Paper to HTML—An Automatic, Seamless Process for Documentation Production

Virginie Ahrens
ILOG S.A.
9, rue de Verdun
94250 Gentilly
FRANCE
Tel: +33 1 49 08 35 14
ahrens@ilog.fr

Valérie Lecompte
ILOG S.A.
1681, Route des Dolines
06560 Valbonne-Sophia Antipolis
FRANCE
Tel. +33 4 92 96 61 53
lecompte@ilog.fr

1. ABSTRACT

This paper describes how ILOG, a French software company designing C++ and Java class libraries, managed the transition between paper-only documentation and extensive HTML online documentation in less than two years. In this paper, we analyze the underlying reasons for making this change, describe the technological choices that were made, and walk through the various steps of the project from its beginning to final completion.

1.1 Keywords

C++ and Java class libraries, Online Documentation, Java Script, HTML, Page-Authoring Tools, Web Design, Modularity, Portability, Reusability.

2. Background

ILOG products are multi-platform C++ and Java class libraries that enable developers to design GUIs and expert systems, or to solve complex optimization problems. Until 1997, the documentation of all ILOG products was provided in paper format only. Because the volume of documentation was quickly growing both in terms of quantity of pages and in number of manuals, the documentation department decided to provide online documentation in addition to the paper manuals. By doing so, we felt that online documentation would facilitate navigation across several manuals and would make the large volume of our documentation more easily accessible. In addition, we wanted our online manuals to convey the new visual identity of ILOG in a consistent manner, to be intuitive to use, and to be easily generated without any post-processing after the conversion. We faced the following challenges:

- We had to find the right format that would ensure the portability of our online documentation.
- We wanted to make the move to online without disrupting the normal writing tasks of the technical writers.
- We wanted to continue to use our FrameMaker source files because we needed them to generate other required output such as PDF files.

3. Selecting a Format and a Tool

It was necessary to find a format that is highly portable, that does not tie our customers to a specific document viewer, and that loads quickly and integrates well with other development tools. HTML was the logical choice that naturally met all these environmental constraints.

To convert our paper manuals to online documentation, we chose WebWorks Publisher from Quadralay for the following reasons:

- Our legacy paper manuals are all in FrameMaker. One of the input formats that WebWorks accepts is MIF, the Maker Interchange Format of FrameMaker.
- WebWorks is robust enough to handle huge books (some of our manuals are more that 1000 pages with large indexes).
- WebWorks easily converts FrameMaker cross-references and markers into hypertext links.
- It is a highly-customizable tool. WebWorks can be used to generate HTML that matches complex paragraph and character styles, and supports drawings, imported graphics, and equations.

4. Online Documentation Structure

Because numerous ILOG customers use several of our products simultaneously, we wanted each product documentation suite to have the same structure and to provide the same navigation mechanisms. We also wanted our users to know at any time which product documentation suite they are consulting, and more precisely which chapter from which manual. To do so, we designed the following structure:

- Each documentation suite has a product home page that

lists the manuals that are available for the product.

- The standard online documentation page features a three-pane window including a top frame that always displays the product name and version number as well as the current manual title, a side frame that displays the Table of Contents of the current manual, and a central frame that shows the page highlighted in the Table of Contents.
- HTML files are split at the level of chapters and at heading levels 1 and 2. At the top of each file, the current chapter title is displayed in the form of a running header placed in a color text inset.

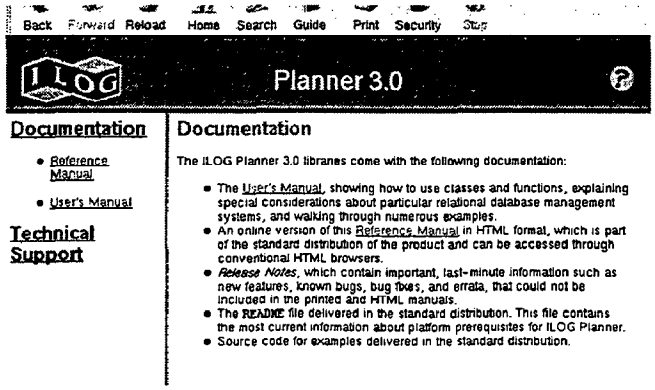


Figure 1. ILOG Planner 3.0 Documentation Home Page

We designed the following navigation mechanisms:

- Within a manual, navigation is enabled through a set of navigation arrows placed at the top and at the bottom of each HTML page. In addition to hypertext links, specific buttons grant access to the home page, the table of contents and the index of the current manual.
- For navigation across manuals, the top frame features a drop-down list box containing the various manuals that are available, and a button to access the documentation home page that also provides access to the various manuals. In addition, hypertext links placed in the text allow navigation between a user's manual and a reference manual.

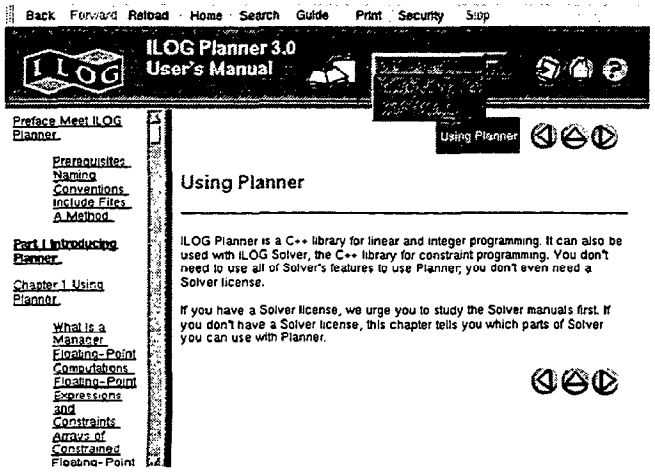


Figure 2. Navigation Mechanisms

5. Building an Efficient Team

As we began this project, it became apparent that special skills would be required to meet our goals. The customization of WebWorks Publisher required advanced knowledge of HTML and browser behavior. To implement the navigation mechanisms and the three-pane format, Java Script programming was required. To control the appearance of fonts and ensure a consistent look across all our online documentation, we needed to use CSS (Cascading Style Sheets).

To meet these requirement, we hired a software developer who became a full-time documentation tool developer within the documentation team. The role of the documentation tool developer consists of gathering information concerning the needs of the technical writers, specifying and implementing the conversion process, and delivering a customized tool that that enables the technical writers to automatically convert their FrameMaker files to HTML.

6. Technical Implementation

The conversion process to HTML consists of mapping each FrameMaker element to an HTML structure. Before doing this, we did the following in our legacy files:

- We studied them carefully to retrieve a consistent set of paragraph and character tags, and "froze" the style sheet. Style sheet changes were controlled by the documentation manager and transmitted to the documentation tool developer.
- We examined each element of our FrameMaker style sheets and specified how we wanted them to look in HTML. We had to pay particular attention to portability issues as we wanted the generated HTML to display correctly in both Explorer and in Netscape versions 3 and 4 on Unix and on Windows platforms.

We then customized WebWorks Publisher specifically to accommodate our reference and user's manuals styles, as the set of predefined conversion formats available in WebWorks was not adapted to our needs.

The following examples show the extensive customization that was performed in WebWorks. For each element, we show how it appears in the paper manual, explain its conversion in WebWorks, and show how the result of the conversion is displayed in a browser.

6.1 Notes and Warnings

Both elements are very similar in FrameMaker: a paragraph tag is embedded within a table. The table allows a vertical side bar to be displayed in front of the note and the warning tags.

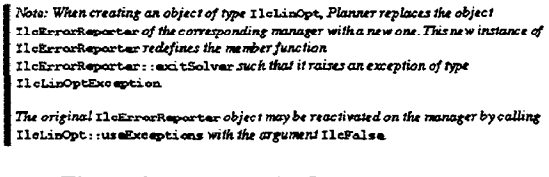


Figure 3. Note Tag in FrameMaker

In HTML, we have created a table in front of which we place a graphic showing a visual cue representing the note or the warning. The note or warning text is displayed in boldface type next to the graphic.

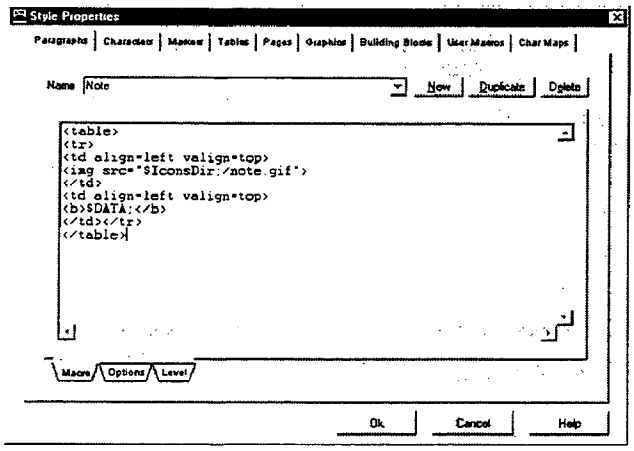


Figure 4. Mapping Definition of a Note

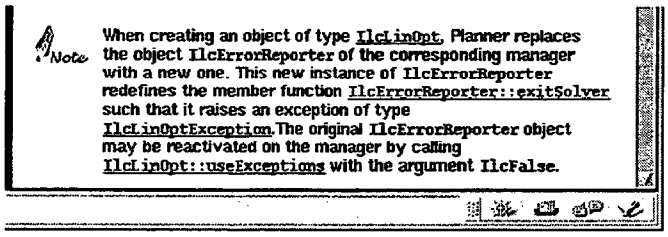


Figure 5. Note Tag in Generated HTML

6.2 Code

In FrameMaker, code is formatted as a series of paragraphs in Courier typeface surrounded by two horizontal lines. This style is mapped to a single HTML table that displays the code with a light color background.

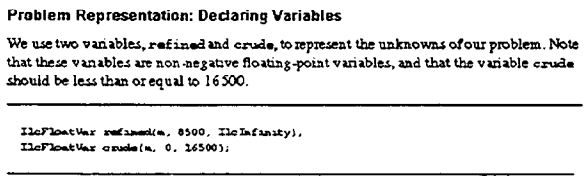


Figure 6. Code in FrameMaker

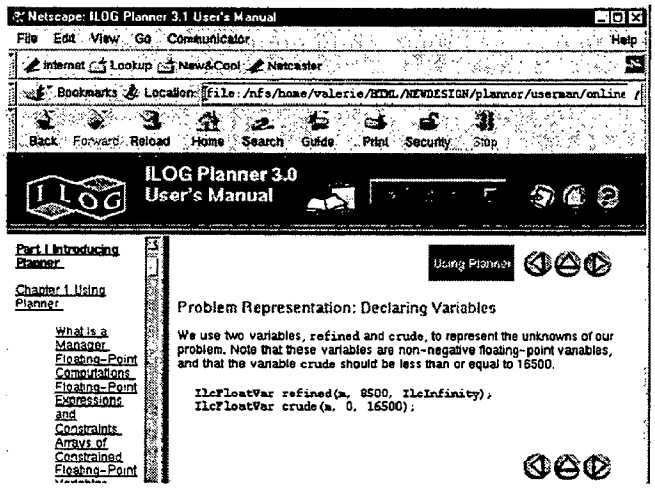


Figure 7. Code in Generated HTML

6.3 Equations

Equations are entered directly in FrameMaker using the equations module. They are converted to GIF files and inserted in the generated HTML.

WebWorks Publisher provides a function called @MAKEIMAGE that calculates the GIF image.

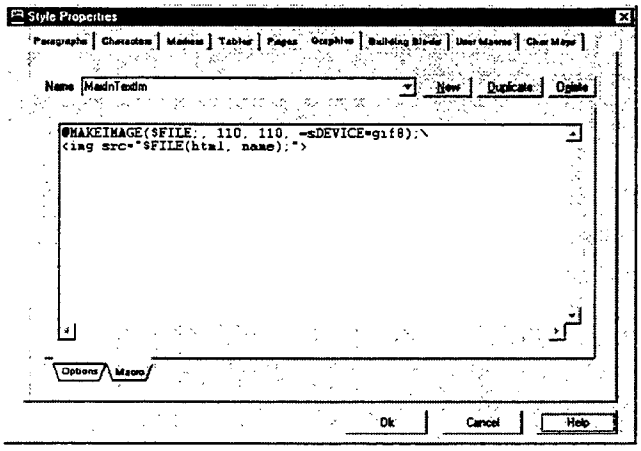


Figure 8. Mapping Definition of an Equation

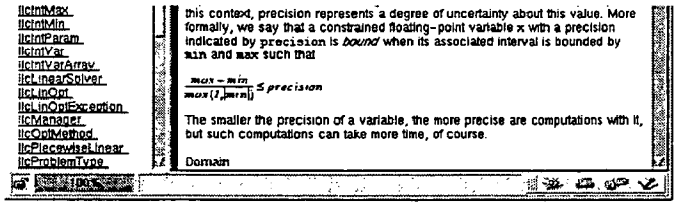
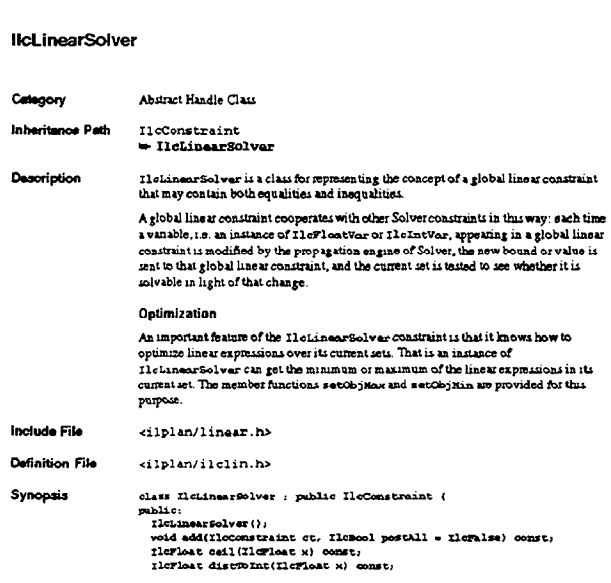


Figure 9. Equation in Generated HTML

6.4 Special Characters

Our paper manuals include special characters that we wanted to keep in the generated HTML. For example, we use a special arrow in the Reference Manuals that shows for each class which classes derive from it. In FrameMaker, inheritance arrows are typed using Esc Shift+a (the \xe5 character in the Zapf Dingbats font).



IlcLinearSolver

Category Abstract Handle Class

Inheritance Path
IlcConstraint
➥ **IlcLinearSolver**

Description
`IlcLinearSolver` is a class for representing the concept of a global linear constraint that may contain both equalities and inequalities.

A global linear constraint cooperates with other Solver constraints in this way: each time a variable, i.e. an instance of `IlcFloatVar` or `IlcIntVar`, appearing in a global linear constraint is modified by the propagation engine of Solver, the new bound or value is sent to that global linear constraint, and the current set is tested to see whether it is solvable in light of that change.

Optimization

An important feature of the `IlcLinearSolver` constraint is that it knows how to optimize linear expressions over its current sets. That is an instance of `IlcLinearSolver` can get the minimum or maximum of the linear expressions in its current set. The member functions `setObjMax` and `setObjMin` are provided for this purpose.

Include File `<ilplan/linear.h>`

Definition File `<ilplan/ilclin.h>`

Synopsis

```
class IlcLinearSolver : public IlcConstraint {
public:
  IlcLinearSolver();
  void add(IlcConstraint ct, IlcBool postAll = IlcFalse) const;
  IlcFloat ceil(IlcFloat x) const;
  IlcFloat distToInt(IlcFloat x) const;
```

Figure 10. Page with Inheritance Arrow in FrameMaker

Because this font does not exist in HTML, we had to replace each arrow with a GIF image.

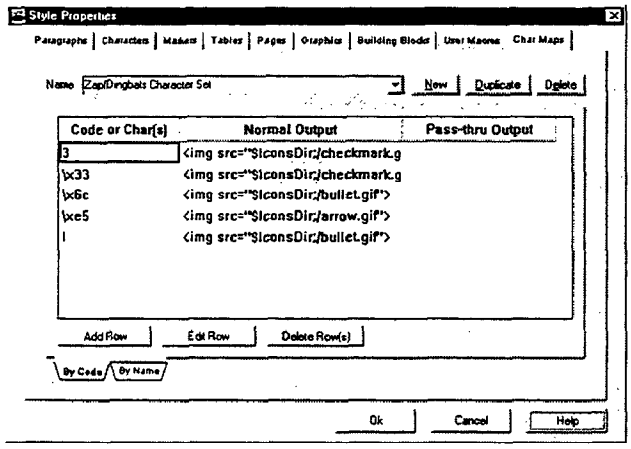


Figure 11. Mapping of Special Characters

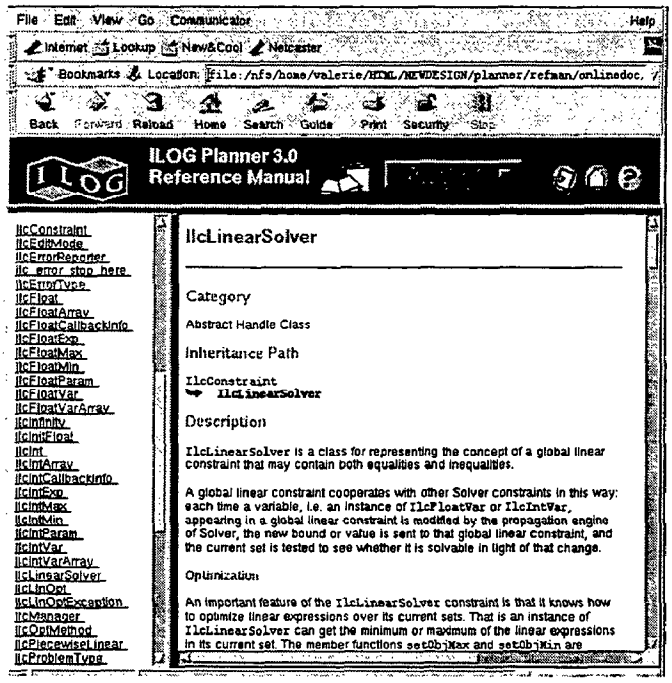


Figure 12. Page with Inheritance Arrow in HTML

6.5 Graphics

Several types of graphics are used in our documentation, whether screen dumps in TIF, GIF or EPS format, or drawings made with the FrameMaker tool box. Some of the graphic formats, like GIF and JPEG, can be used directly in the generated HTML files. Others need to be converted.

Depending on the purpose and the size of an image, we want it to appear either in the middle of the text (this is the case of icons or buttons) or in its own frame like a paragraph (for screen dumps). The default for displaying images in HTML is to align them on the left margin, but in some cases we need to center them. This has caused us to define several mapping styles for images. The technical writer can chose among those styles how the image is to be converted and inserted within the generated HTML.

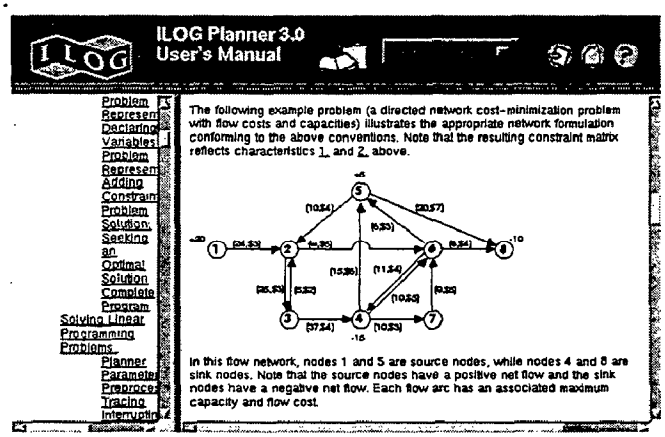


Figure 13. Graphic in Generated HTML

6.6 Hypertext Links

All FrameMaker marker types (cross-references, hypertext links, user-defined markers) are supported by WebWorks and converted into hypertext links. We make extensive use of user-defined hypertext links in FrameMaker to set links to other manuals and to code samples, or to open Java applets. All links that have been set in our FrameMaker files are automatically processed upon conversion.

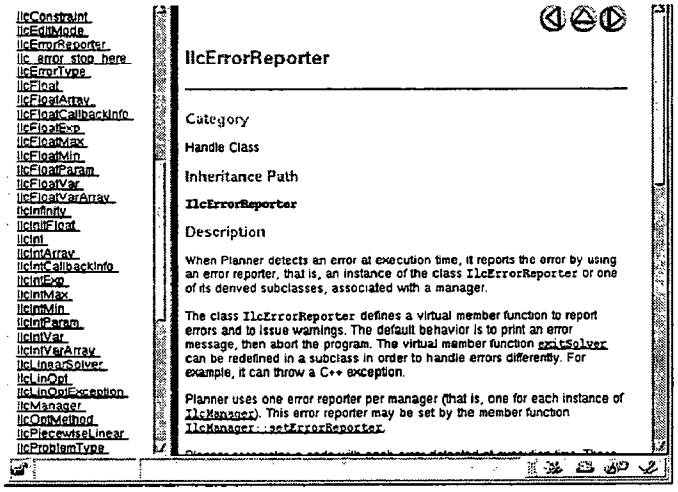


Figure 14. Hypertext Links in Generated HTML

6.7 Index

Our documentation includes indexes that are sometimes more than 100 pages long. Because of this large volume, we decided to create an HTML file for each index letter, as opposed to the single index file produced in FrameMaker.

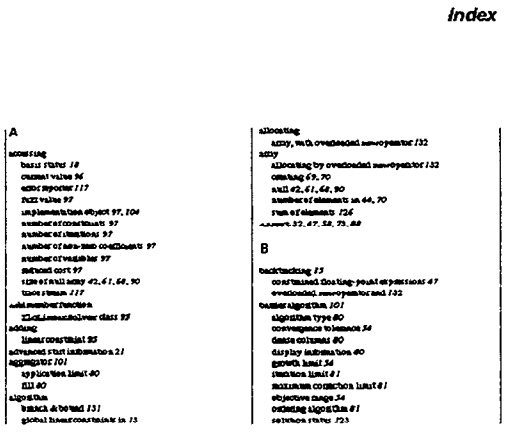


Figure 15. Index in FrameMaker

Our generated HTML index includes a index cover page which lists the letter entries contained in the index. Each letter is an hypertext link that opens the corresponding index file. This cover index page is generated automatically in the following way:

- Each time an index letter tagged `IndexLetter` is converted, a hypertext link corresponding to this letter is added to the HTML index cover page.

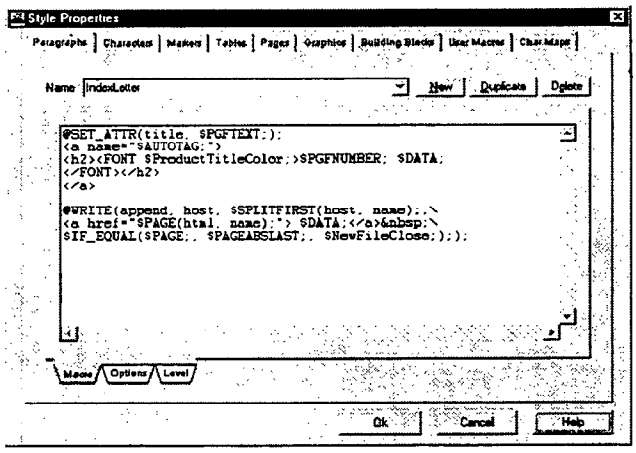


Figure 16. Mapping Index Letters

- Then, each index page is processed separately. When the index is generated in FrameMaker, hypertext links are automatically set between page numbers and their corresponding index entry. As page numbers are not relevant in online documentation, we have replaced them with a GIF image that contains a hypertext link to the index page.

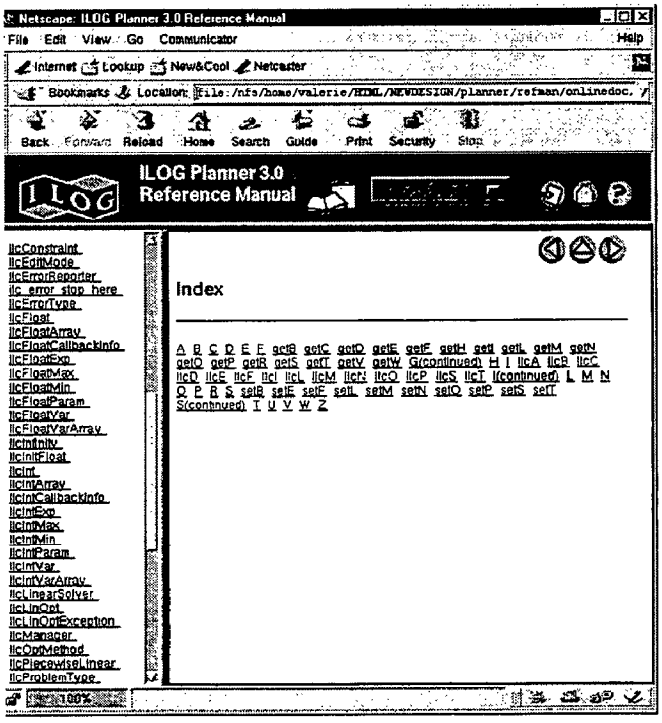


Figure 17. HTML Index Cover Page

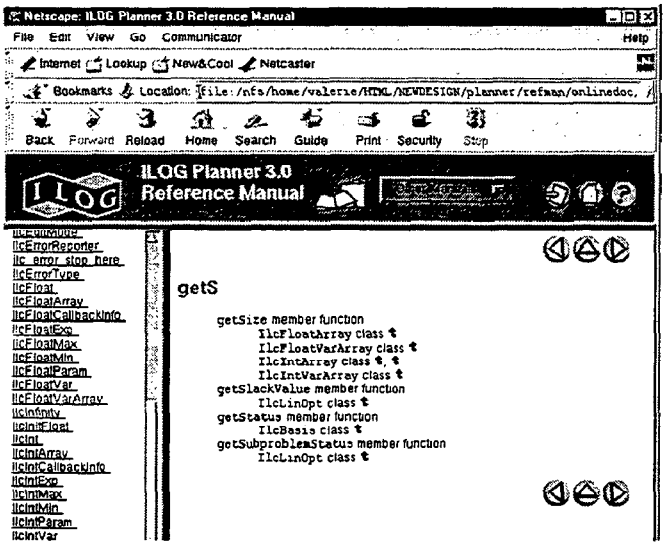


Figure 18. A Graphic Replaces Page Numbers in HTML Index

6.8 Multi-frame Structure

The Table of Contents, which is located in a separate frame, is a file that is automatically generated by FrameMaker with the default option that creates an hypertext link between each TOC entry and its corresponding paragraph in the manual.

Each TOC entry is tagged with a specific paragraph: ChapterTitleTOC, Heading1TOC, ClassNameTOC, etc. When each paragraph tag is processed by WebWorks, we modify the HTML generation and instruct the converter to build a multi-frame link that will make the TOC entries and their corresponding paragraphs appear in two separate frames.

We create this multi-frame display by entering the instruction `target="ManualBody"` in the hypertext link definition.

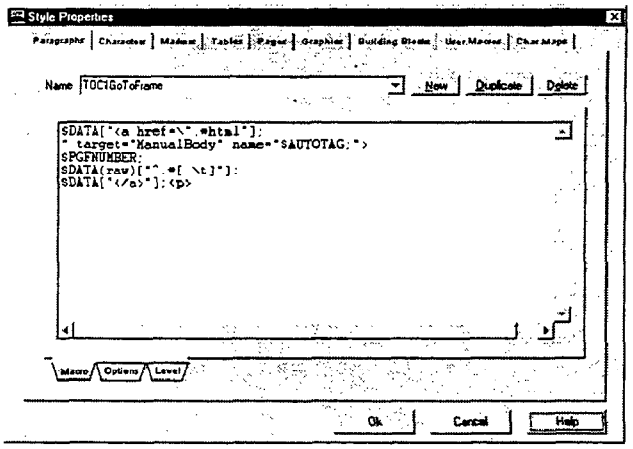


Figure 19. Setting the Appearance of the Table of Contents

7. Benefits

Moving from paper-only documentation to online has brought us the following benefits:

- **A component-based, modular approach** Because ILOG products can be used in combination, online documentation reflects how our products are intertwined through extensive use of hypertext links. Hypertext links exist not only between manuals of the same product suite, but also across various product suites and are complemented by a comprehensive cross-product index.
- **Ease-of-use for our customers** In HTML, the large volume of our documentation is not readily apparent. Navigation across documents is facilitated by hypertext links, the chunking of information into small text units, and a page length that minimizes scrolling within each HTML file.
- **A gradual, non-disruptive progression** We first converted reference manuals only, then extended the conversion to a whole suite of manuals. Finally, we included several documentation suites and made them accessible from a general ILOG home page. This smooth progression was possible because the actual writing tasks take place in FrameMaker, while the HTML mappings and Web design are handled in WebWorks Publisher.
- **A Reusable Knowledge Base** We gained extensive experience from developing Java Script functions, processing images, using high-level HTML features like Cascading Style Sheets, and providing HTML pages that can be read from a variety of browsers. This knowledge will help us meet the new challenges we will face in the future of moving to XML, creating multimedia tutorials, and designing HTML-based Help.