Figma + Compose. Упрощаем верстку

Анна Жаркова
Lead Mobile developer

Обо мне

- В мобильной разработке с 2013
- Ведущий мобильный разработчик в Usetech
- Нативная разработка под iOS и Android (Swift/Objective-C, Kotlin/Java) кросс-платформа (Xamarin, Kotlin multiplatform)
- Ментор, управляю командой направления
- Спикер на конференциях AppsConf, Mobius, TechTrain, DroidCon (2022)
- Преподаватель в Otus (iOS Pro и базовый)
- Автор статей по мобильной разработке (SwiftUI, iOS, KMM)
- Эксперт Skillbox

mb Experts

USETECH

Обсудим:

- Compose, декларативность
- Подключение и настройка Relay
- Создание компонентов Figma
- Генерация кода Compose
- Стили, донастройки, интерактивность

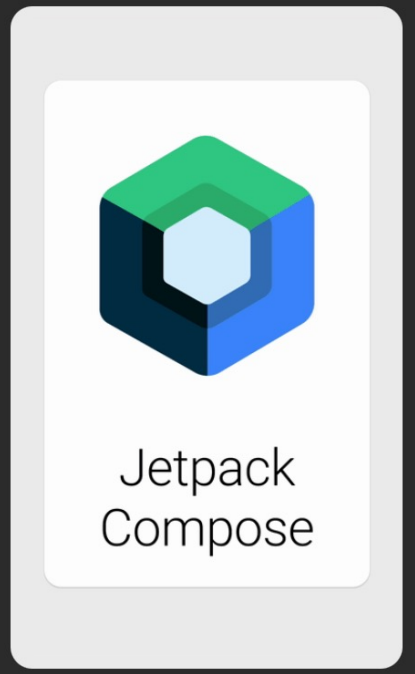
Android UI

- Xml
- Jetpack Compose

Jetpack Compose

Представлен в мае 2019

```
@Composable
fun JetpackCompose() {
    Card {
        var expanded by remember { mutableStateOf(false) }
        Column(Modifier.clickable { expanded = !expanded }) {
            Image(painterResource(R.drawable.jetpack_compose))
            AnimatedVisibility(expanded) {
                Text(
                    text = "Jetpack Compose",
                    style = MaterialTheme.typography.h2,
                )
            }
        }
    }
}
```

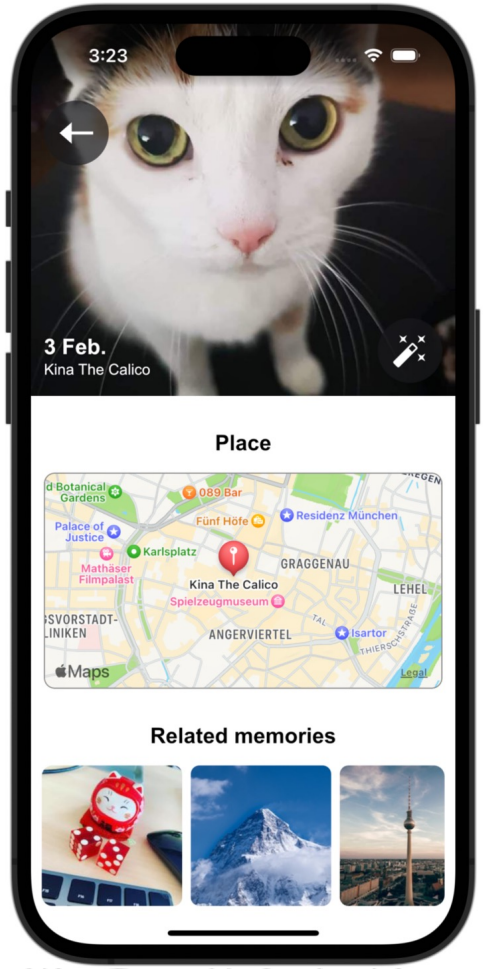


Jetpack Compose

- View – State
- Декларативная навигация
- @Composable функции
- Под капотом стандартные контролы

Compose iOS

https://blog.jetbrains.com/kotlin/2023/05/compose-multiplatform-for-ios-is-in-alpha/



UIKit view embedded in Compose

Compose view embedded in SwiftUI

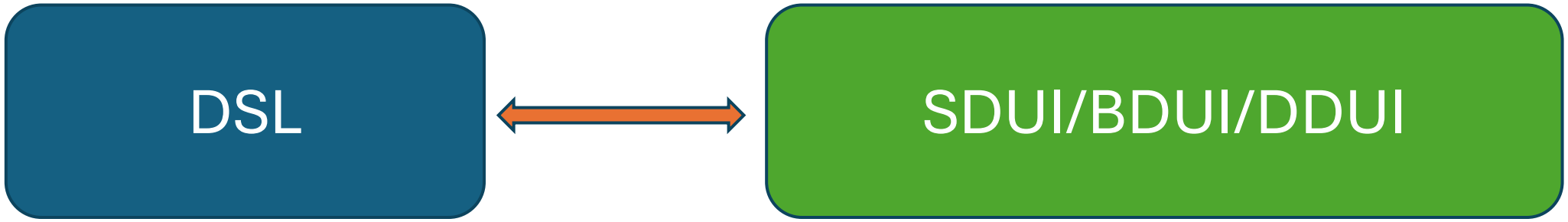
Floating SwiftUI TextField

Compose iOS

Построено на Jetpack Compose

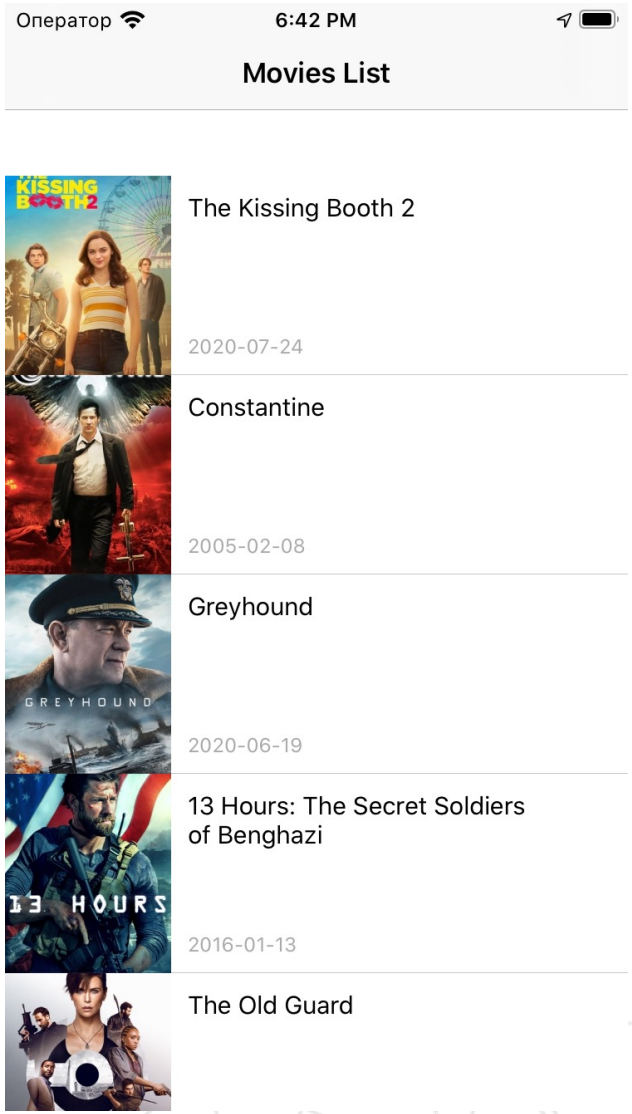
```
@Composable
fun App() {
    MaterialTheme {
        var showImage by remember { mutableStateOf(false) }
        Column(Modifier.fillMaxWidth(), horizontalAlignment
         = Alignment.CenterHorizontally) {
            Button(onClick = {
                showImage = !showImage
            }) {
                Text("Toggle image")
            }
            AnimatedVisibility(showImage) {
                Image(
                    painterResource("compose-multiplatform.xml"),
                    "Compose Multiplatform Logo"
                )
            }
        } }
```

DSL Compose



SDUI/BDUI/DDUI

```
{
"presentation": {
  "title": "Movies List"
  "type": "Toolbar"
  "children": {
    "type": "List"
    "content" : {
    "items": {
      "type": "row"
      "content": {
       ///...
      }
    }
    }
  }
}
```



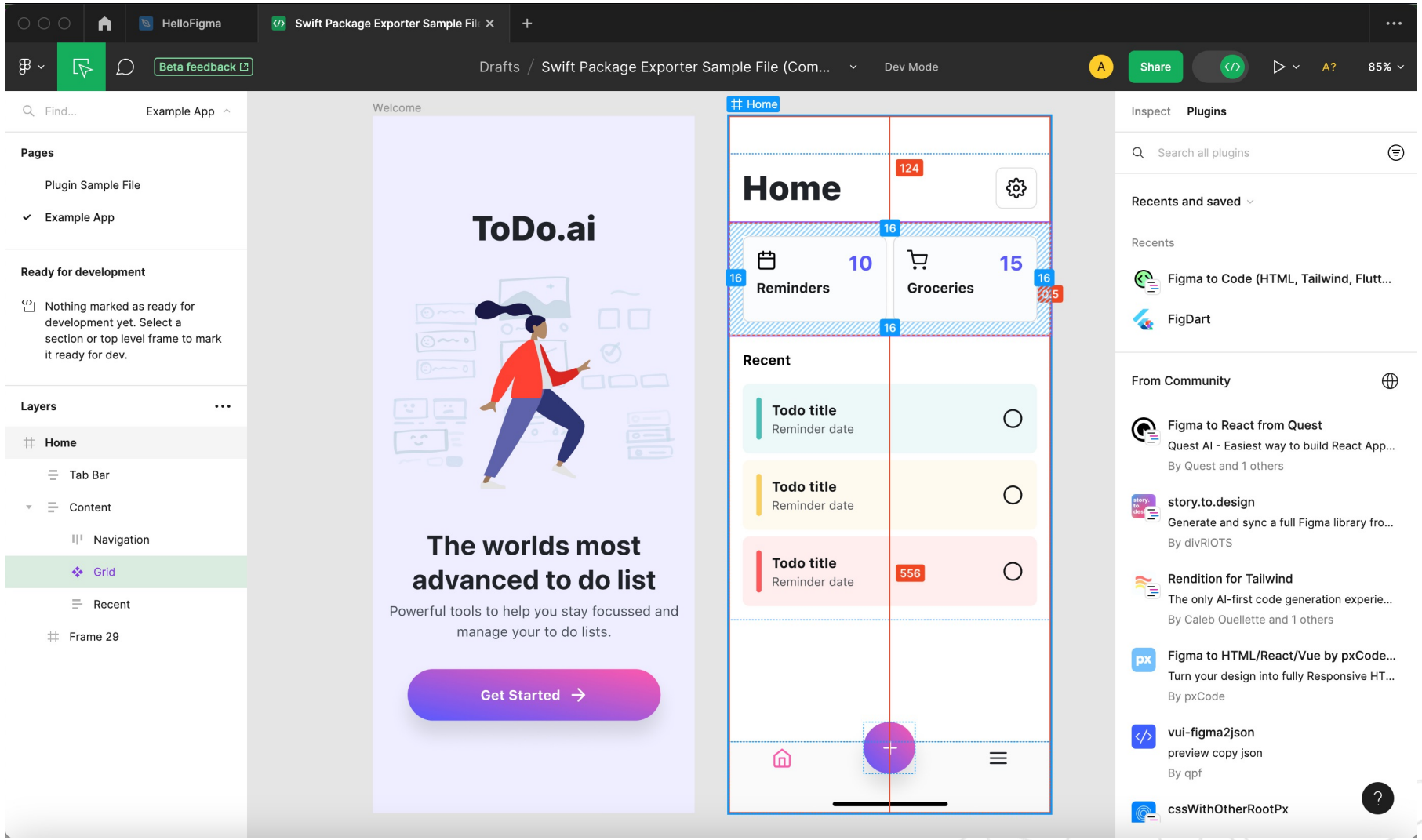
SDUI/BDUI/DDUI

SDUI/BDUI/DDUI. Вопросы

- Как сделать, чтобы UI выглядел +/- одинаково
- Оптимизация создания UI

Выход есть. Нам потребуется

- Figma (https://www.figma.com)
- Плагины Figma

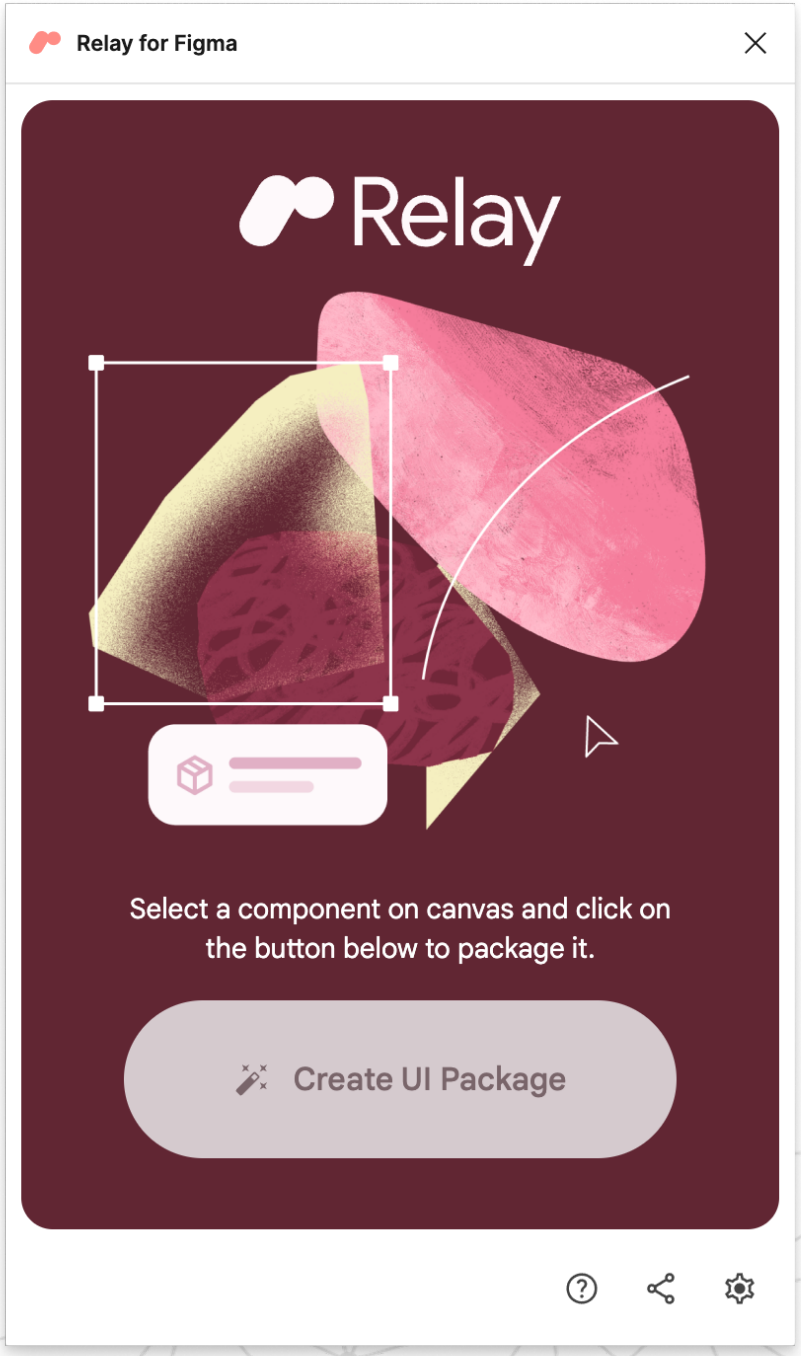


Compose + Relay

Relay

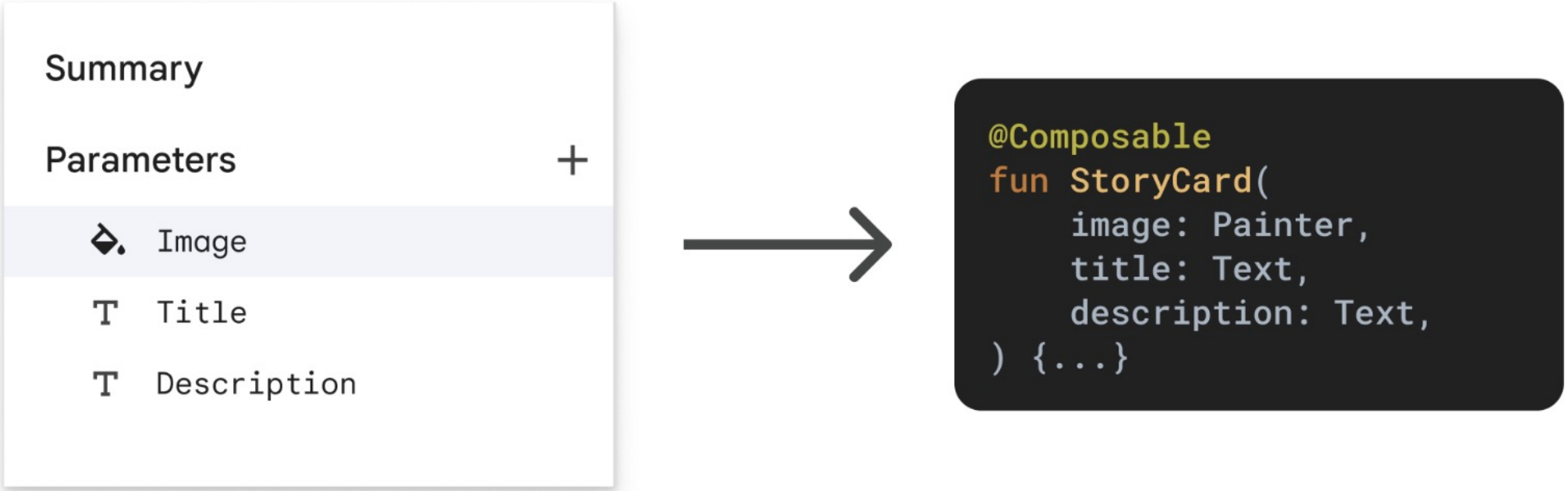
https://relay.material.io/

https://developer.android.com/jetpack/compose/tooling/relay



Relay

https://relay.material.io/

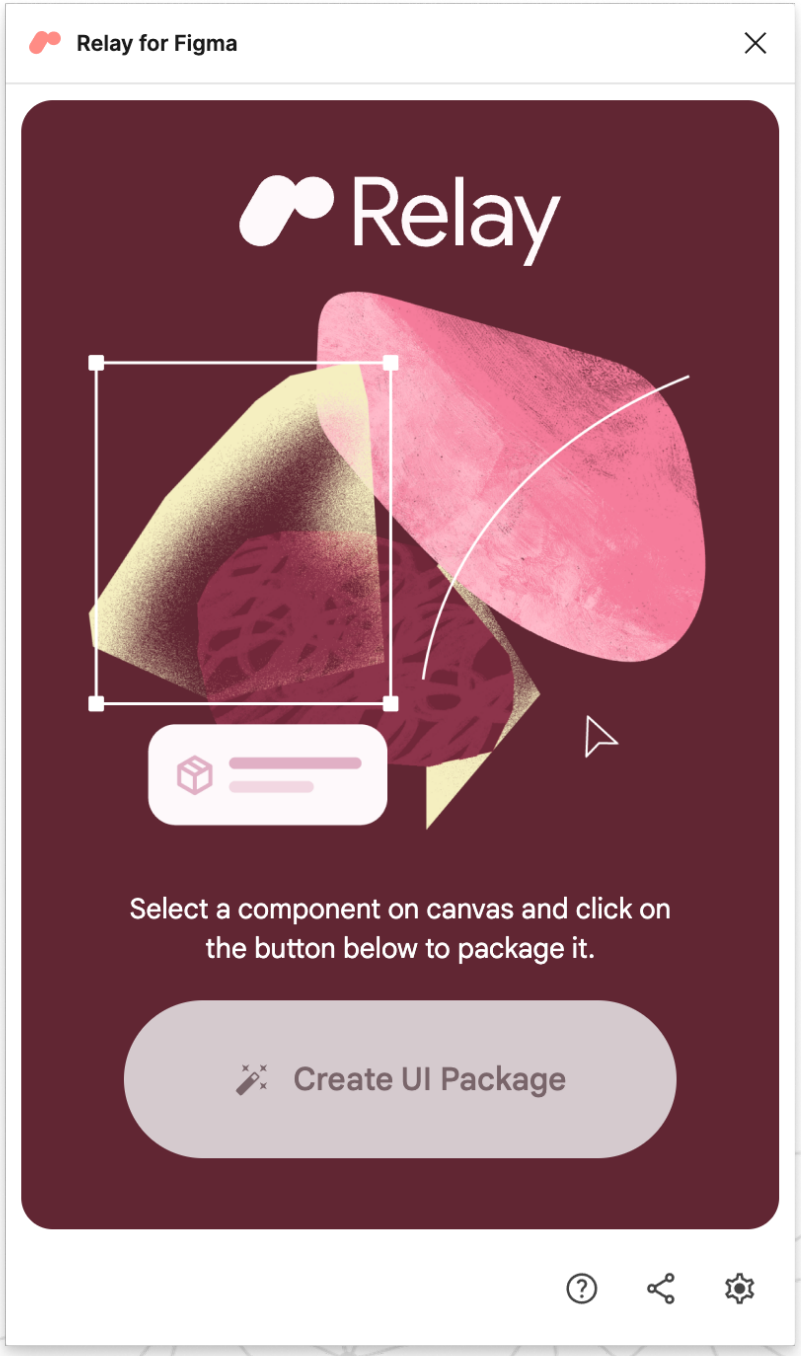


Relay

- **Relay for Figma plugin**
- **Relay for Android Studio plugin**
- Relay Gradle Plugin

Требования

- Android Studio
- Relay Plugin
- Figma аккаунт (+ токен)



Что делаем

- Figma account, токен
- Установка плагина в Figma
- Установка и настройка плагина в Android Studio
- Подключение Relay в проект
- Загрузка данных из макета
- Генерация

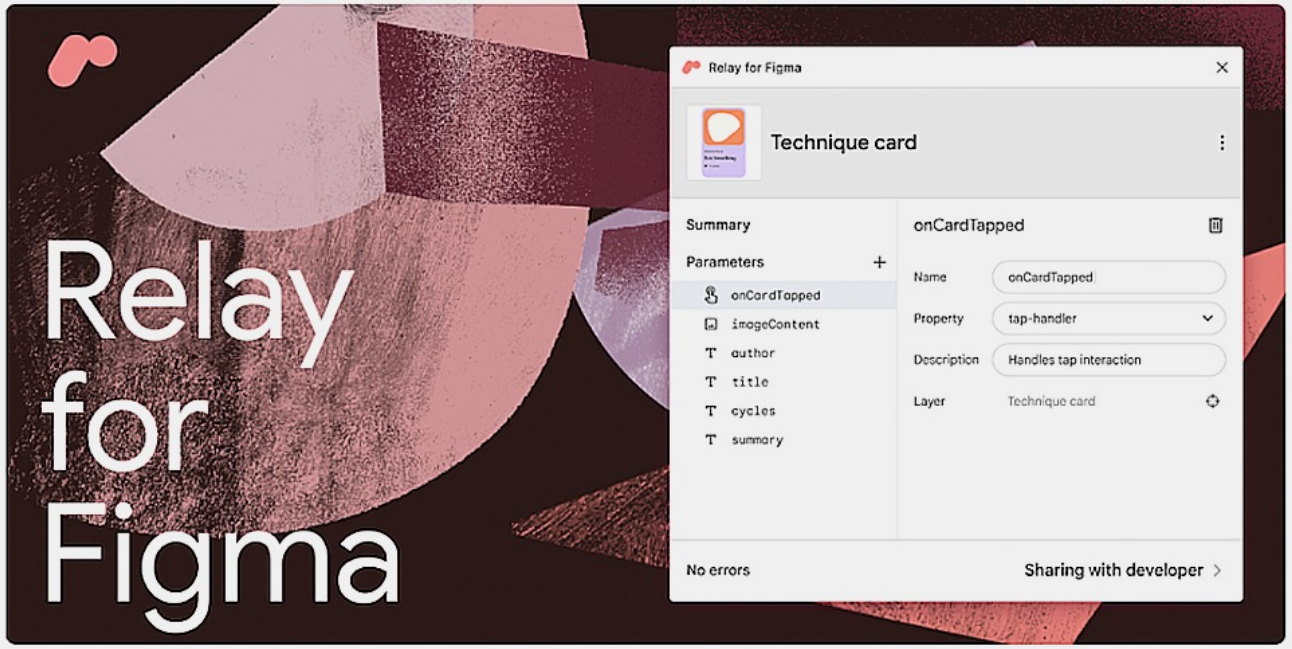
Установка плагина Figma

Material Design

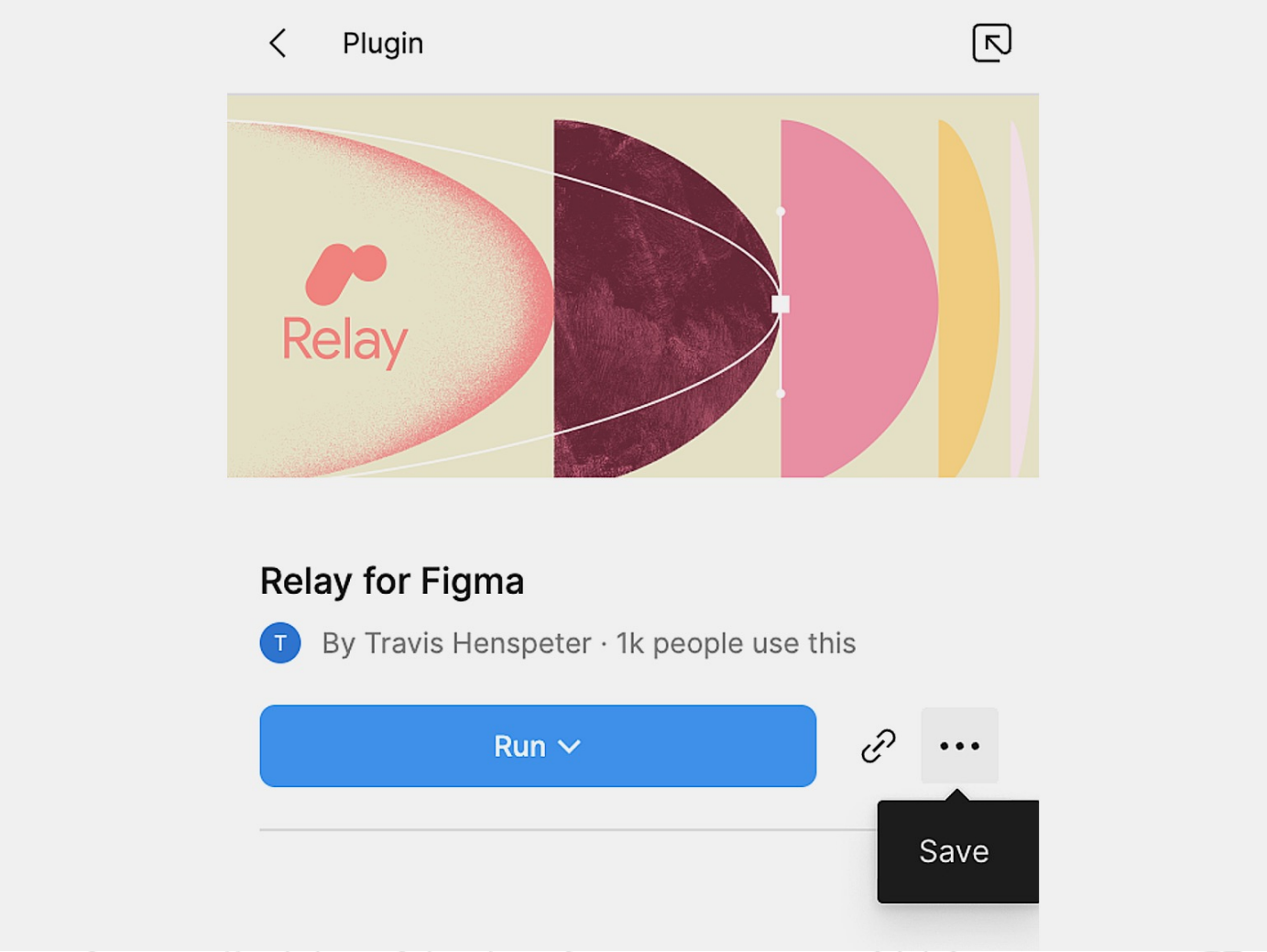
Relay for Figma

Plugin • ♡ 734 • 26.4k users

Open in Dev Mode

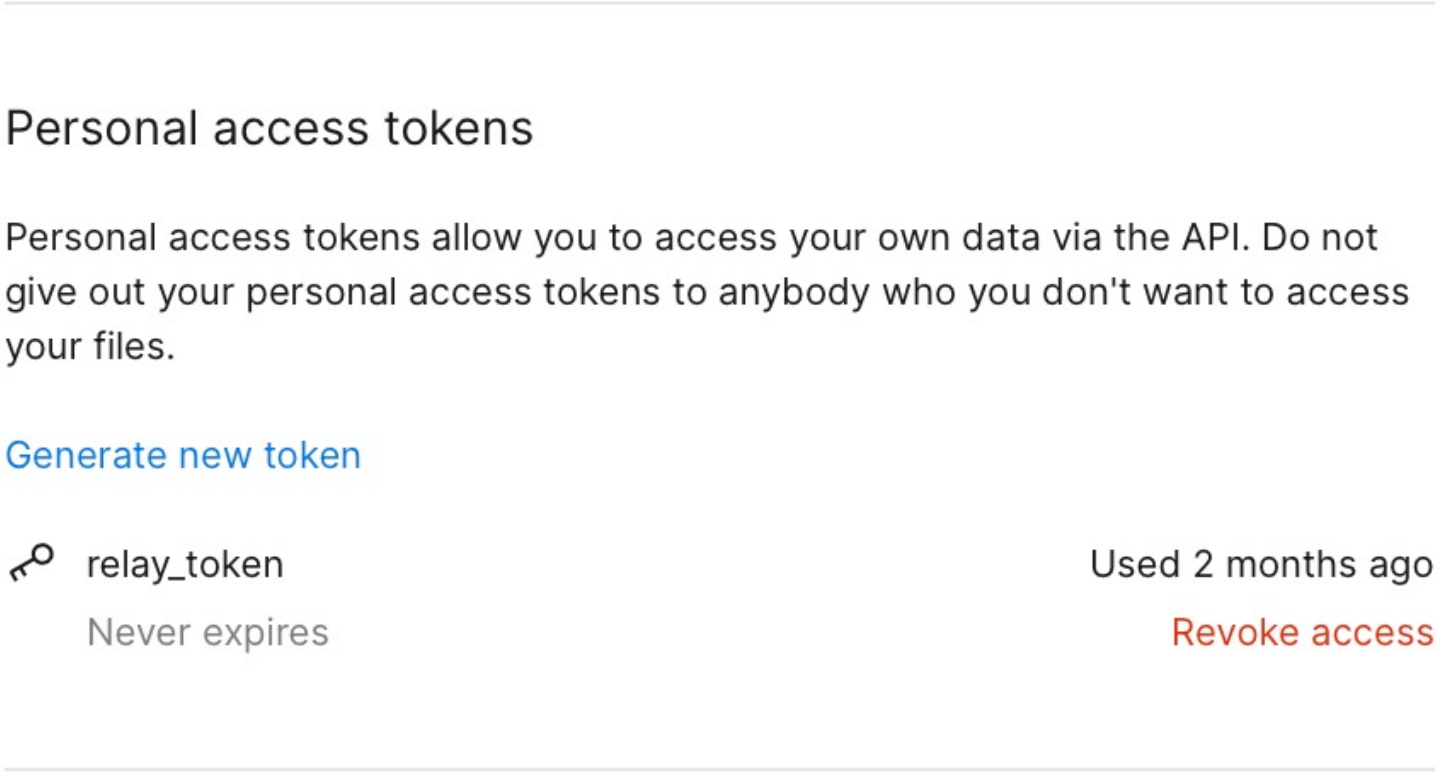


Установка плагина Figma



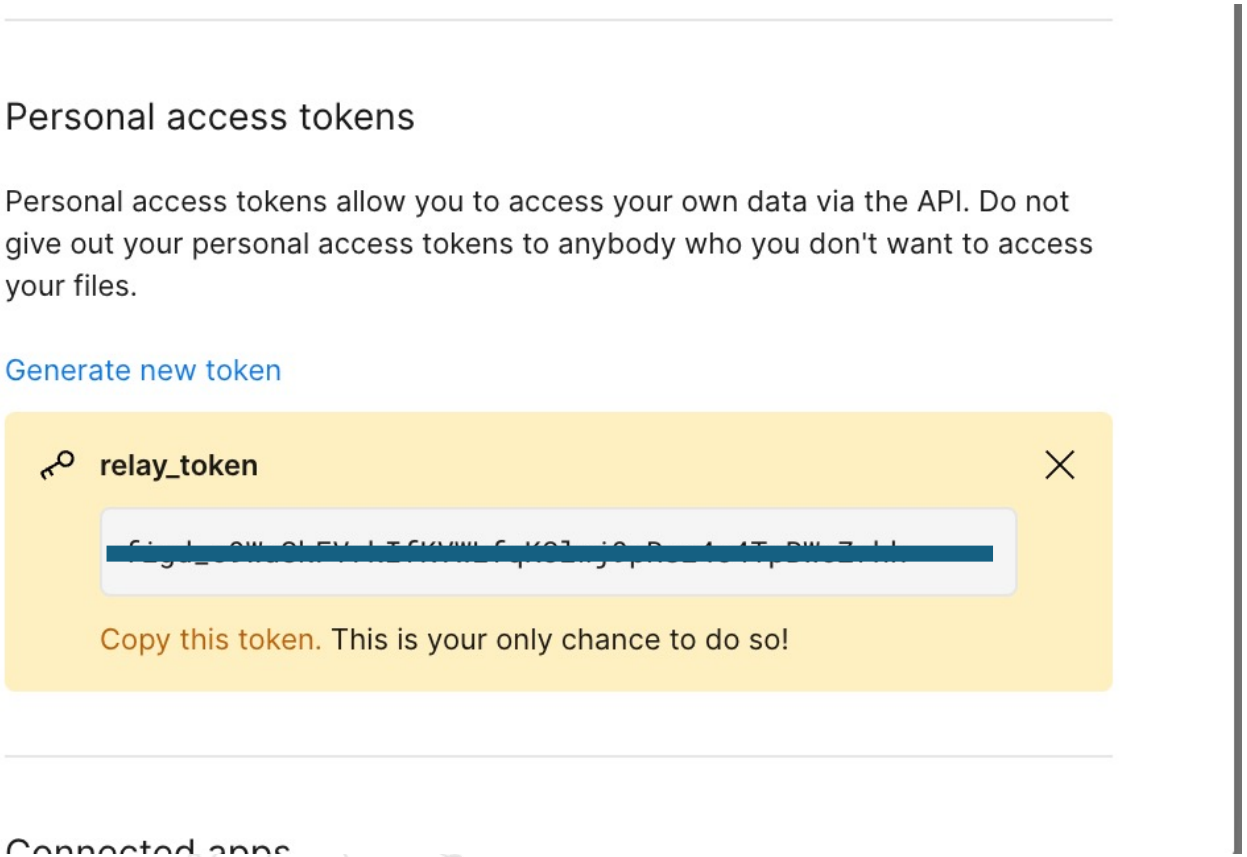
Токен

Токен генерируется в настройках аккаунта в Figma



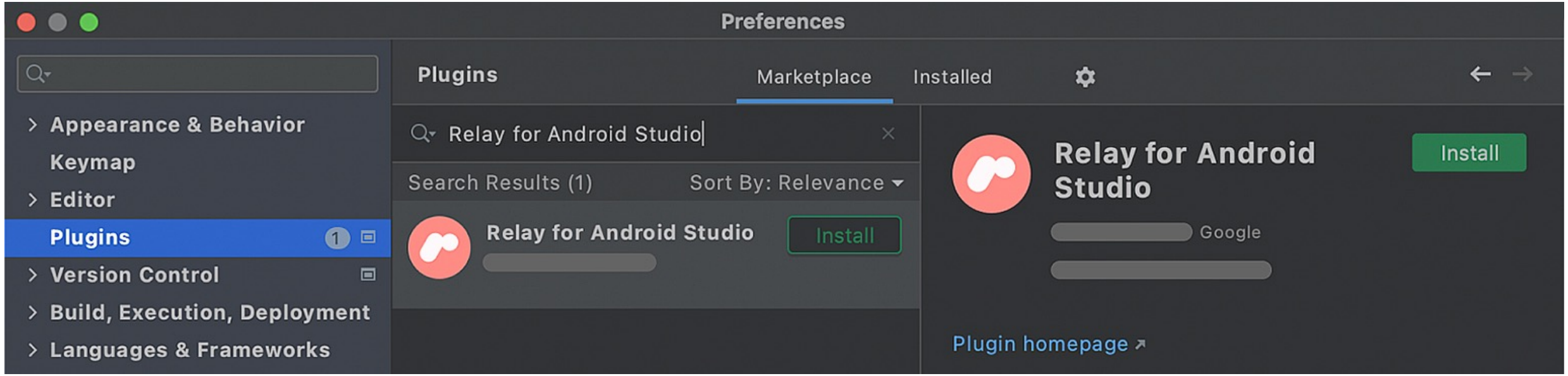
Токен

Токен генерируется в настройках аккаунта в Figma



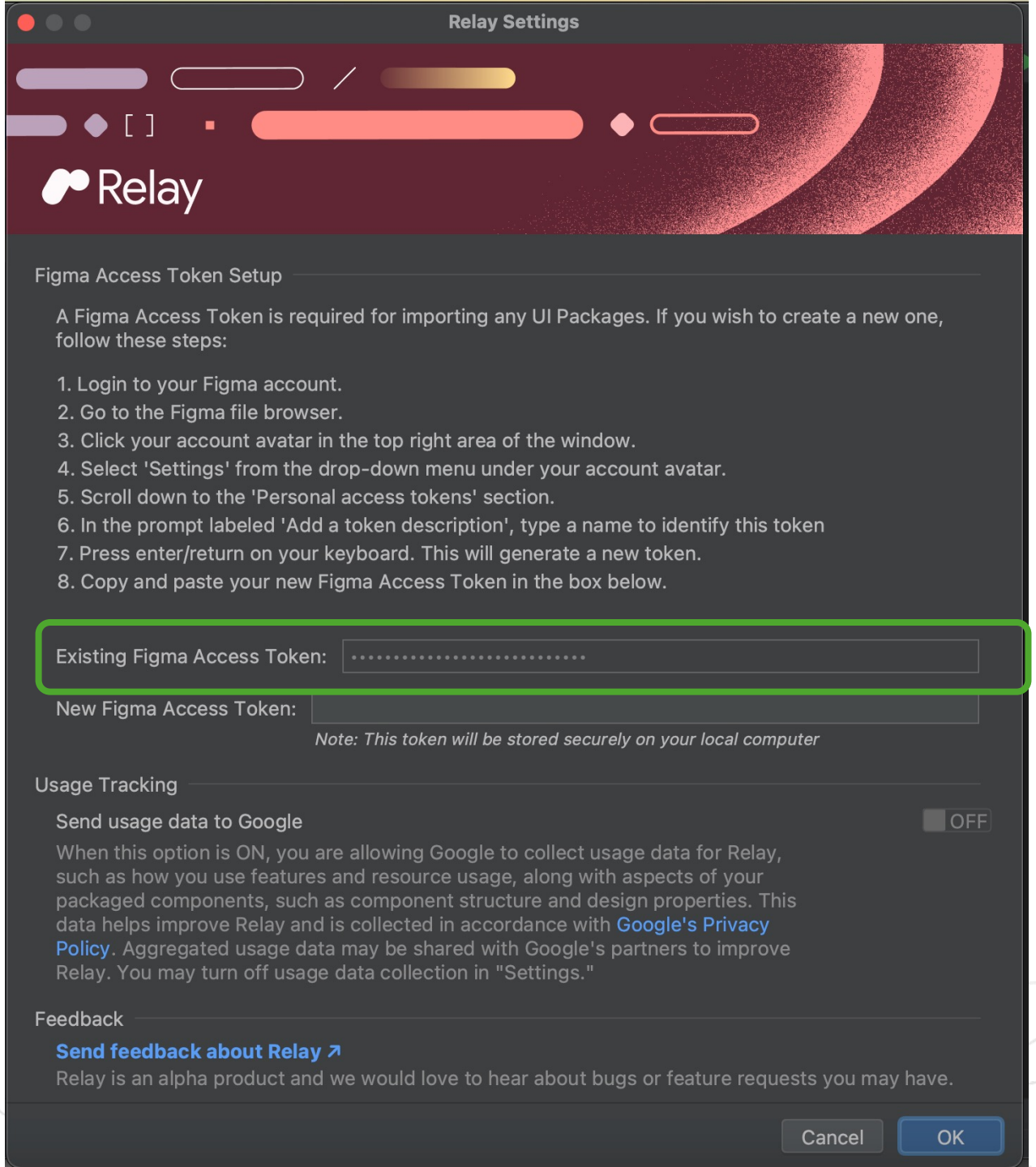
Настройка Android Studio

Установка плагина Preferences -> Plugins



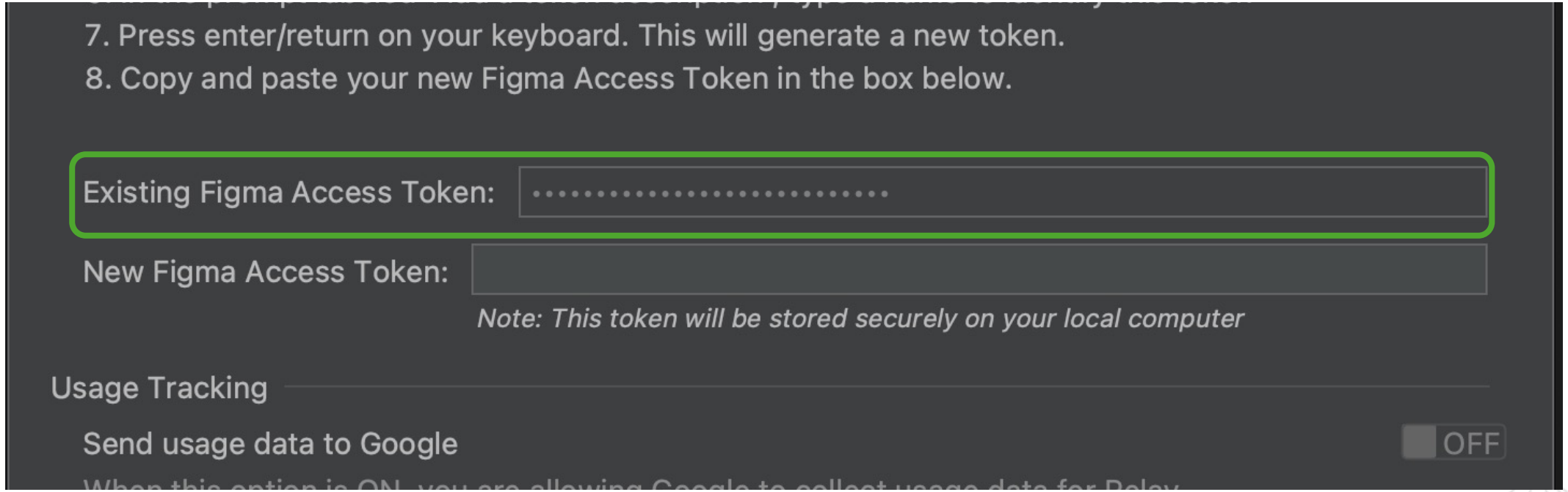
Токен

Прописываем токен в Tools – Relay Settings



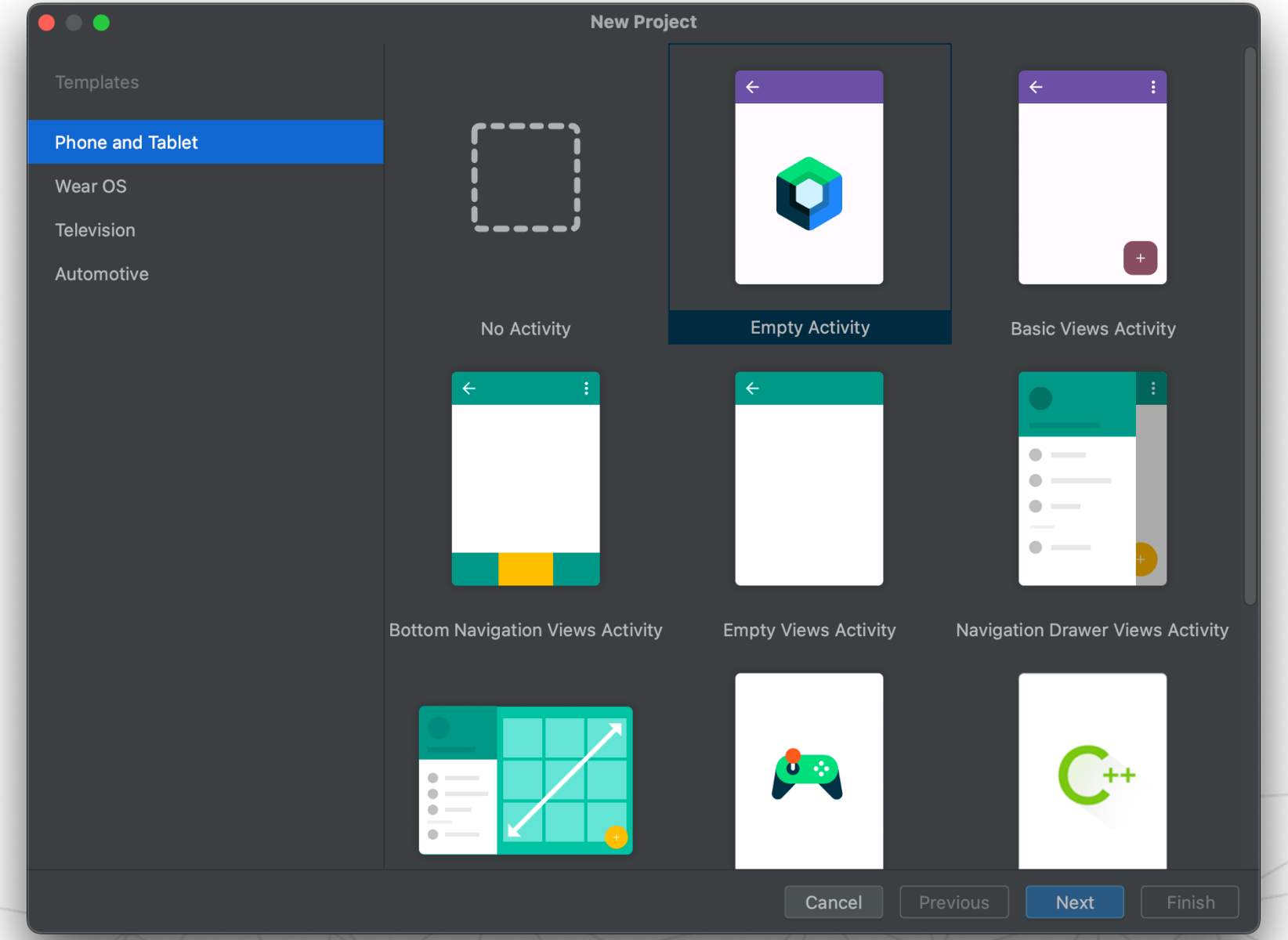
Токен

Прописываем токен в Tools – Relay Settings



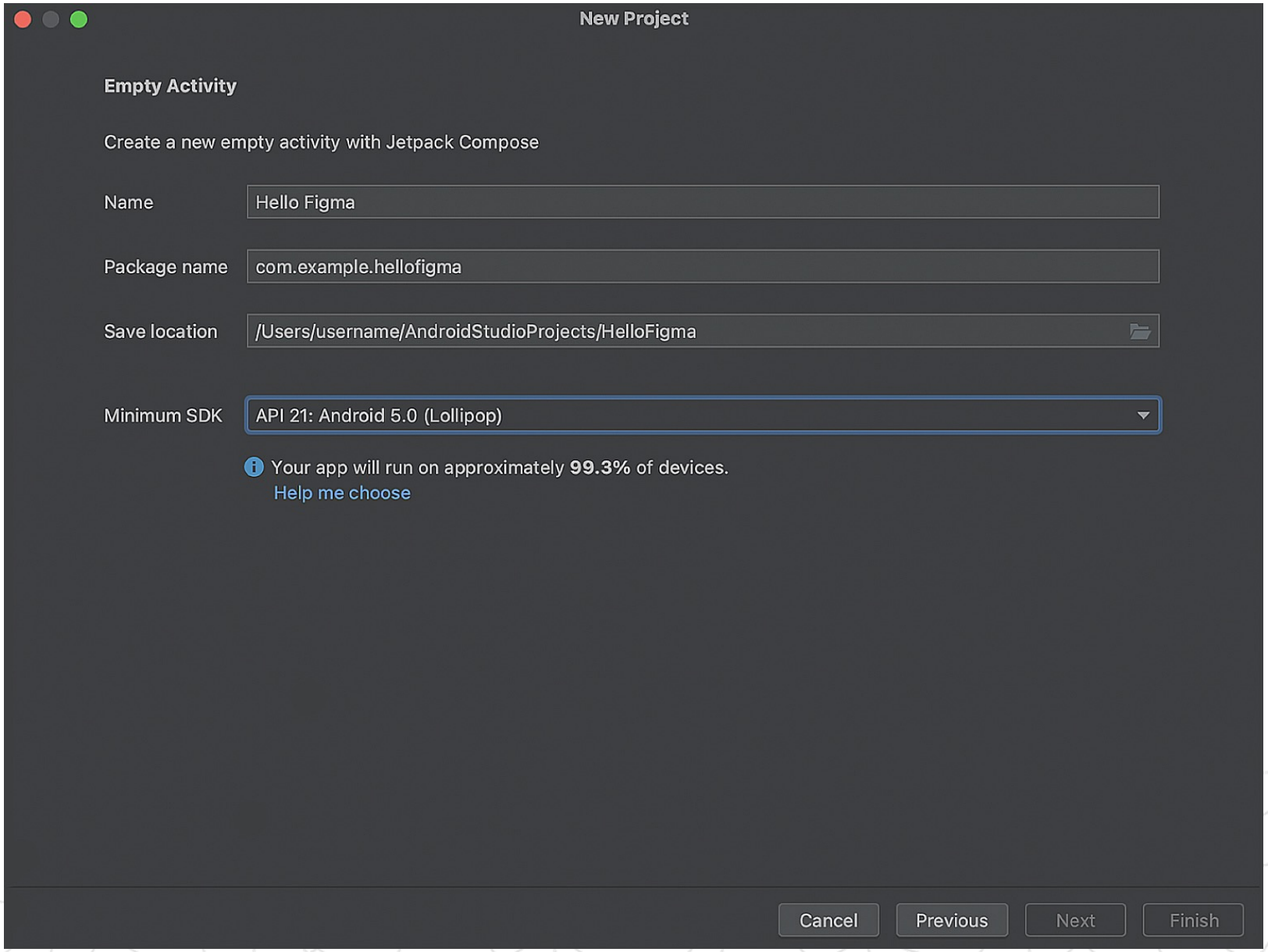
Настройка проекта Android Studio

Создание проекта



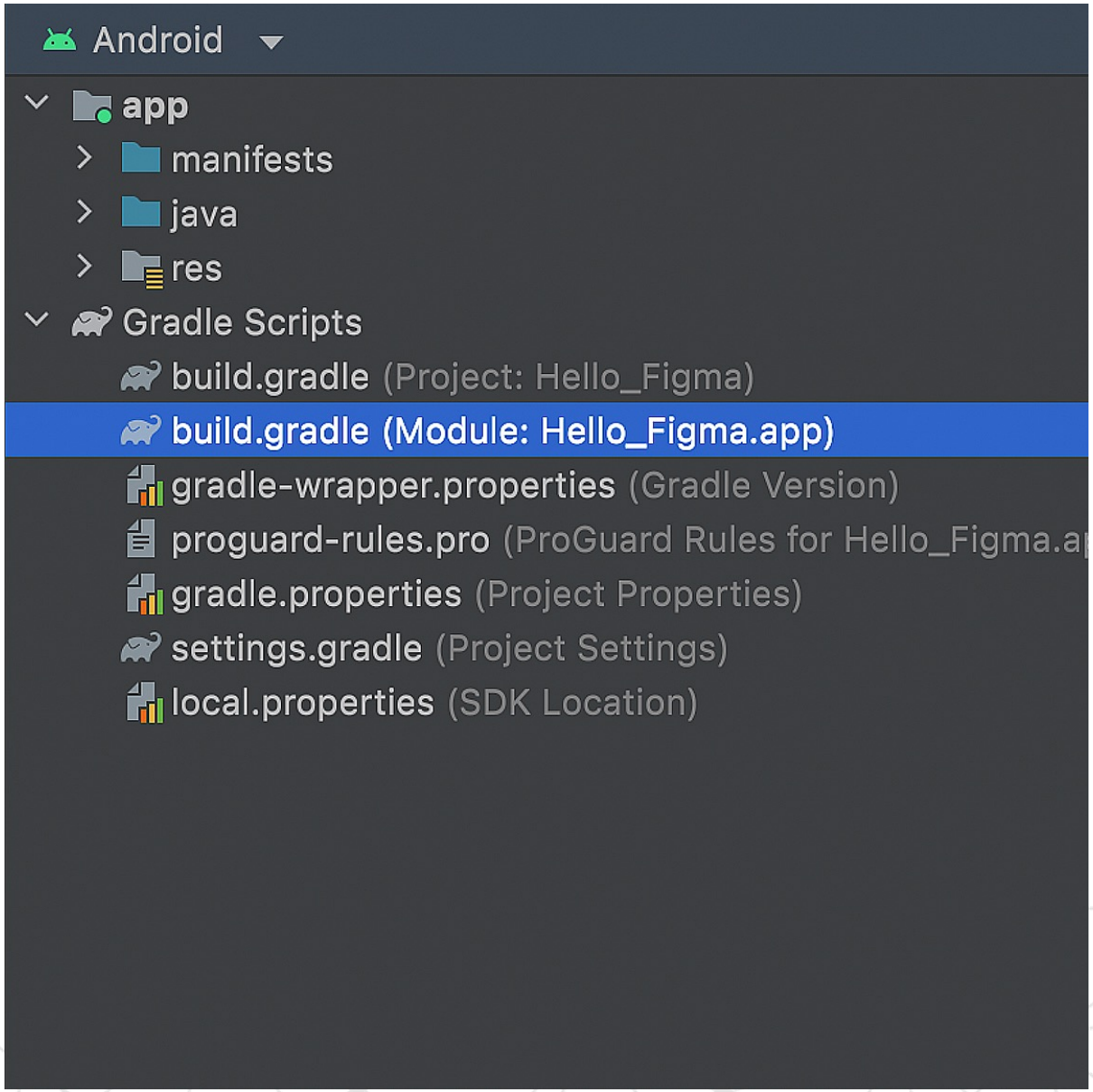
Настройка проекта Android Studio

Создание проекта



Настройка проекта Android Studio

Прописываем плагин



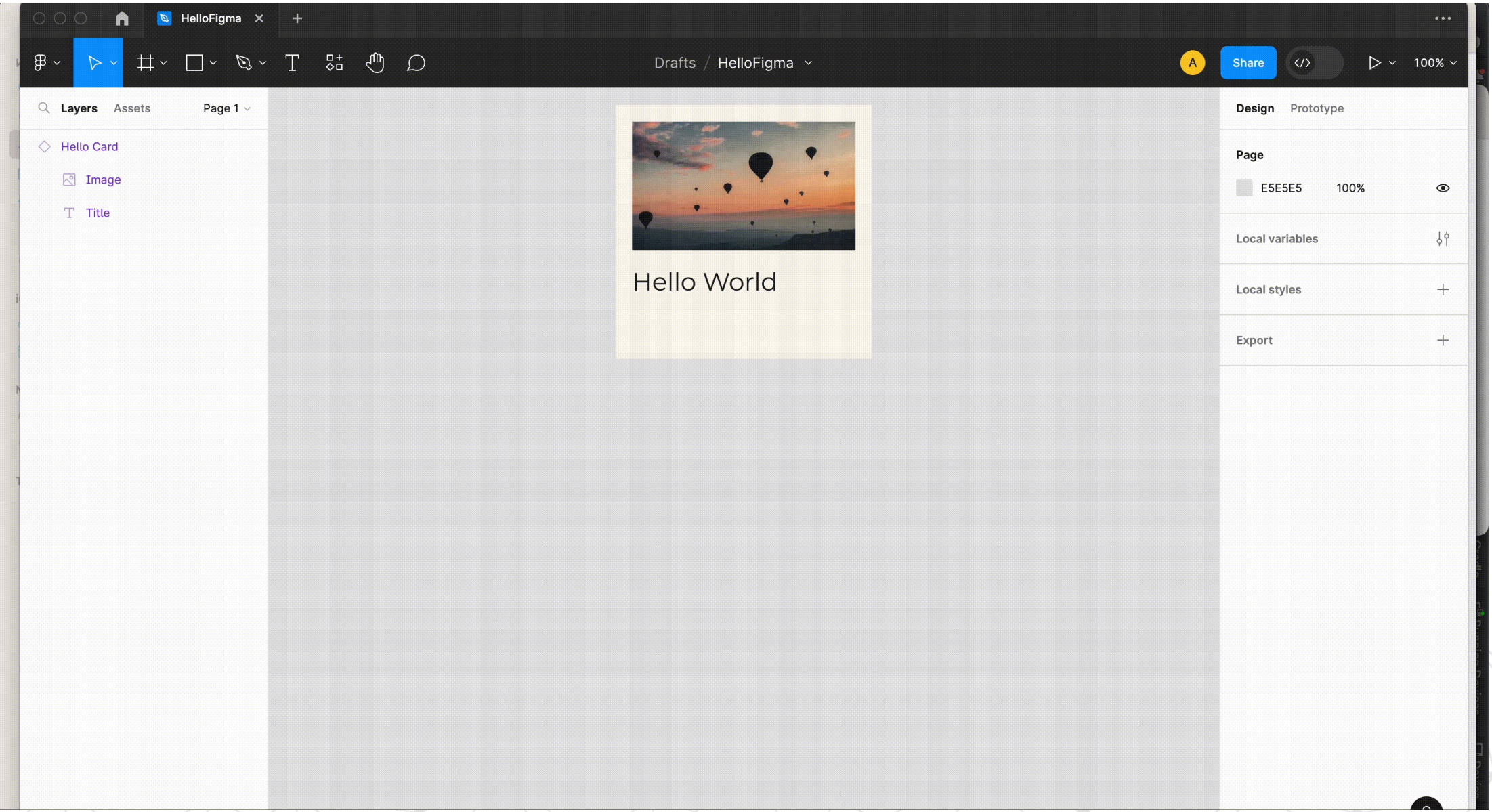
Настройка проекта Android Studio

Прописываем плагин. Текущая версия 0.3.11

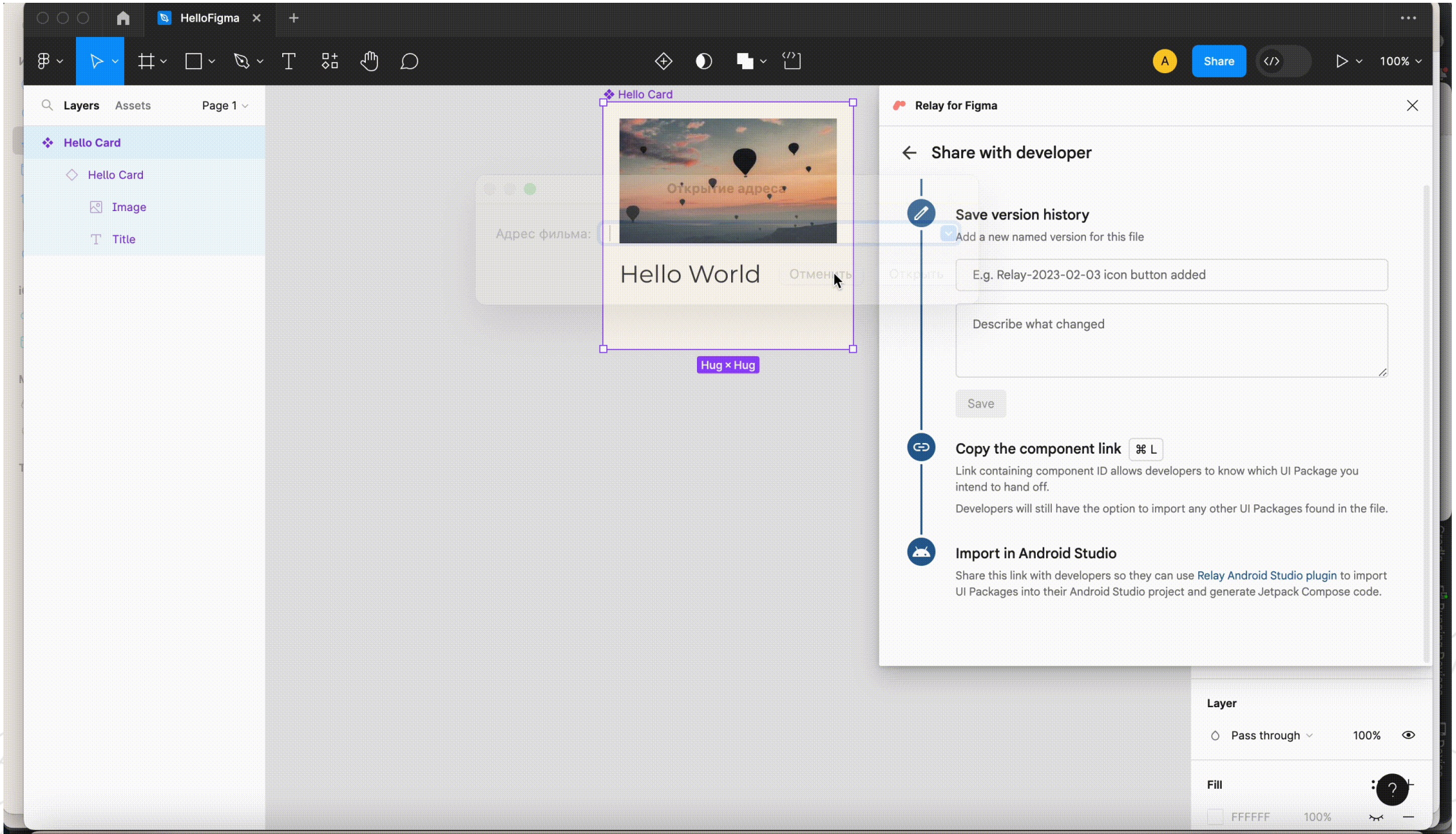
```
//build.gradle.kts
plugins {
    id("com.android.application") version "8.1.1" apply false
    id("org.jetbrains.kotlin.android") version "1.8.10" apply false
    id("com.google.relay") version "0.3.08" apply false
}

//build.gradle.kts app
plugins {
    id("com.android.application")
    id("org.jetbrains.kotlin.android")
    id("com.google.relay")
}
```

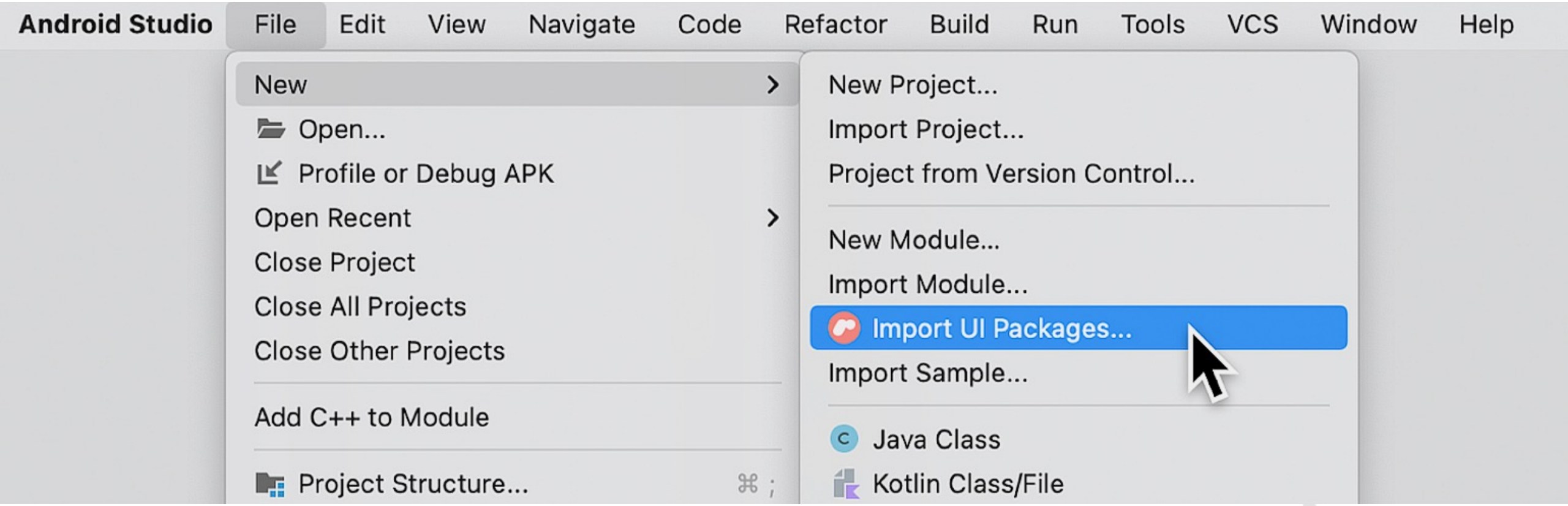
Компонент


HelloFigma
Drafts / HelloFigma
A
Share
100%
Layers
Assets
Page 1
Hello Card
Image
Title
Hello World
Design
Prototype
Page
E5E5E5
100%
Local variables
Local styles
Export

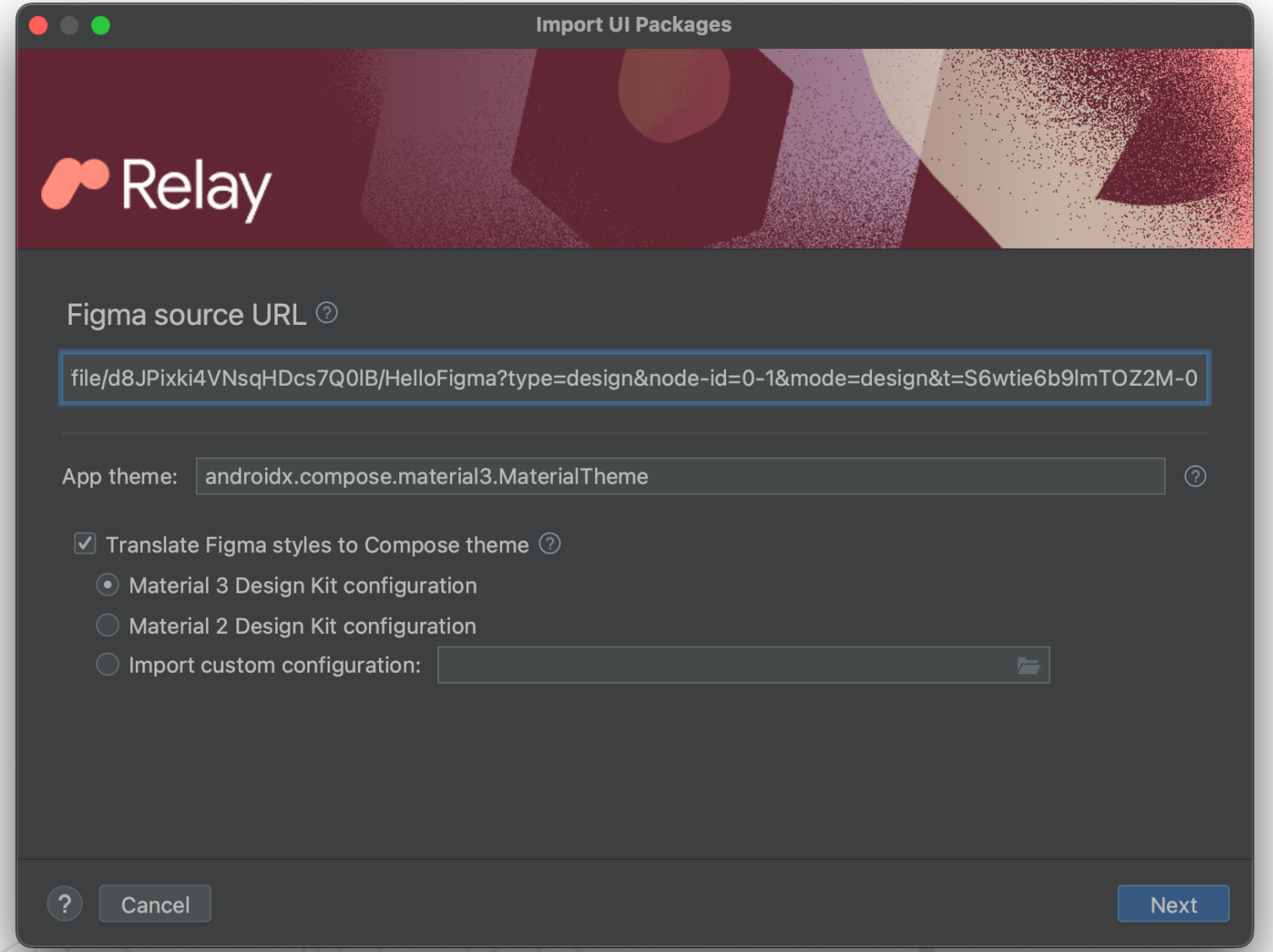
UI Package


HelloFigma
Share
100%
Layers
Assets
Page 1
Hello Card
Hello Card
Image
Title
Hello Card
Hello World
Hug × Hug
Relay for Figma
Share with developer
Save version history
Add a new named version for this file
E.g. Relay-2023-02-03 icon button added
Describe what changed
Save
Copy the component link
⌘ L
Link containing component ID allows developers to know which UI Package you intend to hand off.
Developers will still have the option to import any other UI Packages found in the file.
Import in Android Studio
Share this link with developers so they can use Relay Android Studio plugin to import UI Packages into their Android Studio project and generate Jetpack Compose code.
Layer
Pass through
100%
Fill
FFFFFF
100%

UI Package

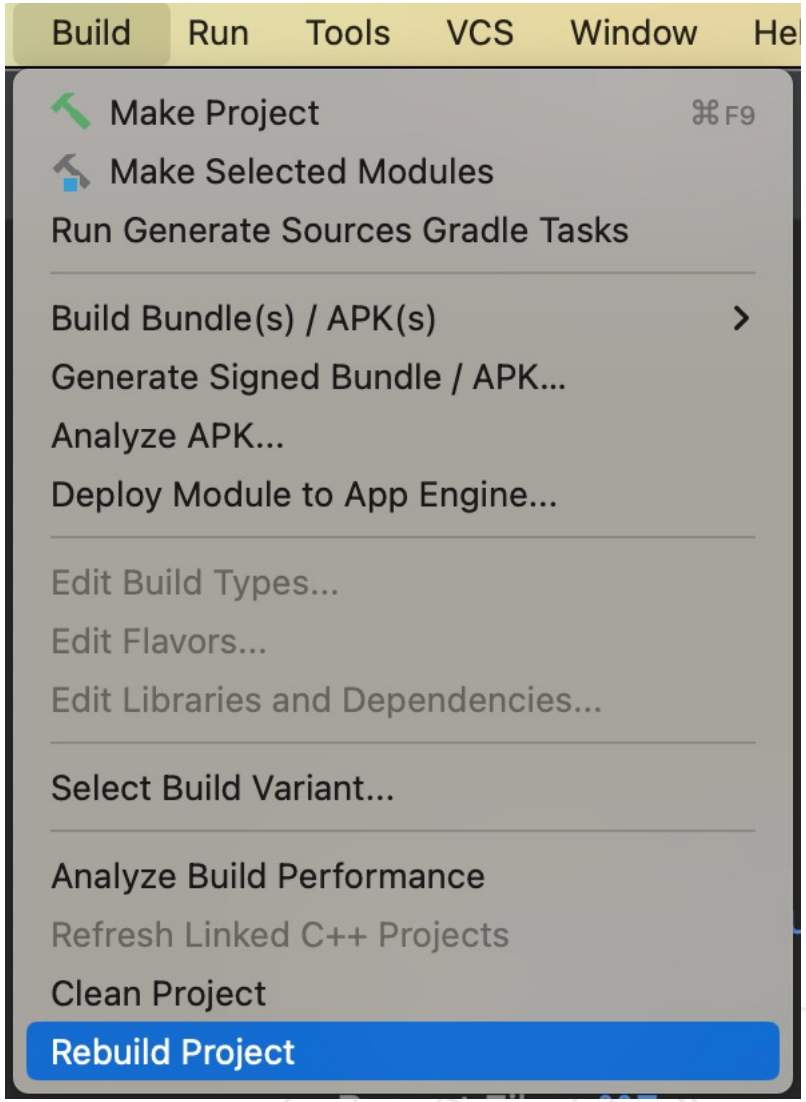

Android Studio
File
Edit
View
Navigate
Code
Refactor
Build
Run
Tools
VCS
Window
Help
New
Open...
Profile or Debug APK
Open Recent
Close Project
Close All Projects
Close Other Projects
Add C++ to Module
Project Structure...
⌘ ;
New Project...
Import Project...
Project from Version Control...
New Module...
Import Module...
Import UI Packages...
Import Sample...
Java Class
Kotlin Class/File

UI Package


Import UI Packages
Relay
Figma source URL
file/d8JPixki4VNsqHDcs7Q0lB/HelloFigma?type=design&node-id=0-1&mode=design&t=S6wtie6b9lmTOZ2M-0
App theme:
androidx.compose.material3.MaterialTheme
Translate Figma styles to Compose theme
Material 3 Design Kit configuration
Material 2 Design Kit configuration
Import custom configuration:
Cancel
Next

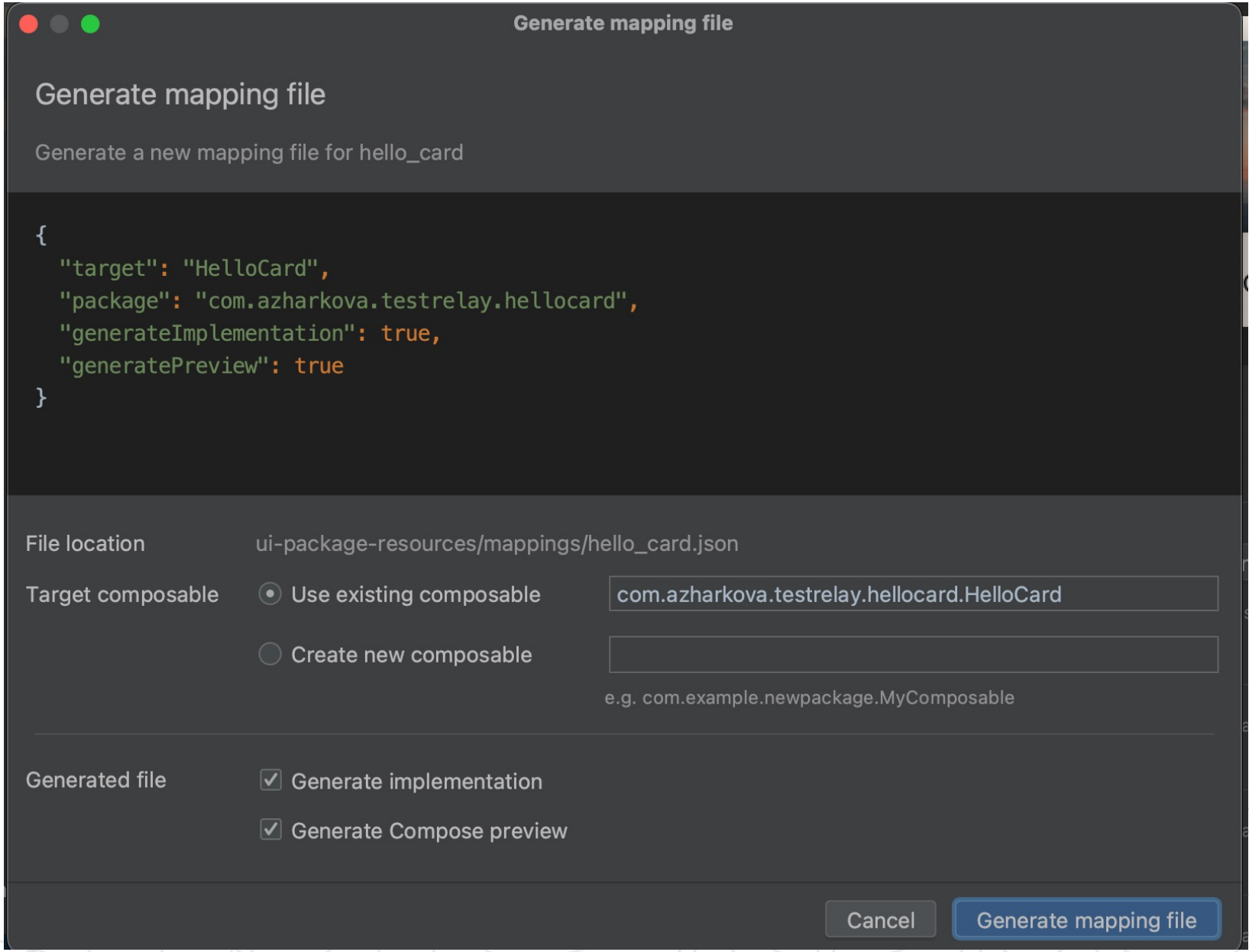
Генерация

Запускаем ребилд для генерации

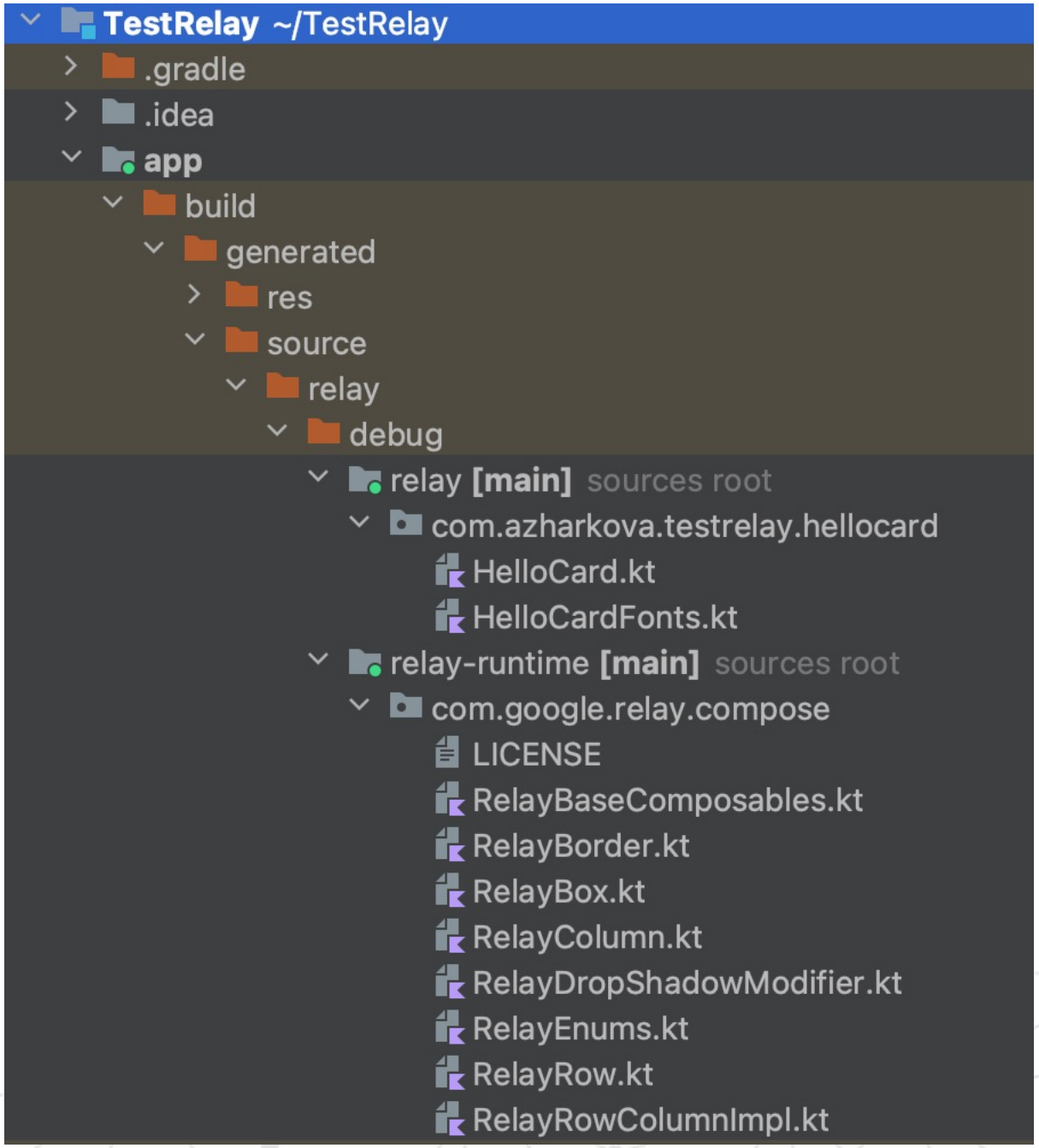


Генерация

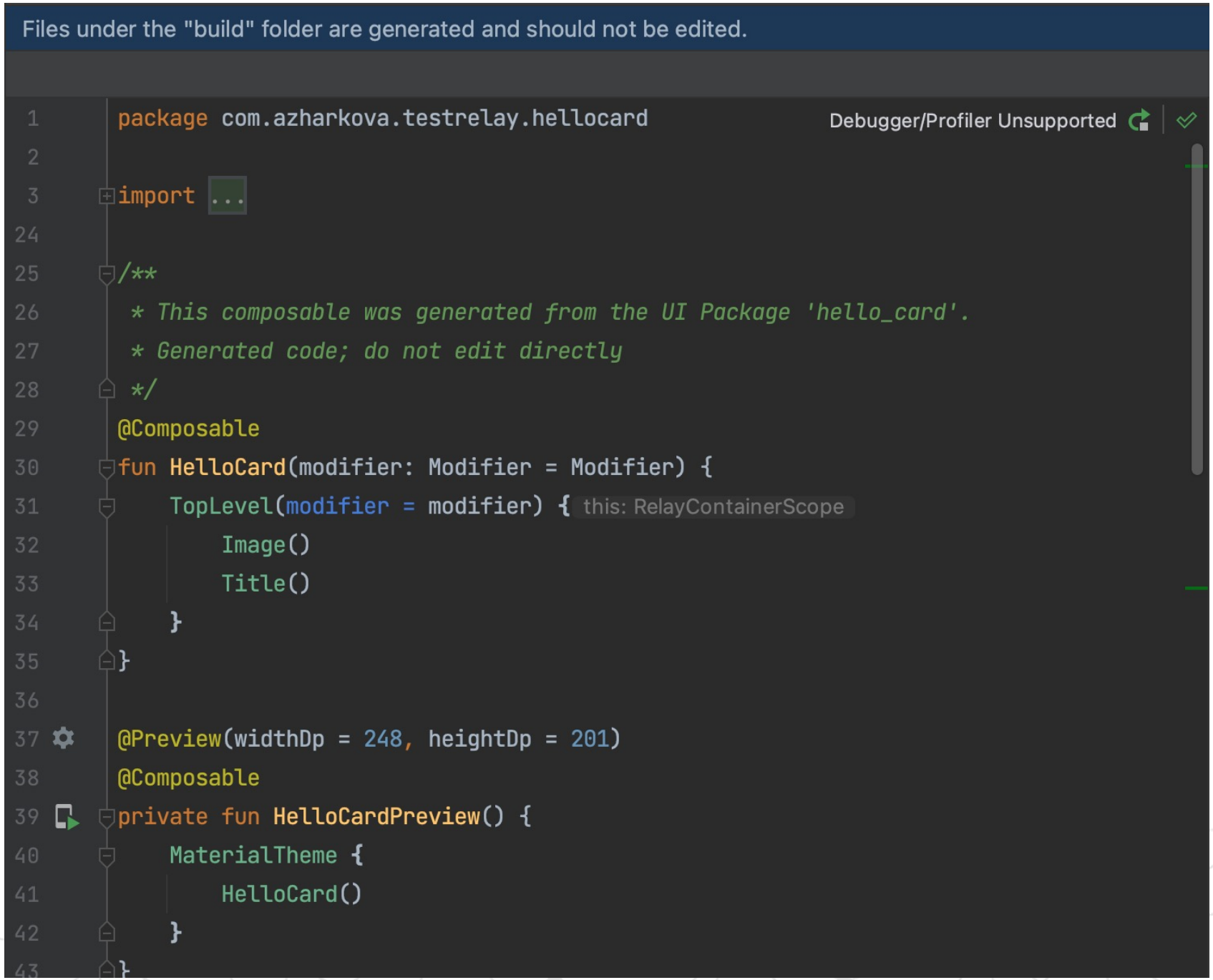
Можно замаппить в свой Composable



Генерация


TestRelay ~/TestRelay
.gradle
.idea
app
build
generated
res
source
relay
debug
relay [main] sources root
com.azharkova.testrelay.hellocard
HelloCard.kt
HelloCardFonts.kt
relay-runtime [main] sources root
com.google.relay.compose
LICENSE
RelayBaseComposables.kt
RelayBorder.kt
RelayBox.kt
RelayColumn.kt
RelayDropShadowModifier.kt
RelayEnums.kt
RelayRow.kt
RelayRowColumnImpl.kt

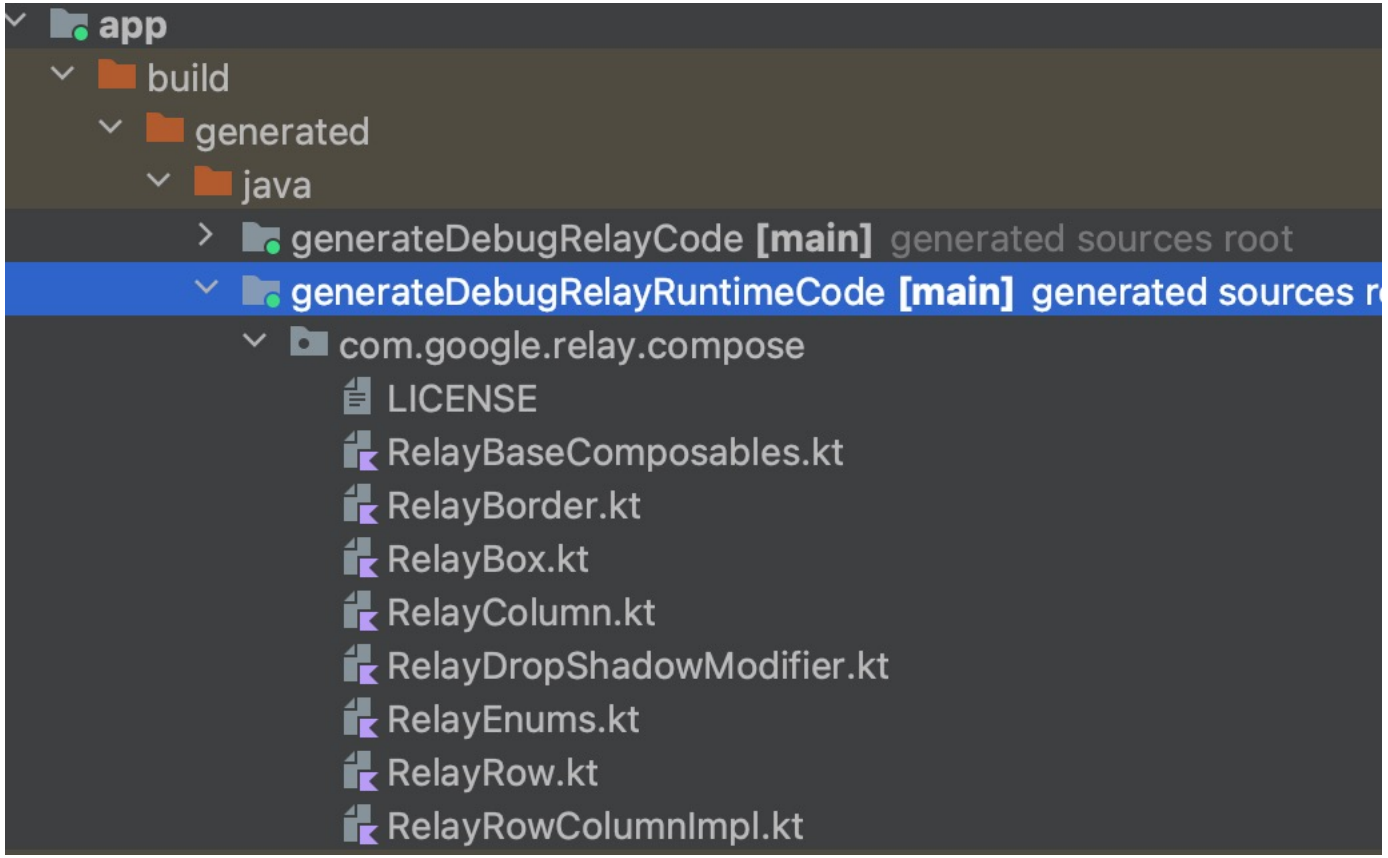
Генерация




```
1  package com.azharkova.testrelay.hellocard
2
3  import ...
24
25 /**
26  * This composable was generated from the UI Package 'hello_card'.
27  * Generated code; do not edit directly
28  */
29 @Composable
30 fun HelloCard(modifier: Modifier = Modifier) {
31     TopLevel(modifier = modifier) {
32         Image()
33         Title()
34     }
35 }
36
37 @Preview(widthDp = 248, heightDp = 201)
38 @Composable
39 private fun HelloCardPreview() {
40     MaterialTheme {
41         HelloCard()
42     }
43 }
```

Что внутри

Под капотом Composable содержат Relay-элементы



Генерация

RelayImage – внутренний Image-компонент библиотеки

```
@Composable
fun Image(modifier: Modifier = Modifier) {
    RelayImage(
        image = painterResource(R.drawable.hello_card_image),
        contentScale = ContentScale.Crop,
        modifier = modifier.requiredWidth(216.0.dp).requiredHeight(124.0.dp)
    )
}
```

Генерация

RelayText – внутренний Text-компонент библиотеки

```
@Composable
fun Title(modifier: Modifier = Modifier) {
    RelayText(
        content = "Hello World",
        fontSize = 24.0.sp,
        fontFamily = montserrat,
        color = Color(
            alpha = 255,
            red = 27,
            green = 28,
            blue = 32
        ),
        height = 1.219000021616618.em,
        textAlign = TextAlign.Left,
        modifier = modifier
    )
}
```

Темы и стили

Files under the "build" folder are generated and should not be edited.

Code Split Design

Debugger/Profiler Unsupported

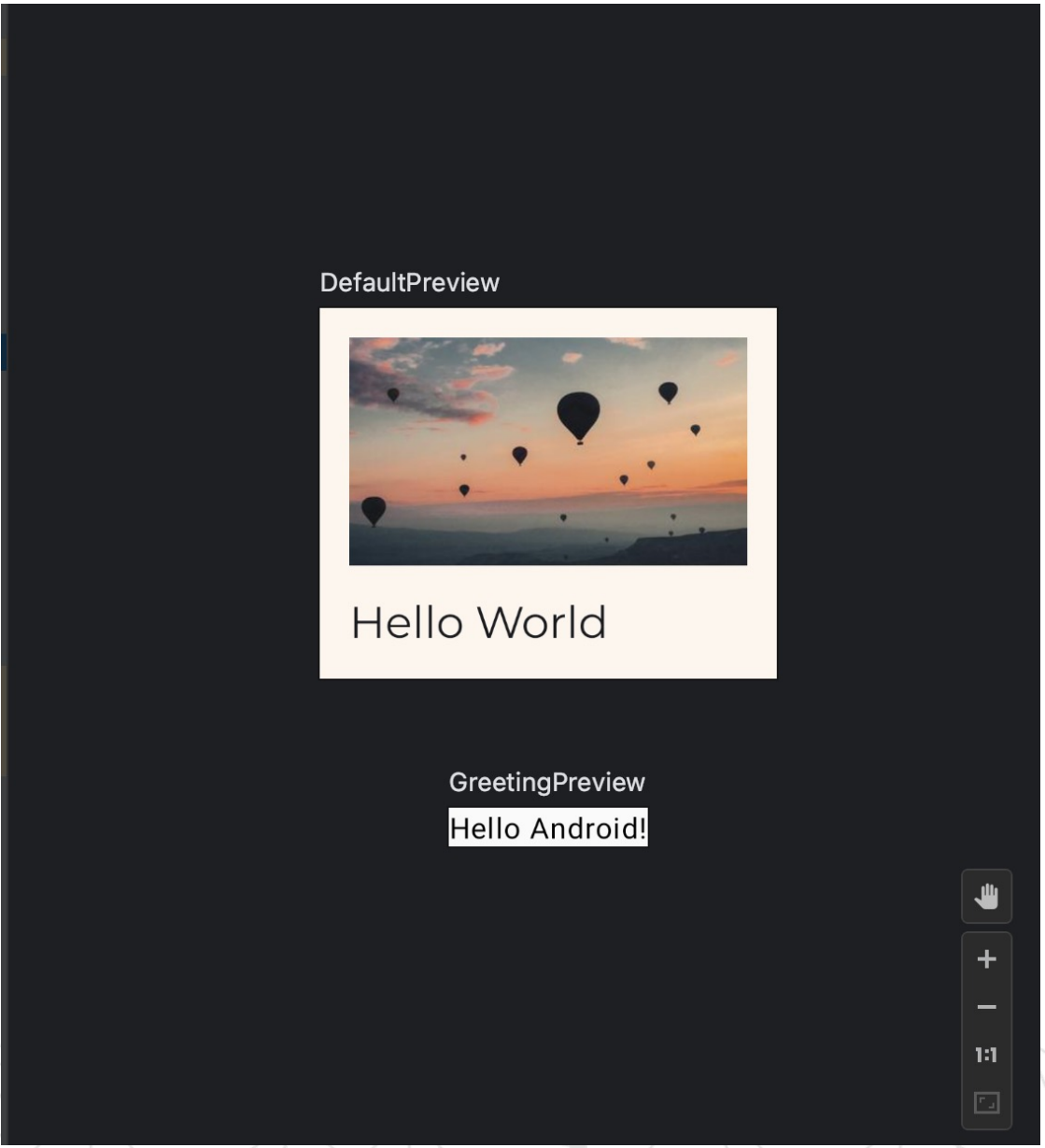
```
1  package com.azharkova.testrelay.hellocard
2
3  import ...
8
9  val montserrat: FontFamily = FontFamily(
10     Font(R.font.relay_montserrat_semibold_italic, weight = FontWeight.W600, style = FontStyle.Italic),
11     Font(R.font.relay_montserrat_medium_italic, weight = FontWeight.W500, style = FontStyle.Italic),
12     Font(R.font.relay_montserrat_extrabold, weight = FontWeight.W800, style = FontStyle.Normal),
13     Font(R.font.relay_montserrat_regular, weight = FontWeight.W400, style = FontStyle.Normal),
14     Font(R.font.relay_montserrat_extralight_italic, weight = FontWeight.W200, style = FontStyle.Italic),
15     Font(R.font.relay_montserrat_black, weight = FontWeight.W900, style = FontStyle.Normal),
16     Font(R.font.relay_montserrat_extralight, weight = FontWeight.W200, style = FontStyle.Normal),
17     Font(R.font.relay_montserrat_medium, weight = FontWeight.W500, style = FontStyle.Normal),
18     Font(R.font.relay_montserrat_bold, weight = FontWeight.W700, style = FontStyle.Normal),
19     Font(R.font.relay_montserrat_light_italic, weight = FontWeight.W300, style = FontStyle.Italic),
20     Font(R.font.relay_montserrat_thin, weight = FontWeight.W100, style = FontStyle.Normal),
21     Font(R.font.relay_montserrat_bold_italic, weight = FontWeight.W700, style = FontStyle.Italic),
22     Font(R.font.relay_montserrat_italic, weight = FontWeight.W400, style = FontStyle.Italic),
23     Font(R.font.relay_montserrat_semibold, weight = FontWeight.W600, style = FontStyle.Normal),
24     Font(R.font.relay_montserrat_light, weight = FontWeight.W300, style = FontStyle.Normal),
25     Font(R.font.relay_montserrat_thin_italic, weight = FontWeight.W100, style = FontStyle.Italic),
26     Font(R.font.relay_montserrat_black_italic, weight = FontWeight.W900, style = FontStyle.Italic),
27     Font(R.font.relay_montserrat_extrabold_italic, weight = FontWeight.W800, style = FontStyle.Italic)
28 )
```

Встраивание

Сгенерированные Compose можно встроить в Activity/Fragment

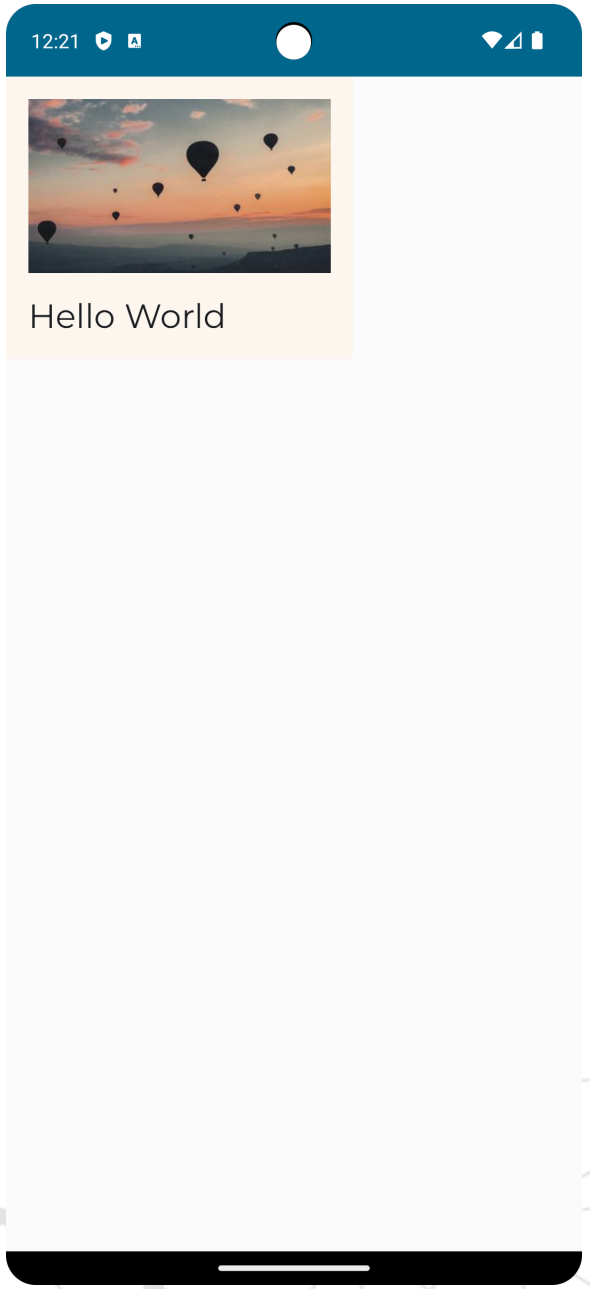
```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            TestRelayTheme {
                // A surface container using the 'background' color from the theme
                Surface(color = MaterialTheme.colorScheme.background) {
                    // Greeting("Android") // Delete this line
                    HelloCard()  // Add this line
                }
            }
        }
    }
}
```

Preview


DefaultPreview
Hello World
GreetingPreview
Hello Android!
1:1

Запускаем и проверяем

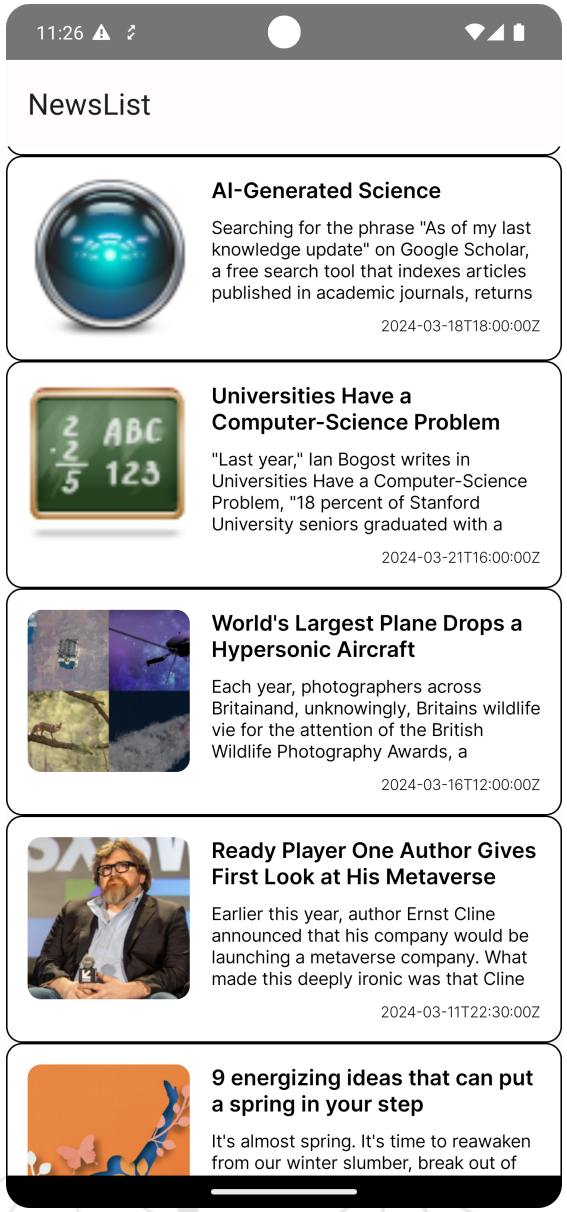
Наше готовое приложение

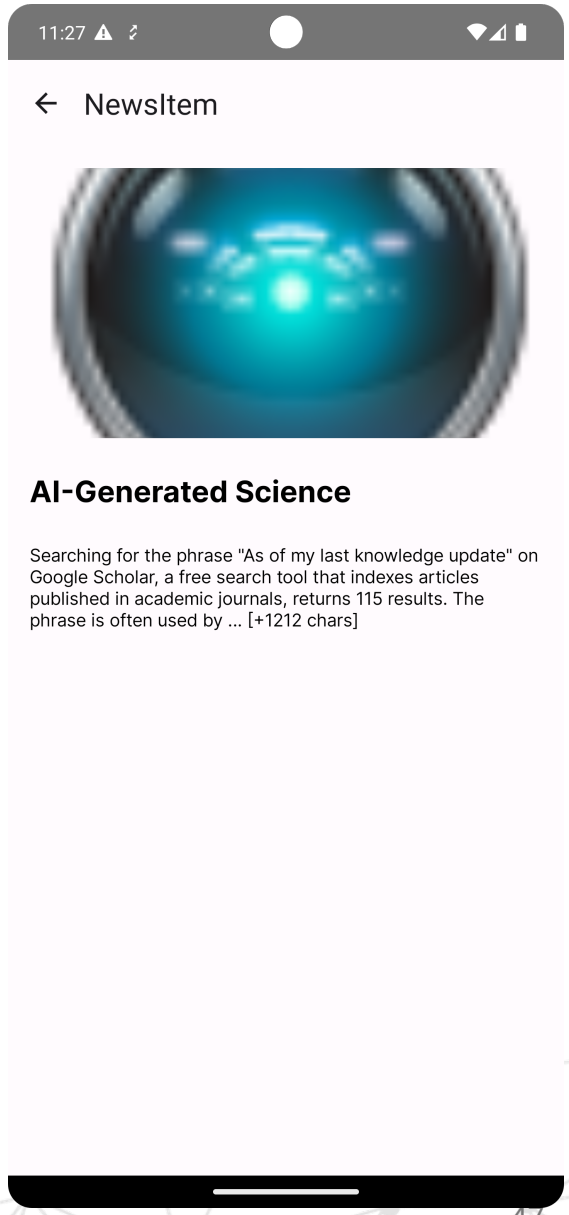


А теперь реальный псевдо-боевой кейс

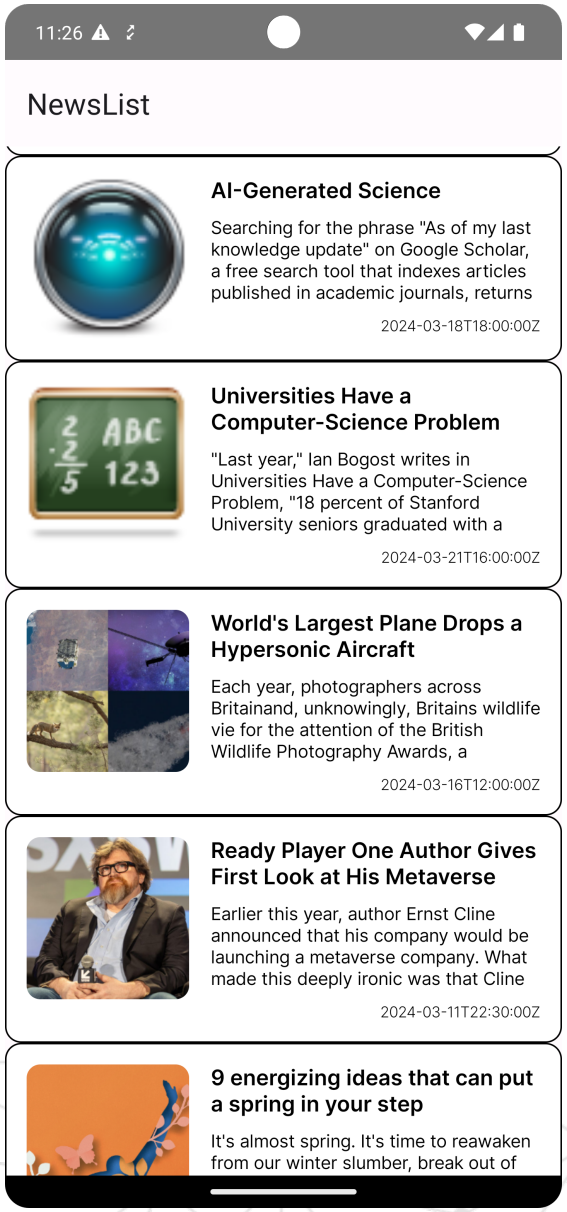
Приложение новостей

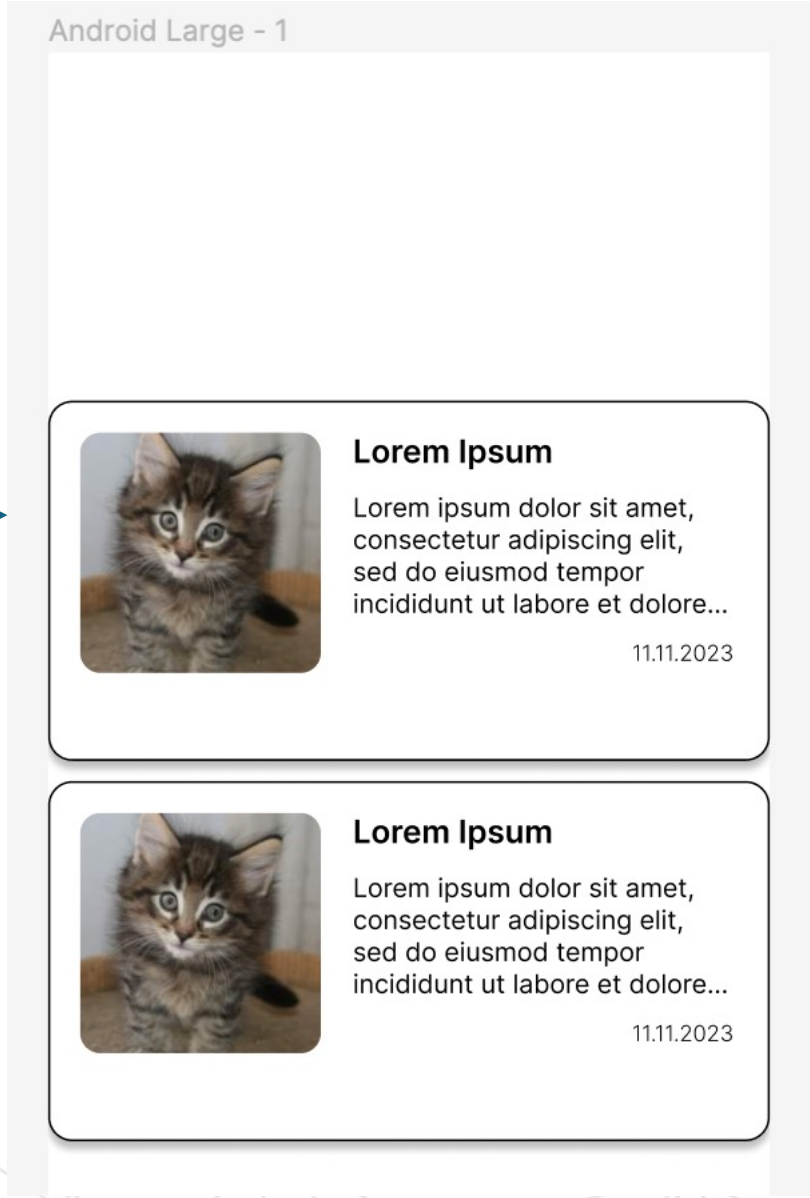
https://github.com/anioutkazharkova/NewsAppRelay





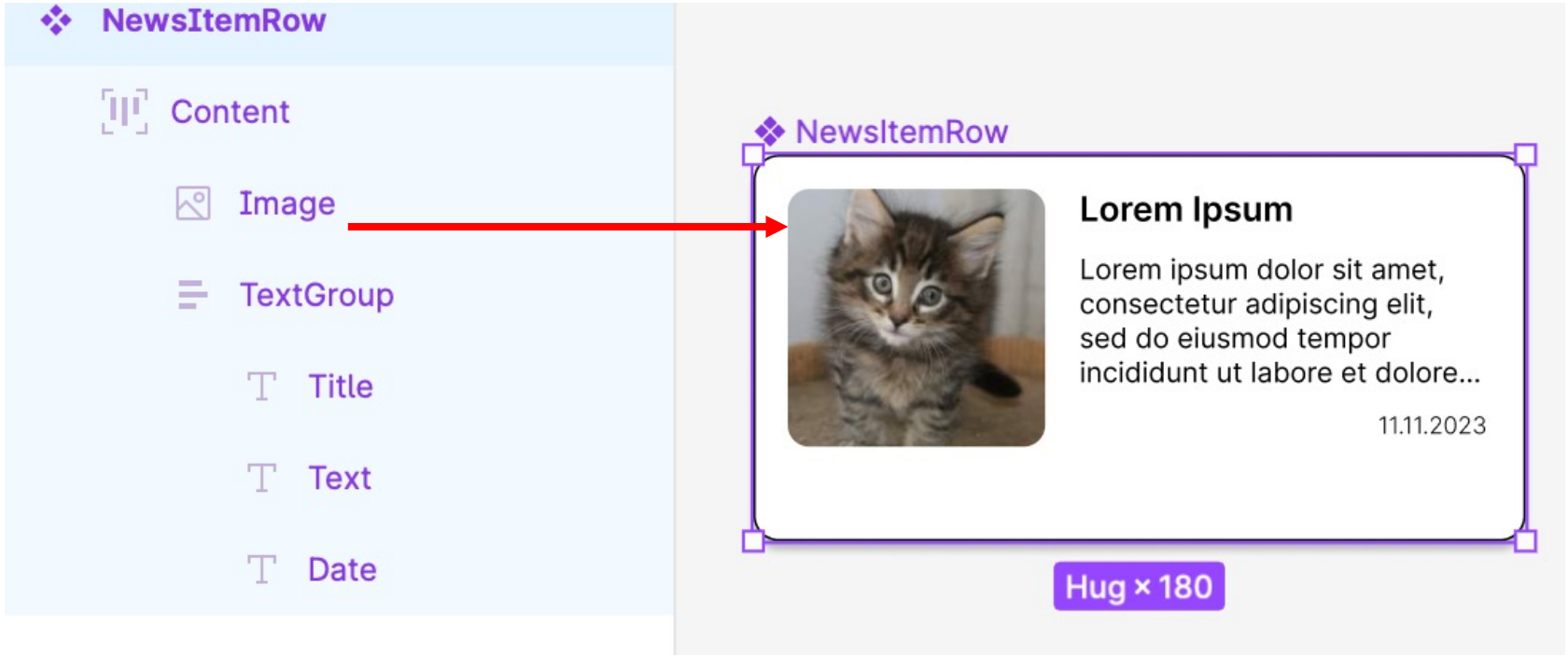
Список новостей





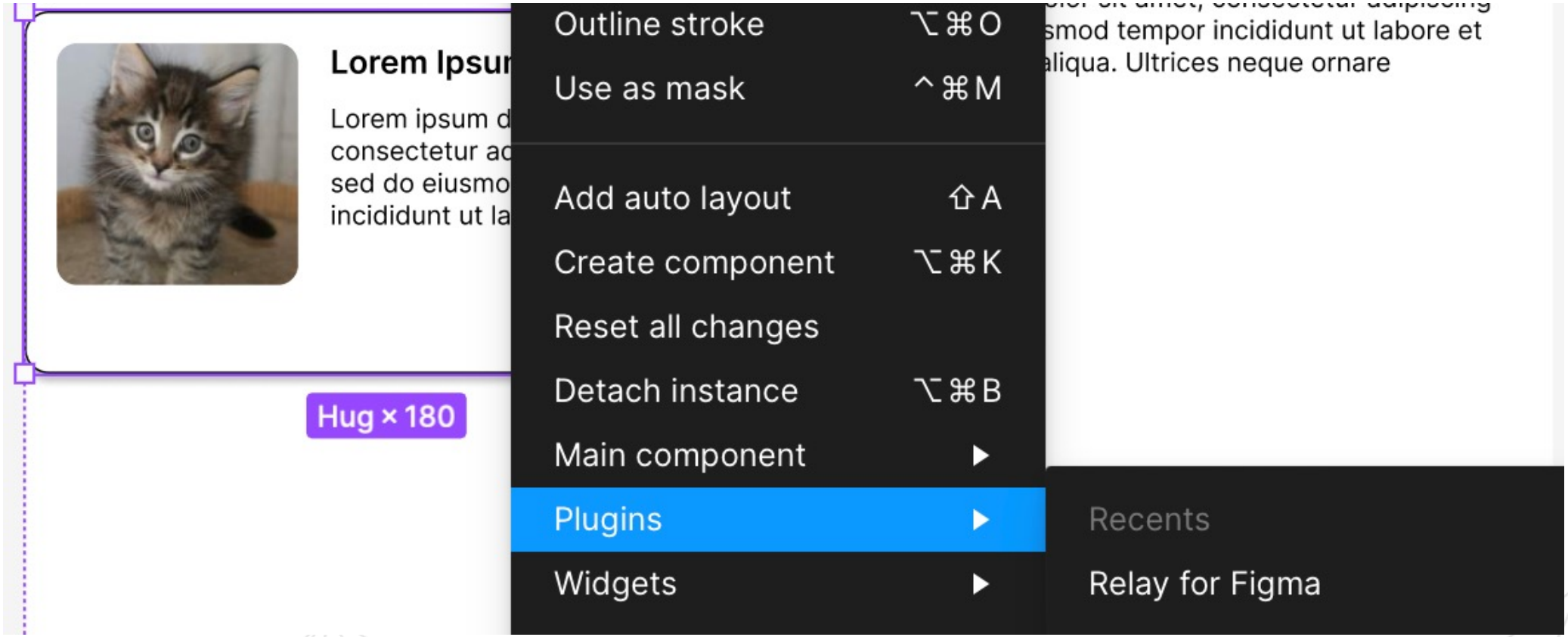
Список новостей

Компонуем в компонент



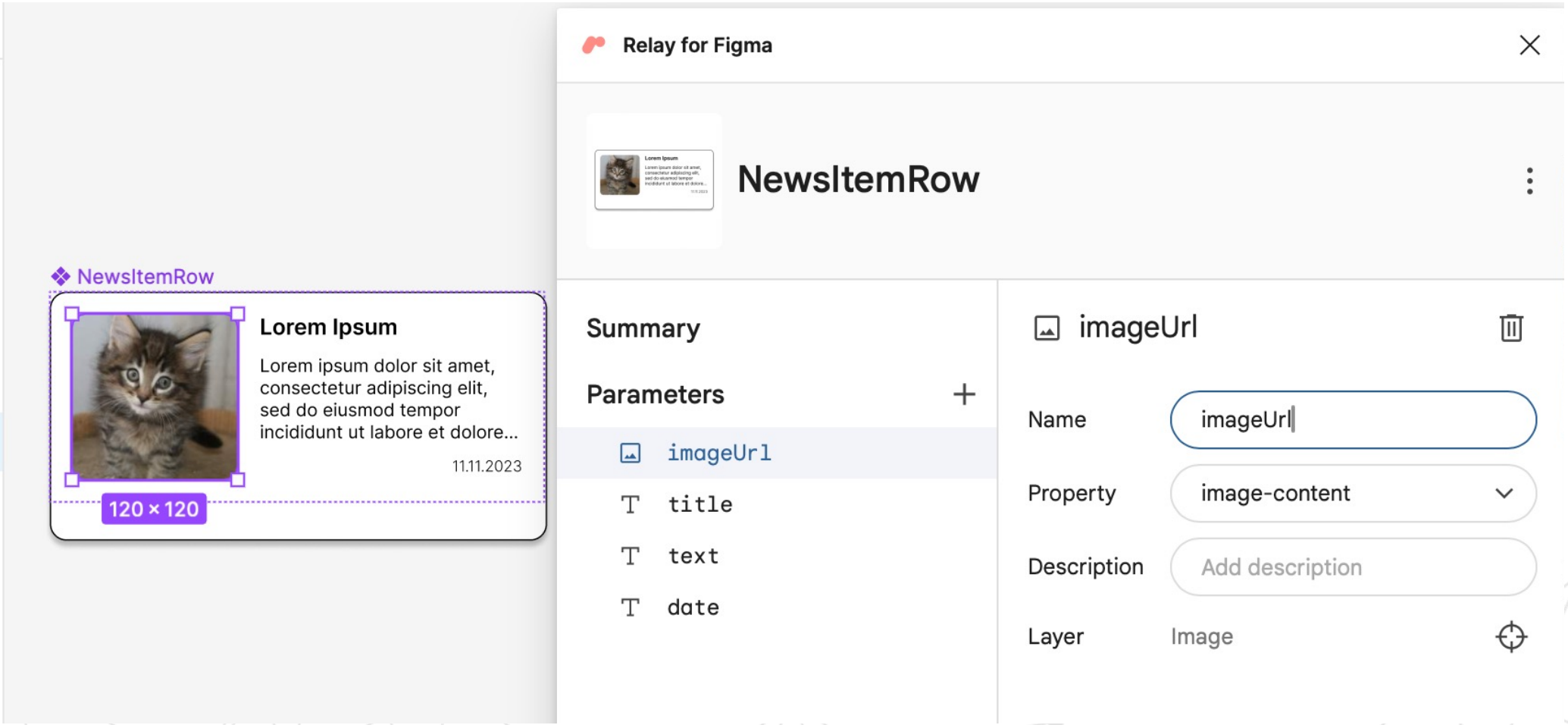
Настройка компонента

Переходим через контекстное меню



Настройка компонента

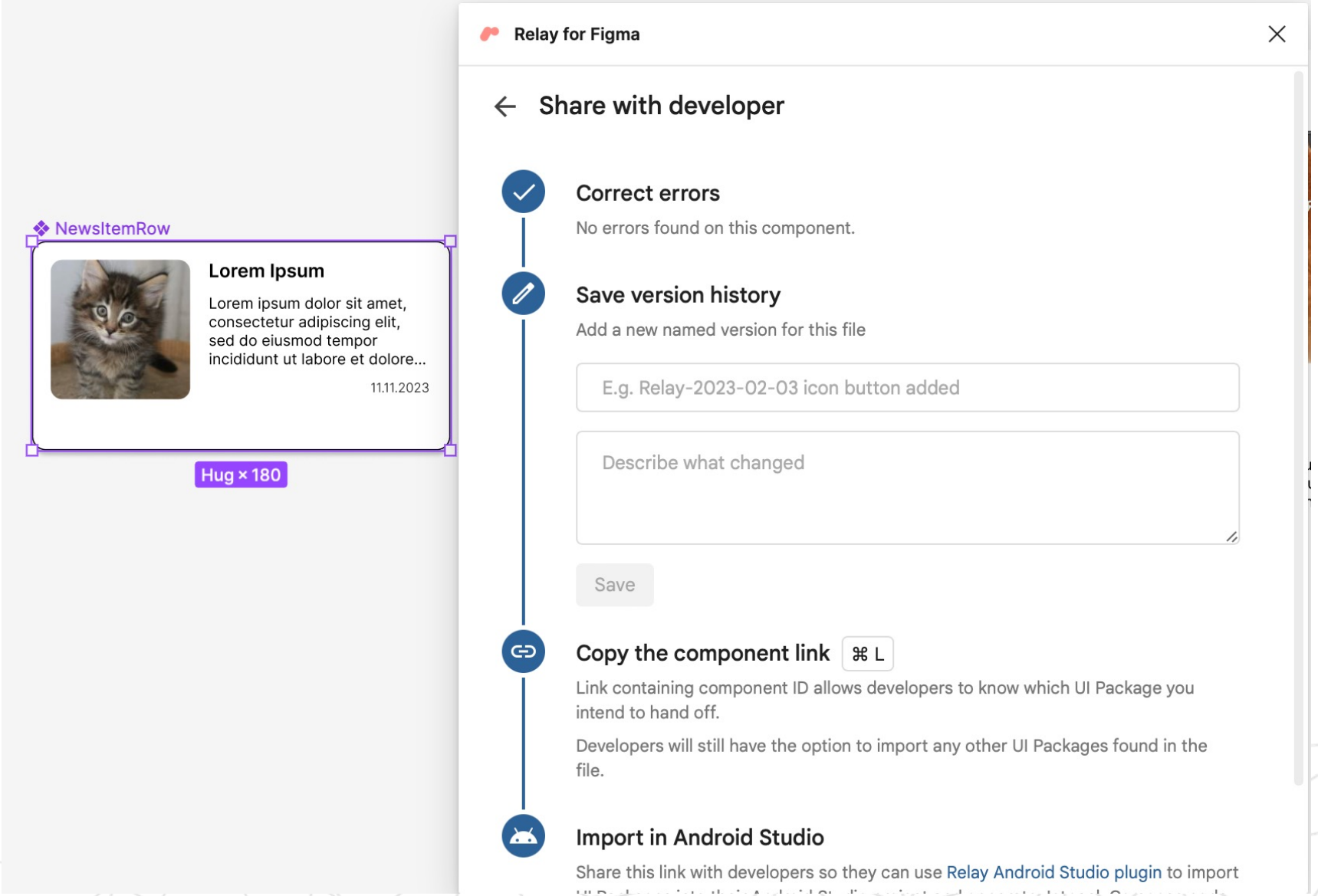
Добавляем свойства



Настройка компонента

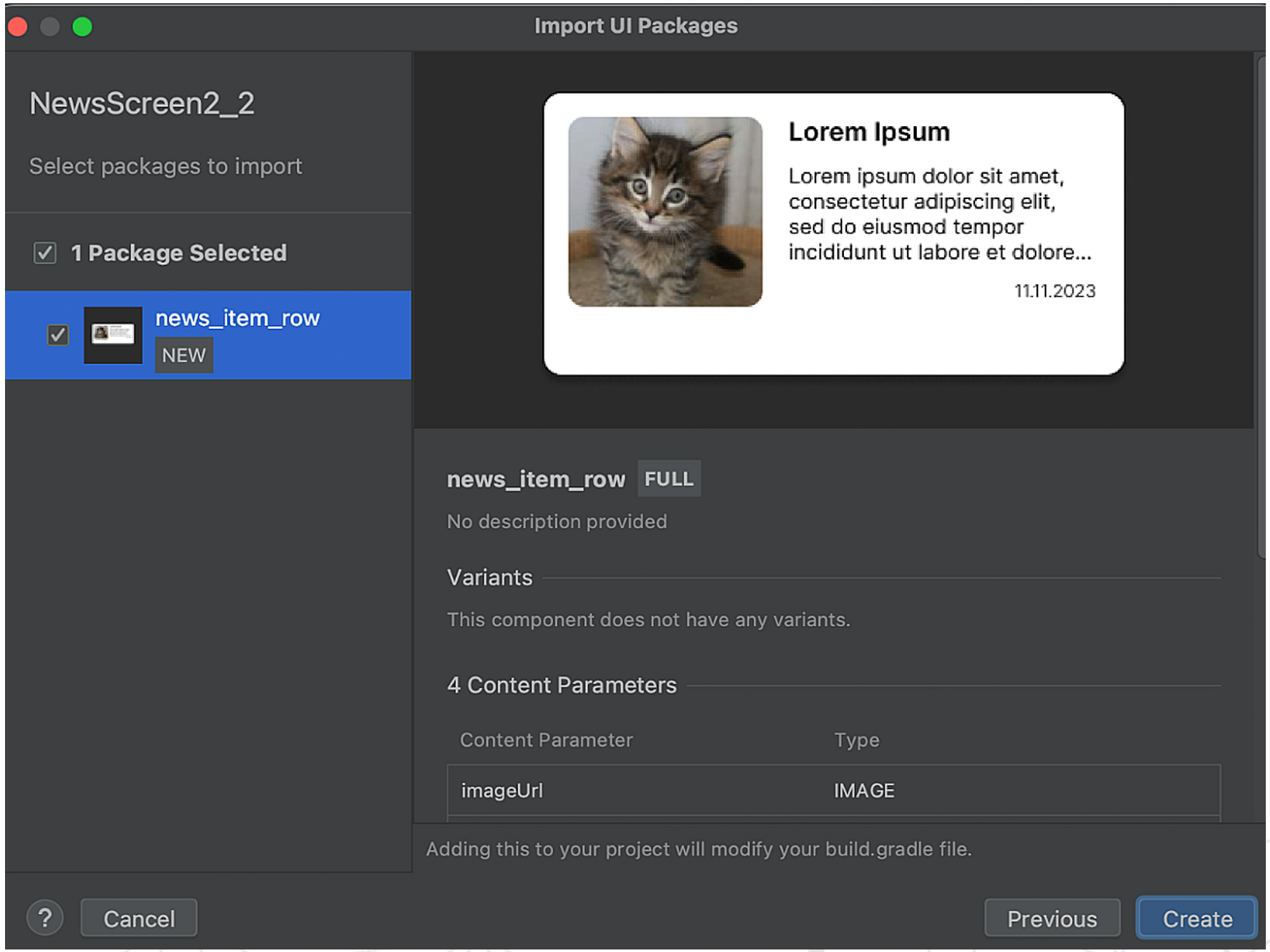
Share with developer

Cmd + L/ Ctrl+ L



Импорт компонента

Нажмите Create для завершения импорта и создания компонента



Json код компонента

News_item_row.json

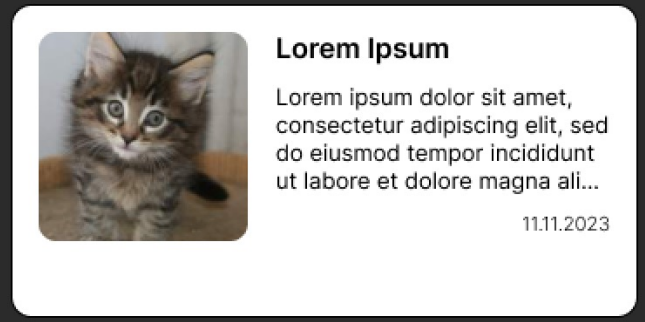
Json код компонента

News_item_row.json

news_item_row.json

```
{
  "name": "news_item_row",
  "version": 32,
  "source-key": {
    "type": "figma",
    "file": "S6zUsU32ma50ZnhAuva7qf",
    "node": "10:2",
    "version": "4486181108",
    "component-id": "439b8f16eae69f70511dfd2b1708e
  },
  "default": "NewsItemRow",
  "design": {
    "atoms": [
      {
        "type": "group",
        "id": "top_level",
        "root": "true"
      },
      {
        "type": "group",
        "id": "Content"
      },
      {
        "type": "group",
```

UI Package



news_item_row

No description provided

Package Options

Mapping file — Generate Mapping File

Parents of this package will invoke the generated code

Variants

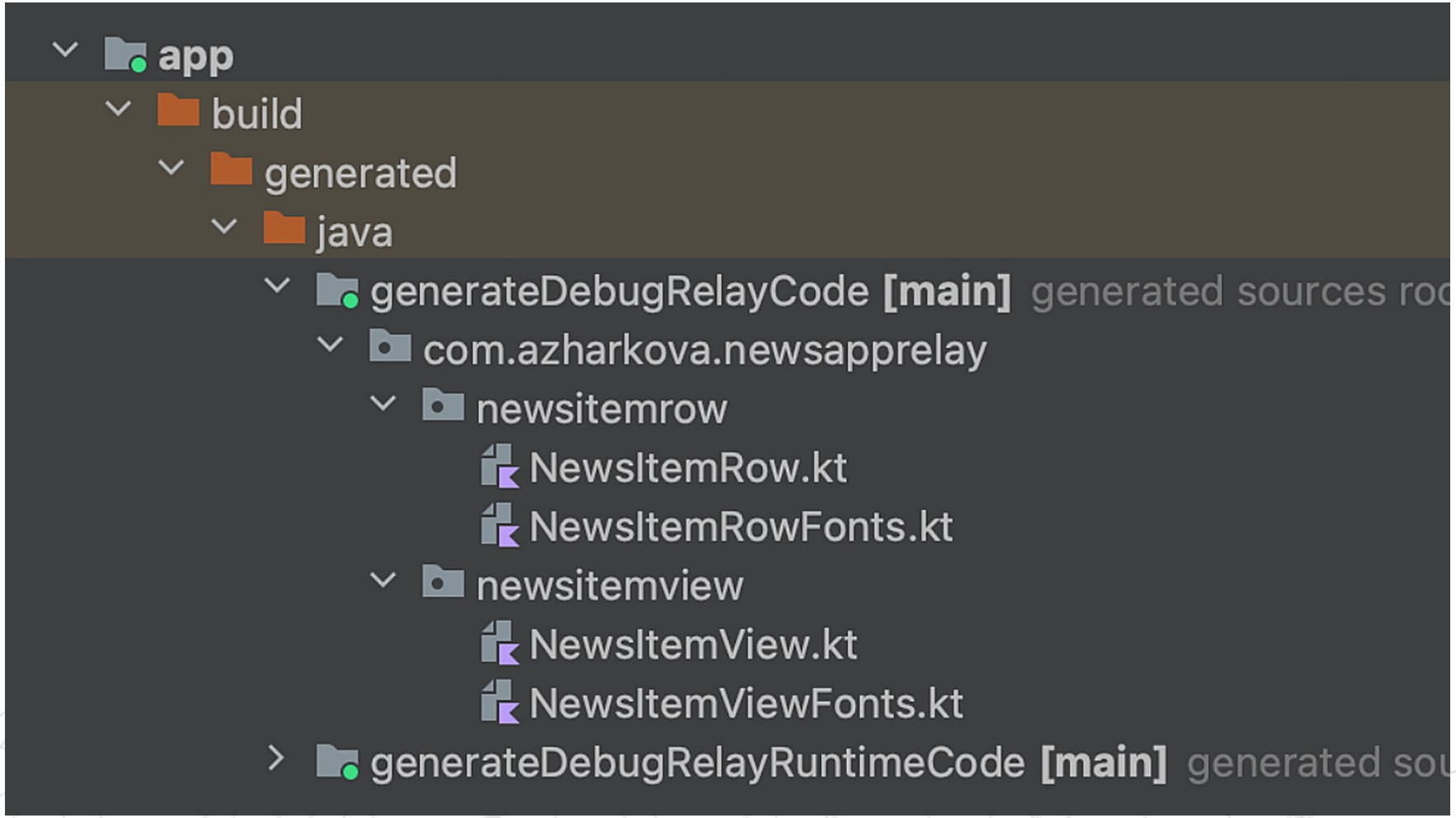
This component does not have any variants.

4 Content Parameters

Content Parameter	Type
image	IMAGE
title	TEXT
text	TEXT
date	TEXT

Сгенерированные файлы

Сгенерированные компоненты доступны в app – build - generated



Сгенерированный код

Generated source files should not be edited. The changes will be lost when sources are regenerated.

Code Split

Debugger/Profiler Unsupported

```
1   package com.azharkova.newsapprelay.newsitemrow
2
3   import ...
32
33  /**
34   * This composable was generated from the UI Package 'news_item_row'.
35   * Generated code; do not edit directly
36   */
37  @Composable
38  fun NewsItemRow(
39      modifier: Modifier = Modifier,
40      image: Painter = EmptyPainter(),
41      title: String = "",
42      text: String = "",
43      date: String = ""
44  ) {
45      TopLevel(
46          modifier = modifier
47      ) {...}
67  }
68
```

Что внутри. Relay

Под капотом Composable содержат Relay-элементы

```
fun NewsItemRow(
    modifier: Modifier = Modifier,
    image: Painter = EmptyPainter(),
    title: String = "",
    text: String = "",
    date: String = "",
) {
    TopLevel(
        modifier = modifier
    ) {
        Content(/*...*/) {
            Image(image = image)
            TextGroup(/*...*/) {
                Title(
                    title = title,
                    modifier = Modifier.rowWeight(1.0f)
                )
                Text(
                    text = text,
                    modifier = Modifier.rowWeight(1.0f)
                )
                Date(
                    date = date,
                    modifier = Modifier.rowWeight(1.0f)
                )
            }
        }
        /*...*/
    }
}
```

Правим адаптивность

News_item_row.json

Вносим изменения в констрейнты

```
{
    "id": "Text",
    "size-constraints": {
      "width-constraints": {
        "sizing-mode": "proportional",
        "value": "1.0"
      },
      "height-constraints": {
        "sizing-mode": "shrink"
      }
    }
    //...
}
```

Правим адаптивность

News_item_row.json

Вносим изменения в констрейнты

Generated source files should not be edited. The changes will be lost when sources are regenerated.

```
124    @Composable
125    fun Text(
126        text: String,
127        modifier: Modifier = Modifier
128    ) {
129        RelayText(
130            content = text,
131            fontSize = 13.0.sp,
132            fontFamily = inter,
133            color = Color(
134                alpha = 255,
135                red = 0,
136                green = 0,
137                blue = 0
138            ),
139            height = 1.2102272327129657.em,
140            textAlign = TextAlign.Left,
141            maxLines = 4,
142            modifier = modifier.fillMaxWidth( fraction: 1.0f)
143        )
144    }
```

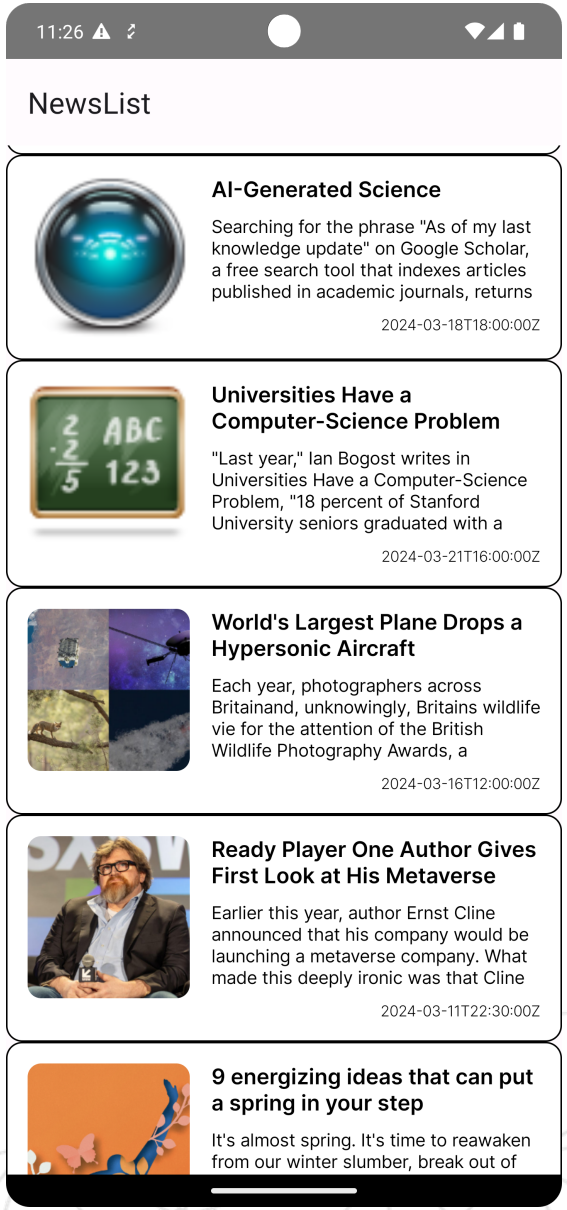
Подключаем в существующий экран

```
@Composable
fun NewsListScreen() {
    /* VM + data
    ...*/

    LazyColumn {
        items(data.orEmpty()){
            NewsItemRow(
                image = rememberAsyncImagePainter(it.urlToImage),
                title = it.title.orEmpty(),
                text = it.content.orEmpty(),
                date = it.publishedAt.orEmpty()
            )
        }
    }
}
```

Запускаем

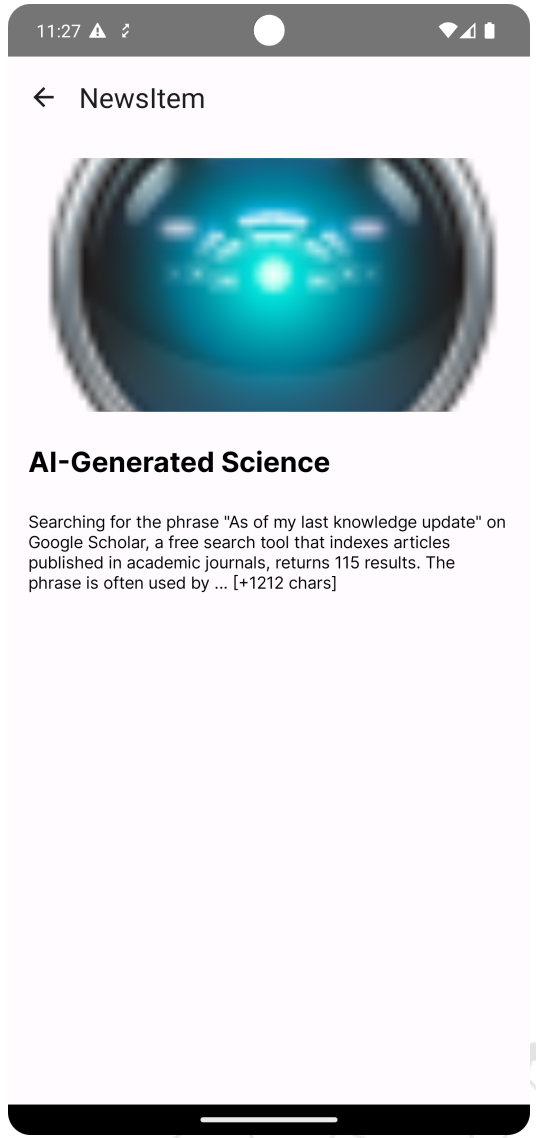
https://github.com/anioutkazharkova/NewsAppRelay



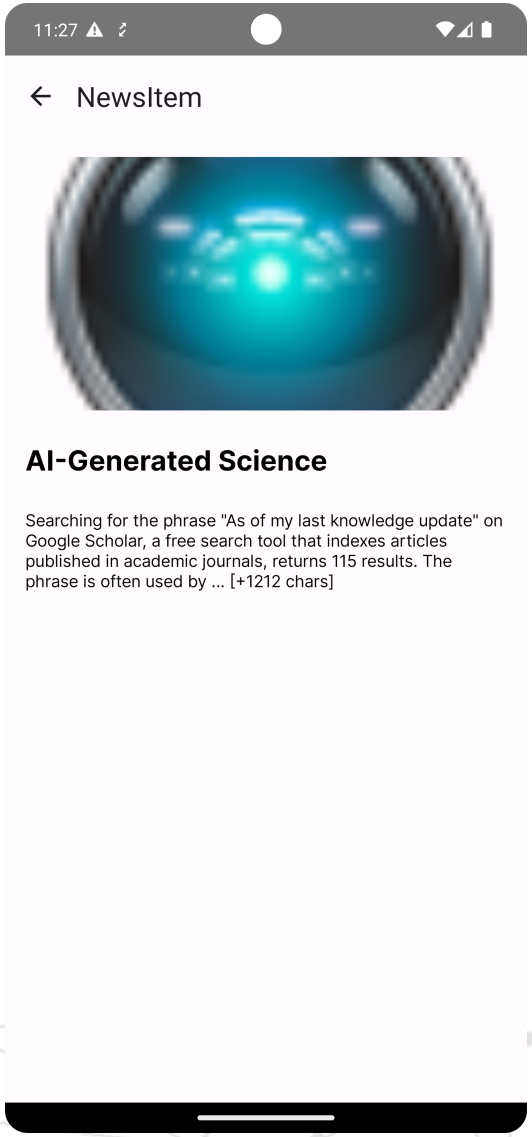
Добавим экран деталей новости

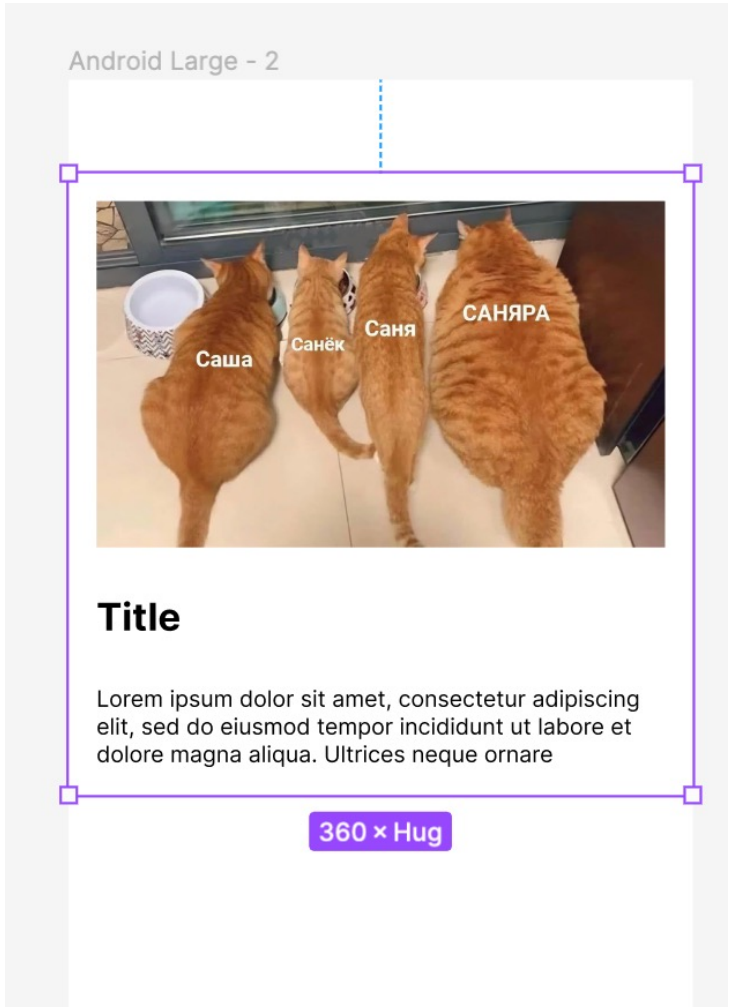
Экран новости

Добавим экран новости



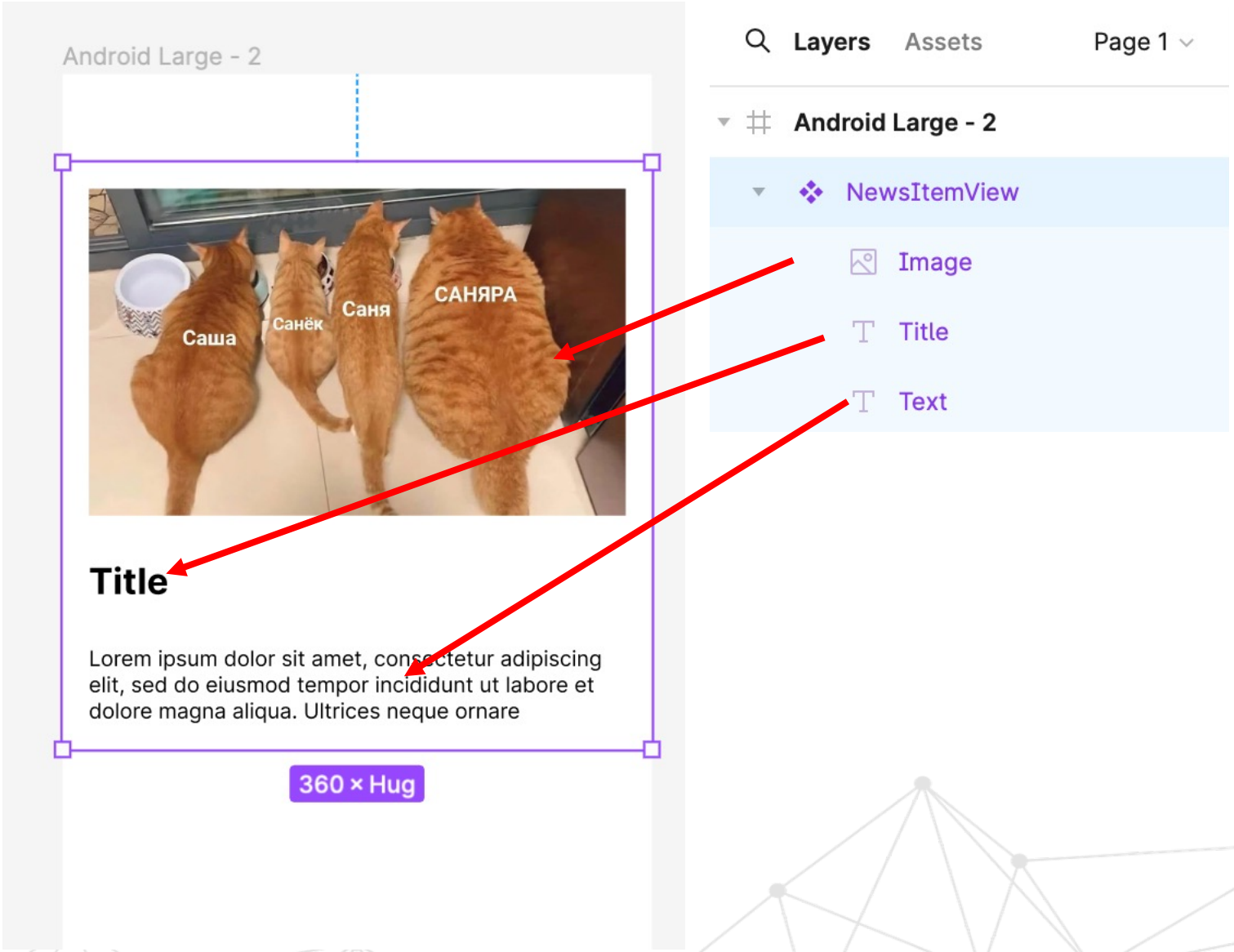
Экран новости





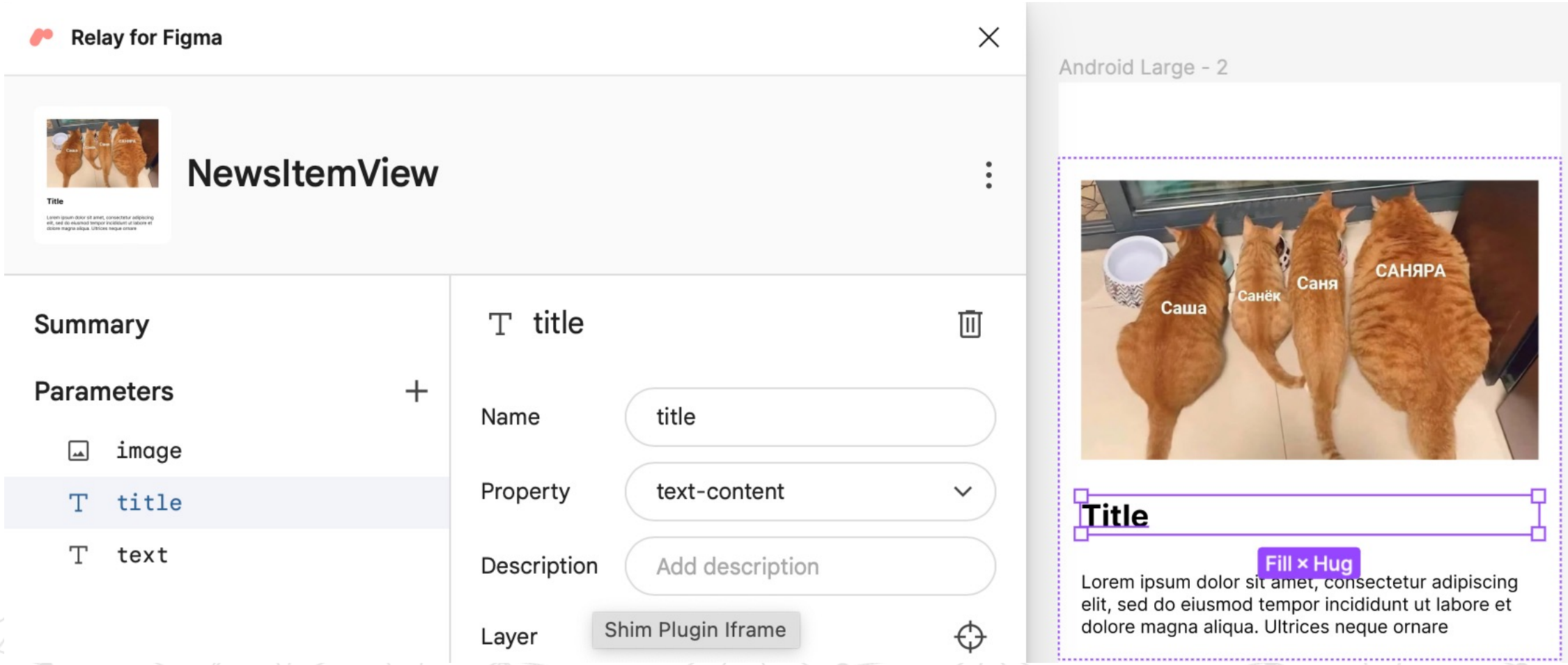
Экран новости

Для портирования дизайна всего экрана превращаем в компонент



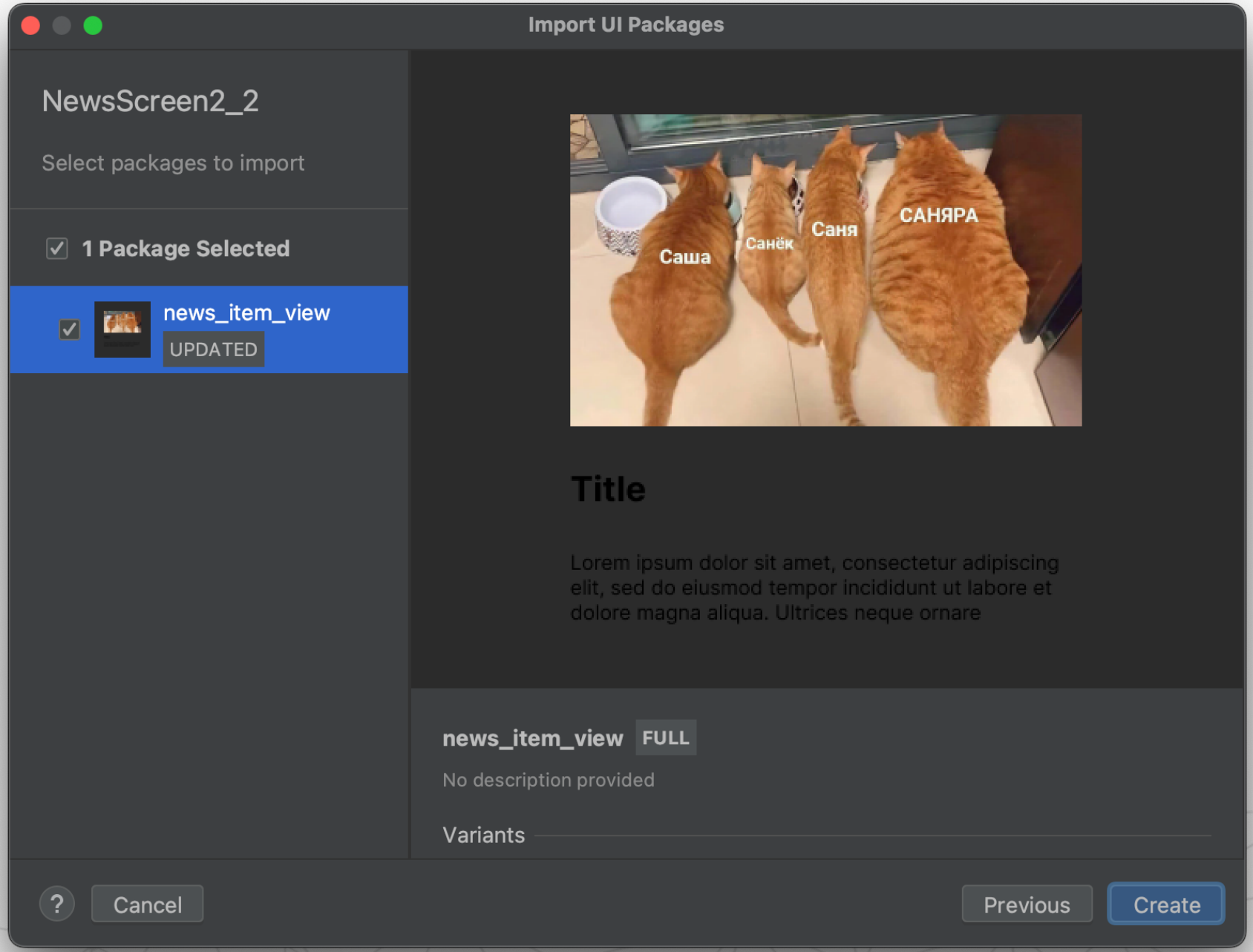
Экран новости

Настраиваем параметры и свойства компонента



Импорт экрана

Импортируем
компонент с экраном



Сгенерированный код

При необходимости правим констрейнты в json

NewsItemView.kt | config.json

Files under the "build" folder are generated and should not be edited.

```
32  @Composable
33  fun NewsItemView(
34      modifier: Modifier = Modifier,
35      image: Painter = EmptyPainter(),
36      title: String = "",
37      text: String = ""
38  ) {
39      TopLevel(modifier = modifier) { this: RelayContainerScope
40          Image(
41              image = image,
42              modifier = Modifier.rowWeight( weight: 1.0f)
43          )
44          Title(
45              title = title,
46              modifier = Modifier.rowWeight( weight: 1.0f)
47          )
48          Text(
49              text = text,
50              modifier = Modifier.rowWeight( weight: 1.0f)
51          )
52      }
53  }
```

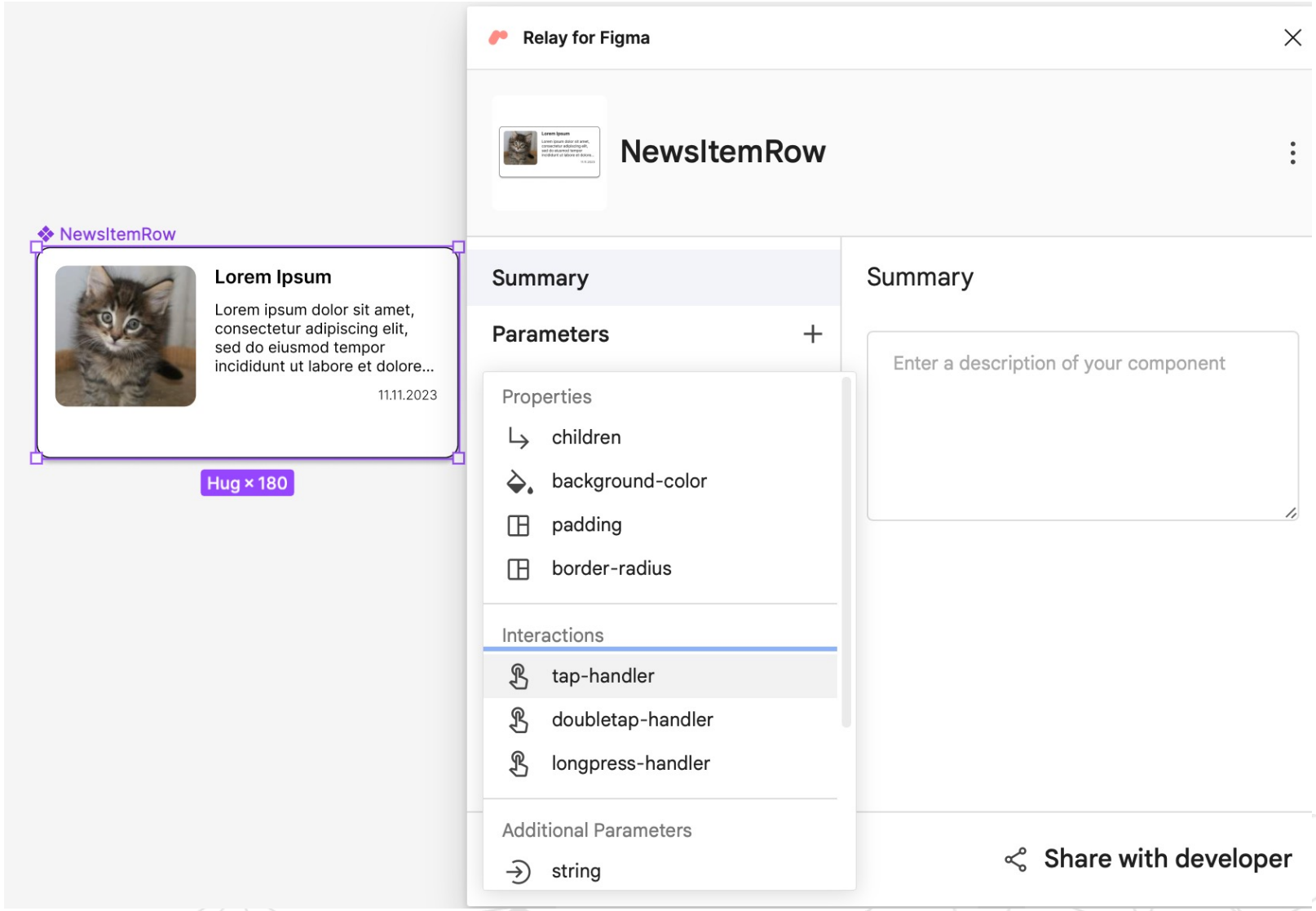
Подключаем в код

```
@Composable
fun NewsItemDataView(viewModel: NewsItemVM) {
    val data by viewModel.model.collectAsState()
    NewsItemView(
        image = rememberAsyncImagePainter(data?.urlToImage),
        title = data?.title.orEmpty(),
        text = data?.content.orEmpty()
    )
}
```

Добавим интерактивности (и навигацию)

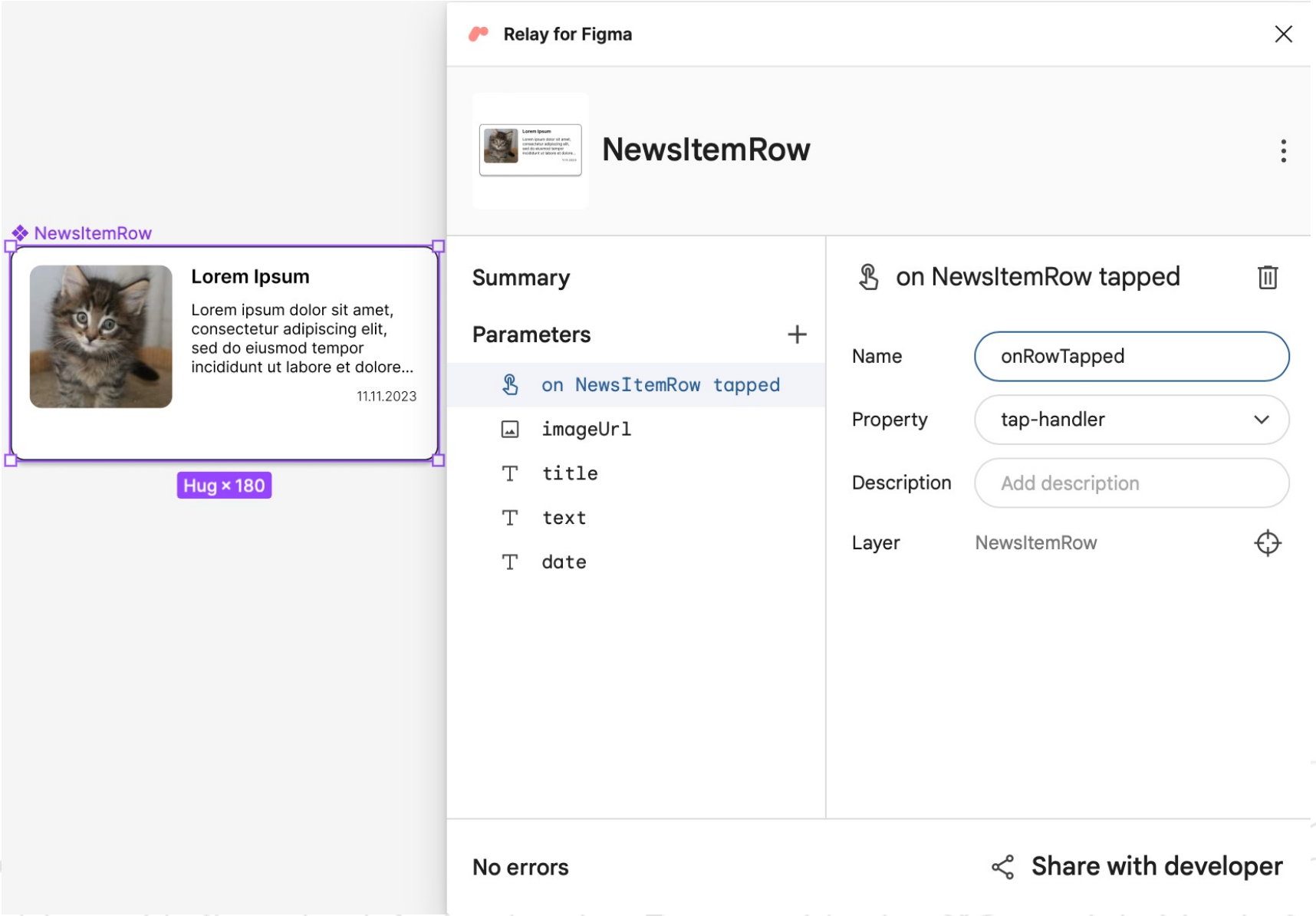
Интерактивность

Добавляем tap-handler
на главный элемент



Интерактивность

Добавляем tap-handler
на главный элемент



Интерактивность

Делаем свой Composable-обертку

combinedClickable

```
@OptIn(ExperimentalFoundationApi::class)
@Composable
fun WrapperComposable(modifier: Modifier,
    onClick: (() -> Unit)? = null) {

    HelloCard(modifier.combinedClickable(enabled = true,
            onClick = { onClick?.invoke() }))
}
```

Сгенерированный код с кликером

Files under the "build" folder are generated and should not be edited.

```
1   package com.azharkova.newsapprelay.newsitemrow
2
3   import ...
32
33  /**
34   * This composable was generated from the UI Package 'news_item_row'.
35   * Generated code; do not edit directly
36   */
37  @Composable
38  fun NewsItemRow(
39      modifier: Modifier = Modifier,
40      image: Painter = EmptyPainter(),
41      title: String = "",
42      text: String = "",
43      date: String = "",
44      onRowTapped: () -> Unit = {}
45  ) {
46      TopLevel(
47          onRowTapped = onRowTapped,
48          modifier = modifier
49      ) {...}
69  }
```

Подключаем в существующий код

```
@Composable
fun NewsListScreen(navigate: (NewsItemModel)->Unit) {
    /* VM + data
    ...*/

    LazyColumn {
        items(data.orEmpty()){
            NewsItemRow(
                image = rememberAsyncImagePainter(it.urlToImage),
                title = it.title.orEmpty(),
                text = it.content.orEmpty(),
                date = it.publishedAt.orEmpty(),
                onRowTapped = {navigate.invoke(it)}
            )
        }
    }
}
```

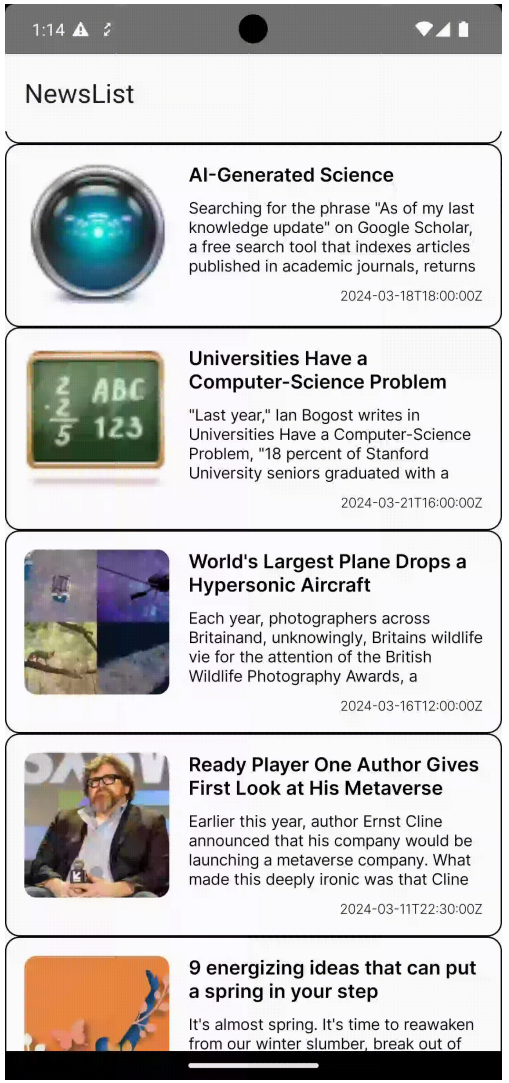
Настраиваем навигацию

Прописываем в NavHost

```
NavHost(
    navController = navController,
    startDestination = NewsScreen.NewsList.name,
    modifier = Modifier.padding(paddingValues)
) {
    composable(route = NewsScreen.NewsList.name) {
        NewsListScreen() {
            navController.navigate(
                NewsScreen.NewsItem.name,
                args = bundleOf(
                    "ITEM" to it
                )
            )
        }
    }
    composable(route = NewsScreen.NewsItem.name) {
        NewsItemDataView(viewModel =
        NewsItemVM(it.arguments?.getParcelable("ITEM"))
        )
    }
}
```

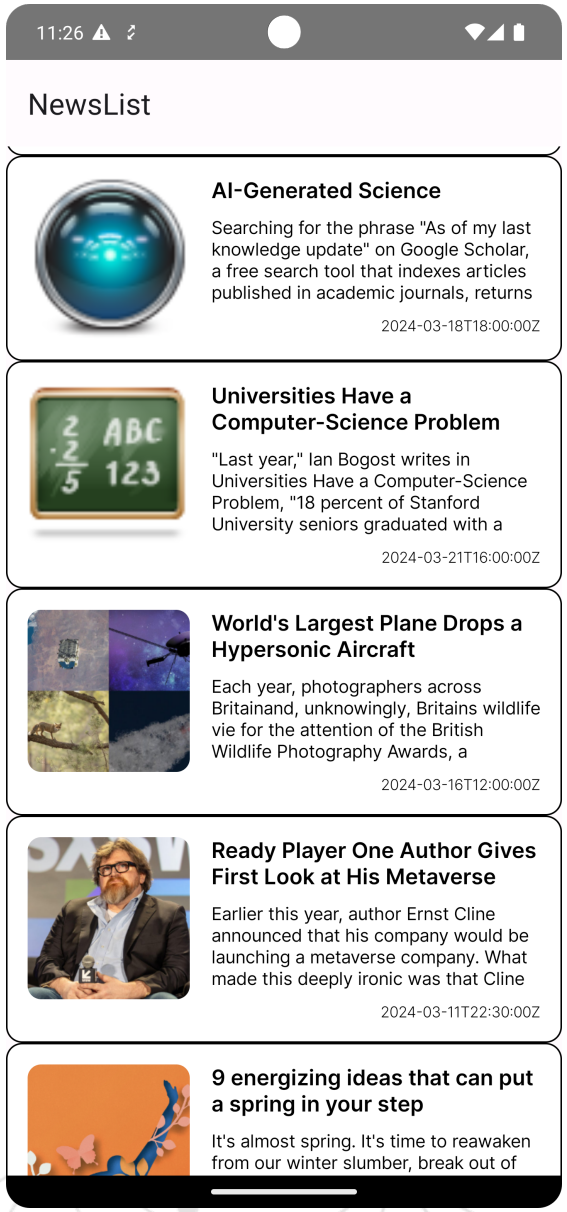
Проверяем работу

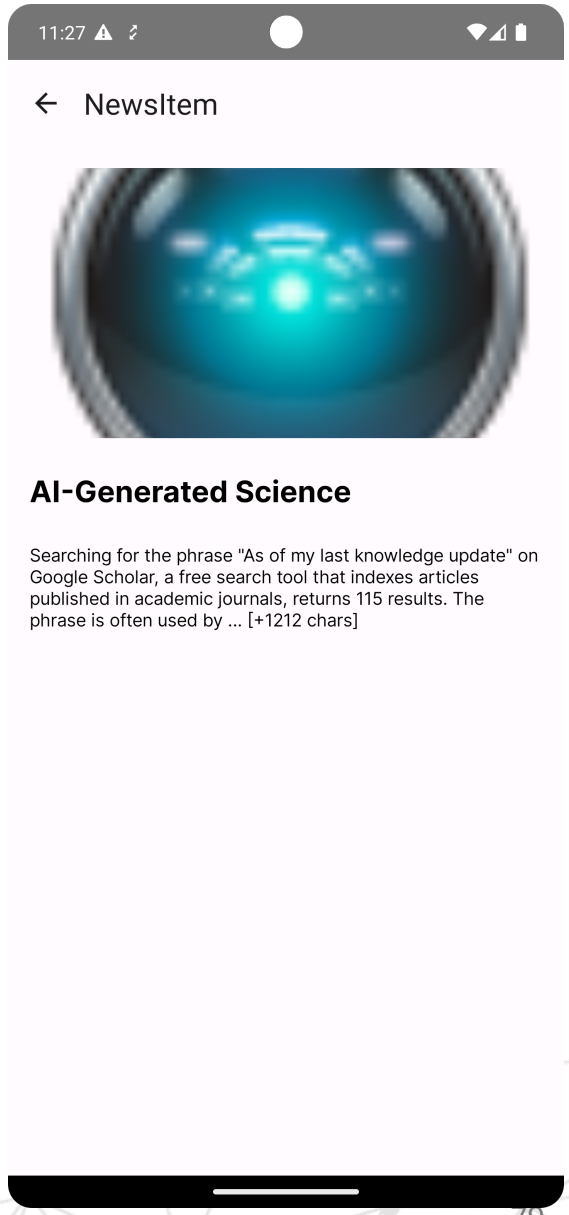
https://github.com/anioutkazharkova/NewsAppRelay



Все работает

https://github.com/anioutkazharkova/NewsAppRelay





Summary.

- Плагин поддерживает настройку и группировку свойство
- Поддерживает добавление событий
- Пока только для Android
- Сложно делать UI на весь экран сразу

Sources

- https://github.com/anioutkazharkova/NewsAppRelay
- https://www.figma.com/community/plugin/1041056822461507786/relay-for-figma
- https://relay.material.io
- https://www.youtube.com/watch?v=vBNmeiHlDHE

Спасибо за внимание!

@anioutkajarkova

azharkova
prettygeeknotes