



jQuery Advanced



Complexe webapps met jQuery



Tevens besproken:

- `.on()` - event handling, met *contextual selector*. Zie boek p. 112 en verder.
- `var` gebruiken als keyword om functions scoping mee te geven en niet op globale namespace te plaatsen.
- `====` gebruiken in plaats van `==` om type safe checking te doen.
- Function hoisting: functies worden door de JavaScript-compiler verplaatst naar de *top van de outer most function*. Terwijl variabelen (`var`) just-in-time worden gecompileerd.
- Let op de regels voor ASI: best practice: *altijd* zelf je semicolon plaatsen.
- Regels voor CSS3-selectors: [] met + en > . Zie ook http://www.w3schools.com/cssref/css_selectors.asp



Tevens besproken (2)

- jQuery Validation Plugin: alles wat je wilt over validatie. Is vaak handiger (en in ieder geval sneller) dan zelf schrijven:
<http://jqueryvalidation.org/>
- jQuery performance tips & tricks.



Introductie

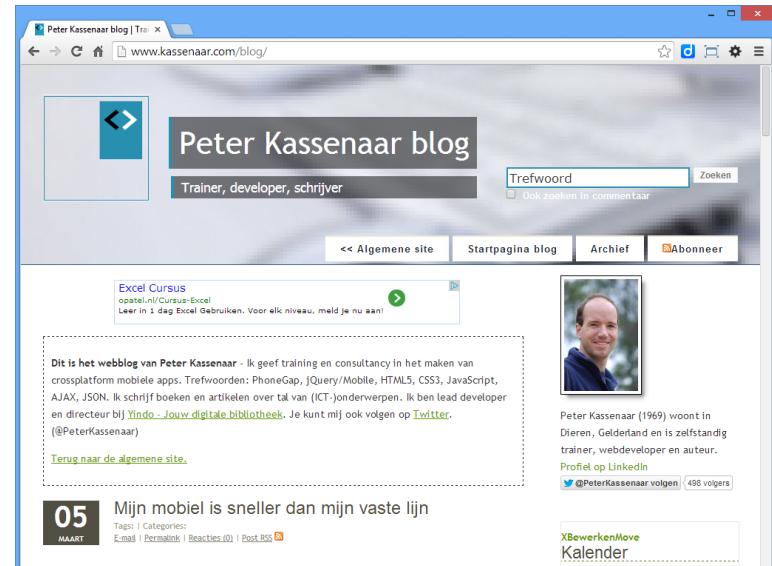
Peter Kassenaar;

- Auteur, developer, docent
- Eigen bedrijf (1996)

www.kassenaar.com/blog

info@kassenaar.com

Twitter: [@PeterKassenaar](https://twitter.com/@PeterKassenaar)



The screenshot shows a blog post titled "Excel Cursus" by Peter Kassenaar. The post discusses cross-platform mobile app development using PhoneGap, jQuery/Mobile, HTML5, CSS3, JavaScript, AJAX, and JSON. It mentions creating books and articles on various ICT topics and being a lead developer and director at Yindo. It also links to his LinkedIn profile and Twitter account. The sidebar features a photo of Peter Kassenaar, information about his location (Dieren, Gelderland), and links to his LinkedIn and Twitter profiles. The footer includes a "Mijn mobiel is sneller dan mijn vaste lijn" section and standard blog navigation links.



Yindo - Jouw digitale bibliotheek

Jouw digitale bibliotheek

Yindo

Start Mijn Yindo Producten Uitgaven Aanmelden

Wat is Yindo? Nieuws Winkelwagen (0 items)

titel / trempoord / auteur / isbn zoek

Computers en internet
Eten en drinken
Geschiedenis en politiek
Gezondheid en psychologie
Hobby, sport en spel
Kind en jeugd
Kunst, cultuur en religie
Literatuur
Management/professioneel
Non-fictie algemeen
Reizen
School en studie
Stripboeken
Stripboeken
Thrillers en fantasy
Tijdschriften
Wonen en tuinieren
Abonnementen

Het familieportret
Bevoren tegoden
Gladiator 1 Vechten voor vrijheid

Jenna Blum
Meukhoff Boekeria
Vanaf € 7,95

Quentin Bates
Karakter
Vanaf € 2,50

Simon Scarrow
Götter Uitgeversgroep
Vanaf € 4,99

meer... meer... meer...

start nieuw populair verras mij

Fact #97

Chuck Norris can solve the Towers of Hanoi in one move.

Like Share

Farmville: Harvest Swap

GRATIS ★★★ (772) Google Play

Play Store

Boekenwolk startpagina

www.boekenwolk.nl

Anmelden

Heb je een code?
Vul in!

START ABONNEMENT INFO

zoek FAVORIETEN

NIEUW SCHRIJVEREN

BOEKENWOLK Kies je leeftijd

BOEKENWOLK

Koningskind
Legenden
Lotgenoten
De legende van de hemelrijders
Een vampier van niks

Web Development Library

www.webdevelopmentlibrary.nl

START TITELS DOWNLOADS CONTACT

Peter Kassenaar
JavaScript

Peter Kassenaar
jQuery Mobile

Peter Kassenaar
AngularJS

Web Development Library

Van Duuren Informatica

Web Development Library

Nederlandstalige kwaliteitsuitgaven over front-end webdevelopment



Over jullie...





Voorkennis webdesign, (mobile) apps?

Voorkennis andere (web-)talen?

Verwachtingen van de cursus?

Concrete projecten?



Materialen

Software

Handouts

Oefeningen

Websites

The screenshot shows a web browser displaying a page from www.vijfhart.nl/oracle-web/opleidingen/oracle-web-cursus-pagina/training-id/1674/jquery-advanced.htm. The page has a red header with the 'vijfhart' logo. The main content area discusses a 'jQuery Advanced' course, mentioning prerequisites like 'jQuery Basis' and various topics such as AJAX, selectors, and performance tips. A large blue button in the center says 'Meer info? Klik hier!' with a small upward arrow icon.

vijfhart®

IT-opleidingen | weblog | over Vijfhart | contact

Vijfhart > Web > Web Development > HTML CSS, JavaScript & PHP

Zoekwoord(en)...

Gerelateerde onderwerpen:
Javascript | Webdesign

Mail deze pagina
Printbare versie van deze pagina
Snel online aanmelden

jQuery Advanced
[Doelgroep](#) | [Voorkennis](#) | [Onderwerpen](#) | [Data & prijzen](#) | [Aanmelden](#) | [Gerelateerd](#)

In de cursus jQuery-advanced bouwt u verder op uw bestaande jQuery-kennis. U leert werken met de meer complexe onderdelen van jQuery zoals AJAX-mogelijkheden en het werken met externe data; zelf user interface-onderdelen maken en eigen plug-ins ontwerpen. Na afloop van de cursus kent u alle ins en outs van dit populaire JavaScript-framework en kunt u jQuery op alle fronten van uw webapplicatie inzetten.

Doelgroep cursus jQuery Advanced
Deze cursus is bedoeld voor meer gevorderde webdevelopers die hun basiskennis van jQuery willen uitbreiden of die niet voldoende hebben aan het bestaande palet aan plug-ins. De cursus is code-georiënteerd en richt zich op developers die webapps moeten bouwen en onderhouden.

Voorkennis
Wij adviseren onderstaande voorkennis:

- De cursus [jQuery Basis](#) of vergelijkbare kennis.
- De cursus [Bouwen van webpagina's met HTML](#) of vergelijkbare kennis.
- De cursus [Cascading Style Sheets \(CSS\)](#) of vergelijkbare kennis.
- Ruime praktijkervaring met verschillende platformen en browsers (Internet Explorer, Firefox, Chrome).
- Kennis van de Engelse taal. Veel websites, documentatie en achtergrondinformatie is alleen in het Engels beschikbaar.

Onderwerpen cursus jQuery Advanced
De cursus jQuery Advanced behandelt de volgende onderwerpen:

- De functie `.each()`
- De functie `.data()`
- Complexe selectors
- JQuery AJAX; achtergronden, mogelijkheden en toepassingen
- De opdrachten `$load()`, `$get()` en `$ajax()`
- Samenwerken met de server, JSON-responses verwerken
- Het deferred object: `$when()`, `$then()`, `$done()`
- JQuery templates gebruiken en ontwikkelen
- Zelf plug-ins ontwikkelen: patterns, parameters, best practices
- JQuery-performance tips
- Praktijkoeferingen: Het geleerde combineren in een eigen site of project

Aanmelden voor cursus jQuery Advanced
Geïnteresseerd geraakt in deze opleiding, gegeven door ervaren docenten?



Indeling

- Geschiedenis - plaatsbepaling – context
- Webapps met jQuery
- In detail:
 - Selectors, .each(), .data(), .extend() en meer.
 - Werken met Ajax
 - Deferred objects
 - Plug-ins ontwikkelen
 - ...



In een notendop

Selectors &
(complexe) API
functions

AJAX-functies
en deferred
objects

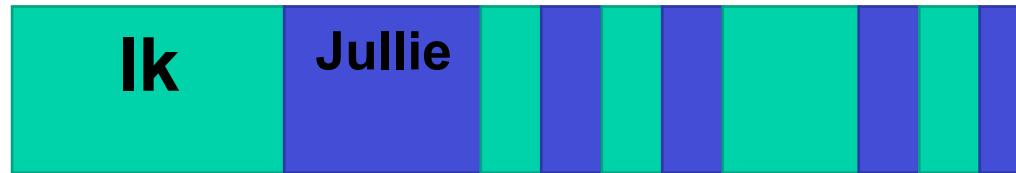
Eigen inbreng,
vragen,
oefeningen, ...

Plug-in
development

Performance
tips & tricks



Werkwijze



...



Vragen?



“jQuery makes JavaScript suck less...”

“jQuery is what the HTML DOM should have been!”





What is jQuery?

- *Javascript HTML DOM Manipulation library*
- Light weight (~80k compressed, all modules)
- Powerful and extremely practical functionality
- Easy to learn and intuitive to use
- *CSS 3 based selector syntax*
 - ID's, Classes, Elements
- Custom Event handling
- Easily to extend through plug-in API



Decompose jQuery statements

selector	action	parameters
jQuery('p')	.css	('color', 'blue');
\$('p')	.css	('color', 'blue');

- *selector* is always `jQuery`, or it's alias `$`
- *Action* is one of the `jQuery API` commands
- *parameters* a list of 0, 1, or more arguments
 - Comma-separated
 - JSON-notation



The Document Object Model

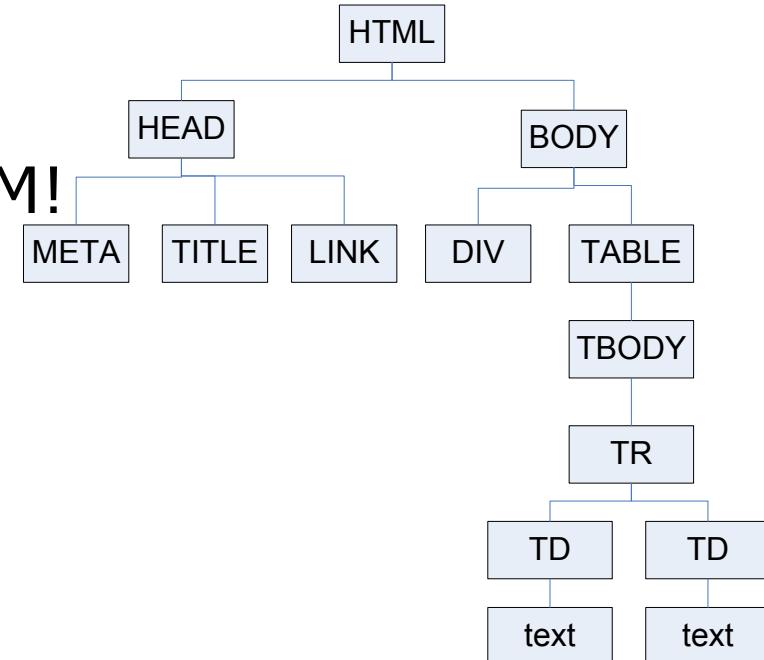
What to select?

- Elements in the DOM!

Use CSS-syntax

to select elements

- elements/tags
- classes
- id's





Always make sure the Page is ready

- All elements needs to be loaded before we can manipulate them
- Always use a document-ready function

```
$ (document) . ready (function () {  
    alert ('Document is ready...') ;  
    // other functions, hooks, etc.  
}) ;
```

In plain JavaScript: `window.onload=function () {...}`



The jQuery API

- All options for `.action()` are described in the *jQuery API*
- Learn and use!

The screenshot shows the official jQuery API documentation page. At the top, there's a navigation bar with links for Download, API Documentation (which is currently selected), Blog, Plugins, and Browser Support. To the right of the navigation is a search bar. The main content area has a sidebar on the left containing a table of contents with categories like Ajax, Attributes, Core, CSS, Data, Deferred Object, Deprecated, and Dimensions. To the right of the sidebar, several API method cards are listed, each with a title, a brief description, and a link to its detailed documentation. The first few cards shown are `.add()`, `.addBack()`, `.addClass()`, and `.after()`.

<http://api.jquery.com/>



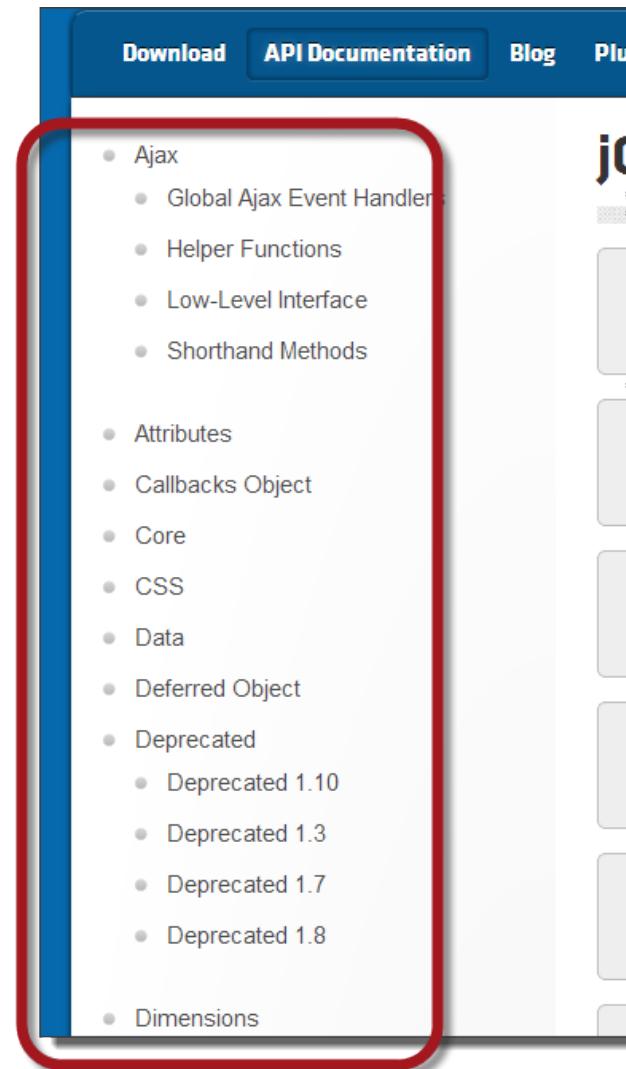
API categories

Most used:

- Ajax
- Attributes
- CSS
- Data
- DOM
- Effects
- Events
- Forms

Supportive

- Utilities
- Dimensions
- Offset
- ...





Enkele complexe jQuery methods

Complexe selectie, .data(), .each() en .extend()



Complex selection

Select on attributes:

- `$("[href='#']")` → alle tags met een href-attribuut dat gelijk is aan #
- `$("[src$='.jpg']")` → alle tags met een src-attribuut eindigt op .jpg



Select on position within an element:

- `$(“ul li:first”)` → alle eerste li-tags in elke lijst
- `$(“tr td:last”)` → elke laatste cel in tabelrij
- `$(“tr td:eq(2) ”)` → elke tweede cel binnen tabelrij
- `$(“tr:even”)` → elke even tabelrij
- `$(“ul li:odd”)` → elke oneven li-tag binnen een ul



Select form elements

- `$(":text")` → alle input elementen van het type tekst
- `$(":password")` → alle input elementen van het type password
- `$(':selected')` → alle elementen die geselecteerd zijn
- `$(':checked')` → alle selectievakjes die geselecteerd zijn
- `$(':visible')` → alle zichtbare elementen



Selecting specific elements

- Child elements only
 - `$('#nav li > a');`
 - `$('#nav li').children();`
 - `$('#nav li').children('a'); // filter specific type of child`
- Terminologie:
 - Descendant: een compleet element, inclusief child elements van een parent element
 - Child : alleen het eerste child-element

```
<ul id="nav">
  <li><a href="#">Item 1</a></li>
  <li><a href="#"><span>Item 2</span></a></li>
  <li><a href="#">Item 3</a></li>
  <li><a href="#">Item 4</a></li>
  <li><a href="#">Item 5</a></li>
</ul>
```



Reference: cheat sheets

- zoek op trefwoord 'jQuery Cheat Sheet'

JQUERY 1.5
VISUAL CHEAT SHEET

★ = NEW IN JQUERY 1.5 / f(x) = FUNCTION / a = ARRAY / jQ = JQUERY / El = ELEMENT / 0-1 = BOOLEAN / Obj = OBJECT / Num = NUMBER / Str = STRING

SELECTORS * CORE * ATTRIBUTES * CSS * TRAVERSING * MANIPULATION * EVENTS * EFFECTS * AJAX * UTILITIES * DEFERRED OBJECT

Designed by Antonio Lupetti © 2011 • http://woorkup.com • http://twitter.com/woork | jQuery is © of John Resig and the jQuery Team.

SELECTORS / 1. BASIC	SELECTORS / 2. HIERARCHY	SELECTORS / 3. BASIC FILTER	SELECTORS / 4. CONTENT FILTER	SELECTORS / 5. ATTRIBUTE	SELECTORS / 6. CHILD FILTER	SELECTORS / 7. VISIBILITY FILTER	SELECTORS / 8. FORM	file Selector	image Selector	input Selector	password Selector	radio Selector	reset Selector	selected Selector	submit Selector	text Selector	.get([index])	.index()	.length	.selector	.size()	.toArray()	.CORE / 1. THE JQUERY FUNCTION	.queue([queueName], newQueue)	.data(obj)	.removeData([name])	.dequeue([queueName])	.CORE / 4. INTEROPERABILITY	jQuery.fn.extend(object)	jQuery.extend(object)	
All Selector ("") Selects all elements.	Child Selector ("parent > child") Selects all direct child elements specified by "child" of elements specified by "parent".	:animated Selector	:header Selector Selects all elements that are headers, like h1, h2, h3 and so on.	:last Selector Selects the last matched element.	:not() Selector Select all elements at an index less than index within the matched set.	:name!=value Selects elements that either don't have the specified attribute, or do have the specified attribute but not with a certain value.	:name^=value Selects elements that have the specified attribute with a value beginning exactly with a given string.	:name[value][name2=value2] Matches elements that match all of the specified attribute filters.	:first-child Selector Selects all elements that are the first child of their parent.	:last-child Selector Selects all elements that are the last child of their parent.	:nth-child Selector Selects all elements that are the nth-child of their parent.	:only-child Selector Selects all elements that are the only child of their parent.	:selected Selector Selects all elements that are selected.	:submit Selector Selects all elements of type submit.	:text Selector Selects all elements of type text.	.get([index]) Retrieve the DOM elements matched by the jQquery object.	.index() Search for a given element from among the matched elements.	.length The number of elements in the jQquery object.	.selector A selector representing selector originally passed to jQquery().	.size() Return the number of DOM elements matched by the jQquery object.	.toArray() Retrieve all the DOM elements contained in the jQquery set, as an array.	jQuery()	.queue([queueName], newQueue)	.data(obj)	.removeData([name])	.dequeue([queueName])	jQuery.when()	jQuery.fn.extend(object)	jQuery.extend(object)		
Class Selector (*.class) Matches all elements with the given name.	Descendant Selector ("ancestor descendant") Selects all elements that are descendants of a given ancestor.	:animated Selector	:last Selector Selects the last matched element.	:lt(i) Selector Select all elements at an index less than index within the matched set.	:odd Selector Selects all elements that do not match the given selector.	:name!=value Selects elements that either don't have the specified attribute, or do have the specified attribute but not with a certain value.	:name^=value Selects elements that have the specified attribute with a value beginning exactly with a given string.	:name[value][name2=value2] Matches elements that match all of the specified attribute filters.	:first-child Selector Selects all elements that are the first child of their parent.	:last-child Selector Selects all elements that are the last child of their parent.	:nth-child Selector Selects all elements that are the nth-child of their parent.	:only-child Selector Selects all elements that are the only child of their parent.	:selected Selector Selects all elements that are selected.	:submit Selector Selects all elements of type submit.	:text Selector Selects all elements of type text.	.get([index]) Retrieve the DOM elements matched by the jQquery object.	.index() Search for a given element from among the matched elements.	.length The number of elements in the jQquery object.	.selector A selector representing selector originally passed to jQquery().	.size() Return the number of DOM elements matched by the jQquery object.	.toArray() Retrieve all the DOM elements contained in the jQquery set, as an array.	jQuery()	.queue([queueName], newQueue)	.data(obj)	.removeData([name])	.dequeue([queueName])	jQuery.when()	jQuery.fn.extend(object)	jQuery.extend(object)		
Element Selector ('element') Selects all elements with the given tag name.	Next Adjacent Selector ("prev + next") Selects all next elements matching "next" that are immediately preceded by a sibling "prev".	:name=*=value Selects elements that have the specified attribute with a value containing the a	:header Selector Selects all elements that are headers, like h1, h2, h3 and so on.	:last Selector Selects the last matched element.	:not() Selector Select all elements at an index less than index within the matched set.	:name!=value Selects elements that either don't have the specified attribute, or do have the specified attribute but not with a certain value.	:name^=value Selects elements that have the specified attribute with a value beginning exactly with a given string.	:name[value][name2=value2] Matches elements that match all of the specified attribute filters.	:first-child Selector Selects all elements that are the first child of their parent.	:last-child Selector Selects all elements that are the last child of their parent.	:nth-child Selector Selects all elements that are the nth-child of their parent.	:only-child Selector Selects all elements that are the only child of their parent.	:selected Selector Selects all elements that are selected.	:submit Selector Selects all elements of type submit.	:text Selector Selects all elements of type text.	.get([index]) Retrieve the DOM elements matched by the jQquery object.	.index() Search for a given element from among the matched elements.	.length The number of elements in the jQquery object.	.selector A selector representing selector originally passed to jQquery().	.size() Return the number of DOM elements matched by the jQquery object.	.toArray() Retrieve all the DOM elements contained in the jQquery set, as an array.	jQuery()	.queue([queueName], newQueue)	.data(obj)	.removeData([name])	.dequeue([queueName])	jQuery.when()	jQuery.fn.extend(object)	jQuery.extend(object)		
ID Selector ("#id") Selects a single element with the given id attribute.	Multiple Selector ("selector1, selector2, selectorN") Selects the combined results of all the specified selectors.	:name*=value Selects elements that have the specified attribute with a value containing the a	:last Selector Selects the last matched element.	:lt(i) Selector Select all elements at an index less than index within the matched set.	:odd Selector Selects odd elements, zero-indexed. See also even.	:name!=value Selects elements that either don't have the specified attribute, or do have the specified attribute but not with a certain value.	:name^=value Selects elements that have the specified attribute with a value beginning exactly with a given string.	:name[value][name2=value2] Matches elements that match all of the specified attribute filters.	:first-child Selector Selects all elements that are the first child of their parent.	:last-child Selector Selects all elements that are the last child of their parent.	:nth-child Selector Selects all elements that are the nth-child of their parent.	:only-child Selector Selects all elements that are the only child of their parent.	:selected Selector Selects all elements that are selected.	:submit Selector Selects all elements of type submit.	:text Selector Selects all elements of type text.	.get([index]) Retrieve the DOM elements matched by the jQquery object.	.index() Search for a given element from among the matched elements.	.length The number of elements in the jQquery object.	.selector A selector representing selector originally passed to jQquery().	.size() Return the number of DOM elements matched by the jQquery object.	.toArray() Retrieve all the DOM elements contained in the jQquery set, as an array.	jQuery()	.queue([queueName], newQueue)	.data(obj)	.removeData([name])	.dequeue([queueName])	jQuery.when()	jQuery.fn.extend(object)	jQuery.extend(object)		



De functie `$(()).each()`

- Een functie uitvoeren voor elk van de elementen in de selectie
- Nodig: als je *meerdere* opdrachten wilt gebruiken voor elementen binnen een set
- Notatie: `$('#selectie').each(function(i, el){...});`
- Parameters
 - `i`: index (nummer) van het element
 - `el`, het betreffende element waarop de `each` van toepassing is.
 - Dus: `el === this`, binnen de functie.



Bijvoorbeeld: hyperlinks selecteren → tooltip instellen
→ click()-event instellen → *en* speciale opmaak
toekennen

```
// state van de accordion met categorieën herstellen
var categoryID = localStorage.getItem('categoryID');
if (categoryID) {
    $('#accordion h2').each(function () {
        if ($(this).data('categoryid') == categoryID) {
            $(this).click();
        }
    });
}
```

<http://api.jquery.com/each/>



De functie .data()

- Werkt samen met HTML5-attribuut `data="..."`
- Zowel getter als setter
- HTML:

```
<div data-customid="5">Data-div met ID 5</div>
```

Script
`$(‘div’).data(‘customid’)`

.data(key) *Returns: Object*

Description: Return the value at the named data store for the first element in the jQuery collection, as set by `data(name, value)` or by an HTML5 `data-*` attribute.

↳ .data(key) version added: 1.2.3

key
Type: `String`
Name of the data stored.

<http://api.jquery.com/data/>



De functie .extend()

- Inhoud van twee of meer objecten samenvoegen in het eerste object
- In de praktijk: vooral bij plug-ins, meegegeven configuratieobject verwerken

```
var fruit = {  
    appel: 5,  
    banaan: 10,  
    peer: 3  
}  
var meerFruit = {  
    kiwi: 13,  
    mango: 8  
}
```

```
$.extend(fruit, meerFruit);
```



jQuery AJAX

AJAX-functies en deferred objects



AJAX-opties

<http://api.jquery.com/category/ajax/>

The screenshot shows a web browser window displaying the jQuery API documentation for the 'Ajax' category. The URL in the address bar is <http://api.jquery.com/category/ajax/>. The page title is 'Ajax - jQuery API'. The main content area is titled 'Ajax' and contains a brief introduction: 'The jQuery library has a full suite of AJAX capabilities. The functions and methods therein allow us to load data from the server without a browser page refresh.' Below this, several jQuery methods are listed:

- jQuery.ajax()** [Low-Level Interface](#)
Perform an asynchronous HTTP (Ajax) request.
- .ajaxComplete()** [Global Ajax Event Handlers](#)
Register a handler to be called when Ajax requests complete. This is an [Ajax Event](#).
- .ajaxError()** [Global Ajax Event Handlers](#)
Register a handler to be called when Ajax requests complete with an error. This is an [Ajax Event](#).
- jQuery.ajaxPrefilter()** [Low-Level Interface](#)
Handle custom Ajax options or modify existing options before each request is sent and before they are processed by `$.ajax()`.
- .ajaxSend()** [Global Ajax Event Handlers](#)
Attach a function to be executed before an Ajax request is sent. This is an [Ajax Event](#).
- jQuery.ajaxSetup()** [Low-Level Interface](#)
Set default options for all future Ajax requests.

On the left side, there is a sidebar with two sections: 'jQuery API' and 'Browse the jQuery API'. The 'jQuery API' section includes links for 'New or Changed in 1.7', 'Raw XML API Dump', 'Dynamic API Browser', and 'jQuery API Book'. The 'Browse the jQuery API' section lists categories such as 'All', '- Ajax', 'Global Ajax Event Handlers', 'Helper Functions', 'Low-Level Interface', 'Shorthand Methods', 'Attributes', 'Callbacks Object', 'Core', 'CSS', 'Data', and 'Deferred Object'.



Ajax-methods in jQuery

- .load()
- Denk aan standaard jQuery-werkwijze:
 - Selecteer element met \$ (...)
 - roep method .load() aan
 - gebruik success callback-functie

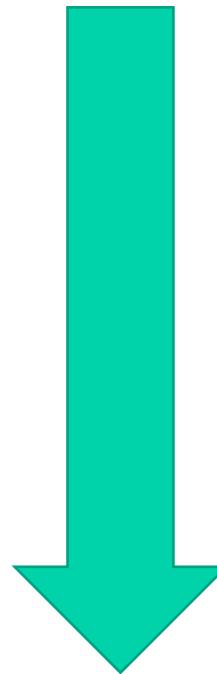


Meer AJAX-functies

- `$.getJSON()`
- `$.get()`
- `$.post()`
- `$.ajax()`

Eenvoudig

Complex





\$.getJSON()

Shorthand method voor ophalen van JSON-gegevens

Parameters: url, data, callbackfunction

```
// de klik op de knop omzetten in een Ajax-request
function handleClick(event) {
    // 1. variabelen instellen
    $.getJSON('ajaxData.json', // dit is de URL waar de call naar toe gaat
              {}, // Het object dat wordt meegestuurd (in dit geval leeg)
              function (data) {
                  // Callback functie bij succes.
                  // Over het object loopen en tonen in de UI
                  console.log(data)
                  for (var i in data) {
                      $('#myDiv').append('<p>' + i + ': <strong>' + data[i] + '</strong></p>');
                  }
              });
} // einde function handleClick
```



\$.get()

- Voert een standaard GET-request uit naar de server

```
// de klik op de knop omzetten in een Ajax-request
function handleClick(event){  
    // 1. $.get() aanroepen. URL als parameter meegeven, callbackfunctie voor resultaat  
    $.get('simple.html', function (result) {  
        $('#myDiv').html(result);  
    })  
}
```



\$.post()

- Voert een standaard POST-request uit
- Vaak gebruikt voor meesturen gegevens

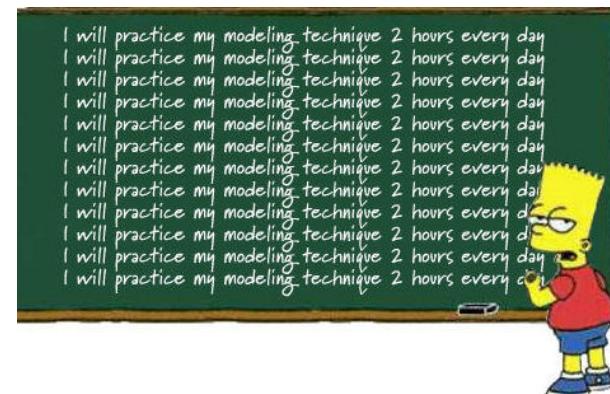
```
// de klik op de knop omzetten in een Ajax-request
function handleClick(event) {
    // 1. data-object maken van tekstveld
    var data = {};
    data.name = $('#txtName').val();
    // 2. $.post() aanroepen en data meegeven. Op de server moet iets v
    $.post('simple.html', data, function (result) {
        $('#myDiv').html(result);
    })
}
```

Kijk in de Developer tools voor details Network Request
(zoals request-header en Form Data)



Oefening

- Schrijf zelf jQuery `$.get()` en `$.post()`-opdrachten
- Retourneer eenvoudige tekst, of haal 'echte' data op.
- Kijk ook naar `$.getJSON()`
- Documentatie op
<http://api.jquery.com>





Generic workhorse: `$.ajax()`

- Alle parameters zelf instellen
- Meeste flexibiliteit
- Configuratieobject meegeven

```
13  function handleClick(event){ 10
14      // 1. variabelen instellen
15      $.ajax({
16          dataType: 'json',
17          contentType: 'application/json',
18          url : 'ajaxData.json',
19          success: handleData,    // success callbackfunctie
20          error : handleError   // error callbackfunctie
21      });
22  }
```



Parameters voor `$.ajax()`

url :	Adres waar call naar toe gaat
type:	GET- of POST-request
contentType :	Gegevenstype voor verzenden
succes:	Callback bij succes
error:	Callback bij optreden van fout
beforesend :	Functie voorafgaand aan verzenden
data:	De gegevens die worden meegezonden

Gebruikelijk: meezendende in configuratie-object {..}



jQuery.ajax()

Categories: [Ajax](#) > [Low-Level Interface](#)

jQuery.ajax(url [, settings])

Returns: [jqXHR](#)

Description: Perform an asynchronous HTTP (Ajax) request.

↳ **jQuery.ajax(url [, settings])**

version added: 1.5

url

Type: [String](#)

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

settings

Type: [PlainObject](#)

A set of key/value pairs that configure the Ajax request. All settings are optional. A default can be set for any option with `$.ajaxSetup()`. See [jQuery.ajax\(settings \)](#) below for a complete list of all settings.

↳ **jQuery.ajax([settings])**

version added: 1.0

settings

Type: [Plain Object](#)

A set of key/value pairs that configure the Ajax request. All settings are optional. A default can be set for any option with `$.ajaxSetup()`.

accepts (default: depends on `DataType`)

Type: [PlainObject](#)

The content type sent in the request header that tells the server what kind of response it will accept in return. If the `accepts` setting needs modification, it is recommended to do so once in the `$.ajaxSetup()` method.

async (default: `true`)

Type: [Boolean](#)

By default, all requests are sent asynchronously (i.e. this is set to `true` by default). If you need synchronous requests, set this option to `false`. Cross-domain requests and `dataType: "jsonp"`

<http://api.jquery.com/jQuery.ajax/>



Ajax-instellingen instellen

Standaard Ajax-instellingen:

- `$.ajaxSetup();`

Basisinstellingen voor :

- type (GET, POST)
- contentType
- error-callback
- ...

```
1 // Generieke instellingen voor jQuery- AJAX-calls
2 $.ajaxSetup({
3   type: 'POST',
4   contentType: 'application/json',
5   error: function (xhr, status) {
6     alert('Er is een jQuery/AJAX-fout opgetreden : ' + xhr.responseText);
7   }
8 });
9 
```

<http://api.jquery.com/jQuery.ajaxSetup/>



Jsonp via \$.ajax()

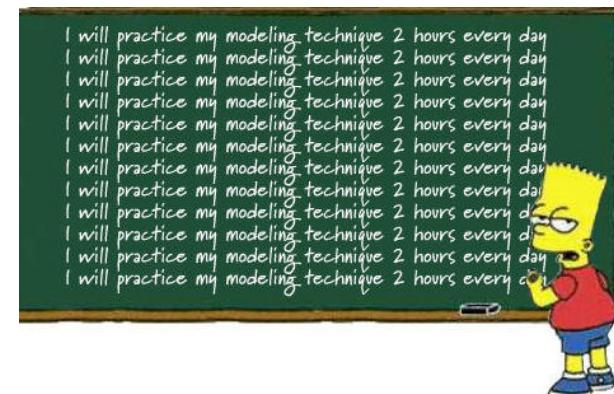
- Voor Cross-domain calls
- Parameter: dataType : 'jsonp'
- Achter de schermen regelt jQuery juiste callback, etc.

```
12 // de klik op de knop omzetten in een Ajax-request
13 function handleClick(event){ 10
14     // 1. variabelen instellen
15     $.ajax({
16         dataType: 'jsonp',
17         contentType: 'application/json',
18         url : 'http://api.yindo.com/api/214f032c-de61-4b86-87d2-8b1f7f5a7ca9/book/new/10/1/10',
19         success: handleData,      // success callbackfunctie
20         error : handleError    // error callbackfunctie
21     });
22 }
```



Oefening

- Schrijf zelf jQuery `$.ajax()`-opdrachten
- Retourneer eenvoudige tekst, of haal 'echte' data op.
- Gebruik het configuratie-object `url`, `dataType`, `success`, etc.





Defferred object

`$.when()` en `$.then()`



Deffered object

- Als een resultaat van meerdere ajax-calls afhankelijk is.
- Of: als meerdere ajax-calls in chain moeten worden uitgevoerd
- Doel: betere leesbaarheid, meer modulaire opbouw, betere foutafhandeling.
- Gebruik jQuery `$.when()` en `$.then()`
- jQuery 1.8+



Download API Documentation

Blog Plugins Browser Support

Search



- Ajax
 - Global Ajax Event Handlers
 - Helper Functions
 - Low-Level Interface
 - Shorthand Methods
- Attributes
- Callbacks Object
- Core
- CSS
- Data
- Deferred Object
- Deprecated
 - Deprecated 1.10
 - Deprecated 1.3
 - Deprecated 1.7
 - Deprecated 1.8
- Dimensions
- Effects
- Basics

Category: Deferred Object

The Deferred object, introduced in jQuery 1.5, is a chainable utility object created by calling the [`jquery.Deferred\(\)`](#) method. It can register multiple callbacks into callback queues, invoke callback queues, and relay the success or failure state of any synchronous or asynchronous function.

The Deferred object is chainable, similar to the way a jQuery object is chainable, but it has its own methods. After creating a Deferred object, you can use any of the methods below by either chaining directly from the object creation or saving the object in a variable and invoking one or more methods on that variable.

[`deferred.always\(\)`](#)

Add handlers to be called when the Deferred object is either resolved or rejected.

[`deferred.done\(\)`](#)

Add handlers to be called when the Deferred object is resolved.

[`deferred.fail\(\)`](#)

Add handlers to be called when the Deferred object is rejected.

[`deferred.isRejected\(\)`](#)

Also in: Deprecated > Deprecated 1.7

Determine whether a Deferred object has been rejected.

<http://api.jquery.com/category/deferred-object/>



Case – 3 ajax calls

jQuery deffered object - 01

`$.get() tekst`



Klaar!

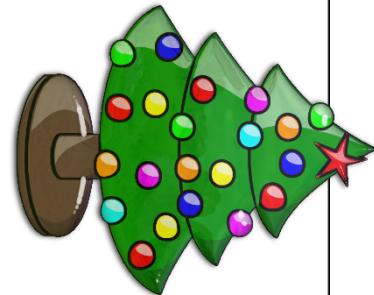


Demo



```
// Event handler
function handleClick(event) {
    // 1. $.get() aanroepen. URL als parameter meegeven, callbackfunctie voor resultaat
    $.get('simpleText.txt', function (result) {
        $('#myDiv1').html(result);
    });
    // 2. maar wat nu als een van de calls heel lang gaat duren... (simuleer met setTimeout())
    setTimeout(function () {
        $.get('simpleText.txt', function (result) {
            $('#myDiv2').html(result);
        })
    }, 2500);
    // 3. derde call + updaten UI
    $.get('simpleText.txt', function (result) {
        $('#myDiv3').html(result);
    });
}

// Event handler
function handleClick(event) {
    // 1. $.get() aanroepen. URL als parameter meegeven, callbackfunctie voor resultaat
    $.get('simpleText.txt', function (result) {
        $('#myDiv1').html(result);
        // 2. tweede call (traagheid gesimuleerd met setTimeout())
        setTimeout(function () {
            $.get('simpleText.txt', function (result) {
                $('#myDiv2').html(result);
                // 3. derde call + updaten UI
                $.get('simpleText.txt', function (result) {
                    $('#myDiv3').html(result);
                    $('.result').addClass('klaar');
                })
            })
        }, 2500);
    });
}
```





Deferred Methods

- **Some methods available on deferred objects**

- Return deferred object for a process flow
 - when
 - promise
 - Attach handler functions to deferred object
 - then
 - done
 - fail
 - progress
 - always
 - Change state of deferred object
 - resolve
 - reject



Ander voorbeeld van deferreds

Case – afhankelijkheid van meerdere events

<http://stackoverflow.com/questions/10945643/correct-way-of-using-jquery-mobile-phonegap-together>

You can use deferred feature of JQuery.

115 var deviceReadyDeferred = \$.Deferred();
 var jqmReadyDeferred = \$.Deferred();

 document.addEventListener("deviceReady", deviceReady, false);

 function deviceReady() {
 deviceReadyDeferred.resolve();
 }

 \$(document).one("mobileinit", function () {
 jqmReadyDeferred.resolve();
 });

 \$.when(deviceReadyDeferred, jqmReadyDeferred).then(dowhenBothFrameworksLoaded);

 function dowhenBothFrameworksLoaded() {
 // TBD
 }



Oefening

- Leer werken met het deferred object
- Schrijf een chain ajax-calls en resolve deze met `.then()`
- Maak foutafhandeling met `.fail()`
- Documentatie:
<http://api.jquery.com/category/deferred-object/>

