Phind* line in the configuration file. Here the specification is

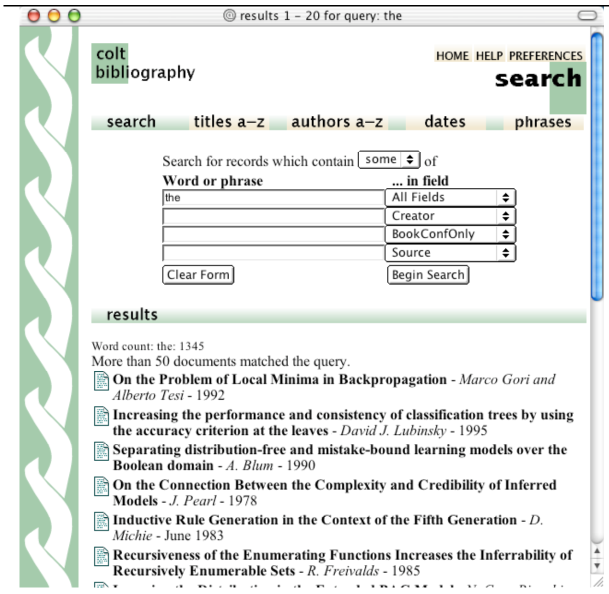
```
document:Title,document:Creator,document:Booktitle,
    document:Publisher
```

—that is, the title, list of authors, title of the collected work (if any) in which this item appears, and publisher. Note that this specification follows the *mg* convention with *level:field*. *Phind* indexes are more usually based on the entire full text of a collection, using the specification *document:text*.

The best way to see what this index does is to play with it. You type a word in the search box, click *Search*, and a list of phrases containing that term appears in the top panel. Click on one of these phrases and a list of phrases containing that *phrase* appears in the bottom panel. You can continue doing this, expanding the phrase more and more. The lists can be lengthened using the *get more phrases* button. At the end of the list of phrases appears a list of documents containing that phrase, in blue text; you can lengthen this list by clicking *get more documents*.

The format statements for the search results list and the title browser are both determined by the *VList* specification. It gives a document icon that links to the document itself (which in this collection is the full reference); the title in bold; *Creator* metadata if there is any, otherwise *Editor* metadata; and *Date* metadata if there is any. Figure 12 shows an example.

The format statement for the author browser (*CL2VList*) is more complex. The *AZCompactList* classifier generates a tree whose nodes are either leaf nodes, representing documents, or internal nodes. A metadata item called *numleafdocs* gives the total number of documents below an internal node. This format statement checks whether *numleafdocs* exists. If so the node must be an internal node, in which case the node is labeled by its *Title*. But beware: this classifier is generated on *Author* metadata, so its title—the title of the classifier—is actually the author's name! This means that the bookshelf nodes in Figure 10 are labeled by author's name. The leaf nodes, however, are labeled the same way as documents (i.e. references) are in the search results list.



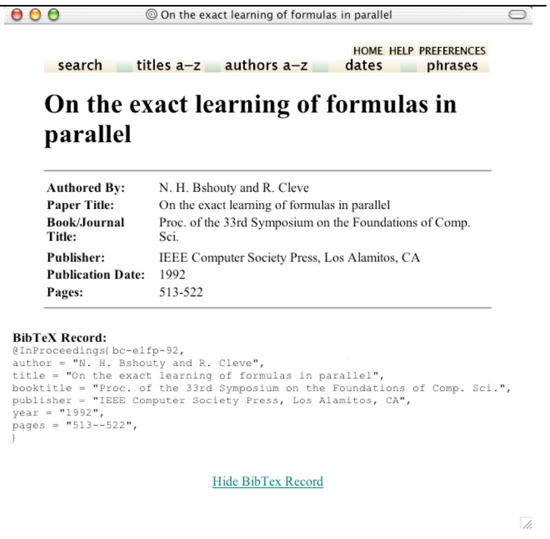
Search for records which contain some of

Word or phrase	**... in field**
the	All Fields
	Creator
	BookConfOnly
	Source

Word count: the: 1345
More than 50 documents matched the query.

On the Problem of Local Minima in Backpropagation - *Marco Gori and Alberto Tesi* - 1992
Increasing the performance and consistency of classification trees by using the accuracy criterion at the leaves - *David J. Lubinsky* - 1995
Separating distribution-free and mistake-bound learning models over the Boolean domain - *A. Blum* - 1990
On the Connection Between the Complexity and Credibility of Inferred Models - *J. Pearl* - 1978
Inductive Rule Generation in the Context of the Fifth Generation - *D. Michie* - June 1983
Recursiveness of the Enumerating Functions Increases the Inferrability of Recursively Enumerable Sets - *R. Freivalds* - 1985

Figure 12. The search results list



On the exact learning of formulas in parallel

Authored By:	N. H. Bshouty and R. Cleve
Paper Title:	On the exact learning of formulas in parallel
Book/Journal Title:	Proc. of the 33rd Symposium on the Foundations of Comp. Sci.
Publisher:	IEEE Computer Society Press, Los Alamitos, CA
Publication Date:	1992
Pages:	513-522

BibTeX Record:

```
@InProceedings{ bc-elfp-92,
author = "N. H. Bshouty and R. Cleve",
title = "On the exact learning of formulas in parallel",
booktitle = "Proc. of the 33rd Symposium on the Foundations of Comp. Sci.",
publisher = "IEEE Computer Society Press, Los Alamitos, CA",
year = "1992",
pages = "513--522",
}
```

Figure 13. A document

An example document is shown in Figure 13. It is generated by two format statements, one (a long one) called *DocumentHeading*, and another called *DocumentText*. The *DocumentHeading*, which is the top two-thirds of Figure 13, contains the document's *Title* followed by a table that gives all the metadata elements that the BibTeX plugin can generate. The role of all of the *If* statements in the configuration file is to determine which elements are defined.

The *DocumentText* shows the BibTeX version of the reference. However, when the document is displayed initially, only a hyperlink reading *Show BibTex Record* appears—this corresponds to the last part (that is, the "else" part) of the *If* statement in *DocumentText*. When this hyperlink is clicked, the *href* goes to the same URL but with *showrecord*=1. This then displays the first part of the *If* statement, which shows the *Text* of the document. With the BibTeX plugin, the text of a document is its unadulterated BibTeX record.

Configuration file

```
creator         jrm21@cs.waikato.ac.nz
public          true

collectionmeta collectionname "COLT Bibliography"
collectionmeta iconcollection "_httpprefix_/collect/coltbib/images/colt.gif"
collectionmeta iconcollectionsmall "_httpprefix_/collect/coltbib/images/coltsm.gif"
collectionmeta collectionextra "This collection is made from the Computational
    Learning Theory (COLT) Bibliography. Here is COLT's
    <a href='http://www.learningtheory.org/'>home page </a> and the bibliography's
    <a href='http://www.i.kyushu-u.ac.jp/~thomas/COLTBIB/coltbib.jhtml'>home page</a>.
    This collection contains _about:numdocs_ BibTeX entries. Also, it uses MGPP
    instead of MG (the underlying backend), which allows fielded searches."

groupsize       200

searchtype      form plain
indexes         text metadata
collectionmeta .text "full records"

plugin          ZIPPlug
plugin          BibTexPlug
plugin          GAPlug
plugin          ArcPlug
plugin          RecPlug

classify        AZList -metadata Title
classify        AZCompactList -metadata Author -buttonname Creator
classify        DateList
classify        Phind
    -text document:Title,document:Creator,document:Booktitle,document:Publisher

format VList "<td valign=top>[link][icon][/link]</td>
    <td valign=top><b>[Title]</b> - <i>{Or}{[Creator],[Editor]}</i>
    {If}{[Year], - [Month] [Year]}</td>"
format CL2VList "<td valign=top>[link][icon][/link]</td>
    <td valign=top>{If}{[numleafdocs],<b>[Title]</b>,
    <b>[Author]</b> ([Creator]) -- <i>[Title]</i>}
    {If}{[Year], - [Month] [Year]}</td>"
format DateList "<td valign=top>[link][icon][/link]</td>
    <td valign=top><b>[Title]</b> - <i>{Or}{[Creator],[Editor]}</i></td>"

format DocumentHeading '<H1>[Title]</H1><hr><table>
{If}{[Creator],<tr><td><b>Authored By:</b></td><td>[Creator]</td></tr>}
{If}{[Title],<tr><td><b>Paper Title:</b></td><td>[Title]</td></tr>}
{If}{[Editor],<tr><td><b>Editor(s):</b></td><td>[Editor]</td></tr>}
{If}{[EditorRole],<tr><td><b>Editor Role:</b></td><td>[EditorRole]</td></tr>}
{If}{[Booktitle],<tr><td><b>Book/Journal Title:</b></td><td>[Booktitle]</td></tr>}
{If}{[Journal],<tr><td><b>In:</b></td><td>[Journal]</td></tr>}
{If}{"[Volume][Number]",<tr><td colspan="2">}
{If}{[Number], <b>Number</b> [Number] }
{If}{[Volume], <b>Vol.</b> [Volume]}
{If}{"[Volume][Number]",</td></tr>}
{If}{[MeetingDate],<tr><td><b>Meeting Date:</b></td><td>[MeetingDate]</td></tr>}
{If}{[MeetingPlace],<tr><td><b>Meeting Place:</b></td><td>[MeetingPlace]</td></tr>}
{If}{[PublicationPlace],<tr><td><b>Publication Place:</b></td><td>[PublicationPlace]
    </td></tr>}
{If}{[Publisher],<tr><td><b>Publisher:</b></td><td>[Publisher]</td></tr>}
{If}{[Year],<tr><td><b>Publication Date:</b></td><td>[Month] [Year]</td></tr>}
{If}{[Pages],<tr><td><b>Pages:</b></td><td>[Pages]</td></tr>}
{If}{[DocType],<tr><td><b>DocType:</b></td><td>[DocType]</td></tr>}
{If}{[Location],<tr><td><b>Location/URL:</b></td><td>[Location]</td></tr>}
{If}{[Notes],<tr><td><b>Annotations:</b></td><td>[Notes]</td></tr>}
{If}{[Abstract],<tr><td><b>Abstract:</b></td><td>[Abstract]</td></tr>}
{If}{[Keywords],<tr><td><b>Keywords:</b></td><td>[Keywords]</td></tr>}
</table><hr>'
format "DocumentText" "{If}{_cgiargshowrecord_,
    <b>BibTeX Record:</b><br/><tt>[Text]</tt><br><center>
    <a href='_gwcgi_?e=_cgiarge_&a=d&d=_cgiargd_'>Hide BibTex Record</a></center>,
    <center><a href='_gwcgi_?e=_cgiarge_&a=d&d=_cgiargd_&showrecord=1'>
    Show BibTex Record</a></center>} "
format "DocumentButtons" ""
format "DocumentContents" "false"
```

6. OAI demonstration collection

The OAI demonstration collection is built from an Open Archives Initiative metadata database and exemplifies Greenstone's *importfrom* feature. Using the Open Archive Protocol (see http://www.openarchives.org), it retrieves a collection of photographs taken at the inaugural Joint Conference on Digital Libraries from http://rocky.dlib.vt.edu/~jcdlpix and displays them in Greenstone. The implementation is flexible enough to cope with the minor syntax differences between OAI 1.1 and OAI 2.0.

The collection configuration file contains an *acquire* line that is interpreted by a special program called *importfrom.pl*. Like other Greenstone programs, this takes as argument the name of the collection, and provides a summary of other arguments when invoked with argument *–help*. It reads the collection configuration file, finds the *acquire* line, and processes it. In this case, it is run with the command

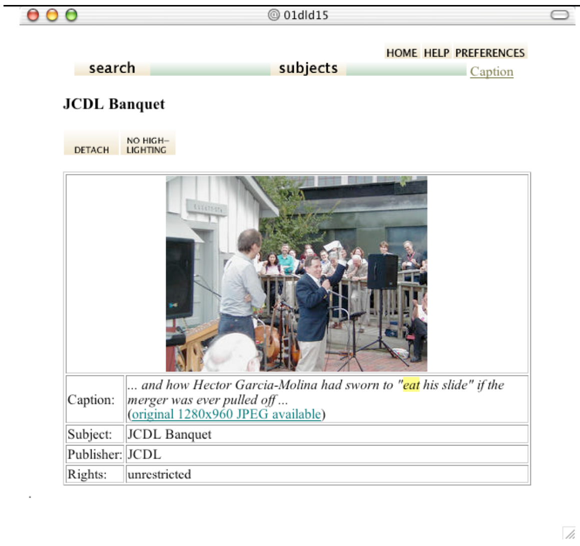
```
importfrom.pl demooai
```

The *acquire* line in the configuration file specifies the OAI protocol and gives the base URL of an OAI repository. The *importfrom* program downloads all the metadata in that repository into the collection's *import* directory. The *getdoc* argument instructs it to also download the collection's source documents, whose URLs are given in each document's Dublin Core *Identifier* field (this is a common convention). The metadata files, which each contain an XML record for one source document, are placed in the *import* file structure along with the documents themselves, and the document filename is the same as the filename in the URL. The *Identifier* field is overridden to give the local filename, and its original value is retained in a new field called *OrigURL*. Here is an example of a downloaded metadata file:

```
<?xml version="1.0" encoding="UTF-8"?>
<OAI-PMH
    xmlns="http://www.openarchives.org/OAI/2.0/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/
             http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
<responseDate>2003-06-18T03:23:29Z</responseDate>
<request verb="GetRecord" identifier="oai:celebration:struther"
    metadataPrefix="oai_dc">http://digital.library.upenn.edu/webbin/OAI-
    celebration</request>
<GetRecord>
<record>
<header>
<identifier>oai:celebration:struther</identifier>
<datestamp>2002-10-18</datestamp>
</header>
<metadata>
<oai_dc:dc
      xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
      xmlns:dc="http://purl.org/dc/elements/1.1/"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/
                         http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
<dc:title>Collected Works of Jan Struther</dc:title>
<dc:creator>Struther, Jan</dc:creator>
<dc:subject>PR6025 .A9 A1</dc:subject>
<OrigURL>http://digital.library.upenn.edu/women/struther/struther.html</OrigURL>
   <identifier>.orig/struther.html</identifier>
<dc:publisher>A Celebration of Women Writers</dc:publisher>
<dc:date>2001-10-13</dc:date><dc:type>Text</dc:type>
</oai_dc:dc>
</metadata>
</record>
</GetRecord>
</OAI-PMH>
```

Once the OAI information has been imported, the collection is processed in the usual way. The configuration file specifies the OAI plugin, which processes OAI metadata, and the image plugin, because in this case the collection's source documents are image files. The OAI plugin has been supplied with an *input_encoding* argument because data in this archive contains extended characters. It also has a *default_language* argument. Greenstone normally determines the language of documents automatically, but these metadata records are too small for this to be done reliably: hence English is specified explicitly in the *language* argument. The OAI plugin parses the metadata and passes it to the appropriate source document file, which is then processed by an appropriate plugin—in this case *ImagePlug*. This plugin specifies the resolution for the screen versions of the images.

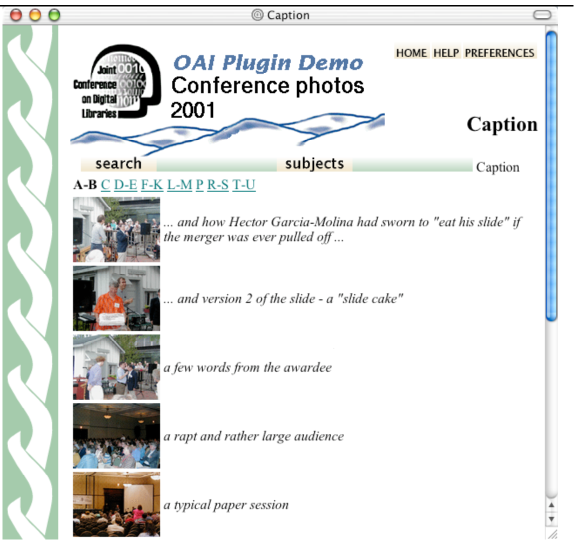
The collection configuration file has a single full-text index containing *Description* metadata. When a document is displayed the *DocumentHeading* format statement puts out its *Subject*. Then the *DocumentText* statement follows this with *screenicon*, which is produced by *ImagePlug* and gives a screen-resolution version of the image; it is hyperlinked to the *OrigURL* metadata—that is, the original version of the image on the remote OAI site. This is followed by the image's *Description*, also with a hyperlink; the image's size and type, again generated as metadata by *ImagePlug*; and then *Subject*, *Publisher*, and *Rights* metadata. The result is shown in Figure 14.



JCDL Banquet

Caption:	*... and how Hector Garcia-Molina had sworn to "eat his slide" if the merger was ever pulled off ...* (original 1280x960 JPEG available)
Subject:	JCDL Banquet
Publisher:	JCDL
Rights:	unrestricted

Figure 14. Document in a simple image collection



A-B C D-E F-K L-M P R-S T-U

... and how Hector Garcia-Molina had sworn to "eat his slide" if the merger was ever pulled off ...

... and version 2 of the slide - a "slide cake"

a few words from the awardee

a rapt and rather large audience

a typical paper session

Figure 15. Browsing by *Caption*

There are two browsing classifiers, one based on *Subject* metadata and the other on *Description* metadata (but with a button named "Caption"). Recall that the *AZCompactList* classifier is like *AZList* but generates a bookshelf for duplicate items. In this collections there are a lot of images but only a few different values for *Subject* metadata.

It's a little more surprising that *AZCompactList* is used instead of *AZList* for the *Description* index, because *Description* metadata is usually unique for each image. However, in this collection the same description has occasionally been given to several images, and some of the divisions in an *AZList* would contain a large number of images, slowing down transmission of that page. To avoid this, the compact version of the list is used with some arguments (*mincompact*, *maxcompact*, *mingroup*, *minnesting*) to control the display—e.g. groups (represented by bookshelves) are not formed unless they have at least 5 (*mingroup*) items. To find out the meaning of the other arguments for this classifier, execute the command *classinfo.pl AZCompactList.* The programs *classinfo.pl* (for classifiers) and *pluginfo.pl* (for plugins) are useful tools for learning about the capabilities of Greenstone modules. Note incidentally the backslash in the configuration file, used to indicate a continuation of the previous line.

The *VList* format specification shows the image thumbnail, hyperlinked to the associated document, followed by *Description* metadata; the result can be seen in Figure 15. The *Vlists* for the classifiers use *numleafdocs* to switch between an icon representing several documents (which will appear as a bookshelf) and the thumbnail itself, if there is only one image.

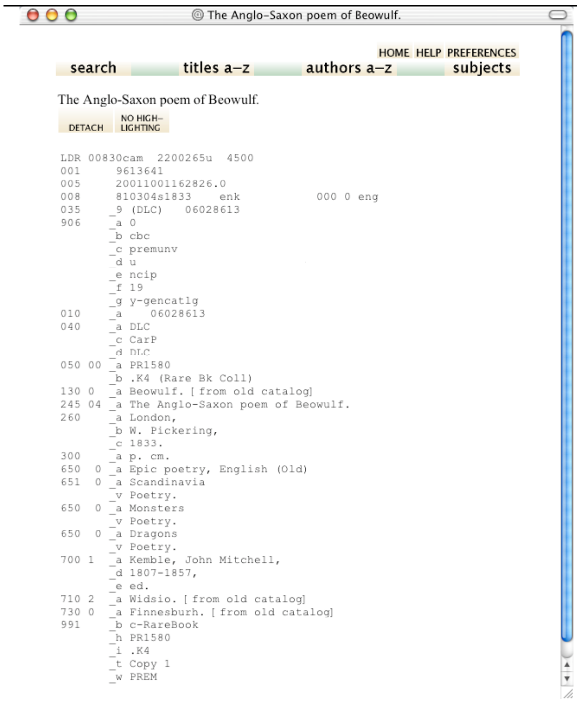
Configuration file

```
creator        davidb@cs.waikato.ac.nz
public         true

collectionmeta collectionname "JCDL 2001 Pictures"
collectionmeta iconcollectionsmall
    "_httpprefix_/collect/demooai/images/jcdl_logo_photo_small.gif"
collectionmeta iconcollection
    "_httpprefix_/collect/demooai/images/jcdl_logo_photo.gif"

collectionmeta collectionextra "This is a demostration collection that exemplifies the
    ImportFrom feature in Greenstone. Using the
    <a href=http://www.openarchives.org>Open Archive Protocol</a> (version 1.1), this
    collection retrieves the metadata available from
    <a href=http://rocky.dlib.vt.edu/~jcdlpix>rocky.dlib.vt.edu/~jcdlpix</a>, a
    collection of photographs taken at the inaugural
    <a href=http://www.acm.org/jcdl/jcdl01/2001>Joint Conference on Digital
    Libraries</a>. Based on the records exported from this OAI data provider (version
    1.1), a Greenstone collection is then built."

acquire OAI -src rocky.dlib.vt.edu/~jcdlpix/cgi-bin/OAI1.1/jcdlpix.pl -getdoc

indexes        document:Description
collectionmeta .document:Description  "photo captions"

plugin         OAIPlug –input_encoding iso_8859_1 –default_language en
plugin         ImagePlug -screenviewsize 300
plugin         GAPlug
plugin         ArcPlug
plugin         RecPlug

classify       AZCompactList -metadata Subject -title Subject -doclevel top
classify       AZCompactList -metadata Description -buttonname Captions\
               -mingroup 10 -mincompact 5 -minnesting 7 -maxcompact 10

format VList   "<td>[link][thumbicon][/link]</td><td
    valign=middle><i>[Description]</i></td>"

format CL1Vlist
    "<td>{If}{[numleafdocs],[link][icon][/link],[link][thumbicon][/link]}</td>
     <td valign=middle>{If}{[numleafdocs],[Title],<i>[Description]</i>}</td>"

format CL2Vlist
    "<td>{If}{[numleafdocs],[link][icon][/link],[link][thumbicon][/link]}</td>
     <td valign=middle>{If}{[numleafdocs],[Title],<i>[Description]</i>}</td>"

format DocumentHeading "<h3>[Subject]</h3>"

format DocumentText
    "<center><table width=_pagewidth_ border=1>
    <tr><td colspan=2 align=center>
     <a href=[OrigURL]>[screenicon]</a></td></tr>
    <tr><td>Caption:</td><td> <i>[Description]</i> <br>
     (<a href=[OrigURL]>original [ImageWidth]x[ImageHeight] [ImageType] available</a>)
     </td></tr>
    <tr><td>Subject:</td><td> [Subject]</td></tr>
    <tr><td>Publisher:</td><td> [Publisher]</td></tr>
    <tr><td>Rights:</td><td> [Rights]</td></tr>
    </table></center>."
```

7. MARC record collection

This collection is based on the MARC records in the Library of Congress Catalog that include Beowulf in their title. Figure 16 shows a sample document in the collection.

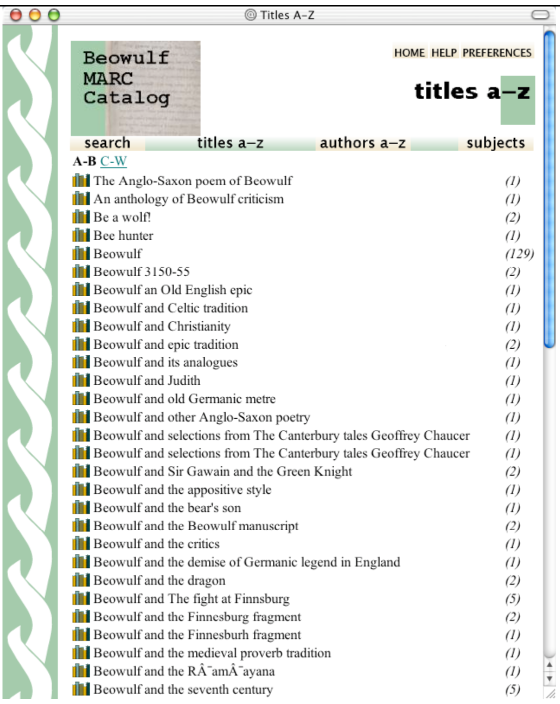
The configuration file uses *ZIPPlug* and *MARCPlug*, apart from the standard three; as in the OAI collection, an *input_encoding* argument is used because data in this archive contains extended characters. There are three classifiers, based on *Title*, *Creator*, and *Subject* metadata. All are *AZCompactList* classifiers, and all specify a *mingroup* of 1—which effectively forces them to create a bookshelf icon even if there is only one item on the shelf. Figure 17 shows an example. The reason is aesthetic: the list has a uniform appearance uninterrupted by any different 1-book entries. (Of course, if you don't like this style you just leave out the *mingroup* argument.) A second argument for the *Title* and *Creator* classifiers removes suffixes from the metadata string (*Title* and *Creator* respectively). This is specified as a PERL regular expression, and trims characters (such as trailing punctuation) from the strings for display. The three format statements are similar: in particular, they each put out the number of leaf documents on the right-hand side of the display, as illustrated in Figure 17.



The Anglo-Saxon poem of Beowulf.

```
LDR 00830cam  2200265u  4500
001      9613641
005      20011001162826.0
008      810304s1833    enk           000 0 eng
035     _9 (DLC)   06028613
906     _a 0
        _b cbc
        _c premunv
        _d u
        _e ncip
        _f 19
        _g y-gencatlg
010     _a    06028613
040     _a DLC
        _c CarP
        _d DLC
050 00  _a PR1580
        _b .K4 (Rare Bk Coll)
130 0   _a Beowulf. [ from old catalog]
245 04  _a The Anglo-Saxon poem of Beowulf.
260     _a London,
        _b W. Pickering,
        _c 1833.
300     _a p. cm.
650  0  _a Epic poetry, English (Old)
651  0  _a Scandinavia
        _v Poetry.
650  0  _a Monsters
        _v Poetry.
650  0  _a Dragons
        _v Poetry.
700 1   _a Kemble, John Mitchell,
        _d 1807-1857,
        _e ed.
710 2   _a Widsio. [ from old catalog]
730 0   _a Finnesburh. [ from old catalog]
991     _b c-RareBook
        _h PR1580
        _i .K4
        _t Copy 1
        _w PREM
```

Figure 16. MARC record document



A-B C-W

The Anglo-Saxon poem of Beowulf	*(1)*
An anthology of Beowulf criticism	*(1)*
Be a wolf!	*(2)*
Bee hunter	*(1)*
Beowulf	*(129)*
Beowulf 3150-55	*(2)*
Beowulf an Old English epic	*(1)*
Beowulf and Celtic tradition	*(1)*
Beowulf and Christianity	*(1)*
Beowulf and epic tradition	*(2)*
Beowulf and its analogues	*(1)*
Beowulf and Judith	*(1)*
Beowulf and old Germanic metre	*(1)*
Beowulf and other Anglo-Saxon poetry	*(1)*
Beowulf and selections from The Canterbury tales Geoffrey Chaucer	*(1)*
Beowulf and selections from The Canterbury tales Geoffrey Chaucer	*(1)*
Beowulf and Sir Gawain and the Green Knight	*(2)*
Beowulf and the appositive style	*(1)*
Beowulf and the bear's son	*(1)*
Beowulf and the Beowulf manuscript	*(2)*
Beowulf and the critics	*(1)*
Beowulf and the demise of Germanic legend in England	*(1)*
Beowulf and the dragon	*(2)*
Beowulf and The fight at Finnsburg	*(5)*
Beowulf and the Finnesburg fragment	*(2)*
Beowulf and the Finnesburh fragment	*(1)*
Beowulf and the medieval proverb tradition	*(1)*
Beowulf and the RÂ¯amÂ¯ayana	*(1)*
Beowulf and the seventh century	*(5)*

Figure 17. Browsing by *Title*

The MARC plugin uses a special file to map MARC field numbers to Greenstone-style metadata. This file resides in the is the *gsdl/etc* directory, and is called *marctodc.txt*. It lists the correspondences between MARC field numbers and Greenstone metadata. Any MARC fields that are not listed simply do not appear as metadata, though they are still present in the Greenstone document. Each line in the file has the format

```
<MARC field number> -> GreenstoneMetadataName
```

Lines in the file that begin with "#" are comments (however, comments have been stripped out of the example file below).

Here is the standard version of this file, which is loosely based on the MARC to Dublin Core mapping found at http://lcweb.loc.gov/marc/dccross.html (which assumes USMARC/MARC21):

gsdl\etc\marctodc.txt

```
720 -> Creator
100 -> Creator
110 -> Creator
111 -> Creator
520 -> Description
856 -> URL
260 -> Publisher
787 -> Relation
540 -> Rights
024 -> MarcIdentifier
786 -> MarcSource
546 -> MarcLanguage
650 -> Subject
653 -> Subject
245 -> Title
655 -> Type
```

Several different MARC fields are mapped on to Dublin Core *Creator*: field 720 is "Uncontrolled name," 100 is "Personal name," 110 is "Corporate name," 111 is "Meeting name." Actual MARC records normally define only one of these fields, and anyway Greenstone allows multi-valued metadata. MARC field 520 ("Summary, note") is mapped to Dublin Core *Description*; field 856 ("Electronic location") is mapped to *URL*; field 787 ("Nonspecific relationship note") to *Relation*; field 540 ("Reproduction note") to *Rights*; field 245 ("Title statement") to *Title*; field 655 ("Index term – genre/form") to *Type*. Both fields 650 ("Subject: topical term") and 653 ("Index term: uncontrolled") are mapped to *Subject*.

MARC field 024 ("Identifier") is not mapped to Greenstone metadata, because Greenstone uses its own *Identifier* metadata; instead it is mapped to a different Greenstone metadata element called *MarcIdentifier*. Likewise field 786 ("Data source entry") is not mapped to *Source*, because Greenstone has *Source* metadata, but to a new metadata field called *MarcSource* instead; and field 546 ("Language") is mapped to *MarcLanguage*.

Cognoscenti will note that some MARC fields with Dublin Core counterparts are simply ignored, e.g. 620 (*Contributor*), 500 (*Coverage*). MARC field 260 is called "Publication, etc") and is mapped in its entirety to *Publisher*. In fact, field 260c (a subfield) is supposed to be publication date, but is not mapped as such.

Of course, different mappings can be defined by altering the above file—which allows the MARC plugin to support other variants of the MARC format. The plugin does not recognize individual MARC subfields: it simply concatenates them together. However, enhancing it to deal appropriately with subfields would not be a difficult job: it would involve altering a couple of pages of PERL code in the MARC plugin.

Configuration file

```
creator        davidb@cs.waikato.ac.nz
public         true

indexes        document:text document:Title
defaultindex   document:text

plugin         ZIPPlug
plugin         GAPlug
plugin         MARCPlug -input_encoding iso_8859_1
plugin         ArcPlug
plugin         RecPlug

classify       AZCompactList -metadata Title   -mingroup 1\
               -removesuffix "(\\s*(\\/|:|;|,|\\.).*)"
classify       AZCompactList -metadata Creator -mingroup 1\
               -removesuffix "(b\\.\\s+)?(\\d+(\\-?))(\\d+(\\.)?)?"

classify       AZCompactList -metadata Subject -mingroup 1

collectionmeta collectionname    "Beowulf"
collectionmeta iconcollection
    "_httpprefix_/collect/beowulf/images/beowulffront.jpg"
collectionmeta collectionextra   "This collection is based on the MARC records in the
    <a href=http://catalog.loc.gov>Library of Congress Catalog</a> that include
    Beowulf in their title."

collectionmeta .document:text   "text"
collectionmeta .document:Title  "titles"

format VList "<td>[link][icon][/link]</td><td>[Title]</td>"

format CL1VList "<td>[link][icon][/link]</td><td>[Title]
    {If}{[Creator], <i>[Creator]</i>}{If}{[Publisher],<i>[Publisher]</i>}</td>
    <td>{If}{[numleafdocs],<i>([numleafdocs])</i>}</td>"
format CL2VList "<td>[link][icon][/link]</td><td>{If}{[numleafdocs],[Title],[Creator];
    <i>[Title]</i>{If}{[Publisher], <i>[Publisher]</i>}}</td>
    <td>{If}{[numleafdocs],<i>([numleafdocs])</i>}</td>"
format CL3VList
    "<td>[link][icon][/link]</td><td>{If}{[numleafdocs]{[Title],[Title]{If}{[Creator],
    [Creator]}{If}{[Publisher], [Publisher]}}</td>
    <td>{If}{[numleafdocs],<i>([numleafdocs])</i>}</td>"
```

Customising Greenstone's appearance

The appearance of Greenstone collections is defined by "macro files" in the *macros* directory, and can be completely altered by changing the macros. As an example, the directory contains a file called *garish.dm* that is used in an alternative version of the demo collection called the "garish" collection.

To separate certain presentation details from the macro files, *garish.dm* uses *Cascading Style Sheets* (see http://www.w3.org/Style/CSS/ for more information), which allow you to specify fonts, colours, spacings, and other elements for HTML pages. A file called *gsdl/images/garish/style.css* contains a rudimentary style sheet for use with *garish.dm* (it is placed in the *images* directory because macro files can easily reference this directory). It includes some comments for those who aren't familiar with cascading style sheets. Some of the macros in *garish.dm* reference images that have been placed in *gsdl/images/garish.*

The *garish* macro file overrides some of the macros that Greenstone uses by default. In macro files, the "#" character signals a comment line (*garish.dm* is commented to help explain what it does). All the definitions in *garish.dm* have [*c=garish*] after the macro name. This gives *garish* as the value of the *c* (for "collection") argument, which means that these definitions apply only to the *garish* collection. Macros themselves are signalled by underscores, and they can reference other macros. This means that the content of a page can be split into many small chunks, and it can be difficult to work through the chain of which macro calls what. However, changes in macros (like changes in format strings) take effect immediately unless you are using the Greenstone local library server, in which case the server needs to be restarted. This makes it easy to experiment by editing the macro files.

Macros are grouped into "packages". But each package is not necessarily in a single file. The *Global* package, which is in file *base.dm* (parts of it are also in the language-specific macro files such as *english.dm* and *french.dm*), contains default macros that can be overridden by macros in other packages. Here is the part of *garish.dm* that redefines global macros.

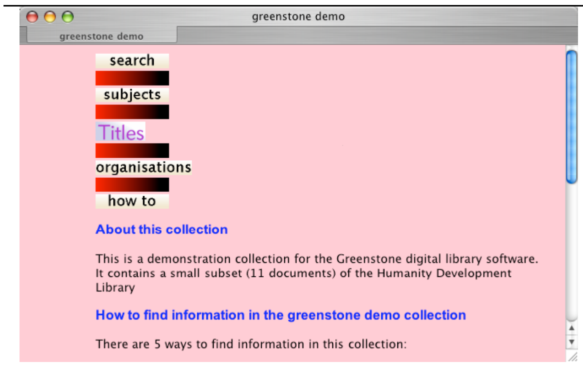
```
#############################################################################
package Global
#############################################################################

_starthighlight_ [c=garish] {<b>}
_endhighlight_   [c=garish] {</b>}

_imagespacer_ [c=garish] { </nobr> <br/>
    <img src="_httpimg_/garish/horizontal_line.gif" width="87" height="17">
    <br/><nobr> }

_httpiconttitlgr_ [c=garish] {_httpimg_/garish/title_green.gif}
_httpiconttitlon_ [c=garish] {_httpiconttitlof_} # make it the same as off
_httpiconttitlof_ [c=garish] {_httpimg_/garish/title_button.gif}

_widthttitlx_ [c=garish] {59}
```

The values of *starthighlight* and *endhighlight* are used to highlight query terms when displaying a document. The default is to set the background to yellow. The first specification above places query terms in bold instead.

Next, the *imagespacer* macro is what separates the search button and classifier buttons in the navigation bar. It is normally Greenstone's familiar horizontal green bar. In fact, the internally-defined *navigationbar* macro is flanked by no-break tags. However, any macro definition can contain arbitrary HTML commands, and these no-break tags can be defeated by inserting appropriate tags for each button:

```
_imagespacer_ [c=garish] { </nobr><br/><nobr> }
```

This has the effect of stacking the buttons vertically, one above the other. The definition of *imagespacer* given earlier uses the image *horizontal_line.gif* as separator (this image appears in the *garish* subdirectory of *images*). The effect can be seen in Figure 18.



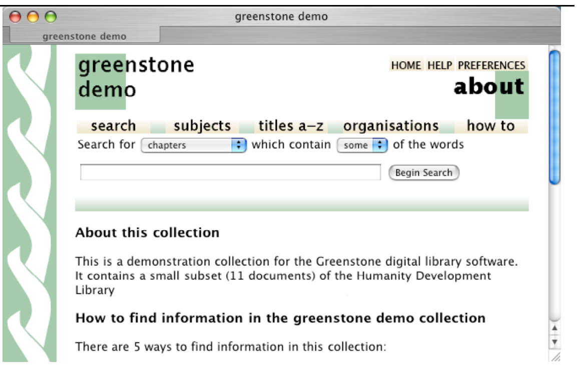
About this collection

This is a demonstration collection for the Greenstone digital library software. It contains a small subset (11 documents) of the Humanity Development Library

How to find information in the greenstone demo collection

There are 5 ways to find information in this collection:

Figure 18. Garish version of the demo collection



About this collection

This is a demonstration collection for the Greenstone digital library software. It contains a small subset (11 documents) of the Humanity Development Library

How to find information in the greenstone demo collection

There are 5 ways to find information in this collection:

Figure 19. The original demo collection

To arrange the buttons horizontally, still using *horizontal_line.gif* instead of the green bar as separator, remove the *nobr* and *br* tags and define the body of the macro (the part in curly brackets) to be

```
<img src="_httpimg_/garish/horizontal_line.gif"
    width="_widthtspace_" height="17">
```

The *widthtspace* variable gives the width of the gap between the buttons, which Greenstone calculates.

The language-specific macro files define all the buttons used in the interface. The *Titles A-Z* button, for instance, has three macros, one for green (for when it is the selected button), another for off (when it is not selected), and a third for on (when the mouse is over it). Their names are rather cryptic, and all begin with *httpicon*. The three lines in the code above that start with this text override the three *Title A-Z* buttons, giving the effect shown in Figure 18. The *width* value is used when calculating the gap between buttons.

These button definitions do not have a language parameter, which means that they apply only when the interface is in English (the default language). To override the French versions, specifications like

```
_httpiconttitlgr_ [c=garish,l=fr]
    {_httpimg_/garish/fr_title_green.gif}
```

would also be needed. The file *english.dm* gives a list of graphical button names—they all begin with *httpicon*.

The part of *garish.dm* reproduced below redefines macros in the *Style* package, which is responsible for creating the header and footer of every page:

```
################################################################################
package Style
################################################################################

_htmlhead_ [c=garish] {<html>
<head>
   <title>_pagetitle_</title>
   <link rel="stylesheet" href="_httpimg_/garish/style.css" type="text/css"/>
 </head>
<body>
}

_pagebanner_ [c=garish] {}
```

Greenstone's page header macro, which is called *header* in *style.dm*, prints the collection's name or logo, and links to the home, help, and preferences pages. It calls the macro *htmlhead* which outputs the beginning of an HTML file. The above specification overrides *htmlhead* so that all pages use the new style sheet. The page header macro also calls *pagebanner* to include the collection's logo, home/help/preferences buttons, and the image at the top left that identifies the page (about page, search page, etc). The code above redefines the banner to suppress these images—as you can see, they are absent in Figure 18.

The next part of *garish.dm* redefines macros in the *about* package (contained in *about.dm*), which generates the "About this collection" page in Figure 18:

```
################################################################################
package about
################################################################################

_content_ [c=garish] {
_navigationbar_

_textabout_

<h3>_help:textsimplehelpheading_</h3>
_help:simplehelp_
}
```

The macro *content* for the *About* page normally contains the navigation bar (with links to *Search* and any classifiers), followed by "About this collection" and "how to find information" text. Because it changes from one collection to another, the *navigationbar* macro is defined internally by Greenstone, and uses the *imagespacer* macro discussed above. The *content* macro for the *About* page given above differs from the regular one shown in Figure 19 in that the navigation bar is not centered, and no search box is included.

The next part of *garish.dm* redefines macros in the *query* package (in *query.dm*), which is used to generate the search page:

```
################################################################################
package query
################################################################################

_content_ [c=garish] {
_navigationbar_

<center>
_If_(_cgiargct_,_selectqueryform_,_queryform_)
</center>
_If_(_searchhistorylist_,<center>_iconsearchhistorybar_</center><br>
<center>
_searchhistorylist_
</center>)
_If_(_cgiargq_,<small>_freqmsg_ _textpostprocess_</small><br>_resultline_,)
<br>
}
```

The definition of *content* in the standard *query* package is similar to the version given above, except that it centers the *navigationbar* macro, whereas the above version does not, and it includes an additional green bar. The difference can be seen in Figures 20 and 21.

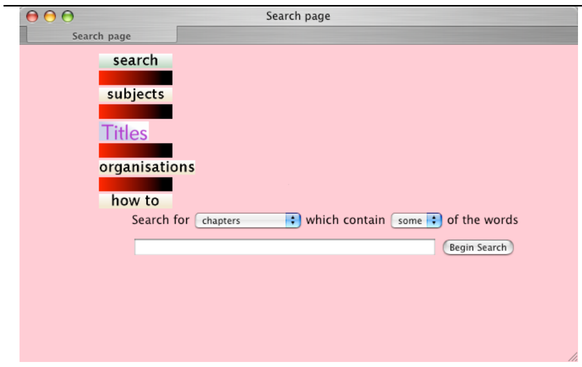


Figure 20. Garish search page

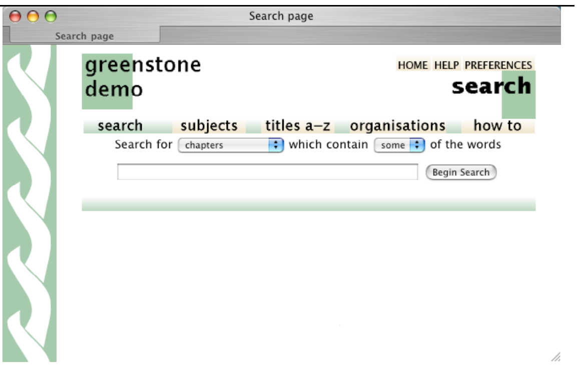

Search page
greenstone demo
HOME HELP PREFERENCES
search
search subjects titles a–z organisations how to
Search for chapters which contain some of the words
Begin Search

Figure 21. Original search page

The file *garish.dm* also redefines macros in the *document* package, which is used to display documents (see *document.dm*). In fact, pages generated by classifiers (e.g. lists of titles) are also governed by these macros.

```
##############################################################################
package document
##############################################################################

_textheader_ [c=garish] {
_cgihead_
_Global:header_
}

_content_ [c=garish] {
_navigationbar_
<p>
<center>
_phindclassifier_
</center>
}
```

Greenstone overrides the *header* macro if a document (and not a classifier) is being displayed. Suppose we don't want it to. The code above redefines the *textheader* macro (normally defined in *document.dm*). The "Global:" forces the use of the *Global* version of the header macro instead of the "document" version, which adds in the normal links and navigation buttons found on the *About* and *Search* pages. The definition of *content* above overrides the default so that the navigation bar is not centered.