Semesteroppgave INFO134

VÅR2018

Kandidatnummer: 139

Mappestruktur:

Mappe oppgave:

- arbeidskraft.html
- favorittLekeplass.html
- index.html
- lekeplasserIBergen.html
- toaletterIBergen.html
- semesteroppgaveINFO134.pdf
- Mappe css:
  - style.css
- Mappe js:
  - arbeidskraftScript.js
  - script.js
  - ScriptFavoritt.js
- Mappe media:
  - bib.png
  - dummyHeader.png

Her har jeg valgt å splitte opp JavaScript i tre dokumenter. Dette for lette å holde oversikt og for små nyanser i funksjoner slik at oppførselen kan være forskjellig etter sitt formål uten at det blir for oversiktlig. Script.js er skript for toaletterIBergen.html og LekeplasserIBergen.html. ScriptFavoritt.js et koblet mot favorittLekeplass.html, og arbeidskraftScript.js er knyttet opp mot arbeidskraft.html. index.html har ikke noe script knyttet opp til seg.

Jeg har valgt å holde CSS-filen samlet. Dette er en enkel nettside og jeg har klart å holde stylingen oversiktlig i CSS-dokumentet.

Videre vil jeg gå gjennom oppgaven stegvis, noen oppgaver er slått sammen til én underoverskrift.

Oppgave 1

For å holde oversikt over alle sidene har jeg øverst på hver side en meny hvor man kan navigere til alle de andre html-dokumentene i oppgaven. Den siden man befinner seg i har en aria-label="current page" slik at man til enhver tid skal vite hvor man befinner seg i dokumentet.

Nettsiden benytter seg av flexbox for å gjøre den responsiv basert på skjermens størrelse.

Oppgave 2-4

I dokumentene toaletterIBergen.html og lekeplasserIBergen.html er det laget script med window.dataUrl og url til datasettet, og window.page med respektivt lekeplass og toalett som verdi. I script.js så initierer funksjonen initApp programmet med å kalle loadData med parameter window.dataUrl. Funksjonen loadData tar lokal parameter url. Hvis URL er hentet fra siden med window.page verdi toalett, så kaller den funksjonen readToaletter, hvis ikke så vil den kalle funksjonen readLekeplass når tilstanden er 4 og staus er OK. Dataene blir presentert som en tabell med tall og navn på toalettet, samt vises på kartet og med markører som korresponderer med tallet i listen.

Oppgave 5

Ved *avansert søk* har jeg valgt å lage et skjema med kombinasjon av fritekst for adressesøk, radio-valg for kjønn, sjekkboks for tilgjengelighet (rullestol og stellerom), åpen nå og gratis. For å velge et spesifikt klokkeslett er det mulig å skrive inn tid, og for makspris er det mulig å skrive inn tall mellom 0 og 20 eller trykke pil opp eller ned for å justere makspris. Søket blir igangsatt ved å trykke på søk.

Da initieres funksjonen readSearchAvansert og opprettet objektet avansertToalett, og sjekker hvilke verdier som er oppdatert og endrer disse og sender objektet videre til funksjon filterSearch.

Ved *hurtigsøk* kan man skrive i fritekstfeltet:

Skrive inn navn på toalettet eller gatenavn – dette vil matche navn, på norsk eller engelsk, eller adresse.
mann – dette vil matche herretoalett og pissoar
dame – dette vil matche dame
stellerom – dette vil matche stellerom
rullestol – dette vil matche rullestol
gratis – dette vil matche gratis
åpen:00:00 – dette vil matche om toalettet er åpen på et gitt tidspunkt i dag, eventuelt neste dag.
åpen – vil matche på de som er åpen nå
pris:00 – vil matche på de toalettene som har pris tilsvarende eller lavere

Alle søkene er case insensitive og vil dermed matche på både store og små bokstaver. Den vil også matche hvis man søker etter navn på stedet eller gate der toalettet befinner seg. Det er også mulig å søke etter mer enn en ting om gangen. Man søker ved å trykke på søk-knappen, og kun søk-knappen.

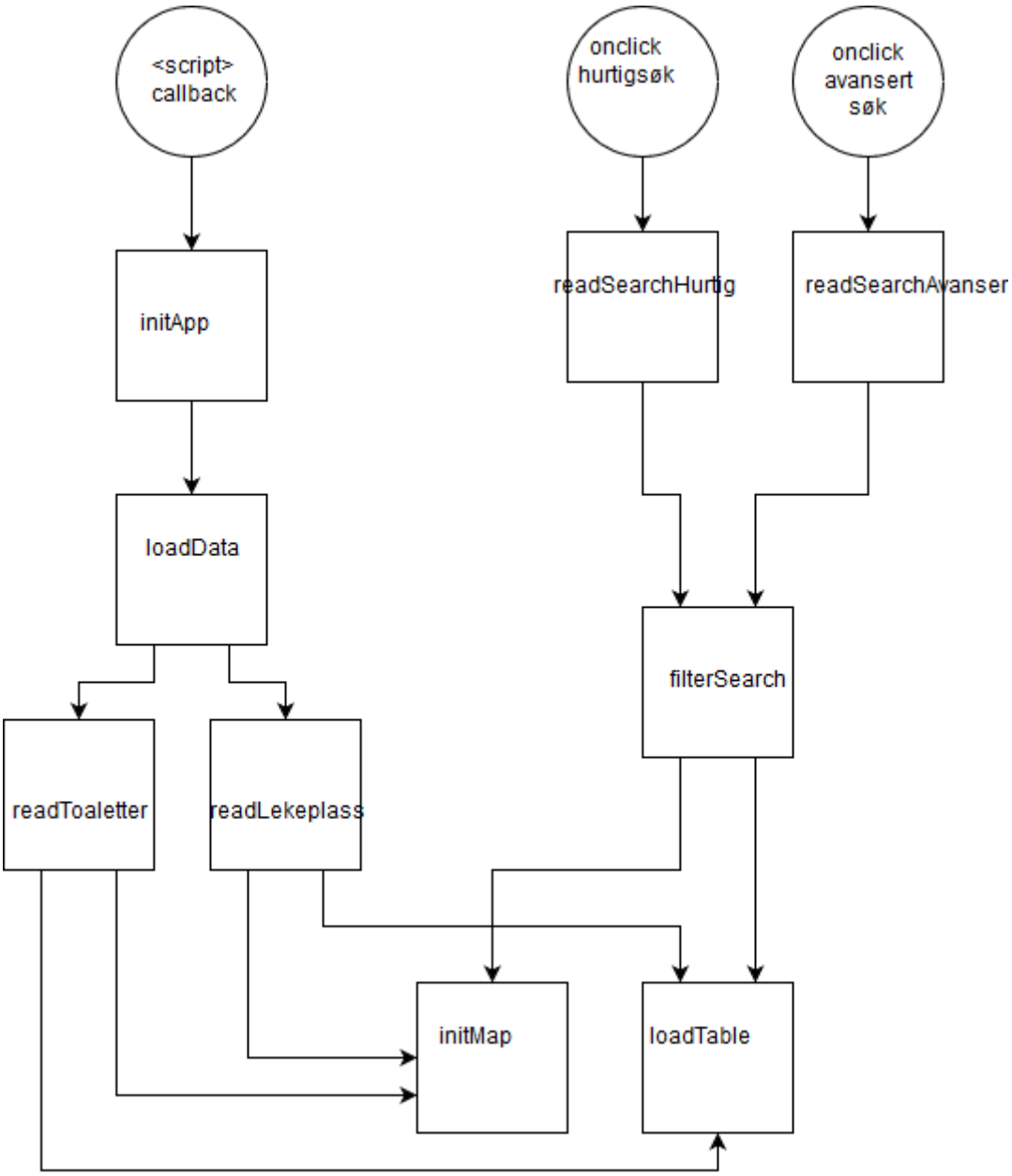
Når man har søkt enten via hurtigsøk eller avansertsøk, så blir filterSearch initiert. Her sjekker filteret om man har søkt etter sted, adresse eller engelske stedsnavnet. Hvis det er tilfellet så vil elementet bli lagt til arrayet locations. Så går den systematisk igjennom og se hvilket kjønn som er valgt, om man har haket av for rullestol, stellebord, gratis eller åpen nå. Den sjekker også makspris og om et toalett er åpen på et gitt klokkeslett i dag, eventuelt i morgen. Funksjonen benytter seg av funksjoner som har som oppgave å finne ut hvilken dag det er, hva klokken er og om toalett er åpen. Dette blir benyttet for å finne ut om et toalett er åpen når man søker etter åpen nå og det vil være åpen på et gitt tidspunkt.

Etter hvert som den sjekker hva som er søkt for så filtrerer funksjonen søkeresultatene slik at man til slutt kun sitter igjen med de toalettene som matcher alle søkekriteriene. Når filtreringen er gjennomfør så sendes data til tabell og kart som da blir oppdatert.

Oppgave 6

Her er dataene presentert på samme måte som i oppgave 3. her er en tabell med nummerert liste og navn på toalettene, samt markører på kartet som korresponderer med den nummererte listen.

Under er en oversikt over logikken i oppbygging av JavaScript-filen script.js som er knyttet til toaletterIBergen.html og lekeplasserIBergen.html:



Oppgave 7

Her har jeg brukt formelen:

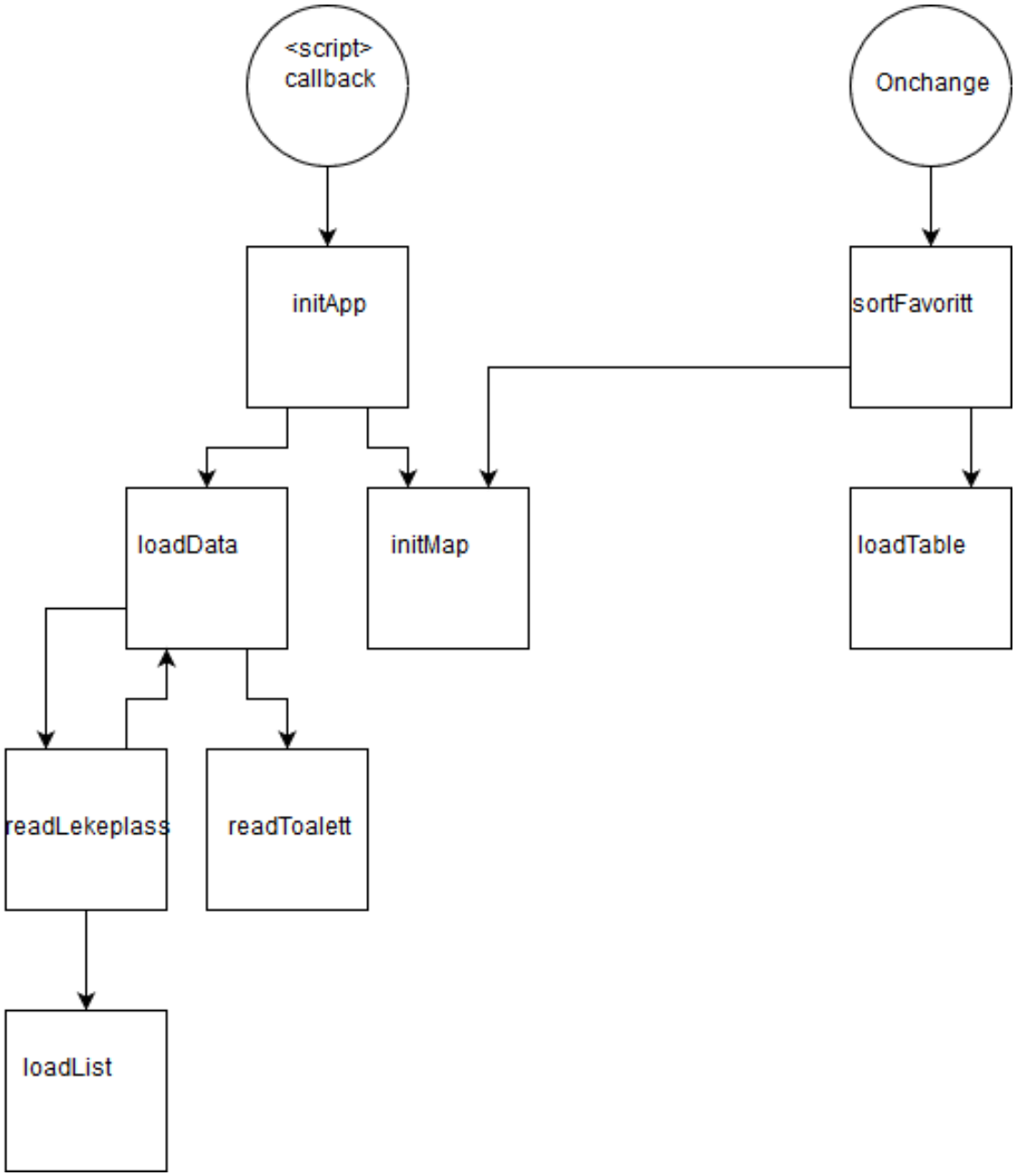
$$\text{distanse} = \sqrt{(\text{lendegradA} - \text{lengdegradB})^2 + (\text{breddegradA} - \text{breddegradB})^2}$$

Oppgave 8 og 9

I disse oppgavene ble det tatt utgangspunkt i lekeplass-datasettet. initApp-funksjonen sender url og funksjon til funksjonen loadData. Når data er lastet, så blir readLekeplasser initiert og parser dataene og legger dataene til i lekeplassArray. Så blir loadList og loadData initiert, loadData med URL til toalettdataene og funksjonen readToaletter. Når loadData har lastet ferdig så initierer den readToaletter. Denne parser dataene og legger dataene inn i toalettArray.

I favorittLekeplass.html er det en select-tag, når man velger favoritt lekeplass fra listen så vil funksjonen sortFavoritt bli initiert. Funksjonen sortFavoritt tar utgangspunkt i valgte lekeplass, så iterer den gjennom listen med toaletter og ser om toalettet er nærmere enn den forrige. Nærmeste toalett blir alltid oppdatert med det toalettet som er nærmest i luftlinje. Den valgte lekeplassen og det toalettet som er nærmest blir begge lagt til favorittArray. Videre blir favorittArray sendt videre til loadTable og initMap som har ansvaret for å vise liste over valgt lekeplass og nærmeste toalett og for å vise toalett og lekeplass på kartet. Kartet her er fleksibel på zoom slik at begge markørene vil vises på kartet. Dette er fordi lekeplassene er fordelt ut over hele Bergen by, mens toalettene bare befinner seg i Bergen sentrum.

Under er en oversikt over den logiske oppbyggingen av ScriptFavoritt.js:



Oppgave 10

I denne oppgaven har jeg valgt å benytte meg av datasett fra NAV's bedriftsundersøkelse som gir en oversikt over etterspørsel etter arbeidskraft basert på yrke og fordelt på fylker. Dette gjelder år 2017 og hvor mange de spurte bedriftene anslår at de trenger å ansette i løpet av året. Mer informasjon om datasettet finnes her:

https://data.norge.no/data/arbeids-og-velferdsetaten-nav/navs-bedriftsunders%C3%B8kelse-ettersp%C3%B8rsel-etter-arbeidskraft

Her er link direkte til JSON-filen:
https://hotell.difi.no/api/json/nav/bedriftsundersokelse/ettersporsel-pr-yrke/2017

Når dokumentet arbeidskraft.html er lastet så initierer det funksjonen loadArbeidskraft, her lastes data fra URL til datasettet og når den er i tilstand 4 og har status OK så initierer den funksjonen readArbeidskraft.

readArbeidskraft parser dataene og legger de til arrayet arbeidskraftArray, og initierer funksjonen loadTable.

loadTable er en funksjon for å sette inn data i html-tabell.

I arbeidskraft.html er det mulig å trykke på en knapp der det står «sorter alfabetisk» og yrkene blir da sortert alfabetisk og listen oppdateres. Den mer avanserte måten å sortere listen er at man kan trykke på kolonnene og man kan da velge om man vil sortere en kolonne etter stigende, synkende og tilbake til utgangspunktet. Trykker man på en kolonne så endrer pilene retning avhengig av om den er synkende eller stigende eller nøytral. Når en kolonne enten er stigende eller synkende, så vil fargen på kolonnen være lyseblå for å markere at sortering er aktiv. Det er mulig å sortere på mer enn én kolonne om gangen. Programmet skal huske hvilken rekkefølge kolonnene er trykket på og om de er stigende eller synkende. Da vil man kunne sortere på mer enn kolonne om gangen samt bestemme hvilken som skal sorteres først og i hvilken retning.

Siden det er problematisk å bruke flexbox på en tabell er det her valgt å bruke overflow: scroll slik at det er lettere å navigere i tabellen med scroll-barer. Avhengig av størrelse på enhet så vil tabellen ha et spesifikt antall piksler til rådighet i høyden, men makshøyde skal aldri overstige mer enn 90% av skjermstørrelsen.

Under er den logiske oppbyggingen av JavaScript-dokumentet arbeidskraftScript.js: