Proposed Changes for CXX2HTML

W. Young and G. van Diepen

February 15, 1996

Contents

1 Introduction

AIPS++ wants to put as much programmer documentation into existing header files as possible. Cxx2html extracts much of the documentation putting it into HTML files that mirror the include files. For the general class description cxx2html is adequate in its present form. However the class summaries and member descriptions could be improved with some formatting and presentation changes.

2 Summary of changes

The following points summarize the changes we propose for Cxx2html:

1. Each header file should produce a summary file of each class. Individual files containing the full description should be created for each class. We suggest the following naming convention, file-summary.html for the summary file, file-class.html for the full class description. We propose private enums, typedefs, and functions be made available via a switch on cxx2html. The default setting of the "private" switch would be off.

2. All class and summary files should have the following AIPS++ specific information:

   (a) pointer to AIPS++ programmer manual
   (b) pointer to package document (when we get one)
   (c) pointer to module document
   (d) include path <package/module/file.h>
   (e) copyright notice
   (f) pointer to bug reporting system
   (g) date and how it was generated.

3. The header summary files should contain:

   (a) List of classes
   (b) Global Enums
   (c) Global Typedefs
   (d) Global Functions

(e) Class summaries

   i. Public Enums
   ii. Public Typedefs
   iii. Public Functions (including inherited functions)
   iv. Protected Enums
   v. Protected Typedefs
   vi. Protected Functions (including inherited functions)
   vii. Public Data
   viii. Protected Data
   ix. Optional Private Enums
   x. Optional Private Typedefs
   xi. Optional Private Functions (including inherited functions)

4. Formatting changes to summary listing; function names are aligned with return types right justified.

5. The class description files should contain:

   (a) only one class per file
   (b) Standard header descriptions
   (c) Functions descriptions in the order specified by summary file. No need for inherited functions though.
   (d) Global functions are put at the end of the class file. If there are multiple classes, global functions should be put in the appropriate active class.

6. Functions maybe grouped and the group named. Groups may be nested. Groups should be identified with a label and indentation scheme. Nested groups in the summary file could be identified with a label that prepends the parent group(s). (We don't feel very strongly about nested groups but, believe cxx2html should support nested groups at sometime. An example, not from AIPS++ code, may be found in http://tarzan.aoc.nrao.edu/~wyoung/protopages/nestedgroups.html.)

7. Formatting changes to member descriptions; argument types are aligned with one argument per line, also horizontal lines separate the functions.

8. The current module documentation will need some revision. Discussion of those revisions is beyond the scope of this document.

9. The "project" level document would tie in all module documentation. Discussions of the project document are also beyond the scope of this document.

3 Justification

We believe the case for changing the formatting is self-evident. Please look at the Table.h, TableVector.h, and Array.h examples on Wes Young's home page (http://tarzan.aoc.nrao.edu/~wyoung/). In summary the improvements are

1. Return types and arguments are more easily identified.
2. Function descriptions are bound more tightly to the functions they describe.
3. Grouped information is easier to identify.

We propose to split the summary listings from the descriptive text and split the descriptive text for each class into separate files. This allows us to produce summary only hardcopy and selective class hardcopy. The current output of cxx2html precludes these options.
We recognize that the documentation system will be used by both code maintainers and developers as well as applications programmers (library users). Each type of programmer has different needs in documentation; maintainers and developers often need access to all information about a class while the application write really only cares about the public and sometimes protected interface of a class. Implementing a user specified switch to display the private information should satisfy both classes of programmers.

4 Details

The HTML that comes out of cxx2html should be HTML 3.0 compliant as much as possible.

4.1 Header and Footer Information

References to the "AIPS++ Programming Manual", the package, and the module are useful for aips++ but, not necessarily for other users. Copyright and problem reporting should also be customizable. Either a switch, a tag in the documentation, or a .cxx2htmlrc file would be acceptable to tell cxx2html whether to produce (and what to put in) these header and footer labels in the HTML output.

4.2 Summary File

We suggest the following sketchy HTML for the header summary files. An example of the summary file may be found at
http://tarzan.aoc.nrao.edu/~wyoung/protopages/TableVectorSum1.html.

```
<!doctype html public "-//IETF//DTD HTML 3.0//EN">
<html>
<head>
!
! style will hopefully allow us to get rid of the netscape table extensions
!
<style notation=css>
H1: text-color = red
TABLE {border-style: none;}
TABLE {padding: 0px;}
TABLE {border-width: 0px;}
</style>
</head>
<body>
!
!At the top
! anchor to Programmer Reference Manual
! anchor to Package document, anchor to Module
! include path for .h file
!
<a href="../../RefManual.html#Tables">AIPS++ Programmer Reference Manual</a>
<h1>Package <a href="../../Aips.html#aips"> aips </a> - Module
    <a href="../Tables.html#TableVector">Tables</a></h1>
<h1> &lt;aips/Tables/TableVector.h&gt; </h1>
<h1>Classes</h1>
<dl>
!List of classes with anchors
</dl>
<hr>
```

```
<h1> Class name with anchors to source code or
<h1> Interface
!
! For the summary table
!define a table with return types right justified and functions left justified
!
<table border=0 cellspacing=0 cellpadding=0>
!
<tr><td align=left><h3>Public Functions</tr>
<tr><td align=right valign=top>return type <td align=left>function with arguments
!
! A Group
!
<tr><td align=left><a href="#group_name"><strong>group name</strong></a></tr>
!
<tr><td align=left><h3>Inherited Functions</tr>
<tr><td align=right valign=top>return type <td align=left>function with arguments
!
<tr><td align=left><h3>Protected Functions</tr>
!
<tr><td align=left><h3>Public Data</tr>
!
<tr><td align=left><h3>Protected Data</tr>
</table>

<hr>
<a href="http://tarzan.aoc.nrao.edu/cgi-bin/wwwgnats.pl"> Comments, suggestions,
and bug reports</a>
<p>Generated on yyyy mmm dd by cxx2html version x.x
<P>Copyright &#169; 1995 Associated Universities Inc., Washington, D.C.</P>
</body>
</html>
```

4.3 Class Description file

We suggest the following sketchy HTML for the class descriptions files. An example of class description file may be found at http://tarzan.aoc.nrao.edu/~wyoung/protopages/split1.html.

```
<!doctype html public "-//IETF//DTD HTML 3.0//EN">
```

```
<html>
<head>
!
! style will hopefully allow us to get rid of the netscape table extensions
!
<style notation=css>
H1: text-color = red
TABLE {border-style: none;}
TABLE {padding: 0px;}
TABLE {border-width: 0px;}
</style>
</head>
<body>
!
!At the top
! anchor to Programmer Reference Manual
! anchor to Package document, anchor to Module
! include path for .h file
!
<a href="../../RefManual.html#Tables">AIPS++ Programmer Reference Manual</a>
<h1>Package <a href="../../Aips.html#aips"> aips </a> - Module
    <a href="../Tables.html#TableVector">Tables</a></h1>
<h1> include file with anchor back to summary file
<hr>

<h1> Class name with anchor to source file or html file
<hr>
<h1> Description
Pretty much as is now
!
! For the member descriptions
!
<hr><hr>
<h1> Member Descriptions </h1>
<hr>
<h2> Public Functions <h2>
<dl compact>
!
! For functions with zero or one argument
```

```
!
<dt><a name="...">Function()
<p><dd>
Description of function
<p>
<hr>
!
! For function with more than one argument we want them to
! line up one under the other so we use a table
<dt>
<table border=0 cellsizing=0 cellpadding=0>
<tr><td align=right valign=top><a name="..."Return type function( <td align=left>
     type arg1,
<tr><td><td align=left>type arg2,
<tr><td><td align=left>type last_arg)
</table>
<p><dd>
Description of function
<p>
<hr>
!
! Grouped functions
!
<dt>
<h3><a name="group_name">Group</a></h3>
Description of group
<hr>
<dl>
!
! Do function as above, Groups maybe nested similarly
!
!End of dl for group functions
</dl>
! Last function put <hr> after </dl>
<hr>

!End of dl for all functions
</dl>
!
```

```
! Last one in definition list put <hr> after </dl>
<h2> Protected Functions</h>
<dl>
!
!  Protected function definitions here
</dl>
<hr>
<a href="http://tarzan.aoc.nrao.edu/cgi-bin/wwwgnats.pl"> Comments, suggestions,
   and bug reports</a>
<p>Generated on yyyy mmm dd by cxx2html version x.x
<P>Copyright &#169; 1995 Associated Universities Inc., Washington, D.C.</P>
</body>
</html>
```

4.4 Inheritance

Identifying inherited and virtual functions is very important. We propose that in the first pass when building the database, cxx2html could also extract all public and protected functions and store their signatures in the database. It could mark virtual functions. In the second pass it could look in the database at the parent(s) of a class, their parents, etc.. and do the following:

- Always precede a virtual function with the word virtual.
- Make a link on the word virtual to link it to the same function in the nearest parent.
- Show inherited functions in the public and protected summary. This could be done by preceding the function with the word inherited and making a link on that word similar as above. In that way inherited functions do not need their own category in the summary and are correctly listed as public or protected.

5 Priorities

We propose that the changes have the following priorities

1. Summary and description layout changes,

2. A method to customize header and footer information with AIPS++ specific information (package, module, bug reporting, copyright and the like),
3. Split files into summary and detailed description,
4. Class function groups,
5. Include base class functions in derived class summary,
6. Switch to display private functions,
7. Nested classes, and
8. Nested function groups.