образование ПОЛИТЕХ одноклассники экосистема

Custom view, animations and touches

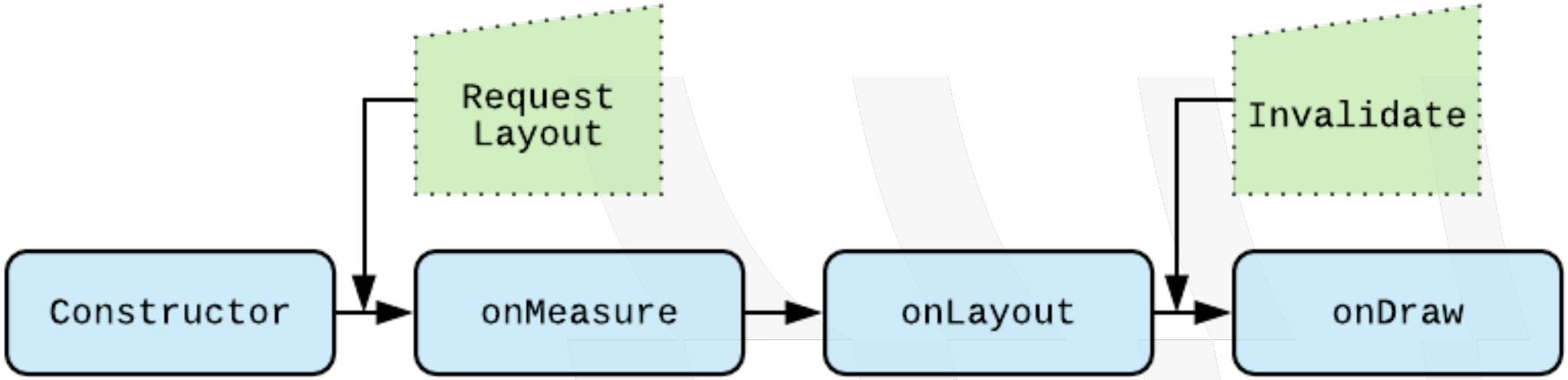
Малков Павел

План лекции
1. Зачем они нужны
2. Чем View отличается от ViewGroup
3. Особенности конструкторов
4. Жизненный цикл вью
5. Как правильно использовать onMeasure (квадратные вью, wrap_content, match_parent)
6. Как правильно располагать элементы внутри onLayout
7. onDraw
8. Зачем нужны onAttachToWindow, onDetachFromWindow
9. Кастомные проперти в declare-styleable
10. Анимации: примеры, когда использовать animate, а когда ValueAnimator
11. Конец

Зачем нужны кастомные вьюшки?

1. Не хватает стандартных лейаутов
2. Нужно оптимизировать рендеринг вью (особенно, в списках)
3. Захотелось красивых анимаций
4. Нужна особая обработка касаний

Жизненный цикл



Создание view

Конструктор

```
public class CustomView @JvmOverloads constructor(
    context: Context,
    attrs: AttributeSet? = null,
    @AttrRes defStyleAttr: Int = 0,
    @StyleRes defStyleRes: Int = 0,
) : View(context, attrs, defStyleAttr)
```

XML

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <androidx.appcompat.widget.AppCompatTextView
        android:id="@+id/title"
        android:layout_width="wrap_content"
        android:text="Hello, custom view!"
        android:layout_height="wrap_content"/>

    <ru.ok.widgets.CustomView
        android:id="@+id/custom_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

Конструктор

```
public class CustomView @JvmOverloads constructor(
    context: Context,
    attrs: AttributeSet? = null,
    @AttrRes defStyleAttr: Int = 0,
    @StyleRes defStyleRes: Int = 0,
) : View(context, attrs, defStyleAttr)
```

Конструктор (в java)

```
public class CustomView extends View {
    public CustomView(Context context) {
        this(context, null);
    }

    public CustomView(Context context, @Nullable AttributeSet attrs) {
        this(context, attrs, 0);
    }

    public CustomView(Context context, @Nullable AttributeSet attrs, @AttrRes int defStyleAttr) {
        this(context, attrs, defStyleAttr, 0);
    }

    public CustomView(Context context, @Nullable AttributeSet attrs, @AttrRes int defStyleAttr,
@StyleRes int defStyleRes) {
        super(context, attrs, defStyleAttr, defStyleRes);
    }
}
```

Как верстать такие view?

1. Руками: то есть мы вручную объявляем какие объекты будут располагаться внутри вью
2. Создать верстку внутри XML: используя тег <merge/> и вызов статической функции View.infalte

образование ПОЛИТЕХ одноклассники экосистема

Как использовать <merge/>

```kotlin
public class CustomView @JvmOverloads constructor(
    context: Context,
    attrs: AttributeSet? = null,
    @AttrRes defStyleAttr: Int = 0,
    @StyleRes defStyleRes: Int = 0
) : FrameLayout(context, attrs, defStyleAttr, defStyleRes) {

    init {
        // здесь мы парсим нашу xml и добавляем две textview как children
        View.inflate(context, R.layout.custom_view, this)

        val title = findViewById<TextView>(R.id.custom_view_title)
        val subtitle = findViewById<TextView>(R.id.custom_view_subtitle)
    }
}
```

```xml
<merge xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:parentTag="ru.ok.widgets.CustomView">

    <androidx.appcompat.widget.AppCompatTextView
        android:id="@+id/custom_view_title"
        android:gravity="top|center"
        android:text="Привет" />

    <androidx.appcompat.widget.AppCompatTextView
        android:id="@+id/custom_view_subtitle"
        android:gravity="bottom|right"
        android:text="Я кастомная вьюшка" />
</merge>
```

Что такое ViewGroup?

Если View не может иметь детей, то ViewGroup это базовый класс для вьюшек, которые содержать в себе список children (это список из экземпляров класса View).
Самые популярные ViewGroup: FrameLayout, LinearLayout, ConstraintLayout, CoordinatorLayout, RecyclerView.

Для чего нужен onMeasure?

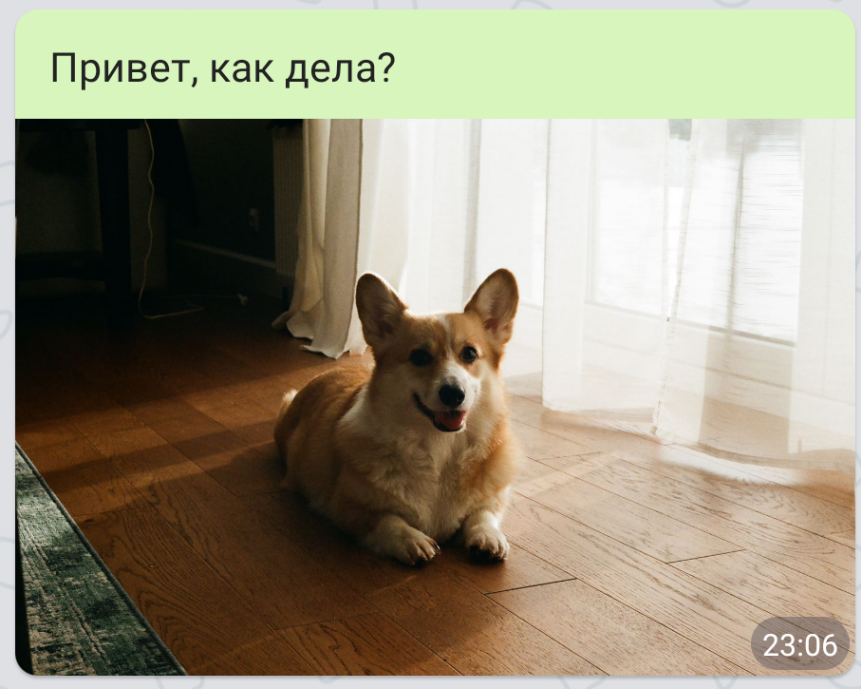
Внутри этого метода идет расчет размеров вью и его детей.
2 передаваемых аргумента это MeasureSpec. По сути это битовая маска, в которой лежит размер и режим работы спеки (значения берутся из layout_width/layout_height).

Есть 2 режима:

- EXACTLY - заданный размер (80dp или match_parent), значения уже посчитаны
- AT_MOST - не более чем какая-то величина, допустим мы объявили wrap_content, а родитель имеет значение 200dp.
- UNPECIFIED - не задано, используется, например в ScrollView, позволяет ей расти неограниченно в высоту

образование ПОЛИТЕХ одноклассники экосистема

Для чего нужен onMeasure?

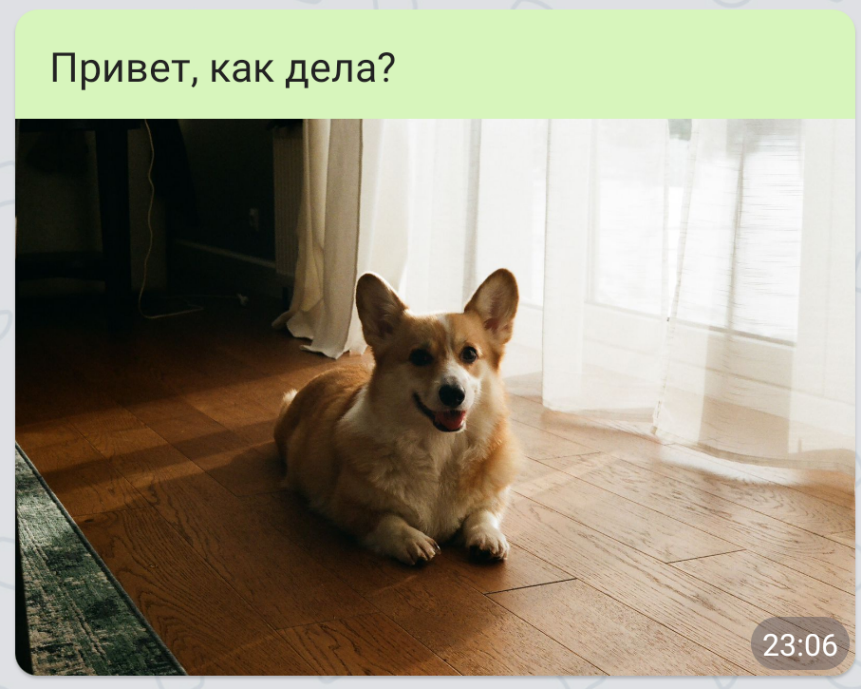


Здесь есть 3 вью внутри MessageView:
1. TextView с текстом «Привет, как дела?
2. ImageView с картинкой
3. TextView с датой

```
override fun onMeasure(widthMeasureSpec: Int, heightMeasureSpec: Int) {
    val widthMeasureSpecForChildren = MeasureSpec.makeMeasureSpec(200.dp, MeasureSpec.AT_MOST)

    var width = 0
    var height = 0

    text.measure(widthMeasureSpecForChildren, heightMeasureSpec)
    height += text.height
    if (text.measuredWidth > width) {
        width = text.measuredWidth
    }

    image.measure(widthMeasureSpecForChildren, heightMeasureSpec)
    height += image.height
    if (image.measuredWidth > width) {
        width = image.measuredWidth
    }

    setMeasuredDimension(width, height)
}
```

образование ПОЛИТЕХ одноклассники экосистема

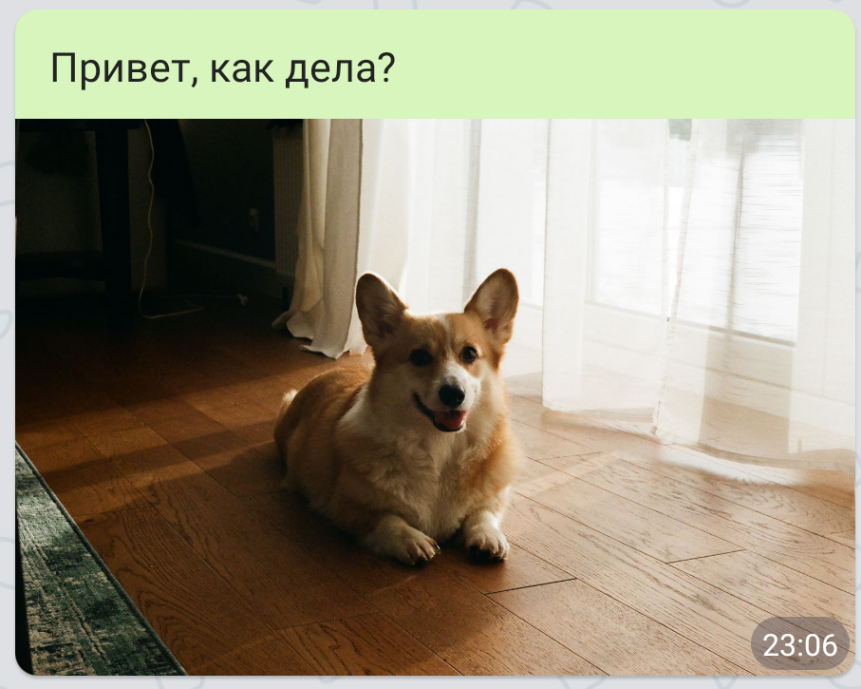
Для чего нужен onLayout?




```
override fun onLayout(changed: Boolean, l: Int, t: Int, r: Int, b: Int) {
    text.layout(
        /*left*/paddingStart,
        /*top*/paddingTop,
        /*right*/paddingStart + text.width,
        /*bottom*/paddingTop + text.height
    )

    val imageTop = text.bottom + 2.dp
    image.layout(0, imageTop, width, imageTop + image.height)

    val dateLeft = width - date.width - 2.dp
    val dateTop = height - date.height - 2.dp
    date.layout(
        dateLeft,
        dateTop,
        dateLeft + date.width,
        dateTop + date.height
    )
}
```

образование ПОЛИТЕХ одноклассники экосистема

Для чего нужен dispatchDraw?



Здесь необходимо закруглить углы
Для этого зададим параметры в path внутри onSizeChanged
А затем в onDraw сделаем clipPath для canvas

```
private val path = Path()
private val cornerRadius = 15.dpf
private val rect = RectF()

override fun onSizeChanged(w: Int, h: Int, oldw: Int, oldh: Int) {
    super.onSizeChanged(w, h, oldw, oldh)
    rect.set(0f, 0f, w.toFloat(), h.toFloat())
    path.reset()
    path.addRoundRect(rect, cornerRadius, cornerRadius, Path.Direction.CW)
    path.close()
}

override fun dispatchDraw(canvas: Canvas) {
    canvas.withSave {
        canvas.clipPath(path)
        super.dispatchDraw(canvas)
    }
}
```

образование ПОЛИТЕХ одноклассники экосистема

Кастомные проперти в xml

Парсинг attrs

```
context.obtainStyledAttributes(attrs, R.styleable.CustomView, defStyleAttr, defStyleRes)
    .use {
        cornerRadius = it.getDimensionPixelSize(R.styleable.CustomView_cv_corner_radius, 15.dp)
    }
```

Файл styleable.xml

```
<declare-styleable name="CustomView">
    <attr name="cv_corner_radius" format="dimension" />
</declare-styleable>
```

Использование в другом лейауте

```
…<ru.ok.widgets.CustomView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:cv_corner_radius="28dp" />
…
```

onAttachedToWindow

Иногда полезно знать, когда вью добавили на экран, для этого есть вызов onAttachedToWindow.
Есть другой метод: onDetachedFromWindow. Он вызывается когда вью удалили с экрана.

Анимации

```
text.animate()
    .scaleX(2f)
    .rotation(90f)
    .setInterpolator(DecelerateInterpolator(1.3f))
    .setDuration(600L)
    .start()
```

```
val textAnimator = ObjectAnimator.ofPropertyValuesHolder(text,
    PropertyValuesHolder.ofFloat(View.SCALE_X, 2f),
    PropertyValuesHolder.ofFloat(View.ROTATION, 90f),
)

val imageAnimator = ObjectAnimator.ofFloat(image, View.SCALE_X, 2f)

val animatorSet = AnimatorSet().apply {
    interpolator = DecelerateInterpolator(1.3f)
    duration = 600L
    playTogether(textAnimator, imageAnimator)
}

animatorSet.start()
```

Спасибо!

образование

ПОЛИТЕХ

одноклассники
экосистема