



Workshop














Somkiat

Home









Somkiat Puisungnoen

Update Info
1


View Activity Log
10+

...


Timeline
About
Friends 3,138
Photos
More ▾



When did you work at Opendream?
×


...
22 Pending Items



Intro

Software Craftsmanship


Software Practitioner at สยามชำนาญกิจ พ.ศ. 2556



Agile Practitioner and Technical at SPRINT3r



Post


Photo/Video


Live Video


Life Event



What's on your mind?


Public ▾

Post



Somkiat Puisungnoen

15 mins · Bangkok ·


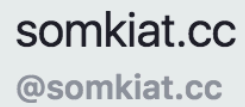
Java and Bigdata

...

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Android Testing

3



Photos



+ Add a Button

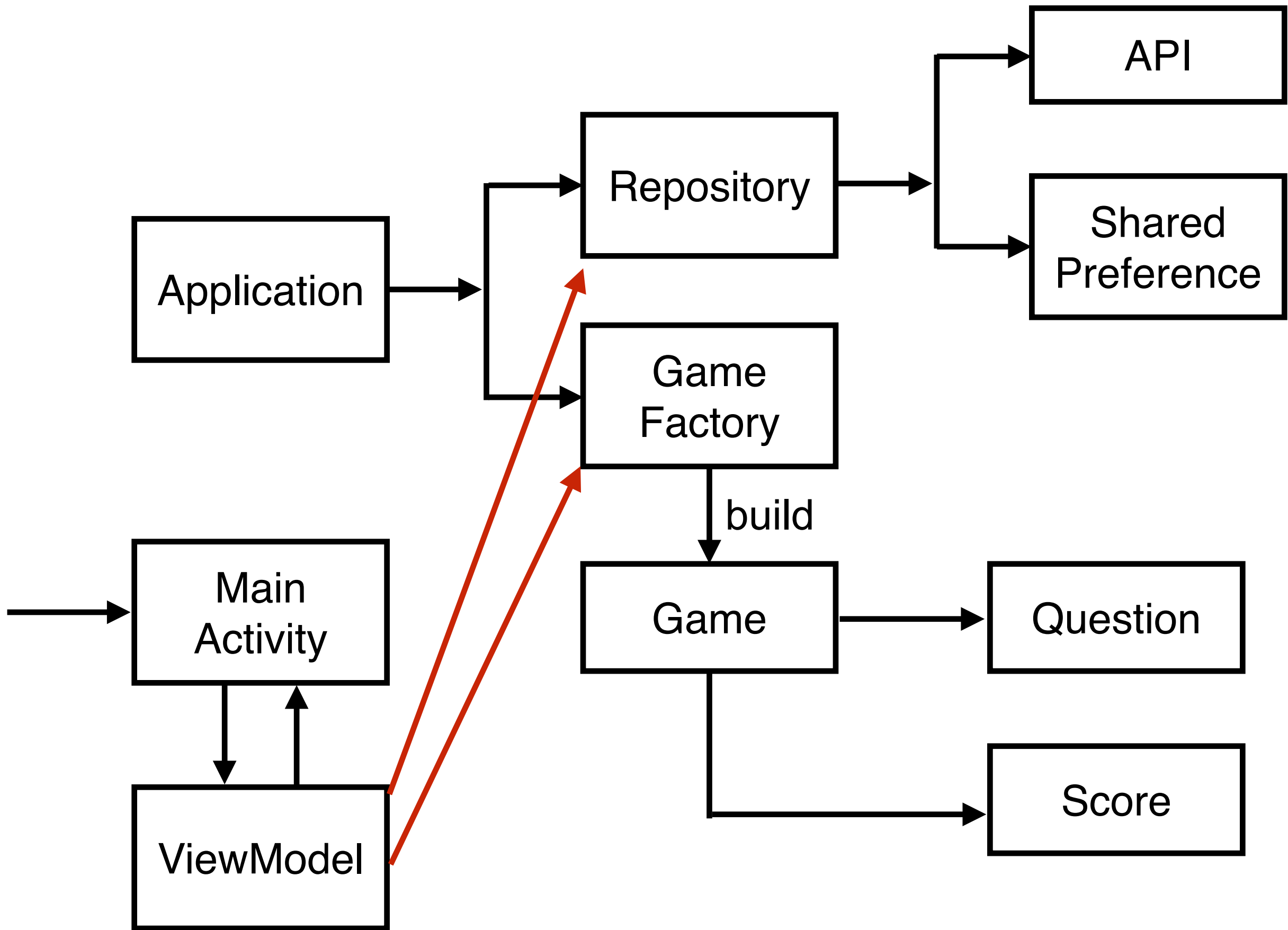
Workshop

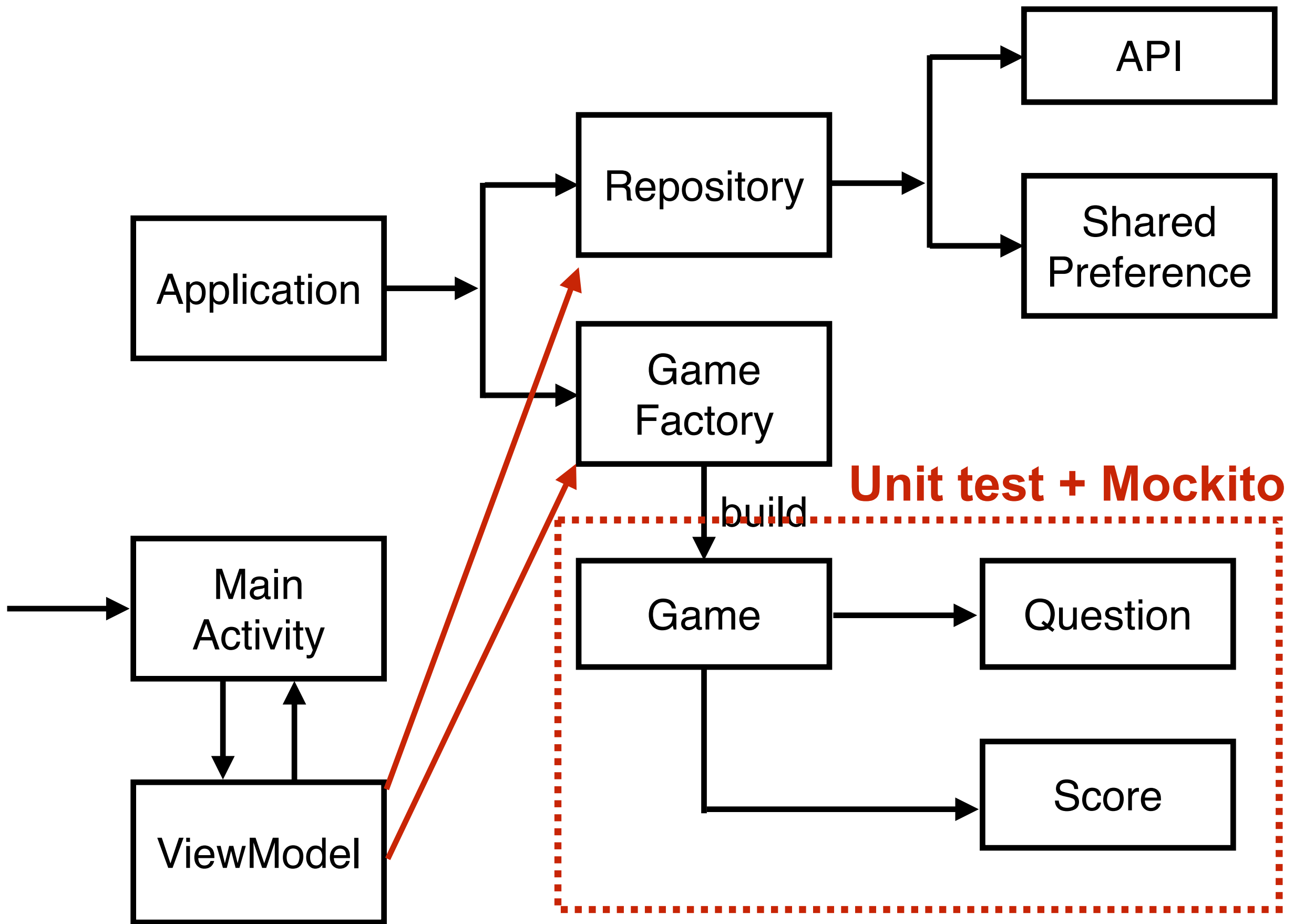


Android Testing

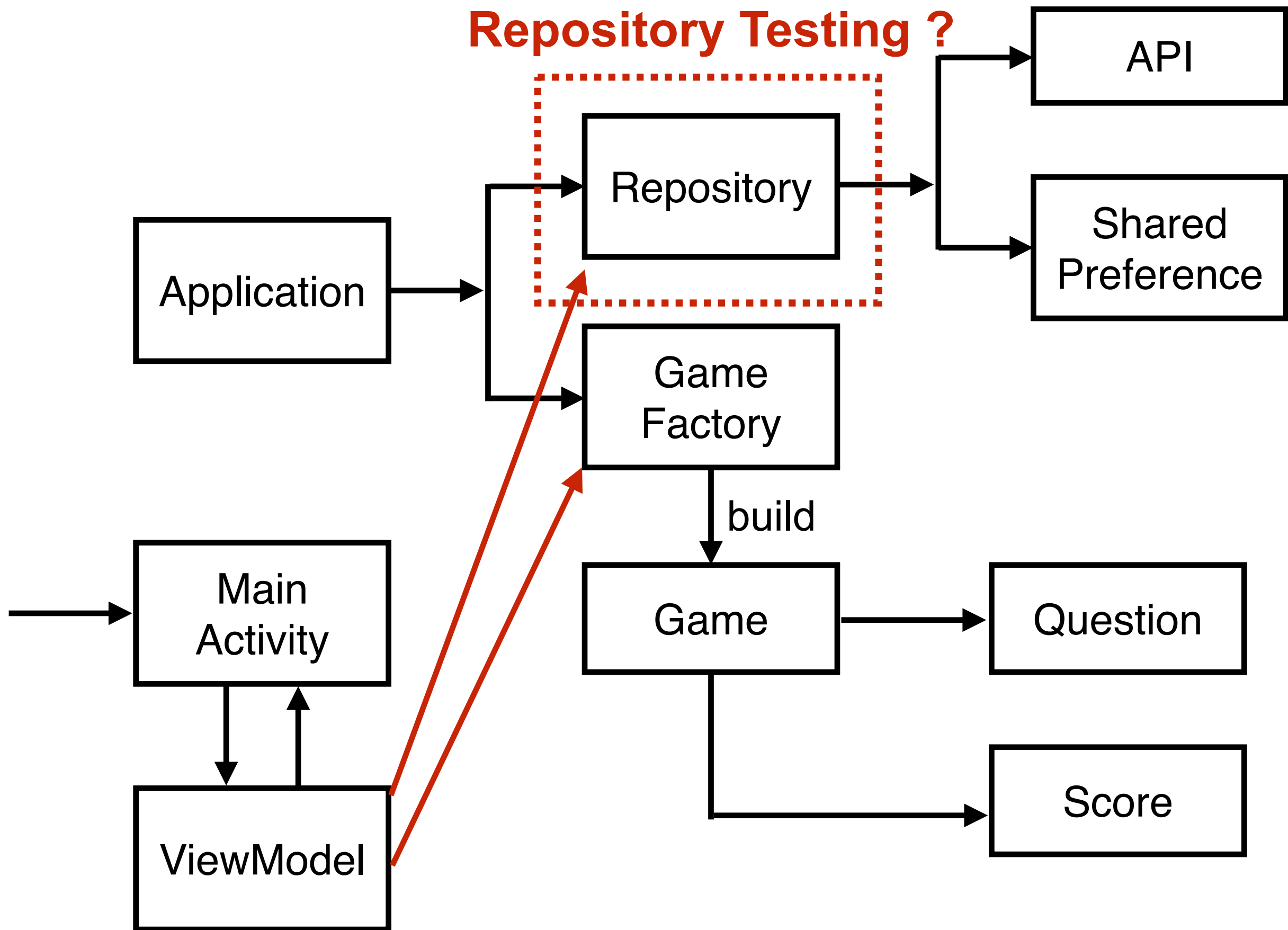
	JVM	Device
Non-UI	JVM unit test	Instrumentation unit test
UI	Robolectric	Instrumentation UI test
	/src/test	/src/androidTest



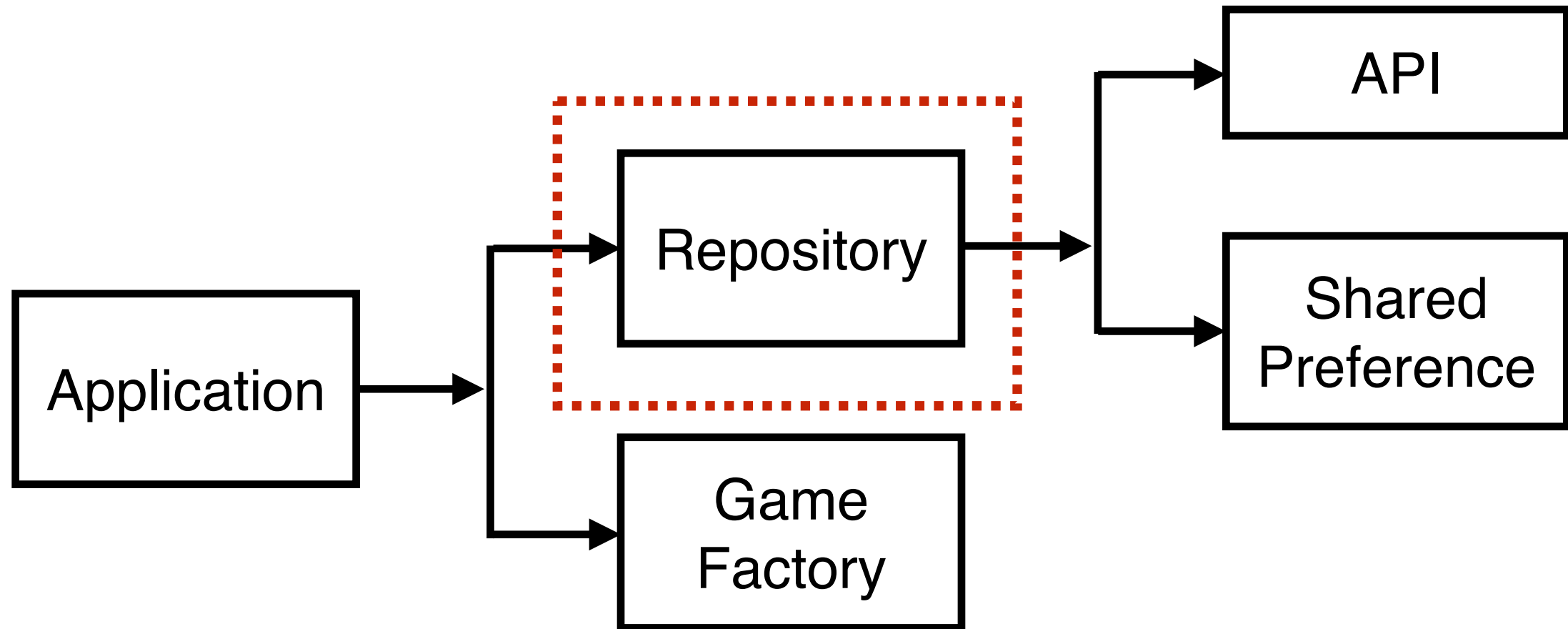




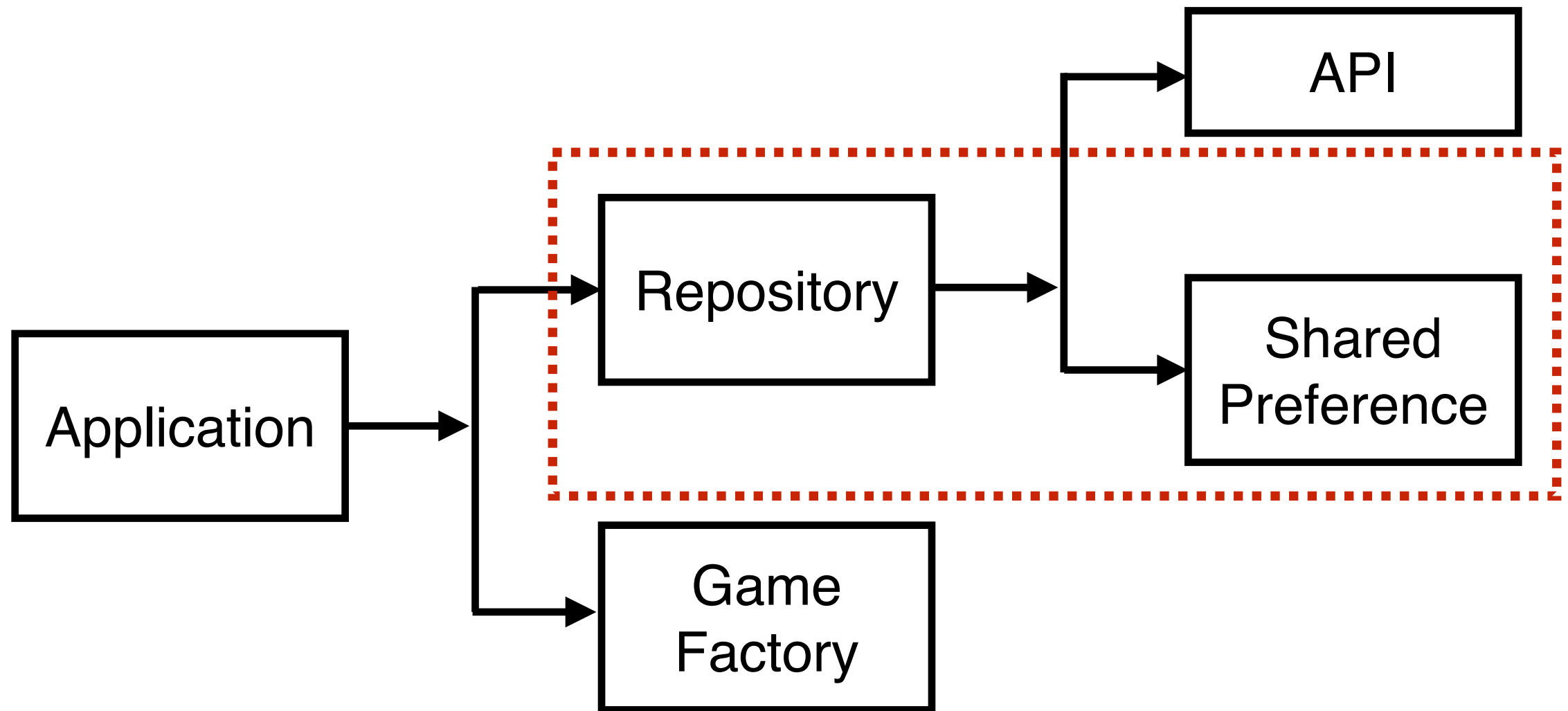
Repository Testing ?



Repository Testing ?



Repository + SharedPreferences ?



Unit test + Mockito



Test case 1

Save a high score in SharedPreferences

```
@RunWith(MockitoJUnitRunner::class)
class CocktailsRepositoryUnitTest {
    @Mock
    private lateinit var api: CocktailsApi

    @Mock
    private lateinit var sharedPreferences: SharedPreferences

    @Mock
    private lateinit var sharedPreferencesEditor: SharedPreferences.Editor
}
```



Test case 1

Save a high score in SharedPreferences

```
fun `save score to sharedPreferences`() {  
    whenever(sharedPreferences.edit())  
        .thenReturn(sharedPreferencesEditor)  
  
    val repository = CocktailsRepositoryImpl(api, sharedPreferences)  
  
    val score = 10  
    repository.saveHighScore(score)  
  
    inOrder(sharedPreferencesEditor) {  
        verify(sharedPreferencesEditor).putInt(any(), eq(score))  
        verify(sharedPreferencesEditor).apply()  
    }  
}
```



Test case 2

Get a high score from SharedPreferences ?



Instrumentation test



Test case

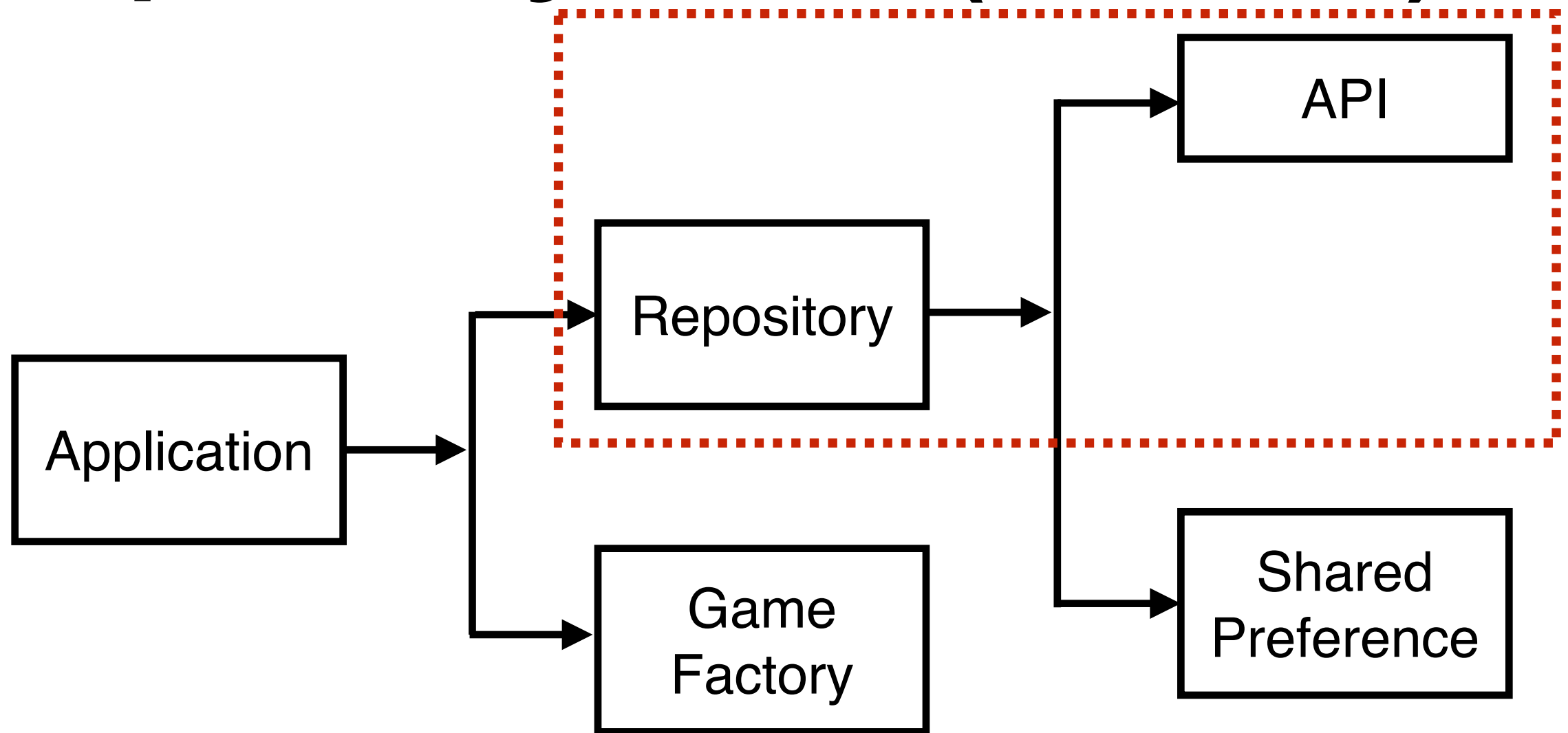
Save and Get a high score from SharedPreferences

@Test

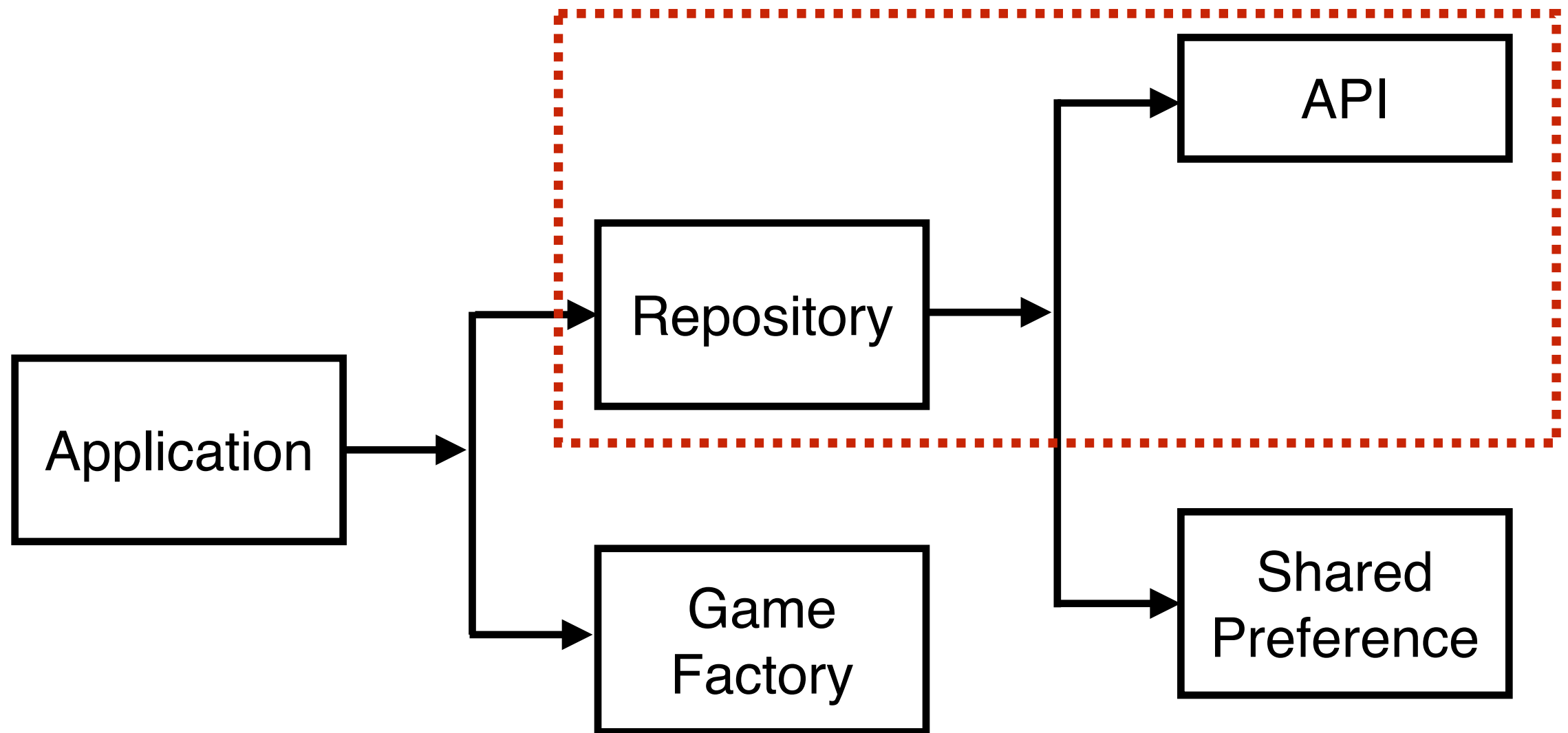
```
fun save_and_get_score_from_sharedpreference() {  
    val api: CocktailsApi = mock()  
    val appContext = InstrumentationRegistry.getInstrumentation().targetContext  
  
    val sharedPreferences = appContext.getSharedPreferences(  
        "TEST", Context.MODE_PRIVATE)  
    val repository = CocktailsRepositoryImpl(api, sharedPreferences)  
    val score = 100  
    repository.saveHighScore(score)  
  
    val actualScore = repository.getHighScore()  
  
    assertEquals(score, actualScore)  
}
```



Repository + API (Retrofit) ?



Unit test + MockWebServer



Test case 1

Success case :: read data from API

```
class CocktailsRepositoryAPITest {  
  
    @get:Rule  
    val mockWebServer = MockWebServer()  
  
    private val retrofit by lazy {  
        Retrofit.Builder()  
            .baseUrl(mockWebServer.url("/"))  
            .addConverterFactory(GsonConverterFactory.create())  
            .build()  
    }  
}
```



Test case 1

Success case :: read data from API

@Test

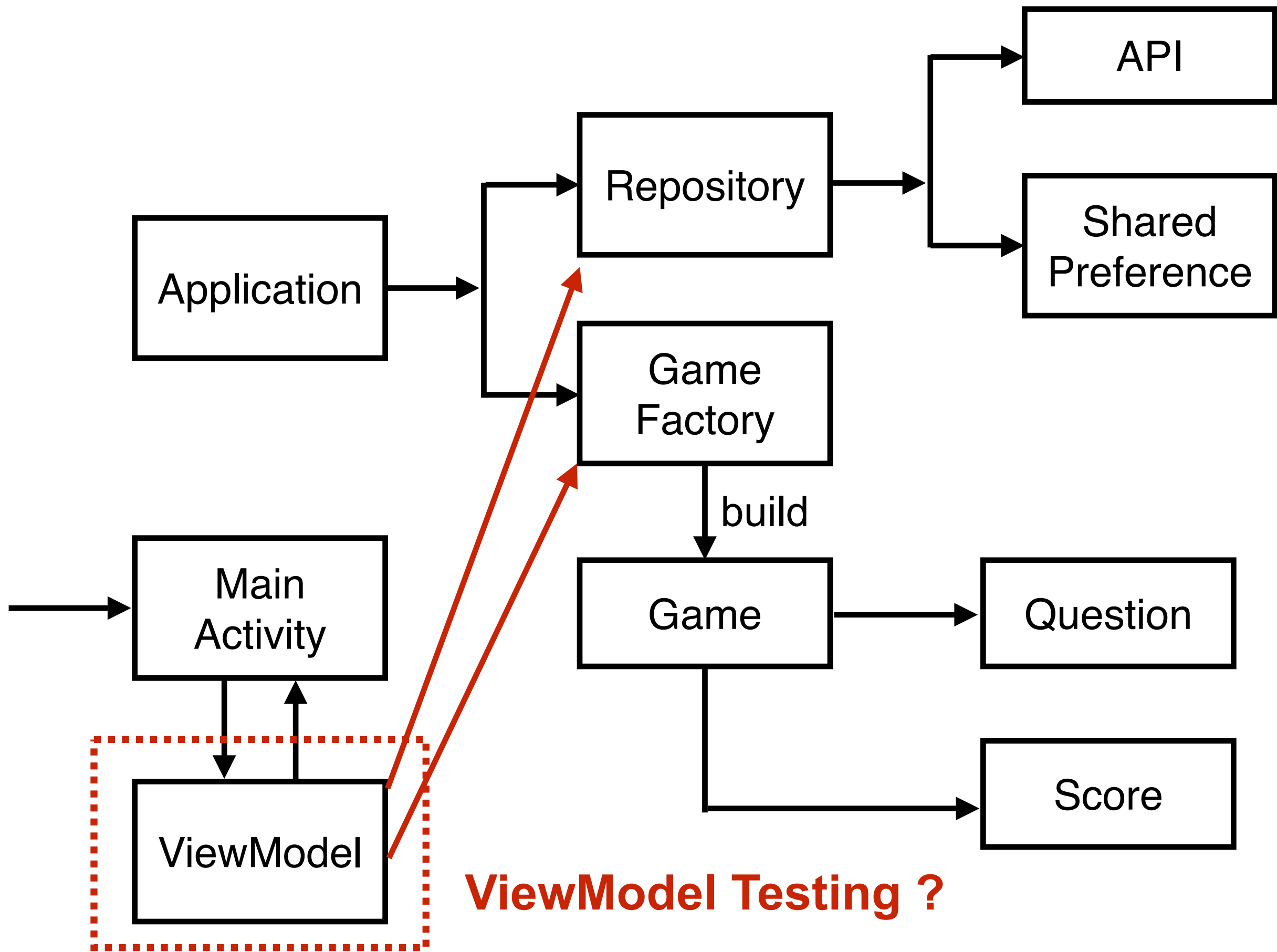
```
fun `Get all questions from API`() {  
    mockWebServer.enqueue(  
        MockResponse()  
            .setBody("{}")  
            .setResponseCode(200)  
    )  
    val sharedPreferences: SharedPreferences = mock()  
    val api = retrofit.create(CocktailsApi::class.java)  
    val repository = CocktailsRepositoryImpl(api, sharedPreferences)  
  
    val callback: RepositoryCallback<List<Cocktail>, String> = mock()  
    repository.getAlcoholic(callback)  
    verify(callback, timeout(100)).onSuccess(any())  
}
```

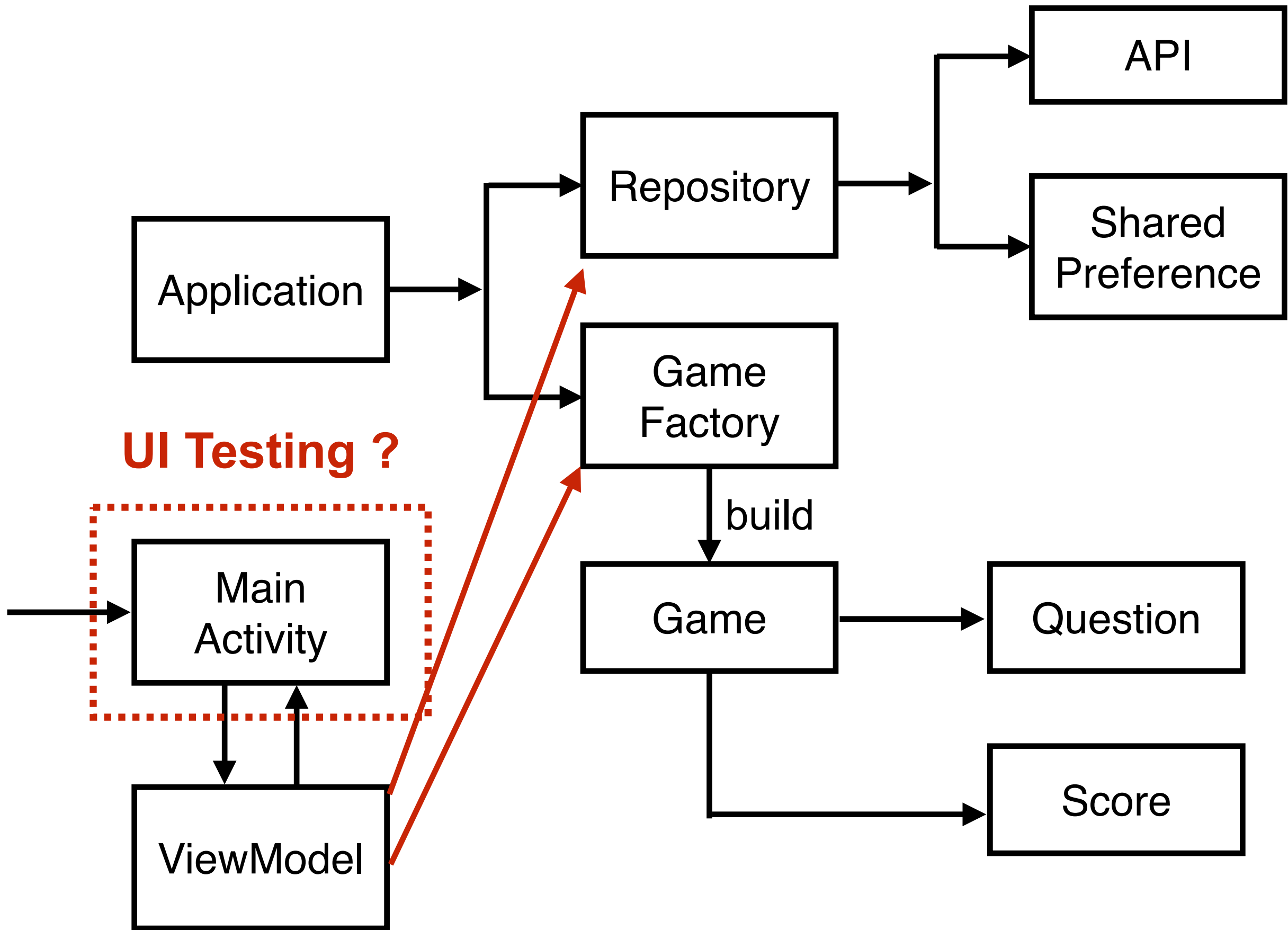


Test case2

Failure case :: read data from API ?







Unit test + Android framework with



<https://github.com/robolectric/robolectric>



Required JDK 9+



Config

/app/build.gradle

```
android {  
  
    testOptions {  
        unitTests {  
            includeAndroidResources = true  
        }  
    }  
}  
  
dependencies {  
  
    testImplementation 'org.robolectric:robolectric:4.4'  
}
```



Test case with Activity

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

    @Test
    fun start_new_game() {
        val scenario = ActivityScenario.launch(MainActivity::class.java)
        scenario.recreate()
    }
}
```



UI Instrumentation test

