Bildungszentrum Uster
Höhere Berufsbildung Uster
HBU

JavaScript
Raphael Ritter

Ein Angebot der Berufsfachschule Uster und der Höheren Fachschule Uster

Agenda

- Prüfungsbesprechung
- jQuery
  - Wiederholung wichtigste Features
- Google Maps API
- JSON
- jQuery
  - Ajax
  - Promise
  - Handlebars

Bildungszentrum Uster
Höhere Berufsbildung
Uster
HBU

Rückblick

Unterlagen der HSR

Einige Inhalte sind aus den Folien / Übungen des CAS Frontend Engineering der Hochschule für Technik Rapperswil.

www.hsr.ch
http://hsr.ch/CAS-Front-End-Engineering.12432.0.html

Besprechung Prüfung

- Schaut euch eure Prüfung an und wir gehen die Lösungen nun gemeinsam durch


jQuery
write less, do more.

jQuery - Wiederhohlung

- JQuery erlaubt einfache, X-Browser-konsistente Abfrage und Manipulation des DOMs (sowie einfache Registrierung von Event Handlern und andere Utility Funktionen wie vereinfacht AJAX Calls)

```
$('#button2').text("**click me**")
[  <button id="button2" value="valueText">**click me**</button>]
```

jQuery - Wiederhohlung

- Die Hauptfunktionalität der JQuery-Library wird über die JQuery Funktion
  bzw. über die $ Funktion zugegriffen (im jQuery.noConflict Mode ist $ freigegeben)

```
> $
<- function ( selector, context ) {
        // The jQuery object is actually just the init constructor 'enhanced'
        // Need init if jQuery is called (just allow error to be thrown if not included)
        return new jQuery.fn.init( selector, context );
    }
> jQuery
<- function ( selector, context ) {
        // The jQuery object is actually just the init constructor 'enhanced'
        // Need init if jQuery is called (just allow error to be thrown if not included)
        return new jQuery.fn.init( selector, context );
    }
```

- Das Resultat einer JQuery Anfrage ist ein JQuery Result Set (Collection/Array)
  welches wiederum JQuery Methoden unterstützt und somit "function chaining" erlaubt

jQuery - Wiederhohlung

- Einbinden der JQuery Lib:
  (Local): <script src=”js/jquery-2.1.3.min.js"></script>
  (CDN): <script src="https://code.jquery.com/jquery-2.1.3.min.js">
  </script>
- Startpunkt (meist) statt window.onload in JQuery:
  $(document).ready(function () {
  ....
  });
  oder
  $(function () {
  ....
  });
- Resultat: Effiziente Beschreibung von Operationen auf Result Sets
  $(‘div’).hide();

jQuery - Wiederhohlung

- JQuery Anfragen nutzen den CSS Selector Syntax

```
$('p > em')
[  <em>with emphasized text</em>]
```

- JQuery Anfragen Resultate können der Startpunkt für weitere Navigation sein parent, parents, prev, next, children, oder die Liste gefiltert werden (first, last, filter, not)

```
$('p')
[ ▶<p id="p1">…</p>]
$('p').parent()
[ ▶<body>…</body>]
$('p').prev()
[  <em>with emphasized text</em>]
$('p').next()
[  <input id="button1" type="button" value="Click Me">]
$('p').children()
[  <em>with emphasized text</em>]
```

jQuery - Wiederhohlung

- Unterbäume können durchsucht werden

```
> $('p').first().find('em')
<- [  <em>with emphasized text</em>]
```

- JQuery bietet einen einfachen Snytax zur DOM Manipulation remove(), append(), val(), text()

```
> $('#text1').remove()
<- [  <input id="text1" type="text" value="replace text">]
> $('#emptyDiv').append($("<input id='text1' value='click
  for new text' />"))
<- [ ▶<div id="emptyDiv">…</div>]
> $('input[type=text]').val('type here')
<- [  <input id="text2" type="text" value="replace text">]
> $('button').text('**Click Me**')
<- [
    <button id="button2" value="#####">**Click
    Me**</button>
  ]
```

jQuery - Wiederhohlung

- JQuery bietet einen einfachen Syntax zur DOM Events zu definieren on(<eventIDString> [, <delegateSelector>], handlerFn)

```
$("body").on("click", bodyClickListener);
$('#emptyDiv').on("focus", "input",
        addNewText1Element);
```

- Wenn ein delegateSelector definiert ist, dann ist diese Event Definition "live" und gilt auch für Elemente die zum Zeitpunkt der Event-Definition noch nicht im DOM Tree sind (nutzt standard Event Bubbling)
- Mit `off` können Event-Handler de-registriert werden

```
function removeButton1ButtonClickListenerE2() {
    $("#button1").off("click", buttonClickListenerE2);
}
```

jQuery - Wiederhohlung

- Mit trigger können Events explizit ausgelöst werden (auch Custom Events)

```
$('#button1').trigger('click')
 (buttonClickListenerE1) clicked button      jQuery-demo.html:17
button1
 (buttonClickListenerE2) clicked button      jQuery-demo.html:24
button1
 (bodyClickListener) clicked body this =     jQuery-demo.html:34
[object HTMLBodyElement] event target =button1
[  <input id="button1" type="button" value="clicked 1 times">]
```

Übungen

- Übung 1.1 selbständig lösen

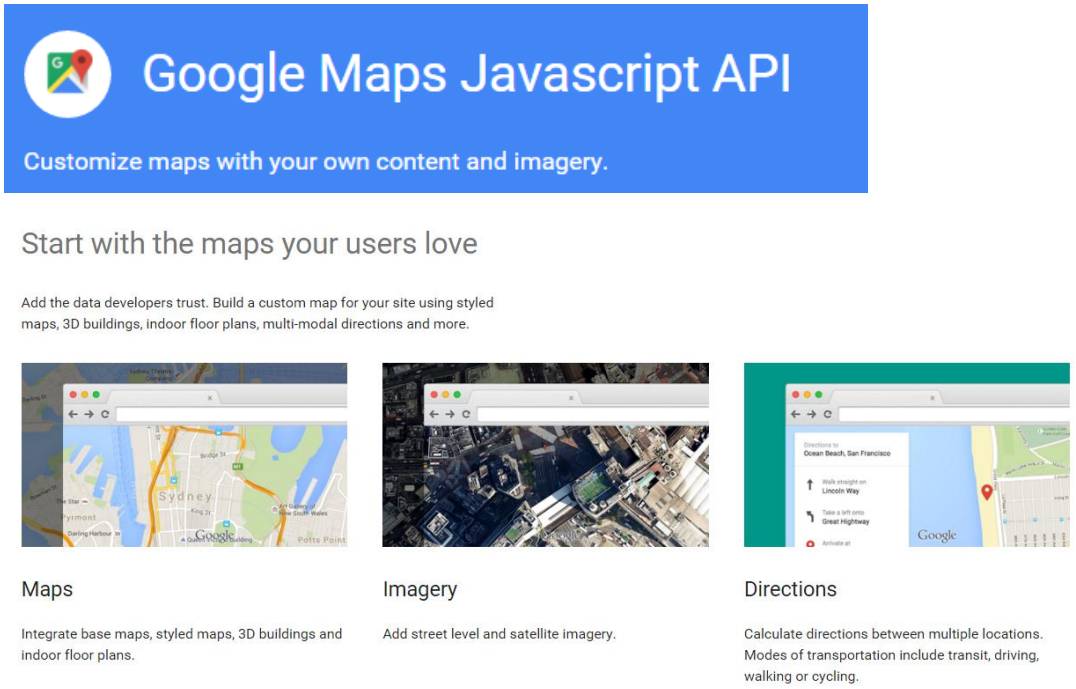
Bildungszentrum Uster
Höhere Berufsbildung Uster
HBU

Google Maps

JavaScript Library

Google Maps

https://developers.google.com/maps/documentation/javascript/tutorial



Google Maps

- Einbinden des JavaScript Files über eine CDN
  - Normalerweise wird ausserdem ein API Key den man Gratis lösen kann

```
<script async defer
  src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&callback=initMap">
</script>
```

  - Es geht momentan auch noch ohne

```
<script src="https://maps.googleapis.com/maps/api/js?v=3.exp"></script>
```

Google Maps

- Div erstellen in welchem die Map angezeigt werden soll

```
<div id="map"></div>
```

- Map Object per JavaScript erstellen
  - Das Zoomlevel bestimmt wie gross der angezeigte Kartenausschnitt ist
  - Die Center Option gibt an welcher Punkt (Koordinaten) zentriert dargestellt werden soll

```
map = new google.maps.Map(document.getElementById('map'), {
  center: {lat: -34.397, lng: 150.644},
  zoom: 8
});
```

Google Maps

- Um auf einer Karte bestimmte Positionen anzuzeigen werden Marker verwendet
  - Marker brauchen eine Position
  - Marker können Optional noch einen Text beinhalten

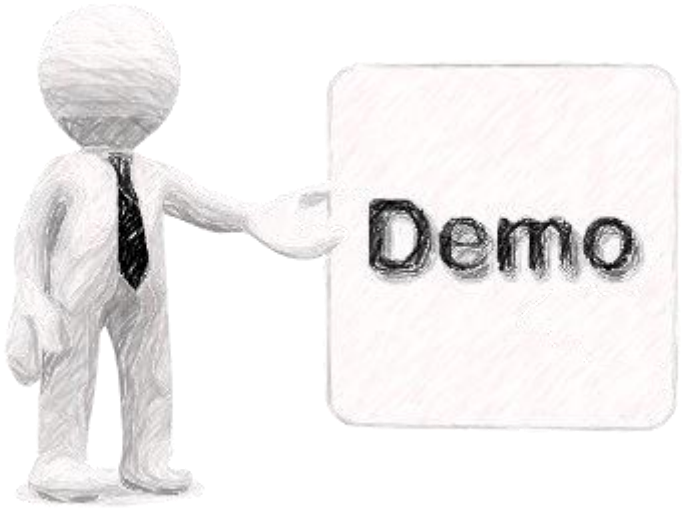
```
function initMap() {
  var myLatLng = {lat: -25.363, lng: 131.044};

  var map = new google.maps.Map(document.getElementById('map'), {
    zoom: 4,
    center: myLatLng
  });

  var marker = new google.maps.Marker({
    position: myLatLng,
    map: map,
    title: 'Hello World!'
  });
}
```

Google Maps

https://developers.google.com/maps/documentation/javascript/examples/marker-labels



Bildungszentrum Uster
Höhere Berufsbildung
Uster
HBU

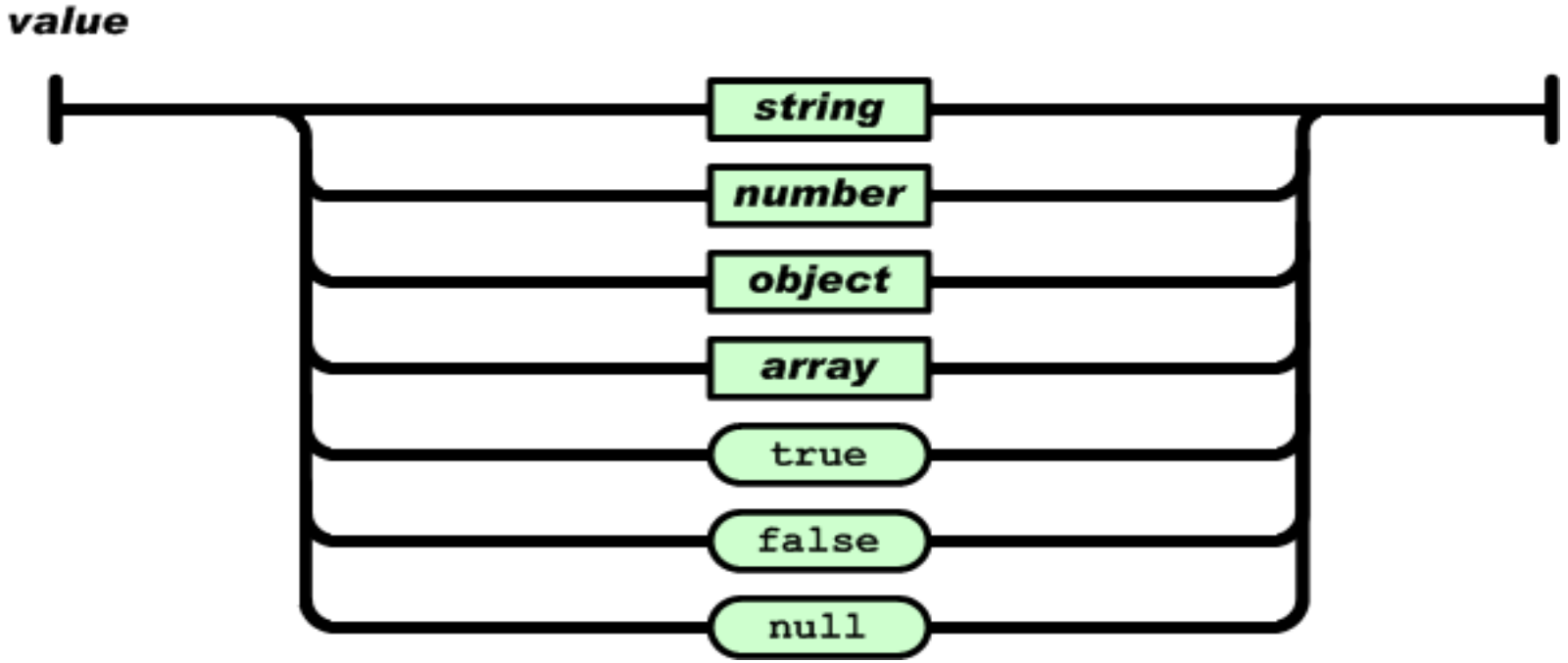
JSON

JavaScript Object Notation

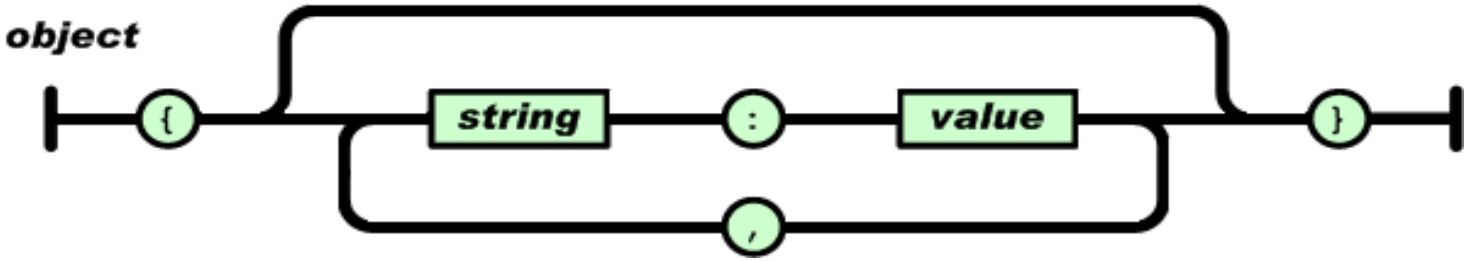
JSON

- JavaScript Object Notation (JSON)
- Leichtgewichtiges Datenaustauschformat
- Basiert auf der Object Literal Notation von JavaScript
- Ein Text Format, lesbar von Menschen und Maschinen
- Aufgrund der Einfachheit von vielen Programmiersprachen unterstützt
- http://www.JSON.org

jQuery - JSON



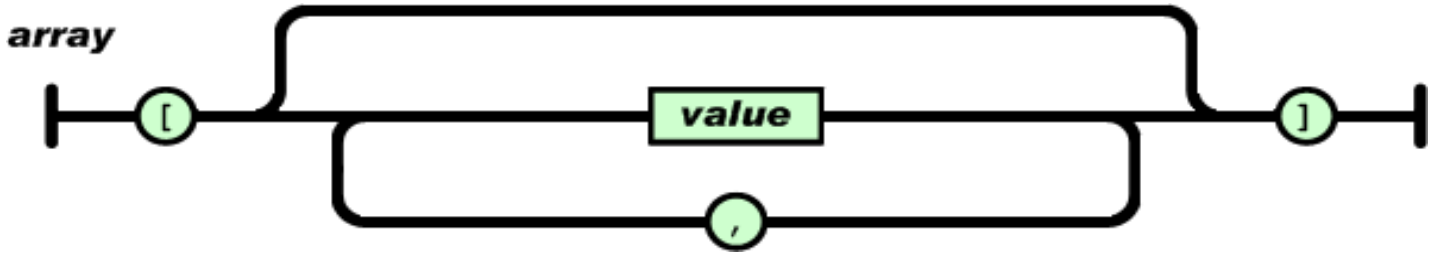
jQuery - JSON




```
var person = {
  “name": "Kant",
  “vorname": "Immanuel",
  “geburtstag": "22.04.1724”
}

alert(person.name);
```

jQuery - JSON




```
var person = [{
   “name": "Kant",
   “vorname" : "Immanuel“
  }, {
   “name" : "Schopenhauer",
   “vorname" : "Arthur",
}];

alert(person[0].name);
alert(person[1].name);
```

jQuery - JSON

- Mit dem eval() Schlüsselwort
- Nachteil: Falls der JSON String Logik enthält, wird diese ausgeführt!

```
var s = '{"name": "Kant", "vorname":"Immanuel"}';
var person = eval('('+s+')');
alert(person.name);
```

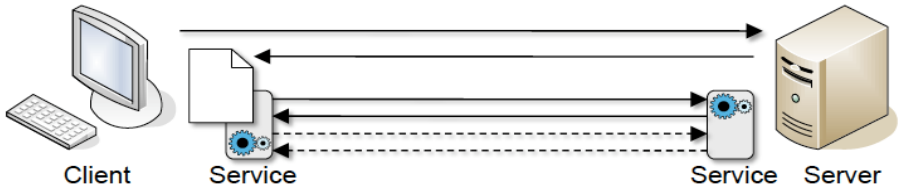
- Mit der JSON Bibliothek
- Wird von modernen Browser unterstützt. Nur Daten, keine Logik!

```
var s = '{"name": "Kant", "vorname":"Immanuel"}';
var person = JSON.parse(s);
alert(person.name);
```

Ajax – Asynchronous JavaScript and XML

jQuery - Ajax

- Was heisst AJAX?
  - Asynchronous JavaScript and XML
- Beispiele von Anwendungen
  - Google Maps, Gmail, Youtube, Facebook,…
- Was ist AJAX?
  - Konzept der asynchronen Datenübertragung zwischen Browser und Server
  - Dies ermöglich HTTP Anfragen durchzuführen, während eine HTML-Seite angezeigt wird. Dabei kann die Seite verändert werden, ohne die Seite komplett neu zu laden (partielle Updates).
  - Verbindet die Technologien HTML, CSS und JavaScript



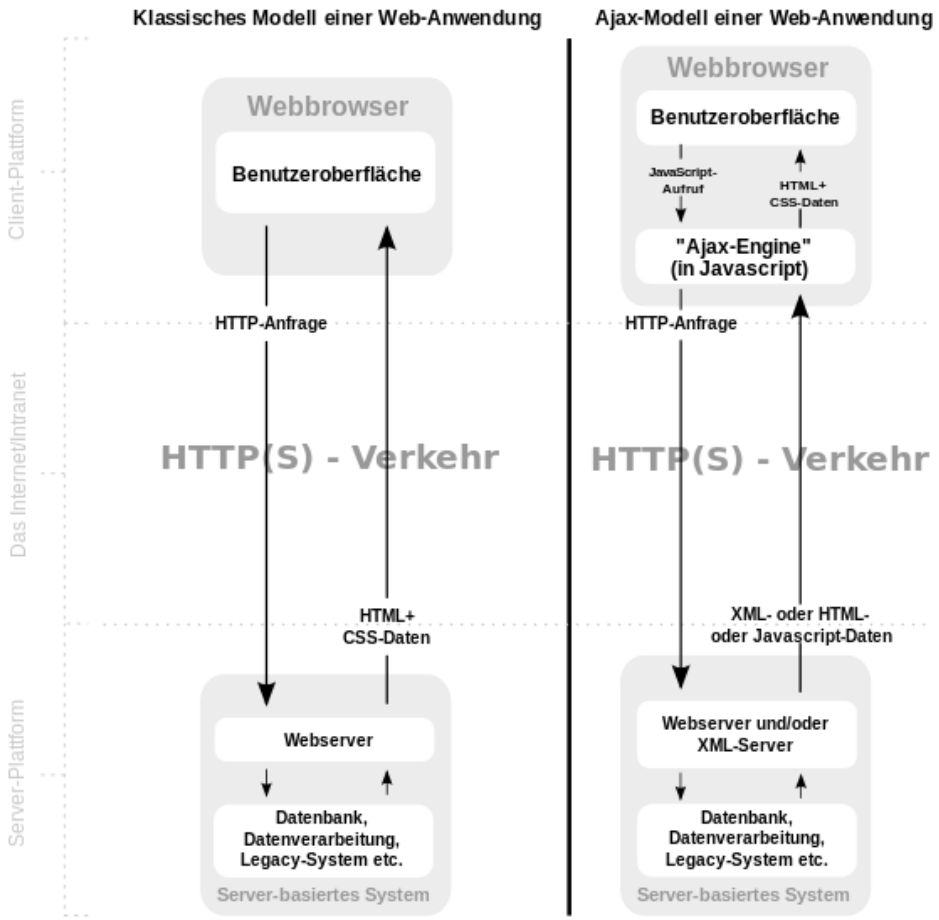
jQuery - Ajax

- Web-Techniken einer AJAX-Anwendung
  - HTML (oder XHTML)
  - Document Object Model (DOM) zur Repräsentation der Daten oder Inhalte
  - JavaScript zur Manipulation des Document Object Models und zur dynamischen Darstellung der Inhalte. JavaScript dient auch als Schnittstelle zwischen einzelnen Komponenten.
  - Das XMLHttpRequest-Objekt, Bestandteil vieler Browser, um Daten auf asynchroner Basis mit dem Webserver austauschen zu können.

jQuery - Ajax

- Datenübertragung
  - Bei der Datenübertragung haben sich verschiedene Verfahren etabliert:
    - reST-ähnliche Verfahren (reStructuredText), um Nutzdaten in Textform zu übertragen
    - JSON (JavaScript Object Notation), ein auf JavaScript zugeschnittenes, textbasiertes Format für Daten und Objekte
    - Diverse proprietäre XML-Formate
    - SOAP (Simple Object Access Protocol), ein Protokoll für Webservices, das meist XML als Austauschformat verwendet

jQuery - Ajax


Klassisches Modell einer Web-Anwendung
Ajax-Modell einer Web-Anwendung
Client-Plattform
Das Internet/Intranet
Server-Plattform
Webbrowser
Benutzeroberfläche
HTTP-Anfrage
HTTP(S) - Verkehr
HTML+
CSS-Daten
Webserver
Datenbank,
Datenverarbeitung,
Legacy-System etc.
Server-basiertes System
Webbrowser
Benutzeroberfläche
JavaScript-
Aufruf
HTML+
CSS-Daten
"Ajax-Engine"
(in Javascript)
HTTP-Anfrage
HTTP(S) - Verkehr
XML- oder HTML-
oder Javascript-Daten
Webserver und/oder
XML-Server
Datenbank,
Datenverarbeitung,
Legacy-System etc.
Server-basiertes System

jQuery - Ajax

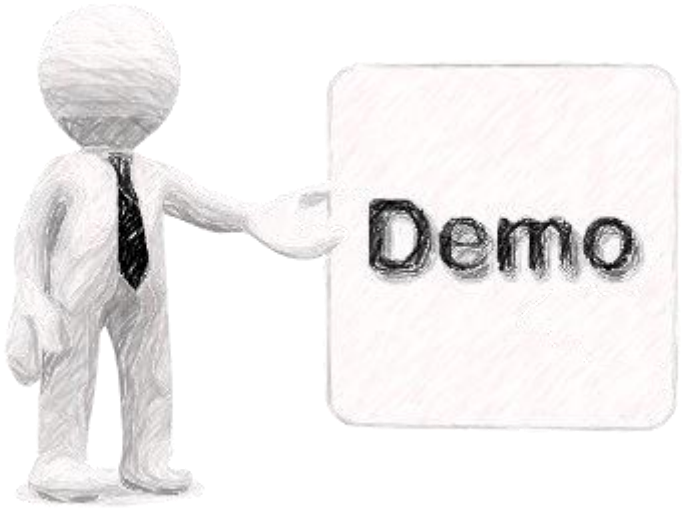
- jQuery bietet eine einfach Möglichkeit um im Hintergrund Anfragen an den Server zu senden
  - Dadurch kann Content der Webseite dynamisch von einem Server nachgeladen werden, ohne das dabei ein «Page Reload» geschieht

```
$.ajax({
  url: "test.html",
  context: document.body
}).done(function() {
  $( this ).addClass( "done" );
});
```

http://www.sitepoint.com/use-jquerys-ajax-function/

jQuery - Ajax

- Demo jQuery Ajax Calls

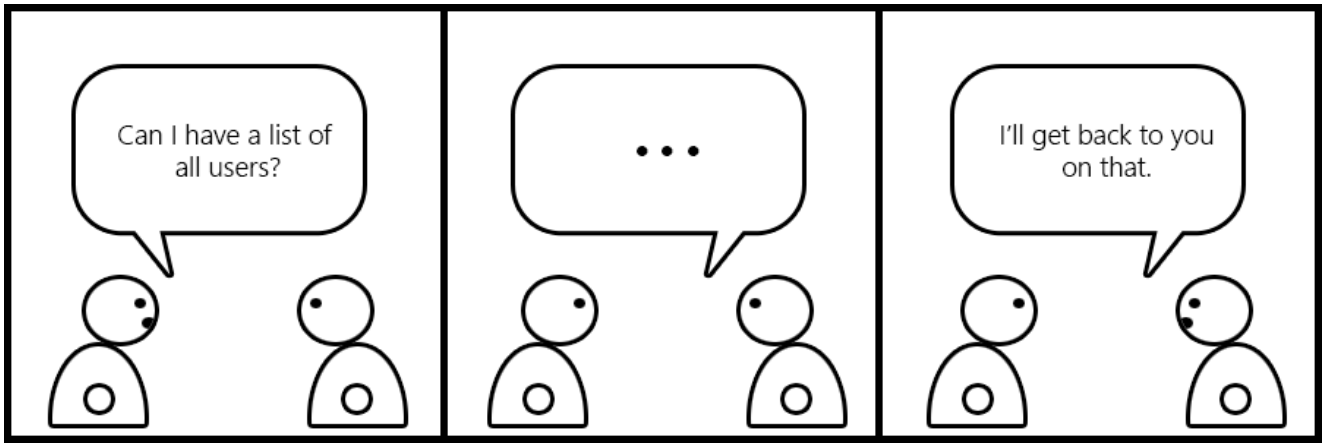


Promises

- In jQuery werden Aufrufe die «quasi» im Hintergrund ausgeführt werden über Promises abgearbeitet
- Ein Promise kann wie die englisches Übersetzung als Versprechen angesehen werden
  - Als Versprechen, dass die Aufgabe welche übergeben wurde gelöst wird und man dann oder im Fehlerfall eine Rückmeldung erhält

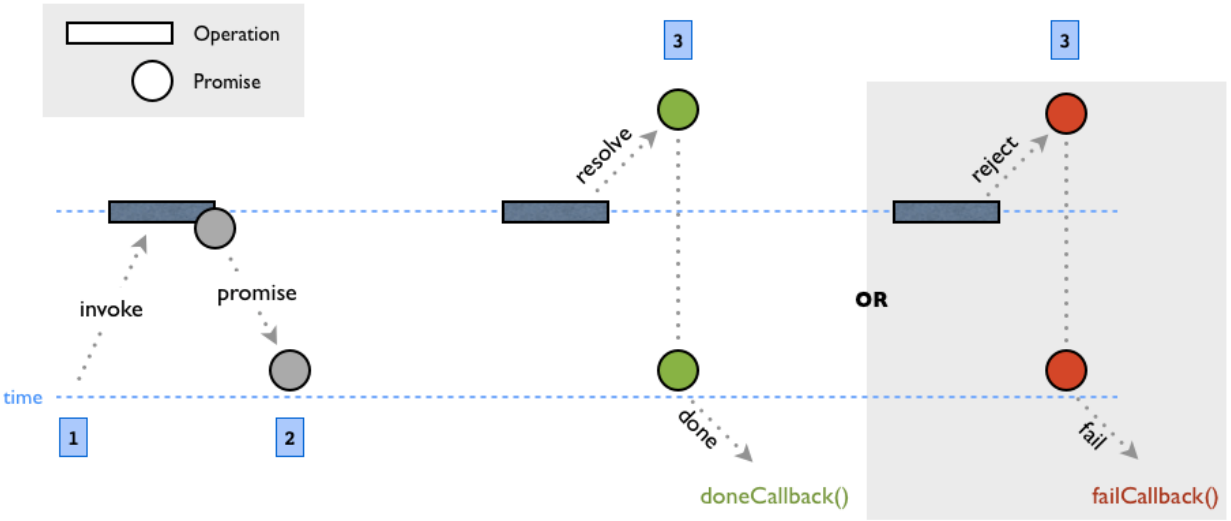
Promises

- Promises können selbst erstellt werden, werden aber auch innerhalb von jQuery oft verwendet
  - Biespielsweise für Ajax Calls, Animationen,…



Promises

- Ein Promise hat folgende API
  - then()
  - always()
  - done()
  - fail()
  - notfiy()



jQuery - Handlebars

- jQuery Handlebars ist ein Framework um dynamische Webseiten zu erstellen
- Mit Handlebars können HTML Templates erstellt werden, welche anschliessend bei bedarf durch JavaScript ein- oder ausgeblendet werden
- Handlebars bietet ausserdem eine einfache Templating Engine an um Client seitig dynamisch Content zu generieren

jQuery - Handlebars

- Der Ablauf um mit Handlebars zu arbeiten ist folgenermassen
  - Handlebar Template(s) in entsprechenden Script Tags erstellen
  - Handlebar Template kompilieren
  - Dem entsprechenden Template einen Kontext übergeben
  - Das durch Handlebars gerenderte HTML im DOM anzeigen lassen

Bildungszentrum Uster
Höhere Berufsbildung
Uster
HBU

jQuery - Handlebars

```
<!-- The #each helper iterates over an array of items. -->
<script id="example-template" type="text/x-handlebars-template">

  <!-- people is looked up on the global context, the one we pass to the compiled template -->

  {{#each people}}

    <!-- Here the context is each individual person. So we can access its properties directly: -->
    <p>{{firstName}} {{lastName}}</p>

  {{/each}}

</script>
```

Bildungszentrum Uster
Höhere Berufsbildung
Uster
HBU

jQuery - Handlebars

```
$(function () {
  // Grab the template script
  var theTemplateScript = $("#example-template").html();

  // Compile the template
  var theTemplate = Handlebars.compile(theTemplateScript);

  // This is the default context, which is passed to the template
  var context = {
    people: [
      { firstName: 'Homer', lastName: 'Simpson' },
      { firstName: 'Peter', lastName: 'Griffin' },
      { firstName: 'Eric', lastName: 'Cartman' },
      { firstName: 'Kenny', lastName: 'McCormick' },
      { firstName: 'Bart', lastName: 'Simpson' }
    ]
  };

  // Pass our data to the template
  var theCompiledHtml = theTemplate(context);

  // Add the compiled html to the page
  $(document.body).append(theCompiledHtml);
});
```

jQuery - Handlebars

- In den Templates gibt es die Möglichkeit über Listen zu iterieren und entsprechend HTML zu rendern
  - Dies kann mit dem #each Helper erreicht werden

```
<h1>Comments</h1>

<div id="comments">
  {{#each comments}}
  <h2><a href="/posts/{{../permalink}}#{{id}}">{{title}}</a></h2>
  <div>{{body}}</div>
  {{/each}}
</div>
```

http://tutorialzine.com/2015/01/learn-handlebars-in-10-minutes/

Übungen

- Übung 1.2 selbständig lösen
- Übung 1.3 selbständig lösen
- Übung 1.4 selbständig lösen

Practice

Recap