Web-Technologien

Teil 4

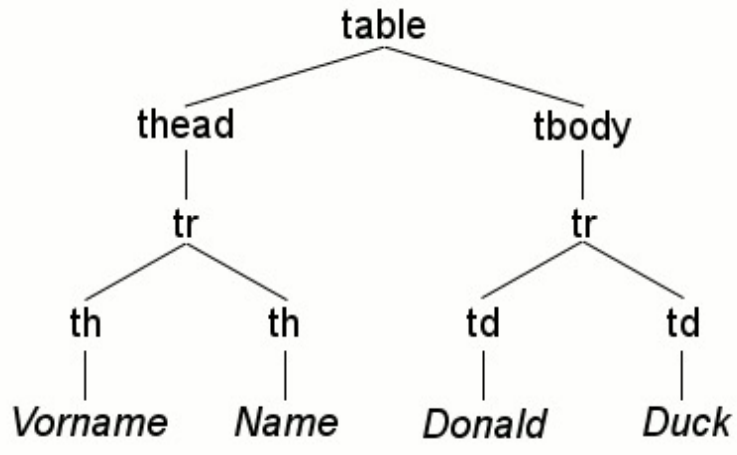
Ajax

- Asynchrones JavaScript and XML
- Asynchron: Webseite bleibt „nutzbar“, während im Hintergrund Daten geladen werden
- Request-Response-Modell wird abgelöst
- Seite muss nicht mehr komplett neugeladen werden
- Z.B.: Google Suchvorschläge
- XMLHTTP seit IE 5 (1999) verfügbar

Document Object Model

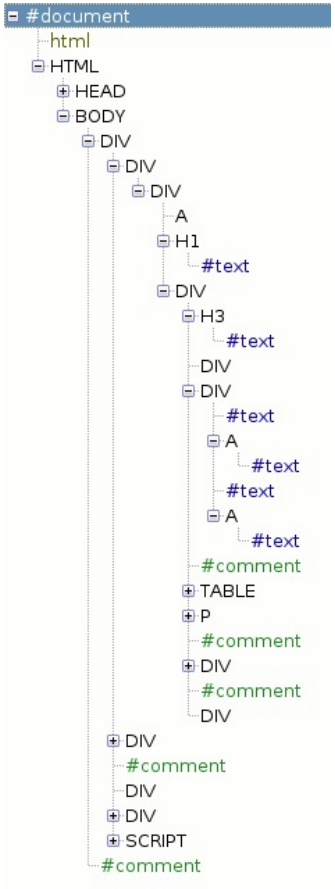
- API für Zugriff auf Web-Dokument
- Möglichkeiten: Knoten erstellen, ändern, löschen
- Anwendbarkeit: auf Client- aber auch Server-Seite

```
<table>
  <thead>
    <tr>
      <th>Vorname</th>
      <th>Name</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Donald</td>
      <td>Duck</td>
    </tr>
  </tbody>
</table>
```



Element Node Access

- getElementById()
- getElementsByName()
- getElementsByTagName()



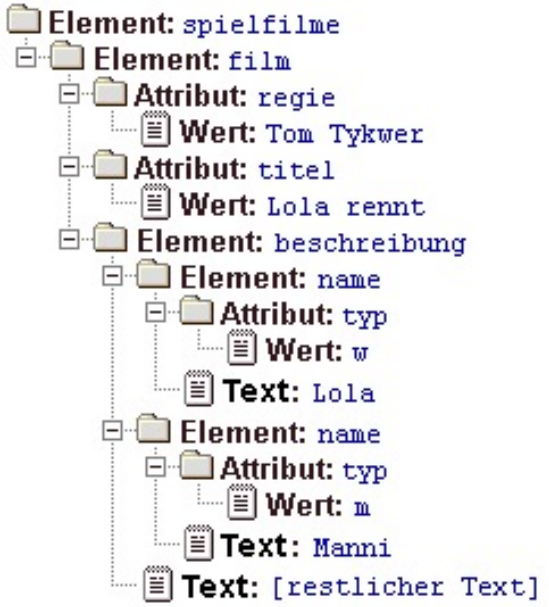
JavaScript Object Notation (JSON) versus XML

```
{
        "Herausgeber": "Xema",
        "Nummer": "1234-5678-9012-3456",
        "Deckung": 2e+6,
        "Waehrung": "EURO",
        "Inhaber":
        {
                "Name": "Mustermann",
                "Vorname": "Max",
                 Hobbys": [ "Reiten", "Golfen", "Lesen" ],
                …
        }
}
```

```
<Kreditkarte
        Herausgeber="Xema"
        Nummer="1234-5678-9012-3456"
        Deckung="2e+6"
        Waehrung="EURO">
        <Inhaber
        Name="Mustermann"
        Vorname="Max"
        maennlich="true"
        Alter="42"
        Partner="null">
                <Hobbys>
                        <Hobby>Reiten</Hobby>
                        <Hobby>Golfen</Hobby>
                        <Hobby>Lesen</Hobby>
                </Hobbys>
        </Inhaber>
</Kreditkarte>
```

Extensible Markup Language (XML)

```
<spielfilme>
        <film regie="Tom Tykwer" titel="Lola rennt">
                <beschreibung>
                        <name typ="w">Lola</name> rennt für <name typ="m">Manni</name>,
                        der 100000 Mark liegengelassen hat und noch 20 Minuten Zeit hat,
                        das Geld auszuliefern.
                </beschreibung>
        </film>
</spielfilme>
```



XML 2

- universelles Konzept für Datenspeicherung und –kommunikation
- nicht beschränkt auf Internet oder WWW
- verlustfreie Übertragbarkeit von Daten
- trennt Daten und Layout, d.h. dieselben Daten können zur Erstellung einer Website, eines Ausdrucks, eines akustischen Signals usw. verwendet werden

XML Derivate

- Extensible HTML (XHTML)
- Scalable Vector Graphics (SVG)
- Mathematical Markup Language (MathML)
- Wireless Markup Language (WML)
- …

XML Dateien

- Text-Dateien
- Tabulatoren und Leerzeichen können verwendet werden
- Kommentare:
  - <!-- The recommended storage temperature -->
    <storagetemperature>4°</storagetemperature>

SVG

```
<svg width="4in" height="3in">

<!-- The element g is used for grouping elements to complex units. --> <g>

<rect x="50" y="80" width="200" height="100" style="fill: #FFFFCC"/> </g>

</svg>
```

Case-Sensitivity

If DTD defines:
<!ELEMENT Chiptype (#PCDATA)>
then the XML file has to use:

<!-- Wrong -->
<chiptype>...</chiptype>
<CHIPTYPE>...</CHIPTYPE>

<!-- Correct -->
<Chiptype>...</Chiptype>

Anführungszeichen

```
<!-- Wrong -->
<area language=german dialect=saxonian>
```

```
<!-- Correct -->
<area language="german" dialect="saxonian">
<!-- Replace ' with &apos; -->
<bar drink="A German&apos;s beer">
```

Internal Document Type Definition

```
<?xml version="1.0"?>
<!DOCTYPE Greeting [ <!ELEMENT Greeting (#PCDATA)>
 ]>
<Greeting>Hello Jupiter!</Greeting>
```

Well-formed XML

- befolgt XML-Syntax-Regeln
- mindestens ein Datenelement (Wurzelelement)
- alle Elemente haben schließendes Tag
- keine sich überlappenden Tags
- beliebige Inhalte sind möglich

Well Formed	Not Well formed
<Personnel> <Employee> <Name>Seagull</Name> < ID> 3674 </ID> <Age>34</Age> </Employee> </Personnel>	<Personnel> <Employee> <Name>Seagull</Name> < ID> 3674 </ID> <Age>34 </Employee> </Personnel>

Valides XML

```
<?xml version="1.0"?>
<!DOCTYPE source [
        <!ELEMENT source (address, description)>
        <!ELEMENT address (#PCDATA)>
        <!ELEMENT description (#PCDATA)>
]>
<source>
        <address>http://www.willy-online.de/</address>
        <description>All interesting things concerning Men</description>
</source>
```

Web-Storage

- Zur Speicherung von Daten am Client
- Besser als Cookies ;-)
- Funktioniert in allen Browsern

Web Storage – So geht‘s

```
localStorage.setItem(“Zucker“, “2kg“);
localStorage[“Eier“] = 12;
alert(“Kaufe Eier“ + localStorage.getItem(“Eier“);
alert(“Kaufe Zucker“ + localStorage.getItem(“Zucker“);
```

Das localStorage-Objekt erlaubt den Zugriff auf Web Storage

Web Storage arbeitet mit Name-Wert-Paaren.

setItem nimmt einen Namen und einen Wert an und legt dieses Paar im Web Storage an

Zugriff auch mit eckigen Klammern möglich

getItem erwartet nur als Parameter nur den Namen des Eintrags

Web Storage auslesen

```
for (var i=0; i < localStorage.length; i++) {

        var name = localStorage.key(i);
        var wert = localStorage[name];
}
```

Web Storage auslesen

```
for (var i=0; i < localStorage.length; i++) {

        var name = localStorage.key(i);
        var wert = localStorage[name];
}
```

length wie viele Elemente liegen im Storage

Die Key-Methode gibt den Key des n-ten Key-Value-Paares zurück.

Mit dem key kann dann der dazugehörige Value geholt werden

Einkaufsliste

- Erstellen Sie eine einfache Einkaufsliste

Ajax 2

- Ablauf:
  - Request-Objekt erzeugen
  - Eventlistener registrieren
  - Request absenden

```
var ajaxRequest = new XMLHttpRequest();

ajaxRequest.addEventListener(“load“, ajaxGeladen);
ajaxRequest.addEventListener(“error“, ajaxFehler);
```

Ajax 3

```
ajaxRequest.open(“get“, “/ajaxURL“);
ajaxRequest.send();
```

Die open-Methode legt fest, was per Ajax aufgerufen werden soll

Die Request-Methode, analog zu den Formularen, ist entweder get oder post

URL, gegen die der Request ausgeführt werden soll

Send führt den Request aus