

# Pengantar XML

Yudi Wibisono ([yudi@upi.edu](mailto:yudi@upi.edu))

FPMIPA UPI

versi dok: 2.0 / Feb 09

# Contoh XML

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<daftar_pengarang>
  <pengarang>
    <nama>budi martami</nama>
    <alamat>sarijadi </alamat>
  </pengarang>
  <pengarang>
    <nama>elfan noviari</nama>
    <alamat>gerlong</alamat>
  </pengarang>
</daftar_pengarang>
```

# Mengapa Belajar XML?

- Standard de-facto untuk bertukar data, menyimpan data, mendeskripsikan data.
- Di-support oleh banyak tools (RDBMS, bahasa pemrograman)
- Webservice → XML pada HTTP
- Platform independen, language independen.

# Apa XML?

- XML = **EX**tensible **M**arkup **L**anguage
- Markup language untuk dokumen yang mengandung informasi terstruktur.
- Ditujukan untuk pertukaran dokumen via web.
- Berbentuk file teks → Cross platform, indepedenden terhadap software dan hardware.

# Aplikasi XML

- Mendeskripsikan dokumen.
- Pertukaran data
- Database

# Isi XML

- Tag: `<nama> budi </nama>`
  - Tag didefinisikan sendiri.
- Tag dapat mempunyai atribut
  - `<daftar_mahasiswa jumlah="30"> ... </daftar_mahasiswa>`
- Struktur → hirarki (tag dalam tag)  
`<mahasiswa><nama> budi </nama></mahasiswa>`

# XML dan HTML

- Persamaan dengan HTML?
  - Sama-sama markup language
- Perbedaan dengan HTML?
  - Semantik HTML telah didefinisikan. Contoh: `<b>` untuk bold
  - XML lebih ketat aturannya. Contoh: setiap tag harus ada penutupnya (`<tag> .... </tag>`)
  - XML case sensitive

# Keuntungan XML

- Self Documenting → dengan melihat tag, dapat diketahui isi dokumen.
- Dapat dibaca software dan manusia.
- Fleksibel
- Dapat dikembangkan tanpa melanggar format lama.
  - Contoh: pada <pengarang>, tambah tag <telepon>, maka XML yang lama tetap dapat dibaca.



# Keuntungan XML (lanj)

- Hirarkis → dapat merepresentasikan data kompleks
- Independen terhadap bahasa pemrograman, OS

# Kerugian XML

- Pengulangan tag → tidak efisien, ukuran membengkak

# Beberapa Aturan XML

- Setiap tag harus ada penutupnya.
- Penamaan Tag
  - case sensitive. <nama> tidak sama dengan <Nama>.
  - Tidak diawali dengan angka
  - Tidak mengandung spasi
  - Hindari '-' dan '.'
- Urutan hirarki harus benar.

## Contoh **yang salah**:

<mahasiswa> <nama> yudi </mahasiswa></nama>

# Aturan XML (lanj)

- Setiap XML harus mengandung root (akar)

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

- Setiap atribut harus dalam tanda petik. Contoh:

```
<surat tanggal = "12/12/2007">
```

```
<catatan oleh = " Jum'at " >
```

```
<catatan oleh = ' Budi "si cepat" Martami '>
```

- Komentar dalam XML:

```
<!-- ini komentar lho --!>
```

# Elemen dan Atribut

- Elemen: tag, isi, sampai tag penutup.

`<tag> isi tag </tag>`

Elemen

- atribut vs elemen, mana yang lebih baik?

```
<surat tanggal = "2/2/2005">  
  <pengirim>yudi</pengirim>  
</surat>
```

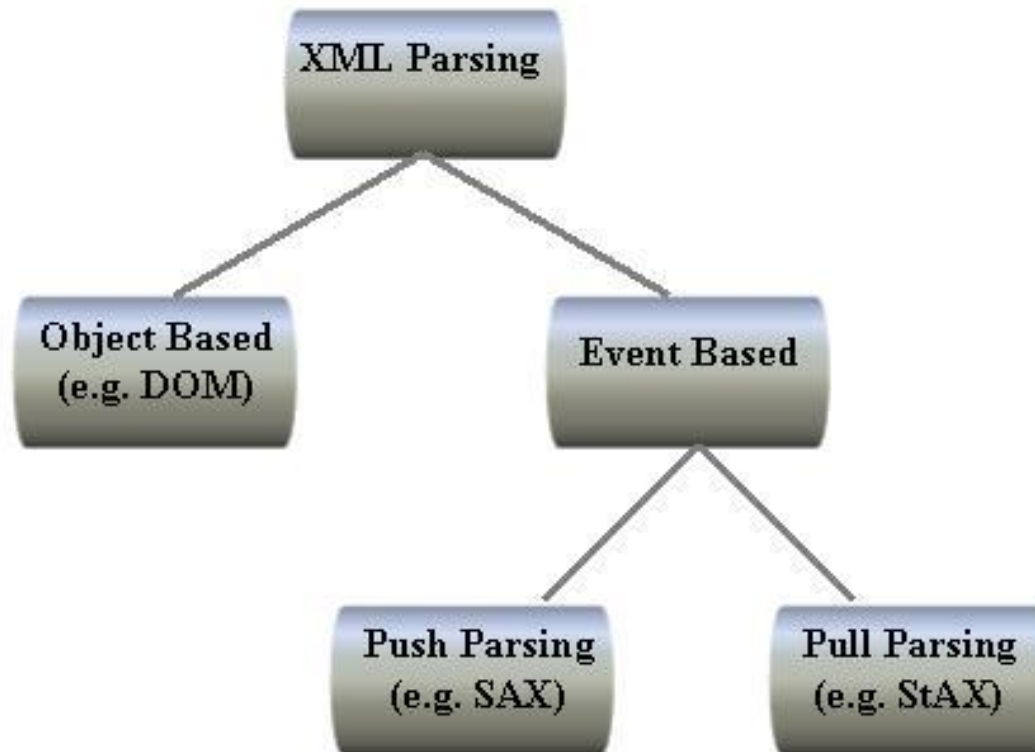
## ATAU

```
<surat>  
  <tanggal> 2/2/2005 </tanggal>  
  <pengirim> Yudi </pengirim>  
</surat>
```

# XML Parser

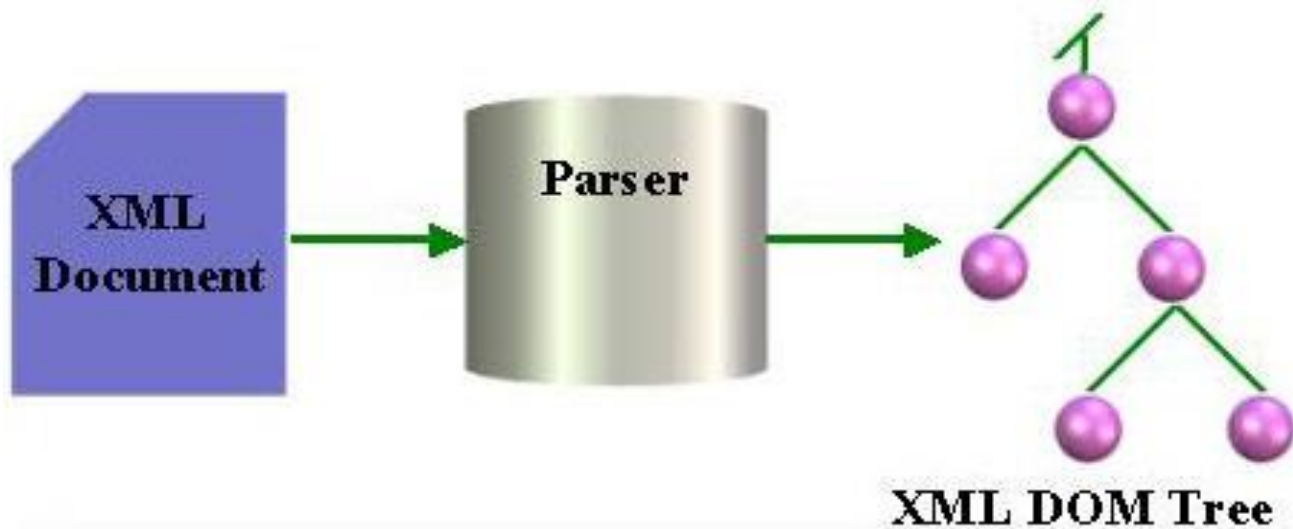
- Parser → program untuk “membaca” dokumen XML
- Dua cara:
  - Tree based, isi XML dipindahkan ke memori terlebih dulu → DOM (Document Object Model).
  - Event based → tidak dipindahkan ke memori. Fungsi callback dipanggil setiap menemui element. → SAX (Simple API XML)

# XML Parser



<http://www.xml.com/2005/07/06/graphics/image001.jpg>

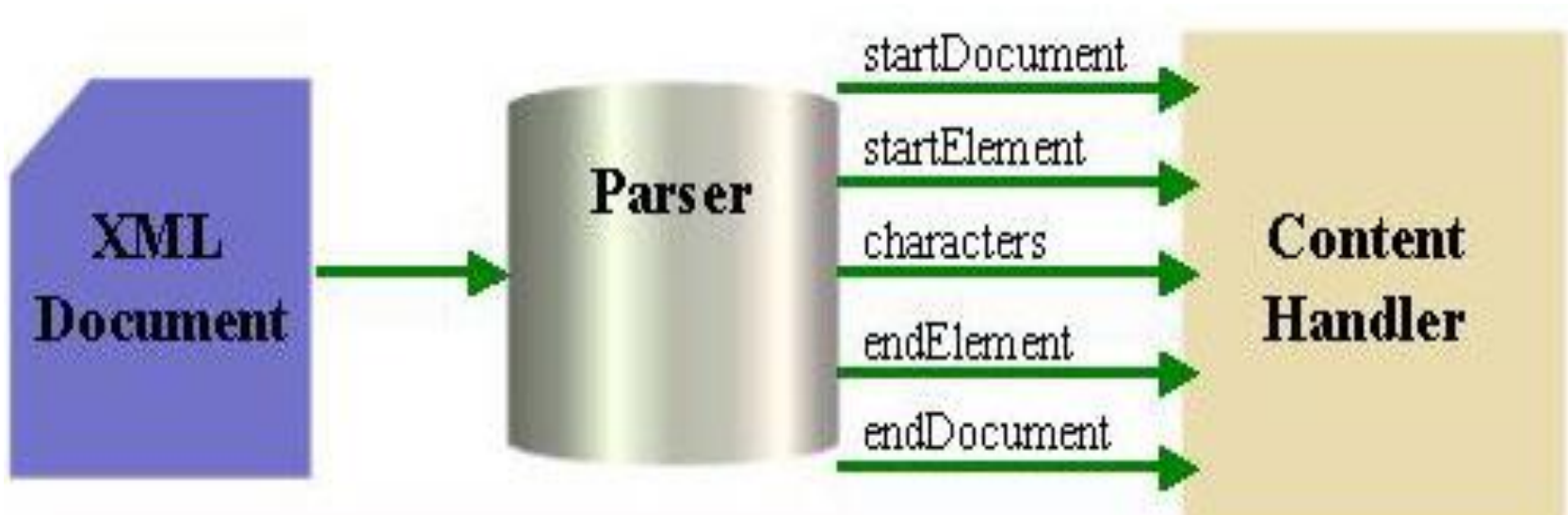
# DOM



<http://www.xml.com/2005/07/06/graphics/image003.jpg>



# SAX: Event Based



<http://www.xml.com/2005/07/06/graphics/image002.jpg>

# Spesifikasi

- DTD → Document Type Declaration
  - Definsi Tag dan atribut
- XML Scheme → lebih general dari DTD
- XSL → eXtensible Style Language  
Bagaimana XML ditampilkan

# DTD: Document Type Declaration

- Dari 1970-an
- Well Formed XML → XML yang mengikuti aturan:
  - Mempunyai root
  - Setiap element mempunyai penutup
  - Case sensitive
  - Nilai atribut dalam tanda petik ( “ ”)
  - Tersusun dengan benar (nested properly)

**Wellformed belum tentu VALID!**

**Agar dapat dikatakan VALID harus mengikuti DTD!**

# DTD (2)

```
<?xml version="1.0"?>
<!DOCTYPE note [
    <!ELEMENT note (to,from,heading,body)>
    <!ELEMENT to (#PCDATA)>
    <!ELEMENT from (#PCDATA)>
    <!ELEMENT heading (#PCDATA)>
    <!ELEMENT body (#PCDATA)> ]>

<note>
    <to>Akhmad</to>
    <from>Budi</from>
    <heading>UTS</heading>
    <body>Jangan Lupa UTS!</body>
</note>
```

**DTD mendefinisikan elemen yang legal dalam sebuah dokumen XML → mengapa menggunakan DTD?**

# DTD Construct

- ELEMENT
- ATTLIST
- ENTITY
- NOTATION

# DTD (2)

## DTD Sebagai file external:

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
<!DOCTYPE note SYSTEM "InternalNote.dtd">  
<note>  
  <to>Akhmad</to>  
  <from>Budi</from>  
  <heading>UTS</heading>  
  <body>Jangan Lupa UTS!</body>  
</note>
```

# DTD: Element

```
<!ELEMENT Keranjang ( Jeruk, (Apel | Jambu)) >
```

Maka contoh XML-nya yang valid :

```
<Keranjang>  
  <Jeruk> </Jeruk>  
  <Apel></Apel>  
</Keranjang>
```

```
<Keranjang>  
  <Jeruk></Jeruk>  
  <Jambu></Jambu>  
</Keranjang>
```

# DTD: Element (kardinalitas)

```
<!ELEMENT Keranjang ( Jeruk+, (Apel | Jambu)* ) >
```

Contoh XML yang valid:

```
<Keranjang>  
  <Jeruk> </Jeruk>  
</Keranjang>
```

```
<Keranjang>  
  <Jeruk></Jeruk>  
  <Jambu></Jambu>  
  <Apel></Apel>  
  <Apel></Apel>  
</Keranjang>
```

? : Opsional, bisa ada atau tidak \*  
: Nol atau banyak  
+ : Satu atau banyak



# DTD: ATRIBUT

Contoh 1:

```
<!ATTLIST Keranjang warna CDATA "hitam" #IMPLIED>
```

XML-nya:

```
<Keranjang warna = "biru"> ... </Keranjang>
```

atau

```
<Keranjang> </Keranjang>
```

Implied artinya tidak wajib, jika kosong menggunakan nilai default yaitu "hitam"

Contoh 2:

```
<!ATTLIST Buku ISBN ID #REQUIRED>
```

(ID artinya ISBN harus unik dan REQUIRED artinya atribut ini wajib ada)

# DTD: Entitas

- Shortcut dari teks yang umum (konstanta)

```
<!ENTITY bm "Budi Martami">
```

di XML-nya gunakan &entity;

```
<Pengarang> &bm; </Pengarang>
```

# DTD:Notation

- Untuk data selain XML

```
<!NOTATION jpg SYSTEM "jpgviewer.exe">
```

```
<!ATTLIST Image type NOTATION (gif|jpg)>
```

XML:

```
<Image type ="jpg"> .... </image>
```

# Newspaper Article DTD

Copied from <http://www.vervet.com/>

```
<!DOCTYPE NEWSPAPER [  
  
<!ELEMENT NEWSPAPER (ARTICLE+)>  
<!ELEMENT ARTICLE (HEADLINE,BYLINE,LEAD,BODY,NOTES)>  
<!ELEMENT HEADLINE (#PCDATA)>  
<!ELEMENT BYLINE (#PCDATA)>  
<!ELEMENT LEAD (#PCDATA)>  
<!ELEMENT BODY (#PCDATA)>  
<!ELEMENT NOTES (#PCDATA)>  
  
<!ATTLIST ARTICLE AUTHOR CDATA #REQUIRED>  
<!ATTLIST ARTICLE EDITOR CDATA #IMPLIED>  
<!ATTLIST ARTICLE DATE CDATA #IMPLIED>  
<!ATTLIST ARTICLE EDITION CDATA #IMPLIED>  
  
<!ENTITY NEWSPAPER "Vervet Logic Times">  
<!ENTITY PUBLISHER "Vervet Logic Press">  
<!ENTITY COPYRIGHT "Copyright 1998 Vervet Logic Press">  
  

```

```

<!DOCTYPE CATALOG [
<!ENTITY AUTHOR "John Doe">
<!ENTITY COMPANY "JD Power Tools, Inc.">
<!ENTITY EMAIL "jd@jd-tools.com">

<!ELEMENT CATALOG (PRODUCT+)>

<!ELEMENT PRODUCT
(SPECIFICATIONS+,OPTIONS?,PRICE+,NOTES?)>
<!ATTLIST PRODUCT
NAME CDATA #IMPLIED
CATEGORY (HandTool|Table|Shop-Professional) "HandTool"
PARTNUM CDATA #IMPLIED
PLANT (Pittsburgh|Milwaukee|Chicago) "Chicago"
INVENTORY (InStock|Backordered|Discontinued) "InStock">

<!ELEMENT SPECIFICATIONS (#PCDATA)>
<!ATTLIST SPECIFICATIONS
WEIGHT CDATA #IMPLIED
POWER CDATA #IMPLIED>

<!ELEMENT OPTIONS (#PCDATA)>
<!ATTLIST OPTIONS
FINISH (Metal|Polished|Matte) "Matte"
ADAPTER (Included|Optional|NotApplicable) "Included"
CASE (HardShell|Soft|NotApplicable) "HardShell">

<!ELEMENT PRICE (#PCDATA)>
<!ATTLIST PRICE
MSRP CDATA #IMPLIED
WHOLESALE CDATA #IMPLIED
STREET CDATA #IMPLIED
SHIPPING CDATA #IMPLIED>

<!ELEMENT NOTES (#PCDATA)>

]>

```

# Contoh DTD (lanjutan) Katalog Produk

# Latihan membuat DTD

- Buatlah DTD XML untuk menyimpan data nilai mahasiswa
- Buatlah DTD XML untuk menyimpan data fasilitas pada suatu hotel

# Kelemahan DTD?

- Sintaksnya bukan format XML standard
- Kurang lengkap
- Tidak fleksibel
- Setiap orang dapat membuat tag sendiri yang mungkin bentrok
- Reuse DTD? sulit.
- Tidak mensupport strong typing (integer, string date)

Solusi terbaru: XML Schema

# XML Schema

- Dari tahun 2001
- Menggunakan format XML
- Calon kuat pengganti DTD di masa depan
- Dikenal juga dengan nama
  - XSD: XML Schema Document
  - WXS: W3C XML Schema
- Organisasi lebih baik (namespace)
- Strong typing → 44 tipe (DTD hanya 10)
- Lebih presisi dan fleksibel (Custom datatype)
  - Contoh: dapat mendefinisikan tipe umur :  $0 < \text{umur} < 120$
  - Dapat mendefinisikan urutan child
- Compatible dengan teknologi XML yang lain: Web Service, Xquery



# Perbandingan DTD vs XML Schema

(dari <http://www.adp-gmbh.ch>)

## With DTD

```
<?xml version="1.0" ?>

<!DOCTYPE name
[
  <!ELEMENT name      (title?, first_name, middle_name?, last_name, suffix?) >
  <!ELEMENT title      (#PCDATA) >
  <!ELEMENT first_name  (#PCDATA) >

```

DTD

```
<?xml version="1.0" encoding="UTF-8"?>

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

```

```
  <xsd:element name="uuml" type="xsd:token" fixed="ü"/>

```

```
  <xsd:complexType name="name">

```

```
    <xsd:sequence>

```

```
      <xsd:element name="title"          type="xsd:string" minOccurs="0" maxOccurs="1" />

```

```
      <xsd:element name="first_name"     type="xsd:string" minOccurs="1" maxOccurs="1" />

```

```
      <xsd:element name="middle_name"    type="xsd:string" minOccurs="0" maxOccurs="1" />

```

```
      <xsd:element name="last_name"      type="xsd:string" minOccurs="1" maxOccurs="1" />

```

```
      <xsd:element name="suffix"         type="xsd:string" minOccurs="0" maxOccurs="1" />

```

```
    </xsd:sequence>

```

```
  </xsd:complexType>

```

```
</xsd:schema>
```

XML Schema

# Struktur Schema

- Pembukaan (Preamble)
- Definisi tipe (simple, kompleks) dan deklarasi elemen
- Content model??
- Model Group??

# Namespace

- Mencegah ambiguity dan name collision. Contoh `<presiden>` → presiden negara? presiden perusahaan? presiden mahasiswa?
- Format: `xmlns:XML_name_space= URI`.

`xmlns:buku = “ http://www.upi.edu/buku.xsd “`

`xmlns:drama = “ http://www.upi.edu/drama.xsd “`

`<buku:pengarang> ... </buku:pengarang>`

`<drama:pengarang> ... </drama:pengarang>`

# Definisi Type

- type = “[jenis tipe]”  
Jenis tipe: date, string, integer, double
- <tgl type = “date”> 2004-03-11 <tgl>
- Format tipe “date” adalah YYYY-MM-DD

# Definisi Custom Tipe & Elemen: Simple

Tipe umur dimana  $0 < \text{umur} \leq 120$ :

```
<xsd:simpleType name="tipeUmur">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minExclusive value="0"/>
    <xsd:maxInclusive value="120"/>
  </xsd:restriction>
</xsd:simpleType>

<xsd:element name="umur" type="tipeUmur"/>
```

Tanggal di bulan Mei YYYY – 05 - DD:

```
<xsd:simpleType name="may_date">
  <xsd:restriction base="date">
    <xsd:pattern value="\d{4}-05-\d{2}"/>
  </xsd:restriction>
</xsd:simpleType>
```

# Pembukaan

```
<?xml version="1.0"?>
```

```
< xsd:schema
```

```
  xmlns:xsd="http://www.w3.org/2001/XMLSchema ">
```

```
...
```

```
</xsd:schema>
```

**Root: “schema”.**

**xsd adalah namespace**

# Deklarasi Element

```
<element name="kodepos" type = "string" />  
<element name="kecepatan" type ="float" />
```

# Contoh Definisi Tipe, Elemen, Atribut: Tipe Kompleks

## **XML :**

```
<note ID="5">  
  <to>Akhmad</to>  
  <date>2005-12-30</date>  
</note>
```

## **DTD :**

```
<!ELEMENT note (to,date)>  
<!ELEMENT to (#PCDATA)>  
<!ELEMENT date (#PCDATA)>  
<!ATTLIST note ID #REQUIRED>
```



# Definisi Tipe, Elemen, Atribut: Tipe Kompleks (Lanj)

## XML Schema:

```
<xsd:element name = "note">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name = "to" type = "xsd:string"/>
      <xsd:element name = "date" type = "xsd:date"/>

    </xsd:sequence>
    <attribute name = "ID" use="required"
type="integer"/>
  </xsd:complexType>
</xsd:element>
```

# Definisi Tipe :

## Tipe Kompleks (lanj)

DTD :

```
<!ELEMENT Keranjang ( Jeruk+, Apel* ) >
```

XML Scheme :

```
<xsd:element name = "Keranjang">  
  <xsd:complexType>  
    <xsd:sequence>  
      <xsd:element name = "Jeruk" type "xsd:string" minOccurs = "1"  
        maxOccurs="unbounded"/>  
      <xsd:element name = "Apel" type "xsd:string" minOccurs = "0"  
        maxOccurs="unbounded"/>  
    </xsd:sequence>  
  </xsd:complexType>  
</xsd:element>
```

# Definisi Tipe :

## Tipe Kompleks (choice)

DTD :

```
<!ELEMENT Keranjang ( Jeruk |Apel) >
```

XML Scheme :

```
<xsd:element name = "Keranjang">  
  <xsd:complexType>  
    <xsd:choice>  
      <xsd:element name = "Jeruk" type "xsd:string"/>  
      <xsd:element name = "Apel" type "xsd:string"/>  
    </xsd:choice>  
  </xsd:complexType>  
</xsd:element>
```

# Definisi Tipe :

## Tipe Kompleks (group

```
<!ELEMENT Keranjang ( Jeruk+, (Apel | Jambu)* )>
```

XML Scheme:

```
<xsd:group name="apeljambu">
  <xsd:choice>
    <xsd:element name="Apel" type="xsd:string"/>
    <xsd:element name="Jambu" type="xsd:string"/>
  </xsd:choice>
</xsd:group>

<xsd:element name = "Keranjang">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name = "Jeruk" type "xsd:string" minOccurs = "1"
        maxOccurs="unbounded />
      <xsd:group ref = "apeljambu" minOccurs = "0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

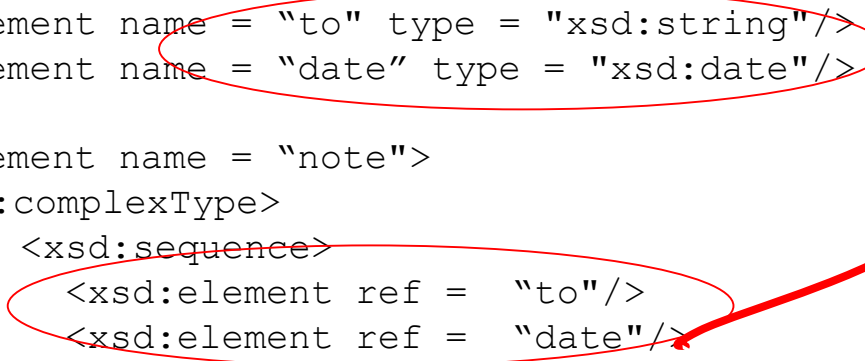
# Penggunaan Ref

```
<xsd:element name = "note">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name = "to" type = "xsd:string"/>
      <xsd:element name = "date" type = "xsd:date"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

dapat ditulis:

```
<xsd:element name = "to" type = "xsd:string"/>
<xsd:element name = "date" type = "xsd:date"/>

<xsd:element name = "note">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref = "to"/>
      <xsd:element ref = "date"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```



# Contoh XML dan Schema-nya

(<http://www.xml.com/pub/a/2000/11/29/schemas/part1.html?page=1>)

## XML

```
<?xml version="1.0" encoding="UTF-8"?>
<book isbn="0836217462">
  <title>
    Being a Dog Is a Full-Time Job
  </title>
  <author>Charles M. Schulz</author>
  <character>
    <name>Snoopy</name>
    <friend-of>Peppermint Patty</friend-of>
    <since>1950-10-04</since>
    <qualification>
      extroverted beagle
    </qualification>
  </character>
  <character>
    <name>Peppermint Patty</name>
    <since>1966-08-22</since>
    <qualification>bold, brash and tomboyish</qualification>
  </character>
</book>
```

## Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" type="xs:string"/>
        <xs:element name="author" type="xs:string"/>
        <xs:element name="character" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="friend-of" type="xs:string" minOccurs="0"
                maxOccurs="unbounded"/>
              <xs:element name="since" type="xs:date"/>
              <xs:element name="qualification" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="isbn" type="xs:string"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

```

<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" type="xs:string"/>
        <xs:element name="author" type="xs:string"/>
        <xs:element name="character" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="friend-of" type="xs:string" minOccurs="0"
                maxOccurs="unbounded"/>
              <xs:element name="since" type="xs:date"/>
              <xs:element name="qualification" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="isbn" type="xs:string"/>
    </xs:complexType>
  </xs:element>
</xs:schema>

```

# Contoh XML dan XML-Schema(dari <http://lucas.ucs.ed.ac.uk/xml-schema/>)

a sample instance document: file [landrover.xml](#)

```
<?xml version = "1.0" encoding = "UTF-8"?>
<vehicles
  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation = "http://lucas.ucs.ed.ac.uk/xml-schema/vehicles/landrover.xsd">

  <vehicle>
    <nickname>Count Zero</nickname>
    <model>Series I, 80</model>
    <construction>
      <start>
        <dom>21</dom>
        <month>July</month>
        <year>1949</year>
      </start>
      <end>
        <dom>9</dom>
        <month>August</month>
        <year>1949</year>
      </end>
    </construction>
    <mods>
      <mod>Change Engine</mod>
      <mod>Change pedals</mod>
      <mod>Change gearbox</mod>
      <mod>Fit Rollcage</mod>
    </mods>
  </vehicle>
</vehicles>
```

The Landrover schema: file [landrover.xsd](#)

```
<?xml version = "1.0" encoding = "UTF-8"?>
<xsd:schema xmlns:xsd = "http://www.w3.org/2001/XMLSchema">

  <xsd:element name = "vehicles">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref = "vehicle" maxOccurs = "unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name = "vehicle">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name = "nickname" type = "xsd:string" maxOccurs = "1"/>
        <xsd:element name = "model" type = "xsd:string"/>
        <xsd:element name = "construction">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element ref = "start"/>
              <xsd:element ref = "end"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name = "mods">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name = "mod" type = "xsd:string" maxOccurs = "unbounded"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name = "start">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref = "dom"/>
        <xsd:element ref = "month"/>
        <xsd:element ref = "year"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name = "end">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref = "dom"/>
        <xsd:element ref = "month"/>
        <xsd:element ref = "year"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name = "dom" type = "xsd:string"/>
  <xsd:element name = "month" type = "xsd:string"/>
  <xsd:element name = "year" type = "xsd:string"/>
</xsd:schema>
```

Schema hal 1

hal 2



```
<?xml version="1.0" encoding="UTF-8"?>

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:element name="uuml" type="xsd:token" fixed="ü"/>

  <xsd:complexType name="name">
    <xsd:sequence>
      <xsd:element name="title" type="xsd:string" minOccurs="0" />
      <xsd:element name="first_name" type="xsd:string" minOccurs="1" />
      <xsd:element name="middle_name" type="xsd:string" minOccurs="0" />
      <xsd:element name="last_name" type="xsd:string" minOccurs="1" />
      <xsd:element name="suffix" type="xsd:string" minOccurs="0" />
    </xsd:sequence>
  </xsd:complexType>

</xsd:schema>
```