Building Offline Websites with SiteMesh 3

A new feature in SiteMesh 3 is being able to apply decorators to content as an offline task, typically as part of a build step.

If both your content and your decorators are static, this offers a few benefits:

- Allows final content to be distributed as pre-generated folder. Useful for including documentation with products without having to include a full Servlet compliant web-server.
- Cuts down server loads - most web-server architectures are optimized for serving static files.
- Provides more flexibility on where you can host content.

It's also possible to reuse decorators and configuration between a web application that generates decorated content on the fly, and offline generated files.

Usage

There are a few different approaches to invoking the SiteMesh offline generator:

1. **Command line interface**
2. Apache Ant **task**
3. **Java API** (this can be embedded in applications, or used from higher level languages such as JRuby, Groovy or Scala)

Each of these can have the mappings of the decorators passed directly to them, or load from the SiteMesh configuration file.

Use the approach that suits your project.

Command line interface

You can invoke the command line interface by running the executable sitemesh.jar. It requires Java 5 but no other dependencies.

Invoking on it's own will output a detailed help message:

```
java -jar sitemesh-3.x.jar
```

Arguments

The following arguments need to be passed to the command line:

-src	Required	Path to source directory, containing content and decorators
-dest	Required	Path to destination directory, where decorated content will be written
-dest	Required	Path to destination directory, where decorated content will be written
-config	One of these	Path to configuration file
-decoratorMapping	TODO	
FILE1 FILE2 FILE3...	Required	List of content files to apply decorators to. These must be relative to the src directory

Example

```
java -jar sitemesh-3.x.jar -src project/src -config project/sitemesh.xml
-dest project/build index.html page1.html page2.html
```

Ant Task

Overview

The sitemesh.jar comes prepackaged with a custom Ant task that can be used for offline processing. In the following examples, we're going to look at how to install and use the SiteMeshTask to generate static content in the offline mode.

SiteMeshTask

The SiteMeshTask defines the following attributes:

1. destDir - The output folder in which all of the decorated files will be placed.
2. config - The location of the SiteMesh configuration file.
3. srcdir - The source directory which contains all of the files to be decorated.
4. includes - An Ant style filter of what files to include.
5. excludes - An Ant style filter of what files to exclude.

The SiteMeshTask can also accept the following child nodes:

1. fileset - The standard Ant FileSet that will use the srcdir of the sitemesh node.
2. sitemeshfileset - A custom SiteMesh FileSet that supports an additional attribute called "decorator" which can be used to associate a decorator with the given FileSet.

Getting started!

The below provides a high level outline of what steps we're going to cover in this section.

1. Create a SiteMesh configuration file.
2. Register the SiteMeshTask with Ant
3. Define the **<sitemesh/>** node.
4. Execute Ant

1. Creating the SiteMesh configuration file

The SiteMeshTask can be given a configuration file to tell SiteMesh how to decorate files. The power of this feature is that the configuration is then externalized from the **build.xml** file. In our first two examples, we're going to show how to use this form of the SiteMeshTask. Below is a very simple SiteMesh configuration file that applies the main.html decorator to all pages.

```
<sitemesh>
  <mapping path="/*" decorator="/decorators/main.html"/>
</sitemesh>
```

2. Registering SiteMeshTask with Ant

In order to use SiteMesh from within Ant, the first thing you will need to do is register the SiteMeshTask with Ant using the following **<taskdef/>** de claration.

```
<project name="my-ant-project">

  <taskdef name="sitemesh"
           classname="org.sitemesh.ant.SiteMeshTask"
           classpath="path/to/sitemesh-3.x.jar"/>

  ...

</project>
```

For more information on registering custom tasks within Ant, please see Writing Ant Tasks.

3. Define the <sitemesh/> node.

Now that we've created the SiteMesh configuration file and registered the SiteMeshTask with Ant, it's time to start using SiteMesh within our tasks.

Using the sitemesh task without a fileset.

Let's look at how to provide **<sitemesh/>** with a configuration file and tell it what directories to include or exclude.

In this example, we're going to process all of the files stored in "project/src" and place the decorated files into "project/build".

```
<project name="my-ant-project">

  <target name="my-target">
    <sitemesh srcdir="project/src"
              config="project/sitemesh.xml"
              destdir="project/build"
              includes="*/.html"
              excludes="decorators/*"/>
  </target>

</project>
```

The benefit to the above is that all configuration is externalized from the the **build.xml** file.

Using the sitemesh task with the sitemeshfileset

In the below example, multiple source folders are used by provided a sitemeshfileset. This provides greater control over what folders should be included or excluded, but still leverage a common destination folder and configuration file.

```
<project name="my-ant-project">


  <target name="my-target">

    <sitemesh destdir="site/documentation"
              config="config/sitemesh.xml">

      <sitemeshfileset dir="documentation">
        <include name="*/.html"/>
        <exclude name="private/*"/>
      </sitemeshfileset>

      <sitemeshfileset dir="presentation">
        <include name="*/.html"/>
      </sitemeshfileset>

    </sitemesh>
  </target>

</project>
```

Using the sitemesh task with the sitemeshfileset with an associated decorator

In our finaly example, a decorator will be used on each sitemeshfileset.

```
<project name="my-ant-project">

  <target name="my-target">
    <sitemesh destdir="site/documentation">
      <sitemeshfileset dir="documentation"
                       decorator="decorators/private.html">
        <include name="private/*.html"/>
      </sitemeshfileset>
    </sitemesh>
  </target>

</project>
```