

Dagbok INF219

Dette er en dagbok om mitt INF219 prosjekt, [HandcraftObjectsTool](#), som ønsker å gjøre det lett å lage serialiserte objekter i JVM med et lett å bruke brukergrensesnitt. Mentor er Erlend Raa Vågset. Dagboken vil handle om alt som skjer rundt prosjektet, inkludert hva jeg har gjort i prosjektet på en viss dag, møter med Erlend og annen informasjon jeg har fått om prosjektet.

08.01.2020 (ingen kode, 1-ish time bruk)

Møte med mentor Erlend om prosjektet. Han godtok prosjektet mitt, men må bare snakke med administrasjonen for å være 100% sikker.

09.01.2020 (*14df6c*, 6-ish timer brukt)

Planen for idag er å starte på selve prosjektet. Noen mindre sub-goals er

1. At GULET skal kompileres (men ikke nødvendigvis funke 100%)
 - Det kompilerer, men den biten som leser inne YAML er bort pga ConfigSection ikke er en ting utenfor bukkit
2. Finne ut av hva som skal gjøres først
 - Eksempelet prosjektet skal gjøres først

Prosjektet har nå fått en struktur som jeg er fornøyd med, jeg tror det var lurt å skille APIet fra selve GULET ettersom da trenger andre ikke å bry seg om GULET.

Eksempelet går ut på at man skal ha en dialog med forskjellige alternative svar/spørsmål man kan stille underveis. For å lage slike dialoger raskt, enkelt og riktig er et verktøy som HOT perfekt (eller det vil bli det :P)

10.01.2020 (*606dab*, 1-ish timer brukt)

Planen var å fortsette på eksempel prosjektet slik at jeg har noe å bruke APIet på, det er jo ganske vanskelig å lage et API uten å vite hvordan det skal brukes.

Fikk ikke gjort mye, men fortsatte på eksempel prosjektet. Se [commit 606dab](#) for det som ble gjort i dag.

11.01.2020 (*df8e5c*, 8-ish timer brukt)

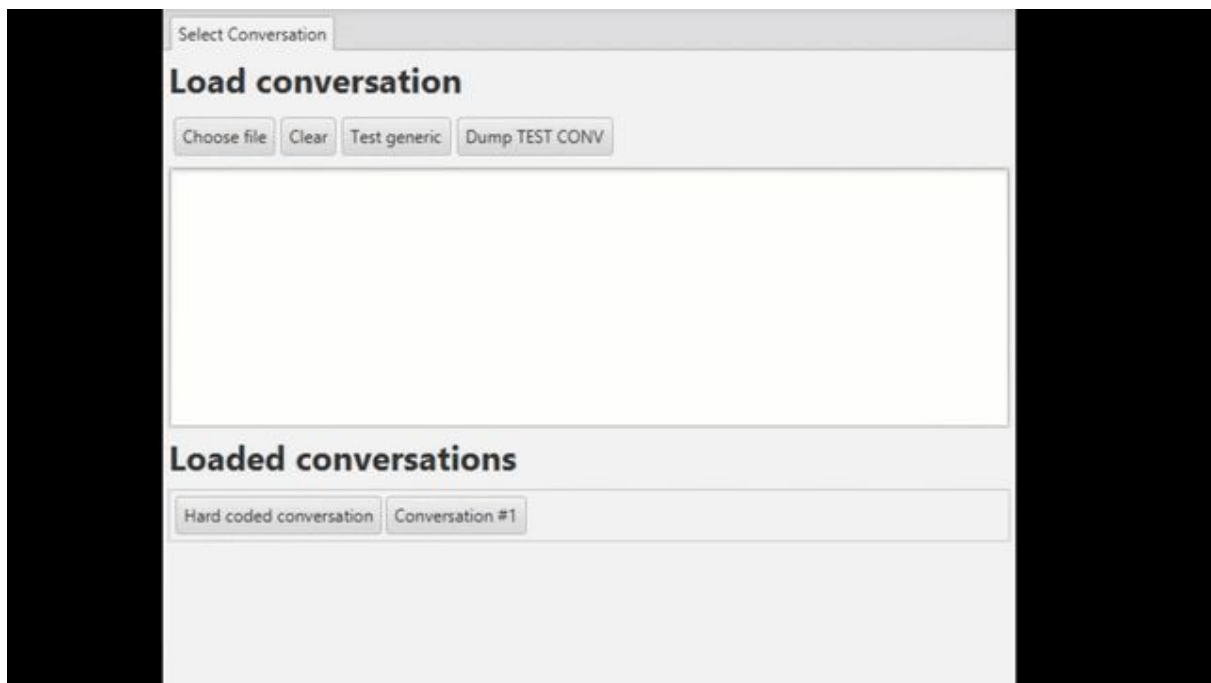
Gjorde ferdig samtale GUIen, og gjorde det mulig å velge flere forskjellige lastet inn samtaler. Nå er alt klart for å begynne å laget APIet som skal sørge for serialisering av objekter (ie api modulen)

Jeg følers også litt bedre å skrive i kotlin, det er fortsatt ikke så smooth som det å skrive java men jeg ser virkelig hvor mye bedre kotlin er for GUI programmering.

12.01.2020 (*øfaa87, 7-ish timer brukt*)

Idag har jeg fått eksempelet applikasjonen til å load inn YAML filer til Conversations. Jeg bruker fortsatt snake YAML, men har tatt [konfigurasjons koden til bukkit](#) for å gjøre det lettere å håndtere yaml. Bukkit har GLP lisens v3 så det ser ut som prosjektet også må ha denne lisensen, men dette må undersøkes nærmere.

Brukeren vil ikke vite at de bruker bukkit config for den tenkte måten å bruke prosjektet på er å bruke [SerializationManager](#). De må fortsatt indirekte utvide [ConfigurationSerializable](#) klassen, men det er det eneste. Her er en liten gif som viser hvordan den ser ut nå



Men legg merke til at rekursjonen som foregår fungerer bare for objekter som er laget i programmatisk, ikke lastet fra disk (det er ikke en feature fra bukkit sin resolver).

13.01.2020 (*20a7cd, 20-ish min*)

Fikk ikke gjort mye idag, prøvde å fikse loading av objekter men fikk det ikke til. Endret bare tilslutt readme-en litt.

Jeg fikk også melding av Erlend om hvordan vurdering blir i faget. Det jeg vet er at jeg må skrive en rapport, som mest sannsynlig blir basert på readme-en og ha en muntlig presentasjon foran mentor og en intern sensor.

14.01.2020 (*e77d4e*, *7-ish timer brukt*)

- 0900 - 1200
- 1600 - 1700
- 1700 - 2000

Jeg har innsett at bukkitt sin konfigurasjon ikke egentlig er nødvendig. Grunnen til at jeg brukte den i starten var fordi jeg allerede visste hvordan man skal bruke den. Men etter at jeg lærte hvordan man skal gjøre det blir det enklere å bruke denne metoden uten noen annet bibliotek, enn snake YAML.

Det gjør det også mulig å deserialisere objekter ved å referere til deres unike id (UUID). Det beste hadde vært å legge dem til som referanser (altså hvordan YAML håndterer rekursjon)

15.01.2020 (*ingen commits, men 3-ish timer brukt, inkl. møte*)

Ikke mye koding ble gjort i dag, problemet var at jeg enda ikke vet hvordan skal se ut. Jeg vil at de som bruker interfacet ikke må bruke snake yaml

Jeg hadde også møte med Erlend, som sa at det måtte være en rapport og en presentasjon (basert på rapporten). Han ga meg også en mal jeg kan følge, men rapporten må ikke være akkurat som den. Jeg fortalte han også om hva jeg har gjort (se innlegg over) og han var enig at det var lurt å lage et eksempel prosjekt ettersom det gjør det enklere for han å forstå prosjektet, men han var bekymret for at det ville gjøre prosjektet for spesifikt.

16.01.2020 (*68d7ee*, *7,5t*)

- 11-18.30

I dag fant jeg ut at det [jackson](#) passer perfekt til prosjektet. Den støtter de fleste serialiserbare formater, som YAML, csv, properties, json etc. Den har også mange annotations som kan brukes i GUIet. under the hood så bruker den også snake yaml (og ikke snake yaml engine, pass på å ikke oppdater til 3.x) som gjør den kompatibel med mitt hovedbruksområde. Den gjør det også overflødig å ha et Serializable interface ettersom alt slik blir gjort med jackson. Den informasjon som guiet nå trenger er ikke nødvendigvis teknisk informasjon om typen, men heller informasjon om hvordan hver komponent skal bli brukt.

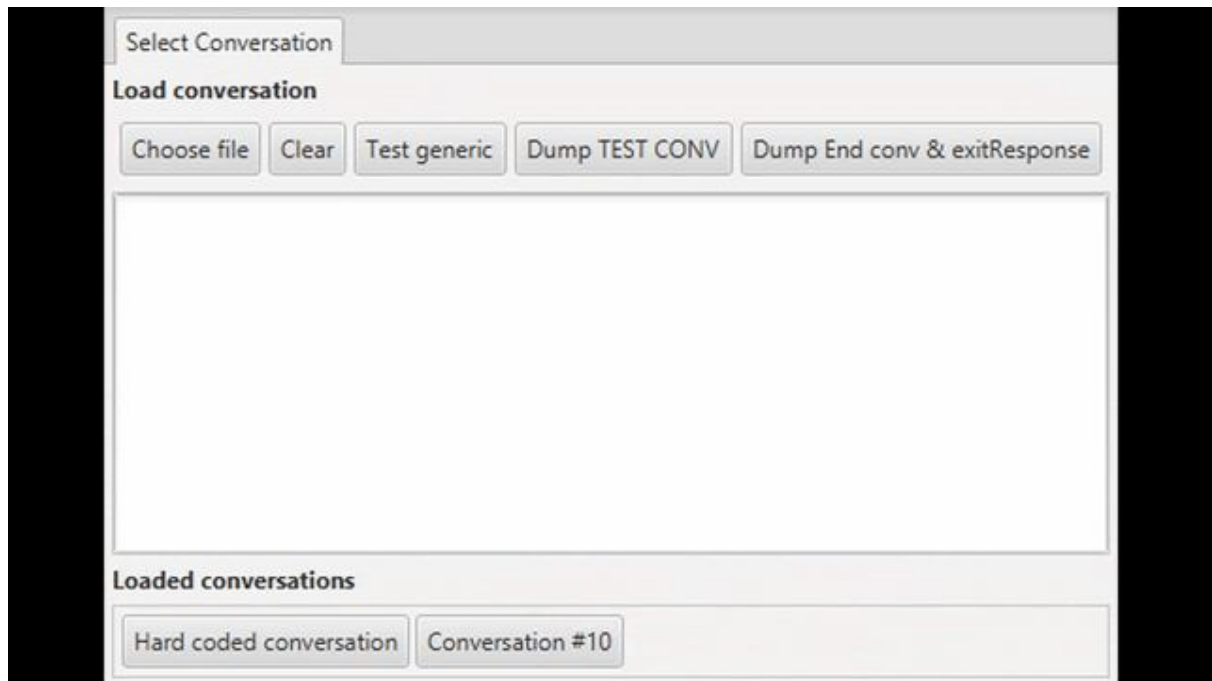
Resten av dagen (etter å ha fått jackson til å funke) ble brukt til å prøve å få rekursjon til å funke igjen, men dette mislyktes.

17.01.2020 (*587d5a*, *5-ish timer*)

- 0930-1530

Jeg fikk til rekursjon! Helt ærlig så vet jeg ikke akkurat hva jeg gjorde feil, men når jeg fjernet visse annotations så plutselig fungket det. I en stund fungket det bare for json og ikke yaml, men etter å fjerne flere annotations så fungket det også.

Svar (responses) kan også nå ha prerequisites (forutsetninger) for å svare på en viss måte. Jeg vil si at alt er klart til å begynne med å lage GULET.



18.01.2020 (*13b50c*, 4 timer)

- 1500-1900

Jeg begynte på hoved GULET i dag. Det blir litt likt som eksempelet, en tabpane med en hoved tab hvor det er en rekke alternativer, som å last inn jar filer etc.

19.01.2020 (*4697e7*, 4-ish timer)

Forbedret hvordan man laster inn klasser fra en jar fil. For hver fil man laster inn leses alle filene inne i jaren (husk at det egentlig er en slags zip) og lager et map med veien (folder0.folder1.fil-format) som nøkkel og en lazy klasse (ie ikke initialisert før brukt første gang) som verdi

20.01.2020 (*fd2639*, 4-ish timer)

- 11-14
- 16-17

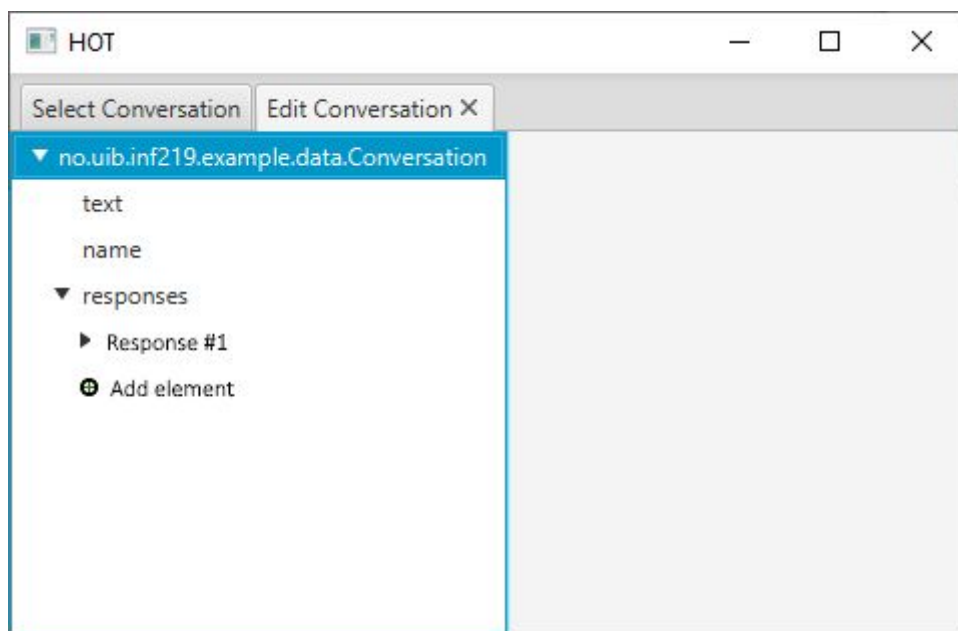
I dag utforsket jeg hvordan man skal få informasjon om objekter fra jackson. Jackson må på en eller annen måte vite hvordan man skal konstruere objekter fra strenger, det er på denne informasjon jeg vil basere hvordan man skal lage nye objekter i guiet.

En annen ting jeg fant ut idag er at det ikke er trivielt å bare velge en random jar fil og prøve å kjøre den. Hvis filen ikke inneholder alle dependensiene blir en `NoDefClassException` kastet ettersom classloader ikke finner klassene som den inn-lastede klassen er avhengig av

21.01.2020 (52c917, 2,5t)

- 10.30-12.40
- 1840-1900

Jeg begynte å gjenskape demo-prosjektet sine features. Man kan nå se alle egenskapene som rot klassen har. Jeg er fortsatt usikker på hvordan det skal funke i bakgrunnen, ie tilbake til serialisert form. Jeg har et konsept på hvordan man skal kunne legge til elementer i egenskaper som er collection-like, se mockup under.



Et annet problem jeg må finne en god løsning på er hvordan underklasser skal velges. Hvis det er spesifisert et interface i klassen som en egenskap, så må det være mulig å velge hvilken underklasse som skal lages. Det går an med jackson å utvide klasser dynamisk, men jeg vil helst unngå det ettersom klienter ikke nødvendigvis klarer å lese inn slike klasser. Dette problemet er forøvrigt for alle klasser som har underklasser. Selv den klassen man velger å lage (altså rot klassen) kan være et interface.

En tredje utfordring er hvordan selve redigeringen av egenskaper skal se ut. Gjenbrukes brukergrensesnittet på demoen, eller skal den se helt annerledes ut? Det er noen elementer som må være på plass:

- En måte å redigere verdien(e) til elementet

- Informasjon om java typen til elementet
- Informasjon som utvikleren vil at du skal se (ie en beskrivelse av hva elementet blir brukt til)

Jeg ønsker nok å lage en oop måte å lage grensesnitt for ulike klasser. Før jeg gjør dette må jeg svare på de to andre spørsmålene over.

22.01.2020 (??, 3,5t)

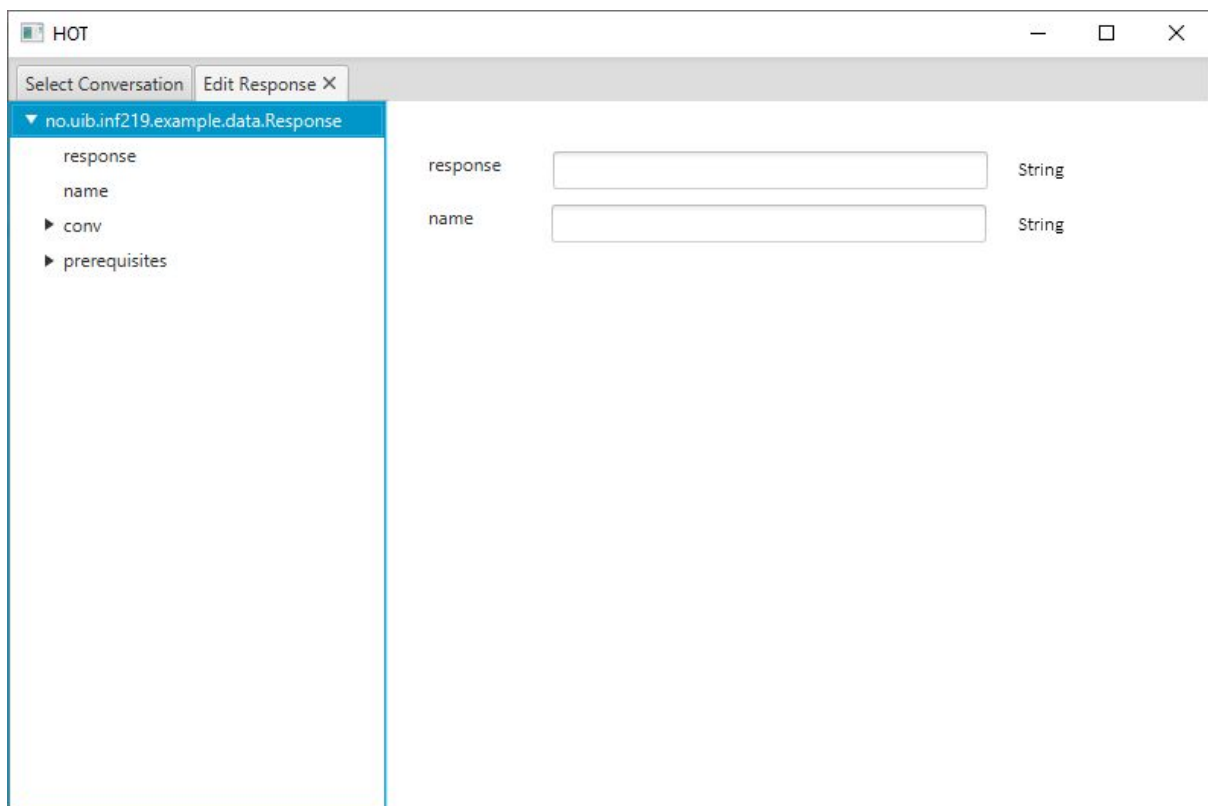
- 1030-1400

Mye av tiden i dag ble brukt til å finne svar til spørsmålene stilt i går. Det viser seg at jackson har en metode i sin `ObjectMapper` klasse som kan konvertere et `Map<String, Any>` til hvilken som helst type. Dette kan brukes til å lagre verdiene objekter vi definerer. Nå må dette bare settes sammen til en form som er lett å verdier endre i.

23.01.2020 (??, 6t)

- 1000-1400
- 1600-1800

Under er et mockup på hvordan redigering av enkle (ie primitive) egenskaper kan se ut.



The screenshot shows a web application window titled "HOT". It has two tabs: "Select Conversation" and "Edit Response X". The "Edit Response X" tab is active. On the left, there is a tree view showing a hierarchy of objects. The selected object is `no.uib.inf219.example.data.Response`. Under this object, there are three properties: `response`, `name`, and `conv`. The `conv` property is expanded, showing a sub-property `prerequisites`. On the right, there is a form for editing the selected object. It has two input fields: one for `response` and one for `name`. Both fields are currently empty. To the right of each input field, the data type is indicated as `String`.

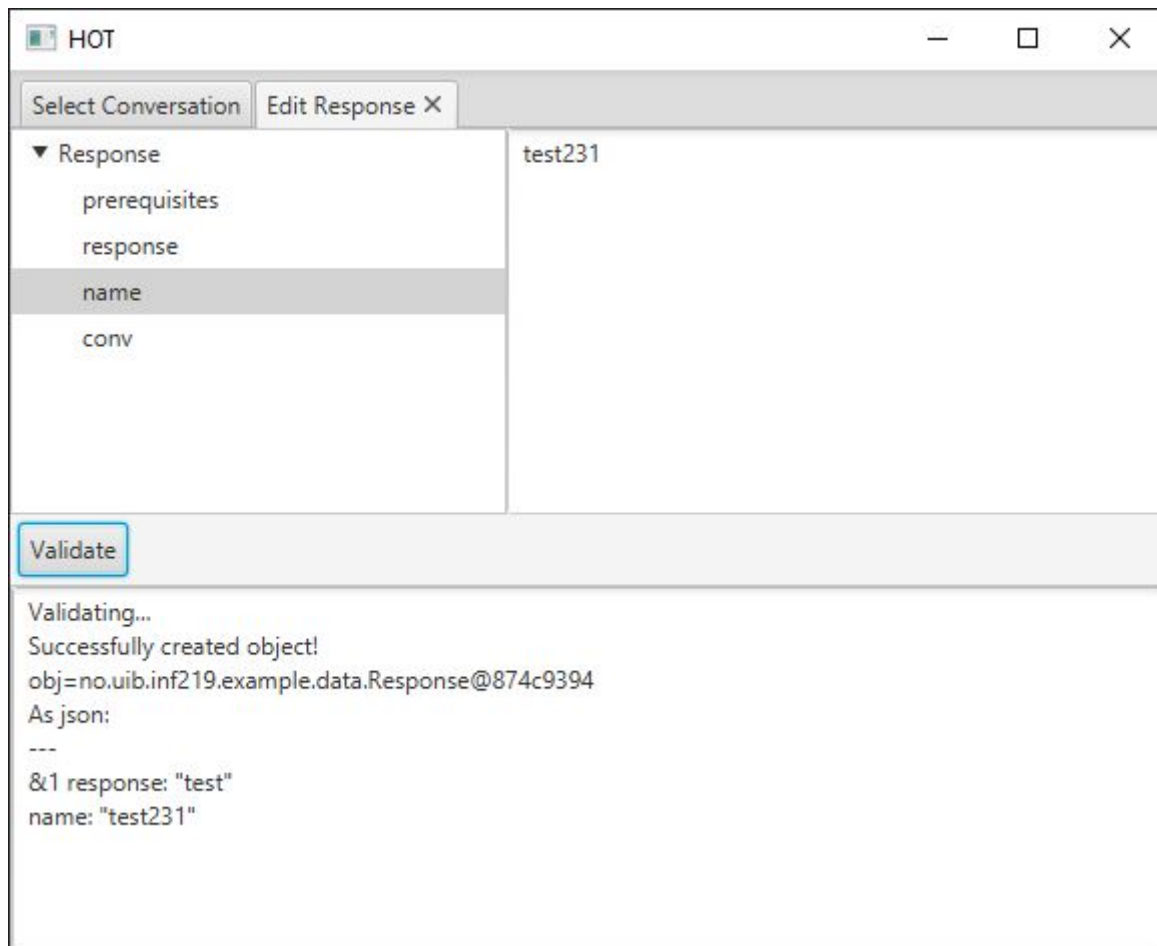
Jeg hadde et møte med Erlend i dag. Det gikk stort sett ut på at jeg oppsummerte de siste dagene med arbeid, viste han litt konsepter (se bilde over) og listet opp utfordringene jeg står overfor akkurat nå.

- Hvordan lagre objektene internt mens de blir laget.
 - Det finnes uncommitted kode (når dette ble skrevet) bruker man en klasse som har et map med nøkler av strenger og verdier av Any?. Denne klassen har en funksjon som konverterer mappet til det ønskede objektet. I fremtiden så burde det bare være lov med nøkler som klassen *faktisk* kan ha (ie en Namable kan bare har "name" som nøkkel).
- Hvordan referanser (spesielt rekursive) skal funke
 - Her ble vi enige om at man må ha en "id" på de objektene som lages, fra et brukergrensesnitt synspunkt er det kanskje best å ha et vindu hvor du kan velge en instans som har blitt laget.
- Hvordan håndtere samlinger (collections) av objekter
 - Se konsept fra 21.01
 - Men hvordan funker dette med første punkt?
- Hvordan man skal redigere objektene visuelt
 - Skal hver egenskap ha sitt eget vindu hvor du redigerer det elementet?
 - Se konsept fra idag
- Hvordan man skal håndtere subklasser av objekter?
 - Et vindu hvor man kan velge alle underklasser av en grunn klasse/interface?
 - I så fall bruk classgraph
- Hvordan på en skikkelig måte få informasjonen fra jackson. Det jeg bruker nå virker litt midlertidig.
- Hvordan la brukeren definere ObjectMapper

28.01.2020 (09944f, 9t)

- 0930-1430
- 1500-1600
- 2230-0230 (29.01)

Nå begynner andre fag å få ting å gjøre så ikke all tiden min vil lengre bli brukt på INF219. Jeg har i dag (og litt 23.) funnet ut hvordan objektene skal bli lagret i bakgrunnen. En `ClassBuilder` vet hvordan man skal lage selve objektet, når vi snakker om klasser med flere felter (altså vanlige klasser) så bruker man `Map`. For primitive verdier så har de sin egen klasse i `ClassBuilder` støtter hver sin primitive type, lister (eller collection-like) er et spesielt tilfelle som enda ikke har blitt løst.



Litt senere (altså kl 2230) fortsatte jeg med koble sammen hvordan man lagrer objektene til hvordan man redigerer dem. Til slutt fikk jeg til en måte å oppdatere ClassBuilder til simple objekter. Det blir enda ikke laget sub objekter når man dobbeltklikker på et komplekst objekt men de primitive fungerer i det minste. Nå gjenstår det bare å utvide det jeg allerede har laget. I første omgang blir det nok å fikse slik at man kan redigere komplekse objekter som ikke har noe med

29.01.2020 (3dff6a, 5t)

- 1100-1500
- 2130-2230

Jeg fortsatte med å utvide guiet, komplekse objekter (ie vanlige klasser) kan nå bli laget, egenskaper som har definert en standardverdi vil bruke den og andre småting. Stort sett har denne dagen gått til å prøve å få programmet brukbarhet. To av spørsmålene som jeg stilte 23. har fått et svar men det er vesentlige elementer som liste og kart håndtering, hvordan man skal kunne velge subklasser av abstrakte (ink. grensesnitt) klasser er ikke løst enda men det er nærmere i det minste.

Fra i dag har jeg loggført rundt 84 timer på INF219, fra en [årsrapporten for 2017](#) bruke en gjennomsnittlig mat-nat student totalt 35,2 timer i selvstudie dette er stort sett delt på tre fag så for et fag kan vi anta at en gjennomsnittlig student bruker 11,7 timer per fag per uke. Hvis

man antar at et semester er på 21 uker (i mitt tilfelle uke 2 til uke 23) så er den forventede antall timer på 246,4 timer. Jeg har, etter bare tre uker, rundt 34% av tids budsjettet mitt på dette faget når jeg egentlig burde ligge på rundt 35,2 timer eller 14% av tidsbudsjettet. I forhold til hvor hvor mange av funksjonene som nå funker vil jeg si at jeg ligger veldig godt an.

31.01.2020 (ingen kommit)

Begynte på liste håndtering, har en fungerende-ish backend men kan enda ikke legge til elementer i GUlet

01.02.2020 (dcb9e4, 3t)

- 0930-1230

Guiet funker nå bedre når man endrer størrelsen på vinduet.

Collections (i hvert fall av typen lister) er nå støttet. Guiet for lister er et rekursivt av hoved guiet, vet ikke om dette er en grei måte å gjøre det på, det gjør det vanskelig å skrive objekter med mange lister i seg så det må nok endres. En ide er at det ikke er et fastsatt gui men heller et plan hvor man kan dra elementene rundt som ark på et bord.

02.02.2020 (c2a9e4, 2t)

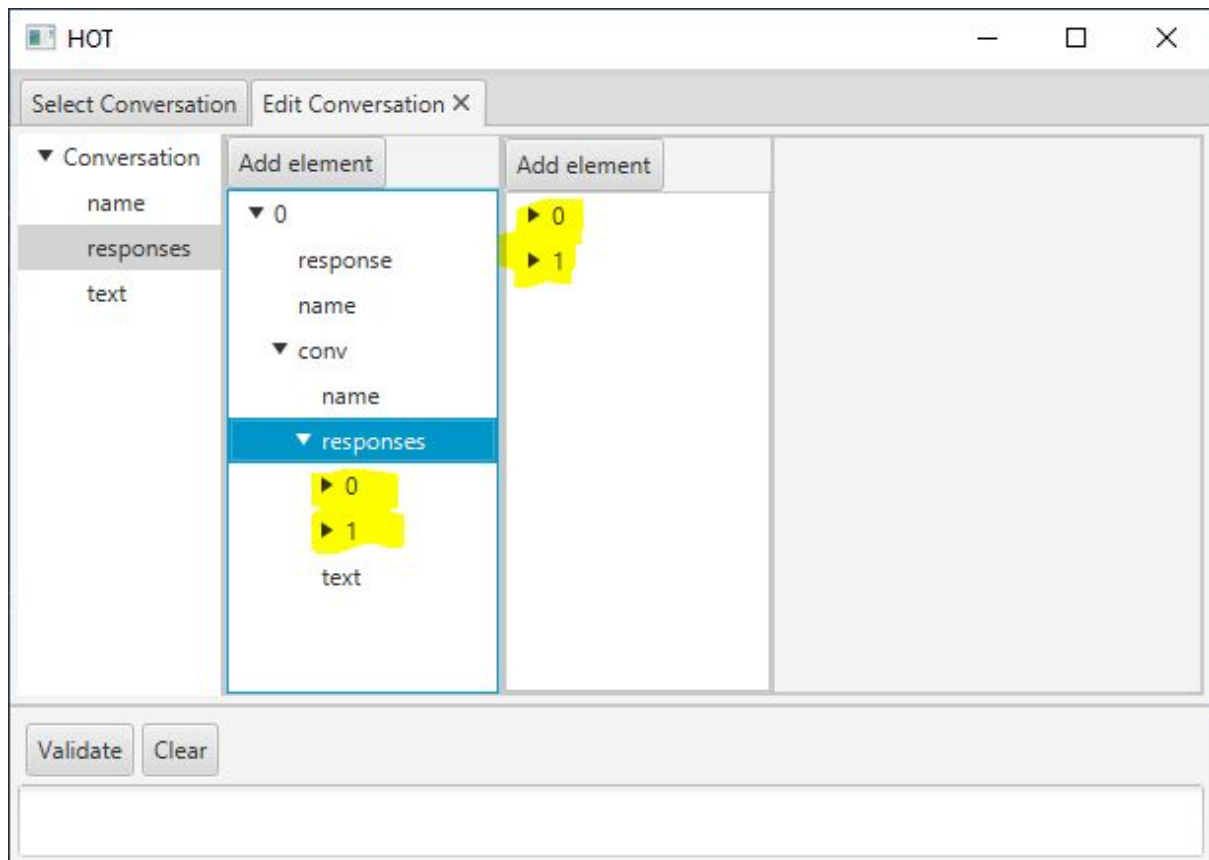
- 10.30-12.30

Stort sett bare grafiske endringer i dag. Fant ut a splitpane er en fin måte å skille elementer når man redigerer objekter. Enkle objekter viser nå også typen som er forventet og om det obligatoriske. Collection objekter vil også vise de nye objektene når de blir laget.

03.02.2020 (11f34f, 3t)

- 11-14 (ink møte 13-14)

Fikset det visuelle med lister og endret hvordan reset fungerer. Hadde møte med Erlend, kom fram til at jeg skal ikke legge til ny funksjonalitet denne uken men heller forbedre og teste den som nå finnes. Under møte fant jeg også en bug-featurer! Hvis man har rekursiv struktur med en liste vil liste elementene ikke bare vises i redigeringsvinduet men også i node strukturen. Se bilde under



04.02.2020 (a1d259, 4t)

- 10.30-12
- 22.30-0100

Refactor SerializationManager to only hold default values
Close all tabs when changing object mapper

05.02.2020

- 0930-1230

Add central view to handle logging. Den funker asynkront, hvordan vet jeg det? Jeg brukte en auto klikker som klikket hvert 100 ms og den kræsjet ikke. Generelt oppdaterte gradle til 6.1.1 og gjorde den klar for 7.0

11.02.2020

- 10-ish til 11-ish

14.02.2020

Rename std to json

18.02.2020

- 1000-1330

Jeg har hatt lite motivasjon i det siste. I dag tvang jeg meg selv til å fortsatte og det føltes faktisk morosomt å jobbe med prosjektet igjen. Som du ser over (pga hvor lite tekst det er) så har det vært sånn fra 4. februar. Jeg tror problemet har vært en kombinasjon av at har jobbet for mye med prosjektet, at jeg skulle begynne med tester (som aldri er gøy, og vanskelig med grafikk), problemer med at det som er vanskelig med prosjektet må nå gjøres, andre fag som også trengs å bruke tid på og utløseren var nok at det ble en pause med informatikk hytteturen.

19.02.2020

- 11-14

Jeg la til map support for konkrete map typer, uiet suger men det fungerer. Jeg vet fortsatt ikke hvordan en map-like type ser ut. Jeg gjorde også det slik at alle class builders må ha et definert navn, det var slik i praksis men i teorien kunne det lages en class builder med et null navn.

Abstrakte Klasser 20.02 - 27.02 ([e6a9ed](#) - [3cacaee](#))

Subtyper kan nå bli valgt fra et interface eller abstrakt klasse. Et vindu åpnes og du kan velge mellom alle subtyper av en gitt klasse. Det var noen problemer i starten med selve serialiseringen, men dette har blitt løst for property inclusion (se JsonTypeInfo). Senere ble dette flyttet ut til complex class builder serializer.

Annet mindre ting ble også gjort, men det trengs ikke å nevnes her

Rekursjon 27.02-06.03 ([25db03](#) - [677252](#))

Serialisering av rekursive objekter vil funke som forventet. Denne var tøff, jeg brukte syv dager på å få dette til. I starten vil jeg bruke Jacksons innebygde serialisering av objekter ettersom det gir mindre kode å holde vedlike. Jeg lagde en enkel serialisering for alle class builder som vil rett og slett kalle den rette serialiseringen for den gitt instansen. Så for eksempel vil en map class builder returnere jackson sin serialisering av map. Men det funket ikke ettersom provider instansen ble laget på nytt av en eller annen grunn.

På et tidspunkt så hadde jeg metoder som “compilet” og “linket” serialiserings objektene. Kompileren vill ta alle seraliserings objektene og rekursiv kompilere dem. Den husket alle objektene som var serialisert og ville markere objekter som den hadde sett før. Etter dette ville “linkeren” ta alle disse markørene og bytte dem ut med instanser av det serialiserte objektet. Tanken bak dette var at dette kompilerte objektet kunne bli converted av jacksons standard implementasjon, men dette funket heller ikke. Det viser seg nemlig at jackson sin map implementasjon ikke tillater rekursive kart.

Til slutt så stjal jeg kode fra jackson sin BeanSerializer. Dette funker for testene og måten jeg lager objektene på, men det ikke garantert å funke med alle de forskjellige måtene å serialisere objekter. Det burde være trivielt å utvide koden slik at den er med kompatibel.

Det ser ut som det er en ganske mye lettere måte å gjøre alt dette på, men noen få problemer må løses før det vil funke. Blant annet er problemet at jackson sjekket at klasser er korrekt type.

JsonMappingException: object is not an instance of declaring class (through reference chain: java.util.HashMap["with"])

blir returnert når man bytter ut ComplexClassBuilderSerializer koden med

```
//Man trenger å vite PropertyWriteren hvis ikke kastes et NPE
val prop = (value.parent as?
ComplexClassBuilder)?.propInfo?.get(value.key?.getPreviewValue())
provider.findValueSerializer(value.type, prop).serialize(value.serObject,
gen, provider)
```

Enums 07.03.2020 (commit [9452ab](#))

Etter å ha jobbet så lenge med rekursjon ville jeg implementere noe som var enkelt. Enums ble til på noen få timer! Det er en implementasjon av SimpleClassBuilder som gjør at ikke mye kode trengte å bli skrevet. Det vanskeligste var å finne en måte å få alle verdiene av enumet fra klassen, som måtte dessverre bli gjort med refleksjon.