Testning

En del av Definition of Done för en User-Story i projektet bestod av att den skulle testas innan den var klar. Detta har gjorts på lite olika sätt beroende på vilken User-Story det var, men ofta innebar det att den nya funktionaliteten visades upp i emulatorn.

Applikationen har testats på följande sätt:

- Konsolen
- Debug
- Emulator
- JUnit

Konsolen
Under utvecklingen av applikationen används ofta konsolen och då mer specifikt System.out.println() för att testa att koden gjorde det som förväntades av den. Detta användes eftersom det ofta är enkelt att se direkt vad det är som går fel och samtidigt får man också en bättre förståelse för hur koden man precis har skrivit faktiskt fungerar.

Alla System.out.println() togs sedan bort i slutet av projektet för att inte påverka applikationens körtid.

Debug
Av samma anledningar som ovan användes också den inbyggda debug-funktionen i Android Studio. Till skillnad från att skriva till konsolen kan man här stega igenom felet och på så sätt ännu enklare hitta exakt vart felet uppstår.

Emulator
En emulator av en Android mobiltelefon kopplad till Android Studio används också frekvent under utvecklingen för att testa hur ändringar såg ut i applikationen. Med denna kunde man även testa hur snabbt olika delar av applikationen laddar och hur navigationen av applikationen kändes för att sedan göra modifieringar utefter detta.

JUnit
Det skrevs även ett antal JUnit-tester. Dessa används bland annat för att testa de fall där vi konverterade information från en JSON-fil till datum och strängar som sedan kunde användas i våra klasser. Det var en utmaning att kunna testa privata metoder men vi lyckades till slut skapa en testklass som använder sig utav “setAccessible(true)” och “method.invoke”. Nu kan vi testa klassen utan att påverka den på något sätt. Vi anropar sedan metoden i klassen för att göra en JUnit-assertEquals på outputen.