**VISION ANDROID**

A PROJECT REPORT

Submitted by

**PIUSH KUMAR**

Under the guidance of

Mr PRATAP TALLA

In partial fulfilment of award of degree

OF

Bachelor of Technology

IN

COMPUTER SCIENCE and ENGINEERING

**NATIONAL INSTITUTE OF TECHNOLOGY, DURGAPUR**

Summary

Mobile application named Great work which is developed by O.C.Tanner Company.

Great work is authorized application which is used to appreciate the colleagues within the organization. This project was developed using java language. Our, task is to convert the existing java code to kotlin.

In present time developers using kotlin instead of java to their application. Since, there are many benefits of using kotlin. Java is old and slow modernize.

Kotlin is new open source language which is integrated with android studio. The main advantage of using kotlin is Nullpointerexception. Kotlin handles this exception at compile time through declaring a variable using ’?’

CONTENTS

Introduction

Description

Modification

Lombok package

Getter and Setter Property

Accessors Property

Build Property

AllArgsConstructor Property

NoArgsConstructor Property

NonNull Property

ToString Property

Changes in Lombok package(Table)

ButterKnife package

BindView

BindColor

OnClick

Optional

BindDrawable

Butteknife

OnTextChanged

Changes in ButterKnife package(Table)

Errors

GitHub

Modified Classes

Introduction: Convert java code to kotlin code in project vision-android.

Description:

Since, kotlin does not support “Lombok package”. So, we removed all the “Lombok package” from every class and generated a code based on removed property.

Within Lombok package we have

Getter

Setter

Accessors

Builder

AllArgsConstructor

NoArgsConstructor

NonNull

ToString

After removing Lombok package throughout the project, replaced apt by kapt in “build.gradle(app)” file as

Applying plugin: “kotlin-kapt”

After that we convert the classes in to kotlin module wise.

Since, butterknife is also not handle by kotlin. So, we removed “Butterknife package” also.

Within butterknife package we have

BindView

BindColor

OnClick

Optional

Butterknife

BindDimen

BindDrawable

OnTextChanged

Modification:

Lombok package

1. Getter and Setter property

@Getter

@Setter

String image Url;

Remove @Getter and @Setter which is assigned for the variables and generates code for the variable.

String imageUrl;

Public String getImageUrl(){

Return imageUrl;

}

Public String setImageUrl(String imageUrl){

This.imageUrl = imageUrl;

}

Finally, Remove import statement (import Lombok Getter and import Lombok Setter)

2. Accessors property

@Accessors(prefix=’m’)

@Setter

@Getter

ResponseOption mResponseOption

When we generate the code for mResponseOption, we get

Public ResponseOption getmResponseOption(){

Return mResponseOption;

}

Public void setmResponseOption(ResponseOption mResponseOption){

This. mResponseOption = mResponseOption;

}

But while generating code for variables, m is present automatically. So, we remove m in get and set property.

Public ResponseOption getResponseOption(){

Return mResponseOption;

}

Public void setResponseOption(ResponseOption mResponseOption){

this. mResponseOption = mResponseOption;

}

Finally, remove import statement (import Lombok.expermental.Accessors)

3. Build property

Builder is used to generate object automatically.

QuestionPage page = QuestionPage.builder().position(position).page Title(page Title).build

Remove builder and generate object using ‘new’ keyword

QuestionPage page = new QuestionPage();

Page.setPosition(position);

Page.setPage Title(page Title);

Finally, remove import statement (import Lombok.builder)

4. AllArgsC onstructor

Used to generate parameterized constructor automatically.

-@AllArgsConstructor(supressConstructorProperties = true)

Public class UserInfo extends User implements Hasselection, Serializable{

}

Remove -@AllArgsConstructor(supressConstructorProperties = true) and create new constructor

Public class UserInfo extends User implements Hasselection, Serializable{

//constructor

Public UserInfo(String name, String id, String thumb){

this.name = name;

this.id = id;

this.avatar = new Avatar(null, null, new Avatar.Info(thumb, null));

}

}

5. NoArgsConstructor

Used to generate default constructor automatically.

@NoArgConstructor

Public class QuestionPage{

}

Remove @NoArgsConstructor for the class and add code manually create the default constructor

Public class QuestionPage{

QuestionPage(){

}

}

Then, remove the import statement(import Lombok.NoArgsConstructor)

6. NonNull property

Since it simply throw a null exception so, we can simply remove @NonNull.

And then, remove import statement (import Lombok.NonNull)

7. ToString property

Since, it is used to represent the object as string. So, we simply remove @ToString.

Then, remove import statement.

Changes in Lombok package

|  |  |
| --- | --- |
| Property Name | Changes |
| Getter and Setter | Remove @Getter and @Setter which is assigned and generates code for the variable. |
| Accessors | Remove m |
| Build | Remove builder and generate object using keyword ‘new’ |
| AllArgsConstructor | Create constructor manually |
| NoArgsConstructor | Create default constructor manually |
| NonNull | Simply remove @NonNull |
| ToString | Simply remove @ToString |

ButterKnife package

1. BindView

@BindView(R.id.search\_view)

CatalogSearchView mSearchView;

Used to bind the id of views present in layout file into class for above code mSearchView instead of id name.

mSearchView??.searchViewClickListerner = searchViewClickListerner

We remove above code and use id search\_view directly instead of using mSeachView.

search\_view .searchViewClickListerner = searchViewClickListerner

2. BindColor

Generally bindcolor is used to assign a color to particular view

@BindColor(R.color.tanner\_gray\_400)

Int mFromBackgroundColor;

In our file we had BindColor as above but we do not use anywhere. So, we simply remove the above code and import statement.

3. OnClick

We use OnClickListener instead of onclick annotation of butterknife.

@OnClick(R.id.searchQuickAction)

Public void on SearchQuickAction(){

hideSearchView(true);

}

So, we change above by using OnClickListener for the view id and we need to call the function initially in onViewCreated or init or etc.

Public void on SearchQuickAction(){

searchQuickAction.setOnClickListener(){

hideSearchView(true);

}

}

4. Optional

@Optional

Public void onDeclineOrPrepareClick(){

Decline\_or\_prepare.setOnClickListener(){

}

}

Simply remove @ Optional for functions and also import statement.

Public void onDeclineOrPrepareClick(){

Decline\_or\_prepare.setOnClickListener(){

}

}

5. BindDrawable

@BindDrawable(R.drawable.ic\_quantity\_plus\_selector)

Drawable mQuantityPlusBackground;

Since we don’t used drawable variable anywhere we simply remove it.

6. ButterKnife

ButterKnife.bind(this)

Above code is used to minimize the existing code. Since, kotlin does not support butterknife. So, we simply remove the above code.

7. OnTextChanged

Instead of using @onTextChanged use addTextChangedListener and override the methods onTextChanged, AfterTextChanged, BeforeTextChanged. The code present inside function with annotation @onTextChanged is written inside method AfterTextChanged

Changes in ButterKnife package

|  |  |
| --- | --- |
| Property | Changes |
| BindView | We used id name instead of variable |
| BindColor | We do not used bindcolor so, we simply remove butterknife bindcolor |
| OnClick | We remove butterknife OnClick by using OnClickListener for the view id and we need to call the function initially in onViewCreated or init or etc. |
| Optional | Simply remove @ Optional for functions and also import statement. |
| BindDrawable | Since we don’t used drawable variable anywhere we simply remove it. |
| ButterKnife | Since, kotlin does not support butterknife. So, we simply remove the above code. |
| OnTextChanged | Instead of using @onTextChanged use addTextChangedListener |

ERRORS:

Dagger Error

Dagger error is due to dependency injection. So, we simply apply field injection.

App Crashing

After changing the code and installing the application on device it may crash

on some click or on some particular functioning.

It is because some method in modified code is not used or by changing name

are not modifying it in other file so, it generate mismatch and the app crash.

Use of Id

When we use id name directly for some variable which are assigned with bindView then, sometime we have to import some header where it id works. We can do it by pressing Alt+Enter on id.

Also, when converted class has reference to adapter we have to use ‘ItemView’ before the name of Id.

Run time Error

When code compile it successfully and also application is running fine. But it may have some error due to which app crashes. This is nothing but run time error.

For this we have check logcat and select error mode and find the file which has error and modify it.

Code migration process

Developers have to migrate the code what they had written or updated so that everyone in that project get the updated code. For this we use GitHub.

After changing the code we have to migrate it to GitHub and from there other member of the group fetch it.

Migration process is very simple just we need to do is write a short description and commit it. Then we have to ‘push Origin’. And for getting other update we have to fetch it and ‘pull origin’ on GitHub.

Symbol:

‘+’ this symbol in github represents that specific class or statement is removed to the project.

‘-‘this symbol in github represents that specific class or statement is added to the project.

‘o’ this symbol in github represents that modification had taken place in project.

Modified Class:

In ui

Fragments package

Class

AwardDetailsFragment

ECardDetailsFragment

NotificationsFragment

SocialWallDetailsFragment

SocialWallFollowDailogFragment

SocialWallLikedUsersFragment

SocialWallFragment

SocialWallViewsFragment

AlertFragments

EproductCancelReasonFragment

EproductCategoryFragment

EproductFavoritesFragment

EproductFragment

EproductTabsFragment

PeopleSearchDetailsFragment

PeopleSearchFragment

ShippingAddressesFragment

BankCountryBody

BankCountrySelectionActivity

CartFragment

FavoritesFragment

SuccessDialog

AlertFragment

Activities package

Class

BankCountryActivity

PeopleSearchActivity

Adapters package

Class

EProductCategoryAdapter

EProductCategoryView

EProductGridAdapter

EProductPagerAdapter

EProductViewHolder

OnProductClickListener

Section

SectionClickListener

SectionStateChangeListener

SectionViewHolder

AwardNomineesListAdapter

TaskAdapter

DashBoardPulseViewHolder

DashBoardSocialWallViewHolder

HistoryViewHolder

PendingHistoryViewHolder

PeopleSearchFollowingViewHolder

PeopleSearchNormalViewHolder

SocialWallCommentsViewHolder

SocialWallFollowViewHolder

SocialWallLikedUsersViewHolder

SocialWallViewHolder

AnswerAdapter

QuestionPageAdapter

AwardTypeAdapter

BankAdapter

CartAdapter

CatlogAdapter

CatlogFavoritesAdapter

ChoosePeoplePagerAdapter

ChoosePeoplesAdapter

CountryAdapter

FilterAdapter

HistoryTabsAdapter

NotificationsAdapter

OptionsAdapter

PairGuidePagerAdapter

PeopleSearchAdapter

PointOptionAdapter

PresenterOptionAdapter

RecognitionHistoryAdapter

ReviewersOptionAdapter

ShippingAddressAdapter

SocialWallAdapter

SocialWallCommentsAdapter

SocialWallFollowAdapter

SocialWallLikedUsersAdapter

SocialWallViewPagerAdapter

In util

Actionbar package

Class

ProgressBarHelper

ProgressIndicator

PropertyAnimator

SimpleAnimatorListener

CartItemComparator

CatalogComparato

CatalogGroupComparato

SectionComparato

CircleTransformation

RingTransformation

RoundedCornersTransformation

DrawableHelper

RetrofitException

RxErrorHandlingCallAdapterFactory

DashboardUtils