



FH

University of
Applied Sciences

TECHNIKUM

WIEN

Ajax im Frontend

Agenda

- Ziel
- Die Ajax Methode in JQuery
- Beispielanwendung
- Nachladen von HTML
- Beispielanwendung

Ziel

Nach durcharbeiten des Foliensatzes sind sie in der Lage:

- Mit JQuery einfache **Ajax Calls durchzuführen** um Daten mit dem Backend auszutauschen
- **JSON Daten** zum Server zu **senden** bzw. vom Server zu **empfangen**

Die Ajax Methode in JQuery

JQuery stellt eine umfassende Ajax Methode zur Verfügung

Siehe auch JQuery API => <https://api.jquery.com/>

Bzw. insbesondere => <https://api.jquery.com/jquery.ajax/>

Die Methode kann durch ein settings Objekt umfangreich konfiguriert werden

Die Ajax Methode in JQuery

JQuery stellt eine umfassende Ajax Methode zur Verfügung

Beispielhafter Aufruf mit häufigen Parametern:

```
$.ajax({
  type: "GET",
  url: "../serviceHandler.php",
  cache: false,
  data: {method: "queryPersonByName", param: searchTerm},
  dataType: "json",
  success: function (response) {

    $("#noOfentries").val(response.length);
    $("#searchResult").show(1000).delay(1000).hide(1000);

  }
});
```

type (default: `'GET'`)

Type: [String](#)

An alias for `method`. You should use `type` if you're using versions of jQuery prior to 1.9.0.

method (default: `'GET'`)

Type: [String](#)

The HTTP method to use for the request (e.g. `"POST"`, `"GET"`, `"PUT"`). (version added: [1.9.0](#))

Quelle: JQuery API

Die Ajax Methode in JQuery

JQuery stellt eine umfassende Ajax Methode zur Verfügung

Beispielhafter Aufruf mit häufigen Parametern:

```
$.ajax({  
  type: "GET",  
  url: "../serviceHandler.php",  
  cache: false,  
  data: {method: "queryPersonByName", param: searchTerm},  
  dataType: "json",  
  success: function (response) {  
    $("#noOfentries").val(response.length);  
    $("#searchResult").show(1000).delay(1000).hide(1000);  
  }  
});
```

url (default: The current page)

Type: [String](#)

A string containing the URL to which the request is sent.

Quelle: JQuery API

Die Ajax Methode in JQuery

JQuery stellt eine umfassende Ajax Methode zur Verfügung

Beispielhafter Aufruf mit häufigen Parametern:

```
$.ajax({
  type: "GET",
  url: "../serviceHandler.php",
  cache: false,
  data: {method: "queryPersonByName", param: searchTerm},
  dataType: "json",
  success: function (response) {
    $("#noOfentries").val(response.length);
    $("#searchResult").show(1000).delay(1000).hide(1000);
  }
});
```

cache (default: `true`, `false` for dataType `'script'` and `'jsonp'`)

Type: [Boolean](#)

If set to `false`, it will force requested pages not to be cached by the browser. **Note:** Setting `cache` to `false` will only work correctly with HEAD and GET requests. It works by appending `"_={timestamp}"` to the GET parameters. The parameter is not needed for other types of requests, except in IE8 when a POST is made to a URL that has already been requested by a GET.

Quelle: JQuery API

Die Ajax Methode in JQuery

JQuery stellt eine umfassende Ajax Methode zur Verfügung

Beispielhafter Aufruf mit häufigen Parametern:

```
$.ajax({
  type: "GET",
  url: "../serviceHandler.php",
  cache: false,
  data: {method: "queryPersonByName", param: searchTerm},
  dataType: "json",
  success: function (response) {
    $("#noOfentries").val(response.length);
    $("#searchResult").show(1000).delay(1000).hide(1000);
  }
});
```

data

Type: [PlainObject](#) or [String](#) or [Array](#)

Data to be sent to the server. If the HTTP method is one that cannot have an entity body, such as GET, the `data` is appended to the URL.

When `data` is an object, jQuery generates the data string from the object's key/value pairs unless the `processData` option is set to `false`. For example, `{ a: "bc", d: "e,f" }` is converted to the string `"a=bc&d=e%2Cf"`. If the value is an array, jQuery serializes multiple values with same key based on the value of the `traditional` setting (described below). For example, `{ a: [1,2] }` becomes the string `"a%5B%5D=1&a%5B%5D=2"` with the default `traditional: false` setting.

Quelle: JQuery API

Die Ajax Methode in JQuery

JQuery stellt eine umfassende Ajax Methode zur Verfügung

Beispielhafter Aufruf mit häufigen Parametern:

```
$.ajax({
  type: "GET",
  url: "../serviceHandler.php",
  cache: false,
  data: {method: "queryPersonByName", param: searchTerm},
  dataType: "json",
  success: function (response) {
    $("#noOfentries").val(response.length);
    $("#searchResult").show(1000).delay(1000).hide(1000);
  }
});
```

dataType (default: `Intelligent Guess (xml, json, script, or html)`)

Type: [String](#)

The type of data that you're expecting back from the server. If none is specified, jQuery will try to infer it based on the MIME type of the response (an XML MIME type will yield XML, in 1.4 JSON will yield a JavaScript object, in 1.4 script will execute the script, and anything else will be returned as a string). The available types (and the result passed as the first argument to your success callback) are:

`"xml"`: Returns a XML document that can be processed via jQuery.

`"html"`: Returns HTML as plain text; included script tags are evaluated when inserted in the DOM.

`"script"`: Evaluates the response as JavaScript and returns it as plain text. Disables caching by appending a query string parameter, `_=[TIMESTAMP]`, to the URL unless the `cache` option is set to `true`. **Note:** This will turn POSTs into GETs for remote-domain requests. Prior to jQuery 3.5.0, unsuccessful HTTP responses with a script `Content-Type` were still executed.

`"json"`: Evaluates the response as JSON and returns a JavaScript object. Cross-domain `"json"` requests that have a callback placeholder, e.g. `?callback=?`, are performed using [JSONP](#) unless the request includes `jsonp: false` in its request options. The JSON data is parsed in a strict manner; any malformed JSON is rejected and a parse error is thrown. As of jQuery 1.9, an empty response is also rejected; the server should return a response of `null` or `{}` instead. (See [json.org](#) for more information on proper JSON formatting.)

`"jsonp"`: Loads in a JSON block using [JSONP](#). Adds an extra `"?callback=?"` to the end of your URL to specify the callback. Disables caching by appending a query string parameter, `_=[TIMESTAMP]`, to the URL unless the `cache` option is set to `true`.

`"text"`: A plain text string.

Die Ajax Methode in JQuery

JQuery stellt eine umfassende Ajax Methode zur Verfügung

Beispielhafter Aufruf mit häufigen Parametern:

```
$.ajax({
  type: "GET",
  url: "../serviceHandler.php",
  cache: false,
  data: {method: "queryPersonByName", param: searchTerm},
  dataType: "json",
  success: function (response) {
    $("#noOfentries").val(response.length);
    $("#searchResult").show(1000).delay(1000).hide(1000);
  }
});
```

success

Type: [Function](#)([Anything](#) data, [String](#) textStatus, [jqXHR](#) jqXHR)

A function to be called if the request succeeds. The function gets passed three arguments: The data returned from the server, formatted according to the `dataType` parameter or the `dataFilter` callback function, if specified; a string describing the status; and the `jqXHR` (in jQuery 1.4.x, XMLHttpRequest) object. **As of jQuery 1.5**, the `success` setting can accept an array of functions. Each function will be called in turn. This is an [Ajax Event](#).

Quelle: JQuery API

Hier wurde eine anonyme Methode verwendet die bei Erfolg ausgeführt wird. Alternativ kann auch auf eine eigene Methode verwiesen werden

Die Ajax Methode in JQuery

Weitere gängige Parameter:

error

error

Type: [Function](#)([jqXHR](#) jqXHR, [String](#) textStatus, [String](#) errorThrown)

A function to be called if the request fails. The function receives three arguments: The jqXHR (in jQuery 1.4.x, XMLHttpRequest) object, a string describing the type of error that occurred and an optional exception object, if one occurred. Possible values for the second argument (besides `null`) are `"timeout"`, `"error"`, `"abort"`, and `"parsererror"`. When an HTTP error occurs, `errorThrown` receives the textual portion of the HTTP status, such as "Not Found" or "Internal Server Error." (in HTTP/2 it may instead be an empty string) **As of jQuery 1.5**, the `error` setting can accept an array of functions. Each function will be called in turn. **Note:** *This handler is not called for cross-domain script and cross-domain JSONP requests.* This is an [Ajax Event](#).

Quelle: JQuery API

Die Ajax Methode in JQuery

Weitere gängige Parameter:

jsonp

jsonp

Type: [String](#) or [Boolean](#)

Override the callback function name in a JSONP request. This value will be used instead of 'callback' in the 'callback=?' part of the query string in the url. So `{ jsonp: 'onJSONPLoad' }` would result in `'onJSONPLoad=?'` passed to the server. **As of jQuery 1.5**, setting the `jsonp` option to `false` prevents jQuery from adding the `"?callback"` string to the URL or attempting to use `"=?"` for transformation. In this case, you should also explicitly set the `jsonpCallback` setting. For example, `{ jsonp: false, jsonpCallback: "callbackName" }`. If you don't trust the target of your Ajax requests, consider setting the `jsonp` property to `false` for security reasons.

Quelle: JQuery API

Wird für Cross domain requests verwendet

Die Ajax Methode in JQuery

Weitere gängige Parameter:

Complete

complete

Type: [Function](#)([jqXHR](#) jqXHR, [String](#) textStatus)

A function to be called when the request finishes (after `success` and `error` callbacks are executed). The function gets passed two arguments: The jqXHR (in jQuery 1.4.x, XMLHttpRequest) object and a string categorizing the status of the request (`"success"`, `"notmodified"`, `"nocontent"`, `"error"`, `"timeout"`, `"abort"`, or `"parsererror"`). **As of jQuery 1.5**, the `complete` setting can accept an array of functions. Each function will be called in turn. This is an [Ajax Event](#).

Quelle: JQuery API

Die Ajax Methode in JQuery

Weitere gängige Parameter:

Username
Password

username

Type: [String](#)

A username to be used with XMLHttpRequest in response to an HTTP access authentication request.

password

Type: [String](#)

A password to be used with XMLHttpRequest in response to an HTTP access authentication request.

Quelle: JQuery API

Beispielanwendung

Aufruf der in php realisierten Schnittstelle mit einem einfachen Client:



Available Names: "Doe" | "Smith"

Search via lastname

Via Ajax Methode und anzeige des Ergebnisses im Browser



Available Names: "Doe" | "Smith"

Doe

Number of entries found:

3

(Beispiel Implementierung in Moodle)

Beispielanwendung

HTML Seite wird mit Bootstrap umgesetzt:

Scripts angeben

- JQuery
- Bootstrap (bundle)
- Einer code (controller.js)

Bootstrap CSS verlinken

```
<html lang="en">
<head>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/js/bootstrap.bundle.min.js"
    integrity="sha384-ygbV9kiqUc6oa4msXn9868pTtWMgiQaeYH7/t7LECLbyPA2x65KgF800JFdroafw"
    crossorigin="anonymous"></script>
  <script src="controller.js" type="text/javascript"></script>

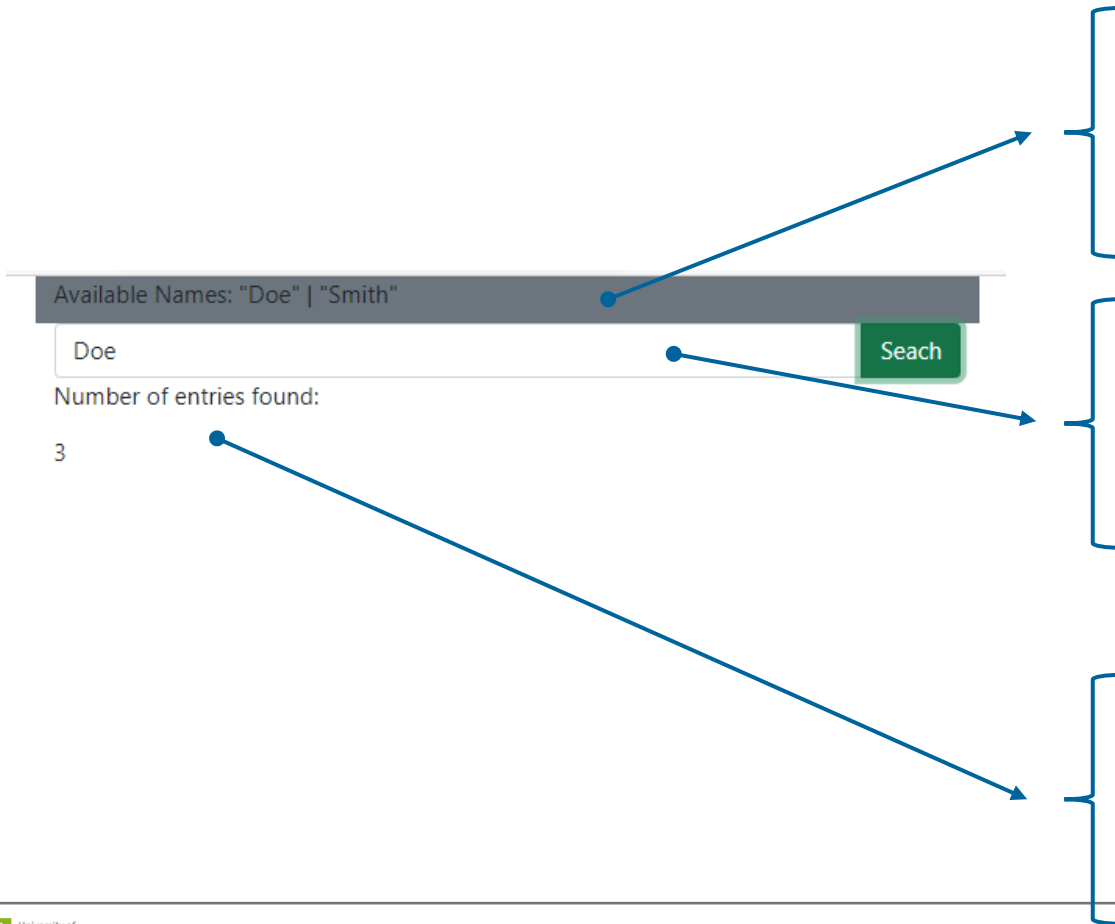
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-giJF6kkoqN000vy+HMDP7azOuL0xtbfIcaT9wjKHr8RbDVddVHyTfAAsrekwKmp1" crossorigin="anonymous"
  >
</head>
<body>
  <div id="container">
    <!--info-->
    <div class="row">
      <div class="col-sm"></div>
      <div class="col-sm-9 bg-secondary">
        <label for="seachfield" class="form-label">Available Names: "Doe" | "Smith"</label>
      </div>
      <div class="col-sm"></div>
    </div>

    <!--Search field-->
    <div class="row">
      <div class="col-sm"></div>
      <div class="col-sm-9">
        <div class="input-group">
          <input type="text" class="form-control" id="seachfield" placeholder="Search via lastname">
          <button type="button" class="btn btn-success" id="btn_Search">Seach</button>
        </div>
      </div>
      <div class="col-sm"></div>
    </div>

    <!-- message -->
    <div class="row">
      <div class="col-sm"></div>
      <div class="col-sm-9 id="searchResult">
        <label for="noOfentries" class="form-label">Number of entries found:</label>
        <input type="text" readonly class="form-control-plaintext" id="noOfentries">
      </div>
      <div class="col-sm"></div>
    </div>
  </div>
</body>
</html>
```

Beispielanwendung

HTML Seite wird mit Bootstrap umgesetzt:



```
<html lang="en">
<head>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/js/bootstrap.bundle.min.js"
    integrity="sha384-ygbV9kiqUc6oa4msXn9868pTtWMgiQaeYH7/t7LECLbyPA2x65KgF800JFdroafw"
    crossorigin="anonymous"></script>
  <script src="controller.js" type="text/javascript"></script>

  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-giJF6kkoqNQ00vy+HMDP7azOuL0xtbfIcaT9wjKHr8RbDVddVHyTfAAsrekwKmp1" crossorigin="anonymous">
</head>
<body>
  <div id="container">
    <!--info-->
    <div class="row">
      <div class="col-sm"></div>
      <div class="col-sm-9 bg-secondary">
        <label for="seachfield" class="form-label">Available Names: "Doe" | "Smith"</label>
      </div>
      <div class="col-sm"></div>
    </div>

    <!--Search field-->
    <div class="row">
      <div class="col-sm"></div>
      <div class="col-sm-9">
        <div class="input-group">
          <input type="text" class="form-control" id="seachfield" placeholder="Search via lastname">
          <button type="button" class="btn btn-success" id="btn_Search">Seach</button>
        </div>
      </div>
      <div class="col-sm"></div>
    </div>

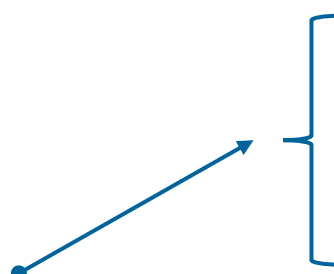
    <!-- message -->
    <div class="row">
      <div class="col-sm"></div>
      <div class="col-sm-9 id="searchResult">
        <label for="noOfentries" class="form-label">Number of entries found:</label>
        <input type="text" readonly class="form-control-plaintext" id="noOfentries">
      </div>
      <div class="col-sm"></div>
    </div>
  </div>
</body>
</html>
```

Beispielanwendung

Controller.js beinhaltet den
jQuery code:

Document Ready function

- Button klick Registrierung
- Aufruf der Ajax Funktion
nach Button klick

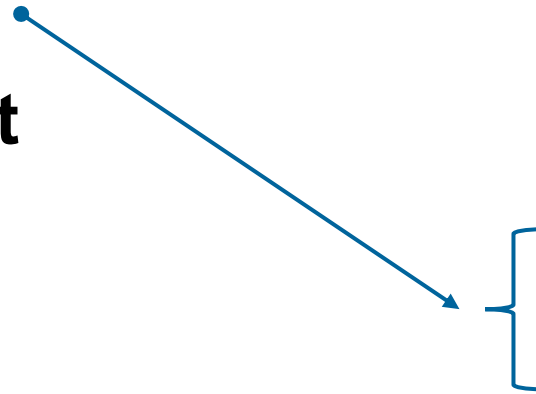


```
1 //Starting point for JQuery init
2 ✓ $(document).ready(function () {
3     $("#searchResult").hide();
4     ✓ $("#btn_Search").click(function (e) {
5         loaddata($("#seachfield").val());
6     });
7
8 });
9
10 //function containing the Ajax call
11 ✓ function loaddata(searchterm) {
12
13     $.ajax({
14         type: "GET",
15         url: "../serviceHandler.php",
16         cache: false,
17         data: {method: "queryPersonByName", param: searchterm},
18         dataType: "json",
19         ✓ success: function (response) {
20
21             $("#noOfentries").val(response.length);
22             $("#searchResult").show(1000).delay(1000).hide(1000);
23         }
24     });
25 }
26 }
27
```

Beispielanwendung

Controller.js beinhaltet den
jQuery code:

Nach erfolgreicher Abfrage am
Backend wird Wert in HTML Feld
geschrieben und
Sichtbarkeit geändert



```
1  //Starting point for JQuery init
2  ✓ $(document).ready(function () {
3      $("#searchResult").hide();
4  ✓  $("#btn_Search").click(function (e) {
5      loaddata($("#seachfield").val());
6  });
7
8  });
9
10 //function containing the Ajax call
11 ✓ function loaddata(searchterm) {
12
13  ✓  $.ajax({
14      type: "GET",
15      url: "../serviceHandler.php",
16      cache: false,
17      data: {method: "queryPersonByName", param: searchterm},
18      dataType: "json",
19  ✓  success: function (response) {
20
21      $("#noOfentries").val(response.length);
22      $("#searchResult").show(1000).delay(1000).hide(1000);
23  }
24  });
25  }
26  }
27
```

Nachladen von HTML

Die „load“ Methode kann als einfachste ajax Methode verwendet werden um dynamisch HTML Komponenten nachzuladen.

Description: Load data from the server and place the returned HTML into the matched elements.

 **.load(url [, data] [, complete])** version added: 1.0

url
Type: String
A string containing the URL to which the request is sent.
data
Type: PlainObject or String
A plain object or string that is sent to the server with the request.
complete
Type: Function(String responseText, String textStatus, jqXHR jqXHR)
A callback function that is executed when the request completes.

Note: Prior to jQuery 3.0, the event handling suite also had a method named `.load()`. Older versions of jQuery determined which method to fire based on the set of arguments passed to it.

This method is the simplest way to fetch data from the server. It is roughly equivalent to `$.get(url, data, success)` except that it is a method rather than global function and it has an implicit callback function. When a successful response is detected (i.e. when `textStatus` is "success" or "notmodified"), `.load()` sets the HTML contents of the matched elements to the returned data. This means that most uses of the method can be quite simple:

```
1 | $( "#result" ).load( "ajax/test.html" );
```

If no element is matched by the selector — in this case, if the document does not contain an element with id="result" — the Ajax request will *not* be sent.

Callback Function

If a "complete" callback is provided, it is executed after post-processing and HTML insertion has been performed. The callback is fired once for each element in the jQuery collection, and `this` is set to each DOM element in turn.

```
1 | $( "#result" ).load( "ajax/test.html", function() {  
2 |     alert( "Load was performed." );  
3 | } );
```

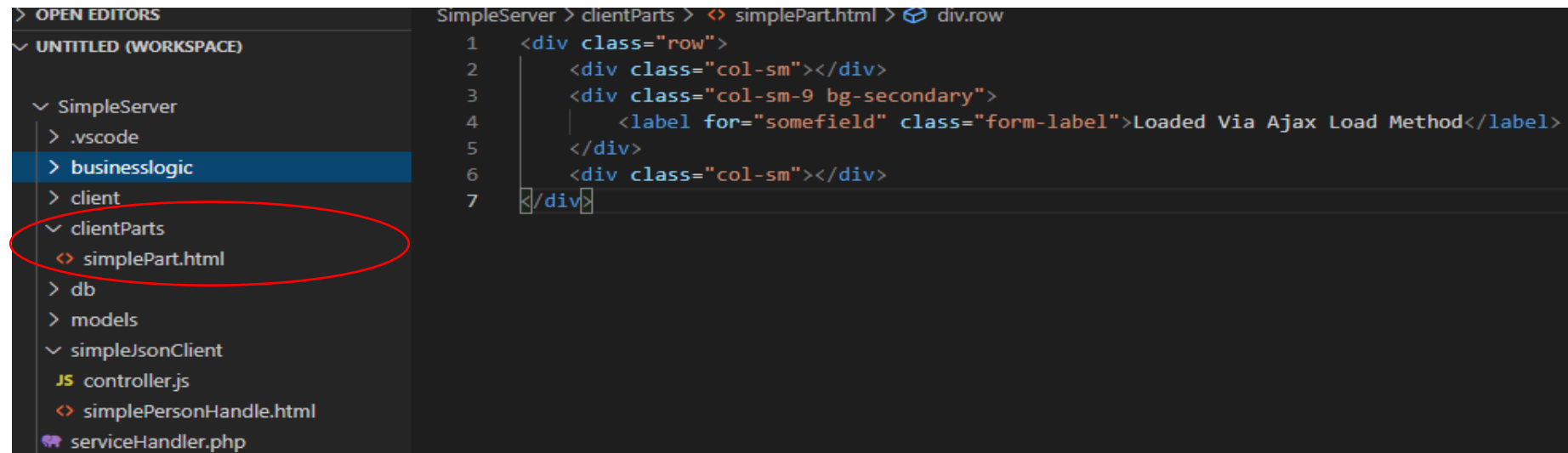
Quelle: JQuery API

Beispielanwendung

Erweiterung des Beispiels sodass nach erfolgreichen Ajax Call
Eine zusätzliche Information im Browser angezeigt wird.

Diese wird durch **Load** Befehl **dynamisch vom Server geladen**.

Hinzufügen eines Folders
Mit dem HTML content



```
> OPEN EDITORS
SimpleServer > clientParts > <> simplePart.html > div.row
1  <div class="row">
2    <div class="col-sm"></div>
3    <div class="col-sm-9 bg-secondary">
4      <label for="somefield" class="form-label">Loaded Via Ajax Load Method</label>
5    </div>
6    <div class="col-sm"></div>
7  </div>
```

Beispielanwendung

Erweiterung des Beispiels sodass nach erfolgreichen Ajax Call
Eine zusätzliche Information im Browser angezeigt wird.

Diese wird durch **Load** Befehl **dynamisch vom Server geladen**.

Controller.js erweitern um **Load** Aufruf

```
10 //function containing the Ajax call
11 function loaddata(searchterm) {
12
13     $.ajax({
14         type: "GET",
15         url: "../serviceHandler.php",
16         cache: false,
17         data: {method: "queryPersonByName", param: searchterm},
18         dataType: "json",
19         success: function (response) {
20
21             $("#noOfentries").val(response.length);
22             $("#searchResult").show(1000).delay(1000).hide(1000);
23             $("#dynamic").load("../clientParts/simplePart.html");
24         }
25     });
26 }
27
```