JavaScript und JavaScript Frameworks

Events

Übersicht und Zielstellung

- In diesem Abschnitt erfahren Sie:
  - welche Events im jQuery genutzt werden,
  - wie man diese Events auswerten kann,
  - was auf der Basis der Events erfolgen kann
  - das Events auch Informationen liefern,

Start in das Thema

Events im jQuery

Interaktionen

jQuery bietet eine Unmenge von Möglichkeiten, den Inhalt und Aufbau einer Webseite zu steuern.

Ein weiterer wichtiger Aspekt ist dabei aber noch, festzulegen WANN! „jQuery“ diese Steuerung übernimmt.

- Interaktion durch den Anwender
  - Einsatz der Maus bzw. deren Alternativen
  - Eingaben
  - Seitenaufrufe
  - …
- Auswertung von Aktivitäten
  - Mauspositionen
  - Änderungen am und im Fenster
  - angeklickte Optionen
  - …

Events im jQuery

Übersicht der wichtigsten Events

Klick-Events:	Diese werden ausgelöst, wenn der Benutzer auf ein HTML-Element klickt.
Tastatur-Events:	Diese werden ausgelöst, wenn der Benutzer eine Taste auf der Tastatur drückt.
Maus-Events:	Diese werden ausgelöst, wenn der Benutzer die Maus über ein HTML-Element bewegt oder auf einem HTML-Element klickt.
Formular-Events:	Diese werden ausgelöst, wenn der Benutzer ein Formular abschickt.
Laden-Events:	Diese werden ausgelöst, wenn das Dokument vollständig geladen wurde.
Unload-Events:	Diese werden ausgelöst, wenn das Dokument geschlossen oder neu geladen wird.
Fehler-Events:	Diese werden ausgelöst, wenn ein Fehler auftritt, z.B. wenn ein Bild nicht geladen werden kann.
Größenänderungs-Events:	Diese werden ausgelöst, wenn die Größe eines HTML-Elements geändert wird.
Scroll-Events:	Diese werden ausgelöst, wenn der Benutzer eine Webseite scrollt.
Fokus-Events:	Diese werden ausgelöst, wenn ein HTML-Element den Fokus erhält.
Blur-Events:	Diese werden ausgelöst, wenn ein HTML-Element den Fokus verliert.
Änderungs-Events:	Diese werden ausgelöst, wenn der Benutzer den Wert eines Formular-Elements ändert.

Events im jQuery

Der Klick-Event

Das Klick-Event ist ein sehr häufig genutztes Event in der Webentwicklung, da es ermöglicht, auf Benutzerinteraktionen wie Mausklicks oder einfache Touch zu reagieren.

Das Klick-Event wird in jQuery mit der Methode **click()** behandelt. Die Syntax ist wie folgt:

```
$(selector).click(function(){
    // Hier wird der Code ausgeführt, wenn das Element geklickt wird
});
```

Als Selektor - **$(selector)** - kann jedes sichtbare Element genutzt werden.

Das Klick-Event kann auch mit anderen Events kombiniert werden, z.B. um einen doppelten Klick **dblclick()** zu erfassen. Hier ist ein Beispiel:

```
$("button").dblclick(function(){
    // Hier wird der Code ausgeführt, wenn der Benutzer einen Button doppelt klickt
});
```

Events im jQuery

Klick-Event

Das Klick-Event kann auch mit der **on()** Methode verwendet werden, um mehrere Events zu behandeln.
Hier ist ein Beispiel:

```
$("button").on({
   click: function(){
      // Hier wird der Code ausgeführt, wenn der Benutzer einen Button klickt
   },
   dblclick: function(){
      // Hier wird der Code ausgeführt, wenn der Benutzer einen Button doppelklickt
   },
   mouseenter: function(){
     // Hier wird der Code ausgeführt, wenn der Benutzer mit der Maus über einen Button geht
   },
   mouseleave: function(){
    // Hier wird der Code ausgeführt, wenn der Benutzer mit der Maus aus dem Button herausgeht
   }
});
```

Events im jQuery

Tastatur-Events

Tastatur-Events ermöglichen es, auf Benutzerinteraktionen mit der Tastatur zu reagieren. Diese Events sehr nützlich, um Benutzereingaben zu validieren oder Aktionen auszuführen, wenn eine bestimmte Taste gedrückt wird.

In jQuery gibt es verschiedene Tastatur-Events, auf die reagiert werden kann, zum Beispiel **keydown**, **keyup** und **keypress**.

- **keydown:** wird ausgelöst, wenn eine Taste auf der Tastatur gedrückt wird.
- **keyup:** wird ausgelöst, wenn eine Taste auf der Tastatur losgelassen wird.
- **keypress:** wird ausgelöst, wenn eine Taste auf der Tastatur gedrückt wird und ein Zeichen generiert wird.

Die Syntax zur Verwendung von Tastatur-Events in jQuery ist ähnlich wie beim Klick-Event:

```
$(selector).keydown(function(event){
    // Hier wird der Code ausgeführt, wenn eine Taste gedrückt wird
});
```

Im obigen Beispiel wird das **keydown-Event** behandelt. Die keydown()-Methode erwartet eine Funktion als Parameter, die ausgeführt wird, wenn das Event ausgelöst wird. Die Funktion nimmt auch ein event-Objekt als Parameter entgegen, das Informationen über das Event enthält, z.B. welche Taste gedrückt wurde.

Events im jQuery

Tastatur-Events

Hier sind einige Beispiele für die Verwendung von Tastatur-Events in jQuery:

```
// Auf alle Eingabefelder anwenden
$("input").keydown(function(event){
    // Hier wird der Code ausgeführt, wenn eine Taste in einem Eingabefeld gedrückt wird
});
// Auf ein bestimmtes Eingabefeld mit der ID "meins" anwenden
$("#meins").keyup(function(event){
    // Hier wird der Code ausgeführt, wenn eine Taste im Elementmit der ID "meins" losgelassen wird });
// Auf die Enter-Taste reagieren
$(document).keypress(function(event){
    if(event.which == 13){
    // Hier wird der Code ausgeführt, wenn die Enter-Taste gedrückt wird
    }
});
```

Events im jQuery

Tastatur-Events

Im letzten Beispiel wird das keypress-Event auf das gesamte Dokument angewendet und dann geprüft, ob die Enter-Taste gedrückt wurde. Wenn dies der Fall ist, wird der Code innerhalb der if-Anweisung ausgeführt.

Es ist auch möglich, Tastenkombinationen zu behandeln, indem man prüft, ob mehrere Tasten gleichzeitig gedrückt wurden. Hier ist ein Beispiel:

```
$(document).keydown(function(event){
    if(event.ctrlKey && event.which == 83){
        // Hier wird der Code ausgeführt, wenn die Strg+S-Tastenkombination gedrückt wird
    }
});
```

In diesem Beispiel wird das keydown-Event auf das gesamte Dokument angewendet und dann geprüft, ob sowohl die Strg-Taste als auch die S-Taste gedrückt wurden. Wenn dies der Fall ist, wird der Code innerhalb der if-Anweisung ausgeführt.

Events im jQuery

Maus-Events

Maus-Events ermöglichen es, auf Benutzerinteraktionen mit der Maus zu reagieren. Diese Events sind in der Regel sehr nützlich, um Benutzereingaben zu validieren oder Aktionen auszuführen, wenn eine bestimmte Mausaktion ausgeführt wird.

Hier sind alle Maus-Events in jQuery und eine kurze Erklärung, was jedes Event auslöst:

- **click:** wird ausgelöst, wenn ein Element mit der Maus geklickt wird. Es entspricht einem Klick mit der linken Maustaste.
- **dblclick:** wird ausgelöst, wenn ein Element mit der Maus doppelt geklickt wird.
- **mousedown:** wird ausgelöst, wenn eine Maustaste gedrückt wird. Es kann mit der linken, mittleren oder rechten Maustaste ausgelöst werden.
- **mouseup:** wird ausgelöst, wenn eine Maustaste freigegeben wird. Es kann mit der linken, mittleren oder rechten Maustaste ausgelöst werden.
- **mousemove:** wird ausgelöst, wenn die Maus über ein Element bewegt wird.
- **mouseover:** wird ausgelöst, wenn die Maus über ein Element bewegt wird und es betritt.
- **mouseout:** wird ausgelöst, wenn die Maus ein Element verlässt.

Events im jQuery

Maus-Events

Hier sind einige Beispiele für die Verwendung von Maus-Events in jQuery:

```
// auf alle Bilder anwenden
$("img").click(function(event){
   // Hier wird der Code ausgeführt, wenn das Bild geklickt wird
});
// auf ein bestimmtes Element mit der ID "meinElement" anwenden
$("#meinElement").mousemove(function(event){
   // Hier wird der Code ausgeführt, wenn die Maus über das Element mit der ID "meinElement" bewegt wird
});
// auf die mittlere Maustaste reagieren
$("p").mousedown(function(event){
    if(event.which == 2){
        // Hier wird der Code ausgeführt, wenn die mittlere Maustaste auf einem <p>-Element gedrückt wird
    }
});
// auf Mausereignisse von Kinderelementen reagieren
$("ul").on("click", "li", function(event){
   // Hier wird der Code ausgeführt, wenn ein <li>-Element in einem <ul>-Element geklickt wird
});
```

Events im jQuery

Maus-Events

Die Syntax zur Verwendung von Maus-Events in jQuery ist ähnlich wie beim Klick-Event:

```
$(selector).click(function(event){
   // Hier wird der Code ausgeführt, wenn das Element geklickt wird
});
```

Im obigen Beispiel wird das click-Event behandelt.
Die click()-Methode erwartet eine Funktion als Parameter, die ausgeführt wird, wenn das Event ausgelöst wird.

```
// Auf die rechte Maustaste reagieren
$("p").mousedown(function(event){
   if(event.which == 3){
     // Der Code wird ausgeführt, wenn die rechte Maustaste auf einem <p>-Element gedrückt wird
   }
});
```

Events im jQuery

Maus-Events

Es ist auch möglich, die Position der Maus zu ermitteln, wenn ein Maus-Event ausgelöst wird.
Hier ist ein Beispiel:

```
$("div").mousemove(function(event){
    var x = event.pageX;
    var y = event.pageY;
    // Hier kann der Code ausgeführt werden, der die Mausposition verwendet
});
```

In diesem Beispiel wird das mousemove-Event auf alle -Elemente angewendet. Die **pageX**- und **pageY**-Eigenschaften des event-Objekts können verwendet werden, um die Position der Maus auf der Seite zu ermitteln.

In der anschließenden Code-Logik kann dann die Mausposition verwendet werden, um beispielsweise ein **Tooltip-Element** anzuzeigen oder die Positionierung eines Elements zu ändern.

Events im jQuery

Formular-Events

Um Aktivitäten bei der Arbeit mit einem Formular auszuwerten, gibt es eine Vielzahl von Events.

submit: wird ausgelöst, wenn ein Formular abgesendet wird.
reset: wird ausgelöst, wenn ein Formular zurückgesetzt wird.
change: wird ausgelöst, wenn der Wert eines Formularelements (z. B. Eingabefeld, Auswahlliste) geändert wird und das Element den Fokus verliert.
focus: wird ausgelöst, wenn ein Formularelement den Fokus erhält (z. B. durch Klicken oder Tabulatortaste).
blur: wird ausgelöst, wenn ein Formularelement den Fokus verliert (z. B. wenn der Benutzer zu einem anderen Element navigiert).
keydown: wird ausgelöst, wenn eine Taste auf der Tastatur gedrückt wird, während sich ein Formularelement im Fokus befindet.
keyup: wird ausgelöst, wenn eine Taste auf der Tastatur losgelassen wird, während sich ein Formularelement im Fokus befindet.
keypress: wird ausgelöst, wenn eine Taste auf der Tastatur gedrückt wird, während sich ein Formularelement im Fokus befindet, und der Tastendruck ein tatsächliches Zeichen erzeugt.
select: wird ausgelöst, wenn der Benutzer Text in ein Textfeld oder eine Auswahlliste auswählt.
focusin: wird ausgelöst, wenn ein Formularelement oder ein Kindelement den Fokus erhält.
focusout: wird ausgelöst, wenn ein Formularelement oder ein Kindelement den Fokus verliert.

Events im jQuery

Formular-Events Hier ein paar ausgewählte Beispiele

```
// auf das Absenden eines Formulars reagieren
$("form").submit(function(event){
   event.preventDefault(); // verhindert das Standardverhalten des Absendens des Formulars
  // Hier wird der Code ausgeführt, wenn das Formular abgesendet wird
});

// auf das Zurücksetzen eines Formulars reagieren
$("form").reset(function(event){ // Hier wird der Code ausgeführt, wenn das Formular zurückgesetzt wird });

// auf eine Änderung in einem Eingabefeld reagieren
$("#meinEingabefeld").change(function(event){ // Hier wird der Code ausgeführt, wenn der Wert des Eingabefelds geändert wird });

// auf das Betreten eines Formularelements reagieren
$("input").focus(function(event){ // Der Code ausgeführt, wenn ein Formularelement den Fokus erhält });

// auf das Verlassen eines Formularelements reagieren
$("input").blur(function(event){ // Hier wird der Code ausgeführt, wenn ein Formularelement den Fokus verliert });

// auf die Eingabe in einem Eingabefeld reagieren
$("#meinEingabefeld").keydown(function(event){ // Hier wird der Code ausgeführt, wenn eine Taste auf der Tastatur gedrückt wird });

// auf die Freigabe einer Taste in einem Eingabefeld reagieren
$("#meinEingabefeld").keyup(function(event){ // Hier wird der Code ausgeführt, wenn eine Taste auf der Tastatur losgelassen wird });
```

Events im jQuery

Laden- und Entladen-Events

Laden-Events in jQuery sind Events, die ausgelöst werden, wenn eine Webseite oder ein Teil davon geladen wird oder wenn der Browser ein neues Element in das DOM einfügt. Hier sind die wichtigsten Laden-Events in jQuery:

- **$(document).ready():** wird ausgelöst, wenn das DOM vollständig geladen ist und alle Elemente verfügbar sind. Es wird empfohlen, Code, der auf Elemente im DOM zugreift, innerhalb dieses Events auszuführen.
- **$(window).load():** wird ausgelöst, wenn die gesamte Seite, einschließlich Bilder und andere Ressourcen, vollständig geladen ist.
- **$(window).unload():** wird ausgelöst, wenn der Benutzer die Seite verlässt oder die Seite neu geladen wird.

```
// auf das vollständige Laden des DOM reagieren
$(document).ready(function(){
   // Hier wird der Code ausgeführt, wenn das DOM vollständig geladen ist
});
// auf das vollständige Laden der Seite reagieren
$(window).load(function(){
   // Hier wird der Code ausgeführt, wenn die gesamte Seite, inclusive Ressourcen, vollständig geladen ist
});
// auf das Verlassen der Seite reagieren
$(window).unload(function(){
   // Hier wird der Code ausgeführt, wenn der Benutzer die Seite verlässt oder die Seite neu geladen wird
});
```

Events im jQuery

Fehler-Events

Fehler-Events in jQuery werden ausgelöst, wenn ein Fehler auf der Webseite auftritt, z.B. wenn eine Ressource nicht geladen werden kann, ein Skriptfehler auftritt oder eine AJAX-Anfrage fehlschlägt. Hier sind die wichtigsten Fehler-Events in jQuery:

- $(window).error(): wird ausgelöst, wenn ein JavaScript-Fehler auftritt oder eine Ressource, wie ein Bild, nicht geladen werden kann.

```
// auf einen JavaScript-Fehler reagieren
$(window).error(function(event) {
   // Hier wird der Code ausgeführt, wenn ein JavaScript-Fehler auftritt
});
```

Events im jQuery

Größenänderungs-Events

Größenänderungs-Events in jQuery werden ausgelöst, wenn sich die Größe eines Elements oder des Browserfensters ändert.

- **$(window).resize():** wird ausgelöst, wenn sich die Größe des Browserfensters ändert.
- **$(document).resize():** wird ausgelöst, wenn sich die Größe des Dokumentelements ändert.
- **$(element).resize():** wird ausgelöst, wenn sich die Größe des Elements ändert.

```
// auf eine Größenänderung des Browserfensters reagieren
$(window).resize(function() {
  // Hier wird der Code ausgeführt, wenn sich die Größe des Browserfensters ändert
});

// auf eine Größenänderung des Dokumentelements reagieren
$(document).resize(function() {
  // Hier wird der Code ausgeführt, wenn sich die Größe des Dokumentelements ändert
});

// auf eine Größenänderung des Elements reagieren
$(element).resize(function() {
  // Hier wird der Code ausgeführt, wenn sich die Größe des Elements ändert
});
```

Events im jQuery

Scroll-Events

Scroll-Events in jQuery werden ausgelöst, wenn der Benutzer eine Seite oder ein Element scrollt, indem er beispielsweise die Mausrad- oder die Pfeiltasten verwendet.

Sie können verwendet werden, um auf den Scrollvorgang zu reagieren und bestimmte Aktionen auszuführen, wenn ein bestimmter Scrollpunkt scrollTop(), scrollLeft() erreicht wird oder wenn der Benutzer den Scrollvorgang beendet.

- $(window).scroll(): wird ausgelöst, wenn das Browserfenster gescrollt wird.
- $(element).scroll(): wird ausgelöst, wenn ein Element gescrollt wird.

- $(window).scrollstart(): wird ausgelöst, wenn der Scrollvorgang beginnt.
- $(element).scrollstop(): wird ausgelöst, wenn das Scrollen gestopp wird.

```
$(window).scroll(function() {
    // Code ausführen, wenn das Fenster gescrollt wird
});

$("#element").scroll(function() {
    // Code ausführen, wenn das Element gescrollt wird
});
```

VIELEN DANK!