

## HTML

### Hypertext Markup Language



**Autor:** Dipl.-Math.  
E. Engelhardt

**Stand:** 10. Nov. 2007

## Inhalt



- ◆ HTML – Hypertext Markup Language 3 - 36
- ◆ CSS - Cascaded Style Sheets 37, 38
- ◆ CGI - Common Gateway Interface 39 – 45
- ◆ XML – eXtensible Markup Language 46, 47
- ◆ JavaScript 48 - 70
- ◆ VRML97 - Virtual Reality Modeling Language 71 - 96

- ◆ SGML entstand als allgemeine Dokumentbeschreibungssprache aus GML (General Markup Language) bei IBM in den siebziger Jahren
- ◆ 1986 wurde die ISO-Norm 8879 unter dem Namen SGML (Standard Generalized Markup Language) verabschiedet
- ◆ Standard, um Dokumente in Form einer Grammatik zu beschreiben
- ◆ Tim Berners Lee wählte die SGML-Definition als Basis für HTML (formale Anwendung der SGML-Definition)
- ◆ Mit Hilfe einer „Document Type Definition“ (DTD) werden die Befehle (Tags) festgelegt

- ◆ HTML 1.0 war Sammlung sehr einfacher Befehle
- ◆ „Mosaic“ war der erste WWW-Browser mit grafischer Oberfläche
  - Durch „Mosaic“ wurden neue Befehle integriert (über HTML 1.0 hinaus)
- ◆ HTML 2.0 war Obermenge von 1.0 + neue Befehle, die teilweise als „obsolete“ (überflüssig) definiert sind (Kompatibilitätsgründe)
- ◆ Vorschlag HTML 3.0 enthielt Tabellen, Farben, Schriftarten, Schriftgrößen, Formeln, Klassen, usw.
- ◆ 1996 HTML 3.2 war gegenüber HTML 3.0 reduziert
- ◆ Seit dem 18. Dezember 1997 gilt HTML 4.0

- ◆ Keine Programmiersprache, sondern nur eine Seitenbeschreibungssprache
- ◆ Dem aktuellen Text werden bestimmte Formatierungen mit einem so genannten „tag“ zugewiesen
- ◆ Einfache Verbindungen von Dokumenten weltweit durch „Hyperlinks“
- ◆ Multimedial durch die Einbindung von Grafiken, Sound, Videos, ...
- ◆ Erweiterbarkeit für diverse Anwendungen durch sog. Plug-Ins für den Browser um z.B. VRML darzustellen
- ◆ Bestandteil von Betriebssystem (Windows 98, Windows 2000, Windows XP, ...)

- ◆ HTML-Dokumente sind reine ASCII-Textdateien und damit mit jedem reinen ASCII-Editor bearbeitbar
- ◆ Damit plattformunabhängig
- ◆ In verschiedenen Anzeigeformaten (Größe) darstellbar
- ◆ Auch auf Systemen mit 7 Bit Zeichenlänge lauffähig (untere Hälfte der ASCII-Tabelle)
- ◆ Auch für offline-Dokumentation geeignet (CDs, Bücher, Handbücher in HTML, usw.)

- ◆ Nach Urheberrechtsgesetz §2 sind Texte, Bilder, Songs, Filme, Filmausschnitte urheberrechtlich geschützt
- ◆ Homepages dürfen in der Regel nicht kopiert werden
- ◆ GEMA (Gesellschaft für musikalische Aufführungs- und mechanische Vervielfältigungsrechte) ist für Musikstücke zuständig, d.h. es müssen
- ◆ Das Teledienstgesetz §5 regelt Verantwortung für eigene Inhalte und für Links, die angegeben werden
- ◆ Das Internet ist kein rechtsfreier Raum! Straftatbestände wie Beleidigung, Verleumdung usw. sind auch auf das Internet anwendbar
- ◆ Gästebücher sollten mindestens einmal täglich auf Inhalte überprüft werden

- ◆ Tags stehen immer in „<“ und „>“, z.B. **<html>**, ...
  - sollten mit Kleinbuchstaben geschrieben werden (XML)
- ◆ Zu jedem der o.g. Anfangs-Tags gibt es korrespondierende End-Tags mit einem „/“, z.B. **<h1>** bla bla **</h1>**
- ◆ Zwei zusammen gehörende Tags werden als „Container“ bezeichnet
- ◆ Ausnahmen, z.B. Zeilenumbruch **<br>** hat keinen korrespondierenden End-Tag
- ◆ Verschachtelung von Tags ist möglich
- ◆ Für Tags können Attribute (Parameter) angegeben werden, z.B. **<h1 align="left">** blablub **</h1>**
  - die Werte der Attribute sollten in Anführungszeichen gesetzt werden (hier wirklich die Zeichen " ... ")

<code>&lt;!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 //EN"&gt;</code>	
<code>&lt;html&gt;</code>	Beginn der HTML-Anweisungen
<code>&lt;head&gt;</code>	Beginn des Kopfteils
<code>&lt;title&gt;</code>	Beginn des Titels
Text des Titels	Titel der HTML-Datei in Titelzeile
<code>&lt;/title&gt;</code>	Ende des Titels
<code>&lt;/head&gt;</code>	Ende des Kopfteils
<code>&lt;body&gt;</code>	Beginn des Hauptteils
Text	Haupttext der HTML- Seite,
Grafik	der im Browser dargestellt wird.
Text	Eingebettet sind Grafiken und
Link	Links zu anderen Seiten
<code>&lt;/body&gt;</code>	Ende des Hauptteils
<code>&lt;/html&gt;</code>	Ende der HTML-Anweisungen

- ◆ Laut HTML-Spezifikation soll jede Seite mit der „**D**ocument **T**ype **D**eclaration“ (DTD) begonnen werden (SGML-Befehl)
  - `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 //EN">` für Seiten, die streng nach der Spezifikation geschrieben wurden
  - `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">` für Seiten mit unerwünschten Tags
  - `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN">` für Seiten mit Frameset
  - `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 //EN" "http://www.w3.org/TR/html4/strict.dtd">` zur Abfrage der Spezifikation durch URL (Uniform Resource Locator)

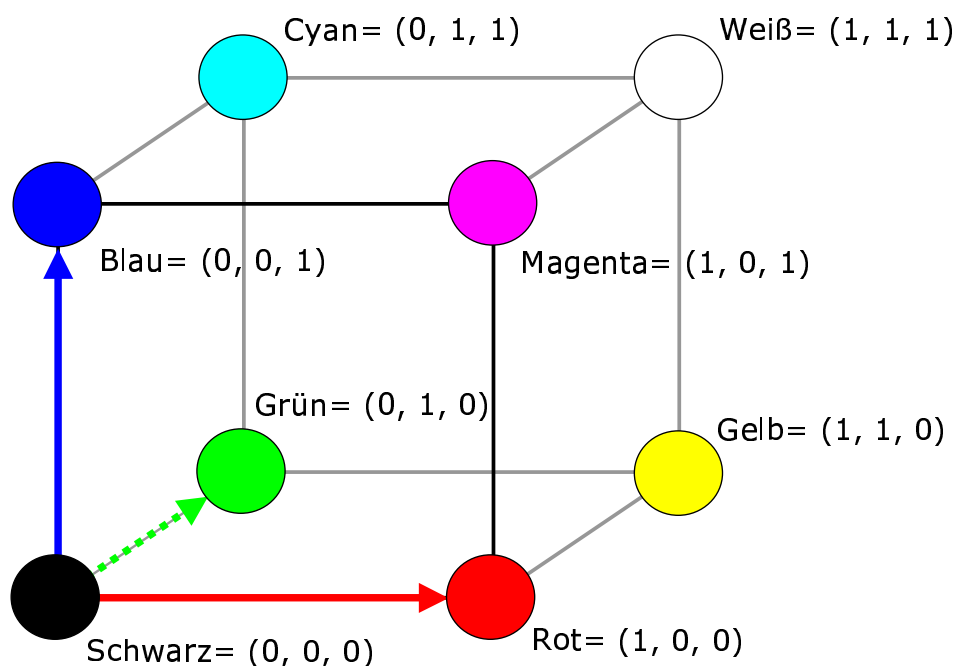
Entity	Kürzel	Entity	Kürzel
Ä	&Auml;	<	&lt;
Ö	&Ouml;	>	&gt;
Ü	&Uuml;	½	&frac12;
ä	&auml;	¼	&frac14;
ö	&ouml;	¾	&frac34;
ü	&uuml;	'	&#146;
ß	&szlig;	,	&#130; (Komma)
&	&amp;	'	&#145;
©	&copy;	„	&#132;
®	&reg;	“	&#147;
™	&trade;	»	&#187;
§	&sect;	«	&#171;

## Satzzeichen

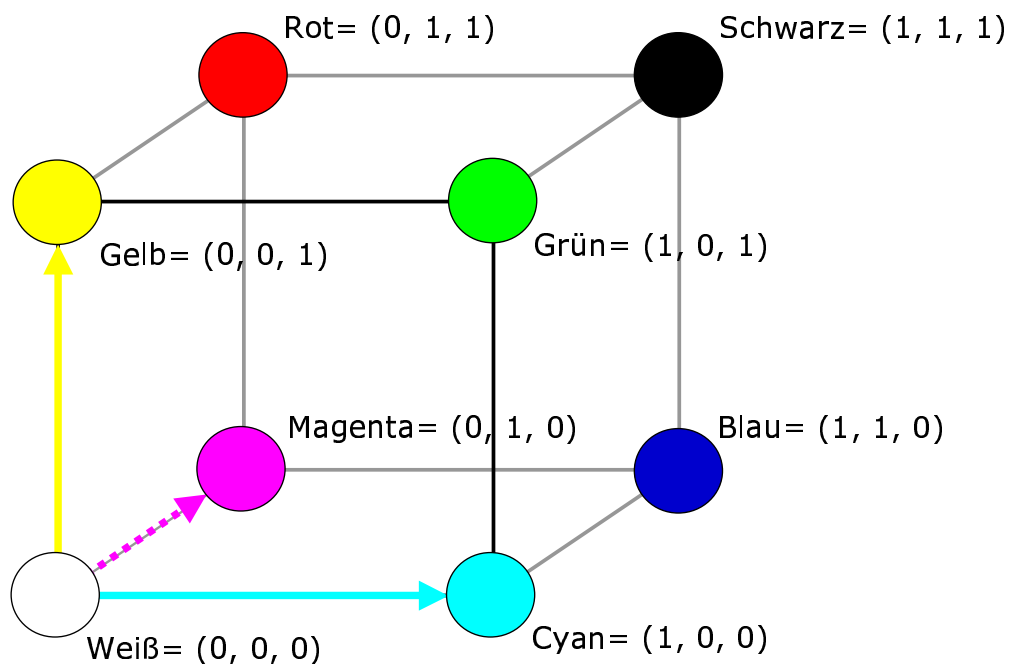
- ◆ **Der PC ist keine Schreibmaschine!**
- ◆ Als Apostroph wird ' verwendet, nicht ´
  - Apostrophe stehen für Auslassungen, nicht für Genitiv oder Plural (**falsch:** Müller's Kiosk, Büro`s frei)!
  - **richtig:** das war's, was gibt's?, Lars' Kneipe
- ◆ Anführungszeichen im Deutschen sind: „...“ oder »...«. (**falsch:** "Alte Feuerwache", der Film "Rossini").  
Sie stehen vor und hinter:
  - Zitaten und wörtliche Rede
  - Titeln und zitierten Überschriften
  - Hervorhebungen von Wort- und Textteilen
- ◆ Anführung innerhalb einer Anführung wird durch ‚...‘ bzw. >...< gekennzeichnet

- ◆ **Unterstreichungen stammen aus der Schreibmaschinenzeit!**
- ◆ Hervorhebungen werden durch *kursiv*, **fett** oder durch größere/andere Schriftart erreicht
- ◆ „normale“ Unterstreichungen zerschneiden Unterlängen, und wirken schwerfällig
- ◆ Unterstreichungen in Ausnahmefällen können durch gezeichnete Linien erreicht werden. Dabei können die Linien bei Unterlängen unterbrochen werden
- ◆ Unterstreichungen der Links können durch **Cascading Style Sheets (CSS)** vermieden werden (sind aber im WWW üblich)

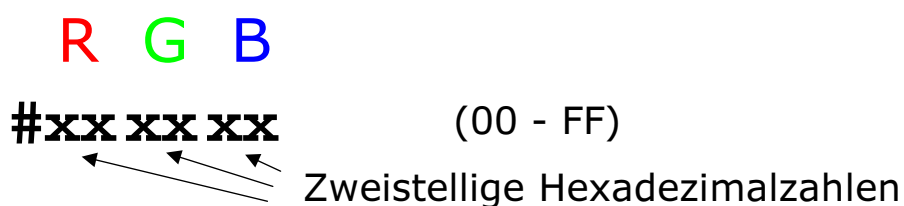
- ◆ RGB-Farbmodell (Rot-Grün-Blau; additiv)






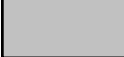
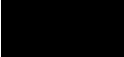



- ◆ CMY-Farbmodell (Cyan-Magenta-Yellow; subtraktiv)






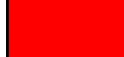




- ◆ Das CMY-Farbmodell ist für reflektierende Medien (z.B. Drucker) weit verbreitet
  - die spektralen Intensitäten werden aus dem weißen Licht entfernt (subtraktiv)
  - additive Mischung der Absorptionskomponenten
- ◆ Das RGB-Farbmodell ist für aktiv lichterzeugende Medien (z.B. Monitore) das meistverwendete Modell
  - die spektralen Intensitäten werden addiert (additiv)
  - In HTML wird **0 ... 1** nach **0 ... 255** abgebildet



- ◆ 16 vordefinierte normierte Farbnamen
  - es gibt viele weitere Farbnamen, die aber nicht von allen Browsern verstanden werden

Farbe	Hex-Wert	Farbname
	#00FFFF	aqua
	#808080	gray (grey)
	#000080	navy
	#C0C0C0	silver
	#000000	black
	#808000	olive
	#008000	green
	#008080	teal

Farbe	Hex-Wert	Farbname
	#0000FF	blue
	#800080	purple
	#00FF00	lime
	#FFFFFF	white
	#FF00FF	fuchsia
	#FF0000	red
	#800000	maroon
	#FFFF00	yellow

## Texte formatieren

Tag	Ende	Beschreibung
<h1>	</h1>	Überschrift Größe 1 (größte)
...	...	
<h6>	</h6>	Überschrift Größe 6 (kleinste)
<b>	</b>	fett (bold)
<i>	</i>	kursiv (italic)
<u>	</u>	unterstrichen (nicht verwenden!)
<sup>	</sup>	Text hoch gestellt
<sub>	</sub>	Text tief gestellt
<big>	</big>	Vergrößerung
<small>	</small>	Verkleinerung
<tt>	</tt>	Schreibmaschinenschrift

- ◆ Schriftgrößen können durch das Tag-Attribut "**size**" von **font** definiert werden
- ◆ `<font size= "7">`                      Ziemlich riesiger Text
- ◆ `<font size= "1">`                      Ziemlich winziger Text
- ◆ `<font size=+1>`                      Text etwas größer als normal
- ◆ `<font size=-2>`                      Text deutlich kleiner als normal

## Textblöcke formatieren

Tag	Ende	Beschreibung
<code>&lt;!--</code>	<code>--&gt;</code>	Kommentar
<code>&lt;br&gt;</code>		neue Zeile (break)
<code>&lt;p&gt;</code>	<code>&lt;/p&gt;</code>	Absatz (paragraph)
<code>&lt;hr&gt;</code>		horizontale Linie (horizontal ruler)

Attribut	Werte	Beschreibung
<b>size</b>	n	Größe (Standardgröße 2)
<b>width</b>	n, n%	Breite (in Pixel bzw. Prozent)
<b>height</b>	n, n%	Höhe (in Pixel bzw. Prozent)
<b>align</b>	LEFT, RIGHT, CENTER	Ausrichtung horizontal
<b>align</b>	TOP, MIDDLE, BOTTOM	Ausrichtung vertikal

- ◆ In HTML gültige Grafikformate sind **GIF**, **JPEG** und **PNG**
- ◆ Ladezeiten für Grafiken sind länger als für Text
- ◆ Verwendung von Thumbnails
  - Thumbnails sind verkleinerte Versionen der Originalbilder, auf denen das Bild erkennbar ist
  - sie werden sehr schnell geladen und können mit Links auf die entsprechenden Originalbilder versehen werden
  - es werden zwei Versionen des Bildes benötigt, den Thumbnail (kleine Datei) und das Bild in Originalgröße
- ◆ Alternativtext
  - Alternativ-Attribut alt="..." bzw. Alternativtext bei <object>
  - es kann beliebiger Text angegeben werden
  - sinnvoll ist der Name des Bildes (wenn aussagekräftig)

- ◆ **GIF (Graphics Interchange Format)** von CompuServe
  - maximal 256 Farben und 16.000 x 16.000 Pixel)
  - verlustfreie Kompression (LZW-Kompression von Unisys)
  - in Version GIF89a sind Animationen möglich (animierte GIFs)
  - Hintergrundfarbe kann transparent gemacht werden
  - geeignet für Computerzeichnungen, Icons, Diagramme, Logos, Screenshots, ...
  - allgemein Bilder mit großen einfarbigen Flächen
  - kleine Dateien mit kurzen Übertragungszeiten
  - Dateien lassen sich mit „Interlaced-Technik“ abspeichern, dadurch werden sie beim Ladevorgang schichtweise aufgebaut

- ◆ **JPEG (Joint Photographic Experts Group)**
  - oft auch mit Dateiendung (Extension) „JPG“
  - 16,7 Millionen Farben (für Fotos geeignet)
  - keine Transparenz oder Animation möglich
  - einstellbare Kompression, aber verlustbehaftet
  - verlustfreie Kompression ist möglich
  - Kompressions- und Dekompressionsalgorithmen liegen teilweise als Quelltext vor
  - bei verlustbehafteter Kompression wird versucht, für das Auge nicht sichtbare Unterschiede im Bild wegzulassen
  - dadurch bis Faktor zehn oder mehr reduzierte Dateigröße
  - Dateien lassen sich als „progressive JPEG“ abspeichern, dadurch werden sie beim Ladevorgang schichtweise aufgebaut

- ◆ **PNG (Portable Network Graphics)**
  - entwickelt wegen Lizenzproblemen mit GIF (LZW-Patent)
  - wird noch nicht von allen Browsern unterstützt
  - als Echtfarbdatei und auch mit reduzierten Farben wie GIF
  - einstellbare Farbauflösung von 15 bis 48 Bit pro Pixel
  - Grad der Transparenz einstellbar (Alpha-Kanal)
  - Animationen noch nicht möglich
  - besserer progressiver Aufbau der Bilder als bei GIF (mittels Adam7-Progression)

- ◆ Grafiken werden über den `<img>` Tag eingebunden. z.B.:  
``
  - Das Attribut „**alt**“ ist für die Anzeige eines alternativen Textes  
z.B.: ``
  - Weiterhin sollten auch Größenangaben zur Grafik gemacht werden, da sich sonst beim Aufbau der Seite der Inhalt ständig verschieben würde, z.B. ``
- ◆ Grafik als Beschreibung eines Links mit „border=0“, da sonst ein Link-Rahmen um die Grafik entsteht
- ◆ Hintergrundgrafik für eine HTML-Seite:  
z.B. `<body background="wolken.gif">`
  - Die Grafik wird dann gekachelt auf der Seite dargestellt

- ◆ Links können viele Ziele haben, so z.B.
  - Stelle innerhalb des Dokuments
  - andere HTML-Datei
  - WWW-Adresse
  - FTP-, Telnet-, ... Adresse
  - Datei im Internet (z.B. zum Download)
  - lokal abgelegte Datei
- ◆ Grundsätzlich: `<a href=[Ziel]> Text </a>`
- ◆ Links innerhalb des Dokuments
  - Zuerst muss ein Anker innerhalb des Dokuments definiert werden, zu dem gesprungen werden kann  
z.B. `<a name="Position2"> bla blub </a>`
  - Dann muss ein Anker mit Verweis auf das definierte Sprungziel + "#..." angegeben werden  
z.B. `<a href="#Position2"> Hip Hopp </a>`

- ◆ Links zu anderen lokalen Dateien sind IMMER relativ anzugeben z.B. **../grafik/xy.html**
- ◆ Beispiel : `<a href="datei2.html"> Datei2 </a>`  
oder : `<a href="../../index.htm"> Anfang </a>`
- ◆ Es ist auch möglich direkt einen definierten Anker in der Zieldatei anzusteuern:  
`<a href="angebot/preise.htm#cpu"> CPUs </a>`

- ◆ Links ins WWW funktionieren wie normale Links zu Dateien, nur wird hier die URL angegeben :  
`<a href="http://www.ba-dresden.de"> BA-Dresden </a>`
- ◆ Ebenso FTP: `<a href="ftp://ftp.uni-mannheim.de"> FTP-Server Uni-Mannheim </a>`
- ◆ Links auf eMail Adressen sind wie folgt zu formulieren :  
`<a href="mailto:billg@microsoft.com"> Billy the Gate</a>`
- ◆ Bei mailto: folgen keine "/" !



Tag	Ende	Beschreibung
<code>&lt;ol&gt;</code>	<code>&lt;/ol&gt;</code>	sortierte Liste (ordered list)
		können geschachtelt werden
<code>&lt;ul&gt;</code>	<code>&lt;/ul&gt;</code>	unsortierte Liste (unsorted list)
		kann mit <code>&lt;ol&gt;</code> kombiniert werden
<code>&lt;li&gt;</code>	<code>&lt;/li&gt;</code>	Listeneintrag

Attribut	Werte	Beschreibung
<code>type</code>	1	arabische Ziffern
	A, a	Buchstaben (groß, klein)
	I, i	römische Zahlen (groß, klein)
	square, circle, ...	Symbole zur Strukturierung
<code>start</code>		Anfangswert

Tag	Ende	Beschreibung
<code>&lt;table&gt;</code>	<code>&lt;/table&gt;</code>	Tabelle

Attribut	Werte	Beschreibung
<code>border</code>	n	Breite des Rahmens in Pixeln
<code>cellpadding</code>	n	Abstand Zelleninhalt zum Rand
<code>cellspacing</code>	n	Abstand der Zellen untereinander
<code>bgcolor</code>	Farbe	Hintergrundfarbe für Tabelle
<code>bordercolor</code>	Farbe	Farbe des Rahmens

Tag	Ende	Beschreibung
<code>&lt;caption&gt;</code>	<code>&lt;/caption&gt;</code>	Tabellenüberschrift

Tag	Ende	Beschreibung
<code>&lt;tr&gt;</code>	<code>&lt;/tr&gt;</code>	Zellenreihe (table row)
<code>&lt;td&gt;</code>	<code>&lt;/td&gt;</code>	einzelne Zelle in der Reihe
<code>&lt;th&gt;</code>	<code>&lt;/th&gt;</code>	Inhalt in Zelle fett und zentriert

Attribut	Werte	Beschreibung
<code>colspan</code>	n	Verbinden von n Zellen rechts
<code>rowspan</code>	n	Verbinden von n Zellen darunter
<code>width</code>	n bzw. n%	Breite in Pixeln bzw. Prozent
<code>height</code>	n bzw. n%	Höhe in Pixeln bzw. Prozent
<code>align</code>	left, right, center	Ausrichtung des Zelleninhalts
<code>valign</code>	top, middle, bottom	Ausrichtung vertikal

- ◆ Vorteile
  - verbessertes Seitenlayout (Navigationsleiste, Logo, ...)
  - muss nicht immer neu geladen werden
  - Sparen von Übertragungszeiten
  - unabhängiges Ändern der einzelnen Rahmen
- ◆ Nachteile
  - längere Dauer bei erstem Bildaufbau
  - nur bei hoher Bildschirmauflösung sinnvoll
  - Aufbau wird komplexer
- ◆ Eine HTML-Seite enthält Framedefinition
  - Framesets können geschachtelt werden
- ◆ Jeder Rahmen enthält eigene HTML-Seite

Tag	Ende	Beschreibung
<b>&lt;frameset&gt;</b>	<b>&lt;/frameset&gt;</b>	Definition eines Frameset
...	...	anstatt <body> ... </body>
<b>&lt;noframes&gt;</b>	<b>&lt;/noframes&gt;</b>	Alternative für Browser ohne
		Frames (innerhalb von Framesets)

Attribut	Werte	Beschreibung
<b>frameborder</b>	n	in Pixel ("0": ohne Rahmen)
<b>&lt;cols&gt;</b>	n, n, ...	Frames nebeneinander in Pixeln
	n%, n%, ...	Frames nebeneinander in Prozent
<b>&lt;rows&gt;</b>	n, n	Frames übereinander in Pixeln
	n%, n%, ...	Frames übereinander in Prozent

Tag	Ende	Beschreibung
<b>&lt;frame&gt;</b>		Inhalt eines Rahmen

Attribut	Werte	Beschreibung
<b>src</b>	Datei	Name der HTML-Datei
<b>name</b>	Text	Name eines Frames
<b>scrolling</b>	no, yes	Rollbalken (nicht) erlauben
	left, right	Rollbalken links oder rechts
	top, bottom	Rollbalken oben oder unten
<b>border</b>	n	Breite des globalen Rahmens

- ◆ Das Grundgerüst einer Frame-Datei ist zu einer HTML-Datei leicht verändert :

```
<html>
  <head>
    <title> Titel der Seite </title>
  </head>
  <frameset>
    Definition der Frames
  </frameset>
  <body>
    Text für Browser ohne Frame-Support
  </body>
</html>
```

- ◆ Aufbau eines Framesets wie beschrieben :

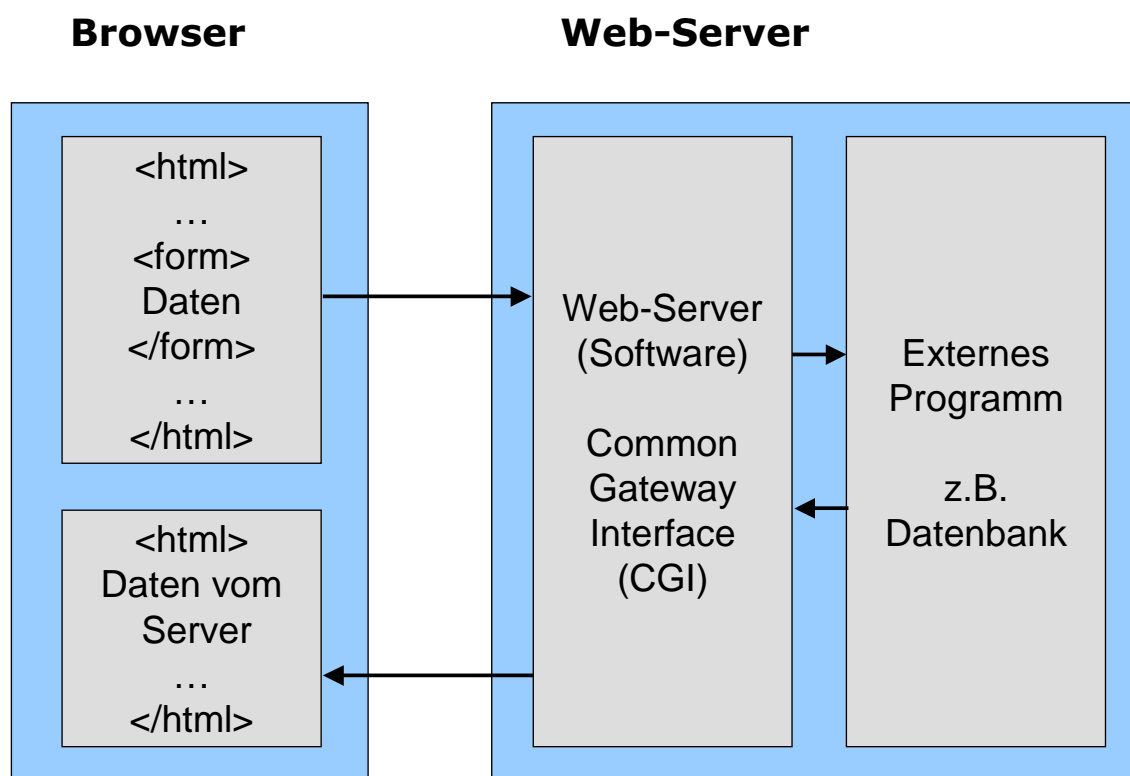
```
<frameset rows="10%, 90%">
  <frame src="banner.htm" name="Banner">
  <frameset cols="20%, 80%">
    <frame src="navi.htm" name="Navigate">
    <frame src="main.htm" name="Main">
  </frameset>
</frameset>
```

- ◆ Um generell ein Ziel für Links zu definieren kann z.B. in navi.htm ein Base Target angegeben werden :  
**<base target="Main">**  
Damit wird das Ziel der Links von navi.htm im Frame "Main" angezeigt

- ◆ Style Sheets sind Formatvorlagen
- ◆ Cascaded Style Sheets sind eine definierte Art von Style Sheets
- ◆ Einheitliches Aussehen aller Seiten einer Web-Seite lässt sich einfacher realisieren
- ◆ Externe CSS werden in einer eigenen Datei definiert
- ◆ Interne CSS sind im Kopfteil der HTML-Seite (**<head> ... </head>**) enthalten
- ◆ CSS können auch auf einzelne Tags angewendet werden
- ◆ Bilden von Klassen für Formatvorlagen ist möglich
- ◆ Es lassen sich Formatvorlagen für verschachtelte Tags definieren (Kaskadierung)

```
<html>
  <head>
    <title> Titel der Seite </title>
    <style type="text/css">
      <!--
        h1 { font-family: Arial; font-size:24pt;
          color:red }
        p { font-family: Arial; font-size:20pt;
          color:green }
      -->
    </style>
  </head>
  <body>
    ...
  </body>
</html>
```

- ◆ Kommunikation zwischen Web-Seite und Web-Server erfolgt über CGI
- ◆ CGI ist eine bei allen Web-Servern vorhandene Schnittstelle zum Aufruf externer Programme
- ◆ Diese externen Programme liefern meist HTML-Code zurück
- ◆ CGI-Programme werden meist mit „Perl“ (einer Interpretersprache auf UNIX-Systemen) programmiert, können aber auch in einer beliebigen anderen Sprache programmiert sein (z.B. „C“)
- ◆ CGI-Programme erhalten Daten von der Web-Seite aus den Eingabefeldern, usw. eines Formulars (**<form> ... </form>**)



- ◆ Formulare werden innerhalb eines „Containers“ mittels **<form> ... </form>** gruppiert
- ◆ Eingabefelder außerhalb von Containern werden nicht erkannt
- ◆ Durch das Attribut „**action**“ wird der URL (Uniform Resource Locator) angegeben, der die Daten der Eingabefelder entgegen nimmt. Dieser URL zeigt auf ein Programm (CGI-Programm) oder E-Mail-Adresse
- ◆ Für die Übertragung der Daten gibt es zwei Methoden: „**get**“ und „**post**“
  - bei „**get**“ sind die Daten Bestandteil der URL, dessen Länge begrenzt ist!
  - bei „**post**“ werden die Daten getrennt versendet. Sie können auch verschlüsselt werden

- ◆ Beispiel: **<form action=" ... " method="post">**
- ◆ Durch den Attribut-Wert „submit“ des Attributs „type“ werden die Daten eines Formulars abgesendet z.B.: **<input type="submit" value="Senden">**
- ◆ Sie werden in der Reihenfolge wie sie im Formular stehen als Wertepaare „**name=value**“ mit „&“ als Trennzeichen zum Web-Server gesendet

```
</html>
```

```
...
```

```
<form action="url" method="get/post">
```

```
Text:
```

```
<input type="..." Name="..." value="..." ...>
```

```
<br>
```

```
...
```

```
<input type="submit" Name="Senden" ...>
```

```
</form>
```

```
...
```

```
</html>
```

Tag	Ende	Beschreibung
<code>&lt;form&gt;</code>	<code>&lt;/form&gt;</code>	Definition eines Formulars
<code>&lt;input&gt;</code>		Eingabe-Element
<code>&lt;select&gt;</code>	<code>&lt;/select&gt;</code>	Auswahlbox
<code>&lt;textarea&gt;</code>	<code>&lt;/textarea&gt;</code>	Auswahlbox

Attribut	Werte	Beschreibung
<code>type</code>	text	Text-Eingabefeld
<code>für</code>	reset	Schaltfläche zum Löschen
<code>Eingabe-</code>	submit	Schaltfläche zum Senden
<code>Elemente</code>	password	Passwort-Eingabe
	radio	Optionsfeld
	hidden	verstecktes Feld

Attribut	Werte	Beschreibung
<b>action</b>	url	Attribut für „<form>“
<b>method</b>	post/get	Sendart für „<form>“
<b>name</b>	Name	Name des Eingabe-Elementes
<b>value</b>	Wert	Wert des Eingabe-Elementes

## XML - eXtensible Markup Language (1/2)

- ◆ Teilmenge von SGML (Standard Generalized Markup Language)
- ◆ Mit XML können eigene Markup-Sprachen oder auch eigene Erweiterungen von HTML bzw. XHTML mit eigenen Tags für bestimmte Elemente definiert werden.
- ◆ Mit XML definierten Markup-Sprachen werden als XML-Anwendungen bezeichnet.
- ◆ Die Syntax, Struktur und Bedeutung der Tags wird für jede XML-Anwendung mit einer DTD (Document Type Definition) oder einem Schema definiert.
- ◆ Die Verarbeitung kann mit XML-Parsern mit DOM (Document Object Model) oder SAX (Simple API for XML) erfolgen.
- ◆ Wie die Elemente sichtbar dargestellt werden sollen, kann mit XSL (Extensible Style Language) oder CSS (Cascading Style Sheets) definiert werden.
- ◆ XML wird oft als Austauschformat verwendet

- ◆ Beispiele für XML-Anwendungen
  - **DocBook** (für gedruckte Texte und Bücher),
  - **WML** (für Online-Informationen auf kleinen Displays wie z.B. Handys),
  - **XHTML** (für Online-Informationen auf großen Displays wie z.B. PCs und Fernsehschirmen),
  - **MathML** (für mathematische Formeln),
  - **CML** (für chemische Formeln),
  - **SVG** (für Vektor-Graphiken),
  - u.v.a.

# Webanwendungen



## JavaScript

**Autor:**

**Dipl.-Math.  
E. Engelhardt**

**Stand:**

**04. Nov. 2007**

## für Syntaxbeschreibung verwendete Zeichen

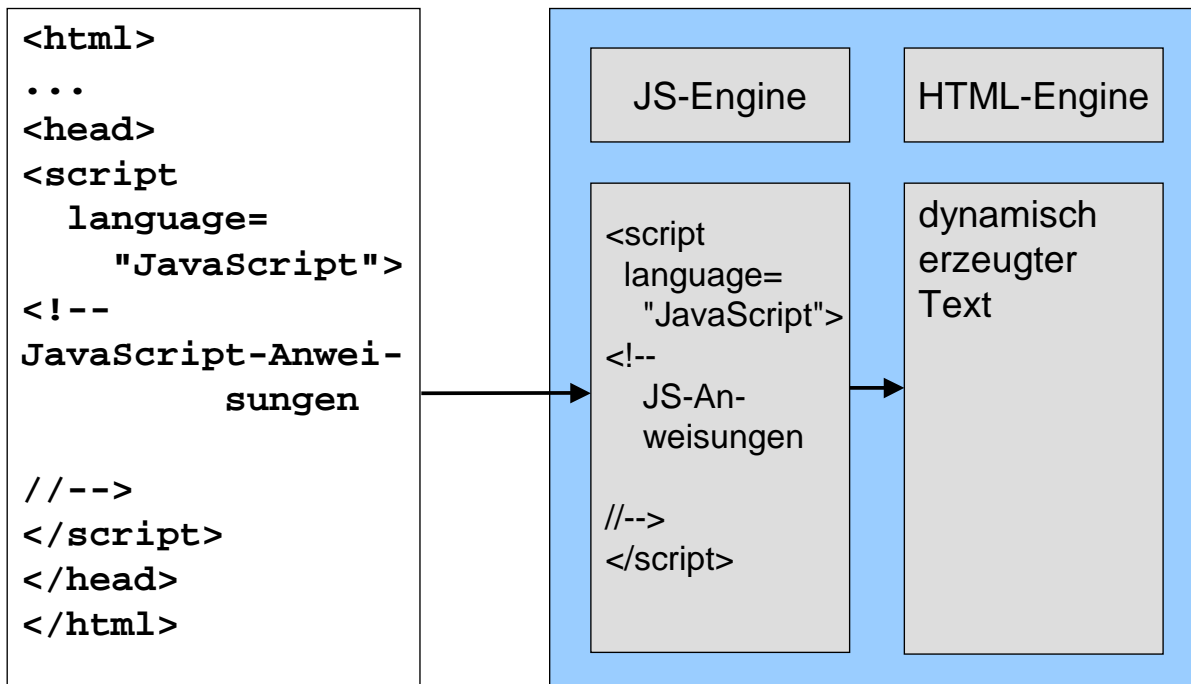
- ◆ [ ... ] was zwischen den eckigen Klammern steht kann, muss aber nicht stehen.
- ◆ ... | ... es gilt eine der Alternativen (die vor oder die nach dem Senkrechtstrich)
- ◆ < ... > was zwischen den spitzen Klammern steht ist ein Platzhalter für einen Bezeichner
- ◆ , Komma ist Trenner bei Aufzählungen
- ◆ { } . . . beliebige Anzahl des in Klammern stehenden Syntax-Elementes

## JavaScript-Charakterisierung

- ◆ JavaScript ist nicht Java!
- ◆ JavaScript wird erst zur Laufzeit in aus dem Quellcode in Maschinencode umgewandelt
- ◆ JavaScript ist objektbasierend (Java objektorientiert)
  - JavaScript kennt und verwendet eingebaute und selbstdefinierte Objekte
  - Klassen und Vererbung sind nicht in JavaScript enthalten
- ◆ JavaScript ist schwach typisiert, d.h. es wird automatisch zwischen den verschiedenen Datentypen konvertiert
- ◆ Dynamische Bindung: Die Objekte werden erst zur Laufzeit überprüft und aufgelöst

## Server

## Browser



## Grundgerüst eines JavaScriptes

```

<html>
<head>
<title> Erstes JavaScript </title>
<script language="JavaScript">
<!--
  alert("JavaScript-Dialogfenster");
//-->
</script>
<noscript>
  Ihr Browser versteht kein JavaScript!
</noscript>
</head>
</html>

```

- ◆ **Ganzzahlen**
  - Dezimalzahlen            -12, 34, 87654
  - Oktalzahlen                00, 012, 077
  - Hexadezimalzahlen        0x0, 0xFF, 0x0D
- ◆ **Gleitkommazahlen**
  - 1.72, 0.99e-4
- ◆ **Zeichenketten (string)**
  - "Zeichenkette", `Text in "Zeichenkette"`, "`Text` umgekehrt"
- ◆ **Wahrheitswerte (boolean)**
  - true oder false
- ◆ **null-Wert (Null-Zeiger)**
- ◆ **Objekte**
- ◆ **Sonderzeichen (\b,\n,\t,\f,\r, ...)**

- ◆ Beginnen mit Buchstaben oder Unterstrich (underscore)
- ◆ Sie werden mit Schlüsselwort „var“ definiert
- ◆ Es kann ein Anfangswert festgelegt werden
- ◆ Variablen erhalten den Typ des zugewiesenen Wertes

- ◆ Für numerische Datentypen (auch Zeichen)
- ◆ „+“ - Addition, „-“ - Subtraktion, „\*“ - Multiplikation
- ◆ „/“ - Division (bei ganzen Zahlen wird abgerundet)
  
- ◆ zusätzlich bei ganzzahligen Datentypen
- ◆ „%“ - Restdivision (modulo)
- ◆ „++“ - Inkrement (Bsp.: i++, ++i)
- ◆ „--“ - Dekrement (Bsp.: j--, --j)

**Bem.:** `i++` und `++i` ist nicht dasselbe!

## Zuweisungen, Bedingungsoperator

- ◆ „=“ - Zuweisung: der links vom Gleichheitszeichen stehenden Variable wird der rechts vom Gleichheitszeichen stehende Ausdruck zugewiesen
- ◆ „+=“, „-=“, „\*=“, „/=“ und „%=“ verkürzte Schreibweise, wenn sich der Wert der Variablen aus dem alten Wert dieser Variablen +, -, \*, /, und % einem Ausdruck ergibt (diese Schreibweise gilt auch für die Bitoperatoren)
  
- ◆ Bedingungsoperator (bedingter Ausdruck) ist der einzige dreistellige Operator

### Syntax:

**Vergleichsausdruck ? Ausdruck\_1 : Ausdruck\_2**

## ◆ Vergleichsoperatoren

- < kleiner,
- > größer,
- == gleich,
- <= kleiner gleich
- >= größer gleich
- != ungleich

## ◆ logische Operatoren

- && logisches „und“ (and)
- || logisches „oder“ (or)
- ! logisches „nicht“ (not)

## ◆ Ausdruck

- beliebiger arithmetischer oder logischer Ausdruck
- Zuweisung oder Funktionsaufruf (jeweils ohne Semikolon als Abschluss)

## ◆ Anweisung

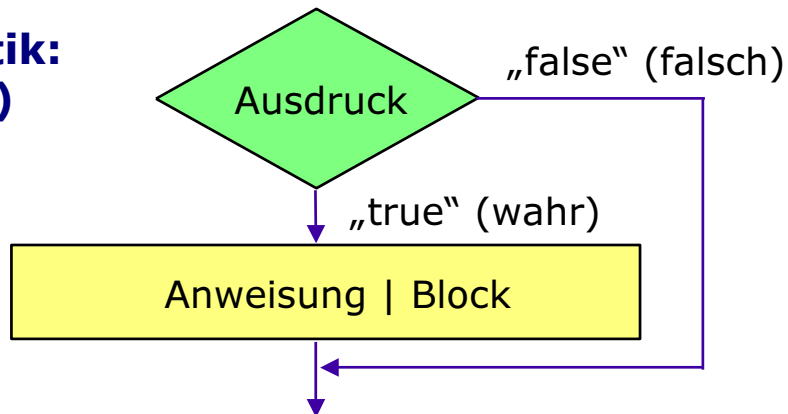
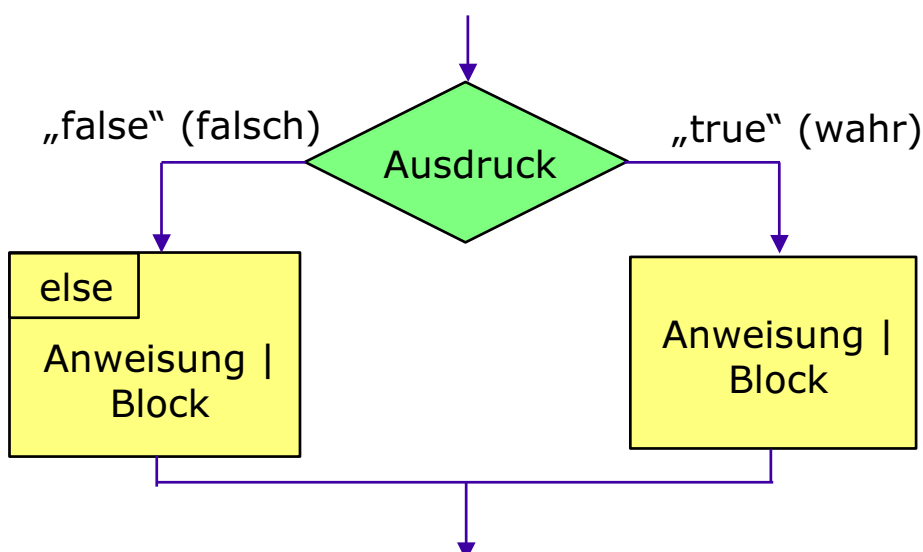
- Zuweisung oder Funktionsaufruf mit Semikolon als Abschluss
- Kontrollstruktur
- Ein Semikolon allein ist die leere Anweisung!

**Syntax der if - Verzweigung:**

```

if (logischer Ausdruck)
  Anweisung | Block
[ else
  Anweisung | Block ]

```

**Semantik:  
(1. Fall)****Semantik der if-Verzweigung (2.Fall mit „else“):**

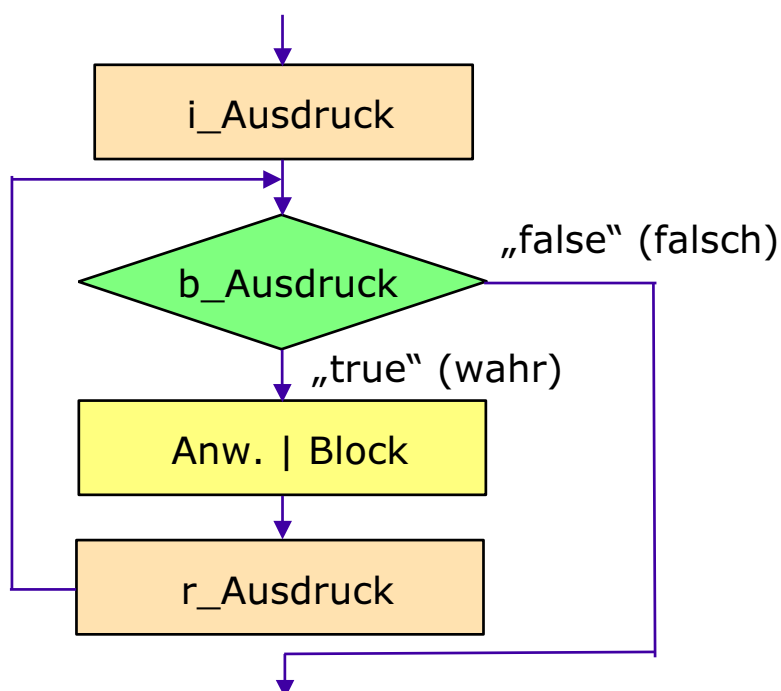
## Syntax der for - Schleife:

```
for ( [ i_Ausdruck ] ; b_Ausdruck ; [ r_Ausdruck ] )  
    Anweisung | Block
```

i\_Ausdruck:       Initialisierung (beliebiger Ausdruck)  
b\_Ausdruck:       Bedingung (logischer Ausdruck)  
r\_Ausdruck:       Reinitialisierung (beliebiger Ausdruck)

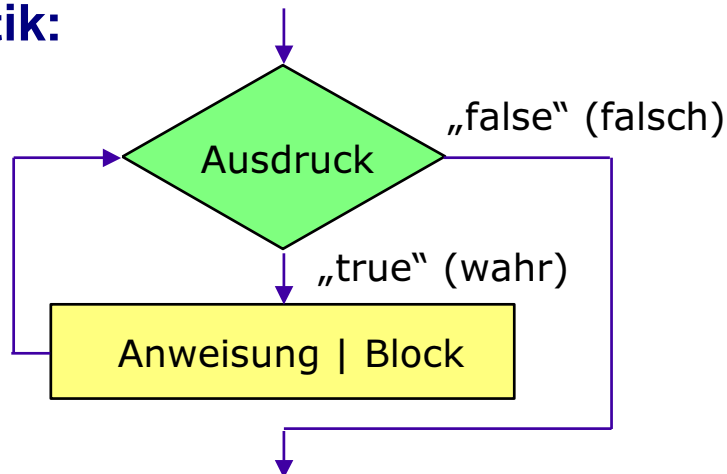
**Beispiel:**    for ( ; i < 10 ; ) ;    ist zulässig!

## Semantik der for-Schleife:

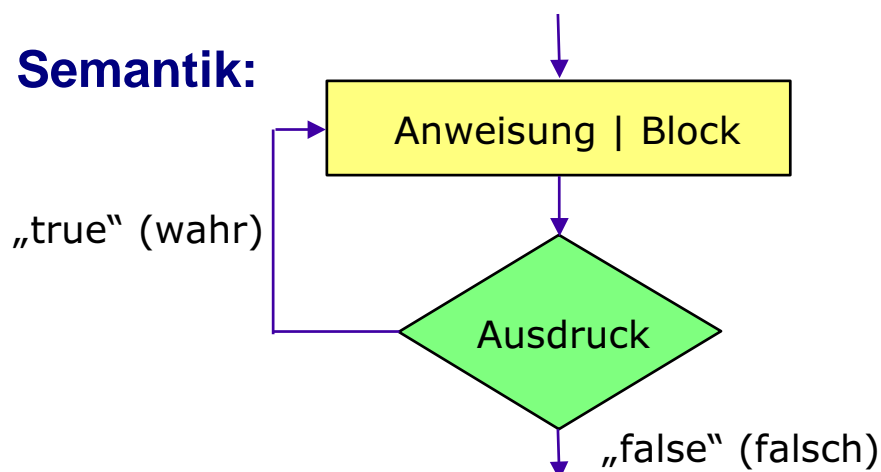


**Syntax der kopfgesteuerten while-Schleife:**

```
while (logischer Ausdruck)
  Anweisung | Block
```

**Semantik:****Syntax der fußgesteuerten while-Schleife:**

```
do
  Anweisung | Block
while (logischer Ausdruck);
```

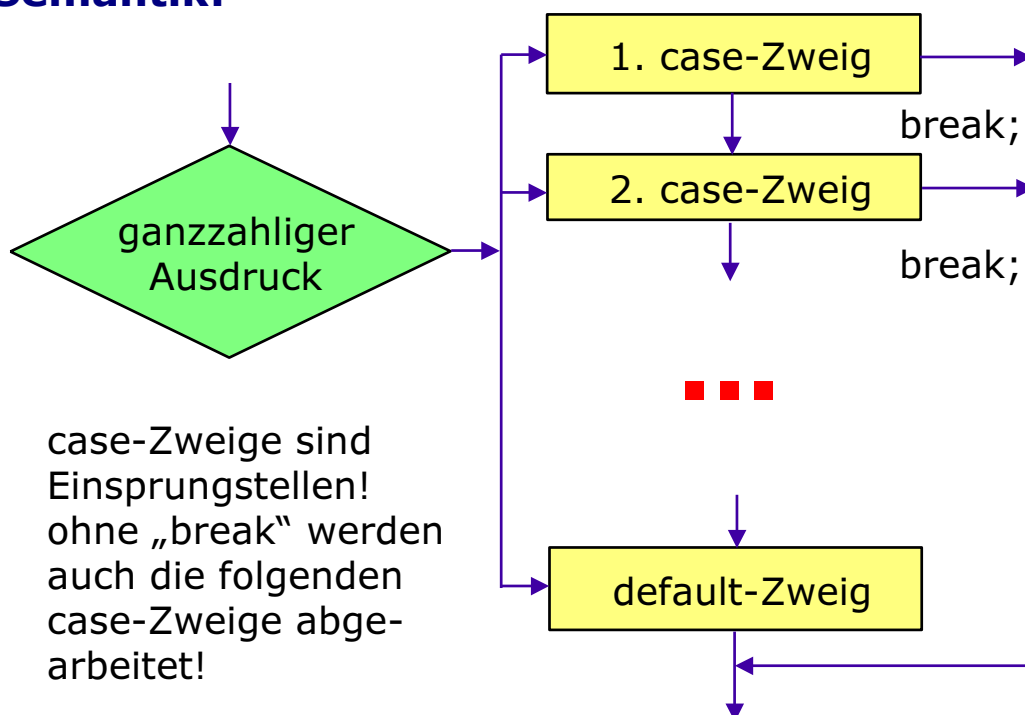
**Semantik:**

### Syntax der switch-Struktur:

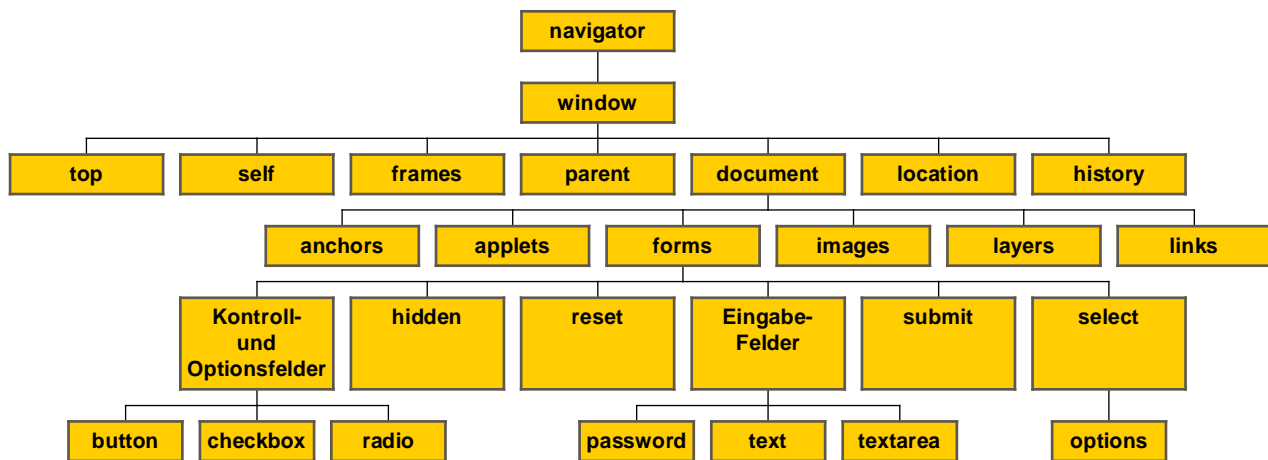
```

switch (ganzzahliger Ausdruck)
{
    case ganzz. Konstante: [Anweisung[en]]
    {
        [Block]
        [break;]
    }
    . . .
    default: [Anweisung[en]]
            [Block]
}
    
```

### Semantik:



## JavaScript-Objektmodell



## Ereignisse in JavaScript (1)

Ereignis	Beschreibung
onAbort (*)	Abbruch des Ladens einer Web-Seite
onBlur	Verlassen eines Elementes
onChange	Änderung von Angaben
onClick	Klick auf ein Element
onDbClick	Doppelklick (wird kaum verwendet)
onError (*)	Fehlerfall
onFocus	Element ist aktiv (hat Focus)
onKeyDown	Taste ist gedrückt
onKeyPress	Taste wurde gedrückt (Betätigen einer Taste)
onKeyUp	Taste wurde losgelassen
onLoad	Laden einer Web-Seite
onUnload	Verlassen einer Web-Seite

Ereignis	Beschreibung
onMouseDown	Betätigen der Maustaste
onMouseMove	Bewegen der Maus über einem Element
onMouseOut	Verlassen eines Elementes mit der Maus
onMouseOver	Maus wird auf ein Element bewegt
onMouseUp	Loslassen der Maustaste
onReset	Zurück setzen eines Formulars
onSelect	Selektieren von Text in Eingabefeldern
onSubmit	Absenden von Formulardaten

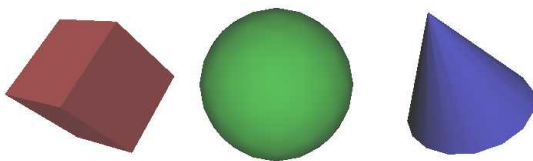
Mit „\*“ gekennzeichnete Ereignisse sind nicht Bestandteil der HTML 4.0 Spezifikation!

## JavaScript-Ereignisse

- ◆ JavaScript ist nicht Java!
- ◆ JavaScript wird erst zur Laufzeit in aus dem Quellcode in Maschinencode umgewandelt
- ◆ JavaScript ist objektbasierend (Java objektorientiert)
  - JavaScript kennt und verwendet eingebaute und selbstdefinierte Objekte
  - Klassen und Vererbung sind nicht in JavaScript enthalten
- ◆ JavaScript ist schwach typisiert, d.h. es wird automatisch zwischen den verschiedenen Datentypen konvertiert
- ◆ Dynamische Bindung: Die Objekte werden erst zur Laufzeit überprüft und aufgelöst



## Virtual Reality Modeling Language



**Autor:** Dipl.-Math.  
E. Engelhardt

**Stand:** 04. Nov. 2007

## VRML-Geschichte



- ◆ 1994 entwickelten Marc Pesce und Toni Parisi VRML (Virtual Reality Markup Language)
- ◆ angelehnt an Standard der Firma Silicon Graphics Inc. (Open Inventor-Format)
- ◆ Verabschiedung von VRML 1.0 im April 1995
- ◆ Dezember 1995 VRML 1.1 durch VRML Architecture Group (VAG)
  - Erweiterung durch Interaktion und Prototypen
  - wurde nie veröffentlicht, Neuerungen gingen in VRML 2.0 auf
- ◆ VRML 2.0 (VAG 4.8.1996)
  - basiert auf Vorschlag von SGI („Moving Worlds“)
  - *ISO/IEC 14772*

- ◆ VRML97 (Dezember 1997)
  - ISO/IEC 14772-1:1997
  - ersetzt VRML 2.0 mit unwesentliche Änderungen
- ◆ VRMLC (VRML-Konsortium), hervorgegangen aus VAG, wird durch Web3D-Konsortium abgelöst
- ◆ VRML97 soll in X3D (Extensible 3D) enthalten sein

- ◆ Keine Programmiersprache, sondern Scriptsprache zur Beschreibung von virtuellen 3D-Welten
- ◆ Steuerung zeitlicher Abläufe ist möglich (Interaktion und Animation)
- ◆ Einfache Verbindungen von Dokumenten weltweit durch „Hyperlinks“
- ◆ Multimedial durch die Einbindung von Grafiken, Sound, Videos, ...
- ◆ problemloser Transport über Internet und Darstellung im Browser (VRML-PlugIn)

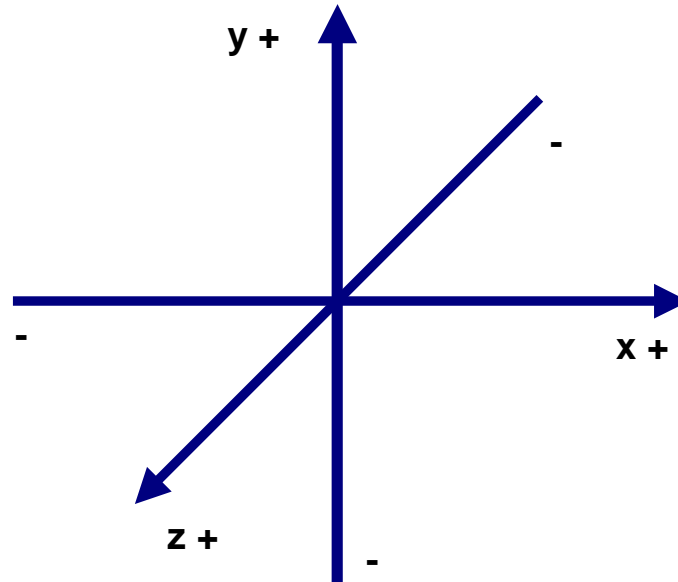
- ◆ VRML-Dokumente sind reine ASCII-Textdateien und damit mit jedem reinen ASCII-Editor bearbeitbar
- ◆ Konvertierung nach Java-3D und umgekehrt möglich
- ◆ schneller in der Darstellung als Java-3D
- ◆ flexibler und komfortabler als Flash-Animationen
- ◆ Programmier-Interface
- ◆ Tools zur Erstellung von VRML-Welten (z.B. Cosmo-World, Spazz3D, ...)
- ◆ CAD-Systeme enthalten Export-Funktion nach VRML

- ◆ Felder (Fields) bezeichnen Eigenschaften von Knoten, wie Größe, Farbe oder Position
- ◆ Felder besitzen einen Namen, einen Feldtyp und einen Wert (bzw. mehrere Werte) z.B.:
  - `field SFFloat radius 0.75634`
  - `field SFString gruss "Hallo Welt"`
- ◆ Unterscheidung nach
  - single-valued fields (`SF---`)
  - multi-valued fields (`MF---`)
- ◆ Komponenten von MF-Fields werden mit eckigen Klammern zusammengefasst und durch Komma getrennt
  - `field MFString adresse [ "01219", "Dresden" ]`

<b>SFBool</b>	Boolean, TRUE oder FALSE
<b>SFInt32</b>	Integer, z.B.: 0, 1, -3, 123
<b>SFFloat</b>	Gleitkomma, z.B.: .123, 3.1415, 2
<b>SFString</b>	String, z.B. „Hallo Welt“, „VRML“
<b>SFNode</b>	Knotenreferenz, z.B. NULL, Cone {}
<b>SFVec2f</b>	2D-Vektor,
<b>SFVec3f</b>	3D-Vektor,
<b>SFRotation</b>	Drehachse und Drehwinkel, z.B. 1 0 0 3.14 (Winkel im Bogenmaß)
<b>SFColor</b>	RGB-Wert(0,...,1), z.B. Grau: 0.5 0.5 0.5
<b>SFTime</b>	Anzahl der Sekunden seit dem 1. Januar 1970, 0 Uhr
<b>SFImage</b>	unkomprimiertes 2D-Pixelimage

Anchor	NavigationInfo	SpotLight
Background	NormalInterpolator	SphereSensor
Billboard	OrientationInterpolator	Switch
Collision	PlaneSensor	TimeSensor
ColorInterpolator	PointLight	TouchSensor
CoordinateInterpolator	PositionInterpolator	Transform
CylinderSensor	ProximitySensor	Viewpoint
DirectionalLight	ScalarInterpolator	VisibilitySensor
Fog Script	WorldInfo	
Group	Shape	
LOD	Sound	

- ◆ Rechtshändiges Koordinatensystem
- ◆ der Ursprung (0 0 0) liegt in der Mitte der Bildfläche



# Grundgerüst einer VRML-Datei

<code>#VRML V2.0 utf8</code>	Information für Browser
<code>Background {   . . . }</code>	Hintergrund
<code>Shape {   appearance Appearance</code>	Grundform Erscheinung
<code>  geometry . . . }</code>	Geometrie

```
Shape {                                # Grundform
    exposedField SFNode appearance NULL
    exposedField SFNode geometry  NULL
}
```

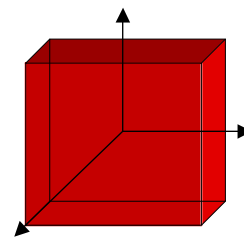
```
Appearance {                            # Erscheinung
    exposedField SFNode material        NULL
    exposedField SFNode texture         NULL
    exposedField SFNode textureTransform NULL
}
```

```
Background {
    eventIn      SFBool    set_bind
    exposedField MFFloat  groundAngle []
    exposedField MFColor   groundColor []
    exposedField MFString  backUrl  []
    exposedField MFString  bottomUrl []
    exposedField MFString  frontUrl  []
    exposedField MFString  leftUrl   []
    exposedField MFString  rightUrl  []
    exposedField MFString  topUrl    []
    exposedField MFFloat   skyAngle  []
    exposedField MFColor   skyColor  [ 0 0 0 ]
    eventOut      SFBool    isBound
}
```

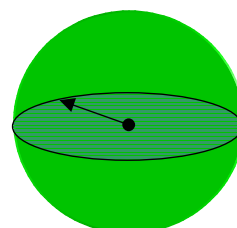
- ◆ Box
  - ◆ Cone
  - ◆ Cylinder
  - ◆ ElevationGrid
  - ◆ Extrusion
  - ◆ IndexedFaceSet
  - ◆ IndexedLineSet
  - ◆ PointSet
  - ◆ Sphere
  - ◆ Text
- Quader
  - Kegel
  - Zylinder
  - Netzgitter (Relief)
  - Sweeping
  - indizierte Flächen
  - indizierte Linien
  - indizierte Punkte
  - Kugel

## Box (Quader) und Sphere (Kugel)

```
Box {  
    field SFVec3f size 2 2 2  
}
```

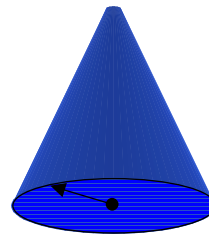


```
Sphere {  
    field SFFloat radius 1  
}
```



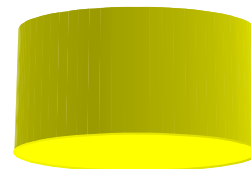
## Cone (Kegel)

```
Cone {  
    field SFFloat bottomRadius 1  
    field SFFloat height      2  
    field SFBool  side        TRUE  
    field SFBool  bottom      TRUE  
}
```



## Cylinder (Zylinder)

```
Cylinder {  
    field SFBool  bottom TRUE  
    field SFFloat height 2  
    field SFFloat radius 1  
    field SFBool  side TRUE  
    field SFBool  top TRUE  
}
```



```
ElevationGrid {
    eventIn      MFFloat set_height
    exposedField SFNode  color          NULL
    exposedField SFNode  normal         NULL
    exposedField SFNode  texCoord       NULL
    field        MFFloat height []
    field        SFBool  ccw            TRUE
    field        SFBool  colorPerVertex TRUE
    field        SFFloat creaseAngle    0
    field        SFBool  normalPerVertex TRUE
    field        SFBool  solid          TRUE
    field        SFInt32 xDimension     0
    field        SFFloat xSpacing       0.0
    field        SFInt32 zDimension     0
    field        SFFloat zSpacing       0.0
}
```

```
Extrusion {
    eventIn MFVec2f    set_crossSection
    eventIn MFRotation set_orientation
    eventIn MFVec2f    set_scale
    eventIn MFVec3f    set_spine
    field     SFBool    beginCap        TRUE
    field     SFBool    ccw             TRUE
    field     SFBool    convex          TRUE
    field     SFFloat   creaseAngle     0
    field     MFVec2f   crossSection
                        [ 1 1, 1 -1, -1 -1, -1 1, 1 1 ]
    field     SFBool    endCap          TRUE
    field     MFRotation orientation     0 0 1 0
    field     MFVec2f   scale           1 1
    field     SFBool    solid           TRUE
    field     MFVec3f   spine           [ 0 0 0, 0 1 0 ]
}
```

```
Text {
    exposedField MFString string []
    exposedField SFNode    fontStyle NULL
    exposedField MFFloat  length []
    exposedField SFFloat  maxExtent 0.0
}
```

```
FontStyle {
    field SFString family      "SERIF"
    field SFBool   horizontal  TRUE
    field SFString justify     "BEGIN"
    field SFString language    ""
    field SFBool   leftToRight TRUE
    field SFFloat  size        1.0
    field SFFloat  spacing     1.0
    field SFString style       ""
    field SFBool   topToBottom TRUE
}
```

## ImageTexture, PixelTexture

```
ImageTexture {
    exposedField MFString url []
    field SFBool  repeatsS TRUE
    field SFBool  repeatT TRUE
}
```

```
PixelTexture {
    exposedField SFImage image 0 0 0
    field SFBool  repeatsS TRUE
    field SFBool  repeatT TRUE
}
```

```

MovieTexture {
    exposedField SFBool    loop        FALSE
    exposedField SFFloat   speed       1
    exposedField SFTime    startTime  0
    exposedField SFTime    stopTime   0
    exposedField MFString  url        []
    field         SFBool    repeats    TRUE
    field         SFBool    repeatT    TRUE
    eventOut      SFFloat   duration_changed
    eventOut      SFBool    isActive
}

```

```

IndexedFaceSet {
    eventIn      MFInt32  set_colorIndex
    eventIn      MFInt32  set_coordIndex
    eventIn      MFInt32  set_normalIndex
    eventIn      MFInt32  set_texCoordIndex
    exposedField SFNode    color          NULL
    exposedField SFNode    coord          NULL
    exposedField SFNode    normal        NULL
    exposedField SFNode    texCoord      NULL
    field        SFBool    ccw           TRUE
    field        MFInt32   colorIndex    []
    field        SFBool    colorPerVertex TRUE
    field        SFBool    convex        TRUE
    field        MFInt32   coordIndex    []
    field        SFFloat   creaseAngle   0
    field        MFInt32   normalIndex   []
    field        SFBool    normalPerVertex TRUE
    field        SFBool    solid          TRUE
    field        MFInt32   texCoordIndex []
}

```

```
IndexedLineSet {
    eventIn      MFInt32 set_colorIndex
    eventIn      MFInt32 set_coordIndex
    exposedField SFNode  color  NULL
    exposedField SFNode  coord  NULL
    field MFInt32 colorIndex []
    field SFBool  colorPerVertex TRUE
    field MFInt32 coordIndex []
}
```

```
Material {
    exposedField SFFloat ambientIntensity 0.2
    exposedField SFColor diffuseColor 0.8 0.8 0.8
    exposedField SFColor emissiveColor 0 0 0
    exposedField SFFloat shininess 0.2
    exposedField SFColor specularColor 0 0 0
    exposedField SFFloat transparency 0
}
```

- [1] Günter Born: *HTML 4 Kompendium*, Markt&Technik Buch- und Software-Verlag GmbH, ISBN 3-8272-5354-3
- [2] Peter A. Henning: *Taschenbuch Multimedia*, Fachbuchverlag Leipzig im Carl Hanser Verlag, ISBN 3-446-21274-4
- [3] Thomas Kobert: *Das Einsteigerseminar HTML 4*, verlag moderne industrie Buch AG & Co. KG, Landsberg ISBN 3-8266-7150-3
- [4] Stefan Münz: *SelfHTML*, <http://www.teamone.de/selfhtml/>
- [5] *Internet HTML-Seiten*, HERDT-Verlag für Bildungsmedien GmbH, Nackenheim

- [1] VRML Architecture Group (VAG):  
*Version 2.0, Official Draft #3, ISO/IEC 14772*,
- [2] Rolf Däßler: *Das Einsteigerseminar VRML*, bhv-Verlags GmbH, ISBN 3-8287-1082-4