Antenna House

Regression Testing System

User Guide

Antenna House Regression Testing System

User Guide V1.5

About this Publication

This publication was created by Michael Hougentogler in Greenville, DE. It was designed and edited to serve as an example of the capabilities of Antenna House's "Cloud Authoring Service for the Universal Book" (CAS-UB). Aside from the cover and some additional graphical elements created using Adobe Photoshop, the author limited himself to the features and built-in style sheets inherent to CAS-UB.

About the Antenna House Regression Testing System

The Antenna House Regression Testing System (AHRTS) is a fast and scalable solution for automating the visual comparison of formatted documents or pages. It is not a solution for tracking changes or locating variations in the content of multiple versions of the same document—its main function is to identify the visual dissimilarities that exist between a pair or set of documents. That being said, AHRTS does have some capability to locate content discrepancies within a limited set of parameters.

AHRTS was originally developed to meet Antenna House's requirement for regression testing new releases of our Formatter software. We are now making AHRTS available commercially to meet the needs of customers who must also perform regression testing of their output whenever any aspect of their production process has been altered. This includes not only the output tool, but also the editing, content management and graphic tools that are used. A slight change in any of the soft- and/or hardware of a system could have unintentional consequences for the integrity of the final output—text, graphics and tables could be altered unexpectedly. Determining whether or not changes occurred during the production process can be a very time consuming and tedious task. If the changes are minute, they may go undetected and ultimately lead to errors in the information being distributed.

Originally designed to work with Antenna House Formatter, the software can now be used with any PDF output software. This includes formatting software such as Antenna House Formatter and desktop publishing systems, to business applications, report generators, graphic packages, word processing software and every other imaginable application that generates a PDF file. A change to any of your system components must be regression tested to ensure the integrity of the resulting product.

What makes this application special is its visually-oriented solution. AHRTS performs a precision pixel-by-pixel comparison of two PDF files and generates a report that allows for the quick visual identification of any discrepancies that exist between the compared documents. This system greatly increases the efficiency of regression testing, reducing the amount of time and effort required to check output discrepancies. AHRTS enables users to perform accurate regression tests by using a more comprehensive test set that can be processed in hours as opposed to days. It will enable users to find even the slightest change in the visual appearance of a document.

AHRTS can also be used to test that changes made to any of the style sheets and/or software produce the desired results. Has a change in margins, a shift of lines or a modification of fonts been applied as expected? Have they produced any unforeseen results? These are cases where the regression testing tool via its visual output can quickly help determine how the changes have been applied, and if any unintended consequences have occurred.

AHRTS is available in four versions. The table below displays the capabilities of each:

Feature	Standard	Plus	Expanded	Distributed
Easy to Use GUI	X	X	X	X
Compare 2 Versions of the Same PDF	X	X	X	X
Compare Multiple PDFs Contained in Directories		X	X	X
Compare Documents Using 2 Versions of Formatter			X	X
Command-line Interface			X	X
Distributed Regression Testing across Multiple Systems to Speed				X

Testing				

The Standard, Plus and Expanded are offered both as Standalone and Server versions. The Expanded and Distributed version includes an API to enable integration of AHRTS into the tool chain. A Standalone license includes a GUI and may be used on a laptop, desktop or workstation by an individual. The Server license is intended for use by multiple people.

Automated Regression Testing and its Benefits

The traditional solution to the regression testing of a formatted paged output has been to compare visually two documents side-by-side and page-by-page to ensure changes in the software have not disrupted the production process in any way. It involves testing either a subset or large collection of documents. The purpose is to compare the visual output to determine if the new file corresponds to, or diverges from, the reference file.

The challenge is that different underlying code can produce the same output. To the best of our knowledge, there has been no commercial product available to manage the regression testing of large numbers of documents.

Traditional regression testing is time consuming, costly, resource intensive and unreliable. It is a monotonous chore that nobody ever wants to do; nevertheless, it is vital to ensure an error-free final product. The fact that this approach is prone to missing minor—but often critical—differences can lead to delays in product release.

The Benefits of Automated Regression Testing

- 90%+ reduction in human effort
- Greatly reduces processing time
- Significantly more accurate
- Able to detect subtle differences
- Easy multiple re-testing
- Transforms what was a labor-intensive chore into a quick and easy process

Table of Contents

Installation

Windows

Prerequisites

Standard/Plus Versions	Expanded/Distributed Versions
Windows XP or later Oracle Java Virtual Machine 7 or greater PDF Viewer	Same as Standard/Plus The two or more versions of AH Formatter you will be regression testing

Installation Steps

Standard/Plus Versions	Expanded/Distributed Versions
1. Download the latest version of AHRTS by contacting us at info@antennahouse.com 2. Run the *ahrts-install.jar* installer by double-clicking the file 3. Follow the steps indicated by the install wizard 4. Copy the license key provided to you into the installation directory	1. Same as Standard/Plus 2. Manually setup location of the versions of Antenna House Formatter in the engine configuration file.*

Linux

Prerequisites

Standard/Plus versions	Expanded/Distributed Versions
Oracle Java Virtual Machine 7 or greater PDF Viewer	Same as Standard/Plus The two or more versions of AH Formatter you will be regression testing

Unix versions, except Mac OS X, need xdg-open from the xgd-utils (freedesktop.org) package in order to automatically open PDF files. If xgd-open isn't available an alternate PDF viewing application can be specified in the 'Settings' menu or in the ahrts.properties configuration file.

Installation Steps

Standard/Plus versions	Expanded/Distributed Versions
1. Download the latest version of AHRTS by contacting us at info@antennahouse.com 2. Run the *ahrts-install.jar* file by double-clicking it or running *java -jar ahrts-install.jar* from the command line 3. Follow the steps indicated by the install wizard 4. Copy the license key provided to you into the installation directory	1. Same as Standard/Plus 2. Manually setup location of the versions of Antenna House Formatter in the engine configuration file.*

* If Antenna House Formatter is installed in the default directories, this step is not necessary.

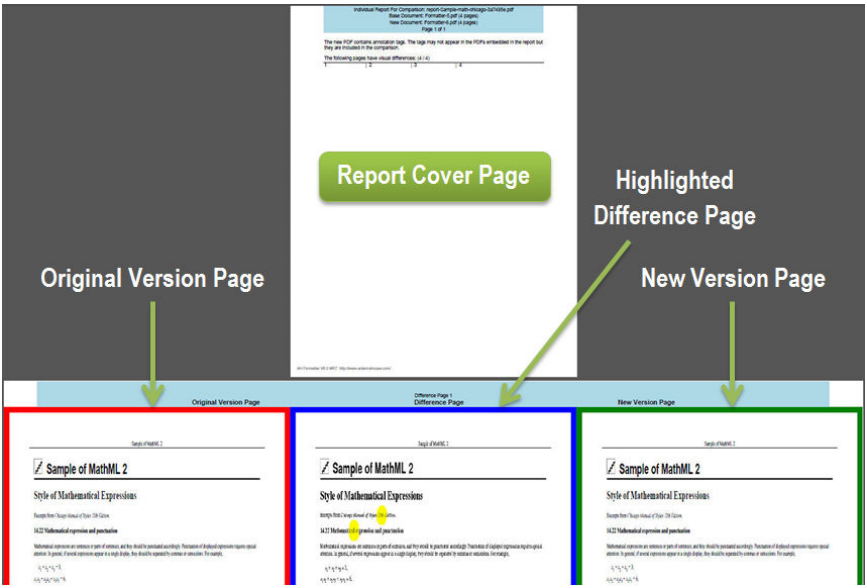
Understanding the Reports

Reports

The compare step generates both an Overview and Individual Document Report. The Overview Report is created only when multiple documents are compared. It displays which documents contain differences, and which don't. The Individual Document Report is a report for each set of compared documents that have been identified as having differences. If you want to send the report PDFs to someone or move them you can, provided you keep it in the same relative position as the rest of the output. An easy way to do this is to create an archive (both .zip and .tar work well) of the output directory and move that to the desired location.

Individual Document Reports

An individual document report is generated for each document containing a difference. The report has a cover page indicating which pages of the document contain discrepancies, and then an additional report page for every page of the tested document with a difference. (In cases where no discrepancies have been identified, only a cover page will be produced.)



Reading the Color-Coded Reports

Both the left and right panes are the actual pages from the original PDFs and thus have all the color attributes that were in the original documents. In the center is a composite/ difference image which is comprised of a grayscale version of the baseline image. Areas that were a lighter color or white in the baseline image than in the new image are colored green. Areas that were a darker color (greyscale) in the baseline image than in the new image are colored red. The areas that were not white in either image but not the same color in the new image are colored blue. The following summarizes the color scheme:

Black/Gray-scale	No Differences. This is a composite of those portions of the left and right pages that are identical.
Red	Is on the original PDF, but not on the new PDF.
Green	Is on the new PDF, but not on the original PDF.
Blue	Content in the same location on both PDFs, but there is a difference.

The following are examples of differences that may appear between PDFs:

- Color change
- Change in the type or weight of line
- Font change
- Slight pixel shift

(**Note**: The reports are generated using Antenna House XSL Formatter. If you have Formatter on your machine, then that copy will be used for the reports. If you do not already own a copy, then a restricted version of Formatter is provided with AHRTS to enable you to generate reports. What makes the report possible is Antenna House Formatter's ability to merge selected individual pages from a PDF into a single PDF file it creates. Formatter creates the individual pages of the report by using a page from each of the PDFs being tested and a bitmap image that was created during the regression testing process.)
In the example below, we can see a clear difference in the appearance of floating objects between the original and new versions of the same document.

Overlay Image

The overlay image is the same as the difference page (center pane in the individual report) but it is added on top of the original and new PDF page. It has options for transparency and removal of red/green color coding for the baseline PDF and new PDF in the individual report.

Overview Reports

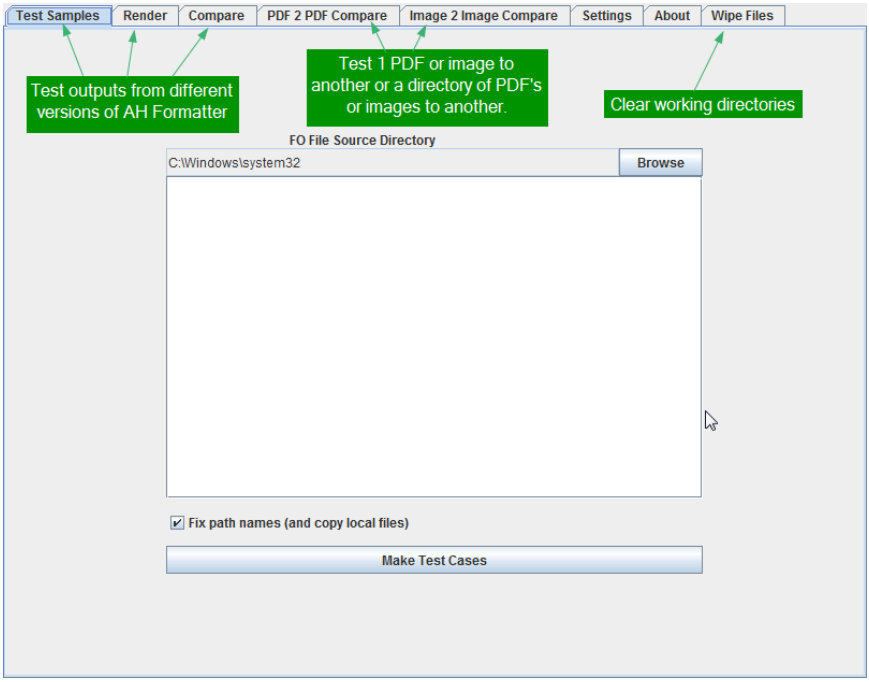
For the Overview Report, the documents containing no differences are displayed in green, while those with differences in red. The documents with differences (red) are hyperlinked to their respective individual document reports. Simply click the link to view any of the reports.

An alternative option to the standard report overview is to have a single file report listing all differences from the compared documents. The documents with differences would then be listed in the PDF bookmarks. To turn on this feature go to Settings and select 'Use a single final report listing all differences'. Command line option: single-report <true|false>.

GUI Operation

Understanding the GUI

The GUI consists of 8 tabs across the top:



1) The first three tabs (*Test Samples*, *Render* and *Compare*) are used when testing different versions of Formatter.
2) The *PDF 2 PDF Compare* tab is for comparing individual PDFs or directories of PDFs against one another.
3) The *Image 2 Image Compare* tab is for comparing individual images or directories of images against one another.
4) The *Settings* tab is for customizing the directory settings used by AHRTS.
5) The *About* tab shows the version of AHRTS and the Open Source software used in its development.
6) The *Wipe Files* tab is used to clear the working directories before starting a new test.

DPI Settings

AHRTS works by converting the PDFs into bitmap images and then doing a pixel-by-pixel comparison of the generated bitmaps. This enables you to compare anything on a page (including color) regardless of the document's content—language has no bearing on the process whatsoever.

The Dots Per Square Inch (DPI) of a comparison is the resolution at which a PDF document from a rendering engine is rasterized. This limits the precision of the comparison, which is useful if you need to process numerous or large-sized documents. (The more pixels to compare per page, the longer the comparison will take.) At 30 DPI, you have 900 dots per square inch. At 60 DPI, you have 3,600 dots per square inch. At 100 DPI, you have 10,000 dots per square inch.
In both the “Compare” and “PDF2PDF Compare” tabs, there is an option for setting the DPI. The purpose of setting the DPI is to allow the user to control the level of comparison and how sharp the composite page appears in the final report.
The table below gives a description of your ability to detect differences at various DPI settings:

DPI Setting	Difference Detection Ability
30 DPI	Provides a composite image that enables locating differences, but the type is not readable.
60 DPI	The type becomes very readable.
90 DPI	The type is displayed at close to screen resolution. Minute changes to small characters (e.g., the accent over a letter) may actually be difficult to identify.
>90 DPI	Possible to generate differences from changes as minute as a very slight shift in a character serif.

In our experience of testing and using AHRTS, we have found 30 DPI to be more than enough resolution to find differences in font types (normal vs. italic or bold, serif vs. sans-serif), minor alterations in spacing and line width, as well as larger ones such as differences in the breaking of pages.
300 DPI is enough resolution to view a document under magnification while maintaining image clarity, so it is far beyond the necessary resolution to locate even half-point changes in font size (72 points equals one inch). Because each pixel in a rasterized page is an average of the area it covers on the vector graphic, the average color value of the underlying vector parts are reflected in the resulting pixel—any ratio alterations will have an impact on the resulting pixel. Even a subpixel-sized difference can be located without identifying the exact vector position of the change.

Edge Radius

This option tries to detect if a pixel difference happens at the edges of rendering and avoids flagging those pixels as different if they happen within a specified radius. The feature also tries to detect if a difference is a result of a color change and will always flag those as different even if they occur within the specified radius. Edge detection is done in both the base and new document up to the radius specified in the ignoreEdgesRadius property. If a pixel is found to be within the radius of an edge in both documents then it is not compared. This helps avoid false positives from slight font rendering differences and having the restriction of the pixel needing to be an 'edge' in both documents allows even small differences in empty areas from being flagged.
Edge detection radius value must be greater then or equal to zero.

Highlight Radius

The differences detected between two PDFs may be as minute as a single letter change, which is difficult to find on a densely packed page. The highlight radius feature was created to solve this issue and help you find the differences visually by providing a yellow halo around them. The larger the radius, the bigger the ring. The Recommended setting

for the highlight radius is 5.

Add to Queue

This feature is found under the Render tab. The rendering of different rendering engines is variable, making it hard to predict when one will finish in order to start the next one. To save time, select a rendering engine, click Add to Queue and repeat until all the desired rendering engines have been added. Then click Render to execute.

Font & Dimensions Scaling

Windows 8 introduced a new method for applications to automatically adjust scaling for GUI(s), which has not been yet been incorporated into many programs including JAVA, which AHRTS runs on. This option allows you to manually set scaling of the AHRTS GUI on a Windows 8 machine. We recommend setting the scaling between 1 & 2. Scaling values can be set using the + and – buttons or by entering values in to the field and pressing the Enter key. Restart AHRTS for changes to take effect.

Allow Extremely High DPI Values

Selecting this option will allow you to set the DPI setting above the cap of 120. Increasing DPI increases accuracy, but lowers performance (refer to DPI Settings). However there is little reason to set the DPI above 96 DPI; the default rendering DPI for the PDF format.

Automatically Switch GUI to the Next Test Phase

Once selected, AHRTS will automatically transition to the next tab in the following sequence: Test Samples > Render > Compare . The transitions occur upon completion of each task and apply only to the tabs in the sequence.

PDF Render Backend

The default renderer used is mupdf; which is optimal for most files. In case there are issues with mupdf, there are alternatives that work with AHRTS: xpdf, pdfbox & ghostscript.

Use Windows Native EMF Support

The option is on by default. If Windows Native EMF isn’t used then Formatter will be used to handle document conversions to PNG.

Use Threaded Conversion of Base and New PDFs to Images

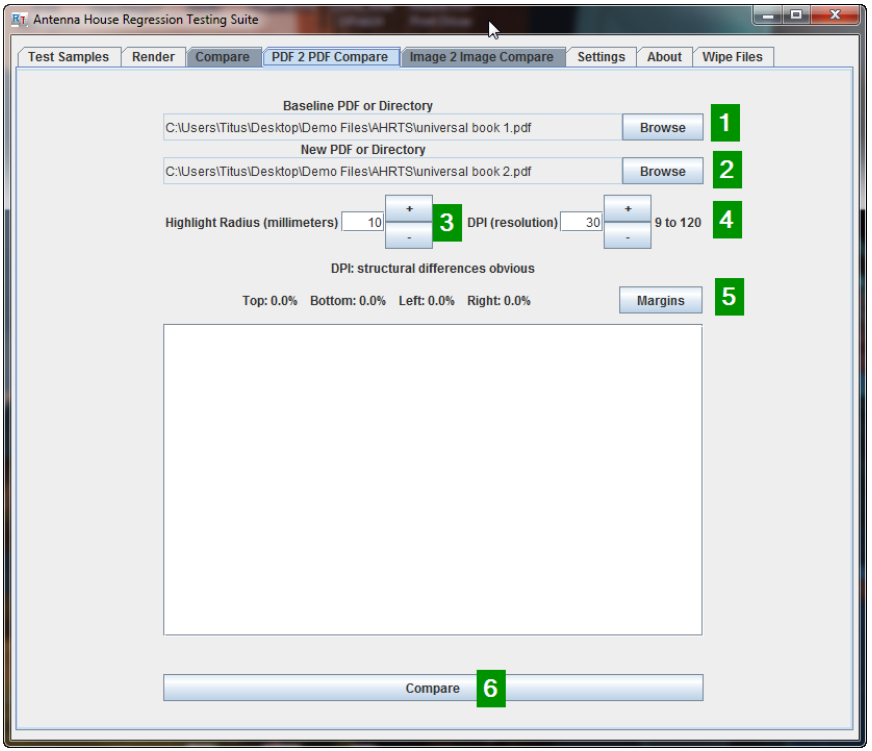
Once this option is enabled, the original and base PDF are processed simultaneously by different CPUs, thereby increasing performance on multi-CPU/Core systems. Otherwise the documents are processed sequentially. Single CPU/Core systems may experience lower performance with this option turned on.

Quick Start Guide

PDF2PDF (Testing Individual PDFs)

(The term "ahrts-gui" will be used in place of *ahrts-gui.bat* [Windows] and *ahrts-gui.sh* [Linux].)
To access *PDF2PDF Compare* open the GUI either via the shortcut in the start menu or the ahrts-gui script in the install directory. Then select the *PDP2PDF Compare* tab.

- Select the first (1) and second (2) PDFs you wish to compare.
- Select highlight radius (3).
- Select your DPI setting (4).
- Select margin exclusions (5)
- Click the *Compare* button (6), click again to cancel.

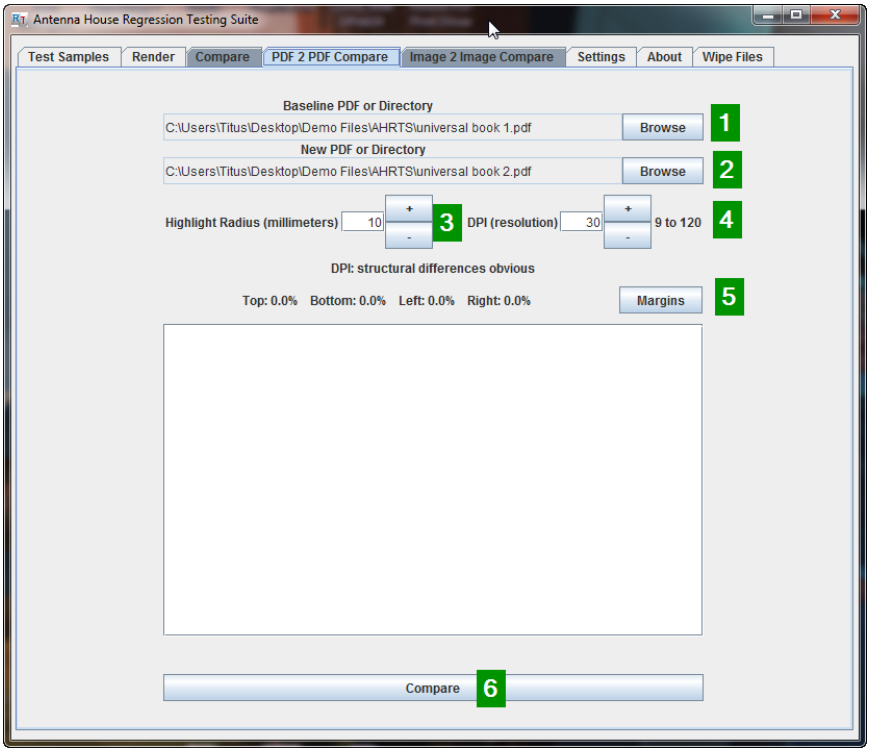


Once the process has finished, a report will open automatically.

PDF2PDF (Testing PDF Directories)

To access *PDF2PDF Compare* open the GUI either via the shortcut in the start menu or the ahrts-gui script in the install directory. Then select the *PDP2PDF Compare* tab.

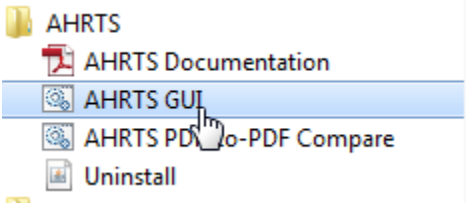
- Select the first (1) and second (2) directories you wish to compare.
- Select highlight radius (3).
- Select your DPI setting (4).
- Select margin exlusions (5).
- Click the *Compare* button (6), click again to cancel.



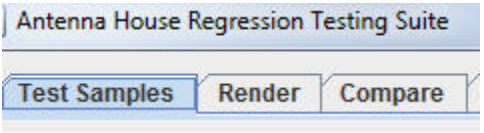
A multi-document report will open automatically once the process has finished.

Testing Multiple XSL-FO Files with Two Versions of Formatter

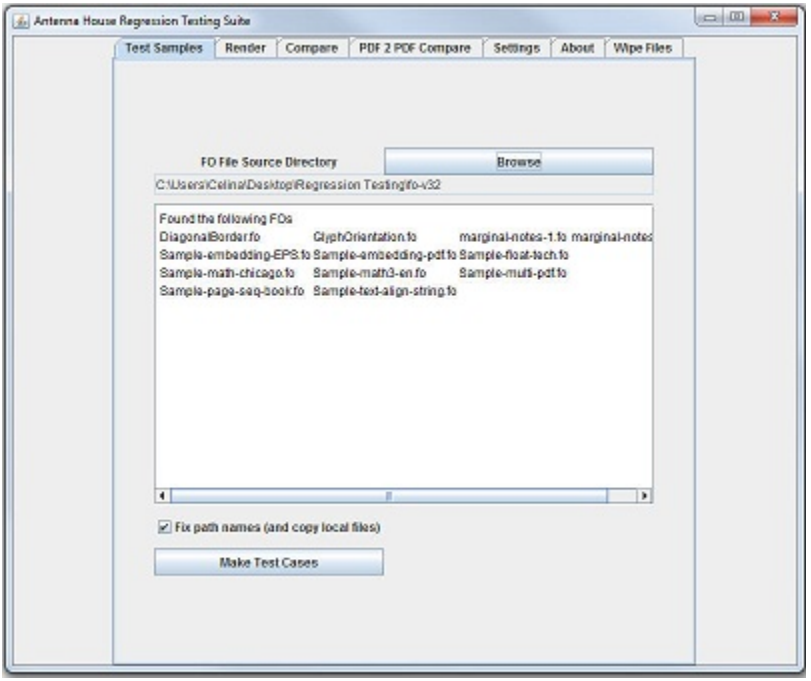
Create a Test Case
Open the GUI either via the shortcut in the start menu or the *ahrts-gui* script in the install directory.



Navigate to the *Test Samples* tab.



Click *Browse* and select a directory that contains your FO files. Uncheck *Fix Path Names* if you do not wish to include local files referenced by the FOs, and adjust the paths to those files to match the new location.
(**Note**: The FO files must not be in sub-directories if you intend to include them. This restriction is in place to help prevent accidentally loading very large sets of FOs.)



Click *Make Test Cases* to make the test cases in the *Test Cases Directory* specified on the *Settings* page. (*Demo/Test-cases* in the *Install Directory* by default)

Render Test Cases

This step may need to be repeated to produce two sets of rendered documents the first time the system is run. One set can be compared to itself, but no differences will be found.

- Select a rendering engine from the Select *Rendering Engine* drop-down list.
- Click *Add to Queue*, select next rendering engine and repeat.
- Click the *Render* button to begin, click it again to cancel.

Compare the two sets of rendered documents:

- Select a baseline rendering engine from the *Baseline Rendering Engine* drop-down list.
- Select a new rendering engine from the *New Rendering Engine* drop-down list.
- Adjust the DPI as needed.
- Click the *Compare* button to begin, click it again to cancel.

A multi-document report will open automatically once the process has finished.

Edit Engine Files

The Engine Files tell AHRTS where to find the different versions of Antenna House Formatter that will be used with the software. AHRTS comes with many premade engine files which should work for default installations of Formatter. For demonstration purposes, using a full license to render once and then an evaluation license to render a second time under a different engine version gives you a visual example of the comparison, as well as an idea of how long the process may take if each document contained discrepancies. If you already have two versions of Formatter installed in the default locations, you should not need to edit anything.
The following is an example of an engine file:

```
<?xml version="1.0"?>
<engine-config>
    <type>FO</type>
    <name>Formatter</name>
    <version>53</version>
    <config-version>1</config-version>
    <cmd-path>C:\Program Files\AntennaHouse\AHFormat-
terV53\AHFCmd.exe</cmd-path>
    <arguments>
        <arg sub="false" name="-x">4</arg>
        <arg sub="true" name="-d"><path name="input-
file"/></arg>
        <arg sub="true" name="-o"><path name="output-
file"/></arg> <!-- The 'output-file' path is generated au-
tomatically, it is not specified in the Manifest.xml -->
        <arg sub="true" name="-i" omit-if-emp-
ty="true"><path name="config-file"/></arg>
        <arg sub="true" name="-s" omit-if-emp-
ty="true"><path name="stylesheet"/></arg>
    </arguments>
    <environment>
  <var name="AHF53_64_FONT_CONFIGFILE"><path name="font-
config"/></var>
  <var name="AHF53_64_HYPDIC_PATH"><path name="hyphenation-
dir"/></var>
  <var name="AHF53_64_DEFAULT_HTML_CSS"><path name="de-
fault-html-css"/></var>
  <var name="AHF53_64_DMC_TBLPATH"><path name="dmc-tbl-
dir"/></var>
  <var name="AHF53_64_BROKENIMG"><path name="broken-
image"/></var>
</environment>
</engine-config>
```

Settings

The settings tab has options that correspond to the contents of the *ahrts.properties* file. The default settings will work out of the box for all the examples cited in this manual.

- **Test Case Directory -** The directory from which AHRTS reads the test cases.
- **Render Output Directory -** The directory where AHRTS stores the output of the rendering step and reads the same as input for the comparison step.
- **Comparison Output Directory -** The directory where AHRTS stores the XML output of the comparison step.
- **Difference Image Output Directory -** The directory where AHRTS stores the difference images created during the comparison step.
- **Report Output Directory -** The directory where AHRTS stores the reports generated during the comparison step.
- **Engine File Directory -** The directory from which AHRTS reads the rendering engine files.
- **Temporary File Directory -** The location AHRTS uses when writing temporary files to disk.
- **MuDraw Executable -** The absolute path to the *mudraw* (mudraw.exe on Windows) executable.

Command-Line Interface

Configuration

Configuring AHRTS involves a three-part process:

1) Edit/create a properties file
2) Edit/create engine files for the software used to render the PDFs
3) Edit/create manifest files for the test cases to be used

Not all of these steps are required. For example, the creation of manifests is done by the script that changes your FOs into test cases, and AHRTS comes with example engine files that may work without editing.

Properties File

The properties file is the file that holds the user-configurable settings. The default location of the properties file is labeled *ahrts.properties* in the installation directory, but it can be specified via the *-p option* in all CLI programs.

Properties File Format - The properties file is a Java properties file and shares its specific formatting rules. The following is an example of this format:

```
# comment
key=value
```

For our system there are a few refinements:

- Spaces are allowed in paths, but not quotes
- Only forward slashes are allowed in path (in Windows replace any backslashes "\" with forward slashes "/")
- No trailing slashes on directories

Property File Values - The system that reads the properties file will ignore keys it does not recognize (e.g., “Key” is not the same as “key”).

The following are valid key and value descriptions:

- **alternateIndividualReportFile** - The absolute path for alternate directory and test case comparisons. Token* replacement maybe included in path.
- **alternateOverviewReportFile** - The absolute path for alternate directory and test case comparisons. Token* replacement maybe included in path.
- **alternatePdfViewer** - The absolute path to the location of alternate PDF viewer executable.
- **axfExt** - Antenna House XSLFO Extension Namespace generally uses axf, but this depends on its definition in the FO.
- **compareOutputDir** - The absolute path to the location of the directory where the compare results will be placed after the compare step and analyzed during the report generation step.
- **configDir** - The absolute path to the location of the directory that contains the configuration files and directories (only change this default setting if the configuration files are located somewhere else).
- **convertPdfThreads** - Render base and new pdf at the same time. Accepted value is

true or false.

- **dataDir -** The absolute path to the location of the test cases.
- **diffImgDir -** The absolute path to the location of the directory where the difference images will be placed after the compare step and analyzed during the report generation step.
- **dpiUnlocked -** Property unlocks 120 dpi limitation.
- **enginesDir -** The absolute path to the location of the engine directory.
- **formatterCheckWarningPopup -** Issues pop-up warning if there is a problem with installation or if a Formatter version was not found. Accepted value is true or false.
- **generateReportForNoDifferences -** For test and directory comparisons this option will force the creation of an individual report even if no differences were found. This will also force the creation of a report if there was an error found with either input document. Rendered images are also saved and available as 'diff-image-orig-base' and 'diff-image-orig-new' in the report xml. Accepted value is true or false.
- **ghostscript -** The absolute path to the ghostscript executable.
- **guiScale -** Property to scale the gui (dimensions & fonts). Accepted values are floating point numbers.
- **highlightRadius -** Radius, in millimeters, of yellow highlight area surrounding a pixel that was found to be different.
- **ignoreEdgeRadius -** Option to detect and exclude edges. Accepted values are in pixels and greater than or equal to 0.
- **includeErrorListingInSingleFinalReport -** This option will include fatal formatting errors in single final report files. Accepted value is true or false.
- **MarginTop -**(see MarginLeft).
- **MarginBottom -**(see MarginLeft).
- **MarginRight -**(see MarginLeft).
- **MarginLeft -** values represent percentage floating point values relative to each respective area. Example: marginLeft=10.1 excludes 10.1% of the document from the left of the page area.
- **overlayHighlightAlpha -** Transparency value of margin exclusion area in overlay image. Accepted values are 0 (fully visible) - 255 (invisible).
- **pdfDraw -** The absolute path to the location of MuPDF's PDFDRAW executable.
- **preserveDocumentAspectRatioInReports -** Option to preserve document aspect ratio in the comparison report. 'true' will preserve the aspect ratio. 'false' will fit the documents into the available area. Note, the 'true' setting uses the values of reportPageWidth and reportPageHeight. These settings will need to specify their values using the same unit (ex: 'in', 'cm', 'pt', etc.).
- **renderBackend -** Method to convert PDF to images. Options are mupdf, xpdf, ghostscript and pdfbox.
- **renderThreads -** Number of threads to use for rendering test cases.
- **reportDifferenceImageOverlay -** Overlay the difference image over the compared files in the individual reports. Accepted value is true or false.
- **reportDir -** The absolute path to the location of the directory where the reports will be placed after the report generation step.
- **reportFormatterThreads -** Number of threads to use for individual report generation (Pdf2Pdf & Compare). Values above the number of CPUs available will degrade performance.
- **reportOverlayPdfLayers -** Property to control use of PDF layers for the difference image overlay. Accepted value is true or false.
- **reportPageHeight -** Report page height. This should use the same units (in, cm, pt,

etc.) as reportPageWidth.
- **reportPageWidth -** Report page width. This should use the same units (in, cm, pt, etc.) as reportPageHeight.
- **reportPdfLayerContentPane -** Displays the layer content pane when report opens. Accepted value is true or false.
- **testOutputDir -** The absolute path to the location of the directory where the test results will be placed after the render step and analyzed during the compare step.
- **tintMargins -** Controls coloring of excluded margin area in the difference image. Accepted value is true or false.
- **tmpDir -** The absolute path to the location of the directory you wish the rendered pages to be written.
- **useAlternateIndividualReportFile -** Option to use an alternate file report name and location for single pdf2pdf/img2img compare. Accepted value is true or false.
- **useAlternateOverviewReportFile -** Option to use an alternate file report name and location for directory and test case comparisons. Accepted value is true or false.
- **useAlternatePdfViewer -** Option to specify a different PDF reader.
- **useSingleReportFile -** Option for a single final report listing all the differences from the compared documents. Documents that contained differences are listed in the pdf bookmarks. Accepted values are true or false. Command line option: single-report.
- **xpdfPdftoppm -** The absolute path to the xpdf executable.

* *Refer to section: Alternate Report File Name & Location (p. 16)*

Common Comand-Line Options

The common options for use in command-line include:
-*h* Print command line help
-*V* Print version information

Command-Line Interface Return Values

Exit Codes:
1 - No differences were found.
0 - Document differences were found.
6 - Fatal Error
The exit codes are produced for single file PDF2PDF compare, directory PDF 2 PDF compare and Image 2 Image compare.

Alternate Report File Name & Location

An alternate final report name and location maybe chosen using the properties; 'useAlternateIndividualReportFile' and 'alternateIndividualReportFile.
- **useAlternateIndividualReportFile -** Option to use an alternate file report name and location for single pdf2pdf/img2img compare. Accepted value is true or false.
- **alternateIndividualReportFile -** The absolute path for alternate directory and test case comparisons. Token replacement maybe included in path.
- **useAlternateOverviewReportFile-** Option to use an alternate file report name and location for directory and test case comparisons. Accepted value is true or false.
- **alternateOverviewReportFile -** The absolute path for alternate directory and test case comparisons. Token replacement maybe included in path.

Available tokens for alternateIndividualReportFile and alternateOverviewReportFile are:

%d date (year-month-day ex: 2014-03-25)
%c clock time (hour-minutes-seconds ex: 15-54-13)
%b 'base' file name minus extension and parent path, for Compare this is the base engine name plus version (ex: Formatter-61-32bit)
%n 'new' file name minus extension and parent path, for Compare this is the new engine name plus version
%t report type ('pdf2pdf', 'img2img', 'pdf2pdf-directories', 'img2img-directories', 'compare')
%u incrementing number that avoids files that already exist
Examples:

```
alternateIndividualReportFile=C:\data\my-report-%d.pdf

-->  report saved to 'C:\data\my-report-2014-03-25.pdf'

alternateIndividualReportFile=C:\data\%t-report-%u.pdf

-->  report saved to 'C:\data\pdf2pdf-report-13.pdf'
                    'C:\data\pdf2pdf-report-14.pdf'
etc...
```

Command line scripts have an option to automatically set both of them:
'-report-file'
Examples:

```
       ./ahrts-pdf2pdf.sh -b ~/original.pdf -n ~/edit.pdf -
report-file ~/tmp/%b-%n.pdf
```

Writes report to /home/user/tmp/original-edit.pdf.

```
       ./ahrts-img2img.sh -b ~/img1/ -n ~/img2/ -report-
file /data/reports/img-dir-compare/overview.pdf
```

Writes the overview report to /data/reports/img-dir-compare/overview.pdf and the img-dir-compare directory will contain the individual reports.
Note: distributed compare ignores these properties and always uses $REPORT_DIR/distributedCompare as the destination directory for the reports.

Engine File Format

Engine files are used by the system to recognize and label the particular rendering software. The following is a common format to all engine file types:

```
<?xml version="1.0"?>
<engine-config>
<type>{input type}</type>
<name>
<!-- for now the file name must match the name specified
here -->
{Software Name}
```

```
</name>
<version>
<!-- for now the file name must match the version speci-
fied here -->
{Software Version}
</version>
<config-version>
{Unsigned  integer matching a config version in the test
case definition}
</config-version>
<!-- engine specific configurations  here -->
</engine-config>
```

The root element is *engine-config* and contains the following child elements:

- **type -** The input type (FO/XSL/CSS/etc.)
- **name -** The name of the software as it will appear in the output file names.
- **version -** The version of the software as it will appear in the output file names.
- **config-version -** The version of the configuration specified in the test-cases.

Engine Command-Line Notes

When an engine is invoked, the command-line from the engine file will run in the system console. This console does not inherit any environmental values. When necessary, those must be set by using a wrapper script or the environmental variable configuration options. Below is the file format for the command-line engine type:

```
<?xml version="1.0"?>
<engine-config>
<type>{input type}</type>
<name>{Software Name}</name>
<version>{Software Version}</version>
<config-version>{Unsigned integer}</config-version>
<cmd-path>{ Absolute path to the software, for example
C:\Program Files\Antenna
House\AHFormatterV65\AHFCmd.exe }</cmd-path>
<arguments>
<arg sub="false" name="-x">
<!-- This says to use this value ("4") directly for option
"-x" -->4
</arg>
<arg sub="true" name="-d">
<!-- This says to substitute the value found when evaluat-
ing the child "path" element as the value of option "-d" --
>
<path name="input-file"/>
<!-- the path element references a correspondingly named
element in the test case manifest file -->
</arg>
<arg sub="true" name="-o"><path name="output-file"/></arg>
<arg sub="true" name="-i" omit-if-empty="true">
<!-- This says to substitute the value of the child "path"
element but it that is empty, then it should not be passed
on the command line -->
<path name="config-file"/>
</arg>
</arguments>
<environment>
  <var name="AHF65_64_FONT_CONFIGFILE"><path name="font-
config"/></var>
  <var name="AHF65_64_HYPDIC_PATH"><path name="hyphenation-
dir"/></var>
  <var name="AHF65_64_DEFAULT_HTML_CSS"><path name="de-
fault-html-css"/></var>
  <var name="AHF65_64_DMC_TBLPATH"><path name="dmc-tbl-
dir"/></var>
  <var name="AHF65_64_BROKENIMG"><path name="broken-
image"/></var>
```

```
  <!-- These will create an environment variable of the
var@name with the value of the evaluation of the child
path element, or the string value just as is done above
with 'arg'. -->
</environment>
</engine-config>
```

The root element is *engine-config*, and the engine specific child elements are as follows:

- **cmd-path** - The absolute path to the script or executable for the program to be run.
- **arguments** - The element that contains the arguments to be passed to the executable specified in cmd-path as child arg elements.
- **arguments/arg** - The individual argument with attributes determining the way it will be passed, and a value that will be processed and/or passed for the option. It has the following attributes:
  - *arguments/arg/@name*: (string) The command-line option as it will be passed to the executable (e.g., -x to set the exit level for Formatter).
  - *arguments/arg/@sub*: (boolean) If true, the system will try to substitute the value of this element for the item it represents (default is true).
  - *arguments/arg/@omit-if-empty*: (boolean) If the value or the item substituted for it is empty, then the system will not pass it on the command line (default is false).
  - *arguments/arg.value*: (string or XML) Either the string representation of the value to be passed to the argument, or a XML element that can be used to reference elements in the test case manifest files.
  - *arguments/arg/path*: An element whose name attribute is a pointer to its counterpart in the test case manifest files. The value of the name is an arbitrary string that has an identical counterpart in one or more test case manifest files.
  - *environment*: The element that contains the environmental variables to be set before running the executable.
  - *environment/var*: The individual element that contains the name of the environmental variable (name) and the value it should be set to.

Manifest File

The manifest file is in the top level directory of the individual test case and contains the test case instructions for AHRTS.
The bare minimum contents of a manifest file are as follows:

```
<test-manifest>
<test-name>test</test-name>
<path name="input-file">test.fo</path>
<conversion type="FO" />
</test-manifest>
```

An example of a manifest for use with Formatter with different configuration requirements between the versions to be compared:

```
<test-manifest>
  <test-name>test</test-name>
  <path name="input-file">test.fo</path>
  <conversion type="FO" />
  <arg name="-pjq">90</arg>
  <engine-configuration name="Formatter" config-ver-
sion="0">
    <path name="config-file" >ahfsettings.xml</path>
    <path name="font-config">font-config-0.xml</path>
  </engine-configuration>

  <engine-configuration name="Formatter" config-ver-
sion="1">
    <path name="config-file" >ahfsettings.xml</path>
    <path name="font-config">font-config-1.xml</path>
  </engine-configuration>
</test-manifest>
```

Test Manifest Element Descriptions

- **test-name -** The name of the test (also the directory inside the test cases directory that contains it).
- **conversion -** Information about what is being converted, and to what it is being converted.
- **conversion/@type -** The type of file(s) being converted.
- **engine-configuration/@name -** The name of the engine targeted by these settings.
- **engine-configuration/@config-version -** The version of these settings (This refers not to the version of the engine, rather the version of the configuration. See configuration version [p. 19] in the glossary).
- **path.value -** The value used to replace the path element matching the *@name* of this path element in any engine file.
- **path/@name -** The name used to reference the correspondingly named path element in any engine file.

More specifically:

- **test-manifest/path/@name="input-file" -** The value of this path element will be used as the input file by any engine (*-d equivalent* for Formatter).
- **engine-configuration/path/@name="config-file" -** The value of this path element will be used as the configuration file by any engine (*-i equivalent* for Formatter).
- **engine-configuration/path/@name="font-config" -** The value of this path element will be used as the font configuration file by any engine.
- **arg/@name -** The name of an additional command line option to use.
- **arg.value -** The value of the arg/@name command line option.

Test Case (Directory Structure)

Directory is the name of the test case, which contains a manifest file with the name *manifest.xml* (case sensitive). The manifest file can point to anything as a test file, but we recommend that all files directly related to a particular test case be placed in the same directory with the manifest file.

Creating Test Cases

(The term "ahrts-make-test-cases" will be used in this documentation in place of *ahrts-make-test-cases.bat* [Windows] and *ahrts-make-test-cases.sh* [Linux].)
Requirements:

- A FO file in a directory or directory full of FO files
- A (preferably empty) directory in which to place the test cases
- A properties file with the *dataDir* value set (optional)

To Run the Script:

```
ahrts-make-test-cases -s <your directory of FOs>  -t <the
directory where your test-cases will be placed>
```

All options: [] are not required

- (-s|-source) <directory containing FOs>

- [(-t|-target) <target directory>]
- [(-d|-deps)|(-f|-fixpaths)]
- [(-p|-properties) <ahrts properties file>]
- [(-h|-help|--help)]
  - Print command line help.
- [-V]
  - Print version information.

Check the test cases to be sure they are set to your desired specifications:

- If you have a properties file already setup, you can omit *-t* and use *-p* to specify the properties file.
- If you want locally stored images to be copied to the test case directories, use *-d* to have this script grab anything it can locate.
- If you want the paths to be adjusted to fit the new location of the FO files, use *–f* (implicitly sets *-d*).

Render Test Cases

(The term "ahrts-render" is used in place of *ahrts-render.bat* [Windows] and *ahrts-render.sh* [Linux].)
To render one test case:

```
ahrts-render -t <test name> -e <engine name> -v <engine version>
```

All options: [] are not required

- (-t <test name>|-all)
- (-e|-engine) <engine name>
- (-v|-version) <engine version>
- [(-p|-properties) <ahrts properties file>]
- [-V]
  - Print version information.
- [(-h|-help|--help)]
  - Print command line help.

Note: The -e and -v options correspond to <name> and <version> in the engine configuration files.

Compare Test Case(s)

To compare one test case:

```
ahrts-compare -t <test name> -be <baseline  engine name> -
bv <baseline engine version> -ne <other/new engine name> -
nv <other/new engine version>
```

To compare all test cases:

```
ahrts-compare  -all -be <baseline  engine name> -bv <base-
line engine version> -ne <other/new engine name> -nv <oth-
er/new engine version>
```

All options: [] are not required

- (-t <test name>|-all)
- (-be|-engine-base) <baseline engine name>
- (-bv|-version-base) <baseline engine version>
- (-ne|-engine-new) <new engine name>
- (-nv|-version-new) <new engine version>
- [(-r|-dpi) <integer 9-120>]
- [(-mt|-margin-top) <float>]
- [(-mb|-margin-bottom) <float>]
- [(-ml|-margin-left) <float>]
- [(-mr|-margin-right) <float>]
- [(-p|-properties) <ahrts properties file>]
- [-report-file <tokenized output report filename>]
- [-single-report <true|false>]
  - Use a single final report listing all the differences.
- [(-ot|-overview-title) <overview report title>]
- [-xsltparam <name> <value>]
  - Pass variables to xslt stylesheets.
- [(-h|-help|--help)]
  - Print command line help.
- [-V]
  - Print version information.

Visually Comparing Test Cases

To render all test cases:

```
ahrts-render -all -e <engine name> -v <engine version>
```

Visual Comparison across Distributed Systems

The distributed comparison is comprised of two components: a server and a client. The server will collect all the tasks in a target directory and then listen for connections from clients that are ready to process those tasks. The client component requests tasks, one at a time, from the server. When it is done processing a task, it requests a new one. When the server reports that there are no more tasks, the client ceases to request new tasks.
Only one server should be run, but there may be multiple clients. If a machine has several processors or cores, it can run multiple clients.
To run server:

```
prompt> ahrts-compare-service.bat <public IP> <engine-A>
<engine-B> [test-name ...]
```

The *engine-A* and *engine-B* arguments are the version strings that identify which engine was used to generate the PDFs you wish to compare.
You can specify an arbitrary number of tests (by name) to be compared. If you do not specify any test names, then all the tests found in the result directory will be tested.
To run client:

```
prompt> ahrts-compare-client.bat <public IP> <server IP>
```

The server should be started first, then the clients can be started and will begin processing the task issued by the server. Once all the clients have completed their task(s) and exited, the server can be stopped with CTRL-C.

Quick Document Compare (PDF to PDF Comparison)

Use the following:

```
ahrts-pdf2pdf -b <baseline file or folder> -n <new version
file or folder>
```

All options: [] are not required
usage: ./ahrts-pdf2pdf.sh (-b|-baseline) <baseline file or folder>

- (-b|-baseline) <baseline file or folder>
- (-n|-new) <new file or folder>
- [(-r|-dpi) <integer 9-120>]
- [(-mt|-margin-top) <float>]
- [(-mb|-margin-bottom) <float>]
- [(-ml|-margin-left) <float>]
- [(-mr|-margin-right) <float>]
- [-o <diff image directory>]
- ((-c <path to pdfdraw>) | ((-p|-properties) <properties file>))

- [-report-file <tokenized output report filename>]
- [-single-report <true|false>]
  - Use a single final report listing all the differences with directory comparisons.
- [(-ot|-overview-title) <overview report title>]
- [-xsltparam <name> <value>]
  - Pass variables to xslt stylesheets.
- [-disable-temporary-output]
  - Stores all the files generated in the comparison into the directories defined in the installation's ahrts.properties file.
- [(-h|-help|--help)]
  - Print command line help.
- [-V]
  - Print version information.

Image to Image Comparison

Use the following:

```
ahrts-img2img -b <baseline file or folder> -n <new version
file or folder>
```

All options: [] are not required
usage: ./ahrts-img2img.sh (-b|-baseline) <baseline file or folder>

- (-b|-baseline) <baseline file or folder>
- (-n|-new) <new file or folder>
- [(-mt|-margin-top) <float>]
- [(-mb|-margin-bottom) <float>]
- [(-ml|-margin-left) <float>]
- [(-mr|-margin-right) <float>]
- [-o <diff image directory>]
- [-report-file <tokenized output report filename>]
- [-single-report <true|false>]
  - Use a single final report listing all the differences with directory comparisons.
- [-xsltparam <name> <value>]
  - Pass variables to xslt stylesheets.
- [-disable-temporary-output]
  - Stores all the files generated in the comparison into the directories defined in the installation's ahrts.properties file.
- [(-T|-image-types)]
- [(-h|-help|--help)]
  - Print command line help.
- [-V]
  - Print version information.

Manual Test Report Generation

The Antenna House Regression Testing System generates a visual comparison output as a XML file. To make these files easier to read, AHRTS is configured by default to use An-

tenna House Formatter to generate a PDF report from the compare result XML.
The overview report lists each of the input documents, and whether or not visual differences were located among the results of the chosen rendering engines.
When discrepancies have been identified, an individual report is generated for each document containing a difference. The individual reports list the pages containing the problems, as well as a triple-wide page displaying each individual difference. For each page containing a discrepancy, three documents will be displayed: a page from the base engine (left side, red border), one from the new engine (right side, green border) and a bit composite image displaying the differences between the two (center, blue border).
Antenna House Formatter is the only FO rendering engine that has the necessary functionality to generate the various reports.
The XSL-FO files are located in the reports output directory configured through the AHRTS.properties file.
The style sheets required to generate the FO files are included in the report directory of the AHRTS install directory.
(**Note**: These style sheets rely on Antenna House Formatter proprietary extensions for styling and linking functions.)

Notes on Usage

DPI Settings

In our experience of testing and using AHRTS, we have found 30 DPI to be more than enough resolution to find differences in font types (normal vs. italic or bold, serif vs. sans-serif), minor alterations in spacing and line widths, as well as larger ones such as differences in the breaking of pages.
300 DPI is enough resolution to view a document under magnification while maintaining image clarity, so it is far beyond the necessary resolution to locate even half-point changes in font size (72 points to one inch). Because each pixel in a rasterized page is an average of the area it covers on the vector graphic, the average color value of the underlying vector parts are reflected in the resulting pixel—any alteration of average color value will have an impact on the resulting pixel. Even a subpixel-sized difference can be located without identifying the exact vector position of the change.

Suggested Precautions

1) Wipe files before starting a new regression test to keep the size of the working directories manageable and to remove test cases that you do not want included in the new reports.
2) Close PDF files before starting the compare process. AHRTS needs to be able to read and write PDFs to process them and generate reports. If a PDF is open in Acrobat, then AHRTS will not be able to gain access to that PDF or write over an old version when generating a new report. (**Note**: If the report PDF is left open, Formatter will not be able to write a new one. A FO file will still be created that can be manually processed through Formatter once the previous PDF is closed. This saves the need to rerun the entire regression test over because a PDF file was not closed.)
3) Use short directory paths. When you browse to a document or directory you have to navigate via the folder structure. Navigating through multiple folders can be very cumbersome.
4) Periodically clear the temporary file directory. AHRTS places the bitmaps in this directory during the testing step.

Temporary File Directory

The temporary file directory can be set to a ram disk to improve performance, or a spinning platter hard drive to protect a solid state drive.

XML catalog file stylesheet support

Catalog settings uses Java system properties (xml.catalog.files, xml.catalog.prefer, xml.catalog.verbosity, xml.catalog.staticCatalog, xml.catalog.allowPI, xml.catalog.className, xml.catalog.ignoreMissing). See xml-commons-resolver-1.2/docs/resolver.html for more information. If xml.catalog.files is not set catalog support is disabled.

Language

Language files are automatically loaded based on user's Locale settings. This can also be set directly in the Java command line. Ex: java -Duser.country=CA -Duser.language=fr.

Multiple Instances

It is safe to run multiple instances of the pdf2pdf and img2img command line programs but the user or calling application must make sure the '-report-file' option is used to specify a unique name for each instance.

Open Source Components

The following Open Source components were used to build the Antenna House Regression Testing System:

Akka

http://akka.io/
The Akka library is included as a Java Library (JAR) and used by AHRTS for distributed threading.
The Akka library is licensed under the Apache 2 license (quoted below). Copyright 2009-2011 Typesafe Inc.
<http://www.typesafe.com>
Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0>.
Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing the permissions and limitations under the License.

Apache XML Commons Resolver

https://xerces.apache.org/xml-commons/
Apache XML Commons Resolver
Copyright 2006 The Apache Software Foundation.
Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0
Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Apache Commons IO

https://commons.apache.org/proper/commons-io/
Apache Commons IO is Copyright 2002-2017 The Apache Software Foundation
Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0
Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Apache Commons Imaging

https://commons.apache.org/proper/commons-imaging/

Ghostscript Fonts

http://sourceforge.net/projects/gs-fonts

Google Diff Match Patch

https://github.com/google/diff-match-patch

ICU4J

http://site.icu-project.org/

to whom the Data Files or Software are furnished to do so, provided that either (a) this copyright and permission notice appear with all copies of the Data Files or Software, or (b) this copyright and permission notice appear in associated Documentation.
THE DATA FILES AND SOFTWARE ARE PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THE DATA FILES OR SOFTWARE.
Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in these Data Files or Software without prior written authorization of the copyright holder.

Legion of the Bouncy Castle Java Cryptography APIs

https://www.bouncycastle.org/java.html
Copyright (c) 2000 - 2018 The Legion of the Bouncy Castle Inc. (https://www.bouncycastle.org)
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MuPDF

http://mupdf.com/
MuPDF's mudraw.exe is included when AHRTS is installed on Windows, Linux, Solaris and Macintosh systems. It can be utilized by AHRTS to convert PDF pages to rasterized images.
MuPDF is Copyright 2006-2013 Artifex Software, Inc.
This program is free software: you can redistribute it and/or modify it under the terms of the GNU Affero General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.
This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR

A PARTICULAR PURPOSE. See the GNU Affero Public License for more details.
You should have received a copy of the GNU Affero Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

PDFBox

http://pdfbox.apache.org/
PDFBox is included when AHRTS is installed on Windows, Linux, Solaris and Macintosh systems. It can be utilized by AHRTS to convert PDF pages to rasterized images.
PDFBox is Copyright 2014 The Apache Software Foundation.
This program is free software: you can redistribute it and/or modify it under the terms of the Apache License as published by The Apache Foundation, version 2 of the license.
This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the Apache License for more details.
You should have received a copy of The Apache Software Foundation Apache License, Version 2.0 along with this program. If not, see <http://www.apache.org/licenses/LICENSE-2.0>.

Saxon

http://saxon.sourceforge.net/
Saxon-HE is included as a Java Library (JAR) for the purpose of executing XSLT 2.0 transformations.
Saxon-HE is Copyright (C) Saxonica Limited and distributed under the Mozilla Public License 1.0. The full text of the license can be accessed at <http://www.mozilla.org/MPL/1.0/>.

Scala

http://www.scala-lang.org
Copyright (c) 2002-2018 EPFL
Copyright (c) 2011-2018 Lightbend, Inc.
All rights reserved.
Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the EPFL nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSE-

QUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

SLF4J

https://www.slf4j.org/
SLF4J source code and binaries are distributed under the MIT license.
Copyright (c) 2004-2017 QOS.ch
All rights reserved.
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Xpdf

http://www.foolabs.com/xpdf/home.html
Xpdf is included when AHRTS is installed on Windows, Linux, Solaris and Macintosh systems. It can be utilized by AHRTS to convert PDF pages to rasterized images. Xpdf binaries are built from the source code of each platform.
Xpdf is Copyright 1996-2011 Glyph & Cog, LLC.
This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 2 or 3 of the License (at your option).
This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Glossary

Baseline
: The version of the rendering engine or document(s) produced by it that are used as the reference for comparison.

Configuration Version
: The version of the configuration that goes with one or more versions of the rendering engine. This is not necessarily the same as the version of the rendering engine.

DPI
: The Dots Per Inch (DPI) of a comparison is the resolution at which the output document from a rendering engine is rasterized. Adjusting the DPI settings may limit the precision of your comparison.

New
: The version of the rendering engine or document(s) produced by it that is used as the object to be measured in a comparison.

Engine Types
: Currently there is only one engine type (command-line).

Rendering Engine
: Software used to process the test cases into output documents for comparison.

Index