

連載

CocoonによるWebサイト構築テクニック

最終回：XML データベースとの連携

Cocoon を使った Web サイト構築について 2 年弱にわたって連載を行ってきた。特に大量のコンテンツを保持しながら、同時に管理しやすいサイトを構築するのに Cocoon が役立つことをご理解いただけたことと思う。最終回となる今回は XML データベースとの連携について取り上げよう。

本連載を通じて、Cocoon は XML データを活用した Web サイトを構築するための環境であり、XML データを整理して Web 上に公開したい場合に有用であることを説明してきた。さて、XML データの量が多くなるにつれて、データの検索を行って、見つけたデータを表示することも必要になってくるだろう。

前々回および前回は、リレーショナルデータベースと Cocoon を連携させる方法について説明した。もともと XML で記述されていたデータであっても、表形式に整理することができればリレーショナルデータベースに格納することによって検索の性能を上げることができる。それを Cocoon と連携させれば、データの効率的な管理が可能になるわけだ。

このように、リレーショナルデータベースは非常に有用な道具だが、XML で記述されるデータの中にはリレーショナルデータベースに格納するには不便なものもある。たと

えば、長い文章を含むデータや、データ構造の中に不規則な繰り返しが見られる場合、表形式のデータへのマッピングが困難になることが少なくない。そのような場合には XML データベースを活用することを検討できる。XML データベースを使用すると、格納した多量の XML データの中から必要なデータを取り出して使用できるようになる。

XML データベース「Xindice」

Cocoon には、Apache XML プロジェクトで開発 / 保守がなされている XML データベース「Xindice」(<http://xml.apache.org/xindice/>) が添付されており、Cocoon をセットアップしただけで Xindice と連携した動作ができるようになっている¹。Cocoon 2.1.7 には Xindice 1.1b4 が添付されており、XPath を使った検索と XUpdate を使った更新をサポートしている。

Xindice を含め、多くの XML データベースはファイルシステムのディレクトリとよく似た構造の「コレクション」という単位に XML ドキュメントを整理して格納する (図 1)。トップレベルのコレクションには「db」という名前が付けられており、「/db」と表記する。各コレクションにはドキュメントを入れるほか、下位のコレクションを作成することもできる。なお、トップレベルコレクション「/db」には直接ドキュメントを格納することはできない。

XML データベースに対する操作は、基本的にコレクションの単位で行われる。コレクションを指定してドキュメントを取り出したり、コレクションを指定して XPath を使っ

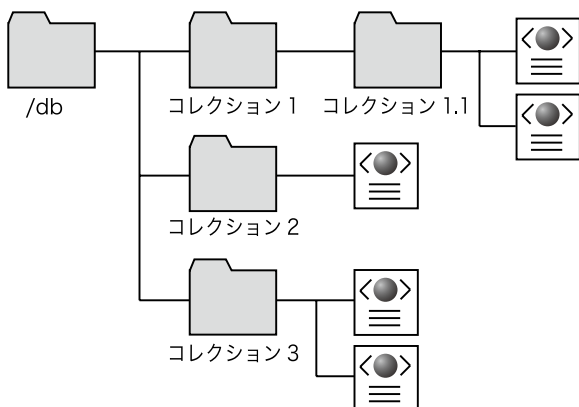


図 1：XML データベースは XML ドキュメントをコレクション単位に分けて格納する

1. Xindice のサイトによると、「Xindice」はイタリア語風に「zeen-dee-chay」と発音すると説明されている。「ジンディーチェ」と読めばいいだろうか？ もっとも、同サイトには発音が正しくなくてもスペルが合っていればよいとも書かれているので発音についてはさほど気にしなくていいだろう。

表 1: Xindice 管理ツールを使用するための環境変数

環境変数	設定する値
XINDICE_HOME	ダウンロードして展開した Xindice ディレクトリ
XINDICE_DB_HOME	Cocoon ディレクトリ下のサブディレクトリ WEB-INF
XINDICE_CONFIG	XINDICE_DB_HOME に設定した WEB-INF ディレクトリにある xindice.xml

たクエリーを行うのである。XPath を使ったクエリーでは、指定したコレクション中のそれぞれのドキュメントに対して同じ検索が実行され、それぞれの検索結果を合わせたものが結果として返却される。

Xindice 管理ツールの使い方

Cocoon には Xindice を管理するためのツールは添付されていない。それで、Xindice を使い始める前に管理ツールを入手しておくとういだろう。

以下において、オリジナルの Xindice に添付されたコマンドラインベースの管理ツール「xindice.bat」を Windows 環境で使う方法を説明する²。

1. Xindice のホームページ <http://xml.apache.org/xindice> から「Download」をクリックし、「Binary (jar)」と示されたファイルをダウンロードして展開する。
2. 表 1 に示す環境変数を設定する。たとえば、C:\Java\jakarta-tomcat-5.5.9 にセットアップした Tomcat に Cocoon を配備し、ダウンロードした xml-xindice-1.1b4-jar.zip を C:\Java に展開した場合、リスト 1 のように設定することになる。

これで、XINDICE_HOME\bin に置かれた管理ツール xindice.bat を使って、Cocoon と統合された Xindice の管理が可能になった。

2. Unix 用にはシェルスクリプト xindice.sh が備えられている。また、Xindice webadmin(<http://metzner.org/projects/webadmin/>) など、Xindice の管理を行うための GUI ベースのツールもリリースされている。どのようなツールを使用する場合にも Cocoon と統合された Xindice を管理するには設定の変更が必要となる。興味のある方は試してみてください。

【リスト 1: Xindice 管理ツールを使用するための環境変数の設定例】

```
set XINDICE_HOME=C:\Java\xindice-1.1b4
set XINDICE_DB_HOME=C:\Java\jakarta-tomcat-5.5.9\webapps\cocoon\WEB-INF
set XINDICE_CONFIG=C:\Java\jakarta-tomcat-5.5.9\webapps\cocoon\WEB-INF\xindice.xml
```

管理ツールを使って行うおもな作業には以下のものがある。

- **コレクションのリスト**: 指定したコレクションに含まれるコレクションのリストを表示する

```
xindice lc -c コレクション URL
```

- **コレクションの作成**: 指定した親コレクションに新しいコレクションを作成する

```
xindice ac -c 親コレクション URL -n 新コレクション名
```

- **コレクションの削除**: 指定した親コレクション内のコレクションを削除する

```
xindice dc -c 親コレクション URL -n 削除するコレクション名
```

- **ドキュメントのリスト**: 指定したコレクションに含まれるドキュメントのリストを表示する

```
xindice ld -c コレクション URL
```

- **ドキュメントの追加**: 指定したコレクションにファイル名で指定したドキュメントを格納する。データベース内では指定したドキュメントキーで識別される³。

```
xindice ad -c コレクション URL -f ファイル名 -n ドキュメントキー
```

3. ドキュメントを追加するときに -n オプションでドキュメントキーを指定しなかった場合は、「7100a7c0664de47400000103c4dfaf7b」といったドキュメントキーが自動生成される。わかりやすくするため、必ずドキュメントキーを指定するようお勧めする。

■ **ドキュメントの削除**: 指定したコレクションからドキュメントキーで指定したドキュメントを削除する

```
xindice dd -c コレクションURL -n ドキュメントキー
```

■ **ドキュメントの取得**: 指定したコレクション内にあるドキュメントキーで指定したドキュメントを取り出してファイルに保存する

```
xindice rd -c コレクションURL -n ドキュメントキー -f 出力ファイル名
```

その他のコマンドや引数の詳細については、Xindice のホームページから「Commandline Tool Guide」をクリックして表示される情報を参照していただきたい。

ここまでで説明したとおり、Xindice 管理ツールのすべてのコマンドで、`-c` オプションを使って処理の対象となるコレクションの URL を指定する必要がある。この URL の指定には、以下のような「XML:DB URL」と呼ばれる記法を使用する。

```
xml:db: データベース ID:// ホスト名または IP アドレス / コレクション名
```

データベース ID とは、XML データベースの種類ごとに決まっている文字列であり、Xindice では「xindice」と「xindice-embed」の 2 種類がサポートされている。

「xindice」はネットワーク経由でホスト名または IP アドレスで指定したマシン上で動作する Xindice にアクセスする⁴。

-
- Cocoon と統合された Xindice にアクセスする場合、Xindice 管理ツールからデータベース ID 「xindice」を使った接続はうまくいかないようだ。たとえば、コレクション「xml:db:xindice://localhost:8080/db」を指定すると、管理ツールは `http://localhost:8080/xindice/` に公開された Xindice のコレクション「db」に接続しようとするが、Cocoon と統合された Xindice はその位置にはないため接続に失敗する。したがって、Xindice 管理ツールではデータベース ID 「xindice-embed」で接続する必要があるだろう。
 - 通常、XML データベースの管理作業を行うには、まず当該データベースを稼働させる必要があるが、「xindice-embed」で接続する場合に限ってはデータベースが稼働していなくても（具体的には Tomcat などのサーブレットエンジンを立ち上げていなくても）管理作業を行うことができる。

一方、「xindice-embed」を使うと、同一マシン上の Xindice にアクセスする⁵。この場合、ホスト名または IP アドレスには何も書かない。同一マシンで動作する Xindice のコレクション「/db」にアクセスするには、以下の XML:DB URL を使う。

```
xml:db:xindice-embed:///db
```

具体的には、リスト 2 のように実行すれば Xindice 管理ツールを使って Xindice のコレクション「/db」に含まれるコレクションのリストを得ることができる。Cocoon と統合した Xindice では、初期状態では Xindice データベース自身が使用するコレクション「system」と「meta」の 2 つが存在し、Cocoon に添付されたサンプルを動作させると「cocoon」というコレクションが生成される（Cocoon に添付されたサンプルについては 39 ページのコラムを参照）。

管理ツールを使ってサンプルデータを登録する

それでは、実際に管理ツールを使ってサンプルデータを Xindice に格納してみよう⁶。以下の手順で 2 つのサンプルデータを登録する。

- 今回のサンプルデータを格納するためのコレクション「accessories」を作成する。
- 1 つめのサンプルデータ「magupi.xml」をドキュメントキー「magupi」で登録する。
- 2 つめのサンプルデータ「ring.xml」をドキュメントキー「ring」で登録する。
- コレクション「accessories」に含まれるドキュメントのリストを表示する。ドキュメント「magupi」と「ring」が登録されていることが確認できる。

この手順を実行するコマンドラインをリスト 3 に、登録するサンプルデータをリスト 4 とリスト 5 に示す。

-
- 今回のサンプル作成にあたっては、株式会社サン宝石 (<http://www.sunhoseki.co.jp/>) の商品リストを参考にさせていただいた。

【リスト2：Xindiceのコレクション/dbに含まれるコレクションをリストする(強調表示したコマンドを入力)】

```

C:¥Java¥xindice-1.1b4¥bin>xindice lc -c xmldb:xindice-embed:///db
trying to register database
[INFO] DatabaseImpl - -Specified configuration file: 'C:¥Java¥jakarta-tomcat-5.5
.9¥webapps¥cocoon¥WEB-INF¥xindice.xml'
[INFO] Database - -Database points to C:¥Java¥jakarta-tomcat-5.5.9¥webapps¥cocoo
n¥WEB-INF¥db

        cocoon
        system
        meta

Total collections: 3

C:¥Java¥xindice-1.1b4¥bin>

```

【リスト3：サンプルデータの登録】

```

C:¥Java¥xindice-1.1b4¥bin>xindice ac -c xmldb:xindice-embed:///db -n accessories ①
trying to register database
[INFO] DatabaseImpl - -Specified configuration file: 'C:¥Java¥jakarta-tomcat-5.5
.9¥webapps¥cocoon¥WEB-INF¥xindice.xml'
(中略)
[INFO] CollectionManager - -Created a new collection named 'accessories'
Created : xmldb:xindice-embed:///db/accessories

C:¥Java¥xindice-1.1b4¥bin>xindice ad -c xmldb:xindice-embed:///db/accessories -f ②
c:¥temp¥magupi.xml -n magupi
trying to register database
(中略)
Added document xmldb:xindice-embed:///db/accessories/magupi

C:¥Java¥xindice-1.1b4¥bin>xindice ad -c xmldb:xindice-embed:///db/accessories -f ③
c:¥temp¥ring.xml -n ring
trying to register database
(中略)
Added document xmldb:xindice-embed:///db/accessories/ring

C:¥Java¥xindice-1.1b4¥bin>xindice ld -c xmldb:xindice-embed:///db/accessories ④
trying to register database
[INFO] DatabaseImpl - -Specified configuration file: 'C:¥Java¥jakarta-tomcat-5.5
.9¥webapps¥cocoon¥WEB-INF¥xindice.xml'
[INFO] Database - -Database points to C:¥Java¥jakarta-tomcat-5.5.9¥webapps¥cocoo
n¥WEB-INF¥db

        magupi
        ring

Total documents: 2

C:¥Java¥xindice-1.1b4¥bin>

```

【リスト 4: マグネットピアスの商品リスト (magupi.xml)】

```
<?xml version="1.0"?>
<productList category=" マグネットピアス ">
<product>
<name> マリンブルー丸プレートマグピ </name>
<id>35279</id>
<price>125</price>
<desc> 全長約 3.5cm</desc>
</product>
<product>
<name> 穴あきフラワーマグピ </name>
<id>35329</id>
<price>125</price>
<desc> 全長約 4.5cm</desc>
</product>
<product>
<name> 変形ハートマグピ </name>
<id>43518</id>
<price>180</price>
<desc> ハート約 7 × 6mm</desc>
</product>
<product>
<name> 6 色スターセットマグピ </name>
<id>43731</id>
<price>170</price>
<desc> 6 色 6 ペアセット 星約 6mm</desc>
</product>
<product>
<name> ハイビスカスマグネットピアス </name>
<id>43878</id>
<price>170</price>
<desc> ハイビスカス約 1.2cm</desc>
</product>
</productList>
```

【リスト 5: リングの商品リスト (ring.xml)】

```
<?xml version="1.0"?>
<productList category=" リング ">
<product>
<name> 赤いリングいっぱいリング </name>
<id>21982</id>
<price>180</price>
<desc> 幅約 6mm</desc>
<size>10</size><size>13</size>
<size>16</size>
```

XML データベースからドキュメント単位でデータを取り出す

ここまでで XML データベースの準備ができたので、いよいよ Cocoon から XML データベースにアクセスしてデータを取り出してみることにしよう。まずは、登録したドキュメントをそのまま取り出し、整形してクライアントに返す方法を紹介する。

Cocoon ディレクトリの下にサブディレクトリ「xmldbSample」を作成して、そこに XSLT スタイルシート(リスト 6) とサイトマップファイル(リスト 7) を配置しよう。

XML データベースにアクセスするために新しいジェネレータを使う必要はなく、ファイルから読み込むのと同じ

```
</product>
<product>
<name> 赤バラリング </name>
<id>24271</id>
<price>200</price>
<desc> バラ約 1cm</desc>
<size>13</size><size>16</size>
</product>
<product>
<name> 二連風リング </name>
<id>24292</id>
<price>105</price>
<desc> 幅最大約 1cm</desc>
<size>10</size><size>13</size>
<size>16</size>
</product>
<product>
<name> 石付き赤ハートのプラリング </name>
<id>24304</id>
<price>100</price>
<desc> 幅最大約 1.5cm</desc>
<size>13</size><size>16</size>
</product>
<product>
<name> 黄色系フルーツリング </name>
<id>24313</id>
<price>200</price>
<desc> チャーム全長約 2.8cm</desc>
<size>10</size><size>16</size>
</product>
</productList>
```

↗

ようにしてアクセスできる。CocoonはXMLデータベースにアクセスするために「擬似プロトコル (pseudo protocol)」と呼ばれる機能を備えており、map:generate 要素の src 属性に XML:DB URL を使った指定を行うだけで XML データベースからのデータの取得ができるのだ。

【リスト6：XSLT スタイルシート (productList.xslt)】

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

<xsl:template match="/">
<html>
<head>
<title> アクセサリー商品リスト </title>
<style>
table { text-align : center }
tr.rowH { background : green; color :
white }
tr { background : #CCFFCC }
span.id { font-size : smaller }
</style>
</head>
<body>
<h1> 商品リスト </h1>
<table cellpadding="5">
<tr class="rowH">
<th> 商品名 <br/><span class="id"> 商品番号
</span></th>
<th> 説明 </th>
<th> サイズ </th>
<th> 価格 (円) </th>
</tr>
<xsl:apply-templates/>
</table>
</body>
</html>
</xsl:template>

<xsl:template match="product">
<tr>
<td><xsl:value-of select="name"/><br/>
<span class="id">No.<xsl:value-of
select="id"/></span></td>
<td><xsl:value-of select="desc"/></td>
<td><xsl:if test="size">
<xsl:for-each select="size">
```

```
<xsl:if test="position() != 1">
<xsl:text>, </xsl:text></xsl:if>
<xsl:value-of select="."/>
</xsl:for-each>
</xsl:if></td>
<td><xsl:value-of select="price"/></td>
</tr>
</xsl:template>

</xsl:stylesheet>
```

【リスト7：サイトマップファイル (sitemap.xmap)】

```
<?xml version="1.0" encoding="UTF-8"?>
<map:sitemap xmlns:map="http://apache.
org/cocoon/sitemap/1.0">

<map:pipelines>
  <map:pipeline>

    <map:match pattern="*.html">
      <map:generate src="xml:db:xindex-
embed:///db/accessories/{1}"/>
      <map:transform src="productList.
xslt"/>
      <map:serialize/> ①
    </map:match>

    <map:match pattern="priceUnder150.
xml">
      <map:generate src="xml:db:xindex-
embed:///db/accessories/#!/product[price
&lt; 150]"/>
      <map:serialize type="xml"/> ②
    </map:match>

    <map:match pattern="priceUnder150">
      <map:generate src="xml:db:xindex-
embed:///db/accessories/#!/product[price
&lt; 150]"/>
      <map:transform src="productList.
xslt"/>
      <map:serialize/> ③
    </map:match>

  </map:pipeline>
</map:pipelines>

</map:sitemap>
```

ドキュメントをそのまま取り出すための指定はコレクションの URL のあとにスラッシュを入れ、続けてドキュメントキーを記述する。

```
<map:generate src="コレクションURL/ドキュメントキー"/>
```

コレクションの URL は、管理ツールの `-c` オプションに渡した文字列と同様であり、「xmldb:データベースID://ホスト名またはIPアドレス/コレクション名」という形になる。たとえば、コレクション「/db/accessories」に登録したドキュメントキー「ring」の XML データは次のように記述すれば取得できる。

```
<map:generate src="xmldb:xindice-embed:///db/accessories/ring"/>
```

さらに、リスト 7-①のように記述すれば、ドキュメントキーの後ろに「.html」を付けた URL でアクセスしてきたクライアントに、XML データベースから取得した XML データを XSLT で整形した結果を送ることができる。ブラウザから URL 「http://localhost:8080/cocoon/xmldbSample/ring.html」にアクセスすれば、図 2 のようにリングの商品リストを整形した結果が得られる。



図 2: ドキュメントキー「ring」の XML データを取得して XSLT スタイルシートで整形した結果

XML データベースから XPath を使ってデータを取り出す

XML データベースを使うことの大きなメリットは、コレクション内の複数の XML データに対して同時に問い合わせを行い、その結果をまとめて取得できることだ。

単一のドキュメントを取り出す場合と同様に、XPath を使ったデータの問い合わせもサイトマップに指定するだけで実行可能だ。map:generate 要素の src 属性には、コレクションの URL のあとに「#」を入れ、続けて XPath を記述する。

```
<map:generate src="コレクションURL#XPath"/>
```

たとえば、コレクション「/db/accessories」に登録したドキュメントの中から価格 150 円未満の商品を取得したい場合、XPath 「//product[price < 150]」を使って次のように記述する。

```
<map:generate src="xmldb:xindice-embed:///db/accessories/#//product[price < 150]"/>
```

この XPath による問い合わせの結果は複数の product 要素から成るのだが、ジェネレータから返される XML データはそれらの問い合わせ結果を 1 つにまとめたものとなる。サイトマップファイルにリスト 7-②のように記述して、まず XML データそのものがどのような形かを確認しよう。ブラウザから「http://localhost:8080/cocoon/xmldbSample/priceUnder150.xml」にアクセスして取得した結果は図 3 のようになる。

取得結果全体は db:results 要素で囲まれ、個々の取得結果は db:result 要素 (要素名末尾の s の有無に注意) で囲まれる。取得結果の中には複数のドキュメントからのデータが含まれるが、取得元のドキュメントキーが db:result 要素の docid 属性、および取得したデータ (この場合は product 要素) の src:key 属性に示される。

このデータを XSLT スタイルシートで整形すると、図 4 の表示を得ることができる。マグネットピアスとリングの

7. XPath 内の不等号は < として記述しなければならないことに注意。

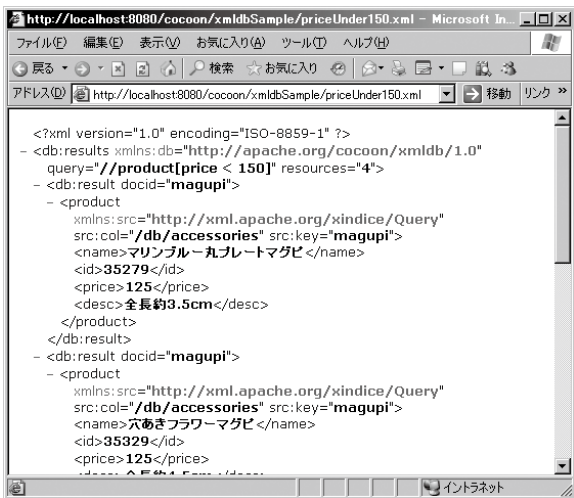


図3：XPath 「//product[price < 150]」に対する問い合わせの結果



図4：XPath 「//product[price < 150]」に対する問い合わせをXSLTスタイルシートで整形した結果

両方の商品リストから、150円未満の商品のみが取得できたことが確認できる。

まとめ

今回は、Cocoonに添付されたXMLデータベース「Xindice」にデータを格納して活用する方法を説明した。紙幅の関係で取り上げることができなかったが、XindiceはXMLデータを更新するための仕様「XUpdate(XML Update Language)」(<http://xmldb-org.sourceforge.net/xupdate/>)をサポートしており、Cocoonからデータベースに格納したXMLデータの更新を行うことができる。Cocoonに添付されたサンプルにXUpdateを使用したものがあるので、参考にさせていただくとよいだろう。

さらには、Xindiceではサポートされていない多くのXMLデータベースでは「XQuery」(<http://www.w3.org/XML/Query>)という仕様に基づいたより複雑なデータ検索をもサポートしており、Cocoonと連携することも可能だ。興味のある方はぜひ試していただきたい⁸。

(太田 純)

8. XQueryとCocoonを連携させる方法については、JavaWorld誌2005年5月号「初めてのXQueryプログラミング eXistとXML対応ツールの連携」(吉田晃伸、太田純、村上靖征)を参照。

コラム：Cocoonに添付されたXMLデータベースサンプル

Cocoonに含まれるXMLデータベースサンプルを動作させるには、<http://localhost:8080/cocoon/>にアクセスし、「samples」→「Block with samples」→「XML:DB Block」を順にクリックする。「XMLDB Block Samples」ページが開くので、まず「Init」をクリックしよう。これによってコレクション「cocoon」が作成され、XMLデータベースにアクセスする他の操作を行えるようになる。

サンプルのソースやサイトマップファイルはCocoonディレクトリの下にsamples%blocks%xmldbにあるので、参考にするとよいだろう。

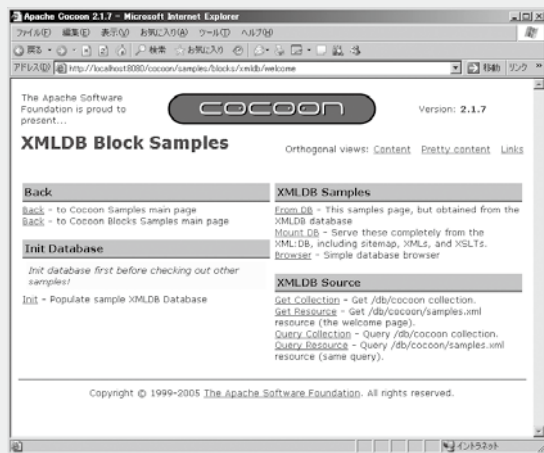


図5：Cocoonに添付されたXMLDB Block Samples