論文

中国語学習コンテンツの制作

XMLデータベースとJavaScriptをつかって

非常勤講師　齊藤正高

1.　はじめに

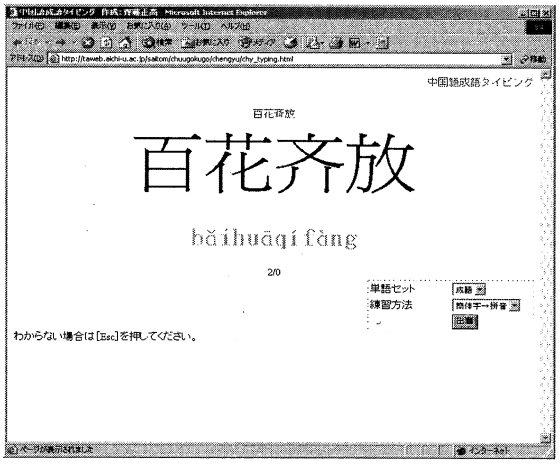
商用のショップサイトに劣らず、教育においてもデータベースが果たす役割は重大である。うまく構造化したデータベースを作成し、学習者による適切な入力を設定し、その入力を判断するしくみを作れば、基本的な学習コンテンツを作ることができる。

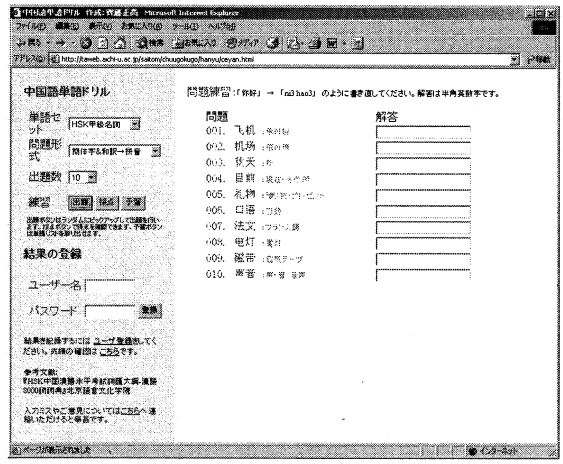
また、Unicodeの伝送形態であるUTF-8の浸透にともない、中国語の学習コンテンツもパッケージ化されたソフトから、ウェブベースのコンテンツに移行しつつある。検定試験問題のデータベース化や講義で使われるCALL副教材にも多くの実績がある。[1]

本稿はデータベースをもつウェブベースの学習コンテンツについて、その作成方法を報告する。筆者が作成したコンテンツは以下である。

1)中国語単語ドリル　2)中国語あいさつ文タイピング　3)音声つき単語タイピング
4)中国語偏旁名テスト　5)中国語量詞テスト　6)成語タイピング　7)発音分類別単語表

http://taweb.aichi-u.ac.jp/saitom/chuugokugo/chuugokugo.html





これらはじつに同工異曲であって、すべてXMLによるデータベースをJavaScriptで問題に加工し、HTMLでインターフェースを作成したものである。以下にくわしく述べる。

1　漢字文献情報処理研究会『漢字文献情報処理研究』創刊号2000年

2. XMLについて

まず、XMLについて簡単にまとめておきたい。XMLは1998年にW3Cから勧告された「拡張可能なマークアップ言語」であり、タグを自由に定義できる点が特長である。HTMLが文書の表示を行う言語だと考えれば、XMLはデータの構造をマークアップする言語と考えることができよう。一般的なXMLデータベースは以下の部分からなる。

2.1 XML宣言

XML宣言はXML文書の最初に書く宣言である。省略するとXMLとして認識されない場合があるので書いた方がよい。

例：

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
```

この宣言の書き方はHTMLよりも厳密で、xmlの部分を大文字にしたり、<?とxmlの間に空白を入れたり、version属性とencoding属性を転倒することはできない。versionは現在1.0である。encodingはShift_JISやUTF-8などの文字エンコーディングスキームを指定する。standaloneは外部にマーク付け宣言があるかどうかを記述する。省略するとnoになる。

2.2 DTD(Docment Type Definiton)

DTDではXML文書の構造（スキーマ）を定義する。本稿の使い方では省略しても動くので省く。ところで、昨年からDTDにかわるスキーマ言語「RELAX」、「XML Schema」をめぐって、XML開発者のあいだで「ボヘミアンと貴族の階級闘争」と呼ばれる対立がある。データの型を明確に定義し、XMLをバイナリでも扱えるようにしようと考える派閥（貴族）と、XMLを基本的にテキストデータと考える派閥（ボヘミアン）の対立である。吉松史彰氏はこれを「スキーマが重要か、それともインスタンスが重要か」という問題としてとらえ、以下のように「XMLの挑戦」をまとめる。

> 私たち現代の技術者にとっては、インスタンスはスキーマなしには存在し得ない。インスタンスの解釈は厳密にスキーマに依存する。ところが、現実を見てみると、実はスキーマなしでも立派にデータが流通し、運用できているシステムが世の中にはたくさん存在する。その代表例がWebだ。（中略）まずインスタンスありきで世界をつなぐシステムを構築する。それがXMLの挑戦であり、

2. XMLについて

まず、XMLについて簡単にまとめておきたい。XMLは1998年にW3Cから勧告された「拡張可能なマークアップ言語」であり、タグを自由に定義できる点が特長である。HTMLが文書の表示を行う言語だと考えれば、XMLはデータの構造をマークアップする言語と考えることができよう。一般的なXMLデータベースは以下の部分からなる。

2.1 XML宣言

XML宣言はXML文書の最初に書く宣言である。省略するとXMLとして認識されない場合があるので書いた方がよい。

例：

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
```

この宣言の書き方はHTMLよりも厳密で、xmlの部分を大文字にしたり、<?とxmlの間に空白を入れたり、versionとencodingの属性を転倒することはできない。versionは現在1.0である。encodingはShift_JISやUTF-8などの文字エンコーディングスキームを指定する。standaloneは外部にマーク付け宣言があるかどうかを記述する。省略するとnoになる。

2.2 DTD(Docment Type Definiton)

DTDではXML文書の構造（スキーマ）を定義する。本稿の使い方では省略しても動くので省く。ところで、昨年からDTDにかわるスキーマ言語「RELAX」、「XML Schema」をめぐって、XML開発者のあいだで「ボヘミアンと貴族の階級闘争」と呼ばれる対立がある。データの型を明確に定義し、XMLをバイナリでも扱えるようにしようと考える派閥（貴族）と、XMLを基本的にテキストデータと考える派閥（ボヘミアン）の対立である。吉松史彰氏はこれを「スキーマが重要か、それともインスタンスが重要か」という問題としてとらえ、以下のように「XMLの挑戦」をまとめる。

> 私たち現代の技術者にとっては、インスタンスはスキーマなしには存在し得ない。インスタンスの解釈は厳密にスキーマに依存する。ところが、現実を見てみると、実はスキーマなしでも立派にデータが流通し、運用できているシステムが世の中にはたくさん存在する。その代表例がWebだ。（中略）まずインスタンスありきで世界をつなぐシステムを構築する。それがXMLの挑戦であり、

パラダイムシフトなのだ。[2]

本稿の立場は、既存のデータを再利用を重視するので、「ボヘミアン」的な立場である。

2.3 XML 文書

XML 文書は HTML と同じように内容をタグで囲む。タグ名は自由に定義でき、漢字・ひらがなど２バイト文字を使うこともできる。ただし、半角カタカナと全角英数字が使えず、一文字目は文字かアンダースコアでなくてはならないという禁則もある。XML 文書としての形式は、HTML よりも厳密にかかねばならない。以下に、XML 文書の必要な形式を書く。

2.3.1 XML 宣言を除いた、すべてのタグは開始タグと終了タグがなければいけない。

例：

<単語リスト>～</単語リスト>

たとえば、HTML の
タグは終了要素がないが、XML の内部にこれを書く場合は、
</br>と書くか
と書かねばならない。/>は終了要素を省略した形である。

2.3.2 XML 宣言を除いた、全ての要素を包み込む「ルート要素」が１つだけ必要である。

×悪い例

```
<?xml version="1.0" encoding="utf-8"?>
<詞><ピンイン>ni3hao3</ピンイン></詞>
<詞><ピンイン>zai4jian4</ピンイン></詞>
```

○よい例……<単語リスト>がルート要素

```
<?xml version="1.0" encoding="utf-8"?>
<単語リスト>
<詞><ピンイン>ni3hao3</ピンイン></詞>
<詞><ピンイン>zai4jian4</ピンイン></詞>
</単語リスト>
```

---

[2] 吉松史彰「XML がもたらす夢と現実」2003/03/20
(http://www.atmarkit.co.jp/fdotnet/opinion/yoshimatsu/onepoint07.html)

2.3.3　タグには属性が付加できるが、属性の代入には必ず引用符（=”〜”か=’　’）が必要である。

例：

<単語リスト language="Chinese" category="あいさつ">〜
</単語リスト>

以上の約束を守れば、XMLとして認識され、ブラウザーでも表示できる。

3. XML文書の作成

XML文書はエディターでも作成できる。また、フリーのXMLエディターもいくつかあるが、大量のデータを扱う際には少々つかいにくい面がある。そこで、表計算ソフトのワークシートを直接マークアップし、XMLデータを作成する手法を紹介したい。この方法は既存のデータベースをXMLに変換する場合に有効である。また、Excel2003にはXML文書を更新する機能がついているので、すでにあるXML文書の追記に使うことができる。また、XML Schema（.xsd）ファイルをよみこめばXMLをエクスポートできる。

たとえば、以下の会話表現のデータがあるとき

	A	B	C	D	E
1					
2					
3		你好	ni3hao3	こんにちは	
4		您好	nin2hao3	(目上に)こんにちは	
5		再见	zai4jian4	さようなら	
6		对不起	dui4bu5qi3	ごめんなさい	
7		谢谢	xie4xie5	ありがとう	
8					

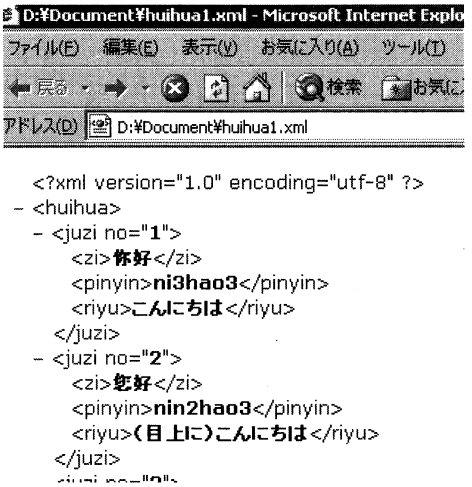
1行をレコードと考え、行を挿入しながら、以下のマークアップを行う。

	A	B	C	D	E	F	G
1							
2							
3	<juzi no='1'><zi>	你好	</zi><pinyin>	ni3hao3	</pinyin><riyu>	こんにちは	</riyu></juzi>
4		您好		nin2hao3		(目上に)こんにちは	
5		再见		zai4jian4		さようなら	
6		对不起		dui4bu5qi3		ごめんなさい	
7		谢谢		xie4xie5		ありがとう	
8							

これにルート要素とXML宣言を書き加え、以下のようにする。

	A	B	C	D	E	F	G
1	<?xml version='1.0' encoding='utf-8'?>						
2	<huihua>						
3	<juzi no='1'><zi>	你好	</zi><pinyin>	ni3hao3	</pinyin><riyu>	こんにちは	</riyu></juzi>
4	<juzi no='2'><zi>	您好	</zi><pinyin>	nin2hao3	</pinyin><riyu>	(目上に)こんにちは	</riyu></juzi>
5	<juzi no='3'><zi>	再见	</zi><pinyin>	zai4jian4	</pinyin><riyu>	さようなら	</riyu></juzi>
6	<juzi no='4'><zi>	对不起	</zi><pinyin>	dui4bu5qi3	</pinyin><riyu>	ごめんなさい	</riyu></juzi>
7	<juzi no='5'><zi>	谢谢	</zi><pinyin>	xie4xie5	</pinyin><riyu>	ありがとう	</riyu></juzi>
8	</huihua>						
9							

このワークシートを「.xml」の拡張子をつけて、Unicode（UTF16）テキストとして保存する。ブラウザーで表示するにはエディターを使い、UTF-8で保存しなおす必要がある。

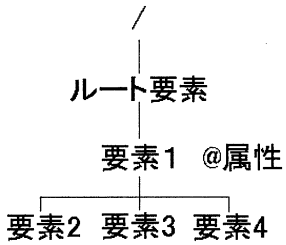


このままでは大量のタブをふくむので、ワードプロセッサの置換機能でとり除いてもよい。

4. XPath

XMLのデータを自在に抽出するにはXPathを用いるのが便利である。XPathを用いるには、事前にXML文書のマークアップ構造を知っておく必要があるが、教材作成者がXMLのマークアップを行った場合、この点は問題にならないと考えてよい。

XPathは次のような構造をもっている。



最上位は半角スラッシュで「/」で、XML文書のルート要素がその後にくる。属性は@を前に書いて表し、各要素のレベルの間はURLのように半角スラッシュで区切って表す。

例：

/huihua/juzi/zi

/huihua/juzi/@no

また、XPathには[〜]をつかい、条件を追加することもできる。

例：

/huihua/juzi[@no='2']/zi

/huihua/juzi[@no='2' or @no='3']/zi

/huihua/juzi[pinyin='ni3hao'][@no='1']/zi

……[～]を追加すればand条件を設定。

/huihua/juzi[@no>2]/zi

……「>」は実体参照（>）で表現する。

さらに、XPathには関数がある。

四則演算（とくに除算の記号に注意）

+	加算
-	減算
*	乗算
div	除算
mod	剰余

数値関数（主なもの）

sum( )	合計をとる
count( )	要素の数を返す
floor( )	小数点以下の切り捨て
ceiling( )	小数点以下の切り上げ
round( )	四捨五入した整数を返す

文字関数（主なもの）

contains(stringA,stringB)	StringAにstringBが含まれる場合True
Substring(StringA,B,C)	StringAのB番目からC文字分切り出す。
Substring-after(StringA,StringB)	StirngAからStringBより後の部分を返す。
Substring-before(StringA,StringB)	StirngAからStringBより前の部分を返す。

例：日本酒店のXMLデータを処理するXPath

- sum(酒[価格>=10000]/価格) div count(酒[価格>=10000] /価格)

→1万円以上の酒の平均価格を算出。

- count(酒[contains(名前,'蓬莱泉')])

→名前のタグに「蓬莱泉」をふくむデータの個数を算出。

XPath は以下のように XSL（eXtensible Stylesheet Language）で使うこともできる。

例：nihonshu.xml

```
<?xml version='1.0' encoding='Shift_JIS'?>
<?xml-stylesheet type="text/xsl" href="kakakuhyou.xsl"?>
<商品データ>
<酒>（以下略）
```

↓kakakuhyou.xsl

```
<?xml version="1.0" encoding="Shift_JIS"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/商品データ">
<p align="center"><font color="Blue">価格表</font></p>
<table align="center" width="50%" border="1">
<xsl:for-each select="酒[価格&lt;3000]">
<xsl:sort select="価格" data-type="number" order="descending"/>
<tr><td width="40%"><xsl:value-of select="名前"/></td>
<td width="30%"><xsl:value-of select="容量"/>ml</td>
<td width="30%">¥<xsl:value-of select="価格"/>－</td></tr>
</xsl:for-each>
</table>
</xsl:template></xsl:stylesheet>
```

（XSL の解説）

- 1 行目 xml 宣言、2 行目は xsl の開始タグ（ネームスペースを指定している）
- xsl:template match="XPath"（match 以下の要素に以下の書式を適応）
- xsl:for-each select="XPath"（select 以下の要素がある限り、以下の書式を適応）
- xsl:sort select="タグ名"（for-each の内容をタグ名をキーに並べかえ）
- xsl:value-of select="XPath"(select 以下の XPath のテキストを取り出す）

5. スクリプト言語による制御

XPath は XSL で用いるほか、XML パーサをつかって、Javascript などのスクリプト言語で制御することができる。XML 文書を呼び出す場合、Javascript では以下のように書く。

```
var DOM = new ActiveXObject("Msxml2.DOMDocument");
DOM.async = false;
DOM.load ("ファイル名.xml");
```

"Msxml2"とはInternetExplore6以降に附属している「XMLパーサ」で、XMLを読むための部品である。これをオブジェクトに作成する。asyncというプロパティは、Trueのとき以下のXML文書を読込ながらスクリプトを実行する。Falseにすれば、XML文書を読み込んでから以下のスクリプトが動く。loadというメソッドはファイル名を指定してXML文書を読み込む。XML文書はインターネットテンポラリーフォルダーに保存される。

このXML文書に対してXPathを用いるには以下のように書く。

```
var riyu=DOM.documentElement.selectNodes("XPath");
```

documtentElmentに対してselectNodesメソッドを実行し、実行結果をオブジェクトに格納する。このXML文書の部分集合からテキストを取り出すには以下のようにする。

```
yaku=riyu.item(0).text
```

item(N).textプロパティが個別の要素のテキストである。selectNodesメソッドの結果が複数である場合はitem(N)が配列となっており、添え字による指定が可能である。結果の長さは以下で知ることができる。

```
nagasa=riyu.length;
```

6. 中国語教材作成の実際

中国語の単語ドリルを作成した際、入力と表示を考慮し以下のデータ構造をつくった。

```
<?xml version="1.0" encoding="utf-8" ?>
- <hsk_jia_dong>
  - <ci no="1">
      <zi lang="zh">爱</zi>
      <pinyin1 lang="zh">ài</pinyin1>
      <pinyin2 lang="en">ai4</pinyin2>
      <english lang="en">love</english>
      <yisi lang="ja">愛する、愛</yisi>
    </ci>
  - <ci no="2">
      <zi lang="zh">安排</zi>
      <pinyin1 lang="zh">ānpái</pinyin1>
      <pinyin2 lang="en">an1pai2</pinyin2>
      <english lang="en">arrange ;arrangement</english>
      <yisi lang="ja">手はずを整える</yisi>
    </ci>
  - <ci no="3">
      <zi lang="zh">摆</zi>
      <pinyin1 lang="zh">b□i</pinyin1>
      <pinyin2 lang="en">bai3</pinyin2>
      <english lang="en">put</english>
      <yisi lang="ja">並べる</yisi>
    </ci>
```

hsk_jia_dong はルート要素であり、漢語水平考試（HSK）の甲級単語の動詞であることを表す。ci は単語のレコードを表す。no 属性はプライマリキーである。zi は簡体字での単語表記を指す。pinyin1 は表示用の発音記号である。pinyin2 は入力の照合用データでピンインの声調部分を数値化してある。english は zi の英訳である。yisi は日本語訳である。

ピンインについては、IME の一般的な入力方法と、表記方法が異なるので、このように分割しておくのがよい。具体的には以下のテーブルを使って、Excel マクロで変換をおこなっている。全部で 132 パターンの変換例を作れば、pinyin2 のデータから pinyin1 のデータを作成できる。

	A	B	C
1	iang1	iāng	
2	iang2	iáng	
3	iang3	iǎng	
4	iang4	iàng	
5	uang1	uāng	
6	uang2	uáng	

この XML データベースをつかって、ランダム出題の単語ドリルをつくる手順は以下になる。