



# 7363 - Web-basierte Anwendungen: **Übung 04, Meilenstein 2**

Umgang mit dynamischen Webseiten:

FCGI-Skripte

*Server Side Includes (SSI),*

*Cookies / Session management*



- Teilziele
  - Fertigstellung einer Entwicklungsumgebung für dynamische Web-Seiten.
  - Vertraut werden mit dem CGI bzw. seiner Alternative
  
- Übungen
  - Performance-Vergleich: CGI vs. Alternativen (FastCGI)
  - Umgang mit Formularen
  - Umgang mit SSI und *Cookies*
  
- Übergeordnetes Ziel
  - Einübung von Grundlagen-Fertigkeiten für den späteren Einsatz von Web Services
  - Vorbereitungen für die Projektarbeit



- Machen Sie sich vertraut mit den bereitgestellten einfachen CGI- und FastCGI-Skripten. Testen Sie diese!
- Wer PHP oder Java Servlets einsetzt, sollte sich statt mit FastCGI mit Beispielen dieser Machart vertraut machen.
- 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".



- SSI
  - Erstellen Sie eine SHTML-Seite `besucherzaehler.shtml` mit einem SSI-Element, das einen Besucherzähler realisiert (vgl. Vorlesung).
  - Entwickeln Sie eine CGI-Anwendung `zaehler.cgi` dazu, die den Zählerwert liefert!
  - Hätte der Weg über FastCGI Vorteile?
- *Cookies*
  - Generieren Sie eine kleine HTML-Seite dynamisch per (F)CGI-Skript `cookiezaehler.cgi` (bzw. fcgi)
  - Realisieren Sie darauf einen Besuchszähler
    - Vergeben Sie *Cookies* an neue Besucher
    - Lesen Sie den *Cookie* von erneuten Besuchern (incl. eines Zählerwertes), geben Sie den Zählerwert aus, inkrementieren Sie den Zähler im *Cookie*.



# B: Formulare, Session-Management

---



- Einarbeitung
  - Den Umgang mit (X)HTML-Formularen können Sie z.B. bei SelfHTML erlernen.
- Szenario:
  - Abgabe von Hausaufgaben (Dateien).
- Anmeldung
  - Erstellen Sie ein (statisches) HTML-Formular **anmeldung.xhtml**
    - Eingabefelder: Name, MatrNr, E-Mail, LV-Nr, Aufgabe-Nr (01 ... 15)
    - *Radio Button*-Gruppe: PL oder SL
    - „Anmeldung“-Knopf soll folgende Anwendung aufrufen:
  - **anmelden.fcgi** Schreiben Sie diese Anwendung!
    - Das Formular soll ausgelesen 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**



- Abgabeteil
  - Die Seite `abgabe.xhtml` soll zur Auswahl einer lokalen Datei auffordern und deren Übertragung an `abgabe.fcgi` einleiten.
  - Die Anwendung `abgabe.fcgi` 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 1.1-Dokumente sein.
  - Nutzen Sie ggf. die APIs Ihrer Entwicklungsumgebung für den Umgang mit CGI bzw. FastCGI und für Session-Management.
  - Empfehlung:
    - Erst CGI-Version, bei Funktionieren erst Umstellung auf FCGI
    - „error.log“ Ihres Web-Servers häufig konsultieren!



- Was bringt FastCGI?
  - Erstellen Sie eine (einfache) statische HTML-Seite `perftest.xhtml`
  - Erstellen Sie ein CGI-Programm `perftest.cgi`, das bei Aufruf die statische Seite liest und ausgibt. Wahlweise können Sie eine gleichartige Seite auch gleich dynamisch generieren.
  - Erstellen Sie ein FastCGI-Programm `perftest.fcgi` mit gleichartigem Verhalten.
  - Verwenden Sie das Testprogramm "**ab**" aus dem „bin“-Ordner des Apache, um die o.g. drei alternativen Dokumentenquellen per HTTP  $n$ -mal abzurufen und die dafür benötigten Zeiten zu ermitteln.
  - Stellen Sie fest, wie viele Sekunden die drei Varianten für dieselbe Anzahl Abrufe benötigen. Wählen Sie  $n$  so, dass die schnellste Variante ca. 2 Sekunden benötigt.
  - Beantworten Sie nun: Lohnt sich FastCGI gegenüber CGI?
  - Was ist bei höherem Initialisierungsaufwand zu erwarten (z.B. umfangreiche Interpreter-Arbeit zu Beginn, Datenbank-Anmeldung)



- Was
  - Zweier-Teams:  
    Bearbeiten Sie mindestens 2 der drei Aufgaben
    - A SSI & Cookies
    - B **Formulare**
    - C Performance-Test cgi vs. fcgi. Messergebnisse festhalten (Kommentare im Code?)
      - Dabei sollte auf jeden Fall die Formularaufgabe sein!
  - Dreier-Teams:  
    Bearbeiten Sie alle 3 Aufgabenteile (A-C)
  - Bemerkungen
    - Ihre Abgaben sollen mit FastCGI funktionieren! CGI = Vorübung.
    - Die statischen Seiten müssen gültigen XHTML 1.1-Code enthalten (→ [validator.w3.org](http://validator.w3.org))





- Wann
  - Abnahmefähig zu Beginn des nächsten Praktikums
- Wie
  - Einmal pro Gruppe; Die Abnahme erfolgt während des Praktikums
  - Jedes Team-Mitglied sollte jeden Aufgabenteil beherrschen!  
(Los-Entscheid!)
- 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