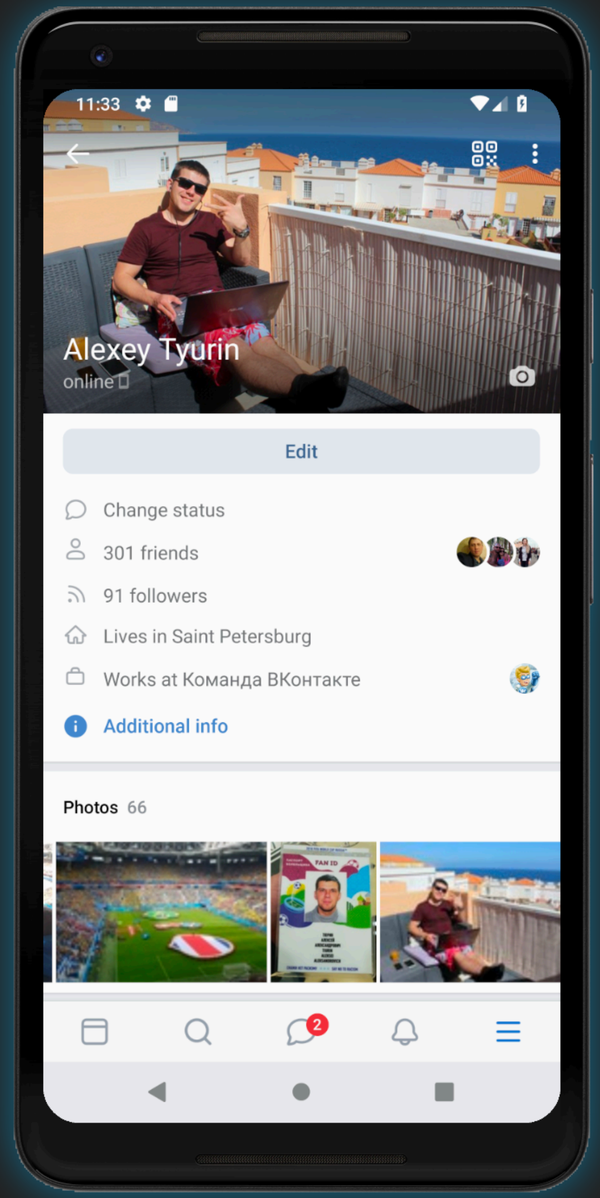
DON'T DO IT IN ANDROID TEST AUTOMATION

ALEKSEI TIURIN, VK

LEAD AUTOMATION ENGINEER AT VK

- vk.com/alex_tiurin
- github.com/alex-tiurin
- linkedin.com/in/aleksei-tiurin




DMC
DO YOU REMEMBER ME?

РЕШАЕМ ПРОБЛЕМЫ ESPRESSO АВТОТЕСТОВ ANDROID В РЕАЛЬНОМ МИРЕ

TYPES OF MISTAKES

TYPES OF MISTAKES

- Test data usage

TYPES OF MISTAKES

- Test data usage
- Test flow

TYPES OF MISTAKES

- Test data usage
- Test flow
- Test framework architecture

TYPES OF MISTAKES

- Test data usage
- Test flow
- Test framework architecture
- Reporting


DMC


MARTY-JUNIOR


DOC-SENIOR
MARTY-JUNIOR

```
class SomeTestClass {
    ...
}
```

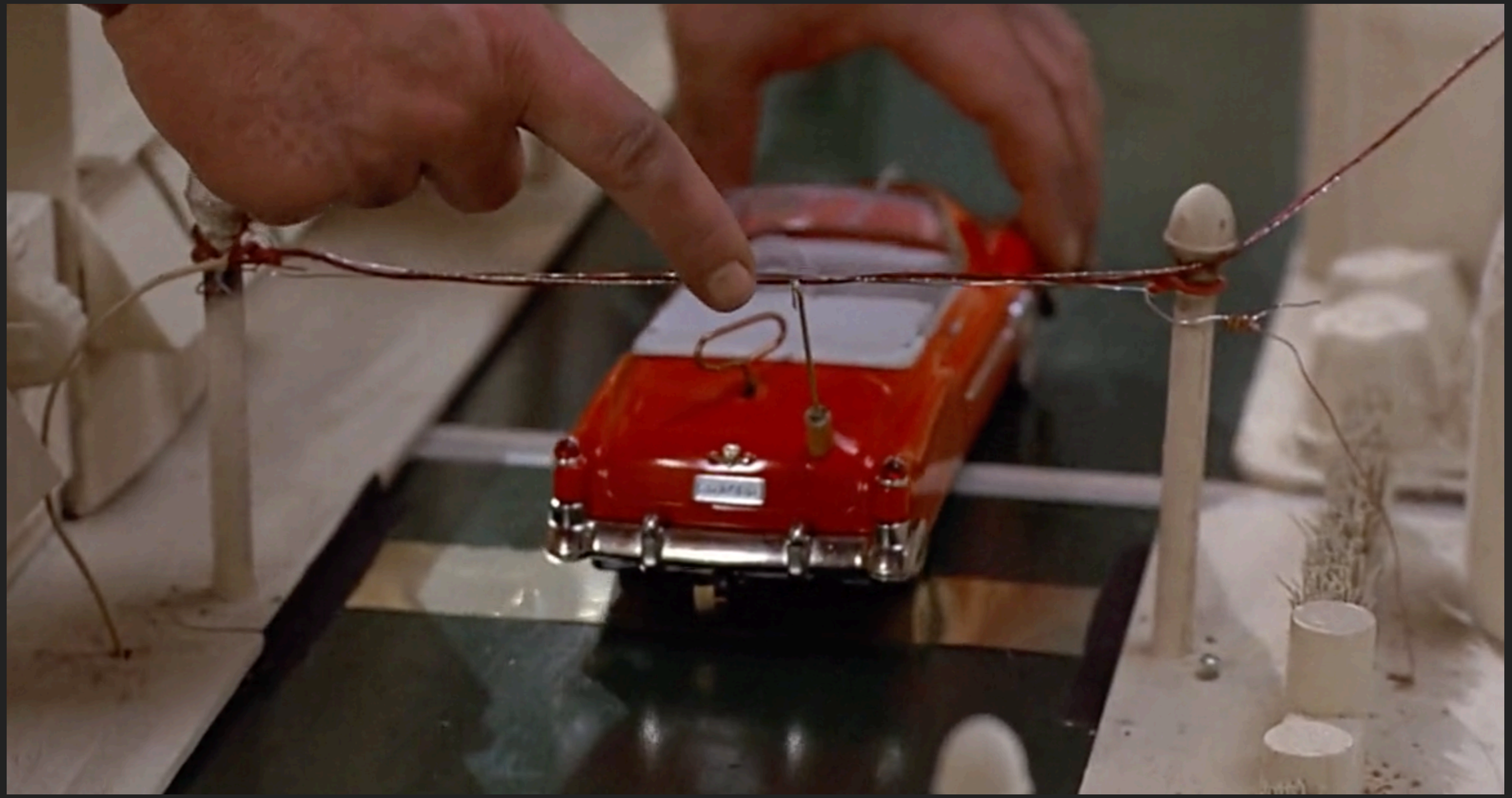
```
class SomeTestClass {
    companion object {
        @BeforeClass @JvmStatic
        fun executeBeforeAllTests(){}
    }
    ...
}
```

```
class SomeTestClass {
    companion object {
        @BeforeClass @JvmStatic
        fun executeBeforeAllTests(){}
    }

    @Before
    fun executeBeforeEachTest(){}
    ...
}
```

```
class SomeTestClass {
    companion object {
        @BeforeClass @JvmStatic
        fun executeBeforeAllTests(){}
    }

    @Before
    fun executeBeforeEachTest(){}

    @Test
    fun assertSmthTest(){}
}
```

```
class SomeTestClass {
    companion object {
        @BeforeClass @JvmStatic
        fun executeBeforeAllTests(){}
    }

    @get:Rule
    val activityRule = ActivityTestRule(MainActivity::class.java)

    @Before
    fun executeBeforeEachTest(){}

    @Test
    fun assertSmthTest(){}
}
```

TEST DATA USAGE




MARTY, STOP USING TEST DATA EVERYWHERE!!!

```
object TestData {
    val chatId = "..."
    val documentId = "..."
}
```

```
object TestData {
    val chatId = "..."
    val documentId = "..."
}
// Somewhere in user steps
fun sendDocInChat(user: User){
    addDocToUser(user, TestData.documentId)
    sendDoc(TestData.chatId,
            TestData.documentId)
}
```

```
object TestData {
    val chatId = "..."
    val documentId = "..."
}
// Somewhere in user steps
fun sendDocInChat(user: User){
    addDocToUser(user, TestData.documentId)
    sendDoc(TestData.chatId,
            TestData.documentId)
}
// Somewhere in test class
@Test fun someTest(){
    sendDocInChat(user)
}
```

WHAT'S A PROBLEM

WHAT'S A PROBLEM

- Lost a control on test data

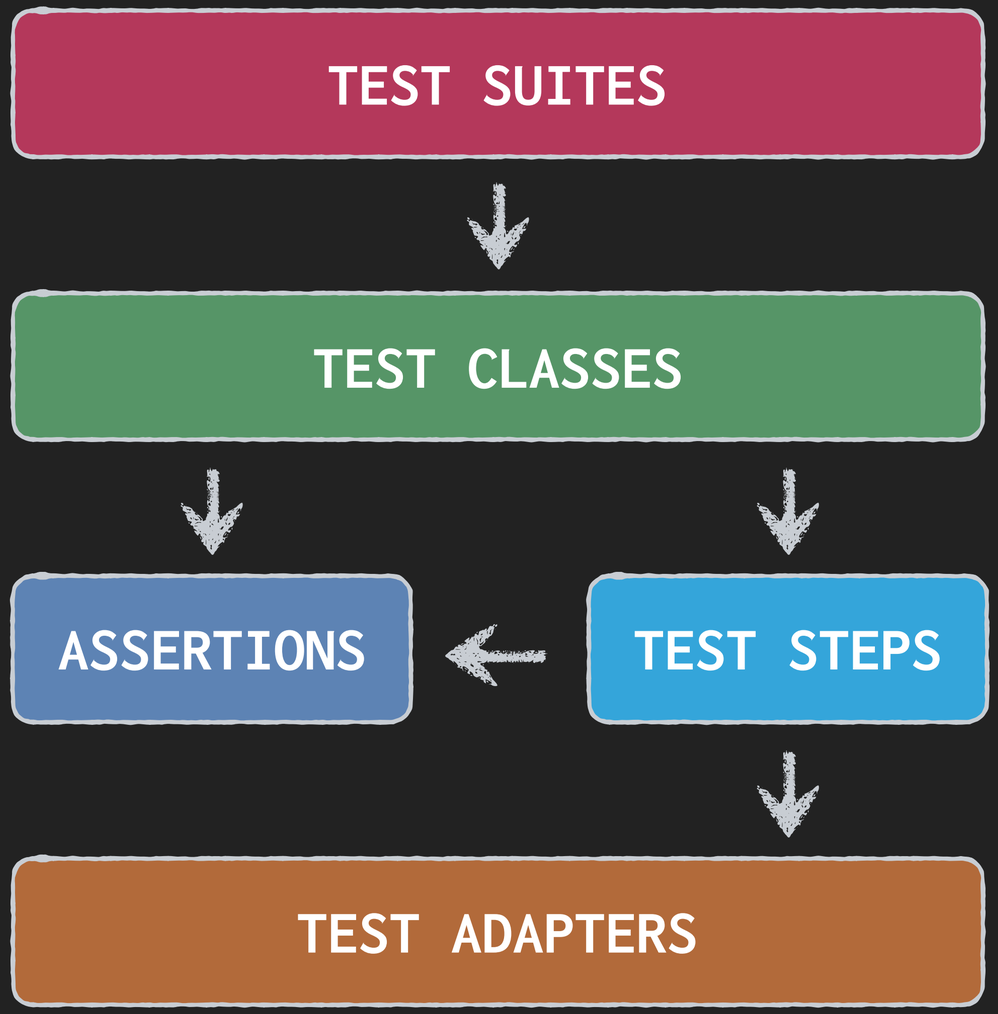
WHAT'S A PROBLEM

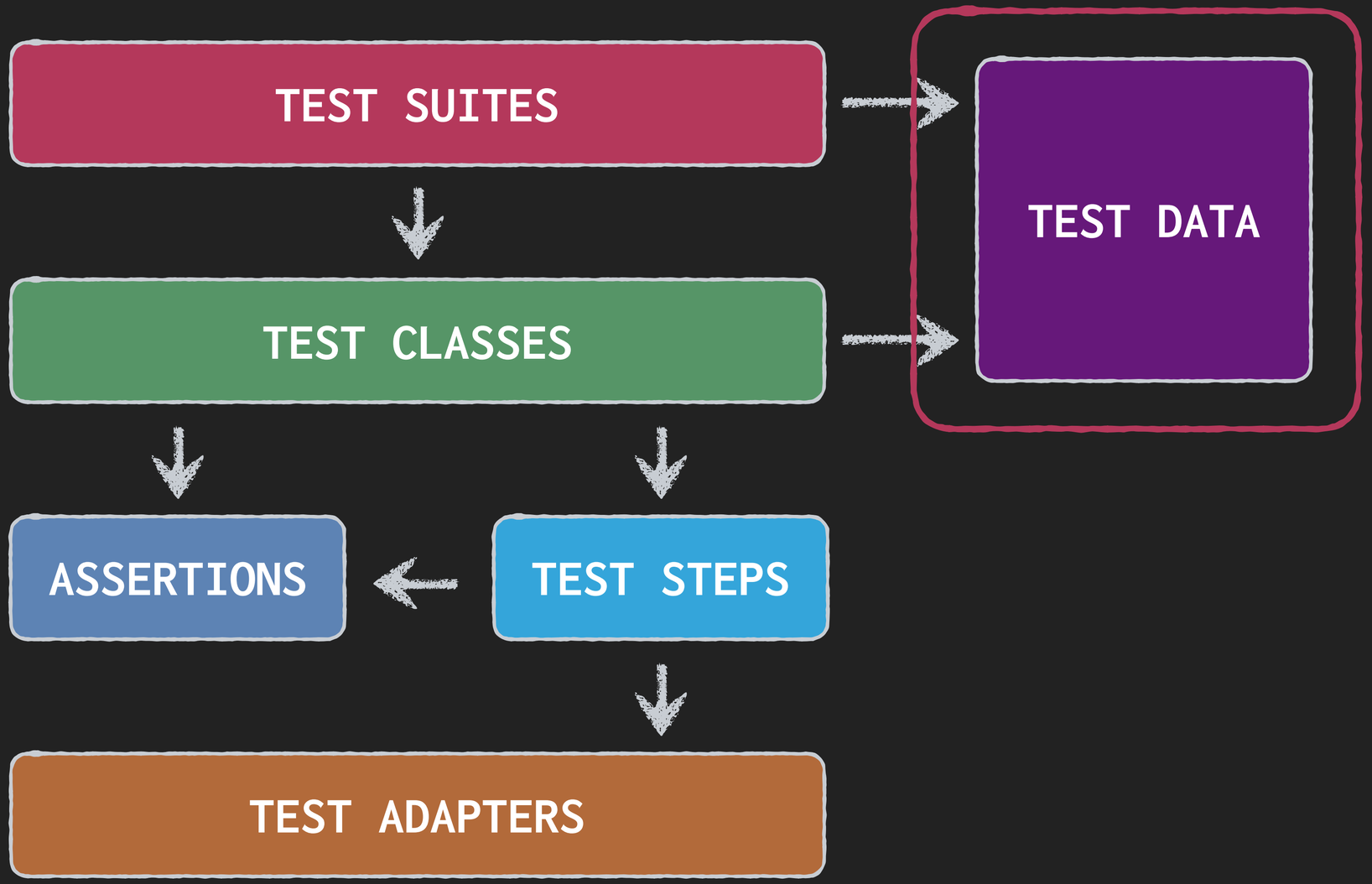
- Lost a control on test data
- Unexpected test application condition

WHAT'S A PROBLEM

- Lost a control on test data
- Unexpected test application condition
- The same data can be used by several tests at the same time

WHERE TEST DATA SHOULD BE USED?




TEST SUITES
TEST DATA
TEST CLASSES
ASSERTIONS
TEST STEPS
TEST ADAPTERS

Д. БУЗДИН – КАК ПОСТРОИТЬ СВОИ ФРЕЙМВОРК ДЛЯ АВТОТЕСТОВ?

SOLUTION

- Restrict the scope of test data definition to test class / test suite

```
object TestData {
    val chatId = "..."
    val documentId = "..."
}
// Somewhere in user steps
fun sendDocInChat(user: User){
    addDocToUser(user, TestData.documentId)
    sendDoc(TestData.chatId,
            TestData.documentId)
}
```

```
object TestData {
    val chatId = "..."
    val documentId = "..."
}
// Somewhere in user steps
fun sendDocInChat(user: User, docId: String, chatId: String){
    addDocToUser(user, docId)
    sendDoc(chatId, docId)
}
```

```
object TestData {
    val chatId = "..."
    val documentId = "..."
}
// Somewhere in user steps
fun sendDocInChat(user: User, docId: String, chatId: String){
    addDocToUser(user, docId)
    sendDoc(chatId, docId)
}
```

```
object TestData {
    val chatId = "..."
    val documentId = "..."
}
// Somewhere in user steps
fun sendDocInChat(user: User, docId: String, chatId: String){
    addDocToUser(user, docId)
    sendDoc(chatId, docId)
}
// Somewhere in test class
@Test fun someTest(){
    sendDocInChat(user, documentId, chatId)
}
```

SOLUTION

- Restrict the scope of test data definition to test class / test suite

SOLUTION

- Restrict the scope of test data definition to test class / test suite
- Store read-only data as project constants

```
object TestData {
    // editable value, should be calculated somehow
    val chatId = TestDataManager.createChat()
    val documentId = "..."
}
// Somewhere in test class
@Test fun someTest(){
    sendDocInChat(user, documentId, chatId)
}
```

```
object TestData {
    // editable value, should be calculated somehow
    val chatId = TestDataManager.createChat()
    val documentId = TestData.documentId // read-only value
}
// Somewhere in test class
@Test fun someTest(){
    sendDocInChat(user, documentId, chatId)
}
```

TEST DATA IS DEFINED AFTER ACTIVITY LAUNCH

TEST DATA IS DEFINED AFTER ACTIVITY LAUNCH

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java)


@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
}
```

REASON: @BEFORE EXECUTES AFTER ACTIVITY LAUNCHED

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java)


@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
}
```

REASON: @BEFORE EXECUTES AFTER ACTIVITY LAUNCHED

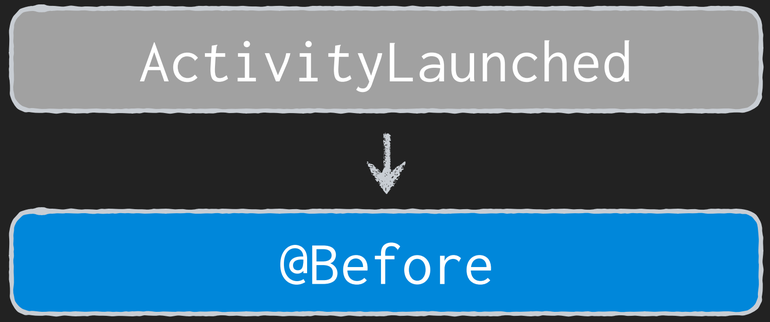
```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java)

@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
}
```

ActivityLaunched

REASON: @BEFORE EXECUTES AFTER ACTIVITY LAUNCHED

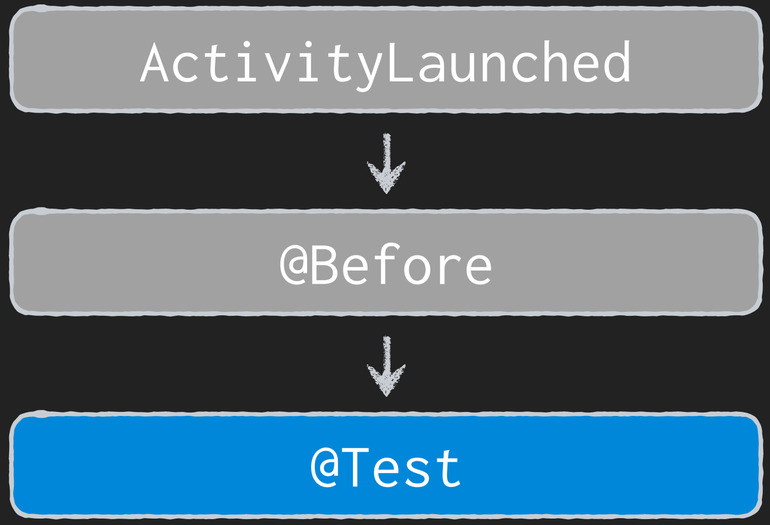
```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java)
```

```
@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
}
```



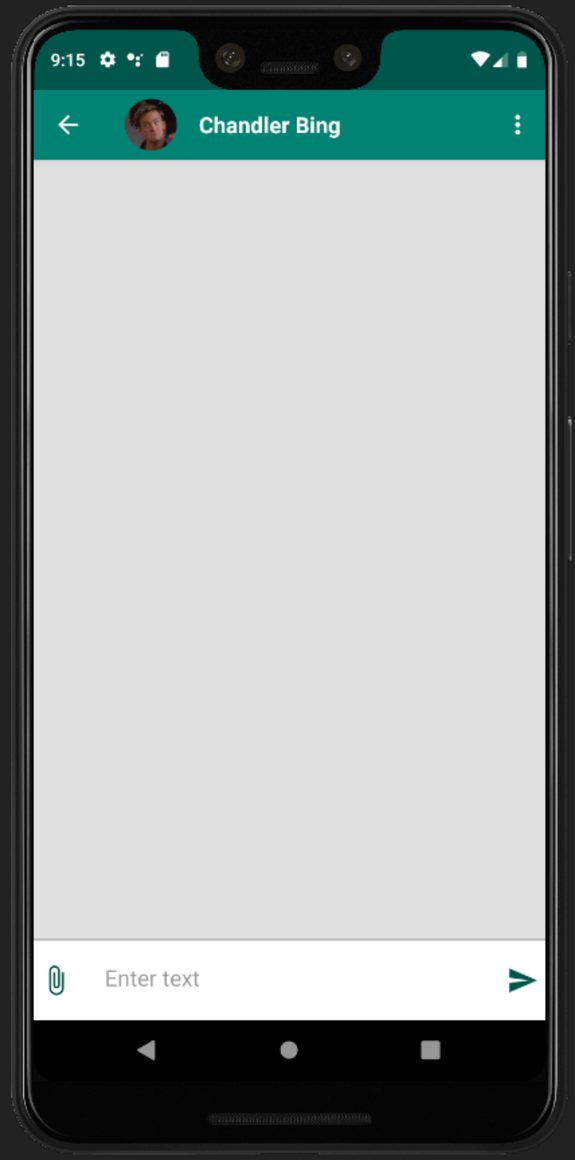
REASON: @BEFORE EXECUTES AFTER ACTIVITY LAUNCHED

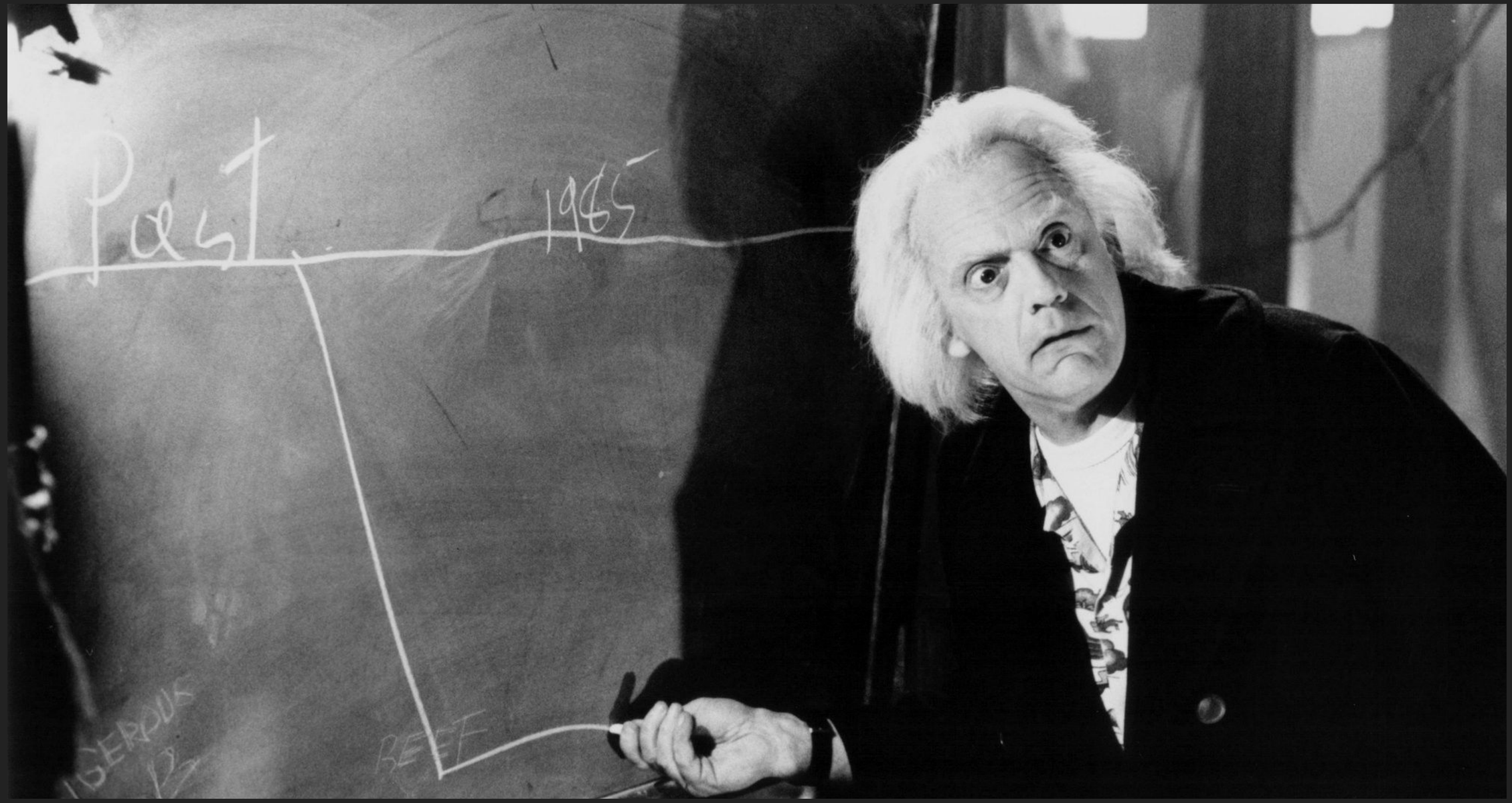
```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java)


@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
}
```

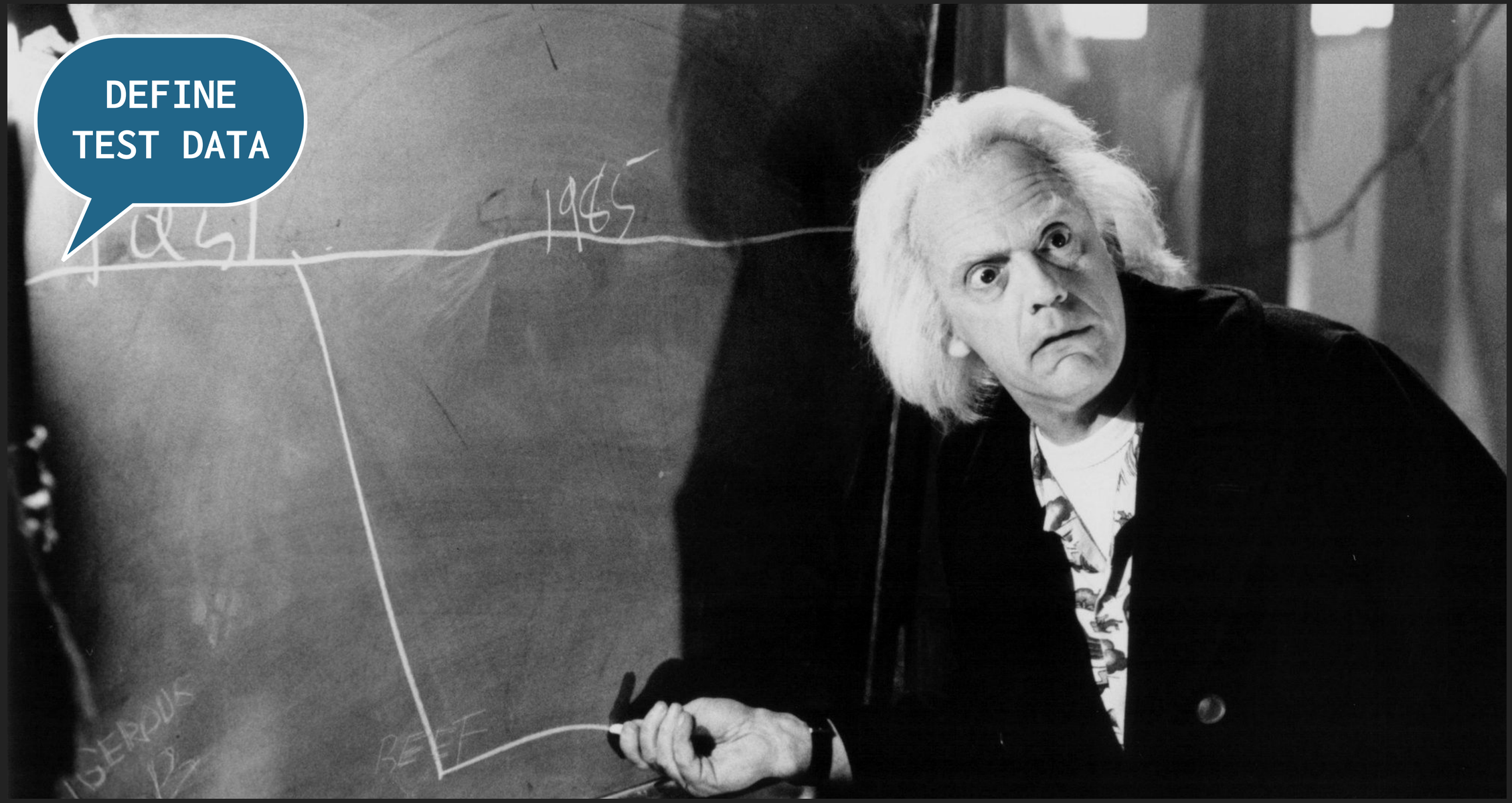


REASON: @BEFORE EXECUTES AFTER ACTIVITY LAUNCHED

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java)


@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
}
```




Past
1985
REEF


DEFINE
TEST DATA
1965
REEF


DEFINE TEST DATA
LAUNCH ACTIVITY


DEFINE
TEST DATA
LAUNCH
ACTIVITY
@TEST 1


DEFINE TEST DATA
LAUNCH ACTIVITY
@TEST 1
@TEST 2

SOLUTION #1: SET UP DATA IN @BEFORECLASS

@BeforeClass

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java)
companion object {
    @BeforeClass @JvmStatic
    fun setUpTestData(){
        Api.addChatMessage(contact.id, "text")
    }
}
```

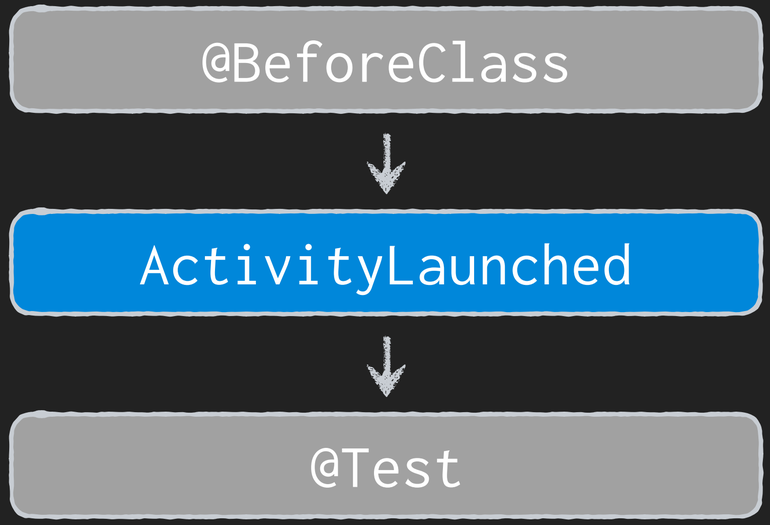
SOLUTION #1: SET UP DATA IN @BEFORECLASS

@BeforeClass

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java)
companion object {
    @BeforeClass @JvmStatic
    fun setUpTestData(){
        Api.addChatMessage(contact.id, "text")
    }
}
```

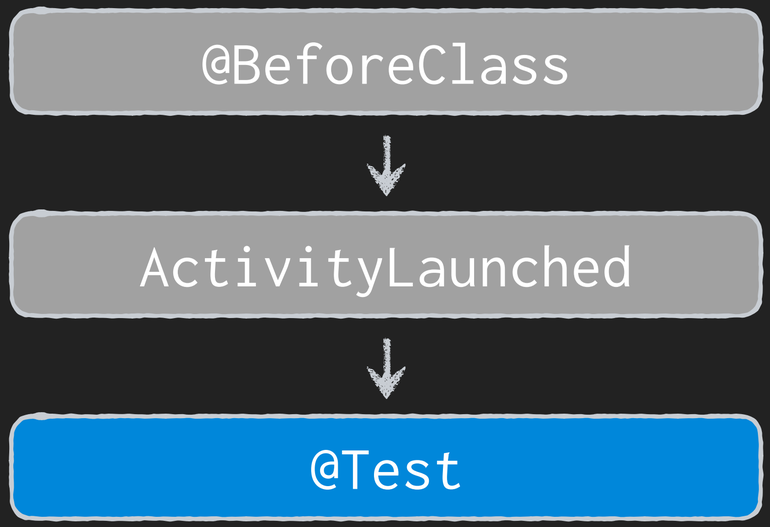
SOLUTION #1: SET UP DATA IN @BEFORECLASS

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java)
companion object {
    @BeforeClass @JvmStatic
    fun setUpTestData(){
        Api.addChatMessage(contact.id, "text")
    }
}
```



SOLUTION #1: SET UP DATA IN @BEFORECLASS

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java)
companion object {
    @BeforeClass @JvmStatic
    fun setUpTestData(){
        Api.addChatMessage(contact.id, "text")
    }
}
```



SOLUTION #2: LAUNCH ACTIVITY = FALSE

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java,
                     initialTouchMode = false,
                     launchActivity = false)

@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
    activityRule.launchActivity(Intent())
}
```

SOLUTION #2: LAUNCH ACTIVITY = FALSE

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java,
                     initialTouchMode = false,
                     launchActivity = false)

@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
    activityRule.launchActivity(Intent())
}
```

SOLUTION #2: LAUNCH ACTIVITY = FALSE

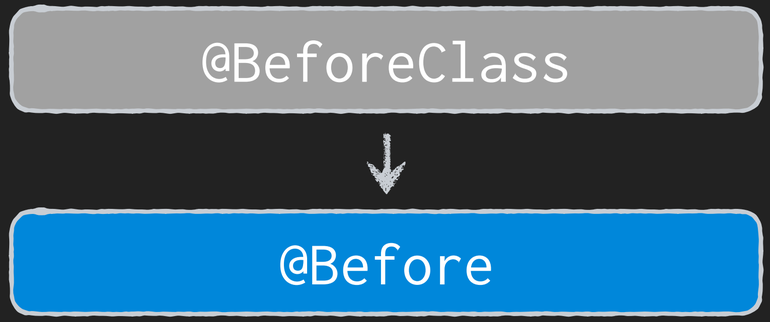
@BeforeClass

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java,
                     initialTouchMode = false,
                     launchActivity = false)

@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
    activityRule.launchActivity(Intent())
}
```

SOLUTION #2: LAUNCH ACTIVITY = FALSE

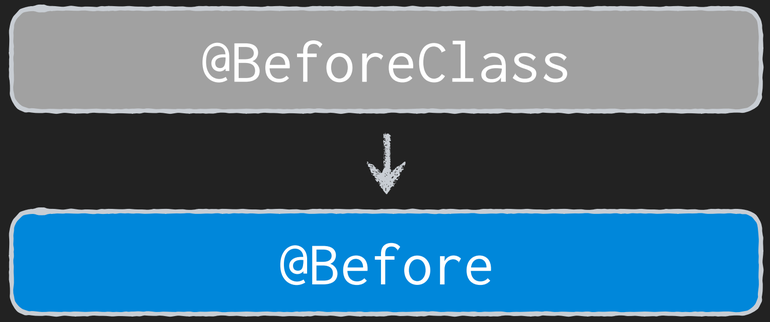
```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java,
                     initialTouchMode = false,
                     launchActivity = false)
```

```
@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
    activityRule.launchActivity(Intent())
}
```



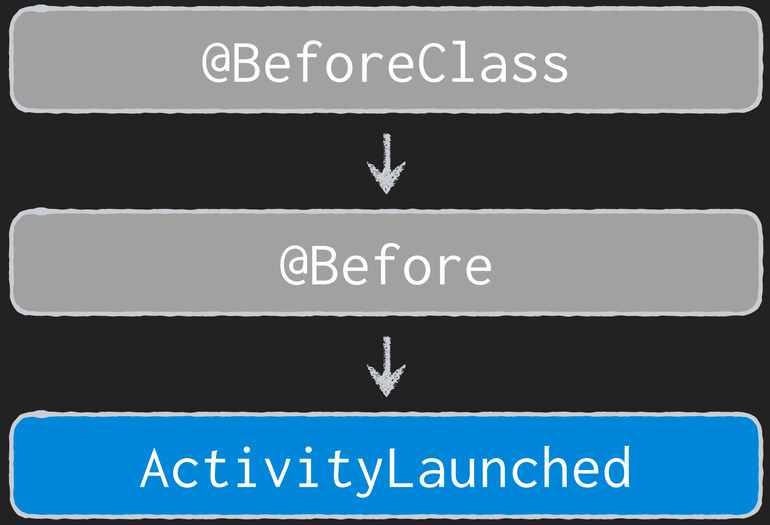
SOLUTION #2: LAUNCH ACTIVITY = FALSE

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java,
                     initialTouchMode = false,
                     launchActivity = false)

@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
    activityRule.launchActivity(Intent())
}
```



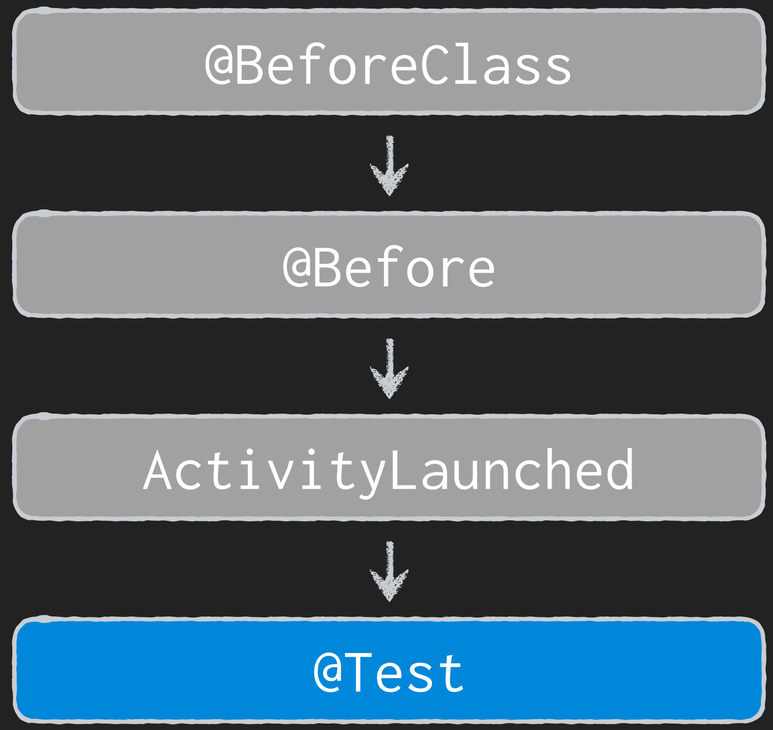
SOLUTION #2: LAUNCH ACTIVITY = FALSE

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java,
                     initialTouchMode = false,
                     launchActivity = false)

@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
    activityRule.launchActivity(Intent())
}
```



SOLUTION #2: LAUNCH ACTIVITY = FALSE

```
@get:Rule
val activityRule =
    ActivityTestRule(MainActivity::class.java,
                     initialTouchMode = false,
                     launchActivity = false)

@Before
fun setUp(){
    Api.addChatMessage(contact.id, "text")
    activityRule.launchActivity(Intent())
}
```



SOLUTION #3: CUSTOM SETUP RULE

```
class SetUpTearDownRule : TestWatcher() {
    val setUps = mutableListOf<Condition>()
    val tearDowns = mutableListOf<Condition>()
}
```

SOLUTION #3: CUSTOM SETUP RULE

```
class SetUpTearDownRule : TestWatcher() {
    val setUps = mutableListOf<Condition>()
    val tearDowns = mutableListOf<Condition>()
}
```

SOLUTION #3: CUSTOM SETUP RULE

```
class SetUpTearDownRule : TestWatcher() {
    val setUps = mutableListOf<Condition>()
    val tearDowns = mutableListOf<Condition>()
}
```

SOLUTION #3: CUSTOM SETUP RULE

```
class SetUpTearDownRule : TestWatcher() {
    private val setUps = mutableListOf<Condition>()
    inner class Condition(
        val counter: Int,
        val key: String,
        val actions: () -> Unit
    )
}
```

SOLUTION #3: CUSTOM SETUP RULE

```
class SomeTestClass {
    @get:Rule
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            Api.addChatMessage(contact.id, "text")
        }
    ...
}
```

SOLUTION #3: CUSTOM SETUP RULE

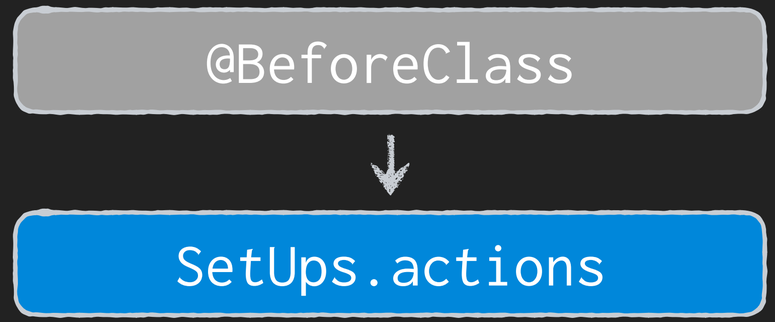
```
class SomeTestClass {
    @get:Rule
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            Api.addChatMessage(contact.id, "text")
        }
    ...
}
```

SOLUTION #3: CUSTOM SETUP RULE

@BeforeClass

```
class SomeTestClass {
    @get:Rule
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            Api.addChatMessage(contact.id, "text")
        }
    @get:Rule
    val activityRule =
        ActivityTestRule(MainActivity::class.java)
    ...
}
```

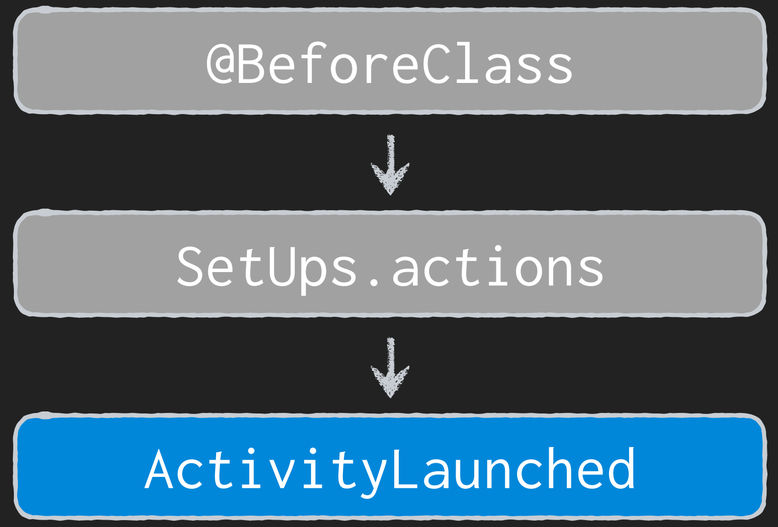
SOLUTION #3: CUSTOM SETUP RULE




```
class SomeTestClass {
    @get:Rule
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            Api.addChatMessage(contact.id, "text")
        }
    @get:Rule
    val activityRule =
        ActivityTestRule(MainActivity::class.java)
    ...
}
```

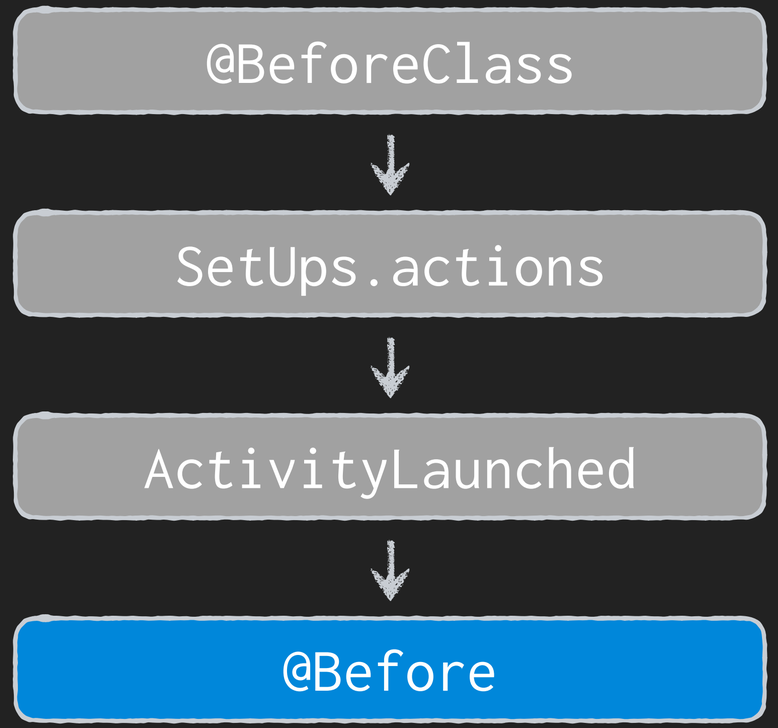
SOLUTION #3: CUSTOM SETUP RULE

```
class SomeTestClass {
    @get:Rule
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            Api.addChatMessage(contact.id, "text")
        }
    @get:Rule
    val activityRule =
        ActivityTestRule(MainActivity::class.java)
    ...
}
```



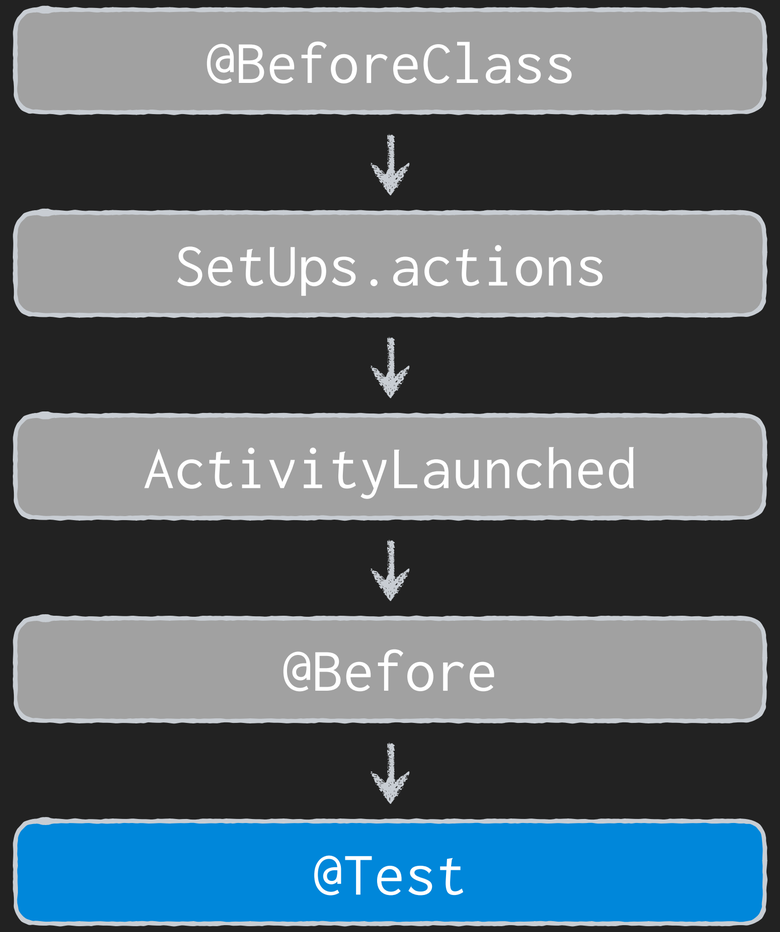
SOLUTION #3: CUSTOM SETUP RULE

```
class SomeTestClass {
    @get:Rule
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            Api.addChatMessage(contact.id, "text")
        }
    @get:Rule
    val activityRule =
        ActivityTestRule(MainActivity::class.java)
    ...
}
```



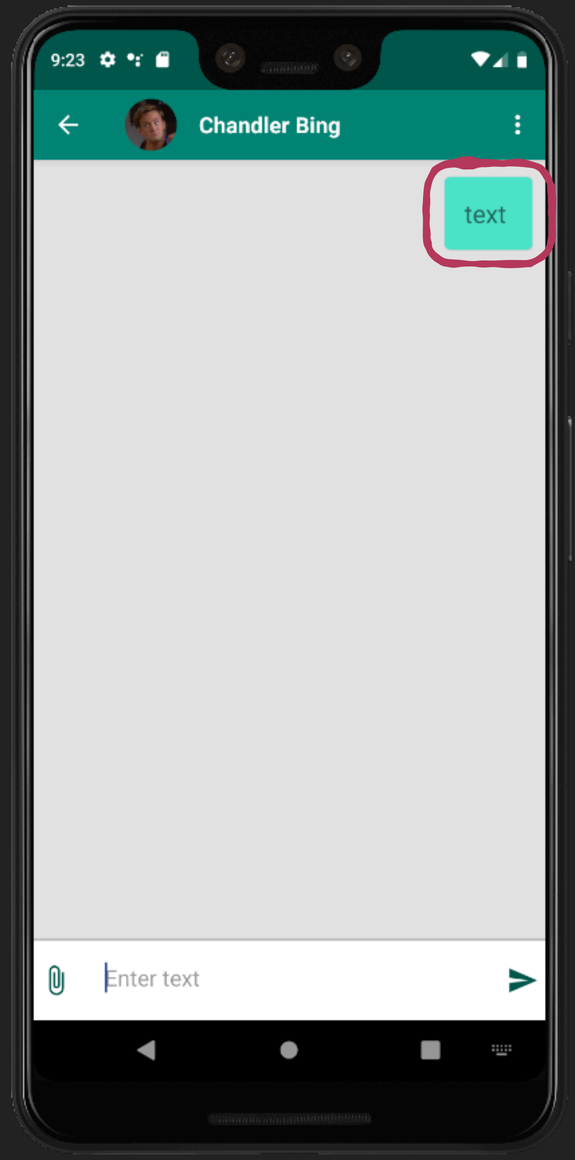
SOLUTION #3: CUSTOM SETUP RULE

```
class SomeTestClass {
    @get:Rule
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            Api.addChatMessage(contact.id, "text")
        }
    @get:Rule
    val activityRule =
        ActivityTestRule(MainActivity::class.java)
    ...
}
```



SOLUTION #3: CUSTOM SETUP RULE

```
class SomeTestClass {
    @get:Rule
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            Api.addChatMessage(contact.id, "text")
        }
    @get:Rule
    val activityRule =
        ActivityTestRule(MainActivity::class.java)
    ...
}
```




SOLUTION #3
DOESN'T WORK
WTF?

SOLUTION #3 DOESN'T WORK ALWAYS! WHY?

```
class SomeTestClass {
    @get:Rule
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            Api.addChatMessage(contact.id, "text")
        }
    @get:Rule
    val activityRule =
        ActivityTestRule(MainActivity::class.java)
    ...
}
```

SOLUTION #3 DOESN'T WORK ALWAYS! WHY?

```
class SomeTestClass {
    @get:Rule
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            Api.addChatMessage(contact.id, "text")
        }
    @get:Rule
    val activityRule =
        ActivityTestRule(MainActivity::class.java)
    ...
}
```

ORDER RULES WITH RULECHAIN

```
@get:Rule
val setupRule = SetUpTearDownRule().addSetUp{...}
@get:Rule
val activityRule = ActivityTestRule(MainActivity::class.java)

@get:Rule
val ruleChain = RuleChain.outerRule(setupRule)
                         .around(activityRule)
```

ORDER RULES WITH RULECHAIN

```
@get:Rule
val setupRule = SetUpTearDownRule().addSetUp{...}
@get:Rule
val activityRule = ActivityTestRule(MainActivity::class.java)

@get:Rule
val ruleChain = RuleChain.outerRule(setupRule)
                         .around(activityRule)
```

ORDER RULES WITH RULECHAIN

```
val setupRule = SetUpTearDownRule().addSetUp{...}
val activityRule = ActivityTestRule(MainActivity::class.java)

@get:Rule
val ruleChain = RuleChain.outerRule(setupRule)
                         .around(activityRule)
```


MARTY, ARE YOU REALLY
TELLING ME YOUR PLAN
IS BASED ON
“BACK TO THE FUTURE”?


MARTY, ARE YOU REALLY
TELLING ME YOUR PLAN
IS BASED ON
“RULE CHAIN” IN 2020?

ORDER RULES WITH RULECHAIN

```
val setupRule = SetUpTearDownRule().addSetUp{...}
val activityRule = ActivityTestRule(MainActivity::class.java)

@get:Rule
val ruleChain = RuleChain.outerRule(setupRule)
                         .around(activityRule)
```

REPLACE RULE CHAIN WITH RULE SEQUENCE

```
val setupRule = SetUpTearDownRule().addSetUp{...}
val activityRule = ActivityTestRule(MainActivity::class.java)

@get:Rule
val ruleChain = RuleChain.outerRule(setupRule)
                         .around(activityRule)
@get:Rule
val ruleSequence = RuleSequence(setupRule, activityRule)
```

ADVANCES OF RULE SEQUENCE

- Friendly interface

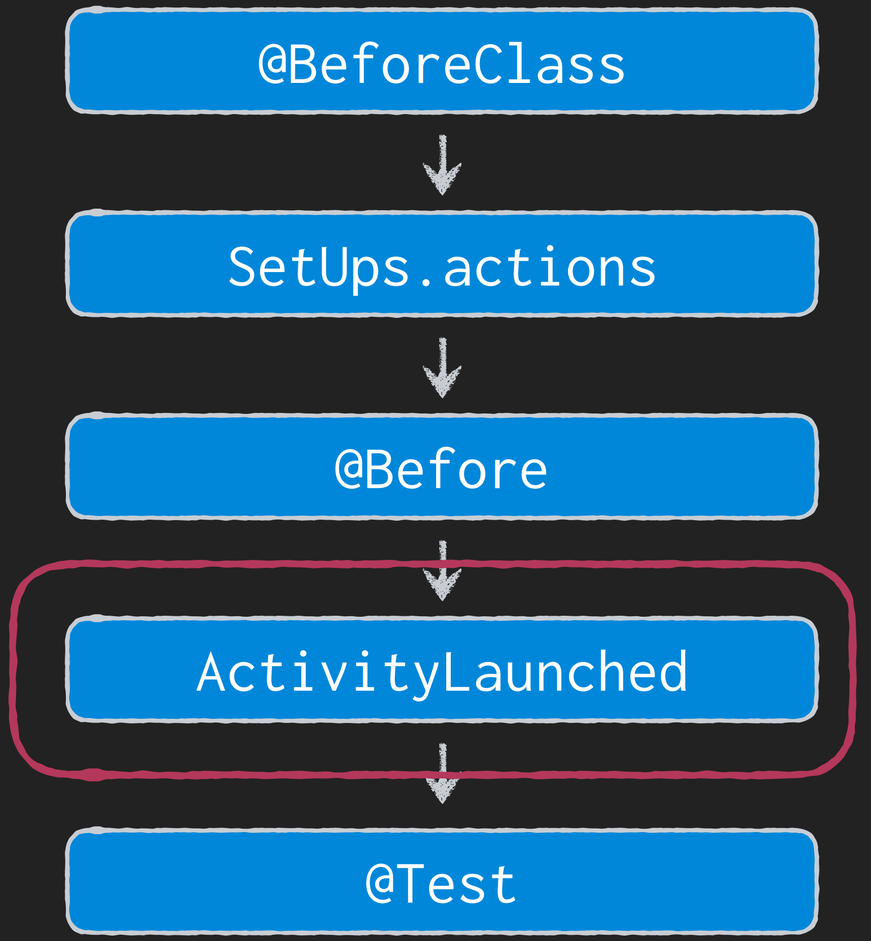
ADVANCES OF RULE SEQUENCE

- Friendly interface
- Better work with test classes inheritance

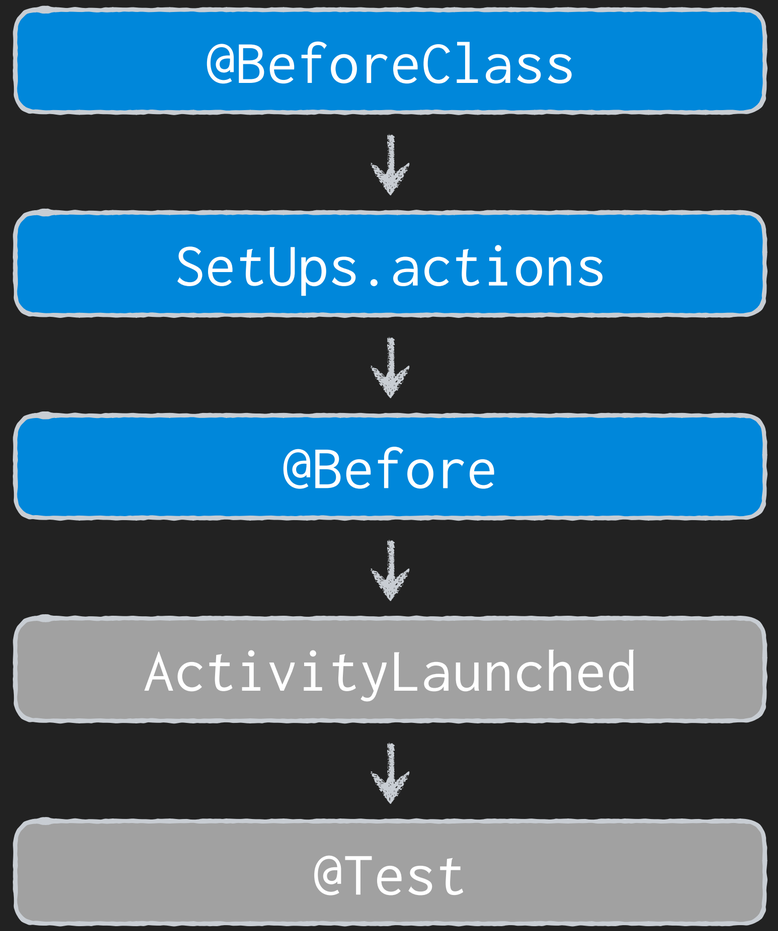
ADVANCES OF RULE SEQUENCE

- Friendly interface
- Better work with test classes inheritance
- Three lists of prioritised rules (add, addFirst, addLast)

SOLUTIONS

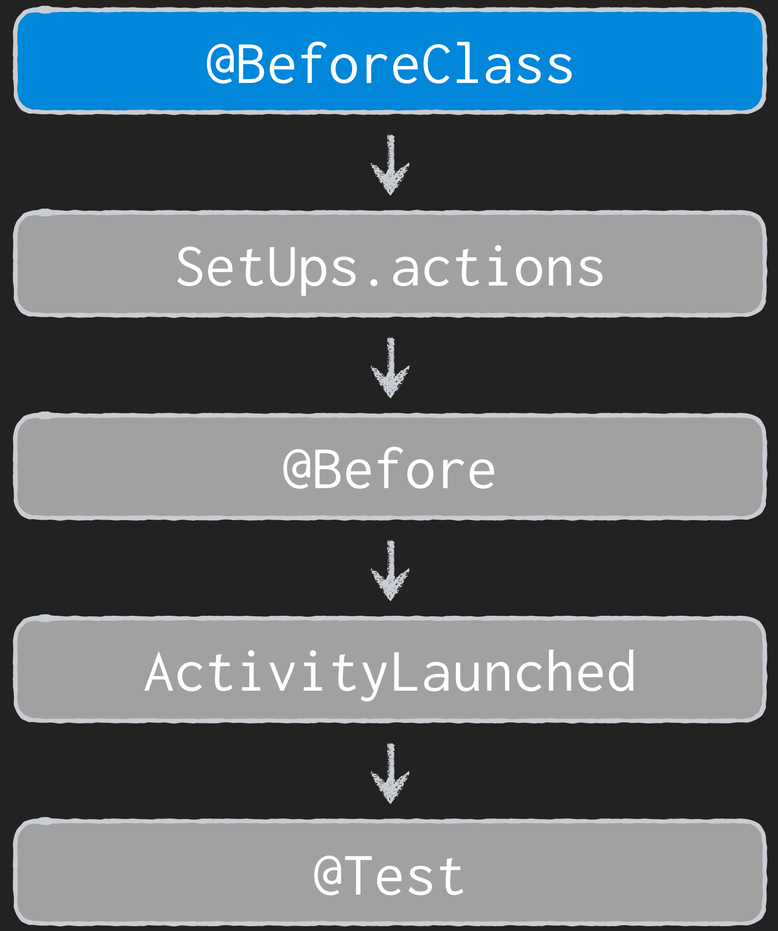

@BeforeClass
SetUps.actions
@Before
ActivityLaunched
@Test

SOLUTIONS


@BeforeClass
SetUps.actions
@Before
ActivityLaunched
@Test

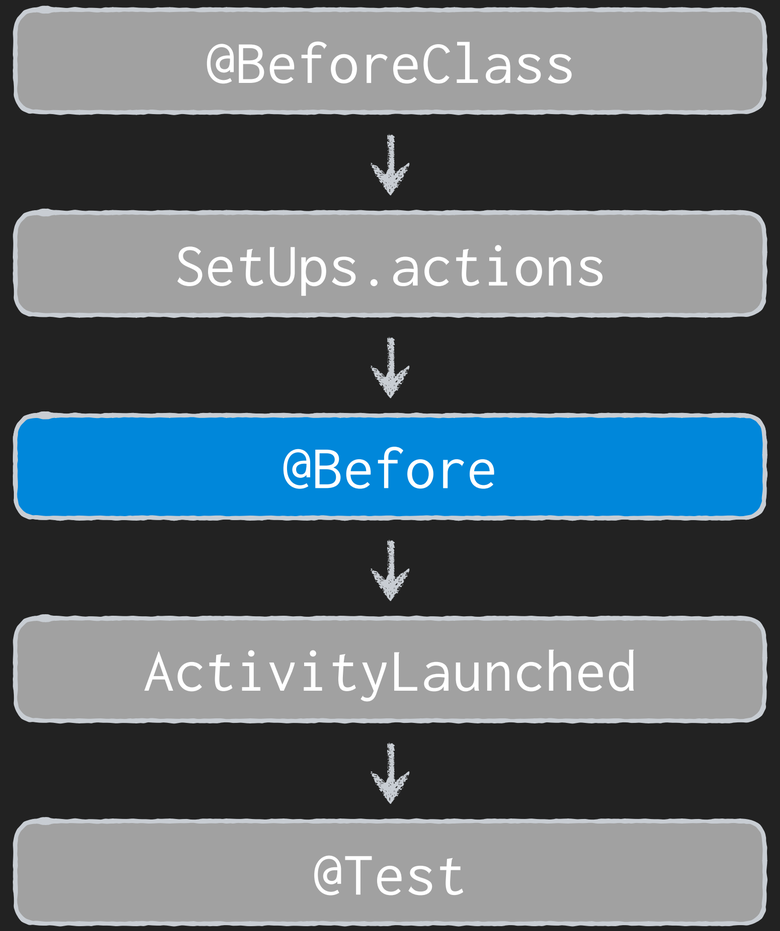
SOLUTIONS

- Set up test data in @BeforeClass



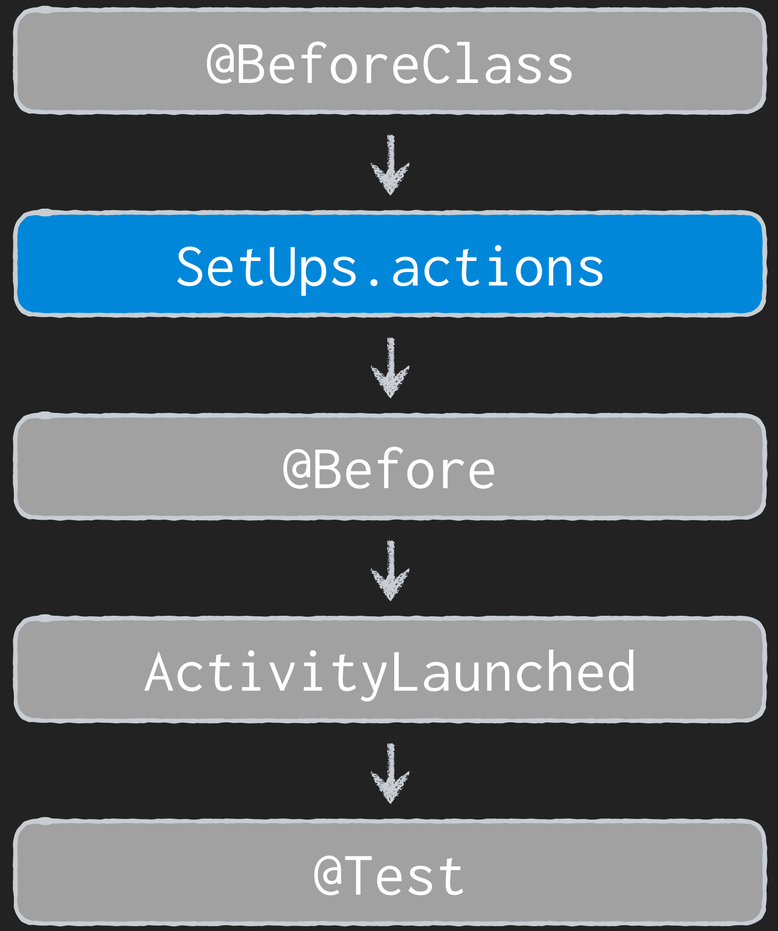
SOLUTIONS

- Set up test data in @BeforeClass
- LaunchActivity = false. Set up in @Before



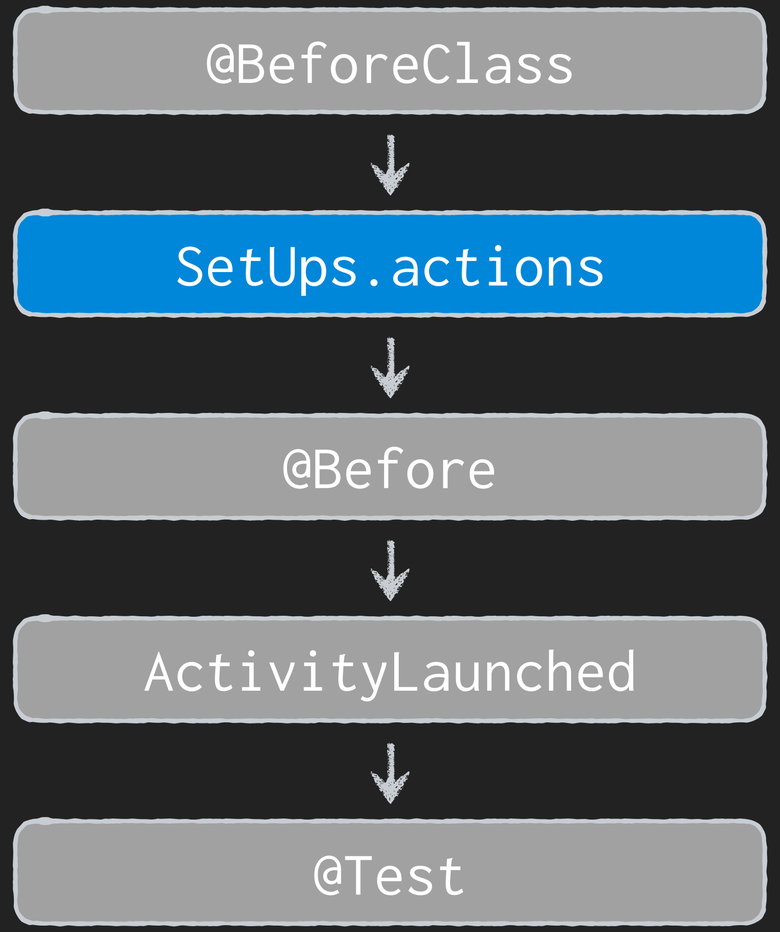
SOLUTIONS

- Set up test data in @BeforeClass
- LaunchActivity = false. Set up in @Before
- Custom SetUpRule + RuleSequence



SOLUTIONS

- Set up test data in @BeforeClass
- LaunchActivity = false. Set up in @Before
- Custom SetUpRule + RuleSequence (best way)

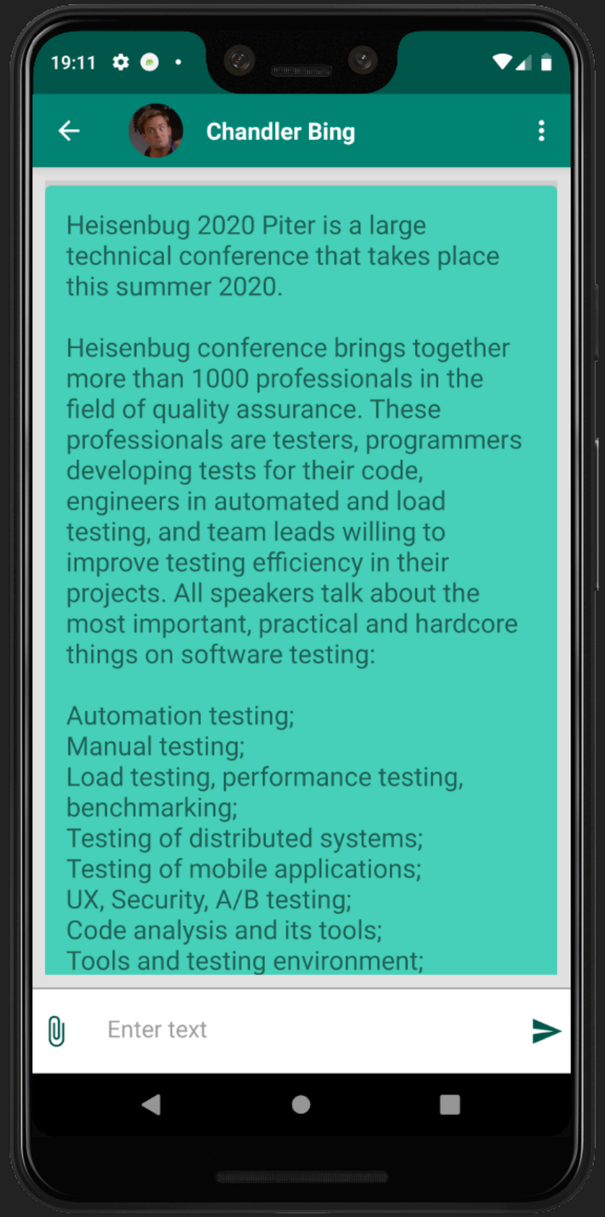


TEST DATA IS DEFINED IN @BEFORECLASS OR @BEFORE METHOD FOR ALL @TEST

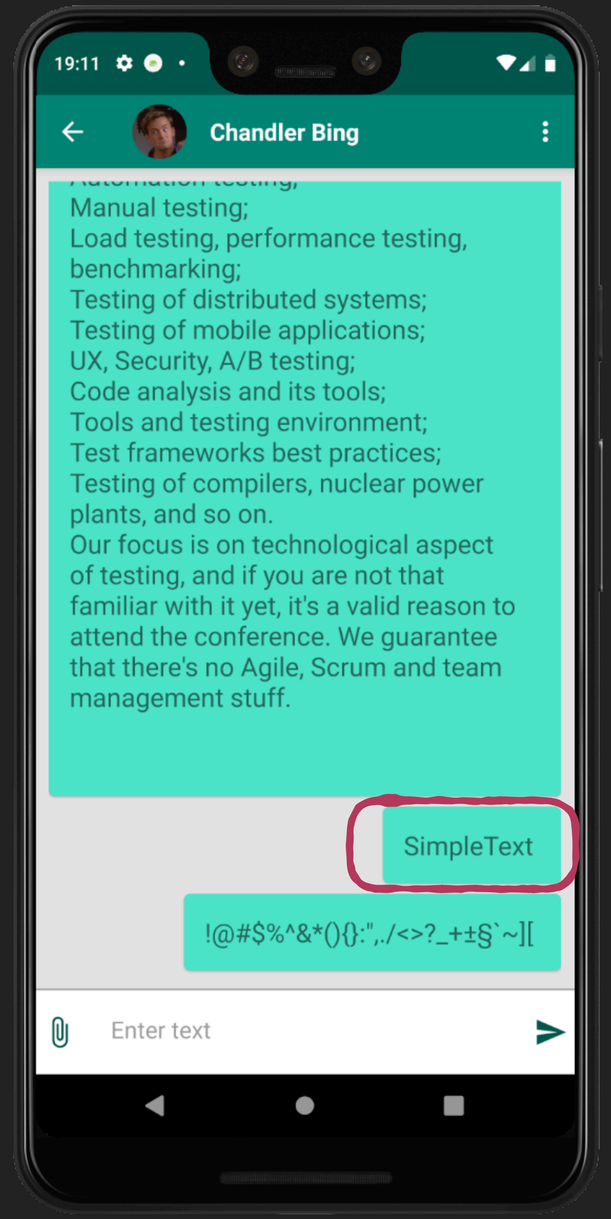
```
class SomeTestClass {
    companion object {
        @BeforeClass @JvmStatic
        fun setUpTestData(){
            Api.clearChatMessages(contact.id)
            Api.addChatMessage(contact.id, longMessage)
            Api.addChatMessage(contact.id, simpleMessage)
            Api.addChatMessage(contact.id, specialCharsMessage)
        }
    }
}
```

```
class SomeTestClass {
    @Before
    fun setUpTestData(){
        Api.clearChatMessages(contact.id)
        Api.addChatMessage(contact.id, longMessage)
        Api.addChatMessage(contact.id, simpleMessage)
        Api.addChatMessage(contact.id, specialCharsMessage)
    }
}
```

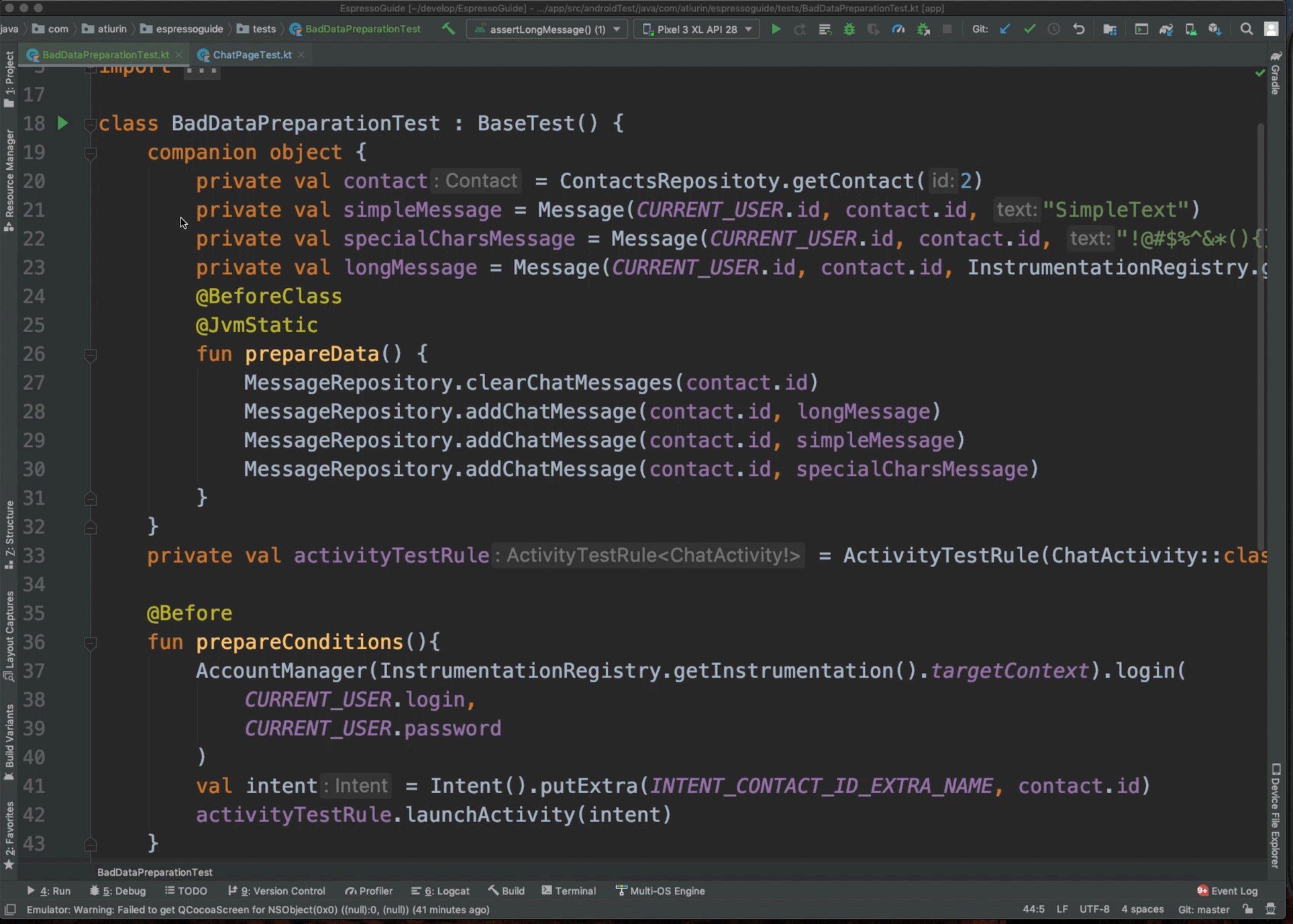
```
@Test
fun assertSimpleMessage() {
    ChatPage(contact)
        .assertMessageDisplayed(simpleMessage.text)
}
```



```
@Test
fun assertSimpleMessage() {
    ChatPage(contact)
        .assertMessageDisplayed(simpleMessage.text)
}
```




BAD WAY
DEMO


EspressoGuide [~/develop/EspressoGuide] - .../app/src/androidTest/java/com/atiurin/espressoguide/tests/BadDataPreparationTest.kt [app]
java com atiurin espressoguide tests BadDataPreparationTest
assertLongMessage() (1)
Pixel 3 XL API 28
Git:
BadDataPreparationTest.kt
ChatPageTest.kt
class BadDataPreparationTest : BaseTest() {
companion object {
private val contact : Contact = ContactsRepositoty.getContact(id: 2)
private val simpleMessage = Message(CURRENT_USER.id, contact.id, text: "SimpleText")
private val specialCharsMessage = Message(CURRENT_USER.id, contact.id, text: "!@#$%^&*(){
private val longMessage = Message(CURRENT_USER.id, contact.id, InstrumentationRegistry.
@BeforeClass
@JvmStatic
fun prepareData() {
MessageRepository.clearChatMessages(contact.id)
MessageRepository.addChatMessage(contact.id, longMessage)
MessageRepository.addChatMessage(contact.id, simpleMessage)
MessageRepository.addChatMessage(contact.id, specialCharsMessage)
}
}
private val activityTestRule : ActivityTestRule<ChatActivity!> = ActivityTestRule(ChatActivity::clas
@Before
fun prepareConditions(){
AccountManager(InstrumentationRegistry.getInstrumentation().targetContext).login(
CURRENT_USER.login,
CURRENT_USER.password
)
val intent : Intent = Intent().putExtra(INTENT_CONTACT_ID_EXTRA_NAME, contact.id)
activityTestRule.launchActivity(intent)
}
BadDataPreparationTest
4: Run
5: Debug
TODO
9: Version Control
Profiler
6: Logcat
Build
Terminal
Multi-OS Engine
Event Log
Emulator: Warning: Failed to get QCocoaScreen for NSObject(0x0) ((null):0, (null)) (41 minutes ago)
44:5 LF UTF-8 4 spaces Git: master
21:10
Espresso...

SOLUTION: CUSTOM SETUP RULE + @SETUP & @TEARDOWN ANNOTATIONS

```
class SomeTestClass {
    val setupRule = SetUpTearDownRule()
        .addSetUp { Api.clearChatMessages(contact.id) }
        .addSetUp("ADD_SIMPLE") { Api.addChatMessage(contact.id, simpleMessage) }
        .addSetUp("ADD_LONG") { Api.addChatMessage(contact.id, longMessage) }
}
```

SOLUTION: CUSTOM SETUP RULE + @SETUP & @TEARDOWN ANNOTATIONS

```
class SomeTestClass {
    val setupRule = SetUpTearDownRule()
        .addSetUp { Api.clearChatMessages(contact.id) }
        .addSetUp("ADD_SIMPLE") { Api.addChatMessage(contact.id, simpleMessage) }
        .addSetUp("ADD_LONG") { Api.addChatMessage(contact.id, longMessage) }
}
```

SOLUTION: CUSTOM SETUP RULE + @SETUP & @TEARDOWN ANNOTATIONS

```
class SomeTestClass {
    val setupRule = SetUpTearDownRule()
        .addSetUp { Api.clearChatMessages(contact.id) }
        .addSetUp("ADD_SIMPLE") { Api.addChatMessage(contact.id, simpleMessage) }
        .addSetUp("ADD_LONG") { Api.addChatMessage(contact.id, longMessage) }
}
```

SOLUTION: CUSTOM SETUP RULE + @SETUP & @TEARDOWN ANNOTATIONS

```
class SomeTestClass {
    val setupRule = SetUpTearDownRule()
        .addSetUp { Api.clearChatMessages(contact.id) }
        .addSetUp("ADD_SIMPLE") { Api.addChatMessage(contact.id, simpleMessage) }
        .addSetUp("ADD_LONG") { Api.addChatMessage(contact.id, longMessage) }
}
```

SOLUTION: CUSTOM SETUP RULE + @SETUP & @TEARDOWN ANNOTATIONS

```
class SomeTestClass {
    val setupRule = SetUpTearDownRule()
        .addSetUp { Api.clearChatMessages(contact.id) }
        .addSetUp("ADD_SIMPLE") { Api.addChatMessage(contact.id, simpleMessage) }
        .addSetUp("ADD_LONG") { Api.addChatMessage(contact.id, longMessage) }
}
```

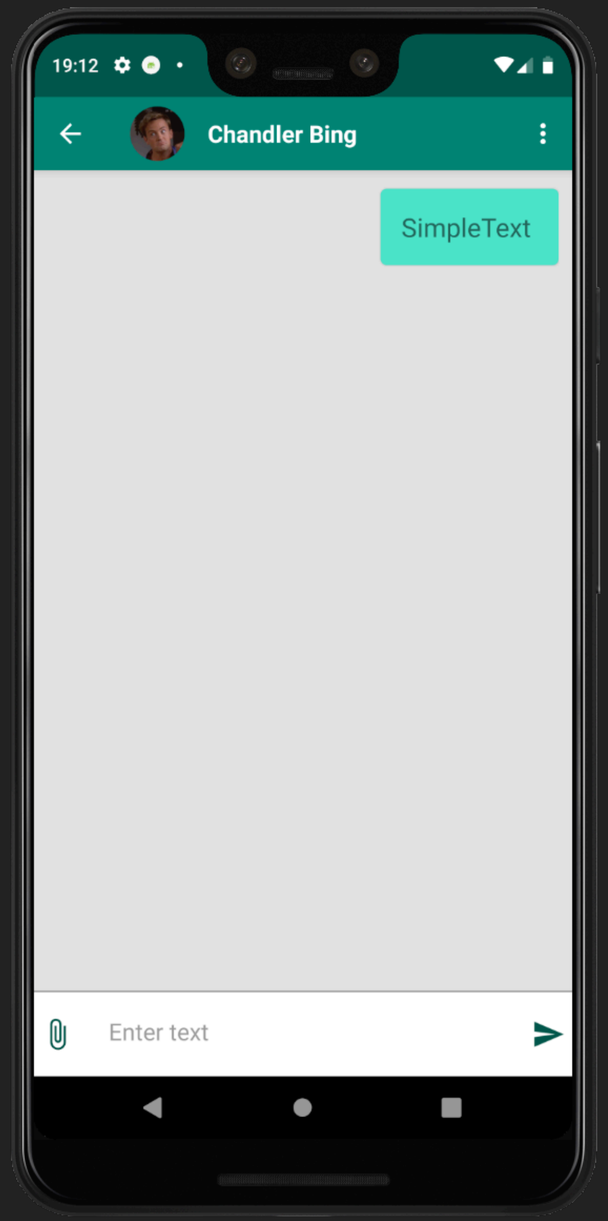
SOLUTION: CUSTOM SETUP RULE + @SETUP & @TEARDOWN ANNOTATIONS

```
class SomeTestClass {
    val setupRule = SetUpTearDownRule()
        .addSetUp { Api.clearChatMessages(contact.id) }
        .addSetUp("ADD_SIMPLE") { Api.addChatMessage(contact.id, simpleMessage) }
        .addSetUp("ADD_LONG") { Api.addChatMessage(contact.id, longMessage) }
}
```

SOLUTION: CUSTOM SETUP RULE + @SETUP & @TEARDOWN ANNOTATIONS

```
class SomeTestClass {
    val setupRule = SetUpTearDownRule()
        .addSetUp { Api.clearChatMessages(contact.id) }
        .addSetUp("ADD_SIMPLE") { Api.addChatMessage(contact.id, simpleMessage) }
        .addSetUp("ADD_LONG") { Api.addChatMessage(contact.id, longMessage) }

    @SetUp("ADD_SIMPLE")
    @Test
    fun assertSimpleMessage() {
        ChatPage(contact).assertMessageDisplayed(simpleMessage.text)
    }
}
```

```
@SetUp("ADD_SIMPLE")
@Test
fun assertSimpleMessage() {
    ChatPage(contact)
        .assertMessageDisplayed(simpleMessage.text)
}
```



SOLUTION: CUSTOM SETUP RULE + @SETUP & @TEARDOWN ANNOTATIONS

```
class SomeTestClass {

    ...
    .addSetUp("ADD_SIMPLE") { Api.addChatMessage(contact.id, simpleMessage) }
    .addSetUp("ADD_LONG") { Api.addChatMessage(contact.id, longMessage) }

    @SetUp("ADD_SIMPLE")
    @Test
    fun assertBothMessagesDisplayed() {}
}
```

SOLUTION: CUSTOM SETUP RULE + @SETUP & @TEARDOWN ANNOTATIONS

```
class SomeTestClass {

    ...
    .addSetUp("ADD_SIMPLE") { Api.addChatMessage(contact.id, simpleMessage) }
    .addSetUp("ADD_LONG") { Api.addChatMessage(contact.id, longMessage) }

    @SetUp("ADD_SIMPLE", "ADD_LONG")
    @Test
    fun assertBothMessagesDisplayed() {}
}
```

SOLUTION: CUSTOM SETUP RULE + @SETUP & @TEARDOWN ANNOTATIONS

```
class SomeTestClass {

    ...
    .addSetUp("ADD_SIMPLE") { Api.addChatMessage(contact.id, simpleMessage) }
    .addSetUp("ADD_LONG") { Api.addChatMessage(contact.id, longMessage) }

    @SetUp("ADD_SIMPLE", "ADD_LONG")
    @Test
    fun assertBothMessagesDisplayed() {
        ChatPage(contact)
            .assertMessageDisplayed(simpleMessage.text)
            .assertMessageDisplayed(longMessage.text)
    }
}
```


GOOD WAY
DEMO

EspressoGuide [~/develop/EspressoGuide] - .../app/src/androidTest/java/com/atiurin/espressoguide/tests/ChatPageTest.kt [app]

java › com › atiurin › espressoguide › tests › ChatPageTest | BadDataPreparationTest | Pixel 3 XL API 28 | Git:

BadDataPreparationTest.kt | ChatPageTest.kt

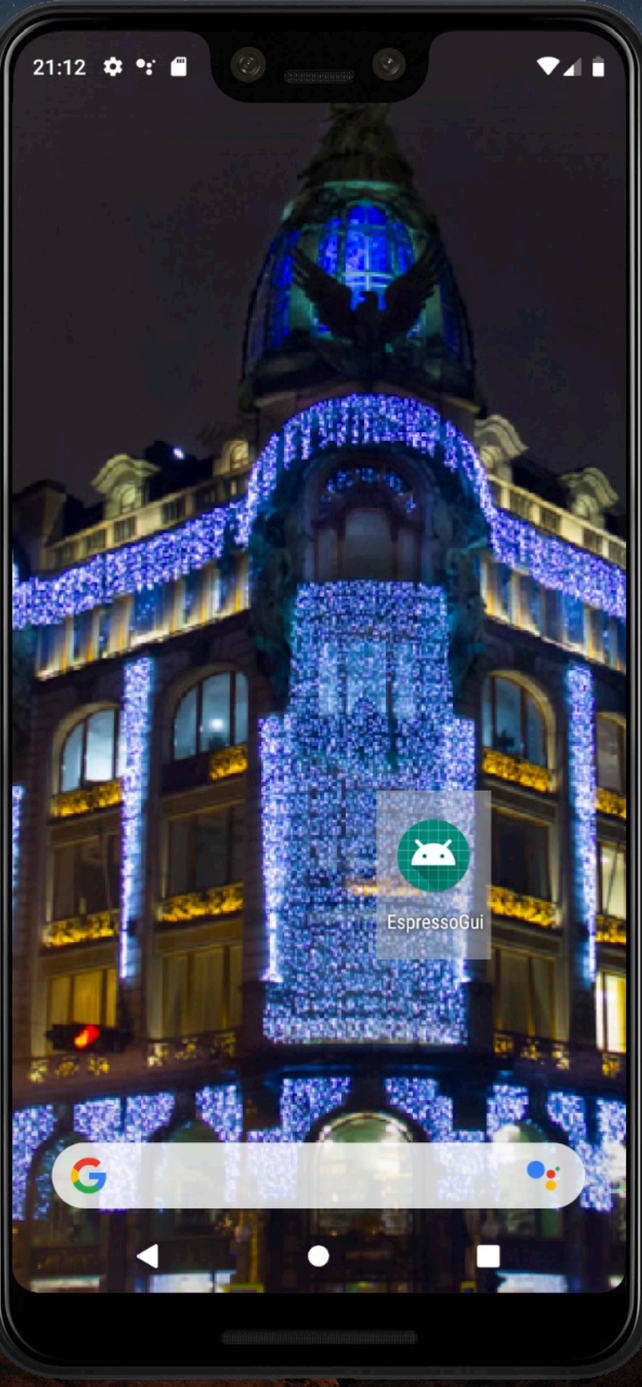
```kotlin
package com.atiurin.espressoguide.tests

import ...

class ChatPageTest : BaseTest() {
    companion object {...}
    private val contact : Contact  = ContactsRepositoty.getContact( id: 2)
    private val simpleMessage = Message(CURRENT_USER.id, contact.id,  text: "SimpleText")
    private val specialCharsMessage = Message(CURRENT_USER.id, contact.id,  text: "!@#$%^&*(){}:\",./<>?_+±§`~][")
    private val longMessage = Message(CURRENT_USER.id, contact.id, InstrumentationRegistry.getInstrumentation().context.assets

    private val activityTestRule : ActivityTestRule<ChatActivity!> = ActivityTestRule(ChatActivity::class.java,  initialTouchMode: false,
    private val setUpTearDownRule : SetUpTearDownRule  = SetUpTearDownRule()
        .addSetUp { MessageRepository.clearChatMessages(contact.id) }
        .addSetUp(ADD_SIMPLE_MESSAGE) {
            MessageRepository.addChatMessage(contact.id, simpleMessage)
        }
        .addSetUp(ADD_SPECIAL_CHARS_MESSAGE) {
            MessageRepository.addChatMessage(contact.id, specialCharsMessage)
        }
        .addSetUp(ADD_LONG_MESSAGE) {
            MessageRepository.addChatMessage(contact.id, longMessage)
        }
        .addSetUp {
            AccountManager(InstrumentationRegistry.getInstrumentation().targetContext).login(
                CURRENT_USER.login,
                CURRENT_USER.password
            )
            val intent : Intent  = Intent().putExtra(INTENT_CONTACT_ID_EXTRA_NAME, contact.id)
            activityTestRule.launchActivity(intent)
        }

    init {
```

ChatPageTest

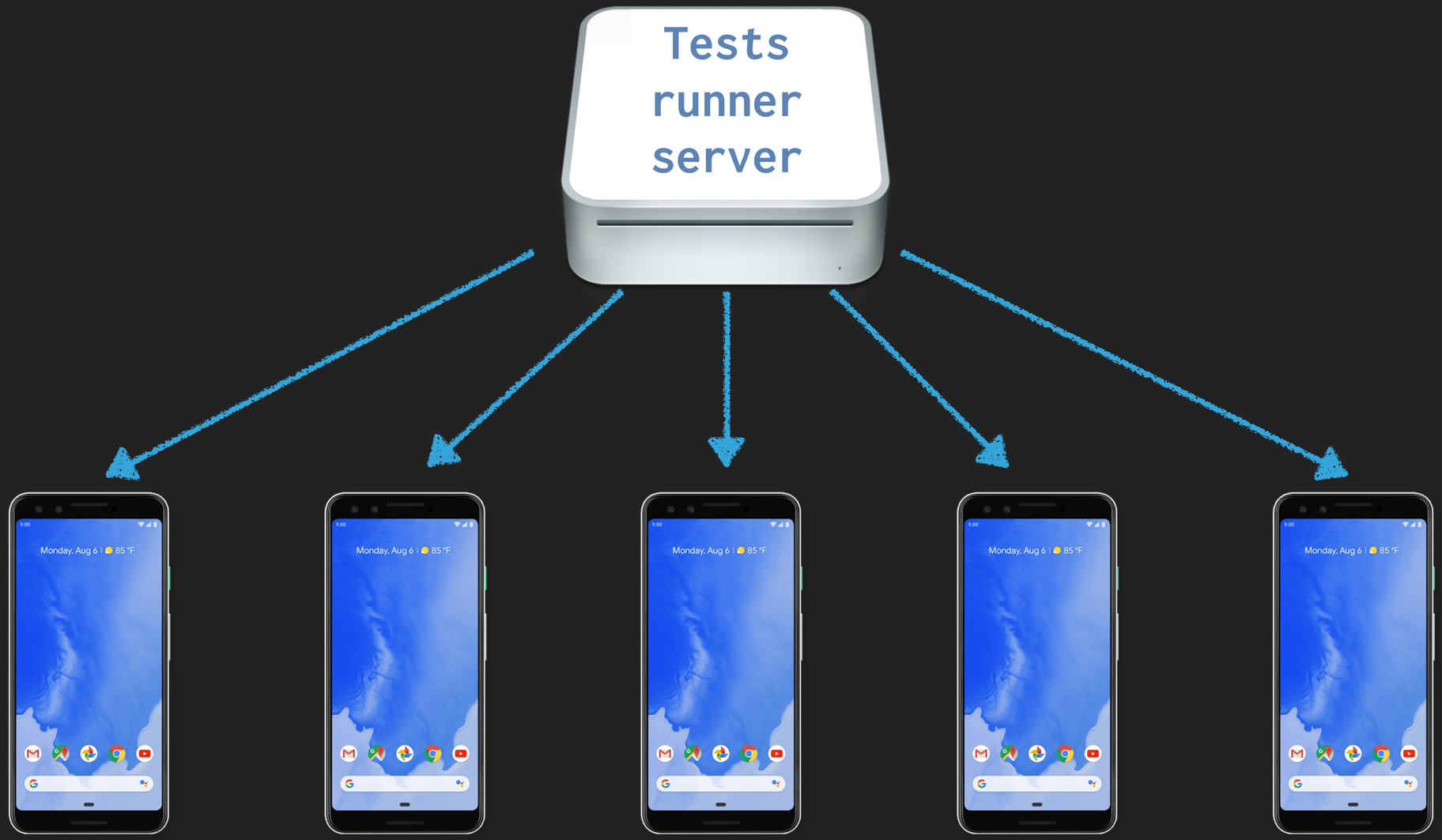
4: Run | 5: Debug | TODO | 9: Version Control | Profiler | 6: Logcat | Build | Terminal | Multi-OS Engine | Event Log

Tests passed: 3 (a minute ago)

21:34 LF UTF-8 4 spaces Git: master



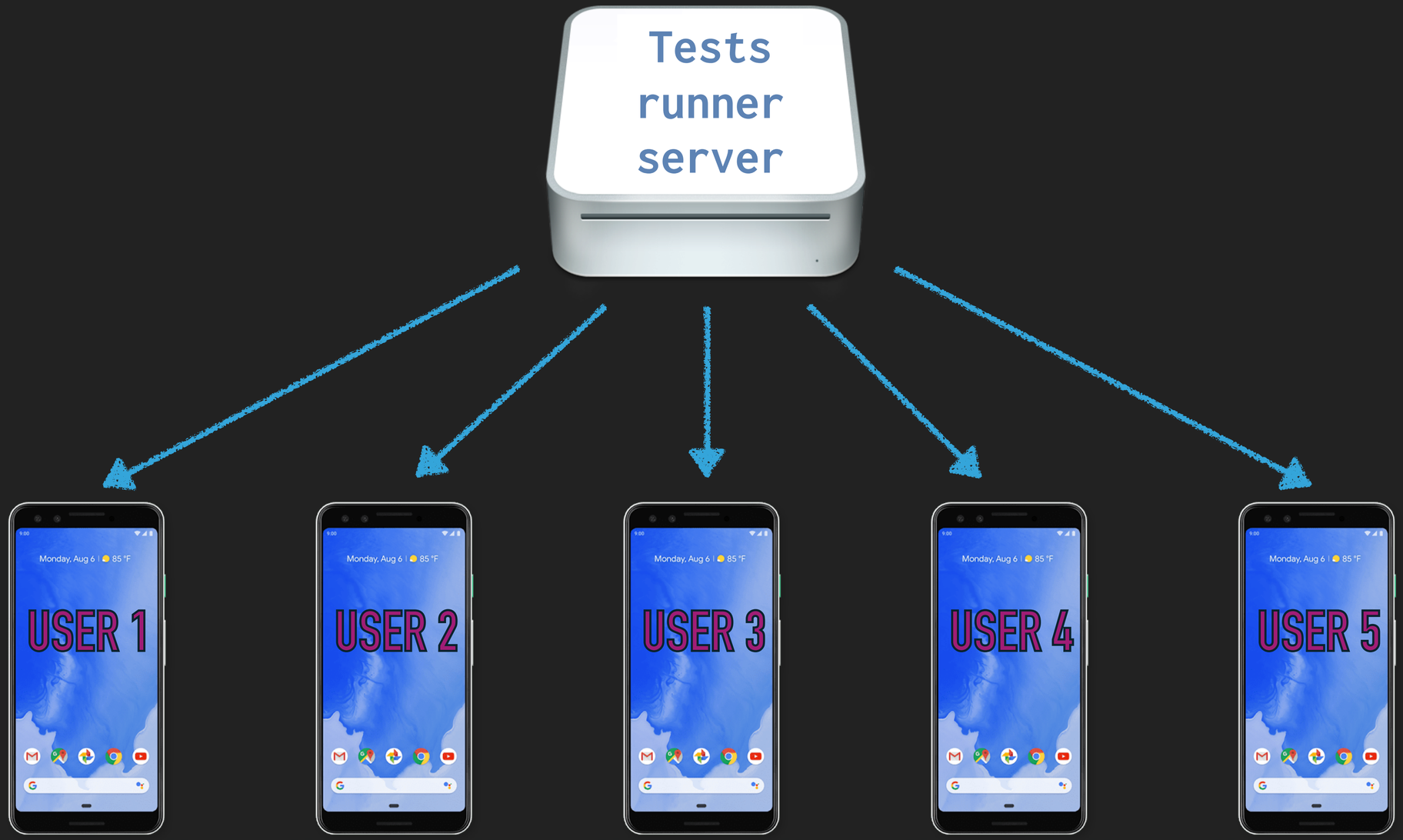
TEST DATA IS RELATED WITH BUILD.SERIAL

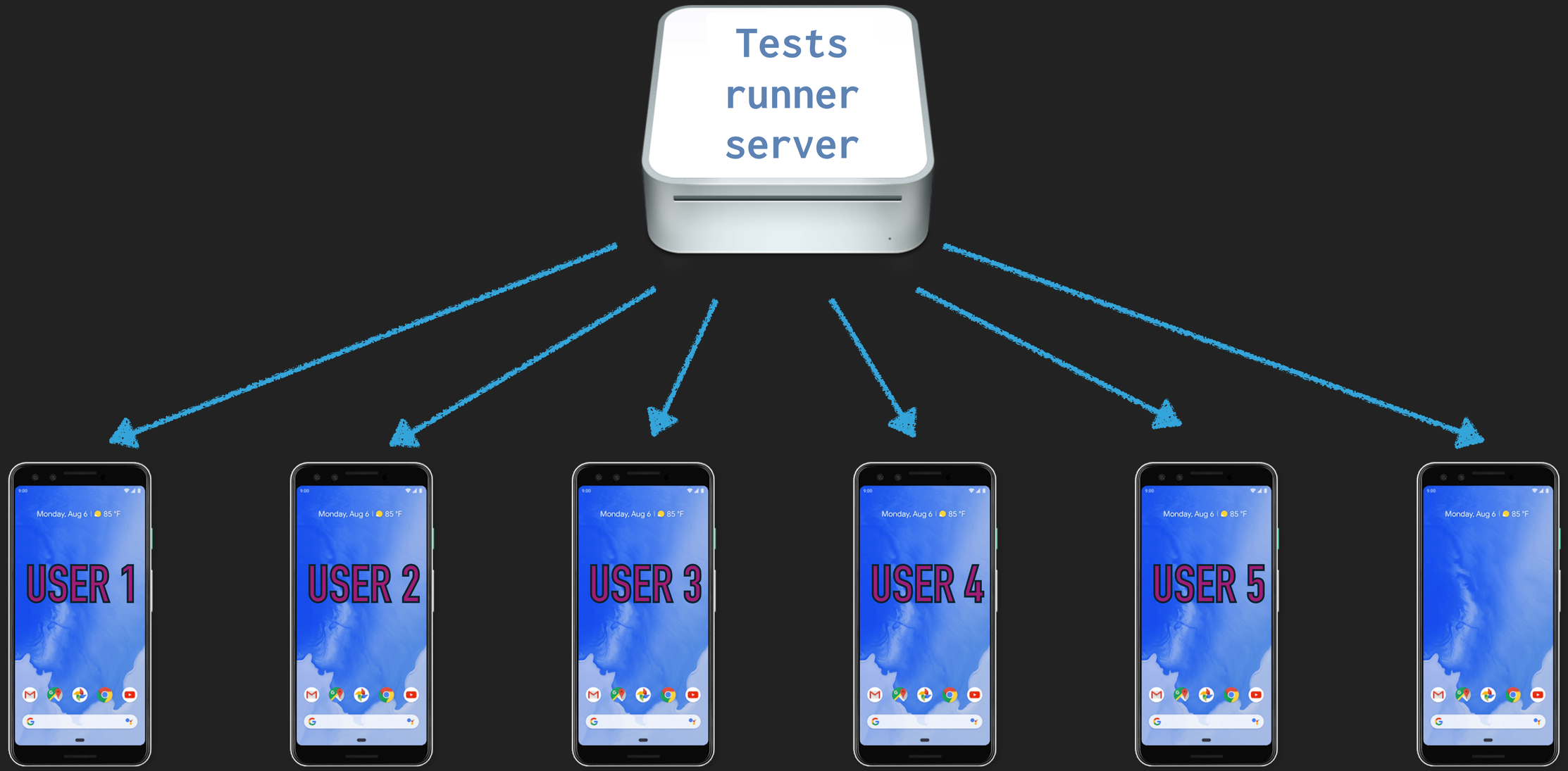


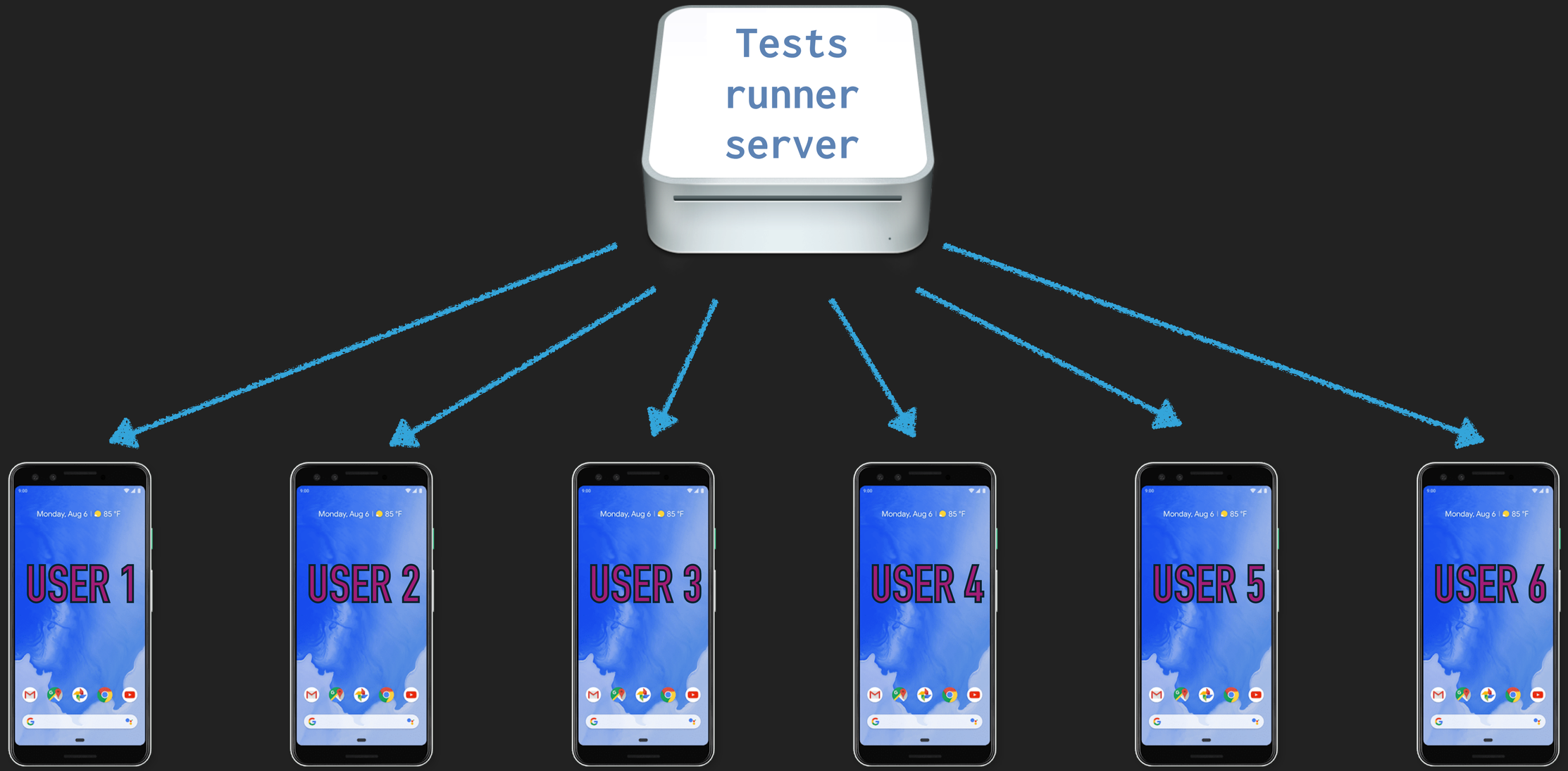
```
class SomeTestClass {
    ...
    val user = TestData.getUser(Build.Serial)
}
```


Tests
runner
server
Monday, Aug 6 | 85 °F
Monday, Aug 6 | 85 °F
Monday, Aug 6 | 85 °F
Monday, Aug 6 | 85 °F
Monday, Aug 6 | 85 °F

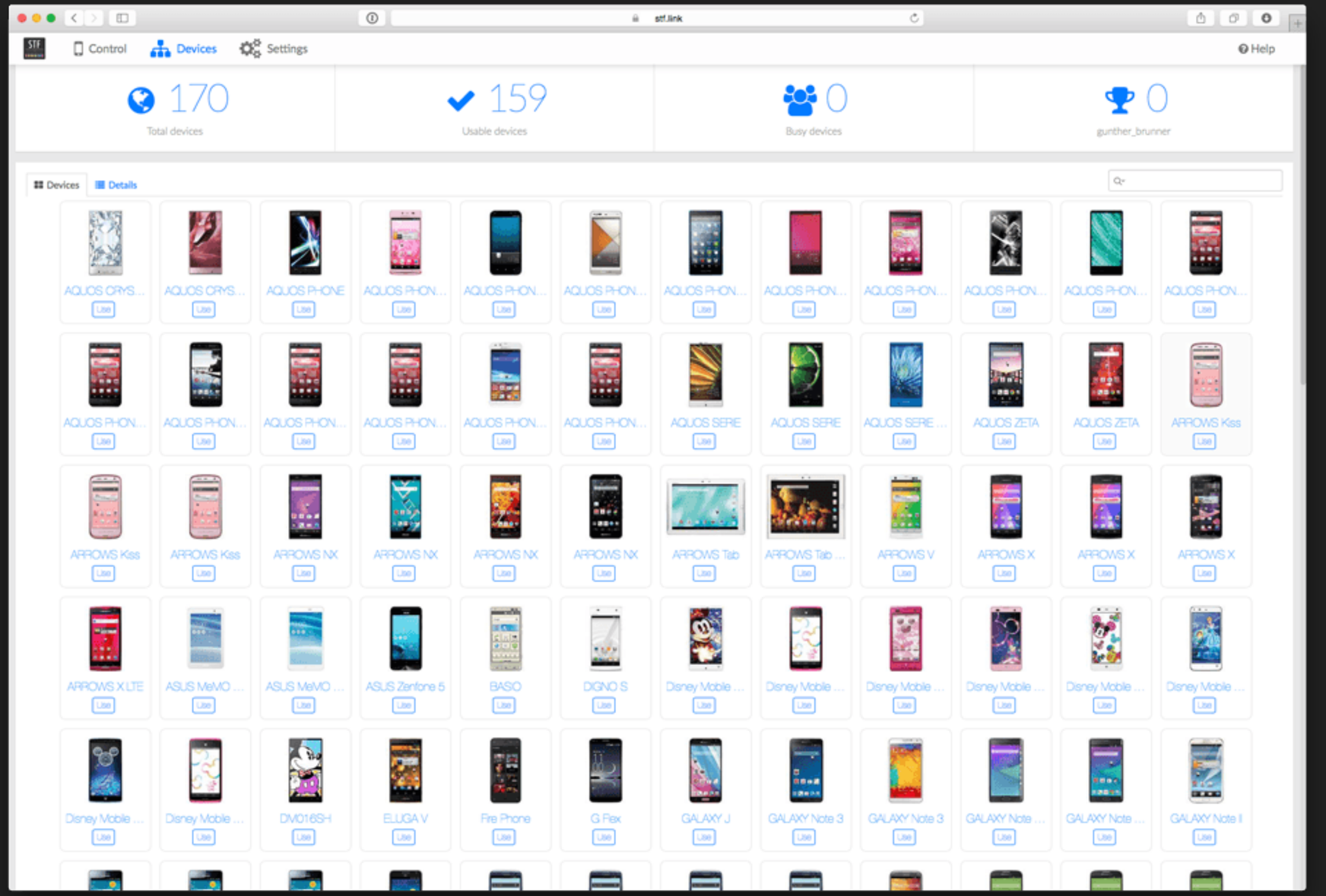

Tests
runner
server
Monday, Aug 6 | 85 °F
USER 1
Monday, Aug 6 | 85 °F
Monday, Aug 6 | 85 °F
Monday, Aug 6 | 85 °F
Monday, Aug 6 | 85 °F


Tests
runner
server
USER 1
USER 2
USER 3
USER 4
USER 5


Tests
runner
server
USER 1
USER 2
USER 3
USER 4
USER 5


Tests
runner
server
USER 1
USER 2
USER 3
USER 4
USER 5
USER 6

TEST DATA IS RELATED WITH BUILD.SERIAL



https://openstf.io/

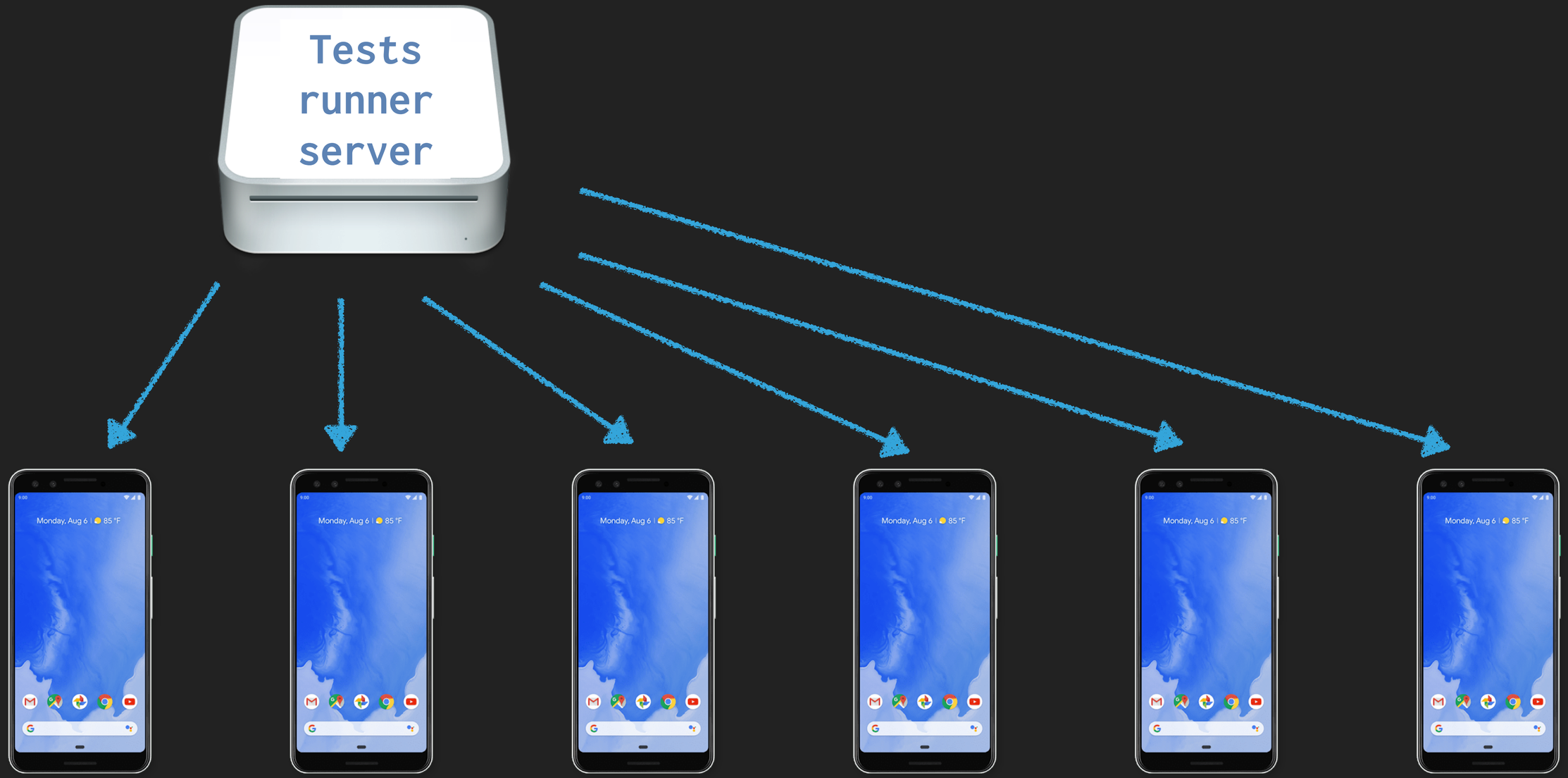
IF WE WANNA ADD OR CHANGE A DEVICE

IF WE WANNA ADD OR CHANGE A DEVICE

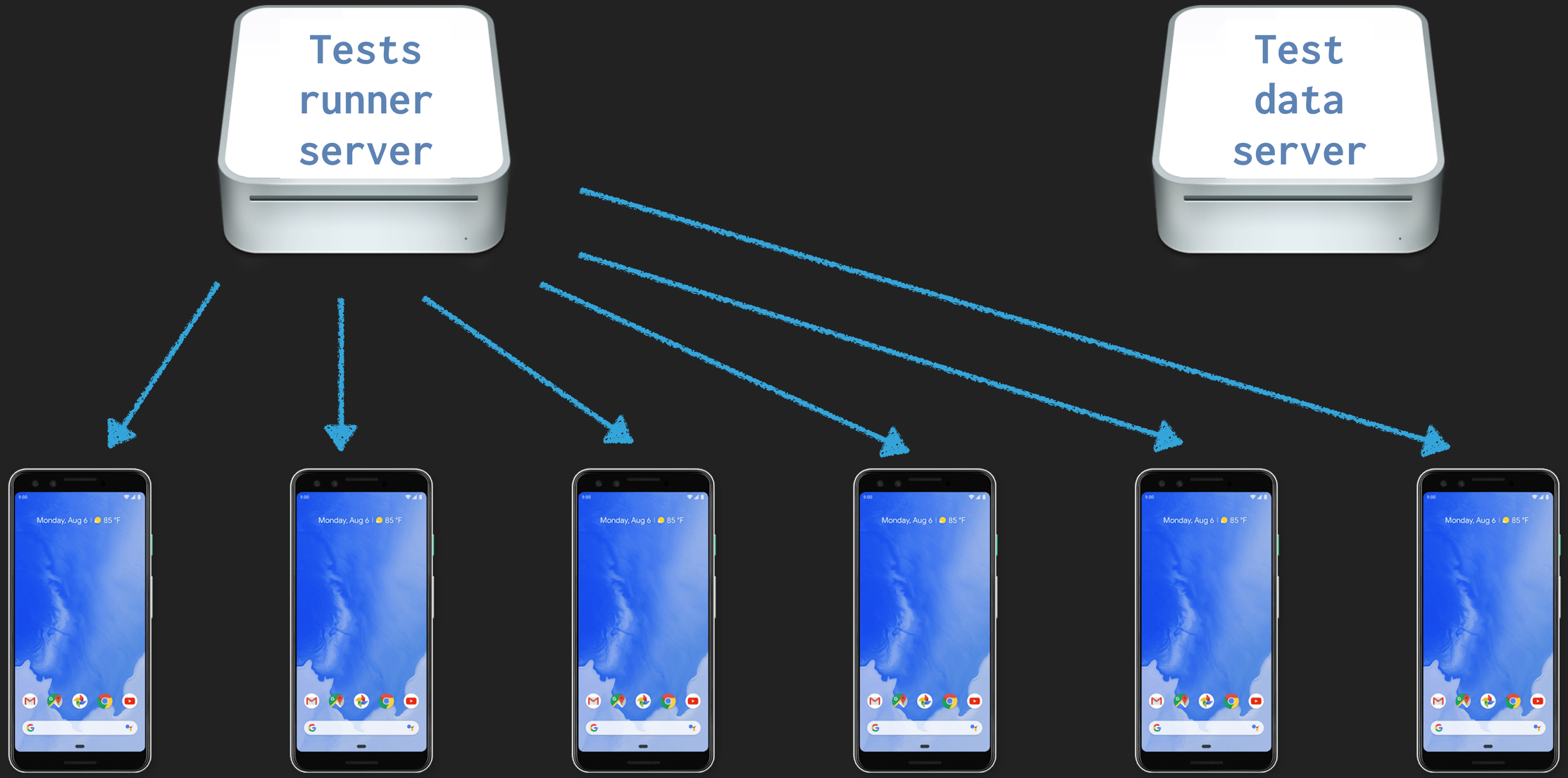
- Support changing in test code

IF WE WANNA ADD OR CHANGE A DEVICE

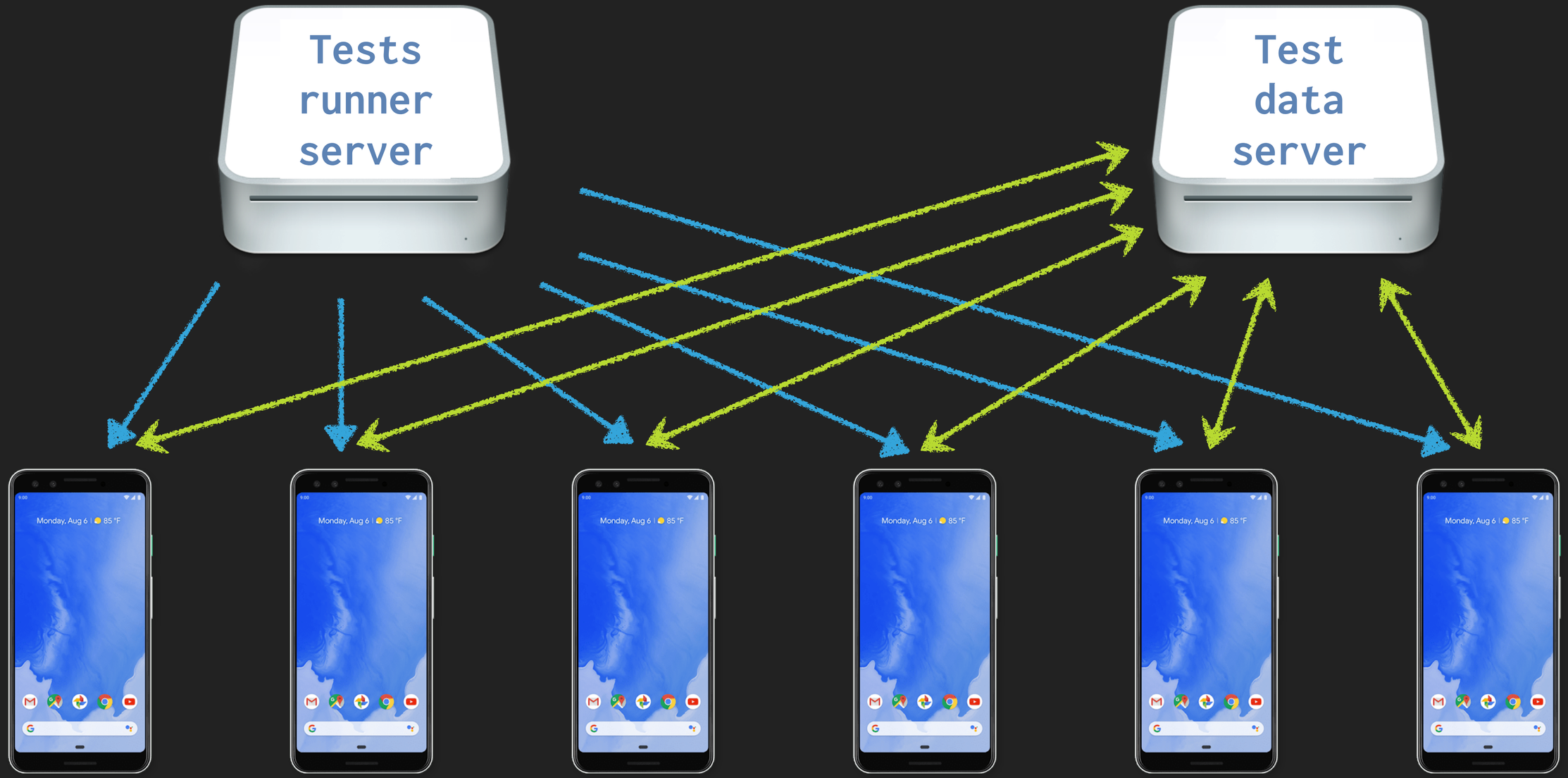
- Support changing in test code
- Configure test user to default test conditions

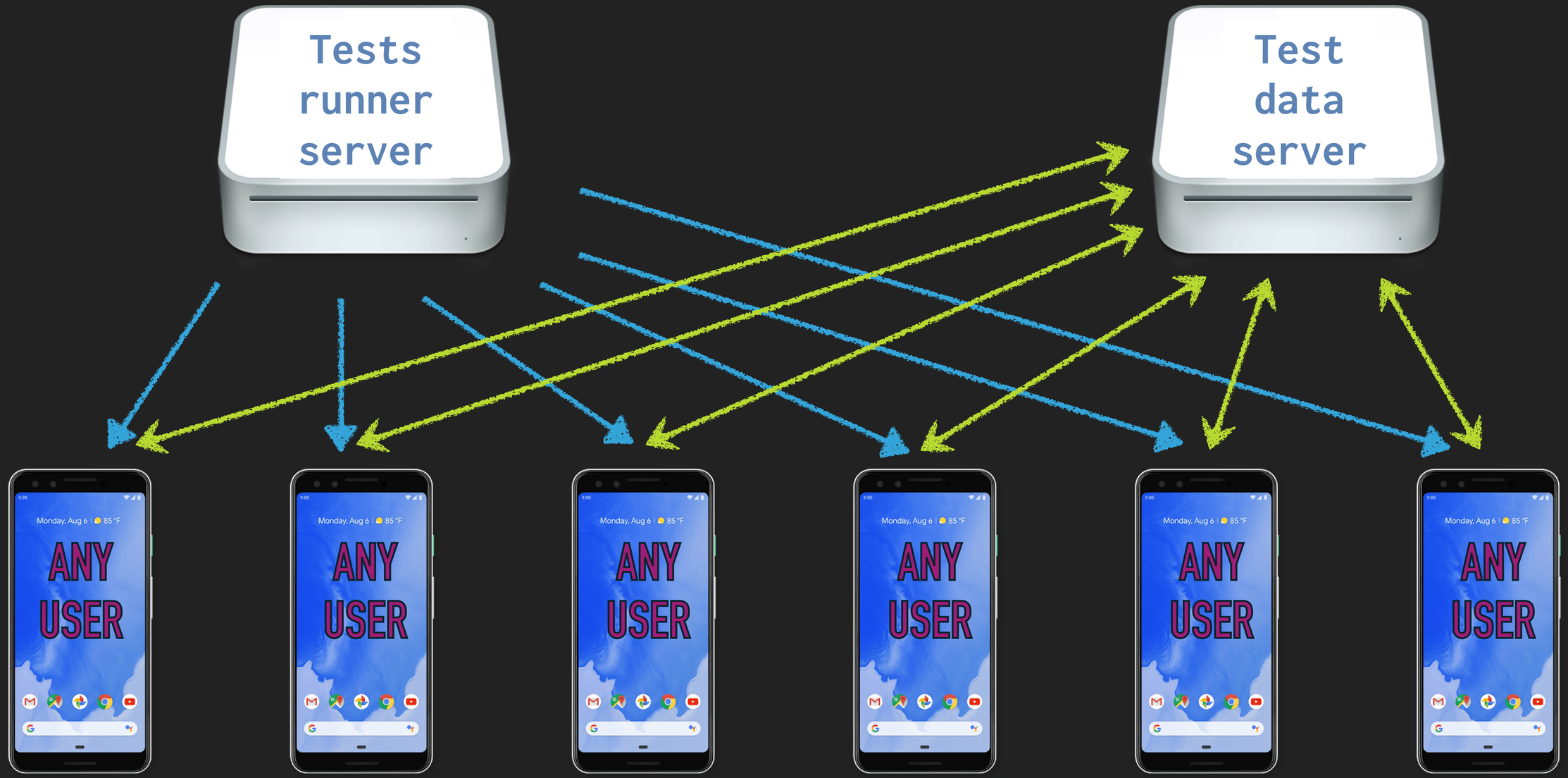

Tests
runner
server
Monday, Aug 6 | 85 °F

TEST DATA IS RELATED WITH BUILD.SERIAL



TEST DATA IS RELATED WITH BUILD.SERIAL




Tests runner server
Test data server
ANY USER
ANY USER
ANY USER
ANY USER
ANY USER
ANY USER


Test
data
server

- Pool of test users and any other required objects

- Pool of test users and any other required objects
- Data management via rest api

- Pool of test users and any other required objects
- Data management via rest api
- Guarantee of test data uniqueness for separate test

- Kotlin
- Spring Boot (https://spring.io/projects/spring-boot)
- JetBrains Exposed (https://github.com/JetBrains/Exposed)

SOLUTION: TEST DATA SERVER

SOLUTION: TEST DATA SERVER

- No need to support devices

SOLUTION: TEST DATA SERVER

- No need to support devices
- Single point to control all test data

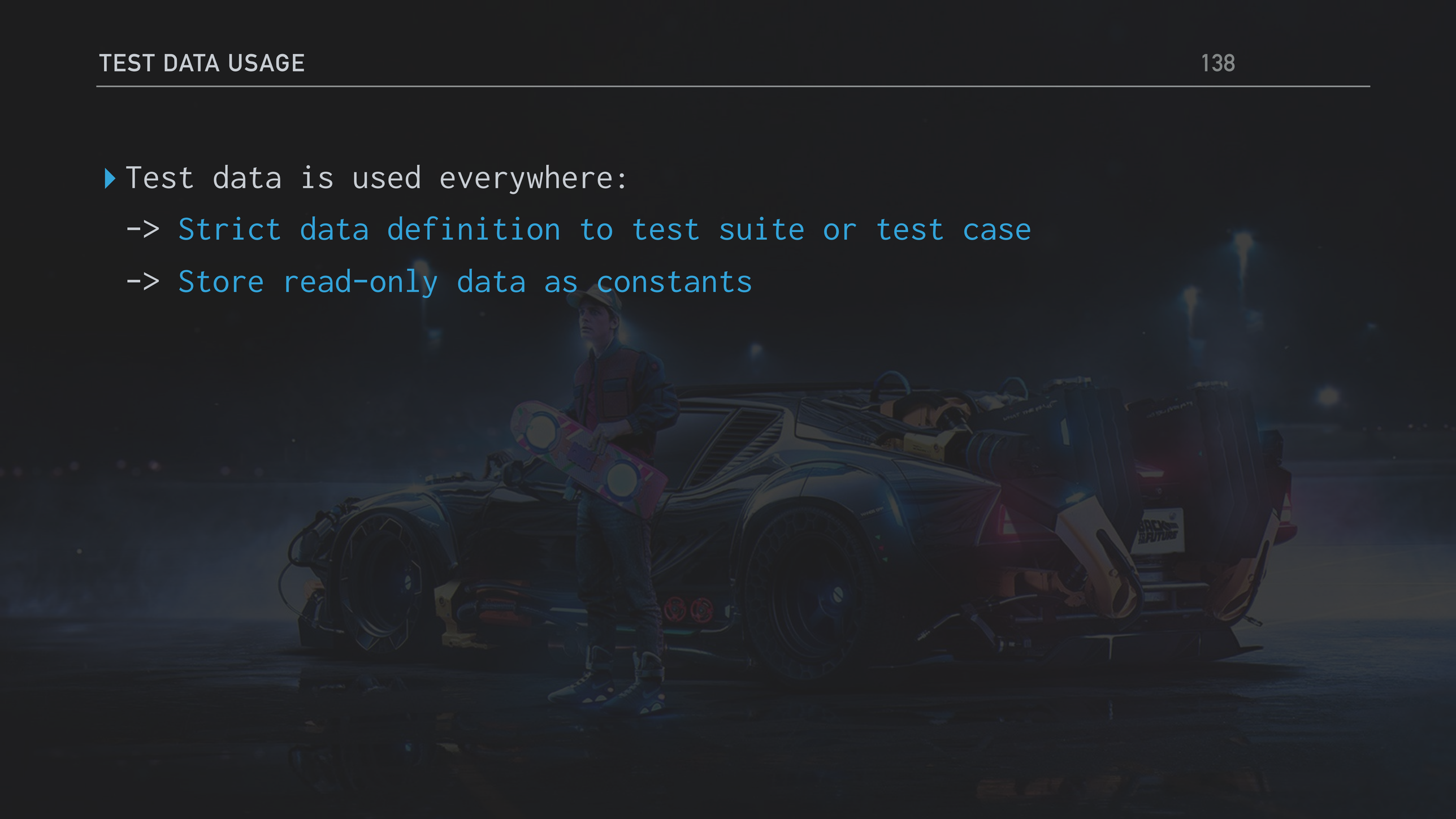
SOLUTION: TEST DATA SERVER

- No need to support devices
- Single point to control all test data
- Multiple test runs at the same time (CI tasks)

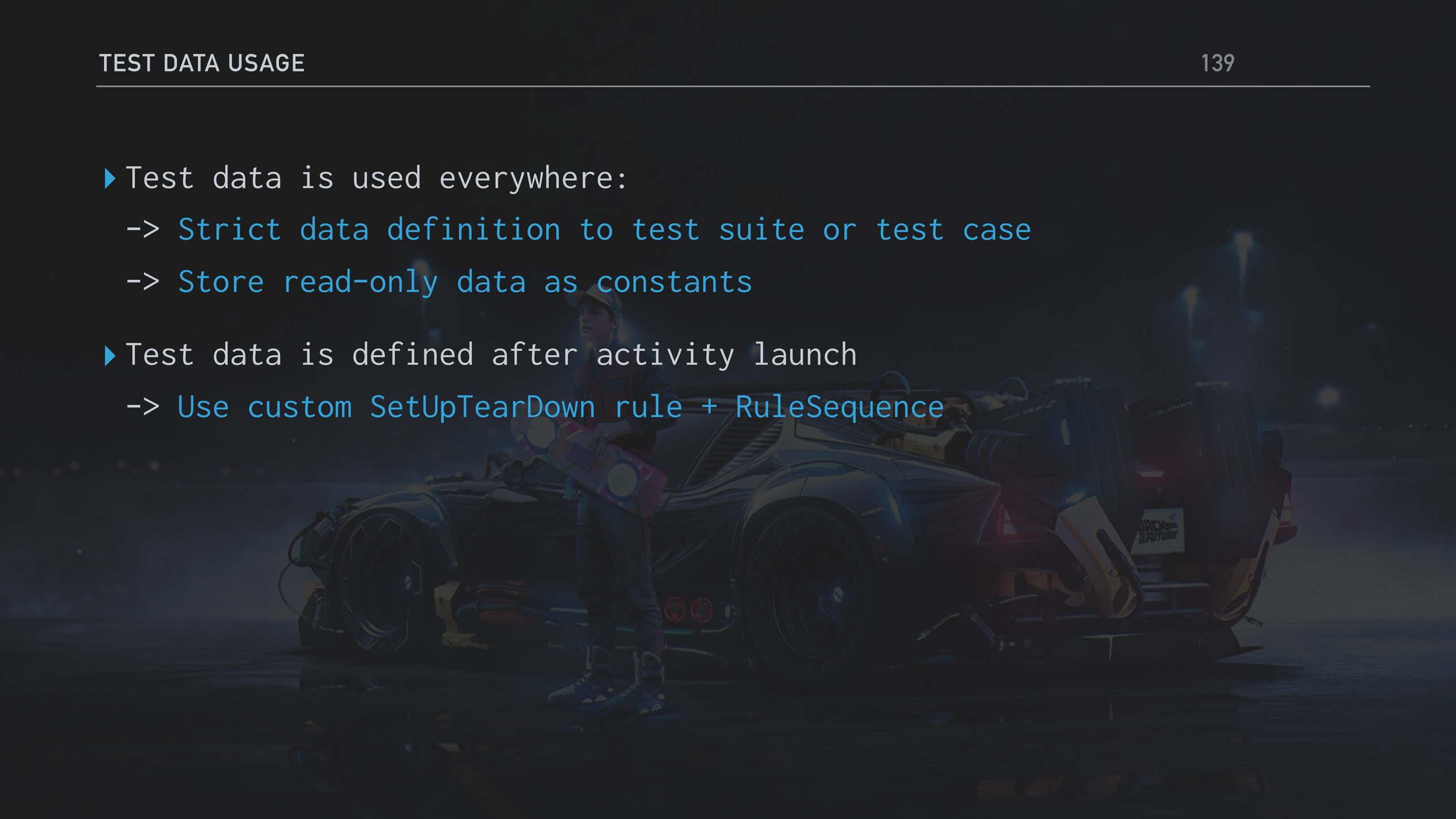
SOLUTION: TEST DATA SERVER

- No need to support devices
- Single point to control all test data
- Multiple test runs at the same time (CI tasks)
- It could be used by other test frameworks (iOS, Web, api tests)

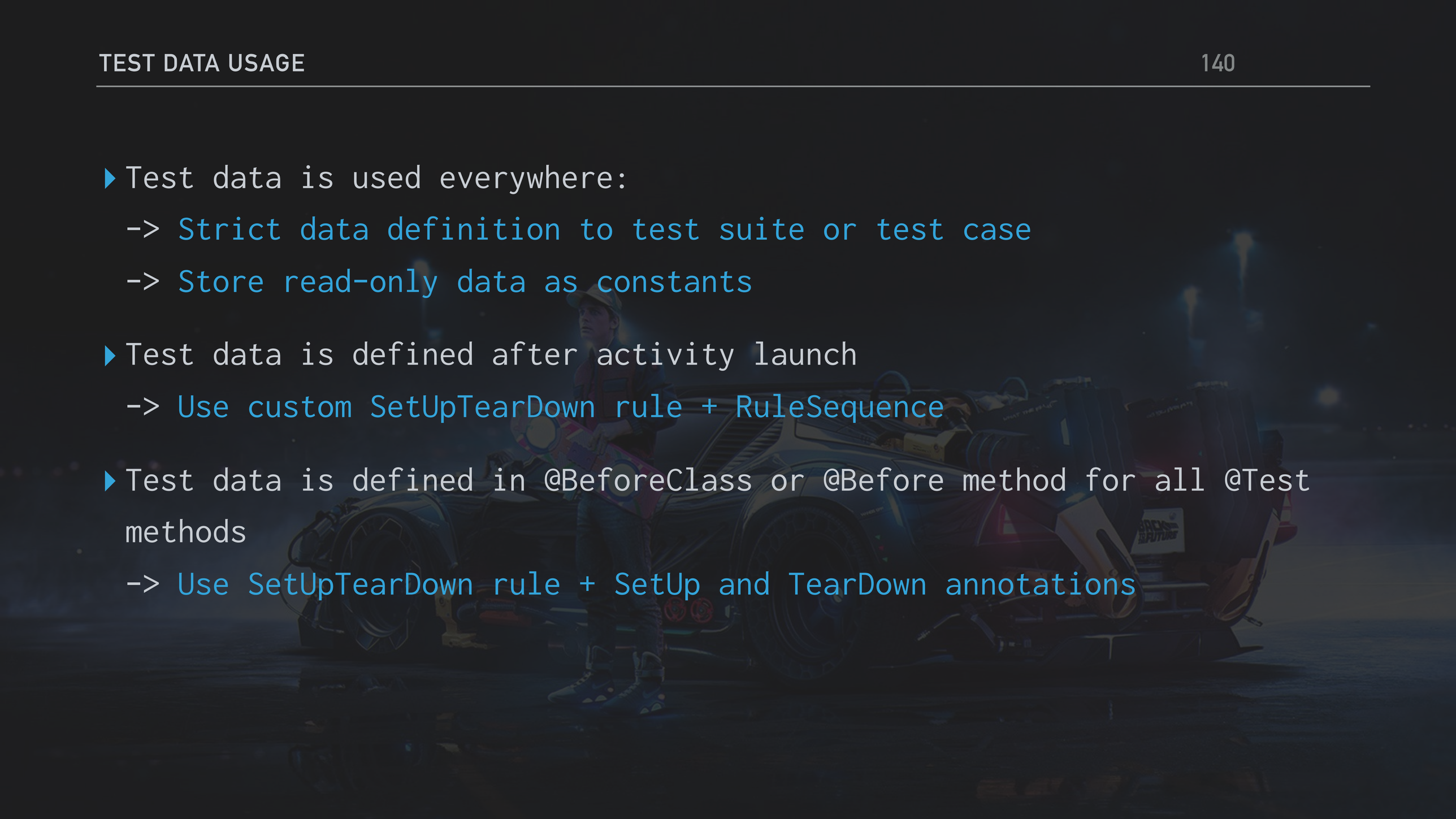
- Test data is used everywhere:
  -> Strict data definition to test suite or test case
  -> Store read-only data as constants



- Test data is used everywhere:
  -> Strict data definition to test suite or test case
  -> Store read-only data as constants
- Test data is defined after activity launch
  -> Use custom SetUpTearDown rule + RuleSequence

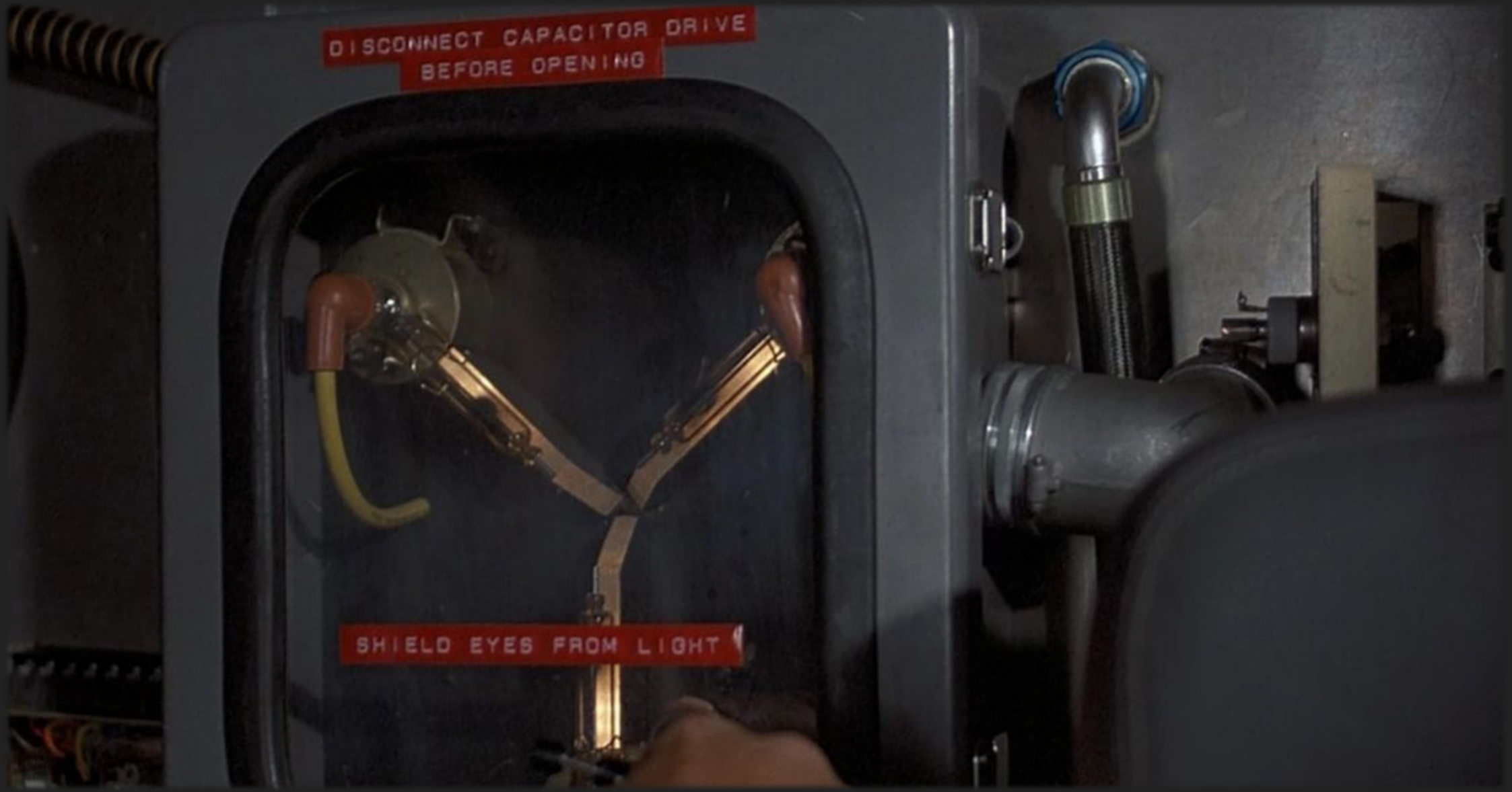


- Test data is used everywhere:
  -> Strict data definition to test suite or test case
  -> Store read-only data as constants
- Test data is defined after activity launch
  -> Use custom SetUpTearDown rule + RuleSequence
- Test data is defined in @BeforeClass or @Before method for all @Test methods
  -> Use SetUpTearDown rule + SetUp and TearDown annotations

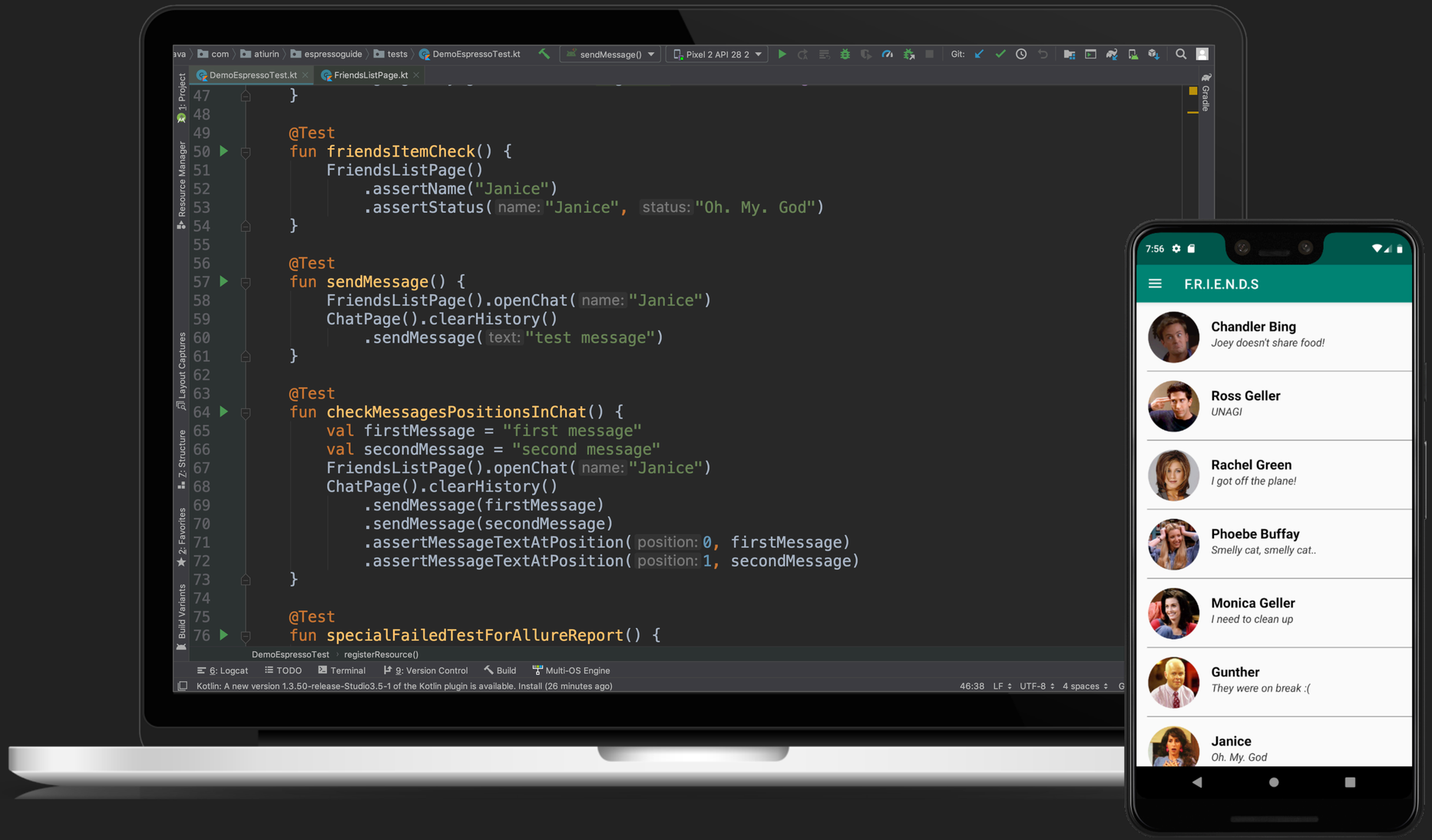


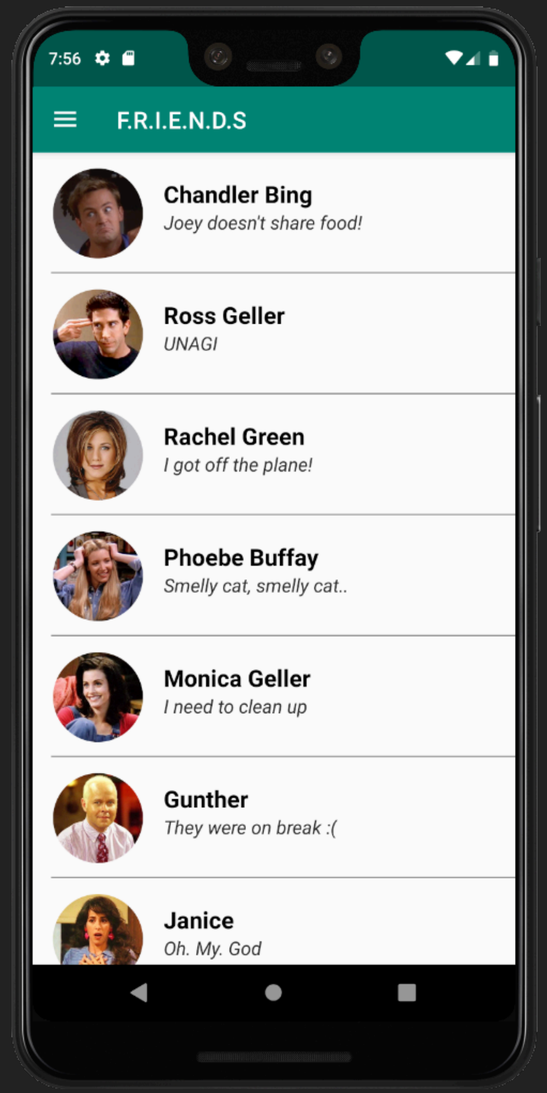
- Test data is used everywhere:
  -> Strict data definition to test suite or test case
  -> Store read-only data as constants
- Test data is defined after activity launch
  -> Use custom SetUpTearDown rule + RuleSequence
- Test data is defined in @BeforeClass or @Before method for all @Test methods
  -> Use SetUpTearDown rule + SetUp and TearDown annotations
- Test user is related with device Build.Serial
  -> Use TestData server

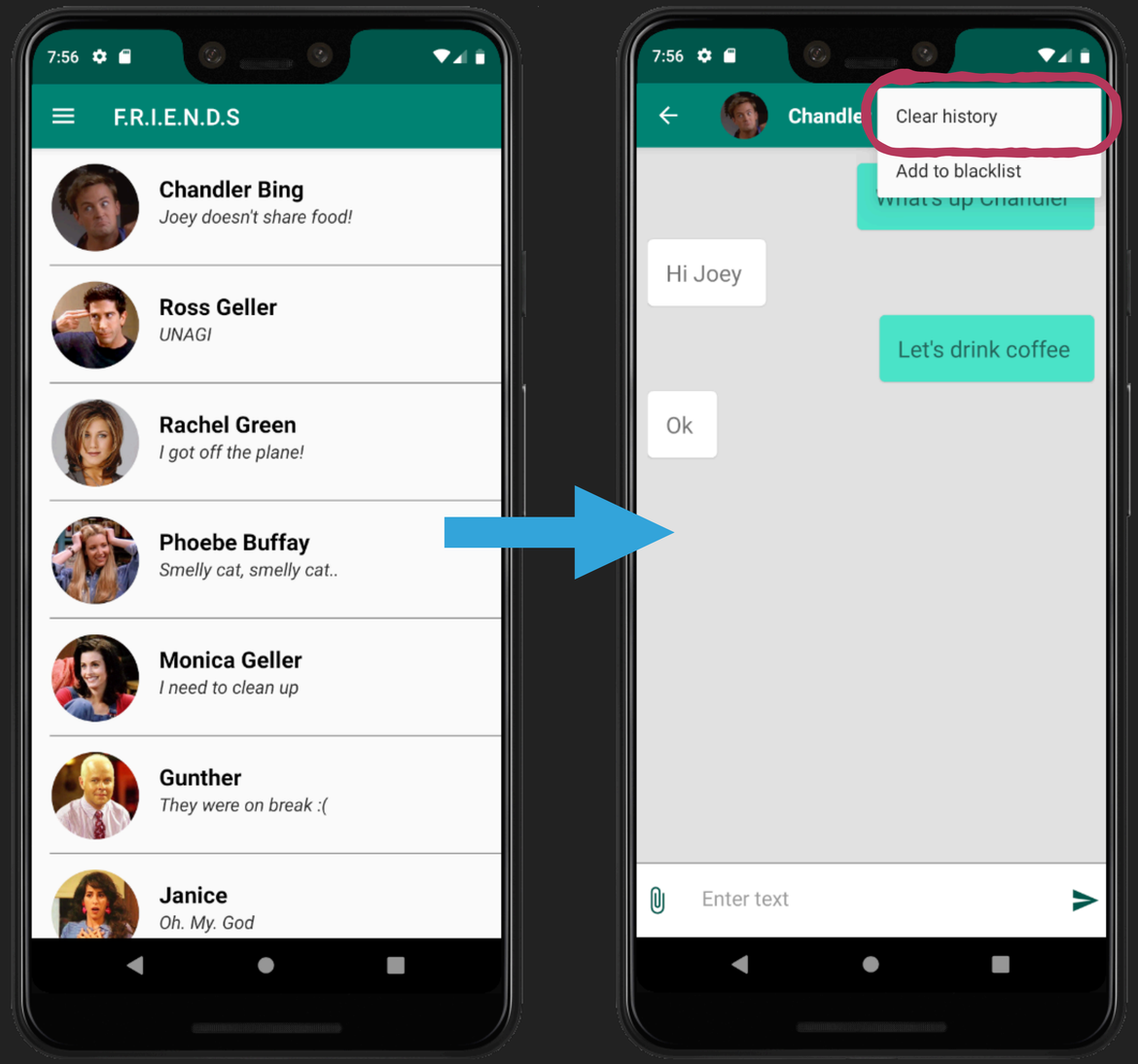
TEST FLOW MISTAKES



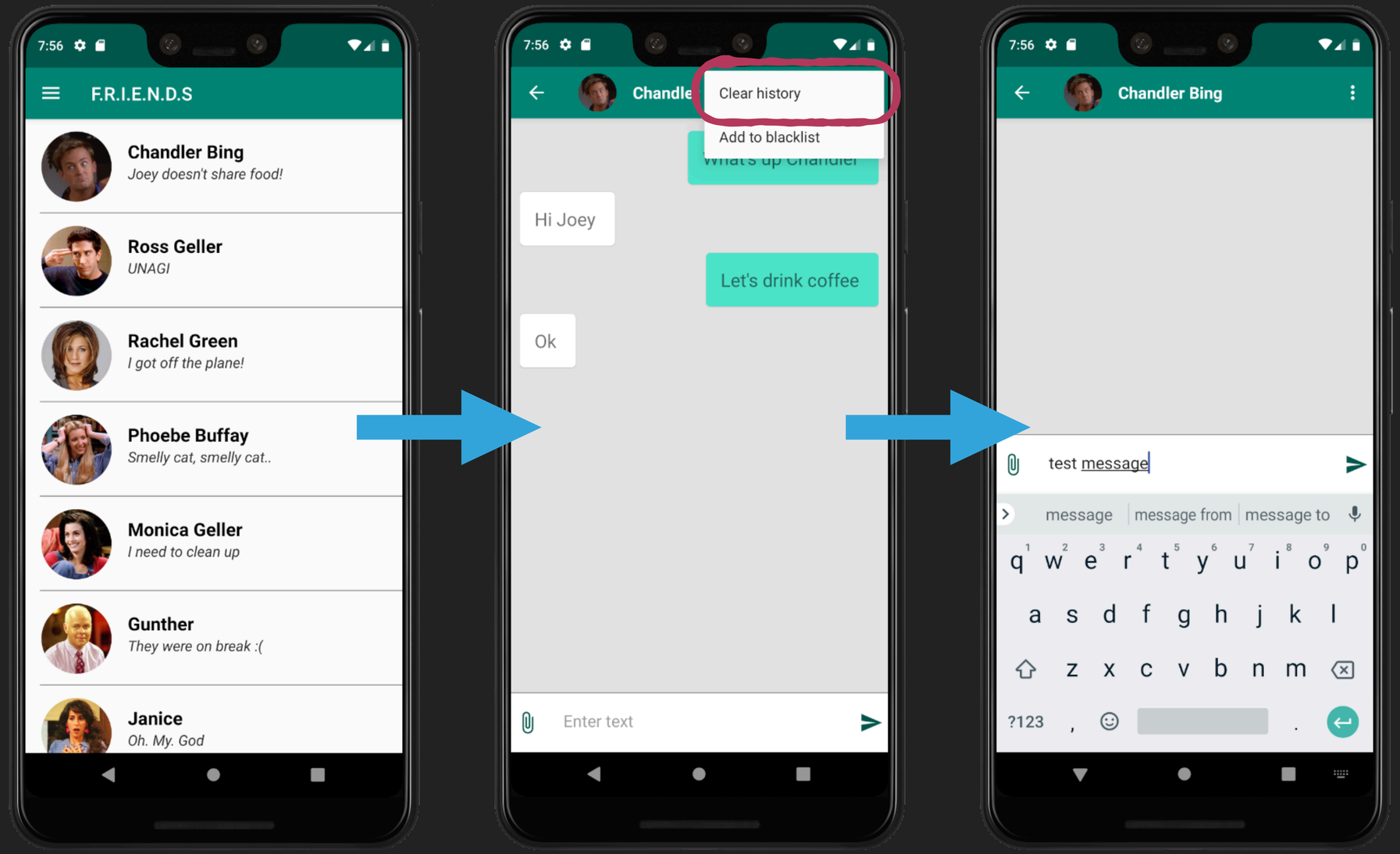
PREVIOUSLY ON HEISENBUG

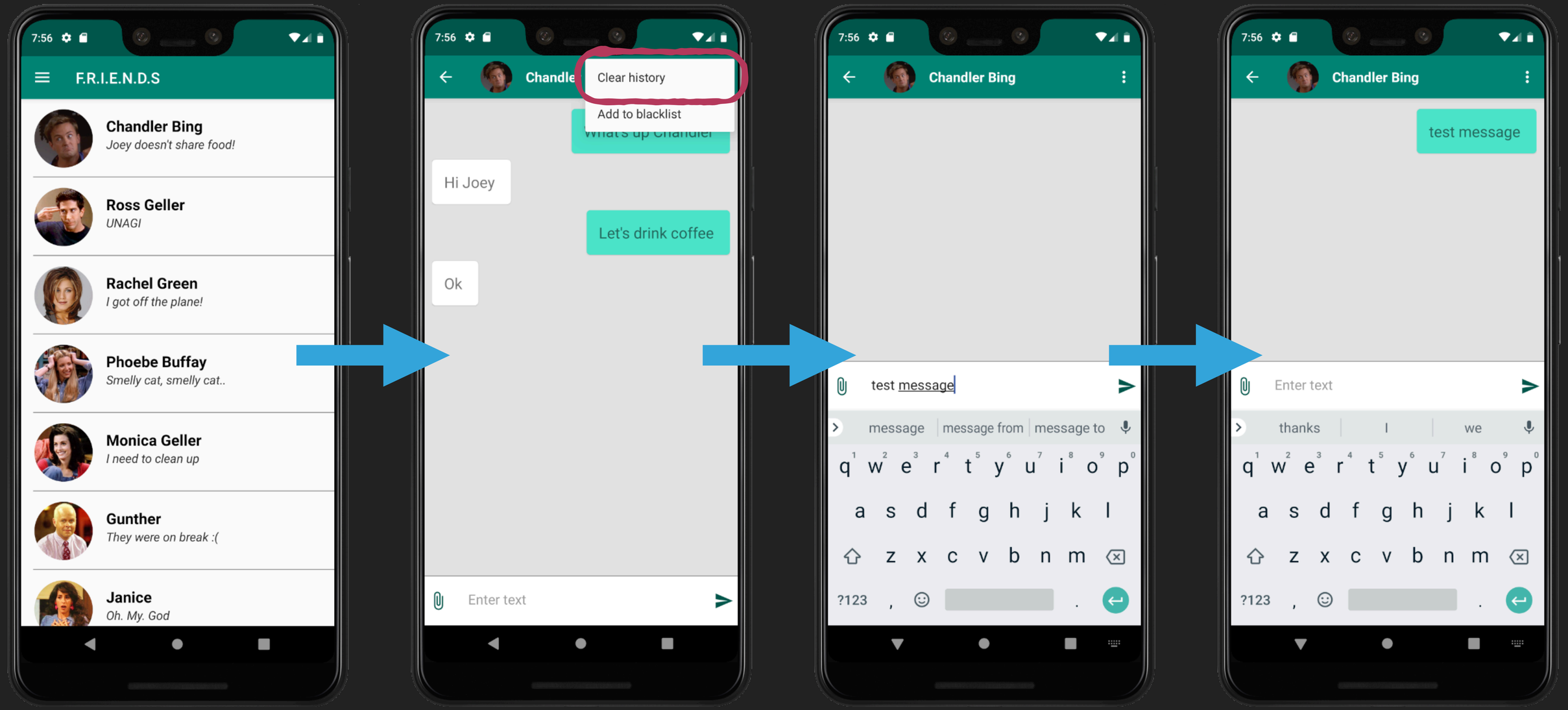



7:56
F.R.I.E.N.D.S
Chandler Bing
Joey doesn't share food!
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God


7:56
F.R.I.E.N.D.S
Chandler Bing
Joey doesn't share food!
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God

7:56
Chandle
Clear history
Add to blacklist
Hi Joey
Let's drink coffee
Ok
Enter text


7:56
F.R.I.E.N.D.S
Chandler Bing
Joey doesn't share food!
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God
7:56
Chandle
Clear history
Add to blacklist
Hi Joey
Let's drink coffee
Ok
Enter text
7:56
Chandler Bing
test message
message
message from
message to
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
?123


7:56
F.R.I.E.N.D.S
Chandler Bing
Joey doesn't share food!
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God
Chandle
Clear history
Add to blacklist
Hi Joey
Let's drink coffee
Ok
Enter text
Chandler Bing
test message
message
message from
message to
Chandler Bing
test message
Enter text
thanks
I
we


DEMO

EspressoGuide [~/develop/EspressoGuide] - .../app/src/androidTest/java/com/atiurin/espressoguide/tests/DemoEspressoTest.kt [app]

com › atiurin › espressoguide › tests › DemoEspressoTest | sendMessageWithLink() | Pixel 3 XL API 28 | Git:

DemoEspressoTest.kt | MainActivity.kt | ContactsPresenter.kt | ChatPageTest.kt | ChatPage.kt | BaseTest.kt | ScreenshotLifecycleListener.kt | AbstractLifecycleListener.kt | LifecycleListener.kt

```
    companion object {
        @BeforeClass
        @JvmStatic
        fun switchOffFailureHandler() {...}
    }

    lateinit var chatPage: ChatPage
    private val contact : Contact  = ContactsRepositoty.getContact( name: "Chandler Bing")

    init {
        ruleSequence
            .add(SetUpTearDownRule()
                .addSetUp {...})
            .addLast(CustomActivityTestRule(MainActivity::class.java))
    }

    @Test
    fun friendsItemCheck() {...}

    @Test
    fun sendMessage() {
        FriendsListPage().openChat(contact)
        ChatPage(contact).clearHistory()
                         .sendMessage( text: "test message")
    }

    @Test
    fun sendMessage3() {...}
```

DemoEspressoTest › sendMessage()

Run | TODO | 9: Version Control | Profiler | 6: Logcat | Build | Terminal | Multi-OS Engine | 2 Event Log

ts passed: 1 (8 minutes ago)

57:54 LF UTF-8 4 spaces Git: master

Gradle | Device File Explorer

COOL TEST

```
@Test
fun goodTest(){
    FriendsListPage().openChat("Ross Geller")
    ChatPage().clearHistory()
              .sendMessage("test message")
}
```

IS IT A GOOD IDEA?

```
@Test
fun sendMessageWithLink(){
    FriendsListPage().openChat("Ross Geller”)
    ChatPage().clearHistory()
              .sendMessage("message with link vk.com")
}
```

ACTUALLY IT ISN'T A GOOD IDEA!

```
@Test
fun sendMessageWithLink(){
    FriendsListPage().openChat("Ross Geller”)
    ChatPage().clearHistory()
              .sendMessage("message with link vk.com")
}
```

LONG TEST SCENARIO

LONG TEST SCENARIO

- It’s a flaky way

LONG TEST SCENARIO

- It’s a flaky way
- It takes a lot of time

LONG TEST SCENARIO

- It’s a flaky way
- It takes a lot of time
- Code duplication


MARTY, LOOK AT THE SHORT TEST!

SOLUTION: SHORT TEST

```
@Test
fun sendMessageWithLink(){
    FriendsListPage().openChat("Ross Geller”)
    ChatPage().clearHistory()
              .sendMessage("message with link vk.com")
}
```

SOLUTION: SHORT TEST

```
@Test
fun sendMessageWithLink(){
    FriendsListPage().openChat("Ross Geller”)
    ChatPage().clearHistory()
    ChatPage().sendMessage("message with link vk.com")
}
```

SOLUTION: SHORT TEST

```
class ChatTest {
    @Test
    fun sendMessageWithLink() {
        ChatPage().sendMessage("message with link vk.com")
    }
}
```

```
class ChatTest {
    @get:Rule
    val activityRule = ActivityTestRule(MainActivity::class.java, false, false)
}
```

```
class ChatTest {
    @get:Rule
    val activityRule = ActivityTestRule(MainActivity::class.java, false, false)
}
```

```
class ChatTest {
    @get:Rule
    val activityRule = ActivityTestRule(ChatActivity::class.java, false, false)
}
```

```
class ChatTest {
    @get:Rule
    val activityRule = ActivityTestRule(ChatActivity::class.java, false, false)
    @Before
    fun prepareData() {
        MessageRepository.clearChatMessages(contact.id)
        val intent = Intent()
        intent.putExtra("INTENT_CONTACT_ID_EXTRA_NAME", contact.id)
        activityRule.launchActivity(intent)
    }
}
```

```
class ChatTest {
    @get:Rule
    val activityRule = ActivityTestRule(ChatActivity::class.java, false, false)
    @Before
    fun prepareData() {
        MessageRepository.clearChatMessages(contact.id)
        val intent = Intent()
        intent.putExtra("INTENT_CONTACT_ID_EXTRA_NAME", contact.id)
        activityRule.launchActivity(intent)
    }
}
```

```
class ChatTest {
    @get:Rule
    val activityRule = ActivityTestRule(ChatActivity::class.java, false, false)
    @Before
    fun prepareData() {
        MessageRepository.clearChatMessages(contact.id)
        val intent = Intent()
        intent.putExtra("INTENT_CONTACT_ID_EXTRA_NAME", contact.id)
        activityRule.launchActivity(intent)
    }
}
```

```
class ChatTest {
    @get:Rule
    val activityRule = ActivityTestRule(ChatActivity::class.java, false, false)
    @Before
    fun prepareData() {
        MessageRepository.clearChatMessages(contact.id)
        val intent = Intent()
        intent.putExtra("INTENT_CONTACT_ID_EXTRA_NAME", contact.id)
        activityRule.launchActivity(intent)
    }
}
```

```
class ChatTest {
    @get:Rule
    val activityRule = ActivityTestRule(ChatActivity::class.java, false, false)
    @Before
    fun prepareData() {
        MessageRepository.clearChatMessages(contact.id)
        val intent = Intent()
        intent.putExtra("INTENT_CONTACT_ID_EXTRA_NAME", contact.id)
        activityRule.launchActivity(intent)
    }
}
```

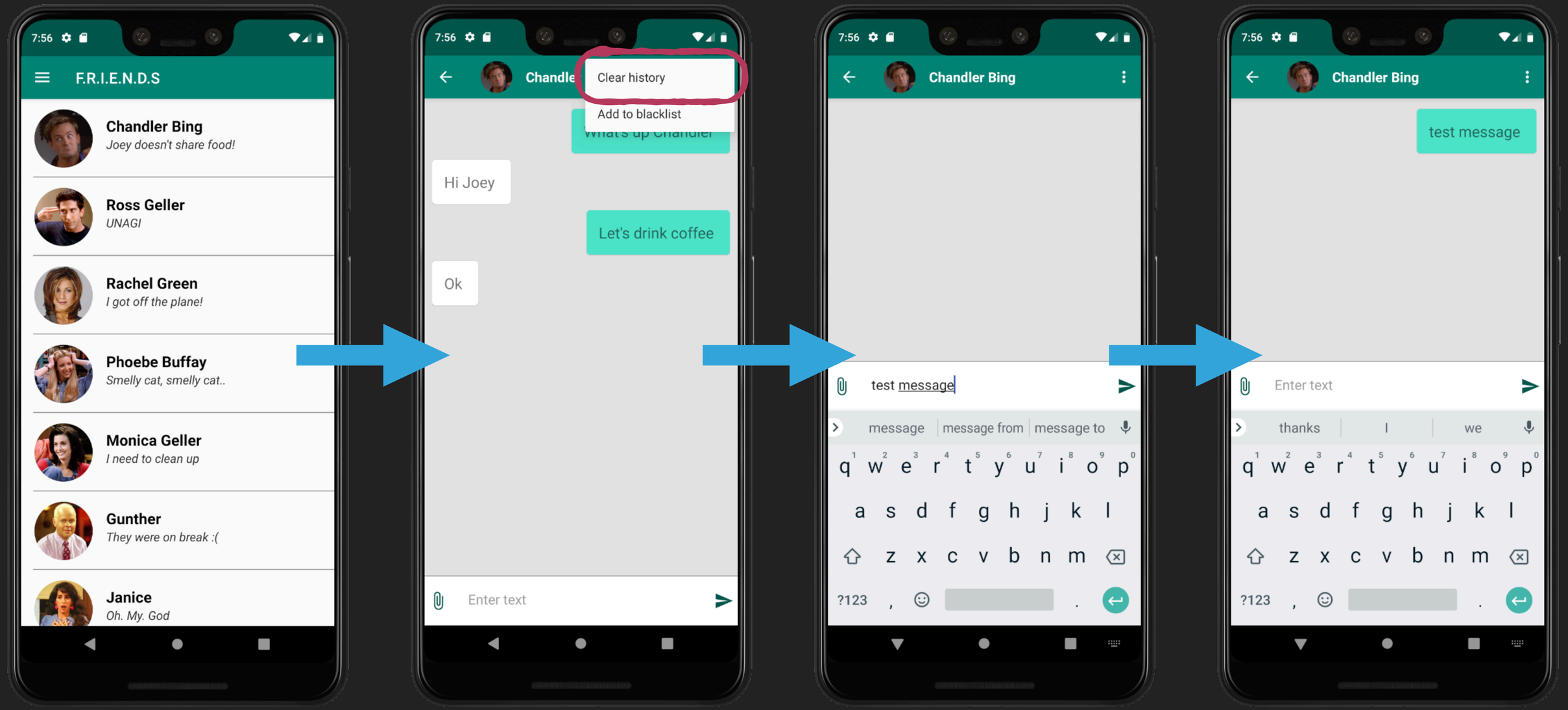
```
class ChatTest {
    @get:Rule
    val activityRule = ActivityTestRule(ChatActivity::class.java, false, false)
    @Before
    fun prepareData() {
        MessageRepository.clearChatMessages(contact.id)
        val intent = Intent()
        intent.putExtra("INTENT_CONTACT_ID_EXTRA_NAME", contact.id)
        activityRule.launchActivity(intent)
    }
    @Test
    fun sendMessageWithLink() {
        ChatPage().sendMessage("message with link vk.com")
    }
}
```

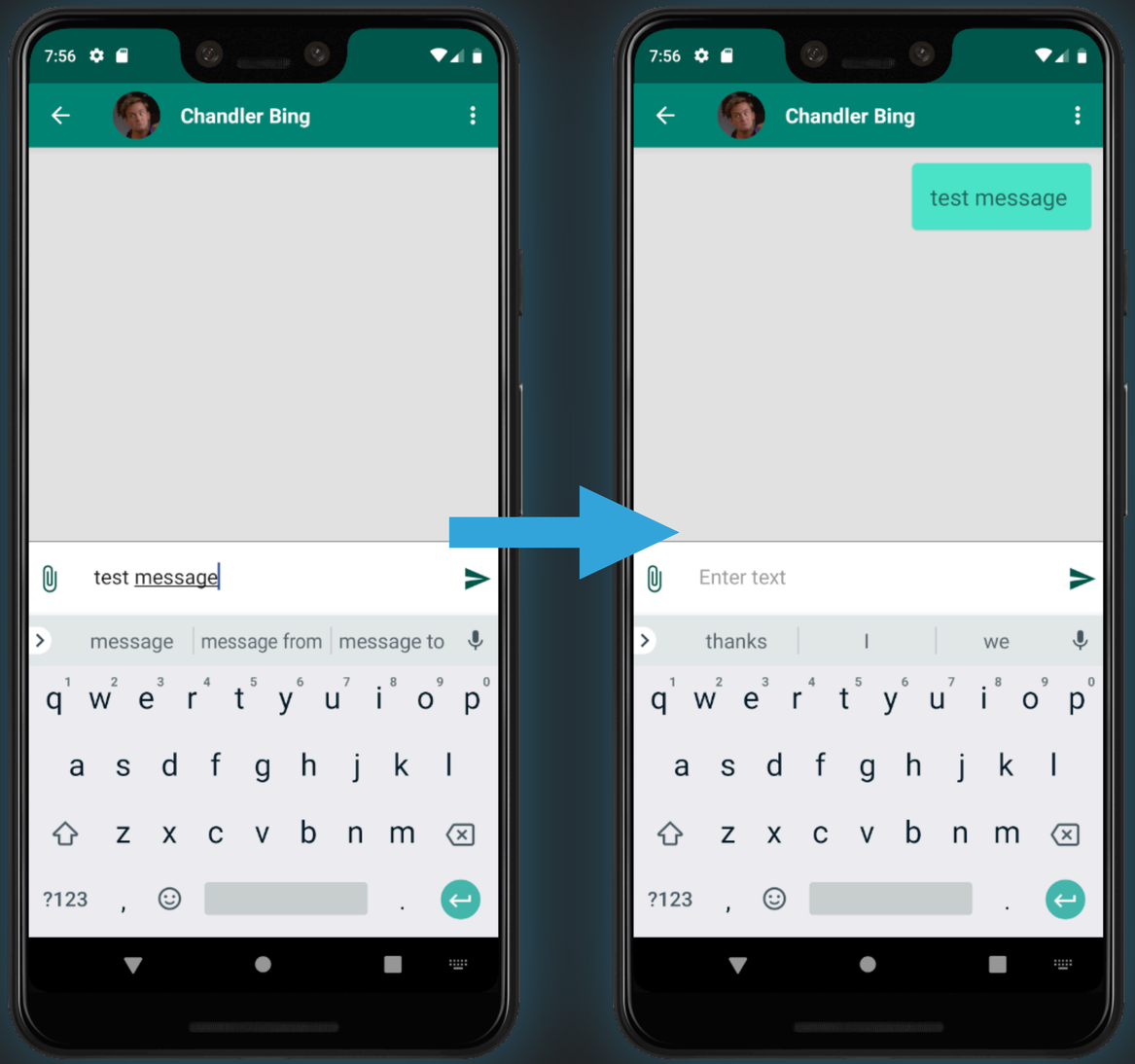
```
class ChatTest {
    @get:Rule
    val activityRule = ActivityTestRule(ChatActivity::class.java, false, false)
    @Before
    fun prepareData() {
        MessageRepository.clearChatMessages(contact.id)
        val intent = Intent()
        intent.putExtra("INTENT_CONTACT_ID_EXTRA_NAME", contact.id)
        activityRule.launchActivity(intent)
    }
    @Test
    fun sendMessageWithLink() {...}
}
```

```
class ChatTest {
    @get:Rule
    val activityRule = ActivityTestRule(ChatActivity::class.java, false, false)

    val setupRule = SetUpTearDownRule()
        .addSetUp {
            MessageRepository.clearChatMessages(contact.id)
            val intent = Intent()
            intent.putExtra("INTENT_CONTACT_ID_EXTRA_NAME", contact.id)
            activityRule.launchActivity(intent)
        }
    @Test
    fun sendMessageWithLink() {...}
}
```

```
class ChatTest {
    @get:Rule
    val activityRule = ActivityTestRule(ChatActivity::class.java, false, false)

    val setupRule = SetUpTearDownRule()
        .addSetUp {
            MessageRepository.clearChatMessages(contact.id)
            val intent = Intent()
            intent.putExtra("INTENT_CONTACT_ID_EXTRA_NAME", contact.id)
            activityRule.launchActivity(intent)
        }
    @Test
    fun sendMessageWithLink() {...}
}
```

```
class ChatTest {
    val activityRule = ActivityTestRule(ChatActivity::class.java, false, false)
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            MessageRepository.clearChatMessages(contact.id)
            val intent = Intent()
            intent.putExtra("INTENT_CONTACT_ID_EXTRA_NAME", contact.id)
            activityRule.launchActivity(intent)
        }
    @get:Rule
    val ruleSequence = RuleSequence(setupRule, activityRule)

    @Test
    fun sendMessageWithLink() {...}
}
```


7:56
F.R.I.E.N.D.S
Chandler Bing
Joey doesn't share food!
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God
Chandle
Clear history
Add to blacklist
Hi Joey
Let's drink coffee
Ok
Enter text
Chandler Bing
test message
message
message from
message to
Chandler Bing
test message
Enter text
thanks
I
we


7:56
Chandler Bing
test message
message
message from
message to
7:56
Chandler Bing
test message
Enter text
thanks
I
we

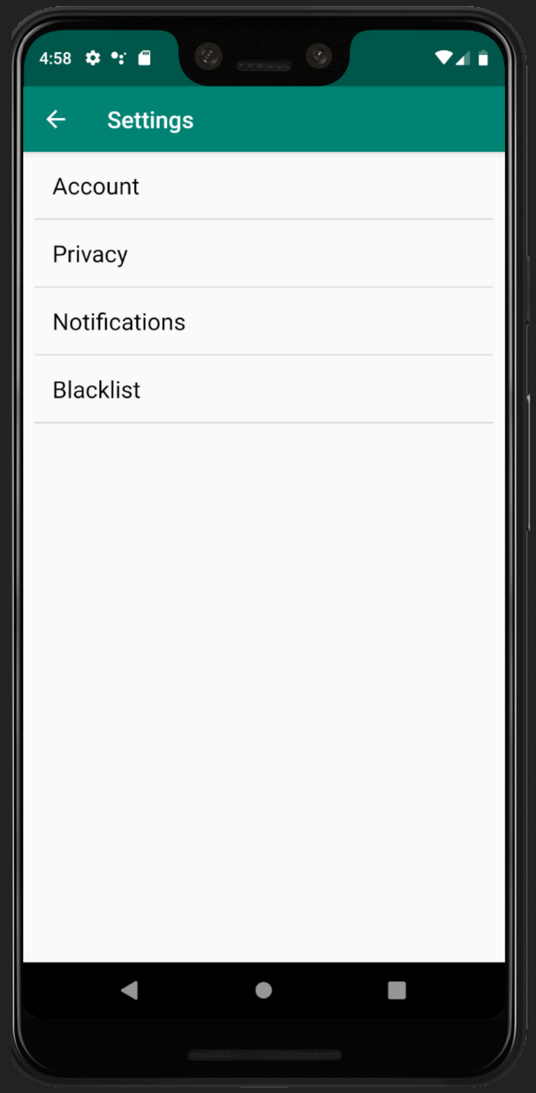

DEMO

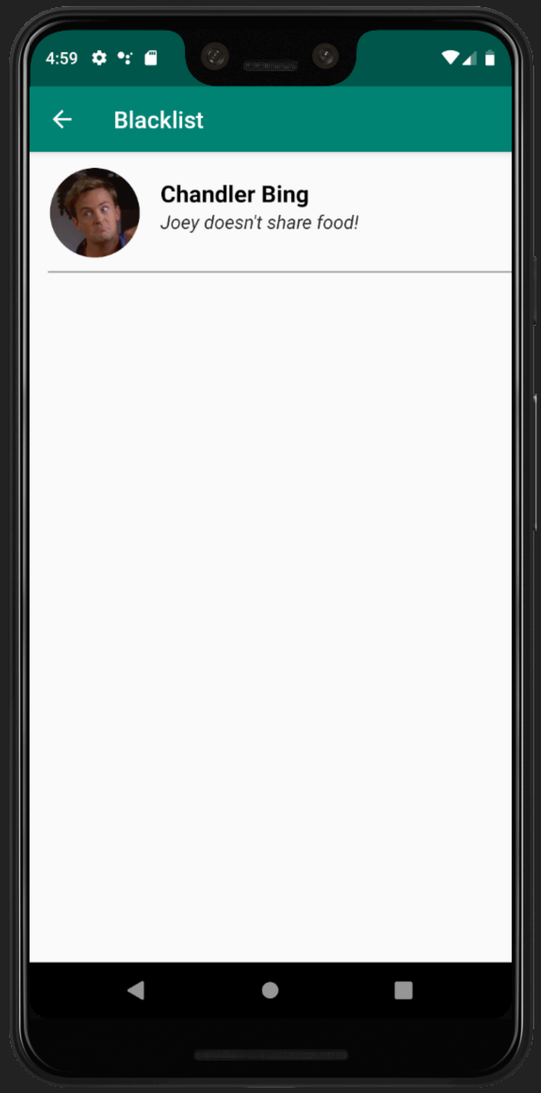
```
private val specialCharsMessage = Message(CURRENT_USER.id, contact.id, text: "!@#$%^&*(){}:\",./<>?_+±§

private val activityTestRule : CustomActivityTestRule<ChatActivity> = CustomActivityTestRule(ChatActivity::cla
private val setUpTearDownRule : SetUpTearDownRule = SetUpTearDownRule()
    .addSetUp {
        MessageRepository.clearChatMessages(contact.id)
    }
    .addSetUp(ADD_SIMPLE_MESSAGE) {
        MessageRepository.addChatMessage(contact.id, simpleMessage)
    }
    .addSetUp(ADD_SPECIAL_CHARS_MESSAGE) {
        MessageRepository.addChatMessage(contact.id, specialCharsMessage)
    }
    .addSetUp {
        AccountManager(InstrumentationRegistry.getInstrumentation().targetContext).login(
            CURRENT_USER.login,
            CURRENT_USER.password
        )
        val intent : Intent = Intent().putExtra(INTENT_CONTACT_ID_EXTRA_NAME, contact.id)
        activityTestRule.launchActivity(intent)
    }

init {
    ruleSequence.add(setUpTearDownRule, activityTestRule)
}

@Test
fun sendMessageWithLink(){
    val message = "message with link vk.com"
```

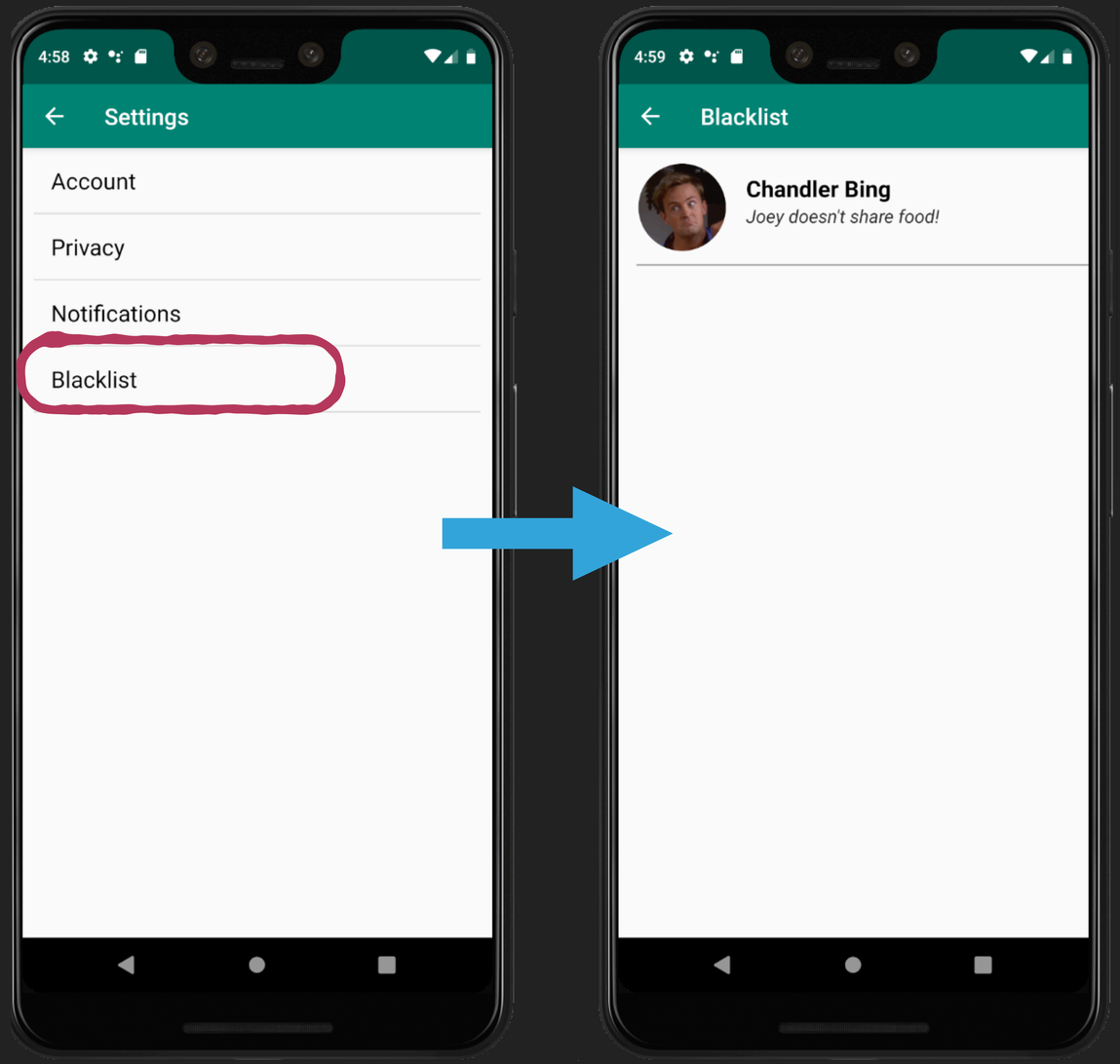
SOLUTION: SHORT TEST

```
class ChatTest {
    @Test
    fun sendMessageWithLink() {
        ChatPage().sendMessage("message with link vk.com")
    }
}
```

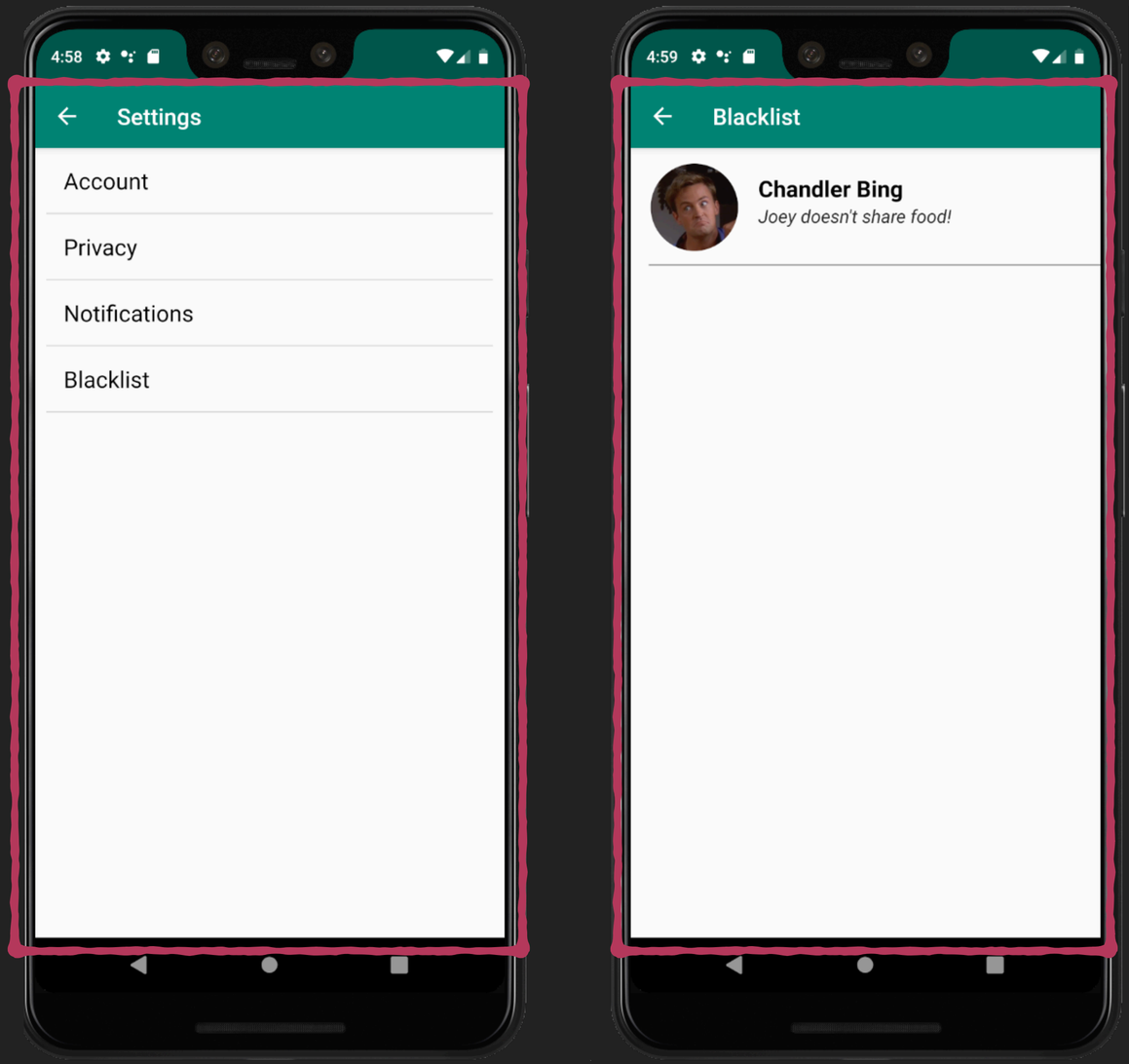
WHAT CAN I DO WITH FRAGMENTS?


4:58
Settings
Account
Privacy
Notifications
Blacklist


4:59
Blacklist
Chandler Bing
Joey doesn't share food!

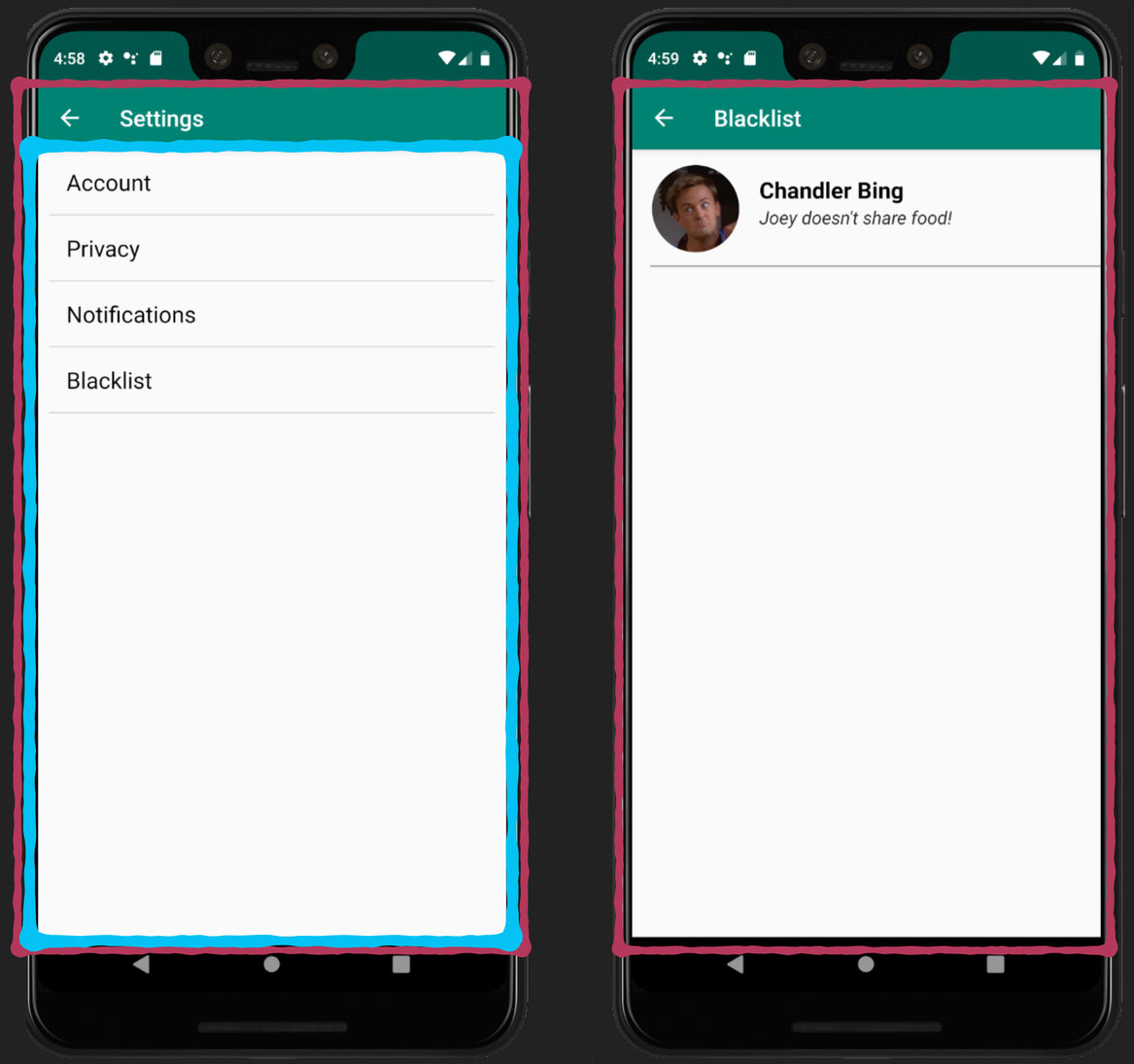

4:58
Settings
Account
Privacy
Notifications
Blacklist
4:59
Blacklist
Chandler Bing
Joey doesn't share food!

SettingsActivity



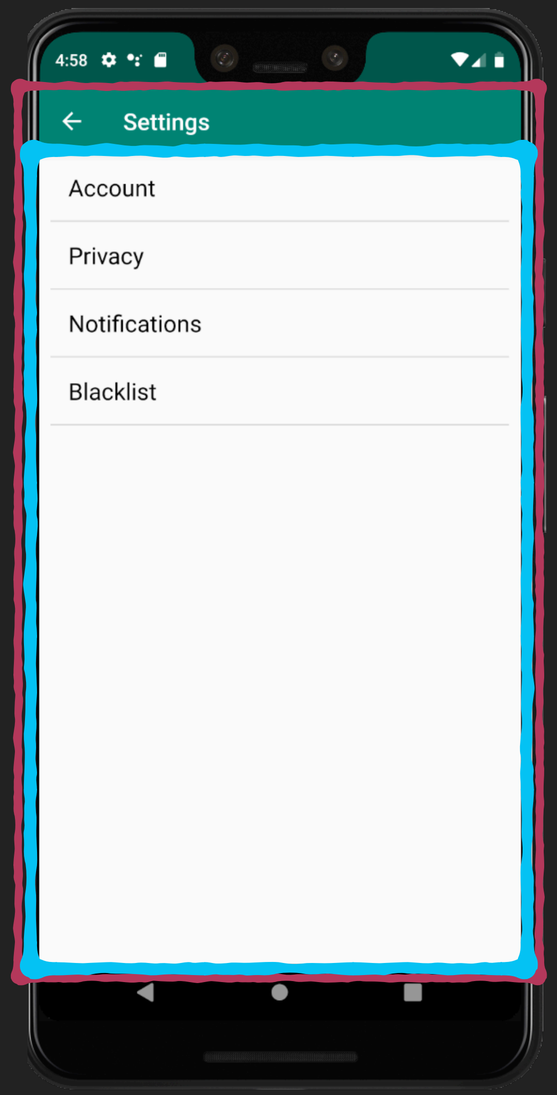
SettingsActivity

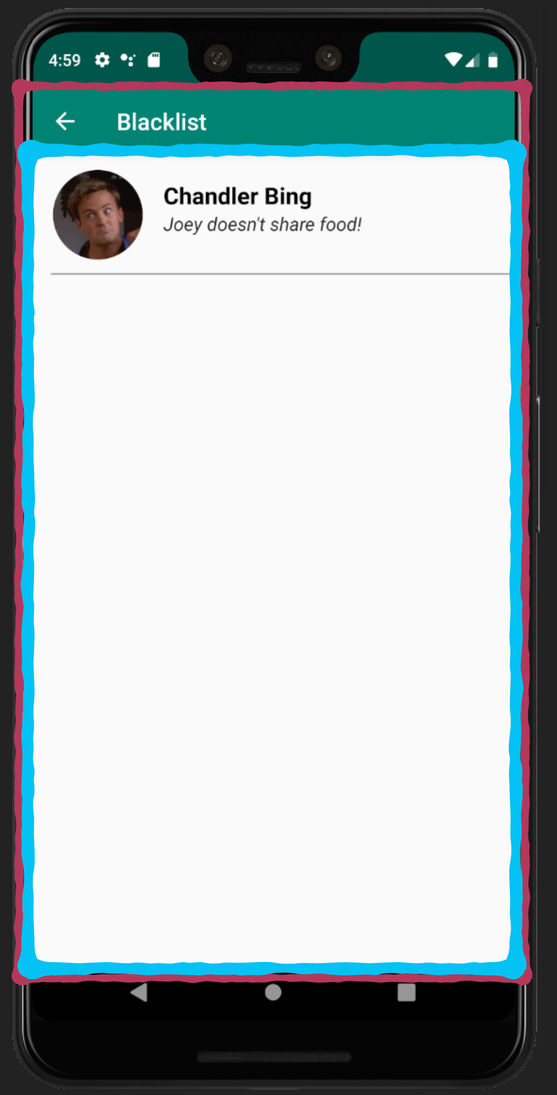
SettingsListFragment



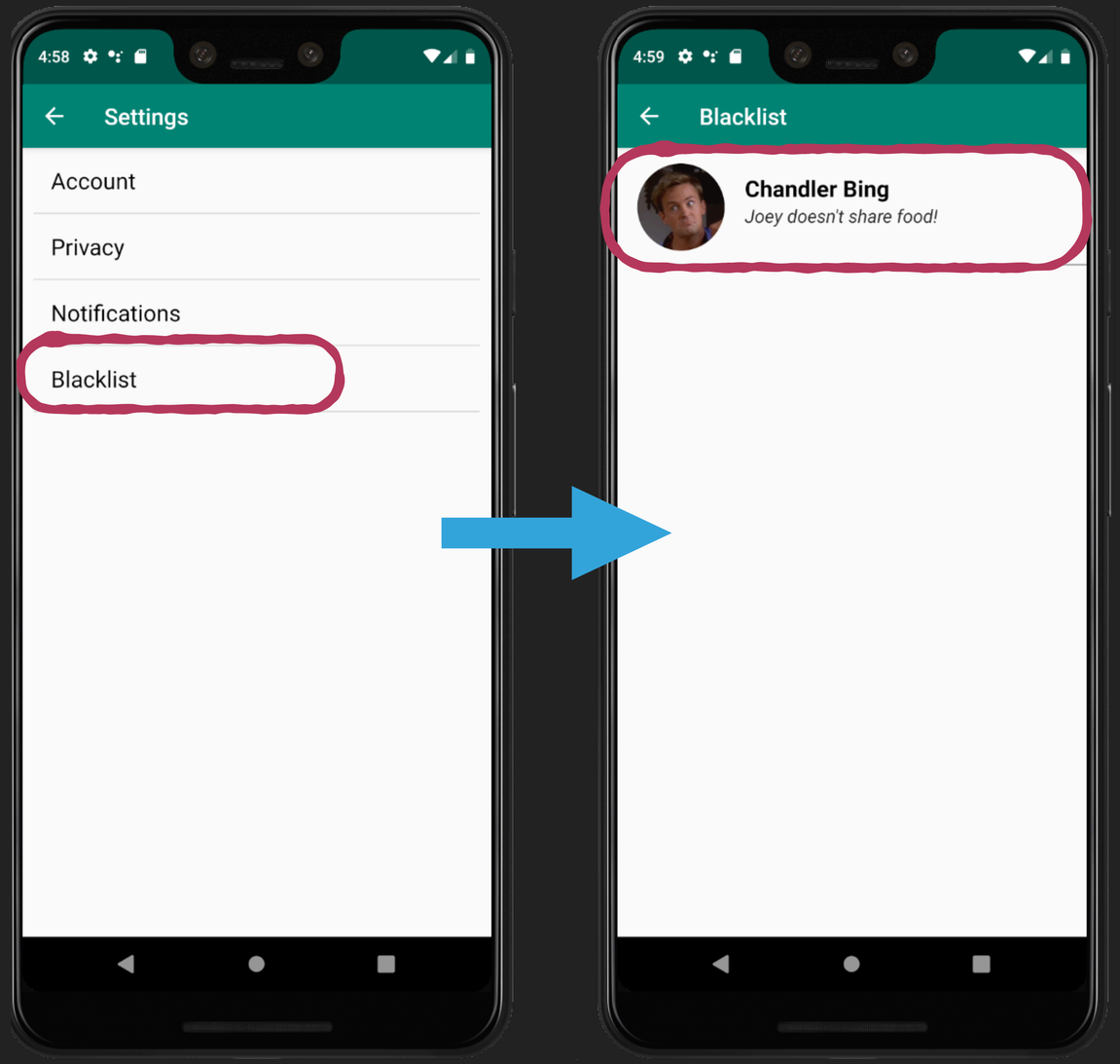
SettingsActivity

SettingsListFragment





BlacklistFragment


4:58
Settings
Account
Privacy
Notifications
Blacklist
4:59
Blacklist
Chandler Bing
Joey doesn't share food!

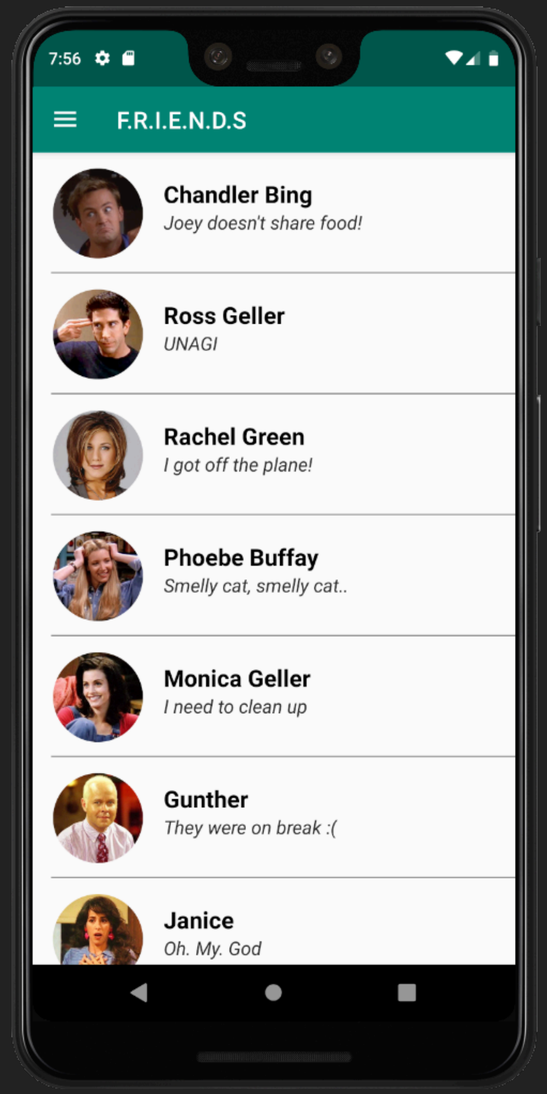
```
class BlacklistTests {
    ...
    @get:Rule
    val ruleSequence = RuleSequence(setupRule, activityRule)
}
```

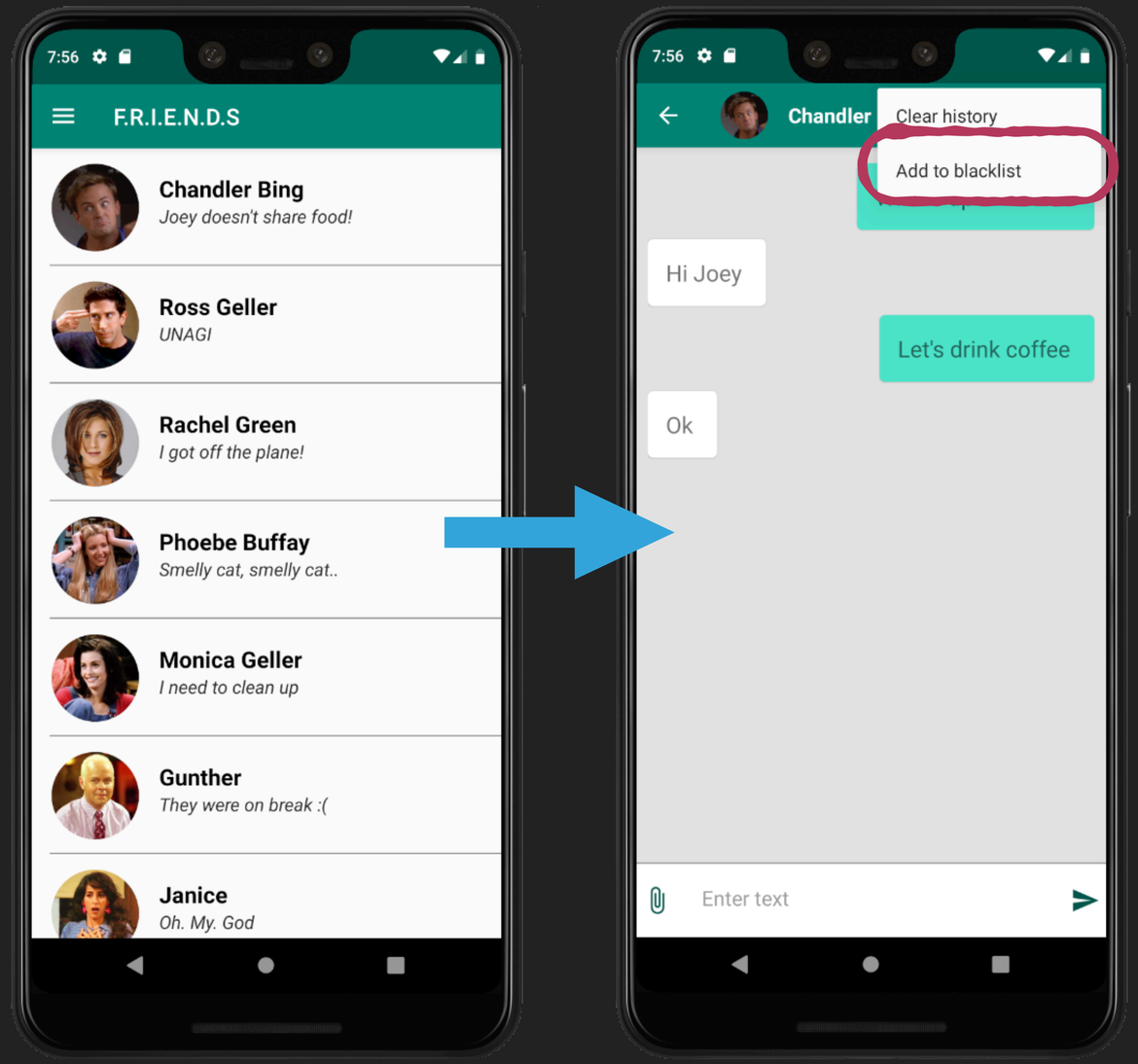
```
class BlacklistTests {
    ...
    @get:Rule
    val ruleSequence = RuleSequence(setupRule, activityRule)
    @Before
    fun openFragment(){
        activityRule.runOnUiThread {
            SettingsFragmentNavigator.go(BlacklistFragment::class.java)
        }
    }
}
```

```
class BlacklistTests {
    ...
    @get:Rule
    val ruleSequence = RuleSequence(setupRule, activityRule)
    @Before
    fun openFragment(){
        activityRule.runOnUiThread {
            SettingsFragmentNavigator.go(BlacklistFragment::class.java)
        }
    }
}
```

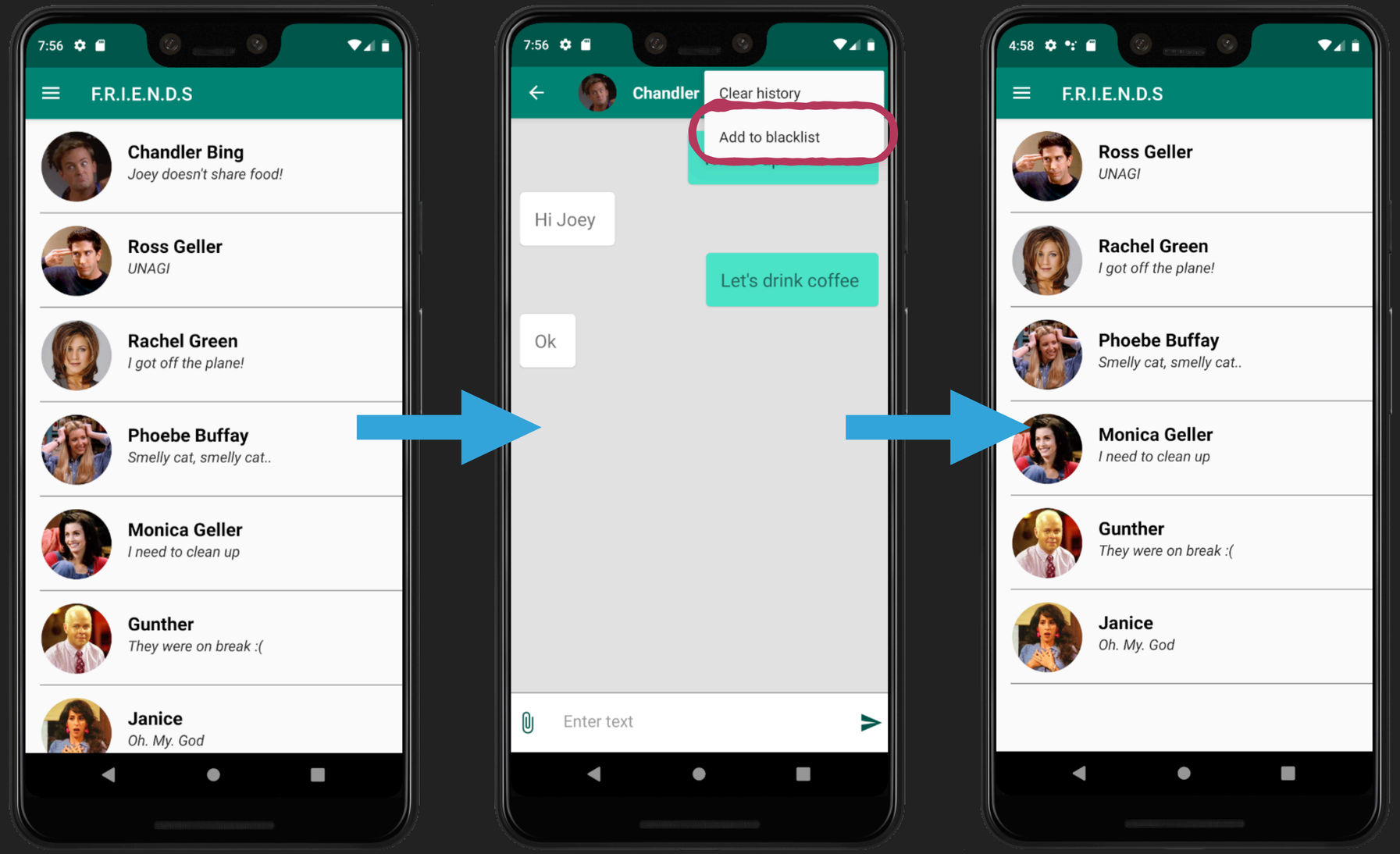
```
class BlacklistTests {
    ...
    @get:Rule
    val ruleSequence = RuleSequence(setupRule, activityRule)
    @Before
    fun openFragment(){
        activityRule.runOnUiThread {
            SettingsFragmentNavigator.go(BlacklistFragment::class.java)
        }
    }
}
```

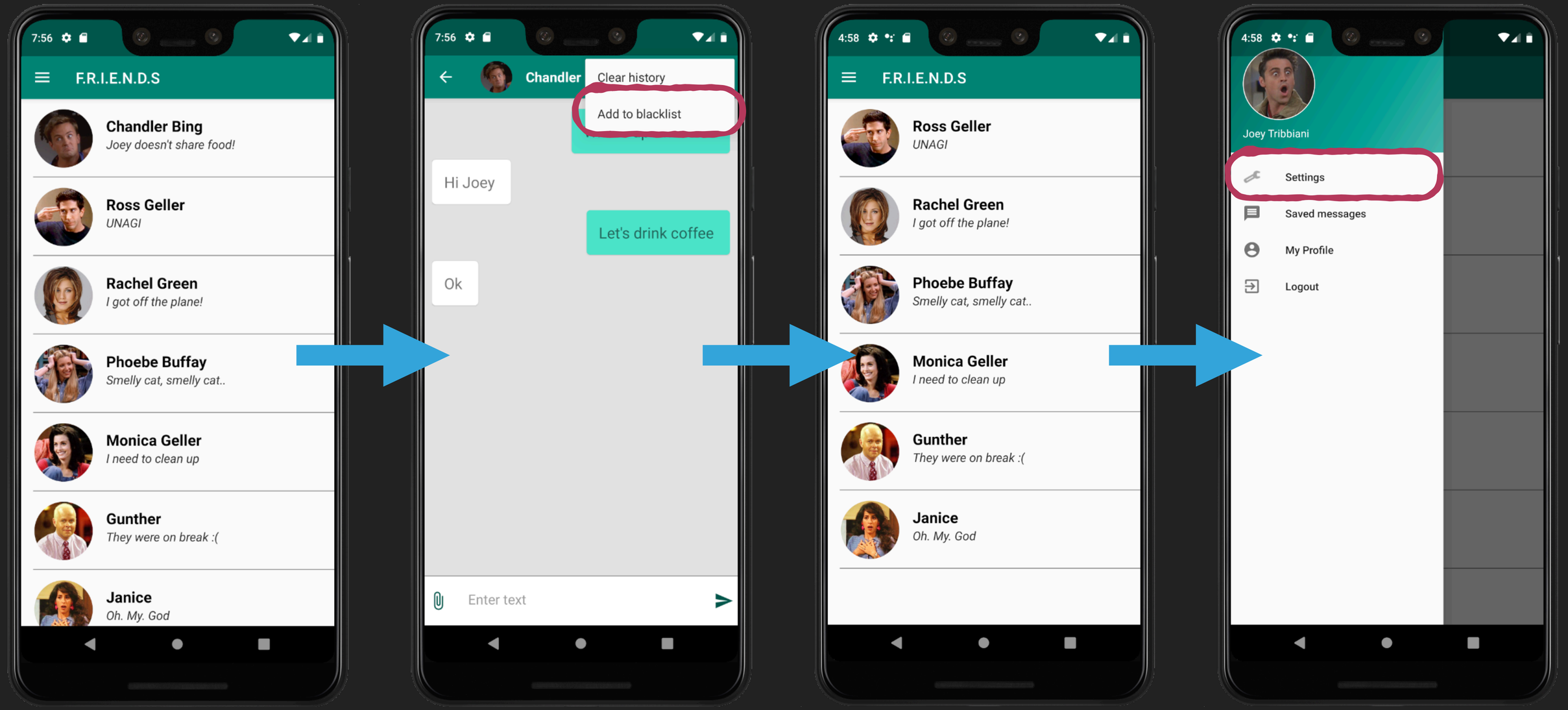
```
class BlacklistTests {
    ...
    @get:Rule
    val ruleSequence = RuleSequence(setupRule, activityRule)
        .addLast(SetUpTearDownRule().addSetUp {
            activityRule.runOnUiThread {
                SettingsFragmentNavigator.go(BlacklistFragment::class.java)
            }
        })
}
```

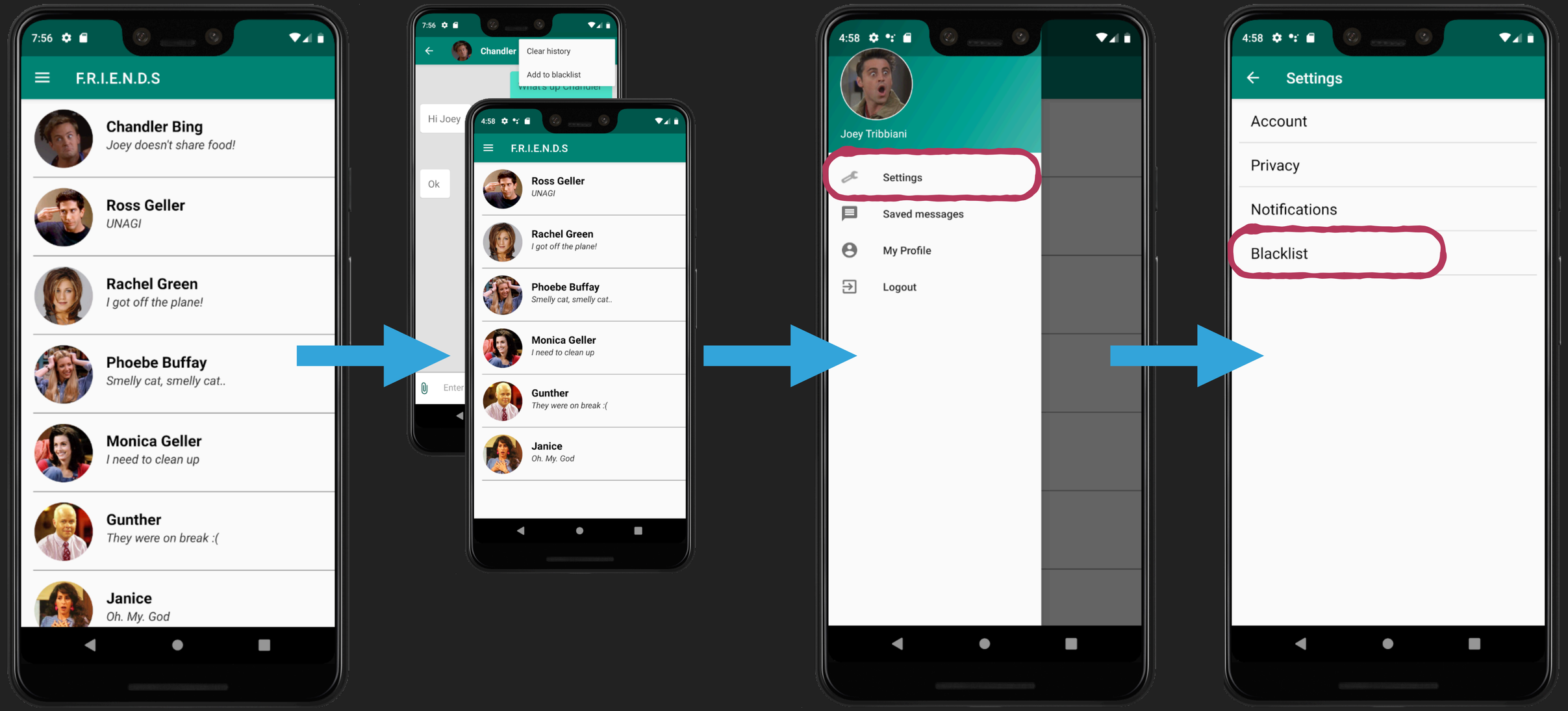

7:56
F.R.I.E.N.D.S
Chandler Bing
Joey doesn't share food!
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God


7:56
F.R.I.E.N.D.S
Chandler Bing
Joey doesn't share food!
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God

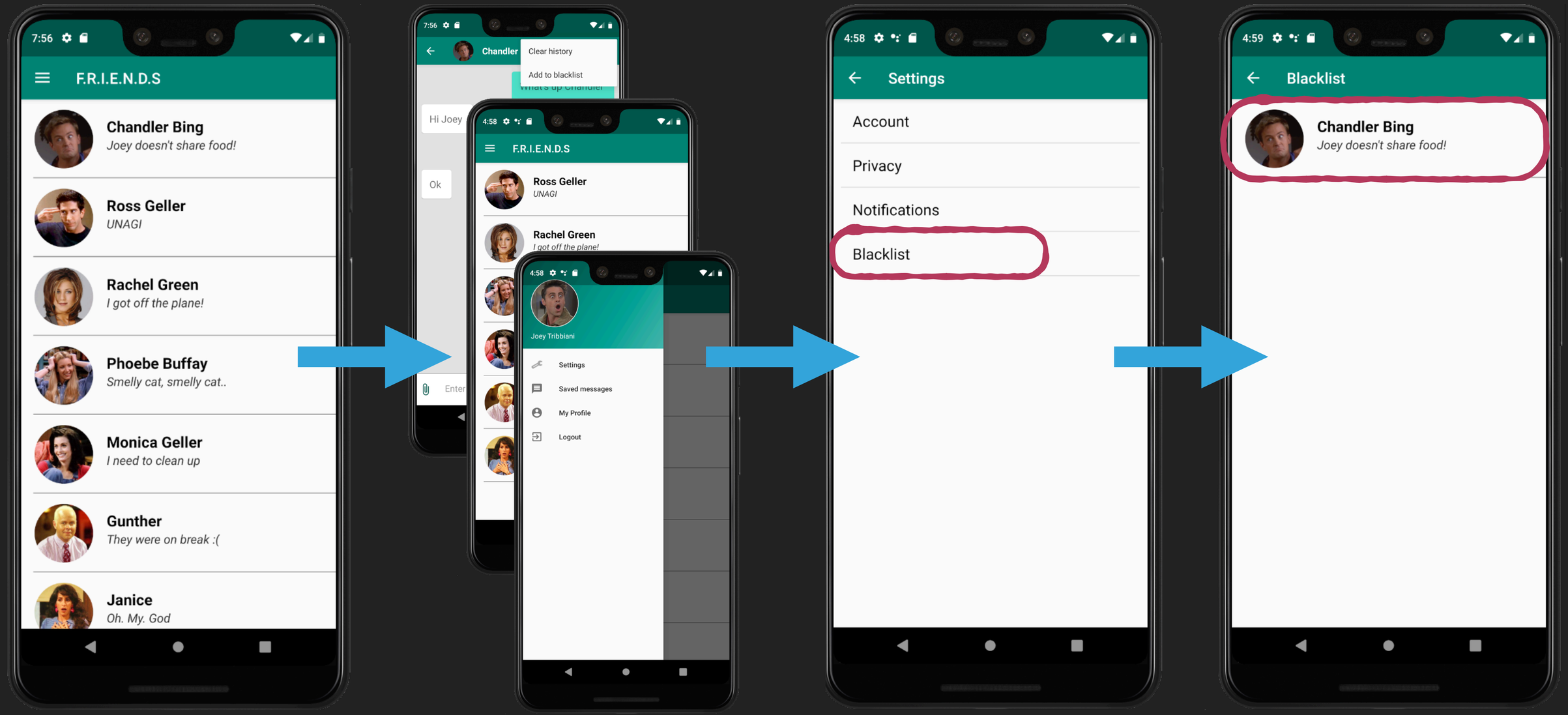
7:56
Chandler
Clear history
Add to blacklist
Hi Joey
Let's drink coffee
Ok
Enter text

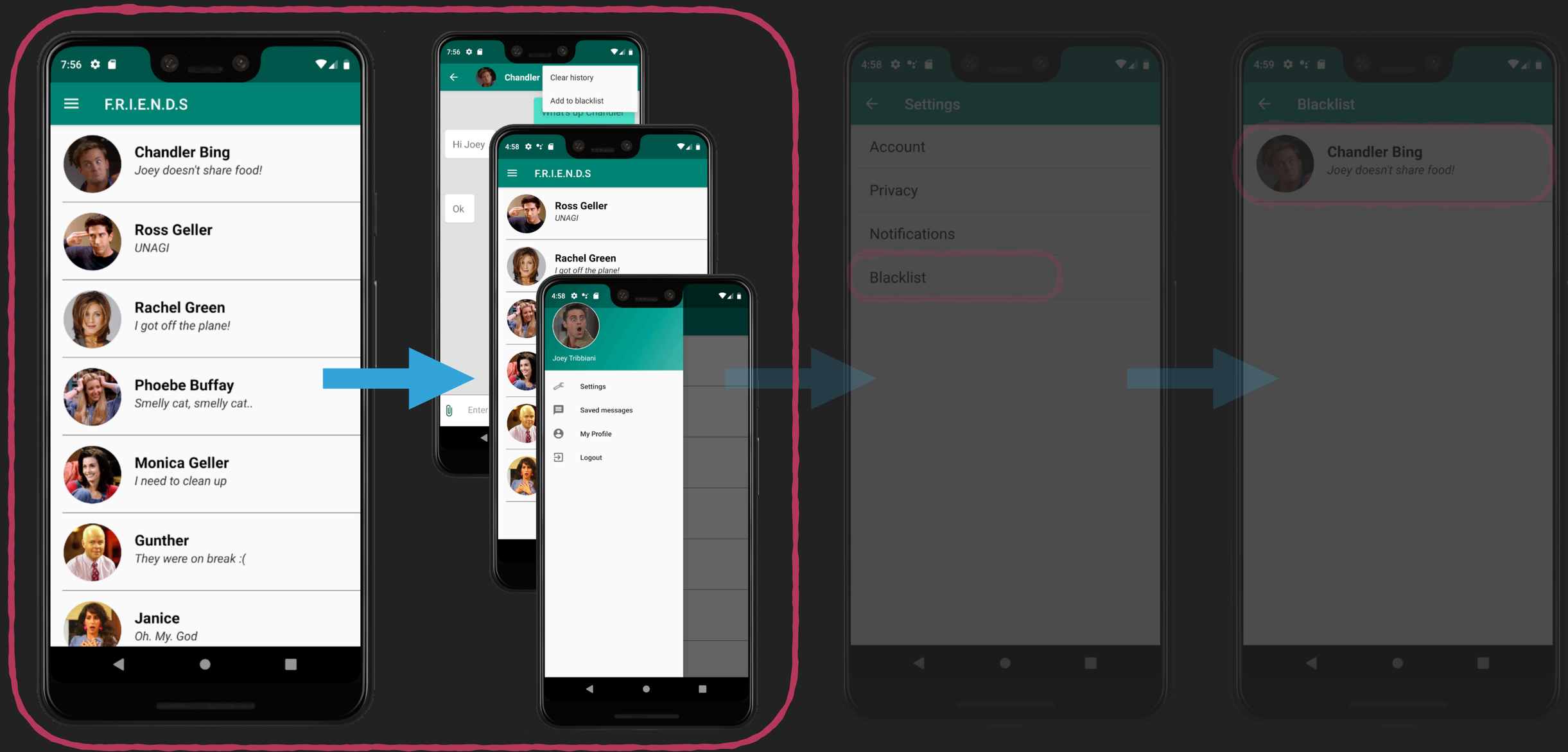

7:56
F.R.I.E.N.D.S
Chandler Bing
Joey doesn't share food!
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God
7:56
Chandler
Clear history
Add to blacklist
Hi Joey
Let's drink coffee
Ok
Enter text
4:58
F.R.I.E.N.D.S
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God


7:56
F.R.I.E.N.D.S
Chandler Bing
Joey doesn't share food!
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God
7:56
Chandler
Clear history
Add to blacklist
Hi Joey
Let's drink coffee
Ok
Enter text
4:58
F.R.I.E.N.D.S
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God
4:58
Joey Tribbiani
Settings
Saved messages
My Profile
Logout


7:56
F.R.I.E.N.D.S
Chandler Bing
Joey doesn't share food!
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God
Chandler
Clear history
Add to blacklist
Hi Joey
Ok
4:58
Joey Tribbiani
Settings
Saved messages
My Profile
Logout
Settings
Account
Privacy
Notifications
Blacklist

BLACKLIST CASE

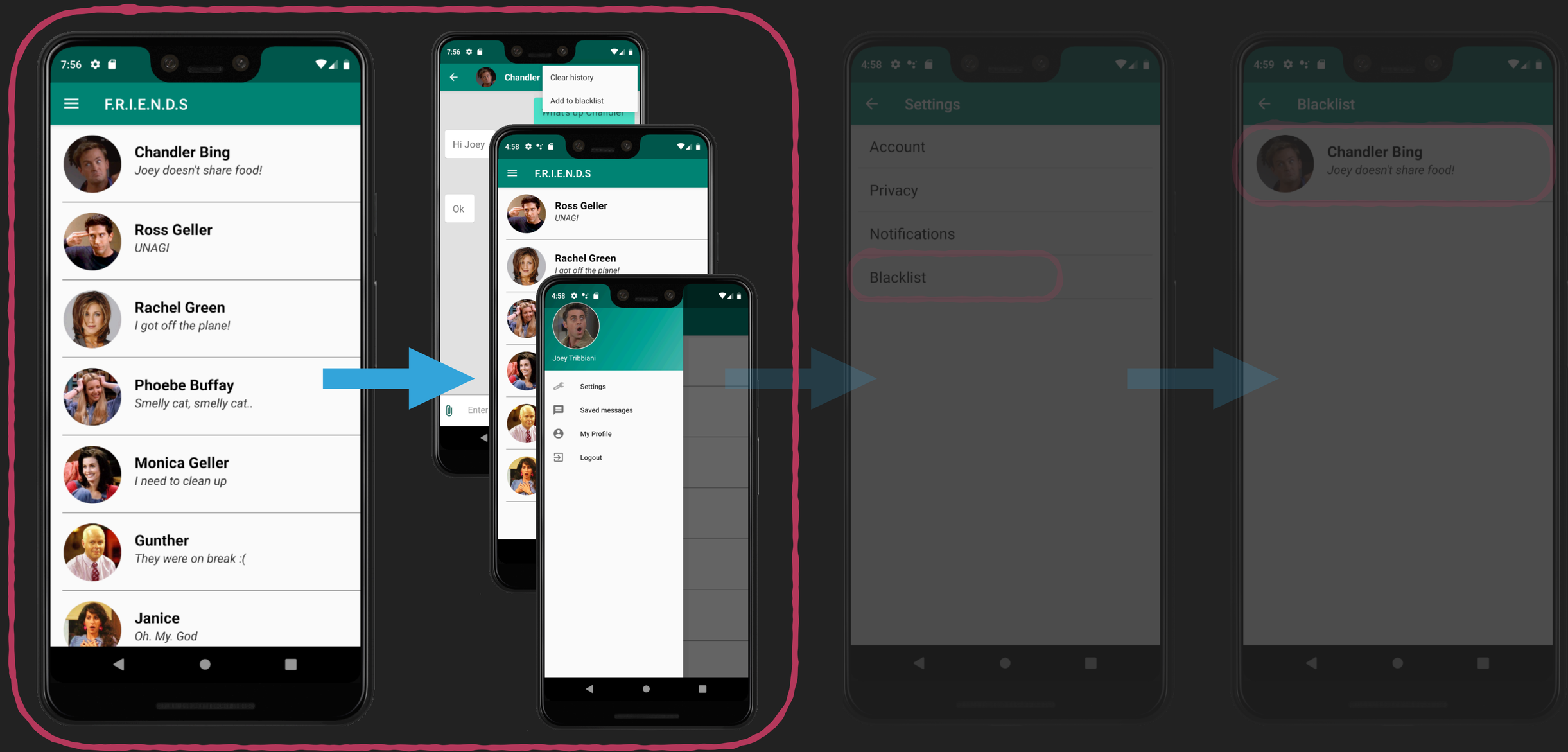



7:56
F.R.I.E.N.D.S
Chandler Bing
Joey doesn't share food!
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God
Chandler
Clear history
Add to blacklist
Hi Joey
Ok
Joey Tribbiani
Settings
Saved messages
My Profile
Logout
4:58
Settings
Account
Privacy
Notifications
Blacklist
4:59
Blacklist
Chandler Bing
Joey doesn't share food!

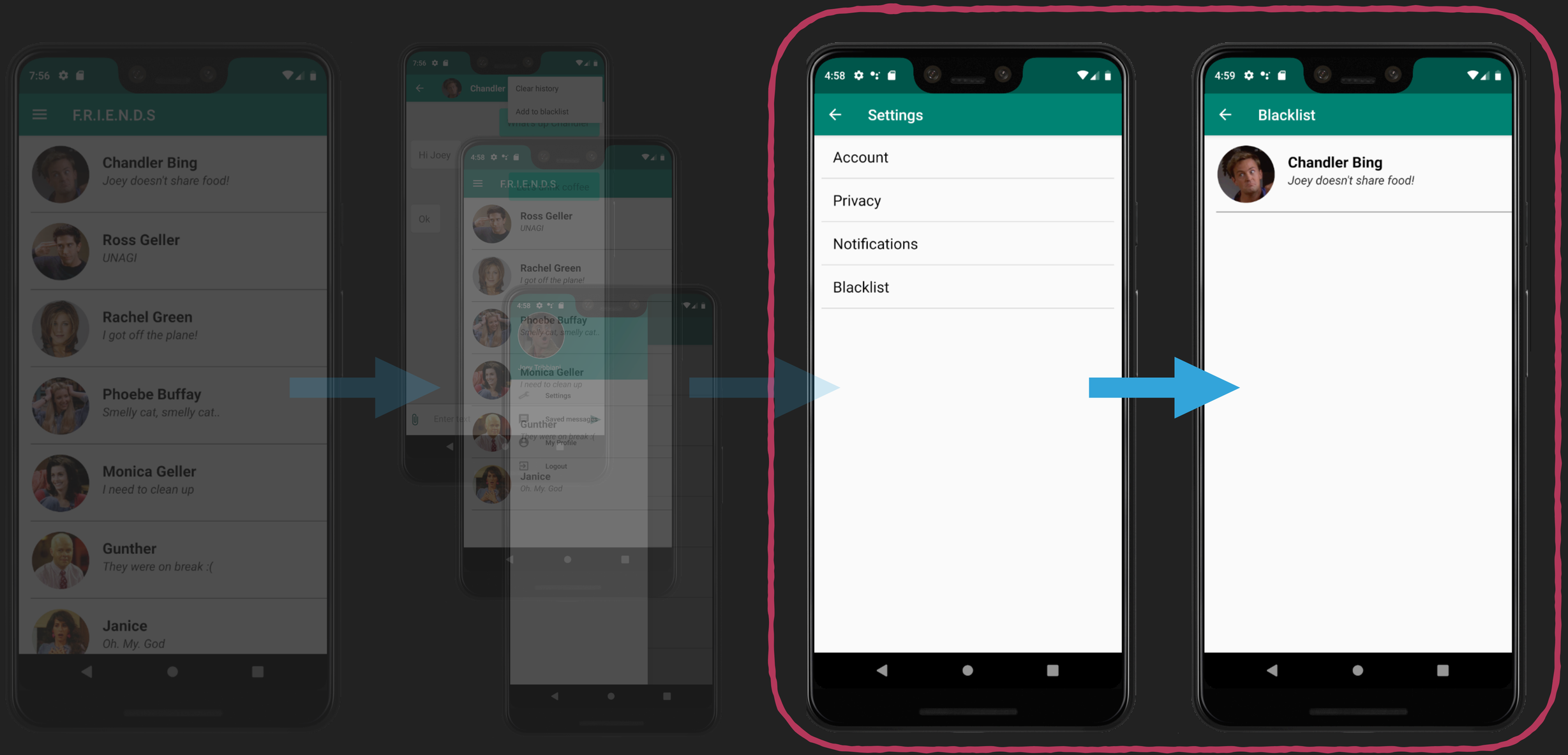
```
class BlacklistTests {
    val activityRule = ActivityTestRule(SettingsActivity::class.java)
    val setUpTearDownRule = SetUpTearDownRule()
        .addSetUp {
            ContactsRepositoty.clearBlacklist()
            ContactsRepositoty.addToBlacklist(contact.id)
        }
    @get:Rule
    val ruleSequence = RuleSequence(setupRule, activityRule)
}
```

```
class BlacklistTests {
    val activityRule = ActivityTestRule(SettingsActivity::class.java)
    val setUpTearDownRule = SetUpTearDownRule()
        .addSetUp {
            ContactsRepositoty.clearBlacklist()
            ContactsRepositoty.addToBlacklist(contact.id)
        }
    @get:Rule
    val ruleSequence = RuleSequence(setupRule, activityRule)
}
```

```
class BlacklistTests {
    val activityRule = ActivityTestRule(SettingsActivity::class.java)
    val setUpTearDownRule = SetUpTearDownRule()
        .addSetUp {
            ContactsRepositoty.clearBlacklist()
            ContactsRepositoty.addToBlacklist(contact.id)
        }
    @get:Rule
    val ruleSequence = RuleSequence(setupRule, activityRule)
}
```

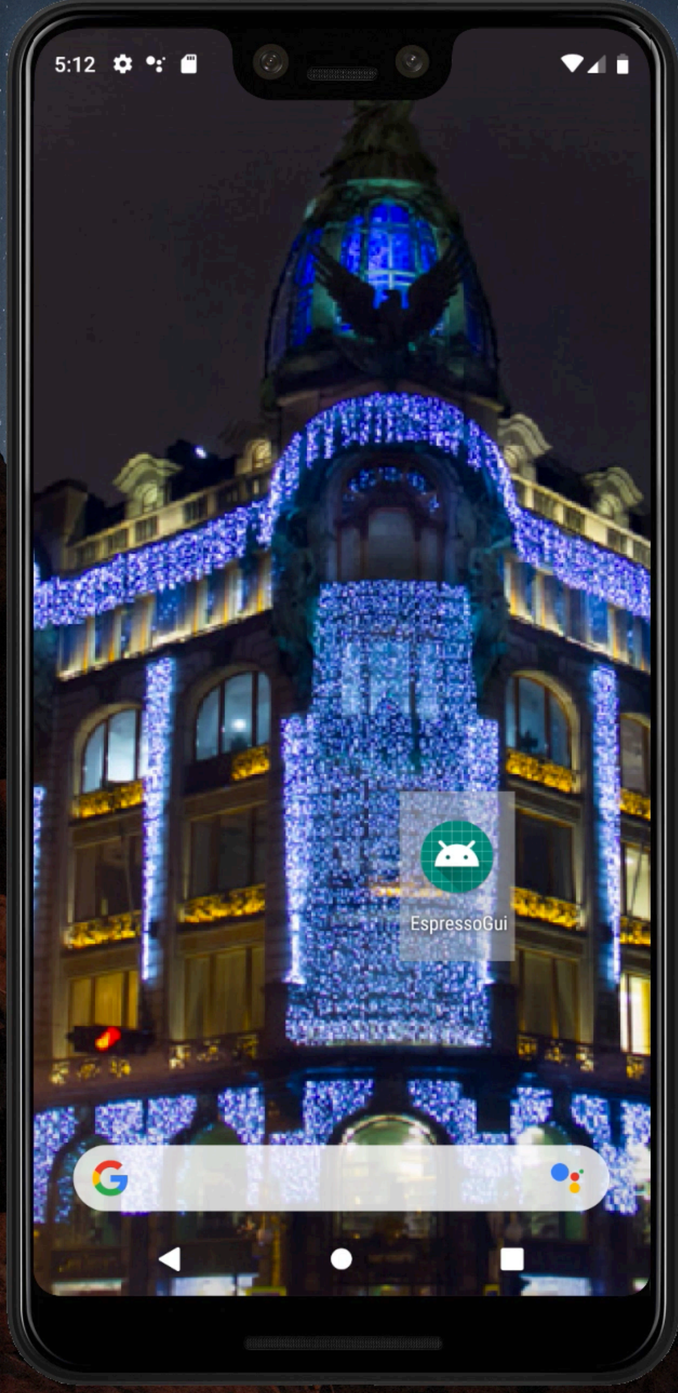

7:56
F.R.I.E.N.D.S
Chandler Bing
Joey doesn't share food!
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Phoebe Buffay
Smelly cat, smelly cat..
Monica Geller
I need to clean up
Gunther
They were on break :(
Janice
Oh. My. God
Chandler
Clear history
Add to blacklist
Hi Joey
Ok
4:58
F.R.I.E.N.D.S
Ross Geller
UNAGI
Rachel Green
I got off the plane!
Joey Tribbiani
Settings
Saved messages
My Profile
Logout
4:58
Settings
Account
Privacy
Notifications
Blacklist
4:59
Blacklist
Chandler Bing
Joey doesn't share food!

BLACKLIST CASE




DEMO

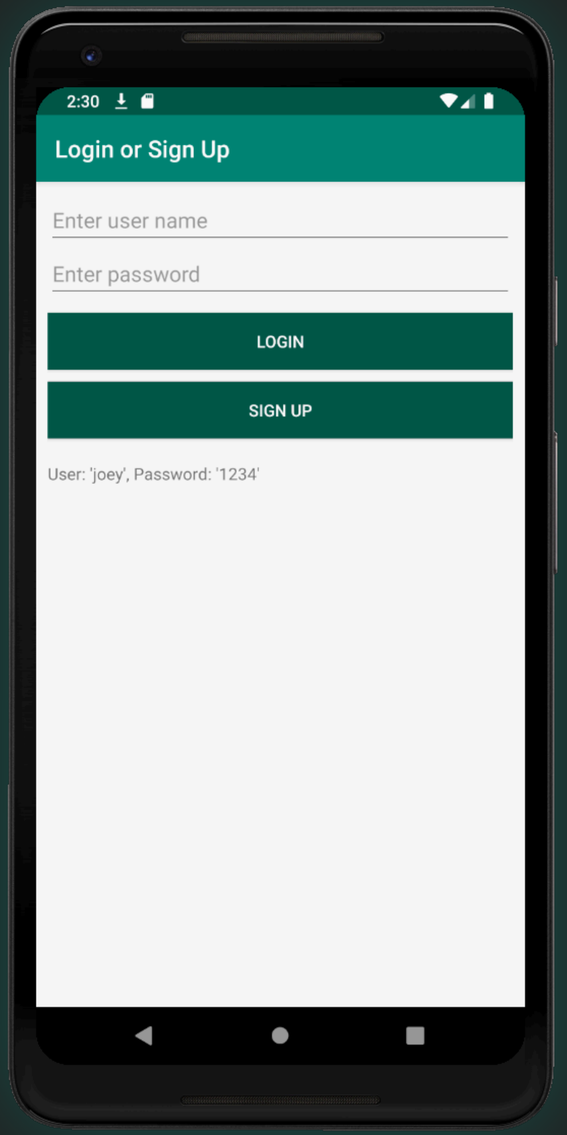
```
        .add(setUpTearDownRule, activityRule)
        .addLast(SetUpTearDownRule().addSetUp {
            activityRule.runOnUiThread {
                SettingsFragmentNavigator.go(BlacklistFragment::class.java)
            }
        })

}

@Before
fun openFragment(){
    activityRule.runOnUiThread {
        SettingsFragmentNavigator.go(BlacklistFragment::class.java)
    }
}

@DisplayName( value: "when contact in blacklist then it displayed in blacklist")
@SetUp(CLEAR_BLACKLIST, ADD_CONTACT_TO_BLACKLIST)
@Test
fun testItemDisplayedInBlacklist() {
    BlacklistPage().assertContactDisplayed(contact.name)
}
```



SOLUTION: USE INTERNAL NAVIGATION MECHANISM

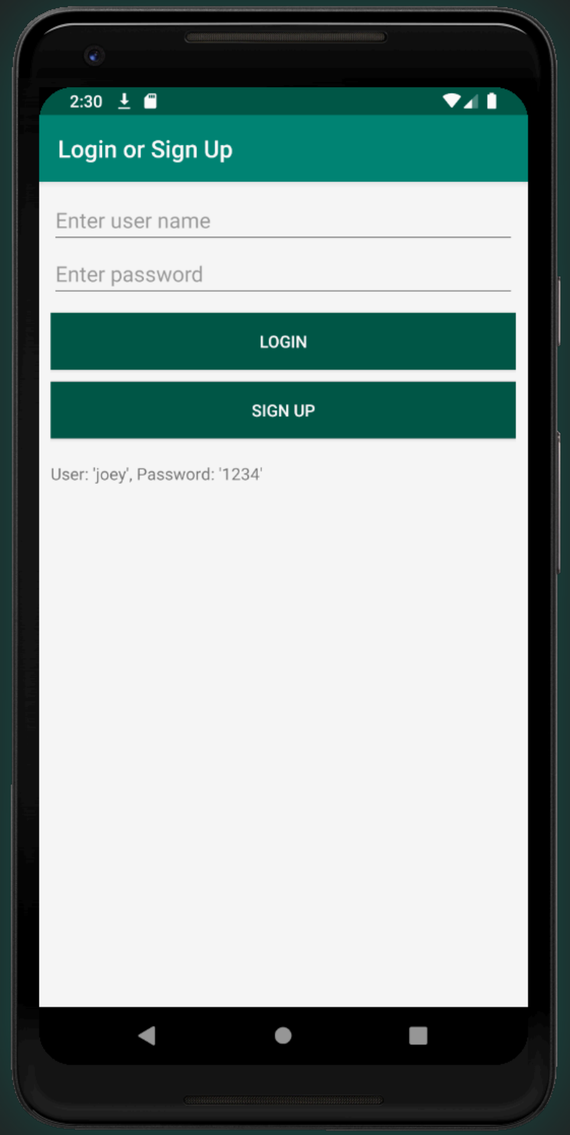
```
activityRule.runOnUiThread {
    SettingsFragmentNavigator.go(BlacklistFragment::class.java)
}
```

WHAT CAN I DO WITH LOGIN PROBLEM?



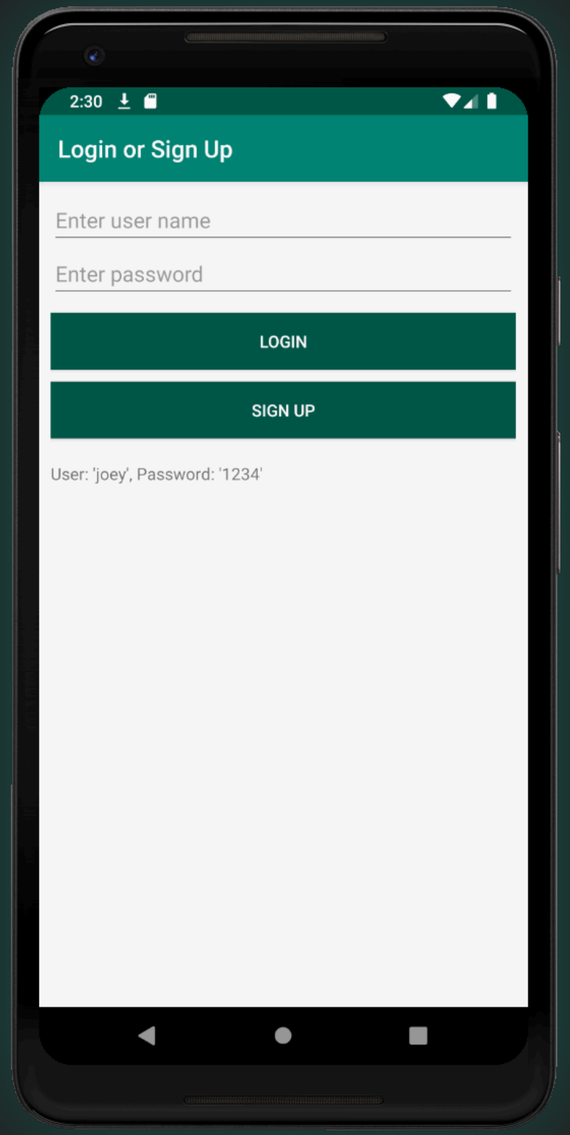
WHAT CAN I DO WITH LOGIN PROBLEM?

- Use authorisation via UI

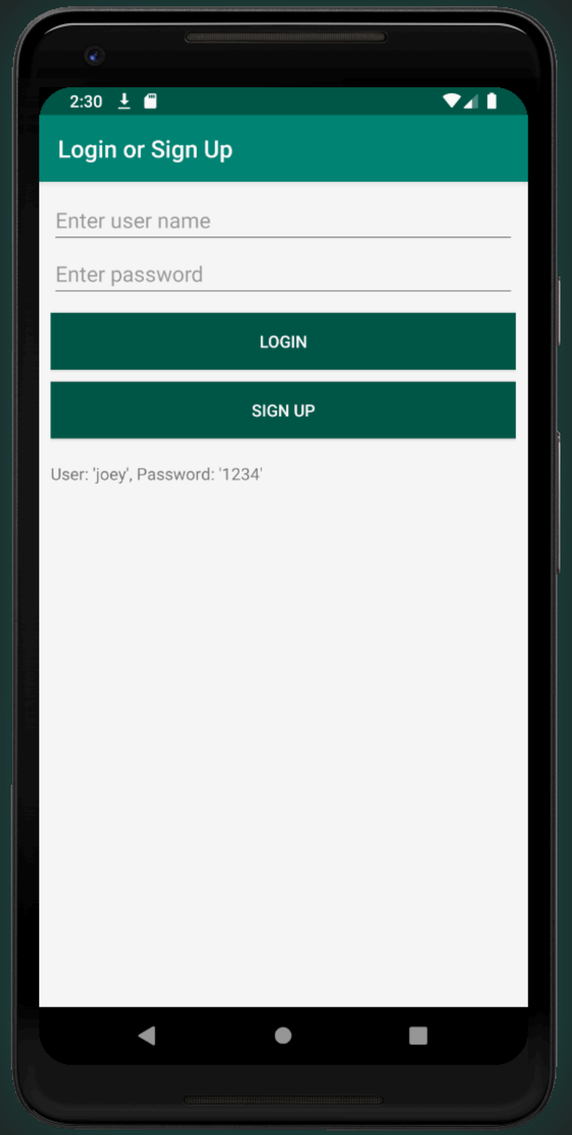


WHAT CAN I DO WITH LOGIN PROBLEM?

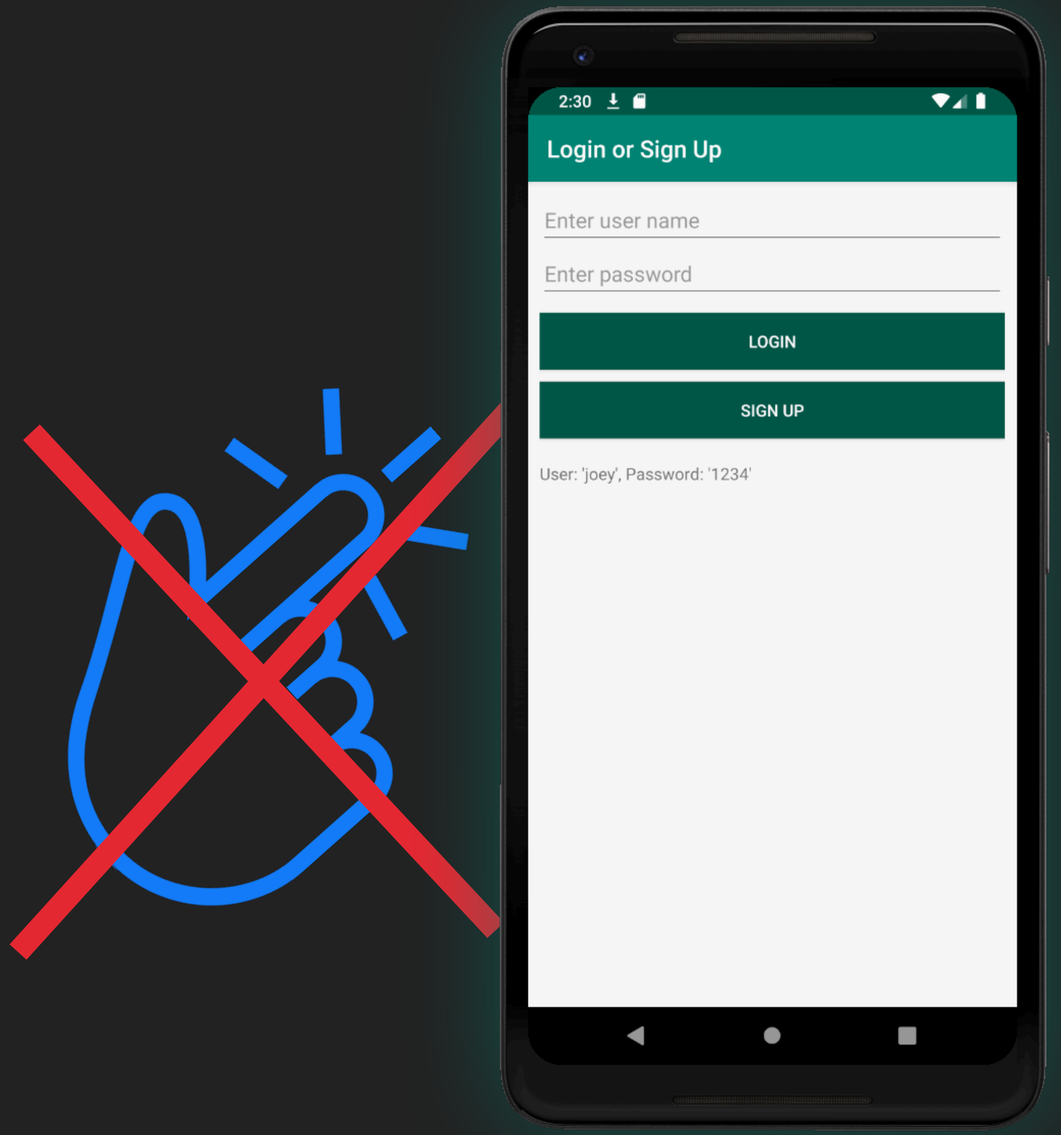
- Use authorisation via UI
- Use internal authorisation mechanism



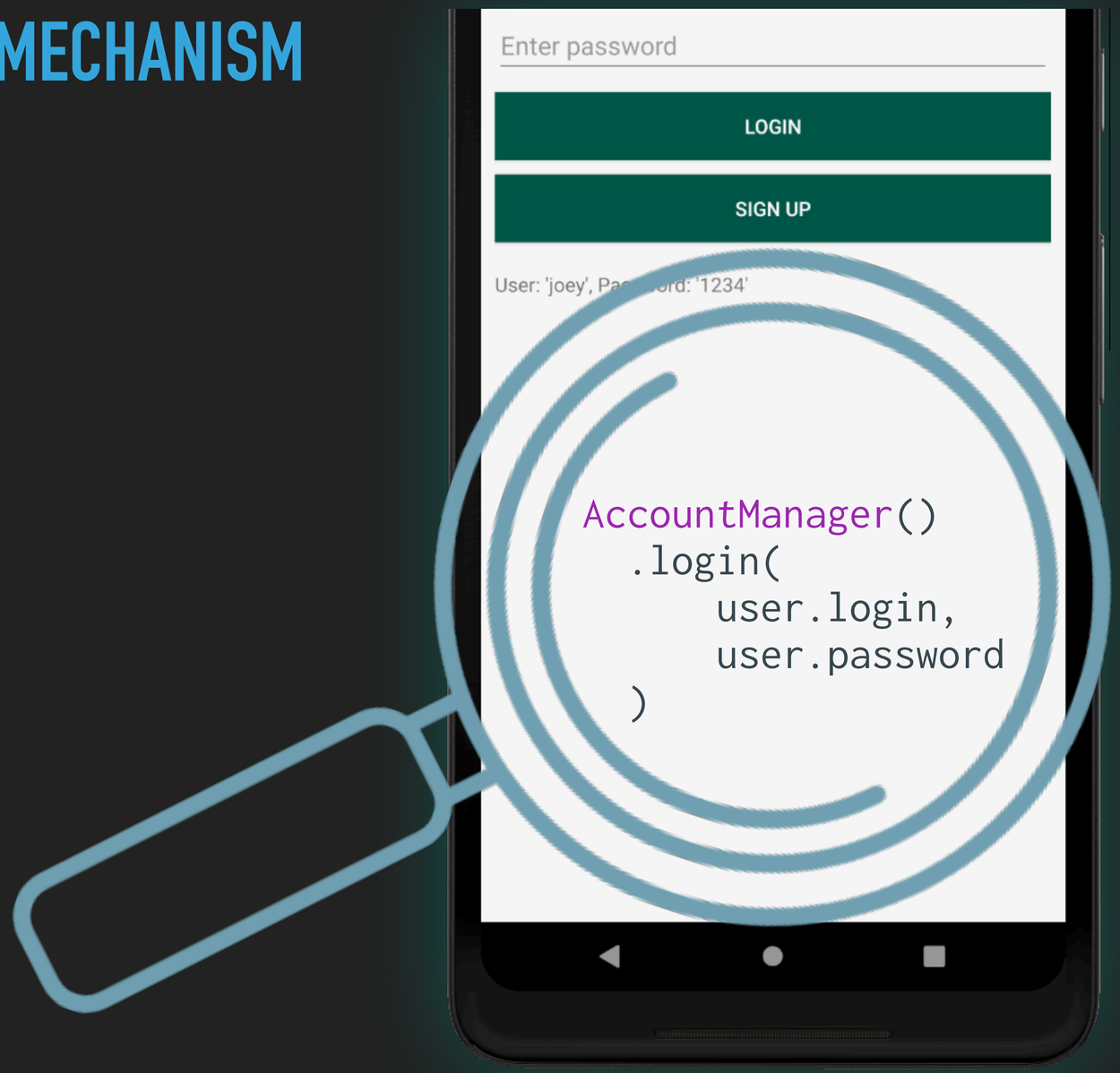
USE AUTHORISATION VIA UI



USE AUTHORISATION VIA UI



USE INTERNAL AUTHORISATION MECHANISM



```
class ChatTest {
    ...
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            MessageRepository.clearChatMessages(contact.id)
            val intent = Intent()
            intent.putExtra("INTENT_CONTACT_ID_EXTRA_NAME", contact.id)
            activityRule.launchActivity(intent)
        }
    @Test
    fun sendMessageWithLink() {... }
}
```

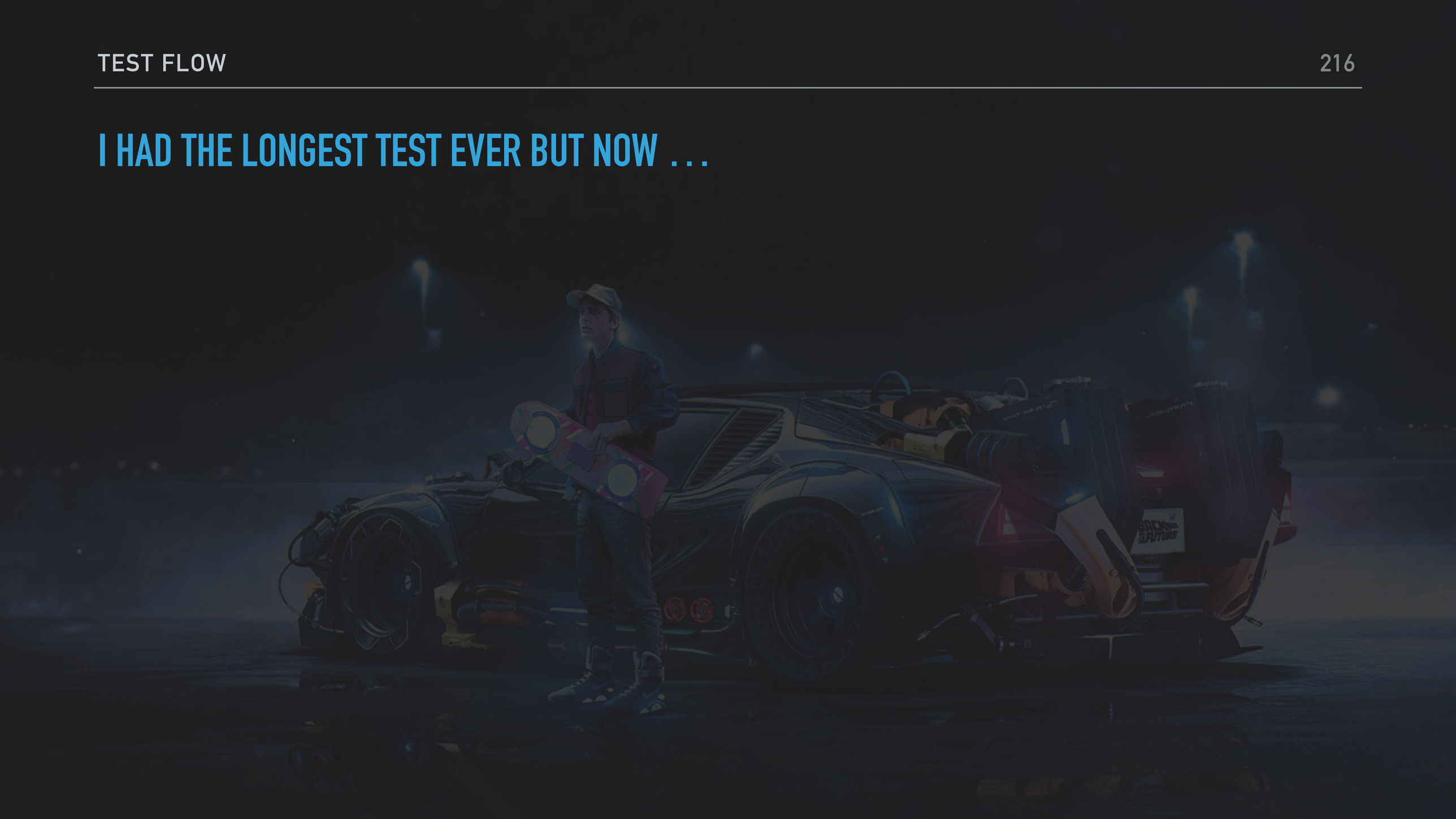
```
class ChatTest {
    ...
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            MessageRepository.clearChatMessages(contact.id)
            ...
        }
    @Test
    fun sendMessageWithLink() {... }
}
```

```
class ChatTest {
    …
    val setupRule = SetUpTearDownRule()
        .addSetUp {
            AccountManager(targetContext).login(user.login, user.password)
            MessageRepository.clearChatMessages(contact.id)
            ...
        }
    @Test
    fun sendMessageWithLink() {... }
}
```

SOLUTION: USE INTERNAL AUTHORISATION MECHANISM

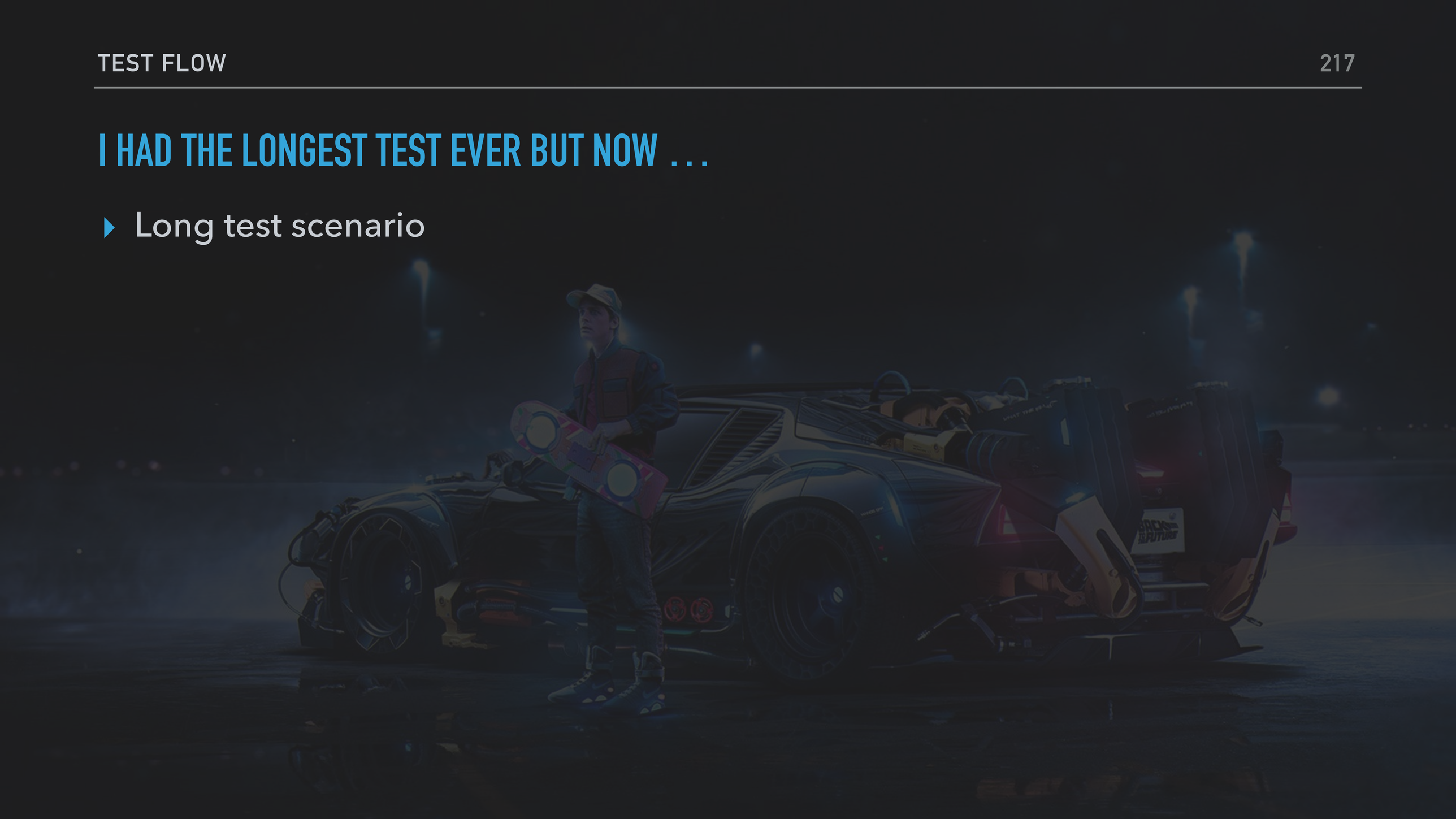
```
AccountManager(targetContext).login(user.login, user.password)
```

I HAD THE LONGEST TEST EVER BUT NOW . . .



I HAD THE LONGEST TEST EVER BUT NOW . . .

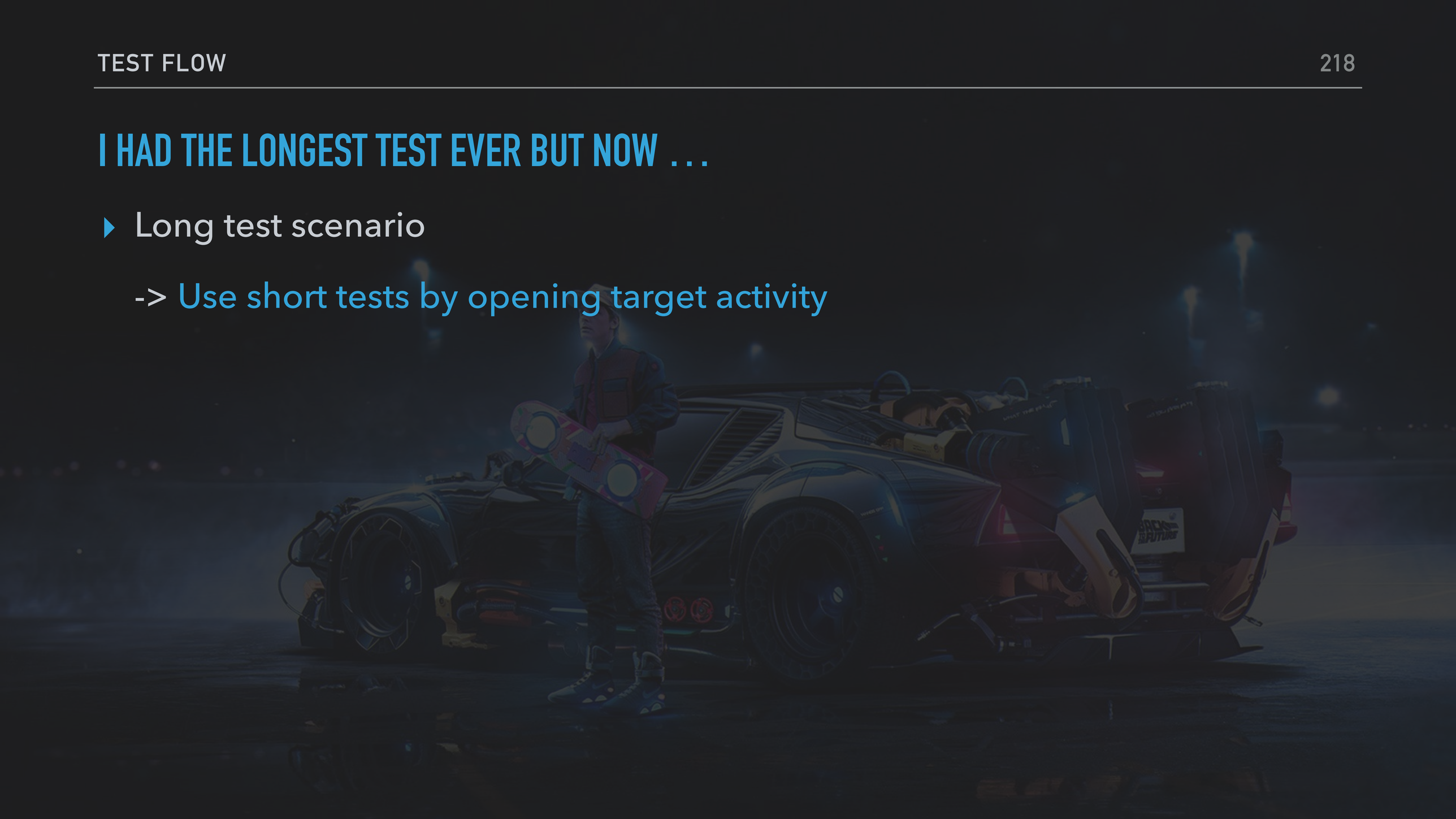
- Long test scenario



I HAD THE LONGEST TEST EVER BUT NOW . . .

- Long test scenario

  -> Use short tests by opening target activity

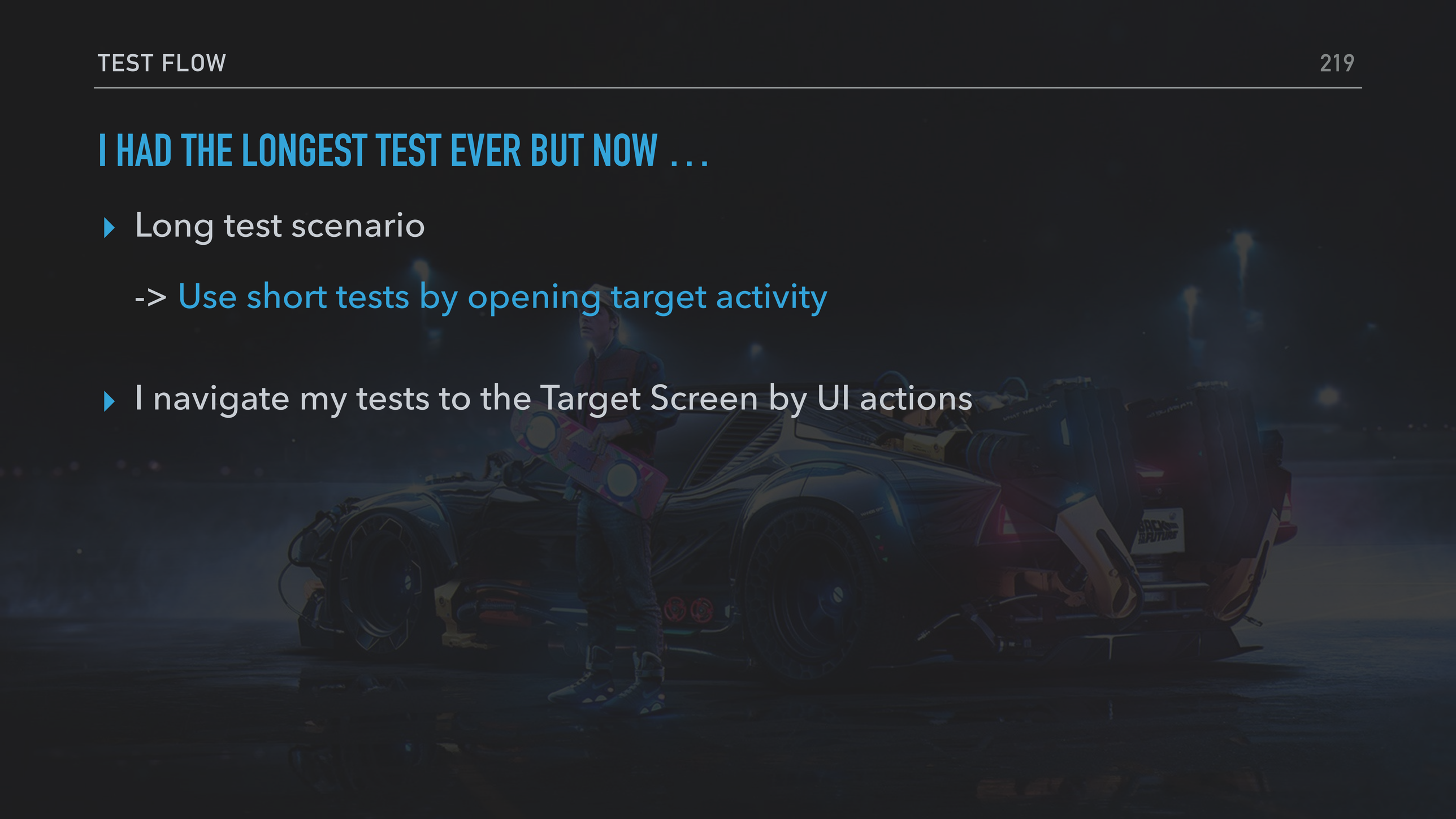


I HAD THE LONGEST TEST EVER BUT NOW . . .

- Long test scenario

  -> Use short tests by opening target activity

- I navigate my tests to the Target Screen by UI actions



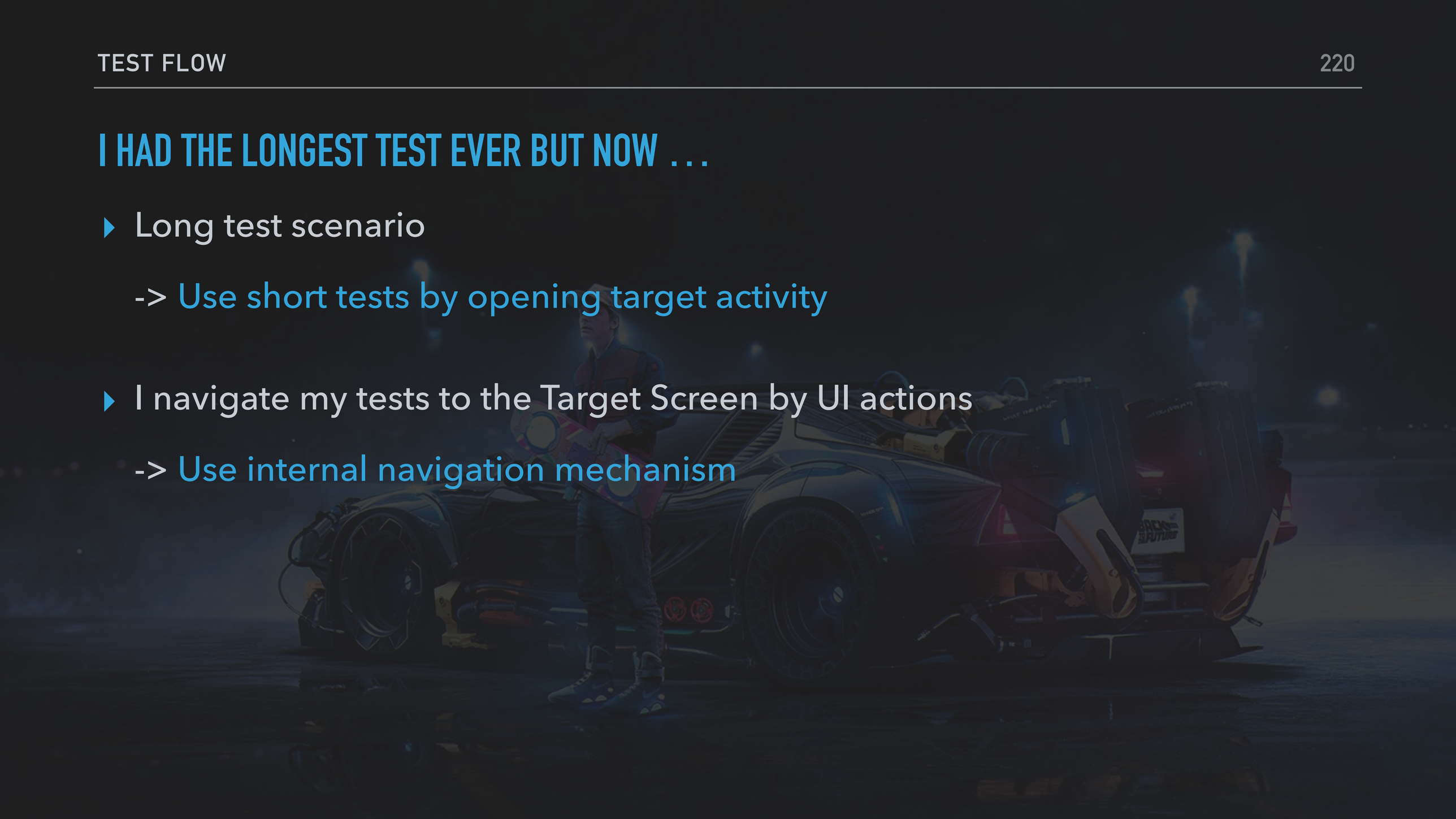
I HAD THE LONGEST TEST EVER BUT NOW . . .

- Long test scenario

  -> Use short tests by opening target activity

- I navigate my tests to the Target Screen by UI actions

  -> Use internal navigation mechanism



I HAD THE LONGEST TEST EVER BUT NOW . . .

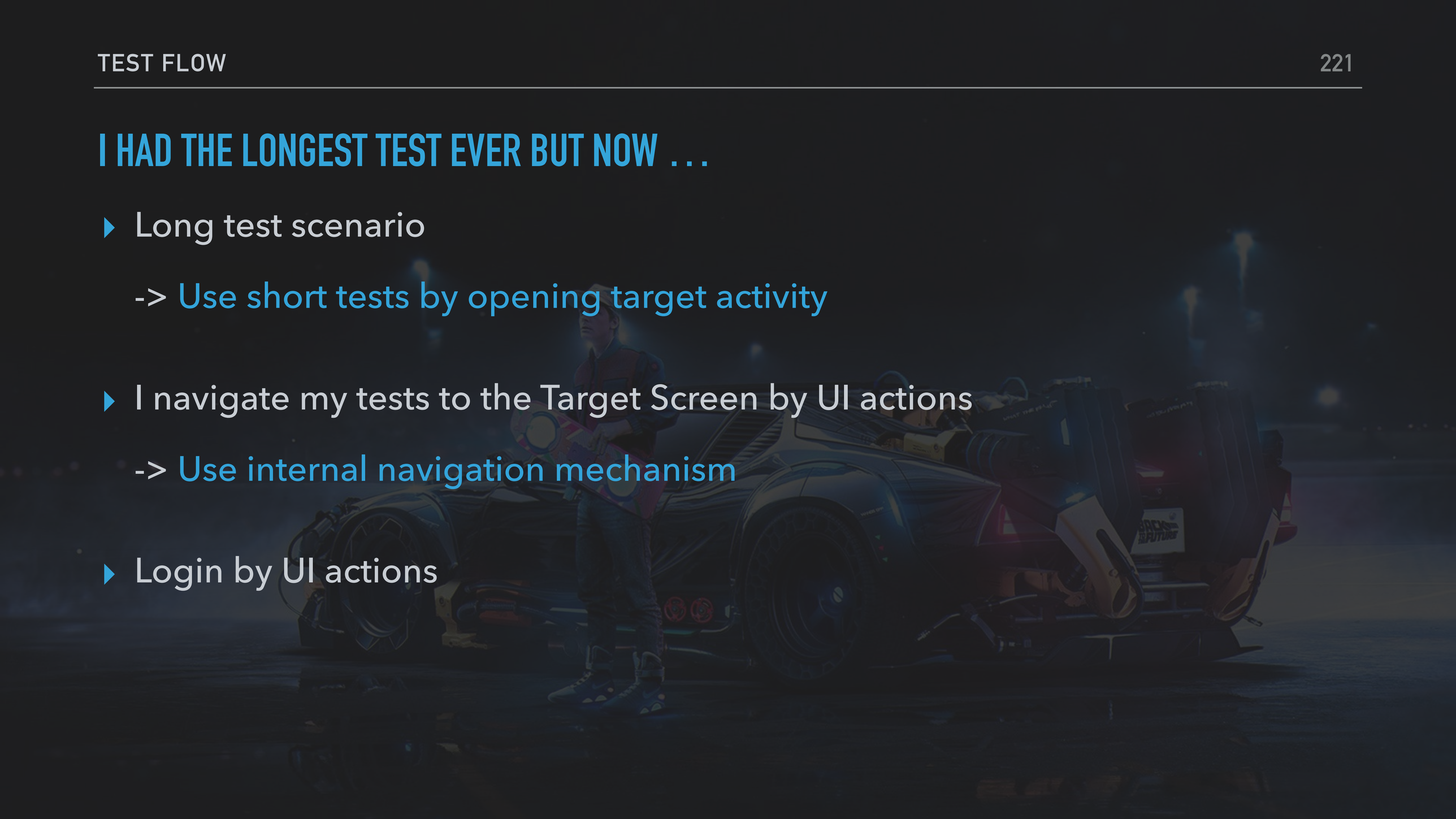
- Long test scenario

  -> Use short tests by opening target activity

- I navigate my tests to the Target Screen by UI actions

  -> Use internal navigation mechanism

- Login by UI actions



I HAD THE LONGEST TEST EVER BUT NOW . . .

- Long test scenario

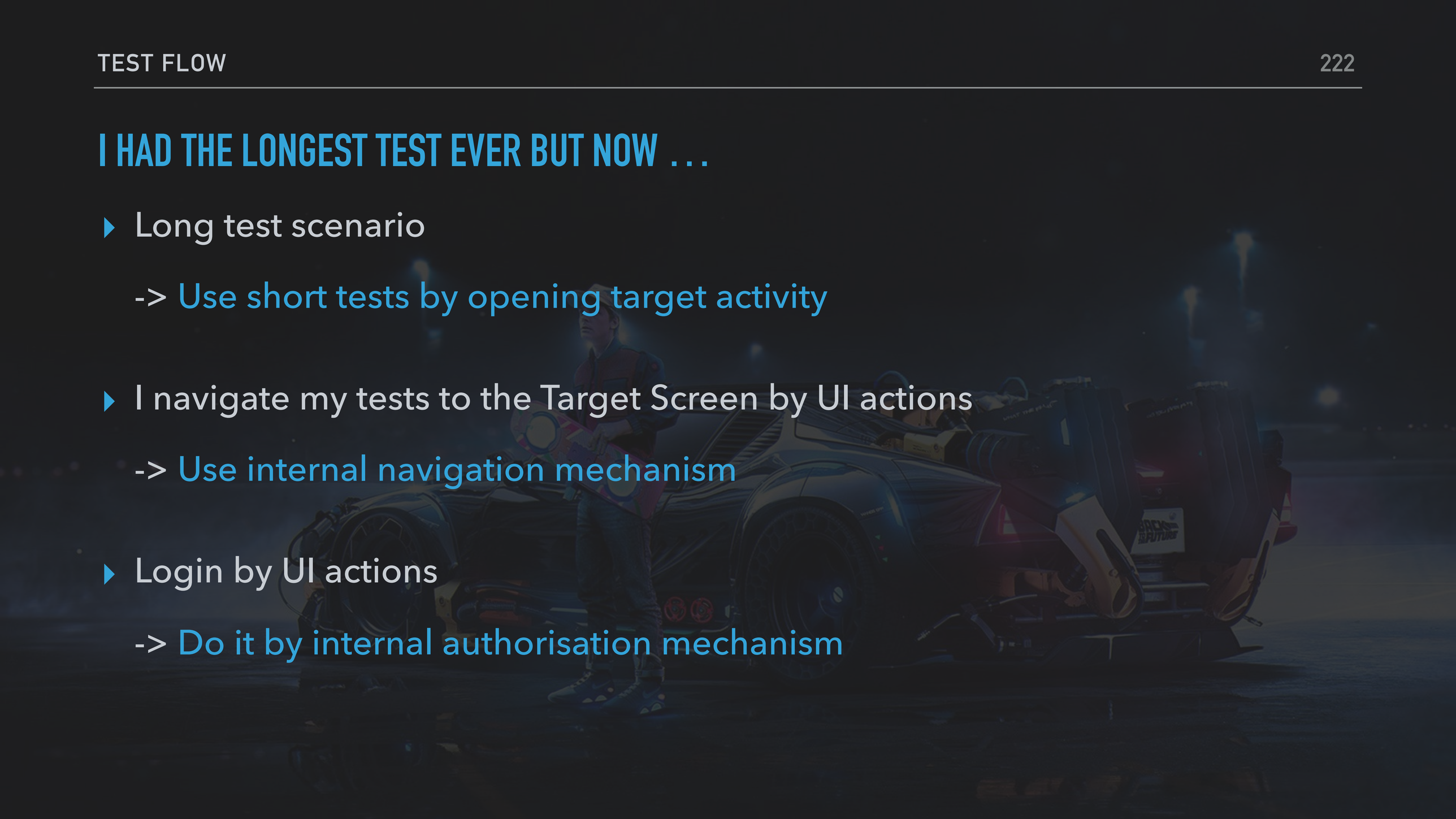
  -> Use short tests by opening target activity

- I navigate my tests to the Target Screen by UI actions

  -> Use internal navigation mechanism

- Login by UI actions

  -> Do it by internal authorisation mechanism



I HAD THE LONGEST TEST EVER BUT NOW I HAVE THE SHORTEST ONE

- Long test scenario

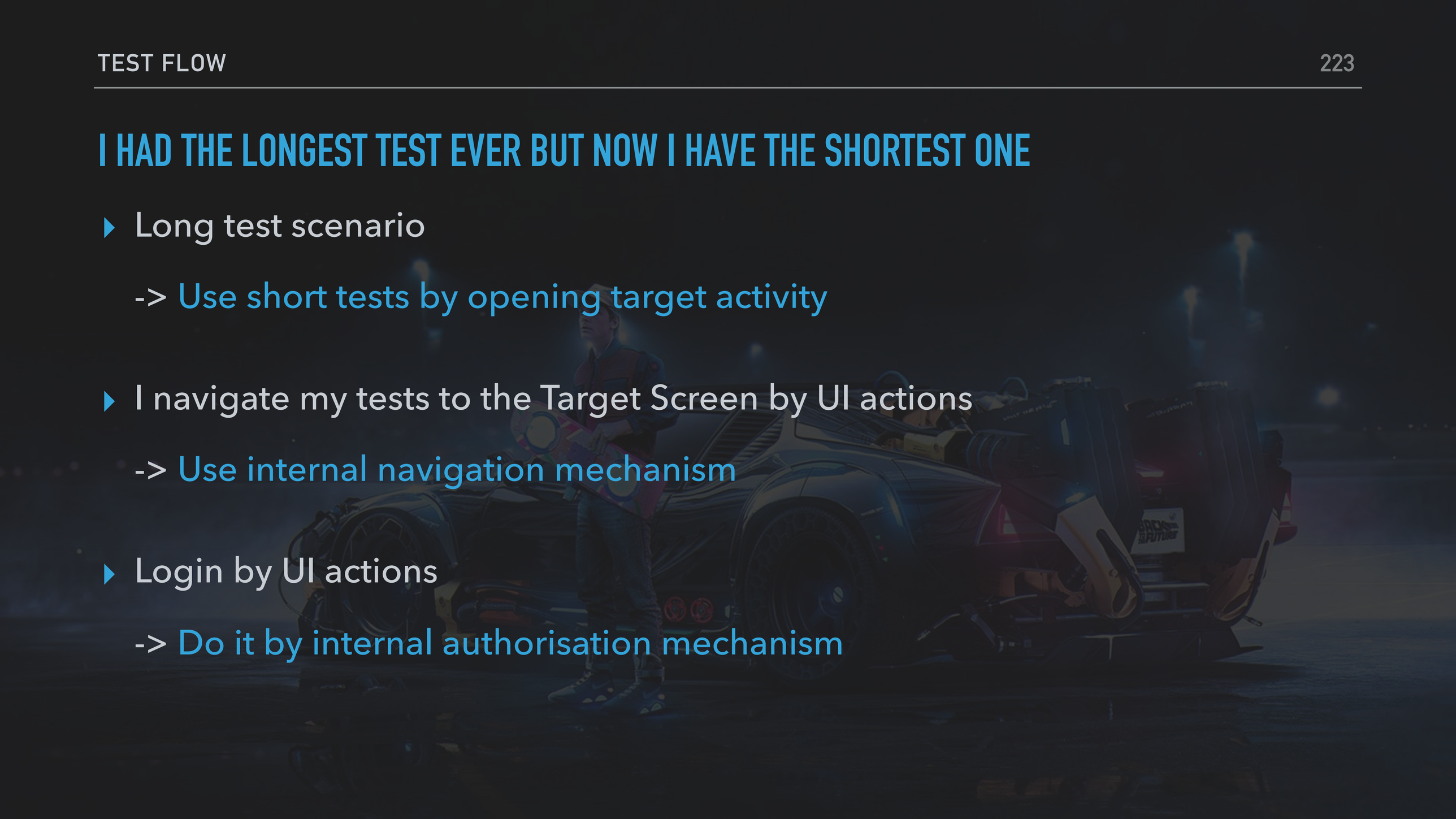
  -> Use short tests by opening target activity

- I navigate my tests to the Target Screen by UI actions

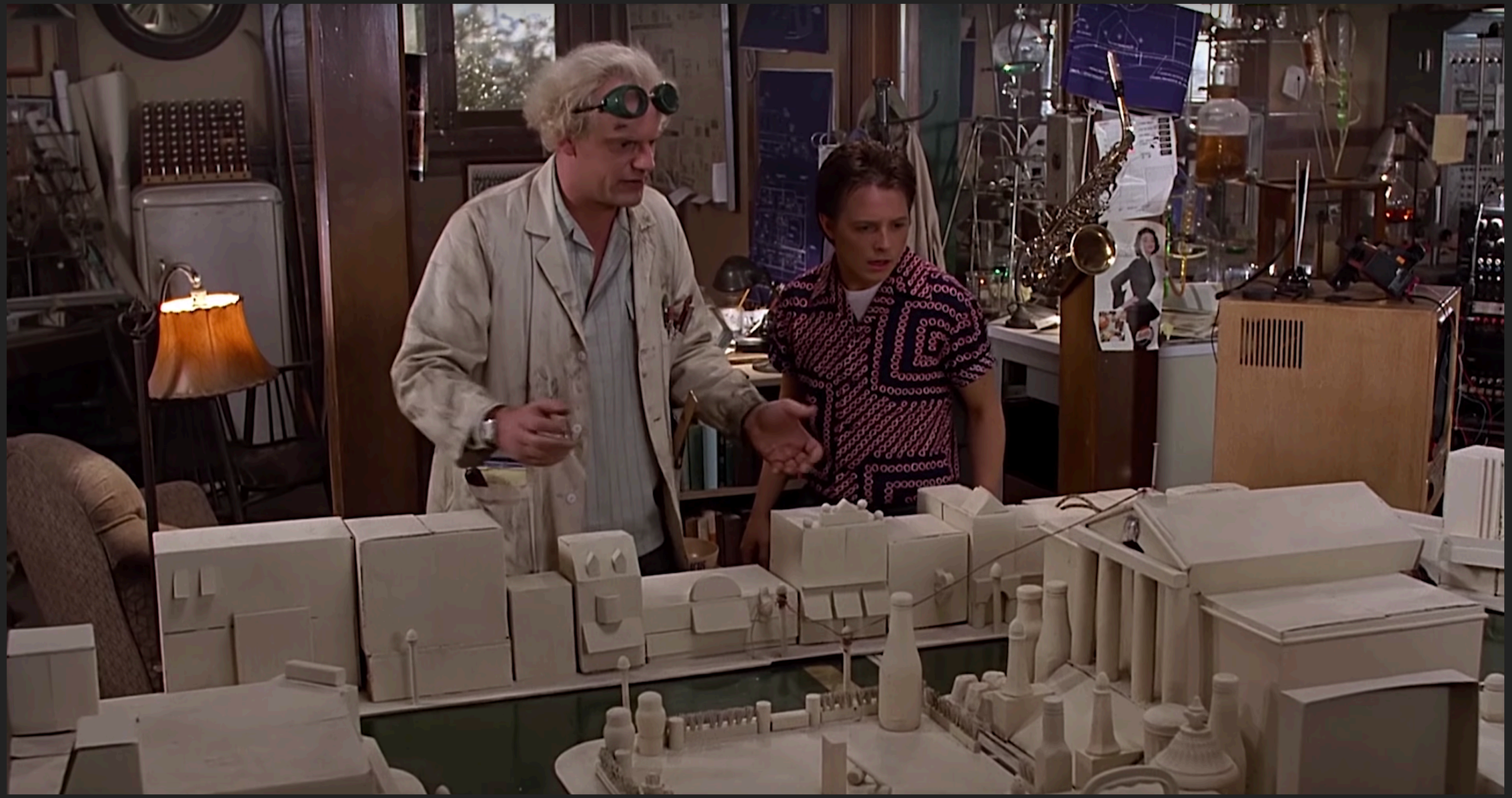
  -> Use internal navigation mechanism

- Login by UI actions

  -> Do it by internal authorisation mechanism

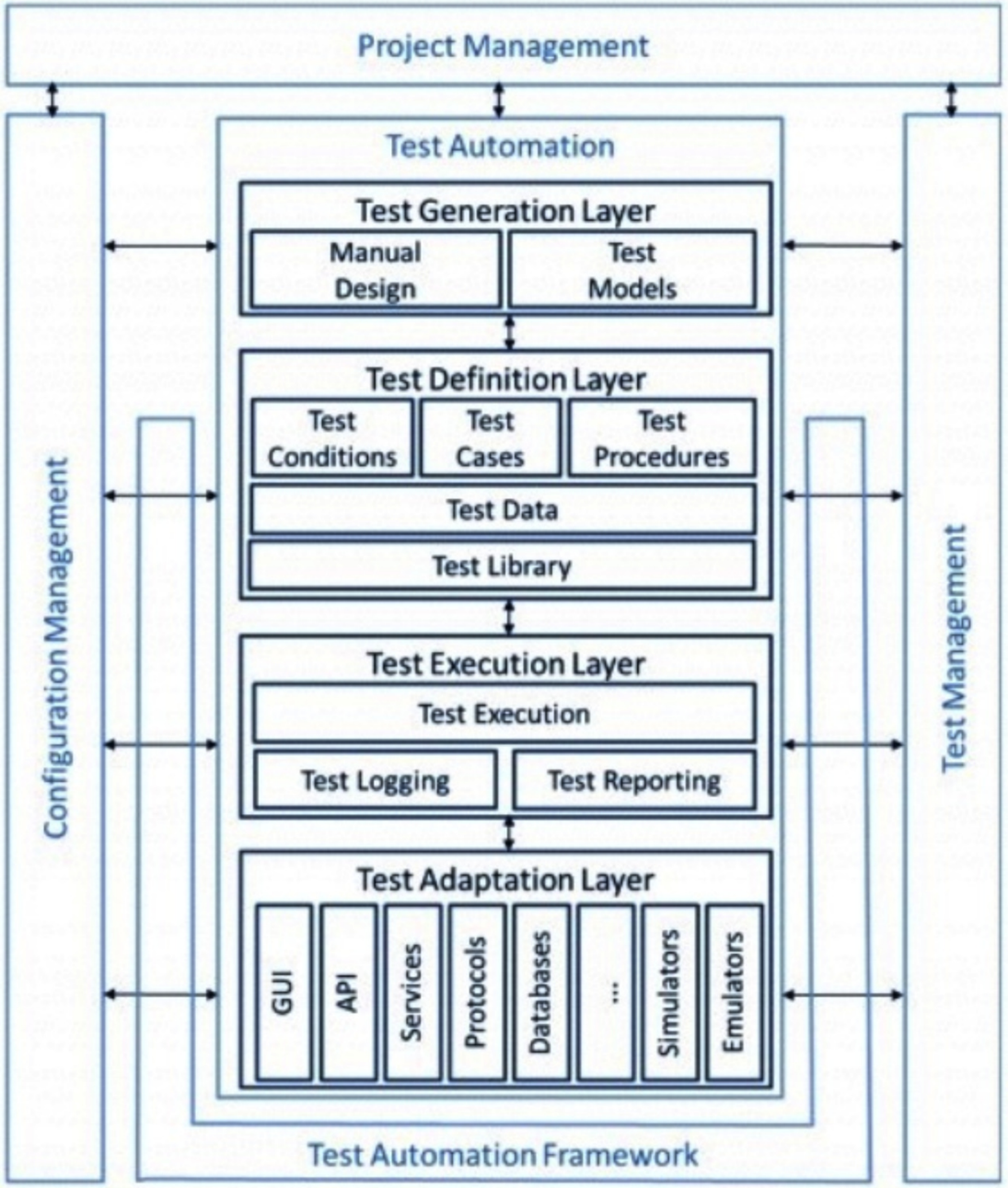


TEST FRAMEWORK ARCHITECTURE

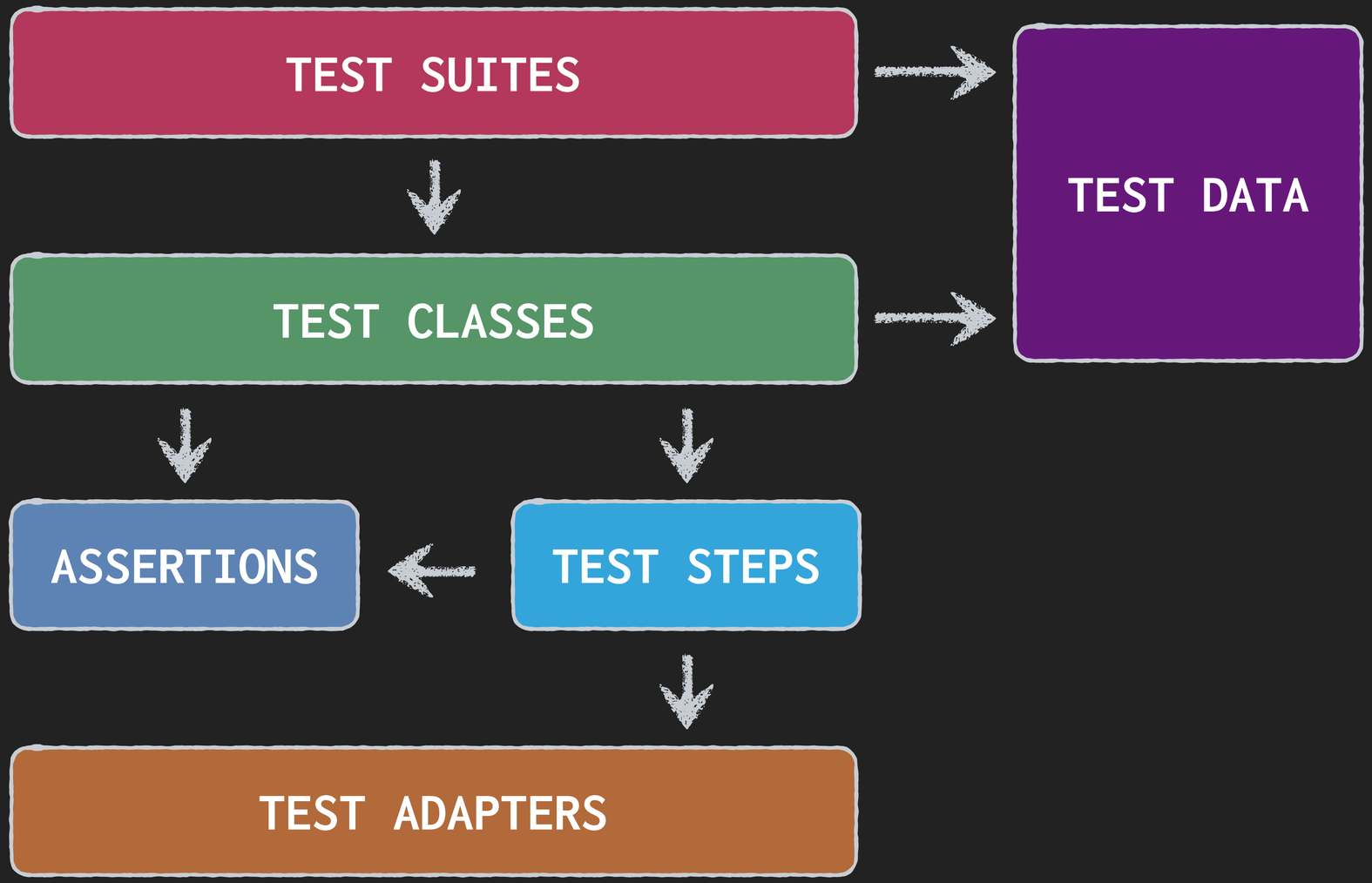


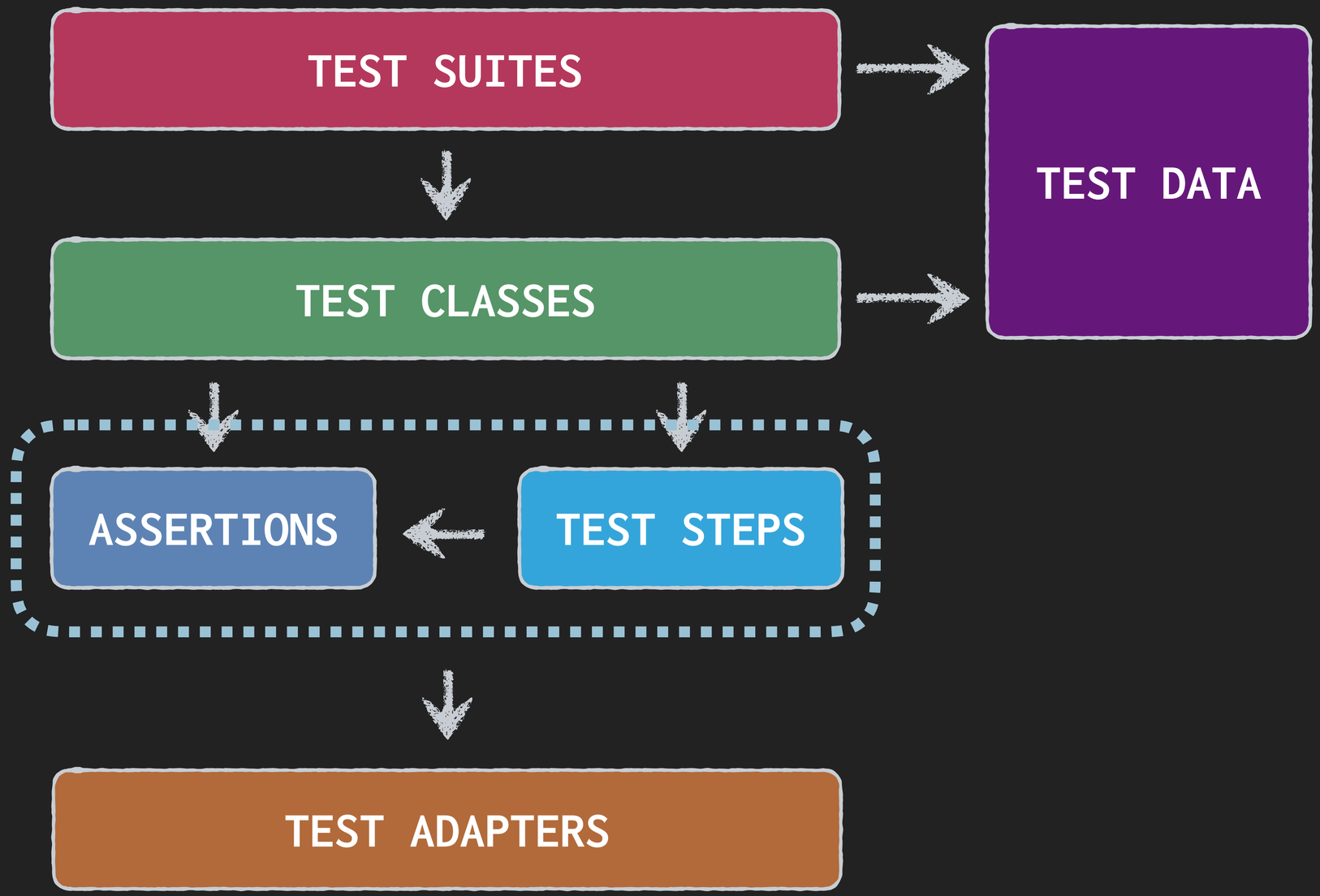
MY FIRST TEST FRAMEWORK

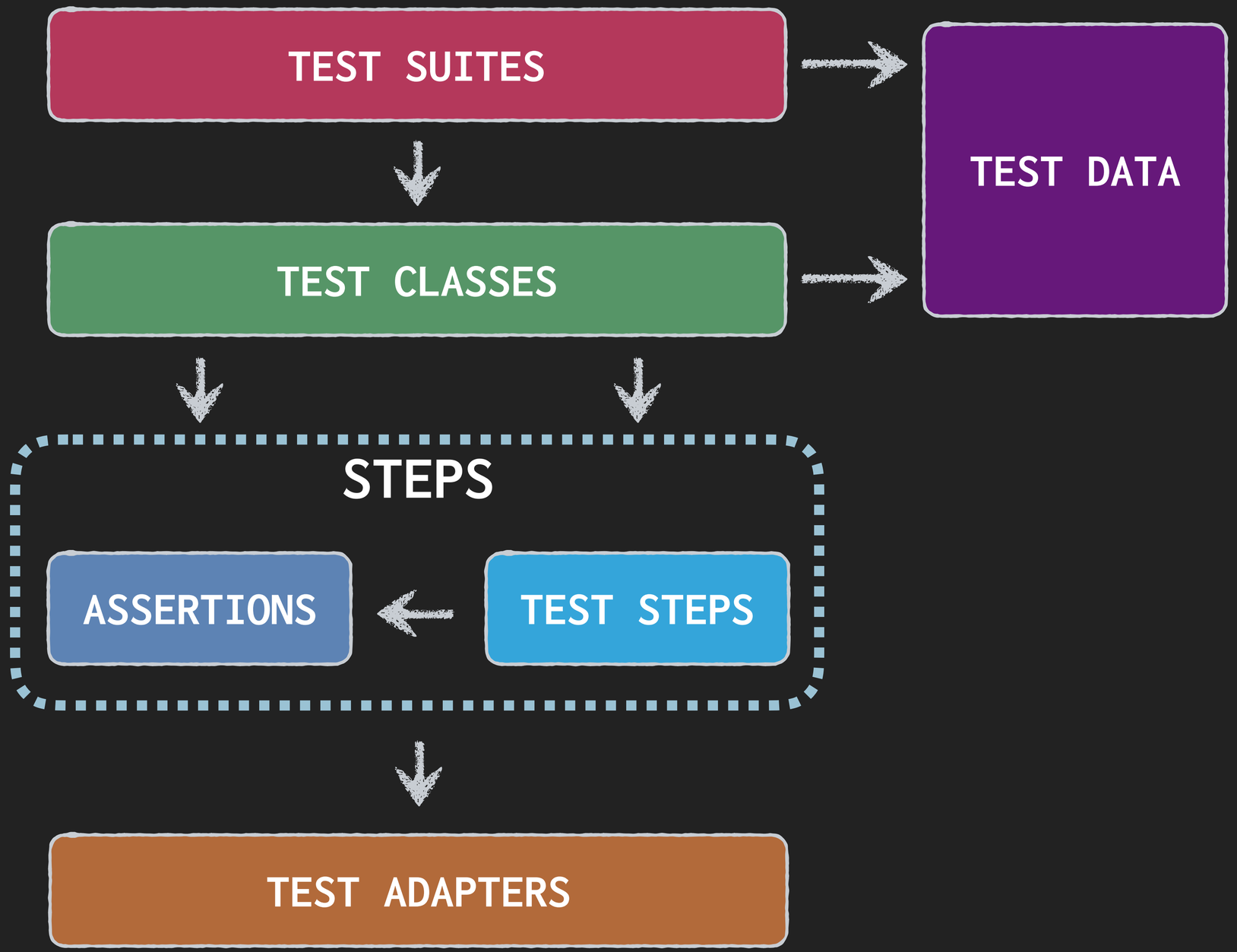
ISTQB Generic Test Automation Architecture

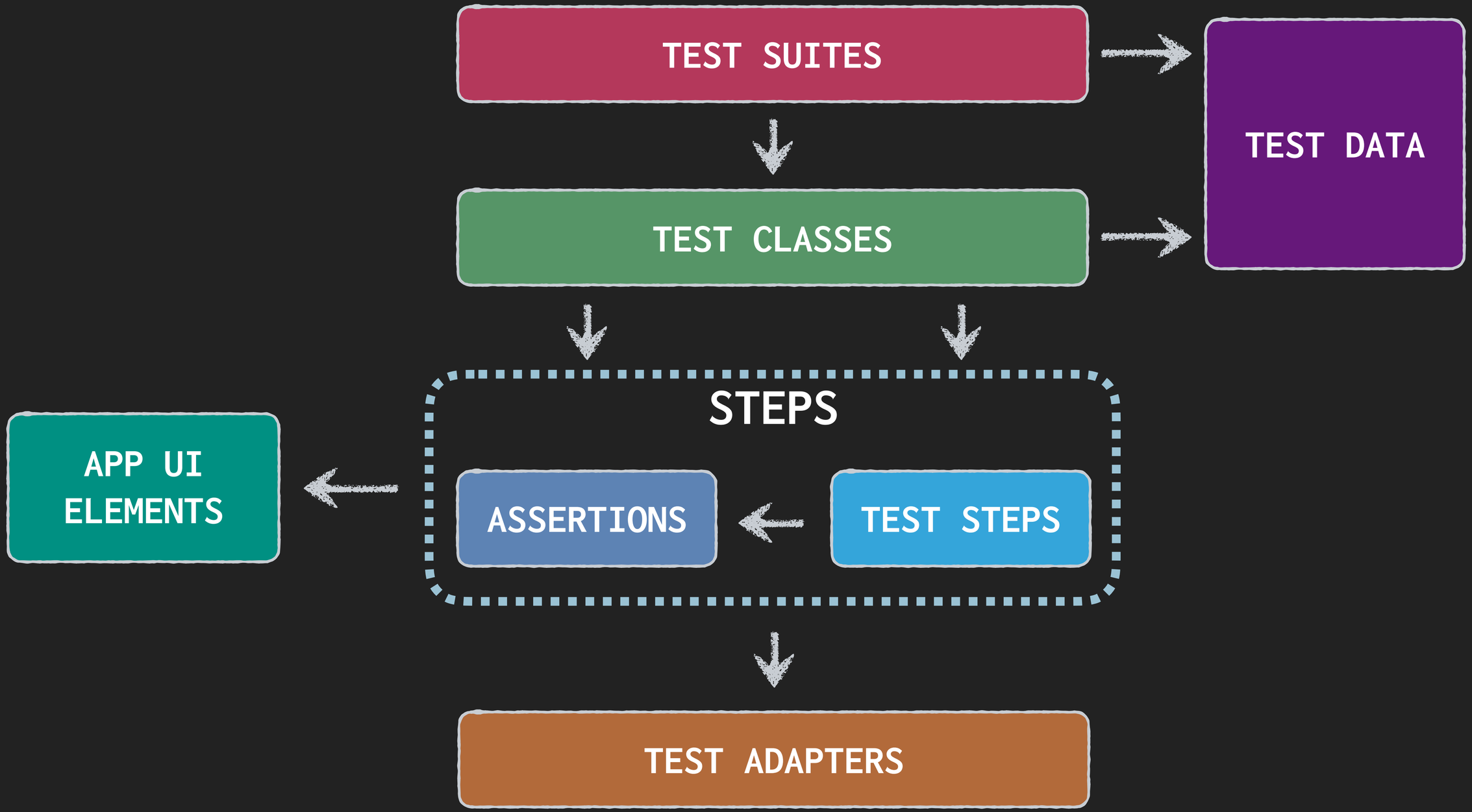


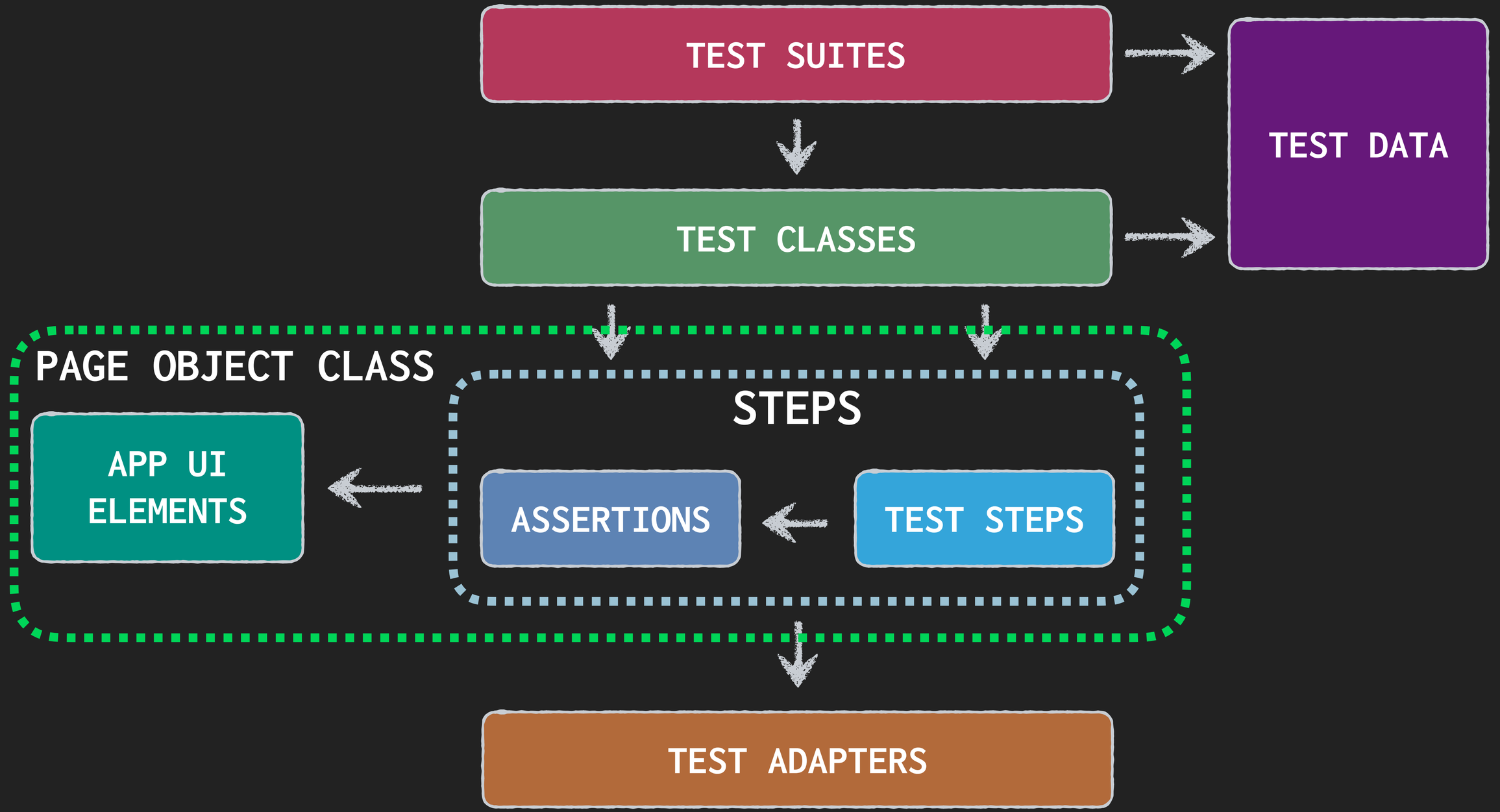
ISTQB International Software Testing Qualifications Board


TEST SUITES
TEST DATA
TEST CLASSES
ASSERTIONS
TEST STEPS
TEST ADAPTERS


TEST SUITES
TEST DATA
TEST CLASSES
ASSERTIONS
TEST STEPS
TEST ADAPTERS


TEST SUITES
TEST DATA
TEST CLASSES
STEPS
ASSERTIONS
TEST STEPS
TEST ADAPTERS


TEST SUITES
TEST DATA
TEST CLASSES
STEPS
APP UI ELEMENTS
ASSERTIONS
TEST STEPS
TEST ADAPTERS


TEST SUITES
TEST DATA
TEST CLASSES
PAGE OBJECT CLASS
STEPS
APP UI ELEMENTS
ASSERTIONS
TEST STEPS
TEST ADAPTERS

MY FIRST TEST FRAMEWORK

MY CURRENT TEST FRAMEWORK

MAKING YOUR OWN REPORT

IT WILL LOOKS LIKE THIS


KEDYDINAMO.RU
BUT IT LOOKS LIKE THIS


ULM
OR LIKE THIS

WHY IT HAPPENS?

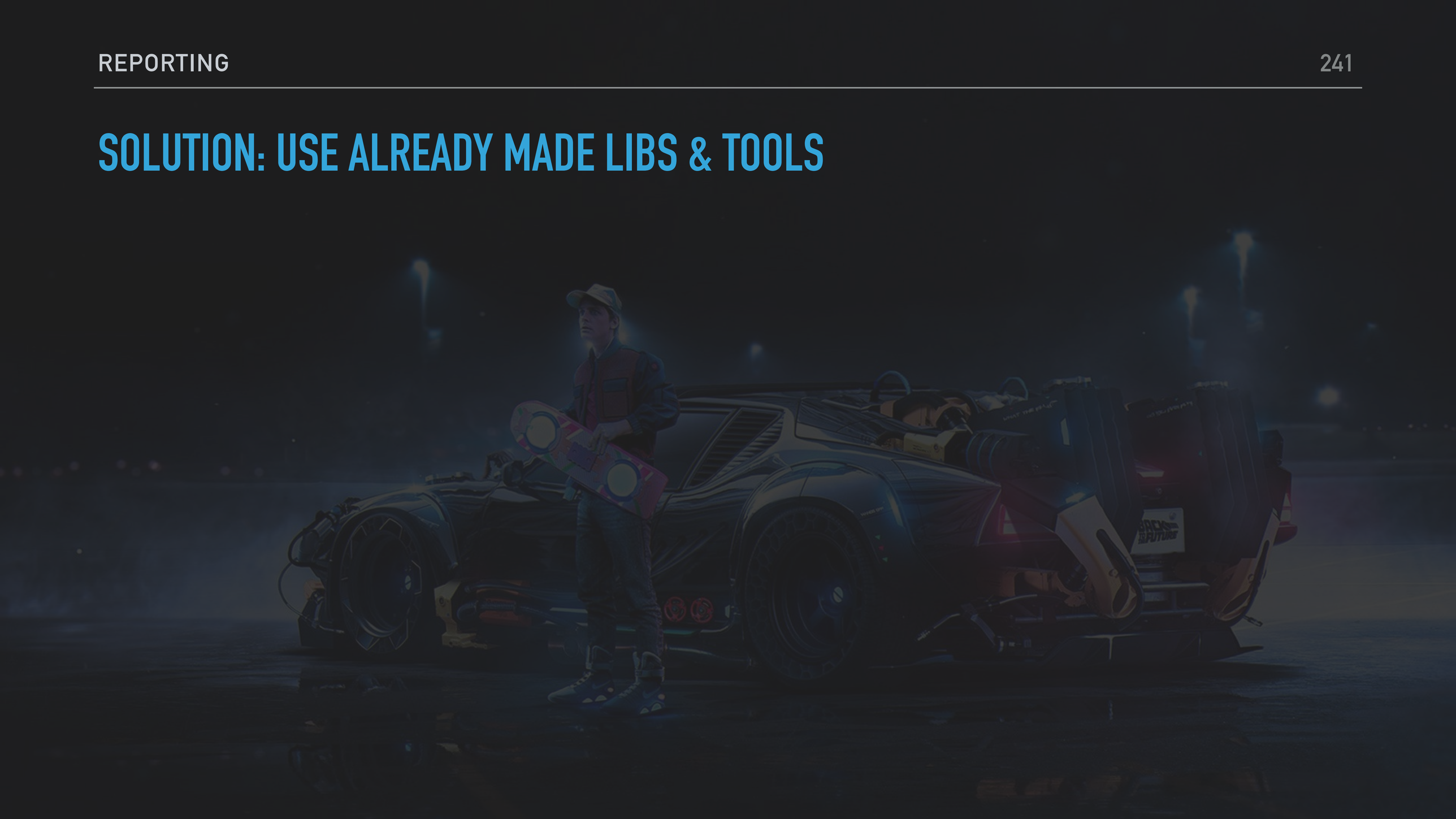
WHY IT HAPPENS?

- Support hell

WHY IT HAPPENS?

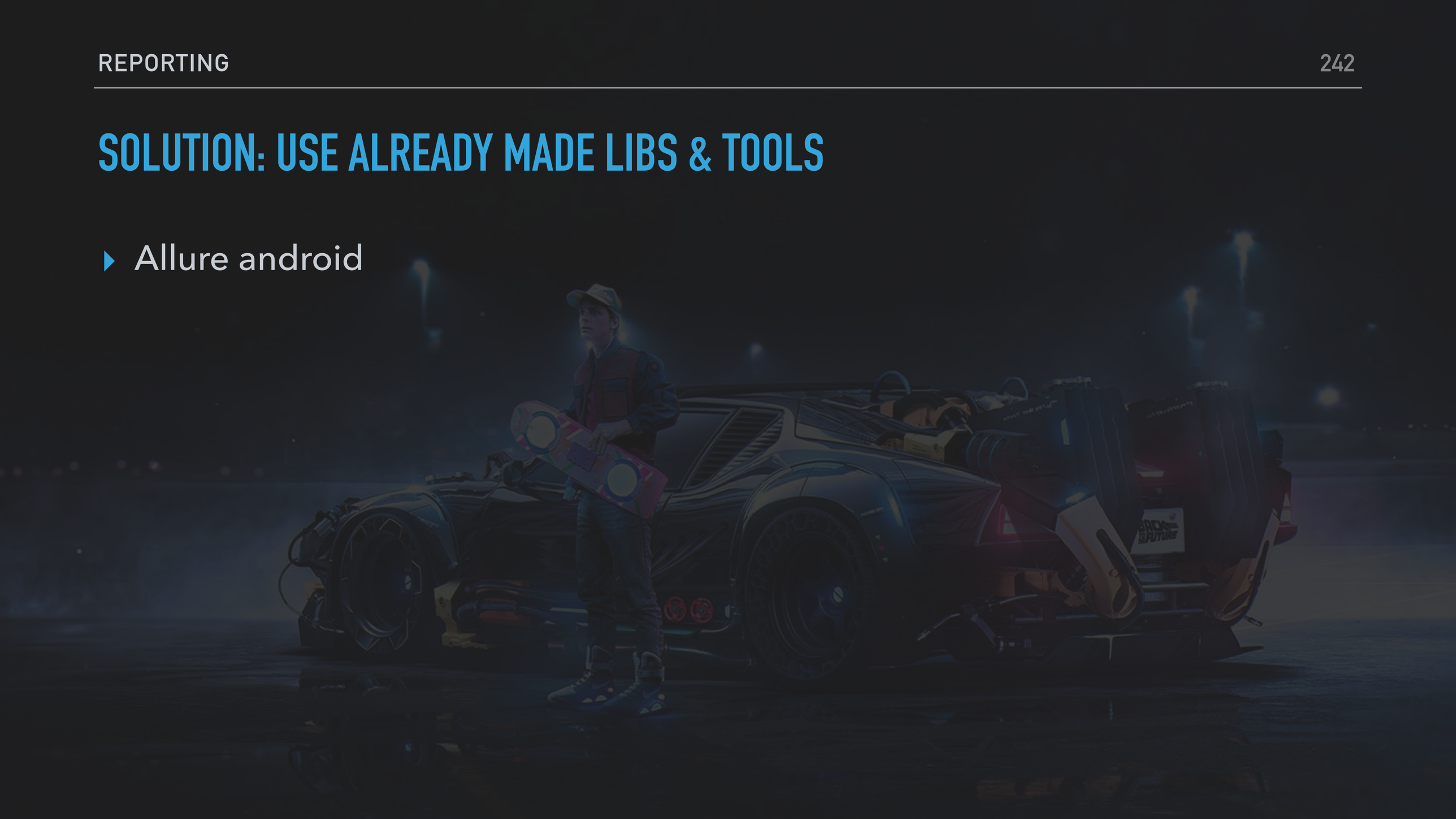
- Support hell
- Integration with CI hell

SOLUTION: USE ALREADY MADE LIBS & TOOLS



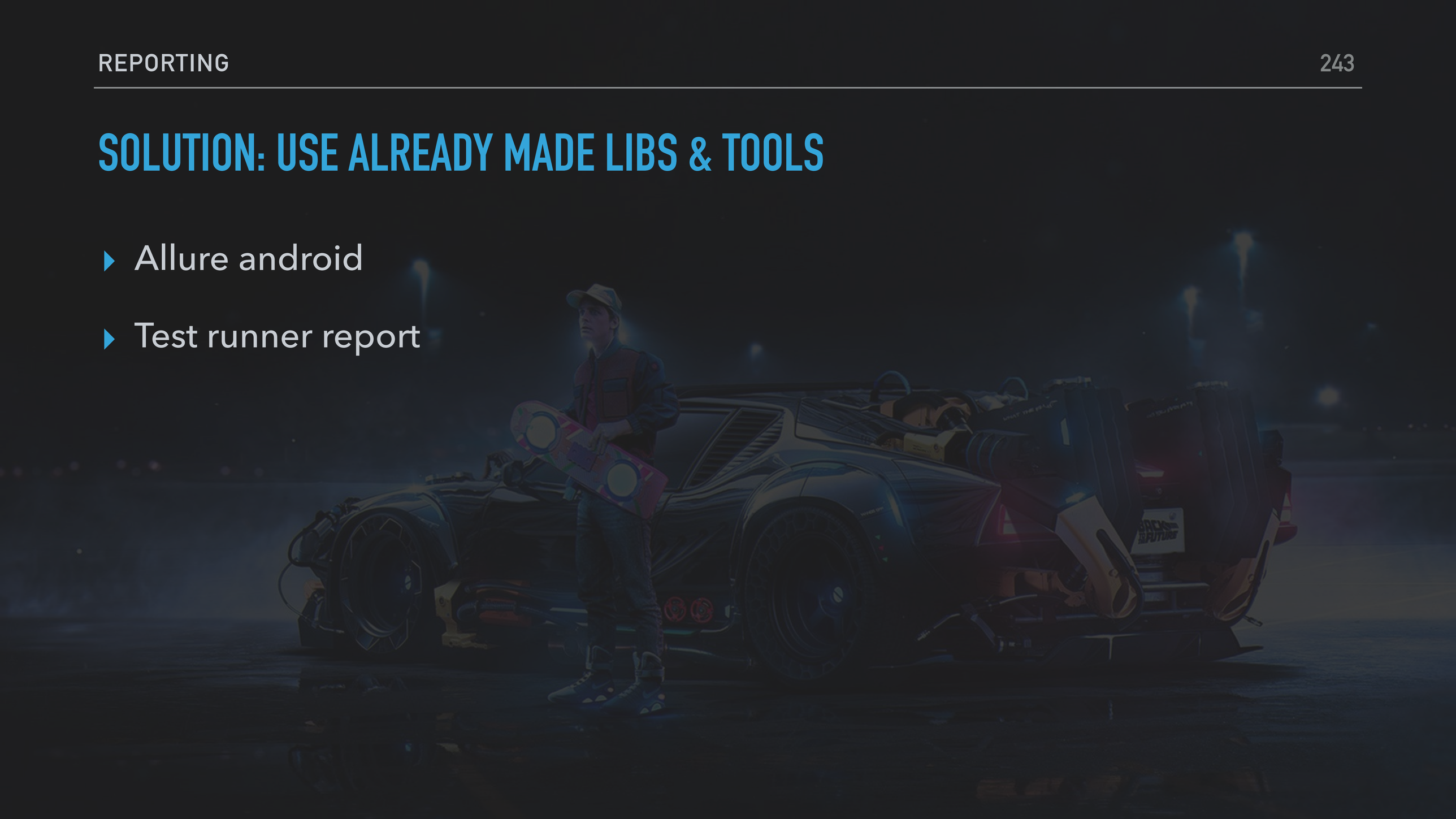
SOLUTION: USE ALREADY MADE LIBS & TOOLS

- Allure android



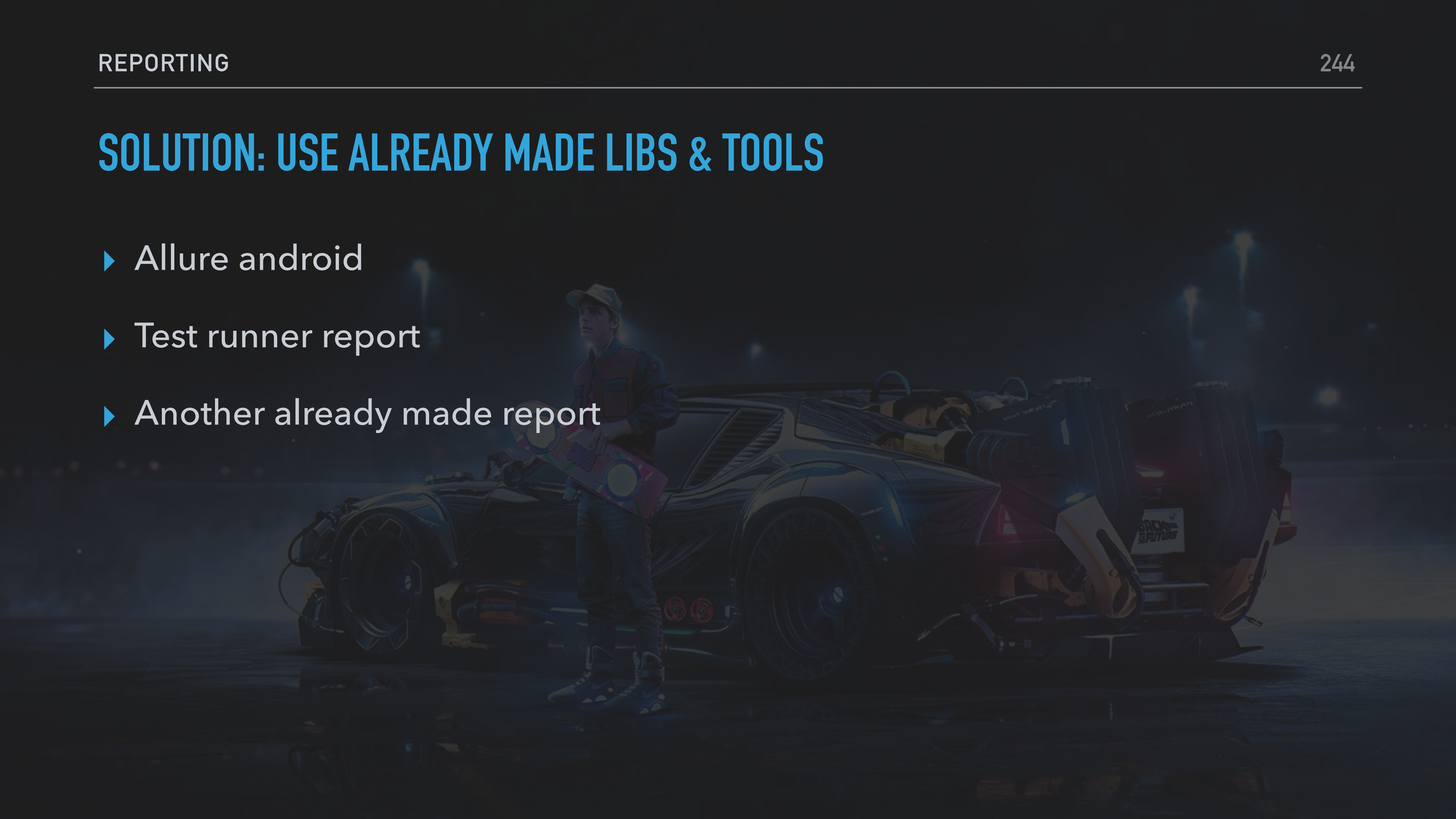
SOLUTION: USE ALREADY MADE LIBS & TOOLS

- Allure android
- Test runner report



SOLUTION: USE ALREADY MADE LIBS & TOOLS

- Allure android
- Test runner report
- Another already made report



TYPES OF MISTAKES WE'VE DISCOVERED

TYPES OF MISTAKES WE'VE DISCOVERED

- Test data usage

- Test data is used everywhere:
  -> Strict data definition to test suite or test case
  -> Store read-only data as constants
- Test data is defined after activity launch
  -> Use custom SetUpTearDown rule
- Test data is defined in @BeforeClass or @Before method for all @Test methods
  -> Use SetUpTearDown rule + SetUp and TearDown annotations
- Test user is related with device Build.Serial
  -> Use TestData server

TYPES OF MISTAKES WE'VE DISCOVERED

- Test data usage
- Test flow

I HAD THE LONGEST TEST EVER BUT NOW I HAVE THE SHORTEST ONE

- Long test scenario

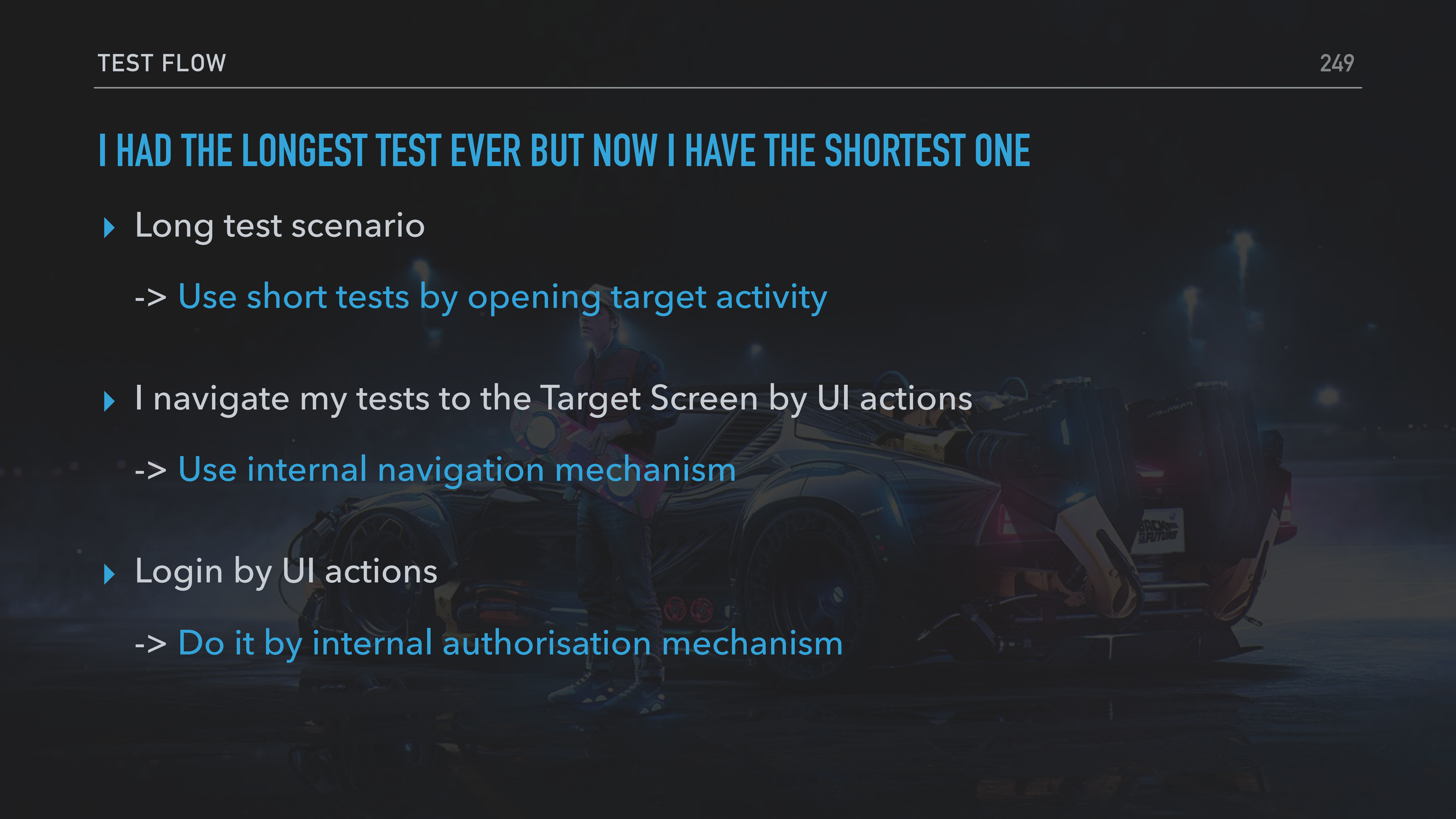
  -> Use short tests by opening target activity

- I navigate my tests to the Target Screen by UI actions

  -> Use internal navigation mechanism

- Login by UI actions

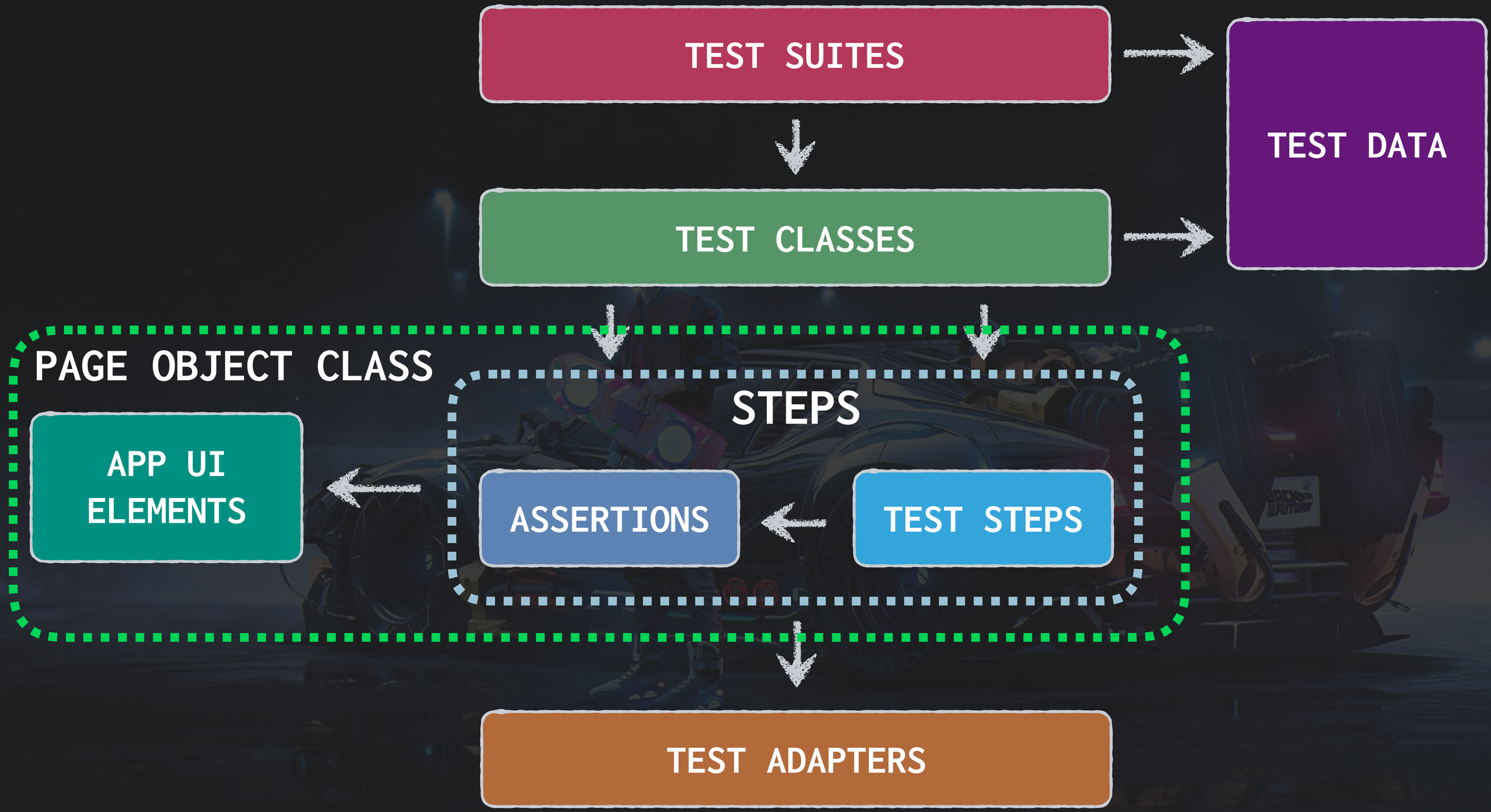
  -> Do it by internal authorisation mechanism



TYPES OF MISTAKES WE'VE DISCOVERED

- Test data usage
- Test flow
- Framework architecture

SOLUTION: USE COMMON PRACTICES AND ADAPT IT

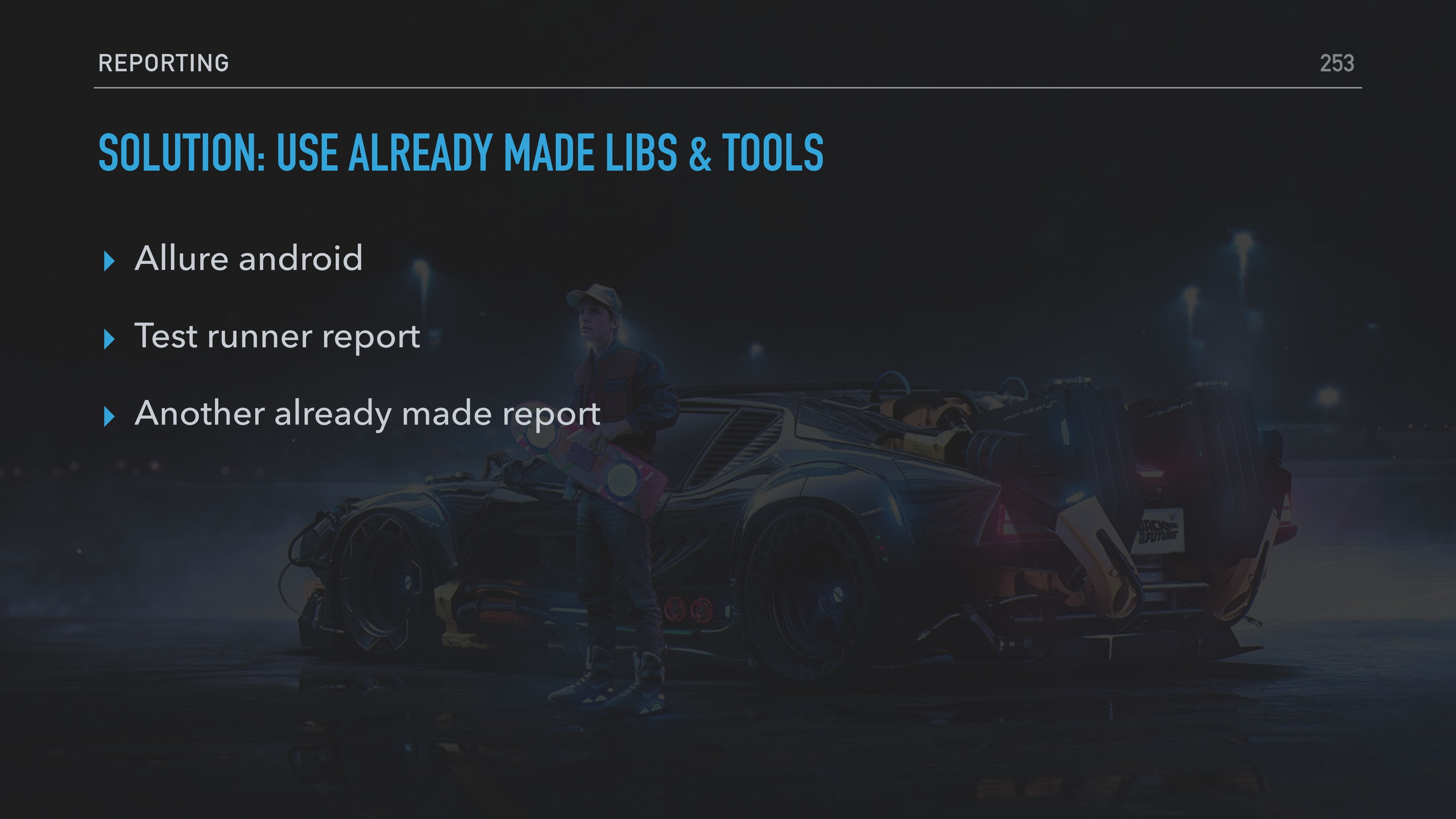


TYPES OF MISTAKES WE'VE DISCOVERED

- Test data usage
- Test flow
- Framework architecture
- Reporting

SOLUTION: USE ALREADY MADE LIBS & TOOLS

- Allure android
- Test runner report
- Another already made report



GitHub:
1) Примеры кода
github.com/alex-tiurin/espresso-guide
2) Библиотека с Espresso DSL
github.com/alex-tiurin/espresso-page-object

Как подключить себе в проект?

```
repositories { jcenter() }
dependencies {
    //espresso-page-object
    androidTestImplementation 'com.atiurin.espresso:espressopageobject:0.1.16'
}
```