ИТМО ЛИПТСОФТ

Портирование Android-приложений в экосистему HarmonyOS NEXT с формальными спецификациями

Иван Миленин
Мобильный разработчик

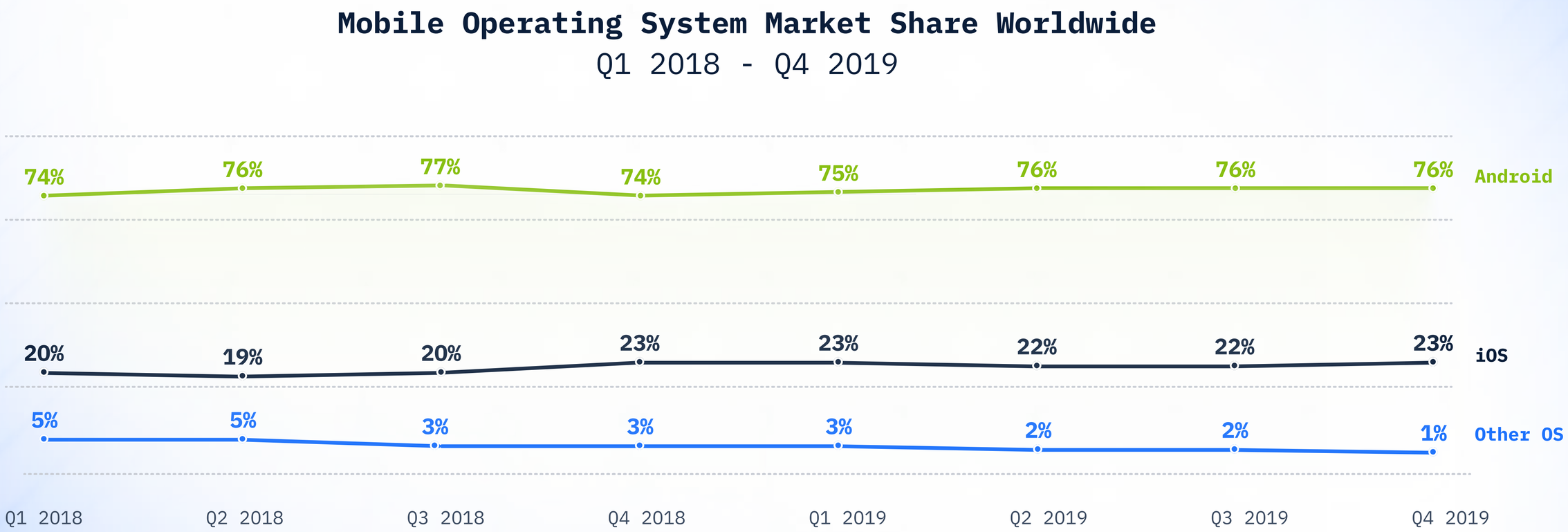
MILKAsuper
MILKA-TOP

План доклада

1. **Контекст и актуальность задачи**
2. Обзор существующих решений
3. Постановка задачи и общий подход
4. Трансляция бизнес-логики Kotlin в ArkTS
5. Библиотеки, формальные спецификации, SMT
6. UI: Jetpack Compose в ArkUI
7. Прототип и результаты

ИТМО ЛИПТСОФТ

Мобильные ОС 2018-2019



HarmonyOS 1.0 (First Gen.)

- Первый публичный релиз в 2019 году
- Использовалась лишь для TV и IoT
- Базировалась на **LiteOS**

Name	OS version	API version	Initial stable release date	Latest security patch version (release date)	AOSP version
HarmonyOS 1.0	1.0.0	5	August 9, 2019	1.0.1.66 SP3 (October 20, 2020)	Android 9 "Pie" (TV)

Unsupported

HarmonyOS 2/3/4 (Second Gen.)

- Первый публичный релиз в 2021 году
- ОС были доступно для использование на смартфонах
- Базировалась на **AOSP**
- Общее число доступных приложений – **3 500 000 +**

HMS (Huawei Mobile Services)

альтернатива GMS (Google Mobile Services), включающая в себя:

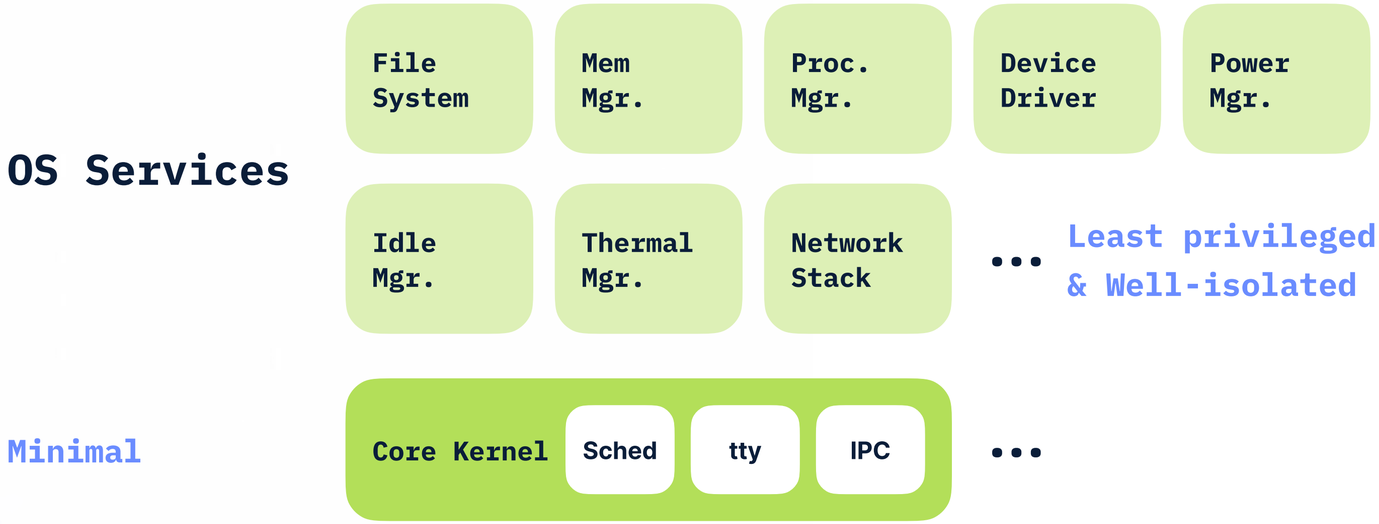
- Huawei ID
- Push Kit
- Analytics Kit
- Map Kit и т.д

```
push-notifications
  common
  scripts
  src
    google [main]
    huawei
    huaweiDebug
    huaweiRelease
    main
  .gitignore
  build.gradle.kts
```

HongMeng Kernel

микроядерная архитектура, разработанная с целью заменить AOSP и Linux-ядро

Представлено в августе 2023 года



Новые языки и фреймворки

ArkTS

язык программирований, основанный на синтаксисе TypeScript

Новые языки и фреймворки

ArkTS

язык программированый, основанный на синтаксисе TypeScript

ArkUI

декларативный UI фреймворк

Новые языки и фреймворки

ArkTS

язык программированый, основанный на синтаксисе TypeScript

ArkUI

декларативный UI фреймворк

DevEco

IDE для разработки приложений под HarmonyOS.

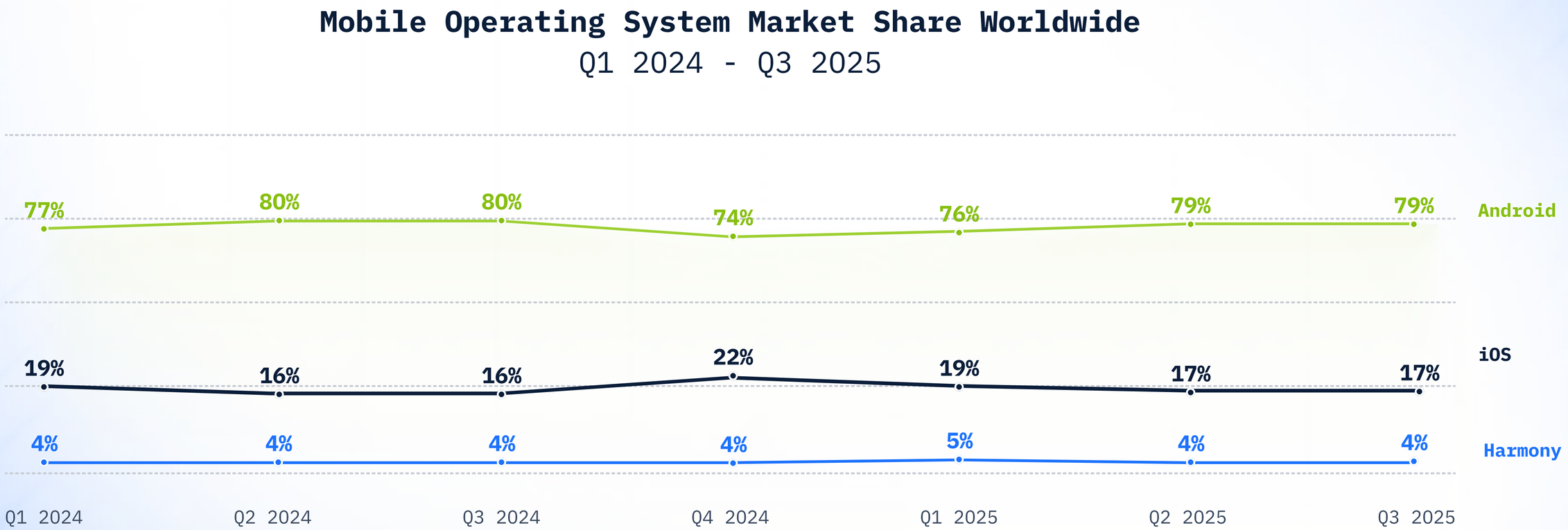
IntelliJ IDEA’s fork

HarmonyOS 5/6 (Third Gen.)

- Первый публичный релиз в 2024 году
- Базировалась на **HongMeng Kernel**
- Использует новые фреймворки и языки: **ArkTS, ArkUI, Canjie**
- Общее число доступных приложений – **20 000+**

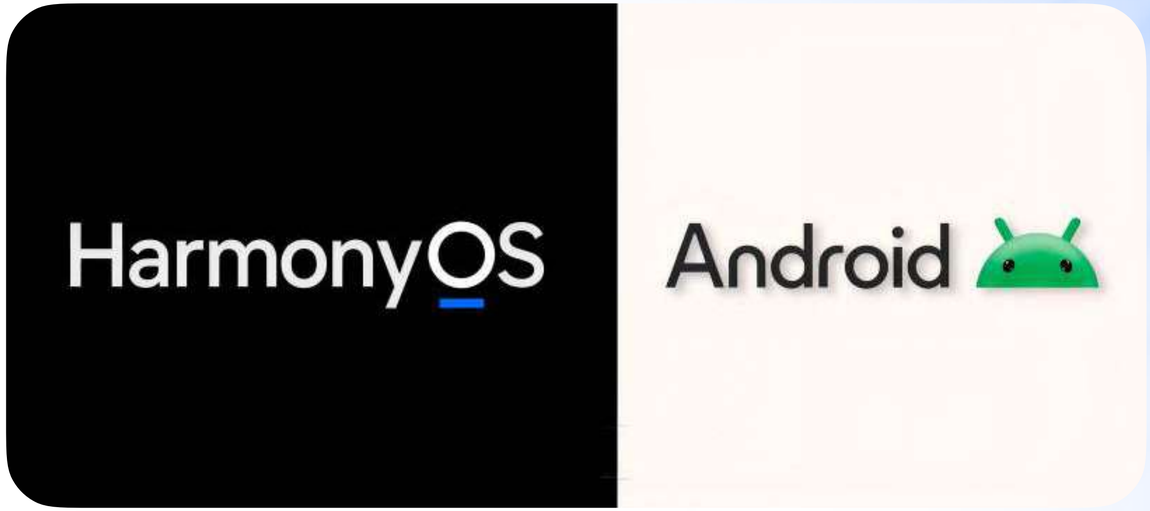
HarmonyOS NEXT

Мобильные ОС (2024-2025)



Актуальность

- Приложений мало
- Китайских туристов все больше и больше
- Ключевая задача – наполнение платформы HarmonyOS NEXT пользовательскими приложениями



План доклада

1. Контекст и актуальность задачи
2. **Обзор существующих решений**
3. Постановка задачи и общий подход
4. Трансляция бизнес-логики Kotlin в ArkTS
5. Библиотеки, формальные спецификации, SMT
6. UI: Jetpack Compose в ArkUI
7. Прототип и результаты

ИТМО ЛИПТСОФТ

Существующие подходы портирования

1. Разработка с нуля
2. Кроссплатформа
3. Виртуальное окружение
4. Виртуальное окружение
5. ML/LLM
6. Автоматическая трансляция кода

Существующие подходы портирования

Разработка с нуля

- “Полный контроль” при написании кода
- Существуют официальные инструменты

- Долго
- Нужны команда разработчиков под ArkTS

Существующие подходы портирования

Кроссплатформа

- 1 код - несколько платформ
- Есть официальная поддержка ArkUI-X

ARKUI-X

ovCompose

- Нет официальной поддержки популярных инструментов

Существующие подходы портирования

Виртуальное окружение

- Затраты на адаптацию кода сведены к минимуму

卓易通 卓易通

- Проблемы с производительностью
- Сложнее использовать нативное API

出境易

Существующие подходы портирования

ML/LLM

Прямой транспиляции Kotlin → ArkT **не существует**,
а ML-подходы дают лишь приблизительные
результаты без гарантий корректности

Gong, L., Wang, C., Cui, D., Huang, Y., Wei, M. (2024). UITrans: Seamless UI Translation from Android to HarmonyOS. Proceedings of the 16th International Conference on Internetware.

Liu, R., Lin, Y., Hu, Y., Zhang, Z., Gao, X. (2024). LLM-Based Java Concurrent Program to ArkTS Converter. 2024 39th IEEE/ACM International Conference on Automated Software Engineering (ASE), 2403-2406.

Существующие подходы портирования

- Сохранение семантики исходного кода
- “Проверяемость” результата портирования

+

- Отсутствие автоматизированного инструментария

−

План доклада

1 Контекст и актуальность задачи

2 Обзор существующих решений

3 Постановка задачи и общий подход

4 Трансляция бизнес-логики Kotlin в ArkTS

5 Библиотеки, формальные спецификации, SMT

6 UI: Jetpack Compose в ArkUI

7 Прототип и результаты

ИТМО ЛИПТСОФТ

В начале был Android...

Kotlin

Kotlin

```
fun main() {
    println("Hello, world!")
}
```

В начале был Android...

Jetpack Compose

Kotlin

```kotlin
fun main() {
    println("Hello, world!")
}
```

Jetpack Compose

```kotlin
@Composable
fun HelloWorld() {
    Text(text = "Hello, world!")
}
```

А потом появился HarmonyOS...

ArkTS

ArkTS

```
export function main(): void {
    console.log("Hello, world!")
}
```

А потом появился HarmonyOS...

ArkUI

ArkTS

```
export function main(): void {
    console.log("Hello, world!")
}
```

ArkUI

```
@ComponentV2
struct HelloWorld {
    build() {
        Text("Hello, world!")
    }
}
```

HarmonyOS NEXT

Сравнительный анализ архитектур Kotlin/ArkTS

Kotlin	ArkTS
`val counter: Int? = 0`	`let counter: number | null = 0;`

Сравнительный анализ архитектур Kotlin/ArkTS

Kotlin	ArkTS
`val counter: Int? = 0`	`let counter: number | null = 0;`

`fun increment(x: Int): Int {`
`  return x + 1`
`}`
	`export function increment(x: number): number {`
	`  return x + 1;`
	`}`

Сравнительный анализ архитектур Kotlin/ArkTS

Kotlin	ArkTS
`val counter: Int? = 0`	`let counter: number | null = 0;`

`fun increment(x: Int): Int {`
`  return x + 1`
`}`
	`export function increment(x: number): number {`
	`  return x + 1;`
	`}`

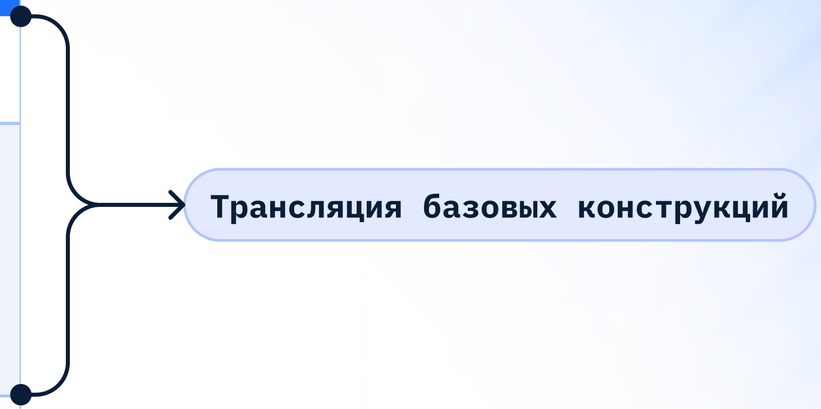
`if (x == "LIPT") {`
`  println("Cool!")`
`} else if (x == "IPKN") {`
`  println("Wow!")`
`} else {`
`  println("Awesome :)")`
`}`
	`if (x == "LIPT") {`
	`  console.log("Cool!")`
	`} else if (x == "IPKN") {`
	`  console.log("Wow!")`
	`} else {`
	`  console.log("Awesome :)")`
	`}`

Сравнительный анализ архитектур Kotlin/ArkTS

Kotlin	ArkTS
`val counter: Int? = 0`	`let counter: number | null = 0;`

`fun increment(x: Int): Int {`
`  return x + 1`
`}`
	`export function increment(x: number): number {`
	`  return x + 1;`
	`}`

`if (x == "LIPT") {`
`  println("Cool!")`
`} else if (x == "IPKN") {`
`  println("Wow!")`
`} else {`
`  println("Awesome :)")`
`}`
	`if (x == "LIPT") {`
	`  console.log("Cool!")`
	`} else if (x == "IPKN") {`
	`  console.log("Wow!")`
	`} else {`
	`  console.log("Awesome :)")`
	`}`



Сравнительный анализ архитектур Kotlin/ArkTS

Kotlin	ArkTS
`val counter: Int? = 0`	`let counter: number | null = 0;`

`fun increment(x: Int): Int {`
`  return x + 1`
`}`
	`export function increment(x: number): number {`
	`  return x + 1;`
	`}`

`if (x == "LIPT") {`
`  println("Cool!")`
`} else if (x == "IPKN") {`
`  println("Wow!")`
`} else {`
`  println("Awesome :)")`
`}`
	`if (x == "LIPT") {`
	`  console.log("Cool!")`
	`} else if (x == "IPKN") {`
	`  console.log("Wow!")`
	`} else {`
	`  console.log("Awesome :)")`
	`}`

Трансляция базовых конструкций

Трансляция вызовов библиотек

Сравнительный анализ архитектур Kotlin/ArkTS

Jetpack Compose	ArkUI

`@Composable`
`fun AnyComponent() {`
`  ...`
`}`
	`@ComponentV2`
	`struct AnyComponent {`
	`  build() {`
	`...`
	`  }`
	`}`

Сравнительный анализ архитектур Kotlin/ArkTS

Jetpack Compose	ArkUI

`@Composable`
`fun AnyComponent() {`
`  ...`
`}`
	`@ComponentV2`
	`struct AnyComponent {`
	`  build() {`
	`...`
	`  }`
	`}`

`Button(onClick = { counter++ }) {`
`  Text("Add")`
`}`
	`Button("Add")`
	`  .onClick(() => { this.counter++; });`

Сравнительный анализ архитектур Kotlin/ArkTS

Jetpack Compose	ArkUI

`@Composable`
`fun AnyComponent() {`
`  ...`
`}`
	`@ComponentV2`
	`struct AnyComponent {`
	`  build() {`
	`...`
	`  }`
	`}`

`Button(onClick = { counter++ }) {`
`  Text("Add")`
`}`
	`Button("Add")`
	`  .onClick(() => { this.counter++; });`

`var counter by remember { mutableStateOf(0) }`	`@State counter: number = 0;`

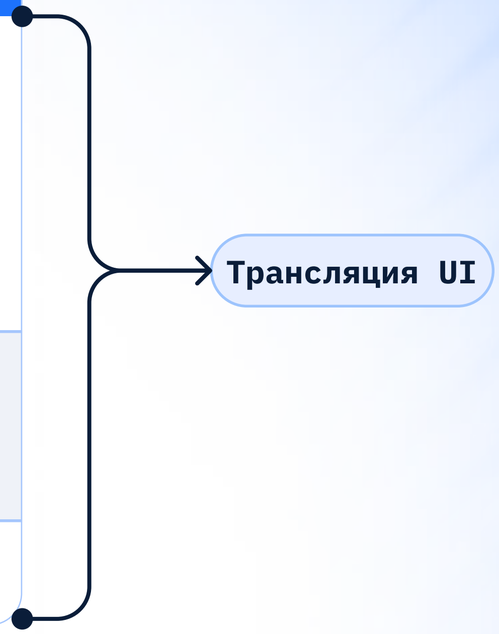
Сравнительный анализ архитектур Kotlin/ArkTS

Jetpack Compose	ArkUI

`@Composable`
`fun AnyComponent() {`
`  ...`
`}`
	`@ComponentV2`
	`struct AnyComponent {`
	`  build() {`
	`...`
	`  }`
	`}`

`Button(onClick = { counter++ }) {`
`  Text("Add")`
`}`
	`Button("Add")`
	`  .onClick(() => { this.counter++; });`

`var counter by remember { mutableStateOf(0) }`	`@State counter: number = 0;`



Предлагаемый подход

1. Трансляция базовых конструкций (Kotlin → ArkTS)
2. Автоматическая трансляция кода
3. Трансляция UI

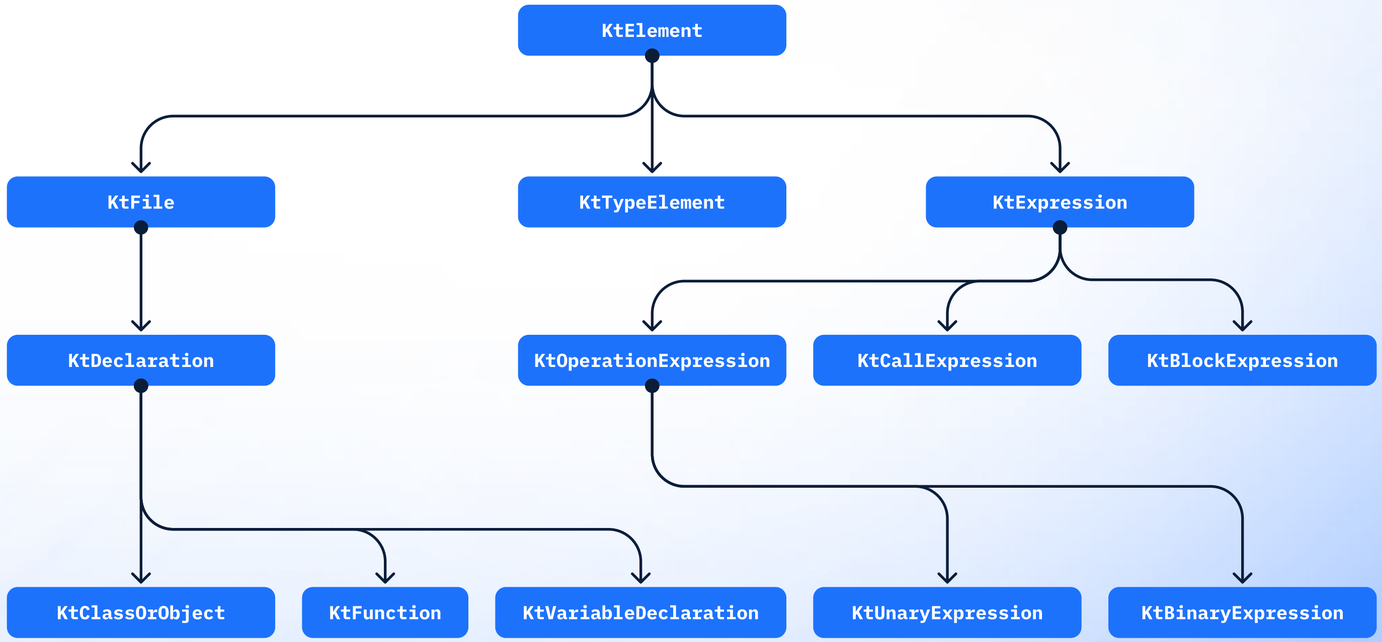
План доклада

1. Контекст и актуальность задачи
2. Обзор существующих решений
3. Постановка задачи и общий подход
4. **Трансляция бизнес-логики Kotlin в ArkTS**
5. Библиотеки, формальные спецификации, SMT
6. UI: Jetpack Compose в ArkUI
7. Прототип и результаты

ИТМО ЛИПТСОФТ

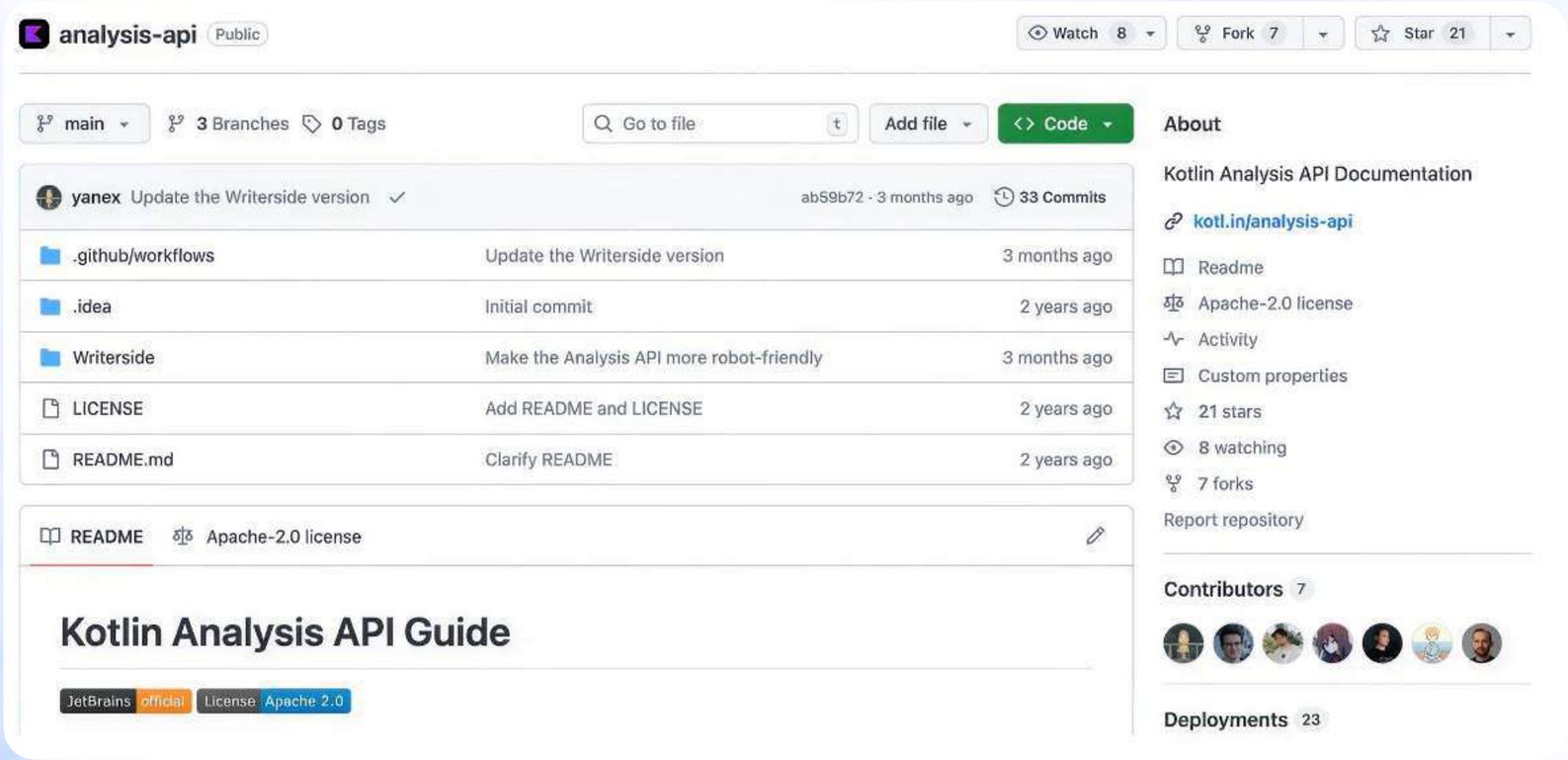
Kotlin PSI

Program Structure Interface (PSI) – это слой, отвечающий за анализ файлов и создание синтаксической и семантической модели кода, которая обеспечивает многие функции платформы



“Синтаксическое дерево, где **каждому элементу кода соответствует вершина графа”**

Трансляция исходного кода. Kotlin → PSI



Для анализа исходного кода используется **PSI (Program Structure Interface)**, формируемый через **Kotlin Analysis API**.

Трансляция исходного кода. Kotlin → PSI

Обход PSI выполняется от корневого узла **KtFile**,
далее обрабатываются классы, функции, выражения и декларации

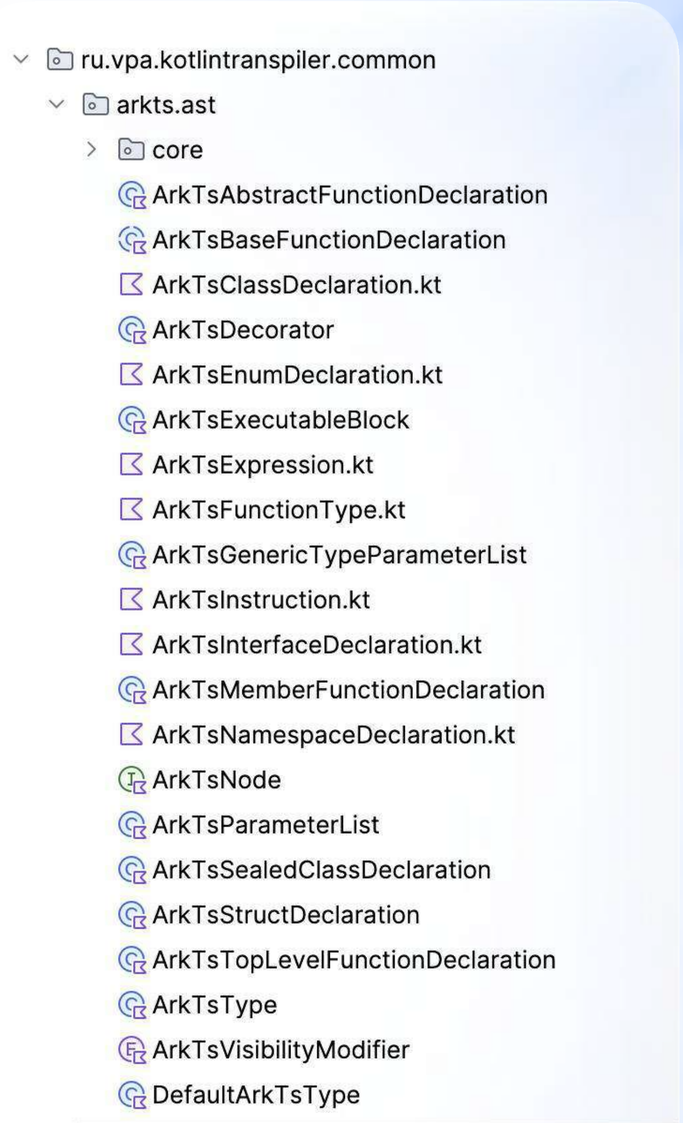
```
fun main() {
    val x = 10
    println(x)
}
```

```
PSI Tree

KtFile
  KtNamedFunction (fun main() {...})
    KtModifierList
    KtIdentifier (main)
    KtParameterList (())
    KtBlockExpression ({...})
      KtProperty (val x = 10)
        KtModifierList
        KtIdentifier (x)
        KtTypeReference (null)
        KtEquals (=)
        KtConstantExpression (10)
      KtCallExpression (println(x))
```

Трансляция исходного кода. PSI → ArkTS IR

На основе PSI формируется **ArkTS IR** — внутреннее представление, отражающее структуру будущей ArkTS-программы

```
/**
 * Base contract for every ArkTS AST element.
 * Each node should be able to render itself back to
ArkTS source code.
 */
interface ArkTSNode {

    /**
     * Converts this AST node back to ArkTS code
representation.
     */
    fun render(): String
}
```



Трансляция исходного кода. PSI → ArkTS IR

ArkTS IR **не копирует** синтаксис Kotlin,
а моделирует программу в терминах **конструкций ArkTS** с учётом различий языков

```kotlin
/**
 * Represents an identifier usage.
 *
 * Example: val a = b + 1;
 *          b - IdentifierExpression
 */
data class ArkTsIdentifierExpression(
    val name: String,
) : ArkTsExpression {

    override fun render(): String = name
}
```

```kotlin
/**
 * Represents an ArkTS expression wrapped in parentheses.
 *
 * Example: (1+2), where are all this expression is Parenthesized
 */
data class ArkTsParenthesizedExpression(
    val expression: ArkTsExpression,
) : ArkTsExpression {

    override fun render(): String {
        return buildString {
            append(ArkTsTokens.LEFT_BRACKET)
            append(expression.render())
            append(ArkTsTokens.RIGHT_BRACKET)
        }
    }
}
```

Трансляция исходного кода. PSI → ArkTS IR

ArkTS IR служит основой
для генерации итогового ArkTS-кода

```
ArkTsFile
  ArkTsTopLevelFunctionDeclaration
    name: main
    visibility: export
    returnType: ArkTsType(void)
    ArkTsParameterList
      (no parameters)
    ArkTsExecutableBlock
      ArkTsVariableDeclarationInstruction
        name: x
        isMutable: false
        initializer:
          ArkTsLiteralExpression
            value: "10"
      ArkTsExpressionInstruction
        ArkTsCallExpression
          callee:
            ArkTsIdentifierExpression
              name: println
          arguments:
            ArkTsIdentifierExpression
              name: x
```

Трансляция исходного кода. Data-классы

Kotlin

```kotlin
data class Foo(
    val a: Int,
    val b: String,
)
```

ArkTS

```typescript
export class Foo {
    public readonly a: number;

    public readonly b: string;

    constructor(a: number, b: string) {
        this.a = a
        this.b = b
    }

    public equals(other: object | null): boolean {
        return other === this || other instanceof Foo && this.a === other.a && this.b === other.b
    }

    public toString(): string {
        return "Foo(" + "a=" + this.a + ", " + "b=" + this.b + ")"
    }

    public copy(a: number = this.a, b: string = this.b): Foo {
        return new Foo(a, b)
    }
}
```

Трансляция исходного кода. Data-классы

Kotlin (.class)

```
public final class Foo {
    private final int a;
    @NotNull
    private final String b;

    public Foo(int a, @NotNull String b) {
        Intrinsics.checkNotNullParameter(b, "b");
        super();
        this.a = a;
        this.b = b;
    }

    public final int getA() { return this.a; }

    @NotNull
    public final String getB() { return this.b; }

    public final int component1() { return this.a; }

    @NotNull
    public final String component2() { return this.b; }
```

ArkTS

```
export class Foo {
    public readonly a: number;

    public readonly b: string;

    constructor(a: number, b: string) {
        this.a = a
        this.b = b
    }
}
```

Трансляция исходного кода. Data-классы

Kotlin (.class)

```
public boolean equals(@Nullable Object other) {
    if (this == other) {
        return true;
    } else if (!(other instanceof Foo)) {
        return false;
    } else {
        Foo var2 = (Foo)other;
        if (this.a != var2.a) {
            return false;
        } else {
            return Intrinsics.areEqual(this.b, var2.b);
        }
    }
}
```

ArkTS

```
public equals(other: object | null): boolean {
    return other === this
        || other instanceof Foo
        && this.a === other.a
        && this.b === other.b
}
```

Трансляция исходного кода. Data-классы

Kotlin (.class)	ArkTS

`@NotNull`
`public String toString() {`
`  return "Foo(a=" + this.a + ", b=" + this.b + ")";`
`}`
	`public toString(): string {`
	`  return "Foo(" + "a=" + this.a + ", " + "b=" + this.b + ")"`
	`}`

Трансляция исходного кода. Data-классы

Kotlin (.class)

```
@NotNull
public final Foo copy(int a, @NotNull String b) {
    Intrinsics.checkNotNullParameter(b, "b");
    return new Foo(a, b);
}

// $FF: synthetic method
public static Foo copy$default(Foo var0, int var1, String
var2, int var3, Object var4) {
    if ((var3 & 1) != 0) {
        var1 = var0.a;
    }

    if ((var3 & 2) != 0) {
        var2 = var0.b;
    }

    return var0.copy(var1, var2);
}
```

ArkTS

```
public copy(a: number = this.a, b: string = this.b): Foo {
    return new Foo(a, b)
}
```

Трансляция исходного кода. Sealed-классы

Kotlin

```kotlin
sealed class Result {

    data class Success(
        val data: String
    ): Result()

    object Failure : Result()

}
```

ArkTS

```typescript
export abstract class Result {
}

export namespace Result {
    export class Success extends Result {
        public readonly data: string;

        constructor(data: string) {
            super()
            this.data = data
        }
    }

    export class FailureClass extends Result {
        constructor() {
            super()
        }
    }
}

export const Failure: Result = new FailureClass();
```

Трансляция исходного кода. Лямбды

Kotlin

```
fun createLambda(): ((String) -> String) -> ((Int) ->
Int) -> (Int) -> String {
    return { stringTransformer ->
        { intTransformer ->
            { value: Int ->

                val transformedInt =
intTransformer(value)
                val base = "value=$value,
transformed=$transformedInt"
                stringTransformer(base)

            }
        }
    }
}
```

ArkTS

```
export function createLambda(): (p0: (p0: string) =>
string) => (p0: (p0: number) => number) => (p0: number)
=> string {
    return (stringTransformer) => {
        return (intTransformer) => {
            return (value) => {
                const transformedInt =
intTransformer(value);
                const base = `value=${value},
transformed=${transformedInt}`;
                return stringTransformer(base)
            }
        }
    }
}
```

Трансляция исходного кода. Форматированные строки

Kotlin	ArkTS
`val base = "value=$value, transformed=$transformedInt"`	`const base = \`value=${value}, transformed=${transformedInt}\`;`
`val msg = "Sum=${a + b}"`	`const msg = \`Sum: ${a + b};`

План доклада

1. Контекст и актуальность задачи
2. Обзор существующих решений
3. Постановка задачи и общий подход
4. **Трансляция бизнес-логики Kotlin в ArkTS**
5. **Библиотеки, формальные спецификации, SMT**
6. UI: Jetpack Compose в ArkUI
7. Прототип и результаты

ИТМО ЛИПТСОФТ

Трансляция вызовов библиотек

Kotlin

```kotlin
fun maskCardNumber(number: String): String {
    return number.dropLast(4) + "****"
}
```

ArkTS

```arkts
export function maskCardNumber(number: string): string {
 return number.substring(0, number.length - 4) + "****"
}
```

Трансляция вызовов библиотек

Kotlin

```kotlin
fun maskCardNumber(number: String): String {
    return number.dropLast(4) + "****"
}
```

ArkTS

```arkts
export function maskCardNumber(number: string): string {
  return number.substring(0, number.length - 4) + "****"
}
```

Трансляция вызовов библиотек

Kotlin

```kotlin
fun maskCardNumber(number: String): String {
    return number.dropLast(4) + "****"
}
```

ArkTS

```arkts
export function maskCardNumber(number: string): string {
 return number.dropLast(4) + "****"
}
```

Property 'dropLast' does not exist on type 'string'. <ArkTSCheck>

any

entry

Трансляция вызовов библиотек

- В коде Android-приложений вызываются функции и классы из **стандартных библиотек Kotlin** и Android SDK, **которые не имеют прямых аналогов** в HarmonyOS NEXT.

- **Невозможно выполнить** простую **синтаксическую замену** вызовов — требуется определить соответствие функций двух платформ.

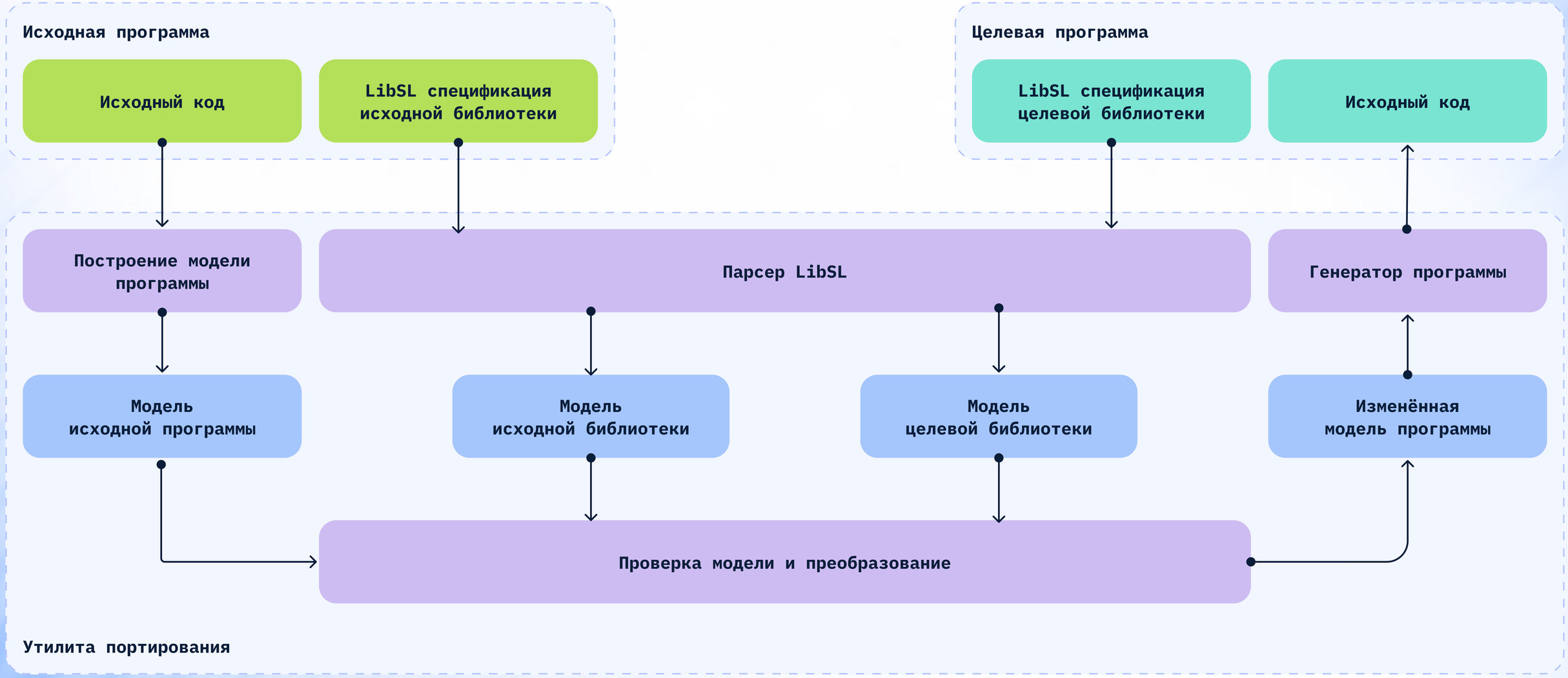
Формальные спецификации библиотек. LibSL

```
libsl "1.0.0";
library `KotlinText`
    version "1.0.0";

typealias bool = bool;
typealias string = string;
typealias int = int32;
typealias double = float64;

define action STRING_LENGTH(value: string): int;
define action STRING_SUBSTRING(value: string, start: int, end: int): string;

type kotlin.text { }

automaton kotlin.text: kotlin.text {

    fun dropLast(receiver: string, n: int): string {
        var length: int = action STRING_LENGTH(receiver);
        var end: int = length - n;
        result = action STRING_SUBSTRING(receiver, 0, end);
    }

}
```

Спецификация видимого поведения функции является частичной спецификацией поведения функции и отражает основное предназначение функции, видимое извне.

LibSL (Library Specification Language) — используется для декларативного **описания поведения библиотек**

Идея подхода



Идея подхода

Нужно **привести dropLast**
к удобному формату и **выразить**
через **String.length**
и **String.substring**

```
fun dropLast(receiver: string, n: int): string {
    var length: int = action STRING_LENGTH(receiver);
    var end: int = length - n;
    result = action STRING_SUBSTRING(receiver, 0, end);
}
```

```
@Extension
automaton String.substring: String {

    fun call(receiver: string, start: int, end: int): string {
        result = action STRING_SUBSTRING(receiver, start, end);
    }
}

@Extension
@Property
automaton String.length: String {

    fun get(receiver: string): int {
        result = action STRING_LENGTH(receiver);
    }
}
```

SSA form

Static Single Assignment form — это представление программы, где **каждая переменная присваивается ровно один раз.**

```
fun foo(): Int {
  x = 1
  y = x + 2
  x = x - 4
  z = x - y
  return z
}
```

SSA →

$x_0 = 1$

↓

$y_0 = x_0 + 2$

↓

$x_1 = x_0 - 4$

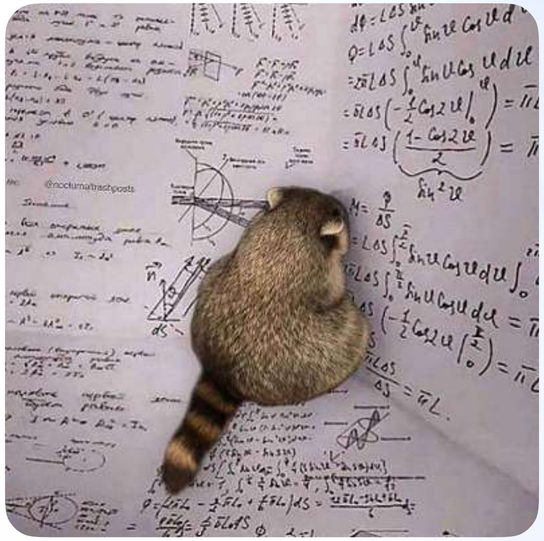
↓

$z_0 = x_1 - y_0$

↓

$r_0 = z_0$

x записывается дважды → x_0 и x_1 — разные имена

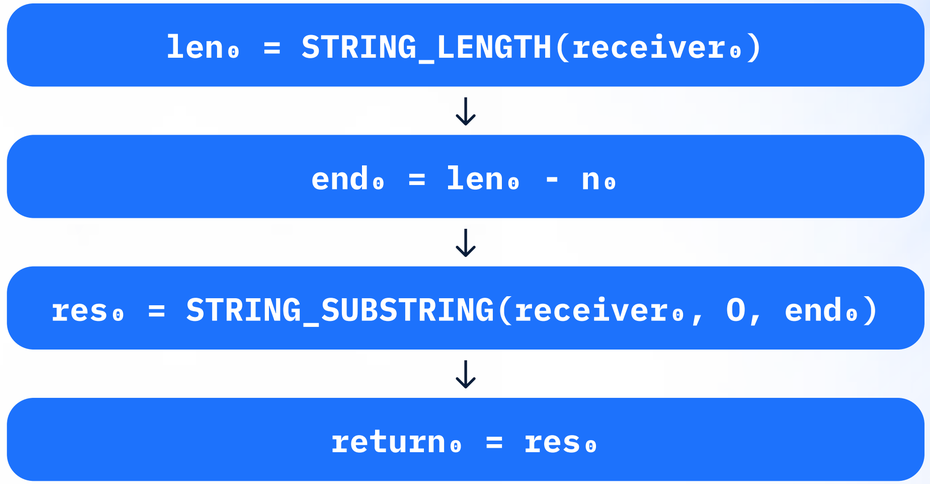


SSA form

Static Single Assignment form — это представление программы, где **каждая переменная присваивается ровно один раз.**

```
fun dropLast(receiver: string, n: int): string {
    var length: int = action STRING_LENGTH(receiver);
    var end: int = length - n;
    result = action STRING_SUBSTRING(receiver, 0, end);
}
```

SSA →



Трасса исполнения

Трасса – это упорядоченная последовательность шагов исполнения функции, зависящая от выполнения условных операторов.

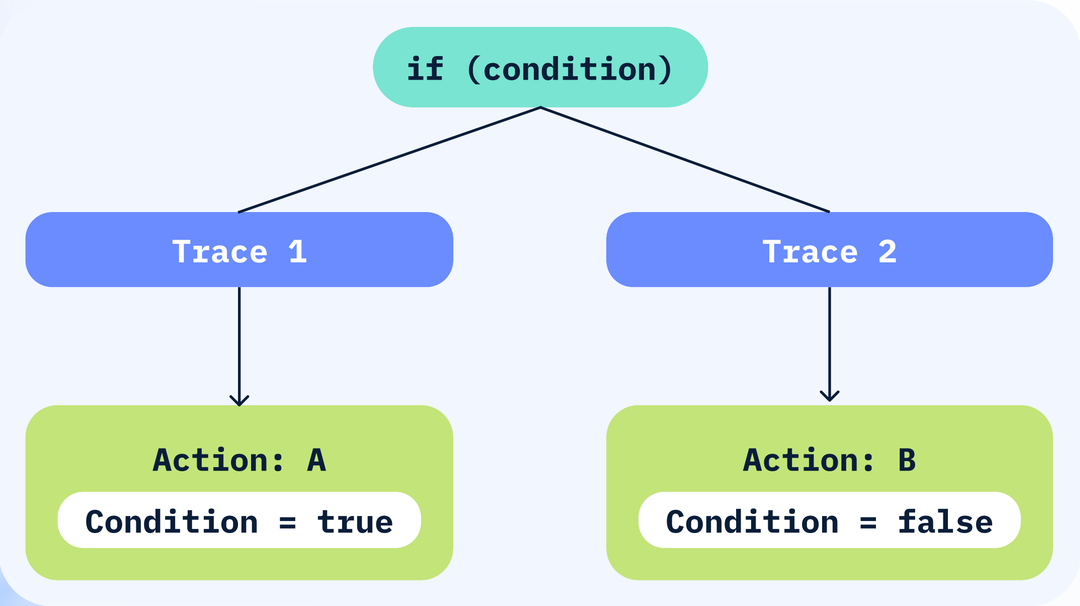
Последовательность Действий **(Action)**, Условий **(Condition)** и Результатов **(Result)**

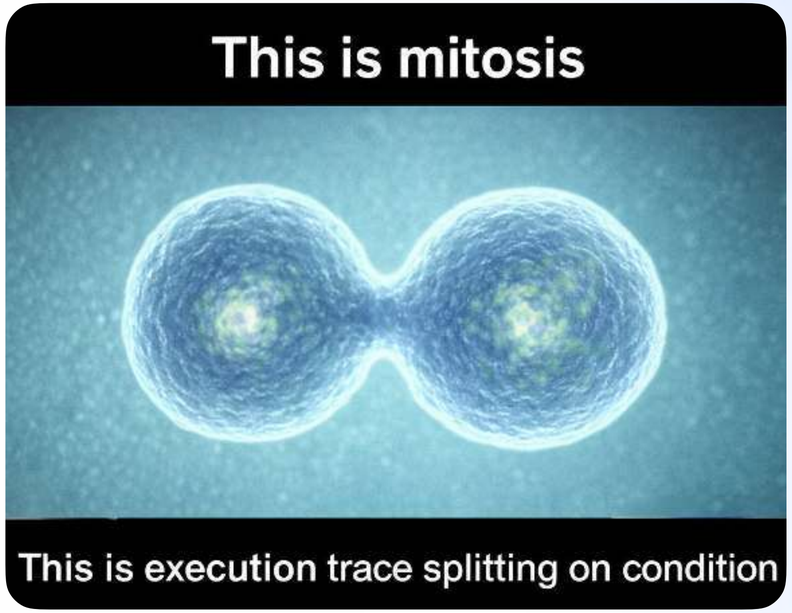
```
data class Trace(
    val function: LibFunction,
    val args: List<DeclStat>,
    val body: List<Step>
)

sealed class Step
data class Action(val step: ActionCallStat) : Step()
data class Result(val step: ResultStat) : Step()
data class Condition(val step: LExpr) : Step()
```

Структуры данных для описания трассы в коде

Трасса исполнения

Если есть условие **(if)** – **трасса раздваивается**. В одну трассу добавляется текущее её условие, в другую – его отрицание.





Пример получения трассы исполнения

Код

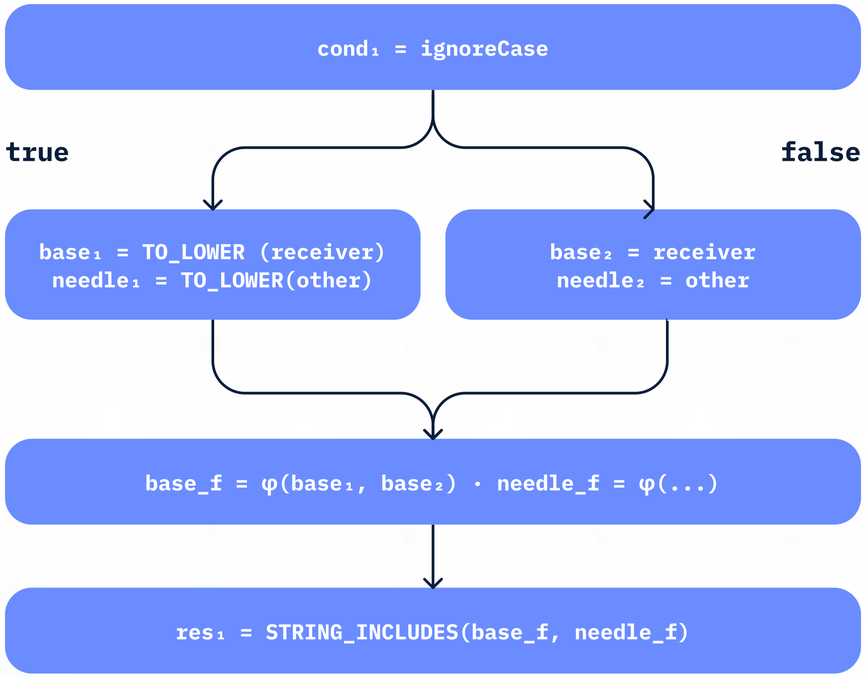
```
fun contains(receiver: string, other: string,
             ignoreCase: bool): bool {

    if (ignoreCase) {
        base = action STRING_TO_LOWER(receiver);
        needle = action STRING_TO_LOWER(other);
    } else {
        base = receiver;
        needle = other;
    }

    result = action STRING_INCLUDES(base, needle);
}
```

Пример получения трассы исполнения

Код

```
fun contains(receiver: string, other: string,
             ignoreCase: bool): bool {

    if (ignoreCase) {
        base = action STRING_TO_LOWER(receiver);
        needle = action STRING_TO_LOWER(other);
    } else {
        base = receiver;
        needle = other;
    }

    result = action STRING_INCLUDES(base, needle);
}
```

SSA

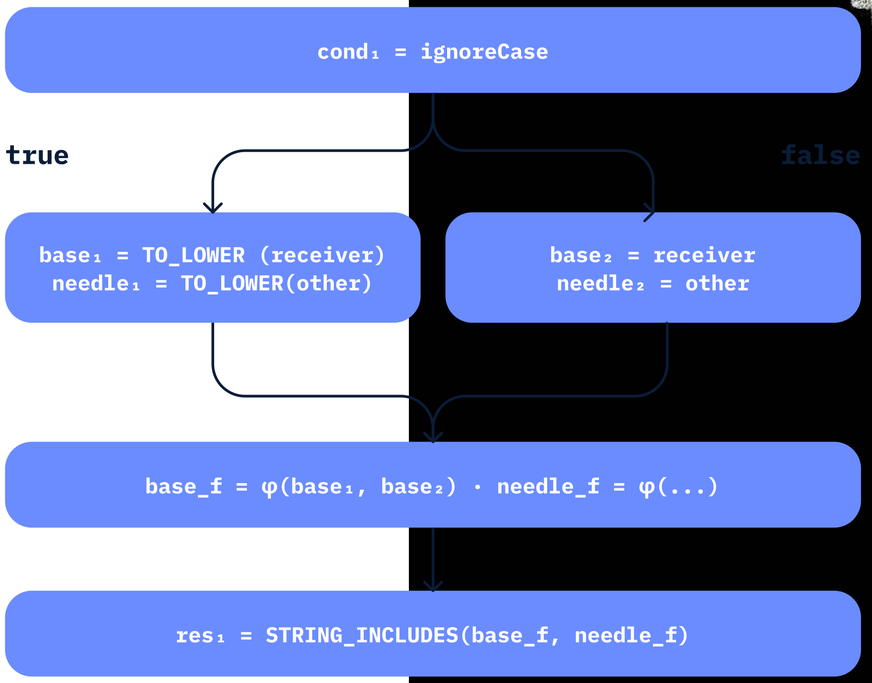


Пример получения трассы исполнения

Код

```
fun contains(receiver: string, other: string,
             ignoreCase: bool): bool {

    if (ignoreCase) {
        base = action STRING_TO_LOWER(receiver);
        needle = action STRING_TO_LOWER(other);
    } else {
        base = receiver;
        needle = other;
    }

    result = action STRING_INCLUDES(base, needle);
}
```

SSA



Трассы

Trace 1 · ignoreCase

```
Action:    TO_LOWER(receiver)
Action:    TO_LOWER(other)
Condition:  ignoreCase
Action:    INCLUDES(base, needle)
Result:    res₁
```

Trace 2 · !ignoreCase

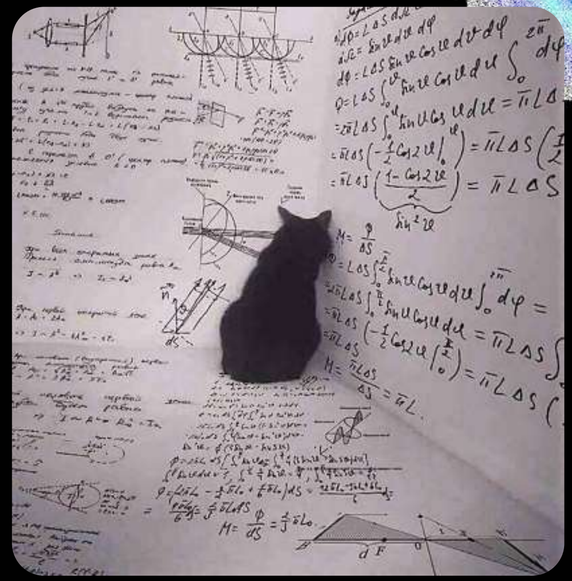
```
Condition: !ignoreCase
Action:    INCLUDES(receiver, other)
Result:    res₁
```

SAT

Boolean Satisfiability Problem – задача булевой выполнимости: нужно определить, существует ли набор значений True/False для переменных, при котором логическая формула (в булевой логике) становится истинной.

$$(x \vee y) \wedge (\neg x \vee z)$$

Есть ли такой набор значений x, y и z, при котором логическое выражение верно (true)



SAT

$$F = (x \vee y) \wedge (\neg x \vee z)$$

Итерация	x	y	z	(x ∨ y)	(¬x ∨ z)	F	Комментарий
0	?	?	?	?	?	?	Начальное состояние
1	false	?	?	true	?	?	Предположение: x = false
2	false	false	?	true	true	false	Предположение: y = false → F = false
3	false	true	true	true	true	true	Unit-propagation: y = true
4	false	true	true	true	true	true	Устанавливаем z, SAT – решение найдено

SAT

$$F = (x \vee y) \wedge (\neg x \vee z)$$

Итерация	x	y	z	(x ∨ y)	(¬x ∨ z)	F	Комментарий
0	?	?	?	?	?	?	Начальное состояние
1	false	?	?	true	?	?	Предположение: x = false
2	false	false	?	true	true	false	Предположение: y = false → F = false
3	false	true	true	true	true	true	Unit-propagation: y = true
4	false	true	true	true	true	true	Устанавливаем z, SAT – решение найдено

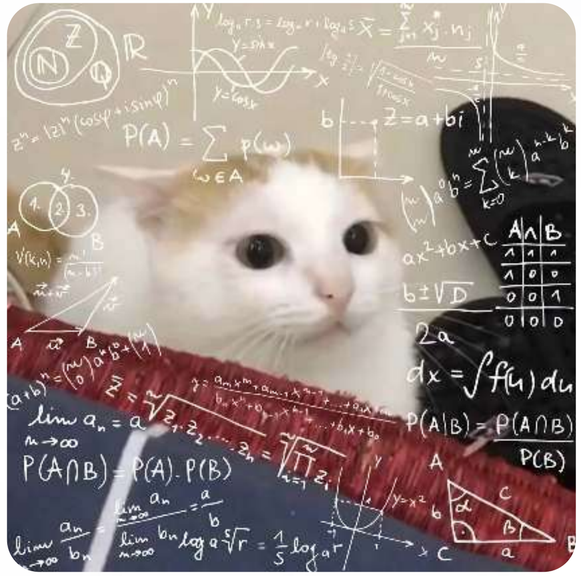
SAT → SMT

Satisfiability Modulo Theories — проверка выполнимости логической формулы с учётом “теорий” (арифметика, массивы, строки, bit-vectors и т.д.).

~~(x ∨ y) ∧ (¬x ∨ z)~~

(x > 0) ∧ (x + y = 10) ∧ (y ≥ 3)

Есть ли такой набор значений x, y и z, при котором логическое выражение верно (true)



SAT → SMT

$$F = (x > 0) \land (x + y = 10) \land (y \geq 3)$$

№	x	y	x > 0	x + y = 10	y ≥ 3	F	Комментарий
0	?	?	?	?	?	?	Начальное состояние
1	-9999	?	false	?	?	false	Предположение: x = -9999 → F = false
2	9999	-9989	true	true	false	false	Предположение: x = 9999 → y = 10 − x = 10 - 9999 = -9989 → y ≥ 3 нарушено
3	?	?	?	?	?	?	Предположение: y = 3
4	7	3	true	true	true	true	x = 10 - y = 10 - 3 = 7 → SAT - решение найдено

SAT → SMT

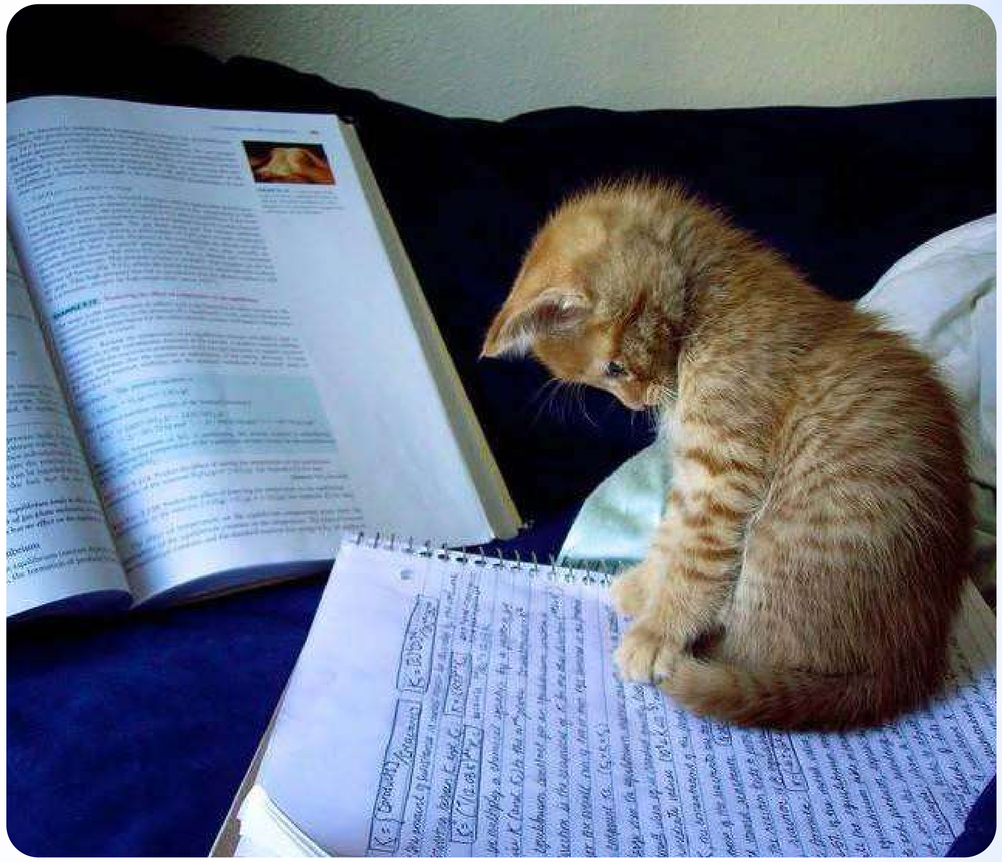
$$F = (x > 0) \land (x + y = 10) \land (y \geq 3)$$

№	x	y	x > 0	x + y = 10	y ≥ 3	F	Комментарий
0	?	?	?	?	?	?	Начальное состояние
1	-9999	?	false	?	?	false	Предположение: x = -9999 → F = false
2	9999	-9989	true	true	false	false	Предположение: x = 9999 → y = 10 − x = 10 - 9999 = -9989 → y ≥ 3 нарушено
3	?	?	?	?	?	?	Предположение: y = 3
4	7	3	true	true	true	true	x = 10 - y = 10 - 3 = 7 → SAT - решение найдено

KSMT

$$F = (x > 0) \land (x + y = 10) \land (y \geq 3)$$

```
(declare-fun x_0 () Int)
(declare-fun y_0 () Int)

(assert
  (and
    (> x_0 0)
    (= (+ x_0 y_0) 10)
    (>= y_0 3)
  )
)
```



Зачем нам SSA?

Исходный код · выполняется и даёт x = 8

```
var x = 5
x = x + 3
```

Зачем нам SSA?

Исходный код · выполняется и даёт x = 8

```
var x = 5
x = x + 3
```

Без SSA

$$x = 5 \;\wedge\; x = x + 3$$

подбираем x:

x = 5 → 5 = 5 + 3 ✗

любое x → x = x + 3 ✗

↓

ни одно значение не подходит

UNSAT

«такой программы не существует»

Зачем нам SSA?

Исходный код · выполняется и даёт x = 8

```
var x = 5
x = x + 3
```

Без SSA

$$x = 5 \;\wedge\; x = x + 3$$

подбираем x:

x = 5 → 5 = 5 + 3 ✗

любое x → x = x + 3 ✗

↓

ни одно значение не подходит

UNSAT

«такой программы не существует»

С SSA

$$x_0 = 5 \;\wedge\; x_1 = x_0 + 3$$

подбираем x_0, x_1:

$x_0 = 5$ → 5 = 5 ✓

$x_1 = 8$ → 8 = 5 + 3 ✓

↓

найдено: $x_0 = 5$, $x_1 = 8$

SAT

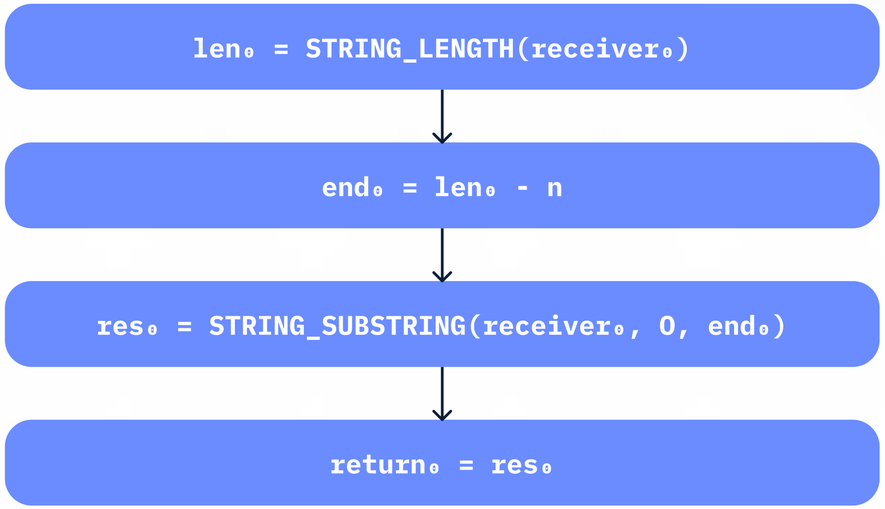
формула выполнима

Зачем нам SSA?

Код

```
fun dropLast(receiver: string, n: int): string {
  var length: int = action STRING_LENGTH(receiver);
  var end: int = length - n;
  result = action STRING_SUBSTRING(receiver, 0, end);
}
```

SSA



Трассы

Trace 1

Action: LENGTH(receiver)

Action: SUBSTRING(receiver, 0, end₀)

Result: res₀

Зачем нам SSA?

Код

```
fun dropLast(receiver: string, n: int): string {
  var length: int = action STRING_LENGTH(receiver);
  var end: int = length - n;
  result = action STRING_SUBSTRING(receiver, 0, end);
}
```

SSA

$len_0 = STRING_LENGTH(receiver_0)$

↓

$end_0 = len_0 - n$

↓

$res_0 = STRING_SUBSTRING(receiver_0, 0, end_0)$

↓

$return_0 = res_0$

Трассы

Trace 1

Action: LENGTH(receiver)

Action: SUBSTRING(receiver, 0, end_0)

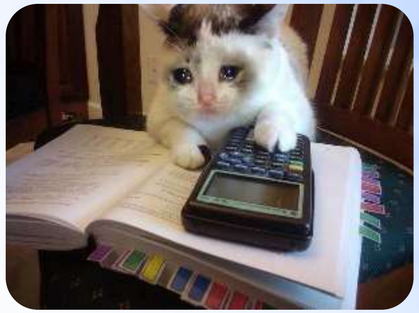
Result: res_0

↓

Что-то, что сравнивает SMT-решатель:

$(len_0 = STRING_LENGTH(receiver_0))$
$\wedge\ (end_0 = len_0 - n)$
$\wedge\ (res_0 = STRING_SUBSTRING(receiver_0, 0, end_0))$
$\wedge\ (return_0 = res_0)$

Зачем нам SSA?



Код

```
@Extension
@Property
automaton String.length: String {

    fun get(receiver: string): int {
        result = action
STRING_LENGTH(receiver);
    }
}
```

SSA

$result_1$ = STRING_LENGTH (receiver)

↓

$return_1 = result_1$

Трассы

Trace 1

Action: LENGTH(receiver)

Result: $result_1$

Код

```
@Extension
automaton String.substring: String {

    fun call(receiver: string, start:
int, end: int): string {
        result = action
STRING_SUBSTRING(receiver, start,
end);
    }
}
```

SSA

$result_1$ = STRING_SUBSTRING (receiver, start, end)

↓

$return_1 = result_1$

Трассы

Trace 1

Action: SUBSTRING(receiver, start, end)

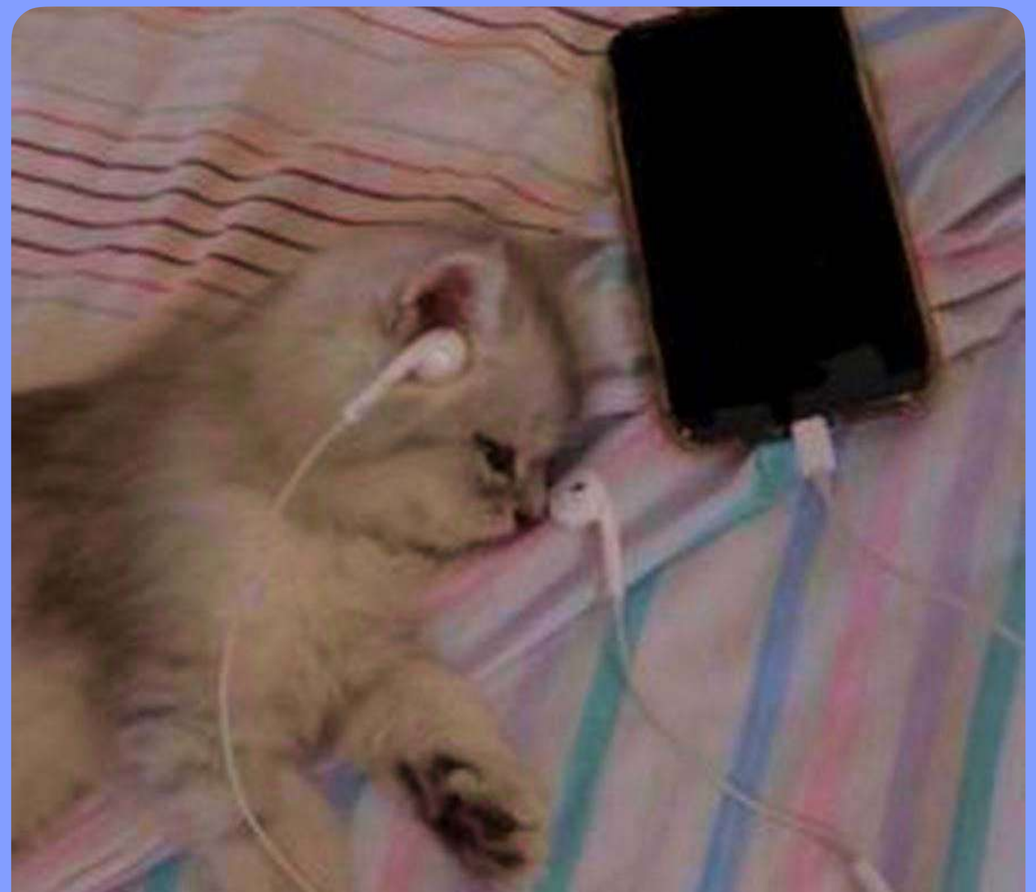
Result: $result_1$

Что-то, что сравнивает SMT-решатель:

$(result_0 = \text{STRING_SUBSTRING}(receiver_0, start, end)) \land (return_0 = result_0)$

Как сравнить поведение двух функций?

Я, пытающийся избавиться
от математики в докладе…



продолжаем…

Задача: собрать исходную трассу из целевых

Исходная — Kotlin

dropLast(s, n)

- LENGTH(s)
- SUBSTRING(s, 0, len-n)

s, n — свободные

→

Словарь целевых трасс — ArkTS

removeLast(r)

- LENGTH(r)
- SUBSTRING(r, 0, len-1)

1 зашит вместо свободного n!

length(s)

- LENGTH(s)

substring(s, e)

- SUBSTRING(s,0,e)

take(s, n)

- SUBSTRING(s,0,n)

trim(s)

- TRIM(s)

toLowerCase(s)

- TO_LOWER(s)

рабочие блоки
не подходит
посторонние (decoy)

Два вопроса

SMT #1 — какие функции и в каком порядке?
Проверяет:
Сумма длин выбранных трасс = длина исходной (полное покрытие)
Действия по позициям совпадают:

SMT #2 — с какими аргументами?
Проверяет:
- Аргументы каждого действия совпадают
- CREATE_FILE(path) = CREATE_FILE("/tmp")? Нет! Путь зашит.

Для ответа используем **SMT-решатель** — дважды.

Два вопроса

- 200 функций, цепочка до 5 → **млрд** комбинаций
- Перебор не работает → используем **SMT-решатель**
- **SMT** = программа, которая ищет значения переменных по ограничениям
- Мы формулируем задачу как набор ограничений — решатель находит ответ за **миллисекунды**

SMT #1: ищем цепочку по названиям

Каждому действию — число: LENGTH =1 SUBSTRING =2 TRIM =3 TO_LOWER =4 INDEX_OF =5

Исходная трасса dropLast: 1 2 m = 2

т	Функция	Действия	Числа	Длина
T_1	length	LENGTH	[1]	1
T_2	substring(start, end)	SUBSTRING	[2]	1
T_3	removeLast()	LENGTH SUBSTRING	[1, 2]	2
T_4	indexOf(sub)	INDEX_OF	[5]	1
T_5	trim()	TRIM	[3]	1
T_6	toLowerCase()	TO_LOWER	[4]	1

SMT #1, N=1: перебираем все трассы

Нужно покрыть 1 2 **(m=2) одной трассой. Проверяем каждую:**

τ	кандидат	действия	длина = 2?	числа = [1,2]?
τ_1	length()	1	1 ≠ 2 ✗	–
τ_2	substring()	2	1 ≠ 2 ✗	–
τ_3	indexOf()	5	1 ≠ 2 ✗	–
τ_4	trim()	3	1 ≠ 2 ✗	–
τ_5	toLowerCase()	4	1 ≠ 2 ✗	–
τ_6	removeLast()	1 2	2 = 2 ✓	[1, 2] ✓

→ **SAT!** Решение при N=1: removeLast()

Аргументы проверим позже (SMT #2)

SMT #1, N=1: перебираем все трассы

Нужно покрыть 1 2 **(m=2) одной трассой. Проверяем каждую:**

τ	кандидат	действия	длина = 2?	числа = [1,2]?
τ_1	length()	1	1 ≠ 2 ✗	–
τ_2	substring()	2	1 ≠ 2 ✗	–
τ_3	indexOf()	5	1 ≠ 2 ✗	–
τ_4	trim()	3	1 ≠ 2 ✗	–
τ_5	toLowerCase()	4	1 ≠ 2 ✗	–
τ_6	removeLast()	1 2	2 = 2 ✓	[1, 2] ✓

→ **SAT!** Решение при N=1: removeLast()

Аргументы проверим позже (SMT #2)

SMT #1, N=2: перебираем все пары

Нужно покрыть 1 2 (m=2)двумя трассами. 6 трасс × 6 = 36 пар. Проверяем:

шаг 1		шаг 2		длины	числа
length	1	length	1	1+1=2 ✓	[1,1] ≠ [1,2] ✗
substring	2	length	1	1+1=2 ✓	[2,1] ≠ [1,2] ✗

SMT #1, N=2: перебираем все пары

Нужно покрыть 1 2 (m=2)двумя трассами. 6 трасс × 6 = 36 пар. Проверяем:

шаг 1		шаг 2		длины	числа
length	1	length	1	1+1=2 ✓	[1,1] ≠ [1,2] ✗
substring	2	length	1	1+1=2 ✓	[2,1] ≠ [1,2] ✗
length	1	removeLast	1 2	1+2=3 ✗	длина ≠ 2
removeLast	1 2	substring	2	2+1=3 ✗	длина ≠ 2

SMT #1, N=2: перебираем все пары

Нужно покрыть 1 2 (m=2)двумя трассами. 6 трасс × 6 = 36 пар. Проверяем:

шаг 1		шаг 2		длины	числа
length	1	length	1	1+1=2 ✓	[1,1] ≠ [1,2] ✗
substring	2	length	1	1+1=2 ✓	[2,1] ≠ [1,2] ✗
length	1	removeLast	1 2	1+2=3 ✗	длина ≠ 2
removeLast	1 2	substring	2	2+1=3 ✗	длина ≠ 2
length	1	trim	3	1+1=2 ✓	[1,3] ≠ [1,2] ✗
substring	2	toLowerCase	4	1+1=2 ✓	[2,4] ≠ [1,2] ✗

SMT #1, N=2: перебираем все пары

Нужно покрыть 1 2 (m=2)двумя трассами. 6 трасс × 6 = 36 пар. Проверяем:

шаг 1		шаг 2		длины	числа
length	1	length	1	1+1=2 ✓	[1,1] ≠ [1,2] ✗
substring	2	length	1	1+1=2 ✓	[2,1] ≠ [1,2] ✗
length	1	removeLast	1 2	1+2=3 ✗	длина ≠ 2
removeLast	1 2	substring	2	2+1=3 ✗	длина ≠ 2
length	1	trim	3	1+1=2 ✓	[1,3] ≠ [1,2] ✗
substring	2	toLowerCase	4	1+1=2 ✓	[2,4] ≠ [1,2] ✗
...		...		...	всего 36 пар

SMT #1, N=2: перебираем все пары

Нужно покрыть 1 2 (m=2)двумя трассами. 6 трасс × 6 = 36 пар. Проверяем:

шаг 1		шаг 2		длины	числа
length	1	length	1	1+1=2 ✓	[1,1] ≠ [1,2] ✗
substring	2	length	1	1+1=2 ✓	[2,1] ≠ [1,2] ✗
length	1	removeLast	1 2	1+2=3 ✗	длина ≠ 2
removeLast	1 2	substring	2	2+1=3 ✗	длина ≠ 2
length	1	trim	3	1+1=2 ✓	[1,3] ≠ [1,2] ✗
substring	2	toLowerCase	4	1+1=2 ✓	[2,4] ≠ [1,2] ✗
...		...		...	всего 36 пар
length	1	substring	2	1+1=2 ✓	[1,2] = [1,2] ✓

SMT #1, N=2: нашли!

1 length + 2 substring = 1 2 → SAT!

Итого SMT #1 нашёл два решения:

N	цепочка	как собрана
1	removeLast()	1 2 - одна функция
2	length() + substring()	1 + 2 - две функции

По числам оба подходят.
Но подходят ли по аргументам? Это проверит SMT #2.

То, что решатель делал на предыдущем слайде, записывается так:

$$\Phi_N = \bigwedge_{s=1}^{N} \Big(x_s \in \{1..k\} \ \wedge \ c_s = n_{x_s} \ \wedge \ \bigwedge_j \varphi(\sigma[p_s + j]) = \varphi(\tau_{x_s}[j]) \Big)$$

$$\wedge \quad \sum_{s=1}^{N} c_s = m$$

	Что означает
x_s	какую трассу взять на шаге s
$c_s = n_{x_s}$	длина выбранной трассы
$\varphi(\sigma[\dots]) = \varphi(\tau[\dots])$	числа действий совпадают
$\sum c_s = m$	сумма длин = длина исходной

SMT #1 нашёл цепочки. Но этого мало.

SMT #1 подобрал **какие** функции вызвать. Но мы пока не проверили **аргументы**. Вспомним, что внутри спецификации у каждого действия есть аргументы:

```
fun dropLast(receiver, n):
    action STRING_LENGTH(receiver)
    action STRING_SUBSTRING(receiver, 0, len - n)
```

n — свободный параметр

```
fun dropLast(receiver, n):
    action STRING_LENGTH(receiver)
    action STRING_SUBSTRING(receiver, 0, len - 1)
```

1 — защитая константа

SMT #1 нашёл цепочки. Но этого мало.

SMT #1 подобрал **какие** функции вызвать. Но мы пока не проверили **аргументы**. Вспомним, что внутри спецификации у каждого действия есть аргументы:

```
fun dropLast(receiver, n):
    action STRING_LENGTH(receiver)
    action STRING_SUBSTRING(receiver, 0, len - n)
```

n — свободный параметр

```
fun dropLast(receiver, n):
    action STRING_LENGTH(receiver)
    action STRING_SUBSTRING(receiver, 0, len - 1)
```

1 — защитая константа

Действия одинаковые: `LENGTH` `SUBSTRING` = `LENGTH` `SUBSTRING`

Но **len – n** и **len – 1** — это разные вещи! Нужна вторая проверка → **SMT #2**: можно ли выразить параметры целевой через параметры исходной?

SMT #2: что проверяем?

Идея: параметры **целевой** функции выразить через параметры **исходной** так, чтобы трассы совпадали при любых значениях источника.

dropLast(s, n) → length() + removeLast(?)

вместо «?» решатель подставит s, n или их комбинацию

Аргументы попозиционно приравниваем — получаем систему уравнений. Решатель находит:

- **SAT** + **подстановку** — что именно поставить вместо «?»
- **UNSAT** + **контрпример** — конкретный вызов, который заменить нельзя

переменная — свободна, **константа** — защита в спеку.

SMT #2: проверяем removeLast()

	dropLast(s, n)	removeLast(s)	должно совпасть
LENGTH	LENGTH(s)	LENGTH(s)	s = s ✓
SUBSTRING	SUBSTRING(s, 0, len-n)	SUBSTRING(s, 0, len-1)	n = 1 ?

Условие: **n** = **1**. Но **n** - это аргумент вызова dropLast, он может быть любым. При **n** ≠ **1** замена не работает.

SMT #2: проверяем removeLast()

	dropLast(s, n)	removeLast(s)	должно совпасть
LENGTH	LENGTH(s)	LENGTH(s)	s = s ✓
SUBSTRING	SUBSTRING(s, 0, len-n)	SUBSTRING(s, 0, len-1)	n = 1 ?

Условие: **n** = **1**. Но **n** - это аргумент вызова dropLast, он может быть любым. При **n** ≠ **1** замена не работает.

Контрпример от решателя: n := 2
требуется **2 = 1** - ложь ⇒ **UNSAT**

Вывод: вызов dropLast(s, 2) **нельзя заменить** на removeLast(s).

Спецификации второго решения

```
fun length():
  action
STRING_LENGTH(receiver)
```

```
fun substring(start, end):
    action STRING_SUBSTRING(receiver,
start, end)
```

Все параметры — **свободные**: **start** и **end** можно подставить как угодно.

Спецификации второго решения

```
fun length():
  action STRING_LENGTH(receiver)
```

```
fun substring(start, end):
    action STRING_SUBSTRING(receiver, start, end)
```

Все параметры — **свободные**: **start** и **end** можно подставить как угодно.

Сравним с исходной:

```
fun dropLast(receiver, n):
  action STRING_LENGTH(receiver)   ← длина строки
  action STRING_SUBSTRING(receiver, 0, len-n)
```

Похоже, можно подставить **start = 0**, **end = len-n**. Проверим!

SMT #2: проверяем .length + .substring(...)

	dropLast(s, n)	length / substring	должно совпасть
LENGTH	LENGTH(s)	LENGTH(s)	s = s ✓
SUBSTRING	SUBSTRING(s, 0, len-n)	SUBSTRING(s, start, end)	start=0, end=len-n ✓

В отличие от **removeLast**, справа нигде нет зашитой **1** —
везде свободные параметры, решатель волен подобрать их под любое **n**.

SMT #2: проверяем .length + .substring(...)

	dropLast(s, n)	length / substring	должно совпасть
LENGTH	LENGTH(s)	LENGTH(s)	s = s ✓
SUBSTRING	SUBSTRING(s, 0, len-n)	SUBSTRING(s, start, end)	start=0, end=len-n ✓

В отличие от **removeLast**, справа нигде нет зашитой **1** —
везде свободные параметры, решатель волен подобрать их под любое **n**.

Подстановка от решателя: **start := 0, end := len-n**
работает при любом **n** ⇒ **SAT**

Похоже, можно подставить **start = 0**, **end = len-n**. Проверим!

Ещё пример: строковые функции

Посмотрим ту же логику на строках. Спецификации:

```
fun take(s, n):                          fun substring(s_t, start, end):
  action STRING_SUBSTRING(s, 0, n)         action STRING_SUBSTRING(s_t, start, end)
```

↑ 0 зашит!

SMT #1 нашёл: оба дают `STR_SUBSTR` → числа совпали.

SMT #2 ставит аргументы друг напротив друга и решает **по шагам.**

SMT #2 пошагово: take → substring

Посмотрим ту же логику на строках. Спецификации:

```
Исходная:  STRING_SUBSTRING( s,    0,      n   )
Целевая:   STRING_SUBSTRING( s_t,  start,  end )
```

Решатель приравнивает аргументы попозиционно и решает по шагам:

Шаг 1. `s = s_t → s_t := s` ✓

Шаг 2. `0 = start → start :=0` ✓ (константа присваивается переменной)

Шаг 3. `n = end → end := n` ✓

SMT #2 пошагово: take → substring

Посмотрим ту же логику на строках. Спецификации:

Исходная: STRING_SUBSTRING(s, 0, n)
Целевая: STRING_SUBSTRING(s_t, start, end)

Решатель приравнивает аргументы попозиционно и решает по шагам:

Шаг 1. s = s_t → s_t := s ✓

Шаг 2. 0 = start → start :=0 ✓(константа присваивается переменной)

Шаг 3. n = end → end := n ✓

Подстановка от решателя: s_t:= s, start:= 0, end:= n ⇒ SAT

take(s, n) → substring(s, 0, n)

SMT #2 пошагово: substring → take (обратно)

Исходная: STRING_SUBSTRING(s, start, end)

Целевая: STRING_SUBSTRING(s_t, 0, n)

↑ зашит в спеку take!

Решатель приравнивает аргументы попозиционно и решает по шагам:

SMT #2 пошагово: substring → take (обратно)

Исходная: STRING_SUBSTRING(s, start, end)

Целевая: STRING_SUBSTRING(s_t, 0, n)

↑ зашит в спеку take!

Решатель приравнивает аргументы попозиционно и решает по шагам:

Шаг 1. s = s_t → s_t := s ✓

Шаг 2. start = 0

start — параметр substring (дан вызовом, любой)

0 — константа из спеки take (зашит, **изменить нельзя**)

→ требуется: start = 0 **всегда**. Но start произвольный!

SMT #2 пошагово: substring → take (обратно)

Исходная: STRING_SUBSTRING(s, start, end)

Целевая: STRING_SUBSTRING(s_t, 0, n)

↑ зашит в спеку take!

Решатель приравнивает аргументы попозиционно и решает по шагам:

Шаг 1. s = s_t → s_t := s ✓

Шаг 2. start = 0

start — параметр substring (дан вызовом, любой)

0 — константа из спеки take (зашит, **изменить нельзя**)

→ требуется: start = 0 **всегда**. Но start произвольный!

Шаг 3. Не проверяем — уже UNSAT на шаге 2.

SMT #2 пошагово: substring → take (обратно)

Исходная: STRING_SUBSTRING(s, start, end)

Целевая: STRING_SUBSTRING(s_t, 0, n)

↑ зашит в спеку take!

Решатель приравнивает аргументы попозиционно и решает по шагам:

Шаг 1. s

Шаг 2. start = 0

start — параметр substring (дан вызовом, любой)

0 — константа из спеки take (зашит, **изменить нельзя**)

→ требуется: start = 0 **всегда**. Но start произвольный!

Шаг 3. Не проверяем — уже UNSAT на шаге 2.

Контрпример от решателя: start:= **2** ⇒ **2** ≠ 0 ⇒ **UNSAT**

То, что решатель проверял в каждом примере SMT #2, записывается так:

$$\Psi = \bigwedge_{j=1}^{m} \bigwedge_{k=1}^{\mathrm{arity}(\alpha_j)} \mathrm{arg}_{\text{исх}}(\alpha_j, k) = \mathrm{arg}_{\text{цел}}(\beta_j, k)$$

	Что означает
α_j, β_j	j-е действие в исходной трассе и в целевой цепочке
$\mathrm{arg}_{\text{исх}}(\alpha_j, k)$	k-й аргумент действия α_j из исходной спеки
$\mathrm{arg}_{\text{цел}}(\beta_j, k)$	то же из целевой спеки
$\bigwedge_{j,k}$	конъюнкция всех попозиционных равенств — это и есть Ψ

Что если SMT #2 отверг?

SMT #1 нашёл цепочку. SMT #2 отверг аргументы.
Что дальше?

Добавляем **запрет** к формуле Φ — «эту цепочку больше не предлагай»:

```
Попытка 1:  Φ                          → цепочка A → SMT #2 отверг
Попытка 2:  Φ ∧ ¬(A)                   → цепочка B → SMT #2 отверг
Попытка 3:  Φ ∧ ¬(A) ∧ ¬(B)            → цепочка C → SMT #2: SAT!
Всё исчерпано:                         → увеличиваем N
```

Формула **растёт** — решатель перебирает варианты, пока не найдёт подходящий.

Полный алгоритм

```
for N = 1, 2, 3, ... , 50:
    Ф = построить_Ф_N()
```

пробуем длину цепочки

формула SMT #1

Полный алгоритм

```
for N = 1, 2, 3, ... , 50:
    Φ = построить_Φ_N()
    while true:
result = solver.checkTraces(Φ)
```

пробуем длину цепочки

формула SMT #1

перебираем варианты

Полный алгоритм

```
for N = 1, 2, 3, ... , 50:
    Φ = построить_Φ_N()
    while true:
result = solver.checkTraces(Φ)
        if UNSAT: break
```

пробуем длину цепочки

формула SMT #1

перебираем варианты

все варианты длины N исчерпаны

Полный алгоритм

```
for N = 1, 2, 3, ... , 50:
    Φ = построить_Φ_N()
    while true:
result = solver.checkTraces(Φ)
        if UNSAT: break
        actions = extract_traces(Φ)
```

пробуем длину цепочки

формула SMT #1

перебираем варианты

все варианты длины N исчерпаны

SMT #1 нашёл цепочку

Полный алгоритм

```
for N = 1, 2, 3, ... , 50:        пробуем длину цепочки
    Φ = построить_Φ_N()           формула SMT #1
    while true:                   перебираем варианты
result = solver.checkTraces(Φ)
        if UNSAT: break           все варианты длины N исчерпаны
        actions = extract_traces(Φ)   SMT #1 нашёл цепочку
       if solver.checkActions(actions):   SMT #2 проверил аргументы
           return actions         оба SAT → готово!
```

Полный алгоритм

```
for N = 1, 2, 3, ... , 50:        пробуем длину цепочки
    Φ = построить_Φ_N()           формула SMT #1
    while true:                   перебираем варианты
result = solver.checkTraces(Φ)
        if UNSAT: break           все варианты длины N исчерпаны
        actions = extract_traces(Φ)   SMT #1 нашёл цепочку
       if solver.checkActions(actions):   SMT #2 проверил аргументы
            return actions        оба SAT → готово!
       Φ = Φ ∧ ¬(actions)         запрещаем, пробуем другую
```

Полный алгоритм

```
for N = 1, 2, 3, ... , 50:        пробуем длину цепочки
    Φ = построить_Φ_N()           формула SMT #1
    while true:                   перебираем варианты
result = solver.checkTraces(Φ)
        if UNSAT: break           все варианты длины N исчерпаны
        actions = extract_traces(Φ)   SMT #1 нашёл цепочку
       if solver.checkActions(actions):   SMT #2 проверил аргументы
            return actions        оба SAT → готово!
       Φ = Φ ∧ ¬(actions)         запрещаем, пробуем другую
return INCOMPATIBLE               функции несовместимы
```

Полный алгоритм

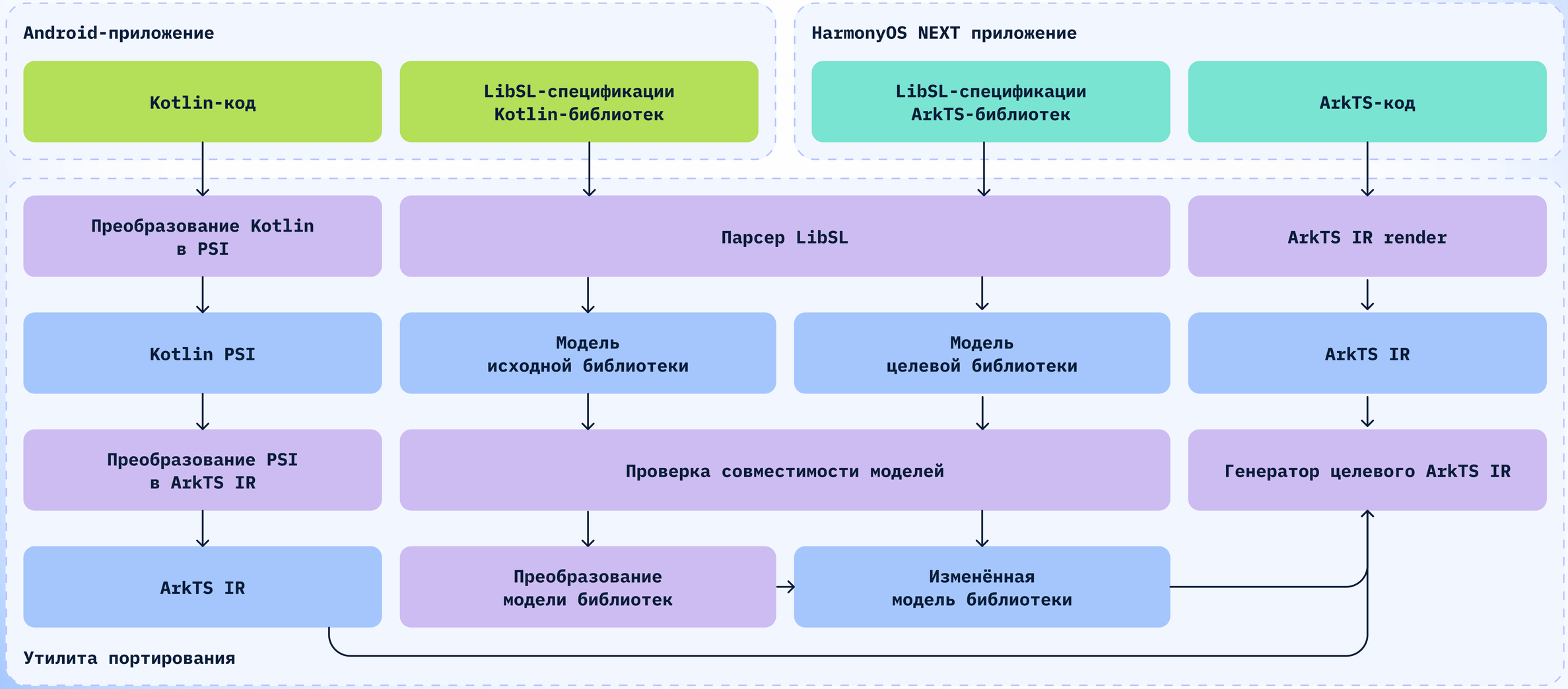
```
for N = 1, 2, 3, ... , 50:
    Φ = построить_Φ_N()
    while true:
result = solver.checkTraces(Φ)
        if UNSAT: break
        actions = extract_traces(Φ)
       if solver.checkActions(actions):
            return actions
       Φ = Φ ∧ ¬(actions)
return INCOMPATIBLE
```

$$\Phi_N = \bigwedge_{s=1}^{N} \Big(x_s \in \{1..k\} \ \wedge \ c_s = n_{x_s} \ \wedge \bigwedge_{j} \varphi(\sigma[p_s+j]) = \varphi(\tau_{x_s}[j]) \Big)$$

$$\wedge \quad \sum_{s=1}^{N} c_s = m$$

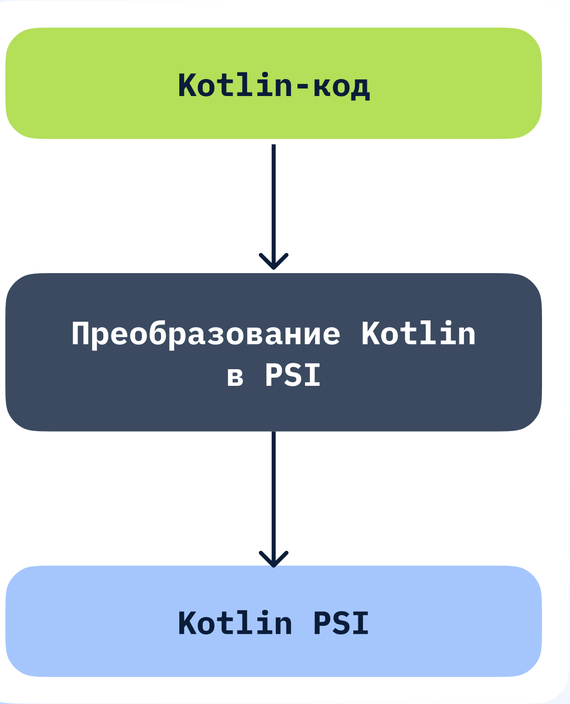
$$\Psi = \bigwedge_{j=1}^{m} \bigwedge_{k=1}^{\mathrm{arity}(\alpha_j)} \mathrm{arg}_{\text{исх}}(\alpha_j, k) \ = \ \mathrm{arg}_{\text{цел}}(\beta_j, k)$$

Используемый подход



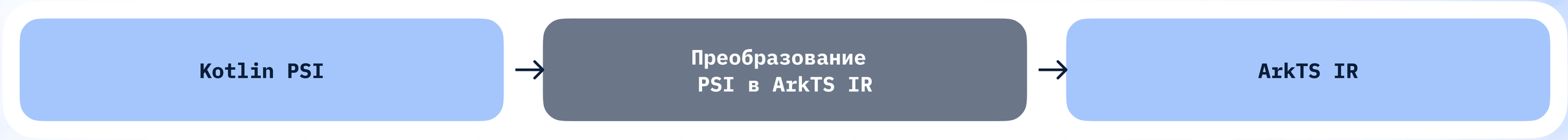
Используемый подход

```kotlin
fun maskCardNumber(number: String): String {
    return number.dropLast(4) + "****"
}
```




```
·KtFile
  └── KtNamedFunction: "maskCardNumber"
      ├── KtParameter: "number"
      │   └── KtTypeReference: "String"
      ├── KtTypeReference: "String"
      └── KtBlockExpression
          └── KtReturnExpression
              └── KtBinaryExpression: "+"
                  ├── KtDotQualifiedExpression
                  │   ├── KtNameReferenceExpression: "number"
                  │   └── KtCallExpression: "dropLast"
                  │       └── KtValueArgument
                  │           └── KtConstantExpression: 4
                  └── KtStringTemplateExpression: "****"
```

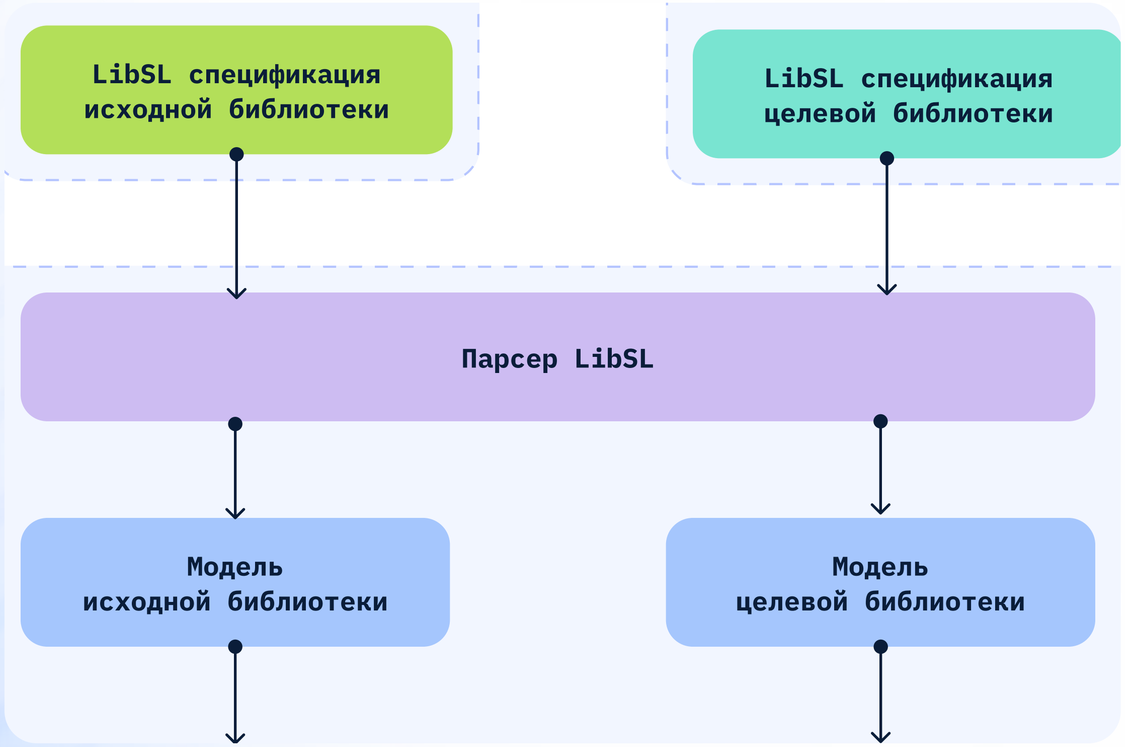
Используемый подход




```
· KtFile
  └── KtNamedFunction: "maskCardNumber"
      ├── KtParameter: "number"
      │   └── KtTypeReference: "String"
      ├── KtTypeReference: "String"
      └── KtBlockExpression
          └── KtReturnExpression
              └── KtBinaryExpression: "+"
                  ├── KtDotQualifiedExpression
                  │   ├── KtNameReferenceExpression: "number"
                  │   └── KtCallExpression: "dropLast"
                  │       └── KtValueArgument
                  │           └── KtConstantExpression: 4
                  └── KtStringTemplateExpression: "****"
```

```
ArkTsFile
  └── FunctionDeclaration: "maskCardNumber" (export)
      ├── Parameter: "number"
      │   └── TypeReference: "string"
      ├── TypeReference: "string"
      └── BlockStatement
          └── ReturnStatement
              └── BinaryExpression: "+"
                  ├── CallExpression: "???"
                  │   ├── Identifier: "???"
                  │   └── NumericLiteral: ???
                  └── StringLiteral: "****"
```

Используемый подход




```
fun dropLast(receiver: string, n: int): string {
    var length: int = action STRING_LENGTH(receiver);
    var end: int = length - n;
    result = action STRING_SUBSTRING(receiver, 0, end);
}

@Extension
automaton String.substring: String {

    fun call(receiver: string, start: int, end: int):
string {
        result = action STRING_SUBSTRING(receiver,
start, end);
    }
}

@Extension
@Property
automaton String.length: String {

    fun get(receiver: string): int {
        result = action STRING_LENGTH(receiver);
    }
}
```

Используемый подход

Код

```
fun dropLast(receiver: string, n:
int): string {
var length: int = action
STRING_LENGTH(receiver);
var end: int = length - n;
result = action
STRING_SUBSTRING(receiver, 0, end);
}
```

SSA

len_1 = STRING_LENGTH(receiver)

↓

$end_1 = len_1 - n$

↓

res_1 = STRING_SUBSTRING(receiver, 0, end_1)

↓

$return_1 = res_1$

Трассы

Trace 1

```
Action:  LENGTH(receiver)
Action:  SUBSTRING(receiver,0,0)
Result:  0
```

↓

SAT!

Проверка совместимости моделей

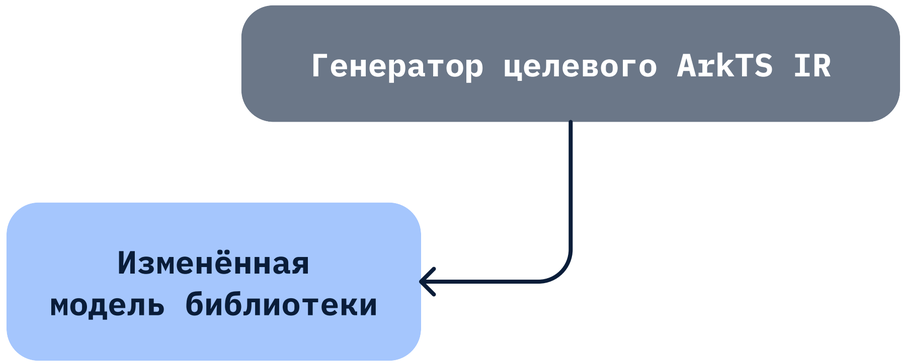
↓

Преобразование модели библиотек → **Изменённая модель библиотеки**

Используемый подход

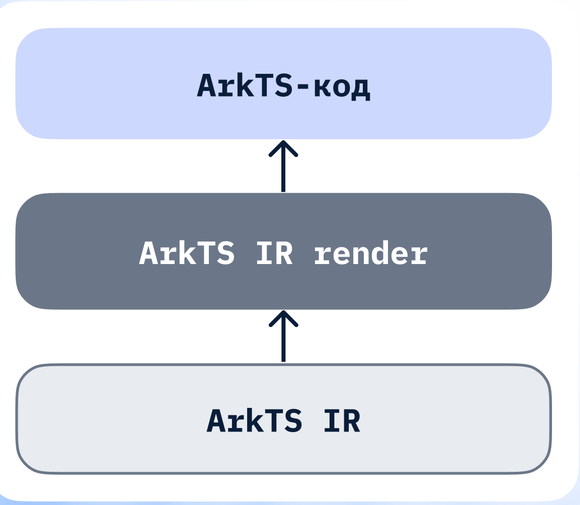
```
dropLast(n) => length + substring
```

```
└── FunctionDeclaration: "generated_dropLast"
    ├── Parameter: "receiver" → string
    ├── Parameter: "n" → number
    ├── ReturnType: string
    └── BlockStatement
        ├── VariableDeclaration: "result0" → number
        │   └── PropertyAccess: "receiver.length"        ← STRING_LENGTH
        ├── VariableDeclaration: "result1" → string
        │   └── CallExpression: "receiver.substring"     ← STRING_SUBSTRING
        │       ├── NumericLiteral: 0
        │       └── BinaryExpression: "-"
        │           ├── Identifier: "result0"
        │           └── Identifier: "n"
        └── ReturnStatement
            └── Identifier: "result1"
```



Используемый подход

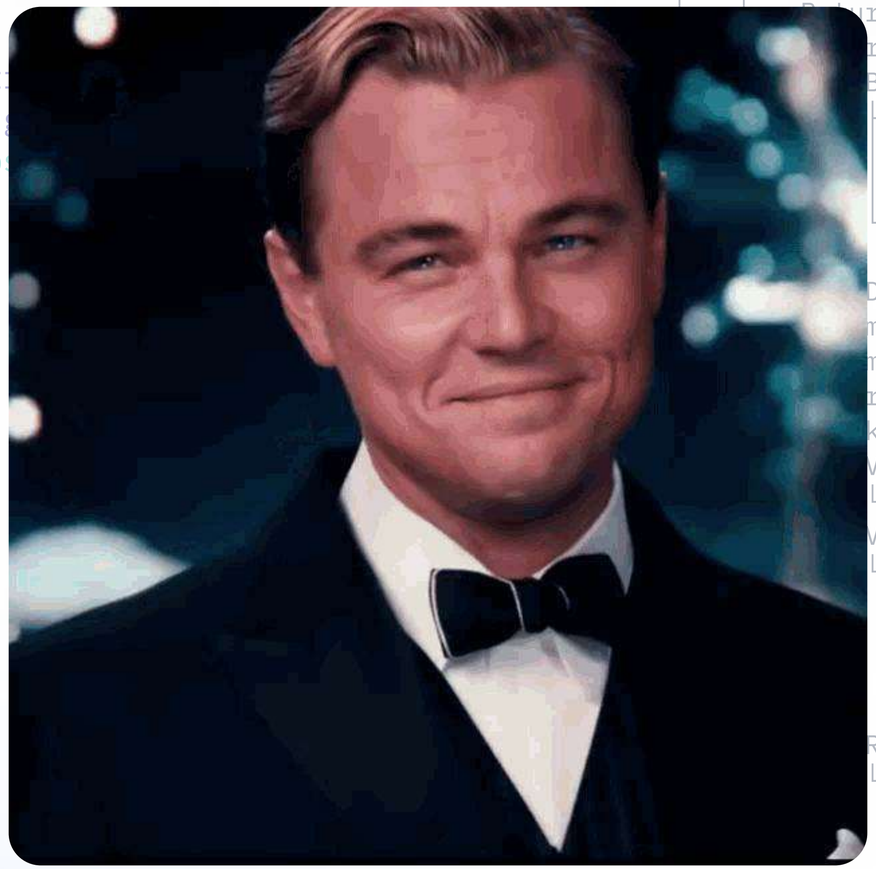
```
export function maskCardNumber(number: string): string {
    return generated_dropLast(number, 4) + "****"
}

function generated_dropLast(receiver: string, n: number): string {
    const result0: number = receiver.length;
    const result1: string = receiver.substring(0, result0 - n);
    return result1
}
```



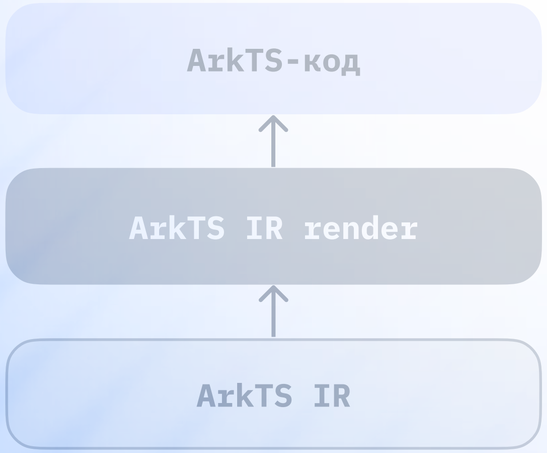

```
ArkTsFile
├── FunctionDeclaration: "maskCardNumber" (export)
│   ├── Parameter: "number" → string
│   ├── ReturnType: string
│   └── ReturnStatement
│       └── BinaryExpression: "+"
│           ├── CallExpression: "generated_dropLast"
│           │   ├── Identifier: "number"
│           │   └── NumericLiteral: 4
│           └── StringLiteral: "****"
│
└── FunctionDeclaration: "generated_dropLast"
    ├── Parameter: "receiver" → string
    ├── Parameter: "n" → number
    ├── ReturnType: string
    └── BlockStatement
        ├── VariableDeclaration: "result0" → number
        │   └── PropertyAccess: "receiver.length"
        ├── VariableDeclaration: "result1" → string
        │   └── CallExpression: "receiver.substring"
        │       ├── NumericLiteral: 0
        │       └── BinaryExpression: "-"
        │           ├── Identifier: "result0"
        │           └── Identifier: "n"
        └── ReturnStatement
            └── Identifier: "result1"
```

Используемый подход

```
export function maskCardNumber(number: string): string {
    return generated_dropLast(number, 4) + "****"
}

function generated_dropLast(receiver: str
    const result0: number = receiver.leng
    const result1: string = receiver.subs
    return result1
}
```



```
ArkTsFile
├── FunctionDeclaration: "maskCardNumber" (export)
│   ├── Parameter: "number" → string
│   ├── ReturnType: string
│   └── ReturnStatement
│       └── BinaryExpression: "+"
│           ├── CallExpression: "generated_dropLast"
│           │   ├── Identifier: "number"
│           │   └── NumericLiteral: 4
│           └── StringLiteral: "****"
│
└── FunctionDeclaration: "generated_dropLast"
    ├── Parameter: "receiver" → string
    ├── Parameter: "n" → number
    ├── ReturnType: string
    └── BlockStatement
        ├── VariableDeclaration: "result0" → number
        │   └── PropertyAccess: "receiver.length"
        ├── VariableDeclaration: "result1" → string
        │   └── CallExpression: "receiver.substring"
        │       ├── NumericLiteral: 0
        │       └── BinaryExpression: "-"
        │           ├── Identifier: "result0"
        │           └── Identifier: "n"
        └── ReturnStatement
            └── Identifier: "result1"
```



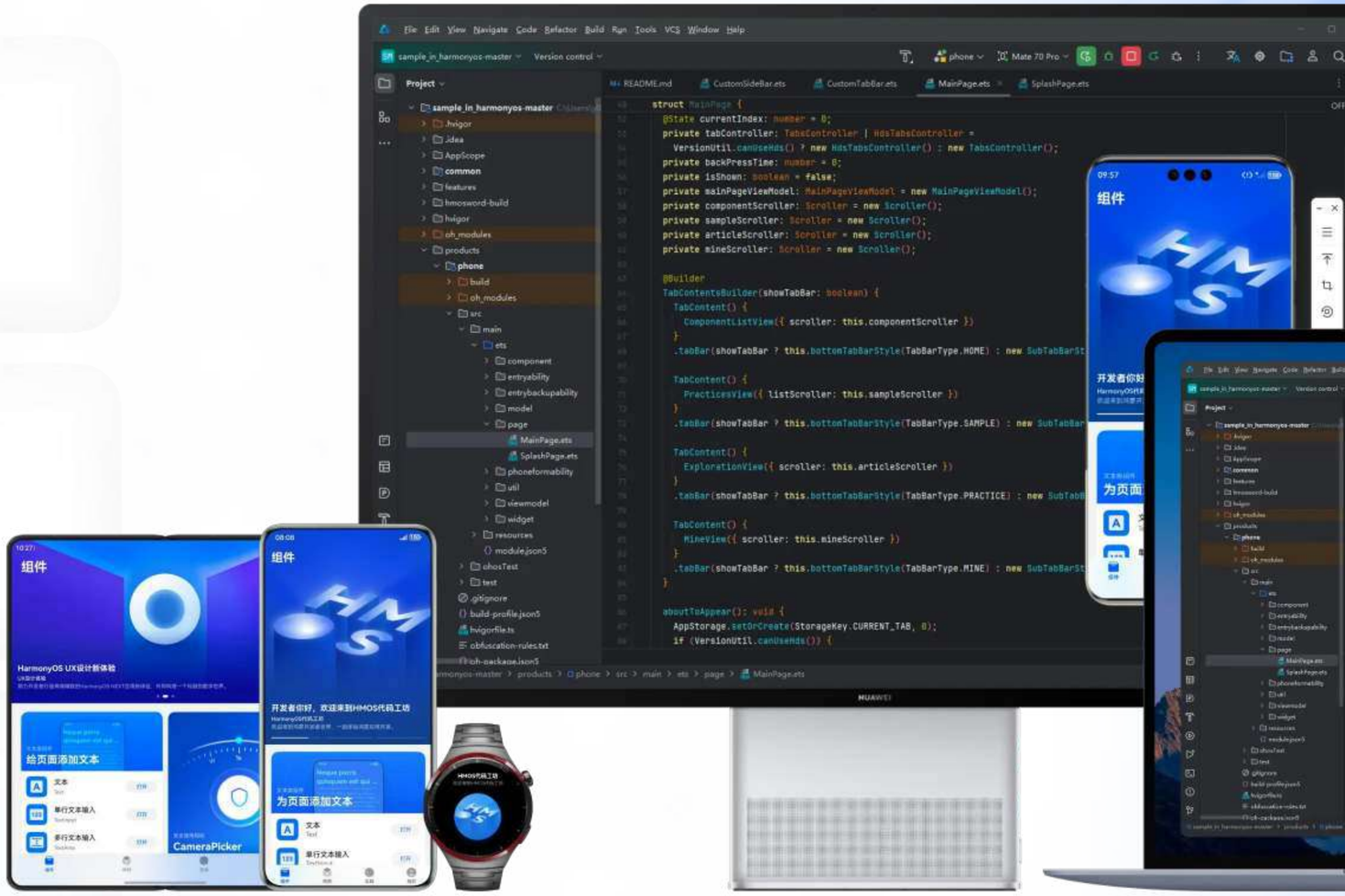
План доклада

1. Контекст и актуальность задачи
2. Обзор существующих решений
3. Постановка задачи и общий подход
4. **Трансляция бизнес-логики Kotlin в ArkTS**
5. Библиотеки, формальные спецификации, SMT
6. **UI: Jetpack Compose в ArkUI**
7. Прототип и результаты

ИТМО ЛИПТСОФТ

ArkUI

1. Декларативный UI фреймворк
2. Разделение “логики” и UI
3. Re-rendering mechanism



Разбор @Composable компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Разбор @Composable компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Наименование компонента

Разбор @Composable компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Передаваемые параметры

Разбор @Composable компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Локальный параметр

Разбор @Composable компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Вызов дочернего компонента

Разбор @Composable компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Content-лямбда

Разбор @Composable компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Наименование вызываемого компонента

Разбор @Composable компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Аргументы вызываемого компонента

Разбор @Composable компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Наименование компонента

```
@ComponentV2
struct CounterButton {

    @Param
    @Require
    baseText: string;

    @Local
    counter: number = 0

    build() {
    }
}
```

Создание ArkUI компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Передаваемые параметры

```
@ComponentV2
struct CounterButton {

    @Param
    @Require
    baseText: string;

    @Local
    counter: number = 0

    build() {
    }
}
```

Создание ArkUI компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Локальный параметр

```
@ComponentV2
struct CounterButton {

    @Param
    @Require
    baseText: string;

    @Local
    counter: number = 0

    build() {
    }
}
```

Создание ArkUI компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Вызов дочернего компонента

```
@ComponentV2
struct CounterButton {

    @Param
    @Require
    baseText: string;

    @Local
    counter: number = 0

    build() {
      Column()
    }
}
```

Создание ArkUI компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Content-лямбда

```
@ComponentV2
struct CounterButton {

    @Param
    @Require
    baseText: string;

    @Local
    counter: number = 0

    build() {
        Column() {
        }
    }
}
```

Создание ArkUI компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Вызов дочернего компонента (1)

```
@ComponentV2
struct CounterButton {

    @Param
    @Require
    baseText: string;

    @Local
    counter: number = 0

    build() {
        Column() {
            Text("Счётчик: " + this.counter)
        }
    }
}
```

Создание ArkUI компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

Вызов дочернего компонента (2)

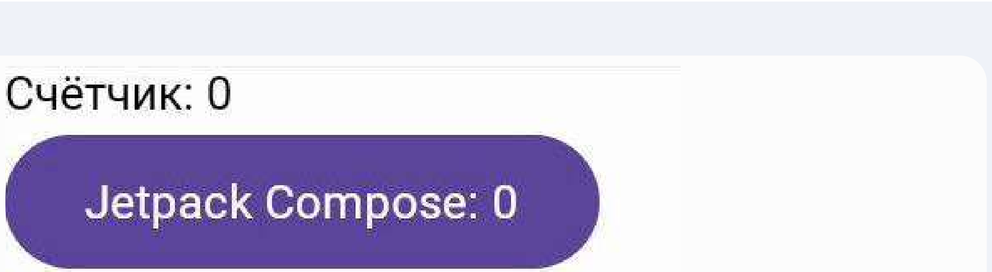
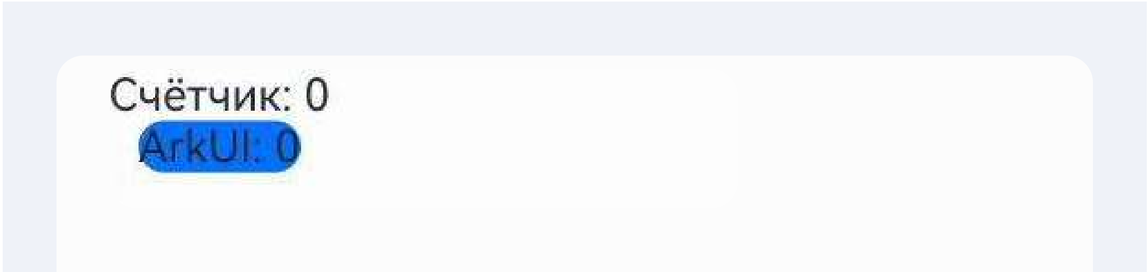
```
@ComponentV2
struct CounterButton {

    @Param
    @Require
    baseText: string;

    @Local
    counter: number = 0

    build() {
        Column() {
            Text("Счётчик: " + this.counter)
            Button() {
              Text(`${this.baseText}: ${this.counter)}`)
              }.onClick(() => {
                 this.counter++
              })
        }
    }
}
```

Итог портирования компонента

```
@Composable
fun CounterButton(
    baseText: String
) {
    var counter = 0

    Column {
        Text(text = "Счётчик: $counter")
        Button(
            onClick = { counter++ }
        ) {
            Text(text = "$baseText: $counter")
        }
    }
}
```

```
@ComponentV2
struct CounterButton {

    @Param
    @Require
    baseText: string;

    @Local
    counter: number = 0

    build() {
        Column() {
            Text("Счётчик: " + this.counter)
            Button() {
              Text(`${this.baseText}: ${this.counter)}`)
              }.onClick(() => {
                 this.counter++
              })
        }
    }
}
```

Итог портирования компонента

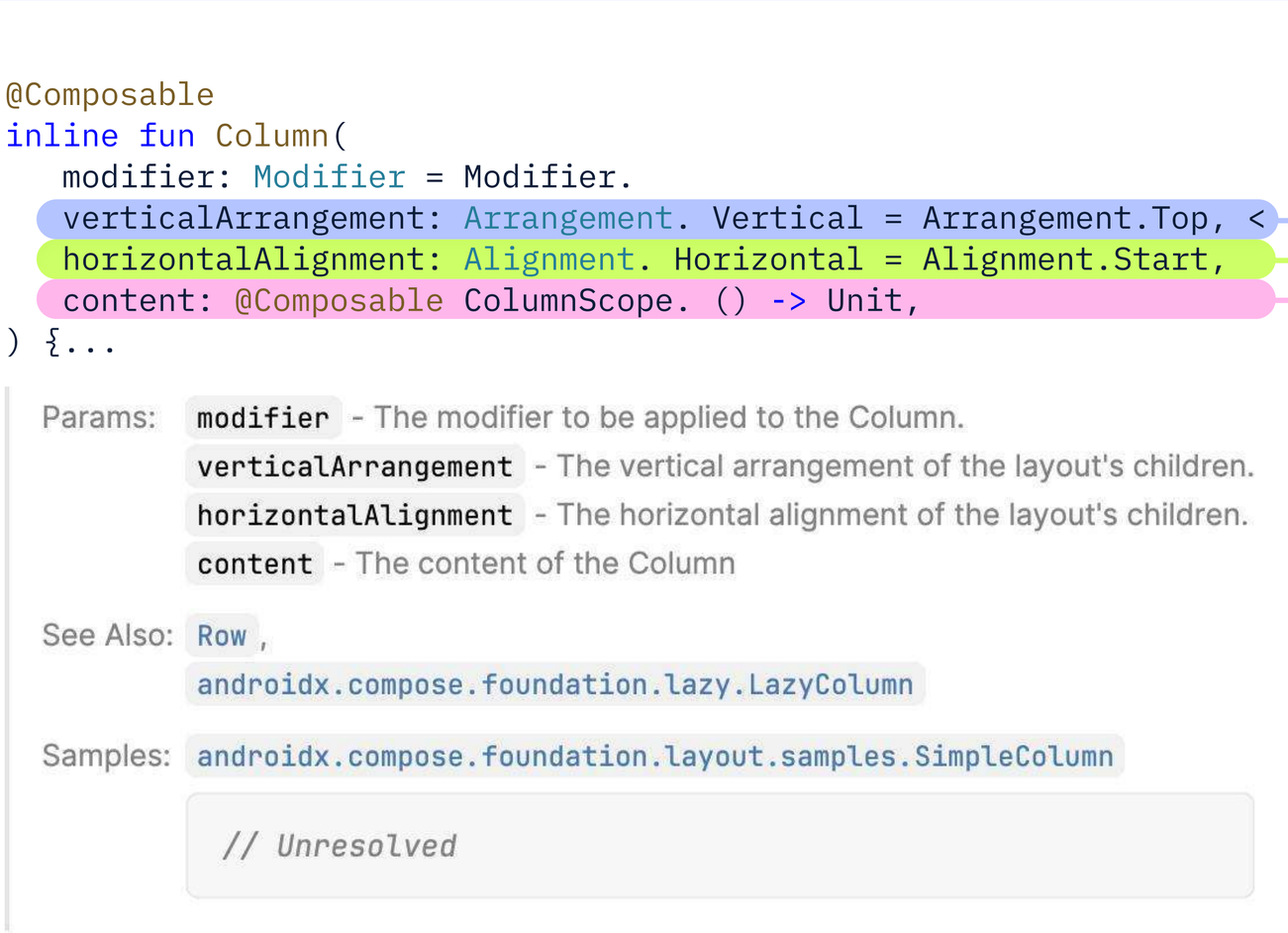
Jetpack Compose	ArkUI
	

Сопоставление компонентов

Jetpack Compose	ArkUI

`@Composable`
`inline fun Column(`
`modifier: Modifier = Modifier.`
`verticalArrangement: Arrangement. Vertical = Arrangement.Top, <`
`horizontalAlignment: Alignment. Horizontal = Alignment.Start,`
`content: @Composable ColumnScope. () -> Unit,`
`) {...`
 	
Params: `modifier` - The modifier to be applied to the Column.
`verticalArrangement` - The vertical arrangement of the layout's children.
`horizontalAlignment` - The horizontal alignment of the layout's children.
`content` - The content of the Column
 	
See Also: `Row`,
`androidx.compose.foundation.lazy.LazyColumn`
 	
Samples: `androidx.compose.foundation.layout.samples.SimpleColumn`
 	
`// Unresolved`
	`Column (`
	`/** { space: 20} **/`
	`) {`
	`/** content **/`
	`}.justifyContent(FlexAlign.Start)`
	`.alignItems(HorizontalAlign.Start)`

Сопоставление компонентов

Jetpack Compose	ArkUI
	

Подход сопоставления компонентов

Component A

Parameter α
place: **constructor**

Parameter β
place: **constructor**

Parameter γ
place: **constructor**

Component B

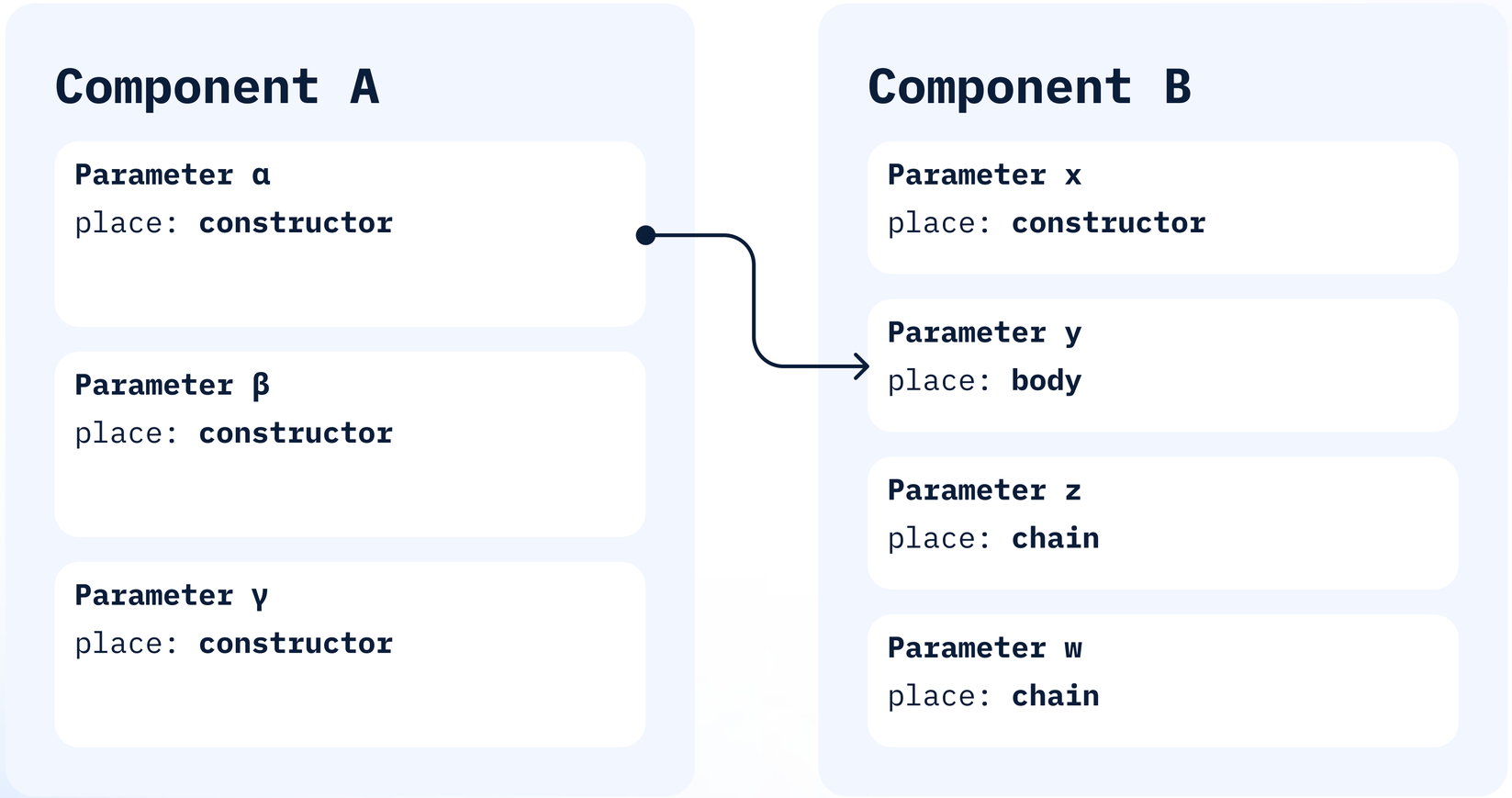
Parameter x
place: **constructor**

Parameter y
place: **body**

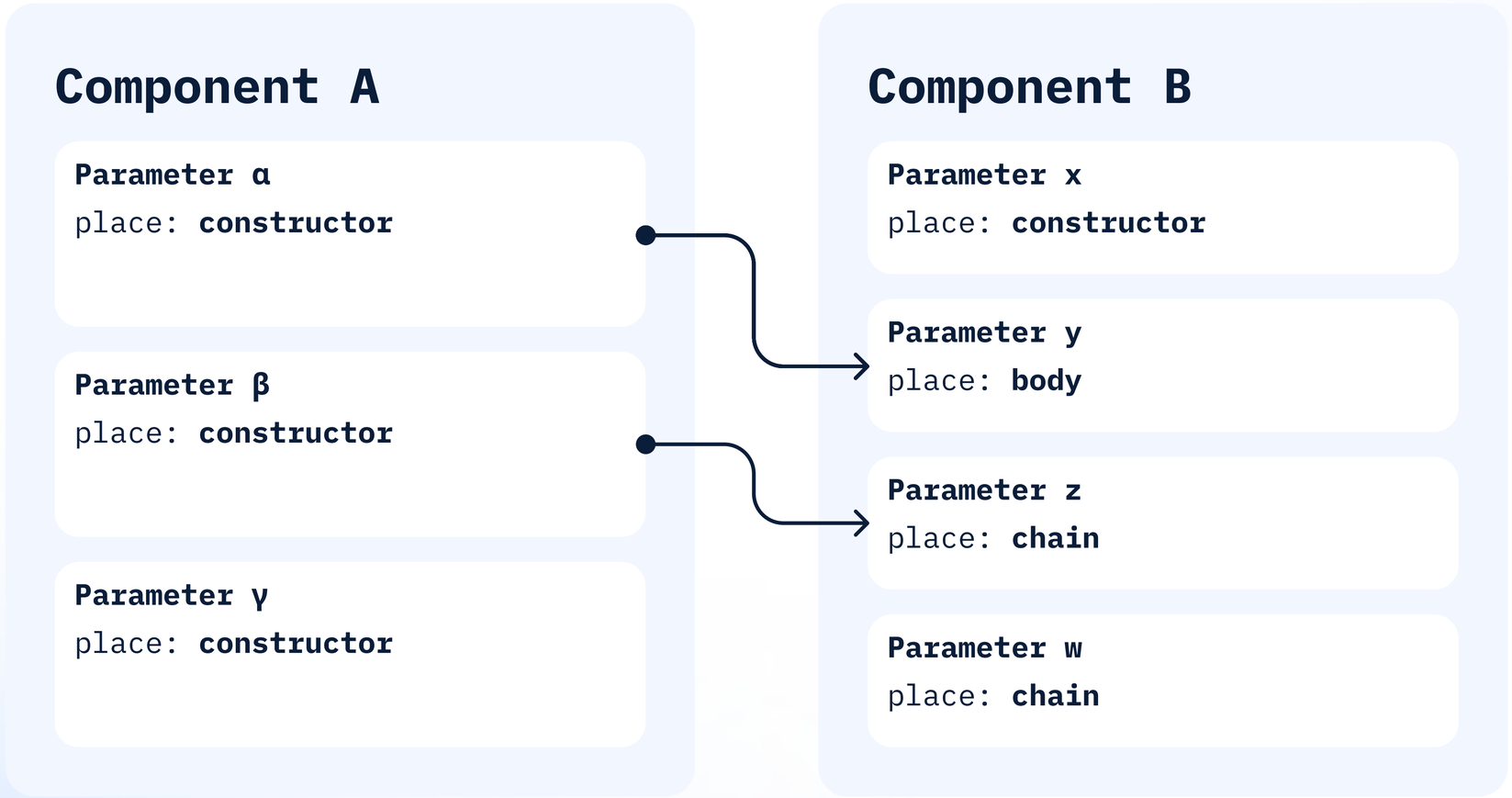
Parameter z
place: **chain**

Parameter w
place: **chain**

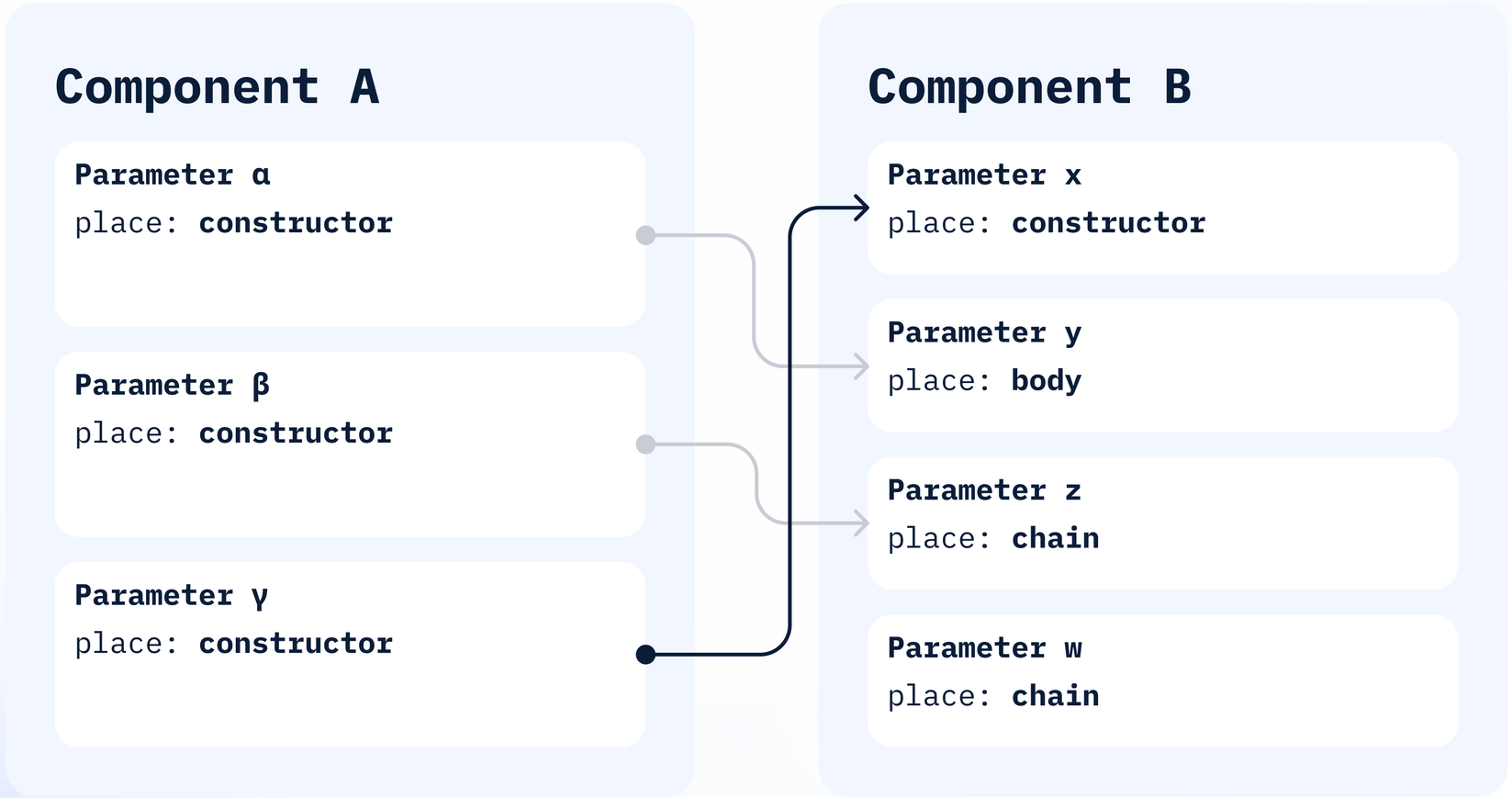
Подход сопоставления компонентов



Подход сопоставления компонентов



Подход сопоставления компонентов



Подход сопоставления компонентов

Jetpack Compose

Column

content
place: **constructor**

horizontalAlignment
place: **constructor**

verticalArrangement
place: **constructor**

ArkUI

Column

content
place: **body**

alignItems
place: **chain**

justifyContent
place: **chain**

space
place: **constructor**

```
Column (
  { space: 20 }
) {
    /** content **/
}.justifyContent(FlexAlign.
Start)
.alignItems
(HorizontalAlign.Start)
```

Шаги добавления новых компонентов

1.Указываем наименование компонента

```
internal object TextComponentRules : ComponentRuleSetProvider {

    override val ruleSets: List<ComponentRuleSet> = listOf(
        createTextRuleSet( sourceFqName = "androidx.compose.material.Text"),
    )
```

Шаги добавления новых компонентов

2.Перечисляем все его параметры

```
private fun createTextRuleSet(sourceFqName: String):
ComponentRuleSet {
    return ComponentRuleSet(
        sourceFqName = sourceFqName,
        parameterRules = listOf(
            ParameterRule(
                parameterName = "text",
                case = …,
            ),
            ParameterRule(
                parameterName = "modifier",
                case = …,
            ),
            ParameterRule(
                parameterName = "color",
                case = …,
            ),
…
```

Шаги добавления новых компонентов

3.Указываем преобразование параметров PSI элемента в ArkTS IR

```
private fun createTextRuleSet(sourceFqName: String):
ComponentRuleSet {
   return ComponentRuleSet(
       sourceFqName = sourceFqName,
       parameterRules = listOf(
           ParameterRule(
               parameterName = "text",
               case = ParameterRuleCase(
                   mutate = { input ->
RuleMutationResult.single(input.mappedExpression) },
                   writes = …,
               ),
           ),
           ParameterRule(
               parameterName = "modifier",
               case = ignoredCase(),
           ),
           ParameterRule(
               parameterName = "color",
               case = ParameterRuleCase(
                   mutate = { input ->
RuleMutationResult.single(input.mappedExpression) },
                   writes = …,
               ),
           ),
…
```

Шаги добавления новых компонентов

4.Указываем стратегию преобразования

```
parameterRules = listOf(
   ParameterRule(
       parameterName = "text",
       case = ParameterRuleCase(
           mutate = { input ->
RuleMutationResult.single(input.mappedExpression) },
           writes = listOf(
               ParameterWrite(channel =
TargetWriteChannel.ConstructorArgument(index = 0)),
           ),
       ),
   ),
   ParameterRule(
       parameterName = "modifier",
       case = ignoredCase(),
   ),
   ParameterRule(
       parameterName = "color",
       case = ParameterRuleCase(
           mutate = { input ->
RuleMutationResult.single(input.mappedExpression) },
           writes = listOf(
               ParameterWrite(channel =
TargetWriteChannel.ChainCall(methodName = CHAIN_FONT_COLOR)),
           ),
       ),
   ),
```

Шаги добавления новых компонентов

Jetpack Compose

```
@Composable
private fun Root() {
    Text(
        text = "Hello",
        color = Color.Red,
        fontSize = 18.sp,
        fontStyle = FontStyle.Italic,
        fontWeight = FontWeight.W600,
        letterSpacing = 1.5.sp,
        textAlign = TextAlign.Center,
        lineHeight = 22.sp,
        overflow = TextOverflow.Visible,
        maxLines = 3,
    )
}
```

ArkUI

```
@ComponentV2
struct Root {
  build() {
    Text("Hello")
      .fontColor(Color.Red)
      .fontSize(18)
      .fontStyle(FontStyle.Italic)
      .fontWeight(600)
      .letterSpacing(1.5)
      .textAlign(TextAlign.Center)
      .lineHeight(22)
      .textOverflow({ overflow: TextOverflow.None })
      .maxLines(3)
  }
}
```

А что по Modifier?

Modifier hoising

А что по Modifier?

Foundation's Modifier

```
@Composable
fun Foo() {
  Text(
    modifier =
Modifier.fillMaxSize()
      .padding(all = 16.dp),
    ...
  )
  ...
}
```

А что по Modifier?

Foundation's Modifier

```
@Composable
fun Foo() {
  Text(
    modifier =
Modifier.fillMaxSize()
      .padding(all = 16.dp),
    ...
  )
  ...
}
```

Hoisted Modifier

```
@Composable
fun Foo(
  customModifier: Modifier
) {
  Fee(modifier = customModifier,
...)
}

@Composable
fun Fee(
  modifier: Modifier
) {
  Text(modifier = modifier, ...)
}
```

Foundation’s Modifier hosting

Jetpack Compose	ArkUI

`@Composable`
`fun Foo() {`
`  Text(`
`  text = "Hello",`
`  modifier = Modifier`
`  .fillMaxSize()`
`  .background(Color.Red)`
`  .padding(16.dp)`
`  )`
`}`
	`@ComponentV2`
	`struct Foo {`
	`  build() {`
	`  Text("Hello")`
	`  .width("100%")`
	`  .height("100%")`
	`  .backgroundColor(Color.Red)`
	`  .padding(16)`
	`  }`
	`}`

Foundation’s Modifier hosting

Jetpack Compose	ArkUI

`@Composable`
`fun Foo() {`
`  Text(`
`  text = "Hello",`
`  modifier = Modifier`
`  .fillMaxSize()`
`  .background(Color.Red)`
`  .padding(16.dp)`
`  )`
`}`
	`@ComponentV2`
	`struct Foo {`
	`  build() {`
	`  Text("Hello")`
	`  .width("100%")`
	`  .height("100%")`
	`  .backgroundColor(Color.Red)`
	`  .padding(16)`
	`  }`
	`}`

Foundation’s Modifier hosting

Jetpack Compose	ArkUI

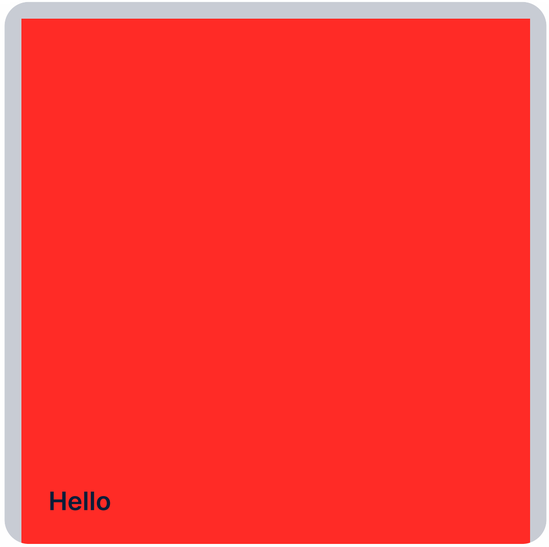
`@Composable`
`fun Foo() {`
`  Text(`
`  text = "Hello",`
`  modifier = Modifier`
`  .fillMaxSize()`
`  .background(Color.Red)`
`  .padding(16.dp)`
`  )`
`}`
	`@ComponentV2`
	`struct Foo {`
	`  build() {`
	`  Text("Hello")`
	`  .width("100%")`
	`  .height("100%")`
	`  .backgroundColor(Color.Red)`
	`  .padding(16)`
	`  }`
	`}`

Foundation’s Modifier hosting

Jetpack Compose	ArkUI

`@Composable`
`fun Foo() {`
`  Text(`
`  text = "Hello",`
`  modifier = Modifier`
`  .fillMaxSize()`
`  .background(Color.Red)`
`  .padding(16.dp)`
`  )`
`}`
	`@ComponentV2`
	`struct Foo {`
	`  build() {`
	`  Text("Hello")`
	`  .width("100%")`
	`  .height("100%")`
	`  .backgroundColor(Color.Red)`
	`  .padding(16)`
	`  }`
	`}`

Foundation’s Modifier hosting

Jetpack Compose	ArkUI
	

Foundation’s Modifier hosting

Jetpack Compose	ArkUI

`@Composable`
`fun Foo() {`
`  Text(`
`  text = "Hello",`
`  modifier = Modifier`
`  .fillMaxSize()`
`  .background(Color.Red)`
`  .padding(16.dp)`
`  )`
`}`
	`@ComponentV2`
	`struct Foo {`
	`  build() {`
	`  Text("Hello")`
	`  .width("100%")`
	`  .height("100%")`
	`  .backgroundColor(Color.Red)`
	`  .padding(16)`
	`  .align(Alignment.TopStart)`
	`  }`
	`}`

Foundation’s Modifier hosting

Hoisted Modifier’s hosting

```
@Composable
fun Foo() {
    Fee(
        customModifier = Modifier

.clip(RoundedCornerShape(20.dp))

.background(Color(0xFF6C63FF))
    )
}
```

Hoisted Modifier’s hosting

```
@Composable
fun Foo() {
    Fee(
        customModifier = Modifier

.clip(RoundedCornerShape(20.dp))

.background(Color(0xFF6C63FF))
    )
}
```

```
@Composable
fun Fee(
    customModifier: Modifier
) {
    Box(
        modifier = customModifier
            .padding(24.dp),
        contentAlignment =
Alignment.Center
    ) {
        Text(
            text = "Hoisted Modifier",
            color = Color.White
        )
    }
}
```

Hoisted Modifier’s hosting

```
@Composable
fun Foo() {
    Fee(
        customModifier = Modifier

.clip(RoundedCornerShape(20.dp))

.background(Color(0xFF6C63FF))
    )
}
```

Hoisted Modifier

```
@Composable
fun Fee(
    customModifier: Modifier
) {
    Box(
        modifier = customModifier
            .padding(24.dp),
        contentAlignment =
Alignment.Center
    ) {
        Text(
            text = "Hoisted Modifier",
            color = Color.White
        )
    }
}
```

AttributeModifier (ArkUI)

```
/**
 * Defines the attribute modifier.
 *
 * @interface AttributeModifier<T>
 * @syscap SystemCapability.ArkUI.ArkUI.Full
 * @crossplatform
 * @atomicservice
 * @since 12
 */
declare interface AttributeModifier<T> {

  /**
   * Defines the normal update attribute function.
   *
   * @param { T } instance
   * @syscap SystemCapability.ArkUI.ArkUI.Full
   * @crossplatform
   * @atomicservice
   * @since 12
   */
  applyNormalAttribute?(instance: T): void;
}
```

https://developer.huawei.com/consumer/en/doc/harmonyos-guides/arkts-user-defined-extension-attributemodifier

AttributeModifier (ArkUI)

```
// demo.ets

import { MyButtonModifier } from './button_modifier'

@Entry
@Component
struct attributeDemo {
  // The modifier is decorated with @State, with behavior
consistent with that of a regular object.
  @State modifier: MyButtonModifier = new
MyButtonModifier(true);

  build() {
    Row() {
      Column() {
        Button("Button")
          .attributeModifier(this.modifier)
          .onClick(() => {
            // When the level-1 attribute of the modifier is
changed, a UI update is triggered, causing applyNormalAttribute
to be executed again.
            this.modifier.isDark = !this.modifier.isDark
          })
      }
    }
      .width('100%')
    }
    .height('100%')
  }
}
```

```
// button_modifier.ets
export class MyButtonModifier implements
AttributeModifier<ButtonAttribute> {
  isDark: boolean = false

  constructor(dark?: boolean) {
    this.isDark = dark ?? false
  }

  applyNormalAttribute(instance: ButtonAttribute): void {
    if (this.isDark) {
      instance.backgroundColor('#707070')
    } else {
      instance.backgroundColor('#17A98D')
        .borderColor('#707070')
        .borderWidth(2)
    }
  }
}
```

CommonModifier

```
/**
 * Defines Common Modifier
 *
 * @extends CommonAttribute
 * @implements AttributeModifier<CommonAttribute>
 * @syscap SystemCapability.ArkUI.ArkUI.Full
 * @atomicservice
 * @since 12
 */
export declare class CommonModifier extends CommonAttribute implements AttributeModifier<CommonAttribute> {
  /**
   * Defines the normal update attribute function.
   *
   * @param { CommonAttribute } instance
   * @syscap SystemCapability.ArkUI.ArkUI.Full
   * @crossplatform
   * @atomicservice
   * @since 12
   */
  applyNormalAttribute?(instance: CommonAttribute): void;
}
```

Hoisted Modifier’s hosting

Pt. 1

```
@Composable
fun Foo() {
    Fee(
        customModifier = Modifier
            .clip(RoundedCornerShape(20.dp))
            .background(Color(0xFF6C63FF))
    )
}
```

```
@ComponentV2
struct Foo {
    build() {
        Fee({...})
    }
}
```

Hoisted Modifier’s hosting

Pt. 2

```
@Composable
fun Foo() {
    Fee(
        customModifier = Modifier
            .clip(RoundedCornerShape(20.dp))
            .background(Color(0xFF6C63FF))
    )
}
```

```
@ComponentV2
struct Foo {
    build() {
        Fee({
            customModifier: {...}
        })
    }
}
```

Hoisted Modifier’s hosting

Pt. 3

```
@Composable
fun Foo() {
    Fee(
        customModifier = Modifier
            .clip(RoundedCornerShape(20.dp))
            .background(Color(0xFF6C63FF))
    )
}
```

```
@ComponentV2
struct Foo {
    build() {
        Fee({
            customModifier: {
                applyNormalAttribute: (instance:
CommonAttribute) => {
                    ...
                }
            } as CommonModifier
        })
    }
}
```

Hoisted Modifier’s hosting

Pt. 4

```
@Composable
fun Foo() {
    Fee(
        customModifier = Modifier
            .clip(RoundedCornerShape(20.dp))
            .background(Color(0xFF6C63FF))
    )
}
```

```
@ComponentV2
struct Foo {
    build() {
        Fee({
            customModifier: {
                applyNormalAttribute: (instance:
CommonAttribute) => {
                    instance.borderRadius(20)
                    instance.clip(true)
                }
            } as CommonModifier
        })
    }
}
```

Hoisted Modifier’s hosting

Pt. 5

```
@Composable
fun Foo() {
    Fee(
        customModifier = Modifier
            .clip(RoundedCornerShape(20.dp))
            .background(Color(0xFF6C63FF))
    )
}
```

```
@ComponentV2
struct Foo {
    build() {
        Fee({
            customModifier: {
                applyNormalAttribute: (instance:
CommonAttribute) => {
                    instance.borderRadius(20)
                    instance.clip(true)
                    instance.backgroundColor(0xFF6C63FF)
                }
            } as CommonModifier
        })
    }
}
```

Hoisted Modifier’s hosting

“Foo()” final

```
@Composable
fun Foo() {
    Fee(
        customModifier = Modifier
            .clip(RoundedCornerShape(20.dp))
            .background(Color(0xFF6C63FF))
    )
}
```

```
@ComponentV2
struct Foo {
    build() {
        Fee({
            customModifier: {
                applyNormalAttribute: (instance:
CommonAttribute) => {
                    instance.borderRadius(20)
                    instance.clip(true)
                    instance.backgroundColor(0xFF6C63FF)
                }
            } as CommonModifier
        })
    }
}
```

Hoisted Modifier’s hosting

Pt.1

Jetpack Compose

```
@Composable
fun Fee(
    customModifier: Modifier
) {
    Box(
        modifier = customModifier
            .padding(24.dp),
        contentAlignment = Alignment.Center
    ) {
        Text(
            text = "Hoisted Modifier",
            color = Color.White
        )
    }
}
```

ArkUI

```
@ComponentV2
struct Fee {

    build() {
        Stack({ alignContent: Alignment.Center }) {
            Text("Hoisted Modifier")
                .fontColor(Color.White)
        }
        .padding(24)
    }
}
```

Hoisted Modifier’s hosting

Pt.2

Jetpack Compose

```
@Composable
fun Fee(
    customModifier: Modifier
) {
    Box(
        modifier = customModifier
            .padding(24.dp),
        contentAlignment = Alignment.Center
    ) {
        Text(
            text = "Hoisted Modifier",
            color = Color.White
        )
    }
}
```

ArkUI

```
@ComponentV2
struct Fee {
    @Param
    @Require
    customModifier: CommonModifier;

    build() {
        Stack({ alignContent: Alignment.Center }) {
            Text("Hoisted Modifier")
                .fontColor(Color.White)
        }
        .padding(24)
    }
}
```

Hoisted Modifier’s hosting

Pt.3

Jetpack Compose

```
@Composable
fun Fee(
    customModifier: Modifier
) {
    Box(
        modifier = customModifier
            .padding(24.dp),
        contentAlignment = Alignment.Center
    ) {
        Text(
            text = "Hoisted Modifier",
            color = Color.White
        )
    }
}
```

ArkUI

```
@ComponentV2
struct Fee {
    @Param
    @Require
    customModifier: CommonModifier;

    build() {
        Stack({ alignContent: Alignment.Center }) {
            Text("Hoisted Modifier")
                .fontColor(Color.White)
        }
        .attributeModifier(this.customModifier)
        .padding(24)
    }
}
```

Hoisted Modifier’s hosting

“Fee()” final

Jetpack Compose

```
@Composable
fun Fee(
    customModifier: Modifier
) {
    Box(
        modifier = customModifier
            .padding(24.dp),
        contentAlignment = Alignment.Center
    ) {
        Text(
            text = "Hoisted Modifier",
            color = Color.White
        )
    }
}
```

ArkUI

```
@ComponentV2
struct Fee {
    @Param
    @Require
    customModifier: CommonModifier;

    build() {
        Stack({ alignContent: Alignment.Center }) {
            Text("Hoisted Modifier")
                .fontColor(Color.White)
        }
        .attributeModifier(this.customModifier)
        .padding(24)
    }
}
```

Hoisted Modifier’s hosting

Jetpack Compose

```
@Composable
fun Foo() {
    Fee(
        customModifier = Modifier
            .clip(RoundedCornerShape(20.dp))
            .background(Color(0xFF6C63FF))
    )
}
```

```
@Composable
fun Fee(
    customModifier: Modifier
) {
    Box(
        modifier = customModifier
            .padding(24.dp),
        contentAlignment = Alignment.Center
    ) {
        Text(
            text = "Hoisted Modifier",
            color = Color.White
        )
    }
}
```

ArkUI

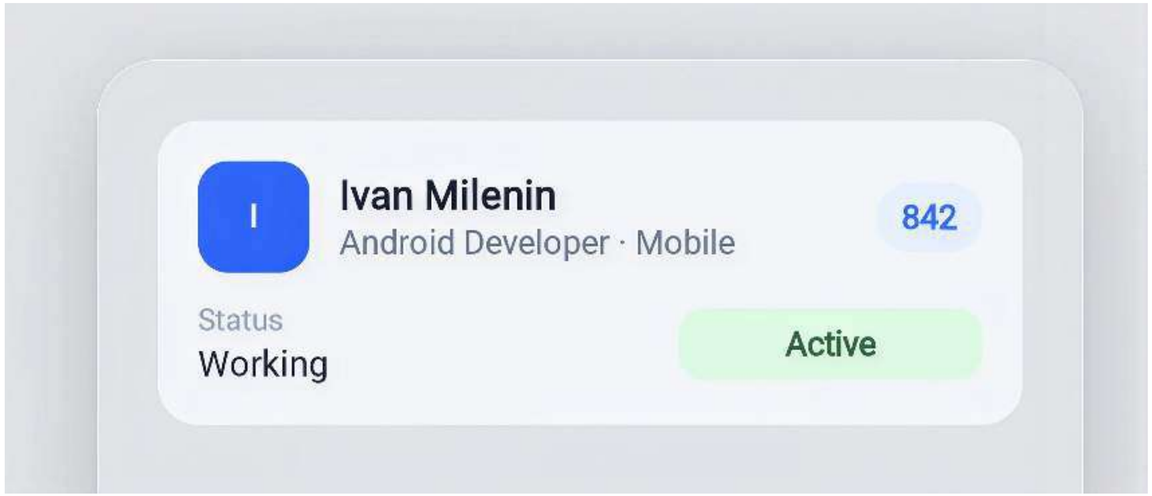
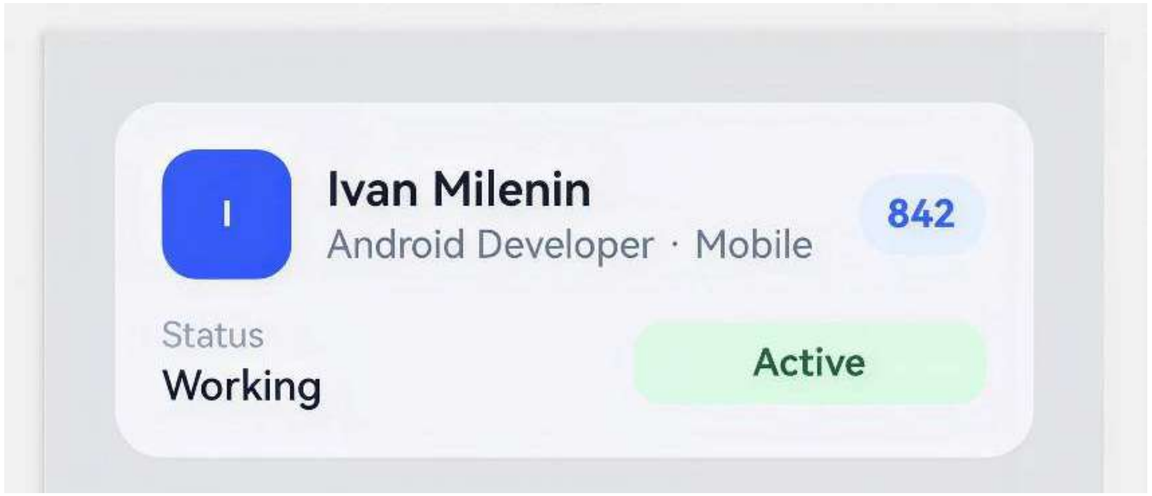
```
@ComponentV2
struct Foo {
    build() {
        Fee({
            customModifier: {
                applyNormalAttribute: (instance: CommonAttribute) => {
                    instance.borderRadius(20)
                    instance.clip(true)
                    instance.backgroundColor(0xFF6C63FF)
                }
            } as CommonModifier
        })
    }
}
```

```
@ComponentV2
struct Fee {
    @Param
    @Require
    customModifier: CommonModifier;

    build() {
        Stack({ alignContent: Alignment.Center }) {
            Text("Hoisted Modifier")
                .fontColor(Color.White)
        }
        .attributeModifier(this.customModifier)
        .padding(24)
    }
}
```

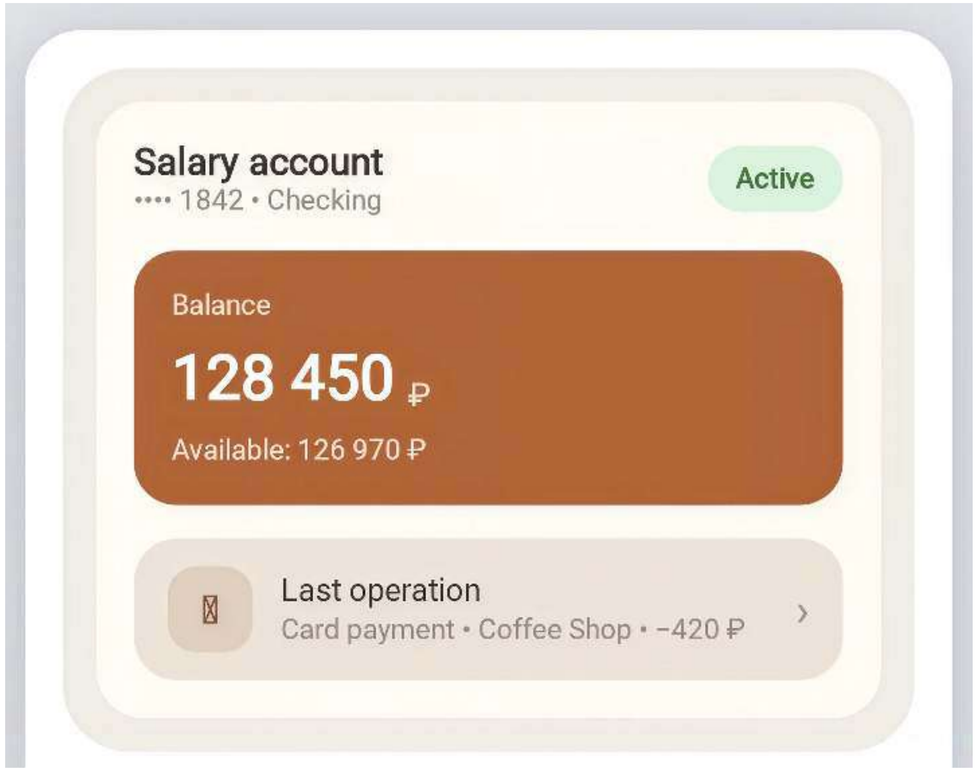
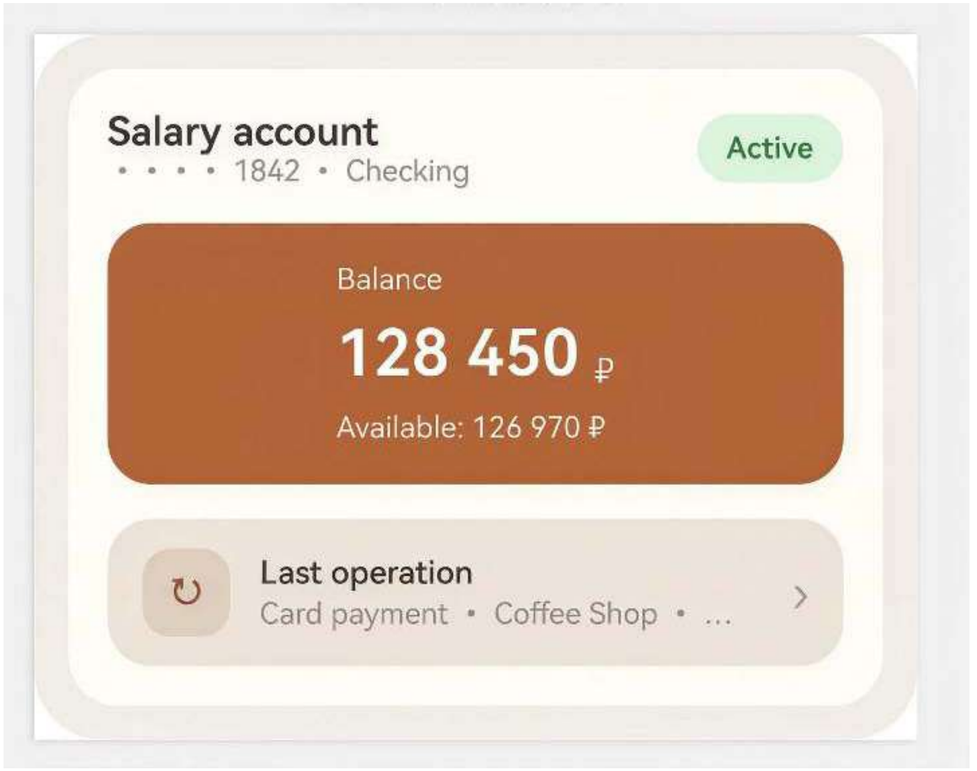
Hoisted Modifier’s hosting

Jetpack Compose	ArkUI

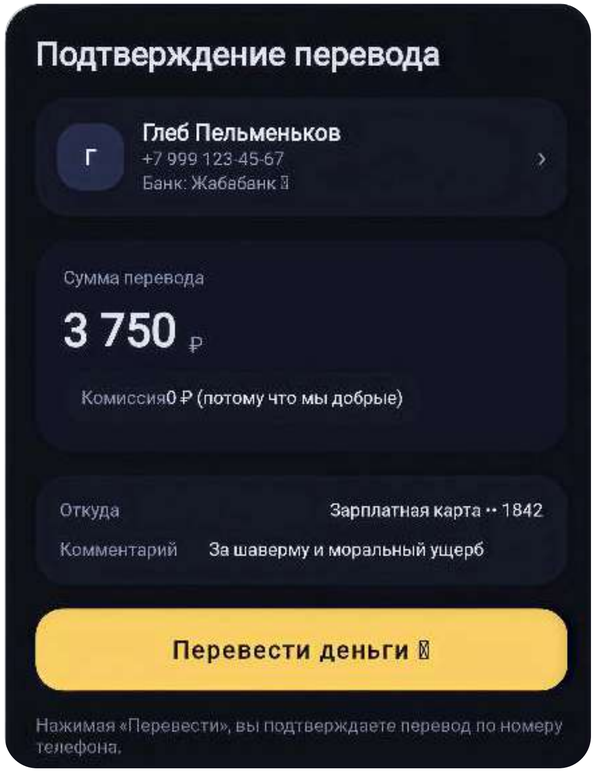
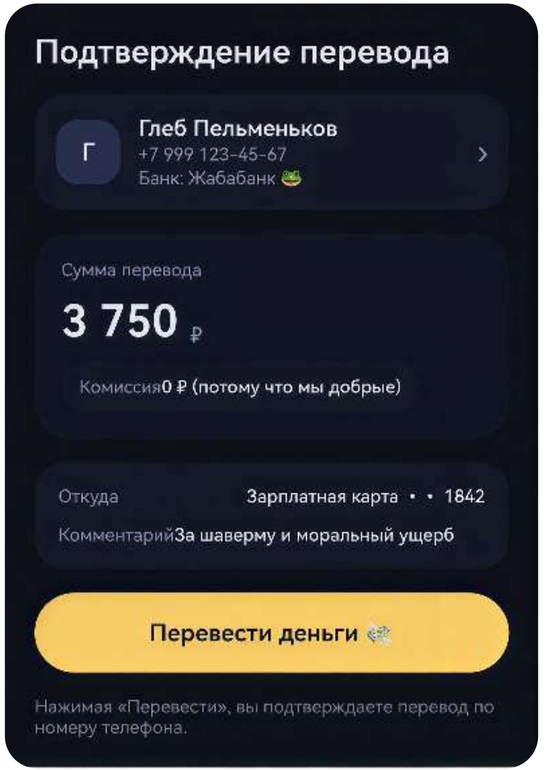
Пример реального использования. Employee card

Jetpack Compose	ArkUI
	

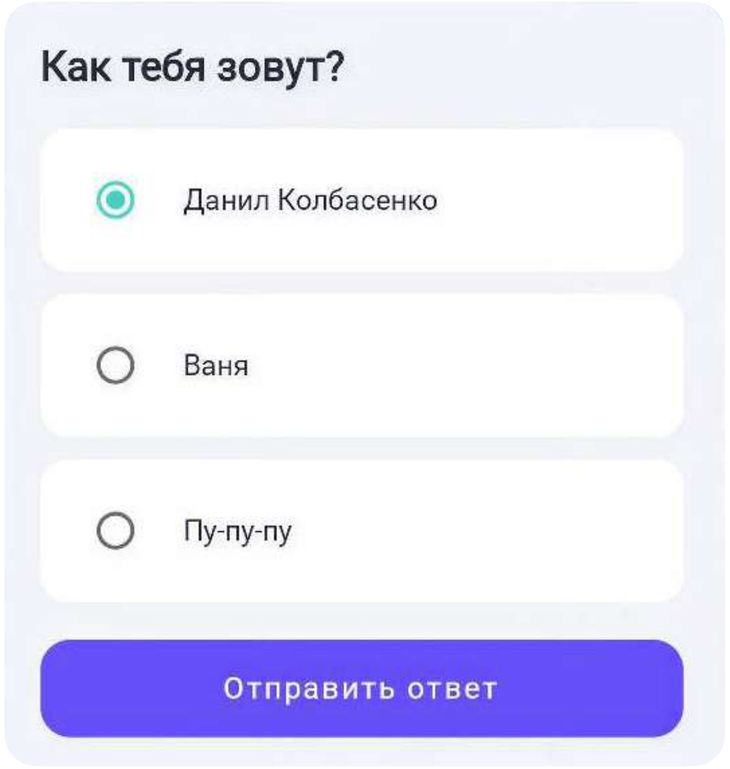
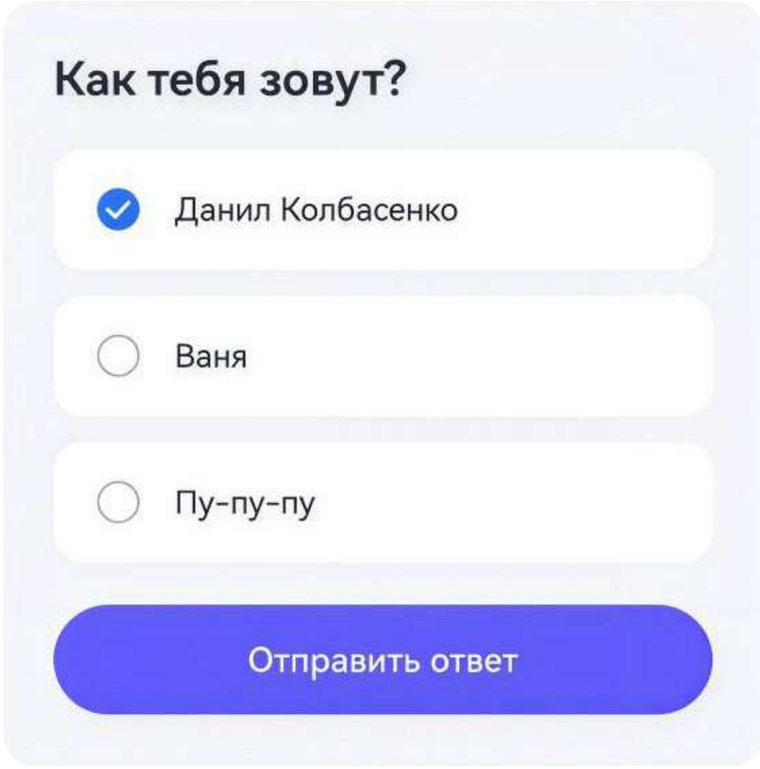
Пример реального использования.
Bank profile header

Jetpack Compose	ArkUI
	

Пример реального использования. Подтверждение перевода

Jetpack Compose	ArkUI
	

Пример реального использования. Подтверждение перевода

Jetpack Compose	ArkUI
	

⚡⚡⚡ UI — ВСЁ

План доклада

1. Контекст и актуальность задачи
2. Обзор существующих решений
3. Постановка задачи и общий подход
4. **Трансляция бизнес-логики Kotlin в ArkTS**
5. Библиотеки, формальные спецификации, SMT
6. UI: Jetpack Compose в ArkUI
7. **Прототип и результаты**

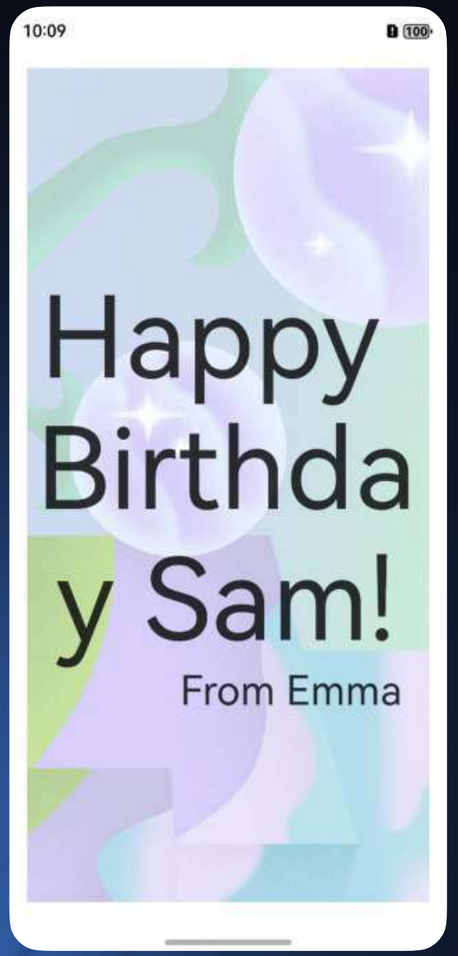
ИТМО ЛИПТСОФТ

Что получилось сделать

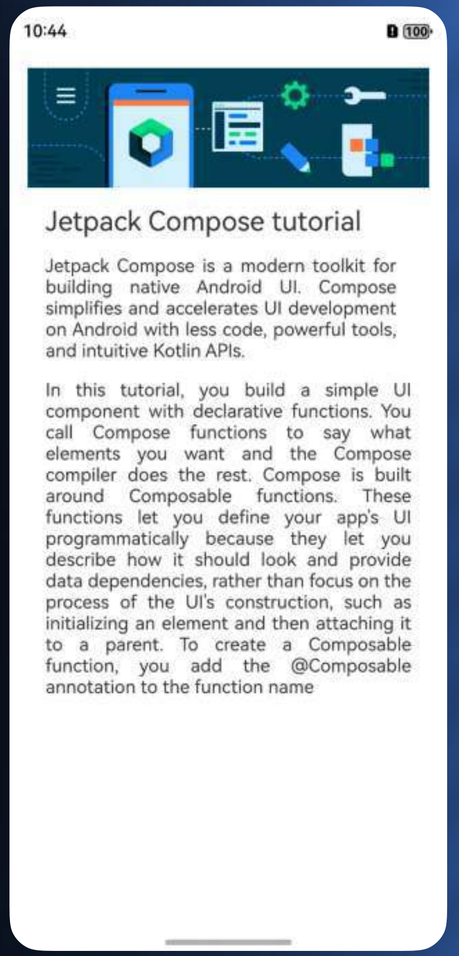
Апробировано приложений	**9** **5 — Google Android Basics,** **1 — GitHub,** **3 — синтезированные)**
Транспилировано деклараций	**53 из 67 (79%)**
Входной код (Kotlin)	**2587 строк, 41 файл**
Выходной код (ArkTS)	**1989 строк, 41 файл**

Примеры приложения

dice-roller

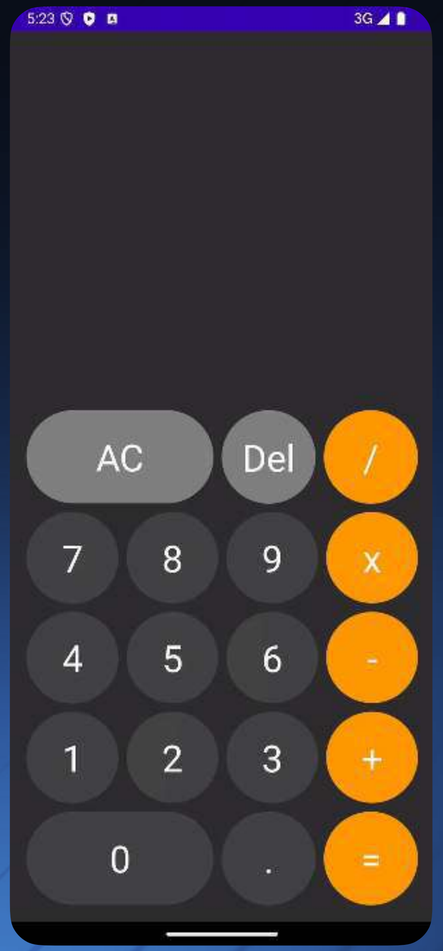


birthday card

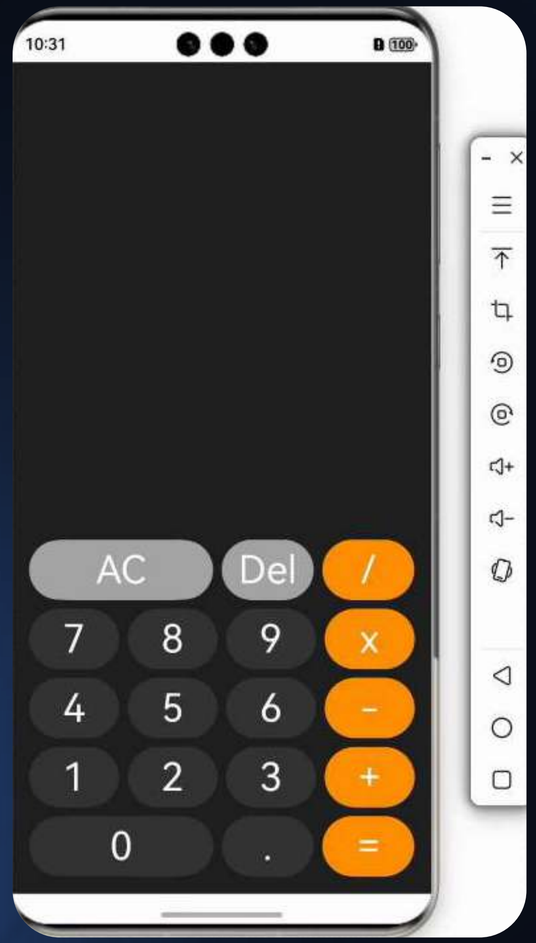


article

Портирование простого приложения



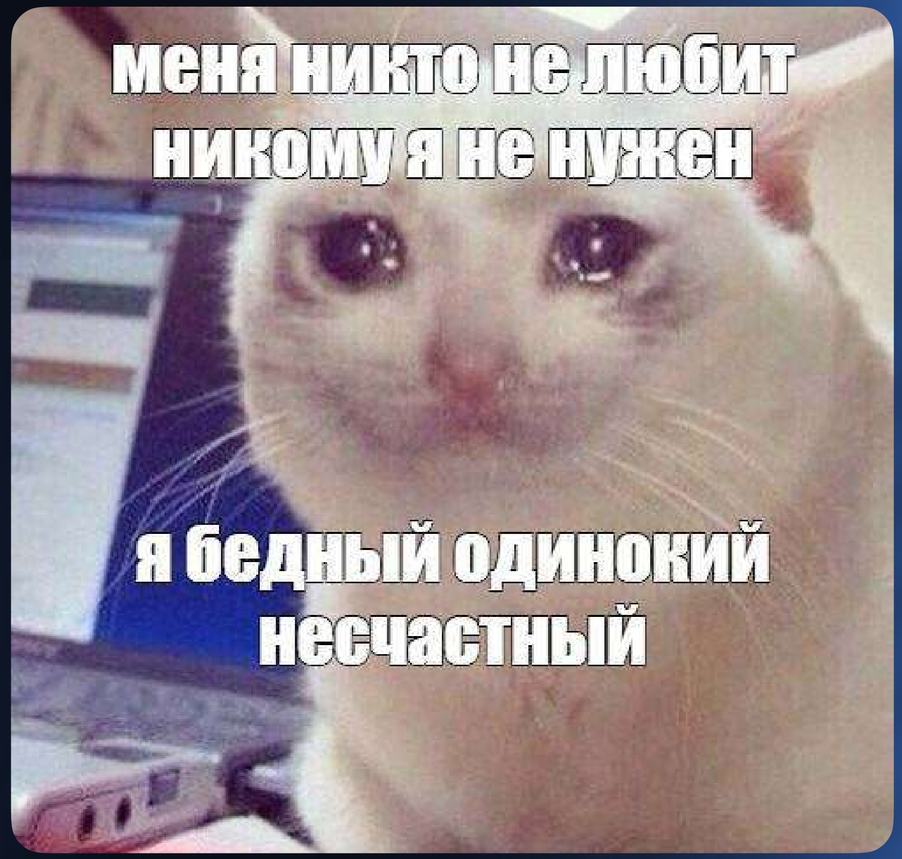
Android



HarmonyOS NEXT

Ограничения портированных приложений

Они **одностраничные**
и **без навигации**



Почему не 100%?

Апробировано приложений	9 5 – Google Android Basics, 1 – GitHub, 3 – синтезированные)
Транспилировано деклараций	**53 из 67 (79%)**
Входной код (Kotlin)	
Выходной код (ArkTS)	

```
// File: MainActivity.kt

// TODO: failed to transpile 'MainActivity': Unknown operator type: SUPER_EXPRESSION | text: super
// class MainActivity : ComponentActivity() {
//     override fun onCreate(savedInstanceState: Bundle?) {
//         super.onCreate(savedInstanceState)
//         setContent {
//             ComposeArticleTheme {
//                 // A surface container using the 'background' color from the theme
//                 Surface(
//                     modifier = Modifier.fillMaxSize(),
//                     color = MaterialTheme.colorScheme.background
//                 ) {
//                     ComposeArticleApp()
//                 }
//             }
//         }
//     }
// }
```

План доклада

1. Контекст и актуальность задачи
2. Обзор существующих решений
3. Постановка задачи и общий подход
4. **Трансляция бизнес-логики Kotlin в ArkTS**
5. Библиотеки, формальные спецификации, SMT
6. UI: Jetpack Compose в ArkUI
7. Прототип и результаты
8. **ИИ?**

ИТМО ЛИПТСОФТ

Что там по нейроночкам?

License: arXiv.org perpetual non-exclusive license
arXiv:2412.13693v3 [cs.SE] 12 Jan 2025

UITrans: Seamless UI Translation from Android to HarmonyOS

Lina Gong

Nanjing University of Aeronautics and Astronautics Nanjing China
gonglina@nuaa.edu.cn
0000-0002-5272-6706

Chen Wang

Nanjing University of Aeronautics and Astronautics Nanjing China
wangchen712@nuaa.edu.cn

Yujun Huang

Nanjing University of Aeronautics and Astronautics Nanjing China
yujunhuang@nuaa.edu.cn

Di Cui

Xidian University Xi'an China
cuidi@xidian.edu.cn

XML -> ArkUI

Example: Differences between Android XML and ArkUI

Layout
Component
Attribute

Android XML:

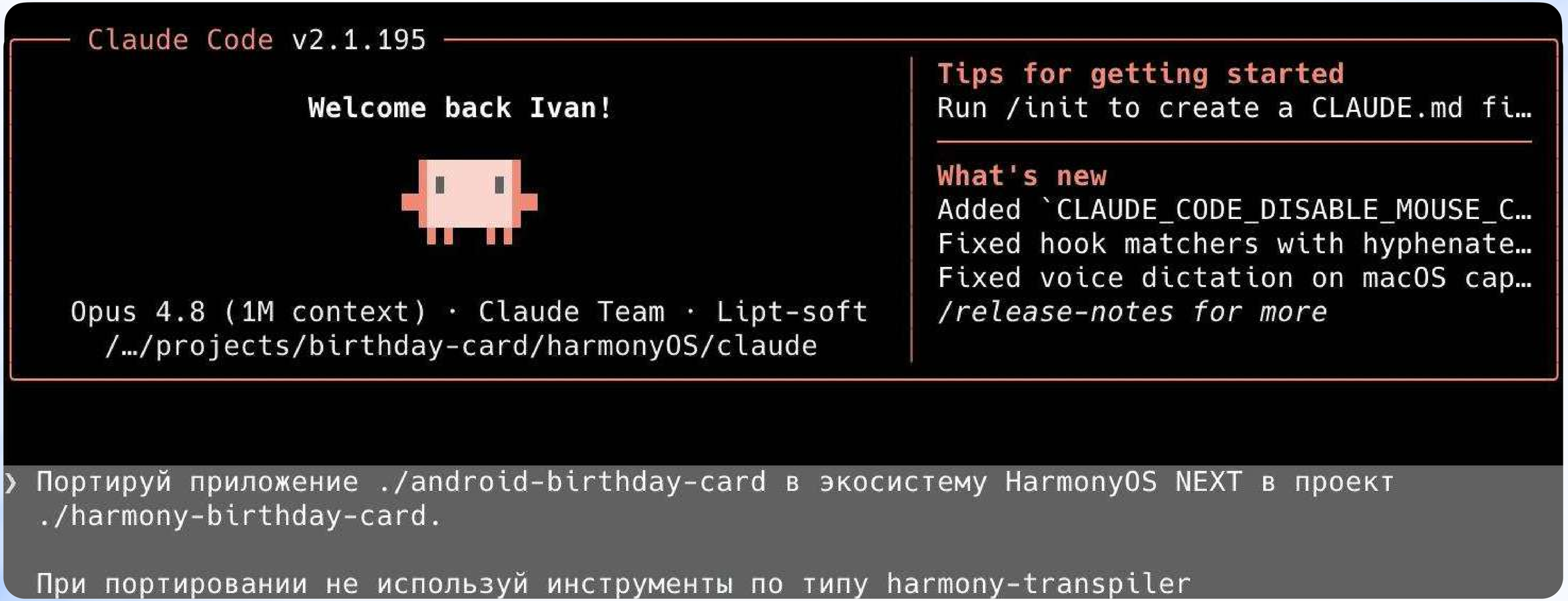
```
<RelativeLayout
  <androidx.appcompat.widget.AppCompatCheckBox
    android:id="@+id/myCheckBox"
    ...
    android:text="A" />
</RelativeLayout>
```

ArkUI:

```
Flex() {
  Row() {
    Checkbox()
    Text('A')
    ...
  }
  .id('myCheckBox')
}
```

Дисклеймер

Попытка №1


Claude Code v2.1.195
Welcome back Ivan!
Opus 4.8 (1M context) · Claude Team · Lipt-soft
/…/projects/birthday-card/harmonyOS/claude
Tips for getting started
Run /init to create a CLAUDE.md fi…
What's new
Added `CLAUDE_CODE_DISABLE_MOUSE_C…
Fixed hook matchers with hyphenate…
Fixed voice dictation on macOS cap…
/release-notes for more
❯ Портируй приложение ./android-birthday-card в экосистему HarmonyOS NEXT в проект
./harmony-birthday-card.
При портировании не используй инструменты по типу harmony-transpiler

Попытка №1

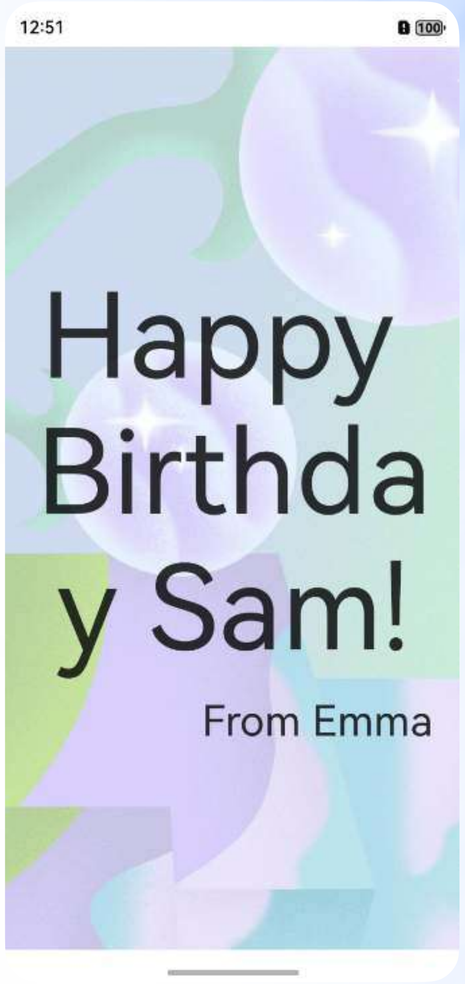
Что сделано

Портировал Android-приложение «Happy Birthday» в HarmonyOS NEXT-проект harmony-birthday-card вручную (без harmony-transpiler), заменив сломанные стаб-файлы транспайлера на корректный ArkTS/ArkUI.

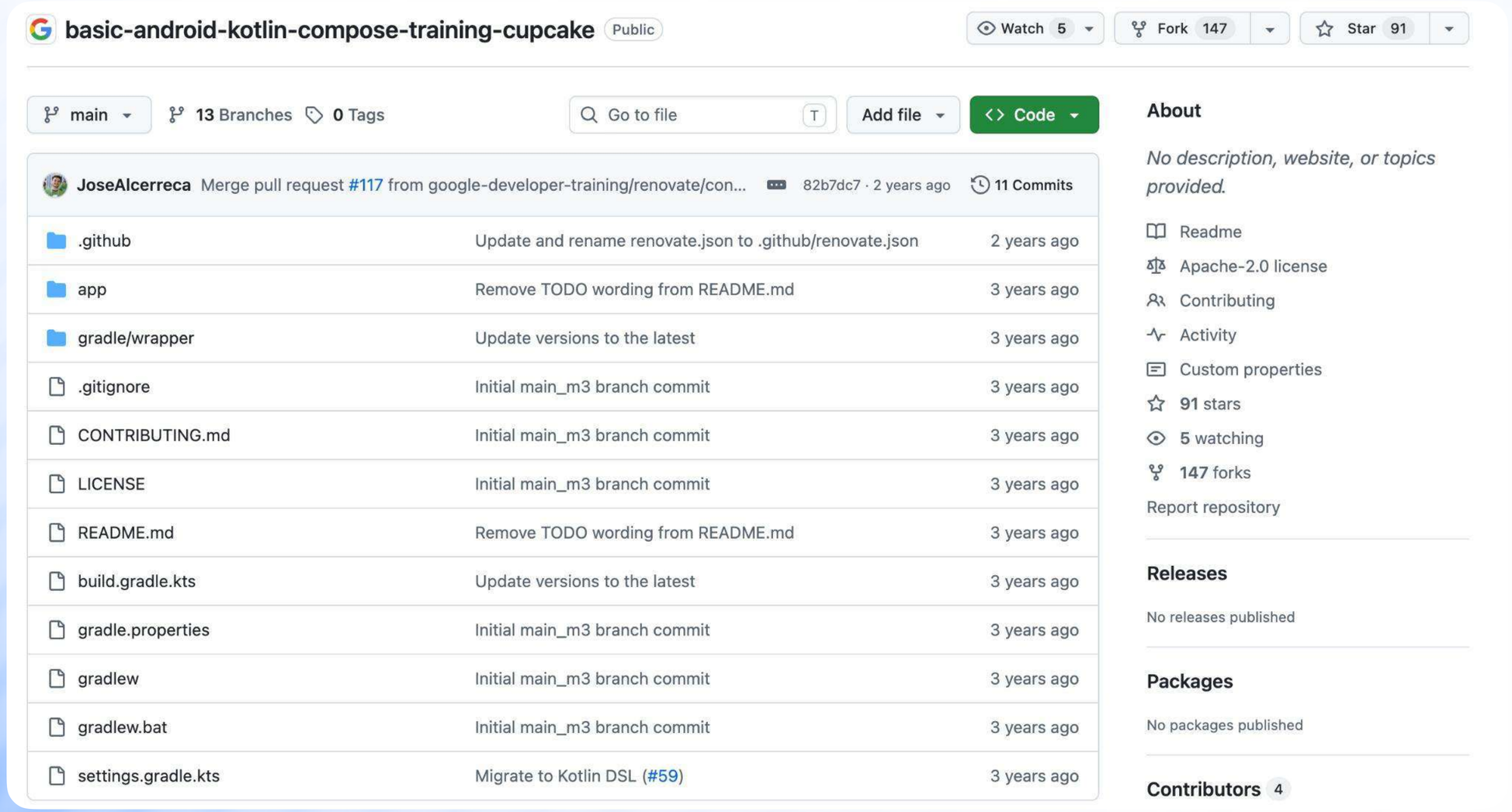
entry/src/main/ets/pages/Index.ets — основной экран. Соответствие Compose → ArkUI:
- GreetingImage (Box с наложением) → Stack
- GreetingText (Column) → @Builder GreetingText с Column
- Image(painterResource(androidparty), ContentScale.Crop, alpha=0.5f) → Image($r('app.media.androidparty')).objectFit(ImageFit.Cover).opacity(0.5)
- Два Text с fontSize 100/116 lineHeight, TextAlign.Center, и 36 с выравниванием Alignment.End → .alignSelf(ItemAlign.End); отступы padding→margin
- Arrangement.Center → .justifyContent(FlexAlign.Center)

Строковые ресурсы (entry/.../base/element/string.json) — добавил happy_birthday_text («Happy Birthday Sam!») и signature_text («From Emma»), на которые ссылается UI через $r('app.string.…').

Тема — заменил TODO-стабы на валидный ArkTS:
- Color.ets — константы цветов (Color(0xAARRGGBB) → ARGB hex-строки)
- Type.ets — Typography/TextStyle как ArkTS-интерфейсы
- Theme.ets — ColorScheme + happyBirthdayColorScheme(darkTheme) вместо MaterialTheme. Динамический цвет Android (dynamicLight/DarkColorScheme) — то, на чём споткнулся транспайлер — не имеет аналога в HarmonyOS и осознанно убран (помечено комментарием).



Попытка №2



basic-android-kotlin-compose-training-cupcake Public

Watch 5 | Fork 147 | Star 91

main | 13 Branches | 0 Tags | Go to file T | Add file | Code

JoseAlcerreca Merge pull request #117 from google-developer-training/renovate/con... 82b7dc7 · 2 years ago 11 Commits

.github	Update and rename renovate.json to .github/renovate.json	2 years ago
app	Remove TODO wording from README.md	3 years ago
gradle/wrapper	Update versions to the latest	3 years ago
.gitignore	Initial main_m3 branch commit	3 years ago
CONTRIBUTING.md	Initial main_m3 branch commit	3 years ago
LICENSE	Initial main_m3 branch commit	3 years ago
README.md	Remove TODO wording from README.md	3 years ago
build.gradle.kts	Update versions to the latest	3 years ago
gradle.properties	Initial main_m3 branch commit	3 years ago
gradlew	Initial main_m3 branch commit	3 years ago
gradlew.bat	Initial main_m3 branch commit	3 years ago
settings.gradle.kts	Migrate to Kotlin DSL (#59)	3 years ago

About

No description, website, or topics provided.

Readme
Apache-2.0 license
Contributing
Activity
Custom properties
91 stars
5 watching
147 forks
Report repository

Releases

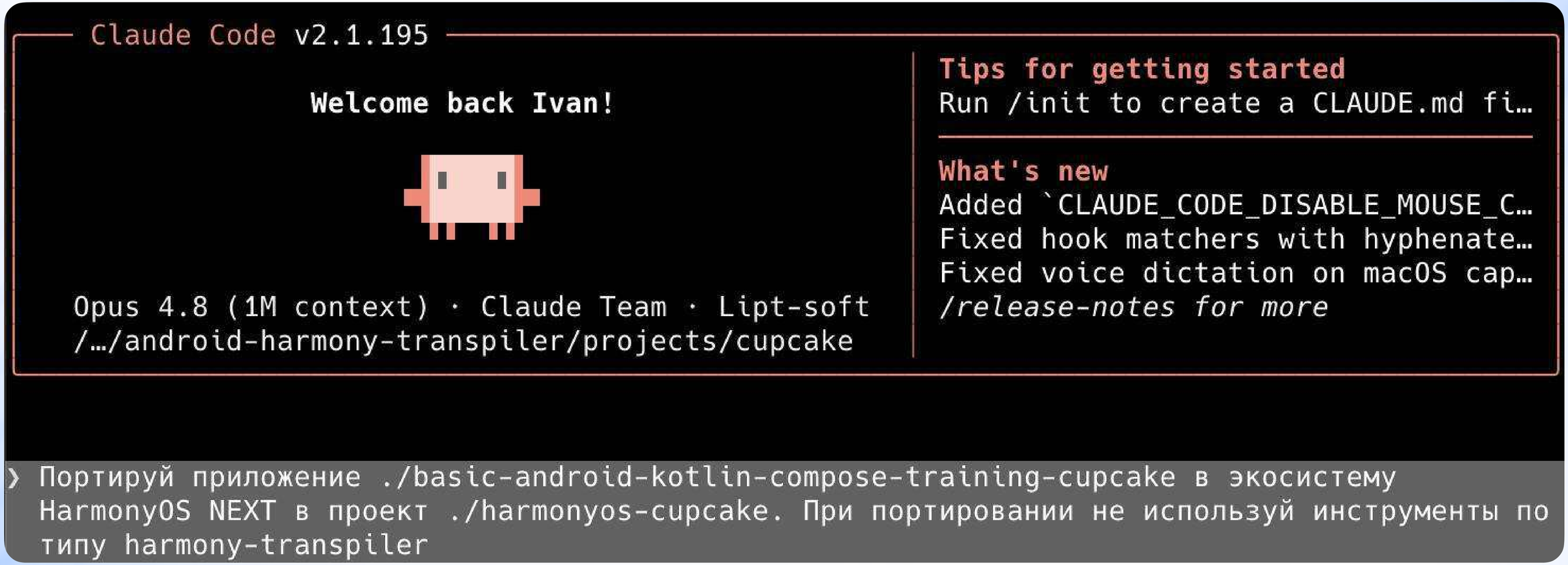
No releases published

Packages

No packages published

Contributors 4

Попытка №2


Claude Code v2.1.195
Welcome back Ivan!
Opus 4.8 (1M context) · Claude Team · Lipt-soft
/…/android-harmony-transpiler/projects/cupcake
Tips for getting started
Run /init to create a CLAUDE.md fi…
What's new
Added `CLAUDE_CODE_DISABLE_MOUSE_C…
Fixed hook matchers with hyphenate…
Fixed voice dictation on macOS cap…
/release-notes for more
› Портируй приложение ./basic-android-kotlin-compose-training-cupcake в экосистему HarmonyOS NEXT в проект ./harmonyos-cupcake. При портировании не используй инструменты по типу harmony-transpiler

Попытка №2

Я вручную портировал приложение **Cupcake** из Android (Kotlin/Jetpack Compose) в **HarmonyOS NEXT** (ArkTS/ArkUI, Stage model) — без использования harmony-transpiler. Код написан идиоматично под ArkUI.

Что сделано

Структура (harmonyos-cupcake/entry/src/main/ets/):
- pages/Index.ets — навигационный хост (Navigation + NavPathStack + @Builder navDestination) и связывание экранов — аналог CupcakeScreen.kt и MainActivity
- viewmodel/OrderViewModel.ets — состояние заказа, расчёт цены и генерация дат
- model/DataSource.ets, model/OrderUiState.ets — вкусы, варианты количества, форма состояния
- view/StartOrderScreen.ets, SelectOptionScreen.ets (общий для Flavor и Pickup), SummaryScreen.ets
- components/CommonUi.ets — FormattedPriceLabel
- util/OrderShare.ets — системный шаринг через @kit.ShareKit

Портируй приложение ./basic-android-kotlin-compose-training-cupcake в экосистему …
16 files changed +919

Попытка №2

```
> hvigor ERROR: Failed :entry:default@CompileArkTS...
> hvigor ERROR: ArkTS Compiler Error
1 ERROR: ArkTS:ERROR File: /Users/m.milenin/Android-Apps/work/CLAUDE/kmp-bspb-demo-claude/claude
 -workspace/android-harmony-transpiler/projects/cupcake/harmonyos-cupcake/entry/src/main/ets/view
 /SelectOptionScreen.ets:60:24
 Argument of type 'Resource' is not assignable to parameter of type 'string | number'.


2 ERROR: ArkTS:ERROR File: /Users/m.milenin/Android-Apps/work/CLAUDE/kmp-bspb-demo-claude/claude
 -workspace/android-harmony-transpiler/projects/cupcake/harmonyos-cupcake/entry/src/main/ets/view
 /SummaryScreen.ets:77:26
 Argument of type 'Resource' is not assignable to parameter of type 'string | number'.


COMPILE RESULT:FAIL {ERROR:3}
> hvigor ERROR: BUILD FAILED in 1 s 637 ms

Process finished with exit code -1
```

Попытка №2.5

LLM — это не «чёрный ящик», который генерирует код из головы. Это агент, который вызывает реальные утилиты и собирает из них результат.

Описанный транспайлер — часть подхода, а не замена ему

Попытка №2.5. Скилл

```
---
name: transpile
description: "Transpile Kotlin/Compose source files to HarmonyOS NEXT ArkTS. Use when user wants to convert
Android/Kotlin code to HarmonyOS, port Compose UI to ArkUI, or transpile Kotlin snippets to ArkTS."
arguments: [input]
allowed-tools: Bash(*) Read Write Glob Grep
---

# Transpile Kotlin/Compose to HarmonyOS NEXT ArkTS

You are a transpilation assistant. Your job is to run harmony-transpiler on user-provided
Kotlin/Compose code and deliver the resulting ArkTS output.

## Prerequisites
- JDK 21 on PATH

## Setup
The transpiler binary lives at
`${CLAUDE_SKILL_DIR}/../../../app/build/install/harmony-transpiler/bin/harmony-transpiler`.
...
TRANSPILER="$(cd "${CLAUDE_SKILL_DIR}/../../.." && pwd)/app/build/install/harmony-transpiler/bin/harmony-transpiler"

## CLI modes
1. Inline:  "$TRANSPILER" --code 'fun foo() {}'
2. Stdin:   echo 'fun foo() {}' | "$TRANSPILER" --stdin
3. Files:   "$TRANSPILER" <input-path>... [--output-dir <dir>] [--android-res <res-dir>]
...
```

Попытка №2.5

How it went

I ran harmony-transpiler over all 12 Kotlin files (--android-res wired up so R.* references resolved). The transpiler handled the **leaf pieces** cleanly — the OrderUiState data class, DataSource, and the simple components (CupcakeAppBar, SelectQuantityButton, FormattedPriceLabel) — but emitted [warn] blocks for the bulk of the app, where it hit constructs it doesn't yet support:

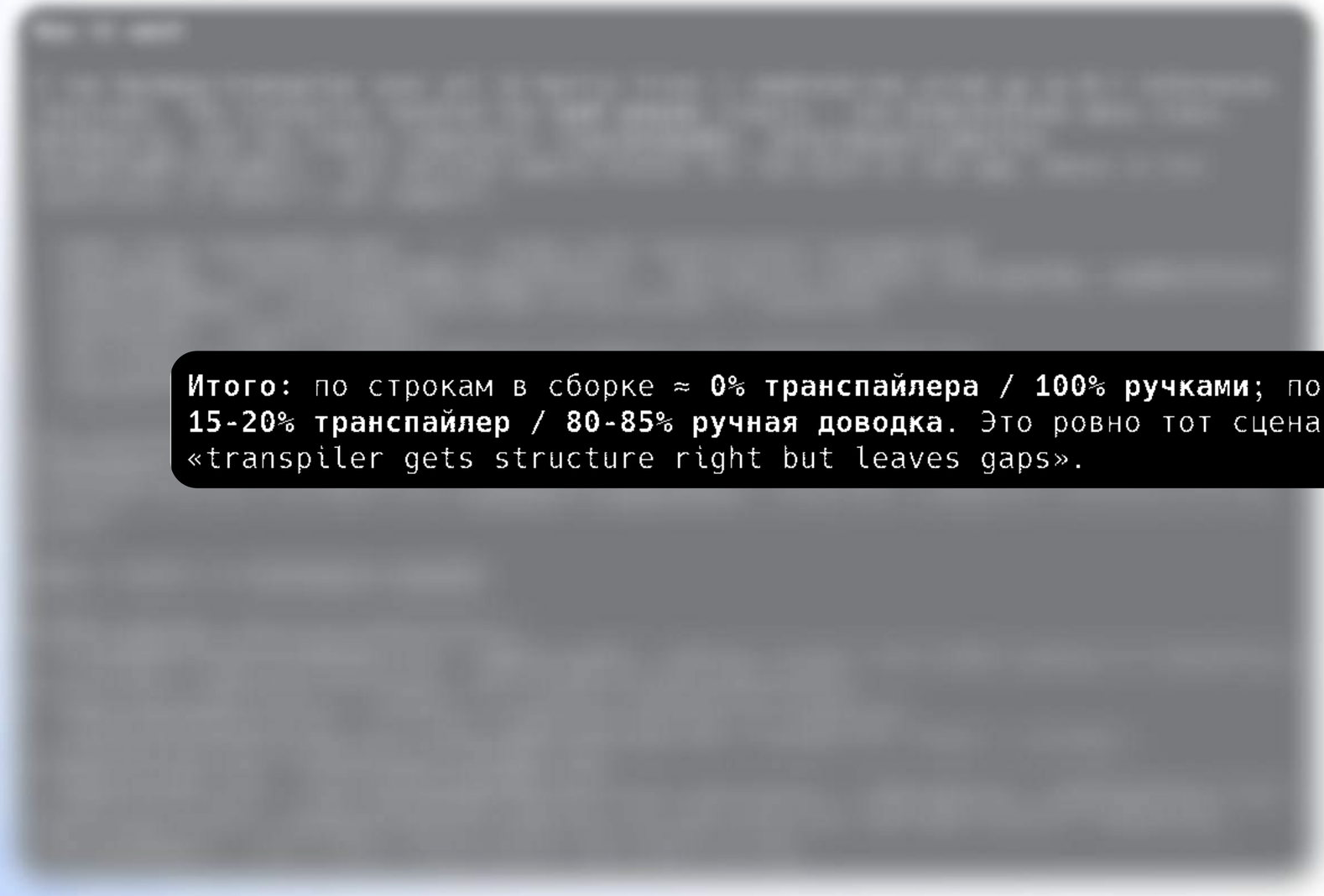
- enum class CupcakeScreen(...) — enums with constructors unsupported
- CupcakeApp / cancelOrderAndNavigateToStart — Navigation Compose (navigateUp, popBackStack)
- OrderViewModel — pickupOptions()[0] array-access + StateFlow
- shareOrder — Android Intent
- the three screens — dimensionResource/padding and getQuantityString
- CupcakeTheme — dynamic color scheme

The emitted output also targeted a Compose-shim runtime (TopAppBar, MaterialTheme, CommonModifier, stringResource) that doesn't exist in HarmonyOS. So per the skill's step 5, I finished the port by hand into **idiomatic ArkUI/ArkTS**, using the transpiler structure as the guide.

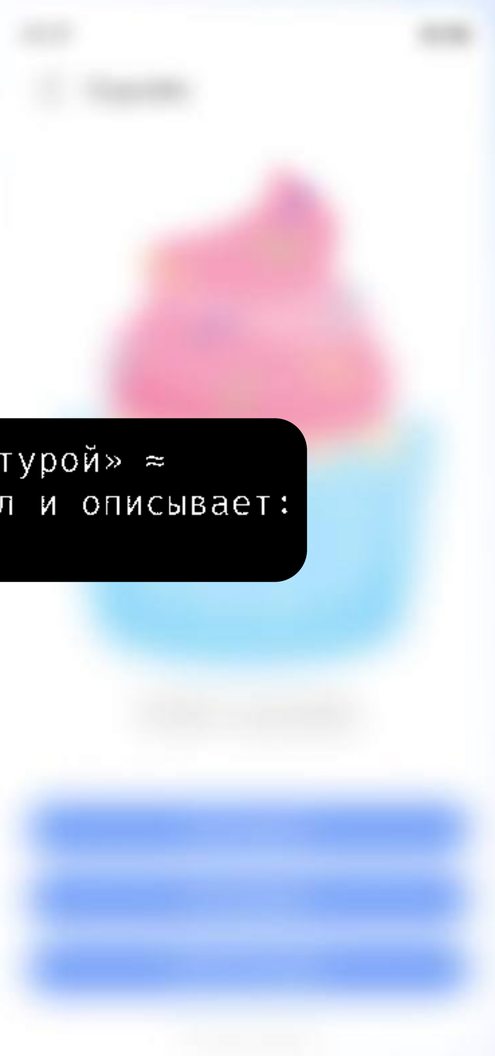
What I built in harmonyos-cupcake

ArkTS sources (entry/src/main/ets/):
- viewmodel/OrderViewModel.ets — @ObservedV2 + @Trace fields (the ArkUI analog of StateFlow); price calc, same-day surcharge, and 4-date pickup generation
- model/DataSource.ets — flavors + quantity options as resources
- view/StartOrderScreen.ets, SelectOptionScreen.ets (reused for flavor + pickup), SummaryScreen.ets, FormattedPriceLabel.ets
- pages/Index.ets — the CupcakeApp/MainActivity equivalent: a Navigation + NavPathStack host where each route's NavDestination supplies its own title bar and back button (replacing CupcakeAppBar), plus order-reset/cancel and share wiring

Попытка №2.5



Итого: по строкам в сборке ≈ **0% транспайлера / 100% ручками**; по «помощи со структурой» ≈ **15-20% транспайлер / 80-85% ручная доводка**. Это ровно тот сценарий, который скилл и описывает: «transpiler gets structure right but leaves gaps».



Итог подхода

AI + LibSL = ❤️

О чем вообще было выступление?

1

Есть не только ИИ

О чем вообще было выступление?

1

Есть не только ИИ

2

Будущее — в связке, а не в выборе

О чем вообще было выступление?

1

Есть не только ИИ

2

Будущее — в связке, а не в выборе

4

Перенос ПО может быть доказуемо корректным

О чем вообще было выступление?

1

Есть не только ИИ

2

Будущее — в связке, а не в выборе

4

Перенос ПО может быть доказуемо корректным

5

Иногда и математика полезна

SMT

1. Портирование библиотек

2. Миграция между версиями API

2. Тестирование

SMT

Что сделали

- Портирование базовых конструкций Kotlin в ArkTS
- Портирование множества компонентов Jetpack Compose в ArkUI
- Портирование вызовов библиотек при помощи формальных спецификаций

Что планируем сделать

- Портирование навигации
- Портирование архитектуры проекта (многомодульность, директории и прочее)
- Портирование системных вызовов

ИТМО ЛИПТСОФТ

Время вопросов

Иван Миленин
Мобильный разработчик

MILKAsuper

svakun@gmail.com

MILKA-TOP