7. Prototype

Prosjektet vårt har blitt laget ved bruk av Java forms. Vi lagde den på plattformen IntelliJ IDEA. Java Forms gir brukeren en ganske lett GUI som man kan sette opp sitt program i. Hvis man vil jobbe videre med protoypen, eller vil bare lese kildekoden så kan man lett gjøre det ved bruk av IntelliJ. For å utføre testing så brukte vi «dependenciene»; JUnit Jupiter API (scope: test, version 5.7.2), JUnit Jupiter engine (scope: test, version 5.7.2), JUnit Jupiter params (scope: test, version 5.7.2) og til slutt maven-failsafe-plugin (versjon 2.22.2). Vi brukte også en siste dependancy kalt gson. Denne lot oss serialisere og deserialisere objekter til JSON så vi kunne skrive/lese til/fra fil. IntelliJ kan bli nedlastet her: <https://www.jetbrains.com/idea/download>.

7.1 Veiledning for å åpne prosjektet (direkte metode)

Trykk «Open» på IntelliJ idea:



Finn frem prosjektmappen (kalt SEG):

Et bilde som inneholder tekst

Automatisk generert beskrivelse

Så lett som det så skal det være mulig å se på kildekoden/jobbe videre med protoypen:

Et bilde som inneholder tekst

Automatisk generert beskrivelse

7.2 Veiledning for å sette opp Maven (i tilfelle 7.1 ikke fungerer)

Trykk «New Project» på IntelliJ.

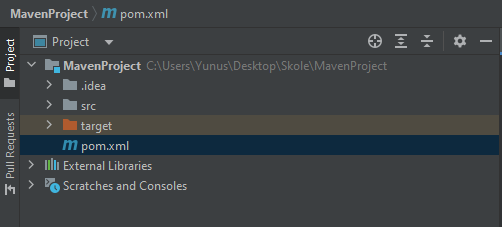


Her så skal du velge egendefinert navn og lokasjon. Deretter velger du Java som språk, og Maven som build system. Behold versjon 17 på openjdk. Trykk create når ferdig.

Et bilde som inneholder tekst

Automatisk generert beskrivelse

Nå skal du over til pom.xml som du kan finne på listen til siden.



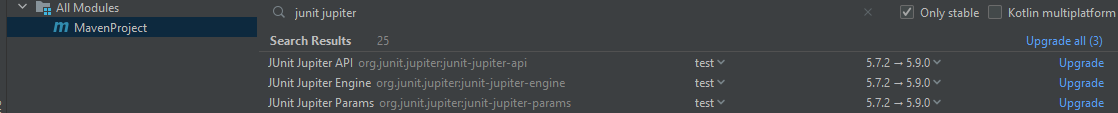
Du kan enten kopiere koden under og build prosjektet, eller manuelt legge til dependencies. Koden ligger på side x.

Trykk dependencies som finnes nederst i vinduet. Her søker du på «junit jupiter».

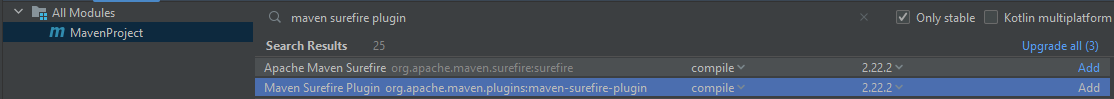
Et bilde som inneholder tekst, elektronikk, datamaskin, skjermbilde

Automatisk generert beskrivelse

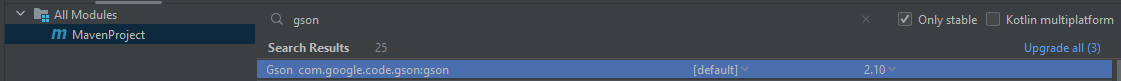
Bytt fra compile til test på; engine, api og params. Du skal i tillegg sette alle 3 versjonene til 5.7.2. Deretter kan du trykke add på hver av dem:



Nå søker du etter maven surefire plugin og legger til den:



Til slutt så søker du etter gson og legger til den:



Nå er alt klart. Hvis gson for en eller annen grunn ikke fungerer (slik som det gjorde med oss på starten), så kan du fikse det ved å gå over til «maven» helt til høyre i vinduet og trykke på «reload» knappen:

Et bilde som inneholder tekst

Automatisk generert beskrivelse

Protoypekoden er delt opp i 2 hovedmapper for kildekoden, og 2 egne javafiler for testing. Det var tidligere en egen mappe for Interfaces, som vi prøvde å sette opp funksjoner som vi kunne kalle til for å skrive til fil/lese fra fil. Dette fungerte dessverre ikke slik som vi ville, og har derfor blitt fjernet i sluttversjonen av protoypen. Endringene hvor vi brukte FileHandler og slikt kan finnes i git endringene.

Et bilde som inneholder tekst

Automatisk generert beskrivelse

I forms mappen så kan vi finne både forms filen som ble brukt til å designe GUI-en. I tilegg så er javafilen som kontroller forms i samme sted. Denne javafilen kontrollerer alt fra hva som blir vist, når det blir vist og hva som skjer når vi trykker på knapper og samhandler med protoypen.

I modules mappen så kan vi se begge klassene som har blitt brukt; CarRegistration og CarRepository. Først så lages et CarRepository som blir brukt til å lagre bilene som blir laget ved bruk av CarRegistration.

