

# JavaScript, HTML und CSS

## Jump Start Training



- Konstantin Kletzander
  - Trainer @ ppedv
  - Webtechnologien

[KonstantinK@ppedv.de](mailto:KonstantinK@ppedv.de)

<http://blog.ppedv.de>

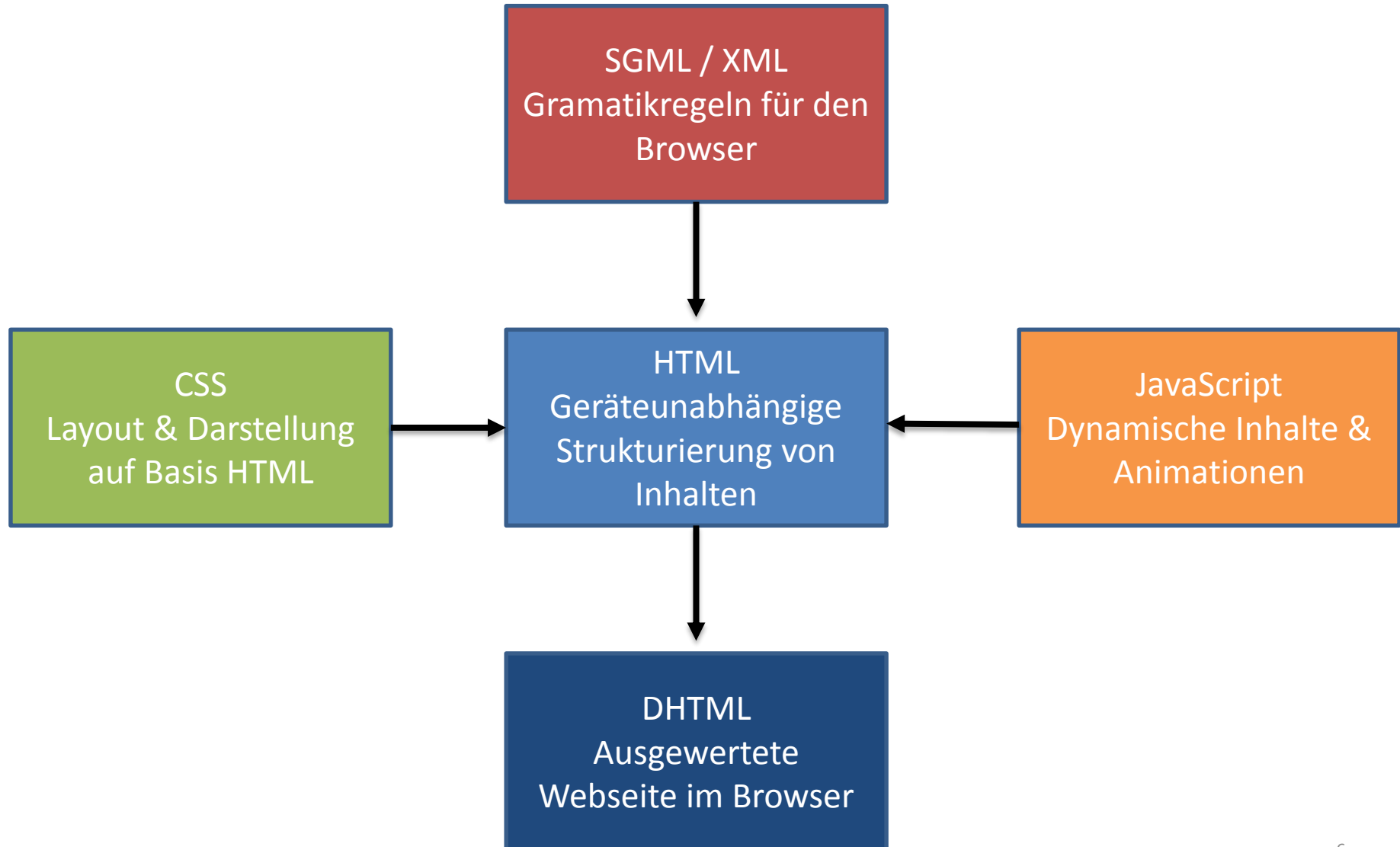


- Aktuelle Web Standards
  - HTML 4.01
  - CSS 2.1
- Überblick über Neuerungen
  - HTML 5
  - CSS 3

- Einführung in JavaScript
- Grundlegende Sprachelemente
- Kontrollstrukturen
- Funktionen
- Objekte
- Vordefinierte Objekte
- Objektmodell (DOM)
- Zugriff auf HTML Dokumente
- EventHandlerer

**„Geht's raus und  
spielt Fussball!“**





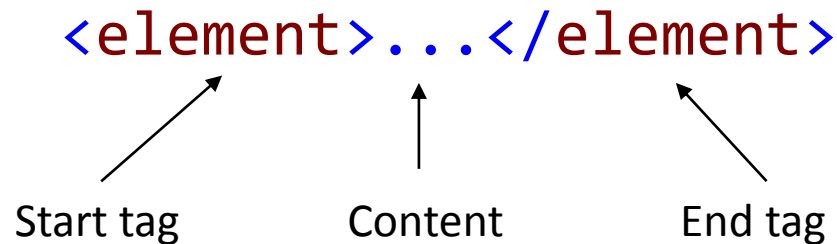
# HTML - Grundlagen

Wichtige Webseiten:

<http://de.selfhtml.org/>

- HTML Elemente, Tags

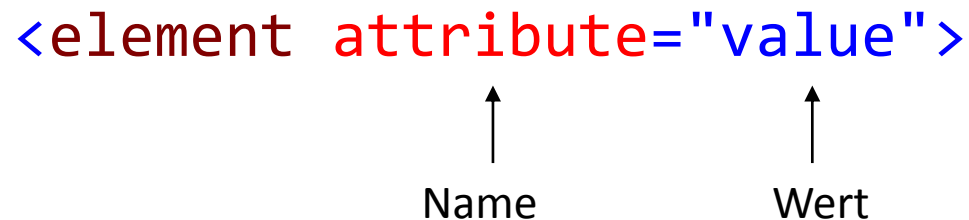
`<element>...</element>`



The diagram illustrates the structure of an HTML element. The text `<element>...</element>` is shown with three arrows pointing to its parts: `<element>` is labeled 'Start tag', `...` is labeled 'Content', and `</element>` is labeled 'End tag'.

- HTML Attribute

`<element attribute="value">`



The diagram illustrates the structure of an HTML attribute. The text `<element attribute="value">` is shown with two arrows pointing to its parts: `attribute` is labeled 'Name' and `"value"` is labeled 'Wert'.



- HTML-Elemente → markiert durch Tags
- Einleitendes & schließendes Tag

`<h1>HTML - Die Sprache im Web</h1>`

- Verschachtelung

✓ Grundregel: Elemente in umgekehrter Reihenfolge schließen, in der sie geöffnet wurden

– Falsch:

`<p><span class="important">HTML</p><p>CSS</span></p>`

– Richtig:

`<p><span class="important">Hallo HTML</span></p>`

`<div><h2>HTML ist cool</h2></div>`

- **Attribute in Tags**

- Attribute mit Wertzuweisung

```
<input type="button" />
```

- Attribute mit freier Wertzuweisung mit weiterer Konvention

```
<style type="text/css"></style>
```

- Attribute mit freier Wertzuweisung ohne weiterer Konvention

```
<p title="Hier steht ein Text"></p>
```

- Allein stehende Attribute

```
<input type="checkbox" checked />
```

```
<html>
```

```
<body>
```

```
<h1>Überschrift</h1>
```

```
<p>Das ist ein Absatz</p>
```

```
<p>Das ist ein anderer Absatz</p>
```

```
</body>
```

```
</html>
```

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>Hallo Muml;nchen</title>
</head>
<body>
    <!-- Sichtbarer Inhalt -->
</body>
</html>
```

- Zusätzliche ‚Einstellungen‘ für die Webseite im HEAD Bereich

```
<!-- Beschreibung -->
<meta name="description" content="Beschreibung bis 160 Zeichen" />
<!-- Favicon -->
<link rel="shortcut icon icon" href="favicon.ico" />

<!-- Mobile Homescreen Icons in px -->
<link rel="apple-touch-icon-precomposed" sizes="54x54" href="icon-54x54.png" />
<link rel="apple-touch-icon-precomposed" sizes="72x72" href="icon-72x72.png" />
<link rel="apple-touch-icon-precomposed" sizes="129x129" href="icon-129x129.png" />
<link rel="apple-touch-icon-precomposed" sizes="144x144" href="icon-144x144.png" />
<link rel="apple-touch-icon-precomposed" href="icon-54x54.png" />
<!-- Für mobiles Web - Viewport-->
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<!-- Styles -->
...
```

- HTML 5

```
<!DOCTYPE html>
```

- HTML 4.01

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01  
Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
"http://www.w3.org/TR/html4/strict.dtd">
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"  
"http://www.w3.org/TR/html4/frameset.dtd">
```

- Beispiele

- UTF-8 (HTML5)

- ```
<meta charset="utf-8" />
```

- ISO-8859-1 (HTML4)

- ```
<meta charset="iso-8859-1" />
```

- Sonderzeichen in encodierter Schreibweise

| HTML-Code | Zeichen |
|-----------|---------|
| &auml;    | ä       |
| &ouml;    | ö       |
| &Uuml     | Ü       |
| &szlig;   | ß       |

| HTML-Code | Zeichen     |
|-----------|-------------|
| &nbsp;    | Leerzeichen |
| &euro;    | €           |
| &lt;      | <           |
| &gt;      | >           |



- <TITLE> Titel der HTML-Datei
- <STYLE> Definition Stylesheet-Bereich
- <SCRIPT> Definition Script-Bereich
- <LINK> Beziehungen zu anderen Quellen
- <META> Meta-Angaben zum Inhalt

- `<a>`                      Hyperlinks, Anker für Hyperlinks
- `<b>`                        Fettdruck (bold)
- `<strong>`                Stilistische Hervorhebung
- `<i>`                         Kursiv (italic)
- `<em>`                     Emphatisch (gefühlsmäßig) betont
- `<br>`                     Zeilenumbruch erzwingen
- `<nobr>`                 Autom. Zeilenumbruch verhindern
- `<hr>`                    Horizontale Linie
- `<img>`                    Grafik-Referenz

- `<h1>...<h6>` Header / Überschriften
- `<p>` Textabsätze definieren
- `<dl>` Definitionsliste
- `<ol>` Nummerierte Liste
- `<ul>` Aufzählungsliste
- `<li>` Neuer Listeneintrag
- `<div>` Allgemeines Block-Element
- `<span>` Allgemeines Inline-Element

- `<table>`      Tabelle
- `<tr>`      Tabellenzeile
- `<td>`      Tabellenspalte
- `<th>`      Tabellenkopfspalte
- `<thead>`      Tabellenkopf
- `<tbody>`      Tabellenkörper
- `<tfoot>`      Tabellenfuß
- `<form>`      Formular
- `<input>`      Formular-Element
- `<textarea>`      Mehrzeiliges Eingabefeld

- `<center>` Zentrieren
- `<font>` Schriftformatierung
- `<big>` Größere Schrift als normal
- `<small>` Kleinere Schrift als normal
- `<frameset>` Definition eines Frameset
- `<frame>` Definition eines einzelnen Frames
- `<s>/<strike>` Durchgestrichener Text
- `<u>` Unterstrichener Text
- `<align>` Ausrichtung von Elementinhalten

- bgcolor Hintergrundfarbe
- background Hintergrundgrafik
- border Rahmendicke
- color Schriftfarbe
- hspace/vspace Abstand zu Elementen
- vlink Farbe für besuchte Links

- `<input type="text" />` Einzeiliges Eingabefeld
- `<input type="password" />` Passwort-Eingabefeld
- `<textarea></textarea>` Mehrzeiliges Eingabefeld
- `<input type="radio" />` Radio-Button
- `<input type="checkbox" />` Checkbox
- `<input type="hidden" />` Verstecktes Element
- `<input type="button" />` Klick-Button
- `<input type="submit" />` Absende-Button
- `<input type="reset" />` Reset-Button

```
<select>  
  <option>Eintrag 1</option>  
  <option>Eintrag 2</option>  
</select>
```

Auswahlliste



- Doctype: `<!DOCTYPE html>`
- Standard Zeichenkodierung: **UTF-8**
- **Semantische** Elemente: `<header>`, `<footer>`, `<article>`, `<section>`
- Bessere Unterstützung für **Multimedia**: `<audio>`, `<video>`
- Neue **Formular** Input-Typen wie `dates`, `email` und `url`
- **Grafische** Elemente: `<svg>` und `<canvas>`
- HTML5 befindet sich aktuell noch im Entwicklungsstadium



- Valides Markup stellt sicher, dass eine Webseite den aktuellen Richtlinien entspricht, ist aber kein Muss!
- <http://validator.w3.org/>
- Warum validieren?
  - Nicht valides Markup kann von Browsern unterschiedlich korrigiert werden
  - Kürzere Ladezeit
  - Barrierefreiheit
  - Besser für Suchmaschinenoptimierung
  - Richtige Darstellung auch in zukünftigen Browser Versionen

# CSS - Grundlagen

A magnifying glass with a black handle and a silver rim is positioned over a snippet of CSS code. The lens of the magnifying glass is focused on the 'body' selector and its properties, making them appear larger and clearer than the rest of the code. The code is written in a monospaced font with syntax highlighting: 'body {' in black, 'margin: 0;' in red, 'padding: 0;' in blue, '/\*resetting default size to 10px or 62.5% of the 16px default\*/' in green, 'font-size: 62.5%;' in red, 'font-family: Arial, Helvetica, sans-serif;' in blue, and 'color: #666;' in red. Below the 'body' block, the 'p' selector is visible with 'font-size: 1.2em;' in green, 'color: #333;' in red, 'margin: 0 0 10px;' in blue, and 'padding: 0;' in red. The magnifying glass is tilted slightly to the right, and its shadow is cast onto the code below it.

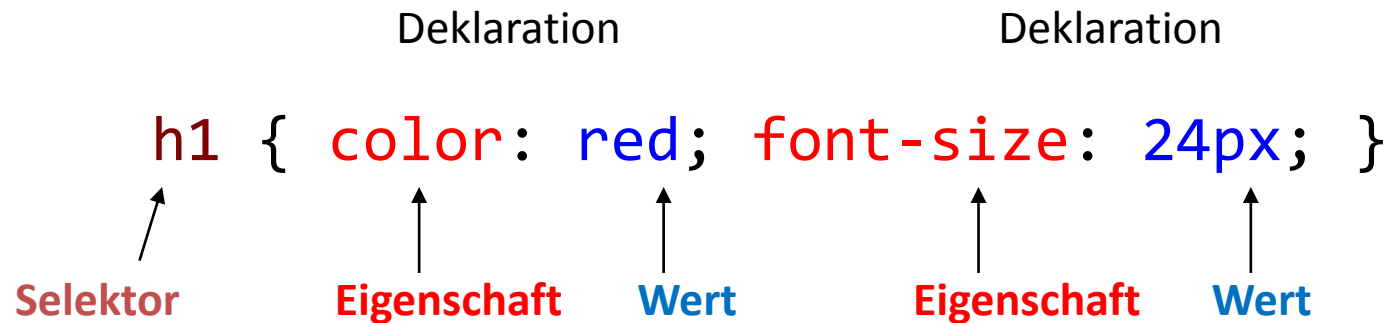
```
body {  
  margin: 0;  
  padding: 0;  
  /*resetting default size to 10px or 62.5% of the 16px default*/  
  font-size: 62.5%;  
  font-family: Arial, Helvetica, sans-serif;  
  color: #666;  
}  
p {  
  font-size: 1.2em; /*equivalent to 16px*/  
  color: #333;  
  margin: 0 0 10px;  
  padding: 0;  
}
```

## Deklarationsblock

Deklaration                      Deklaration

```
h1 { color: red; font-size: 24px; }
```

Selektor                      Eigenschaft   Wert                      Eigenschaft   Wert



The diagram illustrates the components of a CSS declaration block. The selector 'h1' is labeled 'Selektor'. The block contains two declarations: 'color: red;' and 'font-size: 24px;'. Each declaration is labeled 'Deklaration'. Within each declaration, the property name (color, font-size) is labeled 'Eigenschaft' and the value (red, 24px) is labeled 'Wert'.

- Universalselektor \*
- Typselektor p
- Klassenselektor .navigation
- Id-Selektor #navigation
- Attribut-Selektor a[href^="https"]
- Pseudoklassen-Selektor a:hover
- Nachfahren-Kombinator h1 i
- Kind-Kombinator div > p
- Benachbarte-Geschwister-Komb. div + p

- **Text & Farbe**

- color
- background-color
- font-family
- font-size
- font-weight
- text-decoration
- ...

- **Ausrichtung**

- text-align
- vertical-align
- height
- width
- ...

- Positionierung

- position

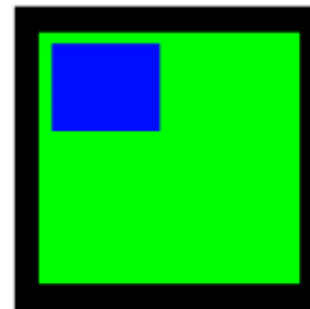
- static
    - absolute
    - relative
    - fixed

- display

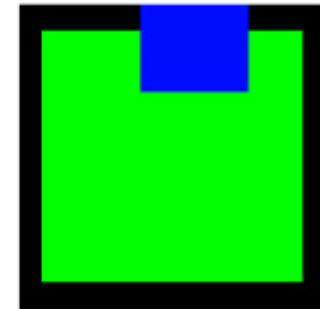
- inline
    - block
    - none

- margin

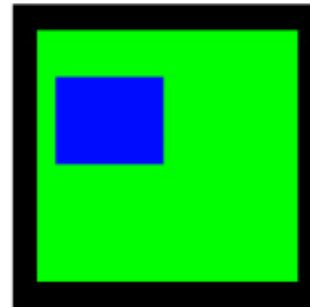
- padding



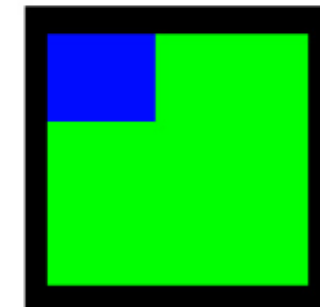
normal position



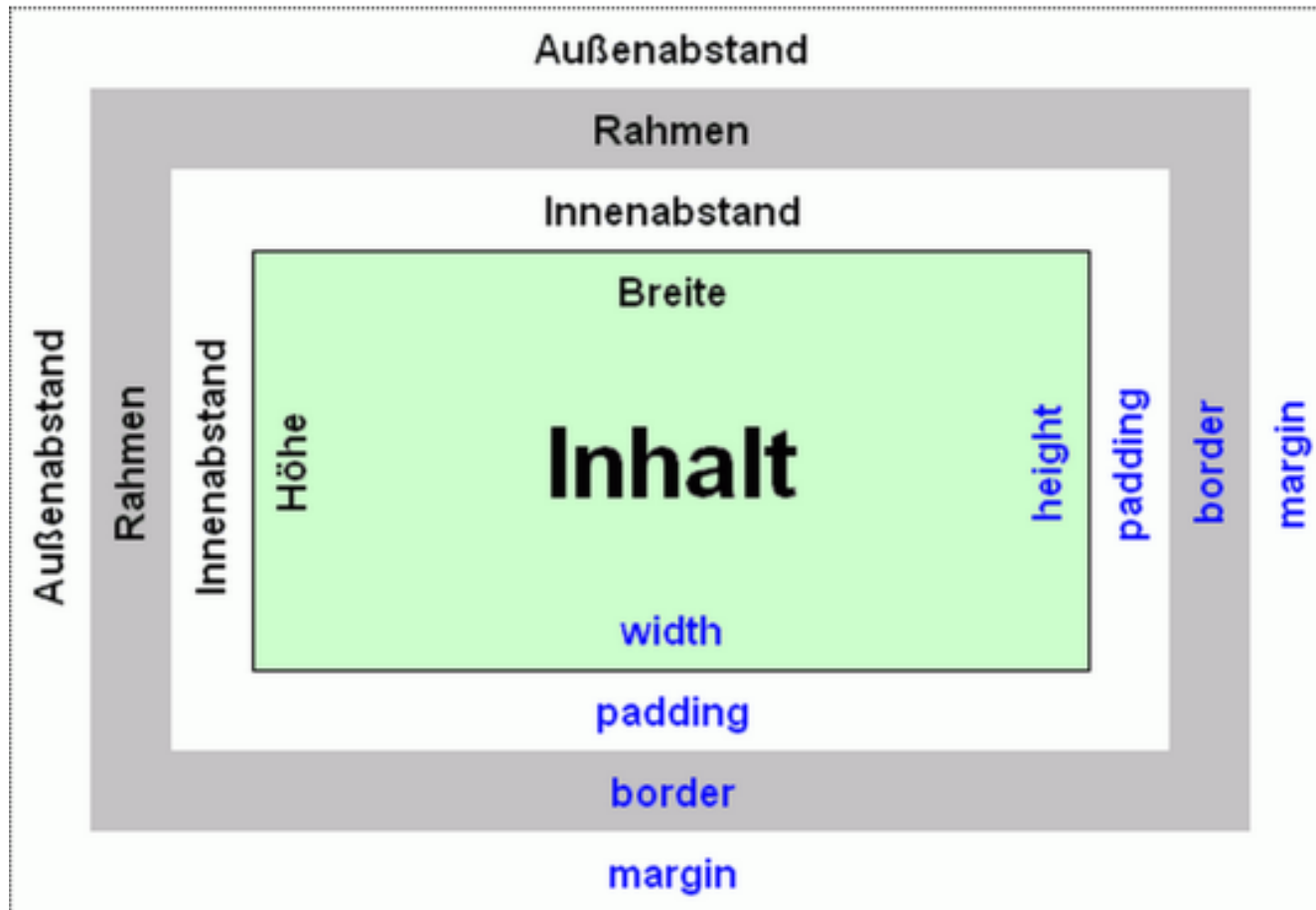
position: absolute;  
left: 200px;



position: relative;  
top: 20px;



position: fixed;  
left: 20px;  
top: 20px;





- Style-Bereich im HTML-Dokumentkopf

```
<head>
  ...
  <style type="text/css">
    h1 {
      font-weight: bold;
      font-size: 26px;
      color: #ff0000;
    }
  </style>
</head>
```

- Inline-Style in HTML-Elementen

```
<body>
...
  <h1 style="font-weight:bold;
    color:ff0000; font-size:26pt;">
    Cascading Style Sheets
  </h1>
...
</body>
```

- Style-Definitionen in separaten CSS-Dateien

- HTML-Dokument

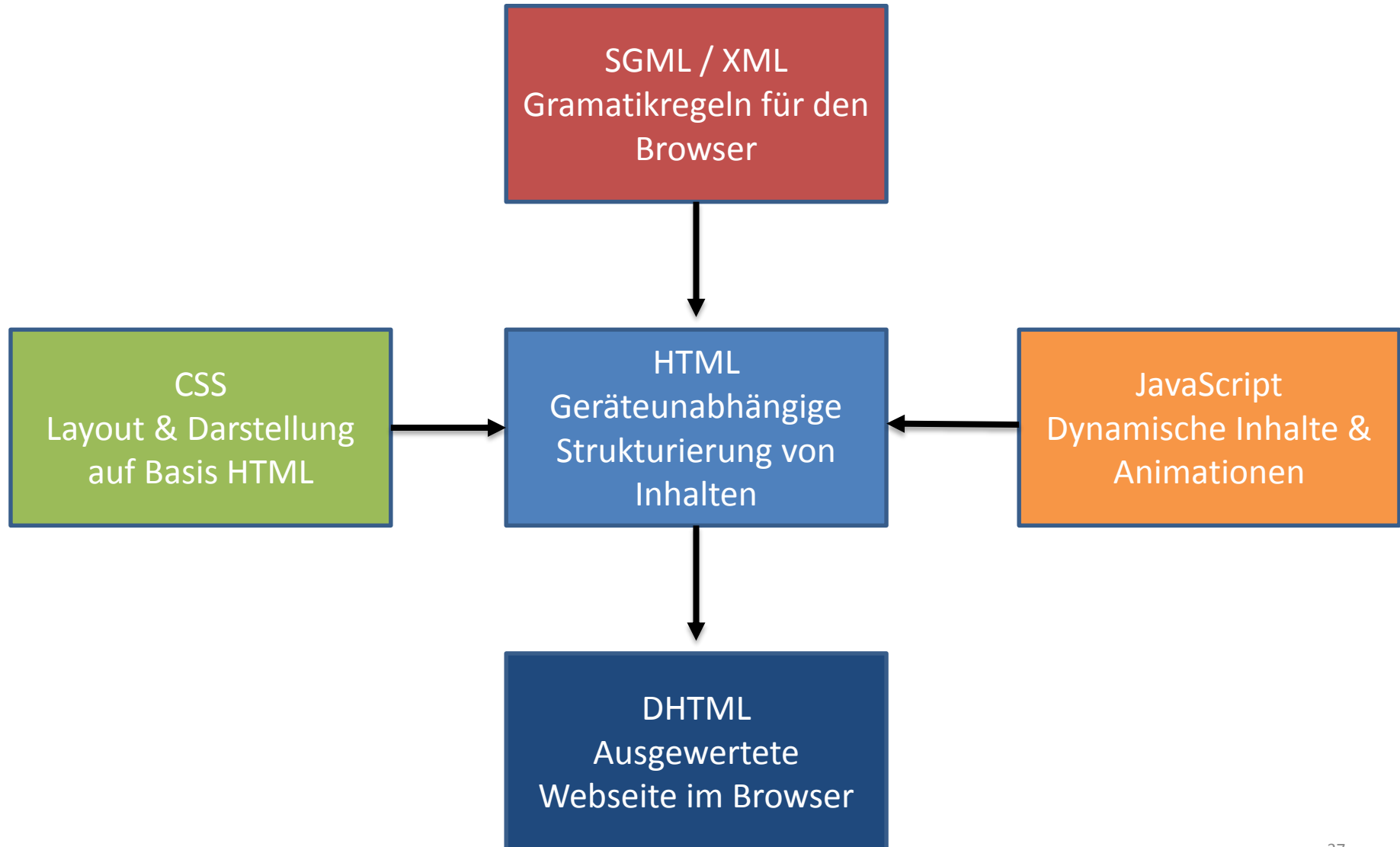
```
<head>
    ...
    <style rel="stylesheet" type="text/css"
          href="styles.css"></style>
</head>
```

- Separate CSS-Datei

```
h1 {
    /* Kommentar */
    font-weight: bold;
    font-size: 26px;
}
```

- CSS Spezifikationen werden erweitert
  - Selektoren
  - Hintergründe und Rahmen
  - Text Effekte
  - 2D/3D Transformationen
- CSS3 befindet sich noch im Entwicklungsstadium





# Einführung JavaScript



- Entwickler: Brendan Eich
  - Softwarehaus: Netscape Com.



- Sep. 1995: LiveScript (Netscape Navigator 2)
- Aug. 1996: JavaScript 1.1. (Navigator 3 Beta)
- Jun. 2008: JavaScript 1.8. (Mozilla)

- JavaScript Browserkompatibilität

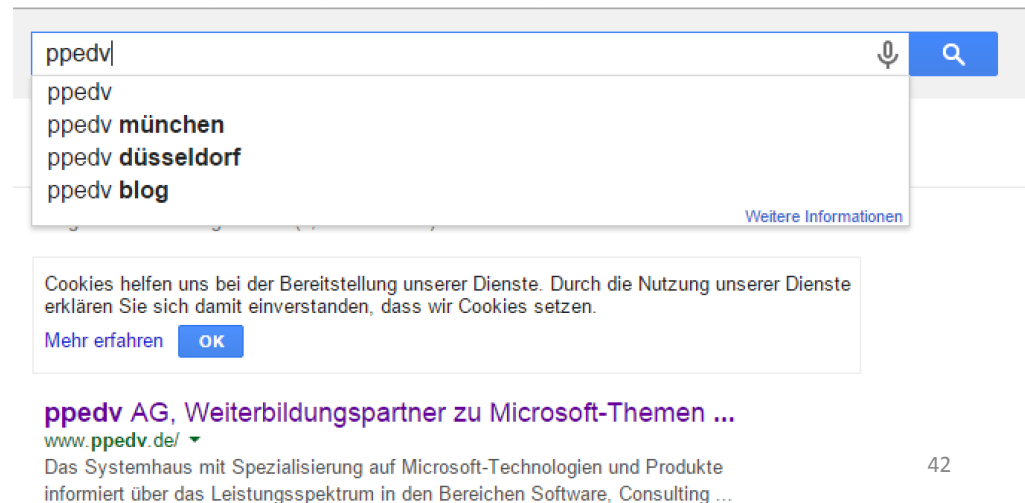
| Version | Release       | Netscape Navigator | Mozilla Firefox | Internet Explorer  | Opera                            | Google Chrome |
|---------|---------------|--------------------|-----------------|--|----------------------------------|---------------|
| 1.0     | März 1996     | ✓ 2.0              |                 | ✓ 3.0  |                                  |               |
| 1.1     | August 1996   | ✓ 3.0              |                 |  |                                  |               |
| 1.2     | Juni 1997     | ✓ 4.0 - 4.05       |                 |  |                                  |               |
| 1.3     | Oktober 1998  | ✓ 4.06 - 4.7x      |                 | ✓ 4.0  |                                  |               |
| 1.4     |               | ✓ Netscape Server  |                 |  |                                  |               |
| 1.5     | November 2000 | ✓ 6.0              | ✓ 1.0           | ✓ 5.5 (JScript 5.5)<br>✓ 6 (JScript 5.6)<br>✓ 7 (JScript 5.7)<br>✓ 8 (JScript 6) | ✓ 6.0<br>✓ 7.0<br>✓ 8.0<br>✓ 9.0 |               |
| 1.6     | November 2005 |                    | ✓ 1.5           |  |                                  |               |
| 1.7     | Oktober 2006  |                    | ✓ 2.0           |  |                                  | ✓ 1.0         |
| 1.8     | Juni 2008     |                    | ✓ 3.0           |  |                                  |               |
| 1.8.1   |               |                    | ✓ 3.5           |  |                                  |               |
| 1.8.5   |               |                    | ✓ 4             | ✓ 9.0 (JScript 9.0)  |                                  |               |



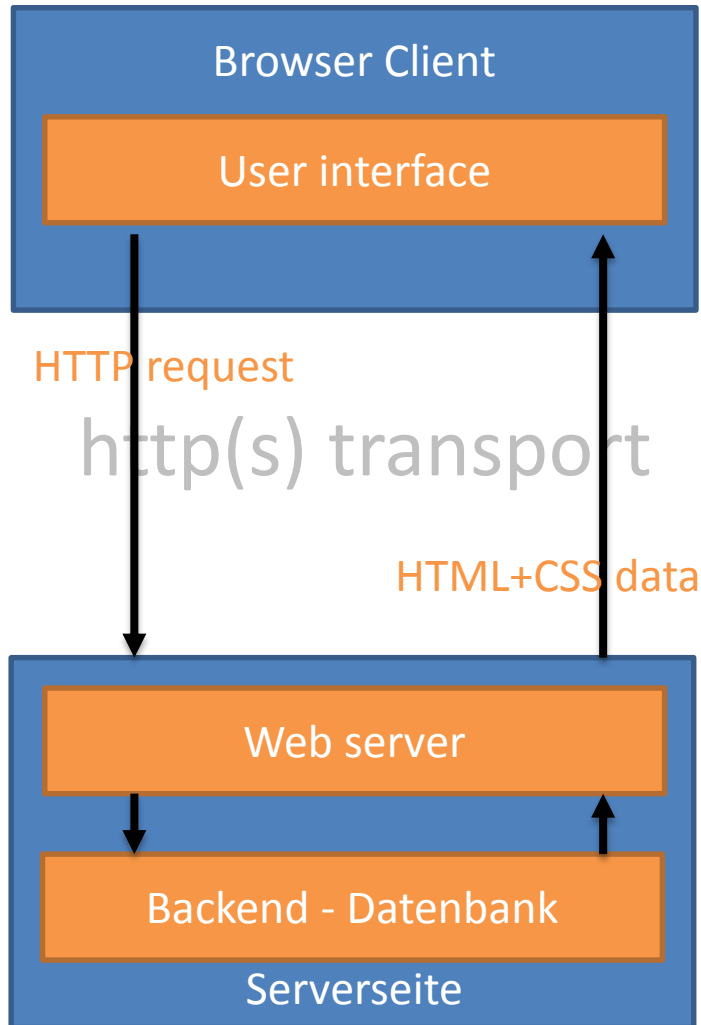
- **Was ist JavaScript?**

- JavaScript ist Scriptsprache und vollwertige Programmiersprache
- JavaScript wird hauptsächlich in Webseiten verwendet, um auf Elemente zugreifen zu können
- ECMAScript (ECMA 262) ist der standardisierte ‚Sprachkern‘
- JavaScript ist dynamisch (lose) typisiert und objektbasierend

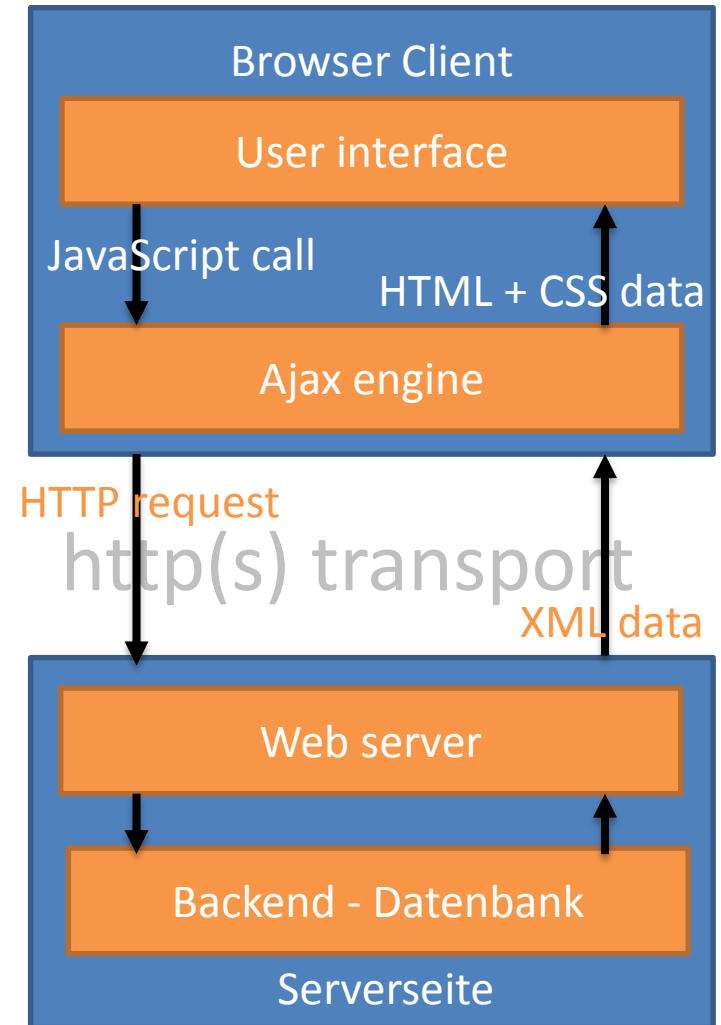
- **Typische Anwendungsgebiete**
  - Dynamische Manipulation von Webseiten über das Document Object Model (**DOM**)
  - Überprüfung von Formulareingaben beim Absender
  - Senden und Empfangen von Daten, ohne dass der Browser die Seite neu laden muss (**AJAX**) z.B. Vorschlägen von Suchbegriffen
  - Rich Internet Applications



The screenshot shows a search bar with the text "ppedv" entered. Below the search bar, a dropdown menu displays suggestions: "ppedv", "ppedv münchen", "ppedv düsseldorf", and "ppedv blog". To the right of the suggestions is a link that says "Weitere Informationen". Below the search bar, there is a cookie notice: "Cookies helfen uns bei der Bereitstellung unserer Dienste. Durch die Nutzung unserer Dienste erklären Sie sich damit einverstanden, dass wir Cookies setzen." Below the notice are two buttons: "Mehr erfahren" and "OK". At the bottom of the screenshot, there is a link to "ppedv AG, Weiterbildungspartner zu Microsoft-Themen ..." with the URL "www.ppedv.de/". Below the link, there is a short description: "Das Systemhaus mit Spezialisierung auf Microsoft-Technologien und Produkte informiert über das Leistungsspektrum in den Bereichen Software, Consulting ...".



Statische Webanwendung



Dynamische Webanwendung

- Kann im Browser deaktiviert werden
- Sicherheitslücken durch fehlerhafte Implementierung
- **Sandbox**
  - Abgeriegelte Umgebung
  - Kein Zugriff auf Dateien des lokalen Rechners
  - Keine Abfrage von Benutzerdaten außerhalb des Browsers
  - Achtung: Beim IE Aktivierung von Active Scripting (ActiveX)!

# Grundlegende Sprachelemente



- Direkt im HTML-Dokument

```
<script type="text/javascript">  
    // Kommentar - Hier die JavaScript-Anweisung einfügen  
</script>
```

- Durch externe Dateien

```
<script type="text/javascript"  
    src="meine_script.js"></script>
```

- Durch EventHandler

```
<button onclick="document.write('Hallo');">Klick</button>
```

- Durch Links

```
<a href="javascript:history.back();">Zurück</a>
```

- Für nicht scriptfähige Browser

```
<noscript>Ihr Browser unterstützt kein JavaScript!</noscript>
```

- Anweisung als atomare Einheit
- Semikolon als Anweisungsabschluss
- Kommentare

```
// Kommentar - Hier die JavaScript-Anweisung einfügen
```

```
/* Mehrzeiliges  
Kommentar */
```



- Benennt Variablen, Konstanten und Funktionen
- Regeln
  - Beginnt mit Buchstaben, \$-Zeichen oder Unterstrich
  - Nur Buchstaben, Ziffern und die Sonderzeichen \$ und \_
  - Case-sensitive (Groß- und Kleinschreibung beachten)
  - Keine Leerzeichen
  - Keine reservierten Wörter verwenden

|          |         |            |              |           |
|----------|---------|------------|--------------|-----------|
| abstract | delete  | function   | null         | throw     |
| boolean  | do      | goto       | package      | throws    |
| break    | double  | if         | private      | transient |
| byte     | else    | implements | protected    | true      |
| case     | enum    | import     | public       | try       |
| catch    | export  | in         | return       | typeof    |
| char     | extends | instanceof | short        | undefined |
| class    | false   | int        | static       | var       |
| const    | final   | interface  | super        | void      |
| continue | finally | long       | switch       | volatile  |
| debugger | float   | native     | synchronized | while     |
| default  | for     | new        | this         | with      |

- Eigenschaften
  - Namen unterliegt den Bezeichner-Vorgaben
  - Definition mit dem Schlüsselwort „var“
  - Können an beliebiger Stelle definiert werden
  - Variablen ohne Wert haben den Zustand „undefiniert“
  - Mehrere Variablen werden durch Kommata getrennt
  - Mehrere Variablen → ein Wert
  - Eine Variable → mehre Datentypen möglich

```
var k;  
var k = 10;  
var i, k = 10;  
var k = 10, i = "Text";
```

- Eigenschaften
  - Name unterliegt den Bezeichner-Vorgaben
  - Definition mit dem Schlüsselwort „const“
  - Wert muss sofort zugewiesen werden
  - Wert kann nicht mehr geändert werden
  - Vereinfachen Lesbarkeit und Wartung eines Programms

```
const MWST = 0.19;  
const rabatt = 10;
```

- Ganze Zahlen

1    23    3874

- Gleitkommazahlen

1190.48    7.6458    188.57

- Zeichenketten (Strings)

"Beispiel"    'Beispiel'    "ein 'wahnsinns' Beispiel"

- Boolesche Werte (Wahrheitswerte)

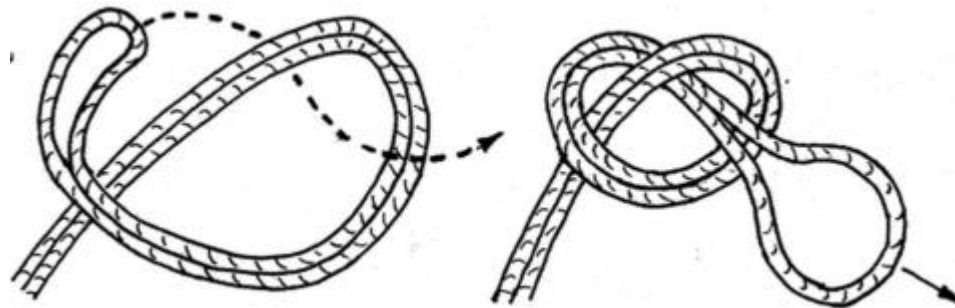
true    false

- Objekte

| Operatoren              | Datentyp                 | Beispiele           |
|-------------------------|--------------------------|---------------------|
| Arithmetischer Operator | Zahlen                   | + - * / ++ -- %     |
| Vergleichsoperatoren    | Zahlen, Strings, Boolean | == != < > <= >= === |
| Verknüpfungsoperatoren  | Strings                  | +                   |
| Logischer Operator      | Boolean                  | &&    !             |
| Bit-Operator            | Zahlen, Boolean          | ~   & ^ << >>       |
| Zuweisungsoperator      | Alle                     | = += -= *= /=       |

| Typ          | Beispiel  | Rückgabewert als Typ |
|--------------|---|----------------------|
| Zahl         | <pre>var Variable = 5;<br/>Typ = typeof Variable;</pre>                   | number               |
| Zeichenkette | <pre>var Variable = "Hallo";<br/>Typ = typeof Variable;</pre>             | string               |
| Boolean      | <pre>var Variable = true;<br/>Typ = typeof Variable;</pre>                | boolean              |
| Undefiniert  | <pre>Typ = typeof VariableX;</pre>  | undefined            |
| Funktion     | <pre>var Variable = new function("5+2");<br/>Typ = typeof Variable;</pre> | function             |
| Array        | <pre>var Variable = [1, 2, 3, 4];<br/>Typ = typeof Variable;</pre>        | object               |
| Null         | <pre>var Variable = null;<br/>Typ = typeof Variable;</pre>                | object               |

# Kontrollstrukturen





- Anweisungsliste zwischen { und }

```
{  
    Anweisung 1;  
    Anweisung 2;  
    Anweisung 3;  
}
```

- Steuermöglichkeit als
  - Auswahl
  - Schleife

```
if(Bedingung) {  
    Anweisungen;  
}
```

- Prüfung der Bedingung
- **true** → Anweisungsblock ausführen
- **false** → Nach Anweisungsblock fortführen

```
var zahl = 5;
```

```
if(zahl == 5) {  
    alert("Die Zahl hat den Wert 5");  
}
```

```
if (Bedingung) {  
    Anweisungen;  
} else {  
    Anweisungen;  
}
```

- Prüfung der Bedingung
- true → Anweisungsblock ausführen
- false → führt else-Anweisungsblock aus

```
var zahl = 4;  
  
if (zahl = 5) {  
    alert("Die Zahl ist 5");  
} else {  
    alert("4 gewinnt");  
}
```

- Der Shocker – Verkürzte Schreibweise

`(Bedingung) ? Anweisung 1 : Anweisung 2;`

- Expertenbeispiel

```
var a = 5;
```

```
(a == 5) ? alert(true) : alert(false);
```



```
var fahrzeug = "Boot";

if (fahrzeug == "Auto") {
    alert("Fahr!");
} else if (fahrzeug == "Boot") {
    alert("Schwimm!");
} else {
    alert("Mach was anderes");
}
```

```
switch (Selektor) {  
    case Wert1:  
        Anweisungen;  
        break;  
    case Wert2:  
        Anweisungen;  
        break;  
    ...  
    default:  
        Anweisungen;  
        break;  
}
```

- Vergleich des Selektors mit den Werten in case
- Zeichenketten oder Ausdrücke möglich
- Wenn **Selektor == case**: Ausführung der Anweisungen bis break;

```
var fahrzeug = "Boot";

switch (fahrzeug) {
  case "Flugzeug":
    alert("Flieg!");
    break;
  case "Boot":
    alert("Schwimm!");
    break;
  case "Auto":
    alert("Fahr!");
    break;
  default:
    alert("Geh zu Fuß!");
    break;
}
```

- Für Wiederholungen eines Anweisungsblocks
- Werden ausgeführt bis
  - Definierte Anzahl an Durchläufen erreicht ist
  - Definierte Bedingung erfüllt ist
- Schleifen
  - for
  - while
  - do...while



```
for (Initialisierung; Bedingung; Aktualisierung) {  
    Anweisung1;  
    Anweisung2;  
    ...  
}
```

```
for(var i = 0; i <= 10; i++){  
    alert(i);  
}
```

```
while(Bedingung){  
    Anweisung1;  
    Anweisung2;  
    ...  
}
```

```
var i = 0;
```

```
while(i <= 10){  
    alert(i);  
    i++;  
}
```

```
do
{
    Anweisung 1;
    Anweisung 2;
    ...
} while (Bedingung)
```

```
var i = 0;
do
{
    alert(i);
    i++;
} while(i <= 10)
```

- **break**
  - Beenden der Schleife
- **continue**
  - Beenden des aktuellen Durchlaufs
  - Fortfahren mit dem nächsten Durchlauf
- **Vorsicht: Darauf achten, dass die Bedingung erfüllt werden kann !!!**

# Funktionen



- Eigenständige Unterprogramme
- Ausführung erst bei Aufruf
- Können einen Wert zurückliefern
- Haben einen eigenen Gültigkeitsbereich
- Syntax

```
function Funktionsname([Parameter1, ... Parameter n])  
{  
    Anweisungen;  
    return Wert;  
}
```

```
...  
Funktionsname([Parameter1, ... Parametern]);  
...
```

```
function bruttoRechner(netto){  
    var brutto = netto * 1.19;  
    return brutto;  
}
```

```
var preis_netto = 125.00;  
var preis_brutto = bruttoRechner(preis_netto);  
document.write(preis_brutto + "Euro");
```

- Keine Methodenüberladung
- Anzahl der Parameter irrelevant
  - Zu wenig: `undefined`
  - Zu viel: `arguments[]`
- Verschachtelung möglich
- Funktionstypen
  - Deklarative Funktionen
  - Anonyme Funktionen
  - Funktionslitterale

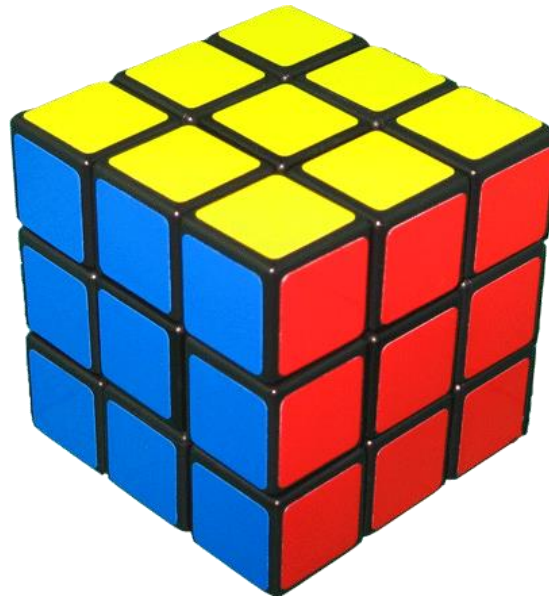


- Lokale Variablen
  - Gültig nur innerhalb von Funktionen
- Globale Variablen
  - Gültig überall!

|   |   |
|---|---|
| <b>eval(Zeichenketten)</b>                              | Gibt Summe der Zeichenkette aus                                     |
| <b>encodeURIComponent, decodeURI</b>                    | Entfernt Zeichen aus URL wie "\"                                    |
| <b>escape(Zeichenkette),<br/>unescape(Zeichenkette)</b> | Kodiert Sonderzeichen mit ein Paar Ausnahmen                        |
| <b>isNaN(Wert)</b>                                      | Ist der Wert keine Zahl?  |
| <b>isFinite(Wert)</b>                                   | Liegt der Wert innerhalb des verarbeitbaren Bereichs von JS         |
| <b>Number(Objekt),<br/>String(Objekt)</b>               | Konvertieren eines Objektes in eine Number oder String Datentyp     |
| <b>parseFloat(Zeichenkette)</b>                         | Konvertierung in Dezimal, bei Fehler gibt diese Funktion NaN zurück |
| <b>parseInt(Zeichenkette)</b>                           | Konvertierung in Integer, bei Fehler gibt diese Funktion NaN zurück |

- Zuweisung statt vergleich
- Groß/Kleinschreibung
- Funktion noch nicht geladen

# Objekte



- Was ist objektorientierte Programmierung?
- Objekte bestehen aus
  - Eigenschaften
  - Methoden
- Vorteile von Objekten
  - Codestrukturen leichter verständlich
  - Kapseln zusammengehörige Daten und Funktionen

- Erstellung der Objektdefinition
- Hinzufügen von Eigenschaften
- Hinzufügen von Methoden
- Erzeugen von Objekten
- Verwenden der Objekte

- Was macht ein Konstruktor?
- Konstruktorfunktion

```
function Auto(Marke, Modell) {  
    this.Marke = Marke;  
    this.Modell = Modell;  
}
```

- Initialisierung des Objektes mit „new“

```
var Golf = new Auto("VW", "Golf VII");
```

- Unterschied: Anonyme Objekte

```
var ObjVar = {  
    Eigenschaft1 : Wert1,  
    Eigenschaft2 : Wert2,  
    Methode1 : new function () {}  
}
```

```
document.write(ObjVar.Eigenschaft1);  
document.write(ObjVar.Eigenschaft2);
```



```
var Golf = new Auto("VW", "Golf VII");
```

- Das Objekt Golf das von der „Klasse“ Auto instanziiert wurde, liegt jetzt im Speicher gefüllt mit den beiden Eigenschaften:
  - Marke
  - Modell
- Das „new“ liefert ebenfalls die Referenz auf die Speicherstelle zurück (object)

```
document.write(typeof (Golf)); // Typ: object
```

- Eigenschaften / Attribute beschreiben Gemeinsamkeiten von Objektklassen
- Die Werte der Eigenschaften können in jedem Objekt unterschiedlich sein
- Deklaration mit dem Schlüsselwort „this“
- Für Eigenschaften gelten die gleichen Regeln wie für Variablen

- Eine Eigenschaft kann auch ein Objekt sein

```
function Person(Vorname, Name) {  
    this.Vorname = Vorname;  
    this.Name = Name;  
}
```

```
var Jogi = new Person("Joachim", "Löw");
```

```
function Auto(Marke, Modell, Fahrer) {  
    this.Marke = Marke;  
    this.Modell = Modell;  
    this.Fahrer = Fahrer;  
}
```

```
var Golf = new Auto("VW", "Golf III", Jogi);
```

- Werte überschreiben
  - Beide Anweisungen führen dieselbe Operation aus:

```
Jogi.Vorname = "Jürgen";
```

```
Golf.Fahrer.Vorname = "Jürgen";
```

```
function Trainer(name) {  
    this.name = name;  
}
```

```
var Kd = new Trainer("Klopp");
```

- Dynamisches Hinzufügen von Eigenschaften

```
Kd.Vorname = "Jürgen";
```

- Dynamisches Löschen von Eigenschaften

```
delete Kd.Vorname;
```

- Methoden = Funktionen einer Objektklasse

```
function Methodenname(Wert1, Wert2, ...) {  
    this.Eigenschaft1 = irgendeinwert;  
}
```

```
function Konstruktorname(Wert1, Wert2, ...) {  
    this.Eigenschaft1 = Wert1;  
    this.eigenschaft2 = Methodenname;  
  
    /* ohne () -> sonst wird der Variablen der  
    Rückgabewert zugeordnet */  
}
```

- **if-Anweisung**

- Objekte auf bestimmte Eigenschaften überprüfen

```
function Kunde(name) {  
    this.name = name;  
}
```

```
var Kd = new Kunde("Konstantin");
```

```
if(Kd.name) alert("Die Eigenschaft existiert");  
else alert("Die Eigenschaft existiert nicht");
```

- **With-Anweisung**

- Vereinfachte Schreibweise

```
function Spieler(name) {  
    this.name = name;  
    this.vorname = vorname;  
}
```

```
var Sp = new Spieler("Alaba", "David");
```

```
with(Sp) {  
    document.write(name + "<br/>");  
    document.write(vorname + "<br/>");  
}
```





- **for-in-Anweisung**

- Alle Eigenschaften eines Objektes durchiterieren

```
for(laufvariable in Sp) {  
    document.write("Eigenschaft: " + laufvariable +  
        ", Wert: " + Sp[laufvariable] + "<br>");  
}
```

Eigenschaft: Name, Wert: Alaba

Eigenschaft: Vorname, Wert: David

- **Instanceof-Operator**

- Überprüfen ob eine Objektinstanz vom Typ eines bestehenden Objektes ist.

```
function Fussballclub() {}
```

```
var Bayern = new Fussballclub();
```

```
if(Bayern instanceof Fussballclub)  
    alert("Bayern ist eine Instanz vom Objekt 'Fussballclub'");
```

# Vordefinierte Objekte



- Beinhalten häufig benötigte Methoden
  - String-Objekt
  - Math-Objekt
  - Number-Objekt
  - Array-Objekt
  - Date-Objekt
  - RegExp-Objekt

- Jede Zeichenkette ist ein String-Objekt
- Eigenschaft: length

```
var zitat = "Das Chancenplus war ausgeglichen"; // Lothar Matthäus
```

| Methode                | Ergebnis                                   |
|------------------------|--|
| zitat.charAt(6)        | a  |
| zitat.indexOf('a')     | 1  |
| zitat.lastIndexOf('e') | 30   |
| zitat.slice(4, 16)     | Chancenplus                                |
| zitat.split(' ')       | (,Das','Chancenplus','war','ausgeglichen') |

| Methode                        | Ergebnis                                      |
|--------------------------------|---|
| zitat.substr(4,11)             | Chancenplus                                   |
| zitat.toLowerCase()            | das chancenplus war ausgeglichen,             |
| zitat.toUpperCase()            | DAS CHANCENPLUS WAR AUSGEGLICHEN,             |
| zitat.concat(' by Lothar M.'); | Das Chancenplus war ausgeglichen by Lothar M. |

- Mathematische Konstanten

| Eigenschaft | Bedeutung                     | Beispiel   | Ergebnis        |
|-------------|-------------------------------|------------|-----------------|
| E           | Eulersche Zahl                | Math.E     | 2.7182818284... |
| LOG2E       | Logarithmus von e zur Basis 2 | Math.LOG2E | 1.4426950408... |
| PI          | Zahl PI                       | Math.PI    | 3.1415926535... |

- Mathematische Funktionen

| Methode      | Bedeutung                      | Beispiel          | Ergebnis        |
|--------------|--------------------------------|-------------------|-----------------|
| cos(Zahl)    | Cosinus                        | Math.cos(Math.PI) | -1              |
| floor(Zahl)  | Abrunden zur nächsten Ganzzahl | Math.floor(3.9)   | 3               |
| random(Zahl) | Zufallszahl zwischen 0 und 1   | Math.random()     | 0.5701611484... |

- Zahlen werden in Number-Objekten abgelegt
- Eigenschaften für numerische Konstanten

| Eigenschaft       | Erläuterung   | Wert                    |
|-------------------|---|-------------------------|
| MAX_VALUE         | Beinhaltet die größte Zahl, die verarbeitet werden kann       | 1.7976931348623157e+308 |
| MIN_VALUE         | Beinhaltet die kleinste Zahl, die verarbeitet werden kann     | 5e-324                  |
| NaN               | Not a Number  | NaN                     |
| NEGATIVE_INFINITY | Wert einer Variablen, wenn die Zahl kleiner ist als MIN_VALUE | -Infinity               |
| POSITIVE_INFINITY | Wert einer Variablen, wenn die Zahl größer ist als MAX_VALUE  | Infinity                |

- Methoden

| Methode                      | Erläuterung   |
|------------------------------|---|
| <code>toExponential()</code> | Gibt eine Zahl in Exponentialschreibweise zurück (ab JavaScript 1.5)<br><code>77.1234.toExponential() = 7.71234e+1</code>                         |
| <code>toFixed()</code>       | Gibt eine Zahl als Dezimalzahl zurück (ab JavaScript 1.5)<br><code>10.1234.toFixed(2) = 10.12</code>  |
| <code>toSource()</code>      | Erzeugt ein neues Objekt auf der Grundlage des angegebenen Objekts (ab JavaScript 1.3)<br><code>10.1234.toSource() = (new Number(10.1234))</code> |
| <code>toString()</code>      | Liefert eine String-Repräsentation des gespeicherten Wertes<br><code>document.write(10.1234 .toString());</code>                                  |
| <code>valueOf()</code>       | Gibt den gespeicherten Wert zurück:<br><code>document.write(10.1234 .valueOf());</code>   |



- Speichern von mehreren Variablen
- Keine Angabe des Typs und der Größe erforderlich
- Muss explizit erzeugt werden
- Array-Index wird in String umgewandelt und als Key im Objekt gespeichert -> Zugriff auch über String-Key möglich
- Array-Objekt erzeugen – Möglichkeit 1

```
var bundesliga = new Array(); // leeres Feld
```

```
bundesliga[1] = 100;  
bundesliga[4] = "Dortmund";
```

- Array-Objekt erzeugen – Möglichkeit 2

```
var bundesliga = new Array(4);
```

```
bundesliga[0] = "Bayern";  
bundesliga[1] = "Dortmund";  
bundesliga[2] = "Hamburg";  
bundesliga[3] = "Hannover";
```

- Array-Objekt erzeugen – Möglichkeit 3

```
var bundesliga = ["Bayern", "Dortmund", "Hamburg", "Hannover"];
```

- **Assoziative Felder**

- Erzeugen

```
var stadt = new Array();  
  
stadt["Wohnort"] = "Salzburg";  
stadt["Hauptstadt"] = "Wien";  
stadt["Kulturhauptstadt"] = "Linz";
```

- Zugriff

```
element = "Hauptstadt";  
stadtauswahl = stadt[element];  
  
// oder  
  
Stadtauswahl = stadt["Hauptstadt"];
```

- Objekt-Eigenschaft: .length
- Werte können später einfach ergänzt werden

```
name[name.length] = "Lehmann";
```

- Elemente löschen

```
myArray = ['a', 'b', 'c', 'd'];
```

```
delete myArray[1]; // ['a', undefined, 'c', 'd']
```

- Weitere Methoden zum Bearbeiten, Suchen und Sortieren

- Datums- und Uhrzeit-Funktionen
- Zugriff auf die Systemzeit
- Kein System-Datum vor dem 01.01.1970
- 3 Methoden um Datumsobjekte zu initialisieren

– Methode 1

```
var Datum = new Date();
```

– Methode 2

```
var Datum = new Date("Monatsname_englisch Tag, Jahr Std:Min:Sek");
```

– Methode 3

```
var Datum = new Date(Jahr, Monat, Tag, St, Min, Sek);
```

- Methoden

| Methode             | Erläuterung                                   | Mögliche Rückgabewerte |
|---------------------|---|------------------------|
| getDate()           | Tag im Monat                                  | 1 bis 31               |
| getDay()            | Nummer des Wochentages                        | 0 bis 6                |
| getHours()          | Stunde  | 0 bis 23               |
| getMinutes()        | Minuten                                       | 0 bis 59               |
| getMonth()          | Nummer des Monats                             | 0 bis 11               |
| getTime()           | Millisekunden seit dem 01.01.1970             | 0 bis ...              |
| getTimezoneOffset() | Abstand zwischen Lokalzeit und GMT in Minuten | -720 bis +720          |
| getFullYear()       | Vierstellige Jahreszahl                       | Jahresangabe           |

- Methoden zum Ändern der Datumsangaben

| Methode                | Erläuterung                     |
|------------------------|---------------------------------|
| setFullYear()          | Ändert das Jahr                 |
| setMonth(Monat)        | Ändert den Monat                |
| setDate(Tag)           | Ändert den Tag des Monats       |
| setHours(Stunde)       | Ändert die Stunde               |
| setMinutes             | Ändert die Minute               |
| setSeconds(Sekunde)    | Ändert die Sekunde              |
| setTime(Millisekunden) | Umrechnung in eine Datumsangabe |





- Regular Expression = Zeichen vergleichen
  - Korrekte Email-Adresse
  - Korrekte Formatierung einer Telefonnummer
- Aufbau eines RegExp-Ausdrucks
  - Pattern
  - Flags (g oder i)
  - Begrenzer „/“ (Delimiter)

**/Pattern/Flag**

- Metazeichen Muster-Definition

| Meta-Zeichen | Findet...                           | Beispiel                     |
|--------------|-------------------------------------|------------------------------|
| \b           | Eine Wortgrenze                     | /\bthe/ = <u>the</u> matisch |
| \B           | Eine Nicht-Wortgrenze               | /\Ber\b = W <u>er</u> t      |
| \d           | Eine Ziffer von 0 bis 9             | /\d\d/ = 42                  |
| \D           | Eine Nicht-Ziffer                   | /\D\D/ = alles außer Zahlen  |
| \s           | Leerzeichen, Tabulator, Umbruch     | /a\s b/ = a b                |
| \S           | Ein Nicht-Leerzeichen               | /a\S b/ = ab                 |
| \w           | Buchstabe, Ziffer oder Unterstrich  | /\w1\w/ = A1_                |
| \W           | Kein Buchstabe, Ziffer, Unterstrich | /1\W/ = 1%                   |
| .            | Alles außer Zeilenumbruch           | /../ = Z3                    |

- Metazeichen Muster-Definition (2)

| Meta-Zeichen        | Findet...  | Beispiel                              |
|---------------------|--|---------------------------------------|
| <code>^</code>      | Beginn einer Zeichenkette                              | <code>/^Frau/</code> = Frau Wagner    |
| <code>\$</code>     | Ende einer Zeichenkette                                | <code>/sie\.\$/</code> = Ich mag sie. |
| <code>[...]</code>  | Irgend ein Zeichen, das in der Klammer aufgelistet ist | <code>/W[oea]rt/</code> = Wert, Wort  |
| <code>[^...]</code> | Keines der in Klammer angegebenen Zeichen              | <code>/W[^ao]rt/</code> = Wirt, Wert  |

- Metazeichen Häufigkeit

| Meta-Zeichen | Findet...                   | Beispiel                                 |
|--------------|-----------------------------|--|
| +            | Ein- oder mehrmals          | <code>/\d+/</code> = 3, 1190             |
| ?            | Kein- oder einmal           | <code>/\d?/</code> = 3, 9 aber nicht 11  |
| *            | kein,- ein- oder mehrmals   | <code>/\d*/</code> = "" oder 18          |
| {n}          | Genau n-mal                 | <code>/\d{3}/</code> = 238 oder 706      |
| {n,}         | n- oder mehrmals            | <code>/\d{2,}/</code> = 28 oder 39746    |
| {n,m}        | Mindestens -, maximal m-mal | <code>/\d{3,5}/</code> = 3 bis 5-stellig |

- Methoden

| Methode            | Erklärung  |
|--------------------|--|
| exec(Zeichenkette) | Diese Methode führt Suche nach dem angegebenen Suchmuster in der Zeichenkette durch. Zurückgeliefert wird ein Array, dass die gefundenen Stellen beschreibt. Wird nichts gefunden, wird <i>null</i> zurückgeliefert. |
| test(Zeichenkette) | Gibt <b>true</b> oder <b>false</b> zurück je nachdem ob der Wert gefunden wurde oder nicht.  |

- Eigenschaften

| Eigenschaft           | Erklärung  |
|-----------------------|--|
| \$_ oder input        | Enthält den durchsuchten Original String                                   |
| \$& oder lastMatch    | Enthält den zuletzt gefundenen String                                      |
| \$+ oder lastParent   | Enthält die zuletzt gefundene Teilsuche                                    |
| \$` oder leftContext  | Enthält den Teilstring links von der gefundenen Stelle                     |
| \$' oder rightContext | Enthält den Teilstring rechts von der gefundenen Stelle                    |
| \$* oder multiline    | Enthält <i>true</i> , wenn die Suche über mehrere Zeilen stattgefunden hat |
| \$1, ..., \$9         | Enthält die Ergebnisse der Teilsuche 1 bis 9                               |

- Reguläre Ausdrücke im String-Objekt

| Methode                        | Erklärung  |
|--------------------------------|--|
| match(Suchmuster)              | Suche nachdem angegebenen Suchmuster durchführen                     |
| search(Suchmuster)             | Liefert Position zurück, an der Stelle an der das Suchmuster zutraf. |
| replace(Suchmuster, Ersetzung) | Der gefundene Teil wird ersetzt                                      |
| split(Suchmuster)              | Die Zeichenkette wird am Suchmuster aufgeteilt                       |

# Objektmodell

Dom (DocumentObjectModell)  
= HTML Dokument





**navigator**

**screen**

**window**

document

**anchor**

**forms**

**elements**

**options**

**images**

**links**

history

location

- Eigenschaften

| Eigenschaft | Bedeutung  |
|-------------|--|
| appName     | Gibt Hersteller Namen des Browsers zurück                |
| appCodeName | Gibt den offiziellen Namen des Browsers zurück           |
| appVersion  | Gibt die Version des Browsers, Plattform und Land zurück |
| userAgent   | Gibt vollständige Browserbezeichnung zurück              |
| platform    | Gibt Betriebssystem zurück                               |
| language    | Gibt Sprache des Client Computers zurück                 |
| plugins     | Gibt Feld mit allen installierten Plugins zurück         |
| mimeTypes   | Gibt alle MIME-Typen des Browser zurück                  |

- Methoden

| Methode       | Bedeutung                                 |
|---------------|---|
| javaEnabled() | Ist Java im Browser aktiviert oder nicht? |

- Benutzung um neue Fenster an der richtigen Position und Größe zu öffnen

| Eigenschaft             | Bedeutung   |
|-------------------------|---|
| height, width           | Höhe und Breite der eingestellten Bildschirmauflösung   |
| availHeight, availWidth | Die Ausmaße der wirklich zur Verfügung stehenden Fläche auf dem Bildschirm, ohne die Windows-Taskleiste |
| colorDepth              | Die verwendete Farbtiefe wird in Bits pro Pixel ausgelesen  |

- Oberste Stelle in der Objekthierarchie
- Elemente des HTML Dokumentes sind Eigenschaften des window-Objektes
- Window enthält
  - document (Enthält alle HTML-Elemente)
  - history (Liste der zuletzt geladenen URIs)
  - location (Infos über die URL der aktuellen Seite)

- Methoden

| Methode                                  | Bedeutung                                 |
|--|---|
| alert()                                  | Meldefenster wird angezeigt               |
| blur()                                   | Deaktiviert aktuelles Browserfenster      |
| close()                                  | Schließt Browserfenster                   |
| confirm()                                | Bestätigungsfenster wird angezeigt        |
| focus()                                  | Aktiviert Browserfenster                  |
| open("URL", "Fenstername", "Optionen")   | Öffnet neues Browserfenster               |
| prompt()                                 | Eingabefenster für Text wird angezeigt    |
| setInterval("JavaScript-Ausdruck", Zeit) | Funktion in einem Zeitintervall ausführen |
| clearInterval(TimeoutID)                 | Laufendes Interval wird gestoppt          |
| setTimeout("JavaScript-Ausdruck", Zeit)  | Laufende Stoppuhr wird gestartet          |
| clearTimeout(TimeoutID)                  | Laufende Stoppuhr wird angehalten         |

- Eigenschaften

| Eigenschaft   | Bedeutung   |
|---------------|---|
| defaultStatus | Text in der Statuszeile des Browsers als Standardtext festlegen |
| opener        | Referenz auf das Fenster, das das aktuelle Fenster geöffnet hat |
| self          | Referenz auf das aktuelle Objekt                                |
| status        | Ändert Text der Statuszeile                                     |
| document      | Zugriff auf HTML Dokument                                       |

- Fenster öffnen und schließen

```
window.open("URL", "Fenstername", [Optionen]);
```

- Wenn URL leer, dann wird ein leeres Dokument geöffnet
- Fenstername vergeben um Fenster anzusprechen
- Über Optionen kann die Anzeige des Fenster konfiguriert werden
- Rückgabewert ist eine Referenz auf das window-Objekt



- Optionen für die Anzeige von Fenstern

| Methode     | Bedeutung                             |
|-------------|---------------------------------------|
| menubar     | Anzeige der Menüleiste (ja oder nein) |
| toolbar     | Anzeige der Symbolleiste              |
| locationbar | Anzeige der Adressleiste              |
| status      | Anzeige der Statusleiste              |
| resizeable  | Fenstergröße veränderbar              |
| scrollbars  | Bildlaufleisten einblenden            |
| height      | Festlegung der Höhe in Pixeln         |
| width       | Festlegung der Breite in Pixeln       |
| top         | Anzeigeposition von oben              |
| left        | Anzeigeposition von links             |

- Beispiel Fenster öffnen

```
var Fenster = window.open("", "Testname", "menubar=no,  
locationbar=no, resizable=no, status= no, height=300,  
width=300");
```

- Beispiel Fenster schließen

```
<a href="javascript:window.close();">schließen</a>
```

- Zeitgeber verwenden
  - Anweisungen oder Funktionen nach einer bestimmten Zeit durchführen
  - Methode: setTimeout()

```
function ZeitVorbei() {  
    alert("Die Zeit ist abgelaufen, Schalke ist Meister");  
    return  
}
```

```
window.setTimeout("ZeitVorbei()", 5000);
```

- Meldungsfenster
  - `window.alert("Meldung")`
  - Zeilenumbrüche innerhalb von Meldungen mit „\n“

```
window.alert(„Gratuliere!\nBayern ist Meister!");
```

- Eingabefenster
  - `window.prompt("Text", "Vorgabewert")`
  - Rückgabewert
    - Der Eingabewert, wenn der Benutzer die Eingabe OK bestätigt hat
    - Der Wert „null“, wenn er die Eingabe über die Schaltfläche „abbrechen“ abgebrochen hat.

- Bestätigungsfenster
  - `window.confirm("Frage")`
  - Rückgabewert
    - **true**, wenn der User die Frage mit OK bestätigt
    - **false**, wenn der User die Frage mit ABBRECHEN bestätigt

**document**

**all**

**style**

**anchor**

**forms**

**elements**

**options**

**images**

**links**

| Eigenschaft  | Bedeutung  |
|--------------|--|
| bgColor      | Hintergrundfarbe lesen oder ändern                             |
| fgColor      | Textfarbe lesen oder ändern                                    |
| linkColor    | Farbe der nicht besuchte Links lesen oder ändern               |
| alinkColor   | Legt Farbe eines aktiven Hyperlinks fest oder liest sie aus    |
| vlinkColor   | Farbe der bereits besuchte Links lesen oder ändern             |
| title        | Titel der Seite auslesen                                       |
| referrer     | Liefert URL zurück die auf das aktuelle Dokument verwiesen hat |
| lastModified | Datum der letzten Veränderung des Dokumentes                   |
| cookie       | Zugriff auf Cookies  |
| URL          | Gibt URL des geladenen Dokumentes zurück                       |



- Methoden

| Methode                     | Bedeutung  |
|-----------------------------|--|
| <code>write(Text);</code>   | Schreibt Text in das HTML-Dokument. Dieser wird an der Stelle eingefügt, an der die Methode aufgerufen wird. |
| <code>writeln(Text);</code> | Wie <code>write(Text)</code> lediglich danach noch ein <code>&lt;br&gt;</code>                               |
| <code>open();</code>        | Webseite wird "gelöscht" und neue Seite wird geöffnet  |
| <code>close();</code>       | Schließt Dokument  |

- Zugriff auf HTML-Elemente über vergebene Namen
- Objekte der Elemente sind auch in Arrays angelegt
  - `window.document.images[]`
  - `window.document.anchor[]`
  - `window.document.links[]`
  - `window.document.forms[]`

- Aufzeichnung der zuletzt besuchten URLs
- Nur vorwärts und rückwärts zugreifbar – nicht direkt

| Eigenschaft | Bedeutung   |
|-------------|---|
| length      | Gibt Anzahl der Einträge im history-Objekt zurück |

| Methode       | Bedeutung                                     |
|---------------|---|
| back();       | Lädt zuvor besuchte Webseite                  |
| forward();    | Lädt das nächste Objekt innerhalb der History |
| go(Schritte); | Lädt einzelne Schritte nach Index             |

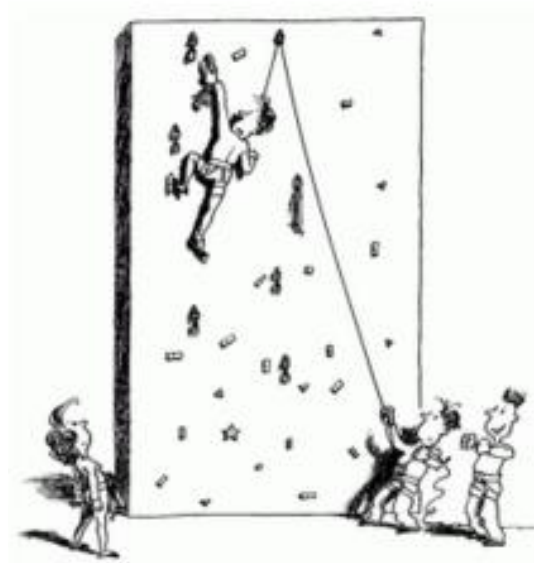
- Objekt zum auslesen und bearbeiten von URLs

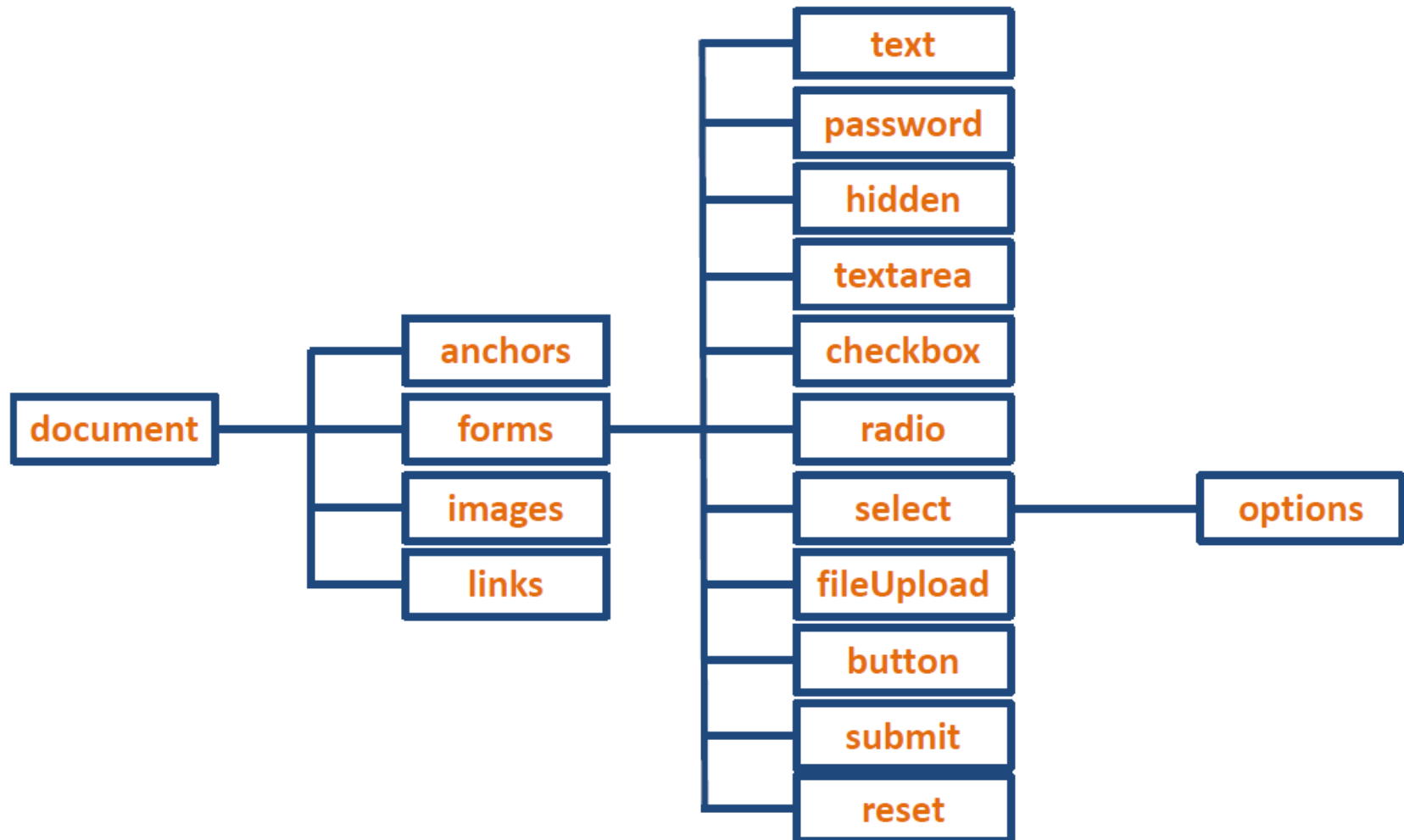
| Eigenschaft | Bedeutung                                    |
|-------------|--|
| protocol    | Gibt verwendetes Internet-Protokoll zurück   |
| hostname    | Liest Host-und Domainnamen aus               |
| port        | Gibt den verwendeten Port des Servers zurück |
| pathname    | Gibt Pfadangabe des Dokuments zurück         |
| search      | Gibt Querystring zurück                      |
| hash        | Gibt URL Anker zurück                        |
| href        | Gibt die vollständige URL zurück             |
| host        | Gibt Hostname:Port zurück                    |

- Methoden

| Methode   | Bedeutung                                     |
|-----------|---|
| assign()  | Lädt ein neues Dokument                       |
| reload()  | Neu laden                                     |
| replace() | Ersetzt das aktuelle Dokument durch ein Neues |

# Zugriff auf HTML-Dokumente





- Verweisanker in einer Webseite

```
<a name="absatz2">Überschrift Absatz 2</a>
```

- Eigenschaften

| Eigenschaft | Bedeutung                                      |
|-------------|--|
| length      | Anzahl der Verweisanker innerhalb der Webseite |
| name        | Zugriff auf den Namen des Ankers               |
| text        | Zugriff auf den Text des Ankers                |



- Formular: Ermöglicht Eingabe von Daten

```
<form name="Formular" action="send.php" method="post"  
encoding="text/plain">
```

- Eigenschaften

| Eigenschaft | Bedeutung  |
|-------------|--|
| action      | URL des Scripts, das die Daten verarbeiten soll  |
| encoding    | Gibt die Kodierung der Daten an                  |
| length      | Anzahl der Formulare                             |
| method      | Setzt POST/GET als Sendemethoden                 |
| name        | Name des Formulars                               |
| target      | Zielfenster wo das Formular geöffnet werden soll |

- Methoden

| Methode  | Bedeutung  |
|----------|--|
| reset()  | Setzt alle Einträge des Formulars auf den Anfangswert zurück |
| submit() | Daten werden an das in action definierte Ziel gesendet       |

- Eigenschaften und Elemente ansprechen:
  - `document.forms[0].action`
  - `document.FormName.Elements[3].Name`
  - `document.FormName.ElementName.Eigenschaft`

- 3 Arten von Textfelder
  - `<input type=„text“>`
  - `<input type="password">`
  - `<textarea></textarea>`
- Methoden

| Methode               | Bedeutung                                  |
|-----------------------|--|
| <code>blur()</code>   | Feld wird verlassen                        |
| <code>focus()</code>  | Setzt den Cursor in das entsprechende Feld |
| <code>select()</code> | Selektiert Inhalt des Feldes               |

- Eigenschaften

| Eigenschaft  | Bedeutung  |
|--------------|--|
| defaultValue | Liest vorgegeben HTML-Wert des Eingabefeldes aus       |
| form         | Name des Formulars indem sich das Eingabefeld befindet |
| name         | Feldname   |
| type         | Typ des Objektes (text, password)                      |
| value        | Inhalt des Eingabefeldes                               |

- Eigenschaften

| Eigenschaft    | Bedeutung                                      |
|----------------|--|
| checked        | Feld markiert oder nicht?                      |
| defaultChecked | Feld beim laden bereits selektiert oder nicht? |
| form           | Name des Formulars                             |
| length         | Anzahl der Felder in einer Optionsgruppe       |
| name           | Name des Controls                              |
| type           | Typ des Objektes (Checkbox, Radio)             |
| value          | Liest Übergabewert                             |

- Methoden

| Methode | Bedeutung   |
|---------|-------------|
| click() | Click Event |

- Eigenschaften

| Eigenschaft   | Bedeutung                                    |
|---------------|--|
| form          | Formularname                                 |
| length        | Anzahl der Optionen                          |
| name          | Name des Controls                            |
| options[]     | Zugriff auf alle Elemente der DropDownList   |
| selectedIndex | Zugriff auf selektierten Wert                |
| type          | Auswahlmethode (select-one, select-multiple) |
| value         | Liest Übergabewert                           |

- Methoden

| Methode | Bedeutung                                  |
|---------|--|
| blur()  | Feld wird verlassen                        |
| focus() | Setzt den Cursor in das entsprechende Feld |
| click() | Click Event                                |

- Nicht sichtbar, werden aber versendet

```
<input type="hidden" />
```

- Eigenschaften

| Eigenschaft | Bedeutung          |
|-------------|--------------------|
| form        | Name des Formulars |
| name        | Name des Objekts   |
| type        | Typ des Objekts    |
| value       | Liest Übergabewert |



- Dienen zum starten von Operationen
- Eigenschaften
  - form, name, type, value
- Methoden
  - click()
- Event zur Button-Nutzung
  - onClick()

- Eingaben prüfen
  - Event onSubmit()
- Sicherheitsabfrage vor Reset
  - Event onReset()

- `document.forms[0].elements[0].value`
- `document.forms[0].elements["Vorname"].value`
- `document.forms[0].Vorname.value`
  
- `document.forms["Form1"].elements[0].value`
- `document.forms["Form1"].elements["Vorname"].value`
- `document.forms["Form1"].Vorname.value`
  
- `document.Form1.Vorname.value`

- Eigenschaften

| Eigenschaft    | Bedeutung                     |
|----------------|-------------------------------|
| border         | Rahmen an/aus                 |
| complete       | Laden der Grafik erfolgreich? |
| height, width  | Höhe u. Breite setzen         |
| hspace, vspace | Abstand einer Grafik zum Text |
| src            | Quelle der Grafik             |
| name           | Name der Grafik               |
| length         | Anzahl der Grafiken           |

- Ansprechen
  - `document.images[0]`
  - `document.Bildname`

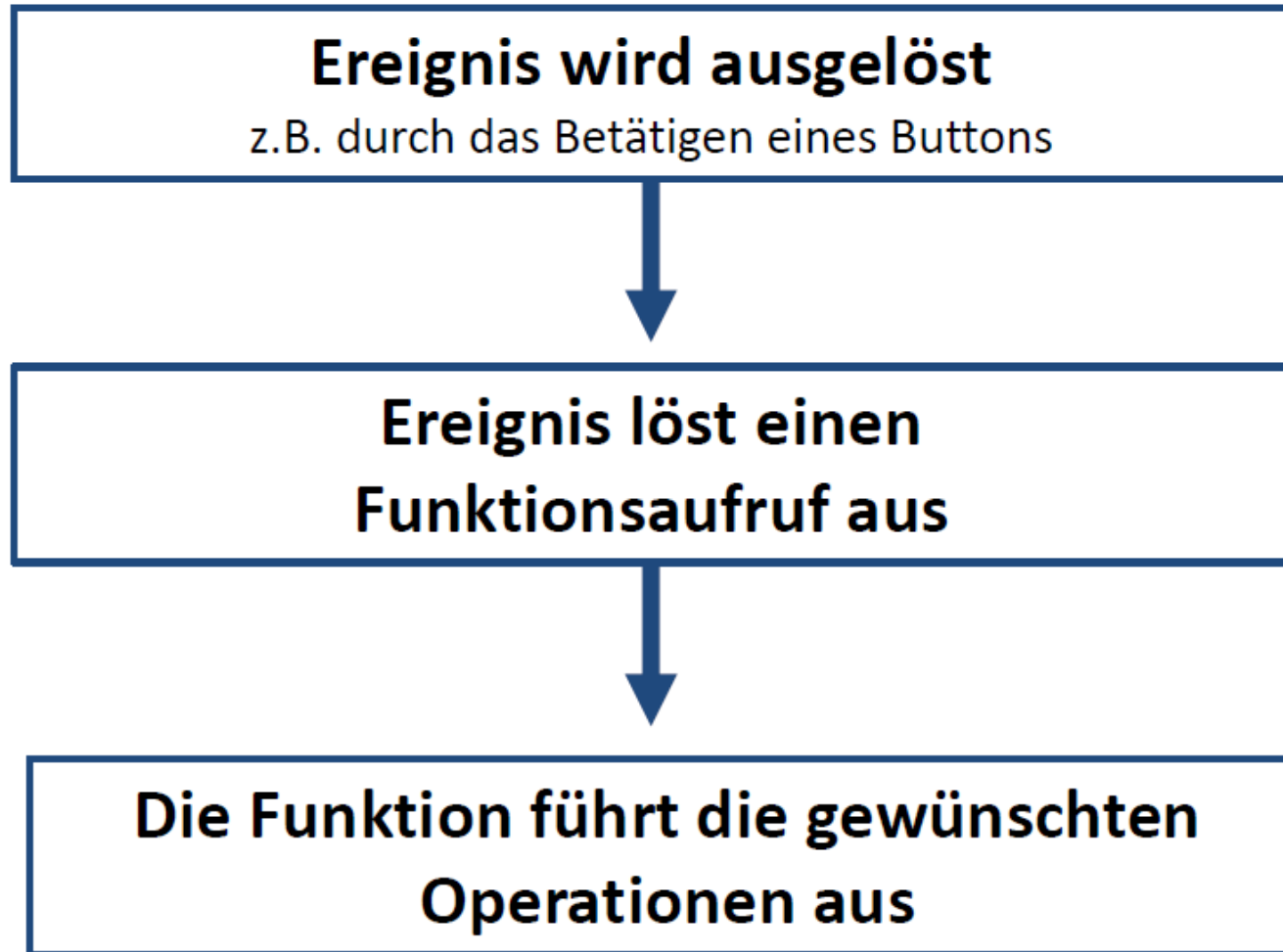
- Zugriff auf alle Verweise der Webseite
  - links[0]
- Eigenschaften

| Eigenschaft | Bedeutung                                  |
|-------------|--|
| length      | Anzahl der Verweise innerhalb der Webseite |
| name        | Name des Verweises                         |
| target      | Ziel-Fenster des Verweises                 |
| text        | Text des Verweises                         |

# Event-Handler



- Events: Ereignisse die durch die Interaktion des Benutzers auftreten
- Beispiel:
  - onClick
  - onSubmit
- Eventhandler verweist bei bestehenden Ereignissen auf dafür vorgesehene Funktionen





| Eigenschaft | Bedeutung                             | Erlaubt in                |
|-------------|---------------------------------------|---------------------------|
| onAbort     | Bei Abbruch des Ladens einer Webseite | <img>                     |
| onBlur      | Beim Verlassen eines Elements         | <a><area><button><input>  |
| onChange    | Bei Änderung von Angaben              | <input><select><textarea> |
| onClick     | Beim Mausklick                        | In fast allen HTML-Tags   |
| onDbClick   | Beim doppelten Anklicken              | In fast allen HTML-Tags   |
| onError     | Im Fehlerfall                         | <img>                     |
| onFocus     | Beim Aktivieren eines Elements        | <a><area><button><input>  |
| onKeyDown   | Beim Betätigen einer Taste            | In fast allen HTML-Tags   |
| onKeyPress  | Beim Betätigen einer Taste            | In fast allen HTML-Tags   |
| onKeyUp     | Nach dem Loslassen einer Taste        | In fast allen HTML-Tags   |

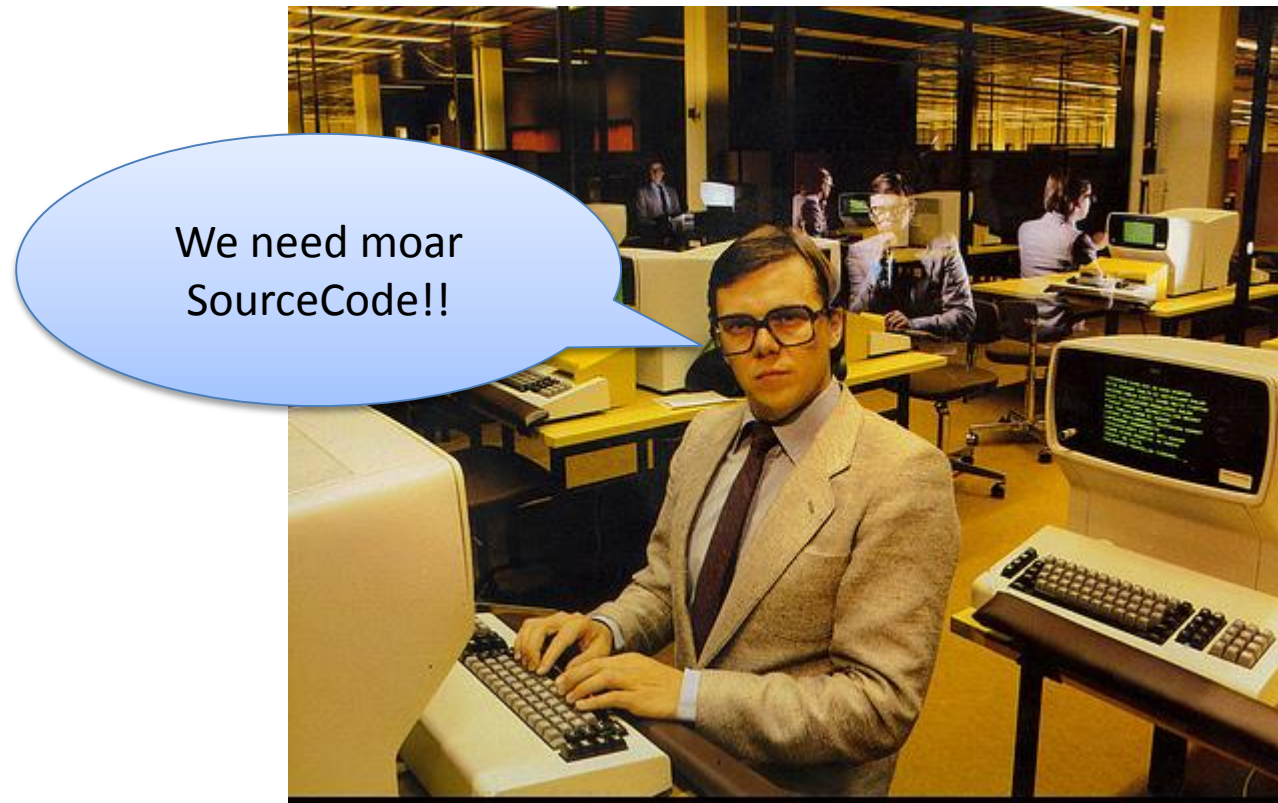
| Eigenschaft | Bedeutung                                  | Erlaubt in              |
|-------------|--|-------------------------|
| onLoad      | Beim Laden einer Webseite                  | <body>                  |
| onMouseDown | Beim Betätigen der Maustaste               | In fast allen HTML-Tags |
| onMouseOut  | Beim Verlassen eines Elementes m. d. Maus  | In fast allen HTML-Tags |
| onMouseOver | Beim Überfahren eines Elementes m. d. Maus | In fast allen HTML-Tags |
| onMouseUp   | Nachdem Loslassen der Maustaste            | In fast allen HTML-Tags |
| onReset     | Beim Zurücksetzen                          | <form>                  |
| onSelect    | Beim Selektieren von Text                  | <input><textarea>       |
| onSubmit    | Beim Absenden von Formulardaten            | <form>                  |
| onUnload    | Beim Verlassen der Webseite                | <body>                  |

| Eigenschaft | HTML-Tags | Wird aktiviert, wenn...                     |
|-------------|-----------|---|
| onAbort     | <img>     | Der User das Laden einer Grafik abbricht    |
| onClick     | <a> <img> | Der User mit der Maus auf einen Link klickt |
| onError     | <img>     | Die Grafik nicht geladen werden konnte      |
| onMouseOut  | <a> <img> | Maus verlässt Grafik-oder Linkbereich       |
| onMouseOver | <a> <img> | Maus befindet sich über Grafik oder Link    |

| Eigenschaft | HTML-Tags  | Wird ausgelöst, wenn...                     |
|-------------|--|---|
| onBlur      | <code>&lt;input type="select   text   textarea"&gt;</code>                   | Der User das Feld verlässt                  |
| onChange    | <code>&lt;input type="select   text   textarea"&gt;</code>                   | Der User das Feld verlässt und geändert hat |
| onClick     | <code>&lt;input type="button   checkbox   radio   reset   submit"&gt;</code> | Der User das Element anklickt               |
| onFocus     | <code>&lt;input type="select   text   textarea"&gt;</code>                   | Der User das Element aktiviert z.B. Tab     |
| onReset     | <code>&lt;form&gt;</code>  | Der User Reset betätigt                     |
| onSelect    | <code>&lt;input type="text   textarea"&gt;</code>                            | Der User Textstelle markiert                |
| onSubmit    | <code>&lt;form&gt;</code>  | Der User Submit betätigt                    |

- Erlaubt Ausführung von JavaScript in Hyperlinks
- Nur in Verbindung mit `<a href="">`

```
<a href="javascript: zeigeMeldung();">
```



# JS Frameworks

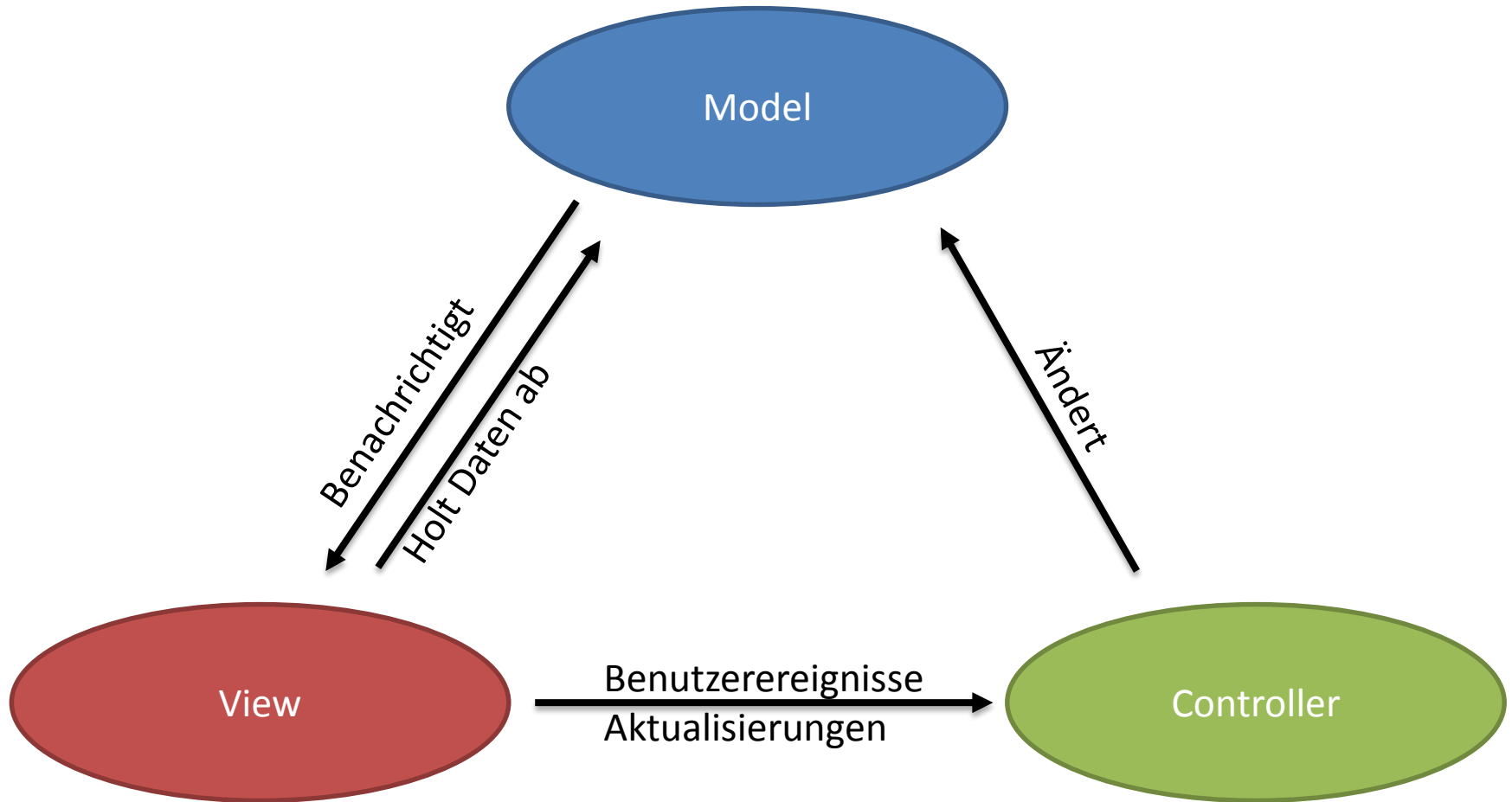
- JavaScript Framework:
  - JQuery Version 1x funktioniert in allen Browsern
  - Version 2.x wird ab IE 9 unterstützt
  - Funktionsumfang erleichtert JavaScript Entwicklung
- Inhalt der Bibliothek:
  - Elementselektion
  - Funktionen zum DOM
  - Animationen und Effekte, mit JQuery-UI auf Oberflächenelemente
  - AJAX-Funktionalitäten



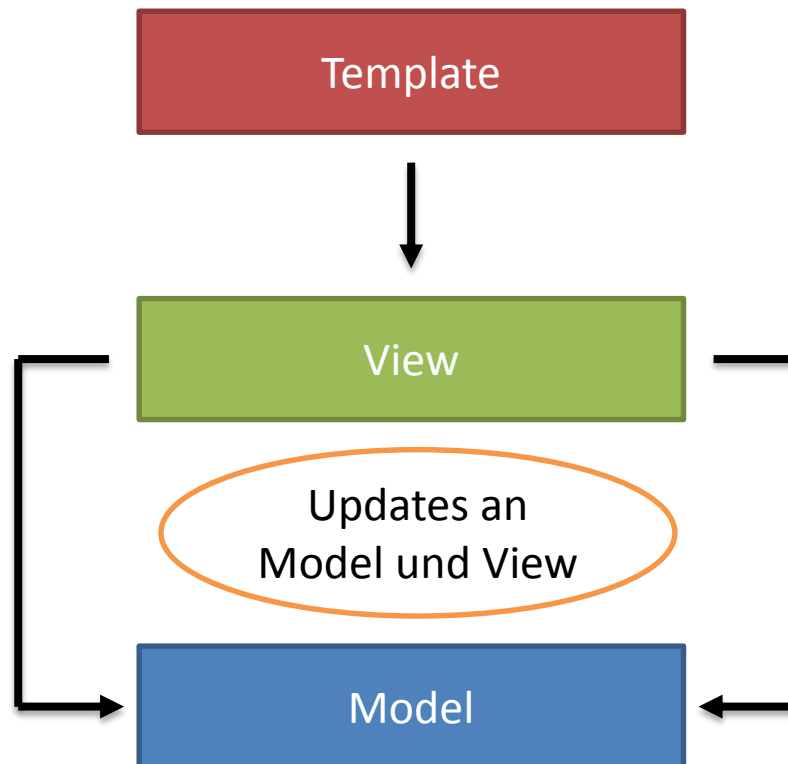


- Web-App, die aus einer HTML-Seite besteht
- Die Seite wird nicht verlassen
- Daten- oder benutzergetrieben werden Inhalte ausgetauscht
- Auf Basis von AJAX, JSON, REST und HTML-Templates
- Routings definieren was wann gezeigt wird, inklusive History-Back





## Zwei-Weg Datenbindung



- Framework zum Entwickeln von User Interfaces auf Basis von HTML und JavaScript
- Single-Page Applikationen für Desktop und mobile

