Jämförelse av funktionsbibliotek för JavaScript

- En prestandajämförelse av funktionsbibliotek för JavaScript vid de vanligaste DOM-interaktionerna i de vanligaste webbläsarna

Eric Ericsson & Petter Andersson
This thesis is submitted to the School of Computing at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science. The thesis is equivalent to 20 weeks of half time studies.

Contact Information:
Author(s):
Petter Andersson
E-mail: petter89@hotmail.com

Eric Ericsson
E-mail: er.ericsson@gmail.com

University advisor:
Stefan Johansson
COM/ Blekinge Institute of Technology

School of Computing
Blekinge Institute of Technology
SE – 371 79 Karlskrona
Sweden

Internet : www.bth.se/com
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57
SAMMANFATTNING

När traditionella skrivbordsapplikationerna görs om till webbapplikationer, ställer det krav på JavaScript som används mer och mer för att få fram ett responsivt gränssnitt på webben. För att underlätta utvecklingen av JavaScriptapplikationer har ett antal funktionsbibliotek skapats.

I vår studie undersöker vi därför vilket av de två populäraste JavaScriptbiblioteken idag, jQuery och Prototype, som presterar bäst i dagens mest använda webbläsare. Dessa tester har utförts i ett testramverk som vi själva utvecklat för att vara webbläseroberoende och inte kräva något av de bibliotek vi testar. Testerna är uppdelade i fyra testfall som körs 20 gånger för att ge ett mer tillförlitligt resultat. Vi har testat hur varje bibliotek hanterar traversering och manipulation av DOM-trädet, sätter och hämtar stilar och attribut på element i DOM-trädet och hanterar event på element i DOM-trädet.

Testerna visade att biblioteket Prototype presterade bättre på alla utom ett testfall i majoriteten av våra utvalda webbläsare; det enda testfallet där jQuery presterade bättre än Prototype var där DOM-trädet skulle manipuleras. Trots att Prototype inte alls är lika omtalat som jQuery, verkar det vara ett bättre bibliotek att använda till webbapplikationer som ska ha ett interaktivt gränssnitt då det i flertalet av våra tester presterar bättre.
INNEHÅLLSÖVERTECKNING

1 INLEDNING .................................................................................................................. 3
1.1 FORSKNINGSFRÅGA .......................................................................................... 3
1.2 HYPOSES ............................................................................................................. 3
1.3 FRÅGSTÄLLNINGAR .......................................................................................... 3
1.4 MÅL OCH SYFTE .............................................................................................. 3
1.5 MÅLGRUPP ......................................................................................................... 3
1.6 METOD ............................................................................................................... 4
1.7 AVGRÄNSNINGAR ............................................................................................. 4

2 BAKGRUND ............................................................................................................ 5
2.1 HUR FUNGERAR DET .......................................................................................... 6
2.2 DOCUMENT OBJECT MODEL – DOM .................................................................. 6
2.2.1 DOM-Scriptande ............................................................................................ 8
2.2.2 Prestanda och DOM ...................................................................................... 9
2.3 FUNKTIONSbibliotek ......................................................................................... 9
2.3.1 jQuery .......................................................................................................... 10
2.3.2 Prototype ...................................................................................................... 10

3 UTFORMNING AV EXPERIMENT ........................................................................... 11
3.1 TESTMILJÖ ......................................................................................................... 12
3.1.1 Hårdvaruspecifikation ................................................................................. 12
3.1.2 Mjukvara och funktionsbiblioteksversioner ............................................... 12
3.2 TESTFALL LADNINGSTIDER ........................................................................... 13
3.3 TESTFALL TRAVERSERING OCH MANIPULATION .......................................... 13
3.4 TESTFALL ATTRIBUT OCH Stilar ..................................................................... 14
3.5 TESTFALL EVENTRESPONS ............................................................................. 14

4 RESULTAT ............................................................................................................... 15
4.1 RESULTAT TESTFALL LADNINGSTIDER ......................................................... 15
4.2 RESULTAT TESTFALL TRAVERSERING OCH MANIPULATION ......................... 16
4.3 RESULTAT TESTFALL ATTRIBUT OCH Stilar ................................................... 20
4.4 RESULTAT TESTFALL EVENTRESPONS .......................................................... 22

5 DISKUSSION ........................................................................................................... 24
5.1 VAD KAN PÅVERKAT MÅTNINGARNA .............................................................. 26

6 SLUTSATS .............................................................................................................. 28
6.1 FÖRSLAG PÅ FORTSATT FORSKNING ............................................................. 28

7 ORDLISTA .................................................................................................................. 30

8 LITTERATURFÖRTECKNING ............................................................................... 31
8.1 BÖCKER ............................................................................................................. 31
8.2 WEBBSIDOR .................................................................................................... 31

9 BILAGOR .................................................................................................................... 32
9.1 RESULTATTABELLER ....................................................................................... 32
9.1.1 Firefox 4.0.1................................................................................................. 32
9.1.2 Chrome 11.0.696/68 .................................................................................... 34
9.1.3 Safari 5.0.5 .................................................................................................. 36
9.1.4 Opera 11.10 .................................................................................................. 38
9.1.5 Internet Explorer 9.0.8112.1421 ................................................................. 40
9.2 BERÄKNINGAR ................................................................................................ 43
9.3 TESTFALLSKOD ............................................................................................... 44
9.3.1 jQuery Index ................................................................................................. 44
9.3.2 jQuery Traversal .......................................................................................... 45
1 INLEDNING

JavaScript har länge varit en del av var webbutvecklares verktygslåda och dess popularitet har gått upp och ner med åren. I och med satsningarna på modernare JavaScriptmotorer i webbläsarna och att dagens applikationer och dynamiska gränssnitt flyttar ut på webben\(^1\) ställs det högre krav på webbläsarnas sätt att hantera JavaScript. Samtidigt som JavaScript i sig utvecklas genom nya funktionsbibliotek som underlättar arbetet för utvecklarna.


1.1 Forskningsfråga

Vilket av funktionsbiblioteken jQuery och Prototype presterar bäst i olika webbläsare vid olika typer av DOM-interaktioner?

1.2 Hypotes

Vi tror att webbläsaroberoende webbgränssnitt med DOM-interaktioner skrivna i jQuery presterar bättre än samma gränssnitt skrivet i Prototype.

1.3 Frågeställningar

1. Hur påverkar sättet vi laddar in funktionsbiblioteken tiden det tar innan vi kan börja använda gränssnittet?

2. Vilket av biblioteken hanterar DOM-manipulation och traversering snabbast?

3. Vilket av biblioteken hanterar förändring av DOM-attribut och stilar snabbast?

4. Vilket av biblioteken skapar, triggar och tar bort DOM-events snabbast?

1.4 Mål och syfte

Syftet med arbetet är att visa vilka prestandakostnader som finns i ett dynamiskt webbgränssnitt skrivet i JavaScript och som använder sig av DOM-interaktioner. Arbetet visar också hur prestandan skiljer sig mellan olika funktionsbibliotek och webbläsare.

1.5 Målgrupp

Målgruppen för jämförelsen är svenska webbutvecklare. Både de som skriver egna JavaScript hemma och de som utvecklar webbtjänster och framförallt gränssnitt för webben i kommersiellt syfte, kan ha intresse av resultaten.

\(^1\) DOM Scripting, kap. 12
1.6 **Metod**

Undersökningen granskar funktionsbiblioteken jQuery och Prototype i flertalet tester som berör prestanda och responsivitet i webbgränssnitt. Motsvarande Funktionalitet i de båda funktionsbiblioteken ställs mot varandra och tester genomförs i ett flertal olika webbläsare för objektivitet och en mer heltäckande bedömning.

1.7 **Avgränsningar**

Arbetet kommer fokusera på de funktioner i de två funktionsbiblioteken som har störst påverkan på DOM. Vi har valt att undersöka funktioner i jQuery och Prototype då de är de två största funktionsbiblioteken för JavaScript i skrivande stund. (Figur 1) Mätningarna sker i de senaste versionerna av de enligt w3c fem mest använda webbläsarna för tillfället, Internet Explorer, Firefox, Safari, Chrome och Opera. (Figur 2)

![Google Trends](http://www.google.com/trends?q=Yahoo+javascript%2C+Prototype+javascript%2C+Jquery+javascript&ctab=0&geo=all&date=all&sort=0)

![Webbbläser fördelning](http://www.w3schools.com/browsers/browsers_stats.asp)

**Figur 1 - Vanligaste biblioteken baserat på Google Trends**

**Figur 2 - Procentuell fördeland av vanligaste webbläsare i dag**
Bakgrund


När JavaScript standardiserades i ECMA-262 var det baserat på flera liknande tekniker, framförallt Netscapes JavaScript och Microsofts JScript.\(^6\) När språket var standardiserat och webbläsartillverkarna varit delaktiga i utvecklingen av en standardiserad DOM passade tillslut utvecklingsplattformen alla webbläsare. Detta ledde till att webbapplikationerna blev mer dynamiska och gränssnitten kunde förändras utan omladdning av hela sidan. JavaScript har haft ett dåligt rykte under en tid på grund av att scripten blev mer och mer resurskrävande men de exekverades på samma gamla JavaScriptmotorer som inte hängde med i utvecklingen.\(^7\)

Trots att JavaScriptmotorerna i webbläsarna har tagit ett stort kliv i utvecklingen de senaste åren finns det fortfarande problem med prestandan inom JavaScript. Det finns funktioner i språket som inte prestandan i JavaScriptmotorn kan påverka. All DOM-integration i JavaScript och även hämtningen av scriptfilerna från servern till klienten är utanför JavaScriptmotorernas scope.

\(^4\) DOM Scripting, kap. 1
\(^5\) High Performance JavaScript, Preface
\(^7\) High Performance JavaScript, Preface
2.1 Hur fungerar det
Till skillnad från till exempel Java och C++ är inte JavaScript ett språk som kompileras utan det tolkas istället av webbläsaren direkt när det körs. Detta leder till att finns det fel i koden visas dessa när sidan visas i webbläsaren.8 När JavaScripten körs i webbläsaren hindrar det resten av sidan att genereras. Finns det då ett script som körs en längre tid tar det en stund innan något syns i webbläsaren och sedan blir redo att användas av användaren.9 Det webbläsaren använder för att tolka JavaScripten kallas JavaScriptmotor, som Tabell 1 visar har alla webbläsare sin egen JavaScriptmotor.

<table>
<thead>
<tr>
<th>Webbläsare</th>
<th>JavaScriptmotor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mozilla Firefox 4</td>
<td>Jägermonkey</td>
</tr>
<tr>
<td>Google Chrome</td>
<td>V8</td>
</tr>
<tr>
<td>Opera</td>
<td>Carakan</td>
</tr>
<tr>
<td>Safari</td>
<td>Nitro</td>
</tr>
<tr>
<td>Internet Explorer 9</td>
<td>Chakra</td>
</tr>
</tbody>
</table>

Tabell 1

2.2 Document object model – DOM


```html
<html>
  <head>
    <title></title>
  </head>
  <body>
    <h1></h1>
    <p></p>
  </body>
</html>

Exempel kan även representeras som trädet i Figur 3
```

8 DOM Scripting, kap. 2
9 High Performance JavaScript, kap. 1
10 https://wiki.mozilla.org/JaegerMonkey
11 http://code.google.com/p/v8/
12 http://my.opera.com/core/blog/2009/02/04/carakan
Figur 3


I exemplet ovan användes endast en typ av nod. I DOM finns ett antal olika noder för att beskriva olika saker.

Elementnod: En elementnod är de noder som syns i Exempel 1. De har en motsvarande tagg i HTML- och XML-dokument. En elementnod kan innehålla andra noder även andra elementnoder. Vilket gör att den karaktäristiska trädstrukturken kan byggas upp.15

Textnod: En textnod är en annan typ av nod som är vanlig i HTML-dokument. Den används för att hålla ren text. Det är vanligt att den är ett barn till, till exempel elementnoden <p> för att hålla dess text.16


15 DOM Scripting, kap. 3
16 DOM Scripting, kap. 3
17 DOM Scripting, kap. 3
2.2.1 DOM-Scriptande

För att kunna dra nytta av DOM-trädets egenskaper behöver man kunna välja ut noder samt kunna förflytta sig i trädet. Detta kan man göra på flera olika sätt. För att välja ut en bestämd elementnod kan man använda sig av HTML-attributet ID. Alltså samma ID som används av cascading style sheets(CSS) när sidan designas. DOM-trädets söks då igenom efter noden som har en attributnod med det unika ID:t.

Man kan även välja ut en mängd av alla noder genom att söka efter en viss typ av elementnoder som till exempel "p" vilket returnerar alla sådana noder i trädet. Som standard söker man igenom hela trädet men man kan även själv definiera vilken av noderna man vill börja sökningen ifrån.

2.2.2 Prestanda och DOM


"DOM scripting är kostsamt för prestanda och en vanlig flaskhals i rika webbapplikationer” – Stoyan Stefanov Kapitel 3 High Performance JavaScript

De tre största prestandabövarna i DOM-scripting är:

- Access och modifiering av DOM-träd
- Modifiering av attribut och stilar via DOM-trädet.
- Hanteringen av användarinteraktion genom DOM-events


2.3 Funktionsbibliotek


jQuery har som slogan “write less do more” vilket stämmer bra in på de olika bibioteken. I de båda funktionsbiblioteken vi använder finns det till exempel möjlighet att anropa en enda funktion för att få element att glida in från sidan. Att skriva detta själv i JavaScript kräver en betydligt större arbetsinsats av utvecklaren mot att göra samma sak i funktionsbiblioteken, där behövs det en enda rad för att få det att fungera. De olika funktionsbiblioteken är helt enkelt utvecklade för att samla den vanligaste funktionaliteten på ett ställe och för att det ska räcka med att anropa en enda funktion för att köra en annars komplex funktionalitet.

Det finns ett antal olika funktionsbibliotek och i vår studie har vi koncentrerat oss på de två populäraste jQuery och Prototype.

18 High Performance JavaScript, kap. 3
2.3.1 jQuery

jQuery använder sig av funktionen $(()) för att med hjälp av samma teknik som CSS hittar element i DOM-trädet, #-tecknet för att välja ett element med ett visst ID och en punkt(,) för att välja element med en viss klass. jQuerys funktionalitet omnämndes första gången augusti 2005 i skaparens blogg där han nämner ett JavaScriptbibliotek som använder samma teknik som CSS. Version 1.0 av jQuery släpptes ett år senare i augusti 2006.19

2.3.2 Prototype

Prototype kan välja element ur DOM-trädet på ett antal olika sätt. Om man ska välja ut ett element med ID:et element-id används funktionen $(()) med ID:et inom citationstecken på detta vis; $("element-id").19 I motsats till jQuery går det bara att hämta element utifrån ett viss ID med $(()-funktionen, för att hämta element med samma teknik som jQuery används istället $$()-funktionen.20 Prototype har även en $F()-funktion som returnerar värdet av ett formulärelement utifrån ID:et på det elementet.21

---

19 http://jquery.org/history/
20 http://api.prototypejs.org/dom/dollar/
21 http://api.prototypejs.org/dom/dollar-dollar/
22 http://api.prototypejs.org/dom/dollar-F/
3 UTFORMNING AV EXPERIMENT

Det första testfallet genomförs genom att varje test laddar in biblioteket tio gånger som sedan körs tio gånger för att få fram ett relevant resultat på hur lång tid det tar för varje bibliotek att laddas in.


Testramverket håller de funktioner som mäts rena från allt annat än metoden som ska testas, for-loopen och accesser till fördefinierade globala variablär. Ramverket hanterar även uppbyggnad samt städning när testfallet kört färdigt. Den kan även gå in mellan två köringar av en metod och aningen skapa noder eller variabler som behövs eller städa efter föregående test. Det håller även övriga DOM-interaktioner till ett minimum för att se till att testfallet kan köras så snabbt som möjligt.

Från medelvärdena som ramverket räknar fram behöver vi hitta en gemensam faktor för alla funktioner så vi kan jämföra dem med varandra. Eftersom funktionerna itererar olika antal gånger så kan vi inte bara jämföra hur lång tid de tar i millisekunder med varandra. För att kunna jämföra räknas hur många gånger en funktion hinner köra per sekund fram.

Körningar per sekund, KPS
T1 = Medelvärdet av mätvärdena
I = Antal iterationer, specifikt för funktionen

KPS = 1/(T1/I/1000)
3.1 Testmiljö
Den dator som testerna körs på är hårdvarumässigt en typisk standarddator som kan finnas i var hem. Vi kör testerna under Windows för att ha möjlighet att köra testfallen i Internet Explorer. Det är just den här typen av datorer som troligtvis kör JavaScript applikationer idag.

3.1.1 Hårdvaruspecifikation

**Processor**
- Specifikation: Intel(R) Core(TM)2 Duo CPU E8500 @ 3,16GHz
- Antal kärnor: 2 (max 2)
- Antal trådar: 2 (max 2)
- Kärnhastighet: 1997,9 MHz
- L1 Data cache: 2 x 32 Kbytes 8-way set associative, 64-byte line size
- L1 Instruction cache: 2x32 Kbytes 8-way set associative, 64-byte line size
- L2 cache: 6144 Kbytes, 24-way set associative, 64-byte line size

**Moderkort**
- Modell: P5Q SE
- Tillverkare: ASUSTeK Computer INC.

**Minne**
- Minnestyp: DDR2
- Minnesstorlek: 4096 MBytes
- Kanaler: Dual, (Symetric)
- Minnesfrekvens: 399,6 MHz
- Databredd: 64 bits

3.1.2 Mjukvara och funktionsbiblioteksversioner

**Mjukvara**
Operativsystem: Windows Vista 64 SP2

**Webbläsarversioner**
- Opera: 11.10
- IE: 9.0.8112.1421
- Firefox: 4.0.1
- Chrome: 11.0.696.68
- Safari: 5.0.5 (7533.21.1)

**Scriptbibliotek**
- jQuery: 1.5.2
- Prototype: 1.7
3.2 Testfall laddningstider

Test ett går ut på att låta testmiljön hämta hem de två olika funktionsbibliotek på ett antal olika sätt. I testet vill vi ta reda på hur platsen som scripten hämtas från påverkar laddningstiden samt hur storleken på filen påverkar laddningstiden. Eftersom laddningstiden påverkar hur snabbt en användare kan börja arbeta med webbgränssnittet ska testet utvisa vilket av de valda funktionsbiblioteken som laddar snabbast totalt över de tre deltesterna.

Två deltester:
- Hämta mot servern/lokalt mot att hämta från repository
- Minimerade mot vanliga scriptfiler


3.3 Testfall traversering och manipulation

I testfall två ställs de två funktionsbiblioteken mot två uppgifter. I den första testas funktioner för att traversera DOM-trädet. Hur snabbt kan de olika biblioteken nå en föräldernod, syskonnod, barnnod, hämta en mängd av barn etc. Den andra uppgiften består av att testa bibliotekens funktioner för att modifiera DOM-trädet vi testar på. Genom att placera nya noder i början och i slutet av nivåer samt lägga till nya barn.

Figur 5 visar trädet som används i detta testfall, det består av en rot som har 150 barn som sedan i sig har 150 barn och dessa barn har ytterligare 130 barn. Ett av dessa barn har ytterligare 100 barn och ett av dessa har 100 barn. Ett av barnen på den här nivån har ett barn och sedan fortsätter trädet med att varje barn har ett barn fram till dess att det finns totalt 104 nivåer i trädet. Alla element har en klass som visar vilken nivå elementet ligger på och vilket element i ordningen det är på just den nivån. Till exempel ch2-3 visar att elementet ligger på nivå 2 och är det fjärde elementet i ordningen.

23 http://refresh-sf.com/yui/
3.4 Testfall attribut och stilar
I testfall tre testas bibliotekens funktioner för att ändra och läsa attributnoder samt hur de dynamiskt kan förändra dokumentets stilmallar efter laddning. Som i test två testas varje funktion enskilt i de båda biblioteken för att sedan läggas ihop till en totalsumma för hela testet.

3.5 Testfall eventrespons
I testfall fyra kommer ett antal eventlyssnare sättas, triggas och tas bort med hjälp av bibliotekens funktioner. Eventen som sätts och triggas är tomma JavaScriptfunktionskroppar.
4 RESULTAT
Alla mätningar är utförda med motsvarande funktioner i de båda biblioteken för att utföra en så rättvis bedömning som möjligt. I figurerna representerar y-axeln antal körningar per sekund, en hög stapel är ett snabbare resultat.

Figur 6 - Testfallen i de båda biblioteken

<table>
<thead>
<tr>
<th>Biblioteksjämförelse över alla webbläsare</th>
<th>Firefox</th>
<th>Chrome</th>
<th>Safari</th>
<th>Opera</th>
<th>IE</th>
</tr>
</thead>
<tbody>
<tr>
<td>jQuery</td>
<td>59 174,50</td>
<td>86 407,4</td>
<td>82 053,56</td>
<td>56 494,83</td>
<td>33 132,99</td>
</tr>
<tr>
<td>Prototype</td>
<td>84 340,54</td>
<td>356 374,5</td>
<td>110 970,9</td>
<td>95 284,9</td>
<td>17 750,29</td>
</tr>
</tbody>
</table>

Tabell 2 – Jämförelse över alla webbläsare, värden representerar antal körningar per sekund

4.1 Resultat testfall laddningstider
Tiden i Tabell 3 är ett medelvärde över 100 hämtningar av den specifika filen. Testet visar att om biblioteket hämtas från en extern källa tar det dubbelt så långt tid för biblioteket att laddas in än om det legat lokalt på datorn.

<table>
<thead>
<tr>
<th>Fil</th>
<th>Storlek (KB)</th>
<th>Tid (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>jquery-min-yui.js</td>
<td>92,8</td>
<td>20,5</td>
</tr>
<tr>
<td>jquery-1.5.2.js</td>
<td>214,1</td>
<td>19,8</td>
</tr>
<tr>
<td><a href="http://code.jquery.com/jquery-1.5.2.js">http://code.jquery.com/jquery-1.5.2.js</a></td>
<td>214,1</td>
<td>44,1</td>
</tr>
<tr>
<td>prototype-min.js</td>
<td>90,0</td>
<td>20,1</td>
</tr>
<tr>
<td>prototype.js</td>
<td>159,5</td>
<td>24,0</td>
</tr>
<tr>
<td><a href="https://ajax.googleapis.com/ajax/libs/prototype/1.7.0/prototype.js">https://ajax.googleapis.com/ajax/libs/prototype/1.7.0/prototype.js</a></td>
<td>159,5</td>
<td>57,3</td>
</tr>
</tbody>
</table>

Tabell 3 - Laddningstider
4.2 Resultat testfall traversering och manipulation

Testfallet visar att jQuery hade stora problem med att hämta ut en nåds alla föräldrar med funktionen parents. Tabellerna nedan visar hur webbläsaren Firefox presterade i testfallet för traversering i de båda biblioteken.

<table>
<thead>
<tr>
<th>Traversering jQuery</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent</td>
<td>873</td>
<td>400 000</td>
<td></td>
<td>458 190,149</td>
</tr>
<tr>
<td>Parents</td>
<td>1 315,85</td>
<td>5</td>
<td></td>
<td>3,800</td>
</tr>
<tr>
<td>Prev</td>
<td>1 091,95</td>
<td>298 000</td>
<td></td>
<td>272 906,269</td>
</tr>
<tr>
<td>PrevAll</td>
<td>6 177,35</td>
<td>30 000</td>
<td></td>
<td>4 856,451</td>
</tr>
<tr>
<td>Next</td>
<td>1 096,10</td>
<td>298 000</td>
<td></td>
<td>271 873,004</td>
</tr>
<tr>
<td>NextAll</td>
<td>6 751,65</td>
<td>30 000</td>
<td></td>
<td>4 443,358</td>
</tr>
<tr>
<td>Siblings</td>
<td>7 620,70</td>
<td>50 000</td>
<td></td>
<td>6 561,077</td>
</tr>
<tr>
<td>Children</td>
<td>1 209,80</td>
<td>300 000</td>
<td></td>
<td>247 974,872</td>
</tr>
</tbody>
</table>

Tabell 4 – Resultat av traversering i jQuery

<table>
<thead>
<tr>
<th>Traversering Prototype</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up</td>
<td>6 035,95</td>
<td>3 000 000</td>
<td></td>
<td>497 022,010</td>
</tr>
<tr>
<td>Ancestors</td>
<td>1 888,85</td>
<td>30 000</td>
<td></td>
<td>15 882,680</td>
</tr>
<tr>
<td>ChildElements</td>
<td>6 505,40</td>
<td>1 490 000</td>
<td></td>
<td>229 040,489</td>
</tr>
<tr>
<td>Next</td>
<td>7 076,75</td>
<td>2 980 000</td>
<td></td>
<td>421 097,255</td>
</tr>
<tr>
<td>Siblings</td>
<td>2 062,50</td>
<td>20 000</td>
<td></td>
<td>9 696,970</td>
</tr>
<tr>
<td>PreviousSiblings</td>
<td>1 891,50</td>
<td>20 000</td>
<td></td>
<td>10 573,619</td>
</tr>
<tr>
<td>Previous</td>
<td>10 854,70</td>
<td>30 000</td>
<td></td>
<td>2 763,780</td>
</tr>
<tr>
<td>NextSiblings</td>
<td>1 899,65</td>
<td>20 000</td>
<td></td>
<td>10 528,255</td>
</tr>
</tbody>
</table>

Tabell 5 – Resultat av traversering i Prototype

Traverseringen i de båda biblioteken hade ett spritt resultat mellan de olika webbläsarna. Prototype presterade mycket bättre i Chrome och Opera medan jQuery presterade bättre i de resterande webbläsarna. Som Figur 9 visar har Up och Next en bidragande orsak till hur resultatet ser ut i Chrome.
Figur 7 - Testfall traversering

Figur 8 - Traversing jQuery
Figur 9 - Traversing Prototype

Testet där biblioteken manipulerar DOM-trädet visar att jQuery har nästan dubbelt så många körningar per sekund som Prototype i samtliga webbläsare.

Figur 10 – Testfall manipulation
Figur 11 - Testfall manipulation jQuery

Figur 12 - Testfall manipulation Prototype
4.3 Resultat testfall attribut och stilar

Figur 13 - Testfall attribut

Figur 14 - Testfall stilar
Figur 15 - Testfall attribut jQuery

Figur 16 - Testfall attribut Prototype
4.4 **Resultat testfall eventrespons**

Återigen visar resultaten att Prototype är klart snabbare jQuery i flertalet webbläsare, det är bara i Opera som jQuery lyckas ha fler körningar per sekund.

![Diagram 17](image17.png)  
*Figur 17 - Testfall eventrespons*

![Diagram 18](image18.png)  
*Figur 18 - Testfall eventrespons jQuery*
Figur 19 - Testfall eventrespons Prototype
5 DISKUSSION

Resultatet i Tabell 3 visar en tydlig skillnad på laddningstider om filen ligger lokalt eller som en extern fil som måste laddas ner från en annan server. När man sedan tittar på resultatet när filerna ligger lokalt och det endast är storleken som skiljer de åt, ser man att filstorleken inte påverkar laddningstiden nämnvärt.

Det vi kan utläsa ur resultaten i Figur 6 har Prototype kunnat köra flest antal iterationer av varje funktion per sekund. Det enda testfallet där jQuery hade ett överläget bättre resultat var där DOM-trädet skulle manipuleras. Som Figur 10 visar kan jQuery köra i princip dubbelt så många iterationer i alla webbläsare mot vad Prototype kan presterera. Om man tittar närmare på hur funktionerna körs i Figur 11 och Figur 12 kan man se att det är Safari som drar upp resultatet för jQuery med nästan tre gånger så många körningar per sekund för nästan alla funktioner. Det är bara Html som körs i princip lika många gånger per sekund i både Prototype och jQuery i Safari.

![Diagram](image.png)

Figur 20 - Snabbaste webbläsaren

I Figur 20 ser vi att Chrome gick snabbast med både Prototype och jQuery sett över ett medelvärde av alla funktioners körningar per sekund i alla våra testfall. Detta var vad vi hade förväntat oss då Chrome har haft ett bra rykte vad det gäller JavaScriptprestanda och var tidiga ute med sin förbättrade JavaScriptmotor. Att IE var långsammast i våra tester är inte speciellt förvånande då de under antal versioner haft markant sämre hantering av JavaScript än övriga webbläsare. Vi förväntade oss dock att IE skulle presterera så pass bra som den gjorde med den senaste versionen Internet Explorer 9 ser Microsoft ut att ha gjort goda framsteg. 24

I Figur 20 ser vi också att Safari som presterade bra i våra tester, så pass bra att den blir näst snabbaste webbläsaren efter Chrome. Vilket är lite imponerande då Safari först och främst är byggt för Mac och nu istället kördes under Windows med de övriga läsarna. Tabell 2 visar även att Opera går snabbare än Firefox med totalt 143 514 mot 151 742 körningar per sekund.

jQuery testerNA i Firefox körde som man kan se i siffrorna nedan marginellt snabbare än Operas men även Opera slår Firefox och får en bättre totaltid på grund av snabbare Prototype exekvering.

Firefox jQuery: 59174 KPS
Opera jQuery: 56494 KPS
Firefox Prototype: 84340 KPS
Opera Prototype: 95284 KPS

Att hämta och sätta attribut som visas i Figur 14 går åter igen snabbast i Prototype i alla webbläsare utom Internet Explorer. Det som gör att resultatet i Chrome är så högt ser vi i Figur 15 och 16, funktionen ReadAttribute kör närmare 1 400 000 körningar per sekund och motsvarande funktion GetAttr i jQuery bara kör runt 530 000 körningar per sekund.

Testfallet Event visar i Figur 17 ett intressant resultat i webbläsaren Opera där jQuery visar sig presteras bättre än vad Prototype gör. Men om man sedan tittar på hur varje funktion kördes i webbläsarna ser man i Figur 18 att funktionen Bind i jQuery får ett väldigt avvikande resultat mot alla andra webbläsare. För oss tyder detta på att något inte stöds eller att funktionen Bind inte riktigt har kört på ett korrekt sätt i just denna webbläsare. Då vi inte alls ser samma pik i Prototype som eventhantering i Figur 19 så tyder detta på att något inte stödseller att funktionen bind inte riktigt har kört på ett korrekt sätt i just denna webbläsare. Annat intressant att notera angående events i Figur 18 och 19 är att det varierar vad som tar längst tid att sätta ett event med Bind och Observe eller att ta bort det med Unbind och StopObserving. Överlag ser det ut som att att ta bort ett event tar längre tid än att knyta event i jQuery medan i Prototype så är det tvärt om.

I figur 20 ser vi också att över alla testfall är Prototype överlag snabbare i alla webbläsare utom Internet Explorer. Jämförs testfall för testfall ser allt ungefär ut som i övriga webbläsare. jQuery går snabbare i manipulationen medan Prototype vinner resten, förutom traverseringen i Figur 7. Där jQuery går snabbare i tre av webbläsarna men Prototype klarar av fler körningar per sekund på grund av ett bra resultat i Chrome. Som man kan se i Figur 8 och 9 så går funktionerna Prev, Next och Children i jQuery betydligt snabbare än i Prototype, Prev och Next går ca 4,5 gånger snabbare i jQuery än motsvarande funktion i Prototype och Children går strax över 8,5 gånger snabbare enligt tabellerna jQuery Traversering och Prototype Traversering i Bilaga 9.1.5 Internet Explorer. Dessa resultat ser ut att vara den största enskilt bidragande faktorn till att jQuery totalt sett går snabbare än Prototype i Internet Explorer.

Testet av traversering med jQuery i Firefox i Tabell 4 visar ett väldigt intressant värde där funktionen Parents endast behöver köras fem gånger för att nå upp i ett medelvärde på 1 315 ms vilket är väldigt lite om man jämför med motsvarande funktion i Prototype i Tabell 5 som kör 30 000 iterationer för att komma upp i 1 888,85 ms.


Problemet ligger i att vi itererar över funktionen och försöker i till exempel Hide dölja ett element. På varv två och framåt i loopen kommer vi då köra Hide på ett

Då Prototype inte har någon removeAttributefunktion saknas en funktion i testfallet för attribut, detta kan påverkat Prototypes resultat åt något håll. Funktionerna för att ta bort ett element helt från DOM-trädet kunde också varit med i manipulationstestet då de bör vara ganska tunga tidsmässigt.

Det hade kanske varit enklare att räkna och jämföra mätvärdena på de olika funktionerna om de hade itererat samma antal gånger i alla olika webbläsare. Samt att alla funktioner i sig hade itererat samma antal gånger. Tanken var att göra så från början men det ställdes till problem då det var för stora skillnader mellan de olika funktionerna och hur snabbt webbläsarna arbetade. Vilket gjorde att vi istället valde att variera antalet iterationer beroende på webbläsare, men fortfarande såg till att funktionen kördes i de 750ms som var satt som gräns för minsta tid ett test fick köra.

5.1 Vad kan påverkat mätningarna

För att mäta tiden i våra testfall använder vi oss av följande kodstycke. Det tar dagens datum och klockslag när funktionen startar och mätningen slutar. Sedan drar vi av starttiden från sluttiden för att få fram tiden vår funktion exekverat.

```javascript
var timeDiff = {
    setStartTime:function(){
        d = new Date();
        time = d.getTime();
    },
    getDiff:function (){d = new Date();
        return (d.getTime()-time);
    }
}
```


För att motverka denna felmarginal har vi i våra test försökt att se till att varje funktion exekverar så många gånger att det tar minst 750 ms. Felmarginalen ligger då runt 1 % av resultatet eftersom medelvärdet av noll och 15 är 7,5 och 7,5 genom 750 är 0,01.

---

25 High Performance JavaScript, kap. 6
Mätvärdena vid inladdning av de olika biblioteken i testfall ett kan påverkas av andra processer på datorn som skriver eller läser på hårddisen där filen ligger. Den del av testet där filen hämtas från en extern källa kan den tillfälliga belastningen på nätet och klientens bandbredd påverka hur snabbt det går att hämta hem de olika filerna. I övriga testfall har internetåtkomsten stängts av för att minska den externa påverkan på mätvärdena. Övriga processer har också hållits till ett minimum.

I testfall två till fyra bortser vi från tiden det tar att iterera över de funktioner som testas då iterationerna inte består av så stor del av exekveringstiden. Nedan följer våra mätningar på iterationstiderna och dessa mätningar påverkas också av 0-15 millisekunders felmarginal som beskrevs tidigare.

Figur 21 visar lite intressanta resultat. Vi antog att iterationskostnaden i de olika webbläsarna är så pass låg att den kan räknas bort eftersom vi inte räknade med att behöva iterera mer än 30 000 gånger över någon funktion. Men för att nå upp till timern på 750 ms visade sig att några av funktionerna behövde fler iterationer på sig i de snabbare webbläsarna. På de flesta funktioner fungerade det bra på 30 000 iterationer och kostnaden för detta ser vi som försumbar och inom den redan befintliga 1% felmarginalen på 0-15 millisekunder. Undantagen från detta var främst Prototypes funktioner Next och Up som båda kör tre miljoner gånger i både Chrome och Firefox. Även i Safari behövde funktionen Next köras ungefär 3 miljoner gånger. Iterationskostnaden i Firefox för tre miljoner iterationer kan bidragit till att Firefox inte presterat så pass bra som vi trodde den skulle göra.
6 Slutsats
Testresultaten har både förvånat och förvirrat oss lite. I början trodde vi att jQuery var populäraast på grund av att det presterade bäst, men som vi nu har sett i resultaten har Prototype presterat bättre än jQuery i nästan alla enskilda testfall överlag. Det finns vissa webbläsare som varit bättre eller sämre på att hantera de båda biblioteken men överlag har Prototype tagit hem de flesta testerna.

Att Firefox ser ut att ha underpresterat något i våra tester kan ha med utformningen av testfallen att göra. I övrigt gick alla webbläsare riktigt snabbt och det ser ut som mycket har hänt på JavaScriptfronten de senaste åren. Snabbast av alla webbläsare var Chrome med ganska god marginal. Internet Explorer var den enda webbläsaren som stack ut ur mängden och jQuery överlag presterade bättre än Prototype i testerna.

Vår hypotes att jQuery skulle prestera bättre än Prototype i ett webbgränssnitt med DOM-interaktioner ser enligt våra testresultat inte ut att stämma. Prototype presterade bäst i våra tester sett över alla testfall och webbläsare. Undantaget på testfallen var manipulation där jQuery var snabbast och det kan tänkas att den finns speciella där en webbapplikations prestandaflaskhals är just manipulationen. Då kan jQuery vara ett alternativ ur prestandaperspektiv, i alla andra fall presterar Prototype bäst.

Vårt resultat gör att alla som använder jQuery inte ska byta till Prototype bara för att det presterar mycket bättre i flertalet tester. Att manipulation görs så mycket snabbare i jQuery kan vara en anledning till att fortsätta använda jQuery och hoppas på att prestandan utvecklas även i de andra fallen. Får Prototype bättre prestanda vid manipulation i kommande versioner utan att påverka de andra funktionernas prestanda kan Prototypes popularitet öka mycket i framtiden.

6.1 Förslag på fortsatt forskning
Under arbetets gång har nya funderingar och frågor dykt upp som vi inte har haft möjlighet att studera. I vårt arbete har vi tittat på hur olika javascriptbibliotek har presterat i DOM-interaktioner. Vad som inte testas mer än i vårt lilla iterationstest är hur prestandan för JavaScript har förändrats i de nya JavaScriptmotorerna till webbläsarna idag. Redan i det testet ser vi att Firefox hamnar långt efter internet Explorer som i sin tur ligger en bit bakom de andra. Det vore intressant att undersöka om det endast gäller för for-loopen eller om det även stämmer för andra loopkonstruktioner och övrig grundläggande JavaScriptprestanda.

Man skulle även kunna titta på andra funktioner i de två JavaScriptbiblioteken, då vi endast testat en del av de funktioner som används vid DOM-interaktioner och inget annat. Till exempel skulle man kunna undersöka hur de båda biblioteken hanterar Ajax-requests och om det är någon skillnad på prestandan där. Samt om det är någon skillnad på vilken webbläsare som används. Då webbaserade applikationer blir mer och mer interaktiva och behöver klara av en högre belastning av Ajax-lösningar än tidigare. Det är förmodligen lite svårtare att mäta Ajax-requests då allt som går över Internet har så många externa faktorer som spelar in. Det finns idag stöd i de flesta webbläsare för någon form av utvecklarverktyg där man enkelt kan se och profilera Ajax-request.
Nya versioner av biblioteken släpps kontinuerligt och den 12 maj släpptes en ny version av biblioteket jQuery. Då detta var mitt i arbetet hade vi inte möjlighet att byta ut vår kod för att stödja det nya biblioteket. Det nya innehåller bland annat förändringar på hur jQuery hanterar DOM-attribut vilket hade varit intressant för vår undersökning. Så att testa detta nya bibliotek och eventuella flera uppdateringar kommer vara intressanta.
7 **ORDLISTA**

DOM - Document Object Model

JS - JavaScript

CSS - Cascading Style Sheets

HTML - Hyper Text Markup Language

XHTML - eXtensible HyperText Markup Language

XML - eXtensible Markup Language

Ajax - Asynchronous JavaScript and XML

IE - Internet Explorer

ECMA - European Computer Manufacturers Association

KPS – Körningar Per Sekund
8 LITTERATURFÖRTECKNING

8.1 Böcker


8.2 Webbsidor
Google Trends, (Besökt 2011-05-20)
http://www.google.com/trends?q=Yahoo+javascript%2C+Prototype+javascript%2C+Jquery+javascript&ctab=0&geo=all&date=all&sort=0

W3Schools Browser Satistics, (Besökt 2011-05-20)
http://www.w3schools.com/browsers/browsers_stats.asp

ECMA International 262 Standard, (Besökt 2011-05-20)

The Mozilla Foundation wiki, (Besökt 2011-05-20)
https://wiki.mozilla.org/JaegerMonkey

About v8 JavaScript Engine, (Besökt 2011-05-20)
http://code.google.com/p/v8/

Opera Core Concerns - Charakan, (Besökt 2011-05-20)
http://my.opera.com/core/blog/2009/02/04/carakan

Safari Browser Information, (Besökt 2011-05-20)

MSDN Blog IEBlog, (Besökt 2011-05-20)

jQuery Project History, (Besökt 2011-05-20)
http://jquery.org/history/

Prototype API Documentation, (Besökt 2011-05-20)
http://api.prototypejs.org/dom/dollar/

Prototype API Documentation, (Besökt 2011-05-20)
http://api.prototypejs.org/dom/dollar-dollar/

Prototype API Documentation, (Besökt 2011-05-20)
http://api.prototypejs.org/dom/dollar-F/

Online YUI Compressor, (Besökt 2011-05-20)
http://refresh-sf.com/yui/

WebKit SunSpider JavaScript Benchmark Results, (Besökt 2011-05-20)
http://ie.microsoft.com/testdrive/benchmarks/sunspider/default.html
9   BILAGOR

9.1   Resultattabeller

9.1.1   Firefox 4.0.1

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent</td>
<td>873</td>
<td>400 000</td>
<td>458 190,149</td>
</tr>
<tr>
<td>Parents</td>
<td>1 315,85</td>
<td>5</td>
<td>3,800</td>
</tr>
<tr>
<td>Prev</td>
<td>1 091,95</td>
<td>298 000</td>
<td>272 906,269</td>
</tr>
<tr>
<td>PrevAll</td>
<td>6 177,35</td>
<td>30 000</td>
<td>4 856,451</td>
</tr>
<tr>
<td>Next</td>
<td>1 096,10</td>
<td>298 000</td>
<td>271 873,004</td>
</tr>
<tr>
<td>NextAll</td>
<td>6 751,65</td>
<td>30 000</td>
<td>4 443,358</td>
</tr>
<tr>
<td>Siblings</td>
<td>7 620,70</td>
<td>50 000</td>
<td>6 561,077</td>
</tr>
<tr>
<td>Children</td>
<td>1 209,80</td>
<td>300 000</td>
<td>247 974,872</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up</td>
<td>6 035,95</td>
<td>3 000 000</td>
<td>497 022,010</td>
</tr>
<tr>
<td>Ancestors</td>
<td>1 888,85</td>
<td>30 000</td>
<td>15 882,680</td>
</tr>
<tr>
<td>ChildElements</td>
<td>6 505,40</td>
<td>1 490 000</td>
<td>229 040,489</td>
</tr>
<tr>
<td>Next</td>
<td>7 076,75</td>
<td>2 980 000</td>
<td>421 097,255</td>
</tr>
<tr>
<td>Siblings</td>
<td>2 062,50</td>
<td>20 000</td>
<td>9 696,970</td>
</tr>
<tr>
<td>PreviousSiblings</td>
<td>1 891,50</td>
<td>20 000</td>
<td>10 573,619</td>
</tr>
<tr>
<td>Previous</td>
<td>10 854,70</td>
<td>30 000</td>
<td>2 763,780</td>
</tr>
<tr>
<td>NextSiblings</td>
<td>1 899,65</td>
<td>20 000</td>
<td>10 528,255</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Html</td>
<td>1 209,85</td>
<td>30 000</td>
<td>24 796,462</td>
</tr>
<tr>
<td>Before</td>
<td>1 017,20</td>
<td>40 000</td>
<td>39 323,634</td>
</tr>
<tr>
<td>After</td>
<td>2 033,70</td>
<td>30 000</td>
<td>14 751,438</td>
</tr>
<tr>
<td>Prepend</td>
<td>2 107,80</td>
<td>30 000</td>
<td>14 232,849</td>
</tr>
<tr>
<td>Append</td>
<td>800,30</td>
<td>30 000</td>
<td>37 485,943</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Html</td>
<td>1 676,15</td>
<td>30 000</td>
<td>17 898,159</td>
</tr>
<tr>
<td>Before</td>
<td>3 476,10</td>
<td>30 000</td>
<td>8 630,362</td>
</tr>
<tr>
<td>After</td>
<td>5 987,95</td>
<td>30 000</td>
<td>5 010,062</td>
</tr>
<tr>
<td>Prepend</td>
<td>2 934,40</td>
<td>30 000</td>
<td>10 223,555</td>
</tr>
<tr>
<td>Append</td>
<td>2 844,75</td>
<td>30 000</td>
<td>10 545,742</td>
</tr>
</tbody>
</table>
### Attribut jQuery

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde (ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddClass</td>
<td>1 321,30</td>
<td>1 500</td>
<td>1 135,246</td>
</tr>
<tr>
<td>RemoveClass</td>
<td>1 352</td>
<td>1 500</td>
<td>1 109,467</td>
</tr>
<tr>
<td>HasClass</td>
<td>1 083,90</td>
<td>3 000</td>
<td>2 767,783</td>
</tr>
<tr>
<td>GetAttr</td>
<td>1 878,40</td>
<td>300 000</td>
<td>159 710,392</td>
</tr>
<tr>
<td>SetAttr</td>
<td>1 455,20</td>
<td>100 000</td>
<td>68 719,076</td>
</tr>
<tr>
<td>RemoveAttr</td>
<td>1 562,30</td>
<td>100 000</td>
<td>64 008,193</td>
</tr>
</tbody>
</table>

### Attribut Prototype

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde (ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddClassName</td>
<td>1 340,90</td>
<td>1 500</td>
<td>1 118,652</td>
</tr>
<tr>
<td>RemoveClassName</td>
<td>1 354,55</td>
<td>1 500</td>
<td>1 107,379</td>
</tr>
<tr>
<td>HasClassName</td>
<td>3 035,50</td>
<td>3 000</td>
<td>988,305</td>
</tr>
<tr>
<td>ReadAttribute</td>
<td>854,25</td>
<td>300 000</td>
<td>351 185,250</td>
</tr>
<tr>
<td>WriteAttribute</td>
<td>1 025,60</td>
<td>100 000</td>
<td>97 503,900</td>
</tr>
</tbody>
</table>

### Stilar jQuery

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde (ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetCss</td>
<td>996,50</td>
<td>50 000</td>
<td>50 175,615</td>
</tr>
<tr>
<td>GetCss</td>
<td>1 012,15</td>
<td>50 000</td>
<td>49 399,793</td>
</tr>
<tr>
<td>Height</td>
<td>1 161,85</td>
<td>15 000</td>
<td>12 910,445</td>
</tr>
<tr>
<td>Width</td>
<td>1 134,60</td>
<td>15 000</td>
<td>13 220,518</td>
</tr>
<tr>
<td>Is</td>
<td>1 852,40</td>
<td>30 000</td>
<td>16 195,206</td>
</tr>
<tr>
<td>Show</td>
<td>1 389,20</td>
<td>300 000</td>
<td>215 951,627</td>
</tr>
<tr>
<td>Hide</td>
<td>1190,45</td>
<td>50 000</td>
<td>42 000,924</td>
</tr>
<tr>
<td>Toggle</td>
<td>765,50</td>
<td>5 000</td>
<td>6 531,679</td>
</tr>
</tbody>
</table>

### Stilar Prototype

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde (ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetStyle</td>
<td>1 067,55</td>
<td>50 000</td>
<td>46 836,214</td>
</tr>
<tr>
<td>GetStyle</td>
<td>877,55</td>
<td>50 000</td>
<td>56 976,810</td>
</tr>
<tr>
<td>GetHeight</td>
<td>1 022,45</td>
<td>50 000</td>
<td>48 902,147</td>
</tr>
<tr>
<td>GetWidth</td>
<td>1 048,20</td>
<td>50 000</td>
<td>47 700,820</td>
</tr>
<tr>
<td>Is</td>
<td>730,05</td>
<td>350 000</td>
<td>479 419,218</td>
</tr>
<tr>
<td>Show</td>
<td>1 066,75</td>
<td>150 000</td>
<td>140 614,015</td>
</tr>
<tr>
<td>Hide</td>
<td>748,10</td>
<td>150 000</td>
<td>200 507,953</td>
</tr>
<tr>
<td>Toggle</td>
<td>1 016,65</td>
<td>100 000</td>
<td>98 362,268</td>
</tr>
</tbody>
</table>

### Event jQuery

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde (ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bind</td>
<td>1 313,15</td>
<td>20 000</td>
<td>15 230,552</td>
</tr>
<tr>
<td>Trigger</td>
<td>1 046,35</td>
<td>15 000</td>
<td>14 335,547</td>
</tr>
<tr>
<td>Unbind</td>
<td>1 418,8</td>
<td>5 000</td>
<td>3 524,105</td>
</tr>
<tr>
<td>Event Prototype</td>
<td>Funktion</td>
<td>Medelvärde(ms)</td>
<td>Antal Iterationer</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------</td>
<td>----------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Observe</td>
<td>911,9</td>
<td>30 000</td>
<td></td>
</tr>
<tr>
<td>Fire</td>
<td>805,45</td>
<td>35 000</td>
<td></td>
</tr>
<tr>
<td>StopObserving</td>
<td>1 126,3</td>
<td>20 000</td>
<td></td>
</tr>
</tbody>
</table>

9.1.2 Chrome 11.0.696.68

<table>
<thead>
<tr>
<th>Traversering jQuery</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent</td>
<td>1 838,10</td>
<td>400 000</td>
<td></td>
<td>217 616,017</td>
</tr>
<tr>
<td>Parents</td>
<td>2 702,95</td>
<td>5</td>
<td></td>
<td>1 850</td>
</tr>
<tr>
<td>Prev</td>
<td>1 150,75</td>
<td>298 000</td>
<td></td>
<td>258 961,547</td>
</tr>
<tr>
<td>PrevAll</td>
<td>1 009,25</td>
<td>30 000</td>
<td></td>
<td>29 725,043</td>
</tr>
<tr>
<td>Next</td>
<td>1 171,70</td>
<td>298 000</td>
<td></td>
<td>254 331,313</td>
</tr>
<tr>
<td>NextAll</td>
<td>1 009,85</td>
<td>30 000</td>
<td></td>
<td>29 707,382</td>
</tr>
<tr>
<td>Siblings</td>
<td>1 145,20</td>
<td>50 000</td>
<td></td>
<td>43 660,496</td>
</tr>
<tr>
<td>Children</td>
<td>1 227,25</td>
<td>300 000</td>
<td></td>
<td>244 448,971</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Traversering Prototype</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up</td>
<td>1 325,45</td>
<td>3 000 000</td>
<td></td>
<td>2 263 382,248</td>
</tr>
<tr>
<td>Ancestors</td>
<td>1 321,60</td>
<td>30 000</td>
<td></td>
<td>22 699,758</td>
</tr>
<tr>
<td>ChildElements</td>
<td>1 903,80</td>
<td>1 490 000</td>
<td></td>
<td>782 645,236</td>
</tr>
<tr>
<td>Next</td>
<td>1 593,10</td>
<td>2 980 000</td>
<td></td>
<td>1 870 566,819</td>
</tr>
<tr>
<td>Siblings</td>
<td>1 396,80</td>
<td>20 000</td>
<td></td>
<td>14 318,442</td>
</tr>
<tr>
<td>PreviousSiblings</td>
<td>1 338,85</td>
<td>20 000</td>
<td></td>
<td>14 938,193</td>
</tr>
<tr>
<td>Previous</td>
<td>2 456,85</td>
<td>30 000</td>
<td></td>
<td>12 210,758</td>
</tr>
<tr>
<td>NextSiblings</td>
<td>1 373,10</td>
<td>20 000</td>
<td></td>
<td>14 565,582</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Manipulation jQuery</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Html</td>
<td>1 940,05</td>
<td>30 000</td>
<td></td>
<td>15 463,519</td>
</tr>
<tr>
<td>Before</td>
<td>845,80</td>
<td>42 000</td>
<td></td>
<td>49 657,129</td>
</tr>
<tr>
<td>After</td>
<td>1 003,70</td>
<td>42 000</td>
<td></td>
<td>41 845,173</td>
</tr>
<tr>
<td>Prepend</td>
<td>1 050,25</td>
<td>42 000</td>
<td></td>
<td>39 990,478</td>
</tr>
<tr>
<td>Append</td>
<td>919,25</td>
<td>42 000</td>
<td></td>
<td>45 689,421</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Manipulation Prototype</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Html</td>
<td>1 074,40</td>
<td>30 000</td>
<td></td>
<td>27 922,561</td>
</tr>
<tr>
<td>Before</td>
<td>1 636,50</td>
<td>30 000</td>
<td></td>
<td>18 331,806</td>
</tr>
<tr>
<td>After</td>
<td>1 611,25</td>
<td>30 000</td>
<td></td>
<td>18 619,085</td>
</tr>
<tr>
<td>Prepend</td>
<td>1 561,75</td>
<td>30 000</td>
<td></td>
<td>19 209,220</td>
</tr>
<tr>
<td>Append</td>
<td>1 678,40</td>
<td>30 000</td>
<td></td>
<td>17 874,166</td>
</tr>
</tbody>
</table>
### Attribut jQuery

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddClass</td>
<td>1 357,45</td>
<td>6 000</td>
<td>4 420,052</td>
</tr>
<tr>
<td>RemoveClass</td>
<td>1 715,35</td>
<td>6 000</td>
<td>3 497,828</td>
</tr>
<tr>
<td>HasClass</td>
<td>1 355,75</td>
<td>5 000</td>
<td>3 687,996</td>
</tr>
<tr>
<td>GetAttr</td>
<td>942,35</td>
<td>500 000</td>
<td>530 588,423</td>
</tr>
<tr>
<td>SetAttr</td>
<td>814,35</td>
<td>110 000</td>
<td>135 077,055</td>
</tr>
<tr>
<td>RemoveAttr</td>
<td>868,05</td>
<td>120 000</td>
<td>138 240,885</td>
</tr>
</tbody>
</table>

### Attribut Prototype

<table>
<thead>
<tr>
<th>Funktion Name</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddClassName</td>
<td>1 189,90</td>
<td>5 000</td>
<td>4 702,034</td>
</tr>
<tr>
<td>RemoveClassName</td>
<td>1 125,75</td>
<td>5 000</td>
<td>4 441,483</td>
</tr>
<tr>
<td>HasClassName</td>
<td>2 131,20</td>
<td>3 000</td>
<td>1 407 658</td>
</tr>
<tr>
<td>ReadAttribute</td>
<td>2 130,70</td>
<td>3 000 000</td>
<td>1 407 987,985</td>
</tr>
<tr>
<td>WriteAttribute</td>
<td>1 219,80</td>
<td>300 000</td>
<td>245 941,958</td>
</tr>
</tbody>
</table>

### Stilar jQuery

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetCss</td>
<td>810,75</td>
<td>80 000</td>
<td>98 674,067</td>
</tr>
<tr>
<td>GetCss</td>
<td>918,60</td>
<td>80 000</td>
<td>87 089,049</td>
</tr>
<tr>
<td>Height</td>
<td>1 348,50</td>
<td>30 000</td>
<td>22 246,941</td>
</tr>
<tr>
<td>Width</td>
<td>1 334,90</td>
<td>30 000</td>
<td>22 473,594</td>
</tr>
<tr>
<td>Is</td>
<td>1 102,55</td>
<td>50 000</td>
<td>45 349,417</td>
</tr>
<tr>
<td>Show</td>
<td>924,95</td>
<td>500 000</td>
<td>540 569,761</td>
</tr>
<tr>
<td>Hide</td>
<td>1 143,85</td>
<td>50 000</td>
<td>43 712,025</td>
</tr>
<tr>
<td>Toggle</td>
<td>796,20</td>
<td>10 000</td>
<td>12 559,658</td>
</tr>
</tbody>
</table>

### Stilar Prototype

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetCss</td>
<td>1 156,85</td>
<td>150 000</td>
<td>129 662,445</td>
</tr>
<tr>
<td>GetCss</td>
<td>1 092,75</td>
<td>100 000</td>
<td>91 512,240</td>
</tr>
<tr>
<td>Height</td>
<td>1 211,5</td>
<td>150 000</td>
<td>123 813,454</td>
</tr>
<tr>
<td>Width</td>
<td>1 229,85</td>
<td>150 000</td>
<td>121 966,093</td>
</tr>
<tr>
<td>Is</td>
<td>1 088,55</td>
<td>2 500 000</td>
<td>2 296 633,136</td>
</tr>
<tr>
<td>Show</td>
<td>826,9</td>
<td>2 000 000</td>
<td>2 418 672,149</td>
</tr>
<tr>
<td>Hide</td>
<td>760,8</td>
<td>300 000</td>
<td>394 321,767</td>
</tr>
<tr>
<td>Toggle</td>
<td>2 308,85</td>
<td>1 000 000</td>
<td>433 116,053</td>
</tr>
</tbody>
</table>
### Event jQuery

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde (ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bind</td>
<td>1 001.85</td>
<td>1 000</td>
<td>998 153</td>
</tr>
<tr>
<td>Trigger</td>
<td>807.25</td>
<td>30 000</td>
<td>3 7163 208</td>
</tr>
<tr>
<td>Unbind</td>
<td>784.05</td>
<td>2 300</td>
<td>2 933 486</td>
</tr>
</tbody>
</table>

### Event Prototype

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde (ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observe</td>
<td>871.15</td>
<td>40 000</td>
<td>45 916 318</td>
</tr>
<tr>
<td>Fire</td>
<td>932.25</td>
<td>100 000</td>
<td>10 7267 364</td>
</tr>
<tr>
<td>StopObserving</td>
<td>835.2</td>
<td>5 000</td>
<td>5 986 590</td>
</tr>
</tbody>
</table>

### 9.1.3 Safari 5.0.5

#### Traversering jQuery

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde (ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent</td>
<td>1 905.30</td>
<td>400 000</td>
<td>209 940 692</td>
</tr>
<tr>
<td>Parents</td>
<td>3 767.10</td>
<td>5</td>
<td>1 327</td>
</tr>
<tr>
<td>Prev</td>
<td>1 233.10</td>
<td>298 000</td>
<td>241 667 342</td>
</tr>
<tr>
<td>PrevAll</td>
<td>2 837.05</td>
<td>30 000</td>
<td>10 574 364</td>
</tr>
<tr>
<td>Next</td>
<td>1 260.75</td>
<td>298 000</td>
<td>236 367 242</td>
</tr>
<tr>
<td>NextAll</td>
<td>2 990.95</td>
<td>30 000</td>
<td>10 030 258</td>
</tr>
<tr>
<td>Siblings</td>
<td>4 236.65</td>
<td>50 000</td>
<td>11 801 777</td>
</tr>
<tr>
<td>Children</td>
<td>1 498.25</td>
<td>300 000</td>
<td>200 233 606</td>
</tr>
</tbody>
</table>

#### Traversering Prototype

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde (ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up</td>
<td>6 462.25</td>
<td>30 000</td>
<td>4 642 346</td>
</tr>
<tr>
<td>Ancestors</td>
<td>764.55</td>
<td>30 000</td>
<td>39 238 768</td>
</tr>
<tr>
<td>ChildElements</td>
<td>5 937.25</td>
<td>1 490 000</td>
<td>250 957 935</td>
</tr>
<tr>
<td>Next</td>
<td>7 605.95</td>
<td>2 980 000</td>
<td>391 798 526</td>
</tr>
<tr>
<td>Siblings</td>
<td>995.70</td>
<td>20 000</td>
<td>20 086 371</td>
</tr>
<tr>
<td>PreviousSiblings</td>
<td>749.80</td>
<td>20 000</td>
<td>26 673 780</td>
</tr>
<tr>
<td>Previous</td>
<td>1 1197.20</td>
<td>30 000</td>
<td>2 679 241</td>
</tr>
<tr>
<td>NextSiblings</td>
<td>755.25</td>
<td>20 000</td>
<td>26 481 298</td>
</tr>
</tbody>
</table>

#### Manipulation jQuery

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde (ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Html</td>
<td>1 058.05</td>
<td>30 000</td>
<td>28 354 048</td>
</tr>
<tr>
<td>Before</td>
<td>850.25</td>
<td>50 000</td>
<td>58 806 233</td>
</tr>
<tr>
<td>After</td>
<td>839.20</td>
<td>50 000</td>
<td>59 580 553</td>
</tr>
<tr>
<td>Prepend</td>
<td>905</td>
<td>50 000</td>
<td>55 248 619</td>
</tr>
<tr>
<td>Append</td>
<td>911.85</td>
<td>50 000</td>
<td>54 833 580</td>
</tr>
<tr>
<td>Manipulation Prototype</td>
<td>Function</td>
<td>Medelvärde(ms)</td>
<td>Antal Iterationer</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------</td>
<td>----------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Html</td>
<td></td>
<td>999,65</td>
<td>30 000</td>
</tr>
<tr>
<td>Before</td>
<td></td>
<td>854,15</td>
<td>18 000</td>
</tr>
<tr>
<td>After</td>
<td></td>
<td>937,20</td>
<td>18 000</td>
</tr>
<tr>
<td>Prepend</td>
<td></td>
<td>858,85</td>
<td>18 000</td>
</tr>
<tr>
<td>Append</td>
<td></td>
<td>894,55</td>
<td>18 000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribut jQuery</th>
<th>Function</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddClass</td>
<td></td>
<td>1 329,5</td>
<td>6 000</td>
<td>4 512,975</td>
</tr>
<tr>
<td>RemoveClass</td>
<td></td>
<td>940,35</td>
<td>7 000</td>
<td>7 444,037</td>
</tr>
<tr>
<td>HasClass</td>
<td></td>
<td>1 485,45</td>
<td>5 000</td>
<td>3 365,983</td>
</tr>
<tr>
<td>GetAttr</td>
<td></td>
<td>874,35</td>
<td>500 000</td>
<td>571 853,377</td>
</tr>
<tr>
<td>SetAttr</td>
<td></td>
<td>1 150,5</td>
<td>200 000</td>
<td>173 837,462</td>
</tr>
<tr>
<td>RemoveAttr</td>
<td></td>
<td>902,1</td>
<td>130 000</td>
<td>144 108,192</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribut Prototype</th>
<th>Function</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddClassName</td>
<td></td>
<td>833,35</td>
<td>5 000</td>
<td>5 999,880</td>
</tr>
<tr>
<td>RemoveClassName</td>
<td></td>
<td>2 937,7</td>
<td>15 000</td>
<td>5 106,035</td>
</tr>
<tr>
<td>HasClassName</td>
<td></td>
<td>2 731,25</td>
<td>3 000</td>
<td>1 098,398</td>
</tr>
<tr>
<td>ReadAttribute</td>
<td></td>
<td>1 504,7</td>
<td>1 000 000</td>
<td>664 584,303</td>
</tr>
<tr>
<td>WriteAttribute</td>
<td></td>
<td>1 250,95</td>
<td>300 000</td>
<td>239 817,739</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stilar jQuery</th>
<th>Function</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetCss</td>
<td></td>
<td>772,5</td>
<td>80 000</td>
<td>103 559,871</td>
</tr>
<tr>
<td>GetCss</td>
<td></td>
<td>841,25</td>
<td>120 000</td>
<td>142 644,874</td>
</tr>
<tr>
<td>Height</td>
<td></td>
<td>948,75</td>
<td>30 000</td>
<td>31 620,553</td>
</tr>
<tr>
<td>Width</td>
<td></td>
<td>958,4</td>
<td>30000</td>
<td>31 302,170</td>
</tr>
<tr>
<td>Is</td>
<td></td>
<td>1 468,05</td>
<td>50 000</td>
<td>34 058,785</td>
</tr>
<tr>
<td>Show</td>
<td></td>
<td>1 912,45</td>
<td>500 000</td>
<td>261 444,744</td>
</tr>
<tr>
<td>Hide</td>
<td></td>
<td>913,9</td>
<td>50 000</td>
<td>54 710,581</td>
</tr>
<tr>
<td>Toggle</td>
<td></td>
<td>861,45</td>
<td>10 000</td>
<td>11 608,335</td>
</tr>
</tbody>
</table>
### Stilar Prototype

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetCss</td>
<td>1 010,55</td>
<td>150 000</td>
<td>148 434,021</td>
</tr>
<tr>
<td>GetCss</td>
<td>1 056,1</td>
<td>100 000</td>
<td>94 688,003</td>
</tr>
<tr>
<td>Height</td>
<td>1 502,6</td>
<td>150 000</td>
<td>99 826,967</td>
</tr>
<tr>
<td>Width</td>
<td>1 589,65</td>
<td>150 000</td>
<td>94 360,394</td>
</tr>
<tr>
<td>Is</td>
<td>7 091,05</td>
<td>2 500 000</td>
<td>352 557,097</td>
</tr>
<tr>
<td>Show</td>
<td>4 728,6</td>
<td>2 000 000</td>
<td>422 958,169</td>
</tr>
<tr>
<td>Hide</td>
<td>1 645,05</td>
<td>300 000</td>
<td>182 365,278</td>
</tr>
<tr>
<td>Toggle</td>
<td>8 060,65</td>
<td>1 000 000</td>
<td>124 059,474</td>
</tr>
</tbody>
</table>

### Event jQuery

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bind</td>
<td>1477</td>
<td>1 000</td>
<td>677 048</td>
</tr>
<tr>
<td>Trigger</td>
<td>1 230</td>
<td>30 000</td>
<td>24 390,244</td>
</tr>
<tr>
<td>Unbind</td>
<td>1 025</td>
<td>2 300</td>
<td>2 243,902</td>
</tr>
</tbody>
</table>

### Event Prototype

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observe</td>
<td>1 056,10</td>
<td>50 000</td>
<td>47 344,002</td>
</tr>
<tr>
<td>Fire</td>
<td>866,05</td>
<td>100 000</td>
<td>115 466,774</td>
</tr>
<tr>
<td>StopObserving</td>
<td>853,55</td>
<td>25 000</td>
<td>29 289,438</td>
</tr>
</tbody>
</table>

#### 9.1.4 Opera 11.10

### Traversering jQuery

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent</td>
<td>935,05</td>
<td>40 000</td>
<td>42 778,46</td>
</tr>
<tr>
<td>Parents</td>
<td>1 726,55</td>
<td>5</td>
<td>2,90</td>
</tr>
<tr>
<td>Prev</td>
<td>1 505,45</td>
<td>59 600</td>
<td>39 589,49</td>
</tr>
<tr>
<td>PrevAll</td>
<td>1 237,65</td>
<td>20 000</td>
<td>16 159,66</td>
</tr>
<tr>
<td>Next</td>
<td>789,45</td>
<td>44 700</td>
<td>56 621,70</td>
</tr>
<tr>
<td>NextAll</td>
<td>1 288,40</td>
<td>20 000</td>
<td>15 523,13</td>
</tr>
<tr>
<td>Siblings</td>
<td>965,90</td>
<td>15 000</td>
<td>15 529,56</td>
</tr>
<tr>
<td>Children</td>
<td>944,50</td>
<td>40 000</td>
<td>42 350,45</td>
</tr>
</tbody>
</table>

### Traversering Prototype

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up</td>
<td>896,05</td>
<td>300 000</td>
<td>334 802,75</td>
</tr>
<tr>
<td>Ancestors</td>
<td>770,80</td>
<td>30 000</td>
<td>38 920,60</td>
</tr>
<tr>
<td>ChildElements</td>
<td>796,10</td>
<td>149 000</td>
<td>187 162,42</td>
</tr>
<tr>
<td>Next</td>
<td>919,50</td>
<td>298 000</td>
<td>324 089,18</td>
</tr>
<tr>
<td>Siblings</td>
<td>1 031,20</td>
<td>20 000</td>
<td>19 394,88</td>
</tr>
<tr>
<td>PreviousSiblings</td>
<td>774,60</td>
<td>21 000</td>
<td>27 110,77</td>
</tr>
<tr>
<td>Previous</td>
<td>904,55</td>
<td>298 000</td>
<td>329 445,58</td>
</tr>
<tr>
<td>NextSiblings</td>
<td>766,40</td>
<td>21 000</td>
<td>27 400,84</td>
</tr>
<tr>
<td>Manipulation jQuery</td>
<td>Funktion</td>
<td>Medelvärde(ms)</td>
<td>Antal iterationer</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------</td>
<td>----------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Html</td>
<td></td>
<td>850,25</td>
<td>35 000</td>
</tr>
<tr>
<td>Before</td>
<td></td>
<td>1 245,85</td>
<td>25 000</td>
</tr>
<tr>
<td>After</td>
<td></td>
<td>1 125,35</td>
<td>12 000</td>
</tr>
<tr>
<td>Prepend</td>
<td></td>
<td>1 154,55</td>
<td>10 000</td>
</tr>
<tr>
<td>Append</td>
<td></td>
<td>1 169</td>
<td>7 000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Manipulation Prototype</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Html</td>
<td></td>
<td>974,10</td>
<td>15 000</td>
<td>15 398,83</td>
</tr>
<tr>
<td>Before</td>
<td></td>
<td>883,15</td>
<td>6 000</td>
<td>6 793,86</td>
</tr>
<tr>
<td>After</td>
<td></td>
<td>1 382,45</td>
<td>5 500</td>
<td>3 978,44</td>
</tr>
<tr>
<td>Prepend</td>
<td></td>
<td>1 507,40</td>
<td>5 000</td>
<td>3 316,97</td>
</tr>
<tr>
<td>Append</td>
<td></td>
<td>2 193,50</td>
<td>3 000</td>
<td>1 367,68</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribut jQuery</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddClass</td>
<td></td>
<td>977</td>
<td>2 500</td>
<td>2 574,67</td>
</tr>
<tr>
<td>RemoveClass</td>
<td></td>
<td>2 535,65</td>
<td>5 000</td>
<td>1 971,88</td>
</tr>
<tr>
<td>HasClass</td>
<td></td>
<td>1 962,40</td>
<td>5 000</td>
<td>2 547,90</td>
</tr>
<tr>
<td>GetAttr</td>
<td></td>
<td>1 359,80</td>
<td>20 000</td>
<td>14 708,05</td>
</tr>
<tr>
<td>SetAttr</td>
<td></td>
<td>1 161,35</td>
<td>200 000</td>
<td>172 213,37</td>
</tr>
<tr>
<td>RemoveAttr</td>
<td></td>
<td>946,25</td>
<td>130 000</td>
<td>137 384,41</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribut Prototype</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddClassName</td>
<td></td>
<td>762,20</td>
<td>2 500</td>
<td>3 279,98</td>
</tr>
<tr>
<td>RemoveClassName</td>
<td></td>
<td>1 380,45</td>
<td>4 000</td>
<td>2 897,61</td>
</tr>
<tr>
<td>HasClassName</td>
<td></td>
<td>1 227,20</td>
<td>5 000</td>
<td>4 074,32</td>
</tr>
<tr>
<td>ReadAttribute</td>
<td></td>
<td>1 104</td>
<td>7 000</td>
<td>6 340,58</td>
</tr>
<tr>
<td>WriteAttribute</td>
<td></td>
<td>1 006,50</td>
<td>170 000</td>
<td>168 902,14</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stilar jQuery</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetCss</td>
<td></td>
<td>1 030,75</td>
<td>150 000</td>
<td>145 525,10</td>
</tr>
<tr>
<td>GetCss</td>
<td></td>
<td>1 097,80</td>
<td>120 000</td>
<td>109 309,53</td>
</tr>
<tr>
<td>Height</td>
<td></td>
<td>1 445,15</td>
<td>30 000</td>
<td>20 759,09</td>
</tr>
<tr>
<td>Width</td>
<td></td>
<td>1 440,90</td>
<td>30 000</td>
<td>20 820,32</td>
</tr>
<tr>
<td>Is</td>
<td></td>
<td>919,95</td>
<td>55 000</td>
<td>59 785,86</td>
</tr>
<tr>
<td>Show</td>
<td></td>
<td>878,65</td>
<td>500 000</td>
<td>569 054,80</td>
</tr>
<tr>
<td>Hide</td>
<td></td>
<td>960,40</td>
<td>100 000</td>
<td>104 123,28</td>
</tr>
<tr>
<td>Toggle</td>
<td></td>
<td>931,50</td>
<td>15 000</td>
<td>16 103,06</td>
</tr>
</tbody>
</table>
### Stilar Prototype

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetCss</td>
<td>5 557,95</td>
<td>2 000</td>
<td>359,84</td>
</tr>
<tr>
<td>GetCss</td>
<td>1 395,20</td>
<td>100 000</td>
<td>71 674,31</td>
</tr>
<tr>
<td>Height</td>
<td>883,60</td>
<td>45 000</td>
<td>50 928,02</td>
</tr>
<tr>
<td>Width</td>
<td>890,60</td>
<td>45 000</td>
<td>50 527,73</td>
</tr>
<tr>
<td>Is</td>
<td>903,10</td>
<td>550 000</td>
<td>609 013,40</td>
</tr>
<tr>
<td>Show</td>
<td>934,70</td>
<td>600 000</td>
<td>641 917,19</td>
</tr>
<tr>
<td>Hide</td>
<td>953,70</td>
<td>300 000</td>
<td>314 564,33</td>
</tr>
<tr>
<td>Toggle</td>
<td>907,80</td>
<td>200 000</td>
<td>220 312,84</td>
</tr>
</tbody>
</table>

### Event jQuery

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bind</td>
<td>1 054,45</td>
<td>150 000</td>
<td>142 254,26</td>
</tr>
<tr>
<td>Trigger</td>
<td>863,75</td>
<td>5 000</td>
<td>5 788,71</td>
</tr>
<tr>
<td>Unbind</td>
<td>877,3</td>
<td>3 500</td>
<td>3 989,51</td>
</tr>
</tbody>
</table>

### Event Prototype

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observe</td>
<td>999,15</td>
<td>12 000</td>
<td>12 010,21</td>
</tr>
<tr>
<td>Fire</td>
<td>1 496,95</td>
<td>50 000</td>
<td>33 401,25</td>
</tr>
<tr>
<td>StopObserving</td>
<td>828,9</td>
<td>30 000</td>
<td>36 192,54</td>
</tr>
</tbody>
</table>

### 9.1.5 Internet Explorer 9.0.8112.1421

#### Traversering jQuery

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent</td>
<td>993,55</td>
<td>20</td>
<td>20,13</td>
</tr>
<tr>
<td>Parents</td>
<td>51 383,75</td>
<td>1</td>
<td>0,02</td>
</tr>
<tr>
<td>Prev</td>
<td>924,7</td>
<td>223 500</td>
<td>241 700,01</td>
</tr>
<tr>
<td>PrevAll</td>
<td>938,1</td>
<td>2 500</td>
<td>2 664,96</td>
</tr>
<tr>
<td>Next</td>
<td>910,05</td>
<td>223 500</td>
<td>245 590,90</td>
</tr>
<tr>
<td>NextAll</td>
<td>933,2</td>
<td>2 500</td>
<td>2 678,95</td>
</tr>
<tr>
<td>Siblings</td>
<td>761,9</td>
<td>2 500</td>
<td>3 281,27</td>
</tr>
<tr>
<td>Children</td>
<td>1 067,3</td>
<td>250 000</td>
<td>234 235,92</td>
</tr>
</tbody>
</table>

#### Traversering Prototype

<table>
<thead>
<tr>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up</td>
<td>1 144,15</td>
<td>75000</td>
<td>65550,85</td>
</tr>
<tr>
<td>Ancestors</td>
<td>1 103</td>
<td>2000</td>
<td>1813,24</td>
</tr>
<tr>
<td>ChildElements</td>
<td>1 354,90</td>
<td>37250</td>
<td>27492,80</td>
</tr>
<tr>
<td>Next</td>
<td>1 385,40</td>
<td>74500</td>
<td>53775,08</td>
</tr>
<tr>
<td>Siblings</td>
<td>1 304,05</td>
<td>1500</td>
<td>1150,26</td>
</tr>
<tr>
<td>PreviousSiblings</td>
<td>1 273,50</td>
<td>1500</td>
<td>1177,86</td>
</tr>
<tr>
<td>Previous</td>
<td>1 1386</td>
<td>74500</td>
<td>53751,80</td>
</tr>
<tr>
<td>NextSiblings</td>
<td>1 263,75</td>
<td>1500</td>
<td>1186,94</td>
</tr>
<tr>
<td>Manipulation jQuery</td>
<td>Funktion</td>
<td>Medelvärde(ms)</td>
<td>Antal iterationer</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------</td>
<td>----------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Html</td>
<td>833,35</td>
<td>6 000</td>
<td></td>
</tr>
<tr>
<td>Before</td>
<td>848,40</td>
<td>7 500</td>
<td></td>
</tr>
<tr>
<td>After</td>
<td>848,40</td>
<td>7 500</td>
<td></td>
</tr>
<tr>
<td>Prepend</td>
<td>956,45</td>
<td>7 500</td>
<td></td>
</tr>
<tr>
<td>Append</td>
<td>863,90</td>
<td>7 500</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Manipulation Prototype</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>Html</td>
<td>968,65</td>
<td>6 000</td>
<td></td>
<td>6 194,19</td>
</tr>
<tr>
<td>Before</td>
<td>1 428,75</td>
<td>5 000</td>
<td></td>
<td>3 499,56</td>
</tr>
<tr>
<td>After</td>
<td>1 562,70</td>
<td>4 500</td>
<td></td>
<td>2 879,63</td>
</tr>
<tr>
<td>Prepend</td>
<td>1 681,40</td>
<td>4 000</td>
<td></td>
<td>2 378,97</td>
</tr>
<tr>
<td>Append</td>
<td>1 687,50</td>
<td>3 500</td>
<td></td>
<td>2 074,07</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribut jQuery</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddClass</td>
<td>810,25</td>
<td>2 000</td>
<td></td>
<td>2 468,37</td>
</tr>
<tr>
<td>RemoveClass</td>
<td>966,30</td>
<td>2 000</td>
<td></td>
<td>2 069,75</td>
</tr>
<tr>
<td>HasClass</td>
<td>1 030,80</td>
<td>3 000</td>
<td></td>
<td>2 910,36</td>
</tr>
<tr>
<td>GetAttr</td>
<td>1 382,30</td>
<td>7 000</td>
<td></td>
<td>5 064,02</td>
</tr>
<tr>
<td>SetAttr</td>
<td>925,65</td>
<td>40 000</td>
<td></td>
<td>43 212,88</td>
</tr>
<tr>
<td>RemoveAttr</td>
<td>888,55</td>
<td>35 000</td>
<td></td>
<td>39 390,02</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribut Prototype</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddClassName</td>
<td>1 146,75</td>
<td>1 800</td>
<td></td>
<td>1 569,65</td>
</tr>
<tr>
<td>RemoveClassName</td>
<td>1 235,80</td>
<td>800</td>
<td></td>
<td>647,35</td>
</tr>
<tr>
<td>HasClassName</td>
<td>1 075,80</td>
<td>600</td>
<td></td>
<td>557,72</td>
</tr>
<tr>
<td>ReadAttribute</td>
<td>988,90</td>
<td>5 000</td>
<td></td>
<td>5 056,12</td>
</tr>
<tr>
<td>WriteAttribute</td>
<td>1 018,30</td>
<td>35 000</td>
<td></td>
<td>34 371,01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stilar jQuery</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetCss</td>
<td>796</td>
<td>15 000</td>
<td></td>
<td>18 844,22</td>
</tr>
<tr>
<td>GetCss</td>
<td>1 003,10</td>
<td>70 000</td>
<td></td>
<td>69 783,67</td>
</tr>
<tr>
<td>Height</td>
<td>935,70</td>
<td>15 000</td>
<td></td>
<td>16 030,78</td>
</tr>
<tr>
<td>Width</td>
<td>933,70</td>
<td>15 000</td>
<td></td>
<td>16 065,12</td>
</tr>
<tr>
<td>Is</td>
<td>1 484,30</td>
<td>50 000</td>
<td></td>
<td>33 685,91</td>
</tr>
<tr>
<td>Show</td>
<td>963,40</td>
<td>100 000</td>
<td></td>
<td>103 799,05</td>
</tr>
<tr>
<td>Hide</td>
<td>1 306,10</td>
<td>100 000</td>
<td></td>
<td>76 563,82</td>
</tr>
<tr>
<td>Toggle</td>
<td>1 442</td>
<td>7 000</td>
<td></td>
<td>4 854,37</td>
</tr>
<tr>
<td>Stilar Prototype</td>
<td>Funktion</td>
<td>Medelvärde(ms)</td>
<td>Antal Iterationer</td>
<td>Körningar per sekund</td>
</tr>
<tr>
<td>------------------</td>
<td>----------</td>
<td>---------------</td>
<td>-------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td></td>
<td>SetCss</td>
<td>889</td>
<td>30 000</td>
<td>33 745,78</td>
</tr>
<tr>
<td></td>
<td>GetCss</td>
<td>934,80</td>
<td>40 000</td>
<td>42 789,90</td>
</tr>
<tr>
<td></td>
<td>Height</td>
<td>5 476,90</td>
<td>10 000</td>
<td>1 825,85</td>
</tr>
<tr>
<td></td>
<td>Width</td>
<td>14 129,40</td>
<td>7 000</td>
<td>495,42</td>
</tr>
<tr>
<td></td>
<td>Is</td>
<td>1 041,70</td>
<td>75 000</td>
<td>71 997,70</td>
</tr>
<tr>
<td></td>
<td>Show</td>
<td>930,50</td>
<td>65 000</td>
<td>69 854,92</td>
</tr>
<tr>
<td></td>
<td>Hide</td>
<td>947,70</td>
<td>65 000</td>
<td>68 587,11</td>
</tr>
<tr>
<td></td>
<td>Toggle</td>
<td>1 076,30</td>
<td>25 000</td>
<td>23 227,72</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Event jQuery</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Bind</td>
<td>1 053,15</td>
<td>1 000</td>
<td>949,53</td>
</tr>
<tr>
<td></td>
<td>Trigger</td>
<td>742,95</td>
<td>7 000</td>
<td>9 421,90</td>
</tr>
<tr>
<td></td>
<td>Unbind</td>
<td>1 149,50</td>
<td>15 000</td>
<td>13 049,15</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Event Prototype</th>
<th>Funktion</th>
<th>Medelvärde(ms)</th>
<th>Antal Iterationer</th>
<th>Körningar per sekund</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Observe</td>
<td>1 001,50</td>
<td>20 000</td>
<td>19 970,04</td>
</tr>
<tr>
<td></td>
<td>Fire</td>
<td>977,20</td>
<td>7 000</td>
<td>7 163,32</td>
</tr>
<tr>
<td></td>
<td>StopObserving</td>
<td>1 307,85</td>
<td>12 000</td>
<td>9 175,36</td>
</tr>
</tbody>
</table>
9.2 Beräkningar

Tabell för att jämföra biblioteken på körningar per sekund totalt sett över alla webbläsare.

<table>
<thead>
<tr>
<th>Totalt medel över alla test</th>
<th>jQuery</th>
<th>Prototype</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traversering</td>
<td>105 615,143</td>
<td>211217,905</td>
</tr>
<tr>
<td>Manipulation</td>
<td>28 320,243</td>
<td>12540,691</td>
</tr>
<tr>
<td>Attribut</td>
<td>81 486,670</td>
<td>130407,498</td>
</tr>
<tr>
<td>Stilar</td>
<td>83 377,955</td>
<td>273002,206</td>
</tr>
<tr>
<td>Event</td>
<td>18 463,287</td>
<td>37552,853</td>
</tr>
</tbody>
</table>

Tabell för att jämföra de båda biblioteken i alla webbläsare baserat på ett medelvärde av alla testfalls medelvärde.

<table>
<thead>
<tr>
<th>Biblioteksjämförelse över alla webbläsare</th>
<th>Firefox</th>
<th>Chrome</th>
<th>Safari</th>
<th>Opera</th>
<th>IE</th>
</tr>
</thead>
<tbody>
<tr>
<td>jQuery</td>
<td>59 174,50</td>
<td>86 407,4</td>
<td>82 053,56</td>
<td>56 494,83</td>
<td>33 132,99</td>
</tr>
<tr>
<td>Prototype</td>
<td>84 340,54</td>
<td>356 374,5</td>
<td>110 970,9</td>
<td>95 284,9</td>
<td>17 750,29</td>
</tr>
</tbody>
</table>

Tabell för att jämföra de båda biblioteken i testfallet traversering.

<table>
<thead>
<tr>
<th>Medelvärde för alla funktioner, traversering</th>
<th>Firefox</th>
<th>Chrome</th>
<th>Safari</th>
<th>Opera</th>
<th>IE</th>
</tr>
</thead>
<tbody>
<tr>
<td>jQuery</td>
<td>158 351,123</td>
<td>134 806,577</td>
<td>115 077,076</td>
<td>28 569,418</td>
<td>91 271,521</td>
</tr>
<tr>
<td>Prototype</td>
<td>149 575,632</td>
<td>624 415,879</td>
<td>95 319,783</td>
<td>161 040,876</td>
<td>25 737,354</td>
</tr>
</tbody>
</table>

Tabell för att jämföra de båda biblioteken i testfallet manipulation.

<table>
<thead>
<tr>
<th>Medelvärde för alla funktioner, manipulation</th>
<th>Firefox</th>
<th>Chrome</th>
<th>Safari</th>
<th>Opera</th>
<th>IE</th>
</tr>
</thead>
<tbody>
<tr>
<td>jQuery</td>
<td>26 118,065</td>
<td>38 529,144</td>
<td>51 364,607</td>
<td>17 308,748</td>
<td>8 280,651</td>
</tr>
<tr>
<td>Prototype</td>
<td>10 461,576</td>
<td>20 391,368</td>
<td>22 274,068</td>
<td>6 171,157</td>
<td>3 405,285</td>
</tr>
</tbody>
</table>

Tabell för att jämföra de båda biblioteken i testfallet attribut.

<table>
<thead>
<tr>
<th>Medelvärde för alla funktioner, attribut</th>
<th>Firefox</th>
<th>Chrome</th>
<th>Safari</th>
<th>Opera</th>
<th>IE</th>
</tr>
</thead>
<tbody>
<tr>
<td>jQuery</td>
<td>49 575,026</td>
<td>135 918,706</td>
<td>150 853,671</td>
<td>55 233,379</td>
<td>15 852,567</td>
</tr>
<tr>
<td>Prototype</td>
<td>90 380,697</td>
<td>332 796,224</td>
<td>183 321,271</td>
<td>37 098,923</td>
<td>8 440,373</td>
</tr>
</tbody>
</table>

Tabell för att jämföra de båda biblioteken i testfallet stilar.

<table>
<thead>
<tr>
<th>Medelvärde för alla funktioner, stilar</th>
<th>Firefox</th>
<th>Chrome</th>
<th>Safari</th>
<th>Opera</th>
<th>IE</th>
</tr>
</thead>
<tbody>
<tr>
<td>jQuery</td>
<td>50 798,226</td>
<td>109 084,314</td>
<td>83 868,739</td>
<td>130 685,130</td>
<td>42 453,366</td>
</tr>
<tr>
<td>Prototype</td>
<td>139 914,931</td>
<td>751 212,167</td>
<td>189 906,175</td>
<td>244 912,210</td>
<td>39 065,550</td>
</tr>
</tbody>
</table>
Tabell för att jämföra de båda biblioteken i testfallet eventrespons.

<table>
<thead>
<tr>
<th></th>
<th>Firefox</th>
<th>Chrome</th>
<th>Safari</th>
<th>Opera</th>
<th>IE</th>
</tr>
</thead>
<tbody>
<tr>
<td>jQuery</td>
<td>11 030,068</td>
<td>13 698,283</td>
<td>9 103,731</td>
<td>50 677,494</td>
<td>7 806,861</td>
</tr>
<tr>
<td>Prototype</td>
<td>31 369,857</td>
<td>53 056,757</td>
<td>64 033,405</td>
<td>27 201,334</td>
<td>12 102,911</td>
</tr>
</tbody>
</table>

9.3 Testfallsdata


9.3.1 jQuery Index

```html
<html>
<head>
  <meta http-equiv="content-type" content="text/html;charset=utf-8" />
  <title>Test Cases</title>

  <!--link to library -->
  <script type="text/javascript" src="jquery.js"></script>

  <!--links to testcases -->
  <script type="text/javascript" src="jQueryTraversal.js"></script>
  <script type="text/javascript" src="jQueryManipulation.js"></script>
  <script type="text/javascript" src="jQueryEvent.js"></script>
  <script type="text/javascript" src="jQueryAttribute.js"></script>
  <script type="text/javascript" src="jQueryStyle.js"></script>
  <script type="text/javascript" src="emptyIterators.js"></script>

  <!--Root tag, all tests will run within this div -->
  <div id="root"></div>

  <!--Defining start and stop functions for the timers
  var timeDiff = {
    setStartTime:function(){
      d = new Date();
      time = d.getTime();
    },
    getDiff:function (){d = new Date();
      return (d.getTime()-time);
    }
  }

  //Creating a new testcase
  var testcase = new emptyIterators();

  //Run the testcase
  var result = testcase.runTestCase(20);
```
9.3.2 jQuery Traversal

/*
 * Testcase jQueryTraversal
 * -------------------------
 * -Library jQuery
 * -Libraryversion 1.5.2
 * -Testcasversion 1.0
 * -Authors Petter Andersson, Eric Ericsson
 * -Date 2011-05-10
 * -Description
 * This testcase tests jQuerys ability to traverse DOM trees.
 * - Using functions: Parent, Parents, Prev, PrevAll, Next, NextAll,
 * Siblings, Children
 * *
 */

function jQueryTraversal() {


}
timeDiff.setStartTime();
eval("this.test"+this.tests[i]+"()" );
this.timers[i][j] = timeDiff.getDiff();
this.clean(i);
}

this.tearDown();
return this.calculateResult();
}
/*
*  =========================================================================
*  *  Setup and tearDown
*  =========================================================================
*/
this.setup = function() {

    this.tests = new Array("Parent","Parents","Prev","PrevAll","Next","NextAll","Siblings","Children");
    this.timers = new Array();
    this.result = new Array();
    for(i=0;i<this.tests.length;i++) {
        this.timers[i] = new Array();
        this.result[i] = new Array();
    }

    totalnrOfElem = 0;
    nrOfElem = 150;
    root = $('div');
    root.attr('id','root');

    var node1;
    var node2;
    var node3;
    var node4;

    for(i=0; i<nrOfElem; i++) {
        node1 = $('<div>');
        node1.attr('class','ch1-'+i);
        for(j=0; j<nrOfElem*20; j++) {
            node2 = $('<div>');
            node2.attr('class','ch2-'+j);
            node1.append(node2);
            totalnrOfElem++;
        }
        root.append(node1);
        totalnrOfElem++;
    }

    node2 = $('.ch2-8');
    for(k=0; k<100; k++) {
        node3 = $('"<div>"');
        node3.attr('class','ch3-'+k);
        node2.append(node3);
        totalnrOfElem++;
    }

    node3 = $('".ch3-6"');
    for(k=0; k<100; k++) {
        node4 = $('"<div>"');
        node4.attr('class','ch4-'+k);
        node3.append(node4);
        totalnrOfElem++;
    }

    totalnrOfElem++;
}
var nodes;
var tmp;

node4 = $('.ch4-3');
tmp = $(node4);

for(i=0; i<100; i++){
    nodes = $('"<div>"');
    nodes.attr('class','ch'+(i+5)+'+'+0);
    tmp.append($(nodes));
    tmp = $('.ch'+(i+5)+'+'+0);
    totalnrOfElem++;
}

testParentRoot = $('".ch104-0'"');
testPrevRoot = $('".ch1-149'"');
testNextRoot = $('".ch1-0'"');
testSiblingsRoot = $('".ch1-50'"');
testChildrenRoot = $('".ch4-3'"');

this.tearDown = function() {
    var node = document.body;
    if (node.hasChildNodes()) {
        while (node.childNodes.length >= 1) {
            node.removeChild(node.firstChild);
        }
    }
}

this.build = function(testId) {
}

this.clean = function(testId) {
}

/*----------------------------------------------------------------------------
 * - Result calculation
 *---------------------------------------------------------------------------
 */
this.calculateResult = function() {
    var testcaseResult = new Array();
    var totalTimer = 0;
    for(i=0;i<this.tests.length;i++) {
        totalTimer = 0;
        for(j=0;j<this.timers[i].length;j++) {
            totalTimer = totalTimer + this.timers[i][j];
        }
        this.result[i][0] = this.tests[i];
        this.result[i][1] = totalTimer / this.timers[i].length;
    }
    testcaseResult[0] = this.result;
    testcaseResult[1] = this.tests;
    testcaseResult[2] = this.timers;
    return testcaseResult;
}
this.testParent = function() {
    var node1 = testParentRoot;
    for(z=0;z<20;z++) {
        node1 = node1.parent();
    }
}

this.testParents = function() {
    var leaf = testParentRoot;
    var node2 = leaf.parents();
}

this.testPrev = function() {
    var node3 = testPrevRoot;
    for(yy=0;yy<1500;yy++) {
        for(zz=0;zz<149;zz++){
            node3 = node3.prev();
        }
    }
}

this.testPrevAll = function() {
    var root1 = testPrevRoot;
    for(xxx=0;xxx<2500;xxx++) {
        var node4 = root1.prevAll();
    }
}

this.testNext = function() {
    var node5 = testNextRoot;
    for(yyy=0; yyy<1500; yyy++) {
        for(zzz=0; zzz<149; zzz++){
            node5 = node5.next();
        }
    }
}

this.testNextAll = function() {
    var root2 = testNextRoot;
    for(xx=0;xx<2500;xx++) {
        var node6 = root2.nextAll();
    }
}

this.testSiblings= function() {
    var root3 = testSiblingsRoot;
    for(x=0;x<2500;x++) {
        var node7 = root3.siblings();
    }
}

this.testChildren = function() {
    var node8 = testChildrenRoot;
    for(yyyy=0;yyyy<2500;yyyy++){
        for(zzzz=0;zzzz<100;zzzz++){
            node8 = node8.children();
        }
    }
}