

Praktikum Web-Technologien

WWW-Technologien (HTML, Formular-Technik, PHP) und Web-Services in der Lehrveranstaltung Verteilte Systeme (RN III)

Zielstellung

Im Rahmen der LVA „Verteilte Systeme“ werden die Komplexe **WWW** (Client/Server-Architektur, Web-Seiten und Interaktionen im WWW) und **Web-Services** (Nutzung von Diensten im WWW zur Realisierung verteilter Anwendungen) behandelt. Das Praktikum dient zur stofflichen Vertiefung und beinhaltet sowohl den praktischen Umgang mit diesen Technologien und als auch das Selbststudium zu WWW und Web-Services unter PHP.

Es werden 3 Teilkomplexe als Aufgabe zu einem „Einfachen Taschenrechner“ gestellt:

- Teilkomplex 1: Erstellung einer eigenen Homepage (Tags, Listen, Email, Hyperlinks)
- Teilkomplex 2: PHP-Taschenrechner. Nutzung der Formulartechnik zur Erstellung dynamischer Webseiten (Dateneingabe) und PHP (Personal Hypertext Pre-Processor),
- Teilkomplex 3: SOAP-Rechner. Programmierung unter Nutzung von Formulareingaben, Web-Services unter PHP und bereitgestellten Web-Services-Implementationen.

Dazu wird das SW-System **xampplite** genutzt, das neben dem Apache-Server auch eine PHP-Implementation mit Web-Services bereitstellt, die über SOAP aufgerufen werden können. Zur Anleitung sind im Verzeichnis *xampplite/htdocs/pki* Musterdateien *index.html* und *index.php* (PHP- und SOAP-Client) sowie für den SOAP-Rechner die WSDL-Spezifikation *freeservices.wsdl* und der SOAP-Server *index.php* angegeben. Der Aufruf erfolgt im Browser über den Apache-Server mit der URL <http://localhost/pki/index.html> (bei lokaler Installation). (Anm.: Installation mit freundlicher Unterstützung durch Prof. J.-A. Müller, HTW Dresden bzw. HfTL Leipzig (FH der Telekom) und Dipl.-Ing. (BA) A. Grimm, BA Gera).

Die Teilaufgaben sind vorzuführen und als Beleg zu dokumentieren (Sourcecode auch elektronisch). Das Ergebnis ist Teil der benoteten Leistungsbewertung zu „Verteilte Systeme“.

Voraussetzungen

Sie benötigen

- einen öffentlich zugängigen Web-Server mit eigenem Home-Verzeichnis (an BA Gera: URL des Web-Server mit Home-Verzeichnis *http://192.168.41.3/pi0x/studname*).

- das Programmpaket xampplite (mit Apache-Server, PHP und Web-Services),
- ein persönliches digitales „Passbild“ (passbild.gif),
- einen Editor (z.B. Total Commander, Text Pad 4, Eclipse, JCreator, VS 2005),
- Kenntnisse zu PHP (Selbststudium). PHP ist eine interpretierte Skriptsprache, die sowohl an der Kommandozeile als auch als Modul des Web-Servers läuft. PHP ist bei Kenntnissen von C/C++ leicht erlernbar.

Im Teilkomplex 3 (Web-Services mit PHP) wird eine Installation PHP 5.2 auf dem Apache-Server genutzt. Diese verfügt über eine leistungsfähige SOAP-Implementierung. Dabei werden im Verzeichnis *freeservices* Beispieldienste angeboten, mit denen eine einfach Realisierung des „Taschenrechners“ möglich ist. Der Aufruf der Dienste erfolgt über SOAP, die Beschreibung der Dienste über WSDL, die über die Implementierung abgerufen werden können.

Literatur

Quellen (Online):

- Irmscher, K.: WebScriptum_IntW3. <http://www.informatik.uni-leipzig.de/~irmscher> .
11.11.2011

WWW:

- Münz, S.: SELFHTML. <http://selfhtml.org/> .
- Tolksdorf, R.: Die Sprachen des Web: HTML und XHTML. dpunkt, Heidelberg, 2000
- Wilde, E.: World Wide Web. Technische Grundlagen. Springer, Heidelberg, 2000.

PHP (Online-Quellen):

- Sebastian Bergmann: Professionelle Software-Entwicklung mit PHP 5 – Objektorientierung. Entwurfsmuster. Modellierung. Fortgeschrittene Datenbankprogrammierung. 1. Auflage, 2005. http://www.professionelle-softwareentwicklung-mit-php5.de/erste_auflage/
- Thomas Theis: PHP 4 – Webserver-Programmierung für Einsteiger. Galileo Computing, 1. Auflage, 2005. <http://www.galileocomputing.de/openbook/php4/>
- PHP Handbuch. <http://www.php.net/manual/de/>

Aufgabenstellung Teilkomplex 1 („Homepage“)

Die Aufgabe besteht darin, eine einfache persönliche Web-Seite mit Angaben zur Person, der Studienrichtung und der Berufsakademie zu erstellen und in dem öffentlichen Verzeichnis abzulegen, aus dem es dann mit einem Browser aufgerufen und dargestellt werden kann.

Es steht Ihnen frei, die inhaltlichen Angaben zu variieren, es sollten aber die geforderten Gestaltungselemente (auch in der angegebenen Reihenfolge) enthalten sein.

Die Programmierung erfolgt in HTML (empfohlen HTML 2.0). Ziel ist, einfache Kenntnisse zur Strukturierung und Darstellung einer Web-Seite nachzuweisen. Es geht dabei um die Auszeichnungssprache HTML, weniger um Fertigkeiten mit kommerziellen graphischen Webtools.

HTML-Programm

Title-Zeile mit Name, Vorname

Darzustellender Inhalt (Browser-Ausgabe auf Display)

Berufsakademie Gera, Datum

[Allgemeine Angaben](#)

Staatliche Studienakademie Thüringen

Berufsakademie Gera

Adresse der BA Gera

<Überschrift der Größe 2>

Absatzmarke

Fettdruck, Umlaut

Seminargruppe Nr.

Studienrichtung Praktische Informatik

Name, Vorname

Geburtsort

Geburtsdatum

<Passbild>

mailto: email-address

Horizontale Linie

Fettdruck

Kursivschrift

Fettdruck

) als beschreibende

) Liste

gif-Datei

Zur nächsten Seite: [Formulareingabe](#)

Horizontale Linie

TK1: leer; TK2/3: Hyperlink 2. Seite

Horizontale Linie

Lehrveranstaltungen im Semester:

1.

2.

...

zurück: [Allgemeine Angaben](#)

)

) als nummerierte Liste

)

Hyperlink

Nachweis

- Dokumentation Beleg TK 1: Deckblatt mit Name, Vorname, SG-Nr., verwendete Programme, Speicherort und Aufruf sowie Quellprogramme und Displayausgabe (Ausdruck).
- Quellprogramme (elektronisch).
- Funktionsnachweis (Vorführung) am Rechner.

Aufgabenstellung Teilkomplex 2 („Formulare und PHP“)

Im Teilkomplex 2 ist die Eingabe von Datenwerten, der Aufruf des Serverprogramms (Apache-Server) mit Übergabe der Datenwerte unter Nutzung der Formulartechnik (form-Tag) und die PHP-Technologie zu gestalten. Als Homepage dient die HTML-Datei aus TK 1.

Erweitern Sie Ihr Programm durch eine 2. Web-Seite, die von Seite 1 über einen Hyperlink zu erreichen ist. Für den form-Tag wird die Methode POST empfohlen. Die HTML-Datei und PHP-Datei sind im gleichen Verzeichnis in Ihrem Home-Verzeichnis als Unterverzeichnis von *hotdocs* abzulegen (BA Gera: *hotdocs/pi0x/studname*).

Datei *<index>.html*: Mit dem form-Tag ist folgendes Formular zu gestalten (Radio Button):

Taschenrechner:

Operation Addition Subtraktion Multiplikation Division

Variable a:

Variable b:

[zurück](#)

Datei *<index>.php*: Das PHP-Programm auf dem Server dient zur Auswertung der Formulareingaben und Berechnung gemäß ausgewählter Operation. Es erzeugt eine 3. Web-Seite mit Möglichkeit zur Wiederholung der Formulareingabe bzw. des Gesamtvorganges.

Taschenrechner mit PHP-Server (Programm *index.php*)

Operation: <...>
<Operanden und Ergebnis>

[Wiederholung Formulareingabe](#)

[zurück](#)

Nachweis

- Dokumentation Beleg TK 2: Deckblatt mit Name, Vorname, SG-Nr., verwendete Programme, Speicherort und Aufruf sowie Quellprogramme und Displayausgabe (Ausdruck).
- Quellprogramme (elektronisch).
- Funktionsnachweis (Vorführung) am Rechner.

Aufgabenstellung Teilkomplex 3 (Web-Services mit PHP)

Zur Vorbereitung ist ein Selbststudium zu PHP empfehlenswert, mit dem Sie sich einen Überblick zu PHP und angebotenen Web-Services verschaffen sollten. Dazu können Sie die angegebenen Online-Quellen benutzen. Es wird empfohlen, diese Selbststudium vor der eigentlichen Lösung des Teilkomplexes 3 durchzuführen.

Erstellen Sie mittels prozeduraler Programmierung einen einfachen Taschenrechner für die vier Grundrechenarten mit jeweils zwei Operanden. Für die Eingabe der Werte ist eine formulargesteuerte Web-Anwendung zu nutzen (analog zu Teilkomplex 2, mit Auswahlliste für die jeweilige Operation). Für die Rechenoperationen sind Web-Services einzusetzen.

PHP, Version 5.2 bietet für die Nutzung von Web-Services sowohl eine leistungsfähige SOAP-Implementierung als auch im Verzeichnis *freeservices* Dienste an, die per SOAP genutzt werden können. Dazu sind die Dienste über WSDL zu beschreiben und über SOAP aufzurufen, die über die PHP-Implementation abgerufen werden können.

Im xampplite-System sind für das Musterbeispiel die Dienste zu finden im Verzeichnis

xampplite/htdocs/pki die Dateien *index.html* und *index.php* sowie im Unterverzeichnis *freeservices* die Dateien für den SOAP-Rechner *freeservices.wsdl* und *index.php*.

Sie können im xampplite nach Start des Apache-Servers genutzt werden und über den Browser mit der URL <http://localhost/htdocs/pki> aufgerufen werden (Anm.: Pfad bei zentraler Installation siehe Anhang).

Ihre Aufgabe besteht darin, folgende Implementierungen durchzuführen:

- einen Web-Client (.html) für die Parametereingabe und Ergebnisausgabe,
- einen SOAP-Client (.php): Parameterauswertung und Aufruf der *freeservices*-Funktionen,
- eine WSDL-Beschreibung (.wsdl) zur Schnittstellenbeschreibung der WS-Funktionen,
- einen SOAP-Server (.php) zur prozeduralen Implementierung des Taschenrechners.

Als Homepage kann die html-Datei aus TK 1 bzw. TK 2 eingesetzt werden.

Nachweis

- Dokumentation Beleg TK 3: Deckblatt mit Name, Vorname, SG-Nr., verwendete Programme, Speicherort und Aufruf sowie Quellprogramme und Displayausgabe (Ausdruck).
- Quellprogramme (elektronisch).
- Funktionsnachweis (Vorführung) am Rechner.

Musterdateien

Im Verzeichnis *xampplite/hotdocs/pki*:

Homepage und Formular (index.html)

```
<html>
<head>
<title>Formular: PHP und SOAP</title>
<body>
HTML und Formular (Aufruf PHP-Script)<BR>
<HR>
<BR>
<form method="POST" action="index.php">
<input name="a" type="text" value="1">
<input name="b" type="text" value="2">
<P>
<input type="submit" value="Addieren">
<input type="reset" value="Reset">
</form>
</body>
</html>
```

Implementierung eines PHP- und SOAP-Clients (index.php)

```
<html>
<head>
<title>Formular mit PHP und SOAP</title>
</head>
<body>
Formularauswertung am Apache-Server <br>
Operation Addition: Parameter und Ergebnis <br>
<hr>
<br>

<?php
echo "Addition mit PHP:" . " <br>\n";
echo $_POST['a'] . " + " . $_POST['b'] . " = " . ($_POST['a'] +
$_POST['b']) . "<br>\n";
?>
<br>
<hr>
<br>

<?php
$client=newSoapClient
    ('http://localhost/pki/freeservices/?wsdl');
echo "Addition mit SOAP: " . " <br>\n";
echo $_POST['a'] . " + " . $_POST['b'] . " = " . $client
->vectoradd(array($_POST['a'],$_POST['b'])) . "<br>\n";
?>

<br>
<hr>
<a href="index.html">zurück</a>
</body>
</html>
```

Im Unterverzeichnis .../pki/freeservices

Implementierung eines SOAP-Servers (index.php)

```
<?php
class freeservices {
    public function test() {
        return "Hello World";
    }
    public function xecho($a) {
        return $a;
    }
    public function echox($a) {
        return strrev($a);
    }
    public function splitstring($a) {
        return split(" ", $a);
    }
    public function vectoradd($vector) {
        $result = 0.0;
        foreach ($vector as $value) {
            $result += $value;
        }
        return $result(" ", $a);
    }
}
try {
    $server = new SOAPServer ('freeservices.wsdl');
    $server->setClass('freeservices');
    if ((($_SERVER["REQUEST_METHOD"] == "POST")
        || (($_SERVER["REQUEST_METHOD"] == "GET")
            && ($_SERVER["QUERY_STRING"] == "wsdl")
        )
    ) {
        $server->handle();
    }
}
catch (SOAPFault $f) {
    print $f->faultstring;
}
?>
```

WSDL-Beschreibung der Web-Services für den Taschenrechner (freeservices.wsdl)

```
<?xml version="1.0"?>
<definitions name="tr"
  targetNamespace="urn:myTargetNamespace"
  xmlns:tns="urn:myTns"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <message name="addRequest">
    <part name="var_a" type="xsd:string"/>
    <part name="var_b" type="xsd:string"/>
  </message>
  <message name="addResponse">
    <part name="Result" type="xsd:string"/>
  </message>

  <message name="subRequest">
    <part name="var_a" type="xsd:string"/>
    <part name="var_b" type="xsd:string"/>
  </message>
  <message name="subResponse">
    <part name="Result" type="xsd:string"/>
  </message>

  <message name="mulRequest">
    <part name="var_a" type="xsd:string"/>
    <part name="var_b" type="xsd:string"/>
  </message>
  <message name="mulResponse">
    <part name="Result" type="xsd:string"/>
  </message>

  <message name="divRequest">
    <part name="var_a" type="xsd:string"/>
    <part name="var_b" type="xsd:string"/>
  </message>
  <message name="divResponse">
    <part name="Result" type="xsd:string"/>
  </message>

  <portType name="TaschenrechnerPortType">
    <operation name="add">
      <input message="tns:addRequest"/>
      <output message="tns:addResponse"/>
    </operation>
    <operation name="sub">
      <input message="tns:subRequest"/>
      <output message="tns:subResponse"/>
    </operation>
    <operation name="mul">
      <input message="tns:mulRequest"/>
      <output message="tns:mulResponse"/>
    </operation>
    <operation name="div">
      <input message="tns:divRequest"/>
      <output message="tns:divResponse"/>
    </operation>
  </portType>

  <binding name="TaschenrechnerBinding" type="tns:TaschenrechnerPortType">
    <soap:binding style="rpc"
      transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="add">
      <soap:operation soapAction=""/>

```

```

    <input>
      <soap:body use="encoded" namespace="urn:myInputNamespace"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </input>
    <output>
      <soap:body use="encoded" namespace="urn:myOutputNamespace"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </output>
  </operation>

  <operation name="sub">
    <soap:operation soapAction="" />
    <input>
      <soap:body use="encoded" namespace="urn:myInputNamespace"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </input>
    <output>
      <soap:body use="encoded" namespace="urn:myOutputNamespace"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </output>
  </operation>

  <operation name="mul">
    <soap:operation soapAction="" />
    <input>
      <soap:body use="encoded" namespace="urn:myInputNamespace"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </input>
    <output>
      <soap:body use="encoded" namespace="urn:myOutputNamespace"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </output>
  </operation>

  <operation name="div">
    <soap:operation soapAction="" />
    <input>
      <soap:body use="encoded" namespace="urn:myInputNamespace"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </input>
    <output>
      <soap:body use="encoded" namespace="urn:myOutputNamespace"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </output>
  </operation>
</binding>

<service name="freeservice">
  <documentation>Einfacher Taschenrechner</documentation>
  <port name="FreePort" binding="tns:freeBinding">
    <soap:address location="http://localhost/pki/freeservices/index.php" />
  </port>
</service>
</definitions>

```

Anhang:

Web-Server mit PHP (Apache-Server), Programmsystem xampplite

a) bei lokaler Installation

Verzeichnisstruktur

```
htdocs
  pki
    freeservices
    index.html
    index.php
```

Start Apache-Server

```
xampplite
  xampplite-control
    Apache Start ~> Running
                          Server läuft im Hintergrund, Port 80
                          ☉ -Zeichen in Fußleiste
```

Browser (z.B. Firefox)

Aufruf der Datei index.html: <http://localhost/pki/index.html>

Beenden Apache-Server

```
☉ -Zeichen in Fussleiste anklicken
  Apache Stop ~> Running erlischt
          Exit ~> ☉ -Zeichen in Fußleiste erlischt
```

Total Commander

<datei> im Verzeichnis hotdocs/pki

F3 Anzeige („Lister“)
Option: „1“ nur Text
über Menü Datei:
- „Speichern unter“
- „Drucken“

F4: Bearbeiten („Editor“)
über Menü Datei:
- „Speichern“
- „Drucken“

b) Zentrale Installation im ET-L der BA Gera

xampplite ist zentral auf einem Web-Server //192.168.41.3 (bzw. //Labor-Technik/) installiert. Start und Beendigung erfolgt durch den Systemadministrator.

Verzeichnisstruktur

```
htdocs
  pi0x (→ freigegeben)
    studname
      pki (→ Musterdateien)
        index.html
        index.php
        freeservices
          freeservices.wsdl
          index.php
```

Verzeichnispfad: <http://192.168.41.3/pi0x/studname>

Aufruf im Browser:
Musterdateien <http://192.168.41.3/pi0x/pki/index.html>

Studentische Dateien <http://192.168.41.3/pi0x/studname/index.html>