ОРГАНИЗАЦИЯ И ИНФРАСТРУКТУРА ДИЗАЙН-СИСТЕМЫ

Avito

Обо мне

Данилов Андрей

Avito @ Design System

@deadmc
@Andrey__Danilov

Avito

Чем занимаются в дизайн-системе?

[По мнению других разработчиков]

- Красят кнопки
- Решительно ничем

Цели дизайн-системы как команды

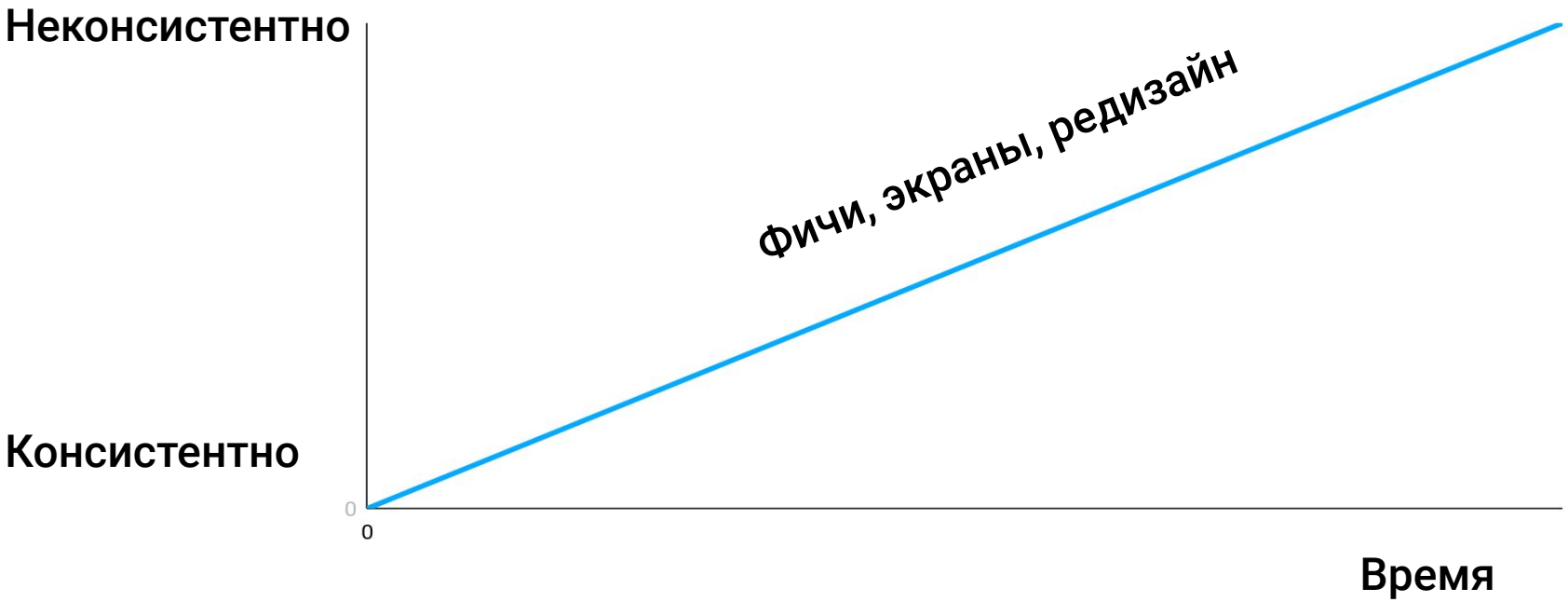
- консистентность
- управляемость

Консистентность

- UI - цвета, размеры, шрифты и компоненты по приложению выглядят единообразно
- UX - одинаковое и ожидаемое поведение на всех экранах приложения

Откуда берется неконсистентность?

- Новые экраны
- Редизайн


Неконсистентно
Консистентно
0
0
Фичи, экраны, редизайн
Время

Avito

Управляемость

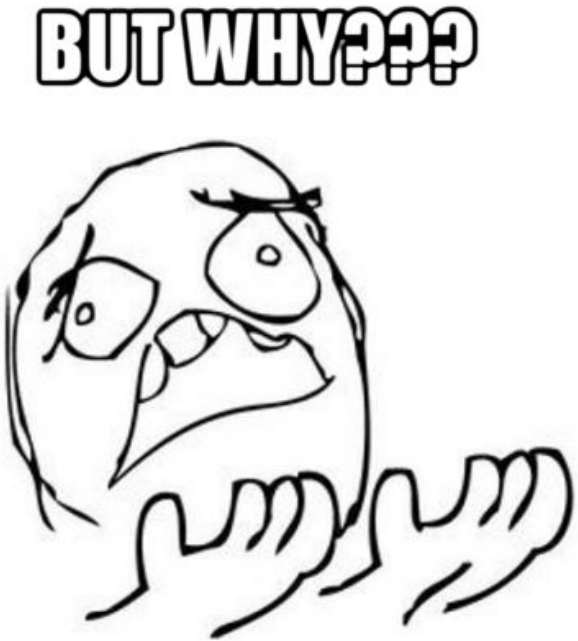
- Цвета
- Размеры
- Шрифты

Управляемость

Состояние $x(t_1)$ линейной системы управляемо, если существует момент времени $t_2 > t_1$ и такой вход, который переводит состояние системы $x(t_1)$ в состояние $x(t_2)=0$ (начало координат), при условии, что интервал $(t_2 - t_1)$ конечен.

Управляемость

Чем меньше действий нужно сделать для смены параметров во всем приложении, тем выше управляемость



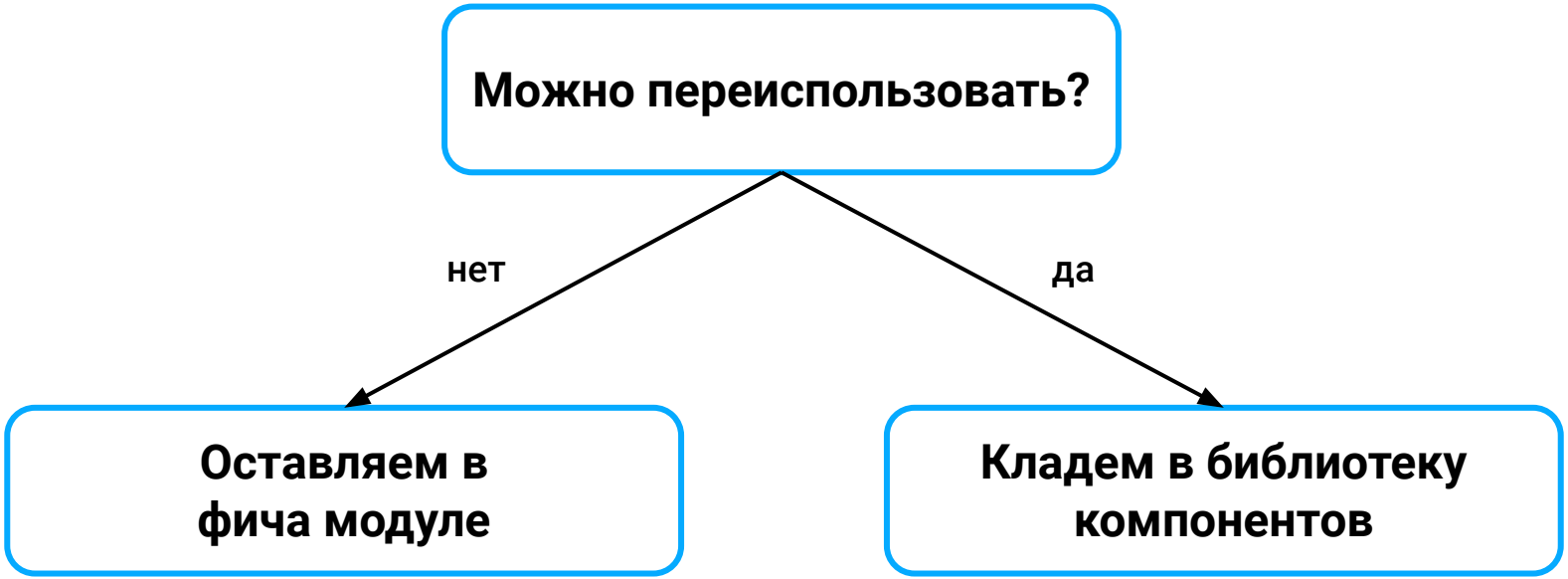
Avito

Как это выглядит со стороны?

Avito

БИБЛИОТЕКА КОМПОНЕНТОВ

Avito


Можно переиспользовать?
нет
да
Оставляем в фича модуле
Кладем в библиотеку компонентов


ON
STACKS
ON
STACKS
ON
STACKS
ON
STACKS
ON
STACKS
ON
STACKS
ON
STACKS
ON
STACKS
ON
STACKS
ON

Avito


OUR SUN
NEUTRON STARS
BLACK HOLES
UI-COMPONENTS
HEAVIEST OBJECTS IN THE UNIVERSE

Avito

Очевидные проблемы

- Некотролируемые изменения в компоненты
- Множественные форки
- Не всегда понятно есть ли подходящий компонент

Avito

Сколько итераций библиотек компонентов было в Авито?

2

3

4

5

Avito

Сколько итераций библиотек компонентов было в Авито?

2

3

4

5

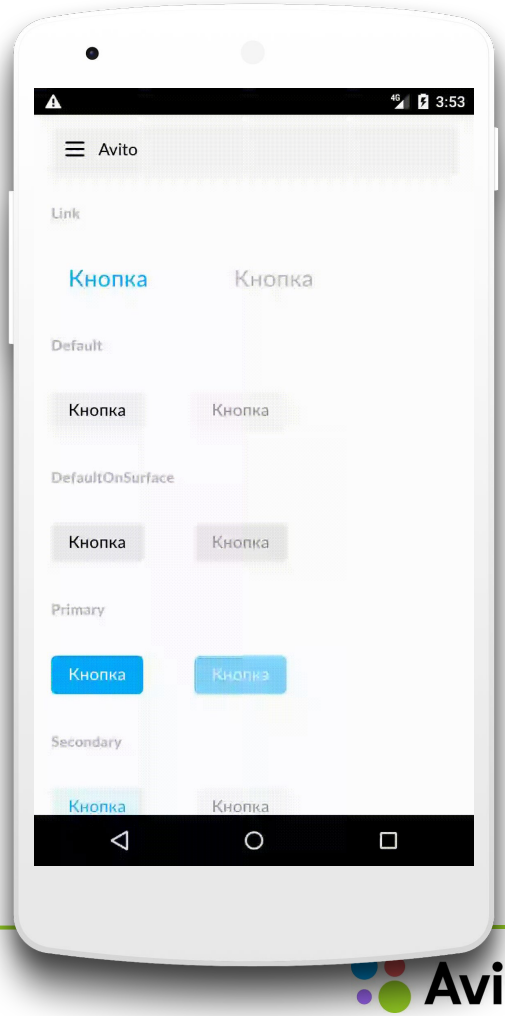
Avito

К чему пришли

- Все компоненты должны иметь четкую спецификацию для дизайнеров и разработчиков
- У библиотеки компонентов обязательно должен быть code owner
- Библиотеку компонентов как часть дизайн системы должен пушить дизайнер

Design App

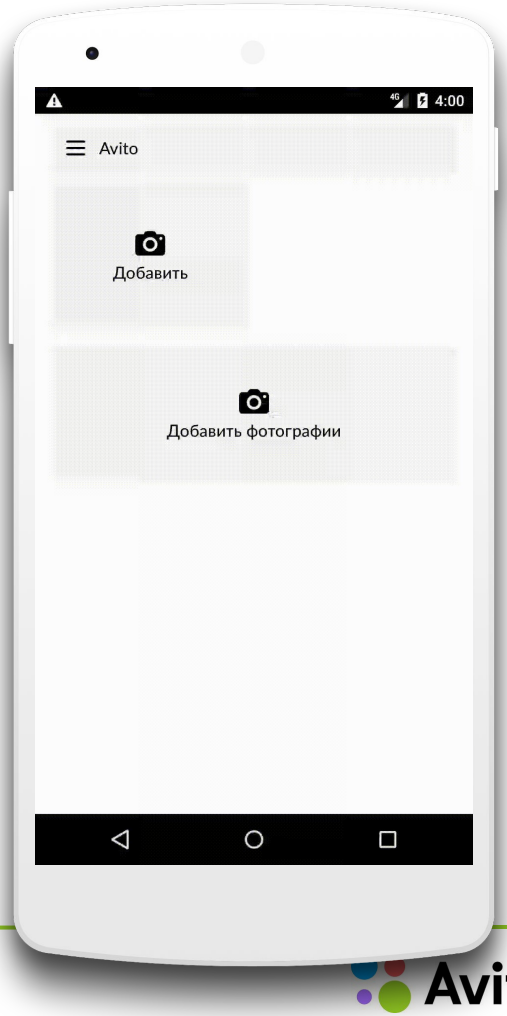
- Содержит в себе все компоненты с реальными примерами использования



Avito

Design App

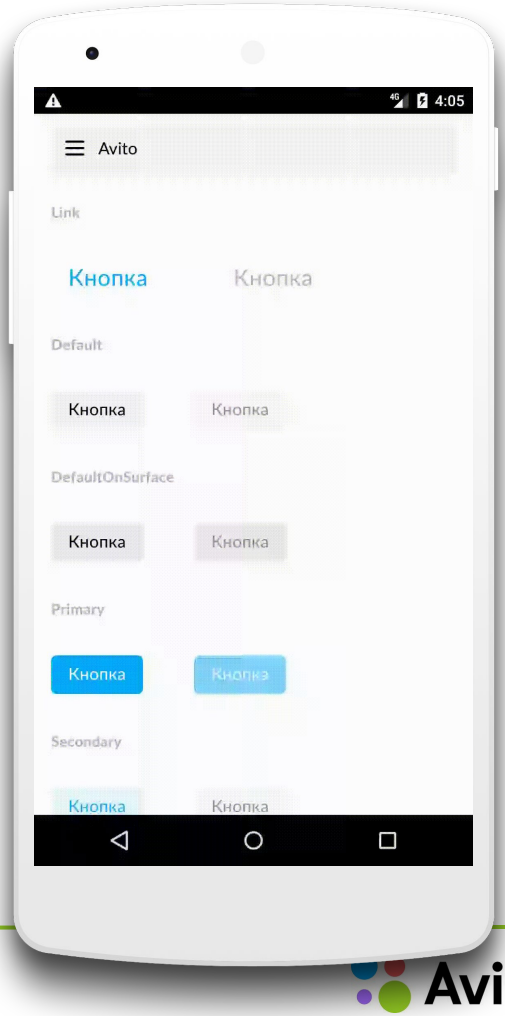
- Выступает витриной как для дизайнеров, так и для разработчиков



Avito

Design App

- Поддерживает темы разных приложений



Avito

РЕФАКТОРИНГ БАЗОВЫХ СУЩНОСТЕЙ

Avito

Масштаб монорепа

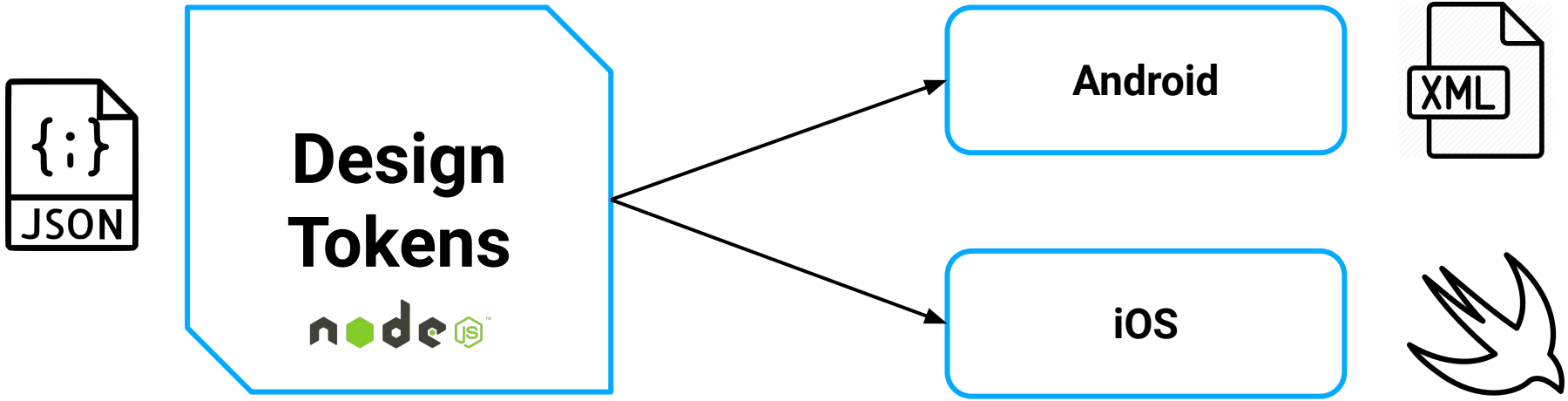
AUTOTEKA

Масштаб монорепа

- 313 продуктовых модулей + 69 в github
- 878 779 строк кода + 58 341 в github
- 1 894 drawable ресурсов
- 1 596 layout ресурсов

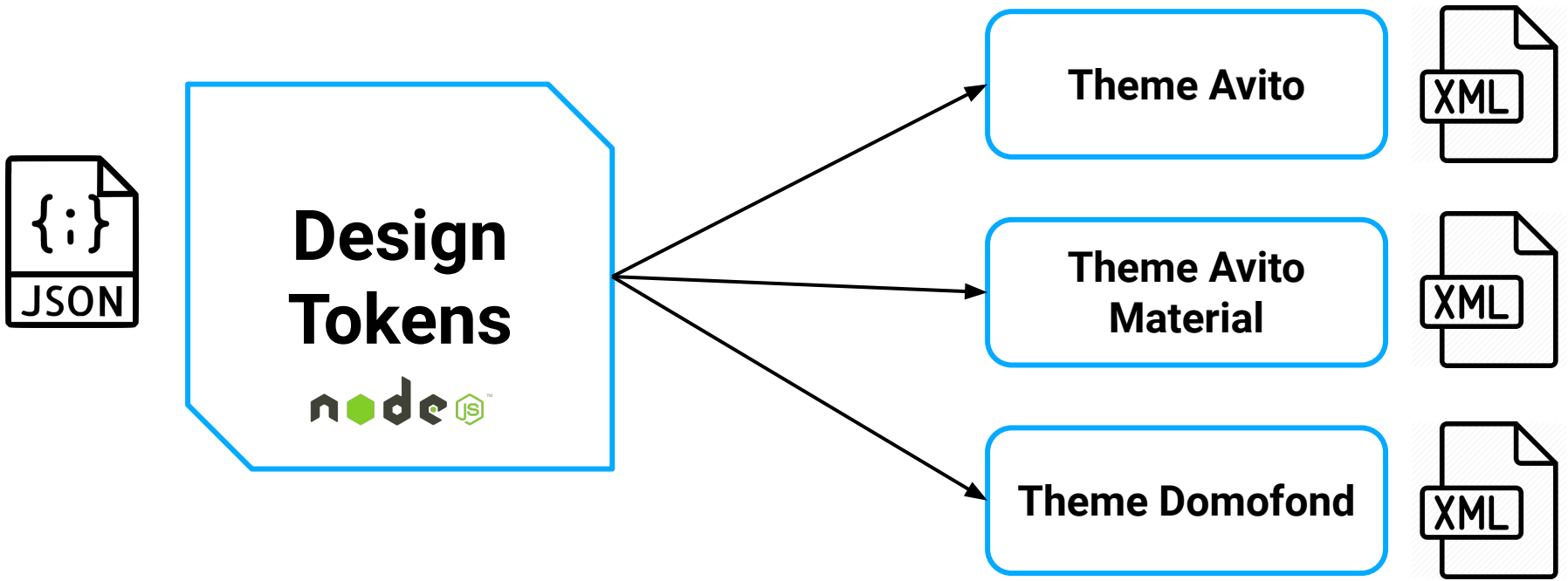
Рефакторинг

Структура системы



Avito

Структура системы



Avito

Design tokens

```
{
  "style": {
    "button": {
      "primaryLarge": {
        "value": {
          "height": "{size.height.button.large.value}",
          "backgroundColor": {
            "normal": "{color.blue.value}",
            "highlighted": "{color.blue600.value}",
            "disabled": "{color.blue300.value}",
            "ripple": "{color.blue700.value}"
          },
        ....
```


Avito

Стили

```
<style name="Avito.Button.PrimaryLarge" parent="Design.Widget.Avito.Button" >
  <item name="android:textAppearance">@style/Avito.Button.PrimaryLarge.TitleFont</item>
  <item name="android:minHeight">48dp</item>
  <item name="android:maxHeight">48dp</item>
  <item name="android:minWidth">48dp</item>
  <item name="android:textColor">@color/avito_button_primarylarge_titlecolor</item>
  <item name="button_backgroundColor">@color/avito_button_primarylarge_backgroundcolor</item>
  <item name="button_iconColor">@color/avito_button_primarylarge_iconcolor</item>
  <item name="button_cornerRadius">5dp</item>
  <item name="button_spinnerStyle">@style/Avito.Button.PrimaryLarge.Spinner</item>
</style>
```

Плюсы

- Идеальная управляемость
- Весь бойлерплейт генерируется

Минусы

- Никто кроме дизайн-системы не понимает до конца как это работает и как вносить изменения
- Требуются серьезные доработки для автоматизации процесса

Рефакторинг

- Поддержка нескольких тем, путем использования атрибутов везде, где возможно
- Вынос хардкода из проекта
- Поддержка темной темы в случае невозможности использования атрибутов

А вы используете атрибуты в проекте?)

Атрибуты

Avito

Атрибуты

```
<item name="gray2">@color/avito_gray_2</item>
<item name="gray4">@color/avito_gray_4</item>
<item name="gray8">@color/avito_gray_8</item>
<item name="gray12">@color/avito_gray_12</item>
<item name="gray28">@color/avito_gray_28</item>
<item name="gray44">@color/avito_gray_44</item>
<item name="gray48">@color/avito_gray_48</item>
<item name="gray60">@color/avito_gray_60</item>
<item name="gray68">@color/avito_gray_68</item>
<item name="gray76">@color/avito_gray_76</item>
<item name="gray84">@color/avito_gray_84</item>
```

Атрибуты

```
fun Context.getColorByAttr(@AttrRes colorAttr: Int): Int {
    val typedArray = obtainStyledAttributes(intArrayOf(colorAttr))
    if (!typedArray.hasValue(0)) {
        val resName = resources.getResourceName(colorAttr)
        throw Resources.NotFoundException("Resource with name = $resName is not defined")
    }
    val result = typedArray.getColor(0, 0)
    typedArray.recycle()
    return result
}
```

Атрибуты

```
context.getColorByAttr(design_R.attr.gray28)
```

Атрибуты

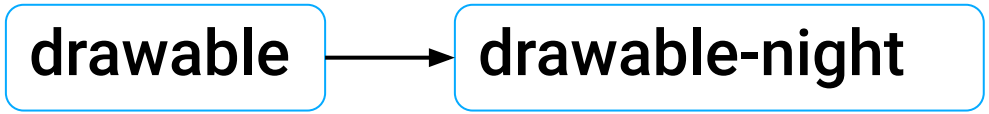
```
<TextView
    android:id="@+id/block_title"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textColor="?black" />
```

Минусы

- Многим попросту непривычно
- Если не заложиться на атрибуты в начале - рефакторить долго
- Не работает до 24 API в vector drawable с fillType или хоть одним <gradient>

Поддержка темной темы

По сути просто дублируем все ресурсы в -night папки и заменяем значения для темной темы



Переключением занимается сам андроид

Как сохранить результат

Avito

Защита от поломок

- Unit тесты
- UI тесты
- Lint rules
- Скриншотные тесты

LINT

Avito

Линт

- Запрет хардкода в цветов в XML

(например @android:color/white или #ffffff)

- Запрет хардкода в коде

(например Color.WHITE или Color.parseColor("#519c3f"))

Пример линта

```
override fun visitQualifiedReferenceExpression(node: UQualifiedReferenceExpression) {
    if (context.file.path.containsExcludes()) return
    val nodeString = node.asSourceString()
    if (nodeString.startsWithValues("Color.", "android.graphics.Color.")) {
        report(node)
    }

    if (nodeString.contains("android.R.color")) {
        report(node)
    }
```

Плюсы

- Дешево в разработке
- Гибкий инструмент
- Работает как локально, так и на CI

Минусы

- На большом проекте проходит нереально долго и сильно жрет ресурсы
- Практически невозможно прогонять только конкретный линт

СКРИНШОТНЫЕ ТЕСТЫ

Avito

Avito Бинго 2020

CI	Machine Learning	UI тесты	PAAS
Kafka	Многомодульность	Микросервисы	Монолит
Server Driven UI	OKR	CD	Масштабирование
Инфраструктура	Мониторинг	Метрики	Архитектура

Avito

Скриншотные тесты

- Обычно покрывают компоненты, а не экраны
- Скриншоты сами по себе pixel-perfect
- Требуют эмулятор и по сути являются инструментальными тестами
- Ожидаемо плохо работают с анимациями

Скриншотные тесты

https://github.com/facebook/screenshot-tests-for-android

Скриншотные тесты

Плюсы:

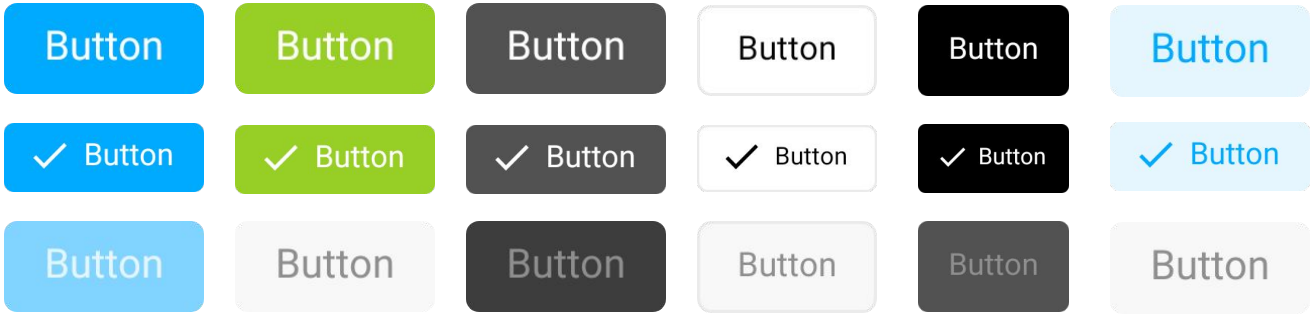
- Удобный синтаксис
- Локально все работает из коробки
- Умеет работать с несколькими девайсами
- Делает не только скриншоты, но и репорты

Пример теста

```
@Test fun buttonTest() {
    val targetContext: Context = InstrumentationRegistry.getInstrumentation().targetContext
    val context = ContextThemeWrapper(targetContext, R.style.Theme_DesignApp_Avito)
    val button = Button(context, null, R.attr.buttonPrimaryMedium)
    button.layoutParams = FrameLayout.LayoutParams(
        FrameLayout.LayoutParams.WRAP_CONTENT,
        FrameLayout.LayoutParams.WRAP_CONTENT,
        FrameLayout.LayoutParams.UNSPECIFIED_GRAVITY
    )
    button.setText("Button")
    ViewHelpers.setupView(button).layout()
    Screenshot.snap(button).setName("primaryButtonMedium").record()
}
```

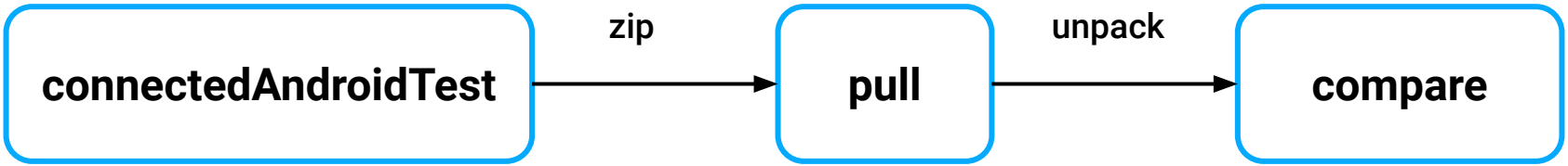
Пример теста

Пример теста



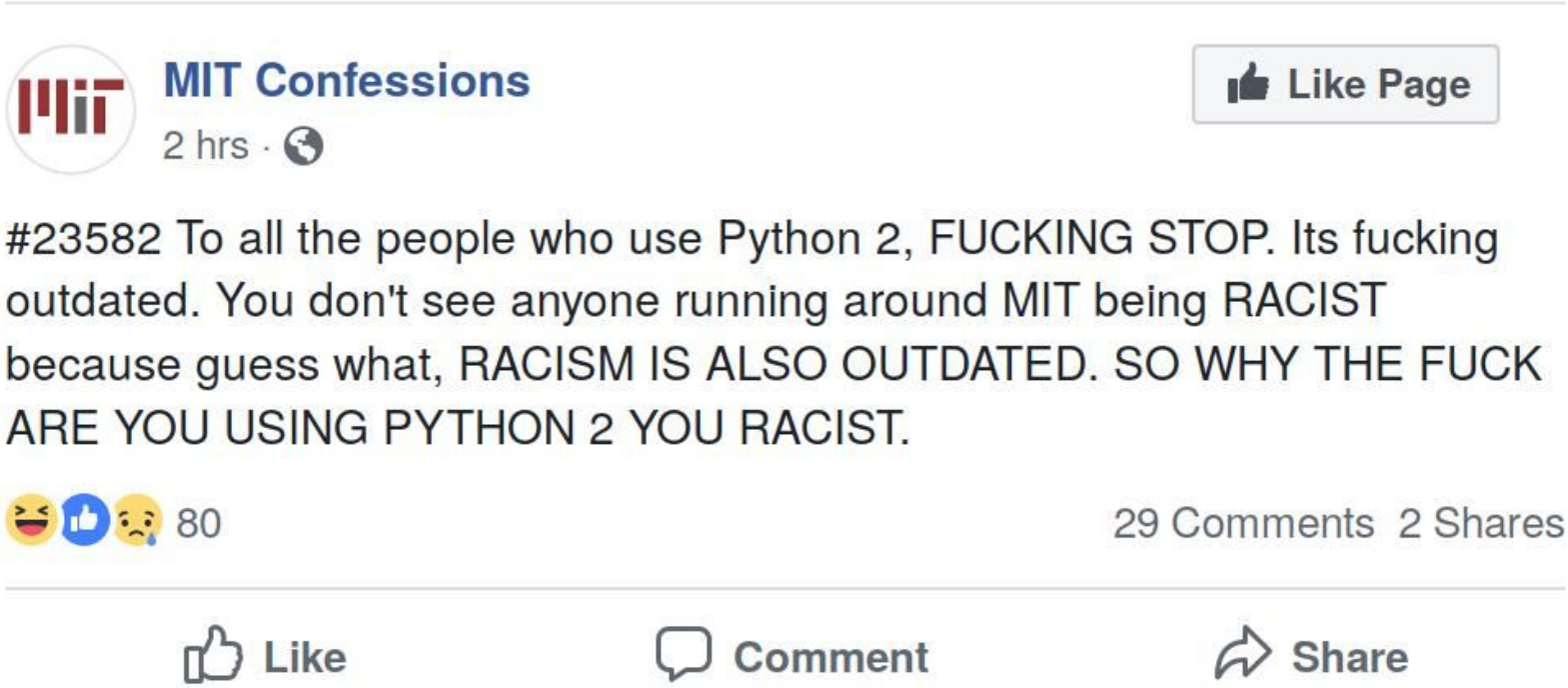
Avito

Как работает



Эталоны лежат в папке с проектом

Требует Python 2.7

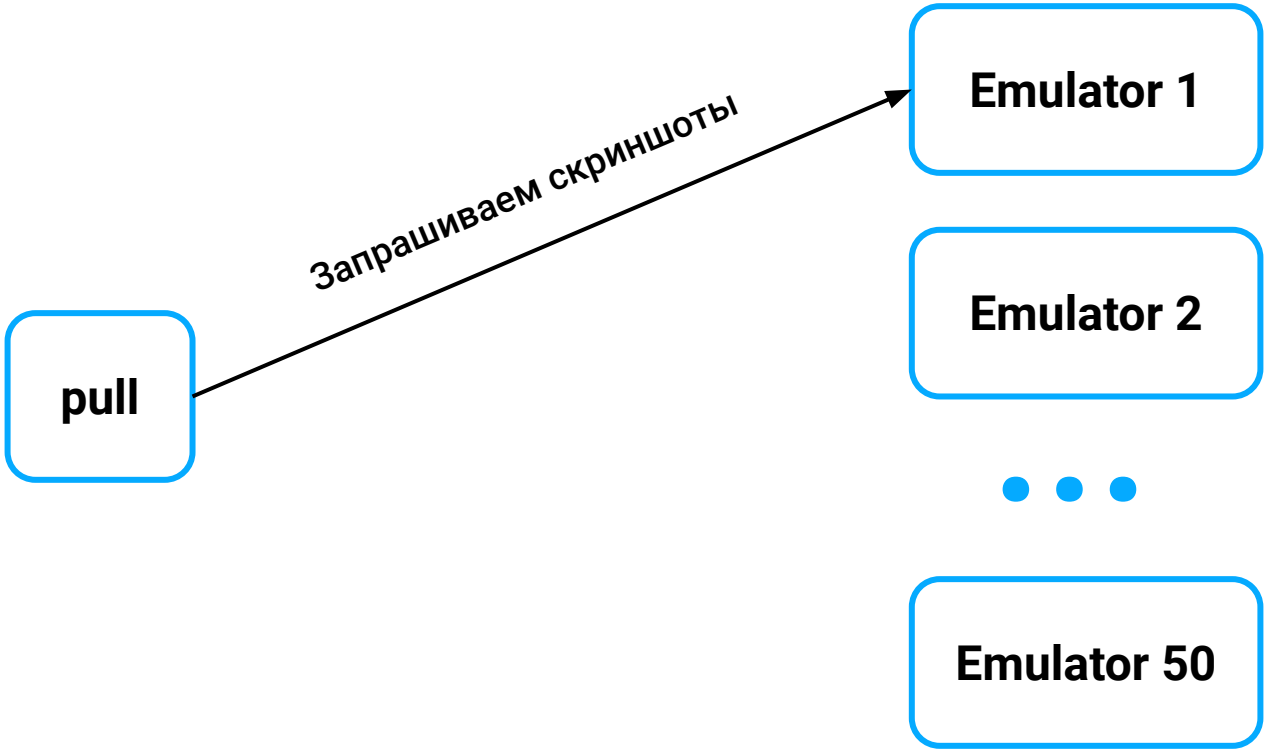


Avito

Работа с девайсами

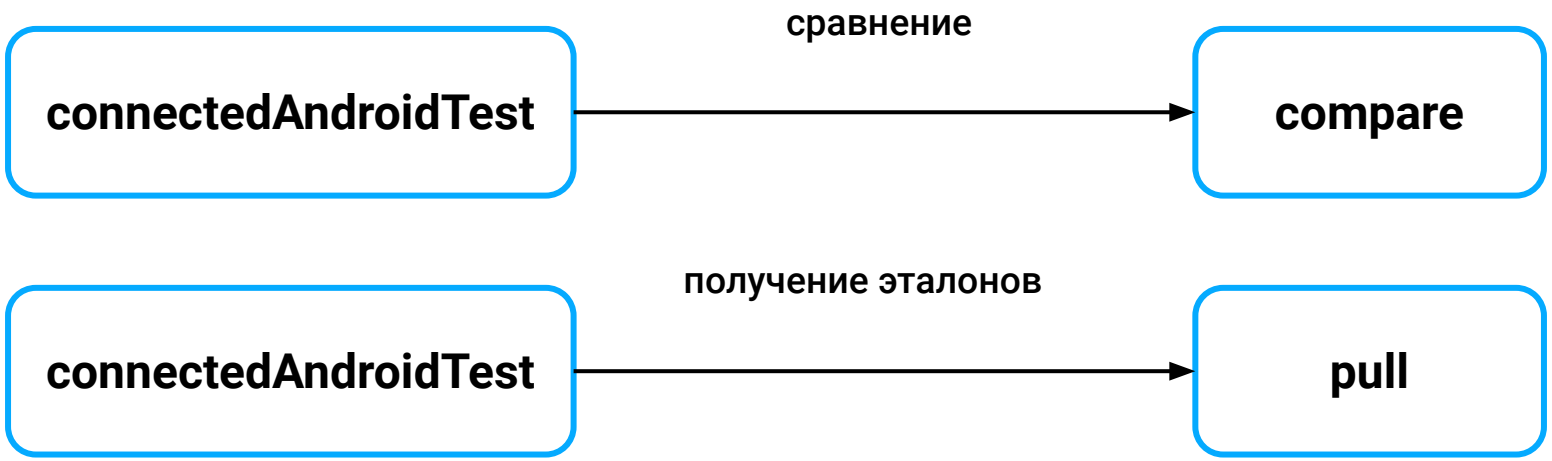
- Использует ADB
- Требует явного серийника для идентификации девайса

На CI все плохо :/



Avito

Как в итоге сделано



Эталоны лежат в androidTest/assets
Скриншоты - только на самом девайсе

Avito

Как сравниваются скриншоты

```
val referenceBitmap = getBitmapFromAsset(context, filePath)

val currentBitmap = getBitmapFromDevice(targetContext, filePath)

Assert.assertTrue(

        "$filePath does not equals reference",

        referenceBitmap?.sameAs(currentBitmap) ?: false

)
```

Что получили

- Любые изменения внешнего вида компонентов сразу видны на PR
- Механизм работает в mvp, но требует значительных доработок, например использование эмулятора на CI напрямую, внятный репортинг и так далее
- Возможно, придется решать проблем с git-lfs

ИТОГИ

Avito

Итоги по дизайн системе

- Главные цели дизайн системы - управляемость и консистентность
- Дизайн система это не только библиотека компонентов, то также сопутствующая инфраструктура и инструменты
- Организационно ничего не выйдет без инициативы со стороны дизайнера

Что можно из доклада вынести?

- Библиотека компонентов умрет без владельцев кода
- Нужно стараться использовать атрибуты с самого начала
- Линт решает проблемы быстро, но слишком требователен к ресурсам
- Скриншотные тесты сильно упростят жизнь, но существующие решения вряд ли хорошо отработают на CI

ВОПРОСЫ?

Avito