Kørselsbog

Contents

Indledning

Der skal oprettes en database for en kørselsdagbog hvor daglig kørsel kan indskrives med fuldt navn, nummerplade, dato og antal kilometer der er kørt.

For denne database skrives et program som kan læse, skrive, ændre og slette i den. Det udføres i Windows Forms og skrives i C#.

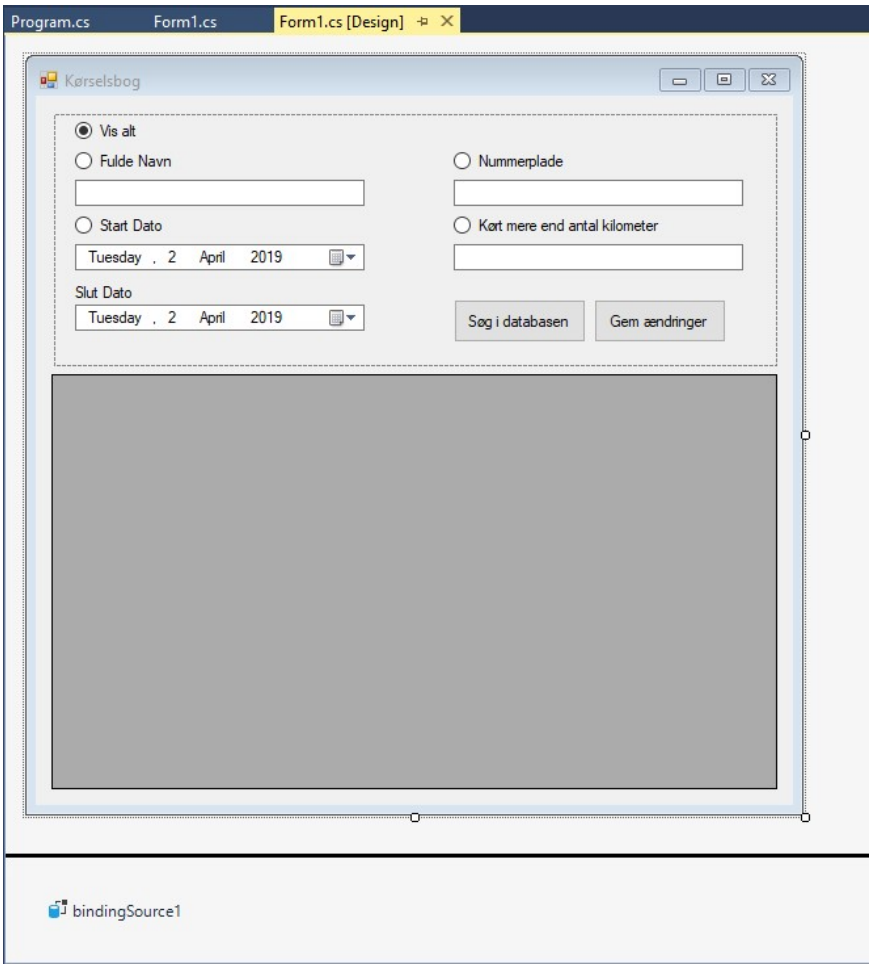
Projekter for Visual Studio og SSMS er inkluderet som bilag i Kørselsbog.zip.

Windows Forms

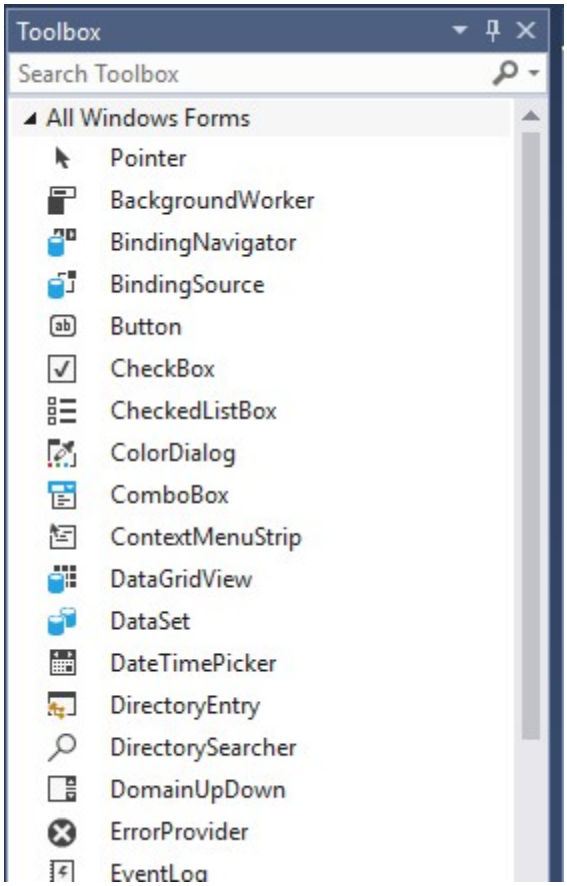
Windows Forms er et grafisk brugerflade system som er indbygget i .NET. Med det kan man bygge applikationer som ser ud som klassiske Windows software som mange kender det.

Windows Forms er lavet af flader som hedder forms hvor man kan placere kontrol objekter så som knapper, tekstbokse, dato og tids vælgere etc.

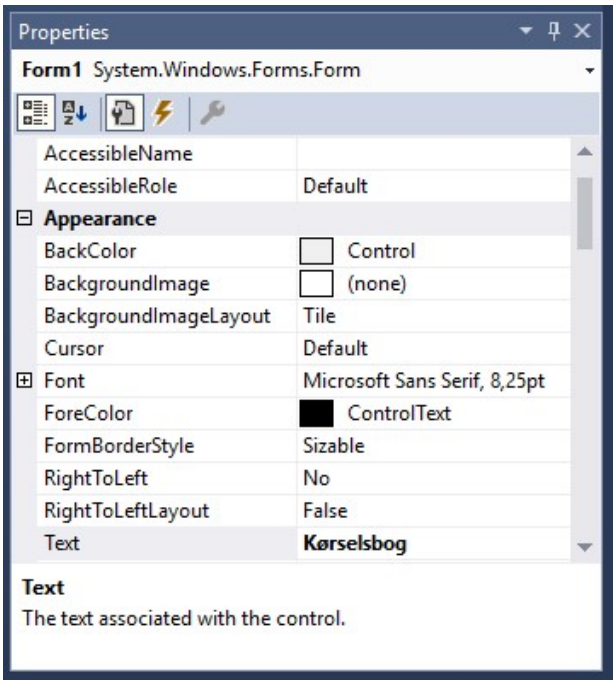
Det ser således ud:



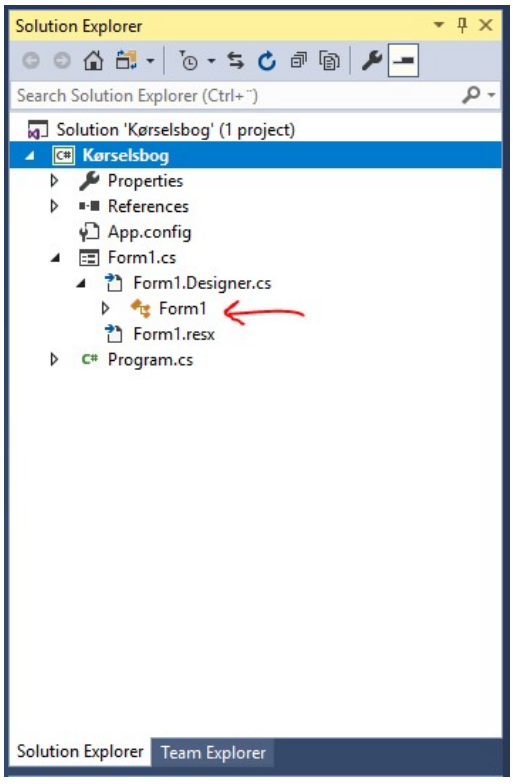
Til venstre i Visual Studio er der en værktøjskasse (Toolbox) som man vælger og trækker kontrol objekter ud i formen .



Nederst til højre i Visual Studio er der properties og events vinduet. Indholdet af den skifter efter hvad man har valgt fokus på i form vinduet.



Der er mange indstillinger som kan sættes herfra. De har indflydelse på hvordan formen vil se ud og hvordan objektet vil optræde og opføre sig i koden. De linjer her som står med fed er en indikation på at man har ændret værdien til noget andet end standard værdien.



Oppe i højre hjørne finder man Solution Explorer.
Koden for formen med alle metoder som hører til vil ligge i ”Form1”.
Hvis man dobbeltklikker på et kontrol objekt vil der oftest genereres en metode automatisk i koden. Den vil linkes automatisk til et event på kontrol objektet og man skal passe på hvis man fjerner metoden uden at fjerne linket.

SQL Database forbindelse

Der bruges to navneområder af .NET til denne funktionalitet. System.Data og System.Data.SqlClient.

I Windows Forms er der et kontroller objekt der hedder DataGridView som kan vise tabeller.
For at få tabellen fra SQL databasen til programmet skal der oprettes en forbindelse til databasen. Et SqlConnection objekt tager en connection string der fortæller den at serveren ligger på maskinen, at der bruges Windows logon for adgang og at den specifikke database hedder ”Kørselsbog” på SQL serveren.

Med SqlComman objekter samler man de SQL kommandoer man vil bruge på databasen. De kan se ud sådan: `fullnameCommand = new SqlCommand("SELECT * FROM MedarbejderKørsel WHERE FuldNavn LIKE @Fullname", con);`
Den laven en forespørgsel på tabellen som hedder MedarbejderKørsel.
@Fullname er en SQL parameter som tilføjes til SQL kommando objektet ved hjælp af et SqlParameter objekt: `SqlParameter fullnameParam = new SqlParameter("@Fullname", SqlDbType.VarChar, 60);`
Parameters er godt at bruge da det forhindrer at tilfældige SQL kommandoer kan indtastes i programmets tekstfelter som så direkte udføres på SQL serveren.

Til at formidle forbindelsen og dataene har vi et SqlDataAdapter objekt. Det bruges til at sende SQL kommandoen til SQL serveren og lægge de tilbagesendte data ind i et DataTable objekt som kan holde en tabel. Dette DataTable objekt sendes til et Windows Forms kontroller objekt som hedder et BindingSource objekt.
Dette BindingSource objekt er blevet bundet til DataGridView objektet når formen indlæstes.
`medarbejderKørselTabel.DataSource = bindingSource1;`
BindingSource formidler data og ændringer mellem DataGridView og SqlDataAdapter.
Når ændringer i tabellen er foretaget og SqlDataAdapterens Update metode bliver kørt bruger den et SqlCommandBuilder objekt til automatisk at sende de nødvendige UPDATE, CREATE og DELETE SQL kommandoer til SQL serveren for at udføre ændringerne i databasen.

SQL søgning og filtrering

Ved forespørgsel af data fra SQL med kommandoen SELECT kan man filtrere på kolonner med kommandoordet WHERE.

Det fulde navn er filtreret med `WHERE FuldNavn LIKE '%Tom%';`
'%Tom%' er hvad parameteren @Fullname bliver til hvis man skriver "Tom" i tekstboksen.
Procent tegnene er wildcard tegn som står for et vilkårligt antal vilkårlige tegn. Det gør at LIKE som prøver at passe de VarChars fra kolonnen med den givne VarChar vil lede hver eneste VarChar fra kolonnen igennem for at se om den er noget sted i den. Det vil sige at for '%Tom%' er 'Tom Khristensen' lige så gyldigt som 'Erik Tomsen' eller 'Patrik Sitom'.

Den samme søgefunktion som med fulde navn er brugt på nummerpladen.

For dato bruges der `WHERE Dato BETWEEN '2019-03-01' AND '2019-03-26';`
Som filtrerer for rækkerne med datoer mellem og inklusive de givne datoer.

For antal kilometer kørt er der brugt en støre end funktion: `WHERE Kilometer > 5;`

Konklusion

Programmet kan søge, rette, oprette og slette i databasen og ville kunne let udvides til at virke med Microsoft SQL servere andre steder.
Jeg har lært mest om SQL forbindelser og hvordan man bruger objekt orienteret programmering til at facilitere dem. Windows Forms har jeg tidligere arbejdet med men synes stadigvæk det er fedt at arbejde med i Visual Studio.
Denne opgave føltes som en rigtig opgave mere end en øvelse da jeg havde stor frihed til selv at udvikle hvordan løsningen skulle virke.

Dagbog

Dag 1 25/03

Installerede Microsoft SQL og SSMS.

Dag 2 28/03

Oprettede SQL queries og startede Windows Forms applikation.

Dag 3 01/04

Fik læst en hel del dokumentation på C# SQL, DataGridView form control og SQL filtre.

Jeg har database connection indlejret i applikationen.

Dag 4 02/04

Startede skrive rapporten.

Fik implementeret søge funktioner med SQL parametre.

Programmet kan hvad den skal kunne nu.

Dag 5 03/04

Fixede en fejl med Convert i kilometer tekstboksen.

Skrev rapport færdig.