# Uebung02 vesys

Daniel Gürber

4. Semester (FS 2013)

# Inhaltsverzeichnis

# 1 Beschreibung

## 1.1 Architektur

Die Bank wird mit Servlets als Website generiert. Für jede Page wurde ein Servlet erstellt, wobei die unterschieden wird ob der Pfad mit GET oder POST aufgerufen wurde. Bei GET werden Informationen angezeigt oder Formulare gestellt, bei POST werden Aktionen ausgeführt und gegebenenfalls weitergeleitet oder eine Fehlermeldung ausgegeben.

# 2 Code

## 2.1 Servlets

Listing 1: Default Servlet

```
1  package servlets;
   import java.io.IOException;
   import java.io.PrintWriter;
   import java.util.Set;

6  import javax.servlet.ServletException;
   import javax.servlet.annotation.WebServlet;
   import javax.servlet.http.HttpServlet;
   import javax.servlet.http.HttpServletRequest;
   import javax.servlet.http.HttpServletResponse;
11
   import bank.Account;
   import bank.implementation.Bank;

   @WebServlet("/")
16 public class DefaultServlet extends HttpServlet {

     /**
      * Generated Version UID
      */
21   private static final long serialVersionUID = -5601445569577136558L;

     @Override
     public void doGet(HttpServletRequest request, HttpServletResponse response)
         throws IOException, ServletException {
26     PrintWriter writer = response.getWriter();

       response.setContentType("text/html");

       Bank bank = Bank.getInstance();
31
       writer.write("<html><head><title>Bank</title></head><body><h1>Bank</h1>");
       writer.write("<a href=\"create\">Create Account</a><br/>");
       writer.write("<h2>Accounts:</h2>");
       writer.write("<table><tr><td><b>AccountNr</b></td><td><b>Owner</b></td><td><b>Balance</
           b></td><td></td></tr>");
36     Set<String> accnumbers = bank.getAccountNumbers();
       for (String accnumber : accnumbers) {
         Account account = bank.getAccount(accnumber);
         writer.write("<tr><td>" + accnumber + "</td>");
         writer.write("<td>" + account.getOwner() + "</td>");
41       writer.write(String.format("<td>%.2f</td>", account.getBalance()));
         writer.write("<td><a href=\"withdraw?number=" + accnumber + "\">Withdraw Money</a>&
             nbsp;");
         writer.write("<a href=\"deposit?number=" + accnumber + "\">Deposit Money</a> ");
         if (accnumbers.size() > 1)
           writer.write("<a href=\"transfer?number=" + accnumber + "\">Transfer Money</a> 
               ;");
46       writer.write("<a href=\"close?number=" + accnumber + "\">Close Account</a></td>");
       }
```

```
      writer.write("</table>");
      writer.write("</body></html>");
    }
  }
```

Listing 2: Create Servlet

```
package servlets;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import bank.Account;
import bank.implementation.Bank;

@WebServlet("/create")
public class CreateServlet extends HttpServlet{

  /**
   * Generated UID
   */
  private static final long serialVersionUID = 8439412655054799485L;

  @Override
  protected void doGet(HttpServletRequest req, HttpServletResponse resp)
      throws ServletException, IOException {

    PrintWriter writer = resp.getWriter();
    resp.setContentType("text/html");

    writer.write("<html><head><title>Bank</title></head><body><h1>Create Account</h1>");
    writer.write("<form action=\"create\" method=\"post\">");
    writer.write("Owner:<br/> <input type=\"text\" name=\"owner\"/><br/>");
    writer.write("Balance:<br/> <input type=\"text\" name=\"balance\"/><br/>");
    writer.write("<input type=\"submit\" name=\"submit\" value=\"Create\"/>");
    writer.write("</form></body></html>");
  }

  @Override
  protected void doPost(HttpServletRequest req, HttpServletResponse resp)
      throws ServletException, IOException {

    String owner = req.getParameter("owner");
    String balance = req.getParameter("balance");

    String error = null;

    if (owner==null || owner=="") {
      error="Owner not set!";
    } else {
      Bank bank = Bank.getInstance();
      String number = bank.createAccount(owner);

      if(number==null){
        error = "Account could not be created";
      }
      else {
        try {
          Account acc = bank.getAccount(number);
          double amount;
          if( balance==null || balance.equals("")) amount=0;
          else amount = Double.parseDouble(balance);
          acc.deposit(amount);
```

```
63            resp.sendRedirect("");
            }
            catch (NumberFormatException e) {
              error = "Illegal Format!";
            }
68          catch (Exception e) {
              error = e.getMessage();
            }
          }
        }
73
        PrintWriter writer = resp.getWriter();
        resp.setContentType("text/html");

        writer.write("<html><head><title>Bank</title></head><body><h1>Error</h1>");
78      if (error != null) {
          writer.write(error + "<br/>");
        }
        writer.write("<a href=\"create\">Back</a>");
        writer.write("</body></html>");
83
    }
  }
```

Listing 3: Deposit Servlet

```
  package servlets;

  import java.io.IOException;
  import java.io.PrintWriter;
5
  import javax.servlet.ServletException;
  import javax.servlet.annotation.WebServlet;
  import javax.servlet.http.HttpServlet;
  import javax.servlet.http.HttpServletRequest;
10 import javax.servlet.http.HttpServletResponse;

  import bank.Account;
  import bank.implementation.Bank;

15 @WebServlet("/deposit")
  public class DepositServlet extends HttpServlet{

    /**
     * Generated UID
20   */
    private static final long serialVersionUID = 8439412655054799485L;

    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
25      throws ServletException, IOException {

      String number = req.getParameter("number");
      if (number==null) number="";

30    PrintWriter writer = resp.getWriter();
      resp.setContentType("text/html");

      writer.write("<html><head><title>Bank</title></head><body><h1>Deposit Money</h1>");
      writer.write("<form action=\"deposit\" method=\"post\">");
35    writer.write("Number:<br/> <input type=\"text\" name=\"number\" value=\"" + number + "
          \"readonly></input><br/>");
      writer.write("Amount:<br/> <input type=\"text\" name=\"amount\"/><br/>");
      writer.write("<input type=\"submit\" name=\"submit\" value=\"Deposit\"/>");
      writer.write("</form></body></html>");
    }
40
    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
```

```
      String error = null;
      String number = req.getParameter("number");
      String amount = req.getParameter("amount");

        if (number == null || number == "") {
          error = "Number not provided!";
        } else {
          if (amount == null || amount == "") {
            error="Amount not provided!";
          } else {
            try {
              Bank bank = Bank.getInstance();
            double dblAmount = Double.parseDouble(amount);
            Account a = bank.getAccount(number);
            if (a==null) {
              error="Account does not exist!";
            } else {
              a.deposit(dblAmount);
              resp.sendRedirect("");
            }
            }
            catch (NumberFormatException e) {
            error = "Illegal Format!";
          }
          catch (Exception e) {
            error = e.getMessage();
          }
          }
        }


      PrintWriter writer = resp.getWriter();
      resp.setContentType("text/html");

      writer.write("<html><head><title>Bank</title></head><body><h1>Error</h1>");
      if (error != null) {
        writer.write(error + "<br/>");
      }
      writer.write("<a href=\"deposit?number=");
      if (number!=null) {
        writer.write(number);
      }
      writer.write("\">Back</a> ");
      writer.write("<a href=\"/bankservlet\">Home</a>");
      writer.write("</body></html>");

    }
  }
```

Listing 4: Withdraw Servlet

```
  package servlets;

  import java.io.IOException;
  import java.io.PrintWriter;

  import javax.servlet.ServletException;
  import javax.servlet.annotation.WebServlet;
  import javax.servlet.http.HttpServlet;
  import javax.servlet.http.HttpServletRequest;
  import javax.servlet.http.HttpServletResponse;

  import bank.Account;
  import bank.implementation.Bank;

  @WebServlet("/withdraw")
  public class WithdrawServlet extends HttpServlet{

    /**
     * Generated UID
```

```
 */
private static final long serialVersionUID = -4854407662295899521L;

@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp)
    throws ServletException, IOException {

  String number = req.getParameter("number");
  if (number==null) number="";

  PrintWriter writer = resp.getWriter();
  resp.setContentType("text/html");

  writer.write("<html><head><title>Bank</title></head><body><h1>Withdraw Money</h1>");
  writer.write("<form action=\"withdraw\" method=\"post\">");
  writer.write("Number:<br/> <input type=\"text\" name=\"number\" value=\"" + number + "
      \"readonly></input><br/>");
  writer.write("Amount:<br/> <input type=\"text\" name=\"amount\"/><br/>");
  writer.write("<input type=\"submit\" name=\"submit\" value=\"Withdraw\"/>");
  writer.write("</form></body></html>");
}

@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp)
    throws ServletException, IOException {
  String error = null;
  String number = req.getParameter("number");
  String amount = req.getParameter("amount");

    if (number == null || number == "") {
      error = "Number not provided!";
    } else {
      if (amount == null || amount == "") {
        error="Amount not provided!";
      } else {
        try {
          Bank bank = Bank.getInstance();
        double dblAmount = Double.parseDouble(amount);
        Account a = bank.getAccount(number);
        if (a==null) {
          error="Account does not exist!";
        } else {
          a.withdraw(dblAmount);
          resp.sendRedirect("");
        }
        }
        catch (NumberFormatException e) {
        error = "Illegal Format!";
      }
      catch (Exception e) {
        error = e.getMessage();
      }
      }
    }


  PrintWriter writer = resp.getWriter();
  resp.setContentType("text/html");

  writer.write("<html><head><title>Bank</title></head><body><h1>Error</h1>");
  if (error != null) {
    writer.write(error + "<br/>");
  }
  writer.write("<a href=\"withdraw?number=");
  if (number!=null) {
    writer.write(number);
  }
  writer.write("\">Back</a> ");
  writer.write("<a href=\"/bankservlet\">Home</a>");
  writer.write("</body></html>");

}
```

```
}
```

Listing 5: Transfer Servlet

```
package servlets;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import bank.Account;
import bank.implementation.Bank;

@WebServlet("/transfer")
public class TransferServlet extends HttpServlet{

  /**
   * Generated UID
   */
  private static final long serialVersionUID = 1837264773636513445L;

  @Override
  protected void doGet(HttpServletRequest req, HttpServletResponse resp)
      throws ServletException, IOException {

    String number = req.getParameter("number");
    if (number==null) number="";

    PrintWriter writer = resp.getWriter();
    resp.setContentType("text/html");

    writer.write("<html><head><title>Bank</title></head><body><h1>Transfer Money</h1>");
    writer.write("<form action=\"transfer\" method=\"post\">");
    writer.write("From:<br/> <input type=\"text\" name=\"number\" value=\"" + number + "\"
        readonly></input><br/>");
    writer.write("To:<br/> <select name=\"to\">");
    for (String accnumber : Bank.getInstance().getAccountNumbers()) {
      if (!accnumber.equals(number))
        writer.write("<option value=\"" + accnumber + "\">" + accnumber + "</option>");
    }
    writer.write("</select><br/>");
    writer.write("Amount:<br/> <input type=\"text\" name=\"amount\"/><br/>");
    writer.write("<input type=\"submit\" name=\"submit\" value=\"Transfer\"/>");
    writer.write("</form></body></html>");
  }

  @Override
  protected void doPost(HttpServletRequest req, HttpServletResponse resp)
      throws ServletException, IOException {
    String error = null;
    String number = req.getParameter("number");
    String to = req.getParameter("to");
    String amount = req.getParameter("amount");

      if (number == null || number == "") {
        error = "Number not provided!";
      } else {
        if (amount == null || amount == "") {
          error="Amount not provided!";
        } else {
          if (to == null || to == "") {
            error="To not provided!";
          } else {
            try {
              Bank bank = Bank.getInstance();
```

```
              double dblAmount = Double.parseDouble(amount);
              Account a = bank.getAccount(number);
              Account b = bank.getAccount(to);
              if (a==null || b==null) {
                error="Account does not exist!";
              } else {
                bank.transfer(a,b,dblAmount);
                resp.sendRedirect("");
              }
              }
              catch (NumberFormatException e) {
              error = "Illegal Format!";
            }
            catch (Exception e) {
              error = e.getMessage();
            }
            }
          }
        }

      PrintWriter writer = resp.getWriter();
      resp.setContentType("text/html");

      writer.write("<html><head><title>Bank</title></head><body><h1>Error</h1>");
      if (error != null) {
        writer.write(error + "<br/>");
      }
      writer.write("<a href=\"transfer?number=");
      if (number!=null) {
        writer.write(number);
      }
      writer.write("\">Back</a> ");
      writer.write("<a href=\"/bankservlet\">Home</a>");
      writer.write("</body></html>");

    }
  }
```

Listing 6: Close Servlet

```
  package servlets;

  import java.io.IOException;
  import java.io.PrintWriter;

  import javax.servlet.ServletException;
  import javax.servlet.annotation.WebServlet;
  import javax.servlet.http.HttpServlet;
  import javax.servlet.http.HttpServletRequest;
  import javax.servlet.http.HttpServletResponse;

  import bank.implementation.Bank;

  @WebServlet("/close")
  public final class CloseServlet extends HttpServlet {

    /**
     * Generated UID
     */
    private static final long serialVersionUID = -3483736040819510349L;


    @Override
    protected void doGet(final HttpServletRequest req,
                final HttpServletResponse resp)
        throws ServletException, IOException {

      String number = req.getParameter("number");
      if (number == null) {
```

```
        number = "";
      }

      PrintWriter writer = resp.getWriter();
      resp.setContentType("text/html");

      writer.write("<html><head><title>Bank</title></head>");
      writer.write("<body><h1>Close Account</h1>");
      writer.write("<form action=\"close\" method=\"post\">");
      writer.write("Number:<br/> <input type=\"text\" name=\"number\"");
      writer.write("value=\"" + number + "\"readonly></input><br/>");
      writer.write("Close this Account?<br/>");
      writer.write("<input type=\"submit\" name=\"submit\" value=\"OK\"/>");
      writer.write("</form></body></html>");
    }

    @Override
    protected void doPost(final HttpServletRequest req,
                final HttpServletResponse resp)
        throws ServletException, IOException {
      String error = null;
      String number = req.getParameter("number");

      if (number == null || number == "") {
        error = "Number not provided!";
      } else {
        try {
          Bank bank = Bank.getInstance();
        if (bank.closeAccount(number)) {
          resp.sendRedirect("");
        } else {
          error = "Could not close Account!";
        }
        } catch (Exception e) {
        error = e.getMessage();
      }
      }


      PrintWriter writer = resp.getWriter();
      resp.setContentType("text/html");

      writer.write("<html><head><title>Bank</title>");
      writer.write("</head><body><h1>Error</h1>");
      if (error != null) {
        writer.write(error + "<br/>");
      }
      writer.write("<a href=\"close?number=");
      if (number != null) {
        writer.write(number);
      }
      writer.write("\">Back</a> ");
      writer.write("<a href=\"/bankservlet\">Home</a>");
      writer.write("</body></html>");

    }
  }
```

## 2.2 Bank

Listing 7: Bank

```
  package bank.implementation;

  import java.io.IOException;
  import java.util.HashMap;
  import java.util.HashSet;
  import java.util.Map;
  import java.util.Set;
```

```
import bank.InactiveException;
import bank.OverdrawException;

public final class Bank implements bank.Bank {
  private static Bank instance;

  public static Bank getInstance() {
    if (instance == null) {
      instance = new Bank();
    }
    return instance;
  }

  private Map<String, Account> accounts = new HashMap<String, Account>();

  @Override
  public Set<String> getAccountNumbers() {
    Set<String> activeNumbers = new HashSet<String>();
    for (Account account : accounts.values()) {
      if (account.isActive()) {
        activeNumbers.add(account.getNumber());
      }
    }
    return activeNumbers;
  }

  @Override
  public String createAccount(final String owner) {
    Account newAccount = new Account(owner);
    accounts.put(newAccount.getNumber(), newAccount);
    return newAccount.getNumber();
  }

  @Override
  public boolean closeAccount(final String number) {
    Account closeAccount = accounts.get(number);
    if (closeAccount != null
        && closeAccount.getBalance() == 0
        && closeAccount.isActive()) {
      closeAccount.setActive(false);
      return true;
    } else {
      return false;
    }
  }


  @Override
  public bank.Account getAccount(final String number) {
    return (bank.Account) accounts.get(number);
  }

  /**
   * Transfers the given amount from account a to account b.
   *
   * @param a account to withdraw amount from
   * @param b account to deposit amount
   * @param amount value to transfer
   * @pre amount >= 0
   * @throws InactiveException if one of the two accounts is not active
   * @throws OverdrawException if the amount is greater than the balance of
   *             account a
   * @throws IllegalArgumentException if the argument is negative
   * @throws IOException if a remoting or communication problem occurs
   */
  @Override
  public void transfer(final bank.Account from,
              final bank.Account to,
              final double amount)
      throws IOException, InactiveException, OverdrawException {

    if (!from.isActive()) {
```

```
        throw new InactiveException("Source account is closed!");
      }

      if (!to.isActive()) {
        throw new InactiveException("Target account is closed!");
      }

      from.withdraw(amount);
      to.deposit(amount);
    }
  }
```

Listing 8: Account

```
package bank.implementation;

import java.util.UUID;

import bank.InactiveException;
import bank.OverdrawException;

public final class Account implements bank.Account {
  private String number;
  private String owner;
  private double balance;
  private boolean active = true;

  Account(final String owner) {
    this.owner = owner;
    this.number = UUID.randomUUID().toString();
  }

  @Override
  public double getBalance() {
    return balance;
  }

  @Override
  public String getOwner() {
    return owner;
  }

  @Override
  public String getNumber() {
    return number;
  }

  @Override
  public boolean isActive() {
    return active;
  }

  void setActive(final boolean active) {
    this.active = active;
  }


  @Override
  public void deposit(final double amount)
      throws InactiveException {
    if (!this.isActive()) {
      throw new InactiveException("Account is closed!");
    }

    if (amount < 0) {
      throw new IllegalArgumentException("Amount can't be negative!");
    }

    this.balance += amount;
  }
```

```
    @Override
59  public void withdraw(final double amount)
        throws InactiveException, OverdrawException {
      if (!this.isActive()) {
        throw new InactiveException("Account is closed!");
      }
64
      if (amount < 0) {
        throw new IllegalArgumentException("Amount can't be negative!");
      }

69    if (amount > balance) {
        throw new OverdrawException("Not enough money on account!");
      }

      this.balance -= amount;
74  }

}
```