Question: Why did they call it FOSI?
Answer: All the other four-letter words were taken.

CHAPTER 1
Introduction

FOSI is Arbortext Editor's native formatting language. It is designed for documents with consistent formatting. As discussed in **Chapter 2**, FOSI is a standardized stylesheet language that supports a batch formatting process.

In Arbortext Editor, FOSI is used to specify formatting for:

- the Edit window display
- printed output
- output to a PDF file
- overriding formatting that is automatically inserted in an HTML file created with File→Save as HTML

Styler™ and XSL-FO stylesheets are mapped to FOSI code in Arbortext Editor. However, Styler™ and XSL-FO do not support all the formatting available with a native FOSI stylesheet. Native FOSI provides more formatting capabilities with Arbortext Editor and faster performance as well.

Short comparisons of FOSI with Styler™, XSL-FO, and Advanced Print Publisher are presented below, beginning on page 3.

First, different formatting processes are discussed followed by an introduction to FOSI formatting features.

TIP

Unlike XSL-FO and Advanced Print Publisher, it is virtually impossible to develop a FOSI stylesheet that is inefficient and slows the formatting process. A highly skilled programmer is not needed to ensure optimum performance. That is because FOSI is a declarative language; efficiency and performance are handled by the interpreting software. The main issue with a FOSI stylesheet is crafting it so it is easy to maintain in the future, which is addressed in this book.

TIP

Things to like about FOSI: no nested parentheses; minimal need to comment; little or no configuration needed.

TRIVIA

Some FOSIs have been in continuous use since the mid–90s.

Interactive versus batch formatting

With interactive publishing software, each page is reviewed in order to tweak the formatting and adjust page breaks as needed.

The alternative to interactive, or attended, formatting is batch formatting, or unattended, formatting. Batch formatting is a "black box" process that runs without human intervention. FOSI stylesheet formatting composes pages in an unattended process that has been dubbed "lights out."

Batch formatting is generally less expensive than attended formatting. The up-front cost of developing a FOSI stylesheet is soon paid off as the FOSI is utilized over and over again.

Layout- versus document-based formatting

Formatting for layout-based documents such as posters or advertisements is generally designed specifically for the content. Such documents are not well suited to SGML/XML, and their formatting is not well suited to FOSI or batch formatting. FOSI is best suited to documents with consistent formatting.

FOSI formatting features

FOSI provides many formatting features, which are listed below and described in more detail later in this book.

- paragraph building (hyphenation and justification)
- page layout
- footnotes
- floated objects
- suppression of element content
- reordering of element content
- generated text, including indexing
- generated graphics
- generated tables
- generated external ASCII file
- cross references
- revision bars
- security classification markings

- conditional formatting
- atthbute-based formatting
- attributes output as text
- Arbortext Command Language (ACL) variable-based formatting
- ACL function calls

FOSI compared to Styler™

Arbortext's Styler™ is an interface that lets you specify formatting without knowing the FOSI language. A Styler stylesheet is ultimately transformed into native FOSI. However, the interface limits the formatting capabilities.

Styler does allow some FOSI source coding, but of course you have to know FOSI to take advantage of this.

FOSI compared to XSL-FO

FOSI and XSL-FO are declarative languages, which describe the problem to be solved while leaving the solution to the interpreting software. While both have some programmatic capabilities, XSL-FO is far more programmatic than FOSI by itself. However, FOSI with ACL is just as powerful, can support greater formatting capabilities, and generally runs much faster than XSL-FO.

XSL-FO does not directly output the SGML/XML document. XSL-FO requires XSLT coding to build an output document from the input document. This approach, which requires a transformation process before formatting is applied, may be overkill for most text documents.

FOSI formatting takes the opposite approach, assuming that SGML/XML documents are essentially linear and that everything should appear in input order unless otherwise specified. The formatter builds the output document so the FOSI developer doesn't have to. Every element in a document is output unless it is deliberately suppressed by FOSI coding. If desired, FOSI can suppress elements in a document and output their content in a different order. However, FOSI has no separate transformation process like XSLT.

In Arbortext Editor, XSL–FO coding is transformed into native FOSI, but not all FOSI capabilities are available with an XSL-FO stylesheet. Also, FOSI has some built-in features that must be created from scratch with XSL-FO.

XSL-FO is a runtime language with a transformation process as well as a formatting process, with the output then mapped to FOSI. The process is slow and requires a lot of computing resources. It is not very well suited for lengthy and/or on-demand output. FOSI is a compiled language, so documents are formatted quickly, making FOSI suitable for print-on-demand applications.

To learn XSL-FO, you must first learn XSLT, which are difficult to learn and use. You have to master a great deal of material in order to format even a simple document. With FOSI, once you understand a few basic concepts, you only have to learn as much as the formatting requirements demand.

For FOSI development, Arbortext Editor provides the tagged editor and the style panels interface, which can be used interchangeably. Arbortext Editor has only the tagged editor for XSL-FO.

With XSL-FO, formatting speed depends on the skill of the developer. With FOSI, performance optimization is handled by the formatter.

FOSI compared to Advanced Print Publisher

Advanced Print Publisher (APP) and FOSI have little in common. APP is designed to handle complex page layout requirements and provide formatting capabilities associated with fine, hand-tuned typography. FOSI is designed to batch format documents with consistent page layouts.

FOSI is a separate stylesheet. APP consists of embedded processing instructions that support document-specific or template-based formatting.

APP can be batch like FOSI or interactive, including post-composition changes to content, formatting and layout. Note that post–composition changes can be a risky practice. If the content in the database is not updated, the database and published documents become out of sync.

APP uses a proprietary language and has a steep learning curve. And, as with XSL but unlike FOSI, formatting speed depends on the skill of the developer.