

# ***Virtuelle Realität und Animation (= Autorensysteme)***

Sommersemester 2023

Carsten Lecon  
Stefan Wehrenberg

# *Info zu Projektaufgaben*

- 4 Projekte
  - **SVG Animation** [5 Punkte]
    - (ca. 1 Monat Bearbeitungszeit)
  - **HTML/CSS/jQuery Animation** [10 Punkte]
    - (ca. 1 Monat Bearbeitungszeit)
  - **Blender Modellierung und Animation** [25 Punkte]
    - (ca. 1 Monat Bearbeitungszeit)
  - **Unity-VR Spiel** [60 Punkte]
    - (ca. 2,5 Monate Bearbeitungszeit)

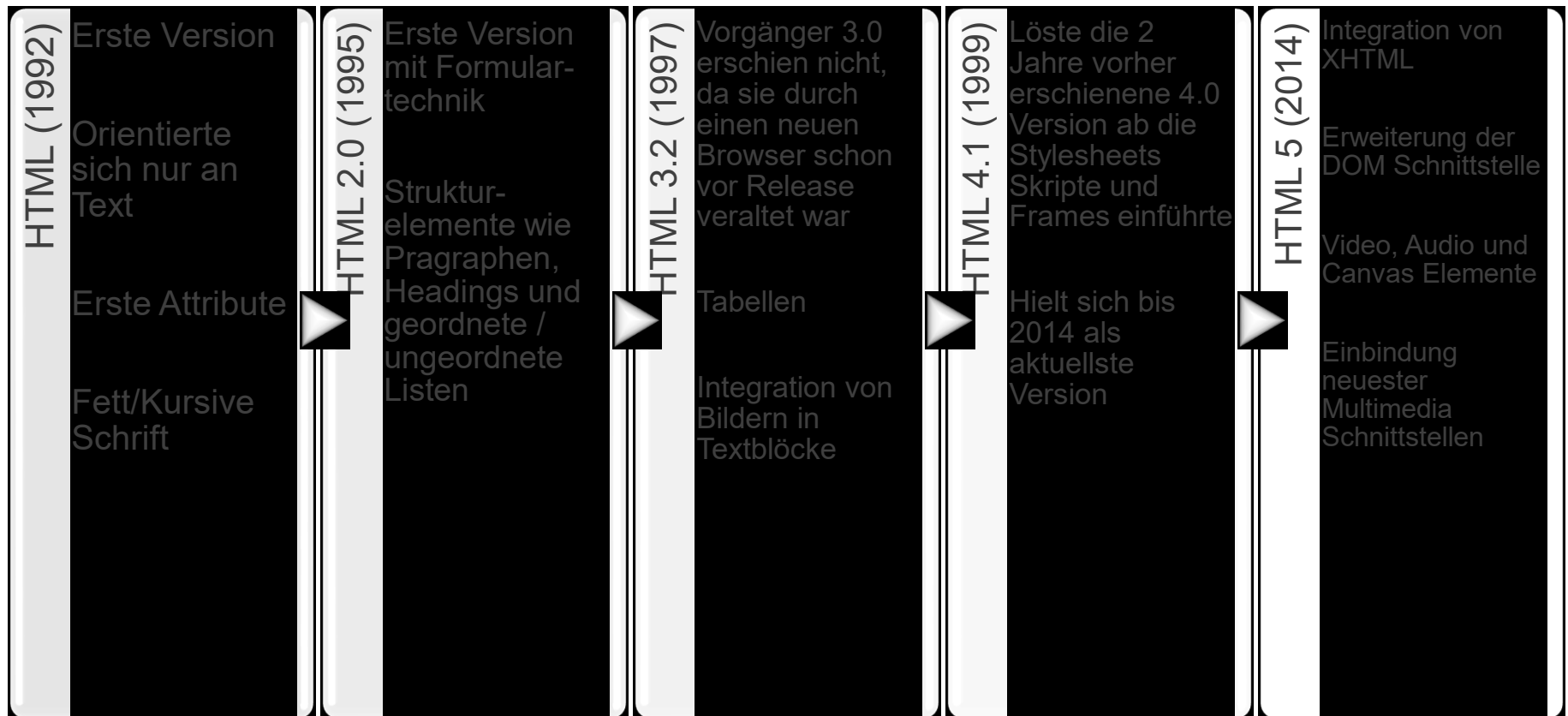
# ***HTML5***

# Übersicht

- Historie
- HTML5-Erweiterungen
- Neue semantische Elemente
- Formulare mit HTML5
- Medienelemente
- Canvas

# *Historie*

# HTML-Historie



# ***HTML5-Features***

# Features von HTML5 (1/3)

- Dokumenttypangabe
  - `<!DOCTYPE html>`
- Einbindung von SVG und MathML
  - Bedingungen:
    - Elemente dürfen keine Namensraumpräfixe enthalten
    - Namensraumspräfix bei XLink muss „xlink“ lauten
  - Benannte Entities aus SVG und MathML sind möglich
- Neue semantische Elemente
- Strukturierende Elemente
  - Bisläng meist `div`, nun:  
`section`, `nav`, `article`, `aside`,  
`hgroup`, `header`, `footer`



## Features von HTML5 (2/3)

- Gruppierungselement **figure**
  - Beschriftung von Abbildungen durch **figcaption**
- Elemente zur Textauszeichnung
  - **time** (Zeitangabe), **mark** (Texthervorhebung),  
**wbr** (Textumbruch), **ruby/rp/rt** (Ruby-Annotierungen)
- Multimedia-Elemente
  - Einfache Einbindung von Audio- und Videodateien
  - Zeichenoberfläche durch **canvas**
- Formularelemente
  - Erweiterung des **input** für bestimmte Datentypen (Validierung)
  - Weitere Elemente: **datalist**, **output**, **progress**, **meter**
  - Automatische Validierung erst beim *Submit!*

# Features von HTML5 (3/3)

- Interaktive Elemente
  - Ein-/Ausblenden von Elementen: `details`, `summary`
  - Werkzeugleisten, (Kontext-) Menüs: `menu`, `command`
- Geänderte Bedeutung:
  - Semantisierung von Elementen: `b`, `i`, `hr`, `small`, ...
  - Umdeutung: `cite`
  - Entfernung von Elementen:
    - `acronym`, `center`, `font`, `blink`, `marquee`, ...
    - `applet` ersetzt durch `embed` (Einbinden von Medien)
    - `frame` ersetzt durch `iframe` (Einbinden weiterer HTML Seiten)
- Liste der HTML-Tags

# *Erweiterung der DOM-Schnittstelle*

- Kontrolle von Multimedia-Elementen
- Manipulation des Browserverlaufs
- Drag & Drop
- Bearbeitbare Inhalte
- Offline-Anwendungen
- Speichern von Anwendungsdaten

# ***Neue semantische Elemente***

# Neue semantische Elemente (1/4)

- **<article>**
  - Eigenständiges Seitenelement, Zusammenstellung weiterer Inhaltselemente
  - Lässt sich unabhängig wiederverwenden
  - Beispiele: Blogbeiträge, Forenbeiträge, Benutzerkommentare, Nachrichtenartikel
- **<section>**
  - Teil einer Seite, die (meist) ein Thema behandelt, mit optionaler Überschrift
  - Abgrenzung einzelner Teile einer Seite, bspw. Kapitel eines Buches

# Neue semantische Elemente (2/4)

- **<header>**
  - Navigationshilfe oder Einführung in ein Thema
  - Enthält meist Überschriften (**<h1>**...**<h6>**) eines Abschnitts oder eine **hgroup**
- **<hgroup>**
  - Abschnittsüberschrift, die mehrere Überschriften gruppieren kann, z.B. alternative Titel oder kurze Beschreibungen neben der Hauptüberschrift

## Neue semantische Elemente (3/4)

- **<nav>**
  - Navigationsbereich mit Verknüpfungen zu anderen Seiten oder Seiteninhalten (nur Seitennavigation, nicht gedacht zur Definition einer Gruppen von Verlinkungen)
- **<footer>**
  - Fußzeile für den Abschnittsbereich
  - Z.B. Angaben über Autor, Copyright, Verlinkungen zu verwandten Dokumenten
  - Ggf. Untergliederung durch **sections**

# Neue semantische Elemente (4/4)

- **<aside>**
  - Inhaltsabschnitt, der mit dem eigentlichen Inhalt thematisch nur am Rande etwas zu tun hat
  - Z.B. für Seitenleisten, Werbung, Begriffserklärungen
- **<time>**
  - Zeitangabe im 24-Stunden-Format
  - Kann Text enthalten („menschenlesbares Format“)
- **<mark>**
  - Markierung von Texten
  - Z.B. Hervorhebung von Treffern einer Suchanfrage

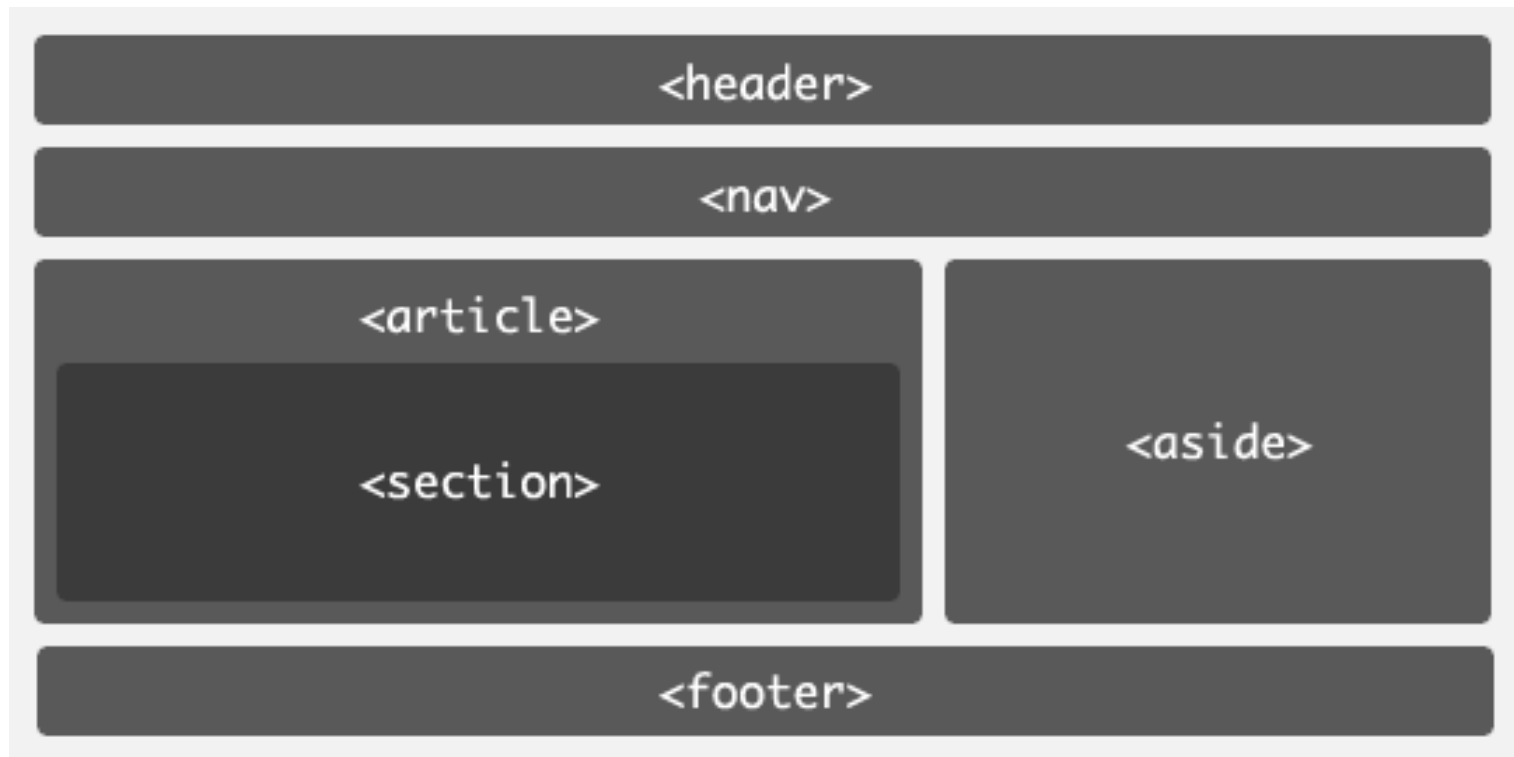


# Strukturierung mit Strukturelementen



Bildquelle: [http://www.normansblog.de/html5-css3/ressources/html5\\_structure.png](http://www.normansblog.de/html5-css3/ressources/html5_structure.png)

# Strukturierung mit Strukturelementen



Bildquelle: <http://www.alistapart.com/d/previewofhtml5/structure-html5.gif>

# Beispiel Strukturelemente

- Nachrichtenseite:
  - Bericht (**article**)
    - Kopf (**header**)
      - Überschrift (**h1**)
      - Erscheinungsdatum (**time**)
    - Text (**p**)
    - Fußzeile (**footer**)
      - Impressum (**i**)
  - Kommentar (**article**)
    - Kopf (**header**)
      - Überschrift (**h4**)
      - Erscheinungsdatum (**time**)
    - Text (**p**)

# Beispiel Strukturelemente



# Beispiel Strukturelemente

```
<article>
  <header>
    <h1>Der Weltuntergang wurde verschoben</h1>
    <time datetime="2012-12-21">21.12.2012</time>
  </header>
  <p>Wider Erwarten wurde...</p>
  <footer>
    <i>PICTURE-Zeitung, 2012</i>
  </footer>
</article>
```

```
<article>
  <header>
    <h4>... Rechtfertigung</h4>
    <time datetime="2012-12-22">22.12.2012</time>
  </header>
  <p>Als erfahrener Wissenschaftler...</p>
</article>
```

1\_h5\_article.html

# ***Formulare mit HTML5***

## Formularfeld: Email ("email")

- Muss eine korrekt formatierte Emailadresse erhalten

`<form>`

Email-Adresse:

`<input type="email" name="email" />`

`</form>`

Email-Adresse:

Bitte geben Sie eine E-Mail-Adresse ein.

2\_h5\_formular.html

## Formularfeld: URL ("url")

- Muss mit "http" beginnen

<form>

URL:

<input type="url" name="url" />

</form>

URL:

Bitte geben Sie eine URL ein.

2\_h5\_formular.html



## Formularfeld: Zahl ("number")

- Validierungsmöglichkeiten:
  - Bereich (von ... bis) (**min**, **max**)

Zahl1:

Zahl2:

**<form>**

Zahl1:

**<input** **type="number"** **name="zahl1"/>**

Zahl2:

**<input** **type="number"** **min="10"** **max="99"**  
**name="zahl2"/>**

**</form>**

2\_h5\_formular.html

# Formularfeld: Bereich ("*range*")

- Per Slider: Einstellung eines Wertes
- Attribute:
  - `min`, `max`, `step`

`<form>`

Bereich:

```
10 <input type="range"  
min="10" max="100" step="2"  
name="range"/> 100
```

`</form>`



2\_h5\_formular.html

## Formularfeld: Bereich ("range")

`<form>`

Bereich:

```
10 <input type="range"  
    min="10" max="100" step="2"  
    value="10" name="range"/> 100
```

`</form>`

Bereich: 10  100

Anzeige des  
aktuellen  
Werts?

2\_h5\_formular.html


# Formularfeld: Bereich ("range")

JavaScript:

```
function rangeChange () {  
    var wert =  
        document.getElementById('range').value;  
    document.getElementById('hier').value = wert;  
}
```

HTML:

```
10 <input type="range"  
    min="10" max="100" step="2" value="10"  
    id="range" name="range"  
    onchange="rangeChange()" /> 100  
<input value="10" type="number" id="hier"  
    readonly />
```



Oninput anstelle von  
onchange updated die  
Eingabe stetig.

2\_h5\_formular.html

# Formularfeld: Text ("text")

Validierung mittels regulärem Ausdruck ("**pattern**")

```
<form>
```

Text:

```
<input type="text" pattern="[A-E] {3}"  
name="text1" />
```

```
</form>
```

Text: Hallo



Ihre Eingabe muss mit dem geforderten Format übereinstimmen.

Text: ABE

2\_h5\_formular.html

## Formularfeld: Suchfeld ("*search*")

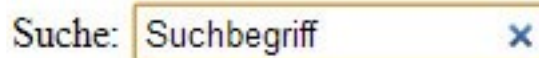
Ggf. anderes Aussehen als normales Textfeld

```
<form>
```

Suche :

```
<input type="search" value="Suchbegriff"  
name="suche" />
```

```
</form>
```



2\_h5\_formular.html

# Formularfeld

- Weitere Typen:

- `date` (Bsp. „2013-01-10“)
- `datetime`, `datetime-local` (Bsp. „2013-01-10T14:00“)
- `time` (Bsp. „14:00:01“)
- `week` (Bsp. „2013-W02“)
- `month` (Bsp. „2013-01“)
- `tel` (Bsp. „07361-5764365“)
- `color` (Bsp. „#FFDDFF“)

# *Medienelemente*



# Medien-Elemente

- Videodatei Tag: `<video>`
  - Attribute:
    - `controls` (boolean): Anzeige von Steuerelementen
    - `autoplay` (boolean): Datei soll automatisch abgespielt werden
    - `preload` (boolean): Gleich Laden oder erst beim Abspielen
    - `loop` (boolean): Wiederholung
    - `poster` (String): Bild (vor Abspielen des Videos)
    - `muted` (boolean): Video wird stumm abgespielt
    - `source` (Unterelement): Angabe der Quelle(n)  
(kann mehrfach auftreten, vgl. nächste Folien)

Boolean Attribute benötigen keine Wertzuweisung!

## Medien-Elemente: `<video>`

- Unterstützte Videotypen:

Format	Mimetype
MP4	video/mp4
WebM	video/webm
Ogg	video/ogg

## Medien-Elemente: `<video>`

```
<video width="996" height="560"
```

```
controls src="media/vornberger.mp4"  
type="video/mp4">
```

Ihr Browser unterstumt leider nicht  
das video-Tag.

Besorgen Sie sich schleunigst einen  
neuen!!!

```
</video>
```

## Medien-Elemente: `<video>`

```
<video width="996" height="560" controls  
  poster="media/platzhalter.jpg">  
  <source src="media/vornbergerger.mp4"  
    type="video/mp4">  
  ...  
</video>
```

4\_h5\_video\_poster.html

HMTL-Bildquelle (Platzhalter): <http://www.vag-customs-nrw.de/images/platzhalter.jpg>

## Medien-Elemente: `<video>`

```
<video width="996" height="560"  
      controls autoplay>  
  <source src="media/vornberger.mp4"  
        type="video/mp4">  
  <source src="media/vornberger.ogg"  
        type="video/ogg">  
  ...  
</video>
```

5\_h5\_video\_auto.html

## Medien-Elemente: `<video>`

```
<video width="996" height="560"  
      autoplay muted>  
  <source src="media/vornberger.mp4"  
        type="video/mp4">  
  ...  
</video>
```


6\_h5\_video\_muted.html

# Medien-Elemente: `<video>`

- Interessante Eigenschaften:
  - `volume`
  - `ended`
  - `pause`
  - `play`
  - `duration`
  - `currentTime`
- Beispiel:
  - Eigene Videosteuerung (Play/Pause)
  - Lautstärkensteuerung
  - Anzeige der verstrichenen Zeit

# Beispiel Videosteuerung

- GUI:
  - Play/Pause-Button
  - Regler für Lautstärke
  - Textfelder für laufende / Gesamtzeit



```
<form>
  <input type="button" onclick="play_pause()"
    class="play_pause" id="play" value="Play"/>
  <br/>
  Zeit:
  <input type="number" id="timer"> /
  <input type="number" id="duration"/>
  <br/>
  Lautstärke:
  <input id="loud" type="range" min="0" max="10" value="10",
    oninput="loudness()"/><br/>
  <input value="10" type="text" id="hier" readonly/>
</form>
```

7\_h5\_video\_interactive.html



# Beispiel Videosteuerung

Lautstärkeregelung:

```
function loudness () {  
    video = document.getElementById('video');  
    var volEl = document.getElementById('loud');  
    var volVal = volEl.value;  
    video.volume = volVal/10;  
    document.getElementById('hier').value = volVal;  
}
```

7\_h5\_video\_interactive.html

# Beispiel Videosteuerung

Start-Pause-Button:

```
function play_pause() {  
    var butEl = document.getElementById('play');  
    var butVal = butEl.value;  
    video = document.getElementById('video');  
    if (butVal == "Play") {  
        document.getElementById('play').value = "Pause";  
        video.play();  
        startCount();  
    } else {  
        document.getElementById('play').value = "Play";  
        video.pause();  
        pauseCount();  
    }  
}
```

7\_h5\_video\_interactive.html

# Beispiel Videosteuerung

Anzeige aktuelle Laufzeit:

```
var t;
function startCount() {
    var timeCurr = document.getElementById('timer');
    var timeMax = document.getElementById('duration');
    video = document.getElementById('video');
    timeMax.value = Math.round(video.duration);
    t = window.setInterval(function() {
        if (video.ended != true) {
            timeCurr.value = Math.round(video.currentTime);
        } else {
            window.clearInterval(t);
        }
    }, 1000); // jede Sekunde Ausführen
}

function pauseCount() {
    window.clearInterval(t);
}
```

7\_h5\_video\_interactive.html

# Medien-Elemente

- Audiodatei Tag: `<audio>`
  - Attribute:
    - `controls` (boolean): Anzeige von Steuerelementen
    - `autoplay` (boolean): Datei soll automatisch abgespielt werden
    - `preload` (boolean): Gleich Laden oder erst beim Abspielen
    - `loop` (boolean): Wiederholung
    - `source` (Unterelement): Angabe der Quelle(n)

# Medien-Elemente: *<audio>*

Unterstützte Audiotypen:

Format	Mimetype
MP3	audio/mp3
Wav	audio/wav
Ogg	audio/ogg

## Medien-Elemente: `<audio>`

```
<audio controls src="media/MHO-0009.mp3"  
autoplay loop>
```

Ihr Browser unterstutzt leider nicht  
das audio-Tag.

Besorgen Sie sich schleunigst einen  
neuen!!!

```
</audio>
```

8\_h5\_audio.html

# Medien-Elemente: `<audio>`

`<audio controls>`

`<source`

`src="media/MHO-0009.mp3"`  
`type="audio/mpeg">`

`<source`

`src="media/MHO-0009.ogg"`  
`type="audio/ogg">`

...

`</audio>`

9\_h5\_audio\_source.html

# *Canvas*



# Canvas

- Dynamische Erstellung von Bitmap-Grafiken (Zeichnen)
  - Einzige Attribute sind **width** und **height**
  - Nullpunkt des Koordinatensystems ist links oben
  - Textinhalt des Elements wird dargestellt wenn Erzeugung des Canvas fehlschlägt

```
<canvas id="ID" width="<integer>" height="<integer>">
```

```
  Fallback
```

```
</canvas>
```

# Canvas: Eigenschaften und Methoden

- Methode `getContext ("2d")` liefert Zeichenobjekt (2D) / auch "`webgl`" möglich (browserabhängig...)
- Kontextobjekt 2D besitzt Eigenschaften, z.B.
  - `fillStyle` (Füllfarbe)
  - `strokeStyle` (Stiftfarbe)
  - `lineWidth` (Stiftbreite)
- und Methoden, z.B.
  - `moveTo ()`
  - `lineTo ()`
  - `fillRect ()`
  - `arc ()`

# Canvas: Skript

- Zeichnen erfolgt über JavaScript
  - Entweder als eigene Funktion oder `<script>` innerhalb des Elements

```
<canvas id="myCanvas">
```

Ihr Browser unterstumt leider nicht...

```
</canvas
```

```
<script>
```

```
var cvsEl = document.getElementById("myCanvas");
```

```
var ctx = cvsEl.getContext("2d");
```

```
ctx.fillStyle = "#FF0000";
```

```
ctx.fillRect(0,0,80,100);
```

```
</script>
```

10\_h5\_canvas\_script.html

# Beispiel Canvas – „Hello World“

```
<canvas id="myCanvas">
```

```
...
```

```
</canvas>
```

```
<script>
```

```
var cvsEl = document.getElementById("myCanvas");
```

```
var ctx = cvsEl.getContext("2d");
```

```
ctx.font = "30px Arial";
```

```
ctx.fillText("Hello World", 10, 50);
```

```
</script>
```

11\_h5\_canvas\_hello\_world.html

# Canvas: Linie

- 10 Pixel breite rote Linie
  - Zum Schreiben muss **stroke()** aufgerufen werden.

```
<canvas id="myCanvas">...</canvas>
```

```
<script>
```

```
var cvsEl = document.getElementById("myCanvas");
```

```
var ctx = cvsEl.getContext("2d");
```

```
ctx.strokeStyle = "#FF0000";
```

```
ctx.lineWidth = "10";
```

```
ctx.moveTo(0,0);
```

```
ctx.lineTo(300,150);
```

```
ctx.stroke();
```

```
</script>
```

12\_h5\_canvas\_line.html

# Canvas: Kreis

- Kreis zeichnen mit `arc(x, y, r, start, stop)`
  - Zum Zeichnen muss `stroke()` oder `fill()` aufgerufen werden.

```
<canvas id="myCanvas">...</canvas>
```

```
var cvsEl = document.getElementById("myCanvas");
```

```
var ctx = cvsEl.getContext("2d");
```

```
ctx.beginPath();
```

```
ctx.arc(95, 50, 40, 0, 2*Math.PI);
```

```
ctx.stroke();
```

13\_h5\_canvas\_arc\_1.html

# Canvas: Kreis

- Style des Kreises wie schon beim Rechteck
  - Hier können `fillStyle` und `strokeStyle` aufgerufen werden

```
var cvsEl = document.getElementById("myCanvas");  
var ctx = cvsEl.getContext("2d");  
ctx.fillStyle = "#00FF00";  
ctx.strokeStyle = "#FF0000";  
ctx.lineWidth = "5";  
ctx.beginPath();  
ctx.arc(95, 50, 40, 0, 2*Math.PI);  
ctx.fill();  
ctx.stroke();
```

14\_h5\_canvas\_arc\_2.html

# Canvas: Kreis

- Teilkreise durch passende Pi-Multiplikation

```
var cvsEl = document.getElementById("myCanvas");  
var ctx = cvsEl.getContext("2d");
```

```
ctx.strokeStyle = "#FF0000";  
ctx.lineWidth = "5";  
ctx.beginPath();  
ctx.arc(95, 50, 40, 0, 1.5*Math.PI);  
ctx.stroke();
```

15\_h5\_canvas\_arc\_3.html



# Canvas: Linearer Farbverlauf

```
var cvsEl = document.getElementById("myCanvas");  
var ctx = cvsEl.getContext("2d");  
  
// Def. linearer Farbverlauf:  
var grd = ctx.createLinearGradient(0,0,200,0);  
grd.addColorStop(0,"red");  
grd.addColorStop(1,"white");  
  
ctx.fillStyle = grd;  
ctx.fillRect(10,10,150,80);
```

16\_h5\_canvas\_gradient\_lin.html

# Canvas: Radialer Farbverlauf

```
var cvsEl = document.getElementById("myCanvas");  
var ctx = cvsEl.getContext("2d");  
  
// Def. radialer Farbverlauf:  
var grd = ctx.createRadialGradient(75, 50, 5, 90, 60, 100);  
grd.addColorStop(0, "red");  
grd.addColorStop(1, "white");  
  
ctx.fillStyle = grd;  
ctx.fillRect(10, 10, 150, 80);
```

17\_h5\_canvas\_gradient\_rad.html

# Canvas: Einbinden von Bildern

- Canvas-Bilder aus externen Dateien laden:

```
function drawPicture() {  
    var cvsEl = document.getElementById("myCanvas");  
    var img = new Image();  
    img.onload = function() {  
        if(cvsEl.getContext) {  
            var ctx = cvsEl.getContext("2d");  
            ctx.drawImage(img, 0, 0, 208, 80);  
        }  
    }  
    img.src = "media/htw-logo-li-208x80.png";  
}
```

18\_h5\_canvas\_image.html

# Canvas: Transformieren von Bildern

- Transformieren mittels `translate()` und `rotate()`:

```
function drawPicture() {  
    var cvsEl = document.getElementById("myCanvas");  
    var img = new Image();  
    img.onload = function() {  
        if(cvsEl.getContext) {  
            var ctx = cvsEl.getContext("2d");  
            ctx.translate(200, 0);  
            ctx.rotate(90*Math.PI/180);  
            ctx.drawImage(img, 0, 0, 80, 208);  
        }  
    }  
    img.src = "media/htw-logo-li-208x80.png";  
}
```

18\_h5\_canvas\_image.html

# Canvas: Mausabfrage

- Abfrage Position (Ergebnis eines Ereignisses):
  - `clientX`, `clientY`
- Typische Abfrage:
  - `x = evt.clientX-cvsEl.offsetLeft;`
  - `y = evt.clientY-cvsEl.offsetTop;`
- Maustasten
  - Ereignisse:  
`cvsEl.onmousedown`  
`cvsEl.onmouseup`

# ***CSS3-Animation***

# CSS3-Animation

- Zweiteilig:
  - Animationsattribute in normaler CSS-Anweisung
    - `animation` [<animation-name> <animation-duration>]
    - `animation-name`
    - `animation-duration`
    - `animation-duration-count`
    - `animation-delay`
  - Keyframe-Definition in Regel (`@keyframes`)
    - z.B. Start- und Endwerte (`from`, `to`)
- Endzustand wird **nicht** beibehalten.

# CSS3-Animation

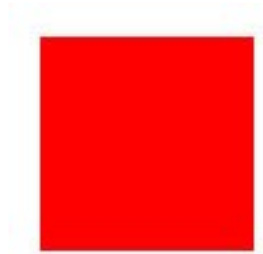
- Obacht: Je nach Browser wird evtl. Präfix benötigt
  - Google Chrome, Safari: **-webkit-**
  - Firefox: **-moz-**
  - Opera: **-o-**
  - Microsoft IE/Edge: **-ms-**

```
6 </script>
7 <style>
8 .quadrat {
9     float: left;
10    width: 80px;
11    height: 80px;
12    background: red;
13    margin-top: 10px;
14    position: relative;
15    -webkit-animation-name: animation02;
16    -webkit-animation-duration: 5s;
17    -webkit-animation-iteration-count: 4;
18 }
19
20 @-webkit-keyframes animation02 {
21     0% { background: red; }
22     33% { background: blue; }
23     66% { background: green; }
24     100% { background: yellow; }
25 }
26 </style>
27 </head>
28 </body>
```



# Beispiel CSS3-Animation (1)

```
.quadrat {  
    float: left;  
    width: 80px;  
    height: 80px;  
    background: red;  
    margin-top: 10px;  
    position: relative;  
    -webkit-animation-name: animation01;  
    -webkit-animation-duration: 5s;  
    -webkit-animation-iteration-count: 4;  
}
```



1\_css3\_animation\_1.html

## Beispiel CSS3-Animation (2)

```
@-webkit-keyframes animation01
{
    from { left: 0px; }
    to   { left: 100px; }
}
```

1\_css3\_animation\_1.html

# CSS3-Animation: Mehrere Werte

```
@-webkit-keyframes animation02 {  
    0%    {background: red;    }  
    33%   {background: blue;   }  
    66%   {background: green;  }  
    100%  {background: yellow; }  
}
```

2\_css3\_animation\_2.html

# CSS3-Animation: Start per JavaScript

- Animationsklasse erst bei Bedarf einbinden
- jQuery-Funktion **addClass**

```
$ (" #b1 " ) . click ( function ( ) {  
    $ (" #qAnim " ) . addClass ( " animationClass " ) ;  
} ) ;
```

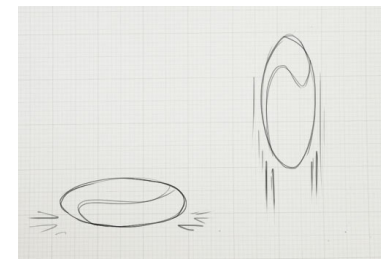
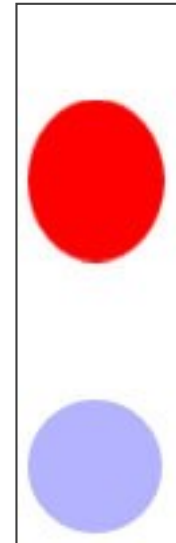
3\_css3\_add\_class.html

# ***CSS3: Prinzipien der Animation***

- CSS3 gut geeignet, die 12 Grundprinzipien der Animation umzusetzen
- Beispiele:
  - Squash & Stretch
  - Arcs / Ease-In & Ease-Out

# CSS3: Squash & Stretch

- Beispiel an Ball
- Transformation:
  - Ball fällt herunter
  - Ball springt hoch
- Verformung:
  - Skalierung in y-Richtung
    - Strecken (Vergrößerung y-Wert)
    - Stauchen (Verkleinerung y-Wert)



Bildquelle: <http://coding.smashingmagazine.com/2011/09/14/the-guide-to-css-animation-principles-and-examples/>

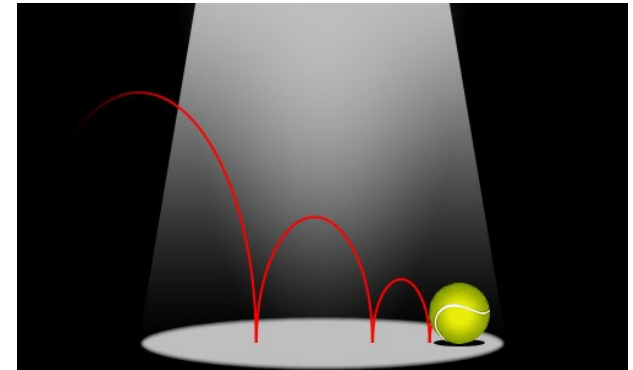
# CSS3: Squash & Stretch

```
@-webkit-keyframes circleAnimation {  
    0% { -webkit-transform: translateY(-200px) scaleY(1.2); }  
    33% { -webkit-transform: translateY(0px) scaleY(1.2); }  
    35% { -webkit-transform: translateY(10px) scaleY(0.8); }  
    66% { -webkit-transform: translateY(-50px) scaleY(1.2); }  
    100% { -webkit-transform: translateY(0px); }  
}
```

4\_css3\_squash\_stretch.html

# CSS3: Arcs, Ease In & Ease Out

- Zwei parallele Animationen:
  - Bewegung nach rechts
  - Hüpfende vertikale Bewegung
- Realisierung per CSS3
  - Zwei Klassen
  - Verschachteltes `div`



Bildquelle: <http://coding.smashingmagazine.com/2011/09/14/the-guide-to-css-animation-principles-and-examples/>

```
<div class="circle2">  
  <div class="circle1"></div>  
</div>
```

5\_css3\_arcs.html



# CSS3: Arcs, Ease In & Ease Out

```
.circle1 {
```

...

```
-webkit-animation: aniY 5.0s;  
}
```

```
.circle2 {
```

```
-webkit-animation: aniX 5.0s;  
}
```

5\_css3\_arcs.html

# CSS3: Arcs, Ease In & Ease Out

Horizontale Bewegung:

```
@-webkit-keyframes aniX {  
    0% { -webkit-transform: translateX(0px); }  
    100% { -webkit-transform: translateX(200px); }  
}
```

5\_css3\_arcs.html

# CSS3: Arcs, Ease In & Ease Out

- Vertikale Bewegung
  - Transformation in y-Richtung
  - *Ease In / Ease Out*
    - **animation-timing-function**
    - Werte:
      - **linear**: konstante Geschwindigkeit
      - **ease** (Default): Langsamer Start, langsames Ende
      - **ease-in**: Langsamer Start
      - **ease-out**: Langsames Ende
      - **cubic-bezier**(**n**,**n**,**n**,**n**): Selbst definierte Geschwindigkeit
- Beim Herunterfallen des Balls:
  - Langsam starten, schnell vom Boden abprallen

5\_css3\_arcs.html

# CSS3: Arcs, Ease In & Ease Out

```
@-webkit-keyframes aniY {  
  0% { -webkit-transform: translateY(-205px);  
        -webkit-animation-timing-function: ease-in; }  
  40% { -webkit-transform: translateY(-100px);  
        -webkit-animation-timing-function: ease-in; }  
  65% { -webkit-transform: translateY(-52px);  
        -webkit-animation-timing-function: ease-in; }  
  82% { -webkit-transform: translateY(-25px);  
        -webkit-animation-timing-function: ease-in; }  
  92% { -webkit-transform: translateY(-12px);  
        -webkit-animation-timing-function: ease-in; }  
  
  25%, 55%, 75%, 87%, 97%, 100% {  
    -webkit-transform: translateY(0px);  
    -webkit-animation-timing-function: ease-out; }  
}
```

5\_css3\_arcs.html

## ***Zwischen-Zusammenfassung***

- **SVG, HTML5, jQuery** und **CSS3** sind Open Source-Techniken, die auch Animationen unterstützen
  - SVG benötigt Plugin (meist im Browser integriert)
- Die Grundprinzipien der Animation können damit mit unterschiedlichsten Methoden umgesetzt werden

# *jQuery*

# *jQuery...*

- ... ist eine JavaScript-Bibliothek
- ... vereinfacht stark die JavaScript-Programmierung
- ... ist einfach zu lernen

(<http://www.w3schools.com/jquery/default.asp>)

- Wird von fast 80% aller Webseiten verwendet
- Man braucht kein Plugin auf Nutzerseite.
- Man braucht keinen speziell konfigurierten Webserver auf Entwicklerseite.



Quelle Verwendung: <https://w3techs.com/technologies/details/js-jquery>  
Bildquelle: [http://de.wikipedia.org/w/index.php?title=Datei:Logo\\_jQuery.svg&filetimestamp=20100719081003](http://de.wikipedia.org/w/index.php?title=Datei:Logo_jQuery.svg&filetimestamp=20100719081003)

# *jQuery*

- *jQuery*: JavaScript-Bibliothek für vereinfachte DOM-Manipulation
  - Element-Selektion (*Sizzle Selector Engine*), v.a. für CSS3-Selektoren
  - DOM-Manipulation
  - Erweitertes Event-System
  - Hilfsfunktionen, z.B. *each*-Funktion
  - Effekte und Animationen
  - Ajax-Funktionalität
  - Erweiterbarkeit durch zahlreiche Plugins, z.B. *jQuery UI*



Bildquelle: [http://de.wikipedia.org/w/index.php?title=Datei:Logo\\_jQuery.svg&filetimestamp=20100719081003](http://de.wikipedia.org/w/index.php?title=Datei:Logo_jQuery.svg&filetimestamp=20100719081003)



# Installation

- Download von (Local Source)
  - <http://jquery.com/download/>
  - Einbinden in HTML5-Code:  

```
<script src="jquery-3.6.1.min.js">  
</script>
```
- Ohne Installation (Online Source)
  - Microsoft Ajax Content Delivery Network  

```
<script src="https://ajax.aspnetcdn.com/ajax/jquery/jquery-3.6.0.min.js">  
</script>
```
  - Google Developers Hosted Libraries  

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js">  
</script>
```

# Beispiel jQuery

```
<script>
```

```
$ (document) .ready (function () {  
    $ ("p") .click (function () {  
        $ (this) .hide () ;  
    }) ;  
}) ;
```

```
</script>
```

```
<body>
```

```
<p>Click this to hide it.</p>
```

```
<p>Works for all other paragraphs like this</p>
```

```
<p>Here, too!</p>
```

```
</body>
```

1\_jquery\_click\_hide.html

# Syntax

- Ziel:
  - Auswahl von Elemente
  - Ausführung von Aktionen auf Elemente
- Basis-Syntax: `$ (selector) .action ()`
  - `$` : jQuery-Kennung
  - `(selector)` : Selektor zur Auswahl von Elemente
  - `action ()` : Ausführung einer jQuery-Aktion
- Beispiele:
  - `$(this).hide ()` : Versteckt das aktuelle Element
  - `$(p).hide ()` : Versteckt alle `<p>`-Elemente
  - `$(".test").hide ()` : Versteckt alle Elemente mit `class="test"`
  - `$("#test").hide ()` : Versteckt alle Elemente mit `id="test"`

# Syntax (Alternative)

- Basis-Syntax: `jQuery(selector).action()`
  - `jQuery` : jQuery-Kennung
  - `(selector)` : Selektor zur Auswahl von Elemente
  - `action()` : Ausführung einer jQuery-Aktion
- Beispiele:
  - `jQuery(this).hide()` : Versteckt das aktuelle Element
  - `jQuery(p).hide()` : Versteckt alle `<p>`-Elemente
  - `jQuery(".test").hide()` : Versteckt alle Elemente mit `class="test"`
  - `jQuery("#test").hide()` : Versteckt alle Elemente mit `id="test"`

2\_jquery\_click\_hide\_alt.html

# Selektoren

- Basieren auf CSS-Selektoren, mit einigen Erweiterungen
- `$ ("p")` : Wählt alle `<p>`-Elemente aus
- `$ ("#test")` : *ID*-Selektor
- `$ (".test")` : *class*-Selektor

# Selektoren

Auswahl:

Muster	Bedeutung
<b>*</b>	Jedes Element
<b>E</b>	Jedes <b>E</b> -Element
<b>E F</b>	<b>F</b> (Nachfolger von <b>E</b> )
<b>E&gt;F</b>	<b>F</b> (Kind von <b>E</b> )
<b>E:first-child</b>	Erstes Kind von <b>E</b>
<b>E+F</b>	<b>F</b> (1. Geschwister von <b>E</b> )
<b>E[foo]</b>	<b>E</b> , welches Attribut " <b>foo</b> " besitzt
<b>E[foo="value"]</b>	<b>E</b> , welches Attribut " <b>foo</b> " mit Wert " <b>value</b> " besitzt
<b>E#id</b>	<b>E</b> , mit <b>id="id"</b> besitzt

Quelle: <http://www.w3.org/TR/CSS2/selector.html>

# Beispiele Selektoren

Syntax	Bedeutung
<code>\$ ("*")</code>	Jedes Element
<code>\$ (this)</code>	Aktuelles HTML-Element
<code>\$ ("p.intro")</code>	Alle <code>&lt;p&gt;</code> Elemente mit <code>class="intro"</code>
<code>\$ ("p:first")</code>	Erstes <code>&lt;p&gt;</code> Element
<code>\$ ("ul li:first")</code>	Erstes <code>&lt;li&gt;</code> Element des ersten <code>&lt;ul&gt;</code>
<code>\$ ("ul li:first-child")</code>	Erstes <code>&lt;li&gt;</code> Element von jedem <code>&lt;ul&gt;</code>
<code>\$ (" [href] ")</code>	Jedes Element, welches Attribut <code>"href"</code> besitzt
<code>\$ ("a [target='_blank'] ")</code>	Jedes Element, welches Attribut <code>"target"</code> mit Wert <code>"_blank"</code> besitzt
<code>\$ ("a [target!='_blank'] ")</code>	Jedes Element, dessen Attribut <code>"target"</code> nicht Wert <code>"_blank"</code> besitzt
<code>\$ (":button")</code>	Alle <code>&lt;button&gt;</code> Elemente und <code>&lt;input&gt;</code> Elemente mit <code>type="button"</code>
<code>\$ ("tr:even")</code>	Alle geraden <code>&lt;tr&gt;</code> Elemente
<code>\$ ("tr:odd")</code>	Alle ungeraden <code>&lt;tr&gt;</code> Elemente

Bildquelle: [http://www.w3schools.com/jquery/jquery\\_selectors.asp](http://www.w3schools.com/jquery/jquery_selectors.asp)

# Document-ready-Event

- Sicherstellen, dass jQuery-Aktionen erst nach vollständigem Laden des Dokuments erfolgen:
  - `$(document).ready(function() { ... })`
- Kürzere Variante:
  - `$function() { ... }`
- Ansonsten Fehlermöglichkeiten (Beispiele):
  - Versuch, Element zu verstecken, bevor es vollständig geladen ist
  - Versuch, die Größe eines Bildes zu ermitteln, bevor es vollständig geladen ist



# Auslagerung von Funktionen

- Auslagerung von Funktionen
  - Bessere Übersicht
  - Wiederverwendung

Angabe mehrerer externer Dateien

```
<script src="jquery-3.6.1.min.js">
```

```
</script>
```

```
<script src="my_jquery_functions.js">
```

```
</script>
```

## Beispiel Vereinfachungen

```
var my_array=[1,2,3,4,5,6,7,8,9,10];
```

```
$(document).ready(function() {  
    $.each(my_array, function() {  
        document.write(this + " ");  
    });  
});
```

3\_jquery\_each.html

# Events

- Auslöser (Beispiele):
  - Mausbewegung (über ein Element)
  - Selektion eines *radio*-Buttons
  - Anklicken eines Elements
- Syntax:
  - `$ (selector) .event (function (...))`
- Beispiel:
  - `$ ("p") .click (function (...))`

# Events

Auswahl von Events:

Mouse Events	Keyboard Events	Form Events	Document / Window Events
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

# *Bedeutung einiger Events*

- **focus ()** :
  - Ein Formularelement erhält den Fokus
- **blur ()** :
  - Ein Formularfeld verliert den Fokus
- **click ()** :
  - Mausklick auf ein HTML-Element
- **dblclick ()** :
  - Doppelklick auf ein HTML-Element
- **mouseenter ()** :
  - Mauszeiger trifft auf ein HTML-Element
- **mouseleave ()** :
  - Mauszeiger verlässt ein HTML-Element

## *Bedeutung einiger Events (Forts.)*

- **mousedown ( )** :
  - Maustaste ist gedrückt, während sie auf dem HTML-Element ist
- **mouseup ( )** :
  - Maustaste wird losgelassen, während sie auf einem HTML-Element ist
- **hover ( )** :
  - Zwei Funktionen:
    - Maus berührt HTML-Element
    - Maus verlässt HTML-Element

## Beispiel Event – scroll-Event

```
$ (document) .ready (function () {  
    $ (document) .scroll (function () {  
        $ ("p") .hide () ;  
    }) ;  
}) ;
```

4\_jquery\_event\_scroll.html

# Beispiel Event – Maus

```
var first = 0;
$("#b1").mouseenter(function() {
    if (first == 0) {
        alert("Mouse over button!");
    }
});
$("#b1").mouseleave(function() {
    if (first == 0) {
        alert("Mouse left Button!");
        first = 1;
    }
});
$("#b1").dblclick(function() {
    if (first == 1) {
        $("#b1").hide();
    }
});
```

5\_jquery\_event\_mouse.html



## Beispiel Event – hover

```
$("#b1").hover(function() {  
    alert("Button hovered!");  
}, function() {  
    document.getElementById("b1").  
        value = "Mouse Moved";  
});
```

6\_jquery\_event\_hover.html

# Zeigen und Verstecken

- `$ (selector) .hide (speed, callback) ;`
- `$ (selector) .hide (speed, callback) ;`
- `$ (selector) .toggle (speed, callback) ;`
- Argumente (optional):
  - `speed` : Geschwindigkeit („slow“, „fast“, <Millisekunden>)
  - `callback` : Funktion, die nach Abarbeitung aufgerufen wird

# Beispiel Zeigen und Verstecken

```
var count = 0;
function increase() {
    count = count + 1;
    document.getElementById("b1").value
        = "Count: " + count;
}
$(document).ready(function() {
    $("#b2").click(function() {
        $("#b1").show("fast", increase());
    });
    $("#b2").dblclick(function() {
        $("#b1").hide(2000);
    });
});
```

7\_jquery\_event\_show\_hide.html

# *Fading (Ein-/Ausblenden)*

- Ein-/Ausblenden von Elementen:
  - `fadeIn()`
  - `fadeOut()`
  - `fadeToggle()`
  - `fadeTo()`
- Syntax:
  - `$(selector).fadeIn(speed, callback)`
  - `$(selector).fadeOut(speed, callback)`
  - `$(selector).fadeToggle(speed, callback)`
  - `$(selector).fadeTo(speed, opacity, callback)`

# Beispiel Fading (1)

```
$ ("#b1").click(function() {  
    $ ("#rect1").fadeIn();  
});  
$ ("#b1").dblclick(function() {  
    $ ("#rect1").fadeOut();  
});  
$ ("#b2").click(function() {  
    $ ("#rect2").fadeIn();  
    $ ("#rect3").fadeIn();  
});  
$ ("#b2").dblclick(function() {  
    $ ("#rect2").fadeOut(5000);  
    $ ("#rect3").fadeOut();  
});
```

8\_jquery\_event\_fade.html

## Beispiel Fading (2)

```
$("#b1").click(function() {  
    $("#rect1").fadeTo("fast", 0.5);  
});  
  
$("#b2").dblclick(function() {  
    $("#rect2").fadeToggle(5000);  
    $("#rect3").fadeToggle();  
});
```

9\_jquery\_event\_fade\_toggle.html

# Sliding

- Herein-/Heraus-“Schieben“ von Elementen
  - `slideDown()`
  - `slideUp()`
  - `slideToggle()`
- Häufig für Menüs verwendet (insbes. für mobile Geräte)
- Syntax:
  - `$(selector).slideDown(speed, callback)`
  - `$(selector).slideUp(speed, callback)`
  - `$(selector).slideToggle(speed, callback)`

## Beispiel Sliding (1)

```
$("#b1").click(function() {  
    $("#rect1").slideDown();  
});  
  
$("#b2").dblclick(function() {  
    $("#rect2").slideToggle(5000);  
    $("#rect3").slideToggle();  
});
```



## Beispiel Sliding (2)

```
<script>
$(document).ready(function() {
    $("#oben").click(function() {
        $("#menu").slideToggle();
    });
});
</script>
<style type="text/css">...</style>
<div id="oben">Click this to toggle open/close
    menu.</div>
<div id="menu">This is the opened menu.</div>
```

11\_jquery\_event\_slide\_css.html

# Animation

- Syntax:
  - `$(selector).animate({params}, speed, callback)`
    - *params*: CSS-Eigenschaften, die animiert werden sollen
      - `<Eigenschaft:Zielwert>`, in „{...}“, getrennt durch „“,“
      - Beispiel: `{left: '250px', height: '220'}`
      - Ausgehend vom aktuellen Wert
- Hinweis:
  - Start der Animation sofort nach Laden des Dokuments
  - Endzustand bleibt erhalten
  - Normalerweise ist die Position von HTML-Elementen statisch und kann nicht geändert werden
    - → CSS-Eigenschaft *position* muss explizit gesetzt werden!
      - `"relative"`, `"fixed"`, `"absolute"`

# Beispiel Animation

```
$("#b1").click(function() {  
    $("#div").animate({left: '250px',  
        height: '220'}, 2000);  
});
```

```
<div style="background: #98bf21;  
    height: 100px; width: 100px;  
    position: absolute;">
```

12\_jquery\_event\_animate.html

## Animation: Relative Werte

- Anstelle absoluter Zielwerte Angabe von Zuwachs/Verringerung
  - `"+="` bzw.
  - `"-="`
- Beispiel:
  - `{left: '+=100px', height: '-=50'}`

13\_jquery\_event\_animate\_relative.html

# Animation: Vordefinierte Werte

- Zielwerte können vordefinierte Werte sein:
  - "show", "hide", "toggle"
- Beispiel:
  - `$("#id1").animate({height: 'toggle'}, 2000);`

14\_jquery\_event\_animate\_predefined.html

# Animation: Sukzessives Ausführen

- *Queue Functionality*: Hintereinander-Ausführen von Animationen
- Beispiel:

```
var div = $("#id1");  
div.animate({height: '300px', opacity: '0.4'}, "slow");  
div.animate({width: '300px', opacity: '0.8'}, "slow");  
div.animate({height: '100px', opacity: '0.4'}, "slow");  
div.animate({width: '100px', opacity: '0.8'}, "slow");
```

15\_jquery\_event\_animate\_queue.html

# Animation: Stoppen

- Man kann laufende Animationen (einschließlich *Fading* und *Sliding*) stoppen.
- Syntax:
  - `$ (selector) .stop (stopAll, goToEnd)`
  - `stopAll` (optional): Die vollständige Animations-Queue wird gestoppt (Default: `false`)
  - `goToEnd` (optional): Angabe, ob die aktuell laufende Animation noch beendet wird (Default: `false`)
- Beispiel:
  - `$ (" #id1 ") .stop (true) ;`

15\_jquery\_event\_animate\_queue.html

# ***Callback-Funktionen***

- Zur Synchronisierung von (z.B.) Effekt-Funktionen
  - Aufruf der nächsten Funktion erst nach vollständiger Durchführung der vorangegangenen Funktion



## Beispiel Callback-Funktion

```
$("#b1").click(function() {  
    $("p").hide("slow", function() {  
        alert("Text is gone!");  
    });  
});  
  
// Without callback function  
$("#b2").click(function() {  
    $("p").hide("slow");  
    alert("Text is gone!");  
});
```

16\_jquery\_event\_callback.html

# Verkettung von Anweisungen

- *Chaining*: Anwendung von mehreren Anweisungen auf ein Element
- Hilfreich insbesondere bei komplexen Selektoren (müssen nur einmal ausgeführt werden)
- Verkettung durch Punkt-Operator (".")
- Beispiel:

```
$ ("p") .css ("color", "red") .  
      css ("font-size", "xx-large") .slideUp (2000) ;
```

17\_jquery\_event\_chain.html

# *DOM-Manipulation: Get Content*

- Inhalte:
  - **html** () : Liefert den Inhalt der ausgewählten Elemente (einschließlich HTML-Markups)
  - **text** () : Liefert den Textinhalt des Elements
  - **val** () : Liefert den Wert von Formularfeldern / Value-Attributen
  - **attr** () : Liefert den Wert von einem Attribut

## Beispiel text(), html()

```
$("#b1").click(function() {  
    alert("Text: " + $("#p").text());  
});
```

```
$("#b2").click(function() {  
    alert("HTML: " + $("#p").html());  
});
```

```
$("#b3").click(function() {  
    alert("HTML: " + $("#h1").html());  
});
```

18\_jquery\_js\_get\_content\_1.html

## Beispiel attr(), val()

```
$("#b1").click(function() {  
    alert("Button Value: " + $("#b1").val());  
});
```

```
$("#b2").click(function() {  
    alert("URL: " + $("p>a").attr("href"));  
});
```

# DOM-Manipulation: Setzen

- Zu setzender Inhalt als Argument
- Beispiel:

```
$("#b1").click(function() {  
    $("#p").text($("#input1").val());  
});  
$("#b2").click(function() {  
    $("#p").html("<b>" + $("#input1").val() + "</b>");  
});  
$("#b3").click(function() {  
    $("#input1").val($("#p").text());  
});
```

20\_jquery\_js\_set\_content.html

# DOM-Manipulation: Callback

- Anstelle von Werten können auch Callback-Funktionen verwendet werden

(bei `html()`, `text()`, `val()`)

– Syntax:

```
$(selector).text(function(index, alterWert) {  
    ...return ...  
});
```

– Analog für `html()`, `val()`

```
$(selector).attr("attr", function(index, alterWert) {  
    ... return ...  
});
```

# Beispiel Callback bei DOM-Manipulation

```
var s;  
$("#b1").click(function() {  
    $("#p").html(function(index, oldValue) {  
        s = oldValue;  
        return $("#input1").val() + " for Index " + index;  
    });  
});  
$("#b2").click(function() {  
    $("#p").html(s);  
});  
$("#b3").click(function() {  
    $("p>a").attr("href", function(index, oldValue) {  
        return oldValue + "/index.jsp";  
    });  
});
```

21\_jquery\_js\_content\_callback.html



# *DOM-Manipulation: Elemente hinzufügen*

- Methoden:
  - **append()**
    - Einfügen am Ende des selektierten Elements
  - **prepend()**
    - Einfügen am Beginn des selektierten Elements
  - **after()**
    - Einfügen nach dem selektierten Element
  - **before()**
    - Einfügen vor dem selektierten Element
- Anmerkungen:
  - Jeweils beliebig viele Argumente erlaubt
  - Argument kann Text oder HTML sein

# Einfügemöglichkeiten

- HTML:

```
s = "<p>Text</p>" ;
```

- jQuery:

```
s = $( "<p></p>" ) .text ( "Text" ) ;
```

- DOM:

```
s = document.createElement ( "p" ) ;  
s.innerHTML = "Text" ;
```

# Beispiel append

```
var s;  
  
$("#b1").click(function() {  
    $("#p1").append($("#i1").val());  
});  
  
$("#b2").click(function() {  
    $("#p1").append($("#<p></p>").text($("#i1").val()));  
});  
  
$("#b3").click(function() {  
    s = document.createElement("p");  
    s.innerHTML = $("#i1").val();  
    $("#p1").append(s);  
});
```

22\_jquery\_js\_append.html

# *DOM-Manipulation: Elemente entfernen*

- Löschen von Elementen:
  - **remove()**: Löscht das Element (samt Kind-Elemente)
    - Kann eingeschränkt werden
      - Z.B. **remove(".format")**
  - **empty()**: Löscht die Kind-Elemente des selektierten Elements

# Beispiel Entfernung von Elementen

```
$("#b1").click(function() {  
    $("#p1").remove();  
});
```

```
$("#b2").click(function() {  
    $("#br").remove(".format");  
    $("#b").remove(".format");  
});
```

```
$("#b3").click(function() {  
    $($("#i1").val()).remove();  
});
```

23\_jquery\_js\_remove.html

# CSS-class-Bearbeitung

- Funktionen zur Bearbeitung von CSS-Klassen:
  - **addClass()** : Eine/mehrere Klassen dem selektierten Element hinzufügen
  - **removeClass()** : Eine/mehrere Klassen von dem selektierten Element entfernen
  - **toggleClass()** : Wechsel zwischen Hinzufügen/Entfernen von Klassen des selektierten Elements
  - **css()** : Setzen oder Auslesen von Style-Attributen

# Beispiel CSS-Bearbeitung (Style)

```
.important {  
    font-weight:bold;  
    font-size:xx-large;  
    color:red;  
}  
  
.blue {  
    color:blue;  
}
```

# Beispiel CSS-Bearbeitung

```
$("#b1").click(function() {  
    $("b").addClass("blue");  
    $("h1,p").addClass("important");  
});
```

```
$("#b2").click(function() {  
    $("p").removeClass("important");  
});
```

```
$("#b3").click(function() {  
    $("p").toggleClass("important");  
});
```

24\_jquery\_css\_1.html



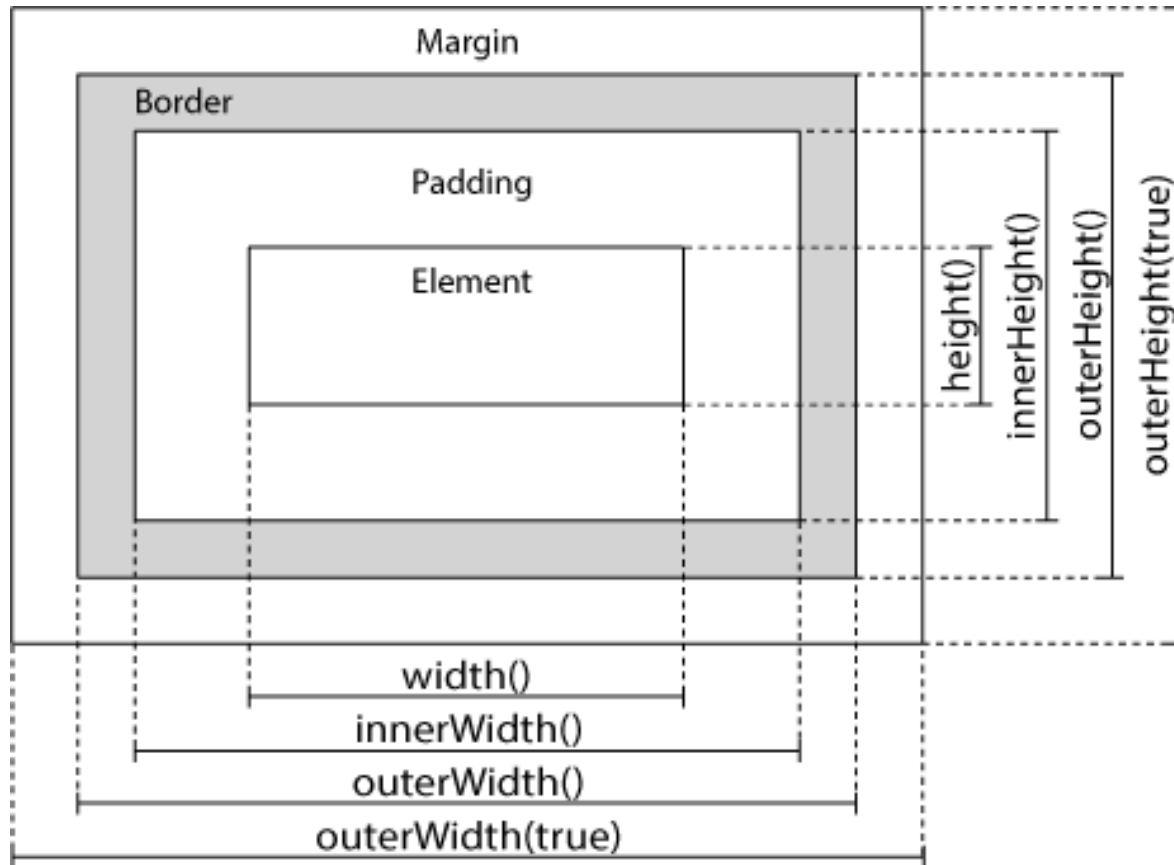
# CSS Attribute mit `css()`

- Lesen / Setzen von einer oder mehreren CSS-Eigenschaften
  - `css("Eigenschaftename")`
    - Auslesen einer Style-Eigenschaft
  - `css("Eigenschaftename", "Wert")`
    - Setzen einer Style-Eigenschaft
  - `css({ "Eigenschaftename": "Wert", "Eigenschaftename", "Wert", ... })`
    - Setzen mehrerer Style-Eigenschaften
- Beispiel:

```
$("h1").css({ "color": "green",  
               "text-decoration": "underline" });
```

25\_jquery\_css\_2.html

# Dimensionen



Bildquelle: [http://www.w3schools.com/jquery/jquery\\_dimensions.asp](http://www.w3schools.com/jquery/jquery_dimensions.asp)

# *Dimensionen: Funktionen*

- `width()`
- `height()`
- `innerWidth()`
- `innerHeight()`
- `outerWidth()` , `outerWidth(true)`
- `outerHeight()` , `outerHeight(true)`

26\_jquery\_dimensions.html

# *jQuery und AJAX*

- Geht auch
  - Siehe z.B.  
[http://www.w3schools.com/jquery/jquery\\_ajax\\_intro.asp](http://www.w3schools.com/jquery/jquery_ajax_intro.asp)
  - <http://www.youtube.com/watch?v=2SkzIzkuEQQ>  
(Evtl. etwas veraltet)

# *jQuery Mobile*

- Erweiterung von jQuery
  - Seiten werden per AJAX geladen
  - Versehen mit Übergangseffekten
  - Spezialisiert auf Entwicklung responsiver (Anpassung UI auf Endgerät) Applikationen
  - Elemente werden teilweise ausgeblendet
    - Z.B. direkt nutzbare Filterfunktionen für Listen
  - CSS-Vorlagen

# *jQuery Mobile: Grundaufbau*

```
<!DOCTYPE html>
<html>
<head>
  <title>Seitentitel</title>
  <meta name="viewport" content="width=device-width,initial-scale=1"/>
  <link rel="stylesheet" href="http://code.jquery.com/mobile/1.0.1/
    jquery.mobile-1.0.1.min.css"/>
  <script src="http://code.jquery.com/jquery-1.6.4.min.js">
  </script>
  <script src="http://code.jquery.com/mobile/1.0.1/
    jquery.mobile-1.0.1.min.js">
  </script>
</head>
<body>
  Inhalte
</body>
</html>
```

# *jQuery Mobile*

- Unterscheidung zwischen
  - Einseitentyp
  - Mehrseitentyp
  - Dialogtyp

# Einseitentyp

```
<div id="homePage" data-role="page">
```

...

```
</div>
```



# Einseitentyp

```
<div data-role="page">
  <div data-role="header">
    <h1>Seitentitel</h1>
  </div> <!-- /header -->
  <div data-role="content">
    <p>Eigentlicher Inhalt</p>
  </div> <!-- /content -->
  <div data-role="footer" data-position="fixed">
    <h4>Fussbereich</h4>
  </div> <!-- /footer -->
</div> <!-- /page -->
```

# Mehrseitentyp

```
<div id="homePage" data-role="page"  
    data-theme="a" class="amse-bkgnd">
```

...

```
</div>
```

```
<div id="aboutPage" data-role="page"  
    data-theme="b">
```

...

```
</div>
```

# Mehrseitentyp

Mehrere Seiten

Aber: nur eine Seite wird angezeigt

```
<div id="homePage" data-role="page"  
      data-theme="a" class="amse-bkgnd">
```

...

```
</div>
```

```
<div id="aboutPage" data-role="page"  
      data-theme="b">
```

...

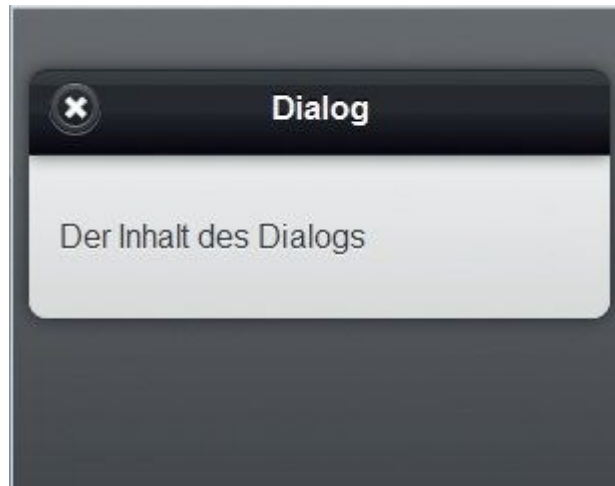
```
</div>
```

# Dialogtyp

```
<div data-role="dialog" id="dialog">
```

...

```
</div>
```



Bildquelle: <http://img1.pc-magazin.de/fQuery-Dialogfenster-r960x752-C-dcbcb277-53834128.jpg>

# Initialisierung

- jQuery
  - `ready(function(event) {...})`
- jQueryMobile
  - `event("pagecreate", function(event) {...})`
    - (ehem. "pageinit" bis 1.4)

```
<div id="homePage" data-role="page">  
  <script type="text/javascript">  
    $("#homePage").on("pagecreate",  
                      function() {...}) ;  
  </script>  
  ...  
</div>
```