Compose for iOS under the hood

Алексей Гладков

Mobile Developer

YOUTUBE mobiledeveloper

EMAIL mobiledevelopercourse@gmail.com

Mobius 2022 Autumn

Алексей Гладков

Обо мне

- Пишу нативные приложения уже 9 лет (Android и iOS)
- Автор канала **Mobile Developer**
- **Head of Mobile**

Что учить новичку в Android: Java vs Kotlin?...

47 тыс. просмотров • 11 месяцев назад •

RxJava - Observable, Flowable. Полный обзор....

32 тыс. просмотров • 3 года назад •

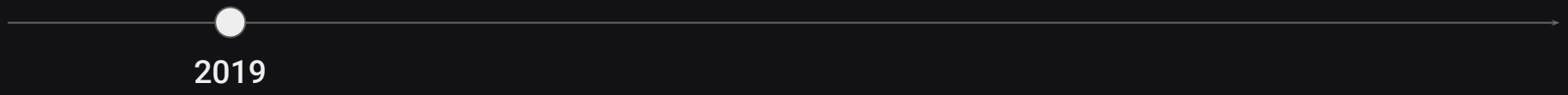
Dagger 2 - @Provides, @Module. Полный обзор....

22 тыс. просмотров • 3 года назад •

Контекст

Контекст

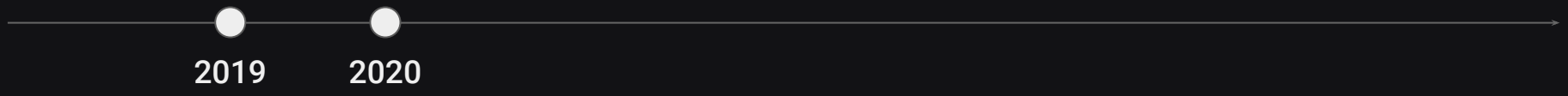
> Jetpack Compose Alpha



Контекст

> Jetpack Compose Alpha

> Compose Multiplatform Desktop

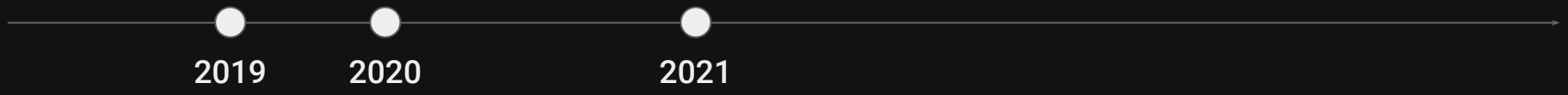


Контекст

> Jetpack Compose Alpha

> Compose Multiplatform Desktop

> Jetpack Compose Release



Контекст

> Compose Multiplatform Release

2019 — 2020 — 2021 — Dec 2021

Контекст

> Compose Multiplatform Release

> Android, Desktop

2019 — 2020 — 2021 — Dec 2021

Контекст

> Compose for iOS Dev Preview

2019 — 2020 — 2021 — Dec 2021 — Aug 2022

Контекст

> Compose for iOS Dev Preview

> Compose for Web Dev Preview

2019 — 2020 — 2021 — Dec 2021 — Aug 2022

Контекст

> Droidcon 2022 Release with Compose for iOS

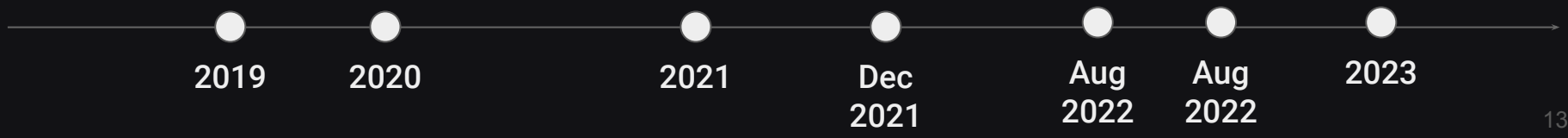
Контекст

> KotlinConf 2023

Контекст

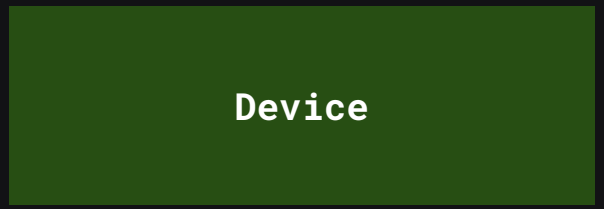
> KotlinConf 2023

> Compose for iOS Alpha

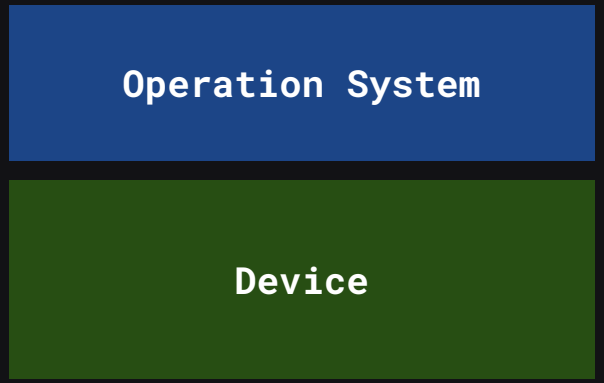



MTV
DK

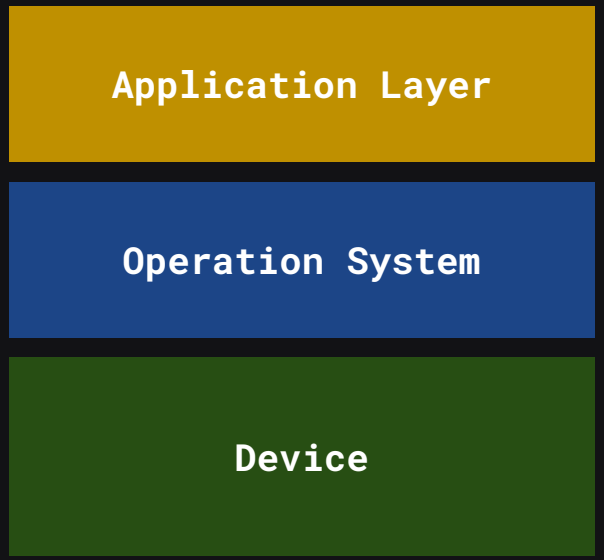
Архитектура мобильных устройств



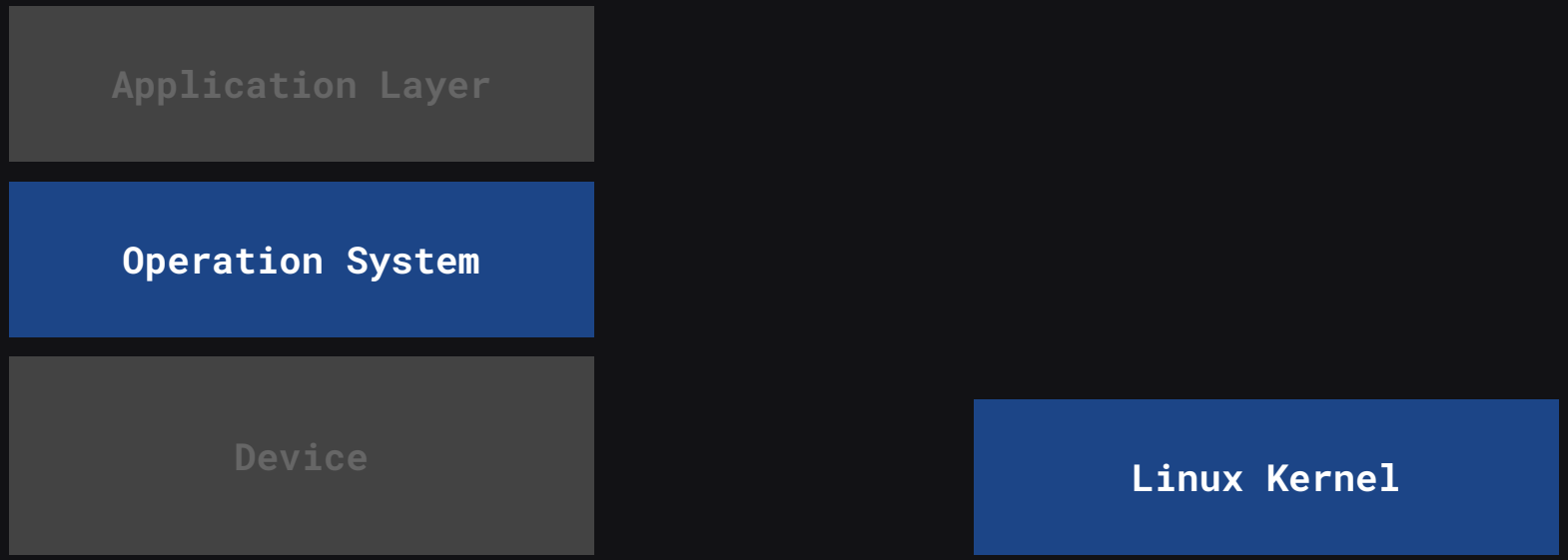
Архитектура мобильных устройств



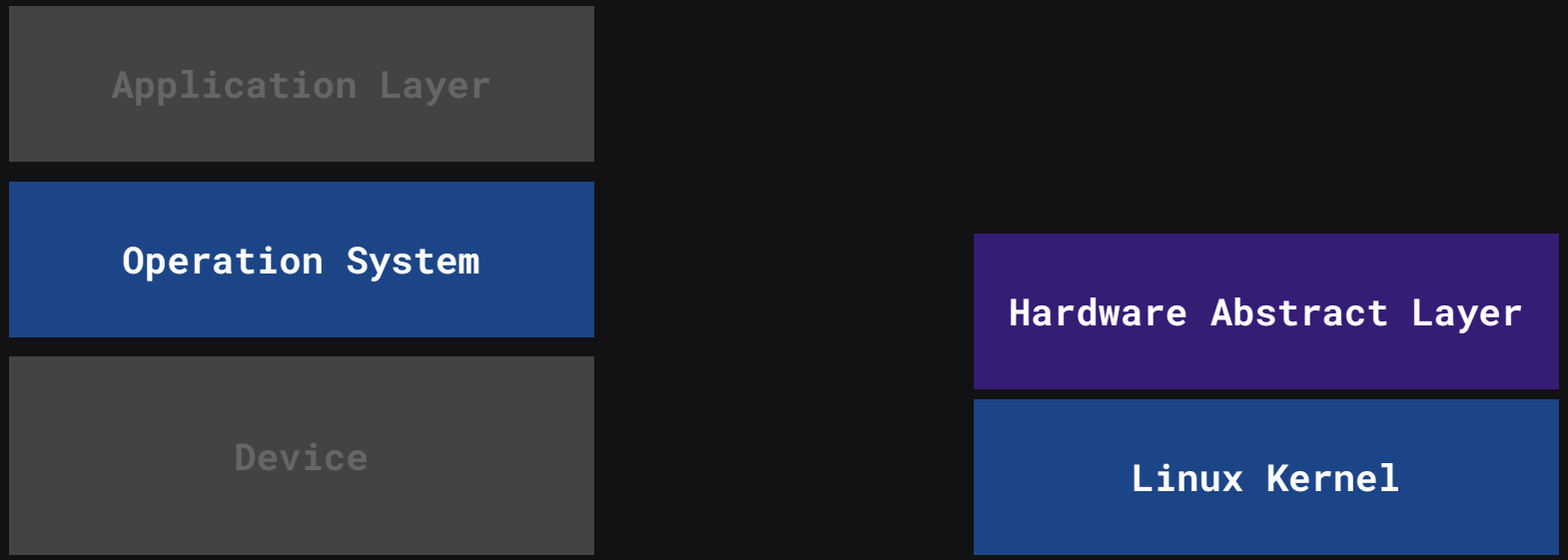
Архитектура мобильных устройств



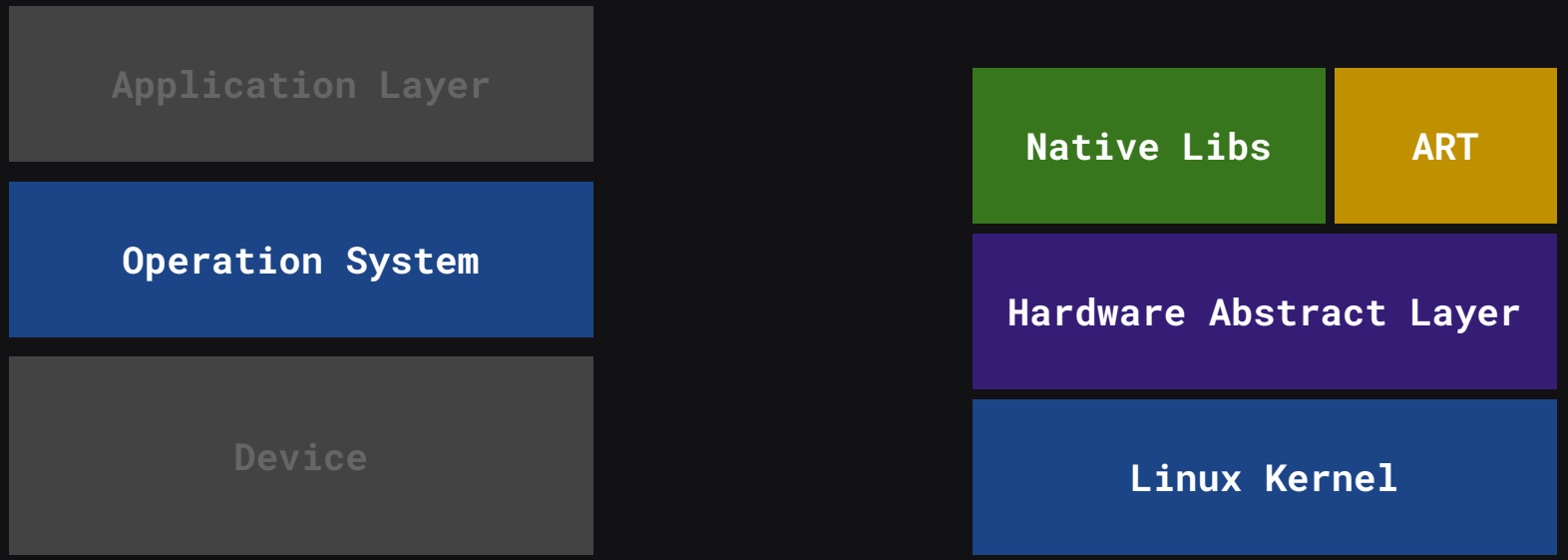
Архитектура Android



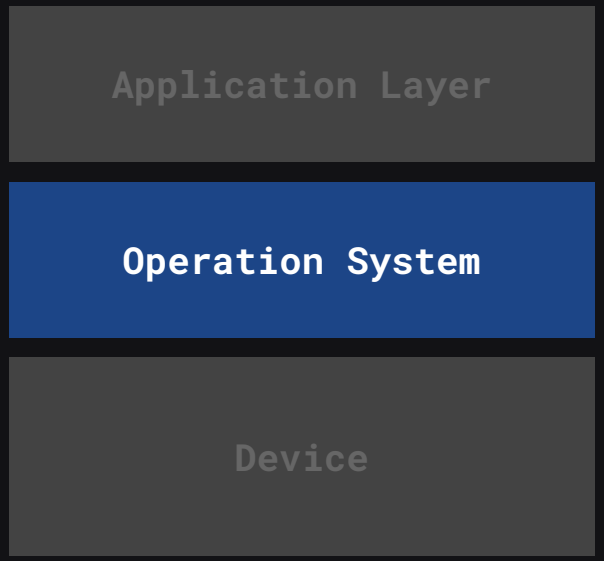
Архитектура Android

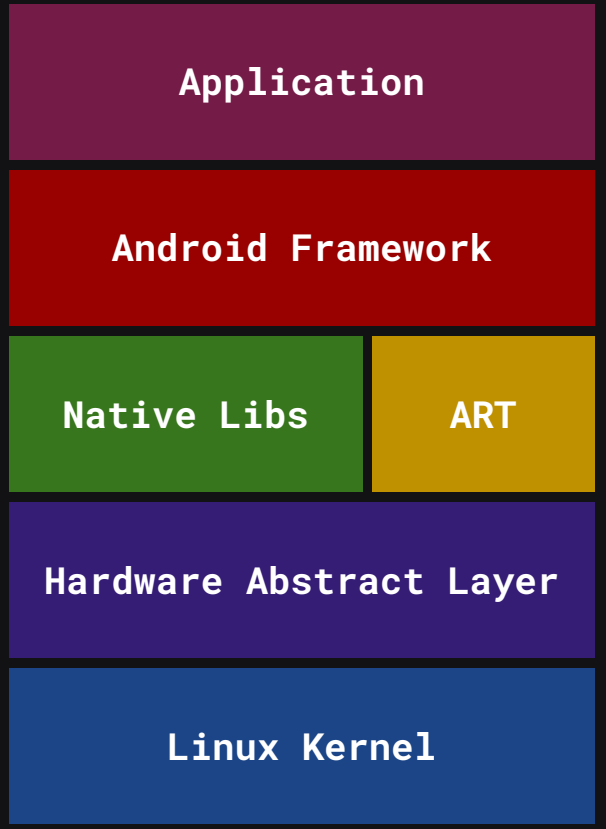


Архитектура Android

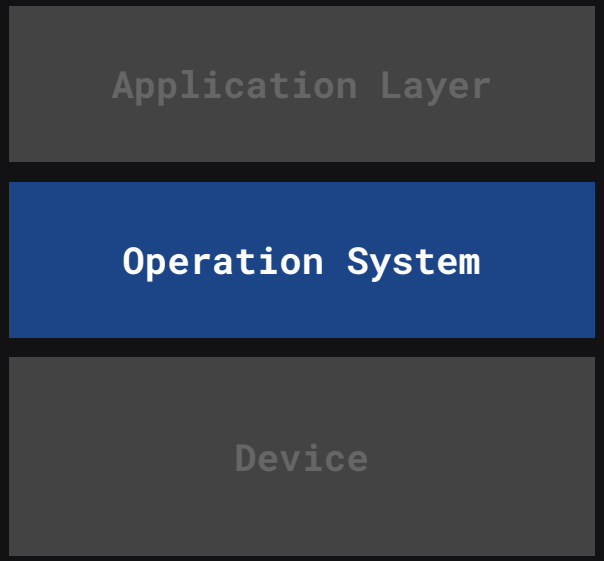


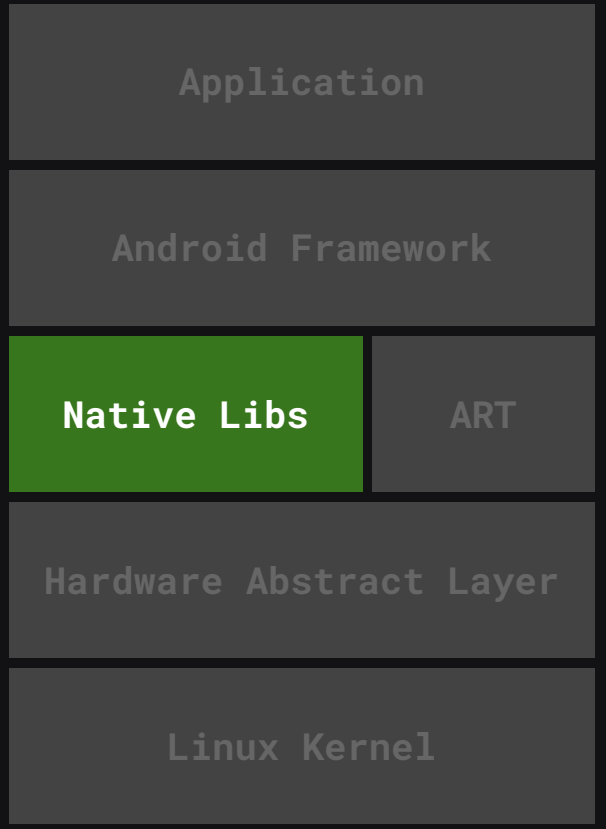
Архитектура Android



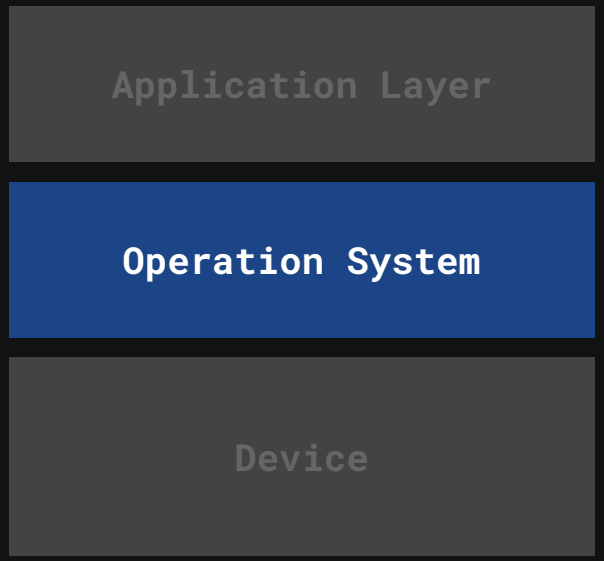


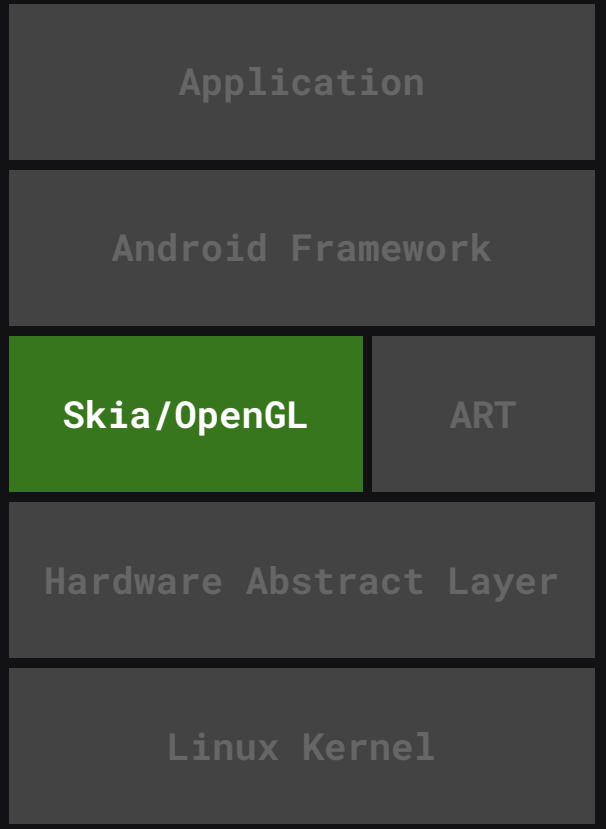
Архитектура Android





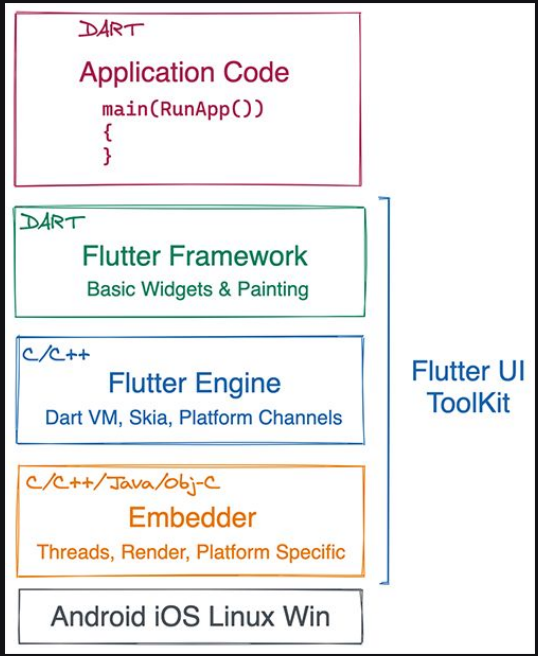
Архитектура Android



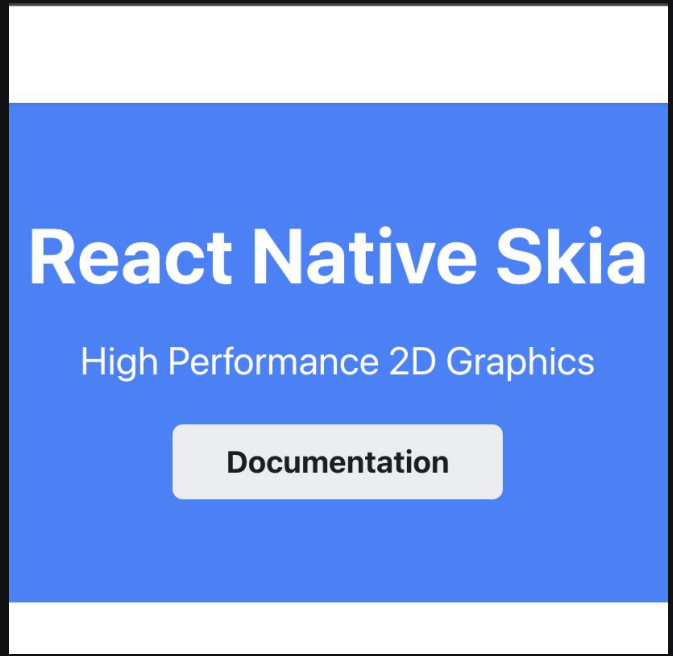


А также

Flutter



React Native



Skiko

Key #1

> Skiko - это мультиплатформенные биндинги к Skia, написанные компанией JetBrains

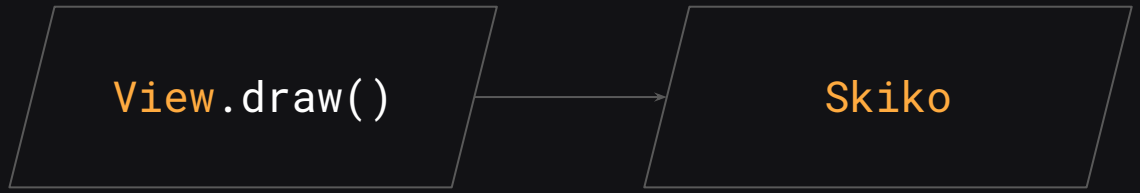
Skiko

androidMain/kotlin/org/jetbrains/skiko	take surface props into account when using direct3d (#613)	2 months ago
awtMain	Linux fix deadlock when we close a window (#646)	2 weeks ago
awtTest/kotlin/org/jetbrains/skiko	Move some files from jvmTest to awtTest and from jvmMain to awtMain (#...	5 months ago
commonMain	Fix invalid bindings (#638)	last month
commonTest	Fix invalid bindings (#638)	last month
darwinMain/kotlin/org/jetbrains/skiko	macos dispatcher (#295)	last year
darwinTest/kotlin/org/jetbrains/skiko	Implement common test resource loading (#303)	last year
iosMain/kotlin/org/jetbrains/skiko	Add keyboard customisation (iOS) (#622)	2 months ago
iosTest	Ios test with resource (#302)	last year
jsMain	take surface props into account when using direct3d (#613)	2 months ago
jsTest/kotlin	add docs 1..15 (#489)	last year
jvmMain	Linux fix deadlock when we close a window (#646)	2 weeks ago
jvmTest	Adapter exclusion (#617)	2 months ago
linuxMain/kotlin/org/jetbrains/skiko	take surface props into account when using direct3d (#613)	2 months ago
linuxTest/kotlin/org/jetbrains/skiko	Implement common test resource loading (#303)	last year
macosMain/kotlin/org/jetbrains/skiko	use kotlin 1.8.0 (#644)	2 weeks ago
nativeJsMain	Fix invalid bindings (#638)	last month
nativeJsTest/cpp	Implement Surface and DirectContext for native (#379)	last year

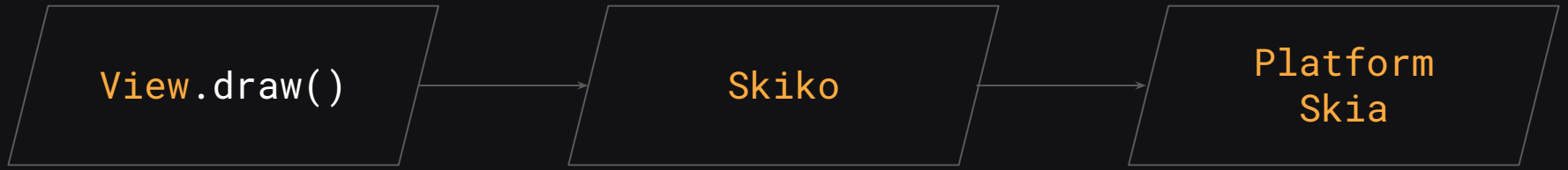
Rendering

```
View.draw()
```

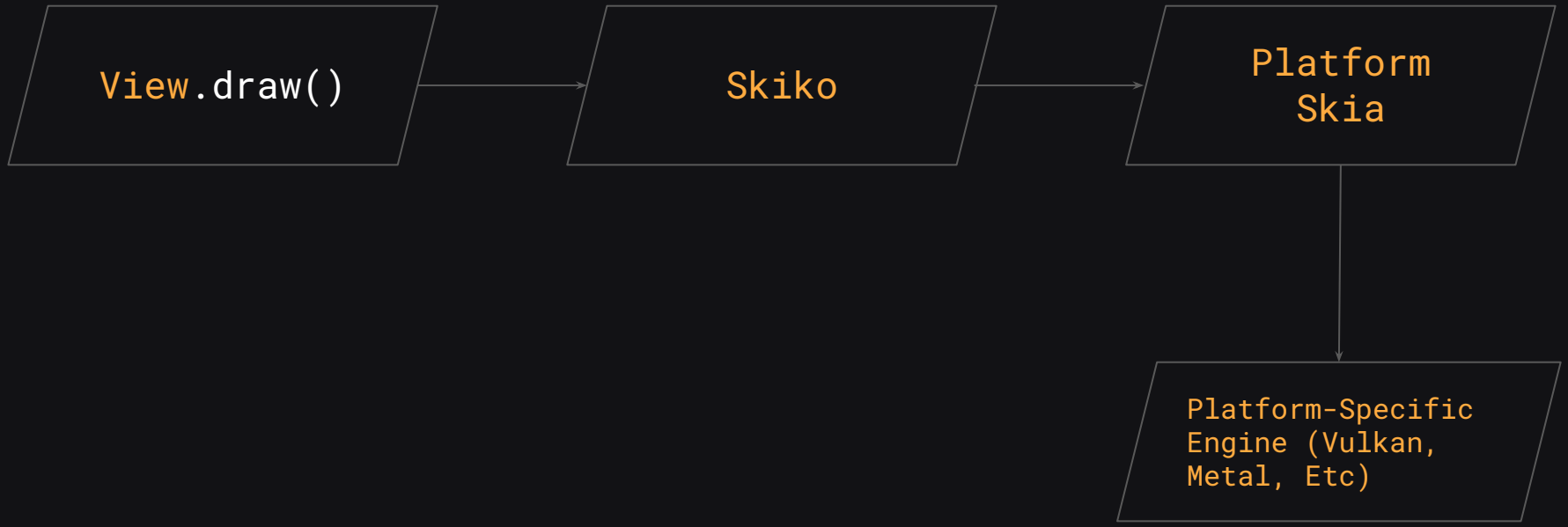
Rendering



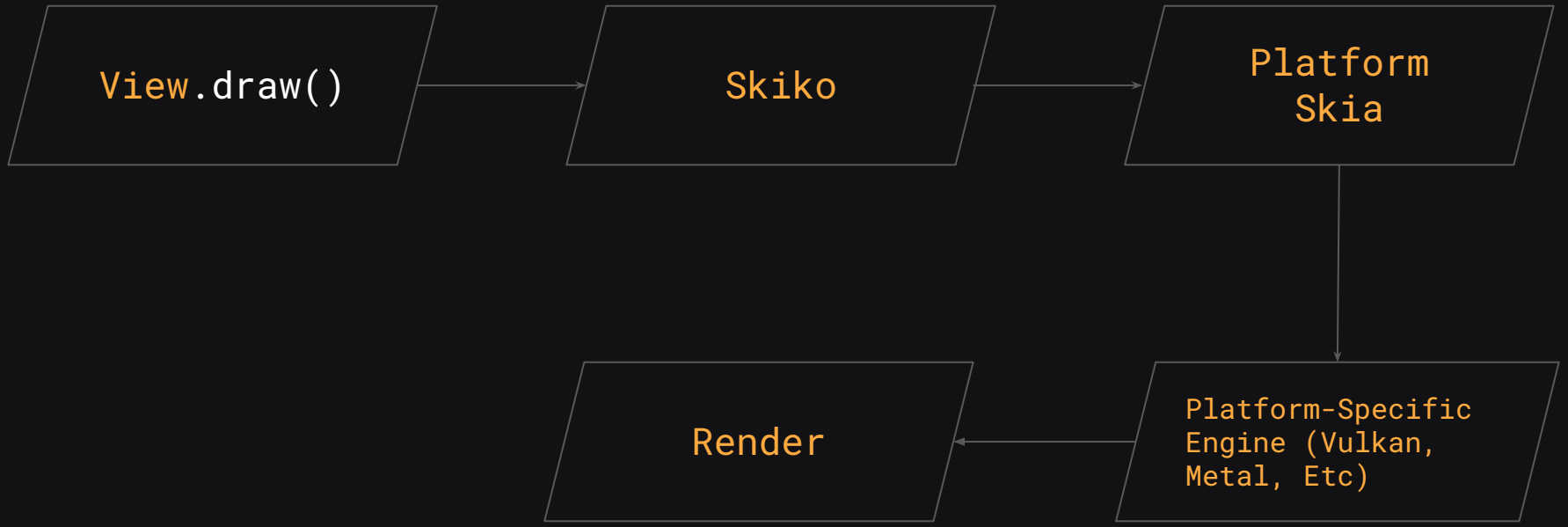
Rendering



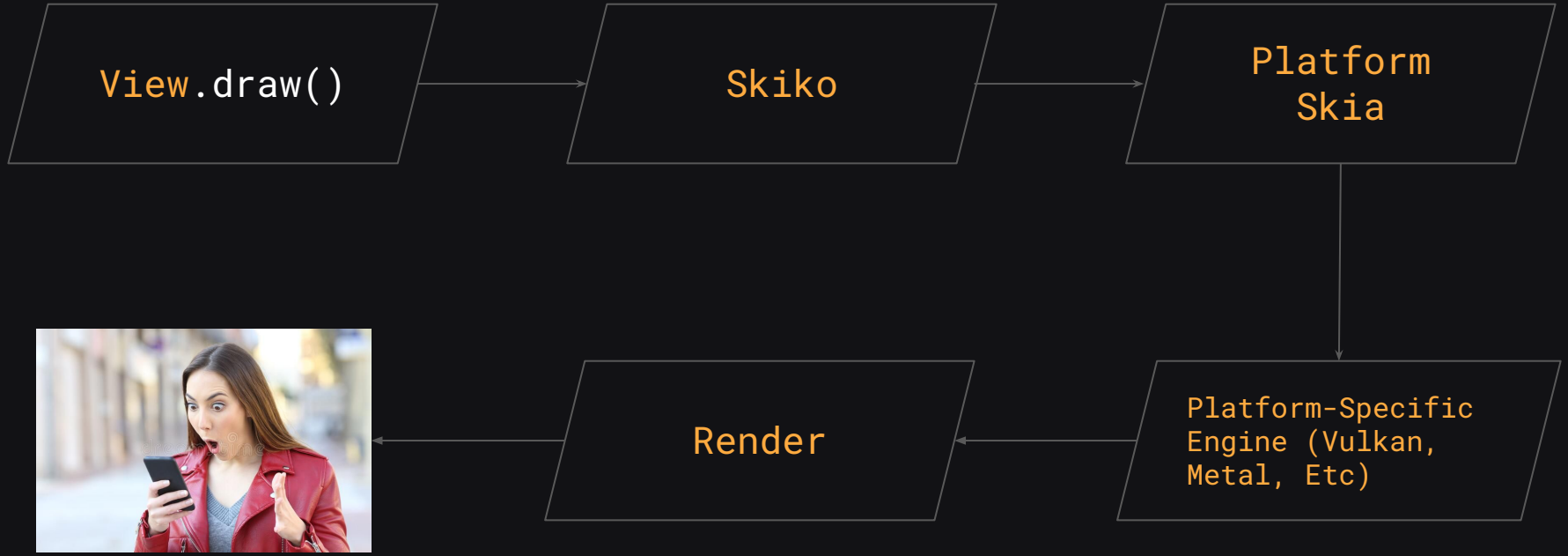
Rendering


View.draw()
Skiko
Platform
Skia
Platform-Specific
Engine (Vulkan,
Metal, Etc)

Rendering


View.draw()
Skiko
Platform Skia
Platform-Specific Engine (Vulkan, Metal, Etc)
Render

Rendering


View.draw()
Skiko
Platform Skia
Platform-Specific Engine (Vulkan, Metal, Etc)
Render

Skiko -> iOS

Skiko -> iOS

> Поддержка клавиатуры

Skiko -> iOS

> Поддержка клавиатуры

> Поддержка Metal

Skiko -> iOS

> Поддержка клавиатуры

> Поддержка Metal

> Поддержка жизненного цикла

Skiko -> iOS

> Поддержка клавиатуры

> Поддержка Metal

> Поддержка жизненного цикла

> Поддержка разных архитектур

Skiko -> iOS

> Поддержка клавиатуры

> Поддержка Metal

> Поддержка жизненного цикла

> Поддержка разных архитектур

> Поддержка жестов

Skiko -> iOS

> Поддержка клавиатуры

> Поддержка Metal

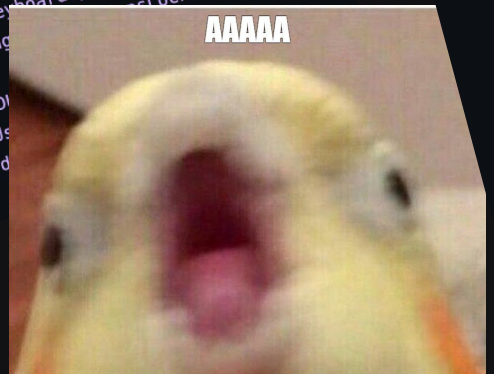
> Поддержка жизненного цикла

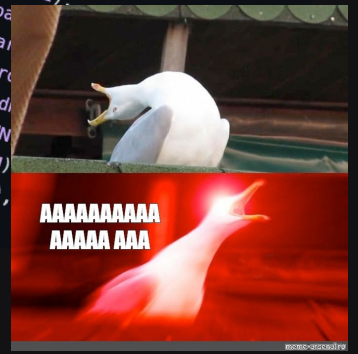
> Поддержка разных архитектур

> Поддержка жестов

> Поддержка системной темы

Клавиатура





Клавиатура

```
fun showScreenKeyboard() = becomeFirstResponder()
fun hideScreenKeyboard() = resignFirstResponder()
fun isScreenKeyboardOpen() = isFirstResponder

/**
 * A Boolean value that indicates whether the text-entry object has any text.
 * https://developer.apple.com/documentation/uikit/uikeyinput/1614457-hastext
 */
override fun hasText(): Boolean {
    return skiaLayer?.skikoView?.input?.hasText() ?: false
}
```

Жизненный цикл

```
@Suppress("CONFLICTING_OVERLOADS")
@ExportObjCClass
class SkikoUIView : UIView, UIKeyInputProtocol, UITextInputProtocol {
    @OverrideInit
    constructor(frame: CValue<CGRect>) : super(frame)

    @OverrideInit
    constructor(coder: NSCoder) : super(coder)

    private var skiaLayer: SkiaLayer? = null
    private var _inputDelegate: UITextInputDelegateProtocol? = null
    private var _currentTextMenuActions: TextActions? = null
    var currentKeyboardType: UIKeyboardType = UIKeyboardTypeDefault
    var currentKeyboardAppearance: UIKeyboardAppearance = UIKeyboardAppearanceDefault
    var currentReturnKeyType: UIReturnKeyType = UIReturnKeyType.UIReturnKeyDefault
    var currentTextContentType: UITextContentType? = null
    var currentIsSecureTextEntry: Boolean = false
    var currentEnablesReturnKeyAutomatically: Boolean = false
    var currentAutocapitalizationType: UITextAutocapitalizationType = UITextAutocapitalizationType.UITextAutocapitalizationTypeSentences
    var currentAutocorrectionType: UITextAutocorrectionType = UITextAutocorrectionType.UITextAutocorrectionTypeYes

    constructor(skiaLayer: SkiaLayer, frame: CValue<CGRect> = CGRectNull.readValue()) : super(frame) {
        this.skiaLayer = skiaLayer
    }
```

Жизненный цикл

```
@ExportObjCClass
class SkikoViewController : UIViewController {

    constructor(skikoUIView: SkikoUIView) : this() {
        this.skikoUIView = skikoUIView
    }

    private var skikoUIView: SkikoUIView? = null
```

Жизненный цикл

```
override fun loadView() {
    if (skikoUIView == null) {
        super.loadView()
    } else {
        this.view = skikoUIView!!.load()
    }
}
```

Жесты

```
@ObjCAction
fun onTap(sender: UITapGestureRecognizer) {
    val (x, y) = sender.locationInView(view).useContents { x to y }
    layer.skikoView?.onGestureEvent(
        SkikoGestureEvent(
            x = x,
            y = y,
            kind = SkikoGestureEventKind.TAP,
            state = toSkikoGestureState(sender.state)
        )
    )
}
```

Жесты

```
@ObjCAction
fun onTap(sender: UITapGestureRecognizer) {
    val (x, y) = sender.locationInView(view).useContents { x to y }
    layer.skikoView?.onGestureEvent(
        SkikoGestureEvent(
            x = x,
            y = y,
            kind = SkikoGestureEventKind.TAP,
            state = toSkikoGestureState(sender.state)
        )
    )
}
```

Жесты

```
@ObjCAction
fun onTap(sender: UITapGestureRecognizer) {
    val (x, y) = sender.locationInView(view).useContents { x to y }
    layer.skikoView?.onGestureEvent(
        SkikoGestureEvent(
            x = x,
            y = y,
            kind = SkikoGestureEventKind.TAP,
            state = toSkikoGestureState(sender.state)
        )
    )
}
```

Тема

```
actual val currentSystemTheme: SystemTheme
    get() = when (UITraitCollection.currentTraitCollection.userInterfaceStyle) {
        UIUserInterfaceStyleDark -> SystemTheme.DARK
        UIUserInterfaceStyleLight -> SystemTheme.LIGHT
        else -> SystemTheme.UNKNOWN
    }
```

Compose -> Skiko

Compose -> Skiko

```
val supportedPlatforms: Set<ComposePlatforms> = ComposePlatforms.ALL
- ComposePlatforms.NO_SKIKO
```

Compose -> Skiko

```
val supportedPlatforms: Set<ComposePlatforms> = ComposePlatforms.ALL
- ComposePlatforms.NO_SKIKO
```

Compose -> Skiko

```
// These platforms are not supported by skiko yet
val NO_SKIKO = EnumSet.of(
    ComposePlatforms.TvosArm64,
    ComposePlatforms.TvosX64,
    ComposePlatforms.TvosSimulatorArm64,
    ComposePlatforms.WatchosArm64,
    ComposePlatforms.WatchosArm32,
    ComposePlatforms.WatchosX86,
    ComposePlatforms.WatchosX64,
    ComposePlatforms.WatchosSimulatorArm64,
    ComposePlatforms.LinuxX64,
    // ComposePlatforms.LinuxArm64, // No coroutines for linuxArm64 yet
    ComposePlatforms.MingwX64,
)
```

Compose -> Skiko

```
@Composable
fun LazyColumn(
    modifier: Modifier = Modifier,
    state: LazyListState = rememberLazyListState(),
    contentPadding: PaddingValues = PaddingValues(0.dp),
    reverseLayout: Boolean = false,
    verticalArrangement: Arrangement.Vertical =
        if (!reverseLayout) Arrangement.Top else Arrangement.Bottom,
    horizontalAlignment: Alignment.Horizontal = Alignment.Start,
    flingBehavior: FlingBehavior = ScrollableDefaults.flingBehavior(),
    userScrollEnabled: Boolean = true,
    content: LazyListScope.() -> Unit
)
```

Compose -> Skiko

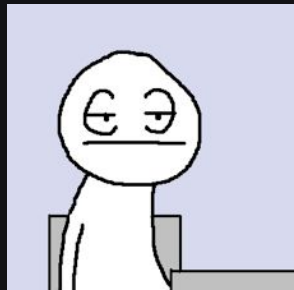
```
@Composable
fun LazyColumn(
    ...
) {
    LazyList(...)
}
```

Compose -> Skiko

```
@Composable
internal fun LazyList(
   ...
) {
    LazyLayout(
        modifier = modifier
            .scrollable(...)
    )
}
```

Compose -> Skiko

```
@ExperimentalFoundationApi
fun Modifier.scrollable(
    ...
) {
    Modifier
        .focusGroup()
        .then(keepFocusedChildInViewModifier.modifier)
        .pointerScrollable(...)
}
```



Compose -> Skiko

```
@Composable
@OptIn(ExperimentalFoundationApi::class)
private fun Modifier.pointerScrollable(
    ...
) {
    val scrollConfig = platformScrollConfig()
}
```

Compose -> Skiko

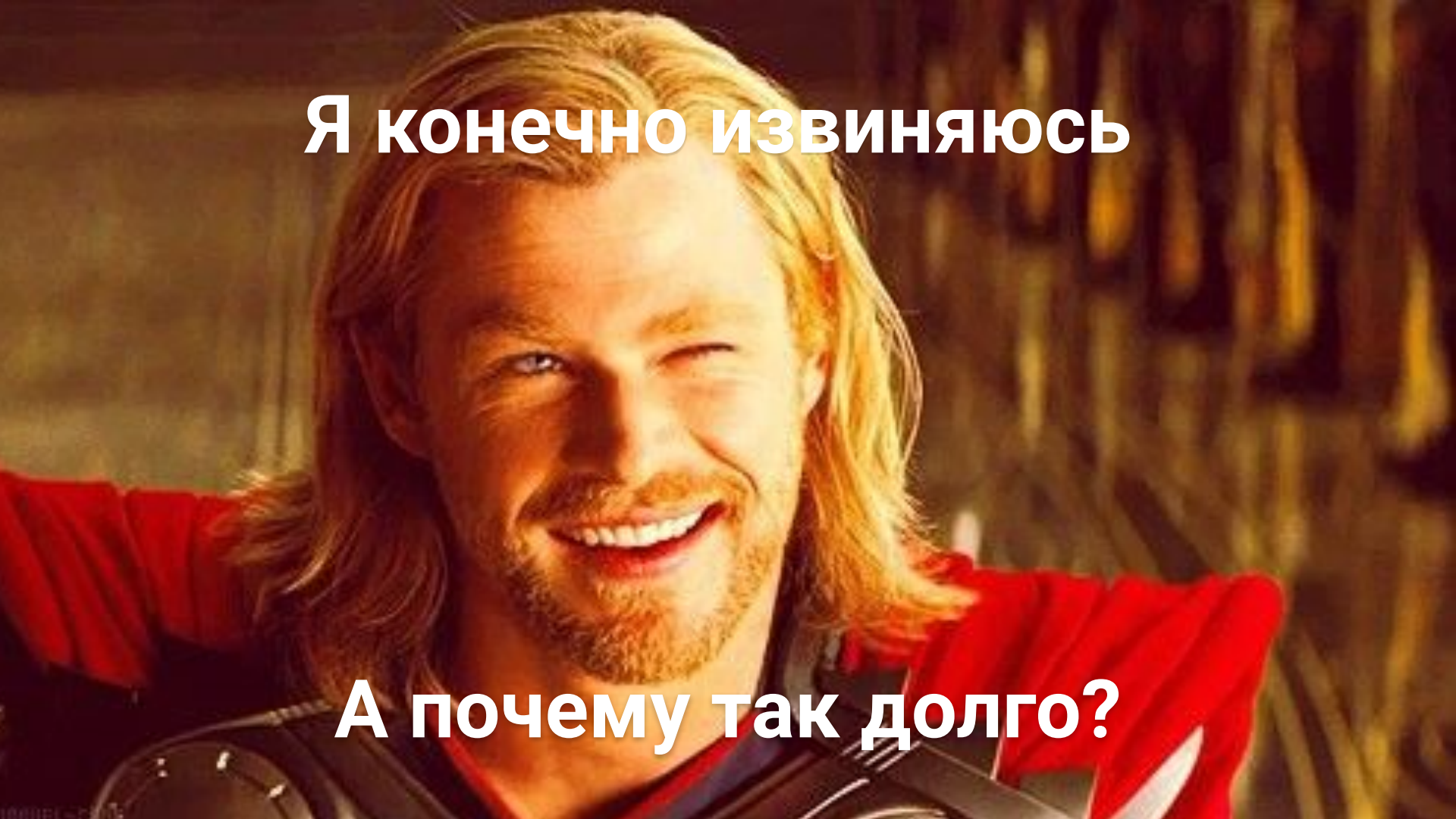
```
// Common Main
@Composable
internal expect fun platformScrollConfig(): ScrollConfig

// UIKit Main
@Composable
internal actual fun platformScrollConfig(): ScrollConfig = UiKitConfig
```

Compose -> Skiko

```
private object UiKitConfig : ScrollConfig {
    /*
     * There are no scroll events produced on iOS,
     * so in reality this function should not be ever called.
     * The implementation is copied from androidMain just for testing
purposes.
     */
    override fun Density.calculateMouseWheelScroll(event: PointerEvent,
bounds: IntSize): Offset =
        event.changes.fastFold(Offset.Zero)
          { acc, c -> acc + c.scrollDelta } * -64.dp.toPx()
}
```

Compose -> Skiko

```
private object UiKitConfig : ScrollConfig {
    /*
     * There are no scroll events produced on iOS,
     * so in reality this function should not be ever called.
     * The implementation is copied from androidMain just for testing
purposes.
     */
    override fun Density.calculateMouseWheelScroll(event: PointerEvent,
bounds: IntSize): Offset =
        event.changes.fastFold(Offset.Zero)
          { acc, c -> acc + c.scrollDelta } * -64.dp.toPx()
}
```


Я конечно извиняюсь
А почему так долго?

Трудности

> Работа с окнами (safe area, window api, sizing)

Трудности

> Работа с окнами (safe area, window api, sizing)

> Взаимодействие с клавиатурой

Трудности

> Работа с окнами (safe area, window api, sizing)

> Взаимодействие с клавиатурой

> Accessibility

Трудности

> Работа с окнами (safe area, window api, sizing)

> Взаимодействие с клавиатурой

> Accessibility

> Interopability с iOS

Трудности

> Работа с окнами (safe area, window api, sizing)

> Взаимодействие с клавиатурой

> Accessibility

> Interopability с iOS

> Platform features (Live Widgets, App Clips, etc)

Трудности

> Работа с окнами (safe area, window api, sizing)

> Взаимодействие с клавиатурой

> Accessibility

> Interopability с iOS

> Platform features (Live Widgets, App Clips, etc)

> Компиляторные проблемы

Как начать?

Как начать?

AlexGladkov/
JetpackComposeDemo

Jetpack Compose Features Demonstration

1 Contributor
2 Issues
156 Stars
14 Forks

https://github.com/AlexGladkov/JetpackComposeDemo

Как начать?

> Работа с ресурсами +/-

Как начать?

> Работа с ресурсами +/-

> Работа с навигацией ✔

Как начать?

> Работа с ресурсами +/-

> Работа с навигацией ✔

> Работа с viewmodel ✔

Как начать? Gradle

Вариант 1. Через iOS Main

Вариант 2. Через iOS Project

Как начать? Gradle

Вариант 1. Через iOS Main

Вариант 2. Через iOS Project

Как начать? Gradle

```
// Targets setup
iosX64("uikitX64")
iosArm64("uikitArm64")
```

Как начать? Gradle

```
// Targets setup
iosX64("uikitX64")
iosArm64("uikitArm64")


// Source Set setup
val iosMain by creating {
    dependsOn(commonMain)
    dependencies {
        ...
    }
}
```

Как начать? Gradle

```
// Targets setup
iosX64("uikitX64")
iosArm64("uikitArm64")


// Source Set setup
val iosMain by creating {
    dependsOn(commonMain)
    dependencies {
        ...
    }
}
```

```
// Source Set setup
val uikitMain by creating {
    dependsOn(iosMain)
}

val uikitX64Main by getting {
    dependsOn(uikitMain)
}

val uikitArm64Main by getting {
    dependsOn(uikitMain)
}
```

Как начать? iOS Project

```
fun MainViewController(): UIViewController =
    ComposeUIViewController {
        // Application content here
}
```

Как начать? iOS Project

Как начать? iOS Project

```
import UIKit
import shared

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?
    func application(_ application: UIApplication, didFinishLaunchingWithOptions
launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
        window = UIWindow(frame: UIScreen.main.bounds)
        let mainViewController = Main_iosKt.MainViewController()
        window?.rootViewController = mainViewController
        window?.makeKeyAndVisible()
        return true
    }
}
```

Как начать? iOS Project

```
import UIKit
import shared

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?
    func application(_ application: UIApplication, didFinishLaunchingWithOptions
launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
        window = UIWindow(frame: UIScreen.main.bounds)
        let mainViewController = Main_iosKt.MainViewController()
        window?.rootViewController = mainViewController
        window?.makeKeyAndVisible()
        return true
    }
}
```

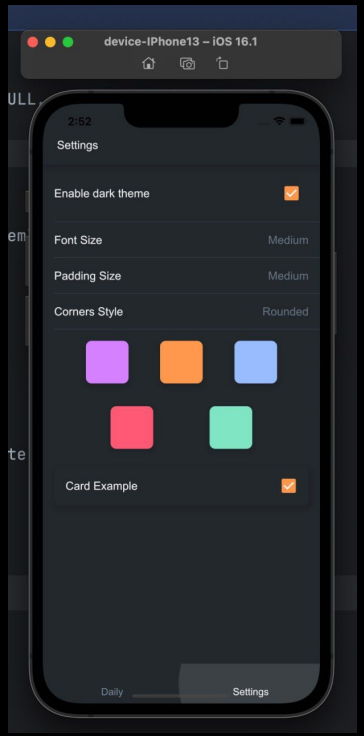
Как начать? iOS Project

```
import UIKit
import shared

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?
    func application(_ application: UIApplication, didFinishLaunchingWithOptions
launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
        window = UIWindow(frame: UIScreen.main.bounds)
        let mainViewController = Main_iosKt.MainViewController()
        window?.rootViewController = mainViewController
        window?.makeKeyAndVisible()
        return true
    }
}
```

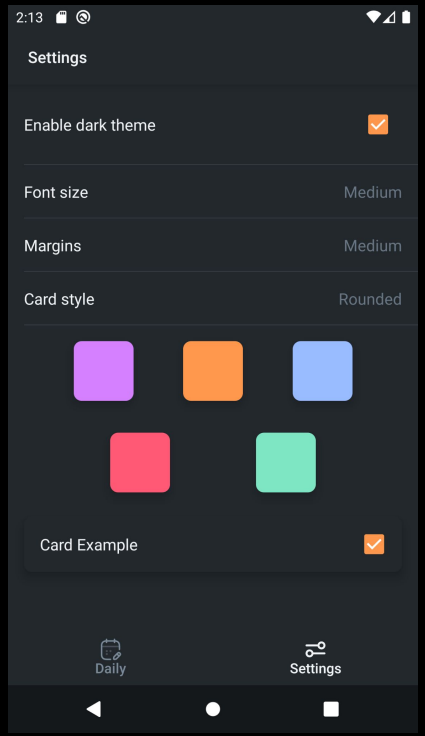
Как начать? iOS Project

```
import UIKit
import shared

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?
    func application(_ application: UIApplication, didFinishLaunchingWithOptions
launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
        window = UIWindow(frame: UIScreen.main.bounds)
        let mainViewController = Main_iosKt.MainViewController()
        window?.rootViewController = mainViewController
        window?.makeKeyAndVisible()
        return true
    }
}
```

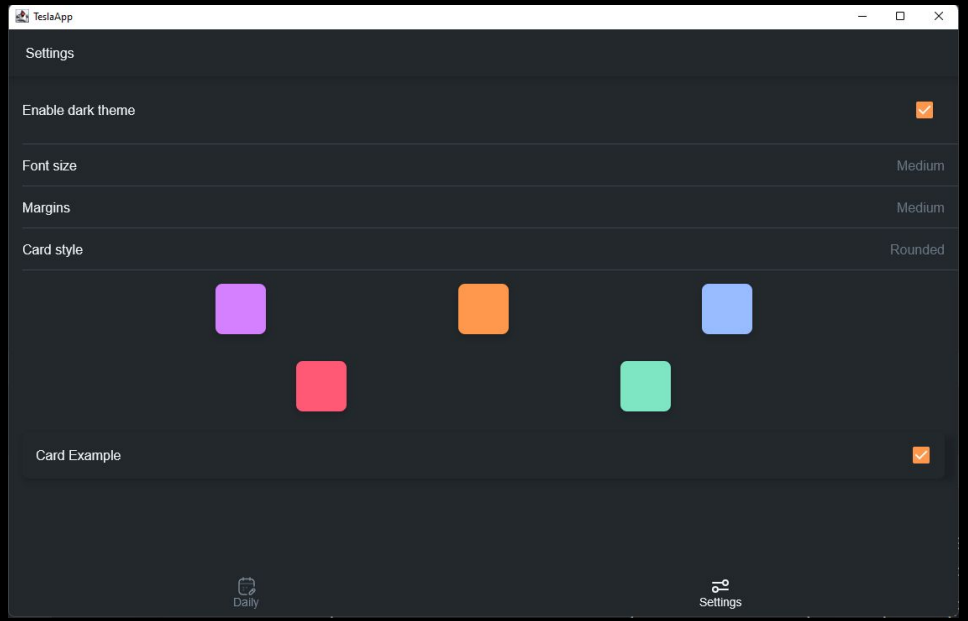
Compose Multiplatform / Demo



iOS



Android



Desktop

Компиляторы

Компиляторы

> Memory Allocation / Memory Management

Компиляторы

> Memory Allocation / Memory Management

> Compiler Errors

Компиляторы

> Memory Allocation / Memory Management

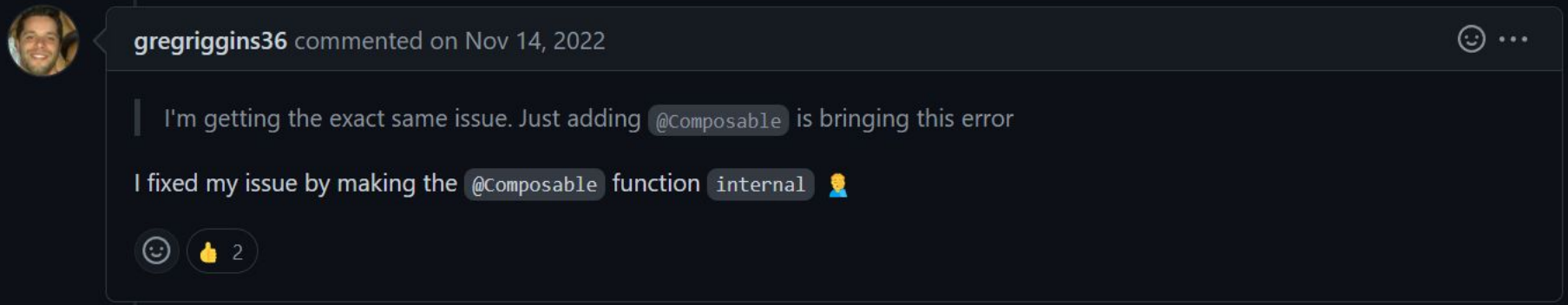
> Compiler Errors

```
> Task :shared:linkDebugFrameworkIosSimulatorArm64 FAILED
w: Mimalloc allocator isn't supported on target ios_simulator_arm64. Used standard mode.
e: Compilation failed: No file for com.github.jetbrains.rssreader/Foo|407190907863226929l[0] (@ com.github.jetbrains.rssr

 * Source files:
 * Compiler version info: Konan: 1.7.10 / Kotlin: 1.7.20
 * Output kind: FRAMEWORK

e: java.lang.IllegalStateException: No file for com.github.jetbrains.rssreader/Foo|4071909078632269291[0] (@ com.github.j
        at org.jetbrains.kotlin.backend.common.serialization.BasicIrModuleDeserializer.deserializeIrSymbol(BasicIrModuleD
```

Компиляторы

> Memory Allocation / Memory Management

> Compiler Errors



Компиляторы

> Memory Allocation / Memory Management

> Compiler Errors

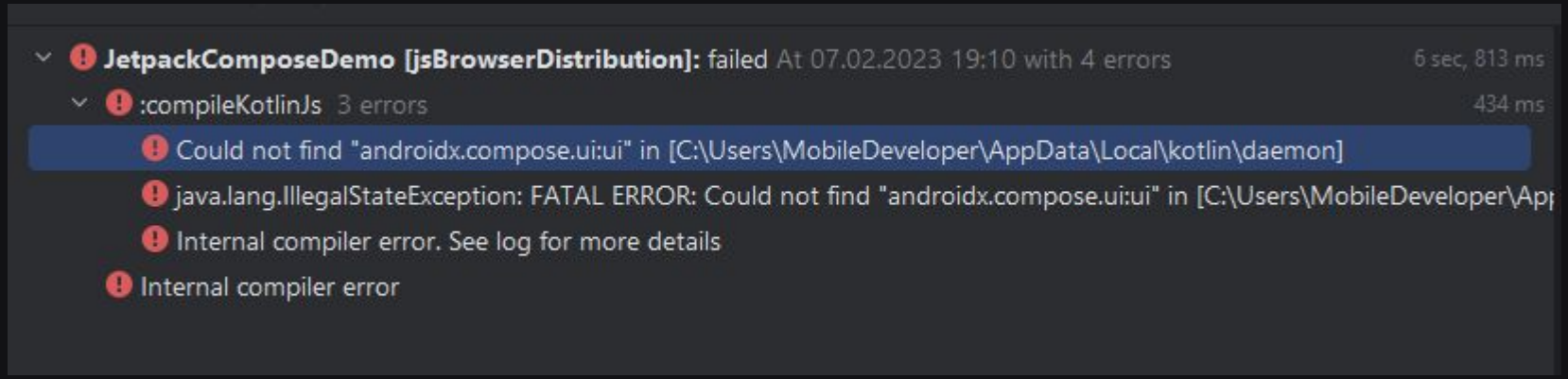
> Gradle Errors

Компиляторы

> Memory Allocation / Memory Management

> Compiler Errors

> Gradle Errors



Итоги

Итоги

> Compose for iOS находится в очень ранней стадии

Итоги

> Compose for iOS находится в очень ранней стадии

> Там нет нативного look and feel

Итоги

> Compose for iOS находится в очень ранней стадии

> Там нет нативного look and feel

> Он довольно быстро развивается

Итоги

> Compose for iOS находится в очень ранней стадии

> Там нет нативного look and feel

> Он довольно быстро развивается

> Большая часть времени уходит на фикс багов

Итоги

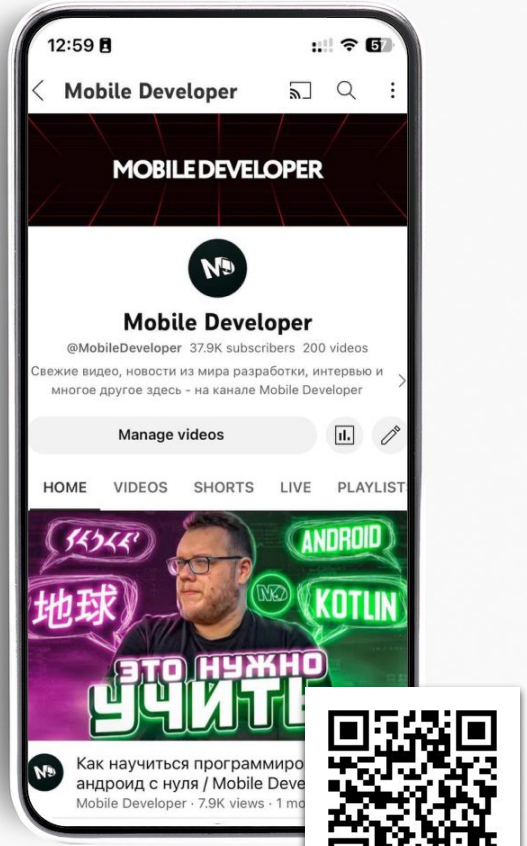
> Compose for iOS находится в очень ранней стадии

> Там нет нативного look and feel

> Он довольно быстро развивается

> Большая часть времени уходит на фикс багов

> Небольшие pet проекты можно релизить полностью на compose



Спасибо за внимание

email: mobiledevelopercourse@gmail.com

telegram: @mobiledevnews