TALLINNA TEHNIKAÜLIKOOL

Infotehnoloogia teaduskond

*Mock object* testimises

Üliõpilane: Enrico Vompa

185787IAIB

Õppekava: IAIB17/17

Tallinn 2020

# Sisukord

[**Sisukord**](#_heading=h.4d34og8) **2**

[**Mis on mock object**](#_heading=h.z5cc5iwwzkl5) **3**

[**Probleem, mida lahendada**](#_heading=h.35jpqt55b6ik) **4**

[**Mock objekti seadistamine**](#_heading=h.xdr7qg9rhuom) **5**

[**Klassi diagramm mock objektiga**](#_heading=h.nvbtr449hpze) **6**

[**Testi käivitamise jadadiagramm**](#_heading=h.uvom8pr6t7ib) **7**

[**Kokkuvõtte**](#_heading=h.d5c3tgk4xyy4) **9**

# 

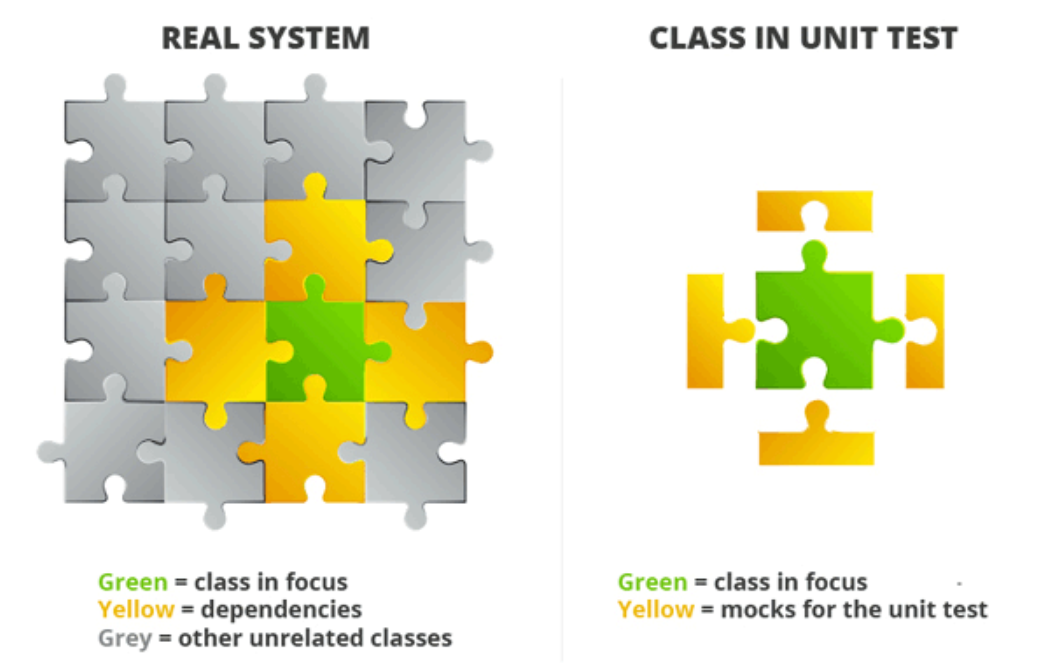
# Mis on *mock object*

Objektorienteeritud programmeerimises on libaobjekt (*mock object*) simuleeritud objekt, mis jäljendab reaalse objekti käitumist kontrollitud viisil. Peamiselt kasutatakse libaobjekte ühiktestimises (*unit testing*), kus proovitakse testida mõne muu objekti käitumist.[[1]](#footnote-0)

Motivatsioon[[2]](#footnote-1):

* Libaobjekt võimaldab testida olukordi, mida muidu võiks olla raske luua või taasesitada.
* Päris objekti veel ei eksisteeri või selle käitumine võib muutuda.
* Mock testid on kiiremad, sest välissüsteemiga suhtlus (võib olla ka teine objekt) jäetakse tegemata.
* Deterministlik väljund, isolatsioon. Kui välissüsteem tagastab muutuvat infot, siis selle ümber on väga raske kasulikke teste kirjutada. Mock objektid isoleerivad testitava objekti teistest objektidest ning see teeb ühiktestimise palju lihtsamaks.

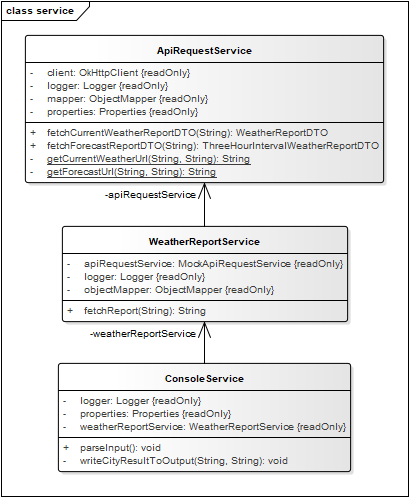
Mock objekti kontseptuaalne visualisatsioon on näha järgmisel pildil:



Joonis 1: ühiktestimise struktuur[[3]](#footnote-2)

# Probleem, mida lahendada

Järgmisel pildil on näha lihtsa süsteemi klassidiagramm, mis teeb API päringu ning salvestab tulemuse muudetud kujul faili:



Joonis 2: Süsteemi klassidiagramm

Pildilt on näha, et *ConsoleService* sõltub *WeatherReportService* teenusest, mis omakorda sõltub *ApiRequestService* teenusest.

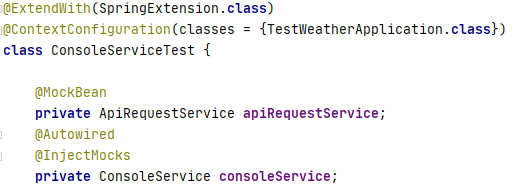
*ConsoleService* klassi testides peamised eesmärgid on järgmised:

* Mitte teha päris API päringuid süsteemi testides, kuna see teeb testimise aeglasemaks ning tülikamaks, kuna API päringut tehes on vaja anda kaasa token ning tegu poleks enam siis ühiktestiga, kuna kogu süsteem pannakse käima.
* Saada deterministlikut infot tavalist flow’d läbides. Kuna ilm on varieeruv ning API’st tulev tagaside samuti, siis raske on ühiktestida kindlat funktsionaalsust.

# Mock objekti seadistamine

Libaobjekti kasutamine parandab eelnevalt mainitud probleemi.

Libaobjekti kasutamise näide sai tehtud Spring Boot’i[[4]](#footnote-3) abil, kus libaobjekti tekitamine oli sama lihtne, kui annotatsiooni lisamine:

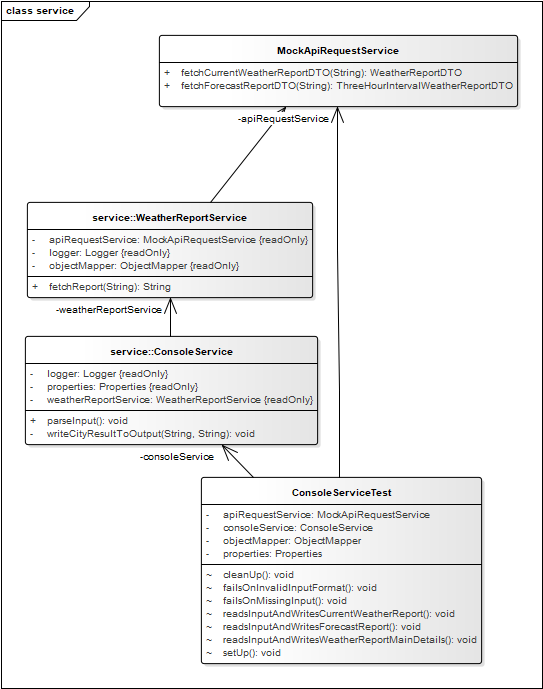


Joonis 3: Libaobjekti annotatsioon

Antud pildil *ConsoleService* on testitav klass ning *apiRequestService* on libaobjekt. @MockBean tähistab seda, et objekt on libaobjekt ning @InjectMocks tähistab seda, et testitav klass kasutaks mock objekte reaalsete objektide üle.

# Klassi diagramm mock objektiga

Uus klassidiagramm, kus kasutatakse eelnevalt seadistatud mock objekti näeb välja järgmine:

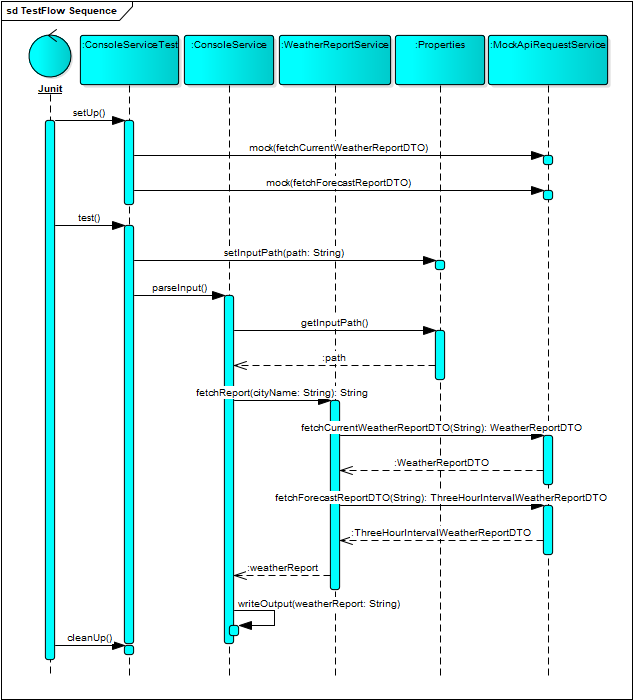


Joonis 4: mockiga klassi diagramm

Nagu ka klassidiagrammist on näha siis *ConsoleService*’i ning *WeatherReportService* ei muutunud ning nende klasside sõltuvused ka mitte.

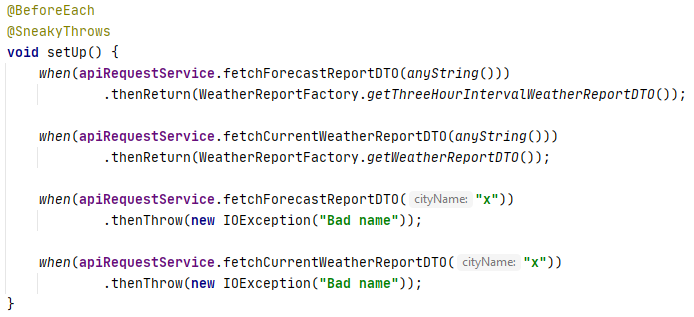
# Testi käivitamise jadadiagramm

Järgmisel pildil on näha tegevused, mis juhtuvad, kui Junit[[5]](#footnote-4) paneb testi käima:



Joonis 5: test execution

Pildilt on näha, et testimise alguses antakse juhised libaobjektile, mis sisendite puhul mida ta väljastama peaks:



Joonis 6: mock objekti seadistamine

Pildilt on näha, et API päringud peaksid tagastama suvalise nimega deterministliku objekti ning kui linna nimi on “x”, siis visatakse erind.

Käivituse ajal pöördutaksegi libaobjekti poole *WeatherReportService*’i poolt ning *ConsoleService* saabki teistelt objektidelt deterministlikke tulemusi ning on võimalik erinevate juhtumite läbi testimine.

# Kokkuvõtte

Kokkuvõtteks võimaldavad libaobjektid ühiktestimist edukalt läbi viia, isoleerides testitava objekti reaalsest maailmast. Antud näiteks reaalseks maailmaks oli hetke ilma API. Libaobjektid võimaldasid kõik erinevad võimalikud olukorrad läbi testida ja saab kindel olla, et iga sisendiga tagastab testitav objekt oodatud väljundi.

Kood on saadaval [GitHub](https://github.com/envomp/2020-automated-testing-project)’is

1. "Mock object - Wikipedia." <https://en.wikipedia.org/wiki/Mock_object>. Accessed 2 Dec. 2020. [↑](#footnote-ref-0)
2. "What is the purpose of mock objects? - Stack Overflow." 2 Sep. 2010, <https://stackoverflow.com/questions/3622455/what-is-the-purpose-of-mock-objects>. Accessed 2 Dec. 2020. [↑](#footnote-ref-1)
3. "What is Mocking in Testing?. It is just an introduction ... - Medium." 21 Oct. 2017, <https://medium.com/@piraveenaparalogarajah/what-is-mocking-in-testing-d4b0f2dbe20a>. Accessed 2 Dec. 2020. [↑](#footnote-ref-2)
4. "Spring Boot." <https://spring.io/projects/spring-boot>. Accessed 2 Dec. 2020. [↑](#footnote-ref-3)
5. "JUnit 5." <https://junit.org/junit5/>. Accessed 2 Dec. 2020. [↑](#footnote-ref-4)