

Web Service

Oleh:

M N Kamal F

POLITEKNIK POS INDONESIA

BANDUNG

2013

BAB I

TIK:

Mahasiswa dapat Memahami dasar-dasar layanan web (web service)

Pokok Bahasan:

- Pendahuluan
- Pengantar Web Service

Sub Pokok Bahasan:

- Definisi
- Arsitektur
- WS-Inspection document format
- Keuntungan Web service

1 Definisi

Web service merupakan sistem software yang dirancang untuk mendukung kerjasama antar mesin ke mesin melalui jaringan, interfacenya berupa format yang dapat dikenali oleh mesin khususnya Web Service Definition Language (WSDL). Sistem yang lain berinteraksi dengan web service menggunakan pesan Simple Object Access Protocol (SOAP), biasanya disampaikan menggunakan HTTP dengan serialisasi XML dalam hubungannya dengan standar Web-terkait lainnya. Webservice adalah layanan yang akan menjalankan tugas/aksi tertentu dapat diakses melalui web (HTTP).

Perlu dibahas mengenai Service Oriented Architecture (SOA). SOA merupakan paradigma arsitektur yang berfokus pada pembangunan sistem melalui penggunaan beragam web service ataupun arsitektur service lainnya dan menghubungkannya sehingga menjadi sistem yang utuh.

SOA tidak perlu harus didasarkan pada web services. SOA adalah sebuah konsep arsitektur. Web Services adalah realisasi SOA, yang memanfaatkan XML dan protokol umum internet (seperti HTTP) untuk menyebarkan suatu layanan. Secara teori, Anda bisa dengan mudah membangun sebuah SOA menggunakan sesuatu selain Web Services.

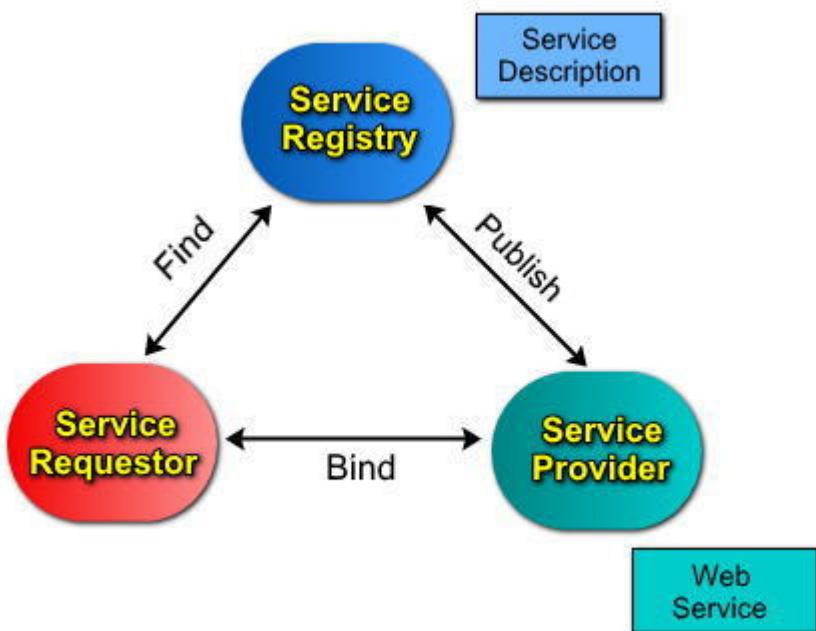
2 Arsitektur

Arsitektur web service didasarkan pada interaksi antara tiga peran utama:

- penyedia service,
- service yang terdaftar, dan
- pemohon service

Peran-peran ini berinteraksi menggunakan pempublikasian, pencarian, dan operasi binding (pengikatan). Penyedia service adalah bisnis yang menyediakan akses ke Web service dan mempublish deskripsi service dalam daftar service. Pemohon service menemukan deskripsi service dalam daftar service dan

menggunakan informasi dalam deskripsi untuk mengikat service tersebut. Pandangan logis dari arsitektur layanan Web ditunjukkan pada Gambar 1. Dari gambar arsitektur layanan Web, daftar service menyediakan lokasi terpusat untuk menyimpan deskripsi service. Sebuah daftar Universal Description, Discovery and Integration (UDDI) adalah contoh dari jenis service yang terdaftar.



Gambar 1

Terdapat kebutuhan akan metode pencarian service yang terdistribusi. Web Services Inspection Language merupakan metode yang digunakan sebagai standar pencarian tersebut, dengan cara memeriksa situs web akan ketersediaan web service. Spesifikasi WS-Inspection menentukan lokasi situs web dimana kita dapat mencari service yang diinginkan.

3 WS-Inspection document format

Sebuah inspeksi dokumen WS menyediakan referensi untuk deskripsi layanan.

Deskripsi layanan ini dapat didefinisikan dalam format seperti WSDL, UDDI, atau HTML biasa.

Sebuah inspeksi dokumen WS dapat berisi daftar referensi untuk deskripsi layanan, serta referensi untuk inspeksi dokumen WS lainnya. Sebuah dokumen WS akan berisi satu atau lebih <service> dan elemen <link>.

<service> akan berisi satu atau lebih referensi untuk berbagai jenis deskripsi layanan pada web service yang sama.

<link> mungkin berisi referensi ke hanya satu jenis deskripsi layanan, tapi deskripsi layanan ini tidak memerlukan referensi web service yang sama.

Perlu diperhatikan perbedaan antara web (aplikasi web) dengan web service.

Aplikasi web adalah aplikasi yang berada pada server, tetapi ditujukan untuk digunakan oleh manusia, yang menggunakan halaman web sebagai lapisan presentasi. Semua interaktivitas pengguna (GUI) dilakukan melalui halaman web, tetapi semua data disimpan dan (kebanyakan) dimanipulasi pada server.

Web service adalah aplikasi berbasis server (seperti di atas) yang dapat diakses melalui web melalui protokol HTTP, tetapi ditujukan terutama untuk interaksi dengan program lain. Dengan demikian, terdapat API yang memberikan tanggapan terhadap HTTP GET dan POST yang dibuat oleh aplikasi remote. Tidak berarti Anda tidak dapat mengakses layanan web dari browser Anda, tetapi aplikasi itu tidak akan selalu memiliki user interface GUI. Biasanya Anda menerima semua hasil GET dan POST sebagai string XML, yang membutuhkan parser di sisi klien.

Kapan Anda perlu menggunakan Web service?

Web Services itu digunakan saat diperlukan transformasi suatu bisnis logik yang terpisah dalam 1 ruang lingkup yang menjadi satu. Web Service cukup diupload ke Web Server dan siap diakses oleh pihak-pihak yang telah diberikan otorisasi.

Web service berjalan (live) di port 80 yang merupakan protokol standar HTTP, dengan demikian mengurangi resiko terblokir oleh firewall. Kendala arsitektur COM/DCOM adalah memerlukan konfigurasi khusus di sisi firewall, dan ini tidak perlu dilakukan ketika mengakses Web Service.

Beberapa vendor luar negeri mulai berkolaborasi satu sama lain dengan konsep web services, diantaranya : IBM , Microsoft , SUN , ORACLE.

4 Keuntungan Web service

4.1 Loosely Coupled

Setiap service berdiri sendiri, independen terhadap service lain pada suatu aplikasi. Bagian-bagian service dapat diubah tanpa mempengaruhi yang lain

4.2 Ease of Integration

Data dibungkus antar aplikasi dan WS berfungsi untuk menghubungkan aplikasi-aplikasi tersebut dan memudahkan komunikasi

4.3 Service Reuse

Kode dapat digunakan kembali dan dikembangkan lebih mudah baik di sisi klien atau server.

Kaitan dengan logistik

Dengan pertumbuhan B2B yang semakin banyak dan cepat dibutuhkan komunikasi dinamis diantaranya (dapat berupa lintas platform dsb), web service adalah sarana yang tepat untuk menjawabnya.

TUGAS:

Buat artikel kelompok: tentang sejarah web service dan penggunaanya, beri contoh real perusahaan/instansi yang menggunakannya.

BAB II

TIK:

Mahasiswa dapat Memahami dasar-dasar XML

Pokok Bahasan:

- Pendahuluan XML

Sub Pokok Bahasan:

- Definisi
- Blok-blok pembangun XML
- XML Namespaces
- XML Schema

Dalam web service biasanya paket data yang dikomunikasikan berupa format XML.

1 Definisi

Extensible Markup Language (XML) adalah rekomendasi W3C untuk membuat markup languages yang memungkinkan strukturisasi, deskripsi dan pertukaran data.

XML bagian dari Standard Generalized Markup Language (ISO 8879:1986 SGML) yang mampu menggambarkan beragam bentuk data. XML memfasilitasi berbagai teks terstruktur dan informasi pada database dan internet.

Bahasa yang berbasiskan XML dijelaskan dalam bentuk yang formal dan memungkinkan program untuk memodifikasi dan memvalidasi dokumen tanpa perlu mengetahui bentuk awalnya dan bersifat independen platform.

2 Blok-blok pembangun XML

- Elements

Pasangan dari tag awal dan tag akhir.

- Attributes

Sepasang nama yang merupakan bagian dari tag awal sebuah Elemen.

- Processing Instructions

Deskripsi arahan khusus kepada aplikasi yang akan memproses XML dokumen.

- Comments

Tag atau pesan yang membantu user memahami kode sumber.

- Character Data

- Characters (encoding)
- Entities
- Whitespace

2.1 Elemen-elemen XML

Elemen merupakan nama teknis untuk tag yang berpasangan dimulai dari tag pembuka dan tag penutup pada dokumen XML.

Contoh:

```
<Mahasiswa>  
</Mahasiswa>
```

Tag pembuka: <.....>

Tag penutup: </.....>

Catatan:

- Elemen XML harus benar-benar bersarang
- Nama elemen bisa berupa huruf, garis bawah, tanda hubung dan koma, harus dimulai dengan huruf.
- Nama elemen bersifat case sensitif

Contoh: yang benar

```
<Mahasiswa>      </Mahasiswa>
```

Contoh: yang salah

```
<Mahasiswa>  
  </Nama>  
  </Mahasiswa>  
</Nama>
```

```
<.Mahasiswa>    <./Mahasiswa>
```

2.2 Atribut

Merupakan komponen yang berfungsi memberikan informasi tambahan terhadap elemen.

Contoh:

```
<Mahasiswa status="aktif">
    <Nama>budi</Nama>
</Mahasiswa>
```

```
<Mahasiswa status="aktif"/>
```

2.3 Processing Instructions

Sebuah direktif khusus untuk pengolahan dokumen XML.

```
<? ....?>
```

Contoh:

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

2.4 Comments

Suatu teks yang membantu user dalam membaca maksud XML atau program dan diabaikan dalam prosesnya.

Contoh:

```
<!-- contoh dokuemnn ini berisi data mahasiswa-->
<Mahasiswa></Mahasiswa>
```

2.5 Character Data

- Encoding

Semua karakter dalam dokumen XML harus mengikuti pengkodean dokumen

- Entities

Komponen yang berfungsi untuk menampilkan output lain pada browser

Contoh:

Entity	karakter
<	<
>	>
&	&
"	"
'	'

- Whitespace

Kebanyakan aplikasi XML mengabaikan spasi

Contoh: XML (sumber: Introduction to Web Services, Ioannis G. Baltopoulos)

```
<?xml version="1.0" encoding="UTF-8"?>
<message from="yiannis" to="family">
    <text>Hey, I'm at the iCSC!
    </text>
    <!-- Attachment is optional -->
    <attachment>
        <desc>Photo from Geneva</desc>
        <item>
            <?BinaryDataStart ?>
            0100100001010001001010010
            <?BinaryDataEnd ?>
        </item>
    </attachment>
</message>
```

3 XML Namespaces

XML Namespace menggunakan Uniform Resource Identifier untuk membuat kelas-kelas elemen.

Contoh: (sumber: Introduction to Web Services, Ioannis G. Baltopoulos)

```
<msg:message from="yiannis" to="family"
  xmlns:msg="http://www.w2c.com/ns/email"
  xmlns:po="http://www.w2c.com/ns/purchase">
  <msg:text>
    <msg:desc>A Purchase Order</msg:desc>
    <msg:item>
      <po:order>
        <po:item>
          <po:desc>Laptop Computer</po:desc>
          <po:price>1300 GBP</po:price>
        </po:item>
      </po:order>
    </msg:item>
  </msg:text>
</msg:message>
```

Perhatikan contoh di atas, berikut pembangun kelas-kelas elemennya:

- xmlns:msg
- xmlns:po

Menambahkan prefiks ke setiap elemen dalam dokumen XML menyulitkan pembacaan dan ukuran dokumen meningkat. Oleh karena itu, XML namespace memungkinkan kita untuk menggunakan namespace default pada dokumen. Elemen default namespace tidak memerlukan prefiks.

4 XML Schema

XML schema berfungsi untuk:

- Identifikasi elemen-elemen yang ada pada dokumen
- Identifikasi urutan dan hubungan antara elemen-elemen
- Identifikasi atribut setiap elemen, dan apakah mereka opsional atau dibutuhkan atau memiliki beberapa properti khusus lainnya
- Identifikasi datatype konten atribut

Kaitan dengan logistik

Pertukaran data/dokumen pada proses B2B secara umum menggunakan format XML SGML dan mungkin masih akan digunakan jika faktor kecepatan masih dapat diterima.

TUGAS:

Buat dokumen XML berisi tentang proses bisnis suatu instansi, contoh: pendidikan, industri, jasa, tracking dlsb.

Anda dapat menggunakan alat bantu editor seperti eclipse dlsb.

BAB III

TIK:

Mahasiswa dapat Memahami DTD pada XML

Pokok Bahasan:

- DTD

Sub Pokok Bahasan:

- Pendahuluan
- Pendeklarasian DTD
- Komposisi XML dilihat dari DTD
- DTD XML

1 Pendahuluan

Pada materi sebelumnya Anda telah mengenal dan dapat membuat dokumen XML, yang perlu diperhatikan adalah bagaimana struktur dari dokumen tersebut agar konsisten, oleh karena itu diperlukan definisi terhadap dokumen XML tersebut. Pertama akan dibahas mengenai Document Type Definitions (DTD).

- DTD berfungsi untuk mendefinisikan tipe dokumen XML.
- DTD ditulis untuk menjelaskan elemen dan entitas yang mungkin muncul di dalam dokumen dan elemen isi serta atributnya.

Sehingga dokumen XML dapat divalidasi dan dicek mengenai kondisi well form nya dengan membandingkannya dengan definisi pada DTD.

2 Pendeklarasian DTD

Dapat dideklarasikan di dalam file xml

```
<!DOCTYPE nama-root-element [deklarasi-element]>
```

Ataupun diluar file xml

```
<!DOCTYPE nama-root-element SYSTEM "filename">
```

3 Komposisi XML dilihat dari DTD

Dokumen XML memiliki:

- Elements
- Attributes
- Entities

3.1 Elements

Elements merupakan komponen utama dari suatu dokumen xml.

Memiliki tag pembuka dan penutup

Contoh:

```
<Mahasiswa>  
  
</Mahasiswa>
```

3.2 Attributes

Merupakan komponen yang berfungsi memberikan informasi tambahan terhadap element

contoh:

```
<Mahasiswa Nama="Maman">  
  
</Mahasiswa>
```

3.3 Entities

Komponen yang berfungsi untuk menampilkan output lain pada browser

contoh:

Entity	karakter
<	<
>	>
&	&
"	"
'	'

4 DTD XML

Dengan mengacu pada struktur xml di atas maka dapat dibentuk file DTD dengan aturan

4.1 DTD: Elements

- Syntax:

```
<!ELEMENT element-name category>
```

Atau

```
<!ELEMENT element-name (element-content)>
```

- Empty Elements

```
<!ELEMENT element-name EMPTY>
```

contoh:

```
<!ELEMENT br EMPTY>
```

XML:

```
<br />
```

- Elements with any Contents

```
<!ELEMENT element-name ANY> contoh: <!ELEMENT siaranTV ANY>
```

- Elements with Children (sequences)

```
<!ELEMENT element-name (child1)>
```

atau

```
<!ELEMENT element-name (child1,child2,...)>
```

contoh:

```
<!ELEMENT siaranTV (chanel,jam,nama,tema)>
```

- Elements with Parsed Character Data (PCDATA)

```
<!ELEMENT element-name (#PCDATA)>
```

contoh:

```
<!ELEMENT binatang(#PCDATA)>
```

Untuk contoh elements yang bertingkat (mempunyai anak), pada penulisan xml harus ditulis berurutan sesuai pada DTD. Dan pada format DTD, elemen-elemen anak tadi harus dideklarasikan mendetail.

Contoh:

```
<!ELEMENT siaranTV (chanel,jam,nama,tema)>

<!ELEMENT chanel (#PCDATA) >

<!ELEMENT jam(#PCDATA) >

<!ELEMENT nama(#PCDATA) >

<!ELEMENT tema(#PCDATA) >
```

Penulisan jumlah anak dari suatu elemen dapat dituliskan dengan menambahkan tanda pada akhir nama elemen.

Tanda	Arti
+	Muncul satu kali atau lebih. Minimal muncul satu kali.
*	Muncul 0 kali atau lebih
?	Boleh tidak muncul, tetapi jika muncul maksimal satu kali.
	Fungsi atau

Contoh:

```
<?xml version="1.0"?>
<!DOCTYPE catatan[
<!ELEMENT catatan(pada+)>
<!ELEMENT pada (#PCDATA)>
]>
<catatan>
<pada>pp</pada>
<pada>pp1</pada>
</catatan>
```

4.2 DTD: Elements

Syntax:

```
<!ATTLIST element-name attribute-name attribute-type default-value>
```

contoh DTD:

```
<!ATTLIST mahasiswa Nim CDATA "-">
```

XML:

```
<mahasiswa Nim="112233" />
```

4.3 DTD – Entities

Entities adalah variabel digunakan untuk mendefinisikan shortcut untuk teks standar atau karakter khusus.

Entitas dapat dideklarasikan secara internal maupun eksternal.

Contoh:

DTD :

```
<!ENTITY copyright "PoltekPos">
```

XML:

<author>©right;</author>

PRAKTIK:

Untuk memudahkan praktikum, Anda dipersilahkan memakai tools untuk mempelajari mata kuliah ini. Anda disarankan menggunakan:

- Eclipse JEE IDE (termasuk wtp plugin)
- Plugin Eclipse Rinzo xml editor

Studi kasus:

Anda memiliki perusahaan logistik jasa pengantar barang (PT X). Anda memiliki banyak klien. Anda dan klien berkomunikasi melalui pertukaran dokumen xml.

Struktur dokumen sebagai berikut:

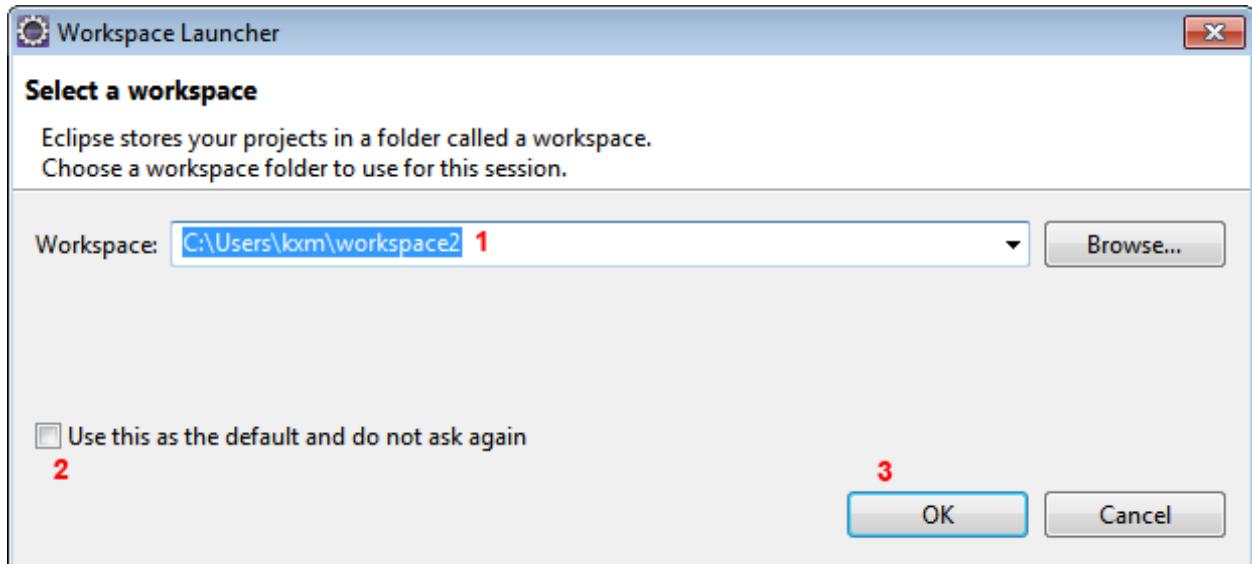
- barang
 - kode
 - nama
 - satuan
 - harga
 - asal
 - PT
 - kodeWIL
 - tujuan
 - PT
 - kodeWIL

Buatlah contoh file xml nya dan DTD.

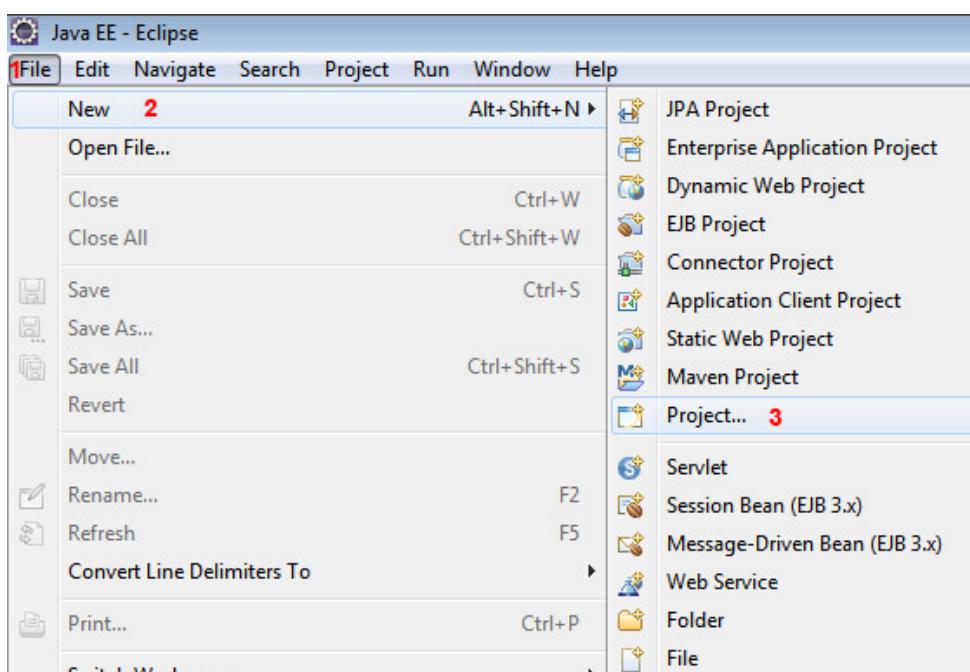
1. File xml dapat diintegrasikan dengan berbagai project, untuk keperluan praktikum ini kita akan membuat project tidak utuh, hanya membuat file xml dan DTD dengan tools dan plugin eclipse, kemudian memeriksa apakah dokumen tersebut valid, well form.
2. Jalankan eclipse JEE IDE



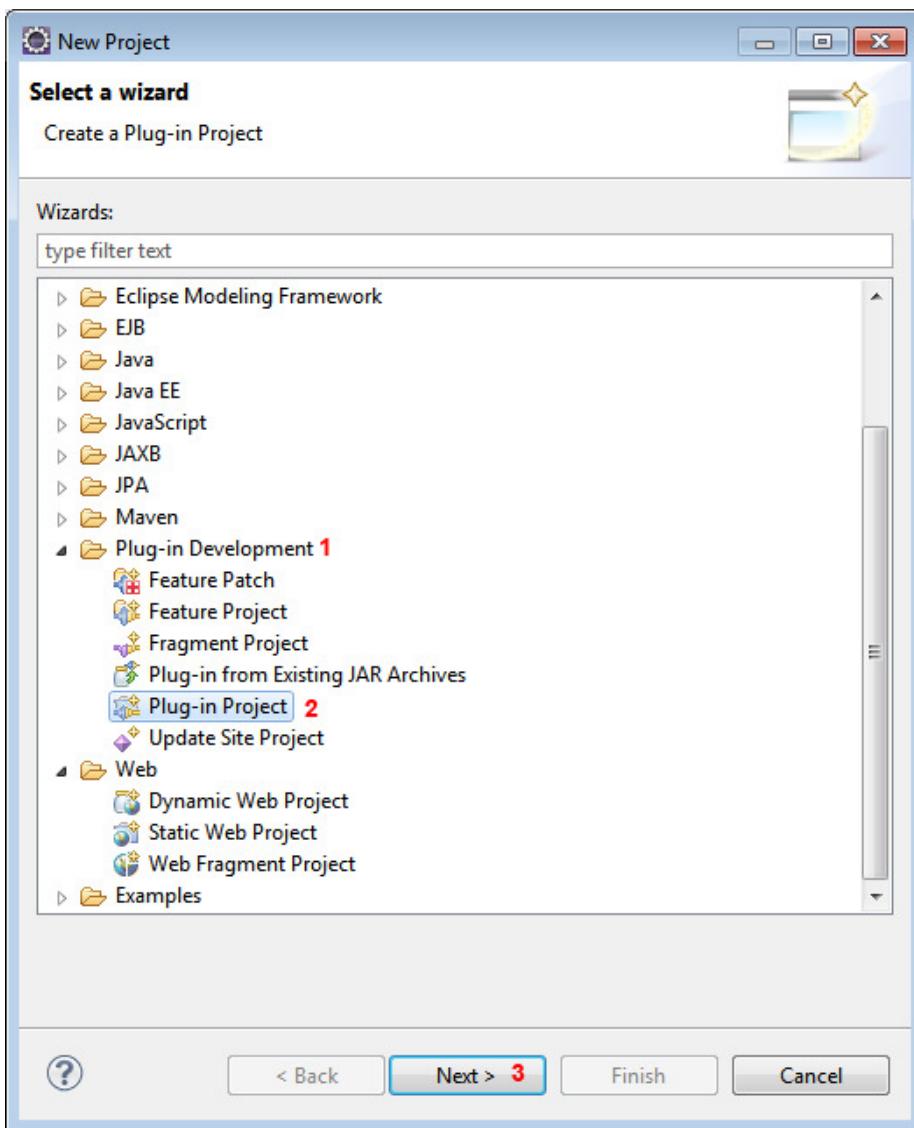
3. Tentukan workspace, tempat project Anda akan dikerjakan. Anda dapat menentukannya atau megikuti default workspace



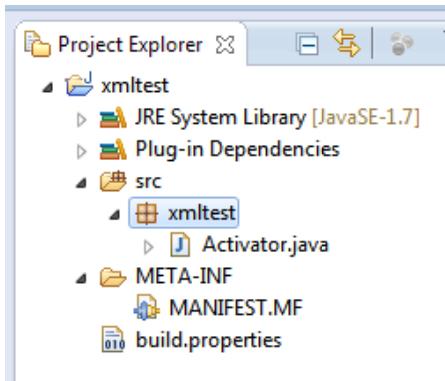
4. Ingat, kita tidak membuat program atau aplikasi, hanya membuat file xml dan DTD dengan eclipse. Dalam hal ini kita akan membuat project plug-in development.
5. Klik file>new>project



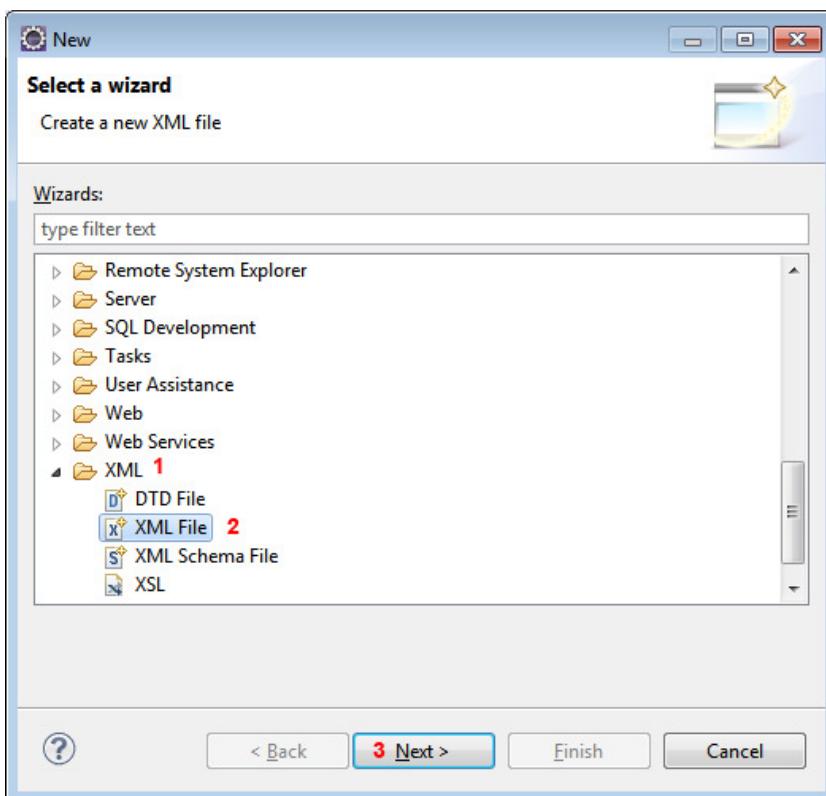
6. Pilih plug-in project



7. Beri nama project: xmltest
8. Klik next>finish
9. Perhatikan pada project explorer, struktur projectnya.



10. Perlu diperhatikan bahwa suatu perusahaan mengolah data dalam bentuk database system seperti (oracle, sql server, mysql dsb) kemudian hasil olahan tadi disimpan dalam bentuk xml yang akhirnya dipertukarkan antar B2B. Kemudian untuk memastikan bahwa data yang dikirim adalah benar dilampirkan definisinya atau DTD. Maka dalam praktikum ini diasumsikan bahwa output bermula dari olahan database adalah file xml.
11. Pada project explorer, fold: src/xmltest. Klik kanan
12. Pilih New>other...
13. XML file dan klik next



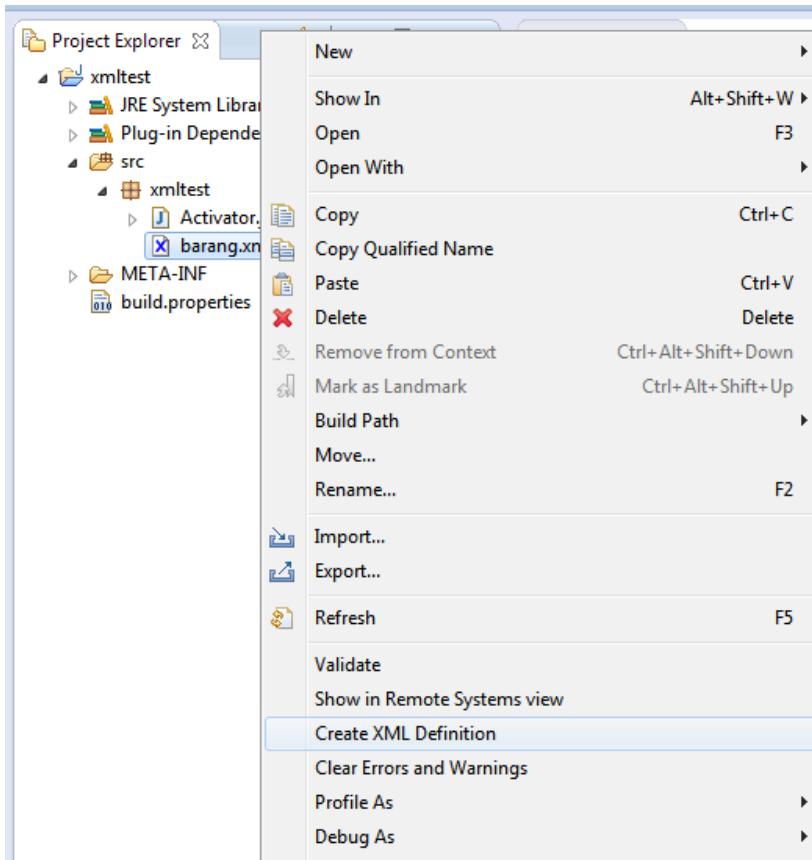
14. Beri nama: barang.xml dan klik finish
15. Berdasarkan struktur dokumen di atas, kita asumsikan dokumen xml nya sebagai berikut: (ingat aturan format xml, nested, casesensitif dll)
Isikan file barang.xml sebagai berikut:

```
<?xml version="1.0" encoding="UTF-8"?>
<logbarang>
<barang>
    <kode>M1112</kode>
    <satuan>pc</satuan>
    <harga>5000</harga>
    <asal>
        <pt>ladyrock</pt>
        <kodewil>10</kodewil>
    </asal>
    <tujuan>
        <pt>union</pt>
        <kodewil>1001</kodewil>
    </tujuan>
</barang>

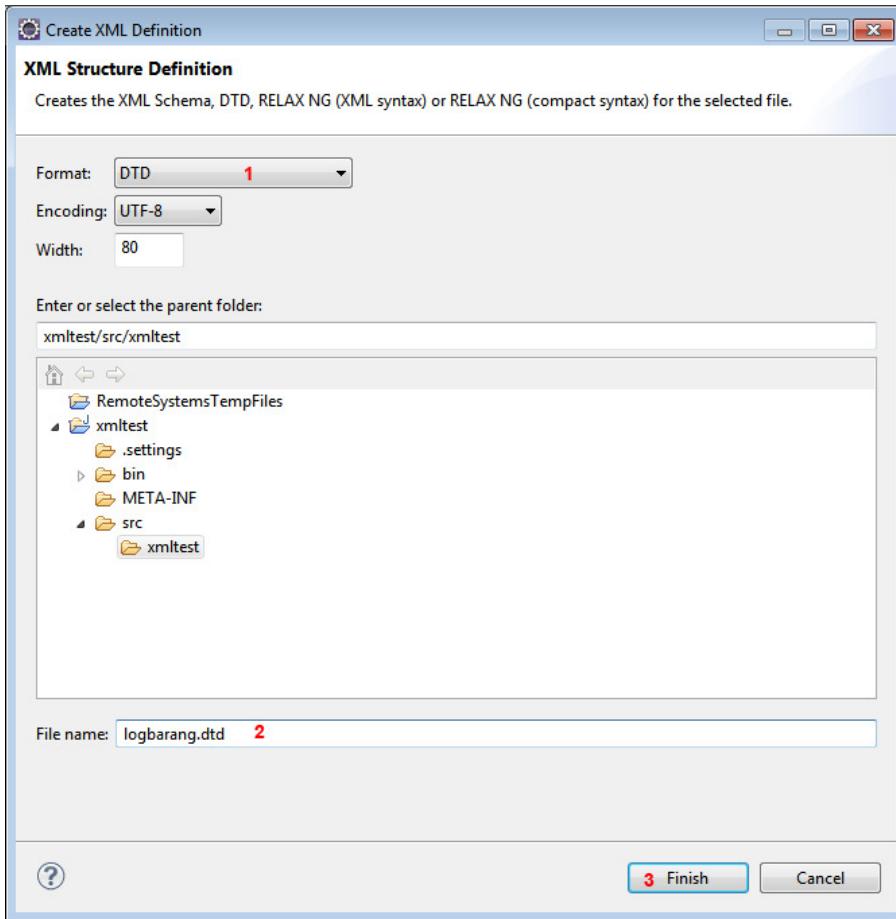
<barang>
    <kode>M1122</kode>
    <satuan>pc</satuan>
    <harga>1000</harga>
    <asal>
        <pt>pmb</pt>
        <kodewil>103</kodewil>
    </asal>
    <tujuan>
        <pt>mitra kencana</pt>
        <kodewil>300</kodewil>
    </tujuan>
</barang>

</logbarang>
```

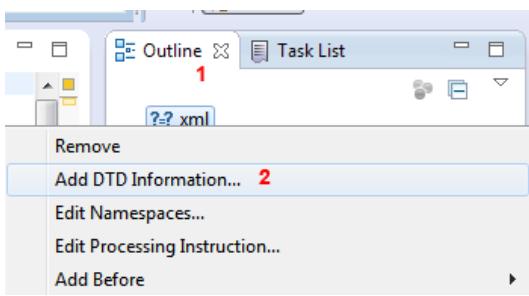
16. Jika Anda menggunakan plugin Rinzo xml editor, pada project explorer, klik kanan file barang.xml dan pilih Create XML Definition



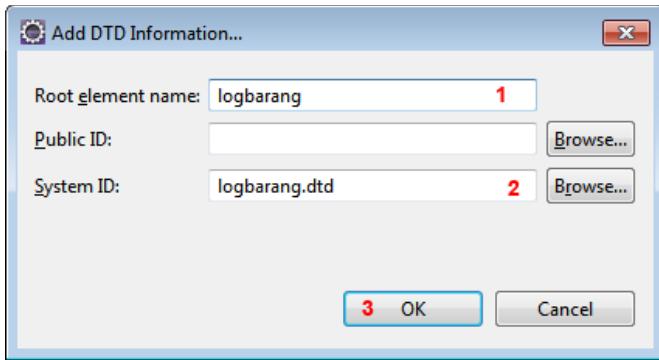
17. Pilih format DTD, beri nama logbarang.dtd



18. Perhatikan struktur logbarang.dtd tersebut
19. Untuk menghubungkan definisi (DTD) dengan xml. Anda bisa menuju project explorer, pada file barang.xml, klik kanan>open with>pilih xml editor.
Catatan: xml editor akan muncul jika Anda menginstal plugin wtp pada eclipse
20. Pada outline>klik kanan>add DTD information



21. Isikan root element: logbarang dan System ID: logbarang.dtd

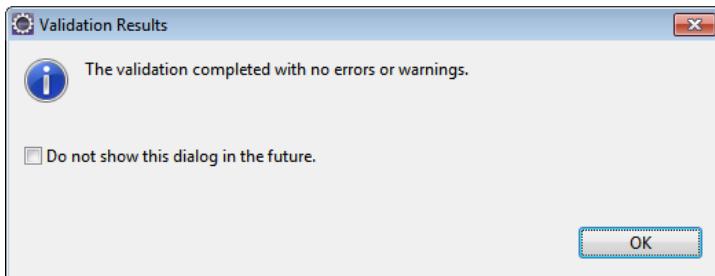


22. Perhatikan header file barang.xml

```
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE logbarang SYSTEM "logbarang.dtd">
```

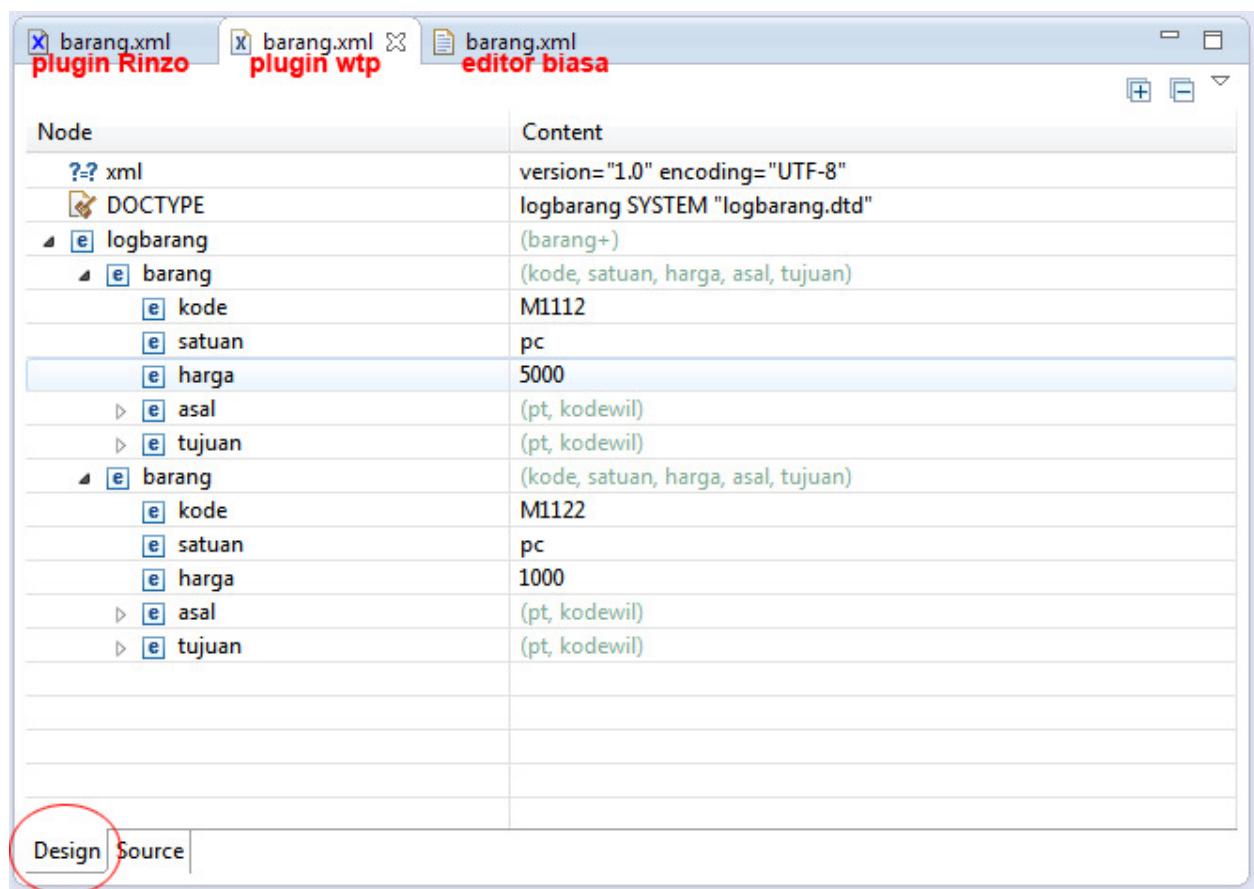
23. Untuk memeriksa validasi, pada jendela Outline>klik kanan>validate

24. Jika valid muncul



25. Untuk debug jika terdapat kesalahan, Anda dapat melihatnya pada jendela Markers

26. Jika Anda menggunakan plugin wtp dalam membangun xml dan definisinya, tampilannya kurang lebih seperti berikut:



TUGAS:

Buatlah file xml dan DTD berdasarkan kebutuhan perusahaan yang bergerak di bidang:

- Industri manufakturing
- Industri jasa
- Industri perhotelan
- Pendidikan

Isi data pada masing-masing file xml minimal 5, libatkan juga penggunaan atribut, minimal elemen 10.

BAB IV

TIK:

Mahasiswa dapat Memahami dasar-dasar layanan Schema

Pokok Bahasan:

- XSD schema

Sub Pokok Bahasan:

- Definisi
- Fungsi dan keunggulan
- XSD tipe Elemen
 - Elemen Kosong
 - Tipe kompleks
 - Mixed Content
 - Controlling Type Derivation

1 Definisi

XML schema menjelaskan / mendefinisikan struktur dari dokumen xml. Schema dapat menjelaskan batasan-batasan yang lebih kompleks pada elemen dan atribut dan struktur definisi mengarah pada pendekatan berbasis objek

2 Fungsi dan keunggulan

Kegunaannya, mendefinisikan:

- elemen dan atribut yang akan muncul pada dokumen xml
- urutan dan banyaknya elemen child
- apakah nilai dari suatu elemen, kosong atau berisi
- Tipe data untuk elemen dan atribut
- Nilai default dan nilai yang tetap pada elemen dan attribut

Beberapa keunggulan schema dari DTD:

- mendukung tipe data
- menggunakan sintaks xml
- Lebih aman dalam hal komunikasi data

3 XSD tipe Elemen

Elemen kompleks memiliki elemen-elemen atau atribut-atribut lainnya. Jenis complex elements:

- Elemen kosong
- Elemen yang hanya memiliki elemen anak
- Elemen berisi nilai teks
- Elemen yang memiliki elemen anak dan berisi nilai teks

File XML Schema biasanya disimpan dengan ekstensi *.xsd.

Namespace

Untuk penulisan namespace pada schema kita dapat menuliskan langsung pada elemen yang akan kita beri namespace. Misalnya kita ingin menamai sebuah elemen, maka kita dapat menuliskannya dengan :

```
<?xml version="1.0"?>
<mahasiswa xmlns:mhs="mahasiswa politeknik pos informatika">
<mhs:nim>11111</mhs:nim>
<mhs:nama>Budi</mhs:nama>
</mahasiswa>
```

Prefiks di atas adalah: mhs

Namespace: mahasiswa politeknik pos informatika

Struktur xml menjadi salah

```
<?xml version="1.0"?>
<mahasiswa xmlns:mhs="mahasiswa politeknik pos informatika">
<mhs:nim>11111</mhs:nim>
<mhs:nama>Budi</mhs:nama>
<coba:alamat>ddd</coba:alamat>
</mahasiswa>
```

Perhatikan prefiks coba.

3.1 Elemen Kosong

Penulisan elemen kosong adalah dengan cara tidak mengisikan elemen kedalam deklarasi. Misalnya pada contoh berikut kita katakan bahwa elemen heading adalah kosong, maka format penulisan untuk XML Schema adalah seperti berikut :

Xml:

```
<?xml version="1.0"?>
<mahasiswa>
<nim>11111</nim>

<nama/><!--elemen nama merupakan elemen kosong-->
</mahasiswa>
```

Maka schema yang didapatkan adalah:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
  <xs:element name="mahasiswa">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="nim" type="xs:string"/>
        <xs:element name="nama"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Perhatikan pada name="nama" tidak didapati type.

3.2 Tipe kompleks

Pada tipe ini dapat terlihat pendekatan berdasarkan objek:

Xml:

```
<?xml version="1.0"?>
<mahasiswa>
  <nim>11111</nim>
  <nama>Budi</nama>
  <ortu>
    <ayah>Iwan</ayah>
    <ibu>Wati</ibu>
  </ortu>
</mahasiswa>
```

Maka xsd nya:

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified">
    <xs:element name="mahasiswa">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="nim" type="xs:string" />
                <xs:element name="nama" type="xs:string" />
                <xs:element name="ortu">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="ayah" type="xs:string" />
                            <xs:element name="ibu" type="xs:string" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>

```

3.3 Mixed Content

Dapat terlihat penggabungan antara atribut dan elemen

xml:

```

<?xml version="1.0"?>
<mahasiswa_polpos_2020>
<mahasiswa aktif="tidak">
<nim>11111</nim>
<nama>Budi</nama>
<ortu>
    <ayah>Iwan</ayah>
    <ibu>Wati</ibu>
</ortu>
</mahasiswa>
<mahasiswa aktif="ya">
<nim>11122</nim>
<nama>Rani</nama>
<ortu>
    <ayah>Iwan</ayah>
    <ibu>Wati</ibu>
</ortu>
</mahasiswa>
</mahasiswa_polpos_2020>

```

Xsd:

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
  <xs:element name="mahasiswa_polpos_2020">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" ref="mahasiswa"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="mahasiswa">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="nim"/>
        <xs:element ref="nama"/>
        <xs:element ref="ortu"/>
      </xs:sequence>
      <xs:attribute name="aktif" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:NCName">
            <xs:pattern value="ya|tidak"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:element name="nim" type="xs:integer"/>
  <xs:element name="nama" type="xs:NCName"/>
  <xs:element name="ortu">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="ayah"/>
        <xs:element ref="ibu"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="ayah" type="xs:NCName"/>
  <xs:element name="ibu" type="xs:NCName"/>
</xs:schema>

```

3.4 Controlling Type Derivation

Atribut final dapat ditambahkan pada sebuah definisi tipe kompleks untuk mengatur *#all*, *extension*, atau *restriction*. Apabila suatu tipe berasal dari tipe

lain yang memiliki atribut final, *schema processor* memverifikasi. Contoh penulisan :

perhatikan xsd di atas pada atribut:aktif terdapat restriction pengisian hanya boleh string ya atau tidak.

PRAKTIK

Studi kasus:

Anda memiliki perusahaan logistik jasa pengantar barang (PT X). Anda memiliki banyak klien. Anda dan klien berkomunikasi melalui pertukaran dokumen xml.

Struktur dokumen sebagai berikut:

- barang
 - kode
 - nama
 - satuan
 - harga
 - asal
 - PT
 - kodeWIL
 - tujuan
 - PT
 - kodeWIL

Buatlah contoh file xml nya dan schema xsd.

1. File xml dapat diintegrasikan dengan berbagai project, untuk keperluan praktikum ini kita akan membuat project tidak utuh, hanya membuat file xml dan xsd dengan tools dan plugin eclipse, kemudian memeriksa apakah dokumen tersebut valid, well form.
2. Buat baru
3. Klik file>new>project >plug-in project
4. Beri nama: xmltest_1
5. Klik next>finish
6. Perhatikan pada project explorer, struktur projectnya.
7. Perlu diperhatikan bahwa suatu perusahaan mengolah data dalam bentuk database system seperti (oracle, sql server, mysql dsb) kemudian hasil olahan

tadi disimpan dalam bentuk xml yang akhirnya dipertukarkan antar B2B. Kemudian untuk memastikan bahwa data yang dikirim adalah benar dilampirkan definisinya atau xsd. Maka dalam praktikum ini diasumsikan bahwa output bermula dari olahan database adalah file xml.

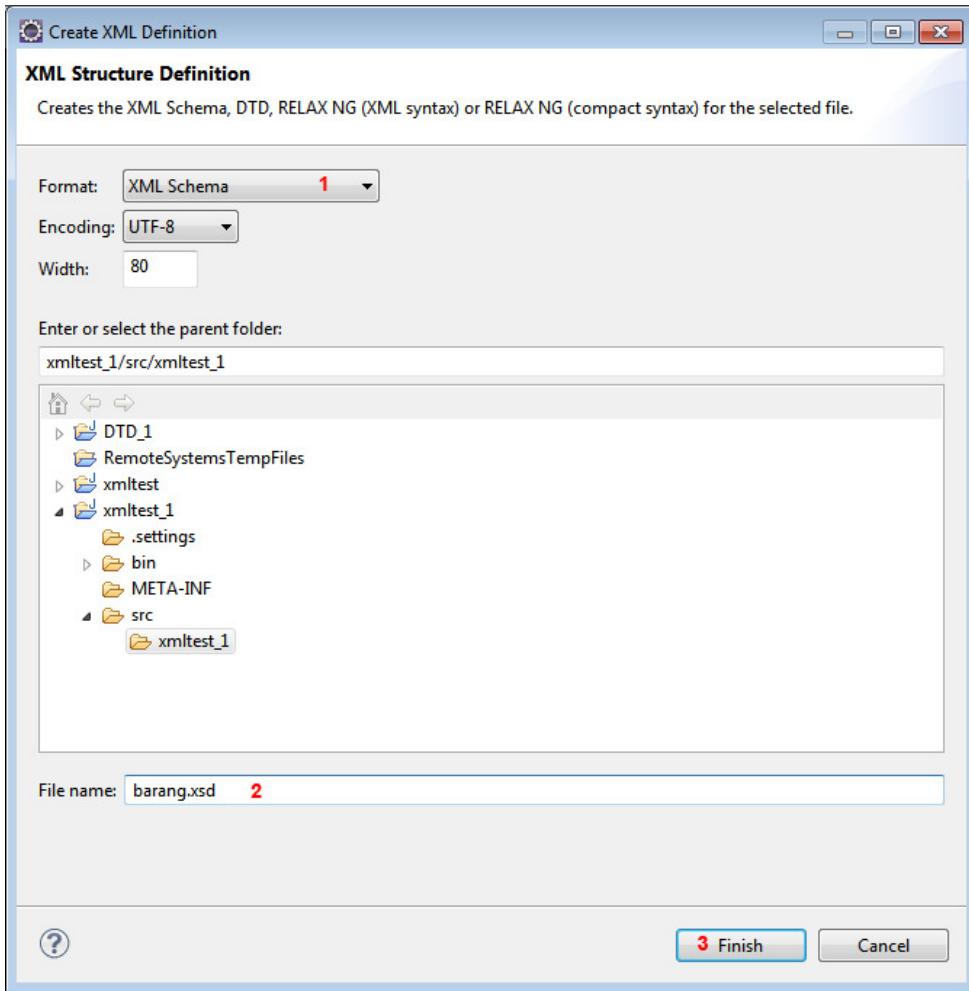
8. Pada project explorer, fold: src/xmltest_1. Klik kanan
9. Pilih New>other...
10. XML file dan klik next
11. Beri nama: barang.xml dan klik finish
12. Berdasarkan struktur dokumen di atas, kita asumsikan dokumen xml nya sebagai berikut: (ingat aturan format xml, nested, casesensitif dll)
13. Isikan file barang.xml sebagai berikut:

```
<?xml version="1.0" encoding="UTF-8"?>
<logbarang>
<barang>
    <kode>M1112</kode>
    <satuan>pc</satuan>
    <harga>5000</harga>
    <asal>
        <pt>ladyrock</pt>
        <kodewil>10</kodewil>
    </asal>
    <tujuan>
        <pt>union</pt>
        <kodewil>1001</kodewil>
    </tujuan>
</barang>

<barang>
    <kode>M1122</kode>
    <satuan>pc</satuan>
    <harga>1000</harga>
    <asal>
        <pt>pbm</pt>
        <kodewil>103</kodewil>
    </asal>
    <tujuan>
        <pt>mitra kencana</pt>
        <kodewil>300</kodewil>
    </tujuan>
</barang>

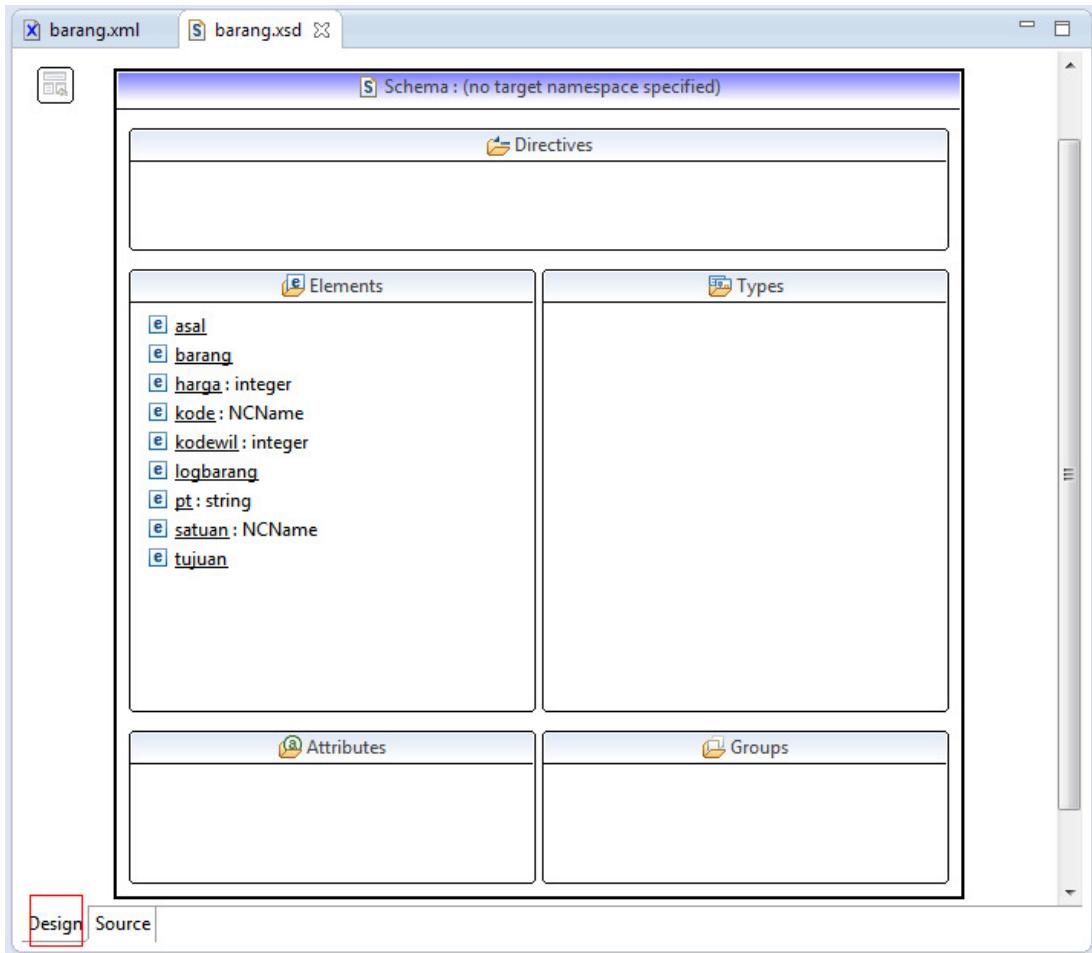
</logbarang>
```

14. Jika Anda menggunakan plugin Rinzo xml editor, pada project explorer, klik kanan file barang.xml dan pilih Create XML Definition



15. Beri nama barang.xsd, klik finish

16. Dapat dilihat design schema



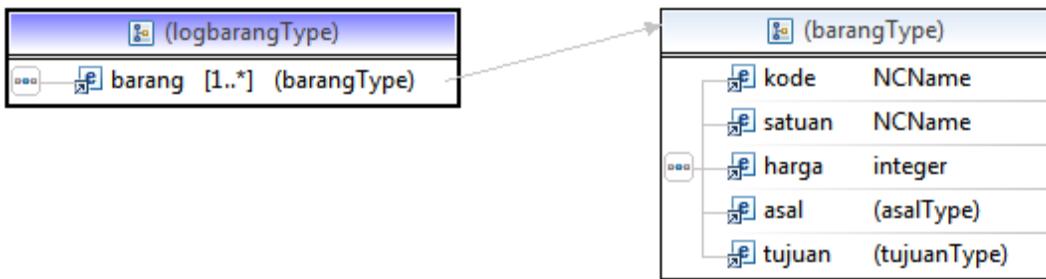
17. Anda dapat memeriksa sesuai struktur yang terlihat pada xml

18. Klik pada elemen logbarang



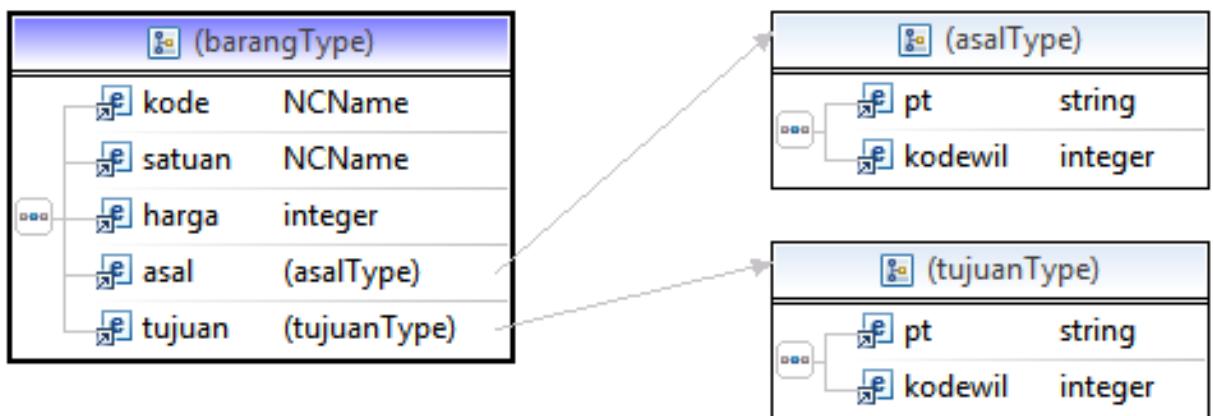
Perhatikan pada jendela properties, Maximum Occurrence: unbounded. Untuk selanjutnya Anda dapat memodifikasi banyaknya elemen yang boleh muncul melalui jendela properties ini.

19. Klik pada '(logbarangType)



Dapat terlihat elemen pembentuk barang, sesuai pada struktur xml

20. Klik pada (barangType)



21. Sampai disini dapat dilihat bahwa struktur schema mengikuti pendekatan berbasis objek.

22. Buatlah pattern

- untuk kode tidak boleh kosong
- Harga harus diatas 1000

23. Anda dapat memeriksa valid/tidaknya dokumen xml dengan cara seperti pada bab sebelumnya

TUGAS:

Buatlah file xml dan xsd berdasarkan kebutuhan perusahaan yang bergerak di bidang:

- Industri manufacturing
- Industri jasa
- Industri perhotelan
- Pendidikan

Isi data pada masing-masing file xml minimal 5, libatkan juga penggunaan atribut, pattern (contoh: pola untuk email abc@xyz.com) minimal elemen 10.

BAB V

TIK:

Mahasiswa dapat Memahami dasar-dasar SAX dan DOM

Pokok Bahasan:

- SAX, DOM

Sub Pokok Bahasan:

- Pendahuluan
- SAX
- DOM

1 Pendahuluan

Terdapat beberapa cara untuk membaca dan memanipulasi dokumen XML. Cara yang umum adalah dengan metode

- Simple Api for XML (SAX) dan
- Document Object Model (DOM)

Bahasa pemrograman telah menyediakan kemampuan untuk ini.

2 SAX

SAX dikembangkan oleh W3C dan dirilis pada tahun 1998. Parser berbasis SAX memanggil metode ketika suatu markup (misalnya tag awal, tag akhir, dll) ditemukan. Tidak ada struktur pohon yang diciptakan. Data diteruskan ke aplikasi dari dokumen XML secara berurut. Parser SAX biasanya digunakan untuk membaca dokumen XML yang tidak akan diubah.

Hasil parsing SAX tidak disimpan dalam memori sehingga lebih ringan ketika digunakan untuk membaca dokumen XML yang besar. Karena tidak disimpan dalam memori maka SAX tidak memiliki kemampuan untuk mengolah data XML. Cara SAX bekerja dengan berjalan secara sequential dari awal hingga akhir dan memanggil fungsi event handler untuk setiap jenis elemen yang ditemui dalam file XML.

Parser berbasis SAX tersedia untuk berbagai bahasa pemrograman, C++, Java, dan Perl yang paling popular. SAX didasarkan pada model event driven menggunakan call backs untuk menangani pengolahan. Berikut hal yang dapat dilakukan program dengan metode SAX:

- Mencari pada dokumen suatu elemen yang mengandung kata kunci
- Mencetak format konten
- Memodifikasi dokumen XML dengan membuat perubahan kecil, seperti memperbaiki ejaan dan elemen penggantian nama
- Membaca data untuk membangun struktur data yang kompleks

Tetapi program tidak dapat:

- Re-order elemen dalam dokumen
- Menyelesaikan cross-references antar elemen
- Verifikasi ID-IDREF Link
- Validasi dokumen XML

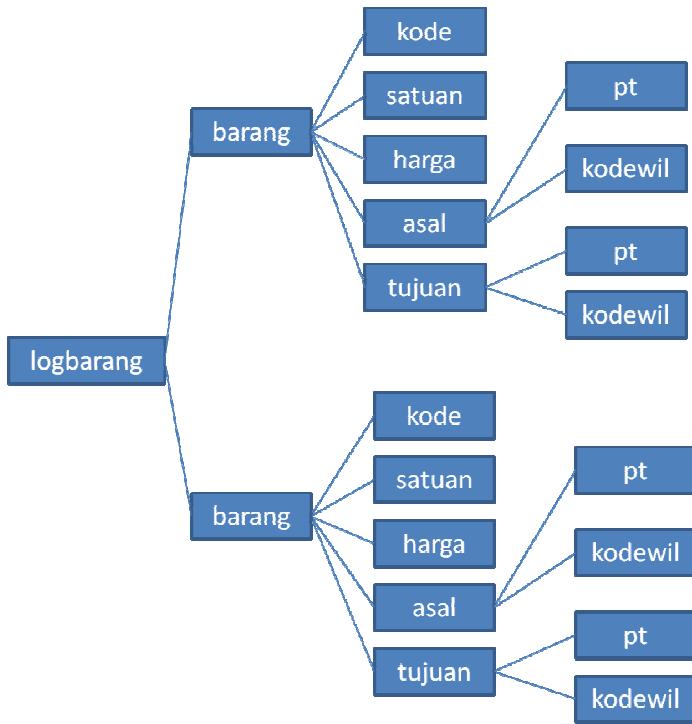
3 DOM

Berbeda dengan SAX, DOM membaca seluruh dokumen XML ke dalam memori dan mempresentasikannya sebagai tree of nodes, sehingga dapat dilakukan manipulasi terhadap data memori, mengakses nilai node pada lokasi dimanapun. DOM direkomendasikan oleh W3C.

Contoh XML:

```
<?XML version="1.0" encoding="UTF-8"?>
<logbarang>
<barang>
<kode>M1112</kode>
<satuan>pc</satuan>
<harga>5000</harga>
<asal>
    <pt>ladyrock</pt>
    <kodewil>10</kodewil>
</asal>
<tujuan>
    <pt>union</pt>
    <kodewil>1001</kodewil>
</tujuan>
</barang>
<barang>
    <kode>M1122</kode>
    <satuan>pc</satuan>
    <harga>1000</harga>
    <asal>
        <pt>pbm</pt>
        <kodewil>103</kodewil>
    </asal>
    <tujuan>
        <pt>mitra kencana</pt>
        <kodewil>300</kodewil>
    </tujuan>
</barang>
</logbarang>
```

XML DOM tree nodes untuk XML di atas:



Selanjutnya kita akan mengolah dokumen XML dengan beberapa bahasa (contoh php, javascript). Untuk tools IDE Anda dapat menggunakan eclipse dlsb (tidak disarankan notepad).

PRAKTIK

(Metode SAX)

1. Diketahui Anda memiliki XML:

```
<?XML version="1.0" encoding="UTF-8"?>
<logbarang>
<barang>
  <kode>M1112</kode>
  <satuan>pc</satuan>
  <harga>5000</harga>
  <asal>
    <pt>ladyrock</pt>
    <kodewil>10</kodewil>
  </asal>
  <tujuan>
    <pt>union</pt>
    <kodewil>1001</kodewil>
  </tujuan>
</barang>

<barang>
  <kode>M1122</kode>
  <satuan>pc</satuan>
  <harga>1000</harga>
  <asal>
    <pt>pbm</pt>
    <kodewil>103</kodewil>
  </asal>
  <tujuan>
    <pt>mitra kencana</pt>
    <kodewil>300</kodewil>
  </tujuan>
</barang>

</logbarang>
```

2. Untuk penggunaan php pastikan server apache berjalan.
3. Buat pada direktori htdocs: folder domsax
4. Anda berada pada workspace domsax folder.
5. Buat file XML beri nama barang.xml
6. Isikan barang.xml dengan data pada langkah no 1.
7. Buat file php, beri nama: sax_baca.php
8. Isikan

1. <?php
2. // membaca xml dengan teknik sequensial

```

3. // Simple API for XML (SAX) parser
4. global $isstart;
5.
6. $isstart = false;
7. function start_element($parser, $name, $attrs) {
8.     global $isstart;
9.     $isstart = true;
10.
11.    echo "Start Element: $name";
12.
13.    foreach ( $attrs as $key => $value ) {
14.        echo ", atribut: $key [ <b>$value</b> ]";
15.    }
16.}
17.function end_element($parser, $name) {
18.    global $isstart;
19.    $isstart = false;
20.
21.    echo "End Element: $name";
22.    echo '</br>';
23.}
24.function characters($parser, $chars) {
25.    global $isstart;
26.    if (trim ( $chars, " \t\n\r\0\x0B" ) != '') {
27.
28.        echo "[<b> $chars </b>]";
29.    } else if ($isstart)
30.        echo '</br>';
31.}
32.
33.$barang_parser = xml_parser_create ();
34.
35.xml_set_element_handler ( $barang_parser, "start_element", "end_element" );
36.
37.xml_set_character_data_handler ( $barang_parser, "characters" );
38.
39.$file = "barang.xml";
40.if ($file_stream = fopen ( $file, "r" )) {
41.
42.    while ( $data = fread ( $file_stream, 4096 ) ) {
43.
44.        $this_chunk_parsed = xml_parse ( $barang_parser, $data, feof (
45.            $file_stream ) );
46.        if (! $this_chunk_parsed) {
47.            $error_code = xml_get_error_code ( $barang_parser );
48.            $error_text = xml_error_string ( $error_code );
49.            $error_line = xml_get_current_line_number ( $barang_parser
50.        );
51.            $output_text = "Parsing problem at line $error_line:
52.                $error_text";
53.        }
54.    } else {

```

```
55.  
56.      die ( "Can't open XML file." );  
57. }  
58.xml_parser_free ( $barang_parser );  
59.  
60.?>
```

9. Catatan kedua file tersebut dalam 1 folder yang sama.
10. Anda dapat melihat hasilnya pada browser.
11. Dengan metode SAX, jika Anda memiliki 1000 record bagaimana Anda mengakses record ke -15 kemudian record ke -900 kemudian kembali lagi ke record ke -2?

PRAKTIK

(Metode DOM)

1. Diketahui Anda memiliki XML:

```
<?XML version="1.0" encoding="UTF-8"?>
<logbarang>
<barang>
  <kode>M1112</kode>
  <satuan>pc</satuan>
  <harga>5000</harga>
  <asal>
    <pt>ladyrock</pt>
    <kodewil>10</kodewil>
  </asal>
  <tujuan>
    <pt>union</pt>
    <kodewil>1001</kodewil>
  </tujuan>
</barang>

<barang>
  <kode>M1122</kode>
  <satuan>pc</satuan>
  <harga>1000</harga>
  <asal>
    <pt>pbm</pt>
    <kodewil>103</kodewil>
  </asal>
  <tujuan>
    <pt>mitra kencana</pt>
    <kodewil>300</kodewil>
  </tujuan>
</barang>

</logbarang>
```

2. Untuk penggunaan php pastikan server apache berjalan.
3. Anda berada pada workspace domsax folder.
4. Buat file XML beri nama barang.xml
5. Isikan barang.xml dengan data pada langkah no 1.
6. Buat file php, beri nama: dom_baca.php
7. Isikan:

1. <?php
2. //membaca file xml, meloadnya menjadi objek dan disimpan di memori
3. //Document Object Model (DOM)
- 4.

```
5. $doc = new DOMDocument ();
6. $doc->load ( 'barang.xml' );
7.
8. $barangs = $doc->getElementsByTagName ( "barang" );
9.
10. foreach ( $barangs as $barang ) {
11.
12.     $ket = $barang->getAttribute ( 'keterangan' );
13.
14.     $kodes = $barang->getElementsByTagName ( "kode" );
15.     $kode = $kodes->item ( 0 )->nodeValue;
16.
17.     $satuans = $barang->getElementsByTagName ( "satuan" );
18.     $satuan = $satuans->item ( 0 )->nodeValue;
19.
20.     $hargas = $barang->getElementsByTagName ( "harga" );
21.     $harga = $hargas->item ( 0 )->nodeValue;
22.
23.     echo "$kode - $satuan - $harga - $ket" . "</br>";
24. }
25.
26. ?>
```

8. Jalankan script tersebut pada browser.

PRAKTIK

(Dengan javascript)

1. Diketahui Anda memiliki XML:

```
<?XML version="1.0" encoding="UTF-8"?>
<logbarang>
<barang>
  <kode>M1112</kode>
  <satuan>pc</satuan>
  <harga>5000</harga>
  <asal>
    <pt>ladyrock</pt>
    <kodewil>10</kodewil>
  </asal>
  <tujuan>
    <pt>union</pt>
    <kodewil>1001</kodewil>
  </tujuan>
</barang>

<barang>
  <kode>M1122</kode>
  <satuan>pc</satuan>
  <harga>1000</harga>
  <asal>
    <pt>pbm</pt>
    <kodewil>103</kodewil>
  </asal>
  <tujuan>
    <pt>mitra kencana</pt>
    <kodewil>300</kodewil>
  </tujuan>
</barang>

</logbarang>
```

2. Ingat javascript adalah client side sehingga bisa saja tidak membutuhkan server untuk dieksekusi. Dalam praktikum ini Anda akan menjalankan script javascript pada localhost.
3. Anda berada pada workspace domsax folder.
4. Buat file XML beri nama barang.xml
5. Isikan barang.xml dengan data pada langkah no 1.
6. Buat file html, beri nama: javascript_baca1.html
7. Isikan:

```

1. <html>
2. <body>
3.
4.
5.     <script type="text/javascript">
6.         if (window.XMLHttpRequest) {// code for IE7+, Firefox, Chrome,
7.             // Opera, Safari
8.             xmlhttp = new XMLHttpRequest();
9.         } else {// code for IE6, IE5
10.             xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
11.         }
12.         xmlhttp.open("GET", "barang.xml", false);
13.         xmlhttp.send();
14.         xmlDoc = xmlhttp.responseXML;
15.
16.         var xlength = xmlDoc.getElementsByTagName("barang").length;
17.
18.         for (i = 0; i < xlength; i++) {
19.             document
20.
21.             .write(xmlDoc.getElementsByTagName("kode")[i].childNodes[0].nodeValue);
22.             document.write("<br>");
23.
24.             .write(xmlDoc.getElementsByTagName("satuan")[i].childNodes[0].nodeValue)
25.             ;
26.             document.write("<br>");
27.             document.write(xmlDoc.getElementsByTagName("barang")[i]
28.                             .getAttributNode("keterangan").nodeValue);
29.             document.write("<br>");
30.
31.         </script>
32.     </body>
33. </html>

```

8. Jalankan pada browser.

PRAKTIK

(Dengan javascript, modifikasi program sebelumnya untuk navigasi)

1. Masih bekerja pada folder domsax dan dokumen barang.xml
2. Buat file html beri nama: javascript_baca2.html
3. Isikan:

```
1. <html>
2. <head>
3. <title>Navigasi xml</title>
4. <script type="text/javascript">
5.     if (window.XMLHttpRequest) {// code for IE7+, Firefox, Chrome, Opera,
Safari
6.         xmlhttp = new XMLHttpRequest();
7.     } else {// code for IE6, IE5
8.         xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
9.     }
10.
11.    xmlhttp.open("GET", "barang.xml", false);
12.    xmlhttp.send();
13.    xmlDoc = xmlhttp.responseXML;
14.
15.    var xlength = xmlDoc.getElementsByTagName("barang").length;
16.
17.    var nom = 0;
18.
19.    function prev() {
20.        nom = nom - 1;
21.        if (nom <= 0) {
22.            nom = 0;
23.        }
24.        tampil(nom);
25.    }
26.
27.    function next() {
28.        nom = nom + 1;
29.        if (nom >= xlength) {
30.            nom = xlength - 1;
31.            alert("Telah mencapai batas akhir");
32.        }
33.        tampil(nom);
34.    }
35.
36.    function tampil(prm1) {
37.        document.getElementById("kode").innerHTML = xmlDoc
38.
39.        .getElementsByTagName("kode")[prm1].childNodes[0].nodeValue;
40.
41.        document.getElementById("satuan").innerHTML = xmlDoc
42.
43.        .getElementsByTagName("satuan")[prm1].childNodes[0].nodeValue;
44.        document.getElementById("keterangan").innerHTML = xmlDoc
```

```

42.                               .getElementsByTagName("barang")[prm1]
43.                               .getAttributeNode("keterangan").nodeValue;
44.                           }
45.</script>
46.</head>
47.
48.
49.
50.
51.
52.<body onload="tampil(0);">
53.    <table width="284" border="1" bordercolor="#000000">
54.        <tr>
55.            <th width="68" bgcolor="#0066FF" scope="row">kode</th>
56.            <td width="200" id="kode">...</td>
57.        </tr>
58.        <tr>
59.            <th bgcolor="#0066FF" scope="row">satuan</th>
60.            <td id="satuan">...</td>
61.        </tr>
62.        <tr>
63.            <th bgcolor="#0066FF" scope="row">keterangan</th>
64.            <td id="keterangan">...</td>
65.        </tr>
66.    </table>
67.    <form>
68.        <table width="282" border="0">
69.            <tr>
70.                <td width="145" align="left"><input type="button"
71. name="prev1"
72.                               onclick="prev(); value=<->"></td>
73.                <td width="127" align="right"><input type="button"
74. name="next1"
75.                               onclick="next(); value=->"></td>
76.            </tr>
77.        </table>
78.</body>
79.</html>

```

4. Jalankan program di atas.

PRAKTIK

(menulis dengan metode DOM pada php)

1. Masih bekerja pada folder domsax dan dokumen barang.xml
2. Buat file php beri nama: dom_tulis.php
3. Isikan:

```
1. <?php
2. // write dengan teknik DOM
3. $logbarang = array ();
4. $logbarang [] = array (
5.     'satuan' => 'pc',
6.     'kode' => 'M1112',
7.     'harga' => "5000",
8.     'keterangan' => 'reject'
9. );
10.$logbarang [] = array (
11.     'satuan' => 'pc',
12.     'kode' => 'M1122',
13.     'harga' => "1000",
14.     'keterangan' => 'good'
15. );
16.
17.$doc = new DOMDocument ();
18.$doc->formatOutput = true;
19.
20.$r = $doc->createElement ( "logbarang" );
21.$doc->appendChild ( $r );
22.
23.foreach ( $logbarang as $barang ) {
24.     $b = $doc->createElement ( "barang" );
25.
26.     //atribut
27.     $ket=$doc->createAttribute("keterangan");
28.     $ket->appendChild( $doc->createTextNode( $barang [ 'keterangan' ] ) );
29.     $b->appendChild($ket);
30.
31.     $kode = $doc->createElement ( "kode" );
32.     $kode->appendChild ( $doc->createTextNode ( $barang [ 'kode' ] ) );
33.     $b->appendChild ( $kode );
34.
35.     $satuan = $doc->createElement ( "satuan" );
36.     $satuan->appendChild ( $doc->createTextNode ( $barang [ 'satuan' ] ) );
37.     $b->appendChild ( $satuan );
38.
39.     $harga = $doc->createElement ( "harga" );
40.     $harga->appendChild ( $doc->createTextNode ( $barang [ 'harga' ] ) );
41.     $b->appendChild ( $harga );
42.
43.     $r->appendChild ( $b );
44. }
```

```
45.  
46. echo $doc->saveXML ();  
47.?>
```

4. Jalankan program di atas.

PRAKTIK

(menulis dokumen XML pada php)

1. Masih bekerja pada folder domsax dan dokumen barang.xml
2. Buat file php beri nama: php_tulis.php
3. Isikan:

```
1. <?php
2. // menulis dengan script php
3. // perlu diperhatikan validasi dan well formed!
4. $logbarang = array ();
5. $logbarang [] = array (
6.     'satuan' => 'pc',
7.     'kode' => 'M1112',
8.     'harga' => "5000",
9.     'keterangan' => 'reject'
10.);
11.$logbarang [] = array (
12.    'satuan' => 'pc',
13.    'kode' => 'M1122',
14.    'harga' => "1000",
15.    'keterangan' => 'good'
16.);
17.?
18.<!--Perhatikan templatanya -->
19.<logbarang>
20.  <?php
21.
22.      foreach ( $logbarang as $barang ) {
23.          ?
24.      <barang>
25.      <kode><?php echo( $barang['kode'] ); ?></kode>
26.      <satuan><?php echo( $barang['satuan'] ); ?>
27.      </satuan> <harga><?php echo( $barang['harga'] ); ?>
28.      </harga> </barang>
29.      <?php
30.          ?
31.          ?
32.  </logbarang>
```

4. Jalankan program di atas.

TUGAS:

Buatlah file xml dan xsd berdasarkan kebutuhan perusahaan yang bergerak di bidang:

Industri manufacturing

Industri jasa

Industri perhotelan

Pendidikan

Isi data pada masing-masing file xml minimal 5, libatkan juga penggunaan atribut, pattern
(contoh: pola untuk email abc@xyz.com) minimal elemen 10.

Dari dokumen XML yang pernah Anda buat di materi sebelumnya lakukan proses pembacaan dengan teknik yang telah dijelaskan pada materi ini.

BAB VI

TIK:

Mahasiswa dapat Memahami dasar-dasar SOAP dan REST

Pokok Bahasan:

- SOAP, REST

Sub Pokok Bahasan:

- Pendahuluan
- SOAP, WSDL
- REST

1 Pendahuluan

Seperti telah diketahui pertukaran data antar mesin dapat dilakukan dengan berbagai cara, pada awalnya RPC, RMI kemudian SOAP dan RESTful. Pada materi hanya akan dibahas mengenai SOAP dan RESTful.

2 SOAP

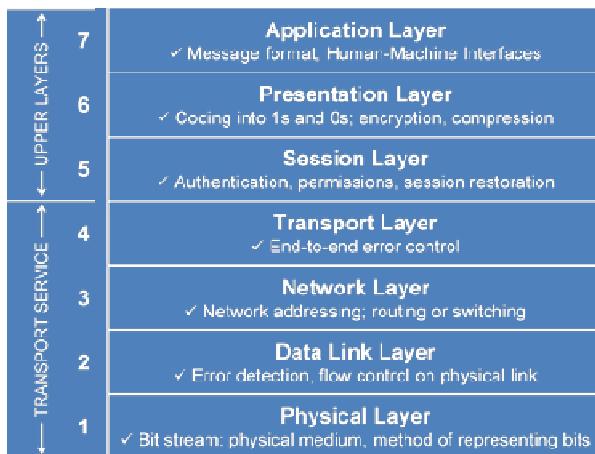
Simple Object Access Protocol (SOAP) adalah spesifikasi protokol standar untuk pertukaran pesan berbasis XML. Komunikasi antara layanan web dan klien terjadi menggunakan pesan XML.

SOAP dirancang sebagai protokol objek-akses pada tahun 1998 dan menjadi rekomendasi W3C pada tahun 2003.

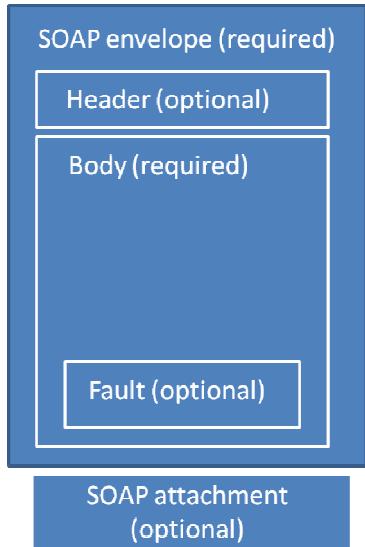
2.1 Spesifikasi framework SOAP terdiri dari:

- processing model, mendefinisikan tentang untuk memproses pesan
- extensibility model, mendefinisikan tentang konsep fitur SOAP dan modul SOAP
- SOAP underlying protocol binding, menggambarkan aturan untuk mendefinisikan protokol pengikat dasar yang dapat digunakan untuk bertukar pesan SOAP antara node-node SOAP
- SOAP message construct, mendefinisikan struktur pesan SOAP

SOAP berada pada lapisan transport OSI layer



Struktur pesan SOAP



- Root elemen dari suatu pesan SOAP adalah elemen: Envelope
- Terdiri dari optional header dan harus terdapat isi/body pesan
- Elemen fault(bersifat optional) dapat digunakan untuk menggambarkan kondisi diluar kewajaran
- Dapat dilampirkan Attachment pada MIME encoding untuk pertukaran data biner

Contoh pesan SOAP:

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope
    soap:encodingStyle="http://soap.org/soap/encoding/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:soap="http://xmlsoap.org/soap/envelope/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soap:Header>
        <!-- Transactions, priorites, etc. -->
    </soap:Header>
    <soap:Body>
        <!-- Some content -->
    </soap:Body>
</soap:Envelope>
```

2.2 SOAP Message Transmission

Transmisi pesan SOAP melibatkan:

- SOAP Sender, membuat dan mengirimkan pesan kepada SOAP receiver
- SOAP Intermediaries, suatu perantara SOAP bersifat optional untuk menangkap pesan antara pengirim dan penerima. Pada bagian ini dapat dilakukan filtering, logging, catching dll
- SOAP Receiver, tujuan pesan dari pengirim.



Dari gambar di atas terlihat bahwa arsitektur umum dari web service memiliki:

- Sender atau Client
- Receiver atau Service provider

untuk berkomunikasi klien harus mengetahui beberapa informasi seperti:

- Lokasi webservices Server
- Fungsi yang tersedia, signature parameter dan tipe return nya.
- komunikasi protokol
- Format input output

Service provider akan membuat file XML standar yang mengkover kebutuhan di atas. Jika file ini diberikan pada klien maka klien dapat mengakses web services. File XML ini disebut Web Service Description Language (WSDL).

2.2.1 WSDL

Web Services Description Language (WSDL) adalah format XML untuk menjelaskan semua informasi yang dibutuhkan untuk permintaan dan komunikasi dengan Web Service. Deskripsi service memiliki dua komponen utama:

- Functional Description

Mendefinisikan rincian tentang bagaimana Web Service dipanggil, di mana service dipanggil. Berfokus pada rincian sintaks pesan dan bagaimana mengkonfigurasi protokol jaringan untuk memberikan pesan.

- Nonfunctional Description

Memberikan rincian sekunder untuk pesan (seperti sebagai security policy), diperlukan instruksi tambahan pada header SOAP.

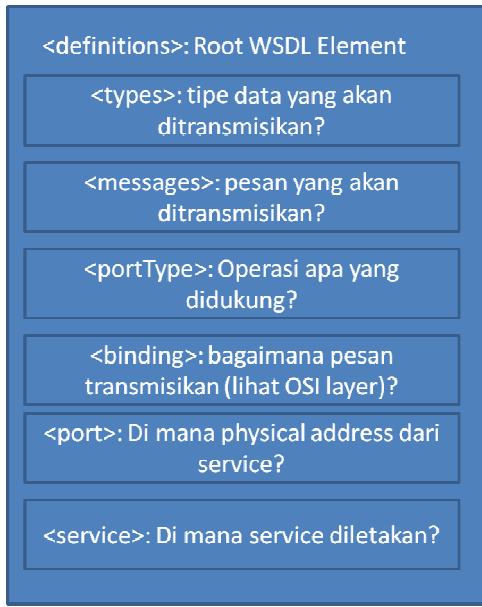
Karena WSDL merupakan XML maka baik user (manusia) maupun mesin dapat membacanya dengan mudah. Dengan menggunakan file WSDL maka Anda akan memahami:

- Port / Endpoint – URL dari web service
- Input message format
- Output message format
- Security protocol yang harus diikuti
- Protocol yang web service gunakan

Struktur dokumen WSDL

Sebuah Dokumen WSDL adalah kumpulan definisi dengan root elemen tunggal. Layanan dapat didefinisikan dengan menggunakan elemen XML berikut:

- Types, seperti tipe data
- Message, seperti methods
- PortType, seperti Interfaces
- Binding, seperti cara pengikatan Encoding Scheme
- Port, seperti URL
- Service, seperti banyak URLs

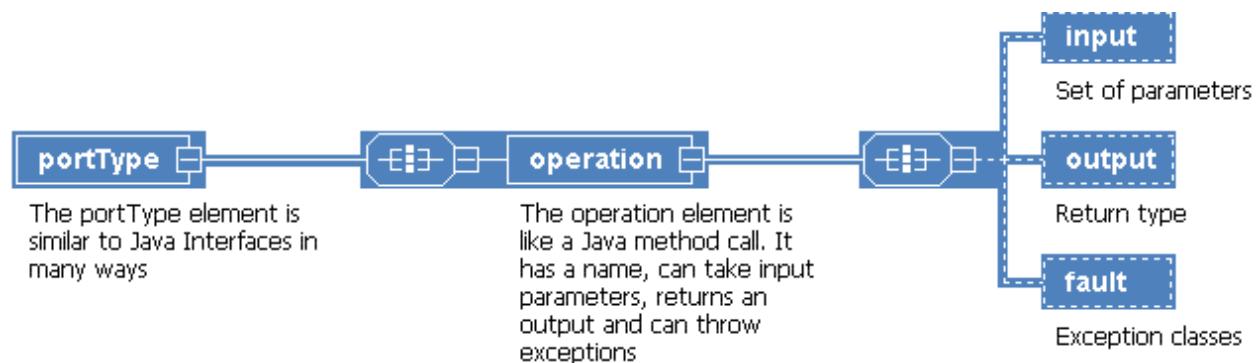


PortType Element

Definisi:

Elemen portType menggambarkan interface pada webservice

- Sebuah Dokumen WSDL dapat berisi nol atau banyak portType
- Elemen portType berisi satu nama atribut.
- Konvensi penamaan nameOfWebService PortType
- PortType berisi satu atau lebih elemen operasi, dengan nama atribut dapat berisi input, output dan fault elemen



Contoh:

```
<!-- Port Type Definition Example -->
<portType name="weatherCheckPortType">
<operation name="checkTemperature">
<input message="checkTemperatureRequest"/>
<output message="checkTemperatureResponse"/>
</operation>
<operation name="checkHumidity">
<input message="checkHumidityRequest"/>
<output message="checkHumidityResponse"/>
</operation>
</portType>
```

Message Element

Definisi:

Message adalah kumpulan bagian argumen.

- Sebuah dokumen WSDL dapat berisi nol atau banyak elemen pesan
- Setiap elemen pesan dapat digunakan sebagai input, output atau fault message dalam suatu operasi.
- Jenis atribut bagian dapat berupa tipe data standar dari XSD Schema atau ditetapkan oleh pengguna.

Contoh:

```
<!-- Message Definitions -->
<message name="checkTemperatureRequest">
<part name="location" type="xsd:string">
</message>
<message name="checkTemperatureResponse">
<part name="result" type="xsd:double">
</message>
<message name="checkHumidityRequest">
<part name="location" type="xsd:string">
</message>
<message name="checkHumidityResponse">
<part name="result" type="ns:HummidityType">
</message>
```

Types Element

Tipe custom user data dapat didefinisikan dengan cara yang abstrak.

- Tipe default di WSDL adalah XML Schema (XSD)
- Sebuah dokumen WSDL dapat memiliki paling banyak satu elemen jenis.
- Unsur jenis dapat berisi simpleType atau complexType.
- Pada elemen-elemen tingkat terendah, elemen didefinisikan dengan nama dan jenis atribut.

Contoh:

```
<!-- Type Definitions -->
<types>
<xsd:schema targetNamespace="http://weather.com/ns"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:complexType name="HumidityType">
<xsd:sequence>
<xsd:element name="loc" type="xsd:string">
<xsd:element name="humd" type="xsd:double">
<xsd:element name="temp" type="xsd:double">
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
</types>
```

Binding Element

Definisi:

Binding elemen menentukan pada service requester tentang bagaimana membuat pesan untuk protokol tertentu.

Setiap portType dapat memiliki satu atau lebih elemen yang terkait.

Untuk portType yang telah ditentukan, elemen yang mengikat harus menentukan pesan dan pasangan transportnya. (SOAP / HTTP, SOAP / SMTP, dll).

Contoh:

```
<binding name="WeatherBinding" type="weatherCheckPortType">
<soap:binding style="rpc"
transport="http://schemas.xmlsoap.org/soap/http" />
<operation name="checkTemperature">
<soap:operation soapAction="" />
<input>
<soap:body use="encoded" namespace="checkTemperature"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</input>
<output>
<soap:body use="encoded" namespace="checkTemperature"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</output>
</operation>
</binding>
```

Port Element

Port element menentukan alamat network dari web service

- Terdapat sebuah protokol, alamat khusus pada elemen terkait
- Port harus bernama dan unik pada dokumen

Contoh:

```
<port name="WeatherCheck"
binding="wc:WeatherCheckSOAPBinding">
<soap:address location="http://host/WeatherCheck"/>
</port>
```

Service Element

Elemen service adalah kumpulan port elemen diidentifikasi dengan satu nama service.

- Sebuah Dokumen WSDL diperbolehkan mengandung beberapa elemen service, tetapi secara biasanya terdapat satu.
- Setiap service harus diberi nama yang unik.
- Konvensi penamaan adalah GeneralInfoService

Contoh:

```
<!-- Service definition -->
<service name="WeatherCheckService">
<port name="WeatherCheckSOAP"
binding="wc:WeatherCheckSOAPBinding">
<soap:address location="http://host/WeatherCheck"/>
</port>
<port name="WeatherCheckSMTP"
binding="wc:WeatherCheckSMTPBinding">
<soap:address location="http://host/WeatherCheck"/>
</port>
</service>
```

Demikian pembahasan mengenai SOAP WSDL, yang harus Anda perhatikan adalah kesesuaian method yang dipanggil baik dari signature parameter, return type dan portnya. Sebagai alternatif webservice, Anda diperkenalkan dengan Representational State Transfer (REST).

3 REST

Representational State Transfer (REST) adalah arsitektur yang mengambil abstrak elemen arsitektur dalam sistem hypermedia terdistribusi. REST mengabaikan detail pelaksanaan komponen dan sintaks protokol dan fokus pada peran komponen, interaksi antar komponen dan interpretasi data. REST muncul sebagai model API web yang dominan.

Istilah representational state transfer diperkenalkan dan didefinisikan pada tahun 2000 oleh Roy Fielding dalam disertasi doktoralnya di UC Irvine.

Dalam istilah web service, REST adalah suatu arsitektur stateless client-server, sehingga web service dipandang sebagai resource dan dapat diidentifikasi melalui Universal Resource Identifier (URI). Klien web service yang akan menggunakan source tersebut harus melalui metode remote yang telah ditentukan oleh resource yang bersangkutan.

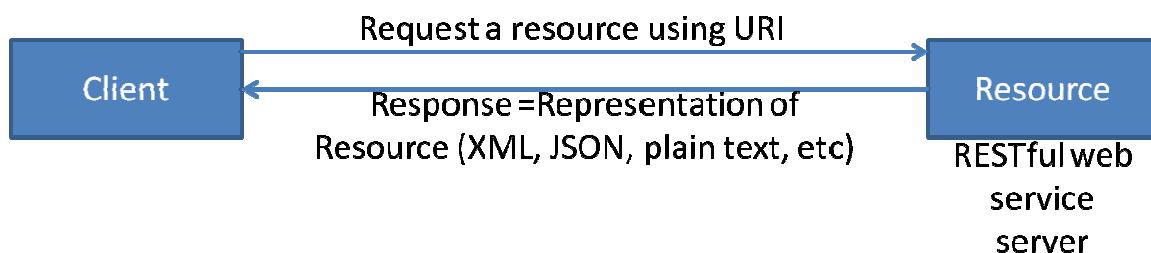
REST terbagi menjadi dua komponen yaitu server REST dan klien REST.

Pada arsitektur REST, klien dan server berkomunikasi dengan interface standar. REST bukanlah protokol secara spesifik tapi biasanya komunikasi REST melalui HTTP.

HTTP methods :

REST web service menggunakan protokol HTTP dalam operasinya. Methodnya antara lain:

- GET, mendefinisikan akses baca terhadap source server
- PUT, menciptakan resource baru
- DELETE, menghapus resource
- POST, mengupdate atau menciptakan resource baru



REST memungkinkan resource memberikan representasi yang berbeda seperti XML, JSON dll, dan komunikasi menggunakan HTTP.

Fitur REST web service

- Resource identification through URI
Resource diidentifikasi lewat URI (biasanya berupa link internet, Anda dapat mengaksesnya melalui address bar pada browser atau interface aplikasi lainnya)

- Uniform interface
Untuk mengakses resource, method standar dapat digunakan, antara lain: PUT, GET, POST, dan DELETE.
- Client-Server
Dapat membantu meningkatkan portabilitas pada klien dan skalabilitas pada server
- Stateless
Setiap permintaan dari klien ke server harus berisi tentang semua informasi yang diperlukan sehingga tidak dapat mengambil konteks lain di server
- Cache
untuk meningkatkan respon efisiensi jaringan dapat dibedakan untuk disimpan di cache atau tidak disimpan di cache.
- Named resources
Resource yang diakses melalui penamaan URL
- Interconnected resource representations
Representasi dari resource terhubung menggunakan URL
- Layered components
Perantara, seperti server proxy, cache server, gateway, dll, dapat disisipkan di antara klien dan source untuk mendukung kinerja, keamanan, dll
- Self-descriptive messages
Resource dipisahkan dari representasi sehingga isinya dapat diakses dalam berbagai format, seperti HTML, XML, teks biasa, PDF, JPEG, JSON dll.

TUGAS:

Buat artikel kelompok mengenai SOAP WSDL dan REST RESTful meliputi:

- sejarah
- perbedaan
- industri dan teknologi yang dipilih SOAP/REST
- tren ke depan berkaitan dengan mobile dan security

BAB VII

TIK:

Mahasiswa dapat Memahami dasar-dasar keamanan pada web service

Pokok Bahasan:

- Keamanan

Sub Pokok Bahasan:

- Aspek keamanan dasar
- Solusi

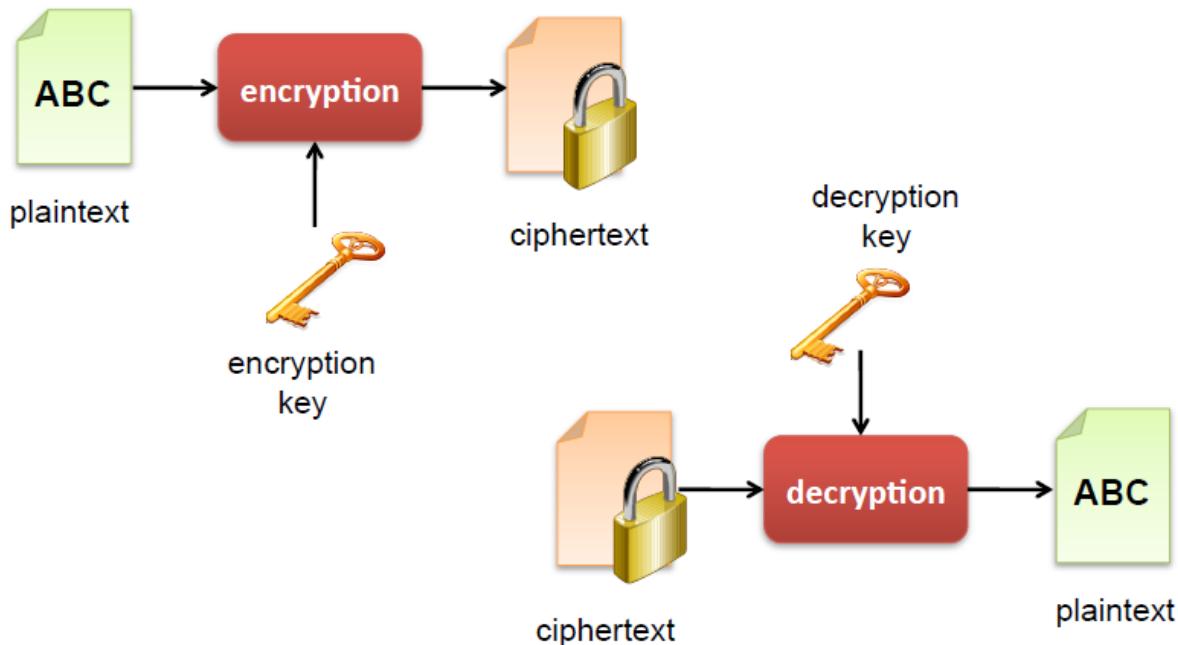
1 Aspek keamanan dasar

- Kerahasiaan
 - Melindungi informasi dari pengungkapan yang tidak sah
- Integritas
 - Pastikan informasi belum diubah oleh pihak yang tidak berwenang
- Otentikasi
 - Pastikan bahwa identitas dipetakan dengan benar
- Otorisasi
 - Mencegah penggunaan yang tidak sah pada resources
- Non-repudiation
 - Pastikan bahwa pihak terkait tidak dapat menyangkal tindakan mereka sebelumnya
- Ketersediaan
 - Pastikan bahwa informasi / layanan yang tersedia bila diperlukan

2 Solusi

2.1 Enkripsi

- Enkripsi
 - Dilakukan untuk menjamin kerahasiaaan informasi
 - Informasi ditransformasi sehingga pihak yang tidak berhak tidak memahaminya
- Hal ini dapat dilakukan dengan beberapa algoritma enkripsi, misalnya:
 - AES (Advanced Encryption Standard),
 - DES (Data Encryption Standard)
- Algoritma ini menggunakan parameter disebut key/kunci
 - Sering kali kunci yang sama digunakan untuk proses enkripsi dan dekripsi.
 - Public key cryptography dapat digunakan untuk membangun kunci rahasia bersama antar pihak yang berkomunikasi. Kunci harus sulit untuk ditebak, oleh karena itu kunci harus memiliki bits yang panjang



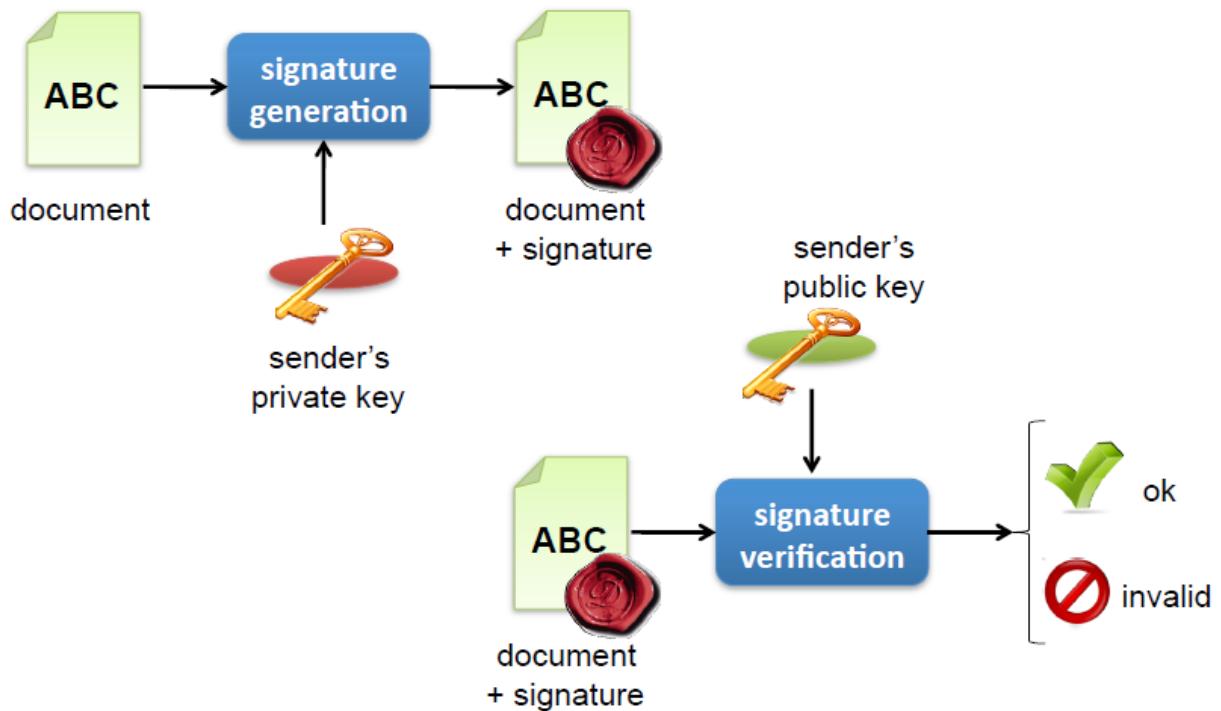
2.2 Digital Signatures

Digital Signature menjamin integritas, otentifikasi pengirim, non repudiation. Terdapat pula cara yang lebih ringan dan sederhana untuk menjamin integritas dan otentifikasi misalnya dengan Message Authentication Codes.

Hal ini dihasilkan menggunakan algoritma digital signature seperti DSA (Digital Signature Algorithm), RSA (Rivest - Shamir - Adleman).

Algoritma menggunakan sepasang kunci (kunci publik dan kunci privat)

- Kunci privat hanya digunakan oleh pemilik dan digunakan untuk sign
- Kunci publik diketahui oleh semua orang dan digunakan untuk memverifikasi signature
- Terdapat hubungan matematis antara kedua kunci



2.3 Certificates

Untuk memverifikasi suatu signature, penerima membutuhkan kunci publik pengirim dan untuk memastikan bahwa kunci tersebut adalah milik pengirim.

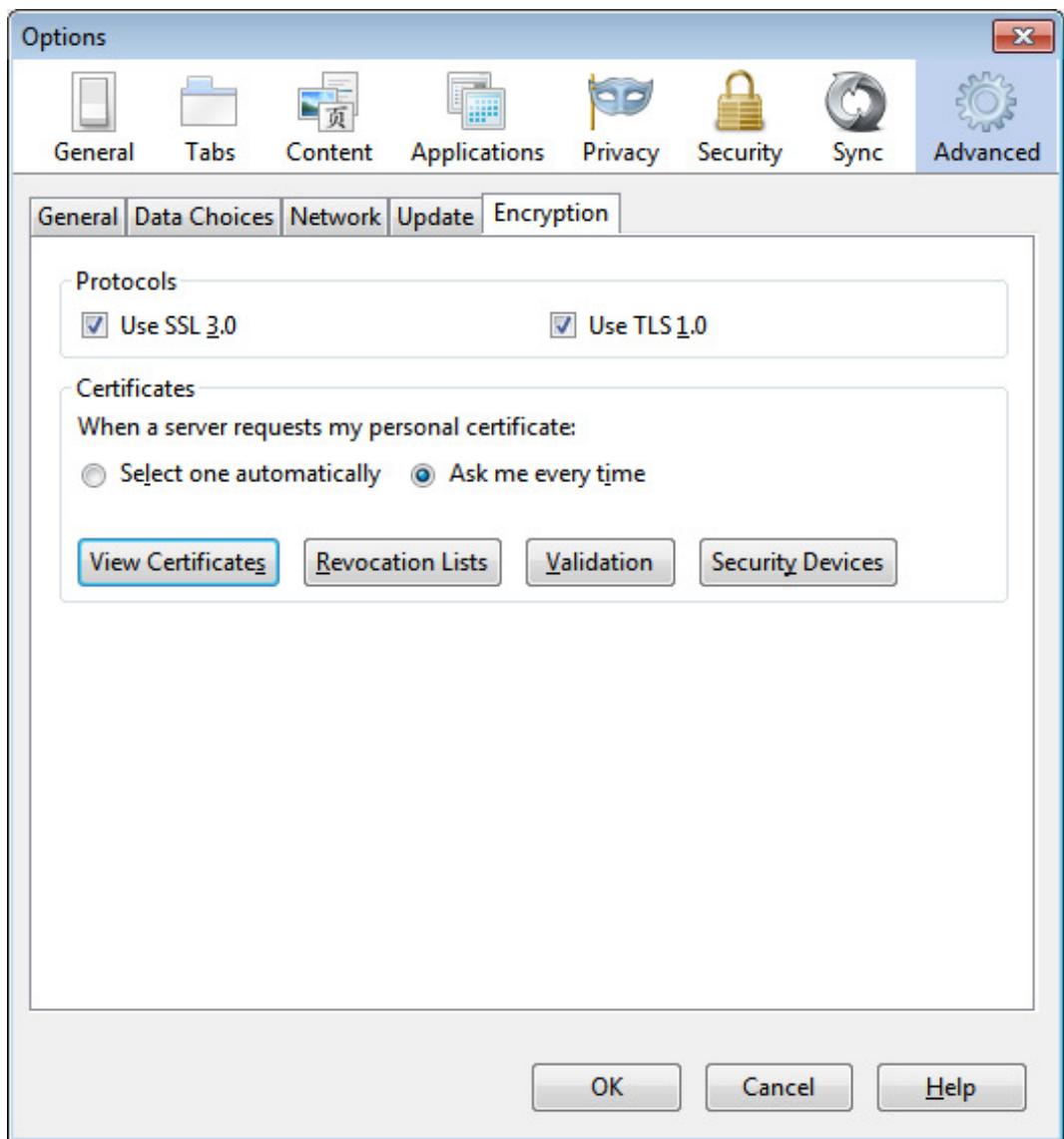
Cara untuk mengamankan distribusi kunci publik ini adalah dengan Digital certificates.

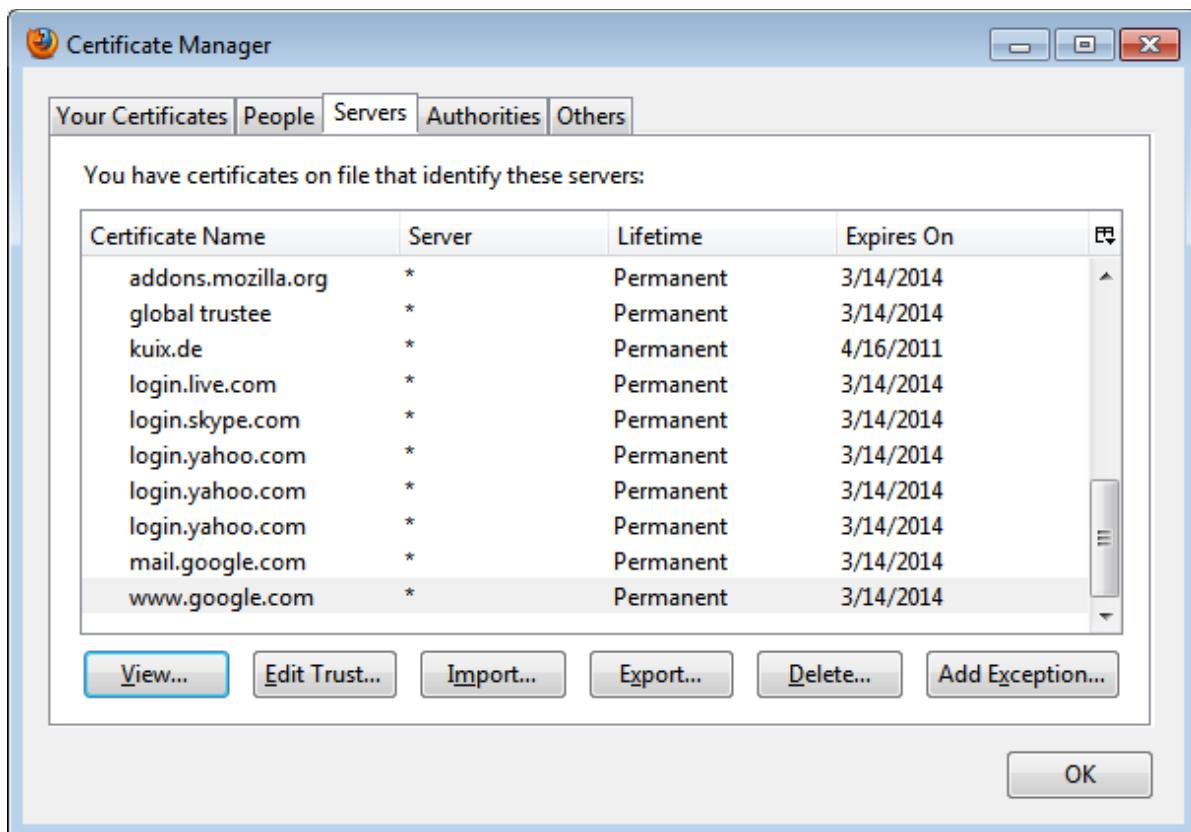
Suatu sertifikat menyatakan hubungan antara subjek (user/orang atau entitas) dan kunci publik. Kunci dan identitas dari subjek dinyatakan dalam sertifikat. Sertifikat yang umum adalah X.509.

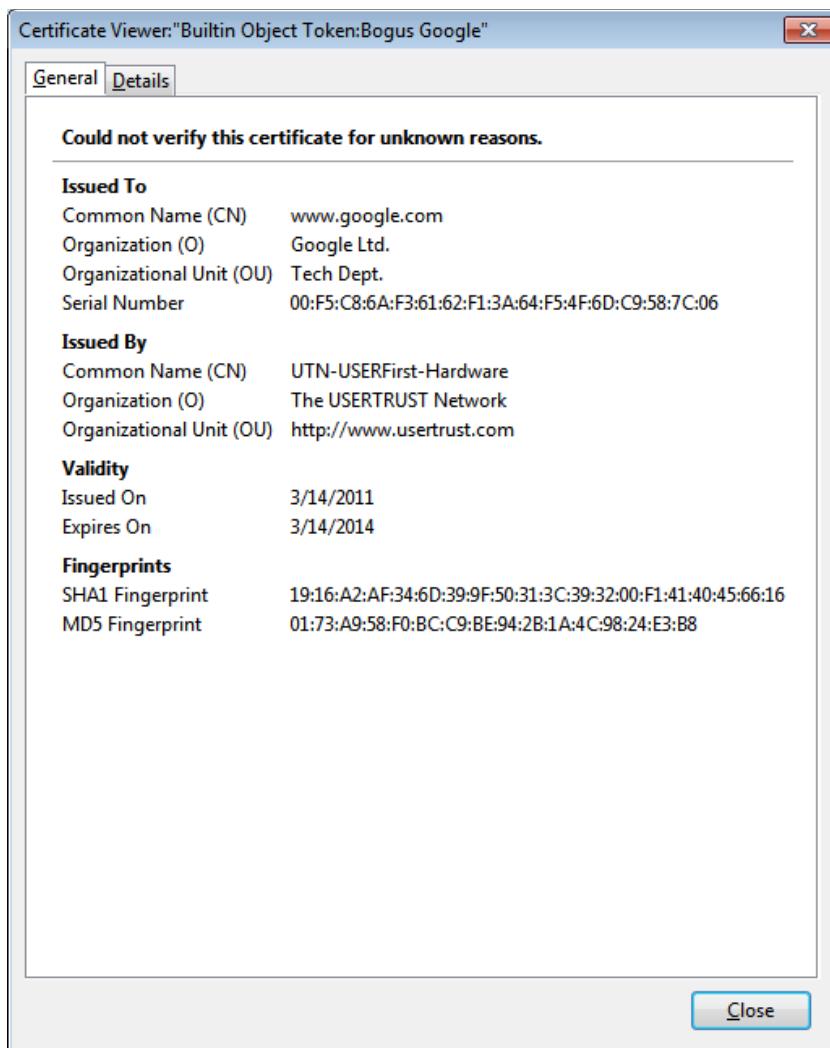
Suatu sertifikat dikeluarkan oleh suatu certificate authority (CA).

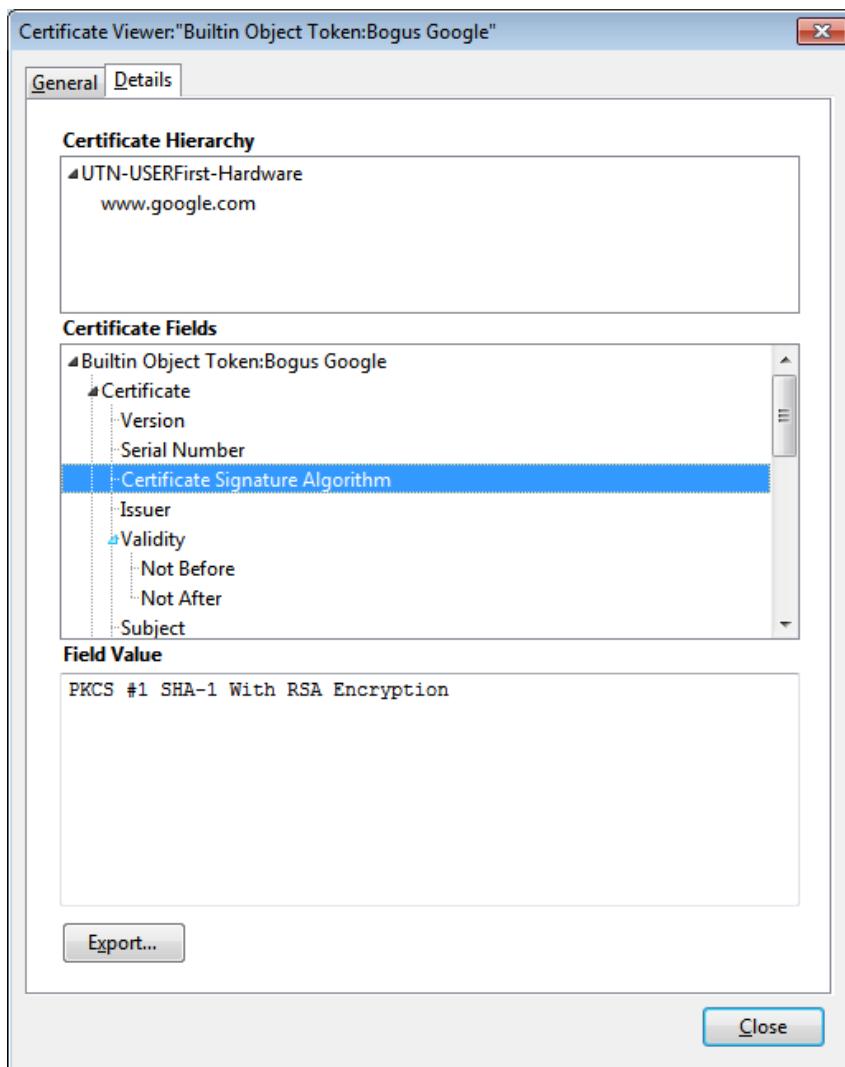
Infrastruktur yang dibutuhkan untuk membuat, mendistribusikan, mengelola, mencabut sertifikat disebut PKI (Public Key Infrastructure).

Jika penerima mempercayai CA sebagai penerbit sertifikat maka dapat diyakini kunci tersebut adalah benar.









Kaitan dengan logistik

Dengan pertumbuhan B2B yang semakin banyak dan cepat dibutuhkan komunikasi dinamis diantaranya (dapat berupa lintas platform dlsb), web service adalah sarana yang tepat untuk menjawabnya. Keamanan dalam pertukaran informasi harus diperhatikan dan merupakan faktor vital pada B2B.

TUGAS:

Kelompok: Dari bab sebelumnya cobalah Anda mengekripsi dan dekripsi file XML yang menjadi pertukaran informasi, algoritma yang dipakai mengacu pada standar di atas.

BAB VIII

TIK:

Mahasiswa dapat Memahami pemrograman web service

Pokok Bahasan:

- Web service pada asp.net dan server iis

Sub Pokok Bahasan:

- IIS, Asp.NET, Web service
- Akses oleh klien

Pada bab ini akan dibahas implementasi dari web service menggunakan asp.net. Skenario yang digunakan adalah:

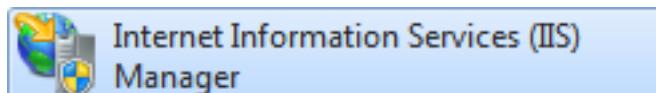
1. Anda memiliki aplikasi web yang dibangun dengan asp.net dan berjalan pada server iis
2. Anda memiliki 2 klien yang membutuhkan data tertentu.
3. Anda menyiapkan Web Service Definition Language (WSDL) untuk melayaninya.
4. Klien Anda mengambil data melalui aplikasi web php dan lainnya melalui aplikasi asp.net
5. Pada contoh permintaan klien sederhana, meminta hasil perkalian dua buah bilangan. Perlu diingat dalam keadaan nyata klien meminta data dari database.
6. Ingat web service tidak selalu perlu GUI (bahkan mungkin diabaikan).

Untuk menjawab kebutuhan di atas,

Penyedia layanan harus memiliki webserver dalam contoh bab ini IIS.

1 IIS, .NET, Web service

IIS berjalan pada O.S windows. Anda dapat mencoba pada komputer lokal.



Jika belum terinstall, silahkan install terlebih dahulu.

Control panel>Programs>Programs and Features. Pilih Internet Information Services.

Catatan: Pada bab ini port IIS akan diganti dari 80 menjadi 81 pada localhost dan port 80 dipersiapkan untuk apache (dengan php).

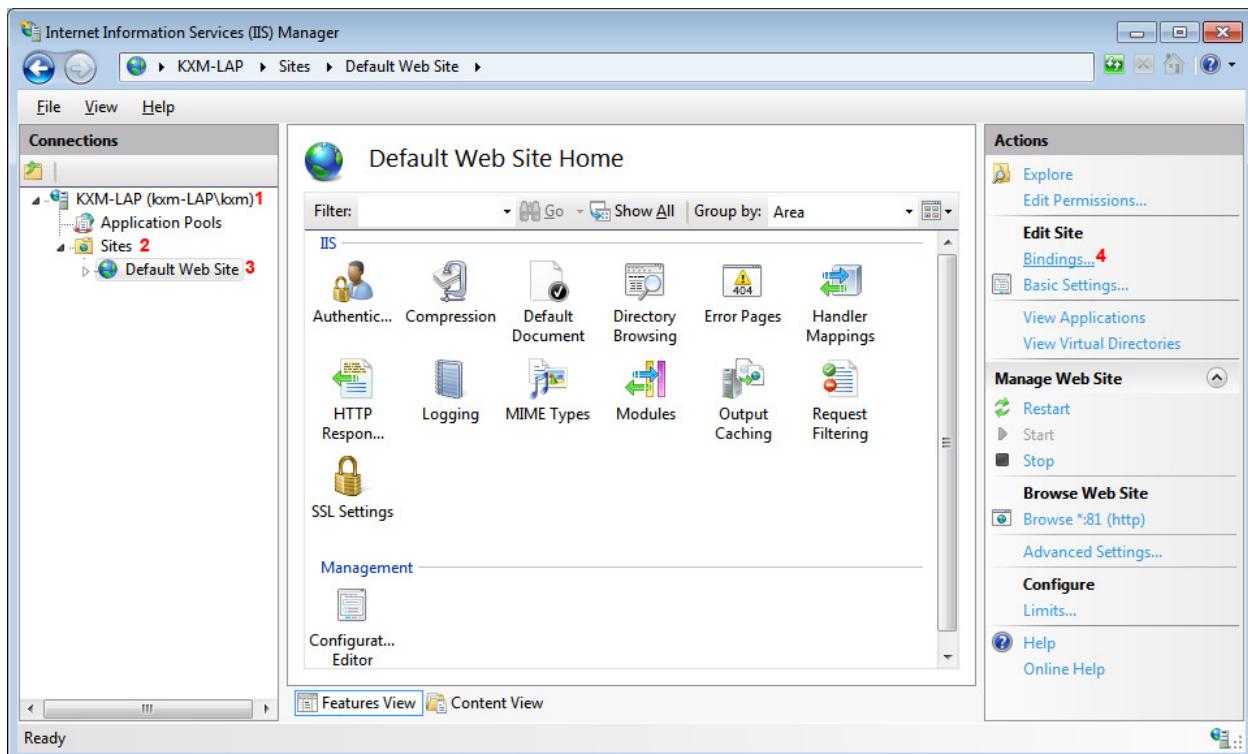
Cara merubah port:

Pada IIS manager, stop terlebih dahulu servicenya

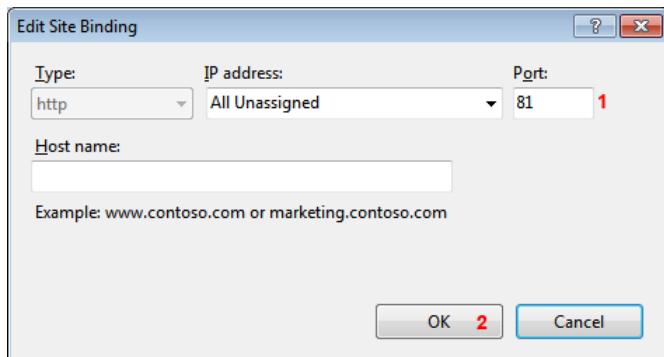


Pada jendela Connections, sites>Default Web Site

Pada jendela Edit site>Bindings



Edit site Bindings, ganti port 80 menjadi 81 klik ok.



Jalankan kembali (start) service, Actions>Manage Server>Start

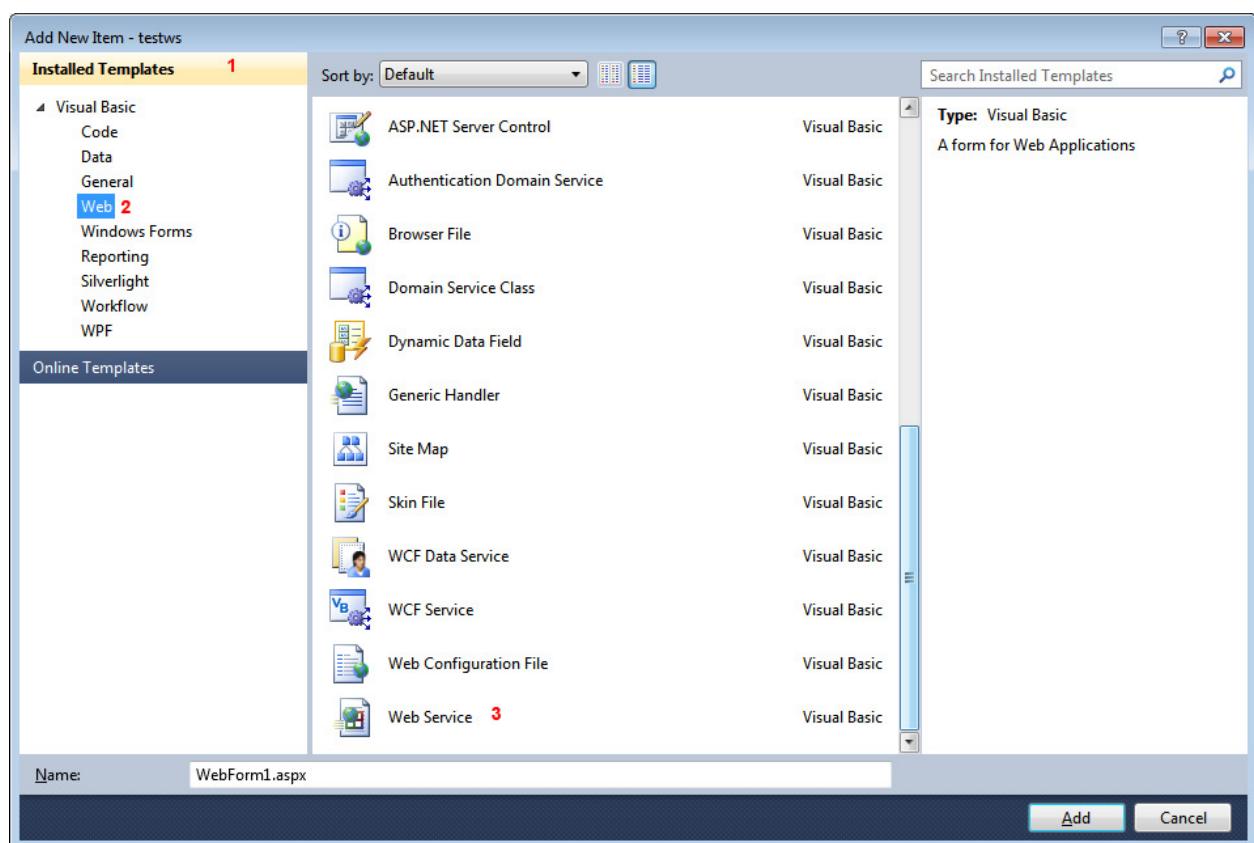
Test konfigurasi tersebut pada browser kesayangan Anda. Ketikan localhost:81



Server IIS telah siap, sekarang Anda akan membuat aplikasi web dan webservice dengan asp.net menggunakan tools IDE visual studio

1. File>New Project

2. Pada bab ini akan menggunakan bahasa VB.net
3. Installed templates>Visual Basic>ASP.NET Web Application dan .NET framework 4
4. Beri nama: testws
5. Ketika Solution project dibangun, visual studio akan membuatkan template halaman muka untuk Anda, untuk sementara diabaikan, fokus pada pembuatan web service.
6. Pada jendela Solution Explorer, klik kanan, Add>New Item
7. Pada installed templates, visual basic>Web, pilih Web Service



8. Beri nama: ws2hitung
9. Perhatikan secara default (pada ws2hitung.asmx), Anda telah memiliki template webmethod HelloWorld
10. Tambahkan method baru pada class ws2hitung sehingga:

1. `Imports System.Web.Services`
2. `Imports System.Web.Services.Protocols`
3. `Imports System.ComponentModel`
- 4.

```

5. ' To allow this Web Service to be called from script, using ASP.NET AJAX,
   uncomment the following line.
6. '<System.Web.Services.ScriptService()> _'
7. <System.Web.Services.WebService(Namespace:="http://tempuri.org/")> _'
8. <System.Web.Services.WebServiceBinding(ConformsTo:=WsiProfiles.BasicProfile1_1)> _'
9. <ToolboxItem(False)> _
10. Public Class ws2hitung
11.     Inherits System.Web.Services.WebService
12.
13.     <WebMethod()> _
14.         Public Function HelloWorld() As String
15.             Return "ws2hitung"
16.         End Function
17.
18.     <WebMethod()>
19.         Public Function kali(ByVal prm1 As Integer, ByVal prm2 As Integer) As Integer
20.             Return prm1 * prm2
21.         End Function
22.
23. End Class

```

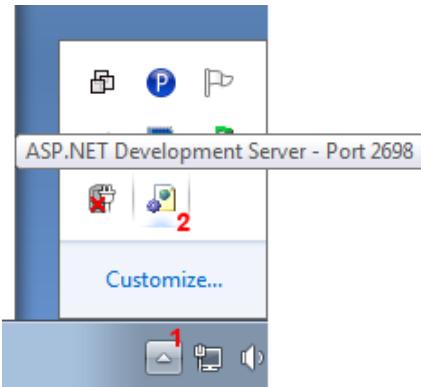
11. Pada contoh di atas Anda akan melayani permintaan klien dengan fungsi kali.
Klien diharuskan mengisi dua parameter untuk mendapatkan hasil.
12. Klik Start debugging (F5)
13. Visual Studio akan menjalankan web aplikasi pada default browser (contoh hasil running: <http://localhost:2698/ws2hitung.asmx>)
14. Terdapat dua service yang tersedia

ws2hitung

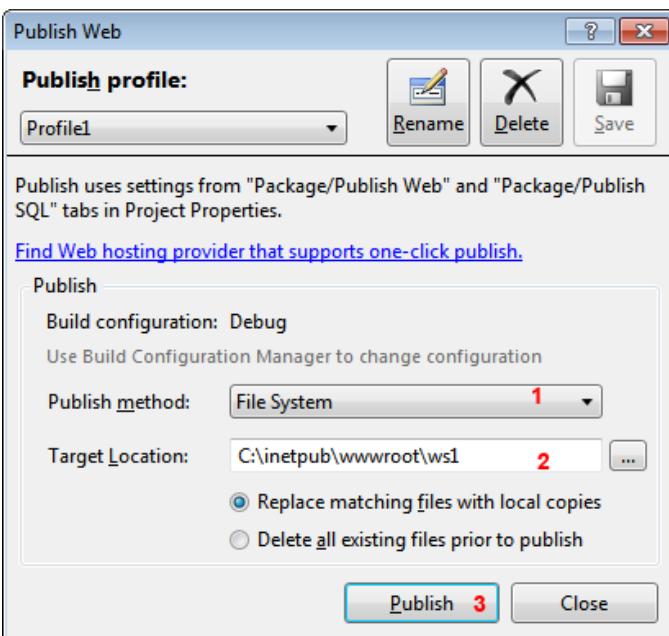
The following operations are supported. For a formal definition, please review the [Service Description](#).

- [HelloWorld](#)
- [kali](#)

15. Namespace dalam contoh ini tidak diubah, pada kondisi ril namespace harus diubah.
16. Cobalah kedua method tersebut.
17. Ketika Anda menggunakan method kali, akan muncul input box: prm1 dan prm2
18. Isikan dan klik invoke
19. Perhatikan url pada link address dan output xml
20. Simpulkan step yang sudah dilalui.
21. Selanjutnya klik stop debugging dan matikan ASP.NET development server



22. Setelah itu Anda akan mempublish aplikasi ini pada localhost port 81
23. Pada jendela Solution Explorer, pada project Anda, klik kanan, Publish
24. Pilih Publish method: File System, Target Location: C:\inetpub\wwwroot\ws1
(contoh pada instalasi xampp apache, secara fisik terdapat pada:
C:\xampp\htdocs)



25. Pada jendela output:

```
===== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped =====
===== Publish: 1 succeeded, 0 failed, 0 skipped =====
```

26. Catatan: problem yang biasa terjadi:

- a. Instalasi Visual Studio baru kemudian IIS akan timbul masalah,

Solusi: Matikan dahulu service IIS. Anda harus menggunakan level Administrator:

Buka command prompt (cmd) akses sebagai administrator (UAC level: Administrator)

Carilah folder framework .NET Anda

Contoh: C:\Windows\Microsoft.NET\Framework\v4.0.30319

Cmd

```
C:\>cd\Windows\Microsoft.NET\Framework\v4.0.30319  
C:\Windows\Microsoft.NET\Framework\v4.0.30319>aspnet_regiis.exe -i
```

- b. Beri hak (permission) pada web.config.

Masih pada folder .NET framework, masuk folder Config

C:\Windows\Microsoft.NET\Framework\v4.0.30319\Config

Bukalah file machine.config (dengan level UAC: Administrator)

Carilah elemen <processModel>

Default:

```
<processModel autoConfig="true" />
```

Ubah menjadi

```
<processModel autoConfig="true" userName="System"/>
```

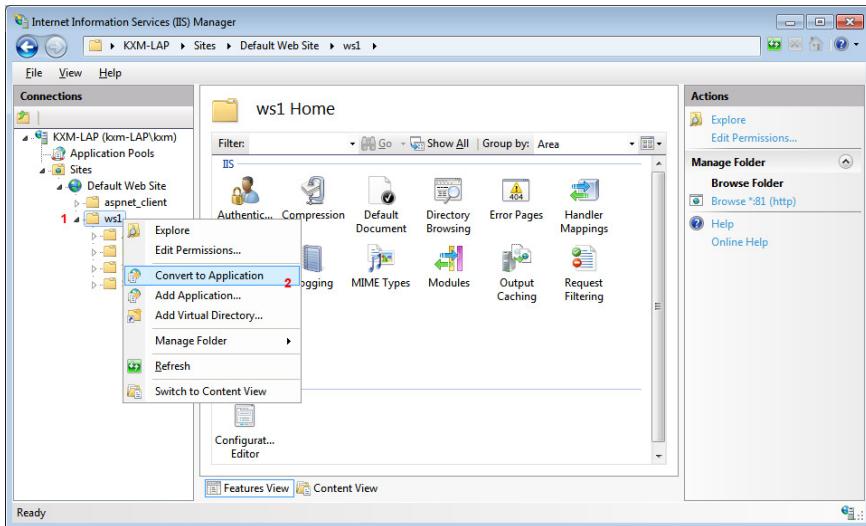
- c. Beri hak akses pada folder:

C:\inetpub

C:\Windows\Temp

- d. Restart IIS

27. Buka kembali IIS manager, pada jendela Connections, Sites>Default Web Site>ws1. Klik kanan. Pilih Convert to Application, OK.



28. Save dan close project dan visual studio Anda.
29. Jalankan pada browser dengan alamat: <http://localhost:81/ws1/>
30. Cek ketersediaan web service Anda dengan, mengubah alamat menjadi:

<http://localhost:81/ws1/ws2hitung.asmx>

31. Klik Service Description

32. Perhatikan url dan output xml

<http://localhost:81/ws1/ws2hitung.asmx?WSDL>

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
  xmlns:tns="http://tempuri.org/" xmlns:s="http://www.w3.org/2001/XMLSchema"
  xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
  xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
  targetNamespace="http://tempuri.org/"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">>
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
      targetNamespace="http://tempuri.org/">
      <s:element name="HelloWorld">
        <s:complexType />
      </s:element>
      <s:element name="HelloWorldResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="HelloWorldResult"
              type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="kali">
    
```

```

<s:complexType>
    <s:sequence>
        <s:element minOccurs="1" maxOccurs="1" name="prm1" type="s:int"
/>
        <s:element minOccurs="1" maxOccurs="1" name="prm2" type="s:int"
/>
    </s:sequence>
</s:complexType>
</s:element>
<s:element name="kaliResponse">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="kaliResult"
type="s:int" />
        </s:sequence>
    </s:complexType>
</s:element>
</s:schema>
</wsdl:types>
<wsdl:message name="HelloWorldSoapIn">
    <wsdl:part name="parameters" element="tns:HelloWorld" />
</wsdl:message>
<wsdl:message name="HelloWorldSoapOut">
    <wsdl:part name="parameters" element="tns:HelloWorldResponse" />
</wsdl:message>
<wsdl:message name="kaliSoapIn">
    <wsdl:part name="parameters" element="tns:kali" />
</wsdl:message>
<wsdl:message name="kaliSoapOut">
    <wsdl:part name="parameters" element="tns:kaliResponse" />
</wsdl:message>
<wsdl:portType name="ws2hitungSoap">
    <wsdl:operation name="HelloWorld">
        <wsdl:input message="tns:HelloWorldSoapIn" />
        <wsdl:output message="tns:HelloWorldSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="kali">
        <wsdl:input message="tns:kaliSoapIn" />
        <wsdl:output message="tns:kaliSoapOut" />
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ws2hitungSoap" type="tns:ws2hitungSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="HelloWorld">
        <soap:operation soapAction="http://tempuri.org/HelloWorld"
style="document" />
        <wsdl:input>
            <soap:body use="literal" />
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal" />
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="kali">
        <soap:operation soapAction="http://tempuri.org/kali" style="document"
/>
        <wsdl:input>

```

```

<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
    <soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:binding name="ws2hitungSoap12" type="tns:ws2hitungSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="HelloWorld">
        <soap12:operation soapAction="http://tempuri.org/HelloWorld"
style="document" />
        <wsdl:input>
            <soap12:body use="literal" />
        </wsdl:input>
        <wsdl:output>
            <soap12:body use="literal" />
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="kali">
        <soap12:operation soapAction="http://tempuri.org/kali" style="document"
/>
        <wsdl:input>
            <soap12:body use="literal" />
        </wsdl:input>
        <wsdl:output>
            <soap12:body use="literal" />
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
<wsdl:service name="ws2hitung">
    <wsdl:port name="ws2hitungSoap" binding="tns:ws2hitungSoap">
        <soap:address location="http://localhost:81/ws1/ws2hitung.asmx" />
    </wsdl:port>
    <wsdl:port name="ws2hitungSoap12" binding="tns:ws2hitungSoap12">
        <soap12:address location="http://localhost:81/ws1/ws2hitung.asmx" />
    </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```

33. Pekerjaan Anda membuat file wsdl telah dicover oleh .NET !

34. Pada file wsdl di atas perhatikan elemen-elemennya

- soap: address location,
- soapAction
- Namespace

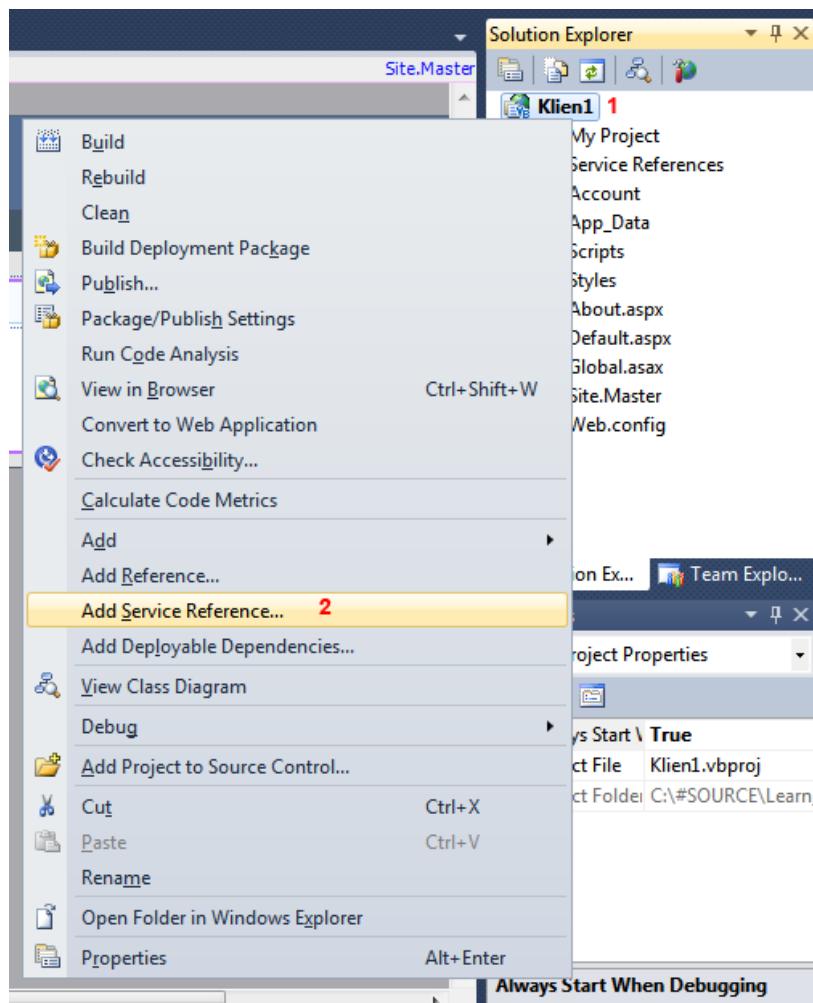
35. Untuk bab-bab selanjutnya file wsdl ini yang akan digunakan contoh

36. Server Anda telah siap melayani klien.

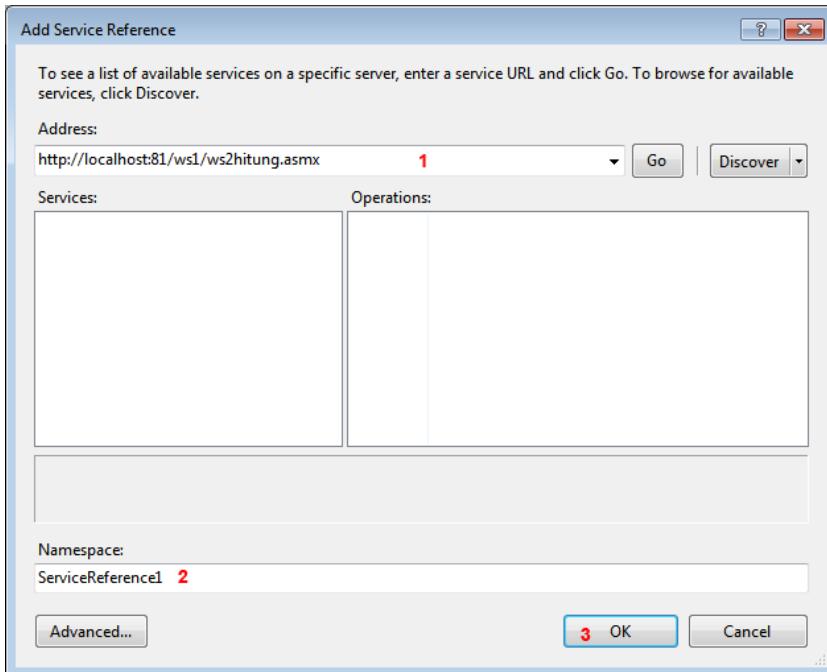
2 Akses oleh klien

Skenario pada klien pertama: komunikasi (service dari server: Anda) melalui web dengan asp.net (aplikasi sebenarnya bisa saja beragam, desktop, mobile dll).

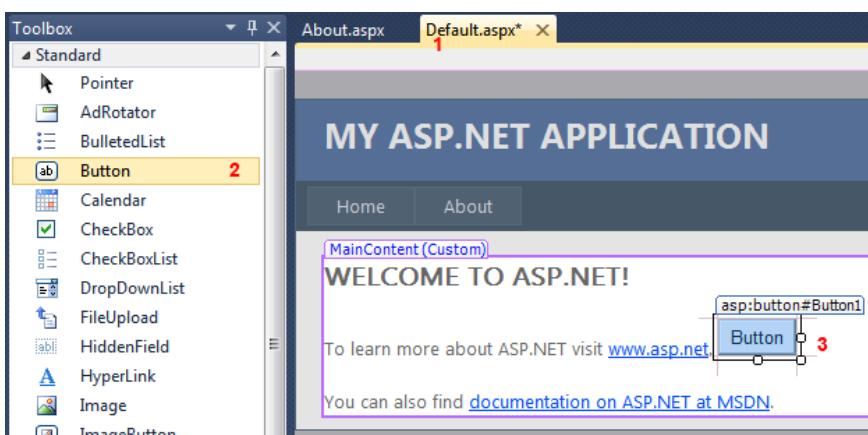
1. Buka IDE Visual Studio
2. File>New Project
3. Framework: .NET framework 4
4. ASP. NET Web Application
5. Beri nama: Klien1
6. Pada Solution Explorer, Klien1 klik kanan >Add Service reference



7. Pada address, ketikan alamat service server Anda:
Contoh: http://localhost:81/ws1/ws2hitung.asmx
8. Klik Ok



9. Untuk keperluan test, klien hanya akan mencoba service kali
10. Pada jendela solution explorer, pilih default.aspx
11. Anda sekarang berada pada design UI.
12. Tambahkan button



13. Double klik button
14. Tambahkan kode pada sub Button1_Click

```

1. Dim tstWS As New ServiceReference1.ws2hitungSoapClient
2.     MsgBox(tstWS.kali(2, 4).ToString())

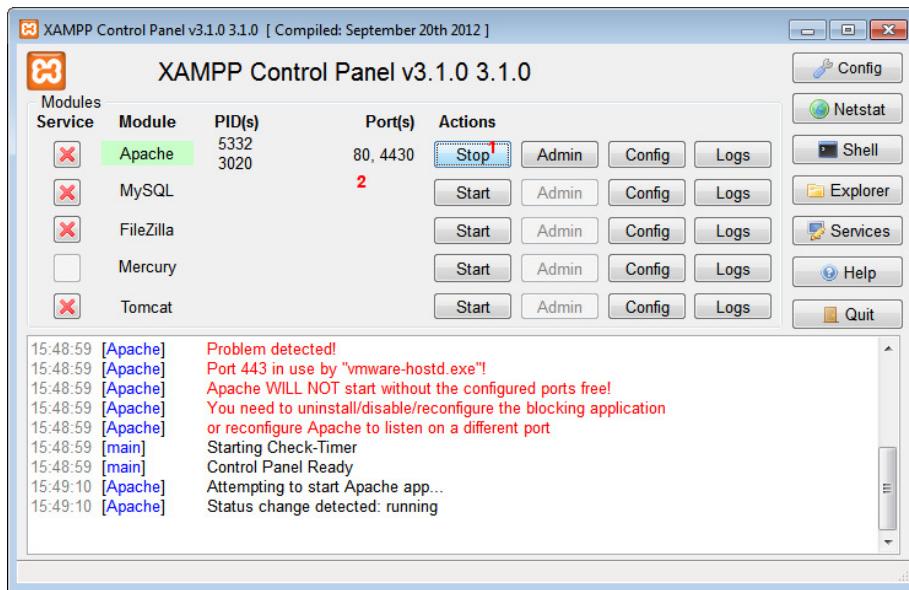
```

15. Klik Start debugging (F5)
16. Perhatikan pada browser address terutama port nya

17. Pada halaman default klik Button
18. Perhatikan popup yang muncul.
19. Catatan dalam kenyataan, service yang diminta klien bisa saja data dalam jumlah besar. Perlu diingat, data yang dipertukarkan dalam bentuk XML, namun Anda tidak melihatnya, semua sudah dicover oleh .NET framework untuk masalah parsing data dlsb.

Skenario pada klien ke dua: komunikasi (service dari server: Anda) melalui web dengan php dan apache webserver (aplikasi sebenarnya bisa saja beragam, desktop, mobile dll).

1. Jika Anda menggunakan xampp, Anda dapat langsung menjalankan service apache untuk menjalankan script php.



2. Dalam contoh ini perhatikan port untuk apache adalah 80 sedangkan IIS 81 dengan ip nama yang sama localhost, dalam kenyataan port 80 sedangkan ip berbeda.
3. Buatlah folder pada htdocs: latihan\ws1
Contoh: C:\xampp\htdocs\latihan\ws1
4. Buatlah file php beri nama: testws1.php
5. Iskan testws1.php

```
1. Tujuan webservice: http://localhost:81/ws1/ws2hitung.asmx/kali
2. <form action='http://localhost:81/ws1/ws2hitung.asmx/kali'
3. method="post" target="_blank">
4. <table>
5.   <tr>
6.     <td>prm1</td>
7.     <td>
8.       <input type="text" size="30" name="prm1">
9.     </td>
10.    </tr>
11.    <tr>
12.      <td>prm2</td>
13.      <td>
```

```

14.    <input type="text" size="30" name="prm2">
15.    </td>
16.  </tr>
17.  <tr>
18.    <td></td>
19.    <td align="right">
20.      <input type="submit" value="Submit" class="button">
21.    </td>
22.  </tr>
23.</table>
24.</form>

```

6. Jalankan pada browser: <http://localhost/latihan/ws1/testws1.php>
7. Output adalah file XML
8. Alternatif yang lain adalah dengan menggunakan library php untuk memarsing nilai return (XML) dari server. Dalam hal ini class yang digunakan adalah SoapClient
9. Buat file php beri nama: testws2.php
10. Isikan

```

1. <?php
2. //Tujuan webservice: http://localhost:81/ws1/ws2hitung.asmx/kali
3. //test kali
4. $wsdl='http://localhost:81/ws1/Ws2hitung.asmx?wsdl';
5. $client2=new SoapClient($wsdl);
6.
7. $hasil=$client2->kali(array('prm1'=>5, 'prm2'=>20));
8. echo $hasil->kaliResult;
9. //die();
10.?>

```

11. Jalankan pada browser: <http://localhost/latihan/ws1/testws2.php>
12. Nilai input berupa array.

Kaitan dengan logistik

Dengan pertumbuhan B2B yang semakin banyak dan cepat dibutuhkan komunikasi dinamis diantaranya (dapat berupa lintas platform dlsb), web service adalah sarana yang tepat untuk menjawabnya.

Dapat Anda perhatikan bahwa sistem yang berbeda dapat dipertukarkan data, nilai yang diterima bebentuk XML sudah cukup terbaca oleh manusia. Kemudian XML tersebut dapat Anda parsing sesuai kebutuhan bisnis Anda.

TUGAS:

- Role play

Buat kelompok maks 4orang/kelompok, masing-masing kelompok membuat aplikasi web server yang menyediakan layanan untuk klien.

Contoh: klien meminta list barang baru merek adidas. Klien tidak boleh memiliki akses terhadap database server, hanya dengan web service.

Spesifikasi pada server: asp.net, iis, sqlserver

Spesifikasi pada klien:php, apache

- Install IDE eclipse JEE untuk keperluan bab berikutnya.

BAB IX

TIK:

Mahasiswa dapat Memahami pemrograman web service

Pokok Bahasan:

- Web service pada php dan server apache

Sub Pokok Bahasan:

- Apache, php 5, Web service
- Akses oleh klien

Pada bab ini akan dibahas implementasi dari web service menggunakan php. Skenario yang digunakan adalah:

1. Anda memiliki aplikasi web yang dibangun dengan php dan berjalan pada server apache
2. Anda memiliki klien yang membutuhkan data tertentu.
3. Anda menyiapkan Web Service Definition Language (WSDL) untuk melayaninya.
4. Klien Anda mengambil data melalui aplikasi web php
5. Pada contoh permintaan klien sederhana, meminta hasil perkalian dua buah bilangan. Perlu diingat dalam keadaan nyata klien meminta data dari database.
6. Ingat web service tidak selalu perlu GUI (bahkan mungkin diabaikan).

Untuk menjawab kebutuhan di atas

1 Apache, php 5, Web service

1. Buatlah folder latihan\ws1 pada direktori htdocs
2. Ingat pada bab-bab sebelumnya SOAP dilakukan dengan mengacu pada file wsdl.
3. Anda perlu membuat file wsdl sebagai jembatan service ini
4. Sebagai contoh, dapat dipergunakan kembali file wsdl hasil generate .NET
5. Buat file wsdl beri nama: hitung.wsdl (isinya mengkopi dari wsdl sebelumnya)

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
   xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
   xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
   xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
   xmlns:tns="http://tempuri.org/" xmlns:s="http://www.w3.org/2001/XMLSchema"
   xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
   xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
   targetNamespace="http://tempuri.org/"
   xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
3.   <wsdl:types>
4.     <s:schema elementFormDefault="qualified"
       targetNamespace="http://tempuri.org/">
5.       <s:element name="HelloWorld">
6.         <s:complexType />
7.       </s:element>
8.       <s:element name="HelloWorldResponse">
9.         <s:complexType>
10.           <s:sequence>
```

```

11.          <s:element minOccurs="0" maxOccurs="1" name="HelloWorldResult"
12.            type="s:string" />
13.        </s:sequence>
14.      </s:complexType>
15.      <s:element name="kali">
16.        <s:complexType>
17.          <s:sequence>
18.            <s:element minOccurs="1" maxOccurs="1" name="prm1" type="s:int" />
19.            <s:element minOccurs="1" maxOccurs="1" name="prm2" type="s:int" />
20.          </s:sequence>
21.        </s:complexType>
22.      </s:element>
23.      <s:element name="kaliResponse">
24.        <s:complexType>
25.          <s:sequence>
26.            <s:element minOccurs="1" maxOccurs="1" name="kaliResult"
27.              type="s:int" />
28.            </s:sequence>
29.          </s:complexType>
30.        </s:element>
31.      </s:schema>
32.    </wsdl:types>
33.    <wsdl:message name="HelloWorldSoapIn">
34.      <wsdl:part name="parameters" element="tns>HelloWorld" />
35.    </wsdl:message>
36.    <wsdl:message name="HelloWorldSoapOut">
37.      <wsdl:part name="parameters" element="tns>HelloWorldResponse" />
38.    </wsdl:message>
39.    <wsdl:message name="kaliSoapIn">
40.      <wsdl:part name="parameters" element="tns:kali" />
41.    </wsdl:message>
42.    <wsdl:message name="kaliSoapOut">
43.      <wsdl:part name="parameters" element="tns:kaliResponse" />
44.    </wsdl:message>
45.    <wsdl:portType name="ws2hitungSoap">
46.      <wsdl:operation name="HelloWorld">
47.        <wsdl:input message="tns>HelloWorldSoapIn" />
48.        <wsdl:output message="tns>HelloWorldSoapOut" />
49.      </wsdl:operation>
50.      <wsdl:operation name="kali">
51.        <wsdl:input message="tns:kaliSoapIn" />
52.        <wsdl:output message="tns:kaliSoapOut" />
53.      </wsdl:operation>
54.    </wsdl:portType>
55.    <wsdl:binding name="ws2hitungSoap" type="tns:ws2hitungSoap">
56.      <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
57.      <wsdl:operation name="HelloWorld">
58.        <soap:operation soapAction="http://tempuri.org>HelloWorld"
59.          style="document" />
60.        <wsdl:input>
61.          <soap:body use="literal" />
62.        </wsdl:input>
63.        <wsdl:output>
64.          <soap:body use="literal" />

```

```

63.      </wsdl:output>
64.    </wsdl:operation>
65.    <wsdl:operation name="kali">
66.      <soap:operation soapAction="http://tempuri.org/kali" style="document" />
67.      <wsdl:input>
68.        <soap:body use="literal" />
69.      </wsdl:input>
70.      <wsdl:output>
71.        <soap:body use="literal" />
72.      </wsdl:output>
73.    </wsdl:operation>
74.  </wsdl:binding>
75.  <wsdl:binding name="ws2hitungSoap12" type="tns:ws2hitungSoap">
76.    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
77.    <wsdl:operation name="HelloWorld">
78.      <soap12:operation soapAction="http://tempuri.org/HelloWorld"
    style="document" />
79.      <wsdl:input>
80.        <soap12:body use="literal" />
81.      </wsdl:input>
82.      <wsdl:output>
83.        <soap12:body use="literal" />
84.      </wsdl:output>
85.    </wsdl:operation>
86.    <wsdl:operation name="kali">
87.      <soap12:operation soapAction="http://tempuri.org/kali" style="document"
    />
88.      <wsdl:input>
89.        <soap12:body use="literal" />
90.      </wsdl:input>
91.      <wsdl:output>
92.        <soap12:body use="literal" />
93.      </wsdl:output>
94.    </wsdl:operation>
95.  </wsdl:binding>
96.  <wsdl:service name="ws2hitung">
97.    <wsdl:port name="ws2hitungSoap" binding="tns:ws2hitungSoap">
98.      <soap:address location="http://localhost/latihan/ws1/server.php" />
99.    </wsdl:port>
100.   <wsdl:port name="ws2hitungSoap12" binding="tns:ws2hitungSoap12">
101.     <soap12:address location="http://localhost/latihan/ws1/server.php"
    />
102.   </wsdl:port>
103. </wsdl:service>
104. </wsdl:definitions>
```

6. Perhatikan elemen soap:address location, berisi alamat service dapat dipergunakan (<http://localhost/latihan/ws1/server.php>). Di sini script php yang akan menangani kebutuhan tersebut adalah server.php
7. Ubah isi elemen tersebut, sesuaikan dengan alamat service Anda
8. Berdasarkan wsdl tersebut, method yang dimiliki adalah:

- HelloWorld
- kali

9. Buatlah file php beri nama: server.php

10. Isikan

```

1. <?php
2. function HelloWorld($isi) {
3.     return 'test';
4. }
5. function kali($symbol) {
6.     $prm1 = $symbol->prm1;
7.     $prm2 = $symbol->prm2;
8.
9.     $hasil=$prm1 * $prm2;
10.
11.    return array('kaliResult'=>$hasil );
12.
13. }
14.
15.ini_set ( "soap.wsdl_cache_enabled", "0" ); // disabling WSDL cache
16.
17.$server = new SoapServer ( "itung.wsdl" );
18.
19.$server->addFunction ( "kali" );
20.$server->addFunction ( "HelloWorld" );
21.
22.$server->handle ();
23.
24.?>
```

11. Dapat dilihat di atas, kelas yang digunakan adalah kelas SoapServer (terdapat pada php 5 dan ke atas)

12. Jalankan di browser: <http://localhost/latihan/ws1/server.php>

2 Akses oleh klien

Sekarang klien akan menggunakan webservice Anda.

Catatan: ini hanya contoh. Pada kondisi nyata: ip dan setup klien-server berbeda.

Pada contoh ini

1. buat file php beri nama: client.php (masih pada direktori latihan\ws1 (CONTOH))
2. pada client.php isikan

```
1. <?php
2.
3. $wsdl='http://localhost:80/latihan/ws1/hitung.wsdl';
4. $client2=new SoapClient($wsdl);
5.
6. $hasil=$client2->kali(array('prm1'=>5, 'prm2'=>20));
7. echo $hasil->kaliResult;
8.
9.
10.
11.
12.?>
```

3. Jalankan pada browser: <http://localhost/latihan/ws1/client.php>
4. Dapat diperhatikan klien menggunakan kelas SoapClient yang bermaksud untuk menggunakan wsdl dan memarsing XML output menjadi string pada contoh di atas.

TUGAS:

Role play:

- Tugas kelompok: dari tugas sebelumnya lakukan pemanfaatan webservice ini lintas kelompok. Setup tiap komputer untuk saling terhubung. Tentukan ip – ip terkait untuk presentasi.

Tugas perorangan: buat 1 file wsdl dengan eclipse dan wtp plugin (pada IDE eclipse JEE)

BAB X

TIK:

Mahasiswa dapat Memahami pemrograman web service

Pokok Bahasan:

- Web service pada Java dan server Apache Tomcat

Sub Pokok Bahasan:

- Apache Tomcat, Java, Web service
- Akses oleh klien

Pada bab ini akan dibahas implementasi dari web service menggunakan Java. Skenario yang digunakan adalah:

1. Anda memiliki aplikasi web yang dibangun dengan Java dan berjalan pada server Apache Tomcat
2. Anda memiliki klien yang membutuhkan data tertentu.
3. Anda menyiapkan Web Service Definition Language (WSDL) untuk melayaninya.
4. Klien Anda mengambil data melalui aplikasi web php
5. Pada contoh permintaan klien sederhana, meminta hasil perkalian dua buah bilangan. Perlu diingat dalam keadaan nyata klien meminta data dari database.
6. Ingat web service tidak selalu perlu GUI (bahkan mungkin diabaikan).
7. Klien Anda menggunakan php dan Java

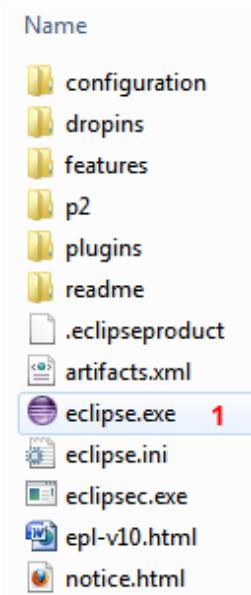
Untuk menjawab kebutuhan di atas

1 Apache Tomcat, Java, Web service

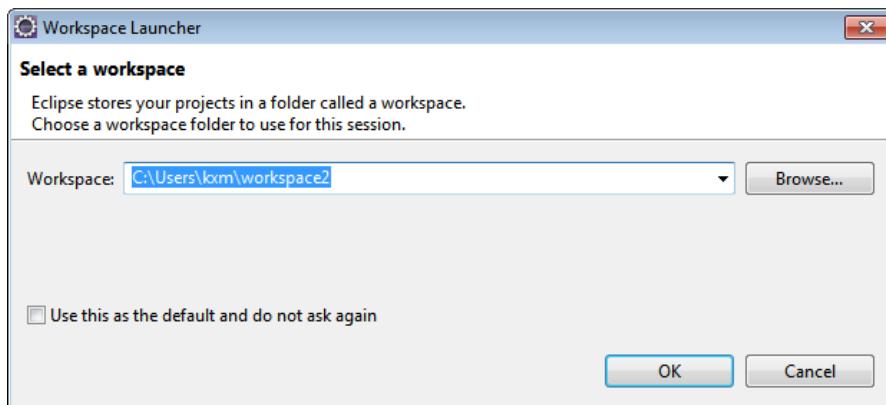
Untuk membangun aplikasi yang baik dan berskala dibutuhkan IDE yang mendukung. Pada bab ini IDE yang digunakan adalah Eclipse Java EE. Seperti halnya Visual Studio, Eclipse juga dipersenjatai dengan plugin-plugin. Hal tersebut dapat memangkas waktu development project berskala.

Pada bab ini dengan menggunakan wizard pada Eclipse, akan dibangun server dan klien untuk test webservice.

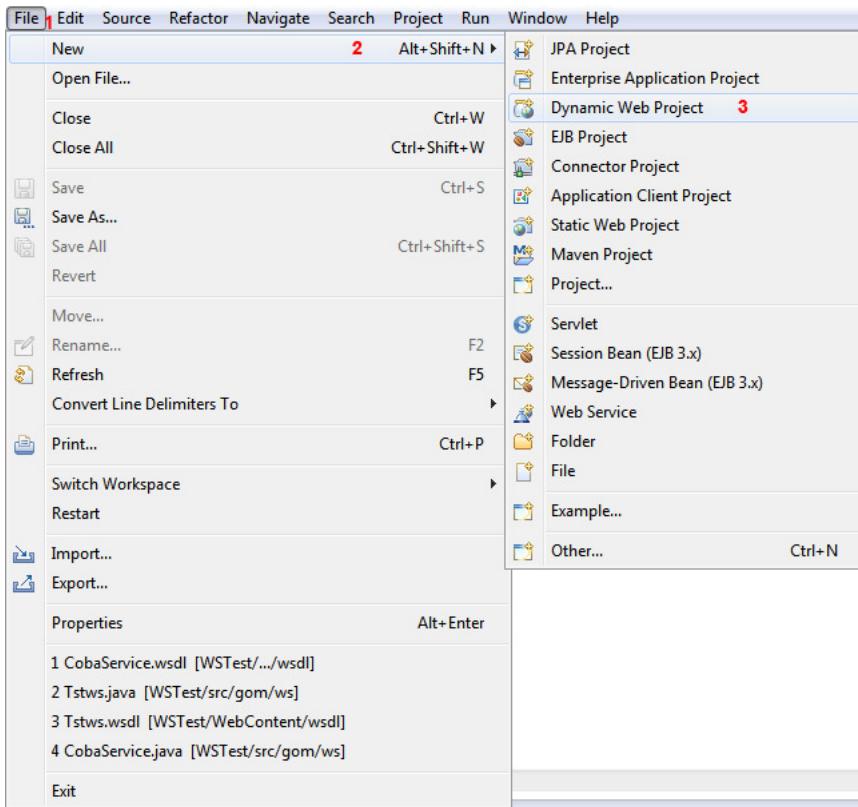
1. Jalankan IDE Eclipse JEE.



2. Tentukan directory tempat project Anda, default adalah workspace, OK



3. File>New>Dynamic Web Project

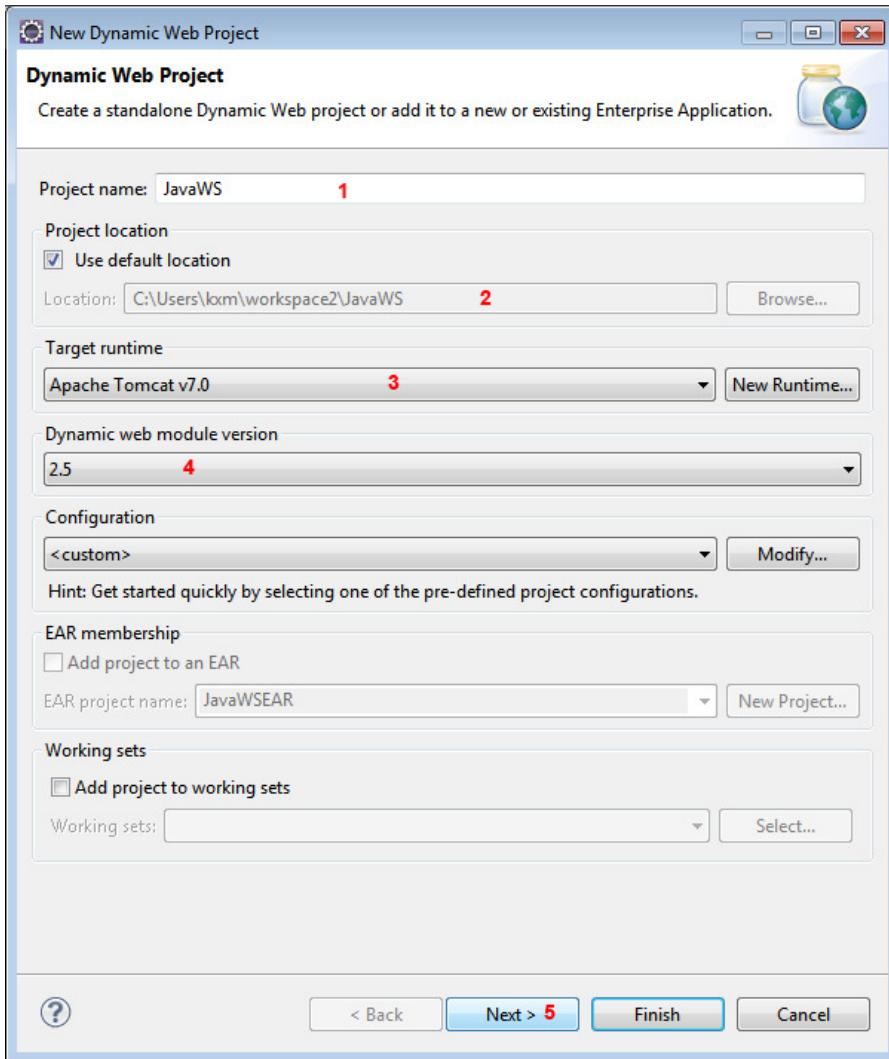


4. Project Name: JavaWS

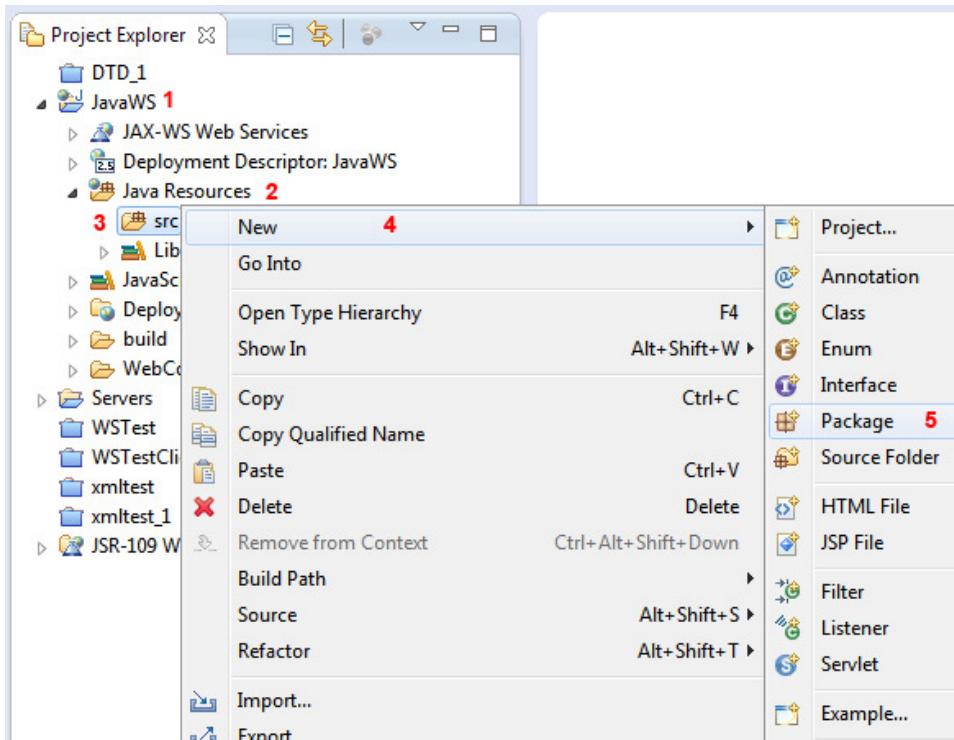
Target Runtime: Apache Tomcat v7.0

Dynamic web module version: 2.5

Next>Next>Finish



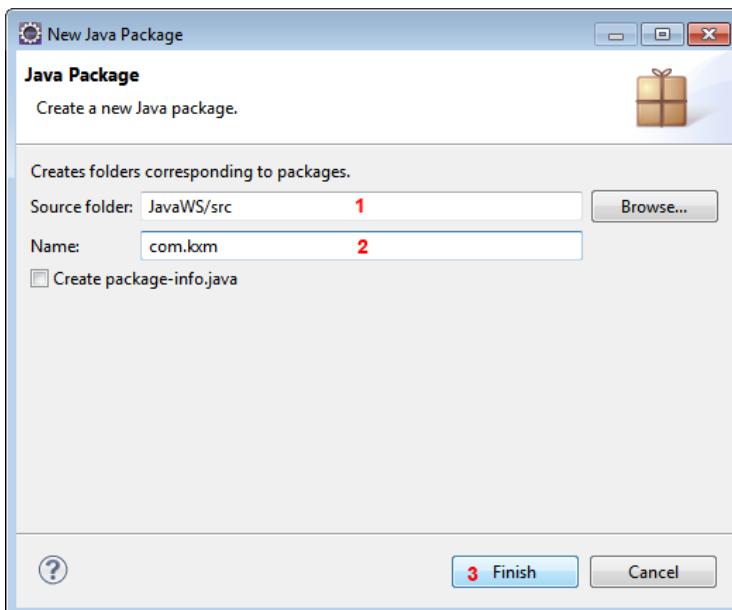
5. Perhatikan jendela Project Explorer
6. Pada Java Resources->src tambahkan package



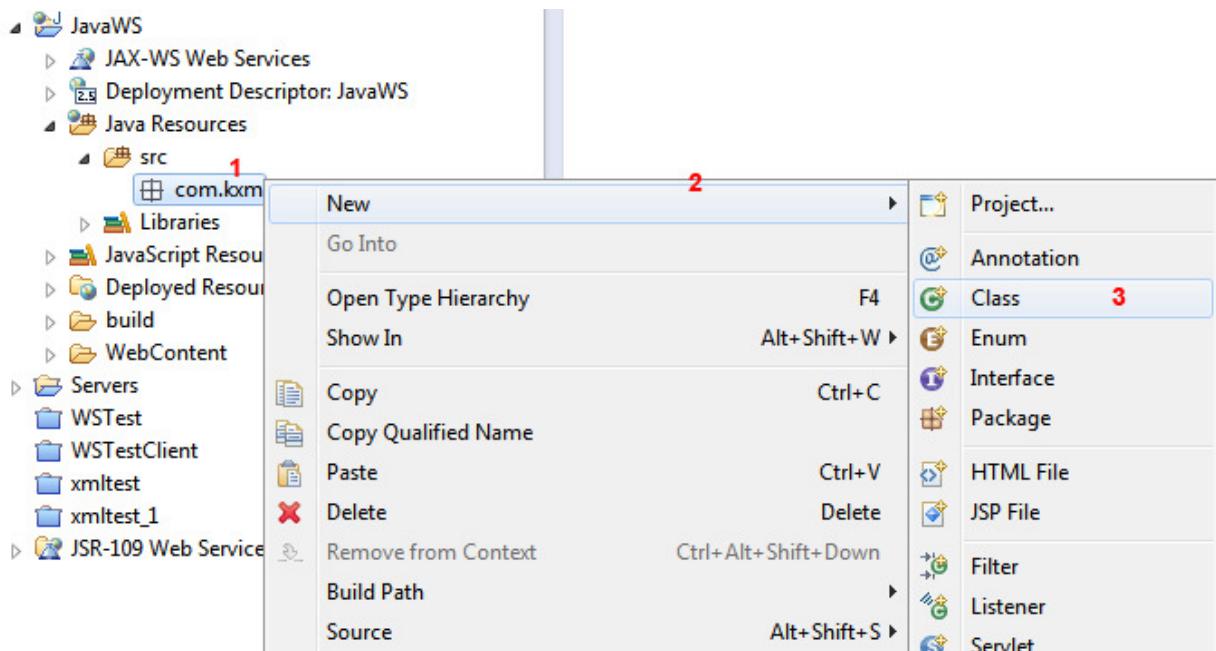
7. Beri nama package, alamat blog Anda

Contoh: com.kxm.ws

Finish



8. Selanjutnya Anda membuat kelas untuk dihubungkan dengan file XML WSDL yang nantinya dikonsumsi oleh klien.
9. Pada package tadi, klik kanan, New>class



10. Beri nama class: CobaService, klik finish

11. Isikan

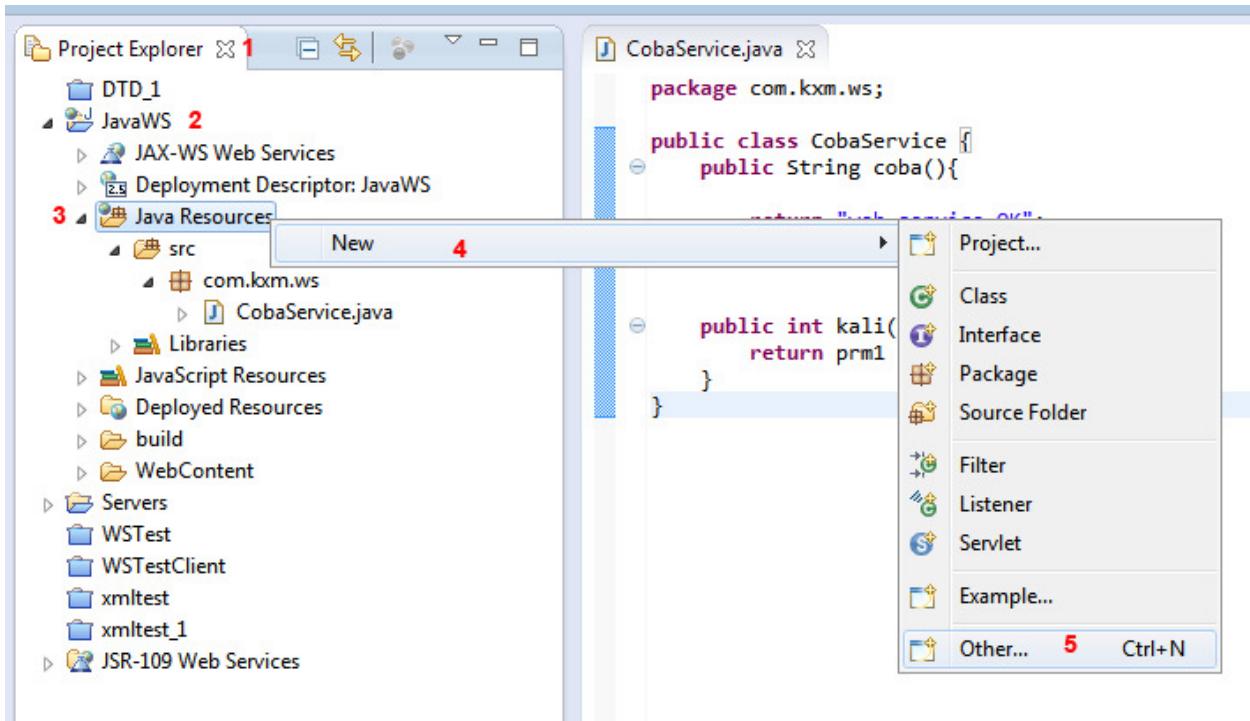
```

1. package com.kxm.ws;
2.
3. public class CobaService {
4.     public String coba(){
5.
6.         return "web service OK";
7.     }
8.
9.
10.    public int kali(int prm1, int prm2){
11.        return prm1 * prm2;
12.    }
13.}
```

12. Jika Anda perhatikan, langkah yang dilalui hampir sama dengan bab sebelumnya ketika menggunakan Visual Studio.

13. Langkah selanjutnya adalah mengenerate webservice, file wsdl (masih melalui wizard)

14. Pada jendela Project Explorer, Project Anda, Java Resources klik kanan.
New>Other>Web Services>Web service



15. Web service type: Bottom up, artinya Anda mengenerate file wsdl dari class yang ada.

16. Pada service implementation: com.kxm.ws.CobaService (package.class)

17. Configuration

Server runtime: Tomcat v7.0 server

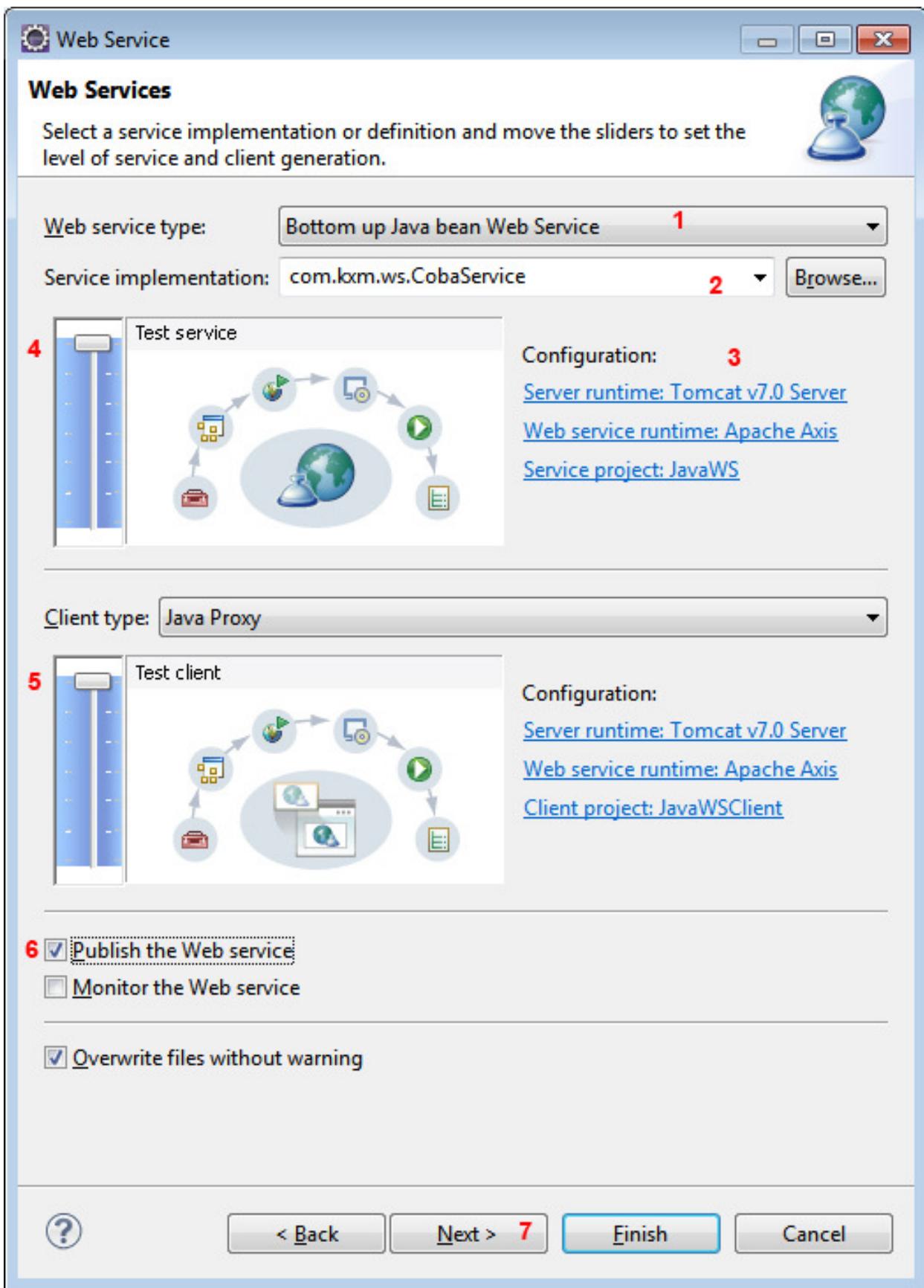
Web service runtime: Apache Axis

Service project: JavaWS

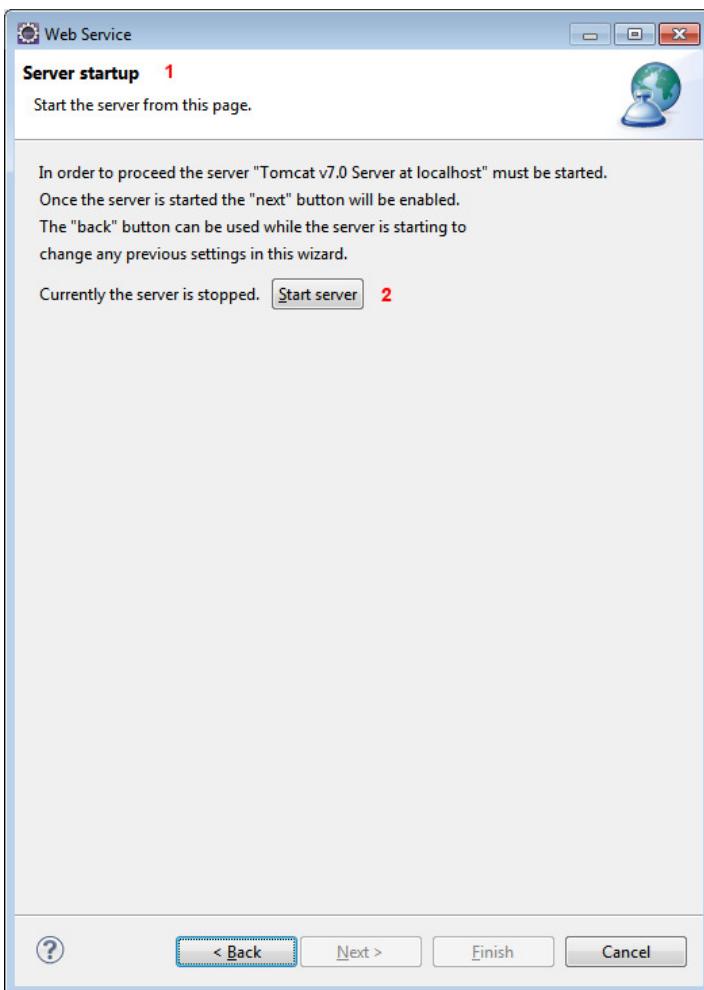
18. Pada server, arahkan slider sehingga Test service

19. Pada client, arahkan slider sehingga Test client

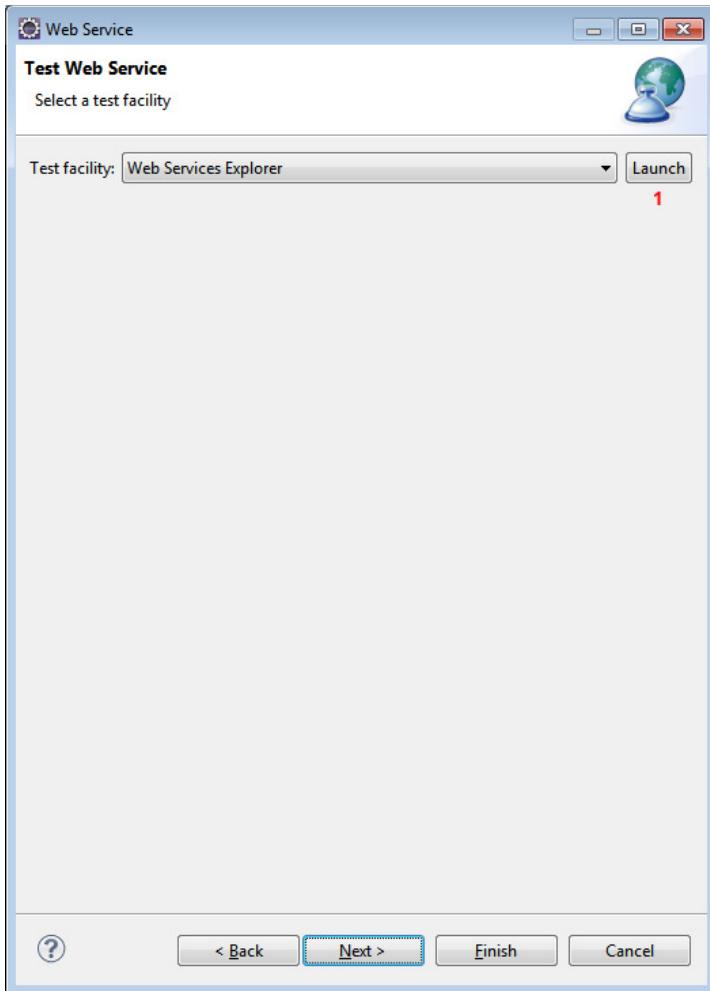
20. Publish the web service



21. Klik next
22. Pilih semua method: coba dan kali dan wsdl file: CobaService.wsdl
23. Next
24. Pada server startup, klik start server



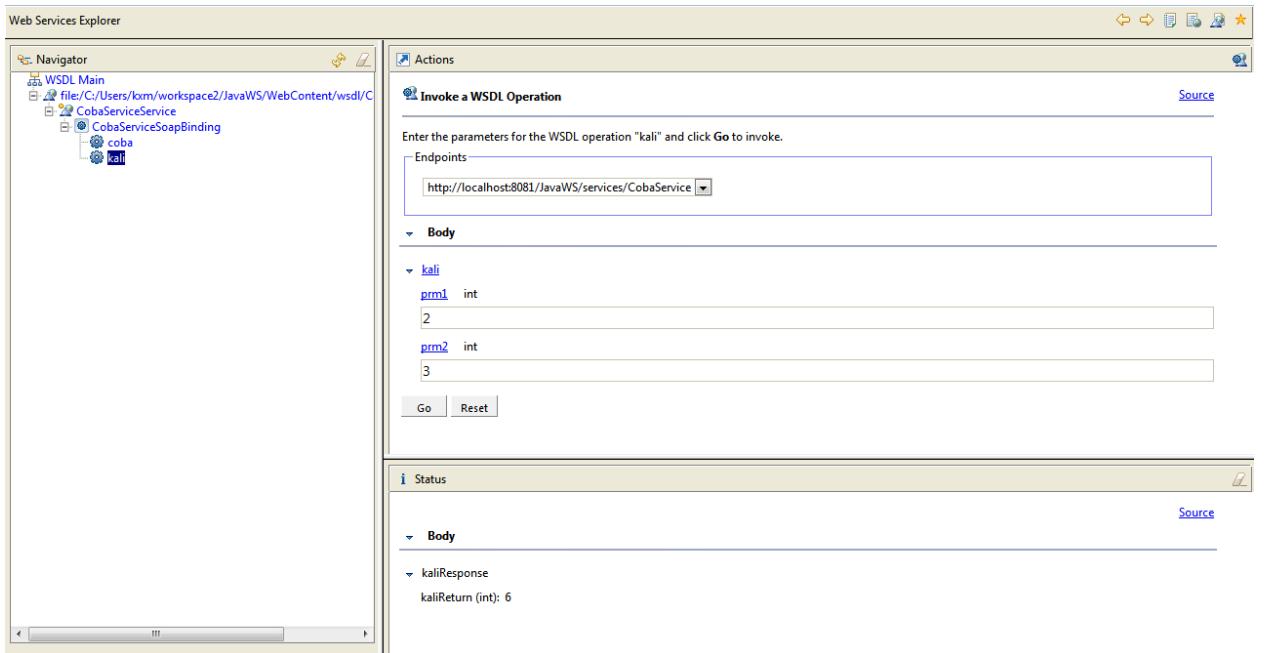
25. Next
26. Test Web Service,



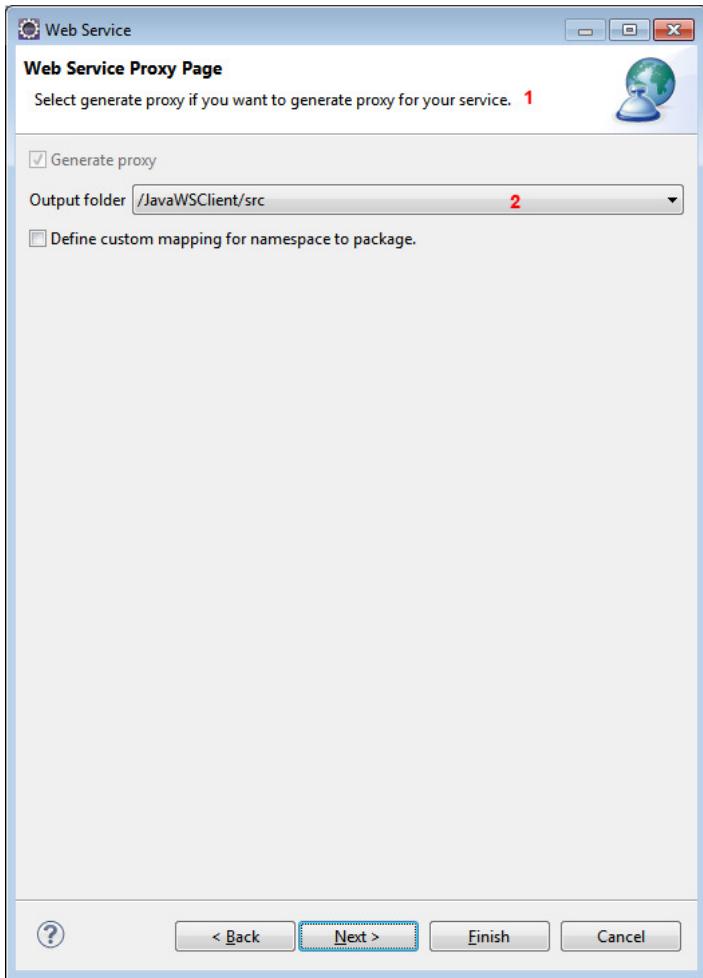
27. Perhatikan browser default Anda

url:

<http://127.0.0.1:9704/wse/wsexplorer/wsexplorer.jsp?org.eclipse.wst.ws.explorer=0>

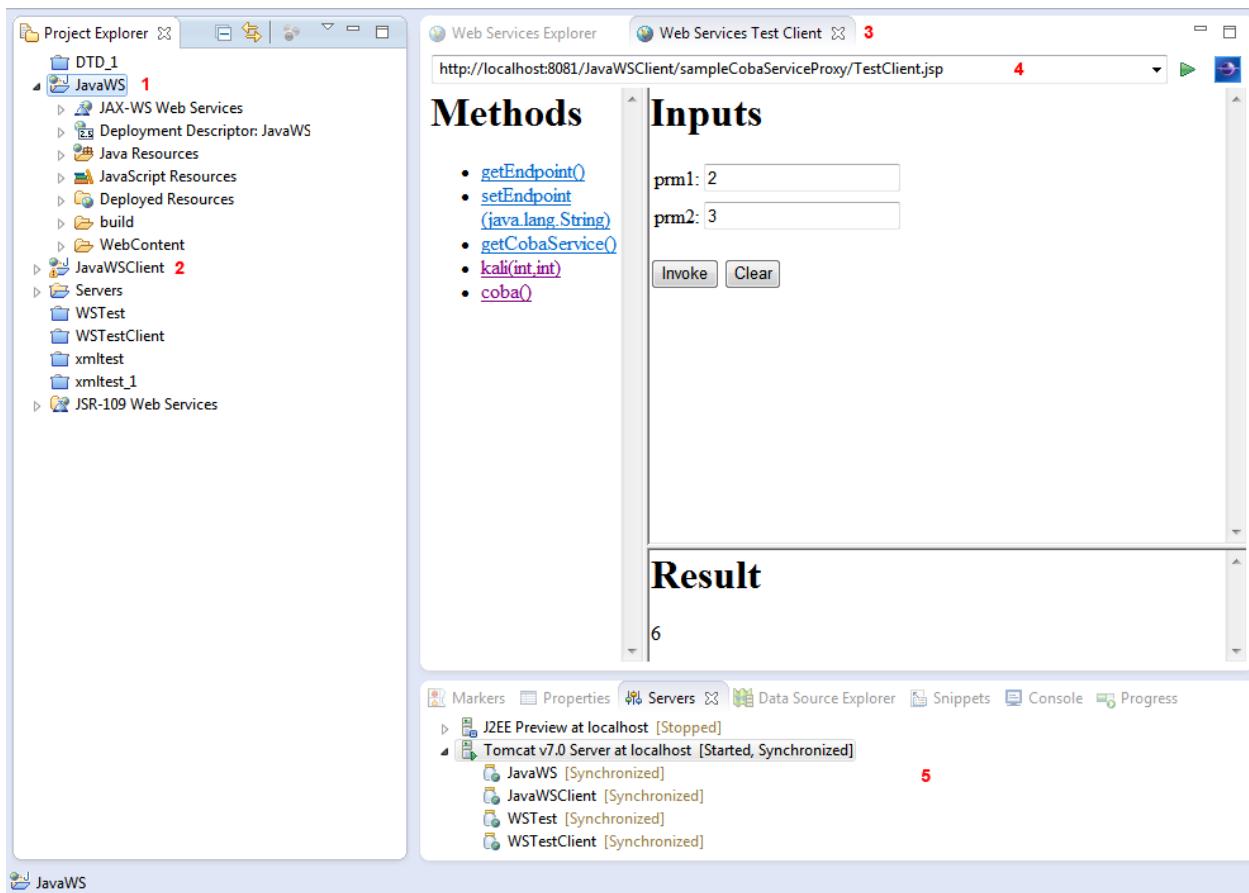


Selanjutnya pada IDE Eclipse, dialog Web Service, klik Next untuk mengenerate project klien (secara otomatis akan membuatkan project JavaWSClient untuk Anda).



2 Akses oleh klien

Telah dijelaskan di atas, IDE Eclipse telah secara otomatis mengenerate project untuk klien.



Berikut contoh kode hasil generate

TestClient.jsp

```
<%@page contentType="text/html; charset=UTF-8"%><HTML>
<HEAD>
<TITLE>Web Services Test Client</TITLE>
</HEAD>
<FRAMESET COLS="220, *">
<FRAME SRC="Method.jsp" NAME="methods" MARGINWIDTH="1" MARGINHEIGHT="1"
SCROLLING="yes" FRAMEBORDER="1">
<FRAMESET ROWS="80%, 20%">
<FRAME SRC="Input.jsp" NAME="inputs" MARGINWIDTH="1" MARGINHEIGHT="1"
SCROLLING="yes" FRAMEBORDER="1">
<%
```

```

StringBuffer resultJSP = new StringBuffer("Result.jsp");
resultJSP.append("?");
java.util.Enumeration resultEnum = request.getParameterNames();while
(resultEnum.hasMoreElements()) {
Object resultObj = resultEnum.nextElement();
resultJSP.append(resultObj.toString()).append("=").append(request.getParameter(result
Obj.toString())).append("&");
}
%>
<FRAME SRC="<%=org.eclipse.jst.ws.util.JspUtils.markup(resultJSP.toString())%>"%
NAME="result" MARGINWIDTH="1" MARGINHEIGHT="1" SCROLLING="yes" FRAMEBORDER="1">
</FRAMESET>
<NOFRAMES>
<BODY>
The Web Services Test Client requires a browser that supports frames.
</BODY>
</NOFRAMES>
</FRAMESET>
</HTML>

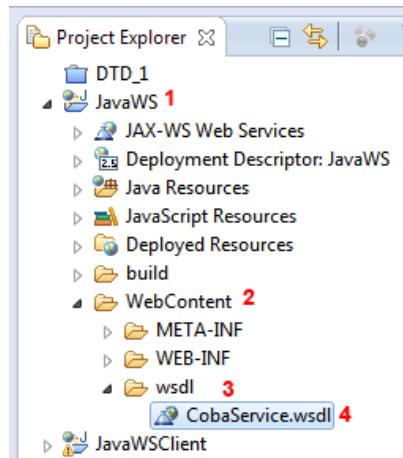
```

Skenario berikutnya adalah jika klien ingin mengakses web service, kurang lebih klien harus memiliki kode seperti dijelaskan pada bab sebelumnya (dengan menggunakan kelas SoapClient).

Namun sebelum itu perlu diketahui kemana klien harus mengakses alamat wsdl?

Untuk itu Anda harus memberi tahu alamat wsdl. Wsdl dapat Anda lihat pada struktur project.

Perhatikan pada project JavaWS->WebContent->wsdl



Jadi wsdl dapat diakses pada: <http://localhost:8081/JavaWS/wsdl/CobaService.wsdl>

Atau melihat Isi file CobaService.wsdl

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://ws.kxm.com"
xmlns:apachesoap="http://xml.apache.org/xml-soap" xmlns:impl="http://ws.kxm.com"
xmlns:intf="http://ws.kxm.com" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<wsdl:types>
  <schema elementFormDefault="qualified" targetNamespace="http://ws.kxm.com"
  xmlns="http://www.w3.org/2001/XMLSchema">
    <element name="kali">
      <complexType>
        <sequence>
          <element name="prm1" type="xsd:int"/>
          <element name="prm2" type="xsd:int"/>
        </sequence>
      </complexType>
    </element>
    <element name="kaliResponse">
      <complexType>
        <sequence>
          <element name="kaliReturn" type="xsd:int"/>
        </sequence>
      </complexType>
    </element>
    <element name="coba">
      <complexType/>
    </element>
    <element name="cobaResponse">
      <complexType>
        <sequence>
          <element name="cobaReturn" type="xsd:string"/>
        </sequence>
      </complexType>
    </element>
  </schema>
</wsdl:types>

<wsdl:message name="cobaResponse">
  <wsdl:part element="impl:cobaResponse" name="parameters">
  </wsdl:part>
</wsdl:message>

<wsdl:message name="kaliRequest">
  <wsdl:part element="impl:kali" name="parameters">
  </wsdl:part>
```

```
</wsdl:message>

<wsdl:message name="kaliResponse">
    <wsdl:part element="impl:kaliResponse" name="parameters">
        </wsdl:part>
    </wsdl:message>
<wsdl:message name="cobaRequest">
    <wsdl:part element="impl:coba" name="parameters">
        </wsdl:part>
    </wsdl:message>
<wsdl:portType name="CobaService">
    <wsdl:operation name="kali">
        <wsdl:input message="impl:kaliRequest" name="kaliRequest">
        </wsdl:input>
        <wsdl:output message="impl:kaliResponse" name="kaliResponse">
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="coba">
        <wsdl:input message="impl:cobaRequest" name="cobaRequest">
        </wsdl:input>
        <wsdl:output message="impl:cobaResponse" name="cobaResponse">
        </wsdl:output>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="CobaServiceSoapBinding" type="impl:CobaService">
    <wsdlsoap:binding style="document"
        transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="kali">
        <wsdlsoap:operation soapAction="" />
```

```

<wsdl:input name="kaliRequest">
    <wsdlsoap:body use="literal"/>
</wsdl:input>

<wsdl:output name="kaliResponse">
    <wsdlsoap:body use="literal"/>
</wsdl:output>

</wsdl:operation>

<wsdl:operation name="coba">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="cobaRequest">
        <wsdlsoap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="cobaResponse">
        <wsdlsoap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>

</wsdl:binding>

<wsdl:service name="CobaServiceService">
    <wsdl:port binding="impl:CobaServiceSoapBinding" name="CobaService">
        <wsdlsoap:address
location="http://localhost:8081/JavaWS/services/CobaService"/>
    </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```

Jadi wsdl dapat diakses pada:

<http://localhost:8081/JavaWS/services/CobaService?wsdl>

Sehingga kode untuk php kurang lebih:

1. Buat file php pada direktori latihan/ws1
2. Beri nama: javatestws.php
3. Isikan kode:

```
1. <?php
2. // test je java
3. // lihat struktur WebContent
4. // $wsdl = 'http://localhost:8081/JavaWS/wsdl/CobaService.wsdl';
5. // lihat struktur wsdl
6. $wsdl="http://localhost:8081/JavaWS/services/CobaService?wsdl";
7. $client2 = new SoapClient ( $wsdl );
8.
9. $hasil = $client2->coba ();
10. echo $hasil->cobaReturn;
11.
12. echo '</br>';
13.
14. $hasil2 = $client2->kali ( array (
15.                 'prm1' => 5,
16.                 'prm2' => 20
17.             ) );
18. echo $hasil2->kaliReturn; // lihar struktur CobaService.wsdl di project javaEE
    Anda
19.                                         // print_r( $client2->coba());
20.
21.?>
```

4. Jalankan pada browser: <http://localhost/latihan/ws1/javatestws.php>
5. Tariklah kesimpulan.

TUGAS:

Kelompok: Buat makalah mengenai web service pada Java, meliputi:

- Sejarah
- Arsitektur
- Perkembangan, pengguna
- Keamanan

(minimal 12 halaman)

BAB XI

TIK:

Mahasiswa dapat Memahami penggunaan REST, JSON sebagai web service

Pokok Bahasan:

- REST, JSON

Sub Pokok Bahasan:

- Sekilas REST
- JSON
- Implementasi (php-php)

1 Sekilas REST

REST merupakan style arsitektur perangkat lunak. Seperti dijelaskan dalam disertasi oleh Roy Fielding, REST adalah "architectural style" yang pada dasarnya memanfaatkan teknologi yang sudah ada dan protokol web.

Perlu Anda ingat web service merupakan SOA Middleware Interoperability Standards, sedangkan REST merupakan Architectural style for the Web.

Representasi format yang umum dipakai REST adalah JavaScript Object Notation (JSON) . Pada REST tidak ada WSDL.

Contoh:

JSON

```
{"menu": {  
    "id": "file",  
    "value": "File",  
    "popup": {  
        "menuitem": [  
            {"value": "New", "onclick": "CreateNewDoc()"},  
            {"value": "Open", "onclick": "OpenDoc()"},  
            {"value": "Close", "onclick": "CloseDoc()"}  
        ]  
    }  
}}
```

XML:

```
<menu id="file" value="File">  
  <popup>  
    <menuitem value="New" onclick="CreateNewDoc()" />  
    <menuitem value="Open" onclick="OpenDoc()" />  
    <menuitem value="Close" onclick="CloseDoc()" />  
  </popup>  
</menu>
```

2 JSON

JavaScript Object Notation (JSON) adalah format untuk pertukaran data yang ringan. Mudah dimengerti oleh manusia dan ringan bagi mesin untuk proses parse dan generate. Hal ini didasarkan pada subset dari JavaScript Programming Language, Standar ECMA-262 Edisi 3 - Desember 1999. JSON adalah format teks yang independen tetapi menggunakan konvensi yang akrab bagi programmer dari varian

bahasa C, termasuk C, C + +, C #, Java, JavaScript, Perl, Python,dll. Hal ini mengakibatkan JSON sebagai bahasa data-interchange yang ideal.

JSON dibangun di atas dua struktur:

- Sebuah koleksi pasangan nama / nilai. Dalam berbagai bahasa dikenal sebagai suatu object, record, struct, dictionary, hash table, keyed list, atau associative array.
- Daftar ordered nilai, dalam banyak bahasa dikenal dengan istilah array, vector, list, atau sequence.

Informasi terbaru dan lebih lengkap dapat Anda lihat pada <http://www.json.org>

3 Implementasi

Untuk memahami contoh penggunaan REST dan JSON diketahui:

Anda memiliki perusahaan manufaktur yang aktif berproduksi, rekanan Anda (klien) ingin mengetahui jumlah barang berdasarkan list kode barang. Rancanglah web service menggunakan REST JSON.

Sebagai contoh, kedua pihak menggunakan teknologi yang sama (tapi berbeda framework dan ISP), server side programming menggunakan php.

Hal pertama yang harus dilakukan adalah menyediakan pelayanan (server).

1. Hidupkan server apache
2. Masih seperti pada bab sebelumnya, workspace direktori adalah:
C:\xampp\htdocs\latihan\ws1
3. Buat file php, beri nama: testjsonserver.php
4. Isikan

```
1. <?php
2. // echo
'<pre>'.print_r(json_decode(file_get_contents("php://input")),1).'</pre>';
```

```

3. $reques = json_decode ( file_get_contents ( "php://input" ) );
4.
5. $r_kdBrang = $reques->data->kdBarang;
6. $r_id = $reques->data->ID;
7.
8. // proses dengan data/database
9. // contoh sederhana
10.
11. switch ($r_kdBrang) {
12.     case 'MG1111' :
13.         $jumlah = 10;
14.         break;
15.     case 'MG1122' :
16.         $jumlah = 2220;
17.         break;
18.     case 'MG007' :
19.         $jumlah = 1;
20.         break;
21.     default :
22.         $jumlah = 0;
23.         break;
24. }
25.
26. echo $jumlah;
27.?>

```

5. Terlihat dari contoh di atas kode barang beserta jumlahnya (hardcoded).
6. File testjsonserver.php inilah yang akan melayani klien, yang perlu dikirimkan oleh klien adalah 2 parameter.

Berikutnya adalah membuat program disisi klien. Ingat klien tidak memiliki akses terhadap database.

1. Untuk contoh, lokasi fisik klien dan server sama, C:\xampp\htdocs\latihan\ws1
2. Buat file php, beri nama: Lib_ws.php
3. Isikan:

```

1. <?php
2. class c_ws_json {
3.     private $pesan = '';
4.     public function cekBarang($kdBarang, $id) {
5.         // web services dengan teknik curl, json
6.         define ( "ALAMAT", "http://localhost/latihan/ws1" );
7.         $url = ALAMAT . "/testjsonserver.php";
8.
9.         $message = array (
10.             "kdBarang" => $kdBarang,
11.             "ID" => $id
12.         );
13.

```

```

14.      $fields = array (
15.          'data' => $message
16.      )
17.      ;
18.      $headers = array (
19.          'Content-Type: application/json'
20.      );
21.
22.      // Open connection
23.      $ch = curl_init ();
24.
25.      // Set the url, number of POST vars, POST data
26.      curl_setopt ( $ch, CURLOPT_URL, $url );
27.
28.      curl_setopt ( $ch, CURLOPT_POST, true );
29.      curl_setopt ( $ch, CURLOPT_HTTPHEADER, $headers );
30.      curl_setopt ( $ch, CURLOPT_RETURNTRANSFER, true );
31.
32.      // Disabling SSL Certificate support temporarily
33.      curl_setopt ( $ch, CURLOPT_SSL_VERIFYPeer, false );
34.
35.      curl_setopt ( $ch, CURLOPT_POSTFIELDS, json_encode ( $fields ) );
36.
37.      // Execute post
38.      $result = curl_exec ( $ch );
39.      if ($result === FALSE) {
40.          die ( 'Curl failed: ' . curl_error ( $ch ) );
41.      }
42.
43.      // Close connection
44.      curl_close ( $ch );
45.
46.      return $result;
47.  }
48. }
49.?>

```

4. Pada contoh di atas, perhatikan penggunaan curl.
5. Kemudian buat file php, beri nama: testjsonklien.php
6. Isikan:

```

1. <?php
2. include_once 'Lib_ws.php';
3.
4. $oc_ws_json = new c_ws_json ();
5. echo $oc_ws_json->cekBarang ( 'MG007', 'Budi' );
6.
7. ?>

```

7. Jalankan testjsonklien pada browser:
<http://localhost/latihan/ws1/testjsonklien.php>

8. Cobalah untuk mengubah kode barang pada testjsonklien, jalankan untuk mengetahui respon server.

Kaitan dengan logistik

Dengan pertumbuhan B2B yang semakin banyak dan cepat dibutuhkan komunikasi dinamis diantaranya (dapat berupa lintas platform dlsb), web service adalah sarana yang tepat untuk menjawabnya. Informasi bisa saja diakses melalui mobile, yang perlu diperhatikan adalah penggunaan sumber daya.

Ukuran file JSON dan gaya REST lebih ringan dibandingkan dengan XML dan SOAP, hal ini berdampak pada kegiatan parse dan generate yang lebih ringan bagi mesin sehingga output dapat dihadirkan dengan cepat dan konsumsi sumber daya lebih sedikit.

TUGAS:

Kelompok.

Pada bab sebelumnya Anda berkomunikasi dengan wsdl, kembangkan kembali. Sehingga data yang ditarik dari database berbentuk JSON.

BAB XII

TIK:

Mahasiswa dapat Memahami penggunaan REST, JSON sebagai web service

Pokok Bahasan:

- REST, JSON

Sub Pokok Bahasan:

- Implementasi REST,JSON (php-Android)

Pada bab sebelumnya telah dibahas web service dengan aplikasi server menggunakan php. Dengan akses internet yang relatif mudah didapat dan teknologi mobile yang lekat dengan kehidupan sehari-hari, konsumsi informasi dapat dipenuhi dengan cepat. Satu hal penting bahwa mobile (smart phone) memiliki keterbatasan akan sumber daya.

Anda sudah dapat membedakan antara JSON dan XML(WSDL). Dari segi ukuran data danstruktur, JSON lebih bersahabat terhadap masalah sumber daya.

Pada bab ini akan dibahas web service yang terjadi antara server (php, apache) dan klien (Android).

1 Implementasi

Untuk memahami contoh penggunaan REST dan JSON diketahui:

Anda memiliki perusahaan manufaktur yang aktif berproduksi, rekanan Anda (klien) ingin mengetahui jumlah barang berdasarkan list kode barang. Rancanglah web service menggunakan REST JSON.

Server: php, apache

Klien: android

Hal pertama yang harus dilakukan adalah menyediakan pelayanan (server).

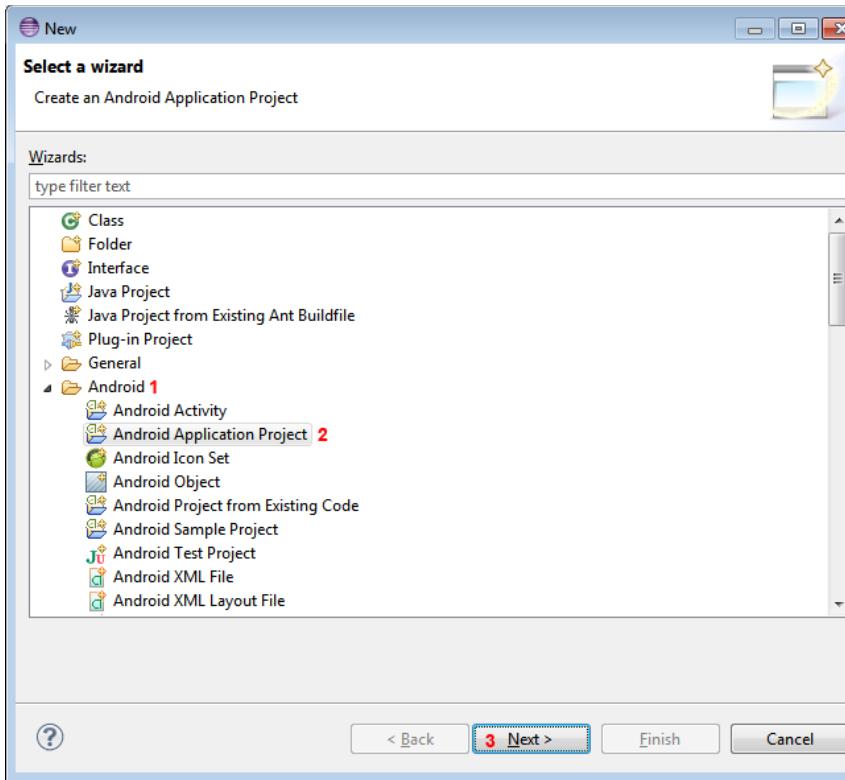
1. Hidupkan server apache
2. Masih seperti pada bab sebelumnya, workspace direktori adalah:
C:\xampp\htdocs\latihan\ws1
3. Buat file php, beri nama: testjsonserver.php
4. Isikan

```
1. <?php
2. // echo
'<pre>'.print_r(json_decode(file_get_contents("php://input")),1).'</pre>';
3. $request = json_decode ( file_get_contents ( "php://input" ) );
```

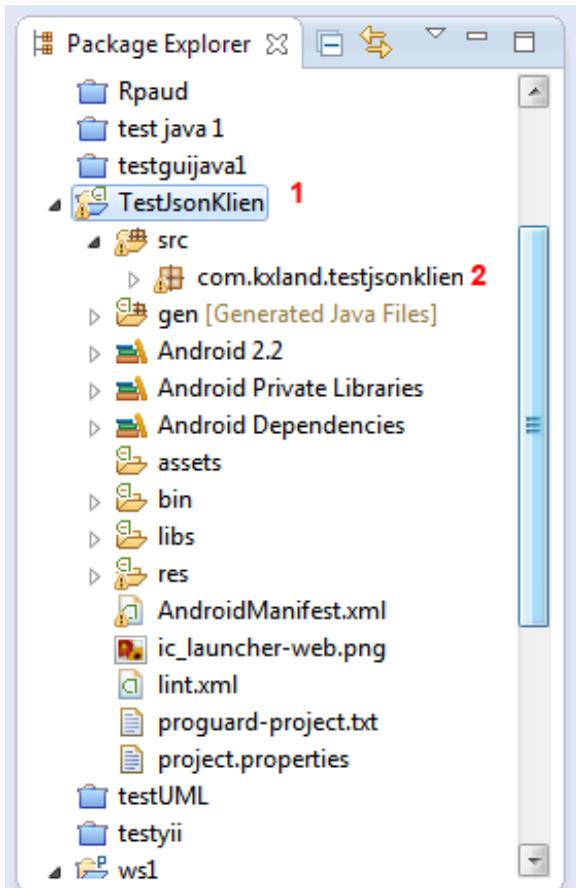
```
5. $r_kdBrag = $reques->data->kdBbarang;
6. $r_id = $reques->data->ID;
7.
8. // proses dengan data/database
9. // contoh sederhana
10.
11. switch ($r_kdBrag) {
12.     case 'MG1111' :
13.         $jumlah = 10;
14.         break;
15.     case 'MG1122' :
16.         $jumlah = 2220;
17.         break;
18.     case 'MG007' :
19.         $jumlah = 1;
20.         break;
21.     default :
22.         $jumlah = 0;
23.         break;
24. }
25.
26. echo $jumlah;
27.?>
```

Pada sisi klien, Anda dapat membuat project Android dengan beragam tools. Di sini akan digunakan IDE Eclipse. Catatan: ADT diasumsikan sudah terinstall

1. Buka Eclipse
2. File>new>Other>Android>Android Application Project

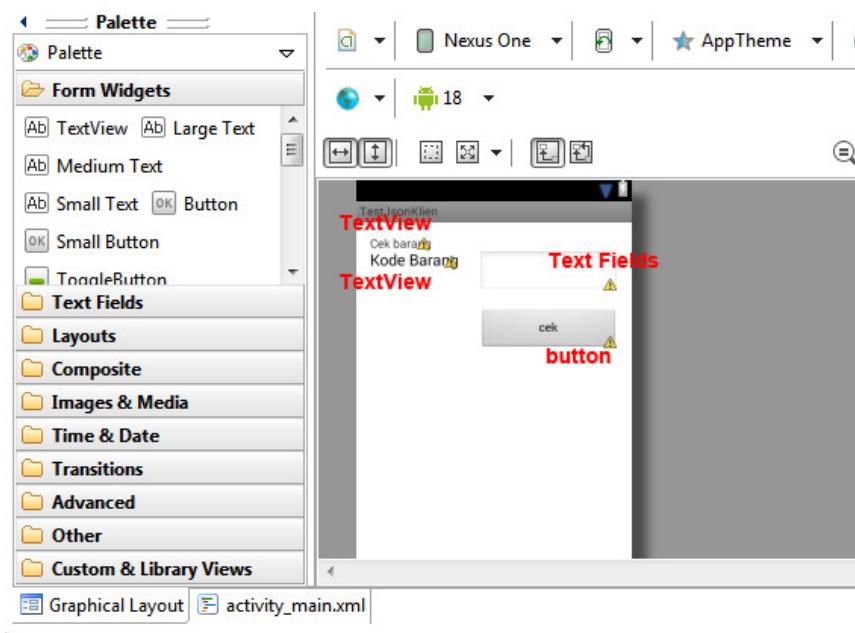


3. Beri nama aplikasi: TestJsonKlien
4. Beri nama package: com.polpos. testjsonklien
5. Next, Next
6. Create Activity>Blank Activity
7. Activity Name: MainActivity
8. Finish
9. Perhatikan jendela Package Explorer



10. Pada res>layout>Activity_main.xml

Ubahlah design sehingga menjadi:



Source:

```

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Cek barang" />

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView2"
        android:text="Kode Barang"
        android:textAppearance="?android:attr/textAppearanceMedium" />

    <EditText
        android:id="@+id/kdbrgTxt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignTop="@+id/textView1"
        android:layout_marginLeft="24dp"
        android:layout_toRightOf="@+id/textView1"
        android:ems="10" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/kdbrgTxt"
        android:layout_alignRight="@+id/kdbrgTxt"
        android:layout_below="@+id/kdbrgTxt"
        android:layout_marginTop="19dp"
        android:onClick="CekJmlBrg"
        android:text="cek" />

</RelativeLayout>

```

11. Pada jendela Package Explorer

TestJsonKlien>src>com.polpos. testjsonklien>MainActivity.java

12. Isikan

1. package com.polpos. testjsonklien;

```
2.  
3. import java.io.BufferedReader;  
4. import java.io.IOException;  
5. import java.io.InputStream;  
6. import java.io.InputStreamReader;  
7.  
8. import org.apache.http.HttpResponse;  
9. import org.apache.http.client.ClientProtocolException;  
10.import org.apache.http.client.HttpClient;  
11.import org.apache.http.client.methods.HttpPost;  
12.import org.apache.http.entity.StringEntity;  
13.import org.apache.http.impl.client.DefaultHttpClient;  
14.import org.json.JSONObject;  
15.  
16.import android.os.AsyncTask;  
17.import android.os.Bundle;  
18.import android.app.Activity;  
19.import android.util.Log;  
20.import android.view.Menu;  
21.import android.view.View;  
22.import android.widget.EditText;  
23.import android.widget.TextView;  
24.import android.widget.Toast;  
25.  
26.public class MainActivity extends Activity {  
27.  
28.    //alamat local host "http://10.0.2.2/....";  
29.    private String url = "http://10.0.2.2/latihan/ws1/testjsonserver.php";  
30.    EditText kdbrgTxt;  
31.  
32.    @Override
```

```
33.     protected void onCreate(Bundle savedInstanceState) {  
34.         super.onCreate(savedInstanceState);  
35.         setContentView(R.layout.activity_main);  
36.         kdbrgTxt= (EditText)this.findViewById(R.id.kdbrgTxt);  
37.  
38.    }  
39.  
40.    @Override  
41.    public boolean onCreateOptionsMenu(Menu menu) {  
42.        // Inflate the menu; this adds items to the action bar if it is present.  
43.        getMenuInflater().inflate(R.menu.main, menu);  
44.        return true;  
45.    }  
46.  
47.    // Async Task to access the web, proses terpisah dari thread UI  
48.    private class JsonReadTask extends AsyncTask<String, Void, String> {  
49.  
50.        @Override  
51.        protected String doInBackground(String... params) {  
52.            HttpClient httpclient = new DefaultHttpClient();  
53.            String message;  
54.            String jsonResult="";  
55.  
56.            HttpPost httppost = new HttpPost(params[0]); //url  
57.            String kdbrg=params[1];//kode barang  
58.            String kdID=params[2];//ID pemohon  
59.  
60.            JSONObject obj = new JSONObject();  
61.            try {  
62.  
63.                JSONObject reqObj = new JSONObject();
```

```
64.             reqObj.put( "kdBarang", kdbrg );
65.             reqObj.put( "ID", kdID );
66.             obj.put("data", reqObj);
67.             //Log.d("ISIJSON", obj.toString(4));
68.
69.         } catch (Exception ex) {
70.
71.     }
72.
73.
74.     try {
75.         message=obj.toString();
76.
77.         httppost.setEntity(new StringEntity(message,
78.             "UTF8"));
79.         httppost.setHeader("Content-type", "application/json");
80.         HttpResponse resp = httpclient.execute(httppost);
81.
82.         Log.d("Status line", "" +
83.             resp.getStatusLine().getStatusCode());
84.     } catch (Exception e) {
85.         e.printStackTrace();
86.     }
87.
88.     try {
89.         HttpResponse response =
90.             httpclient.execute(httppost);
91.         jsonResult = inputStreamToString(
92.             response.getEntity().getContent()).toString();
93.     }
```

```
91.                                //Log.i("selese",jsonResult);
92.                            }
93.
94.                            catch (ClientProtocolException e) {
95.                                e.printStackTrace();
96.                            } catch (IOException e) {
97.                                e.printStackTrace();
98.                            }
99.                            return jsonResult;
100.                        }
101.
102.                    private StringBuilder inputStreamToString(InputStream is) {
103.                        String rLine = "";
104.                        StringBuilder answer = new StringBuilder();
105.                        BufferedReader rd = new BufferedReader(new
InputStreamReader(is));
106.
107.                        try {
108.                            while ((rLine = rd.readLine()) != null) {
109.                                answer.append(rLine);
110.                            }
111.                        }
112.
113.                        catch (IOException e) {
114.                            // e.printStackTrace();
115.                            Toast.makeText(getApplicationContext(),
116.                                    "Error..." + e.toString(),
117.                                    Toast.LENGTH_LONG).show();
118.                        }
119.                    }
```

```
120.  
121.        @Override  
122.        protected void onPostExecute(String result) {  
123.            TampilJmlBarang(result);  
124.        }  
125.    }// end async task  
126.  
127.    public void accessWebService(String prm1) {  
128.        JsonReadTask task = new JsonReadTask();  
129.        // passes values for the urls string array  
130.        task.execute(new String[] { url,prm1,"IWAN" });  
131.    }  
132.  
133.  
134.    public void TampilJmlBarang(String prm1) {  
135.  
136.        try {  
137.            //jsonResult  
138.            Toast.makeText(getApplicationContext(), "Jumlah=" +  
139.                prm1,  
140.                Toast.LENGTH_SHORT).show();  
141.        } catch (Exception e) {  
142.            Toast.makeText(getApplicationContext(), "Error" +  
143.                e.toString(),  
144.                Toast.LENGTH_SHORT).show();  
145.            Log.e("SALAH",""+e.getMessage());  
146.        }  
147.    }  
148.
```

```

149.     public void CekJmlBrg(View view){
150.             String prmlInput;
151.             prmlInput=kdbrgTxt.getText().toString();
152.
153.             //Log.i("dd",prmlInput);
154.             accessWebService(prmlInput.toUpperCase());
155.         }
156.     }

```

13. Perhatikan URL:

<http://10.0.2.2/latihan/ws1/testjsonserver.php>

alamat ip tersebut mengakibatkan Android dapat mengakses localhost (php, apache). Perlu diketahui untuk memisahkan urusan display (tampak oleh mata) dan internet maka thread harus berbeda (contoh di atas: AsyncTask)

14. Tambahkan hak akses internet pada AndroidManifest.xml

Sehingga isi AndroidManifest.xml:

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.kxLand.testjsonklien"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="17" />
    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.kxLand.testjsonklien.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>

```

```
</application>  
</manifest>
```

15. Jalankan emulator

16. Pada package explorer, TestJsonKlien, klik kanan > Run AS > Android application



17. Cobalah isi kode barang dengan: MG1111, MG1122, MG007

Kaitan dengan logistik

Dengan pertumbuhan B2B yang semakin banyak dan cepat dibutuhkan komunikasi dinamis diantaranya (dapat berupa lintas platform dlsb), web service adalah sarana yang tepat untuk menjawabnya. Pelaku bisnis mungkin tidak dapat lepas dari mobile phone oleh karena itu perlu ditentukan format data yang efisien untuk teknologi mobile. Dilihat dari segi ukuran, JSON relatif lebih kecil dibandingkan XML sehingga tepat bila digunakan pada teknologi mobile.

TUGAS:

Buat aplikasi Android dengan menggunakan web service, data format JSON yang diminta klien ditarik dari database server minimal 2 kolom, 5 baris data.

Pustaka

1. IBM, An overview of the Web Services Inspection Language
2. Introduction to Web Services, Ioannis G. Baltopoulos
3. <http://www.ibm.com/developerworks/library/os-xmldomphp/>
4. <http://www.joemarini.com/tutorials/tutorialpages/listoflinks.php>
5. Web Service Security 2010 Davide Cerri & Srdjan Komazec
6. json.org/example.html
7. <http://developer.android.com/reference/org/json/JSONObject.html>