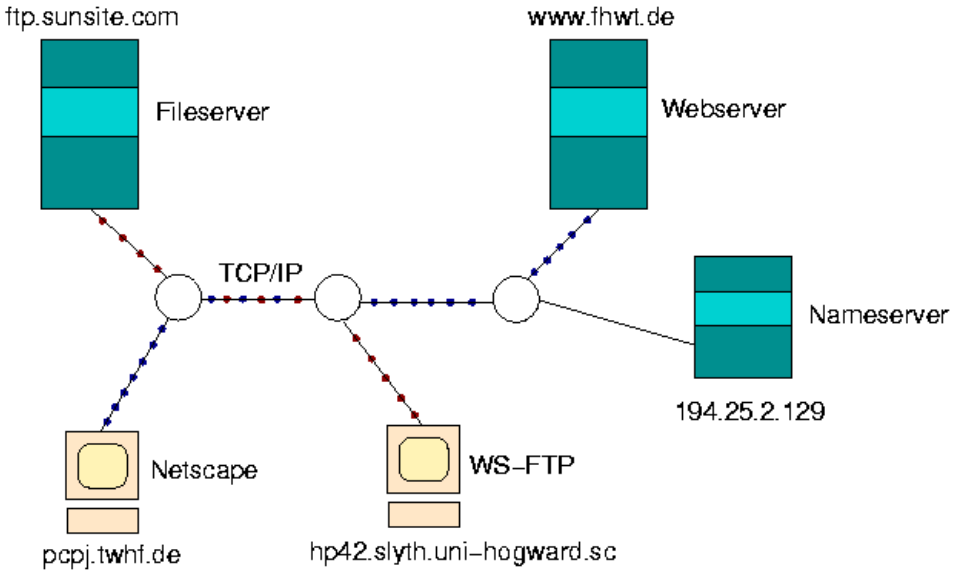
Praktische Informatik 1 (MB)

Peter Junglas 11.08.2016

Inhaltsverzeichnis

Das World Wide Web

- Internet:



  Rechner
  - identifiziert durch Nummern (**IP-Adressen**)
  - stellen Informationen/Dienste zur Verfügung (**Server**)
  - nehmen Dienste in Anspruch (**Clients**)

  physikalisches Netzwerk
  - Leitungen
  - Vermittlungsknoten

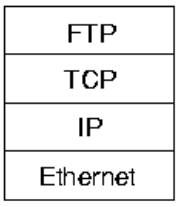
  Protokoll
  - Sprache der Rechner
  - wichtigstes im Internet: TCP/IP

  Netzdienste
  - Email
  - Datei-Übertragung (FTP)
  - Diskussionsforen (News)
  - Arbeiten auf entfernten Rechnern (Telnet)
  - vernetzte Informationen (WWW)
  - div. Infrastruktur-Dienste (Namensdienst, Zeitdienst, ..)

- Protokolle:

  Sprachen zur Kommunikation von Rechnern/Anwendungen

  auf verschiedenen Ebenen in Schichten übereinander



  FTP (**File Transfer Protocol**)
  - erlaubt das Holen und Senden von Dateien

- enthält Kommandos zum Auflisten von Verzeichnissen
- ermöglicht die Identifikation eines Benutzers (Username und Password)

TCP (**Transmission Control Protocol**)
- schickt eine Nachricht zuverlässig an einen Zielrechner
- zerlegt sie dazu in kleine Pakete
- sammelt und sortiert eintreffende Paket am Zielrechner
- setzt daraus die gesamte Nachricht zusammen
- holt ggf. verlorene Pakete auf anderem Weg

IP (**Internet Protocol**)
- schickt ein Paket an einen Zielrechner
- sucht dazu einen Weg durch das Netzwerk

Ethernet
- schickt Daten auf einer Ethernet-Leitung zu einer anderen Netzwerkkarte
- sorgt dafür, dass immer nur eine Nachricht auf einmal auf dem "Draht" liegt

untere Schichten für Anwender in der Regel unsichtbar

- IP-Adresse:

  weltweit eindeutige Nummer aus 4 Byte

  oft geschrieben als 4 Dezimalzahlen, etwa

  205.166.250.94

  erste zwei oder drei Nummern oft für bestimmte Teilnetze/Institutionen

- Rechner-Namen (**Domain-Namen**):

  Klartext-Namen, zur Vereinfachung für den Benutzer

  aufgebaut als: `RECHNERNAME.UNTERNETZ.NETZ.HAUPTNETZ`

`RECHNERNAME`	der eigentliche Rechner
`UNTERNETZ`	ggf. eine (auch mehrfache) Untergliederung von `NETZ`
`HAUPTNETZ`	eine globale Einteilung aller angeschlossenen Teilnetze

  Hauptnetze (Top-Level-Domains) für Länder und einige Organisationen, z.B.

Top-Level-Domain	**Bedeutung**
de	Deutschland
fr	Frankreich
uk	Großbritannien
jp	Japan
edu	Hochschule/Forschungseinrichtung in den USA
gov	US-Regierung
com	Firmen (auch weltweit)

  Umsetzung von Namen in IP-Adressen durch **Nameserver**

- WWW (**World Wide Web**):

  spezieller Dienst im Internet

Aufgabe: möglichst einfacher Transport vernetzter Informationen

Kernelemente

- http-Protokoll (**Hypertext Transfer Protocol**) zur Übertragung von Daten von einem Rechner (**Web-Server**) zum Rechner des Anwenders
- Sprache HTML (**HyperText Markup Language**) zur Beschreibung des Inhalts eines Dokuments incl. Querbezügen zu anderen Dokumenten
- Programm (**Browser**) zum Anzeigen von HTML-Dokumenten und zum einfachen Verfolgen von Querverweisen

- URL (**Uniform Resource Locator**):

  im ganzen Internet eindeutige Bezeichnung eines Dokuments

  Grundaufbau: `PROTOKOLL://RECHNERNAME/LOKALER_NAME`

`PROTOKOLL`	Verfahren zur Übertragung des Dokuments (http, ftp)
`RECHNERNAME`	Domainname des Servers, der das Dokument bereithält
`LOKALER_NAME`	Informationen über den Ort des Dokuments auf dem Server

  Beispiel

  ```
  http://www.peter-junglas.de/fh/index.html
  ```

Grundelemente von HTML

Allgemeine Regeln von HTML

- HTML (**HyperText Markup Language**):

  strukturierte Texte mit Verweisen und eingebetteten Multimedia-Elementen

  beschreibt logische Bestandteile eines Dokuments z.B.
  - Überschrift
  - Absatz
  - Tabelle

  legt nicht das Aussehen fest
  - z.B. Farben, Fonts, Einrückungen
  - werden in **Style Sheets** vereinbart
  - einige (veraltete) HTML-Elemente durchbrechen dieses Prinzip

  verwendet reinen (ASCII-)Text
  - leicht lesbar und verarbeitbar
  - unabhängig von Rechnertyp und Betriebssystem
  - allgemein offene Definition

- Ein erstes Beispiel:

  simple.html (HTML-Quelle)

  HTML-Text der Seite vom Browser aus anzeigen (`Ansicht/Seiten-Quelltext`)

  logische Elemente der Seite
  - Überschrift
  - vier Absätze
  - ein hervorgehobenes Element (*nicht*)
  - ein Verweis
  - ein eingefügtes Bild (Trennlinie)
  - eine Adresse mit Email-Verweis

- Auszeichnungselemente (**Tags**):

  Beispiele:

`body`	Textteil des Dokuments
`p`	Absatz
`em`	betonter Text
`a`	Verweis

  beschreiben ein logisches Element

  werden in spitze Klammern gesetzt

  umschließen (meistens) das bezeichnete Element in der Form

  ```
  <TAG>element</TAG>
  ```

  können groß oder klein geschrieben werden

  lassen sich verschachteln

- Attribute:

  Beispiele

.
- `src="juhu.gif"`
- `href="http://heritage.stsci.edu/"`
- `alt="Trennlinie"`

enthalten zusätzliche Informationen zu einem Tag

stehen in der öffnenden Tag-Klammer

haben die Form

```
NAME="WERT"
```

selten boolesche Attribute
- `=` und `"WERT"` entfallen
- Anwesenheit von `NAME` → Wert ist true, sonst false

- Stilangaben:

  in der Beschreibungssprache CSS (**Cascading Style Sheets**)

  können Tags als Style-Attribut mitgegeben werden

  ```
  <p style="color:#ff2080;">
  ```

  mehr dazu später

  seit langem veraltet:
  - Stil-Tags, etwa `<FONT>` oder `<B>`
  - besondere Stilattribute, z.B. `bgcolor` und `text` bei `BODY`

- Dokumententext:

  grundsätzlich in ASCII

  Zeilenumbrüche und mehrfache Leerzeichen werden ignoriert

  zusätzlich Angabe des Zeichensatzes im Header
  - `<meta charset="utf-8">`
  - Zeichensatz ISO-8859-15 enthält die Sonderzeichen vieler westeuropäischer Sprachen
  - deutsche Umlaute können dann direkt eingeben werden
  - Zeichensatz UTF-8 enthält Zeichen fast aller Sprachen (incl. chinesisch)

  Sonderzeichen, auch Umlaute, in der Form

  ```
  &NAME;
  ```

  einige Beispiele

Sonderzeichen	**HTML-Bezeichnung**
Ä Ö Ü	`&Auml; &Ouml; &Uuml;`
ä ö ü ß	`&auml; &ouml; &uuml; &szlig;`
æ å é	`&aelig; &aring; &eacute;`
< > & "	`< > & &quot;`
geschütztes Leerzeichen	` `

  beliebige Unicode-Zeichen als `&#NNNN;` durch Angabe der (dezimalen) Unicode-Nummer NNNN

Nummern	Aussehen
8805 8804 8800 8776 8801 8764 8793	≥ ≤ ≠ ≈ ≡ ∼ ≙
8594 8592 8596 8658 8656 8660	→ ← ↔ ⇒ ⇐ ⇔
177 8723 8734 8704 8707 8712	± ∓ ∞ ∀ ∃ ∈
9786 9785 128526 128527 128568 128586	☺ ☹ 😎 😏 😸 👻

Grundgerüst einer HTML-Seite

- Schema jeder HTML-Datei:

  Aufbau gemäß HTML-Standard

```
DOKUMENTENTYP
<html>
  <head>
     HEADER
  </head>
  <body>
     TEXTKÖRPER
  </body>
</html>
```

- Dokumententyp

  verwendete HTML-Version

  benutzter Standard

  für HTML5 einfach

```
<!doctype html>
```

- Header mit Informationen über das Dokument, z.B.

  Titel

  verwendeter Zeichensatz (s.o.)

  verwendetes Stylesheet

- Textkörper

  eigentlicher Inhalt des Dokuments

  enthält nach HTML-Standard keinen Text ohne umschließende Tags

- Kontrolle des HTML-Codes

  Browser sind sehr fehlertolerant

  Darstellung bei Fehlern stark browser-abhängig

  Kontrolle auf Standard unter http://validator.w3.org/nu

Basiselemente

- Textelemente:

  Beispiel (HTML-Quelle)

  beschreiben logische Bedeutung eines Textteils

  Beispiele

Tag	Bedeutung	Darstellung (meistens)
`em`	betont	kursiv
`cite`	Zitat	eingerückt, kursiv
`code`	Programmtext	nicht-proportionaler Font
`sub`	Index	kleiner, tiefgestellt
`sup`	Exponent	kleiner, hochgestellt

- Einbinden von Bildern:

  mit `img`-Tag, ohne Endtag

  ```
  <img src="peter.jpg" width="400" height="200" alt="Porträt von Peter">
  ```

  wird wie ein großer Buchstabe behandelt (also als Textelement)

  mögliche Bildquellen
  - Datei im gleichen Verzeichnis (`"photo.jpg"`)
  - Datei in anderem Verzeichnis (`"../bilder/photo.gif"`)
  - Datei direkt aus dem Netz (`"http://www.bilder.de/photo.jpg"`)
  - Copyrights beachten!

  `width`, `height`
  - angezeigte Größe des Bildes in Pixel
  - u.U. verzerrte Darstellung
  - Ladezeit nur von der Originalgröße abhängig!

  `alt`
  - alternativer Text (Vorschau oder für Textbrowser)
  - obligatorisch!

- Absatzelemente:

  Beispiel (HTML-Quelle)

  beschreiben Elemente zur Gliederung

  Grundelement `<p>` (**Paragraph**) normalerweise durch Zeilenumbruch und Leerzeile dargestellt

  Positionierung (links-, rechtsbündig, zentriert) möglich mit Style-Attribut

  ```
  style="text-align:right;"
  ```

  Zeilenumbruch erzwingen mit `<br>` (ohne End-Tag)

  vorformatierter Text mit `<pre>`

  Überschriften mit `<h1>`, `<h2>` ... `<h6>`

semantische Strukturierung einer Seite mit (u.a.)`main`, `header`, `footer`, `nav`

- Listen:

  Beispiel (HTML-Quelle)

  ganze Liste
  - unnummeriert `<ul>`
  - nummeriert `<ol>`

  Elemente der Liste
  - jeweils mit `<li>` eingeleitet

  Darstellung
  - eingerückt
  - jedes Element in einzelner Zeile
  - vor jedem Element Listensymbol (Kreis, Kästchen) oder Nummer

  Schachtelung
  - Liste darf nicht direkt in einem Absatz `<p>` stehen
  - jedes Listenelement darf Absätze enthalten

  Stilarten
  - Listensymbol oder Art der Nummerierung durch Style-Element `list-style-type`
  - Werte für `<ul>`: `disc`, `circle`, `square`, `none`
  - Werte für `<ol>`: `decimal`, `lower-roman`, `upper-roman`, `lower-alpha`, `upper-alpha`, `none`

Verweise

- Aufbau eines Verweises:

  mit `<a>`-Tag (**Anchor**)

  ```
  <a href="Verweisziel">Verweistext</a>
  ```

  Darstellung
  - normalerweise Verweistext unterstrichen und anklickbar
  - kann durch Stilangaben geändert werden
  - Verweistext kann auch ein Bild sein

  Verweisziel
  - WWW-Adresse (`http://www.nasa.gov/`)
  - lokaler Dateiname (`../bilder/image.gif`)
  - Email-Adresse (`mailto:papst@vatikan.va`)
  - URLs mit anderen Protokollen (`ftp`, `telnet`, `smb`)

- Einsatz von Verweisen:

  Navigation innerhalb einer Homepage
  - Blättern, zur Hauptseite, Inhaltsverzeichnis,...
  - am besten immer gleichartig
  - häufig mit Javascript-Techniken kombiniert (Aufklappen, Markieren, Öffnen in anderen Tabs, ...)

  interne oder externe Querverweise

  Kontaktaufnahme (Email-Verweis)

  wichtig:
  - aussagekräftiger Verweistext
  - Groß-/Kleinschreibung bei Dateinamen beachten
  - immer '`/`' als Trennzeichen für Verzeichnisse, nicht '`\`'

Tabellen

- Grundstruktur:

  `<table>` umschliesst ganze Tabelle

  `<tr>` bezeichnet eine Zeile (**Table Row**)

  `<td>` umschliesst ein einzelnes Feld (Spalte) in einer Zeile (**Table Data**)

  Beispiel

```
<table>
  <tr>
    <td> 1. Zeile, 1. Spalte</td>
    <td> 1. Zeile, 2. Spalte</td>
  </tr>
  <tr>
    <td> 2. Zeile, 1. Spalte</td>
    <td> 2. Zeile, 2. Spalte</td>
  </tr>
</table>
```

  optional `<th>` (**Table Header**) statt `<td>`
  - für erste Zeile (Spaltenüberschriften)
  - meistens fett dargestellt

- Gestaltung von Tabellen:

  Beispiel (HTML-Quelle)

  Rahmenbreite festlegen

```
<table style="border:2px solid black;">
```

  Abstand der Elemente von den Gitterlinien

```
<table style="padding:20px; border-spacing:10px;">
```

  Breite der Tabelle
  - automatisch nach Textinhalt
  - Größe in Pixel festlegbar durch

```
<table style="width:100px;">
```

  - Größe relativ zum Browser-Fenster mit

```
<table style="width:60%;">
```

  - `width` auch für einzelne Spalten am besten mit `colgroup`
  - Größe gilt nur, wenn Platz für Tabelleninhalt ausreicht

  Höhe einer Zeile

```
<tr style="height:200px;">
```

  horizontale Ausrichtung
  - `<td style="text-align:left;">` (bzw. `"right"`, `"center"`)
  - bei `td` und `th` für ein Element
  - bei `tr` für eine ganze Zeile

vertikale Ausrichtung

- `<td style="vertical-align:top;">` (bzw. `"middle"`, `"bottom"`)
- wie `text-align` bei `td`, `th` und `tr`

Zentrieren einer ganzen Tabelle auf einer Seite

- mit Stilelementen `margin-left:auto; margin-right:auto;`
- beachte entsprechenden Selfhtml-Artikel

Formulare

- Einsatz von Formularen:

  grundlegende Eingabe-Möglichkeit in HTML

  enthält Eingabeelemente sowie beliebige weitere HTML-Elemente zur Gestaltung

  Beispiel forms1.html (HTML-Quelle)

- Definition eines Formulars:

  Tag `<form>` umschliesst ganzes Formular

  `action`-Parameter: Wohin mit den Daten?

  mögliche Aktionen
  - als Email verschicken (`mailto:peter@junglas.name`)
  - an auswertende Seite (CGI, PHP) übergeben

  `method`-Parameter: Wie werden die Daten verschickt?
  - **get**: kodiert in der URL
    z.B. `http://www.junglas.name/echo.php?user=klaus&passwort=juhu`
    auf kleine Datenmengen beschränkt
    kann man als Bookmark speichern
    Werte sind sichtbar!
  - **post**: als eigener Datenblock
    bei großen Datenmengen
    wenn Inhalt nicht sichtbar sein soll

  Button zum Abschicken (`submit`) löst Aktion aus

- Einzeilige Text-Eingabefelder:

  definiert durch Tag `<input>`

  kein Ende-Tag

  Parameter `size`
  - Länge des Feldes in Zeichen

  Parameter `name`
  - Name des Feldes
  - wichtig für auswertendes Programm

  Parameter `value`
  - zur Vorbelegung des Textes

  Passwort-Felder
  - zeigen eingegebene Zeichen nur als * an
  - erzeugt mit `type="password"`

- Buttons zum Abschicken und Zurücksetzen:

  mit `<button>` erzeugt

  Inhalt des Tags wird Beschriftung

  Abschicken aller Formular-Daten
  - `type="submit"` (default)

- muss in einem Formular auftauchen (sinnvollerweise)

Zurücksetzen aller Formular-Daten

- `type="reset"`
- kann in einem Formular auftauchen
- in der Regel überflüssig

- Ein komplizierteres Beispiel:

  forms2.html (HTML-Quelle)

  enthält die wichtigsten Formular-Elemente

  übersichtliche Anordnung

- Auswahllisten:

  Listen zur Auswahl eines oder mehrerer Elemente

  definiert durch `<select> ... </select>`

  Listeneinträge mit `<option> Listentext`

  Parameter `size`
  - Zahl der gleichzeitig sichtbaren Elemente
  - Rest ggf. über Scrollbar erreichbar
  - `size = 1` → aufklappbare Liste (**drop-down**)

  Parameter `multiple`
  - mehr als ein Element darf ausgewählt werden
  - Default: nur ein Element

  Parameter `name`
  - Name für das auswertende Programm
  - bei `multiple` für Auswertung mit PHP `[]` (= mehrere Werte) anhängen

- Checkboxen:

  Knöpfe zur (Mehrfach-)Auswahl

  definiert durch `<input>`-Tag und Attribut `type="checkbox"`

  Parameter `name`
  - Name für das auswertende Programm
  - für Auswertung mit PHP `[]` anhängen
  - alle Checkboxen mit gleichem Namen bilden eine Gruppe

  Parameter `value`
  - Wert, der für gedrückten Knopf übergeben wird

  Tag `label`
  - fasst Checkbox und Beschriftung zusammen
  - dadurch auch Text wird anklickbar

- Radiobuttons:

  Menge von Knöpfen, von denen genau einer gedrückt ist (außer evtl. am Anfang)

  definiert durch `<input>`-Tag und Attribut `type="radio"`

  Parameter `name` und `value` wie bei Checkboxen

Parameter `name` ohne `[]`

- Textfelder:

  Eingabefeld für mehrzeilige Texte

  definiert durch `<textarea></textarea>`

  erzeugt eigenen Absatz

  Parameter `rows` und `cols`
  - Zahl der Zeilen und Spalten
  - bei größerem Eingabetext automatisch Scrollbars

- Bemerkungen zur Formatierung:

  Anordnung der Elemente durch normale HTML-Anweisungen

  häufiges Hilfsmittel: unsichtbare Tabellen (ohne Rand)

  andere Möglichkeit z.B. Gruppierung mit horizontalen Linien (`<hr>`)

Gestalten mit Stylesheets

- Stylesheet:

  Definition von Formatierungseigenschaften

  eigene Sprache CSS als Ergänzung zu HTML

  W3C-Standard

  kann geprüft werden unter https://jigsaw.w3.org/css-validator/

  beeindruckende Beispiele im CSS-Zengarden

  Grundstruktur eines Stilelements
  - `Selektor { Eigenschaft: Wert;}`
  - Selektor gibt an, auf welche HTML-Elemente der Stil angewendet wird

- Einbinden von Stil-Elementen:

  direkt an einen HTML-Tag
  - `<TAG style="font-size: 10pt;">`
  - unschön, da Inhalt und Stil vermischt werden

  in den Header

```
<style>
   /* Das ist ein Kommentar */
   li {font-size: 10pt;} /* kleine Fontgröße für Listenelemente */
</style>
```

  in eine eigene Datei
  - `<link rel="stylesheet" type="text/css" href="meinstil.css">`
  - kann für mehrere HTML-Dateien verwendet werden
  - → einheitliches Aussehen

- Adressieren mit dem Selektor:

  ein oder mehrere Tags
  - `li, p, h1 {font-size: 12pt;}`
  - gilt für alle `<li>`, `<p>`, `<h1>`

  verschachtelte Tags
  - `td li {color: red;}`
  - gilt nur für `<li>` innerhalb von `<td>`
  - kann auch tief verschachtelt sein

  Klassen
  - `h3.menue {border: solid #908070;}`
  - wirkt auf alle Elemente `<h3 class="menue">`

  Pseudoklassen
  - `a:link {color: #660000;}`
  - `a:visited {color: #990000;}`

  Individualformate
  - `td#portrait {width: 253px;}`
  - wirkt auf das Element `<td id="portrait">`
  - eine id darf nur einmal im Dokument vorkommen

beliebig kombiniert

- `tr#footer td.odd {width: 10px;}`
- wirkt auf ein `<td class="odd">` innerhalb eines `<tr id="footer">`

- Einige Eigenschaften:

  Fonts

  - `font-family`, z.B. `'Times New Roman', 'Nimbus Roman No9', serif`
  - immer Alternativen angeben (als letztes `serif`, `sans-serif` o.ä.)
  - `font-size`, `font-weight`

  Masseinheiten für Längen

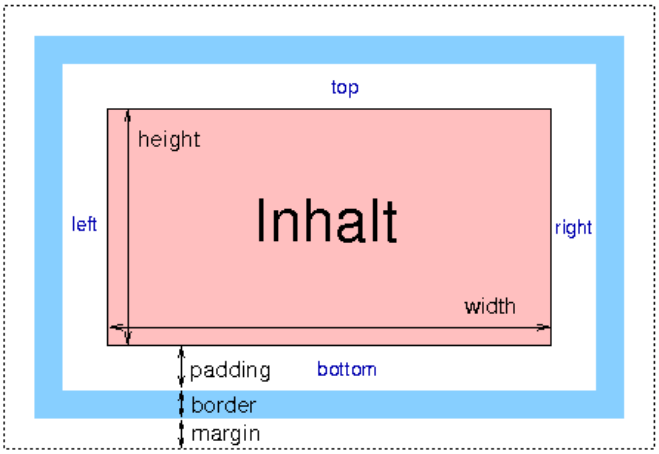
px	Pixel
pt	Typographischer Punkt
mm	Millimeter
em	Breite des 'M' im aktuellen Font
%	Prozent

  Farben

  - `color`, `background-color`
  - Farbnamen (`black`, `blue`), RGB-Werte `#RRGGBB`

  Ränder:

  - `padding`, `border-width`, `margin`,



  - einzeln angebbar (`margin-left: 10px;`)
  - ein Wert für alle (`border-width: 3mm;`)
  - alle vier Werte auf einmal (`padding: 2px 3px 2px 5px;`)
  - Reihenfolge der vier Werte: `top`, `right`, `bottom`, `left` ("TRouBLe")

  Textumfluss (um einen Block herum)

  - `float:left` bzw. `float:right`
  - `clear:left` bzw. `clear:right` startet Bereich unter dem Float-Block

- Beispielseite:

  cssdemo.html mit Stylesheet austen.css

  typische Aufteilung

  - Kopfzeile
  - Fusszeile

- globale Navigationsleiste oben
- lokeles Navigationsmenü links
- Service-Spalte rechts
- Hauptbereich mit normalem Text + Info-Kasten

Seite ohne CSS

Struktur der Seite

- Tabellen rot, Listen blau, Überschriften grün, `div`/`span` schwarz
- erzeugt durch einfaches Stylesheet austen02.css

- Allgemeine Formate (austen03.css):

  spezieller Font (Verdana, falls möglich, aber jedenfalls ohne Serifen)

  Standardfarben schwarz/weiß (überschreibt ggf. Browser-Defaults)

  keinen Platz verschenken bei Body, Tabellen und -feldern

  Tabellen auf 100% Breite

  Inhalt der Tabellenfelder nach oben (statt mittig)

  Farben der Links an verwendetes Farbschema anpassen

  Überschrift im Textteil formatieren

  - Text (Größe, Gewicht, Farbe), Abstand und unteren Rand

  Ergebnis

- Gestaltung des Kopfbereichs (austen04.css):

  Id's vergeben zur eindeutigen Kennzeichnung von Elementen

  - `title` (Kopfbereich links, Titel)
  - `portrait` (Kopfbereich rechts, Bild)

  Titelbereich

  - vertikal mittig
  - Hintergrundbild + Farbe
  - Fontart und -größe
  - Abstand nach links

  Bild

  - `td`-Feld an Bild anpassen (damit linke Spalte den Rest bekommt)
  - Bild nach unten rücken (sonst Platz für Unterlänge)

  Ergebnis

- Gestaltung der Navigationsleiste (austen05.css):

  Ids einführen

  - `content-top` (Navigationsleiste)
  - `content-left` (Navigationsleiste, links)
  - `content-links` (Navigationsleiste, Verweise)

  Formatierungen

  - Fonts und Farben
  - vertikal mittig ausgerichtet
  - Abstand zwischen den Links

  padding bei `a`, nicht bei `td`

Ergebnis

- Gestaltung von Haupttext, Randspalten und Fußzeile (austen06.css):

  Ids einführen
  - `leftside` (Menü links)
  - `content` (Haupttext)
  - `rightside` (Service-Spalte rechts)
  - `footer` (Fußzeile)
  - `feedback` (Footer, Kontakt)
  - `author` (Footer, Name)
  - `date` (Footer, Änderungsdatum)

  linke Spalte
  - Breite und Hintergrundfarbe
  - Art der gesamten Liste (`list-style-type`, `list-style-position`)
  - Font, Unterstreichung (`border-bottom`) und Platz der einzelnen Elemente

  Haupttext
  - Abstand
  - Font

  rechte Spalte
  - Breite
  - Font und Padding von Tabellen-Elementen
  - Spaltenüberschrift (`h3`)

  Fußzeile
  - Text mittig, klein
  - oben Rand, etwas Platz
  - links Farbe an Spalte angepasst
  - `date`-/`author`-Selektoren kompliziert, um Vorrang-Reihenfolge zu garantieren

  Ergebnis

- Besonderheiten (austen.css):

  `cite` (Zitat rechts)
  - Textblock mit Rahmen
  - Überschrift dicht am Text

  `class` statt `id`, wenn etwas mehrmals pro Seite vorkommen kann

  Klasse `infobox`
  - eigene Box mit fester Breite und Abständen
  - wird links vom normalen Text umflossen

  Klasse `rightside-head` zur Gestaltung der Überschriften rechts

  Klassen `even`/`odd` zur Unterscheidung von geraden/ungeraden Tabellenzeilen

  Klasse `preis` zur Heraushebung von Preisen

  Ergebnis

Multimedia-Daten

- Einführung
- Klänge
- Farben
- Bilder
- Videos

Einführung

- Wichtige Multimedia-Typen:

  Bilder (JPG, GIF, PNG, TIFF)

  Klänge (WAV, MP3, MIDI)

  Videos (MPEG, AVI, Quicktime, OggTheora)

  Animationen und Programme (Java-Applet, Flash)

  3D-Modelle (VRML)

- Anzeige im Browser:

  Beispiel mit Ogg Theora (HTML-Quelle)

  früher: Darstellung mit Flash-Plugin im Browser
  - hat den „Siegeszug“ von Flash ermöglicht (YouTube)

  Generelles Problem: Codecs nicht plattformübergreifend vorhanden (wg. Lizenzzahlungen und Abgrenzung der Hersteller)
  - Aktueller Kampf: h.264/h.265 (Microsoft, Apple) vs. WebM (Google)

- MIME-Typ:

  Information über die Art des Multimedia-Elements

  im Tag angegeben oder durch Datei-Endung festgelegt

  legt die Interpretation der Bytes fest
  - so wie der Zeichensatz bei Textdateien
  - z.B. bedeutet das Byte `0xe4` ä (in ISO8859-1, westeuropäisch), ф (in ISO8859-5, kyrillisch) oder δ (in ISO8859-7, griechisch)

  einige Beispiele

Typ	Beschreibung	Endung
image/png	PNG image	png
image/tiff	TIFF image	tiff, tif
audio/wav	Microsoft wave file	wav
audio/mpeg3	MPEG audio	mp3
audio/midi	MIDI audio file	midi, mid
video/mpeg	MPEG animation	mpeg, mpg, mpe
video/ogg	Ogg Theora Video	ogg
video/quicktime	Quicktime animation	mov, qt
video/msvideo	AVI animation	avi

  offizielle Liste

Klänge

- Physikalische Basis:

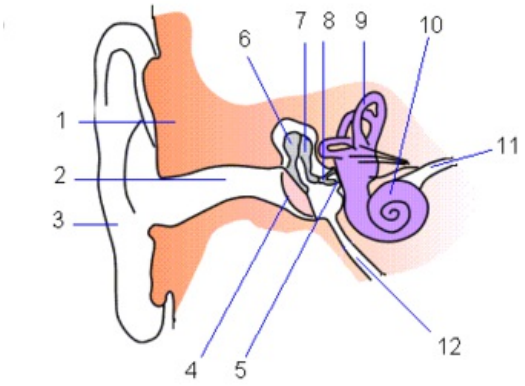
  periodische Druckschwankungen um den (näherungsweise festen) Umgebungsdruck

  breiten sich als Schallwellen aus

  an einem Ort treffen direkte und (mehrfach) reflektierte Wellen ein

- Physiologische Basis:

  Hörapparat



  Druckschwankungen mechanisch (Trommelfell, Hörknöchelchen) zum Innenohr übertragen

  Bewegung feiner Härchen in der Gehörschnecke reizen Nerven

  Frequenzbereich etwa 16 Hz - 16 kHz (altersabhängig)

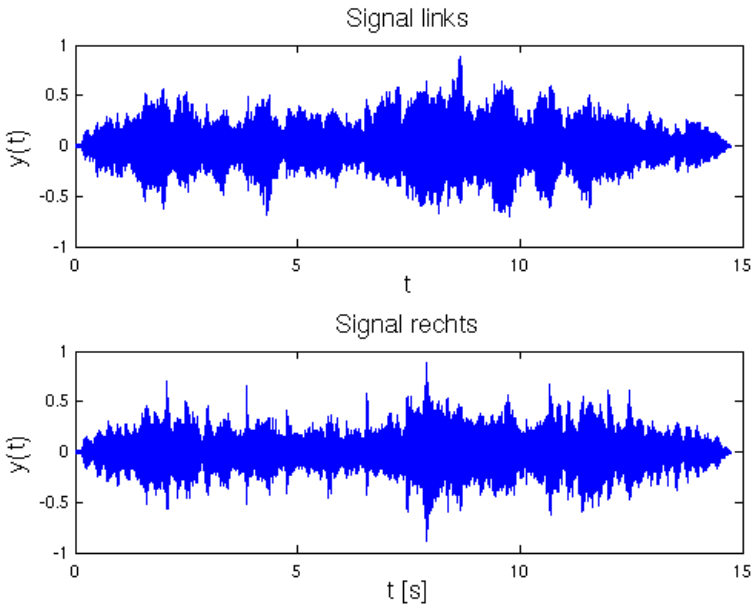
  Dynamikumfang etwa 10^{-12} W/m^2 (Hörschwelle) bis 1 W/m^2 (Schmerzgrenze)

  Empfang an zwei Stellen (Ohren) → Unterschiede (Lautstärke, Zeit, Klang) erlauben Richtungswahrnehmnung

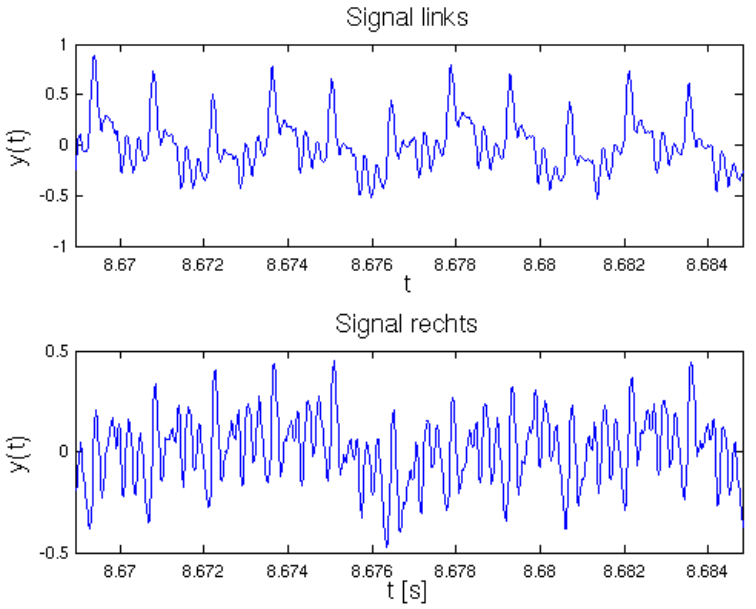
- Umwandlung in digitale Signale:

  2 Mikrofone → 2 analoge Signale (Stereo)

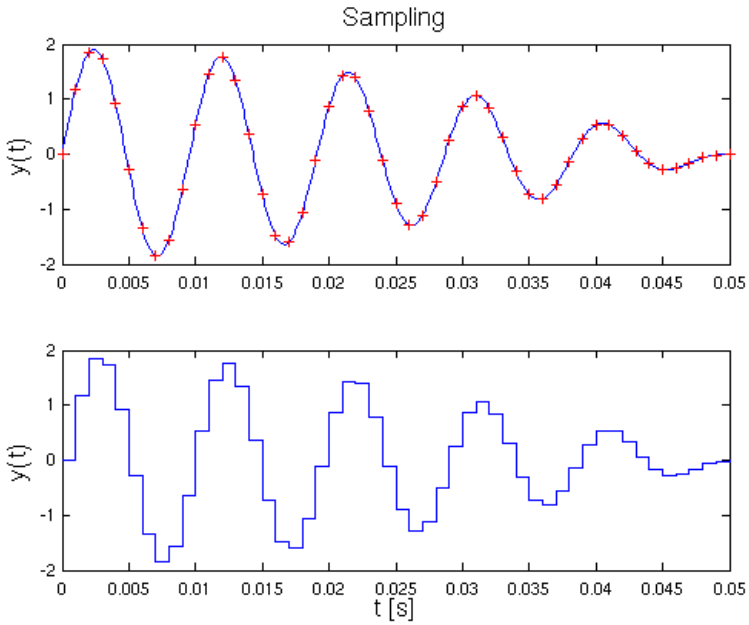
  Beispiel-Clip



- Ausschnitt



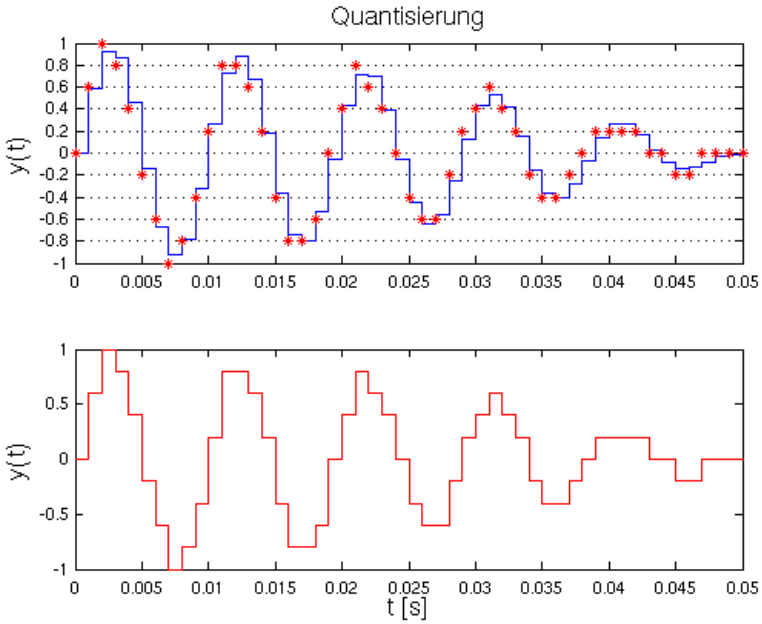
Diskretisierung = Abtastung (Sampling) in festen Abschnitten



- Sample-Zeit dt ≙ Sample-Frequenz f_S = 1/dt
- max. Frequenz im gesampelten Signal = $f_S/2$ (**Nyquist-Shannon-Theorem**)
- typische Sample-Frequenzen

fS [kHz]	**Anwendung**
8	Telefon
44.1	CD
96	DVD-Audio

Quantisierung = Annähern der einzelnen Werte durch feste Anzahl von Stufen



- typische Quantisierungen

Stufenzahl	Bit	Anwendung
256	8	Telefon (logarithmische Kennlinie)
65536	16	CD
16777216	24	DVD-Audio

- Auswirkungen:

  Originalclip (44.1 kHz, 16 Bit)

  niedrige Samplerate
  - Clip 01 (11.02 kHz, 16 Bit)
  - Clip 02 (5.51 kHz, 16 Bit)

  niedrige Auflösung
  - Clip 03 (44.1 kHz, 8 Bit)
  - Clip 04 (44.1 kHz, 4 Bit)

  mit falscher Samplerate abgespielt
  - Clip 05 (22.05 kHz, mit 44.1 kHz abgespielt)
  - Clip 06 (44.1 kHz, mit 22.05 kHz abgespielt)

- WAV-Format:

  von Microsoft definiertes Audioformat

  Aufbau:
  - Header (Dateityp, Größe)
  - Formatbeschreiber (u.a. Samplerate, Bitbreite, Zahl der Kanäle)
  - Daten

  Daten können verschieden kodiert sein (PCM, MP3, DOLBY AC2, MULAW etc.)

  meistens PCM: Werte direkt (unkomprimiert), jeweils links/rechts abwechselnd

  Ergebnis für 1 s Klang in CD-Qualität
  - 2 Kanäle à 44100 Zahlen à 16 bit → 1411 kbit/s

- Datenkompression:

  verlustfrei schwierig, da wenig Wiederholungen von Bytefolgen

  viele verlustbehaftete Formate

  Datenrate ↔ Klangqualität bei der Kompression meistens einstellbar

  nutzen Eigenheiten des Hörapparates aus, z. B.
  - geringe Empfindlichkeit bei sehr tiefen bzw. sehr hohen Frequenzen
  - Bildung von Frequenzgruppen im Innenohr
  - Maskierung von leisen höheren Tönen durch laute Basstöne

  generelle Strategien
  - konstante Datenrate

    Vorteil: Durchsatz (z.B im Internet) und Anforderung an Player konstant
  - variable Datenrate

    Vorteil: "schwierige" Anteile werden aufwändiger kodiert als "einfache" → Qualität konstant

- Kompressionsverfahren:

  MP3
  - wichtige Anwendung: Internet, Player
  - bei 128 kbit/s und neuen Kodierern (Lame) kaum von CD zu unterscheiden

  AC-3 ("Dolby Digital")
  - bei Filmen, DVDs für die Tonspur
  - verwendete Datenraten: 192 kbit/s (DVD) bzw. 320 kbit/s (im Kino)

  AAC ("Nachfolger von MP3")
  - verwendet bei iPod, Nintendo
  - 128 kbit/s für CD-Qualität

- MP3-Klangbeispiele:

  mit jeweils konstanter Datenrate

  kodiert mit "`lame -h -b RATE`"
  - Clip 07 (128 kbit/s, 231.0 kB)
  - Clip 08 (64 kbit/s, 115.3 kB)
  - Clip 09 (32 kbit/s, 57.6 kB)
  - Clip 10 (16 kbit/s, 28.7 kB)

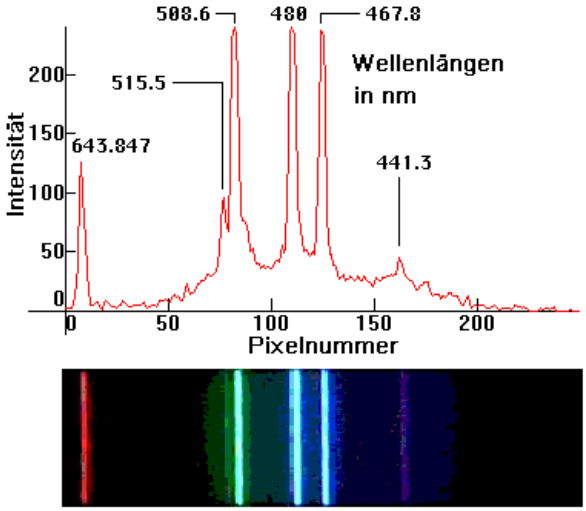
Farben

- Physikalische Basis:

  Lichtwellen verschiedener Wellenlängen werden von Oberflächen reflektiert oder gestreut

  Anteil der verschiedenen Wellenlängen kann als Spektrum dargestellt werden

  Beispiel: Spektrum einer Niederdruck-Cadmiumdampflampe

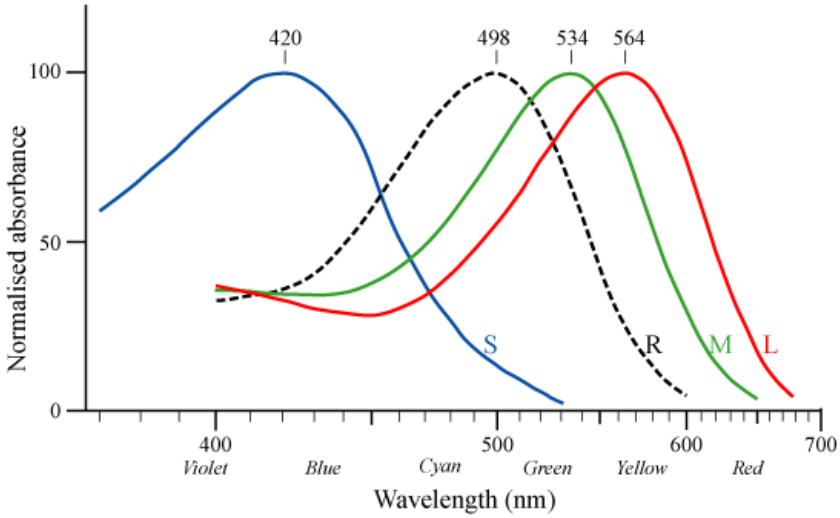


- Physiologische Basis:

  zwei Typen von lichtempfindlichen Zellen in der Netzhaut

  - **Stäbchen**: besonders empfindlich, keine Farbunterscheidung, vor allem zum Dämmerungssehen
  - **Zäpfchen**: drei Arten mit unterschiedlichen spektralen Empfindlichkeiten, als Rot-, Grün- und Blau-Rezeptor bezeichnet

  Empfindlichkeitskurven der Zäpfchen



  Menschen unterscheiden etwa 1 Million Farben

  bei Säugetieren häufig nur 2 Zapfentypen, bei anderen Wirbeltieren und Insekten häufig 4 Typen

- Farbmodelle:

  Farbmodell = Methode, Farben als Zahlenwerte abzuspeichern

Farbmodell = Methode, Farben als Zahlenwerte abzuspeichern

meistens mit drei Parametern (≙ Zahl der verschiedenen Zäpfchenarten!)

RGB-Modell

- aus der additiven Farbmischung (Bildröhre)
- Helligkeiten jeweils für Rot, Grün und Blau

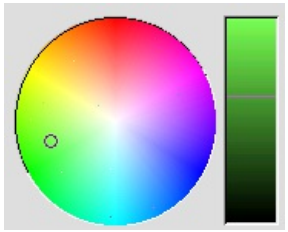
- häufig jeweils 8 bit (0 .. 255) für die drei Werte
- → insgesamt 256^3 = 16 Mio. Farben

CMYK-Modell

- Basis der subtraktiven Farbmischung (Drucker)
- Grundfarben: Cyan, Magenta, Yellow, Black (K = "Key" aus der Drucktechnik)
- Schwarz theoretisch aus CMY mischbar, ist aber in der Praxis farbstichig, daher eigener Kanal

HSB-Modell

- aus dem Farbenkreis abgeleitet



- beschrieben durch

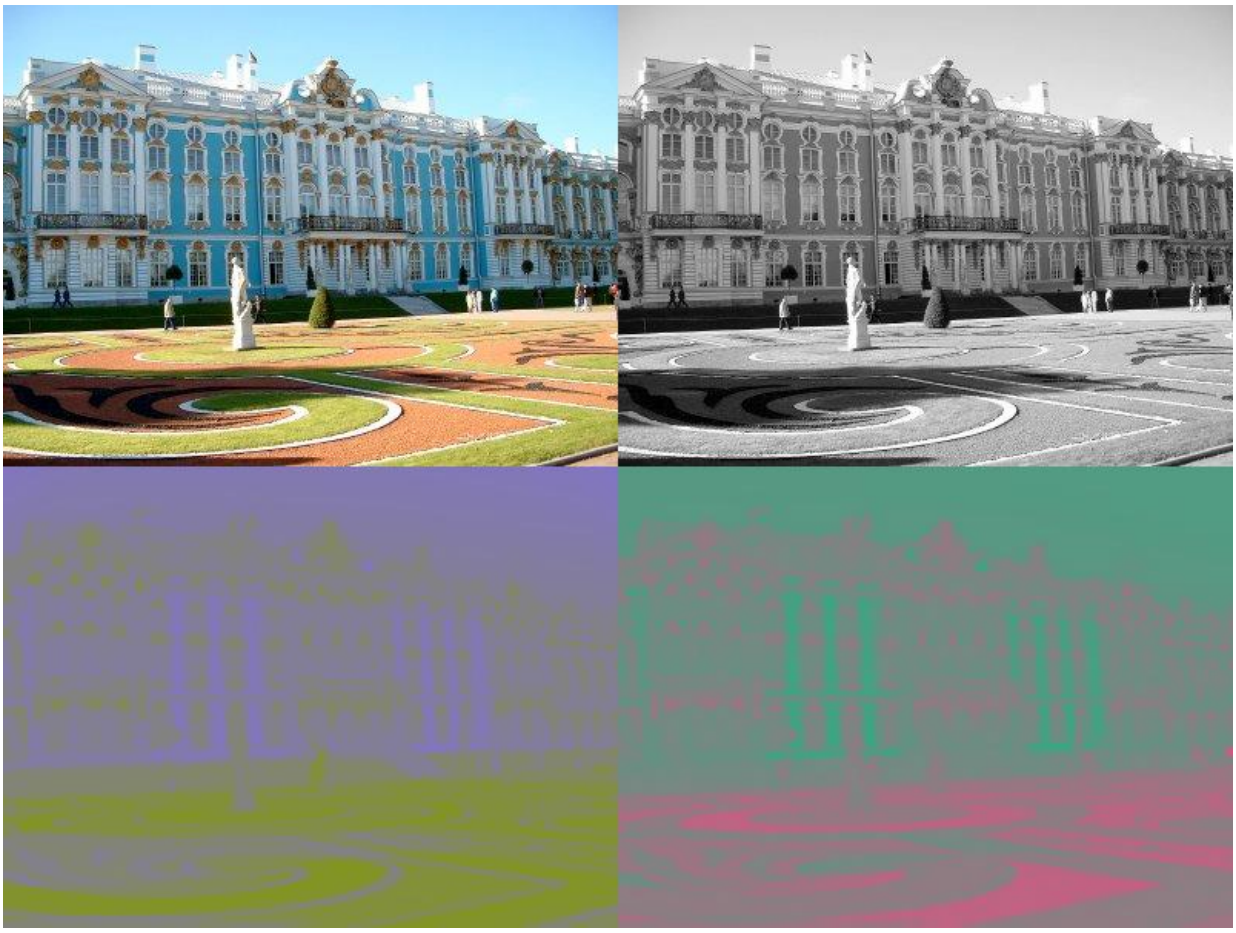
Parameter	Bedeutung	Werte
Hue	Winkel im Farbenkreis	0 .. 360
Saturation	Farbsättigung	0 .. 255
Brightness	Helligkeit	0 .. 255

YCbCr-Modell

- unterscheidet Grundhelligkeit Y, Anteil der Farbe in blau/gelb-Richtung Cb und

Anteil in rot/türkis-Richtung Cr

- Auge viel empfindlicher für Helligkeitsvariationen als für Änderungen in Cb oder Cr



- Basis für Bildkompression: Y mit mehr Bit abspeichern als Cb, Cr
- verwendet bei digitalem Fernsehen, JPEG-Bildformat, MPEG-Videoformaten (z. B. für DVD)

- Farbtabellen:

  Tabelle fester Länge mit Auswahl an Farben

  nützlich bei beschränkter Farbanzahl (ältere Grafikkarten)

  Farben der Tabelle oft an das Bild anpassbar, z.B. 256 freie Farben bei GIF-Bild

  Farbe angeben durch Nummer in der Tabelle → Reduktion des Speicherbedarfs

  Beispiel GIF-Bild
  - Farbe in RGB ≙ 24 Bit
  - stattdessen Nummer in der Tabelle≙ 8 Bit
  - also Platzreduktion um Faktor 3
  - dazu kommt noch der Platz für die Tabelle selbst (256 x 24 Bit)

Bilder

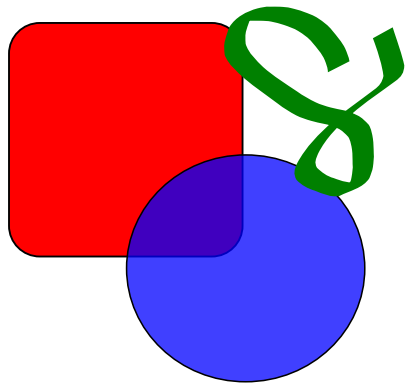
- Digitalisierung von Bildern:

  Zerlegungung in rechteckige Bereiche (**Pixel**) mit einem mittleren Farbwert

  technisch realisiert mit Scanner oder Digitalkamera

  einfachste Abspeicherung
  - RGB-Farbwerte für jedes Pixel
  - alle Pixel hintereinander
  - Header mit Breite und Höhe
  - wird verwendet beim TIFF-Format (u.a.)

  Alternative für Graphiken: Vektorgrafik
  - Objekte (Polygonzüge, Kreise, Splines etc.) und ihre Eigenschaften (z. B. Farbe, Strichdicke, Füllungsmuster) werden abgespeichert
  - Formate z.B. PS (Postscript), CDR (Corel Draw)
  - Standardformat im Internet: SVG (Scalable Vector Graphics)
  - wird von allen aktuellen Browsern außer Internet Explorer direkt unterstützt
  - Plugin für Internet Explorer verfügbar
  - Vorteil: optimale Darstellung in jeder Auflösung



- GIF-Format:

  von Compuserve eingeführt

  gut geeignet für Graphiken und Zeichnungen, weniger für Fotos

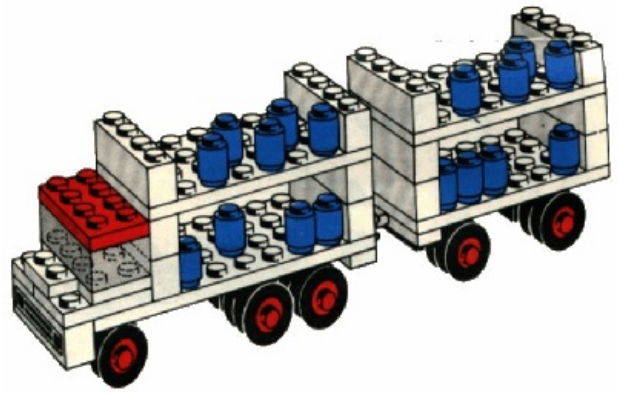
  reduziert Farben auf 256 Werte aus freier Farbtabelle

  folgende Komprimierung nicht verlustbehaftet

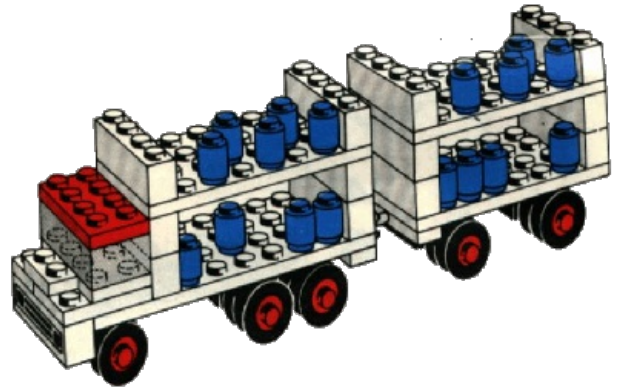
  enthaltene Daten
  - Verwaltungs-Informationen (Größe des Bilds, Kommentare etc)
  - Farben der Farbtabelle
  - Bild: für jeden Punkt die Farbnummer

  Prinzip der (LZW-)Komprimierung
  - häufig auftauchende Muster suchen
  - Muster in extra Tabelle abspeichern
  - im Bild statt Muster nur Nummer in der Mustertabelle

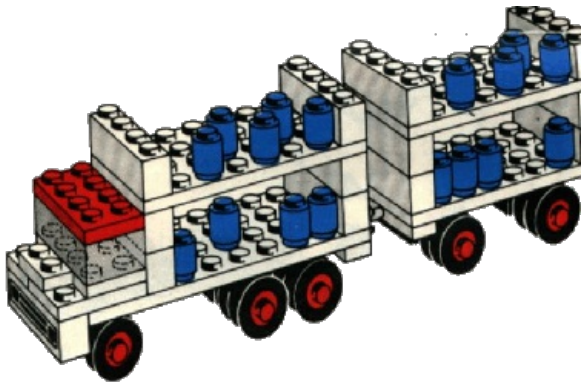
  Spezialitäten



Interlacing = Bild wird erst grob geladen, dann genau



Transparenz = Durchscheinen des Hintergrunds



Animation = Folge von Bildern in einer Datei als Filmchen

- JPEG-Format:

  für Bilder mit Echtfarben

  besonders geeignet für Fotos und Farbverläufe

  Basis der MPEG-Filmkodierung

  verlustbehaftete Komprimierung
  - nutzt Eigenschaften des Auges
  - bei hoher Komprimierung sichtbare Fehler ("Artefakte")

(Qualität 75%, 15.4 kB)

(Qualität 20%, 5.7 kB)

(Qualität 2%, 1.8 kB)

Prinzip der Komprimierung

- Wandeln in YCbCr-Farben, Y mit mehr Bit abspeichern als CbCr
- Bild zerlegen in 8x8-Blöcke
- in jedem Block kleine "hochfrequente" Anteile weglassen
- Rest mit verlustfreier Methode komprimieren

- PNG-Format:

  Nachfolger von GIF

  kann größere Farbtabellen und Echtfarben

  verlustfreie Komprimierung

  erlaubt Transparenz

Videos

- Daten eines Videos:

  Bildsequenzen

  Tonspur

  Synchronisierung und weitere Zeitinformationen

  u. U. auch Verschlüsselungsinformationen (Content Scrambling System, Digital Rights Management)

- Datenvolumen (unkomprimiert) für 90 Minuten Film in DVD-Qualität:

  Bildgröße 720x576x3 Byte = 1.19 MB

  25 Bilder/s, also 135 000 Bilder → 156.4 GB

  Ton: 44100 Sample/s * 16 bit/Sample * 2 (Stereo) * 90 min = 172.3 kB/s * 90 min = 0.887 GB

- verbreitete Formate

  MPEG-2 (DVD, DVB, HDTV)

  MPEG-4/H.264 (Blue Ray Disk)

  RealVideo, WMV (Internet-Streaming)

  häufig Rechteprobleme bei freier Verwendung (z.B. verwendet MPEG-2 über 800 Patente!)

  OggTheora, WebM (OpenSource-Alternativen ohne Rechteprobleme, z.B. bei Wikipedia)

- Details zu MPEG-2:

  Tonspur verwendet meistens AAC oder MP3

  Bilder sind grundsätzlich JPEG-kodiert, mit festen JPEG-Parametern

  Unterschied aufeinanderfolgender Bilder meistens klein, oft aufgrund von Bewegung →
  - gelegentlich komplette Standbilder (**I-Frames**), Rate ≈ 1 bit/pixel
  - dazwischen Bilder mit Unterschieden zum letzten Standbild (**P-Frames**), nutzt Bewegungsvektoren, Rate ≈ 0.1 bit/pixel
  - zwischen P-Frames Bilder mit Unterschieden zum letzten und zum nächsten Standbild (**B-Frames**), Rate ≈ 0.015 bit/pixel

  gesamte Datenrate (incl. Ton) bei DVD-Parametern maximal 9.8 Mbit/s, also 90 Minuten ≙ 6.5 GB

  viele Details bei Wikipedia

Sicherheit und Verschlüsselung im Internet

- Angriff durch Abhören des Passworts:

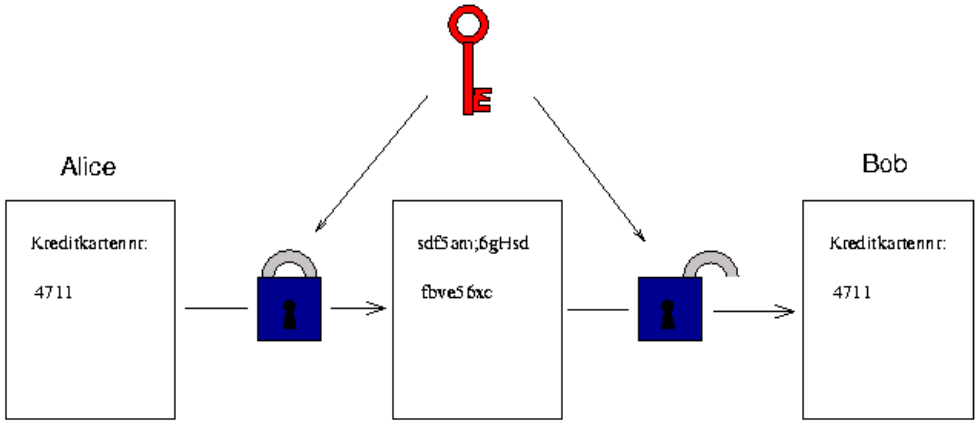
  an verschiedenen Stellen möglich
  - Netzwerk-Überwachungssoftware am Neben-PC (missgünstiger Student)
  - bei einem Internetknoten, der Nachricht weiterleitet (betrügerischer Mitarbeiter)
  - induktives Abhören des (Kupfer-)Telefonkabels (Werksspion)
  - Auffangen der Streustrahlung von Tastatur oder Monitor (Geheimagent)

  Abhilfe gegen die ersten drei Attacken
  - Nachrichten verschlüsseln

  in der Praxis am Browser
  - "sichere" Internetseiten
  - **https-Protokoll** (s = "secure") → Webserver verschlüsselt
  - Browser entschlüsselt
  - angezeigt durch Vorhängeschloss (bei Netscape)

- Symmetrische Verschlüsselungsverfahren:

  Funktionsweise



  Beispiele:
  - DES (**Data Encryption Standard**)
  - RC4 (von Rivest)

  schnell auch bei großen Datenmengen

  Problem: Schlüsselaustausch

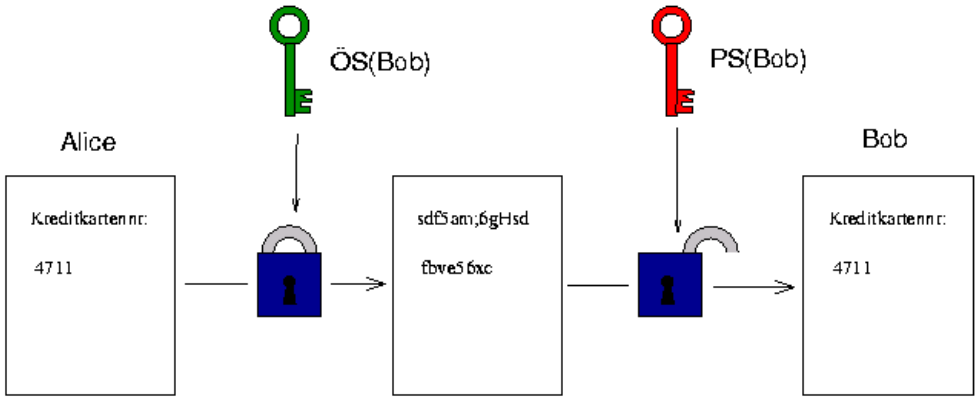
  bei persönlich bekannten Partnern auf "sichere" Weise verabreden

  für Anwendung im Homebanking
  - ein Schlüssel nötig für jeden Kunden der Bank
  - Austausch kompliziert
  - ähnliches Prinzip: TAN auf Postweg

- Unsymmetrische Verschlüsselungsverfahren:

  benutzt zwei zusammen gehörende Schlüssel
  - privater Schlüssel PS (geheim!)
  - öffentlicher Schlüssel ÖS (kennen alle)
  - PS nicht aus ÖS ableitbar

Funktionsweise



Beispiele:

- RSA (von Rivest, Shamir, Adleman)
- Diffie-Hellman
- DSA (**Digital Signature Algorithm** von Kravitz)

bei Anwendung im Homebanking

- ein öffentlicher Schlüssel der Bank reicht für alle Kunden

funktioniert in beiden Richtungen

- mit PS verschlüsseln, mit ÖS entschlüsseln
- mit ÖS verschlüsseln, mit PS entschlüsseln

Nachteil:

- Verfahren sehr aufwändig
- sehr langsam für längere Texte

Ausweg:

- Verschlüsseln des Texts mit symmetrischem Verfahren
- Austausch des Schlüssels mit Public-Key-Verfahren

- RSA-Verfahren:

  weit verbreitet und gut analysiert

  gilt als sicher

  - Grundlage: Schwierigkeit, große Zahlen zu faktorisieren
  - Rekord bisher (Ende 2005) 200 Dezimalstellen (663 Bits)
  - lief als Wettbewerb mit hohemPreisgeld für Faktorisierung

  Bestimmung eines Schlüsselpaares

  - wähle zwei (möglichst große) Primzahlen p, q
  - setze n = p*q, m = (p-1)*(q-1)
  - wähle eine Zahl e, die teilerfremd ist zu m
  - berechne Zahl d mit e*d = 1 mod m (mit erweitertem euklidschen Algorithmus)
  - öffentlicher Schlüssel: (e,n)
  - privater Schlüssel: (d,n)

  Verschlüsseln einer Nachricht b (als Zahl)

  - $c = b^e \bmod n$

  Entschlüsseln der Botschaft c

  - $b = c^d \bmod n$

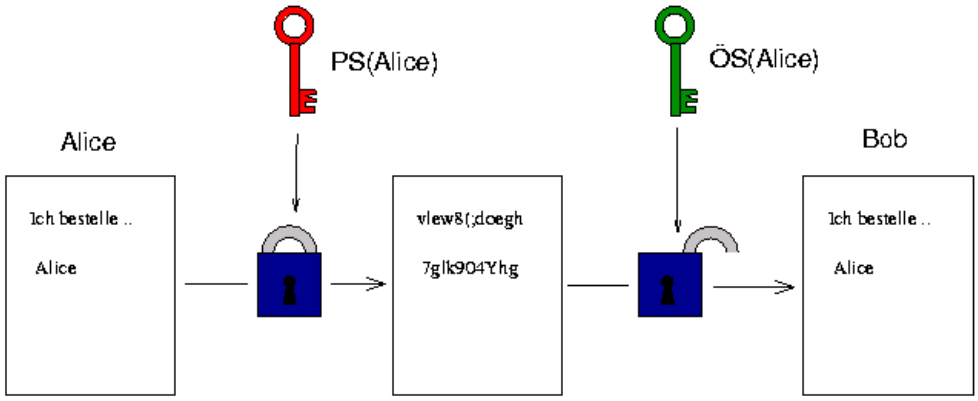
praktisch Ausprobieren (mit kleinen Zahlen)

- Digitale Unterschrift (**Signatur**):

  Problem: übertragener Text könnte gefälscht worden sein ("Man-in-the-Middle"-Attack)

  Signatur soll nachweisen
  - Absender ist korrekt
  - Text wurde nicht verfälscht

  Funktionsweise



  Text mit öffentlichem Schlüssel entschlüsselbar
  → Absender ist wirklich Besitzer des privaten Schlüssels

  zur Kontrolle reicht Verschlüsseln des Fingerabdrucks (z.B. mit MD5) statt des ganzen Textes

- Zertifikat:

  weiterhin problematisch: woher öffentlicher Schlüssel?

  nicht einfach von der Webseite des Benutzers wegen "Man-in-the-Middle"-Attack

  Zertifikat = "digitaler Personalausweis"
  - öffentlicher Schlüssel + zugehöriger Name (Person, Firma, Website)
  - u.U. weitere Informationen (Gültigkeitsdauer!)
  - unterschrieben (signiert) von "vertrauenswürdiger Stelle"

  Standard-Formate für Zertifikate
  - OpenPGP
  - X.509

- Vertrauensmodelle:

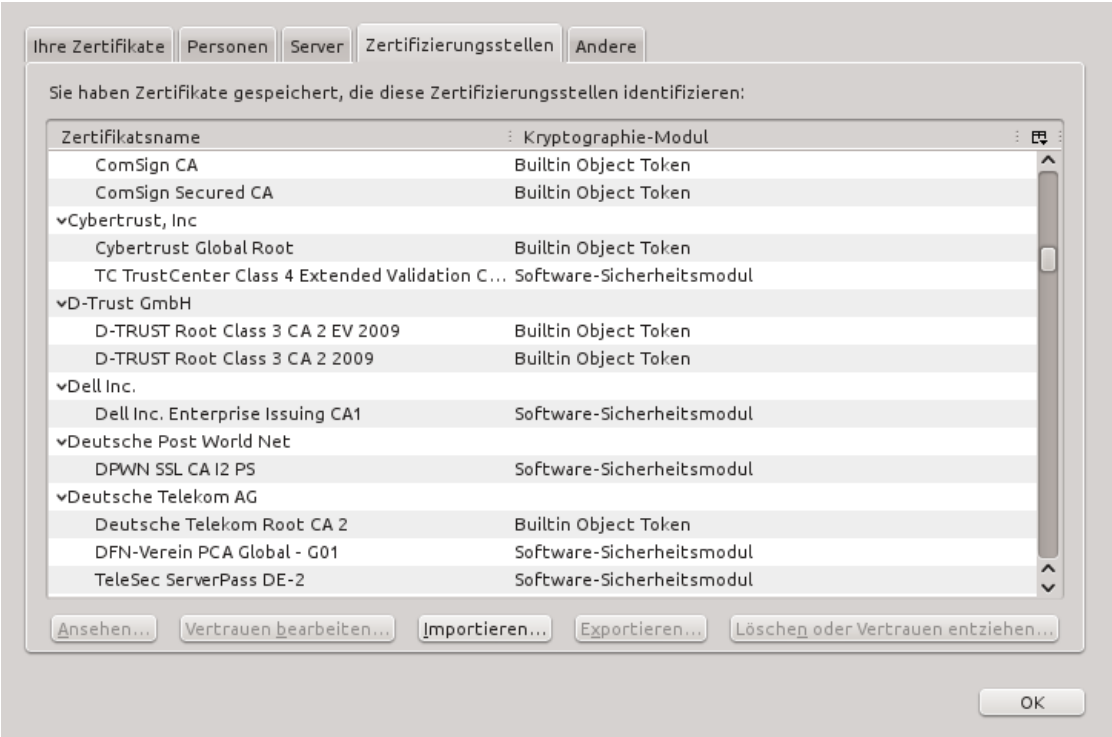
  Wer darf Zertifikate unterschreiben?
  - jeder, dem man vertraut (direktes Vertrauen)
  - jeder, dem jemand vertraut, dem man selbst vertraut (Netz des Vertrauens)
  - spezielle **Certification Authorities** (CA) (hierarchisches Vertrauen)

  einige bekannte CA's
  - Regulierungsbehörde für Telekommunikation und Post (RegTP)
  - Trust Center der Bundesnotarkammer
  - Trust Center der Deutschen Telekom (Telesec)
  - Trust Center der Deutschen Post (Signtrust)
  - Firma Verisign (amerikanischer Marktführer)

oberste CAs zertifizieren sich selbst

Unterstützung durch Webbrowser

- kennen Zertifikate der wichtigsten CAs



- zeigen Zertifikate anderer Seiten an

- erlauben Verwaltung von eigenen und fremden Zertifikaten

Datenbanken und SQL

Datenbank-Managementsysteme

- Datenbank:

  Sammlung von Daten an zentraler Stelle

  verschiedene Organisationsformen

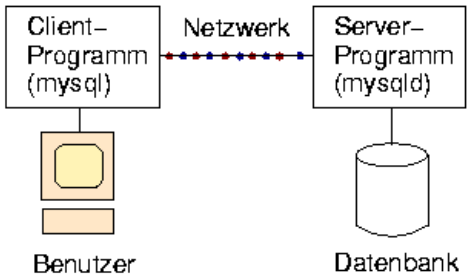
  relationale Datenbank ≙ Menge von Tabellen

- Datenbank-Managementsystem (**DBMS**):

  Programm zur Verwaltung von (großen) Datenbanken

  viele Systeme auf dem Markt, z.B.
  - Oracle
  - Sybase
  - Ingres
  - DB2
  - MySQL (Open Source, im Kurs verwendet)

  erlaubt einheitlichen, netzwerkweiten Zugriff auf Daten
  - nach außen immer gleiches Vorgehen für alle Daten
  - internes Speicherschema versteckt
  - Client-Server-Architektur



  gewährleistet Konsistenz der Daten, z.B. durch
  - Typen mit erlaubten Bereichen (Wert für Datumsfeld muss gültig sein)
  - Kontrolle beim Einfügen von Daten (Kundennummer muss zu einem Kunden gehören)
  - Kontrolle beim Löschen von Daten (Kunde löschen → sein Konto wird auch gelöscht)

  garantiert Sicherheit und Vertraulichkeit
  - eigene Benutzerverwaltung
  - Anmelden etwa mit

    ```
    mysql -h meinrechner -u meinuser -p <RET>
    meinpasswort
    ```

  - ausgefeiltes System von Zugriffsrechten
  - z.B. Rechte zum Erstellen, Ändern, Ansehen oder Löschen von Daten

- SQL (**"Structured Query Language"**):

  Sprache zur Definition und Manipulation von Daten

  beschreibt Aufgabe, nicht den Lösungsweg (nicht-prozedural)

  Syntax an normales Englisch angelehnt

standardisiert durch ISO

- grundlegende Version: SQL92 (oder SQL2)
- SQL:1999 (SQL3) u.a. mit objektorientierten Erweiterungen
- neu: SQL:2003 u.a. mit XML-Unterstützung

Aufbau von Tabellen

- Ein erstes Beispiel:

autor	titel	seiten	verlag	jahr	auflage	isbn
Hering	Physik für Ingenieure	744	Springer	1999	7	3540661352
Stöcker	Taschenbuch der Physik	1092	Harri Deutsch	2000	4	3817116284
Halliday	Fundamentals of Physics Extended	1198	Wiley	1997	5	471105597
Pitka	Physik - Der Grundkurs	426	Harri Deutsch	1999	1	3817115768

- Datensatz:

  Zeile in einer Tabelle, z.B.

autor	titel	seiten	verlag	jahr	auflage	isbn
Stöcker	Taschenbuch der Physik	1092	Harri Deutsch	2000	4	3817116284

  Beschreibung eines Objekts (**Entität**) durch Werte für die interessierenden Eigenschaften (**Attribute**)

  Attribute haben Wertebereiche (**Domänen**), z.B.

Attribut	Bereich
autor, titel, verlag	Zeichenkette
seiten	ganze Zahl
jahr	gültige Jahreszahl
auflage	kleine ganze Zahl
isbn	10stellige ganze Zahl

  spezieller Wert **NULL** = Hilfswert bei unbekanntem oder undefinierten Wert

- Tabelle:

  Menge von Datensätzen

  Reihenfolge der Datensätze beliebig

  keine doppelten Datensätze (in der Regel)

- Schlüssel:

  jeder Datensatz beschreibt ein festgelegtes Objekt (ein bestimmtes Buch, eine bestimmte Person, Produkt, etc)

  ein oder mehrere Attribute (**Schlüssel**) legen Objekt fest

  Beispiele:

Objektart	Schlüssel
Buch	ISBN
Auto	Kennzeichen
Vorlesungsraum	Gebäudenummer + Raumnummer

Anforderungen an Schlüssel

- Eindeutigkeit (verschiedene Objekte haben verschiedene Schlüssel)
- Definiertheit (jedes Objekt hat einen Schlüsselwert != NULL)
- Konstanz (jedes Objekt hat immer denselben Schlüsselwert)

natürliche Schlüssel häufig ungeeignet

z.B. Vorname und Name

- nicht eindeutig (Hans Müller)
- nicht definiert (direkt nach der Geburt)
- nicht konstant (Namensänderungen, etwa bei Hochzeit)

Aushilfe: künstliche Schlüssel wie ISBN, Kundennummer, Matrikelnummer etc.

Grundlagen von SQL

- Allgemeine Syntax:

  Schlüsselwörter unabhängig von Klein-/Großschreibung

  bei Namen für Tabellen, Attribute etc. Klein-/Großschreibung relevant!

  Befehle über mehrere Zeilen möglich

  Ende eines Befehls mit `;`

  i.f. Großbuchstaben für Schlüsselwörter

- Datenbank in SQL:

  erzeugen mit SQL-Kommando

  ```
  CREATE DATABASE datenbank;
  ```

  danach zum Benutzen "in Datenbank wechseln" mit

  ```
  USE datenbank;
  ```

  erspart bei Tabellen Angabe des vollen Namens `datenbank.tabelle`

  Liste aller vorhandenen Tabellen in der ausgewählten Datenbank:

  ```
  SHOW TABLES;
  ```

  Liste aller vorhandenen Datenbanken mit

  ```
  SHOW DATABASES;
  ```

- Datentypen für Attribute:

  viele verschiedene Typen definiert

  einige Beispiele

Name	Bedeutung
`CHAR(n)`	Zeichenkette mit genau n Zeichen
`VARCHAR(n)`	Zeichenkette variabler Länge mit höchstens n Zeichen
`INTEGER`	ganze Zahl mit Standardbereich
`DECIMAL(p,n)`	Zahl mit p Stellen und n Nachkommastellen
`DATE`	Datum (Format "YYYY-MM-DD")
`YEAR`	Jahreszahl

Arbeiten mit Tabellen

- Erzeugen von Tabellen:

  Syntax

  ```
  CREATE TABLE tabelle (feld1 typ1, feld2 typ2, ...);
  ```

  zusätzliche Parameter bei `CREATE`

`NOT NULL`	darf nicht den Wert NULL haben
`UNIQUE`	Werte für alle Datensätze verschieden
`PRIMARY KEY`	zentraler Zugriffsschlüssel erfordert NOT NULL impliziert UNIQUE
`AUTO_INCREMENT`	automatisches Hochzählen des Index für PRIMARY KEY nur bei Integer-Datentypen

  Beispiel

  ```
  CREATE TABLE Buch
     (autor       varchar(40),
      titel       varchar(255),
      isbn        decimal(10,0) NOT NULL,
      PRIMARY KEY(isbn)
      );
  ```

- Löschen von Tabellen:

  Syntax

  ```
  DROP TABLE tabelle;
  ```

- Information über Tabellen:

  Syntax

  ```
  SHOW COLUMNS FROM tabelle;
  ```

  andere Form

  ```
  DESCRIBE tabelle;
  ```

- Einfügen von Daten:

  Syntax

  ```
  INSERT INTO tabelle (feld1, feld2 ..)
     VALUES(wert1, wert2 ...);
  ```

  Beispiel

  ```
  INSERT INTO Buch (isbn, autor, titel)
      VALUES(1236548792, "Paul Panther", "Der rosa Riese");
  ```

  fehlende Felder bekommen NULL-Werte

Datenbankentwurf

- Beispiel Bücher-Datenbank:

Titel	Autor	Seiten	Verlag	Jahr	Auflage	ISBN	Schlagwörter
Physik für Ingenieure	Hering, Martin, Stohrer	744	Springer	1999	7	3540661352	Physik, Lehrbuch, Mechanik, Quantenmechanik, Thermodynamik
Taschenbuch der Physik	Stöcker	1092	Harri Deutsch	2000	4	3817116284	Physik, Lehrbuch, Mechanik, Quantenmechanik, Thermodynamik
Fundamentals of Physics Extended	Halliday, Resnick, Walker	1198	Wiley	1997	5	471105597	Physik, Lehrbuch, Mechanik, Quantenmechanik, Thermodynamik
Physik - Der Grundkurs	Pitka, Bohrmann, Stöcker, Terlecki	426	Harri Deutsch	1999	1	3817115768	Physik, Lehrbuch, Mechanik, Thermodynamik
Einführung in die Thermodynamik	Cerbe, Hoffmann	NULL	Carl Hanser	1999	12	3446211101	Lehrbuch, Thermodynamik
Thermodynamik für Ingenieure	Langeheinecke, Jany, Sapper	NULL	Vieweg	1999	2	3528147857	Lehrbuch, Thermodynamik
Relationale Datenbanken und SQL	Matthiesen, Unterstein	352	Addison-Wesley	2000	2	3827315581	Lehrbuch, SQL, Datenbank
MySQL & mSQL	Yarger, Reese, King	512	O'Reilly	2000	1	3897211637	SQL, Datenbank, Handbuch

- Probleme mit der Tabelle:

  Felder mit mehreren Einträgen (Autor, Schlagwort)
  - → kann nicht nach einzelnem Autor oder Schlagwort durchsucht werden

  hohe Redundanz
  - gleiche Begriffe an mehreren Stellen (Schlagwörter, Autoren, Verlag)
  - → vergrößerter Speicherbedarf
  - → Gefahr von Inkonsistenzen (z.B. durch Schreibfehler)

  Datenbanken brauchen klaren Entwurf!

- Entity-Relationship-Modell (**ERM**):

  Verfahren zum systematischen Entwurf von Datenbanken

  Entity
  - eindeutig identifizierbares Objekt
  - beschrieben durch eine Menge von Attributen
  - Typ des Objekts = Liste der Attributsnamen und -typen
  - Entitäten ergeben Tabellen

  Vergleich mit objektorientierter Programmierung
  - Tabelle ≙ Klasse
  - Attribut ≙ Datenfeld
  - Datensatz ≙ Objekt

Relationship

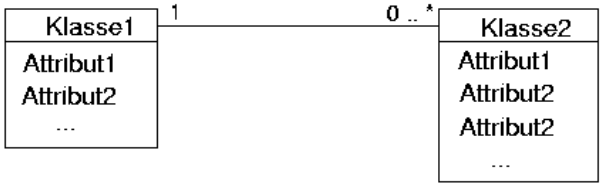
- Beziehung zwischen Entitäten
- kann selbst Eigenschaften (Attribute) haben
- kann zwei oder mehr verschiedene Arten von Entitäten verknüpfen
- kann verschiedene Vielfachheiten haben, z.B.

1:1	ein Objekt A mit einem Objekt B
1:N	ein Objekt A mit mehreren Objekten B
N:M	ein A mit mehreren B und ein B mit mehreren A

- ein Objekt möglicherweise mit keinem anderen verknüpft
- Notation z.B.

  1:0..* = ein Objekt A mit gar keinem, einem oder mehreren B verknüpft

graphische Darstellung z.B. als Klassendiagramm (**UML-Diagramm**)

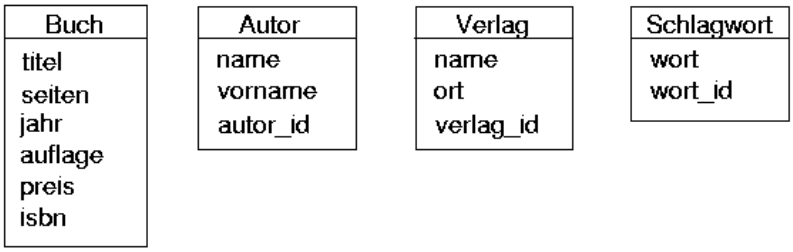


- Anwendung auf Beispiel:

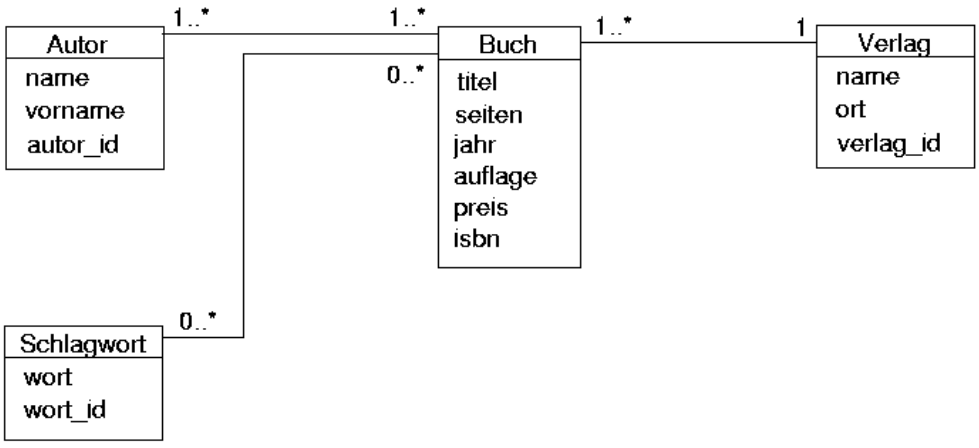
Entitäten sind

- Buch
- Autor
- Verlag
- Schlagwort

Attribute



Beziehungen zwischen den Entitäten



z.B. Beziehung zwischen Buch und Verlag

- ein Buch hat genau einen Verlag (1)
- ein Verlag hat ein oder mehrere Bücher (1..*)

einige Designentscheidungen

- nur Autoren aufgeführt, für die Bücher in der Datenbank existieren
- zu einem Schlagwort kann evtl. noch kein Buch vorhanden sein

- Umsetzung der Relationen:

  unterschiedlich nach Grundtypen 1:1, 1:N, N:M

  bei 1:N
  - Schlüssel der 1-Seite als Attribut zur Tabelle der N-Seite hinzufügen (**Fremdschlüssel**)
  - im Beispiel: Tabelle `Buch` bekommt Attribut `verlag_id`

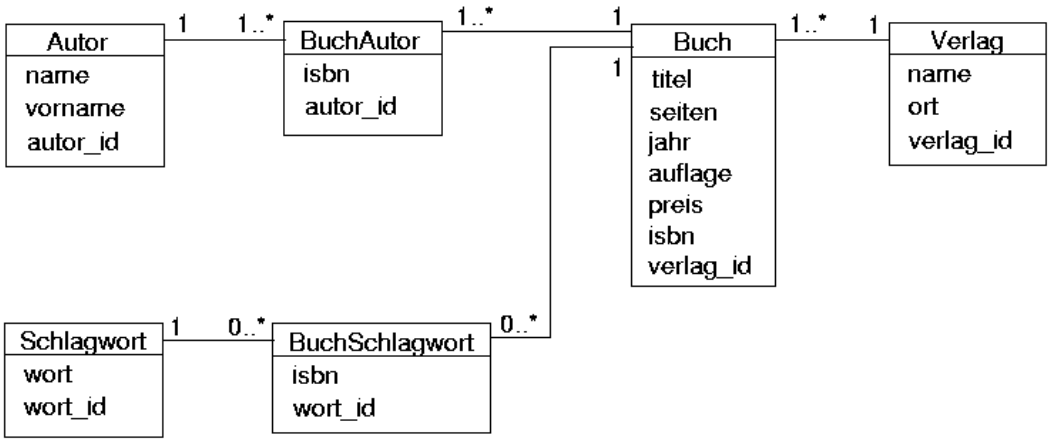
  bei 1:1
  - beide Tabellen zu einer mit allen Attributen zusammenfassen
  - häufig durch Hinzufügen der Attribute einer Tabelle zur anderen (z.B. Telefon-Nr. zu Autor)

  bei N:M
  - neue Tabelle für die Relation einführen (**Verbindungsentität**)
  - enthält Fremdschlüssel beider Partner
  - Primärschlüssel ist häufig Paar der Fremdschlüssel
  - enthält manchmal weitere Attribute

  im Beispiel:
  - neue Tabelle `BuchAutor`
  - Attribute `autor_id` und `isbn`
  - bilden zusammen Primärschlüssel
  - anlegen mit `PRIMARY KEY(autor_id, isbn)`
  - einen Eintrag für jede Autor/Buch-Kombination
  - analog für `BuchSchlagwort`

- Fertige Bücher-Datenbank:

  als Klassendiagramm



  Tabellen mit Beispieldaten

Abfragen in Datenbanken

- Einfache Abfragen
- Abfragen über mehrere Tabellen
- Komplizierte Abfragen

Einfache Abfragen

- `SELECT`-Kommando:

  grundlegendes Kommando für Abfragen (**Queries**)

  sehr vielfältig

  große Zahl an optionalen Klauseln

  Grundabfrage zur Anzeige aller Daten einer Tabelle

  ```
  SELECT * FROM tabelle;
  ```

- Auswahl von Spalten (**Projektion**):

  nicht alle Angaben in einer Tabelle für die aktuelle Abfrage relevant

  Auswahl über Liste von gewünschten Attributen

  ```
  SELECT attribut1, attribut2, .. FROM tabelle;
  ```

  Ausgabe kann sortiert werden mit der Zusatzklausel

  ```
  ORDER BY attribut1, attribut2, ..
  ```

  Default: aufsteigende Reihenfolge, absteigend mit

  ```
  ORDER BY attribut DESC
  ```

  Beispiel

  ```
  SELECT titel, seiten, jahr FROM Buch
     ORDER BY jahr, seiten DESC;
  ```

  Resultat

- Auswahl von Zeilen (**Selektion**):

  mit der `WHERE`-Klausel

  viele gängige Operationen

Vergleiche	`<, >, <=, =, !=`
Logik	`AND OR NOT`
Arithmetik	`+, - *, /, %`
`LIKE`	String-Vergleich mit Wildcards `%` (beliebig viele Zeichen) `_` (genau ein Zeichen)

  Beispiel

  ```
  SELECT titel, seiten, jahr FROM Buch
     WHERE jahr >= 2000;
  ```

  Resultat

  Beispiel

  ```
  SELECT titel, seiten, jahr FROM Buch
     WHERE titel LIKE "%Physi%";
  ```

  Resultat

Abfragen auf NULL

- mit `attribut=NULL` immer leer
- stattdessen mit `attribut IS NULL`

- Funktionen im `SELECT`:

  bei Auswahl-Attributen oder in `WHERE`-Bedingung

  große Anzahl vordefiniert, z.B.

numerische Funktionen	`SIN, LOG`
String-Funktionen	`SUBSTRING, CONCAT, LENGTH, TRIM`
Datums-Funktionen	`NOW, WEEK, DATE_ADD`

  Beispiel: Umrechnung in Dollar

```
SELECT titel, preis/2.04 FROM Buch
   WHERE titel LIKE "%Physics%";
```

  Resultat

- Weitere Anwendungen von `WHERE`:

  Ändern von Daten

```
UPDATE tabelle SET attribut1=wert1, ...
   WHERE bedingung;
```

  Löschen von Daten

```
DELETE FROM tabelle WHERE bedingung;
```

Abfragen über mehrere Tabellen

- `SELECT` mit mehreren Tabellen (**Join**):
  - mehrere Tabellen angeben

    ```
    SELECT attribut1, attribut2, ...
      FROM table1 JOIN table2 ...
    ```

    ohne weitere Einschränkung alle Kombinationen aus allen Tabellen

    Beispiel

    ```
    SELECT titel, name FROM Buch JOIN Verlag;
    ```

    liefert 70 Einträge (10 Bücher x 7 Verlage)
  - Verknüpfung der Tabellen durch Übereinstimmung der Schlüssel

    ```
    SELECT titel, name FROM Buch INNER JOIN Verlag
       ON Buch.verlag_id = Verlag.verlag_id;
    ```

    bei gleichen Attributbezeichnungen Verknüpfung automatisch (**Natural Join**)

    ```
    SELECT titel, name FROM Buch NATURAL JOIN Verlag;
    ```

    Resultat
  - bessere Attributsbezeichnungen bei Joins mit AS, z.B.

    ```
    SELECT titel, name AS verlag
           FROM Buch NATURAL JOIN Verlag
           ORDER BY verlag;
    ```

    Resultat
- Gruppierung:

  weitere Klausel für `SELECT`

  ```
  GROUP BY attribut1, attribut2, ..
  ```

  fasst Datensätze mit gleichen Werten für Attribute zusammen

  übrige (nicht-gleiche) Attribute brauchen Funktionen zum Kombinieren

  wichtigste Funktionen

`COUNT`	Anzahl der zusammengefassten Werte
`SUM`	Summe der jeweiligen Attribute
`AVG`	Durchschnitt
`MAX,MIN`	Maximum, Minimum

- Beispiele:

  Wieviele Bücher gibt es für jedes Jahr, wie groß ist der Gesamtpreis pro Jahr?

  ```
  SELECT COUNT(*), SUM(preis), jahr FROM Buch
       GROUP BY jahr;
  ```

Resultat

Wieviele Bücher gibt es von jedem Verlag?

```
SELECT name as verlag, COUNT(*)
      FROM Buch NATURAL JOIN Verlag
      GROUP BY name;
```

Resultat

Komplizierte Abfragen

- Einige beispielhafte Fragestellungen:
  - Bestimme den Gesamtpreis aller Bücher der einzelnen Verlage.
  - Bestimme zu jedem Schlagwort die Zahl der passenden Bücher
  - Bestimme für jeden Autor alle Bücher, an denen er mitgeschrieben hat.
  - Ermittle die Namen aller Autoren, mit denen Stöcker zusammen ein Buch verfasst hat.
- Vorgehensweise bei komplizierten Abfragen:

  alle benötigten Tabellen (für Attribute und benötigte Relationen) zusammensuchen
  - graphisch: Klassenkästchen mit benötigten Attributen nebeneinander stellen

  benötigte Verknüpfungen über die Fremdschlüssel formulieren
  - graphisch: entsprechende Schlüssel verbinden und mit Relation (=, !=, Wert) beschriften
  - bei = ggf. NATURAL JOIN benutzen

  bei Gruppierung: zu gruppierende Attribute bestimmen, Zusammenfassungsfunktionen bei den anderen
  - graphisch: zu gruppierende Attribute einkreisen, an alle anderen Funktion schreiben

  Feinarbeit: Umbennung von Attributen, Sortierung
- Anwendung bei Beispielfrage 1:

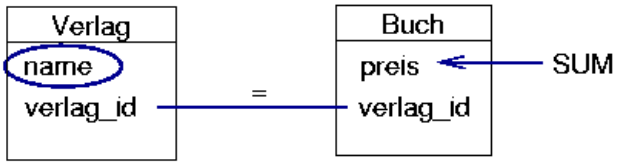
  benötigt werden

Tabelle	Attribute
Verlag	`name`, `verlag_id`
Buch	`preis`, `verlag_id`

  verknüpft wird über Gleichheit der `verlag_id`-Schlüssel (`NATURAL JOIN`)

  gruppiert wird über `name`, über `preis` wird summiert

  graphisch also



  Feinschliff
  - Spalten umnennen in `verlag` und `gesamtpreis`
  - absteigend sortieren nach `gesamtpreis`

  SQL-Kommando

```
SELECT name AS verlag , SUM(preis) AS gesamtpreis
      FROM Buch NATURAL JOIN Verlag
      GROUP BY verlag
      ORDER BY gesamtpreis DESC;
```

  Resultat

- Formulieren der anderen Abfragen als Aufgabe

Graphischer Zugriff mit LibreOffice Base

- Base als Benutzeroberfläche:

  Grundfunktionen
  - Tabelle anlegen
  - Daten eingeben
  - Abfragen

  Formulare
  - bequeme Eingabe und Datenansicht
  - automatische Verknüpfung von Tabellen

  Abfragen
  - mit Assistent erzeugte Select-Kommandos
  - ohne SQL-Wissen nur einfache Abfragen möglich

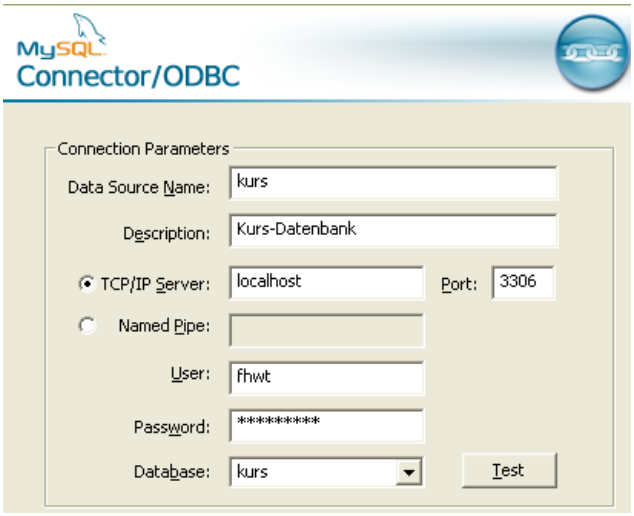
- Verbindung von Base und MySQL:

  drei Treiber verfügbar
  - ODBC (unter Windows verbreitet)
  - JDBC (plattformunabhängig, braucht Java)
  - MySQL Native Connector (bisher nur Linux)

  i.F. ODBC (alternativ JDBC)

- Neue Verbindung aufbauen:

  beim erstem Mal (ODBC)
  - Verbindung zu einer bestehenden Datenbank herstellen: `ODBC`
  - Durchsuchen/Verwalten/Hinzufügen: `MySQL ODBC 5.3 Unicode Driver`
  - Connection Parameters



  - mehrere Male bestätigen
  - `Datenbank/Erweiterte Eigenschaften.../Benannte Parameter durch ? ersetzen`: aktivieren
  - Verbindungsdaten abspeichern als `kurs.odb`

  beim erstem Mal (JDBC)
  - Verbindung zu einer bestehenden Datenbank herstellen: `JDBC`
  - URL der Datenquelle: `jdbc:mysql://192.168.5.251/kurs`

    - JDBC-Treiberklasse: `com.mysql.jdbc.Driver`
    - Benutzername: `fhwt`, Kennwort erforderlich
    - DB anmelden + zum Bearbeiten öffnen
    - Verbindungsdaten abspeichern als `kurs.odb`

  an der PHWT
    - Verbindung zu einer bestehenden Datenbank herstellen: `ODBC`
    - Durchsuchen/Datenquelle `Kurs`
    - `Datenbank/Erweiterte Eigenschaften.../Benannte Parameter durch ? ersetzen`: aktivieren
    - Verbindungsdaten abspeichern als `kurs.odb`

  danach
    - Bestehende Datenbank-Datei öffnen: `kurs`

- Arbeiten mit Tabellen:

  Klicken auf `Tabellen` (links oben)
    - ggf. Passwort eingeben
    - alle Tabellen werden angezeigt

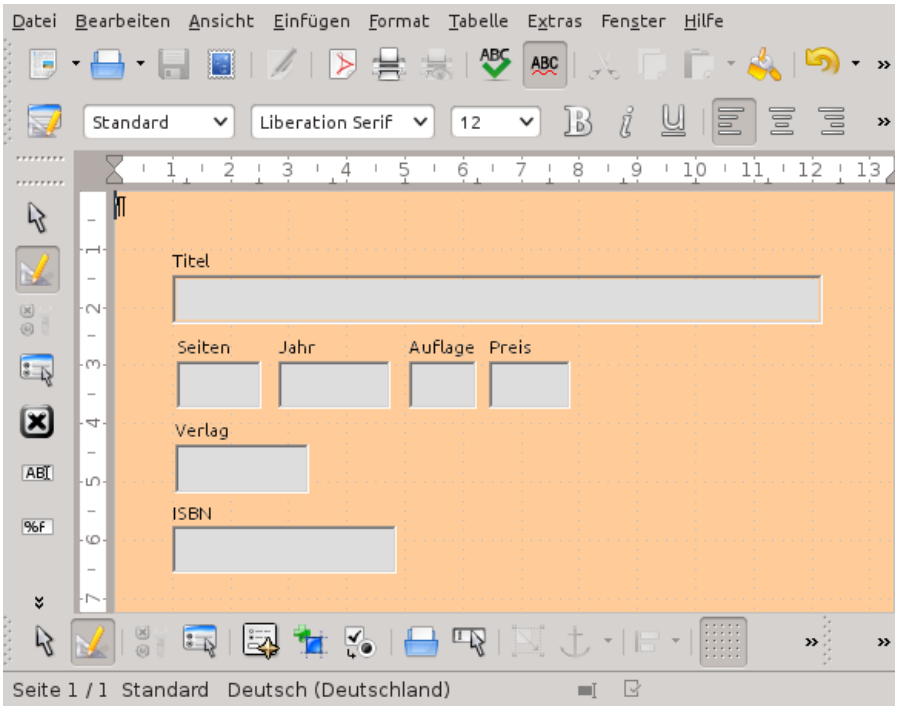
  Doppelklick öffnet Datenfenster
    - tabellarische Übersicht
    - direkte Dateneingabe möglich

  Erstellen von Tabellen (unter `Aufgaben`)
    - möglich, aber buggy
    - besser direkt mit mysql

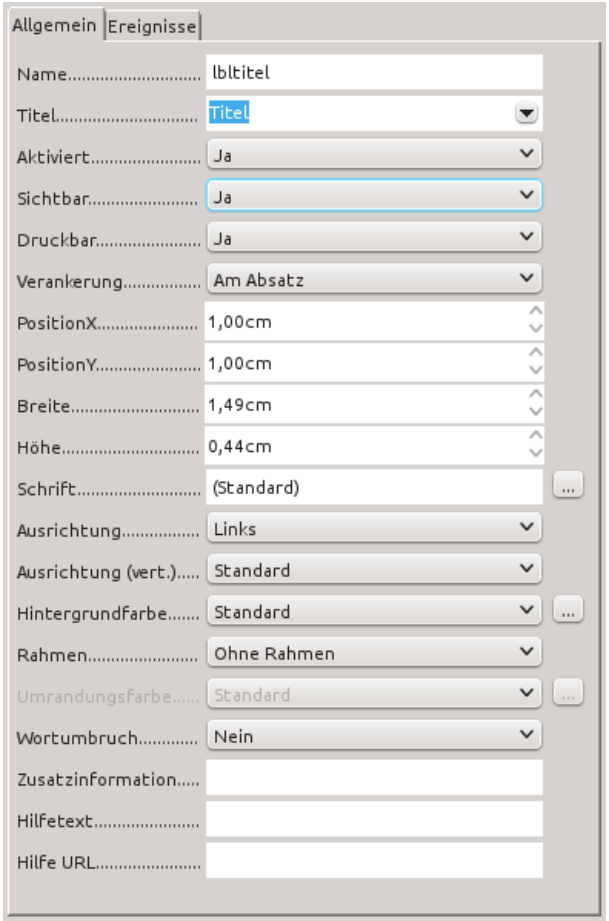
- Formular für einfache Tabelle `Buch`:

  einfache Startvorlage erzeugen
    - links `Formulare` anklicken
    - Assistent verwenden
    - Tabelle `kurs.Buch` und alle Felder auswählen
    - kein Unterformular
    - Anordnung: in Blöcken

  Nachbearbeiten
    - Rechts-Klick `kurs.Buch` / `bearbeiten`
    - Blöcke mit der Maus schöner anordnen



- zum Bearbeiten von Text oder Feld Gruppierung betreten oder aufheben
- alle Eigenschaften eines Feldes mit Doppelklick



- abspeichern

Formular `kurs.Buch` zum Ansehen, Ändern und Löschen von Daten

Nachbearbeiten der Felder

- `Auflage-Feld/Eigenschaften/Formatierung`: Zahl ohne Nachkommastellen und führende Nullen
- ebenso bei ISBN

Sortierung nachträglich einbauen

- Formular-Eigenschaften ( ) öffnen
- `Daten/Sortierung/...`
- `titel/aufsteigend`

- Einbau der 1:N-Verknüpfung Buch - Verlag:

  Ziel
  - Name des Verlags anzeigen
  - gewünschten Verlag aus Liste auswählen

  Verlag einbauen statt `verlag_id`
  - Textfeld löschen
  - Listenfeld einfügen (mit , linke Leiste) → Assistent wird geöffnet
  - `kurs.Verlag` auswählen (Tabelle mit den anzuzeigenden Daten)
  - `name` auswählen (Feld in der Tabelle mit den Daten)
  - `verlag_id/verlag_id` auswählen (Schlüssel zum Verbinden in beiden Tabellen)

  Formular ausprobieren
  - Liste der Verlage wird angezeigt
  - gewählter Wert ist markiert
  - Änderung durch Anklicken eines neuen Verlagsnamens (und speichern)
  - → `verlag_id` in `Buch` wird entsprechend geändert

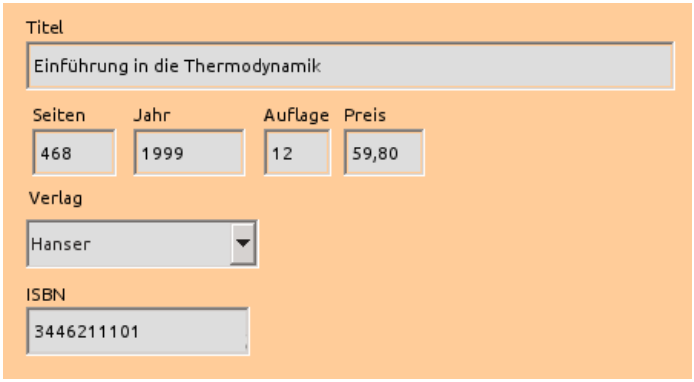
  alternativ ohne Assistent
  - `Textfeld` (Kontext)/`Ersetzen durch/Listenfeld`
  - `Eigenschaften` (Doppelklick)/`Daten`

Datenfeld	`verlag_id`
Eingabe erforderlich	`ja`
Art des Listeninhalts	`SQL`
Listeninhalt	`SELECT name, verlag_id FROM kurs.Verlag`
Gebundenes Feld	`1`

  verschönern
  - `Eigenschaften/Allgemein`

Aufklappbar	`Ja`
Anzahl der Zeilen	`10`

  Ergebnis



- Einbau der N:M-Verknüpfung Buch - Autor:

  Ziel
  - Namen der Autoren anzeigen
  - beliebig Autoren hinzufügen oder löschen

  Unterformular einfügen
  - `kurs.Buch` bearbeiten
  - Formular-Navigator ( , unten) starten
  - `MainForm` (Kontext)/`Neu`/`Formular`

  Unterformular bearbeiten
  - `Formular` (Kontext)/`Eigenschaften`

Allgemein/Name	`Autoren`
Daten/Inhalt	`kurs.BuchAutor`
Daten/Verknüpfen von/...	`isbn` in BuchAutor und Buch

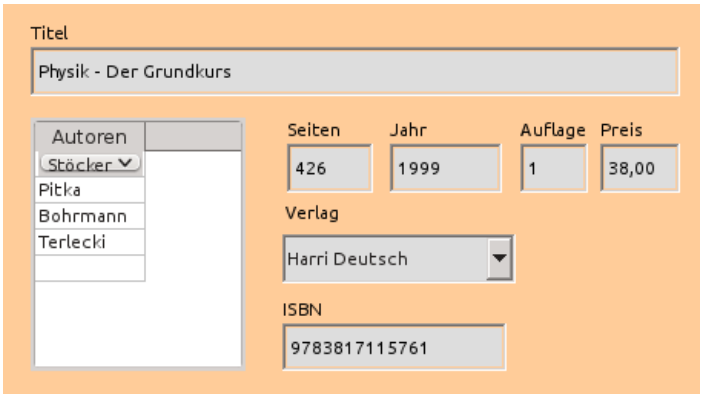
  Hinzufügen eines Tabellen-Steuerelements
  - Platz für Autorenliste schaffen
  - im Formular-Navigator Autoren anklicken
  - weitere Steuerelemente ()/Tabellen-Steuerelement () anklicken
  - Bereich im Formular aufziehen
  - Assistenten abbrechen
  - Doppelklick auf Feld → Fenster `Eigenschaften`/`Allgemein` kommt

Symbolleiste Navigation	`Nein`
Datensatzmarkierer	`Nein`

  Listenfeld einfügen
  - `Tabellenkopf` (Kontext)/`Spalte einfügen`/`Listenfeld`
  - `Listenfeld1` (Kontext)/`Spalte...`
  - Allgemein/Titel: `Autoren`
  - Daten

Datenfeld	`autor_id`
Art des Listeninhalts	`SQL`
Listeninhalt	`SELECT name, autor_id FROM kurs.Autor ORDER BY name`
Gebundenes Feld	`1`

  fertiges Ergebnis


Titel
Physik - Der Grundkurs
Autoren
Stöcker
Pitka
Bohrmann
Terlecki
Seiten
426
Jahr
1999
Auflage
1
Preis
38,00
Verlag
Harri Deutsch
ISBN
9783817115761

Aufgaben

Bei den Aufgaben handelt es sich um kleine Projekte, die jeweils in Gruppen von 3 bis 4 Personen bearbeitet werden sollen.

- Aufgabe 1
- Aufgabe 2
- Aufgabe 3
- Aufgabe 4
- Aufgabe 5
- Aufgabe 6
- Aufgabe 7
- Aufgabe 8

Aufgabe 1

- Erstellen Sie in Gruppen von drei bis vier Personen eine kleine Homepage aus mehreren Seiten zu einem Thema Ihrer Wahl. Sie soll im Semester durch diverse Elemente erweitert werden; wählen Sie also ein hinreichend ergiebiges Thema.
- Einige Vorschläge bei Ideenmangel:
  - Der Kurs MB999 an der FHWT
  - Wohnen in Diepholz (oder sonstwo)
  - Meine Briefmarkensammlung
  - Saurier in Museen der Umgebung
  - Haltung des Zwergkrallenfroschs
  - **nicht**: Meine Lieblingsautos (hatte ich gefühlte 297 mal!)
- Benutzen Sie an Software und Hardware, was Ihnen an der FH oder privat bzw. im Internet zur Verfügung steht.
- Einhalten des HTML5-Standards ist obligatorisch!

Aufgabe 2

- Ergänzen Sie Ihre Seiten um Elemente zur Kommunikation mit dem Benutzer. Setzen Sie geeignete Formular-Elemente ein und wählen Sie zum Testen als Wert des `action`-Feldes immer meine allgemeine echo-Seite

Aufgabe 3

- Entfernen Sie alle Stil-Attribute aus Ihren bisherigen HTML-Seiten und erstellen Sie stattdessen ein globales Stylesheet für Ihr Projekt. Versehen Sie dabei alle Seiten mit einem ansprechenden einheitlichen Layout.

Aufgabe 4

- Überprüfen Sie, ob die von Ihnen eingesetzten Bilder zweckdienliche Formate verwenden. Versuchen Sie, durch geeignete Bearbeitung der Bilder bzw. der Format-Eigenschaften die Größe der Bilder möglichst weitgehend zu reduzieren. Erwägen Sie auch, kleinere Vorschaubilder einzusetzen.
- Ergänzen Sie Ihre Seiten durch multimediale Elemente (zumindest ein Klang- und ein Video-Element). Achten Sie darauf, dass die Seiten sowohl im Firefox als auch im Internet Explorer angezeigt werden; installieren Sie ggf. benötigte Plugins.
- Erstellen Sie weitere Multimedia-Dateien (keine Bilder) oder verändern Sie aus dem Netz geladene Dateien für Ihre Zwecke. Suchen Sie dafür geeignete Programme.

Aufgabe 5

- Besorgen Sie sich ein eigenes Email-Zertifikat, entweder von einem entsprechenden Provider (z. B. Comodo) im Netz oder mit Hilfe von GnuPG.
- Importieren Sie Ihr Zertifikat in Ihr Email-Programm (z.B. Thunderbird) und tauschen Sie zunächst signierte, danach auch verschlüsselte Emails mit Kommilitonen aus.

Aufgabe 6

- Entwickeln Sie eine Datenbank, die Bestellvorgänge enthält. Eine Bestellung geht von einem Kunden aus und enthält eine Menge von Artikeln. Im einzelnen:

  a. Überlegen Sie sich die auftretenden Entitäten und Attribute und stellen Sie ein Klassendiagramm auf, das die Beziehungen aufzeigt.

  b. Entwickeln Sie daraus eine Menge von Tabellen, wobei ggf. einige Tabellen aus Relationen dazukommen.

  c. Implementieren Sie die Tabellen und fügen Sie einige charakteristische Datensätze ein.

  d. Formulieren Sie einige wichtige Fragen an die Datenbank und entwickeln Sie die zugehörigen SQL-Kommandos.

Aufgabe 7

- Entwickeln Sie SQL-Kommandos zur Beantwortung der folgenden Fragestellungen und testen Sie sie an der Beispiel-Datenbank:

  a. Bestimmen Sie zu jedem Schlagwort die Zahl der passenden Bücher.

  b. Bestimmen Sie für jeden Autor die Titel aller Bücher, an denen er mitgeschrieben hat.

  c. Erstellen Sie eine Liste aller Autoren und der Verlage, bei denen sie Bücher veröffentlicht haben, sortiert nach Namen der Autoren. Achten Sie darauf, dass keine Einträge mehrmals vorkommen.

  d. Ermitteln Sie die Namen aller Autoren, die Bücher zum Schlagwort "Thermodynamik" geschrieben haben, mit bzw. ohne Angabe der Buchtitel, jeweils sortiert nach Autornamen.

  e. Ermitteln Sie die Namen aller Autoren, mit denen Stöcker zusammen ein Buch verfasst hat.

- Lösung

Aufgabe 8

- Laden Sie die Datei kurs.sql herunter, ersetzen Sie den Platzhalter `XXX` durch den eigenen Datenbanknamen `VornameNachname` (ohne Leer- und Sonderzeichen) und legen Sie damit die entsprechenden Tabellen an.
- Erzeugen Sie Formulare für Autor-, Verlag- und Schlagwort-Tabellen. Fügen Sie keine Eingabefelder für die Primärschlüssel hinzu - wegen `auto_increment` werden neue Werte automatisch erzeugt.
- Erweitern Sie das Buch-Formular um eine bequeme Schlagwort-Eingabe.
- Verwenden Sie Ihre Formulare, um folgende Daten einzugeben:
  - Günter Cerbe, Gernot Wilhelms: Technische Thermodynamik
    Hanser Verlag, 17. Auflage 2013, 545 Seiten, 29,99 €
    ISBN: 978-3446436381
    Schlagworte: Lehrbuch, Thermodynamik
  - P. Junglas: cliXX Webapplikationen.
    Verlag Harri Deutsch, 2004, 209 Seiten, 24.80 €
    ISBN: 9783808554661
    Schlagworte: Webentwicklung, Lehrbuch
  - Stefan Münz, Wolfgang Nefzger: HTML-Handbuch
    Franzis Verlag, 2004, 1300 Seiten, 40,00 €
    ISBN: 978-3772370069
    Schlagworte: Webentwicklung, Lehrbuch

Anhang

Literatur

1. Gull/Münz: HTML5 Handbuch
   Franzis-Verlag, 2014, ISBN: 978-3645603454
   auch online verfügbar!

2. Beschreibung des HTML4-Standards beim W3-Consortium
   http://www.w3.org/TR/html5/

3. Beschreibung des CSS-Standards
   http://www.w3.org/Style/CSS/

4. E. Meyer: Eric Meyer on CSS
   New Riders Publishing, 2002, ISBN: 978-3827268952

5. P. Henning: Taschenbuch Multimedia
   Hanser Fachbuch, 4. Auflage 2007, ISBN: 978-3446409712

6. M. Kofler: MySQL - Einführung, Programmierung, Referenz
   Addison-Wesley, 2007, ISBN: 978-3827326362

7. Marco Emrich: Datenbanken & SQL für Einsteiger:
   CreateSpace Independent Publishing Platform, 2013, ISBN: 978-1492951049

8. P. Junglas: cliXX Webapplikationen.
   Verlag Harri Deutsch, 2004, ISBN: 978-3808554661

Nachweise

- apollo15.ogg basiert auf Video Apollo_15_feather_and_hammer_drop.ogg aus der freien Enzyklopädie Wikipedia und ist gemeinfrei. Es stammt von der NASA.
- Der Clip Vivaldi-Winter und seine Bearbeitungen basieren auf der Multimedia-Datei 11_-_Vivaldi_Winter_mvt_2_Largo_-_John_Harrison_violin.ogg aus der freien Enzyklopädie Wikipedia und steht unter der Creative Commons Attribution ShareAlike 1.0 License. Der Rechteinhaber ist John Harrison. Es handelt sich um eine Einspielung des Concerto No. 4 in f-Moll, Op. 8, RV 297, "L'inverno" (Winter) - 2. Satz: Largo, aus "Die vier Jahreszeiten" von Antonio Vivaldi. Die Einspielung erfolgt am 6. 2. 2000 live in der Wiedemann Recital Hall, Wichita State University. Ausführende: John Harrison - Violine / Robert Turizziani - Leitung / Wichita State University Chamber Players.
- Bild 22 basiert auf dem Bild Ear-anatomy.png aus der freien Enzyklopädie Wikipedia und steht unter der GNU-Lizenz für freie Dokumentation. Der Urheber des Bildes ist User:Bemoeial2.
- Bild 03 basiert auf dem Bild Cd_Niederdruck_Spektrum.png aus der freien Enzyklopädie Wikipedia und ist gemeinfrei. Der Urheber des Bildes ist User:Herbertweidner.
- Bild 04 basiert auf dem Bild Cone-response.png aus der freien Enzyklopädie Wikipedia und steht unter der GNU-Lizenz für freie Dokumentation. Der Urheber des Bildes ist User:Maxim Razin. Es basiert auf Bowmaker J.K. and Dartnall H.J.A., "Visual pigments of rods and cones in a human retina." J. Physiol. 298: pp501-511 (1980).

RSA-Algorithmus zum Ausprobieren

Hier kann man den RSA-Algorithmus an einfachen Beispielen ausprobieren.

Achtung:

- Zahlbereich ist nur der normaler JavaScript-Werte
- Eingabefehler werden nur rudimentär abgeprüft

Eingabe der Primzahlen:	p =	223	q =	251
Berechnen ergibt	n =		m =	

Eingabe des öffentlichen Schlüssels (teilerfremd zu m):	e =	23
Berechnen ergibt privaten Schlüssel	d =	

Verschlüsseln von	b =	
Berechnen ergibt	c =	

Entschlüsseln von	c =	
Berechnen ergibt	b =	

HTML-Beispiele

- simple.html (HTML-Code)
- text.html (HTML-Code)
- block.html (HTML-Code)
- list.html (HTML-Code)
- table.html (HTML-Code)
- forms1.html (HTML-Code)
- forms2.html (HTML-Code)
- multimedia.html (HTML-Code)

simple.html

```
<!doctype html>
<html>
  <head>
    <title>Praktische Informatik 1</title>
    <meta charset="utf-8">
  </head>

  <body style="background-color:#0fffa7;color:#2339ff;">
    <h1>Die kleine Homepage</h1>

    <p>Jeder fängt mal klein an - so auch diese Seite.</p>

    <p style="color:#ff2080;">
    Über Geschmack lässt sich bekanntlich <em>nicht</em>
    streiten. Völlig unbestreitbar ist aber, dass ich bei der Gestaltung
    dieser Seite keinen bewiesen habe.</p>

    <p>Eine meiner Lieblingsseiten zeigt <a href="http://heritage.stsci.edu/">
    das Schönste vom Hubble-Teleskop</a>.
    </p>

    <p><img src="../images/barspec3.gif" alt="Ruler"></p>

    <address>Viele Grüße von
    <a href="mailto:peter@peter-junglas.de">Peter</a></address>
  </body>
</html>
```

text.html

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Textelemente</title>
  </head>

  <body>
    <h1>Textelemente</h1>

    <p>Die richtige <em>Betonung</em> ist wichtig, vor allem bei diesem
    berühmten Gedicht:</p>
    <p>
      <cite> Zu Dionys, dem Tyrannen schlich Damon, den Dolch im Gewande</cite>
    </p>

    <p>Weniger wichtig ist die Betonung bei diesem Werk der Prosa-Literatur:</p>
    <p>
      <code>for (i=0;i&amp;lt;100;i++) {System.out.println(&amp;quot;Du sollst dem
       Lehrer zuhören(&amp;quot;);}</code>
    </p>

     <p>Im folgenden altgriechischen Gedicht liegt die Betonung ganz klar auf
     den Höhen und Tiefen</p>
     <p><cite> a<sup>2</sup> + b<sup>2</sup> = c<sup>2</sup></cite></p>
  </body>
</html>
```

block.html

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Absatzelemente</title>
  </head>

  <body>
    <h1>Absatzelemente</h1>

    <h2>Deutschstunde (von Heinz Friedlich Fr&uuml;hling)</h2>

<p>Wer wagt es, Rattersmann oder Knipp,<br>
       zu schlauchen in diesen Tund?</p>
    <p> es folgt ... <br><br><br><br><br>
        g&auml;hnende Lehre</p>

    <h2>Sportstunde</h2>
    <p>Zum Aufwachen ein wenig Gymnastik:</p>
    <p style="text-align:right;">Wir drehen den Kopf alle mal nach links,</p>
    <p style="text-align:center;"> dann schauen wir nach vorne.</p>
    <p style="text-align:left;">und schliesslich nach rechts.</p>

    <h2>Informatik</h2>
    <p>Wir wiederholen vom letzten Mal:</p>
     <pre>
     for (i=0; i &lt; 100; i++) {
        System.out.println(&quot;Du sollst dem Lehrer zuh&ouml;ren&quot;);
     }
     </pre>
  </body>
</html>
```

list.html

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Praktische Informatik 1</title>
  </head>

  <body>

    <h1>Listen</h1>

    <p>Liste mit disc:</p>
    <ul style="list-style-type:disc">
      <li>Taschentuch</li>
      <li>Gebetbuch</li>
      <li>Kamm</li>
    </ul>

    <p>Liste mit square:</p>
    <ul style="list-style-type:square">
      <li>Kinder</li>
      <li>Küche</li>
      <li>Kirche</li>
    </ul>

    <p>gemischte Liste:</p>
    <ul>
      <li style="list-style-type:square">Welt</li>
      <li style="list-style-type:none">Weites</li>
      <li style="list-style-type:circle">Warten</li>
    </ul>

    <p>Zählen mit Buchstaben:</p>
    <ol style="list-style-type:lower-alpha">
      <li>Schafft</li>
      <li>Arbeits-</li>
      <li>Plätze</li>
    </ol>

    <p>Und jetzt wirds verrückt:</p>
    <ol>
      <li style="list-style-type:decimal">Ich</li>
      <li style="list-style-type:lower-roman">Bin</li>
      <li style="list-style-type:upper-alpha">Müde</li>
    </ol>
  </body>
</html>
```

table.html

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Praktische Informatik 1</title>
  </head>

  <body>
    <h1>Tabellen</h1>

    <p>Eine Tabelle, die sich breit macht:</p>
    <table style= "border:12px solid black; padding:20px; border-spacing:10px;">
      <tr>
        <td> 1. Zeile, 1. Spalte</td>
        <td> 1. Zeile, 2. Spalte</td>
      </tr>
      <tr>
        <td> 2. Zeile, 1. Spalte</td>
        <td> 2. Zeile, 2. Spalte</td>
      </tr>
    </table>

    <p>Eine Tabelle mit Ordnung</p>
    <table style="border:2px solid black; width:60%">
      <tr>
        <td style="border:1px solid black; text-align:left">links</td>
        <td style="border:1px solid black; text-align:left">rechts</td>
      </tr>
      <tr>
        <td style="border:1px solid black; height:30px;
     text-align:center">links</td>
        <td style="border:1px solid black; text-align:center">rechts</td>
      </tr>
      <tr>
        <td style="border:1px solid black; height:50px;
     text-align:right">links</td>
        <td style="border:1px solid black; text-align:right">rechts</td>
      </tr>
    </table>

    <p>Eine unauffällige Tabelle mit Höhen und Tiefen</p>
    <table style="width:60%">
      <tr style="height:50px;">
        <td style="vertical-align:top;">Schau</td>
        <td style="vertical-align:middle;">nach</td>
        <td style="vertical-align:bottom;">rechts:</td>
        <td></td>
      </tr>
      <tr>
        <td>wohin?</td>
        <td>wohin?</td>
        <td>wohin?</td>
        <td> <em>Ach da!</em> </td>
      </tr>
      <tr style="height:50px;">
        <td style="vertical-align:bottom;">Schau</td>
        <td style="vertical-align:middle;">nach</td>
```

```
        <td style="vertical-align:top;">rechts:</td>
        <td></td>
      </tr>
    </table>
  </body>
</html>
```

forms1.html

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Praktische Informatik 1</title>
  </head>
  <body style="background-color:#d2ffc8">
    <h1>Einfaches Formular</h1>

    <p>Willkommen zum Online-Kurs. Bitte melden Sie sich an:</p>

    <form method="post" action="http://www.peter-junglas.de/fh/vorlesungen/praktinfMB1/html/echo.php">
      <p>
 Benutzer: <input name="user" size="20" value="blabla"><br>
 Passwort: <input type="password" name="passwort" size="20"><br>
 <button>Anmelden</button>
 <button type="reset">Zurücksetzen</button>
      </p>
    </form>

  </body>
</html>
```

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Praktische Informatik 1</title>
  </head>
  <body style="background-color:#d2ffc8">
  <h1>Java - die Klausur</h1>

  <p>Willkommen zur Java-Klausur. Bitte beantworten Sie die folgenden
  Fragen:</p>

    <form method="post" action="http://www.peter-junglas.de/fh/vorlesungen/praktinfMB1/html/echo.php">
    <table style="border-spacing:10px;">
      <tr style="vertical-align:top;">
 <td>Welche Eigenschaften hat Java:<br>
          <select name="java_eigenschaften[]" size="3" multiple>
            <option>objektorientiert</option>
            <option>leicht verständlich</option>
            <option>modern</option>
            <option>internet-fähig</option>
            <option>weit verbreitet</option>
          </select></td>

 <td>Wie wird ein Java-Programm (normalerweise) übersetzt:<br>
          <select name="compiler" size="1">
     <option>mit einem Java-Compiler in Maschinensprache</option>
            <option>mit einem Java-Compiler in einen universellen Bytecode</option>
            <option>zeilenweise von einem Interpreter</option>
            <option>gar nicht, Java ist eine Skriptsprache</option>
            <option>gar nicht, der Prozessor versteht Java direkt</option>
          </select></td>
      </tr>

      <tr style="vertical-align:top;">
        <td>Wie lautet die erste aufgerufene Methode eines Java-Programms:<br>
          <label>
     <input type="radio" name="methode" value="Program">program
          </label> <br>
          <label>
            <input type="radio" name="methode" value="System">system
          </label> <br>
          <label>
            <input type="radio" name="methode" value="Main">main
          </label>
        </td>

        <td>Wie gibt man in Java eine Zeichenkette s aus:<br>
          <label>
            <input type="radio" name="output" value="cpp">cout << s;
          </label> <br>
          <label>
            <input type="radio" name="output" value="java">System.out.print(s);
          </label> <br>
          <label>
            <input type="radio" name="output" value="fortran">write (2,'(A)'), s
          </label> <br>
          <label>
            <input type="radio" name="output" value="c">printf("%s", s);
          </label>
        </td>
      </tr>

      <tr style="vertical-align:top;">
        <td>Welche der folgenden Klassen sind in Java vordefiniert: <br>
          <label>
     <input type="checkbox" name="klassen[]" value="1">System
          </label><br>
          <label>
     <input type="checkbox" name="klassen[]" value="2">Input
          </label><br>
          <label>
     <input type="checkbox" name="klassen[]" value="3">Double
```

```
          </label><br>
          <label>
     <input type="checkbox" name="klassen[]" value="4">Complex
          </label><br>
          <label>
     <input type="checkbox" name="klassen[]" value="5">Image
          </label>
        </td>

        <td>Schreiben Sie eine Java-Methode, die Fibonacci-Zahlen
     berechnet:<br>
          <textarea name="java_programm" rows="10" cols="40">
int fibo(int n) {
   // Ihr Text hier
}
          </textarea>
        </td>
 </tr>

      <tr style="vertical-align:top;">
 <td colspan="2" style="text-align:center">
   <button><strong>Absenden</strong></button>
 </td>
      </tr>
    </table>
  </form>

  </body>
</html>
```

multimedia.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Praktische Informatik 1</title>
  </head>
  <body>
    <h1>Kino mit Plugin</h1>

    <p><video src="../multimedia/apollo15.ogg"
              width="768" height="576" controls>
       </video></p>

    <p>Ein historischer Versuch bei der Apollo 15-Mission</p>
  </body>
</html>
```

CSS-Beispiele

- austen.css (cssdemo.html)
- ohne Stylesheet (cssdemo01.html)
- austen02.css (cssdemo02.html)
- austen03.css (cssdemo03.html)
- austen04.css (cssdemo04.html)
- austen05.css (cssdemo05.html)
- austen06.css (cssdemo06.html)

```
body {
  font-family: Verdana, Arial, Helvetica, sans-serif;
  color: black;
  background: white;
  margin: 0px;
}
table {
  width: 100%;
  margin: 0px;
  border-spacing: 0px;
}
table td {
  padding: 0px;
  border-width: 0;
  vertical-align: top;
}
a:link {
  color: #660000;
}
a:visited {
  color: #990000;
}
h1 {
  font-size: 180%;
  font-weight: bold;
  color: #602020;
  border-bottom: 3px solid #804040;
  padding-bottom: 2px;
}

td#title {
  vertical-align: middle;
  background: url(../images/bgpaper.jpg);
  color: #501000;
  font: bold 600% "Comic Sans MS", Verdana, sans-serif;
  padding-left: 5px;
}
td#portrait {
  width: 253px;
}
td#portrait img {
  vertical-align: bottom;   /* Unterlänge des "Buchstabens" muss weg */
}

#content-top td {
  vertical-align: middle;
  color: white;
  font-weight: bold;
  padding: 0.1em 0.2em 0;
}
td#content-left {
  background: #501000;
  font: bold 115% Arial, Helvetica, Verdana, sans-serif;
}
td#content-links {
  background: #997753;
  font-size: 85%;
}
td#content-links a:link {
  color: white;
  padding: 10px;
}
td#content-links a:visited {
  color: gray;
  padding: 10px;
```

```
}

td#leftside {
  width: 120px;
  background: #f0e0d0;
}
td#leftside ul {
  list-style-type: none;
  list-style-position: inside;
  margin-top: 3px;
  padding: 0px;
}
td#leftside li {
  font-size: 85%;
  border-bottom: 1px solid #A98763;
  padding: 0 0 1px 4px;
}

td#content {
  padding: 17px 42px;
}
td#content p {
  font: 10pt Arial, Helvetica, sans-serif;
}

td#rightside {
  width: 150px;
}
td#rightside h3 {
  text-align: center;
  font-size: 85%;
  background: #774411;
  color: white;
  padding: 2px;
  margin: 0px;
}
td#rightside td {
  font-size: 66%;
  padding: 1px 3px 1px 1px;
}

tr#footer td {
  text-align: center;
  vertical-align: middle;
  font-size: 66%;
  border-top: 3px solid #d6c3ab;
  padding: 0.2em;
}
td#feedback {
  background: #EFE1D1;
}
tr#footer td#author {
  font-size: 85%;
}
tr#footer td#date {
  text-align: right;
  font-style: italic;
  color: #999;
}

div#cite {
  border: 3px solid #804040;
  background: #EBDAC6;
  text-align: center;
  margin-top: 1.5em;
  margin-right:3px;
  padding: 8px;
  font-size: 66%;
```

```
}
div#cite h4 {
  margin: 0px;
}

div.infobox {
  float: right;
  width: 140px;
  color: #A09080;
  border: solid #908070;
  border-width: 7px 0;
  font: bold 1em Arial, Helvetica, Verdana, sans-serif;
  padding: 3px 2px;
  margin: 1px 70px 1px 7px;
}

h4.rightside-head {
  background: #D6B58C;
  text-align: center;
  font-weight: bold;
  font-size: 66%;
  margin: 0px;
  padding: 1px 0 1px 0;
}
tr.even td {
  background: #F7F0E7;
}
tr.odd td {
  background: white;
}
td.preis {
  color: #660;
  width: 5em;
  text-align: right;
}
```

austen02.css

```
table {border: 2px solid red; margin: 3px;}
td {border: 1px dotted red; padding: 2px;}
ul {border: 2px solid blue;}
h1, h2, h3, h4, h5, h6, h7 {border: 1px solid green;}
div {border: 1px solid black;}
span {border: 1px dotted black;}
```

austen03.css

```
body {
  font-family: Verdana, Arial, Helvetica, sans-serif;
  color: black;
  background: white;
  margin: 0px;
}
table {
  width: 100%;
  margin: 0px;
  border-spacing: 0px;
}
table td {
  padding: 0px;
  border-width: 0;
  vertical-align: top;
}
a:link {
  color: #660000;
}
a:visited {
  color: #990000;
}
h1 {
  font-size: 180%;
  font-weight: bold;
  color: #602020;
  border-bottom: 3px solid #804040;
  padding-bottom: 2px;
}
```

austen04.css

```
body {
  font-family: Verdana, Arial, Helvetica, sans-serif;
  color: black;
  background: white;
  margin: 0px;
}
table {
  width: 100%;
  margin: 0px;
  border-spacing: 0px;
}
table td {
  padding: 0px;
  border-width: 0;
  vertical-align: top;
}
a:link {
  color: #660000;
}
a:visited {
  color: #990000;
}
h1 {
  font-size: 180%;
  font-weight: bold;
  color: #602020;
  border-bottom: 3px solid #804040;
  padding-bottom: 2px;
}

td#title {
  vertical-align: middle;
  background: url(../images/bgpaper.jpg);
  color: #501000;
  font: bold 600% "Comic Sans MS", Verdana, sans-serif;
  padding-left: 5px;
}
td#portrait {
  width: 253px;
}
td#portrait img {
  vertical-align: bottom;   /* Unterlänge des "Buchstabens" muss weg */
}
```

austen05.css

```
body {
  font-family: Verdana, Arial, Helvetica, sans-serif;
  color: black;
  background: white;
  margin: 0px;
}
table {
  width: 100%;
  margin: 0px;
  border-spacing: 0px;
}
table td {
  padding: 0px;
  border-width: 0;
  vertical-align: top;
}
a:link {
  color: #660000;
}
a:visited {
  color: #990000;
}
h1 {
  font-size: 180%;
  font-weight: bold;
  color: #602020;
  border-bottom: 3px solid #804040;
  padding-bottom: 2px;
}

td#title {
  vertical-align: middle;
  background: url(../images/bgpaper.jpg);
  color: #501000;
  font: bold 600% "Comic Sans MS", Verdana, sans-serif;
  padding-left: 5px;
}
td#portrait {
  width: 253px;
}
td#portrait img {
  vertical-align: bottom;   /* Unterlänge des "Buchstabens" muss weg */
}

#content-top td {
  vertical-align: middle;
  color: white;
  font-weight: bold;
  padding: 0.1em 0.2em 0;
}
td#content-left {
  background: #501000;
  font: bold 115% Arial, Helvetica, Verdana, sans-serif;
}
td#content-links {
  background: #997753;
  font-size: 85%;
}
td#content-links a:link {
  color: white;
  padding: 10px;
}
td#content-links a:visited {
  color: gray;
  padding: 10px;
```

```
}
```

austen06.css

```
body {
  font-family: Verdana, Arial, Helvetica, sans-serif;
  color: black;
  background: white;
  margin: 0px;
}
table {
  width: 100%;
  margin: 0px;
  border-spacing: 0px;
}
table td {
  padding: 0px;
  border-width: 0;
  vertical-align: top;
}
a:link {
  color: #660000;
}
a:visited {
  color: #990000;
}
h1 {
  font-size: 180%;
  font-weight: bold;
  color: #602020;
  border-bottom: 3px solid #804040;
  padding-bottom: 2px;
}

td#title {
  vertical-align: middle;
  background: url(../images/bgpaper.jpg);
  color: #501000;
  font: bold 600% "Comic Sans MS", Verdana, sans-serif;
  padding-left: 5px;
}
td#portrait {
  width: 253px;
}
td#portrait img {
  vertical-align: bottom;   /* Unterlänge des "Buchstabens" muss weg */
}

#content-top td {
  vertical-align: middle;
  color: white;
  font-weight: bold;
  padding: 0.1em 0.2em 0;
}
td#content-left {
  background: #501000;
  font: bold 115% Arial, Helvetica, Verdana, sans-serif;
}
td#content-links {
  background: #997753;
  font-size: 85%;
}
td#content-links a:link {
  color: white;
  padding: 10px;
}
td#content-links a:visited {
  color: gray;
  padding: 10px;
```

```
}

td#leftside {
  width: 120px;
  background: #f0e0d0;
}
td#leftside ul {
  list-style-type: none;
  list-style-position: inside;
  margin-top: 3px;
  padding: 0px;
}
td#leftside li {
  font-size: 85%;
  border-bottom: 1px solid #A98763;
  padding: 0 0 1px 4px;
}

td#content {
  padding: 17px 42px;
}
td#content p {
  font: 10pt Arial, Helvetica, sans-serif;
}

td#rightside {
  width: 150px;
}
td#rightside h3 {
  text-align: center;
  font-size: 85%;
  background: #774411;
  color: white;
  padding: 2px;
  margin: 0px;
}
td#rightside td {
  font-size: 66%;
  padding: 1px 3px 1px 1px;
}

tr#footer td {
  text-align: center;
  vertical-align: middle;
  font-size: 66%;
  border-top: 3px solid #d6c3ab;
  padding: 0.2em;
}
td#feedback {
  background: #EFE1D1;
}
tr#footer td#author {
  font-size: 85%;
}
tr#footer td#date {
  text-align: right;
  font-style: italic;
  color: #999;
}
```

Multimedia-Beispiele

- Video Apollo 15
- Clip Vivaldi-Winter
- Clip 01
- Clip 02
- Clip 03
- Clip 04
- Clip 05
- Clip 06
- Clip 07
- Clip 08
- Clip 09
- Clip 10
- Applet "Additive Farbmischung"
- Applet "Subtraktive Farbmischung"

SQL-Beispiele

Ergebnisse der SQL-Abfragen für die Beispiel-Datenbank

- Beispiel 1

  Abfrage

  ```
  SELECT titel, seiten, jahr FROM Buch
    ORDER BY jahr, seiten DESC;
  ```

  Ergebnis

titel	seiten	jahr
Grundlagen der Gastechnik	515	NULL
Fundamentals of Physics Extended	1198	1997
Physik für Ingenieure	744	1999
Einführung in die Thermodynamik	468	1999
Physik - Der Grundkurs	426	1999
Thermodynamik für Ingenieure	NULL	1999
Taschenbuch der Physik	1092	2000
MySQL & mSQL	512	2000
Thermodynamics and Statistical Mechanics	463	2000
Relationale Datenbanken und SQL	352	2000

- Beispiel 2

  Abfrage

  ```
  SELECT titel, seiten, jahr FROM Buch
    WHERE jahr >= 2000;
  ```

  Ergebnis

titel	seiten	jahr
Taschenbuch der Physik	1092	2000
Relationale Datenbanken und SQL	352	2000
MySQL & mSQL	512	2000
Thermodynamics and Statistical Mechanics	463	2000

- Beispiel 3

  Abfrage

  ```
  SELECT titel, seiten, jahr FROM Buch
    WHERE titel LIKE "%Physi%";
  ```

  Ergebnis

titel	seiten	jahr
Physik für Ingenieure	744	1999
Taschenbuch der Physik	1092	2000
Fundamentals of Physics Extended	1198	1997
Physik - Der Grundkurs	426	1999

- Beispiel 4

  Abfrage

```
SELECT titel, preis/2.04 FROM Buch
  WHERE titel LIKE "%Physics%";
```

  Ergebnis

titel	preis/2.04
Fundamentals of Physics Extended	143.8088

- Beispiel 5

  Abfrage

```
SELECT titel, name FROM Buch NATURAL JOIN Verlag;
```

  Ergebnis

titel	name
Physik für Ingenieure	Springer
Taschenbuch der Physik	Harri Deutsch
Fundamentals of Physics Extended	Wiley
Physik - Der Grundkurs	Harri Deutsch
Einführung in die Thermodynamik	Hanser
Thermodynamik für Ingenieure	Vieweg
Relationale Datenbanken und SQL	Addison-Wesley
MySQL & mSQL	O'Reilly
Thermodynamics and Statistical Mechanics	Springer
Grundlagen der Gastechnik	Hanser

- Beispiel 6

  Abfrage

```
SELECT titel, name AS verlag
       FROM Buch NATURAL JOIN Verlag
       ORDER BY verlag;
```

  Ergebnis

titel	verlag
Relationale Datenbanken und SQL	Addison-Wesley
Grundlagen der Gastechnik	Hanser
Einführung in die Thermodynamik	Hanser
Taschenbuch der Physik	Harri Deutsch
Physik - Der Grundkurs	Harri Deutsch
MySQL & mSQL	O'Reilly
Physik für Ingenieure	Springer
Thermodynamics and Statistical Mechanics	Springer
Thermodynamik für Ingenieure	Vieweg
Fundamentals of Physics Extended	Wiley

- Beispiel 7

  Abfrage

```
SELECT COUNT(*), SUM(preis), jahr FROM Buch
     GROUP BY jahr;
```

  Ergebnis

COUNT(*)	SUM(preis)	jahr
1	79.80	NULL
1	293.37	1997
4	230.80	1999
4	321.56	2000

- Beispiel 8

  Abfrage

```
SELECT name as verlag, COUNT(*)
      FROM Buch NATURAL JOIN Verlag
      GROUP BY name;
```

  Ergebnis

verlag	COUNT(*)
Addison-Wesley	1
Hanser	2
Harri Deutsch	2
O'Reilly	1
Springer	2
Vieweg	1
Wiley	1

- Beispiel 9

  Abfrage

```
SELECT name AS verlag , SUM(preis) AS gesamtpreis
      FROM Buch NATURAL JOIN Verlag
```

```
GROUP BY verlag
ORDER BY gesamtpreis DESC;
```

Ergebnis

verlag	gesamtpreis
Wiley	293.37
Springer	193.66
Hanser	139.60
Harri Deutsch	106.00
O'Reilly	79.00
Addison-Wesley	59.90
Vieweg	54.00

Lösung von Aufgabe 7

a. SQL-Kommando:

```
SELECT wort, COUNT(*) AS anzahl
       FROM Schlagwort NATURAL JOIN BuchSchlagwort
       GROUP BY wort
       ORDER BY anzahl DESC;
```

Ergebnis:

```
+---------------------+--------+
| wort                | anzahl |
+---------------------+--------+
| Thermodynamik       |      7 |
| Lehrbuch            |      6 |
| Physik              |      5 |
| Quantenmechanik     |      3 |
| Datenbank           |      2 |
| Handbuch            |      2 |
| SQL                 |      2 |
| Ingenieurwesen      |      1 |
| Statistische Physik |      1 |
+---------------------+--------+
```

b. SQL-Kommando:

```
SELECT name AS autor, titel
      FROM Buch NATURAL JOIN BuchAutor NATURAL JOIN Autor
      ORDER BY autor, titel;
```

Ergebnis:

```
+--------------+-----------------------------------------+
| autor        | titel                                   |
+--------------+-----------------------------------------+
| Bohrmann     | Physik - Der Grundkurs                  |
| Cerbe        | Einführung in die Thermodynamik         |
| Cerbe        | Grundlagen der Gastechnik               |
| Greiner      | Thermodynamics and Statistical Mechanics |
| Halliday     | Fundamentals of Physics Extended        |
| Hering       | Physik für Ingenieure                   |
| Hoffmann     | Einführung in die Thermodynamik         |
| Jany         | Thermodynamik für Ingenieure            |
| King         | MySQL & mSQL                            |
| Langeheinecke | Thermodynamik für Ingenieure           |
| Martin       | Physik für Ingenieure                   |
| Matthiesen   | Relationale Datenbanken und SQL         |
| Neise        | Thermodynamics and Statistical Mechanics |
| Pitka        | Physik - Der Grundkurs                  |
| Reese        | MySQL & mSQL                            |
| Resnick      | Fundamentals of Physics Extended        |
| Sapper       | Thermodynamik für Ingenieure            |
| Stöcker      | Physik - Der Grundkurs                  |
```

```
| Stöcker       | Taschenbuch der Physik                    |
| Stöcker       | Thermodynamics and Statistical Mechanics  |
| Stohrer       | Physik für Ingenieure                     |
| Terlecki      | Physik - Der Grundkurs                    |
| Unterstein    | Relationale Datenbanken und SQL           |
| Walker        | Fundamentals of Physics Extended          |
| Yarger        | MySQL & mSQL                              |
+---------------+-------------------------------------------+
```

c. SQL-Kommando:

```
SELECT Autor.name AS autor, Verlag.name AS verlag
  FROM Autor NATURAL JOIN BuchAutor NATURAL JOIN Buch
     INNER JOIN Verlag ON Buch.verlag_id = Verlag.verlag_id
  GROUP BY autor, verlag
  ORDER BY autor;
```

Ergebnis:

```
+---------------+----------------+
| autor         | verlag         |
+---------------+----------------+
| Bohrmann      | Harri Deutsch  |
| Cerbe         | Hanser         |
| Greiner       | Springer       |
| Halliday      | Wiley          |
| Hering        | Springer       |
| Hoffmann      | Hanser         |
| Jany          | Vieweg         |
| King          | O'Reilly       |
| Langeheinecke | Vieweg         |
| Martin        | Springer       |
| Matthiesen    | Addison-Wesley |
| Neise         | Springer       |
| Pitka         | Harri Deutsch  |
| Reese         | O'Reilly       |
| Resnick       | Wiley          |
| Sapper        | Vieweg         |
| Stohrer       | Springer       |
| Stöcker       | Harri Deutsch  |
| Stöcker       | Springer       |
| Terlecki      | Harri Deutsch  |
| Unterstein    | Addison-Wesley |
| Walker        | Wiley          |
| Yarger        | O'Reilly       |
+---------------+----------------+
```

d. SQL-Kommando (mit Buchtitel):

```
SELECT Autor.name AS autor, titel
  FROM Autor NATURAL JOIN BuchAutor NATURAL JOIN Buch
          NATURAL JOIN BuchSchlagwort NATURAL JOIN Schlagwort
  WHERE wort = "Thermodynamik"
  ORDER BY autor;
```

Ergebnis:

```
+---------------+-----------------------------------------+
| autor         | titel                                   |
+---------------+-----------------------------------------+
| Bohrmann      | Physik - Der Grundkurs                  |
| Cerbe         | Einführung in die Thermodynamik         |
| Greiner       | Thermodynamics and Statistical Mechanics |
| Halliday      | Fundamentals of Physics Extended        |
| Hering        | Physik für Ingenieure                   |
| Jany          | Thermodynamik für Ingenieure            |
| Langeheinecke | Thermodynamik für Ingenieure            |
| Martin        | Physik für Ingenieure                   |
| Neise         | Thermodynamics and Statistical Mechanics |
| Pitka         | Physik - Der Grundkurs                  |
| Resnick       | Fundamentals of Physics Extended        |
| Sapper        | Thermodynamik für Ingenieure            |
| Stohrer       | Physik für Ingenieure                   |
| Stöcker       | Physik - Der Grundkurs                  |
| Stöcker       | Taschenbuch der Physik                  |
| Stöcker       | Thermodynamics and Statistical Mechanics |
| Terlecki      | Physik - Der Grundkurs                  |
| Walker        | Fundamentals of Physics Extended        |
+---------------+-----------------------------------------+
```

SQL-Kommando (ohne Buchtitel):

```
SELECT Autor.name AS autor
  FROM Autor NATURAL JOIN BuchAutor
        NATURAL JOIN BuchSchlagwort NATURAL JOIN Schlagwort
  WHERE wort = "Thermodynamik"
  GROUP BY autor
  ORDER BY autor;
```

Ergebnis:

```
+---------------+
| autor         |
+---------------+
| Bohrmann      |
| Cerbe         |
| Greiner       |
| Halliday      |
| Hering        |
| Jany          |
| Langeheinecke |
| Martin        |
| Neise         |
| Pitka         |
| Resnick       |
| Sapper        |
| Stohrer       |
| Stöcker       |
| Terlecki      |
```

```
| Walker         |
+----------------+
```

e. SQL-Kommando:

```
SELECT A2.vorname, A2.name
      FROM Autor AS A1 NATURAL JOIN BuchAutor AS BA1
         INNER JOIN BuchAutor AS BA2 ON BA1.isbn = BA2.isbn
         INNER JOIN Autor AS A2 ON BA2.autor_id = A2.autor_id
      WHERE A1.name = "Stöcker"
         AND BA1.autor_id != BA2.autor_id;
```

Ergebnis:

```
+----------+----------+
| vorname  | name     |
+----------+----------+
| Walter   | Greiner  |
| Ludwig   | Neise    |
| Rudolf   | Pitka    |
| Steffen  | Bohrmann |
| Günther  | Terlecki |
+----------+----------+
```