Testing FOP

1. "Build" Testing

Apache projects use an automated build tool called "gump" to create nightly builds from the CVS repository. Gump sends "nag" messages if the build fails. This can be considered a sort of basic test of the build process. To view the most recent logs of the gump builds:

- Gump build for the Trunk
- Gump build for the Maintenance Branch

2. Functional Testing

2.1. Running and Using Tests

Testing is an important part of getting FOP to operate correctly and conform to the necessary standards.

A testing system has been set up that works with as a build target when developing with FOP. A developer can run the tests after making changes to the code, the aim is to have the tests run to verfiy that nothing working has been broken.

To setup the testing the developer must place a reference fop.jar in the "<cvs_repository>/test/reference/" directory. This jar will be dynamically loaded to create the reference output.

2.2. W3C TestSuite

The testing is set up so that you can download the testsuite from http://www.w3.org/Style/XSL/TestSuite/, unzip the file into the base directory of FOP. Then you can uncomment the lines in the build.xml file in the test target and itwill run through all the tests in the testsuite distribution.

2.3. Writing a Test

A test belongs to one of a few catagories. A basic test should excercise one element in a number of situations such as changing a property. This should have at least one normal value,

one border value and one invalid value. If the property can be of different types then this should also be included.

A bug test is a test that is specifically aimed at a problem with FOP. That is, the test is not excercising the specification but rather a problem with FOP in handling a particular situation that is not exposed with the other testing.

A system test is one that tests the abitlity of FOP to handle a number of different elements together.

A test can consist of a complete fo document or a part of the document such as some elements that will be placed into the flow of a standard document.

2.4. Submitting a Test

If you have a test which you think would be useful you should supply the test and a diff to the appropriate test suite xml file. Make sure that the test works as would be expected against the current build.

2.5. How Testing Works

The tests are stored in the "<cvs_repository>/test" directory.

You can run the tests by specifying the build target "test" ie:

```
build.sh test
```

This will then compare the current code in the local src directory to a specified release of FOP. Any differences between the current code and the output from the reference version will be reported. If the test previously passed then the test run will have failed.

The testing is done by reading a test suite xml file, which corresponds to the standard testsuite.dtd supplied from w3c. This xml file contains a test xml file and an xsl file (which may simply copy the file). It also contains information such as if the test has passed and any comments.

For FOP the testing is done by rendering all the testing documents using the XML renderer. The XML files are then compared to see if there are any differences.

2.6. SVG Testing

The testing of SVG is not part of this testing system. SVG is tested for its rendering accuracy by using the transcoding mechanism via Batik. So that the only part that needs testing is how the SVG image is embedded inside the flow of the fo document.