Популярные библиотеки

Червяков Алексей

Наш план

- Набор starter pack
- Набор advanced pack
- Бонус

Организационные моменты

Напоминание отметиться на портале

Starter Pack

Data

Data слой

- okhttp client
- retrofit
- gson converter
- glide
- picasso
- logging interceptor

OkHttp

```
implementation "com.squareup.okhttp3:okhttp:4.10.0"

OkHttpClient client = new OkHttpClient();

String run(String url) throws IOException {
  Request request = new Request.Builder()
      .url(url)
      .build();

  try (Response response = client.newCall(request).execute()) {
    return response.body().string();
  }
}
```

Gson

```
implementation 'com.google.code.gson:gson:2.10.1'
```

```
Gson gson = new Gson();

User user = gson.fromJson(json_string, User.class);
String json_string2 = gson.toJson(user);
```

```
Gson gson = new GsonBuilder()
    .registerTypeAdapter(Id.class, new IdTypeAdapter())
    .enableComplexMapKeySerialization()
    .serializeNulls()
    .setPrettyPrinting()
    .setVersion(1.0)
    .create();
```

retrofit

```
implementation 'com.squareup.retrofit2:retrofit:2.9.0

public interface GitHubService {
  @GET("users/{user}/repos")
  Call<List<Repo>> listRepos(@Path("user") String user);
}

Retrofit retrofit = new Retrofit.Builder()
    .baseUrl("https://api.github.com/")
    .build();

GitHubService service = retrofit.create(GitHubService.class);
```

Gson Converter

```
implementation 'com.squareup.retrofit2:converter-gson:2.9.0'
```

```
Retrofit retrofit = new Retrofit.Builder()
        .baseUrl("https://api.github.com")
        .addConverterFactory(GsonConverterFactory.create())
        .build();
```

Glide

```
implementation 'com.github.bumptech.glide:glide:4.14.2'
```

```
ImageView imageView = (ImageView)
findViewById(R.id.my_image_view);
Glide.with(this).load("http://goo.gl/gEgYUd").into(imageView);
```

Logging Interceptor

```
implementation"com.squareup.okhttp3:logging-interceptor:4.10.0"
```

```
HttpLoggingInterceptor logging = new HttpLoggingInterceptor(new
Logger() {
  @Override public void log(String message) {
    Log.d("client", message)
  }
});
```

Domain

Domain

- parcel plugin

Parcel plugin

```
apply plugin: 'kotlin-parcelize'


@Parcelize
class Country(
    val name: String
)
```

Presentation

Presentation

- core
- fragment
- annotation
- constraint layout
- recycler view
- coordinator layout
- material

Jetpack

```
implementation "androidx.core:core:1.9.0"
implementation "androidx.fragment:fragment:1.5.5"
implementation "androidx.annotation:annotation:1.5.5"
implementation "androidx.constraintlayout:constraintlayout:2.2.0-alpha07"
implementation "androidx.coordinatorlayout:coordinatorlayout:1.2.0"
implementation "androidx.recyclerview:recyclerview:1.2.1"
implementation 'com.google.android.material:material:5.0'
implementation 'androidx.lifecycle:lifecycle-viewmodel-ktx:2.5.1'
```

Common

Common

- ktx
- navigation component
- hilt

Ktx

```
implementation "androidx.core:core-ktx:1.9.0"


sharedPreferences.edit { putBoolean("key", value) }

sharedPreferences.edit(commit = true) { putBoolean("key",
value) }
```

Navigation

```
implementation "androidx.navigation:navigation-fragment:2.5.3"
implementation "androidx.navigation:navigation-ui:2.5.3"
```

```
  <?xml version="1.0" encoding="utf-8"?>
<navigation xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:android="http://schemas.android.com/apk/res/android"
    app:startDestination="@id/blankFragment">
    <fragment
        android:id="@+id/blankFragment"
        android:name="com.example.cashdog.cashdog.BlankFragment"
        android:label="@string/label_blank"
        tools:layout="@layout/fragment_blank" />
</navigation>
```

Hilt

```
apply plugin: 'kotlin-kapt'
apply plugin: 'com.google.dagger.hilt.android'

dependencies {
implementation "com.google.dagger:hilt-android:2.44"
  kapt "com.google.dagger:hilt-compiler:2.44"
}
```

Advanced Pack

Data

Data слой

- retrofit
- okhttp client
- glide
- ktx
- logging interceptor
- **room**
- **paginig3**

Room

```
implementation
"androidx.room:room-runtime:2.5.0"

kapt
"androidx.room:room-compiler:2.5.0
```

```
@Entity
data class User(@PrimaryKey val uid: Int)

@Dao
interface UserDao {
    @Query("SELECT * FROM user")
    fun getAll(): List<User>
}

@Database(entities = [User::class], version = 1)
abstract class AppDatabase : RoomDatabase() {
    abstract fun userDao(): UserDao
}
```

Paging3

```
implementation "androidx.paging:paging-runtime:3.1.1"
```

```
 interface FooRepository {
    fun fetchFoo(): Flow<PagingData<Foo>>
}
```

Presentation

Presentation

- jetpack
- fragment
- annotation
- constraint layout
- recycler view
- coordinator layout
- material
- **adapter delegate**
- **lottie**

Adapter delegate

```
implementation 'com.hannesdorfmann:adapterdelegates4:4.3.2'
```

```
    delegatesManager.addDelegate(CatAdapterDelegate(activity))
        .addDelegate(DogAdapterDelegate(activity))
        .addDelegate(GeckoAdapterDelegate(activity));
```

Lottie

```
implementation
'com.airbnb.android:lottie:6.0.0'
```

```
<com.airbnb.lottie.LottieAnimationView
       android:id="@+id/animation_view"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"

       app:lottie_rawRes="@raw/hello_world"
       // or
       app:lottie_fileName="hello_world.json"


       // Loop indefinitely
       app:lottie_loop="true"
       // Start playing as soon as the
animation is loaded
       app:lottie_autoPlay="true" />
```

Common

Common

- navigation component
- **firebase**
- **cicerone**
- **dagger**
- **timber**
- **kaspresso/espresso**

firebase

```
implementation platform('com.google.firebase:firebase-bom:31.2.2')
implementation 'com.google.firebase:firebase-crashlytics-ktx'

buildscript {
    dependencies {
        classpath 'com.google.firebase:firebase-crashlytics-gradle:2.9.4'
    }
}

plugin apply: 'com.google.firebase.crashlytics'
```

Cicerone

```
implementation("com.github.terrakok:cicerone:7.1.0")

class App : Application() {
    private val cicerone = Cicerone.create()
    val router get() = cicerone.router
    val navigatorHolder get() = cicerone.getNavigatorHolder()
}

router.navigateTo(SomeScreen())
```

Timber

```
implementation 'com.jakewharton.timber:timber:5.0.1'

  private static class CrashReportingTree extends Timber.Tree {
    @Override protected void log(
        int priority, String tag, @NonNull String message, Throwable t
        ) {
    }
  }

  Timber.plant(new CrashReportingTree());

   Timber.d("Activity Created");
```

Dagger

```
implementation 'com.google.dagger:dagger:2.45'
 annotationProcessor 'com.google.dagger:dagger-compiler:2.45'

@Component(modules = AppModule.class)
interface AppComponent {
  App app();

  @Component.Builder
  interface Builder {
    @BindsInstance Builder userName(@UserName String userName);
    AppComponent build();
  }
}
```

Kaspresso

```
dependencies {
    androidTestImplementation
'com.kaspersky.android-components:kaspresso:1.5.1'
}

@Test
fun shouldPassOnNoInternetScanTest() =
    beforeTest { }
    .afterTest { }
    .run { step("Open Simple Screen") { MainScreen { nextButton { click()
} } } }
```

Вопросы?

Rx vs Coroutines

Rx vs Coroutine

Что больше нравится

Пишем код

Оставьте отзыв!

Спасибо за внимание!

Червяков Алексей