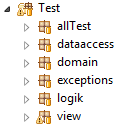
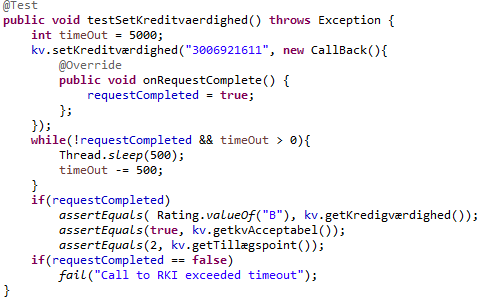
# Test

Vi har i vores project selvfølgelig testet vores system undervejs i processen. Dette har vi gjort på en systematisk måde, så hvis vi senere i processen skulle ændre noget i koden, skulle testene stadig kunne køre. Vi har valgt og fokusere på centrale dele af systemet, og lave test suites til disse. Grunden til at hele system ikke bliver testet er, at man kan ende med og dobbel teste bestemte dele af systemet, og vi ikke har set det nødvendigt og gøre. Vores test har hjulpet os til en bedre forståelse for dele af vores kode, samt også gjort det muligt for os, at identificere fejl tideligere. De centrale dele vi har fokuseret på er UC5, UC6 og UC7.

Sådan ser vores test folder ud. Vi har delt det op på denne måde, da vi har alle domain test klasser i domain packagen osv. Dette gør det muligt for os og test hvad der nu skulle ligge vores domain package i vores src folder. I hver package har vi også en TestSuite for alle klasser i den package. De bliver så alle sammen samlet i allTest, som er en TestSuite til alle vores tests.



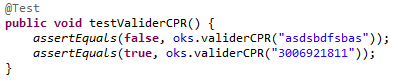
Her er koden for vores KreditvaerdighedTest klasse. Dette er den eneste metode i køre i denne klasse. Dette skyldes at der kun skal testets på en ting i vores Kreditvaerdighed klasse. Om den gør det rigtige ud fra hvilken kreditværdighed kunden nu har. Der bliver instansieret 2 private variabler i denne klasse også:

kv som er af typen Kreditværdighed, samt en boolean kaldet requestcompleted.

Vi også at det er nødvendigt for os at lave en variable i metoden: timeOut. Dette skyldes at hvis vi kørte metoden uden vores timeOut, ville metoden simpelthen fejle, da vores kreditvaerdighed metode der bliver kørt, er en Thread. Og denne Thread eksistere da vi både skal kunne hente kreditværdighed og rentesats. Uden og skulle vente på at den anden blev færdig. Vi tæller så timeOut ned i metoden så længe requestCompleted er false, og timeOut er størrere end 0. Vores onRequestComplete() er en metode der bliver overrided fra Kreditværdighed klassen.

Vi forvente så i vores assertEquals, at kreditværdigheden er ”B”, at kunden er acceptabel for lån, og at der bliver lagt 2 tillægspoint til vores tillægspoints variable. Hvis dette fejler, vil der blive skrevet en fejl besked ud.

Testen for RenteSats er for sin vis den samme. Metoderne til disse to klasser har hjulpet os med og forstå hvordan vores threads virker, og ikke mindst hvordan vores CallBack variable virker.

Vi har selvfølgelig også test på vores view. Her har vi fokuseret på og teste vores valider metoder, og om de virkelig kan valider input fra brugeren. Klassen tester alt fra CPRnummer, til navn på kunden. Denne klasse kan ses i sin helhed i view pakken i vores Test folder.

Test suites for systemet vil være og finde i bilagene.