Datenbank-basierte Webserver

Struktur eines Datenbank-basierten Webservers

Karl Riedling
Institut für Sensor- und Aktuatorsysteme

Struktur eines Datenbank-basierten Webservers

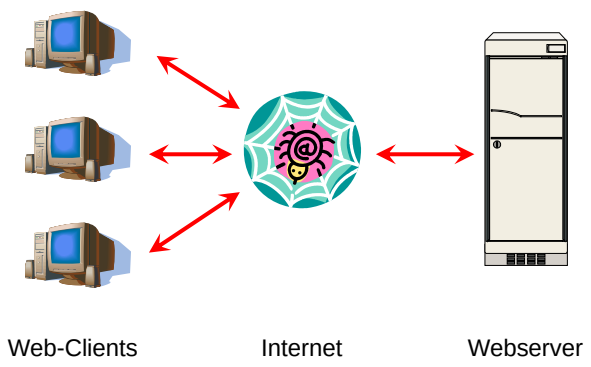
- Konventioneller ↔ Datenbank-basierter Webserver
- Das TCP/IP-Protokoll
- Erstellung einer Datenbank-basierten Webseite
- Moderne Web-Technologien am Beispiel von *Ajax*
- Erforderliche Komponenten eines Datenbank-basierten Webservers
- Die LAMP-Systemstruktur

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Struktur eines Datenbank-basierten Webservers

- **Konventioneller ↔ Datenbank-basierter Webserver**
- Das TCP/IP-Protokoll
- Erstellung einer Datenbank-basierten Webseite
- Moderne Web-Technologien am Beispiel von *Ajax*
- Erforderliche Komponenten eines Datenbank-basierten Webservers
- Die LAMP-Systemstruktur

Konventioneller Webserver

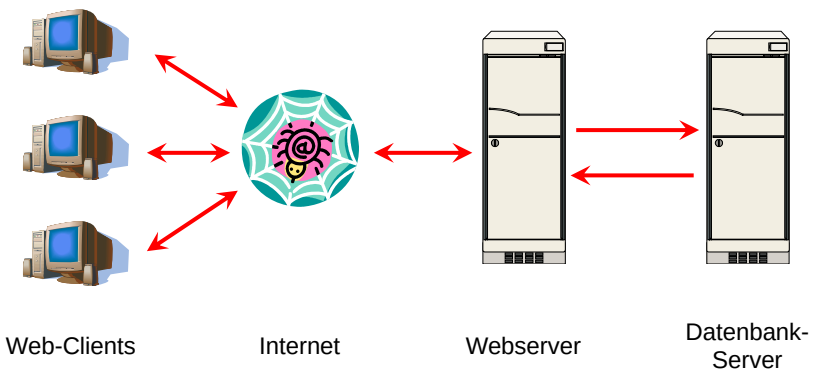


© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Konventioneller Webserver

- Webserver stellt den Clients statische Dateien (HTML-Dateien, Bild-Dateien, usw.), die auf dem Webserver (oder einem für ihn zugänglichen Netzwerkslaufwerk) liegen, auf Anforderung zur Verfügung
- Änderung des Inhalts der Website erfordert Änderung der statischen Dateien

Datenbank-basierte Webserver



© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Datenbank-basierte Webserver

- Webserver stellt den Clients statische Dateien *und* dynamisch generierte Daten zur Verfügung
- Anfragen der Clients nach bestimmten Informationen werden an den Datenbank-Server weitergeleitet
- Aus Antworten des Datenbank-Servers werden dynamische Seiten aufgebaut und an den anfragenden Client geschickt

Datenbank-basierte Webserver

- Webserver und Datenbank-Server können verschiedene physikalische Maschinen oder aber Prozesse sein, die auf der gleichen Maschine laufen
- Abfragen von *mehreren* (physikalischen oder logischen) Datenbank-Servern sind möglich

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Struktur eines Datenbank-basierten Webservers

- Konventioneller ↔ Datenbank-basierter Webserver
- **Das TCP/IP-Protokoll**
- Erstellung einer Datenbank-basierten Webseite
- Moderne Web-Technologien am Beispiel von *Ajax*
- Erforderliche Komponenten eines Datenbank-basierten Webservers
- Die LAMP-Systemstruktur

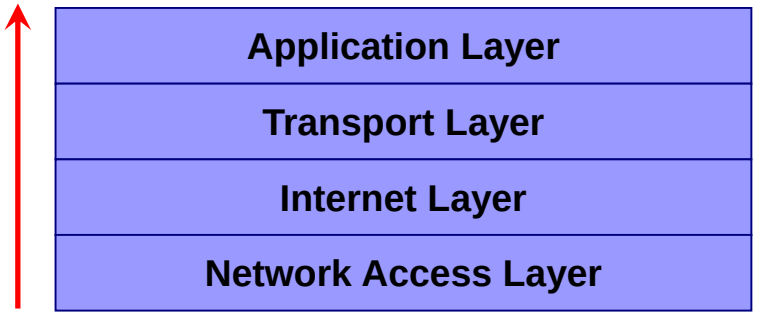
Grundlagen des TCP/IP-Protokolls

- TCP = *Transmission Control Protocol*
- IP = *Internet Protocol*
- TCP/IP-*Stack* bestehend aus vier *Layers*:
  - *Application Layer*
  - *Transport Layer*
  - *Internet Layer*
  - *Network Access Layer*

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Grundlagen des TCP/IP-Protokolls

- TCP/IP Stack:



Grundlagen des TCP/IP-Protokolls

- Network Access Layer:
  - Zugriff auf das physikalische Netzwerk; eigentlicher Datentransport
  - Transport erfolgt von Gerät zu Gerät:
    Client-Rechner (– Switch – …) – Router (– Switch – …) – Router – …– Server
  - Die einzelnen Teilstrecken sind häufig mit unterschiedlichen Technologien realisiert (Kupfer, Glasfaser, drahtlos…)
  - Protokolle z.B.: Ethernet, Token Ring

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Grundlagen des TCP/IP-Protokolls

- Internet Layer:
  - Verantwortlich für das Routen von Paketen
  - Transport erfolgt von Router zu Router; Router bestimmen den Weg, den die Datenpakete nehmen:
  - Client-Rechner – Router – Router – …– Router – Server
  - Weg der Daten kann mit Programm `traceroute` nachvollzogen werden
  - Protokoll: IP

Grundlagen des TCP/IP-Protokolls

- Transport Layer:
  - Ermöglicht „End-to-end“-Datenverbindungen
  - Details der Datenverbindung werden verborgen; Verbindungen werden unabhängig von der Anzahl und Art involvierter Zwischenschritte definiert (durch Angabe der *Adresse* des Ziel-Systems)
  - Protokolle z.B.: TCP, UDP

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Grundlagen des TCP/IP-Protokolls

- Application Layer:
  - Beliebig viele Anwendungen können die gleiche Netzwerksverbindung benutzen (Browser, Mail-Client, FTP-Client, …)
  - Zwischen einzelnen Applikationen wird durch eine *Port*-Nummer unterschieden
  - Protokolle z.B.: SSH, FTP, HTTP, SMTP, …

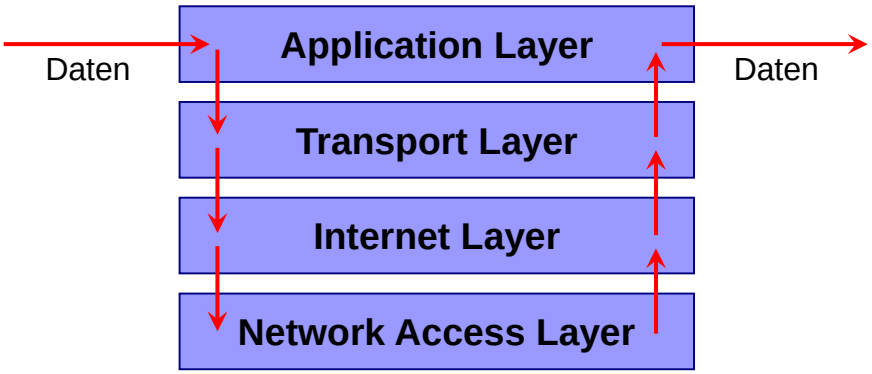
Grundlagen des TCP/IP-Protokolls

- Einkapselung (*Encapsulation*):
  - Senden von Daten: Das Datenfeld einer oberen Ebene wird mit zusätzlichen Informationen ergänzt und ergibt ein Datenfeld der jeweils darunter liegenden Ebene
  - Empfang von Daten: Die Interpretation des Datenfelds einer unteren Ebene hängt vom verwendeten Protokoll ab; das Datenfeld und die zusätzlichen Informationen der oberen Ebene können damit rekonstruiert werden

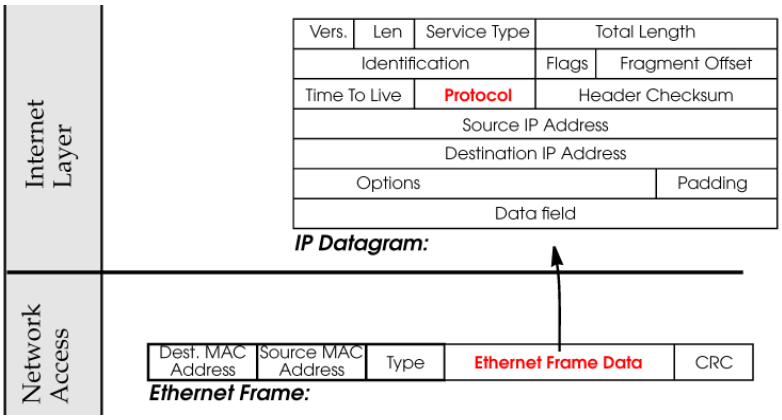
© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Grundlagen des TCP/IP-Protokolls

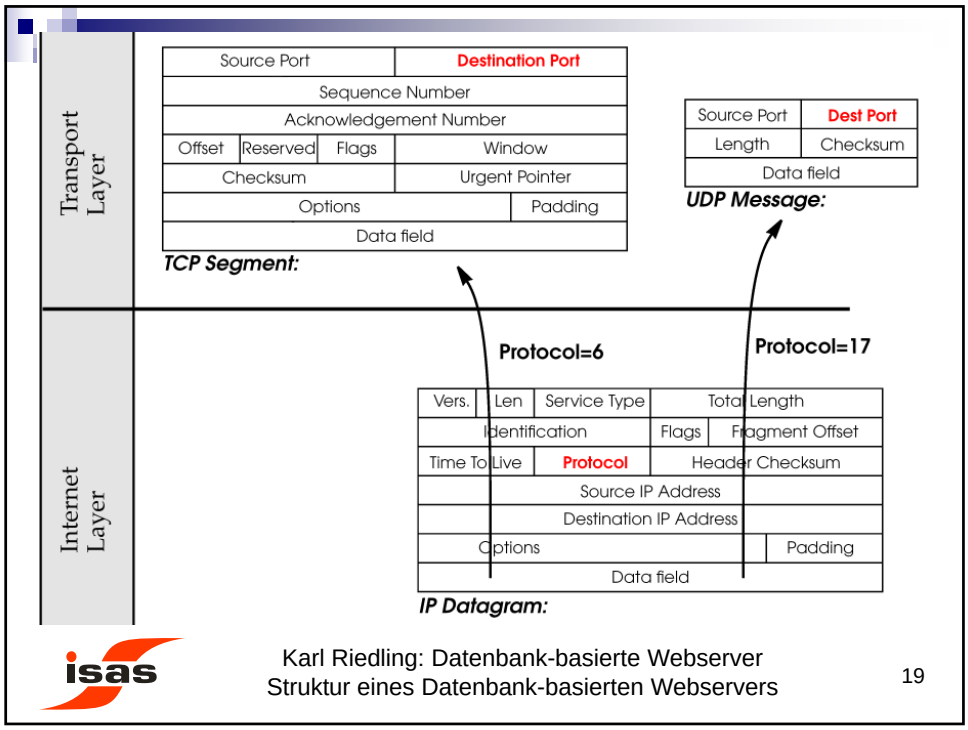
- Einkapselung :



Grundlagen des TCP/IP-Protokolls



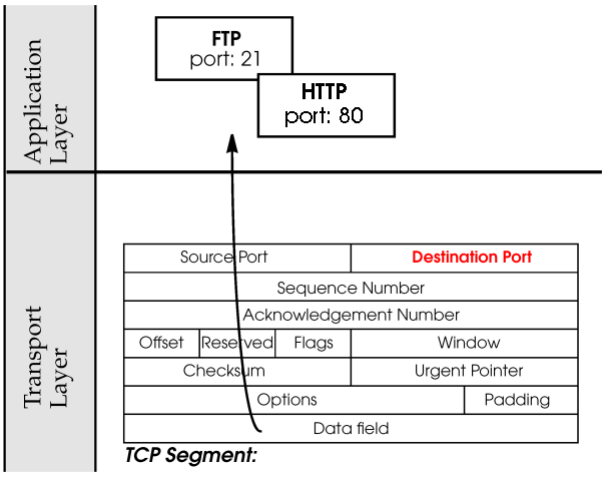
© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme



Karl Riedling: Datenbank-basierte Webserver
Struktur eines Datenbank-basierten Webservers

19

Grundlagen des TCP/IP-Protokolls



Karl Riedling: Datenbank-basierte Webserver
Struktur eines Datenbank-basierten Webservers

20

IP-Adressierung

- IP-Adresse: Numerische Adresse für jeden Internet-Host:
  - IPv4: 32-Bit-Adressen (gegenwärtiger Standard)
  - IPv6: 128-bit-Adressen
- Schreibweisen (IPv4):
  - Vier dezimale Zahlen („*dot notation*“), z.B.: `128.130.35.76`
  - Hexadezimal, z.B.: `0x8082234C`
  - Zulässig: *ein* Dezimalwert, z.B.: `2156012364`

IP-Adressierung

- Domain Name Service (DNS): ermittelt die numerischen *IP-Adressen* aus *IP-Namen*, z.B.: `www.tuwien.ac.at`

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Ports und Sockets

- *Port*: Numerischer Wert (16 Bit), der mit einer Anwendung der *Application Layer* verknüpft ist
- Beispiele (Standard-Ports):
  - 21 – FTP (*File Transfer Protocol*)
  - 22 – SSH (*Secure Shell*-Login)
  - 25 – SMTP (*Simple Mail Transfer Protocol*)
  - 80 – HTTP (*Hypertext Transmission Protocol*)
  - 443 – HTTPS (Secure HTTP)

Ports und Sockets

- Fix zugeordnete Port-Nummern für Server („*privileged*“): 0 … 1023
- Frei verfügbar („*unprivileged*“):
  - 1024 … 4999 für Clients
  - 5000 … 32767 für Server
  - 32768 … 65535 für Clients

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Ports und Sockets

- *Socket*: vollständige Start- oder Zieladresse einer TCP/IP-Verbindung
- Besteht aus IP-Adresse plus Port-Nummer, z.B.: `128.130.35.76:80` (Zieladresse für HTTP-Anfragen an den Server `www.tuwien.ac.at`)

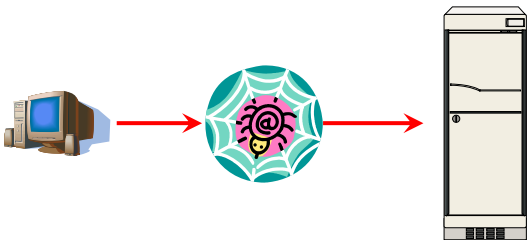
Struktur eines Datenbank-basierten Webservers

- Konventioneller ↔ Datenbank-basierter Webserver
- Das TCP/IP-Protokoll
- **Erstellung einer Datenbank-basierten Webseite**
- Moderne Web-Technologien am Beispiel von *Ajax*
- Erforderliche Komponenten eines Datenbank-basierten Webservers
- Die LAMP-Systemstruktur

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

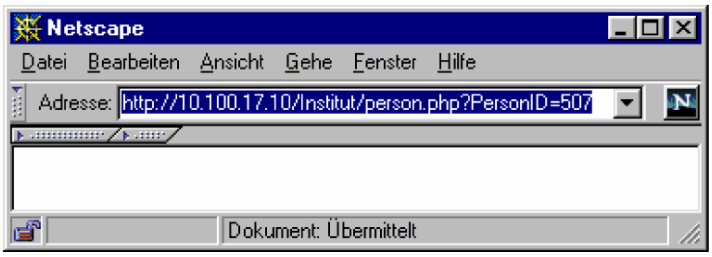
Erstellung einer Datenbank-basierten Webseite

- Anfrage eines Clients an den Webserver (Protokoll: HTTP oder HTTPS)



Erstellung einer Datenbank-basierten Webseite

- Anfrage eines Clients an den Webserver

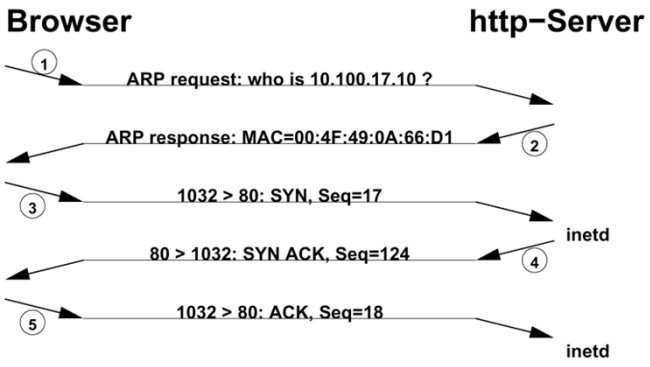


(Ablaufdiagramme © Harald Kleiner)

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Erstellung einer Datenbank-basierten Webseite

- Anfrage eines Clients an den Webserver, 1. Teil



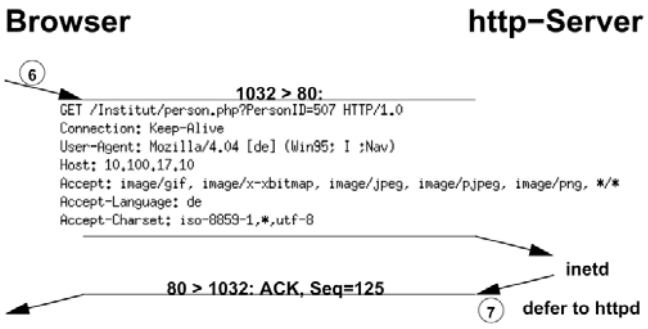
Erstellung einer Datenbank-basierten Webseite

- ARP – *Address Resolution Protocol*
- MAC – *Medium Access Control*
- SYN – *Synchronize*
- ACK – *Acknowledge*

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

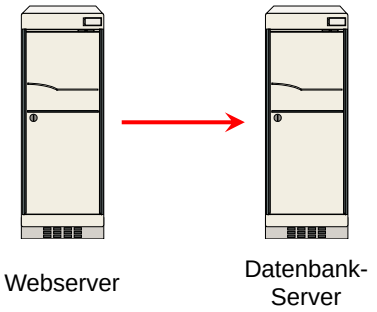
Erstellung einer Datenbank-basierten Webseite

- Anfrage eines Clients an den Webserver, 2. Teil



Erstellung einer Datenbank-basierten Webseite

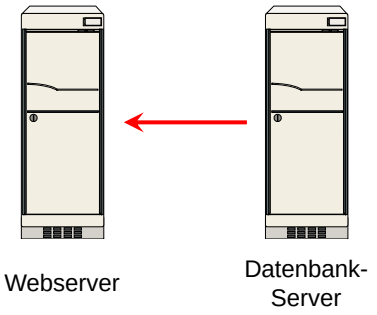
- Anfrage des Webservers an den Datenbankserver (lokal oder TCP/IP)



© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Erstellung einer Datenbank-basierten Webseite

- Antwort des Datenbankservers an den Webserver (lokal oder TCP/IP)



Erstellung einer Datenbank-basierten Webseite

- Kommunikation Webserver – Datenbankserver

/usr/local/httpd/htdocs/Institut/person.php - Texteditor

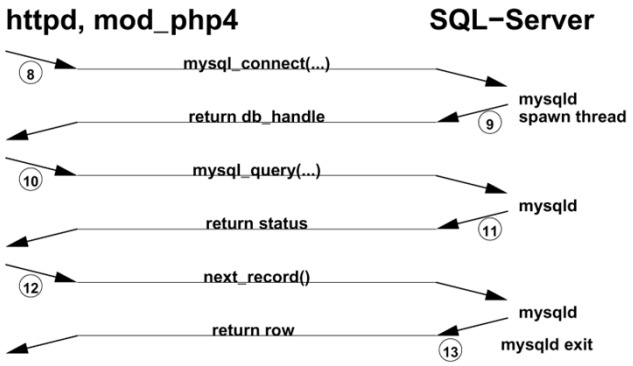
```
<?php
 $db = new DB_Institut;
 $db->query("SELECT * FROM Personen WHERE PersonID='" . addslashes($PersonID) . "';" );
 $db->next_record();
 echo '<html><body>' . $db->f('Vorname') . ' ' . $db->f('Nachname') . '</body></html>';
?>
```

(Ablaufdiagramme © Harald Kleiner)

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

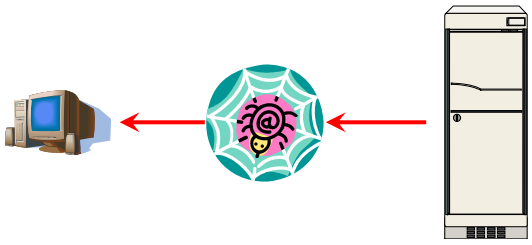
Erstellung einer Datenbank-basierten Webseite

- Kommunikation Webserver – Datenbankserver



Erstellung einer Datenbank-basierten Webseite

- Antwort des Webservers an den Client (Protokoll: HTTP oder HTTPS)

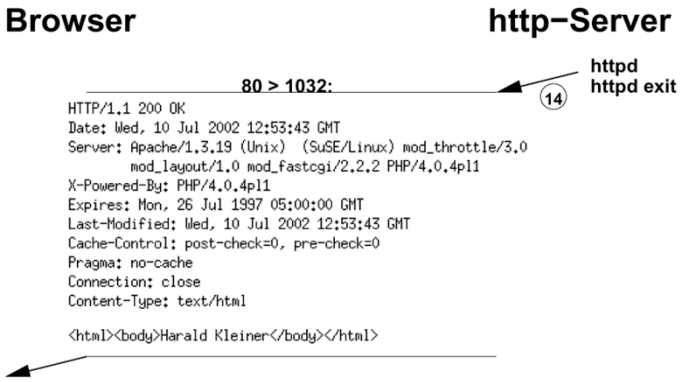


Web-Client Internet Webserver

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

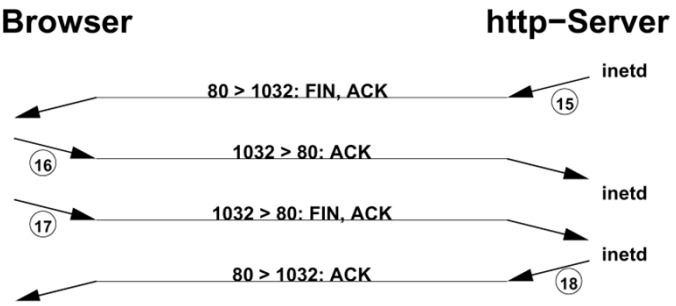
Erstellung einer Datenbank-basierten Webseite

- Antwort des Webservers an den Client, 1. Teil



Erstellung einer Datenbank-basierten Webseite

- Antwort des Webservers an den Client, 2. Teil



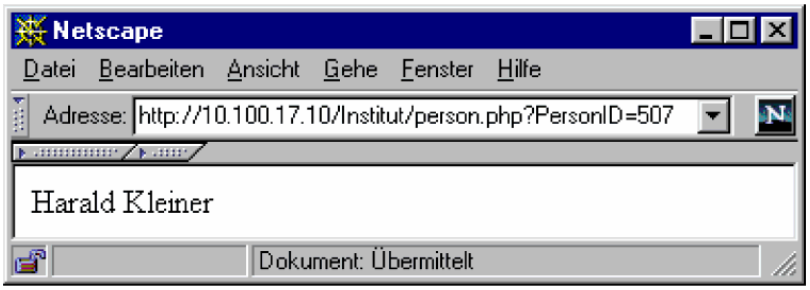
© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Erstellung einer Datenbank-basierten Webseite

- FIN – *Finished*
- ACK – *Acknowledge*

Erstellung einer Datenbank-basierten Webseite

- Antwort des Webservers an den Client



(Ablaufdiagramme © Harald Kleiner)

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Zustandslosigkeit des HTTP-Protokolls

- Für jede Anfrage an den Webserver wird eine neue TCP/IP-Verbindung aufgebaut
- Aufeinanderfolgende Anfragen sind daher für den Webserver nicht als zusammengehörig zu erkennen
- Die IP-Adresse oder der *Socket* reichen *nicht* zu einer eindeutigen Identifikation bei der nächsten Anfrage (z.B. bei NAT – *Network Address Translation*)!

Struktur eines Datenbank-basierten Webservers

- Konventioneller ↔ Datenbank-basierter Webserver
- Das TCP/IP-Protokoll
- Erstellung einer Datenbank-basierten Webseite
- **Moderne Web-Technologien am Beispiel von *Ajax***
- Erforderliche Komponenten eines Datenbank-basierten Webservers
- Die LAMP-Systemstruktur

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Moderne Web-Technologien am Beispiel von *Ajax*

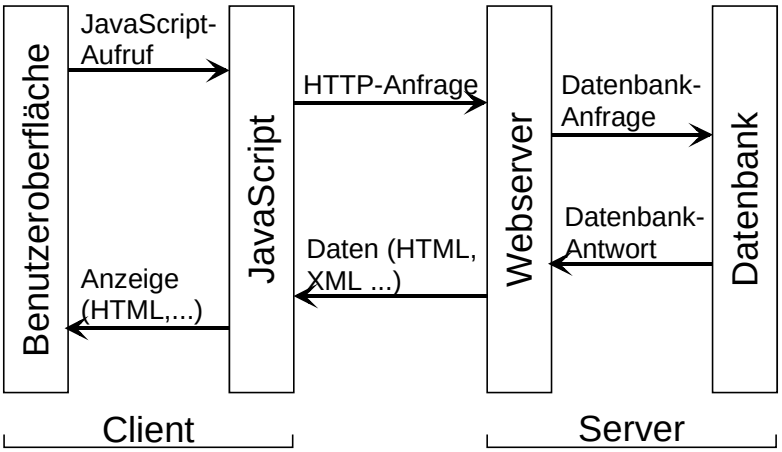
- „Klassischer“ Ablauf (wie im vorigen Beispiel):
  - Client schickt eine Anfrage
  - Server schickt eine (komplette) Seite
- Vollständiger Neu-Aufbau einer Seite ist aber oft suboptimal:
  - Großer Datenverkehr und längere Ladezeiten
  - Orientierung auf der Seite für die User schwieriger
- Besser wäre lokale Änderung einzelner Teile einer Seite (ähnlich wie bei Desktop-Applikationen)

Moderne Web-Technologien am Beispiel von *Ajax*

- *Ajax* – **A**synchronous **J**avaScript **a**nd **X**ML: Asynchrone Datenübertragung zwischen Server und Client
- Beim bestimmten User-Aktionen wird ähnlich wie im vorigen Beispiel eine Anfrage an den Server geschickt und vom Server Inhalt generiert, der aber *innerhalb* der bestehenden Seite angezeigt wird
- Statt des „klassischen“ *Pull*-Prinzip des Webs (Client holt sich Informationen vom Server) ist damit auch ein *Push*-Verhalten möglich (Server schickt dann Informationen, wenn er sie hat)

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Moderne Web-Technologien am Beispiel von *Ajax*



Struktur eines Datenbank-basierten Webservers

- Konventioneller ↔ Datenbank-basierter Webserver
- Das TCP/IP-Protokoll
- Erstellung einer Datenbank-basierten Webseite
- Moderne Web-Technologien am Beispiel von *Ajax*
- **Erforderliche Komponenten eines Datenbank-basierten Webservers**
- Die LAMP-Systemstruktur

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Standard-Aufgaben eines Webservers

- Entgegennahme von Anfragen der Clients nach einem Dokument (HTML-Seite, Bild-Datei, usw.)
- Bereitstellung dieses Dokuments für den Client

Zusätzliche Aufgaben eines Datenbank-basierten Webservers

- Entgegennahme *zusätzlicher Parameter* vom Client (z.B. Eingaben des Benutzers in ein Web-Formular)
- Umsetzung dieser Parameter in Anfragen an den Datenbank-Server
- Entgegennahme der Antwort des Datenbank-Servers
- Umsetzung dieser Antwort in eine Ausgabe an den Client

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Erforderliche Komponenten eines Datenbank-basierten Webservers

- Betriebssystem
- Webserver (-Prozess)
- Datenbank-Server (-Prozess)
- Geeignetes Bindeglied (= Programm) zwischen Webserver und Datenbank-Server
- Allenfalls Client-seitige Software

Erforderliche Komponenten eines Datenbank-basierten Webservers

- **Betriebssystem**
- Webserver (-Prozess)
- Datenbank-Server (-Prozess)
- Geeignetes Bindeglied (= Programm) zwischen Webserver und Datenbank-Server
- Allenfalls Client-seitige Software

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Anforderungen an ein Betriebssystem

- Stabile, leistungsfähige und sichere Umgebung für Webserver- und Datenbank-Prozess sowie die Programme zur Integration der beiden Prozesse
- Graphische Benutzeroberfläche ist nicht unbedingt erforderlich

Gängige Server-Betriebssysteme

- Windows
  - Weit verbreitet
  - Nur Server-Versionen für Produktionssysteme stabil genug
  - Vom Konzept her nur *Microsoft Internet Information Server* (IIS) als Webserver vorgesehen; andere Webserver ggf. mit Einschränkungen
  - IIS ist berüchtigt für Kompatibilitätsprobleme!
  - Ressourcenfresser
  - Server-Versionen sind relativ teuer

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Gängige Server-Betriebssysteme

- UNIX
  - In vielen unterschiedlichen Varianten verfügbar
  - Speziell auf höchste Leistungsfähigkeit hin ausgelegt
  - Wesentlich Ressourcen-schonender als Windows
  - Große Auswahl für alle erforderlichen Software-Komponenten (Webserver, Datenbankserver, Programmiersprache) verfügbar
  - Teuer und gelegentlich auf teure Spezial-Hardware beschränkt

Gängige Server-Betriebssysteme

- LINUX
  - \+ Hat alle positiven Eigenschaften der kommerziellen UNIX-Varianten
  - \+ Ist (zumindest theoretisch) kostenlos
  - \+ Im Prinzip auf jedem PC lauffähig – billige Hardware
  - \- Im Höchstleistungsbereich den kommerziellen UNIX-Varianten eventuell leistungsmäßig unterlegen

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Erforderliche Komponenten eines Datenbank-basierten Webservers

- Betriebssystem
- **Webserver (-Prozess)**
- Datenbank-Server (-Prozess) (auf der gleichen oder einer anderen Maschine)
- Geeignetes Bindeglied (= Programm) zwischen Webserver und Datenbank-Server
- Allenfalls Client-seitige Software

Anforderungen an die Webserver-Software

- Leistungsfähigkeit
- Stabilität
- Sicherheit gegen (Hacker-) Angriffe
- Mit der Server-seitigen Software zur Kommunikation mit dem Datenbank-Server kompatibel und möglichst integrierbar

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Webserver-Software

- Microsoft Internet Information Server (IIS)
  - + Mit Windows als Betriebssystem integriert
  - - Mit potenziellen Kompatibilitätsproblemen behaftet

Webserver-Software

- Apache
  - + *Open Source*-Software; kostenlos
  - + Für praktisch alle Betriebssystems-Umgebungen verfügbar
  - + Standard-Webserver für UNIX/Linux-Systeme
  - + Mit einer Reihe von Scriptsprachen kompatibel oder integrierbar
  - - Wegen seiner großen Verbreitung häufiges Ziel von Hacker-Attacken

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Webserver-Software

- NGINX
  - + *Open Source*-Software; kostenlos
  - + Signifikant (Faktor 2,5) schneller als Apache für statische Inhalte
  - + Die meisten Features von Apache verfügbar
  - + Häufig verwendet in Kombination mit Apache als Proxy, für *Load Balancing* und Authentisierung
  - - Nur für Unix/Linux-Umgebung

Erforderliche Komponenten eines Datenbank-basierten Webservers

- Betriebssystem
- Webserver (-Prozess)
- **Datenbank-Server (-Prozess)**
- Geeignetes Bindeglied (= Programm) zwischen Webserver und Datenbank-Server
- Allenfalls Client-seitige Software

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Anforderungen an die Datenbank-Server-Software

- In der Regel: *Relationale* Datenbank
  - Beliebig komplexe Zusammenhänge zwischen Informationen sind darstellbar
- Datenbank auf SQL- (*Structured Query Language*) Basis
  - Datenaustausch mit anderen Datenbanken wird erleichtert

Anforderungen an die Datenbank-Server-Software

- Objektorientierte Datenbanken
  - Basieren meist auf relationalen Datenbanken
  - Können Vorteile bei der Erstellung der Server-Software bieten
  - Gleiche Funktionalität ist aber auch durch objektorientierte Features der Schnittstellen-Software zwischen Webserver und Datenbank zu erreichen
  - Objektorientierung ist meist mit höherem Ressourcen-Bedarf verbunden – Performance im Höchstleistungs-Bereich!

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Anforderungen an die Datenbank-Server-Software

- Speziell im Web-Bereich vielfach Geschwindigkeit wichtiger als aufwändige Maßnahmen zur Sicherung der Datenintegrität
- Features wie *Transaktionen* sind oft nicht erforderlich (speziell bei Datenbanken, bei denen keine komplexen und/oder finanziell relevanten Dateneingaben von den Clients erfolgen)

Gängige Datenbank-Programme

- Oracle
  - \+ Extrem leistungsfähig
  - \+ Standard im professionellen Datenbank-Bereich
  - \- Wegen seiner vielen Features eher schwerfällig
  - \- Keine Notwendigkeit für zahlreiche seiner Features
  - \- Hoher Ressourcen-Bedarf
  - \- Kommerzielle Lizenzen extrem teuer

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Gängige Datenbank-Programme

- Microsoft Access
  - + Standard in einer Windows-Umgebung
  - - Primär Desktop-Applikation; kaum mit Webserver-Prozess integrierbar
  - - Nur unter Windows lauffähig
  - - Langsam und schwerfällig
  - - Hoher Ressourcen-Bedarf
  - - Mangelhafte Kompatibilität zwischen unterschiedlichen Software-Versionen

Gängige Datenbank-Programme

- Microsoft SQL Server
  - + Standard in einer Windows-Server-Umgebung
  - - Nur unter Windows lauffähig
  - - Hoher Ressourcen-Bedarf
  - - gebündelt mit Windows-Server – sehr teuer!

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Gängige Datenbank-Programme

- PostgreSQL
  - + Freie Software
  - + Sehr leistungsfähig
  - - Oftmals keine Notwendigkeit für zahlreiche seiner Features

Gängige Datenbank-Programme

- MySQL
  - + Freie Software (für nicht-kommerzielle Anwendungen)
  - + Sehr schnell
  - + Weit verbreitet für Web-Applikationen
  - + Meist in *Web-Hosting*-Angeboten enthalten
  - + Gute Dokumentation
  - + Hinreichend leistungsfähig
  - + Für viele Plattformen verfügbar
  - - Weniger komplexe Abfrage-Konstrukte möglich als mit PostgreSQL

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Erforderliche Komponenten eines Datenbank-basierten Webservers

- Betriebssystem
- Webserver (-Prozess)
- Datenbank-Server (-Prozess)
- **Geeignetes Bindeglied (= Programm) zwischen Webserver und Datenbank-Server**
- Allenfalls Client-seitige Software

Schnittstellen-Programm

- Unabhängiges Programm:
  - CGI (*Common Gateway Interface*) – UNIX, Linux
  - ASP (*Active Server Pages*) – Windows
  - \+ Programm kann kompromisslos optimiert werden
  - \+ Compilierte Programme sind schneller
  - – Aufruf erfordert Start (und Beendigung) eines eigenen Prozesses – aufwändig!
- Plug-in für Webserver-Prozess
  - \+ Läuft als Teil des Webserver-Prozesses
  - – Meist weniger gute Performance

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Schnittstellen-Programm

- Wird vom Client-Browser explizit oder implizit bei der Übergabe von (Formular-) Daten-Inhalten aufgerufen
- Erstellt unter Verwendung der Informationen aus der Datenbank (und anderen Quellen) Inhalte (HTML-Code, Bilder, usw.), die vom Webserver an den Client weitergegeben werden

Anforderungen an ein Schnittstellen-Programm

- Kompatibilität mit Webserver-Software
- Komfortable Schnittstelle zur Datenbank
- Hohe Stabilität
- Hohe Performance
- Leichte Erstell- und Wartbarkeit des Programmcodes

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Anforderungen an ein Schnittstellen-Programm

- Unterstützung für modulares und objektorientiertes Programmieren
- Gute Dokumentation
- Leichte Erlernbarkeit
- Stabilität der Eigenschaften bei Versions-Änderungen

Gängige Programmiersprachen

- Server Side Includes (SSI) (in HTML-Quelltext eingebettete Konstrukte)
  - \+ Vom Webserver-Prozess ausgewertet
  - \- Nur für einfachste Anwendungen geeignet

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Gängige Programmiersprachen

- Java Server Pages (JSP)
  - + Zugriff auf die Java Enterprise Edition
  - + Sehr leistungsfähige Umgebung
  - - Hoher Ressourcen-Bedarf
  - - Speziell in *Web-Hosting*-Umgebungen vielfach nicht verfügbar

Gängige Programmiersprachen

- VBScript
  - + BASIC-artige Script-Sprache
  - - Nur von Windows (*Internet Information Server*) unterstützt

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Gängige Programmiersprachen

- JavaScript (Serverseitig)
  - + Objektorientiert
  - + Relativ Plattform-unabhängig
  - - Keine integrierte Unterstützung für Datenbank-Zugriffe
  - - Nur selten in *Web-Hosting*-Angeboten enthalten

Gängige Programmiersprachen

- C, C++
  - + Compilierter Programmcode – schnell
  - + Objektorientiert (C++)
  - + Weitgehend Plattform-unabhängig
  - - Größerer Software-Entwicklungsaufwand
  - - Nur als CGI-Programm realisierbar
  - - CGI-Programme erfordern das Starten eines eigenen Prozesses – Server stärker belastet
  - - Installation und Betrieb von CGI-Programmen erfordert bei *Web-Hosting* oft nicht verfügbare Rechte

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Gängige Programmiersprachen

- C# („*C sharp*“ – Weiterentwicklung von C++; von Microsoft als Konkurrenz zu Java entwickelt)
  - + An Anforderungen dynamischer Webseiten-Erstellung angepasst
  - - Derzeit nur in einer Windows-Umgebung verfügbar

Gängige Programmiersprachen

- Perl („*Practical Extraction and Report Language*”)
  - + Leistungsfähige Script-Sprache
  - + Weit verbreitet unter UNIX/Linux
  - - Gewöhnungsbedürftige Syntax
  - - Programme sind schwer lesbar und wartbar

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Gängige Programmiersprachen

- PHP (für „*PHP: Hypertext Preprocessor*“, ursprünglich: „*Personal Home Page*“)
  - + Script-Sprache mit leistungsfähigem Sprach-Umfang
  - + C-artige Syntax (prozedural und objektorientiert)
  - + In HTML-Code eingebettet
  - + Schnittstellen zu vielen Datenbanken
  - + (Auch) als Plug-in für Apache-Webserver
  - + Auf vielen Plattformen verfügbar
  - - Script-Sprache, daher nicht sehr gut für sehr komplexe Berechnungen usw. geeignet

Erforderliche Komponenten eines Datenbank-basierten Webservers

- Betriebssystem
- Webserver (-Prozess)
- Datenbank-Server (-Prozess)
- Geeignetes Bindeglied (= Programm) zwischen Webserver und Datenbank-Server
- **Allenfalls Client-seitige Software**

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Client-seitige Software

- Gewisse Datenverarbeitungs-Schritte erfolgen auf dem Client-Rechner und nicht am Server
- Für moderne Web-Features (z.B. *Ajax*!) unerlässlich

Client-seitige Software

- Vorteile:
  - Verringerung des Datenverkehrs
  - Schnellere Reaktion auf Benutzereingaben
  - Entlastung des Webservers

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Client-seitige Software

- Nachteile:
  - Abhängigkeit von Browser-Eigenschaften (Type, Version)
  - Direkte Datenbank-Zugriffe sind nicht möglich
  - Größerer Einfluss des Benutzers (sicherheitskritische Anwendungen!)

Client-seitige Software

- Nachteile:
  - Wegen Vielfalt der Kombinationen von Hardware, System-Software und Browser nur schwer testbar
  - Fehler auf Client-Rechnern sind kaum vorhersagbar, schwer zu detektieren und schwer zu korrigieren

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Client-seitige Software

- Anwendungen daher zweckmäßigerweise:
  - Dynamische Änderungen der Ausgabeseite (z.B. *Rollovers*, Aktualisierung von Formularelementen nach Dateneingaben)
  - Bestimmte Ausgabeoperationen (z.B. Ausgabe von Graphen)
  - Überprüfung von Eingabedaten auf Vollständigkeit und Sinnhaftigkeit
  - Reaktion auf Formular-Operationen
  - Asynchrone Änderungen der Seiteninhalte (*Ajax*)

Client-seitige Software

- Vermeiden:
  - Sicherheitskritische Operationen
  - Echte Verarbeitung von Eingabedaten (außer es erfolgt anschließend eine serverseitige Überprüfung)

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Client-seitige Software

- Beachten:
  - Manche Benutzer deaktivieren Client-seitige Software in ihren Browsern – bei modernen Browsern aber meist nicht mehr möglich
  - Es können aber auf manchen Browsern einzelne Features (z.B. Client-seitige Dialogboxen) vom User deaktiviert werden – kann zu Fehlfunktionen Client-seitigen Codes führen!

Client-seitige Software

- Beachten:
  - Features, die nur von den neuesten oder nur von bestimmten Browsern unterstützt werden, schließen einen Teil der potentiellen Benutzer aus oder erfordern aufwändigen Browser-abhängigen Programmcode!
- Daher: Minimalistische Anforderungen an die Browser stellen!

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Gängige Programmiersprachen

- Java
  - + Extrem leistungsfähig
  - + Vollständige Client-seitige Applikationen realisierbar
  - - Auf vielen Client-Rechnern ist Java gar nicht installiert
  - - Unbekanntes Verhalten der Client-Konfiguration (Java-Version, Schnittstelle zum Browser)
  - - Hoher Aufwand zum Testen der Client-seitigen Software (mit möglichst vielen Browsern auf möglichst vielen Plattformen)
  - - Nicht alle Browser unterstützen Java!

Gängige Programmiersprachen

- Active-X
  - + Extrem leistungsfähig
  - + Vollständige Client-seitige Applikationen realisierbar
  - - Reines Microsoft-Produkt; nur mit Internet Explorer verfügbar
  - - Verwendung neuer Active-X-Funktionen kann einen Client-seitigen Upgrade von Active-X und/oder Internet Explorer notwendig machen – Benutzer-Akzeptanz schlecht!

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Gängige Programmiersprachen

- JavaScript
  - + Von praktisch allen modernen Browsern unterstützt
  - + Ausgereift – fast alle Browser seit ca. 2002 unterstützen kompatible JavaScript-Funktionalität
  - - Nur relativ einfache Operationen möglich
  - - In Grenzbereichen (noch immer) Browser-Typ- und Browser-Versions-abhängig
  - - Hoher Aufwand zum Testen der Client-seitigen Software (mit möglichst vielen Browsern auf möglichst vielen Plattformen)

Struktur eines Datenbank-basierten Webservers

- Konventioneller ↔ Datenbank-basierter Webserver
- Das TCP/IP-Protokoll
- Erstellung einer Datenbank-basierten Webseite
- Moderne Web-Technologien am Beispiel von *Ajax*
- Erforderliche Komponenten eines Datenbank-basierten Webservers
- **Die LAMP-Systemstruktur**

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Komponenten eines Datenbank-basierten Webservers

System	*Webserver*	*Datenbank*	*Sprache*	*Client*
Windows	IIS	Oracle	SSI	Java
UNIX	APACHE	Access	Java	Active-X
Linux	NGINX	MS SQL	VBScript	JavaScript
		PostgreSQL	JavaScript	
		MySQL	C, C++	
			C#	
			Perl	
			PHP	

Komponenten eines Datenbank-basierten Webservers

System	*Webserver*	*Datenbank*	*Sprache*	*Client*
Windows	IIS	Oracle	SSI	Java
UNIX	APACHE	Access	Java	Active-X
Linux	NGINX	MS SQL	VBScript	JavaScript
		PostgreSQL	JavaScript	
		MySQL	C, C++	
			C#	
			Perl	
			PHP	

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

LAMP-System

- LAMP = „Linux – Apache – MySQL – PHP“
- Weit verbreitete Konfiguration für Datenbank-basierte Webserver

LAMP-System

- Vorteile:
  - *Free Software* – kostenlos (zumindest für nicht-kommerzielle User)
  - Moderate Anforderungen an die Hardware
  - Stabile Komponenten
  - Weit verbreitet, daher große Know-how-Basis
  - Relativ einfache Implementierung
  - Verwendete Sprachen (SQL, PHP) sind relativ einfach und leicht erlernbar – erste Ergebnisse sind schnell zu realisieren

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme

Andere System-Strukturen

- Server-Betriebssystem hat im Allgemeinen nur geringen Einfluss auf Funktionalität eines Datenbank-basierten Webservers
- Für andere Strukturen, z.B. WAMP (Windows anstelle von Linux), gelten daher im Wesentlichen die gleichen Überlegungen
- Beim Wechsel von WAMP auf LAMP oder umgekehrt ist jedenfalls ein gründlicher Test der Webserver-Software erforderlich! (z.B.: Linux ist *case sensitive*, Windows nicht)

Entwicklungsumgebungen

- „Klassische " Entwicklungsumgebung: Server unter Linux oder Windows – auch für Produktionsbetrieb geeignet
- Reine Entwicklungsumgebung: z.B. XAMPP: http://www.apachefriends.org/
- XAMPP enthält vorinstalliert und -konfiguriert Apache-Webserver, PHP, MySQL und viele andere Komponenten in den jeweils aktuellen Versionen
- XAMPP ist kostenlos und für zahlreiche Betriebssysteme verfügbar (z.B. Linux, Windows, Mac OS und Solaris)

© Karl Riedling, TU Wien, Institut für Sensor- und Aktuatorsysteme