avito.tech

Google Developer Groups
Omsk, Russia

devfest2021

Jetpack Compose:
тернистый путь от виджета до полноценного приложения

Андрей Берюхов

Обо Мне

Android Engineer

Статьи и доклады про:

- Jetpack (Compose)
- Мультиплатформу
- Многомодульность

Open-source проекты:

- Coffeegram (Compose Android & Desktop) - 132+ ★

Ментор и спикер Android Academy

Largest Android Active Community

> 10,000 members

Android Academy Fundamentals

Android Academy Advanced (now)

avito.tech

О чем пойдет речь?

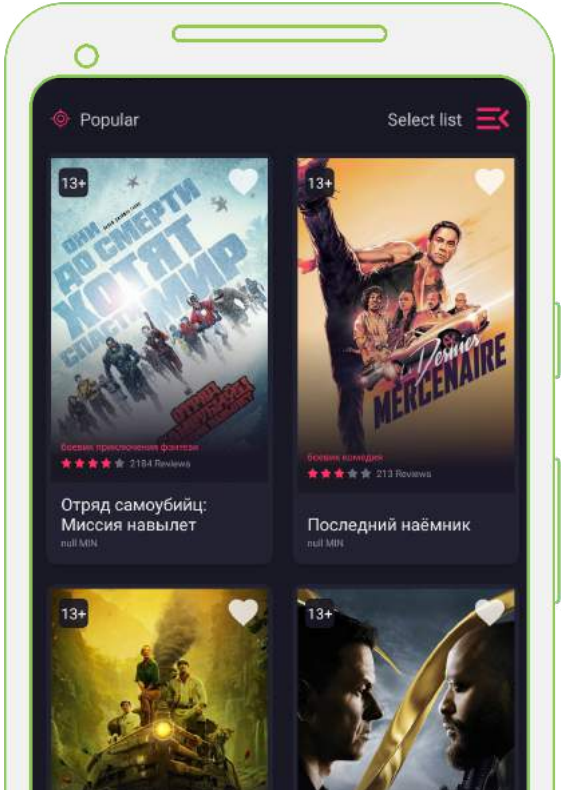
- https://github.com/phansier/AFProject

View-app

- Kotlin
- Coroutines + Flow
- Single Activity - Fragments
- Dagger 2
- MVVM (ViewModel + LiveData)
- Retrofit2 (TMDB API), Room
- Navigation Component
- Paging
- Glide

Compose-app

- Kotlin
- Coroutines + Flow
- Single Activity - No Fragments
- Dagger 2
- MVVM (ViewModel + LiveData)
- Retrofit2 (TMDB API), Room
- Custom navigation
- Можно NC-Compose
- Paging-compose
- Coil-compose



План

01. Переверстаем View в Compose

02. Рассмотрим схемы миграции

03. Мигрируем View-слой

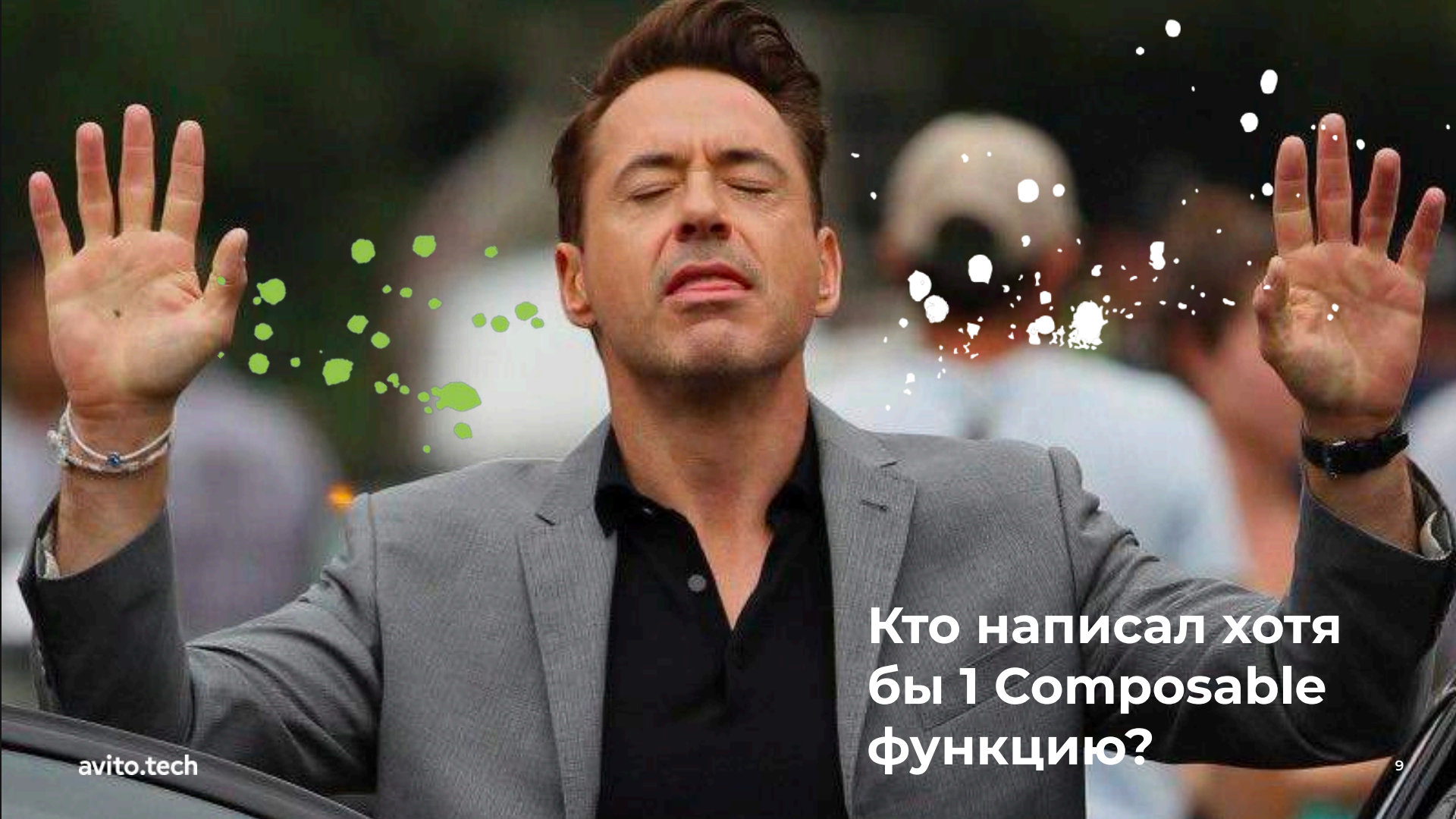
04. Другие возможности Compose

1. Переверстываем View

Но сначала

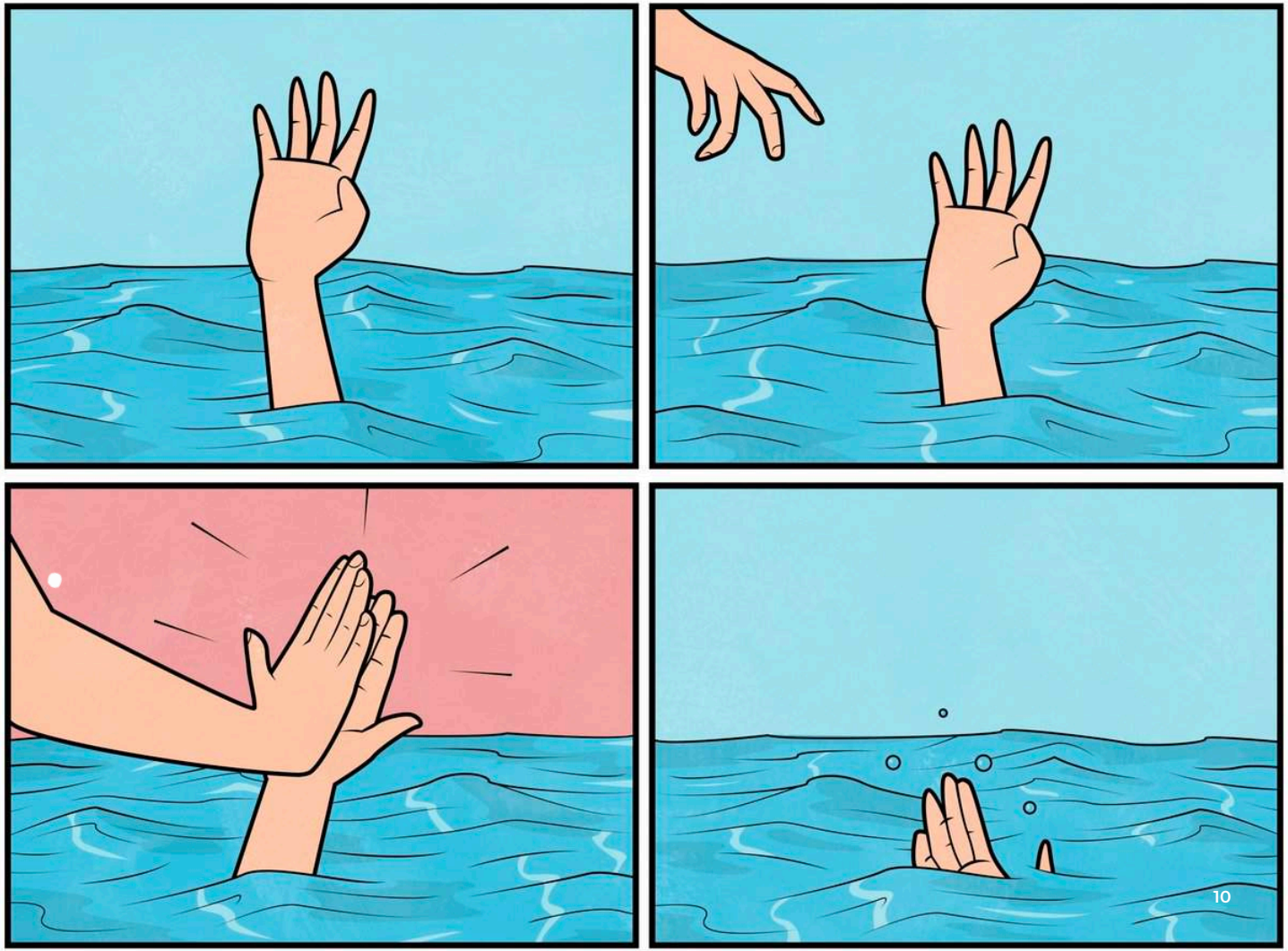
Интерактив

Кто читал статьи / смотрел доклады про Compose

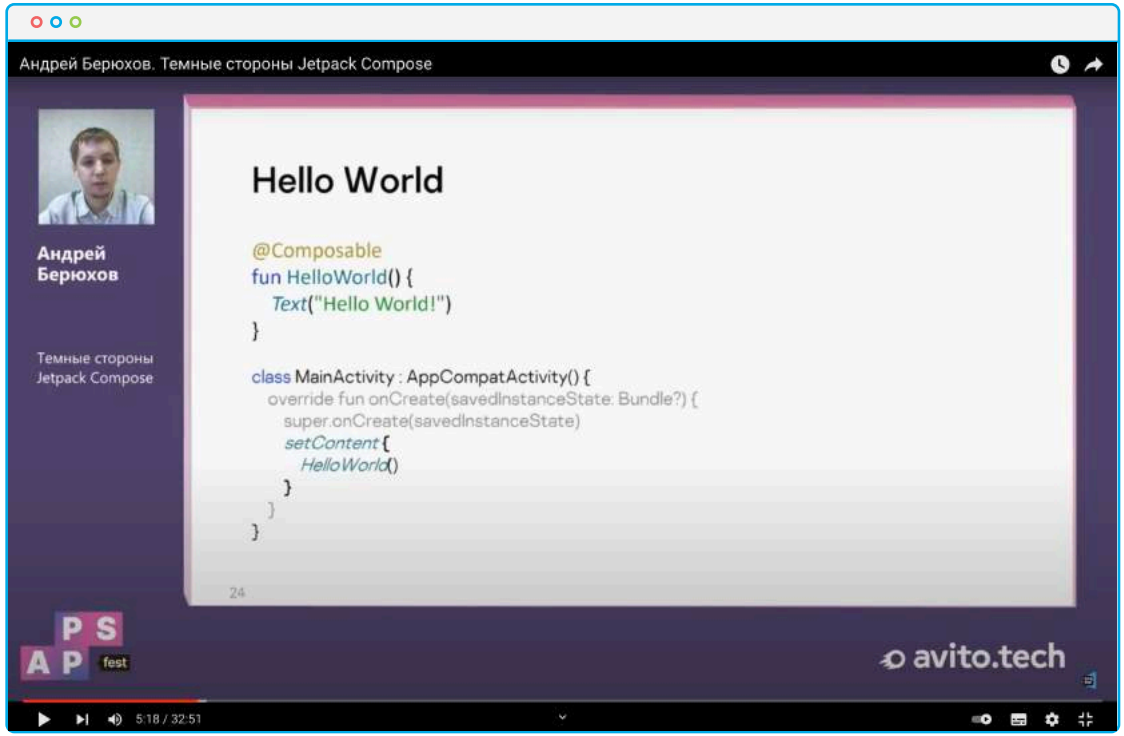


Кто написал хотя бы 1 Composable функцию?

У кого есть приложение с Compose?



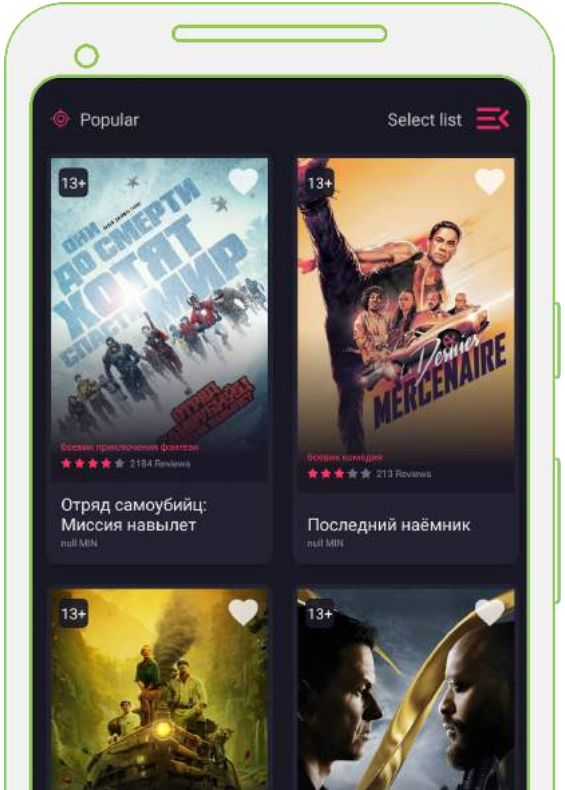
В предыдущих сериях



https://youtu.be/CuCV-SGUuCQ

Но там не было про
ConstraintLayout и LazyGrid

Верстаем Grid



Верстаем Item

Верстаем Card

View

```
<androidx.cardview.widget.CardView
        app:cardBackgroundColor="@color/background_card"
        app:cardCornerRadius="8dp"
        android:layout_margin="8dp"
/>
```

Верстаем Card

Compose

```
@Composable
fun FilmItem() {
    Card(
        onClick = {/*TODO*/ },
        backgroundColor = colorResource(R.color.background_card),
        shape = RoundedCornerShape(8.dp),
        modifier = Modifier.padding(8.dp)
    ) {}
}
```

Верстаем ConstraintLayout

View

```
<androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:padding="4dp">

    <ImageView android:id="@+id/ivBackgroundPoster" />

    <ImageView android:id="@+id/ivMask" />

    <ImageView android:id="@+id/imageView2" />

    <TextView android:id="@+id/tvAge" />

    <TextView android:id="@+id/tvTitle" />

    <TextView android:id="@+id/tvTag" />

    <TextView android:id="@+id/tvReviewsCount" />

    <ImageView android:id="@+id/liked_item" />

    <include android:id="@+id/rating" />

    <TextView android:id="@+id/tvReviews" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Верстаем ConstraintLayout

Compose

```
ConstraintLayout(Modifier.padding(4.dp)){
    val (ivBackgroundPoster, ivMask, ageBg,
        tvAge, tvTitle, tvTag,
        tvReviewsCount, likedItem, ratingBar,
        tvReviews) = createRefs()
}
```

Верстаем ImageView

View

```
<ImageView
        android:id="@+id/ivBackgroundPoster"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:scaleType="centerCrop"
        app:layout_constraintBottom_toBottomOf="@+id/ivMask"
        app:layout_constraintEnd_toEndOf="@+id/ivMask"
        app:layout_constraintStart_toStartOf="@+id/ivMask"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@drawable/poster"/>
```

Верстаем Card

Compose

```
Image(painter = painterResource(id = film.photo),
    contentDescription = "",
    modifier = Modifier.constrainAs(ivBackgroundPoster){
        top.linkTo(parent.top, 8.dp)
        start.linkTo(ivMask.start)
        end.linkTo(ivMask.end)
        bottom.linkTo(ivMask.bottom)
    }
)
```

Верстаем Card

Compose

```
Image(painter = painterResource(id = film.photo),
    contentDescription = "",
    modifier = Modifier.constrainAs(ivBackgroundPoster){
        top.linkTo(parent.top, 8.dp)
        start.linkTo(ivMask.start)
        end.linkTo(ivMask.end)
        bottom.linkTo(ivMask.bottom)
    }
)
```

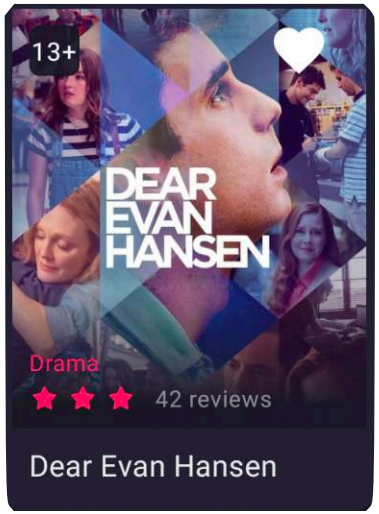
Важно
не забывать про копипасту
в ConstraintLayout

Дублирование id в `.constrainAs()`
никак не подсвечивается
и ломает вёрстку

“Databinding”

Compose

```
@Composable
fun FilmItem(film: Film, isLiked: Boolean = false) {
    Image(
        painter = painterResource(
            id = if (isLiked) R.drawable.ic_heart_liked
            else R.drawable.ic_heart
        ),
    )
}
```



Preview

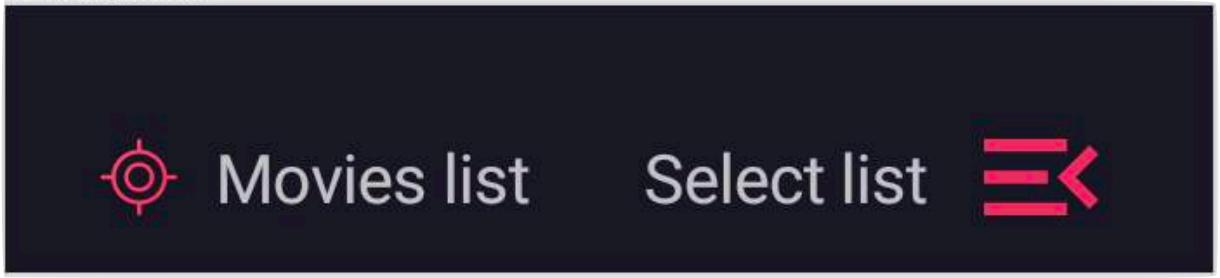
помогает проверять все констрейнты в ConstraintLayout

```
@Preview(device = Devices.PIXEL_4)
@Preview(device = Devices.PIXEL_C)
@Composable
fun FilmItemPreview()
```

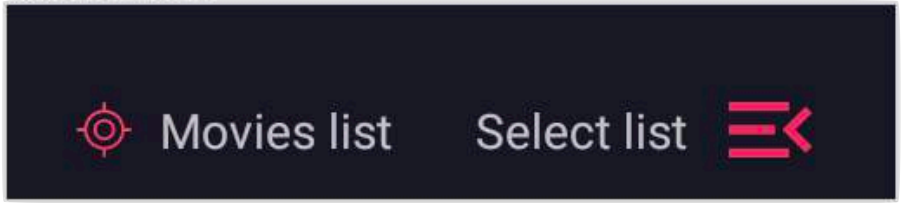
Preview

Expected

MoviesPagePreview



MoviesPagePreview



Preview

Actual

MoviesPagePreview

MoviesPagePreview

RecyclerView

~~RecyclerView~~ LazyColumn

~~RecyclerView~~ LazyColumn

```
LazyColumn( ... ) {
    itemsIndexed(items = films,
        itemContent = { _, item → FilmItem(film = item) }
    )
}
```

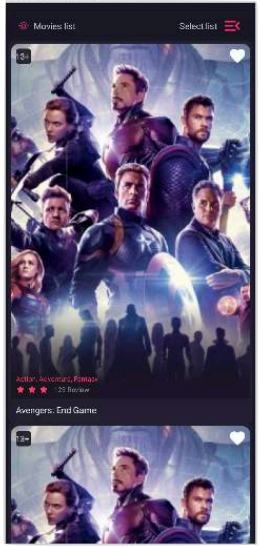
~~RecyclerView~~ LazyColumn

```
LazyColumn( ... ) {
    itemsIndexed(items = films,
        itemContent = { _, item → FilmItem(film = item) }
    )
}
```

LazyColumn

LazyColumn

MoviesPagePreview



MoviesPagePreview

LazyColumn + Row = LazyVerticalGrid

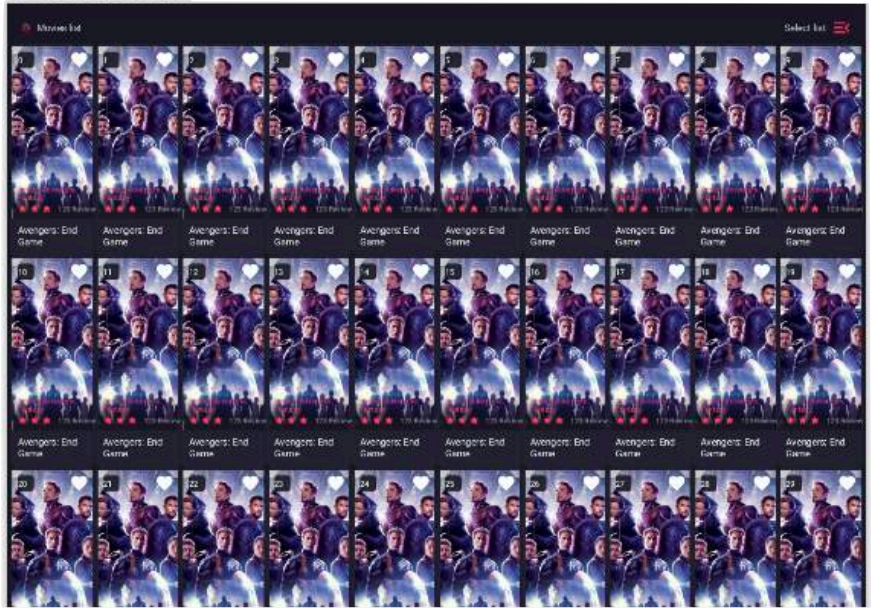
```
LazyVerticalGrid(
    modifier = modifier,
    cells = GridCells.Adaptive(minSize = 128.dp)
) {
    itemsIndexed(items = films,
        itemContent = { index, item →
            FilmItem(film = item) }
    )
}
```

GridCells.Adaptive

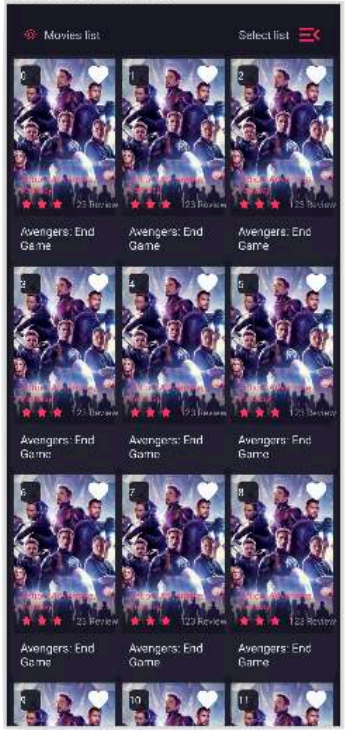
```
LazyVerticalGrid(
    modifier = modifier,
    cells = GridCells.Adaptive(minSize = 128.dp)
) {
    itemsIndexed(items = films,
        itemContent = { index, item →
            FilmItem(film = item) }
    )
}
```

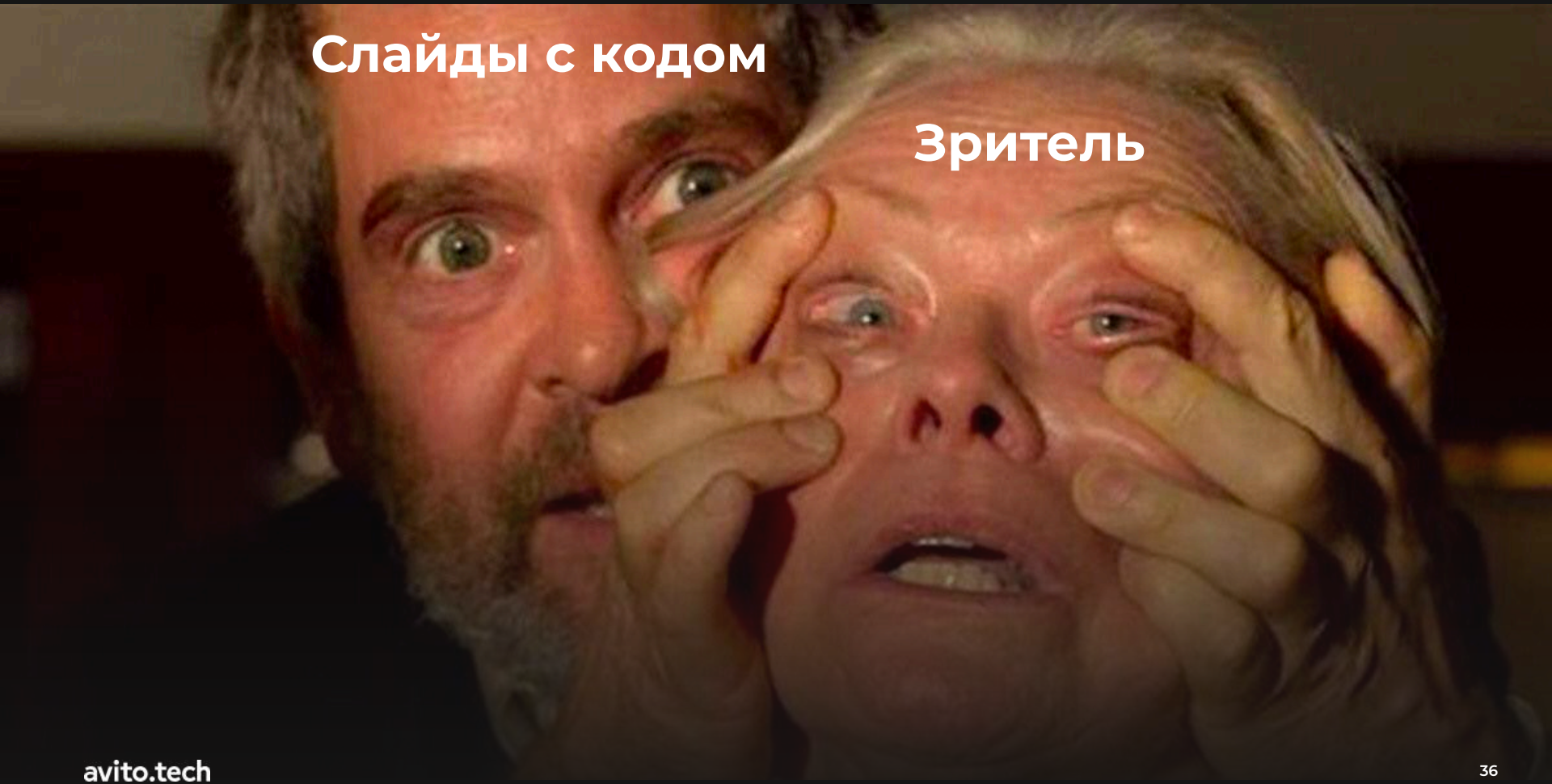
LazyVerticalGrid

MoviesPagePreview



MoviesPagePreview




Слайды с кодом
Зритель

2. Стратегии миграции

Interop API

ComponentActivity.setContent(@Composable)

```
class ComposeActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            PagesContent()
        }
    }
}
```

<ComposeView/>

```
<androidx.compose.ui.platform.ComposeView
        android:id="@+id/compose_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
```

ComposeView.setContent()

```
findViewById<ComposeView>(R.id.composeView)
    .setContent { PagesContent() }
```

ComposeView in Fragment

```
class SomeFragment : Fragment() {
    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View {
        return ComposeView(requireContext())
            .apply { setContent { PagesContent() } }
    }
}
```

@Composable AndroidView

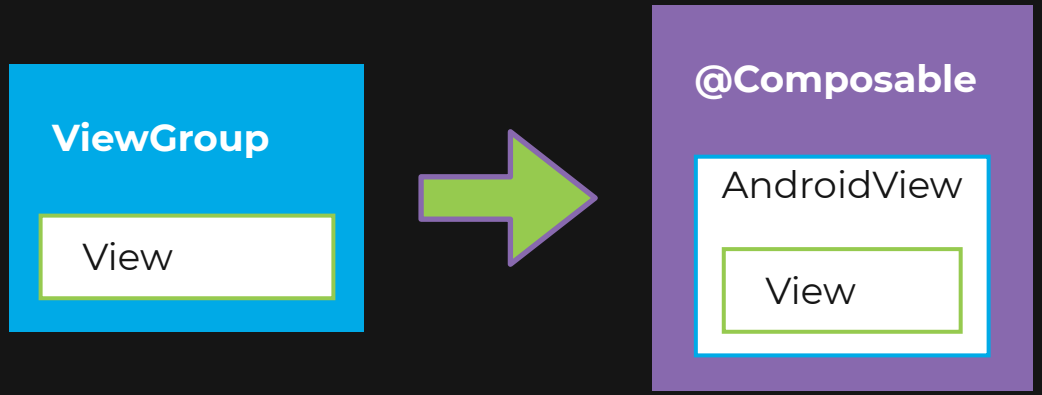
```
@Composable
fun <T : View> AndroidView(
    factory: (Context) -> T,
    modifier: Modifier = Modifier,
    update: (T) -> Unit = NoOpUpdate
)
```

AndroidView use

```
AndroidView(
    factory = { context →
        TextView(context).apply {
            layoutParams = ViewGroup.LayoutParams(
                ViewGroup.LayoutParams.MATCH_PARENT,
                ViewGroup.LayoutParams.MATCH_PARENT,
            )
            text = "Hello Omsk"
        }
    }
)
```

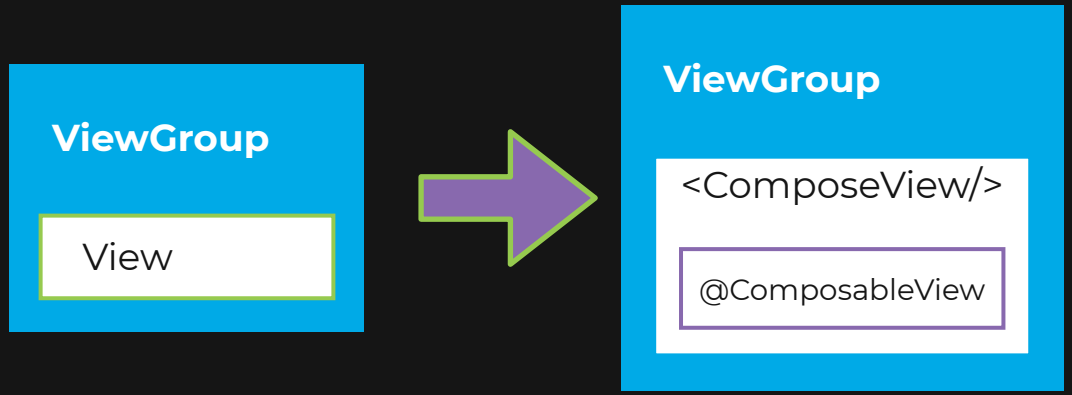
Top-down

Views inside @Composables



Bottom-up

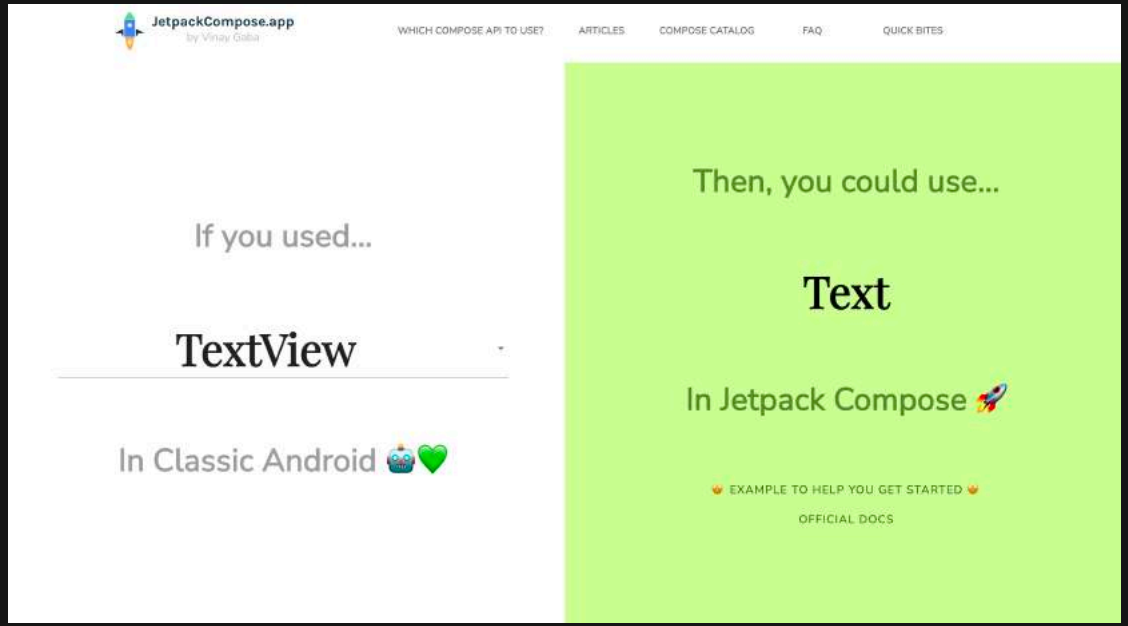
Composable inside ViewGroups



Design System

Material Components

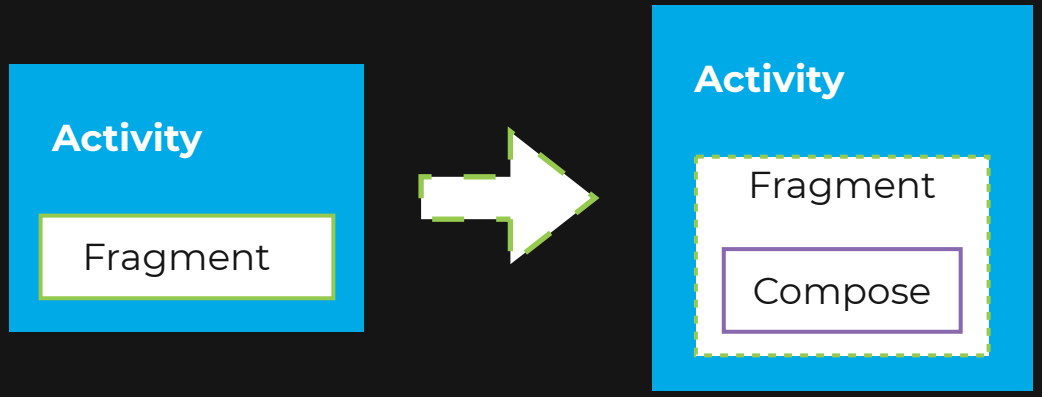
Bottom-up



https://www.jetpackcompose.app/What-is-the-equivalent-of-X-in-Jetpack-Compose

Внутри фрагментов

Можно оставлять Composable во фрагменте, не меняя навигацию
Потом уже избавляться от фрагментов



Но есть нюанс

Меняется способ связи View с данными

```
//было
textView.setText("Hello Omsk")
```

Но есть нюанс

Меняется способ связи View с данными

```
//стало
var textState: String by remember {
    mutableStateOf(
        "Hello Omsk"
    )
}
Text(text = textState)
```

3. Мигрируем View-слой

Мы посмотрим на случай со свежим проектом

Если не так - надо мигрировать с Java на Kotlin

Jetpack Compose is Kotlin exclusive

@Composable -> Kotlin compiler

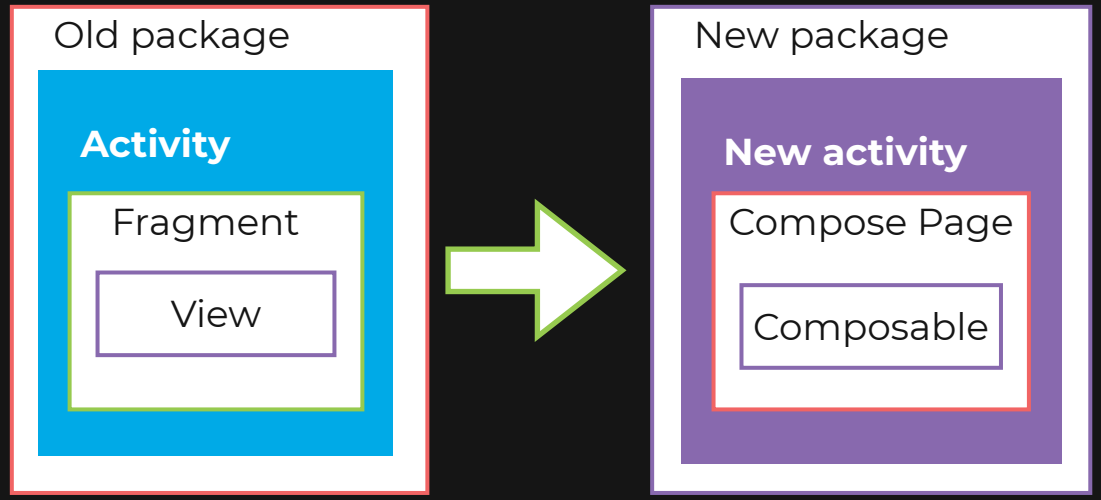
Мы посмотрим на случай со свежим проектом

Если не так - надо мигрировать
с Java на Kotlin
с RxJava на Coroutines
...

Или использовать
`androidx.compose.runtime:runtime-rxjava2/3`

Мы посмотрим на случай со свежим проектом

Если не так - надо мигрировать
с Java на Kotlin
с RxJava на Coroutines
...

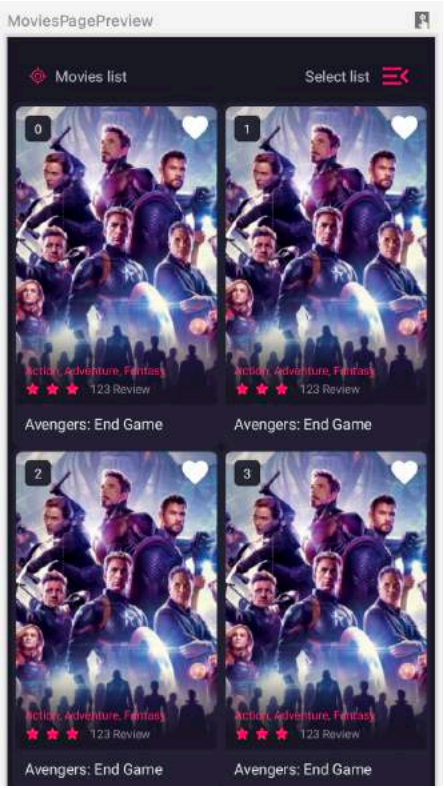


Business Logic

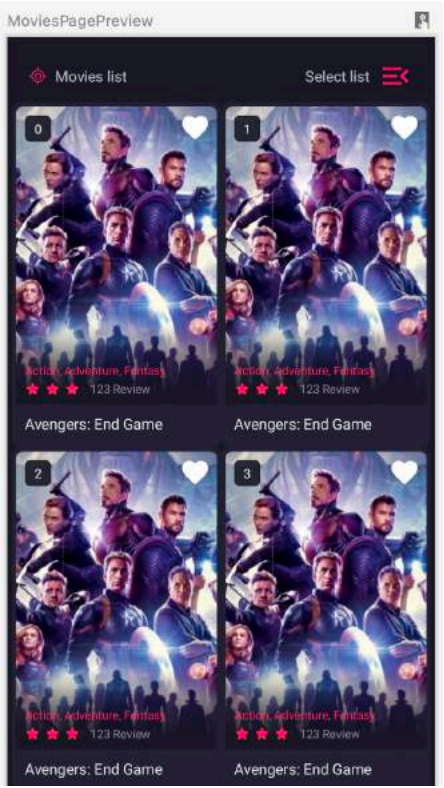
Repository (Paging)

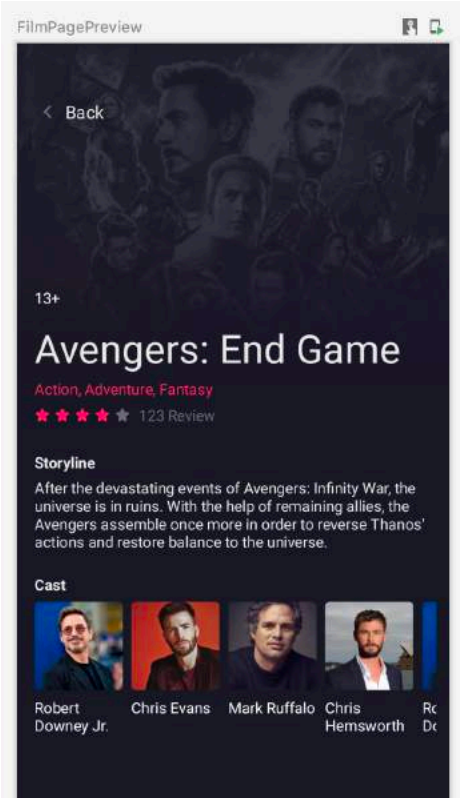
Data

У нас уже был Grid

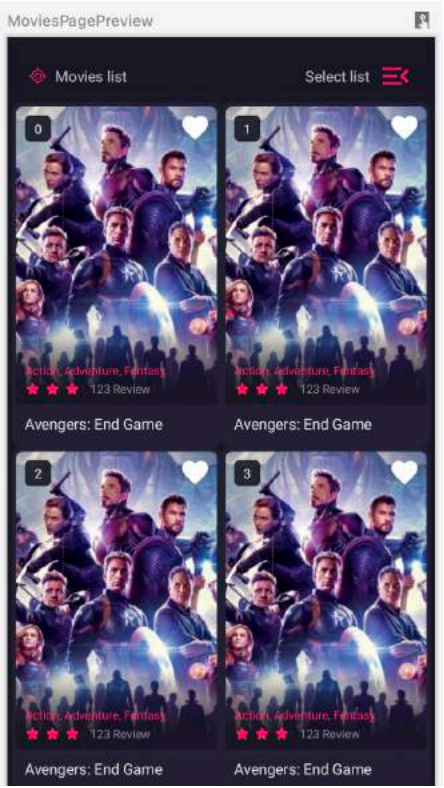


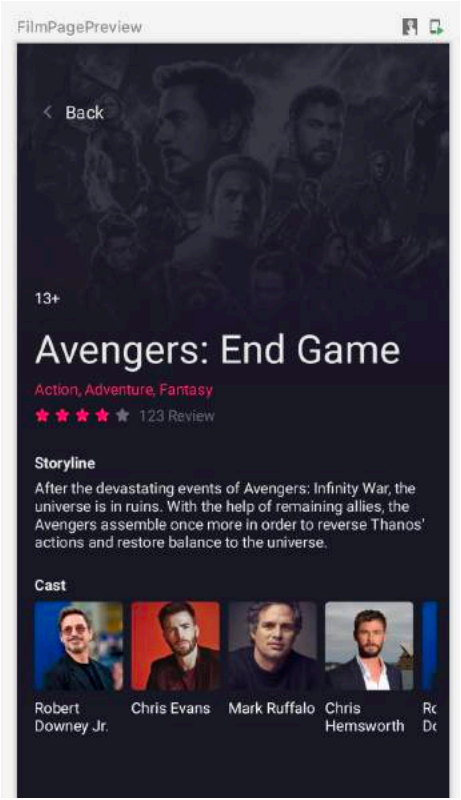
Добавляем Detail Fragment





Добавляем Detail ~~Fragment~~ Page





Compose Activity

```
class ComposeActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            PagesContent()
        }
    }
}
```

Pages Content()

```
@Composable
fun PagesContent() {
    var filmState: Film? by remember { mutableStateOf(null) }
    MyTheme {
        Scaffold(
            backgroundColor = colorResource(id = R.color.background)
        ) { ... }
    }
}
```

Pages Content()

```
@Composable
fun PagesContent() {
    var filmState: Film? by remember { mutableStateOf(null) }
    MyTheme {
        Scaffold(
            backgroundColor = colorResource(id = R.color.background)
        ) { ... }
    }
}
```

Pages Content()

```
@Composable
fun PagesContent() {
    var filmState: Film? by remember { mutableStateOf(null) }
    MyTheme {
        Scaffold(
            backgroundColor = colorResource(id = R.color.background)
        ) { ... }
    }
}
```

Pages Content()

```
@Composable
fun PagesContent() {
    var filmState: Film? by remember { mutableStateOf(null) }
    MyTheme {
        Scaffold(
            backgroundColor = colorResource(id = R.color.background)
        ) { ... }
    }
}
```

Навигация

```
when (filmState) {
    null → MoviesPage(
        films = films,
    )
    else → FilmPage(film = filmState!!)
}
```

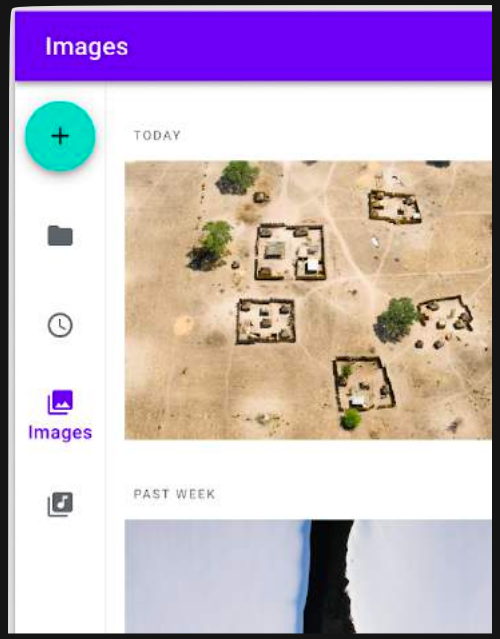
Навигация

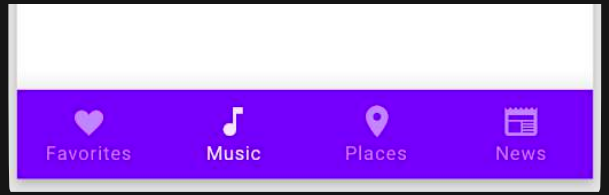
```
when (filmState) {
    null → MoviesPage(
    films = films,
    )
    else → FilmPage(film = filmState!!)
}
```

Меняем состояние

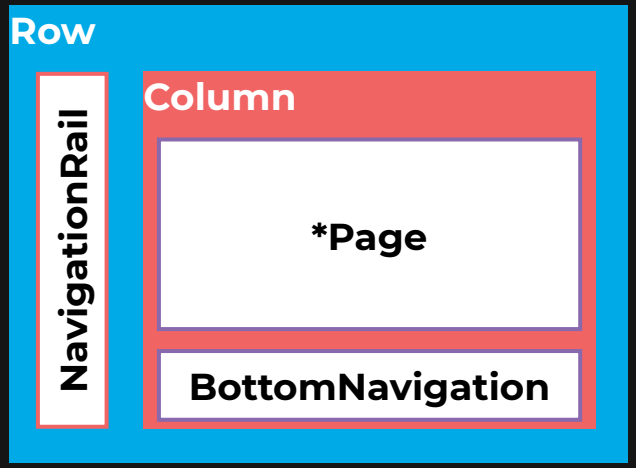
```
when (filmState) {
    null → MoviesPage(
        films = films,
        navCallback = { filmState = it }
    )
    else → FilmPage(film = filmState!!)
}
```

Добавляем BottomNavigation & NavigationRail



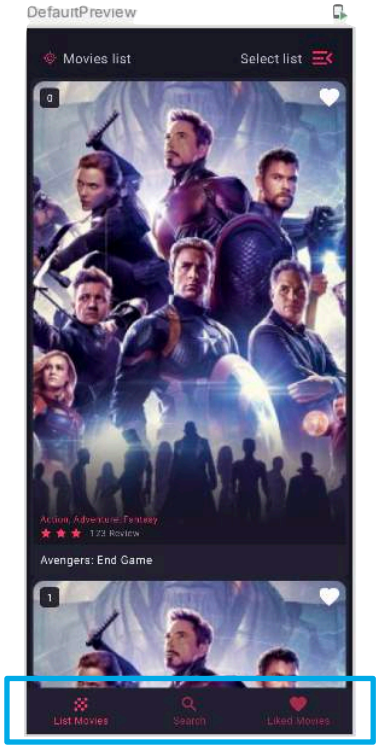


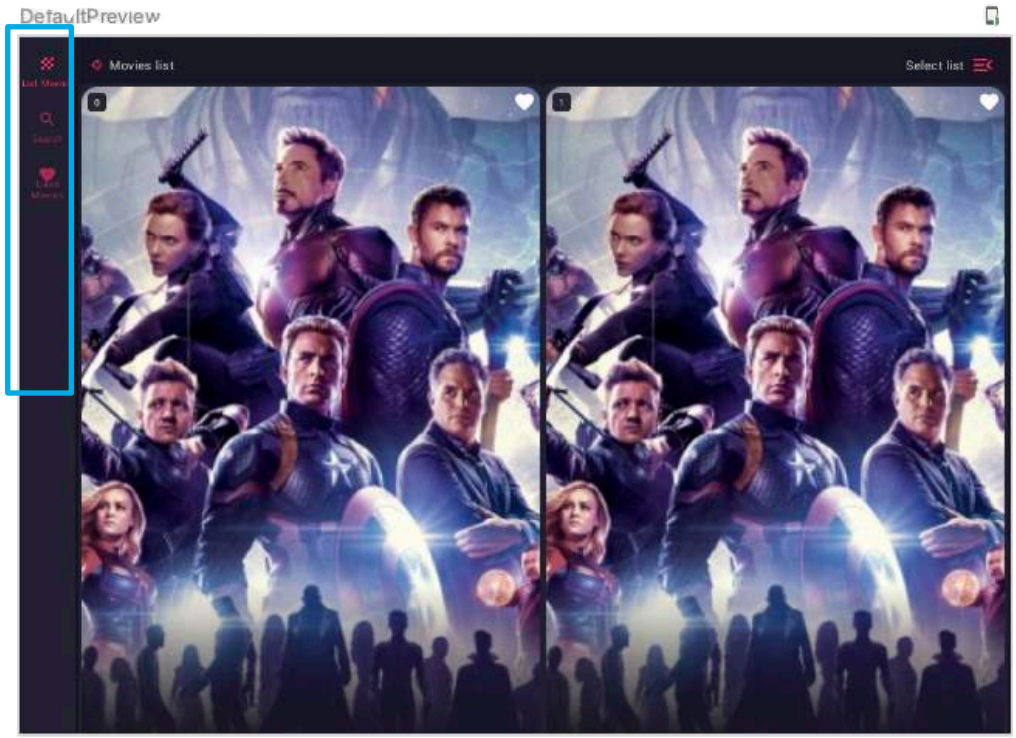
Добавляем BottomNavigation & NavigationRail



https://github.com/phansier/AFProject

BottomNavigation & NavigationRail

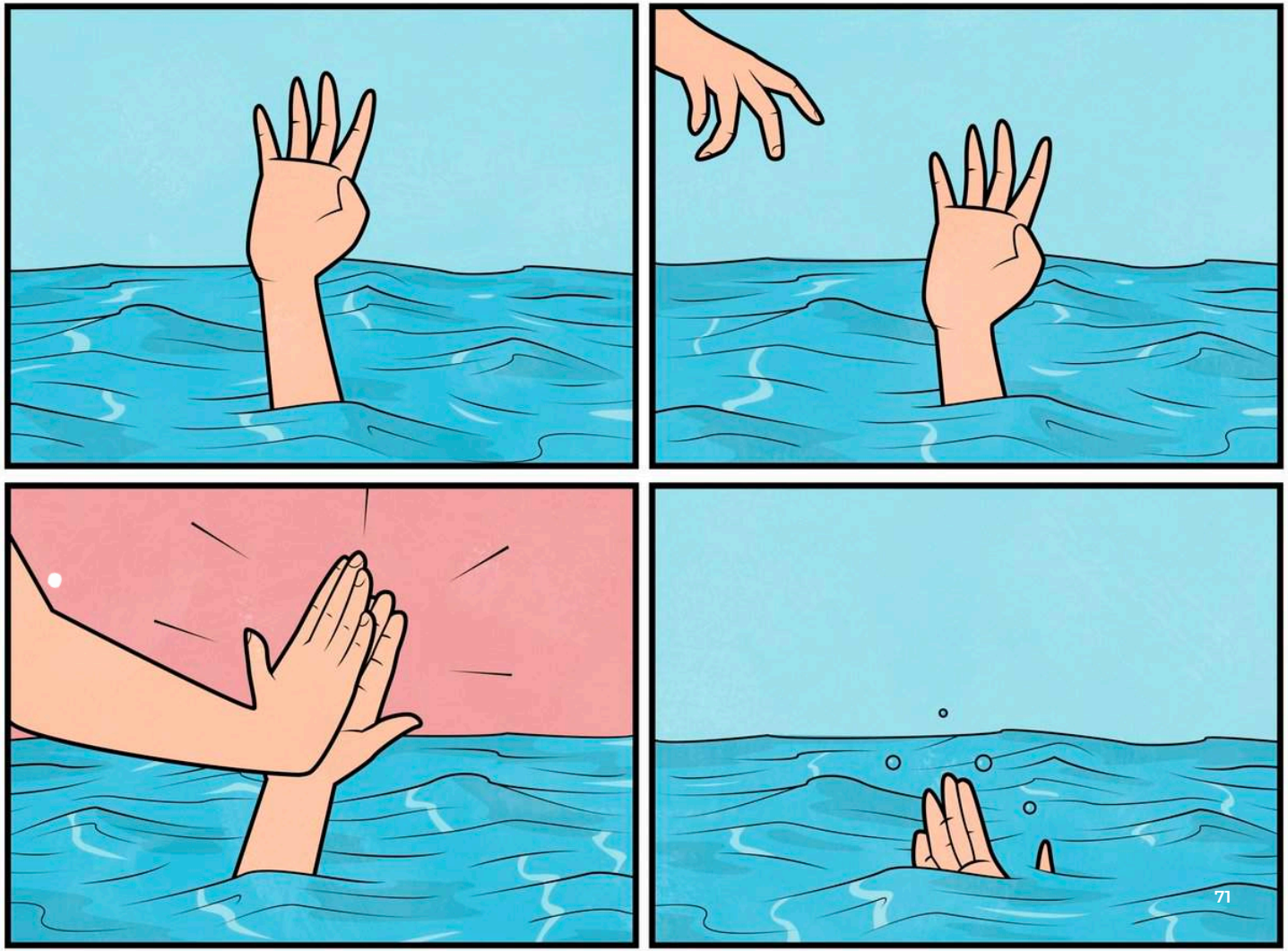




Paging & Viewmodel

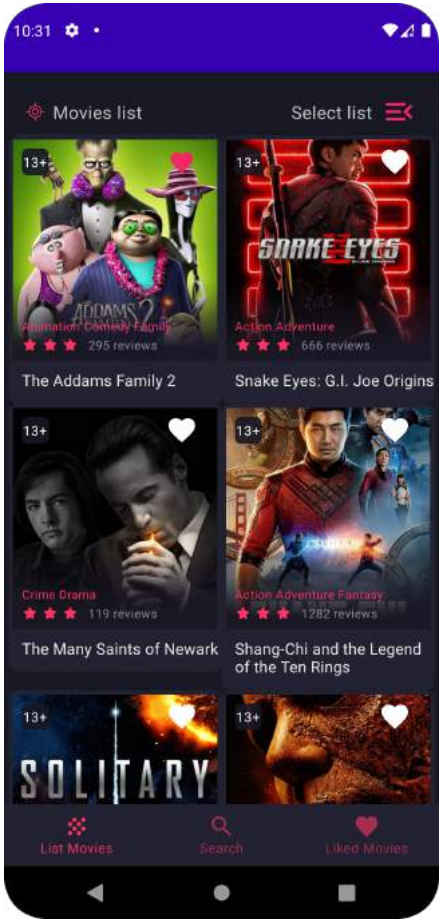
Кто использовал Paging?

Кто работал с ViewModel?



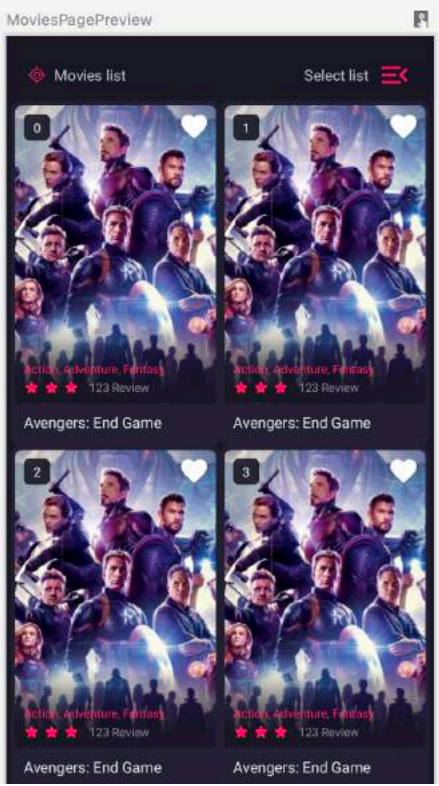
avito.tech

Что делаем


10:31
Movies list
Select list
13+
Animation Comedy Family
295 reviews
The Addams Family 2
13+
SNAKE EYES
Action Adventure
666 reviews
Snake Eyes: G.I. Joe Origins
13+
Crime Drama
119 reviews
The Many Saints of Newark
13+
Action Adventure Fantasy
1282 reviews
Shang-Chi and the Legend of the Ten Rings
13+
SOLITARY
13+
List Movies
Search
Liked Movies

Что нам нужно

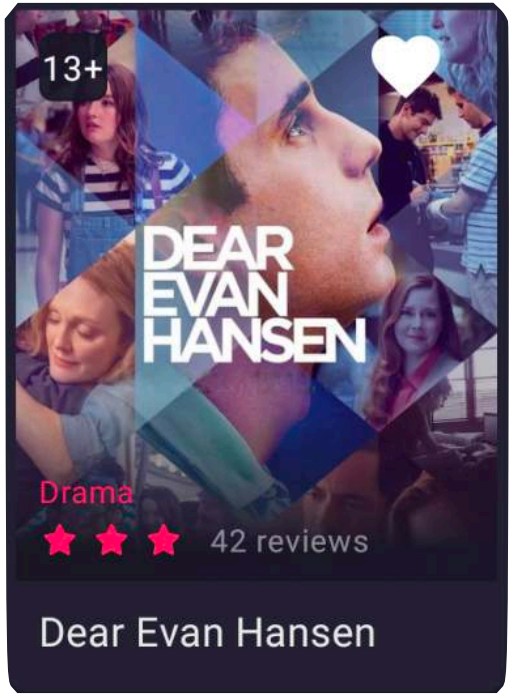
1. Lazy[Column|Row|Grid*]



Что нам нужно

1. Lazy[Column|Row|Grid*]

2. @Composable Item(Data, onClick)



Что нам нужно

1. Lazy[Column|Row|Grid*]

2. @Composable Item(Data, onClick)

3. Paging Source

```
// MoviesDataRepository
override fun getMovieListResultStream()
        : Flow<PagingData<Movie>> =
    Pager( ... )
        .flow
```

Что нам нужно

4. ViewModel

```
// MoviesViewModel
fun loadList() = repository.getStream().toStateFlow()

private fun Flow<PagingData<Movie>>.toStateFlow()
        = this.cachedIn(viewModelScope)
    .stateIn(scope = viewModelScope, ... )
```

Собираем [0]

```
class ComposeActivity : AppCompatActivity() {
    @Inject
    lateinit var factory: ViewModelFactory.Factory

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        appComponent.inject(this)
        setContent {
            PagesContent(factory.create())
        }
    }
}
```

Собираем [1]

```
val viewModel: MoviesViewModel = viewModel(factory = viewModelFactory)
val films: Flow<PagingData<Film>> = viewModel.loadFilmList()
val filmListItems: LazyPagingItems<Film> = films.collectAsLazyPagingItems()

LazyVerticalGrid( ... ) {
    itemsIndexed(items = filmListItems, ... )
    ...
}
```

Собираем [2]

```
val viewModel: MoviesViewModel = viewModel(factory = viewModelFactory)
val films: Flow<PagingData<Film>> = viewModel.loadFilmList()
val filmListItems: LazyPagingItems<Film> = films.collectAsLazyPagingItems()

LazyVerticalGrid( ... ) {
    itemsIndexed(items = filmListItems, ... )
     ...
}
```

Собираем [3]

```
val viewModel: MoviesViewModel = viewModel(factory = viewModelFactory)
val films: Flow<PagingData<Film>> = viewModel.loadFilmList()
val filmListItems: LazyPagingItems<Film> = films.collectAsLazyPagingItems()

LazyVerticalGrid( ... ) {
    itemsIndexed(items = filmListItems, ... )
    ...
}
```

Собираем [4]

```
val viewModel: MoviesViewModel = viewModel(factory = viewModelFactory)
val films: Flow<PagingData<Film>> = viewModel.loadFilmList()
val filmListItems: LazyPagingItems<Film> = films.collectAsLazyPagingItems()

LazyVerticalGrid( ... ) {
    itemsIndexed(items = filmListItems, ... )
     ...
}
```

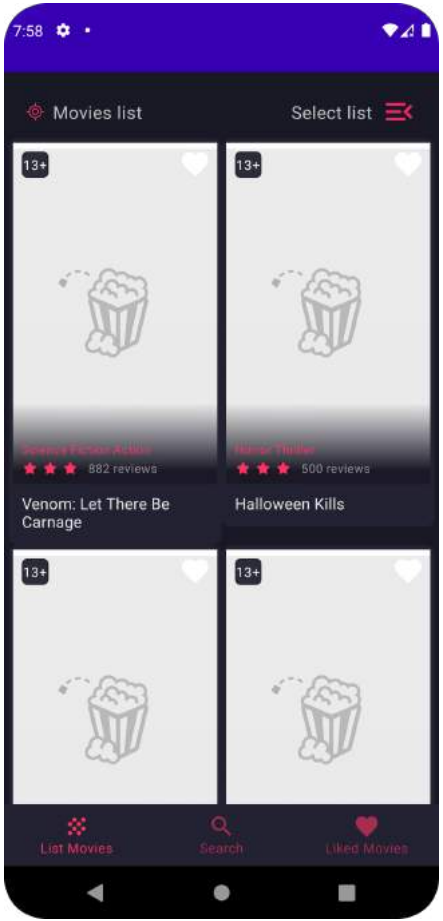
Собираем [4]

```
val viewModel: MoviesViewModel = viewModel(factory = viewModelFactory)
val films: Flow<PagingData<Film>> = viewModel.loadFilmList()
val filmListItems: LazyPagingItems<Film> = films.collectAsLazyPagingItems()

LazyVerticalGrid( ... ) {
    itemsIndexed(items = filmListItems, ... )
     ...
}
```

```
implementation 'androidx.paging:paging-compose:1.0.0-alpha13'
```

Что получили



Glide -> Coil

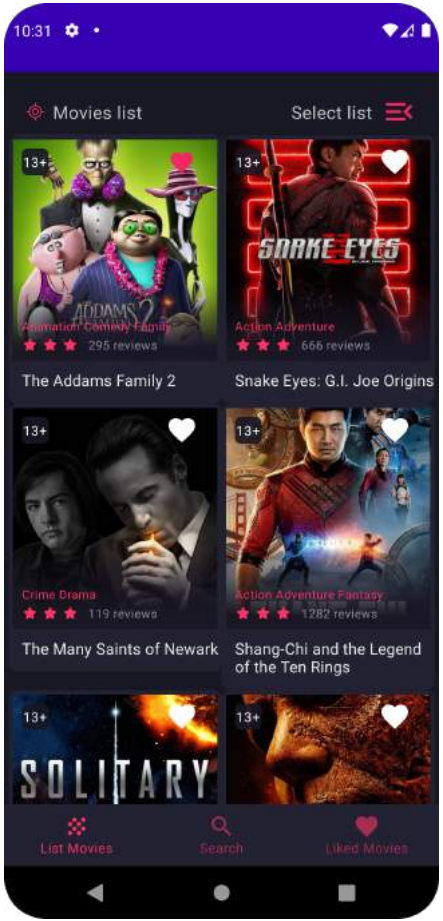
```
Image(
    painter = rememberImagePainter(
        data = film.imageUrl,
        builder = {
            placeholder(R.drawable.film_placeholder)
        }
    ),
    ...
)
```

```
implementation("io.coil-kt:coil-compose:1.4.0")
```

Glide -> Coil

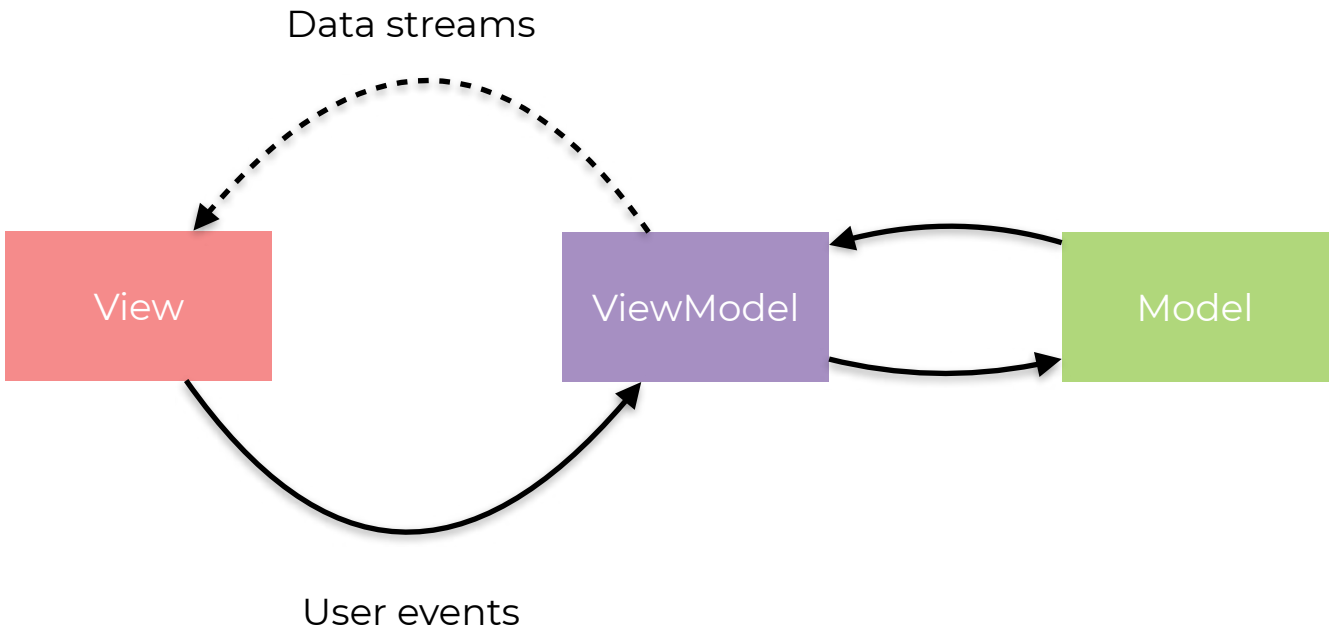
```
Image(
    painter = rememberImagePainter(
        data = film.imageUrl,
        builder = {
            placeholder(R.drawable.film_placeholder)
        }
    ),
    ...
)
```

Что получили



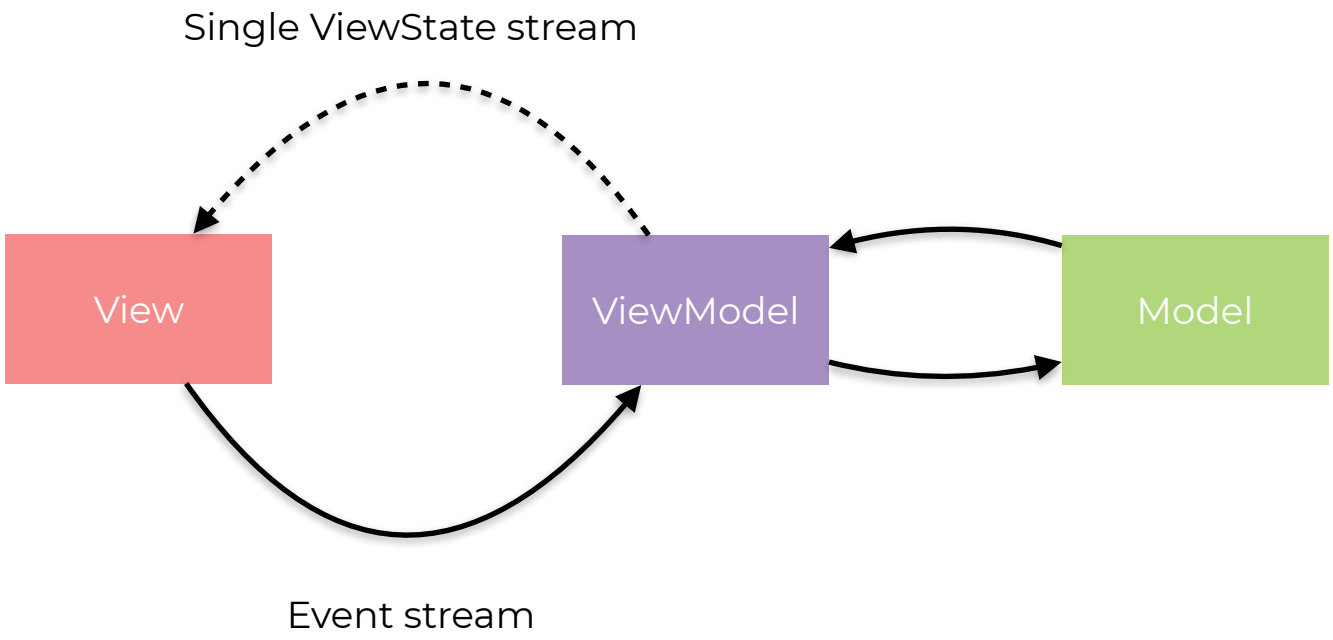

ГДЕ ПАПА, МАЛЫШ?
ВЕРНО!
А ГДЕ МАМА?
ДА!
А ТЕПЕРЬ СКАЖИ
Где MVI?

MVVM -> MVI



https://medium.com/mobile-at-octopus-energy/17d8170a13fd

MVVM -> MVI



MVVM -> MVI

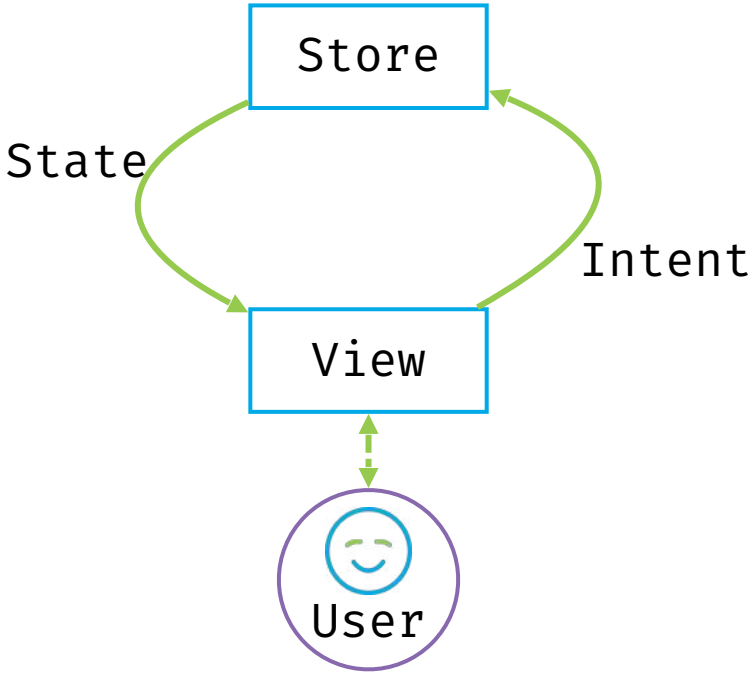
```
@Composable
fun PagesContent(
    navigationViewModel: NavigationViewModel,
    filmsViewModel: FilmsViewModel,
    likesViewModel: LikesViesModel
)
```

```
@Composable
fun PagesContent(
    commonViewModel: CommonViewModel,
)
```

MVI

Jetpack Viewmodel

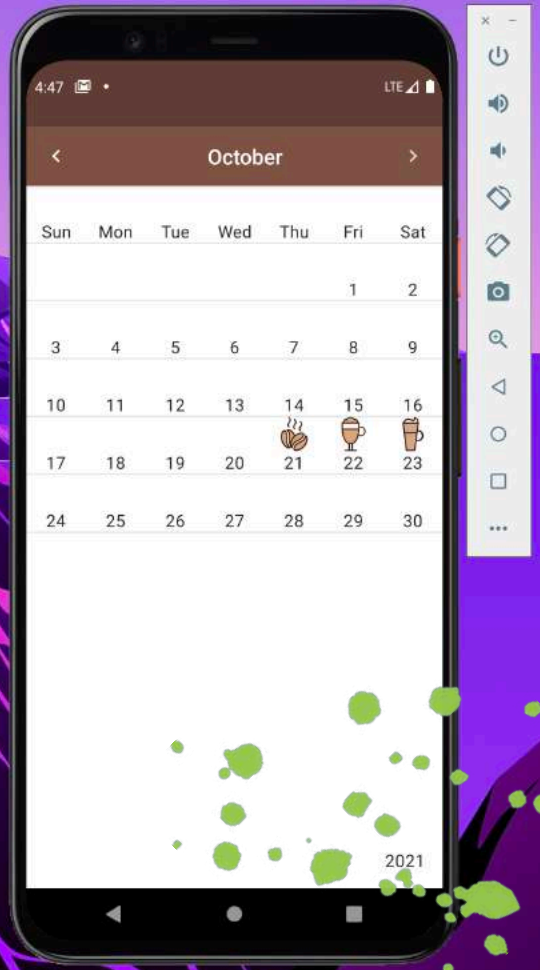
MVI




```
abstract class Store<Intent : Any, State : Any> {
    val state: StateFlow<State>
        get() = _state

    fun newIntent(intent: Intent) {
        _intentChannel.offer(
            intent
        )
    }
}
```

Трансформация Android-разработки с Jetpack Compose и Корутинами



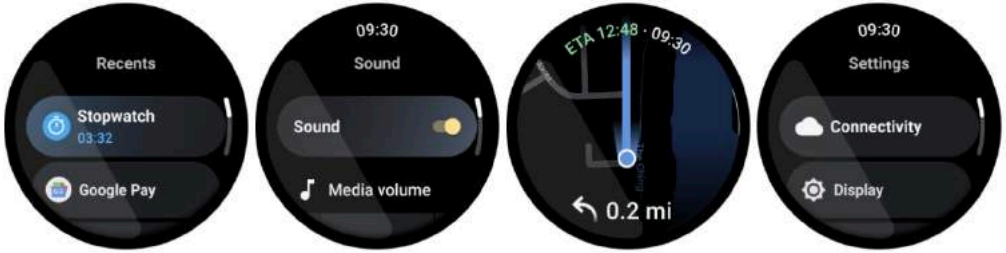
4. Другие возможности Compose

Интеграция в Jetpack

- Activity Result API - `androidx.activity.compose`
- ViewModel - `androidx.lifecycle.viewmodel.compose`
- LiveData - `androidx.compose.runtime:runtime-livedata`
- Navigation Component - `androidx.navigation:navigation-compose`
- Hilt - `androidx.hilt:hilt-navigation-compose`
- Paging - `androidx.paging:paging-compose`
- RxJava - `androidx.compose.runtime:runtime-rxjava2/3`

Compose for Wearables

13 октября -
androidx.wear.
compose.material:1.0.0-alpha08

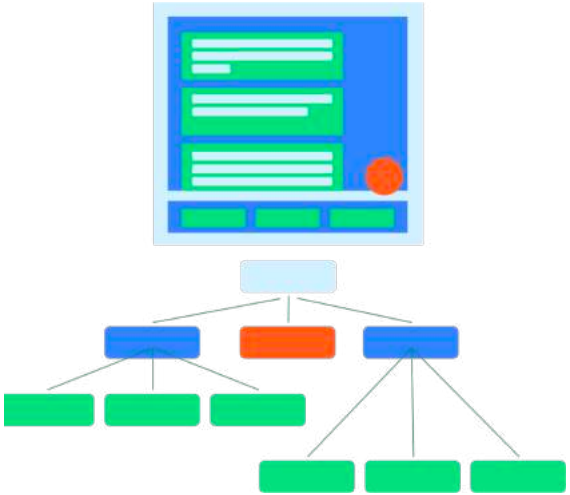


Chip ToggleChip BasicCurvedText TimeText

В планах Homescreen widgets

https://developer.android.com/jetpack/androidx/releases/wear-compose
https://developer.android.com/jetpack/androidx/compose-roadmap

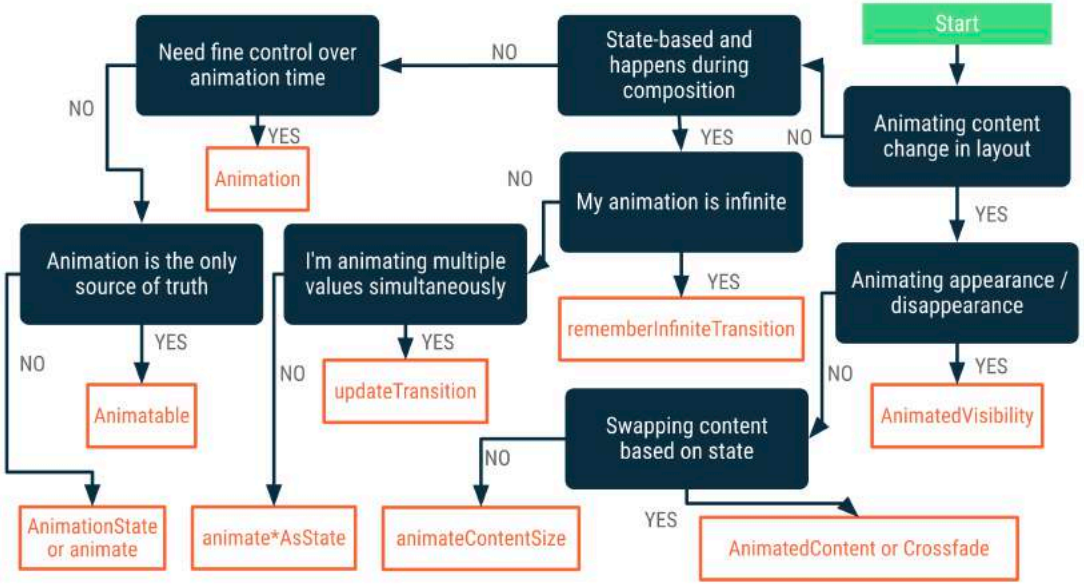
Тестирование



Kakao Compose

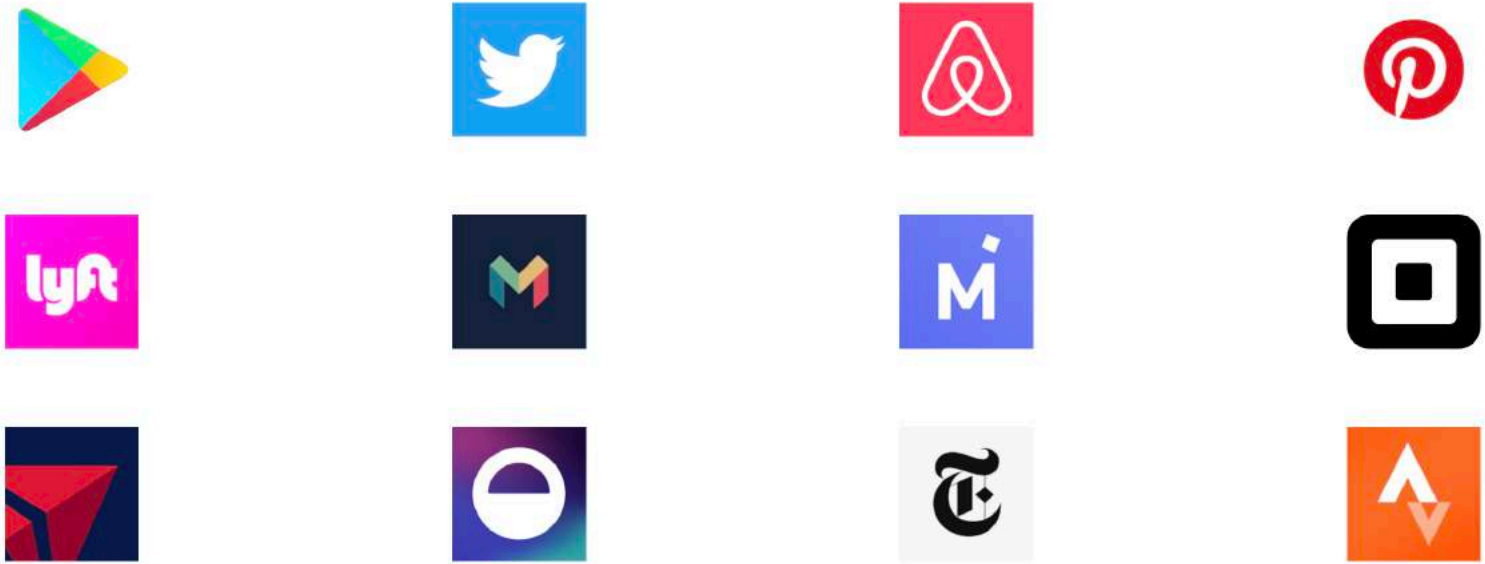
Kaspresso Compose (WIP)

Анимации



Lottie Compose

Apps built with Compose

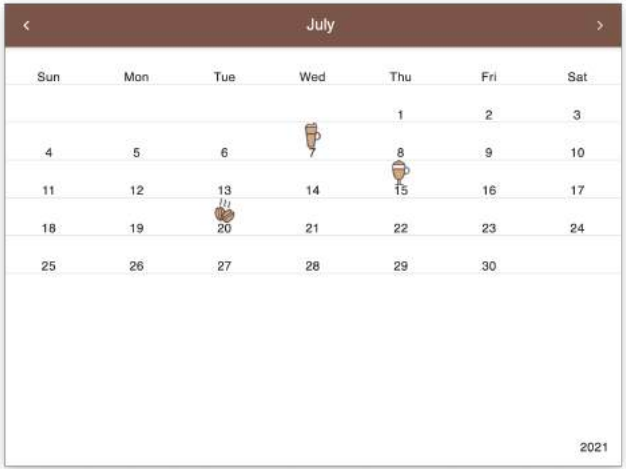


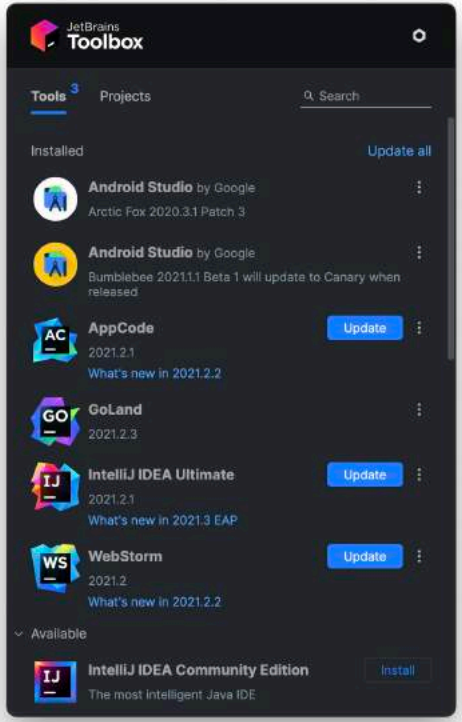
https://developer.android.com/jetpack/compose

Compose Multiplatform

Desktop + Web - v1.0.0-alpha3

https://github.com/phansier/Coffeegram-Desktop





Stable release - 2021

https://github.com/JetBrains/compose-jb

https://compose-web.ui.pages.jetbrains.team

Что посмотреть?

01. Доклад “Темные стороны Jetpack Compose”

02. Статья Трансформация Android-разработки с Jetpack Compose и Корутинами

03. Примеры кода https://github.com/phansier/AFProject - для этого доклада https://github.com/phansier/Coffeegram https://github.com/phansier/Coffeegram-Desktop

04. Рассылка https://jetc.dev/

avito.tech

Google Developer Groups
Omsk, Russia

Jetpack Compose

Андрей Берюхов

https://github.com/phansier

https://t.me/phansier

Ссылки:

https://beryukhov.ru

https://github.com/phansier/AFProject