LV 4752 / 7363
Web-Engineering/WBA
Übung 03, Meilenstein

Umgang mit dynamischen Webseiten:

CGI-Skripte, Formulare,

Cookies / Session management

Ziele

- Teilziele
  - Fertigstellung einer Entwicklungsumgebung für dynamische Web-Seiten.
  - Vertraut werden mit dem CGI bzw. seiner Alternative
- Übungen
  - Umgang mit Formularen
  - Umgang mit *Cookies / Sessions*
  - *Debugging: (*X)HTML-Validierung, Monitoring des HTTP-Datenverkehrs
- Übergeordnetes Ziel
  - Einübung von Grundlagen-Fertigkeiten für den späteren Einsatz von Web Services
  - Vorbereitungen für die Projektarbeit

Vorübung

- Machen Sie sich vertraut mit den bereitgestellten einfachen CGI-Skripten. Testen Sie diese!

- Anregung, freiwillig:
  - Schreiben Sie als CGI-Anwendung ein kleines C-Programm, das ein einfaches HTML-Dokument (oder auch nur ASCII-Text) ausgibt, und testen Sie seine Wirksamkeit durch Installation im "cgi-bin"-Verzeichnis. Achten Sie auf die *extension* ".cgi".

Formulare, Session-Management

- Einarbeitung
  - Den Umgang mit (X)HTML-Formularen können Sie z.B. bei SelfHTML erlernen.
- Szenario:
  - Abgabe von Hausaufgaben (Dateien).
- Beschreibung (Übersicht)
  - Teilnehmer einer LV geben Dateien als Ergebnisse von Übungen ab
  - Sie rufen dazu eine statische Seite im Intranet ab: anmeldung.xhtml
  - Diese Seite ist der Beginn einer kleinen „session“!
  - Zunächst identifiziert sich der Teilnehmer / die Teilnehmerin und kennzeichnet, zu welcher Übung eine Datei abgegeben werden soll, etc.
  - Abschicken dieses Formularinhalts leitet die Kontrolle an ein CGI-Skript, das eine *session* einleitet (verwaltet per *session id* in einem *cookie*), die Formulardaten auswertet & prüft, sie in den *session*-Daten speichert, und schließlich an eine (statische) Folgeseite weiterleitet.
  - Folgeseite = allgemeine Fehlerseite → Ende der *session*, sonst: abgabe.xhtml
  - abgabe.xhtml gestattet die Eingabe/Auswahl einer Datei (Upload) und leitet an „abgabe.cgi“ weiter. Dieses Skript speichert die Datei und beendet die *session*.

Formulare, Session-Management

- Anmeldung
  - Erstellen Sie ein (statisches) HTML-Formular **`anmeldung.xhtml`**
  - Eingabefelder:
    - Name, MatrNr, E-Mail (text)
    - LV-Nr (selection, Text anzeigen, aber LV-Nr. übertragen)
      - 4752 Web-Engineering
      - 7363 Web-basierte Abwendungen
    - Aufgabe-Nr (selection, Voreinstellung nach aktuellem Datum!)
      - 01 ... 14 Entsprechen den Kalenderwochen 42..51/2008 und 2..5/2009
      - Beispiel: Wenn Aufruf am 5.11.2008 (KW45), sollte Aufgabe 04 voreingestellt sein
      - Bem.: Dynam. Voreinstellung erfordert spezielle Techniken (hier nicht gefordert).
    - *Radio Button*-Gruppe „Art der Leistung“ (radio):
      - „PL“ oder „SL“
    - Checkbox-Gruppe „Hinweise“ (checkbox):
      - „ZIP-Archiv“, „Enthält ausführbare Dateien“, „Enthält Multimediadaten“
    - „Anmeldung"-Knopf (submit)
      - Folgende Anwendung soll aufgerufen werden: anmelden.cgi

Formulare, Session-Management

- Anmeldung (Forts.)
  - **`anmelden.cgi`** Schreiben Sie diese Anwendung!
    - Die übertragenen Formulardaten sollen ausgewertet werden. Speichern Sie die Angaben unter einer *Session ID* server-seitig persistent!
    - *Session Management* per *Cookie* (*Session ID* darin speichern).
    - Fehlerfall: "Weiterleitung" an eine statische Seite **`abgabefehler.xhtml`**
    - Normalfall: Weiterleitung an Formular **`abgabe.xhtml`**
  - Fehlerfälle:
    - Pflichtangaben fehlen (alles außer „Hinweise“ sind Pflichtfelder)
    - Ungültige Matrikelnummer, unplausible E-Mail-Adresse

Formulare (Forts.)

- Abgabeteil
  - Die Seite **`abgabe.xhtml`** soll zur Auswahl einer lokalen Datei auffordern und deren Übertragung an **`abgeben.cgi`** einleiten.
  - Die Anwendung **`abgeben.cgi`** soll
    - die Verwaltung der *session* fortsetzen (*Cookie* lesen, Eingabedaten wiederherstellen)
    - die eingehenden Daten annehmen und unter **`matnr/aufgXX`** in einem geeigneten Verzeichnis speichern.
    - im Fehlerfall (insb. "kein *Cookie*") per interner Weiterleitung an eine statische Fehlerseite **`nicht_angemeldet.xhtml`** weiterleiten.
- Hinweise, Vorgaben
  - Alle statischen Seiten müssen ***gültige* XHTML 1.x-Dokumente** sein.
  - Nutzen Sie die Ruby-Standardbibliotheken „CGI“ und „CGI::Session“!
    - Online-Dokumentation verwenden
  - Empfehlung:
    - „error.log“ Ihres Web-Servers häufig konsultieren!

Abgabe

- Statische Seiten:
  - **`anmeldung.xhtml, abgabe.xhtml, abgabefehler.xhtml, nicht_angemeldet.xhtml`**
  - Die statischen Seiten müssen gültigen XHTML 1.0- oder 1.1-Code enthalten (→ validator.w3.org, Validierung ist Teil der Abnahme)
  - Wunsch: XHTML 1.1; XHTML 1.0 „notfalls“ akzeptabel

- CGI-Skripte:
  - **`anmelden.cgi, abgeben.cgi`**
  - Implementierung in Ruby, unter Verwendung von „cgi“ und „cgi/session“
  - Ihre Abgaben sollen CGI-Technik demonstrieren. Implementierung mit FastCGI ist erlaubt (nur für besonders ambitionierten Gruppen).

Abgabe

- Wann
  - Erste Hälfte muss zu Beginn des nächsten Praktikums vorführbar sein
  - Abgabe aller Ergebnisse vor Beginn des übernächsten Praktikums
- Wie
  - Abgabe der Dateien vorab einmal pro Gruppe;
  - Die Abnahme erfolgt später während des Praktikums
    - Bitte beachten: Incl. XHTML-Validierung und Nutzung von TCPmon (s.u.)
  - Jedes Team-Mitglied sollte jeden Aufgabenteil beherrschen!
- Anregung (freiwillig, ohne Wertung):
  - Performance-Vergleich zwischen Apache und Lighttpd
    - Kooperation zwischen einem Apache- und einem Lighttpd-Team?
    - Nutzung der Tests „cgi vs. fcgi“, Übertragung auf die Web-Server
    - Abrufen mit „ab“

`tcpmon`:
Ein Werkzeug zum HTTP-Debugging

Bitte verwenden – während der Abnahme
ist tcpmon erforderlich…

Nutzung von tcpmon

- Quelle
  - $srcdir/03/axis.jar.

- Installation
  - Kopieren Sie "axis.jar" in Ihren Bereich oder nutzen Sie direkt die Datei im Dozentenverzeichnis.
  - Aufruf: Am besten per "alias" in ~/.bashrc, nach folgendem Muster:

```
alias tcpmon=
  'java -cp /path/to/axis.jar org/apache/axis/utils/tcpmon'
```

  - Alternativ auf der Kommandozeile, mit optionalen Argumenten:

```
java -cp /path/to/axis.jar org/apache/axis/utils/tcpmon
  [listenPort targetHost targetPort]
```

- Konfiguration
  - Entfällt. Alles was Sie benötigen ist ein 2. Port (siehe Funktionsweise)

Nutzung von tcpmon

- Funktionsweise
  - **`tcpmon`** arbeitet auf der TPC/IP-Ebene (daher der Name)
  - Das Tool wird zwischen HTTP-Client und -Server geschaltet:
    - Der Client baut eine Verbindung zu **`tcpmon`** statt zum Server auf
    - **`tcpmon`** reicht diese Daten an den Server weiter und umgekehrt.
    - Die durchgereichten Daten zeigt **`tcpmon`** an!
  - Wählen Sie dazu einen freien Port
    - Teilen Sie dem Client mit, den Server auf diesem Port zu erwarten.
    - Konfigurieren Sie **`tcpmon`** so, dass das Tool auf diesem Port lauscht.
    - Teilen Sie **`tcpmon`** den wahren Server-Port für die Weiterleitung mit.
- Nutzen
  - Details der HTTP *header* und die *message*-Teile werden sichtbar
  - Auf HTTP beruhende höhere Protokolle wie XML-RPC und SOAP verdecken die auf der HTTP-Ebene ausgetauschten Details.
  - Im Fehlerfall benötigt man Werkzeuge wie **`tcpmon`**. Es hilft z.B. sehr, die genauen Daten zu erfahren, die ein Web-Server erhält.

Nutzung von tcpmon

- Alternativen?
  - FAQ 1: Warum nicht **`ethereal/wireshark`** nehmen?
  - A: Dieses Werkzeug zeigt (a) zu viel an und (b) unterstützt XML/XHTML nicht, d.h. protokollierte Daten sind viel schwerer lesbar als mit **`tcpmon`**.

- Test
  - Verwenden Sie **`tcpmon`**, um den Datenverkehr zwischen Client und Server Ihrer CGI-Anwendung mitzulesen.
    - Entsprechen die Client-Daten genau Ihren Erwartungen?
    - Antwortet der Server ebenfalls wie erwartet?
    - Finden Sie Erklärungen für eventuelle Abweichungen!
  - Führen Sie diese Analyse auch aus, wenn keine Probleme mit CGI auftreten. Ziel ist die sichere Beherrschung des Werkzeugs für Debuuging-Zwecke im Laufe der Projektaufgaben!