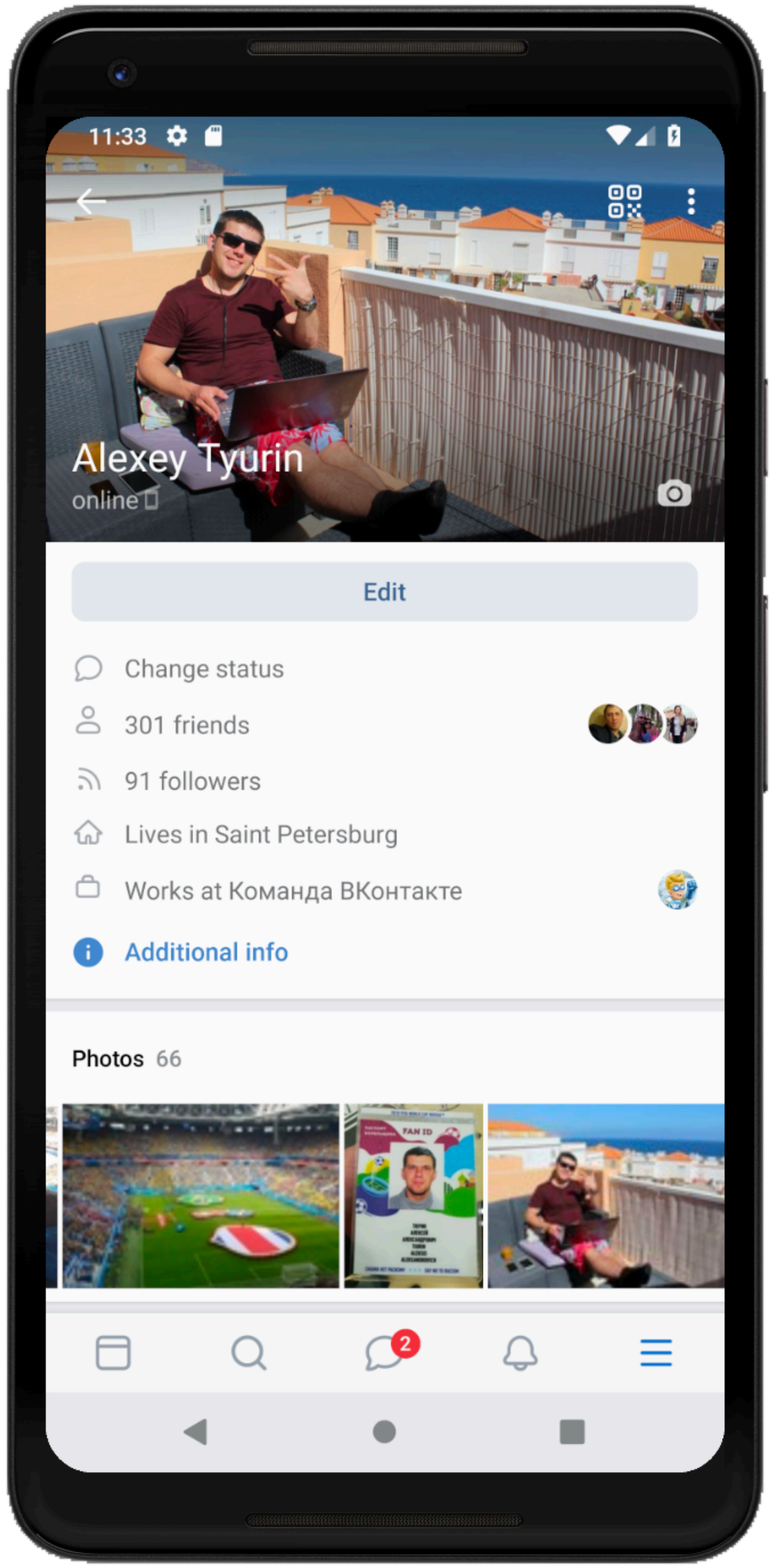
Решаем проблемы Espresso автотестов Android в реальном мире

Алексей Тюрин
vk.com/alex_tiurin

Алексей Тюрин

Lead Android Automation Engineer at VK

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



11:33

Alexey Tyurin

online

Edit

Change status

301 friends

91 followers

Lives in Saint Petersburg

Works at Команда ВКонтакте

Additional info

Photos 66

UI TESTING FOR ANDROID
espresso

UI TESTING FOR ANDROID
espresso
appium

Espresso

Appium

- ☆ Выполняется на устройстве

Espresso

- Выполняется на устройстве

Appium

- Выполняется на хосте

Espresso

Appium

⭐ Выполняется на устройстве

⭐ Выполняется на хосте

✓ Быстрый

Espresso

☆ Выполняется на устройстве

✓ Быстрый

Appium

☆ Выполняется на хосте

x Медленный

Espresso

☆ Выполняется на устройстве

Appium

☆ Выполняется на хосте

✓ Быстрый

х Одно устройство

х Медленный

Espresso

Appium

☆ Выполняется на устройстве

☆ Выполняется на хосте

✓ Быстрый

x Одно устройство

x Медленный

✓ Несколько устройств

Espresso

- ☆ Выполняется на устройстве
- ☆ Есть доступ к коду

✓ Быстрый

x Одно устройство

Appium

- ☆ Выполняется на хосте

x Медленный

✓ Несколько устройств

Espresso

- ☆ Выполняется на устройстве
- ☆ Есть доступ к коду

✓ Быстрый

x Одно устройство

Appium

- ☆ Выполняется на хосте
- ☆ Нет доступ к коду

x Медленный

✓ Несколько устройств

Espresso

- ☆ Выполняется на устройстве
- ☆ Есть доступ к коду

✓ Быстрый

x Одно устройство

✓ Интеграция с App

Appium

- ☆ Выполняется на хосте
- ☆ Нет доступ к коду

x Медленный

✓ Несколько устройств

Espresso

☆ Выполняется на устройстве

☆ Есть доступ к коду

✓ Быстрый

x Одно устройство

✓ Интеграция с App

Appium

☆ Выполняется на хосте

☆ Нет доступ к коду

x Медленный

✓ Несколько устройств

x Ограниченный

Espresso

- ☆ Выполняется на устройстве
- ☆ Есть доступ к коду
- ☆ Java & Kotlin

✓ Быстрый

x Одно устройство

✓ Интеграция с App

Appium

- ☆ Выполняется на хосте
- ☆ Нет доступ к коду

x Медленный

✓ Несколько устройств

x Ограниченный

Espresso

☆ Выполняется на устройстве

☆ Есть доступ к коду

☆ Java & Kotlin

✓ Быстрый

x Одно устройство

✓ Интеграция с App

Appium

☆ Выполняется на хосте

☆ Нет доступ к коду

☆ Многоязычный

x Медленный

✓ Несколько устройств

x Ограниченный

UI TESTING FOR ANDROID
espresso
appium

UI TESTING FOR ANDROID
espresso

Простой Espresso тест

```
@Test
fun espressoTest() {
    onView(withId(R.id.send_button)).perform(click())
    onView(withText("Success")).check(matches(isDisplayed()))
}
```

Простой Espresso тест

```
@Test
fun espressoTest() {
    onView(withId(R.id.send_button)).perform(click())
    onView(withText("Success")).check(matches(isDisplayed()))
}
```

Простой Espresso тест

```
@Test
fun espressoTest() {
    onView(withId(R.id.send_button)).perform(click())
    onView(withText("Success")).check(matches(isDisplayed()))
}
```

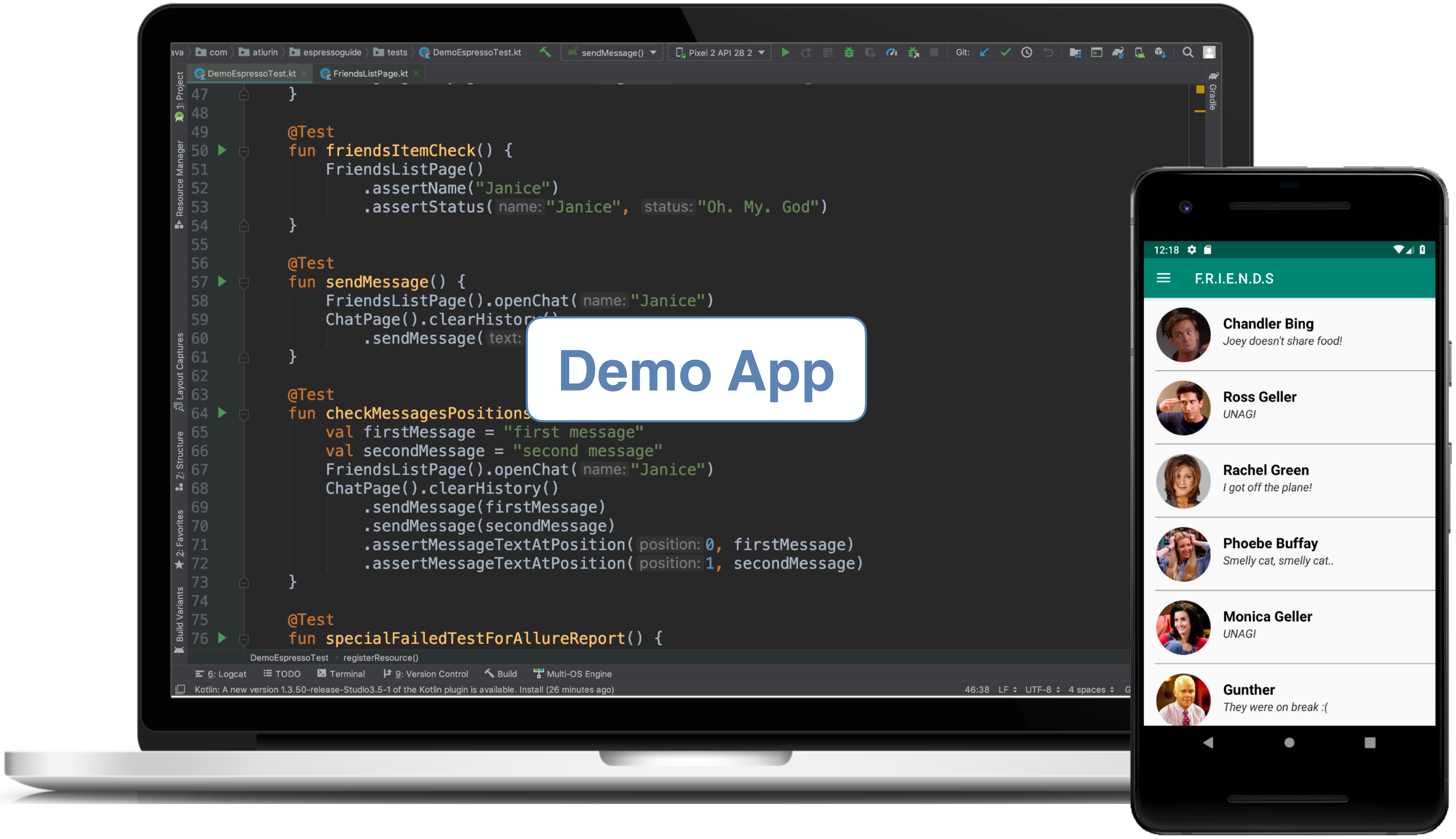
Простой Espresso тест

```
@Test
fun espressoTest() {
    onView(withId(R.id.send_button)).perform(click())
    onView(withText("Success")).check(matches(isDisplayed()))
}
```

Простой Espresso тест

```
@Test
fun espressoTest() {
    onView(withId(R.id.send_button)).perform(click())
    onView(withText("Success")).check(matches(isDisplayed()))
}
```

Demo App



```
    }

    @Test
    fun friendsItemCheck() {
        FriendsListPage()
            .assertName("Janice")
            .assertStatus( name: "Janice",  status: "Oh. My. God")
    }

    @Test
    fun sendMessage() {
        FriendsListPage().openChat( name: "Janice")
        ChatPage().clearHistory()
            .sendMessage( text:
    }

    @Test
    fun checkMessagesPositions
        val firstMessage = "first message"
        val secondMessage = "second message"
        FriendsListPage().openChat( name: "Janice")
        ChatPage().clearHistory()
            .sendMessage(firstMessage)
            .sendMessage(secondMessage)
            .assertMessageTextAtPosition( position: 0, firstMessage)
            .assertMessageTextAtPosition( position: 1, secondMessage)
    }

    @Test
    fun specialFailedTestForAllureReport() {
```

Kotlin: A new version 1.3.50-release-Studio3.5-1 of the Kotlin plugin is available. Install (26 minutes ago)

12:18

F.R.I.E.N.D.S

Chandler Bing
Joey doesn't share food!

Ross Geller
UNAGI

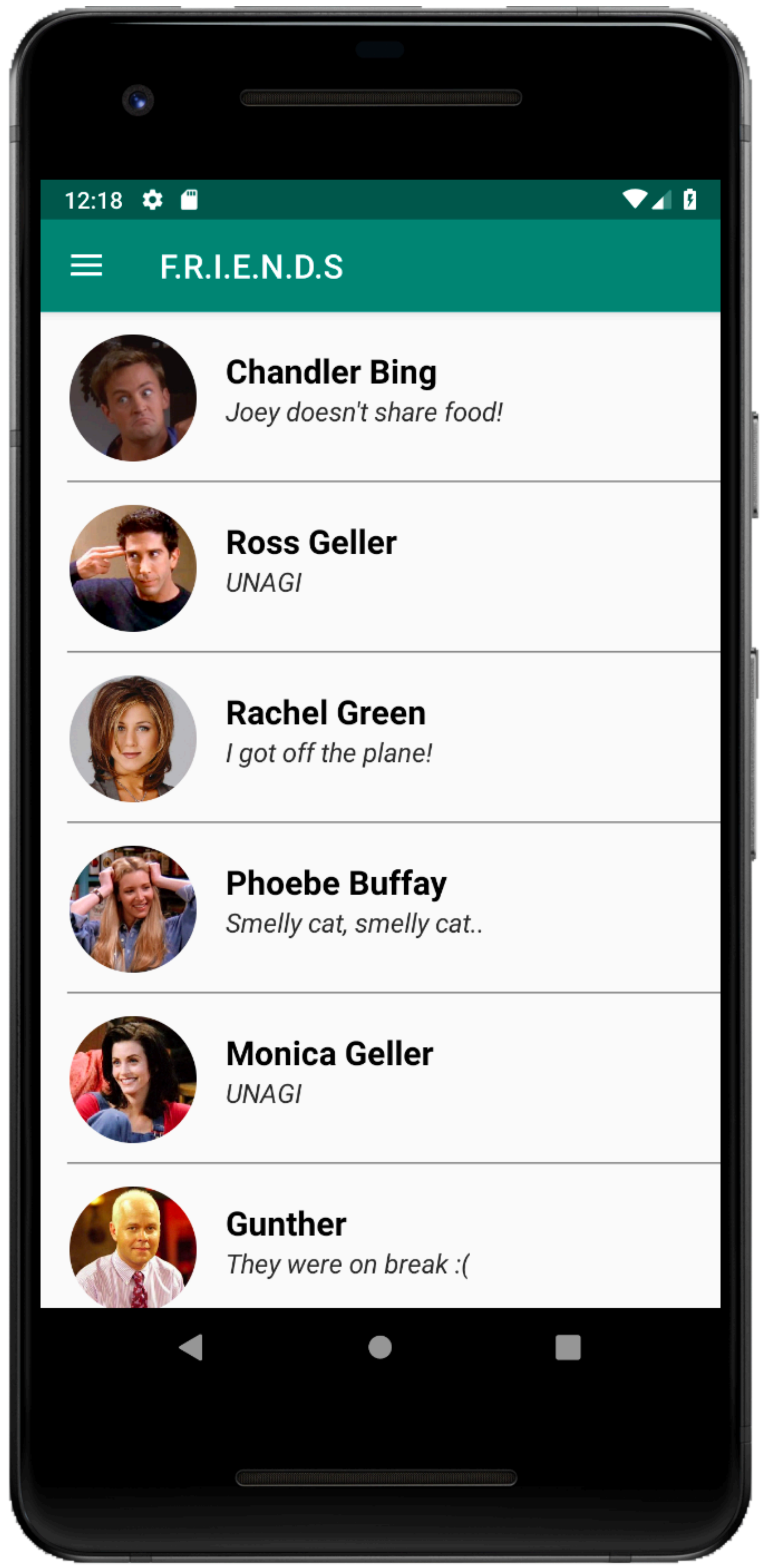
Rachel Green
I got off the plane!

Phoebe Buffay
Smelly cat, smelly cat..

Monica Geller
UNAGI

Gunther
They were on break :(

Сценарий demo теста


12:18

F.R.I.E.N.D.S

Chandler Bing
Joey doesn't share food!

Ross Geller
UNAGI

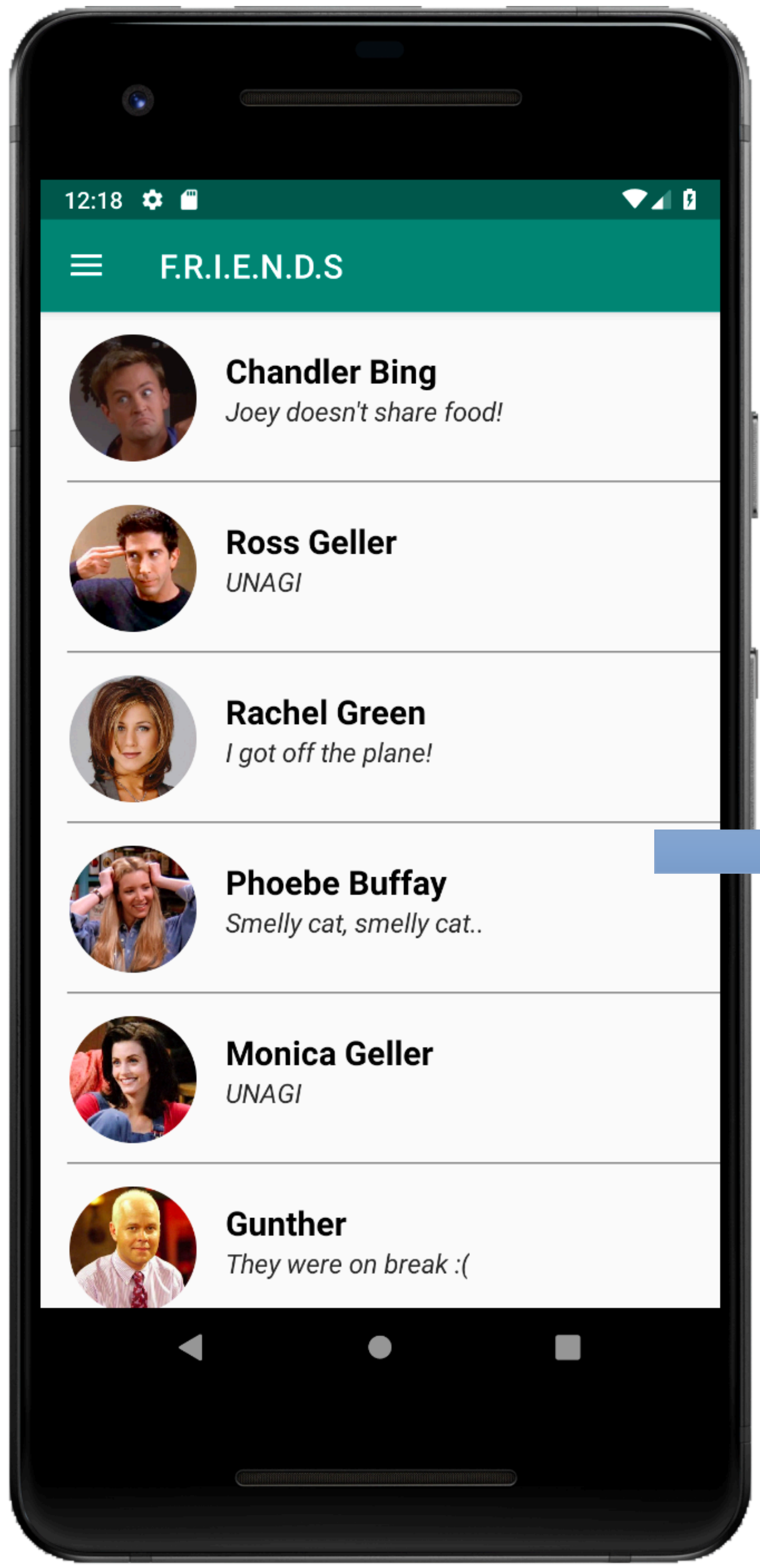
Rachel Green
I got off the plane!

Phoebe Buffay
Smelly cat, smelly cat..

Monica Geller
UNAGI

Gunther
They were on break :(

Сценарий demo теста


12:18

F.R.I.E.N.D.S

Chandler Bing
Joey doesn't share food!

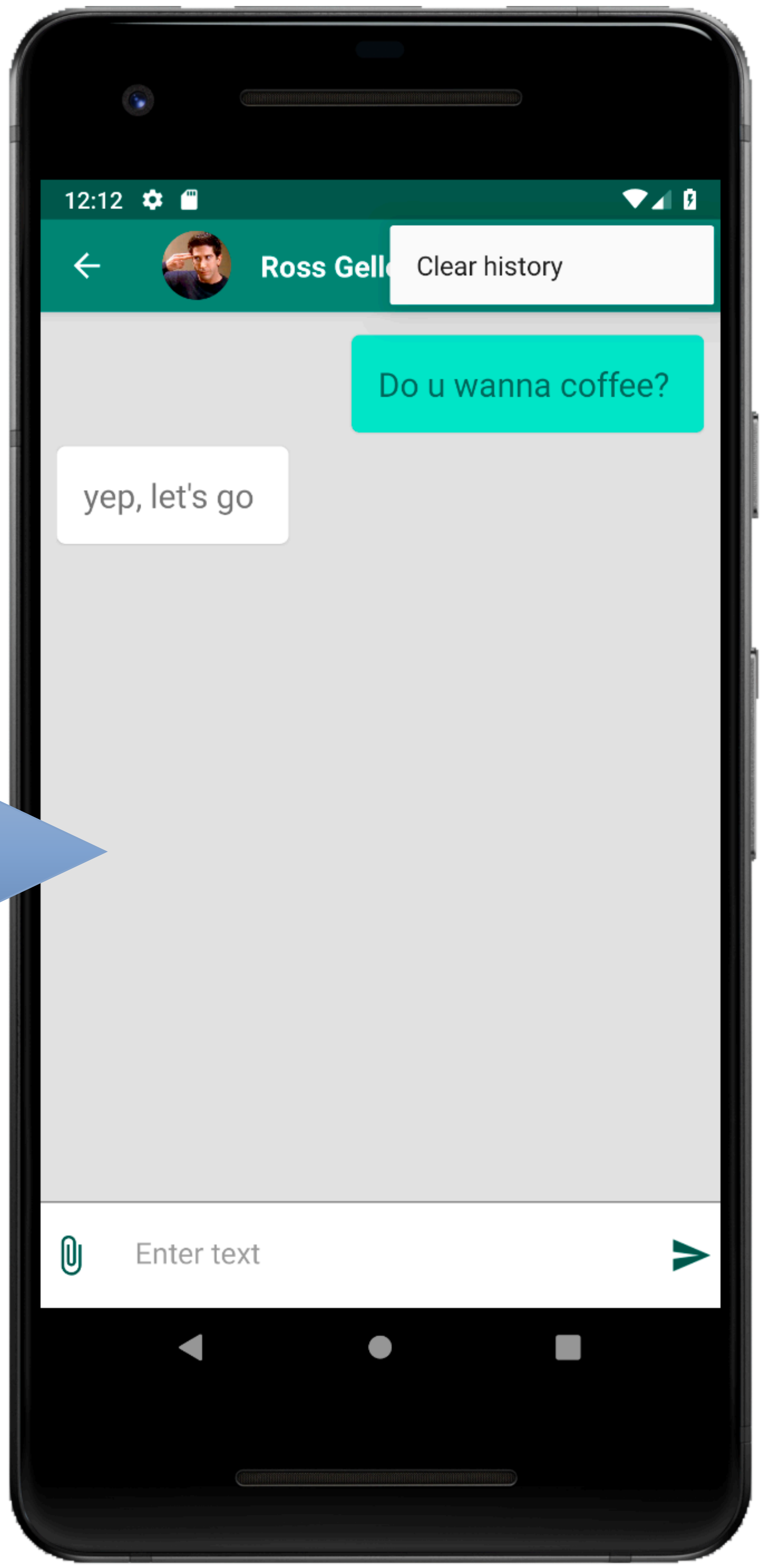
Ross Geller
UNAGI

Rachel Green
I got off the plane!

Phoebe Buffay
Smelly cat, smelly cat..

Monica Geller
UNAGI

Gunther
They were on break :(


12:12

Ross Gell

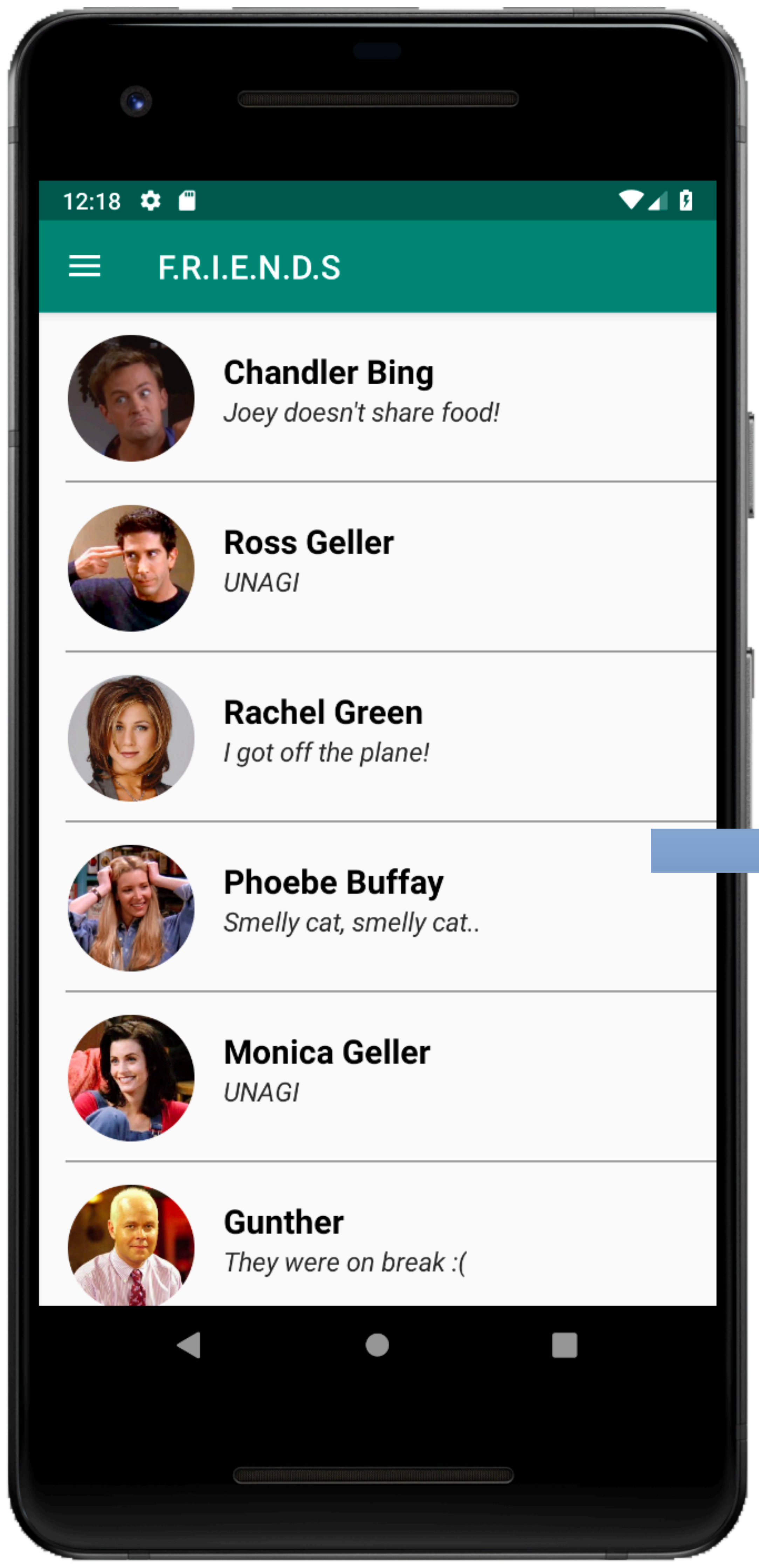
Clear history

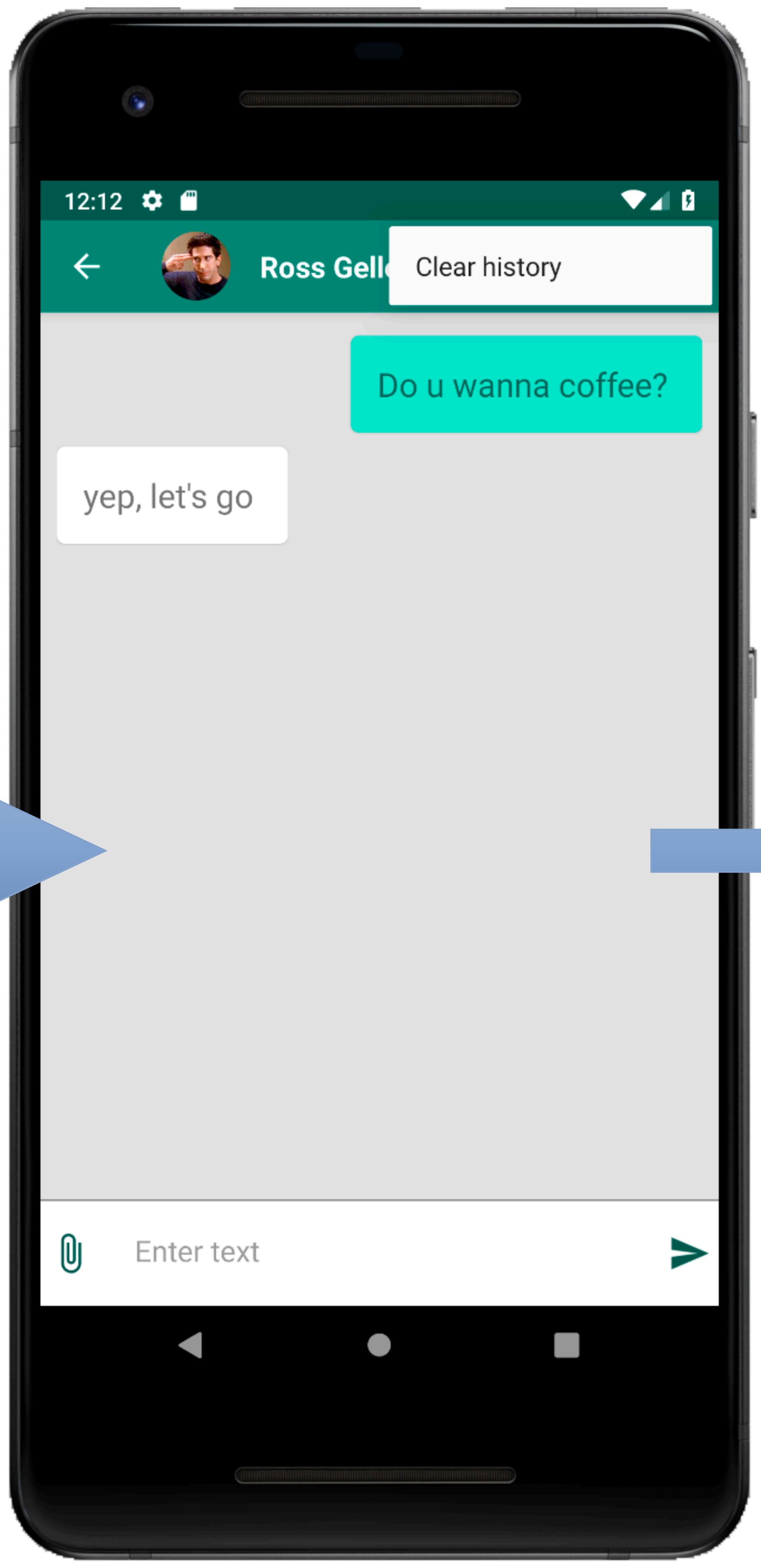
Do u wanna coffee?

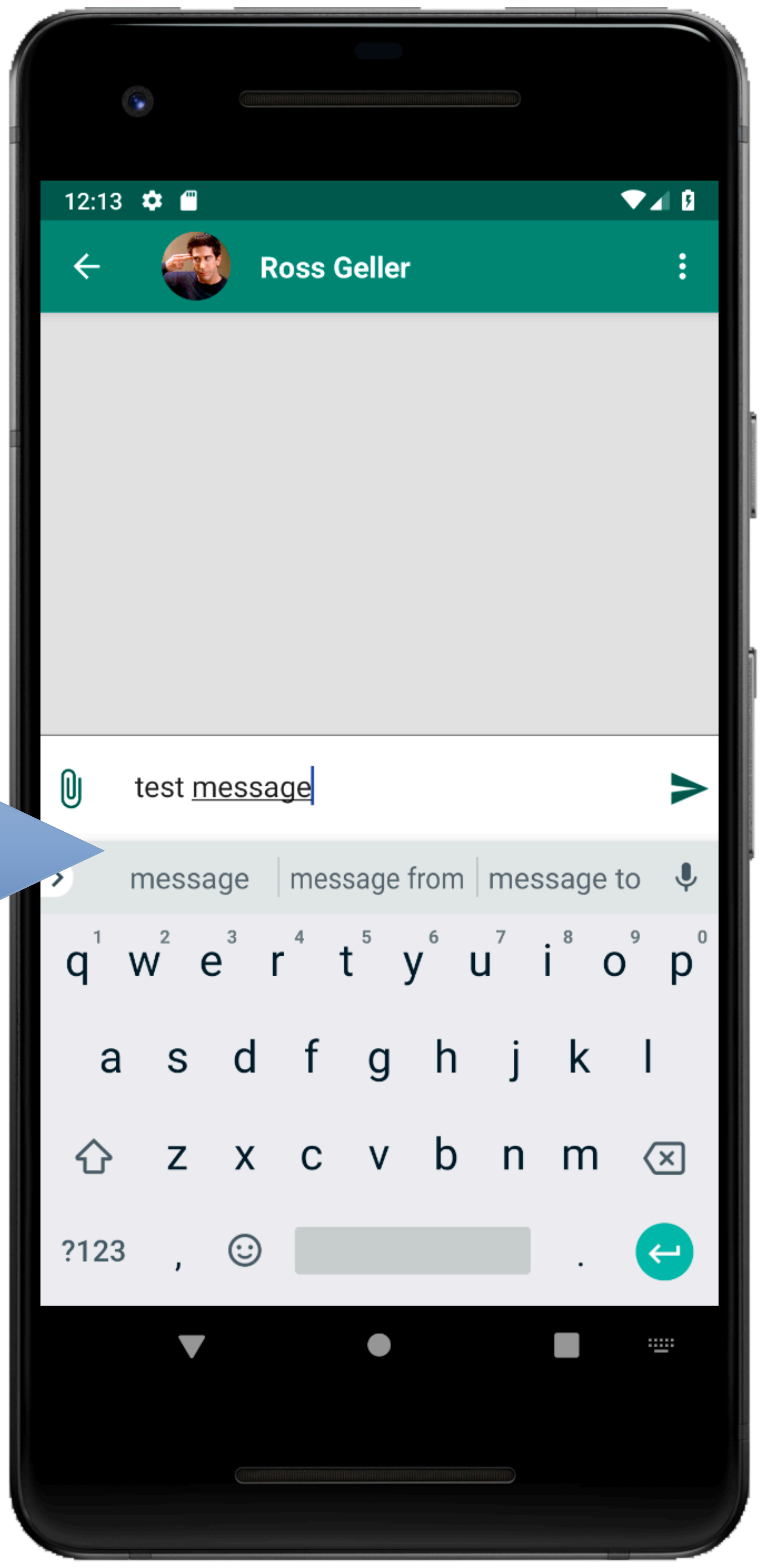
yep, let's go

Enter text

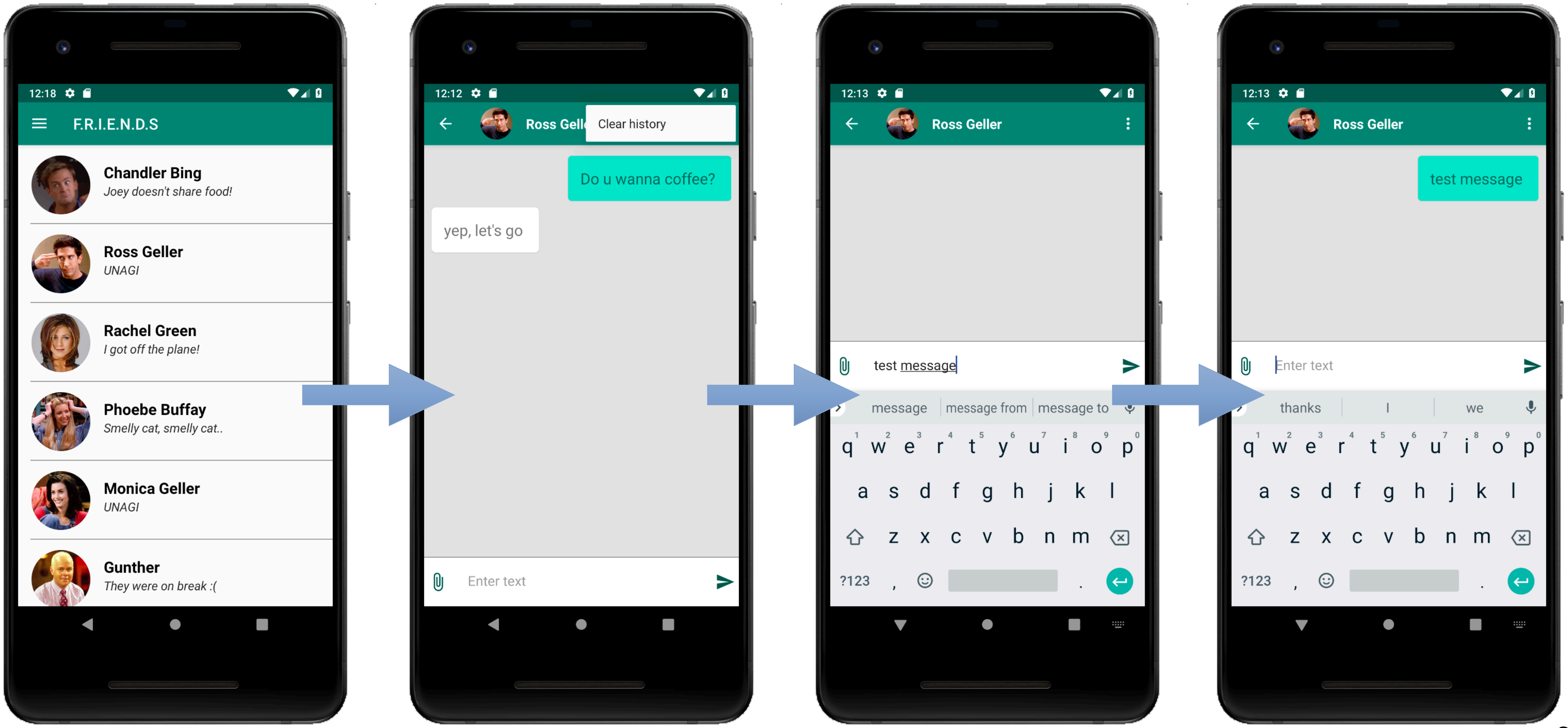
Сценарий demo теста


12:18
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
UNAGI
Gunther
They were on break :(


12:12
Ross Gell
Clear history
Do u wanna coffee?
yep, let's go
Enter text


12:13
Ross Geller
test message

Сценарий demo теста



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
UNAGI

Gunther
They were on break :(

Do u wanna coffee?

yep, let's go

test message

test message

Реальный Espresso тест

```
@Test
fun oldTest() {
    val messageText = "test message"
    val itemMatcher = hasDescendant(allOf(withId(R.id.tv_name),
        withText("Ross Geller")))
    onView(withId(R.id.recycler_friends))
        .perform(
            RecyclerViewActions
                .actionOnItem<RecyclerView.ViewHolder>(itemMatcher, click())
        )
    openActionBarOverflowOrOptionsMenu(getInstrumentation().context)
    onView(withText("Clear history")).perform(click())
    onView(withId(R.id.message_input_text)).perform(typeText(messageText))
    onView(withId(R.id.send_button)).perform(click())
    onView(withId(R.id.messages_list))
        .perform(
            RecyclerViewActions
                .scrollTo<RecyclerView.ViewHolder>(
                     hasDescendant(withText(messageText))))
    onView(allOf(withText(messageText), withId(R.id.message_text)))
        .check(matches(isDisplayed()))
}
```

```
@Test
fun oldTest() {
    val messageText = "test message"
    val itemMatcher = hasDescendant(allOf(withId(R.id.tv_name),
        withText("Ross Geller")))
    onView(withId(R.id.recycler_friends))
        .perform(
            RecyclerViewActions
                .actionOnItem<RecyclerView.ViewHolder>(itemMatcher, click())
        )
    openActionBarOverflowOrOptionsMenu(getInstrumentation().context)
    onView(withText("Clear history")).perform(click())
    onView(withId(R.id.message_input_text)).perform(typeText(messageText))
    onView(withId(R.id.send_button)).perform(click())
    onView(withId(R.id.messages_list))
        .perform(
            RecyclerViewActions
                .scrollTo<RecyclerView.ViewHolder>(
                     hasDescendant(withText(messageText))))
    onView(allOf(withText(messageText), withId(R.id.message_text)))
        .check(matches(isDisplayed()))
}
```

```kotlin
@Test
fun oldTest() {
    val messageText = "test message"
    val itemMatcher = hasDescendant(allOf(withId(R.id.tv_name),
        withText("Ross Geller")))
    onView(withId(R.id.recycler_friends))
        .perform(
            RecyclerViewActions
                .actionOnItem<RecyclerView.ViewHolder>(itemMatcher, click())
        )
    openActionBarOverflowOrOptionsMenu(getInstrumentation().context)
    onView(withText("Clear history")).perform(click())
    onView(withId(R.id.message_input_text)).perform(typeText(messageText))
    onView(withId(R.id.send_button)).perform(click())
    onView(withId(R.id.messages_list))
        .perform(
            RecyclerViewActions
                .scrollTo<RecyclerView.ViewHolder>(
                     hasDescendant(withText(messageText))))
    onView(allOf(withText(messageText), withId(R.id.message_text)))
        .check(matches(isDisplayed()))
}
```

- Тяжело читаемый тест

```
@Test
fun oldTest() {
    val messageText = "test message"
    val itemMatcher = hasDescendant(allOf(withId(R.id.tv_name),
        withText("Ross Geller")))
    onView(withId(R.id.recycler_friends))
        .perform(
            RecyclerViewActions
                .actionOnItem<RecyclerView.ViewHolder>(itemMatcher, click())
        )
    openActionBarOverflowOrOptionsMenu(getInstrumentation().context)
    onView(withText("Clear history")).perform(click())
    onView(withId(R.id.message_input_text)).perform(typeText(messageText))
    onView(withId(R.id.send_button)).perform(click())
    onView(withId(R.id.messages_list))
        .perform(
            RecyclerViewActions
                .scrollTo<RecyclerView.ViewHolder>(
                    hasDescendant(withText(messageText))))
    onView(allOf(withText(messageText), withId(R.id.message_text)))
        .check(matches(isDisplayed()))
}
```

- Тяжело читаемый тест
- Изменение локаторов элементов

```
@Test
fun oldTest() {
    val messageText = "test message"
    val itemMatcher = hasDescendant(allOf(withId(R.id.tv_name),
        withText("Ross Geller")))
    onView(withId(R.id.recycler_friends))
        .perform(
            RecyclerViewActions
                .actionOnItem<RecyclerView.ViewHolder>(itemMatcher, click())
        )
    openActionBarOverflowOrOptionsMenu(getInstrumentation().context)
    onView(withText("Clear history")).perform(click())
    onView(withId(R.id.message_input_text)).perform(typeText(messageText))
    onView(withId(R.id.send_button)).perform(click())
    onView(withId(R.id.messages_list))
        .perform(
            RecyclerViewActions
                .scrollTo<RecyclerView.ViewHolder>(
                    hasDescendant(withText(messageText))))
    onView(allOf(withText(messageText), withId(R.id.message_text)))
        .check(matches(isDisplayed()))
}
```

- Тяжело читаемый тест
- Изменение локаторов элементов
- Изменение пользовательского сценария

И еще немного проблем

И еще немного проблем

- Ожидание элементов

И еще немного проблем

- Ожидание элементов
- Tests sharding

И еще немного проблем

- Ожидание элементов
- Tests sharding
- Flaky тесты

И еще немного проблем

- Ожидание элементов
- Tests sharding
- Flaky тесты
- Отсутствие понятной отчетности

Problems

Support

Stability

Useless

Problems

Support

Stability

Useless

Problems.support

Problems.support

- Тяжело читаемый тест

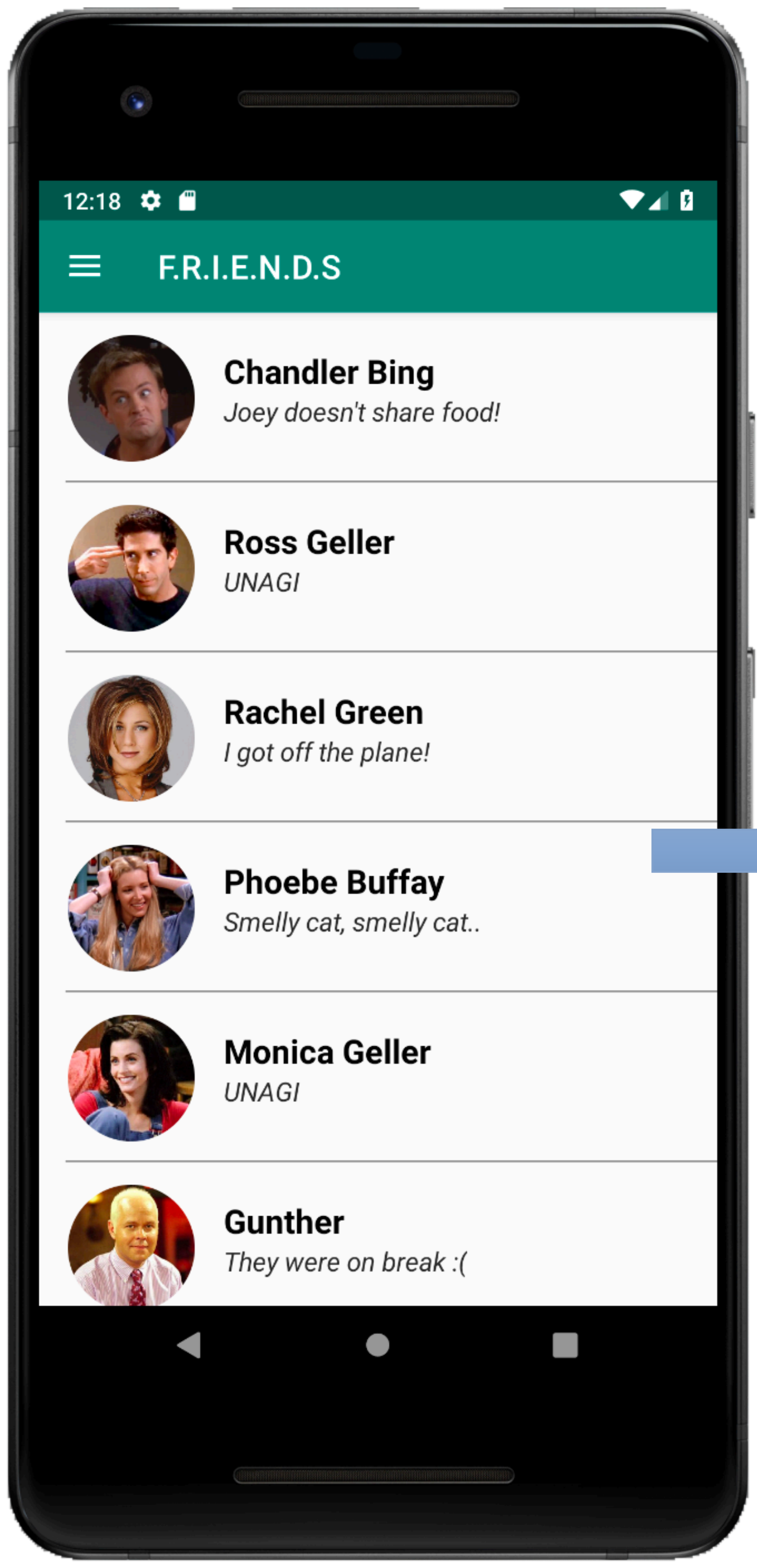
Problems.support

- Тяжело читаемый тест
- Изменение локаторов элементов

Problems.support

- Тяжело читаемый тест
- Изменение локаторов элементов
- Изменение пользовательского сценария

Сценарий demo теста


12:18

F.R.I.E.N.D.S

Chandler Bing
Joey doesn't share food!

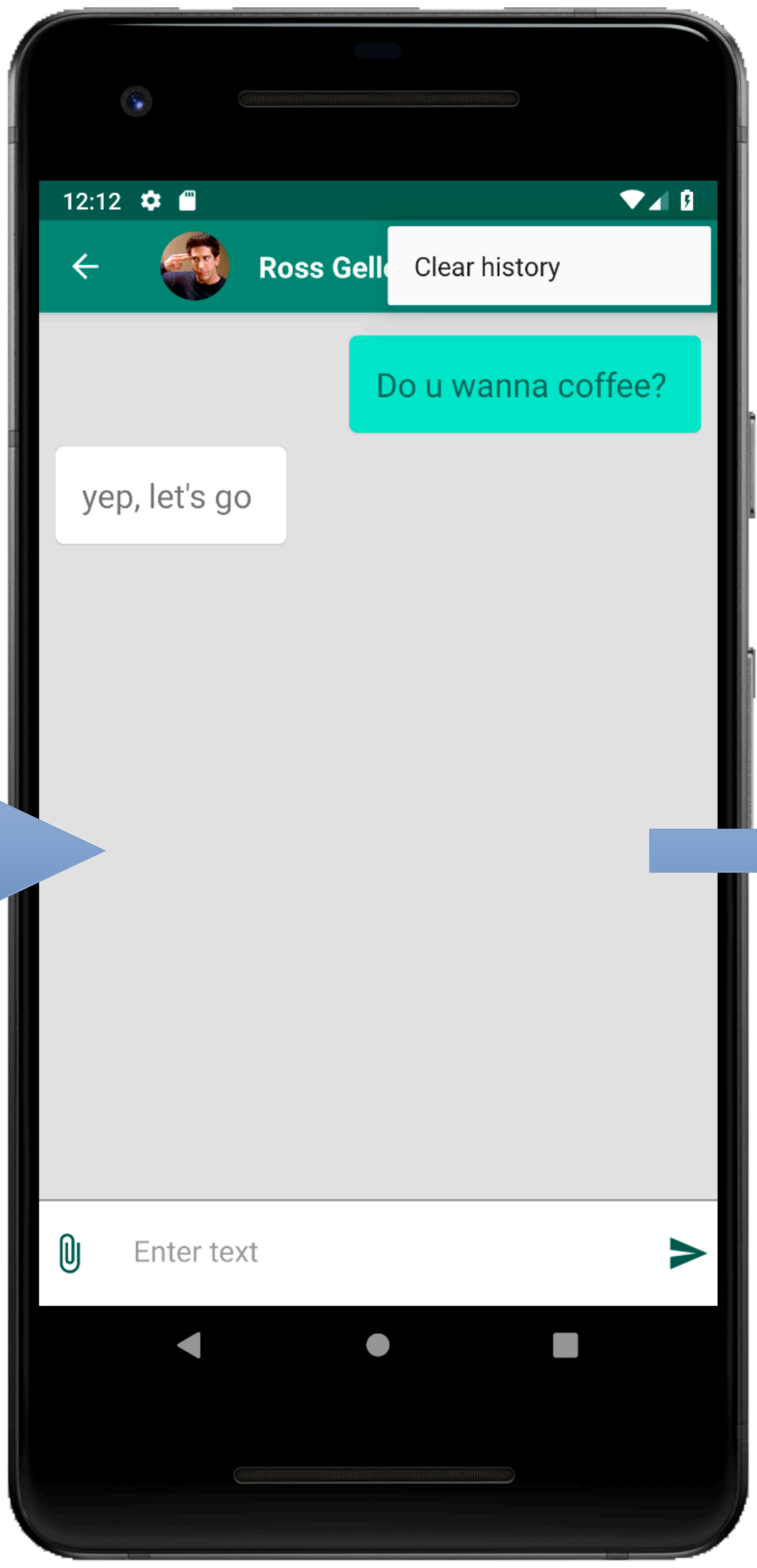
Ross Geller
UNAGI

Rachel Green
I got off the plane!

Phoebe Buffay
Smelly cat, smelly cat..

Monica Geller
UNAGI

Gunther
They were on break :(


12:12

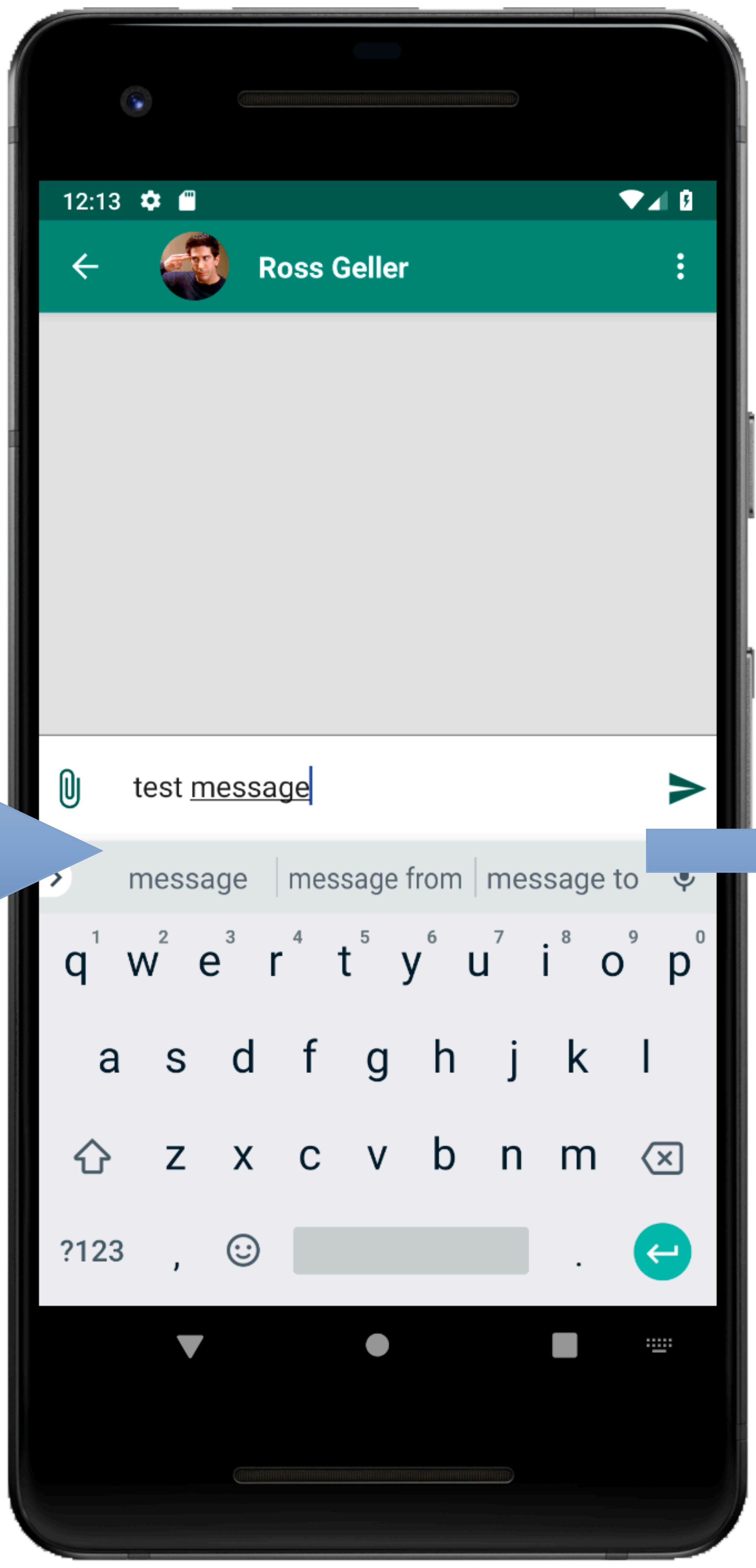
Ross Gell

Clear history

Do u wanna coffee?

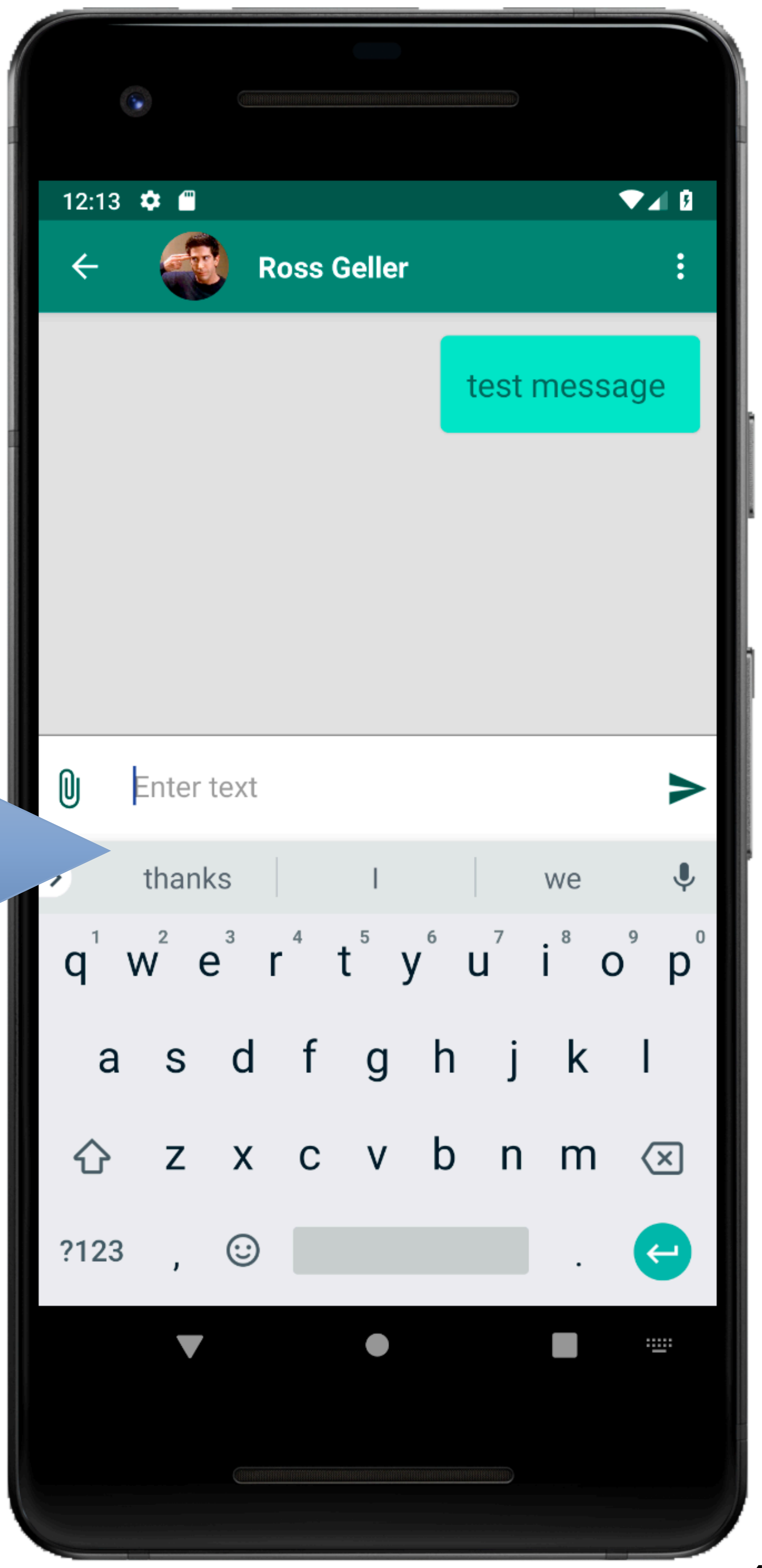
yep, let's go

Enter text


12:13

Ross Geller

test message


12:13

Ross Geller

test message

Enter text

Problems.support

TestCase:

```
@Test
fun goodTest(){



}
```

Problems.support

TestCase:

- Открыть чат

```
@Test
fun goodTest(){
    FriendsListPage().openChat("Ross Geller")


}
```

Problems.support

TestCase:

- Открыть чат
- Очистить историю

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

}
```

Problems.support

TestCase:

- Открыть чат
- Очистить историю
- Отправить сообщение

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

Problems.support

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

Problems.support

✓ PageObject

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

Problems.support

✓ PageObject + Steps

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

Problems.support

✓ PageObject + Steps

✓ DSL

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

Problems.support

- ✓ PageObject + Steps
- ✓ DSL ???

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

Варианты DSL

Варианты DSL

- Kakao

Варианты DSL

- Kakao
- Barista

Варианты DSL

- Kakao
- Barista
- Kaspresso

Варианты DSL

- Kakao
- Barista
- Kaspresso
- Свой DSL

Варианты DSL

- Kakao
- Barista
- ***Kaspresso***
- Свой DSL

Варианты DSL

- **Kakao**
- Barista
- Kaspresso
- **Свой DSL**

Kakao ?

Kakao provides different types depending on the type of view:

- `KView`
- `KEditText`
- `KTextView`
- `KButton`
- `KImageView`
- `KWebView`
- `KCheckbox`
- `KViewPager`
- `KSeekBar`
- `KSwitch`
- **and more**

https://github.com/agoda-com/Kakao

Kakao ?

Kakao provides different types depending on the type of view:

- `KView`
- `KEditText`
- `KTextView`
- `KButton`
- `KImageView`
- `KWebView`
- `KCheckbox`
- `KViewPager`
- `KSeekBar`
- `KSwitch`
- **and more**

https://github.com/agoda-com/Kakao

Kotlin extension function

```kotlin
fun String.addPrefix(prefix: String): String {
    return "$prefix-$this"
}
```

Kotlin extension function

```
fun String.addPrefix(prefix: String): String {
    return "$prefix-$this"
}

println("example".addPrefix("my"))
```

Kotlin extension function

```
fun String.addPrefix(prefix: String): String {
    return "$prefix-$this"
}

println("example".addPrefix("my"))

output: my-example
```

DSL on extension functions

```
@Test
fun oldTest() {
    onView(withId(R.id.send_button)).perform(click())
}
```

DSL on extension functions

```
@Test
fun oldTest() {
    onView(withId(R.id.send_button)).perform(click())
}
```

↓

```
Espresso documentation:

onView(viewMatcher: Matcher<View>) : ViewInteraction
```

DSL on extension functions

```
@Test
fun oldTest() {
    onView(withId(R.id.send_button)).perform(click())
}
```

↓

```
Espresso documentation:

onView(viewMatcher: Matcher<View>) : ViewInteraction
```

DSL on extension functions

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    onView(viewMatcher).perform(action) //на самом деле посложнее тут
}

fun Matcher<View>.click() = apply {
    actionOnView(this, CustomViewActions.click())
}

fun Matcher<View>.typeText(text: String) = apply {
    actionOnView(this, CustomViewActions.typeText(text))
}
```

DSL on extension functions

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    onView(viewMatcher).perform(action) //на самом деле посложнее тут
}

fun Matcher<View>.click() = apply {
    actionOnView(this, CustomViewActions.click())
}

fun Matcher<View>.typeText(text: String) = apply {
    actionOnView(this, CustomViewActions.typeText(text))
}
```

DSL on extension functions

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    onView(viewMatcher).perform(action) //на самом деле посложнее тут
}

fun Matcher<View>.click() = apply {
    actionOnView(this, CustomViewActions.click())
}

fun Matcher<View>.typeText(text: String) = apply {
    actionOnView(this, CustomViewActions.typeText(text))
}
```

DSL on extension functions

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    onView(viewMatcher).perform(action) //на самом деле посложнее тут
}

fun Matcher<View>.click() = apply {...}

fun Matcher<View>.typeText(text: String) = apply {...}

@Test
fun oldTest() {
    onView(withId(R.id.send_button)).perform(click())
}
```

DSL on extension functions

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    onView(viewMatcher).perform(action) //на самом деле посложнее тут
}

fun Matcher<View>.click() = apply {...}

fun Matcher<View>.typeText(text: String) = apply {...}

@Test
fun oldTest() {
    ~~onView(withId(R.id.send_button)).perform(click())~~
    withId(R.id.send_button).click()
}
```

DSL + PageObjects

```
class ChatPage : Page {
    private val sendMessageBtn = withId(R.id.send_button)
    private val inputMessageText = withId(R.id.message_input_text)
}
```

DSL + PageObjects + Steps

```
class ChatPage : Page {
    private val sendMessageBtn = withId(R.id.send_button)
    private val inputMessageText = withId(R.id.message_input_text)

    fun sendMessage(text: String) = apply {
        inputMessageText.typeText(text)
        sendMessageBtn.click()
    }
}
```

DSL + PageObjects + Steps

```
class ChatPage : Page {
    private val sendMessageBtn = withId(R.id.send_button)
    private val inputMessageText = withId(R.id.message_input_text)

    fun sendMessage(text: String) = apply {
        step("Send message with text '$text'") {
            inputMessageText.typeText(text)
            sendMessageBtn.click()
        }
    }
}
```

DSL + PageObjects + Steps

```
class ChatPage : Page {
    ...
    fun sendMessage(text: String) = apply {
        step("Send message with text '$text'") {
            inputMessageText.typeText(text)
            sendMessageBtn.click()
        }
    }
}

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

DSL + PageObjects + Steps

```
class ChatPage : Page {
    ...
    fun sendMessage(text: String) = apply {
        step("Send message with text '$text'") {
            inputMessageText.typeText(text)
            sendMessageBtn.click()
        }
    }
}

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

PageObjects trick

```
class ChatPage : Page {




}
```

PageObjects trick

```
class ChatPage : Page {
    constructor(action: ChatPage.() -> Unit){
        this.action()
    }


}
```

PageObjects trick

```
class ChatPage : Page {
    constructor(action: ChatPage.() -> Unit){
        this.action()
    }


}
```

PageObjects trick

```
class ChatPage : Page {
    constructor(action: ChatPage.() -> Unit){
        this.action()
    }


}
```

PageObjects trick

```
class ChatPage : Page {
    constructor(action: ChatPage.() -> Unit){
        this.action()
    }


}

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

PageObjects trick

```
class ChatPage : Page {
    constructor(action: ChatPage.() -> Unit){
        this.action()
    }


}

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

PageObjects trick

```
class ChatPage : Page {
    constructor(action: ChatPage.() -> Unit){
        this.action()
    }


}

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

PageObjects trick

```
class ChatPage : Page {
    constructor(action: ChatPage.() -> Unit){
        this.action()
    }
    constructor()
    ...
}

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

Problems.support

```
@Test
fun oldTest() {
    val messageText = "test message"
    val itemMatcher = hasDescendant(allOf(withId(R.id.tv_name),
        withText("Ross Geller")))
    onView(withId(R.id.recycler_friends))
        .perform(
            RecyclerViewActions
                .actionOnItem<RecyclerView.ViewHolder>(itemMatcher, click())
        )
    openActionBarOverflowOrOptionsMenu(getInstrumentation().context)
    onView(withText("Clear history")).perform(click())
    onView(withId(R.id.message_input_text)).perform(typeText(messageText))
    onView(withId(R.id.send_button)).perform(click())
    onView(withId(R.id.messages_list))
        .perform(
            RecyclerViewActions
                .scrollTo<RecyclerView.ViewHolder>(
                     hasDescendant(withText(messageText))))
    onView(allOf(withText(messageText), withId(R.id.message_text)))
        .check(matches(isDisplayed()))
}
```

Problems.support

```
@Test
fun oldTest() {
    val messageText = “test message"
    val itemMatcher = hasDescendant(allOf(withId(R.id.tv_name),
        withText("Ross Geller")))
    onView(withId(R.id.recycler_friends))
        .perform(
            RecyclerViewActions
                .actionOnItem<RecyclerView.ViewHolder>(itemMatcher, click())
        )
    openActionBarOverflowOrOptionsMenu(getInstrumentation().context)
    onView(withText("Clear history")).perform(click())
    onView(withId(R.id.message_input_text)).perform(typeText(messageText))
    onView(withId(R.id.send_button)).perform(click())
    onView(withId(R.id.messages_list))
        .perform(
            RecyclerViewActions
                .scrollTo<RecyclerView.ViewHolder>(
                     hasDescendant(withText(messageText))))
    onView(allOf(withText(messageText), withId(R.id.message_text)))
        .check(matches(isDisplayed()))
}
```

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

Problems.support

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

Problems.support

✓ ~~Тяжело читаемый тест~~

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

Problems.support

✓ ~~Тяжело читаемый тест~~

✓ ~~Изменение локаторов элементов~~

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

Problems.support

- ✓ ~~Тяжело читаемый тест~~
- ✓ ~~Изменение локаторов элементов~~
- ✓ ~~Изменение пользовательского сценария~~

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

Problems

Support

Stability

Useless

Problems.stability

- Ожидание элементов

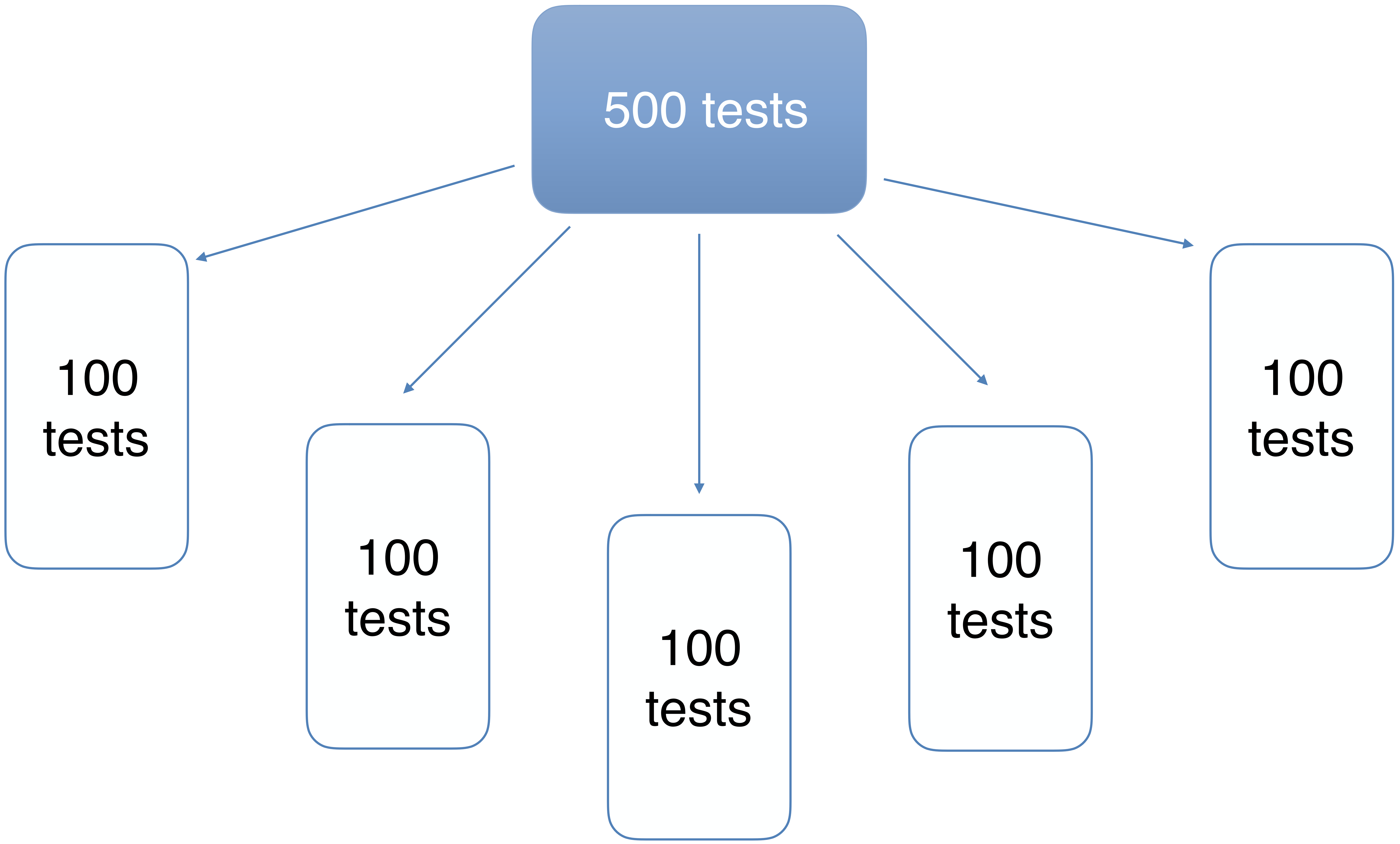
Problems.stability

- Ожидание элементов
- Tests sharding

Problems.stability.sharding

500 tests

Problems.stability.sharding



Problems.stability

- Ожидание элементов
- Tests sharding

Problems.stability

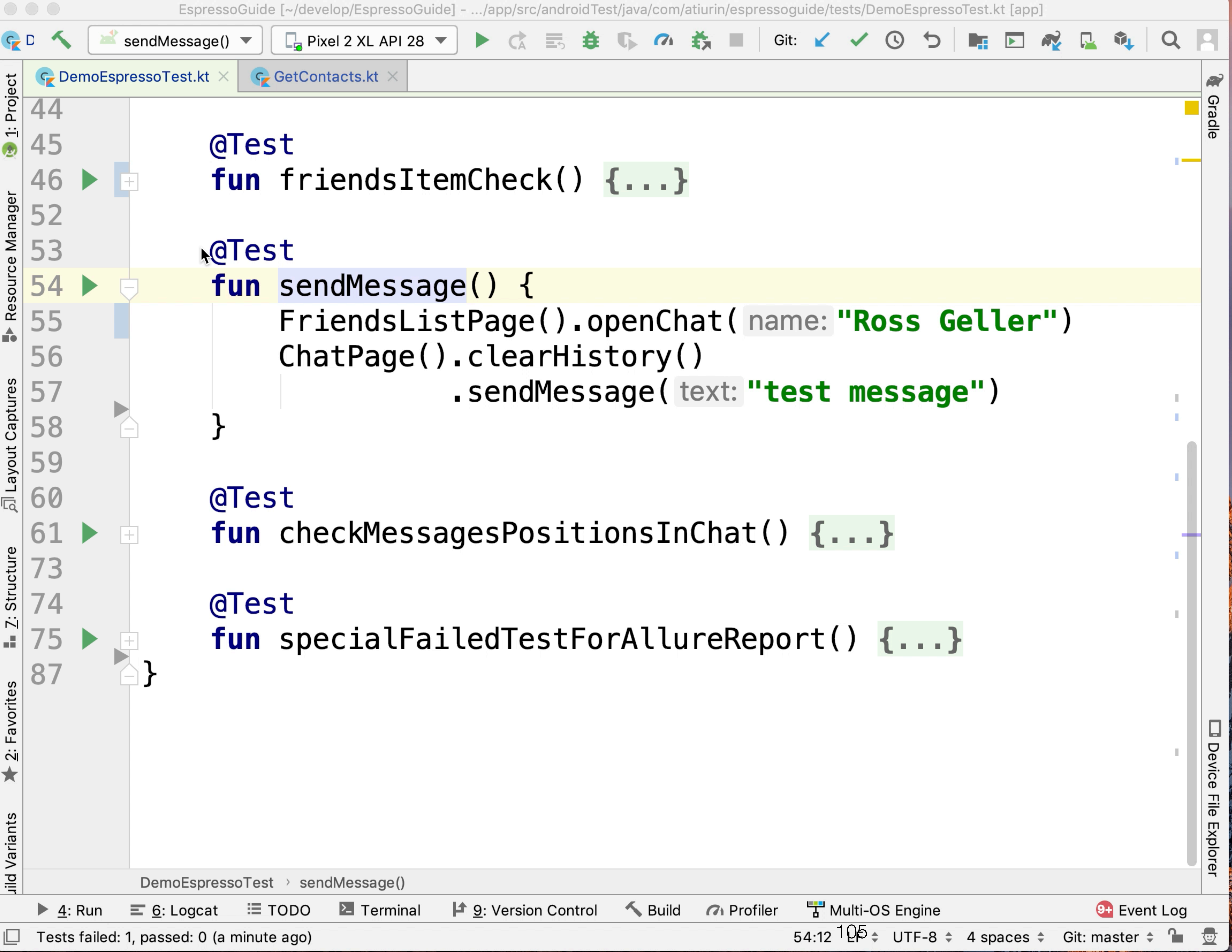
- Ожидание элементов
- Tests sharding
- Flaky тесты

Problems.stability

- Ожидание элементов
- Tests sharding
- Flaky тесты

Problems.stability.waits

Demo



```kotlin

    @Test
    fun friendsItemCheck() {...}

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

    @Test
    fun checkMessagesPositionsInChat() {...}

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

Tests failed: 1, passed: 0 (a minute ago)   54:12 LF UTF-8 4 spaces Git: master

Problems.stability.waits

✓ FailureHandler

Problems.stability.waits

- ✓ FailureHandler
- ✓ IdlingResources

Problems.stability.waits.failureHandler

Problems.stability.waits.failureHandler

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    onView(viewMatcher).perform(action) //на самом деле посложнее тут
}

fun Matcher<View>.click() = apply {
    actionOnView(this, CustomViewActions.click())
}

fun Matcher<View>.typeText (text: String) = apply {
    actionOnView(this, CustomViewActions.typeText(text))
}
```

Problems.stability.waits.failureHandler

```kotlin
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    onView(viewMatcher).perform(action) //на самом деле посложнее тут
}

fun Matcher<View>.click() = apply {
    actionOnView(this, CustomViewActions.click())
}

fun Matcher<View>.typeText (text: String) = apply {
    actionOnView(this, CustomViewActions.typeText(text))
}
```

Problems.stability.waits.failureHandler

```kotlin
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    var exception: Throwable? = null
    var operationResult: Boolean
    val startTime = SystemClock.elapsedRealtime()
    val endTime = startTime + ViewActionsConfig.ACTION_TIMEOUT
    do {
        operationResult = true
        onView(viewMatcher).withFailureHandler { error, viewMatcher ->
            if (error::class.java in ViewActionsConfig.allowedExceptions){
                operationResult = false
                exception = error
            } else throw error
        }.perform(action)
        if (!operationResult) Thread.sleep(50)
    } while (SystemClock.elapsedRealtime() < endTime && !operationResult)
    if (!operationResult && exception != null){
        throw exception as Throwable
    }
}
```

Problems.stability.waits.failureHandler

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    var exception: Throwable? = null
    var operationResult: Boolean
    ...
}
```

Problems.stability.waits.failureHandler

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    var exception: Throwable? = null
    var operationResult: Boolean
    val startTime = SystemClock.elapsedRealtime()
    val endTime = startTime + ViewActionsConfig.ACTION_TIMEOUT
    ...
}
```

Problems.stability.waits.failureHandler

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    var exception: Throwable? = null
    var operationResult: Boolean
    val startTime = SystemClock.elapsedRealtime()
    val endTime = startTime + ViewActionsConfig.ACTION_TIMEOUT
    do {
        operationResult = true
        ...// может сделать result = false
    } while (SystemClock.elapsedRealtime() < endTime && !operationResult)
    ...
}
```

Problems.stability.waits.failureHandler

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    var exception: Throwable? = null
    var operationResult: Boolean
    val startTime = SystemClock.elapsedRealtime()
    val endTime = startTime + ViewActionsConfig.ACTION_TIMEOUT
    do {
        operationResult = true
        onView(viewMatcher).withFailureHandler { error, viewMatcher ->
            ...
        }.perform(action)
        ...
    } while (SystemClock.elapsedRealtime() < endTime && !operationResult)
    ...
}
```

Problems.stability.waits.failureHandler

```kotlin
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    var exception: Throwable? = null
    var operationResult: Boolean
    val startTime = SystemClock.elapsedRealtime()
    val endTime = startTime + ViewActionsConfig.ACTION_TIMEOUT
    do {
        operationResult = true
        onView(viewMatcher).withFailureHandler { error, viewMatcher ->
            if (error::class.java in ViewActionsConfig.allowedExceptions){
                operationResult = false
                exception = error
            } else throw error
        }.perform(action)
        ...
    } while (SystemClock.elapsedRealtime() < endTime && !operationResult)
    ...
}
```

Problems.stability.waits.failureHandler

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    var exception: Throwable? = null
    var operationResult: Boolean
    val startTime = SystemClock.elapsedRealtime()
    val endTime = startTime + ViewActionsConfig.ACTION_TIMEOUT
    do {
        operationResult = true
        onView(viewMatcher).withFailureHandler { error, viewMatcher ->
            if (error::class.java in ViewActionsConfig.allowedExceptions){
                operationResult = false
                exception = error
            } else throw error
        }.perform(action)
        ...
    } while (SystemClock.elapsedRealtime() < endTime && !operationResult)
    ...
}
```

Problems.stability.waits.failureHandler

```
var allowedExceptions = mutableListOf<Class<out Throwable>>(
    PerformException::class.java,
    NoMatchingViewException::class.java
)
```

Problems.stability.waits.failureHandler

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    var exception: Throwable? = null
    var operationResult: Boolean
    val startTime = SystemClock.elapsedRealtime()
    val endTime = startTime + ViewActionsConfig.ACTION_TIMEOUT
    do {
        operationResult = true
        onView(viewMatcher).withFailureHandler { error, viewMatcher ->
            if (error::class.java in ViewActionsConfig.allowedExceptions){
                operationResult = false
                exception = error
            } else throw error
        }.perform(action)
        ...
    } while (SystemClock.elapsedRealtime() < endTime && !operationResult)
    ...
}
```

Problems.stability.waits.failureHandler

```
private fun actionOnView(viewMatcher: Matcher<View>, action: ViewAction){
    var exception: Throwable? = null
    var operationResult: Boolean
    ...
    do {
        operationResult = true
        onView(viewMatcher).withFailureHandler { error, viewMatcher ->
            if (error::class.java in ViewActionsConfig.allowedExceptions){
                operationResult = false
                exception = error
            } else throw error
        }.perform(action)
        if (!operationResult) Thread.sleep(50)
    } while (SystemClock.elapsedRealtime() < endTime && !operationResult)
    if (!operationResult && exception != null){
        throw exception as Throwable
    }
}
```

Demo

EspressoGuide [~/develop/EspressoGuide] - .../app/src/androidTest/java/com/atiurin/espressoguide/tests/DemoEspressoTest.kt [app]
sendMessage()
Pixel 2 XL API 28
Git:
DemoEspressoTest.kt
GetContacts.kt
1: Project
Resource Manager
Layout Captures
Z: Structure
2: Favorites
Build Variants
Gradle
Device File Explorer
DemoEspressoTest › val mActivityRule
4: Run
6: Logcat
TODO
Terminal
9: Version Control
Build
Profiler
Multi-OS Engine
Event Log
28:80 LF UTF-8 4 spaces Git: master
12:17
Thursday, Nov 28
VK

```kotlin
    @Before
    fun backgroundLogin() {...}

    @Test
    fun friendsItemCheck() {...}

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

    @Test
    fun checkMessagesPositionsInChat() {...}

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

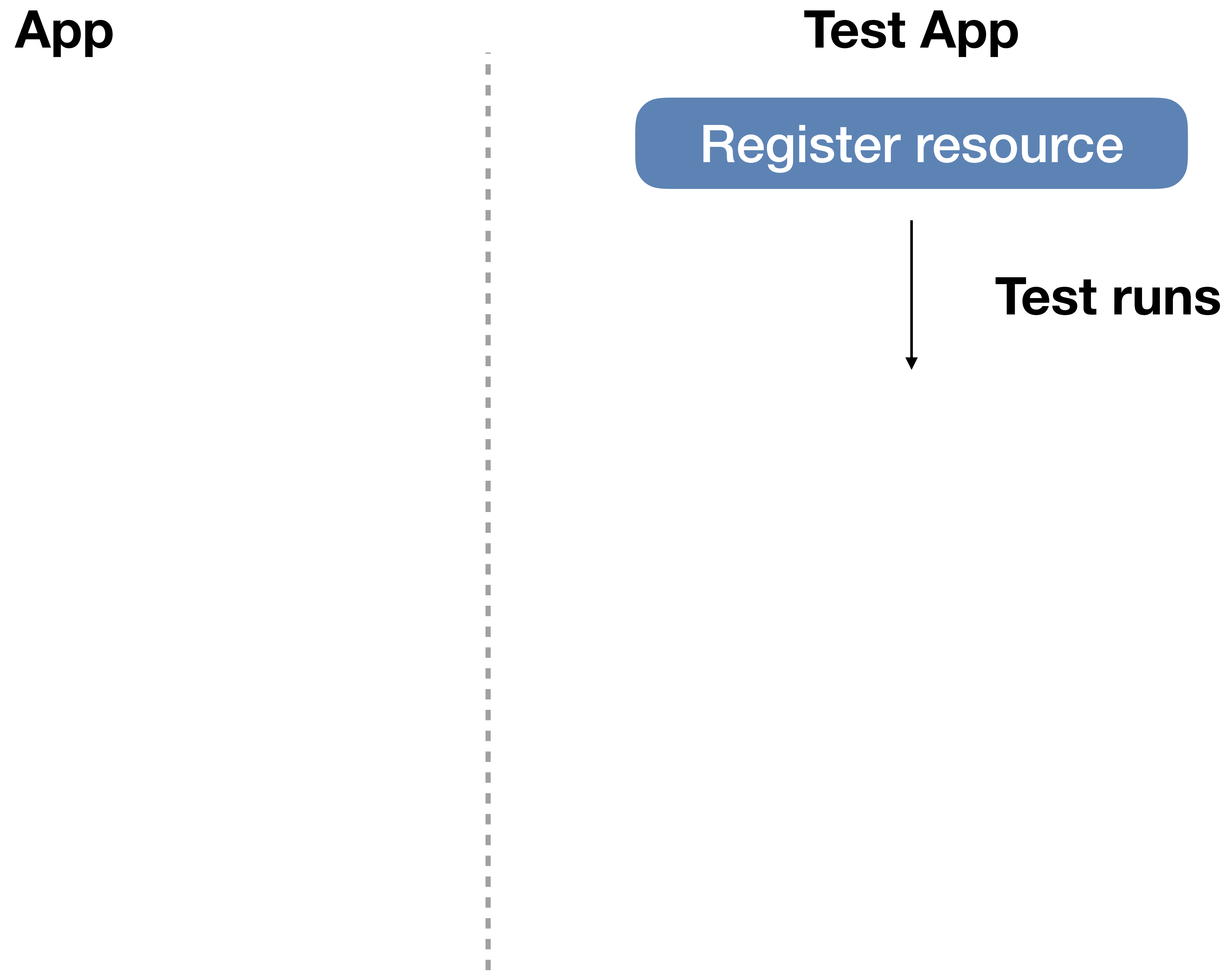
Tests failed: 1, passed: 0 (7 minutes ago)

Problems.stability.waits.IdlingResources

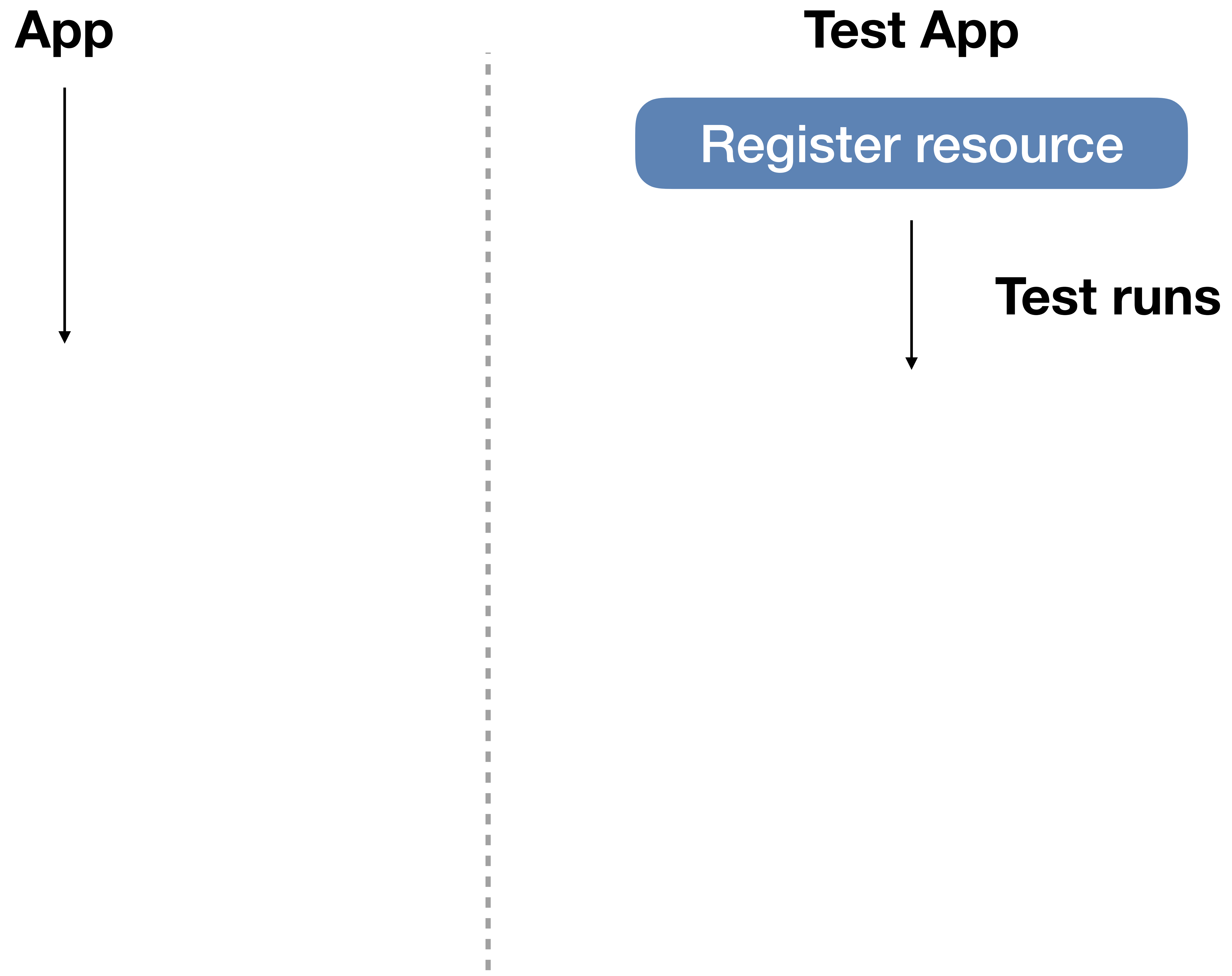
Problems.stability.waits.IdlingResources

- Синхронизация выполнения кода приложения и тестов

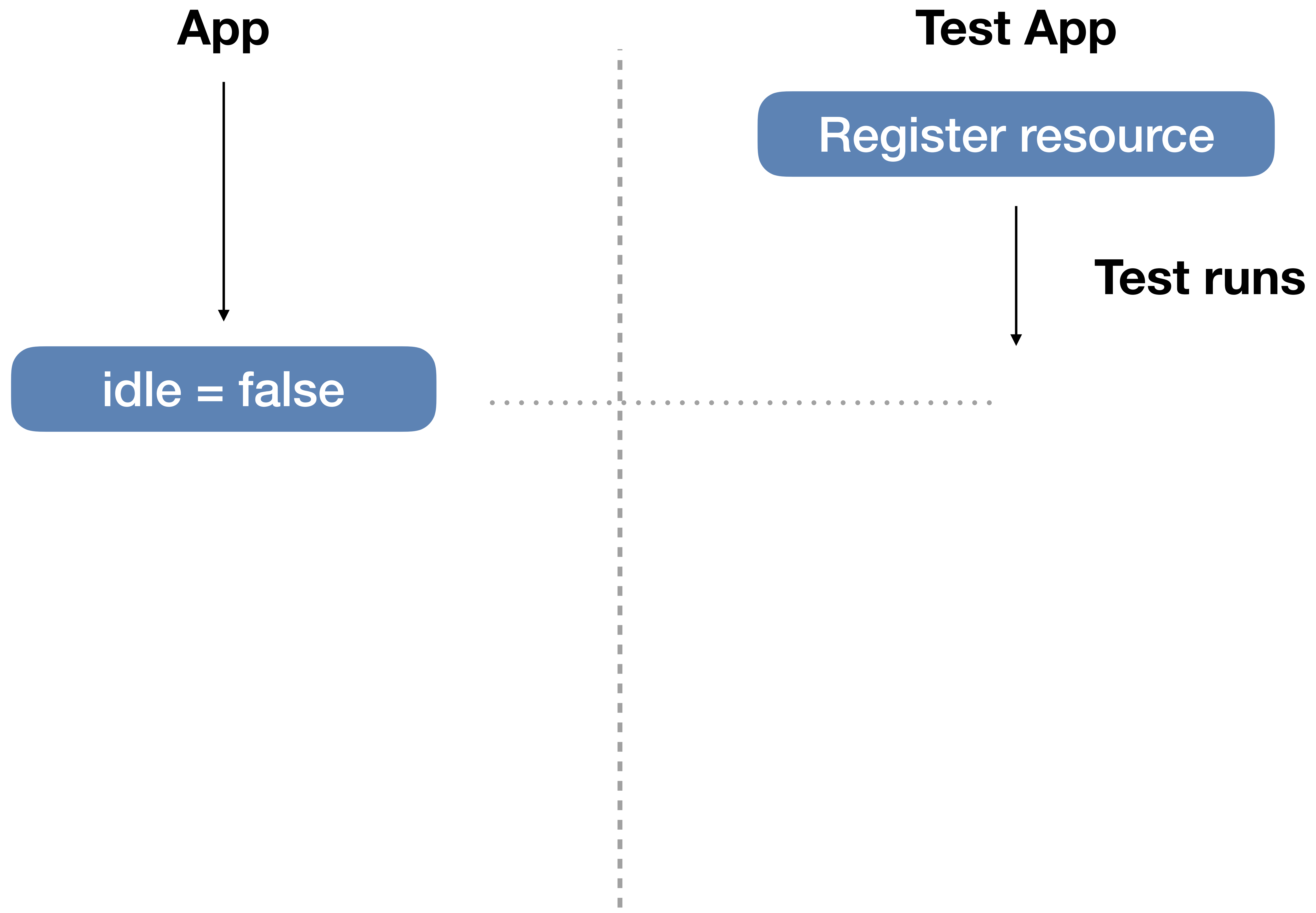
Problems.stability.waits.IdlingResources



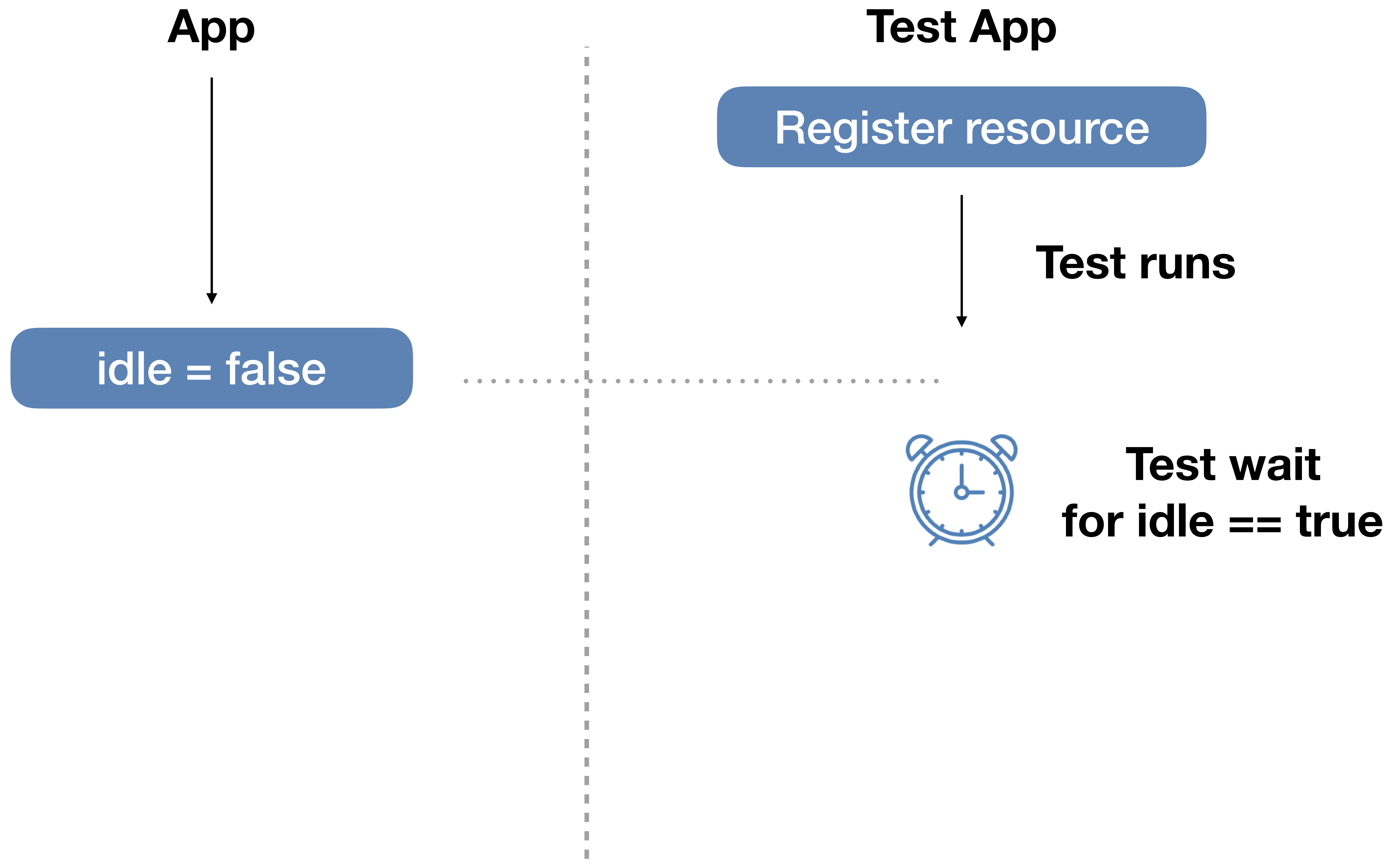
Problems.stability.waits.IdlingResources



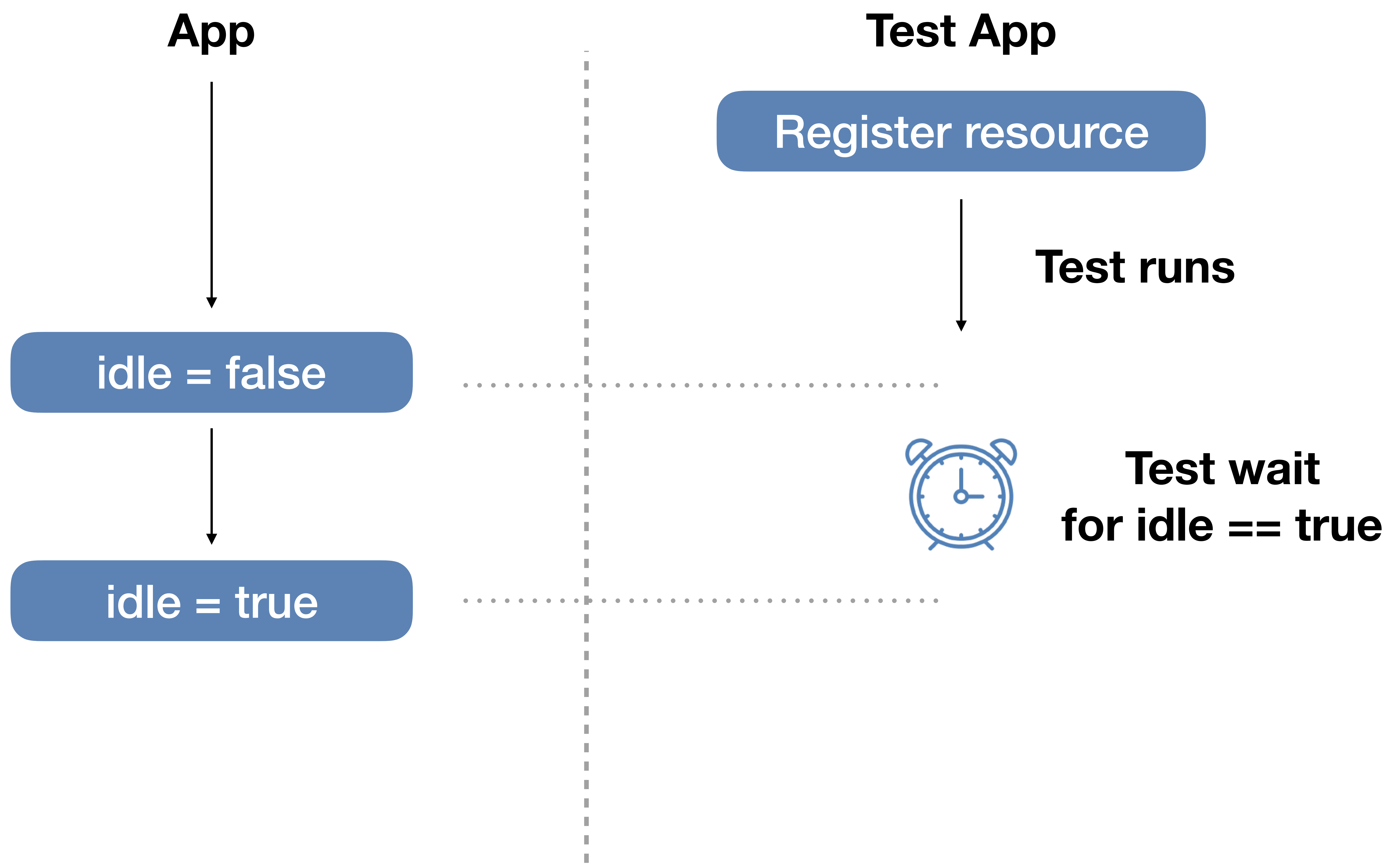
Problems.stability.waits.IdlingResources



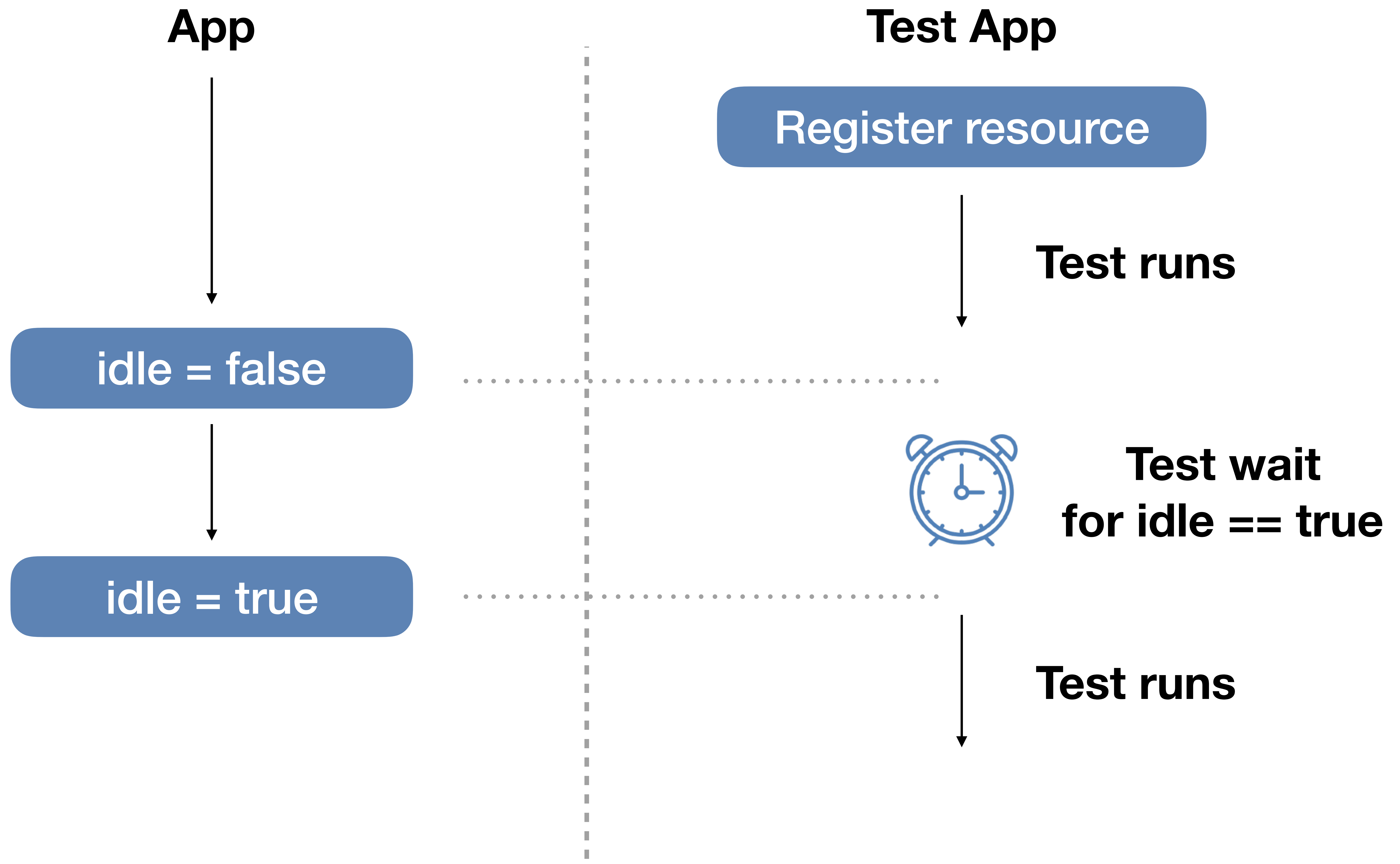
Problems.stability.waits.IdlingResources



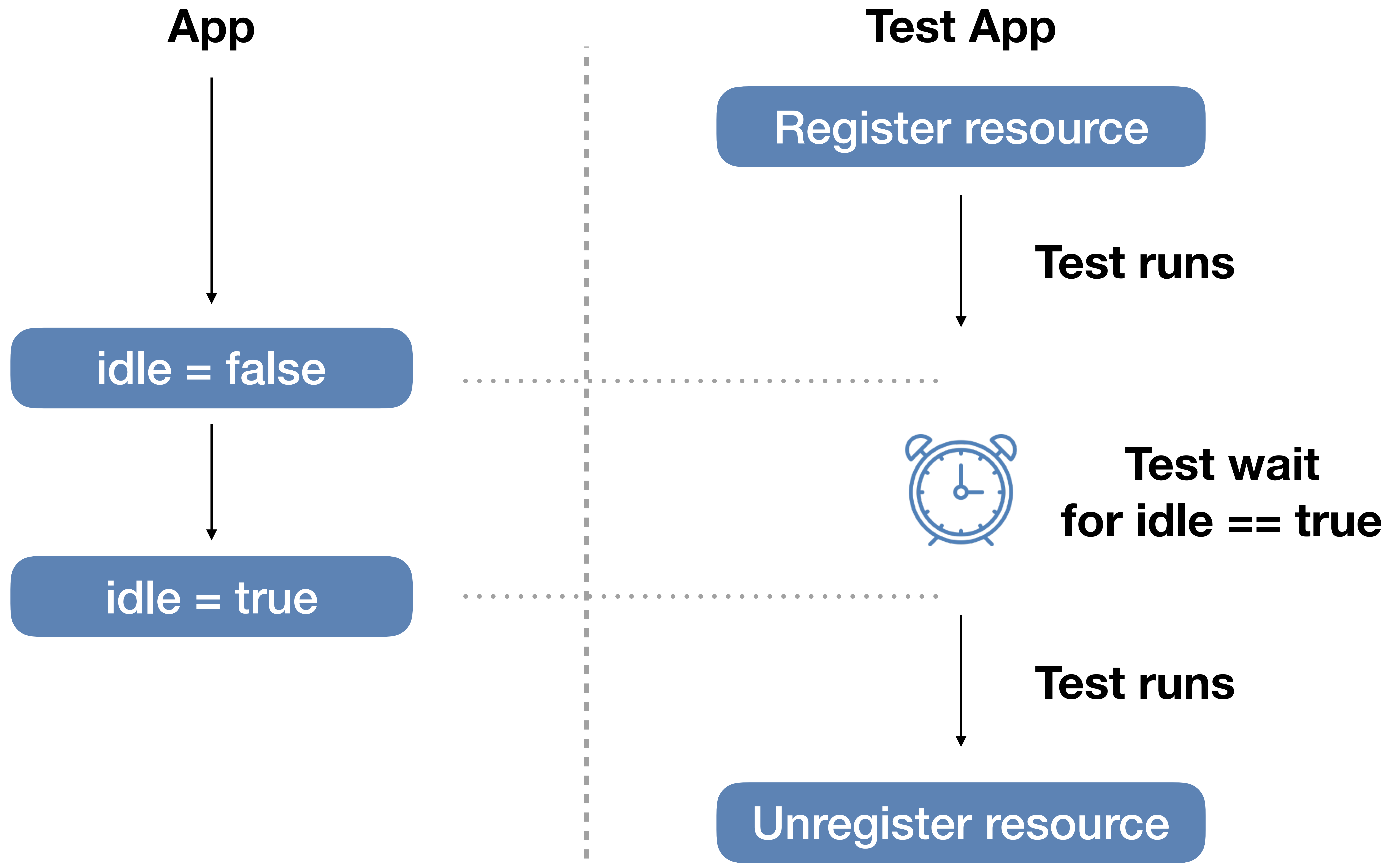
Problems.stability.waits.IdlingResources



Problems.stability.waits.IdlingResources



Problems.stability.waits.IdlingResources



IdlingResource Google samples

https://github.com/googlecodelabs/android-testing

IdlingResource Google samples

```
override suspend fun getTasks(forceUpdate: Boolean): Result<List<Task>> {
    EspressoIdlingResource.increment() // Set app as busy.
    return withContext(ioDispatcher) {
        EspressoIdlingResource.decrement() // Set app as idle.
        ...
    }
}
```

https://github.com/googlecodelabs/android-testing

IdlingResource Google samples

```
override suspend fun getTasks(forceUpdate: Boolean): Result<List<Task>> {
    EspressoIdlingResource.increment() // Set app as busy.
    return withContext(ioDispatcher) {
        EspressoIdlingResource.decrement() // Set app as idle.
        ...
    }
}
```

https://github.com/googlecodelabs/android-testing

IdlingResource Google samples

```
override suspend fun getTasks(forceUpdate: Boolean): Result<List<Task>> {
    EspressoIdlingResource.increment() // Set app as busy.
    return withContext(ioDispatcher) {
        EspressoIdlingResource.decrement() // Set app as idle.
        ...
    }
}
```

https://github.com/googlecodelabs/android-testing

IdlingResource Google samples

```
override suspend fun getTasks(forceUpdate: Boolean): Result<List<Task>> {
    EspressoIdlingResource.increment() // Set app as busy.
    return withContext(ioDispatcher) {
        EspressoIdlingResource.decrement() // Set app as idle.
        ...
    }
}
object EspressoIdlingResource {
    private const val RESOURCE = "GLOBAL"
    @JvmField val countingIdlingResource = SimpleCountingIdlingResource(RESOURCE)
    fun increment() {
        countingIdlingResource.increment()
    }
    fun decrement() {
        if (!countingIdlingResource.isIdleNow) {
            countingIdlingResource.decrement()
        }
    }
}
```

https://github.com/googlecodelabs/android-testing

IdlingResource Google samples

```
override suspend fun getTasks(forceUpdate: Boolean): Result<List<Task>> {
    EspressoIdlingResource.increment() // Set app as busy.
    return withContext(ioDispatcher) {
        EspressoIdlingResource.decrement() // Set app as idle.
        ...
    }
}
object EspressoIdlingResource {
    private const val RESOURCE = "GLOBAL"
    @JvmField val countingIdlingResource = SimpleCountingIdlingResource(RESOURCE)
    fun increment() {
        countingIdlingResource.increment()
    }
    fun decrement() {
        if (!countingIdlingResource.isIdleNow) {
            countingIdlingResource.decrement()
        }
    }
}
```

https://github.com/googlecodelabs/android-testing

IdlingResource Google samples

- github.com/googlecodelabs/android-testing
- github.com/android/testing-samples

IdlingResource Google samples

- github.com/googlecodelabs/android-testing
- github.com/android/testing-samples

Problems.stability.waits.IdlingResources

AppCode

Problems.stability.waits.IdlingResources

AppCode

```
idling { contactsIdling.setIdleState(false) }
```

Problems.stability.waits.IdlingResources

AppCode

```
idling { contactsIdling.setIdleState(false) }
// load contacts
```

Problems.stability.waits.IdlingResources

AppCode

```
idling { contactsIdling.setIdleState(false) }
// load contacts
idling { contactsIdling.setIdleState(true) }
```

Problems.stability.waits.IdlingResources

AppCode

```
idling { contactsIdling.setIdleState(false) }
// load contacts
idling { contactsIdling.setIdleState(true) }

idling { messagesIdling.setIdleState(false) }
// load messages
idling { messagesIdling.setIdleState(true) }
```

Problems.stability.waits.IdlingResources

AppCode

```
class ContactsIdling : AbstractIdlingResource()
class MessagesIdling : AbstractIdlingResource()
```

Problems.stability.waits.IdlingResources

```
AppCode

class ContactsIdling : AbstractIdlingResource()
class MessagesIdling : AbstractIdlingResource()

interface IdlingScope {
    val contactsIdling: ContactsIdling
    val messagesIdling: MessagesIdling
}
```

Problems.stability.waits.IdlingResources

AppCode

```
class ContactsIdling : AbstractIdlingResource()
class MessagesIdling : AbstractIdlingResource()

interface IdlingScope {
    val contactsIdling: ContactsIdling
    val messagesIdling: MessagesIdling
}

val idlingContainer = AtomicReference<IdlingScope>() //value is null by default
```

Problems.stability.waits.IdlingResources

AppCode

```
class ContactsIdling : AbstractIdlingResource()
class MessagesIdling : AbstractIdlingResource()

interface IdlingScope {
    val contactsIdling: ContactsIdling
    val messagesIdling: MessagesIdling
}

val idlingContainer = AtomicReference<IdlingScope>() //value is null by default

fun idling(action: IdlingScope.() -> Unit){
    if (!isReleaseBuild()){
        idlingContainer.get()?.action()
    }
}
```

Problems.stability.waits.IdlingResources

AppCode

```
val idlingContainer = AtomicReference<IdlingScope>() //value is null by default
```

Problems.stability.waits.IdlingResources

AppCode

```
val idlingContainer = AtomicReference<IdlingScope>() //value is null by default
```

TestCode

```
@Before
fun registerIdling() {
    idlingContainer.set(getDefaultIdlingScope())
    ...
}
```

Problems.stability.waits.IdlingResources

AppCode

```
val idlingContainer = AtomicReference<IdlingScope>() //value is null by default
```

TestCode

```
@Before
fun registerIdling() {
    idlingContainer.set(getDefaultIdlingScope())
    ...
}
fun getDefaultIdlingScope(): IdlingScope {
    return object : IdlingScope {
        override val messagesIdling: MessagesIdling = MessagesIdling()
        override var contactsIdling: ContactsIdling = ContactsIdling()
    }
}
```

Problems.stability.waits.IdlingResources

AppCode

```
val idlingContainer = AtomicReference<IdlingScope>() //value is null by default
```

TestCode

```
@Before
fun registerIdling() {
    idlingContainer.set(getDefaultIdlingScope())
    ...
}
fun getDefaultIdlingScope(): IdlingScope {
    return object : IdlingScope {
        override val messagesIdling: MessagesIdling = MessagesIdling()
        override var contactsIdling: ContactsIdling = ContactsIdling()
    }
}
```

Problems.stability.waits.IdlingResources

TestCode

```
@Before
fun registerIdling() {
    idlingContainer.set(getDefaultIdlingScope())
    ...
}
```

Problems.stability.waits.IdlingResources

TestCode

```
@Before
fun registerIdling() {
    idlingContainer.set(getDefaultIdlingScope())
    idling {
      IdlingRegistry.getInstance().register(contactsIdling)
    }
}
```

Problems.stability.waits.IdlingResources

```
TestCode

@Before
fun registerIdling() {
    idlingContainer.set(getDefaultIdlingScope())
    idling {
      IdlingRegistry.getInstance().register(contactsIdling)
    }
}

@After
fun unregisterIdling() {
    idling {
      IdlingRegistry.getInstance().unregister(contactsIdling)
    }
}
```

Problems.stability.waits.IdlingResources

✓ Выполняется только на debug сборке

```
fun idling(action: IdlingScope.() -> Unit){
    if (!isReleaseBuild()){
        idlingScope.get()?.action()
    }
}
```

Problems.stability.waits.IdlingResources

✓ Выполняется только на debug сборке

✓ Выполняется только с контекстом автотестов

```
fun idling(action: IdlingScope.() -> Unit){
    if (!isReleaseBuild()){
        idlingScope.get()?.action()
    }
}
```

Demo

EspressoGuide [~/develop/EspressoGuide] - .../app/src/androidTest/java/com/atiurin/espressoguide/tests/DemoEspressoTest.kt [app]
sendMessage()
Pixel 2 XL API 28
Git:
DemoEspressoTest.kt
GetContacts.kt
BaseTest.kt
IdlingScope.kt
ScreenshotLifecycleListener.kt
1: Project
Resource Manager
Layout Captures
7: Structure
2: Favorites
Build Variants
Gradle
Device File Explorer
DemoEspressoTest › sendMessage()
4: Run
6: Logcat
TODO
Terminal
9: Version Control
Build
Profiler
Multi-OS Engine
Event Log
8:34
Sunday, Dec 1
VK

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

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

    @Test
    fun checkMessagesPositionsInChat() {...}

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

Tests passed: 1 (5 minutes ago)   50:16 LF UTF-8 4 spaces Git: master

Что в итоге с ожиданиями?

Что в итоге с ожиданиями?

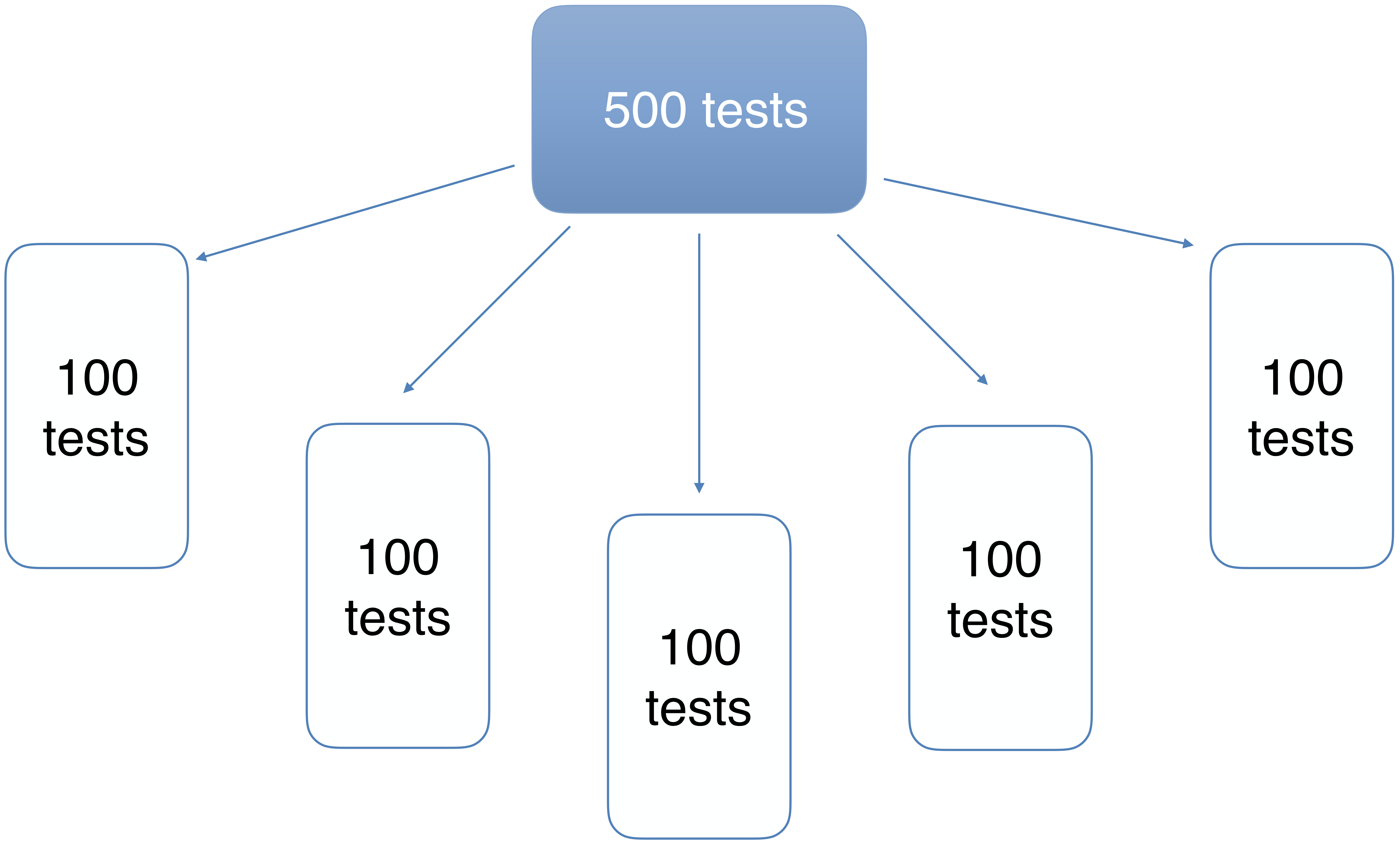
✓ FailureHandler — стабильность каждого действия

Что в итоге с ожиданиями?

✓ FailureHandler — стабильность каждого действия

✓ IdlingResources — стабильность долгих операций

Problems.stability.sharding



Problems.stability.sharding

- Spoon (github.com/square/spoon)
- Composer (github.com/gojuno/composer)
- Marathon (github.com/Malinskiy/marathon)
- Fork (github.com/shazam/fork)
- …

Problems.stability.sharding

- Spoon (github.com/square/spoon)
- Composer (github.com/gojuno/composer)
- Marathon (github.com/Malinskiy/marathon)
- Fork (github.com/shazam/fork)
- …
- Свой TestRunner

Problems.stability.sharding

- Запуск на нескольких девайсах

Problems.stability.sharding

- Запуск на нескольких девайсах
- Подключение различных устройств

Problems.stability.sharding

- Запуск на нескольких девайсах
- Подключение различных устройств
- Мониторинг flaky тестов

Problems.stability.sharding

- Запуск на нескольких девайсах
- Подключение различных устройств
- Мониторинг flaky тестов
- Отчет с результатами тестов

Problems.stability.flakiness

Problems.stability.flakiness

Problems.stability.flakiness

- **Fork —** flakiness reporter с интеграцией в Jenkins

Problems.stability.flakiness

- **Fork —** flakiness reporter с интеграцией в Jenkins
- **Marathon —** вычисляет Retry count на основе предыдущих прогонов

Problems.stability.flakiness

- **Fork —** flakiness reporter с интеграцией в Jenkins
- **Marathon —** вычисляет Retry count на основе предыдущих прогонов

- Flaky тесты продолжают выполняться!

Spoon
Fork
Сделать своё
Marathon
Composer

Сделать своё

Problems.stability.flakiness

Config ALL

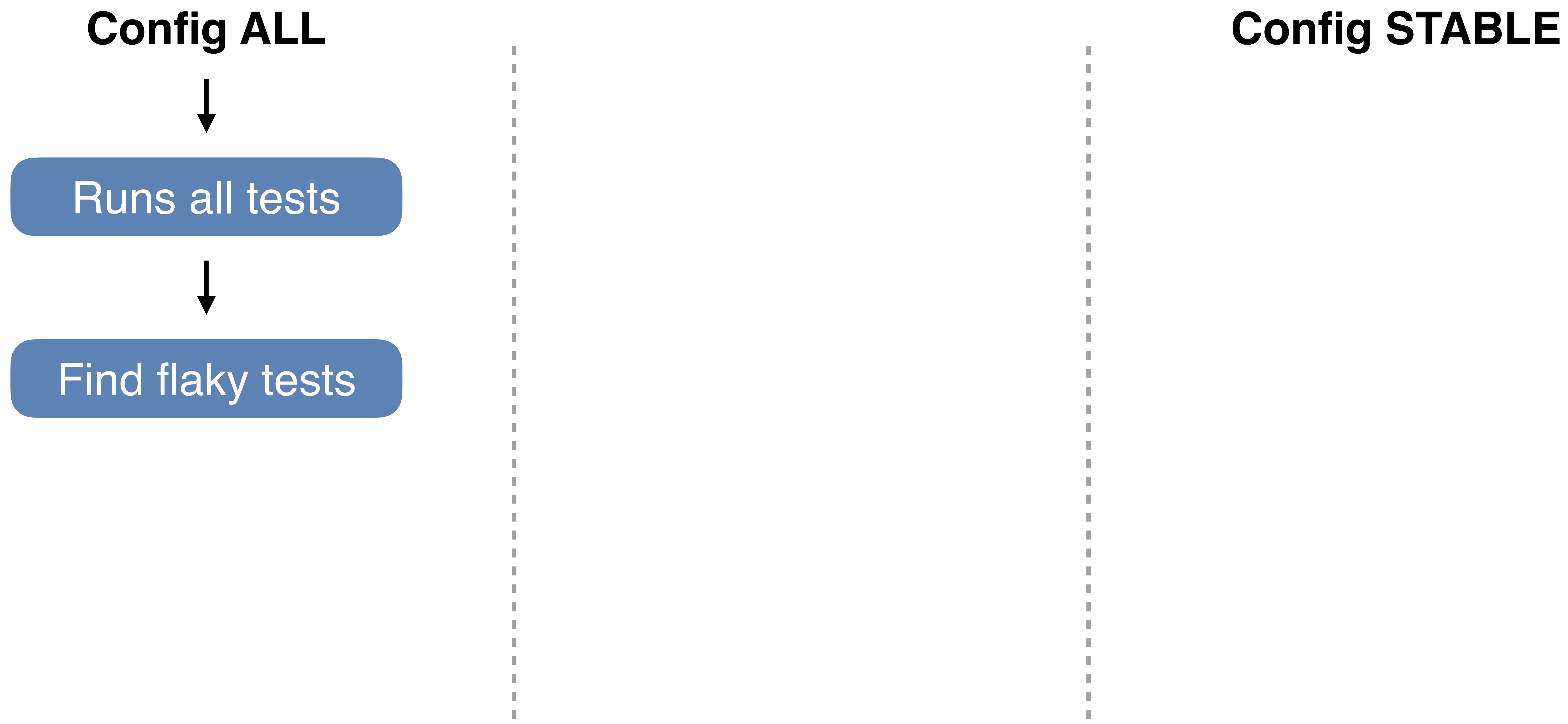
Config STABLE

Problems.stability.flakiness

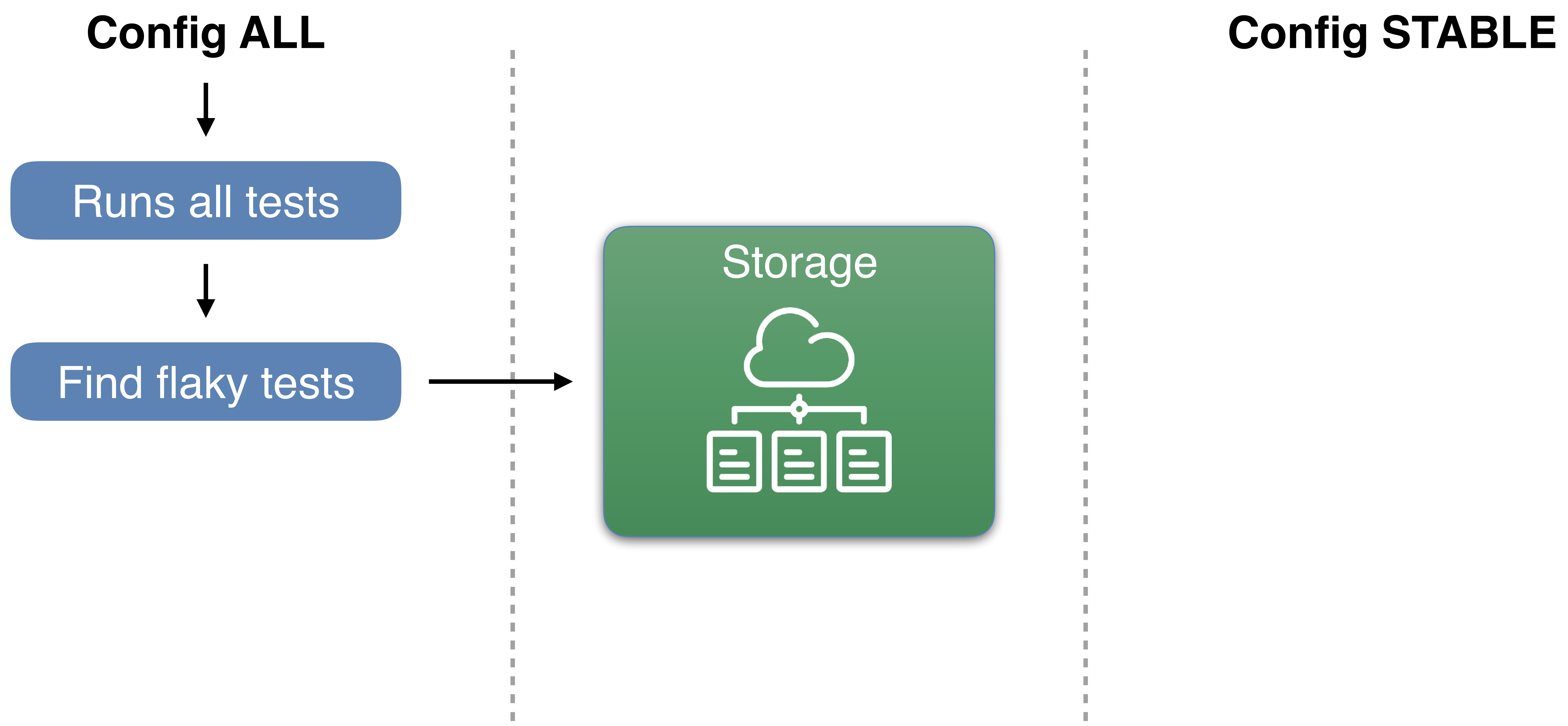
$$\begin{tikzcd}\text{Config ALL} \arrow[d] \\ \boxed{\text{Runs all tests}}\end{tikzcd}$$

Config STABLE

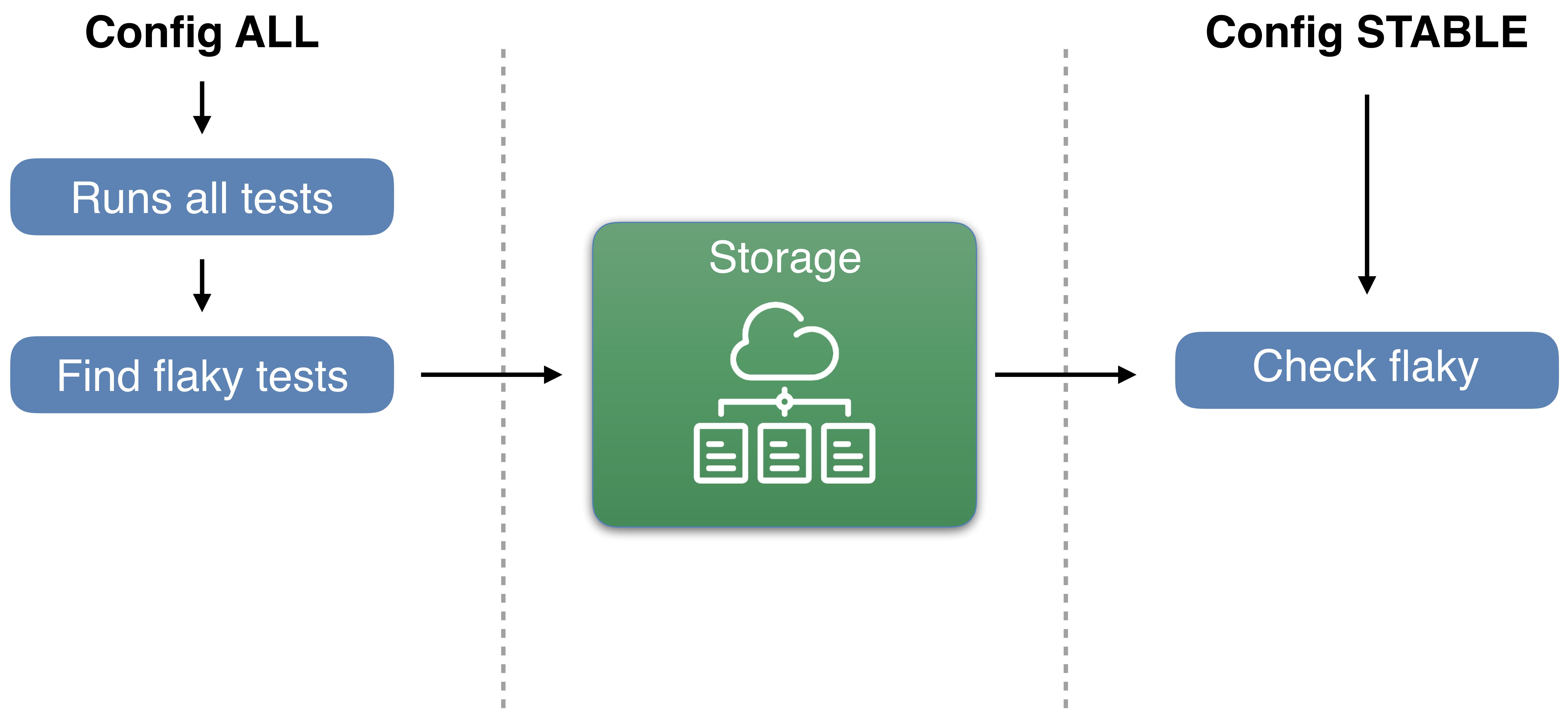
Problems.stability.flakiness



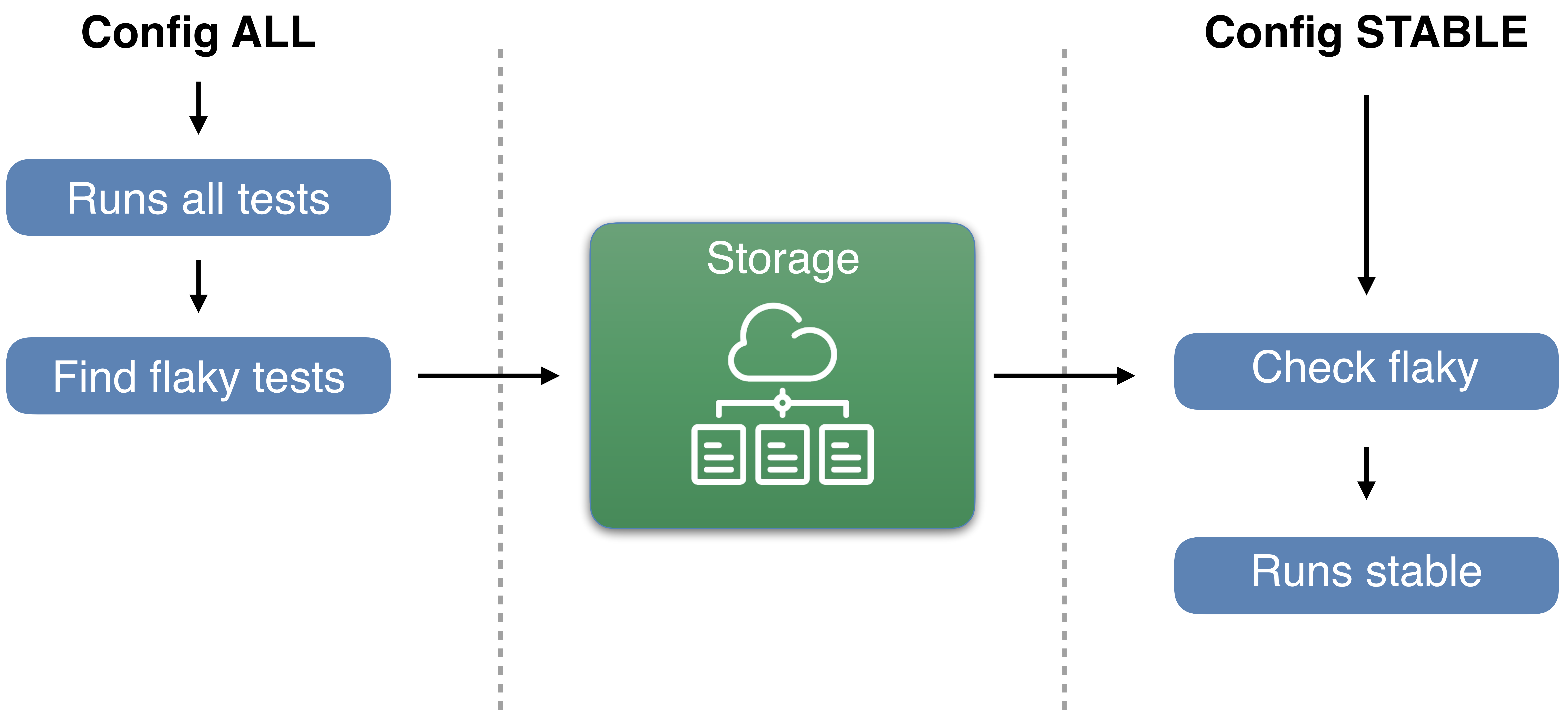
Problems.stability.flakiness



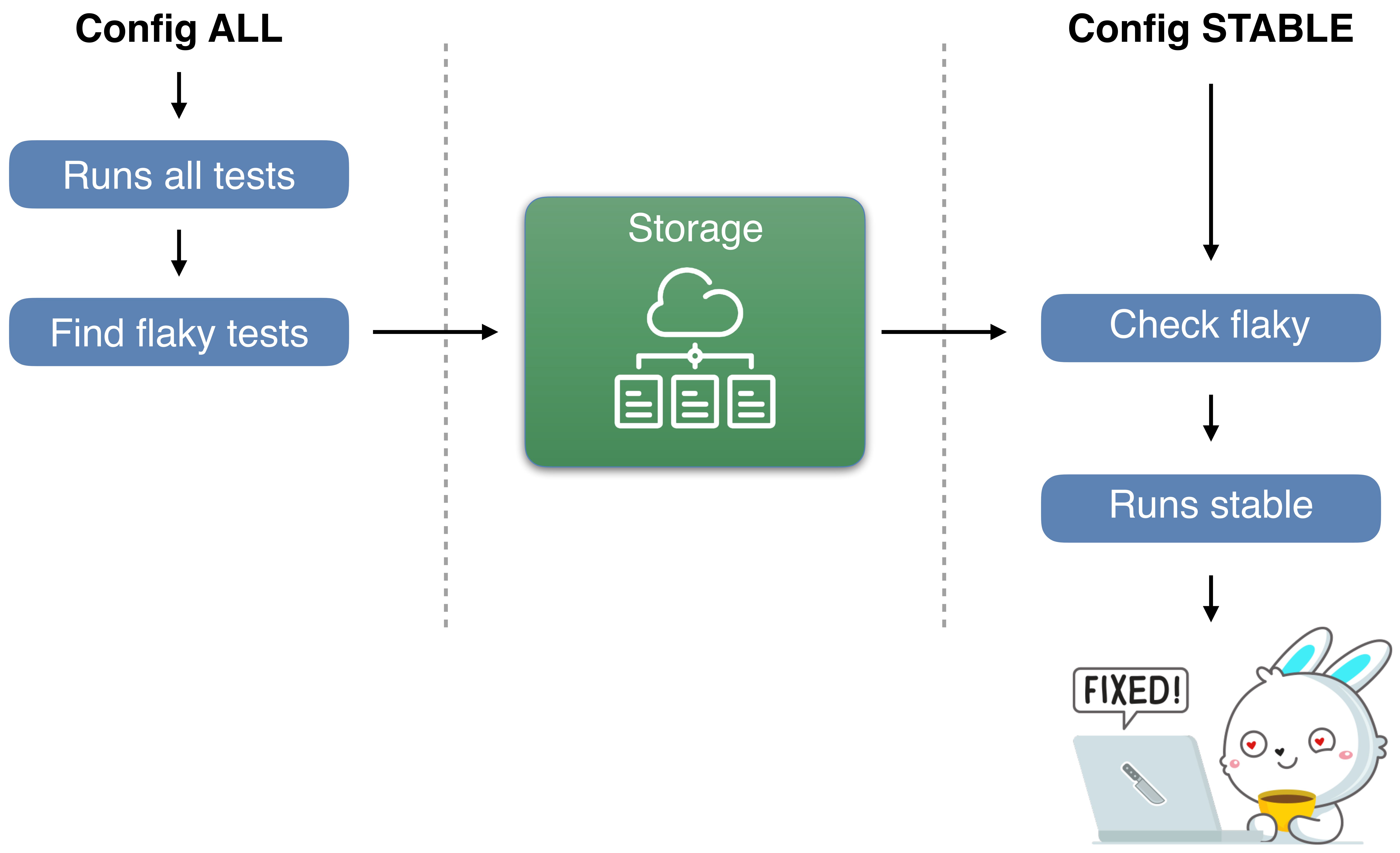
Problems.stability.flakiness



Problems.stability.flakiness



Problems.stability.flakiness



Какие проблемы решает TestRunner?

Какие проблемы решает TestRunner?

- Управление девайсами (подключение удаленных устройств)

Какие проблемы решает TestRunner?

- Управление девайсами (подключение удаленных устройств)
- Запуск тестов в параллель

Какие проблемы решает TestRunner?

- Управление девайсами (подключение удаленных устройств)
- Запуск тестов в параллель
- Мониторинг и управление flaky-тестами

Какие проблемы решает TestRunner?

- Управление девайсами (подключение удаленных устройств)
- Запуск тестов в параллель
- Мониторинг и управление flaky-тестами
- Сбор отчетов со всех девайсов

Problems

Support

Stability

Useless

Когда автотесты приносят пользу?

Когда автотесты приносят пользу?

✓ Поддержка тестов не заставляет вас ночевать на работе (PageObject + Steps + DSL)

Когда автотесты приносят пользу?

✓ Поддержка тестов не заставляет вас ночевать на работе (PageObject + Steps + DSL)

✓ К ним есть доверие (FailureHandler, IdlingResources, Flaky Management)

Когда автотесты приносят пользу?

- ✓ Поддержка тестов не заставляет вас ночевать на работе (PageObject + Steps + DSL)
- ✓ К ним есть доверие (FailureHandler, IdlingResources, Flaky Management)
- ✓ Тесты не выполняются часами (Tests Sharding)

Когда автотесты приносят пользу?

- ✓ Поддержка тестов не заставляет вас ночевать на работе (PageObject + Steps + DSL)
- ✓ К ним есть доверие (FailureHandler, IdlingResources, Flaky Management)
- ✓ Тесты не выполняются часами (Tests Sharding)
- Результат выполнения предельно понятен

Allure Android

Allure Android

build.gralde

```
testInstrumentationRunner "io.qameta.allure.espresso.AllureAndroidRunner"
```

Allure Android

build.gralde

```
testInstrumentationRunner "io.qameta.allure.espresso.AllureAndroidRunner"

dependencies {
   //Allure
   androidTestImplementation "io.qameta.allure:allure-android-commons:2.0.0"
   androidTestImplementation "io.qameta.allure:allure-android-model:2.0.0"
   androidTestImplementation "io.qameta.allure:allure-espresso:2.0.0"
}
```

Allure Android

```
abstract class BaseTest {





}
```

Allure Android

```
abstract class BaseTest {
    //attach screenshot to allure report in case of failure
    @get:Rule
    val failshot = FailshotRule()






}
```

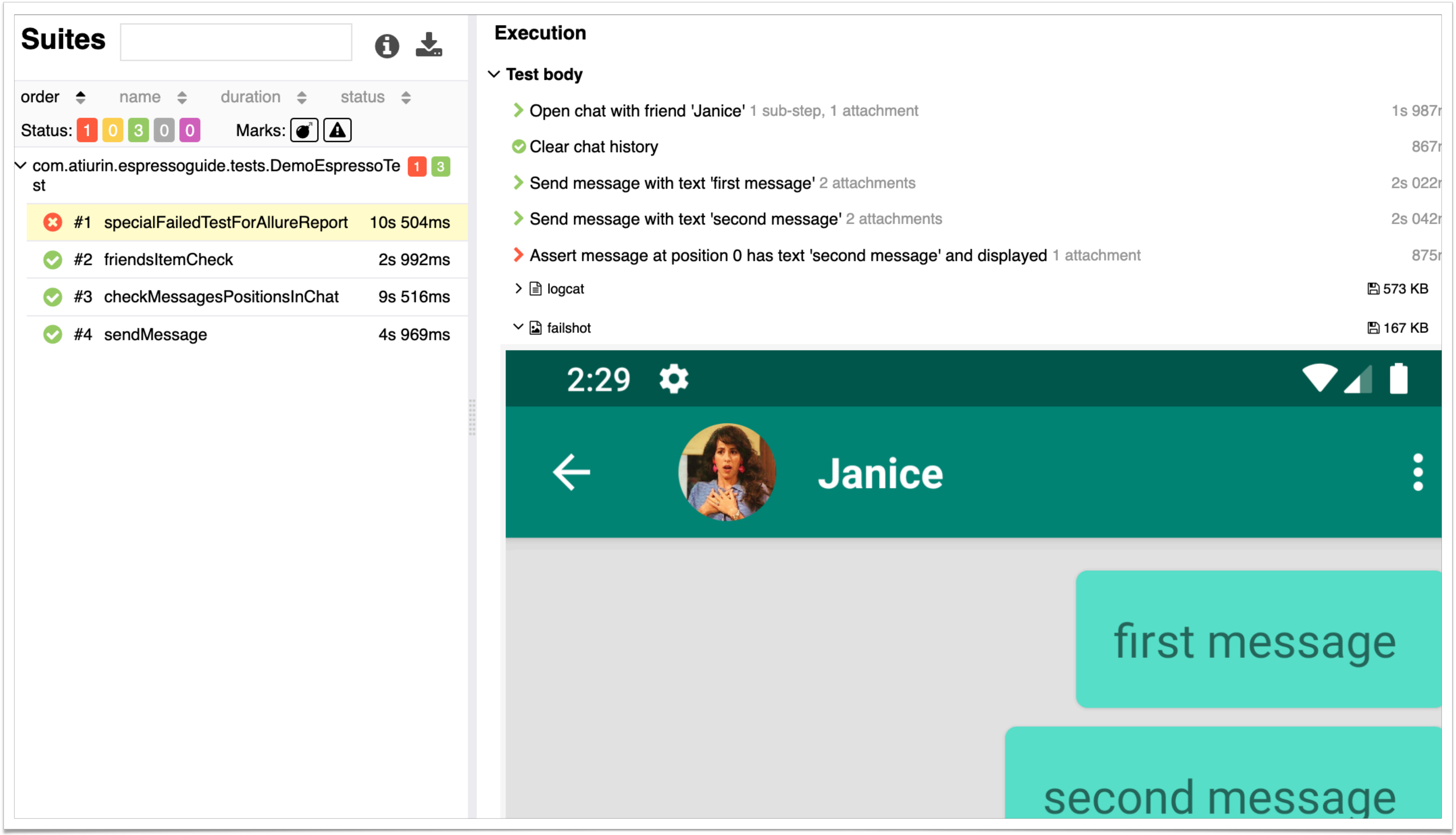
Allure Android

```
abstract class BaseTest {
    //attach screenshot to allure report in case of failure
    @get:Rule
    val failshot = FailshotRule()

    //attach logcat to allure report in case of failure
    @get:Rule
    val ruleChain = RuleChain.outerRule(LogcatClearRule())
                                   .around(LogcatDumpRule())
}
```

EspressoGuide [~/develop/EspressoGuide] - .../app/src/androidTest/java/com/atiurin/espressoguide/tests/DemoEspressoTest.kt [app]
DemoEspressoTest
Pixel 2 XL API 28
Git:
DemoEspressoTest.kt
GetContacts.kt
ChatPage.kt
Step.kt
BaseTest.kt
ScreenshotLifecycleListener.kt
1: Project
Resource Manager
Layout Captures
7: Structure
2: Favorites
Build Variants
Gradle
Device File Explorer
DemoEspressoTest
4: Run
6: Logcat
TODO
Terminal
9: Version Control
Build
Profiler
Multi-OS Engine
Event Log
32 chars
71:41
LF
UTF-8
4 spaces
Git: master
9:10
Sunday, Dec 1
VK

```kotlin
    @Test
    fun checkMessagesPositionsInChat() {...}

    @Test
    fun specialFailedTestForAllureReport() {
        val firstMessage = "first message"
        val secondMessage = "second message"
        FriendsListPage().openChat( name: "Janice")
        ChatPage { this: ChatPage
            clearHistory()
            sendMessage(firstMessage)
            sendMessage(secondMessage)
            assertMessageTextAtPosition( position: 0, secondMessag
            assertMessageTextAtPosition( position: 1, firstMessage
        }
    }
}
```

Tests failed: 1, passed: 3 (3 minutes ago)



Suites

order ⬍ name ⬍ duration ⬍ status ⬍

Status: 1 0 3 0 0 Marks:

com.atiurin.espressoguide.tests.DemoEspressoTest 1 3

✗	#1	specialFailedTestForAllureReport	10s 504ms
✓	#2	friendsItemCheck	2s 992ms
✓	#3	checkMessagesPositionsInChat	9s 516ms
✓	#4	sendMessage	4s 969ms

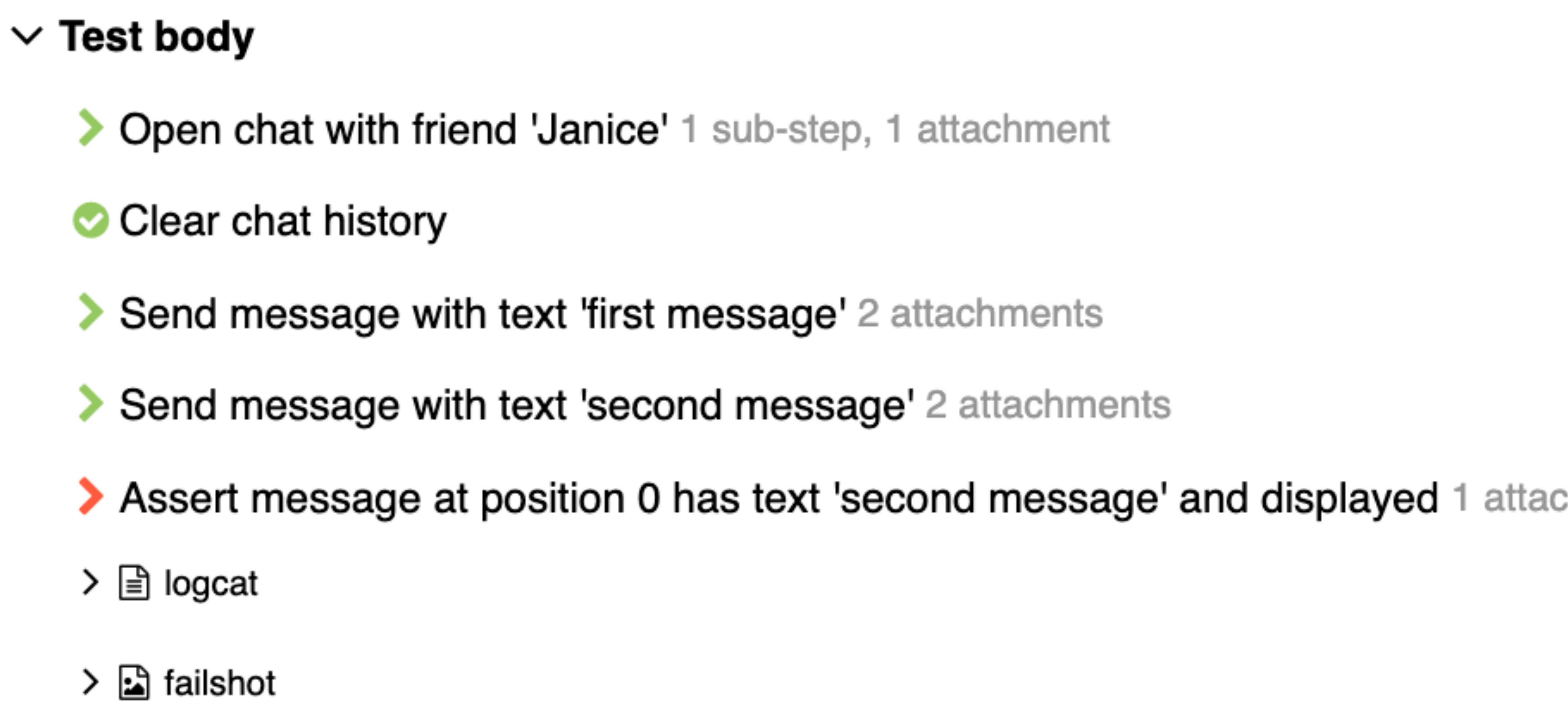
Execution

Test body

- Open chat with friend 'Janice' 1 sub-step, 1 attachment 1s 987r
- Clear chat history 867r
- Send message with text 'first message' 2 attachments 2s 022r
- Send message with text 'second message' 2 attachments 2s 042r
- Assert message at position 0 has text 'second message' and displayed 1 attachment 875r

logcat 573 KB

failshot 167 KB



Test body

Open chat with friend 'Janice' 1 sub-step, 1 attachment

Clear chat history

Send message with text 'first message' 2 attachments

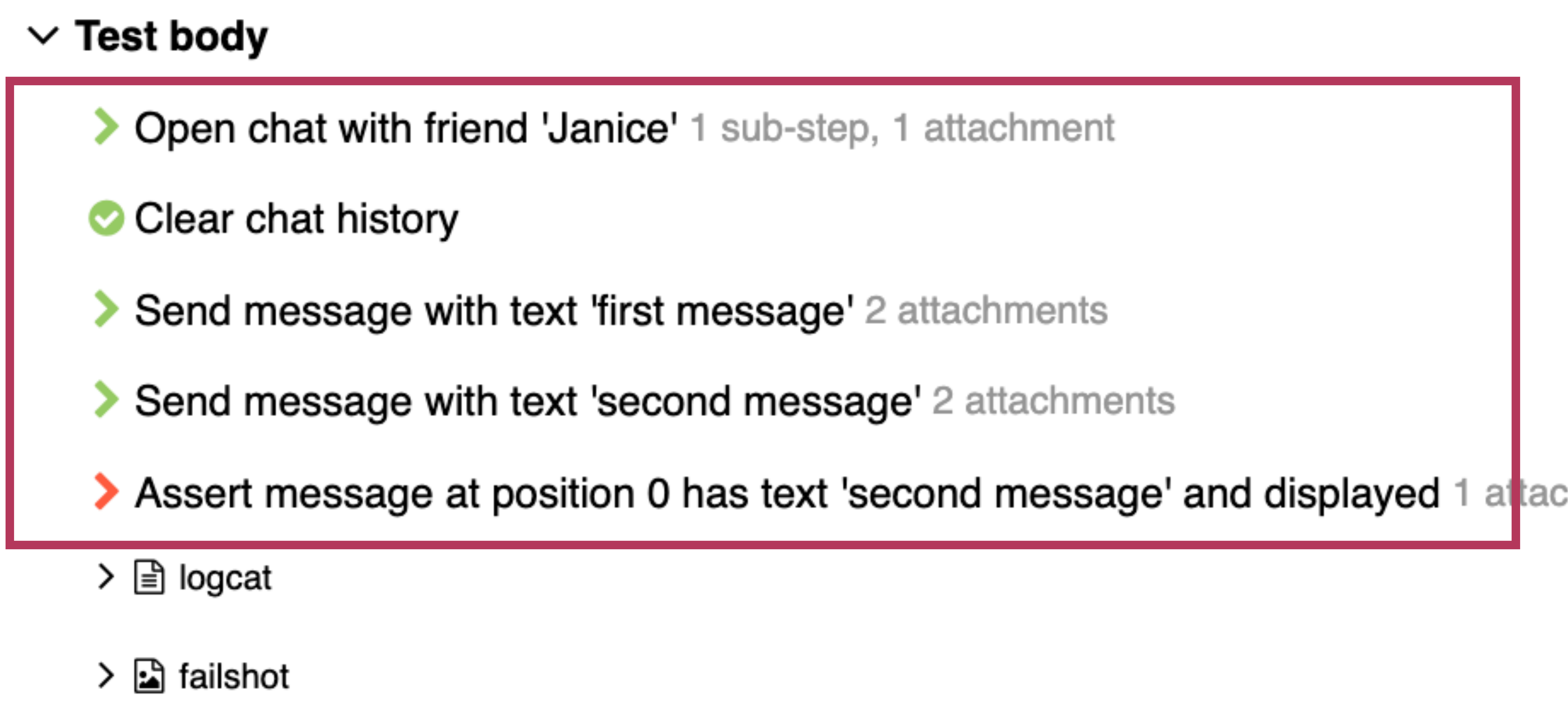
Send message with text 'second message' 2 attachments

Assert message at position 0 has text 'second message' and displayed 1 attac

logcat

failshot

```kotlin
@Test
fun specialFailedTestForAllureReport() {
    val firstMessage = "first message"
    val secondMessage = "second message"
    FriendsListPage().openChat("Janice")
    ChatPage {
        clearHistory()
        sendMessage(firstMessage)
        sendMessage(secondMessage)
        assertMessageTextAtPosition(0, secondMessage)
        assertMessageTextAtPosition(1, firstMessage)
    }
}
```


∨ Test body

› Open chat with friend 'Janice' 1 sub-step, 1 attachment

✓ Clear chat history

› Send message with text 'first message' 2 attachments

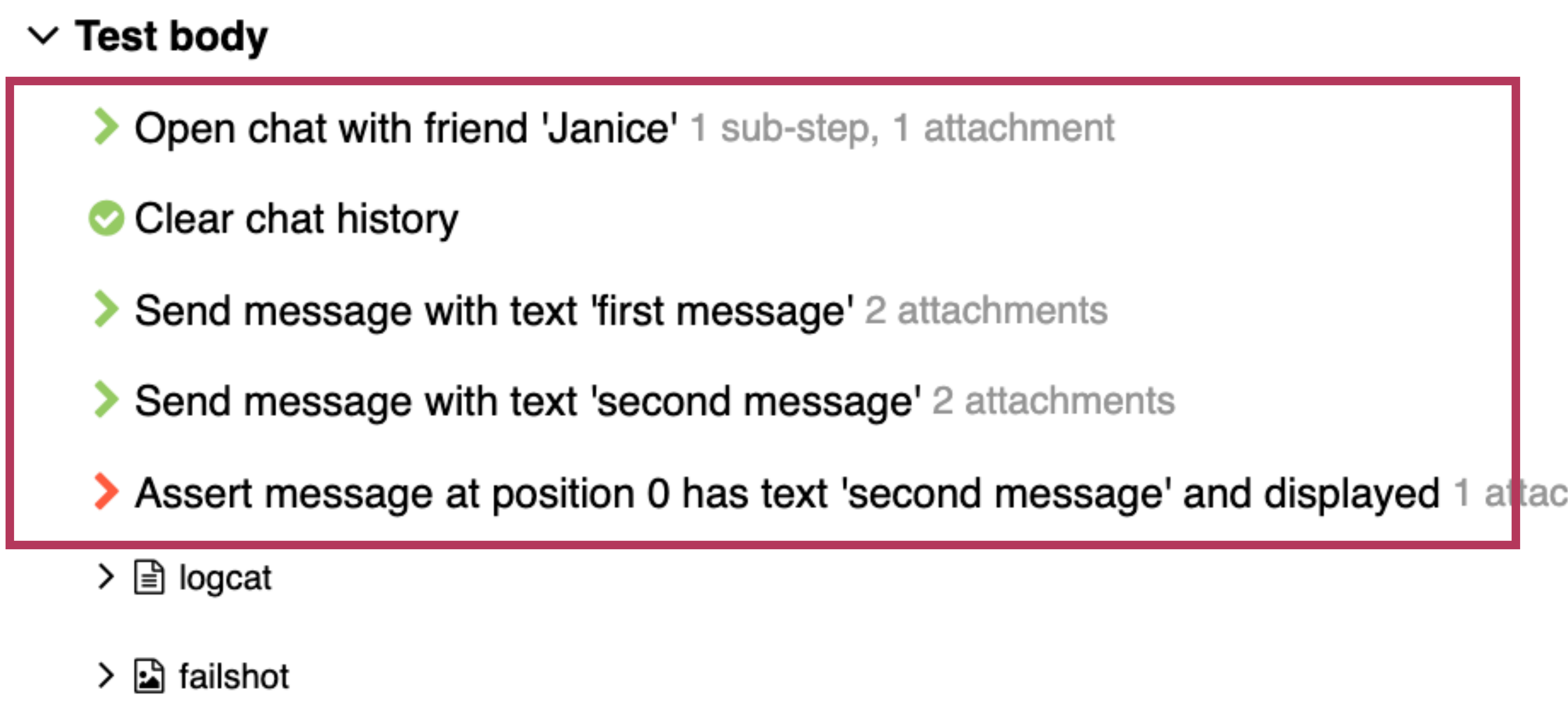
› Send message with text 'second message' 2 attachments

› Assert message at position 0 has text 'second message' and displayed 1 at

› logcat

› failshot

```kotlin
@Test
fun specialFailedTestForAllureReport() {
    val firstMessage = "first message"
    val secondMessage = "second message"
    FriendsListPage().openChat("Janice")
    ChatPage {
        clearHistory()
        sendMessage(firstMessage)
        sendMessage(secondMessage)
        assertMessageTextAtPosition(0, secondMessage)
        assertMessageTextAtPosition(1, firstMessage)
    }
}
```



Test body

Open chat with friend 'Janice' 1 sub-step, 1 attachment

Clear chat history

Send message with text 'first message' 2 attachments

Send message with text 'second message' 2 attachments

Assert message at position 0 has text 'second message' and displayed 1 attac

logcat

failshot

```
@Test
fun specialFailedTestForAllureReport() {
    val firstMessage = "first message"
    val secondMessage = "second message"
    FriendsListPage().openChat("Janice")
    ChatPage {
        clearHistory()
        sendMessage(firstMessage)
        sendMessage(secondMessage)
        assertMessageTextAtPosition(0, secondMessage)
        assertMessageTextAtPosition(1, firstMessage)
    }
}
```

Когда автотесты приносят пользу?

✓ Поддержка тестов не заставляет вас ночевать на работе (PageObject + Steps + DSL)

✓ К ним есть доверие (FailureHandler, IdlingResources, Flaky Management)

✓ Тесты не выполняются часами (Tests Sharding)

- Результат выполнения предельно понятен

Когда автотесты приносят пользу?

✓ Поддержка тестов не заставляет вас ночевать на работе (PageObject + Steps + DSL)

✓ К ним есть доверие (FailureHandler, IdlingResources, Flaky Management)

✓ Тесты не выполняются часами (Tests Sharding)

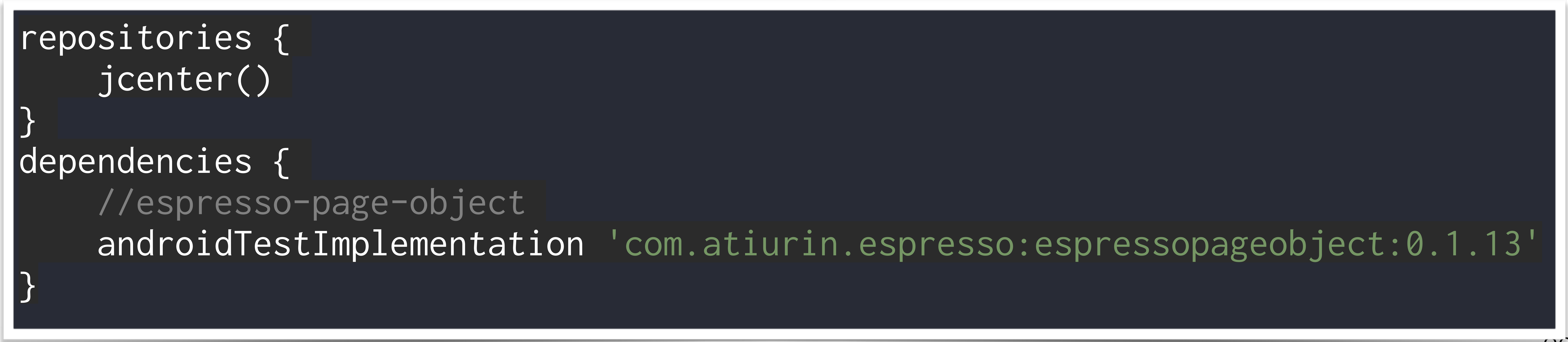
✓ Результат выполнения предельно понятен (Allure Report)

Когда автотесты приносят пользу?

- ✓ Поддержка тестов не заставляет вас ночевать на работе (PageObject + Steps + DSL)
- ✓ К ним есть доверие (FailureHandler, IdlingResources, Flaky Management)
- ✓ Тесты не выполняются часами (Tests Sharding)
- ✓ Результат выполнения предельно понятен (Allure 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:0.1.13'
}
```