NOTE 215 – AIPS++ Documentation Notes

W. Young

Nov 12, 1997

Contents

1 Overview

The *AIPS++*documentation is a combination of standards and tools. The standards are HTML, LaTeX, and *AIPS++*guidelines. The tools are cxx2html, help2tex, and latex2html, with a dash of shell programming thrown in. All *AIPS++*documentation is available via a web interface. Most of the user and system documentation and *AIPS++*Notes and Memos are written using the LaTeX mark up language. Some older documents may be in texinfo which was abandoned in favor of LaTeX. C++ classes are documented in header files using special cxx2html tags.

Cxx2html, help2tex, and latex2html are all perl scripts. A working knowledge of perl is helpful to understand them and essential for debugging.

All documentation source is found in the "code" tree. The make system's docsys target generates and moves postscript/html files into the "docs" tree.

2 LaTeX based documents

2.1 General Documentation

2.1.1 General Notes

Documents in the *AIPS++*Notes and Memo series are mostly written using LaTeX. All new documents should be written in LaTeX. There are a few notes that are plain ASCII (.txt) files. The make system generates postscript files from files with .latex and .tex extensions. Makefile.doc located in code/install contains all the rules for generating postscript and html files and moving them into the "docs" tree.

The *AIPS++*Notes and Memos series each contain a latex file that has the titles and authors for each Note and Memo. Each time a note or memo is checked in, the index.tex file in code/doc/memos/memos.dir and code/doc/notes/notes.dir needs to be edited to include the new note or memo.

2.1.2 Generating HTML

HTML documents are not automatically generated from LaTeX source. The author needs to add a line to the directory's makefile similar to:

```
documentroot := -split +2
```

where documentroot is the file's name less the .latex extension. After specifying this line latex2html will be run on that document during a sneeze. Most *AIPS++*documents should use

```
-split +1 or +2.
```

The default behavior of latex2html may altered using at .latex2html-init file. Typically the default split level, and appearance of the HTML pager header and footer will be customized.

Latex2html uses absolute path names to its icons, the shell script redoicons.sh (code/install/docutils) turns the absolute paths into relative ones. Using relative icons avoids have latex2html generated HTML pages use NRAO as an icon server. To get relative icons sites should specify two variables in thier makedefs file ICONSERVER and ICONS2LOCAL. ICONSERVER is the ICONSERVER variable specified in the latex2html script. At the AOC, we have the following two lines in our makedefs to generate local icons:

```
ICONSERVER := http://www.nrao.edu/icons/latex2html
ICONS2LOCAL := $(AIPSARCH)/bin/redoicons.sh
```

For more details about the html commands available from latex2html please consult the latex2html documentation found at http://www-dsed.llnl.gov/files/programs/unix/latex2html/manual/manual.html.

Latex2html should be found in the system area (/opt/local/bin/latex2html at the AOC). *AIPS++*uses version 98.2beta5 (1998-Jul-28). Latex2html requires several utilities

- Perl (5.003 or later).
- dvips (5.516 or later, preferably 5.62 or later)
- gs (4.03 or later).
- netpbm (1 March 1994 version)

Please consult the README distributed with latex2html for more details.

2.1.3 HTML Links to Other Documents

There are several methods for producing links to other documents. To reference an HTML document, use:
`\htmladdnormallink{text}{link to html page}`.
Provided an author uses `\label` commands you may reference other LaTeX based document using `\externallabels` command and `\htmlref`.

To make a link to the copy command of the catalog object in the catalog module use `\htmlref`

`\htmlref{copy}{catalog:catalog.copy.function}`

The `\externallabels{URL to document}{path to labels file}` command needs to appear at the top of the document before the `\begin{document}`. The URL and path to labels file need to be relative paths. So for Note 215

`\externallabels{../../user/Refman}{../../user/Refman/labels.pl}`

Help files auto-generate some labels based on the following scheme:

- module
- module:function
- module:tool
- module:object.method

There are also two special case where a function and constructor have the same name:

- module:tool.constructor
- module:tool.function

Note the constructor and function are added to the end of the label, i.e. images:image.calc.constructor or images:image.calc.function to identify them uniquely. Additional labels may be put in the .help file by the author using the `\label` command.

The order of when documents are run through latex2html is important for external cross referencing. All released documentation needs two passes through latex2html to resolve all external references (insures the use of the proper labels.pl file).

Funny things can happen if labels are multiply defined.

Cross referencing into HTML files created by cxx2html is possible using `\htmladdnormallink`. To make a reference to the array class array class from a note or memo use:

Please consult the cxx2html documentation to see how anchors are created.

Cross referencing from an HTML document to something in the User's Reference Manual or another LaTeX based document is possible. The link must be identified in the generated HTML. This is not recommended since latex2html generated file names and anchors will likely change.

A word of caution, relative links in the docs tree may be different than those found in a programmer's code tree. Within the code/doc tree relative links should be OK (i.e. builds of LaTeX documents should cross reference properly amongst themselves for programmer and system builds). Relative links to html pages generated by cxx2html will not be the same for programmer and system builds! The implement directory is missing in the docs tree.

2.2 Adding Notes and Memos

Contributed by Wim Brouw, ATNF
It took me a while to insert a Note in the system and considering the difficulty others have had, I make the following points:

1. Select a note number (I just take the next number available, but maybe there is somebody to ask?) - say nnn (e.g. 225)

2. Create (check in) directory code/doc/notes/nnn.dir *(If it's there then others will know the number is taken.)*

3. Put the body of your latex Note (without embracing document information like documentstyle, title, `\begin{document}` etc) in nnn.dir (e.g. ai -l mynote.latex)

4. Create, (in code/doc/notes itself) a file called nnn.latex containing the document information, and which includes mynote.latex (without a directory!) Example:

```
\documentclass[11pt]{article}
\usepackage{html, epsf}
%%----------------------------------------------------------------------------

\begin{document}

\title{NOTE 224 -- AIPS++ Least Squares background}


\author{Wim Brouw}
```

```
\date{22 January 1999}

\maketitle
%%-----------------------------------------------------------------------------
\begin{htmlonly}
\htmladdnormallink{A postscript version of this note is available
(124kB).}{../224.ps}
\end{htmlonly}

\tableofcontents

\input{lsq.latex}
\end{document}
```

5. Add your note number to the code/doc/notes/makefile (just do like the other notes, e.g.:

   ```
   224 := -split 0
   ```

6. Test if all ok, by making a Postscript version: gmake nnn.ps
   And an html version:
   gmake nnn

7. If all is ok, then add your Note to the Notes index:

   ```
      code/doc/notes/notes.dir/nindex.tex
      E.g:
   \item[224]
   \htmladdnormallink{\textit{AIPS++ Least Squares background}}{../224/224.html}
   \linebreak  1999/01/25 Brouw
   ```

Note: Other notes, AIPS++ documents can include your note or parts of your note by adding the following line to the makefile

```
TIROOT := $(word 1, $(AIPSPATH))/code/doc
EXTRA_TEXINPUTS := $(TIROOT)/memos/111.dir:$(TIROOT)/notes/156.dir
/notes/196.dir
```

you then use the `\input{file}` *to include it the other document. The HowTos (code/doc/reference/HowTos.latex) is a good example.*

2.3 Help System

The help system is a special type of LaTeX document. User documentation for packages, modules, objects, and functions are written in .help files. Authors use the aips2help.sty package to format their help text. The perl script help2tex is run on the .help file to produce "standard" LaTeX so no special modules are needed by latex2html. Help2tex provides a generate consistent looking documentation if the aips2help.sty commands and environments are used.

2.3.1 How it works

The .help files are usually found in same directory as the "code". The docsys target for code/doc/Refman has been enhanced to do the following:

1. Collect all the .help files into a temporary directory.
2. run help2tex (a perl script, code/install/docutils) on each package in the system. Help2tex is run run twice on each package first to to generate a .htex file (found in docs/user/helpfiles) and then to generate an atoms.g file used for glish command-line help.
3. run latex to generate the postscript file in docs/user/Refman.ps.
4. run latex2html to generate the html files in docs/user/Refman.

2.3.2 Adding New Packages

The reference manual has been broken into several books due to pool space limitation of LaTeX. Creating a new reference book requires several steps.

1. Create the top-level Package.latex file similar to General.latex. You will need to either include the package.help file in this new Package.latex file or write some text.
2. edit the makefile in code/doc/user
   (a) add the new package to the HELPPACKS variable
   (b) define a PACKAGE.s2ps variable (will create lots of ps.gz files for package, module, and methods)
   (c) set the split level of Package.latex file (should be := -split +4 -link 2 -short_index)

(d) edit Refman.latex to add the new package to the top-level user reference manual.

(e) if needed add the mmands to other .latex files in the system (only necessary if you wish to cross reference from a .latex document to the User Reference Manual).

3 HTML based documents

3.1 Top-level WWW pages

Several WWW pages are found in code/doc. These pages are "straight" HTML and copied directly into the docs tree. Each *AIPS++*web page has a standard header which contains a navigation bar for moving around the documentation tree. There are hooks in for javascript menus but these have proven to be problematical (i.e. unrepeatable problems with page display)

Here are the top-level *AIPS++*web pages relative code/doc:

aips++.html *AIPS++*top-level documentation page.

gettingstarted.html how to get *AIPS++*and use it.

user/documentation.html user documentation material.

glish/glish.html glish specific material.

learnmore.html where to get more information about *AIPS++*.

programmer/programmer.html programmer and system documentation.

contactus/contactus.html defect status, email reflectors, personnel, and mirrors.

newsletters/index.html latest news from *AIPS++*.

faq/faq.html frequentyl asked questions about *AIPS++*.

search/search.html Search pages for the documentation.

Specific informational HTML pages hang off of these control pages. Please try and keep the web tree shallow to avoid burying it.

3.2 Cxx2html generated documents

C++ class documentation is contained in the C++ header files. *AIPS++*uses the perl script cxx2html (code/install/docutils) to extract and assemble the class documentation.

4 Searching *AIPS++*Documentation

*AIPS++*uses the htdig system (http://htdig.sdsu.edu) to perform searches. Htdig traverses the web from a specified starting point (*AIPS++*home page) and creates a database of words and documents. A daily cron job (/aips++/local/htdig/bin/rundig) is run to recreate the databases. The configuration file (/aips++/local/htdig/conf/htdig.conf) offers many ways to tune the database. The search page supplied with htdig has been customized for *AIPS++*and checked into the system (code/doc/html/search.html).

Root privileges are needed to install/modify htsearch in the httpd/cgi-bin directory and add the htdig icons (/tarzan/httpd/icons).

All document searching is currently done on tarzan.aoc.nrao.edu. Distributing the search database with the system is not contemplated until there is a demand for searching local documentation.

5 *AIPS++*Problem Reporting System

The *AIPS++*problem reporting system uses ClearDDTS as the underlying system for managing and tracking problems (This replaces the older GNATs system). There are three ways to submit "a defect":

1. using ClearDDTS, or
2. using the iAIPS++ bug web page, or
3. bug() (from glish).

Extensive hard-copy documentation is available for ClearDDTs.

6 Mailing list setup

From time-to-time new mailing lists are need. The process is as follows

1. Check-in the new mailing list in code/admin/system.

2. Supply the mail alias to NRAO-AOC helpdesk), they put the alias into the master alias table. *AIPS++*mailing lists have the form aips2-mailinglist, where mailinglist is the name of the mailing list.

3. On tarzan as aips2mgr, edit the /etc/mail/tarzan.aliases-tail file and put in three lines like

```
#A comment about the new mailinglist
aips2-newlist: :include:/export/aips++/master/etc/aips2-newlist.list
aips2-newlist-log: "| /export/aips++/master/etc/aipsmail aips2-newlist"
```

4. A daily cron job run by root appends the tarzan.aliases-tail file to tarzan's alias list.

5. Send a test message. Put a symbolic link to the list recipients. On tarzan, cd /export/aips++/Mail/aips2-newlist, ln -s ../../master/etc/aips2-newlist list

More details on *AIPS++*mailing lists may be found in the *AIPS++*System Manual.