Forrest を使う（Using Forrest）

あなた自身のプロジェクトにおいて Forrest を使う方法のチュートリアル
A tutorial on how to use Forrest in your own projects

This is documentation for current version v0.9

これは現行バージョンの v0.9 に対するドキュメントです。

注意

本書は Using Forrest v0.9 (http://forrest.apache.org/docs_0_90/your-project.html) の日本語訳です。

翻訳者：Apache Forrest 日本語プロジェクト

序文（Introduction）

このチュートリアルは、Forrest をインストールし、新しいプロジェクトを作成する、もしくは、Forrest ベースの文書を既存のプロジェクトに追加するために、それを使うプロセスを通してあなたを導きます。

This tutorial will lead you through the process of installing Forrest, and using it to create a new project, or add Forrest-based docs to an existing project.

Forrest のインストール（Installing Forrest）

Forrest の最新リリースをダウンロードし、トップレベルにある `index.html` に従ってください。もしくは、もし開発バージョンを試したいならば、ソースから Forrest をビルドしてください。

Download the latest release of Forrest and follow the `index.html` in the top-level, or if you want to try the development version, then build Forrest from source.

環境のセットアップ（Setting up the Environment）

Forrest をダウンロードして解凍した後、環境変数を追加する必要があります。この理由は、'`forrest`' コマンドがどこでも使えるようにすることと、そのホームディレクトリとリソースを位置づけることが可能にすることです。あなたのオペレーティングシステムの管理方法を説明することは、Forrest の範囲を超えます。いくつかのヒントは以下に示されます。また、このメッセージは、Windows を使うことについて、さらに説明し、他のヒントを提供します。

After downloading and extracting Forrest, you need to add environment variables. The reason for this is so that the '`forrest`' command is available everywhere and it can locate its home directory and resources. It is beyond the scope of Forrest to explain how to manage your operating system. Some tips are listed below and this message explains further and provides other tips about using Windows.

Unix/Linux の場合：（In Unix/Linux:）

forrest ディストリビューションのトップレベルにディレクトリ移動して、以下を実行する。

change directory to the top-level of the forrest distribution and do ...

- `~/apache-forrest$ export FORREST_HOME=`pwd``
- `~/apache-forrest$ export PATH=$PATH:$FORREST_HOME/bin`

Linux/Unix に対する環境変数の永続的な設定（Permanently Setting The Environment Variables for Linux/Unix）

Export は、そのユーザに対するそのターミナル・セッションの間における変数を変更するだけで、これはテストにとって有効です。永続的に変数を追加するために、/etc/bash.bashrc（すべてのユーザに対して）または~/.bash_profile（個々のユーザに対して）を編集します。あなたが編集するファイルの最後にこれらの行を追加します：

Export only changes the variables during that terminal session for that user, this is useful for testing. To permanently add the variable edit either /etc/bash.bashrc (for all users) or ~/.bash_profile (for individual users). Add these lines to the end of the file you edit:

```
FORREST_HOME=/full/path/to/forrest
export FORREST_HOME

PATH=$PATH:$FORREST_HOME/bin
export PATH
```

Debian update-alternatives システム（The Debian update-alternatives system）

もし、あなたのシステムがアプリケーションのバイナリとその位置を管理するために、alternatives システムを使っているならば、環境変数を明示的に export する代わりに、あなたのシステムのバイナリ・ディレクトリの中に Forrest をリンクするためにそれを使うことができます：

If your system uses the alternatives system to manage application binaries and their locations, you may use that to link Forrest into your system's binary directory, instead of explicitly exporting environment variables. To check if your system has the alternatives system installed, execute this command:

```
update-alternatives --version
```

もし update-alternatives がインストールされていれば、そのバージョンが表示されるでしょう。もしそうならば、あなたは Forrest に対するエントリーを追加してもよいでしょう。update-alternatives エントリーをインストールすることは、root 特権で実行される必要があるかもしれません。いくつかのシステムにおいては、スーパーユーザとして一つのコマンドを実行する、”sudo”コマンドとともに達成されることができます：たとえば、Ubuntu の GNU/Linux システムはこの機能を使います。もしくは、あなたはインストールするためにシステム管理者に連絡をとる必要があるかもしれません。

update-alternatives will print its version if it is installed. If so, you may then add an entry for Forrest. Installing an update-alternatives entry may need to be run with root privileges. On some systems, that can be achieved with the ”sudo” command, which executes single commands as the super user: for example, Ubuntu's GNU/Linux system uses this feature. Or you may need to contact a system administrator to install.

Forrest エントリーをインストールするために、まず、Forrest の実行できるものへパスを集めて、そのバージョンも記録します。この例において、Forrest は’/opt/Apache/apache-forrest-0.8’（実行できるものは bin/forrest）の中に解凍されて置かれているとします。そしてこのコマンドを実行します（ここで、私たちは root 特権にて update-alternatives を実行するために sudo を使います。）：

To install a Forrest entry, first, gather the path to Forrest's executable, and also note its version. In this example, Forrest has been unpacked and placed into ’/opt/Apache/apache-forrest-0.8’ (the executable is at bin/forrest). Then execute this command (here we use sudo to execute update-alternatives with root privileges):

```
sudo update-alternatives --install /usr/bin/forrest forrest
'/opt/Apache/apache-forrest-0.8/bin/forrest' 800
```

このコマンドへの引数は、バイナリ・リンクに対するシステムの位置として'/usr/bin/forrest'を、このエントリーに対する update-alternatives の短縮名として'forrest'を、実際のバイナリの位置として'/opt/Apache/apache-forrest-0.8/bin/forrest'を、優先度として'800'を含みます。

The arguments to this command include '/usr/bin/forrest' as the system's location for the binary link, 'forrest' as the update-alternatives short name for this entry, '/opt/Apache/apache-forrest-0.8/bin/forrest' as the actual binary's location, and '800' as the priority.

あなたが Forrest の他のバージョンをインストールし、update-alternatives を使ってそれらを切り替えるかもしれないので、Alternatives エントリーは優先度をもちます：あなたが他のものをアクティブにするために明示的に選択するまで、最も高い優先度のエントリーがデフォルトとして選択されます。ここで、私たちは Forrest のバージョンに基づいた数値を選択します：バージョン 0.8 がインストールされていて、そして優先度 800 はいくつかの他のバージョンを追加するための余地を残します;たとえば、0.90 は優先度 900 を使うかもしれません、また 1.0 は 1000 かもしれません。

Alternatives entries have a priority because you may install other versions of Forrest, and switch among them using update-alternatives: the highest priority entry will be selected as the default until you explicitly select another one to become active. Here, we have chosen a number based on Forrest's version: version 0.8 is installed, and priority 800 leaves room for adding several other versions; for instance, 0.90 may use priority 900, and 1.0 may be 1000.

Forrest は、あなたのコマンドラインにおいて、いま入手可能でしょう。実行：forrest --help

Forrest should now be available on your command line. Execute: forrest --help

Forrest はそのヘルプ・テキストを表示すべきです。alternatives システムのさらなるヘルプのために、次を使ってください：

Forrest should print its help text. For more help with the alternatives system, use:

```
update-alternatives --help or man update-alternatives
```

Windows 2000（Windows 2000）

「マイ　コンピュータ」、「プロパティ」、「詳細設定」、「環境変数」を開きます。

Go to "My Computer", "Properties", "Advanced", "Environment Variables"

- C:\full\path\to\apache-forrest として、新しい変数 FORREST_HOME を追加します。
  add a new variable FORREST_HOME as C:\full\path\to\apache-forrest
- %PATH%;%FORREST_HOME%\bin として PATH を編集します。
  edit PATH as %PATH%;%FORREST_HOME%\bin

Windows XP の場合：（In Windows XP:）

「マイ　コンピュータ」、「プロパティ」、「詳細設定」、「環境変数」を開きます。

Go to "My Computer", "Properties", "Advanced", "Environment Variables"

- 名前：FORREST_HOME　値：C:\full\path\to\apache-forrest をもつ新しい変数をつくります。
  Create a New variable with name: FORREST_HOME value: C:\full\path\to\apache-forrest

- 現在の値の最後に;%FORREST_HOME%\bin を追加することにより PATH を編集します。
  Edit PATH by adding ;%FORREST_HOME%\bin to the end of the current value.

ForrestBar（ForrestBar）

Firefox ブラウザのツールバー拡張である ForrestBar をインストールします。さまざまな Forrest リソースに対する支援されたナビゲーションと検索を提供します。

Install the ForrestBar a toolbar extension for the Firefox browser. It provides assisted navigation and search of various Forrest resources.

forrest コマンド（The 'forrest' command）

'forrest'コマンドが何をできるのかを確認するために、’forrest -projecthelp’と入力します。*でマークされているビルドターゲットが一般的に使われます。

To see what the 'forrest' command can do, type ’forrest -projecthelp’. The build targets that are marked with * are the commonly used ones.

```
Apache Forrest.  Run 'forrest -projecthelp' to list options

Buildfile: /usr/local/svn/forrest/src/core/bin/../forrest.build.xml

    *=======================================================*
    |                 Forrest Site Builder                  |
    |                       X.Y-dev                         |
    *=======================================================*

            Call this through the 'forrest' command

Main targets:

 available-plugins     What plugins are available?
 available-skins       What skins are available?
 clean                 * Clean all directories and files generated during
                        the build
 init-plugins          Ensure the required plugins are available locally,
                       if any are not, download them automatically
 install-skin          Install the needed skin from the remote repository
 package-skin          Make a package of an existing skin
 run                   * Run Jetty (instant live webapp)
 run_custom_jetty      Run Jetty with configuration file found in the project
 run_default_jetty     Run Jetty with configuration file found in Forrest
 seed                  * Seeds a directory with a template project doc structure
 site                  * Generates a static HTML website for this project
 validate              Validate all: xdocs, skins, sitemap, etc
 validate-sitemap      Validate the project sitemaps
 validate-skinchoice   Validate skin choice
 validate-skinconf     Validate skinconf
 validate-skins        Validate skins
```

```
  validate-stylesheets  Validate XSL files
  validate-xdocs        Validate the project xdocs
  war                   * Generates a dynamic servlet-based website
                          (a packaged .war file)
  webapp                Generates a dynamic servlet-based website
                          (an unpackaged webapp).
  webapp-local          Generates a dynamic servlet-based website
                          (an unpackaged webapp). Note this webapp is suitable
                          for local execution only, use the 'war' or 'webapp'
                          target if you wish to deploy remotely.
Default target: site
```

'site'がデフォルトのターゲットなので、単にオプションなしで'forrest'を実行すると、「静的な HTML ウェブサイト」が生成されます。たとえば、トップレベルの"forrest/site-author"ディレクトリで 'forrest'を入力すると、Forrest 自身のウェブサイトがビルドされます。しかし、私たちは、あなたのプロジェクトに対する新しいサイトを構築していきます、そのため読み進めます。

As 'site' is the default target, just running 'forrest' without options will generate a "static HTML website". For example, typing 'forrest' in the top-level "forrest/site-author" directory would build Forrest's own website. But we're going to be building a new site for your project, so read on.

新しいプロジェクトの種をまく（Seeding a new project）

プロジェクトの'種をまく'とは、あなたがカスタマイズできる、あなたのプロジェクトにテンプレート・ドキュメントを追加することに対する、私たちの樹木に関する用語です。

'Seeding' a project is our own arborial term for adding a template documentation set to your project, which you can then customize.

これを試すために、（Forrest ディストリビューションの外に）完全に新しいディレクトリを作ります。そして、そこにディレクトリを変更し、'forrest seed'を実行します。これで多くのデモンストレーションのドキュメントを持つ新しいサイトがあなたにもたらされます。あなたは、"forrest seed-business"を実行することもできます。これはサイトに関する多くの質問をあなたに尋ねるでしょう。そして、標準的なシード・サイトのすべてのデモンストレーション・ページがない、より小さなサイトをつくります。

To try this out, create a completely new directory (outside the Forrest distribution), then change directory to it, and do 'forrest seed'. This will give you a new site with lots of demonstration documents. You can also do "forrest seed-business", this will ask you a number of questions about the site and will create a smaller site without all the demonstration pages of the standard seed site.

注意（Note）

forrest seed は、Forrest が何をできるのか確認するために有効です。しかし、あなたが本物のサイトをつくっているならば、forrest seed-business が初期により少ないコンテンツを持ちます。そして、そのため、編集することがより簡単になります（たとえビジネスのサイトでなかったとしても）。私たちは、将来的により多くの seed サイトを含めることを期待しています。

forrest seed is useful to see what is possible within Forrest, but if you are creating a real site forrest seed-business has less content initially, and is therefore easier to edit (even if it is not a business site). We hope to include more seed sites in the future.

もし forrest seed を実行するならば、この下にあるようなアウトプットを見るはずです：

If you run forrest seed you should see output like this below:

```
[/home/me/forrest/my-test]$ forrest seed

Apache Forrest.  Run 'forrest -projecthelp' to list options

Buildfile: /usr/local/svn/forrest/src/core/bin/../forrest.build.xml

init-props:
Loading project specific properties from
 /home/me/forrest/my-test/forrest.properties
...
echo-settings:

check-contentdir:

ensure-nocontent:

seed:
Copying 41 files to /home/me/forrest/my-test

------------------------------
~~ Template project created! ~~

Here is an outline of the generated files:

/                       # /home/me/forrest/my-test
/forrest.properties     # Optional file describing your site layout
/src/documentation/     # Doc-specific files
/src/documentation/skinconf.xml    # Info about your project used by the skin
/src/documentation/content         # Site content.
/src/documentation/content/xdocs   # XML content.
/src/documentation/content/xdocs/index.xml # Home page
/src/documentation/content/xdocs/site.xml  # Navigation file for site structure
/src/documentation/content/xdocs/tabs.xml  # Skin-specific 'tabs' file.
/src/documentation/content/xdocs/*.html,pdf # Static content files, may have subdirs
/src/documentation/resources/images   # Project images (logos, etc)
# you can create other directories as needed (see forrest.properties)

What to do now?
- Render this template to static HTML by typing 'forrest'.
  View the generated HTML in a browser to make sure everything works.
- Alternatively 'forrest run' and browse to http://localhost:8888/ live demo.
- Start adding content in xdocs/ remembering to declare new files in site.xml
- Follow the document http://forrest.apache.org/docs/your-project.html
- Provide any feedback to dev@forrest.apache.org
```

```
Thanks for using Apache Forrest
------------------------------

BUILD SUCCESSFUL
Total time: 5 seconds
```

注意（Note）

おそらくあなたが気がついたように、セオリーでは人々は必死なときだけオンラインドキュメントを読むでしょうが、私たちはまさにスクリプトにドキュメント化することが好きです。
As you have probably noticed, we like to document things right in the script, on the theory that people only read online docs when desperate :)

あなたは今すべてセットアップされたテンプレート・ドキュメント構造を持ちます：

You now have a template documentation structure all set up:

```
[/home/me/forrest/my-test]$ tree
.
|-- build
|   `-- tmp
|       `-- projfilters.properties
|-- forrest.properties
|-- src
|   `-- documentation
|       |-- README.txt
|       |-- classes
|       |   `-- CatalogManager.properties
|       |-- content
|       |   `-- xdocs
|       |       |-- images
|       |       |   |-- group-logo.gif
|       |       |   |-- group.svg
|       |       |   |-- icon.png
|       |       |   |-- project-logo.gif
|       |       |   `-- project.svg
|       |       |-- index.xml
|       |       |-- samples
|       |       |   |-- ascii-art.xml
|       |       |   |-- cocoon-pyramid.aart
|       |       |   |-- faq.xml
|       |       |   |-- ihtml-sample.ihtml
|       |       |   |-- index.xml
|       |       |   |-- openoffice-writer.sxw
|       |       |   |-- sample.xml
|       |       |   |-- sample2.xml
|       |       |   |-- sdocbook.xml
|       |       |   |-- subdir
```

```
|       |       |   |   |-- book-sample.xml
|       |       |   |   `-- index.xml
|       |       |-- site.xml
|       |       |-- tabs.xml
|       |       |-- hello.pdf
|       |       |-- test1.html
|       |       `-- test2.html
|       `-- resources
|           `-- images
|           `-- schema
|           `-- stylesheets
|       |-- sitemap.xmap
|       |-- skinconf.xml
|       `-- translations
|           |-- langcode.xml
|           |-- languages_en.xml
|           |-- languages_es.xml
|           |-- menu.xml
|           |-- menu_af.xml
|           |-- menu_de.xml
|           |-- menu_es.xml
|           |-- menu_it.xml
|           |-- menu_no.xml
|           |-- menu_ru.xml
|           |-- menu_sk.xml
|           |-- tabs.xml
|           `-- tabs_es.xml
```

これを HTML に書き出すために、'forrest'とタイプします。build/site ディレクトリの中に書き出された HTML サイトが手に入るでしょう。

To render this to HTML, type 'forrest'. You should have a HTML site rendered into the build/site directory:

新しいコンテンツを追加する練習をします。src/documentation/content/xdocs ディレクトリに移動し、新しいドキュメントとして my-new-file.xml を作るために index.xml をコピーします。少しテキストを変更するためにそれを編集します。他のエントリーの一つをコピーし、それを適切に変更することにより、site.xml にエントリーを追加します。今、結果を見るために'forrest'を実行します。

Practise with adding new content. Change to the directory src/documentation/content/xdocs and copy the file index.xml to create my-new-file.xml as a new document. Edit it to change some text. Add an entry to site.xml by copying one of the other entries and changing it to suit. Now do 'forrest' to see the result.

既存のプロジェクトの種をまく（Seeding an existing project）

上のセクションにおいて、新しいプロジェクトを作るために、空のディレクトリにおいて'forrest seed'を実行します。もし Forrest ドキュメントに追加したい既存のコードベースがあれば、あなたのプロジェクトのベース・ディレクトリにおいて'forrest seed'を実行してください。そして、Forrest ドキュメント構造はあなたのプロジェクトに融合するでしょう。この手順は一度だけ実行される必要があります。

In the section above, we have run 'forrest seed' in an empty directory to create a new project. If you have an existing codebase to which you want to add Forrest docs, then run 'forrest seed' in your project base directory, and the Forrest doc structure will be grafted onto your project. This procedure only needs to be done once.

あなたのプロジェクトが既に XML ドキュメントを持っているならば、Forrest に適合させるためにあなたのプロジェクトのディレクトリを再配置するよりは、XML ソースがどこにあるかを Forrest に教えるほうが簡単でしょう。これは、forrest.properties を編集することによりできます（より詳細は Changing the layout セクションを参照ください）。

If your project already has XML documentation, it may be easier to tell Forrest where the XML sources are, rather than rearrange your project directories to accommodate Forrest. This can be done by editing forrest.properties (consult the Changing the layout section for more details).

あなたのプロジェクトをカスタマイズする（Customizing your project）

テンプレート・ドキュメントでプロジェクトの種を植えるならば、それをあなたのプロジェクトへとカスタマイズしたいと思うでしょう。ここで我々は、スキンを修正したり、プロジェクトのレイアウトを変えたりします。

Having seeded a project with template docs, you will now want to customize it to your project's needs. Here we will deal with configuring the skin, and changing the project layout.

Forrest スキンを編集する：skinconf.xml（Configuring the Forrest skin: skinconf.xml）

多くの Forrest スキンは、以下の例のように見られる、ひとつの XML ファイル src/documentation/skinconf.xml によりカスタマイズされることができます。警告：以下の例は、古いものかもしれません。新しい'forrest seed'サイトから最新の設定ファイルを手に入れてください。

Most Forrest skins can be customized through a single XML file, src/documentation/skinconf.xml, which looks like the following example. Warning: The following example might be out-of-date. Obtain a recent configuration file from a new 'forrest seed' site.

```
<!--
Skin configuration file. This file contains details of your project,
which will be used to configure the chosen Forrest skin.
-->

 <!DOCTYPE skinconfig PUBLIC
        "-//APACHE//DTD Skin Configuration V0.8-2//EN"
        "skinconfig-v08-2.dtd">

<skinconfig>
  <!-- To enable lucene search add provider="lucene"
    Add box-location="alt" to move the search box to an alternate location
    (if the skin supports it) and box-location="all" to show it in all
    available locations on the page.  Remove the <search> element to show
    no search box.
  -->
  <search name="MyProject" domain="example.org"/>

  <!-- Disable the print link? If enabled, invalid HTML 4.0.1 -->
  <disable-print-link>true</disable-print-link>
  <!-- Disable the PDF link? -->
  <disable-pdf-link>false</disable-pdf-link>
  <!-- Disable the xml source link? -->
  <!-- The xml source link makes it possible to access the xml rendition
    of the source frim the html page, and to have it generated statically.
    This can be used to enable other sites and services to reuse the
    xml format for their uses. Keep this disabled if you don't want other
    sites to easily reuse your pages.-->
  <disable-xml-link>true</disable-xml-link>
```

```
<!-- Disable navigation icons on all external links? -->
<disable-external-link-image>false</disable-external-link-image>

<!-- Disable w3c compliance links? -->
<disable-compliance-links>false</disable-compliance-links>
<!-- Render mailto: links unrecognisable by spam harvesters? -->
<obfuscate-mail-links>true</obfuscate-mail-links>

<!-- mandatory project logo
     skin: forrest-site renders it at the top -->
<project-name>MyProject</project-name>
<project-description>MyProject Description</project-description>
<project-url>http://example.org/myproj/</project-url>
<project-logo>images/project.png</project-logo>
<!-- Alternative static image:
<project-logo>images/project-logo.gif</project-logo> -->

<!-- optional group logo
     skin: forrest-site renders it at the top-left corner -->
<group-name>MyGroup</group-name>
<group-description>MyGroup Description</group-description>
<group-url>http://example.org</group-url>
<group-logo>images/group.png</group-logo>
<!-- Alternative static image:
<group-logo>images/group-logo.gif</group-logo> -->

<!-- optional host logo (e.g. sourceforge logo)
     skin: forrest-site renders it at the bottom-left corner -->
<host-url></host-url>
<host-logo></host-logo>

<!-- relative url of a favicon file, normally favicon.ico -->
<favicon-url></favicon-url>

<!-- The following are used to construct a copyright statement -->
<year>2005</year>
<vendor>The Acme Software Foundation.</vendor>
<!-- The optional copyright-link URL will used as a link in the
  copyright statement
<copyright-link>http://www.apache.org/licenses/</copyright-link>
-->

<!-- Some skins use this to form a 'breadcrumb trail' of links.
  If you don't want these, then set the attributes to blank.
  The DTD purposefully requires them.
  Use location="alt" to move the trail to an alternate location
```

```
  (if the skin supports it).
-->
<trail>
  <a1 name="myGroup" href="http://www.apache.org/"/>
  <a2 name="myProject" href="http://forrest.apache.org/"/>
  <a3 name="" href=""/>
</trail>

<!-- Configure the TOC, i.e. the Table of Contents.
@max-depth
 how many "section" levels need to be included in the
 generated Table of Contents (TOC).
@min-sections
 Minimum required to create a TOC.
@location ("page","menu","page,menu")
 Where to show the TOC.
-->
<toc max-depth="2" min-sections="1" location="page"/>

<!-- Heading types can be clean|underlined|boxed  -->
<headings type="boxed"/>

<extra-css>
  <!-- A sample to show how the class attribute can be used -->
  p.quote {
    margin-left: 2em;
    padding: .5em;
    background-color: #f0f0f0;
    font-family: monospace;
  }
</extra-css>

<colors>
<!-- CSS coloring examples omitted for brevity -->
</colors>

<!-- Settings specific to PDF output.  -->
<pdf>
  <!--
     Supported page sizes are a0, a1, a2, a3, a4, a5, executive,
     folio, legal, ledger, letter, quarto, tabloid (default letter).
     Supported page orientations are portrait, landscape (default
     portrait).
     Supported text alignments are left, right, justify (default left).
  -->
  <page size="letter" orientation="portrait" text-align="left"/>
```

```
    <!--
       Margins can be specified for top, bottom, inner, and outer
       edges. If double-sided="false", the inner edge is always left
       and the outer is always right. If double-sided="true", the
       inner edge will be left on odd pages, right on even pages,
       the outer edge vice versa.
       Specified below are the default settings.
    -->
    <margins double-sided="false">
      <top>1in</top>
      <bottom>1in</bottom>
      <inner>1.25in</inner>
      <outer>1in</outer>
    </margins>

    <!--
      Print the URL text next to all links going outside the file
    -->
    <show-external-urls>false</show-external-urls>
  </pdf>

  <!-- Credits are typically rendered as a set of small clickable
    images in the page footer -->
  <credits>
    <credit>
      <name>Built with Apache Forrest</name>
      <url>http://forrest.apache.org/</url>
      <image>images/built-with-forrest-button.png</image>
      <width>88</width>
      <height>31</height>
    </credit>
    <!-- A credit with @role='pdf' will have its name and url
      displayed in the PDF page's footer. -->
  </credits>

</skinconfig>
```

あなたのプロジェクトのために、このファイルをカスタマイズしてください。'project-logo'と'group-logo'要素において言われている images/ディレクトリは、src/documentation/resources/imagesディレクトリ（この対応付けはサイトマップにより自動的に実行されます）に対応します。

Customise this file for your project. The images/ directory mentioned in 'project-logo' and 'group-logo' elements corresponds to the src/documentation/resources/images directory (this mapping is done automatically by the sitemap).

このファイルを編集した（そして、それが有効な XML であることが保証される）ならば、サイトのルートにて'forrest'コマンドを再実行してください、そうすればサイトは更新されるでしょう。

Having edited this file (and ensured it is valid XML), re-run the 'forrest' command in the site root, and the site would be updated.

レイアウトを変更する：forrest.properties（Changing the layout: forrest.properties）

あなたが主要なファイル・タイプをどこに置いたのか Forrest に伝える限り、Forrest はあなたのプロジェクトにおいて、あなたが望むどこにでもファイルを置くことを許します。

Forrest allows you to place files anywhere you want in your project, so long as you tell Forrest where you have placed the major file types.

forrest.properties ファイルは、あなたのディレクトリ・レイアウトから Forrest のものへと対応付けます。もしあなたがあなたのサイトを'forrest seed'で生成しているならば、すべてのエントリーがコメントアウトされている、事前に書かれたものを手に入れるでしょう。

The forrest.properties file maps from your directory layout to Forrest's. If you generated your site with 'forrest seed', you will have one pre-written, with all the entries commented out.

注意（Note）

あなたはそれらを何か違うものに変更しようとするならば、エントリーをアンコメントだけする必要があります。もしあなたが'forrest seed'のデフォルトと同期を保つならば、更新するたびに差分をとることが簡単です。
You only need to un-comment entries if you are going to change them to something different. If you keep in synchronisation with the 'forrest seed' defaults, then it is easy to diff each time that you update.

おもな（デフォルト値をもつ）エントリーは以下です：

The main entries (with default values) are:

```
# Properties that must be set to override the default locations
#
# Parent properties must be set. This usually means uncommenting
# project.content-dir if any other property using it is uncommented

#project.content-dir=src/documentation
#project.conf-dir=${project.content-dir}/conf
#project.sitemap-dir=${project.content-dir}
#project.xdocs-dir=${project.content-dir}/content/xdocs
#project.resources-dir=${project.content-dir}/resources
#project.stylesheets-dir=${project.resources-dir}/stylesheets
#project.images-dir=${project.resources-dir}/images
#project.schema-dir=${project.resources-dir}/schema
#project.skins-dir=${project.content-dir}/skins
#project.skinconf=${project.content-dir}/skinconf.xml
#project.lib-dir=${project.content-dir}/lib
#project.classes-dir=${project.content-dir}/classes
```

たとえば、もし src/documentation/content/xdocs よりも src/xdocs に XML ドキュメントを保持したいと思うならば、project.xdocs-dir に対する定義を変更するだけです。

For example, if you wish to keep XML documentation in src/xdocs rather than src/documentation/content/xdocs simply change the definition for project.xdocs-dir

```
project.xdocs-dir=src/xdocs
```

たとえば、簡単な Maven フォーマットをまねるために：

For example, to emulate the simple Maven format:

```
/xdocs
/xdocs/images
/xdocs/stylesheets
```

これが必要となるプロパティ定義です：

Here are the required property definitions:

```
project.content-dir=xdocs
project.sitemap-dir=${project.content-dir}
project.xdocs-dir=${project.content-dir}
project.stylesheets-dir=${project.content-dir}/stylesheets
project.images-dir=${project.content-dir}/images
project.skinconf=${project.content-dir}/skinconf.xml
```

注意（Note）

内部的に、Forrest は、特定のディレクトリをデフォルトの src/documentation/content/xdocs 構造の中に再配置します。上のレイアウトにおいて、私たちは重なり合うディレクトリを持ちます。そのため、重複したファイルとなるでしょう。この小さな不具合は通常問題を起こしません；ただ、すべてのリンクはサイトマップを通して決められることを常に覚えておいてください。
Internally, Forrest rearranges the specified directory into the default src/documentation/content/xdocs structure. In the layout above, we have overlapping directories, so you will end up with duplicate files. This small glitch doesn't usually cause problems; just always remember that all links are resolved through the sitemap.

コンテンツを追加する（Adding content）

これからあなた自身のコンテンツを src/documentation/content/xdocs に追加することを始めます。

Now you can start adding content of your own, in src/documentation/content/xdocs

site.xml（site.xml）

新しい xml ドキュメントを追加するときに、あなたはプロジェクトの site.xml ファイルにエントリーを追加します。この site.xml はサイトのインデックスのようなもので、標準のスキンにおいてリンクの垂直列として書かれます。例として、Forrest 自身の xdocs を見てください。site.xml に関するより詳細の情報は、Menus and Linking ドキュメントにて提供されます。

When adding a new xml document, you would add an entry to the project's site.xml file. This site.xml is like a site index, and is rendered as the vertical column of links in the default skin. Look at Forrest's own xdocs for an example. More detailed info about site.xml is provided in the document Menus and Linking.

tabs.xml（tabs.xml）

tabs.xml ファイルは、「タブ」を作るときに使われます。これは、ユーザがあなたのサイトのセクションへと素早くジャンプすることを可能にします。より詳細については menu generation ドキュメントを参照ください。そして再び、使い方の例に対して、Forrest 自身のドキュメントを参照します。

The tabs.xml file is used to produce the 'tabs'. which enable users to quickly jump to sections of your site. See the menu generation documentation for more details, and again, consult Forrest's own docs for a usage example.

あなたは、タブの一つか二つのレベルを持つことができます。上の画像は、単一レベルを示しています。しかしながら、その親タブが選択されたときだけ、第二レベルが表示されるように作ることができます。たとえば、以下の tabs.xml 断片は、どのトップレベルのタブが選択されたかに依存して、一つか二つのタブを表示します。第一行は二つのタブを持ちます：一つは How-Tos とラベルされたもの、もう一つは Apache XML Projects とラベルされたもの。How-Tos タブが選ばれたとき、タブの第二行はないでしょう。しかしながら、Apache XML Projects タブが選択されたとき、タブの第二行はまず以下が表示されるでしょう。

You can have one or two levels of tabs. The images above show a single level. However, you can create a second level that will only be displayed when its parent tab is selected. For example, the tabs.xml snippet below will display either one or two rows of tabs, depending on which of the top level tabs is selected. The first row will have two tabs: one labelled How-Tos and the other labelled Apache XML Projects. When the How-Tos tab is selected there will be no second row of tabs. However, when the Apache XML Projects tab is selected, a second row of tabs will be displayed below the first.

```
<tab label="How-Tos" dir="community/howto/"/>
<tab label="Apache XML Projects" href="http://xml.apache.org">
  <tab label="Forrest" href="http://forrest.apache.org/"/>
  <tab label="Xerces" href="http://xml.apache.org/xerces"/>
</tab>
```

画像（Images）

画像は通常 resources/images/ディレクトリにあります。デフォルトのサイトマップは、image タグが典型的に<figure src="images/project-logo.png" alt="Project Logo"/>のように見えるために、このディレクトリを images/にマップします。

Images usually go in the resources/images/ directory. The default sitemap maps this directory to images/ so image tags will typically look like <figure src="images/project-logo.png" alt="Project Logo"/>

応用カスタマイズ:sitemap.xmap(Advanced customizations: sitemap.xmap)

Cocoon サイトマップは、XML ソースからコンテンツ（HTML、PDF 他）を生成するためのルールのセットです。Forrest は、日常的なサイトに適している、デフォルトのサイトマップを持ちます。たとえば、Forrest website それ自身はデフォルトのサイトマップを使います。

The Cocoon sitemap is a set of rules for generating content (HTML, PDFs etc) from XML sources. Forrest has a default sitemap, which is adequate for everyday sites. For example, the Forrest website itself uses the default sitemap.

ときどき、ある人はルールのデフォルト・セットを超える必要があります。その Cocoon バックエンドが仮想的にすべての処理パイプラインが定義されるのを許可するので、これは Forrest が本当に輝いているところです。たとえば、ある人はこうすることができます：

Sometimes, one needs to go beyond the default set of rules. This is where Forrest really shines, because its Cocoon backend allows virtually any processing pipeline to be defined. For example, one can:

- XSLT スタイルシートを用いて、カスタム XML コンテント・タイプを変換します
  Transform custom XML content types with XSLT stylesheets.
- SVG XML ファイルから PNG もしくは JPEG イメージを作ります（注意：Forrest のサイトマップは今これをネイティブに実行します。）
  Generate PNG or JPEG images from SVG XML files. (**Note:** Forrest's sitemap now does this natively.)
- あなたのサイトのコンテンツに外部 XML フィード（たとえば RSS）を統合します（注意：例として `issues.xmap` を参照ください）
  Integrate external XML feeds (e.g. RSS) into your site's content. (**Note:** See `issues.xmap` for an example.)
- XML ソースをアグリゲーション経由で統合します、もしくは、「仮想的な」ファイルとして手に入れられる大きな XML ソースからコンテンツをつくります。（注意：Forrest はデフォルトのサイトマップにおいてアグリゲーションを広大に使用します。それは、標準的な名前 `wholesite.html` と `wholesite.pdf` として手に入れられる、サイト全体の HTML と PDF も定義します。）
  Merge XML sources via aggregation, or make content from large XML sources available as "virtual" files. (**Note:** Forrest makes extensive use of aggregation in the default sitemaps. It also defines a whole-site HTML and PDF, available as the standard names `wholesite.html` and `wholesite.pdf`.)
- XML databases のような外来のソースからコンテンツを読みます
  Read content from exotic sources like XML databases.
- 能力のすべての Cocoon の巨大な配列を統合します。可能性は、最新の Cocoon ディストリビューションをダウンロードしてサンプルを動かしてみることで、もっとも評価されます。
  Integrate any of Cocoon's vast array of capabilities. The possibilities are best appreciated by downloading the latest Cocoon distribution and playing with the samples.

Forrest のサイトマップは `main/webapp/*.xmap` にあります。

The Forrest sitemaps are at `main/webapp/*.xmap`

あなたは、あなたのプロジェクトの src/documentation ディレクトリ（または${project.sitemap-dir}が指し示すどこか）へ事前に処理されたサイトマップを追加することができます。'forrest seed site'から簡単な sitemap.xmap のコピーを手に入れてください。

You can add pre-processing sitemaps to your project src/documentation directory (or wherever ${project.sitemap-dir} points to). Get a copy of a simple sitemap.xmap from a 'forrest seed site'.

あらゆるマッチは、デフォルトの Forrest サイトマップによって取り扱われることを通して、取り扱われません ー 明らかにきわめてパワフルです。能力は、「Using project sitemaps」に記載されていて、いくつかの動いた例が以下のここのセクションにおいて示されています。

Any match that is not handled, passes through to be handled by the default Forrest sitemaps - obviously extremely powerful. The capability is described in "Using project sitemaps" and some worked examples are shown in the following sections here.

注意（Note）

私たちは、Apache Cocoon サイトマップを理解するために時間を費やすようあなたにアドバイスします。Cocoon sitemap、Cocoon concepts と関連するコンポーネントのドキュメントを参照してください。Forrest サイトマップは複数のファイルへと分割されています。おもな一つは、他のものへ委譲している **sitemap.xmap** です。デフォルトのサイトマップのツアーに対して、Sitemap Reference を参照してください。

We advise you to spend time to understand the Apache Cocoon sitemap. See Cocoon sitemap and Cocoon concepts and related component documentation. The Forrest sitemap is broken into multiple files. The main one is **sitemap.xmap** which delegates to others. See the Sitemap Reference for a tour of the default sitemap.

例：新しいコンテント・タイプを追加する（Example: Adding a new content type）

ドキュメントのタイプを検知して、特別な取り扱いをするために、二つの方法があります。より完全なソリューションが、以下の応用セクションで記載されています。しかしながら、この基本的な方法も理解する価値があります。

There are two methods for detecting types of documents and doing special handling. The more complete solution is described in the advanced section below. However, this basic method is also worth understanding.

この作られた例にしたがってください。新しいディレクトリで、'forrest seed'を実行してください、そして、このセクションに記載されているステップにしたがってください。

Follow this worked example. In a fresh directory do 'forrest seed' then follow the steps described in this section.

サンプル・シナリオは、私たちが特定のソフトウェア・パッケージに対する特別なダウンロード・リストを持つというものです。ダウンロード情報をカスタマイズした XML フォーマットで表現することがベストです。このことは、それがそれ自身のドキュメント・タイプ宣言をもつということを意味します。私たちは、この新しいドキュメント・タイプを私たちのプロジェクトのサイトマップを通して検知する必要があります。また、この DTD のローカルのコピーへのマッピングも提供する必要があります。

An example scenario is that we have a specialised list of downloads for a certain software package. It would be best to represent the download information in a custom XML format. This means that it will have its own document type declaration. We will need to detect this new document type via our project sitemap and also provide a mapping to a local copy of this DTD.

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE document PUBLIC "-//Acme//DTD Download Documentation V1.0//EN"
  "dtd/download-v10.dtd">
<document>
  <header>
    <title>Downloading Binaries</title>
  </header>
  <body>
    <section id="download">
      <title>Downloading binaries</title>
      <p>
        Here are the binaries for FooProject
      </p>
      <release version="0.9.13" date="2002-10-11">
        <downloads>
          <file
            url="http://prdownloads.sf.net/aft/fooproj-0.9.13-bin.zip?download"
            name="fooproj-0.9.13-bin.zip"
            size="5738322"/>
          <file
            url="http://prdownloads.sf.net/aft/fooproj-0.9.13-src.zip?download"
            name="fooproj-0.9.13-src.zip"
            size="10239777"/>
        </downloads>
      </release>
      <release version="0.9.12" date="2002-10-08">
        <downloads>
          <file
            url="http://prdownloads.sf.net/aft/fooproj-0.9.12-src.zip?download"
            name="fooproj-0.9.12-src.zip"
            size="10022737"/>
         </downloads>
       </release>
    </section>
    <section id="cvs">
      <title>Getting FooProject from CVS</title>
      <p>....</p>
    </section>
  </body>
</document>
```

この"download.xml"と呼ばれるファイルは、あなたのコンテンツ・ディレクトリ（一般的にsrc/documentation/content/xdocs）とsite.xmlに追加されたエントリーに置かれるでしょう。

This file called "download.xml" would be placed in your content directory (typically src/documentation/content/xdocs) and an entry added to site.xml

これらの特別なタグを取り扱うために、ある人は、それらを中間的な Forrest xdocs 構造に変換するためにスタイルシートを書くでしょう。ここにそのようなスタイルシート（"download-to-document.xsl"と呼ばれる）があります。

To handle these special tags, one would write a stylesheet to convert them to the intermediate Forrest xdocs structure. Here is such a stylesheet, called "download-to-document.xsl" ...

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet
  version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="release">
    <section id="{@version}">
      <title>Version <xsl:value-of select="@version"/> (released
          <xsl:value-of select="@date"/>)</title>
      <table>
        <tr><th>File</th><th>Size</th></tr>
        <xsl:apply-templates select="downloads/*"/>
      </table>
    </section>
  </xsl:template>

  <xsl:template match="file">
    <tr>
      <td><a href="{@url}"><xsl:value-of select="@name"/></a></td>
      <td><xsl:value-of
           select="format-number(@size div (1024*1024), '##.##')"/> MB</td>
    </tr>
  </xsl:template>

  <xsl:template match="@* | node() | comment()">
    <xsl:copy>
      <xsl:apply-templates select="@*"/>
      <xsl:apply-templates/>
    </xsl:copy>
  </xsl:template>

</xsl:stylesheet>
```

デフォルトのスタイルシートのディレクトリ、src/documentation/resources/stylesheets（もしくは${project.stylesheets-dir}が指し示すどこか）に、このファイルを置いてください。

Place this file in the default stylesheets directory, src/documentation/resources/stylesheets (or wherever ${project.stylesheets-dir} points).

今、私たちは Forrest の中間 xdocs 構造の中へカスタマイズした xml 構造の変換を制御するためにプロジェクト・サイトマップを作るでしょう。

Now we will create a project sitemap to control the transformation of our custom xml structure into the Forrest intermediate xdocs structure.

注意（Note）

Sitemap Reference は、サイトマップがどのように働くのかについて詳細を提供します。
The Sitemap Reference provides details about how the sitemap works.

ファイル src/documentation/sitemap.xmap に以下のマッチを追加します。

Add the following match to the file src/documentation/sitemap.xmap ...

```
<?xml version="1.0"?>
<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">
 <map:pipelines>
  <map:pipeline>
  ...

   <map:match pattern="**download.xml">
    <map:generate src="{properties:content.xdocs}{1}download.xml" />
    <map:transform src="{properties:resources.stylesheets}/download-to-document.xsl" />
    <map:serialize type="xml"/>
   </map:match>

  </map:pipeline>
 </map:pipelines>
</map:sitemap>
```

この特定の"download"ドキュメントのみに対して、本体コンテンツに対するリクエストを割り込みます。そして、それを習慣的な Forrest "document"フォーマットへと変換します。標準的な Forrest の機構は、ナビゲーション・メニュー等を集約して取り扱い、一般的なスキンを適用します。

That will intercept the request for the body content, for only this specific "download" document, and will transform it into the intermediate Forrest "document" format. The normal Forrest machinery will handle the aggregation with navigation menus etc. and will apply the normal skin.

新しい DTD を登録する（Registering a new DTD）

言い換えれば、デフォルトで、Forrest は有効であるすべての XML ファイルを必要とします。DOCTYPE 宣言と関連付けられた DTD を持ち、それを有効にしなければいけません。私たちの新しい'downloads' ドキュメント・タイプは例外がありません。XML Validation ドキュメントは、新しいドキュメント・タイプを登録する方法を示す、この例を続けます。要約すると、これは以下を取り込みます：

By default, Forrest requires that all XML files be valid, i.e. they must have a DOCTYPE declaration and associated DTD, and validate against it. Our new 'downloads' document type is no exception. The XML Validation document continues this example, showing how to register a new document type. Briefly, this involves:

- 新しい DTD を作ります、もしくは（私たちのケースでは）既存のものを拡張します
  Create a new DTD or (in our case) extend an existing one.
- ${project.schema-dir}/dtd ディレクトリに新しい DTD を配置します
  Place the new DTD in the ${project.schema-dir}/dtd directory.

- DOCTYPE パブリック ID から関連する DTD ファイルへのマップを有効にするために、XML カタログを追加します。
  Add an XML Catalog to enable a mapping from the DOCTYPE public id to the relevant DTD file.
- あなたのカタログについてシステムに教えてください。
  Tell the system about your catalog.

注意（Note）

これらのステップに対する完全な記述のために XML Validation を参照してください。
Please see XML Validation for the complete description for those steps.

例：新しいコンテントタイプを追加する（応用）（Example: Adding a new content type (advanced)）

前の例にあるシンプルなユーザ・サイトマップは、シンプルなシミュレーションにとって素晴らしいです。「Download DTD」記事への完全なソリューションに対して、私たちは、ソース・ドキュメントのタイプに依存して異なる処理をする、より応用的なサイトマップが必要です。

The simple user sitemap in the previous example is fine for simple situations. For a complete solution to the "Download DTD" issue we need a more advanced sitemap which will do different processing depending on the type of the source document.

私たちは、脱線して、パワフルな SourceTypeAction (content aware pipelines) を説明する必要があります。それは、ドキュメントのタイプに関するヒントを探すために、ドキュメントのトップ部分にかすかに見える、Cocoon のサイトマップ・コンポーネントです。それは四つの方法を持ちます：ドキュメント宣言、ドキュメント要素と名前空間、処理命令、w3c-xml スキーマ。

We need to digress and explain the powerful SourceTypeAction (content aware pipelines). It is a Cocoon sitemap component that peeks at the top-part of a document to look for hints about the type of the document. It has four methods: document-declaration, document-element and namespace, processing-instruction, w3c-xml-schema.

今から、ドキュメント・タイプ宣言を検知するために SourceTypeAction を使う、特定の例に戻ります。私たちにサイトマップを示して、それを説明してください。

Now to return to our specific example which uses SourceTypeAction to detect the Document Type Declaration. Let us show the sitemap and then explain it.

```
<?xml version="1.0"?>
<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">

 <map:components>
  <map:selectors default="parameter">
   <map:selector logger="sitemap.selector.parameter"
       name="parameter" src="org.apache.cocoon.selection.ParameterSelector" />
  </map:selectors>
  <map:actions>
   <map:action logger="sitemap.action.sourcetype" name="sourcetype"
       src="org.apache.forrest.sourcetype.SourceTypeAction">
    <sourcetype name="download-v1.0">
     <document-declaration
        public-id="-//Acme//DTD Download Documentation V1.0//EN" />
```

```
    </sourcetype>
    <sourcetype name="download-v1.1">
     <document-declaration
        public-id="-//Acme//DTD Download Documentation V1.1//EN" />
    </sourcetype>
   </map:action>
  </map:actions>
 </map:components>

 <map:pipelines>
  <map:pipeline>
   <map:match pattern="**download.xml">
    <map:generate src="{properties:content.xdocs}{1}download.xml" />
    <map:act type="sourcetype" src="{properties:content.xdocs}{1}download.xml">
     <map:select type="parameter">
      <map:parameter name="parameter-selector-test" value="{sourcetype}" />
      <map:when test="download-v1.0">
       <map:transform
          src="{properties:resources.stylesheets}/download-to-document.xsl" />
      </map:when>
      <map:when test="download-v1.1">
       <map:transform
          src="{properties:resources.stylesheets}/downloadv11-to-document.xsl" />
      </map:when>
     </map:select>
    </map:act>
    <map:serialize type="xml"/>
   </map:match>
  </map:pipeline>
 </map:pipelines>
</map:sitemap>
```

これは、メインの main/webapp/forrest.xmap サイトマップの中で起こる、処理のタイプです。私たちは、同じような取り扱いを私たちのプロジェクトのサイトマップに追加しました。基本的に、これは、ドキュメント・タイプを検知するために、SourceTypeAction (content aware pipelines)を使います。新しい download-v11.dtd は、上述されたようにあなたのプロジェクトのカタログへも追加する必要があります。

This is the type of processing that happens in the main main/webapp/forrest.xmap sitemap. We have added similar handling to our project sitemap. Basically, this uses the SourceTypeAction (content aware pipelines) to detect the doctype. The new download-v11.dtd needs to be also added to your project Catalog as described above.

プロジェクトのサイトマップは参照されるための最初のサイトマップなので、サイトマップ要素は使われうる前に宣言されなければいけないことに注意してください。

Note that any sitemap component must be declared before it can be used, because the project sitemap is the first sitemap to be consulted.

例：外部 RSS コンテンツを統合する（Example: integrating external RSS content）

前の例と似ていて、私たちのプロジェクトの sitemap.xmap に対応付けを提供することにより、単に私たちのサイトの中に RSS を統合することができます。

Similar to the previous example, we can integrate RSS into our site simply by providing a match in our project sitemap.xmap ...

```
<?xml version="1.0"?>
<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">
 <map:pipelines>
  <map:pipeline>

   <map:match pattern="**weblog.xml">
    <map:generate src="http://blogs.cocoondev.org/stevenn/index.rss"/>
    <map:transform src="{forrest:forrest.stylesheets}/rss-to-document.xsl"/>
    <map:serialize type="xml"/>
   </map:match>

   <map:match pattern=".......">
    <!-- handle other project-specific matches -->
   </map:match>
  </map:pipeline>
 </map:pipelines>
</map:sitemap>
```

あなたは、中心的な Forrest の rss-to-document.xsl をあなたのプロジェクトにコピーし、あなたの要望に合わせてカスタマイズし、src="{properties:resources.stylesheets}/rss-to-document.xsl"でそれを参照したいと思うでしょう。それからもちろん、あなたは weblog.html にリンクするために、エントリーを site.xml に加えるでしょう。

You will probably want to copy the core Forrest rss-to-document.xsl to your project, customise it to your needs, and refer to it with src="{properties:resources.stylesheets}/rss-to-document.xsl". Then of course you would add an entry to site.xml to link to weblog.html

Forrest スキン（Forrest skins）

Forrest はコンテンツを表示から分離しているので、すぐにサイトのルック・アンド・フィールを変更するために、異なる「スキン」をプラグインすることができます。Forrest は、一つの主なスキン、pelt、開発のさまざまな状態にあるいくつかの他のものを提供します。

As Forrest separates content from presentation, we can plug in different "skins" to instantly change a site's look and feel. Forrest provides one primary skin, pelt, and some others in various states of development.

スキンを変更するために、project.skin=pelt またはいくつかの他のスキン名をセットするために、forrest.properties ファイルを編集してください。もしダイナミック・モードで実行しているならば、あなたは新しいスキンを参照するために Forrest を再起動する必要があります。

To change the skin, edit the forrest.properties file to set project.skin=pelt or some other skin name. If running in dynamic mode you need to restart Forrest in order to see the new skin.

注意（Note）

Forrest は、編集可能であり、それゆえほとんどのプロジェクトの要求を満たすべきである default skins のコレクションを提供します。狙いは、外部スキンが必要とされないように、多くの能力を提供することです。
Forrest supplies a collection of default skins which are configurable and so should meet the needs of most projects. The aim is to provide many capabilities so that extra skins are not needed.

スキンの設定（Configuration of skins）

すべての設定は、あなたのプロジェクトの src/documentation/skinconf.xml ファイル経由で行われます。それは、それぞれの可能性を記述するために、多くのコメントを含みます。どうぞそれらを読んでください、ここで繰り返す点は何もありません。

All configuration is done via your project src/documentation/skinconf.xml file. It contains many comments to describe each capability. Please read those, there is no point repeating here.

新しいスキンを定義する（Defining a new skin）

メーリングリストであなたの要望を議論することを検討します。中心的なスキンの予定されている強化があるかもしれません。あなた自身のスキンを書くよりは、中心的なスキンへのあなたの拡張を貢献することも考えてください。あなたが更新と管理の問題をつくられうることを心に留めておきます。なんとしても、…

Consider discussing your needs on the mailing lists. There may be planned enhancements to the core skins. Also consider contributing your extensions to the core skins, rather than write your own skin. Bear in mind that you could be creating an update and management issue. Anyway, ...

プロジェクトは、src/documentation/skins ディレクトリ（または${project.skins-dir}が指し示すどこか）にあなた自身のスキンを定義できます。デフォルトのサイトマップは特定のスキンのレイアウトを仮定します、そのため新しいスキンを作るもっとも簡単な方法は、既存の Forrest スキンをコピーすることです。たとえば、main/webapp/skins/pelt を、src/documentation/skins/my-fancy-skin にあるあなたのプロジェクトのエリアへとコピーし、forrest.properties へと project.skin=my-fancy-skin を追加します。

Projects can define their own skin in the src/documentation/skins directory (or wherever ${project.skins-dir} points). The default sitemap assumes a certain skin layout, so the easiest way to create a new skin is by copying an existing Forrest skin. For example, copy main/webapp/skins/pelt to your project area at src/documentation/skins/my-fancy-skin and add project.skin=my-fancy-skin to forrest.properties

もっとも面白い二つの関係する XSLT スタイルシートは、次のとおりです：

The two most interesting XSLT stylesheets involved are:

xslt/html/document-to-html.xsl
: このスタイルシートはそれぞれの Forrest xdocs XML ファイルへと適用され、より大きな HTML ページに組み込むために適切な HTML へとそれらを変換します。
This stylesheet is applied to individual Forrest xdocs XML files, and converts them to HTML suitable for embedding in a larger HTML page.

xslt/html/site-to-xhtml.xsl
: このスタイルシートは、他のスタイルシートによって作られる中間的な'site'構造から最終的なHTML ファイルを生成します。それは一般的なレイアウトを定義し、ヘッダーとフッターを追加します。
This stylesheet generates the final HTML file from an intermediate 'site' structure produced by the other stylesheets. It defines the general layout, and adds the header and footer.

一般的にスキンの間に多くの共通性があります。XSLT は、一つの XSLT がもう一つを拡張できることにより、'インポート'機能を提供します。Forrest は一般的に共通ベースから'インポート'します。

Typically there is a lot of commonality between skins. XSLT provides an 'import' mechanism whereby one XSLT can extend another. Forrest XSLTs typically 'import' from a common base:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:import href="lm://transform.skin.common.html.document-to-html"/>

  ... overrides of default templates ...

</xsl:stylesheet>
```

インポート状態において *lm* プロトコルが使われることに注意してください。*lm* プロトコルは、指示されたスタイルシートの場所を解決するために、locationmap を使うことを Forrest に指示します。もしあなたがサイトマップを通してこの呼び出しを追跡するならば、あなたは `main/webapp/locationmap-transforms.xml` の以下のセクションを見つけるでしょう。

Notice the use of the *lm* protocol in the import statement. The *lm* protocol directs Forrest to use the locationmap to resolve the location of the indicated stylesheet. If you trace this call through the sitemap, you will find the following section of `main/webapp/locationmap-transforms.xml`:

```
<match pattern="transform.skin.*.*.*">
  <select>
    <location src="{properties:skins-dir}/{1}/xslt/{2}/{3}.xsl"/>
    <location src="{forrest:forrest.context}/skins/{1}/xslt/{2}/{3}.xsl"/>
  </select>
</match>
```

これは、まず位置情報が（あなたの `forrest.properties` ファイルの`${project.skins-dir}`プロパティによる）あなたのプロジェクト空間をチェックすることを意味します。そして、もしファイルがそこに見つからなければ、あなたの Forrest のインストールをチェックします。

This means that the locationmap first checks your project space (according to the `${project.skins-dir}` property of your `forrest.properties` file) and, if the file is not found there, it then checks in your installation of Forrest.

注意（Note）

カスタマイズしたスキンを作るとき、カスタマイズしたスキンをあなたのプロジェクトにコピーすることが、過去には必要でした。これはもはや本当ではありません。
It has been necessary in the past to copy the common skin to your project when creating a custom skin. This is no longer the case.

対話的な Forrest：あなたのドキュメントを開発するときにより早いターンアラウンド（Interactive Forrest: faster turnaround when developing your docs）

（Anakia のような）シンプルなツールと比較して、Cocoon のコマンドライン・モード（と、それゆえ Forrest コマンドライン・モード）は遅いです。dream list が記載しているように、Forrest は元々動的なサイトに対して使われることを意図していて、Cocoon クローラはミラーするために静的なスナップショットを作るためだけに使われます。このセクションは、"生の"Forrest webapp インスタンスを使うことによって、Forrest ベースのドキュメント開発は相当するツールを用いるよりも、より早く、より簡単にできます。

In comparison to simpler tools (like Anakia) the Cocoon command-line mode (and hence Forrest command-line mode) is slow. As the dream list notes, Forrest was originally intended to be used for dynamic sites, and the Cocoon crawler used only to create static snapshots for mirroring. This section describes how, by using a "live" Forrest webapp instance, the Forrest-based documentation development can be faster and easier than with comparable tools.

webapp として実行する（Running as a webapp）

Forrest の組み込みの Jetty ウェブサーバを開始するために、プロジェクトルートで’`forrest run`’とタイプしてください。一度それを始めたら、あなたの Web サイトを表示する、http://localhost:8888/をあなたのブラウザに示して、それぞれのリンクが従うように要望に応じて描画されます。

Type ’`forrest run`’ in your project root to start Forrest's built-in Jetty web server. Once it has started, point your browser at http://localhost:8888/, which will show your website, rendered on demand as each link is followed.

（代わりに、もし既存のサーブレット・コンテナの中から Forrest を実行したいならば、`build/webapp/` でオープンな Web アプリケーションを構築するために `forrest webapp` とタイプしてください。

(Alternatively, if you wish to run Forrest from within an existing servlet container, type `forrest webapp` to build an open webapp in `build/webapp/`)

webapp を使う（Using the webapp）

あなたは、`build/webapp/content/xdocs` にある XML コンテンツを編集することができます、そしてブラウザで即座に変更を確認することができます。

You can now edit the XML content in `build/webapp/content/xdocs` and see the changes immediately in the browser.

Ant から Forrest を実行する（Invoking Forrest from Ant）

Ant は、Ant から Forrest を実行するために使うことができる<import>タスクを持ちます。すべてのターゲットとプロパティは、インポートされ、あなたのプロジェクト構築に使われることができます。これは簡単な例です：

Ant has an <import> task which can be used to invoke Forrest from Ant. All targets and properties are imported and can be used in your project build. Here is a simple example:

```
<project name="myproject" default="hello">
     <!-- FORREST_HOME must be set as an environment variable -->
     <property environment="env"/>
     <property name="forrest.home" value="${env.FORREST_HOME}"/>
     <import file="${env.FORREST_HOME}/main/forrest.build.xml"/>

     <!-- 'site' is a target imported from forrest.build.xml -->
     <target name="post-build" depends="site">
       <echo>something here</echo>
     </target>
</project>
```

警告（Warning）

Ant のターゲット名のクラッシュを引き起こす FOR-145 記事を参照ください。
See issue FOR-145 which causes clashes of Ant target names.

警告（Warning）

Forrest 0.7 にはプラグインダウンロード機能にバグがあります。それは ANT から Forrest を呼び出すときに、あなたのプラグインが正しくインストールされるのを妨げるかもしれません。あなたは、forrest.properties においてプラグインに対してバージョン番号が定義されていることを確実にするか、要求されるプラグインを手動でインストールするかどちらかによって、このバグを取り巻いてうまくいくことができます。
There is a bug in the plugin download mechanism in Forrest 0.7 that may prevent your plugins being installed correctly when calling Forrest from ANT. You can work around this bug by either ensuring a version number is defined for the plugin in forrest.properties or by manually installing the required plugins.

あなたは Forrest が動くためにあなた自身のバージョンの Ant を使うことができるので、補助的なカタログ・エントリー・リゾルバを提供する必要があります：

Because you are using your own version of Ant to do Forrest's work, you will need to provide the supporting catalog entity resolver:

```
'cp forrest/lib/core/xml-commons-resolver-1.1.jar $ANT_HOME/lib'
```

注意：上述されたテクニックは Ant 1.6+を要求します。さもなくば、<import>タスクはあなたが使うことはできないでしょう。Forrest は最新版の Ant を同梱しています、そのため、あなたはこのようにあなたのプロジェクトを呼び出します：'forrest -f myproject.xml'。これは'forrest'コマンドを実行しません。それは単に Forrest の Ant とあなたのビルドファイルを実行するためのリゾルバを使います。

Note: The technique described above requires Ant 1.6+ otherwise the <import> task will not be available for you to use. Forrest bundles the latest version of Ant, so you can invoke your project like this: 'forrest -f myproject.xml'. This will not run the 'forrest' command. It will just use Forrest's Ant and resolver to execute your buildfile.

もう一つのオプションは、Krysalis プロジェクトの Antworks Importer から Forrest Antlet を使うことです。

Another option is to use the Forrest Antlet from the Krysalis Project's Antworks Importer.

Forrestbot は、ソースを取得し、ビルドし、配備し、注意するために、workstages を提供します。これは自動ビルドに非常に便利です。あなたは Forrestbot を使うことを考えたいと思うかもしれません。

The Forrestbot provides workstages to get source, build, deploy, and notify. This is very useful for automating builds; you may want to consider using the Forrestbot.

SVN: $Revision: 887646 $ $Date: 2009-12-06 19:23:52 +1100 (Sun, 06 Dec 2009) $