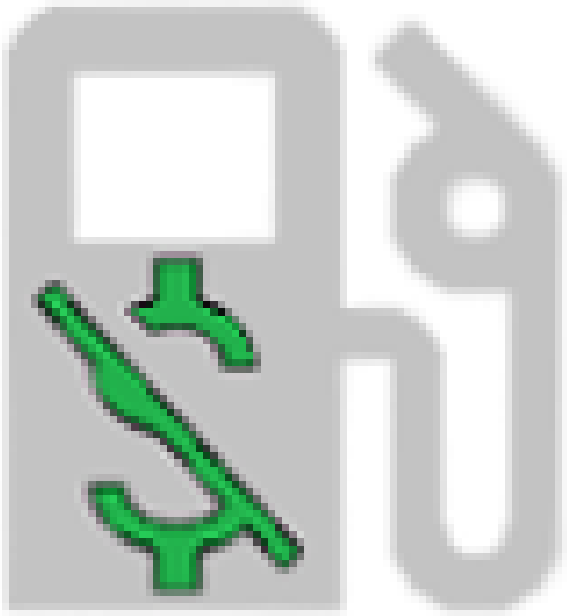
27.11.2015

Obligtorisk oppgave 3

Dokumentasjon «CheapFuel»



Halvor Rønneseth, s172589
ANVENDT DATATEKNOLOGI 2013 - 2016

Innhold

Om applikasjonen

CheapFuel er en applikajson som lister opp bensinstasjoner i brukerens nærområde, samt priser og distanse fra brukerens posisjon hvis GPS er slått på. Applikasjonen sjekker etter oppdateringer hvert minutt fra en ekstern database, og gir beskjed til brukeren om hvilken oppdatering dette er snakk om.

Applikasjonen kan ikke kjøres på emulator siden den snakker med GPS, samt håndterer Google Maps fra nettet.

Brukerinteraksjon

Når brukeren åpner applikasjonen vises alle stasjoner lagret på den eksterne databasen. I toppen vises den stasjonen som er billigst av de som finnes i databasen, uavhengig av posisjon. Hvis brukeren holder lenge på stasjonene i den nederste listen får de beskjed om de vil slette denne stasjonen.

Når brukeren trykker på en av disse blir de tatt med til et kart hvor logoen vises. Her kan brukeren trykke på markøren for å få opp info om stasjonen og pris, og hvis det blir trykket en gang til får man spørsmål om man vil få sin posisjon i forhold til dette punktet.

I menyen vises det to tegn, den ene for «Kart» og den andre for «Innstillinger». I «Innstillinger» kan brukeren velge om han vil sette på GPS eller automatisk oppdatering av stasjoner fra ekstern DB. Brukeren slår på GPS ved å gå ut fra applikasjonen til telefonens egne innstillinger. Dette får bruker beskjed om når de trykker på knappen på «Innstillinger»-akriviteten.

I «Kart»-aktiviteten tas brukeren til et kart hvor alle stasjonene som vises er. Når brukeren trykker på denne får de opp informasjon, og hvis de trykker en gang til får de beskjed om de vil oppdatere pris på den aktuelle stasjonen. Brukeren kan ved å holde lenge på et punkt legge til en ny stasjon. Ved oppstart får brukeren en melding om hva kan gjøres. Når de åpnes opp for første gang, vises en melding med hva som kan gjøres innenfor aktivitetene.

Applikasjonen støtter også engelsk, med norsk som standardspråk.

Innhold i applikasjonen:

- 12 separate javaklasser
    - CheapBroadcast.java
    - DBHandler.java
    - EnkelStasjonKart.java
    - Forside.java
    - Innstillinger.java
    - Kart.java
    - LeggTilStasjon.java
    - ListViewAdapter.java
    - LocationHandler.java
    - Prisoppdatering.java
    - SjekkEksternDatabase.java

  - Stasjon.java
- 8 layoutfiler
  - Alle med støtte for landskap
  - 3 med egne landskapsfiler
  - To layoutfiler for å vise ListView på forsiden.
- 13 ikoner
  - 4 ikoner for stasjoner
  - 7 navigasjonsikoner
  - 3 logoer med forskjellig størrelse

Innhold på ekstern server

4 forskjellige PHP-filer. Disse ligger som vedlegg nederst i dokumentasjonen.

Valg gjort i utviklingen

Kart

I denne applikasjonen har jeg benyttet Google Maps API for å kunne vise stasjoner og brukerens posisjon på nett. Dette API'et har også blitt brukt til å håndtere det som har med posisjon på kartet, samt avstander. Avstandene vist i appen kan kun vises med bruk av GPS. Avstandene som er målt i appen er målt i luftlinje fra brukers markør til den aktuelle stasjonen.

Med tanke på at Google Maps kun er et kart-API, kunne jeg ha brukt Google Location API, Google Routes API og Google Places API for å implementere ruter og hurtigere oppdatering av brukerens lokasjon på kartet. Dette er noe jeg vil se på ved videre utvikling, da jeg i denne applikasjonen fokuserte på et brett aspekt rundt brukerinteraksjon og lagring til database, både internt og eksternt. Jeg valgte derfor å kun fokusere på Google Maps API for å kunne beherske dette på best mulig måte før en eventuell utvidelse med en mer kompleks utførelse.

Databasehåndtering

I denne appen kan brukeren legge til en stasjon ved å holde lenge på et punkt. Stasjonen blir da lagret som et Stasjon-objekt med tilhørende get-og set-metoder. Stasjonen blir da lagret i databasen Stasjoner inne på telefonen, og samtidig lagres den i den eksterne databasen, som finnes på mitt hjemmeområde på HiOA. Databasene har begge id, men jeg velger å bruke navn som primærnøkkel, siden hver stasjon har et tilhørende stedsnavn. Dermed var det mest naturlig å gjøre endringer i databasene ut fra navnet.

Gjennom DBHandler.java foregår all kommunikasjon til begge databasene. Når en stasjon blir slettet, lagt til eller oppdatert, oppdaterer DBHandleren dette fra til den lokale databasen og til den eksterne databasen. Skriving til den eksterne databasen skjer via en privat klasse for hver operasjon som er linket til tre forskjellige PHP-filer med tilhørende kode for databasehåndteringen (vedlegg 1). Disse klassene utvider AsyncTask som opererer på egne tråder, da det å åpne opp en kobling mot nettet på hovedtråden ikke er gunstig.

Kjøring i bakgrunnen

Ved å ha to databaser, en på telefonen og en ekstern, gjør at applikasjonen kan sjekke om en oppdatering har skjedd ved å sammenligne disse to. Applikasjonen har en BroadcastReciever (CheapBroadcast.java) som stares når brukeren setter på knappen for automatisk oppdatering. Denne setter i gang Prisoppdatering.java som en Service som kjører SjekkEksternDatabase.java hvert minutt. Dette skjer helt til brukeren trykker på knappen for å slå av oppdatering i innstillinger.

All oppdatering fra den eksterne databasen skjer i SjekkEksternDatabase.java. Denne klassens sjekker etter en JSON-streng på mitt hjemmeområde, her. Den sender da inn JSON-strengen til onPostExecute() som håndterer strengen og gjør det til Stasjon-objekter som blir lagt i en List<Stasjoner>. Deretter sammenlignes listene fra den eksterne databasen med den databasen som ligger på telefonen med hverandre, og en notifikasjon blir lagt til på telefonens drop-down-meny med hvilke typer oppdateringer som har blitt gjort på siden siste gjennomkjøring, enten en eller flere stasjoner er slettet, oppdaterte pris eller lagt til.

Applikasjonen har også en LocationManager-klasse som med LocationListener lytter etter posisjon når GPS er slått på. Denne koden kjøres i bakgrunnen på forsiden og i EnkelStasjonKart.java for å kunne gi brukeren feedback om at han flytter på seg. Koden for å hente plasseringen blir kjørt hvert 10 sekund, og stopper når brukeren slår av GPS fra instillinger.

Design

Fargene som er brukt i appen er havblå for å gi en beroligende følelse, samt en lysere farge som bakgrunn til listene for å gi en indikasjon på at det er det innholdet som er viktig. Dette står i stil med «Material Design».

Jeg har prøvd å begrense antall ord brukt i appen, men jeg har latt det være noe tekst, for eksempel i AlertDialog-boksenes «Ja/Nei»-knapper i kart-klassene. Dette kan bli gjort noe med i videre utvikling av applikasjonen. Dette har jeg ikke gjort på forsiden ved slette-dialogen, da denne er bygget opp fra grunnen av for å kunne gi sletteknappen en advarsel om hva som skjer når knappen trykkes.

Ikonet skal symbolisere en bensinpumpe med et dollar-tegn med strek over. Dette dollartegnet viser at man kan spare penger på bensin, noe som passer det appen er designet for å kunne gi informasjon om.

Navigering

I appens forside vil man kunne navigere seg til «Innstillinger» og «Kart» ved bruk av menyen i toppen. Denne har beskrivende ikoner for hva de forestiller, og dermed fylles ikke menyen opp med tekst. I begge kart-klassene kan man enkelt navigere seg tilbake til forsiden ved å trykke på tilbakeknappen øverst i venstre hjørne. Dette gjelder også inne på «Innstillinger».

Hvis brukeren trykker på notificationen som dukker opp når det har skjedd en oppdatering, tas de med til Forsiden.

Feilhåntering

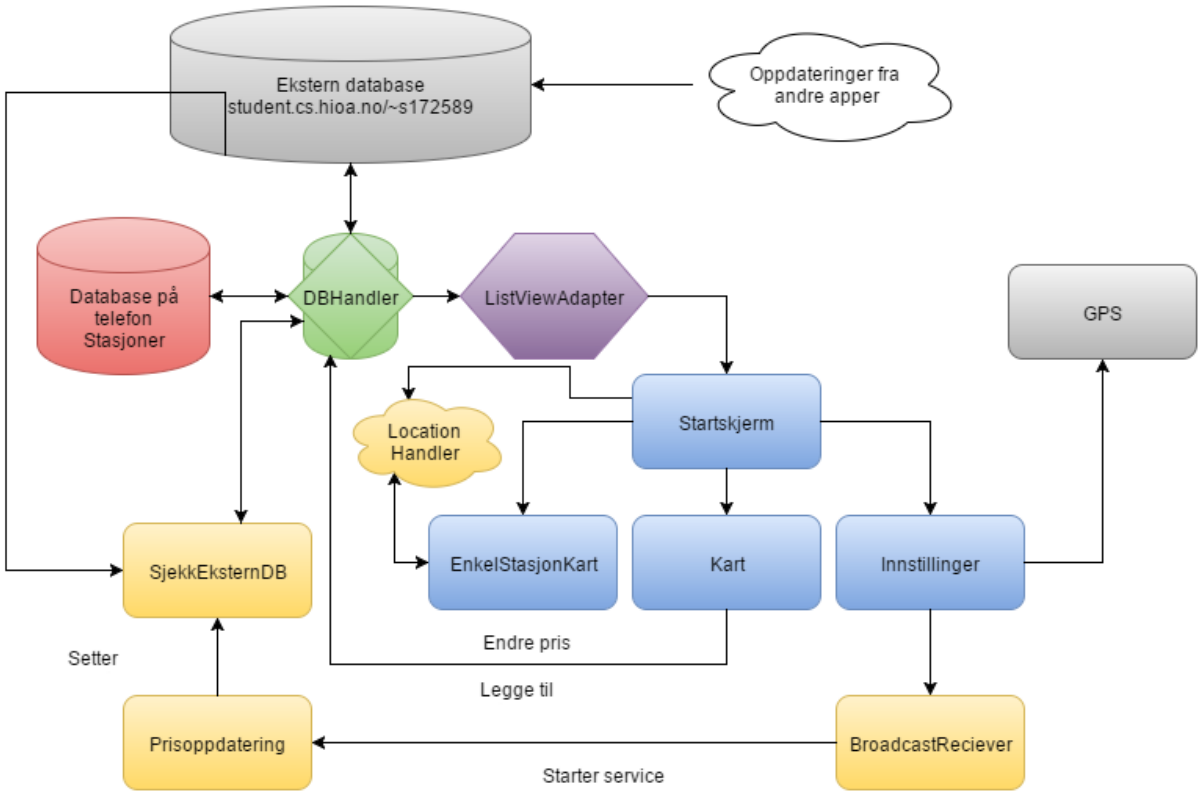
Applikasjonen feilhåndterer samtlige hendelser som måtte skje under bruk, og gir intuitive beskjeder til brukeren når noe eventuelt har gått galt, eller når de gjør noe galt.

Dette kan for eksempel være å legge til en stasjon med samme navn, har ikke internett for å kunne hente oppdateringer, melding når det ikke finnes noe å oppdatere i den eksterne databasen, og generell inputvalidering under opprettelse av ny stasjon og endring i pris.

Videre utvikling

Hvis denne applikasjonen skulle blitt lansert på Google Play, ville jeg vært nødt til å fjerne «Legge til» og «Slette»-funksjonene. Det ville da vært kun mulig å oppdatere pris, samt at å implementere oppdateringen slik at den også registrerer alle stasjoner som finnes i nærområdet, ved bruk av Googles egne API'er.

Programflyt



Vedlegg:

Her ligger alle filene som brukes for håndtering av den eksterne databasen på mitt hjemmeområde.

Cheapfuel_db_leggTilStasjon.php

```
<?php

$server = "student.cs.hioa.no";
$bruker = "s172589";
$passord = "";
$db = "s172589";

$conn = new mysqli($server,$bruker,$passord,$db);

if($conn->connect_error) {
     die("Tilkobling feilet " . $conn->connect_error);
}

$navnInn = $_REQUEST['navn'];
$latInn = $_REQUEST['latitude'];
$lngInn = $_REQUEST['longitude'];
$prisInn = $_REQUEST['pris'];
$ikonInn = $_REQUEST['ikon'];

$navn = (String) $navnInn;
$lat = (String) $latInn;
$lng = (String) $lngInn;
$pris = (String) $prisInn;
$ikon = (int) $ikonInn;


$sql = "INSERT INTO Stasjoner (Navn, Latitude, Longitude,
Pris, Ikon) VALUES ('$navn','$lat','$lng','$pris','$ikon');";

if ($conn->query($sql) === true) {
    return true;
} else {
     return false;
}
mysql_close();
?>
```

Cheapfuel_db_oppdaterPris.php

```
<?php
$server = "student.cs.hioa.no";
$bruker = "s172589";
$passord = "";
$db = "s172589";

$conn = new mysqli($server,$bruker,$passord,$db);

if($conn->connect_error) {
     die("Tilkobling feilet " . $conn->connect_error);
}

$navnInn = $_REQUEST['Navn'];
$prisInn = $_REQUEST['Pris'];

$navn = (String) $navnInn;
$pris = (String) $prisInn;

$sql = "UPDATE Stasjoner SET Pris = '$pris' WHERE Navn =
'$navn'";

if ($conn->query($sql) === true) {
    return true;
} else {
     return false;
}

mysql_close();
?>
```

Cheapfuel_db_slettStasjon.php

```
<?php
$server = "student.cs.hioa.no";
$bruker = "s172589";
$passord = "";
$db = "s172589";

$conn = new mysqli($server,$bruker,$passord,$db);

if($conn->connect_error) {
     die("Tilkobling feilet " . $conn->connect_error);
}

$navnInn = $_REQUEST['Navn'];

$navn = (String) $navnInn;

$sql = "DELETE from Stasjoner where Navn = '$navn'";

if ($conn->query($sql) === true) {
    return true;
} else {
     return false;
}

mysql_close();
?>
```

Cheapfuel_db_finnAlle.php

```
<?php
mysql_connect("student.cs.hioa.no","s172589","");

mysql_select_db("s172589");

$sql = ("select * from Stasjoner");

$tabell = mysql_query($sql);

while($rad = mysql_fetch_assoc($tabell)) {
     $ut[] = $rad;
}

print(json_encode($ut));
mysql_close();

?>
```