DataGenerator

Generatoren bruker Play som du må laste ner:

<http://www.playframework.com/> version 1.1

java sdk

koden er i scala:

<https://github.com/steria/datagen>

Der er to måter å få test data til serveren din for lagring og aggregering, via pull og push. Dataformatet er det samme, så du kan enkelt bruke den som passer best for det du tester

**Formatet** er:

* stationId:String - unik id på målestasjonen. Det er mulig å parse den, men du gjør det sikkert ikke
* council:String - kommunen dataen er fra
* timeStamp:Timestamp i formatet YYYY-MM-DDTHH:mm:SS.xxx+01:00 - tidspunkten dataen er fra
* kw:Double - forbrukning siden sist rapportert
* cumkwh:Double - kummulativ forbrukning fra starten på året

**Patterns i data**

Der finns en konfigurasjonsfil application.conf med kommune og antal målestasjoner. I tillegg så genererer den noen data patterns som du kan bruke for analyse:

* tid på dagen
* måned i året, mer forbruk om vinteren og foretninger er lukket i Juli)
* ulike hushold typer (i ulike proportioner i store og små kommuner)
* små og store foretninger (i større kommuner)
* koden er i scala/github, så fritt frem og endre i filene MeasurementSource.scala og Generator.scala

Push

For å bruke push fra generatoren til å pushe data til din server så kalle REST PUSH mot url:

/push/start med argumenter (JSON):

* intervalCount:Int - hvor mange sykler skal den sende ut data (bruk små verdier når du tester)
* measurementFrequencyMin:Int - hvor mye skal timestampet i dataen oppdateres og kw verdiet oppdateres
* sendDelaySec:Int - hvor langt delay (sleep i koden) skal det vare mellom hver sykel, gir serveren din litt tid å hvile og det natorlige verdiet hadde vart measurementFrequencyMin\*60
* startDate:String i formatet “YYYY-MM-DD” - dato på timestampet i først sykeln
* url:String - destinasjonen på dataen
* councilFilter:String - kommaseparert liste på kommuner du ønsker data for, eller “” for alle kommuner. F.eks. “0118,0111”
* parallel:Int - hvor mange parallelle kall som skal sendes
* dataPerCall:Int - hvor mange målinger som hver melding skal inneholde (max)

f.eks. kan argumentene vare:

{"intervalCount":3,"measurementFrequencyMin":15,"sendDelaySec":5,"startDate":"2013-01-01","url":"[http://localhost:9000/receivePush","councilFilter":"0118,0111](about:blank)",”parallel”:1,”dataPerCall”:1000}

**Pull**

For å sette opp pull så må du først kalle REST PUSH /pull/init med argumenter:

* measurementFrequencyMin:Int - hvor mye skal timestampet i dataen oppdateres og kw verdiet oppdateres
* startDate:String i formatet “YYYY-MM-DD” - dato på timestampet i først sykeln
* councilFilter:String - kommaseparert liste på kommuner du ønsker data for, eller “” for alle kommuner. F.eks. “0118,0111”

f.eks. kan argumentene vare:

{"measurementFrequencyMin":15,"startDate":"2013-01-01","councilFilter":"0118,0111"}

for å senden hente data kaller du REST GET /pull/chunk uten argumenter, og får hver gang ett komplett dataset for et tidspunkt (som er automatisk oppdatert til neste gang du spør). Når du tester lokalt kan det vare lurt å bruke curl til å hente data til en lokal fil, et intervall med alle målepunkter er ca 300MB data

TODO: generere datafiler slik at man ikke trenger å generere selv

**Aggregert data**

For å stubbe ut servene som lagrer og aggregerer data kan du kalle generatoren med PUSH /aggData/init og argumenter:

* startDate:String i formatet “YYYY-MM-DD” - dato på timestampet i først sykeln
* councilFilter:String - kommaseparert liste på kommuner du ønsker data for, eller “” for alle kommuner. F.eks. “0118,0111”

f.eks. kan argumentene vare:

{"startDate":"2013-01-01","councilFilter":"0118,0111"}

for å seden hente data kaller du GET /aggPull/day eller aggPull/month som gir deg aggregert data per målepunkt og dag/måned

TODO: generere datafiler slik at man ikke trenger å generere selv