XML

1. Definisi XML

XML kependekan dari ***eXtensible Markup Language***, dikembangkan mulai tahun 1996 dan mendapatkan pengakuan dari W3C (http://www.w3c.org) pada bulan Februari 1998. Teknologi yang digunakan pada XML sebenarnya bukan teknologi baru, tapi merupakan turunan dari SGML yang telah dikembangkan pada awal 80-an dan telah banyak digunakan pada dokumentasi teknis proyek-proyek berskala besar. Ketika HTML dikembangkan pada tahun 1990, para penggagas XML mengadopsi bagian paling penting pada SGML dan dengan berpedoman pada pengembangan HTML menghasilkan markup language yang tidak kalah hebatnya dengan SGML.

Seperti halnya HTML, XML juga menggunakan elemen yang ditandai dengan **tag pembuka** (diawali dengan '<' dan diakhiri dengan '>'), **tag penutup** (diawali dengan '</ 'diakhiri '>') dan atribut elemen(parameter yang dinyatakan dalam tag pembuka misal <form name="isidata">). Hanya bedanya, HTML medefinisikan dari awal tag dan atribut yang dipakai didalamnya, sedangkan pada XML kita bisa menggunakan tag sendiri.

Contoh:

```
<football>
    <players>
        <playername>Cristiyan Gonzales</playername>
        <team>PSSI</team>
        <number>9</number>
        <country>Indonesia</country>
    </players>
    <players>
        <playername>Markus Horison</playername>
        <team>PSSI</team>
        <number>1</number>
        <country>Indonesia</country>
    </players>
</football>
```

pada contoh diatas <football>, <players> <playername>,dan <team> bukanlah tag standar yang telah tetapkan dalam XML. Tag-tag itu kita buat sendiri sesuai keinginan kita.

2. Mengapa XML ?

XML untuk saat ini bukan merupakan pengganti HTML. Masing-masing dikembangkan untuk tujuan yang berbeda. Kalau HTML digunakan untuk menampilkan informasi dan berfokus pada bagaimana informasi terlihat, XML mendeskripsikan **susunan informasi** dan berfokus pada informasi itu sendiri. XML terutama dibutuhkan untuk menyusun dan menyajikan informasi dengan format yang tidak mengandung format standar layaknya heading, paragraf, table, dsb. Sebagai contoh apabila kita ingin menyimpan dan menyajikan informasi terkait dengan pemain sepak bola, kita dapat menyimpannya dengan xml sebagai berikut:

```
<?xml version="1.0" encoding="UTF-8"?>
<sepakbola>
        <pemain>
                <nama>Gabriel Omar Batagor</nama>
                <kesebelasan>Argentina</kesebelasan>
        </pemain>
        <pemain>
                <nama>Ronaldinyo</nama>
                <kesebelasan>Real Madried</kesebelasan>
        </pemain>
        <pemain>
                <nama>Andrea Pirlo</nama>
                <kesebelasan>Milan</kesebelasan>
        </pemain>
    </sepakbola>
```

3. Komponen Dokumen XML

a. **Prolog**

Mendefinisikan versi XML, definisi entitas, dan DOCTYPE. Contoh:

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

- versi XML yang digunakan
- dokumen eksternal, misalnya : DTD/Entitas, XSL/XSLT/CSS, XPath, Xquery – *Processing Intruction; perintah pengolahan dokumen XML*

b. **Komponen dokumen**

- Tag dan Atribut
- CDATA (Character Data)
- Entitas (node root)
- Instruksi Pemrosesan (xsl/xslt/DTD)->Processing Instruction
- Komentar

4. Detail Dokumen XML

Sebuah dokumen XML terdiri dari bagian bagian yang disebut dengan node. Node-node itu adalah:

a. **Root node** yaitu node yang melingkupi keseluruhan dokumen. Dalam satu dokumen XML hanya ada satu root node. Node-node yang lainnya berada di dalam root node.

b. **Element node (Child node)** yaitu bagian dari dokumen XML yang ditandai dengan tag pembuka dan tag penutup, atau bisa juga sebuah tag tunggal elemen kosong seperti <anggota nama="budi"/> . element node biasa juga disebut root element

c. **Attribute node ()** termasuk nama dan nilai atribut ditulis pada tag awal sebuah elemen atau pada tag tunggal.

d. **Text node**, adalah text yang merupakan isi dari sebuah elemen, ditulis diantara tag pembuka dan tag penutup

e. **Comment node** adalah baris yang tidak dieksekusi oleh parser , kecenderungan berisi omentar/ penjelasan dari dokumen XML (<!-- comment-→)

f. **Processing Instruction node**, adalah **perintah pengolahan dalam dokumen XML**. Node ini ditandai awali dengan karakter <? Dan diakhiri dengan ?>. Tapi perlu diingat bahwa header

standard XML <?xml version="1.0" encoding="iso-8859-1"?>, <?xml version="1.0" encoding="UTF-8"?> bukanlah processing instruction node. Header standard bukanlah bagian dari hirarki pohon dokumen XML.

g. **NameSpace Node**, node ini mewakili deklarasi namespace

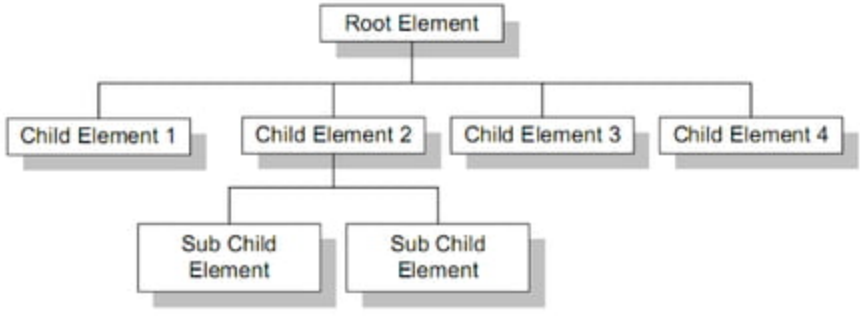
```
<?xml version="1.0" encoding="UTF-8"?>                                   ← Standar Header
<?xml-stylesheet type="text/xsl" href="D:\KULIAH_S2\Semester 2\Database Management Systems\Latihan XML\latihan xml
Kamid\TabelPemainBaru.xsl"?>                                             ← Processing Instruction
<!-- Dokumen ini menjelaskan tentang sepak bola-->                       ← Comment
<sepakbola>                                                              ← Root node
        <pemain status="inti">
                <nama kondisi="aktif">David Beckham</nama>               ← Atribut node
                <kesebelasan>AC Milan</kesebelasan>
        </pemain>
        <pemain status="Inti">
                <nama kondisi="cedera">Ronaldinyo</nama>
                <kesebelasan>Barcelona</kesebelasan>
        </pemain>
        <pemain status="cadangan">
                <nama kondisi="aktif">Alesandro Nesta</nama>
                <kesebelasan>AC Milan</kesebelasan>
        </pemain>
        <pemain status="inti">
                <nama kondisi="aktif">Messi</nama>
                <kesebelasan>New Castle United</kesebelasan>
        </pemain>
</sepakbola>
```

5. Aturan Dokumen XML

Dibandingkan dengan HTML, XML lebih rumit/ **CEREWET**. Kalau kita menulis sebuah dokumen HTML, beberapa kesalahan penulisan masih ditolerir. Misalnya kita menempatkan tag bersilangan seperti *** Huruf Tebal *** meskipun tidak dianjurkan, HTML masih bisa bekerja dan menampilkan hasil seperti yang kita inginkan. Tidak demikian dengan XML. Lebih jelasnya kita akan bahas di bawah bagaimana membuat dokumen XML yang baik.

a. **Heading standard untuk Document XML**
   Biasakanlah setiap membuat dokumen XML diawali dengan heading standard XML. Formatnya adalah sebagai berikut:
   <?xml version="1.0" encoding="iso-8859-1"?> atau <?xml version="1.0" encoding="UTF-8"?>

b. **Dokumen XML harus memiliki Root tag**
   Sebuah dokumen XML yang baik harus memiliki ***root tag***. Yaitu tag yang melingkupi keseluruhan dari dokumen. Tag-tag yang lain, disebut ***child tag***, berada didalam root membentuk hirarki seperti gambar1:



Gambar1. Diagram hirarki dokumen XML

Contoh:

```
<root>
    <child>
        <subchild></subchild>
    </child>
</root>
```

c. **Tag pada XML harus lengkap berpasangan**
Pada HTML beberapa elemen tidak harus berpasangan. Contoh berikut ini diperbolehkan dalam penulisan HTML.
* paragraph pertama*
* paragarap kedua*
yang demikian tidak berlaku pada XML. Kita harus menulis pula tag penutup untuk setiap tag yang kita buat. Penulisannya harus seperti ini
* paragraph pertama *
* paragarap kedua *
Tag tunggal hanya diperbolehkan untuk elemen kosong. Contoh penulisannya sebagai berikut:
<anggota nama="budi"/>

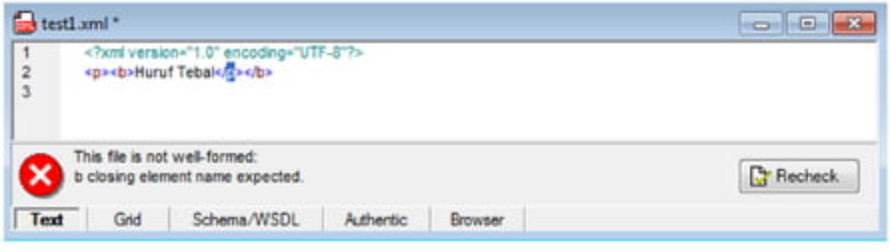
d. **XML membedakan huruf besar dengan huruf kecil**
Pada XML, <tanggal> berbeda dengan <Tanggal>. Tag pembuka dan tag penutup harus sama susunan huruf besar dan kecilnya.
<contoh>ini penulisan yang salah</Contoh>
<contoh>ini baru betul</contoh>

e. **Penyarangan tag harus benar (*not overlap*).**
Penulisan tag pada XML harus mengikuti aturan **Last In First Out (LIFO).** seperti yang kita bahas terdahulu, pada XML kita tidak bisa membuat tag yang saling bersilang seperti dibawahini
** aku belajar webdb **

*** Huruf Tebal *** tapi harus disusun seperti ini

* Huruf tebal *

Bila dipaksakan akan muncul komentar bahwa dokumen tidak sesuai dengan format (*not well-formed*)



f. **Nilai atribut harus diletakkan diantara tanda petik**

Seperti HTML, XML memiliki atribut. Nilai atribut harus diletakkan diantara dua tanda petik. Tidak masalah apakah tanda petik tunggal atau tanda petik ganda. Contoh dibawah ini dua-duanya benar <pesan dari="**lusy**"> atau

<pesan dari='lusy'>

g. **Penamaan tag (element) dan atribut**

Nama tag bisa terdiri dari huruf, angka dan underscore("_"). Karakter awal nama tag harus berupa huruf atau underscore ("_"), tidak diawali dengan kata xml atau XML, (misal:<xmlstring>), dan tidak mengandung spasi. Aturan penamaan atribut sama dengan aturan penamaan tag.

Contoh:

<xmlberita>Saya sedang belajar XML</xmlberita>

h. **Menyisipkan komentar**

Pada bahasa pemrograman atau scripting kita mengenal adanya komentar (*comment*). Komentar adalah kalimat/baris yang tidak dieksekusi oleh compiler, browser atau parser. Untuk menyisipkan komentar pada dokumen XML menggunakan <!-- *comment* --> caranya adalah sebagai berikut:

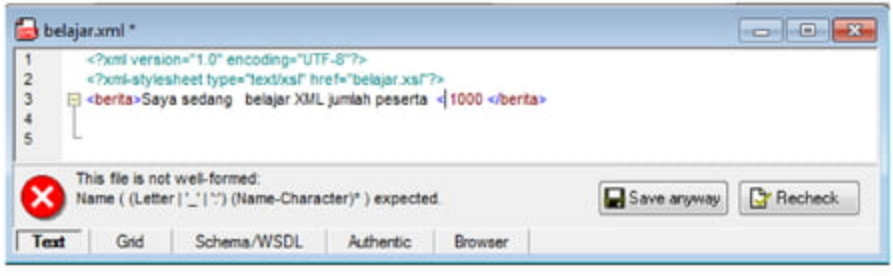
Contoh :

<!—Baris ini tidak di eksekusi oleh parser -->

i. **Menggunakan karakter ilegal apada XML**

Sama seperti pada HTML, anda tidak bisa menggunakan karakter seperti kurung siku (< atau >), petik tunggal (‘), dan petik ganda (“) .

Contoh dibawah ini akan menghasilkan error kalau di eksekusi :



Gambar2. Kesalahan pada dokumen XML dengan menggunakan karakter ilegal

<syarat>Saya sedang belajar XML jumlah peserta < 1000 </syarat>

Untuk menghindarinya, kita harus menggantikannya dengan **entity reference** seperti di bawah ini:

<syarat> Saya sedang belajar XML jumlah peserta **<** 1000 </syarat>

Entity reference selengkapanya ditunjukkan didalam tabel berikut:

Entity refences	Character	Character Name
&lt;	<	Lest Than
&gt;	>	Greater Then
&amp;	&	Ampersand
&apos;	'	Apostrophe
&quot;	"	Quatation Mark

Kesimpulan :

Jika dokumen makin komplek maka aturan tetap sama:

a. Mulai dengan deklarasi XML dokumen (Prolog)
b. Paling tidak harus ada satu elemen / node root
c. Tag harus berpasangan dan benar
d. Gunakan tag awal dan akhir untuk elemen-elemen yang tidak kosong
e. Beri tanda kutip untuk nilai atribut.

Dokumen yang tidak *wel formed* tidak akan diproses dengan benar dan dapat menyebabkan kesalahan pada parser, karena itu setiap dokumen yang dinyatakan hatuslah ***well-formed***. Karena setiap well-formed dokumen tidak harus mengikuti sebuah struktur tertentu, tp susunannya ditentukan oleh bagaimana data akan diinterpretasikan dan digunakan dan juga dibandingkan dengan **DTD**.

JIKA SEBUAH DOKUMEN SUDAH *WELL-FORMED* DAN MENGIKUTI ATURAN (DTD), MAKA AKAN MENJADI DOKUMEN YANG SYAH.

6. **Struktur dokumen XML**

   Aturan yang ditetapkan untuk struktur dokumen XML adalah dokumen harus formal dan ringkas. Setiap rancangan dokumen XML memiliki *logical structure* dan *phisical structure*

   a. ***Logical structure*** mendefinisikan **unit dan subunits** dari *data container* (elemen-elemen), datatype, atribut.
   b. ***Phisical structure*** mendefinisikan data yang akan diletakkan didalam elemen seperti text, image, atau media yang lain seperti ditetapkan didalam logical strucure.

7. **Well-formed** adalah bagian dari logical structure. Sebagai contoh setiap dokumen berisi satu, dan hanya satu root, atau elemen dokumen yang membuka dan menutup dokumen.
8. **DTD adalah** kunci untuk arti elemen , atribut, dan context. DTD, *Data Type Definition* diperlukan agar dokumen valid. DTD adalah sehimpunan aturan yang secara eksplisit menentukan :

   a. Nama (node root)
   b. Isi (elemen root)
   c. Context dari setiap elemen

   Sehingga DTD dasar dari sebuah dokumen XML. DTD diperlukan agar sebuah dokumen menjadi valid.