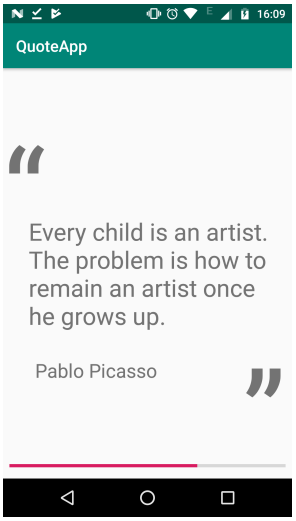
Labo Gebruikersinterfaces
Reeks 7: ViewModel introduction

28-29 april 2020

In this lab session we will introduce the ViewModel component that stores and manages all UI-related data. We will update an existing app to use a ViewModel and LiveData. The app shows a quote together with a progressbar. Each quote is visible for five seconds before the next quote is displayed.



1 Assignment

- Download the startcode from Ufora. Open the project in Android studio and explore the source code. We used data binding and observable fields to easily update our View.
- The app still has a bug however. Rotating the device destroys and recreates the current activity which means that we again see the first quote.
- Read briefly the 4 tasks below, and look at what tips are available at the end of this document. Then start with task number 1.
- Modify the app to use a ViewModel to keep track of all the UI related data:
  1. Add a ViewModel class to your project. This should be a subclass of `AndroidViewModel`. Move the right data (`dao`, `currentQuoteIndex`, `quote`, `author`, `progress` and `timer`) and their initialisations from `MainActivity.kt` to your ViewModel class. Keep in mind

that the QuoteDAO class needs a *Context* to be able to read the json file that contains the quotes.

2. Change the type of the attributes from *ObservableField* to *MutableLiveData*. LiveData is also observable but in addition it is also lifecycle aware which means that it will not trigger an update of the view if the view is not visible. You can find all information about LiveData here: https://developer.android.com/topic/libraries/architecture/livedata

3. Obtain a reference to the ViewModel in your activity and bind it to the variable you created in your layout file. To get a reference of the ViewModel use the `by viewModels()` property delegate. Before you can use `by viewModels()` you should add `implementation 'androidx.fragment:fragment-ktx:1.2.4'` to your dependencies. You should also add the following lines to your build.gradle file (Module:app):

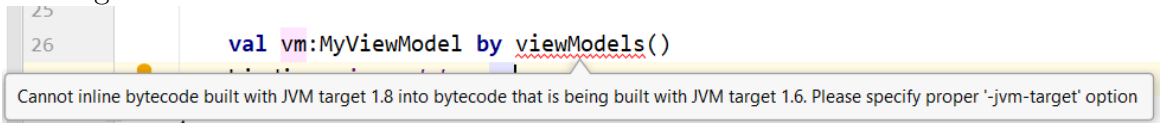
```
android{
    // ...
    compileOptions {
        sourceCompatibility = 1.8
        targetCompatibility = 1.8
    }

    kotlinOptions {
        jvmTarget = "1.8"
    }
}
```

4. Do not forget to set the *LifeCycleOwner* of your DataBinding class. This will enable the View to listen to changes of your LiveData.

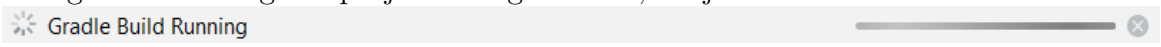
Enkele tips

1. Heb je aanpassingen gedaan aan `build.gradle (Module:app)` en krijg je toch nog foutmeldingen zoals hieronder?



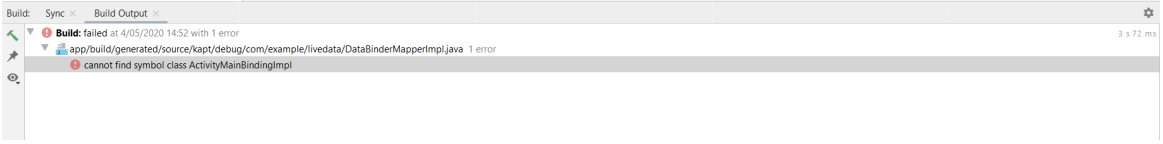
De aanpassingen zijn dan misschien nog niet verwerkt. Kies in het hoofdmenu `Build -> Clean Project` en daarna `Build -> Rebuild project`.

2. Heb geduld - zolang het project niet gebuild is, zie je helemaal onderaan

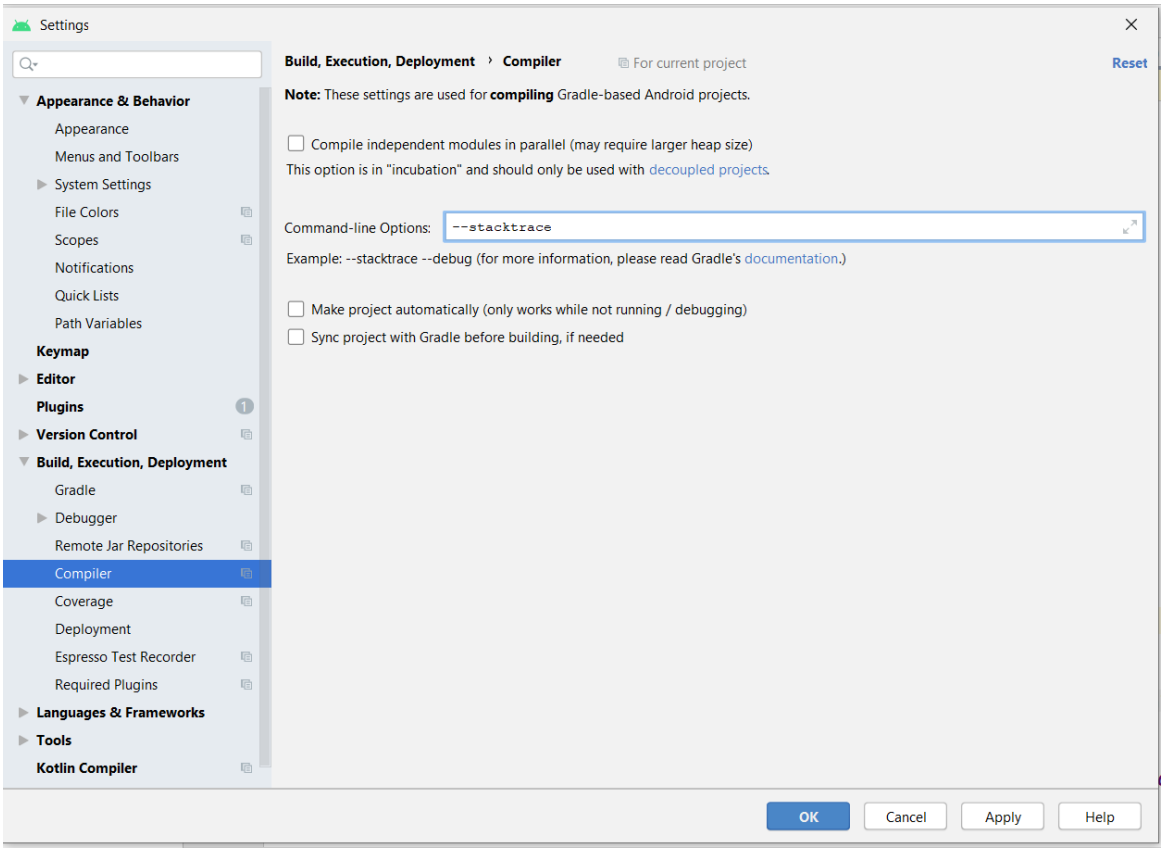


.

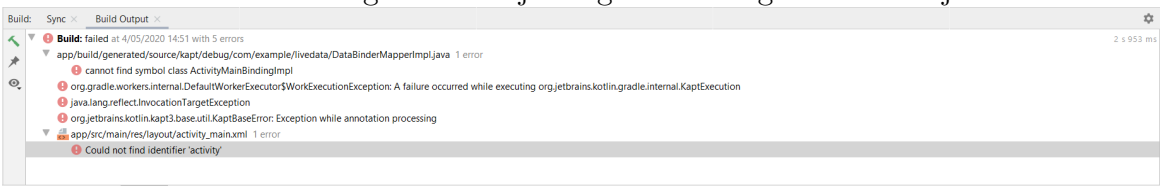
3. Zijn de foutmeldingen onverstaanbaar (= te kort)?



Kies in het hoofdmenu `Files -> Settings... (Ctrl+Alt+S)` en daarna voor `Build, Execution, Deployment -> Compiler` en voeg `--stacktrace` toe bij de *Command line Options*.



Dan zouden de foutmeldingen merkelijk langer en nuttiger moeten zijn.



4. Kijk ook na of je de juiste informatie bekijkt in het *Logcat*-venster: kies de juiste emulator en vraag álle meldingen (*Verbose*).