



TECHNISCHE  
UNIVERSITÄT  
DRESDEN

# Grundlagen (X)HTML + CSS

## XHTML

Hanjo Müller ([hanjo.mueller@bunix.de](mailto:hanjo.mueller@bunix.de))  
<http://htmlkurs.bunix.de>

- Überblick
- **Grundlagen (X)HTML 1.0**
- Grundlagen Cascading Stylesheets
- Grundlagen XML
- Grundlagen Javascript
- Webtechnologien



- Dokumentaufbau
  - Syntax
  - Strikte Baumstruktur
  - Zeichenkodierung
  - Validierbarkeit
- Grundlegende Elemente
  - Blockelemente (Absätze und Tabellen)
  - Inlineelemente (Textformat und Links)
  - Leerelemente (Bilder und Metaangaben)
  - Tabellen
  - Formulare

- Elemente werden durch spitze Klammern (< >) gekennzeichnet
- Jedes Element besteht aus einem öffnenden und einem schließenden Tag
- Das schließende Tag beginnt mit einem Schrägstrich (/)
- Elementnamen werden klein geschrieben
- Jedes Element kann durch Attribute näher beschrieben werden

```
<h1>Eine Hauptüberschrift</h1>
```

```
<h1 class="rot">Eine CSS-Formatierte Überschrift</h1>
```

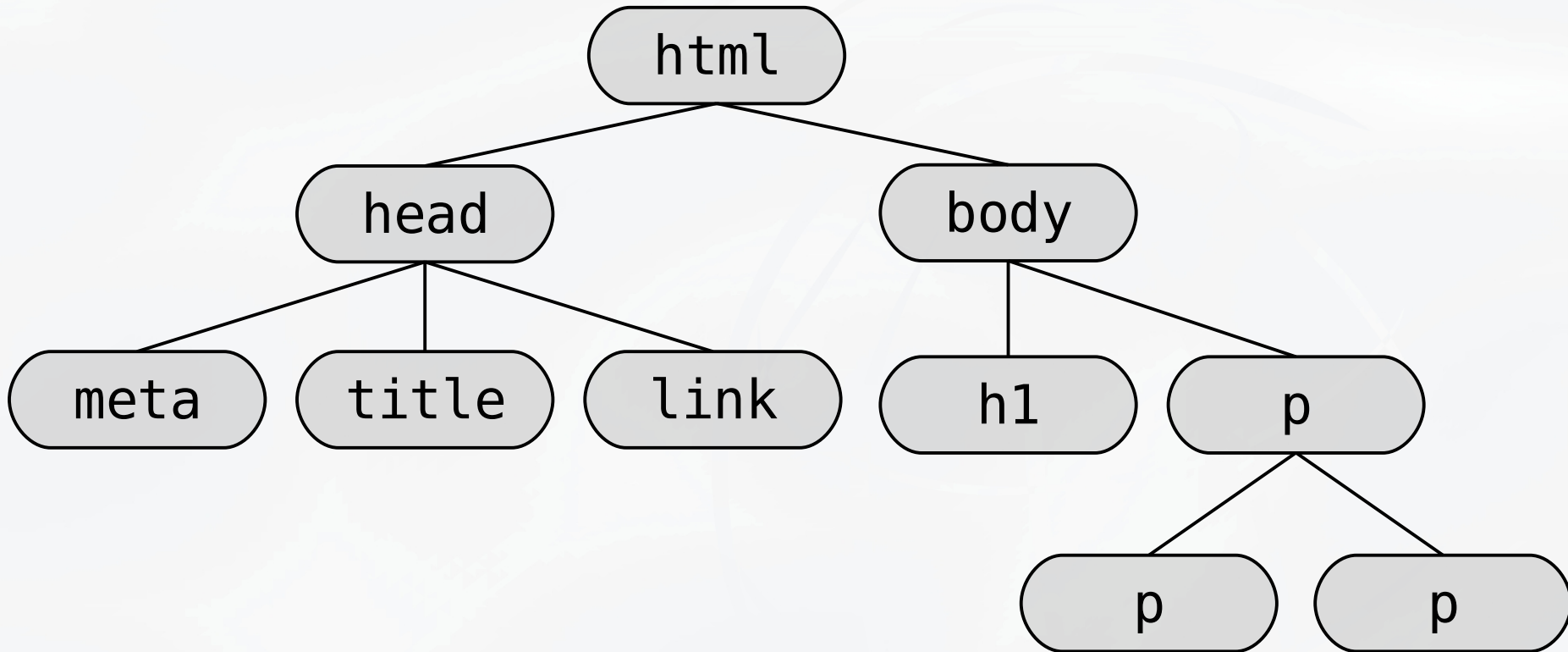
# XHTML – Dokumentstruktur (Baum)

---

- Jedes Dokument beginnt mit einer XML- und einem DOCTYPE-Deklaration
- Jedes Dokument besteht aus Elementen
  - Blockelemente bilden die Struktur ab
  - Inlineelemente dienen der Formatierung
- Die Elemente bilden einen Baum
  - Der Baum beginnt mit dem html-Element
  - Der Baum enthält genau einen head- und einen body-Zweig
  - Zweige können geschachtelt werden, nicht aber überlappen

# XHTML – Dokumentstruktur (Baum)

---



- Die XML-Deklaration legt die XML Version und den Zeichensatz fest

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
```

- Die DOCTYPE-Deklaration legt das Dokumentformat fest

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
```

```
Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

# XHTML – Document Types

---

- HTML 4.01  
<http://www.w3.org/TR/html4/strict.dtd>
- HTML 4.01 Transitional  
<http://www.w3.org/TR/html4/loose.dtd>
- HTML 4.01 Frameset  
<http://www.w3.org/TR/html4/frameset.dtd>
- XHTML 1.0 Strict  
<http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd>
- XHTML 1.0 Transitional  
<http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd>

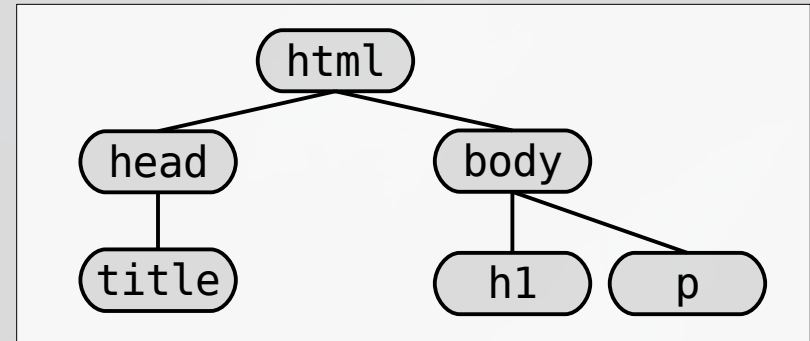
- Weltweit verschiedene Schriftzeichen
- Standardisierung von Zeichensätzen
  - ASCII – 128 Zeichen große Tabelle
  - ISO 8858 – 15 verschiedene Versionen
  - UTF-8 – 8-Bit große Kodierungstabelle
- In HTML manuelle Zeichenkodierung
  - Für alle Landestypischen Sonderzeichen
  - Für in HTML verwendete Steuerzeichen
- Heute typisch Verwendung automatischer Zeichenkodierung

# XHTML – Zeichenkodierung

		Zeichen-Referenz	Entity-Referenz
Ä		&#196;	&Auml;
ä		&#228;	&auml;
Ö		&#214;	&Ouml;
ö		&#246;	&ouml;
Ü		&#220;	&Uuml;
ü		&#252;	&uuml;
ß		&#223;	&szlig;
<	öffnende spitze Klammer	&#60;	&lt;
>	schließende spitze Klammer	&#62;	&gt;
&	Kaufmanns-Und	&#38;	&amp;
"	Doppeltes Anführungszeichen	&#34;	&quot;
	geschütztes Leerzeichen	&#160;	&nbsp;
€	Euro-Zeichen	&#8364;	&euro;

# XHTML – Beispieldokument

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
  <head>
    <title>Titel der Webseite</title>
  </head>
  <body>
    <h1>Dies ist eine Überschrift ersten Grades</h1>
    <p>Ich bin nur ein kleiner Blindtext.</p>
  </body>
</html>
```



- Syntax und Zeichensatz von XHTML Dokumenten sind automatisiert prüfbar
- W3C bietet Webbasierenden Validator an
  - Prüft auf konsistente Baumstruktur
  - Prüft auf fehlerhafte Zeichenkodierung
  - <http://validator.w3.org/>
- Hilfreiche Extensions für den Firefox
  - <http://chrispederick.com/work/webdeveloper/>
  - <http://users.skynet.be/mgueury/mozilla/>

- Block-Elemente definieren Text-Blöcke und sind durch Zeilenumbrüche getrennt
- für Überschriften werden die Block-Elemente **<h1>** bis **<h6>** verwendet
- für Absätze wird das Blockelement **<p>** verwendet
- Aufzählungen können als Gliederung (**<ol>**) und ungeordnete Liste (**<ul>**) realisiert werden. Jeder Eintrag wird in ein eigenes Listenelement (**<li>**) geschrieben

# XHTML – Blockelemente (Beispiel)

---

```
<body>  
<h1>Testseite</h1>  
<h2>Dies ist die erste Unterüberschrift</h2>  
<p>Dies ist der erste Absatz</p>  
<h2>Dies ist die zweite Unterüberschrift</h2>  
<p>  
    Dies ist der zweite Absatz  
<ul>  
    <li>Erstes Element</li>  
    <li>Zweites Element</li>  
</ul>  
</p>  
</body>
```

- Erstellen Sie ein XHTML-Dokument mit allen notwendigen Elementen.
- Erzeugen Sie einen Beispieldinhalt mit den bekannten Elementen wie Überschriften, Absätzen und Listen.
- Nutzen Sie die folgende Vorlage als Beispiel.
- Nutzen Sie zur Validierung des Quellcodes den Onlinevalidator des W3C.

# XHTML – Blockelemente (Übung/Screenshot)



- Tabellen ermöglichen es, Inhalte übersichtlich darzustellen
- Tabellen sollten nicht verwendet werden um eine Seite zu strukturieren
- Jede Tabelle beginnt mit einem **table**
- Jedes table Element enthält wenigstens ein **tr** Element
- Jedes **tr** (Zeile) Element enthält wenigstens ein **td** (Spalte) Element
- Jede Zeile sollte gleich viele Spalten haben

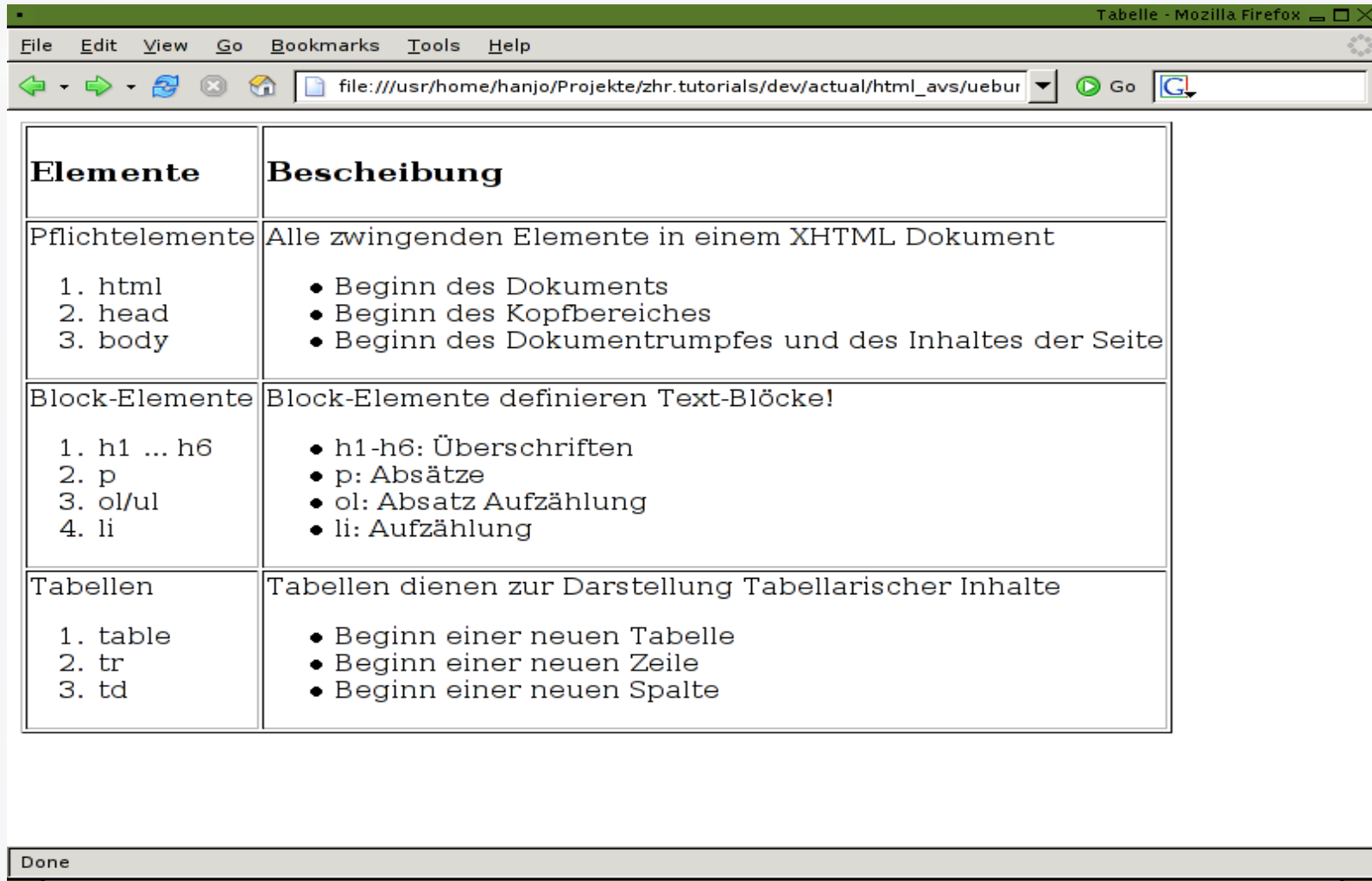
# XHTML – Tabellen (Beispiel)

```
<body>
  <table>
    <tr>
      <td>Zeile1 Spalte1</td>
      <td>Zeile1 Spalte2</td>
    </tr>
    <tr>
      <td>Zeile2 Spalte1</td>
      <td>Zeile2 Spalte2</td>
    </tr>
  </table>
</body>
```

Zeile1 Spalte1	Zeile1 Spalte2
Zeile2 Spalte1	Zeile2 Spalte2

- Erstellen Sie ein neues XHTML konformes Dokument mit beliebigem Titel. Das Dokument soll eine Tabelle mit allen bisher verwendeten Elementen und einer kurzen Erklärung zu jedem der Elemente enthalten.
- Achten Sie darauf alle notwendigen Bestandteile und Elemente zu verwenden!

# XHTML – Tabellen (Übung/Screenshot)



The screenshot shows a Mozilla Firefox browser window with the title 'Tabelle - Mozilla Firefox'. The address bar contains the file path: file:///usr/home/hanjo/Projekte/zhr.tutorials/dev/actual/html\_avs/uebur. The main content area displays a table with the following structure:

Elemente	Bescheinung
<b>Pflichtelemente</b> 1. html 2. head 3. body	Alle zwingenden Elemente in einem XHTML Dokument <ul style="list-style-type: none"><li>• Beginn des Dokuments</li><li>• Beginn des Kopfbereiches</li><li>• Beginn des Dokumentrumpfes und des Inhaltes der Seite</li></ul>
<b>Block-Elemente</b> 1. h1 ... h6 2. p 3. ol/ul 4. li	Block-Elemente definieren Text-Blöcke! <ul style="list-style-type: none"><li>• h1-h6: Überschriften</li><li>• p: Absätze</li><li>• ol: Absatz Aufzählung</li><li>• li: Aufzählung</li></ul>
<b>Tabellen</b> 1. table 2. tr 3. td	Tabellen dienen zur Darstellung Tabellarischer Inhalte <ul style="list-style-type: none"><li>• Beginn einer neuen Tabelle</li><li>• Beginn einer neuen Zeile</li><li>• Beginn einer neuen Spalte</li></ul>

Done

- Inline-Elemente erzeugen keinen Zeilenumbruch wie die Block-Elemente.
- Inline-Elemente dienen meist zur Textformatierung
- Fetter Text wird mit dem **b** (bold) Element erzeugt
- Kursiver Text wird mit dem *i* (italic) Element erzeugt
- Inline-Elemente können geschachtelt werden

```
<b>fett<i>fett-kursiv</i>fett</b><i>kursiv</i>
```

# XHTML – Inlinenelemente (Verweise)

---

- Verknüpfungen zwischen Dokumenten erfolgen mittels Links

```
<a href="URL">Zu verlinkender Text</a>
```

- Lokale Links über Pfadangaben

```
<a href="ordner/ziel.html">Zu verlinkender Text</a>
```

- Entfernte Links über URL mit Protokollzusatz

```
<a href="http://www.ziel.org/ziel.html">Zu verlinkender Text</a>
```

# XHTML – Inlinenelemente (Attribute)

---

- Jedes Element kann durch Attribute näher definiert werden.

```
<a href="http://de.selfhtml.org/">SelfHTML</a>
```

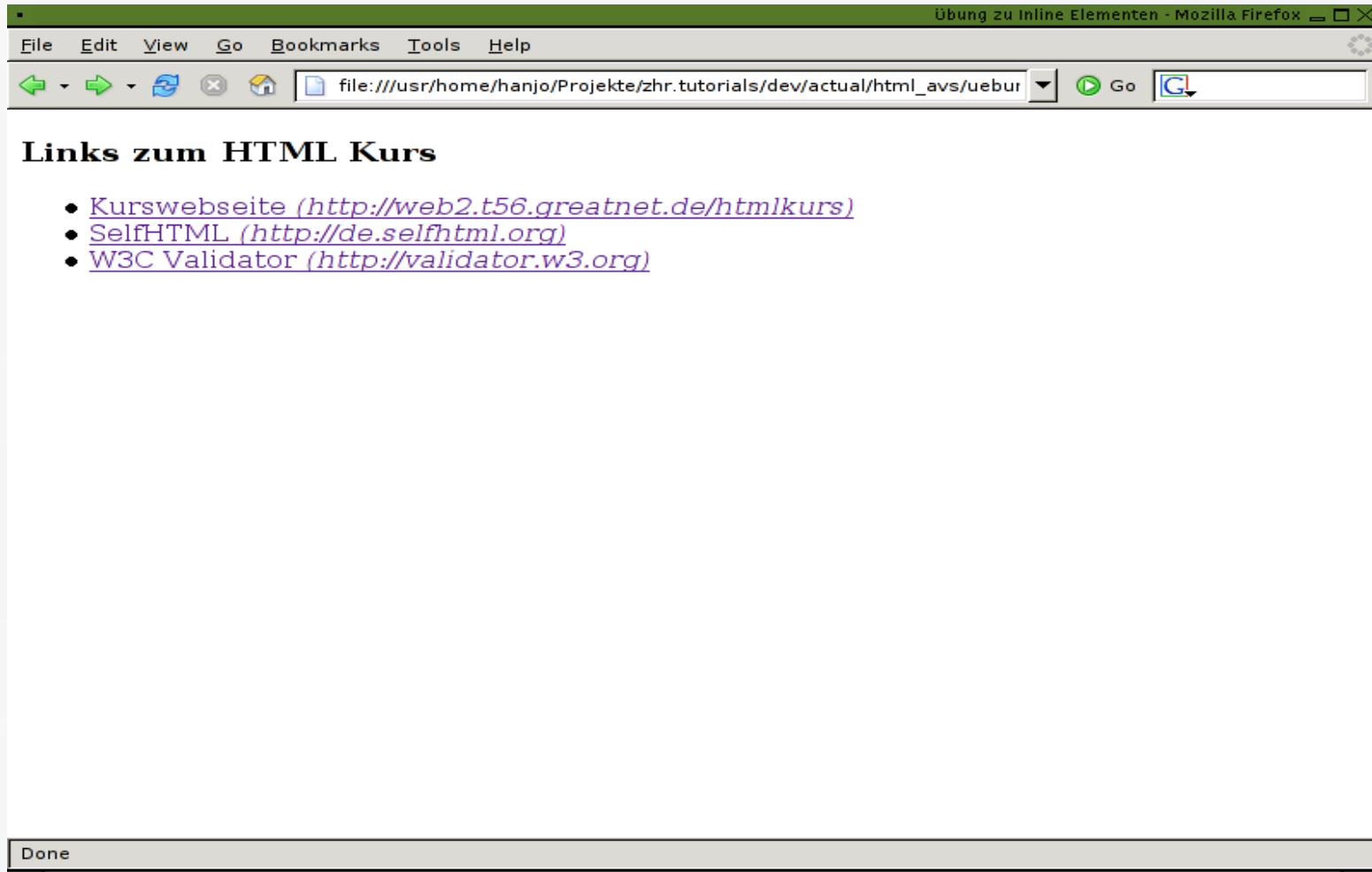
- Attributnamen werden wie die Elementnamen nur in Kleinbuchstaben geschrieben.
- Attributwerte nach dem Gleichheitszeichen müssen in Anführungszeichen geschrieben werden.
- Jedem Attribut muss ein Wert zugewiesen werden. "Leere" Attribute wie in HTML sind nicht erlaubt

# XHTML – Inlineelemente (Übung)

---

- Erstellen Sie eine kleine ungeordnete Liste mit Internetadressen die bei diesem Kurs hilfreich sind.
- Verwenden Sie bei der Gestaltung kennengelernte InlineElemente.
- Achten Sie dabei auf die korrekte Schachtelung der Inline-Elemente!
- Verwenden Sie die folgenden Links:
  - Kurs (<http://htmlkurs.bunix.de>)
  - SelfHTML (<http://de.selfhtml.org>)
  - W3C Validator (<http://validator.w3.org>)

# XHTML – Inlineelemente (Übung/Screenshot)



- Einige Elemente in XHTML sind nur durch ein einziges Tag definiert, sie besitzen kein End-Tag.
- Das Tag für ein leeres Element schließt mit einem Schrägstrich (/) vor der abschließenden spitzen Klammer.

```
<meta name="test" content="test" />
```

```

```

```
<br />
```

- Zur Wahrung der Kompatibilität gehört ein Leerzeichen vor den Schrägstrich.

# XHTML – Leerelemente (Grafiken)

---

- es lassen sich JPG, GIF und PNG Grafiken nutzen
- Verwendung des img Elements

```

```

- Regeln für die Verwendung von Bildern
  - Erschließt sich der Inhalt auch ohne Grafik?
  - Wie groß ist die Grafikdatei (MByte)?
  - Welches Format hat die Grafik (Bildpunkte)?
  - In welchem Dateiformat liegt die Grafik vor?

# XHTML – Leerelemente (Metaangaben)

---

- enthalten Informationen u.a. für Suchmaschinen
- nicht zwingend notwendig
- Wichtige Meta Angaben sind die Beschreibung, der Autor und die Stichworte

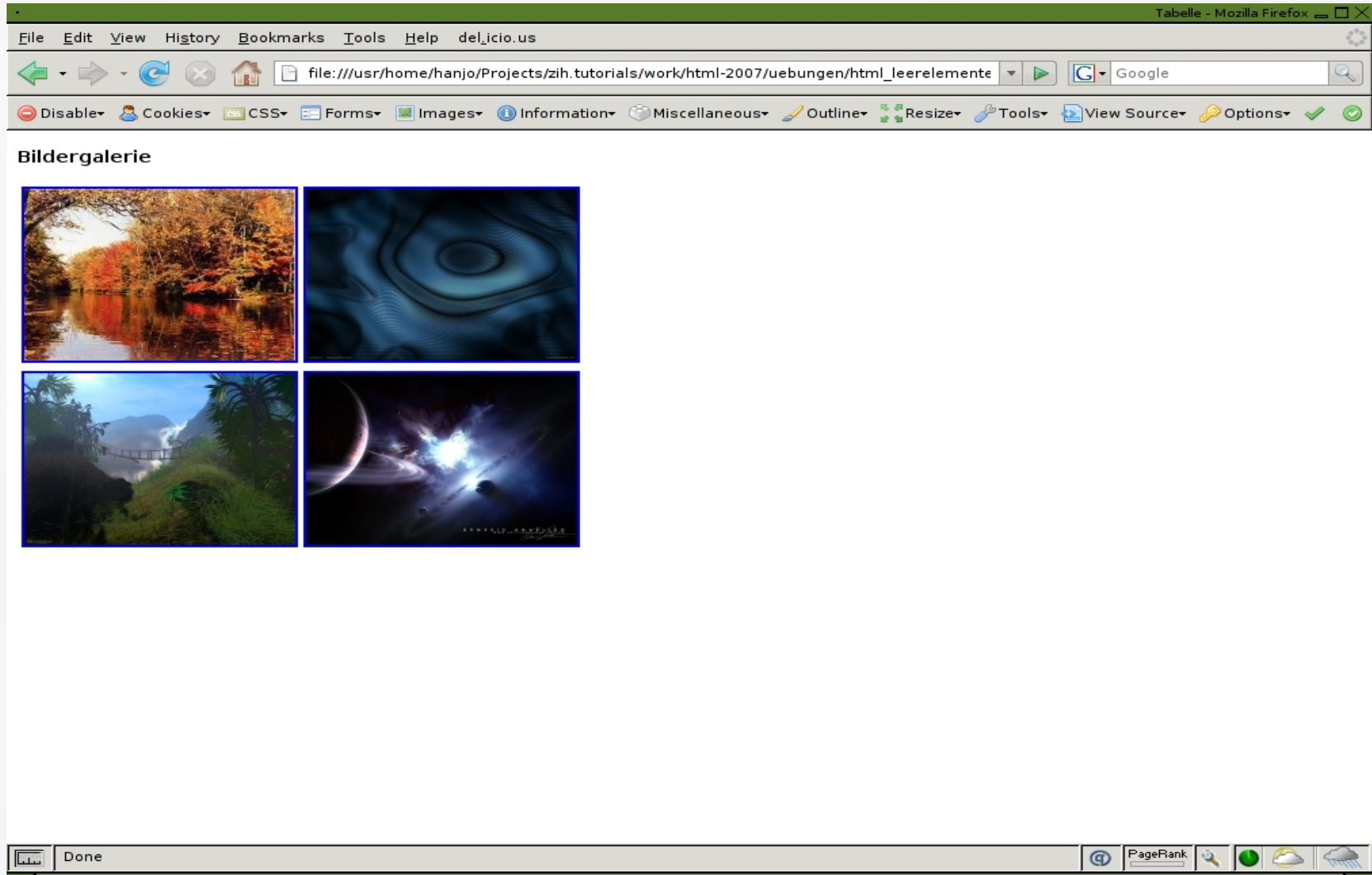
```
<meta name="description" content="Beschreibungstext" />
```

```
<meta name="author" content="Hans Otto (Firma)" />
```

```
<meta name="keywords" content="HTML, Kurs" />
```

- Erstellen Sie eine kleine Bildergalerie aus den vorliegenden Bildern.
- Erstellen Sie dazu zuerst eine Seite auf der die vier Vorschaubilder mit einer Tabelle angeordnet werden.
- Erstellen Sie dann für jedes große Bild eine weitere Datei die das Bild enthält.
- Verlinken Sie die Vorschaubilder auf die entsprechenden Seiten mit dem großen Bild.
- Ergänzen Sie die Detailseite mit einem Link, welcher zur Vorschau zurückführt.

# XHTML – Leerelemente (Übung)



- Beginnt mit **<form>** und endet mit **</form>**
- enthält verschiedene Formularelemente
- Wichtige Attribute
  - **action** -> wohin mit den Daten
  - **method** -> wie wird übermittelt
  - **enctype** -> mit welcher Kodierung

```
<form action="script.php" method="post">  
  <input type="text" name="text1" />  
  <input type="submit" value="Abschicken">  
</form>
```

- Steuert die Art in der Datenübertragung
- **METHOD=GET** (Standard)
  - Datenversand über URL-Parameter
  - wird benötigt wenn Daten mit JavaScript verarbeitet werden sollen
- **METHOD=POST**
  - Datenversand an die Standardeingabe eines Webservers
  - muss in jedem Fall verwendet werden wenn Dateien übertragen werden

# XHTML – Übertragungskodierung (enctype)

---

- Steuert die Art in der Datenkodierung
- ***application/x-www-form-urlencoded***
  - Standard wenn nichts angegeben ist
  - Geeignet zur Übertragung von Texten
  - Automatische Zeichenkodierung
- ***multipart/form-data***
  - Voraussetzung für die Dateiübertragung
  - Automatische Zeichenkodierung
- ***text/plain***
  - Keine Zeichenkodierung
  - Geeignet nur zur Übertragung von Sonderzeichen freien Texten

# XHTML – Formulare (`<input type="text" />`)

---

- einzeliges Texteingabefeld (text / password)
- wichtige Attribute
  - **name** – eindeutiger Name des Elementes
  - **value** – default-Wert im Element
  - **maxlength** – maximale Anzahl von Zeichen
  - **readonly** – Elementwert nicht änderbar
- Formatierung über CSS möglich

```
<input type="text" name="email" value="ihr.name@domain.de" />
```

# XHTML – Formulare (<textarea>...</textarea>)

---

- mehrzeiliges Texteingabefeld
- Wert steht zwischen öffnendem und schließendem Tag
- wichtige Attribute
  - **name** – eindeutiger Name des Elementes
  - **rows / cols** – Anzahl von Zeilen und Spalten
  - **readonly** – Elementwert nicht änderbar
- Formatierung über CSS möglich

```
<textarea name="eintext">  
  hier steht der default text  
</textarea>
```

# XHTML – Formulare (`<input type="hidden" />`)

---

- nicht sichtbares Textfeld
- verwendbar für versteckte Daten
- wichtige Attribute
  - **name** – eindeutiger Name des Elementes
  - **value** – übermittelter Wert im Element

```
<input type="hidden" name="versteckt" value="12345" />
```

# XHTML – Formulare (`<input type="submit" />`)

---

- Button zum Absenden eines Formulars
- Verwendung des Buttons übermittels Daten an den die Form-Action
- Wichtige Attribute
  - **name** – eindeutiger Name des Elements
  - **value** – Wert und Beschriftung des Elements
- Formatierung mit CSS möglich

```
<input type="submit" value="abschicken" />
```

# XHTML – Formulare (`<input type="img" src="" />`)

---

- Button zum Absenden eines Formulars
- Als Button wird eine Grafik angezeigt
- Wichtige Attribute
  - **name** – eindeutiger Name des Elements
  - **src** – Pfad zur anzuzeigenden Grafik
- **!!!** Werte aus einem Value Attribut werden nur vom Firefox korrekt übermittelt

```
<input type="image" name="abschicken" src="img/absend.jpg" />
```

# XHTML – Formulare (<select>...</select>)

---

- Auswahl bzw. Drop-Down Liste
- Wichtige Attribute
  - **name** – eindeutiger Name des Elements
  - **size** – Anzahl der Zeilen (1=DropDown)
  - **multiple** – Mehrfachauswahl erlauben
- Formatierung über CSS NICHT möglich

```
<!-- dropdown Liste -->
```

```
<select name="liste"> ... </select>
```

```
<!-- dreizeilige Liste -->
```

```
<select name="liste" size="3"> ... </select>
```

```
<!-- liste zur Mehrfachauswahl -->
```

```
<select name="liste" size="3" multiple="multiple"> ... </select>
```

# XHTML – Formulare (<option>...</option>)

---

- Elemente innerhalb einer Liste
- Wichtige Attribute
  - **select** – Vorauswahl des Elementes
  - **value** – übermittelter Wert
- Formatierung mit CSS NICHT möglich

```
<select name="liste" size="3" multiple="multiple">  
  <option value="1">Pizza Napoli</option>  
  <option value="2" selected="selected">Pizza Salami</option>  
</select>
```

# XHTML – Formulare (`<input type="radio" />`)

---

- boolesches Auswahl Element
- es kann immer nur ein Element gewählt sein
- Wichtige Attribute:
  - **name** – ordnet Elemente einer Gruppe zu
  - **value** – legt den übermittelten Wert fest
  - **checked** – Vorauswahl von Elementen
- Formatierung über CSS NICHT möglich

```
Visa <input type="radio" name="radio1" value="Visa" />
```

```
Master <input type="radio" name="radio1" value="Mastercard" />
```

```
Amex <input type="radio" name="radio1" value="American Express" />
```

# XHTML – Formulare (`<input type="checkbox" />`)

---

- Auswahl von Optionen über Checkboxes
- Wichtige Attribute:
  - **name** – ordnet Elemente einer Gruppe zu
  - **value** – legt den übermittelten Wert fest
  - **checked** – Vorauswahl von Elementen
- Formatierung über CSS NICHT möglich

```
Visa <input type="checkbox" name="radio1" value="Visa" />  
Master <input type="checkbox" name="radio1" value="Mastercard" />  
Amex <input type="checkbox" name="radio1" value="American Express" />
```

# XHTML – Formulare (`<input type="file" />`)

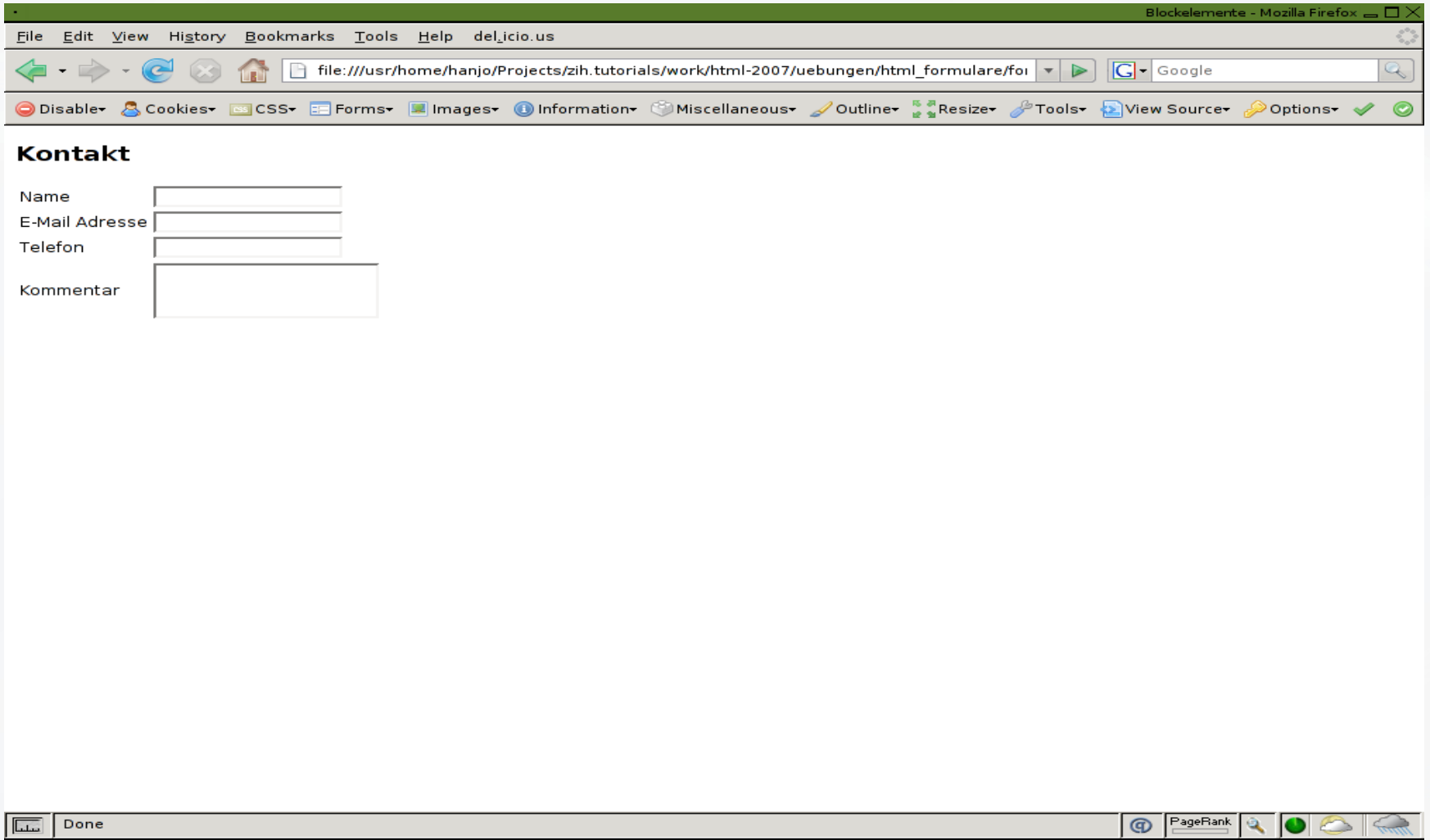
---

- Auswahl von lokalen Dateien für den Upload
- Aussehen vom Browser abhängig
- Dialogfelder nicht formatierbar
- wichtige Attribute
  - **name** – eindeutiger Name des Elementes
  - **maxlength** – maximale Dateigröße in Byte
- Formatierung über CSS NICHT möglich

```
<form method="post" enctype="multipart-form-data">  
  <input type="file" name="datei" maxlength="10000" />  
</form>
```

- Erstellen Sie ein einfaches Kontaktformular mit den folgenden Feldern:
  - Name
  - E-Mail Adresse
  - Telefonnummer
  - Kommentar
- Mit der action `mailto:user@host.tld` kann das Formular auch versendet werden. Dies geht aber leider nicht hier im Kabinett :(

# XHTML – Formulare (Übung)



The screenshot shows a Mozilla Firefox browser window with the title "Blockelemente - Mozilla Firefox". The address bar contains the file path: `file:///usr/home/hanjo/Projects/zih.tutorials/work/html-2007/uebungen/html_formulare/foi`. The search bar contains "Google". The browser's menu bar includes "File", "Edit", "View", "History", "Bookmarks", "Tools", and "Help". The status bar at the bottom shows "Done" and various icons including PageRank, search, and weather.

## Kontakt

Name

E-Mail Adresse

Telefon

Kommentar