



JAVASCRIPT 04



JQUERY

EINFÜHRUNG

- T jQuery => JavaScript Bibliothek, die viele praktische Funktionen anbietet
<https://trends.builtwith.com/javascript/jquery>
- T Biete einen einfachen Zugriff auf die einzelnen DOM-Elemente
- T Ansprechende Effekte mit einfachen Mitteln
- T jQuery = reines JavaScript
- T Vorgefertigte Funktionen reduzieren Aufwand für das Erstellen von Java-Script Programmen deutlich
- T Muss eingebunden werden (entweder Download oder CDN Link)

SELEKTOREN: HTML-ELEMENTE ÜBER JQUERY ANSTEUERN

```
<body>
<h1 id="headlineElement"></h1>
<p id="pElement"></p>
<!-- jQuery muss vor dem script-Tag eingebunden werden -->
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
    /* um ein Element anzusteuern muss lediglich das Dollarzeichen vorangestellt werden
    * mit der anschließenden ID des Elements inkl. Rautezeichen */
    /* Die Index-Nummer wird benötigt da jQuery ein Array zurückgibt */
    $('#headlineElement')[0].innerHTML = 'Überschrift';
    /* bei .text wird kein Array zurück gegeben, daher kann man sich den
    * Index hier sparen */
    $('#pElement').text('Hier steht ein Absatz.');
```

```
</script>
</body>
```

BEISPIEL: AUSGABE INHALTE ÜBER JQUERY

T Der erste Befehl gilt für alle p-Tags, der zweite hingegen nur für das dritte Element im Dokument

```
<body>
<p></p>
<p></p>
<p></p>
<p></p>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
    $('p').text('Hier steht ein Absatz');
    $("p")[2].innerHTML = "Der dritte Absatz wird gezielt angesteuert.";
</script>
</body>
```

BEISPIEL: DARSTELLUNG DER ROTEN ABSÄTZE

```
<body>
<p></p>
<p class="rot"></p>
<p></p>
<p class="rot"></p>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
  /* Alle p-Tags erhalten einen Text */
  $('p').text("Hier steht ein Absatz");
  /* Array erzeugen mit allen Elementen mit der Klasse rot */
  let rot = $('.rot');
  /* jedes Element im Array durch gehen und style ändern */
  for (let i = 0; i < rot.length; i++) {
    rot[i].style.color = "red";
  }
  /* p-Tags mit der Klasse rot erhalten einen anderen Inhalt */
  $('.rot').text("Diese Absätze erscheinen in roter Farbe");
</script>
</body>
```

BEISPIEL: VERSCHIEDENE ELEMENTE PRÄZISE ANSTEUERN

```
<body>
<h1 class="rot"></h1>
<p></p>
<p class="rot"></p>
<p></p>
<p class="rot"></p>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
    $('p').text("Hier steht ein Absatz");
    let rot = $('.rot');
    for (let i = 0; i < rot.length; i++) {
        rot[i].style.color = "red";
    }
    $('p.rot').text("Diese Absätze erscheinen in roter Farbe.");
    $('h1.rot').text("Headline in roter Farbe");
</script>
</body>
```

INHALTE DER SEITE MIT JQUERY VERÄNDERN UND AUSWERTEN

T Inhalte werden über drei verschiedene Methoden ausgegeben

```
<body>
<h1></h1>
<p></p>
<input id="eingabe">
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
    /* .text() erlaubt nur reinen Text */
    $('h1').text('Überschrift');
    /* .html() erlaubt auch html-Tags */
    $('p').html("Absatz mit einem <strong>fett</strong> gedruckten Wort");
    /* .val() erlaubt das value-Attribute vorzugeben */
    $('#eingabe').val("Eingabefeld");
</script>
</body>
```


INHALTE DER SEITE MIT JQUERY VERÄNDERN UND AUSWERTEN

T Methoden nehmen die Inhalte der übrigen Elemente auf

```
<body>
<p>
  Absatz mit einem <strong>fett</strong> gedruckten Wort
</p>
<input id="eingabe" value="Eingabefeld Beispiel">
<input id="ausgabe1" size=60>
<input id="ausgabe2" size=60>
<input id="ausgabe3" size=60>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
  $('#ausgabe1').val($('p').text());
  $('#ausgabe2').val($('p').html());
  $('#ausgabe3').val($('#eingabe').val());
</script>
</body>
```

INHALTE DER SEITE MIT JQUERY VERÄNDERN UND AUSWERTEN

T Verändern der Größe des input-Tags

```
<body>
<input id="eingabe" value="Eingabefeld Beispiel">
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
    $('#eingabe').attr('size', 100);
    alert('Länge: ' + $("#eingabe").attr("size"));
</script>
</body>
```

INHALTE DER SEITE MIT JQUERY VERÄNDERN UND AUSWERTEN

T Text wird je nach verwendeter Funktion am Anfang oder am Ende eingefügt

```
<body>
<p id="1">Absatz</p>
<p id="2">Absatz</p>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
    $('#1').append(" * Am Ende eingefügter Text *");
    $('#2').prepend(" * Am Anfang eingefügter Text *");
</script>
</body>
```

INHALTE DER SEITE MIT JQUERY VERÄNDERN UND AUSWERTEN

T Mit den Methoden `before()` und `after()` kann Inhalte vor oder nach einem bestimmten Element eingefügt werden.

```
<body>
<p id="absatz">ursprünglicher Absatz</p>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
$('p').after("<p>nach dem ursprünglich eingesetzten Inhalt ein neues Element</p>");
$('#absatz').before("<h1>Vor dem ursprünglichen Inhalt eingefügtes Element.</h1>");
</script>
</body>
```

EVENTS MIT JQUERY BEARBEITEN

T jQuery bietet für alle gängigen Events eine Methode an, die genau gleich lautet wie dessen Bezeichnung.

```
<body>
<p>Absatz</p>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
function funktion1() {
    alert("Mouseout");
}
function funktion2() {
    alert("Mousedown");
}
$('p').onmouseout(funktion1);
$('p').onmousedown(funktion2);
</script>
</body>
```

EVENTS MIT JQUERY BEARBEITEN

- T Weitere wichtige Methode: .on()
- T Möglichkeit, mehrere Events mit einem einzigen Element zu verbinden

```
<body>
<p>Absatz</p>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
function funktion1() {
    alert("Mouseout");
}
function funktion2() {
    alert("Mousedown");
}
$('p').on({mouseout: funktion1, mousedown: funktion2});
</script>
</body>
```

SPEZIELLE EFFEKT MIT JQUERY EINFÜGEN

T Einfache Beispiele Methoden:

show() => zeigt ein Element an, das bislang versteckt war

hide() => versteckt ein Element

toggle() => verbindet show() und hide() in einem

T Alle drei Funktionen erlauben Übergabewerte für Geschwindigkeit

T Ähnliche Effekte bei Methoden: fadeIn(), fadeOut(), fadeToggle(), fadeTo()

Mit langsamen Übergang ("slow"): Größe bleibt konstant, Transparenz ändert sich

T Weitere Effekte: slideDown(), slideUp(), slideToggle()

Element in vertikaler Richtung aus-/eingefahren

```
<body>
<button id="btn1" onclick="hide()" type="button">Verstecken</button>
<button id="btn2" onclick="show()" type="button">Zeigen</button>
<button id="btn3" onclick="toggle()" type="button">Toggle</button>
<div id="rechteck" style="background: #ee9900; height: 200px; width: 200px;"></div>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
function hide() {
    $('#rechteck').hide();
}
function show() {
    $('#rechteck').show("fast");
}
function toggle() {
    $('#rechteck').toggle("slow");
}
</script>
</body>
```

SPEZIELLE EFFEKT MIT JQUERY EINFÜGEN

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Übung</title>
  <style>
    #rechteck {
      background: #ee9900; height: 200px; width: 200px;
      font-size: 100px; position: absolute;
      /* damit entsprechendes Element seine Position
      auf der Seite verändern kann, position
      absolute, relative oder fixed
      notwendig. */
    }
  </style>
</head>
<body>
<button id="btn1" onclick="animation()"
type="button">Animation</button>
<div id="rechteck">Text</div>
<script src="https://code.jquery.com/jquery-
3.5.1.min.js"></script>
```

```
<script>
function animation() {
  $('#rechteck').animate({
    left: '100px',
    top: '200px',
    width: '500px',
    height: '50px',
    fontSize: '20px'
  }, 3000); /* 3000 Millisekunden */
}
</script>
</body>
</html>
```


AUFGABE

1. Gestalte eine Seite mit einer Überschrift, drei Absätzen, einem Eingabefeld und einem Button. Füge zunächst einen passenden Text für die Überschrift ein. Danach sollen alle drei Absätze den gleichen Inhalt erhalten. Wenn der Anwender auf den Button drückt, soll der zweite Absatz den Inhalt des Eingabefelds erhalten. Dabei soll es auch möglich sein, HTML-Tags zu verwenden. Nutze für all diese Aufgaben jQuery.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Übung</title>
</head>
<body>
<h1></h1>
<p></p>
<p id="absatz2"></p>
<p></p>
<input id="eingabe">
<button id="btn" type="button">Inhalt austauschen</button>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
function austauschen() {
  $('#absatz2').html($('#eingabe').val());
}
$('#h1').text("Überschrift");
$('#p').text('Absatz');
$('#btn').click(austauschen);
</script>
</body>
</html>
```

AUFGABE

1. Gestalte ein farbiges div-Element. Dieses soll auf drei verschiedene Events (mouseout, dblclick und contextmenu) reagieren und dabei jeweils die Art des Events per alert-Befehl ausgeben. Verwende dafür nur eine einzige Methode.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Übung</title>
  <style>
    div {
      background-color: #669966;
      height: 50px;
      width: 50px;
    }
  </style>
</head>
<body>
<div></div>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
```

```
<script>
function funktion1() {
  alert('mouseout');
}
function funktion2() {
  alert('dblclick');
}
function funktion3() {
  alert('contextmenü');
}
$('div').on({
  mouseout: funktion1,
  dblclick: funktion2,
  contextmenu: funktion3
});
</script>
</body>
</html>
```

AUFGABE

1. Gestalte eine Seite mit zwei div-Elementen mit identischer Größe aber unterschiedlichen Farben. Wenn die Seite aufgerufen wird, soll jedoch eines der beiden Elemente mit der `hide()`-Methode (ohne Verzögerung) versteckt werden. Füge darunter einen Button ein. Wenn der Anwender diesen anklickt, soll das Programm die beiden Rechtecke langsam austauschen, indem es auf beide die `toggle()`-Methode anwendet.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Übung</title>
  <style>
    #rechteck1 {
      background-color: #ee9900;
      height: 100px;
      width: 200px;
    }
    #rechteck2 {
      background-color: #ee6600;
      height: 100px;
      width: 200px;
    }
  </style>
</head>
<body>
<div id="rechteck1">Textbox 1</div>
<div id="rechteck2">Textbox 2</div>
<button id="btn" type="button">Elemente tauschen</button>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>

<script>
function tauschen() {
  $('#rechteck1').toggle(2000);
  $('#rechteck2').toggle(2000);
}
$('#rechteck2').hide();
$('#btn').click(tauschen);
</script>
</body>
</html>
```



AJAX

WAS IST AJAX & WELCHE VORTEILE BIETET DIESE TECHNIK

- T Asynchronous JavaScript und XML => asynchroner Ablauf des Programms
- T Mit AJAX möglich, präzise Anfragen an den Server zu senden, um genau die benötigten Inhalte anzufordern, die dann in die bestehende Seite eingefügt werden können
- T Es werden nur Bausteine ausgetauscht => reduziert Ladezeiten
- T Ladevorgang läuft im Hintergrund ab
- T Für nachfolgende Beispiele bei lokaler Entwicklung wird von XAMPP das Apache Modul benötigt

ZUSÄTZLICHE INFORMATIONEN MIT AJAX ANFORDERN

T Datei: nachrichten.txt

Ein warmes Hallöchen aus der TXT-Datei

T Datei: index.html (Im: xampp\htdocs\ajaxUebung\)

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Übung</title>
  <style>
    #ausgabe { height: 100px; width: 200px; }
  </style>
</head>
<body>
<h1>Nachricht anzeigen</h1>
<textarea id="ausgabe"></textarea>
<button type="button" id="btn" onclick="laden()">Inhalt laden</button>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
  function laden() {
    /* vordefiniertes Objekt, dass in allen
       modernen Browsern vorhanden ist */
    let ajaxObj = new XMLHttpRequest();

    /* .onreadystatechange enthält eine Funktion,
       * die dann ausgeführt wird,
       * wenn sich der Status der aktuellen Anfrage ändert.
       * Bei jeder Statusänderung wird die Funktion ausgeben()
       * aufgerufen */
    ajaxObj.onreadystatechange = ausgeben;
```

```
    /* Art der Übermittlung festlegen, für dieses Beispiel egal...
       * danach folgt die Datei, die aufgerufen werden soll
       * true erlaubt es, asynchron zu arbeiten. Mit false
       * würde der Programmteil
       * synchron ablaufen */
    ajaxObj.open("GET", "nachrichten.txt", true);

    /* .send() sorgt dafür, dass die Seite entsprechende
       * Anfrage an Server absendet und die Daten anfordert */
    ajaxObj.send();
  }
  /* wird immer dann ausgeführt wenn sich der Status ändert */
  function ausgeben() {
    /* es gibt auch Status-Änderungen auf die nicht reagiert werden
       * sollen wenn XMLHttpRequest-Objekt
       * das readyState-Attribut den Wert 4
       * && wenn Übermittlung erfolgreich war (200) */
    if(this.readyState == 4 && this.status == 200) {
      document.getElementById('ausgabe').innerHTML =
this.responseText;
    }
  }
</script>
</body>
</html>
```

EINE ALLGEMEINE FUNKTION FÜR DIE ANFORDERUNG DER DATEN

- T Beispiel umgebaut um allgemeiner auf Anfrage reagieren zu können
- T Der Link, zur .txt-Datei sowie die ID vom Zielelement werden der Funktion übergeben

```
<body>
<h1 id="ueberschrift"></h1>
<textarea id="ausgabe"></textarea>
<button type="button" id="btn" onclick="laden('nachrichten.txt', 'ausgabe')">Inhalt laden</button>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
    function laden(url, element) {
        let ajaxObj = new XMLHttpRequest();
        ajaxObj.onreadystatechange = function () {
            if(this.readyState == 4 && this.status == 200) {
                document.getElementById(element).innerHTML = this.responseText;
            }
        };
        ajaxObj.open("GET", url, true);
        ajaxObj.send();
    }
    laden('ueberschrift.txt', "ueberschrift");
</script>
</body>
```

AUFGABE

- T Gestalte eine Seite mit einer Überschrift, einer Unterüberschrift und mit einem Absatz. Dabei sollen jedoch nur die entsprechenden HTML-Tags vorhanden sein. Die Inhalte werden nach der Betätigung eines Buttons per AJAX abgerufen – aus drei unterschiedlichen Dateien. Verwende für den Abruf der Inhalte in allen drei Fällen die gleiche Funktion.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Übung</title>
</head>
<body>
<h1 id="ueberschrift1"></h1>
<h2 id="ueberschrift2"></h2>
<p id="absatz"></p>
<button type="button" id="btn" onclick="anzeigen()">Inhalt
laden</button>
<script src="https://code.jquery.com/jquery-
3.5.1.min.js"></script>
<script>
  function laden(url, element, eingabe = "") {
    let ajaxObj = new XMLHttpRequest();
    ajaxObj.onreadystatechange = function () {
      if(this.readyState == 4 && this.status == 200) {
        document.getElementById(element).innerHTML =
          this.responseText;
      }
    };
  }
};
```

```
    ajaxObj.open("GET", url + "?wert=" + eingabe, true);
    ajaxObj.send();
  }
  function anzeigen() {
    laden('ueberschrift1.txt', 'ueberschrift1');
    laden('ueberschrift2.txt', 'ueberschrift2');
    laden('absatz.txt', 'absatz');
  }
</script>
</body>
</html>
```




ENDE

QUELLE: JAVASCRIPT
PROGRAMMIEREN FÜR EINSTEIGER
ISBN: 978-3-96645-016-4