STAXDoc User's Guide

STAX Documentation Generator (STAXDoc) User's Guide
Version 1.0.4

February 26, 2008

Contents

Introduction

STAXDoc is used to generate documentation for your STAX xml files. As you grow your library of STAX functions, you will probably find it useful to document the STAX functions to make it easier to reuse them and share them with other test groups.

STAXDoc is a Java application that parses the documentation elements in a set of STAX xml files and generates an HTML document describing all of the functions defined in the STAX xml files. STAXDoc uses an XSLT stylesheet processor to transform function information provided in STAX xml files into HTML files which are nicely formatted.

You can run STAXDoc on a set of directories that contains STAX xml files. Each sub-directory is considered a source "package" and can be passed to the STAXDoc command line.

> **Note:** When you pass in package names to STAXDoc, all `.xml` files in the specified package directories are processed.

STAXDoc produces one complete document each time it is run; it cannot do incremental builds -- that is, it cannot modify or *directly* incorporate results from previous runs of STAXDoc.

Requirements

1. Java Runtime Environment (JRE) 1.4 or later

2. `STAXDoc.jar` provided with the STAX service.

Note: You can obtain the STAXDoc.jar file by downloading the STAX V3 tar/zip file obtained from the Download STAX webpage and extracting the tar/zip file.

Syntax

```
java -jar STAXDoc.jar [-options] packagename(s)...
```

`options`

The command line options that can be specified. The options include:

```
-d <directory>          Specifies the destination directory for output
                        files.  The current directory is the default.
-doctitle <html-code>   Specifies to include the title for the package
                        index (first) page.
-functionsummary <FirstSentence | All>
                        Specifies what to include for the description on
                        each "Function Summary" page:
                        - FirstSentence: Include only the first sentence
                          of the function-prolog.  This is the default.
                        - All: Include the entire contents of the
                          function-prolog.
-help                   Specifies to display the help.
-overview <file>        Specifies to read overview documentation from the
                        HTML file.
-sourcepath <directory> Specifies the root directory of the packages.
                        The current directory is the default.
-verbose                Specifies to output messages about what STAXDoc is
                        doing.
-windowtitle <title>    Specifies the title to be placed in the HTML
                        <title> tag.
```

See the Options section for a more detailed description of the available options.

`packagename(s)...`

The names of one or more subdirectories in the `-sourcepath` containing STAX xml files that you want to document. The subdirectory names must be separated by one or more spaces. You must separately specify each package (subdirectory) that you want to document as subdirectories are not recursively traversed.

Package names can be overridden using the `=` keyword. For example, if you specify `src1=P1 src2` in the command line, the first package will appear named `P1` in the generated documentation.

Examples

```
java -jar STAXDoc.jar -d C:\stax\mydocs -sourcepath C:\stax\xml src1
src2
```

```
java -jar STAXDoc.jar -sourcepath C:\stax\xml -verbose libraries

java -jar STAXDoc.jar -d C:\staxdocs -functionsummary All -sourcepath
C:\stax xml

java -jar STAXDoc.jar -sourcepath C:\user\src utils/memory
```

See the Examples section for more examples of using STAXDoc.

Source Files

STAXDoc will generate output originating from four different types of "source" files:

1. STAX xml files (`.xml`)
2. Package comment files (`package.html`)
3. Overview comment files (typically `overview.html`)
4. Miscellaneous unprocessed files (optional)

STAX Xml Files

Each STAX xml file contains at least one function. Each function can documented using the standard documentation elements defined for a STAX xml file. These include:

- `function-prolog` (or the deprecated `function-description` element)
- `function-epilog`
- Description text for function arguments: `function-required-arg`, `function-optional-arg`, `function-other-arg`, and `function-arg-def`

For more details about these STAX documentation elements, see the STAX User's Guide.

Package Comment Files

Each package can have its own documentation comment, contained in its own HTML file, that STAXDoc will merge into the package summary page that it generates. You typically include any documentation that applies to the entire package in this HTML file.

To create a package comment file, you must name it **`package.html`** and place it in the package directory in the source tree along with the `.xml` files. STAXDoc will automatically look for this filename in this location. Notice that the filename is identical for all packages.

The content of the package comment file is one big documentation comment, written in HTML, like all other comments. When writing the comment, you should make the first sentence a summary about the package, and not put a title or any other text between `<body>` and the first sentence.

When STAXDoc runs, it will automatically look for this file; if found, STAXDoc inserts all content between `<body>` and `</body>` tags of the file at the bottom of the package summary page it generates.

Overview Comment File

Each application or set of packages that you are documenting can have its own overview documentation comment,

kept in its own HTML file, that STAXDoc will merge into the overview page that it generates. You typically include any documentation that applies to the entire application or set of packages in this HTML file.

To create an overview comment file, you can name the file anything you want, typically **`overview.html`** and place it anywhere, typically at the top level of the source tree.

The content of the overview comment file is one big documentation comment, written in HTML, like the package comment file described previously. See that description for details.

When you run STAXDoc, you specify the overview comment file name with the -overview option. The file is then processed similar to that of a package comment file.

Miscellaneous Unprocessed Files

You can also include in your source any miscellaneous files that you want STAXDoc to copy to the destination directory. These typically include graphic files, example STAX xml files, and self-standing HTML files.

To include unprocessed files, put them in a directory called **`doc-files`** which can be a subdirectory of any package directory. You can have one such subdirectory for each package. You might include images, example code, source files, applets and HTML files.

Typically these unprocessed files are referenced from STAX documentation tags or package and overview comment files. For example, a `function-prolog` tag in a STAX xml file may look like:

```
<function-prolog>
  <![CDATA[
  This is a custom image <img src="doc-files/MyImage.gif">
  ]]>
</function-prolog>
```

Options

Available command-line options are:

-d
-doctitle
-functionsummary
-help
-overview
-sourcepath
-verbose
-windowtitle

Options description

-d *directory*

Specifies the **d**estination directory where STAXDoc saves the generated HTML files. Omitting this option causes the files to be saved to the current directory. The value *directory* can be absolute or relative to the

current working directory.

-functionsummary `FirstSentence | All`
Specifies what to include on each "Function Summary" page:
- `FirstSentence`: Include only the first sentence of the function-prolog (or function-description). This is the default.
- `All`: Include the entire contents of the function-prolog (or function-description).

-doctitle *title*
Specifies the title to be placed at the top of the overview summary file. The title will be placed as a centered, level-one heading directly beneath the upper navigation bar. The title may contain html tags and white spaces, though if it does, it must be enclosed in quotes.

-help
Displays the online help, which lists the STAXDoc command line options.

-overview *path\filename*
Specifies that STAXDoc should retrieve the text for the overview documentation from the "source" file specified by *path\filename* and place it on the Overview page. The *path\filename* is relative to the `-sourcepath`.

For information about the file specified by *path\filename*, see overview comment file.

The title on the overview page is set by `-doctitle`.

-sourcepath *directory*
Specifies the root directory of the source tree for the package(s) you are documenting. If `-sourcepath` is not specified, STAXDoc looks in the current directory for the source files.

For example, suppose you want to document a package called `utils/memory` whose source files are located at:

```
C:\user\staxsrc\utils\memory\*.xml
```

In this case you would specify the `sourcepath` to `C:\user\src`, the directory that contains `utils/memory`, and then supply the package name `utils/memory`:

```
java -jar STAXDoc.jar -sourcepath C:\user\src utils/memory
```

-verbose
Provides more detailed messages while STAXDoc is running.

-windowtitle *title*
Specifies the title to be placed in the HTML <title> tag. This appears in the window title and in any browser bookmarks (favorite places) that someone creates for this page.

Examples

You can run STAXDoc on entire packages of STAX xml files. Each package is simply a subdirectory of your root

directory containing a set of STAX `.xml` files. In the following examples, the STAX xml files are located at `C:\user\src\utils\*.xml`. The destination directory is `C:\user\mystaxdoc`.

Documenting One or More Packages

You can run STAXDoc either of the following two ways -- by changing directories (with `cd`) or by using `-sourcepath` option. You cannot use wildcards to specify groups of packages.

- **Case 1 - Changing to the package directory** - Change to the parent directory of the fully-qualified package. Then run STAXDoc, supplying names of one or more packages you want to document:

  ```
  cd C:\user\src\
  java -jar STAXDoc.jar -d C:\user\mystaxdoc utils utils/memory
  ```

- **Case 2 - From any directory** - In this case, it doesn't matter what the current directory is. Run STAXDoc supplying `-sourcepath` with the parent directory of the fully-qualified package, and supply names of one or more packages you want to document:

  ```
  java -jar STAXDoc.jar -d C:\user\mystaxdoc -sourcepath C:\user\src utils
  utils/memory
  ```

Overiding Package Names

You can override package names using the = keyword. In the following example the package utils/memory is renamed as 'mem':

```
java -jar STAXDoc.jar -d C:\user\mystaxdoc -sourcepath C:\user\src utils
utils/memory=mem
```

Using the Verbose Option

Here's an example of the output you can get when using the `-verbose` option when documenting the samples and libraries packages in the C:\STAF\services\stax directory:

```
C:\STAF\services\stax>java -jar STAXDoc.jar -verbose -d C:\STAXDoc\output samples
libraries
Option set: verbose=true
Option set: d=C:\STAXDoc\output
STAX source package:samples
STAX source package:libraries

Package:samples
  File:samples\sample1.xml
  File:samples\sample2.xml
Package:libraries
  File:libraries\STAXUtil.xml

Generating index.html
Generating overview-summary.html
Generating overview-frame.html
Generating allfiles-frame.html
Generating package-overview.html - libraries
```

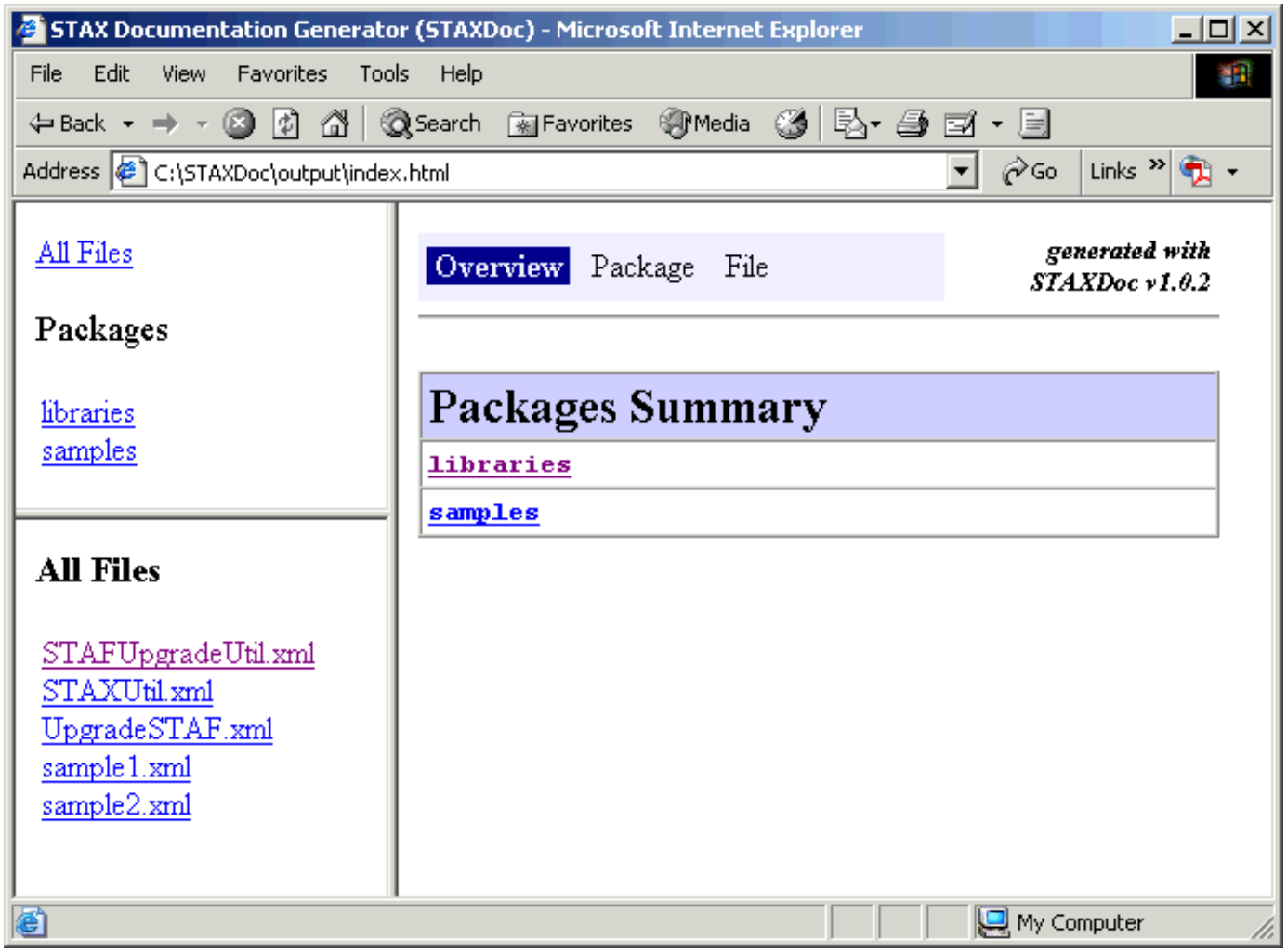
```
Generating package-frame.html
Generating package-overview.html - samples
Generating package-frame.html
Generating .\libraries\STAXUtil.html
Transforming .\libraries\STAXUtil.xml
Generating .\samples\sample1.html
Transforming .\samples\sample1.xml
Generating .\samples\sample2.html
Transforming .\samples\sample2.xml
STAXDoc ended with success
```

Generated Documentation

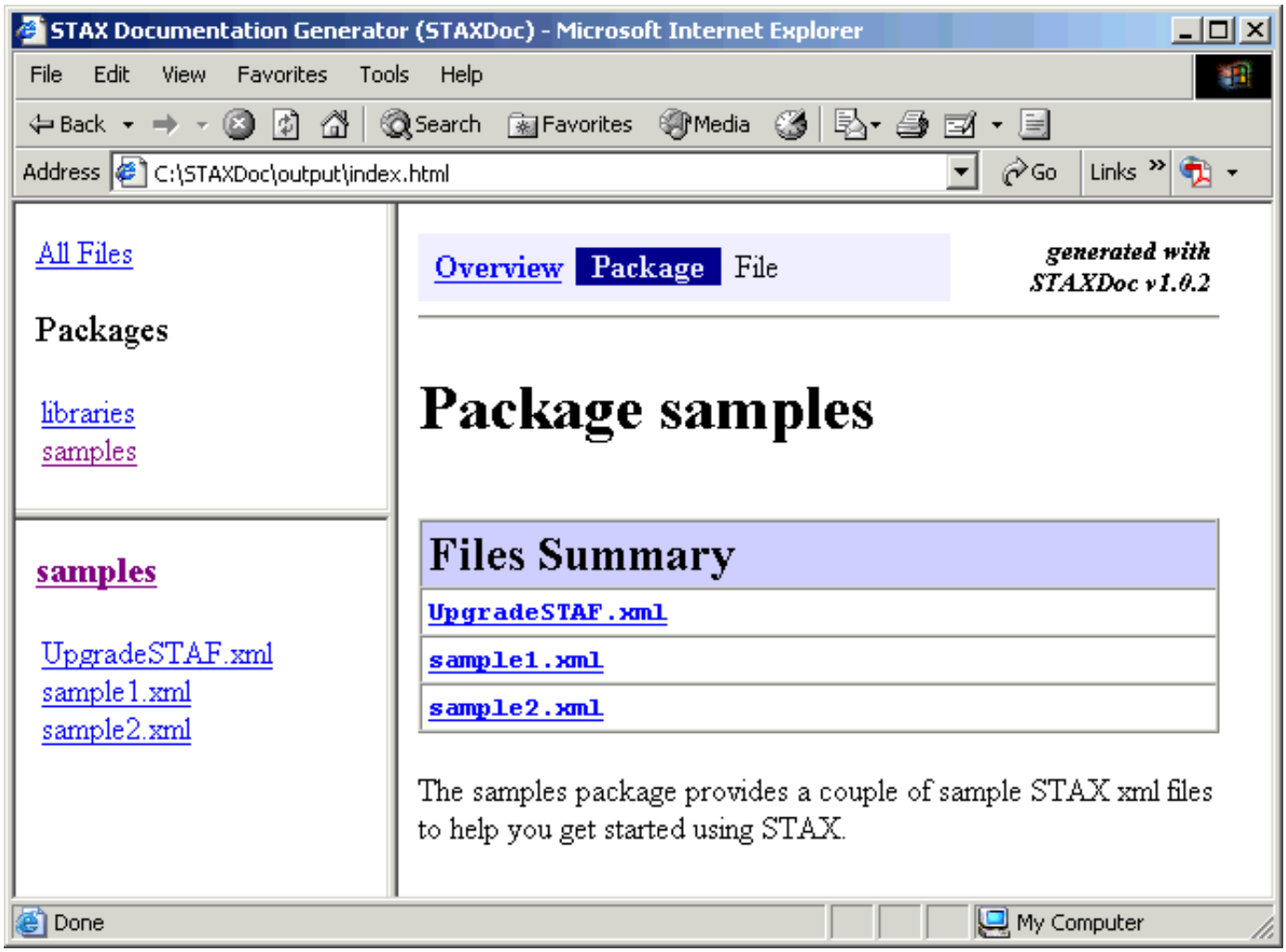
Suppose you generated HTML documentation for the `.xml` files in the "`samples`" and "`libraries`" directories in source path `C:\STAF\services\stax` as follows:

```
cd C:\STAF\services\stax
java -jar STAXDoc.jar -d c:\STAXDoc\output samples libraries
```

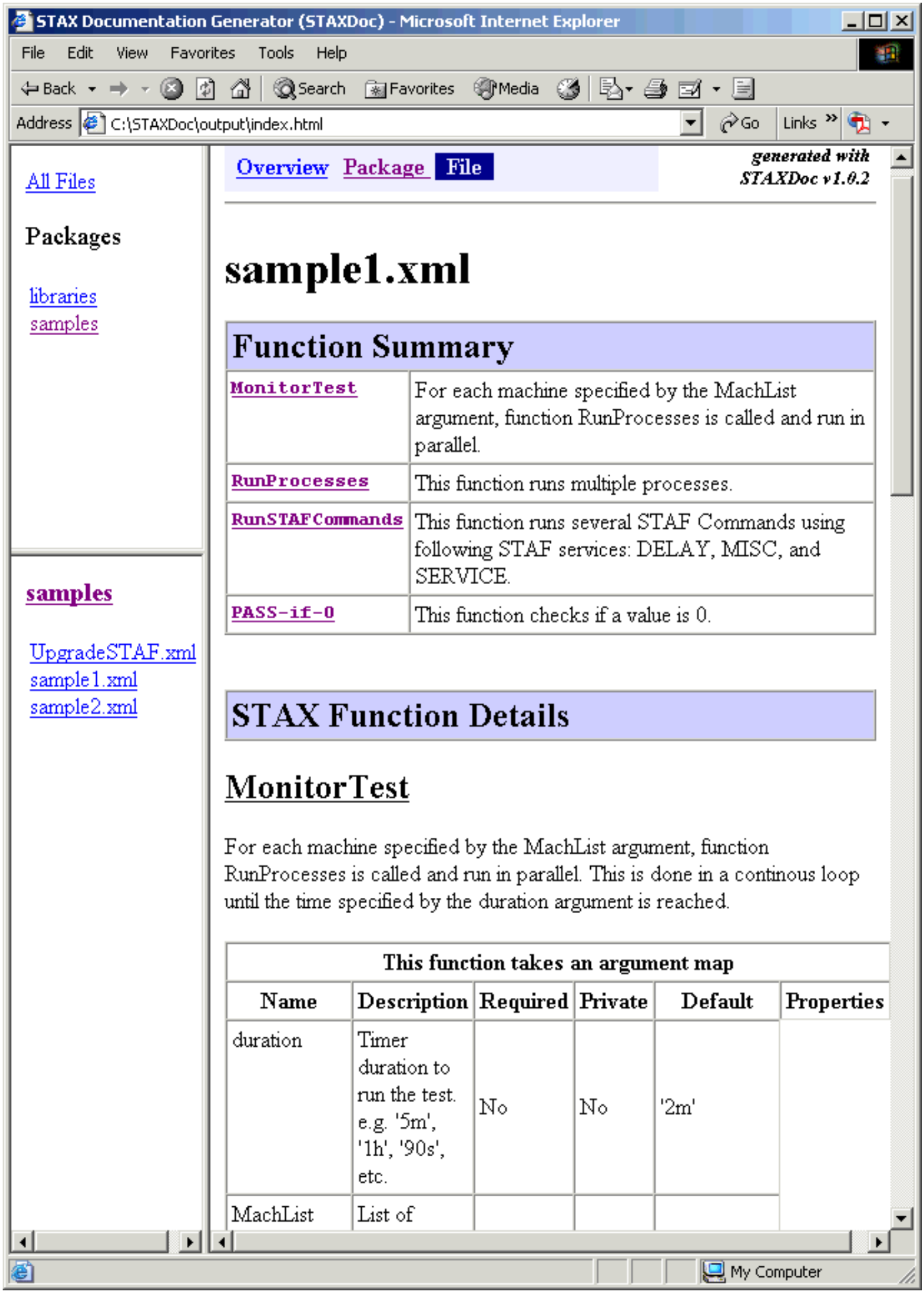
Here's a view of the HTML documentation generated by STAXDoc for the overall documentation obtained by specifying the `index.html` file in the destination directory:

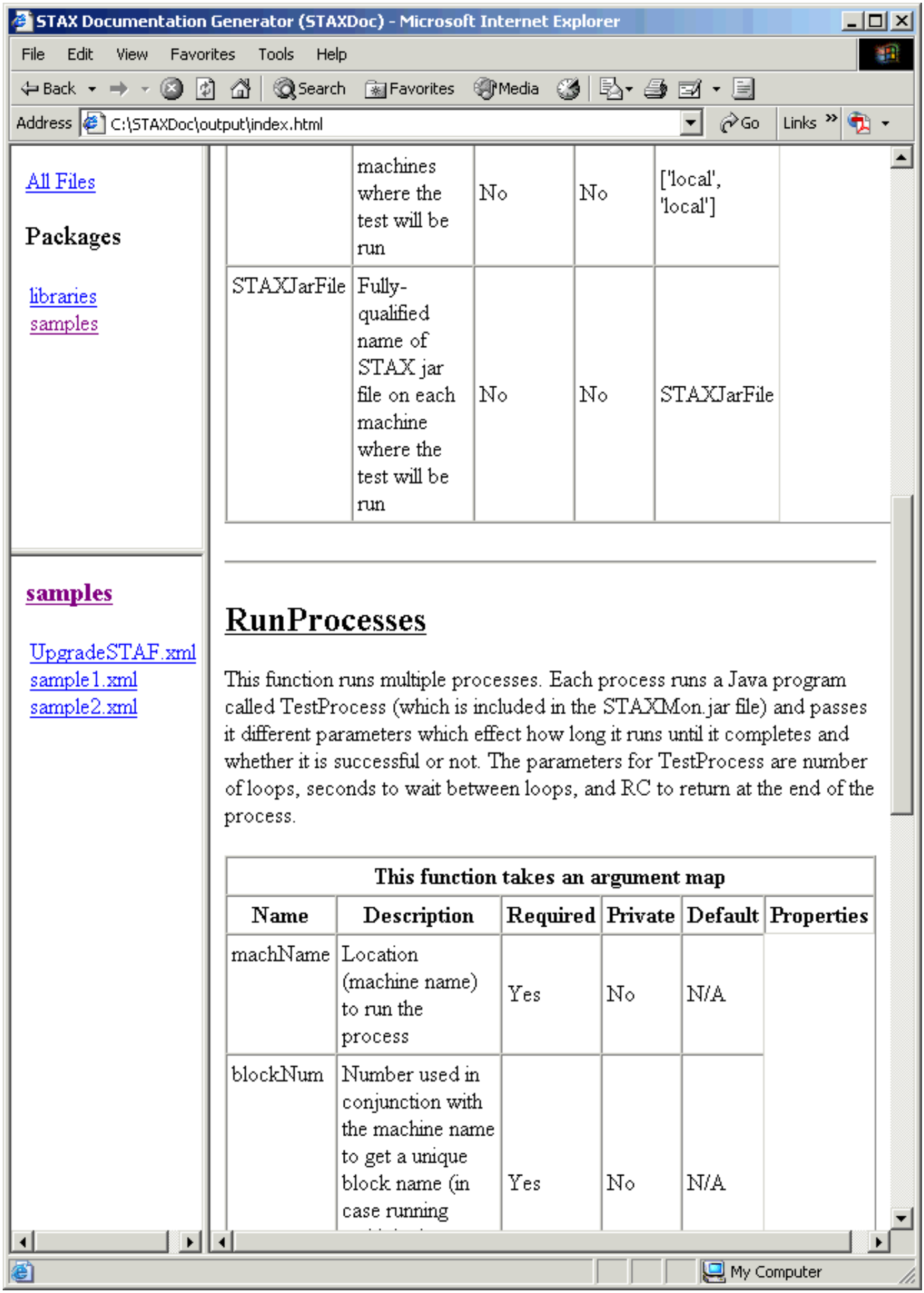


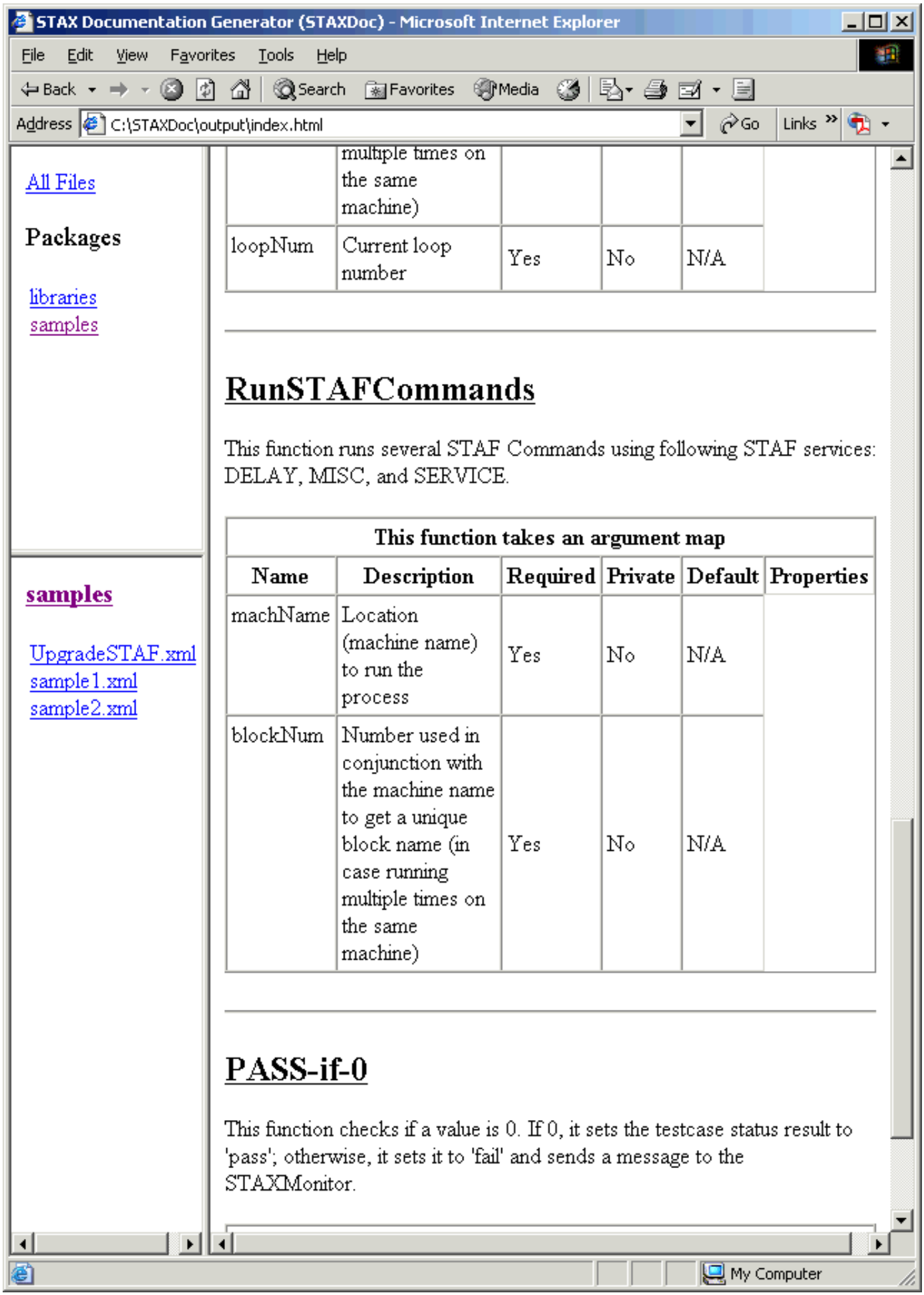
Here's a view of the HTML documentation generated by STAXDoc for the `samples` package obtained by clicking on samples in the upper left panel under "**Packages**" and then clicking on **samples** in the lower-left panel:



Here's a view of the HTML documentation generated by STAXDoc for file `sample1.xml` obtained by clicking on **sample1.xml**. Note that a summary of all of the functions defined in the xml file are shown first, followed by a detailed description of each function.


STAX Documentation Generator (STAXDoc) - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address C:\STAXDoc\output\index.html
Go Links
All Files
Packages
libraries
samples
samples
UpgradeSTAF.xml
sample1.xml
sample2.xml
Overview Package File
generated with STAXDoc v1.0.2
sample1.xml
Function Summary
MonitorTest
For each machine specified by the MachList argument, function RunProcesses is called and run in parallel.
RunProcesses
This function runs multiple processes.
RunSTAFCommands
This function runs several STAF Commands using following STAF services: DELAY, MISC, and SERVICE.
PASS-if-0
This function checks if a value is 0.
STAX Function Details
MonitorTest
For each machine specified by the MachList argument, function RunProcesses is called and run in parallel. This is done in a continous loop until the time specified by the duration argument is reached.
This function takes an argument map
Name
Description
Required
Private
Default
Properties
duration
Timer duration to run the test. e.g. '5m', '1h', '90s', etc.
No
No
'2m'
MachList
List of
My Computer


STAX Documentation Generator (STAXDoc) - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address C:\STAXDoc\output\index.html
Go Links
All Files
Packages
libraries
samples
samples
UpgradeSTAF.xml
sample1.xml
sample2.xml
machines where the test will be run
No
No
['local', 'local']
STAXJarFile
Fully-qualified name of STAX jar file on each machine where the test will be run
No
No
STAXJarFile
RunProcesses
This function runs multiple processes. Each process runs a Java program called TestProcess (which is included in the STAXMon.jar file) and passes it different parameters which effect how long it runs until it completes and whether it is successful or not. The parameters for TestProcess are number of loops, seconds to wait between loops, and RC to return at the end of the process.
This function takes an argument map
Name
Description
Required
Private
Default
Properties
machName
Location (machine name) to run the process
Yes
No
N/A
blockNum
Number used in conjunction with the machine name to get a unique block name (in case running
My Computer



	multiple times on the same machine)				
loopNum	Current loop number	Yes	No	N/A	

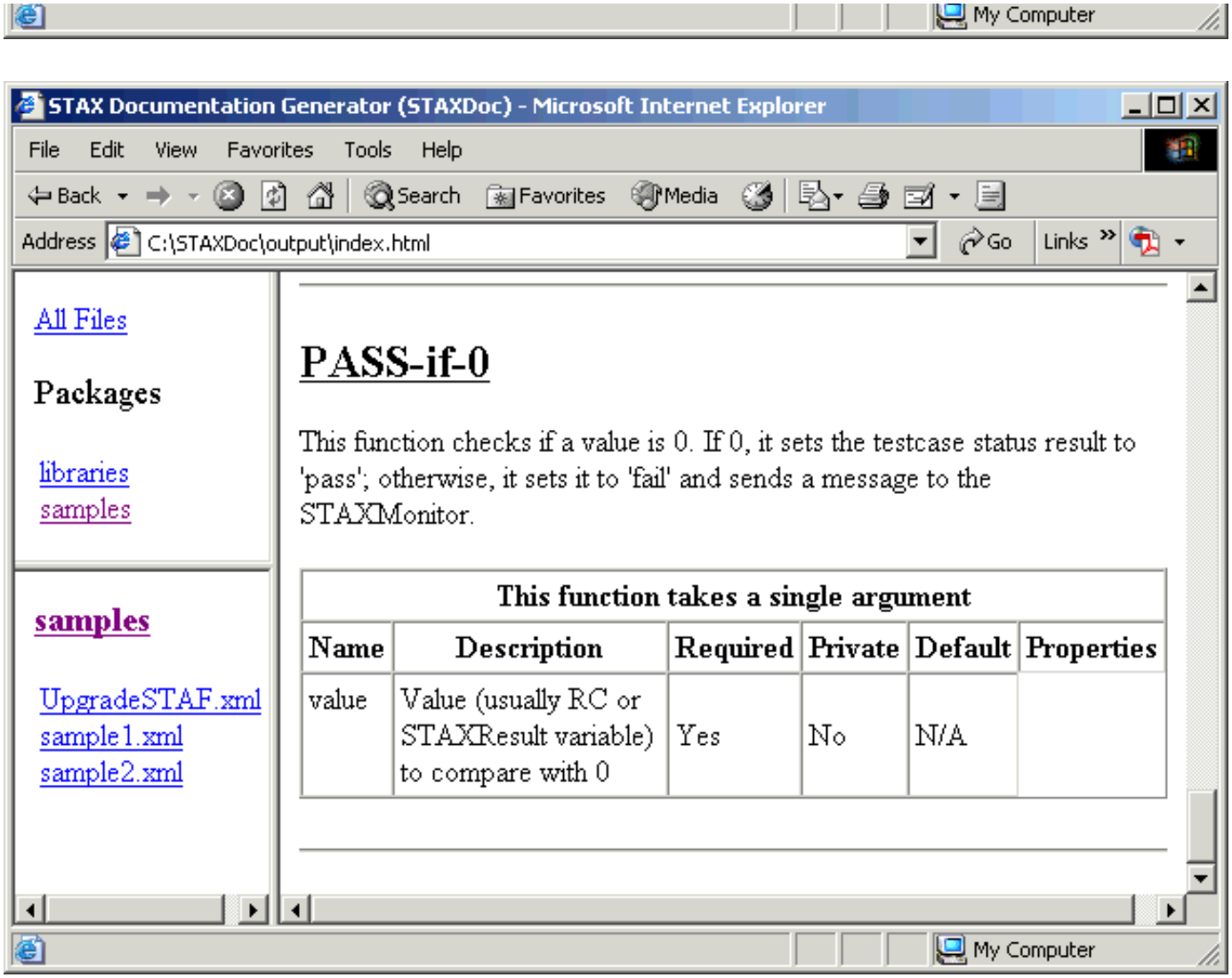
RunSTAFCommands

This function runs several STAF Commands using following STAF services: DELAY, MISC, and SERVICE.

This function takes an argument map					
Name	**Description**	**Required**	**Private**	**Default**	**Properties**
machName	Location (machine name) to run the process	Yes	No	N/A	
blockNum	Number used in conjunction with the machine name to get a unique block name (in case running multiple times on the same machine)	Yes	No	N/A	

PASS-if-0

This function checks if a value is 0. If 0, it sets the testcase status result to 'pass'; otherwise, it sets it to 'fail' and sends a message to the STAXMonitor.



References

For additional information on STAF and STAX, see:

- STAF User's Guide
- STAX User's Guide (PDF manual)