App Lab

Vigtige kommandoer

Malene Cramer Engebjerg

Version 2

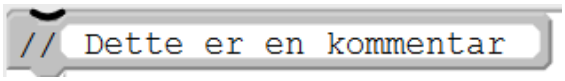
1. Indhold

I App Lab kan man lave det, der hedder "blok-programmering". Dvs. at man kan programmere en app, uden at skulle *skrive* en masse tekst, hvor man er nødt til at huske på en masse kommandoer og sørge for at skrive dem meget præcist. I stedet kan man trække "blokke" over i et arbejdsområde i App Lab og arbejde derfra.

Denne note er ikke tænkt som en egentlig introduktion til App Lab og kan således ikke stå alene. Den er mere tænkt som et supplement og et opslagsværk med uddybende forklaringer til nogle af de vigtigste kommandoer, man får brug for i App Lab.

2. Kommentarer

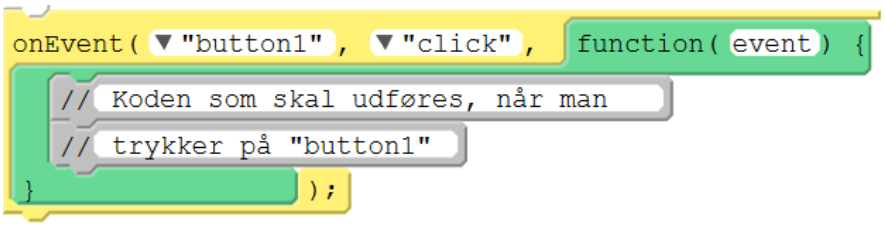
En kommentar i koden (som du bør bruge ofte!) ser sådan ud:



Den findes i værktøjskassen under "Functions" (grøn).

3. Der skal ske noget, når man trykker på en knap

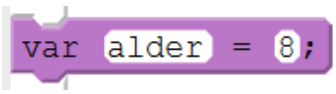
Brug følgende kommando:



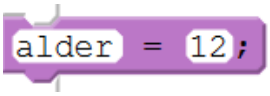
Her er "button1" id'et på knappen og "click" betyder, at der sker noget, når man trykker/klikker på knappen.

4. Variable

Man kan tænke på en variabel som en lille skuffe i "maven" af computeren, som har et navn. Inde i denne skuffe kan man så opbevare en værdi. En variabel i App Lab laves sådan her:



Her er der lavet (erklæret) en variabel, som har navnet "alder" og denne variabel har fået tildelt værdien 8. Hvis man får brug for denne værdi, kan man bare skriver "alder" i stedet for 8. Hvis man gerne vil ændre værdien af variablen - lad os sige at den nu skal have værdien 12 - så skriver man bare:



Læg mærke til at der ikke står ”var” foran længere. Det er fordi variablen allerede er ”lavet”, den har bare fået en ny værdi (man har taget værdien 8 op af skuffen og puttet 12 ned i stedet for).

Hvis man i stedet gerne vil lægge 1 til den nuværende værdi af variablen, gør man sådan her:

```
alder = alder + 1;
```

Nu vil variablen ”alder” have værdien 13 i stedet for 12.

Variable behøver ikke at indeholde tal, de kan også indeholde tekst (på engelsk siger man ”string” og på dansk nogle gange ”streng” eller ”tekststreng”).

Her er et eksempel:

```
var navn = "Emma";
```

Denne variabel har navnet ”navn” og værdien ”Emma”. Bemærk at tekst skrives i dobbelt anførselstegn (”plinger”), mens variabelnavne og tal ikke gør det. Lad os sige at vi ikke længere vil gemme navnet ”Emma”, men i stedet ”Julie” i variablen ”navn”, så gøres det sådan her:

```
navn = "Julie";
```

Hvis vi i stedet for ”Julie” også gerne vil gemme Julies efternavn, som er ”Jensen”, så kan man tilføje noget til en tekst-variabel på følgende måde:

```
navn = navn + " Jensen";
```

Nu indeholder variablen ”navn” værdien ”Julie Jensen”. Man kan tænke på det på den måde, at man lægger noget ekstra ned i ”skuffen”. Læg mærke til at der står et mellemrum lige inden ”Jensen”. Hvis det ikke havde stået der, ville variablen ”navn” i stedet have fået værdien

”JulieJensen”

og det var ikke helt det, vi havde tænkt. Ønsker man at indsætte en ny linje bruges ”\n” (n står for newline). F.eks. vil

```
var navne = "Julie Jensen \n Peter Hansen";
```

Give resultatet (mere korrekt: værdien af variablen ”navne” vil være):

Julie Jensen

Peter Hansen

Alternativt kan det også skrives:

```
var navne = "Julie Jensen" + "\n" + "Peter Hansen";
```

5. At få fat i tekst fra et tekstfelt

Nedenfor ses en enkel app med et tekst input felt, som har id "text_input1" (vælg selv bedre og mere sigende navne!).

Bemærk, når man holder musen henover tekst input feltet, så vises feltets placering og feltets id.

Brugeren kan skrive noget i dette feltet, og hvis vi gerne vil *læse*, det brugeren har skrevet, og gemme det i en variabel, gøres det vha. kommandoen "getText()":

```
var navn = getText( "text_input1" );
```

Bemærk, at i parentesen skriver man tekst input feltets id.

6. At skrive tekst til et tekstfelt

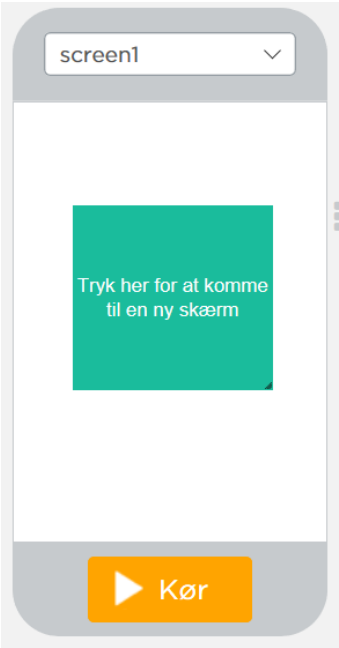
Ønsker man omvendt at *skrive* til et tekstfelt gøres det vha. "setText()" kommandoen:

```
setText( "text_input1" , "Her er min tekst" );
```

Det første man skriver i parentesen er id'et på tekstfeltet, mens det andet er den tekst, man gerne vil have skrevet.

7. At skifte skærm

Hvis man gerne vil skifte til en ny skærm, når man f.eks. trykker på en knap gøres det vha. setScreen() kommandoen.



Knappen på skærmen til venstre har id'et "button1". Nedenstående kode sørger for at skifte til en ny skærm med id'et "screen2", når man trykker på knappen ("button1").

```
onEvent( "button1" , "click" , function( event ) {
  setScreen( "screen2" );
} );
```

Se også videoen "Tilføje skærmbillede og knap".

8. Forgrening med én gren (if)

Hvis man kun skal udføre et stykke kode (B), hvis en given betingelse er opfyldt, skal man bruge en if-sætning:

```
if ( A==sand ) {
  // udfør B
}
```

Her udføres koden B kun, hvis betingelsen A er sand. Betingelsen kunne f.eks. være en af følgende:

```
alder > 18
```

```
farve == "blå"
```

```
BMI <= 25
```

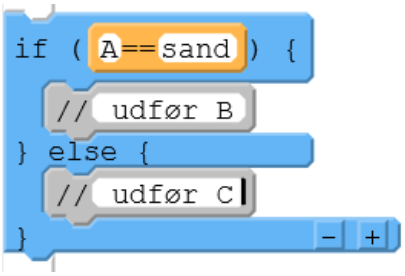
```
antal != 6
```

Her vil betydningerne henholdsvis være:

1) Hvis variablen "alder" har en værdi, der er større end 18, udføres koden i B
2) Hvis variablen "farve" har værdien "blå", udføres koden i B
3) Hvis variablen "BMI" har en værdi, som er mindre end eller lig med 25, udføres koden i B
4) Hvis variablen "antal" *ikke* er 6, udføres koden i B

9. Forgrening med to grene (if-else)

Hvis man ønsker at få udført et stykke kode (B), hvis betingelsen A er sand, og et andet stykke kode (C), hvis betingelsen A ikke er sand (dvs. den er falsk) skal man bruge en if-else sætning:

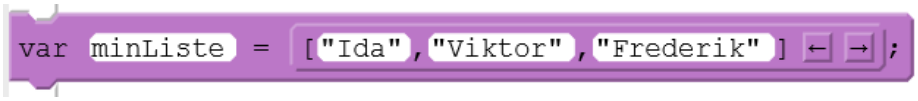


Betingelsen, som skal vurderes, kan igen være én af de foregående.

Se også videoen "Sådan laves en if-else sætning".

10. Liste/array

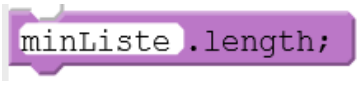
Hvis man har brug for en variabel, som kan indeholde mere end én værdi af gangen, skal man bruge en liste (kaldes også for et array). Det gøres sådan her:



Ønsker man at få fat i det første element i listen skriver man:

Bemærk, at man starter ved 0. Derfor fås det tredje element i listen ved at skrive:

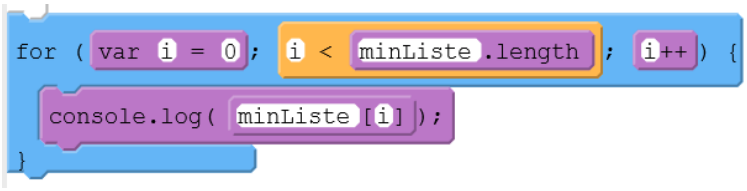
Ønsker man længden af listen, skriver man:



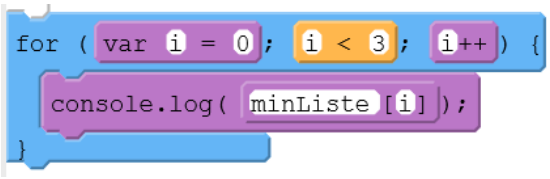
I dette eksempel er længden af listen 3, fordi der er 3 elementer i listen.

11. For-løkke

Når man ønsker at gentage noget kode et bestemt antal gange, kan man bruge en for-løkke. Hvis man f.eks. vil udskrive alle navnene i arrayet i det foregående eksempel til konsollen (som vises nederst på siden, når man har App Lab åben), så skriver man:



Det kræver lidt forklaring. For det første har vi set, at minListe.length har værdien 3, fordi der er 3 elementer i listen. Dvs. at i dette eksempel svarer ovenstående til:



Overvej hvorfor det er smartere at skrive minListe.length i stedet for 3!

Dernæst ser vi, at der i parentesen lige efter "for" står tre sætninger:

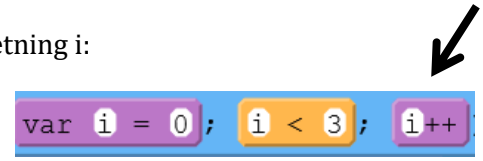
Den første sætning:

definerer midlertidigt en variabel "i", som starter med at have værdien 0. Dernæst undersøges det, om betingelsen i den anden sætning er opfyldt. Dvs. vi spørger om

Da 0 er mindre en 3, er betingelsen opfyldt, og koden der står inde i løkken udføres. I dette eksempel svarer dette til, at følgende kode udføres:

Bemærk, at "i" er skiftet ud med 0, fordi variablen "i" har værdien 0. Det, der står i "minListe" på plads nr. 0 (man starter som tidligere nævnt altid med at tælle fra 0), er "Ida", og derfor skrives der nu "Ida" i konsollen.

Nu sker der det at den tredje sætning i:



udføres. Det ser igen lidt mærkeligt ud, men det betyder: gør værdien af variable "i" én større. Da værdien af variablen "i" er 0, så skal værdien af "i" altså ændres til 1. Og så kører det hele en gang til. Vi spørger igen - er:

```
i < 3
```

Og ja 1 er mindre end 3 og derfor udføres følgende kode:

```
console.log( minListe [1] );
```

Elementet på plads nr. 1 i arrayet er ”Viktor”, og derfor skrives der nu ”Viktor” i konsollen. Så hopper vi igen tilbage til den tredje sætning:

```
i++
```

og tæller værdien af variablen ”i” én op. Den har altså nu værdien 2. Igen ser vi at 2 er mindre end 3 og vi udfører derfor koden:

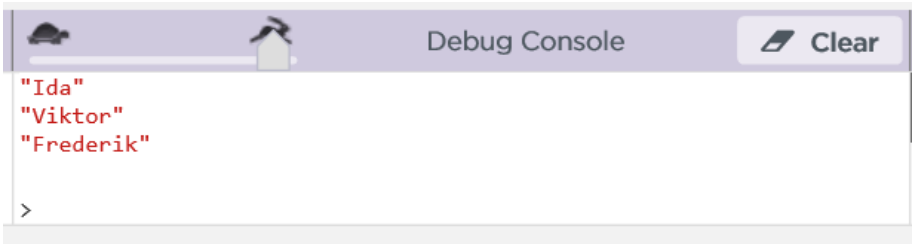
```
console.log( minListe [2] );
```

Der skrives altså nu ”Frederik” i konsollen. Så tæller vi igen værdien af variable ”i” én op, så den nu har værdien 3. Og vi spørger igen - er:

```
i < 3
```

Men da 3 ikke er mindre end 3, er betingelsen altså ikke længere sand, og vi forlader derfor for-løkken.

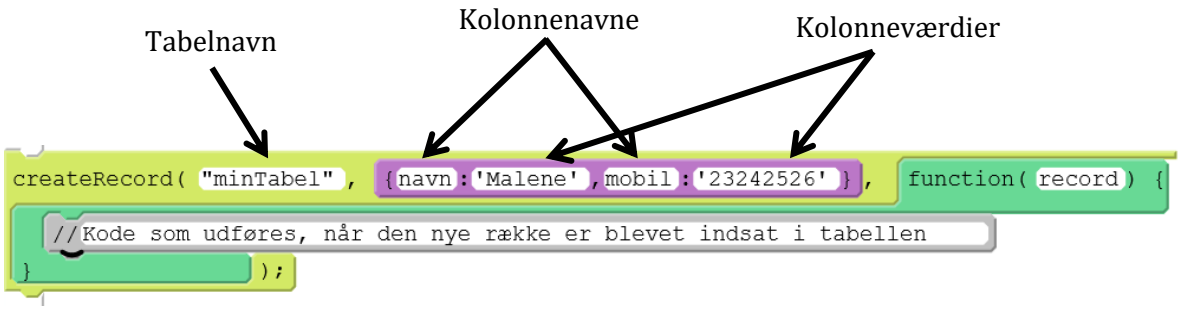
Resultatet af denne lille stump kode bliver, at der i konsollen til sidst står:



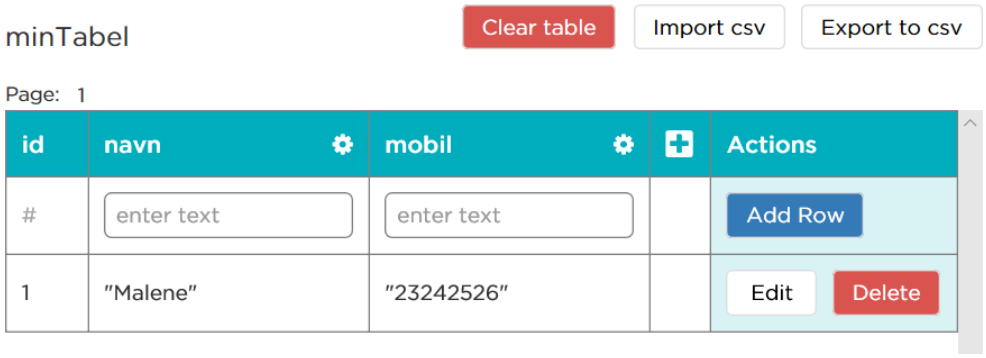
Se også videoen ”Sådan laves en løkke”.

12. Gemme data i datalaget

Ønsker man at indsætte en række i en tabel i datalaget i App Lab skal createRecord()-kommandoen benyttes. Nedenfor er vist et eksempel på en kommando, hvor man indsætter en række i en tabel, som hedder ”minTabel”. Tabellen har to kolonner (dvs. søjler), som hedder ”navn” og ”mobil”. Kommandoen herunder indsætter værdien ”Malene” i kolonnen ”navn” og værdien ”23242526” i kolonnen ”mobil”.



Det giver følgende resultat i datalaget:



Bemærk, at det faktisk ikke er nødvendigt at oprette tabellen i datalaget først. Det sker automatisk, første gang App Lab bliver bedt om at indsætte data i en tabel, som ikke allerede er oprettet.

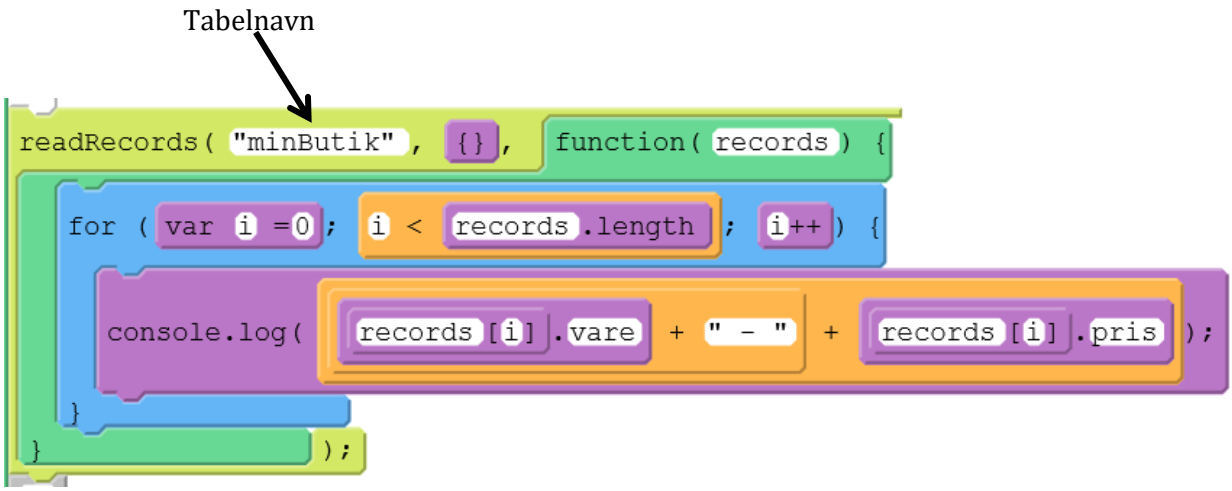
Se også videoen "Sådan gemmes i datalaget".

13. Læse alle data fra datalaget

Lad os sige at vi har nedenstående tabel (kaldet for "minButik") gemt i datalaget:

id	vare	pris	farve
#	enter text	enter text	enter text
1	"nederdel"	300	"sort"
2	"busker"	450	"blå"
3	"bluse"	200	"hvid"
4	"t-shirt"	100	"sort"

Hvis vi gerne vil læse alle data fra tabellen gøres det vha. kommandoen readRecords():



Det ser en anelse kompliceret ud, så vi tager det lidt ad gangen. Det første, vi bemærker, er, at lige efter readRecords() skrives navnet på tabellen (her "minButik"). Når man vha. readRecords() læser fra en tabel, bliver alle tabellens rækker lagt over i et array, som kaldes for "records". Den første række kan man således få fat i ved at skrive records[0] (husk vi altid starter ved 0 og ikke 1), den anden række fås ved at skrive records[1] osv. Se nedenstående figur:

id	vare	pris	farve	
#	enter text	enter text	enter text	
1	"nederdel"	300	"sort"	← records[0]
2	"busker"	450	"blå"	← records[1]
3	"bluse"	200	"hvid"	← records[2]
4	"t-shirt"	100	"sort"	← records[3]

Hvis vi nu f.eks. gerne vil have fat i det, der står i kolonnen "pris" i den tredje række, så skriver vi blot

records[2].pris

Se flere eksempler i figuren herunder:

id	vare	pris	farve
#	enter text	enter text	enter text
1	"nederdel"	300	"sort"
2	"busker"	450	"blå"
3	"bluse"	200	"hvid"
4	"t-shirt"	100	"sort"

records[0].vare

records[1].farve

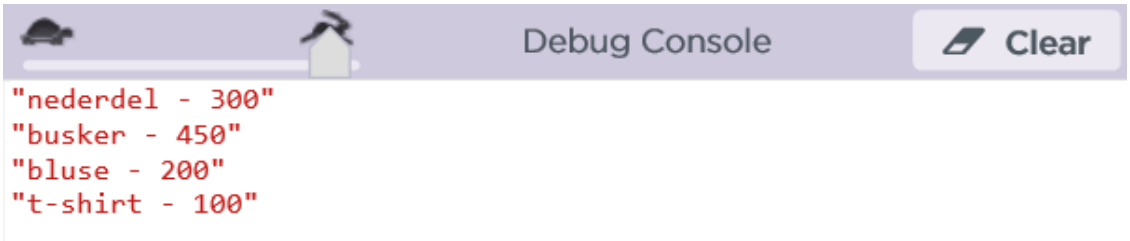
records[3].pris

Går vi tilbage til den oprindelige readRecords()-kommando:

```
readRecords( "minButik", {}, function( records ) {
    for ( var i =0; i < records.length; i++ ) {
        console.log( records[i].vare + " - " + records[i].pris );
    }
} );
```

Så kan vi nu se, at for-løkken gennemløber alle rækkerne i tabellen og for hver række udskrives det, der står i kolonnen "vare" og "pris" i konsollen (adskilt af en bindestreg).

I konsollen kommer der derfor til at stå:



Se også videoen "Sådan hentes fra datalaget".

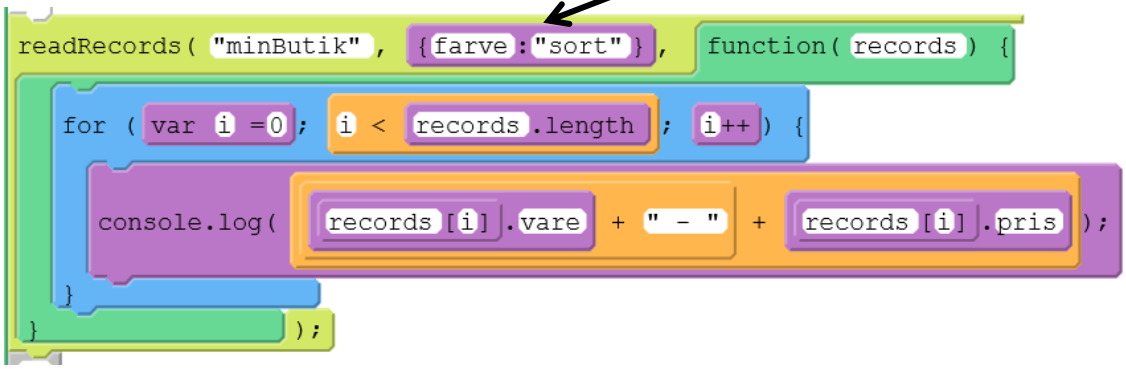
14. Læse udvalgte data fra datalaget

Vi går videre med eksemplet fra forrige afsnit og tænker nu, at vi gerne vil have fat i alle de stykker tøj, hvor farven er sort:

id	vare	pris	farve
#	enter text	enter text	enter text
1	"nederdel"	300	"sort"
2	"busker"	450	"blå"
3	"bluse"	200	"hvid"
4	"t-shirt"	100	"sort"

Det gøres næsten som før:

Betingelse som angiver, at vi kun ønsker at læse de rækker, hvor kolonnen "farve" har værdien "sort"



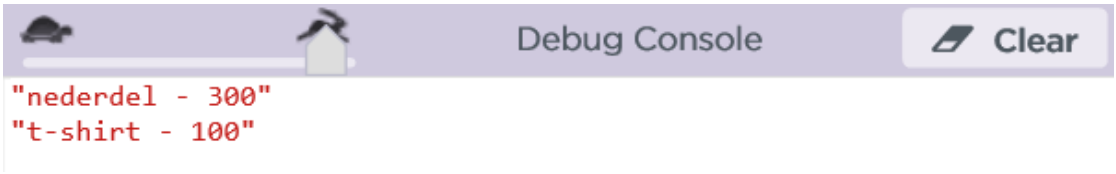
Da der kun er to rækker, hvor kolonnen "farve" har værdien "sort", bliver "records" nu et array med to elementer:

id	vare	pris	farve
#	enter text	enter text	enter text
1	"nederdel"	300	"sort"
2	"busker"	450	"blå"
3	"bluse"	200	"hvid"
4	"t-shirt"	100	"sort"

← records[0]

← records[1]

Derfor bliver resultatet af den ovenstående readRecord()-kommando følgende:



Se også videoen "Sådan hentes fra datalaget".

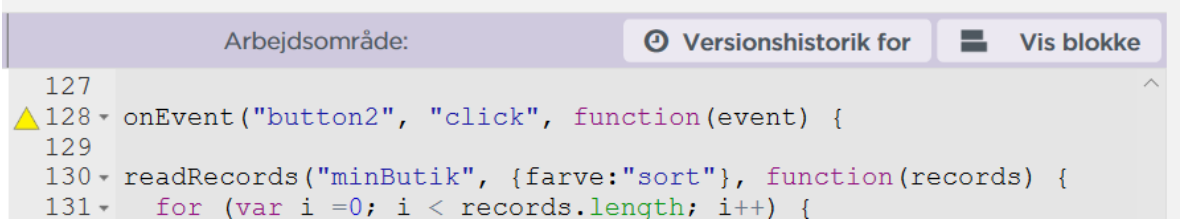
15. Yderligere hjælp at hente

Når man har valgt en kommando i værktøjskassen i App Lab, kan man holde musen henover kommandoen, og der dukker en blå boks op, hvor det er muligt at trykke på "See examples". Her står lidt generelt om, hvordan kommandoen fungerer, og der er vist nogle eksempler. Det er ofte en stor hjælp.

Desuden er navnene på kommandoerne ofte ret sigende, så det er tit en hjælp bare at sidde og bladre lidt i de forskellige værktøjskasser.

Når man programmerer i App Lab, programmerer man i virkeligheden i et programmeringssprog, som hedder Javascript. I stedet for at se kommandoerne som blokke, kan man i stedet se selve teksten ved at vælge "</> Vis tekst" i øverste højre hjørne i arbejdsområdet:

Trykker man her får man:



Nogle gange er det nemmere at redigere her, men man skal også være varsom, for man kan komme til at lave (syntaks-)fejl, som gør, at man ikke kan komme tilbage til blokkene.

En anden fordel er, at man her kan indsætte Javascript kode, der ikke på forhånd er lavet blokke til.

Mere generel hjælp til Javascript kan findes på w3schools:

https://www.w3schools.com/js/default.asp

Hvis man f.eks. har brug for lidt mere matematik (udregning af potenser eller kvadratrødder), så kan man få hjælp her:

https://www.w3schools.com/js/js_math.asp