Modern data transfer in Android

Denis Aleksandrov, Arcadia

SECR

About me

3 years legacy

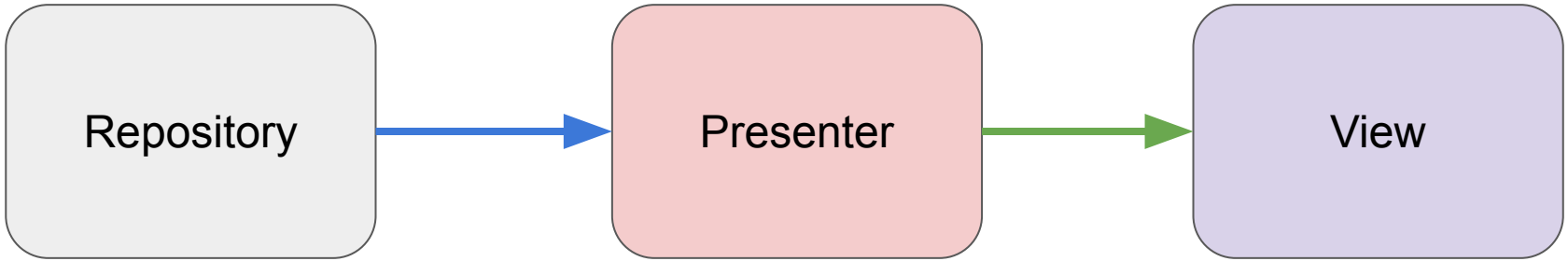
5 years enterprise

2+ years outsource

1+ years education

Leading Engineer, Lecturer

data transfer



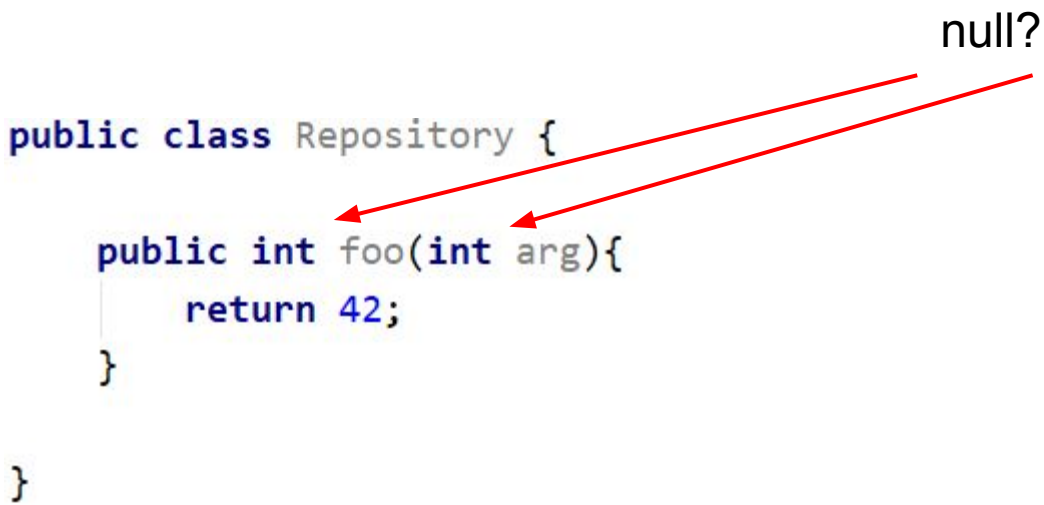
```java
public class Repository {

    public int foo(int arg){
        return 42;
    }

}
```

That hurts

- Nullability


null?
public class Repository {
public int foo(int arg){
return 42;
}
}

```java
@Nullable
public Integer foo(@Nullable Integer arg){
    return 42;
}
```

Integer == int?

@Nullable/@NotNull everywhere?

It’s works!

```
@Nullable
@NotNull
public Integer foo(@Nullable Integer arg){
    return 42;
}
```

That hurts

- Nullability
- Exceptions

```java
@Nullable
public Integer foo(@Nullable Integer arg) throws Exception{
    return 42;
}
```

That hurts

- Nullability
- Exceptions
- Async calls

```
@NotNull
public void foo(@Nullable Integer arg, Result handler){
   handler.onSuccess( result: 42);
}

interface Result{
    public void onSuccess(Integer result);
    public void onFailure(Throwable exceprion);
}
```

That hurts

- Nullability
- Exceptions
- Async calls
- Single format

Variants

Call

Promise

Completable

Callbacks

Async

Future

Observable

Livedata

That hurts

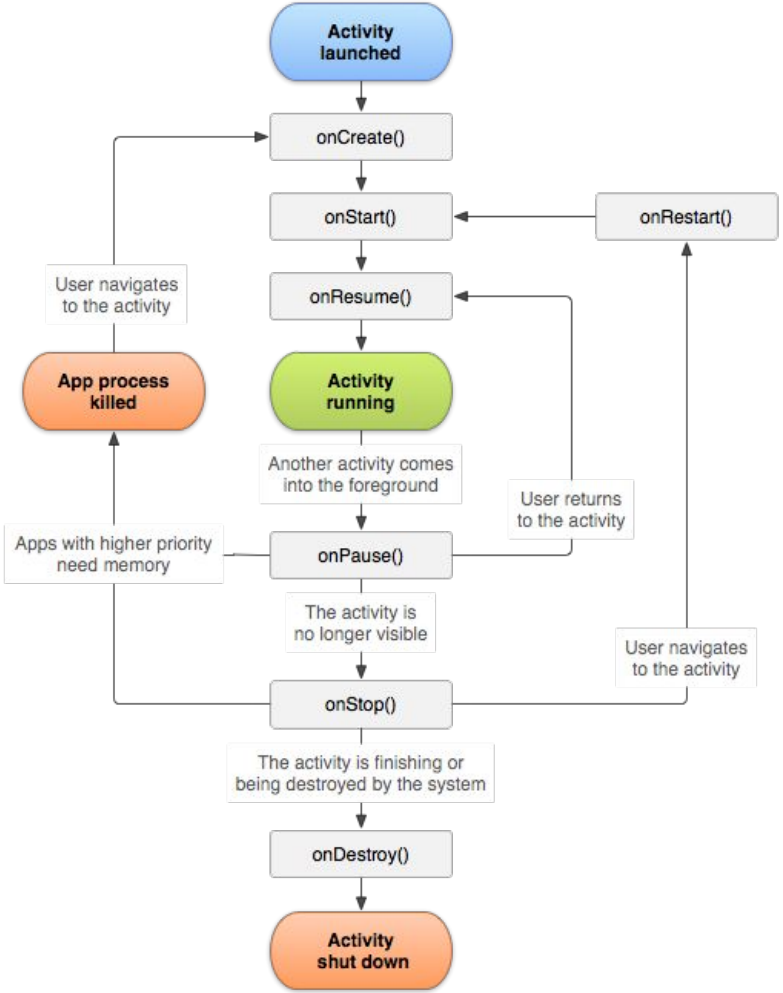
- Nullability
- Exceptions
- Async calls
- Single format
- Thread dependencies

```java
// call me in background thread
public void foo(@Nullable Integer arg, Result handler){
    handler.onSuccess( result: 42);
}
```

```
@MainThread
public void observe(@NonNull LifecycleOwner owner
    assertMainThread( methodName: "observe");
```

That hurts

- Nullability
- Exceptions
- Async calls
- Single format
- Thread dependencies
- Life cycles


Activity launched
onCreate()
onStart()
onRestart()
onResume()
User navigates to the activity
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

Android dev is painful
but
we have new tablets

Android pain pills

- Kotlin

```
@NotNull
public void foo(@Nullable Integer arg, Result handler){
   handler.onSuccess( result: 42);
}

interface Result{
    public void onSuccess(Integer result);
    public void onFailure(Throwable exceprion);
}
```

```
fun foo(
    arg: Int?,
    onSuccess: (Int?) -> Unit,
    onFailure: (Throwable) -> Unit
) {
    onSuccess(42)
}
```

Android pain pills

- Kotlin
- Coroutines

```
fun foo(
    arg: Int?,
    onSuccess: (Int?) -> Unit,
    onFailure: (Throwable) -> Unit
) {
    onSuccess(42)
}
```

```
fun foo(arg: Int?): Int?

suspend fun fooToo(arg: Int?): Int?
```

```java
public interface ktRepo {
    @Nullable
    Integer foo(@Nullable Integer var1);

    @Nullable
    Object fooToo(@Nullable Integer var1, @NotNull Continuation var2);
}
```

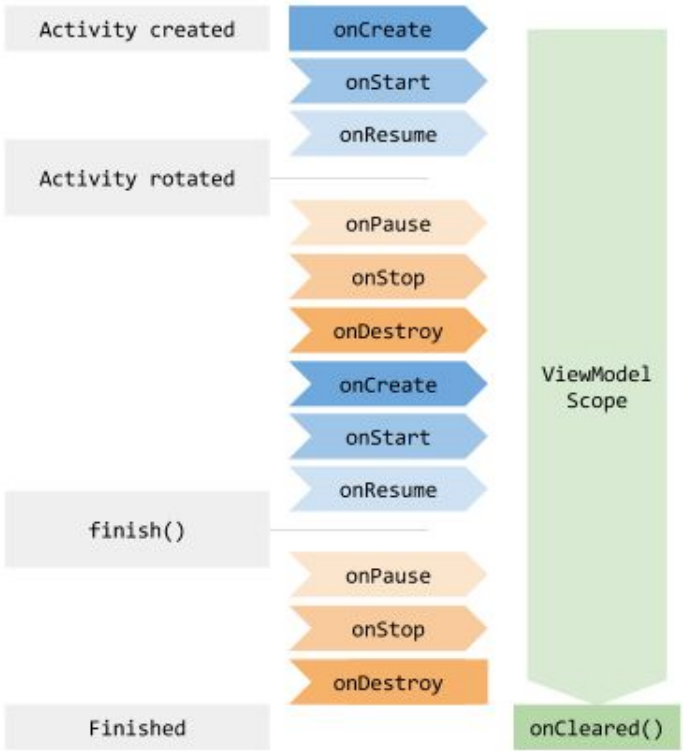
```
suspend fun fooToo(arg: Int): Int = withContext(Dispatchers.IO) {
    return@withContext 42
}
```

Async code
looks like
sync code

Lifecycle management

```
val ViewModel.viewModelScope: CoroutineScope
```

```
/**
 * [CoroutineScope] tied to this [ViewModel].
 * This scope will be canceled when ViewModel will be cleared,
 * i.e [ViewModel.onCleared] is called
 *
 * This scope is bound to
 * [Dispatchers.Main.immediate]
 * [MainCoroutineDispatcher.immediate]
 */
```


Activity created
onCreate
onStart
onResume
Activity rotated
onPause
onStop
onDestroy
onCreate
onStart
onResume
finish()
onPause
onStop
onDestroy
Finished
ViewModel Scope
onCleared()

Scopes + Jobs

```kotlin
val job = launch {
    println("Coroutine start")
    try {
        delay(Long.MAX_VALUE)
    } catch (e: CancellationException) {
        println("Coroutine cancelled - ${e.message}")
        throw e
    }
}
delay(1000)
job.cancelAndJoin()
println("Done")
```

https://proandroiddev.com/kotlin-coroutine-job-hierarchy-finish-cancel-and-fail-2d3d42a768a9

Android pain pills

- Kotlin
- Coroutines
- LiveData

```
class SampleViewModel:ViewModel() {



}
```

Android pain pills

- Kotlin
- Coroutines
- LiveData
- Jetpack compose

```
@Composable fun Counter() {
  // introduce a state value (of type `Int`, with initial value of `0`.
  // Note: the `+` syntax is temporary
  val count = +state { 0 }
  // use it to compose your UI. pass it into other composables as parameters
  Text(text="Count: ${count.value}")
  // modify the value inside of event handlers, for instance
  Button(text="Increment, onClick = { count.value += 1; })
}
```

https://blog.karumi.com/android-jetpack-compose-review/

Benefits

- Single format
- Complex details
- Sync code looks like Async code
- UI transfer must be auto
- Less technical code, more business

Questions?

https://t.me/guitariz

github.com/d-aleksandrov