# Hochschule Darmstadt
# Fachbereich Informatik

## Entwicklung webbasierter Anwendungen

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

# Hochschule Darmstadt
# Fachbereich Informatik

## Wiederholung

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

# Verweise

Der Verweistext sollte eine klare Information über das Ziel des Verweises geben !

- Allgemeine Form

```
<a href="Dienst://Server:Port/Verz/Datei#Anker">
  Text</a>
```

  Teile davon können weggelassen werden

- Datei im selben / unter- / übergeordneten Verzeichnis

```
<a href="start.htm">Text</a>
<a href="sub/Datei.html">Text</a>
<a href="../inhalt.htm">Text</a>
```

relativ

- Datei auf anderem Server

```
<a href="http://www.xyz.de/datei.htm">Text</a>
```

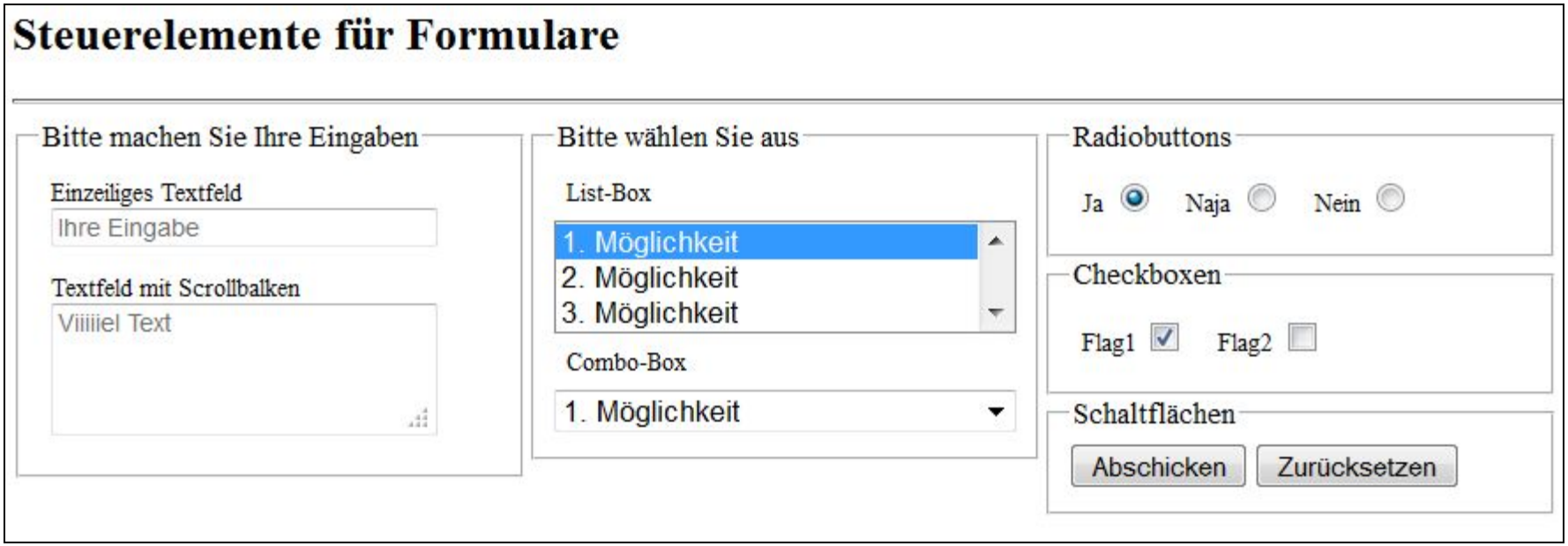
auch: localhost

absolut

- Groß-/Kleinschreibung beachten
  - Server laufen meist unter Unix und Unix ist case sensitive bezüglich Datei- und Verzeichnisnamen

beliebter Fehler unter Windows

# Formular - Beispiel



# Funktion von Formularen


- Formulare dienen der Eingabe von Daten
  - eingegebene Daten werden an Server übermittelt und dort ausgewertet
  - es gibt 2 Möglichkeiten der Datenübertragung
    - `get` übermittelt Parameter für Abfrage (z.B. Suchmaschine)
    - `post` übermittelt Daten zwecks Speicherung (z.B. Bestellung)

vgl. Reload im Browser

- Bereich mit Eingabeelementen im HTML-Body markieren

```
<form action="/cgi-bin/Echo.pl" id="form1"
    accept-charset="UTF-8" method="get">
 Steuerelemente (Eingabefelder, Auswahllisten,
 Buttons...) und sonstige HTML-Tags und CSS-Formatierung
</form>
```

hier: Übergabe der Daten an Perl-Skript

  - accept-charset zur Sicherheit gegen willkürliche Benutzereinstellung
  - falls das Steuerelement außerhalb des Formulars liegt, kann der Bezug über `form="form1"` hergestellt werden

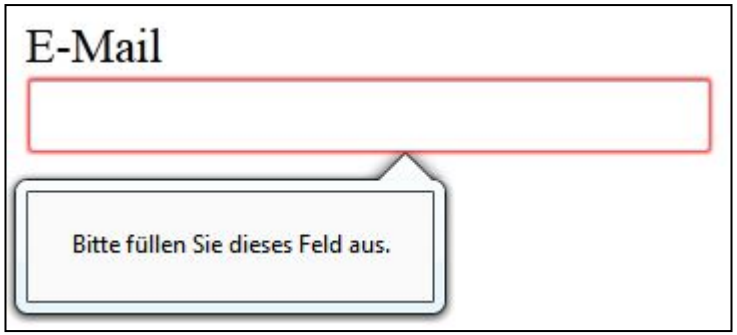
aber nicht mit Internet Explorer

- Alternative Aktion: Formulardaten per eMail verschicken
  - `action="mailto:Meier@xyz.de"`
  - unsicher, weil von der Installation beim Surfer abhängig

# Attribute zur Validierung von Eingabefelder

- required

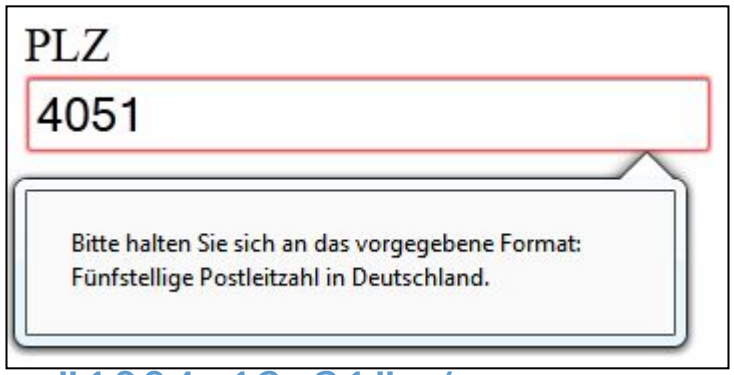
```
<input type="email"
  required />
```




- pattern

```
<input pattern="[0-9]{5}" name="plz"
  title="Fünfstellige Postleitzahl in Deutschland."/>
```

Ohne required, darf das Feld trotz pattern auch leer bleiben!




- min..max

```
<input name="bday" type="date" max="1994-12-31" />
```

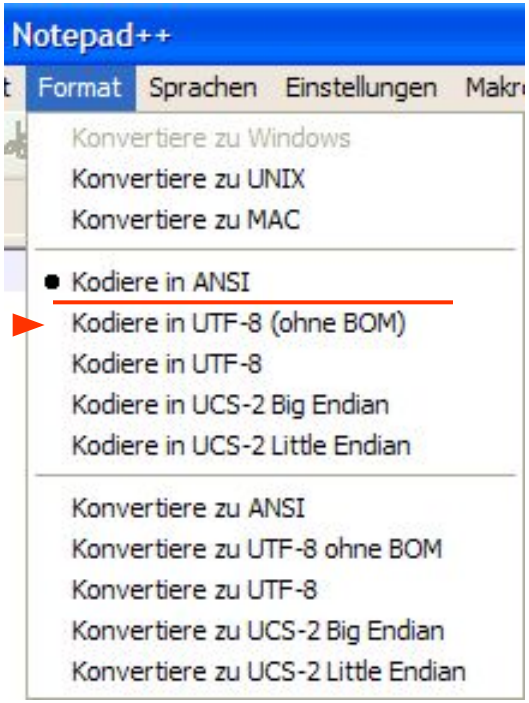
# Hochschule Darmstadt
# Fachbereich Informatik

## 2.1.4 HTML Werkzeuge

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

# Zeichenkodierung im Editor

- die Datei muss auch wirklich mit der angegebenen Zeichencodierung erstellt sein
  - Einstellung des Editors
  - Default des Betriebssystems



# Zeichenkodierung systemweit einheitlich

- vorzugsweise UTF-8 systemweit als Zeichenkodierung einsetzen
  - Projekt von vorneherein mit UTF-8 aufsetzen
  - nachträgliche Umstellung ist mühsam
  - uneinheitliche Kodierung würde explizite Konvertierungen erfordern
- PHP-Dateien in UTF-8 ohne BOM (Byte Order Mark) kodieren
  - BOM besteht aus Bytesequenz EF BB BF am Dateianfang
- Zeichenkodierung und Sortierreihenfolge gleich beim Anlegen der Datenbank festlegen
  - ```
    CREATE DATABASE `db`
    DEFAULT CHARACTER SET utf8
    COLLATE utf8_unicode_ci;
    ```
  - vorzugsweise einheitlich für alle Tabellen und Felder
- Zeichensatz für die Kommunikation zwischen PHP und Datenbank definieren
  - ```
    $mysqli->set_charset("utf8");
    ```

# HTML Browser




- Es gibt eine Vielzahl verschiedener Browser in verschiedenen Versionen
  - Die Unterstützung von "neueren" Features ist nicht sicher
    - Im Web gibt es diverse Seiten, welche die Umsetzung verfolgen z.B. http://caniuse.com oder http://html5readiness.com/
    - Für ältere Browser muss oft eine Rückfalllösung entwickelt werden
  - Webseiten unbedingt für verschieden Browser testen
    - z.B. bei http://browsershots.org

# Audio element - Working Draft

*Method of playing sound on webpages (without requiring a plug-in)*

| *Usage stats: | Global |
|---|---|
| Support: | 83.73% |
| Partial support: | 0.02% |
| Total: | 83.75% |

| Show all versions | IE | Firefox | Chrome | Safari | Opera | iOS Safari | Opera Mini | Android Browser | Blackberry Browser | IE Mobile |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 2.1 | | |
| | | | | | | | | 2.2 | | |
| | | | | | | 3.2 | | 2.3 | | |
| | | | | | | 4.0-4.1 | | 3.0 | | |
| | 8.0 | | | | | 4.2-4.3 | | 4.0 | | |
| | 9.0 | | 31.0 | | | 5.0-5.1 | | 4.1 | | |
| | 10.0 | 26.0 | 32.0 | | | 6.0-6.1 | | 4.2-4.3 | 7.0 | |
| Current | 11.0 | 27.0 | 33.0 | 7.0 | 19.0 | 7.0 | 5.0-7.0 | 4.4 | 10.0 | 10.0 |
| Near future | | 28.0 | 34.0 | | 20.0 | | | | | |
| Farther future | | 29.0 | 35.0 | | 21.0 | | | | | |
| 3 versions ahead | | 30.0 | 36.0 | | | | | | | |

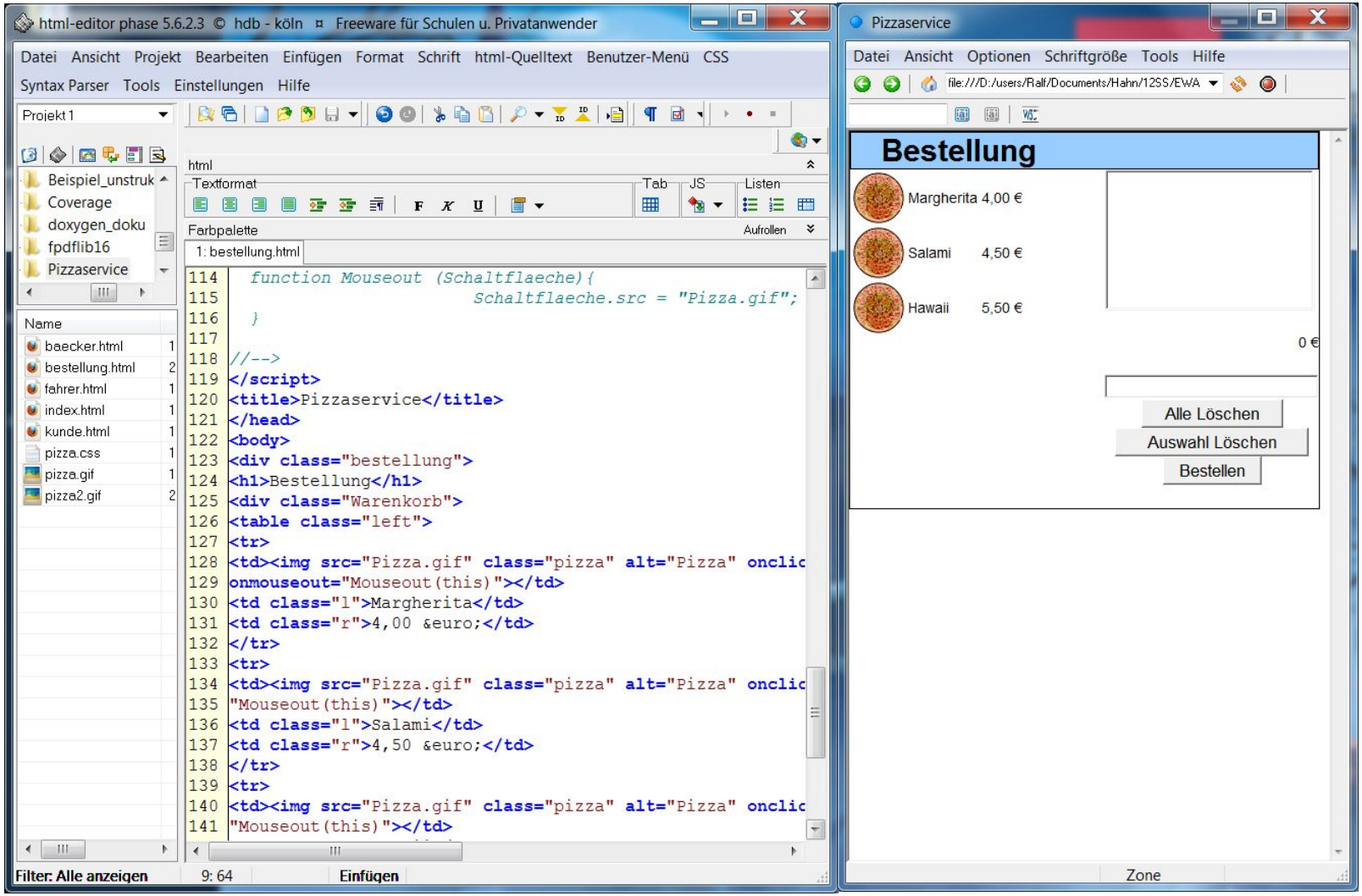
http://caniuse.com

# HTML Editor

z.B. Phase 5

- vergleichbar mit komfortabler Programmierumgebung
  - oft mit Preview des Ergebnisses
- Vorteil:
  - hand-optimierter HTML-Code, neueste Features nutzbar
- Nachteil:
  - Programmierer muss die Schnittmenge der Browser finden

# HTML Editor - Beispiel

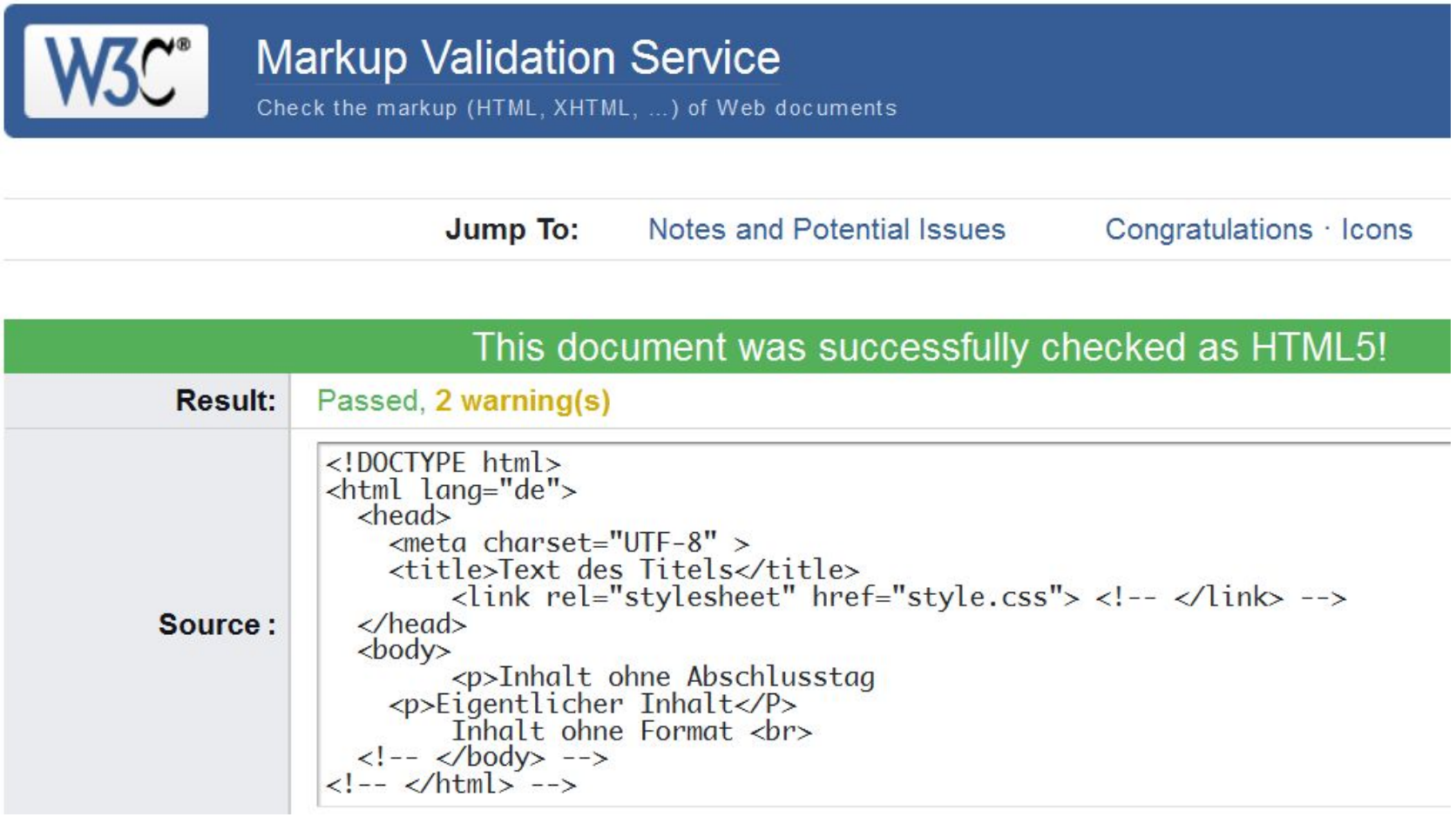


# HTML Prüfung

- Browser ignorieren normalerweise unbekannte oder falsche Tags und Attribute
  - es gibt keine Fehlermeldung, allenfalls Fehlverhalten
  - das Verhalten im Fehlerfall hängt stark vom Browser ab
- seit HTML5 sind ganz offiziell viele Konstrukte erlaubt, die in HTML 4 noch Fehler gewesen wären
  - diverse (schließende) Tags sind optional
  - unbekannte Attribute werden ignoriert
- deshalb: HTML Code vor der Veröffentlichung prüfen
  - anhand der Spezifikation: Syntax, Tag- und Attributnamen, Schachtelungsregeln, etc. mit einem Validator (z.B. validator.w3.org)
  - für HTML5 eventuell zusätzlich auf die Einhaltung von "Programmierrichtlinien" prüfen
  - auch generiertes HTML (z.B. aus PHP) prüfen!

# HTML Validator – W3C

http://validator.w3.org




- HTML5 wird validiert, obwohl es einige bedenkliche Konstrukte enthält

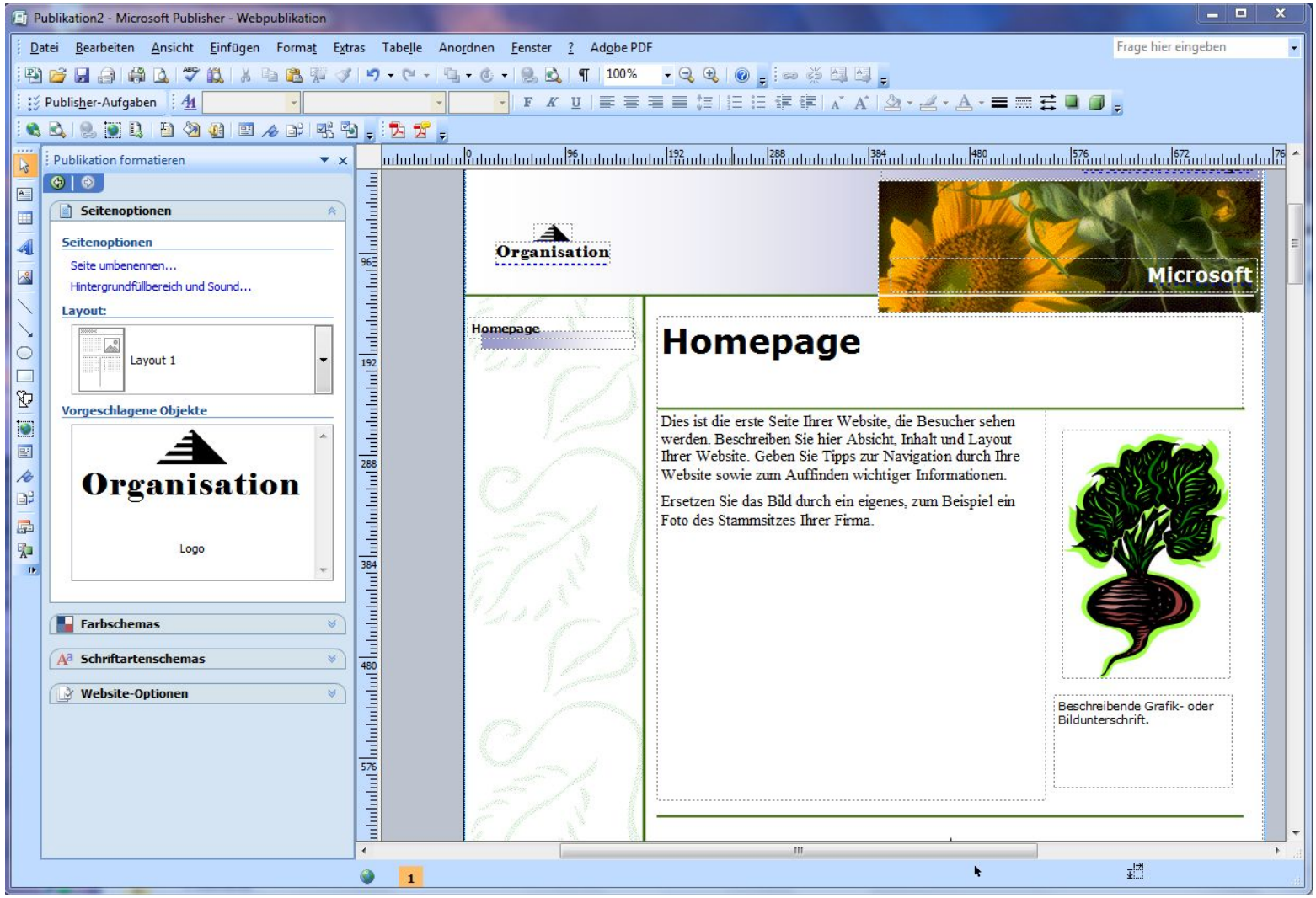
# WYSIWYG Tools

GoLive, Dreamweaver, FrontPage, Publisher

- Hoher Komfort (ähnlich Word)
  - HTML wird nicht mehr "programmiert"; Anweisungen und Attribute werden problemorientiert über Dialoge definiert
  - HTML ist nur "internes Datenformat"
  - HTML kann vom Autor betrachtet werden; muss aber nicht
- nur die Formatiermöglichkeiten von HTML sind erlaubt
  - Tabellen und Grafikeinbindung gemäß HTML
- Darstellung etwa so wie im Browser
  - eine mögliche WYSIWYG-Variante unter vielen
  - zusätzlich Vorschau mit verschiedenen Browsern
- generiertes HTML ist meist "multi-browser-tauglich"

man muss die Prinzipien verstehen

# WYSIWYG Tool – Beispiel: Microsoft Publisher 2007



# Export aus anderen Tools

- früher unbrauchbar, mittlerweile etwas besser
- generierter HTML-Code sehr komplex, viele Dateien
  - ⇨ praktisch schon fast wieder proprietäres Dateiformat

# Zusammenfassung HTML

- HTML-Grundlagen
  - Grundgerüst: DOCTYPE, <html>, <head>, <body>, <title>, charset...
  - Schreibregeln und Syntax
  - Tags und Attribute
  - Hyperlinks
- Formulare
  - Buttons, Listen, Datenübermittlung
- Werkzeuge

Jetzt wissen Sie alles um eine komplette und logisch strukturierte HTML-Seite zu entwickeln!

# Hochschule Darmstadt
# Fachbereich Informatik

## 2.1.5 HTML Layout

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

# Problematik des Layouts

- HTML ...
  - Anordnung der Tags erfolgt nach der Reihenfolge in der HTML-Datei
- Darstellung hängt vom System des Betrachters ab
  - Egal ob Computermonitor, FullHD-Fernseher, Handy oder Netbook
  - Informationsdarstellung mit sehr verschiedenen Auflösungen und Schriftgrößen (ggfs. mit automatischem Zoom)


# Problematik des Layouts (II)

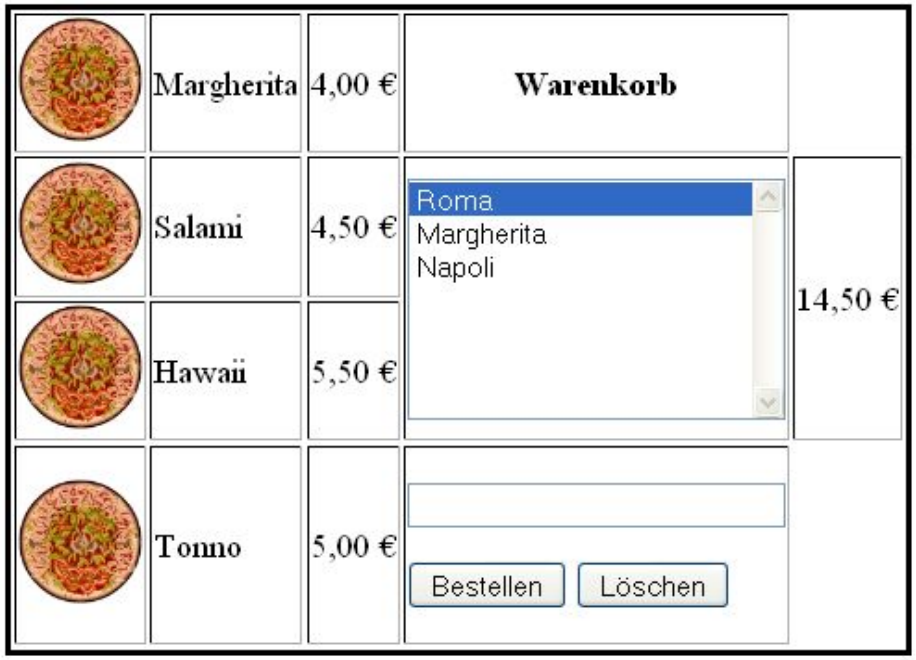
- Dynamische (sich anpassende) Layouts sind schwierig zu entwerfen
  - die meisten Seitengestalter denken in statischen Layouts
  - traditionell sind in HTML zwei "Layoutmanager" verfügbar:
    `<table>` für Layoutzwecke verpönt
    `<frameset>` seit HTML5 verboten
  - stattdessen wird heute CSS verwendet

- Eine Tabelle ist (immer noch) häufig Basis des Seitenlayouts
  - statisches Layoutraster durch Bemaßung in Pixel
  - dynamisches Layoutraster durch Bemaßung in Prozent

# Tabelle als Layoutmanager

verpönt im Hinblick auf Barrierefreiheit !

- Eine Tabelle ist (immer noch) häufig Basis des Seitenlayouts
  - normalerweise „blinde“ Tabelle, d.h. ohne Rand




  - ein Screenreader liest die Tabelle von links nach rechts und von oben nach unten

# Layoutvorbereitung für die CSS-Formatierung

- In HTML wird eine Seite als inhaltlich logische Sequenz von Blöcken aufgebaut
  - jeder Block wird mit einem Tag gekennzeichnet
    - mit einem passenden Standard-Tag z.B. `<h1>…</h1>`
    - oder sonst mit `<div>...</div>`
  - Elemente, die speziell formatiert werden sollen, aber keinen Block erzeugen sollen, werden durch `<span>...</span>` gekennzeichnet
  - die Reihenfolge innerhalb der HTML-Seite ist so gewählt, dass sie der logischen Reihenfolge entsprechen
  - diese Sequenz definiert auch die Vorlesereihenfolge des Screenreaders

- Die einzelnen Blöcke werden dann später mit CSS formatiert, positioniert und ausgerichtet
  - z.B. CSS-Attribute für den Textfluss: `float, margin, clear`
  - www.fbi.h-da.de ist so aufgebaut

# Layoutvorbereitung für die CSS-Formatierung - Beispiel

```
<header><h1>Kopfzeile</h1></header>
<nav><ul>
  <li>Menu1</li> <li>Menu2</li>
</ul></nav>
<section>
   <article>Inhalt1</article >
   <article>Inhalt2</article >
</section>
<footer>Fußzeile</footer>
```

Nicht gerade schön – aber logisch genau das, was wir mit HTML wollen!

# Zusammenfassung

- Problematik des Layouts
  - WYSI(not)WYG
  - Barrierefreies Layout
- Layout in HTML unerwünscht
  - Tabelle für Layout-Zwecke
  - Nachteile mit Screenreader
- Vorbereitung für die CSS-Formatierung

Jetzt beherrschen Sie die Grundlagen von HTML und können die Elemente auf einer HTML-Seite anordnen!