Aturan Praktikum:

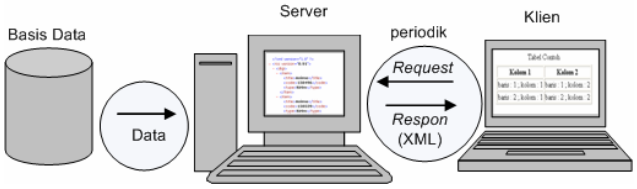
- Keterlambatan maksimal 10 menit
- Hasil praktikum, selesai atau tidak selesai ditanggung mahasiswa
- Nilai praktikum adalah nilai perorangan, dan digunakan sebagai nilai tambahan jika nilai quiz atau tes kurang mencukupi
- Mahasiswa login ke komputer
- Persiapan dan penjelasan praktikum 10 menit
- Praktikum dilakukan selama 70 menit terdiri dari praktikum sesuai model dan sebuah soal praktikum
- Hasil praktikum di-zip dan dikumpulkan
- Sisa waktu 20 menit digunakan untuk tes kecil hasil praktikum, saat tes tampilan komputer adalah background desktop dan mahasiswa tidak diperkenankan menggunakan komputer dan melihat modul praktikum
- Mahasiswa logout dari komputer
- Praktikum selesai

Asyncronous Javascript and XML (AJAX) dengan Servlet
(Minggu 7 – Praktikum II)

1. Sekilas AJAX

AJAX atau *Asyncronous JavaScript and XML* merupakan kumpulan teknologi dimana klien berinteraksi dengan server dengan menggunakan JavaScript dengan proses asinkron (*background*) secara periodik, dimana pengiriman data menggunakan XML. AJAX biasanya digunakan untuk terus menampilkan *update* terbaru dari halaman web. Menggunakan AJAX berarti proses interaksi dengan server dilakukan terus-menerus selama halaman web dibuka secara periodik, hal ini menyebabkan memakan *bandwidth* yang cukup besar.

Mekanisme kerja AJAX seperti pada gambar berikut:



- Klien mengirimkan *request* ke server
- Server melakukan proses pengambilan data yang diperlukan guna memenuhi permintaan klien
- Server mengirimkan data ke klien dengan menggunakan XML
- Klien menerima data dan memperbarui tampilan halaman web sesuai dengan data yang dikirimkan oleh server

Mekanisme di atas akan berlangsung secara periodik dan terus menerus selama halaman web masih dibuka.

Pada sisi server biasanya digunakan aplikasi web yang bertanggung jawab untuk membuat XML yang akan dikirimkan ke klien. Sisi Server dapat menggunakan PHP, ASP, JSP, maupun servlet.

2. Praktikum

a. Persiapan

- Membuat direktori kerja dengan nama SI319-P7-2-Kelas-NIM misalnya SI319-P7-2-A-23507024
- Di dalam direktori di atas, buat direktori berikut:
  - Direktori common dan isinya seperti yang digunakan pada praktikum servlet beginner
  - Direktori commonweb dan isinya seperti yang digunakan pada praktikum servlet beginner
  - Direktori XMLServlet untuk membuat servlet XML di dalamnya ada direktori src dan web seperti struktur direktori servlet

b. MySQL Connector

1. Cari file mysql-connector-java-5.1.2-beta-bin.jar pada komputer yang Anda pakai
2. Masukkan mysql-connector-java-5.1.2-beta-bin.jar pada direktori instal_jdk/jre/lib/ext dan pada jre/lib/ext
   (cara ini digunakan untuk menambah *library* atau pustaka pada Java)

c. Membuat Basis Data Pada MySQL

1. Buka command prompt, masuk ke direktori bin pada MySQL (jangan dari PHPThriad), ketik perintah:

   ```
   mysql –uroot -pithb
   ```

   jika perintah tidak dikenali maka start MySQL dengan mengetik perintah:

   ```
   mysqld
   ```

2. pada command prompt (di direktori bin MySQL), kemudian ketik perintah sebelumnya sampai muncul tampilan berikut:

```
C:\xampp\mysql\bin>mysql -uroot
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.0.37 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

3. Setiap query SQL yang berhasil dieksekusi , maka akan keluar tulisan seperti contoh berikut:

```
mysql>     INSERT     INTO     mata_kuliah(kode,     nama,     semester)
VALUES('IF3281','Pemrograman Berorientasi Objek', 3);
Query OK, 1 row affected (0.00 sec)
```

Yang penting adalah hasil: `Query OK, 1 row affected`

Ketik query SQL berikut pada command prompt di atas.

```
CREATE DATABASE IF NOT EXISTS Customer;
```

```
USE Customer;
```

```
CREATE TABLE  Customer(
id VARCHAR(10) NOT NULL,
username VARCHAR(255) NOT NULL,
password VARCHAR(10)  NOT NULL,
PRIMARY KEY (id)
);
```

```
INSERT        INTO        Customer(id,        username,        password)
VALUES('001','Nana','nana');
```

```
INSERT INTO Customer(id, username, password) VALUES('002','Ina','ina');
```

```
INSERT        INTO        Customer(id,        username,        password)
VALUES('003','Tina','tina');
```

d. Membuat XMLServlet

1. Masukkan file-file berikut pada direktori XMLServlet/src/

Nama file : DB.java

```
/**
 * Kelas DB
 * Melakukan koneksi pada basis data MySQL
 */

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;
import java.util.Date;
import java.util.UUID;

public class DB {
    private Statement stmt = null; // koneksi query
    private ResultSet rs = null; // hasil query
```

```
    private Connection conn = null; // koneksi MySQL dan basis data

    public DB(String ConAddress) throws Exception, SQLException {
    /**
     * Method DB
     * Konstruktor : melakukan koneksi ke MySQL dan basis data
     * Menerima masukan berupa string alamat koneksi ke MySQL dan basis
data
     */
      try {
            // membuat driver MySQL
            Class.forName("com.mysql.jdbc.Driver").newInstance();
            // membuat koneksi MySQL dan basis data
            conn = DriverManager.getConnection(ConAddress);

      conn.setTransactionIsolation(conn.TRANSACTION_READ_UNCOMMITTED);
      } catch(SQLException es) {
            // mengeluarkan pesan error jika koneksi gagal
            System.out.println("Connection DB Error :"+ es);
            throw es;
      }
    }

    public void createQuery(String Query)throws Exception, SQLException
{
    /**
     * Method createQuery
     * Mengeksekusi query
     * Menerima masukan berupa string query
     */
      try {
            stmt = conn.createStatement();
            // eksekusi query
            rs = stmt.executeQuery(Query);
            if (stmt.execute(Query)) {
                  // ambil hasil query
                    rs = stmt.getResultSet();
            }
         } catch(SQLException es) {
            // eksepsi jika query gagal dieksekusi
            System.out.println("createQuery Error :"+ es);
            throw es;
         }
    }

    public    void    createUpdate(String    Query)throws    Exception,
SQLException {
    /**
     * Method createQuery
     * Mengeksekusi query
     * Menerima masukan berupa string query
     */
      try {
            stmt = conn.createStatement();
            // eksekusi query
            int hasil = stmt.executeUpdate(Query);
         } catch(SQLException es) {
```

```
            // eksepsi jika query gagal dieksekusi
            System.out.println("createUpdate Error :"+ es);
            throw es;
        }
    }

    public ResultSet getResult()throws Exception {
    /**
     * Method getResult
     * Memberikan hasil query
     */
      ResultSet Temp = null;
      try{
                return rs;
          }catch (Exception ex) {
                  // eksepsi jika hasil tidak dapat dikembalikan
                  System.out.println("getResult Error :"+ex);
                  return Temp;
          }
    }

    public void closeResult()throws SQLException, Exception {
    /**
     * Method closeResult
     * Menutup hubungan dari eksekusi query
     */
      if (rs != null) {
              try {
                  rs.close();
              } catch (SQLException sqlEx) {
                  rs = null;
                  throw sqlEx;
              }
      }
      if (stmt != null) {
              try {
                  stmt.close();
              } catch (SQLException sqlEx)
              {
                  stmt = null;
                  throw sqlEx;
              }
      }
    }

    public void closeConnection()throws SQLException, Exception {
    /**
     * Method closeConnection
     * Menutup hubungan dengan MySQL dan basis data
     */
              if (conn != null) {
                  try {
                        conn.close();
                  } catch(SQLException sqlEx)
                  {
                        conn = null;
                  }
```

```
            }
    }
}
```

Nama File: XMLServlet.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import javax.naming.*;
import java.io.*;
import java.rmi.RemoteException;
import java.util.logging.*;
import javax.rmi.PortableRemoteObject;


public class XMLServlet extends HttpServlet {

    public void handleRequest(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {

        response.setContentType("text/xml");

      try{
             DB db =
new DB("jdbc:mysql://localhost:3306/Customer?user=root&password=ithb");
             db.createQuery("SELECT * FROM CUSTOMER");

             StringBuffer xml = new StringBuffer();
             xml.append("<customers>");

             xml.append("<fields>");
             xml.append("<field>id</field>");
             xml.append("<field>username</field>");
             xml.append("<field>password</field>");
             xml.append("</fields>");

            while(db.getResult().next()){
                   // ambil hasil query
                  String id = db.getResult().getString(1);
                  String username = db.getResult().getString(2);
                  String password = db.getResult().getString(3);

                  xml.append("<customer>");
                  xml.append("<id>" + id + "</id>");
                  xml.append("<username>" + username + "</username>");
                  xml.append("<password>" + password + "</password>");
                  xml.append("</customer>");
         }

         db.closeResult();
         db.closeConnection();
```

```
            xml.append("</customers>");
            response.getWriter().println(xml.toString());

            }catch(Exception e){
        }
    }

    public void doGet(HttpServletRequest request,
    HttpServletResponse response)
            throws ServletException, IOException {
            this.handleRequest(request, response);
      }

      public void doPost(HttpServletRequest request,
      HttpServletResponse response)
            throws ServletException, IOException {
            this.handleRequest(request, response);
      }
}
```

2. Masukkan file-file yang biasa dipakai untuk membuat servlet pada direktori XMLServlet seperti:
    - build.xml
    - web.xml
    - sun-web.xml

    ganti isi web.xml menjadi seperti berikut:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app      xmlns="http://java.sun.com/xml/ns/j2ee"      version="2.4"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
  <display-name>XMLServlet</display-name>
</web-app>
```

Dan isi sun-web.xml seperti berikut:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE    sun-web-app    PUBLIC    "-//Sun    Microsystems,    Inc.//DTD
Application         Server         8.0         Servlet         2.4//EN"
"http://www.sun.com/software/appserver/dtds/sun-web-app_2_4-0.dtd">

<sun-web-app>
  <context-root>/XMLServlet</context-root>
</sun-web-app>
```

Buat file .war sebagai berikut:

1. Buka jendela command prompt, masuk ke direktori XMLServlet

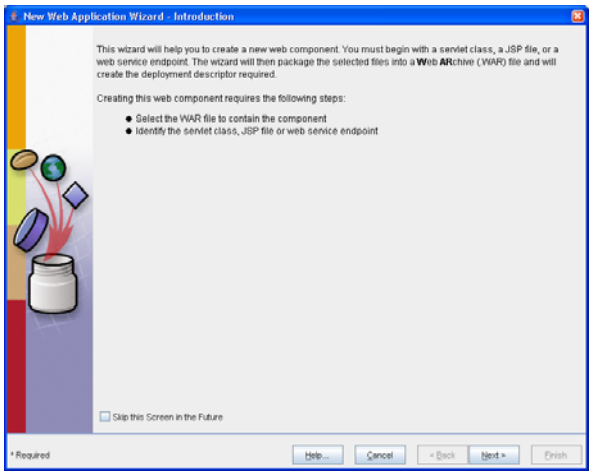
2. ketik perintah: asant build. Perintah asant build digunakan untuk mengkompilasi *source code*, jika *source code* telah lulus kompilasi maka akan muncul pesan sebagai berikut:

```
Buildfile: build.xml

init:

prepare:

copy:

build:

BUILD SUCCESSFUL
Total time: 2 seconds
```

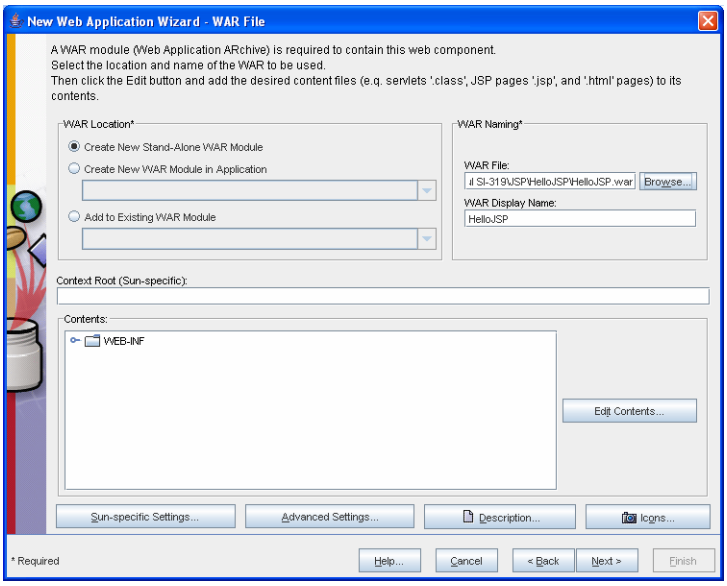
Jika perintah asant build tidak dikenali, ketik perintah sebagai berikut:

path=C:/Sun/AppServer/bin;%path%

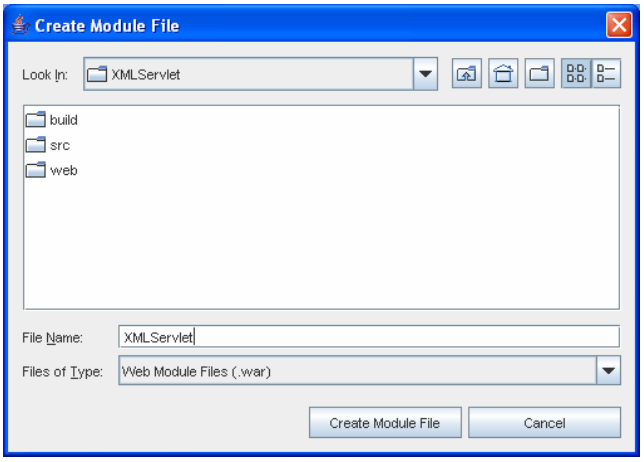
3. Buka Deploytool dari Sun untuk melakukan *packaging* menjadi file WAR. Klik Start->All Program->Sun Microsystems->Application Server PE->Deploytool.
4. Klik File->New->Web Component... . Hingga muncul jendela seperti pada gambar berikut.



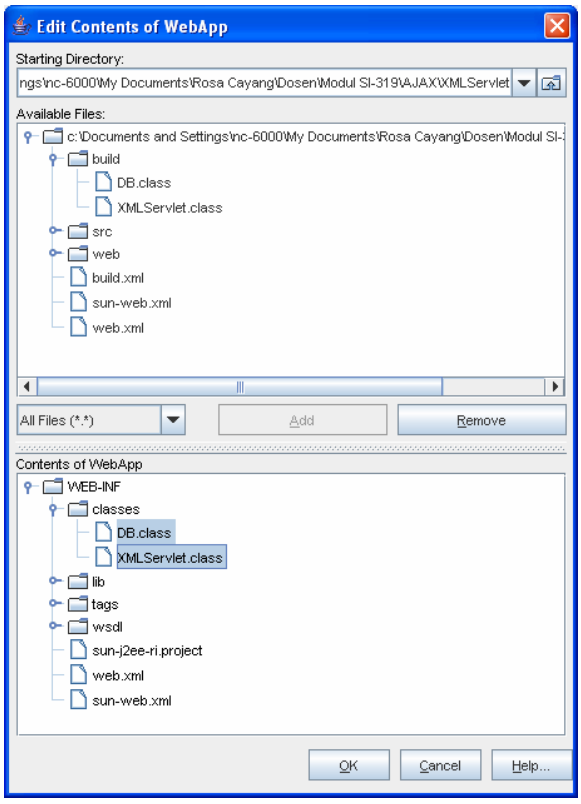
5. Klik Next. Hingga muncul jendela seperti pada gambar berikut.



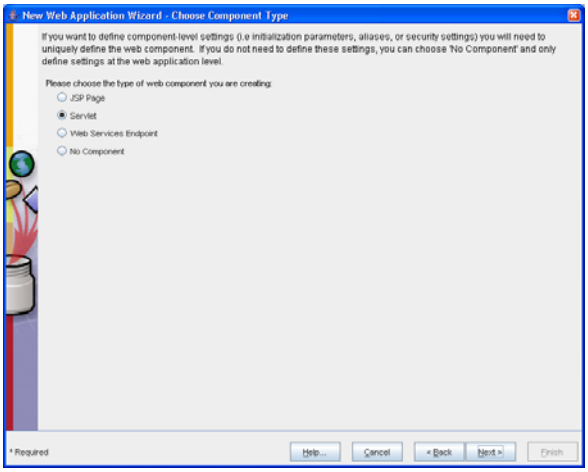
Pilih Create New Stand-Alone WAR Module, isi *field* WAR file dengan mengeklik browse untuk mencari direktori XMLServlet, dan isi file name dengan XMLServlet seperti pada gambar berikut dan klik create module file.



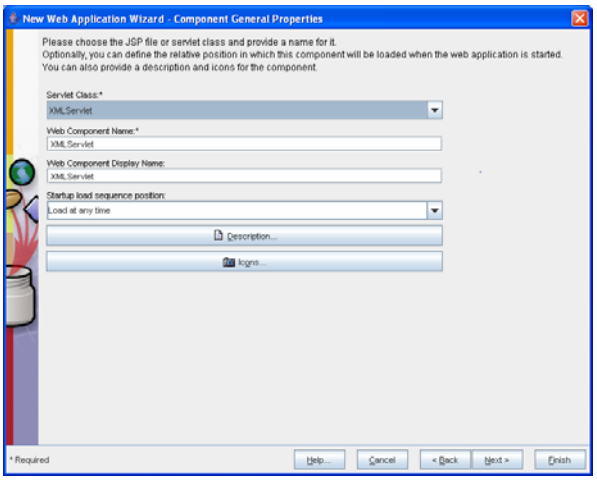
Klik edit contents (gambar sebelumnya) hingga muncul jendela seperti gambar berikut. Tambahkan direktori contoh pada available file dengan mengeklik direktori build, kemudian klik file XMLServlet.class dan DB.class dan klik tombol add kemudian klik ok. Kemudian klik next.



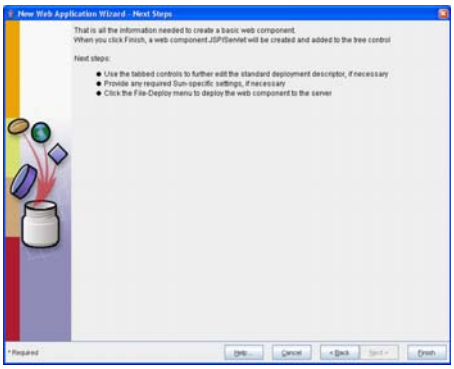
6. Pada jendela seperti gambar berikut, pilih Servlet. Klik next.



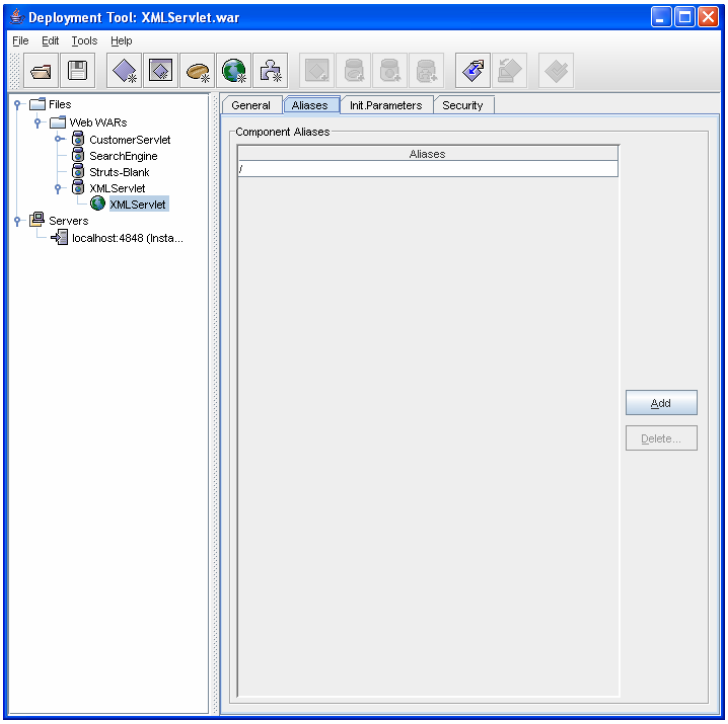
7. Kemudian akan muncul jendela seperti gambar berikut. Isi field yang ada dengan mengeklik tanda panah dan pilih pilihan yang ada, maka beberapa *field* akan terisi. Klik next.



8. Setelah muncul jendela seperti pada gambar berikut, klik Finish.



9. Setelah kembali ke jendela awal, pilih tab Aliases, klik Add dan isi *field* Aliases dengan / lalu tekan enter.



10. Setelah kembali ke jendela awal, klik tombol save yang bergambar disket, maka file WAR akan terbentuk di dalam direktori XMLServlet.

11. Buat file customer.html sebagai berikut:

```
<html>
  <script type="text/javascript" language="JavaScript">

            var http_request = false;
                  var customerRecordScript =
"http://localhost:3030/XMLServlet";
                  var firstRecordIdx = 0;
                  var recordCountId = "VJxmrz4H";
                  var orderId = "7gwAofWW";

            function makeRequest(url) {

                http_request = false;

                if (window.XMLHttpRequest) { // Mozilla, Safari,...
                    http_request = new XMLHttpRequest();
                    if (http_request.overrideMimeType) {
                        http_request.overrideMimeType('text/xml');
                    }
                } else if (window.ActiveXObject) { // IE
                    try {
                        http_request = new
ActiveXObject("Msxml2.XMLHTTP");
                    } catch (e) {
                        try {
                        http_request = new
```

```
ActiveXObject("Microsoft.XMLHTTP");
                        } catch (e) {}
                    }
                }

                if (!http_request) {
                    alert('Cannot create an XMLHTTP instance');
                    return false;
                }
                http_request.onreadystatechange = alertContents;
                http_request.open('GET', url, true);
                http_request.send(null);
            }

     function updateTableView(http_request) {
  // mengambil elemen HTML dengan id centerTD
  var leftTD = document.getElementById("centerTD");

  // mengambil elemen tabel dengan id customerTable
  var table1 =
    document.getElementById("customerTable");

  // mengambil xml respon dari server
  var xmldoc = http_request.responseXML;

  // parsing xml, mengambil data dalam tag customer
  var customer =
    xmldoc.getElementsByTagName('customers').item(0);

  // jika elemen customer tidak ada dalam xml
  if (!customer) {
     alert('simpul akar tidak terdefinisi');
  }

  // jika elemen tabel tidak ada pada dokumen HTML
  if (!table1) {
     // Buat elemen tabel
     table1 = document.createElement("table");
     table1.id = "customerTable";
     table1.border = "1";

     // mengaitkan tabel
     centerTD.appendChild(table1);

     /* mengambil data tag xml fields untuk header tabel */
     var fields =
       customer.getElementsByTagName('fields').item(0);
     if (!fields) {
      alert('fields gagal didapatkan');
     }

     // mengambil data tag field
     var fieldList =
       fields.getElementsByTagName('field');

     // menghitung banyak field
     var fieldCount = fieldList.length;
```

```
    // jika data field bukan kosong
    if (fieldCount > 0) {
       // membuat elemen thead pada dokumen HTML
       var thead1 = document.createElement("thead");

        // mengaitkan thead
      table1.appendChild(thead1);

        // membuat elemen tr pada dokumen HTML
      var tr1 = document.createElement("tr");

       // mengaitkan tr pada thead
      thead1.appendChild(tr1);

       // membuat elemen td
      var i;
      for (i = 0; i < fieldCount; ++i) {
        var field = fieldList.item(i);
        var td1 = document.createElement("td");
        tr1.appendChild(td1);
        var fieldText1 =
       document.createTextNode(field.firstChild.data);
        td1.appendChild(fieldText1);
                                                }
    }
      }
    var customerList =
      customer.getElementsByTagName('customer');
    var customerCount = customerList.length;
    if (customerCount > 0) {
     var i;
     var tbody1 = document.createElement("tbody");
     tbody1.id = "customerTableBody";
     for (i = 0; i < customerCount; ++i) {
        var tr1 = document.createElement("tr");
        tbody1.appendChild(tr1);
        var customer = customerList.item(i);
        var td1 = document.createElement("td");
        tr1.appendChild(td1);
        var customerId =
          customer.getElementsByTagName('id').item(0);
        fieldText1 =
  document.createTextNode(customerId.firstChild.data);
        td1.appendChild(fieldText1);
        var customerName =
        customer.getElementsByTagName('username').item(0);
        fieldText1 =
document.createTextNode(customerName.firstChild.data);
        td1 = document.createElement("td");
        td1.appendChild(fieldText1);
        tr1.appendChild(td1);
         var customerPassword =
    customer.getElementsByTagName('password').item(0);
        fieldText1 =
           document.createTextNode(
             customerPassword.firstChild.data);
```

```
          td1 = document.createElement("td");
          td1.appendChild(fieldText1);
          tr1.appendChild(td1);
        }

        var oldTbody1 =
           document.getElementById("customerTableBody");
        if (!oldTbody1) {
          table1.appendChild(tbody1);
        } else {
          table1.replaceChild(tbody1, oldTbody1);
         }
    }
}


      function alertContents() {
            if (http_request.readyState == 4) {
                  if (http_request.status == 200) {
                        updateTableView(http_request);
                        self.setTimeout("loadBody()", 5000);
                  } else {
                        alert('There was a problem with the request.');
                  }
            }
      }


      function loadBody() {
            makeRequest(customerRecordScript + "?fridx=" +
firstRecordIdx + "&rcid=" + recordCountId + "&orid=" + orderId);
      }
    //-->
  </script>
  <title>AJAX with Servlet</title>
  <body onload="loadBody()">
<h3 align='center'>Nama/NIM : <!-- isi dengan nama / NIM--></h3>

<p align='center'>Tabel Customer</p>

<table align='center'>
  <tbody>
      <tr>
      <td valign="top" id="centerTD"></td>
      </tr>
  </tbody>
</table>

</body>
</html>
```

12. Simpan file customer.html pada C:\Program Files\Apache Software Foundation\Tomcat 5.0\webapps\ROOT

13. Copy kan file XMLServlet.war pada direktori C:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps

14. Start Tomcat: All Program->Apache Tomcat->Monitor Tomcat
15. Begitu jendela monitor muncul, klik tombol start.
    Jika jendela tidak muncul, klik dua kali icon seperti gambar berikut pada menu sisi kanan bawah desktop hingga muncul jendela monitor tomcat.

16. Buka web browser, ketik alamat:

```
http://localhost:8080/
```

Jika tidak muncul halaman web apache, maka harus start manual menjalankan program C:\Program Files\Apache Software Foundation\Tomcat 6.0\bin\ tomcat6.exe

Hingga muncul command prompt, tunggu hingga muncul tulisan “server startup in”, jangan menutup jendela command prompt ini.

17. Tes XMLServlet dengan mengetik alamat :
    `http://localhost:8080/XMLServlet/`
pada web browser.

Jika error, perbaiki error.

18. Tes AJAX dengen mengetik alamat
    `http://localhost:8080/customer.html`
pada web browser.

19. Tambahkan query berikut pada MySQL:

```
INSERT INTO Customer(id, username, password)
VALUES('004','Edo','edo');
```

Amati apa yang terjadi pada web browser