Teori

Oppgave 1.1 - Ord og begreper

Lag deg en oversikt over hva følgende ord/begreper/teknologier betyr/er:

- **Javalin**

Javalin er et web-framework. Som lar deg skrive HTML, CSS og javascript front-end og Java back-end. Javalin starter en lokal webserver og host for HTML-sider. HTML sidene kan bruke java til å lese og skrive data. Som webserver bruker Javalin jetty.

- **Vue.js**

  Vue.js er en javascript framework som er lagd for og lage netsider. Ved å kombinere med Javalin kan du få en enklere syntaks fordi Vue har andre versjoner av Javalin-kode. Men Vue hoster og behandler HTML-filer på en annen måte. Det betyr at når man skal lage front-end kan man bruke mindre tid på å gjøre ting.

- **Anonym Klasse**

  Anonym klasse bruker man når du lager en klasse, men klassen trenger en underklasse for å fungere.

  kode eksempel:

  PlanetSystem planetSystem = new PlanetSystem(Name, New planet(Name))

  Her vil du lage et planets system som inneholder et navn og en planet. Denne nye planeten er annonym fordi man ikke gir den et navn som man vanligvis ville ha gjort.

  (Planet planet = new Planet(name))

  Dette gjør det lettere å lage klasser som trenger underklasser.

- **MVC (konseptet, og hver enkelt del)**

  MVC (Model View Controller) er et konsept som skiller front-end og back-end. Fordelen med og bruke MVC er at du kan endere front-end og back-end uten og påvirke andre delen. Om man endrer informasjonen i back-end, Så lenge man fyller kravene controlleren vil ha, trenger man ikke og endre noe i front-end.

  - **Model**

  Her er det meste av logikken utført. Det er Java delen. Controlleren henter informasjon herfra for å oppdatere UI(User Interface/Brukergrensesnitt). Kort oppsumert: holder på data.

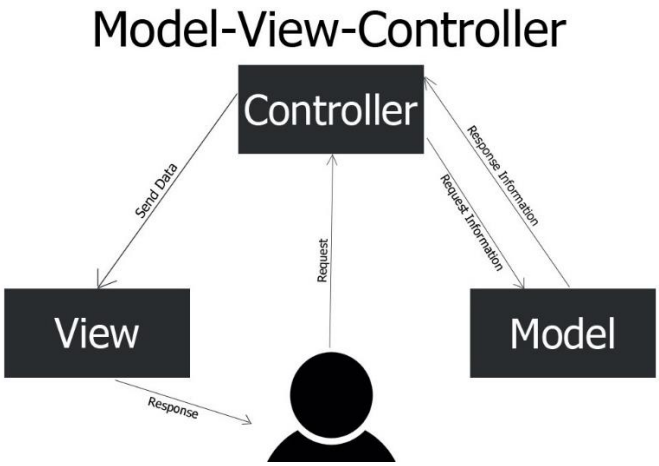
- **View**

  Dette er hva brukeren ser og hva som er relevant for brukeren. Det er HTML, CSS og Java. Controlleren oppdaterer informasjonen her basert på informasjonen hentet fra Model. Kort oppsumert: viser fram data

- **Controller**

  Dette er all logikken om hva som blir hentet fra Model og View. Hvis noen trykker på en knapp i Model, vil en funksjon i Controll aktivert. Og den kjører en eller flere funksjoner i Model. Model vil gjøre noe of deretter sende informasjon tilbake til Controlleren. Controlleren behandler informasjonen og oppdaterer View. Controlleren har " tilgang" til " alle" klassene i Model. Kort oppsumert: flytter på data.

  Visuel visning:



Oppgave 1.2 – Kodesammenligning

Jeg sammenligner med løsninsforsalget.

Sammenligning av CelestialBody class-en.

Jeg ser i løsningsforsalget at Gravitational constanten ble sat som en public static final double i CelestialBody og er sat to public abstract double funksjoner for getMassInKg og getRadiusKm. Dette er noe jeg ikke har i min CelestialBody. Kort sakt (jeg mangler anonyme klasser i CelestialBody).

hente ut en planet fra et PlanetSystem basert på navn

Her er det ganske likt uten om vi har forskjellige metode navn. Jeg satte min som getPlanetByName og i løsningsforslaget er den satt som getPlanet. i sammenlingningen er det satt en equalsIgnoreCase i String classen imens jeg hadde satt dobble er lik == istedenfor. Til neste gang brude jeg heller bruke .equals i stedet for ==. For å sammenlingne to strenger. For de andre oppgavene mangler jeg en del og har endel feil i utregings oppgavene.