

(X)HTML

Netzwelt
Katlenburg-Lindau e.V.
www.Netzwelt-KaLi.de

Themen

0. Beispiel-Quelltext

1. Die Geschichte von (X)HTML

1.1. HTML-Versionen

2. HTML-Aufbau / Aussehen der Tags / Schreibweise / Syntax-Regeln

3. Einfacher Seitenaufbau

3.1. Bedeutung und Inhalte von <head>

3.2. Bedeutung und Inhalte von <body>

4. Grundlegende Tags (Überschriften, Absätze, Zeilenumbrüche, ...)

4.1. Textabsätze <p> und Zeilenumbrüche

4.2. Überschriften

4.3. Textauszeichnungen

4.3.1. Physische Textauszeichnungen

4.3.2. Logische Textauszeichnungen

4.4. Trennlinien

4.5. Listen

4.5.1. Nummerierte Listen

4.5.2. Aufzählungslisten

4.5.3. Definitionslisten

5. Tabellen und Tabellenattribute

5.1. Aufbau einer Tabelle

5.2. Kopf-, Körper- und Fußbereich einer Tabelle

5.3. Formatierungsmöglichkeiten von Tabellen

5.3.1. Angaben im <table>-Tag

5.3.2. Angaben in Tabellenzeilen (<tr>) und Tabellenzellen (<th>, <td>)

6. Einsetzen von Bildern/Grafiken

6.1. Zusätzliche Attribute

7. Verweise (Links)

7.1. Verknüpfungen von HTML-Seiten

7.2. Verweisziele

7.3. Relative und absolute Adressierung

7.4. Zusätzliche Angaben im <a href...>-Tag

7.5. Anker

7.5.1. Definieren eines Ankers

7.5.2. Anspringen eines Ankers

7.6. Image Map

7.6.1. Definieren einer Image Map

7.6.2. Definieren eines verweissensitiven Bereichs

7.6.3. Grafik für Image Map

8. Frames

8.1. Definieren eines Framesets

8.2. Definieren eines Frames

8.2.1. Angaben im <frame ...>-Tag

8.3. Hinweise zum Konstruieren eines verschachtelten Framesets

8.3.1. Vom Frameset zum Quelltext

8.4. Verweise auf Frames

8.5. Der <noframes>-Tag

8.6. Beispieldefinition eines kompletten Framesets

8.7. Vor- und Nachteile eines Framesets

9. Formulare

[9.1. Definieren eines Formulars](#)

[9.2. Eingabefelder definieren](#)

[9.2.1. Einzeilige Eingabefelder](#)

[9.2.2. Mehrzeilige Eingabefelder](#)

[9.3. Auswahllisten](#)

[9.4. Checkboxes](#)

[9.5. Radiobuttons](#)

[9.6. Buttons allgemein](#)

[9.6.1. Button per <input> definieren](#)

[9.6.2. Button per <button> definieren](#)

[9.6.3. Der Reset-Button](#)

[9.6.4. Der Sende-Button](#)

10. Text- und Seitenformatierungen in HTML

[10.1. Textformatierungen](#)

[10.1.1. Schriftarten definieren](#)

[10.1.2. Schriftgröße definieren](#)

[10.1.3. Schriftfarbe definieren](#)

[10.1.4. Textausrichtung definieren](#)

[10.2. Seitenformatierungen](#)

11. Ergänzende Informationen

[11.1. "Document Type Definitions" \(DTDs\) angeben](#)

[11.1.1. DTD-Angaben für HTML](#)

[11.1.2. DTD-Angaben für XHTML 1.0](#)

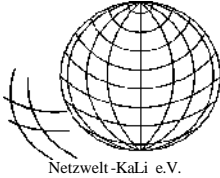
[11.2. Umlaute und Sonderzeichen in HTML](#)

[11.3. Informationen zu einem Impressum](#)

[11.3.1. Beispielimpressum](#)

[11.4. Einige Links zum Thema Webdesign](#)

[11.5. Webprojekte in das Internet stellen](#)



(X)HTML

Netzwelt
Katlenburg-Lindau e.V.
www.Netzwelt-KaLi.de

1. Die Geschichte von (X)HTML

Die Ursprünge des heutigen Internet reichen in die 60er Jahre zurück. Das World Wide Web (WWW) entstand dagegen erst um 1990 und ist Teil des Internet ([siehe auch unseren Artikel: "Geschichte des Internet"](#)).

Der Vorteil des WWW gegenüber den anderen Internetdiensten ist, dass die Informationen gleich beim Aufruf einfach und übersichtlich erscheinen. Um dies zu verwirklichen wurde das Internetprotokoll HTTP (Hypertext Transfer Protokol) und das Dateiformat / die Sprache HTML (Hypertext Markup Language) entwickelt. Gleichzeitig dazu natürlich auch WWW-Server und WWW-Browser, erst Textbrowser und später dann grafische Browser (Mosaic, ...).

HTML ist eine Auszeichnungssprache, die wiederum mit Hilfe von SGML (Standard Generalized Markup Language [existiert seit 1986]) definiert wird. Die Aufgabe von HTML ist es die logischen Bestandteile eines Dokumentes zu definieren, es beschreibt also Dokumente. Damit HTML Systemunabhängig ist, wird es in einem Klartextformat abgelegt. Es kann somit von jedem Texteditor erstellt werden, der reine Textdateien speichern kann.

1.1. HTML-Versionen

Das W3-Konsortium ist für die Standardisierung von HTML verantwortlich und entscheidet somit, welche Befehle (Tags) zu den Sprachstandard von HTML gehören.

HTML 1.0 enthielt nur Befehle für Elemente wie Überschriften, Textabsätze, Grafikreferenzen und Verweise (Links).

HTML 2.0 wurde Ende 1995 der Sprachstandard. Er enthielt wenig neues, gilt aber als kleinster gemeinsamer Nenner.

HTML 3.0 ist nie offiziell geworden und wurde weiter überarbeitet.

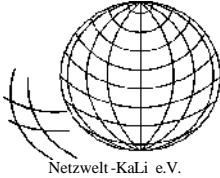
HTML 3.2 wurde Anfang 1997 offizieller Sprachstandard. Ab dieser Version waren auch Tabellen offizieller Bestandteil von HTML.

HTML 4.0 wurde Anfang 1998 offizieller Sprachstandard. Neben Frames gab es jetzt auch die Möglichkeit zur Einbindung von Cascading Style-Sheet (CSS) und Script-Sprachen in HTML. Nach ein paar Überarbeitungen liegt HTML jetzt in der Version 4.01 vor.

XHTML 1.0 wurde Anfang 2000 verabschiedet und ist der Nachfolger von HTML 4.01.

XHTML basiert nicht mehr auf SGML, sondern auf XML und ist dadurch syntaktisch kompatibel zu anderen XML-basierenden Sprachen (SVG, WML, ...). XHTML hat einen strengeren Syntax als HTML.

XHTML 1.1 wurde Ende Mai 2001 verabschiedet. Der Hauptunterschied gegenüber Version 1.0 ist die Entfernung missbilligter Features.



(X)HTML

Netzwelt
Katlenburg-Lindau e.V.
www.Netzwelt-KaLi.de

2. HTML-Aufbau / Aussehen der Tags / Schreibweise / Syntax-Regeln

Die HTML-Steuerbefehle bezeichnet man als Tags. Diese Tags werden durch spitze Klammern `<>`, die Kleiner- und Größerzeichen, markiert. Fast alle HTML-Befehle besitzen einen einleitenden und einen abschließenden Tag. Auf alles was zwischen diesen beiden Tags steht, bezieht sich dieser Befehl.

Beispiel:

```
<h1>Dies ist eine Ueberschrift</h1>
```

`<h1>` ist der einleitende Tag, den abschließenden Tag erkennt man an dem Slash (/) und heißt hier `</h1>`. Alles was zwischen `<h1>` und `</h1>` steht wird also entsprechend formatiert.

In HTML ist die Groß- und Kleinschreibung egal, in XHTML dagegen sollen alle Tags klein geschrieben werden, darum ist zu empfehlen alle Tags klein zu schreiben.

Natürlich können auch mehrere Tags auf ein Element angewendet werden, dabei ist aber auf die richtige Verschachtelung der Tags zu achten.

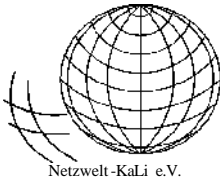
Beispiel:

```
<h2>Dies <tt>ist ein <sup>formatierter</sup> Text</tt></h2>
```

Aussehen:

Dies ist ein formatierter Text

Hier erkennt man, dass der Tag der als letztes geöffnet wird, als erstes wieder geschlossen werden muss.



(X)HTML

Netzwelt
Katlenburg-Lindau e.V.
www.Netzwelt-KaLi.de

3. Einfacher Seitenaufbau

```
<html>
  <head>
  </head>
  <body>
  </body>
</html>
```

Diese HTML-Tags zaubern eine leere, meist weiße Seite in den Browser und somit auf den Monitor.

Dabei umschließt der `<html>`-Tag die HTML-Seite und teilt dem Browser mit, was für Daten kommen (Programm, Bild, Seite, ...).

3.1. Bedeutung und Inhalte von `<head>`

Der `<head>`-Bereich ist der Seitenkopf, er geht von `<head>` bis `</head>` und beinhaltet Informationen zur Seite. Diese Informationen können z.B. sein: der Seitentitel, der Autor, globale Formatierungen, Herausgeber, Schlüsselwörter, ... Im Seitenkopf stehen also Informationen, die auf der Seite selbst nicht angezeigt werden (Titel wird im Browser angezeigt, aber nicht auf der Seite), es sind sozusagen die Eigenschaften dieses Dokumentes.

Beispiele:

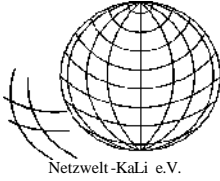
<code><meta name="generator" content="HTML-Editor"></code>	Angabe über den verwendeten Editor
<code><meta name="content-language" content="de"></code>	Angabe über die Sprache des Dokumentes
<code><meta name="author" content="Karl Fahrtmann"></code>	Angabe über den Autor des Dokumentes
<code><meta name="publisher" content="Netzwelt Katlenburg-Lindau e.V."></code>	Angabe über den Herausgeber dieses Dokumentes
<code><meta name="keywords" content="Stichwort1, Stichwort2, Stichwort3"></code>	Auflistung von Stichwörtern zum Seiteninhalt.
<code><meta name="description" content="Kurzbeschreibung"></code>	Kurzbeschreibung des Seiteninhaltes.

3.2. Bedeutung und Inhalte von `<body>`

Der `<body>`-Bereich geht von `<body>` bis `</body>` und enthält darzubietenden Inhalt der Seite. Dieser Inhalt kann bestehen aus Text/Tabellen, Bildern, Videos, Klängen, Geräuschen, Musik, Sprache.

Für die Gestaltung der Seite stehen eine ganze Reihe HTML-Tags zur Verfügung.

Der `<body>`-Bereich beschreibt also den Inhalt der Seite und die Art der Darstellung dieses Inhaltes.



(X)HTML

Netzwelt
Katlenburg-Lindau e.V.
www.Netzwelt-KaLi.de

4. Grundlegende Tags

4.1. Textabsätze <p> und Zeilenumbrüche

Beim Erstellen von Webseiten reicht es nicht aus den Text durch Leerzeichen, Tabulatoren und Zeilenumbrüche (RETURN) zu formatieren, denn ein Browser fasst mehrere Leerzeichen zu eins zusammen und entfernt Zeilenumbrüche die mit RETURN eingefügt wurden sind.

Normaler Text sollte in HTML in <p>-Tags (p=paragraph) eingeschlossen sein, so erzeugt man einen Absatz. Jeder neue Absatz muss wieder in neuen <p>-Tags eingeschlossen werden. In HTML wird es geduldet, wenn an der Stelle an der der Absatz erscheinen soll, nur ein einfaches <p> gesetzt wird. In XHTML ist dies nicht mehr erlaubt, es muss immer ein schließendes Tag vorhanden sein. Darum meine Empfehlung, <p>-Tags immer schließen.

Für einen einfachen Zeilenumbruch in einem Text muss an der entsprechenden Textstelle ein
 (br=break) geschrieben werden. Im Quelltext der Seite kann direkt hinter dem
-Tag weitergeschrieben werden, aber der Übersicht halber sollte der Text in einer neuen Zeile fortgesetzt werden.

Beispiele für <p> und
:

```
<p>Dies ist der 1. Absatz.</p>
<p>Und hier ist der 2. Absatz,<br>mit zwei Zeilen!</p>
```

Und so sieht das Ergebnis aus:

Dies ist der 1. Absatz.

Und hier ist der 2. Absatz,
mit zwei Zeilen!

Am Ende des Browserfensters wird natürlich ein automatischer Zeilenumbruch erzeugt, es sei denn ein Element das nicht umbrochen werden kann ist breiter als das Browserfenster. Wollen Sie verhindern das der Umbruch zwischen zwei bestimmten Wörtern geschieht, so muss das Leerzeichen durch ein geschütztes Leerzeichen ersetzt werden. Ein geschütztes Leerzeichen wird geschrieben (nbsp = no breaking space), das &-Zeichen und das Semikolon sind mit einzugeben. Aus Wort1 Wort2 wird somit Wort1 Wort2. Anstelle von kann auch geschrieben werden.

4.2. Überschriften

In HTML gibt es sechs verschiedene Überschriftenebenen, die dazu genutzt werden können um Hierarchieverhältnisse in Dokumenten dazustellen. Eine Überschrift erster Ordnung erzeugt man mit <h1>Text</h1>. Das h steht für heading (=Überschrift) und die Zahl für die Überschriftenebene. Dabei ist 1 die höchste und 6 die niedrigste Ebene. Es ist darauf zu achten, dass der abschließende Tag die gleiche Nummer hat wie der einleitende Tag. Außerdem erzeugt eine Überschrift einen Absatz davor und dahinter und darf somit auch nicht in <p>-Tags gesetzt werden.

4.3. Textauszeichnungen

Es gibt in HTML physische und logische Textauszeichnungen. Textauszeichnungen dienen dazu Textabschnitte zu formatieren. Wie bei fast allen Tags steht am Anfang des zu formatierenden Textes der einleitende und am Ende des Textes der abschließende Tag. Mit physischen Textauszeichnungen wird das Aussehen des Textes beschrieben, mit logischen dagegen wird die Textfunktion beschrieben.

4.3.1. Physische Textauszeichnungen

<code><i>...</i></code>	kursiv (italic)	normaler und <i>kursiver Text</i>
<code>...</code>	fett (bold)	normaler und fetter Text
<code><tt>...</tt></code>	Fernschreiber (Teletyper)	normaler und Fernschreibertext
<code><big>...</big></code>	größer formatiert	normaler und größerer Text
<code><small>...</small></code>	kleiner formatiert	normaler und kleinerer Text
<code><sup>...</sup></code>	hochgestellt	normaler und hochgestellter Text
<code><sub>...</sub></code>	tiefgestellt	normaler und tiefgestellter Text

4.3.2. Logische Textauszeichnungen

<code>...</code>	Bedeutung "betont" (emphatisch)	normal und <i>betont</i>
<code>...</code>	Bedeutung "stark betont"	normal und stark betont
<code><code>...</code></code>	Bedeutung "Quellcode"	normal und Quelltext
<code><samp>...</samp></code>	Bedeutung "Beispiel"	normal und Beispiel
<code><kbd>...</kbd></code>	Bedeutung "Tastatureingabe"	normal und Tastatureingabe
<code><cite>...</cite></code>	Bedeutung "Zitat"	normal und <i>Zitat</i>
<code><dfn>...</dfn></code>	Bedeutung "Begriffsdefinition"	normal und <i>Begriffsdefinition</i>
<code><acronym>...</acronym></code>	Bedeutung "Abkürzung"	normal und Abkürzung
<code><var>...</var></code>	Bedeutung "Variable"	normal und <i>Variable</i>

Logische Textauszeichnungen eignen sich auch für nichtgrafische Browser und somit auch für Menschen mit Sehschwierigkeiten.

4.4 Trennlinien

In HTML können horizontale Trennlinien `<hr>` eingefügt werden zur optischen Trennung von Elementen. Jede Trennlinie erzeugt einen Absatz und darf somit nicht in `<p>`-Tags eingeschlossen werden. Trennlinien bestehen in HTML nur aus einem Tag, es gibt also kein abschließenden Tag.

Beispiel:

```
...Text...<hr>...weiterer Text...
```

Aussehen:

```
...Text...
```

```
...weiterer Text...
```

In XHTML dagegen muss immer ein abschließender Tag vorhanden sein. Eine Schreibweise ist `<hr />`, somit wird dieser Tag auch von nicht-XHTML fähigen Browsern verstanden.

4.5. Listen

Listen sind Blockelemente, d.h. sie erzeugen vor und hinter der Liste einen Absatz und dürfen somit nicht in `<p>`-Tags eingeschlossen werden.

4.5.1. Nummerierte Listen

Nummerierte Listen sind dafür da, um etwas der Reihe nach durchzunummerieren.

Beispiel:

```
<ol>
  <li>Eintrag 1</li>
  <li>Eintrag 2</li>
  <li>Eintrag 3</li>
</ol>
```

Aussehen:

1. Eintrag 1
2. Eintrag 2
3. Eintrag 3

`` leitet die Definition einer nummerierten Liste ein (1, 2, 3, ...) und `` leitet ein Listenpunkt ein. Durch Angabe von Attributen im einleitenden ``-Tag kann man die Art der Nummerierung ändern.

<code><ol type="A"></code>	Nummeriert die Liste mit A,B,C, ... durch.
<code><ol type="a"></code>	Nummeriert die Liste mit a, b, c, ... durch.
<code><ol type="I"></code>	Römisch nummerierte Liste mit I, II, III, IV, ...
<code><ol type="i"></code>	Römisch nummerierte Liste mit i, ii, iii, iv, ...

4.5.2. Aufzählungslisten

Aufzählungslisten sind dazu da etwas Aufzuführen, ohne dabei Wert auf die Reihenfolge zu legen, also unsortiert.

Beispiel:

```
<ul>
  <li>Eintrag 1</li>
  <li>Eintrag 2</li>
  <li>Eintrag 3</li>
</ul>
```

Aussehen:

- Eintrag 1
- Eintrag 2
- Eintrag 3

`` leitet die Definition einer unsortierten Liste ein (ul=unordered list) und `` leitet ein Listenpunkt ein. Durch Angabe von Attributen im einleitenden ``-Tag kann man die Art der Aufzählungszeichen (Bullet) ändern.

<code><ul type="circle"></code>	Bullet-Typ ist ein Kreis.
<code><ul type="disc"></code>	Bullet-Typ ist ein ausgefüllter Kreis.
<code><ul type="square"></code>	Bullet-Typ ist ein ausgefülltes Quadrat.

4.5.3. Definitionslisten

Definitionslisten sind eine Art Aufzählungslisten. Zu jedem Eintrag kann aber noch eine nähere Beschreibung gemacht werden nach dem Schema: Begriff - Erklärung (z.B. Glossar).

Beispiel:

```
<dl>
  <dt>Eintrag 1</dt>
  <dd>Erklärung/Definition von Eintrag 1</dd>
  <dt>Eintrag 2</dt>
  <dd>Erklärung/Definition von Eintrag 2</dd>
  <dt>Eintrag 3</dt>
  <dd>Erklärung/Definition von Eintrag 3</dd>
</dl>
```

Aussehen:

Eintrag 1
Erklärung/Definition von Eintrag 1

Eintrag 2
Erklärung/Definition von Eintrag 2

Eintrag 3
Erklärung/Definition von Eintrag 3



(X)HTML

Netzwelt
Katlenburg-Lindau e.V.
www.Netzwelt-KaLi.de

5. Tabellen und Tabellenattribute

In HTML können Tabellen auf zweierlei Arten verwendet werden. Einmal um Inhalte tabellarisch darzustellen und zum Anderen für gestalterische Zwecke wie z.B. ein Seitenlayout für eine HTML-Seite.

5.1. Aufbau einer Tabelle

```
<table>
  <tr>
    <td>Zeile 1; Spalte 1</td>
    <td>Zeile 1; Spalte 2</td>
  </tr>
  <tr>
    <td>Zeile 2; Spalte 1</td>
    <td>Zeile 2; Spalte 2</td>
  </tr>
</table>
```

Aussehen:

Zeile 1; Spalte 1	Zeile 1; Spalte 2
Zeile 2; Spalte 1	Zeile 2; Spalte 2

Der Tag `<table>` leitet die Definition der Tabelle ein und `</table>` beendet die Definition. Zwischen `<tr>` und `</tr>` steht jeweils eine Tabellenzeile (tr = tablerow). Und mit `<td>...</td>` wird in der entsprechenden Tabellenreihe eine Tabellenzelle (Spalte) definiert. Jede Zeile sollte die gleiche Anzahl an Spalten haben, die Anzahl der Spalten wird dabei in der ersten Zeile definiert (= Anzahl der `<td>...</td>`). Bei `<td>` handelt es sich um Datenzellen, darüber hinaus gibt es noch die Möglichkeit Kopfzellen zu definieren. Dazu verwendet man `<th>`.

Beispiel:

```
<table>
  <tr>
    <th>Kopfzelle 1</th>
    <th>Kopfzelle 2</th>
  </tr>
  <tr>
    <td>Datenzelle Nr. 1</td>
    <td>Datenzelle Nr. 2</td>
  </tr>
</table>
```

Aussehen:

Kopfzelle 1	Kopfzelle 2
Datenzelle Nr. 1	Datenzelle Nr. 2

5.2. Kopf-, Körper- und Fußbereich einer Tabelle

Man kann den Aufbau einer Tabelle noch weiter als in Kopf- und Datenzellen unterteilen. Es besteht die Möglichkeit eine Tabelle in drei logische Bereiche zu teilen. Diese Bereiche sind der Kopfbereich (<thead>...</thead>), der Körper (<tbody>...</tbody>) und der Fußbereich (<tfoot>...</tfoot>). Durch festlegen dieser Bereiche ist es möglich eine Tabelle einfach zu formatieren.

```
<table border="2" rules="cols">
  <thead>
    <tr>
      <th>Kopfzelle 1</th>
      <th>Kopfzelle 2</th>
    </tr>
  </thead>
  <tfoot>
    <tr>
      <td>Fußzelle Nr.
1</td>
      <td>Fußzelle Nr.
2</td>
    </tr>
  </tfoot>
  <tbody>
    <tr>
      <td>Körperzelle Nr.
1</td>
      <td>Körperzelle Nr.
2</td>
    </tr>
  </tbody>
</table>
```

Aussehen:

Kopfzelle 1	Kopfzelle 2
Körperzelle Nr. 1	Körperzelle Nr. 2
Fußzelle Nr. 1	Fußzelle Nr. 2

Das Aussehen der Tabelle wurde einfach über die Angabe `rules="cols"` definiert.

Beim Arbeiten mit Kopf-, Körper- und Fußbereich ist darauf zu achten diese in der richtigen Reihenfolge zu definieren. Als erstes den Kopfbereich (<thead>...</thead>), danach den Fußbereich (<tfoot>...</tfoot>) und jetzt erst den Tabellenkörper (<tbody>...</tbody>). Kopf- und Fußbereich dürfen in einer Tabelle maximal einmal vorkommen, der Körperbereich darf beliebig oft. Bei einmaligem vorkommen kann <tbody>...</tbody> auch weggelassen werden.

5.3. Formatierungsmöglichkeiten von Tabellen

Durch die Angabe von Attributen in den Tabellen-Tags kann man das Aussehen der Tabelle beeinflussen.

5.3.1. Angaben im <table>-Tag

border	Dicke des Außenrahmens der Tabelle.	<table border="5">
cellspacing	Dicke der Gitternetzlinien, nur mit border sichtbar.	<table border="2" cellspacing="3">
cellpadding	Abstand des Zelleninhaltes zum Zellenrand.	<table cellpadding="5">
frame	Regeln für den Außenrahmen, nur mit border sichtbar (nicht NN 4.x).	<table border="1" frame="void">
	frame="void" kein Rahmen	
	frame="above" nur oberen Rand	
	frame="below" nur unterer Rand	
	frame="hsides" nur oberer und unterer Rand	
	frame="vsides" nur linker und rechter Rand	
	frame="lhs" nur linker Rand	
rules	Regeln für die Gitternetzlinien, nur mit border sichtbar (nicht NN 4.x).	<table border="3" rules="rows">
	rules="none" keine Linien (none=keine)	
	rules="rows" nur Linien zwischen den Tabellenzeilen (row=Zeile)	
	rules="cols" nur Linien zwischen den Tabellenspalten (colum=Spalte)	
	rules="groups" nur Linien zwischen Kopf, Körper und Fuß der Tabelle	
	rules="all" alle Linien zwischen den Tabellenzellen werden gezogen.	
width	Breite der Tabelle. Angabe in Pixel oder Prozent. Hinweis: Die Prozentangabe bezieht sich auf das übergeordnete Element. Ist z.B. das Browserfenster 800 Pixel breit, so ist die Tabelle mit 80% 640 Pixel breit. Befindet sich in dieser Tabelle eine weitere Tabelle mit 80%, so ist diese dann 512 Pixel breit. Die Breitenangabe hat nur Wirkung, wenn der Inhalt kleiner ist.	<table width="300"> =Tabellenbreite 300 Pixel
		<table width="80%"> =Tabellenbreite 80%
height	Höhe der Tabelle. Anwendung wie "width". Attribut "height" ist nicht HTML-Standard und wird nicht von allen Browsern unterstützt.	<table height="400"> =Tabellenhöhe 400 Pixel
		<table height="50%"> =Tabellehöhe 50%

5.3.2. Angaben in Tabellenzeilen (<tr>) und Zellen (<th>, <td>)

width	Breite der Spalte. Siehe "width" bei <table>. Durch Angabe von width="50%" hat die Zelle und damit auch die Spalte eine Breite von 50% der Tabelle. Bei Angabe unterschiedlicher Breiten in einer Spalte zählt die breiteste Zelle als Spaltenbreite. Im Konfliktfall hat die Angabe der Tabellenbreite Vorrang (z.B. Tabellenbreite 100 Pixel und Spaltenbreite 500 Pixel).	<td width="50%">
height	Höhe der Tabellenzeile. Durch Angabe von height="50%" hat die Zelle und damit auch die Zeile eine Höhe von 50% der Tabelle. Bei Angabe unterschiedlicher Höhen in einer Zeile zählt die höchste Zelle als Zeilenhöhe. Im Konfliktfall hat die Angabe der Tabellenhöhe Vorrang (z.B. Tabellenhöhe 100 Pixel und Zeilenhöhe 500 Pixel).	<td height="50%">
align	Horizontales Ausrichten von Zellen. Mögliche Angaben sind: left (=links), center (=zentriert) und right (=rechts).	<td align="left"> <td align="center"> <td align="right">
valign	Vertikales Ausrichten von Zellen. Kann in <tr>, <td> und <th> angegeben werden. Bei Angabe in <tr> zählt die Ausrichtung für die ganze Tabellenzeile, sonst nur für die Zelle. Mögliche Angaben sind: top (=oben), middle (=mitte) und bottom (=unten).	<th valign="left"> <tr valign="middle"> <td valign="bottom">
colspan	Zellen innerhalb einer Tabellenzeile verbinden. Kann auch zusammen mit rowspan verwendet werden.	<table border="2"> <tr> <td colspan="3">Zelle 3 Spalten breit</td> </tr> <tr> <td>X</td><td colspan="2">Zelle 2 Spalten breit</td> </tr> <tr> <td>X</td><td>X</td><td>X</td> </tr> </table>

Aussehen:

Zelle 3 Spalten breit		
X	Zelle 2 Spalten breit	
X	X	X

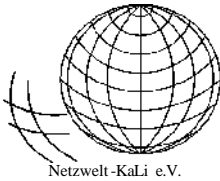
(X)HTML-Programmierung bei Netzwelt Katlenburg-Lindau e.V. - Tabellen

`rowspan` Zellen innerhalb einer Tabellenspalte verbinden. Kann auch zusammen mit `colspan` verwendet werden.

```
<table border="2">
<tr>
<td rowspan="3">Zelle 3 Zeilen
hoch</td><td>X</td><td>X</td>
</tr>
<tr>
<td rowspan="2">Zelle 2 Zeilen
hoch</td><td>X</td>
</tr>
<tr>
<td>X</td>
</tr>
</table>
```

Aussehen:

Zelle 3 Zeilen hoch	X	X
	Zelle 2 Zeilen hoch	X
		X



(X)HTML

Netzwelt
Katlenburg-Lindau e.V.
www.Netzwelt-KaLi.de

6. Einsetzen von Bildern/Grafiken

Zum Einbinden einer Grafik in eine HTML-Datei wird der ``-Tag verwendet.

Beispiel:

```

```

`img` steht für image (=Bild) und `src` für source (=Quelle). Es wird das Bild nicht physisch in die HTML-Datei eingefügt, sondern es wird an der Stelle, an der das Bild später erscheinen soll, auf die Bilddatei verwiesen (referenziert). Der Dateiname der Bilddatei kann natürlich noch Pfadangaben, sowie noch Domainangaben (Serverangaben) beinhalten. Beim Einfügen von Bildern in HTML-Dateien sollten Sie darauf achten, dass diese nicht zu groß sind, bzw. das zu viele Bilder pro HTML-Datei verwendet werden. Bilder verlängern die Ladezeit einer Seite!

Beispiel für die Bildgröße:

Ein Bild welches 13 cm breit und 9 cm hoch ist wird mit einem Scanner mit 200 dpi eingelesen (200 dpi = 200 Punkte pro Inch = 200 Punkte auf 2,54 cm).

Dieses Bild ist nun im Computer 1024 Punkte breit und 709 Punkte hoch.

Ein Monitor stellt maximal 96 dpi da, d.h. das Bild ist auf dem Monitor 27 cm breit und 18,8 cm hoch.

Bei Fotoqualität (32 Bit Farbtiefe) hat das Bild eine unkomprimierte Größe von ca. 2,8 MB.

Verkleinern Sie das Bild auf eine Bildschirmgröße von 13x9 cm, so hat es noch 491x340 Punkte.

Braucht man dazu nur 256 Farben (8 Bit), so hat das Bild eine unkomprimierte Größe von ca. 163 KB, also ca. 17 mal kleiner als das erste Bild.

Anmerkung:

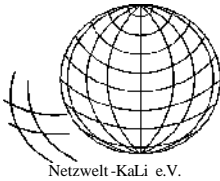
Ein Monitor kann 96 dpi (Dots per inch) anzeigen, d.h. auf 25,4 mm können 96 unterschiedliche Farbpunkte (Dots) angezeigt werden.

Beispielsweise ein 19" Röhren-Monitor:

Er hat ca. 17" Sichtgröße, daraus folgt etwa 13" horizontal. Dies entspricht bei 96 dpi rund 1280 Punkte. Die optimale Einstellung eines 19" Monitors beträgt somit 1280x1024 Pixel. Ist die eingestellte Auflösung größer, so müssen sich teilweise zwei Pixel einen Dot teilen, was bedeutet das bei einem Bild nicht alle Punkte angezeigt werden können (es sei denn es wird vergrößert angezeigt). Will man also ein Bild dessen Größe 13x9 cm ist auch in Originalgröße anzeigen, ohne Pixelverlust, so braucht es nur 491x340 Punkte (96 dpi) haben bei 1280x1024 Auflösung.

6.1. Zusätzliche Attribute

alt	Alternativer Text. Dieser Text erscheint immer, wenn die Grafik nicht angezeigt werden kann, oder soll, oder auch wenn die Maus über dem Bild steht. alt sollte immer angegeben werden, aber wo keine sinnvolle Beschreibung existiert, sollte alt leer bleiben.	<pre> </pre>
width	Angabe zur Breite der Grafik in Pixel, oder in Prozent. Bei der Prozentangabe bezieht sich der Prozentwert wieder auf das übergeordnete Element, z.B. 50% der Fensterbreite.	<pre> </pre>
height	Angabe zur Höhe der Grafik in Pixel, oder in Prozent. Anmerkung: Angaben zur Breite und zur Höhe des Bildes sollten, wenn bekannt, immer gemacht werden. Dadurch erzielt man einen schnelleren Seitenaufbau. Die Größenangaben brauchen aber nicht der wirklichen Bildgröße zu entsprechen. Das Bild wird dann größer oder kleiner dargestellt.	<pre> </pre>
border	Rahmendicke um die Grafik. Angaben in Pixel.	<pre></pre>
align	Definiert wie der Beschriftungstext/Textfluss bei einer Grafik platziert werden soll. <u>align="left"</u> Grafik links, Text rechts. <u>align="right"</u> Grafik rechts, Text links. <u>align="top"</u> Beschriftungstext oben. <u>align="middle"</u> Beschriftungstext mittig. <u>align="bottom"</u> Beschriftungstext unten	<pre> </pre>
hspace	Gibt an, wie viel Pixel Abstand ein Element rechts und links von der Grafik haben soll.	<pre></pre>
vspace	Gibt an, wie viel Pixel Abstand ein Element über und unter der Grafik von der Grafik haben soll.	<pre></pre>
usemap	Name der benutzten Image Map. Wobei Bildname der Name ist, der in der Image Map angegeben wurde.	<pre></pre>



(X)HTML

Netzwelt
Katlenburg-Lindau e.V.
www.Netzwelt-KaLi.de

7. Verweise (Links)

Der große Vorteil von HTML-Dokumenten gegenüber Büchern, Zeitungen, Zeitschriften, oder anderen (Word-) Dokumenten ist die Möglichkeit der weltweiten Verknüpfung von Dokumenten und dem damit verbundenen einfachen hin- und herbewegen zwischen diesen Dokumenten. Es müssen nicht mehr alle Informationen in einem Dokument vorhanden sein, sondern Sie können, sinnvoll gegliedert, auf mehreren Seiten eines, oder mehrerer Projekte verteilt sein. Diese einzelnen Seiten müssen dann nur noch verknüpft werden, damit einfach per Mausklick hin- und hergesprungen werden kann.

Bei der (optischen) Gestaltung eines Projektes ist darauf zu achten, dass alle Seiten ein einheitliches Grundgerüst (Aussehen) haben, damit der Benutzer nicht jedes Mal sich wieder neu orientieren muss. Außerdem sollten die Informationen zu einem Thema nicht über zu viele Seiten verteilt sein, denn jeder zusätzliche Link kostet Besucher. Je spezieller die Informationen, um so mehr Seitensprünge nimmt ein Besucher in Kauf, d.h. um so tiefer können die Informationen in der Seitenhierarchie liegen. Bei der Gestaltung eines Links sollte dem Benutzer deutlich werden, dass es sich um einen Link handelt und wohin er führt, bzw. was ihn dort erwartet. Es ist also schlecht den ganzen Text blau und unterstrichen zu schreiben und dort einen Link zwischen zu setzen, der nur "neue Seite" heißt.

7.1. Verknüpfungen von HTML-Seiten

Eine Verknüpfung (Link) beginnt mit dem einleitenden Tag `` und endet mit dem abschließenden Tag ``.

Beispiel:

```
<a href="seite2.htm">weiter</a>
```

In diesem Fall wird der Text "weiter" im Browser angezeigt, oftmals unterstrichen und andersfarbig. Der Text "weiter" ist also der Link den man anklicken kann und der die Seite mit dem Namen "seite2.htm" aufruft. Anstelle von "weiter" kann auch jeder andere Text dort stehen, auch ganze Sätze mit Zeilenumbruch `
`. Sie können auch anstatt eines Textes dort ein Bild einfügen ([siehe: Bilder/Grafiken](#)), welches dann beim Anklicken die angegebene Seite aufruft.

7.2. Verweisziele

<code></code>	Andere Seite im gleichen Verzeichnis.
<code></code>	andere Seite im untergeordnetem Verzeichnis mit dem Namen "info".
<code></code>	Andere Seite aus dem direkt übergeordnetem Verzeichnis.
<code></code>	Andere Seite aus dem Hauptverzeichnis (ROOT-Verzeichnis).
<code></code>	Selbe Seite zum Sprungziel mit dem Namen "ziel".
<code></code>	Andere Seite, gleiches Verzeichnis zum Sprungziel mit dem Namen "ziel".
<code></code>	Sprung zum Server "www.netzwelt-kali.de" auf die Seite "start.htm".
<code></code>	Die Datei "datei.zip" auf einem FTP-Server.
<code></code>	Verweis auf die E-Mailadresse "info@netzwelt-kali.de". Achtung: Muß vom Browser unterstützt werden. Eingabe ohne die beiden Slashes (//) nach "mailto:". Siehe auch unseren Beitrag "Ergänzungsangaben für den Mailto-Befehl".
<code></code>	Verweis auf die lokale Datei "c:\html\seite.htm" (Windows-System). Slash/Backslash beachten. Nach "file:" müssen drei Slashes (///) angegeben werden, oder "file://localhost/" schreiben.

Es gibt noch zahlreiche weitere Möglichkeiten und Kombinationen, z.B. "/thema2/seite3.htm#ziel4", href="telnet://...", href="news:...", href="gopher://...".

7.3. Relative und absolute Adressierung

Bei der Angabe des Verweiszieses gibt es die Möglichkeit dieses relativ, oder absolut anzugeben. Absolute Adressierung geht von einem immer gleich bleibenden Punkt aus. In unserem Fall von dem Rootverzeichnis.

Beispiele für absolute Adressierung:

```
<a href="http://www.netzwelt-kali.de/rechts/links/index.html">  
<a href="/rechts/links/index.html">
```

Vorteil: Die aufrufende Html-Datei kann nachträglich in jedes beliebige Verzeichnis verschoben werden und die gelinkten Seiten werden immer gefunden.

Bei der relativen Adressierung ist der Ausgangspunkt immer die aktuelle Datei, also die Datei, die auf die anderen verweist.

Beispiele für relative Adressierung:

```
<a href="../../../thema2/seite.htm"> Vom aktuellen Verzeichnis erst zwei  
Verzeichnisebenen höher, dann in das Verzeichnis  
"thema2" und dort die Seite "seite.htm" öffnen.  
<a href="thema3/seite.htm"> Vom aktuellen Verzeichnis in das Verzeichnis  
"thema3" wechseln und dort die Seite "seite.htm"  
öffnen.
```

Vorteil: Die Seiten können auch ohne Internetverbindung benutzt und getestet werden.

7.4. Zusätzliche Angaben im <a href...>-Tag

title	Beim Überfahren des Links mit dem Mauszeiger wird ein zusätzlicher Text angezeigt.	Infos
target	Zielfenstername / Framename in dem der Link geöffnet werden soll.	Neu
	target= "_parent" Anzeige im übergeordnetem Fenster, welches vor dem Frameset aktiv war (z.B. Frameset im Frameset).	
	target= "_top" Anzeige im gesamten Browserfenster.	
	target= "_blank" Anzeige in einem neuen Browserfenster. Das alte bleibt erhalten.	
	target= "Name" Anzeige in einem Fenster / Frame mit dem angegebenen Namen (hier "Name"). Ist dieses Fenster / Frame nicht vorhanden, so wird wie bei "_blank" verfahren.	

Anmerkung: Das Attribut "target" soll nach Wünschen des W3C in Zukunft durch Script-Sprachen, z.B. JavaScript ersetzt werden.

7.5. Anker

In HTML ist es nicht nur möglich über einen Link eine neue Seite zu laden, sondern auch gleich an eine bestimmte Stelle auf einer neuen Seite, bzw. auf der selben Seite zu springen. Um dies zu ermöglichen muss an der anzuspringenden Stelle ein Anker (Sprung- bzw. Verweisziel) definiert sein.

7.5.1. Definieren eines Ankers

Der anzuspringenden Stelle muss ein Name zugewiesen werden, dies geschieht mit Anzuspringender Text. Wobei "Ankername" ein beliebiger Name ist, der aus Buchstaben, ohne Umlaute und ß, sowie Zahlen und Unterstrich bestehen kann. Anstelle eines Textes kann auch ein Bild die Ansprungstelle sein. Es darf nur der Zwischenraum zwischen <a name...> und in HTML nicht leer sein. Pro Seite dürfen beliebig viele Anker definiert sein, aber jeder Ankername darf auf einer Seite nur einmal vorkommen, wobei der Name "case sensitiv" ist, d.h. es wird Groß- und Kleinschreibung unterschieden.

7.5.2. Anspringen eines Ankers

Um einen Anker anzuspringen wird ein ganz normaler Verweis (Link) definiert, der nur noch um #Ankername ergänzt werden muss.

	Normaler Link ohne Anker.
	Link mit Anker. Beim Klick auf diesen Link wird die Seite geladen und gleich zum Anker mit dem Namen "Ankername" gesprungen.
	Wird nur der Ankername angegeben, so wird zum Anker auf der gleichen Seite gesprungen.

Befindet sich der angegebene Anker nicht auf der entsprechenden Seite, so wird zum Seitenanfang gesprungen (= normaler Seitenaufruf).

7.6. Image Map

Eine Image Map ist eine Grafik, die verschiedene anklickbare Bereiche hat. Jeder Bereich dieser Grafik kann dabei einen anderen Verweis öffnen.

7.6.1. Definieren einer Image Map

Das einleitende Tag einer Image Map lautet `<map name="Bildname">` und das abschließende Tag `</map>`. Zwischen dem einleitenden und abschließenden Tag werden die verweissensitiven Bereiche der Grafik definiert.

7.6.2. Definieren eines verweissensitiven Bereichs

Mit `<area ...>` wird der verweissensitive Bereich definiert, dazu werden allerdings noch ein paar zusätzliche Angaben benötigt. Als erstes muss angegeben werden wie der Bereich aussieht. Dazu gibt es drei Möglichkeiten.

<code><area shape="rect" ...></code>	Für einen rechteckigen Bereich.
<code><area shape="circle" ...></code>	Für einen kreisförmigen Bereich.
<code><area shape="polygon" ...></code>	Für einen vieleckigen Bereich.
Danach folgen die Koordinaten dieses Bereiches mit <code>... coords="..."</code> .	
Für <code>"rect"</code>	<code>coords="x1,y1,x2,y2"</code> Dabei ist <code>x1,y1</code> die linke obere Ecke und <code>x2,y2</code> die rechte untere Ecke.
Für <code>"circle"</code>	<code>coords="x,y,r"</code> Dabei ist <code>x,y</code> der Kreismittelpunkt und <code>r</code> der Radius.
Für <code>"polygon"</code>	<code>coords="x1,y1,x2,y2,x3,y3,..."</code> Dabei ist <code>x1,y1</code> die erste Ecke, <code>x2,y2</code> die zweite Ecke, <code>x2,y3</code> die dritte Ecke, usw.

Als letztes muss noch das Verweisziel angegeben werden. dies geschieht über `href="seite.htm"`. Eine komplette Bereichsdefinition sieht nun folgendermaßen aus:

```
<area shape="rect" coords="x1,y1,x2,y2" href="seite.htm">
```

oder so:

```
<area shape="circle" coords="x,y,r" href="seite.htm">
```

Bei der Angabe der Koordinaten auf einer Grafik ist darauf zu achten, dass die linke obere Ecke die Koordinaten `x=1` und `y=1` hat. Die `x`-Werte steigen von links nach rechts und die `y`-Werten von oben nach unten.

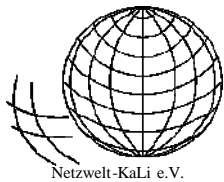
7.6.3. Grafik für Image Map

Die Grafik für eine Image Map wird genauso definiert wie eine gewöhnliche Grafik. Es muss aber das Attribut `usemap="#Bildname"` angegeben werden. `"Bildname"` ist dabei der gleiche Name, der bei `<map name=...>` verwendet wird, aber mit vorangestelltem `#`.

Beispiel:

```

```



(X)HTML

Netzwelt
Katlenburg-Lindau e.V.
www.Netzwelt-KaLi.de

8. Frames

Durch Frames kann man ein Browserfenster in mehrere rechteckige Teilfenster unterteilen. Dabei kann anschließend in jedem Framefenster eine andere HTML-Seite angezeigt werden. So kann z.B. in einem Frame die Navigationsleiste geladen sein, in einem weiteren der Homepage- oder Projektname und in einem dritte Frame der wechselnde Seiteninhalt. Den Möglichkeiten und der Anzahl verwendeter Frames sind dabei kaum Grenzen gesetzt, so können Frames z.B. auch zu Designzwecken eingesetzt werden.

8.1. Definieren eines Framesets

Die ganzen Frames eines Browserfensters nennt man Frameset. Dieses Frameset wird in einer eigenen HTML-Datei definiert, in der auch die Seiten für die einzelnen Frames angegeben werden. Die Definition eines Framesets wird mit `<frameset ...>` eingeleitet und mit `</frameset>` abgeschlossen und muss zwischen `</head>` und `<body>` stehen. Im einleitenden Tag muss noch angegeben werden, ob der freie Bereich horizontal (rows) oder vertikal (cols) geteilt werden soll und in wie viele Teile.

Beispiel 1:

```
<frameset rows="20%,80%">  
Hier werden zwei horizontale Frames definiert. 1. Frame 20% der  
Browserfensterhöhe, 2. Frame 80% der Browserfensterhöhe.  
</frameset>
```

Beispiel 2:

```
<frameset cols="150*,100">  
Hier werden drei vertikale Frames definiert. 1. Frame 150 Pixel  
Breite, 3. Frame 100 Pixel Breite, 2. Frame die Restbreite des  
Browserfensters.  
</frameset>
```

Mit `<frameset>...</frameset>` definieren Sie also die Anzahl und Größe der Frames, sowie die Fensteraufteilung. Zwischen dem einleitenden und abschließenden Tag müssen jetzt noch die Frames definiert werden.

8.2. Definieren eines Frames

Mit `<frame ...>` wird ein Frame definiert. Es gibt keinen abschließenden Tag (in XHTML: `<frame ... />`).

Beispiel:

```
<frame src="seite.htm" name="Fenster_1">
```

Mit "src" wird dem Frame eine Startseite zugewiesen, wird "src" weggelassen, so wird keine Seite geladen. Durch "name" wird dem Framefenster ein Name zugeordnet. Dies ist wichtig um den Frame später ansprechen zu können. Für einen Namen dürfen Buchstaben, ohne Umlaute und ß, sowie Zahlen und Unterstrich verwendet werden. Die Namen "_blank", "_self", "_parent" und "_top" dürfen nicht verwendet werden.

8.2.1. Angaben im <frame ...>-Tag

scrolling	Dient dem Ein- und Ausschalten der Rollbalken eines Framefensters. scrolling="yes" Rollbalken ist immer vorhanden. scrolling="no" Rollbalken ist nie vorhanden, auch wenn der Inhalt größer ist. scrolling="auto" Rollbalken ist nur vorhanden, wenn es der Inhalt erfordert. Grundeinstellung.	<frame src="seite.htm" name="inhalt" scrolling="yes">
noresize	Verhindert, dass der Anwender die Größe der Frames mit seiner Maus ändern kann.	<frame src="seite.htm" name="inhalt" noresize>
marginheight	Gibt den Abstand in Pixel vom oberen und unteren Fensterrand zum Seiteninhalt an.	<frame src="seite.htm" name="inhalt" marginheight="0">
marginwidth	Gibt den Abstand in Pixel vom linken und rechten Fensterrand zum Seiteninhalt an.	<frame src="seite.htm" name="inhalt" marginwidth="0">
frameborder	Gibt an, ob der Framerahmen angezeigt werden soll ("1" oder "yes") oder nicht ("0" oder "no"). Für den Internet Explorer und Netscape sollte noch zum Unterdrücken des Rahmens framespacing="0" border="0" hinzugefügt werden.	<frame src="seite.htm" name="inhalt" frameborder="0" framespacing="0" border="0">

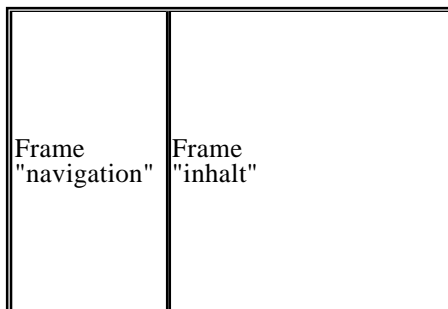
8.3. Hinweise zum Konstruieren eines verschachtelten Framesets

Als erstes zeigen wir den Quelltext und das Aussehen eines einfachen vertikalen Framesets und anschließend den Quelltext und das Aussehen eines horizontalen Framesets.

Beispiel eines vertikalen Framesets:

```
<frameset cols="200,*">  
  <frame src="navi.htm" name="navigation" noresize>  
  <frame src="inhalt1.htm" name="inhalt" noresize>  
</frameset>
```

So etwa würde das Frameset im Browserfenster aussehen:



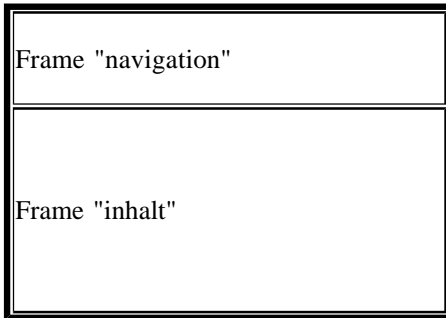
(X)HTML-Programmierung bei Netzwelt Katlenburg-Lindau e.V. - Frames

Sie können natürlich ohne Probleme weitere Spalten in dem vertikalen Frameset einfügen.

Beispiel eines horizontalen Framesets:

```
<frameset rows="200,*">  
  <frame src="navi.htm" name="navigation" noresize>  
  <frame src="inhalt1.htm" name="inhalt" noresize>  
</frameset>
```

So etwa würde das Frameset im Browserfenster aussehen:

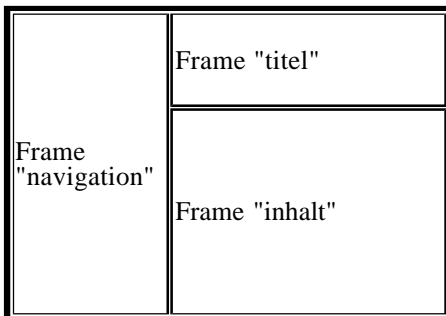


Auch hier können ohne Probleme weitere Reihen in das horizontale Frameset eingefügt werden. Möchten sie aber in einem vertikalen Frameset zusätzlich horizontale Unterteilungen einfügen, oder umgekehrt, so muss an der entsprechenden Stelle in der Definition statt des <frame>-Tags eine weitere <frameset>-Definition eingesetzt werden.

Beispiel:

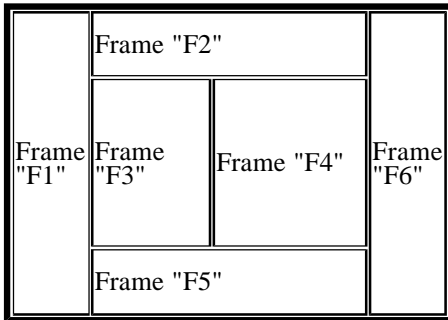
```
<frameset cols="200,*">  
<frame src="navi.htm" name="navigation" noresize>  
  <frameset rows="150,*">  
    <frame src="titel.htm" name="titel" noresize>  
    <frame src="inhalt1.htm" name="inhalt" noresize>  
  </frameset>  
</frameset>
```

So etwa würde das Frameset im Browserfenster aussehen:



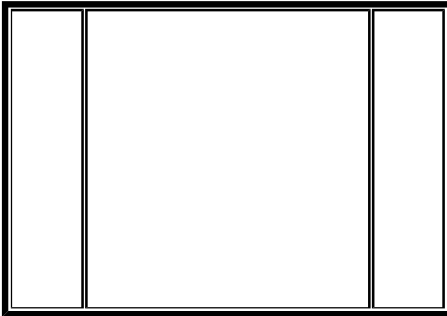
8.3.1. Vom Frameset zum Quelltext

Problem: Sie haben sich ein Frameset ausgedacht und möchten dafür den Quelltext schreiben. Dieses Frameset besitzt einen Rahmen aus vier Frames (F1, F2, F5, F6), einen Navigationsframe (F3) und einen Inhaltsframe (F4). Und so sieht das Frameset etwa aus:

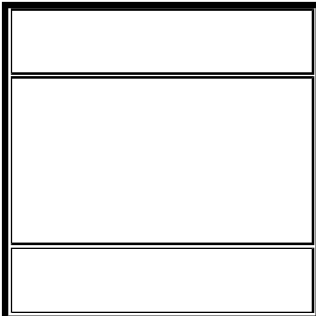


Vorgehen:

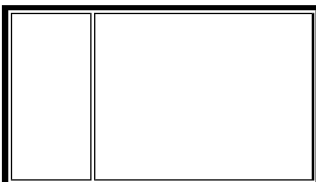
Schritt 1: Suchen Sie sich das äußere Frameset. Das sind die Frames, die über das komplette Browserfenster gehen. In unserem Fall sind es die drei Spalten. Sie bilden ein vertikales Frameset mit drei Frames.



Schritt 2: Danach wird das mittlere Frame in drei Teile aufgeteilt, es wird hier also ein horizontales Frameset mit drei Frames gebildet.



Schritt 3: Als letztes muss wieder das mittlere Frame aufgeteilt werden. Diesmal wird wieder ein vertikales Frameset gebildet, aber nur mit zwei Frames.



Quelltext:

```
<frameset cols="50,* ,50">
  <frame name="F1" noresize scrolling="no">
  <frameset rows="50,* ,50">
    <frame name="F2" noresize>
    <frameset cols="100,*">
      <frame name="F3" src="navi.htm" noresize>
      <frame name="F4" src="seitel.htm" noresize>
    </frameset>
    <frame name="F5" noresize>
  </frameset>
  <frame name="F6" noresize>
</frameset>
```

In dem Frameset aus Schritt 1 werden die drei Frames (Spalten) mit `cols="50,* ,50"` gebildet. Der linke Frame hat den Namen "F1" und der rechte Frame den Namen "F2". Für den mittleren Frame werden die anderen Framesets gebildet.

Im dem Frameset aus Schritt 2 werden die drei Frames (Zeilen) mit `rows="50,* ,50"` gebildet. Der obere Frame hat den Namen "F2" und der untere Frame hat den Namen "F5".

Für den mittlerem Frame wird hier das letzte Frameset gebildet.

Im dem Frameset aus Schritt 3 werden die zwei Frames (Spalten) mit `cols="100,*"` gebildet. Der linke Frame hat den Namen "F3" und der rechte Frame hat den Namen "F4".

8.4. Verweise auf Frames

Wird in einem Frame ein Link angeklickt, der das Format `Text` hat, so wird dieser Link im aufrufenden Fenster, bzw. Frame geöffnet. D.h. das Framefenster in dem dieser Link steht, in diesem Fenster wird die neue Seite angezeigt. Um einen Link und damit die neue Seite in einem anderem Framefenster anzuzeigen, muss der Framename als Ziel (`target`) mit in dem Verweis angegeben werden.

Beispiel:

```
<a href="seite" target="inhalt">Text</a>
```

In diesem Fall wird die neue Seite im Frame mit dem Namen "inhalt" geöffnet. Als Alternative zur Angabe des Zieles in jedem Verweis kann im Dateikopf jeder HTML-Datei, zwischen `<head>` und `</head>`, ein Basiszielfenster angegeben werden. Dies geschieht mit `<base target="Fenstername">`. Jetzt kann aus den Verweisdefinitionen das `target`-Attribut weggelassen werden, es sei denn ein Link soll in einem anderen Fenster als das Basiszielfenster geöffnet werden.

Sollte `target="_top"` angegeben werden, so wird das Frameset beendet und es ist wieder ein normales Browserfenster zu sehen. Für weitere Möglichkeiten als `target` siehe auch "[Zusätzliche Angaben im <a href...>-Tag](#)".

8.5. Der <noframes>-Tag

Es gibt Browser die können keine Frames anzeigen, bzw. man kann diese Eigenschaft abschalten. Ein Projekt das nur auf Frames ausgelegt ist kann mit diesen Browsern kaum, oder gar nicht benutzbar sein. Um Seiten auch für diese Browser wenigstens benutzbar zu halten, kann in eine HTML-Seite ein <noframes>-Bereich eingefügt werden. Dieser Bereich beginnt mit dem Tag <noframes> und endet mit </noframes>. Dazwischen kann dann der Seiteninhalt stehen, der immer dann angezeigt werden soll, wenn der Browser keine Frames anzeigt.

8.6. Beispieldefinition eines kompletten Framesets

In diesem Beispiel definieren wir wieder unser bekanntes, dreigeteiltes Frameset, diesmal in einer kompletten Datei mit allen dazugehörigen Seiten.

Framesetdefinitionsdatei (index.htm)

```
<html>
  <head>
    <title>Framesettitel</title>
  </head>
  <frameset cols="200,*">
    <frame src="navi.htm" name="navi" noresize>
    <frameset rows="150,*">
      <frame src="titel.htm" name="titel" noresize>
      <frame src="seitel.htm" name="inhalt" noresize>
    </frameset>
  </frameset>
  <noframes>
    <body>
      Dieser Inhalt wird angezeigt, wenn kein Frameset angezeigt
werden kann. Evtl. hier zum Weitersurfen Links angeben.
    </body>
  </noframes>
</frameset>
</html>
```

Seite für Frame "navi" (navi.htm)

```
<html>
  <head>
    <title>Framesettitel</title>
  </head>
  <body>
    <h1>Navigation</h1>
    <a href="seitel.htm" target="inhalt">Thema 1 / Seite
1</a><br>
    <a href="seite2.htm" target="inhalt">Thema 2 / Seite 2</a>
  </body>
</html>
```

Seite für Frame "titel" (titel.htm)

```
<html>
  <head>
    <title>Framesettitel</title>
  </head>
  <body>
    <h1>Titel / Thema</h1>
    <noframes>
      Ihr Browser zeigt keine Frames an, oder diese Seite wurde
      direkt aufgerufen.<br>
      Sie haben folgende Wahl:<br>
      <a href="index.htm" target="_top">Frameset laden</a><br>
      <a href="seite1.htm">Thema 1 / Seite 1 laden</a><br>
      <a href="seite2.htm">Thema 2 / Seite 2 laden</a>
    </noframes>
  </body>
</html>
```

Seite für Frame "inhalt" (seite1.htm)

```
<html>
  <head>
    <title>Framesettitel</title>
  </head>
  <body>
    <noframes>
      Ihr Browser zeigt keine Frames an, oder diese Seite wurde
      direkt aufgerufen.<br>
      Sie haben folgende Wahl:<br>
      <a href="index.htm" target="_top">Frameset laden</a><br>
      <a href="seite2.htm">Thema 2 / Seite 2 laden</a>
    </noframes>
    Inhalt der Seite 1 ...
  </body>
</html>
```

Seite für Frame "inhalt" (seite2.htm)

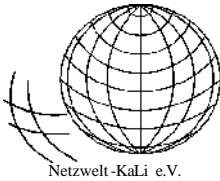
```
<html>
  <head>
    <title>Framesettitel</title>
  </head>
  <body>
    <noframes>
      Ihr Browser zeigt keine Frames an, oder diese Seite wurde
      direkt aufgerufen.<br>
      Sie haben folgende Wahl:<br>
      <a href="index.htm" target="_top">Frameset laden</a><br>
      <a href="seite1.htm">Thema 1 / Seite 1 laden</a>
    </noframes>
    Inhalt der Seite 2 ...
  </body>
</html>
```

8.7. Vor- und Nachteile eines Framesets

Ein Browserfenster in Frames aufzuteilen hat eine gewisse Faszination. Es braucht nur das neu geladen zu werden was sich auch wirklich ändert, der Rest ist immer zu sehen. Frames bieten auch die Möglichkeit Inhalte besser zu vergleichen, oder sie können zu künstlerischen Zwecken (Gestaltung) eingesetzt werden.

Diesen Vorteilen gegenüber stehen eine ganze Menge Nachteile. Hier die wichtigsten:

- Es gibt Browser die keine Frames anzeigen können. Um diese Browser zu berücksichtigen müssen in die Seiten `<noframes>`-Bereiche eingefügt werden, oder gar extra Seiten erstellt werden. Dies erhöht den Programmieraufwand und verlängert damit die Ladezeiten. Viele Webseitenersteller scheuen diesen Aufwand, oder begnügen sich mit der Meldung: "Diese Seite verwendet Frames. Ihr Browser unterstützt keine Frames.". Dies ist für den Benutzer aber wenig hilfreich.
- Um mit Frames sinnvoll zu arbeiten ist eine gewisse Bildschirmauflösung notwendig, abhängig von der Anzahl der verwendeten Frames. Unter anderem aus diesem Grund werden oft die Rollbalken der Frames ausgeschaltet. Benutzer mit geringer Auflösung können dann auf Inhalt, der nicht mehr auf dem Monitor zu sehen ist, nicht mehr zugreifen.
- Es besteht die Möglichkeit das Seiten, die zur Anzeige in einem Frameset gedacht sind, direkt angesprungen werden und somit ohne das dazugehörige Frameset angezeigt werden. Oftmals besteht dann keine Möglichkeit zur Navigation, oder inhaltlichen Zuordnung. Dieser Direktaufruf kann durch Suchmaschinenverweise, Bookmarks/Favoriten, oder Eingabe der Adresse in der Adresszeile zustande kommen. Es besteht zwar die Möglichkeit das Frameset nachzuladen, doch wird hierzu JavaScript benötigt.



(X)HTML

Netzwelt
Katlenburg-Lindau e.V.
www.Netzwelt-KaLi.de

9. Formulare

Formulare sind HTML-Seiten in denen der Benutzer Auswahlen treffen und Eingaben machen kann. Diese gemachten Angaben können danach dann weiter verarbeitet werden.

9.1. Definieren eines Formulars

Die Definition eines Formulars wird mit dem Tag `<form ... >` eingeleitet und mit `</form>` abgeschlossen. Ein Formular darf nicht in `<p>` und `</p>` eingeschlossen werden, da es sich schon um ein Blockelement handelt. Alles was zwischen dem einleitenden und abschließenden Tag steht gehört zum Formular. Dies müssen nicht nur die eigentlichen Elemente des Formulars sein, sondern können auch ganz gewöhnliche HTML-Tags für Tabellen, Bilder, Absätze und Zeilenumbrüche, sowie Text sein. Im einleitenden `<form>`-Tag kann noch angegeben werden, was mit den Formulardaten geschehen soll und wie diese übertragen werden sollen.

Beispiel:

```
<form action="mailto:info@netzwelt-kali.de" method="post"
  enctype="text/plain">
... Formularelemente ....
</form>
```

Mit der Angabe "action" im einleitenden `<form>`-Tag wird die Aktion angegeben, die beim Senden des Formulars ausgeführt werden soll. Im obigen Beispiel wird der Inhalt des Formulars an die angegebene E-Mail-Adresse gesendet. Genau so gut kann aber auch ein Serverseitiges Programm aufgerufen werden. Es wird dann statt der E-Mail-Adresse der Programmname, wenn nötig auch der Pfad, angegeben.

Mit der Angabe "method" wird geregelt wie die Daten aus dem Formular übergeben werden. Bei `method="get"` werden die Daten in die CGI-Umgebungsvariable `QUERY_STRING` gespeichert und können bei Bedarf von (CGI-) Programmen ausgelesen und verarbeitet werden. Bei `method="post"` werden die Daten als Benutzereingabe behandelt. Programme bekommen diese Daten dann über ihre Standardeingabeschnittstelle (`stdin`). Sollen die Daten des Formulars per E-Mail verschickt werden, so muss immer `method="post"` verwendet werden.

Anmerkung: Formularelemente immer in `<form action="">...</form>` einschließen, sonst können einige Browser Probleme mit dem Anzeigen der Elemente bekommen.

9.2. Eingabefelder definieren

Bei Formularen wird zwischen ein- und mehrzeiligen Eingabefeldern unterschieden. Einzeilige Eingabefelder werden benutzt, wenn nur wenige Zeichen/Wörter eingegeben werden sollen, z.B. Name, Passwort, Alter. In mehrzeiligen Eingabefeldern können dagegen längere Texte eingegeben werden, z.B. Geschichten, Rezepte, E-Mail-Text.

9.2.1. Einzeilige Eingabefelder

Einzeilige Eingabefelder werden mit dem Tag `<input ...>` definiert. Es gibt keinen abschließenden Tag, in XHTML wird aber ein abschließender Slash (`/`) vor dem Größerzeichen (`>`) geschrieben.

Beispiel:

```
<input type="text" name="vorname" size="20" maxlength="30">
```

Aussehen:

Wie im Beispiel gesehen gibt es eine Reihe von Attributen für den `<input>`-Tag. Hier eine Auflistung der wichtigsten:

type	Definiert die Art (Typ) der einzugebenen Zeichen (bei Eingabefeldern).	
	type="text" Als Eingabe wird Text erwartet.	
	type="password" Die Eingabe wird als Passwort behandelt. Statt der Buchstaben erscheinen Platzhalter. Doch Achtung: Die Zeichenübermittlung erfolgt im Klartext.	
	type="hidden" Hiermit handelt es sich um ein verstecktes Eingabefeld. D.h. hiermit kann der Programmierer beim Senden weitere, für ihn wichtige, Daten übermitteln.	
name	Hiermit wird dem Eingabefeld ein Name zugeordnet. Dieses Attribut ist erforderlich.	
size	Gibt die Anzahl der anzuzeigenden Zeichen des Eingabefeldes an.	
maxlength	Gibt die Anzahl der maximal einzugebenen Zeichen in das Eingabefeld an.	
value	Hiermit wird dem Eingabefeld eine Textvorbelegung zugewiesen.	<code><input type="text" name="text" size="10" maxlength="30" value="Ein Text"></code>
readonly	Setzt das Eingabefeld auf "Nur lesen", d.h. es kann keine Eingabe mehr gemacht werden, sondern die Anzeige kann nur gelesen werden. In XHTML wird <code>readonly="readonly"</code> geschrieben.	<code><input type="text" name="anzeige" size="10" maxlength="10" value="TOLL" readonly></code> <input type="text" value="TOLL"/>
disabled	Mit diesem Attribut wird das Element ausgegraut, d.h. es wird als inaktiv gekennzeichnet und kann somit auch nicht angesprungen werden. Nur Sinnvoll anwendbar mit Script-Sprachen wie z.B. JavaScript. In XHTML wird <code>disabled="disabled"</code> geschrieben.	<code><input type="text" name="inaktiv" size="10" maxlength="10" value="Nicht aktiv" disabled></code> <input type="text" value="Nicht aktiv"/>
accesskey	Hiermit wird dem Element ein Tastaturkürzel zugewiesen, d.h. beim Drücken von <code>[Alt]+[Taste]</code> wird dieses Element angesprungen.	<code><input type="text" name="taste" size="15" maxlength="40" accesskey="k"></code>

9.2.2. Mehrzeilige Eingabefelder

Mehrzeilige Eingabefelder werden mit dem Tag `<textarea ... >` eingeleitet und mit `</textarea>` abgeschlossen.

Beispiel:

```
<textarea name="geschichte" rows="4" cols="30"></textarea>
```

Aussehen:



Auch beim mehrzeiligem Eingabefeld gibt es Attribute als Ergänzung im einleitenden Tag.

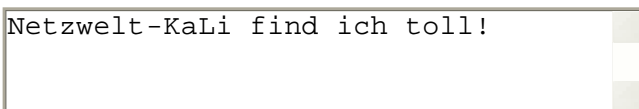
name	Hiermit wird dem Eingabefeld ein Name zugeordnet. Dieses Attribut ist erforderlich.	
rows	Gibt die Anzahl der sichtbaren Zeilen des Eingabefeldes an.	
cols	Gibt die Anzahl der sichtbaren Spalten des Eingabefeldes an.	
readonly	Setzt das Eingabefeld auf "Nur lesen", d.h. es kann keine Eingabe mehr gemacht werden, sondern die Anzeige kann nur gelesen werden. In XHTML wird <code>readonly="readonly"</code> geschrieben.	
disabled	Mit diesem Attribut wird das Element ausgegraut, d.h. es wird als inaktiv gekennzeichnet und kann somit auch nicht angesprochen werden. Nur Sinnvoll anwendbar mit Script-Sprachen wie z.B. JavaScript. In XHTML wird <code>disabled="disabled"</code> geschrieben.	
accesskey	Hiermit wird dem Element ein Tastaturkürzel zugewiesen, d.h. beim Drücken von [Alt]+[Taste] wird dieses Element angesprochen.	
wrap	Steuert den Zeilenumbruch am Eingabefeldzeilenende. <code>wrap</code> ist nicht offizieller HTML-Standard. <code>wrap="virtual"</code> Der Text wird am Zeilenende automatisch umbrochen, die Zeilenumbrüche werden aber nicht mit dem Text übertragen. <code>wrap="physical"</code> Wie "virtual", nur werden die Zeilenumbrüche beim Senden mit übertragen. <code>wrap="off"</code> Stellt den Zeilenumbruch aus.	<code><textarea name="text" rows="5" cols="30" wrap="virtual"></textarea></code>

Hat der Inhalt mehr Zeilen als das Eingabefeld, so kann gescrollt werden. Soll ein mehrzeiliges Eingabefeld mit Text vorbelegt werden, so muss der Text zwischen dem einleitendem und dem abschließendem Tag geschrieben werden.

Beispiel:

```
<textarea name="meinung" rows="3" cols="35">Netzwelt-KaLi find ich toll!</textarea>
```

Aussehen:



9.3. Auswahllisten

Die Definition einer Auswahlliste wird mit `<select ...>` eingeleitet und mit `</select>` abgeschlossen. Hier eine Liste der Attribute im einleitenden `<select>`-Tag:

<code>name</code>	Hiermit wird der Auswahlliste ein Name zugordnet. Dieses Attribut ist erforderlich.	
<code>size</code>	Gibt an, wieviel Zeilen der Auswahlliste auf einmal angezeigt werden sollen. Mit <code>size="1"</code> definiert man eine Dop-Down-Liste.	<code><select name="auswahl" size="1"></code>
<code>multiple</code>	Hiermit gibt man an, dass mehr als ein Eintrag aus der Auswahlliste ausgewählt werden kann. Ohne dieses Attribut kann nur einer ausgewählt werden. Für XHTML muss <code>multiple="multiple"</code> geschrieben werden.	<code><select name="auswahl2" size="5" multiple></code>
<code>disabled</code>	Mit diesem Attribut wird das Element ausgegraut, d.h. es wird als inaktiv gekennzeichnet und kann somit auch nicht angesprungen werden. Nur Sinnvoll anwendbar mit Script-Sprachen wie z.B. JavaScript. In XHTML wird <code>disabled="disabled"</code> geschrieben.	
<code>accesskey</code>	Hiermit wird dem Element ein Tastaturkürzel zugewiesen, d.h. beim Drücken von <code>[Alt]+[Taste]</code> wird dieses Element angesprungen.	

In einer Auswahlliste können beliebig viele Einträge vorhanden sein. Ist die Anzahl der Einträge größer als die Zeilenanzahl der Auswahlliste, so kann gescrollt werden. Eine Definition eines Eintrages in einer Auswahlliste wird mit `<option>` eingeleitet und mit `</option>` abgeschlossen. Der dazwischen stehende Text ist der Listeneintrag. Die Breite einer Auswahlliste richtet sich nach dem längsten Listeneintrag.

Mit `<option selected>Eintrag</option>` kann der angegebene Listeneintrag vorselektiert werden, d.h. er ist in der Liste schon markiert. Beim Absenden eines Formulars werden die markierten Listeneinträge mitgesendet. Soll statt des markierten Eintrages ein anderer Wert gesendet werden, so muss dies wie folgt angegeben werden.

Beispiel:

```
<option value="Hund">Ich habe einen Hund als Haustier</option>
<option value="Katze">Ich habe eine Katze als Haustier</option>
```

Eine Definition einer kompletten Auswahlliste könnte beispielsweise so aussehen:

```
<select name="wetter" size="3">
  <option value="regen">Es regnet</option>
  <option value="schnee">Es schneit</option>
  <option value="sonne">Es scheint die Sonne</option>
</select>
```

Aussehen:

Es regnet Es schneit Es scheint die Sonne

9.4. Checkboxes

Checkboxes sind eine Art Knöpfe, die durch anklicken markiert werden (z.B. durch ein Kreuz), bzw. wieder gelöscht werden. Wie bei den Eingabefeldern wird auch eine Checkbox mit `<input ...>` definiert, danach wird aber ein anderer Typ angegeben.

Beispiel:

```
Vorlieben:<br>
<input type="checkbox" name="hobby" value="kino" checked>
Kino<br>
<input type="checkbox" name="hobby" value="disco"> Disco<br>
<input type="checkbox" name="hobby" value="theater"> Theater<br>
<input type="checkbox" name="hobby" value="essen"> Essen<br>
<input type="checkbox" name="hobby" value="lesen"> Lesen
```

Aussehen:

Vorlieben:

- Kino
- Disco
- Theater
- Essen
- Lesen

Wie im Beispiel gut zu erkennen bekommen alle Checkboxes die zur gleichen Gruppe gehören den gleichen Namen, aber unterschiedliche Werte (value). Befinden sich unterschiedliche Gruppen auf einer Seite, so brauchen diese unterschiedliche Namen. Hier eine Liste der Attribute für Checkboxes:

name	Hiermit wird der Checkbox ein Name und damit eine Gruppe zugeordnet. Dieses Attribut ist erforderlich.	
value	Einem Checkbox-Button wird der angegebene Wert zugewiesen.	<code><input type="checkbox" name="hobby" value="disco"></code>
checked	Hiermit wird die Checkbox vorgewählt. Für XHTML muss <code>checked="checked"</code> geschrieben werden.	<code><input type="checkbox" name="hobby" value="kino" checked></code>
disabled	Mit diesem Attribut wird das Element ausgegraut, d.h. es wird als inaktiv gekennzeichnet und kann somit auch nicht angesprungen werden. Nur Sinnvoll anwendbar mit Script-Sprachen wie z.B. JavaScript. In XHTML wird <code>disabled="disabled"</code> geschrieben.	
accesskey	Hiermit wird dem Element ein Tastaturkürzel zugewiesen, d.h. beim Drücken von <code>[Alt]+[Taste]</code> wird dieses Element angesprungen.	

9.5. Radiobuttons

Radiobuttons sind ähnlich wie Checkboxes eine Art Knöpfe, die durch anklicken markiert werden können. Aber im Gegensatz zu Checkboxes kann bei Radiobuttons aus einer Gruppe immer nur ein Knopf aktiv sein. Auch bei Radiobuttons sollte neben jedem Button ein Text zur Erklärung stehen.

Beispiel:

```
Altersgruppe:<br>
<input type="radio" name="alter" value="alter1"> bis 15 Jahre<br>
<input type="radio" name="alter" value="alter2"> 16 bis 29
Jahre<br>
<input type="radio" name="alter" value="alter3" checked> 30 bis
44 Jahre<br>
<input type="radio" name="alter" value="alter4"> 45 bis 59
Jahre<br>
<input type="radio" name="alter" value="alter5"> 60+ Jahre
```

Aussehen:

Altersgruppe:

- bis 15 Jahre
- 16 bis 29 Jahre
- 30 bis 44 Jahre
- 45 bis 59 Jahre
- 60+ Jahre

Hier nun mögliche Attribute für Radiobuttons:

name	Hiermit wird dem Radiobutton ein Name und damit eine Gruppe zugeordnet. Dieses Attribut ist erforderlich.	
value	Einem Checkbox-Button wird der angegebene Wert zugewiesen.	<code><input type="radio" name="alter" value="alter1"></code>
checked	Hiermit wird die Checkbox vorgewählt. Für XHTML muss <code>checked="checked"</code> geschrieben werden.	<code><input type="radio" name="alter" value="alter3" checked></code>
disabled	Mit diesem Attribut wird das Element ausgegraut, d.h. es wird als inaktiv gekennzeichnet und kann somit auch nicht angesprungen werden. Nur Sinnvoll anwendbar mit Script-Sprachen wie z.B. JavaScript. In XHTML wird <code>disabled="disabled"</code> geschrieben.	
accesskey	Hiermit wird dem Element ein Tastaturkürzel zugewiesen, d.h. beim Drücken von <code>[Alt]+[Taste]</code> wird dieses Element angesprungen.	<code><input type="radio" name="taste" value="test" accesskey="t"></code>

9.6. Buttons allgemein

Außer den Radiobuttons gibt es noch weitere Buttons, die aber teilweise nur in Verbindung mit einer Scriptsprache, wie z.B. JavaScript, sinnvoll einzusetzen sind.

9.6.1. Button per `<input>` definieren

Beispiel:

```
<input type="button" value="Beschriftung" name="Name1">
```

Aussehen:



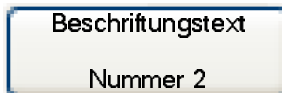
Mit `<input type="button" ...>` wird der Button definiert und mit `value` wird dem Button ein Beschriftungstext zugewiesen. Nur mit HTML kann der Button zwar angeklickt werden, er hat aber sonst keine Wirkung.

9.6.2. Button per `<button>` definieren

Beispiel:

```
<button type="button" value="Beschriftung 1" name="Name2">
Beschriftungstext<br>
Nummert 2
</button>
```

Aussehen:



Die Definition dieses Buttons wird mit `<button type="button" ...>` eingeleitet und mit `</button>` abgeschlossen. Diese Art der Definition ist erst ab HTML 4.0 erlaubt und funktioniert im Internetexplorer ab der Version 4, in Netscape aber leider in der Version 4 noch nicht. Das Attribut `value` dient der Beschriftung, wird aber vom IE 4.0 noch ignoriert. Dafür wird der Button mit allem was zwischen dem einleitenden und abschließenden Tag steht beschriftet, sogar mit einer Grafik. Auch dieser Button hat ohne Einsatz einer Script-Sprache keine Wirkung.

9.6.3. Der Reset-Button

Beim Reset-Button handelt es sich um einen Button, der alle gemachten Eingaben in einem Formular wieder löscht. Er funktioniert mit reinem HTML, es ist also kein JavaScript nötig. Ein solcher Button wird definiert wie ein normaler Button mit `<input ...>`, oder mit `<button ...>`. Aber statt `type="button"` muss `type="reset"` angegeben werden.

Beispiel:

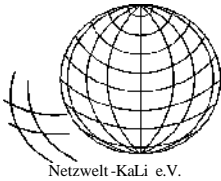
```
<input type="reset" value="Beschriftung" name="Name">
```

oder:

```
<button type="reset" value="Beschriftung 1" name="Name">
Beschriftung 2
</button>
```

9.6.4. Der Sende-Button

Dieser Button wird verwendet, um ein ausgefülltes Formular abzusenden. Der Sende-Button funktioniert nach dem gleichem Prinzip wie der [Reset-Button](#), als Typ muss `type="submit"` stehen. Was beim Absenden des Formulars geschehen soll, wird im einleitenden `<form>`-Tag mit dem `action`-Attribut angegeben ([siehe "9.1 Definieren eines Formulars"](#)).



(X)HTML

Netzwelt
Katlenburg-Lindau e.V.
www.Netzwelt-KaLi.de

10. Text- und Seitenformatierungen in HTML

Mit HTML hat man nicht nur die Möglichkeit den Inhalt eines Dokumentes zu beschreiben, sondern auch das Aussehen des Inhaltes. Nach dem Willen des W3C soll diese Aufgabe aber CSS (Cascading Style-Sheet) übernehmen. CSS bietet mehr Formatierungsmöglichkeiten und kann bei Bedarf in eine separate Datei ausgelagert werden und somit von jedem HTML-Dokument hinzu geladen werden. Änderungen brauchen dann nur einmal vorgenommen werden und wirken sich auf das ganze Projekt aus. Wer mit dem Gedanken spielt ein größeres Projekt zu gestalten, sollte sich mit CSS auseinander setzen. In diesem Kapitel stellen wir die Möglichkeiten von HTML vor.

10.1. Textformatierungen

In [Kapitel 4.3](#) haben wir schon einige Textauszeichnungen kennen gelernt, die auch zur Textformatierung eingesetzt werden können.

10.1.1. Schriftarten definieren

Mit `<font face=" ...` wird die Schriftartendefinition eingeleitet und mit `` abgeschlossen. Alles an Text was zwischen dem einleitendem und abschließendem Tag steht wird in dieser Schriftart dargestellt, wenn sie vorhanden ist. Es besteht auch die Möglichkeit mehrere Schriftarten, durch Komma getrennt, anzugeben.

Beispiel:

```
<font face="Verdana, Arial, sans-serif">
```

Sollte die erste Schriftart (hier Verdana) nicht auf dem anzeigendem Rechner vorhanden sein, so wird die zweite (Arial) verwendet. Ist auch diese nicht vorhanden, so wird eine serifenlose Schrift verwendet. Bei der Gestaltung einer Webseite sollte darauf geachtet werden weit verbreitete Schriftarten zu verwenden, z.B. Verdana, Arial, Courier, Times New Roman, Symbol. Außerdem gibt es ein paar reservierte Schriftnamen die verwendet werden können. Dies sind:

<code>serif</code>	Serifenschrift wie Times New Roman
<code>sans-serif</code>	Serifenlose Schrift wie Arial
<code>cursive</code>	Kursive (schräge) Schrift
<code>fantasy</code>	Zierschrift
<code>monospace</code>	Konstantschrift wie Courier

Sollten die angegebenen Schriftarten auf dem System nicht vorhanden sein, so wird die Standard-Systemschrift verwendet.

10.1.2. Schriftgröße definieren

Eingeleitet wird die Definition mit `<font size=" ...` und abgeschlossen mit ``. Als Schriftgröße können absolute Zahlen zwischen 1 und 7 angegeben werden, wobei 3 die Normalschriftgröße ist, oder relative Zahlen in der Form von `+Zahl` und `-Zahl`. Zu beachten ist das die Schriftgröße nicht in Pixel angegeben wird und somit abhängig von der benutzerdefinierten Schriftgröße ist.

10.1.3. Schriftfarbe definieren

Die Schriftfarbendefinition wird mit `<font color=" ...` eingeleitet und mit `` abgeschlossen. Die Farbe selbst wird entweder als Hexadezimalwert angegeben (je zwei Stellen für R (rot), G (grün) und B (blau), also von #000000 bis #ffffff), oder als Farbname wie z.B. `red`, `green`, `blue`, `yellow`, `black`, `white`.

Beispiel:

```
<font color="#000000">Schwarz</font>
<font color="blue">Blau</font>
```

Anmerkung: Die font-Angaben sind auch kombinierbar.

Beispiel:

```
<font face="Verdana, Arial" size="+4" color="red">Text</font>
```

10.1.4. Textausrichtung definieren

Um Text auszurichten muss in den entsprechenden Tags wie z.B. <p> oder <h...> das Attribut align= hinzugefügt werden, sowie die Ausrichtungsart.

```
align="left"    Linksbündiger Text
align="right"   Rechtsbündiger Text
align="center"  Zentrierter Text
align="justify" Blocksatz
```

Beispiel:

```
<p align="right">Dieser Text wird an der<br>
rechten Seite ausgerichtet.</p>
```

Größere Bereiche und sogar ganze Seiten können mit <div> ausgerichtet werden.

Beispiel:

```
<div align="center">Seiteninhalt wie Text, Bilder, Tabellen</div>
```

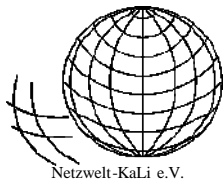
10.2. Seitenformatierungen

Seitenformatierungen sind Attribute, die im einleitenden <body>-Tag platziert werden:

text=	Hiermit wird die Farbe für den Seitentext angegeben. Für Farbangaben siehe "10.1.3. Schriftfarbe definieren" .
link=	Gibt die Farbe für noch nicht besuchte Links an.
vlink=	Dies ist die Farbe für bereits besuchte Links (visited links).
alink=	Und hiermit wird die Farbe für den gerade aktivierten Link angegeben.
bgcolor=	Mit diesem Attribut kann die Hintergrundfarbe einer Seite angegeben werden. Dieses Attribut sollte auch bei Verwendung einer Hintergrundgrafik verwendet werden, da es möglich ist, dass die Seite ohne diese Grafik nicht lesbar ist.
background=	Hiermit wird eine Hintergrundgrafik für die Seite angegeben. Sollte die Seite größer als die Grafik sein, so wird das Bild so oft wie nötig wiederholt. Der Internet Explorer erlaubt in diesem Zusammenhang das Attribut bgproperties="fixed". Dies hat zur Wirkung, dass der Hintergrund nicht mitscrollt, es entsteht ein Wasserzeichen-Effekt.

Beispiel:

```
<body bgcolor="#ffffff" text="#000000" link="blue" vlink="red"
alink="green" background="bild.gif" bgproperties="fixed">
```



(X)HTML

Netzwelt
Katlenburg-Lindau e.V.
www.Netzwelt-KaLi.de

11. Ergänzende Informationen

11.1. "Document Type Definitions" (DTDs) angeben

Eine DTD ist eine HTML-Spezifikation. Mit ihr wird angegeben, welche Sprachversion zum Erstellen dieser Seite verwendet wurden ist. Es ist nicht zwingend vorgeschrieben diese Angaben zu machen, bzw. sich an die gemachten Angaben zu halten.

Empfehlung: Man sollte diese Angaben machen und sich daran halten. Es ist nicht auszuschließen das neue Browser irgendwann sich mal daran halten werden.

Die DTD-Angaben sind die ersten Angaben einer HTML-Seite, noch vor dem einleitenden `<html>`.

11.1.1. DTD-Angaben für HTML

<code><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 2.0//EN"></code>	Hiermit wird HTML-Sprachstandard 2.0 angegeben.
<code><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN"></code>	Hiermit wird HTML-Sprachstandard 3.2 angegeben.
<code><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"></code>	Hiermit wird HTML-Sprachstandard 4.0 angegeben.
<code><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"></code>	Hiermit wird HTML-Sprachstandard 4.0 angegeben, sowie die Verwendung von Style-Sheet oder Script-Sprachen.
<code><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN"></code>	Hiermit wird HTML-Sprachstandard 4.0 angegeben, sowie dass in dieser Datei ein Frameset definiert wird.

Für HTML 4.0 können noch die Adressen der DTDs angegeben werden. Diese sind:

<code>"http://www.w3.org/TR/REC-html40/loose.dtd"</code>	Für HTML 4.0 Standard, einschließlich der Verwendung von Elementen die entfernt werden sollen.
<code>"http://www.w3.org/TR/REC-html40/strict.dtd"</code>	Für strikte Haltung an den HTML 4.0 Standard.
<code>"http://www.w3.org/TR/REC-html40/frameset.dtd"</code>	Für HTML 4.0 Standard und in dieser Datei wird ein Frameset definiert.

Beispiel:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"
"http://www.w3.org/TR/REC-html40/loose.dtd">
```

11.1.2. DTD-Angaben für XHTML 1.0

<pre><!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"></pre>	Für XHTML 1.0 und strikte Haltung daran.
<pre><!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"></pre>	Hiermit wird XHTML-Sprachstandard 1.0 angegeben, sowie die Verwendung von Style-Sheet oder Script-Sprachen.
<pre><!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd"></pre>	Hiermit wird XHTML-Sprachstandard 1.0 angegeben, sowie dass in dieser Datei ein Frameset definiert wird.

Außerdem sollte vor diesen XHTML 1.0 DTDs noch eine XML-Deklaration geschrieben werden.

Beispiel:

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

11.2. Umlaute und Sonderzeichen in HTML

Normalerweise können in HTML-Dateien nicht ohne weiteres Umlaute und Sonderzeichen enthalten sein und müssen deshalb maskiert werden. Mit den Umlauten kann das ganze noch umgangen werden, wenn ein entsprechender Zeichensatz im Seitenkopf angegeben wird.

Beispiel:

```
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">
```

Da aber Webseiten weltweit aufrufbar sind, sollte man sich angewöhnen auch Umlaute zu maskieren.

Zeichen Ersetzen durch

ä	ä
Ä	Ä
ö	ö
Ö	Ö
ü	ü
Ü	Ü
ß	ß
<	<
>	>
&	&
"	"
€	€

Beispiel:

Aus muß wird muß und aus Äpfel wird Äpfel.

11.3. Informationen zu einem Impressum

Ein Impressum brauchen alle gewerblichen Homepages, egal ob dort E-Commerce betrieben wird oder sie nur informativen Zwecken dient. Fehlt ein Impressum dann droht ein Bußgeld, oder eine kostenpflichtige Abmahnung eines Mitbewerbers. Rein private Homepage benötigen kein Impressum, aber schon allein aus Höflichkeit sollte ersichtlich sein wem die Homepage gehört. Das Impressum sollte so platziert sein, dass es leicht zugänglich, erkennbar und von jeder Seite erreichbar ist (Achtung: Auch bei Frames!).

11.3.1. Beispielimpressum

Je nach Art eines gewerbetreibenden Unternehmen gibt es unterschiedliche Pflichtangaben. In diesem Beispiel handelt es sich um ein Impressum eines eingetragenen Vereines (e.V.).

Beispiel:

Sportverein e.V.
Sportplatz 1
37191 Katlenburg-Lindau
Telefon: +49 5552 0815
Telefax: +49 5552 0816
E-Mail: info@sportverein-kali.de
Internet: www.sportverein-kali.de

Vertretungsberechtigter Vorstand: Max Sportmann, Hans Sportler
(Vorsitzender), Ilse Sieger
Registergericht: Amtsgericht Northeim
Registernummer: VR 000
Haftungshinweis: Trotz sorgfältiger inhaltlicher Kontrolle
übernehmen wir keine Haftung für die Inhalte externer Links. Für
den Inhalt der verlinkten Seiten sind ausschließlich deren
Betreiber verantwortlich.

Hinweis:

Die Angabe eines Postfachs als Adresse ist nicht ausreichend.
Auch mit eine Haftungshinweis kann man für externe Links mit zur Verantwortung gezogen werden. D.h. mindestens beim Setzen der Links sollte die fremde Seite einmal kontrolliert werden und danach, je nach Art der anderen Webseite, in unregelmäßigen Abständen.

11.4. Einige Links zum Thema Webdesign

http://www.digi-info.de/webimpressum/	Webimpressum-Generator
http://www.metacolor.de/	Informationen über Farben
http://www.mywebresource.de/html/guides/webfarben.html	Informationen über Farben auf Webseiten
http://validator.w3.org/	Onlineprüfung der Webseiten auf Fehler
http://www.webmasterplan.com/de/	Diverse Dienste für Webmaster
http://www.netzwelt-kali.de/rechts/links/web.htm	Linksammlung mit Seiten zum Thema Webdesign
http://www.meybohm.de/	HTML Editor Phase 5 ² (Freeware)
http://www.teamone.de/selfhtml/index.htm	SelfHTML

11.5. Webprojekte in das Internet stellen

Nachdem eine Webseite bzw. ein Webprojekt fertig gestellt wurden ist, muss dieses noch in das Internet gebracht werden. Dazu braucht man Speicherplatz auf einem Rechner der an das Internet angebunden ist (Web-Server). Von dem Anbieter des Speicherplatzes bekommt man die benötigten Daten um auf diesen Platz zuzugreifen. Außerdem wird zur Übertragung der Daten ein FTP-Programm (FTP = File Transfer Protocol) benötigt. Um mit Hilfe des FTP-Programms Kontakt herzustellen, muss die Internetadresse des Web-Serverspeicherplatzes angegeben werden.

Beispiel:

```
ftp://ftp.netzwelt-kali.de
```

Danach wird noch ein zulässiger Benutzername und ein gültiges Passwort verlangt. Grafische FTP-Programme sind oftmals ähnlich einem Dateimanager (z.B. Explorer) aufgebaut. Auf der einen Seite sieht man die heimische Festplatte, auf der anderen Seite den Web-Serverspeicherplatz. Hier können jetzt Verzeichnisse erstellt, Dateien kopiert, verschoben und gelöscht werden.

Hinweis:

- Datei- und Verzeichnisstruktur auf dem Web-Server sollte genau so wie auf dem lokalen Computer sein.
- Viele Web-Server unterscheiden Groß- und Kleinschreibung. Am Besten die Datei- und Verzeichnisnamen durchweg klein schreiben.
- Kontrollieren ob alle Daten übertragen wurden sind. Evtl. Dateien nach Datum sortiert anzeigen lassen.
- Die neuen Seiten nach dem Hochladen gleich im Web-Browser anzeigen lassen (Funktionstest).
- Eine Fehlerquelle ist, dass die Daten nicht im ASCII-Format, sondern im Binärformat übertragen werden.