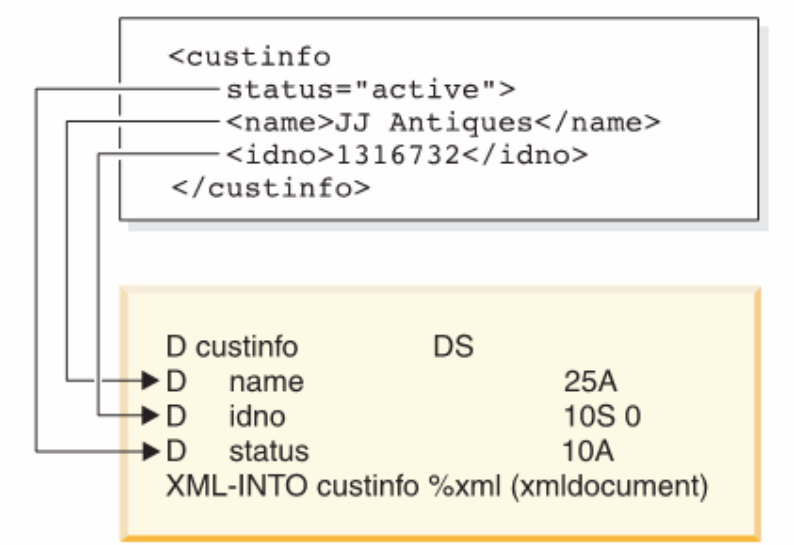
使用 V5R4 RPGIV 處理 XML 文件

處理簡單 XML 的文件

RPG 程式要如何從 XML 文件中取得所要的資料？你是使用字串掃描或使用其他 XML Parser(分析器)如 Open source JDOM。在 i5/OS V5R4 RPG 中已新加入一個新工具 XML-INTO 運算碼供解析 XML。

如果你需要 XML 的基礎觀念，可以使用 google 尋找“XML 介紹”，在此提供一個網頁供參考: http://www.cs.nchu.edu.tw/~djchou/xml/xml.htm。在我們詳細說明運算碼 XML-INTO 前，我們先來看看使用 RPG 程式碼來處理 XML 文件，有多簡單。



上圖上半爲原始 XML 文件內容，「<keyword>」稱爲起始標籤(start tag)，而相對「</keyword>」則稱爲結束標籤(end tag)，也就是說起始標籤與結束標籤必須成對出現(<custinfo></custinfo>, <name></name>, <idno></idno>)。對於不需結束標籤的空元素(empty element)以「/>」做結尾。(例 <empty/>)，屬性(attribute)值要以「”」括起(status=”active”)。所以 custinfo, name, idno 可以稱爲元素(element)或節點(node)，而 status 則稱爲屬性(attributes)。

而下半部爲處理 XML 文件的 RPG 程式碼，如果你比對 XML 的片斷與 RPG 程式碼中的資料結構定義，你會發現他們有相似的地方。 XML 最外圍的元素名稱(custinfo)與 RPG 的資料結構名稱相同，而且 XML 的子元素名稱與 RPG 資料結構中的欄位名稱相同(name, idno, status)。
使用 XML-INTO 的關鍵在於要定義一個與 XML 文件元素同名的 RPG 變數，使用下列二個步驟：

1. 寫下 XML 文件中每一個完整路徑屬性(Attributes)及路徑中含有資料的元

素，而非子元素或屬性。這張表上所列 XML 名稱的順序都必須要確認能從 XML 文件中取得資訊。例如從上圖中 XML 文件取得完整路徑名稱爲 custinfo.name, custinfo.idno 及 custinfo.status.

2. 爲了要直譯 XML 路徑當成 RPG 的變數，所以需要將 RPG 的資料結構宣告爲支援限制名稱(qualified names)，例如：

   D custinfo ds qualified

   而此範例中僅有一階，所以可以不用使用限制名稱。

資料結構中所定義資料的型態(文字或數字)取決於你對該 XML 文件中各元素值的認知。例如，如果 idno 的值總是只有數字資料，那就使用數字型態而非文字型態，但若此資料是要寫入資料庫中，這也要視資料庫中該欄位的定義而定，資料型態並非不可變動。 XML-INTO 支援所有資料型態，只有 指標(pointer)及物件(object)除外。可以參閱 ILE RPG Language Reference (publib.boulder.ibm.com/infocenter/iseries/v5r4/index.jsp?topic=/rzahg/rzahgrpgile.htm) 關於 XML-INTO 支援 date，time 及 timestamp 型態的處理。當你選擇資料型態時要謹記在心，在將 XML 資料放入 RPG 資料結構的子欄位時，若遭遇不正確的資料，此時 XML-INTO 會立即失敗，如果你懷疑有可能是不正確的數字或日期資料，最好將這些欄位定義爲文字型態，緊接著於 XML-INTO 後自行使用 %DEC 或 %DATE 轉換資料。

文件及程式範例：

文件１：

```
<order>
  <id>15327</id>
  <part>X24-A</part>
  <quantity>14</quantity>
  <cust>942234</cust>
</order>
```

文件２：

```
<order id="15327"
       quantity="14"
       cust="942234"
       part="X24-A" />
```

文件３：

```
<order id="15327">
```

```
   <part>X24-A</part>
   <cust>942234</cust>
   <quantity>14</quantity>
</order>
```

程式１：

此處理程式可以處理上述三份 XML 文件

處理程式片斷：

```
D order           DS
D   id                            15a   varying
D   part                          25a   varying
D   quantity                      15p 0
D   cust                          15a   varying

 /free

    XML-INTO order %XML(filename : 'doc=file');

    *InLr = *On;
 /end-free
```

說明：’doc=file’ 指 XML 文件來源爲置於 IFS 目錄中之檔案

文件４：

```
<order>
   <customer id="942234"
     location="Alexander Street"/>
   <part id="X24-A" quantity="14"/>
</order>
```

此 XML 文件比較複雜，階層數大於二階，無法使用前一處理程式。要使用 XML-INTO 先決要件是 RPG 的資料結構要與 XML 文件中巢狀結構中的元素或屬性及 RPG 資料結構中的子欄位要相配。依照先前所提的二個步驟記下 XML 文件中的元素或屬性路徑名稱 (order.customer.id, order.customer.location, order.part.id, and order.part.quantity)，接著定義 RPG 資料結構支援限制名稱 (qualified name)，RPG 資料結構 order 需要一個子欄位 id 相對應到 XML 文件中 id 屬性，而且需要子欄位 customer 及 part 相對應到 XML 文件中同名的元素。所要注意的是 RPG 資料結構中子欄位 customer 及 part 均是一個資料結構，各有兩個子欄位。下列爲處理文件４的程式片斷：

使用 V5R4 RPGIV 處理 XML 文件

程式２：

```
D order           DS                  qualified
D   id                           15a  varying
D   customer                          likeds(customer_type)
D   part                              likeds(part_type)


D customer_type   DS                  qualified
D   id                           15a  varying
D   location                    100a  varying

D part_type       DS                  qualified
D   id                           25a  varying
D   quantity                     15p 0

 /free

    XML-INTO order %XML(filename : 'doc=file allowextra=yes
                                    allowmissing=yes');
```

上述程式中資料結構 order 有三個欄位，id, customer, part 其中子欄位 customer 及 part 爲資料結構，各自定義 XML 中 customer 及 part 的屬性。

處理 XML 中重複結構的資料

文件５：

```
<orders>
 <cust>942234</cust>
 <order>  <id>14327</id>  <quantity>14</quantity> </order>
 <order>  <id>3412</id> <quantity>100</quantity> </order>
 <order>  <id>84823</id> <quantity>75</quantity> </order>
</orders>
```

一般的 XML 文件大多是含有重複相同元素但不同資料的類型。寫下文件５每一元素的完整路徑如下：

- orders.cust
- orders.order.id
- orders.order.id
- orders.order.id
- orders.order.quantity
- orders.order.quantity
- orders.order.quantity

由以上表列得知 orders.order.id 及 orders.order.quantity 是重複的元素，所以重複的子欄位 order 要使用索引如下：

- orders.cust
- orders.order(i).id
- orders.order(i).quantity

下列程式爲處理上述 XML 文件５重複結構資料的程式：

程式３：

```
D MAX_ORDERS      C                   32767

D orders          DS                  qualified
D   cust                         15a  varying
D   order                            likeds(order_type)
D                                    dim(MAX_ORDERS)

D order_type      DS                  qualified
D   id                           15a  varying
D   quantity                     10i 0

 /free

     clear orders;
     XML-INTO orders
              %XML(filename
                 : 'doc=file allowmissing=yes');

     for i = 1 to MAX_ORDERS;
        if orders.order(i).id = *blanks;
```

```
            leave;
          endif;
          ... process the order ...
      endfor;
```

上述程式中資料結構 orders 所有欄位均對應到 XML 文件中的元素，其中 子欄位 order 被定義成索引矩陣(array)。文件 5 中只有三個 order 元素，但 RPG 中宣告 order 的矩陣大小爲 32767，但 XML-INTO 預設的處理程序是 XML 文件需要全部對應到 RPG 資料結構的欄位名稱(XML 的元素或屬性)、巢狀及數目。如果 XML 文件中 order 確實只重複三次，RPG 資料結構矩陣是定義爲 DIM(3) 而不是 DIM(32767)，你能使用簡單的 XML-INTO 語法執行而不會發生問題。

但是事實上有時會出現更多次，所以我們難於限定 XML 的資料比數，因此我們就預設 RPG 中矩陣的最大數量 32767。但是 XML-INTO 預設是所有元素及屬性都需要有相對應的欄位擺放空間，執行才不會有錯誤，此時我們所定義接收的矩陣大小大於實際應有的筆數，而 %XML 的選項(option)中提供一個參數值 allowmissing=yes 允許 RPG 子欄位或矩陣子欄位不接收 XML 資料，所以當使用此種方法時，於執行前先將矩陣清空(clear orders)，在處理矩陣時，去測試子欄位值爲空白或數字爲零作爲資料是否中止的判斷。

雖然我們使用選項 allowmissing=yes 允許接收比較少個數的值填入矩陣，這也意味著 XML 文件中其他的值也有可能遺失，例如，文件中缺少了 quantity 元素，如果你懷疑 XML 文件不完整，當你處理 XML 的時候，就要判斷欄位值爲空白或數字爲零。

另一種處理方式較複雜，可以將處理 XML 文件分爲二部份，一是只處理 cust 資訊，二是處理 order 矩陣。當你指定矩陣變數於 XML-INTO 第一個運算元時，這種方式較可以信賴，因爲 RPG 不會規範 XML 元素的數量要與 RPG 矩陣元素的大小一樣。在這個案例中，所處理真正 XML 元素的真正筆數存在於 Program Status Data Structure (PSDS) 的子欄位 ”number of XML elements” 中，那意味著你不必指定 allowmissing=yes 的選項。

使用 V5R4 RPGIV 處理 XML 文件

下列爲分二部份處理 XML 文件５程式碼：

程式４：

```
D psds            SDS                   qualified
D   numXml                         20i 0 overlay(psds:372)

D MAX_ORDERS      C                     32767

D orders          DS                    qualified
D   cust                           15a  varying
D   order                               likeds(order_type)
D                                       dim(MAX_ORDERS)

D order_type      DS                    qualified
D   id                             15a  varying
D   quantity                       10i 0


 /free

     // 取單一欄位值
     XML-INTO orders.cust
               %XML(filename :
                     'doc=file  path=orders/cust');

     // 取重複的 order 填入陣列
     XML-INTO orders.order
               %XML(filename :
                     'doc=file path=orders/order');

     for i = 1 to psds.numXml;

        // process the order

     endfor;
```

此程式使用文件５所衍生的資料結構，但與程式４所使用的 XML-INTO 運算元不同，因爲程式４使用了 path(路徑) 選項。其中 path=orders/cust 只取單一欄位，path=orders/order 取重複的 order 填入矩陣，而 order 的真正筆數會填入 psds.numXml

中，程式即可處理真正 XML 的筆數。

當程式執行第一個 XML-INTO 完成時，由於元素 cust 位於文件５前段，此時一旦找到 cust XML 元素所需的資訊，解析程序立即停止。當第二個 XML-INTO 執行時，他會忽略 cust ，因爲 cust 並不在 path(路徑) orders/order 內。然而如果 cust XML 元素位於 XML 文件的尾端(在所有 order 元素的後面)，此時二個 XML-INTO 均需要解析整份文件，亦及第一個 XML-INTO 需要忽略文件中所有資訊，除了 cust 外。所以 XML 文件的大小，XML 元素間解析的先後與執行效能有關，需要注意。

處理超過 32767 筆重複資料

如果你的 XML 重複結構的資料超過 RPG 的最大矩陣限制 32767，你必須使用 XML-INTO 的 %HANDER 運算元爲第一個運算子，而不是使用 RPG 的欄位或資料結構當成第一個運算子，%HANDER 爲內建函式，%HANDER 有二個運算子：

1. 處理 XML 的函式名稱，此函式需定義爲一個整數回傳值及三個參數如下：
   i. 隨意型態的一個變數(可以是 Indicator，整數或複雜的資料結構)
   ii. 矩陣(用以接收 XML 重複結構資料的矩陣，矩陣的大小代表每次所處理的最大筆數，如矩陣大小爲 10，則每次處理 10 筆，直到全部完成)
   iii. 第二個參數矩陣內所含真正有資料的筆數

2. 一個與 XML-INTO %HANDER 所指定處理函式第一個參數相同的參數，此參數爲 XML-INTO 及%HANDER 所指定處理函式的介面，可將資料傳入所指定處理函式。

程式５爲利用 %HANDER 指定處理 XML 函式

程式５：

```
D customer        S              15a

D order_type      DS                    qualified
D   id                           15a   varying
D   quantity                     10i 0
```

使用 V5R4 RPGIV 處理 XML 文件

```
D orderHandler    PR            10i 0
D   custId                            like(customer)
D   orders                            likeds(order_type)
D                                     const dim(32767)
D   num                         10i 0 value

 /free

    XML-INTO customer %xml(filename :
                           'doc=file path=orders/cust');

    XML-INTO %handler(orderHandler : customer)
                           'doc=file path=orders/order');

 /end-free

P orderHandler    B
D orderHandler    PI            10i 0
D   custId                            like(customer)
D   orders                            likeds(order_type)
D                                     const dim(32767)
D   num                         10i 0 value

D i               S             10i 0

 /free

    for i = 1 to num;
       // process orders(i) along with the
       // first "custId" parameter
    endfor;

    return 0;
 /end-free
P orderHandler    E
```

1. 當 XML-INTO 使用選項 path 時，因爲沒有 RPG 變數可以對應 XML 元

素，所以就必須使用 %HANDLER。而 %HANDLER 所指定函數的第二個矩陣參數若是資料結構(以 LikeDS，LikeREC 定義)，只會使用到有對應到 XML 元素或屬性子欄位的名稱。

2. 接收 cust 資料的 RPG 變數爲 customer 不是 cust，當使用選項 path 時，RPG 變數並不用對應到 XML 的元素名稱，所以他不需要與 XML 元素同名。

3. 在此範例中，customer 欄位被傳輸到 %HANDELER 所指定的處理 XML 函式 orderHandler 的第一個參數，所以 XML 元素 cust 的值可以運用到所有 order 的 XML 元素。

4. 當處理 XML 函式全部處理完成時，會回傳一個 0 的值，代表已處理完成。如果處理 XML 函式決定不繼續處理，就回傳一個非 0 的值，當有此種情況發生時，XML-INTO 運作會收到一個解析運作未完成的錯誤。這個錯誤可以使用 XML-INTO(E) 來處理。

其他 XML-INTO 選項 Options 說明：

前面僅提及 allowmissing and path 兩個選項，還有其他選項需要客制以符合 XML-INTO 的使用方式。

case : 到目前爲止，所有 XML 文件中元素名稱的範例均是使用小寫，RPG 預設所有 XML 元素名稱均是小寫。如果 XML 元素名稱不是小寫，則 RPG 解析器就不會找到相對映的子欄位名稱，這選項預設是 case=lower，但你能指定 case=upper 或 case=any。case=upper 及 case=lower 處理上會比 case=any 快，因爲稍後與 RPG 子欄位名稱比對前會將每個 XML 元素名稱轉換爲大寫，然而 case=any 讓你的 XML 文件具有彈性，不管你的 case 選項爲何，XML 解析器需要起始標籤與結束標籤名稱的大小寫要一致，縱然你指定 case=any，你也不能指定起始標籤爲<TAG>，結束標籤爲</tag>。如果起始標籤及結束標籤不一致，XML-INTO 會產生解析錯誤。

allowextra： 偶然間，XML 文件會含有你沒有興趣或不想要的資訊。而 RPG 預設是 RPG 資料結構的子欄位全部要對應到 XML 文件的所有元素，爲了要允許額外的 XML 元素存在，但又不用對應到 RPG 的欄位名稱，亦及忽略未被 RPG 定義的 XML 元素，需要指定選項 allowextra=yes。

trim：RPG 預設是 trim=yes，RPG 針對 XML 的文字資料(character, UCS-2, or DBCS)，會自動將前後空白刪掉，字串中字元間只保留一個空白。若要保留原始

的資料內容，使用 trim=no。

ccsid：當 XML 解析器不支援 Job 的 CCSID 時，就需要使用這個選項。如果需要使用這個選項，指定 ccsid=ucs2 讓 XML 文件轉換到 UCS-2 以讓解析程序得以順利完成，當所解析的資料配給 RPG 子欄位時，系統會自動從 UCS-2 轉換到 RPG 變數所需的 CCSID。若你的系統 DBCS 時，就必須指定這個選項 ccsid=ucs2。

doc：前面所有範例 XML 文件均是放置於 IFS 目錄中的檔案，所以使用 doc=file，但 RPG 變數預設是 doc=string。下列爲 XML 字串使用方式：

```
D xmlData          S           65535a

 /free

   XML-INTO orders %XML(xmlData);
```

如果你有 XML 字串超過 65535 時，將 XML 字串寫入檔案，然後使用 doc=file 解析 XML 文件，請參照 *ILE RPG Programmer's Guide*(publib.boulder.ibm.com/infocenter/iseries/v5r4/index.jsp?topic=/rzahg/rzahgrpgile.htm) Figure 76, "Writing data to an Integrated File System file”