

Apache Forrest и Javable

Подготовка структурированной документации для новой версии сайта.

Сергей Миссан

Этот документ поможет вам получить основные сведения об использовании системы Apache Forrest для подготовки материалов для сайта Javable. Вы узнаете, какими преимуществами обладает система подготовки документации Apache Forrest, как ее эффективно использовать, и как конвертировать статьи из обычного HTML в новый формат.

За последние несколько лет сайт Javable вырос из небольшого проекта в крупный сайт, лидирующий в области предоставления документации о Java и сопутствующих технологиях на русском языке. За время существования сайта с ним сотрудничало более 50-и авторов, каждый из которых использовал свой подход к подготовке документации, и, зачастую, разные программные продукты. Это привело к накоплению мелких, но часто раздражающих отличий в оформлении материалов. Также, использование разных стилей в оформлении HTML документации затрудняет обмен и обработку материалов.

Назрела необходимость в использовании единого формата для подготовки документации для сайта, и внедрения общего стиля (Javable look&feel) для ее отображения.

После серьезного изучения существующих решений, я предлагаю использовать (достаточно новый) проект [Apache Forrest](#).

1. Apache Forrest — основные особенности

Система Apache Forrest была создана для оформления сайта [xml.apache.org](#) в едином

стиле. Поскольку разные разделы сайта поддерживаются разными группами разработчиков (прямая аналогия с нашими колонками, руководствами и статьями), нужен был удобный общий формат и набор стилей, который бы позволял автоматизировать создание и поддержку сайта.

Apache Forrest написан на Java и использует код из проектов Jakarta Ant, Apache Cocoon, Apache FOP.

Вся документация создается в формате XML (см. ниже) и преобразуется в статический HTML и PDF. Одновременно создается навигация по сайту, список обновлений, могут подключаться внешние RSS/RDF feeds, и тд.

Идеология Apache Forrest близка к идеологии таких систем, как LaTeX и DocBook — формат Apache Forrest обеспечивает *логическое* форматирование документа, а форматирование *визуальное* обеспечивается соответствующим стилем (в Forrest набор XSL стилей называется skin). Достаточно изменить пару строчек в XSL стиле, "перекомпилировать" исходный XML код, и отображение всех документов изменится.

Помимо того, что вы получаете качественно оформленные HTML и PDF документы из одного комплекта XML "исходников", вы также:

- получаете документы, соответствующие стандарту (в текущей версии HTML 4.01, в ближайшей перспективе XHTML)
- автоматическую нумерацию рисунков и таблиц
- мини-содержание в начале статьи
- структурированность разделов и подразделов
- удобную навигацию для документов, состоящих из нескольких частей
- возможность использования документов с ревизиями (последовательными исправлениями)
- возможность превратить Javable в "распределенный" сайт (см. ниже)

2. Использование Apache Forrest для Javable

Forrest — относительно молодой проект. Многие возможности еще не реализованны, но даже в текущей версии использование этой системы позволяет получить профессионально оформленную документацию и сэкономить время.

Для Javable мной (СМ) была создана слегка переработанная версия Forrest, основанная

Apache Forrest u Javable

на Forrest 0.5.1.

Замечание:

Я буду постепенно синхронизировать код Javable Forrest с основной веткой Apache Forrest. В перспективе, я надеюсь, необходимость в fork-е может отпасть.

Основные изменения:

- skin javable-skin — оформляет ваши документы в соответствии со стандартами сайта Javable
- руссифицированный FOP — добавлены метрики для русских TTF шрифтов, необходимых для генерации PDF документации
- пара изменений в Ant скриптах, для упрощения жизни :)

2.1. С чего начать?

Документация на сайте Forrest: [The Forrest Primer](#), [Using Forrest](#).

Дополнительные инструкции:

Замечание:

В этой документации \$FORREST_ROOT обозначает каталог, в который был распакован архив javable-forrest, например, D:\Java\jovable-apache-forrest-0.5.1-bin. Модифицируйте пути соответствующим образом, если вы используете *NIX.

1. Установите последнюю версию SUN JDK для своей платформы.
2. Загрузите архив [jovable-apache-forrest](#) и распакуйте его в подходящий каталог.
3. Откройте файл \$FORREST_ROOT\src\documentation\sitemap.xmap. Найдите строчку `<user-config src="d:/java/jovable-apache-forrest-0.5.1-bin/context/userconfig.xml"/>` и измените абсолютный путь к файлу userconfig.xml в соответствии с вашей инсталляцией.
4. Откройте файл \$FORREST_ROOT\context\userconfig.xml. Найдите строчки, задающие метрики шрифтов (раздел ``) и измените абсолютные пути к файлам метрик и файлам шрифтов, на соответствующие вашей инсталляции и платформе.
5. Зайдите в каталог \$FORREST_ROOT, запустите `.\bin\forrest`.

6. В каталоге `$FORREST_ROOT\build\site` вы должны увидеть итоговую версию ваших документов.
7. Попробуйте модифицировать `$FORREST_ROOT\src\documentation\content\xdocs\site.xml`, и преобразовать документы из каталога `javaworld`.
8. Попробуйте модифицировать исходные XML документы `$FORREST_ROOT\src\documentation\content\xdocs\...`, и посмотрите как это отразится на результирующих документах (HTML и PDF).
9. Не забывайте запускать `.\bin\forrest` после изменений!

Предупреждение:

Не используйте команду `forrest seed` — ваш проект уже готов к использованию (в каталоге `src\documentation\content\xdocs\`)!

2.2. Краткое описание формата

Forrest использует достаточно простой XML формат для разметки документов. Многие тэги похожи на соответствующие тэги HTML, так что преобразовывать документы в этот формат можно достаточно быстро. Самый простой способ начать работать с Forrest — взять за основу один из документов в каталоге `$FORREST_ROOT\src\documentation\content\xdocs\`, скопировать его, изменить, и попробовать сгенерировать документацию. Не забывайте, что все новые документы должны быть добавлены в файл `$FORREST_ROOT\src\documentation\content\xdocs\site.xml`, иначе Forrest о них не узнает и не будет преобразовывать (подробнее см. [Menus and Linking](#)).

Исправление: (Forrest team):

Для включения изображений в PDF документы создайте подкаталог `images` в каталоге, где находится ваша статья, и добавьте туда свои графические файлы. Вы можете использовать относительный или абсолютный (по отношению к `xdocs!`) путь к файлу изображения внутри тэга `<figure>`.

Описание формата документа на сайте Forrest: [The document-v1.1 DTD](#). В текущей версии Forrest используется обновленный тип документа — [v. 1.2](#). Этот формат является предпочтительным, вся документация сайта уже преобразована в `document-v12`.

Дополнительные инструкции:

- Не забывайте устанавливать кодировку `<?xml version="1.0" encoding="windows-1251" ?>` в документах.
- XML документы должны быть *well-formed* — парные тэги закрыты, непарные содержать слеш (`
`).
- Я рекомендую использовать `<subtitle>` и `<abstract>` для статей, это улучшает их восприятие (см. исходный код этого документа).
- Таблицы могут содержать тэги заголовка колонок (`<th>`).
- Таблицы могут содержать подпись (`<caption>`) — при этом в подписи будет автоматически добавлено слово "Таблица", и таблице будет присвоен номер.
- Forrest предлагает 3 способа добавить рисунок в документ — ``, `<icon>` и `<figure>`. Я рекомендую для иллюстраций использовать `<figure>` — атрибут `alt` при этом будет преобразован в подпись к рисунку (будет автоматически добавлено слово "Рисунок", и рисунку будет присвоен номер).
- Короткие фрагменты кода могут оформляться тэгом `<code>`, для фрагментов, содержащих полную строку или несколько строк примера исходного кода следует использовать тэг `<source>`.
- Не забывайте, что некоторые тэги не могут быть использованы внутри других тэгов. Типичная ошибка — использование тэгов `<source>` или `<figure>` внутри `<p> . . . </p>`. Эти тэги должны быть использованы вне параграфа. Полный список доступных тэгов и их вложений описан в [Forrest DTD documentation](#).

Замечание:

Где заголовок моей статьи?!

Если вы уже преобразовали свою документацию и пытаетесь открыть файлы `main.shtml` — вы могли заметить, что у них не виден заголовок статьи и имя автора. На самом деле они включены, но только в виде комментариев. Это нужно для правильной интеграции файлов `main.shtml` в шаблонную структуру сайта Javable.

3. Преобразование legacy HTML документов в формат Apache Forrest

Преобразование первого документа может занять у вас порядка 1 часа. При приобретении небольшого опыта, конвертация становится относительно простым делом — затраты времени порядка 10-15 мин на документ. Основная сложность —

получить XML документ, который будет соответствовать Forrest document-v12.dtd и правильное оформление логической структуры разделов.

3.1. Приборы и материалы

Вам понадобятся:

- [HTML Tidy](#)
- Хороший текстовый редактор с заменой (желательно не только с заменой в одном файле, но и наборе файлов). Очень желательно, чтобы поддерживались регулярные выражения для поиска и замены. Многие редакторы умеют использовать HTML Tidy, если он установлен. Могу порекомендовать [JEdit](#) с плагином JTidy.

3.2. Порядок действий

HTML Tidy позволяет задавать список опций во внешнем файле (html_tidy.cnf), я рекомендую такой набор (он даже сможет вычистить кошмарный HTML код, который создает Word 2000):

```
// Javable.com HTML Tidy config file
add-xml-decl: yes
doctype: omit
tidy-mark: no
wrap: 0
word-2000: yes
alt-text: ""
output-xhtml: yes
char-encoding: raw
quote-marks: yes
logical-emphasis: yes
fix-bad-comments: no
newline: LF
//dangerous! replaces the original file. Set to "no" to be safe
write-back: no
```

Для преобразования серии документов используйте такой скрипт (работает на Windows NT):

Apache Forrest u Javable

```
@echo off

if NOT "%1" == "" goto INIT

echo Please enter drive and directory on command line.
goto EXIT

:INIT

set DirSpec=%1

if "%DirSpec:~-1%"=="\" goto START

set DirSpec=%DirSpec%

:START

REM ** You'd replace *.dat with whatever your drive
REM ** specification was. Also, you'd replace the echo
REM ** command with your command.
for /R %DirSpec% %%f in (*.html) do
    tidy -f d:\tidy.err -config html_tidy.cnf -m %%f

:EXIT

set DirSpec=
```

Этот скрипт (tidy.bat) должен быть записан в тот же каталог, где находится tidy.exe, в качестве параметра используйте имя каталога, в котором нужно преобразовать файлы.

В результате запуска HTML Tidy мы должны получить документы, которые содержат well-formed XML, мы уже в двух шагах от получения нашего Forrest документа!

Теперь нам на помощь должен прийти редактор с заменой.

Добавьте заголовок, например

```
<?xml version="1.0" encoding="windows-1251"?>
<!DOCTYPE document PUBLIC "-//APACHE//DTD Documentation V1.2//EN"
"http://apache.org/forrest/dtd/document-v12.dtd">
<document>
```

```
<header>
  <title>Apache Forrest и Javable</title>
  <subtitle>Подготовка структурированной
    документации для новой версии сайта.</subtitle>

  <authors>
    <person name="Сергей Миссан" email="webmaster@javable.com" />
  </authors>

<abstract>
  ...
</abstract>

</header>

<body>
```

Добавьте завершение документа,

```
</body>

</document>
```

Замените все `<pre>` на `<source>`. Не забудьте про закрывающие тэги.

Замените все `` на `<link href=...>`.

Внимательно просмотрите документ и замените `` на `<figure ...>`, там, где это необходимо. Не забудьте про атрибуты `alt`, `width` и `height`.

Добавьте `<caption>` в таблицы, если необходимо.

Наконец, самое важное — разбивка статьи по разделам и подразделам. Вы можете либо использовать плоскую структуру разделов

```
<section>
<title>Раздел 1</title>
....
</section>
```

```
<section>
<title>Раздел 2</title>
....
</section>
```

либо вложить раздел в другой раздел, получив подраздел

```
<section>
<title>Раздел 1</title>
....
  <section>
    <title>Раздел 1.1</title>
      ....
    </section>
  </section>
```

Каким образом разбивать документ на разделы — зависит от его логической структуры. Старайтесь избегать чрезмерной вложенности (выше 3-го уровня).

Теперь мы можем попробовать преобразовать документ, запустив `.\bin\forrest`. Если документ плохо сформирован, его валидация выдаст ошибку, и документ преобразован не будет. Типичные проблемы — незакрытые тэги или недопустимый тэг внутри другого тэга. Если документ корректен, мы получим сгенерированную копию в каталоге `$FORREST_ROOT\build\site`.

4. Использование XML редакторов

В этом разделе я кратко остановлюсь на использовании XML редакторов для подготовки документации в формате Apache Forrest.

Для начала уточним, что редактор мы хотим использовать:

1. Бесплатный
2. Визуальный (т.е. позволяющий работать с документом, как со стилизованным текстом, а не деревом тэгов)
3. Кросс-платформенный (лучше на Java)

[Список](#) не такой уж большой, как может показаться, и даже с теми редакторами, что *могут* подойти, существуют определенные сложности (например, отсутствие поддержки кириллических кодировок).

4.1. Morphon XML Editor

Перепробовав практически все, что попадало мне в руки, я решил остановиться на [Morphon XML Editor](#), тем более, что он недавно стал доступен для свободной загрузки.

Morphon использует стандартный механизм стилей CSS для "раскраски" документов, это очень удобно, потому что такой же способ используется в некоторых других XML редакторах, например, Vex. Значит есть надежда, что стиль можно будет использовать повторно.

Для редактирования forrest document v. 1.2 вам нужно загрузить архив [forrest-css-0.5.1.zip](#), и распаковать его в каталог, скажем, MORPHON_HOME\Examples. В полученной директории (Examples\forrest) вы найдете набор DTDs для Apache Forrest (каталог schema), CSS (forrest.css) и документ, который вы сейчас читаете (about.xml).

Для удобства работы в диалоге Edit/Preferences выберите закладку Catalogs и добавьте новый каталог D:\Java\XML-Editor 3.1.1\Examples\forrest\schema\catalog (с учетом пути к своей инсталляции, конечно).

Теперь нужно загрузить XML документ (/Examples/forrest/about.xml), указав путь к файлу, и указать путь к стилю (/Examples/forrest/forrest.css) и вы должны увидеть этот документ стилизованный для редактирования (см. Рисунок 1).

Можно также использовать Morphon в режиме редактирования исходного кода.

5. Известные проблемы

Я уже упоминал, что Forrest создает не только сам документ, но и навигационную структуру целого сайта. К сожалению, этот механизм еще недостаточно гибок, и создает некоторые проблемы.

Во-первых, в конце каждого документа создается список документов, которые находятся в этом же каталоге (исходя из структуры site.xml). Это не всегда удобно, поэтому иногда приходится преобразовывать документы только из одного раздела сайта, закомментировав остальные ссылки в site.xml.

Во-вторых, в локальные линки, которые не содержат имя файла (href="foobar/") Cocomoon в конце добавляет "index.html", который нам совершенно не нужен. Эта проблема решена с помощью подификации Ant скриптов. Дополнительно, недавно были внесены исправления в CVS Cocomoon, которые исправляют эту ошибку.

На этом этапе вас, как автора статьи или редактора колонки, эти проблемы не затрагивают. Вы можете просто прислать мне XML код статьи, а я преобразую его в нужный формат, включая "ручные" исправления сгенерированных документов.

6. Что дальше?

Я думаю, у вас уже созрел вопрос: "Зачем нужны эти мучения, чтобы просто получить (пусть и хорошо оформленные) HTML и PDF документы?"

На самом деле, мы получим основные преимущества от новой технологии при использовании [ForrestBot](#) — подпроекта Apache Forrest.

ForrestBot позволит нам создать полностью распределенный Javable — разные разделы сайта могут находиться на разных машинах в сети, периодически ForrestBot будет забирать XML документы, форматировать их и выкладывать на сайте.

Типичный сценарий использования:

Вы — редактор колонки, или автор руководства, или цикла статей.

- Сейчас вы присылаете свои статьи для публикации по e-mail.
- Каждый раз, когда вы хотите внести изменения в свои статьи, вам нужно прислать либо обновленную версию по e-mail, либо список исправлений для того, чтобы их внес редактор сайта.
- При использовании ForrestBot вы храните свои материалы (в XML формате Forrest) на любой сетевой машине, к которой у вас есть доступ.
- Если вам нужно внести изменения или добавить новую статью, вы делаете все поправки в своей *локальной* копии материалов, а ForrestBot обработает их автоматически и скопирует на Javable при следующем "обходе" сайтов, например, раз в неделю.
- ForrestBot достаточно "всеяден" (на самом деле, это обеспечивается возможностями Ant), и может получить XML документы либо по протоколам ftp или http, либо из вашего репозитория CVS, либо из базы данных. Вам не нужно вкладывать деньги в оборудование или хостинг — вы можете хранить свои XML документы на любом бесплатном web хостинге.

7. Заключение

Вы увидели, что использование Apache Forrest для подготовки публикаций для сайта Javable позволяет получить соответствующие стандартам и стилю сайта HTML и PDF документы. Близость формата Forrest к HTML упрощает конвертацию legacy документов. Использование ForrestBot позволяет создать полностью распределенный ресурс, дав свободу авторам и редакторам разделов в подготовке и модификации своих материалов, при этом сохраняя единый стиль и автоматизируя поддержку сайта. Дополнительные возможности Forrest включают интеграцию RSS feeds, ревизий документов, faqs, позволяющие насытить "полустатический" сайт информацией.

8. Ресурсы

- Архив дистрибутива Apache Forrest, адаптированный для сайта Javable: [javable-apache-forrest-0.5.1-bin.zip](#) (~13 Mb)
- Архив стилей CSS для визуального редактирования документов Apache Forrest в Morphon XML Editor: [forrest-css-0.5.1.zip](#) (~300 Kb)
- Сайт Apache Forrest:

- xml.apache.org/forrest/
- Morphon XML Editor:
<http://www.morphon.com/>
- Хороший редактор с массой плагинов, включая JTidy и XML:
<http://www.jedit.org/>