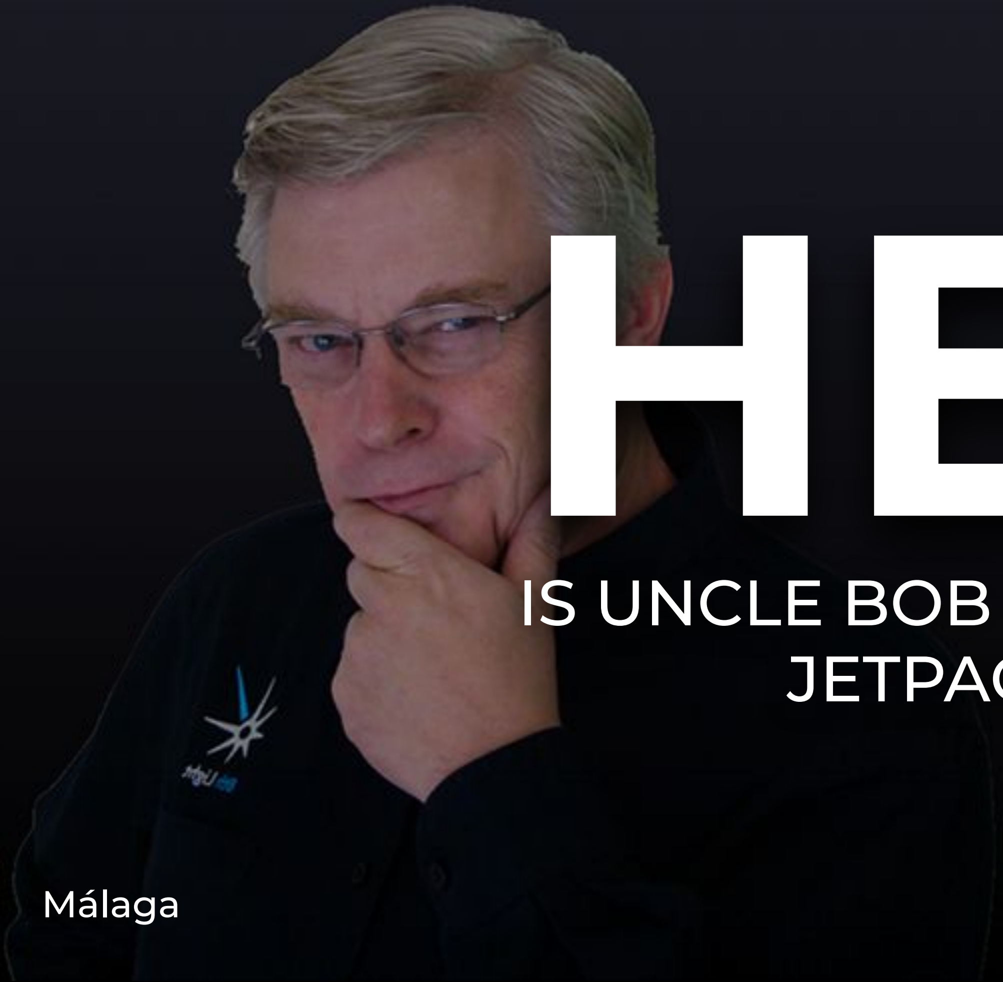
18th June Twenty**20**



HEY

IS UNCLE BOB WEARING A JETPACK?!

Málaga

Rafa Ortega

@ortiOficial

Málaga

AGENDA

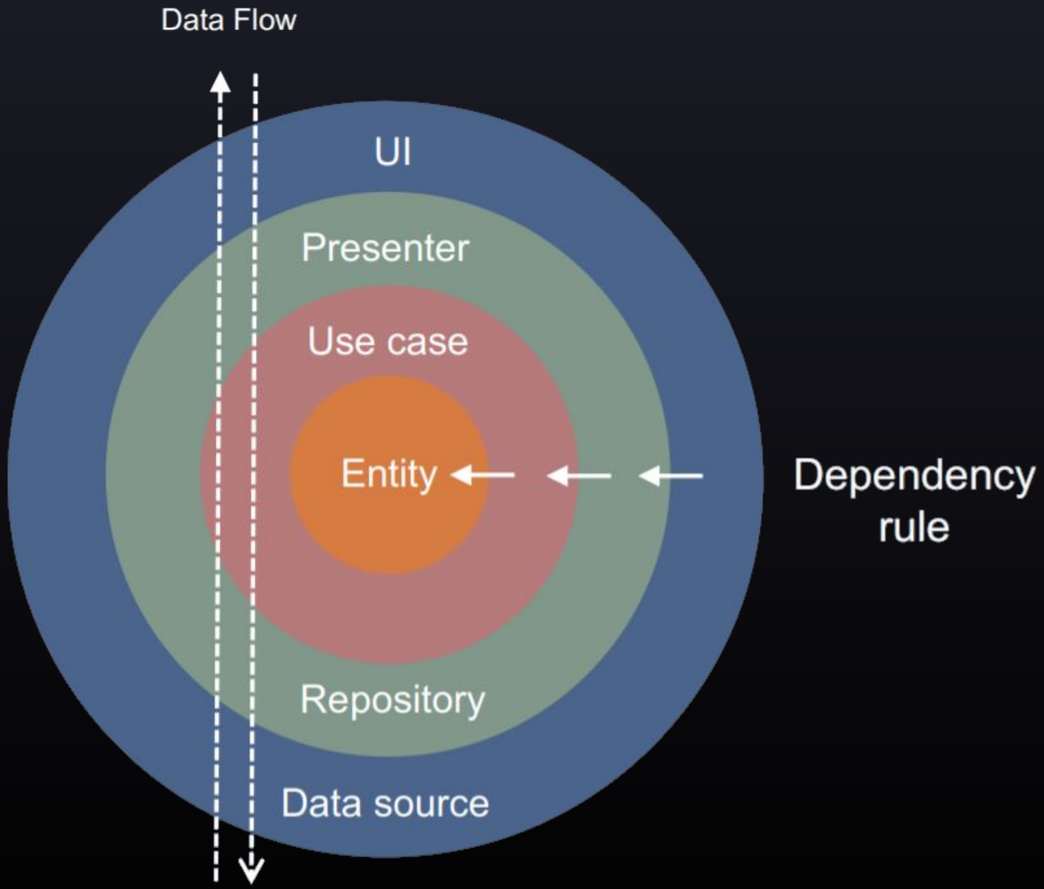
- A word on architecture
- Jetpack
- Our way
  - Live data
  - Model mapping
  - Data binding
  - Testing
- Navigation
- Coming soon

18th June Twenty**20**

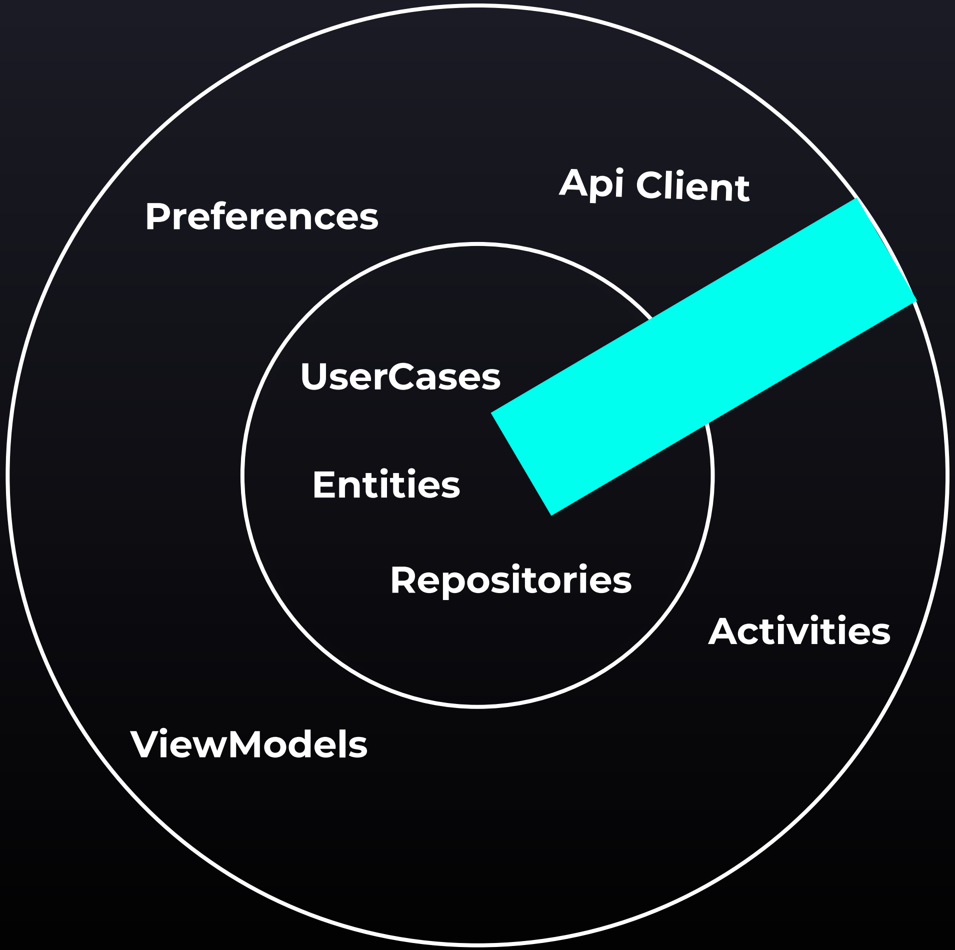
ARCHITECTURE

Málaga

ARCHITECTURE

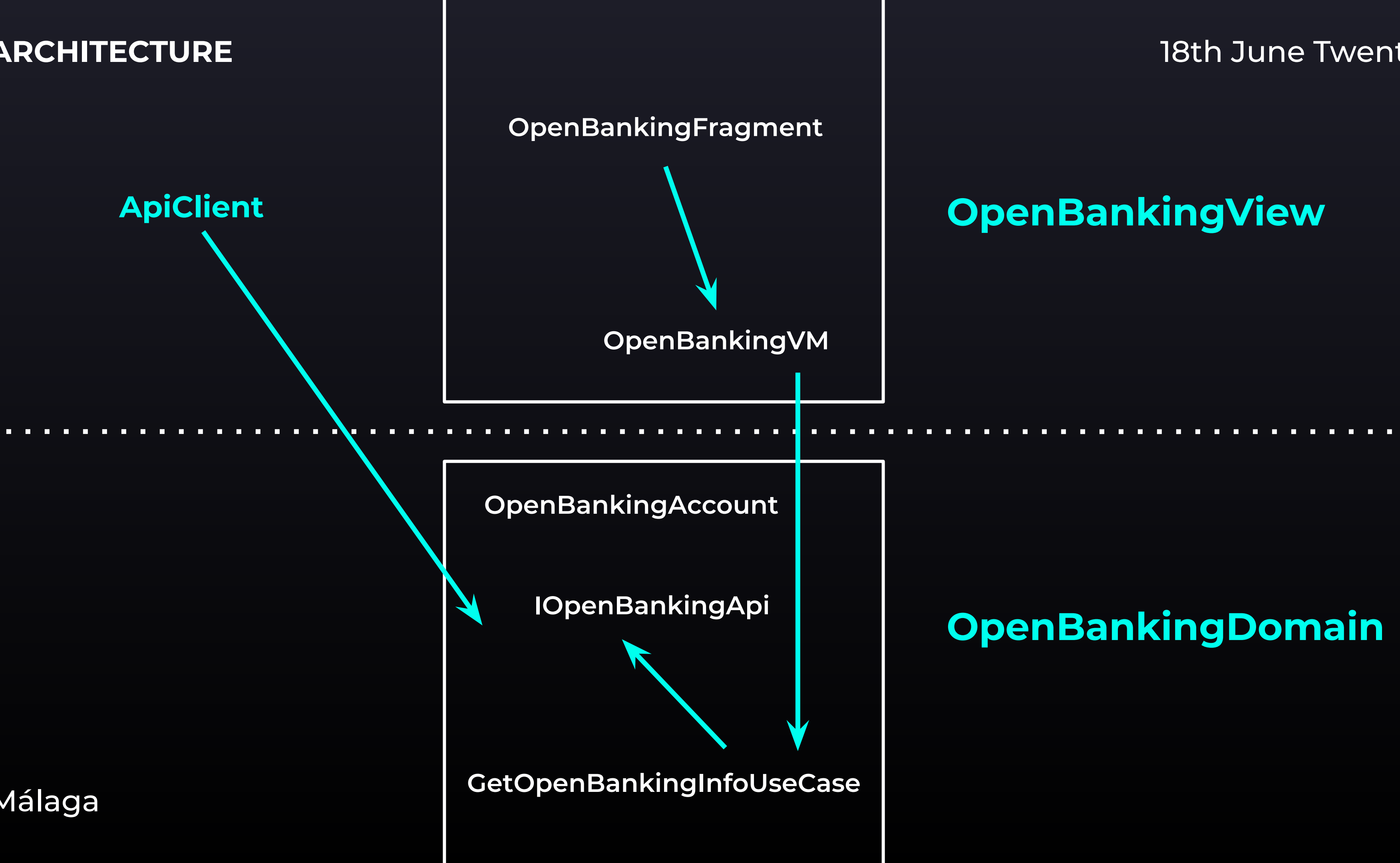


ARCHITECTURE



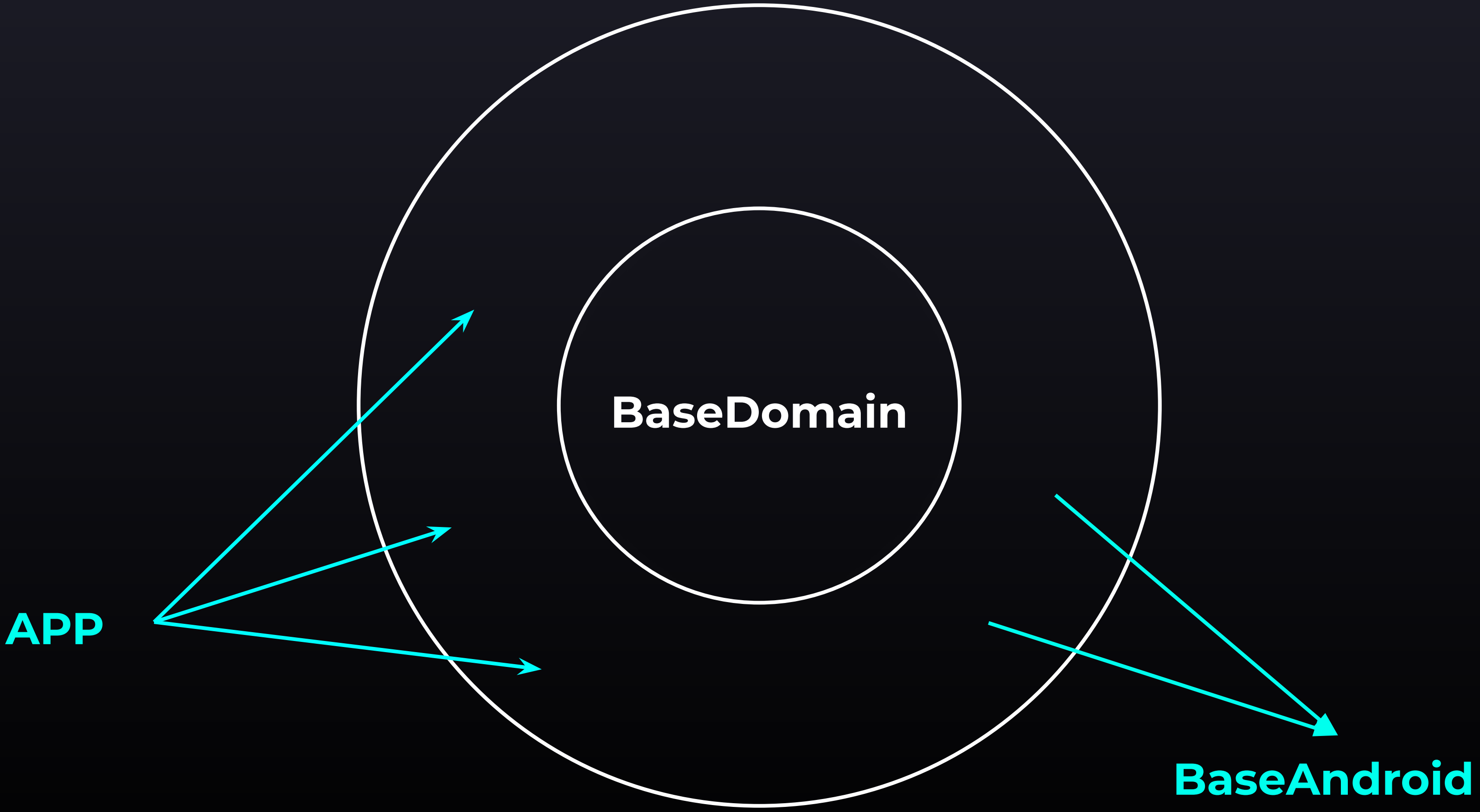
ARCHITECTURE

18th June Twenty**20**



Málaga

SPECIAL MODULES


BaseDomain
APP
BaseAndroid

JETPACK

- Livedata (transformations, LiveDataEvent)
- Databinding (custom adapters)
- Navigation component
- Architecture components (VM, viewModelScope)
- Motion layout

“Jetpack is a suite of libraries, tools, and guidance to help developers write high-quality apps easier. These components help you follow best practices, free you from writing boilerplate code, and simplify complex tasks, so you can focus on the code you care about.”

18th June Twenty**20**

OUR WAY

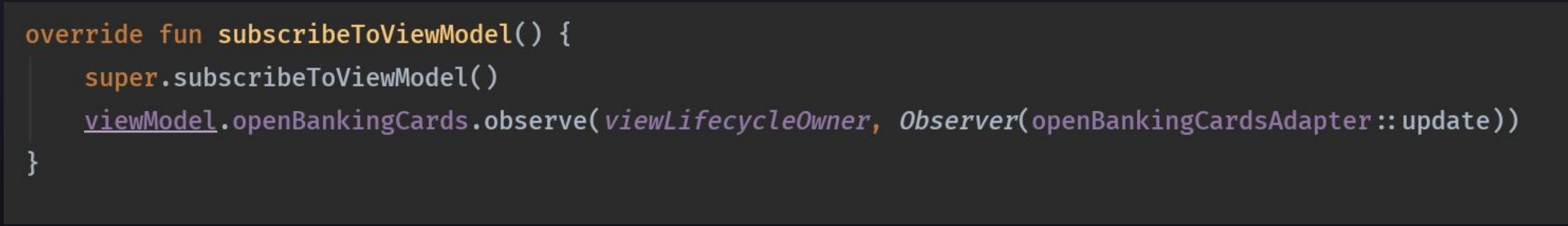
Málaga

18th June Twenty**20**

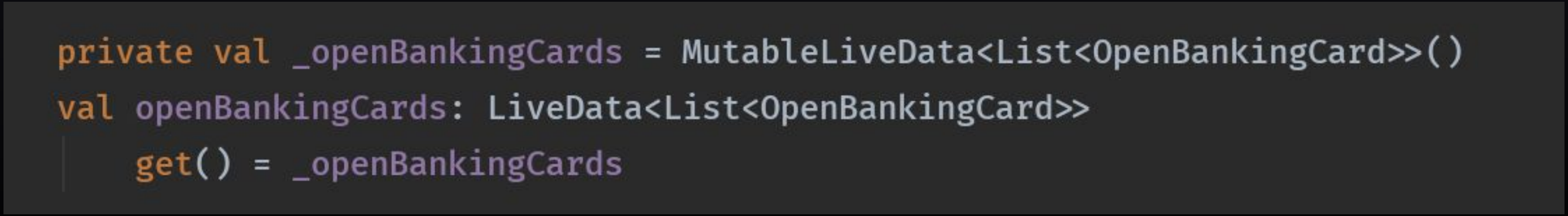
LIVE DATA

Málaga

LIVE DATA

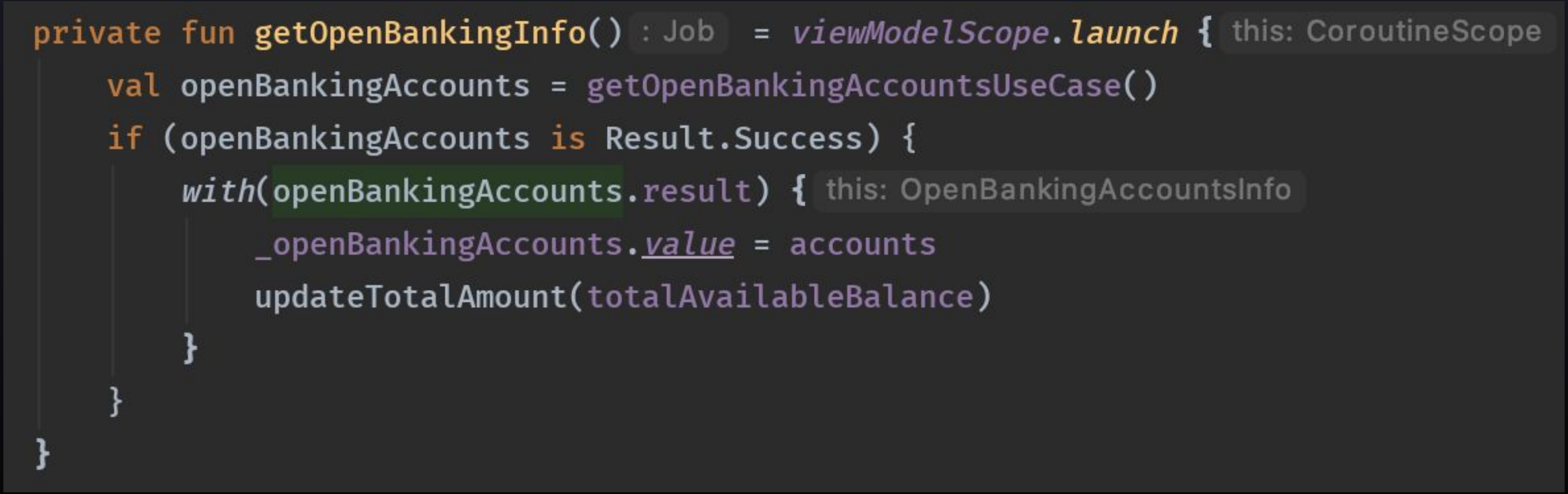


```kotlin
override fun subscribeToViewModel() {
    super.subscribeToViewModel()
    viewModel.openBankingCards.observe(viewLifecycleOwner, Observer(openBankingCardsAdapter::update))
}
```



```kotlin
private val _openBankingCards = MutableLiveData<List<OpenBankingCard>>()
val openBankingCards: LiveData<List<OpenBankingCard>>
    get() = _openBankingCards
```

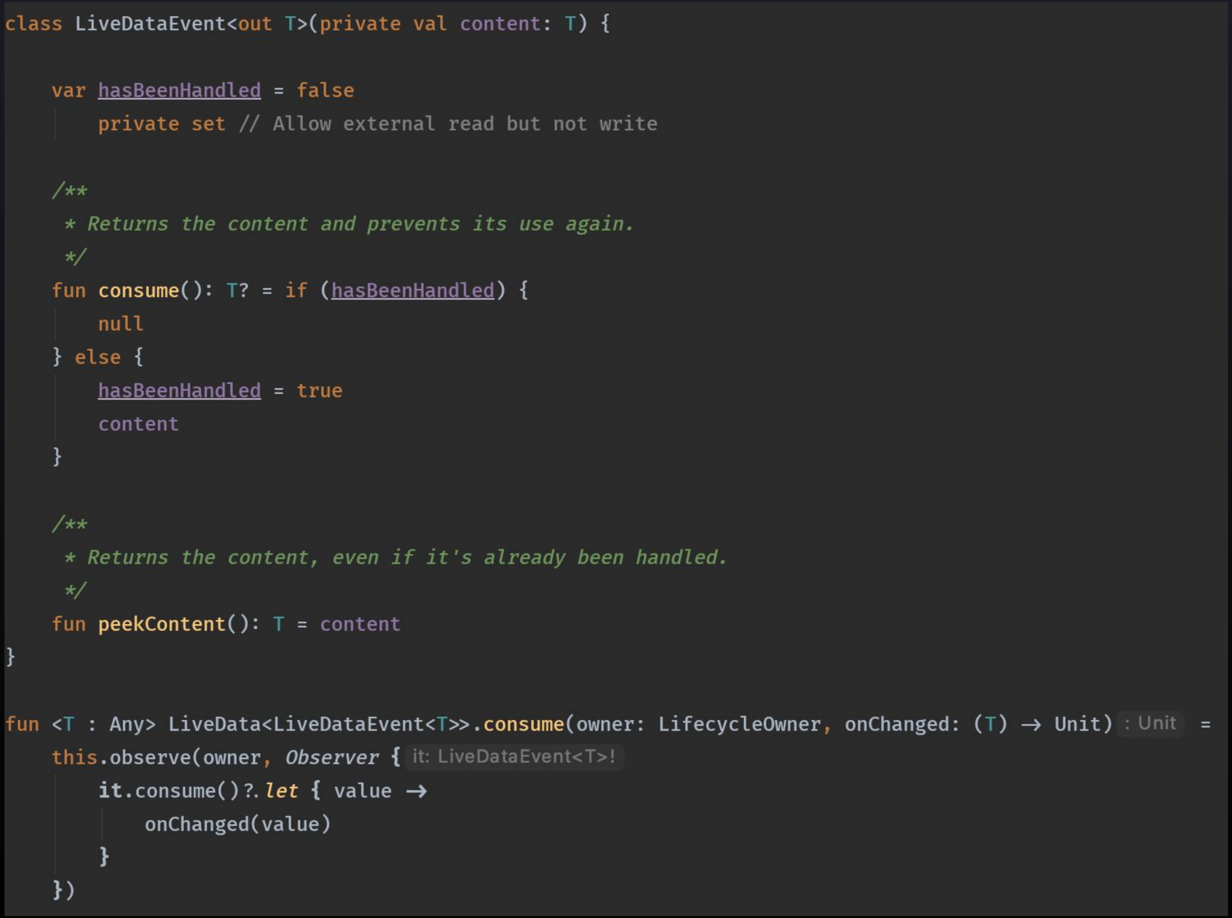
LIVE DATA



```kotlin
private fun getOpenBankingInfo() : Job  = viewModelScope.launch { this: CoroutineScope
    val openBankingAccounts = getOpenBankingAccountsUseCase()
    if (openBankingAccounts is Result.Success) {
        with(openBankingAccounts.result) { this: OpenBankingAccountsInfo
            _openBankingAccounts.value = accounts
            updateTotalAmount(totalAvailableBalance)
        }
    }
}
```

LIVE DATA

18th June Twenty20



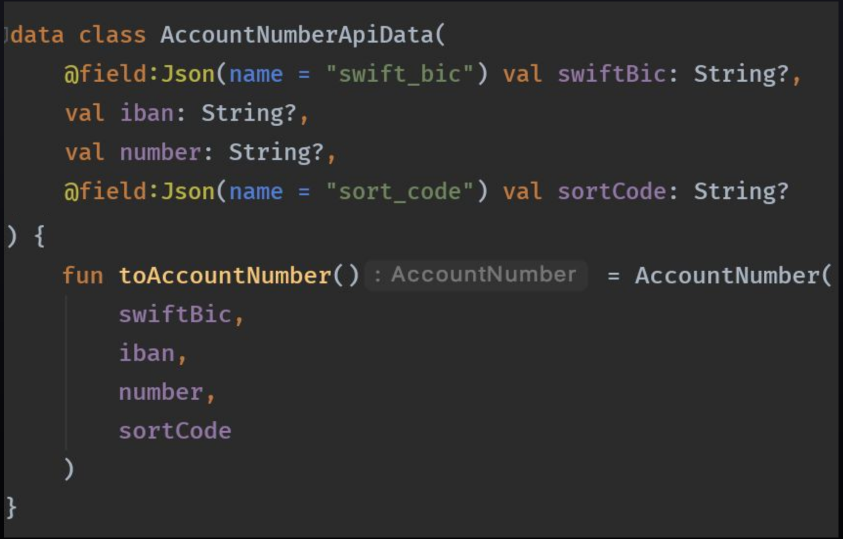
```kotlin
class LiveDataEvent<out T>(private val content: T) {

    var hasBeenHandled = false
        private set // Allow external read but not write

    /**
     * Returns the content and prevents its use again.
     */
    fun consume(): T? = if (hasBeenHandled) {
        null
    } else {
        hasBeenHandled = true
        content
    }

    /**
     * Returns the content, even if it's already been handled.
     */
    fun peekContent(): T = content
}

fun <T : Any> LiveData<LiveDataEvent<T>>.consume(owner: LifecycleOwner, onChanged: (T) → Unit) : Unit  =
    this.observe(owner, Observer { it: LiveDataEvent<T>!
        it.consume()?.let { value →
            onChanged(value)
        }
    })
```

Málaga

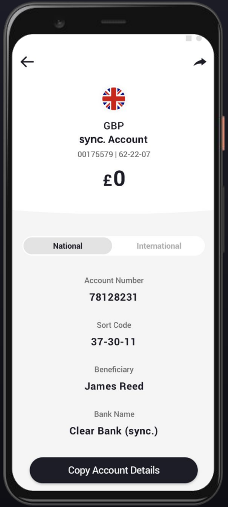
18th June Twenty**20**

MODEL MAPPING

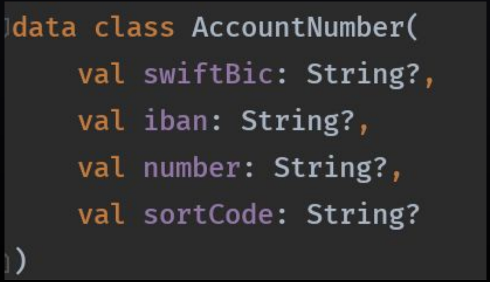
Málaga



```
data class AccountNumberApiData(
    @field:Json(name = "swift_bic") val swiftBic: String?,
    val iban: String?,
    val number: String?,
    @field:Json(name = "sort_code") val sortCode: String?
) {
    fun toAccountNumber() : AccountNumber  = AccountNumber(
        swiftBic,
        iban,
        number,
        sortCode
    )
}
```

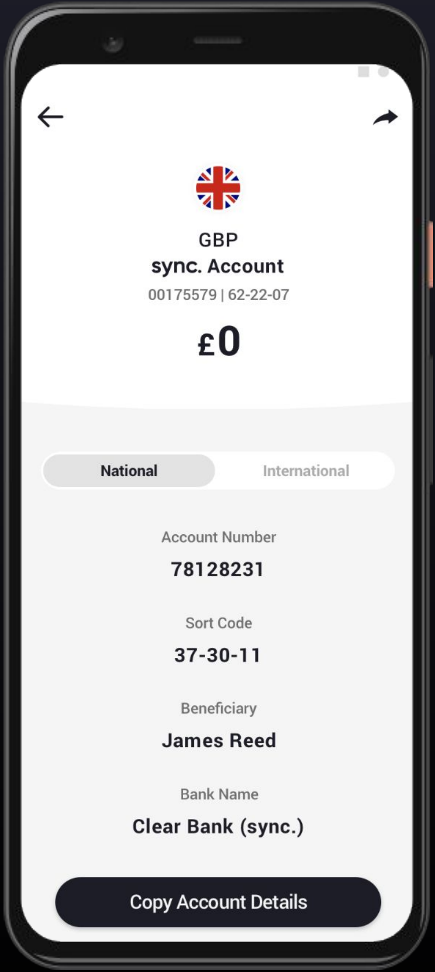


Business Logic



```
data class AccountNumber(
    val swiftBic: String?,
    val iban: String?,
    val number: String?,
    val sortCode: String?
)
```

18th June Twenty**20**

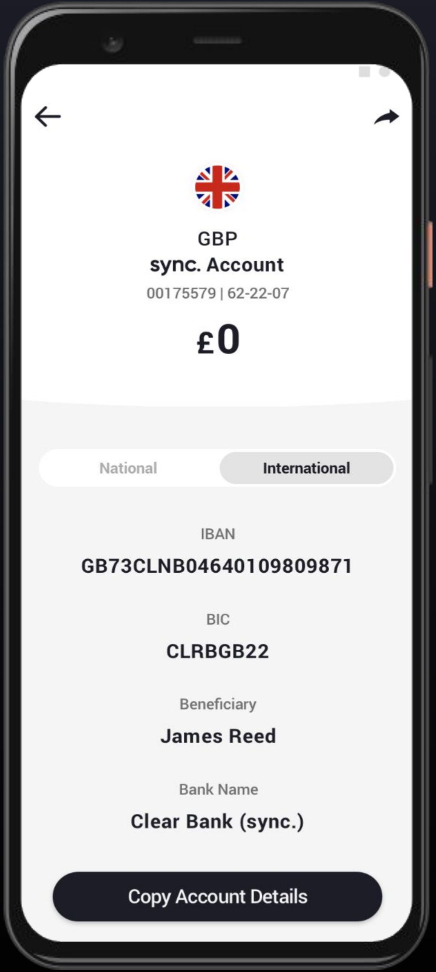

GBP
sync. Account
00175579 | 62-22-07
£0

Account Number
78128231

Sort Code
37-30-11

Beneficiary
James Reed

Bank Name
Clear Bank (sync.)


GBP
sync. Account
00175579 | 62-22-07
£0

IBAN
GB73CLNB04640109809871

BIC
CLRBGB22

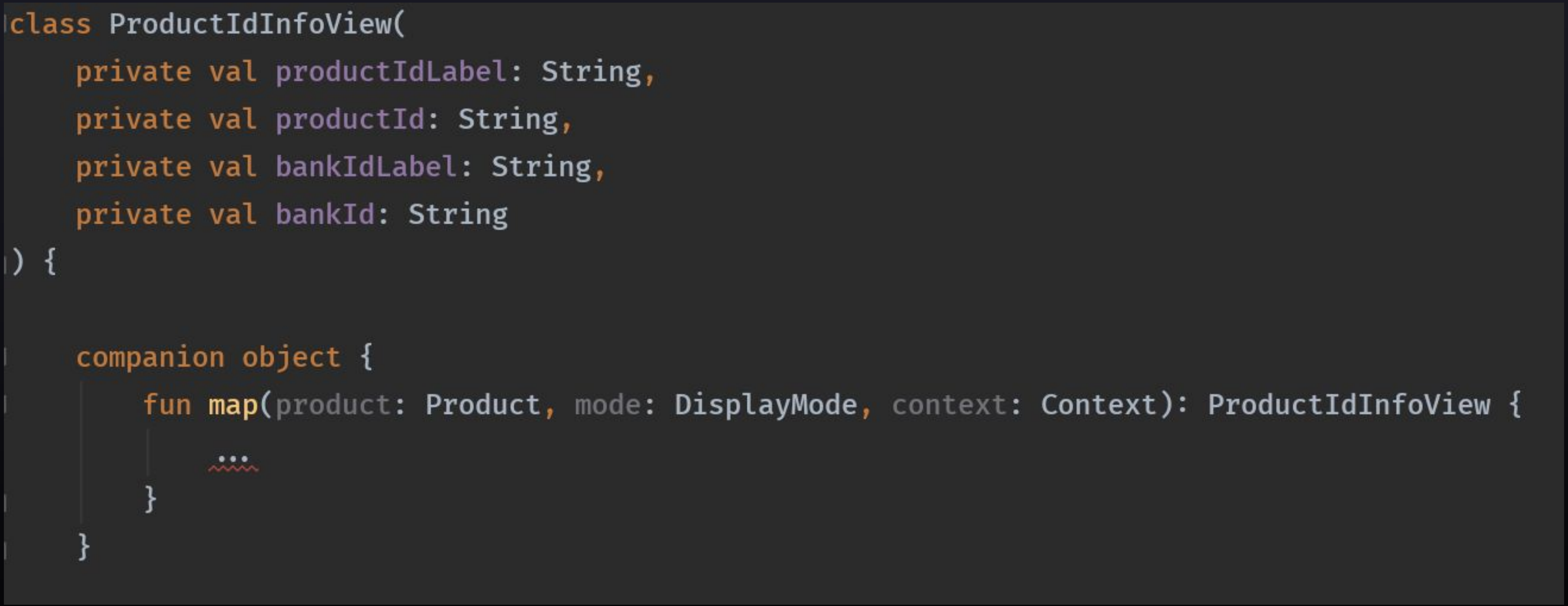
Beneficiary
James Reed

Bank Name
Clear Bank (sync.)

Málaga

MODEL MAPPING

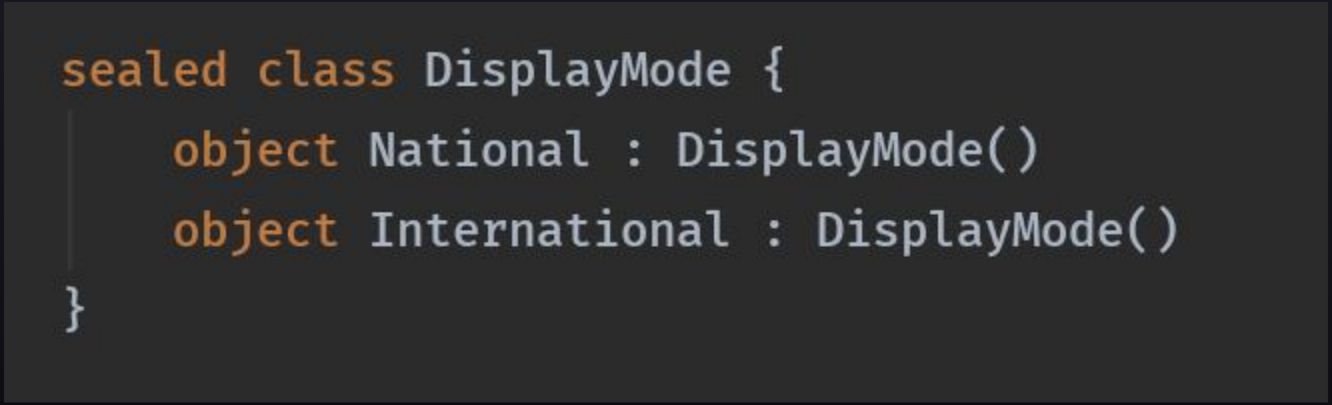
18th June Twenty**20**



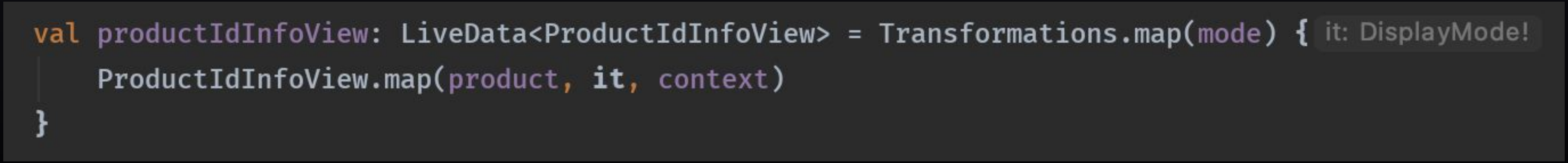
```kotlin
class ProductIdInfoView(
    private val productIdLabel: String,
    private val productId: String,
    private val bankIdLabel: String,
    private val bankId: String
) {

    companion object {
        fun map(product: Product, mode: DisplayMode, context: Context): ProductIdInfoView {
            ...
        }
    }
```

Málaga

MODEL MAPPING

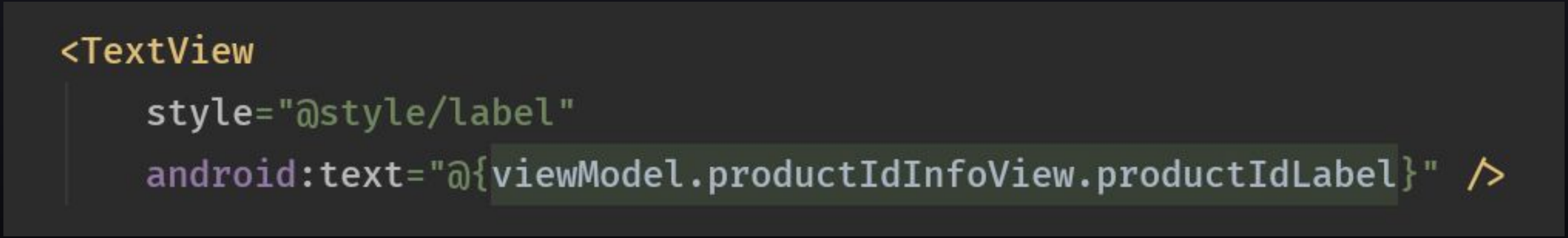


```kotlin
sealed class DisplayMode {
    object National : DisplayMode()
    object International : DisplayMode()
}
```

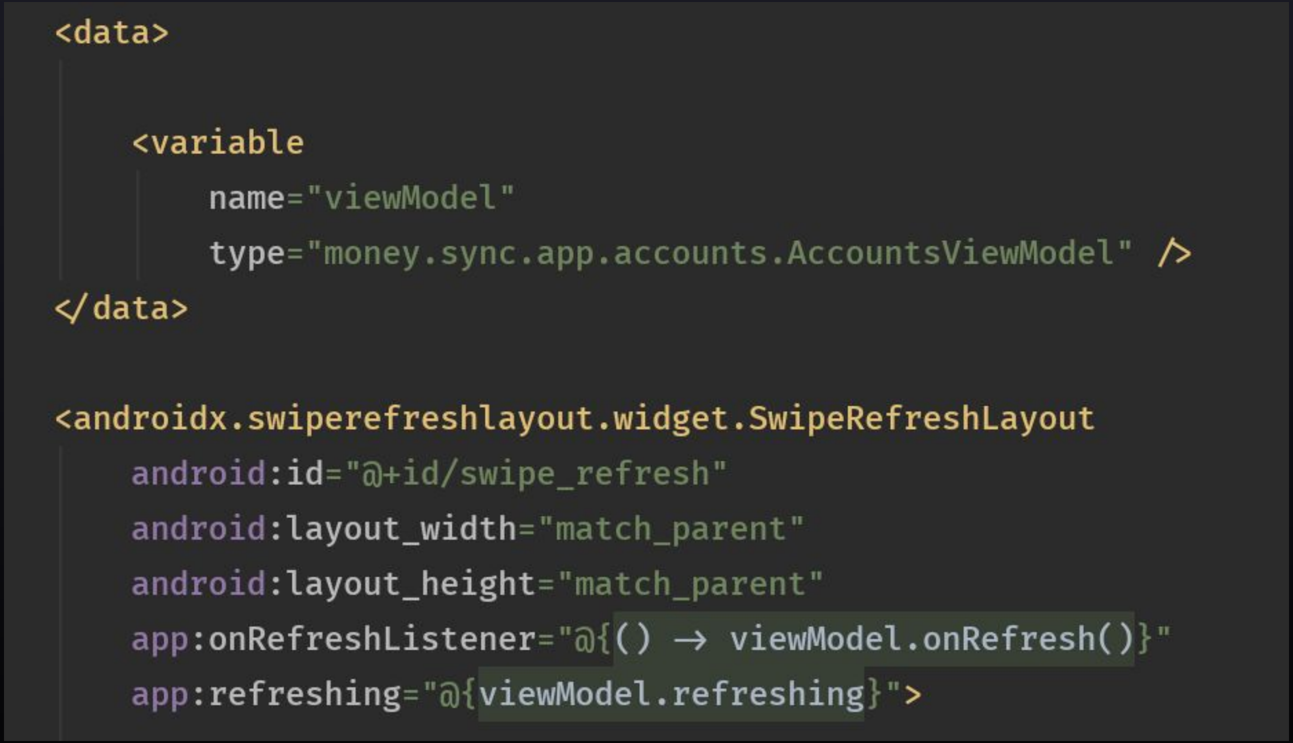


```kotlin
val productIdInfoView: LiveData<ProductIdInfoView> = Transformations.map(mode) { it: DisplayMode!
    ProductIdInfoView.map(product, it, context)
}
```

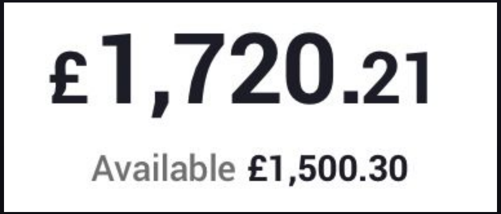
DATA BINDING

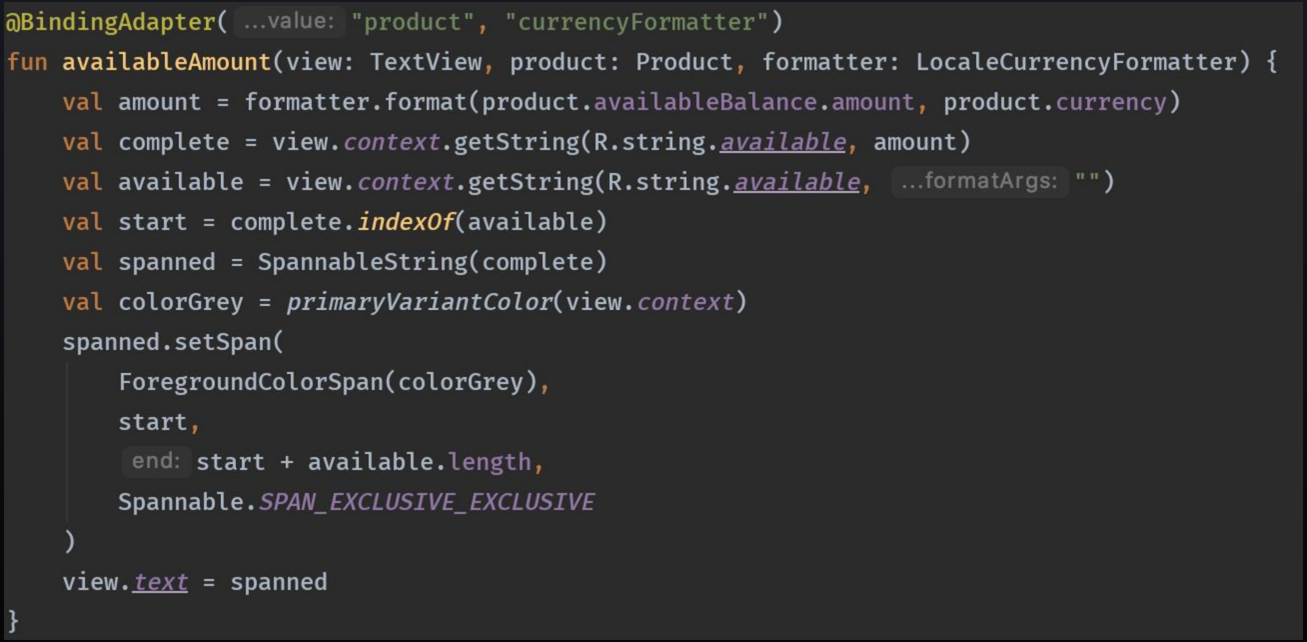


```
<TextView
    style="@style/label"
    android:text="@{viewModel.productIdInfoView.productIdLabel}" />
```



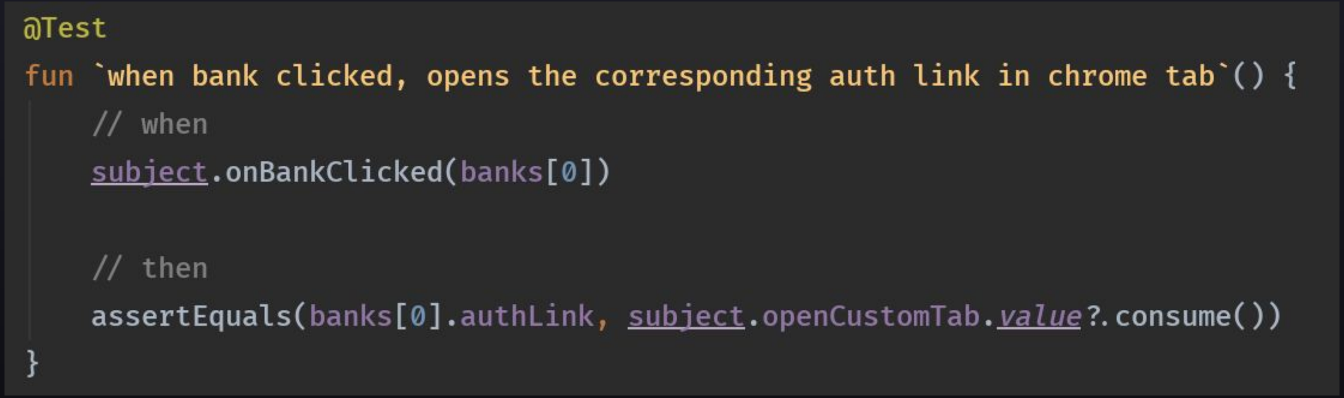
```xml
<data>


    <variable
        name="viewModel"
        type="money.sync.app.accounts.AccountsViewModel" />
</data>

<androidx.swiperefreshlayout.widget.SwipeRefreshLayout
    android:id="@+id/swipe_refresh"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:onRefreshListener="@{() -> viewModel.onRefresh()}"
    app:refreshing="@{viewModel.refreshing}">
```



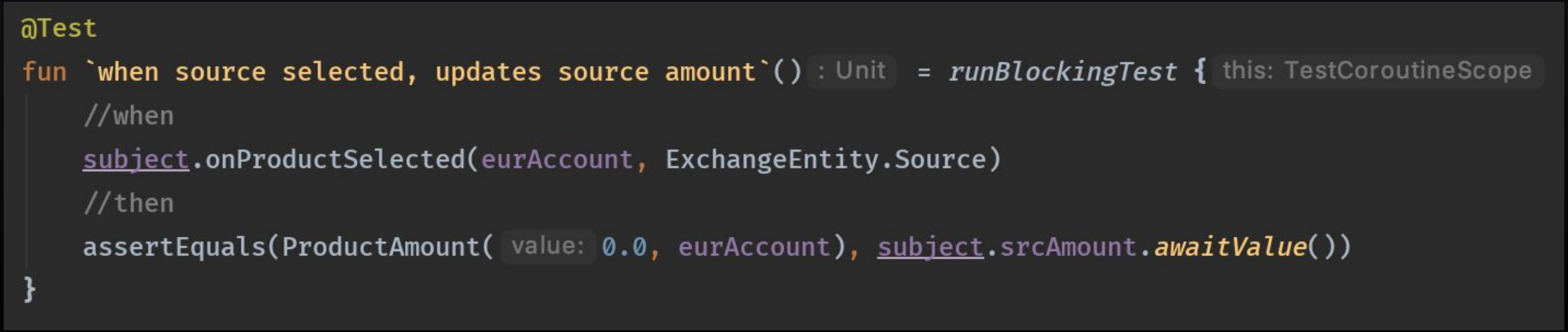


```kotlin
@BindingAdapter( ...value: "product", "currencyFormatter")
fun availableAmount(view: TextView, product: Product, formatter: LocaleCurrencyFormatter) {
    val amount = formatter.format(product.availableBalance.amount, product.currency)
    val complete = view.context.getString(R.string.available, amount)
    val available = view.context.getString(R.string.available, ...formatArgs: "")
    val start = complete.indexOf(available)
    val spanned = SpannableString(complete)
    val colorGrey = primaryVariantColor(view.context)
    spanned.setSpan(
        ForegroundColorSpan(colorGrey),
        start,
        end: start + available.length,
        Spannable.SPAN_EXCLUSIVE_EXCLUSIVE
    )
    view.text = spanned
}
```

TESTING

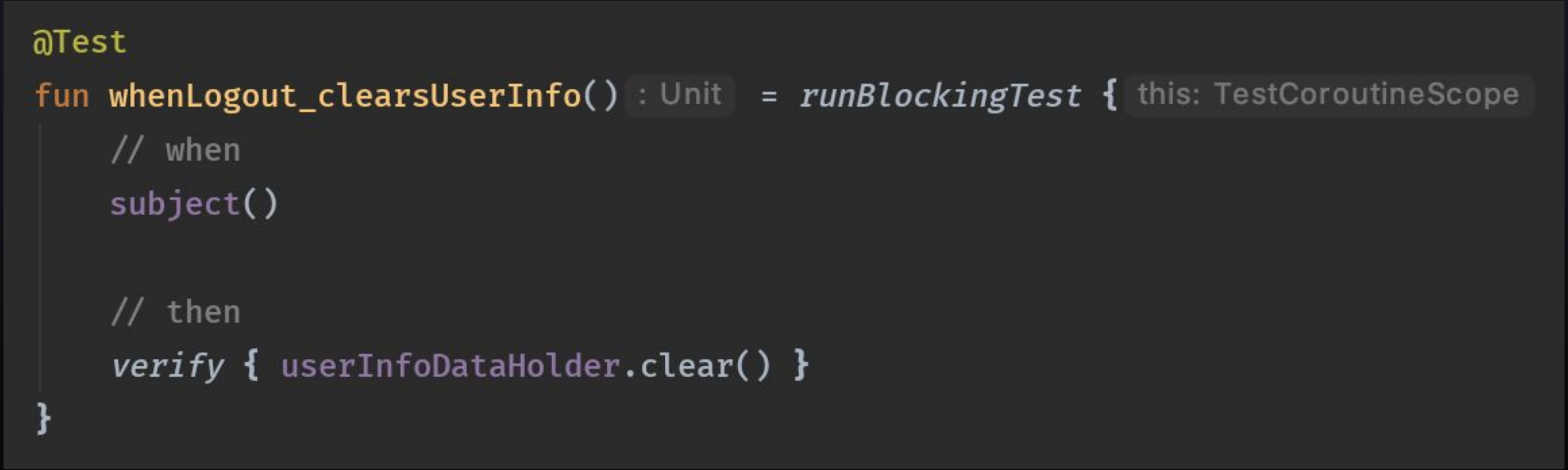


```kotlin
@Test
fun `when bank clicked, opens the corresponding auth link in chrome tab`() {
    // when
    subject.onBankClicked(banks[0])

    // then
    assertEquals(banks[0].authLink, subject.openCustomTab.value?.consume())
}
```



```kotlin
@Test
fun `when source selected, updates source amount`() : Unit  = runBlockingTest { this: TestCoroutineScope
    //when
    subject.onProductSelected(eurAccount, ExchangeEntity.Source)
    //then
    assertEquals(ProductAmount( value: 0.0, eurAccount), subject.srcAmount.awaitValue())
}
```

Málaga



```
@Test
fun whenLogout_clearsUserInfo() : Unit  = runBlockingTest { this: TestCoroutineScope
    // when
    subject()

    // then
    verify { userInfoDataHolder.clear() }
}
```

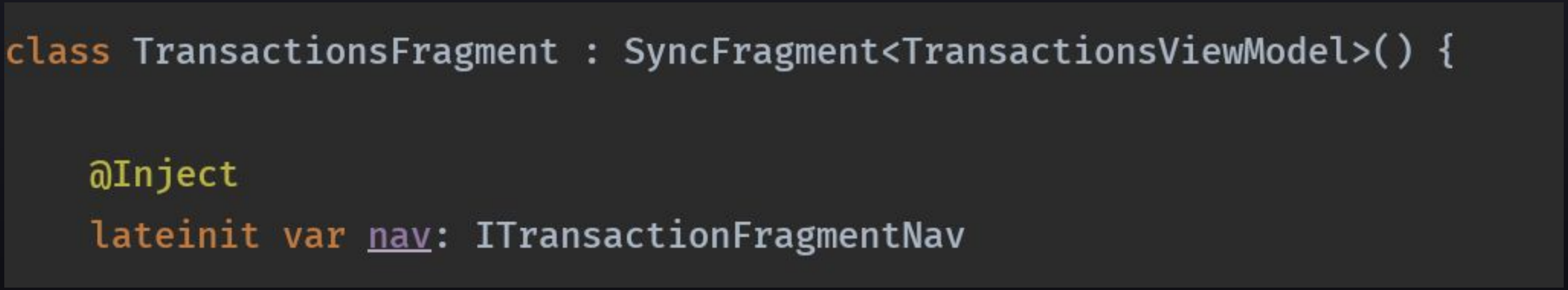
18th June Twenty**20**

NAVIGATION

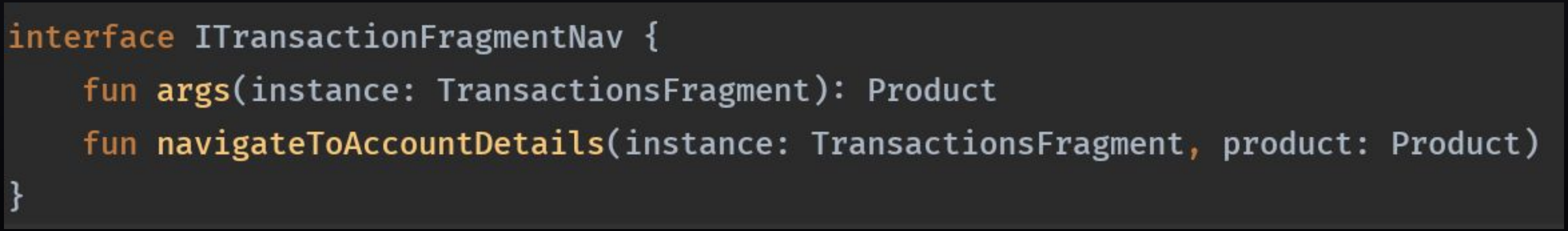
Málaga

- Navigation component
- Plugin SafeArgs
- Modular
- Isolated

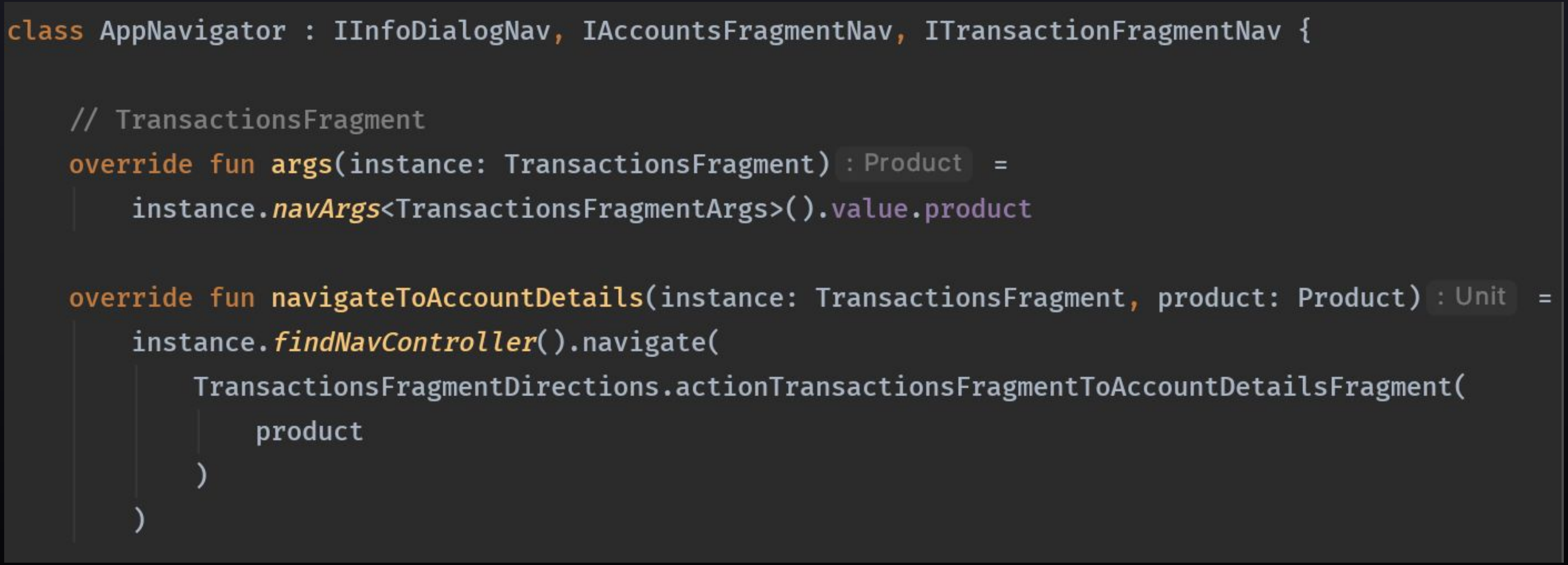
NAVIGATION



```kotlin
class TransactionsFragment : SyncFragment<TransactionsViewModel>() {

    @Inject
    lateinit var nav: ITransactionFragmentNav
```



```kotlin
interface ITransactionFragmentNav {
    fun args(instance: TransactionsFragment): Product
    fun navigateToAccountDetails(instance: TransactionsFragment, product: Product)
}
```



```
class AppNavigator : IInfoDialogNav, IAccountsFragmentNav, ITransactionFragmentNav {

    // TransactionsFragment
    override fun args(instance: TransactionsFragment) : Product  =
        instance.navArgs<TransactionsFragmentArgs>().value.product

    override fun navigateToAccountDetails(instance: TransactionsFragment, product: Product) : Unit  =
        instance.findNavController().navigate(
            TransactionsFragmentDirections.actionTransactionsFragmentToAccountDetailsFragment(
                product
            )
        )
```

18th June Twenty**20**

COMING SOON

Málaga

- Room
- Pagination
- Flow

Forget framework

Use it on your benefit, don’t let it drive you

TAKEAWAYS

On going (no silver bullets)

18th June Twenty**20**

WE ARE HIRING

rafa.o.es@sync.money

sync.

Málaga