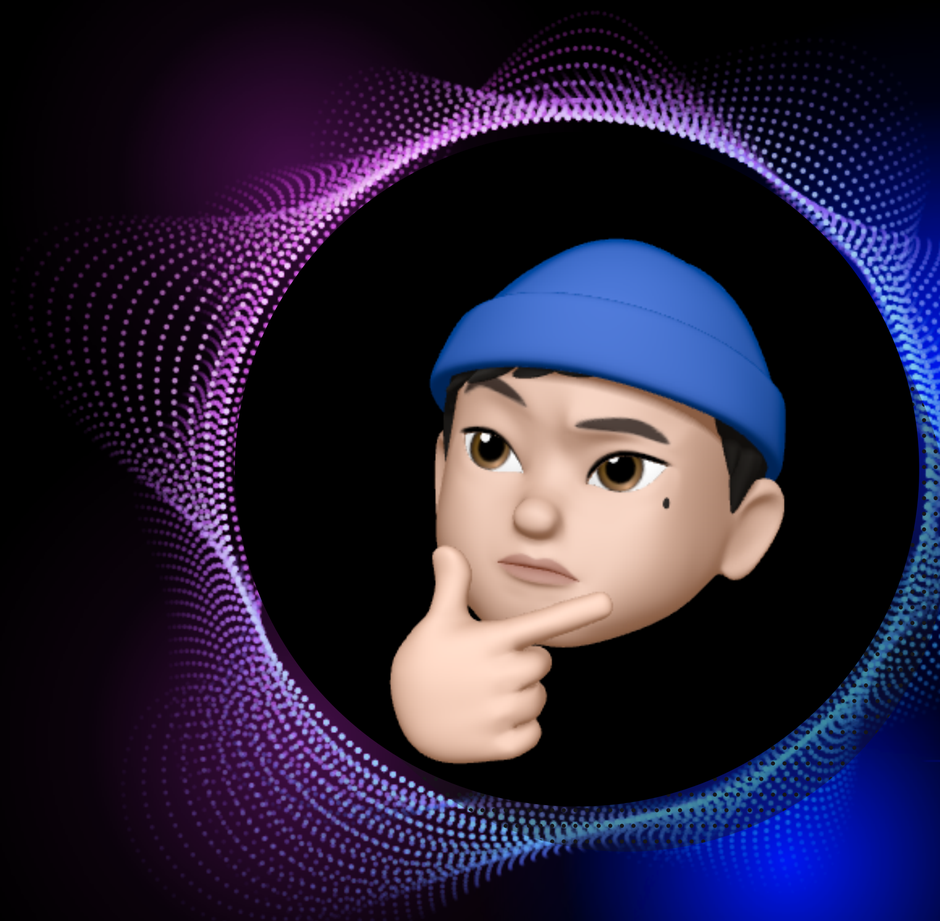
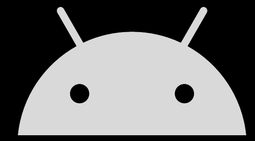
2022
찰스의 안드로이드
컨퍼런스

Test Code in Android



Speaker | 안홍범

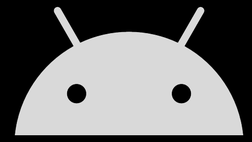
Test Code in Android



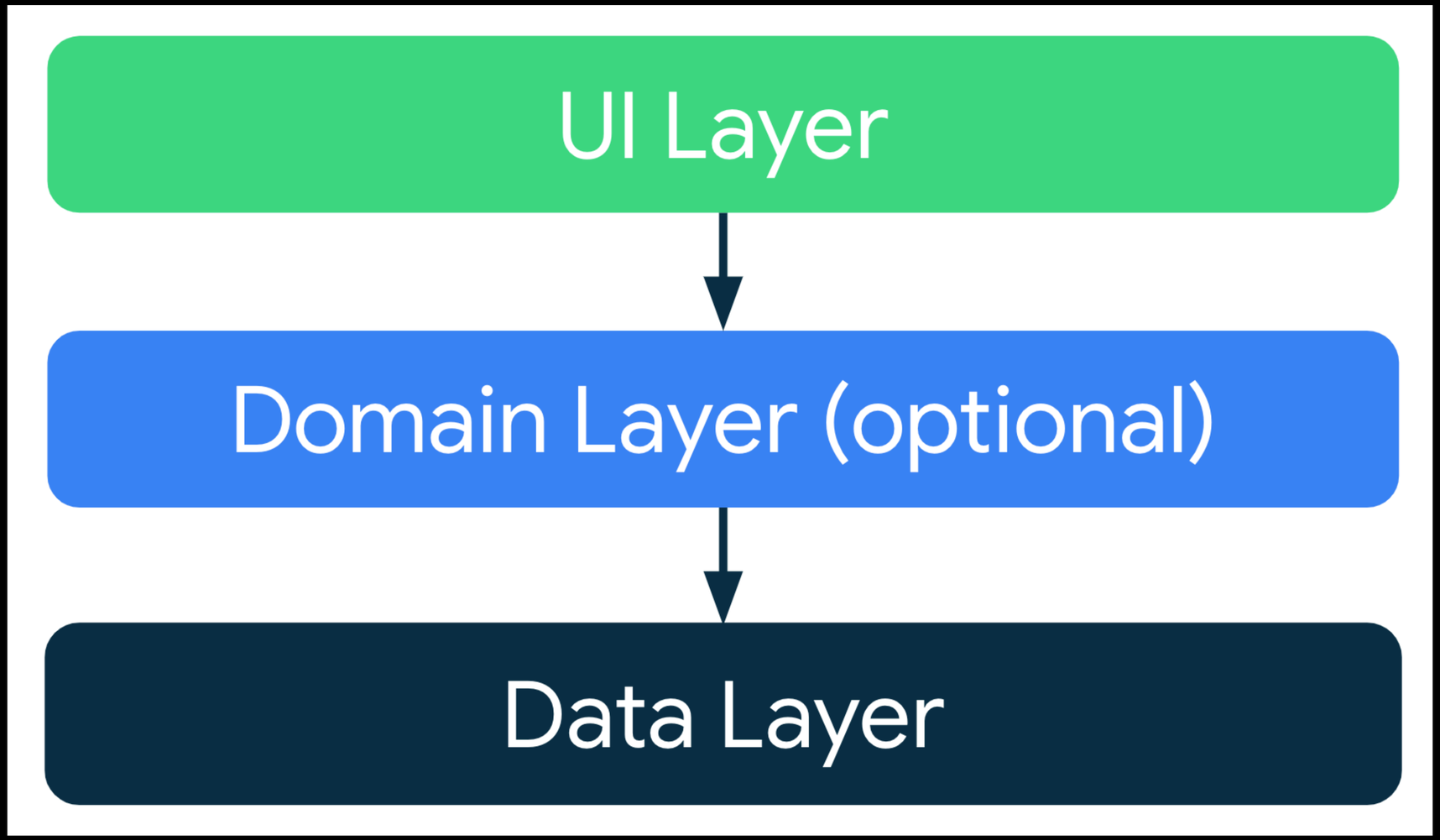
2022
찰스의 안드로이드
컨퍼런스

Testable Architecture
Android의 테스트 종류
무엇을 테스트 해야할까?
어떻게 테스트 코드를 작성할 수 있을까?
Etc

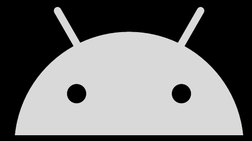
Testable Architecture



2022
찰스의 안드로이드
컨퍼런스

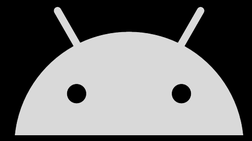


https://developer.android.com/topic/libraries/architecture/images/mad-arch-overview.png



2022
찰스의 안드로이드
컨퍼런스

Android의 테스트 종류



2022
찰스의 안드로이드
컨퍼런스

프로젝트 공간

Unit 테스트
Integration 테스트

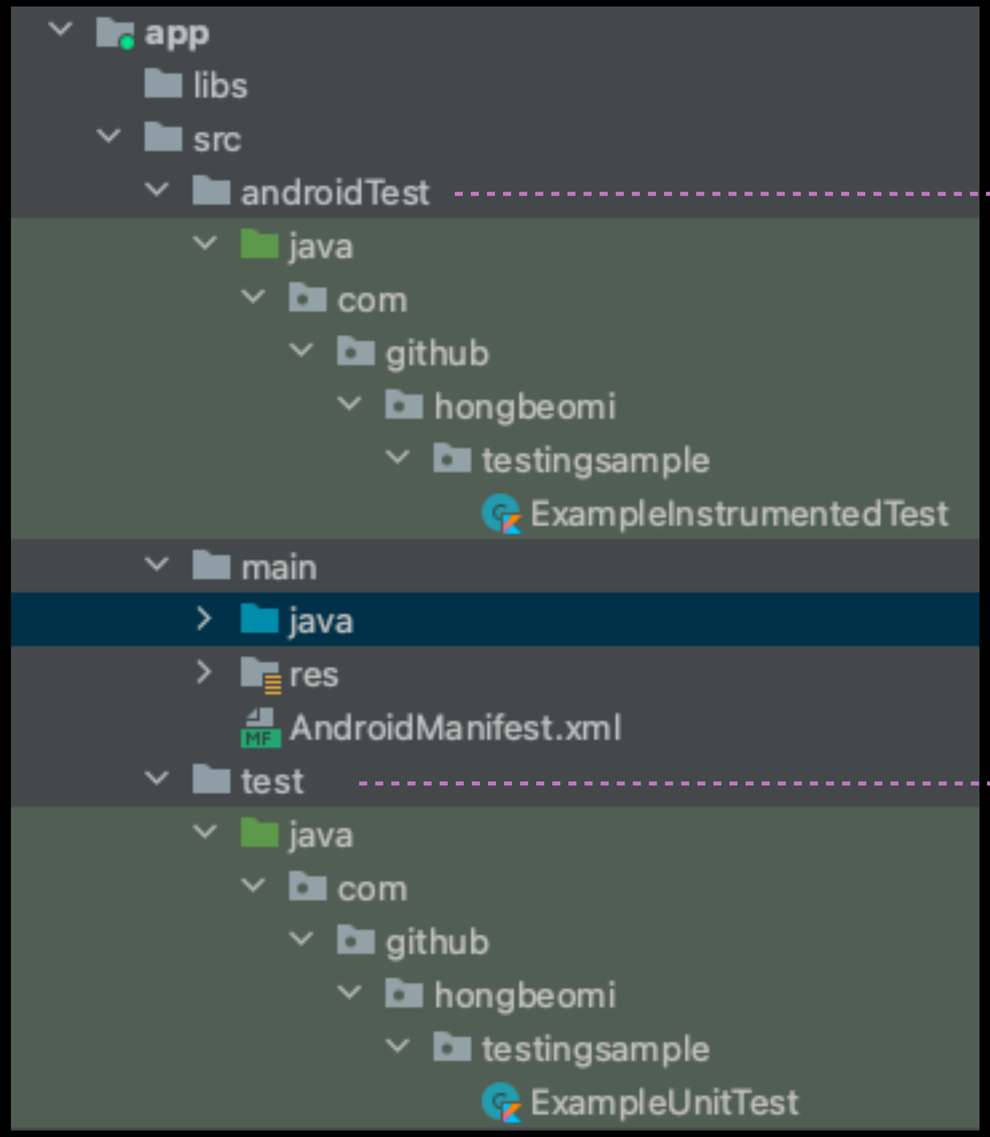
에뮬레이터

실기기

로컬 테스트

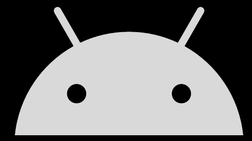
계측 테스트

- End-to-End 테스트
- Integration 테스트



Instrument test

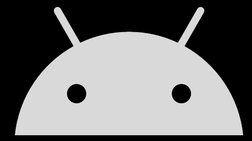
Local test



2022
찰스의 안드로이드
컨퍼런스

무엇을 테스트 해야할까?

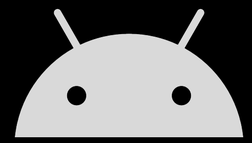
✅ Best Practice - Unit Test



2022
찰스의 안드로이드
컨퍼런스

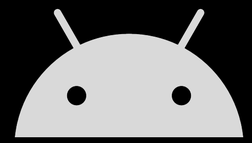
- Repository
- ViewModel 혹은 Presenter
- Domain 레이어
- Util 클래스

❌ Best Practice - 피해야할 Unit Test



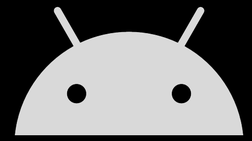
- 코드가 아닌 프레임워크 또는 라이브러리의 올바른 작동
- Activity, Fragment 혹은 Service 같은 Entry point

✅ Best Practice - UI 테스트



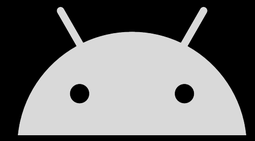
- 화면의 UI (클릭, 텍스트 입력 등등…)
- UI Flow / Navigation

✅ Best Practice - 기타



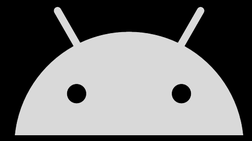
2022
찰스의 안드로이드
컨퍼런스

- 성능 테스트
  - https://developer.android.com/topic/performance
- Monkey 테스트
  - https://developer.android.com/studio/test/other-testing-tools/monkey
- 기타 (접근성, 스크린샷, 호환성 등등…)

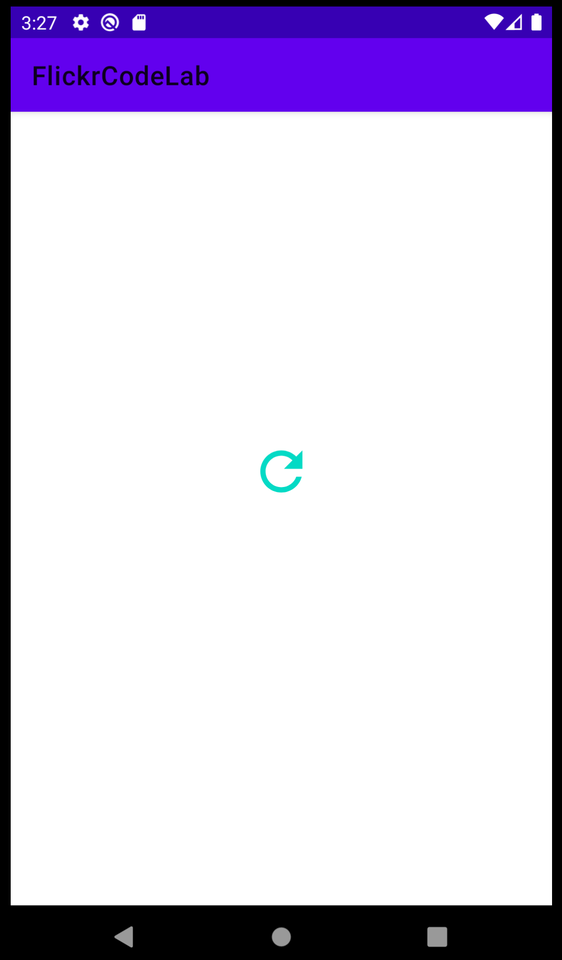


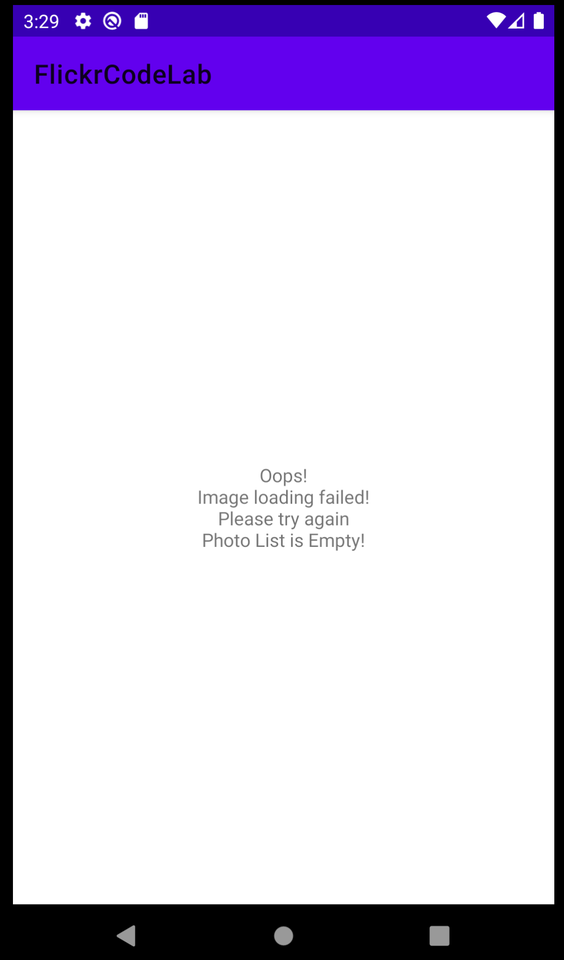
어떻게 테스트 코드를 작성할 수 있을까?

https://github.com/hongbeomi/AndroidCodeLab

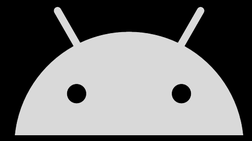


2022
찰스의 안드로이드
컨퍼런스





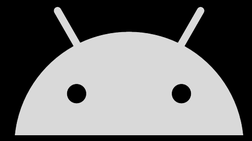
Unit Test 준비



2022
찰스의 안드로이드
컨퍼런스

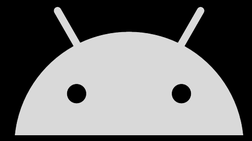
```
testImplementation "junit:junit:$junitVersion"
testImplementation "org.jetbrains.kotlinx:kotlinx-coroutines-test:$coroutinesVersion"
```

✅ Best Practice - Unit Test



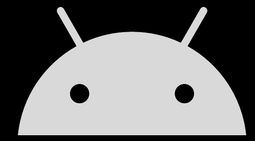
2022
찰스의 안드로이드
컨퍼런스

- Repository
- ViewModel / Presenter
- Domain 레이어
- Util 클래스



2022
찰스의 안드로이드
컨퍼런스

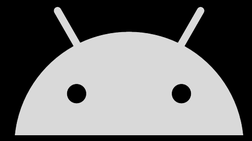
```
class DefaultPhotoListRepository(
    private val localPhotoListDataSource: PhotoListLocalDataSource,
    private val remotePhotoListDataSource: PhotoListRemoteDataSource,
    ...
): PhotoListRepository {

    override suspend fun getAllPhotoList(isForceUpdate: Boolean): Flow<List<Photo>> {
        withContext(...) {
            if (isForceUpdate) {
                refreshPhotoList()
            }
            localPhotoListDataSource.getSearchPhotoList()
        }
    }

}
```



2022
찰스의 안드로이드
컨퍼런스

```
class DefaultPhotoListRepository(
    private val localPhotoListDataSource: PhotoListLocalDataSource,
    private val remotePhotoListDataSource: PhotoListRemoteDataSource,
    ...
): PhotoListRepository {

    override suspend fun getAllPhotoList(isForceUpdate: Boolean): Flow<List<Photo>> {
        withContext(...) {
            if (isForceUpdate) {
                refreshPhotoList()
            }
            localPhotoListDataSource.getSearchPhotoList()
        }
    }

}
```

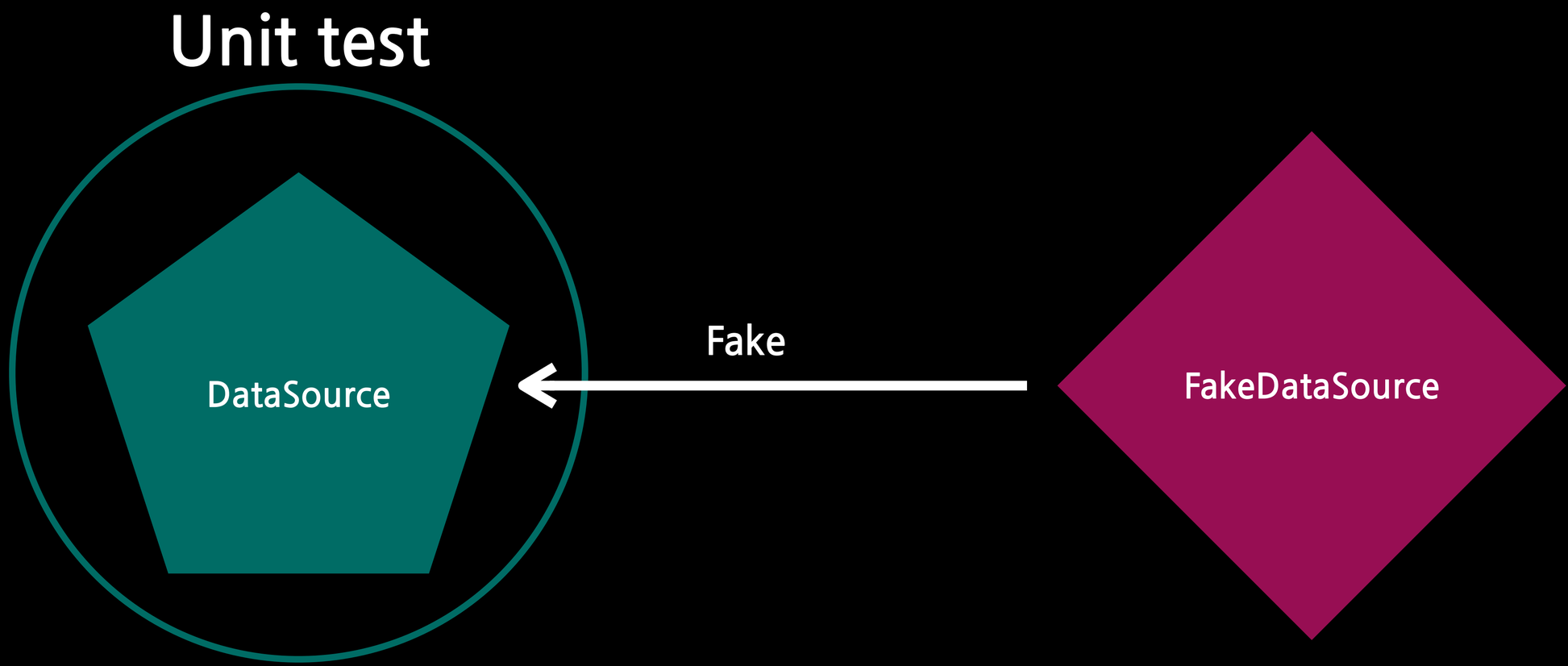
Test Double

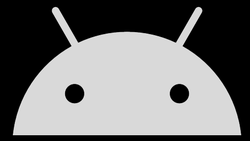


2022
찰스의 안드로이드
컨퍼런스

Fake	클래스에 작동에 대한 구현이 존재하지만 프로덕션 코드에는 적합하지 않은 방법으로 구현된 테스트 더블 - in memory DB
Mock	임의의 기대값을 넣어주는 대로 동작하는 테스트 더블. 일반적으로 라이브러리로 많이 사용됨 - 특정 객체의 메소드가 호출되었는지 확인
Stub	상호 작용에 대한 기대는 할 수 없지만 동작을 강제할 수 있는 테스트 더블
Dummy	파라미터 전달 같은, 전달되지만 사용되지 않는 테스트 더블 - 콜백으로 전달되는 빈 함수
Spy	Mock 객체와 유사한 실제 객체에 대한 래퍼. 사용하고자 하는 메소드들만 mocking하고 나머지는 원본 그대로 사용함.
Shadow	Robolectic에서 제공되는 fake

https://martinfowler.com/articles/mocksArentStubs.html
https://developer.android.com/training/testing/fundamentals/test-doubles

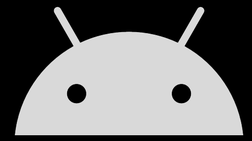

Unit test
DataSource
Fake
FakeDataSource



2022
찰스의 안드로이드
컨퍼런스

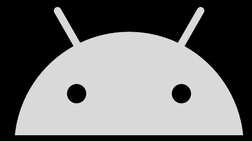
```
interface PhotoListDataSource

class PhotoListLocalDataSource(...): PhotoListDataSource
class PhotoListRemoteDataSource(...): PhotoListDataSource

class DefaultPhotoListRepository(
    private val localPhotoListDataSource: PhotoListDataSource,
    private val remotePhotoListDataSource: PhotoListDataSource,
    ...
): PhotoListRepository
```

Fake



2022
찰스의 안드로이드
컨퍼런스

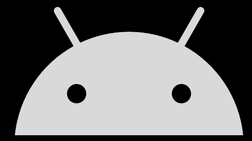
```
class FakePhotoListDataSource(
    private var photoList: MutableList<Photo> = mutableListOf()
): PhotoListDataSource {
    ...
}
```



2022
찰스의 안드로이드
컨퍼런스

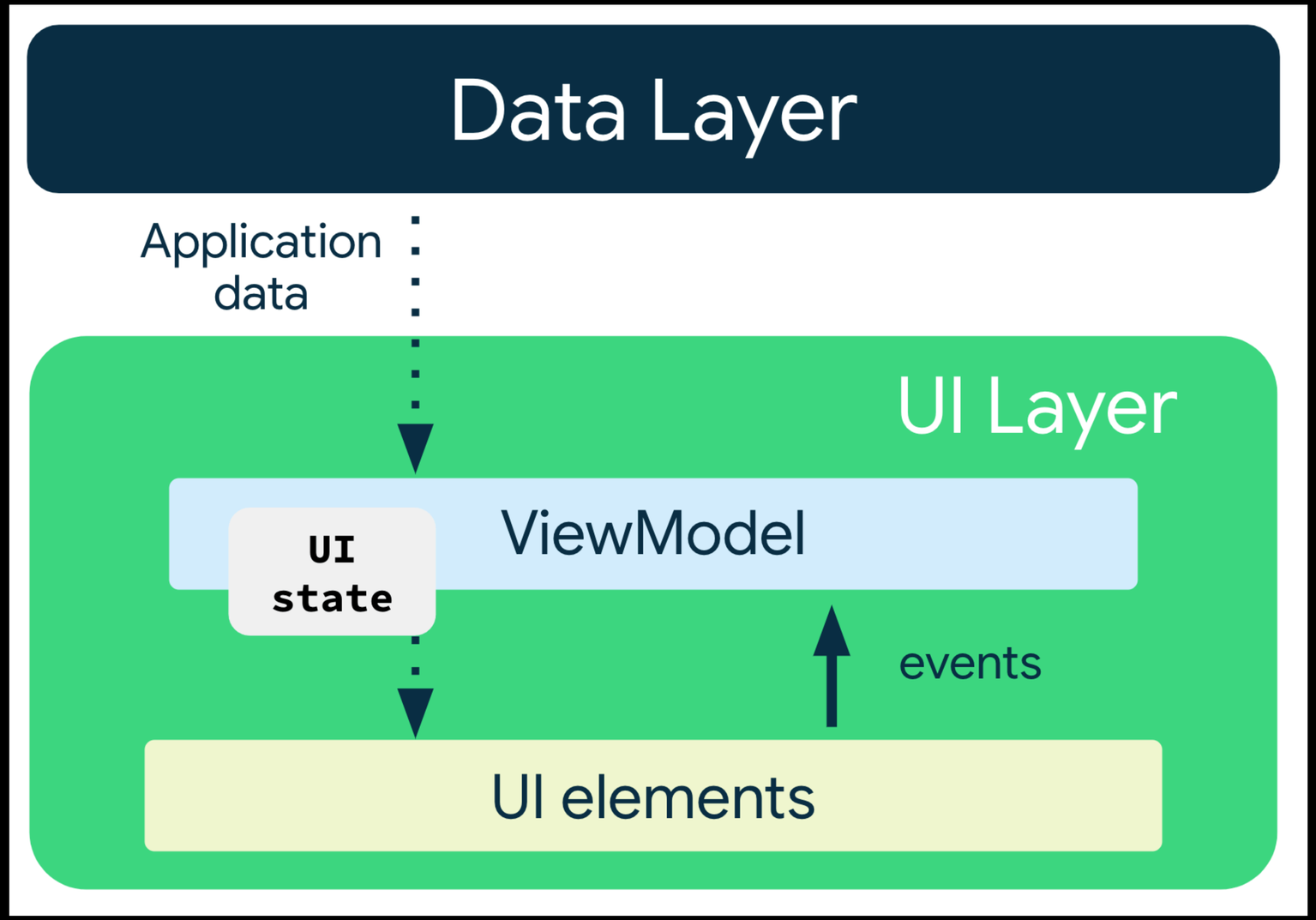
```
class DefaultPhotoListRepositoryTest {

    ...
    private lateinit var defaultPhotoListRepository: DefaultPhotoListRepository
    private lateinit var localPhotoListDataSource: FakePhotoListDataSource
    private lateinit var remotePhotoListDataSource: FakePhotoListDataSource

    @Before
    fun setUp() {
        localPhotoListDataSource = FakePhotoListDataSource(localPhotoList)
        remotePhotoListDataSource = FakePhotoListDataSource(remotePhotoList)

        defaultPhotoListRepository = DefaultPhotoListRepository(
            localPhotoListDataSource,
            remotePhotoListDataSource,
            ...
        )
    }
    ...

}
```

✅ Best Practice - Unit Test

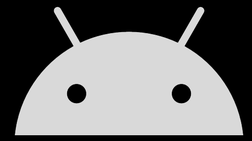


2022
찰스의 안드로이드
컨퍼런스

- Repository
- ViewModel / Presenter
- Domain 레이어
- Util 클래스

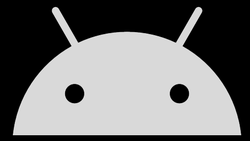


https://developer.android.com/topic/libraries/architecture/images/mad-arch-ui-udf.png



2022
찰스의 안드로이드
컨퍼런스

```
sealed class MainUiState {
    data class Success(val photoList: List<Photo> = emptyList()): MainUiState()
    object Loading: MainUiState()
    data class Error(val exception: Throwable): MainUiState()
    ...
}

sealed class MainUiAction {
    data class Refresh(val isForceUpdate: Boolean): MainUiAction()
    object LoadMore: MainUiAction()
}
```



2022
찰스의 안드로이드
컨퍼런스

```
class MainViewModel(
    private val photoListRepository: PhotoListRepository
): ViewModel() {

    ...
    private val _mainUiState: MutableStateFlow<MainUiState> = MutableStateFlow(MainUiState.Loading)
    val mainUiState: StateFlow<MainUiState> = _mainUiState.asStateFlow()

    private val mainUiAction: MutableSharedFlow<MainUiAction> = MutableSharedFlow()
    ...

    private fun refresh(isForceUpdate: Boolean) {
        _mainUiState.update { MainUiState.Loading }

        viewModelScope.launch(...) {
            ...
            photoListRepository
                .getAllPhotoList(isForceUpdate)
                .distinctUntilChanged()
                .collect {
                    _mainUiState.update { MainUiState.Success(it) }
                }
            ...
        }
    }

}
```

2022
찰스의 안드로이드
컨퍼런스

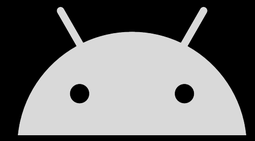
```
class MainViewModel(
    private val photoListRepository: PhotoListRepository
): ViewModel() {

    ...
    private val _mainUiState: MutableStateFlow<MainUiState> = MutableStateFlow(MainUiState.Loading)
    val mainUiState: StateFlow<MainUiState> = _mainUiState.asStateFlow()

    private val mainUiAction: MutableSharedFlow<MainUiAction> = MutableSharedFlow()
    ...

    private fun refresh(isForceUpdate: Boolean) {
        _mainUiState.update { MainUiState.Loading }

        viewModelScope.launch(...) {
            ...
            photoListRepository
                .getAllPhotoList(isForceUpdate)
                .distinctUntilChanged()
                .collect {
                    _mainUiState.update { MainUiState.Success(it) }
                }
            ...
        }
    }

}
```

2022
찰스의 안드로이드
컨퍼런스

```
class MainViewModel(
    private val photoListRepository: PhotoListRepository
): ViewModel() {

    ...
    private val _mainUiState: MutableStateFlow<MainUiState> = MutableStateFlow(MainUiState.Loading)
    val mainUiState: StateFlow<MainUiState> = _mainUiState.asStateFlow()

    private val mainUiAction: MutableSharedFlow<MainUiAction> = MutableSharedFlow()
    ...

    private fun refresh(isForceUpdate: Boolean) {
        _mainUiState.update { MainUiState.Loading }

        viewModelScope.launch(...) {
            ...
            photoListRepository
                .getAllPhotoList(isForceUpdate)
                .distinctUntilChanged()
                .collect {
                    _mainUiState.update { MainUiState.Success(it) }
                }
            ...
        }
    }

}
```

2022
찰스의 안드로이드
컨퍼런스

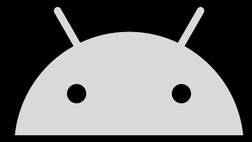
```
class MainViewModelTest {

    private lateinit var viewModel: MainViewModel
    private lateinit var repository: FakePhotoListRepository

    @Before
    fun setUp() {
        repository = FakePhotoListRepository()
        viewModel = MainViewModel(repository)
    }


    @Test
    fun givenNewPhotoList_WhenRefreshViewModel_ThenLoadingAndRemotePhotoList() = runTest {
        ...
        repository.addPhotoList(remoteData)

        viewModel.setUiAction(MainViewModel.MainUiAction.Refresh(true))

        val result = viewModel.mainUiState.take(2).toList()
        val loading = result.first() as MainViewModel.MainUiState.Loading
        val success = result.last() as MainViewModel.MainUiState.Success

        assertThat(loading,  IsEqual(MainViewModel.MainUiState.Loading))
        assertThat(success.photoList, IsEqual(remoteData))
    }

}
```



2022
찰스의 안드로이드
컨퍼런스

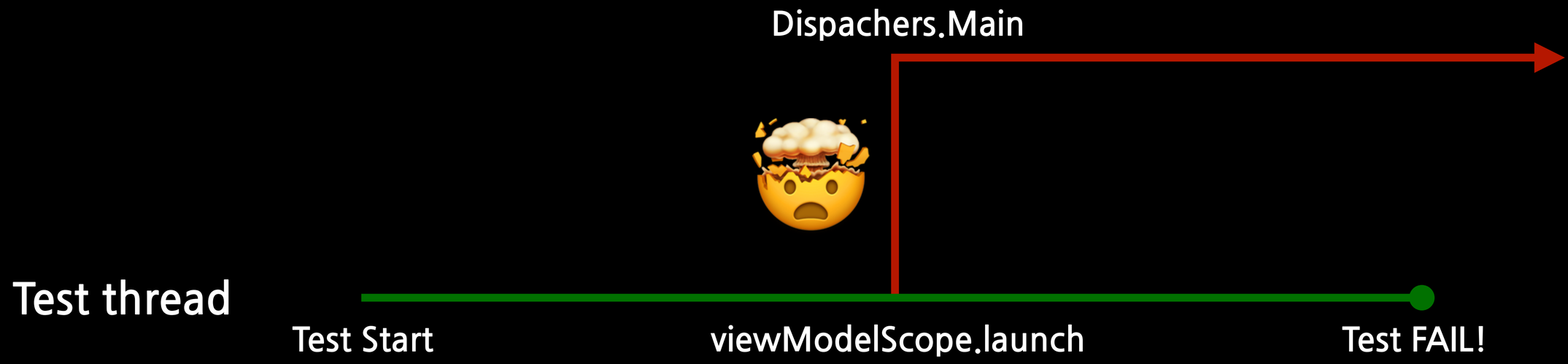
하지만..?

```
Module with the Main dispatcher had failed to initialize. For tests Dispatchers.setMain from kotlinx-coroutines-test module can be used
java.lang.IllegalStateException Create breakpoint : Module with the Main dispatcher had failed to initialize. For tests Dispatchers.setMain from kotlinx-coroutines-test module can be used
```



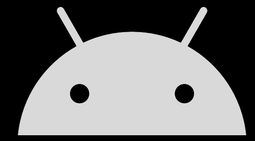
2022
찰스의 안드로이드
컨퍼런스

```
public val ViewModel.viewModelScope: CoroutineScope
    get() {
        val scope: CoroutineScope? = this.getTag(JOB_KEY)
        if (scope ≠ null) {
            return scope
        }
        return setTagIfAbsent(
            JOB_KEY,
            CloseableCoroutineScope(SupervisorJob() + Dispatchers.Main.immediate)
        )
    }
```



```kotlin
class TestCoroutineRule(
    val dispatcher: TestDispatcher = StandardTestDispatcher()
) : TestWatcher() {

    override fun starting(description: Description?) {
        Dispatchers.setMain(dispatcher)
    }

    override fun finished(description: Description?) {
        Dispatchers.resetMain()
    }

}
```

https://github.com/Kotlin/kotlinx.coroutines/blob/master/kotlinx-coroutines-test/MIGRATION.md



2022
찰스의 안드로이드
컨퍼런스

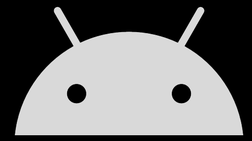
Test Start

viewModelScope.launch

Test Assertion

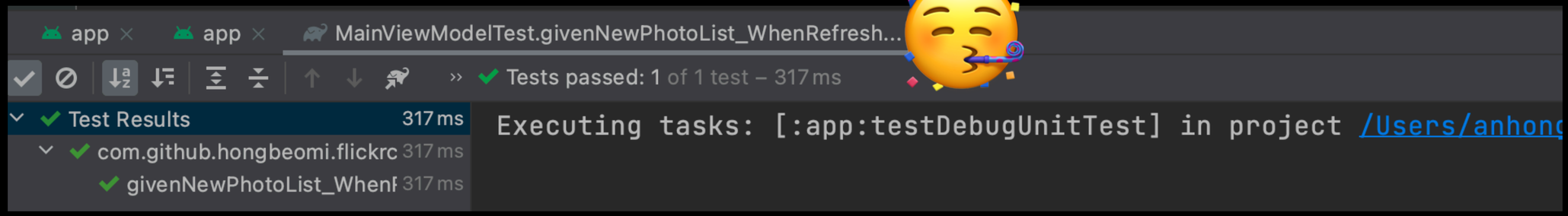
Test thread

Test Finished

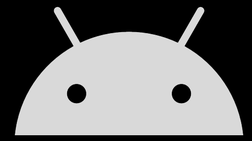


2022
찰스의 안드로이드
컨퍼런스

```kotlin
class MainViewModelTest {

    @get:Rule
    val coroutineRule = TestCoroutineRule()
    ...

     @Test
    fun givenNewPhotoList_WhenRefreshViewModel_ThenLoadingAndRemotePhotoList() = runTest {
        ...
    }

}
```



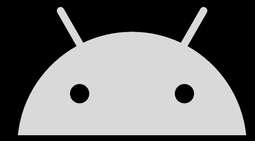
✅ Best Practice - UI 테스트



2022
찰스의 안드로이드
컨퍼런스

- 화면 UI (클릭, 텍스트 입력 등등…)
- Flow / Navigation

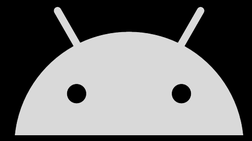
Ui Test 도구



2022
찰스의 안드로이드
컨퍼런스

- Espresso 14+
- Jetpack Compose 21+
- UI Automator 18+
- Robolectric 16+

Ui Test 준비 - with Espresso

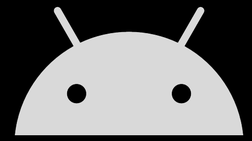


2022
찰스의 안드로이드
컨퍼런스

```
androidTestImplementation "androidx.test:runner:$androidXTestRunnerVersion"
androidTestImplementation "androidx.test.ext:junit:$androidXTestExtVersion"
androidTestImplementation "androidx.test.espresso:espresso-core:$espressoVersion"

// for Another Views Matchers - RecyclerView
androidTestImplementation "androidx.test.espresso:espresso-contrib:$espressoVersion"
androidTestImplementation "org.jetbrains.kotlinx:kotlinx-coroutines-test:$coroutinesVersion"
```

```
android {
    ...
    defaultConfig {
        ...
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }
    ...
}
```

Ui Test 준비



2022
찰스의 안드로이드
컨퍼런스

설정 -> 개발자 옵션의

Window Animation Scale, Transition Animation Scale, Animator Duration Scale 비활성화

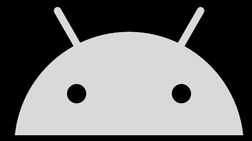
혹은

```
android {
    ...
    testOptions {
        animationsDisabled = true
    }
    ...
}
```

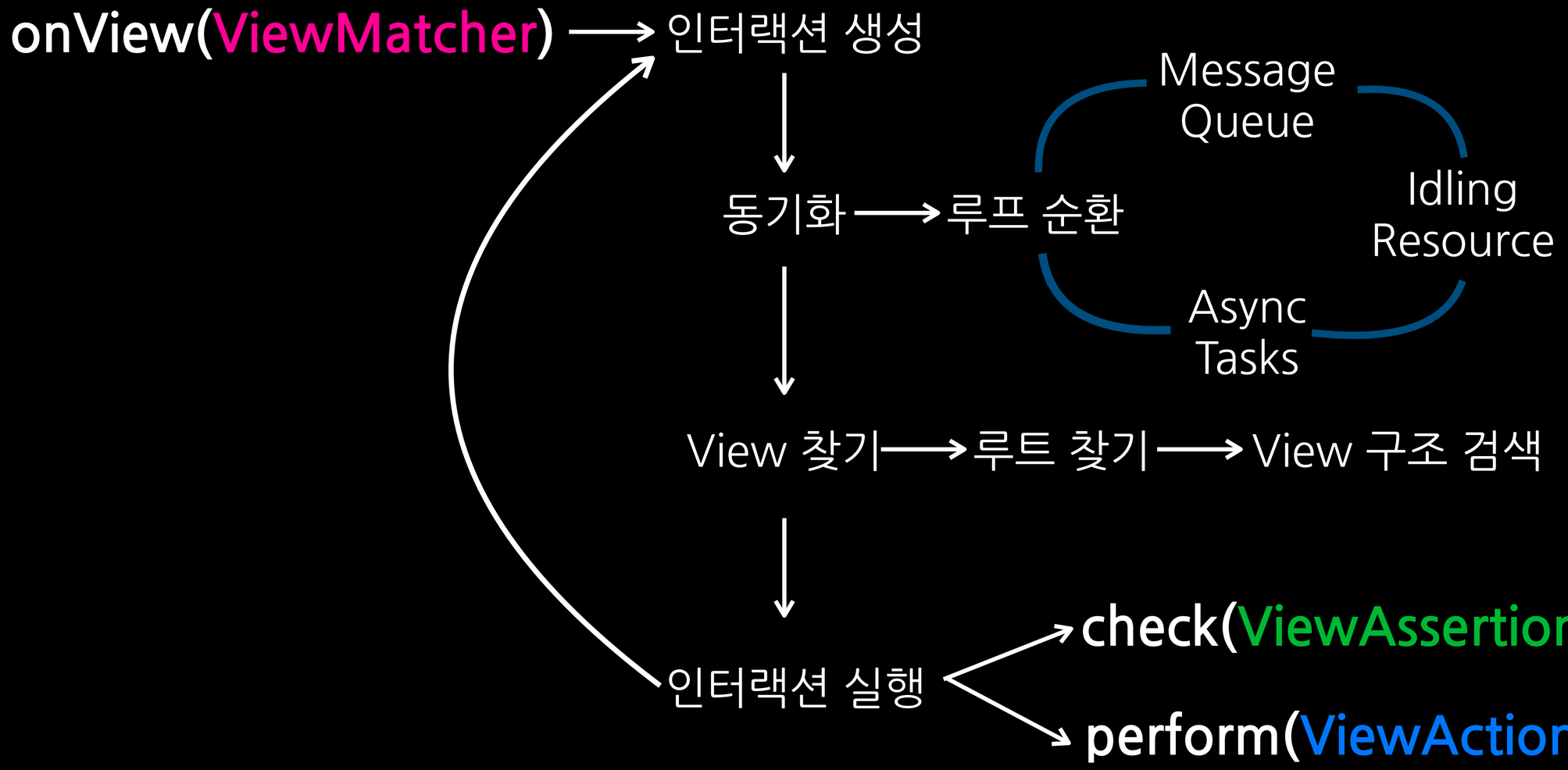
Espresso

```
onView(ViewMatcher)
  .perform(ViewAction)
  .check(ViewAssertion)
```

https://developer.android.com/training/testing/espresso

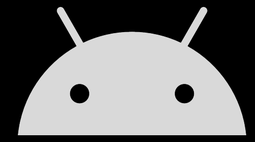


2022
찰스의 안드로이드
컨퍼런스



https://www.youtube.com/watch?v=pK7W5npkhho

```kotlin
@RunWith(AndroidJUnit4::class)
class MainActivityTest {

    private lateinit var activityScenario: ActivityScenario<MainActivity>
    private lateinit var repository: FakePhotoListRepository
    ...

     @Test
    fun givenRemotePhotoList_WhenStartMainPage_ThenShowRemotePhotoList() {
        // given
        repository.addPhotoList(Fixtures.photo())

        // when
        activityScenario = ActivityScenario.launch(MainActivity::class.java)
        dataBindingIdlingResource.monitorActivity(activityScenario)

        // then
        onView(withId(R.id.progressBar_main)).check(matches(isDisplayed()))
        onView(withId(R.id.recyclerView_main)).check(matches(isDisplayed()))
    }

}
```



2022
찰스의 안드로이드
컨퍼런스

Run: app × givenRemotePhotoLi...() ×

Status 1 failed 1 tests, 4 s 792 ms

Filter tests:

Tests	Duration	Tester
Test Results	1 s	0/1
MainActivityTest	1 s	0/1
givenRemotePhotoList_WhenStartMainPage_T	1 s	

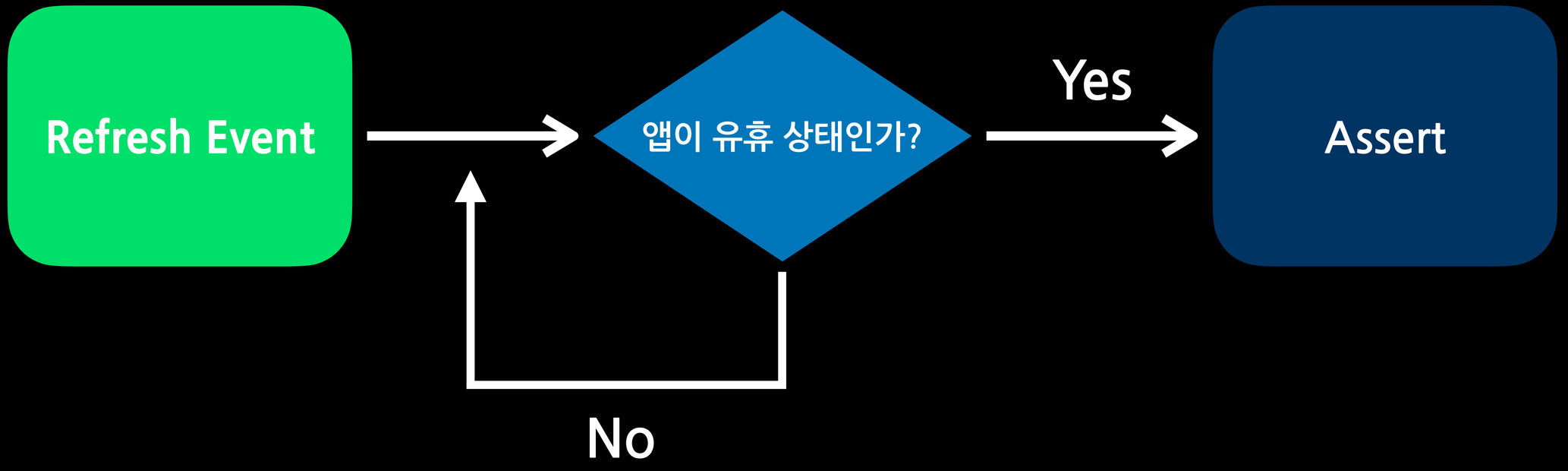
com.github.hongbeomi.flickrcodelab.ui.MainActivityTest.givenRemotePhotoList_WhenStartMainPage_ThenShowRemotePhotoList

androidx.test.espresso.base.DefaultFailureHandler$AssertionFailedWithCauseError: '(view has effective visibility <VISIBLE> and view.getGlobalVisibleRect() to return non-empty rectangle)' doesn't match the selected view.
Failed on Tester

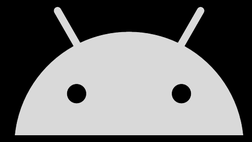
Logs Device Info

```
androidx.test.espresso.base.DefaultFailureHandler$AssertionFailedWithCauseError: '(view has effective visibility <V
Expected: (view has effective visibility <VISIBLE> and view.getGlobalVisibleRect() to return non-empty rectangle)
Got: view.getVisibility() was <GONE>
View Details: RecyclerView{id=2131231064, res-name=recyclerView_main, visibility=GONE, width=1080, height=1584, has

at dalvik.system.VMStack.getThreadStackTrace(Native Method)
at java.lang.Thread.getStackTrace(Thread.java:1720)
at androidx.test.espresso.base.DefaultFailureHandler.getUserFriendlyError(DefaultFailureHandler.java:12)
at androidx.test.espresso.base.DefaultFailureHandler.handle(DefaultFailureHandler.java:7)
at androidx.test.espresso.ViewInteraction.waitForAndHandleInteractionResults(ViewInteraction.java:5)
at androidx.test.espresso.ViewInteraction.check(ViewInteraction.java:12)
at com.github.hongbeomi.flickrcodelab.ui.MainActivityTest.givenRemotePhotoList_WhenStartMainPage_ThenShowRemotePhot
... 29 trimmed
Caused by: junit.framework.AssertionFailedError: '(view has effective visibility <VISIBLE> and view.getGlobalVisible
Expected: (view has effective visibility <VISIBLE> and view.getGlobalVisibleRect() to return non-empty rectangle)
Got: view.getVisibility() was <GONE>
View Details: RecyclerView{id=2131231064, res-name=recyclerView_main, visibility=GONE, width=1080, height=1584, has

at androidx.test.espresso.matcher.ViewMatchers.assertThat(ViewMatchers.java:16)
at androidx.test.espresso.assertion.ViewAssertions$MatchesViewAssertion.check(ViewAssertions.java:11)
```

Ui Test 동기화

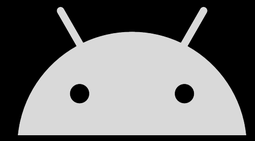


Sync!



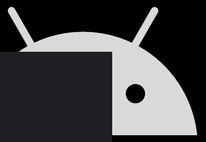
2022
찰스의 안드로이드
컨퍼런스

```
object EspressoIdlingResource {

    private const val RESOURCE = "GLOBAL"

    @JvmField
    val countingIdlingResource = CountingIdlingResource(RESOURCE)

    fun increment() {
        countingIdlingResource.increment()
    }

    fun decrement() {
        if (!countingIdlingResource.isIdleNow) {
            countingIdlingResource.decrement()
        }
    }
}
```



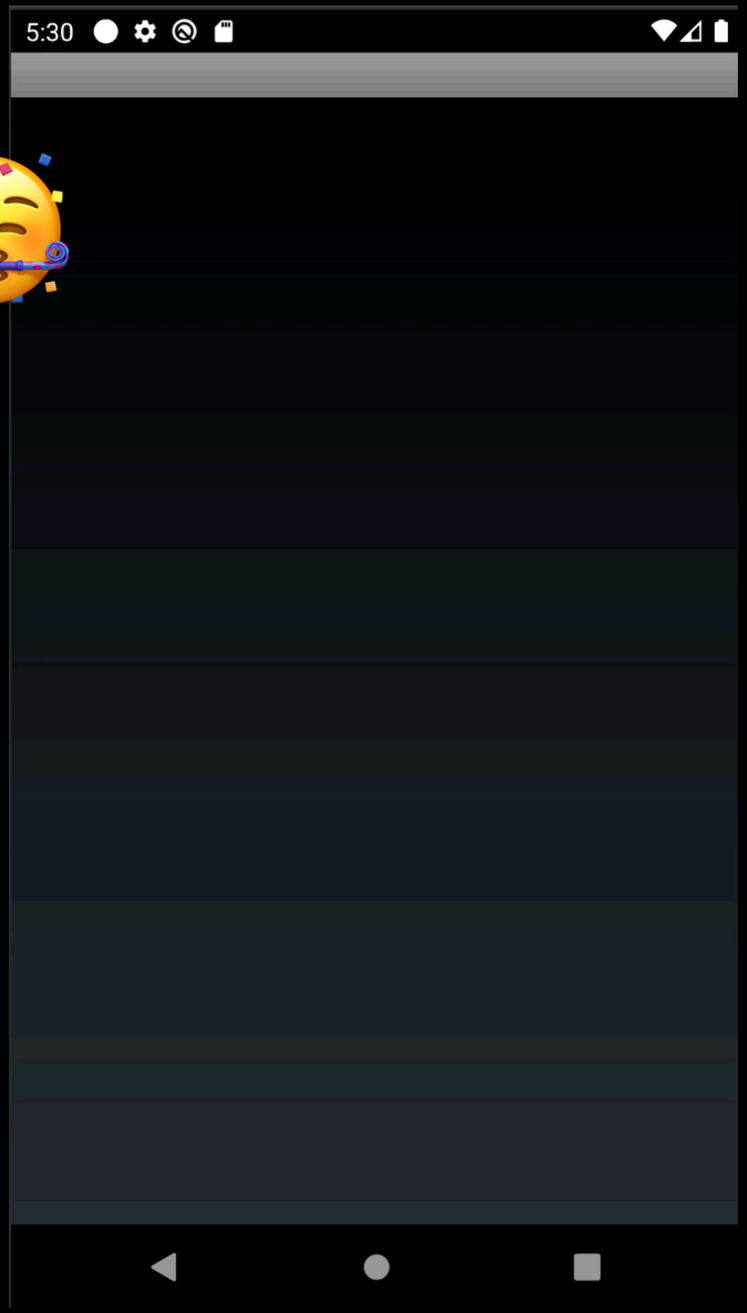
2022
찰스의 안드로이드
컨퍼런스

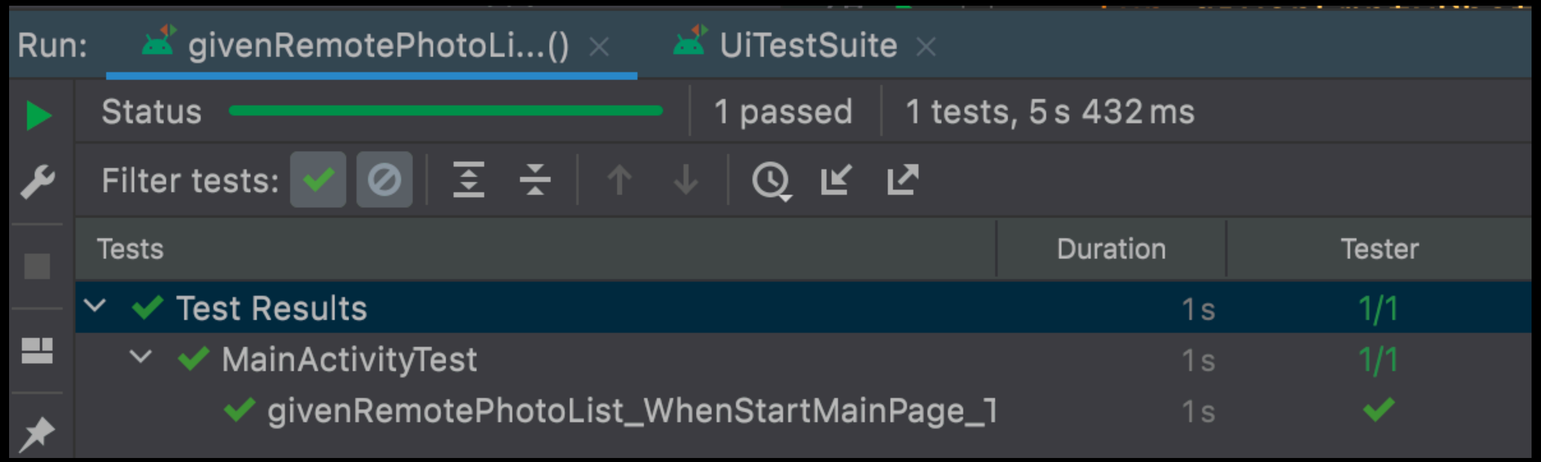
```
class FakePhotoListRepository : PhotoListRepository {

    ...
    override suspend fun getAllPhotoList(isForceUpdate: Boolean): Flow<List<Photo>> {
        wrapEspressoIdlingResource(isUseIdlingResource) {
            if (isForceUpdate) {
                refreshPhotoList()
            }
            return photoListLocalFlow
        }
    }
    ...

}
```



2022
찰스의 안드로이드
컨퍼런스

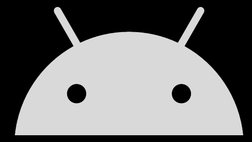
```
@RunWith(AndroidJUnit4::class)
class MainActivityTest {

    ...
    @Before
    fun setUp() {
        ...
        IdlingRegistry
            .getInstance()
            .register(EspressoIdlingResource.countingIdlingResource)
    }

    @After
    fun tearDown() {
        IdlingRegistry
            .getInstance()
            .unregister(EspressoIdlingResource.countingIdlingResource)
    }
    ...

}
```





2022
찰스의 안드로이드
컨퍼런스

Etc

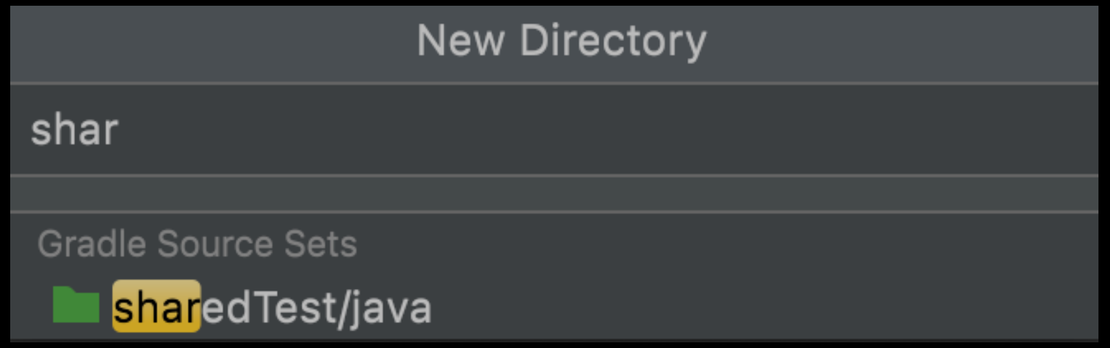


2022
찰스의 안드로이드
컨퍼런스

테스트 코드에서 공통적으로 사용하는 중복되는 코드가 있다면..?
Ex) FakeRepository, Mock Data…

Chipmunk 이전

```
sourceSets {
    androidTest.java.srcDirs += "src/sharedTest/java"
    test.java.srcDirs += "src/sharedTest/java"
}
```

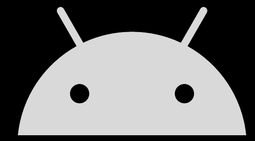


Chipmunk 이후

TestFixtures

https://developer.android.com/reference/tools/gradle-api/7.3/com/android/build/api/dsl/TestFixtures

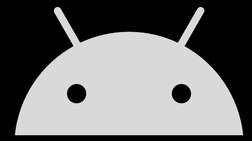
Etc



UI Test시 공유되는 상태로 인해 원인을 알 수 없는 이유로
테스트가 실패하고 이에 따라 후속 테스트가 모두 실패하는 경우

Android Test Orchestrator

Etc - Test Orchestrator



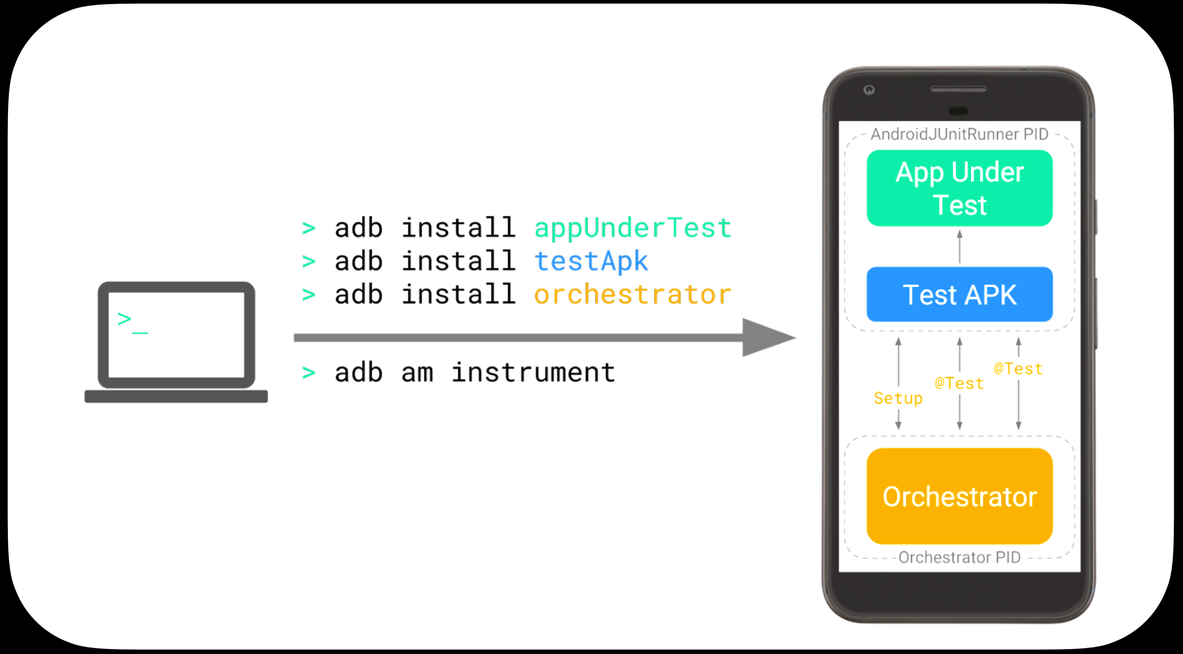
2022
찰스의 안드로이드
컨퍼런스

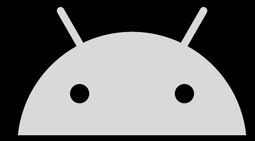
```
androidTestImplementation "androidx.test:runner:$androidXTestRunnerVersion"
androidTestUtil "androidx.test:orchestrator:$orchestratorVersion"
```

```
defaultConfig {
    testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    // 각 테스트 호출 후 pm clear 명령을 수행하여 테스트 사이에 공유되는 앱 상태가 없도록 한다.
    testInstrumentationRunnerArguments clearPackageData: 'true'
```

```
testOptions {
    execution 'ANDROIDX_TEST_ORCHESTRATOR'
    ...
}
```

https://developer.android.com/images/training/testing/android_test_orchestrator_flow_2x.png





2022
찰스의 안드로이드
컨퍼런스

감사합니다

- https://developer.android.com/training/testing/instrumented-tests/androidx-test-libraries/runner
- https://developer.android.com/training/testing/espresso/cheat-sheet
- https://developer.android.com/training/testing/instrumented-tests
- https://developer.android.com/training/testing/fundamentals
- https://developer.android.com/training/testing