Internet/Intranet-Serverprogrammierung Entwicklung von CGI-Skripten in Perl

Wahlpflichtfach: Internet/Intranet

Lehrveranstaltung: Serverprogrammierung

Studiengang: alle

Prof. Dr. Wolf-Fritz Riekert
Hochschule für Bibliotheks- und Informationswesen Stuttgart

mailto:riekert@hbi-stuttgart.de

http://v.hbi-stuttgart.de/~riekert

Das World-Wide Web (WWW)

Client: Internet-Browser (z.B. Netscape Navigator, Microsoft Internet Explorer)

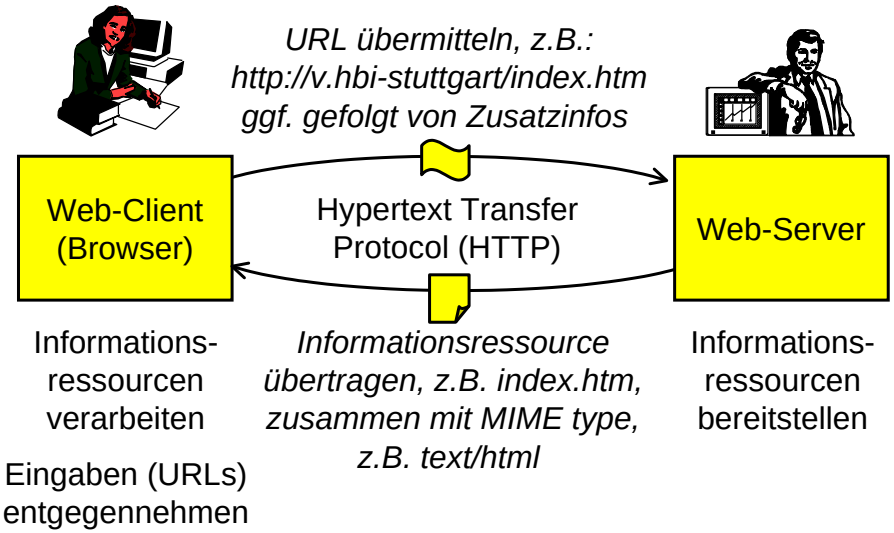
Server: Webserver (z.B. Microsoft Internet Information Server, Apache)

Dienst: Bereitstellen von Hypertextseiten und anderen Informationsressourcen (typisiert mit MIME Types) nach Angabe einer Adresse, der URL (Uniform Resource Locator)

Art des Dienstes: Verbindungsloser Anfrage/Antwort-Dienst

Protokoll: Hypertext Transfer Protokoll (HTTP).

Web-Client (Browser) und Web-Server



Web-Server

- erhält eine Informationsressourcenanforderung, welche im wesentlichen aus einer URL besteht
- stellt die Informationsressource bereit
  - ⇨ statisch: Informationsressource wird aus dem Dateisystem geholt
  - ⇨ oder dynamisch: Informationsressource wird durch ein Programm generiert
- legt den MIME-Type der Informationsressource fest: z.B. text/html oder anderer Typ (z.B. GIF-Graphik, Winword-Dokument, PDF-Dokument, ...)
- und schickt die Informationsressource zusammen mit dem MIME-Type an den Client (Internet-Browser) zurück

Web-Client (Internet Browser)

- verarbeitet die vom Web-Server erhaltenen Informationsressourcen abhängig von deren Typ
  - ⇨ direkte Anzeige: HTML-Seiten, GIF- bzw. JPEG-Grafiken
  - ⇨ direkte Ausführung: JavaScript, Java Applets, ActiveX Controls (letzteres nur Microsoft Internet Explorer)
  - ⇨ Anzeige/Ausführung über Plug-In (nachladbare Browser-Erweiterung): z.B. Acrobat Reader
  - ⇨ Anzeige/Ausführung durch sog. Helper Application: z.B. Winword für Doc-Files usw.
- nimmt Eingaben von URLs an und leitet diese weiter an Web-Server
  - ⇨ Direkteingabe über Tastatur
  - ⇨ Anklicken von Hyperlinks (mit URL hinterlegte Bereiche)

Uniform Resource Locator (URL)

URLs adressieren weltweit eindeutig Informationsressourcen (d.h. Daten, Dienstprogramme und multimediale Dokumente):

Aufbau: *Protokoll://Domain:Port/Pfad*

Beispiel: **http://www.faw.uni-ulm.de:9876/RESEDA/RESEDA.html**

(Die Zeichen //, :, / sind syntaktische Kennzeichnungen für die verschiedenen Elemente der URL)

Protokoll: = Übertragungsprotokoll (http: = Hypertext Transfer Protocol)

//Domain = Bezeichnung des Servercomputers im Internet

:Port = Kommunikationsport des Webserver-Programms, i.d.R. nicht erforderlich, da Standardwert = 80

/Pfad = Ortsangabe im Dateisystem des Servers, bestehend aus Verzeichnis(pfad) und Dateiname

URLs: Varianten

Relative URLs: Hypertextseiten enthalten oft relative Links. Das Protokoll, die Domain und der Schrägstrich vor dem Verzeichnispfad werden dann weggelassen. Beispiele:

- english.html (d.h. die Seite liegt im gleichen Verzeichnis wie aktuelle Hypertextseite)
- ../cgi-bin/test.cgi (liegt im Nachbarverzeichnis cgi-bin)

Andere Protokolle: Außer http: sind noch andere Protokolle möglich: **ftp:, gopher:, file:** (lokaler Dateizugriff ohne Server).

Wie ein Protokoll behandelt werden **mailto:** und **telnet:** (Aufruf des Mailsystems bzw. des Telnet-Clients für eine bestimmte Adresse)

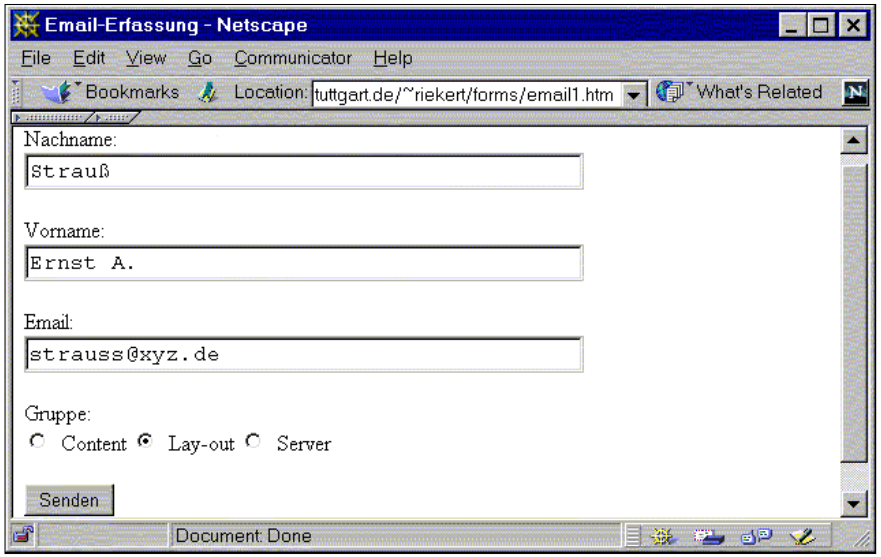
Web-Server-Programmierung

- Vom Web-Server bereitgestellte Informationsressourcen können dynamisch durch Programme erzeugt werden
- Der Pfadname in der URL bezeichnet das Programm, ggf. gefolgt durch mit Fragezeichen getrennte Parameter, z.B.: http://v.hbi-stuttgart.de/~sailer/cgi-bin/plan.cgi?Raum=w011

  Programmverzeichnis | Programm | Parameter
- Aufruf des Programms über spezielle Schnittstelle, am weitesten verbreitet CGI = „Common Gateway Interface“, da von allen Webservern unterstützt
- Alternative Schnittstellen: ISAPI* (Microsoft) oder NSAPI** (Netscape) - herstellerabhängig aber effizienter als CGI

* ISAPI = (Microsoft Internet) Information Server Application Programmer Interface
** NSAPI = Netscape Application Programmer Interface

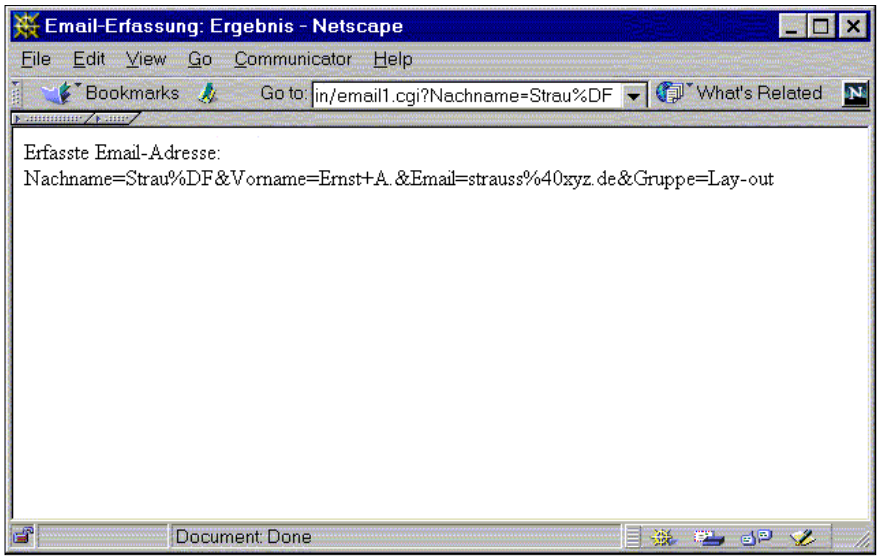
Formulare im WWW
Beispiel: Erfassung von Emailadressen



HTML-Code des Formulars zur Erfassung von Emailadressen

```
<HTML>
 <HEAD> <TITLE>Email-Erfassung</TITLE> </HEAD>
 <BODY>
  <FORM name="Formular"
        action="../cgi-bin/email1.cgi"
        method=get>
    <P>Nachname:<BR><INPUT type=text name="Nachname" size=40>
    <P>Vorname:<BR><INPUT type=text name="Vorname" size=40>
    <P>Email:<BR><INPUT type=text name="Email" size=40>
    <P>Gruppe:<BR>
      <INPUT type=radio name="Gruppe" value="Content"> Content
      <INPUT type=radio name="Gruppe" value="Lay-out"> Lay-out
      <INPUT type=radio name="Gruppe" value="Server"> Server
    <P><INPUT type=submit value="Senden">
  </FORM>
 </BODY>
</HTML>
```


Ergebnis des CGI-Scripts email1.cgi



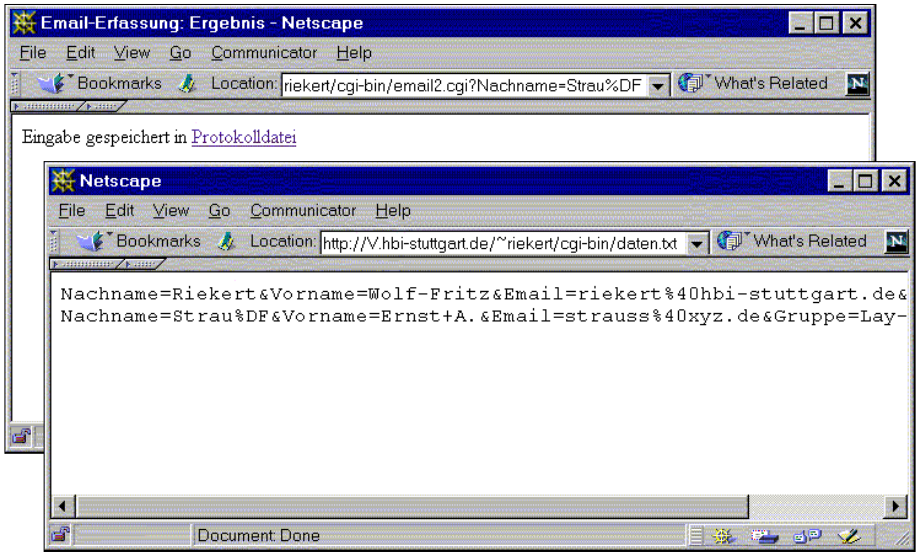
Perl-Code des CGI-Skripts email1.cgi

```
#! /usr/bin/perl

$query_string = $ENV{"QUERY_STRING"};

print "Content-type: text/html\n\n";

print "<html>\n";
print "<head><title>Email-Erfassung: Ergebnis";
print "</title></head>\n\n";

print "<body>\n";
print "Erfasste Email-Adresse:<br>\n";
print "$query_string\n";
print "</body></html>\n";
```

Variante email2.cgi: Anfügen der Emailadresse an das Ende einer Datei

```
#! /usr/bin/perl

$query_string = $ENV{"QUERY_STRING"};

open (DATEN, ">>daten.txt");
print DATEN "$query_string\n";

print "Content-type: text/html\n\n";

print "<html>\n";
print "<head><title>Email-Erfassung: Ergebnis";
print "</title></head>\n\n";

print "<body>\n";
print "Eingabe gespeichert in <A
HREF=\"../forms/daten.txt\">Protokolldatei</A>";
print "</body></html>\n";
```


Ergebnis des CGI-Skripts email2.cgi



Variante email3.cgi: Sonderzeichen und Semikolon-separiertes Textformat

```
#! /usr/bin/perl
$query_string = $ENV{"QUERY_STRING"};

%Parameter = ();
@Formularfelder = split(/&/, $query_string);
foreach $Formularfeld (@Formularfelder)  {
   ($name, $wert) = split(/=/, $Formularfeld);
   $wert =~ tr/+/ /;
   $wert =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
   $wert =~ s/<!--(.|\n)*-->//g;
   $Parameter{$name}=$wert;
   };

open (DATEN, ">>email.txt");
print DATEN "$Parameter{Nachname};$Parameter{Vorname};“
print DATEN "$Parameter{Email};$Parameter{Gruppe}\n";

print "Content-type: text/html\n\n";
                ... usw...
```


Ergebnis des CGI-Skripts email3.cgi: