Servlets

Servlets sind Java Klassen, die durch einen geeigneten Server geladen und instanziert werden.

Die bekanntesten Server sind

tomcat

jboss

Bea weblogic

Sevlets sind nicht an spezielle Protokolle gebunden.

Eine spezielle Ausprägung der Servlets sind HttpServlets, sie bilden auch den Schwerpunkt dieser Betrachtungen

Tomcat

Kostenloser Download unter
http://tomcat.apache.org/download-60.cgi

Steht meist als Package bei Linuxdistributionen zur Verfügung

Benutzt standardmäßig Port 8080

Kann auch in einem user-Verzeichnis installiert und gestartet werden, ggf müssen Umgebungsvariablen, wie TOMCAT_HOME u.a. gesetzt werden.

javax.servlet

Classlibrary in einem gesonderten jar-File, Doku nach Serverinstallation über Serverhomepage erreichbar

Angabe des Classpath ist erforderlich

-cp .:servlet.jar

-cp .:servlet-api.jar (tomcat 7)

Ggf. ist servlet.jar mit find zu suchen

```
find /usr -name servlet.jar
```

Servlets werden über den „deploy“ Mechanismus mit dem Server verbunden.

Zuvor sollte die Webanwendung in ein war-file gepackt werden.

Lebenszyklus

Servlets werden einmalig erzeugt und leben dann bis zur Beendigung des Servers.

Lebenszykus:

init einmalig (Öffnen und Lesen von Daten)

service (doGet, doPost, doPut, doDelete) wird mehrfach aufgerufen, bei Bedarf auch als Thread

destroy einmalig

Über Sessionmanagement können Daten innerhalb einer Sitzung über mehrere service-requests aufbewahrt werden.

Zur Laufzeit existiert von jedem Servlet in der Regel eine Instanz

Für jeden Request wird ein Thread erzeugt, der den Request behandelt.

Von diesem Thread wird die service-Methode aufgerufen, die wiederum die do... Methoden aufruft.

In der Kosequenz heißt das, dass alle Requests die Instanzvariablen eines Servlets teilen und den Zugriff darauf ggf. synchronisieren müssen.

Über das Markerinterface SingleThreadModel kann das Starten mehrerer Threads zu einem Servlet verhindert werden, verschlechtert aber u.Ust. die Preformance.

Lokale Variablen sind threadsicher.

Verzeichnisstruktur für Servlets

Zur Entwicklung eines Servlets empfiehlt sich folgende Verzeichnisstruktur:

- MyServletApplication
  - META-INF
  - WEB-INF
    - classes
      - MyServlet.class
      - *.class
    - web.xml

bleibt leer

Servletclasses ev. in Unterverzeichnissen, wenn packages verwendet wurden.

Deploymentdecriptor

Methoden von Servlet

Methode	*Bemerkung*
service	Behandelt die Requests
getServletConfig	Liefert ein Initialisierungsobjekt (→ init)
init	Initialisierung (einmalig)
destroy	Beim Beenden des Servers
getServletInfo	Liefert Info zu Servlet

Methoden von HttpServlet

Methode	*Bemerkung*
doGet,	Behandeln die entsprechenden Requests
doPost,	
doPut	
doDelete	
init	Initialisierung (einmalig)
destroy	Beim Beenden des Servers
getServletInfo	Liefert Info zu Servlet

Verbindung zum http Client

Webseite – Formular, Applet

Die doXXX-Methoden haben folgenden Aufbau:

```
void do....(HttpServletRequest req, HttpServletResponse resp)
```

Über die Parameterobjekte req und resp wird die Verbindung zum http Client realisiert.

```
HttpSession Sess=req.getSession(true);
String Passw=req.getParameter("Password");
String Query=req.getQueryString();
String User =req.getRemoteUser();
String Addr =req.getRemoteAddr();
String Host =req.getRemoteHost();
Writer out  =res.getWriter();
```

true: neue Session anlegen, wenn keine Session vorhanden
false: keine neue Session anlegen

Über diesen Writer wird der Output zum Client gesendet

Beispiel: unvermeidliches Hello

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class HelloTestServ extends HttpServlet
{
  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
  throws IOException, ServletException
  {
     response.setContentType("text/html");
     PrintWriter out = response.getWriter();
     out.println("<html>");
     out.println("<body>");
     out.println("<head>");
     out.println("<title>Hello HelloTestServ!</title>");
     out.println("</head>");
     out.println("<body>");
     out.println("<h1>Javaspass mit Servlets!</h1>");
     out.println("</body>");
     out.println("</html>");
  }
}
```

Quelltext fertig – und nun?

Verzeichnisstruktur, wenn nicht schon geschehen, einrichten

Compilieren

```
javac -cp /usr/share/java/servlet.jar:. HelloTestServ.java
```

das Classfile sollte nun (im einfachsten Fall) im Verzeichnis classes liegen

web.xml einrichten → next Page

http://content.hccfl.edu/pollock/ajava/war/myservletwar.htm

http://www.caucho.com/resin-3.0/servlet/tutorial/helloworld/index.xtp

. . .

.war File erzeugen: jar -cvf myServletWAR.war .

Deploy mit Tomcat Manager App

web.xml

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

<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN"

"http://java.sun.com/j2ee/dtds/web-app_2_2.dtd">

<web-app>
  <servlet>
    <servlet-name>HelloTestServ</servlet-name>
    <servlet-class>HelloTestServ</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>HelloTestServ</servlet-name>
    <url-pattern>/*</url-pattern>
    <url-pattern>/hello2011</url-pattern>

  </servlet-mapping>

</web-app>
```

Mit dem Mapping wird festgelegt, unter welchem Namen das Servlet später aufgerufen werden soll, es kann mehrere Mappingeinträge geben

The Apache Software Foundation
http://www.apache.org/

Tomcat Web Application Manager

Message:	OK

Manager			
List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Display Name	Running	Sessions	Commands
/	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/Hello2011		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/addrServ		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	Servlet and JSP	true	0	Start Stop Reload Undeploy

Das Deploying

Nutzung Tomcat Manager

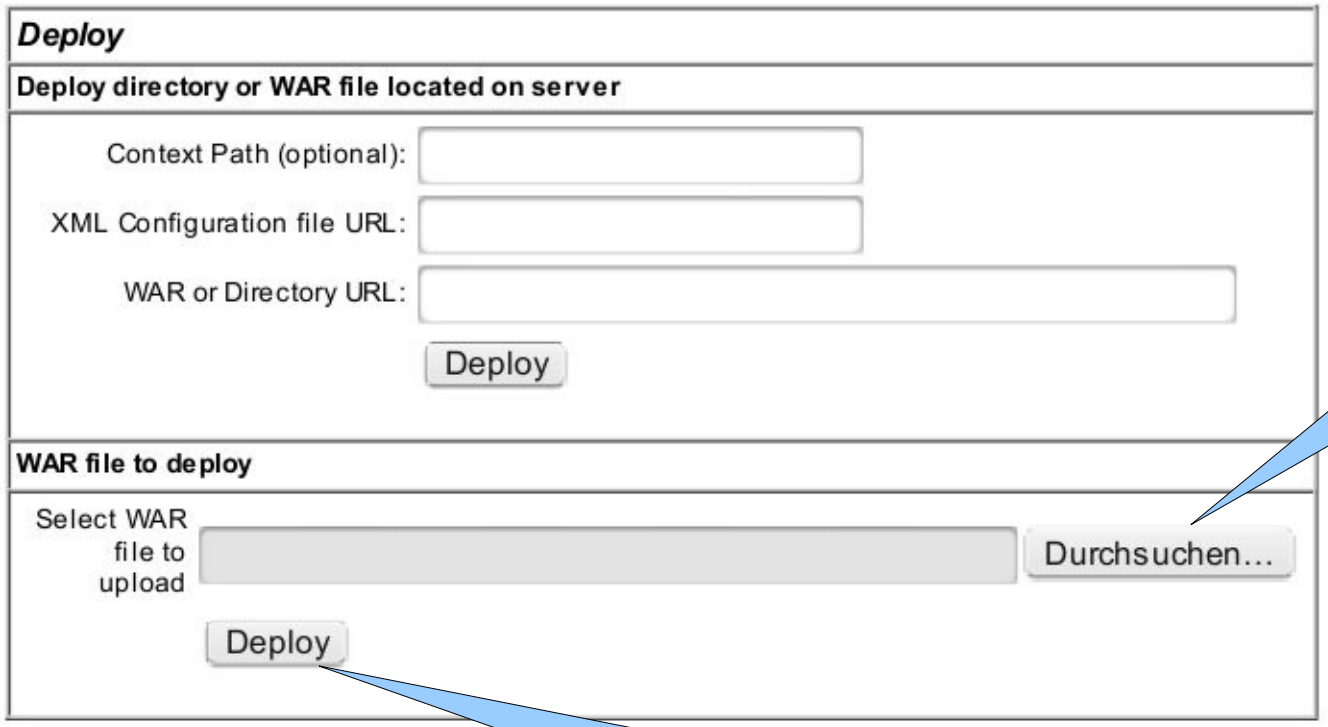
Password eintragen in

$CATALINA_HOME/conf/**tomcat-users.xml**

Der Pfad kann insbesondere bei Installation über eine Distribution variieren. (/etc/tomcat6 oder /usr/share/**tomcat6/conf**)

Ev. sind zusätzliche Einstellungen in server.xml nötig.

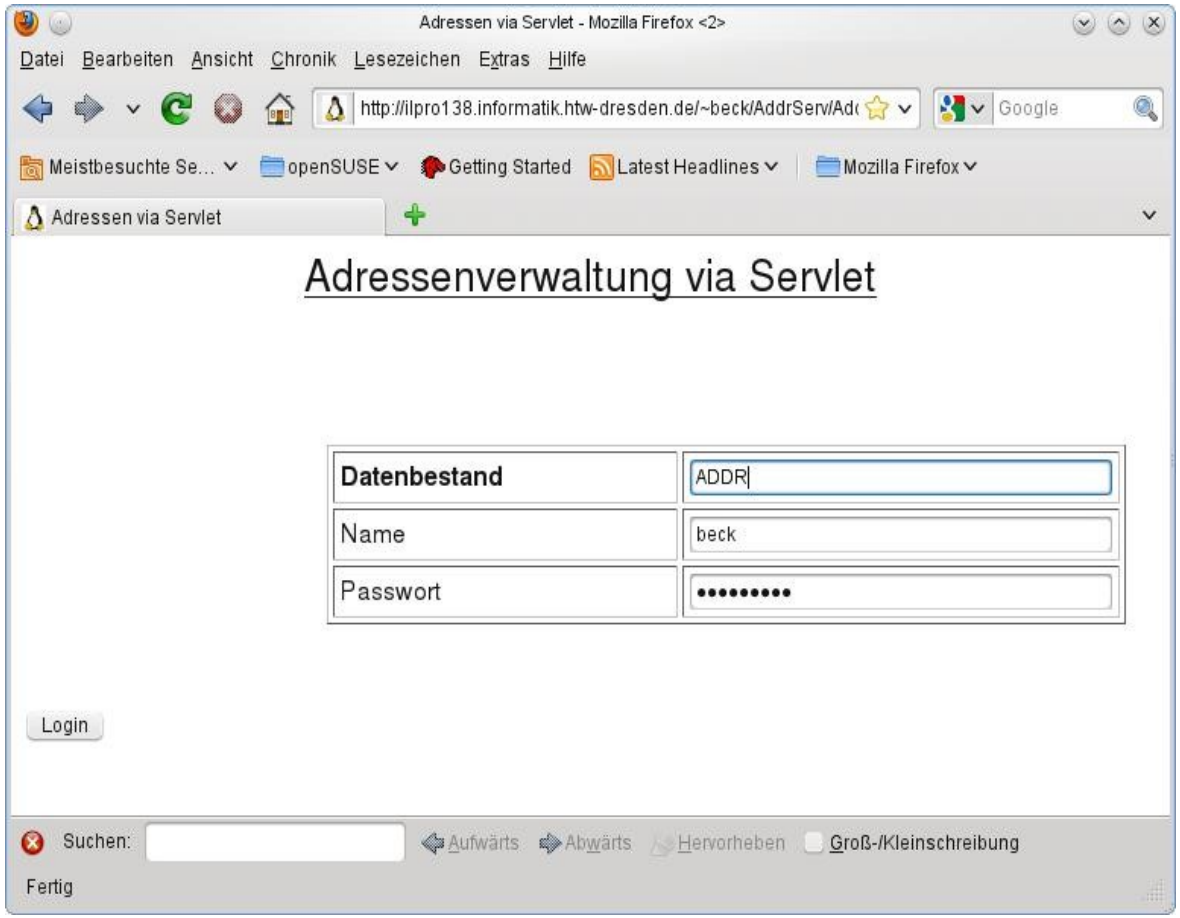
Aber nu - deploy!



War-file auf dem lokalen Rechner suchen

Und Deploy!
Das Warfile wird zum Server transportiert und eingebunden.
Es sollte nun in der Managerapp. erscheinen und aufrufbar sein

Webanwendung mit Servlet



html Formular

Eingabe der Daten, mit denen sich zur Datenbank verbunden wird.

Action:
Aufruf des Servlets
Anlegen einer Session mit Logindaten

Unterverzeichnis, falls vorhanden

ServletName, Dir unter webapps

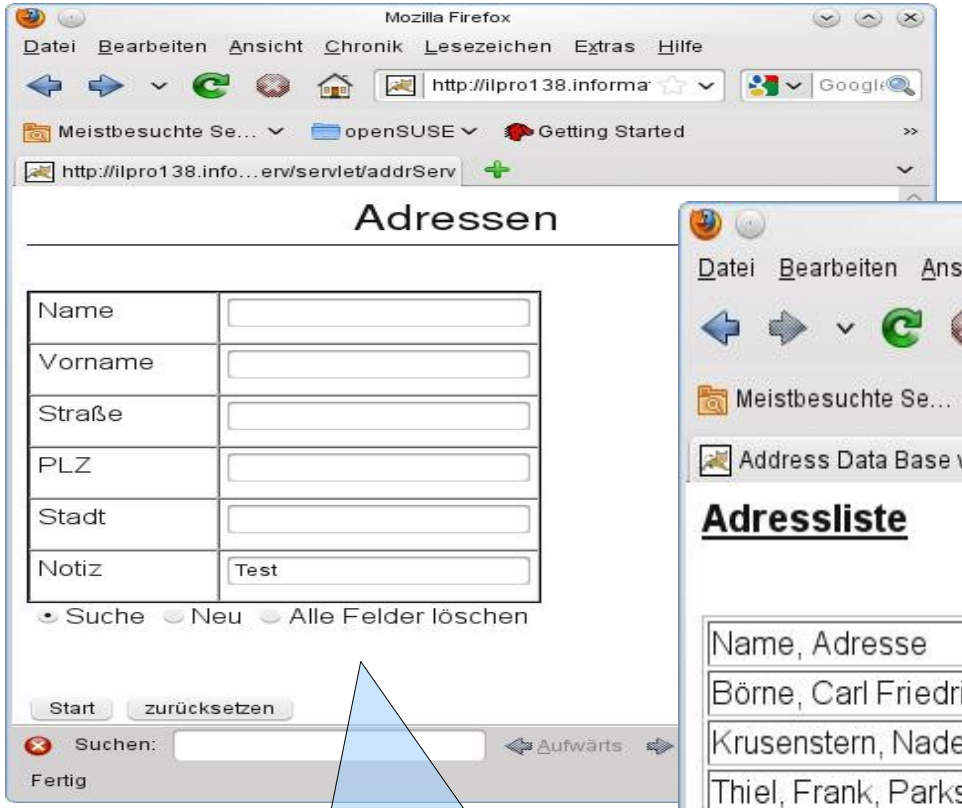
Class

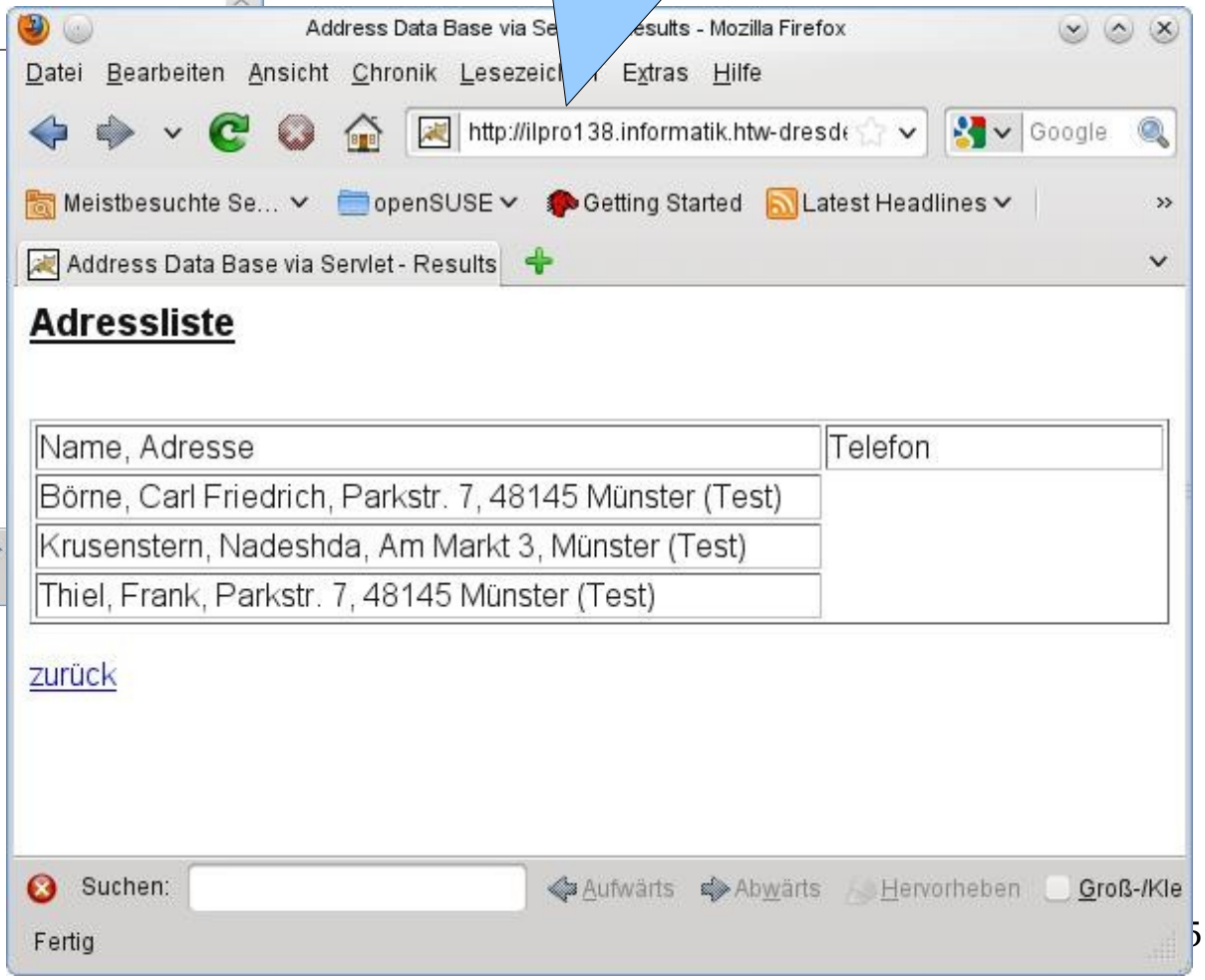
form action="http://ilpro138.informatik.htw-dresden.de:8080/addrServ/servlet/addrServ" method=POST>

Servlet in Dir classes

form action="http://ilpro138:8080/addrServ" method=POST>

Webanwendung mit Servlet





Ergebnis (html Table)

Eingabeformular:
Eingabe des Typs des Request
Eingabe von suchrelevanten Daten

```
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.net.*;
public class addrServ extends HttpServlet
{
  /*************************************************/
  /* Instanzvariable werden nur einmal angelegt und von */
  /* allen Sessions benutzt, sie tragen gewissermassen  */
  /* globalen Charakter                                 */
  /*************************************************/
  public void doGet(HttpServletRequest req, HttpServletResponse res)
  throws IOException, ServletException
  {
     PrintWriter out = res.getWriter();

     File F=new File("/srv/tomcat6/. . ./myAddrForm.html");
     String Line;
     BufferedReader bf=new BufferedReader(
                 new InputStreamReader(new FileInputStream(F)));
     while((Line=bf.readLine())!=null)
     {
        out.println(Line);
     }
  }
  . . .
}
```

doGet:
Eingabeformular anzeigen, es ist in der Datei `myAddrForm.html` gespeichert

doPost

In der Methode doPost steckt die Funktionalität des Servlets.

es werden verschiedene Arten Request behandelt

- Login
- Search
- New
- Clear

Nach Login wird eine Session mit den Verbindungssdaten zur Datenbank angelegt.

In allen anderen Fällen werden die Daten einer vorhandenen Session gelesen.

Die Session

Öffnen einer Session, ggf. neue Sesion anlegen

```
HttpSession Sess=req.getSession(true);
```

Öffnen einer Session, Session muss vorhanden sein

```
HttpSession Sess=req.getSession(false);
```

Eintragen von Attributen in die Session

```
Sess.setAttribute("User",User);
```

Lese von Attributen aus der Session

```
User=(String)Sess.getAttribute("User");
```

Das Formular

```
<form action="addrServ" method=POST>

<TABLE WIDTH=60% BORDER=1 BORDERCOLOR="#000000" CELLPADDING=4
CELLSPACING=0>
<TBODY>
<TR VALIGN=TOP>
      <TD WIDTH=50%> <P>Name</P> </TD>

      <TD WIDTH=50%> <P> <input type=text size=20 name=Name> </P> </TD>
</TR>
.  .  .
<TR VALIGN=TOP>
      <TD WIDTH=50%> <P> Notiz</P> </TD>

      <TD WIDTH=50%> <P> <input type=text size=20 name=Note> </P> </TD>
</TR>
</TBODY>
</TABLE>
<INPUT TYPE="radio" NAME="Command" VALUE="Suche"CHECKED>Suche
<INPUT TYPE="radio" NAME="Command" VALUE="Neu">Neu
<INPUT TYPE="radio" NAME="Command" VALUE="Clear">Alle Felder l&ouml;schen
<P ALIGN=LEFT STYLE="margin-bottom: 0cm"><BR>
<P ALIGN=LEFT   STYLE="margin-bottom: 0cm">
<input type=submit value=Start>
<input type=reset value=zur&uuml;cksetzen>
```

Ausgabe der Ergebnisse

Wird beim Start des Servlets angelegt.
stellt doConnect, doSelect … Methoden,
ist von der Class jdbcAddrJdbc aus der
JDBC-Application

```
Vector<Address> results=theDataBase.doSelect(A);
File f=new File(".../myAddrAnswer.html");
FileReader fr=new FileReader(f)
String Line;
BufferedReader bf=new BufferedReader(fr);
```

html-Dokument
der Antwortseite

Das Ergebnisdokument

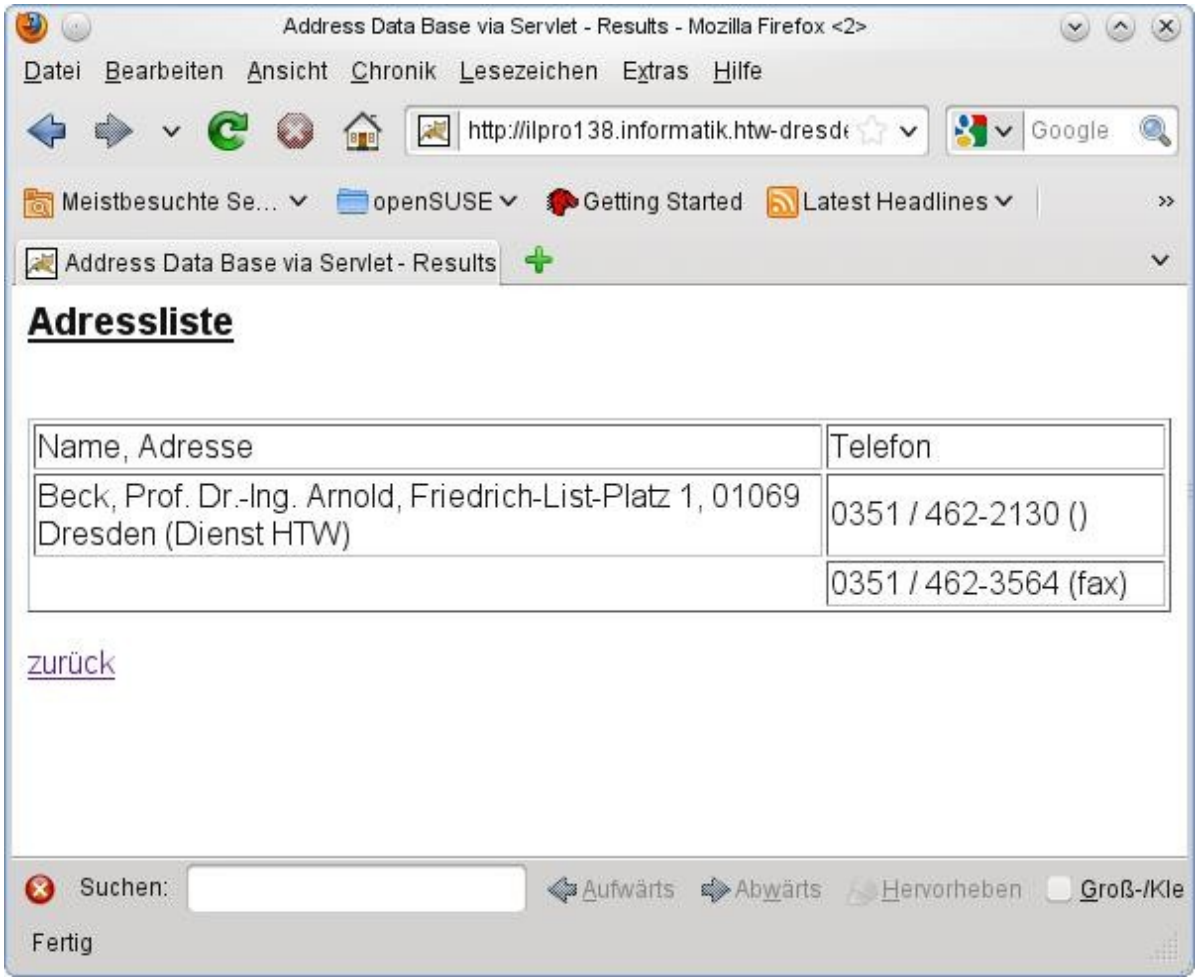
```
<body>

<h3>
<u>Adressliste</u></h3>
 
<table BORDER WIDTH="100%" NOSAVE >
<tr NOSAVE>
<td WIDTH="70%" NOSAVE>Name, Adresse</td>

<td WIDTH="30%" NOSAVE>Telefon</td>
</tr>

<!Tabelleneintraege>

</table>

<p><a href="http://ilpro138.informatik.htw-
dresden.de:8081/addr/servlet/addrServ">zur&uuml;ck</a>
</body>
```

Hier werden die Ergebnisse des Requests in die Tabelle eingefügt

Aufbau der Ergebnisseite

```
while((Line=bf.readLine())!=null)
{
 if (Line.indexOf("Tabelleneintraege")!=-1)
 {
  for (int i=0; i<Results.size(); i++)
  {
   A=(Address)(Results.get(i));
   out.print("<tr><td>");
   out.print(+A.Name+", "+A.FName+", "+A.Street+", "
                    +A.ZIP+" "+A.Town+" ("+A.Note+")"+"</td>");
   for (int j=0; j<A.Phones.size();j++)
   {
    out.print("<td>"
              +((Address.PhonNote)(A.Phones.get(j))).getString()
              +"</td></tr>");
    if (j+1<A.Phones.size()) out.print("<tr><td> </td>");
   }
  }
  Line=bf.readLine();
 }
 out.println(Line);
}
```

Leere Zelle für Adresse,
wenn es mehrere
Kontakteinträge
zu einer Adresse gibt


Address Data Base via Servlet - Results - Mozilla Firefox <2>
Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
http://ilpro138.informatik.htw-dresd
Google
Meistbesuchte Se...
openSUSE
Getting Started
Latest Headlines
Address Data Base via Servlet - Results
Adressliste
Name, Adresse
Telefon
Beck, Prof. Dr.-Ing. Arnold, Friedrich-List-Platz 1, 01069 Dresden (Dienst HTW)
0351 / 462-2130 ()
0351 / 462-3564 (fax)
zurück
Suchen:
Aufwärts
Abwärts
Hervorheben
Groß-/Kle
Fertig

Logging / Debugging

Es ist möglich, auf die Standardausgabe logging Informationen zu schreiben

Die Ausgabe landet in einem File catalina.out

/var/log/tomcat6 (Rootrechte nötig)

Bei einer Userinstallation liegt es im Verzeichnis logs.

Weiterführende Informationen zum Thema logging unter http://wiki.apache.org/tomcat/FAQ/Logging

```
beck@silent:~/apache-tomcat-7.0.6/logs> ls
catalina.2011-01-23.log  host-manager.2011-01-23.log  localhost_access_log.2011-01-23.txt
catalina.out             localhost.2011-01-23.log     manager.2011-01-23.log
beck@silent:~/apache-tomcat-7.0.6/logs>
```

Benutzerinstallation

Tomcat 6 oder 7 per kostenlosem Download als Archiv speichern.

Core:

* zip (pgp, md5)
* tar.gz (pgp, md5)
* 32-bit Windows zip (pgp, md5)
* 64-bit Windows zip (pgp, md5)
* 64-bit Itanium Windows zip (pgp, md5)
* 32-bit/64-bit Windows Service Installer (pgp, md5)

Archiv auspacken

edit conf/tomcat-users.xml → next Page

starten bin/startup.sh, stopp it bin/shutdown.sh

fertig in 10 min! oder weniger

tomcat-users.xml

```
<role rolename="tomcat"/>
<role rolename="role1"/>
<role rolename="manager-gui"/>
<role rolename="admin-gui"/>
<role rolename="manager-script"/>
<role rolename="manager-jmx"/>
<role rolename="manager-status"/>
<user username="tomcat"
      password=". . ."
      roles="tomcat,
             admin-gui,
             manager-gui,
             manager-script,
             manager-jmx,
             manager-status"/>
<user username="both"
      password=". . ."
      roles="tomcat,role1"/>
<user username="role1"
      password=". . ."
      roles="role1"/>
```

Diese Konfigurationsdatei
ist für Testzwecke konfiguriert.