Apache™ FOP Design: Introduction

Version 1298724

by Keiron Liddle

Table of contents

> Note:
>
> The articles in this section pertain mainly to the *redesign* or *trunk* line of development. The redesign is mainly focusing on parts of the layout process (converting the FO tree into the Area Tree). Therefore other (non-layout) sections in this document are probably largely accurate for the maintenance branch, but should be used with care in that context.

1 The Black Box View

From a user's standpoint, Apache™ FOP is a black box that an xml file as input, performs some magic, then creates the desired output:

Process	Result
.	XSL-FO document
FOP	Output: PDF, Postscript, Print, etc.

Table 1: Apache# FOP from a User's Standpoint

Although this is simple, it is useful in defining the outer limits of FOP's core processing. There may be other things going on under FOP's control that are not really part of FOP. For example, FOP provides a convenience mechanism that takes semantic XML + an XSLT transformation as input, instead of XSL-FO. This is done outside of FOP's core processing (by Xalan), and it is therefore outside the scope of FOP's design, and outside the scope of the FOP design documents.

2 Primary Design Goals

A discussion of project design properly begins with a list of the goals of the project. Out of these goals will flow the design issues and details, and eventually, the implementation.

2.1 Conformance to the XSL-FO Specification

The current design goal is to reach the "basic" level of conformance, and to have enough flexibility in the design to reach "complete" conformance without major rewriting. After "basic" conformance is achieved, it is probable that higher levels of conformance will be sought.

2.2 Process Files of Arbitrary Size

Except for user storage limitations, the design goal is to be able to process files of any size. In a separate but related issue, the design goal is to be able to process page-sequence elements of any size. (See Recycling FO Tree Memory for a discussion of the use of page-sequence as a logical subdivided "chunk" on an FO document).

3 Secondary Design Goals

3.1 Minimize Memory Use

Many FOP design decisions revolve around trying to minimize the use of memory. The primary purpose here is to reduce the amount of data that must be serialized to storage during processing. Since our primary design goals include the ability to process files of arbitrary size, there is no way to avoid the need to serialize. However, many FOP users provide web access to documents that are created in real time. Performance is therefore an important issue in these real-world applications. To the extent that it can be done so without jeopardizing the primary design goals, FOP developers have identified keeping a small memory footprint as being an important secondary goal.

4 The Big Picture View

With our design goals outlined, we'll now open the Black Box and look at the major processes inside. FOP has adopted the basic structure of the XSL-FO standard itself as a convenient model for the major processes in FOP. The Result in each row is the input for the next.

Process	Process Result/Input for Next	Notes
.	XSL-FO document	.
Parsing	FO Tree	.
Refinement	Refined FO Tree	.
Layout	Area Tree	Layout and Area Tree are not needed or used for the structural outputs (MIF and RTF), as they are not paginated.
Renderer	Output: PDF, Postscript, Print, etc.	.

Table 1: FOP's Big Picture Design

In general, each piece of data will be processed in the same way. However, some information may be used more than once, and some may be used out of order. To reduce memory, one process may start before the previous process is completed.

For a detailed discussion of the design of any component, follow its link in the table above. Each component outlines the design issues which have already been addressed. These resolution of these design issues is in support of the primary and secondary goals, so they are not necessarily written in stone. However, most of them have been discussed at length among the developers, and are reasonably well settled.

5 Vocabulary

This section will attempt to provide information about any jargon used in the design documentation.

There is a rough relationship between terms used to describe the various trees in XSL-FO processing, all of which come from the XML and XSL-FO standards. In the table below, the *terms* (but not the actual items) in each column are roughly equivalent to each other:

Tree Concept	Thing (Noun)	Descriptive Item (Adjective)
XML	Element	Attribute
FO Tree	Object	Property
Area Tree	Area	Trait

- **LM**: Layout Manager.

- **PLB**: PropertyListBuilder.