

Android ViewModel 引入协程



simpleeeeeee 关注

3 2019.05.12 00:12:44 字数 460 阅读 6,543

AndroidX Lifecycle v2.1.0 在 ViewModel 中引入 `viewModelScope`，当 ViewModel 被销毁时它会自动取消协程任务，这个特性真的好用。本文介绍 `viewModelScope` 使用和内部实现方式，分析 ViewModel 是如何自动取消协程的。

ViewModel 引入协程

当我们在 ViewModel 里面需要引入协程，首先要在 ViewModel 中新建一个 `CoroutineScope`，用来管理所有协程任务，同时需要 `onCleared()` 方法里面取消协程任务，模板代码实现如下：

```
1 class MyViewModel : ViewModel() {
2
3     private val viewModelJob = SupervisorJob()
4
5     private val uiScope = CoroutineScope(Dispatchers.Main + viewModelJob)
6
7     override fun onCleared() {
8         super.onCleared()
9         viewModelJob.cancel() // Cancel all coroutines
10    }
11
12    fun launchDataLoad() {
13        uiScope.launch {
14            sortList()
15            // Modify UI
16        }
17    }
18
19    suspend fun sortList() = withContext(Dispatchers.Default) {
20        // Heavy work
21    }
22 }
```

然而，很多情况我们会经常忘记取消协程，导致出现内存泄漏等各种问题。遇到这种场景，可以使用 ViewModel 扩展属性 `viewModelScope` 来优化代码。

viewModelScope 方式

注意 lifecycle-viewmodel-ktx 版本号: 2.1.0-beta01

`viewModelScope` 管理协程的方式与我们在 ViewModel 引入协程的方式一样，使用非常简单，模板代码实现如下：

```
1 class MyViewModel : ViewModel() {
2
3     fun launchDataLoad() {
4         viewModelScope.launch {
5             sortList()
6             // Modify UI
7         }
8     }
9
10    suspend fun sortList() = withContext(Dispatchers.Default) {
11        // Heavy work
12    }
13 }
```

viewModelScope 内部实现

```

1 // androidx.lifecycle.ViewModel.viewModelScope
2
3 private const val JOB_KEY = "androidx.lifecycle.ViewModelCoroutineScope.JOB_KEY"
4
5 val ViewModel.viewModelScope: CoroutineScope
6     get() {
7         val scope: CoroutineScope? = this.getTag(JOB_KEY)
8         if (scope != null) {
9             return scope
10        }
11        return setTagIfAbsent(JOB_KEY,
12            CloseableCoroutineScope(SupervisorJob() + Dispatchers.Main))
13    }
14
15 internal class CloseableCoroutineScope(context: CoroutineContext) : Closeable, CoroutineScope {
16     override val coroutineContext: CoroutineContext = context
17
18     override fun close() {
19         coroutineContext.cancel()
20     }
21 }

```

分析 `viewModelScope` 源码有 3 点需要关注：

1. 注意使用 `SupervisorJob` 而不是用 `Job`
2. 为了 `ViewModel` 能够取消协程，需要实现 `Closeable` 接口
3. `viewModelScope` 默认使用 `Dispatchers.Main`，方便 `Activity` 和 `Fragment` 更新 UI

ViewModel 内部取消协程

`ViewModel` 类通过 `HashMap` 存储 `CoroutineScope` 对象，当使用 `getTag(JOB_KEY)` 方法获取对象不存在时，创建一个新的 `CoroutineScope` 并调用 `setTagIfAbsent(JOB_KEY, scope)` 方法存储新建的 `CoroutineScope` 对象。`ViewModel` 被销毁时内部会执行 `clear()` 方法，在 `clear()` 方法中遍历调用 `closeWithRuntimeException` 取消了 `viewModelScope` 的协程，实现流程非常清晰。相关代码如下：

```

1 // androidx.lifecycle.ViewModel
2
3 // clear() -> closeWithRuntimeException() -> coroutineContext.cancel()
4
5 private final Map<String, Object> mBagOfTags = new HashMap<>();
6
7 <T> T getTag(String key) {
8     synchronized (mBagOfTags) {
9         return (T) mBagOfTags.get(key);
10    }
11 }
12
13 <T> T setTagIfAbsent(String key, T newValue) {
14     T previous;
15     synchronized (mBagOfTags) {
16         previous = (T) mBagOfTags.get(key);
17         if (previous == null) {
18             mBagOfTags.put(key, newValue);
19         }
20     }
21     T result = previous == null ? newValue : previous;
22     if (mCleared) {
23         closeWithRuntimeException(result);
24     }
25     return result;
26 }
27
28 @MainThread
29 final void clear() {
30     mCleared = true;
31     if (mBagOfTags != null) {
32         for (Object value : mBagOfTags.values()) {
33             closeWithRuntimeException(value);
34         }
35     }
36 }

```


```
35     }
36     onCleared();
37 }
38
39 private static void closeWithRuntimeException(Object obj) {
40     if (obj instanceof Closeable) {
41         try {
42             ((Closeable) obj).close();
43         } catch (IOException e) {
44             throw new RuntimeException(e);
45         }
46     }
47 }
```

结论

如果你也正在使用 MVVM 和协程，非常推荐在 ViewModel 中使用 `viewModelScope` 方式。不仅简化 ViewModel 代码，而且还能管理协程生命周期。

 8人点赞 > 

 Android 



simpleeeeeee Android 开发

内推阿里、虎牙、bigo、yy

ba...
总资产9 共写了5419字 获得356个赞 共177个粉丝

[关注](#)