Inhaltsverzeichnis

Projekt anlegen

Projekt anlegen

```
New Project -> Empty Activity
```

Android SDK

```
SDK Oreo 8.0 API 26
```

Gradle File

Wichtig: Imports im build.gradle (module)

Plugins

```
plugins {
    id 'com.android.application'
```

```
    id 'kotlin-android'
    id 'kotlin-android-extensions'
    id 'kotlin-kapt'
}
```

Databinding enablen (in Android Tag)

```
dataBinding {
    enabled true
}
```

Dependencies (für Corona Test Tracker)

```
dependencies {
    implementation 'androidx.legacy:legacy-support-v4:1.0.0'
    def nav_version = "2.3.5"
    def gradle_version = "7.0.2"

    implementation 'androidx.core:core-ktx:1.7.0'
    implementation 'androidx.appcompat:appcompat:1.3.1'
    implementation 'com.google.android.material:material:1.4.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.1'

    implementation "androidx.navigation:navigation-fragment-ktx:$nav_version"
    implementation "androidx.navigation:navigation-ui-ktx:$nav_version"

    implementation "com.google.android.material:material:1.1.0"

    testImplementation 'junit:junit:4.+'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'

    kapt "com.android.databinding:compiler:$gradle_version"

    def lifecycle_version = "2.2.0"
    // ViewModel
    implementation "androidx.lifecycle:lifecycle-viewmodel-ktx:$lifecycle_version"
    // LiveData
    implementation "androidx.lifecycle:lifecycle-livedata-ktx:$lifecycle_version"
    // Lifecycles only (without ViewModel or LiveData)
    implementation "androidx.lifecycle:lifecycle-runtime-ktx:$lifecycle_version"
    implementation "androidx.lifecycle:lifecycle-extensions:$lifecycle_version"
    // needed for: val viewModel . ... by viewModels()
    implementation "androidx.activity:activity-ktx:1.1.0"
}
```

Navigation

Navigation erstellen

```
Rechtsklick res -> New -> Android Resource File

File name: nav_graph
Resource Type: navigation
```

... Ordner mit dem Namen navigation wird in res erstellt. Darin befindet sich nav_graph.xml

Navigation einbinden

1. activity_main.xml auswählen
2. NavHostFragment einfügen
3. nav_graph auswählen
4. Constraints setzen

activity_main.xml refactor

```
Vorher: androidx.fragment.app.FragmentContainerView
Nachher: fragment
```

Fragment erstellen

1. nav_graph.xml auswählen
2. Mobile Icon mit grünem + klicken
3. Fragment (Blank) auswählen
4. Fragment umkonvertieren von FrameLayout in ConstraintLayout
5. ConstraintLayout in DataBinding Layout umwandeln

Action zwischen Fragments

1. Navgraph auswählen
2. Fragments verbinden wuhuuu

Beispiel Fragment

```
class TestListFragment : Fragment() {

    private lateinit var binding: FragmentTestListBinding

    private val sharedMainViewModel : MainViewModel by activityViewModels()
```

```
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
    }

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {

        binding = DataBindingUtil.inflate(inflater, R.layout.fragment_test_list,
container, false)

        sharedMainViewModel.testList.observe(viewLifecycleOwner, Observer {
entries ->
            updateTestList(entries)
        })

        binding.btnewTest.setOnClickListener { view ->

view.findNavController().navigate(R.id.action_testListFragment_to_inputFragment)
        }

        binding.btToDaily.setOnClickListener { view ->

view.findNavController().navigate(R.id.action_testListFragment_to_dayFragment)
        }

        return binding.root
    }

    fun updateTestList(entries:MutableList<Test>){
        val adapter: ArrayAdapter<Test>? = context?.let {
            ArrayAdapter<Test>(
                it,
                android.R.layout.simple_list_item_1, android.R.id.text1, entries
            )
        }

        binding.lvTests.adapter = adapter


    }

}
```

Die Methode updateTestList fügt die Daten in eine ListView ein. Zum vertikalen Skalieren kann die ListView in ein LinearLayout gepackt werden.

Beispiel Activity Main

```
class MainActivity : AppCompatActivity() {

    private val mainViewModel : MainViewModel by viewModels()

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        val binding = DataBindingUtil.setContentView<ActivityMainBinding>(
            this, R.layout.activity_main)
            .apply {
                this.lifecycleOwner = this@MainActivity
                this.viewModel = mainViewModel
            }

        val navController = this.findNavController(R.id.nav_host_fragment)
        NavigationUI.setupActionBarWithNavController(this, navController)
    }

    override fun onSupportNavigateUp(): Boolean {
        val navController = this.findNavController(R.id.nav_host_fragment)
        return navController.navigateUp()
    }
}
```

Activity Main XML ViewModel

```
<data>
    <variable name="viewModel"
type="at.htl.coronatesttracker.viewModel.MainViewModel"/>
</data>
```

MainViewModel

```
class MainViewModel: ViewModel() {

    private val _testList : MutableLiveData<MutableList<Test>> =
MutableLiveData<MutableList<Test>>()

    val testList: LiveData<MutableList<Test>>
        get() = _testList
```

```
    fun addTestToList(test: Test){
        val value = this._testList.value ?: arrayListOf()
        value.add(test)
        this._testList.value = value
    }

}
```

Klasse Test

```
class Test(var id:String, var dateTime:LocalDateTime, var result:String, var
place:String) {


    override fun toString(): String {
       return "Es ist Freitag 28.01.2022, 21:17. Bitte helfts uns :((((((((((( 
${id}"
    }
}
```

DatePicker & TimePicker

Textfield (für OnClick)

```
<EditText
            android:id="@+id/etDate"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginStart="8dp"
            android:layout_marginTop="16dp"
            android:ems="10"
            android:minHeight="48dp"
            android:inputType="none"
            android:hint="@string/default_date_message"
            app:layout_constraintStart_toStartOf="@+id/guideline2"
            app:layout_constraintTop_toBottomOf="@+id/etId" />
```

File, wo click listener sein soll

```
    var datePicker: DatePickerDialog? = null
    var date:LocalDate = LocalDate.now()
    var timePicker: TimePickerDialog? = null
    var time:LocalTime = LocalTime.now()
```

onCreateView

```
binding.etDate.inputType = 0
    binding.etDate.setOnClickListener{
        view ->
            showDatePickerDialog(view)
}
```

Datepicker anzeigen

```
private fun showDatePickerDialog(v: View) {
        val cldr: Calendar = Calendar.getInstance()
        val day: Int = cldr.get(Calendar.DAY_OF_MONTH)
        val month: Int = cldr.get(Calendar.MONTH)
        val year: Int = cldr.get(Calendar.YEAR)

        datePicker =
            context?.let {
                DatePickerDialog(
                    it,
                    { _, year, monthOfYear, dayOfMonth ->
                        run {
                            binding.etDate.setText(
                                dayOfMonth.toString() + "/" + (monthOfYear + 1) +
"/" + year
                            )
                            date = LocalDate.of(year,monthOfYear+1,dayOfMonth)
                        }
                    },
                    year,
                    month,
                    day
                )
            }

        datePicker!!.show()
    }
```

Show Timepicker Dialog

```
private fun showTimePickerDialog(v: View) {
        val cldr = Calendar.getInstance()
        val hour = cldr[Calendar.HOUR_OF_DAY]
        val minutes = cldr[Calendar.MINUTE]

        timePicker = TimePickerDialog(
            context,
```

```
            { _, sHour, sMinute -> run {
                    binding.etTime.setText("$sHour:$sMinute")
                    time = LocalTime.of(sHour,sMinute)
                }
            },
            hour,
            minutes,
            true
        )
        timePicker!!.show()
    }
```

eventuell hilfreich:

```
val dateTime = LocalDateTime.of(date, time)
```