Android Data Binding

Yeonhwa Woo

- 이름: 우연화
- 성별: 남. 남. 남. 남. 남. 남. 남.
- 개발을 하고 있습니다.
  다양한 것들에 관심이 많습니다.
  뒹굴거립니다.
  재밌는걸 좋아합니다.
- 블로그: http://novafactory.net (올라오는 글이 없습니다.)
  이메일: yhwoo.croute@gmail.com

Previously…

-
  ```
  TextView textView = (TextView)findViewById(R.id.text_view);
  textView.setText(“text”);
  ```

-
  ```
  public View getView(…) {
      Holder h;
      if (convertView == null) {
          …
          h = new Holder();
          convertView = inflater…;
          convertView.setTag(h);
          h.textView = (TextView) convertView.findViewById(R.id…);
      } else {
          …
          h = (Holder) convertView.getTag();
      }
      return convertView;
  }
  ```

- RecyclerView

Data binding

- ~~findViewById(…)~~
  ~~setText~~
  ~~setImage…~~

- onCreate In Activity
  LayoutNameBinding binding = DataBindingUtil.setContentView(..);

- onViewCreated In Fragment
  LayoutNameBinding binding = DataBindingUtil.bind(..);

Require

- **Android 2.1 (API 7) +**
- Android Gradle Plugin **1.3.0-beta1** or higher
- Data binding Code-completion and layout-preview support in **Android Studio 1.3+**

root/build.gradle

```
• buildscript {
      repositories {
          jcenter()
      }
      dependencies {
          classpath "com.android.tools.build:gradle:1.3.0-beta2"
          classpath "com.android.databinding:dataBinder:1.0-rc0"
      }
  }

  allprojects {
      repositories {
          jcenter()
      }
  }
```

root/app/build.gradle

- apply plugin: 'com.android.databinding'

Basic DataBinding

Data class

-
  ```
  public class User {
      private final Drawable profile;
      private final String firstName;
      private final String lastName;

      public User(Drawable profile, String firstName, String lastName) {
          this.profile = profile;
          this.firstName = firstName;
          this.lastName = lastName;
      }

      public Drawable getProfile() {
          return profile;
      }

      public String getFirstName() {
          return this.firstName;
      }

      public String getLastName() {
          return this.lastName;
      }
  }
  ```

Layout

```
•   <data>
        <variable name="user" type="mypackage.User" />
    </data>

 …
        <ImageView
            android:id="@+id/user_iv_profile"
            android:layout_width="200dp"
            android:layout_height="200dp"
            android:adjustViewBounds="true"
            android:scaleType="centerCrop"
            android:src="@{user.profile}"/>

        <TextView
            android:id="@+id/user_tv_first_name"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_toRightOf="@id/user_iv_profile"
            android:layout_toEndOf="@id/user_iv_profile"
            android:text="@{user.firstName}" />

        <TextView
            android:id="@+id/user_tv_last_name"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_toEndOf="@id/user_iv_profile"
            android:layout_toRightOf="@id/user_iv_profile"
            android:layout_below="@id/user_tv_first_name"
            android:text="@{user.lastName}" />
    </RelativeLayout>
 …
```

Auto-generated LayoutNameBinding

- Layout file: user.xml

- Auto-generated binding class: UserBinding
(mypackage.databinding.UserBinding)

implementation data binding code in Activity or Fragment

```
• public class UserFragment extends Fragment {

      private User user;

      @Override
      public void onCreate(@Nullable Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          user = new User(ContextCompat.getDrawable(getActivity(), R.drawable.iu), "IU", "아이유");
      }

      @Nullable
      @Override
      public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
          return inflater.inflate(R.layout.user, container, false);
      }

      @Override
      public void onViewCreated(View view, @Nullable Bundle savedInstanceState) {
          super.onViewCreated(view, savedInstanceState);
          UserBinding binding = DataBindingUtil.bind(view);
          binding.setUser(user);
      }
  }
```

Observable

- BaseObservable

Data class

-
  ```
  public class ObservableUser extends BaseObservable {
      …

      @Bindable
      public Drawable getProfile() {
          return profile;
      }
      public void setProfile(Drawable profile) {
          this.profile = profile;
          notifyPropertyChanged(net.novafactory.example.BR.profile);
      }

      @Bindable
      public String getFirstName() {
          return firstName;
      }
      public void setFirstName(String firstName) {
          this.firstName = firstName;
          notifyPropertyChanged(net.novafactory.example.BR.firstName);
      }

      @Bindable
      public String getLastName() {
          return lastName;
      }
      public void setLastName(String lastName) {
          this.lastName = lastName;
          notifyPropertyChanged(net.novafactory.example.BR.lastName);
      }
  }
  ```

Observable

- BaseObservable
- ObservableFields

Data class

```
• public class ObservableFieldsUser extends BaseObservable {

     @Bindable
    public final ObservableField<Drawable> profile = new ObservableField<>();

    @Bindable
    public final ObservableField<String> firstName = new ObservableField<>();

    @Bindable
    public final ObservableField<String> lastName = new ObservableField<>();
•
 }
```

Observable

- BaseObservable
- ObservableFields
- Observable Collection

Observable Collection Fragment

-
  ```
  public class ObservableCollectionFragment extends Fragment {

      public final ObservableArrayMap<String, Object> user = new ObservableArrayMap<>();

      @Override
      public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          user.put("profile", ContextCompat.getDrawable(getActivity(), R.drawable.iu));
          user.put("firstName", "IU");
          user.put("lastName", "아이유");
      }

      @Nullable
      @Override
      public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
          return inflater.inflate(R.layout.observable_collection, container, false);
      }

      @Override
      public void onViewCreated(View view, @Nullable Bundle savedInstanceState) {
          super.onViewCreated(view, savedInstanceState);
          ObservableCollectionBinding binding = DataBindingUtil.bind(view);

          binding.setUser(user);
          …
      }
  }
  ```

RecyclerView

Data class and Layout

- Data class: User 재사용
- Item layout: user.xml 재사용

ViewHolder

-
  ```
  public class ViewHolder extends RecyclerView.ViewHolder {

      ImageView ivProfile;
      TextView tvFirstName;
      TextView tvLastName;

      public ViewHolder(View itemView) {
          super(itemView);

          ivProfile = (ImageView) itemView.findViewById(R.id.user_iv_profile);
          tvFirstName = (TextView) itemView.findViewById(R.id.user_tv_first_name);
          tvLastName = (TextView) itemView.findViewById(R.id.user_tv_last_name);
      }
  }
  ```

in RecyclerView Adapter

-
  ```
  @Override
  public ViewHolder onCreateViewHolder(ViewGroup viewGroup, int viewType) {
      View view = LayoutInflater.from(viewGroup.getContext()).inflate(R.layout.user,
  viewGroup, false);
      return new ViewHolder(view);
  }

  @Override
  public void onBindViewHolder(ViewHolder holder, int position) {
      User user = list.get(position);
      holder.ivProfile.setImageDrawable(user.getProfile());
      holder.tvFirstName.setText(user.getFirstName());
      holder.tvLastName.setText(user.getLastName());
  }
  ```

RecyclerView DataBinding

Data class and Layout

- Data class: ObservableUser 재사용
- Item layout: observable_user.xml 재사용

ViewHolder

-
  ```
  public class BindingHolder extends RecyclerView.ViewHolder {

      private ObservableUserBinding binding;

      public BindingHolder(View itemView) {
          super(itemView);
          binding = DataBindingUtil.bind(itemView);
      }

      public ObservableUserBinding getBinding() {
          return binding;
      }
  }
  ```

in RecyclerView Adapter

-
  ```
  @Override
  public BindingHolder onCreateViewHolder(ViewGroup viewGroup, int viewType) {
      View view = LayoutInflater.from(viewGroup.getContext()).
                              inflate(R.layout.observable_user, viewGroup, false);
      return new BindingHolder(view);
  }

  @Override
  public void onBindViewHolder(BindingHolder holder, int position) {
      ObservableUser user = list.get(position);
      holder.getBinding().setUser(user);
  }
  ```

in RecyclerView Adapter

-
  ```
  @Override
  public BindingHolder onCreateViewHolder(ViewGroup viewGroup, int viewType) {
      View view = LayoutInflater.from(viewGroup.getContext()).
                              inflate(R.layout.observable_user, viewGroup, false);
      return new BindingHolder(view);
  }

  @Override
  public void onBindViewHolder(BindingHolder holder, int position) {
      ObservableUser user = list.get(position);
      holder.getBinding().setUser(user);
  }
  ```

Image load

-
  ```
  <data>
      <variable
          name="image"
          type="String" />
  </data>

  <ImageView
      android:id="@+id/image_item_imageview"
      android:layout_width="300dp"
      android:layout_height="300dp"
      android:adjustViewBounds="true"
      android:scaleType="centerCrop"
      app:imageUrl="@{image}"
      app:error="@{@drawable/iu}">
  </ImageView>
  ```

-
  ```
  @Override
  public void onBindViewHolder(BindingHolder holder, int position) {
      String image = list.get(position);
      holder.getBinding().setImage(image);
  }
  ```

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

Others

Expression language

- Mathematical: + - / * %
  String concatenation: +
  Logical: && ||
  Binary: & | ^
  Unary: + - ! ~
  Shift: >> >>> <<
  Comparison: == > < >= <=
  instanceOf
  Grouping: ()
  Listerals: character, String, numeric, null
  Cast
  Method calls
  Field access
  Array access: []
  Ternary operator: ?:

Expression example ..

-
  ```
  android:text="@{String.valueOf(index + 1)}"
  android:visibility="@{age &lt; 13 ? View.GONE : View.VISIBLE}"
  android:transitionName='@{"image_" + id}'
  ```

- Collections
  ```
  <data>
      <import type="android.util.SparseArray"/>
      <import type="java.util.Map"/>
      <import type="java.util.List"/>
      <variable name="list" type="List<String>"/>
      <variable name="sparse" type="SparseArray&lt;String>"/>
      <variable name="map" type="Map&lt;String, String>"/>
      <variable name="index" type="int"/>
      <variable name="key" type="String"/>
  </data>
  …
  android:text="@{list[index]}"
  …
  android:text="@{sparse[index]}"
  …
  android:text="@{map[key]}"
  ```

Expression example ..

- Class, method call

```
<data>
    <import type="com.example.MyStringUtils"/>
    <variable name="user" type="com.example.User"/>
</data>
…
<TextView
    android:text="@{MyStringUtils.capitalize(user.lastName)}"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

- Casting

```
<TextView
    android:text="@{((User)(user.connection)).lastName}"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

available variable, interface or any class

-
  ```
  <data>
          ...
          <variable name="fragment" type="mypackage.Fragment" />

  </data>

  <Button
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:onClickListener="@{fragment.onViewClickListener}"/>
  ```

Null Coalescing Operator

- android:text="@{user.lastName != null ? user.lastName : user.firstName}”

- android:text="@{user.lastName ?? user.firstName}"

Data binding

- **장점**

  - findViewById 등 귀찮은 작업들을 안해도 된다.
  - 코드가 짧아진다.
  - Data class와 View의 재사용이 쉽다.
  - Annotation의 추가로 알아서 동작한다.
  - 데이터 변경시 일일이 뷰 업데이트를 하지 않아도 된다.
  - 예전 보다 좀 더 객체지향 프로그래밍을 하는것 같다.

- **단점**

  - layout xml 에 data 관련 variable 들을 추가해주어야 한다.
  - ~Binding 으로 감싸여져 있고, 알아서 동작하므로
  뭘 하는 코드인지가 한 눈에 잘 보이지 않는다.
  - Annotation을 추가해야한다.
  - 예전 보다 좀 더 객체지향 프로그래밍을 하는것 같다.
  - 아직 xml 에서 자동완성이 잘 되질 않는다.. (java 파일에서는 잘 되는듯)

?

- https://github.com/croute/DataBindingExample
- https://www.dropbox.com/s/e8h9eu12hvn72da/Databinding.key?dl=0
- yhwoo.croute@gmail.com
- http://novafactory.net