



# 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

# 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,
                )
            }
        }
    }
}
```



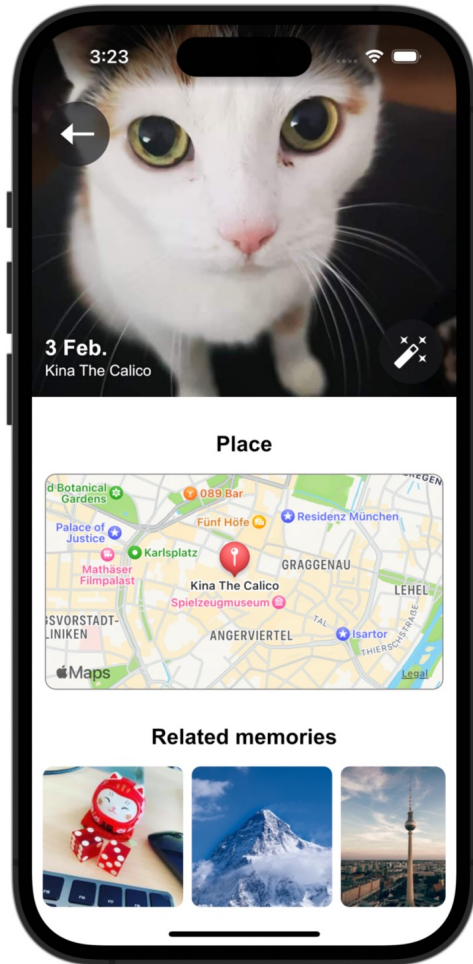
Jetpack  
Compose

# Jetpack Compose

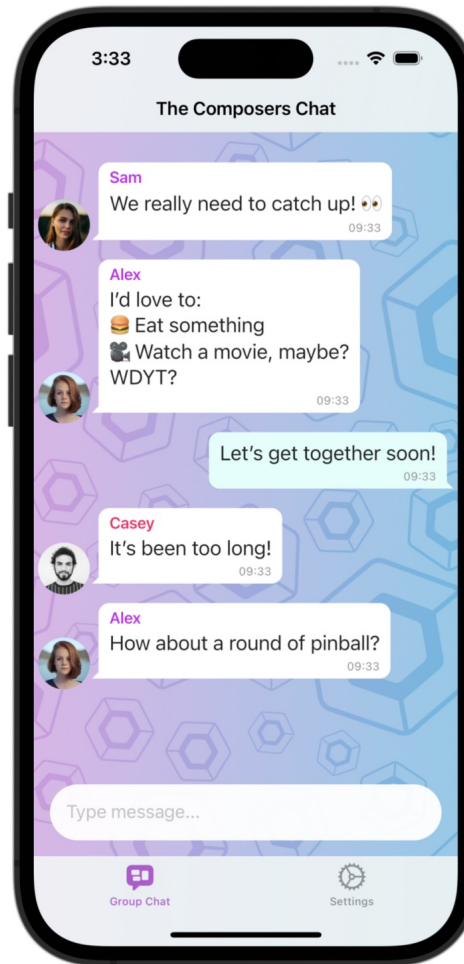
- View – State
- Декларативная навигация
- @Composable функции
- Под капотом стандартные контролы

# 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

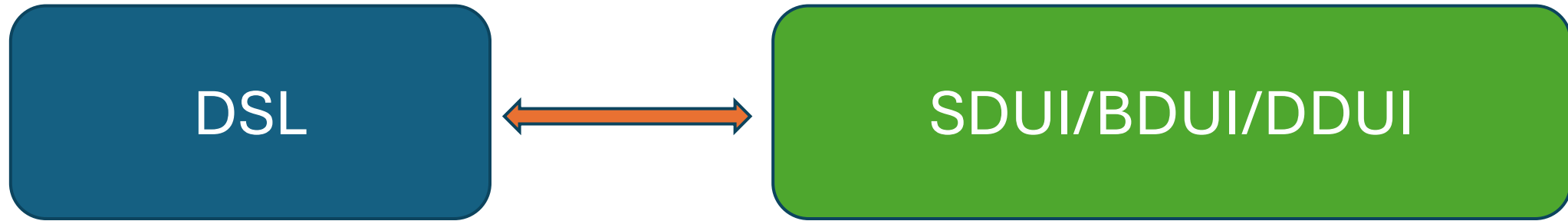
Floating SwiftUI  
TextField

# Compose iOS

Построено на Jetpack Compose

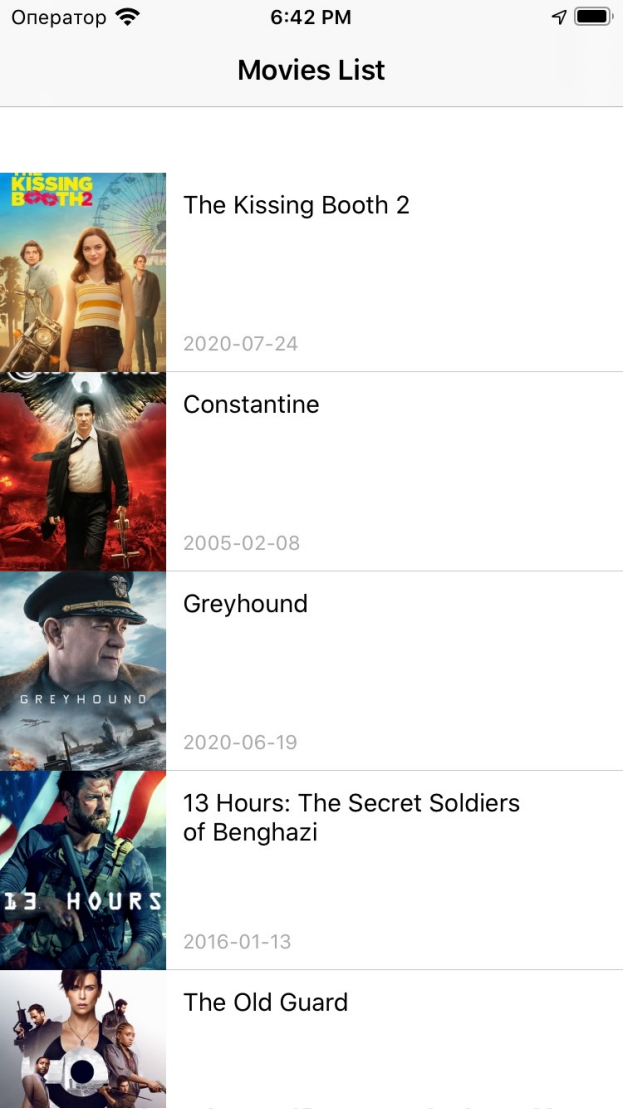
```
@Composable
fun App() {
    MaterialTheme {
        var showImage by remember { mutableStateOf(false) }
        Column(Modifier.fillMaxWidth(), horizontalAlignment
            = Alignment.CenterHorizontally) {
            Button(onClick = {
                showImage = !showImage
            }) {
                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

Server



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



Client



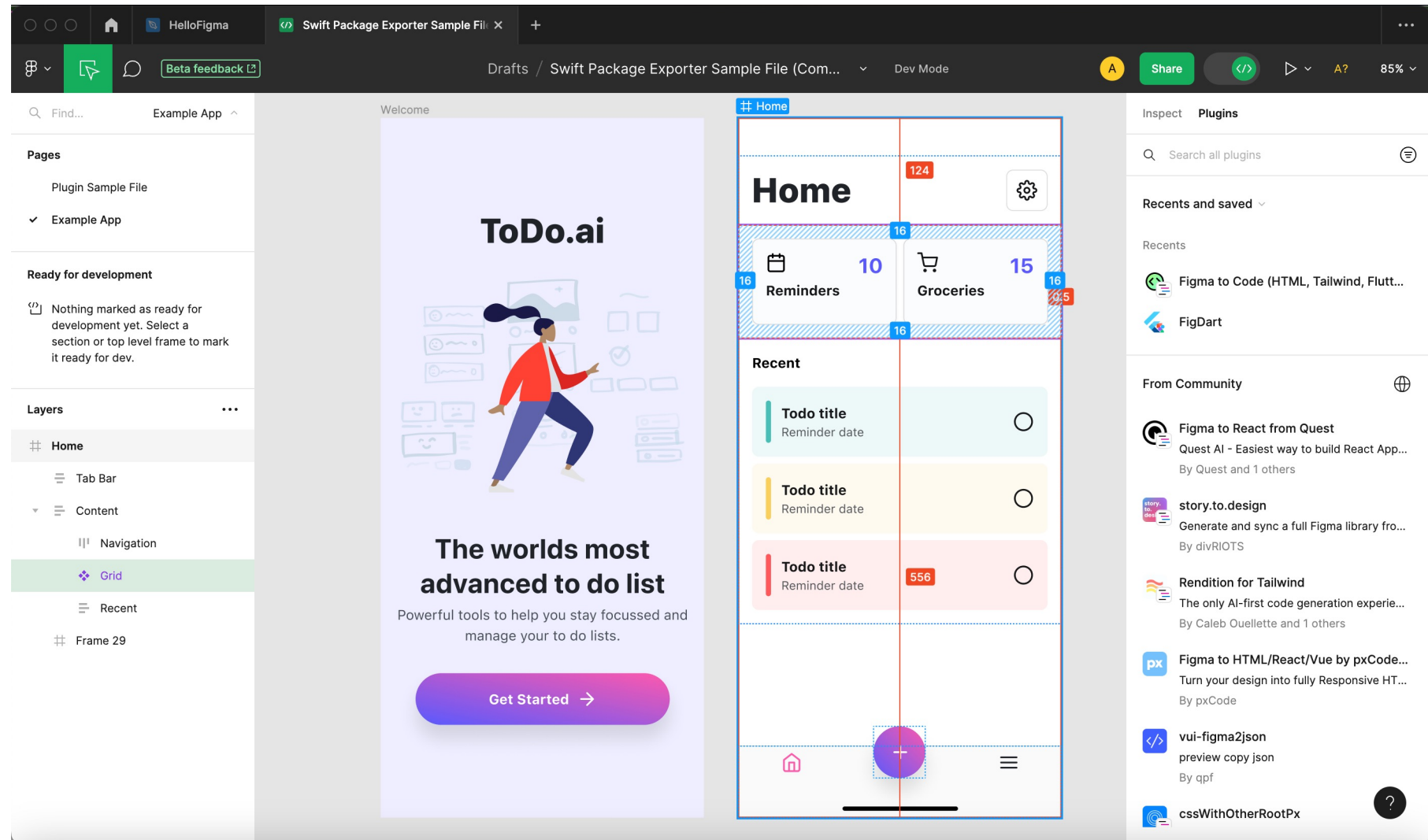


# SDUI/BDUI/DDUI. Вопросы

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

# Выход есть. Нам потребуется

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



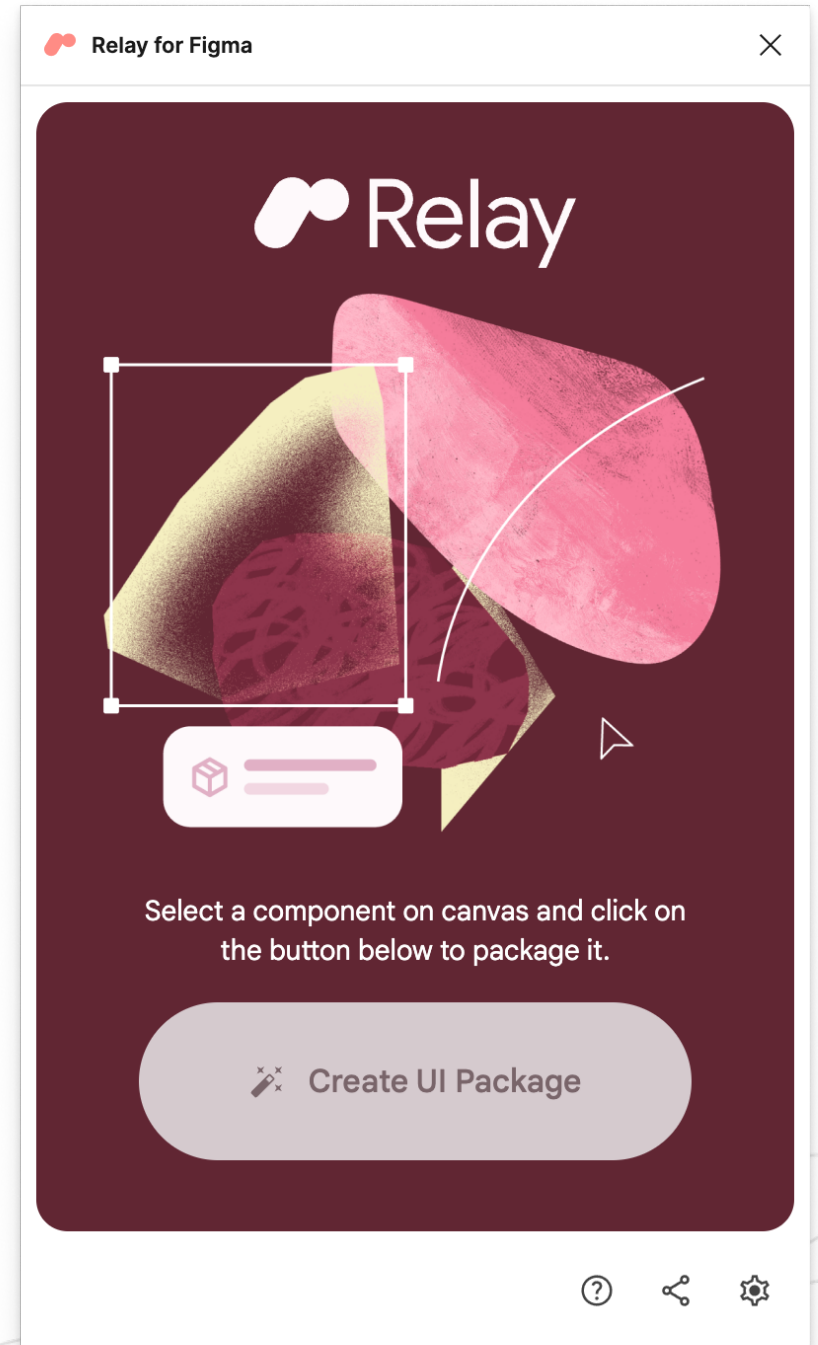
# Compose + Relay



# Relay

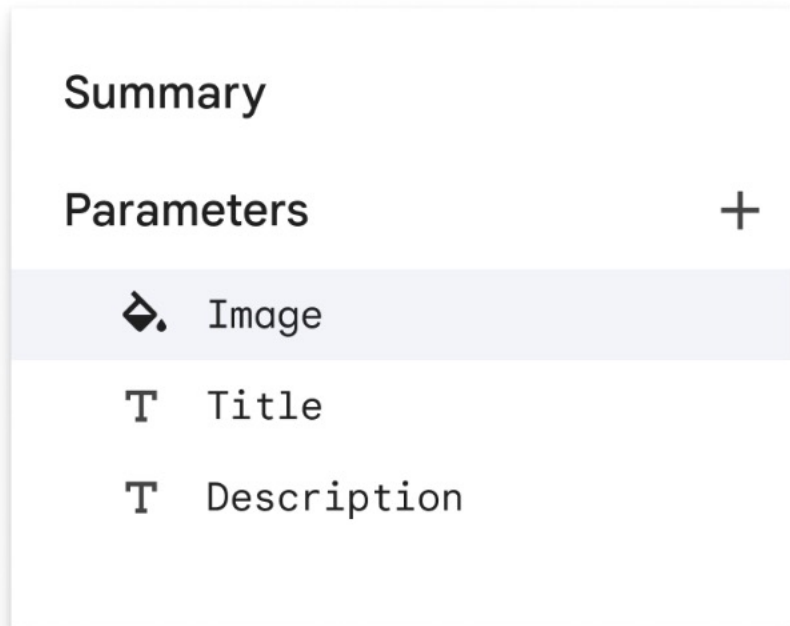
<https://relay.material.io/>

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



# Relay

<https://relay.material.io/>



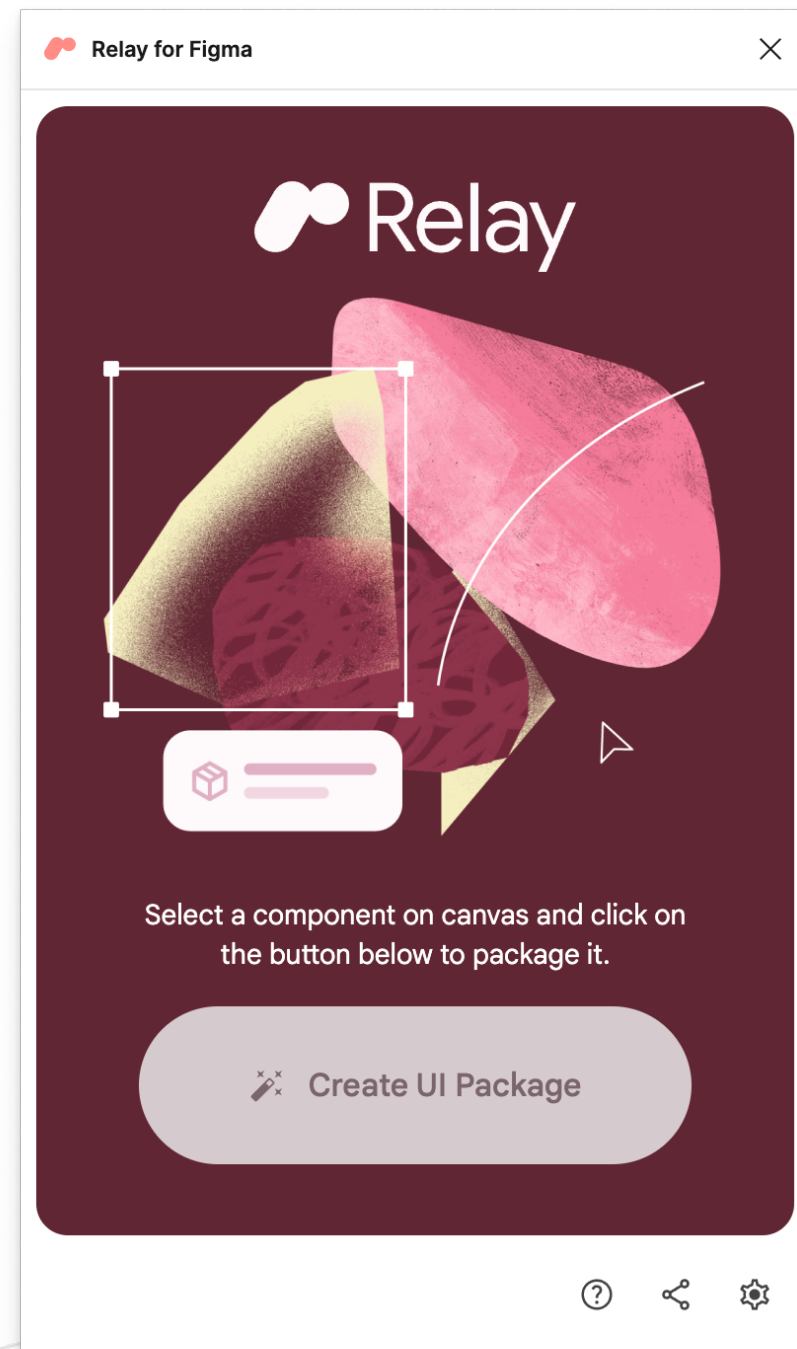
```
@Composable
fun StoryCard(
    image: Painter,
    title: Text,
    description: Text,
) {...}
```

# 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

The image shows the Figma Relay for Figma plugin page on the left and a screenshot of the plugin's configuration interface on the right. The page on the left includes the Material Design logo, the plugin name 'Relay for Figma', and statistics: 'Plugin • 734 • 26.4k users'. A green button labeled 'Open in Dev Mode' is visible. The screenshot on the right shows the 'Technique card' configuration window with a 'Parameters' list and an 'onCardTapped' configuration section.

**Relay for Figma**

Material Design

Relay for Figma

Plugin • 734 • 26.4k users

Open in Dev Mode

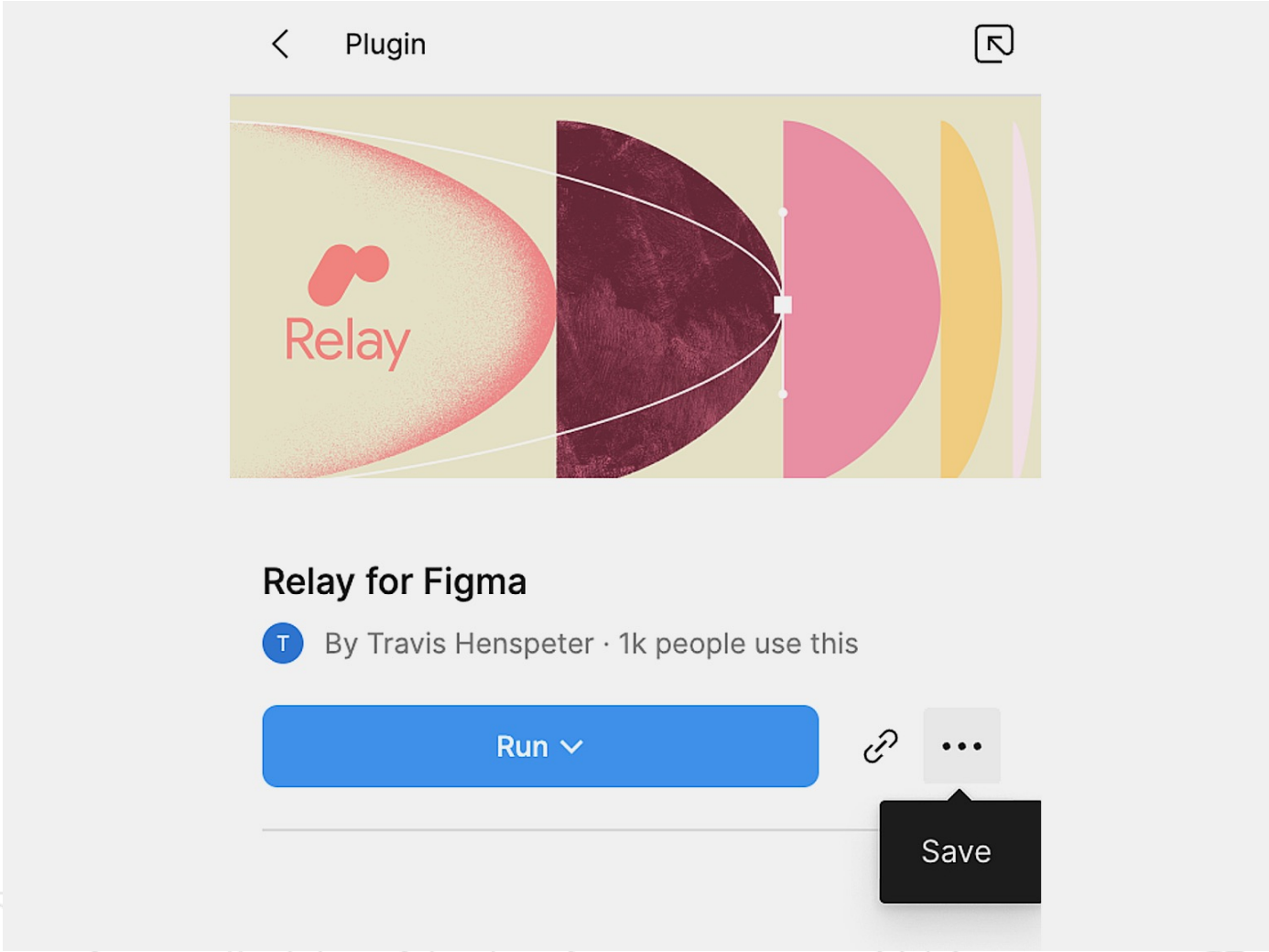
**Technique card**

Summary	onCardTapped
Parameters	Name: onCardTapped
onCardTapped	Property: tap-handler
imageContent	Description: Handles tap interaction
author	Layer: Technique card
title	
cycles	
summary	

No errors

Sharing with developer >

# Установка плагина Figma



# Токен

## Токен генерируется в настройках аккаунта в Figma

---

### Personal access tokens

Personal access tokens allow you to access your own data via the API. Do not give out your personal access tokens to anybody who you don't want to access your files.

#### [Generate new token](#)

🔑 relay\_token  
Never expires

Used 2 months ago

[Revoke access](#)

# Токен

## Токен генерируется в настройках аккаунта в Figma

### Personal access tokens

Personal access tokens allow you to access your own data via the API. Do not give out your personal access tokens to anybody who you don't want to access your files.

[Generate new token](#)

🔑 relay\_token ✕

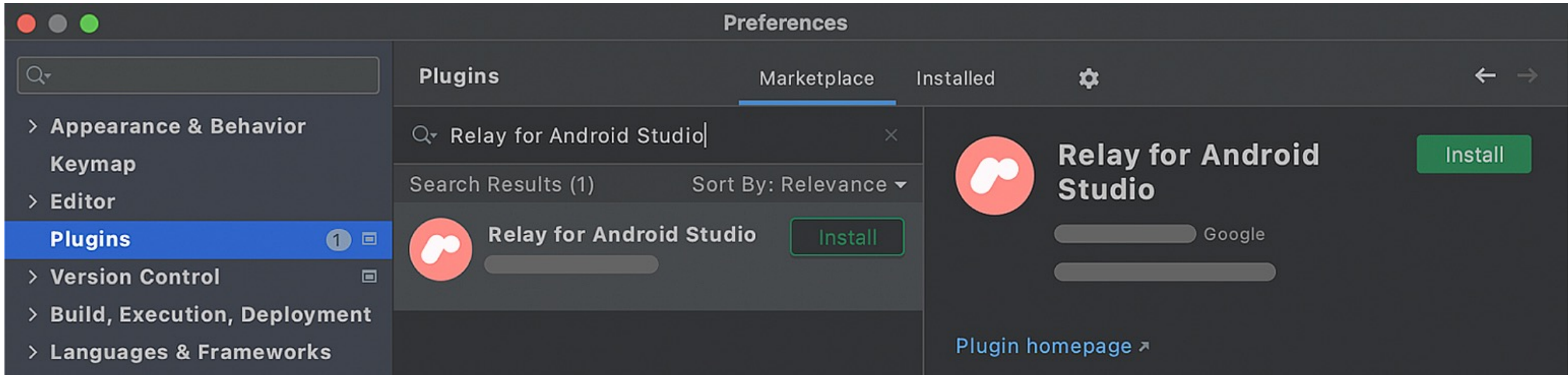
```
fig_1_0W_0L5V_1T_6K4WU_6_K07_30_D_1_1T_DW_7_HH
```

Copy this token. This is your only chance to do so!

### Connected apps

# Настройка Android Studio

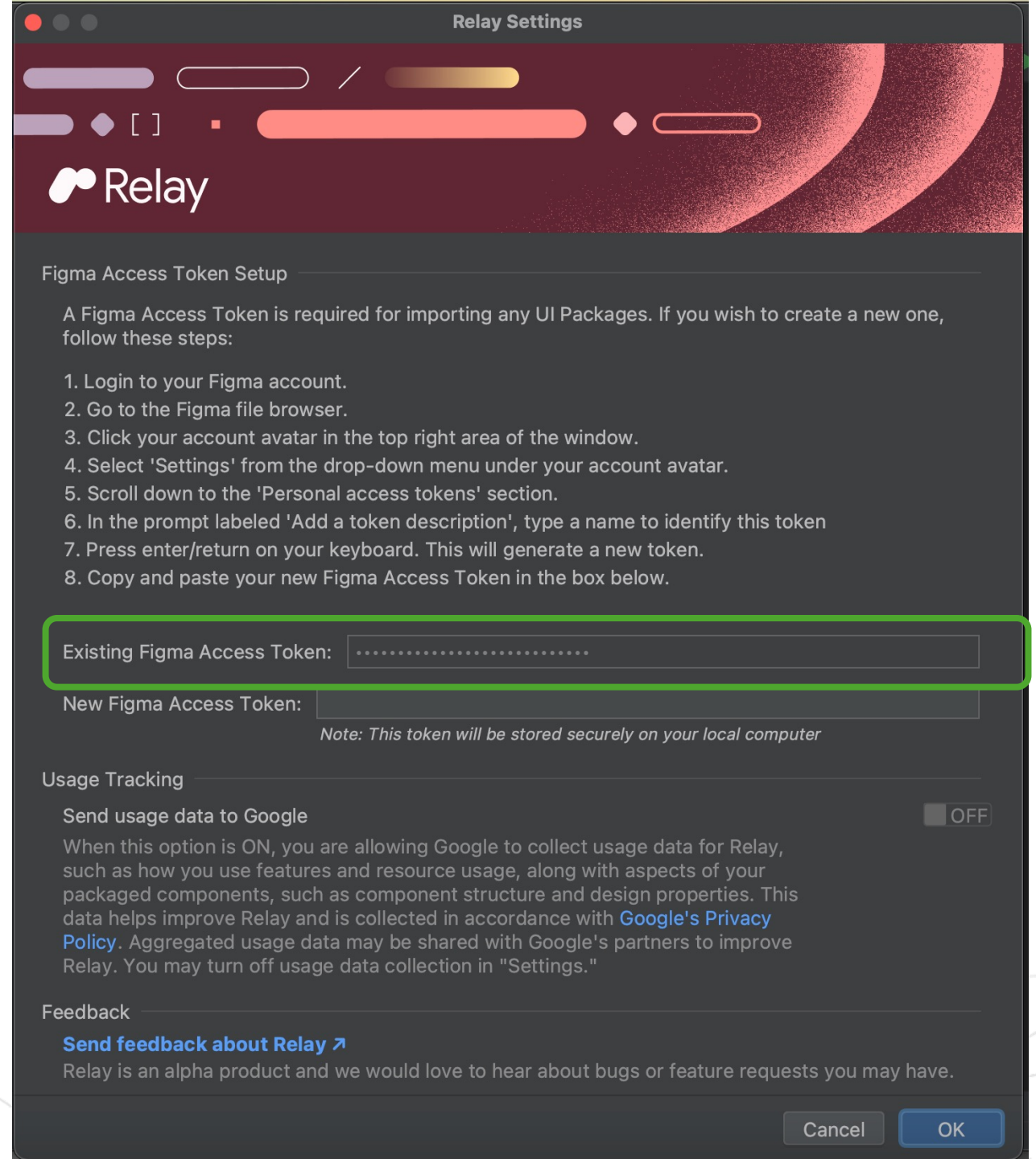
Установка плагина Preferences -> Plugins





# Токен

## Прописываем токен в Tools – Relay Settings





# Токен

## Прописываем токен в Tools – Relay Settings

7. Press enter/return on your keyboard. This will generate a new token.
8. Copy and paste your new Figma Access Token in the box below.

Existing Figma Access Token:

New Figma Access Token:

*Note: This token will be stored securely on your local computer*

### Usage Tracking

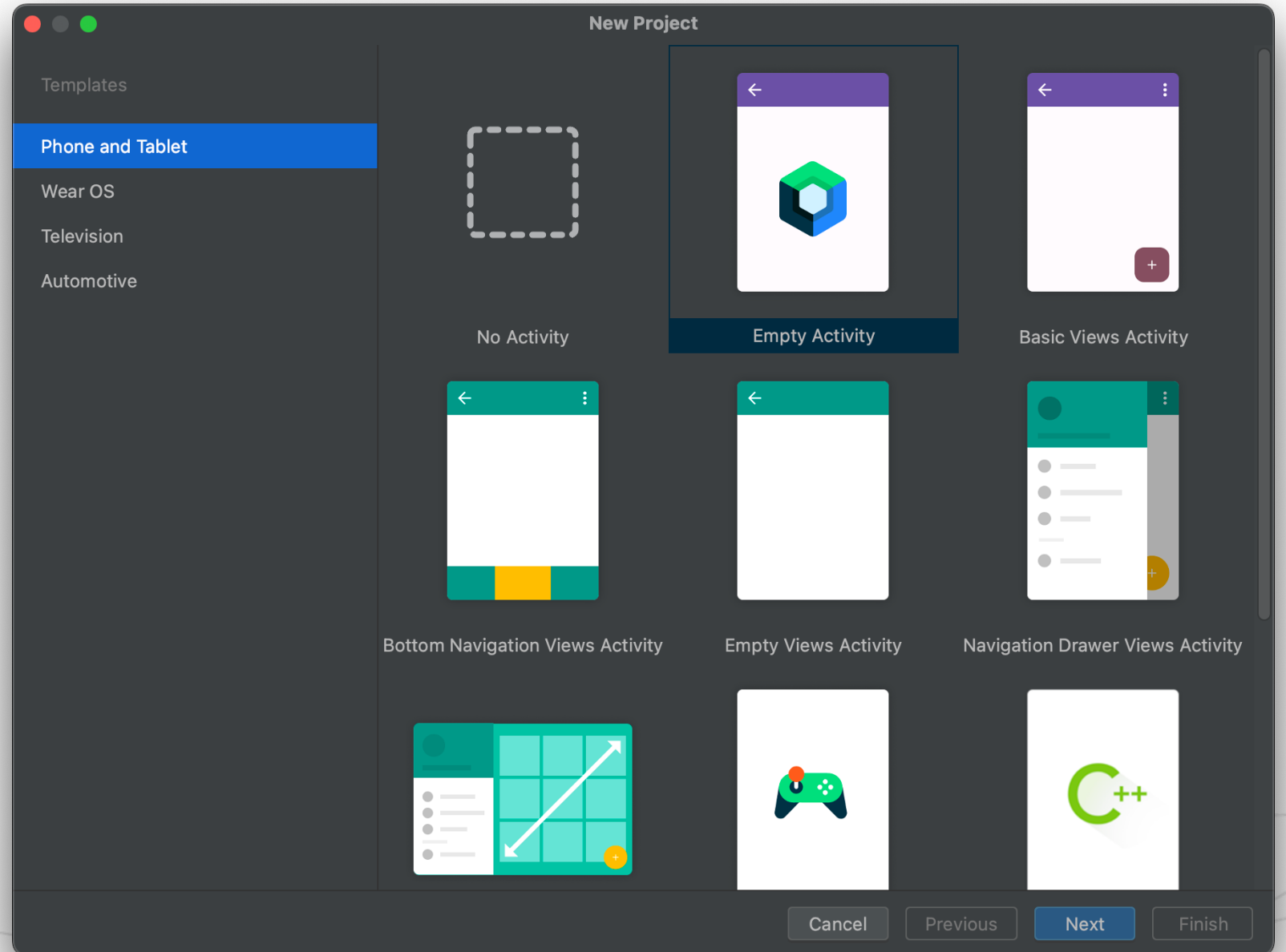
Send usage data to Google

OFF

When this option is ON, you are allowing Google to collect usage data for Relay.

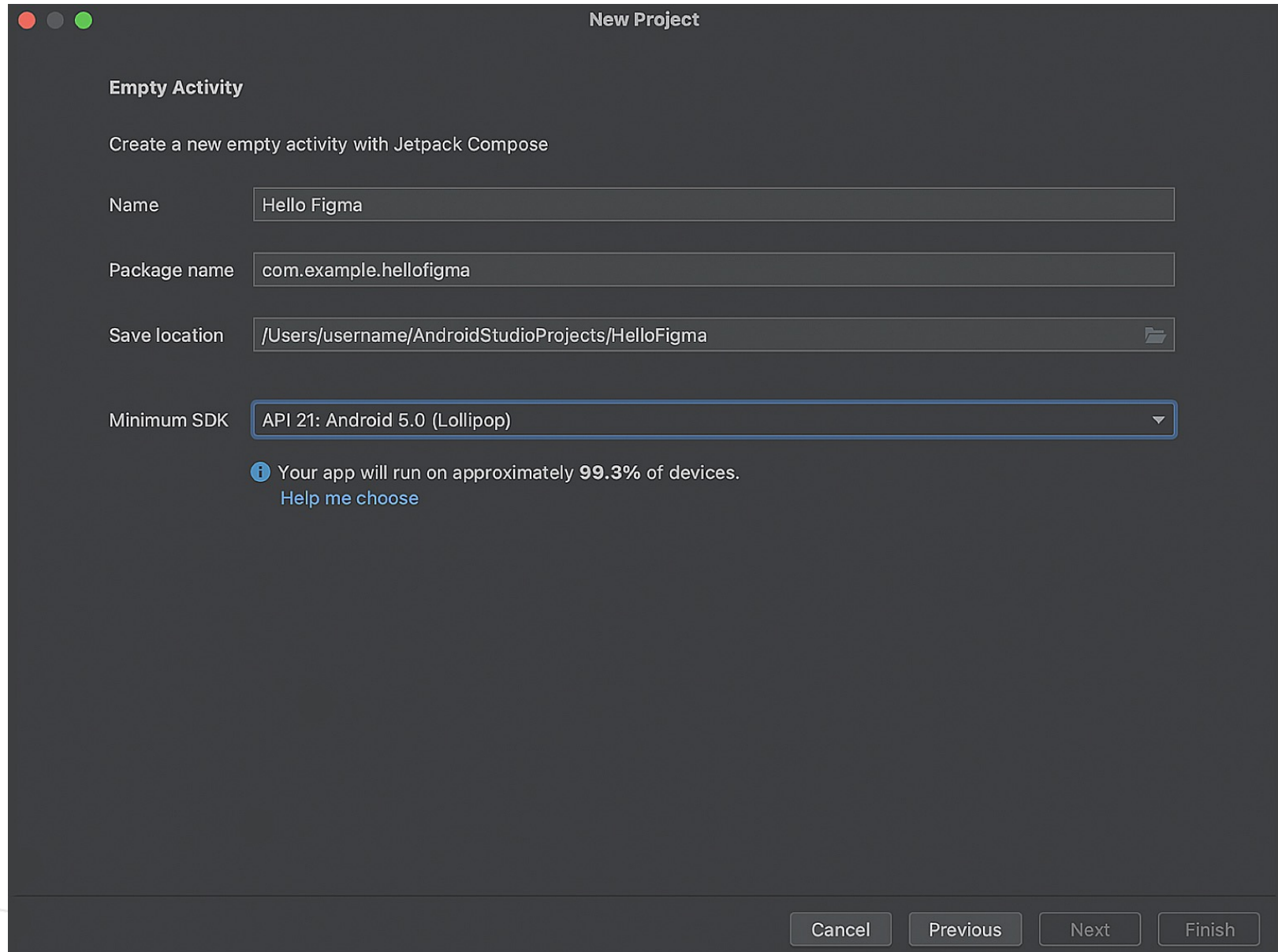
# Настройка проекта Android Studio

## Создание проекта



# Настройка проекта Android Studio

## Создание проекта



New Project

Empty Activity

Create a new empty activity with Jetpack Compose

Name

Package name

Save location

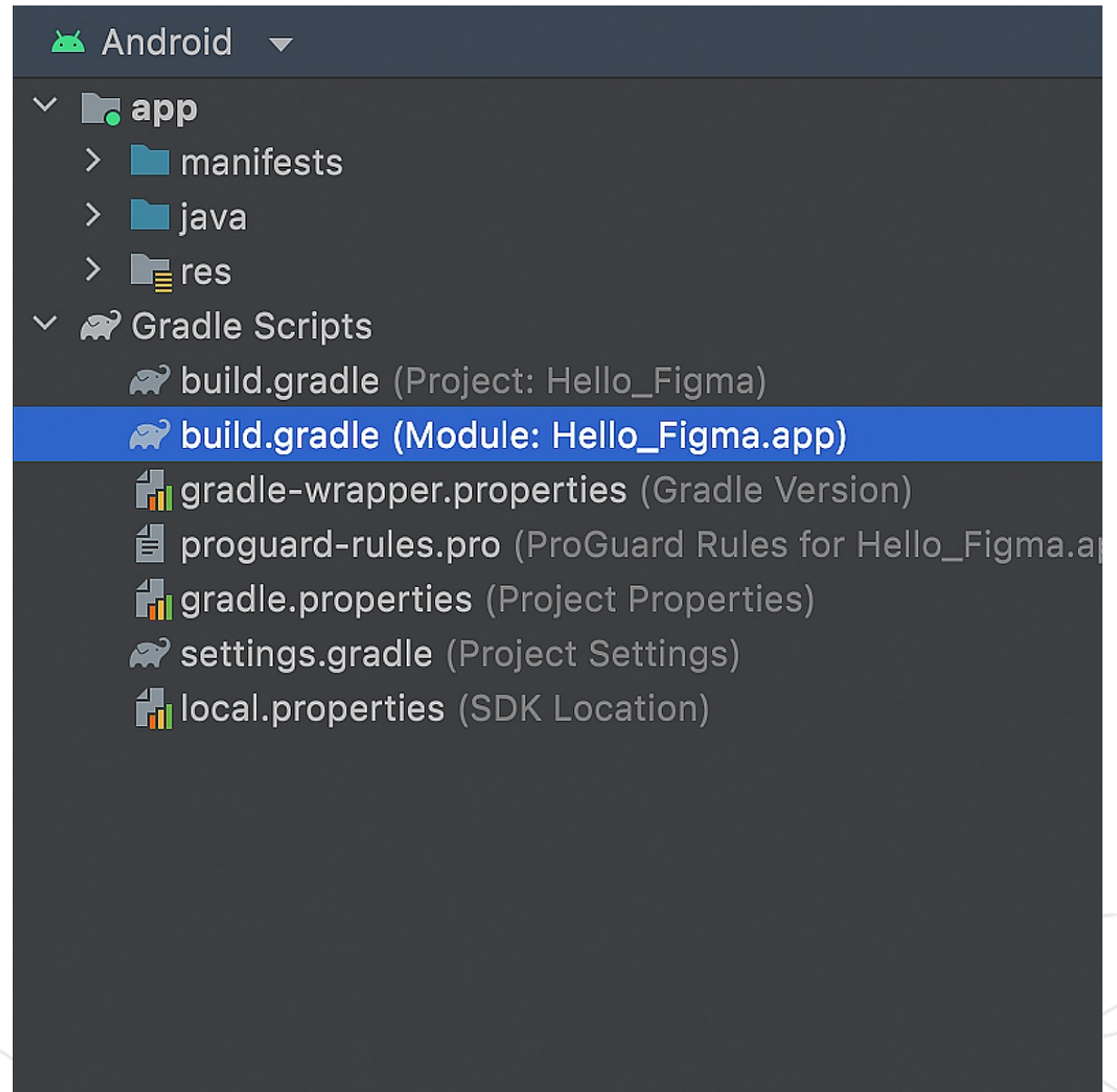
Minimum SDK

**i** Your app will run on approximately **99.3%** of devices.  
[Help me choose](#)

Cancel Previous Next Finish

# Настройка проекта Android Studio

Прописываем плагин



# Настройка проекта Android Studio

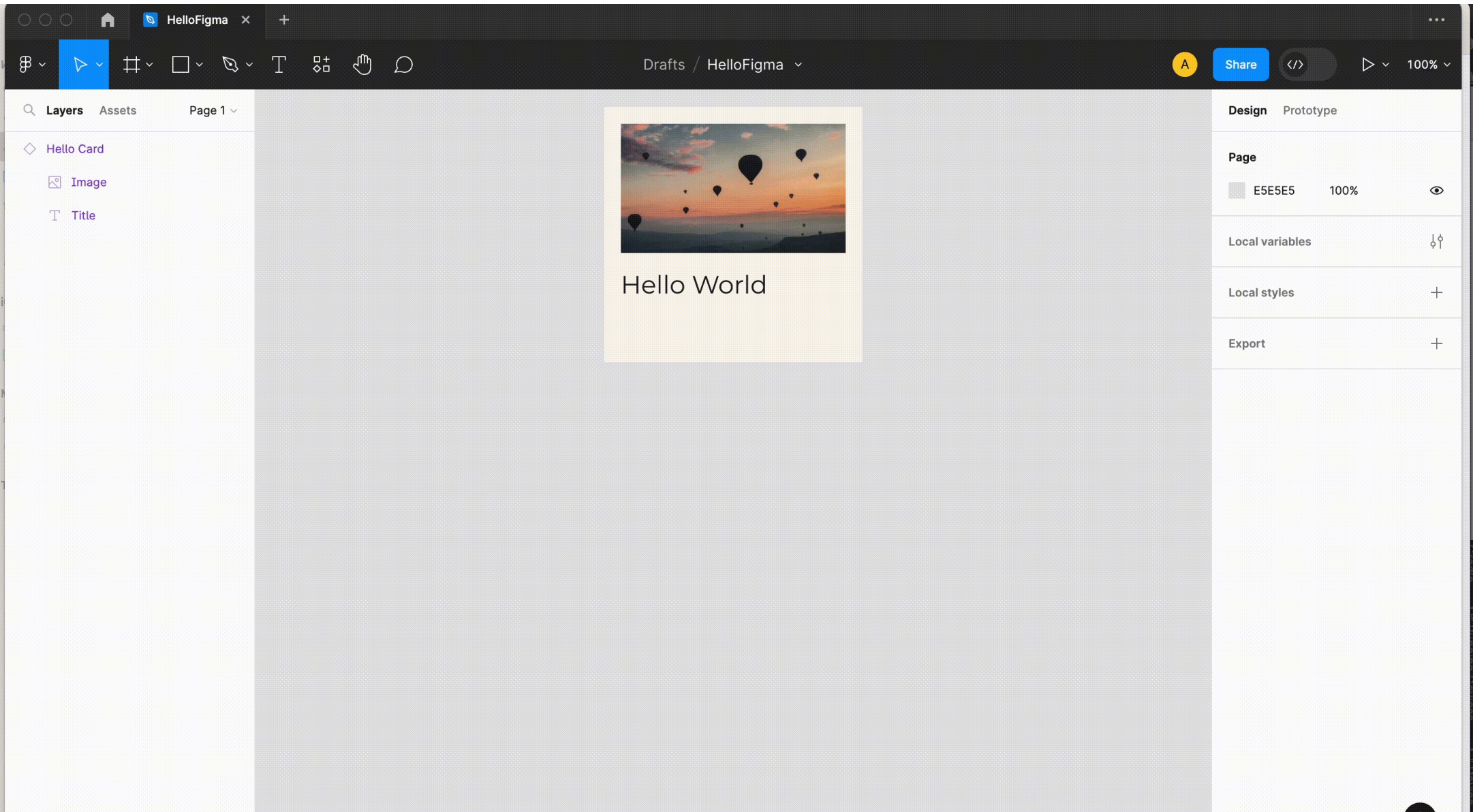
Прописываем плагин. Текущая версия 0.3.11

```
//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





# UI Package

The image shows a Figma workspace with a 'Hello Card' component selected. The component is a card with a sunset background, the text 'Открытие адреса' (Address opening), 'Hello World', and a button labeled 'Отменить' (Cancel). A 'Hug x Hug' label is visible below the card. The 'Relay for Figma' panel is open, showing the 'Share with developer' workflow. The 'Save version history' step is active, with a text input field containing 'E.g. Relay-2023-02-03 icon button added' and a 'Describe what changed' text area. Below this, the 'Copy the component link' step is shown with a link icon and a 'L' icon. The 'Import in Android Studio' step is also visible, with instructions on how to use the 'Relay Android Studio plugin' to import the UI Package into an Android Studio project and generate Jetpack Compose code. The bottom right corner of the Figma interface shows the 'Layer' and 'Fill' properties for the selected element, with 'Pass through' and 'FFFFFF' visible.

Layers Assets Page 1

Hello Card

Hello Card

Image

Title

Address of the film: [input field]

Открытие адреса

Hello World

Отменить

Hug x 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

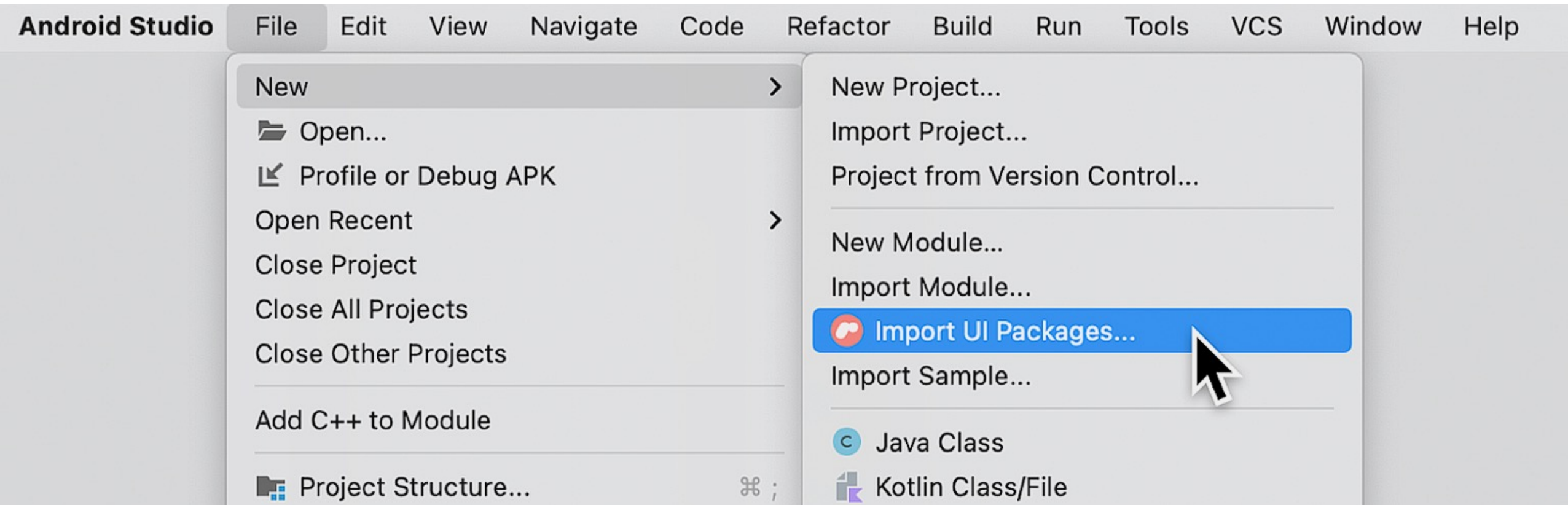
Pass through 100%

Fill

FFFFFF 100%



# UI Package



# UI Package

Import UI Packages



Figma source URL [?](#)

`file/d8JPixki4VNsqHDcs7Q0IB/HelloFigma?type=design&node-id=0-1&mode=design&t=S6wtie6b9ImTOZ2M-0`

App theme:  [?](#)

Translate Figma styles to Compose theme [?](#)

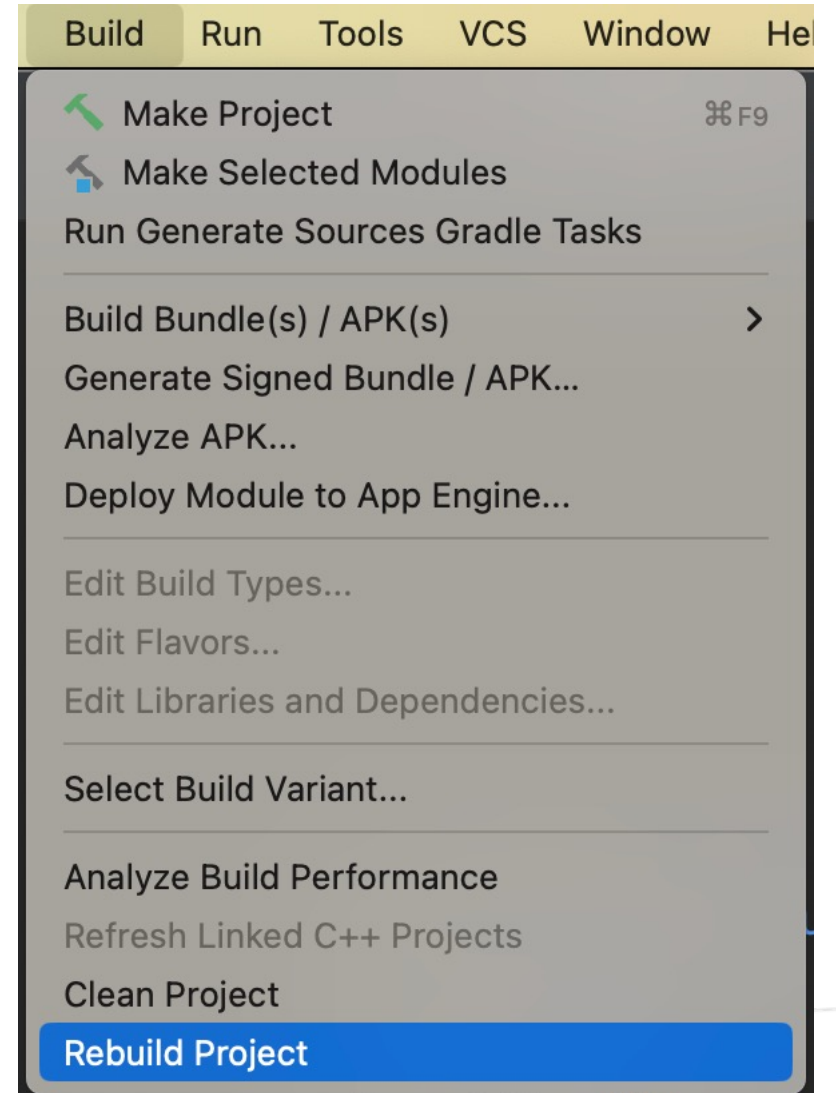
- Material 3 Design Kit configuration
- Material 2 Design Kit configuration
- Import custom configuration:

[?](#) Cancel

Next

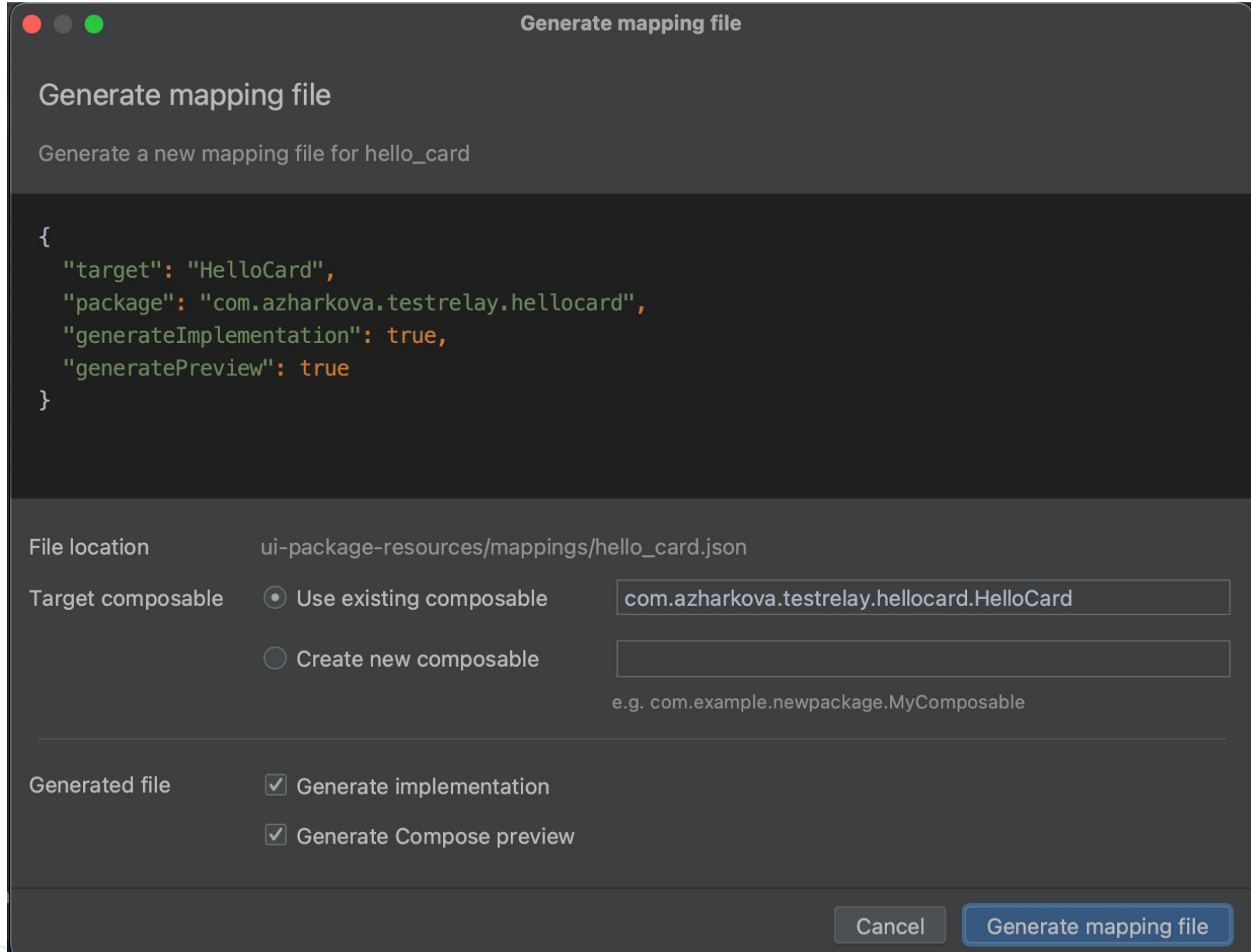
# Генерация

Запускаем ребилд для генерации



# Генерация

Можно замаппить в  
свой Composable



Generate mapping file

Generate a new mapping file for hello\_card

```
{  
  "target": "HelloCard",  
  "package": "com.azharkova.testrelay.hellocard",  
  "generateImplementation": true,  
  "generatePreview": true  
}
```

File location `ui-package-resources/mappings/hello_card.json`

Target composable

Use existing composable `com.azharkova.testrelay.hellocard.HelloCard`

Create new composable

e.g. `com.example.newpackage.MyComposable`

Generated file

Generate implementation

Generate Compose preview

Cancel Generate mapping file



# Генерация

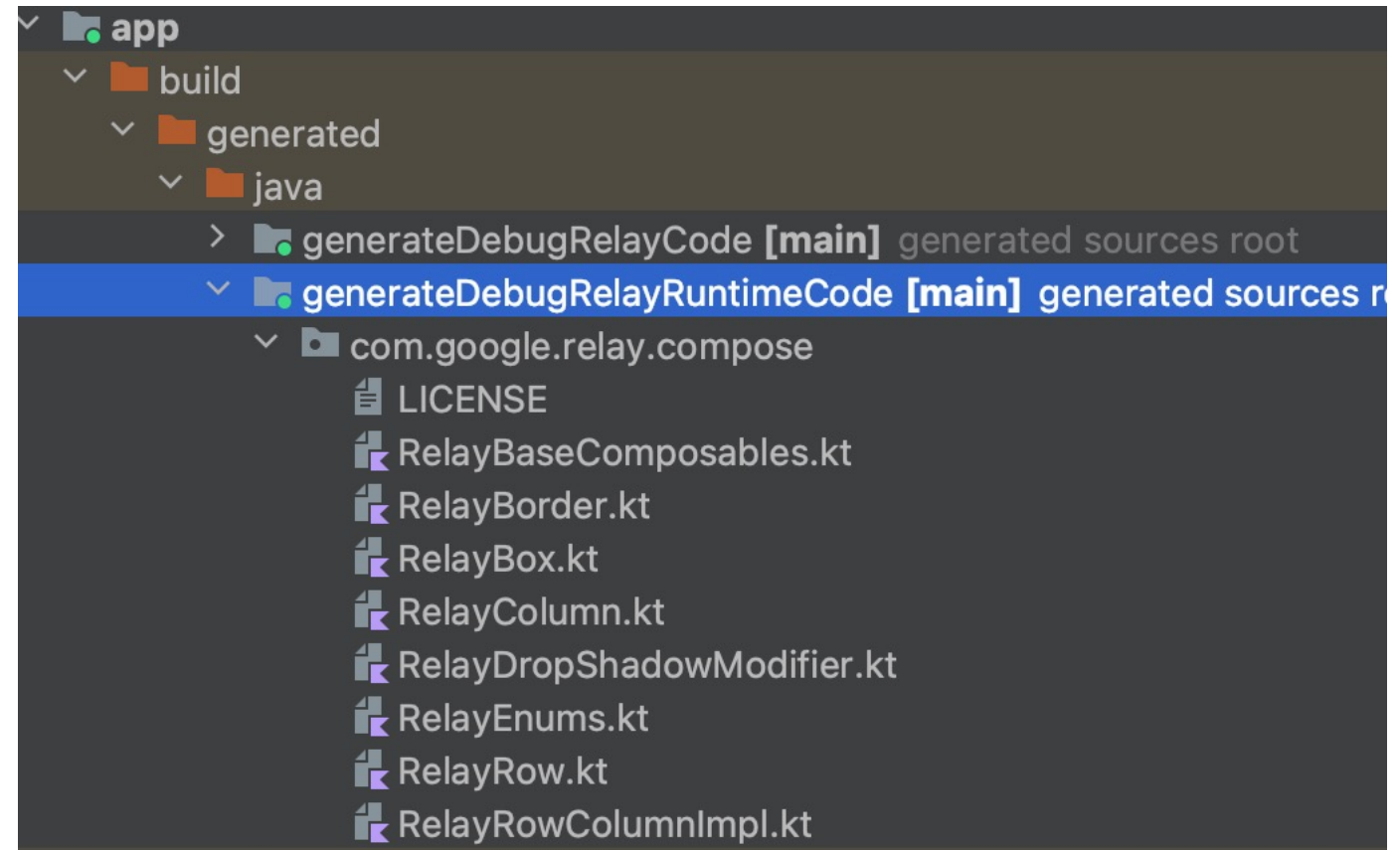
Files under the "build" folder are generated and should not be edited.

```
1 package com.azharkova.testrelay.hellocard
2
3 import ...
4
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) { this: RelayContainerScope
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 ...
4
5
6
7
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



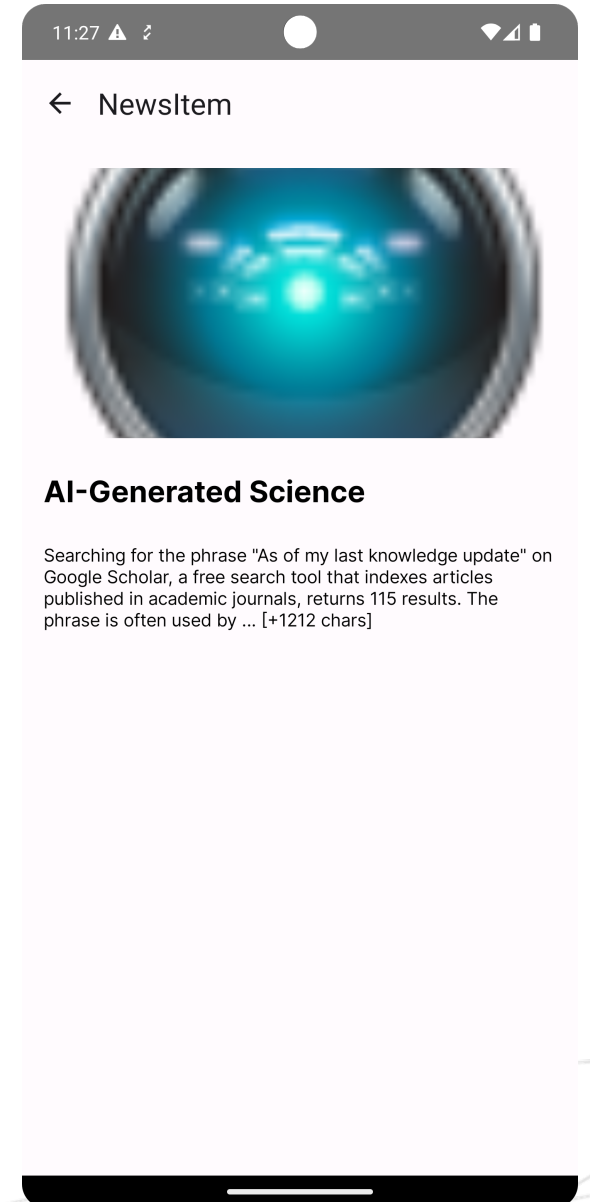
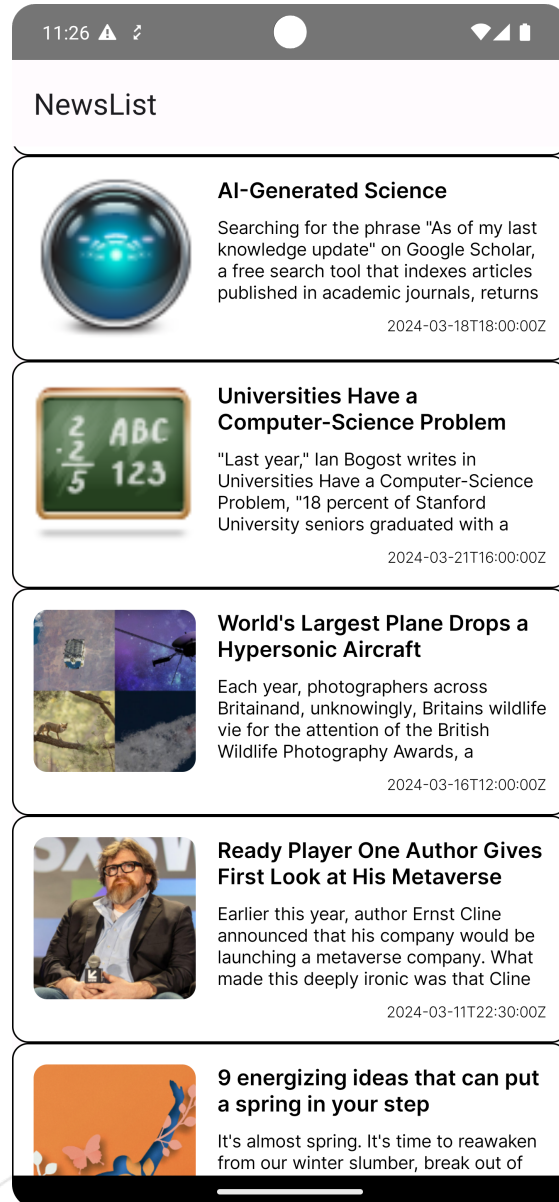
# А теперь реальный псевдо-боевой кейс





# Приложение новостей


<https://github.com/anioutkazhar kova/NewsAppRelay>




# Список новостей

11:26


NewsList




**AI-Generated Science**  
Searching for the phrase "As of my last knowledge update" on Google Scholar, a free search tool that indexes articles published in academic journals, returns  
2024-03-18T18:00:00Z




**Universities Have a Computer-Science Problem**  
"Last year," Ian Bogost writes in Universities Have a Computer-Science Problem, "18 percent of Stanford University seniors graduated with a  
2024-03-21T16:00:00Z



**World's Largest Plane Drops a Hypersonic Aircraft**  
Each year, photographers across Britain and, unknowingly, Britain's wildlife vie for the attention of the British Wildlife Photography Awards, a  
2024-03-16T12:00:00Z




**Ready Player One Author Gives First Look at His Metaverse**  
Earlier this year, author Ernest Cline announced that his company would be launching a metaverse company. What made this deeply ironic was that Cline  
2024-03-11T22:30:00Z




**9 energizing ideas that can put a spring in your step**  
It's almost spring. It's time to reawaken from our winter slumber, break out of



Android Large - 1



**Lorem Ipsum**  
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore...  
11.11.2023



**Lorem Ipsum**  
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore...  
11.11.2023



# Список новостей

Компонуем в компонент

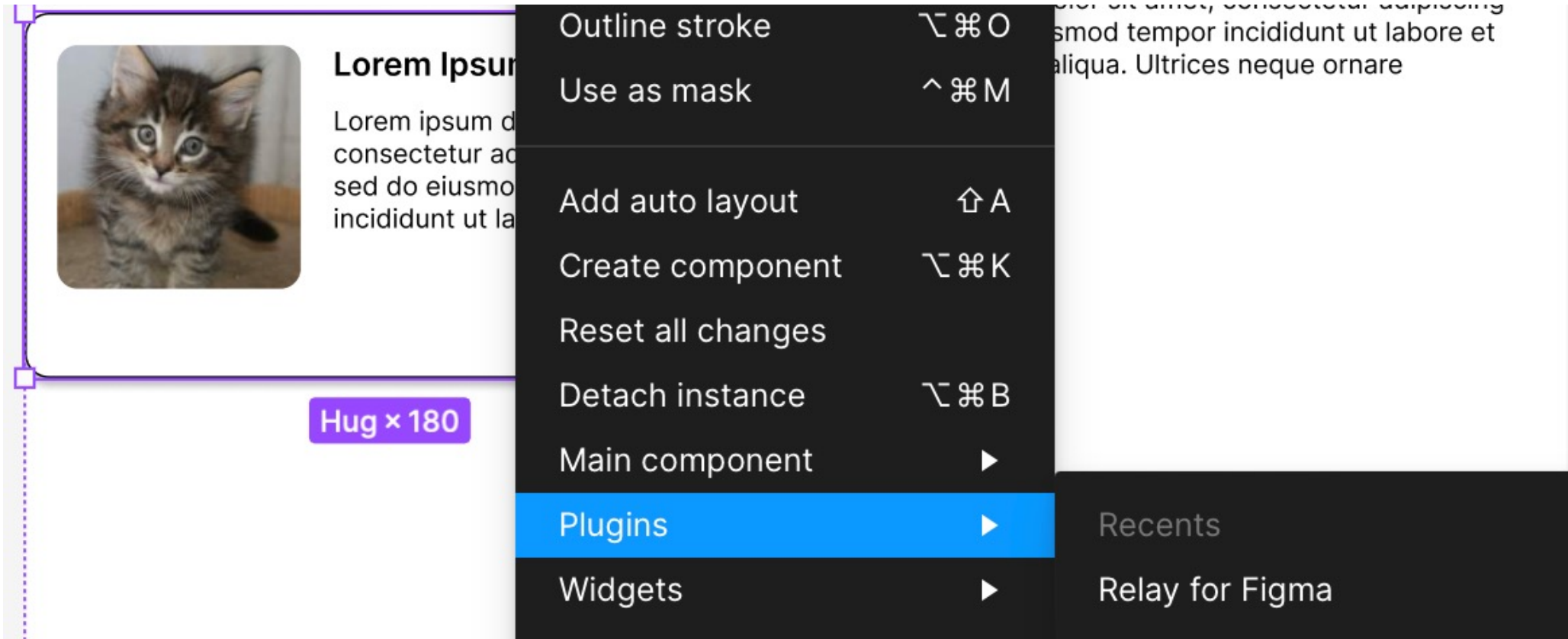
The image shows a component editor interface. On the left, a light blue sidebar lists the components used in the `NewsItemRow` component:

- `NewsItemRow` (root component)
- `Content` (container)
- `Image` (with a red arrow pointing to the kitten image in the preview)
- `TextGroup` (container for text)
- `Title` (text element)
- `Text` (text element)
- `Date` (text element)

On the right, a preview window shows the rendered `NewsItemRow` component. It features a rounded rectangle with a purple border and corner handles. Inside, there is a square image of a kitten, a bold title "Lorem Ipsum", a paragraph of Lorem Ipsum text, and a date "11.11.2023". A purple button at the bottom of the preview is labeled "Hug × 180".

# Настройка компонента

Переходим через контекстное меню



The image shows a design tool interface with a component menu open. The component being edited is a card with a kitten image and placeholder text. The menu options are:

- Outline stroke (⌘ ⌘ O)
- Use as mask (^ ⌘ M)
- Add auto layout (⇧ A)
- Create component (⌘ ⌘ K)
- Reset all changes
- Detach instance (⌘ ⌘ B)
- Main component (▶)
- Plugins (▶)
- Widgets (▶)

The 'Plugins' option is highlighted in blue. A secondary menu is visible to the right of the 'Plugins' option, containing:

- Recents
- Relay for Figma

Below the component, a purple label indicates 'Hug x 180'.

# Настройка компонента

Добавляем свойства

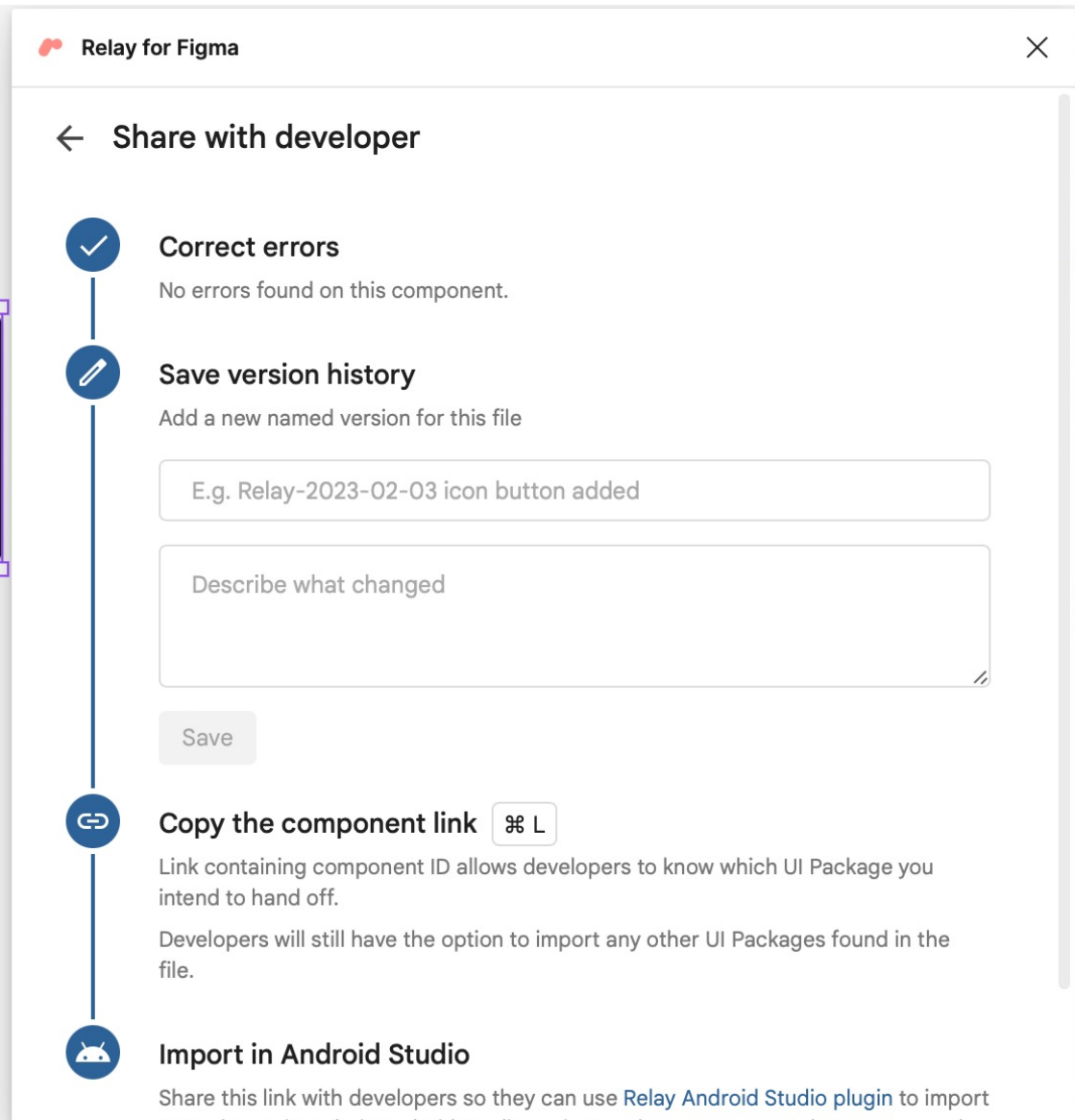
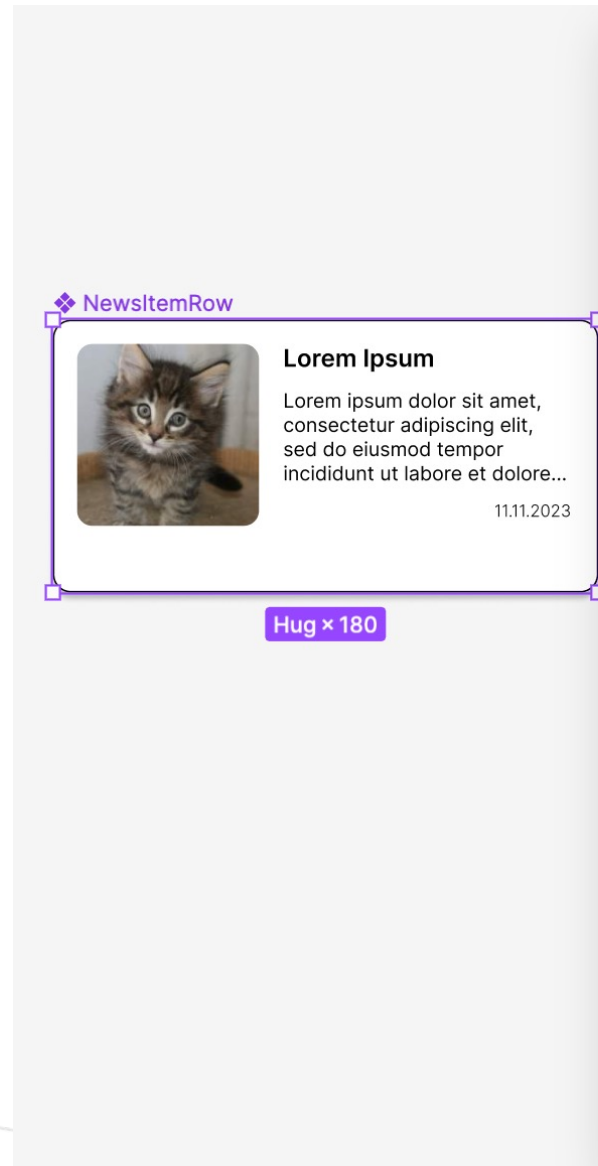
The image shows a Figma component editor for a 'NewsItemRow' component. On the left, a preview of the component is shown with a purple dashed border and a '120 x 120' label. The preview contains a kitten image, the text 'Lorem Ipsum', a truncated text block, and a date '11.11.2023'. On the right, the 'Relay for Figma' interface displays the component's name 'NewsItemRow' and a list of properties. The 'imageUrl' property is selected and expanded, showing its name, property type, description, and layer type.

Property	Value
imageUrl	imageUrl
Name	imageUrl
Property	image-content
Description	Add description
Layer	Image

# Настройка компонента

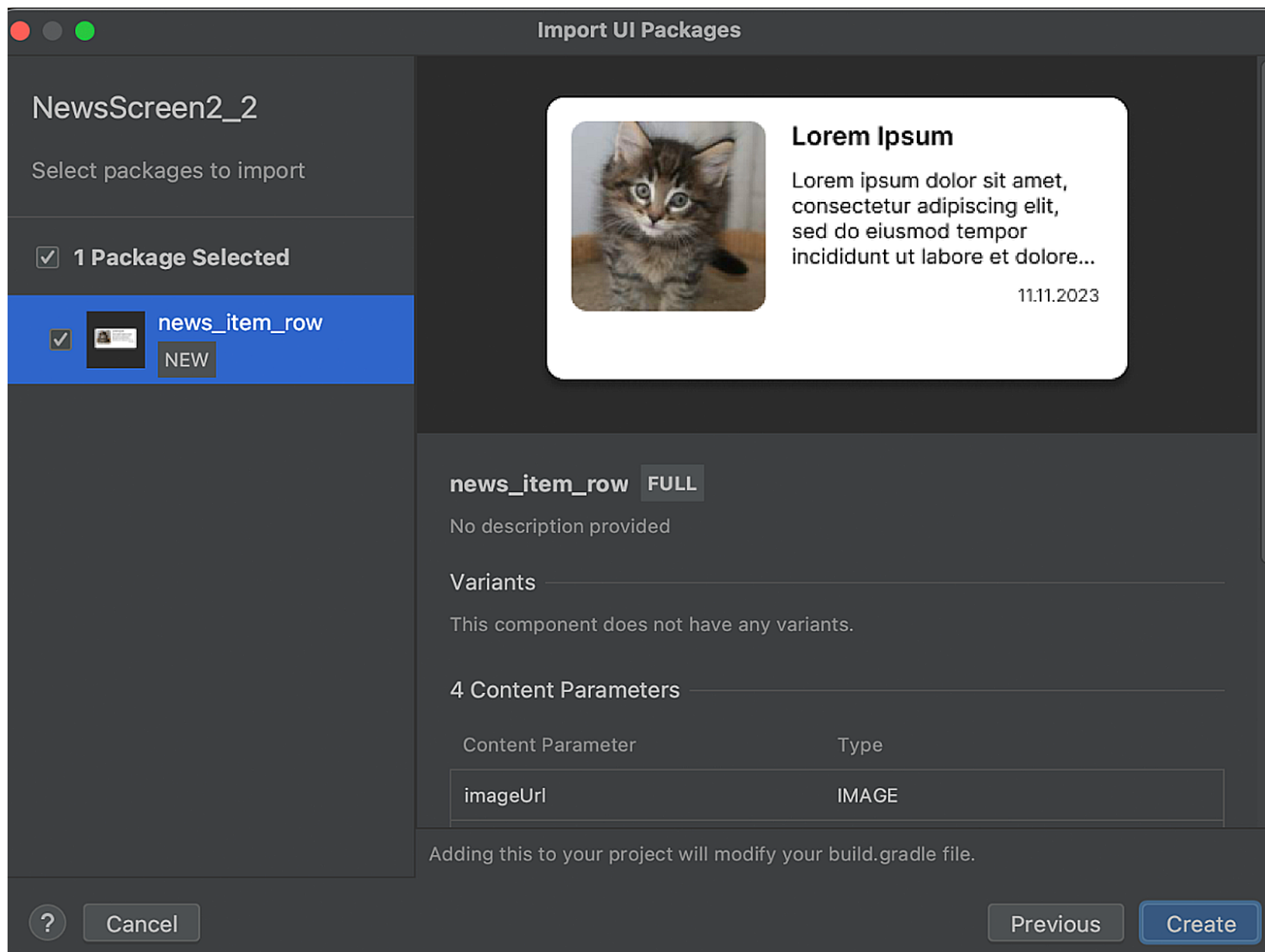
Share with developer

Cmd + L / Ctrl+ L



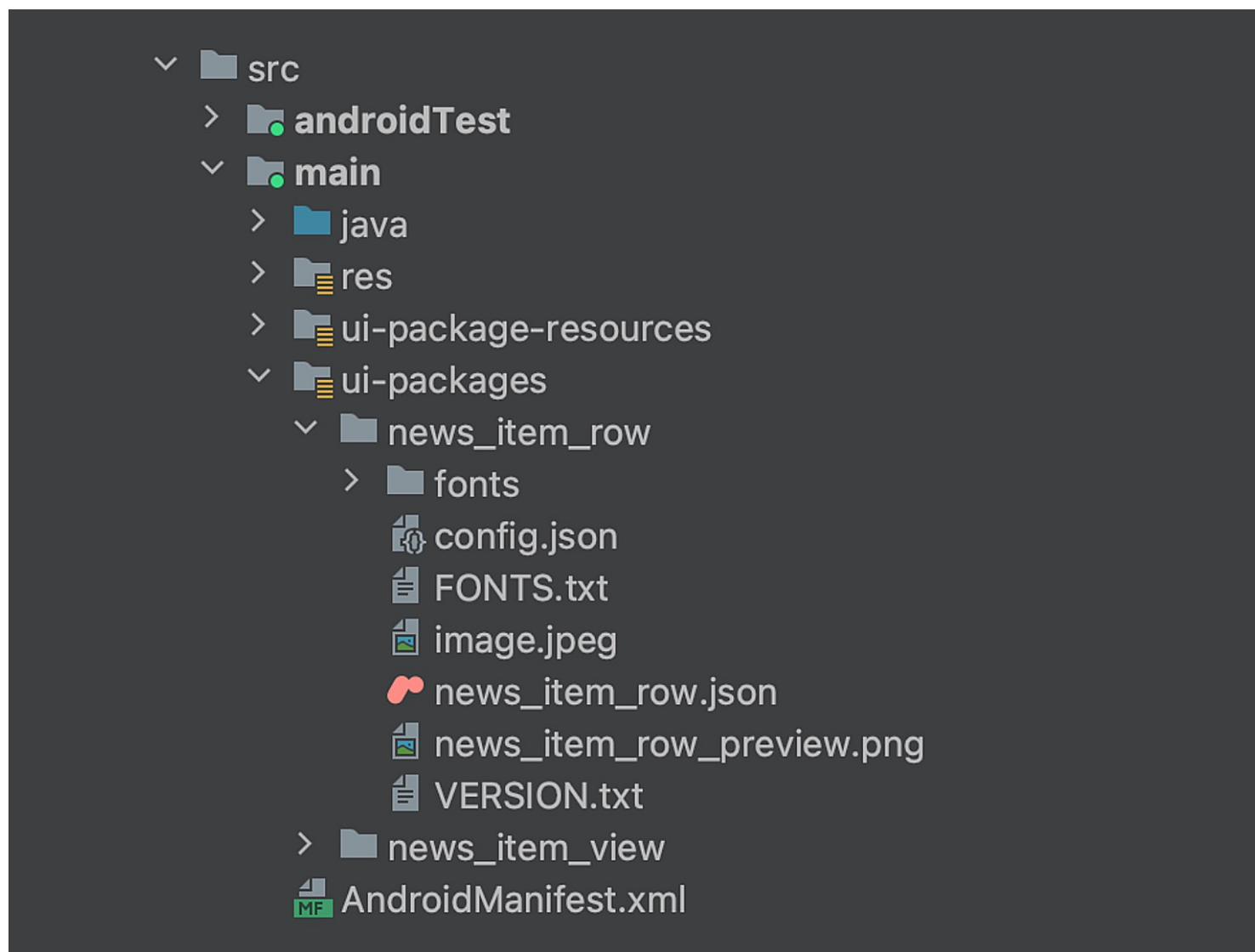
# Импорт компонента

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 x
1 {
2   "name": "news_item_row",
3   "version": 32,
4   "source-key": {
5     "type": "figma",
6     "file": "S6zUsU32ma50ZnhAuyva7qf",
7     "node": "10:2",
8     "version": "4486181108",
9     "component-id": "439b8f16eae69f70511dfd2b1708e
10  },
11  "default": "NewsItemRow",
12  "design": {
13    "atoms": [
14      {
15        "type": "group",
16        "id": "top_level",
17        "root": "true"
18      },
19      {
20        "type": "group",
21        "id": "Content"
22      },
23      {
24        "type": "group",
```

UI Package



**news\_item\_row**  
No description provided

Package Options

Mapping file  ?  
Parents of this package will invoke the generated code

Variants  
This component does not have any variants.

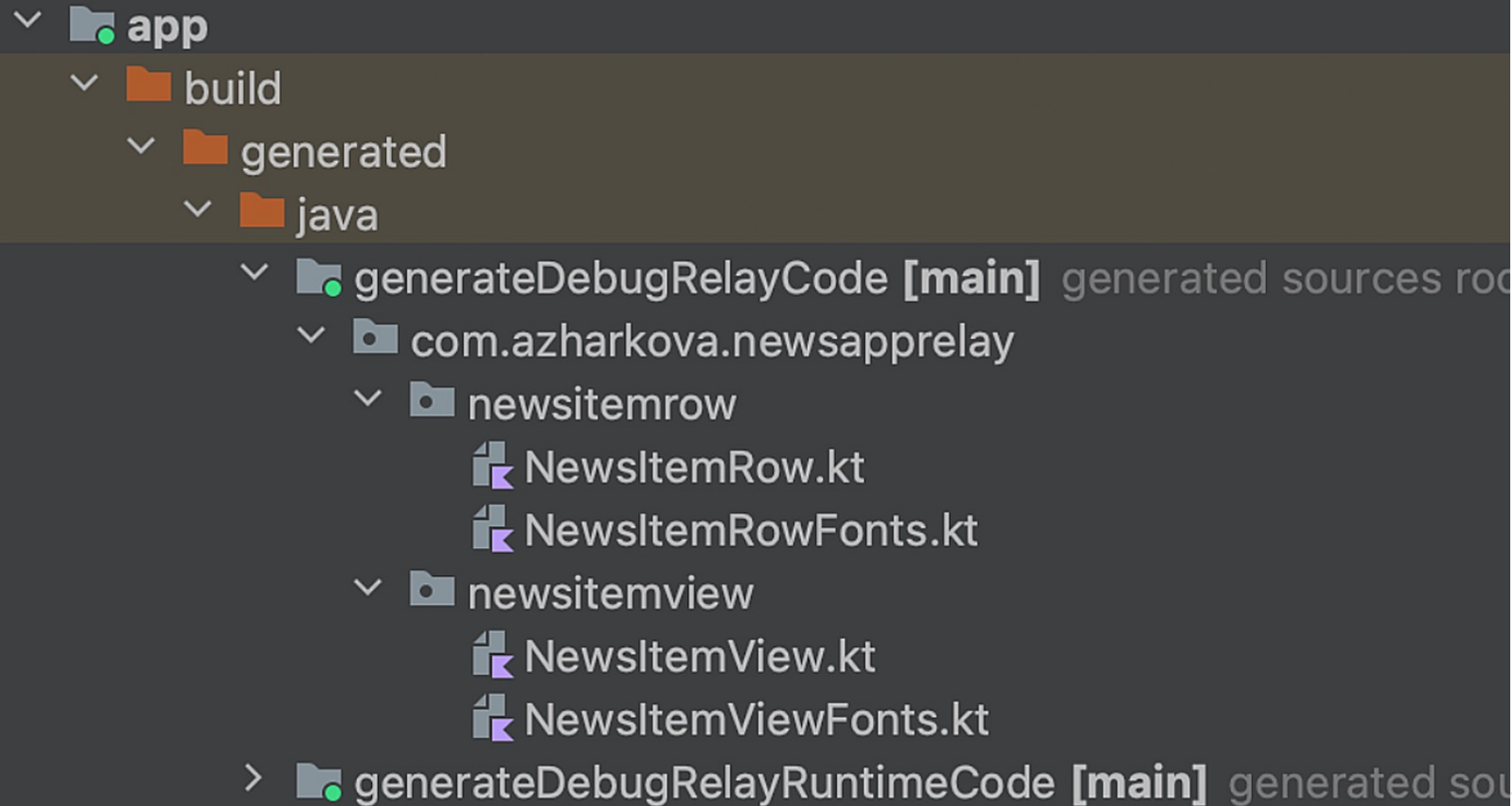
4 Content Parameters

Content Parameter	Type
image	IMAGE
title	TEXT
text	TEXT
date	TEXT



# Сгенерированные файлы

Сгенерированные компоненты доступны в app – build - generated



```

└─ app
  └─ build
    └─ generated
      └─ java
        └─ generateDebugRelayCode [main] generated sources root
          └─ com.azharkova.newsapprelay
            └─ newsitemrow
              ├── Newsletter.kt
              ├── NewsletterFonts.kt
            └─ newsitemview
              ├── NewsletterView.kt
              └── NewsletterViewFonts.kt
        > generateDebugRelayRuntimeCode [main] generated sources root

```



# 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     }
```

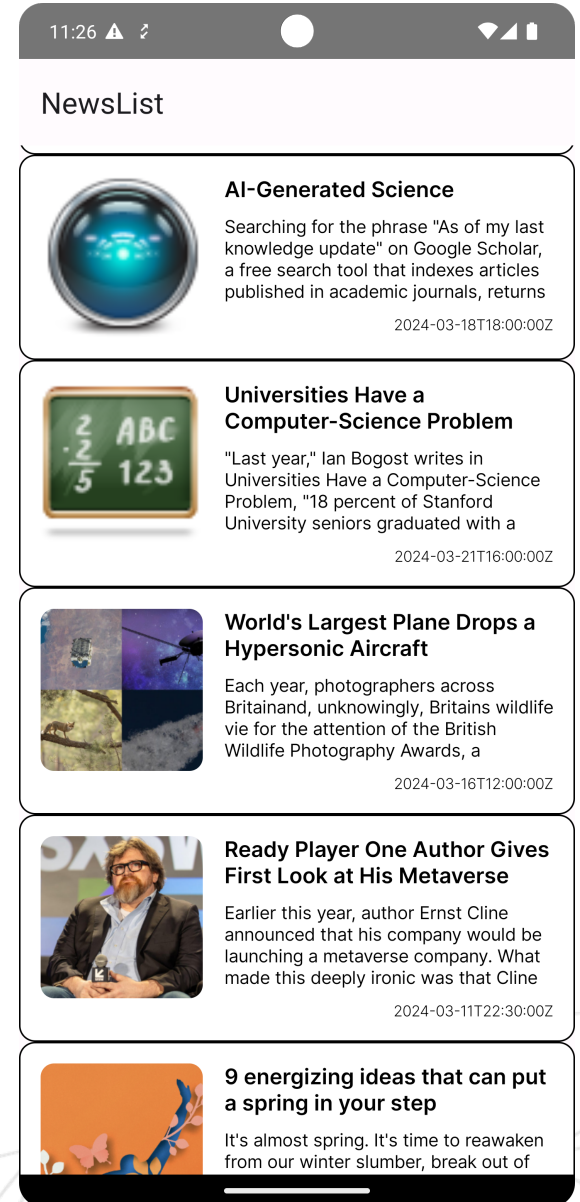
# Подключаем в существующий экран

```
@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>



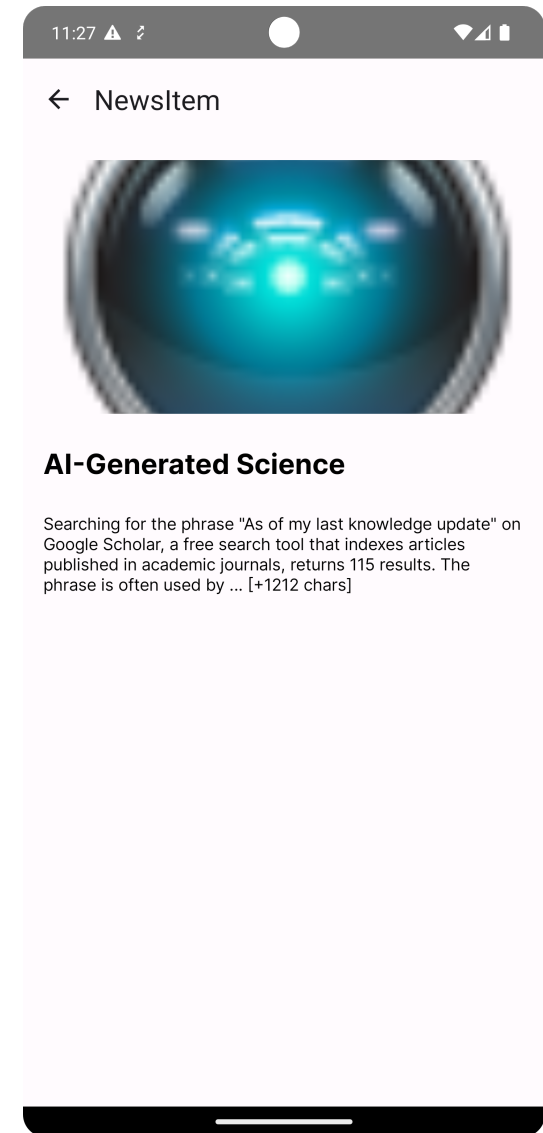


# Добавим экран деталей новости

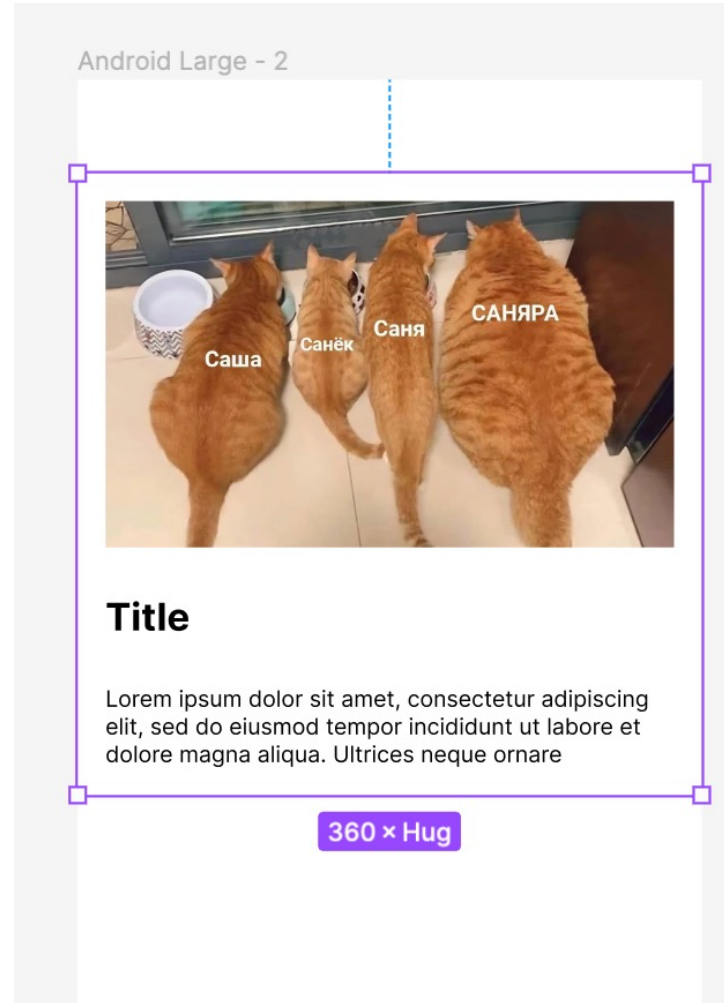
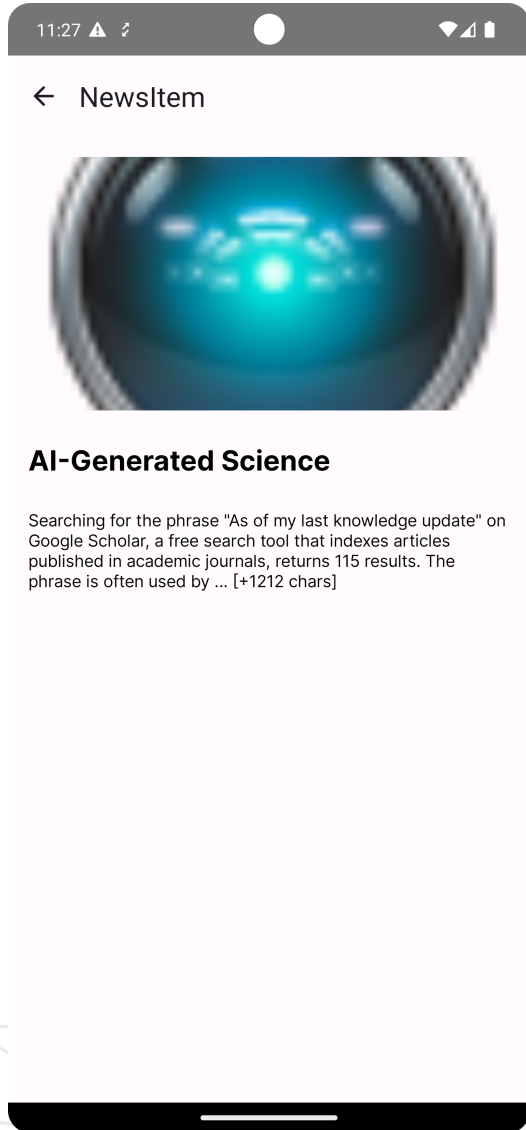


# Экран новости

Добавим экран новости



# Экран новости



# Экран новости

Для портирования дизайна всего экрана превращаем в компонент

Android Large - 2

Layers Assets Page 1 ▾

▾ # Android Large - 2

- ▾ NewsItemView
  - Image
  - Title
  - Text

**Title**

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ultrices neque ornare

360 × Hug

# Экран новости

## Настраиваем параметры и свойства компонента

Relay for Figma



### NewsletterView

Title

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ultrices neque ornare

#### Summary

#### Parameters

image

**title**

text

T title

Name

title

Property

text-content

Description

Add description

Layer

Shim Plugin Iframe

Android Large - 2



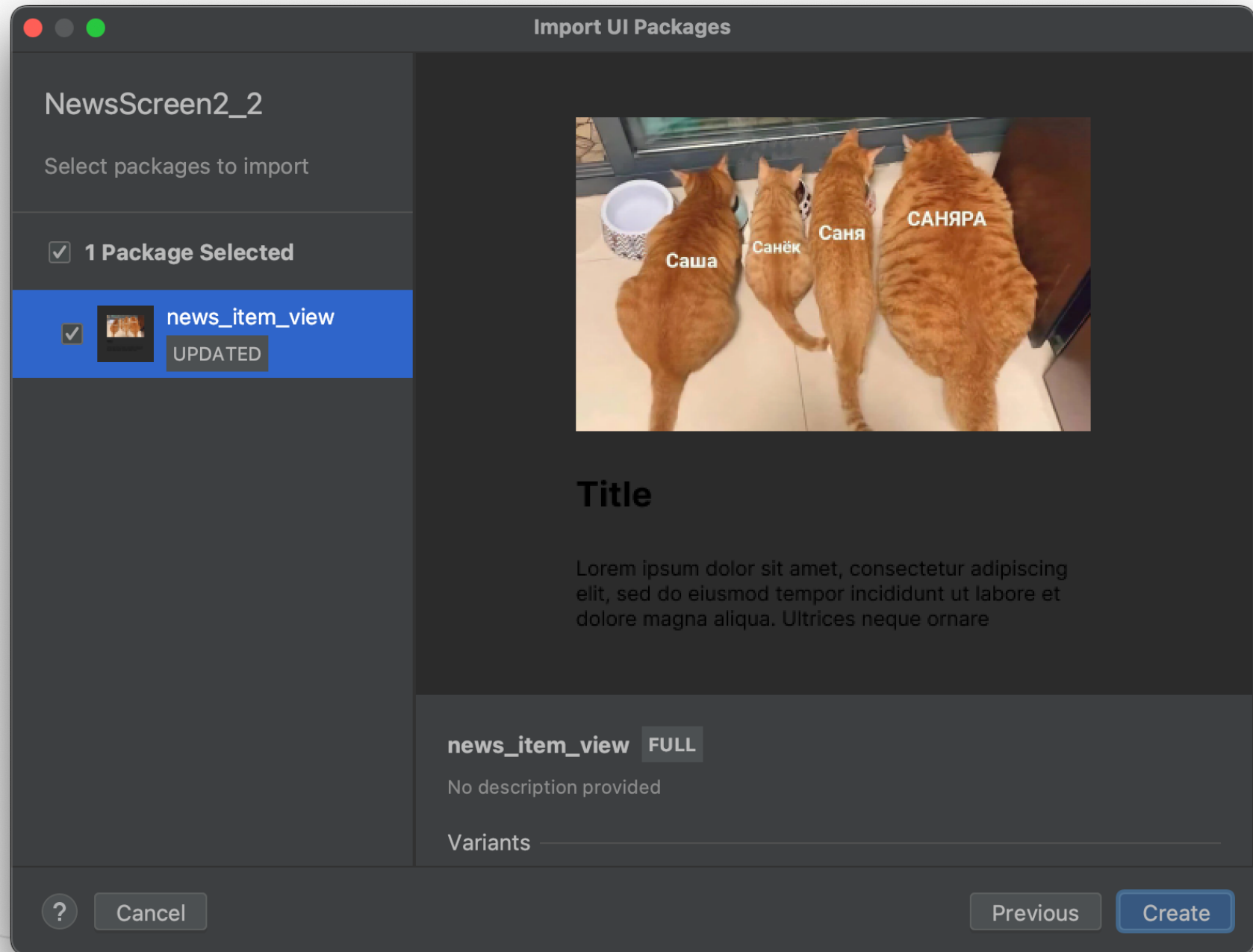
**Title**

Fill x Hug

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ultrices neque ornare

# 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

The image shows a design tool interface for a component named 'NewsletterRow'. The component is a card with a kitten image, a title 'Lorem Ipsum', a paragraph of placeholder text, and a date '11.11.2023'. A purple border indicates it is selected, and a 'Hug x 180' label is visible below it. The right-hand panel displays the component's configuration, including a 'Summary' section with a text input field, a 'Parameters' section with a plus sign, and an 'Interactions' section where 'tap-handler' is selected. Below the interactions are 'Additional Parameters' with a 'string' type. At the bottom right, there is a 'Share with developer' button.

Relay for Figma

NewsletterRow

Summary

Parameters +

Properties

- children
- background-color
- padding
- border-radius

Interactions

- tap-handler
- doubletap-handler
- longpress-handler

Additional Parameters

- string

Enter a description of your component

Share with developer

# Интерактивность

Добавляем tap-handler  
на главный элемент

The image shows a design tool interface for Relay for Figma. On the left, a design canvas displays a 'NewsItemRow' component, which is a card containing a kitten image, a title 'Lorem Ipsum', a paragraph of placeholder text, and a date '11.11.2023'. A purple selection box highlights the entire card, and a purple label 'Hug x 180' is positioned below it. On the right, a properties panel for the 'NewsItemRow' component is open. It features a 'Summary' section, a 'Parameters' section with a plus sign, and a 'Properties' section. The 'Parameters' section lists 'on NewsItemRow tapped' as the selected parameter. The 'Properties' section shows the 'Name' as 'onRowTapped', the 'Property' as 'tap-handler', and the 'Layer' as 'NewsItemRow'. At the bottom of the interface, it displays 'No errors' and a 'Share with developer' button.

Relay for Figma

NewsItemRow

NewsItemRow

Summary

Parameters +

- on NewsItemRow tapped
- imageUrl
- title
- text
- date

on NewsItemRow tapped

Name onRowTapped

Property tap-handler

Description Add description

Layer NewsItemRow

No errors

Share with developer

# 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 ...
4
52
53 /**
54  * This composable was generated from the UI Package 'news_item_row'.
55  * Generated code; do not edit directly
56  */
57 @Composable
58 fun NewsItemRow(
59     modifier: Modifier = Modifier,
60     image: Painter = EmptyPainter(),
61     title: String = "",
62     text: String = "",
63     date: String = "",
64     onRowTapped: () -> Unit = {}
65 ) {
66     TopLevel(
67         onRowTapped = onRowTapped,
68         modifier = modifier
69     ) {...}
70 }
```

# 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)}
            )
        }
    }
}
```

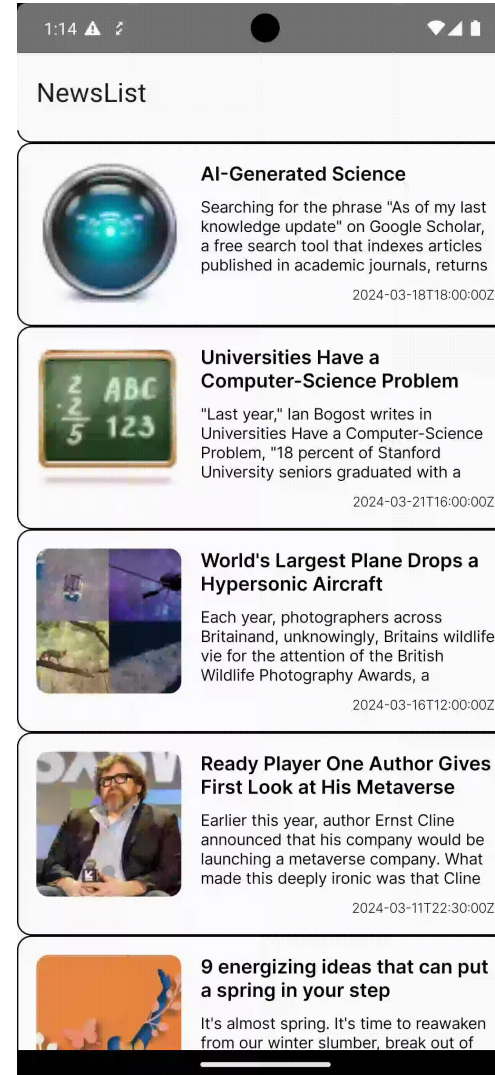
# Настраиваем навигацию

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"))  
        )  
    }  
}
```

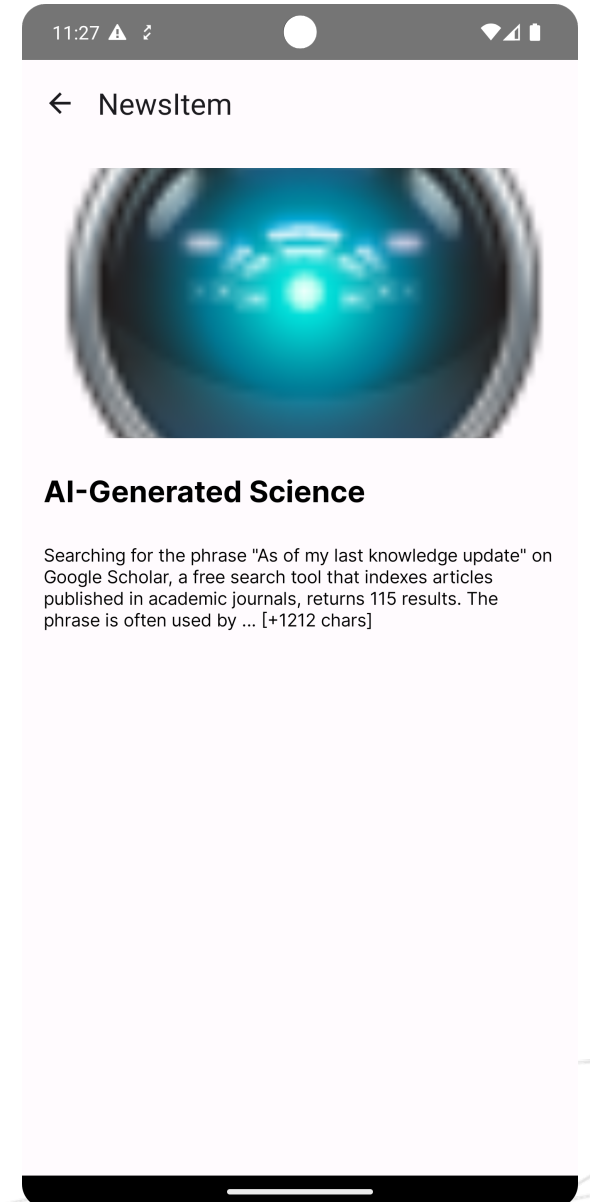
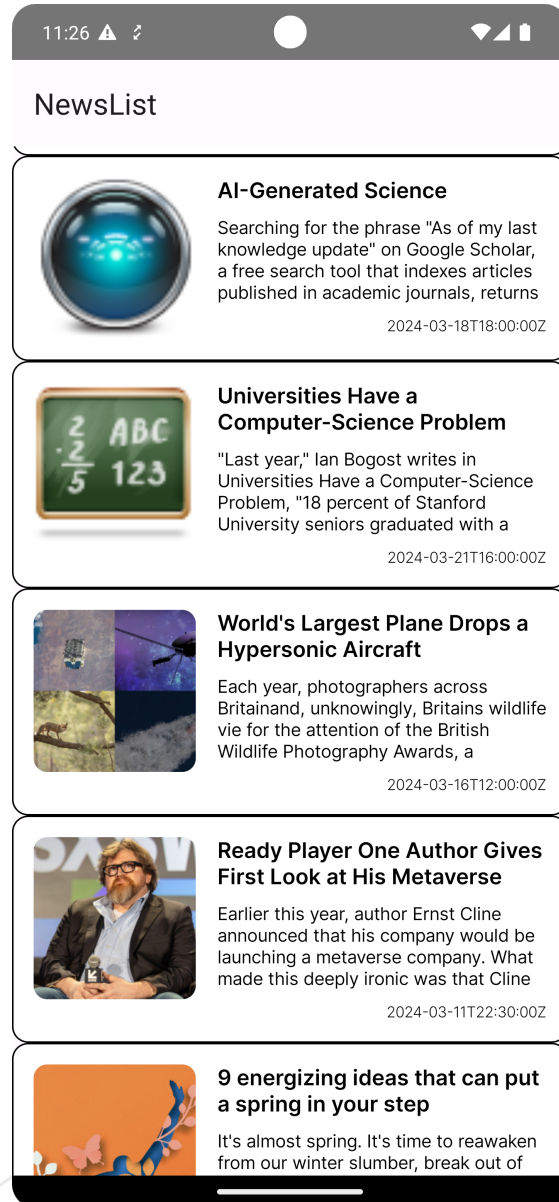
# Проверяем работу

<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=vBNmeiHIDHE>



# Спасибо за внимание!



@anioutkajarkova



azharkova  
prettygeeknotes