jQuery - selectors, attributes, traversing og manipulation

Jeg vil med denne første artikel fra mig om jQuery komme kort ind på jQuerys vidunderlige verden. Jeg har ikke gjort specielt meget ud af forklaringerne, så vil nok mene det er for lidt mere erfarne brugere.

Men der er mange eksempler - det er jo sjove

Skrevet den **03. Feb 2009** af **martin1000ben** I kategorien **Programmering / JavaScript** |

Javascript kan være noget så kedeligt at arbejde med. Lange koder for simple ting - ofte temmelig uforstående kode.

Hvis du vil lave fede ting i javascript, med temmelig simple koder, som er til at forstå, så er jQuery løsningen.

Hvad er jQuery
jQuery er et såkaldt framework, eller et javascript library, alt efter hvem man spørger :)

Hvorfor jQuery
Som sagt, hvis man er træt af lange koder som

```
document.getElementById()
```

-

```
document.getElementsByTagName()
```

osv osv
Så er jQuery perfekt.

jQuery arbejder ud fra XPath.
XPath er et sprog til at finde rundt i information i XML filer, og da HTML er opbygget ligesom XML, så er XPath faktisk en kanon god måde at finde rundt i et html dokument.

Lidt mere om XPath kan læses her
[LINK]http://www.w3schools.com/xpath/default.asp[/LINK]

De andre javascript frameworks jeg lige kender til arbejder kun ud fra document.getElementById() - og derfor er jeg nu 100% gået fra fx. prototype til jQuery.

Lad os nu lege lidt!
Ja.. det kedelige stuff blev lidt kort, det er jo ikke særlig interessant :)
Så nu skal der eksempler på bordet!

Først lidt HTML, det skal der jo til.

```
<div id="foo">
  <p>Lidt tekst her</p>
  <p class="bar">Ekstra tekst her</p>
```

[/div]

Nu vil vi så gerne gøre teksten inde i p med klassen bar til blå tekst

```
$('div#foo p.bar').css('color','blue');
```

Lad os så ændre farven på det andet p tag, når man trykker på p.bar tagget.

```
$('div#foo p.bar').click(function(){$(this).prev().css('color','blue'); });
```

Vi vil nu tilføje et ekstra p tag nedenunder de 2 andre p tags.

```
$('div#foo').append('<p>Ny tekst</p>');
```

Nu kommer det som jeg synes er noget af det bedste chaining (kædereaktion).
Lad os prøve at lave en masse ting på samme tid.

Først lige lidt kode, så kommer forklaringen bagefter, jeg vil prøve at sætte det pænt op - ved hvert punktum er det en ny funktion.

```
$('div#foo')
  .css('color','blue')
  .attr('title','Titel tekst')
  .next()
  .css('color','green')
  .next()
  .css('color','yellow')
  .append('<p>Ny Tekst</p>')
```

Lav os starte fra toppen..
Vi finder først vores udgangpunkt, som i dette tilfælde er
så laver vi css attributten color om til blue.
.attr betyder at vi indsætter/rediger en html attribut på div tagget. Vi indsætter altså nu title="Titel tekst" i div tagget.
Så finder vi det næste element i XPath, som i dette tilfælde er Lidt tekst her og laver lidt css om.
Så finder vi igen det næste element i XPath stien og sætter lidt css.
Til sidst så indsætter vi et nyt p tag i bunden af

```
tagget.
(Havde vi istedet bruge prepend() så ville vi sætte det nye p tag i toppen.)

Alt i alt, så ser vores HTML faktisk sådan her ud nu
[div]
<div id="foo" style="color:blue;" title="Titel tekst">
  <p style="color:green;">Lidt tekst her</p>
```

```
  <p class="bar" style="color:yellow;">Ekstra tekst her</p>
  <p>Ny Tekst</p>
```

[/div]
(Dette kan ses direkte med firebug til firefox, som er den bedste javascript debugger jeg endnu har set til en browser)

Lad os prøve at finde alle p elementer som ikke har en klasse der er bar
$('p[class!=bar]').css('color','blue');

Faktisk bliver mange af disse metoder at finde elementer på også standard i CSS3.

jQuery og Ajax
Kan jQuery også klare dette...?
Selvfølgelig, ethvert framework med respekt for sig selv har nogle metoder til at gøre Ajax endnu nemmere...
Vil dog sige at jQuerys metode ikke er den bedste, men den fungerer, og er stadig rimelig nem at arbejde med.

Kom nu videre...
Ja ja.. nu skal jeg komme med et par eksempler

Et hurtigt GET kald til example.php

```
$.get('example.php',{data:value},function(){
  // Ajax er returneret med success
});
```

example.php kan så hente $_GET['data'] og få fat i value.

Desværre kan $.get og $.post kun bruge en "success" funktion, så ved fejl sker der ikke noget.

Men det er der selvfølgelig lavet en anden funktion til som kan klare alle forskellige options.

```
$.ajax({
  type: 'GET',
  url: 'example.php',
  data: {data:value},
  success: function(ret) {
    // ret vil indeholde det som kommer fra example.php
  },
  error: function(ret) {
    // Do something with ret
  }
});
```

Vi vil ha mer'
Ja.. hvis ellers denne bliver modtaget godt, så vil jeg i min næste artikel om jQuery skrive lidt om animation og komme ind på jQuery UI, som byder på endnu mere...

Måske vil jeg også komme ind på kodning af plugins, og selvfølgelig lige fyre de plugins jeg bruger mest af.

Disclaimer
Vil lige starte med at sige at jeg ikke har testet noget af det ovenstående, men har dog en hel del erfaring med det, da jeg har siddet som frontendprogrammer hos en større dansk virksomhed.

Om Mig
Navn: Martin
Alder: 25
Bopæl: Sverige
Erfaring:
- 2 år som frontend programmør (Javascript, AJAX, XML)
- 4 år som backend programmør (PHP, XML, MySQL)

Startede min karriere som backend programmør, men har så fundet "lykken" i frontend, synes bare det er så meget dejligere når man straks ser et resultat og kan lege lidt med forskellige animationer til forskellige sjove ting.

Kommentar af olebole d. 04. Apr 2008 | 1

I JavaScript kan - som coderdk også understreger - en identifier ikke begynde med tegnet $ (ifølge ECMA 262). Udover at gøre flittigt brug af innerHTML - som aldrig har været valid i nogen somhelst standard - hedder en af jQuery's centrale funktioner slet og ret $. Med andre ord består jQuery temmelig gennemført af invalid kode. Ikke blot er innerHTML et invalidt levn fra version 4 browserne - dengang der ikke eksisterede bedre metoder til at manipulere DOM-træet - den kan også være overordentlig uhensigtsmæssig at bruge sammen med moderne webkode. Læs evt. artiklen: http://dengodekode.dk/artikler/DOM/no_innerhtml.php . Karakteren får du ikke så meget for din behandling af emnet, men fordi jQuery indeholder så meget slamkode og derfor ikke bør finde vej til seriøse websider. Til coderdk: Artiklen er én lang hyldestsang til et dårligt library. Derfor er det ikke en god artikel, uanset hvor rigtig eller forkert library'ets brug er beskrevet og uagtet artiklens pædagogiske niveau. Tag dig dog nu sammen og anmeld brugerens artikel. Din anmeldelse består af ordene: 'Den er lidt tynd' ... resten er kommentarer til mig. Hvis du er så forhippet på at diskutere med mig, så åben et spørgsmål. Det er da utroligt, ingen her kan læse. I citterer begge sætningen: "The dollar sign is intended for use only in mechanically generated code". Hvad er det, I ikke forstår? Der står jo højt og tydeligt, at $ ikke er tilladt i alm. JS-kode - og I citterer det sågar selv. Hvem af Jer skriver mon nogensinde maskingenereret JS-kode? :D happycow >> Jeg kender ikke din kode, men personlig skriver jeg langt bedre kode, end folkene bag jQuery! Hvad innerHTML angår, så læs dog koden. Så kan I vel ikke være i tvivl om, hvad property'en bruges til. Hvorfor basere sine holdninger på tro, alene? - og tag Jer ikke af jubelglade coderdk, der tydeligvis accepterer, hvad somhelst - bare det er hyped! Han fatter åbenbart ikke, at når den mest centrale funktion i et library er invalid, består library'et 'temmelig gennemført af invalid kode' ... og det er da ellers ret indlysende! Hvad angår '$', så er tegnet helt korrekt _kun_ beregnet til _maskingenereret_ kode. Det er JavaScript-kode i et webdokument ikke! Hvor svært kan det dog være at fatte _så_ simple ting? :D

Kommentar af roenving d. 12. May 2008 | 2

Tjah, jQuery bliver ikke mindre invalidt af at blive anprist ... Og desværre viser denne artikel også noget om kvaliteten på udviklere, bl.a. i 'en større dansk virksomhed'

Kommentar af coderdk d. 02. Apr 2008 | 3

Ok artikel. Den er lidt tynd, men giver da et indblik i hvad man kan med jQuery. $ tilladt hvor som helst i en identifier. Det er MENINGEN den skal bruges til maskingeneret kode. Tag jer ikke af den negative olebole, han har åbenbart set sig sur på et udemærket library, der som mange andre gratis (og andre) ting

har nogle bugs og uhensigtsmæssigheder. Han skriver at det nærmest består af ugyldig kode, men han skriver kun to ting. jQuery er i øvrigt GPL, så føl jer fri til at submitte bugs og rettelser *HINT* ;P Kan i øvrigt anbefale dokumentationen her: http://visualjquery.com/

Kommentar af jokkejensen d. 22. Apr 2008 | 4

jquery er fint, man opnår hurtigt et godt resultat - enig.

Men hvorfor forsøge at dokumentere noget der allerede er dokumenteret ?
http://docs.jquery.com/Main_Page Der gør det det nu noget bedre, og dybere.

Og samtidigt syntes jeg alternativer som YUIblog's library, som er en del bedre, og generelt større skal nævnes.

Om YUI er mere valid kode, vil jeg ikke komme ind på - det sætter jeg mig ikke ind i.

Kommentar af happycow d. 31. Mar 2008 | 5

olebole: Det er egentlig lidt spøjst, når man så slår op i ECMA 262 og finder følgende kommentar:
"The dollar sign ($)
and the underscore (_) are permitted anywhere in an identifier. The dollar sign is intended for use only in mechanically generated code." -- jeg vil vove den påstand og sige du tager ret meget fejl med hensyn til jQuery's brugbarhed. Hvad angår innerHTML giver jeg dig dog ret! Men jeg tror dog (eftersom jeg kun har fundet en compactet udgave af jQuery) at brugen af innerHTML skyldes at jQuery tager hensyn til hvilken browser man bruger... Jeg tror nu ikke nogen der laver noget der fungerer så lækkert som jQuery gør det, ikke tager hensyn til en sådan detalje, tror du ?