The Forrest Primer

Don't panic!

Forrest is a so-called fledgling project that will have a broad impact on xml.apache.org projects. This document helps you to better understand the vision and scope of Forrest, so that you learn what to expect (or not) from it, and eventually will help you discovering places where your contribution could be valuable to all of us.

Table of contents

Warning:

This document is *very* out of date. There is a lot of good information here, but the focus of the project has shifted away from the Sourceforge-like project management system described here, towards being a simpler project-centric documentation tool -- JT

1. History

Forrest has come into existence because of the abysmal state of the xml.apache.org website in comparison with other open source community sites such as Sourceforge. The old site had no consistent visual look and feel, which was largely due to each and every sub-project managing its own site. Furthermore, much information which could potentially support community-based open source development was hidden inside CVS repositories, mailing lists or word of mouth. Once we experienced the usefullness of cross-project collaboration supported by the Jakarta Gump project, we reckoned having a single application responsible for the management of the xml.apache.org site could be of benefit to our visitors. And if we added aggregated access to other available resources such as download stats or mailing list archives, the new xml.apache.org website could be a true information clearinghouse for interested parties, both users and contributors alike.

The Forrest vision was articulated by Stefano Mazzocchi and Sam Ruby, both long-time contributors to Apache projects, in the beginning of 2002, and was rapidly picked up by a bunch of other contributors as well, after a headstart by Nicola Ken Barozzi. So here we are, plenty of work-in-progress to erect what eventually will become a true community website infrastructure for Apache open source development.

2. What is Forrest

Forrest is a framework that supports the cross-project generation and management of development project websites using Cocoon as its XML publishing framework. It not only provides access to project documentation, but also to other types of information that open source developers depend upon daily: source code repositories, mailing lists, contact info and the like. It aggregates all these resources and publishes them on a regular basis to a website, ensuring a consistent look and feel using skins implemented with XSLT stylesheets. While Forrest's primary focus is XML Apache project websites, it can be adapted to other community development projects as well, as long as they are willing to commit to proven best practices such as Ant for build automation, CVS for source code control and XML as a documentation source format.

Forrest is currently based on an Ant-based project build system called Centipede that drives a Cocoon-based document publication system. It contains a set of standard XML document type declarations (DTDs) for project documentation, and different 'skins' consisting of XSLT stylesheets that produce HTML renditions of XML documents using these DTDs.

The primary mode of operations for Forrest will be as follows:

Note:

This process is not quite ready for prime time yet, but it gives you an idea where we are heading to. Website generation with skins currently works, try using the `docs` target when invoking the `build` script. Add a `project.skin` property when invoking the build script to experience Forrest skins: `build{.bat|.sh} -Dproject.skin=<thenameoftheskintouse> docs`. Read our CVS crash course to get hold of the current codebase and start playing with it.

1. Forrest will harvest documentation and related source files from each of the projects within the community that uses Forrest for their website, usually direct from the CVS repository. Which

projects are included, and how they are retrieved is configured by a project descriptor file. This is an automated process that occurs several times a day to ensure Forrest has the latest information available.
2. Forrest then uses Cocoon to generate an HTML rendition of each project's website, configured by a generic sitemap. The result is a static collection of HTML documents and related images and stylesheets comprising the project's website. The impact Forrest has on the participating projects should be minimal, i.e. one should simply author XML documents, put them in a well-specified filesystem hierarchy, and Forrest will do its work.
3. Forrest will enrich the documentation source files with common information: a cross-project navigation structure (and rendition, of course), and useful 'community indicators' such as download statistics, number of contributors with commit access, ...
4. If the individual project build runs are successful, the project's website is automagically (re-)published to the (Apache) website, also several times day.

The Forrest website and the overall xml.apache.org website are maintained and published using the same mechanism.

3. Forrest roles

Depending on your interests, your involvement with Forrest may vary, hence your *role*. We currently envision three different roles:

- **User** you want or need to use Forrest for your project because it uses Forrest to manage its documentation.
- **Adaptor** you want to adapt Forrest to support your individual project needs, presumably outside the XML Apache context, building your own skins or DTDs and the like.
- **Contributor** you are a fledgling Forresteer and want to contribute to the further development of it. If your contributions are valuable and in true community spirit, you can possibly gain commit access to the Forrest CVS repository and become an Apache committer. The first stage towards becoming a contributor is to join the forrest dev mailing list, the second is to download Forrest and start playing with it (see below).

Depending on your role, your potential area of interest in Forrest will vary:

Role	Interests
User	Forrest DTDs and documentation filesystem hierarchy (Cocoon sitemap)
Adaptor	+ skin system and build environment
Contributor	+ the Forrest codebase and runtime environment

4. Getting your local copy of Forrest through CVS

4.1. System requirements

Forrest requires the following systems to be already installed on your system:

- *Java Virtual Machine* A Java virtual machine must be present. Forrest has been tested against the latest Sun 1.3 JDK.

4.2. Getting Forrest

You can retrieve Forrest from its CVS repository or download here.
Some help with CVS follows (courtesy of our friends of the Cocoon project).

4.3. Step-by-step cvs instructions for Windows

1. Download a recent release of WinCVS (homepage is http://www.wincvs.org/);
2. Install it;
3. Start it;
4. Click on Admin->Preferences;
5. In "Enter the CVSROOT:" enter "`:pserver:anoncvs@cvs.apache.org:/home/cvspublic`" (without quotes);
6. In "Authentication:" choose "passwd file on the cvs server";
7. Click "Ok";
8. Click Admin->Login;
9. When asked for the password: answer "`anoncvs`" (without quotes);
10. Click "Create->Checkout module";
11. Module name and path on the server is "`xml-forrest`" (no quotes);
12. Choose a dir to put the source code in;
13. Click "Ok";
14. If everything goes well, messages will start to appear in the log window;
15. Wait until you see "`*****CVS exited normally with code 0*****`" in the log window;
16. The Forrest source is now on your harddrive.

4.4. Step-by-step cvs instructions for Unix

1. Make sure you have a CVS client package installed on your Unix system.
2. Start the shell of your choice.
3. Enter "`cvs -d :pserver:anoncvs@cvs.apache.org:/home/cvspublic login`".
4. When asked for the password: answer "`anoncvs`".
5. Enter "`cvs -d :pserver:anoncvs@cvs.apache.org:/home/cvspublic -z3 checkout xml-forrest`". This will create a directory called "`xml-forrest`" where the Forrest source will be stored.
6. Wait until cvs has finished.
7. The Forrest source is now on your harddrive.

In case you want to update your Forrest source tree to the current version, change to the "`xml-forrest`" directory and invoke "`cvs -z3 update -d -P`".

5. Forrest distribution

Once you retrieved Forrest from its CVS repository, you will end up with a filesystem hierarchy similar to this inside the `xml-forrest` home directory:

Warning:
This is highly volatile information!

```
+---legal                               various licenses for included projects
```

```
+---lib                               jar library
+---src
|   +---documentation                 Forrest's documentation (not generally
reusable)
|   |   +---content                   content of the Forrest website
|   |   |   +---xdocs                 Forrest website XML documents
|   |   +---resources                 Forrest-specific doc resources
|   |   |   +---images
|   +---resources                     Generic resources for any Forrest-using
project.
|   |   +---conf                      Default (overridable) Forrest config files
|   |   +---library                   common components (not skin-specific)
|   |   |   +---xslt                  document format transformers e.g. faq->xdoc
|   |   +---convert                   XSLTs for aiding a transition to Forrest
|   |   +---skins                     Forrest skins
|   |       +---basic
|   |       +---forrest-site          the future xml.apache.org skin
|   |       |   +---css               Cascading Stylesheets
|   |       |   +---images            skin-specific images
|   |       |   +---xslt              the skin stylesheets (per medium)
|   |       |       +---fo
|   |       |       +---html          html rendering skins
|   |       +---jakarta-site
|   |       +---scarab-site
|   |       +---xml-apache-site
|   |   +---schema                    Generic Forrest DTDs
|   |       +---dtd
|   |       +---relaxng
|   |       +---entity
|   |   +---images                    Reusable skin-agnostic images
|   |   +---fresh-site                A template project structure
|   |   +---forrest-shbat             'shbat' Forrest distribution files
|   |   +---forrestbot                Ant-based Forrest deployment tool
|   |   +---forrestbar                Mozilla Forrest toolbar
|   |   +---charts                    charting trials
|   |   +---layout                    HTML page mock-ups
|   |   |   +---resources
|   |   |   +---xml.apache.org
|   |   |       +---images
|
+---tools                             Tools used to build Forrest
    +---ant                           Ant 1.6-dev scripts and jars
+---stylesheets                       Stylesheets used for project root XML files
```

The `xml-forrest` home directory consists of the main Ant build script (`build.xml`) and platform-specific batch files/shell scripts to invoke it. Forrest comes with Ant included, so you do not need to install Ant separately.

Running Forrest is a batch operation you can start using the provided `build.{sh|bat} <targetname>`. The current main targets are:

- **`docs`** - generates an HTML rendition of the Forrest website using the default `forrest-site` skin
- **`clean`** - cleans out the `build` directory
- **`webapp`** - for those who cannot resist running Forrest live instead of its commandline invocation, this target builds a WAR file you can deploy in your servlet container (currently only tested for Tomcat 4.0.1). Mount-point of the web application will be `xml-forrest`.

After a build run, Forrest creates a `build` directory. You can find the generated website in the `build/xml-forrest/docs/` directory. Forrest also creates a `tools/tmp/anttasks/` upon its first invocation. These are Centipede-specific compiled Ant tasks.

6. The Forrest DTDs

Forrest is the reference repository for the XML Apache documentation DTDs. Special care is taken to provide a set of modular, extensible and well-maintained DTDs for project documentation purposes. This modularity is ensured using the OASIS catalog mechanism, extensive use of external parameter entities and an entity resolver capable of resolving entities through the aforementioned catalog mechanism. For the docheads amongst us, this means we adhere to the strict use of `PUBLIC` entity identifiers both in document instances and DTD modules.

We have currently identified the following document types:

- General documents (`document-v11.dtd`),
- How-Tos (`howto-v10.dtd`),
- Collections of FAQs (`faq-v11.dtd`).

Some work is also under its way for other document types, in close collaboration with the Cocoon project. You will also find some older document types such as `changes`, `javadoc`, `specification` and `todo`, which are currently under consideration for automatic generation and maintenance using Gump or Centipede descriptors and the like. DTDs will be subject of serious version management as soon as Forrest has a 1.0 release: they are made to depend upon.

The DTDs are located in `src/resources/schema/dtd` and also refer to some character entity collections stored in the `src/resources/schema/entity` directory. These are referred to by the declarations found in the `src/resources/schema/catalog` OASIS Catalog file. Take special care using the correct `PUBLIC` identifiers in the DTD declaration of your instances:

```
<?xml version="1.0"?>
<!DOCTYPE document PUBLIC "-//APACHE//DTD Documentation V1.2//EN"
"http://apache.org/forrest/dtd/document-v12.dtd">
<document>
  ...
```

The exact local location of the DTD for validation purposes is obtained by the entity resolver evaluating the mapping scheme as defined in the `catalog` file. This makes sure that you can move and re-arrange your document instances apart from your DTD files. Later on, the DTDs will be web-accessible from the Forrest website for your perusal.

7. Forrest site generation using Cocoon

The `docs` target of the Forrest build environment invokes Cocoon as a command-line application to generate an HTML rendition of the project's documentation. It is not within the scope of this document to explain the Cocoon internals, please read its own documentation to fully understand the power of Cocoon.

Cocoon's site rendition behaviour is configured in a so-called *sitemap*, a switchboard that binds URLs to an XML processing pipeline. This pipeline typically consists of a Generator, one or more Transformers and a Serializer. Forrest also makes use of Cocoon's aggregation capabilities that merge multiple pipelines into one resulting output document.

A typical page generated using Forrest looks like this:

This page is currently composed of two XML sources which are transformed by a different XSLT stylesheet, aggregated by Cocoon with a post-aggregation stylesheet adding the overall page grid and look & feel. This simple example is handled by the following *sitemap* snippets (`src/documentation/conf/sitemap.xmap`):

```
<map:match pattern="*.html">
  <map:aggregate element="site">
    <map:part src="cocoon:/book-{1}.xml"/>
    <map:part src="cocoon:/body-{1}.xml" label="content"/>
  </map:aggregate>
  <map:call resource="skinit">
    <map:parameter name="type" value="site2xhtml"/>
  </map:call>
</map:match>
<map:match pattern="**book-**.xml">
  <map:generate src="content/xdocs/{1}book.xml"/>
  <map:call resource="skinit">
    <map:parameter name="type" value="book2menu"/>
  </map:call>
</map:match>
<map:match pattern="body-**.xml">
  <map:generate src="content/xdocs/{1}.xml"/>
  <map:call resource="skinit">
    <map:parameter name="type" value="document2html"/>
  </map:call>
</map:match>
<map:resource name="skinit">
  <map:transform src="skins/@skin@/xslt/html/{type}.xsl">
    <map:parameter name="isfaq" value="{isfaq}"/>
  </map:transform>
  <map:serialize/>
</map:resource>
```

When an URL (e.g. `http://forrest.apache.org/index.html`) is passed through the Cocoon system to generate the required page, the pipeline flow is basically as follows:

1. The URL is matched by the `*.html` pattern

2. Cocoon responds by aggregating two 'sub-requests'. The first is for the resource `book-{1}.xml`, the second is for the resource `body-{1}.xml`. The `{1}` parameter is replaced by the values of the first wildcard in the matching pattern above. These 'sub-requests' are passed through the Cocoon pipeline just like any other request. This results in the following flow:
   1. The first 'sub-request' (for `book-index.xml` is matched by the `**book-**.xml` pattern. This results in the file `content/xdocs/book.xml` being read. This document is then run through the `book2menu` stylesheet (which produces an HTML fragment comprising the site navigation, the red area in the image above.
   2. The second 'sub-request' is matched by the `body-**.xml` pattern. This results in the file `index.xml` being transformed using the `document2html` stylesheet, the yellow area in the screenshot.
3. The aggregation result is then transformed using the `site2xhtml` stylesheet which adds the cherries to the cake. The grey zone.

These *skin-specific* stylesheets are located in `src/documentation/skins/<nameoftheskin>/xslt/html`, so if you want to add your own skin, this is the place to be. Apart from these, there exist a number of other stylesheets located in `src/documentation/library/xslt` and more importantly:

- `faq2document`: transforms documents following the `faq-v11` DTD to `document-v11` grammar
- `howto2document`: transforms documents following the `howto-v10` DTD to `document-v11` grammar
- and some others.

As you see, all documents, regardless of their original DTD, are transformed to the `document` DTD prior to rendition. This alleviates the burden of adding new skins to implementing 3 simple stylesheets: `book2menu`, `document2html` and `site2xhtml`.

8. Where we are heading to

We have been explaining so far where we are now and what already works. The purpose of this document however is to attract newcomers and entice them to start contributing to Forrest. We have a decent generation system for static project documentation, a nice set of skins and some simple but effective DTDs. Our goals however are much more ambitious: we have compiled a dream list that lists most of them.

- Our first ambition is to support the project site generation and maintenance of other Apache projects in an automated manner, starting with our own website as a showcase. We are in the process of setting up the shell scripts and Ant tasks for this and will assist projects transitioning to Forrest.
- As it is often the case with collaborative open source development, there is no formal planning nor task assignments, and we will stick to that practice. We have however compiled a number of functional work areas:

URI Namespace Management	Forrest will offer access to a broad set of information resources using durable URIs: please review Tim Berners-Lee's and Jakob Nielsen's opinion on this. We need a unified URI Namespace management approach, bearing in mind mirroring and 'hackable' URIs.
Skins	We currently have a nice set of skins which should be solidified. Furthermore, we need

	some serious finetuning of the `forrest-site` skin that will become the new xml.apache.org look&feel.
Aggregation and Syndication	We plan to aggregate on a per-project basis a number of relevant developer resources, such as project-related news, download statistics, committer bio pages (with photos!), navigable source code listings and the like. Some of these resources need to be made available across content syndication methods such as RSS.
Build Management	Fool-proof automation of Forrest runs and site publication using secure transfer methods and `cron` jobs.
Document Types	Expanding the collection of DTDs, documenting them using formal How-Tos and example documents.
xml.apache.org	Formation of an editorial team for the main xml.apache.org website, working in close collaboration with the PMC and the different sub-project leads.
Integration	Forrest needs to coexist with existing cross-project collaboration tools such as Gump, Scarab and Eyebrowse and provide integrated access to them.
Authoring support	Supporting document authors with preconfigured XML editing solutions.
Content Management	Establish an efficient content management practice, supporting versioning, remote access and work flow, presumably supported by a CMS such as Slide.
Information Accessibility	We need to be accessible using a wide range of browsing devices operating on different platforms. Special care should be taken to support the WAI guidelines.

9. Where you can help

By now, you should have a better understanding of Forrest (if that is not the case, consider contributing clarifications to this document). We need more people to get the job done. Forrest is a fun project to work on, and there is something in it for all of us:

- XML docheads with skills for document analysis and DTDs development
- Cocoon developers creating custom Cocoon components connecting Forrest with external resources
- Graphical whizzkids for true cross-browser HTML/CSS development
- People who believe XSLT will bring peace to earth (it will, but keep that quiet)
- Ant wizards able to compete with Nicola and Stefan
- Unix shell scripting / CVS / cron gurus, preferably bearded

Just drop us a line at the forrest-dev mail list.

That is all, folks.

Revision history	
2002-05-22	Initial version, Steven Noels, stevenn.at.apache.org
2002-05-23	Various rephrasings and clarifications thanks to Ross Gardler, ross.at.saafe.org
2002-09-23	Updated the directory outline (jefft.at.apache.org)