## 프로젝트 생성

* Empty Views Activity 선택 >> MyFrameDagger

## 모듈 gradle 수정

1. androidManifest.xml : uses-permission INTERNET, usesCleartextTraffic
2. android 에 buildFeatures { viewBinding = true }
3. dependencies에 추가

**File >> Project Structure >> Dependencies >> Add Library Dependency**

* lifecycle-viewmodel-ktx 입력: androidx.lifecycle lifecycle-viewmodel-ktx Google ( 2.8.3 )
* lifecycle-livedata-ktx 입력 : androidx.lifecycle lifecycle-livedata-ktx Google ( 2.8.3 )
* retrofit 입력: com.squareup.retrofit2 retrofit Maven Central ( 2.11.0 )
* converter-gson 입력 : com.squareup.retrofit2 converter-gson Maven Central ( 2.11.0 )
* okhttp 입력: com.squareup.okhttp3 okhttp Maven Central ( 5.0.0-alpha.14 )
* logging-interceptor 입력: com.squareup.okhttp3 logging-interceptor Maven Central (상동)
* dagger 입력: com.google.dagger dagger Maven Central ( 2.51.1 )
* dagger-compiler 입력: com.google.dagger dagger-compiler Maven Central ( 2.51.1 )
* implementation >> kapt >> plugin에 id("kotlin-kapt") 추가
* glide 입력: com.github.bumptech.glide glide Maven Central ( 5.0.0-rc01 )
* jwtdecode 입력: com.auth0.android jwtdecode Maven Central ( 2.0.2 )

( <https://github.com/auth0/JWTDecode.Android> )

* androidx.room입력: Artifact Name에서 **room-compiler, room-ktx, room-runtime** 선택
* compiler implementation을 kapt 수정하고 plugin에 id("kotlin-kapt") 추가
* // Rx
* implementation 'io.reactivex.rxjava3:rxkotlin:3.0.0'
* implementation 'io.reactivex.rxjava3:rxandroid:3.0.0'

implementation 'com.squareup.retrofit2:adapter-rxjava3:2.9.0'

1. JAR/ARR Dependencies
   * ./libs/device.sdk.jar 추가

## Retrofit simple

. <https://futurestud.io/tutorials/retrofit-2-how-to-trust-unsafe-ssl-certificates-self-signed-expired>

. <https://0391kjy.tistory.com/46>

. <https://xmobile.tistory.com/entry/Android-SSL-%EC%9D%B8%EC%A6%9D%EC%84%9C-%EC%A0%81%EC%9A%A9%ED%95%B4%EB%B3%B4%EA%B8%B0>

. openssl s\_client -connect jsonplaceholder.typicode.com:443

## Retrofit My Web Api

. Retrofit Client 정의

class RetrofitInstance {  
  
 companion object {  
 private val retrofit by lazy {  
 val logger = HttpLoggingInterceptor()  
 //to be able to read the response body  
 logger.setLevel(HttpLoggingInterceptor.Level.BODY)  
 val client = OkHttpClient.Builder()  
 .protocols(listOf(Protocol.HTTP\_1\_1))  
 .addInterceptor(logger)  
 .connectTimeout(15, TimeUnit.SECONDS)  
 .readTimeout(15, TimeUnit.SECONDS)  
 .writeTimeout(15, TimeUnit.SECONDS)  
 .build()  
  
 Retrofit.Builder()  
 .baseUrl(BASE\_URL)  
 .addConverterFactory(GsonConverterFactory.create())  
 .client(client)  
 .build()  
 }  
  
 val api by lazy {  
 retrofit.create(ApiService::class.java)  
 }  
  
 }  
  
}

. DTO 정의 return data class 정의

. Api interface 정의

interface ApiService {  
 @POST("api/Auth/Login")  
 suspend fun authLogin (@Body loginRequest: LoginRequest) : Response<LoginResponse>  
  
 @POST("api/Db/ExecNonQuery")  
 suspend fun execNonQuery(@Body body: List<MyDbCommand>): Response<ExecReturn>  
  
 @POST("api/Db/GetDataSet")  
 suspend fun getDataSet(@Body body: MyDbCommand): Response<JsonObject>  
}

. JSON To Kotlin Plugin 설치

>> Settings >> Plugins >> Json to Kotlin 찾아서 설치

. https://minchanyoun.tistory.com/44

. <https://velog.io/@jiwon30931/AndroidkotlinRetrofit2>

. <https://medium.com/@aleslam12345/use-retrofit-with-kotlin-81cb938dfd10>

## Retrofit My Web Api with rxjava3

<https://velog.io/@seokzoo/Retrofit-Rxjava-%EC%82%AC%EC%9A%A9%ED%95%98%EA%B8%B0>

<https://adinugroho.medium.com/chaining-multiple-retrofit-call-using-rxjava-177b64c8103e>

## Scan모듈 추가

. Background Service 추가

## ViewModel

. https://readystory.tistory.com/176

## Simple Dagger and

. Dagger 모듈을 만들어서 @Inject에 자동으로 주입할 대상을 정의한다.

🡺 Dagger Component Interface를 작성 : @Inject로 표시한 생성자나, 멤버 변수에

Dagger모듈에서 정의된 값을 가져와서 주입한다.

🡺 Rebuild 후 Dagger Component를 사용하면 됨

. https://github.com/google/dagger

. https://reintech.io/blog/kotlin-dagger-2-dependency-injection-tutorial

. <https://tourspace.tistory.com/326>

## 권한 물어 보기 정리r

## Refrofit with Dagger

스캔, Api 호출, 비프음, dagger, MVVM

## PM90

1. PM90 라이브러리 적용:

. app 폴더 밑에 적용 libs폴더를 만들고 device.sdk.jar 파일복사

. android view에서 project view로 바꾸고 libs\device.sdk.jar 파일 우클릭

* Add as Library… >> implementation(files("libs\\device.sdk.jar"))

. https://blog.voidmainvoid.net/182

1. Receiver클래스 만들기
2. 백그라운드 서비스로 BroadcastReceiver
3. 서비스에서 스캔한 바코드 sendBroadcast 하고 각 화면에서 BroadcastReceiver 구현

## Retrofit Dagger MVVM

1. dagger Module class 생성

* fun providesRetrofit() is to retrun Retrofit object whereever it needed and you can see the @Provides annotation. Dagger2가 필요로 할 때 Retrofit Object를 제공한다.
* @Singleton will restrict the object creation to one time for the app lifecycle.

1. dagger Component interface 생성
2. Application class 생성

* manifest 파일에 application name속성에 추가
* 애플리케이션 프로세스가 시작되면 애플리케이션의 구성요소보다 먼저 이 클래스가 인스턴스화됩니다.

1. 참조

.https://medium.com/@otherTallguy/dagger2-with-mvvm-retrofit-roomdb-in-android-kotlin-3e9eab8875de

## Dagger 기초1

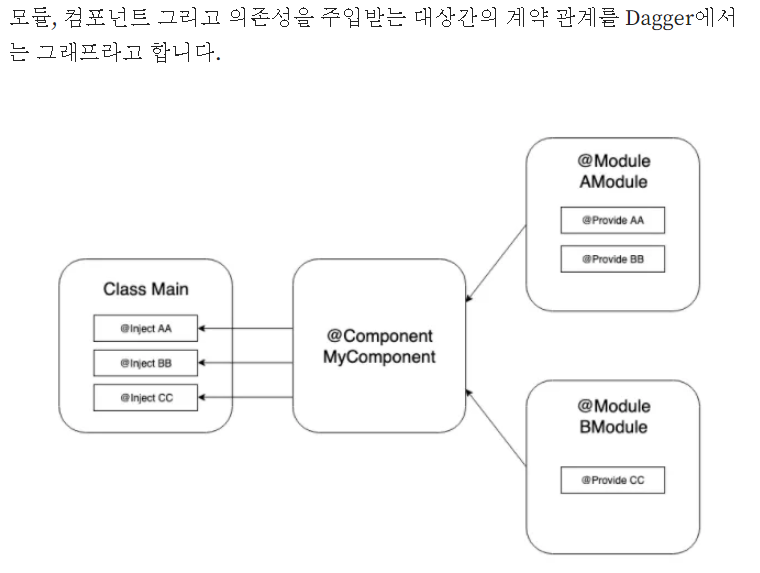
1. https://medium.com/@ramkid91/dagger2-%EA%B8%B0%EB%B3%B8%EA%B0%9C%EB%85%90-910d3ddfa104
2. @Inject annotation : @Component로부터 의존성 객체 주입을 요청
3. @Module : 클래스의 인스턴스를 모아 놓은 곳. Component에 의존성을 제공

* Module: 특정 기능을 위한 코드의 묶음 ( Network Module, DB Module..)
* @Module Class에만 붙임
* @Provides는 반드시 @Module 클래스에 선언된 메서드에만 사용합니다

1. @Component : @Component는 Interface 혹은 abstract class에서만 사용이 가능

* 컴파일 타임에 애노테이션 프로세서에 의해 생성된 클래스는 접두어 Dagger 와 @Component가 붙은 클래스이름이 합쳐진 형식의 이름을 갖습니다
* 연결된 Module을 이용하여 의존성 객체를 생성하고, Inject로 요청받은 인스턴스에 생성한 객체를 전달(주입) 합니다
* 의존성을 요청받고 전달(주입)하는 Dagger의 주된 역할을 수행하는 부분입니다.

1. Dagger Graph



### Login and Main Activity Flow

-. <https://medium.com/android-news/login-and-main-activity-flow-a52b930f8351>

-. <https://stackoverflow.com/questions/68908829/decode-token-jwt-android-kotlin>

Session관리

-. <https://www.geeksforgeeks.org/android-login-and-logout-using-shared-preferences-in-kotlin/>

### Room Db 사용하기

<https://velog.io/@soyoung-dev/AndroidKotlin-ROOM-Database-%EC%82%AC%EC%9A%A9%ED%95%98%EA%B8%B0>

* 데이터베이스 확인: View >> Tool Windows >> App Inspection

### Retrofit Response Sealed Class

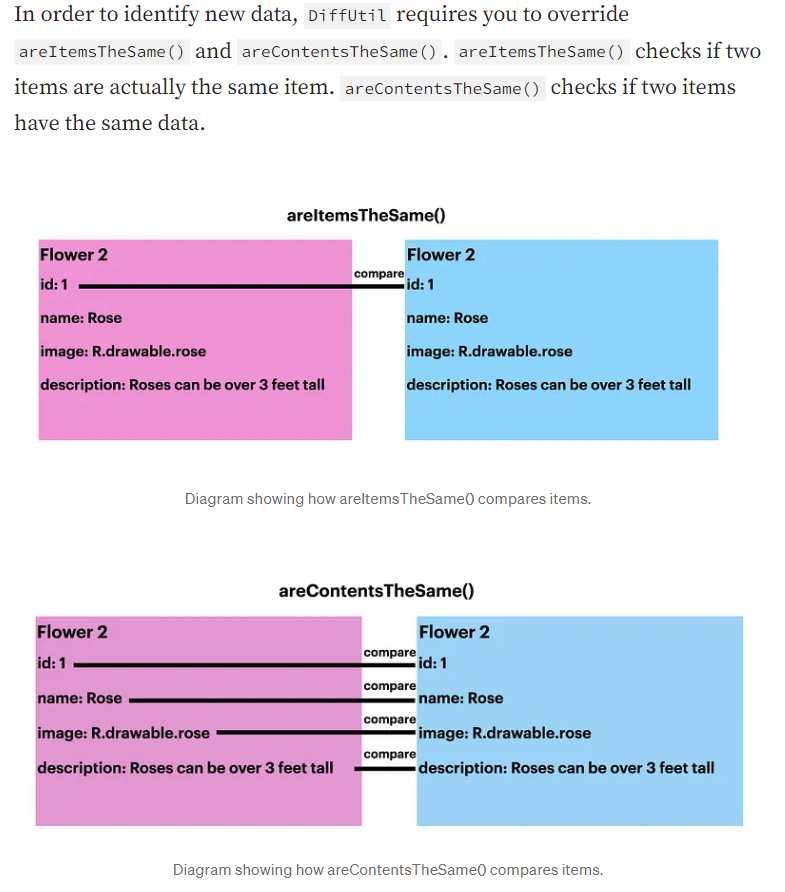
<https://medium.com/@hyzam.ali/modern-network-calls-in-android-with-retrofit-coroutines-and-sealed-classes-8c9453a6b331>

### RecyclerView에 ListAdapter 적용

1. 개요

* Recyclerview의 데이터가 변하면 Recyclerview Adapter가 제공하는 notifyItem 메소드를 사용해서 ViewHolder 내용을 갱신할 수 있습니다.
* Notify를 일일이 해 주는 것이 번거롭기도 하다.
* DiffUtil은 두 데이터셋을 받아서 그 차이를 계산해주는 클래스입니다. DiffUtil을 사용하면 두 데이터 셋을 비교한 뒤 그중 변한부분만을 파악하여 Recyclerview에 반영할 수 있습니다
* Recyclerview 어댑터를 ListAdapter로 구현하면 데이터가 어떻게 바뀌든간에 submitList로 전체 리스트를 넘겨주기만 하면 어댑터가 알아서 백그라운드 스레드를 사용해 리스트 차이를 계산하여 화면을 갱신시켜주게 됩니다.

1. 앱 작성
   1. Recyclerview에 표시할 데이터를 data class로 작성
   2. data class에 해당되는 레이아웃 작성
   3. Adapter class 작성
   4. Adaper class내 inner viewholder class작성
   5. Adaper class내 DiffUtil.ItemCallback 작성
   6. <https://cliearl.github.io/posts/android/recyclerview-listadapter/>



### 참조정리

<https://velog.io/@dldmswo1209/MVVM-%ED%8C%A8%ED%84%B4-%EA%B3%B5%EB%B6%80-Retrofit-ViewModel>

loggin sample

<https://harshitabambure.medium.com/login-api-with-retrofit-and-mvvm-with-auto-login-in-android-kotlin-bb6907092e0c>

Menu

<https://www.youtube.com/watch?v=Ttcwc3-q1ns>

MVVM

Add Library Dependency >>

. lifecycle-viewmodel-ktx 입력: androidx.lifecycle lifecycle-viewmodel-ktx Google ( 2.8.3 )

. lifecycle-livedata-ktx 입력 : androidx.lifecycle lifecycle-livedata-ktx Google ( 2.8.3 )

. retrofit 입력: com.squareup.retrofit2 retrofit Maven Central ( 2.11.0 )

// . gson 입력: com.google.code.gson gson Maven Central ( 2.11.0 )

. converter-gson 입력 : com.squareup.retrofit2 converter-gson Maven Central ( 2.11.0 )

. okhttp 입력: com.squareup.okhttp3 okhttp Maven Central ( 5.0.0-alpha.14 )

. logging-interceptor 입력: com.squareup.okhttp3 logging-interceptor Maven Central (상동)

. dagger 입력: com.google.dagger dagger Maven Central ( 2.51.1 )

. dagger-compiler 입력: com.google.dagger dagger-compiler Maven Central ( 2.51.1 )

* implementation >> kapt >> plugin에 id("kotlin-kapt") 추가

. glide 입력: com.github.bumptech.glide glide Maven Central ( 5.0.0-rc01 )

. jwtdecode 입력: com.auth0.android jwtdecode Maven Central ( 2.0.2 )

( <https://github.com/auth0/JWTDecode.Android> )

* PM90 라이브러리 적용:

. app 폴더 밑에 적용 libs폴더를 만들고 device.sdk.jar 파일복사

. android view에서 project view로 바꾸고 libs\device.sdk.jar 파일 우클릭

* Add as Library… >> implementation(files("libs\\device.sdk.jar"))

. https://blog.voidmainvoid.net/182

build.gradle.kts ( Module: app )

plugins **{** alias(libs.plugins.androidApplication)  
 alias(libs.plugins.jetbrainsKotlinAndroid)  
 id("kotlin-kapt")  
**}**

buildFeatures**{** viewBinding = true  
**}**

mane

### point mobile

<http://idnext.co.kr/bbs/board.php?bo_table=4010&wr_id=366&sst=wr_hit&sod=asc&sop=and&page=2>

. How to retain the errorBody() in retrofit response?

<https://stackoverflow.com/questions/49557988/android-how-to-retain-the-errorbody-in-retrofit-response>

* errorBody is of type ResponseBody is from OkHttp. The docs make it clear you can only read from it once. This is because it is a stream, not already stored in memory

### Web Api parameter 관련

public enum CommandType

{

//

// Summary:

// An SQL text command. (Default.)

Text = 1,

//

// Summary:

// The name of a stored procedure.

StoredProcedure = 4,

//

// Summary:

// The name of a table.

TableDirect = 512

}

public enum SqlDbType

{

//

// Summary:

// System.Int64. A 64-bit signed integer.

BigInt = 0,

//

// Summary:

// System.Array of type System.Byte. A fixed-length stream of binary data ranging

// between 1 and 8,000 bytes.

Binary = 1,

//

// Summary:

// System.Boolean. An unsigned numeric value that can be 0, 1, or null.

Bit = 2,

//

// Summary:

// System.String. A fixed-length stream of non-Unicode characters ranging between

// 1 and 8,000 characters.

Char = 3,

//

// Summary:

// System.DateTime. Date and time data ranging in value from January 1, 1753 to

// December 31, 9999 to an accuracy of 3.33 milliseconds.

DateTime = 4,

//

// Summary:

// System.Decimal. A fixed precision and scale numeric value between -10 38 -1 and

// 10 38 -1.

Decimal = 5,

//

// Summary:

// System.Double. A floating point number within the range of -1.79E +308 through

// 1.79E +308.

Float = 6,

//

// Summary:

// System.Array of type System.Byte. A variable-length stream of binary data ranging

// from 0 to 2 31 -1 (or 2,147,483,647) bytes.

Image = 7,

//

// Summary:

// System.Int32. A 32-bit signed integer.

Int = 8,

//

// Summary:

// System.Decimal. A currency value ranging from -2 63 (or -9,223,372,036,854,775,808)

// to 2 63 -1 (or +9,223,372,036,854,775,807) with an accuracy to a ten-thousandth

// of a currency unit.

Money = 9,

//

// Summary:

// System.String. A fixed-length stream of Unicode characters ranging between 1

// and 4,000 characters.

NChar = 10,

//

// Summary:

// System.String. A variable-length stream of Unicode data with a maximum length

// of 2 30 - 1 (or 1,073,741,823) characters.

NText = 11,

//

// Summary:

// System.String. A variable-length stream of Unicode characters ranging between

// 1 and 4,000 characters. Implicit conversion fails if the string is greater than

// 4,000 characters. Explicitly set the object when working with strings longer

// than 4,000 characters. Use System.Data.SqlDbType.NVarChar when the database column

// is nvarchar(max).

NVarChar = 12,

//

// Summary:

// System.Single. A floating point number within the range of -3.40E +38 through

// 3.40E +38.

Real = 13,

//

// Summary:

// System.Guid. A globally unique identifier (or GUID).

UniqueIdentifier = 14,

//

// Summary:

// System.DateTime. Date and time data ranging in value from January 1, 1900 to

// June 6, 2079 to an accuracy of one minute.

SmallDateTime = 15,

//

// Summary:

// System.Int16. A 16-bit signed integer.

SmallInt = 16,

//

// Summary:

// System.Decimal. A currency value ranging from -214,748.3648 to +214,748.3647

// with an accuracy to a ten-thousandth of a currency unit.

SmallMoney = 17,

//

// Summary:

// System.String. A variable-length stream of non-Unicode data with a maximum length

// of 2 31 -1 (or 2,147,483,647) characters.

Text = 18,

//

// Summary:

// System.Array of type System.Byte. Automatically generated binary numbers, which

// are guaranteed to be unique within a database. timestamp is used typically as

// a mechanism for version-stamping table rows. The storage size is 8 bytes.

Timestamp = 19,

//

// Summary:

// System.Byte. An 8-bit unsigned integer.

TinyInt = 20,

//

// Summary:

// System.Array of type System.Byte. A variable-length stream of binary data ranging

// between 1 and 8,000 bytes. Implicit conversion fails if the byte array is greater

// than 8,000 bytes. Explicitly set the object when working with byte arrays larger

// than 8,000 bytes.

VarBinary = 21,

//

// Summary:

// System.String. A variable-length stream of non-Unicode characters ranging between

// 1 and 8,000 characters. Use System.Data.SqlDbType.VarChar when the database column

// is varchar(max).

VarChar = 22,

//

// Summary:

// System.Object. A special data type that can contain numeric, string, binary,

// or date data as well as the SQL Server values Empty and Null, which is assumed

// if no other type is declared.

Variant = 23,

//

// Summary:

// An XML value. Obtain the XML as a string using the System.Data.SqlClient.SqlDataReader.GetValue(System.Int32)

// method or System.Data.SqlTypes.SqlXml.Value property, or as an System.Xml.XmlReader

// by calling the System.Data.SqlTypes.SqlXml.CreateReader method.

Xml = 25,

//

// Summary:

// A SQL Server user-defined type (UDT).

Udt = 29,

//

// Summary:

// A special data type for specifying structured data contained in table-valued

// parameters.

Structured = 30,

//

// Summary:

// Date data ranging in value from January 1,1 AD through December 31, 9999 AD.

Date = 31,

//

// Summary:

// Time data based on a 24-hour clock. Time value range is 00:00:00 through 23:59:59.9999999

// with an accuracy of 100 nanoseconds. Corresponds to a SQL Server time value.

Time = 32,

//

// Summary:

// Date and time data. Date value range is from January 1,1 AD through December

// 31, 9999 AD. Time value range is 00:00:00 through 23:59:59.9999999 with an accuracy

// of 100 nanoseconds.

DateTime2 = 33,

//

// Summary:

// Date and time data with time zone awareness. Date value range is from January

// 1,1 AD through December 31, 9999 AD. Time value range is 00:00:00 through 23:59:59.9999999

// with an accuracy of 100 nanoseconds. Time zone value range is -14:00 through

// +14:00.

DateTimeOffset = 34

}

public enum OracleDbType

{

BFile = 101,

Blob = 102,

Byte = 103,

Char = 104,

Clob = 105,

Date = 106,

Decimal = 107,

Double = 108,

Long = 109,

LongRaw = 110,

Int16 = 111,

Int32 = 112,

Int64 = 113,

IntervalDS = 114,

IntervalYM = 115,

NClob = 116,

NChar = 117,

NVarchar2 = 119,

Raw = 120,

RefCursor = 121,

Single = 122,

TimeStamp = 123,

TimeStampLTZ = 124,

TimeStampTZ = 125,

Varchar2 = 126,

XmlType = 127,

Array = 128,

Object = 129,

Ref = 130,

BinaryDouble = 132,

BinaryFloat = 133,

Boolean = 134,

Json = 135,

ArrayAsJson = 136,

ObjectAsJson = 137

}

public enum MySqlDbType

{

//

// Summary:

// MySql.Data.MySqlClient.MySqlDbType.Decimal

//

// A fixed precision and scale numeric value between -1038 -1 and 10 38 -1.

Decimal = 0,

//

// Summary:

// MySql.Data.MySqlClient.MySqlDbType.Byte

//

// The signed range is -128 to 127. The unsigned range is 0 to 255.

Byte = 1,

//

// Summary:

// MySql.Data.MySqlClient.MySqlDbType.Int16

//

// A 16-bit signed integer. The signed range is -32768 to 32767. The unsigned range

// is 0 to 65535

Int16 = 2,

//

// Summary:

// Specifies a 24 (3 byte) signed or unsigned value.

Int24 = 9,

//

// Summary:

// MySql.Data.MySqlClient.MySqlDbType.Int32

//

// A 32-bit signed integer

Int32 = 3,

//

// Summary:

// MySql.Data.MySqlClient.MySqlDbType.Int64

//

// A 64-bit signed integer.

Int64 = 8,

//

// Summary:

// System.Single

//

// A small (single-precision) floating-point number. Allowable values are -3.402823466E+38

// to -1.175494351E-38, 0, and 1.175494351E-38 to 3.402823466E+38.

Float = 4,

//

// Summary:

// MySql.Data.MySqlClient.MySqlDbType.Double

//

// A normal-size (double-precision) floating-point number. Allowable values are

// -1.7976931348623157E+308 to -2.2250738585072014E-308, 0, and 2.2250738585072014E-308

// to 1.7976931348623157E+308.

Double = 5,

//

// Summary:

// A timestamp. The range is '1970-01-01 00:00:00' to sometime in the year 2037

Timestamp = 7,

//

// Summary:

// Date The supported range is '1000-01-01' to '9999-12-31'.

Date = 10,

//

// Summary:

// Time

//

// The range is '-838:59:59' to '838:59:59'.

Time = 11,

//

// Summary:

// DateTime The supported range is '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.

DateTime = 12,

//

// Summary:

// Datetime The supported range is '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.

[Obsolete("The Datetime enum value is obsolete. Please use DateTime.")]

Datetime = 12,

//

// Summary:

// A year in 2- or 4-digit format (default is 4-digit). The allowable values are

// 1901 to 2155, 0000 in the 4-digit year format, and 1970-2069 if you use the 2-digit

// format (70-69).

Year = 13,

//

// Summary:

// Obsolete Use Datetime or Date type

Newdate = 14,

//

// Summary:

// A variable-length string containing 0 to 65535 characters

VarString = 15,

//

// Summary:

// Bit-field data type

Bit = 16,

//

// Summary:

// JSON

JSON = 245,

//

// Summary:

// New Decimal

NewDecimal = 246,

//

// Summary:

// An enumeration. A string object that can have only one value, chosen from the

// list of values 'value1', 'value2', ..., NULL or the special "" error value. An

// ENUM can have a maximum of 65535 distinct values

Enum = 247,

//

// Summary:

// A set. A string object that can have zero or more values, each of which must

// be chosen from the list of values 'value1', 'value2', ... A SET can have a maximum

// of 64 members.

Set = 248,

//

// Summary:

// A binary column with a maximum length of 255 (2^8 - 1) characters

TinyBlob = 249,

//

// Summary:

// A binary column with a maximum length of 16777215 (2^24 - 1) bytes.

MediumBlob = 250,

//

// Summary:

// A binary column with a maximum length of 4294967295 or 4G (2^32 - 1) bytes.

LongBlob = 251,

//

// Summary:

// A binary column with a maximum length of 65535 (2^16 - 1) bytes.

Blob = 252,

//

// Summary:

// A variable-length string containing 0 to 255 bytes.

VarChar = 253,

//

// Summary:

// A fixed-length string.

String = 254,

//

// Summary:

// Geometric (GIS) data type.

Geometry = 255,

//

// Summary:

// Unsigned 8-bit value.

UByte = 501,

//

// Summary:

// Unsigned 16-bit value.

UInt16 = 502,

//

// Summary:

// Unsigned 24-bit value.

UInt24 = 509,

//

// Summary:

// Unsigned 32-bit value.

UInt32 = 503,

//

// Summary:

// Unsigned 64-bit value.

UInt64 = 508,

//

// Summary:

// Fixed length binary string.

Binary = 754,

//

// Summary:

// Variable length binary string.

VarBinary = 753,

//

// Summary:

// A text column with a maximum length of 255 (2^8 - 1) characters.

TinyText = 749,

//

// Summary:

// A text column with a maximum length of 16777215 (2^24 - 1) characters.

MediumText = 750,

//

// Summary:

// A text column with a maximum length of 4294967295 or 4G (2^32 - 1) characters.

LongText = 751,

//

// Summary:

// A text column with a maximum length of 65535 (2^16 - 1) characters.

Text = 752,

//

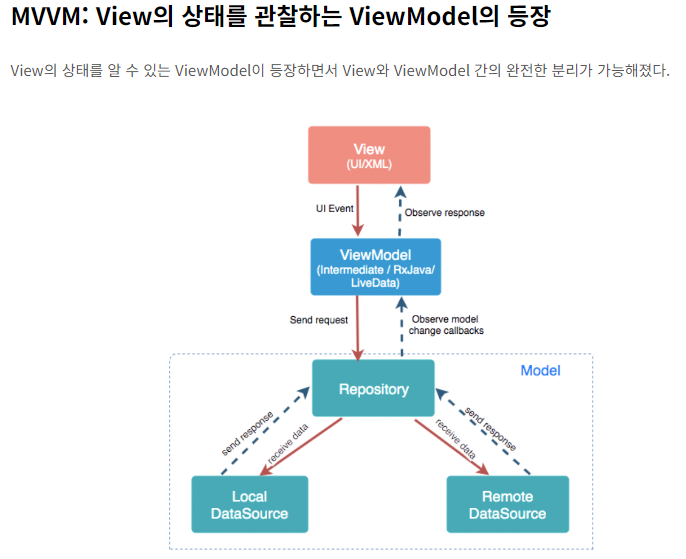
// Summary:

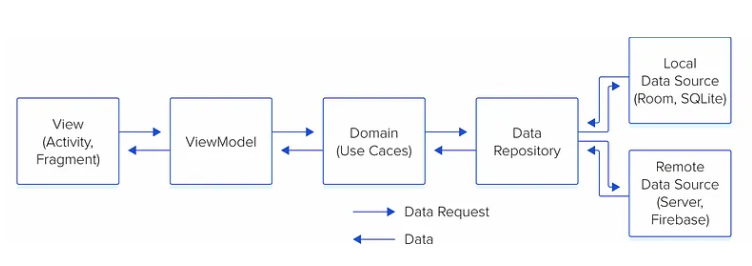
// A guid column.

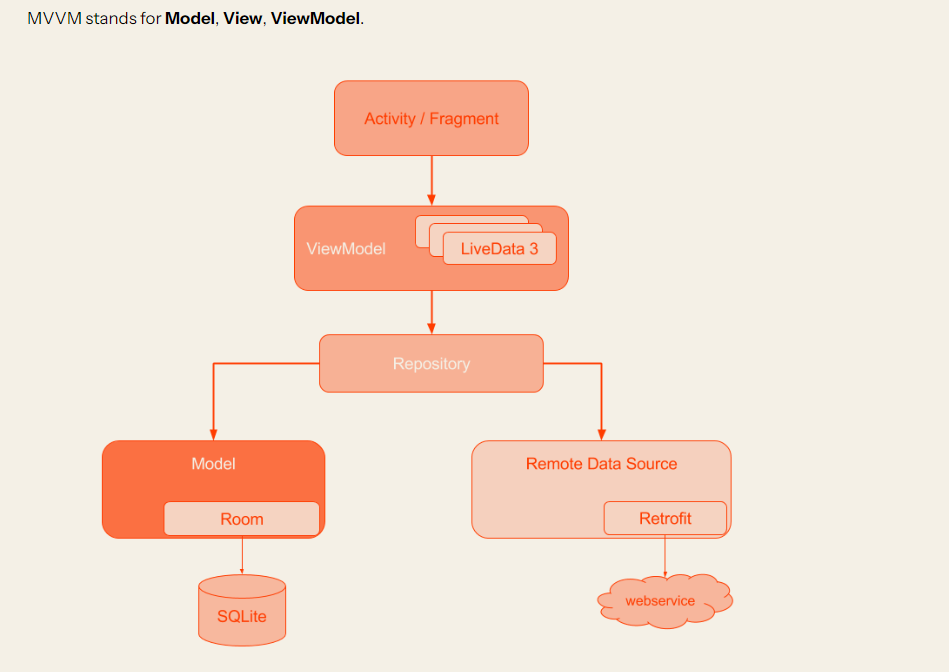
Guid = 854

}

### ㄴㅇㄻㄹㅇㄴㄹㅇㄴㄹㄴㅇ







### RecyclerView 무작정 따라하기

1. 데이터 클래스 정의

data class DogData(

val dog\_img : String,

val dog\_name : String,

val dog\_age: Int,

val dog\_gender:String

)

1. layout추가 : item\_list\_row.xml , DogData 표현
2. RecyclerViewAdapter 만들기 >> ViewHolder 만들기 >>
3. MainActivity에서 RecyclerView추가

. Recyclerview에 나타낼 데이터 만들기

. 코드에서 recyclerview의 adapter및 layoutManager추가

1. 참조

<https://developer.android.com/develop/ui/views/layout/recyclerview?hl=ko>

### RecyclerView 무작정 따라하기2 : retrofit2와 함께 처리

1. 데이터 클래스 정의

. data class GithubRepoData(val name: String?, val description: String?, val owner: Owner?)

. data class Owner(val avatar\_url: String?)

1. layout추가 : github\_repo\_row.xml , GithubRepoData 표현
2. RecyclerViewAdapter 만들기 >> ViewHolder 만들기 >> RecyclerView.Adapter<VH> 상속
3. sdfsd
4. glide는 사내 네트워크를 사용하면 이미지 로딩 시 오류가 나타남