ROMMENS Christof

Abstract

Dit document beschrijf het design en de wijze waarop de applicatie is geïmplementeerd

Refuel DESIGN

Javadocs 3

Styleguide 3

MVP Design/Clean Code 3

Model 3

View 3

Presenter 3

Domain 3

Databinding 3

Application 4

Model 4

View 4

Presenter 4

Login 4

Model 4

View 4

Presenter 4

Todos 5

Applicaiton 5

Login 5

# Javadocs

## Styleguide

* [Java styleguide](http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html#styleguide)

# MVP Design/Clean Code[[1]](#footnote-1) [[2]](#footnote-2) [[3]](#footnote-3)

## Model

* The **data** layer. Responsible for handling the business logic and communication with the network and database layers.

## View

* The **UI** layer. Displays the data and notifies the Presenter about user actions.
* All Views implement the same **BaseView interface** that allows setting a Presenter.
* The View notifies the Presenter that it is ready to be updated by calling the **subscribe method** of the Presenter in **onResume**. The View calls **presenter.unsubscribe()** in **onPause** to tell the Presenter that it is no longer interested in being updated.

## Presenter

* **Retrieves** the **data** from the **Model**, applies the UI logic and manages the state of the View
* The relationship between the Presenter and its corresponding View is defined in a **Contract** **interface** class, making the code more readable and the connection between the two easier to understand.

## Domain

Bevat de klassen die data bijhoudt omtrent objecten binnen de applicatie zowel de user, vehicle, refuel.

De beschrijving hiervan werd ook opgenomen in de glossary.

## Databinding

De binding tussen de data uit het model en de view gebeurd op basis van databinding in de layout XML-bestanden.

Er wordt voor elke layout die databinding implementeerd (<data>...</data>) een bindingklasse gegenereerd bij het builden van de app.

# Application

**Packagename**: be.ehb.digx.refuel

## Model

**Packagename**: be.ehb.digx.refuel.domain.model

* + User

## View

**Packagename**: be.ehb.digx.refuel

**Interfaces:** BaseView

* Maakt gebruik van een generiek type

[*https://docs.oracle.com/javase/tutorial/java/generics/types.html*](https://docs.oracle.com/javase/tutorial/java/generics/types.html)

* Voorziet een methode setPresenter() zodat de view de presenter kan aanspreken.

## Presenter

**Packagename**: be.ehb.digx.refuel

**Interfaces:** BasePresenter

* Voorziet een start() methode  
  Deze methode kan door de view opgeroepen worden bij het starten onCreate() of opnieuw tonen van het scherm onResume().  
  Bij het oproepen van de Presenter.start() kan er in de presenter voor gezorgd worden dat de data wordt opgeladen.

# Login

**Packagename**: be.ehb.digx.refuel.login

## Model

**Packagename**: be.ehb.digx.refuel.login.login.domain.model

* + Login

## View

**Packagename**: be.ehb.digx.refuel.login.login

**Impelemntations**: LoginView

* Logon()
  + Succesvol
    - Naar volgend scherm navigeren: VehicleList.
  + Error
    - Error weergeven en gebruiker mogelijkheid geven om opnieuw in te loggen.

## Presenter

**Packagename**: be.ehb.digx.refuel.login.login.domain.model

**Implementations:** LoginPresenter

* Logon()
  + Aanloggen bij firebase

# Todos

## Applicaiton

* Multiscreensize
* Unit-test framework implementeren

## Login

* Validatie van de inputvelden
* Progresbar tonen tijdens het aanloggen

1. https://github.com/googlesamples/android-architecture/ [↑](#footnote-ref-1)
2. https://upday.github.io/blog/model-view-presenter/ [↑](#footnote-ref-2)
3. https://codelabs.developers.google.com/codelabs/android-testing/#4 [↑](#footnote-ref-3)