

# The Forrest Include Extension

## Table of contents

1 Introduction.....	2
2 Download and installation.....	2
3 Configuration of your XML editor.....	3
4 Creating a New Document.....	3
5 The Include Element.....	4
6 Hyperlinking.....	6
7 Frequently Asked Questions.....	6
8 Feedback.....	7

## 1. Introduction

It is sometime convenient to split up content in several files and include content instead of referring to it using a hyperlink. There are several reasons why this is useful.

- Sometimes a document gets too big and splitting it up provides more overview and allows multiple people to work on different parts of the document
- Sometimes a piece of content must be reused in several other documents but you want to have the content inline and not through a hyperlink.
- Sometimes a configuration, properties, XML file, or a longer code snippet must be included. In this case, pasting the content in the document would make it less readable.

Forrest (0.7) already has support for XInclude but unfortunately, the include mechanism is a bit limited. In particular,

- Adding XInclude elements to your document breaks the DTD so that you can no longer validate your documents.
- XInclude requires either full path names of files or files relative to the root of the site. Unfortunately, the existing link rewriting in Forrest generates links relative to the current file.
- Typically you do not want to include the full contents of another Xdoc document but only the content of the body tag. Using Xinclude directly requires an author to write non-trivial XML with xpointer expressions.

Therefore, an extension for includes is needed, at least until Forrest supports it natively. The extension is nothing more than a usability improvement: for its implementation it uses the XInclude mechanism provided by Forrest. The extension consists of:

- A custom DTD to use for your documents which extends the Forrest Xdoc format version 2.0 by adding an include element.
- Custom stylesheets to perform link rewriting especially for XInclude.
- Custom configuration in the sitemap to perform the appropriate actions for the new document type.

Beyond this, no additional work is required. Just change the DTD declaration of your document when you want to use the include extension. Existing documents will continue to work.

## 2. Download and installation

File	Installation location	Purpose
catalog.xcat	project.schema-dir	XML catalog for the extensions

CatalogManager.properties	project.classes-dir	Configuration to tell Forrest where to find the catalog file.
includeExtension-v1.dtd	project.schema-dir	Document type declaration for Xdoc with includes.
absolutize-includes.xsl	project.stylesheets-dir	Stylesheet to absolutize links to links relative to the site root.
add-cocoon-prefix.xsl	project.stylesheets-dir	Stylesheet to add a <code>cocoon:/</code> prefix to the included paths so that cocoon can find them.
dotdots.xsl	project.stylesheets-dir	Stylesheet to transform a path into a sequence of <code>'../'</code> to go back from the given path to the root of the site. This script is actually copied from Forrest itself because I could not figure out how to include it otherwise.
relativize-links.xsl	project.stylesheets-dir	Stylesheet to process relative links in included file to make them relative to the current file. Also, it relativizes absolute links (those starting with <code>'/'</code> ) into relative ones.
sitemap.xmap	project.sitemap-dir	Sitemap file. Comments in the file indicate which parts you need to copy in your own sitemap file.

### 3. Configuration of your XML editor

To use the extension, it is important to configure your XML editor with the catalog file of Forrest and of the catalog file of our extension. A catalog file tells your XML editor where the DTDs for the used documents can be found so that internet access is not required for validation. The following catalog files must be configured in your XML editor for the extensions:

- **The Forrest catalog:** You can find this at `main/webapp/resources/schema/catalog.forrest.xcat`
- **The extensions catalog:** You can find this at `resources/schema/catalog.xcat`

### 4. Creating a New Document

To create a new document, start from the following template:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE document
  PUBLIC "-//WAMBLEE.ORG//ENTITIES Include Extension for Xdoc V1//EN"
  "includeExtension-v1.dtd">

<document>
  <header>
    <title>Your new document</title>
  </header>
  <body>
    <section id="introduction">
      <title>Introduction</title>
      <p>Your text.</p>
    </section>
  </body>
</document>
```

#### Note:

Always validate your document against the DTD. If the document is invalid, Forrest is almost guaranteed to produce strange results.

## 5. The Include Element

The syntax of the include element is as follows:

```
<include file="site:REF" parse="text|xml"
pointer="XPOINTER_EXPR"/>
```

The `file` attribute specifies the file to be included. For the value of `file` attribute, the following types of references can be used:

- **site:a/b/c**: A reference using a URL defined in the `site.xml`. The site scheme is currently the only supported one but other schemes supported by Forrest can be easily added.
- **/path/to/resource/myfile.myextension**: A reference relative to the root of the site.
- **relative/path/to/resource/myfile.myextension**: A relative path. The simplest version of this is simply the file name in case the file resides in the same directory.

In any case. The `parse` attribute describes how to include the source. To include the content of another Xdoc document, you would typically use the value `xml` for this attribute or omit it. The `pointer` attribute finally allows you to specify a part of the included XML document using `xpointer` syntax. The default `xpointer` expression used is `xpointer(//body/*)` which means that the content of the `body` tag of the XML

document is included. Beware however that xpointer support does not appear to be that mature right now in Forrest.

To include another Xdoc document you would use

```
<include file="site:MYFILE"/>
```

This includes the content from within the `body` tag of the other document.

To include another file as a code example you would use

```
<source><include parse="text" file="site:MYFILE"/></source>
```

**Note:**

It is essential that no spaces or newlines are put between the source element and the include element otherwise the first line of your document will not be properly indented.

**Note:**

When including file with `parse="text"`, the `xpointer` attribute is ignored so that always the entire document will be included.

For downwards compatibility with Xdoc documents, the include extension allows `.html` extensions to imply Xdoc files. It replaces the `.html` extension by `.xml` to look for the corresponding Xdoc file. This allows standard Xdoc files and the files using the include extension to share the same definitions in the `site.xml` file.

To include raw XML file without any processing use the `.rawxml` extension. When Forrest generates the site, it will then replace the extension by `.xml`. So if your `site.xml` refers to a file `mydir/myfile.rawxml` then Forrest will look for a file at `mydir/myfile.xml`

**Note:**

To include raw XML files in your document, use `parse="text"` in the `include` element and replace the `.xml` extension by `.rawxml` to prevent Forrest from transforming the XML. Also, if the included document contains a schema or DTD declaration, then the parser must be able to validate your document. To do this, make the DTD or schema available in `project.schema-dir` and declare it in the catalog file.

The `include` functionality is *transitive*, that is, it is perfectly ok to include another document which includes another document etc. (who tries out an infinite recursion?).

**Warning:**

The use of the `xpointer` attribute is experimental, use at your own risk/frustration.

## 6. Hyperlinking

Hyperlinks in included documents also work. You can use the same types of URLs as with the include element, namely `site:` URLs, relative URLs, and URLs relative to the site root. The last type of URL is an extension with respect to Forrest but can be convenient to include a document whose path is known but where you don't want to define the document in the `site.xml`.

## 7. Frequently Asked Questions

- *My Xdoc document contains an include but it looks like this include is not processed at all and I also get no error message. What is wrong?*

Make sure that the document type declaration of your xdoc file refers to the correct DTD (see above), otherwise includes will not be processed.

- *My document gives some obscure error and I don't know what to do.*

Try these steps in order:

- Validate your document. If your document isn't valid you shouldn't expect Forrest to do anything with it.
- Replace the `.html` extension by `.rawxml` and try to retrieve the document. This retrieves the original document as XML without any processing done by Forrest.
- Replace the `.html` extension by `.xml` and try to retrieve the document. This retrieves the XML document converted to the internal document format that Forrest uses.
- *I am using 'forrest run' but the changes I make do not show up.*

It can occur that changes to included documents do not become visible. This is because forrest sometimes caches HTML pages. If you request the same page as the XML source then you will see the changes. An alternative is to view the included file you are editing on its own and not as part of a document in which it is included. Of course, you can also restart forrest.

- *Cannot get input stream for `cocoon:/a/b/c/myfile.ext`*

If you get this error, you need to configure a rule in your `sitemap.xmap` for retrieving this type of file. For example:

```
<map:match pattern="**.ext">
  <map:read src="{project:content.xdocs}{1}.txt"/>
</map:match>
```

I do not understand yet why this is occurring but my understanding of cocoon and Forrest is growing.

## **8. Feedback**

I am always interested in getting feedback from you. Don't hesitate to drop me a mail.