.NET Architecture Forum

XMLデータベースの検索速度を最速にする

新連載

郵便番号データ検索

PROJECT KySS
http://www.projectkyss.net/

Technology Tools

- ☑ Visual Basic .NET
- ☐ Visual C# .NET
- ☐ SQL Server 2000
- ☐ Oracle 9i
- ☐ Access 2002
- ☑ ASP.NET
- ☑ Internet Information Services
- ☑ Other:
  - .NET Framework 1.1
  - XML

Level

★ ★ ★ ★ ★

Samples

・この記事で取り上げたソースコードおよびサンプルプログラムは、付録CD-ROMの¥DOTNET¥NETARCH02ディレクトリに収録しています。

¥DOTNETMAGAZINE
本稿で紹介した以下のサンプルデータおよびコード
- ・PSOTALAREACLASS　PostalAreaクラス
- ・POSTALGUIDE_DOM　サンプル1
- ・POSTALGUIDE_DATAGRID_XPATH　サンプル2
- ・POSTALGUIDE_DATALIST　サンプル3
- ・POSTALGUIDE_DATAGRID_XPATH_ALLDATA　全国郵便番号表示コラムのサンプル
- ・POSTALGUIDE_DOM2　サンプル4
- ・POSTALGUIDE_DOM2_2　サンプル5
- ・POSTALGUIDE_DATAGRID_XPATH2　サンプル6
- ・POSTALGUIDE_LABEL_DOM3　サンプル7
- ・NETMENU2003　サンプルのメニュープロジェクト
- ・DATA　サンプルで利用するXMLファイル

はじめに

いまここに、15MBのXMLファイルがあります。これを使って検索処理を実装する仕事の依頼が舞い込みました。しかし、クライアントさんにはXMLデータベースを導入する予算がなく、XML文書をそのまま扱わざるをえません。みなさんなら、どうしますか？

50KBや500KB程度の小さいXMLファイルを処理するプログラムが、15MBもの巨大なXMLファイルにも通用するとは限りません。ファイルを読み込んだ途端にマシンが沈黙したり、コーヒーをおかわりしても処理が続行中……、ということになりかねません。

そこで、今回の企画です。どのようにプログラミングすれば、検索結果の表示というゴールに、1秒でも速くたどりつけるか試行錯誤してみることにします。みなさんも、知恵を絞って、パフォーマンス向上を目指し、自己ベストの更新にチャレンジしてみてください。

サンプルデータの概要～3種類の構造のXMLファイル

今回のテストには、全国の郵便番号を記録したXMLファイルを使います。データは同じで構造の異なる3種類のXMLファイルを用意しました[注1]。ファイルサイズは10MB～18MBです。

リスト1は、ルート要素の直下に<データ>要素が郵便番号の数だけ繰り返す、フラットなXML文書です。1個の<郵便番号>に対し、各々1個の<都道府県><市郡町村><町域>という情報が対応しており、RDBでも表現可能な構造です（ファイルサイズ18MB：総ノード数＝599,436ノード）。

リスト2は、各<都道府県>によって<市郡町村>と<町域>の出現回数が異なり、テーブルに格納するのが難しい不定形のXML文書です。<都道府県>の中に<市郡町村>があり、その中に<町

注1）これらのサンプルデータは、CSV形式のフリーデータをもとに、XSL変換によって作成しています。市町村合併などにより、現行の郵便番号とはデータが異なる場合があります。

リスト1：フラットな階層のXML文書（postalGuide_basic.xml）

```
<?xml version="1.0" encoding="Shift_JIS"?>
<新郵便番号>
 <データ>
  <都道府県>神奈川県</都道府県>
  <市郡町村>横浜市西区</市郡町村>
  <町域>みなとみらいクイーンズタワーＡ（１９階）</町域>
  <郵便番号>2206019</郵便番号>
 </データ>
 （略：以下<データ>繰り返し）
</新郵便番号>
```

サンプル動作環境における注意!!

本稿のサンプルは、巨大なファイルサイズのXML文書を扱っているため、マシンのスペックによっては動作が不安定になる場合があります。筆者の動作確認環境（Pentium4 2.4 GHz、メモリ512MBおよび756MB、HD100G、Windows XP Professional、IE6）未満の環境では、安定した動作の保証はできません。テストについては、サンプルデータのデータ件数を調整して実施するか、もしくは1回のテストごとに再起動するなど、個々の責任において判断してください。

域>があるという、データ同士の関係がわかりやすいツリー構造になっています（ファイルサイズ13.7MB：総ノード数＝365,390ノード）。

リスト3は、リスト2の<都道府県名>や<市郡町村名>や<町域名>要素を、それぞれ属性として扱ったXML文書です（ファイルサイズ10MB：要素ノード数＝242,454ノード、属性ノード数＝122,936ノード）。

これらのXMLファイルから、指定した「都道府県」と「市郡町村」に完全一致し、かつ「町域」に部分一致する「郵便番号」データを、いかに速く効率よく検索／抽出できるか、そのタイムを競ってみます。どのクラスの、どのメソッドやプロパティを使用し、どのように記述すればよいか、その組み合わせ方がポイントです。

都道府県を表示させるクラスライブラリの作成

postalAreaClass¥postalArea.vb

サンプルプログラムには、検索対象とするXMLファイル

リスト2：不定形の階層構造のXML文書（postalGuide_space_yes.xml）

```
<?xml version='1.0' encoding='Shift_JIS'?>
<新郵便番号>
<都道府県>
   <都道府県名>神奈川県</都道府県名>
   <市郡町村>
     <市郡町村名>横浜市西区</市郡町村名>
     <町域>
       <町域名>みなとみらいクイーンズタワーＡ（１９階）</町域名>
       <郵便番号>2206019</郵便番号>
     </町域>
     （略：以下<町域>繰り返し）
   </市郡町村>
   （略：以下<市郡町村>繰り返し）
</都道府県>
（略：以下<都道府県>繰り返し）
</新郵便番号>
```

※空白を削除すると、7.8MBまでサイズダウンすることができます（postalGuide_space_no.xmlを参照）。

リスト3：属性を持つXML文書（postalGuide_attribute.xml）

```
<?xml version="1.0" encoding="Shift_JIS"?>
<新郵便番号>
<都道府県 都道府県名="神奈川県">
   <市郡町村 市郡町村名="横浜市西区">
     <町域 町域名="みなとみらいクイーンズタワーＡ（１９階）">
       <郵便番号>2206019</郵便番号>
     </町域>
     （略：以下<町域>繰り返し）
   </市郡町村>
   （略：以下<市郡町村>繰り返し）
</都道府県>
（略：以下<都道府県>繰り返し）
</新郵便番号>
```

※空白を削除すると、7MBまでサイズダウンすることができます（postalGuide_attribute_space_no.xmlを参照）。

（リスト1～3のうちのいずれかひとつ）以外に、「都道府県」データのみ記録したXMLファイル（postalGuideArea.xml）も使います（図1）。このデータをもとに「都道府県」選択用DropDownList内に都道府県名を表示させる処理は、すべてのサンプルで共通です。そのため、DLLファイルとして作成し、各プロジェクトから参照して再利用します。

まずは、この都道府県名表示用XMLファイルにアクセスし、<都道府県名>ノードリストの取得をDLL化する手順をみてみましょう。