JAVA SERVLET (1)

I. Tujuan

1. Mahasiswa mampu menjalankan Servlet pada web server Tomcat
2. Mahasiswa mengenal Servlet dan dasar-dasar Servlet
3. Mahasiswa mengenal istilah-istilah Servlet dan pemakaiannya
4. Mahasiswa mampu membuat fungsi memakai Servlet

II. Software yang Dibutuhkan

1. Editor untuk menulis HTML
2. Browser yang dipakai untuk menjalankan HTML
3. Web Application Server Tomcat

III. Dasar Teori

Jakarta Tomcat

Jakarta Tomcat adalah web application server, yang mempunyai kemampuan sebagai Servlet container dan JSP container di mana Anda bisa mendeploy Servlet dan JSP. Di atas Jakarta Tomcat, Servlet dan JSP akan bekerja melayani request dari client, yang lumrahnya adalah berupa browser.

Untuk bisa menjalankan Jakarta Tomcat, Anda membutuhkan Java Development Kit (JDK). Untuk instalasi Jakarta Tomcat, Anda bisa mendownload binary dari http://jakarta.apache.org , dalam format .zip, .tar.gz. Yang Anda perlu lakukan hanyalah mendecompress file tersebut.

Dalam bekerja dengan Jakarta Tomcat, Anda mempunyai sebuah directory yang dikenal sebagai TOMCAT_HOME. TOMCAT_HOME adalah directory di mana Jakarta Tomcat diinstall.

Selanjutnya di bawah TOMCAT_HOME Anda akan menemukan beberapa subdirectory, diantaranya bin/, conf/, logs/ dan webapp/. Di dalam subdirectory bin/ terdapat file-file executable terutama untuk menjalankan dan menghentikan Jakarta Tomcat. Di dalam subdirectory conf/ terdapat file-file untuk configuration. Di dalam subdirectory logs/ terdapat file-file log. Dan subdirectory webapp/ adalah di mana Anda bisa meletakkan aplikasi Web yang Anda bangun dengan Servlet dan JSP.

Di bawah subdirectory webapp/ Anda bisa mengcreate subdirectory. Sub directory ini akan dijadikan sebagai Context oleh Jakarta Tomcat.

Anda menjalankan Jakarta Tomcat dengan mengexecute startup.sh di subdirectory bin/. Sedangkan untuk menghentikan Tomcat Anda mengexecute shutdown.sh di sub directory bin/ juga.

Secara default Jakarta Tomcat siap melayani request dari client melalui port 8080. Melalui Web browser, Anda bisa menghubungi http://localhost:8080

Aplikasi WEB

Teknologi inti untuk mengembangkan aplikasi Web dengan Java adalah Servlet. Servlet adalah sebuah class yang digunakan untuk menerima request dan memberikan response, terutama melalui protokol HTTP. Anda menulis source code dari Servlet, lalu mengcompile dan mendeploy di java web server. Selanjutnya client dapat berinteraksi dengan Servlet melalui browser.

Servlet bisa dipandang sebagai class yang bisa digunakan untuk menulis response dalam format HTML. Ia ditulis sebagaimana lumrahnya sebuah class di dalam bahasa pemrograman Java. Servlet disimpan sebagai file .java. Untuk

mengirimkan response dalam format HTML, Anda bisa menulisnya melalui obyek PrintWriter, yang didapatkan dari HttpServletResponse.

Dalam perjalananya, dikembangkan teknologi Java Server Page (JSP) di mana Anda bisa menulis script untuk aplikasi Web dengan bahasa Java.

Berbeda dengan Servlet, JSP bisa dipandang sebagai HTML yang di dalamnya bisa mempunyai kode-kode Java. JSP disimpan sebagai file .jsp. Menulis JSP adalah seperti menulis file HTML, kecuali di dalamnya dapat disisipkan kode-kode Java sebagai presentation logic. Kode-kode Java ini disisipkan melalui directive, sebagai scriplet, atau sebagai expression.

Servlet dan JSP mempunyai kemampuan yang kembar. Keduanya bisa membaca input yang dikirimkan melalui form di Web, mengakses database melalui JDBC, mengolah data dan menulis response ke browser. Response lumrahnya dalam format HTML.

Perbedaan Servlet dan JSP lebih kepada proses pengembangannya. Sedangkan dalam operasinya, keduanya adalah sama. Oleh web application server, JSP akan direwrite menjadi Servlet, dicompile dan selanjutnya akan diperlakukan sebagaimana Servlet.

Cara Kerja Servlet

Servlet bekerja melayani request dari client, yang lumrahnya adalah Web browser. Untuk bisa melayani client, Servlet terlebih dahulu harus dideploy di web application server, yang menyediakan kemampuan sebagai Servlet container.

Client memanggil Servlet dengan mengirimkan HTTP request ke web application server. HTTP request ini bisa di transfer dengan method GET, POST atau lainnya. Method GET selalu terjadi jika user membuka sebuah URL. Method POST bisa digunakan saat user mensubmit sebuah form.

Saat web application server menerima HTTP request dari client, ia akan menyerahkan request ini ke Servlet container. Servlet container akan mengcreate dua buah obyek yaitu obyek HttpServletRequest dan obyek HttpServletResponse. Obyek HttpServletRequest mengencapsulate HTTP request dari client, sedangkan obyek HttpServletResponse dipersiapkan untuk mengencapsulate HTTP response ke client.

Selanjutnya Servlet container akan menginvoke method dari Servlet dengan melewatkan dua obyek ini. Servlet yang diinvoke oleh Servlet container ditentukan oleh URL yang dikirimkan oleh Web browser, dan pemetaan yang dibuat melalui configuration. Dalam configuration dapat ditentukan bahwa URL dengan pola tertentu akan dilayani oleh Servlet tertentu.

Servlet bisa membaca data yang dikirimkan oleh client melalui obyek HttpServletRequest. Melalui obyek ini Servlet membaca parameter, cookies, dan juga informasi tentang client.

Selanjutnya untuk mengembalikan response ke client, Servlet bisa melakukannya melalui obyek HttpServletResponse. Lumrahnya Servlet menuliskan response dalam format HTML.

Sebelum menuliskan response, Servlet terlebih dahulu bisa mengolah data yang dikirimkan oleh client, mengakses dengan database dan melakukan proses-proses lain.

Context

Sebuah Context adalah sebuah aplikasi Web yang terpisah, berdiri sendiri, independen. Sebuah Context mempunyai configuration masing-masing. Library dari

sebuah Context juga tidak bisa dibaca oleh Context lain. Obyek di sebuah Context tidak bisa mengakses obyek di Context lain.

Di atas sebuah web application server seperti Jakarta Tomcat bisa dideploy lebih dari satu Context.

Anda bisa membuat sebuah Context dengan mengcreate sebuah subdirectory di bawah TOMCAT_HOME/webapp/.

Sebuah Context yang lengkap mempunyai subdirectory WEB-INF/ di mana terdapat web.xml yang merupakan configuration file dari Context ini. Di dalam WEB-INF/ bisa terdapat subdirectory classes/ dan lib/.

Subdirectory classes/ adalah di mana file-file .class diletakkan, sedangkan lib/ adalah di mana file-file .jar, yang merupakan kumpulan file-file .class, diletakkan.

Java Servlet

Java Servlet ditulis sebagai lumrahnya Java class lain. Ia disimpan dalam file berekstension .java. Sebuah Java Servlet harus merupakan subclass dari HttpServlet

Untuk melayani request dari client, Anda perlu mengoverride method service(). Parameter yang dilewatkan ke dalam method service() ini berupa obyek HttpServletRequest dan obyek HttpServletResponse.

Untuk lebih spesifik terhadap HTTP method, Anda juga bisa mengoverride method doGet() dan atau doPost(). Kedua method ini mempunyai parameter yang sama dengan service().

Method doGet() akan dijalankan jika client mengirimkan HTTP request dengan method GET. Contoh dari method GET, adalah jika user meng-click sebuah link di halaman Web. Dalam kasus ini, Web browser akan mengirimkan HTTP request dengan method GET ke server.

Method doPost() akan dijalankan jika client mengirimkan HTTP response dengan method POST. Ini terjadi misalnya, saat user mengisi HTML form dengan method POST, dan mensubmit request tersebut ke server.

Di dalam method service(), doGet(), atau doPost() ini Anda bisa membaca parameter yang dikirimkan client, mengolah data, mengakses database dan menulis response ke client.

Deployment Servlet di Jakarta Tomcat

Untuk melakukan deployment, Anda perlu mempunyai Context. Ini bisa dibuat dengan membuat subdirectory di bawah TOMCAT_HOME/webapp/

Untuk mendeploy Servlet, pertama Anda meng-compile Servlet. Lalu Anda meletakkan file-file .class ke sub-directory WEB-INF/classes di bawah directory yang dibuat untuk Context Anda.

Jika dibutuhkan Anda bisa meletakkan file-file .jar di sub-directory WEB-INF/lib.

Selanjutnya Anda bisa memanggil dari browser, sesuai Context dan nama Servlet Anda.

Anda bisa juga meng-edit web.xml, di subdirectory WEB-INF/ di bawah directory dari Context Anda. Melalui web.xml Anda, di antaranya, bisa membuat mapping yang mengatur bahwa pola URL tertentu akan dilayani oleh Servlet tertentu.

Initialization

Anda bisa melakukan initialization terhadap Servlet sebelum Servlet melayani client.

1. init()

2. init(ServletConfig)
3. ServletConfig
4. getInitParameter()
5. getInitParameterNames()

HttpServletRequest

Class HttpServletRequest digunakan untuk meng-encapsulate HTTP request dari client.

1. Untuk membaca parameter-parameter yang dikirimkan client, tersedia method-method getParameter(), getParameterNames() dan getParameterValues().
2. Untuk mendapatkan header dari HTTP request tersedia method getHeader(), dan membaca cookie tersedia method getCookie().
3. Untuk mendapatkan informasi tentang server di mana Servlet bekerja, tersedia method getServerName() dan getServerPort().
4. Untuk mendapatkan informasi tentang client yang memanggil Servlet, tersedia method getRemoteAddr(), getRemoteHost() dan getRemoteUser().

HttpServletResponse

Class HttpServletResponse digunakan untuk meng-encapsulate HTTP response yang dikirimkan Servlet ke client.

Method getWriter() bisa digunakan untuk mendapatkan obyek PrintWriter. Method getOutputStream() bisa digunakan untuk mendapatkan obyek ServletOutputStream.

Melalui obyek PrintWriter atau ServletOutputStream, Anda bisa menuliskan response ke client. Obyek PrintWriter cocok jika response Anda adalah character, misal dalam format HTML, sedangkan obyek ServletOutputStream cocok jika response Anda adalah binary, misalnya berupa graphics.

Method setHeader() bisa digunakan untuk menuliskan header, dan method setCookie() bisa digunakan untuk menuliskan cookie.

Method setStatus() bisa digunakan untuk mengirimkan status code ke client.

IV. Tugas Pendahuluan

1. Buatlah program Servlet sederhana beserta file html-nya!
2. Sebutkan fungsi-fungsi yang ada dalam servlet dan jelaskan masing-masing!

V. Percobaan

Percobaan 1.

Sebuah praktikum dengan Servlet yang menggenerate satu halaman Web yang menampilkan sebuah pesan.

Persiapan

Buat sebuah directory untuk bekerja
misalnya /Tomcat4.1/webapps/process
lalu buat sub directory WEB-INF
dan sub directory WEB-INF/classes

Langkah

Langkah ke-1

Tulis SalamKeadilanServlet.java, simpan di-directory yang sudah dipersiapkan WEB-INF/classes

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;


public class SalamKeadilanServlet extends HttpServlet
{

   public void service(HttpServletRequest request, HttpServletResponse response)
   throws IOException, ServletException
   {
      response.setContentType("text/html");
      PrintWriter out = response.getWriter();

      out.println("<html><body>");
      out.println("Salam keadilan !");
      out.println("</body></html>");
   }

}
```

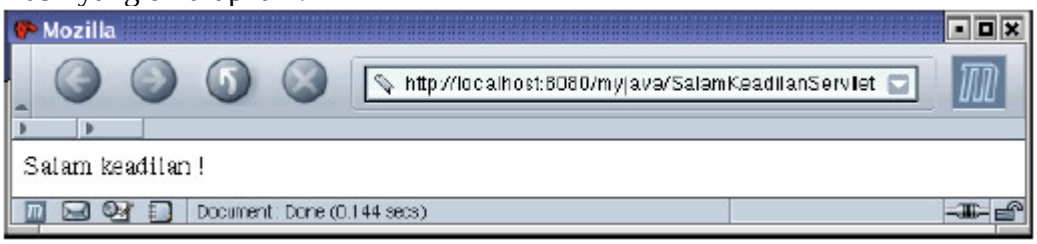
Langkah ke-2
Compile ...
`$ javac WEB-INF/classes/SalamKeadilanServlet.java`

Langkah ke-3
Buat file dengan nama web dan berektensi .xml (file web.xml), simpan ke sub directory WEB-INF
`WEB-INF/web.xml`

```
<web-app>

<servlet>
<servlet-name>SalamKeadilanServlet</servlet-name>
<servlet-class>SalamKeadilanServlet</servlet-class>
</servlet>

<servlet-mapping>
<servlet-name>SalamKeadilanServlet</servlet-name>
<url-pattern>/SalamKeadilanServlet</url-pattern>
</servlet-mapping>

</web-app>
```

Langkah ke-4
Servlet telah di-deploy di atas Tomcat web application server, siap melayani request dari browser.
melalui URL :
`http://localhost:8080/process/SalamKeadilanServlet`

Hasil yang diharapkan :

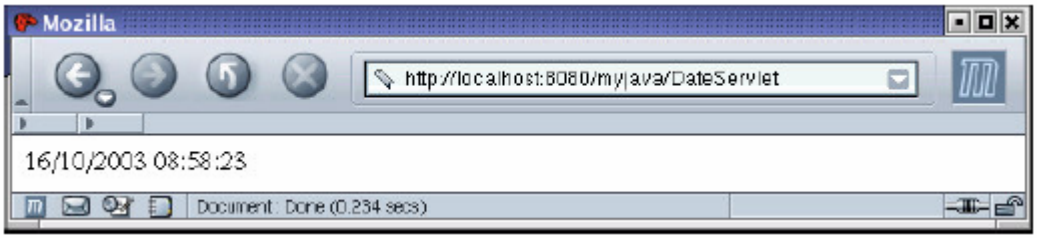


Percobaan 2.

Menampilkan tanggal dan waktu saat ini.
Lakukan langkah yang sama dengan praktikum 1.

```
import java.io.*;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.servlet.*;
import javax.servlet.http.*;

public class DateServlet extends HttpServlet {
   public void service(HttpServletRequest request, HttpServletResponse response)
   throws IOException, ServletException {
      response.setContentType("text/html");
      PrintWriter out = response.getWriter();
      Date date = new Date();
      SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy
hh:mm:ss");
      out.println("<html><body>");
      out.println(format.format(date));
      out.println("</body></html>");
   }
}
```

```
<web-app>
<servlet>
<servlet-name>SalamKeadilanServlet</servlet-name>
<servlet-class>SalamKeadilanServlet</servlet-class>
</servlet>
<servlet>
<servlet-name>DateServlet</servlet-name>
<servlet-class> DateServlet </servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>SalamKeadilanServlet</servlet-name>
<url-pattern>/SalamKeadilanServlet</url-pattern>
</servlet-mapping>
<servlet-mapping>
<servlet-name> DateServlet </servlet-name>
<url-pattern>/ DateServlet </url-pattern>
</servlet-mapping>
</web-app>
```



Percobaan 3.

Lakukan langkah yang sama dengan praktikum 1.
Simple servlet that outputs plain text (not html).

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class PlainText extends HttpServlet {

   public void doGet(HttpServletRequest request, HttpServletResponse response)
   throws IOException, ServletException
   {
      response.setContentType("text/plain");
      PrintWriter out = response.getWriter();
      out.println("<H1>Hello World!</ H1 ><BR>");
   }
}
```

Modifikasi file web.xml!

Percobaan 4.

Latihan JDBC dengan Servlet

```
import java.io.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class LatJDBCServlet extends HttpServlet
{

   public void service(HttpServletRequest request,
                       HttpServletResponse response)
      throws IOException, ServletException
   {
      response.setContentType("text/html");
      PrintWriter out = response.getWriter();

      out.println( "<html>"
         + "<head><title>Latihan JDBC dengan Servlet</title></head>"
         + "<body>"
         + "<h1>PATIENT TABLE</h1>");

      Connection conn = null;
      Statement stmt = null;
      ResultSet rs = null;
```

```
try
      {
         String jdbcDriver
            = "sun.jdbc.odbc.JdbcOdbcDriver";
         Class.forName(jdbcDriver);

         String url = "jdbc:odbc:Northwind";
         String user = "";
         String pwd  = "";

         conn = DriverManager.getConnection(
               url, user, pwd);
         stmt = conn.createStatement();

         String selectSQL =
            "SELECT EmployeeID, LastName, FirstName, Title FROM employees";
         rs = stmt.executeQuery(selectSQL);

         out.println("<table border=1>");
         out.println(
            "<tr>"
            + "<th>EmployeeID</th>"
            + "<th>Last Name</th>"
            + "<th>First Name</th>"
            + "<th>Title</th>"
            + "</tr>");

         while(rs.next())
         {
            String eID = rs.getString("EmployeeID");
            String lName = rs.getString("LastName");
            String fName = rs.getString("FirstName");
            String title = rs.getString("Title");

            out.println("<tr>");
            out.println("<td>" + eID + "</td>");
            out.println("<td>" + lName + "</td>");
            out.println("<td>" + fName + "</td>");
            out.println("<td>" + title + "</td>");
            out.println("</tr>");
         }
         out.println("</table>");
      }
      catch(ClassNotFoundException cnfe)
      {
         out.println(cnfe.toString());
      }
      catch(SQLException sqle)
      {
         out.println(sqle.toString());
      }
      out.println("</body></html>");
   }
}
```

Percobaan 5.

Lakukan langkah yang sama dengan praktikum 1.
Simple servlet that outputs ms-excel (not html).

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class Excel extends HttpServlet {
   public void doGet(HttpServletRequest request, HttpServletResponse response)
   throws IOException, ServletException
   {
      response.setContentType("application/vnd.ms-excel");
      ByteArrayOutputStream bytes = new ByteArrayOutputStream(1024);
      PrintWriter out = response.getWriter();
      out.println("1997  1998  1999  2000");
      out.println("100   300   250   350");
   }
}
```

VI. Laporan Resmi

1. Tuliskan hasil percobaan yang anda lakukan
2. Berikan kesimpulan terhadap percobaan yang anda buat.