

GitHub og Brackets

Versionsstyring fra Brackets

Git - et recap

Hvad er versionsstyring?

Hvorfor lave versionsstyring?

Commit messages - hvorfor skal jeg ikke bare skrive noget sludder?

GitHub er også dokumentation!

Git og GitHub - terminologi

- Repo
- Branch
- Clone
- Commit
- Push
- Pull
- Fork

Github og Brackets workflow

1. Installer Git (du har måske allerede gjort det på 1. sem.)
2. Opret et nyt Repository på gitHub
3. Opret ny lokal mappe
4. Clone Repo via Brackets
5. Opret og rediger filer
6. Commit
7. Push

OBS!

For at undgå konflikter er det **MEGET** vigtigt at den lokale mappe er **helt** tom når der klones!

Litteratur til Github

Rigtig god manual og tutorials:

<https://practicalseries.com/1002-vcs/index.html>

Github og Brackets video (made for you by me):

<https://youtu.be/UKF0VhwhfYuY>

Øvelse 0 - opret en github-mappe til dagens øvelser

1. Opret et **tema7** repository på github
2. Lav en mappe, **tema7**, på din computer (måske under en **2-semester**-mappe, som ligger i en **KEA**-mappe?).
3. I Brackets - åbn **tema7** mappen: Klon **tema7**-repository'et til tema7 mappen.
4. Lav i **tema7** en undermappe, **undervisningsopgaver**.
5. Lav i undervisningsopgaver en mappe, **01-js-basics** til dagens øvelser.
6. Åbn mappen **01-js-basics** i Brackets
7. Læg et genbrugeligt, tomt html-skelet, **00-skelet.html** ind i mappen
8. Udfyld dette [regneark](#)

tema7
undervisningsopgaver
01-js-basics
00-skelet.html

Javascript basics

repetition

Dagens program:

1. Om programmeringssprog og javascript generelt
2. Erklæringer af variabler og konstanter
3. Indbyggede metoder til tekst og tal
4. If-statement (betingelser)
5. Metoder til test uden DOM manipulation (console.log)
6. Funktioner
7. Øvelser (mange!)

Programmeringssprog?

- **Computersprog:** forskellige sprog, som bruges på computeren, herunder kommandoer, programmeringssprog, machine code, markup sprog mfl.
- **Programmeringssprog:** Computersprog som bruges til at få computeren til at udføre opgaver. Indeholder anvisninger/kommandoer til computeren.

[Wikipedia: forskellige typer computersprog](#)

[Mest anvendte programmeringssprog](#)

[Wikipedia, definition af programmeringsprog](#)

Programmeringssprog og andre computersprog

html er **ikke** et programmeringssprog

det er et **opmærkningssprog**

hvilke elementer har vi på en webside (**DOM**'en)

css er **ikke** et programmeringssprog

det er et **layoutsprog**

hvordan skal elementerne på en webside præsenteres.

javascript er et programmeringssprog!

besked til computeren i et program - hvad skal der ske.

https://en.wikipedia.org/wiki/Computer_language

Programmeringssprog

- Et programmeringssprog er et **kunstigt** sprog
- I forhold til naturlige/menneskesprog er de **simple** og meget **formelle**
- Et computersprog er beskrevet i form af nogle **regler**
- **Syntax**: hvilke regler har vi for hvordan ordene kan sættes sammen?
- **Semantik**: Hvilken betydning har sætningerne?
- Regler for syntax og semantik beskrives i referencemanualer

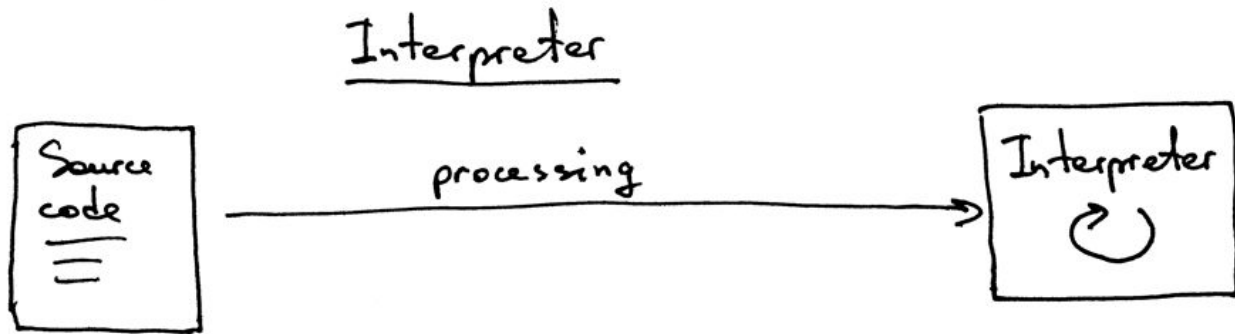
Spørgsmål på Stackoverflow: [What is the difference between syntax and semantics of programming languages:](https://stackoverflow.com/questions/17930267/what-is-the-difference-between-syntax-and-semantics-of-programming-languages)

<https://stackoverflow.com/questions/17930267/what-is-the-difference-between-syntax-and-semantics-of-programming-languages>

Fortolkede sprog

- Programmet skrives og gives til en **fortolker/interpreter**.
- Fortolkeren oversætter sætningerne en ad gangen til maskinsprog
- Computeren udfører sætningerne efterhånden, som den får dem.

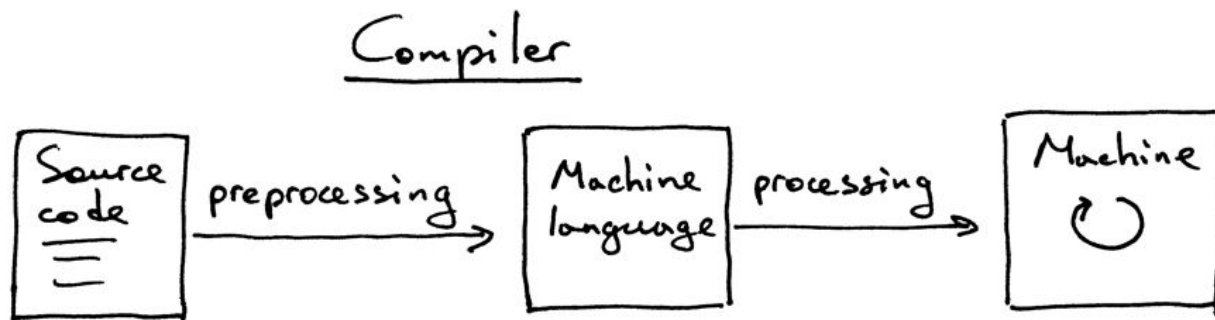
Eksempler: javascript, php, python, ruby, perl



Oversatte sprog

- Programmes skrives og gives til en **oversætter/compiler**.
- Compileren oversætter det hele til maskinsprog
- Computeren udfører det oversatte program.

Eksempler: Java, C#, Swift



Script-sprog

- Script-sprog bruges ofte blot som **synonym for fortolkede sprog**.
- Men kan også betyde et programmeringssprog til småprogrammer (scripts), som fungerer sammen med et andet computersprog.

F.eks: **javascript**, som fungerer sammen med html og css i en browser

Javascript er et

- **Programmeringssprog**
- **Fortolket sprog**
- **Scriptsprog**
- **Objektorienteret sprog**
 - Alle browsere har indbygget en javascript-fortolker/engine
 - javascript kan manipulere web-dokumentets html-elementer (DOM)
 - javascript kan også manipulere html-elementernes layout (CSS'en)
 - Et standardiserings-organisation, **ECMA** tager sig af definere javascript

Sidste version: ECMA-script6 (https://www.w3schools.com/js/js_es6.asp)

Wikipedia [om Javascript: https://en.wikipedia.org/wiki/JavaScript](https://en.wikipedia.org/wiki/JavaScript)

javascript - et objektorienteret sprog

- Javascript har mange **indbyggede objekter**
- Desuden er det muligt at **skabe objekter** i javascript
- Et eksempel på et indbygget objekt i javascript er **Math**
- **Math** er et indbygget objekt, som har en lang række **egenskaber** og **metoder**, f.eks:
 - **Math.PI** (egenskab)
 - **Math.random()** (metode - bemærk parentesen)
- Punktum . mellem objektnavn og egenskab/metode kaldes **dot-notation**

Opsamling: Begreber, hvad var det nu, det betød...?

Computersprog

DOM

Opmærkningssprog

Layoutsprog

Programmeringssprog

Compilet sprog

Fortolket sprog

Script-sprog

Javascriptfortolker (hvor?)

objektorienteret sprog

indbyggede objekter

dot-notation

Erklæringer af variable i javascript

Variabel-erklæringer

- En **variabel** er et **navn**, som man kan tildele en **værdi**.
- En variabels værdi kan senere i programmet ændres til noget andet.
- (globale) variabler **erklæres** ofte øverst i scriptet.

Eksempler:

```
let forNavn = "Martin";  
let alder = 58;  
let enlig = false;  
let køn; // erklæring uden værditildeling
```

Best practice:
camel-case til variabelnavne

forNavn, alder og enlig har fået **tildelt værdier** af tre forskellige **datatyper**: tekst, tal og boolean.

Konstanter

- Tit har man brug for navne (variabler) med værdier, som **ikke** skal ændres senere i programmet.
- Her bruger man **konstanter** (det er dog teknisk set stadig en variabel)
- Erklæring af en konstant:
 - **const MOMS = 0.25;** // momsen er konstant - skal ikke ændres i programmet.
- Konstanter værdi kan ikke ændres, men bruges ellers ligesom andre variabler.

Opsamling: Ord og begreber - hvad betød de?

- variabel erklæring
- camelcase
- konstant
- tildeling af værdi
- datatyper:
 - tekst/string
 - tal
 - boolean

Operatorer og window-metoder

Window-objektet

- Browserens vigtigste indbyggede objekt er **window-objektet**
- window er **browserens vindue** - med eller uden html-elementer (DOM)
- window-objektet har en lang række **egenskaber/properties** og metoder, som kan tilgås fra javascript.

F.eks.

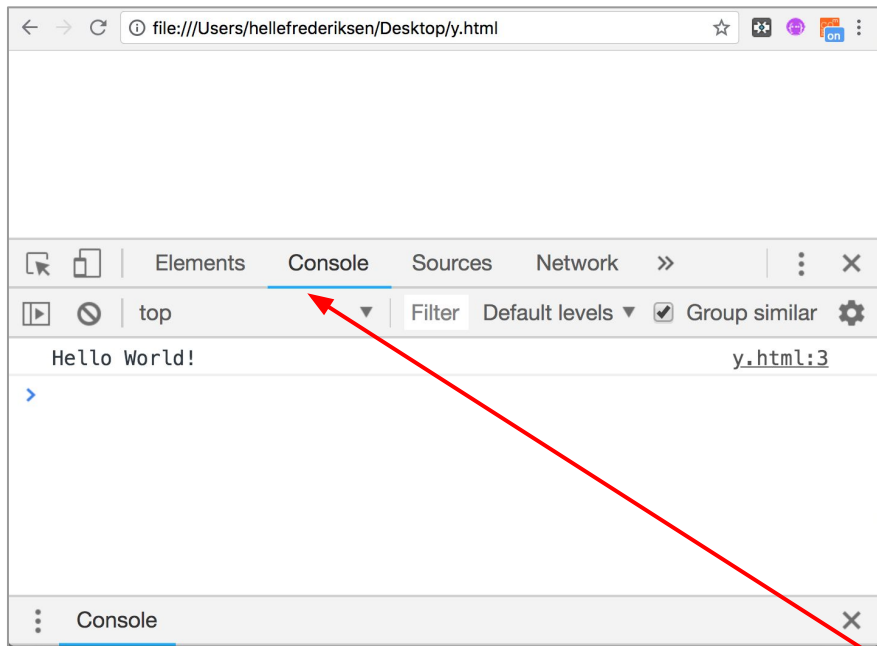
- window.**console.log()** - skriver noget i browser-consollen.
- window.**alert()** - åbner en dialogboks i browservinduet med en tekst.
- window.**prompt()** - åbner en dialogboks i browservinduet med et input felt.
- window.**confirm()** - åbner en dialogboks i browservinduet med OK/Annuller knapper.

BEMÆRK window er underforstået, og udelades som regel.

VI BENYTTER KUN console.log()

ref: https://www.w3schools.com/jsref/obj_window.asp

window.console.log()



Indbygget metode, som kan udskrive en værdi i browserens console-vindue

Eksempel:

```
let greeting = "Hello World";  
console.log(greeting);
```

I Chrome (og de fleste andre browsere):

Højreklik i browservinduet -

Vælg inspect / undersøg

Vælg fanebladet Console

Sammenlægning af tekster

Efter erklæringen, kan variablerne ændres i **statements**

```
let minTekst = "Her er en tekst";  
minTekst = minTekst + " , som fortsætter her!";
```

+ er en operator til sammenlægning af tekster, **text concatenation**

linje 2 kan også skrives som:

```
minTekst += " , som fortsætter her!";
```

+= er også en operator til **concatenation**.

Tager den værdi, variabelen har i forvejen, og tilføjer højresiden.

Template literals

Når man arbejder med tekster, kan man i stedet for tekst-concatenation med `+`-operatoren bruge en ny syntax/mulighed i js kaldet Template literals

```
let minTekst = "her er en tekst";  
minTekst = `${minTekst}, som fortsætter her`;
```

Udenom hele teksten bruge ``...`` (accent grave)

I teksten, kan man indsætte variabler eller udtryk.

Man bruger `$` foran og `{...}` omkring variabler eller udtryk.

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template_literals

Indbyggede metoder til tekster

I javascript kan man meget andet end blot at sammenlægge tekster.

Tekst-variabler har en række **metoder**, f.eks. (søg efter flere, når du får brug for det):

- `let tekst = "Eksempel på en tekst";` (eller: `let tekst = `Eksempel på en tekst`;`)
- `let len = tekst.length;` // tekstens længde (her 20)
- `tekst = tekst.toUpperCase();` // tekst er nu “EKSEMPEL PÅ EN TEKST”
- `tekst = tekst.toLowerCase();` // tekst er nu “eksempel på en tekst”

JavaScript Stings, W3schools: https://www.w3schools.com/jsref/jsref_obj_string.asp

Statements til beregning af tal

I statements kan man udregne tal.

Eksempel 1:

```
let pris = 100;  
const moms = 0.25;  
pris = pris + pris*moms;
```

Tal-operatorer

+ Addition

eks: **pris = indkobsPris + moms;**

- Subtraction

eks: **indkobsPris = pris - moms;**

*** Multiplication**

eks: **totalPris = antal * pris;**

/ Division

eks: **pris = totalPris/antal;**

++ Increment (læg 1 til)

eks: **antal++;**

-- Decrement (træk 1 fra)

eks: **antal--;**

Indbyggede metoder til beregninger

Math, som har en række nyttige metoder til tal:

```
let tal = 3.5;
```

```
tal = Math.round(tal); // tal er nu afrundet til (4)
```

```
tal = Math.pow(tal,2); // tal opløftes til 2. potens (16)
```

```
tal = Math.random(); // tal er et tilfældigt tal mellem 0 og 1 (fx. 0.843219827740112)
```

```
tal = Math.round(Math.random()*10); // tal er nu et heltal mellem 0 og 10
```

```
erTal = isNaN(tal); // er sand, hvis tal ikke er et tal - her er den false (NaN - Not a Number)
```

Math, W3schools: https://www.w3schools.com/js/js_math.asp

Øvelse 01 - udregn areal

Åbn [00-skelet.html](#), og gem den som en ny fil, [01-areal.html](#) i [01-js-basics](#)

Skriv i script-tagget et program, som kan udregne areal ud fra en længde og en bredde.

[Længde](#), [bredde](#) og [areal](#) skal erklæres som variabler i programmet.

Resultatet skal vises i console-vinduet, og have denne form:

[Længden er 3 meter og bredden er 5 meter. Arealet er længde*bredde](#)

Test programmet med forskellige værdier for længde og bredde

Commit og push til gitHub, når du er tilfreds med opgaven

tema7
undervisningsopgaver
01-js-basics:
[00-skelet.html](#)
[01-areal.html](#)

Opsamling - hvad var det nu, det betød?

window-objektet

window.console.log()

string

concatenation

Template literals

Math.round(tal)

Math.pow(n, m)

Math.random(tal)

tekst.length

tekst.toUpperCase()

tekst.toLowerCase()

isNaN(tal)

Operatorer

tekster: + og +=

tal: +, -, /, *

Betingelse / condition
if-statement

if-statement

Hvis et eller flere statements KUN skal udføres, hvis en betingelse er opfyldt:

```
if(betingelse/condition){  
    statement1;  
    statement2;  
}
```

Hvor betingelsen er en boolsk værdi (sand eller falsk) eller et boolsk udtryk:

```
if(alder < 18) {  
    console.log("Barn");  
};
```

< er en **logisk operator**

alder < 18 er en **betingelse/condition**

if else else if: w3school: https://www.w3schools.com/js/js_if_else.asp

Logiske operatorer

== Er lig med

!= Er forskellig fra

> Er større end

< Er mindre end

>= Er større end eller lig med

&& Og (mellem to betingelser)

|| Eller (mellem to betingelser) alt+i på mac-keyboard

! Ikke / negation (foran en betingelse)

if-statement - flere eksempler

```
if(alder >=18) {  
    console.log("Voksen");  
}
```

```
if(sex == "female"){  
    console.log("Kvinde");  
}
```

```
if(sex != "female"){  
    console.log("Mand");  
}
```

```
if(sex != "female" && alder < 18){  
    console.log("Dreng");  
}
```

```
if(alder < 18 || alder > 67){  
    console.log("Ikke erhvervsaktiv");  
}
```

if-else-statement

Et eller flere statements skal KUN udføres, hvis en betingelse er opfyldt, ELLERS skal nogle andre udføres:

```
if(betingelse){  
    statement1;  
else {  
    statement2;  
}
```

Eksempel:

```
if(alder < 18){  
    console.log("Barn");  
} else {  
    console.log("Voksen");  
}
```

Forgrenede betingelser (if-statements)

Man kan også konstruere en if-sætning med mange else-grene:

```
if(betingelse 1){  
    statement1;  
} else if(betingelse 2){  
    statement2;  
} else if(betingelse 3){  
    statement3;  
} else{  
    statement4;  
}
```

Eksempel:

```
if(alder < 6) {  
    console.log("før skolealder");  
} else if(alder < 15) {  
    console.log("skolealder");  
} else if(alder < 65) {  
    console.log("erhvervsaktiv alder");  
} else {  
    console.log("pensionsalder");  
}
```

Indlejrede betingelser (nested if-statements)

Et eller flere statements skal KUN udføres, hvis en anden betingelse er opfyldt først, f.eks:

```
function tjek(alder, køn) {  
  if (alder < 18) {  
    if (køn == "m") {  
      return ("dreng");  
    } else if (køn == "k") {  
      return ("pige");  
    } else {  
      return ("barn");  
    }  
  } else {  
    if (køn == "m") {  
      return ("mand");  
    } else if (køn == "k") {  
      return ("kvinde");  
    } else {  
      return ("voksen");  
    }  
  }  
}
```

Øvelse 02 - er arealet mellem 100 og 200?

Gem [01-areal.html](#) i en ny kopi, [02-arealTest.html](#)

I programmet skal du erklære to variabler: [længde](#) og [bredde](#).

Programmet skal udregne arealet og fortælle om arealet er for lille, ok eller for stort. Resultatet skal vises i console.

Hvis arealet er under 100, skal der stå: **Arealet er for lille**

Er arealet mellem 100 og 200, skal der stå: **Arealet er ok**

Er det større end (eller lig med) 200, skal der stå: **Arealet er for stort**

Test programmet, så du ser alle tre muligheder for respons i funktion.

Commit og push til gitHub, når du er tilfreds med opgaven

tema7
undervisningsopgaver
01-js-basics:
[01-areal.html](#)
[02-arealTest.html](#)

Opsamling - hvad var det nu, det betød????

Begreber og ord

if

betingelse/condition

sammenlignende operator

else

else if

Sammenlignende operatorer

== (er lig med)

!= (er forskellig fra)

< (er mindre end)

> (er større end)

<= (er mindre end eller lig med)

>= (er større end eller lig med)

&& (og)

|| (eller)

Interaktivitet

1. Elementer der skal bruges i de interaktive øvelser

Knap

For at lave opgaverne skal du bruge en knap. Hvad som helst kan bruges som knap, hvis man lægger en click event listener på den som her:

```
minKnap.addEventListener("click", funktionDerAktiveresNårDerKlikkes) ;  
  
function funktionDerAktiveresNårDerKlikkes() {  
    console.log("Du har klikket på knappen") ;  
}
```

2. Elementer der skal bruges i de interaktive øvelser

Input felt

Du skal også bruge et inputfelt:

```
<input type="text" id="name">
```

For at finde ud af hvad der står i dette tekstfelt, kan vi bruge flg. javascript:

```
document.querySelector("#name").value
```

Eksempel

```
<body>
  <h1>Tjek myndighed</h1>

  Indtast din alder: <input type=number>
  <button>Tjek</button>

  <script>
    const knap = document.querySelector("button");
    knap.addEventListener("click", tjek);

    function tjek() {
      const alder = document.querySelector("input").value;
      if (alder < 18) {
        console.log("Du må ikke købe øl");
      } else {
        console.log("Du må købe al det øl du har råd til");
      }
    }

  </script>
</body>
```

Tjek myndighed

Indtast din alder:

Øvelse 03 Sig goddag

Lav en ny html-fil, **03-goddag.html**, og gem den i 01 **01-js-basics**

Skriv et program, som spørger brugeren om hans/hendes navn (inputFelt), og ved et tryk på en knap herefter svarer med “Goddag Alan” (console.log), hvor navnet selvfølgelig er det angivne.

Commit og push til gitHub, når du er tilfreds med opgaven

tema7
undervisningsopgaver
01- js-basics:
01-areal.html
02-areaTest.html
03-goddag.html

Øvelse 04 udregn areal i interaktiv dialog med bruger

html-filen til denne øvelse, skal hedde **04-arealInteraktiv.html**

Lav en ny udgave af areal-programmet.

Programmet skal bede brugeren angive en længde og en bredde ved hjælp af inputfelter.

Ved klik på en knap skal programmet udregne arealet, og give brugeren resultatet efter formen:

Længde: 5 meter, **bredde:** 7 meter, **areal:** 35 meter(længde*bredde)

tema7
undervisningsopgaver
01- js-basics:
01-areal.html
02-areaTest.html
03-goddag.html
04-arealInteraktiv.html

Funktioner

Funktioner - hvorfor

Vi kikker lige på funktioner igen, vi har jo kikket på dem højere oppe i forbindels med eventlistenerer. En funktion er et stykke isoleret kode.

Funktionen skal erklæres:

```
function showMessage() {  
  alert( 'Hello everyone!' );  
}
```

The diagram illustrates the structure of a JavaScript function declaration. Red arrows point from descriptive text to specific parts of the code: 'The name of the function' points to 'showMessage()', 'Parameters (empty here)' points to the empty parentheses '()', and 'The body of the function (the code)' points to the code block between the curly braces '{ }'.

Når en funktion er erklæret, sker der **ingenting**.
Først når den **kaldes**, udføres funktionen:
`showMessage();`

Når vi henviser til en funktion i en eventListener, f.eks
`addEventListener("click", showMessage);`
kalder vi den **ikke**, da der er **ikke** nogen `()` bagefter, for hvis
der var det, ville funktionen blive kaldt **når** eventListeneren
bliver indlæst og ikke når vi klikker på knappen - altså med
det samme!

Funktioner - hvorfor?

DRY - Don't Repeat Yourself!

- Man undgår at gentage kode
- Programmet bliver modulopbygget og mere overskueligt
- Funktioner kan også tit genbruges i andre programmer

Derfor forsøger man altid at oprette funktioner, hvor man kan.

Funktion med parameteroverførsel

```
function visBesked(txt){  
    console.log(txt);  
}
```

Erklæring af funktionen

Funktionen vil vise en værdi i consolen i inspectoren.

txt er det variabelnavn funktionen bruger om værdien, kaldes også et parameter

```
visBesked("Goddag");
```

Funktionen kaldes med værdien "Goddag" som parameter/argument. I funktionen får variabelen txt denne værdi, og den logges i consolen.

```
let besked = "Farvel";
```

```
visBesked(besked);
```

Funktionen kaldes med variabelen **besked** som parameter/argument. **besked** har værdien "Farvel" fra linjen før. funktionen får variabelens værdi som parameter. I funktionen får variabelen txt denne værdi, og den logges i consolen.

Funktioner som returnerer en værdi

```
function visBesked(message){  
  let first = "Info: ";  
  return `${first} ${message}`;  
}
```

Funktionen sætter teksten "Info:" foran den værdi den modtager som parameter.
Den nye tekst returneres

```
let besked="Kamilla vejleder i eftermiddag";  
console.log(visBesked(besked));
```

```
let info="Martin underviser i dag";  
console.log(visBesked(info));
```

```
console.log(visBesked("Louise underviser først i næste uge"));
```

I console.log-sætningen kaldes funktionen med variabelen *besked* som parameter.
besked har værdien "Kamilla underviser i morgen"
i funktionen kaldes værdien for *message*

Opsamling

- funktion
- parameteroverførsel
- funktion som returnerer en værdi

Øvelse 05 Arealer og funktioner

Filens navn: **05-arealFunktion.html**

Lav en funktion, **areal**, som med længde og bredde som parametre, kan udregne et areal, og returnere resultatet.

Programmet skal:

- Bede brugeren om længde og bredde med input elementer
- Videregive resultatet til brugeren ved hjælp af consolen. HVIS arealet ikke er et gyldigt tal, dvs. at det er 0 og derunder, skal brugeren have at vide at det ikke er en gyldig indtastning.

Commit og push til gitHub, når du er tilfreds med opgaven

tema7

undervisningsopgaver

01- js-basics:

01-areal.html

02-areaTest.html

03-goddag.html

04-arealInteraktiv.html

05-arealFunktion.html

Øvelse 06 Arealudregning med fejlmeddelelse

Filens navn: **06-arealFejl.html**

I det forgående program, arealFunktioner.html:

Test, hvad der sker, hvis brugeren angiver tekst i stedet for tal som længde eller bredde

Prøv at ændre på programmet, så det tager højde for fejldata og giver brugeren en fejlmelding

Commit og push til gitHub, når du er tilfreds med opgaven

Husk at man kan teste om noget er et tal med isNaN(). NaN betyder: Not a Number

tema7

undervisningsopgaver

01- js-basics

01-areal.html

02-areaTest.html

03-goddag.html

04-arealInteraktiv.html

05-arealFunktion.html

06-arealFejl.html

Eftermiddagsopgaver

Øvelse 07 Sig godmorgen

html-filen til denne øvelse, skal hedde `03-godmorgen.html`

Skriv et program, som i consolen siger:

Godmorgen mellem kl. 5 og kl 10,

Goddag mellem kl 10 og 18,

Godaften mellem 18 og 24 og

Godnat mellem 24 og 5.

obs: denne javascript-funktion fortæller, hvilken time, man befinder sig i:

`new Date().getHours()`

Commit og push til gitHub, når du er tilfreds med opgaven

Øvelse 08 - simpel CO2 beregning

html-filen til denne øvelse, skal hedde **08-co2_beregning.html**

Du skal lave et program, som regner ud hvor meget CO2 du bruger på en flyrejse:

$$\text{co2UdledtIKgVedFlyrejse} = \text{timer} \times 109$$

Du skal ikke i dialog med brugeren i denne opgave - læg timer ind som en variabel.

Resultatet skal vises i console-vinduet efter denne form:

Ved 3 timers flyrejse har du udledt 327 kg CO2

Commit og push til gitHub, når du er tilfreds med opgaven

Øvelse 09 - udregn omkreds #2

html-filen til denne øvelse, skal hedde

09-type_af_klimasynder

Programmet skal udregne udledning af CO₂ ved flyrejse på baggrund af flyrejsens længde i timer

(tag udgangspunkt i en kopi af øvelse 2).

Ud fra dette skema, skal du i consolen udskrive både timer og hvor meget CO₂ der er udledt.

Fx: **Du har fløjet 3 timer og udledt 327 kg CO₂.**

Det er godt for turismen i nordeuropa.

Commit og push til gitHub, når du er tilfreds med opgaven

(galgenhumoristisk) respons	CO ₂ i kg
Tæt på Thunberg	mindre end 100
Ja ja, du behøver ikke at melde dig ud af Å!	Mellem 100 og 200
Det er godt for turismen i nordeuropa.	Mellem 200 og 600
Nå ja vi kan jo altid kolonisere en anden planet ...	Over 600

Øvelse 10 Udregn bmi i interaktiv dialog med bruger

html-filen til denne øvelse, skal hedde [10-type_af_klimasynder_interaktiv.html](#)

Lav en ny udgave af CO2-programmet.

Programmet skal bede brugeren angive længden på sin flyrejse i timer.

Ved klik på en knap skal programmet så udregne CO2-udledningen og give respons i consolen

Hvis brugeren laver et forkert input, skal programmet give en fejlmedling.

Commit og push til gitHub, når du er tilfreds med opgaven

Øvelse 11 - quiz-program

html-filen til denne øvelse, skal hedde `quiz.html`

Lav et program, som kan stille 5 spørgsmål til brugeren, et ad gangen, og give et point for hvert rigtige svar.

Når alle spørgsmål er besvaret, skal programmet fortælle brugeren, hvor mange svar, der var rigtige.

Undervejs skal programmet fortælle om hver enkelt svar var rigtigt.

Bestem selv, hvad quizen skal handle om.

Din løsning er bedst, hvis den indeholder en funktion, som kan tage sig af at sammenligne det rigtige svar med et svar fra brugeren.

Commit og push til gitHub, når du er tilfreds med opgaven

Øvelse 12 - Gæt et tal

html-filen til denne øvelse, skal hedde `guess.html`

Programmet skal finde et tilfældigt tal mellem 0 og 20, og bede brugeren om at gætte tallet det.

Når brugeren har gættet, fortæller programmet om tallet var rigtig, eller om det var for højt eller for lavt.

Så får brugeren lov at gætte igen, og sådan fortsætter programmet til brugeren har fundet det rigtige tal.

Når brugeren har gættet tallet, fortæller programmet, hvor mange gæt, der blev brugt, og spørger om brugeren vil prøve igen med et nyt tal.

Commit og push til gitHub, når du er tilfreds med opgaven

Øvelse 13 - Date-rådgivning

html-filen til denne øvelse, skal hedde **date.html**

Når man skal date, bør man finde en, som hverken er for ung eller for gammel.

Der findes en regel, som hedder “half your age plus seven”.

Lav et program, som kan tage imod din egen alder og din dates alder, og fortælle dig, om reglen er overholdt - både til den ene og den anden side.

Sørg endelig for, at programmet ikke kan benyttes af mindreårige eller pædofile - begge skal være over 15!

Commit og push til gitHub, når du er tilfreds med opgaven