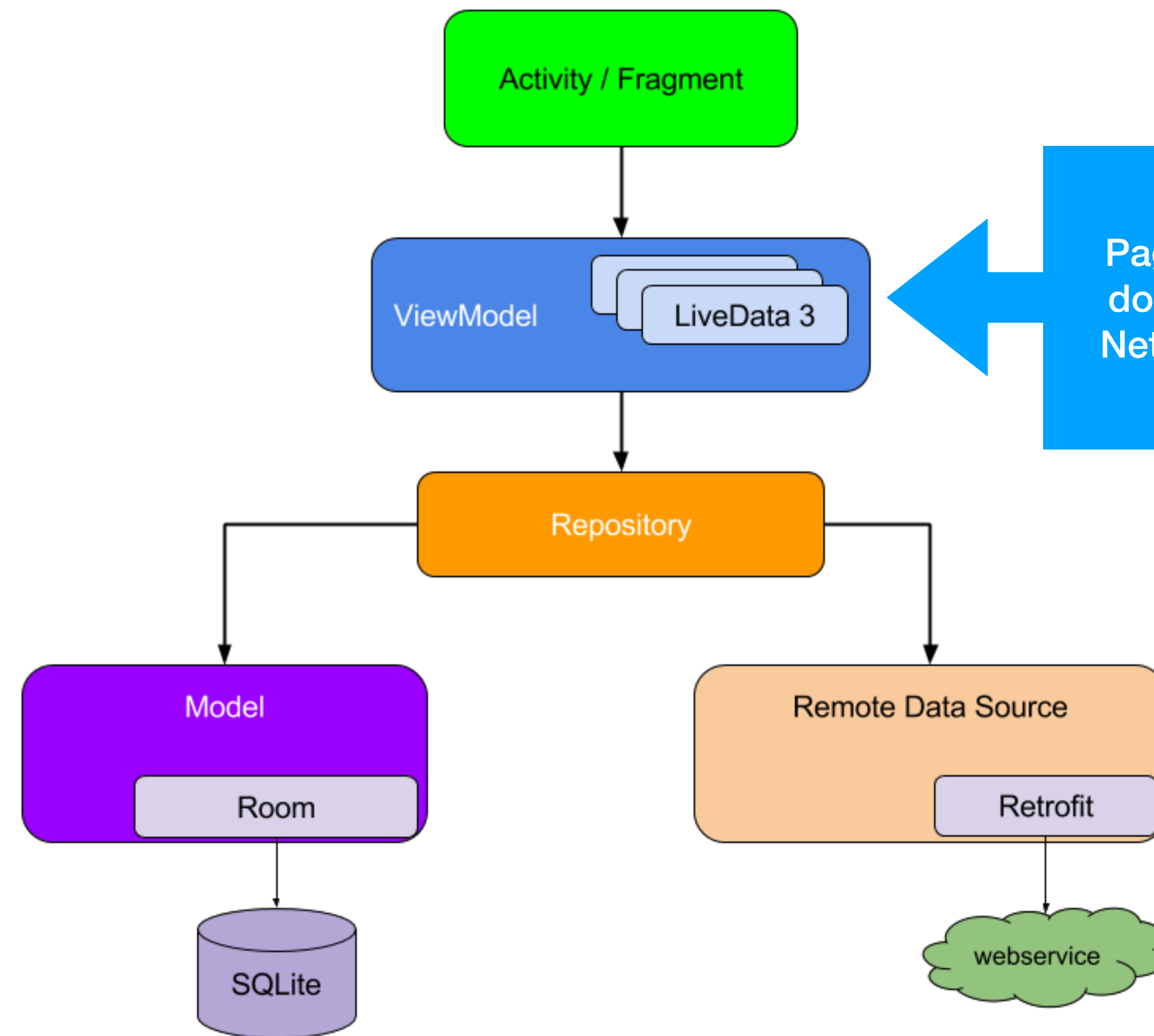


The background features the green Android mascot holding a white and blue rocket. The rocket is tilted upwards, and a trail of white and grey smoke is visible at its base. The scene is set against a light blue sky and a dark grey ground with several white and grey spheres.

Paging

Jetpack Architecture

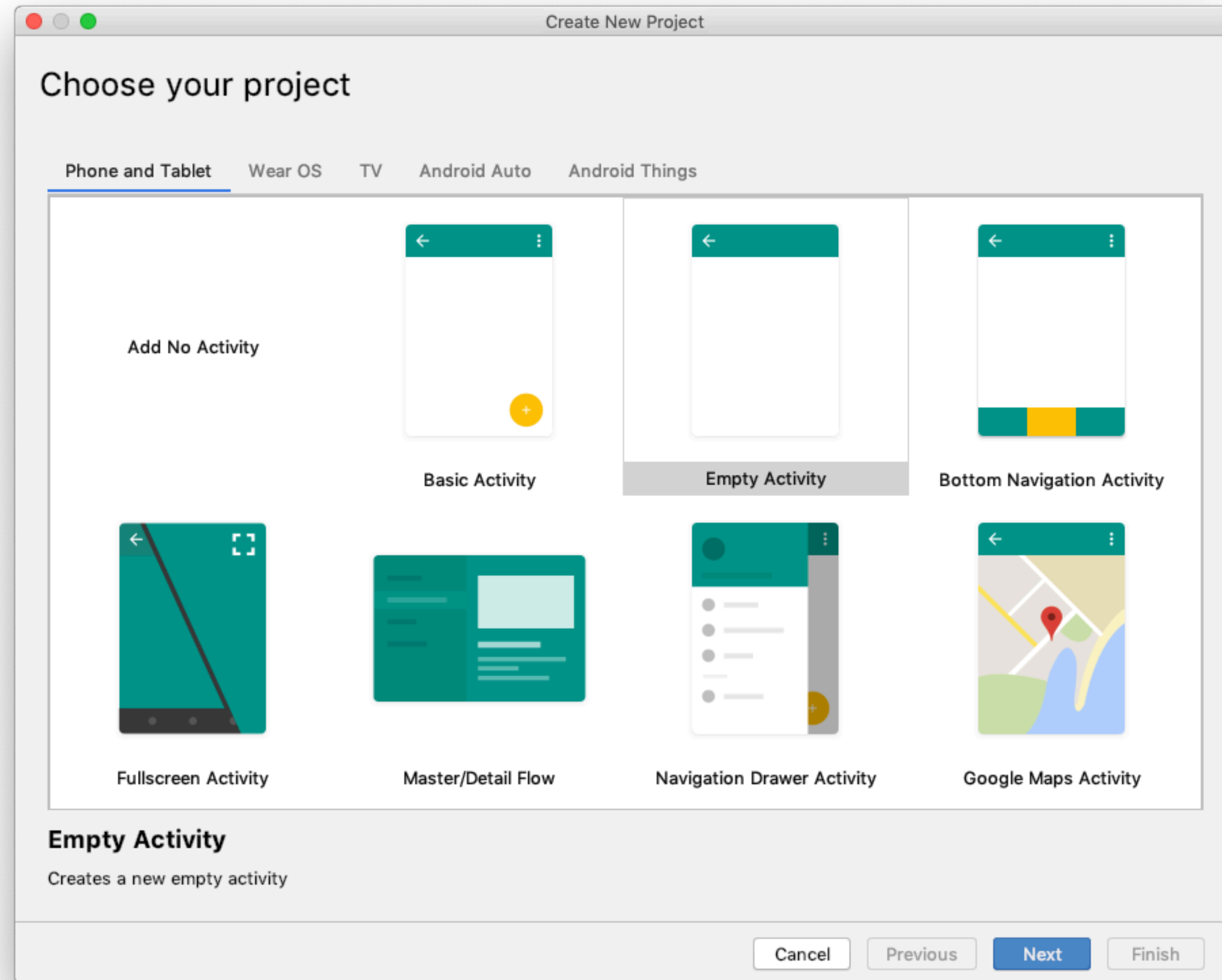
Architektur



Paging: Beinhaltet das seitenweise Anzeigen und downloaden von Daten. Dabei ist es wichtig, den Netzwerkverkehr so niedrig als möglich zu halten.

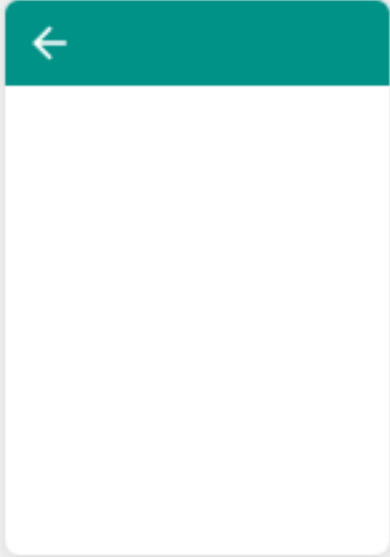
Features of Paging

- Makes it easy to load data gradually into the app's RecyclerView
- Load and display data efficiently in chunks
- Advantages:
 - Requests consume less network bandwidth and system resources
 - Quick response even during data update and refresh



Create New Project

Configure your project



Empty Activity

Creates a new empty activity

Name
Paging

Package name
at.htl.paging

Save location
roid.kotlin/2019.jetpack.udemy/04.Paging/android-jetpack-paging

Language
Kotlin

Minimum API level
API 26: Android 8.0 (Oreo)

i Your app will run on approximately **6.0%** of devices.
[Help me choose](#)

This project will support instant apps

Use AndroidX artifacts

⚠ 'android-jetpack-paging' already exists at the specified project location.

Cancel Previous Next **Finish**

build.gradle (app)

```
apply plugin: 'com.android.application'

apply plugin: 'kotlin-android'

apply plugin: 'kotlin-android-extensions'

apply plugin: 'kotlin-kapt'

android {
    compileSdkVersion 28
    defaultConfig {
        applicationId "at.htl.paging"
        minSdkVersion 26
        targetSdkVersion 28
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
}
```

```
dependencies {

    def paging_version = "2.1.0"
    def lifecycle_version = "2.0.0"

    // Retrofit
    implementation 'com.squareup.retrofit2:retrofit:2.5.0'
    implementation 'com.squareup.retrofit2:converter-gson:2.5.0'

    // Logging (ACHTUNG: Version muss zur okhttp3-Library von Retrofit passen)
    implementation 'com.squareup.okhttp3:logging-interceptor:3.12.0'

    // Paging
    implementation "androidx.paging:paging-runtime-ktx:$paging_version"

    // ViewModel and LiveData
    implementation "androidx.lifecycle:lifecycle-extensions:$lifecycle_version"
    // AndroidX libraries use this lightweight import for Lifecycle
    implementation "androidx.lifecycle:lifecycle-runtime:$lifecycle_version"
    // For Kotlin use kapt instead of annotationProcessor
    kapt "androidx.lifecycle:lifecycle-compiler:$lifecycle_version"

    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
    implementation 'androidx.appcompat:appcompat:1.0.2'
    implementation 'androidx.core:core-ktx:1.0.1'
    implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
    implementation 'com.google.android.material:material:1.0.0'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test.ext:junit:1.1.0'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'
}
```

<https://developer.android.com/topic/libraries/architecture/adding-components>

<https://material.io/develop/android/docs/getting-started/>

<https://square.github.io/retrofit/#download>

<https://mvnrepository.com/search?q=com.squareup.okhttp3%3Alogging-interceptor>

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="at.htl.paging">

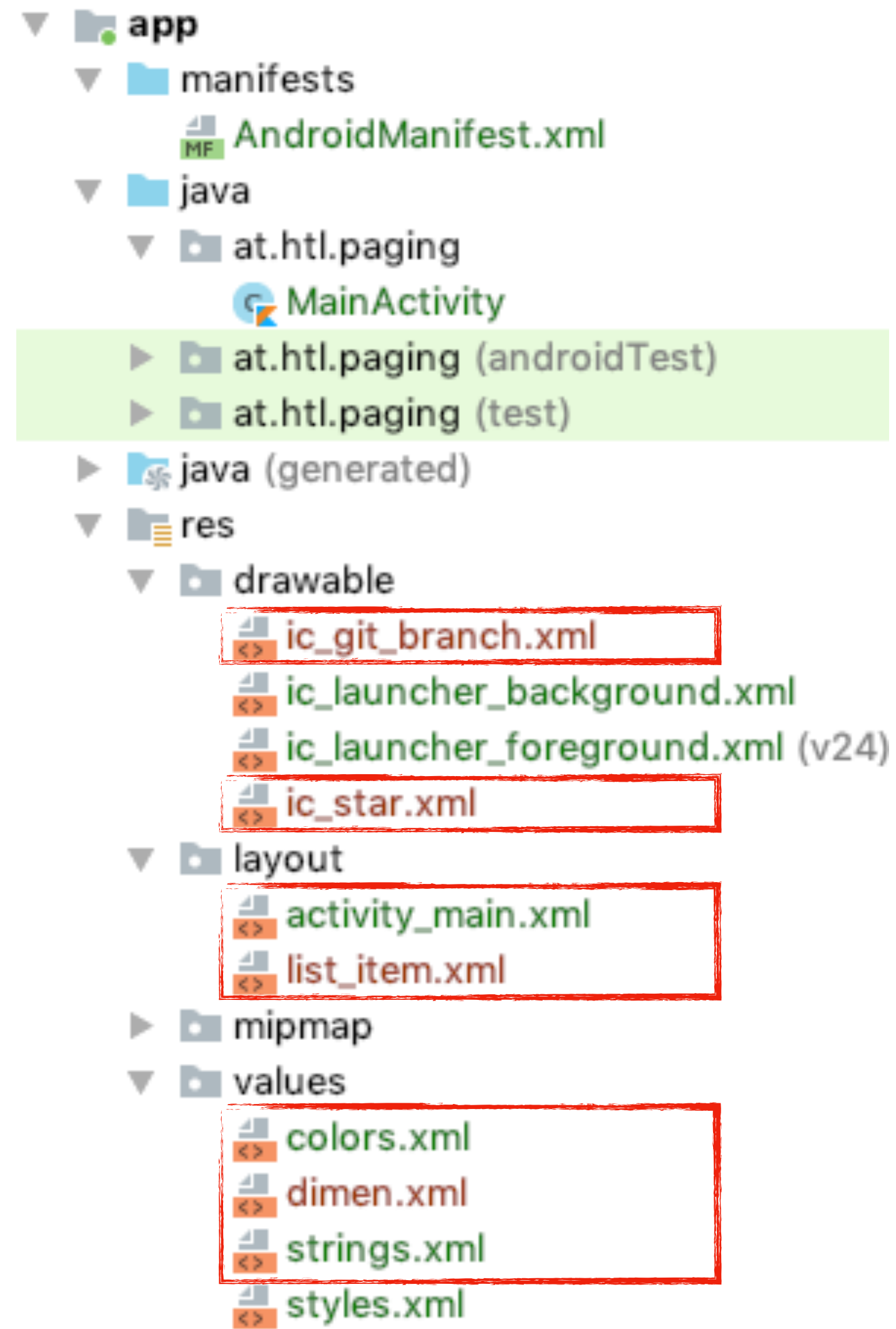
    <uses-permission android:name="android.permission.INTERNET"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Paging"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>

                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Files



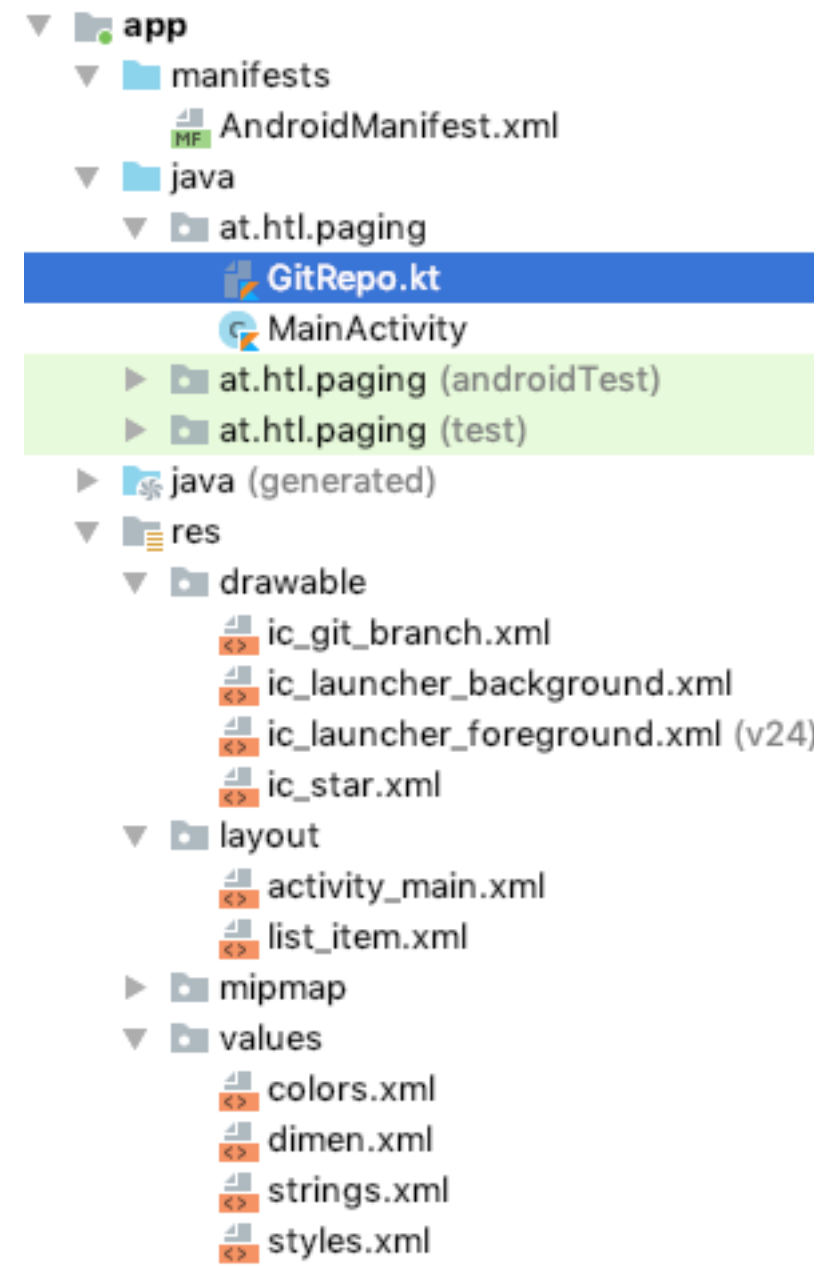
Url

```
JSON Raw Data Headers
Save Copy Collapse All Expand All
total_count: 821924
incomplete_results: false
items:
  0:
    id: 12544093
    node_id: "MDEwOJlJcG9zaXRvcnkxMjU0NDASw=="
    name: "Android"
    full_name: "hmkcode/Android"
    private: false
    owner:
      login: "hmkcode"
      id: 3790597
      node_id: "MDQ6VXNlcjM3OTA10Tc="
      avatar_url: "https://avatars3.githubusercontent.com/u/3790597?v=4"
      gravatar_id: ""
      url: "https://api.github.com/users/hmkcode"
      html_url: "https://github.com/hmkcode"
      followers_url: "https://api.github.com/users/hmkcode/followers"
      following_url: "https://api.github.com/users/hmkcode/following{/other_user}"
      gists_url: "https://api.github.com/users/hmkcode/gists{/gist_id}"
      starred_url: "https://api.github.com/users/hmkcode/starred{/owner}/{repo}"
      subscriptions_url: "https://api.github.com/users/hmkcode/subscriptions"
      organizations_url: "https://api.github.com/users/hmkcode/orgs"
      repos_url: "https://api.github.com/users/hmkcode/repos"
      events_url: "https://api.github.com/users/hmkcode/events{/privacy}"
      received_events_url: "https://api.github.com/users/hmkcode/received_events"
      type: "User"
      site_admin: false
    html_url: "https://github.com/hmkcode/Android"
    description: "Android related examples"
    fork: false
    url: "https://api.github.com/repos/hmkcode/Android"
    forks_url: "https://api.github.com/repos/hmkcode/Android/forks"
    keys_url: "https://api.github.com/repos/hmkcode/Android/keys{/key_id}"
    collaborators_url: "https://api.github.com/repos/hmkcode/Android/collaborators{/collaborator}"
    teams_url: "https://api.github.com/repos/hmkcode/Android/teams"
    hooks_url: "https://api.github.com/repos/hmkcode/Android/hooks"
    issue_events_url: "https://api.github.com/repos/hmkcode/Android/issues/events{/number}"
    events_url: "https://api.github.com/repos/hmkcode/Android/events"
    assignees_url: "https://api.github.com/repos/hmkcode/Android/assignees{/user}"
    branches_url: "https://api.github.com/repos/hmkcode/Android/branches{/branch}"
    tags_url: "https://api.github.com/repos/hmkcode/Android/tags"
    blobs_url: "https://api.github.com/repos/hmkcode/Android/git/blobs{/sha}"
    git_tags_url: "https://api.github.com/repos/hmkcode/Android/git/tags{/sha}"
    git_refs_url: "https://api.github.com/repos/hmkcode/Android/git/refs{/sha}"
    trees_url: "https://api.github.com/repos/hmkcode/Android/git/trees{/sha}"
    statuses_url: "https://api.github.com/repos/hmkcode/Android/statuses/{sha}"
```

```
JSON Raw Data Headers
Save Copy Collapse All Expand All
contributors_url: "https://api.github.com/repos/hmkcode/Android/contributors"
subscribers_url: "https://api.github.com/repos/hmkcode/Android/subscribers"
subscription_url: "https://api.github.com/repos/hmkcode/Android/subscription"
commits_url: "https://api.github.com/repos/hmkcode/Android/commits{/sha}"
git_commits_url: "https://api.github.com/repos/hmkcode/Android/git/commits{/sha}"
comments_url: "https://api.github.com/repos/hmkcode/Android/comments{/number}"
issue_comment_url: "https://api.github.com/repos/hmkcode/Android/issues/comments{/number}"
contents_url: "https://api.github.com/repos/hmkcode/Android/contents/{+path}"
compare_url: "https://api.github.com/repos/hmkcode/Android/compare/{base}...{head}"
merges_url: "https://api.github.com/repos/hmkcode/Android/merges"
archive_url: "https://api.github.com/repos/hmkcode/Android/{archive_format}/{ref}"
downloads_url: "https://api.github.com/repos/hmkcode/Android/downloads"
issues_url: "https://api.github.com/repos/hmkcode/Android/issues{/number}"
pulls_url: "https://api.github.com/repos/hmkcode/Android/pulls{/number}"
milestones_url: "https://api.github.com/repos/hmkcode/Android/milestones{/number}"
notifications_url: "https://api.github.com/repos/hmkcode/Android/notifications?since=all,participating"
labels_url: "https://api.github.com/repos/hmkcode/Android/labels{/name}"
releases_url: "https://api.github.com/repos/hmkcode/Android/releases{/id}"
deployments_url: "https://api.github.com/repos/hmkcode/Android/deployments"
created_at: "2013-09-02T16:12:28Z"
updated_at: "2019-02-21T01:32:25Z"
pushed_at: "2019-01-21T13:45:59Z"
git_url: "git://github.com/hmkcode/Android.git"
ssh_url: "git@github.com:hmkcode/Android.git"
clone_url: "https://github.com/hmkcode/Android.git"
svn_url: "https://github.com/hmkcode/Android"
homepage: null
size: 1838
stargazers_count: 2745
watchers_count: 2745
language: "Java"
has_issues: true
has_projects: true
has_downloads: true
has_wiki: true
has_pages: false
forks_count: 3310
mirror_url: null
archived: false
open_issues_count: 39
license: null
forks: 3310
open_issues: 39
watchers: 2745
default_branch: "master"
score: 115.66879
1:
  id: 82128465
```

api.github.com/search/repositories?q=android&sortby=stars&page=1&per_page=3

GitRepo.kt



```
package at.htl.paging
```

```
import com.google.gson.annotations.SerializedName
```

```
class GitRepo {
```

```
    @field:SerializedName("full_name")
```

```
    var fullName: String? = null
```

```
    val description: String? = null
```

```
    @field:SerializedName("stargazers_count")
```

```
    val stars: Int = 0
```

```
    @field:SerializedName("forks_count")
```

```
    val forks: Int = 0
```

```
    @field:SerializedName("language")
```

```
    val language: String? = null
```

```
}
```

```
class GitRepoResponse {
```

```
    @field:SerializedName("total_count")
```

```
    var totalCount: Int = 0
```

```
    var items: List<GitRepo>? = null
```

```
}
```

SerializedName bedeutet, daß im Programm CamelCase verwendet wird (totalCount), im JSON-File jedoch der snake-case (total_count)

Webservice-Interface mit Retrofit

```
package at.htl.paging

import retrofit2.Call
import retrofit2.http.GET
import retrofit2.http.Query

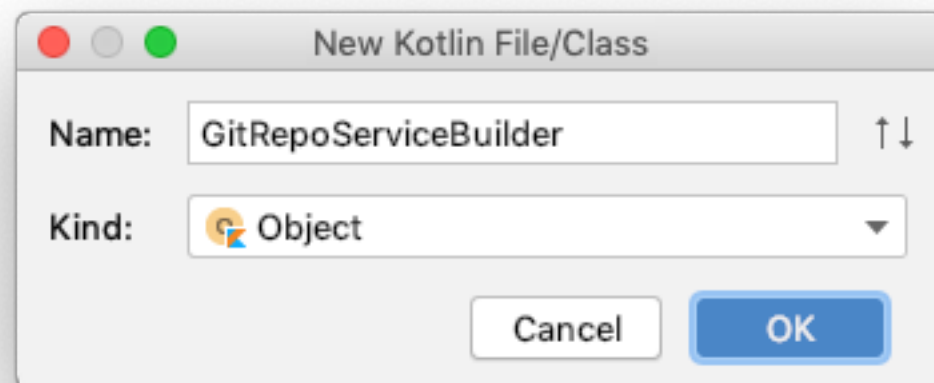
interface GitRepoService {

    @GET("search/repositories?sort=stars")
    fun getRepositories(
        @Query("page") page: Int,
        @Query("per_page") size: Int,
        @Query("q") topic: String
    ): Call<GitRepoResponse>
}
```

Zuerst wird ein Interface erstellt.
Anschließend muß die Function noch
implementiert werden. Dazu erstellen wir einen
ServiceBuilder.

<https://square.github.io/retrofit/>

GitRepoServiceBuilder.kt



```
package at.htl.paging

import okhttp3.OkHttpClient
import okhttp3.logging.HttpLoggingInterceptor
import retrofit2.Retrofit
import retrofit2.converter.gson.GsonConverterFactory

object GitRepoServiceBuilder {

    // Base URL
    private const val URL = "https://api.github.com/"

    // Create Logger
    private val logger = HttpLoggingInterceptor()
        .setLevel(HttpLoggingInterceptor.Level.BODY)

    // Create OkHttpClient
    private val okhttp = OkHttpClient
        .Builder()
        .addInterceptor(logger)

    // Create Retrofit Builder
    private val builder = Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .client(okhttp.build())

    // Create Retrofit Instance
    private val retrofit = builder.build()

    fun <T> buildService(serviceType: Class<T>): T {
        return retrofit.create(serviceType)
    }
}
```

Beachte: Am Ende der URL ist ein Slash

Logger

Client
Der Logger loggt den Request und die Response

Builder
Die Instanz von Retrofit wird zurückgegeben

Instance

MainActivity.kt 1

```
package at.htl.paging
```

```
import androidx.appcompat.app.AppCompatActivity  
import android.os.Bundle  
import retrofit2.Call  
import retrofit2.Callback  
import retrofit2.Response
```

```
class MainActivity : AppCompatActivity() {
```

```
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)
```

```
        val service = GitRepoServiceBuilder.buildService(GitRepoService::class.java)
```

```
        val call = service.getRepositories(1, 10, "android")
```

```
        call.enqueue(object: Callback<GitRepoResponse> {
```

```
            override fun onResponse(call: Call<GitRepoResponse>, response: Response<GitRepoResponse>) {  
            }  
        })
```

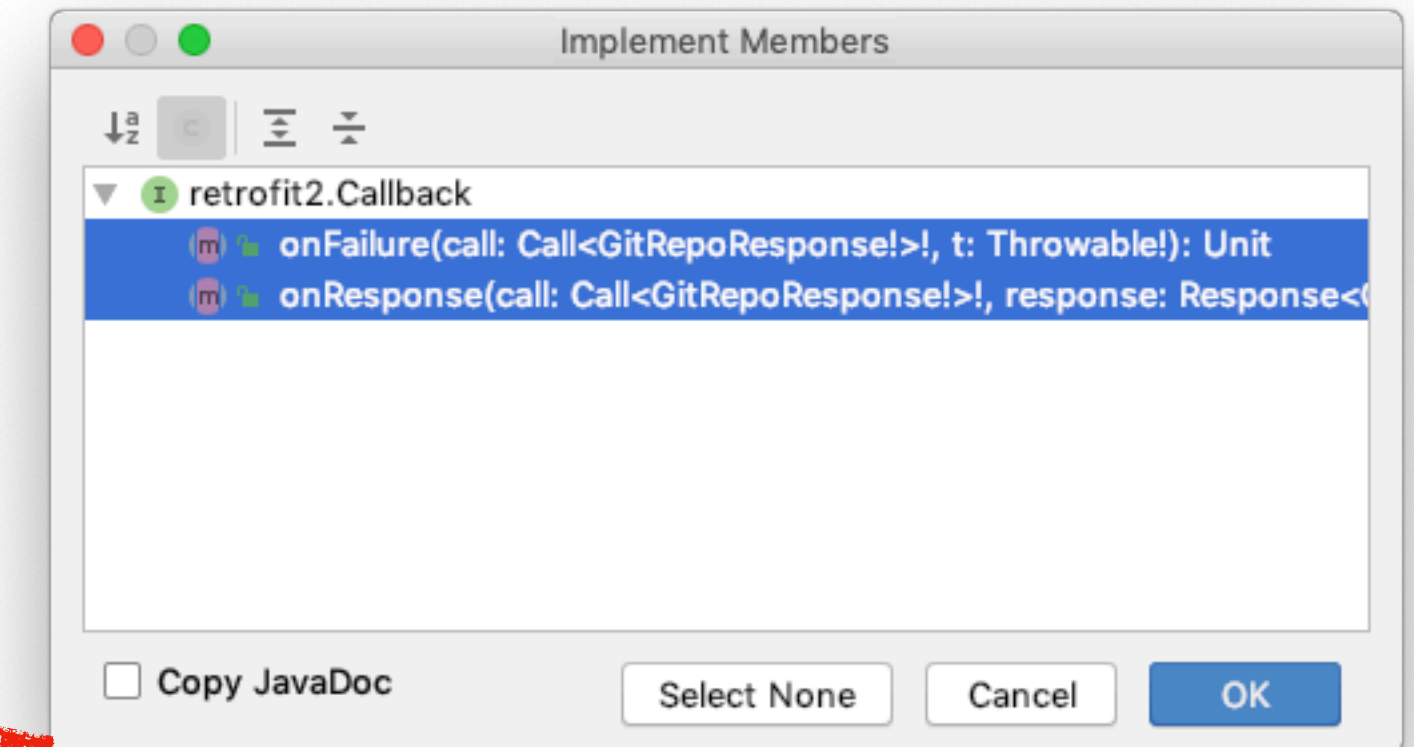
```
        override fun onFailure(call: Call<GitRepoResponse>, t: Throwable) {
```

```
        }
```

```
    })
```

```
}
```

```
}
```



MainActivity.kt 2

```
import ...
```

```
class MainActivity : AppCompatActivity() {
```

```
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)
```

```
        val service = GitRepoServiceBuilder.buildService(GitRepoService::class.java)
```

```
        val call = service.getRepositories(1, 10, "android")
```

```
        call.enqueue(object: Callback<GitRepoResponse> {
```

```
            override fun onResponse(call: Call<GitRepoResponse>, response: Response<GitRepoResponse>) {
```

```
                if (response.isSuccessful) {  
                    val apiResponse = response.body()!!  
                    val responseItems = apiResponse.items
```

```
                    val size = responseItems?.let {  
                        responseItems.size.toString()
```

```
                    }  
                    Toast.makeText(this@MainActivity, size, Toast.LENGTH_LONG).show()
```

```
                }  
            }
```

```
            override fun onFailure(call: Call<GitRepoResponse>, t: Throwable) {
```

```
            }
```

```
        })
```

```
    }
```

```
}
```

JSON	Raw Data	Headers
Save	Copy	Collapse All
Expand All		
total_count:	821924	
incomplete_results:	false	
items:		
0:		
id:	12544093	
node_id:	"MDEwO111cG9zaXBvcnkyMjU0NDQ5MjU="	

android-jetpack-paging [~/CloudStation/htl/skripten/themen/android.kotlin/2019.jetpack.udemy/04.Paging/android-jetpack-paging] - app

android-jetpack-paging > app > build.gradle

Android > app

Configure project in Project Structure dialog.

```

26 dependencies {
27
28
29     def paging_version = "2.1.0"
30     def lifecycle_version = "2.0.0"
31
32     // Retrofit
33     implementation 'com.squareup.retrofit2:retrofit:2.4.0'
34     implementation 'com.squareup.retrofit2:converter-gson:2.4.0'
35 }
dependencies{}

```

Logcat

Emulator Pixel_2_API_28 Android at.htl.paging (16785) Verbose Q okhttp

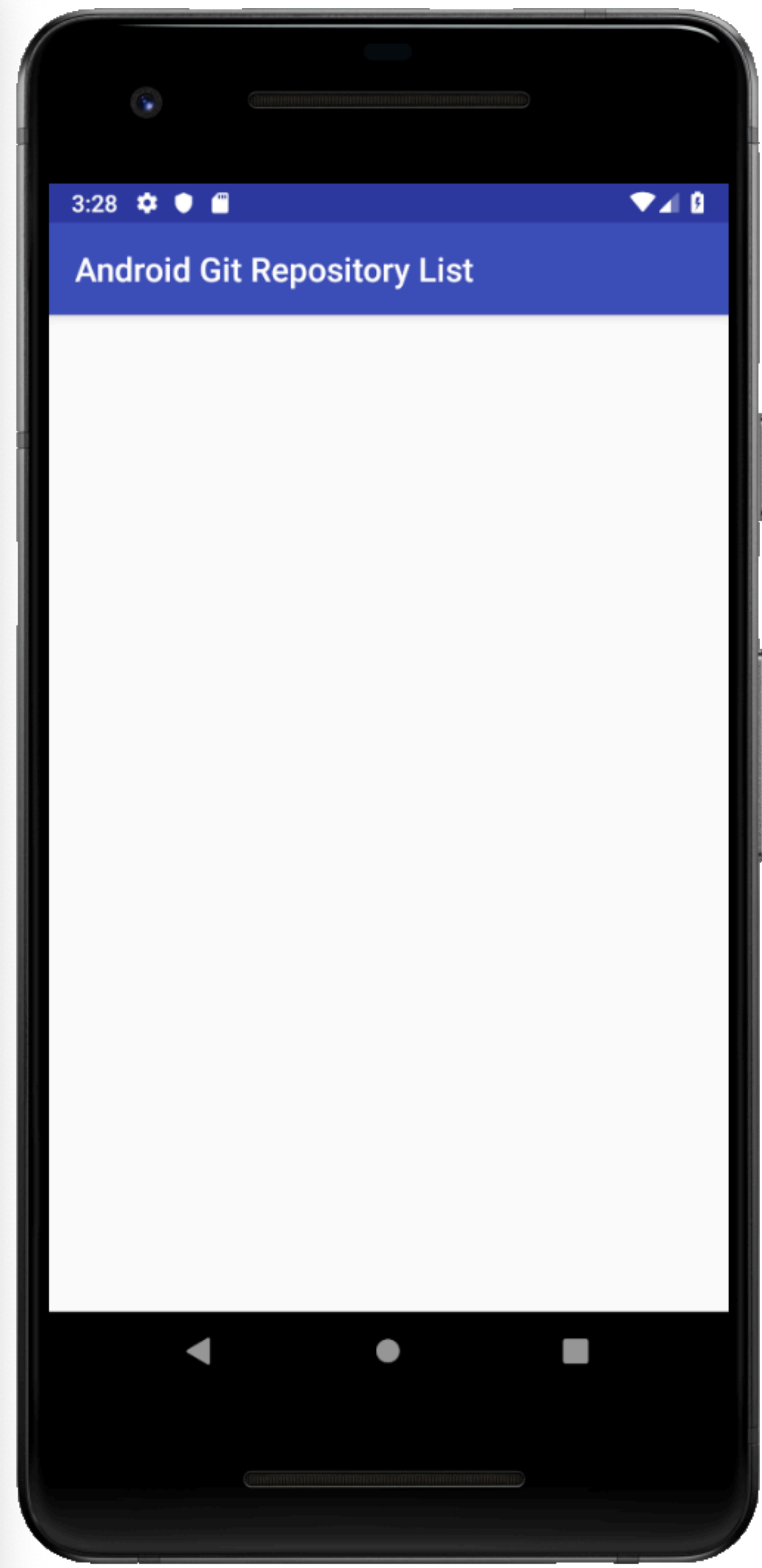
```

2019-02-22 15:25:17.640 16785-16815/at.htl.paging D/OkHttp: --> GET https://api.github.com/search/repositories?sort=stars&page=1&per_page=10&q=android
2019-02-22 15:25:17.640 16785-16815/at.htl.paging D/OkHttp: --> END GET
2019-02-22 15:25:19.789 16785-16815/at.htl.paging D/OkHttp: <-- 200 OK https://api.github.com/search/repositories?sort=stars&page=1&per_page=10&q=android (2147ms)
2019-02-22 15:25:19.790 16785-16815/at.htl.paging D/OkHttp: Server: GitHub.com
2019-02-22 15:25:19.792 16785-16815/at.htl.paging D/OkHttp: Date: Fri, 22 Feb 2019 14:25:19 GMT
2019-02-22 15:25:19.792 16785-16815/at.htl.paging D/OkHttp: Content-Type: application/json; charset=utf-8
2019-02-22 15:25:19.792 16785-16815/at.htl.paging D/OkHttp: Transfer-Encoding: chunked
2019-02-22 15:25:19.792 16785-16815/at.htl.paging D/OkHttp: Status: 200 OK
2019-02-22 15:25:19.792 16785-16815/at.htl.paging D/OkHttp: X-RateLimit-Limit: 10
2019-02-22 15:25:19.793 16785-16815/at.htl.paging D/OkHttp: X-RateLimit-Remaining: 9
2019-02-22 15:25:19.793 16785-16815/at.htl.paging D/OkHttp: X-RateLimit-Reset: 1550845579
2019-02-22 15:25:19.793 16785-16815/at.htl.paging D/OkHttp: Cache-Control: no-cache
2019-02-22 15:25:19.793 16785-16815/at.htl.paging D/OkHttp: X-GitHub-Media-Type: github.v3; format=json
2019-02-22 15:25:19.793 16785-16815/at.htl.paging D/OkHttp: Link: <https://api.github.com/search/repositories?sort=stars&page=2&per_page=10&q=android>; rel="next", <https://api.github.com/
2019-02-22 15:25:19.793 16785-16815/at.htl.paging D/OkHttp: Access-Control-Expose-Headers: ETag, Link, Location, Retry-After, X-GitHub-OTP, X-RateLimit-Limit, X-RateLimit-Remaining, X-Rate
2019-02-22 15:25:19.794 16785-16815/at.htl.paging D/OkHttp: Access-Control-Allow-Origin: *
2019-02-22 15:25:19.794 16785-16815/at.htl.paging D/OkHttp: Strict-Transport-Security: max-age=31536000; includeSubdomains; preload
2019-02-22 15:25:19.794 16785-16815/at.htl.paging D/OkHttp: X-Frame-Options: deny
2019-02-22 15:25:19.794 16785-16815/at.htl.paging D/OkHttp: X-Content-Type-Options: nosniff
2019-02-22 15:25:19.794 16785-16815/at.htl.paging D/OkHttp: X-XSS-Protection: 1; mode=block
2019-02-22 15:25:19.794 16785-16815/at.htl.paging D/OkHttp: Referrer-Policy: origin-when-cross-origin, strict-origin-when-cross-origin
2019-02-22 15:25:19.794 16785-16815/at.htl.paging D/OkHttp: Content-Security-Policy: default-src 'none'
2019-02-22 15:25:19.794 16785-16815/at.htl.paging D/OkHttp: X-GitHub-Request-Id: E84B:7A18:EC3D4D:2181498:5C70064E
2019-02-22 15:25:19.869 16785-16815/at.htl.paging D/OkHttp: {"total_count":822173,"incomplete_results":false,"items":[{"id":31792824,"node_id":"MDEwOJlJlcG9zaXRvcnkzMTC5MjgyNA==","name":"fl
2019-02-22 15:25:19.869 16785-16815/at.htl.paging D/OkHttp: eases{/id},"deployments_url":"https://api.github.com/repos/flutter/flutter/deployments","created_at":"2015-03-06T22:54:58Z","
2019-02-22 15:25:19.871 16785-16815/at.htl.paging D/OkHttp: ogramming-books-zh_CN/subscribers","subscription_url":"https://api.github.com/repos/justjavac/free-programming-books-zh_CN/subscri
2019-02-22 15:25:19.871 16785-16815/at.htl.paging D/OkHttp: s://api.github.com/repos/google/material-design-icons/forks","keys_url":"https://api.github.com/repos/google/material-design-ico
2019-02-22 15:25:19.874 16785-16815/at.htl.paging D/OkHttp: //api.github.com/licenses/apache-2.0","node_id":"MDc6TGJlZW5zZTI="},"forks":7511,"open_issues":538,"watchers":37204,"default_bra
2019-02-22 15:25:19.875 16785-16815/at.htl.paging D/OkHttp: ssues_url":"https://api.github.com/repos/wasabeef/awesome-android-ui/issues{/number},"pulls_url":"https://api.github.com/repos/
2019-02-22 15:25:19.876 16785-16815/at.htl.paging D/OkHttp: lesamples/android-architecture/git/refs{/sha},"trees_url":"https://api.github.com/repos/googlesamples/android-architecture/git/
2019-02-22 15:25:19.877 16785-16815/at.htl.paging D/OkHttp: pi.github.com/users/square/repos","events_url":"https://api.github.com/users/square/events{/privacy},"received_events_url":"http
2019-02-22 15:25:19.877 16785-16815/at.htl.paging D/OkHttp: ":31322,"default_branch":"master","score":40.95126},{id":5152285,"node_id":"MDEwOJlJlcG9zaXRvcnk1MTUyMjgy","name":"okhttp","full
2019-02-22 15:25:19.877 16785-16815/at.htl.paging D/OkHttp: :42:552","updated_at":"2019-02-22T12:25:55Z","pushed_at":"2019-02-21T22:18:37Z","git_url":"git://github.com/square/okhttp.git","
2019-02-22 15:25:19.878 16785-16815/at.htl.paging D/OkHttp: subscription_url":"https://api.github.com/repos/Hack-with-Github/Awesome-Hacking/subscription","commits_url":"https://api.github
2019-02-22 15:25:19.878 16785-16815/at.htl.paging D/OkHttp: _url":"https://api.github.com/repos/Trinea/android-open-project/collaborators{/collaborator},"teams_url":"https://api.github.com
2019-02-22 15:25:19.878 16785-16815/at.htl.paging D/OkHttp: :MPAndroidChart","full_name":"PhilJay/MPAndroidChart","private":false,"owner":{"login":"PhilJay","id":6759734,"node_id":"MDQ6VXN
2019-02-22 15:25:19.879 16785-16815/at.htl.paging D/OkHttp: thub.com/repos/PhilJay/MPAndroidChart/notifications?since=all,participating},"labels_url":"https://api.github.com/repos/PhilJay
2019-02-22 15:25:19.879 16785-16815/at.htl.paging D/OkHttp: <-- END HTTP (53218-byte body)

```

Install successful (2 minutes ago)

28:1 LF ⇅ UTF-8 ⇅ 4 spaces ⇅ Git: master ⇅



Troubleshooting

```
Logcat
Emulator Nexus_5X_API_27 And at.htl.paging (7004) [DEAD] Verbose
2019-02-22 00:36:30.842 7004-7004/? I/zygote: Not late-enabling -Xcheck:jni (already on)
2019-02-22 00:36:30.906 7004-7004/? W/zygote: Unexpected CPU variant for X86 using defaults: x86
2019-02-22 00:36:31.632 7004-7004/at.htl.paging I/zygote: Rejecting re-init on previously-failed class java.lang.Class<androidx.core
↳ java.lang.NoClassDefFoundError: Failed resolution of: Landroid/view/View$OnUnhandledKeyListener;
2019-02-22 00:36:31.633 7004-7004/at.htl.paging I/zygote: at void androidx.core.view.ViewCompat.setBackground(android.view.View,
.java:2341)
2019-02-22 00:36:31.633 7004-7004/at.htl.paging I/zygote: at void androidx.appcompat.widget.ActionBarContainer.<init>(android.co
(ActionBarContainer.java:62)
2019-02-22 00:36:31.633 7004-7004/at.htl.paging I/zygote: at java.lang.Object java.lang.reflect.Constructor.newInstance0(java.la
```

le...@gmail.com <le...@gmail.com> #14 Dec 1, 2018 10:42PM

Same problem here. AndroidX project running on an emulator with API level 26

<https://issuetracker.google.com/issues/110162198>

AVD mit mind. API 28 verwenden

Troubleshooting 2

```
Logcat
Emulator Pixel_2_API_28 Android No debuggable processes Verbose
2019-02-22 15:11:24.126 16220-16220/? E/AndroidRuntime: FATAL EXCEPTION: main
Process: at.htl.paging, PID: 16220
java.lang.BootstrapMethodError: Exception from call site #8 bootstrap method
at okhttp3.logging.HttpLoggingInterceptor.<clinit>(HttpLoggingInterceptor.java:11)
at okhttp3.logging.HttpLoggingInterceptor.<init>(HttpLoggingInterceptor.java:15)
at at.htl.paging.GitRepoServiceBuilder.<clinit>(GitRepoServiceBuilder.java:16)
at at.htl.paging.MainActivity.onCreate(MainActivity.kt:16)
at android.app.Activity.performCreate(Activity.java:7136)
at android.app.Activity.performCreate(Activity.java:7127)
at android.app.Instrumentation.callActivityOnCreate(Instrumentation.java:1271)
at android.app.ActivityThread.performLaunchActivity(ActivityThread.java:2893)
at android.app.ActivityThread.handleLaunchActivity(ActivityThread.java:3048)
at android.app.servertransaction.LaunchActivityItem.execute(LaunchActivityItem.java:78)
at android.app.servertransaction.TransactionExecutor.executeCallbacks(TransactionExecutor.java:108)
at android.app.servertransaction.TransactionExecutor.execute(TransactionExecutor.java:68)
at android.app.ActivityThread$H.handleMessage(ActivityThread.java:1808)
at android.os.Handler.dispatchMessage(Handler.java:106)
at android.os.Looper.loop(Looper.java:193)
at android.app.ActivityThread.main(ActivityThread.java:6669) <1 internal call>
at com.android.internal.os.RuntimeInit$MethodAndArgsCaller.run(RuntimeInit.java:493)
at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:858)
Caused by: java.lang.ClassCastException: Bootstrap method returned null <16 more...> <1 internal call> <2 more...>
2019-02-22 15:11:24.180 1926-4886/? W/ActivityManager: Force finishing activity at.htl.paging/.MainActivity
```

Ursache des Problems (und Lösung)

```
dependencies {
    def paging_version = "2.1.0"
    def lifecycle_version = "2.0.0"

    // Retrofit
    implementation 'com.squareup.retrofit2:retrofit:2.5.0'
    implementation 'com.squareup.retrofit2:converter-gson:2.5.0'

    // Logging
    implementation 'com.squareup.okhttp3:logging-interceptor:3.12.0'
```

In my case this pair of dependencies solved the problem:

```
compile 'com.squareup.retrofit2:retrofit:2.1.0'
compile 'com.squareup.okhttp3:logging-interceptor:3.3.1'
```

Dieser Tipp funktioniert. Wir wollen natürlich die neueste Library verwenden!

Anyway, the problem is in dependency consistency...

answered Jan 31 '17 at 11:43
Andrey
18.3k ● 7 ● 93 ● 78
<https://stackoverflow.com/a/41956711>

Gradle View - /Users/stuetz/CloudStation/htl/skripten/t...

- debugUnitTestRuntimeOnlyDependenciesMetadata
- debugWearBundling
- default
- implementationDependenciesMetadata
 - com.squareup.retrofit2:retrofit:2.5.0
 - com.squareup.okhttp3:okhttp:3.12.0
 - com.squareup.retrofit2:converter-gson:2.5.0
 - com.squareup.retrofit2:retrofit:2.5.0
 - com.google.code.gson:gson:2.8.2
 - com.squareup.okhttp3:logging-interceptor:3.9.0
 - androidx.paging:paging-runtime-ktx:2.1.0
 - androidx.lifecycle:lifecycle-extensions:2.0.0
 - androidx.lifecycle:lifecycle-runtime:2.0.0
 - org.jetbrains.kotlin:kotlin-stdlib-jdk7:1.3.21
 - androidx.appcompat:appcompat:1.1.0-alpha02
 - androidx.core:core-ktx:1.1.0-alpha04
 - androidx.constraintlayout:constraintlayout:2.0.0-alpha3

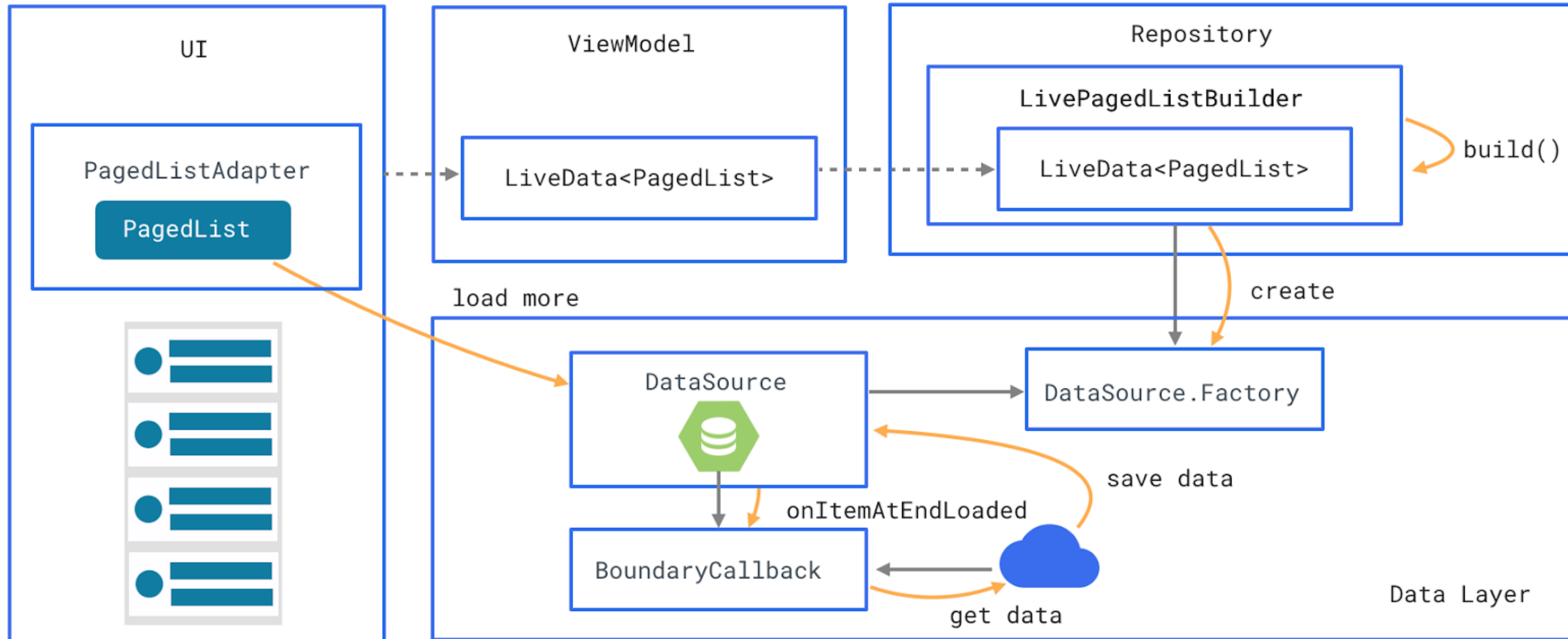
Retrofit basiert (bei allen Netzwerkzugriffen) auf okhttp3. Daher ist diese Library bereits indirekt im Projekt enthalten und muß nicht mehr als Dependency eingetragen werden.

Wird nun eine zusätzliche okhttp3-Bibliothek hinzugefügt (für das Logging), so muß diese die selbe Versionsnr, wie die der okhttp3-Core-Library sein.

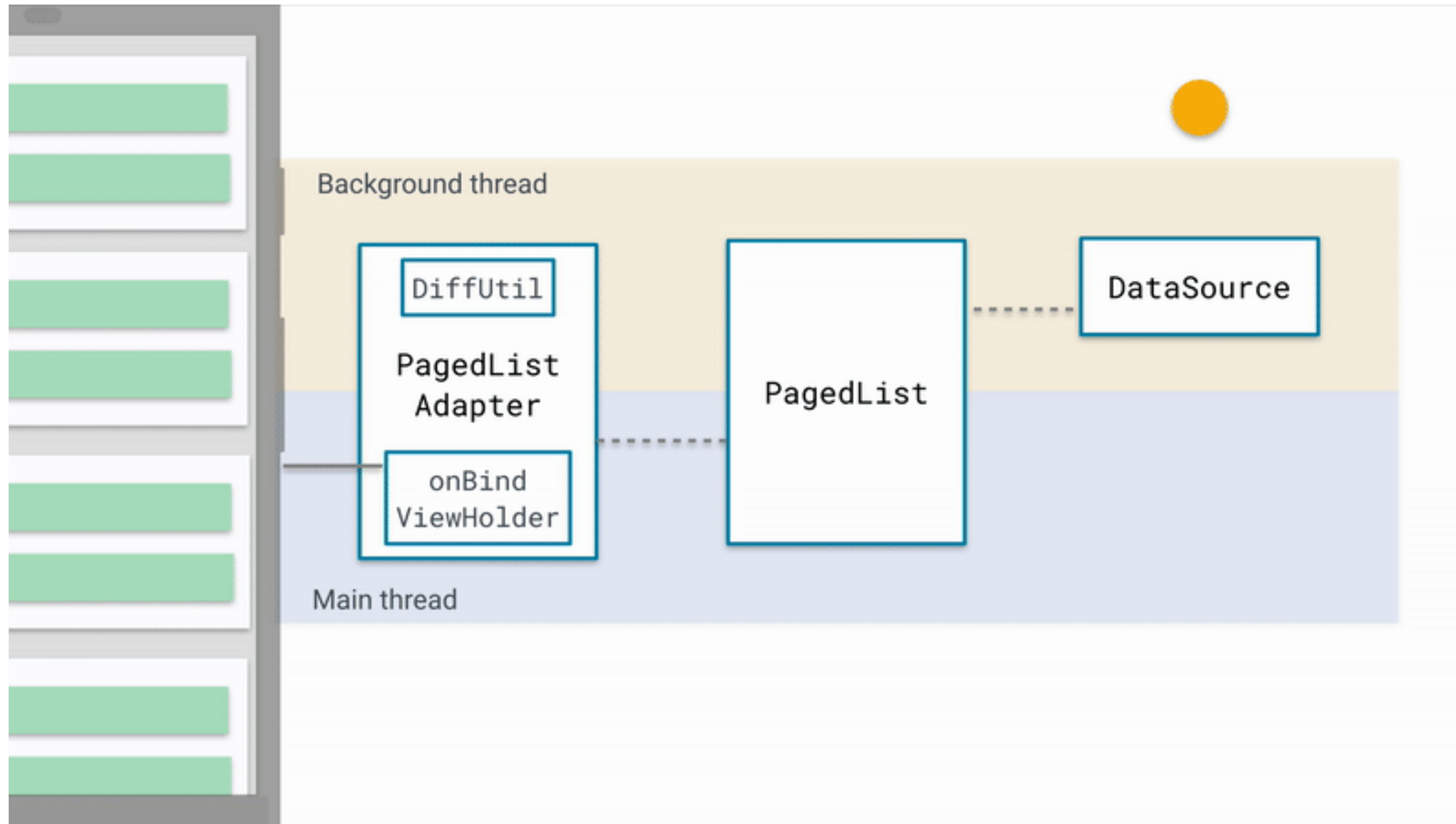
Man kann hierzu das Plugin Gradle View verwenden. Man sieht, das retrofit die okhttp3-Bibliothek in der Version 3.12.0 verwendet. Daher nimmt man auch für das okhttp3-Logging die Version 3.12.0 und nicht 3.13.0

PageListAdapter and DataSource

MVVM-Architecture

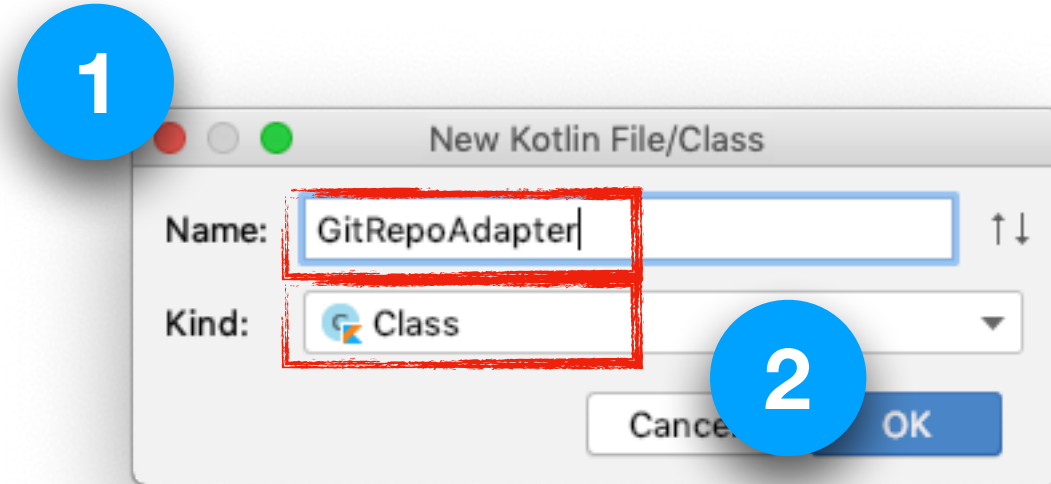


<https://medium.com/@eladb4382/paging-library-viewmodel-livedata-room-and-retrofit-66bf6a0eef9d>



<https://medium.com/@anvith/paging-android-architecture-components-3134212b83bb>

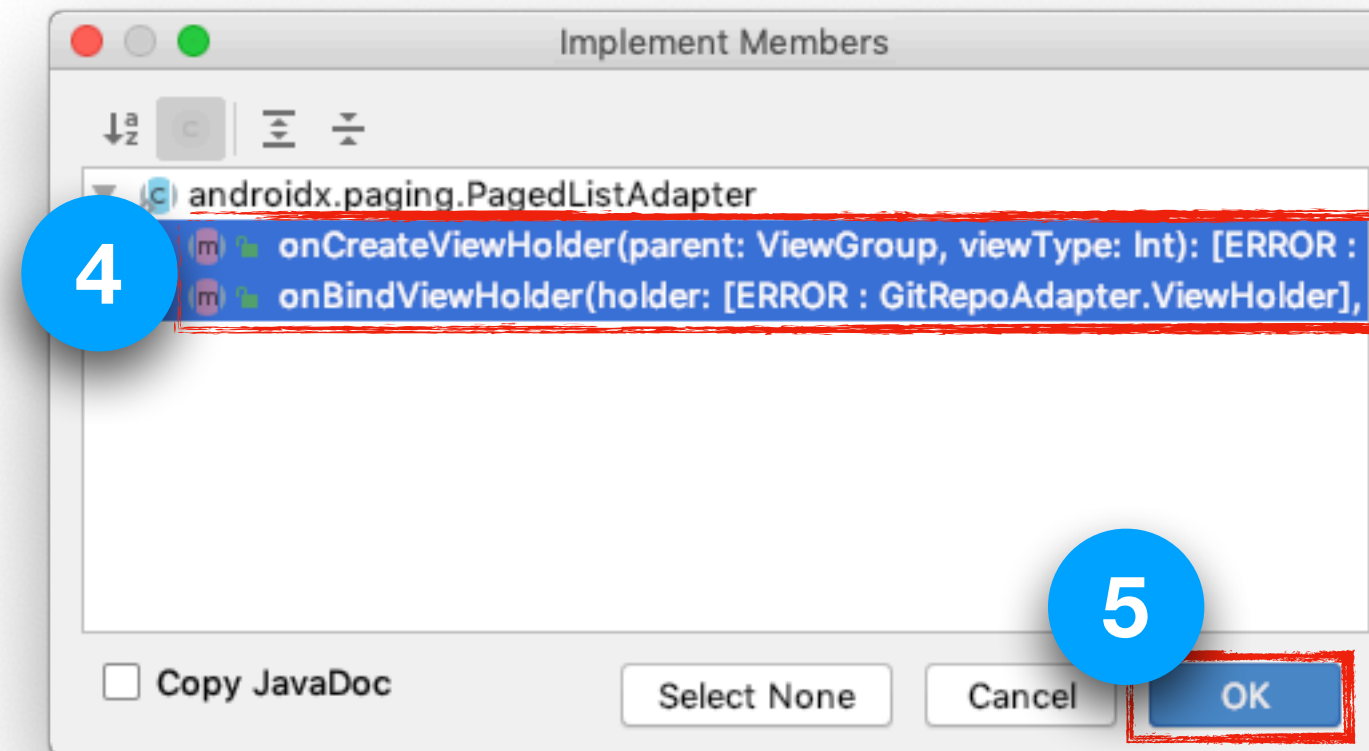
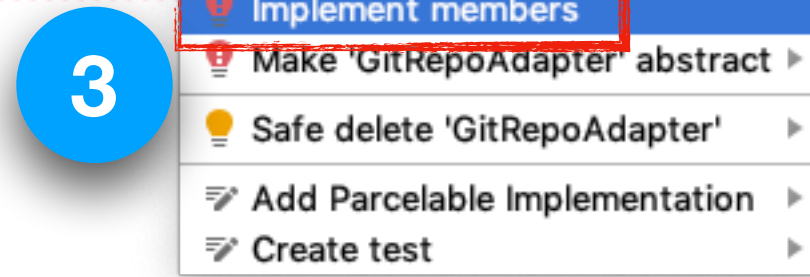
GitRepoAdapter 1



```
package at.htl.paging

import android.view.ViewGroup
import androidx.paging.PagedListAdapter

class GitRepoAdapter: PagedListAdapter<GitRepo, GitRepoAdapter.RepoViewHolder>() {
```



Constructor

```
package at.htl.paging
```

```
import android.view.ViewGroup
import androidx.paging.PagedListAdapter
```

```
class GitRepoAdapter: PagedListAdapter<GitRepo, GitRepoAdapter.RepoViewHolder>() {
```

```
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): GitRepoAdapter.RepoViewHolder {
        TODO("not implemented") //To change body of created functions use File | Settings | File Templates.
    }

```

```
    override fun onBindViewHolder(holder: GitRepoAdapter.RepoViewHolder, position: Int) {
        TODO("not implemented") //To change body of created functions use File | Settings | File Templates.
    }

```

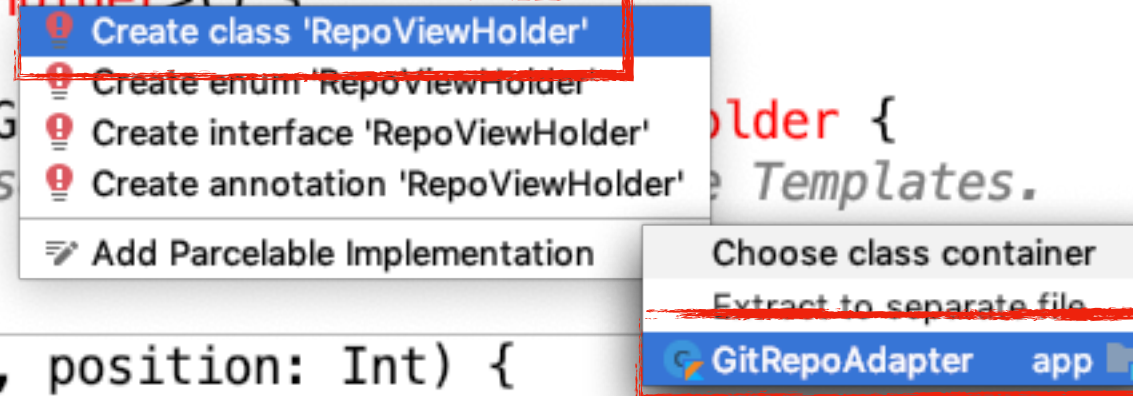
```
}
```

GitRepoAdapter 2

```
package at.htl.paging
```

```
import android.view.ViewGroup  
import androidx.paging.PagedListAdapter
```

```
class GitRepoAdapter: PagedListAdapter<GitRepo, GitRepoAdapter.RepoViewHolder>() {  
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): GitRepoAdapter.RepoViewHolder {  
        TODO("not implemented") //To change body of created functions use File | Settings | File Templates.  
    }  
    override fun onBindViewHolder(holder: GitRepoAdapter.RepoViewHolder, position: Int) {  
        TODO("not implemented") //To change body of created functions use File | Settings | File Templates.  
    }  
}
```



```
package at.htl.paging
```

```
import android.view.View  
import android.view.ViewGroup  
import androidx.paging.PagedListAdapter  
import androidx.recyclerview.widget.RecyclerView
```

```
class GitRepoAdapter : PagedListAdapter<GitRepo, GitRepoAdapter.RepoViewHolder>() {  
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): GitRepoAdapter.RepoViewHolder {  
        TODO("not implemented") //To change body of created functions use File | Settings | File Templates.  
    }  
    override fun onBindViewHolder(holder: GitRepoAdapter.RepoViewHolder, position: Int) {  
        TODO("not implemented") //To change body of created functions use File | Settings | File Templates.  
    }  
    class RepoViewHolder(val view: View) : RecyclerView.ViewHolder(view) {  
    }  
}
```

Klassendefinition ergänzen

GitRepoAdapter 3

```
import android.view.View
import android.view.ViewGroup
import androidx.paging.PagedListAdapter
import androidx.recyclerview.widget.DiffUtil
import androidx.recyclerview.widget.RecyclerView

class GitRepoAdapter : PagedListAdapter<GitRepo, GitRepoAdapter.RepoViewHolder>(REPO_COMPARATOR) {

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): GitRepoAdapter.RepoViewHolder {
        TODO("not implemented") //To change body of created functions use File | Settings | File Templates.
    }

    override fun onBindViewHolder(holder: GitRepoAdapter.RepoViewHolder, position: Int) {
        TODO("not implemented") //To change body of created functions use File | Settings | File Templates.
    }

    class RepoViewHolder(val view: View) : RecyclerView.ViewHolder(view) {
    }

    companion object {

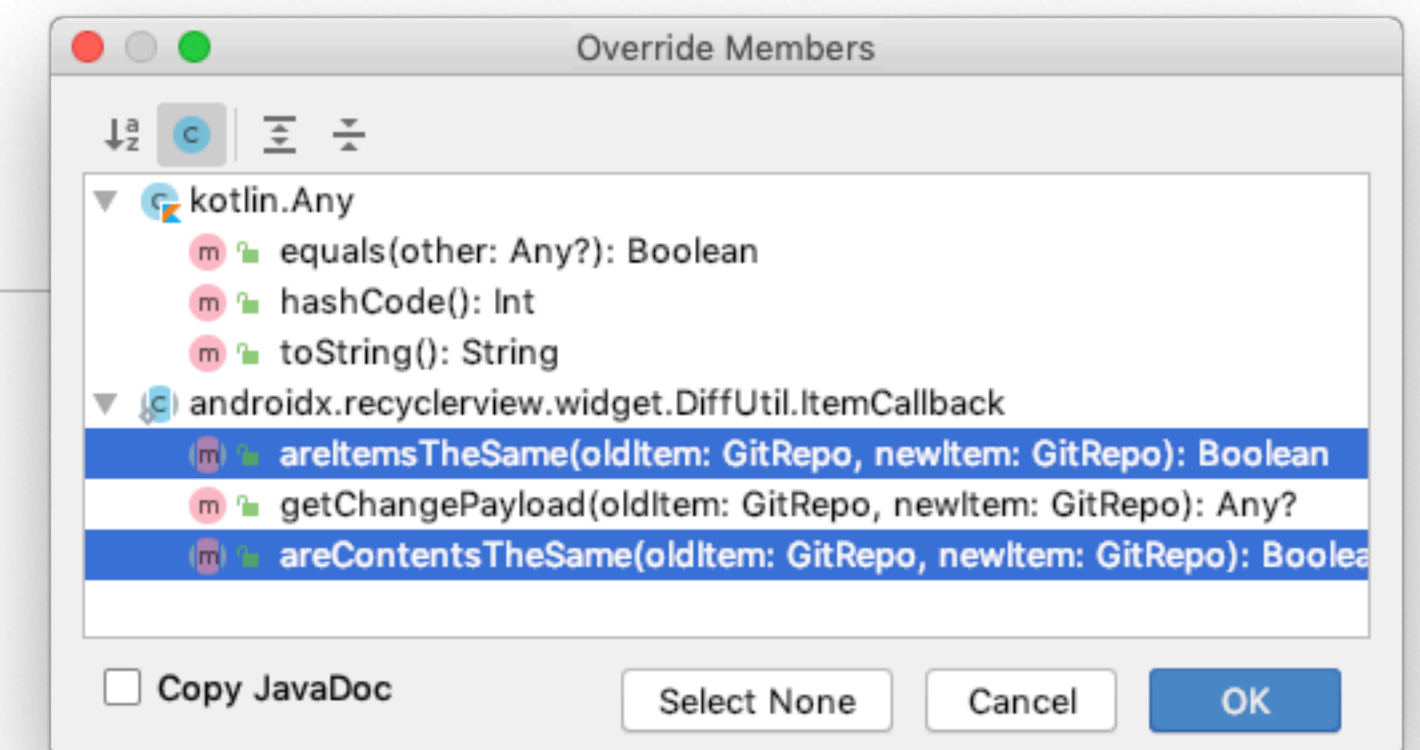
        private val REPO_COMPARATOR = object : DiffUtil.ItemCallback<GitRepo>() {
            override fun areItemsTheSame(oldItem: GitRepo, newItem: GitRepo): Boolean =
                oldItem.fullName == newItem.fullName

            override fun areContentsTheSame(oldItem: GitRepo, newItem: GitRepo): Boolean =
                oldItem == newItem
        }

        // Suspicious equality check: equals() is not implemented in GitRepo more... (%F1)
    }
}
```

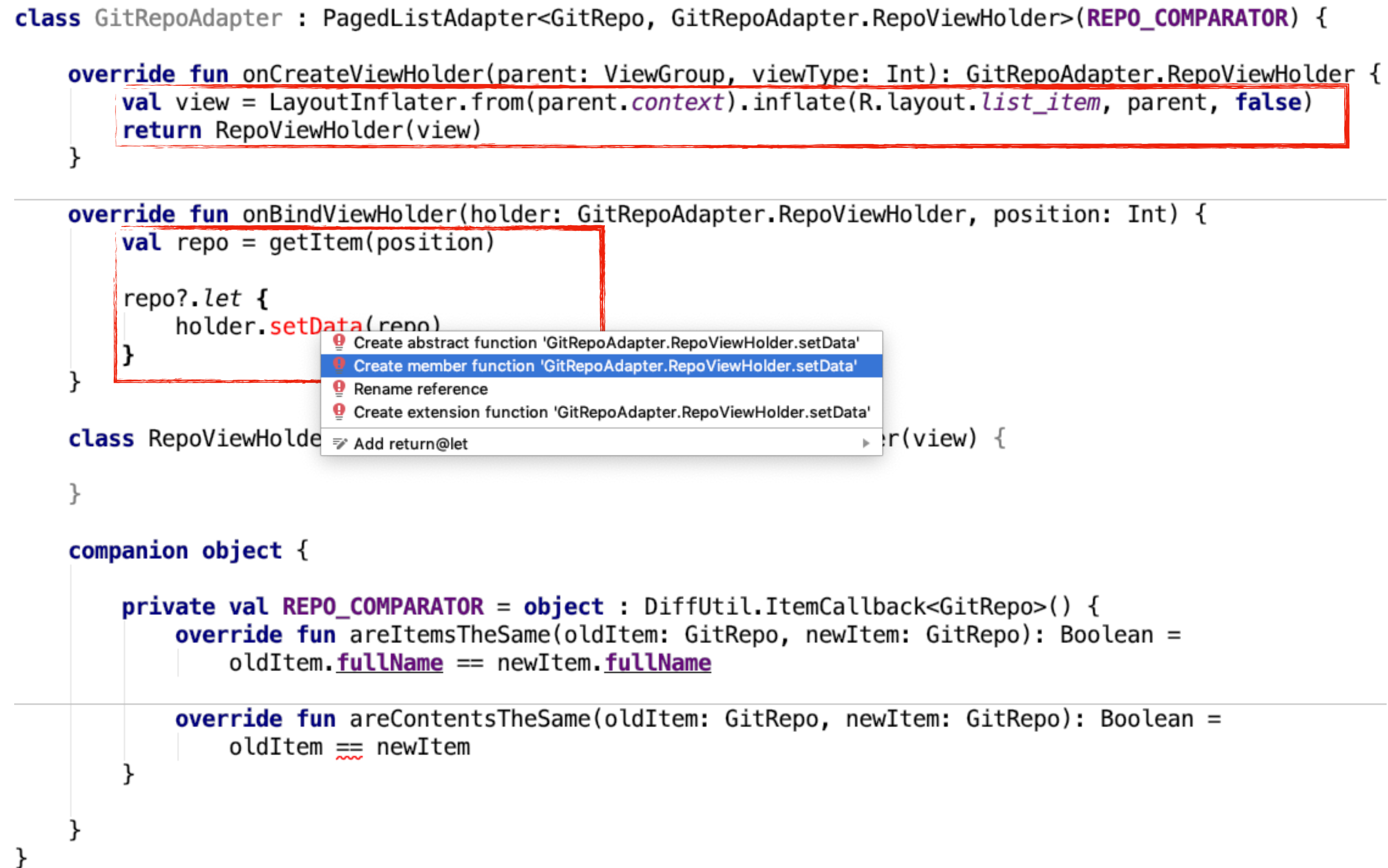
Diese Callback-Methode ermittelt die Unterschiede zwischen Daten und Liste

dazu später ...



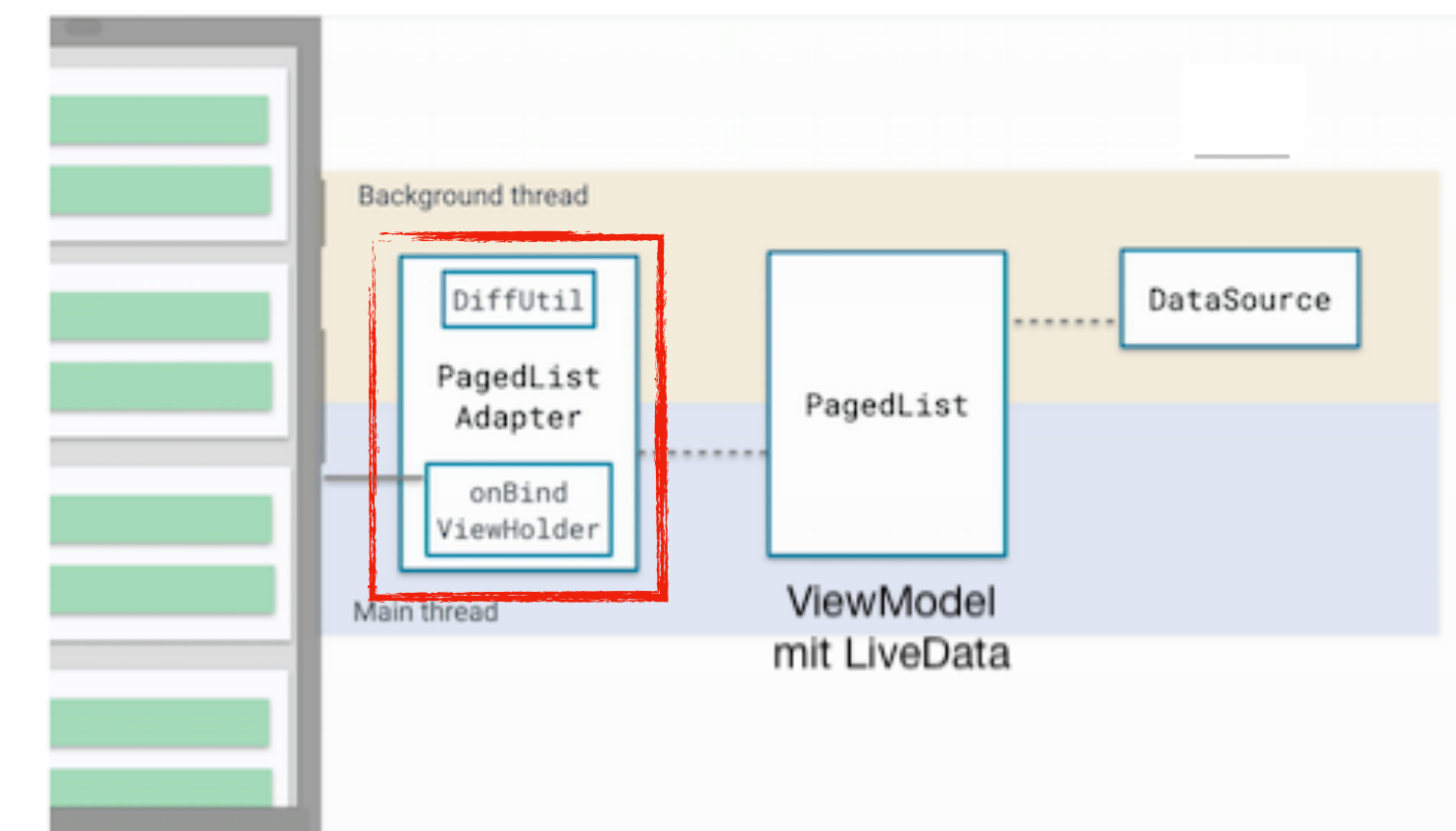
GitRepoAdapter 4

```
class GitRepoAdapter : PagedListAdapter<GitRepo, GitRepoAdapter.RepoViewHolder>(REPO_COMPARATOR) {  
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): GitRepoAdapter.RepoViewHolder {  
        val view = LayoutInflater.from(parent.context).inflate(R.layout.list_item, parent, false)  
        return RepoViewHolder(view)  
    }  
    override fun onBindViewHolder(holder: GitRepoAdapter.RepoViewHolder, position: Int) {  
        val repo = getItem(position)  
        repo?.let {  
            holder.setData(repo)  
        }  
    }  
    class RepoViewHolder(view: View) {  
    }  
    companion object {  
        private val REPO_COMPARATOR = object : DiffUtil.ItemCallback<GitRepo>() {  
            override fun areItemsTheSame(oldItem: GitRepo, newItem: GitRepo): Boolean =  
                oldItem.fullName == newItem.fullName  
            override fun areContentsTheSame(oldItem: GitRepo, newItem: GitRepo): Boolean =  
                oldItem == newItem  
        }  
    }  
}
```



RepoViewHolder

```
class RepoViewHolder(val view: View) : RecyclerView.ViewHolder(view) {  
  
    fun setData(gitRepo: GitRepo) {  
        with(view) {  
            repo_name.text = gitRepo.fullName  
            repo_description.text = gitRepo.description  
            repo_language.text = gitRepo.language  
            repo_stars.text = gitRepo.stars.toString()  
            repo_forks.text = gitRepo.forks.toString()  
        }  
    }  
}
```



3 implementations of DataSource

- **PositionalDataSource** offers position-based data loader for a fixed-size, countable data set. For instance, it allows scrolling through a list of contacts or jumping to a particular position in the list (i.e. jumping to contacts that start with a particular letter). This is how Room implements paging by default under the hood. The repo for this article contains examples of implementing PositionalDataSource with LiveData or PositionalDataSource with RxJava.
- **ItemKeyedDataSource** can be used when you can identify items before the start of the current list and after the end of the current list based on particular item key because the keys are ordered.
- **PageKeyedDataSource** is often used for remote API calls to load data in chunks where the response returned will contain the data itself as well as pointers to next and previous pages of data (much like a LinkedList). It reduces the size of the data transfer for a given scroll. Since the total count is usually not available in remote APIs, placeholders should be disabled.

GitRepoDataSource

```
package at.htl.paging
```

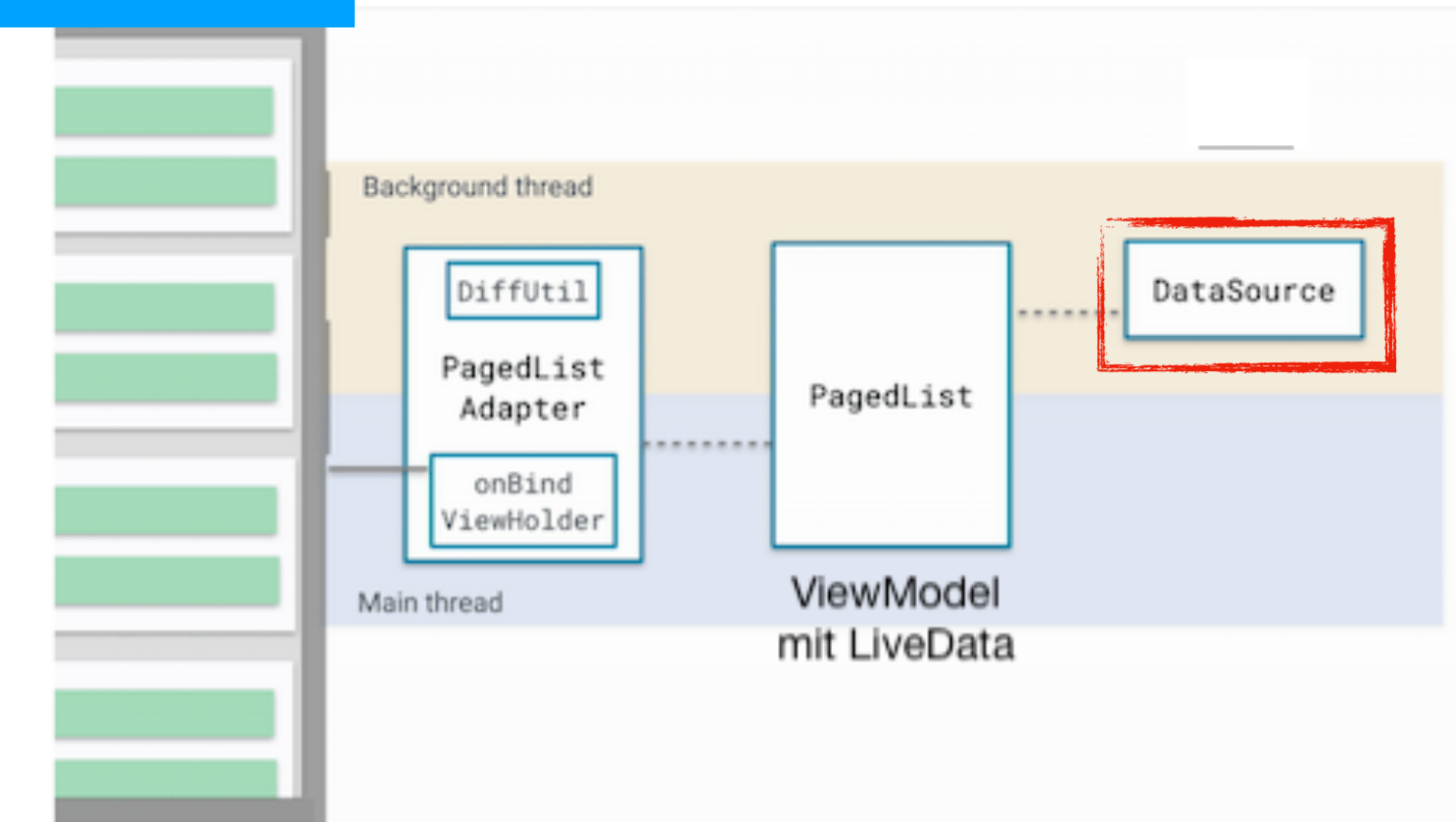
```
import androidx.paging.PageKeyedDataSource
```

```
class GitRepoDataSource : PageKeyedDataSource<Int, GitRepo>() {  
    override fun loadInitial(params: LoadInitialParams<Int>, callback: LoadInitialCallback<Int, GitRepo>) {  
        TODO("not implemented") wird beim ersten Laden ausgeführt  
    }  
}
```

```
    override fun loadAfter(params: LoadParams<Int>, callback: LoadCallback<Int, GitRepo>) {  
        TODO("not implemented") wird beim Wechsel zu nächsten Seite ausgeführt  
    }  
}
```

```
    override fun loadBefore(params: LoadParams<Int>, callback: LoadCallback<Int, GitRepo>) {  
        TODO("not implemented") wird beim Wechsel zu vorigen Seite ausgeführt  
    }  
}
```

```
companion object {  
    const val PAGE_SIZE = 10  
    const val FIRST_PAGE = 1  
    const val TOPIC = "android"  
}
```



loadInitial(...)

```
package at.htl.paging

import androidx.paging.PageKeyedDataSource
import retrofit2.Call
import retrofit2.Callback
import retrofit2.Response

class GitRepoDataSource : PageKeyedDataSource<Int, GitRepo>() {

    override fun loadInitial(params: LoadInitialParams<Int>, callback: LoadInitialCallback<Int, GitRepo>) {

        val service = GitRepoServiceBuilder.buildService(GitRepoService::class.java)
        val call = service.getRepositories(FIRST_PAGE, PAGE_SIZE, TOPIC)

        call.enqueue(object : Callback<GitRepoResponse> {
            // if you receive a HTTP response, then this method is executed
            override fun onResponse(call: Call<GitRepoResponse>, response: Response<GitRepoResponse>) {

                if (response.isSuccessful) {
                    val apiResponse = response.body()!!
                    val responseItems = apiResponse.items
                    responseItems?.let {
                        callback.onResult(responseItems, null, FIRST_PAGE + 1)
                    }
                }
            }
        })

        // invoked in case of network error or establishing connection with server
        // or error creating http request or error processing http response
        override fun onFailure(call: Call<GitRepoResponse>, t: Throwable) {
        }
    }
}
```

nächste Seite

vorige Seite

loadAfter(...)

```
override fun loadAfter(params: LoadParams<Int>, callback: LoadCallback<Int, GitRepo>) {  
    val service = GitRepoServiceBuilder.buildService(GitRepoService::class.java)  
    val call = service.getRepositories(params.key, PAGE_SIZE, TOPIC)  
  
    call.enqueue(object : Callback<GitRepoResponse> {  
  
        override fun onResponse(call: Call<GitRepoResponse>, response: Response<GitRepoResponse>) {  
  
            if (response.isSuccessful) {  
                val apiResponse = response.body()!!  
                val responseItems = apiResponse.items  
  
                val key = if (apiResponse.totalCount > params.key) {  
                    params.key + 1  
                } else {  
                    0  
                }  
  
                responseItems?.let {  
                    callback.onResult(responseItems, null)  
                }  
            }  
        }  
    })  
  
    override fun onFailure(call: Call<GitRepoResponse>, t: Throwable) {  
    }  
}
```

Wenn die Gesamtanzahl > ist als unser derzeitiger Schlüsselwert, wird die nächste Seite geholt, sonst nicht

loadBefore(...)

```
override fun loadBefore(params: LoadParams<Int>, callback: LoadCallback<Int, GitRepo>) {
    val service = GitRepoServiceBuilder.buildService(GitRepoService::class.java)
    val call = service.getRepositories(params.key, PAGE_SIZE, TOPIC)

    call.enqueue(object : Callback<GitRepoResponse> {

        override fun onResponse(call: Call<GitRepoResponse>, response: Response<GitRepoResponse>) {

            if (response.isSuccessful) {
                val apiResponse = response.body()!!
                val responseItems = apiResponse.items

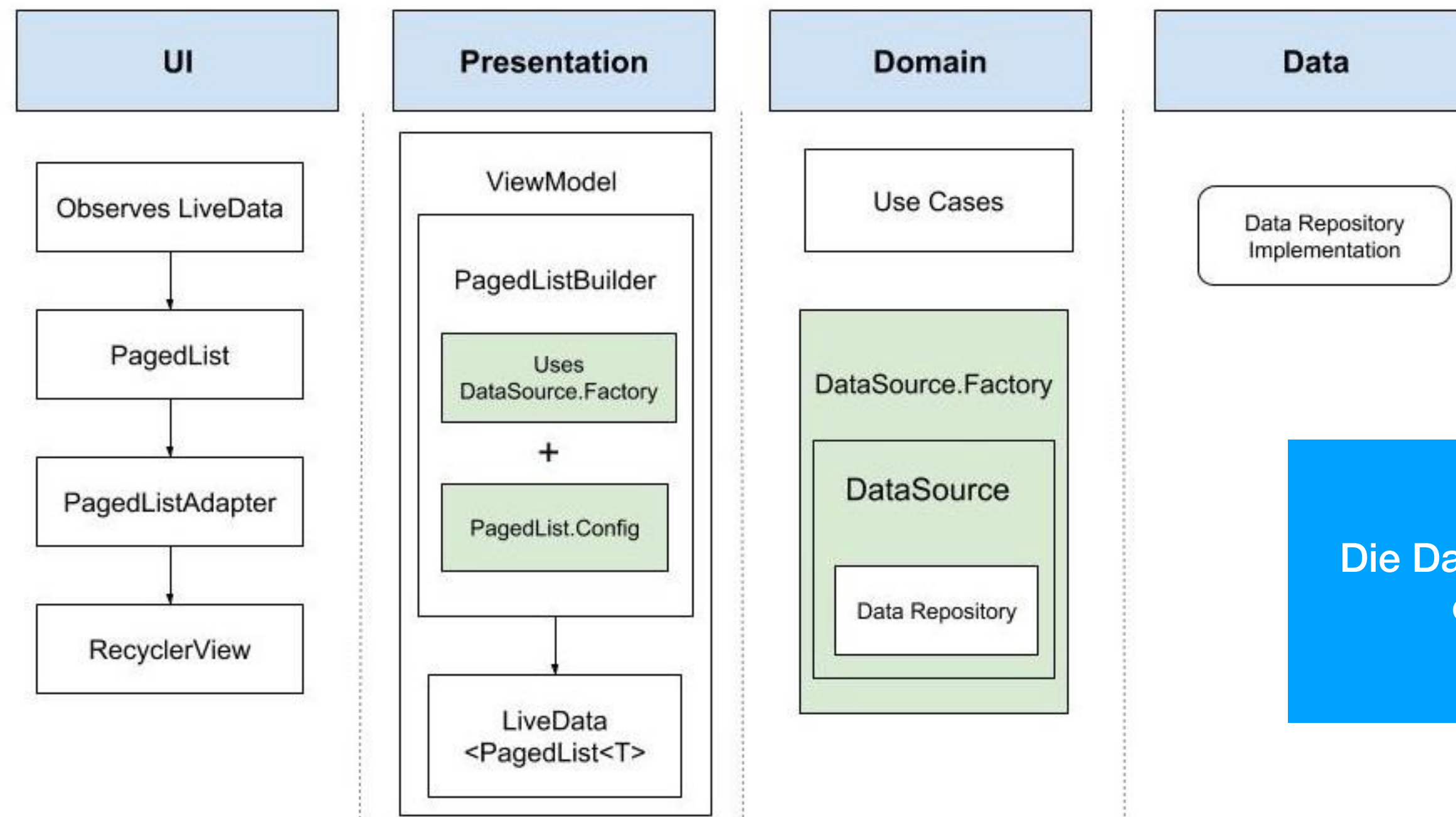
                val key = if (params.key > 1) {
                    params.key - 1
                } else {
                    0
                }

                responseItems?.let {
                    callback.onResult(responseItems, null)
                }
            }
        }

        override fun onFailure(call: Call<GitRepoResponse>, t: Throwable) {
        }

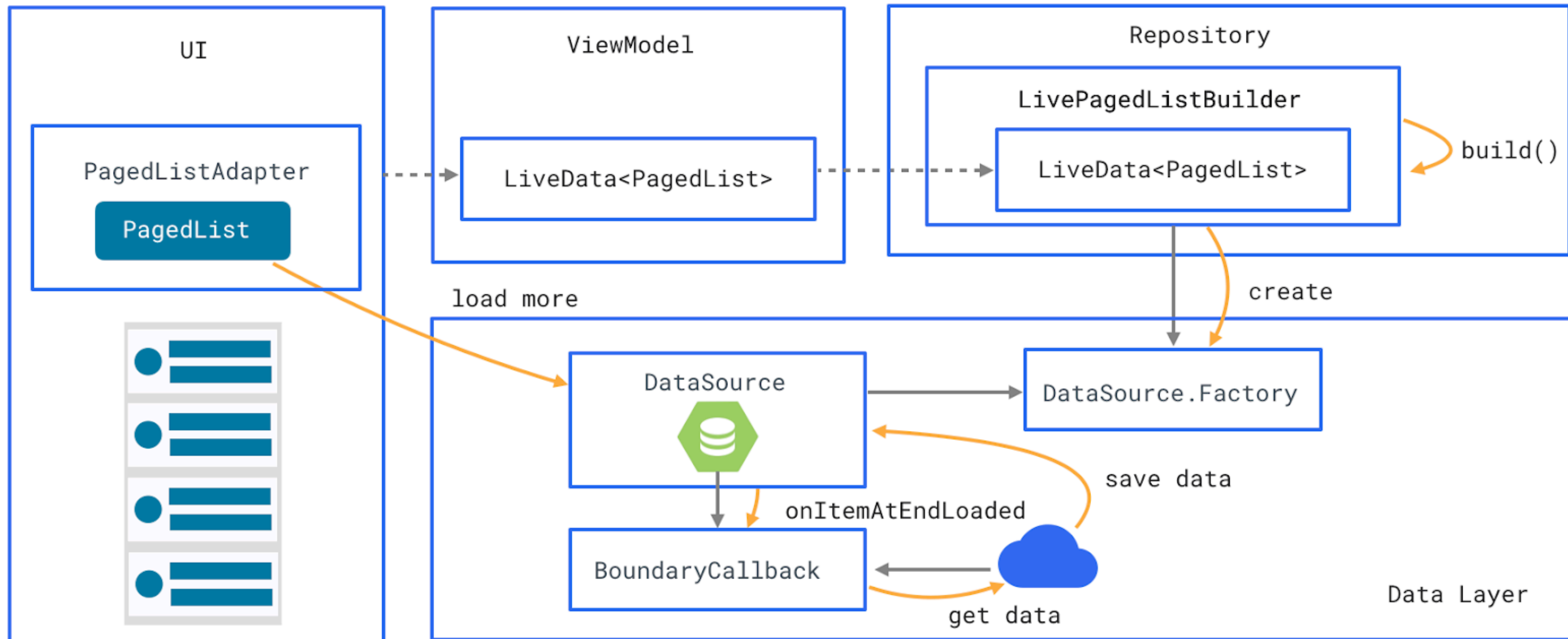
    })
}
```

Observable List and populationg UI



Die DataSourceFactory wird benötigt um eine observable PagedList zu erzeugen

<https://proandroiddev.com/exploring-paging-library-from-jetpack-c661c7399662>



<https://www.capttechconsulting.com/blogs/an-overview-of-android-jetpack>

<https://medium.com/@sharmadhiraj.np/android-paging-library-step-by-step-implementation-guide-75417753d9b9>

GitRepoDataSourceFactory

```
package at.htl.paging

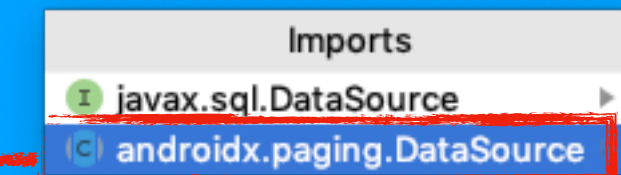
import androidx.lifecycle.MutableLiveData
import androidx.paging.DataSource

class GitRepoDataSourceFactory : DataSource.Factory<Int, GitRepo>() {

    val gitRepoLiveDataSource = MutableLiveData<GitRepoDataSource>()

    override fun create(): DataSource<Int, GitRepo> {
        val repoDataSource = GitRepoDataSource()
        gitRepoLiveDataSource.postValue(repoDataSource)
        return repoDataSource
    }
}
```

ACHTUNG: Verwenden sie den korrekten Import



GitRepoViewModel

```
package at.htl.paging
```

```
import androidx.lifecycle.LiveData
import androidx.lifecycle.ViewModel
import androidx.paging.LivePagedListBuilder
import androidx.paging.PagedList
```

```
class GitRepoViewModel : ViewModel() {
```

```
    var gitRepoPagedList: LiveData<PagedList<GitRepo>>
    private var liveDatasource: LiveData<GitRepoDataSource>
```

```
    init {
        val itemDataSourceFactory = GitRepoDataSourceFactory()
        liveDatasource = itemDataSourceFactory.gitRepoLiveDataSource

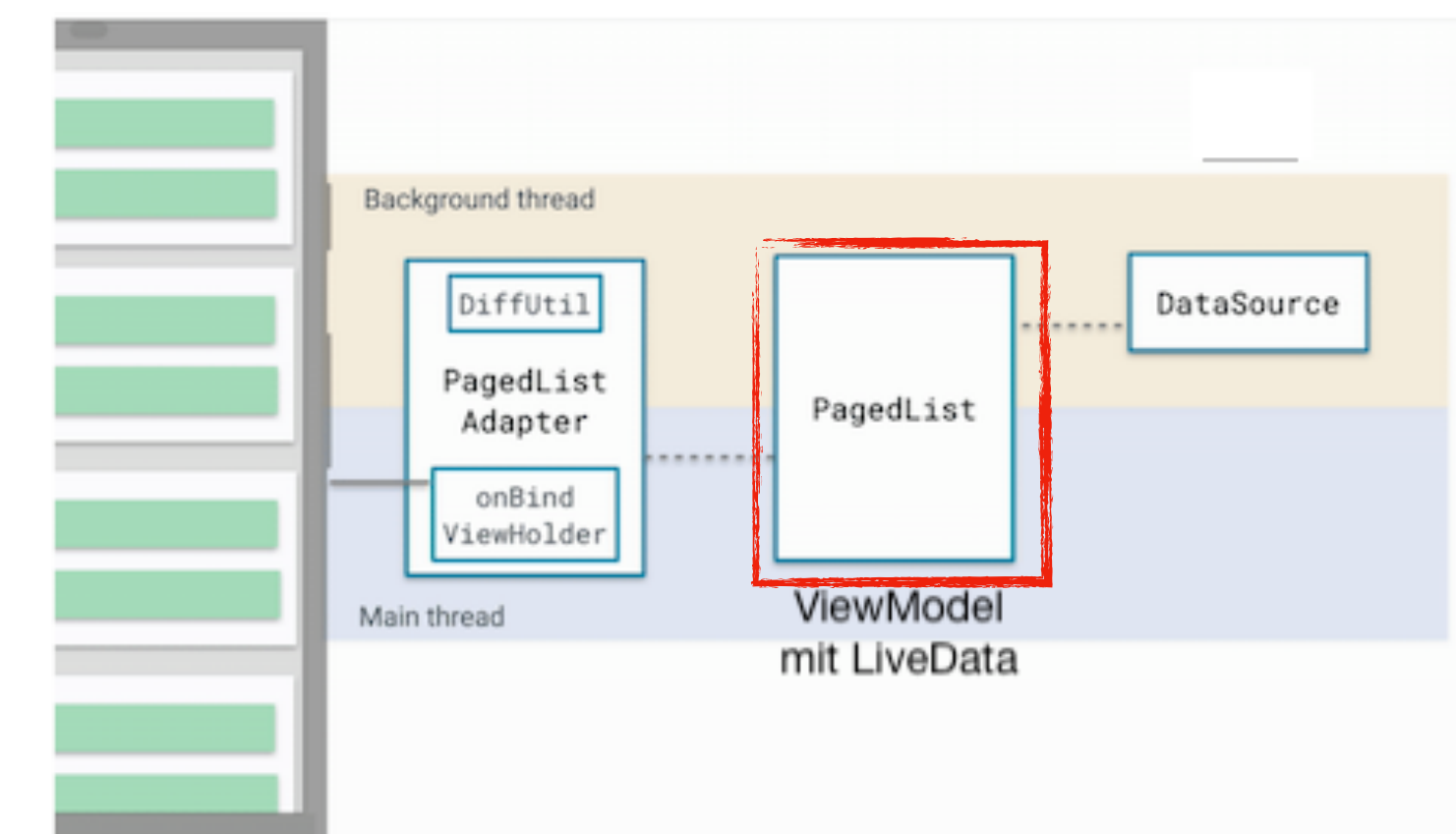
        val config = PagedList.Config.Builder()
            .setEnablePlaceholders(false)
            .setPageSize(GitRepoDataSource.PAGE_SIZE)
            .build()

        gitRepoPagedList = LivePagedListBuilder(itemDataSourceFactory, config)
            .build()
    }
}
```

Hier verwenden wir die vorhin erstellte Factory

Placeholder geben an, ob Platzhalter angezeigt werden, wenn die Daten noch nicht geladen sind

<https://developer.android.com/reference/android/arch/paging/PagedList#placeholders>



Aus MainActivity aufrufen

```
package at.htl.paging
```

```
import androidx.appcompat.app.AppCompatActivity  
import android.os.Bundle  
import androidx.lifecycle.Observer  
import androidx.lifecycle.ViewModelProviders  
import androidx.recyclerview.widget.LinearLayoutManager  
import kotlinx.android.synthetic.main.activity_main.*
```

```
class MainActivity : AppCompatActivity() {
```

```
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)
```

```
        1 val adapter = GitRepoAdapter()  
          recycler_view.layoutManager = LinearLayoutManager(this)
```

```
        2 val itemViewModel = ViewModelProviders.of(this).get(GitRepoViewModel::class.java)
```

```
        3 itemViewModel.gitRepoPagedList.observe(this, Observer {  
          adapter.submitList(it) 4  
        })
```

```
        5 recycler_view.adapter = adapter
```

```
    }
```

```
}
```

1 Erstellen einer Instanz des
LinearLayoutManagers

2 Erstellen einer Instanz eines
ViewModels

3 es wird gitRepoPagedList
beobachtet

4 Mit submit wird die neue List in
der RecyclerView gesetzt

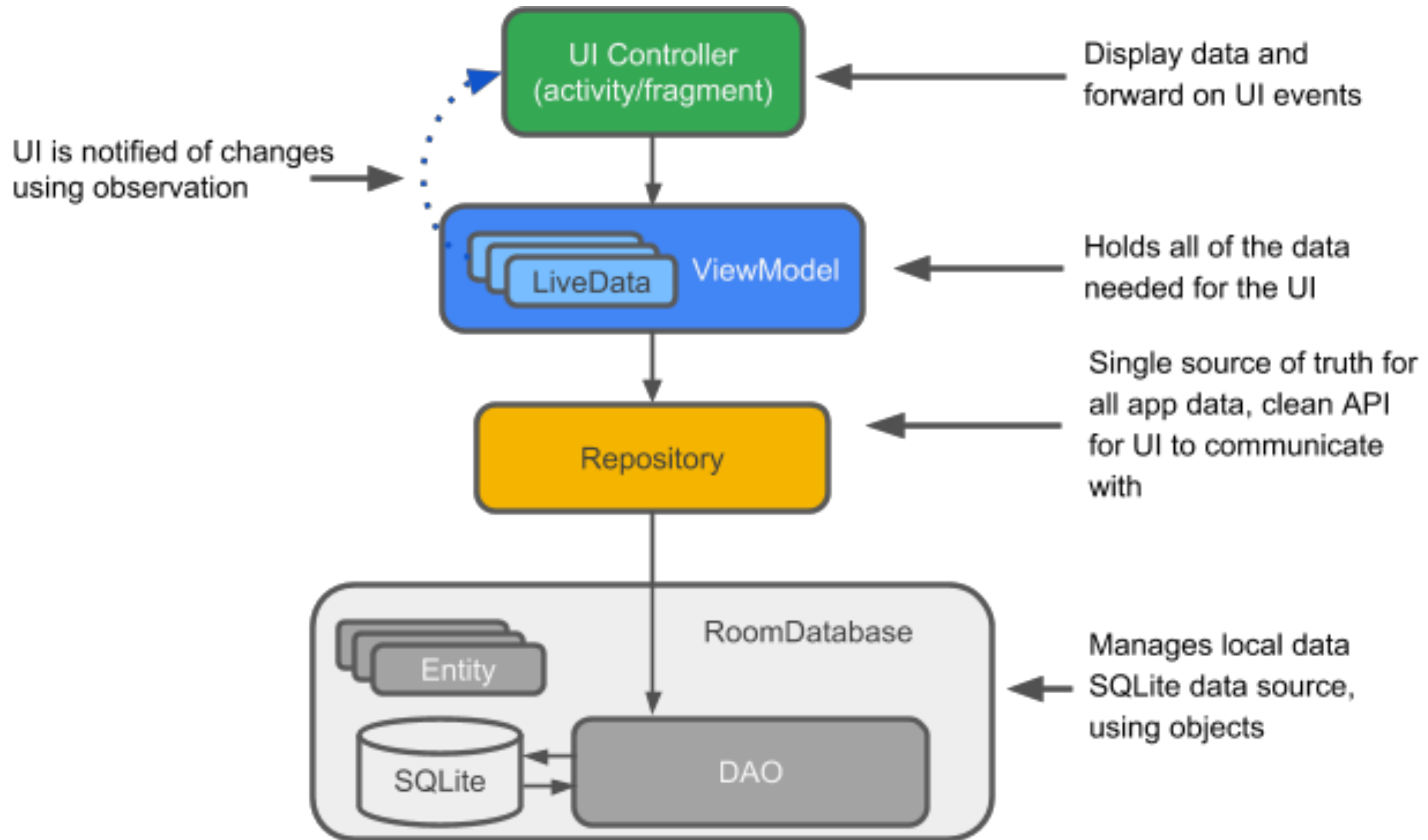
5 Setzen des LayoutAdapters in
der RecyclerView

Sources

- <http://androidkt.com/paging-library/>
- <http://thetechnocafe.com/understanding-livedata-in-android-architecture-components/>
-

Wie gehts weiter?

- <https://codelabs.developers.google.com/codelabs/android-room-with-a-view/#0>
-





Noch
Fragen?

Source

<https://www.udemy.com/android-jetpack-architecture-components/>

The screenshot shows the UdeMy course page for 'Android Jetpack Architecture Components'. The page features a dark blue header with the UdeMy logo, navigation links for 'Categories', 'Search for any skill', 'UdeMy for Business', 'Become an instructor', 'Log In', and 'Sign Up'. Below the header, the course title is prominently displayed in white text on a dark background. The course description reads: 'Utilize Android Jetpack Architecture components to make your Android application development flexible and maintainable'. It also shows a 'NEW' badge, a rating of 0.0 (0 ratings), and 4 students enrolled. The course is created by Packt Publishing and was last updated in 2/2019. A 'Preview this course' button is visible on the right side of the course card.



What you'll learn

- ✓ Get introduced to Android architecture components
- ✓ Provide stability in your app by handling life cycles, view models, and live data
- ✓ Load data gradually and gracefully in Recycler View by using the Paging library
- ✓ Explore how to perform CRUD operations in the Room database
- ✓ Use the Data Binding library to bind data to the UI
- ✓ Implement effective in-app navigation by using the Navigation architecture component
- ✓ Implement a local database to store structured data by using the Room database
- ✓ Schedule tasks asynchronously by using Work Manager

€10.99 ~~€124.99~~ 91% off
🕒 5 hours left at this price!

[Add to cart](#)

[Buy now](#)

30-Day Money-Back Guarantee

Includes

- 📺 3 hours on-demand video
- 📄 1 downloadable resource
- 🌐 Full lifetime access
- 📱 Access on mobile and TV
- 🏆 Certificate of Completion