



Grundlagen (X)HTML + CSS

JavaScript

Hanjo Müller (hanjo.mueller@bunix.de)
<http://htmlkurs.bunix.de>

- Überblick
- Grundlagen (X)HTML 1.0
- Grundlagen Cascading Stylesheets
- Grundlagen XML
- **Grundlagen Javascript**
- Webtechnologien



JavaScript – Ziele der Lektion

- Die Idee von JavaScript
- Integration von JavaScript in HTML Seiten
- Syntax und Kontrollstrukturen
- JavaScript und das Document Object Model
- Mouseovereffekte für Bilder und Texte
- Inhaltsprüfung von Formularen
- Ausblick auf das JavaScript von Heute

JavaScript – Integration in HTML (1)

- direkt über ein **script** Element

```
<script type="text/javascript">  
  alert( „Hallo Welt“ );  
</script>
```

- aus externer Datei über **script** Element

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

- direkt in einem Eventhandler

```
<input type="button" onclick="alert( 'Hallo Welt' );" />
```

- JavaScript Code wird immer dann ausgeführt wenn er im Quellcode auftaucht
- Ausnahme: Funktionen werden erst beim Aufruf ausgeführt
- Aufruf von Funktionen kann im JS Quelltext passieren oder auch direkt im Eventhandler

- Gängige und gültige Eventhandler werden von allen Browsern unterstützt
 - **onclick** – beim klicken mit der Maus
 - **onmouseover** – beim draufschieben der Maus
 - **onmouseout** – beim wegschieben der Maus
 - **onsubmit** – beim Absenden des Formulars
 - **onload** – beim Laden des Elementes
 - **onunload** – nach dem Parsen des Elementes

- objektorientierte, clientseitige Skriptsprache
- Syntax an Java / C++ orientiert
- Variablen müssen nicht deklariert werden
- Kein festes Typenmodell
- Strings in „“ oder " einschließen
- Variablenbezeichner sind Case-Sensitiv
- Jeder Befehl wird mit ; beendet

```
Zeichenkette = „Hallo Welt“;  
alert( Zeichenkette );
```

- Bedingungsprüfung mit **if** und **else**

```
if( wert1 == wert2 ) {  
    alert( 'werte sind gleich' );  
}  
else {  
    alert( 'werte sind verschieden' );  
}
```

- Mehrfachauswahlen mit **switch**

```
switch( wert ) {  
    case '1':  
        alert( 'Wert ist Eins' );  
        break;  
    case '2':  
        alert( 'Wert ist Zwei' );  
        break;  
}
```

- Zählschleife mit **for**

```
for( i = 1; i < 10; i++ ) {  
    alert( 'Die Zählvariable ist jetzt: ' + i + '.' );  
}
```

- Kopfgesteuerte Schleife mit **while**

```
i = 1  
while( i < 1 ) {  
    alert( 'Die Zählvariable ist jetzt: ' + i + '.' );  
    i++;  
}
```

- Fußgesteuerte Schleife mit **do ... while**

```
i = 1  
do {  
    alert( 'Die Zählvariable ist jetzt: ' + i + '.' );  
    i++;  
} while( i < 1 );
```

- **function** gefolgt von Name und Parameterliste in **()** sowie dem Funktionsrumpf in **{ }**
- Aufruf Name und Parameterliste in **()**
- Rückgabewerte mit **return**

```
function Ausgabe( text )  
{  
  if( text.length != 0 ) {  
    alert( text );  
    return true;  
  }  
  else {  
    return false;  
  }  
}
```

```
Ausgabe( „Hallo Welt“ );
```

- Zugriff erfolgt heute über das Document Object Model
- alle Elemente eines Dokuments werden im **document** Objekt als Felder verwaltet

```
<form onsubmit="alert( document.forms[0].elements[0].value );">  
  <input type="text" name="pflichtfeld" id="pflichtfeld" />  
</form>
```

- Zugriff auf Elemente eines bestimmten Typs über **getElementsByTagName('tagname')**

JavaScript – Document Object Model (2)

```
<html>
  <head>
    <script type="text/javascript">
      function showParagraphs()
      {
        var ps = document.getElementsByTagName( 'p' );

        for( var i=0; i < ps.length; i++ )
        {
          alert( ps[i].innerHTML );
        }
      }
    </script>
  </head>

  <body onload="showParagraphs();">
    <p>Das ist der erste Absatz</p>
    <p>Das ist der zweite Absatz</p>
    <p>Das ist der dritte Absatz</p>
  </body>
</html>
```

- direkter Zugriff über Element ID

```
<script type="text/javascript">  
  alert( document.getElementById( 'pflichtfeld' ).value );  
</script>
```

```
<form onsubmit="tollefunktion();">  
  <input type="text" name="pflichtfeld" id="pflichtfeld" />  
</form>
```

- Zugriff auf Werte von Formularelementen über **value** Attribut
- Zugriff auf Daten innerhalb der Tags über **innerHTML** Attribut

JavaScript – Mouseover Effekte bei Bildern

- Eventhandler **onmouseover** und **onmouseout**
- Zugriff auf Quelle eines Bildes über **src** Attribut

```
<html>
  <head>
    <title>Mouseover Effekt</title>
  </head>

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

JavaScript – Mouseover Effekte mit CSS

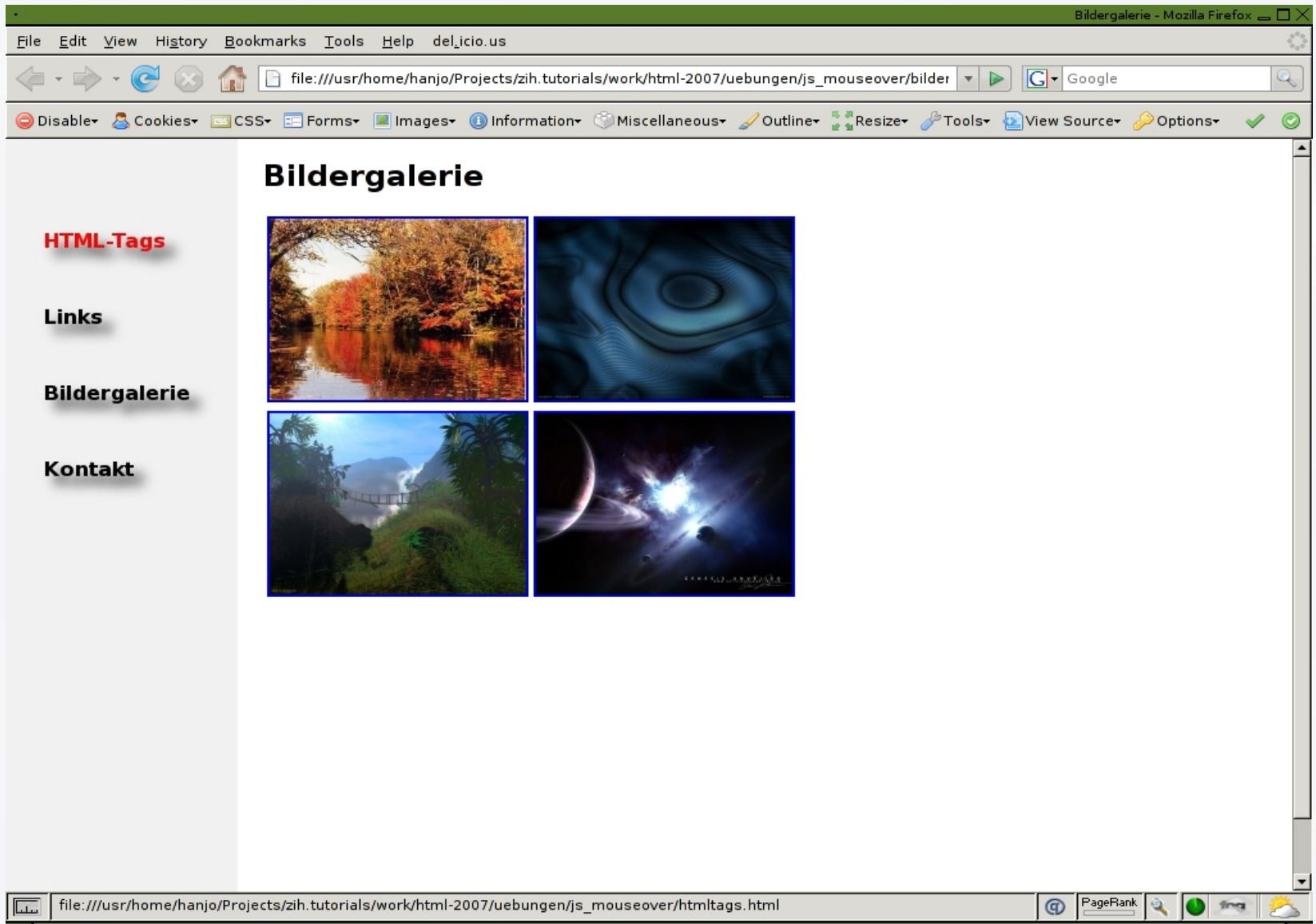
- Eventhandler **onmouseover** und **onmouseout**
- Zugriff auf CSS mit über **style** Objekt
- Einzelne Styles werden per CSS angesprochen

```
<html>
  <head>
    <title>CSS Mouseover</title>
  </head>

  <body>
    <p onmouseover="this.style.color='red';"
      onmouseout="this.style.color='black';">Ich werde Rot</p>
    <p onmouseover="this.style.color='green';"
      onmouseout="this.style.color='black';">Ich werde Grün</p>
    <p onmouseover="this.style.color='blue';"
      onmouseout="this.style.color='black';">Ich werde Blau</p>
  </body>
</html>
```

- Tauschen Sie die Textlinks im Menü der CSS Übung durch die bereitstehenden Grafiken aus.
- Erstellen Sie mit Hilfe der durch „_a“ gekennzeichneten Grafiken Mouseover Effekte.
- *Für ganz schnelle:* Automatisieren Sie den Vorgang mit einer Funktion.

JavaScript – Übung Mouseover Effekte



- Zugriff auf Formulare erfolgt entweder über **forms** Array oder bevorzugt über **DOM**
- Das versenden des Formulars kann mit dem **onsubmit** Eventhandler abgefangen werden
- Liefert die Prüffunktion **false** wird der Transfer unterbrochen
- Man sollte sich nie auf eine **JS** Prüfung der Daten verlassen!

JavaScript – Inhaltsprüfung von Formularen (2)

```
<html>
  <head>
    <title>Checkform</title>
    <script type="text/javascript">
      function checkform()
      {
        if( document.forms[0].elements[0].value == '' )
        {
          alert( 'Bitte das Pflichtfeld ausfüllen' );
          return false;
        }
      }
    </script>
  </head>
  <body>
    <form action="success.html" onsubmit="return checkform();">
      Pflichtfeld: <input name="pflichtfeld" /><br />
      Optionalfeld: <input name="optionalfeld" /><br />
      <input type="submit" />
    </form>
  </body>
</html>
```



JavaScript – Inhaltsprüfung von Formularen (3)

```
<html>
  <head>
    <title>Checkform</title>
    <script type="text/javascript">
      function checkform()
      {
        if( document.getElementById( 'pflichtfeld' ).value == '' )
        {
          alert( 'Bitte das Pflichtfeld ausfüllen' );
          return false;
        }
      }
    </script>
  </head>
  <body>
    <form action="success.html" onsubmit="return checkform();">
      Pflichtfeld: <input name="pflichtfeld" id="pflichtfeld" /><br />
      Optionalfeld: <input name="optionalfeld" /><br />
      <input type="submit" />
    </form>
  </body>
</html>
```

- Versehen Sie das Kontaktformular aus der HTML Übung mit einer Inhaltsprüfung.
- Die folgenden Daten muss der Nutzer zwingend eingeben:
 - E-Mail
 - Nachricht
- Nutzen Sie zur Überprüfung bevorzugt das DOM

- JS ist in allen gängigen Grafikbrowsern integriert
- Google arbeitet an einer JS-Engine für den Crawler
- AJAX geht gar nicht ohne JS



- AJAX ist ein modernes Konzept in der Web-Programmierung
- **A**synchronous **J**avaScript **a**nd **X**ML
- Nutzt das XMLHttpRequest-Objekt das in nahezu allen JS Implementierungen zu finden ist
- Statt die ganze Seite bei einer Nutzerinteraktion neu zu laden wird per DOM die Seite manipuliert
- Die serverseitige Sprache ist egal (PHP / JSP / Perl)