

Introduktion til JavaScript & jQuery



Af Finn Vilsbæk

Medieskolernes Webintegrator Uddannelse, Efterår 2017

Indholdsfortegnelse

Grundlæggende JavaScript	5
Placering af JavaScript	8
Variabler	9
Betingelser	11
Switch	14
Popups	17
Funktioner	18
Løkker - loops	21
For loop.....	21
While.....	23
For In.....	24
Events og vinduer	25
Try – Catch konstruktionen	28
Datoer	29
Math	30
Formvalidering.....	35
Formvalidering: CPR Check.....	36
Timing	44
Creating objects.....	45
Afspilning af medier.....	48
Get Element.....	51
Dynamic Style	56
Opgaver til JavaScript	61
Opgave 1.....	61
Opgave 2.....	61
Opgave 3.....	61
Opgave 4.....	61
Opgave 5.....	61
Opgave 6.....	62
Opgave 7.....	62
Opgave 8.....	62
Opgave 9.....	62

Opgave 10.....	62
Opgave 11.....	63
Opgave 12.....	63
Opgave 13.....	63
Opgave 14.....	63
Kald af funktioner fra formelementer.....	64
Klokken der går.....	65
Højre klik forbudt.....	66
Countdown.....	67
Keypress.....	68
Udskriv side.....	69
Tilfældigt billede.....	69
Bannerrotation.....	71
Galleri.....	72
SlideInSideMenu.....	74
Slide menu med ikoner.....	77
Introduktion til jQuery.....	82
Valg af elementer.....	85
Element Selectors.....	85
Attribute Selectors.....	85
CSS Selectors.....	86
Events.....	86
CSS manipulation.....	87
Læsning af en CSS-property.....	87
Tilføj en værdi til en CSS-property.....	89
Tilføj værdier til flere CSS-properties ad gangen.....	90
Effekter.....	91
Slide & Fade.....	95
HTML Manipulation.....	100
Attributter.....	101
AJAX.....	104
Plug-ins.....	107
jQuery .on – event handler scripts.....	108
Custom events med jQuery .on.....	111

jQuery galleri	112
Scroll navigation	117
Fade navigation	119

Grundlæggende JavaScript

Oprindeligt hed Javascript 'Mocha', og den første fungerende prototype blev udviklet på kun 10 dage i maj 1995 af Brendan Eich.

Eich havde fået til opgave af Marc Andreessen, stifteren af det nu hedengangne firma Netscape, at lave et scripting sprog til browseren, som ikke-programmører kunne anvende til at føje interaktivitet til HTML-sider. Det kunne *kun* gå for langsomt, da Netscape var midt i 'the browser wars' hvor de var oppe imod store, slemme Microsoft og deres Internet Explorer browser – så prototypen på Javascript, Mocha blev smækket ind i Netscape Communicator 2.0 i sommeren 1995 og sluppet løs på en intetanende verden, der på dette tidspunkt så småt var begyndt at opdage et sæt helt nye ord: man kunne *surfe* med en *browser*, der brugte *hyperlinks*.

Et par måneder senere skiftede Mocha navn til 'LiveScript', og i december 1995 skiftede sproget navn igen til Javascript, da Netscape på dette tidspunkt havde indgået en aftale med Sun om at præsentere JS som et scripting sprog til at klare små client-side tasks i browseren, mens det 'rigtige' programmeringssprog Java ville blive præsenteret som det større, professionelle redskab til at bygge 'rich web components' med. Hvis du har hørt ordet Java i andre sammenhæng må du aldrig forveksle det med JavaScript. Java hører til i den tunge ende af programmeringssprog og bliver blandt andet brugt til at udvikle Windows programmer lige som C++ og .Net / C#.

JavaScript er altså oprindeligt blevet udviklet for at føje interaktivitet til HTML-sider, lige som CSS er udviklet til at style HTML-sider.

JavaScript er et såkaldt *scripting* sprog, det vil sige at det bliver skrevet i script blokke rundt omkring på dine HTML-sider, som regel afgrænset af `<script type="text/javascript"></script>` tags, der tillader browseren at parse den mellemliggende tekst som programkode.

Fast-forward til i dag: efter en længere standardiseringsøvelse over de mellemliggende 22-23 år hedder JavaScript i dag stadig JavaScript, men også ECMAScript, og er nu i skikkelse af ECMAScript version 6.0 i stadig fremdrift: nyskabelser som Node.js tillader os nu at bruge JS server side, og via interaktion med HTML5 API'er kan vi styre brugeroplevelserne meget mere præcist, åbne web sockets til 'always-on' kommunikation med klientsiden, skaffe geolokationsdata og udnytte mobilens særlige features som accelerometer og meget mere.

"It is an exciting time to learn JavaScript."

Men før vi går ombord i mobilens herligheder, vil jeg starte med en hel del eksempler i browseren. Vi begynder med et 'Hello World' eksempel:

```
Ex_01.html  ↗  ✕
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6  </head>
7  <body>
8      <script type="text/javascript">
9          document.write("Hej verden :-");
10     </script>
11 </body>
12 </html>
```

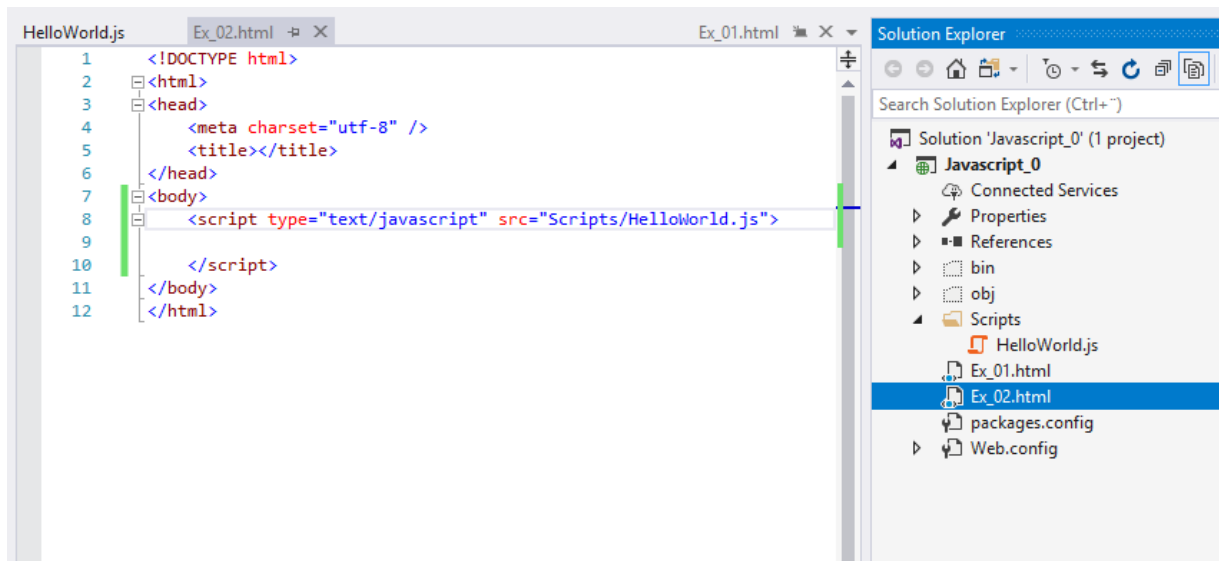
I det første eksempel her stifter vi bekendtskab med funktionen `document.write`, som er den funktion som bruges når noget skal udskrives på en html side. Byg eksemplet ovenfor, og test det i din browser.

Prøv at lave en 'Scripts' folder i dit webprojekt, og gem en fil 'HelloWorld' der – husk endelsen .js:

```
HelloWorld.js  ↗  ✕  Ex_02.html
Javascript_0
1  document.write("Hej verden :-");
```

Nu kan du kalde indholdet af din script fil i html dokumentet ved at angive `src` attributten:

```
<script type="text/javascript" src="Scripts/HelloWorld.js">
```



Test resultatet i din browser.

Placering af JavaScript

JavaScript kan placeres flere steder på en side, et script som er placeret i sidens body vil blive afviklet når sidens indhold indlæses i browseren. Scripts, som er placeret i sidens head tag vil blive 'indlæst' inden HTML sidens indhold bliver renderet i browservinduet.

Du kan placere JavaScript både i head og i body på samme tid, det bruges typisk for at få indlæst metoder i sidens head og så kalde den fra sidens body. På denne måde sikrer man sig at ens metoder er indlæst inden de bliver kaldt.

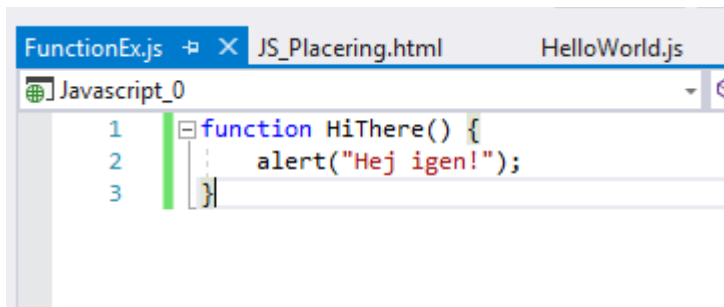
Det er også muligt at placere script i eksterne filer med henblik på genbrug af kode, ligesom i CSS. Disse filer hedder .js til efternavn og kan indeholde metoder som kan bruges igen og igen. Filerne kan importeres i sidens head sektion.

Her er et eksempel, hvor vi benytter os af 1) et script indlæst i HEAD tag'et (sætter en funktion op til senere brug), 2) et inline script i BODY tag'et og 3) et funktionskald i BODY tag'et.

Lav følgende HTML side:

```
FunctionEx.js  JS_Placering.html  HelloWorld.js  Ex_02.html
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <meta charset="utf-8" />
5    <title></title>
6    <script type="text/javascript" src="Scripts/FunctionEx.js"></script>
7  </head>
8  <body>
9
10   <!-- inline script -->
11   <a href="#" onclick='javascript:return alert("Hej med dig!");'>Halli Hallo</a>
12   <br />
13   <!-- funktionskald - ref. til fil loaded i head tag -->
14   <a href="#" onclick="HiThere()">Hi There</a>
15
16 </body>
17 </html>
```


.. og placer en fil mere i din Scripts mappe, 'FunctionEx.js':



```
FunctionEx.js JS_Placering.html HelloWorld.js
Javascript_0
1 function HiThere() {
2     alert("Hej igen!");
3 }
```

Test resultatet i din browser.

Variabler

En variabel er et lille stykke hukommelse som man kan navngive. Variablen opbevarer data imens en side kører. Herunder er der vist et par eksempler på brug af variabler.



```
Variabler_01.html
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="utf-8" />
5     <title></title>
6 </head>
7 <body>
8     <script type="text/javascript">
9         var test1 = "Sic transit";
10        test2 = "gloria mundi";
11        document.write(test1 + " " + test2);
12    </script>
13 </body>
14 </html>
```

Som du nok kan se i linie 9 og 10 er der to måder at skrive variabler på. I hver af variablerne navngivet test1 og test2 fylder vi tekst i. Læg mærke til at der er citationstegn (" ") omkring teksten, det betyder at variabelens datatype skal være af typen tekst.

I linie 11 udskriver vi indholdet af vores variabler adskilt af et mellemrum.

Der findes mange andre datatyper end tekst - dem kommer vi løbende ind på gennem materialet. I det næste eksempel bruger vi tal i variabler.

Læg mærke til at der i dette tilfælde ikke er citationstegn (" ") om vores tal. Hvis der havde været det, ville deres datatype været tekst - og vi ville ikke kunne bruge dem til at regne med.

```
Variable_02.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6  </head>
7  <body>
8      <script type="text/javascript">
9          var tal1 = 22;
10         var tal2 = 10;
11         document.write("Tal 1 PLUS tal 2 er: " + (tal1 + tal2));
12         document.write("<br/>");
13         document.write("Tal 1 MINUS tal 2 er: " + (tal1 - tal2));
14         document.write("<br/>");
15         document.write("Tal 1 GANGE tal 2 er: " + (tal1 * tal2));
16         document.write("<br/>");
17         document.write("Tal 1 DELT MED tal 2 er: " + (tal1 / tal2));
18     </script>
19 </body>
20 </html>
```

I linie 9 og 10 tildeler vi variablerne tal1 og tal2 hver deres værdi.

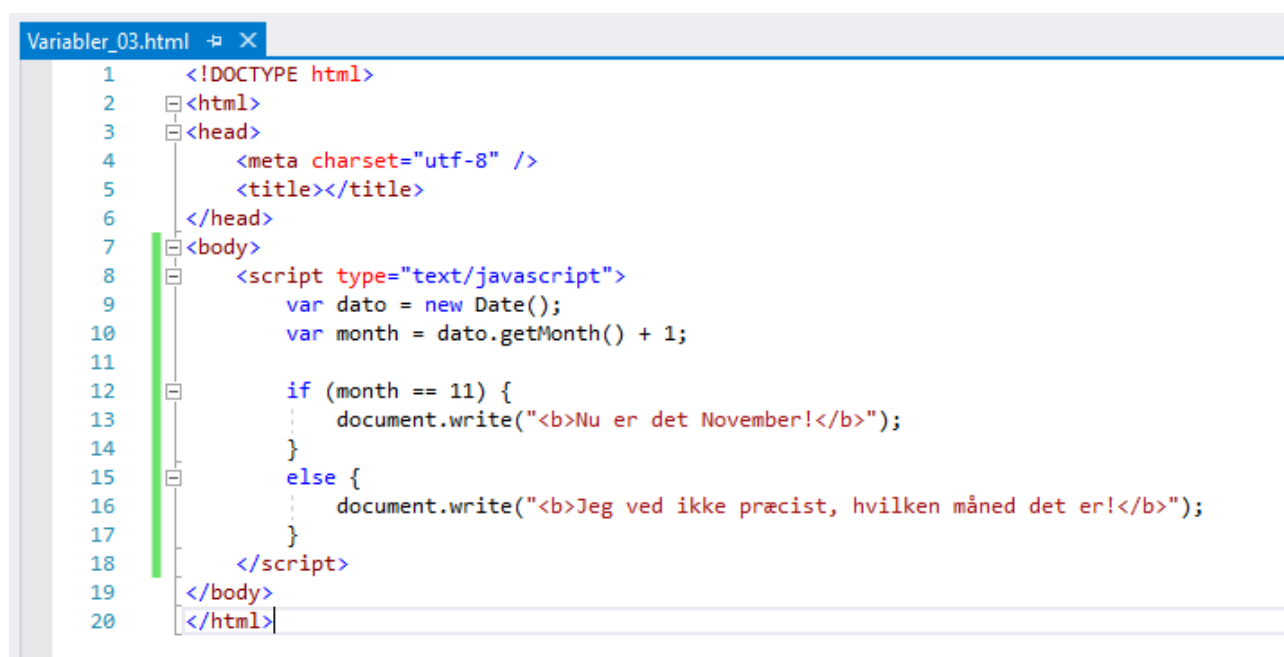
I linie 11 – 17 bruger vi *operanderne* plus, minus, gange og divider på vores variabler og adskiller resultaterne med et br-tag.

Skriv koden ovenfor ind i din html editor, og test resultatet i din browser.

Betingelser

Ved at lave såkaldte betingelser kan man afbryde og ændre afviklingen af kode. Ud fra en betingelse kan du bede browseren om at træffe et valg og alt efter om en betingelse er opfyldt eller ej kan du få udskrevet forskellige resultater.

Herunder er vist nogle eksempler på betingelser som vælger ud fra et tal.



```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6  </head>
7  <body>
8      <script type="text/javascript">
9          var dato = new Date();
10         var month = dato.getMonth() + 1;
11
12         if (month == 11) {
13             document.write("<b>Nu er det November!</b>");
14         }
15         else {
16             document.write("<b>Jeg ved ikke præcist, hvilken måned det er!</b>");
17         }
18     </script>
19 </body>
20 </html>
```

I linie 9 laver vi en variabel som vi tildeler datoen i dag som vi får returneret fra funktionen `new Date()`

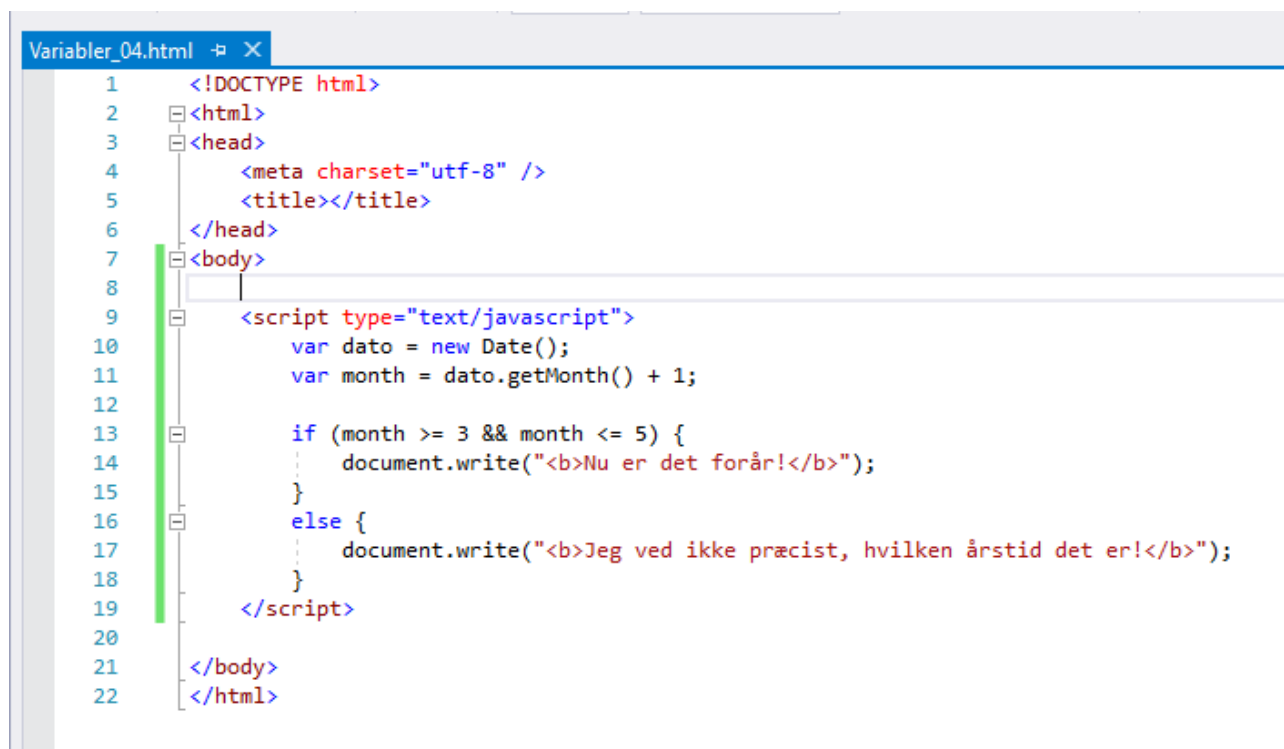
I linie 10 laver vi en variabel med navnet `month` og putter tallet for den aktuelle måned i som vi får ud fra `dato`, ved at bruge funktionen `.getMonth()`.

Så nu ligger der altså et tal i `month`, hvis vi nu siger det er august måned ville tallet så være 8. I linie 14 starter vi så vores betingelse hvor vi spørger om tallet i `month` er lig med 11.

Hvis tallet er lig med 11 vil linie 13 blive afviklet og vi ville få udskrevet teksten **"Nu er det November!"**

Hvis ikke `month` er lig med 11, vil vi få afviklet alt der står efter `else` og teksten **"Jeg ved ikke præcist, hvilken måned det er!"** vil blive udskrevet.

Det er også muligt at stille flere kriterier i en betingelse. Det er der vist et eksempel på her under. I eksemplet er kriterierne adskilt med && som betyder at begge kriterier skal være opfyldt. Hvis man satte || ind i stedet for && skulle blot et af dem være opfyldt.



```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6  </head>
7  <body>
8
9      <script type="text/javascript">
10         var dato = new Date();
11         var month = dato.getMonth() + 1;
12
13         if (month >= 3 && month <= 5) {
14             document.write("<b>Nu er det forår!</b>");
15         }
16         else {
17             document.write("<b>Jeg ved ikke præcist, hvilken årstid det er!</b>");
18         }
19     </script>
20
21 </body>
22 </html>
```

I linie 10 laver vi igen en variabel som vi tildeler datoen I dag som vi får returneret fra funktionen new Date(). I linie 11 laver vi en variabel med navnet month og putter tallet for den aktuelle måned i som vi får ud fra dato, ved at bruge funktionen .getMonth().

I linie 13 starter vi så vores betingelse hvor vi først spørger, om tallet i month er større end (>) eller lig med 3 og om month er mindre end (<) eller lig med 5.

Hvis tallet er lig med eller mellem 3 og 5 ville linie 14 blive afviklet og vi ville få udskrevet teksten: "Nu er det forår!".

Hvis ikke begge kriterier er opfyldt vil vi få afviklet alt der står efter mellem linie 16 - 18 og teksten: "Jeg ved ikke præcist, hvilken årstid det er!", vil blive udskrevet.

Det er også muligt at stille flere betingelser i en længere konstruktion, dvs. hvis en betingelse ikke er blevet opfyldt kan man blive sendt videre til den næste betingelse med et 'else if' statement.. Se eksemplet 'Variabler_05' på næste side.

```

Variabler_05.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6  </head>
7  <body>
8      <script type="text/javascript">
9          var dato = new Date();
10         var month = dato.getMonth() + 1;
11
12         if (month >= 3 && month <= 5) {
13             document.write("<b>Nu er det forår!</b>");
14         }
15         else if (month >= 6 && month <= 8) {
16             document.write("<b>Nu er det sommer</b>");
17         }
18         else {
19             document.write("<b>Det er efterår eller vinter!</b>");
20         }
21     </script>
22 </body>
23 </html>

```

Her er koden identisk med det foregående eksempel, indtil linie 15. Hvis ikke den første betingelse i linie 12 er blevet opfyldt bliver vi sendt videre til den næste i linie 15. Denne betingelse spørger, om tallet i month er større end (>) eller lig med 6, og om month er mindre end (<) eller lig med 8.

Hvis tallet er lig med eller mellem 6 og 8 vil linie 16 blive afviklet og vi vil få udskrevet teksten: "Nu er det sommer!".

Hvis ikke begge kriterier er opfyldt vil vi få afviklet alt der står efter mellem linie 18-20 og teksten: "Det er efterår eller vinter!" vil blive udskrevet.

Du kan evt. prøve at ændre i koden til det sidste eksempel, du har skrevet ind, så koden også tager højde for om vi er i December måned. Test resultatet i din browser.

Switch

Hvis man har mange sådanne betingelser der skal checkes for i en kodeblok, bliver det nemt indviklet at se på et større antal 'else if' statements - og her er en *switch struktur* en bedre løsning.

Strukturen er som vist her under, man putter en værdi i sin switch som så vælger hvad der skal udskrives ud fra den modtagne værdi.

```
Switch_statement.html X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6  </head>
7  <body>
8
9      <script type="text/javascript">
10         var dato = new Date();
11         var dag = dato.getDay();
12         switch (dag)
13         {
14             case 1:
15                 document.write("Mandag");
16                 break;
17             case 2:
18                 document.write("Tirsdag");
19                 break;
20             case 3:
21                 document.write("Onsdag");
22                 break;
23             case 4:
24                 document.write("Torsdag");
25                 break;
26             case 5:
27                 document.write("Fredag");
28                 break;
29
30             default:
31                 document.write("Nu er det weekend!");
32         }
33     </script>
34
35 </body>
36 </html>
```

I linie 10 laver vi igen en variabel, som vi tildeler datoen I dag som vi får returneret fra funktionen `new date()`.

I linie 11 laver vi en variabel med navnet `dag` og putter tallet for den aktuelle dag i som vi får ud fra `dato`, ved at bruge funktionen `.getDay()`, ligesom med månederne tidligere.

I linie 12 modtager vores `switch` så tallet fra vores variabel `dag`.

I linie 14 til 31 løbes alle `cases` igennem til den som har det aktuelle tal er blevet fundet.

Hvis vi nu siger der står fem i variabelen `dag`, ville vores browser stoppe ved linie 27 og afvikle alt hvad der står mellem `case` og `break`, altså udskrive dagens navn.

Hvis nu tallet ikke stod på listen ville vores `switch` vælge vores default `case` som starter i linie 30, og udskrive teksten fra linie 31.

Det er også muligt at få en `switch` til at vælge ud fra en tekst eller en dato. Herunder er vist et lille eksempel på valg ud fra en tekst.

```
Switch_statement_02.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6  </head>
7  <body>
8      <script type="text/javascript">
9          var browser = prompt("Skriv navnet på din browser:");
10
11         switch (browser.toLowerCase())
12         {
13             case "netscape":
14                 window.location = "http://www.netscape.com";
15                 break;
16             case "explorer":
17                 window.location = "http://microsoft.com";
18                 break;
19             case "firefox":
20                 window.location = "http://www.mozilla.com";
21                 break;
22
23             default:
24                 window.location = "http://www.panmedia.dk";
25                 break;
26         }
27     </script>
28
29 </body>
30 </html>
```

I linie 9 laver vi en ny variabel med navnet browser. I browser åbner vi en prompt som er en lille popup med et tekst felt hvor en bruger kan indtaste en værdi. Når der klikkes på knappen i prompten lægges det indtastede i variabelen browser.

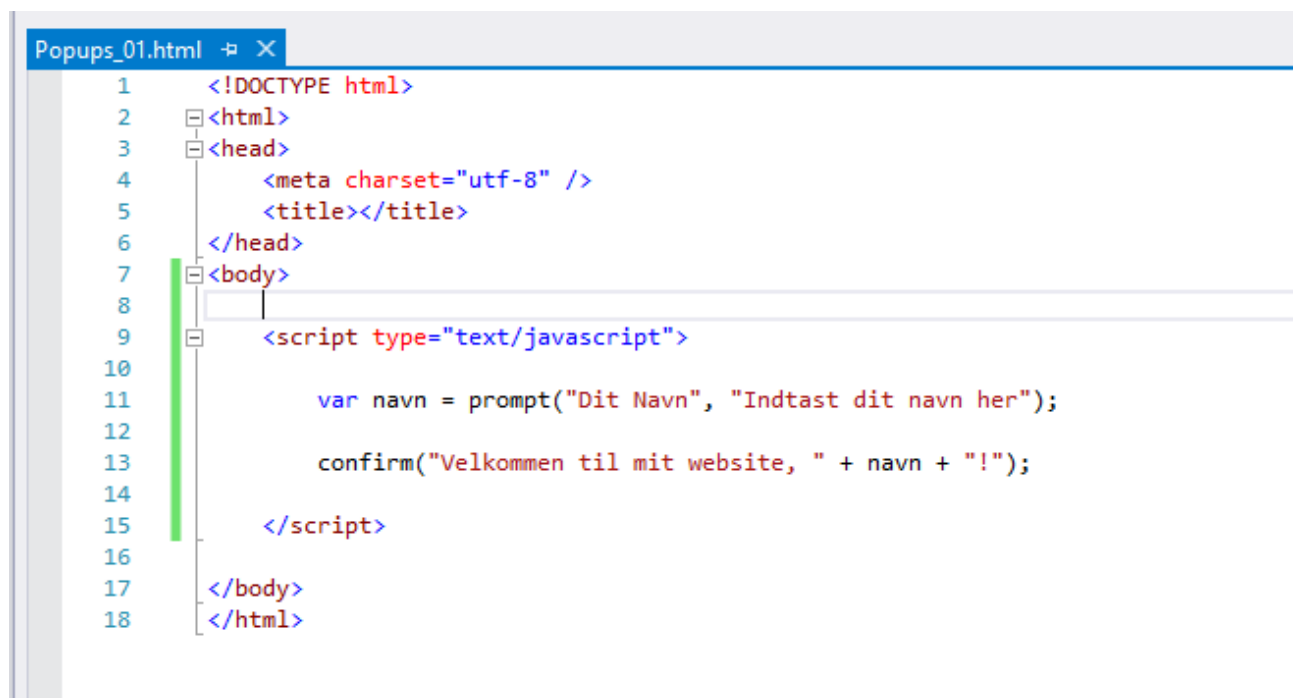
I linie 11 modtager vores switch værdien fra browser.

I linie 13 – 25 tjekker vi om værdien findes som case i vores switch. Hvis den gør vil vores browser vindue blive videresendt til den pågældende side. Hvis ikke værdien bliver fundet sendes vi videre til www.panmedia.dk.

Popups

Vi har lige stiftet bekendtskab med popup boxen prompt, og i de første kodeeksempler så du, hvordan du kan bruge en alert popup. Der er et par nyttige popups mere som er aldeles velegnet til at advare brugeren når der sker forskellige ting på ens side.

Det er fx muligt at give en bruger et valg, til det bruges funktionen confirm. Byg nedenstående eksempel, og test det i din browser:

A screenshot of a code editor window titled 'Popups_01.html'. The editor shows a mix of HTML and JavaScript code. Line numbers 1 through 18 are visible on the left. The code starts with a DOCTYPE declaration, followed by HTML tags for head and body. Inside the head, there's a meta charset declaration and a title tag. In the body, there's a script block containing two JavaScript statements: a prompt function to get a name and a confirm function to show a welcome message with the name. The code is color-coded: HTML tags are in blue, JavaScript keywords in blue, and strings in red.

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="utf-8" />
5     <title></title>
6 </head>
7 <body>
8
9     <script type="text/javascript">
10
11         var navn = prompt("Dit Navn", "Indtast dit navn her");
12
13         confirm("Velkommen til mit website, " + navn + "!");
14
15     </script>
16
17 </body>
18 </html>
```

Prøv at indtaste ovenstående eksempel i din editor, og test resultatet i din browser.

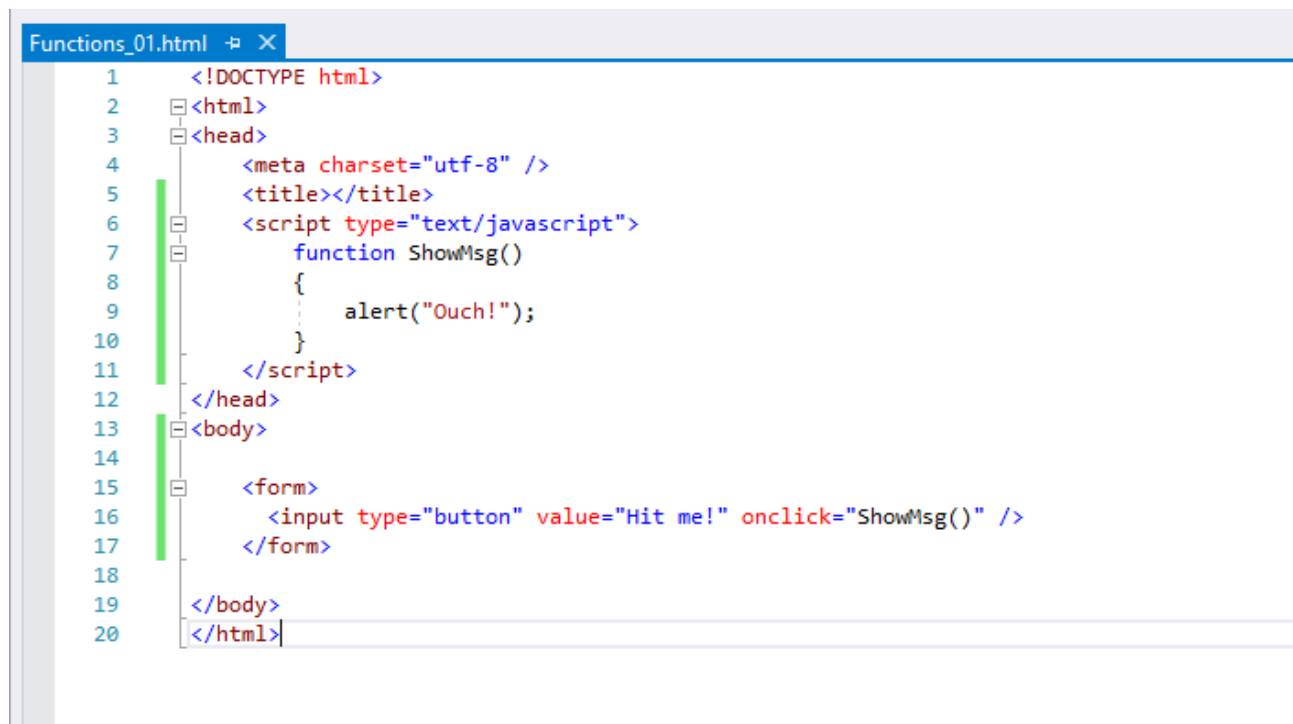
I linie 11 får vi en værdi fra brugeren for variabelen 'navn'.

I linie 13 kalder vi funktionen confirm med den tekst som vi gerne vil have vist for brugeren.

Funktioner

En funktion kan indeholde forskellige kodelumper som kan blive afviklet ved kald af funktionen.

Herunder er et simpelt eksempel på kald af en funktion fra en knap.



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8" />
5   <title></title>
6   <script type="text/javascript">
7     function ShowMsg()
8     {
9       alert("Ouch!");
10    }
11  </script>
12 </head>
13 <body>
14   <form>
15     <input type="button" value="Hit me!" onclick="ShowMsg()" />
16   </form>
17
18
19 </body>
20 </html>
```

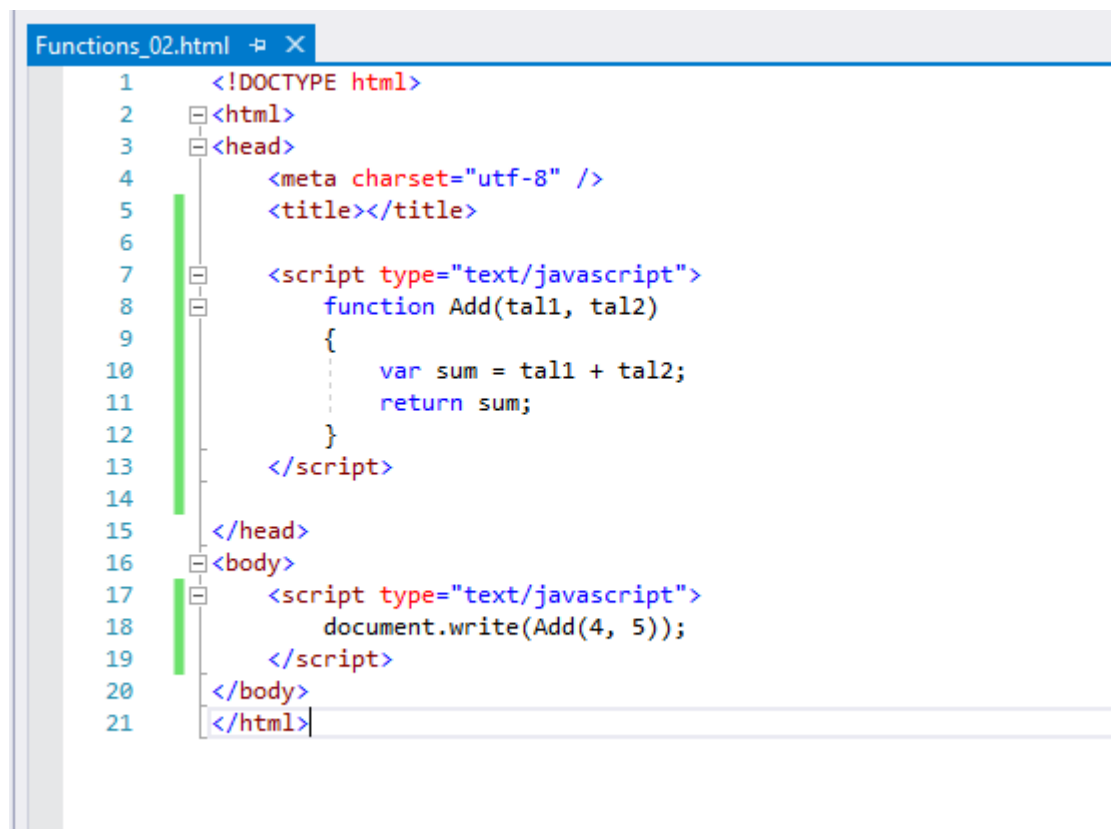
Byg eksemplet, og test det i din browser.

I linie 7, i HEAD tag'et, opretter vi en ny funktion med navnet ShowMsg.

I linie 9 inde i funktionens krop laver vi en alert box som skriver teksten "Ouch!".

I linie 16 laver vi en knap som i dens OnClick attribut kalder funktionen ShowMsg() som så vil blive afviklet.

En funktion kan også modtage værdier og behandle dem inde i sin krop og derefter sende resultatet tilbage til der hvor funktionen blev kaldt. Se eksemplet herunder.



```
1      <!DOCTYPE html>
2      <html>
3      <head>
4          <meta charset="utf-8" />
5          <title></title>
6
7          <script type="text/javascript">
8              function Add(tal1, tal2)
9              {
10                  var sum = tal1 + tal2;
11                  return sum;
12              }
13          </script>
14
15      </head>
16      <body>
17          <script type="text/javascript">
18              document.write(Add(4, 5));
19          </script>
20      </body>
21      </html>
```

I linie 8 opretter vi en funktion med navnet Add. Funktionen kan modtage to ting, variabelen tal1 og variabelen tal2.

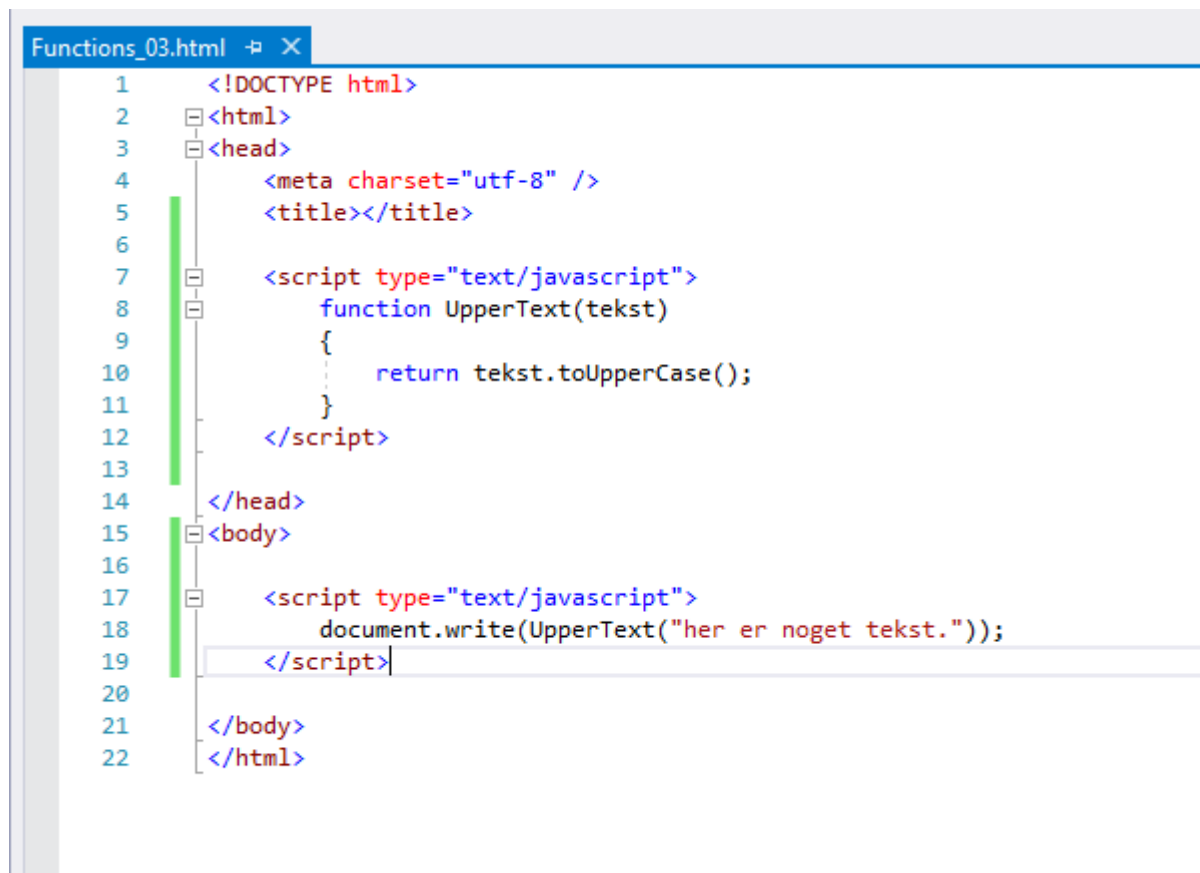
I linie 10 laver vi en ny variabel med navnet sum og lægger tal1 og tal2 sammen i variabelen.

I linie 11 returnerer vi resultatet i variabelen sum.

Funktionen 'Add' udnyttes ved et kald i BODY tag'et – her modtager funktionen document.write selv en funktion som *parameter*.

Byg eksemplet, og test det i din browser.

Herunder er vist et eksempel på brug af tekst i en funktion. Byg eksemplet, og test det i din browser.



```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6
7      <script type="text/javascript">
8          function UpperText(tekst)
9          {
10             return tekst.toUpperCase();
11          }
12      </script>
13
14  </head>
15  <body>
16
17      <script type="text/javascript">
18          document.write(UpperText("her er noget tekst."));
19      </script>
20
21  </body>
22  </html>
```

I linie 8 laver vi en ny funktion som vi kalder UpperText. Funktionen modtager en tekststreng via variablen tekst.

I linie 10 returnerer vi den modtagende tekst lavet til store bogstaver med funktionen toUpperCase().

I linie 18 kalder vi vores funktion - med en lille tekst af små bogstaver sendt ind som *parameter*.

Løkker - loops

En løkke bruges til at få gentaget taloperationer eller funktioner. For eksempel hvis jeg nu skal have udskrevet tallene fra 1 til 100 kan jeg bruge en løkke til det.

Overordnet findes der 3 typer løkker "DO", "FOR" og "WHILE". Under de 3 løkker findes der 3 strukturer: en som looper et fast antal gange, en hvor betingelsen er i starten og en hvor den er i slutningen. Når det gælder løkker skal man passe meget på. Hvis man ikke får dem skrevet rigtig kan man være så uheldig at få lavet en løkke der kører uendeligt og derved bruger al maskinens hukommelse og processorkraft.

For loop

Her er et eksempel på en for-løkke med sin betingelse i toppen.

I linie 9 laver vi en variabel med navnet i og tildeler den værdien 0.

I linie 11 laver vi så vores for løkke. Hvis vi ser på betingelsen mellem de to parenteser er den delt op i 3 dele.

- 1) ($i=0$): Vi sætter variabelen $i = 0$.
- 2) ($i \leq 5$): I det næste kriterium siger vi at løkken skal køre så længe at i er mindre eller lig med 5.
- 3) ($i++$): Til sidst lægger vi en til i hver gang løkken køres, det gør vi med $++$.

I linie 15 inde i løkkens krop udskrives vi en lille tekst omkredset af overskriftstypografi 1 til 5 alt efter hvilket tal, der står i variabelen i .

Når du har bygget eksemplet ovenfor, skulle du gerne kunne se følgende output i din browser:



While

En anden type løkke er While. Den fungerer næsten på samme måde, bortset fra at der kun er et enkelt kriterium i løkkens betingelse.

```
While_Loop_01.html  ▢ ✕
1      <!DOCTYPE html>
2      <html>
3      <head>
4          <meta charset="utf-8" />
5          <title></title>
6      </head>
7      <body>
8
9          <script type="text/javascript">
10             var i = 1;
11             while (i <= 5)
12             {
13                 document.write("<h" + i + ">Her er tekst nr. " + i + "</h" + i + ">");
14                 i = i + 1;
15             }
16         </script>
17
18     </body>
19 </html>
```

I linie 10 laver vi en variabel med navnet i og tildeler den værdien 1.

I linie 11 laver vi så vores while løkke. Hvis vi ser på betingelsen mellem de to parenteser er der kun en betingelse angivet her, løkken kører til i er mindre end eller lig med 5.

I linie 13 inde i løkkens krop udskriver vi igen en lille tekst omkredset af overskriftstypografi 1 til 5 alt efter hvilket tal der står i variabelen i.

I linie 14 lægger vi en til i hver gang løkken kører indtil vi rammer 5 og betingelsen stopper løkken.

For In

Den næste løkke vi skal se på er også en for løkke, her bruger vi blot en anden type betingelse, dette kaldes en for in løkke og kan bruges til at loope indhold ud af objekter. For eksempel en tabel eller et array.

Her under er der et lille eksempel på brug af en for in løkke sammen med et array.

```
For_In_Loop_01.html + X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6  </head>
7  <body>
8
9      <script type="text/javascript">
10         var t;
11         var sprog = new Array();
12         sprog[0] = "HTML";
13         sprog[1] = "CSS";
14         sprog[2] = "JavaScript";
15         sprog[3] = "Python";
16         sprog[4] = "C#";
17         sprog[5] = "PHP";
18         for (t in sprog)
19         {
20             document.write(sprog[t] + "<br />");
21         }
22     </script>
23
24 </body>
25 </html>
```

Byg eksemplet, og test det i din browser.

I linie 10 laver vi variabelen t.

I linie 11 laver vi et nyt array med navnet sprog.

I linie 12 til 17 fylder vi seks poster med sprognavne i vores array.

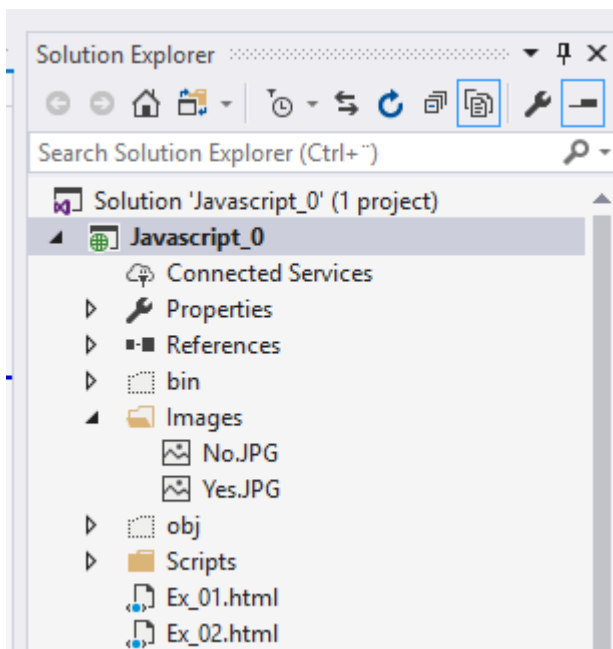
I linie 18 starter vi vores for in løkke som vil køre ligeså mange gange som der er poster i arrayet sprog.

I linie 20 udskriver vi så posterne en efter en.

Events og vinduer

Jeg sagde tidligere at JavaScript var lavet for at skabe interaktivitet på kedelige html-sider. Det har du ikke set meget til i eksemplerne ind til nu, men jeg lover dig at fra nu bliver det sjovere.

Det første vi skal se på nu er et eksempel på brugen af muse events. For at eksemplerne i det efterfølgende materiale vil virke, er det vigtigt at du nu laver en ny mappe i dit webprojekt ved navn 'Images', og at du trækker de billeder der ligger i jeres facebook gruppe i en zip fil 'Images' over i den folder:



Når du har etableret din 'Images' folder, er du klar til at bygge det næste eksempel:

```
Mouse_Events_01.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <meta charset="utf-8" />
5    <title></title>
6  </head>
7  <body onkeypress="alert('KeyPress event'); return false">
8
9    
11
12    <br />
13    <br />
14
15    <a href="Ex_01.html" onmouseover="style.color='black';" onmouseout="style.color='red';">StyleMouseOver</a>
16
17  </body>
18 </html>
```

I linie 7 laver vi en onKeyPress event på vores body-tag. Det vil sige, at hvis der klikkes på en knap på tastaturet vil det udløse en event, i dette tilfælde har vi lavet en alert box.

I linie 9 viser vi et billede med en onMouseOver event som viser et andet billede, når musen bevæges over det oprindelige billede.

I linie 15 laver vi en css effekt, hvor de samme to events bruges til at style farven på et link med.

Når du bygger eksemplet og tester det i din browser, så bemærk forskellen imellem hvornår koden bruger citationstegn ("") og 'enkeltgnyffer' (' '). Hvad sker der på siden, hvis du kommer til at bytte rundt på nogen af tegnene?

I det næste eksempel som vi skal se på prøver vi at få en tekst til at rulle i en div boks. Byg eksemplet i din editor, og test det i din browser.



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8" />
5   <title></title>
6 </head>
7 <body>
8   <div id="tekst" style="width: 650px; height:30px;"></div>
9   <script type="text/javascript">
10    var msg = "Her er en rullende tekst, som gerne skulle trille hen over din skærm.. indtil den kommer igen!";
11    var spacer = " ";
12    var pos = 0;
13    function scrollMessage()
14    {
15      document.getElementById("tekst").innerHTML =
16        msg.substring(pos, msg.length) + spacer + msg.substring(0, pos); pos++;
17      if (pos > msg.length) pos = 0;
18      window.setTimeout("scrollMessage()", 200);
19    }
20    scrollMessage();
21  </script>
22 </body>
23 </html>
```

I linie 10 laver vi en variabel, msg som vi tilskriver en lille tekst.

I linie 11 laver vi en variabel, spacer som vi fylder med mellemrum som skal bruges til at få afstand mellem ordene i teksten, når den starter forfra.

I linie 12 laver vi en variabel, pos som skal bruges til at holde styr på tekstens position når vi skal have den til at rulle hen over statuslinien.

I linie 13 til 18 opretter vi så en funktion som skal indeholde vores funktionalitet til at få teksten til at bevæge sig.

I linie 16 skriver vi vores tekst ud i "tekst" div boksen to gange adskilt af vores variabel spacer. Funktionen substring bruges til at få teksten til at bevæge sig ved at skifte tallet for positionen i variabelen pos.

I linie 17 tjekker vi om tallet i pos er større end tekstens længde, hvis det er tilfældet sætter vi pos lig med 0 og teksten vil starte forfra.

I linie 18 bliver vi ved med at kalde vores funktion hvert 200 millisekund.

I linie 20 aktiverer vi vores funktion, ScrollMessage().

I eksemplet herunder vil jeg vise en lille smule om hvordan man åbner og lukker vinduer i browseren via Javascript. Byg selv eksemplet for at se det køre.

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8" />
5 <title></title>
6 </head>
7 <body>
8
9 <form name="winform">
10 <input type="button" value="Åbn vindue"
11 <input type="button" value="Luk vindue" onclick="MyWin.close();" />
12 <br />
13 <input type="button" value="Luk vindue" onclick="MyWin.close();" />
14 <br />
15 </form>
16
17 </body>
18 </html>
```

I linie 10-11 laver vi en ny knap, det vigtige her er onClick attributten. Den er der vist en forklaring på herunder.

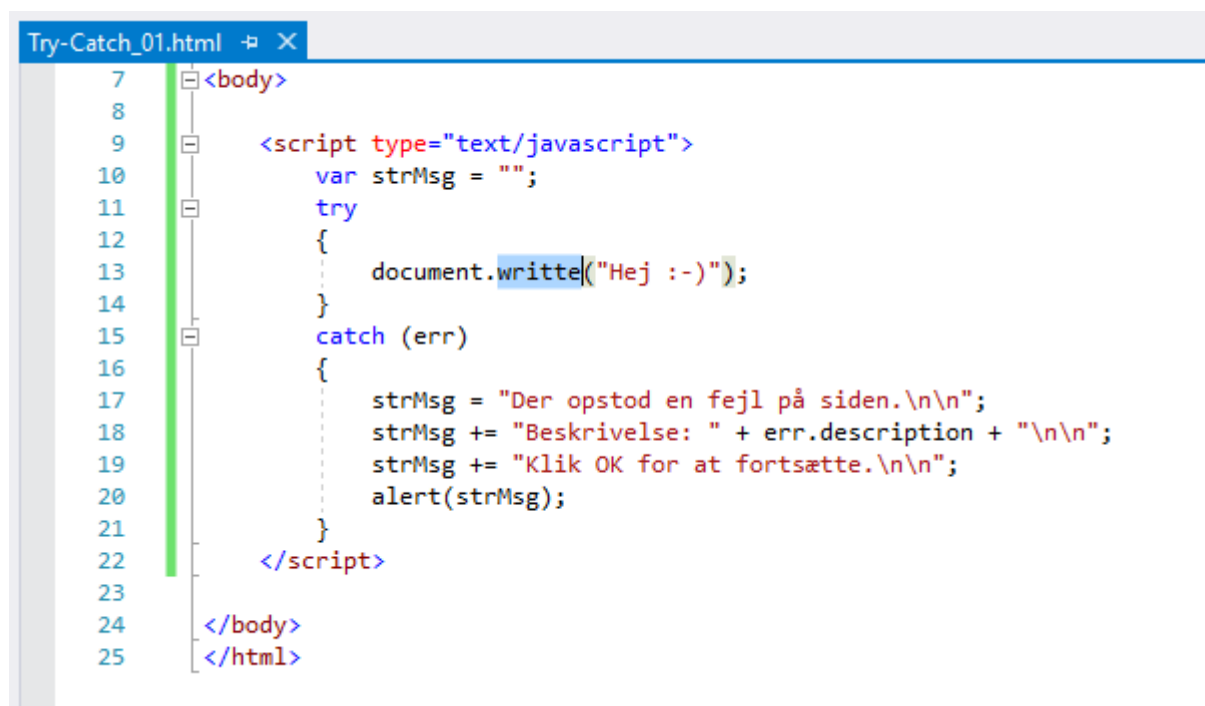
- MyWin: Er navnet på det vindue som vi laver.
- Window.Open: Er funktionen til at åbne vinduet med.
- ToolBar=no: Fjerner toolbars i browseren.
- Location=no: Fjerner adresselinien.
- Status=no: Fjerner status baren i bunden af vinduet.
- Width: Bredden på vinduet i px.
- Height: Højden på vinduet i px.

I linie 13 laver vi en knap som lukker det nye vindue når der klikkes på den.

Try – Catch konstruktionen

Når man koder kan der nemt opstå fejl. Det kan for eksempel være en regnefejl eller en tastefejl som vist i eksemplet her under, hvor den uheldige programmør er kommet til at skrive et t for meget i 'write' - eller en bruger som gør noget forkert så der opstår en fejl.

Disse fejl kan man nemt fange i sit script og behandle dem så de enten ikke kommer ud til brugeren eller lave sine egne fejlmeddelelser. I eksemplet herunder er fejlen pakket ind i en såkaldt Try-Catch struktur som gør at man kan fange fejlen inden den når brugeren.

A screenshot of a code editor window titled 'Try-Catch_01.html'. The editor shows a JavaScript code snippet within an HTML body. The code uses a try-catch block to handle a potential error in the document.write function. Line 13 contains the code document.write("Hej :-");. The catch block (lines 16-21) constructs a message strMsg with the error description and displays it using alert(strMsg).

```
7 <body>
8
9 <script type="text/javascript">
10   var strMsg = "";
11   try
12   {
13     document.write("Hej :-");
14   }
15   catch (err)
16   {
17     strMsg = "Der opstod en fejl på siden.\n\n";
18     strMsg += "Beskrivelse: " + err.description + "\n\n";
19     strMsg += "Klik OK for at fortsætte.\n\n";
20     alert(strMsg);
21   }
22 </script>
23
24 </body>
25 </html>
```

I linie 10 opretter vi en variabel med navnet strMsg. Denne skal bruges til at opbevare vores fejlmeddelelse.

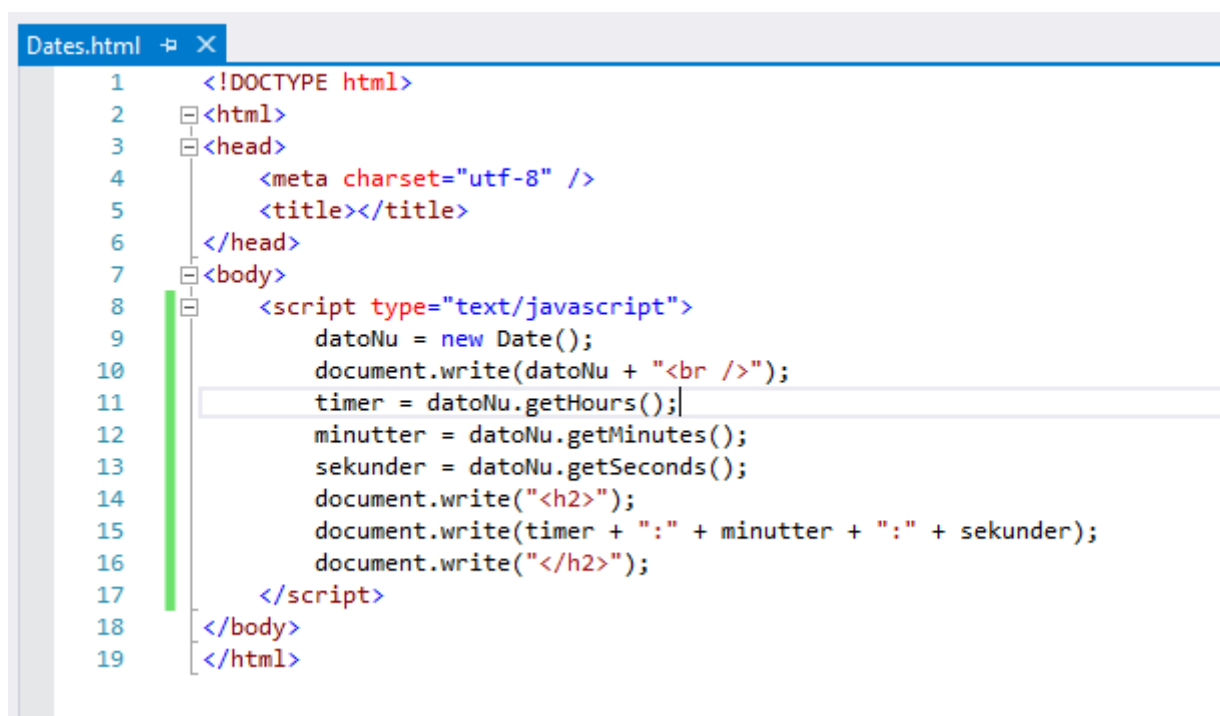
I linie 11 starter vi så vores try, og alt hvad der er mellem tuborgklammerne / de krøllede parenteser efter try, vil blive opfanget ved fejl.

I linie 15 fanger vi så vores fejl i variablen err.

I linie 17 til 20 udskriver vi så en beskrivelse af hvilken fejl der opstod.

Datoer

JavaScript har også et dato objekt som bruges til at manipulere med datoer. Herunder er vist et eksempel på nogle af de mest brugte funktioner når man arbejder med datoer.



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8" />
5 <title></title>
6 </head>
7 <body>
8 <script type="text/javascript">
9     datoNu = new Date();
10    document.write(datoNu + "<br />");
11    timer = datoNu.getHours();
12    minutter = datoNu.getMinutes();
13    sekunder = datoNu.getSeconds();
14    document.write("<h2>");
15    document.write(timer + ":" + minutter + ":" + sekunder);
16    document.write("</h2>");
17 </script>
18 </body>
19 </html>
```

I linie 9 opretter vi en variabel ved navnet datoNu. I den lægger vi den aktuelle dato og klokkeslæt.

I linie 10 udskriver vi indholdet af datoNu efterfulgt af et linesskift.

I linie 11 opretter vi en variabel med navnet timer, og trækker timerne ud af datoNu via funktionen getHours().

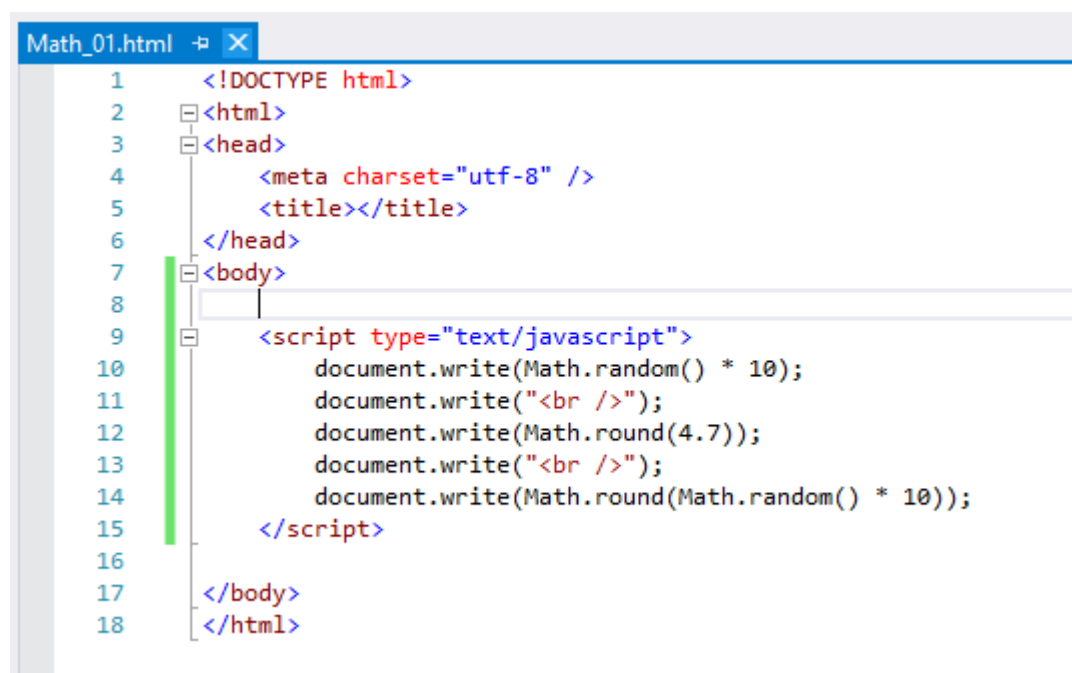
I linie 12 opretter vi en variabel med navnet minutter, og trækker minutterne ud af datoNu via funktionen getMinutes().

I linie 13 opretter vi en variable med navnet sekunder, og trækker sekunderne ud af datoNu via funktionen getSeconds().

I linie 15 udskriver vi så de tre variabler adskilt af kolon.

Math

Der er også indbygget et regneobjekt i JavaScript. Herunder er vist et lille eksempel på to af de metoder der er i objektet og et eksempel på at kombinere dem.



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8" />
5   <title></title>
6 </head>
7 <body>
8
9   <script type="text/javascript">
10     document.write(Math.random() * 10);
11     document.write("<br />");
12     document.write(Math.round(4.7));
13     document.write("<br />");
14     document.write(Math.round(Math.random() * 10));
15   </script>
16
17 </body>
18 </html>
```

I linie 10 bruger vi Math.Random til at få genereret et tal mellem 0 og 10. Denne funktion genererer et tilfældigt kommatotal indenfor rammebetingelsen (et sted imellem 0 og 10).

I linie 12 bruger vi funktionen Math.Round til at afrunde 4.7. Resultatet bliver 5.

I linie 14 kombinerer vi så de to funktioner så Math.Ramdom returnerer et komma tal og funktionen Math.Round afrunder det til et helt tal.

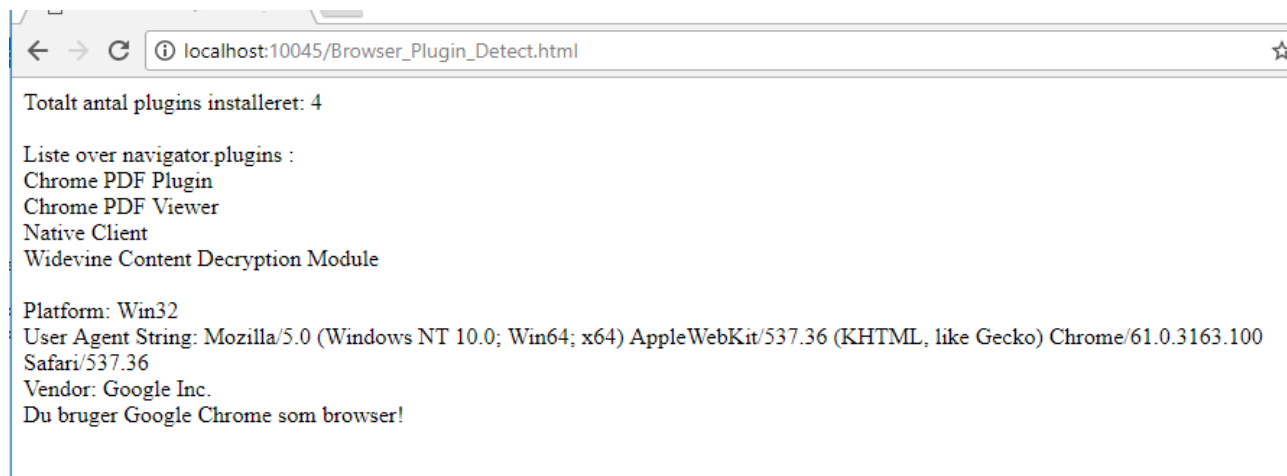
I det næste eksempel skal vi se på den information, man med javascript kan trække ud af browserens navigator objekt – denne info kan eksempelvis bruges til at lave statistikker over, hvilke browsere der besøger ens webside, hvilken platform (OS) brugeren afvikler sin browser på, hvilke browser plugins brugeren har installeret etc.

Hvis du har set en besøgsstatistik for en webside, fx i admin delen for en hjemmeside hosted hos Surftown.dk eller One.com, er data heri ofte trukket fra brugerens User Agent String, som er en tekststreng der fortæller serveren, hvilken type klient det er der laver http forespørgslen. Det er dog ret nemt at 'spoofe' en user agent string, så man kan ikke nødvendigvis regne med at det altid er 'sandt' hvad der står i denne.

```
Browser_Plugin_Detect.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6  </head>
7  <body>
8
9      <div id="plugins"></div>
10
11     <script type="text/javascript">
12         document.write("<br />");
13         document.write("Platform: " + navigator.platform);
14         document.write("<br />");
15         document.write("User Agent String: " + navigator.userAgent);
16         document.write("<br />");
17         document.write("Vendor: " + navigator.vendor);
18         document.write("<br />");
19
20         if (navigator.userAgent.includes("Chrome", 0) && navigator.vendor.includes("Google", 0)) {
21             document.write("Du bruger Google Chrome som browser!");
22         }
23
24         else if (navigator.userAgent.includes("Firefox", 0)) {
25             document.write("Du bruger Firefox som browser!");
26         }
27
28         else if (navigator.userAgent.includes("Edge", 0)) {
29             document.write("Du bruger MS Edge som browser!");
30         }
31     </script>
```

```
Browser_Plugin_Detect.html  X
31
32     var x = navigator.plugins.length;
33     var txt = "Totalt antal plugins installeret: " + x + "<br/>" + "<br/>";
34     txt += "Liste over navigator.plugins : " + "<br/>";
35     for (var i = 0; i < x; i++) {
36         txt += navigator.plugins[i].name + "<br/>";
37     }
38     document.getElementById("plugins").innerHTML = txt;
39 </script>
40
41 </body>
42 </html>
```

Byg eksemplet i din editor, og test resultatet i din browser. Du skulle gerne kunne se et output fra koden, der ligner nedenstående:



Koden her er lidt mere kompleks end i de foregående eksempler, men nu er du også ved at være vant til at skrive javascript!

I linie 20-22 stiller vi en dobbelt betingelse op med brug af operatoren '&&' der betyder AND:

```
if (navigator.userAgent.includes("Chrome", 0) && navigator.vendor.includes("Google", 0)) {  
    document.write("Du bruger Google Chrome som browser!");  
}
```

Dvs. begge betingelser skal være opfyldt, for at "Du bruger Google Chrome som browser!" vil blive skrevet ud til skærmen. De to næste checks stiller kun en enkelt betingelse, idet 'vendor' string property'en er tom i henholdsvis Firefox og Edge browseren.

Challenge: hvis du tester websiden her i Internet Explorer (populært kaldet Internet Exploder blandt *geeks*, gæt selv hvorfor) vil scriptet højst sandsynligt ikke blive afviklet korrekt. Kan du med egne ord forklare hvorfor det sker?

I linie 32-38 udnytter vi, at navigator.plugins property'en vil returnere et array, som vi kan hente længden af (antal elementer i) ved at skrive:

```
var x = navigator.plugins.length;
```

Dette tal, x, bruger vi til at bestemme hvor mange gange for loopet skal køre:

```
for (var i = 0; i < x; i++) {  
    txt += navigator.plugins[i].name + "<br/>";  
}
```

+= tegnet efter txt vil lægge det aktuelle elements navn ind oveni det, der allerede måtte stå i variabelen txt.

I den sidste linie kode, på linie 38 benytter vi javascripts .getElementById() metode til at få fat i den tomme div 'plugins', som vi deklarerede i starten af dokumentet, og fylder dens innerHtml property med den info, vi har samlet op i txt variabelen omkring de installerede browser plugins:

```
document.getElementById("plugins").innerHTML = txt;
```


I næste eksempel skal vi se lidt på hvordan vi kan aflæse flere egenskaber om hvor vores fil befinder sig på nettet og i filsystemet.



```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6  </head>
7  <body>
8      <script type="text/javascript">
9          document.write("Host: " + window.location.host + "<br />");
10         document.write("Hostname: " + window.location.hostname + "<br />");
11         document.write("Href: " + window.location.href + "<br />");
12         document.write("Path: " + window.location.pathname + "<br />");
13     </script>
14 </body>
15 </html>
```

Byg eksemplet, og test det i din browser.

Herunder er der en beskrivelse af hvad de forskellige funktioner returnerer:

- `window.location.host`: Returnerer adressen på webstedet.
- `window.location.hostname`: Returnerer domænenavn.
- `window.location.href`: Returnerer linket til filen.
- `window.location.pathname`: Returnerer den fysiske sti til filen.

Herunder er vist et lille eksempel på brug af objektet screen. Med screen er det muligt at trække en lang række informationer ud om det browservindue som siden bliver vist i. I dette eksempel vil vi blot prøve at læse højden og bredden på vinduet.

```
Screen_01.html  [X]
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6  </head>
7  <body>
8
9      <script type="text/javascript">
10         W = screen.availWidth;
11         H = screen.availHeight;
12         document.write("Screen width: " + W + " pixels.<br/>");
13         document.write("Screen height: " + H + " pixels.<br/>");
14     </script>
15
16 </body>
17 </html>
```

I linie 10 indlæser vi bredden på vinduet i variabelen W.

I linie 11 indlæser vi højden på vinduet i variabelen H.

I linie 12 og 13 udskriver vi så indholdet af de to variabler.

Formvalidering

Det næste vi skal se på er formvalidering med JavaScript. Det er et af de punkter hvor JavaScript virkeligt er nyttigt.

I eksemplet herunder vil vi prøve at lave en form hvor alle felter skal være udfyldt for at formen kan sendes.

```
ValidateForm_01.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6      <script type="text/javascript">
7          function ValidateForm() {
8              error = 0;
9              if ((document.forms[0].ForNavn.value == '') && (error == 0))
10             {
11                 alert('Du skal udfylde feltet med dit Fornavn!');
12                 document.forms[0].ForNavn.focus();
13                 error = 1;
14             }
15
16             if ((document.forms[0].EfterNavn.value == '') && (error == 0))
17             {
18                 alert('Du skal udfylde feltet med dit Efternavn!');
19                 document.forms[0].EfterNavn.focus();
20                 error = 1;
21             }
22
23             if (error == 0) {
24                 document.forms[0].submit();
25             }
26         }
27     </script>
28 </head>
```

.. og i sidens body tag bygger vi selve formen op:

```
ValidateForm_01.html  X
26  <body>
27
28  <form action="ValidateForm_01.html" method="post" onsubmit="ValidateForm()">
29      Fornavn: <input type="text" name="ForNavn" size="30" />
30      <br />
31      Efternavn:
32      <input type="text" name="EfterNavn" size="30" />
33      <br />
34      <input type="submit" value="Send" />
35  </form>
36
37  </body>
38  </html>
```

I linie 7 starter vi en ny funktion med navnet ValidateForm.

I linie 8 laver vi en variabel med navnet error som skal bruges til at holde styr på om et felt ikke er udfyldt.

I linie 9 tjekker vi så om der er indtastet noget i feltet ForNavn og om error er lig med 0.

Hvis error ikke er lig med 0 har der været et felt som ikke er udfyldt. Hvis der ikke står noget i feltet Fornavn og hvis error indeholder 0 sker der 3 ting.

I linie 11 laves en alert box som gør brugeren opmærksom på at der skal skrives noget i feltet Fornavn.

I linie 12 sættes der fokus på feltet ForNavn.

I linie 13 sættes variabelen error lig med 1.

I linie 16 til 21 tjekkes feltet EfterNavn på nøjagtig samme måde.

I linie 23 tjekker vi om error er lig med 0. Hvis den er det bliver formen sendt (submitted) i linie 24.

Formvalidering: CPR Check

Det næste eksempel er lidt mere avanceret – her skal vi checke om et indtastet CPR nummer overholder de danske regler for opbygning af personnumre. Start med at oprette en ny html fil i dit project, 'CPR_Check.html', og indtast følgende javascript i head tag'et:

```
CPR_Check.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>CPR Check</title>
6
7      <script type="text/javascript">
8
9          function Trim(txt) {
10              for (var i = 0, j = txt.length - 1; i <= j;) {
11                  if (txt.charAt(i) == ' ' && txt.charAt(j) == ' ') {
12                      ++i;
13                      --j;
14                  }
15                  else if (txt.charAt(i) == ' ')
16                      ++i;
17                  else if (txt.charAt(j) == ' ')
18                      --j;
19                  else
20                      return txt.substring(i, j + 1);
21              }
22              return "";
23          }
24      </script>
```

.. og samme fil, fortsat:

```
CPR_Check.html  X
25
26      function CprnrOk(cpr) {
27          if (cpr.length == 10) {
28              var sum = parseInt(cpr.charAt(9));
29              for (var i = 0, v = 4; i < 9; ++i) {
30                  sum += parseInt(cpr.charAt(i)) * (v--);
31                  if (v < 2) v = 7;
32              }
33              return sum % 11 == 0;
34          }
35          return false;
36      }
37
38      function Kontrol() {
39          alert(CprnrOk(Trim(document.form.cpr.value)) ?
40              "Cprnummer Ok" : "Ulovligt cprnummer");
41      }
42  </script>
43  </head>
44  <body>
45      <form name="form">
46          <H2>Indtast et cprnummer: <input type="password" name="cpr" size="10"></H2>
47          <p><input type="button" value=" Ok " OnClick="Kontrol()"></p>
48      </form>
49  </body>
50  </html>
```

Funktionen Trim() har en parameter, der er et String-objekt. Funktionen returnerer en streng, hvor alle indledende og afsluttende mellemrum er fjernet fra parameterobjektet.

Det centrale element i Trim funktionen er et for-loop i kodelinie 10, som dem du allerede kender til, blot med en udvidet brug af betingelserne du har brugt sidst:

```
for (var i = 0, j = txt.length - 1; i <= j;) {
```

For loopet starter med at sætte en variabel i lig 0, og lader loopet løbe ved at *inkrementere* værdien af i indtil enden af tekststrengen nås.

String-objektet har en række metoder. Bemærk, hvordan substring() anvendes til at returnere en delstreng.

Funktionen CprnrOk() har en parameter, der også er et String-objekt. Den tester, om det repræsenterer et lovligt cprnummer.

parseInt() er en indbygget funktion i JavaScript, der konverterer en streng til et heltal. Bemærk også charAt(), som er en metode på String-objektet, der returnerer det i'te tegn.

HTML-dokumentet definerer en form med et indtastningsfelt og en knap. Knappen har en event-handler Kontrol(), der udføres, når man klikker på knappen.

Bemærk, at funktionen Kontrol() anvender ?-operatoren. Denne kaldes også den ternære operator – du kan læse en udførlig forklaring på URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Conditional_Operator

I vores tilfælde (se kodelinien herunder) tester vi, om udtrykket på venstre side af ? tegnet returnerer true eller false, ved først at kalde Trim funktionen med det indtastede nummer som parameter, for at fjerne evt. mellemrum. Herefter kaldes CprnrOk funktionen med den trimmede streng som parameter, og CprnrOk vil så returnere true hvis cpr nummeret er lovligt, og false hvis cpr nummeret er ulovligt.

```
(CprnrOk(Trim(document.form.cpr.value)) ? "Cprnummer Ok" : "Ulovligt cprnummer");
```

Hvis venstre side af ligningen returnerer true, vil operanden ? returnere den første *expression*, ("Cprnummer Ok") men hvis resultatet er false, returneres den anden *expression* ("Ulovligt cprnummer").

Det kan være en rigtig god programmør-øvelse at læse sig til hvordan ting fungerer ved selv at gå ud og finde de steder i online dokumentationen, hvor anvendelsen af en specifik funktion er beskrevet. Nedenstående screendump er hentet fra developer.mozilla.org's javascript referencesider:

Syntax

```
condition ? expr1 : expr2
```

Parameters

`condition` (or `conditions`)

An expression that evaluates to `true` or `false`.

`expr1`, `expr2`

Expressions with values of any type.

Description

If `condition` is `true`, the operator returns the value of `expr1`; otherwise, it returns the value of `expr2`.

Du vil sikkert give mig ret i, at det er mindre ordrigt, og måske også mere præcist beskrevet end den danske version ovenfor – hvis man har det fint med at skulle tænke lidt mere matematisk.

Når siden vises, får man følgende vindue:



Når du har bygget eksemplet, kan du prøve at teste en indtastning i din browser – prøv med en tilfældig talrække på 7 cifre, og med dit eget CPR nr. Hvad får du at vide i alert boksen?

Det næste er et lille sjovt eksempel hvor vi skal prøve at få to billeder og en tekst til at flytte sig på skærmen når musen føres hen over dem. Byg eksemplet, og test det i din browser.

```
Moving_Objects_01.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Moving Objects</title>
6      <style type="text/css">
7          .move {
8              position: relative;
9              cursor: move
10         }
11     </style>
12
13     <script type="text/javascript">
14         function Move(obj) {
15             if (obj.className == "move") {
16                 var ran1 = Math.random() * 400;
17                 var ran2 = Math.random() * 300;
18                 obj.style.left = ran1 + "px";
19                 obj.style.top = ran2 + "px";
20             }
21         }
22     </script>
23
24 </head>
```

.. og i Body Tag'et:


```
Moving_Objects_01.html -p X
25 <body>
26
27 <div onmouseover="Move(this);" class="move">
28     Hej med dig :-)
29 </div>
30 
31 <br />
32 
33 <br />
34
35 </body>
36 </html>
```

I linie 6 laver vi en CSS-klasse som vi kalder move, den skal bruges til at style og identificere vores billeder.

I linie 14 laver vi funktionen Move som skal bruges til at flytte vores billeder et tilfældigt sted hen når den bliver aktiveret.

I linie 15 tjekker vi om det element som har aktiveret vores funktion har en css klasse ved navn move. Hvis den har det, så udføres resten af kodeblokken, linie 16 til 19.

I linie 16 og 17 genererer vi to tilfældige tal.

I linie 18 og 19 flytter vi så elementet til den nye placering ved hjælp af de genererede tal.

I det næste eksempel skal vi se på hvordan vi på en nem måde kan få elementer til at bevæge sig hen over skærmen.

```

Moving_Objects_02.html
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6      <script type="text/javascript">
7
8          var posBlaa = 50;
9          var posRoed = 50;
10         function next() {
11             document.getElementById("roed").style.left = (posRoed += 5) + "px";
12             document.getElementById("blaa").style.left = (posBlaa += 10) + "px";
13             window.setTimeout("next();", 100);
14         }
15     </script>
16 </head>
17 <body onload="next();">
18     <div id="roed" style="position:absolute; left:0px; top:100px; width:100px; height:100px">
19         
20     </div>
21     <div id="blaa" style="position:absolute; left:0px; top:300px; width:100px; height:100px">
22         
23     </div>
24 </body>
25 </html>

```

I linie 8 og 9 laver vi to variabler som skal bruges til at holde styr på positionerne af vores billeder.

I linie 10 starter vi så funktionen next som skal flytte vores billeder hen over skærmen.

I linie 11 finder vi vores element roed og lægger 5 px til positionen fra venstre.

I linie 12 finder vi også vores element blaa og lægger 10 px til positionen fra venstre.

I linie 13 får funktionen sat en timeout på 100 ms, inden den kaldes næste gang.

Du har sikkert tidligere på nettet set små billedgallerier som dette her til højre, hvor man kan føre musen over et lille billede hvorefter det så vil blive vist i stort format nedenunder. Sådan et eksempel skal vi til at lave.



```
Image_Gallery_01.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Image Gallery</title>
6
7      <script type="text/javascript">
8          function SkiftBillede(img) {
9              document.images.StortBillede.src = img;
10          }
11      </script>
12  </head>
13  <body>
14      <div align="center">
15          
16          <br /><br />
17          <a onmouseover='SkiftBillede("Images/Red.jpg");'>
18              
19          </a>
20          <a onmouseover='SkiftBillede("Images/Blue.jpg");'>
21              
22          </a>
23          <a onmouseover='SkiftBillede("Images/No.jpg");'>
24              
25          </a>
26          <a onmouseover='SkiftBillede("Images/Yes.jpg");'>
27              
28          </a>
29      </div>
30  </body>
31  </html>
```

I linie 8 laver vi en funktion som modtager et billednavn i variablen img. I linie 9 skifter vi så billede filen på img-tagget med navnet StortBillede.

I linie 15 laver vi så img-tagget StortBillede.

I linie 17 til 28 laver vi så 4 billedlinks som kalder funktionen SkiftBillede med hver deres filnavn som parameter. Byg eksemplet, og test billedgalleriet i din browser.

Det næste eksempel er blot for at vise hvordan du med de metoder du har lært kan lave en simpel bannerrotation. Byg eksemplet, og test din banner-rotations side i din browser.

```
Banner_Rotation_01.html  + X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Rotating Banner Images</title>
6      <script type="text/javascript">
7          var Img = new Array();
8          Img[0] = new Image();
9          Img[0].src = "Images/Guitar_1.jpg";
10         Img[1] = new Image();
11         Img[1].src = "Images/Piano_1.jpg";
12         Img[2] = new Image();
13         Img[2].src = "Images/Piano_2.jpg";
14         Img[3] = new Image();
15         Img[3].src = "Images/Guitar_2.jpg";
16         Img[4] = new Image();
17         Img[4].src = "Images/Microphone_1.jpg";
18         function SkiftBillede()
19         {
20             rnd = Math.round(Math.random() * 4);
21             document.image1.src = Img[rnd].src;
22             setTimeout("SkiftBillede()", 1000);
23         }
24     </script>
25 </head>
26 <body onload="javascript:SkiftBillede();">
27     
28 </body>
29 </html>
```

Timing

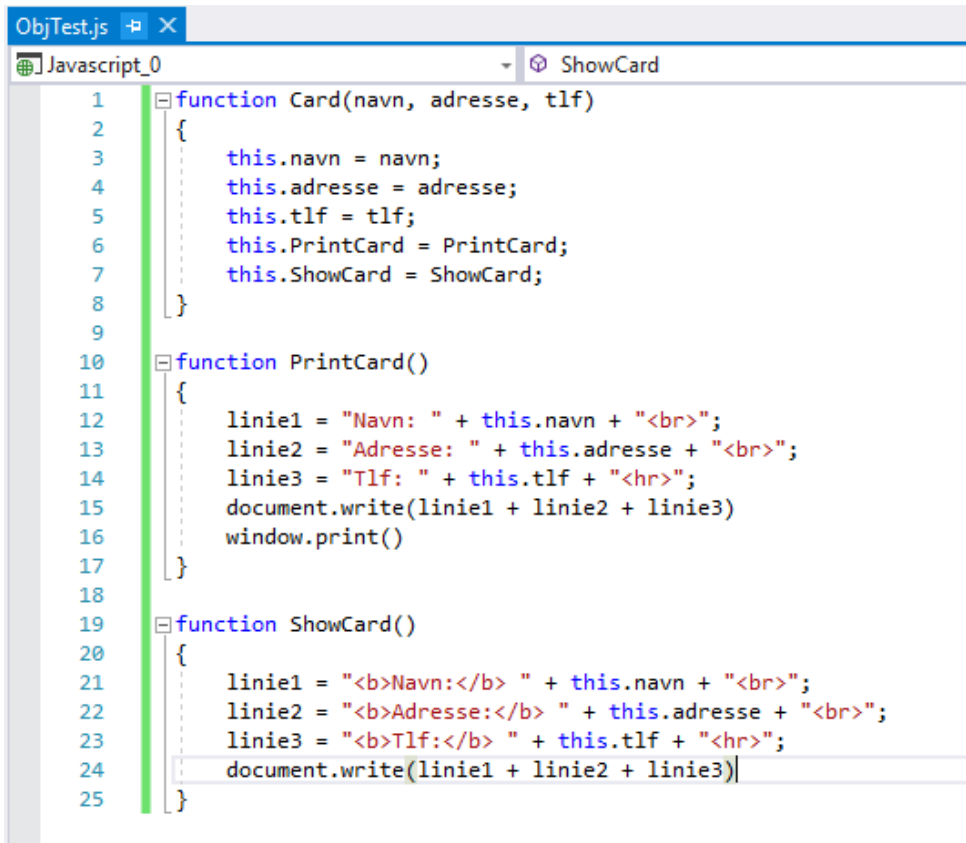
Timing har du efterhånden arbejdet en del med men herunder er alligevel et simpelt eksempel på brug af timing. Byg eksemplet, og test output fra funktionen i din browser – husk, det tager lige fem sekunder inden der sker noget!

```
Timer_Simple_01.html  +  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Simple Timer</title>
6
7      <script type="text/javascript">
8          function Advarsel(tid)
9          {
10             var strMsg = setTimeout("alert('Forsinket advarsel på " + tid / 1000 + " sek.!')", tid);
11          }
12      </script>
13
14  </head>
15  <body onload="Advarsel(5000);">
16
17  </body>
18  </html>
```

Creating objects

Jeg har tidligere fortalt at JavaScript er et scripting sprog men det er dog muligt at kode noget som lugter lidt af objektorienteret programmering. En af de største fordele ved at programmere objektorienteret er genbrug af kode. Det er muligt at lægge sine script i eksterne filer/objekter hvorefter det er muligt at bruge dem igen og igen på sit site.

I denne øvelse skal du fremstille en tekstfil med endelsen .js, der skal placeres i din 'Scripts' folder. Kald filen for ObjTest.js:

A screenshot of a code editor window titled 'ObjTest.js'. The editor shows JavaScript code for a 'Card' object. The code is as follows:

```
1 function Card(navn, adresse, tlf)
2 {
3     this.navn = navn;
4     this.adresse = adresse;
5     this.tlf = tlf;
6     this.PrintCard = PrintCard;
7     this.ShowCard = ShowCard;
8 }
9
10 function PrintCard()
11 {
12     linie1 = "Navn: " + this.navn + "<br>";
13     linie2 = "Adresse: " + this.adresse + "<br>";
14     linie3 = "Tlf: " + this.tlf + "<hr>";
15     document.write(linie1 + linie2 + linie3)
16     window.print()
17 }
18
19 function ShowCard()
20 {
21     linie1 = "<b>Navn:</b> " + this.navn + "<br>";
22     linie2 = "<b>Adresse:</b> " + this.adresse + "<br>";
23     linie3 = "<b>Tlf:</b> " + this.tlf + "<hr>";
24     document.write(linie1 + linie2 + linie3)
25 }
```

I linie 1 laver vi en property funktion som modtager 3 variabler.

I linie 3 laver vi en property som vi kalder navn og fylder det modtagne navn der i.

Det samme gør vi med adresse og tlf i linie 4 og 5.

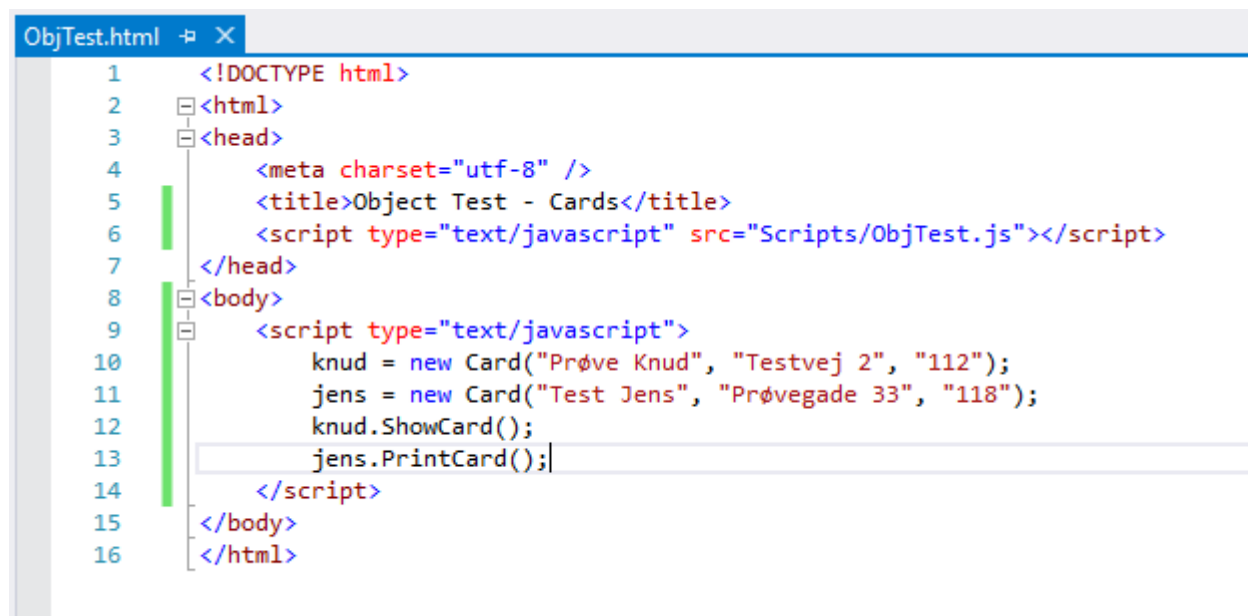
I linie 6 og 7 tilføjer vi også navnene på de to funktioner som vi nu skal kode - og derved skaber vi en reference mellem vores property og vores funktioner.

I linie 10 til 17 laver vi en funktion som udskriver vores indtastede oplysninger og åbner printerdialogen.

I linie 19 til 25 laver vi endnu en funktion som udskriver vores indtastede oplysninger med fede overskrifter.

Nu skal vi så i gang med at bruge selve objektet.

Opret et nyt HTML dokument, ObjTest.html:



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8" />
5 <title>Object Test - Cards</title>
6 <script type="text/javascript" src="Scripts/ObjTest.js"></script>
7 </head>
8 <body>
9 <script type="text/javascript">
10 knud = new Card("Prøve Knud", "Testvej 2", "112");
11 jens = new Card("Test Jens", "Prøvegade 33", "118");
12 knud.ShowCard();
13 jens.PrintCard();
14 </script>
15 </body>
16 </html>
```

I linie 6 importerer vi vores javascript-kode.

I linie 10 laver vi en property som vi kalder knud og fylder knuds kontaktoplysninger i funktionen card.

I linie 11 laver vi endnu en property som vi kalder jens og fylder jens' kontaktoplysninger i funktionen card.

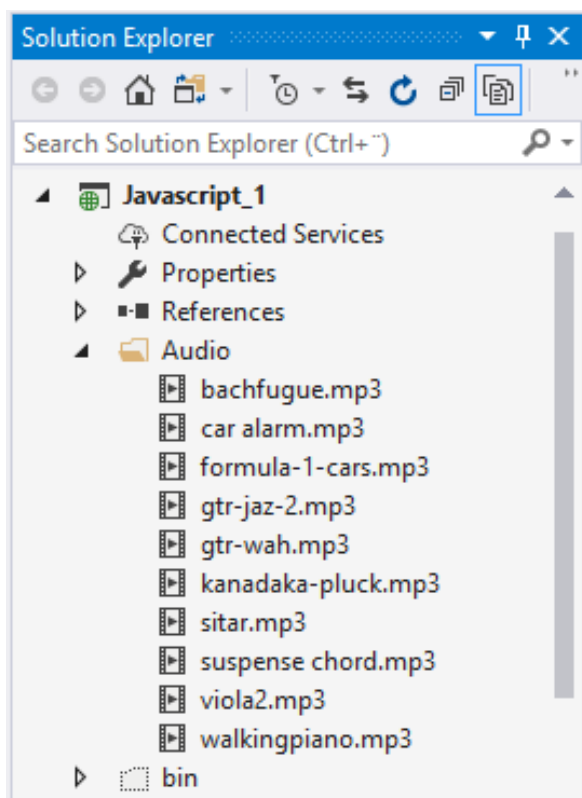
I linie 12 kalder vi funktionen ShowCard på vores property knud som udskriver hans data på siden med fede overskrifter.

I linie 13 kalder funktionen PrintCard på vores property jens, som udskriver hans data på siden og åbner printerdialogen.

Afspilning af medier

Nu skal vi se på nogle af de muligheder vi har for at få afspillet lyd og video med JavaScript. Til denne opgave skal du bruge filerne, du finder i Audio mappen under materialer.zip (se fb gruppe).

Lav en 'Audio' mappe i dit projekt, og træk filerne fra Audio mappen over i denne:



Det første eksempel vi skal se på er hvordan vi kan få afspillet, pauset og stoppet en mp3-fil ved at klikke på et link.


```
JS_2.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Fantastic Music Player V.1.0</title>
6  </head>
7  <body>
8
9      <script type="text/javascript">
10         var audio = new Audio('/Audio/walkingpiano.mp3');
11
12         function playSound() {
13             audio.play();
14         }
15         function pauseSound() {
16             audio.pause();
17         }
18         function stopSound() {
19             audio.pause();
20             audio.currentTime = 0;
21         }
22         //function setPlaybackRate() {
23         //    audio.playbackRate(2);
24         //}
25     </script>
26
27     <a href="#" onclick="playSound();">Play</a><br />
28     <a href="#" onclick="pauseSound();">Pause</a><br />
29     <a href="#" onclick="stopSound();">Stop</a><br />
30     <!--<a href="#" onclick="setPlaybackRate(2);">Double Playback Speed</a><br />-->
31
32 </body>
33 </html>
```

I linie 10 laver vi et audio objekt, der kan bruges til at afspille forskellige lydfiler. I linie 12-21 laver vi 3 funktioner, playSound, pauseSound og stopSound. Funktionaliteten er meget enkel i de tre funktioner. De starter, stopper eller pauser blot vores audio objekt, når de bliver kaldt.

I linie 27-30 laver vi til slut 3 links som kalder de 3 funktioner når der klikkes på dem.

Nu har vi set lidt på hvordan vi kan få afspillet lyd på vores sider. Det sidste eksempel jeg vil vise i denne lektion er, hvordan vi nemt kan få afspillet video på en html-side. Opret en mappe, 'Video' i dit projekt og

overfør filerne fra 'Video' mappen i materialer.zip til denne. Opret derefter en ny html fil, og indtast nedenstående kode i den:

```
JS_3_Video.html  + X
1  <!DOCTYPE html>
2  <html>
3  <body>
4
5      <div style="text-align:center">
6          <button onclick="playPause()">Play/Pause</button>
7          <button onclick="makeBig()">Big</button>
8          <button onclick="makeSmall()">Small</button>
9          <button onclick="makeNormal()">Normal</button>
10         <br><br>
11         <video id="video1" width="420">
12             <source src="/Video/Rock_Steady.mp4" type="video/mp4">
13             Your browser does not support HTML5 video.
14         </video>
15     </div>
16
```

.. og samme fil, fortsat:

```
JS_3_Video.html  + X
16
17     <script type="text/javascript">
18         var myVideo = document.getElementById("video1");
19
20         function playPause() {
21             if (myVideo.paused)
22                 myVideo.play();
23             else
24                 myVideo.pause();
25         }
26
27         function makeBig() {
28             myVideo.width = 560;
29         }
30
31         function makeSmall() {
32             myVideo.width = 320;
33         }
34
35         function makeNormal() {
36             myVideo.width = 420;
37         }
38     </script>
39 </body>
40 </html>
```

For at du kan se dette eksempel køre, kræver det at din browser understøtter HTML 5's <video> tag – det gør de fleste browsere i dag, men skulle siden komme op på en 'ældre model', kan vi i stedet for videoen vise en kort besked om, at browseren ikke understøtter HTML 5 video. Dette princip kaldes for 'graceful degradation', hvor brugerens oplevelse af eventuelle fejl er i fokus, og så vidt muligt styres af udvikleren.

Get Element

Med getElement-metoderne har vi mulighed for at få fat i forskellige tags i vores applikation og manipulere med dem. Der findes 4 metoder til at definere hvilket objekt vi gerne vil have fat i:

`getElementByClassName`, `getElementById`, `getElementsByTagName`, `getElementsByAttribute`.

Vi vil kun beskæftige os med `getElementById` i denne lektion, men du er mere end velkommen til at udforske mulighederne i de andre på nettet.

Herunder er vist en række eksempler på brug af `getElementById`. Efter at have lavet disse fire eksempler skulle du gerne være i stand til at løse opgaverne, som kommer umiddelbart efter eksemplerne – sådan, at du kan lade en bruger selv have mulighed for at indtaste noget data, klikke på en knap, hvorefter dine funktioner kan behandle den indtastede data og vise resultatet.

Det første vi skal se på er hvordan vi egentlig får fat i et specifikt html-tag. Vi skal prøve at finde et bestemt div-tag og så hive indholdet ud af det.

```
GetElement_01.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6      <script type="text/javascript">
7          function GetElements() {
8              alert(document.getElementById('divID').innerHTML);
9          }
10     </script>
11 </head>
12 <body onload="GetElements()">
13     <div id="divID">Hej her er en Div box.</div>
14 </body>
15 </html>
```

I linie 7 laver vi en funktion som vi kalder GetElements.

I linie 8 kalder vi en alert og udskriver teksten fra det div-tag som har id divID.

I linie 12 kalder vi funktionen GetElements når siden loades.

I linie 13 laver vi et div-tag og giver det id'et divID og fylder noget tekst i det.

I det næste eksempel skal vi prøve at lave en funktion som kan tjekke om der står noget i et input-tag og udskrive teksten, hvis der gør eller bede én om at indtaste noget tekst hvis feltet er tomt.

```
GetElement_02.html  ↗ ✕
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6      <script type="text/javascript">
7          function PostBack() {
8              var strTextField = document.getElementById('Text');
9              if (strTextField.value != "")
10                 alert("Du indtastede: " + strTextField.value)
11             else
12                 alert("Du skal indtaste noget tekst!")
13         }
14     </script>
15 </head>
16 <body>
17
18     <input type="text" id="Text" />
19     <input type="button" onclick="PostBack()" value="Send" />
20
21 </body>
22 </html>
```

I linie 7 laver vi en funktion som vi kalder PostBack.

I linie 8 laver vi en variabel med navnet strTextField og fylder informationerne om vores input felt deri.

I linie 9 tjekker vi om der er indtastet noget i vores input-tag.

I linie 10: Er der indtastet noget, kalder vi en alert og udskiver teksten "Du indtastede:" efterfulgt af det indtastede.

I linie 12 udskriver vi teksten "Du skal indtaste noget tekst!" hvis der ikke er indtastet noget.

I linie 18 laver vi et input-tag og giver det id'et Text.

I linie 19 laver vi en knap hvor vi i attributten "onClick" kalder funktionen PostBack, når der klikkes på knappen.

I det tredje eksempel skal vi have lavet to input-tag til indtastning af to tal. Når vi klikker på en knap skal vi så hente værdien fra de to input-tag og lægge dem sammen. Derefter skriver vi resultatet ud i et p-tag.

```
GetElement_03.html
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>GetElement 03</title>
6      <script type="text/javascript">
7
8          function PostBack() {
9              var intTal1 = document.getElementById('Tal1');
10             var intTal2 = document.getElementById('Tal2');
11             var intRes = parseInt(intTal1.value) + parseInt(intTal2.value);
12             document.getElementById('Resultat').innerHTML = "Resultat: " + intRes;
13         }
14     </script>
15 </head>
16 <body>
17
18     <input type="text" id="Tal1" /> + <input type="text" id="Tal2" />
19     <br />
20     <input type="button" onclick="PostBack()" value="Send" />
21     <br />
22     <p id="Resultat"></p>
23
24 </body>
25 </html>
```

I linie 8 laver vi igen en funktion, som vi kalder PostBack.

I linie 9 og 10 laver vi to variabler, som vi fylder informationerne om vores inputfelter i.

I linie 11 laver vi en variabel, hvor vi konverterer indholdet af vores to input's til tal og lægger dem sammen.

I linie 12 finder vi tagget med id'et "Resultat". Heri udskriver vi så variabelen med resultatet.

I linie 18 laver vi to input-tag til indtastning af de to tal.

I linie 20 laver vi en knap hvor vi i attributten "onClick" kalder funktionen PostBack, når der klikkes på knappen.

I linie 22 laver vi et p-tag som vi kan udskrive resultatet i.

I det fjerde og sidste eksempel i denne lektion skal vi prøve dynamisk at skifte baggrundsfarven på en side, efter at have indtastet den i et tekstfelt.

```
GetElement_04.html  ▢ ×
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>GetElement 04</title>
6      <script type="text/javascript">
7          function PostBack() {
8              var strColor = document.getElementById('Color');
9              if (strColor.value != "") {
10                  document.bgColor = strColor.value;
11              }
12              else {
13                  alert("Du skal indtaste en farvekode!");
14              }
15          }
16      </script>
17  </head>
18  <body>
19      <input type="text" id="Color" /> eks. (#FF0000)
20      <br />
21      <input type="button" onclick="PostBack()" value="Skift" />
22  </body>
23  </html>
```

I linie 7 laver vi igen en funktion, som vi kalder PostBack.

I linie 8 laver vi en variabel, som vi fylder informationerne om vores input felt i.

I linie 9 tjekker vi om der er indtastet noget i vores input-felt.

I linie 10 ændrer vi sidens baggrundsfarve, hvis der er indtastet noget i vores input.

I linie 13 udskriver vi teksten "Du skal indtaste en farvekode!" hvis der ikke er indtastet noget i vores input-felt.

I linie 19 laver vi input-feltet til indtastning af en farvekode.

I linie 21 laver vi en knap hvor vi i attributten "onClick" kalder funktionen PostBack, når der klikkes på knappen.

Dynamic Style

I denne lektion skal vi se lidt på nogle af mulighederne for at manipulere med StyleSheet (CSS) via JavaScript. Der er kun vist tre korte eksempler men det burde være nok til at komme i gang med de uanede muligheder der er for at style sine sider via JavaScript.

I det første eksempel skal vi se på, hvordan vi med meget lidt kode kan lave en simpel dropdown-menu.

```
DynamicStyle_01.html  + X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Dynamic Style 01</title>
6      <script type="text/javascript">
7
8          function show(object) {
9              document.all[object].style.visibility = 'visible';
10             document.all[object].style.zIndex = 100;
11         }
12
13         function hide(object) {
14             document.all[object].style.visibility = 'hidden';
15         }
16     </script>
17 </head>
```

.. og samme fil, fortsat:

```
DynamicStyle_01.html  + X
18 <body>
19
20     <a href="#" onMouseOver="show('test');">Menu 1</a>
21     <br />
22     <div id="test" style="visibility: hidden; width: 100px;" onMouseOut="hide('test');">
23         - link 1 <br />
24         - link 2 <br />
25         - link 3 <br />
26         - link 4 <br />
27     </div>
28
29 </body>
30 </html>
```


I linie 8 laver vi en funktion, Show, som modtager et objekt.

I linie 9 sætter vi visibility prop'en til visible på det objekt som vi får navnet på i variabelen object. Dette gør at objektet vil blive vist på siden, hvis det er skjult.

I linie 10 sætter vi Z-index til 100. Så vi er sikre på at objektet ligger øverst på siden.

I linie 13-14 laver vi en funktion med navnet Hide, som modtager et objekt lige som Show gjorde.

I linie 14 sætter vi visibility til hidden på det objekt som vi får navnet på i variabelen object. Det gør at objektet vil blive skjult igen.

I linie 20 laver vi et link som kalder funktionen Show, når musen køres over det.

I linie 22 laver vi en div-boks som vi navngiver test og vi skjuler den. Vi kalder også Hide-funktionen når musen fjernes fra boksen.

I linie 23-26 fylder vi noget tekst i vores div-boks test.

I det næste eksempel herunder skal vi se lidt på hvordan vi finder et objekt med getElementById og styler det dynamisk.

```
DynamicStyle_02.html  ➤ ✕
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Dynamic Style 02</title>
6      <script type="text/javascript">
7          function show(object) {
8              if (object == 'test2') {
9                  document.getElementById('Box1').style.backgroundColor = '#FF0000';
10                 document.getElementById('Box1').style.borderStyle = 'solid';
11                 document.getElementById('Box1').style.borderWidth = '4px';
12                 document.getElementById('Box1').style.borderColor = '#0000FF';
13             }
14             else {
15                 document.getElementById('Box1').style.backgroundColor = '#c0c0c0';
16                 document.getElementById('Box1').style.borderStyle = 'solid';
17                 document.getElementById('Box1').style.borderWidth = '1px';
18                 document.getElementById('Box1').style.borderColor = '#FF0000';
19             }
20         }
21
22         function reset(object) {
23             document.getElementById('Box1').style.backgroundColor = "ffffff";
24             document.getElementById('Box1').style.borderWidth = '0px';
25         }
26     </script>
27 </head>
```

.. og samme fil, fortsat:

```
DynamicStyle_02.html  X
28  <body>
29      <a href="#" onMouseOver="show('test');" onMouseOut="reset('test');">Style 1</a>
30      <a href="#" onMouseOver="show('test2');" onMouseOut="reset('test2');">Style 2</a>
31      <br />
32      <div id="Box1" style="width: 100px; height: 100px;">
33          <br /> Her er en lille tekst.
34          <br />
35      </div>
36  </body>
37  </html>
```

I linie 7 laver vi en funktion vi kalder show som modtager et objekt.

I linie 8 tjekker vi om det modtagne objekt er lig med test2.

I linie 9-12 definerer vi så stylen på objektet, hvis det er lig med test2.

I linie 15-18 definerer vi så en anden style, hvis objektet ikke hedder test2

I linie 22 laver vi en funktion, som vi kalder reset som også modtager et objekt.

I linie 23-24 sætter vi baggrundsfarven til hvid og border til 0, så det ser ud som om vi har fjernet stylen igen.

I linie 29 og 30 laver vi to links, som kalder funktionerne med hver deres objektnavn når musen køres over eller væk fra linket.

I linie 32-35 laver vi den div-boks, som vi vil skifte stylen på når en af de to funktioner bliver kaldt.

I det 3. og sidste eksempel i denne lektion skal vi se på hvordan vi dynamisk kan få skiftet en class på et objekt, når musen køres over det.

Sæt en ny fil op i din editor, 'DynamicStyle_03.html' og skriv følgende kode ind i HEAD tag'et:

```
DynamicStyle_03.html  + X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Dynamic Style 03</title>
6      <script>
7          function changeClass(object, value) {
8              object.className = value;
9          }
10     </script>
11     <style type="text/css">
12         .normal {
13             color: white;
14             font-weight: bold;
15             border: 1px solid black;
16             padding: 2px;
17             background-color: #000000;
18             text-decoration: none;
19         }
20
21         .hover {
22             color: black;
23             font-weight: bold;
24             border: 1px solid black;
25             padding: 2px;
26             background-color: #C5C5C5;
27             text-decoration: none;
28         }
29     </style>
30 </head>
```

.. og samme fil, fortsat med Body tag'et:

```
DynamicStyle_03.html  X
31 <body>
32   <table>
33     <tr>
34       <td>
35         <a href="http://www.panmedia.dk" target="_blank">
36           <div class="normal" onmouseover="changeClass(this,'hover')"
37             onmouseout="changeClass(this,'normal')">
38             Panmedia ApS
39           </div>
40         </a>
41       </td>
42     </tr>
43     <tr>
44       <td>
45         <a href="http://sang-skriver.azurewebsites.net" target="_blank">
46           <div class="normal" onmouseover="changeClass(this,'hover')"
47             onmouseout="changeClass(this,'normal')">
48             Århus Sangskriverværksted
49           </div>
50         </a>
51       </td>
52     </tr>
```

.. og samme fil, fortsat:

```
DynamicStyle_03.html  X
53   <tr>
54     <td>
55       <a href="http://www.frogpill.dk" target="_blank">
56         <div class="normal" onmouseover="changeClass(this,'hover')"
57           onmouseout="changeClass(this,'normal')">
58           Frog Pill Inc.
59         </div>
60       </a>
61     </td>
62   </tr>
63 </table>
64 </body>
65 </html>
```

I linie 7-9 laver vi funktionen ChangeClass, som skal bruges til at skifte klassen på et objekt. Funktionen modtager et objektnavn og navnet på en class og skifter den ud fra de oplysninger der er med via funktionskaldet.

I linie 11 til 29 laver vi et lille stylesheet med to klasser, normal og hover.

I linie 32 til 63 laver vi en tabel, hvor vi sætter nogle links rundt om nogle div-bokse. Div-boksene kalder vores funktion ChangeClass, når musen køres over dem eller væk fra dem med funktionen this og navnet på den class den skal have. Keywordet 'this' i javascript betyder 'det aktuelle objekt' og definerer at det er dette objekt som skal styles.

Opgaver til JavaScript

Opgave 1

Lav en lille funktion, der som input modtager et tal. Når siden afvikles, vises hvilket tal der er indtastet og hvor mange gange tallet kan divideres med to.

Opgave 2

Udvid opgaven, så man nu kan indtaste en divisor i et tekstfelt, når funktionen kaldes, i stedet for at dividere med to hver gang

Opgave 3

Lav en lille funktion, der som input modtager et antal timer, minutter og sekunder i tre variabler og returnerer hvor mange sekunder det svarer til i alt.

F.eks. Input: 1 time, 4 minutter og 12 sekunder

Output: 3852 sekunder

Opgave 4

Lav derefter den omvendte beregning, hvor der som input skrives et antal sekunder i en variabel og svaret udskrives på skærmen i timer, minutter og sekunder

F.eks. Input: 3852 sekunder

Output: 1 time, 4 minutter og 12 sekunder

Opgave 5

Lav en lille funktion, der som input tager en tekst. Når den køres får man svar på, hvor mange tegn, der er i strengen.

Opgave 6

Lav en funktion, der kan modtage en tekst og få det skrevet ud på skærmen, med henholdsvis store og små bogstaver.

Lav også en udgave, hvor første bogstav er med stort og resten med småt.

Opgave 7

Lav en funktion, der som input modtager en tekst og derefter viser de første 5 tegn i en linie, viser 5 tegn inde i teksten i en anden linie og de sidste 5 tegn i en tredje linie. I en fjerde linie udskrives antallet af tegn i teksten. Selve teksten skal også udskrives i en separat linie.

Opgave 8

Lav en funktion, der som input modtager henholdsvis en tekst og et tegn og som viser, hvor mange gange det indtastede tegn forekommer i teksten.

F.eks. Input: Dette er en tekst

Søgetegn: t

Output: 4

Opgave 9

Lav en lille funktion, som modtager en tekst og som retur får teksten skrevet både retvendt og baglæns

F.eks. Input: rosentyper

Output: repytnesor

Opgave 10

Lav en lille funktion, som kan modtage fornavn, mellemnavne og efternavn og som retur få vist Efternavn, Fornavn Mellemnavn i nævnte rækkefølge.

F.eks. Input: Hans Erik Hansen

Output: Hansen, Hans Erik

Opgave 11

Lav en lille funktion, der som input modtager et fornavn og et efternavn. Når den køres, udskrives følgende:

HEJ "Fornavn Efternavn"

VELKOMMEN

Har man ikke udfyldt fornavn eller efternavn, så skal velkomstteksten være HEJ Fremmede.

Opgave 12

Lav en funktion, som kan modtage et tal og hvor man som svar får udskrevet talrækken fra 1 til og med det indtastede tal. Tallene skal være komma-separerede.

Opgave 13

I opgave 6 skrev man det første bogstav med stort og resten med småt. I denne opgave skal hvert andet bogstav skrives med stort og hver andet med småt.

Opgave 14

I stedet for, som i opgave 6, at skrive hvert andet bogstav med stort, kan man jo også lægge en farve på de store bogstaver og en anden farve på de små bogstaver

Jeg har på de følgende lavet en række små scripts som du kan bruge eller lave for øvelsens skyld.

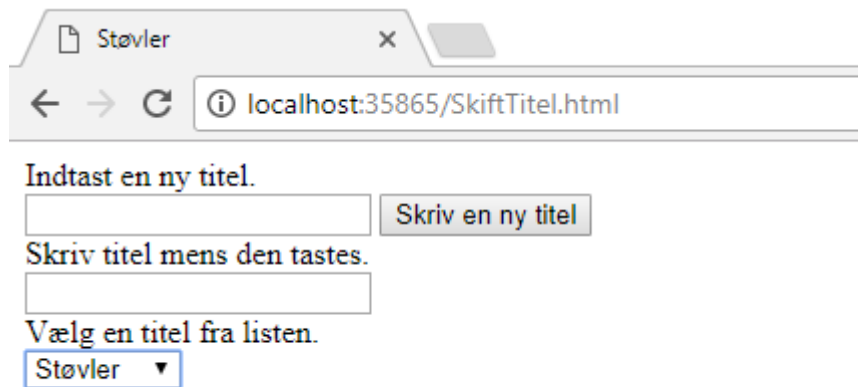
Kald af funktioner fra formelementer.

I eksemplet her under demonstreres tre forskellige måder at kalde en funktion fra nogle forskellige formelementer. Tilføj en html-fil til dit projekt kald den SkiftTitel.htm.

Tilføj koden her under til filen du lige har oprettet. Når du afvikler siden, kan du skifte sidens titel på tre måder (hold øje med fanebladet øverst, det er der sidens titel vises og opdateres)

```
SkiftTitel.html  + X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>Sidens titel</title>
5      <script>
6          function changeTitle(tekst) {
7              document.title = tekst;
8          }
9      </script>
10 </head>
11 <body>
12     <form>
13         Indtast en ny titel.
14         <br />
15         <input type="text" name="text">
16         <input type="button" onclick="changeTitle(this.form.text.value)" value="Skriv en ny titel">
17     </form>
18     Skriv titel mens den tastes.
19     <br />
20     <input type="text" onkeyup="changeTitle(this.value)">
21     <br />
22     Vælg en titel fra listen.
23     <br />
24
25     <select onchange="changeTitle(this.value)">
26         <option value="Sko">Sko</option>
27         <option value="Støvler">Støvler</option>
28         <option value="Sandaler">Sandaler</option>
29     </select>
30 </body>
31 </html>
```

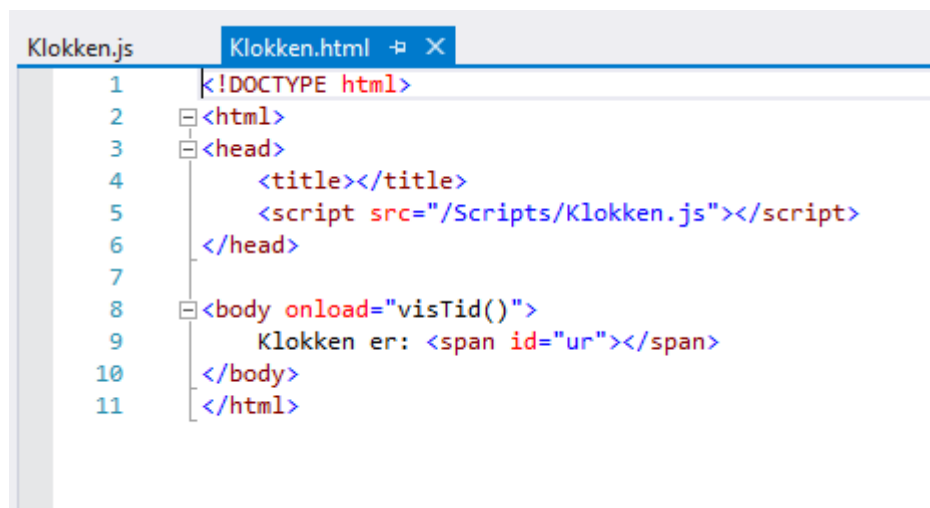

Output:



A screenshot of a web browser window. The tab is titled 'Støvler'. The address bar shows 'localhost:35865/SkiftTitel.html'. The page content includes three instructions: 'Indtast en ny titel.' followed by a text input field and a 'Skriv en ny titel' button; 'Skriv titel mens den tastes.' followed by another text input field; and 'Vælg en titel fra listen.' followed by a dropdown menu currently showing 'Støvler'.

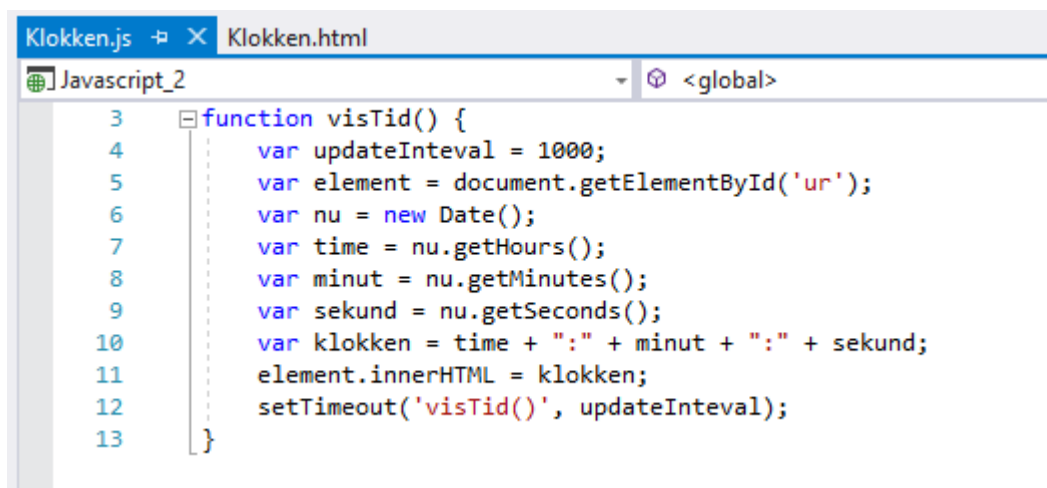
Klokken der går

Et eksempel på en klokke der går. Opret en ny HTML-fil, 'Klokken.html'. Tilføj koden her under til filen du lige har oprettet.



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title></title>
5     <script src="/Scripts/Klokken.js"></script>
6 </head>
7
8 <body onload="visTid()">
9     Klokken er: <span id="ur"></span>
10 </body>
11 </html>
```

Tilføj en ny JavaScript-fil med navnet Klokken.js til din 'Scripts' mappe, og tilføj følgende kode til din script-fil:



```
3 function visTid() {  
4     var updateInterval = 1000;  
5     var element = document.getElementById('ur');  
6     var nu = new Date();  
7     var time = nu.getHours();  
8     var minut = nu.getMinutes();  
9     var sekund = nu.getSeconds();  
10    var klokken = time + ":" + minut + ":" + sekund;  
11    element.innerHTML = klokken;  
12    setTimeout('visTid()', updateInterval);  
13 }
```

Højre klik forbudt

I eksemplet her ses et eksempel på hvordan man kan lave en funktion hvis en event udløses.

Tilføj en html-fil til projektet, kald den HøjreKlikForbudt.htm.

Tilføj koden her under til filen du lige har oprettet.

```
HøjreklikForbudt.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6      <script type="text/javascript">
7          document.oncontextmenu = function () {
8              alert('Højreklik forbudt!!!');
9              // note: udkommenter return false for at se kontekstmenuen v højreklik
10             return false;
11          };
12      </script>
13
14  </head>
15  <body>
16      Her må ikke højre klikkes.
17  </body>
18  </html>
```

Countdown

I eksemplet her skal vi se på hvordan vi kan kalde en funktion med et tidsinterval og kald af flere funktioner samtidig.

Tilføj en html-fil til projektet, kald den Countdown.html. Tilføj koden her under til filen du lige har oprettet.

```
Countdown.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Countdown</title>
6
7      <script type="text/javascript">
8          var pause = 3000;
9          function countdown() {
10              if (pause <= 0) window.location = "http://www.wired.com";
11              else {
12                  document.getElementById('tid').innerHTML = pause / 1000 + ' ';
13                  pause = pause - 1000;
14              }
15          }
16      </script>
17  </head>
18  <body onload="countdown();setInterval('countdown()', 1000)">
19      Du viderestilles til WIRED MAGAZINE om <span id="tid"></span>sekunder.
20  </body>
21  </html>
```

Keypress

Her under er et simpelt eksempel på håndtering af tastaturtryk. Tilføj en html-fil til projektet, kald den KeyPress.htm.

Tilføj koden her under til filen du lige har oprettet.

```
KeyPress.html  [icon] X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6      <script type="text/javascript">
7
8          // tryk på mellemrumstasten for at blive videresendt..
9          function handler(e) {
10              if (e.charCode == "32") {
11                  window.location = "http://www.wired.com";
12              } else {
13                  alert(e.charCode);
14              }
15          }
16          document.onkeypress = handler;
17      </script>
18  </head>
19  <body>
20
21  </body>
22  </html>
```

Tryk på mellemrumstasten for at blive sendt til www.wired.com.

Udskriv side

Tilføj en html-fil til projektet, kald den Print.html. Tilføj koden herunder til filen du lige har oprettet.

```
Print.html  + X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6      <script type="text/javascript">
7          function printPage() {
8              window.print();
9          }
10     </script>
11 </head>
12 <body>
13
14     <input type="button" value="Udskriv denne side" onclick="printPage()">
15     <!--<a href="#" onclick="printPage()">Udskriv denne side</a-->
16 </body>
17 </html>
```

Tilfældigt billede

I eksemplet her skal vi lave en funktion der kan vise et tilfældigt billede hver gang siden opdateres.

Tilføj mappen RandomImages til dit projekt (hent den fra materialer.zip, der ligger i jeres fb gruppe), mappen indeholder nogle forskellige billeder. Tilføj en html-fil til dit projekt, kald den RandomImage.html.

Tilføj koden her under til filen du lige har oprettet.

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Random Images</title>
6  </head>
7  <body>
8
9      <script type="text/javascript">
10         // note: reload page w F5 to see next random cat image
11         // usage ex: place script on landing page to randomize site background pic
12         var img = new Array();
13         img[0] = '1.jpg';
14         img[1] = '2.jpg';
15         img[2] = '3.jpg';
16         img[3] = '4.jpg';
17         img[4] = '5.jpg';
18         img[5] = '6.jpg';
19         img[6] = '7.jpg';
20         img[7] = '8.jpg';
21         img[8] = '9.jpg';
22         var random = Math.round(Math.random() * (img.length - 1));
23         document.write('');
24     </script>
25
26 </body>
27 </html>
```

Bannerrotation

En traditionel bannerrotation der skifter på tid. Tilføj mappen BannerImages til dit projekt, og kopier filerne fra mappen BannerImages under Materialer.zip ind i din nye projektmappe.

Tilføj en html-fil til projektet, kald den ImageRotation.html. Tilføj koden her under til filen du lige har oprettet.

```
ImageRotation.html  ↗ ✕
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>Banner Rotation Example</title>
5      <script src="Scripts/ImageRotation.js" type="text/javascript"></script>
6  </head>
7  <body onload="rotator()">
8      <div id="banner"> </div>
9  </body>
10 </html>
```

Tilføj en script-fil til din Scripts mappe med navnet ImageRotation.js – og tilføj følgende kode til filen.

```
ImageRotation.js  ↗ ✕  ImageRotation.html
Javascript_2      ↗  rotator  ↗  imgString
1  //Pause inden nyt billede vises. Angives i millisekunder.
2  var pause = 2000;
3
4  var img = new Array();
5  img[0] = 'Guitar_1.jpg';
6  img[1] = 'Piano_1.jpg';
7  img[2] = 'Piano_2.jpg';
8  var imgID = 0;
9  imgID = Math.round(Math.random() * (img.length - 1));
10
11 function rotator() {
12     var imgString = "";
13     imgString = imgString + '';
14     if (imgID == img.length - 1)
15     {
16         imgID = 0;
17     }
18     else
19     {
20         imgID++;
21     }
22     document.getElementById('banner').innerHTML = imgString;
23     setTimeout('rotator()', pause);
24 }
```

Galleri

Et simpelt eksempel på hvordan man med JavaScript kan lave et brugbart billedgalleri. Tilføj en html-fil til projektet, kald den Galleri.html.

Tilføj koden her under til filen du lige har oprettet.

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title></title>
5   <link href="CSS/Galleri.css" rel="stylesheet" type="text/css" />
6   <script src="Scripts/Galleri.js" type="text/javascript"></script>
7 </head>
8 <body>
9   <div>
10    <div id="imgHolder">
11    </div>
12  </div>
13  <div id="wrapper">
14    <div id="imgBox">
15      <a onclick="javascript:ShowImage('none');">
16        <div id="myImg"></div>
17      </a>
18    </div>
19  </div>
20  <script type="text/javascript">
21    loadImages("cirkel1.gif,cirkel2.gif,cirkel3.gif,cirkel4.gif,cirkel5.gif,roed.gif,groen.gif,gul.gif");
22  </script>
23 </body>
24 </html>
```

Tilføj et style sheet med navnet Galleri.css og tilføj følgende kode der til.


```
Galleri.css  X  Galleri.js  X  Galleri.html
1  .images {
2      float: left;
3      height: 105px;
4      width: 140px;
5      margin: 5px;
6      border: 1px solid black;
7      cursor: pointer;
8  }
9
10 #wrapper {
11     position: absolute;
12     z-index: 2;
13     width: 90%;
14     margin-top: 50px;
15 }
16
17 #imgBox {
18     margin-left: auto;
19     margin-right: auto;
20     width: 760px;
21     background-color: Black;
22 }
23
24 #bigImg {
25     margin: 20px;
26     border: 0px;
27     cursor: pointer;
28 }
```

Tilføj en JavaScript-fil, kald den Galleri.js og tilføj følgende kode der til. NB! I linie 6,7, og 8 herunder, er én lang linie kode delt i tre for at kunne passe på screendumpet til siden – når du har indtastet de tre linier, skal de samles til én linie kode, der ser sådan her ud:

```
"<a onClick=\"javascript:ShowImage('\" + arrImages[i] + '\"');\"> <img
src=\"GalleryImages/Thumb/\" + arrImages[i] + \"\" class=\"images\" alt=\"Vis billede\"
title=\"Vis billede\" /></a>";
```

```

Galleri.css  Galleri.js  X  Galleri.html
Javascript_2  loadImages  i
1  function loadImages(images) {
2      var arrImages = images.split(",");
3      var html = "";
4      for (var i = 0; i < arrImages.length; i++) {
5          html +=
6              "<a onClick=\"javascript:ShowImage('\" + arrImages[i] + \"')\">
7              \> <img src=\"GalleryImages/Thumb/\" + arrImages[i] + \"
8              \>\" class=\"images\" alt=\"Vis billede\" title=\"Vis billede\" /></a>";
9      }
10     imgPreloader(arrImages, "GalleryImages/Big/");
11     document.getElementById('imgHolder').innerHTML = html;
12 }
13
14 function ShowImage(image) {
15     if (image == 'none') {
16         document.getElementById('myImg').innerHTML = '';
17     } else {
18         document.getElementById('myImg').innerHTML =
19             '';
21     }
22 }
23
24 function imgPreloader(arrImages, folder) {
25     var preImg = new Image();
26     for (var i = 0; i < arrImages.length; i++) {
27         preImg.src = folder + arrImages[i];
28     }
29 }

```

SlideInSideMenu

I det følgende eksempel vil vi lave en side menu, der 'slider' ind fra venstre side af skærmen. Start med at oprette en ny .js fil i din 'Scripts' folder, kald den 'SlideInSideMenu.js':

```

SlideInSideMenu.js  X  SlideInSideMenu.css  SlideInSideMenu.html
Javascript_2  closeNav
1  function openNav() {
2      document.getElementById("mySidenav").style.width = "250px";
3      document.getElementById("main").style.marginLeft = "250px";
4      document.body.style.backgroundColor = "rgba(0,0,0,0.4)";
5  }
6
7  function closeNav() {
8      document.getElementById("mySidenav").style.width = "0";
9      document.getElementById("main").style.marginLeft = "0";
10     document.body.style.backgroundColor = "white";
11 }

```

Nu skal vi have sat noget CSS op, der vil få vores slide menu til at tage sig godt ud i sammenhængen. Lav en ny CSS fil i din CSS mappe, kald den 'SlideInSideMenu.css':

.. og samme fil, fortsat:

```
SlideInSideMenu.js  SlideInSideMenu.css  SlideInSideMenu.html
27
28     .sidenav a:hover {
29         color: #f1f1f1;
30     }
31
32     .sidenav .closebtn {
33         position: absolute;
34         top: 0;
35         right: 25px;
36         font-size: 36px;
37         margin-left: 50px;
38     }
39
40     #main {
41         transition: margin-left .5s;
42         padding: 16px;
43     }
44
45     @media screen and (max-height: 450px) {
46         .sidenav {
47             padding-top: 15px;
48         }
49
50         .sidenav a {
51             font-size: 18px;
52         }
53     }
54
```

Nu kan vi sætte vores HTML side op i projektet, med referencer i Head tag'et til de to andre filer:

```
SlidelnSideMenu.js  SlidelnSideMenu.css  SlidelnSideMenu.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <link href="CSS/SlideInSideMenu.css" rel="stylesheet" />
5    <script type="text/javascript" src="Scripts/SlideInSideMenu.js"></script>
6  </head>
7  <body>
8
9    <div id="mySidenav" class="sidenav">
10      <a href="javascript:void(0)" class="closebtn" onclick="closeNav()">&times;</a>
11      <a href="#">Om os</a>
12      <a href="#">Services</a>
13      <a href="#">Kunder</a>
14      <a href="#">Kontakt</a>
15    </div>
16
17    <div id="main">
18      <h2>Sidenav Push Eksempel</h2>
19      <p>
20        Klik på elementet nedenunder for at åbne en side navigationsmenu, og skubbe content til højre.
21        Bemærk, at body i doukumentet får en mørk gennemsigtig baggrundsfarve, når sidemenuen er aktiv.
22      </p>
23      <span style="font-size:30px;cursor:pointer" onclick="openNav()">&#9776; menu</span>
24    </div>
25
26  </body>
27  </html>
```

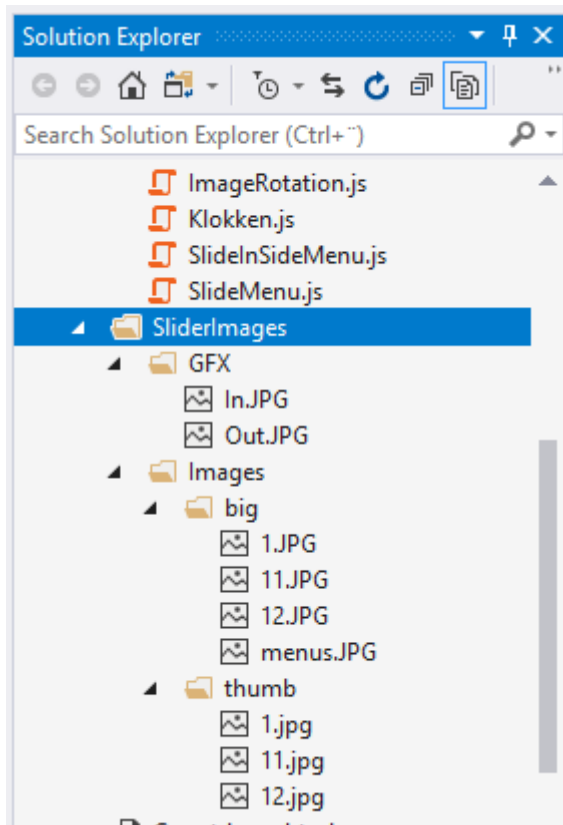
Byg eksemplet, og test resultatet i din browser. Opfører menuen sig, sådan som du ville forvente?

Ekstraopgave: byg en af de fire sider, der er repræsenteret ved # links i menuen, og sæt menuen til at linke til denne i stedet for #. Hvordan kan du nu komme retur til din hovedside, hvis du klikker på linket i menuen?

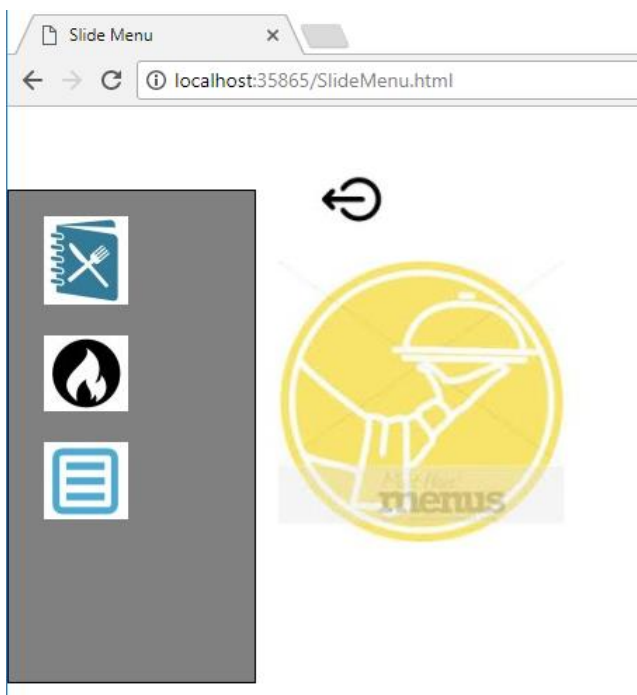
Slide menu med ikoner

Et eksempel på en animeret slide menu med billeder. Opret en ny html fil, kald den SlideMenu.html. Opret også en folder til billeder i dit projekt, kald den for 'SliderImages'

Tilføj undermapperne Gfx og Images til SliderImages, igen med to underfoldere Big og Thumb under 'Images'. Din folderstruktur i projektet skal gerne matche den, du kan finde i materialer.zip under 'SliderImages'. og her finder du også de rigtige billedfiler, som skal tilføjes for at menuen vil virke korrekt:



Vi vil gerne bygge en side, der ender med at have dette udseende:



Tilføj en html-fil til projektet, kald den SlideMenu.htm.

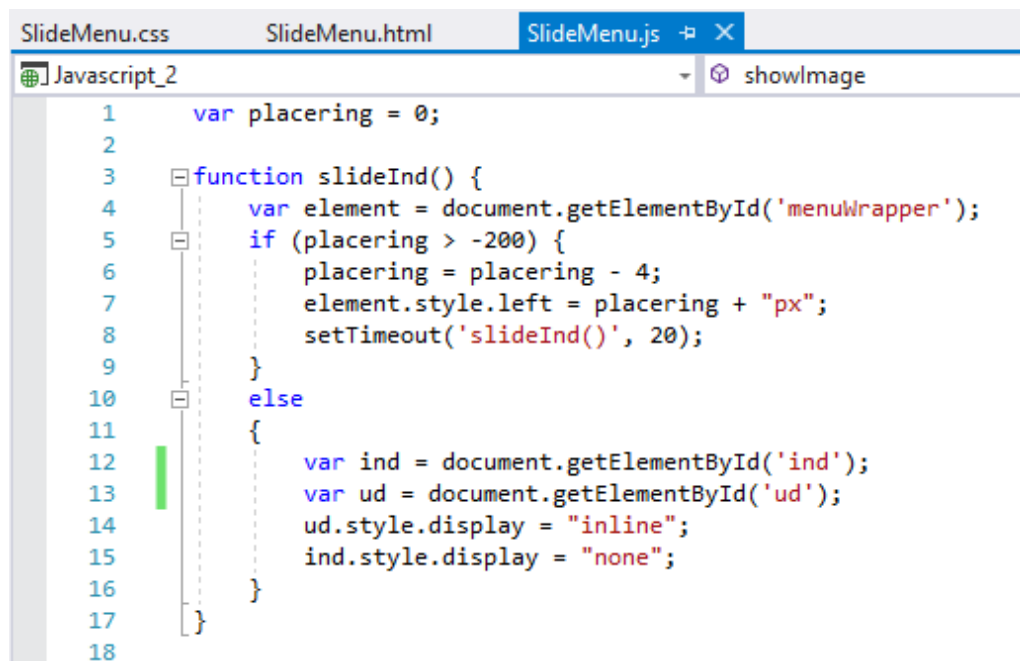
Tilføj koden herunder til filen du lige har oprettet.

```
SlideMenu.css  SlideMenu.html  SlideMenu.js
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Slide Menu</title>
6      <link href="CSS/SlideMenu.css" rel="stylesheet" type="text/css" />
7      <script src="Scripts/SlideMenu.js" type="text/javascript"></script>
8  </head>
9  <body>
10
11      
12      <div id="menuWrapper">
13          <div id="top">
14              <a id="ind" onclick="slideInd()">
15                  
16              </a><a id="ud" onclick="slideUd()">
17                  
18              </a>
19          </div>
20          <div id="menu">
21              <br />
22              
23              <br />
24              <br />
25              
26              <br />
27              <br />
28              
29          </div>
30      </div>
31
32  </body>
33  </html>
```

Tilføj et style sheet med navnet SlideMenu.css til din CSS mappe i projektet, og tilføj følgende kode der til.

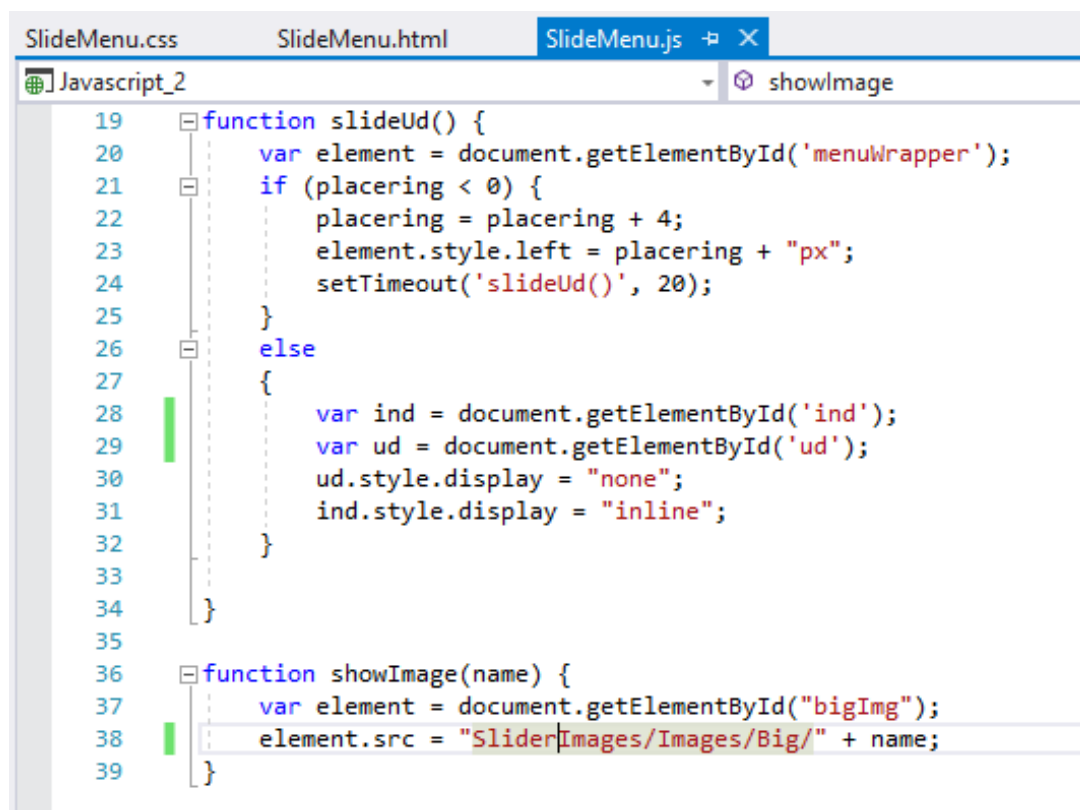
```
SlideMenu.css  X SlideMenu.html  SlideMenu.js
1  body {
2      margin-left: 0px;
3  }
4
5  #menuWrapper {
6      position: relative;
7      left: 0px;
8      top: 20px;
9      width: 150px;
10     height: 350px;
11     z-index: 2;
12 }
13
14 #menu {
15     width: 150px;
16     height: 350px;
17     background-color: Gray;
18     border: 1px solid black;
19     padding-left: 25px;
20 }
21
22 #ud {
23     display: none;
24 }
25
26 #bigImg {
27     position: absolute;
28     top: 10px;
29     left: 10px;
30     z-index: 1;
31 }
32
```


Tilføj en JavaScript-fil til din 'Scripts' folder i projektet, kald den SlideMenu.js og tilføj følgende kode der til.



```
SlideMenu.css  SlideMenu.html  SlideMenu.js  X
Javascript_2  showImage
1  var placering = 0;
2
3  function slideInd() {
4      var element = document.getElementById('menuWrapper');
5      if (placering > -200) {
6          placering = placering - 4;
7          element.style.left = placering + "px";
8          setTimeout('slideInd()', 20);
9      }
10     else
11     {
12         var ind = document.getElementById('ind');
13         var ud = document.getElementById('ud');
14         ud.style.display = "inline";
15         ind.style.display = "none";
16     }
17 }
18
```

.. og samme fil, fortsat:



```
SlideMenu.css  SlideMenu.html  SlideMenu.js  X
Javascript_2  showImage
19 function slideUd() {
20     var element = document.getElementById('menuWrapper');
21     if (placering < 0) {
22         placering = placering + 4;
23         element.style.left = placering + "px";
24         setTimeout('slideUd()', 20);
25     }
26     else
27     {
28         var ind = document.getElementById('ind');
29         var ud = document.getElementById('ud');
30         ud.style.display = "none";
31         ind.style.display = "inline";
32     }
33 }
34
35
36 function showImage(name) {
37     var element = document.getElementById("bigImg");
38     element.src = "Slider/Images/Images/Big/" + name;
39 }
```

Test funktionaliteten i din browser- i dette eksempel er det i højere grad javascript, der styrer hvad der sker på skærmen, hvor det forrige eksempel udnyttede css transitions. Hvis du senere kan se et brugsscenarie, hvor du kan udnytte enten den ene eller den anden vinkel på en slider menu, så kan du prøve at tilpasse eksemplerne her til et af dine egne projekter.

Introduktion til jQuery

jQuery er et JavaScript bibliotek, der har til formål at lette programmeringsprocessen for såvel nye som mere erfarne JavaScript brugere. Med jQuery kan man tilføje avancerede JavaScript funktioner med meget lidt kode, ved at bruge stumper fra det færdige programmerede bibliotek og de mange plugins, som tredjepart har skrevet til brug med jQuery. jQuerys slogan er "write less, do more", hvilket giver ret god mening hvis man kender historien bag jQuerys opbygning. For en god ordens skyld skal jeg lige nævne at jQuery ikke er det eneste interessante JavaScript bibliotek men et godt sted at starte. Af andre biblioteker kan jeg blandt andet nævne, Prototype, CoffeeScript, Scriptaculous, Dojo, MooTools og Spry, nogle mere kendte end andre.

Før du starter på jQuery er det en god ide at have et godt kendskab til HTML, CSS og JavaScript.

Opsætning af jQuery

jQuery er ganske nemt at tilføje til dit website, du skal blot downloade biblioteket, lægge det i dit projekt og så tilføje en enkelt linje på de sider som skal bruge biblioteket og så er du flyvende.

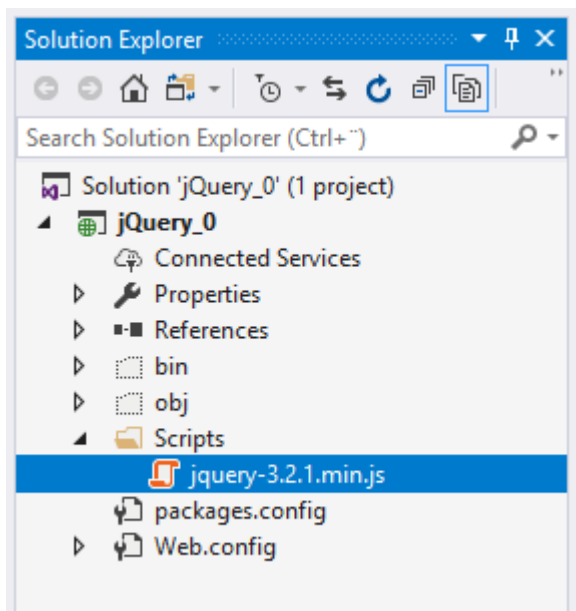
Biblioteket kan downloades fra www.jquery.com. jQuery funktionerne er kodet i en enkelt JavaScript-fil, som du kan lægge i dit 'Scripts' bibliotek og hele biblioteket kan tilføjes med en enkelt linje som vist her under.

```
<head>
```

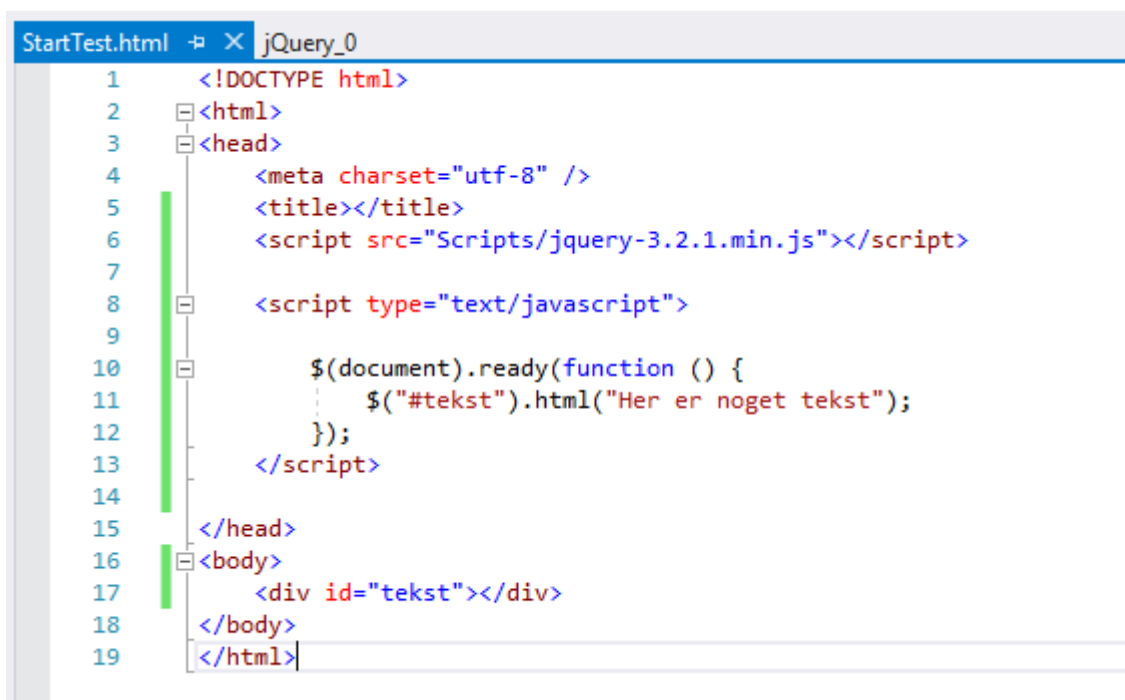
```
<script type="text/javascript" src="Scripts/jquery-3.2.1.min.js"></script>
```

```
</head>
```

For at kunne arbejde med de følgende jQuery eksempler, så opret et nyt projekt 'jQuery_0', og læg den jQuery fil du henter på jQuery.com i en ny 'Scripts' folder, som du tilføjer til projektstrukturen:



Her under er vist et eksempel på brug af en jQuery funktion.



Den grundlæggende jQuery syntaks er bygget op så man først vælger et HTML-element og derefter en action/funktion. Hvis vi ser på linje 11 i eksemplet her over vælger vi elementet med Id "tekst", og tilføjer teksten "Her er noget tekst" som html værdien (innerHTML tag'et - for nørder). Se eksemplet her under.

```
$("#tekst").html("Her er noget tekst");
```

```
$(SELECTOR).ACTION(VÆRDI);
```

Dollartegnet (\$) definerer at det er en jQuery funktion vi vil kalde, på den måde ved browseren at funktionen skal findes i biblioteket. I linje 10-12 laver vi en funktion document.ready(). Ved at sætte vores jQuery kode ind i denne funktion, forhindrer vi at vores jQuery kode bliver kørt, før hele dokumentet er indlæst. Grunden til dette er hvis ikke at hele dokumentet er indlæst og vi prøver for eksempel at skrive i en div som ikke er indlæst endnu, så vil vi få en fejl på siden.

Valg af elementer

En af de mere vigtige ting i jQuery er valg af elementer. For at du kan manipulere med HTML-elementer er det vigtigt at du har valgt de rigtige elementer at ændre på. Med jQuery er det muligt at ændre et eller flere elementer af gangen. Udvælgelsesmetoderne kan overordnet deles ind i tre kategorier, Element, Attribut og CSS selectors. Herunder er beskrevet en række metoder til udvælgelse af elementer.

Element Selectors

Valg af elementer via tag, class og id.

`$("div")` : Vælger alle div-tags.

`$("div.myClass")` : Vælger alle div-tags hvor class=" myClass ".

`$("div#myDiv")` : Vælger alle div-tags hvor id=" myDiv ".

Attribute Selectors

Det er også muligt at vælge elementer ud fra deres attributter, for eksempel hvis vi skal vælge alle elementer der har en width eller en bestemt width.

`$("[width]")` : Vælger alle elementer der har en width attribut.

`$("[width ='300px']")` : Vælger alle elementer der har en width på 300px.

`$("[width !='300px']")` : Vælger alle elementer der IKKE har en width på 300px;

`$("[width$='px']")` : Vælger alle elementer der har en width der slutter med px.

CSS Selectors

Det er også muligt at vælge et element ud fra dets CSS class eller id.

`$(".myClass")` : Vælger alle elementer hvor class=" myClass ".

`$("#myID")` : Vælger elementet hvor id=" myID ".

Events

jQuery indeholder også funktionalitet til håndtering af events. En event er når en bestemt hændelse opstår som vi så i eksemplet tidligere kan det være når et dokument er færdigindlæst eller der trykkes på en knap eller lignende.

Til at håndtere events har vi nogle bestemte funktioner, kaldet eventhandlers, lige som `(document).ready` funktionen fra det første eksempel. Eksemplet herunder viser hvordan man nemt kan håndtere et klik på en knap.

```
ClickEvent.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Click Event</title>
6      <script src="Scripts/jquery-3.2.1.min.js"></script>
7
8      <script type="text/javascript">
9          $(document).ready(function () {
10              $("button").click(function () {
11                  $("div").hide();
12              });
13          });
14      </script>
15
16  </head>
17  <body>
18
19      <button>Slå mig</button>
20      <div>Jeg er en divbox!!!</div>
21
22  </body>
23  </html>
```

CSS manipulation

jQuery indeholder også en del funktionalitet til manipulation med CSS. Her under gennemgår vi de mest brugte metoder.

Læsning af en CSS-property

Det er muligt at læse CSS værdierne på de Properties, der er sat i din CSS. Byg eksemplet herunder for at se rgb-værdien af backgroundColor: gray.

```
GetCSSValue.html  ➤ ✕
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Get CSS Value</title>
6
7      <style type="text/css">
8          #myDiv {
9              width: 300px;
10             height: 300px;
11             background-color: gray;
12         }
13     </style>
14
15     <script src="Scripts/jquery-3.2.1.min.js"></script>
16
17     <script type="text/javascript"> $(document).ready(function () {
18         alert($("#myDiv").css("background-color"));
19     });
20 </script>
21 </head>
22 <body>
23     <div id="myDiv">Her er en boks.</div>
24 </body>
25 </html>
```


Tilføj en værdi til en CSS-property

```
SetCSSValue.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <meta charset="utf-8" />
5    <title></title>
6
7    <style type="text/css">
8      #myDiv {
9        width: 300px;
10       height: 300px;
11       background-color: gray;
12     }
13   </style>
14
15   <script src="Scripts/jquery-3.2.1.min.js"></script>
16
17   <script type="text/javascript">
18     $(document).ready(function () {
19       $("#myDiv").css("background-color", "blue");
20     });
21   </script>
22 </head>
23
24 <body>
25   <div id="myDiv">Her er en boks.</div>
26 </body>
27
28 </html>
29
```

I filen SetCssValue.html sætter du initielt værdien af background-color for feltet på siden til 'grey' som i det forrige eksempel der hvor du deklarerer stylen øverst i dokumentet (hint: genbrug koden!) men i doc ready funktionen sætter du så css background-color prop'en til blue i stedet. Når siden loader, vil du derfor se en blå baggrund i div boksen.

Tilføje værdier til flere CSS-properties ad gangen

SetCssMultiValue.htm

```
SetCssMultiValue.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6      <style type="text/css">
7          #myDiv {
8              width: 300px;
9              height: 300px;
10             background-color: gray;
11         }
12     </style>
13
14     <script src="Scripts/jquery-3.2.1.min.js"></script>
15
16     <script type="text/javascript"> $(document).ready(function () {
17         $("#myDiv").css({ "background-color": "blue", "width": "500px" });
18     });
19     </script>
20 </head>
21 <body>
22     <div id="myDiv">Her er en boks.</div>
23 </body>
24 </html>
```

Det er også muligt at tilføje eller fjerne hele CSS classes, se eksemplet her under:

```
$("#myDiv").addClass("myClass");
```

```
$("#myDiv").removeClass("myClass");
```

Skift imellem to classes:

```
$("#myDiv").toggleClass("myClass2");
```

Effekter

Der er en lang række indbyggede effekter i jQuery, en af dem er Hide som vi har set på tidligere men der findes flere langt mere imponerende effekter som ikke nødvendigvis kræver en hel masse kode. Jeg vil på de efterfølgende sider demonstrere nogle stykker af dem.

I eksemplet her under vises hvordan du dynamisk kan vise og skjule objekter med jQuery.

```
ShowHide.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6      <script src="Scripts/jquery-3.2.1.min.js"></script>
7
8      <script type="text/javascript">
9          $(document).ready(function () {
10              $("#hide").click(function () {
11                  $("div").hide();
12              });
13              $("#show").click(function () {
14                  $("div").show();
15              });
16          })
17      </script>
18  </head>
19  <body>
20
21      <button id="hide">Skjul</button>
22      <button id="show">Vis</button>
23      <div>Her er noget tekst!!!</div>
24
25  </body>
26  </html>
```

Byg eksemplet, og test det i din browser. Du skulle gerne kunne vise og gemme en linie tekst på skærmen ved at klikke på de to knapper øverst på siden.

Hide() og show() funktionerne kan modtage to parametre, se eksemplet her under.

```
$("div").hide(2000, doSomething() );
```

Den første parameter, tallet skrevet med rødt er hvor lang tid det skal tage at skjule det valgte element, i millisekunder. Den anden parameter, skrevet med grønt, er en mulighed for at definere en funktion eller lignende der skal afvikles når elementet skjules. Dette bliver populært kaldt for et **callback** og sådan en callback-funktion findes på flere af jQuerys funktioner.

Prøv eventuelt at udvide dit eksempel med de to parametre som vist her under.

```
HideCallback_01.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Hide Callback 01</title>
6      <script src="Scripts/jquery-3.2.1.min.js"></script>
7
8      <script type="text/javascript">
9          function doSomething() {
10              alert("Jeg er doSomething!!!");
11          }
12
13          $(document).ready(function () {
14              $("#hide").click(function () {
15                  $("div").hide(1000, doSomething());
16              });
17          })
18      </script>
19  </head>
20  <body>
21
22      <button id="hide">Skjul</button>
23      <div>Her er noget tekst!!!</div>
24
25  </body>
26  </html>
```

Herunder er vist et eksempel på hvordan du kan lave en callback funktion i samme linje – altså definere en primær funktion (document.ready) med to yderligere funktioner inde i. Denne måde at programmere på – at passe en funktion som parameter til en anden funktion - er et ofte brugt trick i javascript sammenhæng. Byg og test begge eksempler i din browser.

```
HideCallback_02.html
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Hide Callback 02</title>
6      <script src="Scripts/jquery-3.2.1.min.js"></script>
7
8      <script type="text/javascript">
9          $(document).ready(function () {
10              $("#hide").click(function () {
11                  $("div").hide(1000, function () {
12                      alert("Dette er en callback funktion!");
13                  });
14              });
15          })
16      </script>
17  </head>
18  <body>
19
20      <button id="hide">Skjul</button>
21      <div>Her er noget tekst!!!</div>
22
23  </body>
24  </html>
```

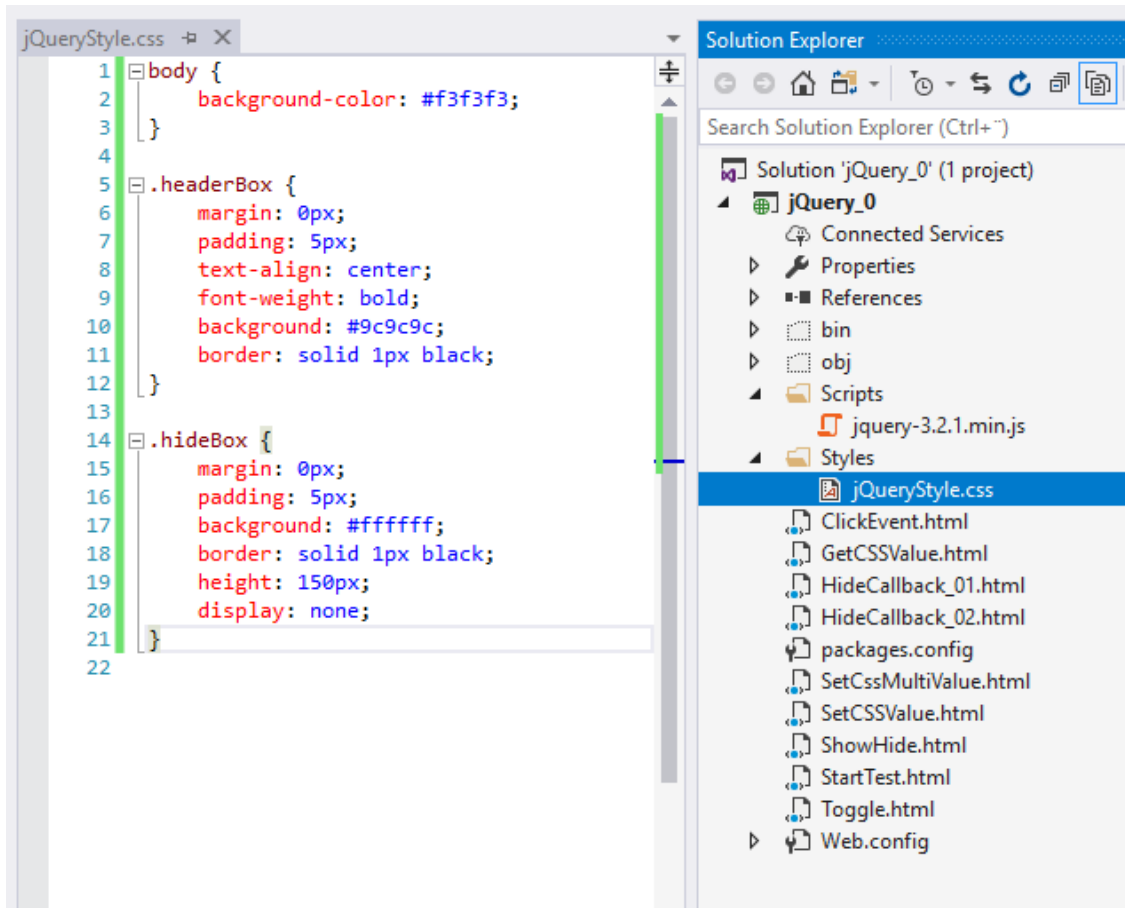
Vi har lige set på hvordan vi kan vise og skjule elementer. I stedet for – som i ShowHide.html at have to knapper, en til at vise og en til at skjule, så findes der en funktion (toggle) som kan bruges til at vise elementer hvis de er skjult og skjule dem hvis de er vist. Eksemplet her under viser hvordan du kan bruge jQuery's toggle funktion.

Toggle.html

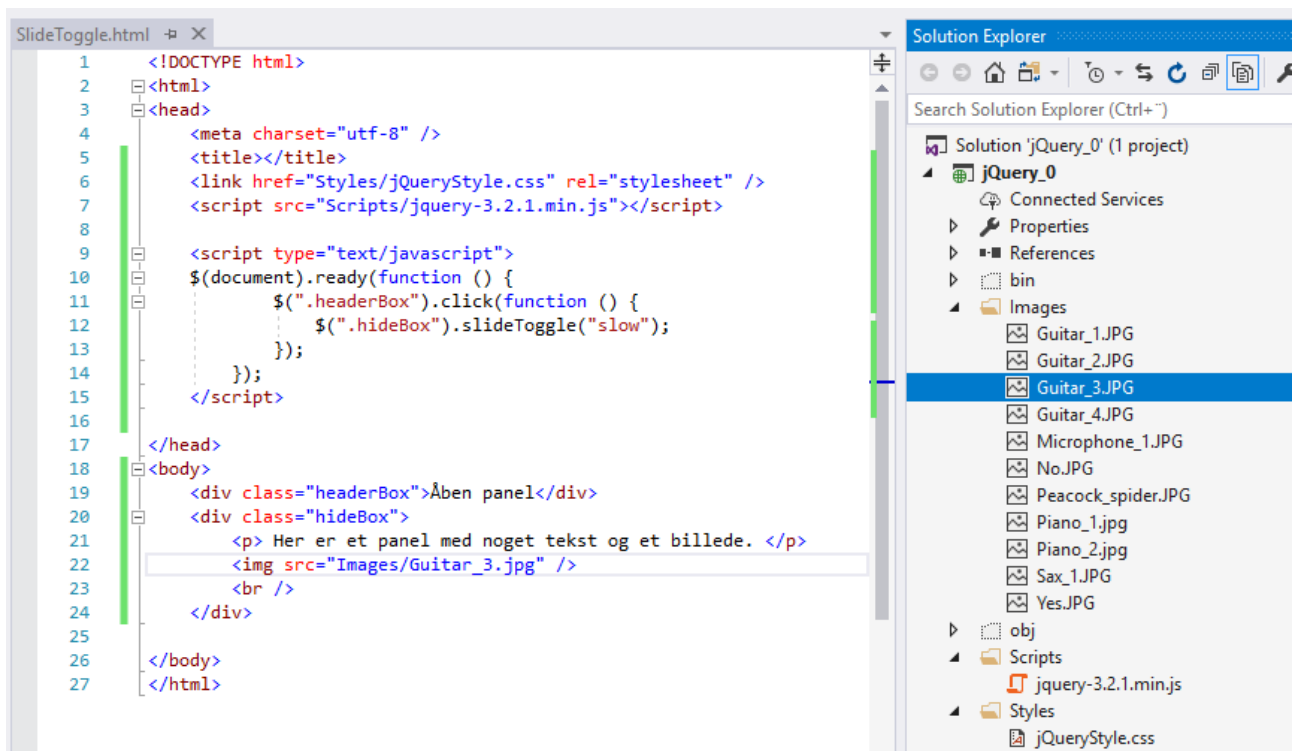
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8" />
5   <title></title>
6   <script src="Scripts/jquery-3.2.1.min.js"></script>
7
8   <script type="text/javascript">
9     $(document).ready(function () {
10       $("button").click(function () {
11         $("div").toggle(1000);
12       });
13     });
14   </script>
15 </head>
16 <body>
17
18   <button>Vis/Skjul</button>
19   <div>Her er noget tekst!!!</div>
20
21 </body>
22 </html>
```

Slide & Fade

Af andre interessante effekter kan nævnes slide og fade der på forskellige måde kan få objekter til at vises eller skjules med forskellige effekter. Inden vi starter skal vi lige have lavet et StyleSheet til de næste eksempler. Opret en ny mappe i dit projekt, 'Styles', og lav en ny css fil der, 'jQueryStyle.css':



I eksemplet her under bruger vi SlideToggle til at åbne og lukke et panel med noget tekst og et billede. For at få vist billedet, skal du en tur forbi mappen Materialer/Images (fra Materialer.zip på jeres fb side) – træk mappen Images over til Solution Explorer vinduet og tilføj på den måde mappen samt filerne i den til dit VS projekt.



Hvis du skulle få brug for kun at åbne eller lukke et panel findes der også to funktioner til det.

`$(selector).slideDown(speed,callback)`

`$(selector).slideUp(speed,callback)`

Næste eksempel minder meget om det forrige, her fader vi blot panelet frem i stedet for at slide det.

```
FadeIn.html  ➤ ✕
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Fade In Panel</title>
6      <link href="Styles/jqueryStyle.css" rel="stylesheet" />
7      <script src="Scripts/jquery-3.2.1.min.js"></script>
8
9      <script type="text/javascript"> $(document).ready(function () {
10         $(".headerBox").click(function () {
11             $(".hideBox").fadeIn(2000);
12         });
13     });
14 </script>
15 </head>
16 <body>
17     <div class="headerBox">Åben panel</div>
18     <div class="hideBox">
19         <p> Her er et panel med noget tekst og et billede. </p>
20         
21         <br />
22     </div>
23
24 </body>
25 </html>
```

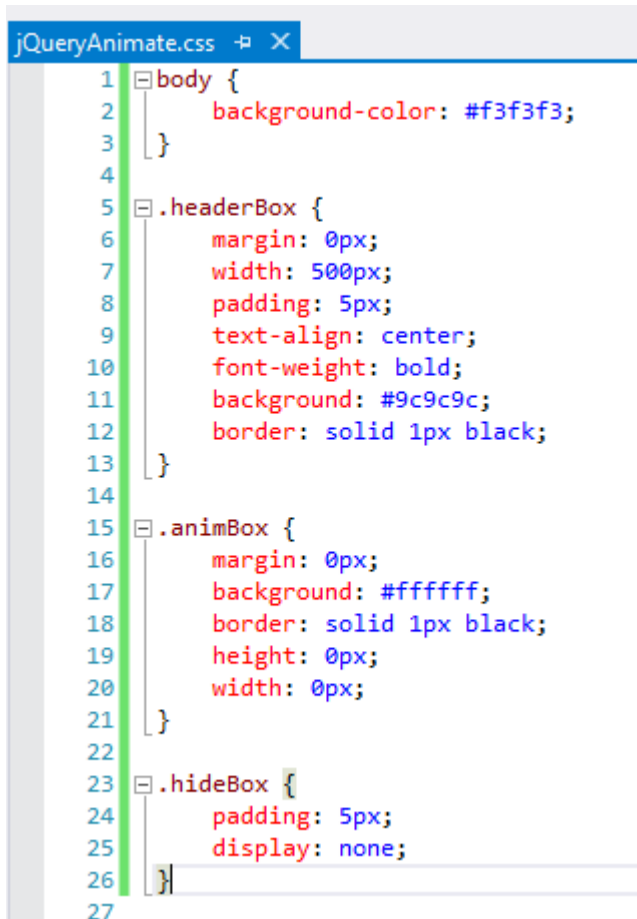
Under fade har vi også to andre funktioner fadeOut der er det modsatte af fadeIn funktionen og fadeTo som kan fade til og fra i procent.

\$(selector).fadeOut(speed,callback)

\$(selector).fadeTo(speed,opacity,callback)

Den sidste effekt vi skal se på er animate, den kan bruges til at animere attributter på elementer, så vi for eksempel kan animere højder, breder, tekststørrelser, farver mm. på objekter.

Start med at lave et nyt stylesheet, 'jQueryAnimate.css' i din Styles mappe:



```
1 body {  
2     background-color: #f3f3f3;  
3 }  
4  
5 .headerBox {  
6     margin: 0px;  
7     width: 500px;  
8     padding: 5px;  
9     text-align: center;  
10    font-weight: bold;  
11    background: #9c9c9c;  
12    border: solid 1px black;  
13 }  
14  
15 .animBox {  
16     margin: 0px;  
17     background: #ffffff;  
18     border: solid 1px black;  
19     height: 0px;  
20     width: 0px;  
21 }  
22  
23 .hideBox {  
24     padding: 5px;  
25     display: none;  
26 }  
27
```

Nu kan vi bygge en html side, der bruger stylesheetet og jQuery's animate() funktion:

```
Animate.html  X  jQueryAnimate.css
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6      <link href="Styles/jqueryAnimate.css" rel="stylesheet" />
7      <script src="Scripts/jquery-3.2.1.min.js"></script>
8
9      <script type="text/javascript">
10
11          $(document).ready(function () {
12              $(".headerBox").click(function () {
13                  $(".animBox").animate({ height: 310 }, "slow");
14                  $(".animBox").animate({ width: 510 }, "slow");
15                  $(".hideBox").fadeIn(2000);
16              });
17          });
18      </script>
19  </head>
20  <body>
21      <div class="headerBox"> Åben panel</div>
22      <div class="animBox">
23          <div class="hideBox">
24              <p> Her er et panel med noget tekst og et billede. </p>
25              
26              <br />
27          </div>
28      </div>
29
30  </body>
31  </html>
```

Byg eksemplet, og test funktionen i din browser. Prøv evt. at sætte hastigheden op eller ned på slider effekten i indholdsboxen (animBox elementet)

Understøttede effekter i jQuery

.animate()	Animere css-property.
.clearQueue()	Fjerner alle element der ikke er kørt fra køen.
.delay()	Kan lave en forsinkelse.
.dequeue()	Afvikler det næste item i køen.
.fadeIn()	Fader et element ind.
.fadeOut()	Fader et element ud.
.fadeTo()	Fader til en defineret gennemsigtighed.
.fadeToggle()	Kan fade bade ind og ud.
.hide()	Gemmer et element.
.queue()	Viser den kø af funktioner der er på et bestemt element.
.show()	Viser et skjult object.
.slideDown()	Slider ned, viser.
.slideToggle()	Slider både op og ned, viser og skjuler.
.slideUp()	Slider up, skjuler.
.stop()	Stopper en kørende animation
.toggle()	Viser og skjuler et element.

HTML Manipulation

Tilføjelse af tekst og/eller HTML i til eksisterende HTML-elementer kan gøres på flere forskellige måder. Herunder er de listet hver især.

\$(selector).append(content)

Tilføjer tekst før eksisterende indhold.

\$(selector).prepend(content)

Tilføjer tekst efter eksisterende indhold.

\$(selector).after(content)

Tilføjer tekst efter eksisterende indhold.

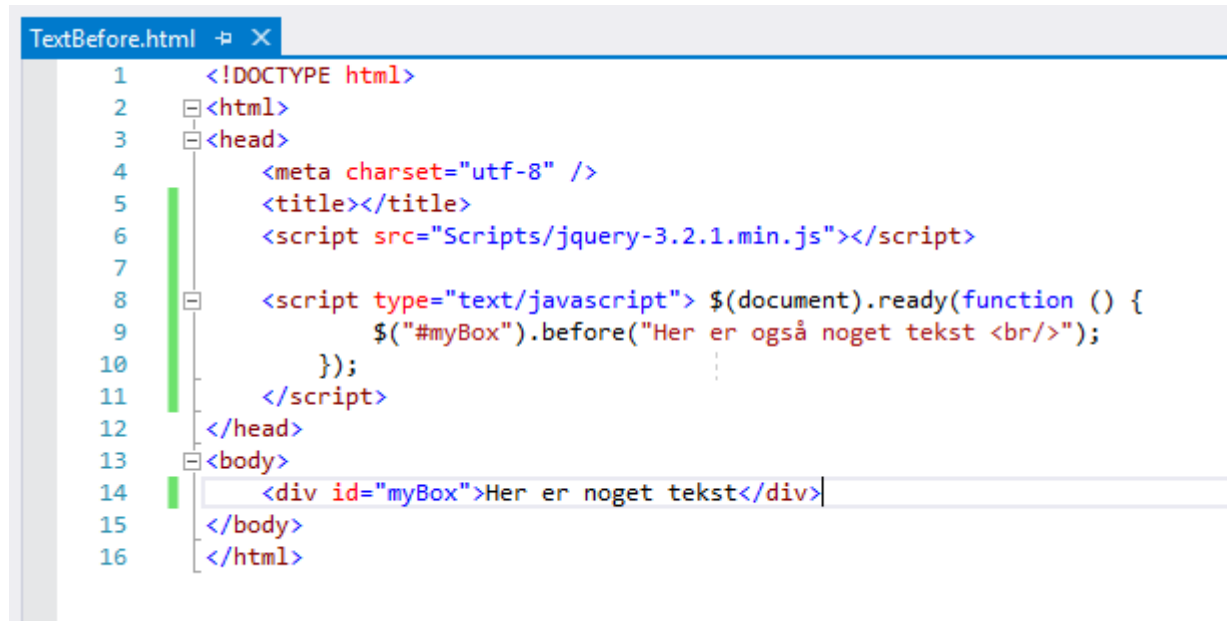
\$(selector).before(content)

Tilføjer tekst før eksisterende indhold.

`$(selector).html(content);`

Overskriver eksisterende indhold.

Eksemplet herunder viser hvordan vi tilføjer tekst til en box, med funktionen `before`.



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="utf-8" />
5     <title></title>
6     <script src="Scripts/jquery-3.2.1.min.js"></script>
7
8     <script type="text/javascript"> $(document).ready(function () {
9         $("#myBox").before("Her er også noget tekst <br/>");
10    });
11 </script>
12 </head>
13 <body>
14     <div id="myBox">Her er noget tekst</div>
15 </body>
16 </html>
```

Prøv at ændre ovenstående kode ved at skrive `'after'` i stedet for `'before'`. Har det en effekt på det. Du ser som output i browseren?

Attributter

Det er også muligt at manipulere med attributter i HTML-tags. Her under er vist et lille eksempel på hvordan du kan ændre en eller flere attributter på en gang.

```
1      <!DOCTYPE html>
2      <html>
3      <head>
4          <meta charset="utf-8" />
5          <title>Add Attributes</title>
6          <script src="Scripts/jquery-3.2.1.min.js"></script>
7          <script type="text/javascript">
8
9              $(document).ready(function () {
10                  $("button").click(function () {
11                      $("div").attr("width", "200");
12                      $("img").attr({
13                          src: "Images/Guitar_2.jpg",
14                          title: "En guitar",
15                          alt: "Billede af en guitar"
16                      });
17                  });
18              });
19          </script>
20
21      </head>
22      <body>
23
24          <button>Slå mig</button>
25          <br />
26          
27          <br />
28          <div>Her er lidt tekst.</div>
29
30      </body>
31  </html>
```

Det følgende eksempel her viser hvordan du kan læse værdien på en attribut.

```
ReadAttributes.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Read Attributes</title>
6      <script src="Scripts/jquery-3.2.1.min.js"></script>
7
8      <script type="text/javascript">
9          $(document).ready(function () {
10              $("button").click(function () {
11                  var srcAt = $("img").attr("src");
12                  alert("Stien til billedet er: " + srcAt);
13              });
14          });
15      </script>
16  </head>
17  <body>
18
19      
20      <br />
21      <br />
22      <button>Slå mig</button>
23
24  </body>
25  </html>
```

Det sidste eksempel her viser så hvordan du kan fjerne en attribut fra et HTML-tag. Prøv at køre eksemplet i din browser – når du klikker på knappen, og img src attributten fjernes, hvad tror du så der sker?

```
RemoveAttribute.html  ↗ ✕
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Remove Attribute</title>
6      <script src="Scripts/jquery-3.2.1.min.js"></script>
7
8      <script type="text/javascript">
9          $(document).ready(function () {
10              $("button").click(function () {
11                  $("img").removeAttr("src");
12              });
13          });
14      </script>
15
16  </head>
17  <body>
18
19      
20      <br />
21      <br />
22      <button>Slå mig</button>
23
24  </body>
25  </html>
```

AJAX

jQuery indeholder en lang række AJAX funktioner. AJAX er en forkortelse for Asynchronous JavaScript and XML og bruges til at opbygge hurtige dynamiske sider der blandt andet kan hente og sende data asynkront og kun opdatere dele af en side, i stedet for at loade det hele fra serveren igen – det, nogle populært kalder 'blink' effekten ved page refresh i browseren.

Her under er vist et simpelt eksempel på hvordan vi med jQuery metoden Load, kan hente tekst fra tekstfiler asynkront.

Start med at lave en ny mappe i dit projekt, 'TextFiles' som skal indeholde to tekstfiler med en enkelt linje tekst som vist her under.

TextFile1.txt

```
1 Fil nr. 1 indeholder denne tekst.
```

TextFile2.txt

```
1 Her er indholdet fra fil nr. 2.
```

Når tekstfilerne er lavet, opretter du et nyt HTML-dokument, kald det 'LoadText'.

```
LoadText.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6      <script src="Scripts/jquery-3.2.1.min.js"></script>
7
8      <script type="text/javascript">
9
10         $(document).ready(function () {
11             $("#button1").click(function () {
12                 $("#indhold").load('TextFiles/TextFile1.txt');
13             });
14             $("#button2").click(function () {
15                 $("#indhold").load('TextFiles/TextFile2.txt');
16             });
17         });
18     </script>
19
20 </head>
21 <body>
22
23     <h2>AJAX TEST</h2>
24     <div id="indhold">Vi starter med denne tekst.</div>
25     <br />
26     <br />
27     <button id="button1">Hent tekstfil 1</button>
28     <button id="button2">Hent tekstfil 2</button>
29
30 </body>
31 </html>
```

Byg eksemplet, og test det i din browser. Du skulle gerne kunne se teksterne skifte, uden at skærmen 'blinker' (full page refresh)

Plug-ins

Det er muligt at downloade en lang række plug-ins til jQuery, plug-ins er noget funktionalitet der ikke er i biblioteket i forvejen. På plugins.jquery.com kan du finde en lang række sjove og brugbare plug-ins udviklet af andre jQuery brugere og stillet gratis til rådighed.

For at bruge et plug-in skal du have fat i js-filen hvor plug-in'et er kodet i. Plugin-filer hedder typisk noget som eksemplet her "jQuery.snow.min.1.0.js" som er et plug-in til at lave sne på et website.

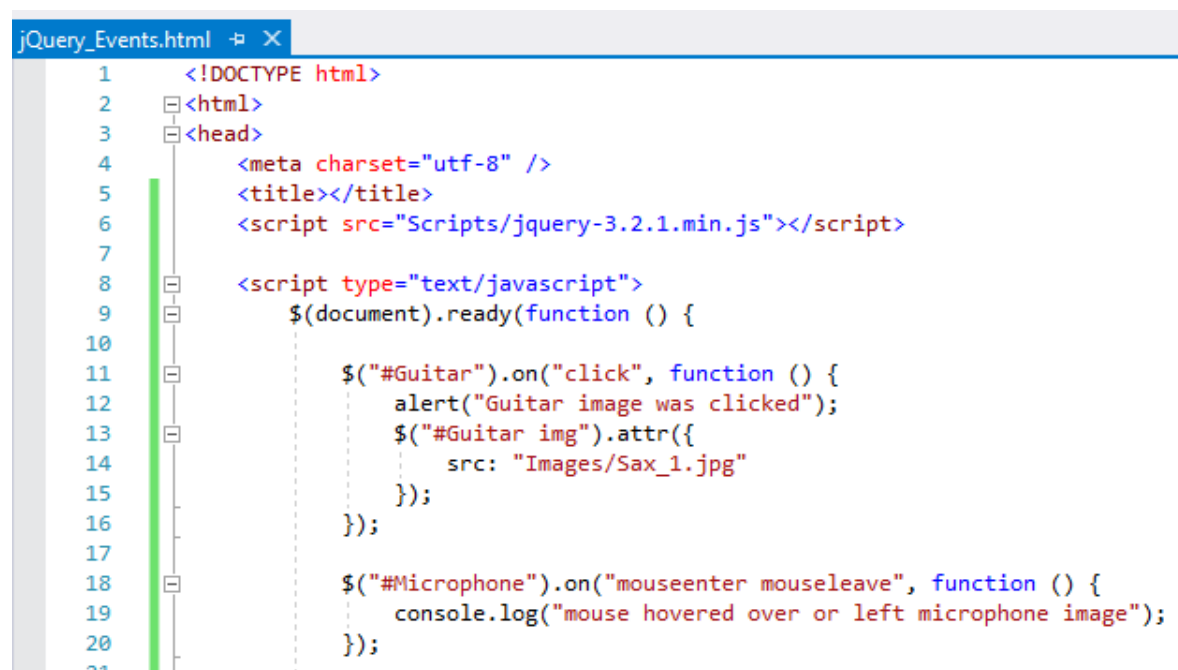
Plug-in filerne downloades og lægges i samme mappe som dine jQuery filer ('Scripts' mappen i dit projekt). Til det følgende eksempel har jeg i forvejen fundet et lille, kompakt 'sne-plugin' til jQuery; kodefilen ligger i mappen 'jQuery Plugins' under 'Materialer' mappen.

Byg eksemplet herunder, og kørs det for at se blå snefnug på en sort baggrund. Du kan evt. prøve at sætte indstillingerne for minSize, maxSize, newOn og flakeColor til andre værdier for at variere effekten, du ser på skærmen.

```
Snowfall.html  + X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Snevej</title>
6      <script src="Scripts/jquery-3.2.1.min.js"></script>
7      <script src="Scripts/jquery.snow.min.1.0.js"></script>
8
9      <script type="text/javascript">
10         $(document).ready(function () {
11             $.fn.snow({
12                 minSize: 5,
13                 maxSize: 50,
14                 newOn: 750,
15                 flakeColor: '#0099FF'
16             });
17         });
18     </script>
19 </head>
20 <body style="background-color: #000000;">
21
22 </body>
23 </html>
```

jQuery .on – event handler scripts

En af de ting, som jQuery med fordel kan bruges til er event handling på en webside. Herunder er der et eksempel, som illustrerer brugen af jQuerys .on handler til at håndtere flere events ad gangen for et givet element på siden:

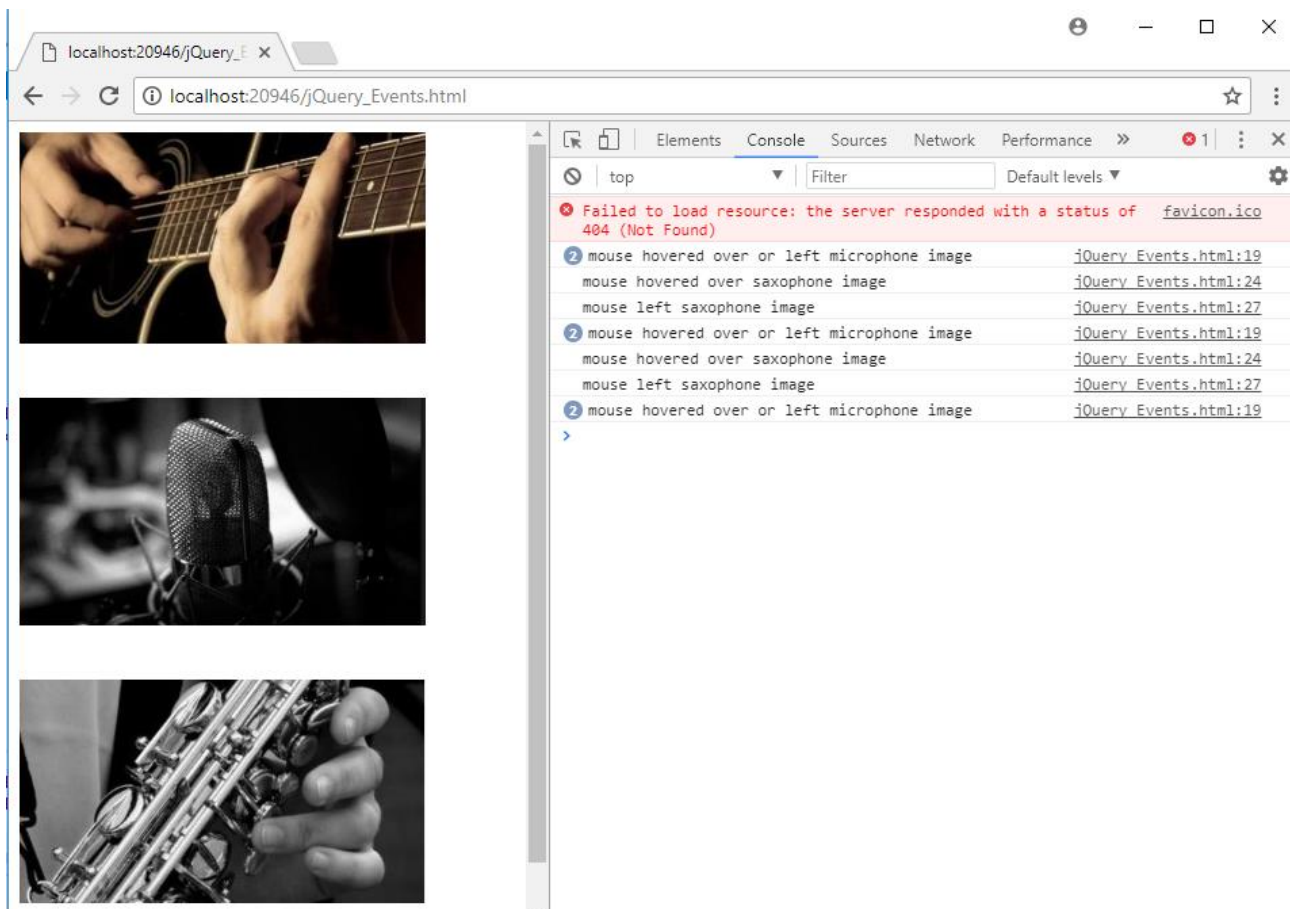


```
jQuery_Events.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title></title>
6      <script src="Scripts/jquery-3.2.1.min.js"></script>
7
8      <script type="text/javascript">
9          $(document).ready(function () {
10
11              $("#Guitar").on("click", function () {
12                  alert("Guitar image was clicked");
13                  $("#Guitar img").attr({
14                      src: "Images/Sax_1.jpg"
15                  });
16              });
17
18              $("#Microphone").on("mouseenter mouseleave", function () {
19                  console.log("mouse hovered over or left microphone image");
20              });
21          });
22      </script>
23  </head>
24  <body>
25  </body>
26  </html>
```

..og samme fil, fortsat:

```
jQuery_Events.html  X
22      $("#Sax").on({
23          mouseenter: function () {
24              console.log("mouse hovered over saxophone image");
25          },
26          mouseleave: function () {
27              console.log("mouse left saxophone image");
28          },
29          click: function () {
30              alert("User clicked on saxophone image");
31          }
32      });
33
34  });
35  </script>
36  </head>
37  <body>
38      <div id="Guitar">
39          
40      </div>
41      <br />
42      <br />
43      <div id="Microphone">
44          
45      </div>
46      <br />
47      <br />
48      <div id="Sax">
49          
50      </div>
51      <br />
52      <br />
53  </body>
54  </html>
```

Byg eksemplet ovenfor, og kørs det i din browser. For at du kan se de beskeder, som scriptet sender til javascript debuggerens konsol med console.log skal du åbne din browsers js debugger / js toolbox, dette sker med et tastetryk på F12 i de fleste browsere. Klik på 'console' tab'en for at se beskeder og eventuelle javascript runtime fejl på siden – her er Chrome vist, men andre browsere har naturligvis tilsvarende værktøjer:



Klik på guitar billedet øverst, og se det skifte til det nederste (sax) billede. Nu vil du måske forvente en ændring i sidens opførsel.. men sker der nogen ændring i de beskeder, du nu får ud i konsollen? Hvorfor tror du, at det stadig kun er det nederste billede, der giver lyd fra sig i konsollen?

Udfordring: lav en funktion, der både skifter id på div tagget, og sætter en ny onclick event handler op, med en ny alert message for det nu ændrede id.

Diskussion: hvordan bliver jeg af med den første alert på guitar billedet? Hint: jQuery .off gør det modsatte af .on.

Custom events med jQuery .on

I det næste eksempel vil vi konstruere en custom event handler – dvs. en event handler som du selv definerer, der reagerer på andre standard events i browseren som fx et museklik.

```
jQuery_Custom_Event.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <title>jQuery .on demo - custom event</title>
6      <script src="Scripts/jquery-3.2.1.min.js"></script>
7
8      <style>
9          p {
10             color: red;
11         }
12
13         span {
14             color: blue;
15         }
16     </style>
17
18 </head>
19
```

.. og samme fil, fortsat:

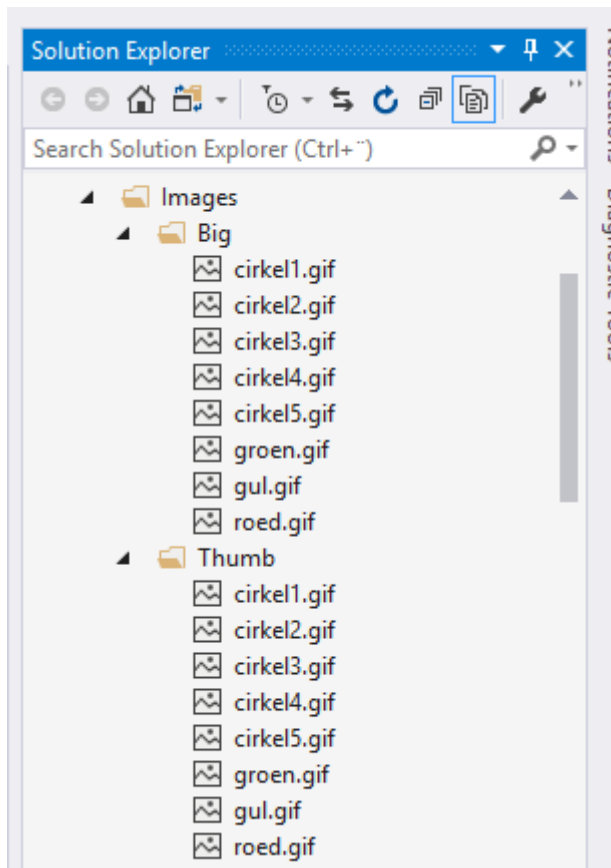
```
jQuery_Custom_Event.html  X
19
20 <body>
21
22 <p>Siden har en custom event.</p>
23 <button>Trigger for custom event</button>
24 <span style="display:none;"></span>
25
26 <script>
27     $("p").on("myCustomEvent", function (event, myName) {
28         $(this).text(myName + ", halli hallo!");
29         $("span")
30             .stop()
31             .css("opacity", 1)
32             .text("myName = " + myName)
33             .fadeIn(30)
34             .fadeOut(1000);
35     });
36     $("button").click(function () {
37         $("p").trigger("myCustomEvent", ["Spradebasse 1"]);
38     });
39 </script>
40
41 </body>
42 </html>
```

Byg eksemplet, og kør det i din browser. Virker fade animationen på elementet? Kan du ændre nogen af de viste parametre, sådan at det går hurtigere eller langsommere med at fade myName ind og ud?

jQuery galleri

Her under er en udgave af galleriet vi lavede i JavaScript blot tilføjet lidt jQuery effekter og funktionalitet. Eksemplet består af flere elementer blandt andet en mappe med billeder "Images" et StyleSheet, en Script-fil og sidst men ikke mindst HTML-dokumentet hvor vi binder det hele sammen.

Start med at hente de billedfiler, vi skal bruge ind i dit projekt. I mappen 'GalleryImages' under Materialer ligger der to undermapper, 'Big' og 'Thumb'. Træk begge mapper over i den 'Images' folder, du før har oprettet i dit VS projekt:



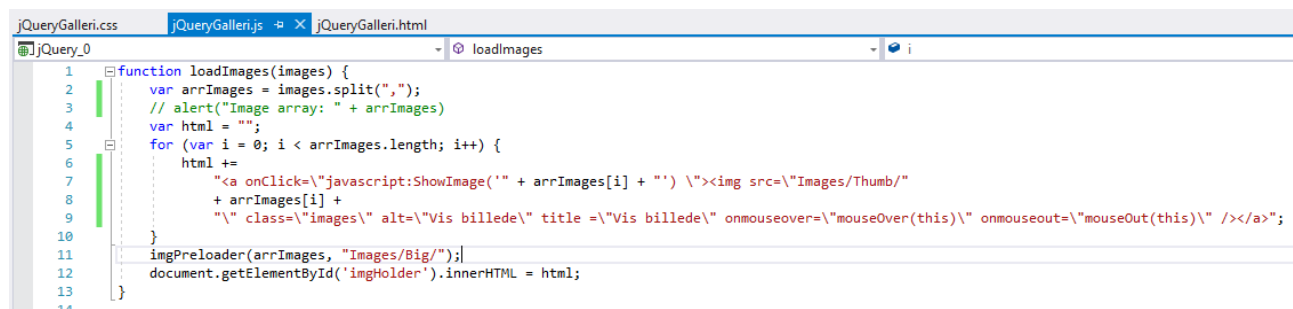
Nu kan du oprette en ny css fil i mappen 'Styles' – kald den 'jQueryGalleri.css':

```
jQueryGalleri.css  X jQueryGalleri.js  X jQueryGalleri.html
1  .images {
2      float: left;
3      height: 105px;
4      width: 140px;
5      margin: 5px;
6      border: 1px solid black;
7      cursor: pointer;
8  }
9
10 #wrapper {
11     position: absolute;
12     z-index: 2;
13     width: 90%;
14     margin-top: 50px;
15 }
16
17 #imgBox {
18     margin-left: auto;
19     margin-right: auto;
20     width: 760px;
21     background-color: Black;
22 }
23
24 #bigImg {
25     margin: 20px;
26     border: 0px;
27     cursor: pointer;
28 }
```

.. og samme fil, fortsat:

```
jQueryGalleri.css  X jQueryGalleri.js  X jQueryGalleri.html
29
30 .thumpPop {
31     float: left;
32     height: 115px;
33     width: 150px;
34     margin: 0px;
35     border: 1px solid black;
36     cursor: pointer;
37 }
38
```

Nu er det tid at oprette script filen 'jQueryGalleri.js' i din Scripts mappe:



```
1 function loadImages(images) {
2     var arrImages = images.split(",");
3     // alert("Image array: " + arrImages)
4     var html = "";
5     for (var i = 0; i < arrImages.length; i++) {
6         html +=
7             "<a onClick='\"javascript:ShowImage(\" + arrImages[i] + \"') '\"><img src=\"Images/Thumb/\"
8             + arrImages[i] +
9             "\" class=\"images\" alt=\"Vis billede\" title =\"Vis billede\" onmouseover=\"mouseOver(this)\" onmouseout=\"mouseOut(this)\" /></a>";
10    }
11    imgPreloader(arrImages, "Images/Big/");
12    document.getElementById('imgHolder').innerHTML = html;
13 }
```

De første 15 linier i scriptet her er de værste at skrive, fordi der skal bygges en meget lang html string op i linierne 6 til 10. Derfor får du lidt hjælp her, så du kan kopiere nedenstående for loop ind i dit script – husk at tage alle strings som denne over i notepad inden du kopierer dem ind i din editor:

```
for (var i = 0; i < arrImages.length; i++) {
html += "<a onClick='\"javascript:ShowImage(\" + arrImages[i] + \"') '\"><img
src=\"Images/Thumb/\" + arrImages[i] + "\" class=\"images\" alt=\"Vis billede\" title =\"Vis
billede\" onmouseover=\"mouseOver(this)\" onmouseout=\"mouseOut(this)\" /></a>";
}
```

Vi fortsætter med at bygge js filen:

```

jQueryGalleri.css  jQueryGalleri.js  jQueryGalleri.html
jQuery_0  mouseOut
15  function ShowImage(image) {
16      $("#imgBox").fadeTo(700, 0, function () {
17          if (image == 'none') {
18              $("#myImg").html('');
19          } else {
20              $("#myImg").html('');
23          }
24          $("#imgBox").fadeTo(700, 1, function () { });
25      });
26  }
27
28  function imgPreloader(arrImages, folder) {
29      var preImg = new Image();
30      for (var i = 0; i < arrImages.length; i++) {
31          preImg.src = folder + arrImages[i];
32      }
33  }
34
35  function mouseOver(object) {
36      object.className = "thumpPop";
37  }
38
39  function mouseOut(object) {
40      object.className = "images";
41  }

```

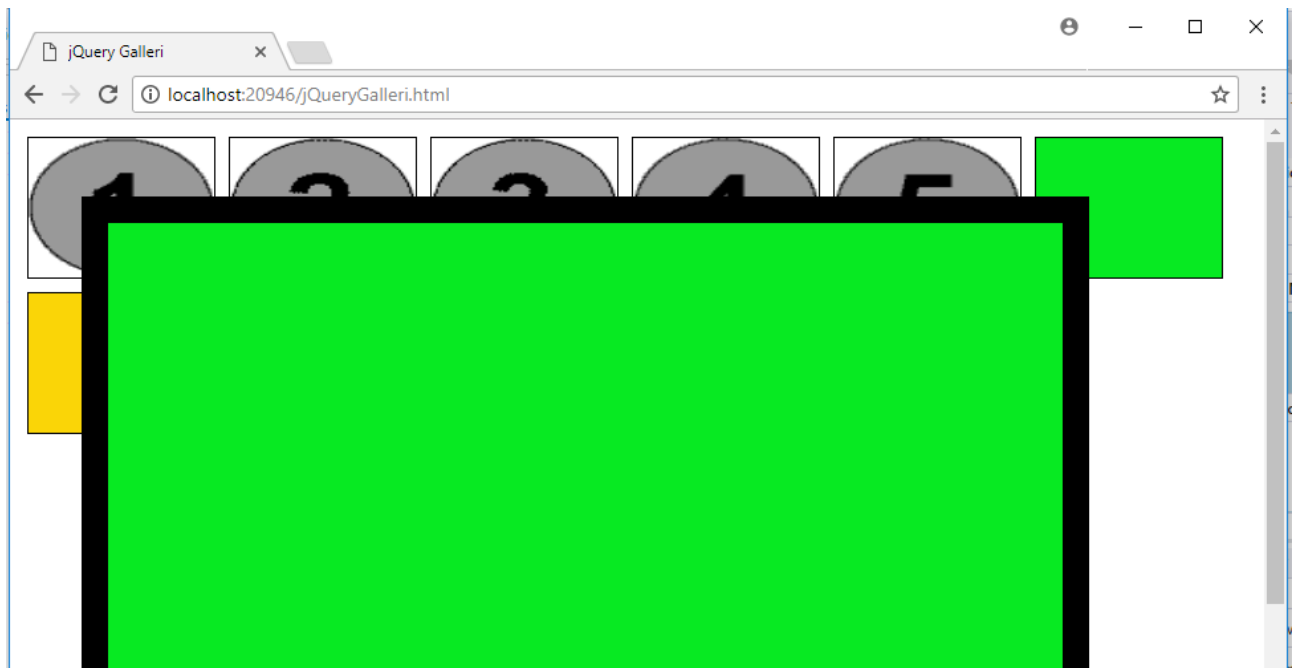
.. og tilsidst HTML filen:

```

jQueryGalleri.css  jQueryGalleri.js  jQueryGalleri.html
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>jQuery Galleri</title>
6      <link href="Styles/jqueryGalleri.css" rel="stylesheet" />
7      <script src="Scripts/jquery-3.2.1.min.js"></script>
8      <script src="Scripts/jqueryGalleri.js"></script>
9  </head>
10 <body>
11
12     <div>
13         <div id="imgHolder">
14             <div>
15
16             <div id="wrapper">
17                 <div id="imgBox">
18                     <a onclick="javascript:ShowImage('none');">
19                         <div id="myImg">
20
21                         </div>
22                     </a>
23                 </div>
24             </div>
25             <script type="text/javascript">
26                 loadImages("cirkel1.gif,cirkel2.gif,cirkel3.gif,cirkel4.gif,cirkel5.gif,groen.gif,gul.gif,roed.gif");
27             </script>
28         </div>
29     </body>
30 </html>

```

Byg eksemplet, og test det i din browser. Der skulle gerne komme et større billede frem i forgrunden, når du klikker på et af de små billeder; det store billede kan du fjerne igen ved at klikke på det:



Scroll navigation

Et kort eksempel på brug af jQuerys animate function – ScrollNav.html.

Start med at lave en css fil, placeret i dit 'Styles' bibliotek:

```
jQueryScrollNav.css  X ScrollNav.html
1  body {
2      margin-top: 0px;
3  }
4
5  .content {
6      height: 400px;
7      width: 400px;
8      background-color: Gray;
9      margin: 20px;
10     padding-top: 40px;
11 }
12
13 #menu {
14     position: fixed;
15     top: 10px;
16     left: 10px;
17     height: 40px;
18     width: 100%;
19     background-color: black;
20     margin: 0px;
21 }
22
```

Nu kan vi sætte HTML filen op:

```
ScrollNav.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>Scroll Navigation</title>
6      <link href="Styles/jqueryScrollNav.css" rel="stylesheet" />
7      <script src="Scripts/jquery-3.2.1.min.js"></script>
8
9      <script type="text/javascript">
10         function ScrollTo(id) {
11             $('html,body').animate({
12                 scrollTop: $("#" + id).offset().top
13             }, 'slow');
14         }
15     </script>
16 </head>
17
```

.. og samme HTML fil fortsat:

```
ScrollNav.html - X
18 <body>
19
20 <div id="menu">
21 <a href="javascript:void(0)" onclick="ScrollTo('home')">Home</a> |
22 <a href="javascript:void(0)" onclick="ScrollTo('side1')">Side1</a> |
23 <a href="javascript:void(0)" onclick="ScrollTo('side2')"> Side2</a> |
24 <a href="javascript:void(0)" onclick="ScrollTo('side3')">Side3</a>
25 </div>
26
27 <div id="home" class="content">
28 <h1>
29 Home
30 </h1>
31 </div>
32 <div id="side1" class="content">
33 <h1>
34 Side 1
35 </h1>
36 </div>
37 <div id="side2" class="content">
38 <h1>
39 Side 2
40 </h1>
41 </div>
42 <div id="side3" class="content">
43 <h1>
44 Side 3
45 </h1>
46 </div>
47
48 </body>
49 </html>
```

Byg eksemplet, og test scroll navigationen i din browser.

Fade navigation

Det sidste eksempel i kompendiet bygger på jQuery's 'fade' animate funktionalitet; her er det teksten underde enkelte menupunkter / indholdssider, der fades ind:

```

FadeNav.html  X
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8" />
5      <title>jQuery Fade Navbar</title>
6      <script src="Scripts/jquery-3.2.1.min.js"></script>
7      <script type="text/javascript">
8
9          function skiftBox(name) {
10             $("#content_text").fadeTo(500, 0, function () {
11                 $("#content_text").html($("#" + name).html());
12                 $("#content_text").fadeTo(500, 1, function () {
13                     // no jedi things to see here, move on inspector Clouseau
14                 });
15             });
16         }
17
18         $(document).ready(function () {
19             skiftBox('home');
20         });
21     </script>
22 </head>

```

.. og samme fil, fortsat:

```

FadeNav.html  X
22 </head>
23
24 <body>
25     <a href="#" onclick="skiftBox('home');">Home</a> |
26     <a href="#" onclick="skiftBox('box1');">Vis Box 1</a> |
27     <a href="#" onclick="skiftBox('box2');">Vis Box 2</a> |
28     <a href="#" onclick="skiftBox('box3');">Vis Box 3</a>
29     <br />
30
31     <div id="content_text">Bwhahahah!
32     </div>
33     <div id="home" style="display: none;">
34         HER ER FORSIDEN!!!
35     </div>
36     <div id="box1" style="display: none;">
37         Test1
38     </div>
39     <div id="box2" style="display: none;">
40         Test2
41     </div>
42     <div id="box3" style="display: none;">
43         Test3
44     </div>
45
46 </body>
47 </html>

```