Data Binding

Francisco Cavedon
Coordenador Técnico @ B2W Digital
francisco.cavedon@b2wdigital.com

B2W Digital

Informação Institucional Pendente de Aprovação

Agenda

1. Sobre Data Binding
2. Configuração
3. Data Binding na Prática
4. Expression Language
5. Two-Way Binding
6. Binding Adapter
7. Exemplo de Arquitetura
8. Dúvidas?

O que é Data Binding?

1. Lançado no I/O 2015
2. “Angular para Android”
3. Layouts declarativos
4. Minimiza a necessidade de “cola” entre layout e lógica

Porque usar?

1. Separação de responsabilidades
    1. Lógica de visualização fica no layout
2. Testabilidade
3. Simplificar a relação layout-código
    1. Chega de findViewById!
    2. Expression Language

Configuração

```
android {
…
    dataBinding {
        enabled = true
    }
}
```

Data Binding na Prática

```
<layout>
    <android.support.design.widget.CoordinatorLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:fitsSystemWindows="true"
        tools:context=".activity.MainActivity">

        <include
            android:id="@+id/include_search_appbar"
            layout="@layout/include_search_appbar"/>

        <fragment
            android:id="@+id/home_container"
            android:name=".fragment.HomeFragment"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            tools:layout="@layout/fragment_home" />

    </android.support.design.widget.CoordinatorLayout>
</layout>
```

Data Binding na Prática

```
<layout>
    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:contentInsetEnd="8dp"
            app:contentInsetStart="8dp"
            app:popupTheme="@style/AppTheme.PopupOverlay">

…

        </android.support.v7.widget.Toolbar>
    </android.support.design.widget.AppBarLayout>
</layout>
```

Data Binding na Prática

```
ActivityMainBinding activityMainBinding;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    activityMainBinding = DataBindingUtil.setContentView(this, R.layout.activity_main);
    Toolbar toolbar = activityMainBinding.includeSearchAppbar.toolbar;
    setSupportActionBar(toolbar);
}
```

Data Binding na Prática

```
<layout>
    <data>
        <variable name=“filterItem” type="model.item.filter.FilterItemViewModel"/>
    </data>

   <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <TextView android:id="@+id/filter_name"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@{filterItem.name}"/>

        <TextView android:id="@+id/filter_item_quantity"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@{filterItem.itemQuantity}"/>

    </RelativeLayout>
</layout>
```

Data Binding na Prática

```
public class SampleFilter extends RelativeLayout {

    ViewFilterItemBinding filterItemBinding;

    public SampleFilter(Context context) {
        super(context);

        filterItemBinding = DataBindingUtil.inflate(LayoutInflater.from(context),
R.layout.view_filter_item, null, true);
    }

    public void setFilter(FilterItemViewModel viewModel) {
        filterItemBinding.setFilterItem(viewModel);
    }
}

…

SampleFilter sampleFilter = new SampleFilter(context);
sampleFilter.setFilter(viewModel);
```

Expression Language

```
<TextView
    android:id="@+id/review_result_positive_feedback"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@{@plurals/feedback_count(viewModel.positiveFeedback)}"
    tools:text="10 pessoas curtiram isso" />
```

```
<TextView
    android:id="@+id/rating_total_count"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@{@string/rating_total_count(viewModel.totalReviewCount)}"
    tools:text="287 avaliacoes" />

<TextView
    android:id="@+id/review_banner"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/product_rating_recommended_product"
    android:visibility="@{viewModel.isRecommended? View.VISIBLE: View.GONE}" />
```

Two-Way Binding

```
public class AccountStepViewModel extends BaseObservable {

    @Bindable
    public String getEmail() {
        return accountRequest.getId();
    }

    @Bindable
    public String getPassword() {
        return accountRequest.getPassword();
    }

    public void setEmail(String email) {
        this.accountRequest.setId(email);
        notifyPropertyChanged(BR.email);
    }

    public void setPassword(String password) {
        this.accountRequest.setPassword(password);
        notifyPropertyChanged(BR.password);
    }
}
```

```
<android.support.design.widget.TextInputEditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="@={account.email}"
    android:singleLine="true"
    android:inputType="textEmailAddress"
    android:hint="E-mail"/>
```

Binding Adapter

```
@Bindable
public String getImageUrl() {
    return line.getProductImage();
}

@BindingAdapter({"imageUrl"})
public static void loadImage(ImageView view,
String imageUrl) {
    Picasso.with(view.getContext())
            .load(imageUrl)
            .into(view);
}
```

```
<ImageView
    android:id=“@+id/item_image”
    android:layout_width="94dp"
    android:layout_height="94dp"
    app:imageUrl="@{viewModel.imageUrl}"
    tools:src="@mipmap/ic_launcher" />
```

Binding Adapter

```java
@BindingAdapter({"font"})
public static void setFont(TextView textView, String fontName) {
    if (fontMap.isEmpty()) {
        AssetManager assetManager = textView.getContext().getAssets();

        fontMap.put("hind-bold", Typeface.createFromAsset(assetManager, "fonts/Hind-Bold.ttf"));
        fontMap.put("hind-light", Typeface.createFromAsset(assetManager, "fonts/Hind-Light.ttf"));
        fontMap.put("hind-medium", Typeface.createFromAsset(assetManager, "fonts/Hind-Medium.ttf"));
        fontMap.put("hind-regular", Typeface.createFromAsset(assetManager, "fonts/Hind-Regular.ttf"));
        fontMap.put("hind-semibold", Typeface.createFromAsset(assetManager, "fonts/Hind-Semibold.ttf"));
    }

    textView.setTypeface(fontMap.get(fontName));
}
```

```xml
<TextView
    android:id="@+id/product_basic_info_name"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@id/product_basic_info_seller_label"
    app:font="@{`hind-semibold`}"
    android:text="@{viewModel.productName}"
    android:textColor="@color/black"
    android:textSize="15sp"
    android:textStyle="bold"
/>
```

Exemplo de Arquitetura

A. Dentre arquiteturas mais conhecidas, temos MVC (Model-View-Controller), MVP (Model-View-Presenter) e MVVM (Model-View-View Model)

B. Optamos por seguir o modelo MVVM, com o Databinding fazendo papel de VM

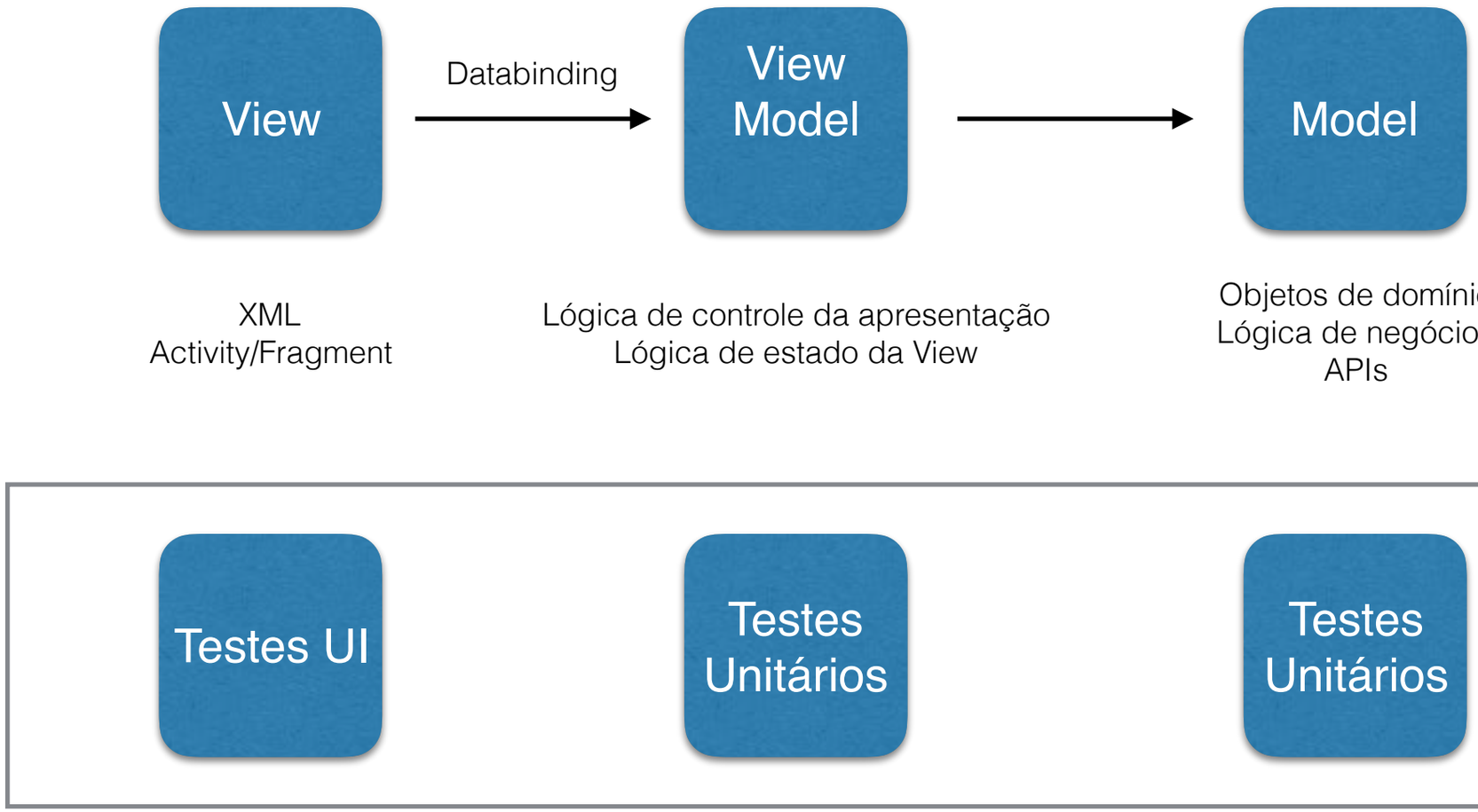
C. Neste modelo, um VM é uma classe que representa uma tela qualquer, contendo todo o seu estado

D. Essa abordagem nos dá: Testabilidade, Menos Código, Separação de Conceitos

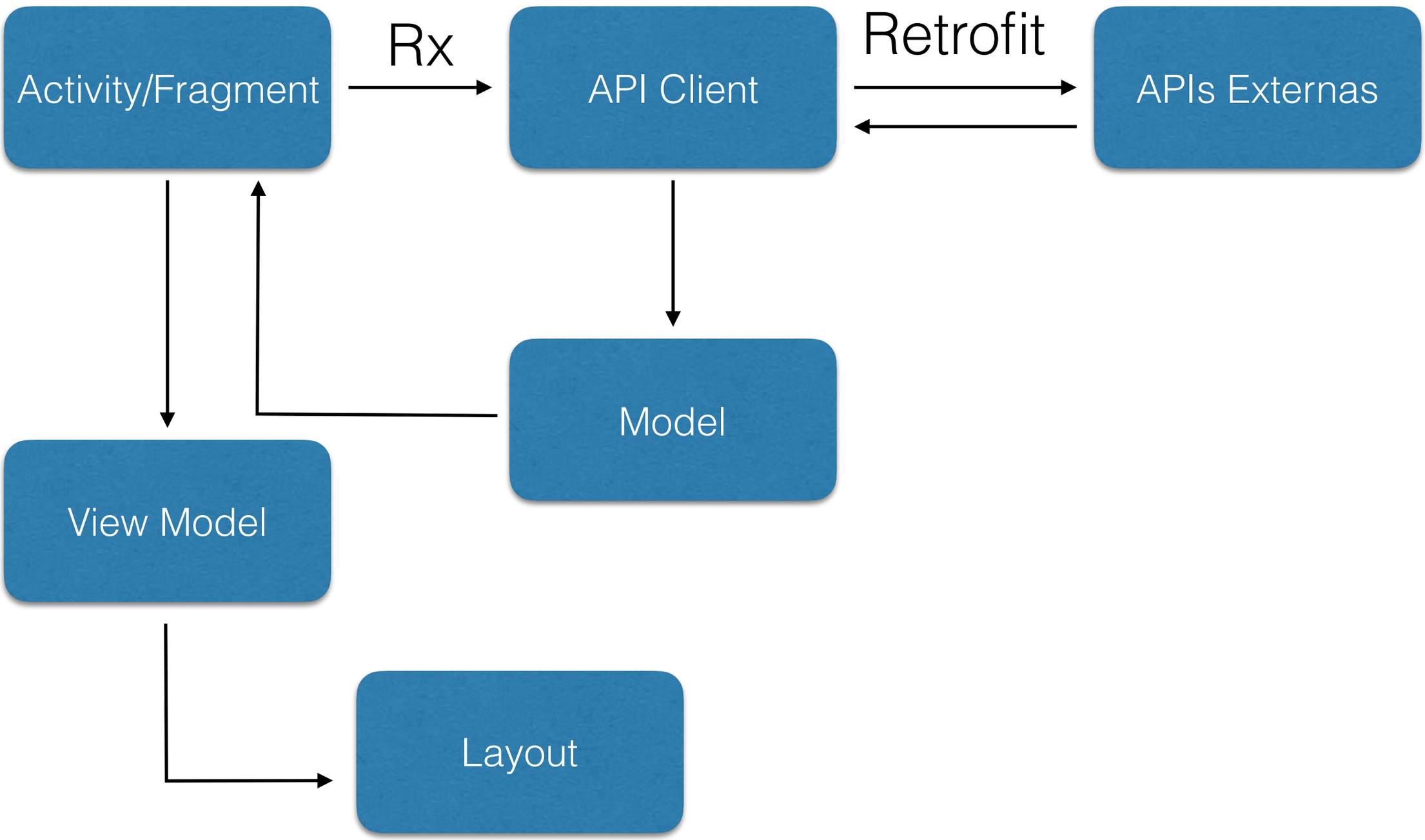
Exemplo de Arquitetura

1. Testabilidade:
    1. testes unitários nas classes de VM garantem a exibição correta dos layouts
2. Menos código
    1. muito menos código é necessário para integrar lógica e layout
    2. não só exibição de valores, mas também handlers, estilização e coisas mais avançadas
3. Separação de conceitos
    1. a fronteira entre lógica de visualização e lógica de negócios fica bem mais definida, sem “vazamento” para outras camadas

Exemplo de Arquitetura



Exemplo de Arquitetura



Dúvidas?

Obrigado!