VK Звонки: соединяем тысячи людей с Android и iOS по WebRTC

Никита Разумный и Иван Шафран

Звонки нужны всем

Общение

Звонки нужны всем

Общение

Учёба

Звонки нужны всем

Общение

Учёба

Приватность
(такси, объявления)

Звонки нужны всем

Общение

Учёба

Приватность
(такси, объявления)

Телемедицина

VK Звонки

5м

Звонков в день

VK Звонки

5м

Звонков в день

20м

Пользователей звонков в месяц

VK Звонки

5м

Звонков в день

20М

Пользователей звонков в месяц

90%

Звонков с мобильных устройств

План

1. 1 на 1 звонки

План

1. 1 на 1 звонки
2. Групповые звонки

1 на 1 звонки

Что узнаем

1. Аудио

Что узнаем

1. Аудио
2. Видео

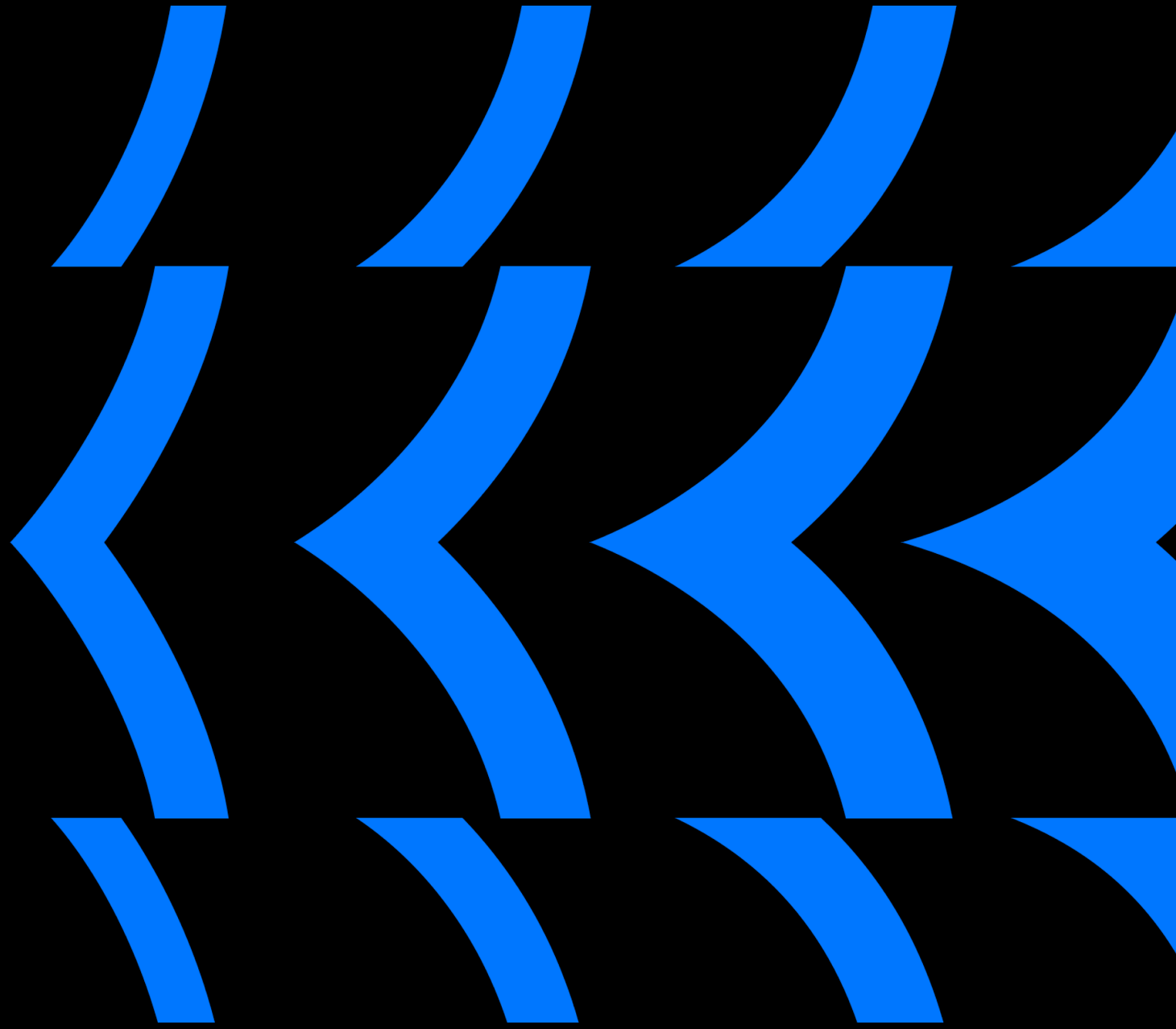
Что узнаем

1. Аудио
2. Видео
3. AR-эффекты

Что узнаем

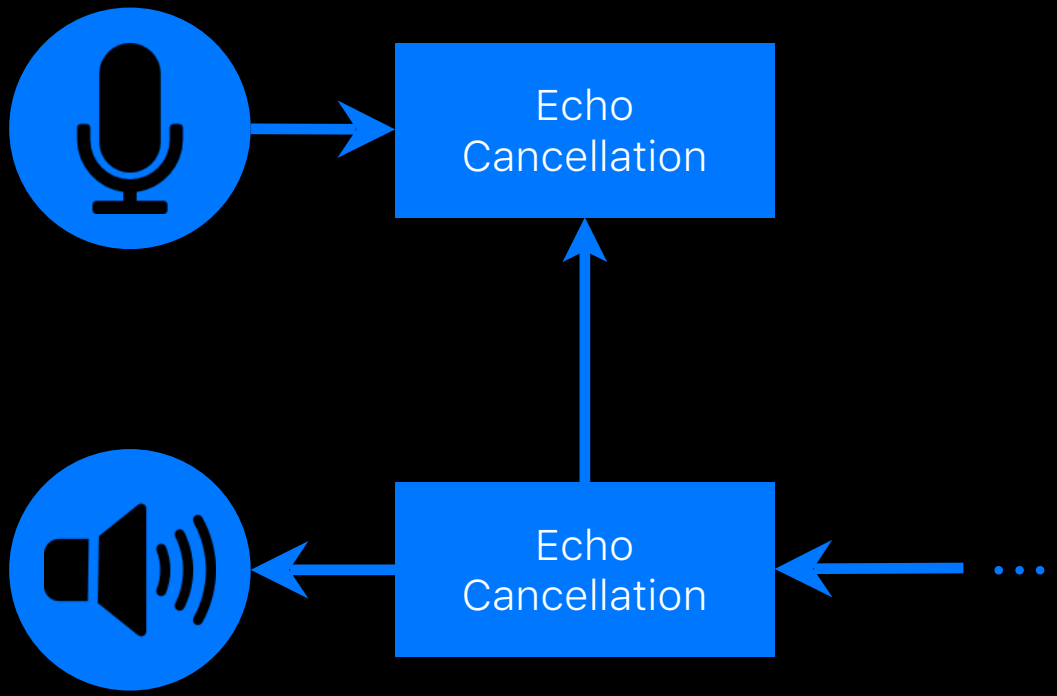
1. Аудио
2. Видео
3. AR-эффекты
4. Системные API

Аудио

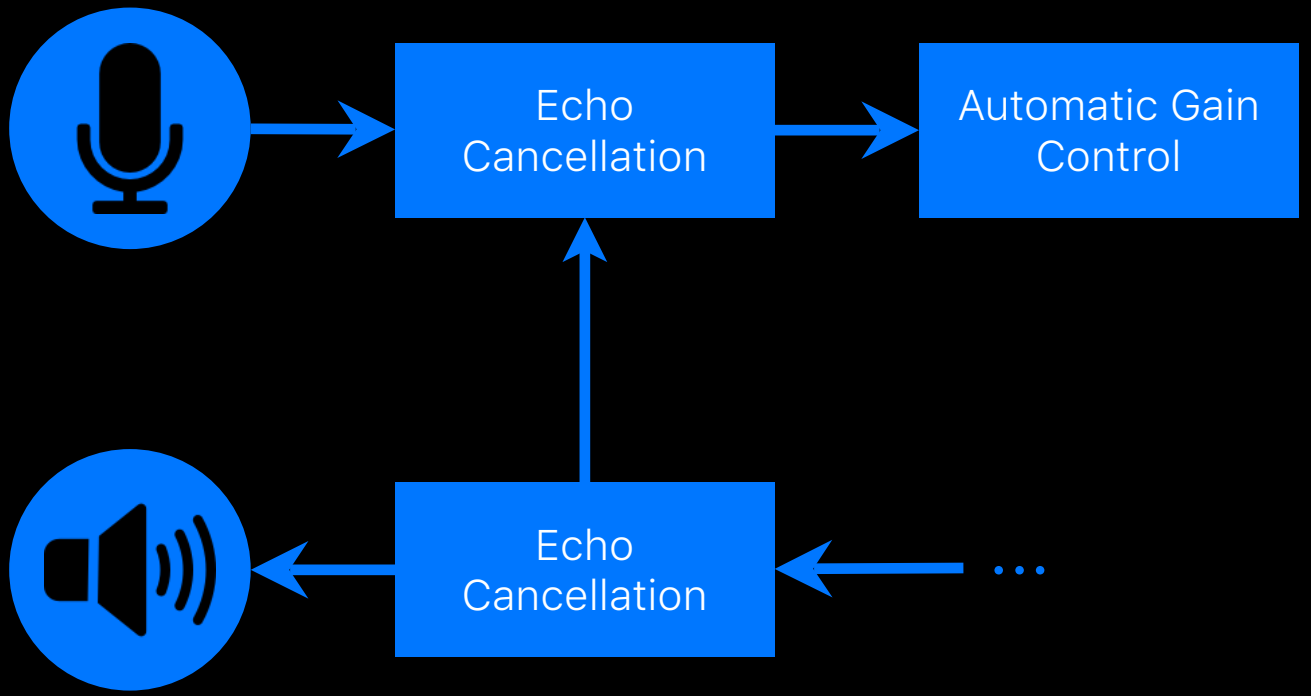


Аудио состоит из нескольких этапов

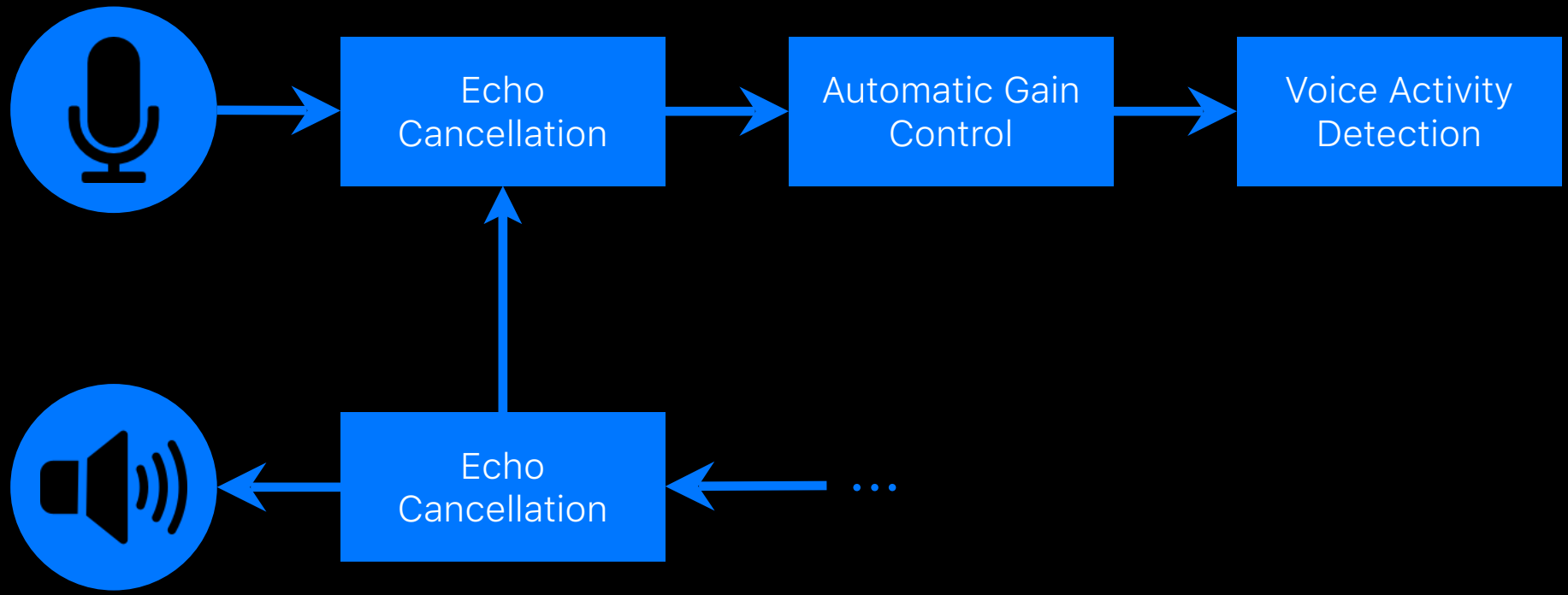
Убираем эхо, чтобы собеседник не слышал себя



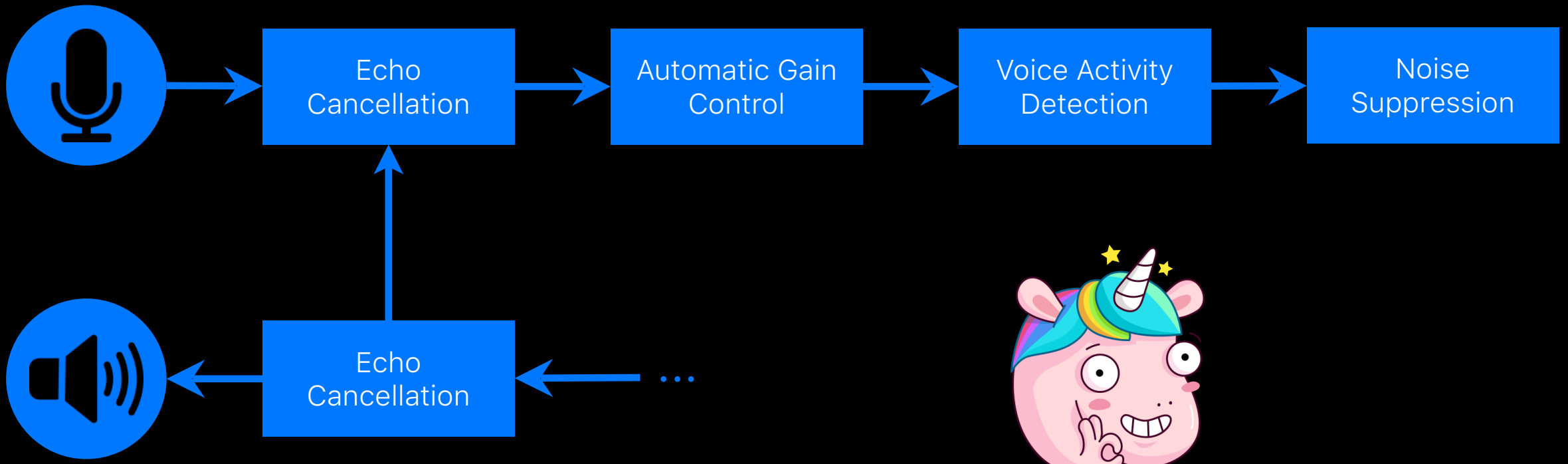
Выравниваем громкость аудио, чтобы слышать всех одинаково



Определяем на аудио голос. Если тишина, то пакеты аудио можно не отправлять в сеть



Убираем шумы, соседей, пылесос на фоне



WebRTC содержит простые реализации всех этапов
Но можно лучше

Noise Suppression

Voice Activity Detection

Современные нейронки выигрывают у традиционных методов

- Noise Suppression
- Voice Activity Detection

Про устройство сетки https://habr.com/ru/company/vk/blog/572950/

Что мы получили от доработки аудио

60%

Уменьшение числа ускорений звука
с искажениями

Что мы получили от доработки аудио

60%

Уменьшение числа ускорений звука
с искажениями

30%

Экономия аудио трафика сети

Что мы получили от доработки аудио

60%

Уменьшение числа ускорений звука с искажениями

30%

Экономия аудио трафика сети

40%

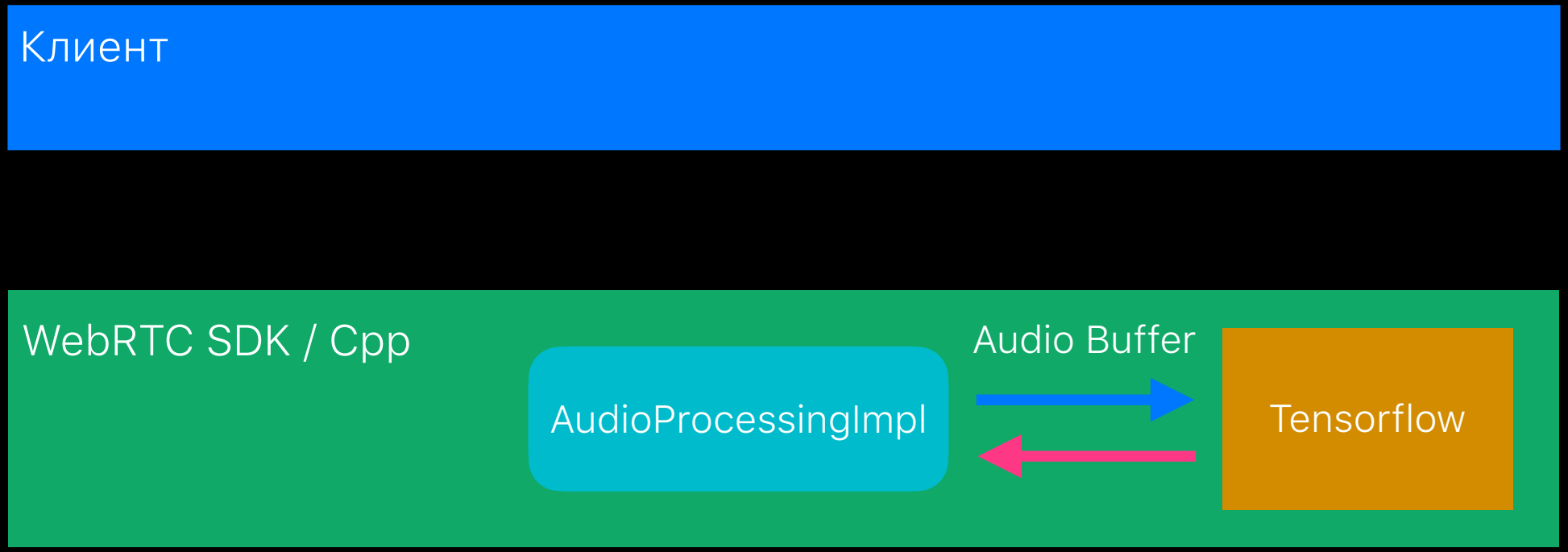
Улучшение качества звука по метрике PESQ

Подробнее про аудио пайплайн: https://habr.com/ru/company/vk/blog/574604/

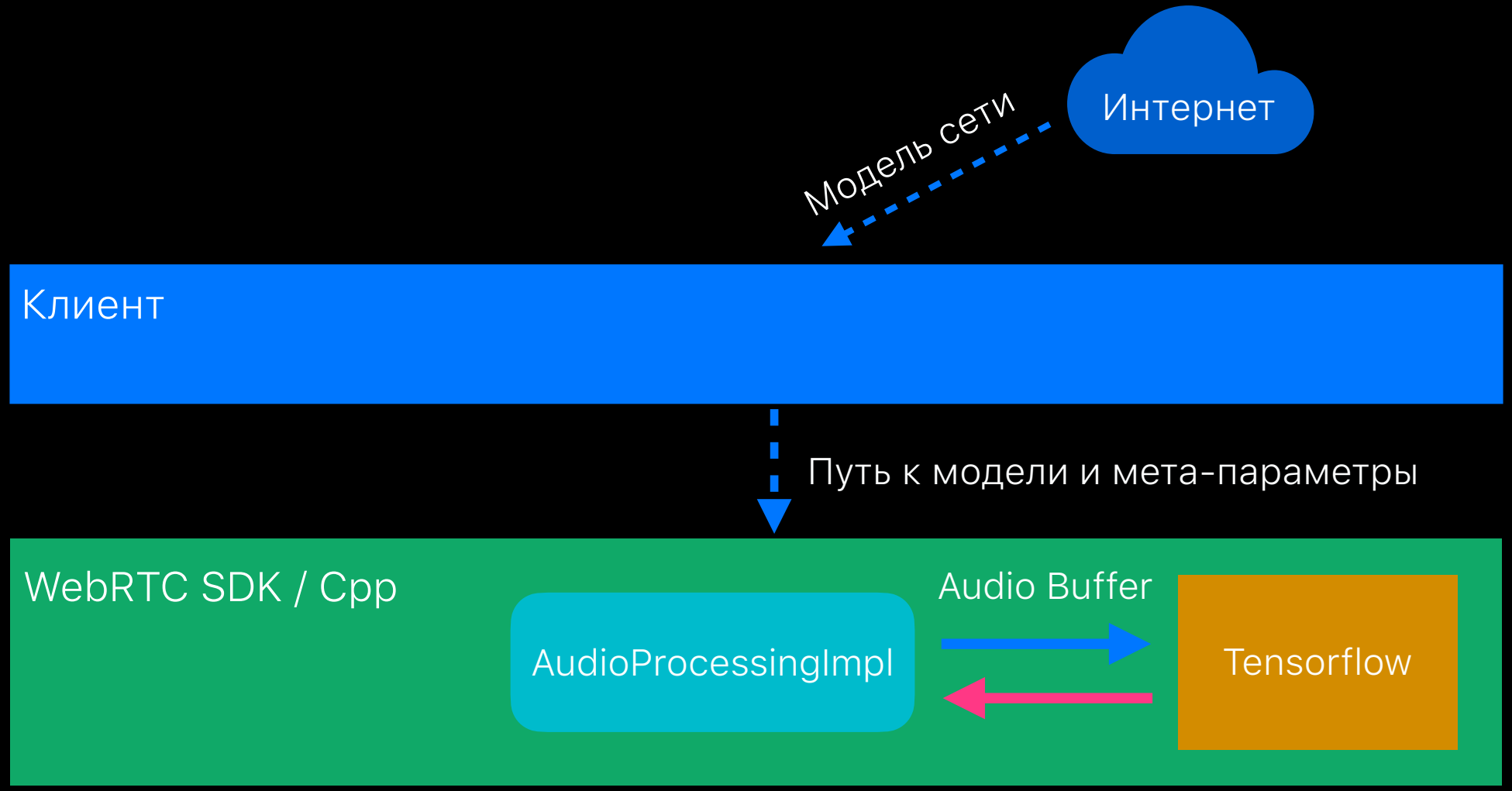
Для каждой платформы выбираем наиболее удобный формат

Tensorflow подключаем в нативном коде

Аудио поток обрабатываем в AudioProcessingImpl



ML модель загружаем из интернета



Встраивание CoreML-моделей в WebRTC

- Инжектим инференс в AudioProcessingImpl через RTCPeerConnectionFactory

Встраивание CoreML-моделей в WebRTC

- Инжектим инференс в AudioProcessingImpl через RTCPeerConnectionFactory
- WebRTC под iOS использует только системный voice processing

Встраивание CoreML-моделей в WebRTC

- Инжектим инференс в AudioProcessingImpl через RTCPeerConnectionFactory
- WebRTC под iOS использует только системный voice processing
- Смотрим на конфигурацию AudioUnit-а
  - kAUVoiceIOProperty_BypassVoiceProcessing
  - kAUVoiceIOProperty_VoiceProcessingEnableAGC

Встраивание CoreML-моделей в WebRTC

- Профилируем, избегаем рантайма objc/swift

Встраивание CoreML-моделей в WebRTC

- Профилируем, избегаем рантайма objc/swift
- AudioProcessingImpl работает на потоке, критичным к блокировкам

Встраивание CoreML-моделей в WebRTC

- Профилируем, избегаем рантайма objc/swift
- AudioProcessingImpl работает на потоке, критичным к блокировкам
- Переносим обработку звука в audio transport (AsyncAudioProcessing)

Встраивание CoreML-моделей в WebRTC

- Профилируем, избегаем рантайма objc/swift
- AudioProcessingImpl работает на потоке, критичным к блокировкам
- Переносим обработку звука в audio transport (AsyncAudioProcessing)
- Контролируем задержку

Если обработка аудио работает долго,
то делаем fallback на WebRTC шумодав

Аудио: резюме

- Улучшили качество звука
- Снизили нагрузку на сервер
- Унифицировали пользовательский опыт на разных платформах

Видео

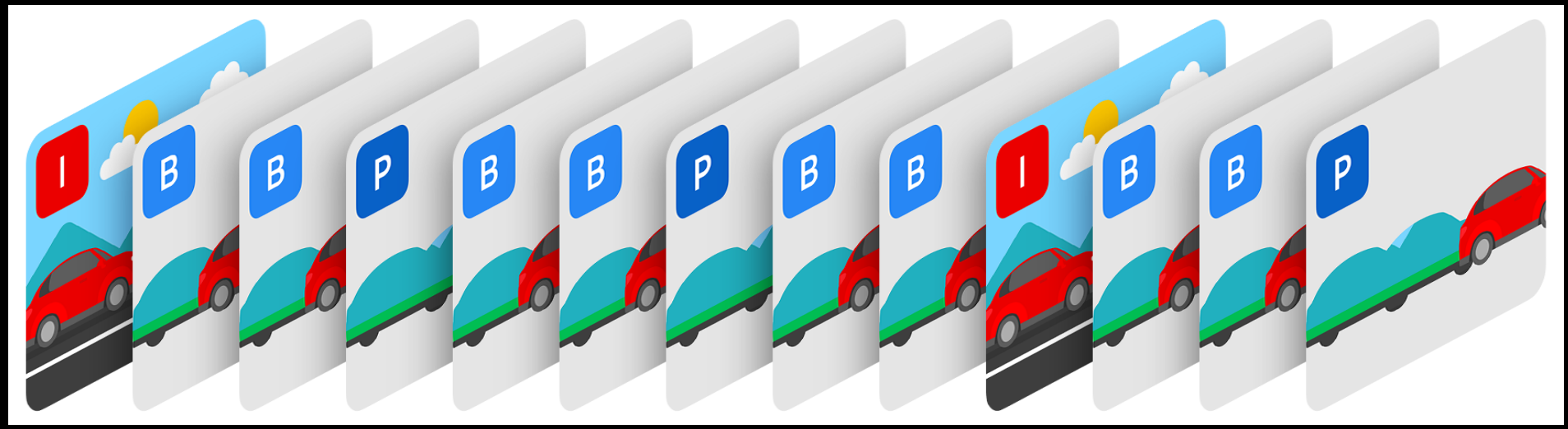
Проблемы видео в p2p звонке

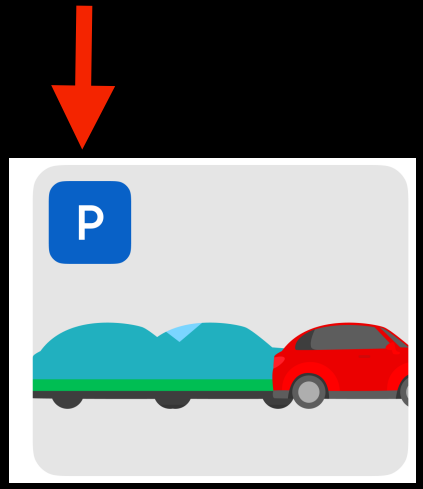
- Батарейка
- Нагрузка сети

Проблемы видео в p2p звонке

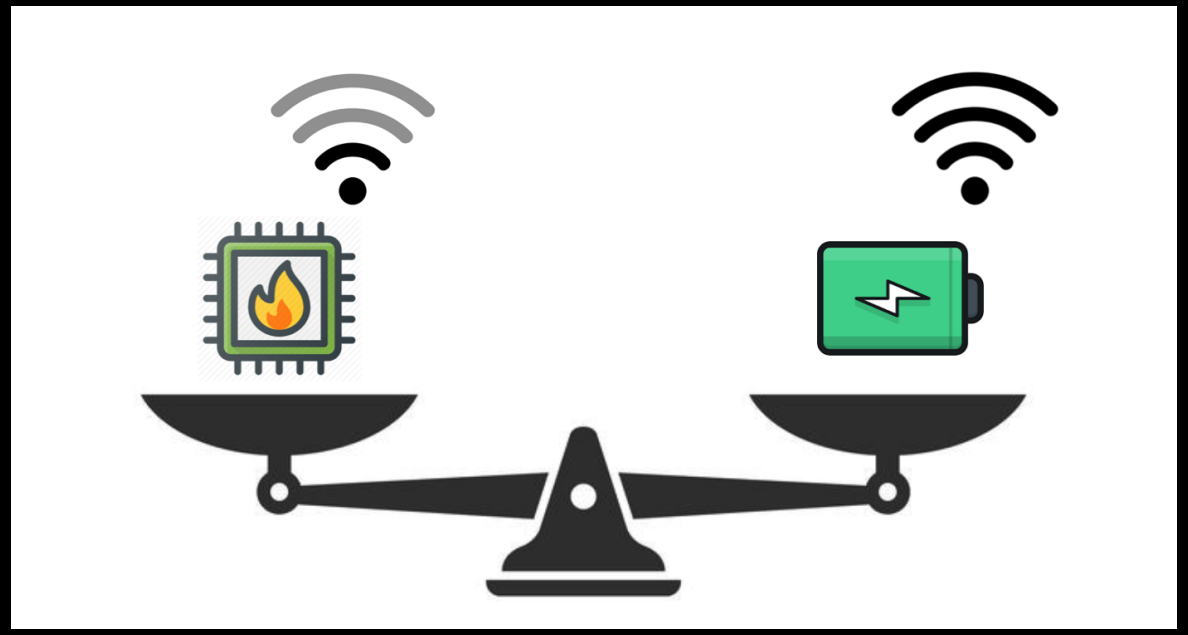
- Батарейка
- Нагрузка сети
- Особенности скриншаринга

Видеокодек

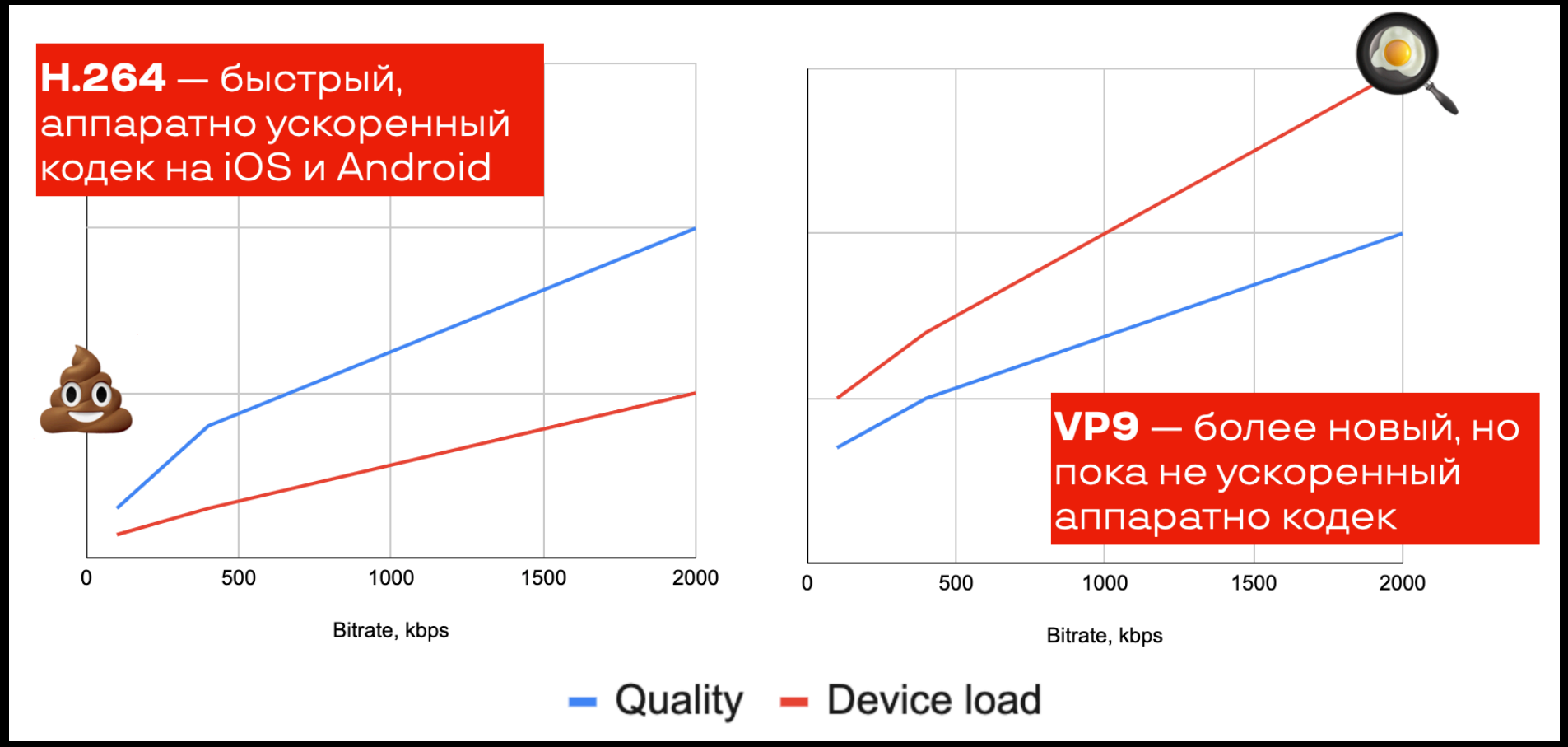




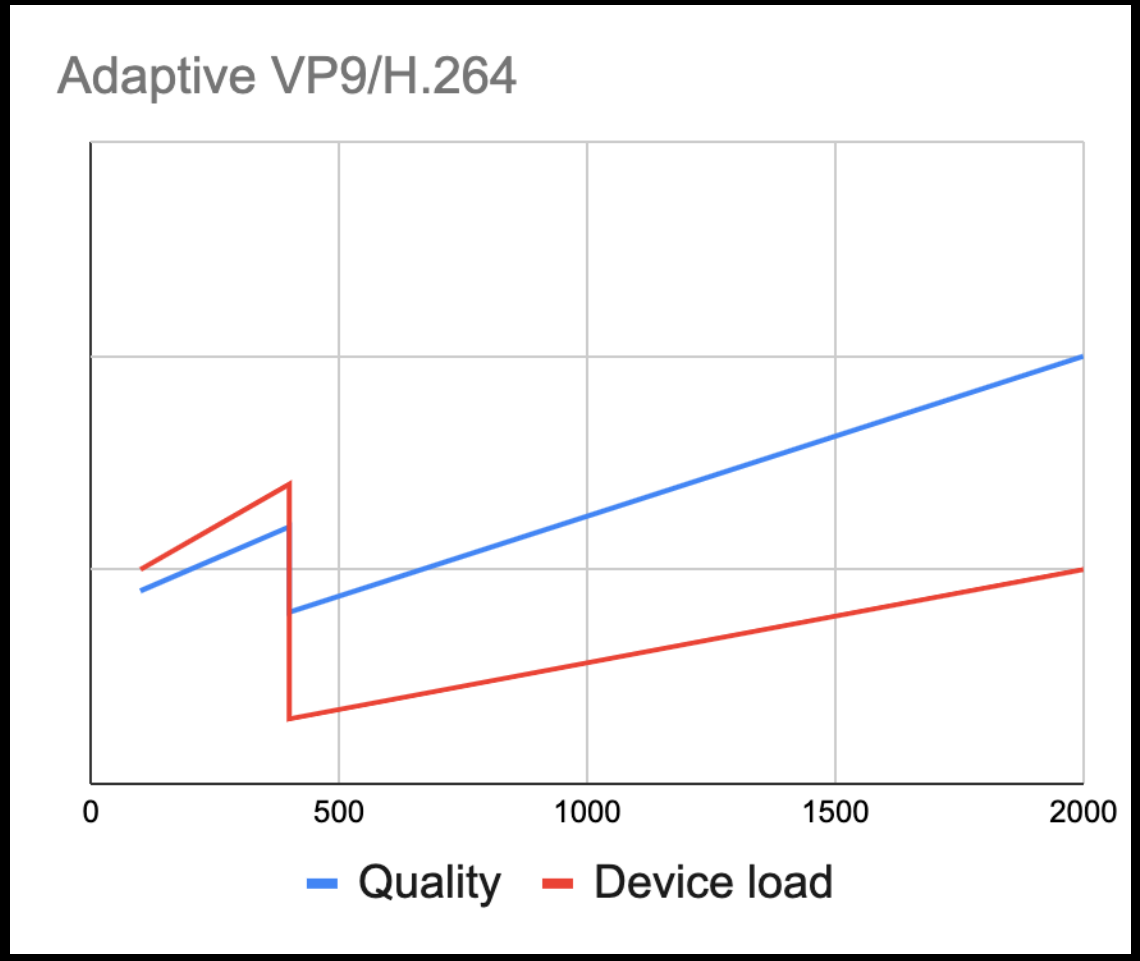
Видеокодек



Видеокодек


H.264 — быстрый, аппаратно ускоренный кодек на iOS и Android
0
500
1000
1500
2000
Bitrate, kbps
VP9 — более новый, но пока не ускоренный аппаратно кодек
0
500
1000
1500
2000
Bitrate, kbps
Quality
Device load

Видеокодек



Демонстрация экрана

- Чем отличается от видео с камеры?

Демонстрация экрана

- Чем отличается от видео с камеры?
- Акцент на читаемости текста

Демонстрация экрана

```
typedef NS_ENUM(NSInteger, RTCDegradationPreference) {
  RTCDegradationPreferenceDisabled,
  RTCDegradationPreferenceMaintainFramerate,
  RTCDegradationPreferenceMaintainResolution,
  RTCDegradationPreferenceBalanced
};
```

Демонстрация экрана

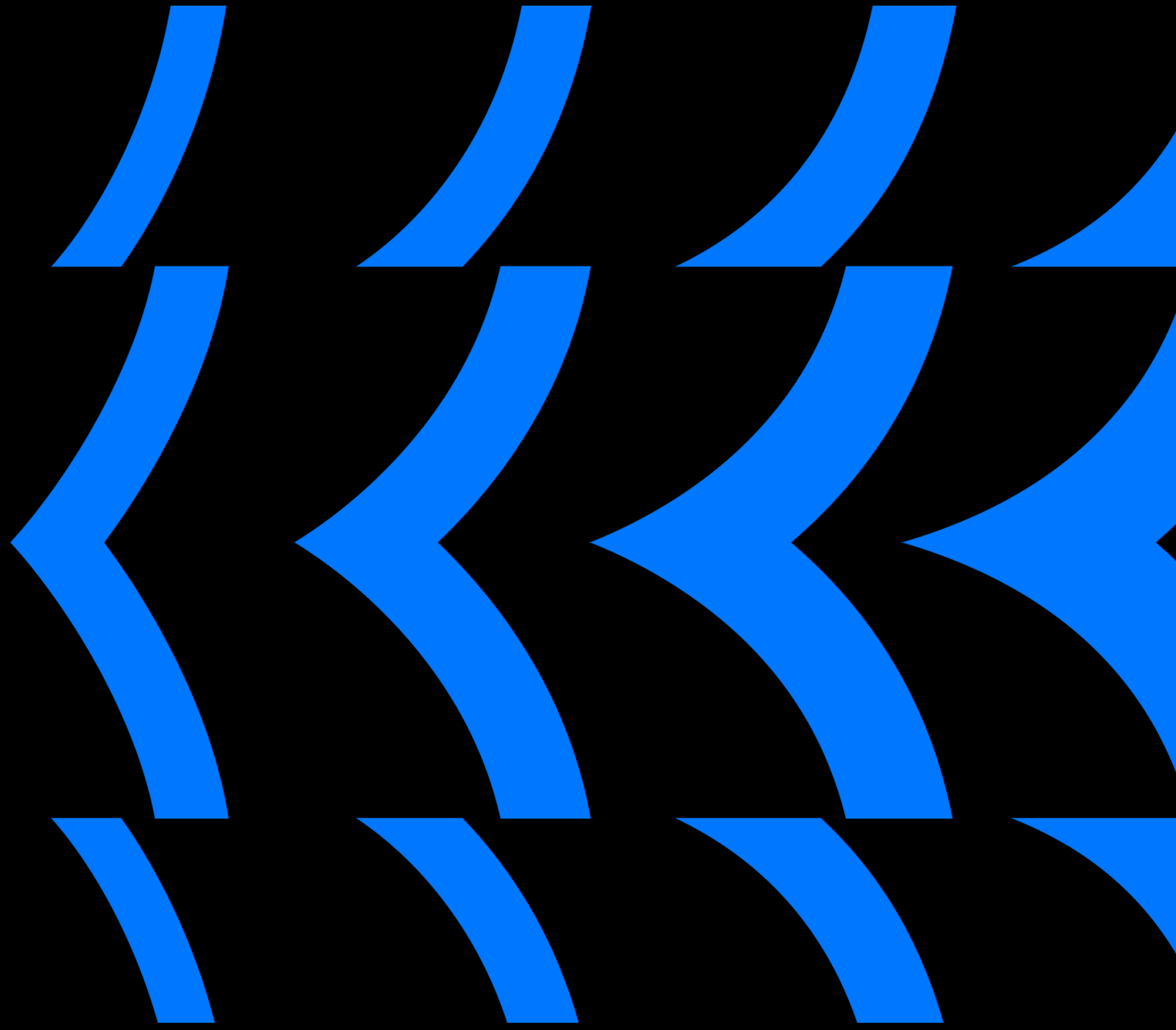
```
typedef NS_ENUM(NSInteger, RTCDegradationPreference) {
  RTCDegradationPreferenceDisabled,
  RTCDegradationPreferenceMaintainFramerate,
  RTCDegradationPreferenceMaintainResolution,
  RTCDegradationPreferenceBalanced
};
```

```
typedef NS_ENUM(NSInteger, RTCVideoTrackContentHint) {
    RTCVideoTrackContentHintNone,
    RTCVideoTrackContentHintFluid,
    RTCVideoTrackContentHintDetailed,
    RTCVideoTrackContentHintText
};
```

Видео

- Адаптация кодека под условия сети
- Отдельная конфигурация треков под демонстрацию экрана

AR-эффекты



AR-эффекты

- Маски

AR-эффекты

- Маски
- Виртуальные фоны

AR-эффекты

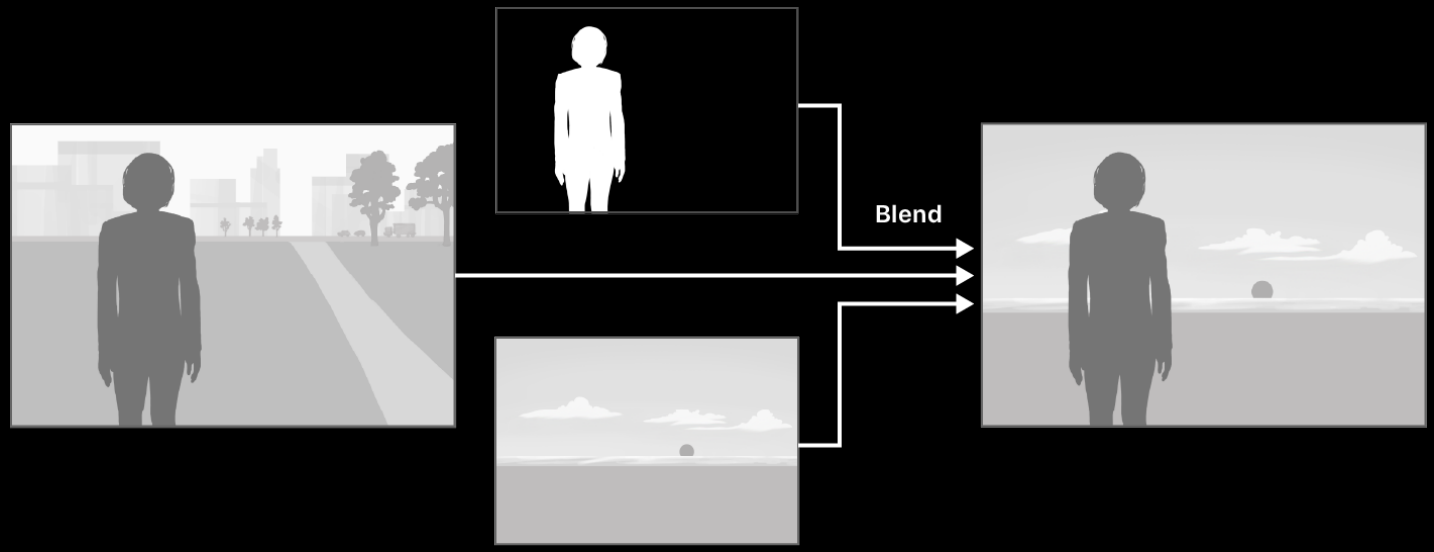
- Маски
- Виртуальные фоны
- Бьюти-фильтр

AR-эффекты

- Маски
- Виртуальные фоны
- Бьюти-фильтр
- Сезонные активности

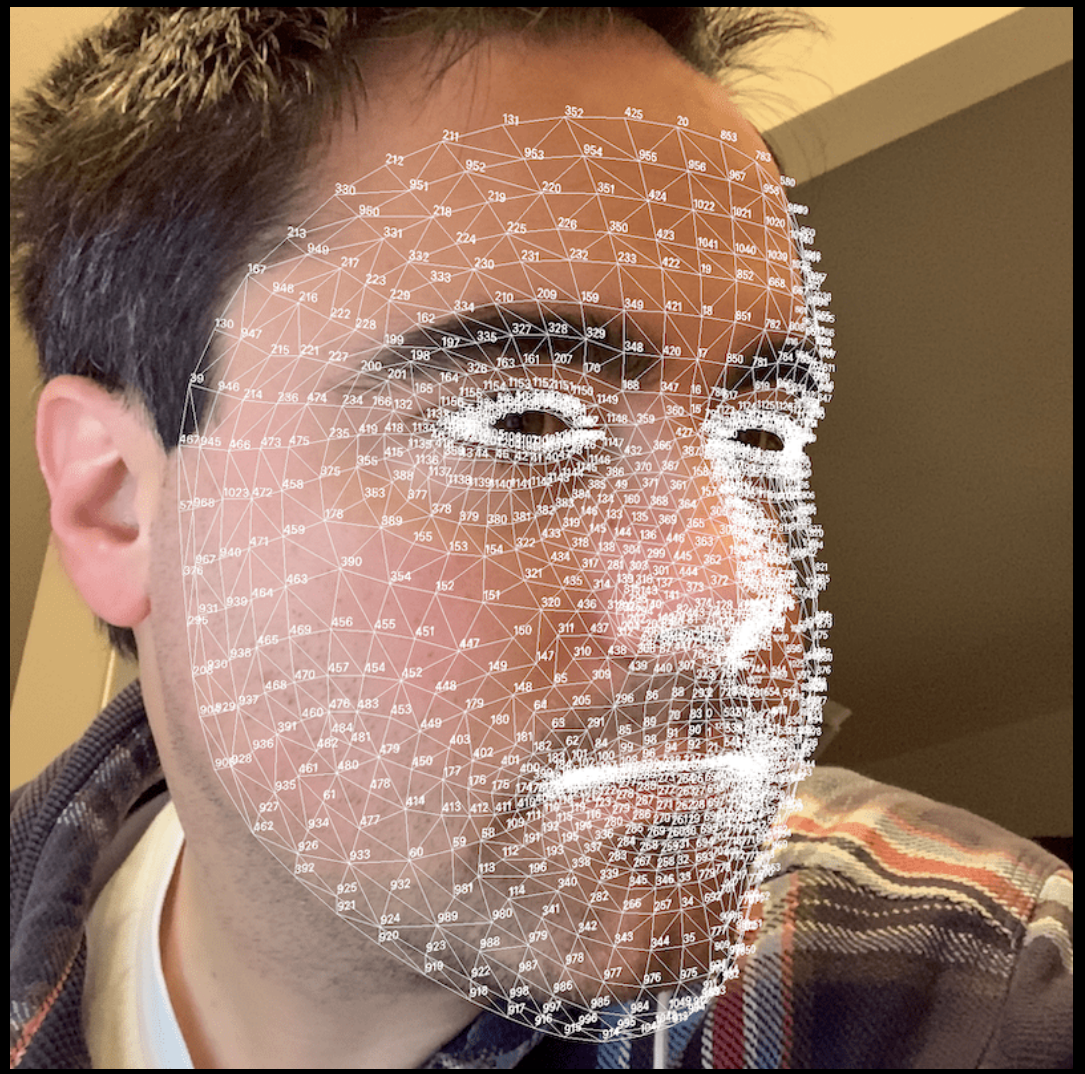
AR-эффекты: iOS

- Vision



AR-эффекты: iOS

- Vision
- ARKit



AR-эффекты: iOS

- Vision
- ARKit
- Metal

AR-эффекты: iOS

- https://developer.apple.com/documentation/arkit/content_anchors/tracking_and_visualizing_faces
- https://developer.apple.com/documentation/vision/applying_matte_effects_to_people_in_images_and_video

AR-эффекты: iOS

- CVPixelBufferPool

AR-эффекты: iOS

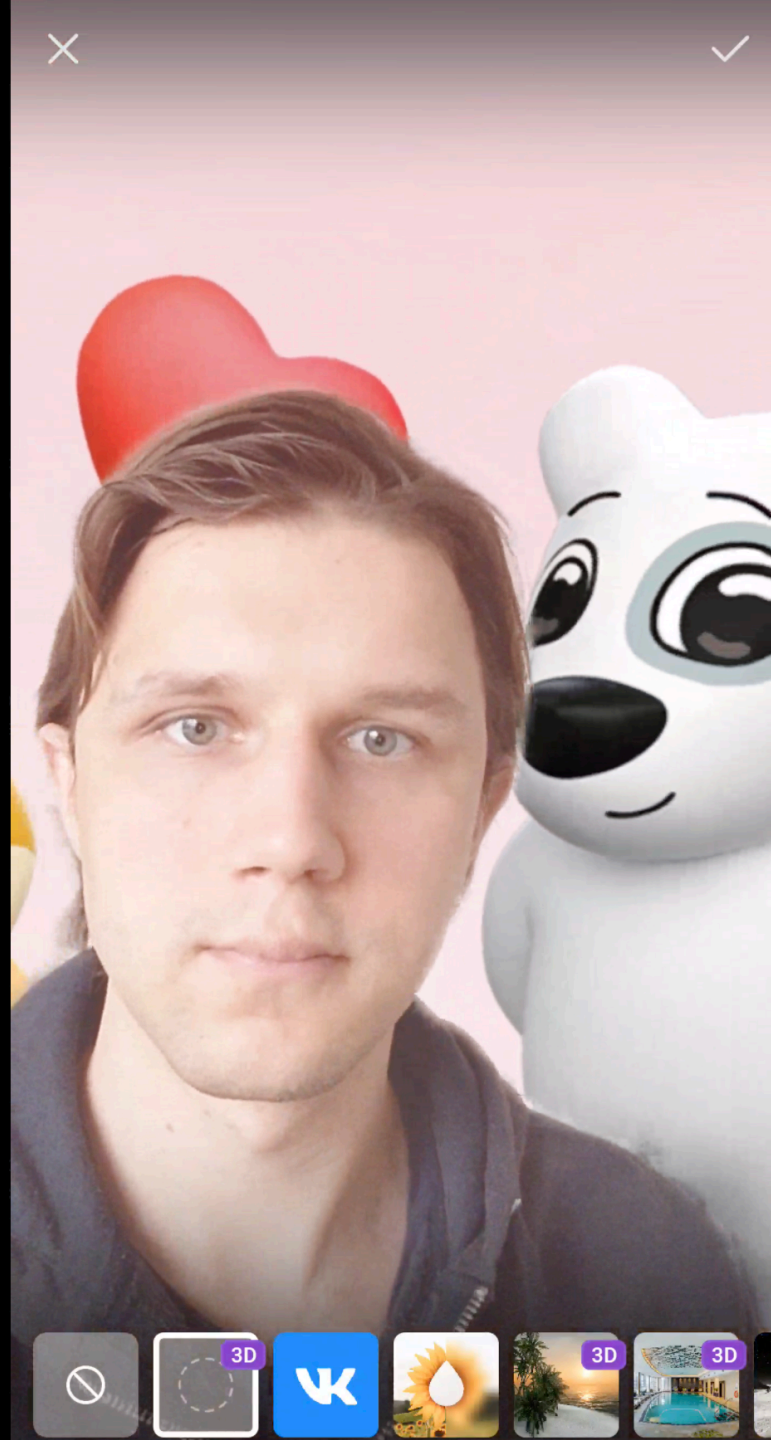
- CVPixelBufferPool
- Async Pipeline

AR-эффекты: iOS

- CVPixelBufferPool
- Async Pipeline
- yuv

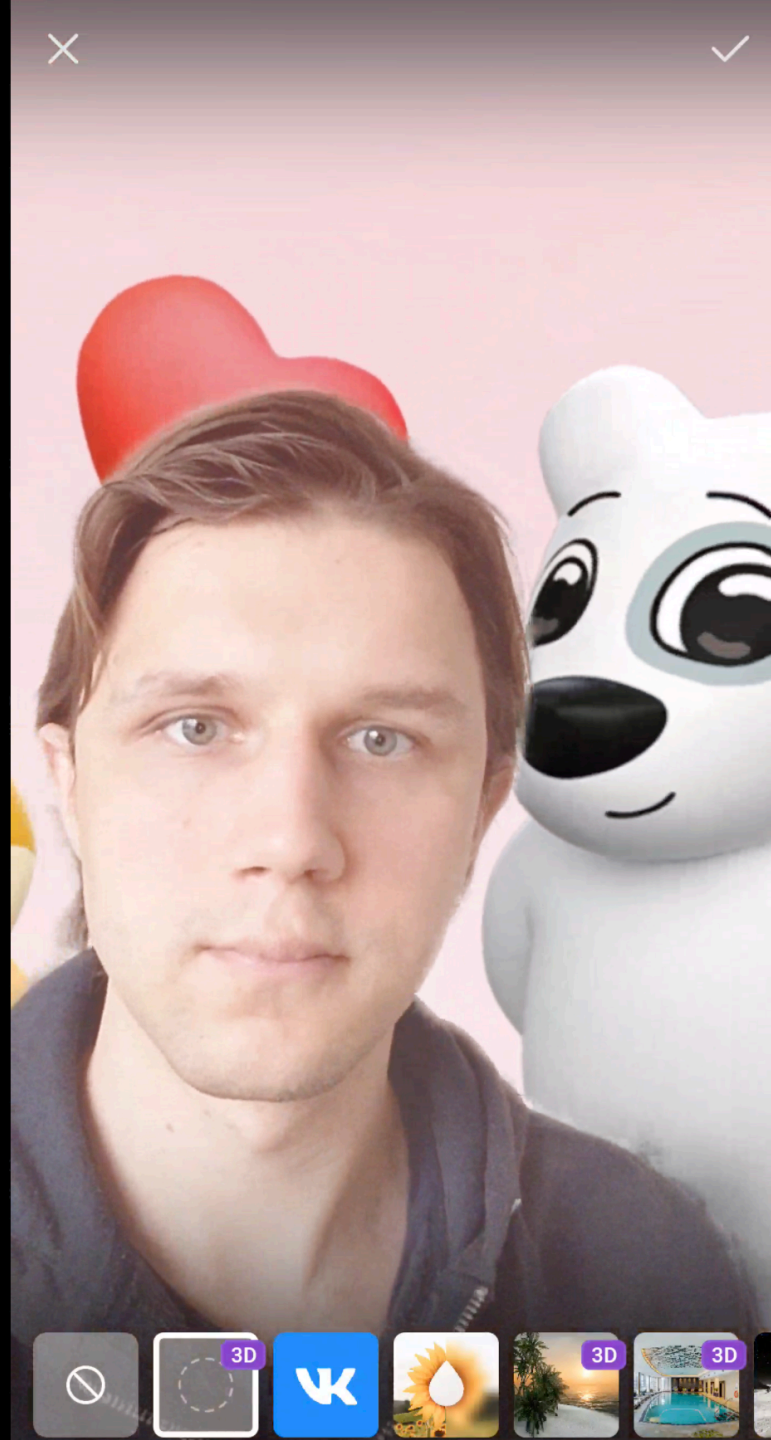
Перехватываем кадры в Android

1. Переопределяем org.webrtc.VideoCapturer



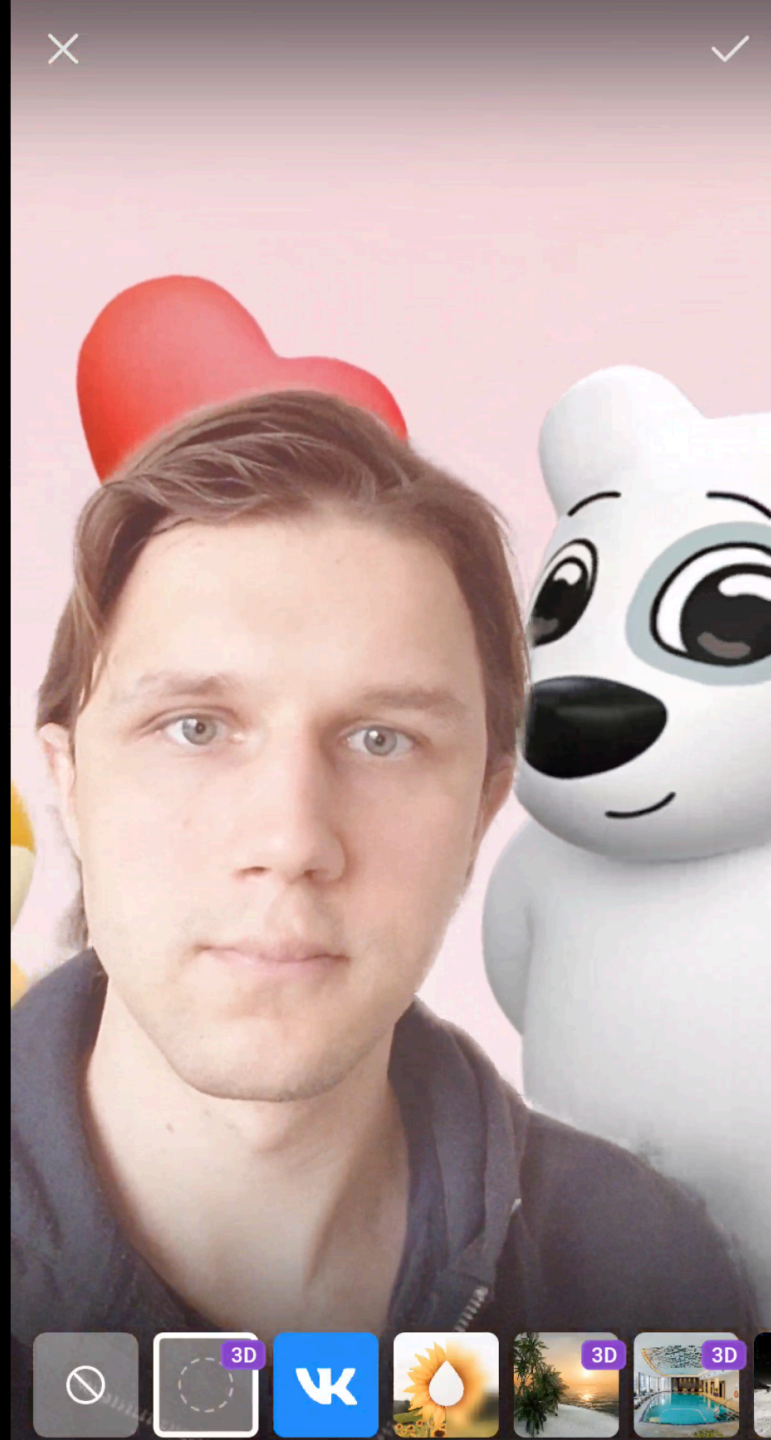
Перехватываем кадры в Android

1. Переопределяем org.webrtc.VideoCapturer
2. В него передаём org.webrtc.CapturerObserver,
   где слушаем start/stop/onFrame



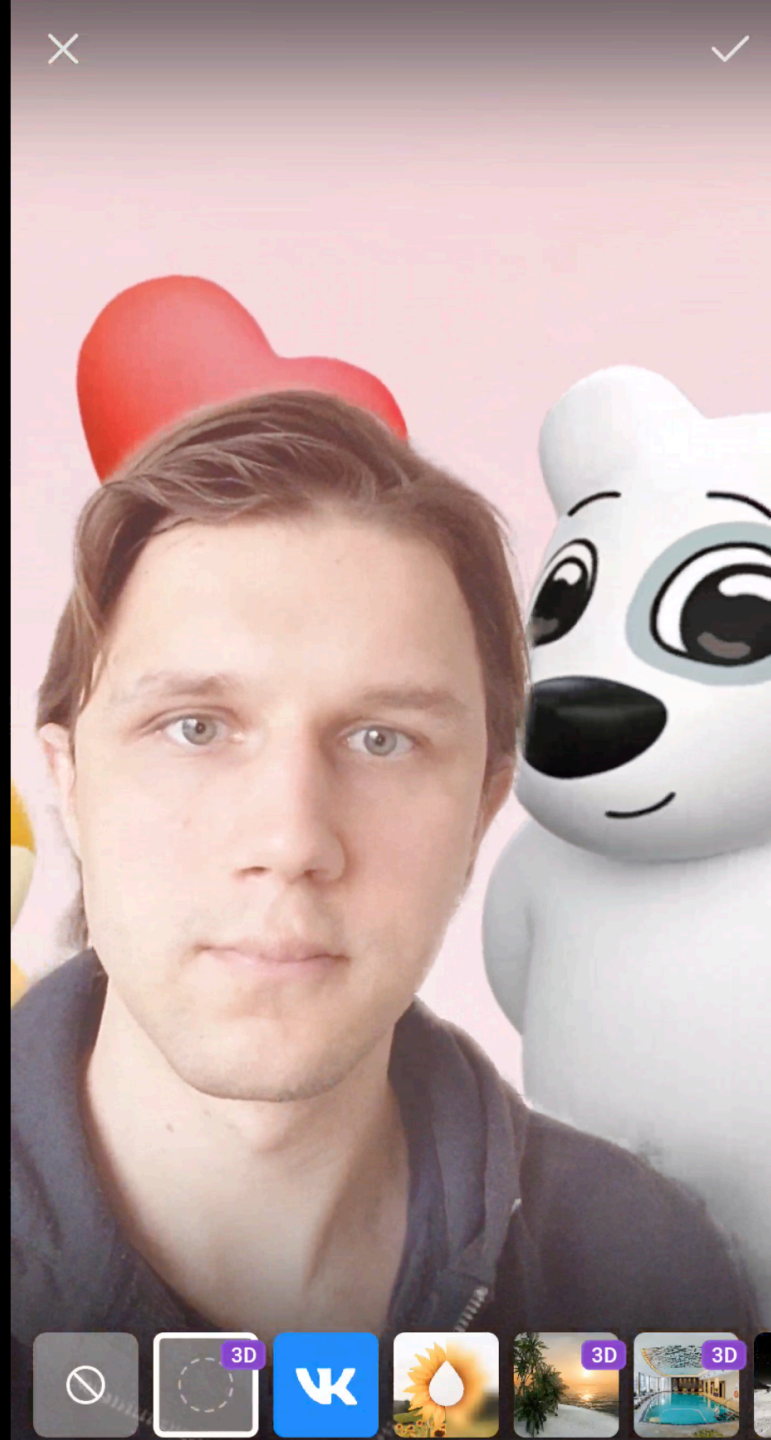
Перехватываем кадры в Android

1. Переопределяем org.webrtc.VideoCapturer
2. В него передаём org.webrtc.CapturerObserver, где слушаем start/stop/onFrame
3. В onFrame перехватываем кадр, изменяем, возвращаем новый



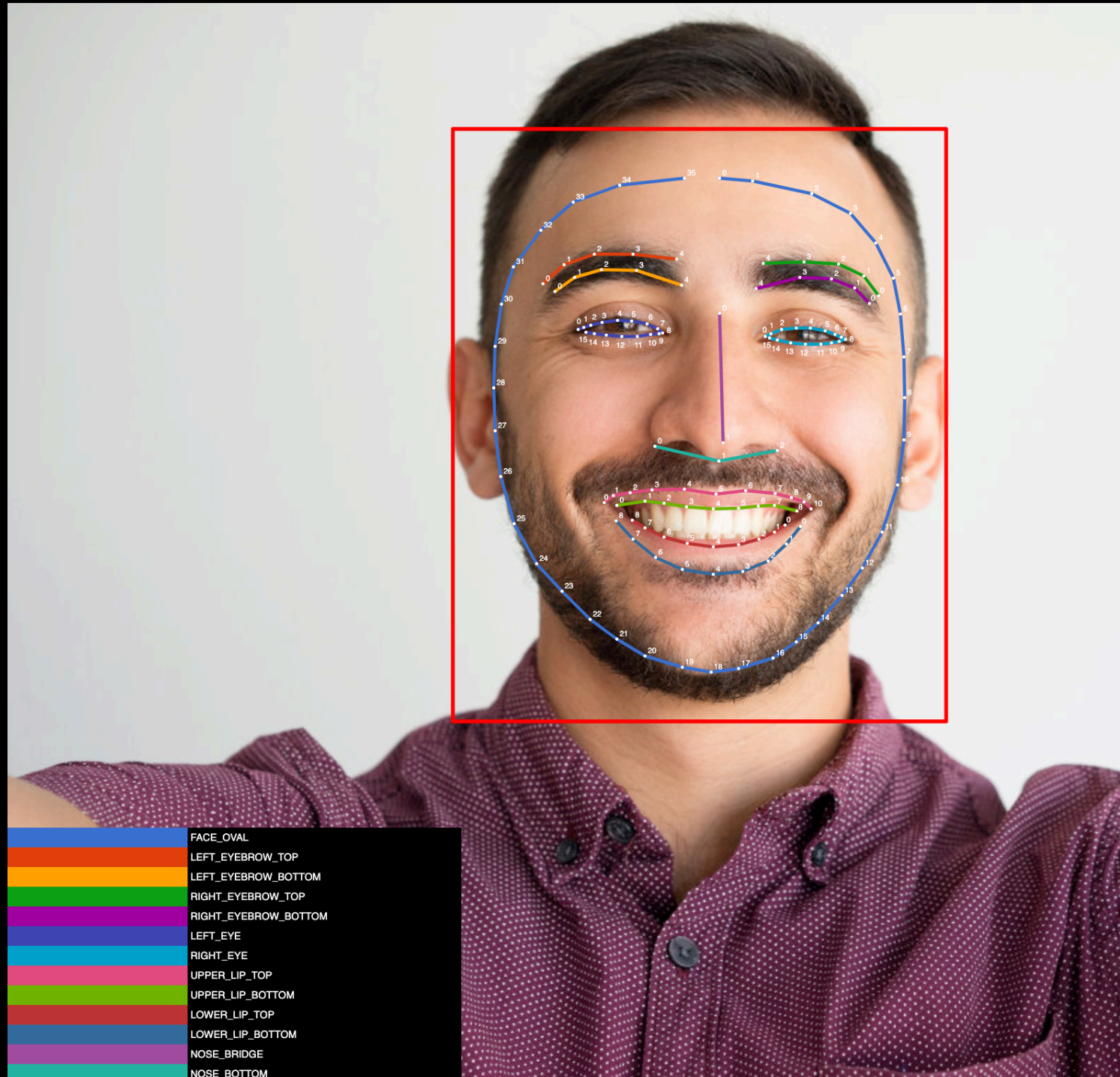
Перехватываем кадры в Android

1. Переопределяем org.webrtc.VideoCapturer
2. В него передаём org.webrtc.CapturerObserver, где слушаем start/stop/onFrame
3. В onFrame перехватываем кадр, изменяем, возвращаем новый
4. На Android по умолчанию все VideoFrame.buffer являются наследниками TextureBuffer



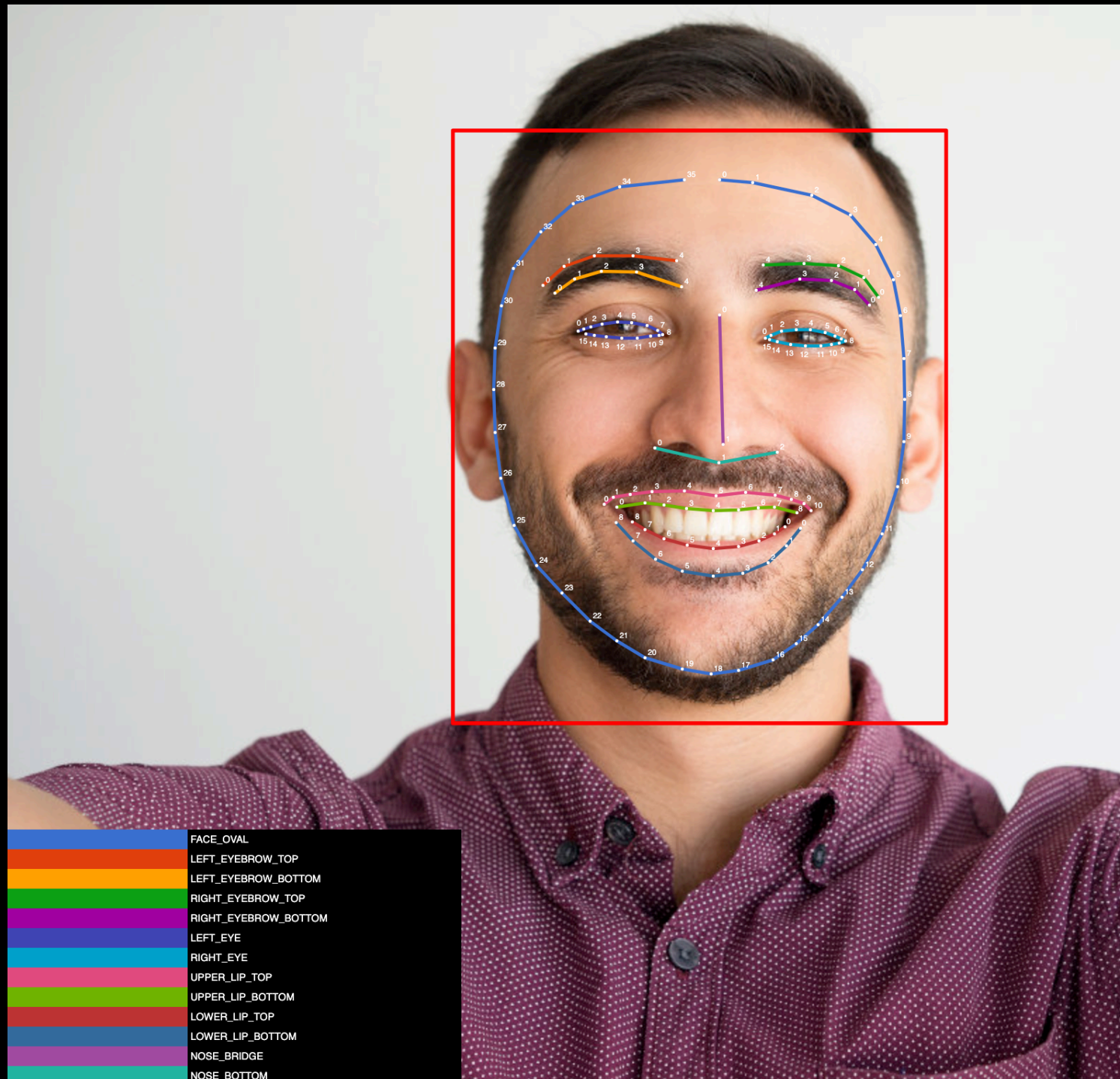
ML Kit

- Бесплатная библиотека от Google



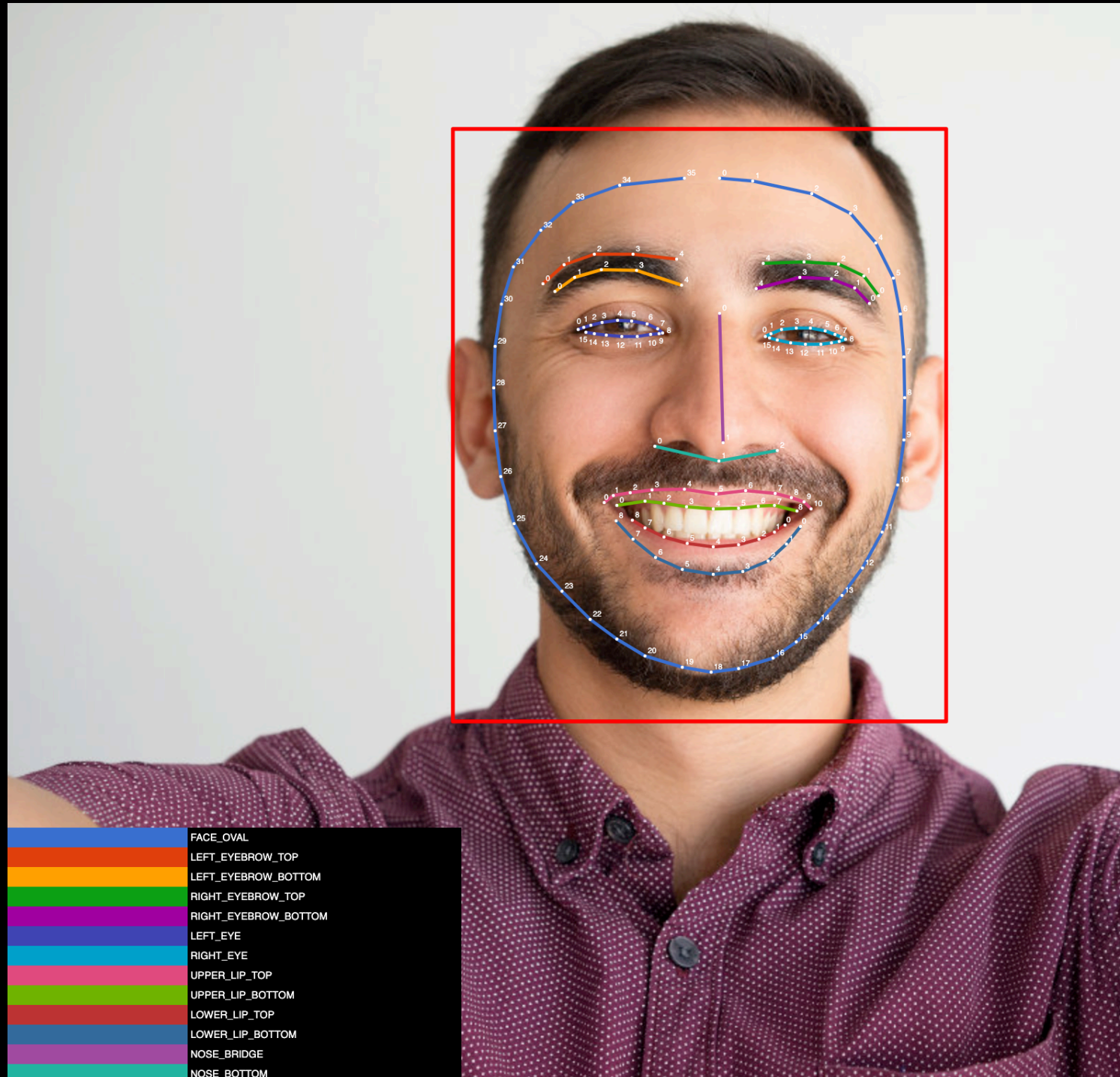
ML Kit

- Бесплатная библиотека от Google
- Android и iOS



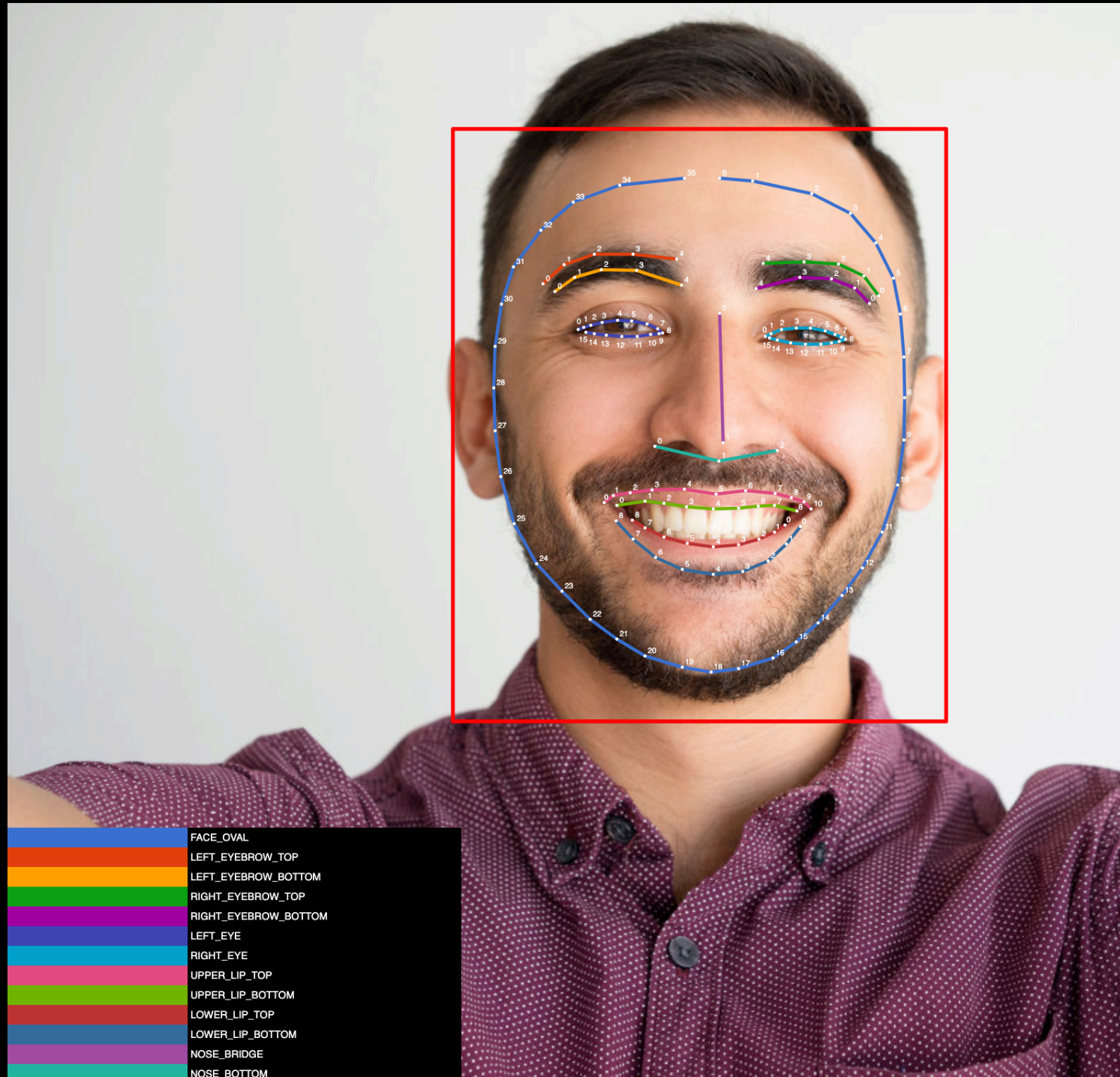
ML Kit

- Бесплатная библиотека от Google
- Android и iOS
- Оффлайн (на девайсе)



ML Kit

- Бесплатная библиотека от Google
- Android и iOS
- Оффлайн (на девайсе)
- Для масок используем — face detection

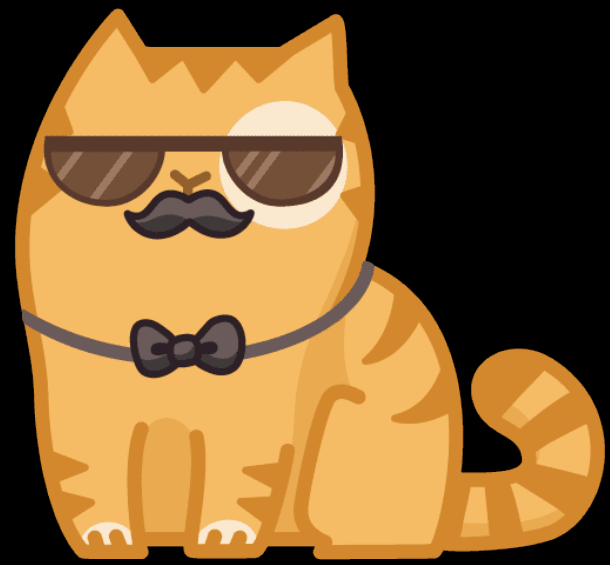


ML Kit

- Бесплатная библиотека от Google
- Android и iOS
- Оффлайн (на девайсе)
- Для масок используем — face detection
- Для виртуального фона — selfie segmentation

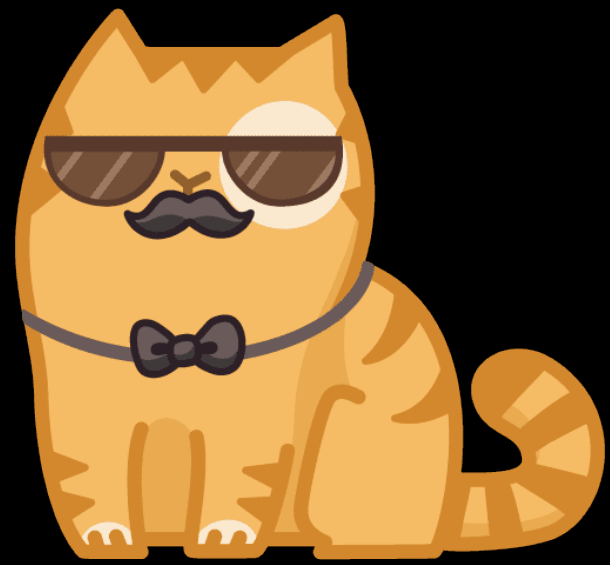
AR-эффекты — это не сложно

- Перехватить кадры в WebRTC легко



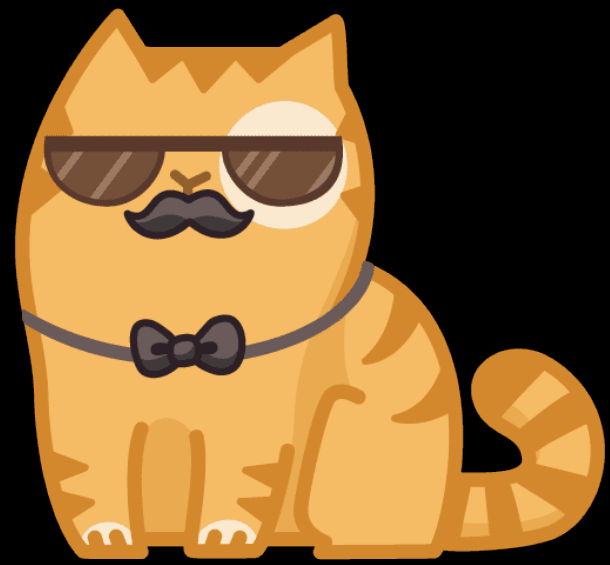
AR-эффекты — это не сложно

- Перехватить кадры в WebRTC легко
- Избегаем копирования данных для оптимальной работы масок

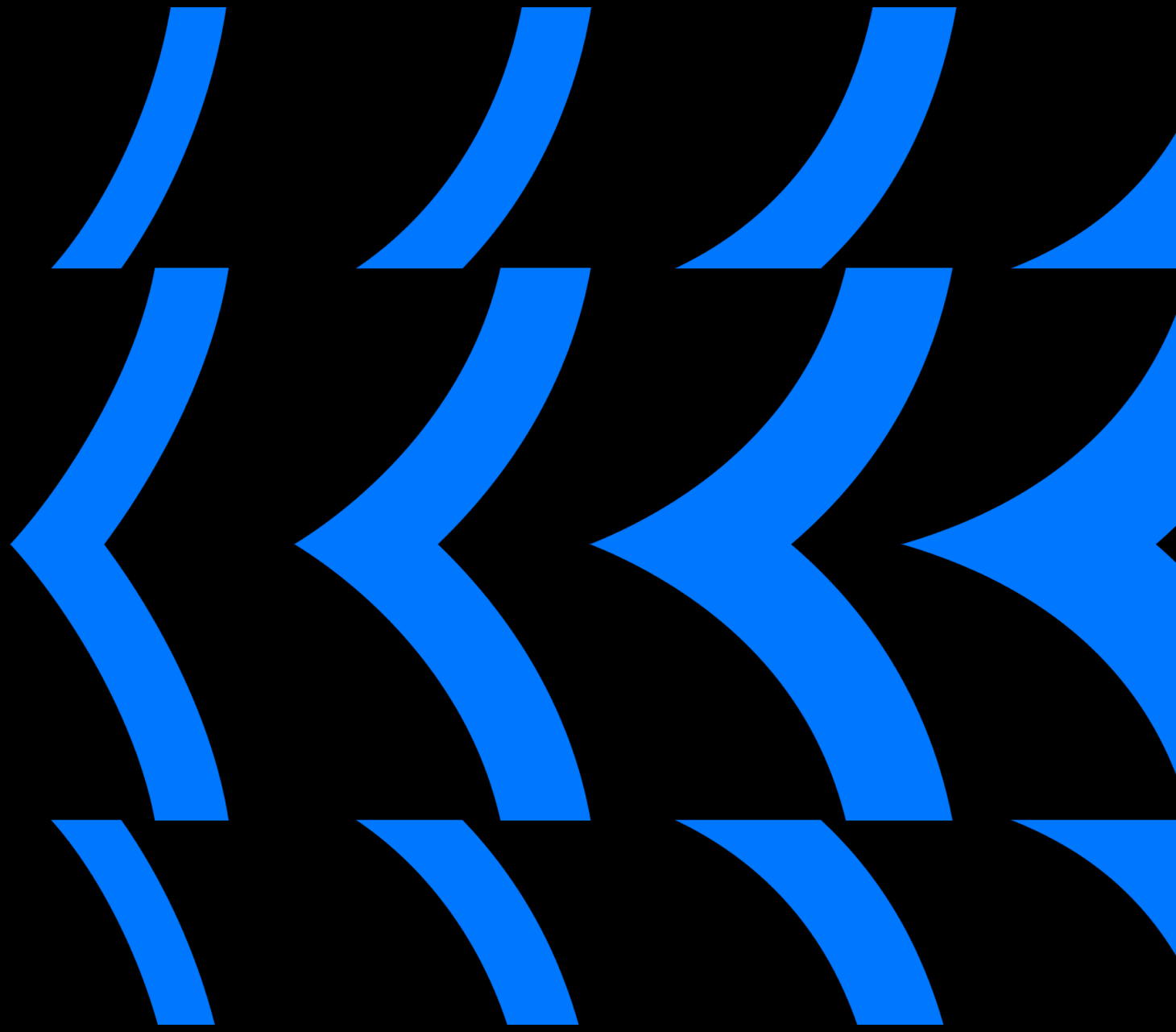


AR-эффекты — это не сложно

- Перехватить кадры в WebRTC легко
- Избегаем копирования данных для оптимальной работы масок
- Есть бесплатные API для магии



Системные API для звонков



Интегрируем звонки с операционной системой

ConnectionService на Android

CallKit на iOS

Интегрируем звонки с операционной системой

ConnectionService на Android

CallKit на iOS

ОС остановит звонок из другого приложения при принятии вашего входящего

Интегрируем звонки с операционной системой

ConnectionService на Android

ОС остановит звонок из другого приложения при принятии вашего входящего

CallKit на iOS

Интеграция в системное приложение звонков

Интегрируем звонки с операционной системой

ConnectionService на Android

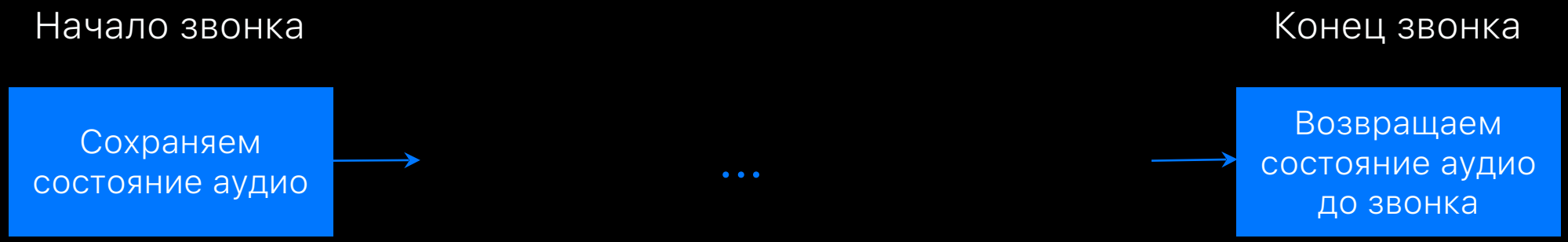
CallKit на iOS

ОС остановит звонок из другого приложения при принятии вашего входящего

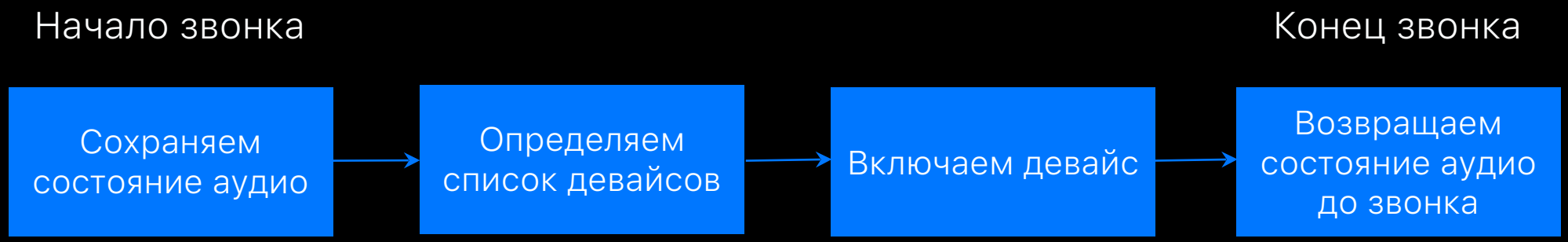
Интеграция в системное приложение звонков

На Android наблюдали баги и разное поведение у разных вендоров. Встраивайте с осторожностью

Управляем аудио выходом на Android



Управляем аудио выходом на Android



Определяем список устройств

Динамик

Наушники
(проводные)

Громкоговоритель

Bluetooth

Динамик и громкоговоритель есть всегда

Динамик

Наушники
(проводные)

Громкоговоритель

Bluetooth

Проводные наушники запрашиваем через AudioManager

```kotlin
fun hasWiredHeadset(): Boolean {
  return if (Build.VERSION.SDK_INT < Build.VERSION_CODES.M) {
    audioManager.isWiredHeadsetOn
  } else {
    val types = listOf(
      AudioDeviceInfo.TYPE_WIRED_HEADPHONES,
      AudioDeviceInfo.TYPE_WIRED_HEADSET,
      AudioDeviceInfo.TYPE_USB_DEVICE
    )
    audioManager
        .getDevices(AudioManager.GET_DEVICES_ALL)
        .any { it.type in types }

    }
  }
}
```

Динамик

Наушники (проводные)

Громкоговоритель

Bluetooth

Для Bluetooth нужно несколько шагов

1. android.permission.BLUETOOTH

Динамик

Наушники (проводные)

Громкоговоритель

Bluetooth

Для Bluetooth нужно несколько шагов

1. android.permission.BLUETOOTH
2. Слушаем наличие девайса через BluetoothAdapter + BluetoothProfile

Динамик

Наушники (проводные)

Громкоговоритель

Bluetooth

Для Bluetooth нужно несколько шагов

1. android.permission.BLUETOOTH
2. Слушаем наличие девайса через BluetoothAdapter + BluetoothProfile
3. Дополнительно обновляем девайс по broadcast-у BluetoothHeadset: ACTION_CONNECTION_STATE_CH ANGED и ACTION_AUDIO_STATE_CHANGED

Динамик

Наушники (проводные)

Громкоговоритель

Bluetooth

Для Bluetooth нужно несколько шагов

1. android.permission.BLUETOOTH
2. Слушаем наличие девайса через BluetoothAdapter + BluetoothProfile
3. Дополнительно обновляем девайс по broadcast-у BluetoothHeadset: ACTION_CONNECTION_STATE_CH ANGED и ACTION_AUDIO_STATE_CHANGED
4. val hasBluetoothHeadset = adapter.getProfileConnectionState(HEADSET) == BluetoothProfile.*STATE_CONNECTED*

Динамик

Наушники (проводные)

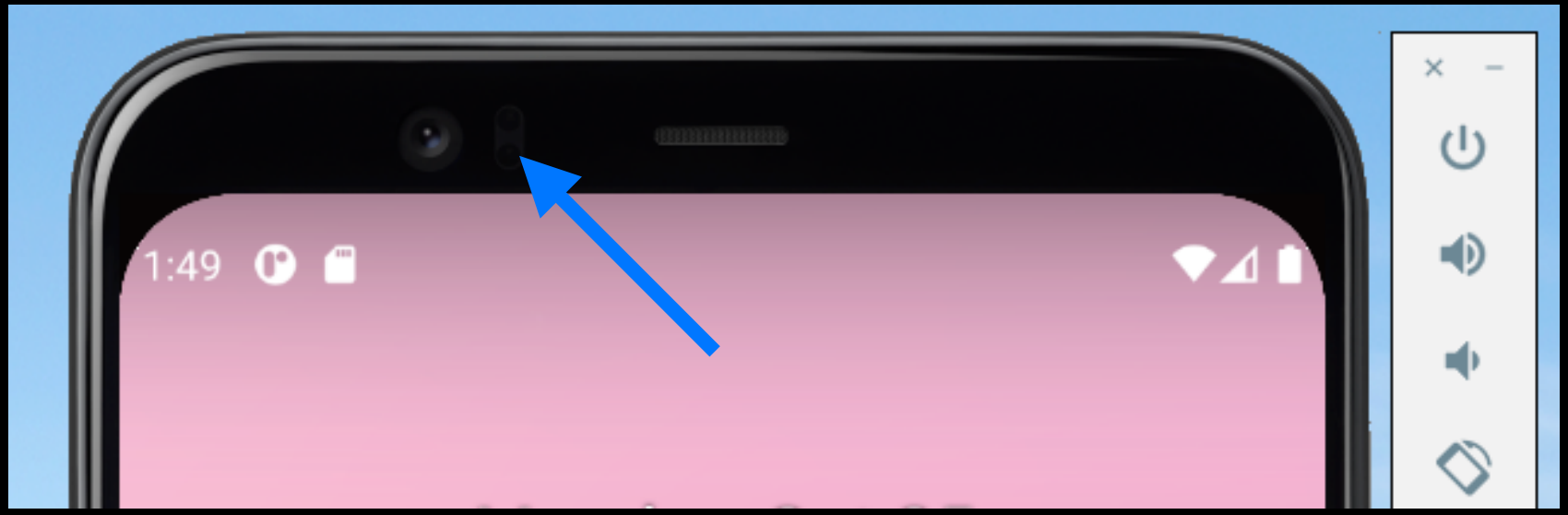
Громкоговоритель

Bluetooth

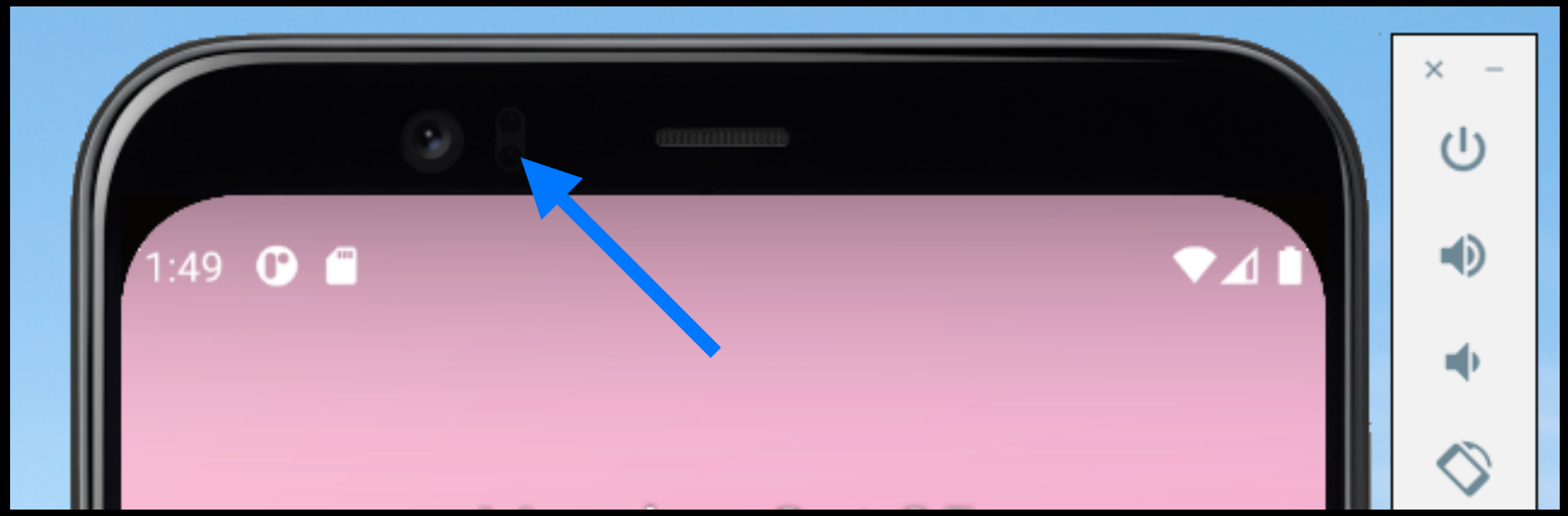
Icons from www.flaticon.com

При касании экрана ухом можно случайно нажать кнопку

Специальный датчик оценивает расстояние



Специальный датчик оценивает расстояние




```
powerManager.newWakeLock(PowerManager.PROXIMITY_SCREEN_OFF_WAKE_LOCK, WAKE_LOCK_TAG)
```

Делаем как в sample

https://chromium.googlesource.com/external/webrtc/+/refs/heads/main/examples/androidapp/src/org/appspot/apprtc/

Делаем как в sample. Но переводим код на background thread

https://chromium.googlesource.com/external/webrtc/+/refs/heads/main/examples/androidapp/src/org/appspot/apprtc/

Применение выбранного девайса может занимать секунду и более!

На iOS

- На iOS все хорошо

На iOS

- На iOS все хорошо
- Не стоит забывать про sample rate

На iOS

- На iOS все хорошо
- Не стоит забывать про sample rate
- prefferedSampleRate не панацея

ОС друг и враг наш

- На iOS CallKit обязателен, но зато делает всё за нас

ОС друг и враг наш

- На iOS CallKit обязателен, но зато делает всё за нас
- На Android всё опционально, но много подводных камней

ОС друг и враг наш

- На iOS CallKit обязателен, но зато делает всё за нас
- На Android всё опционально, но много подводных камней
- Код из семплов не всегда идеален :)

Групповые звонки

Групповые звонки на WebRTC из коробки не заработают

- Оптимизируем аудио

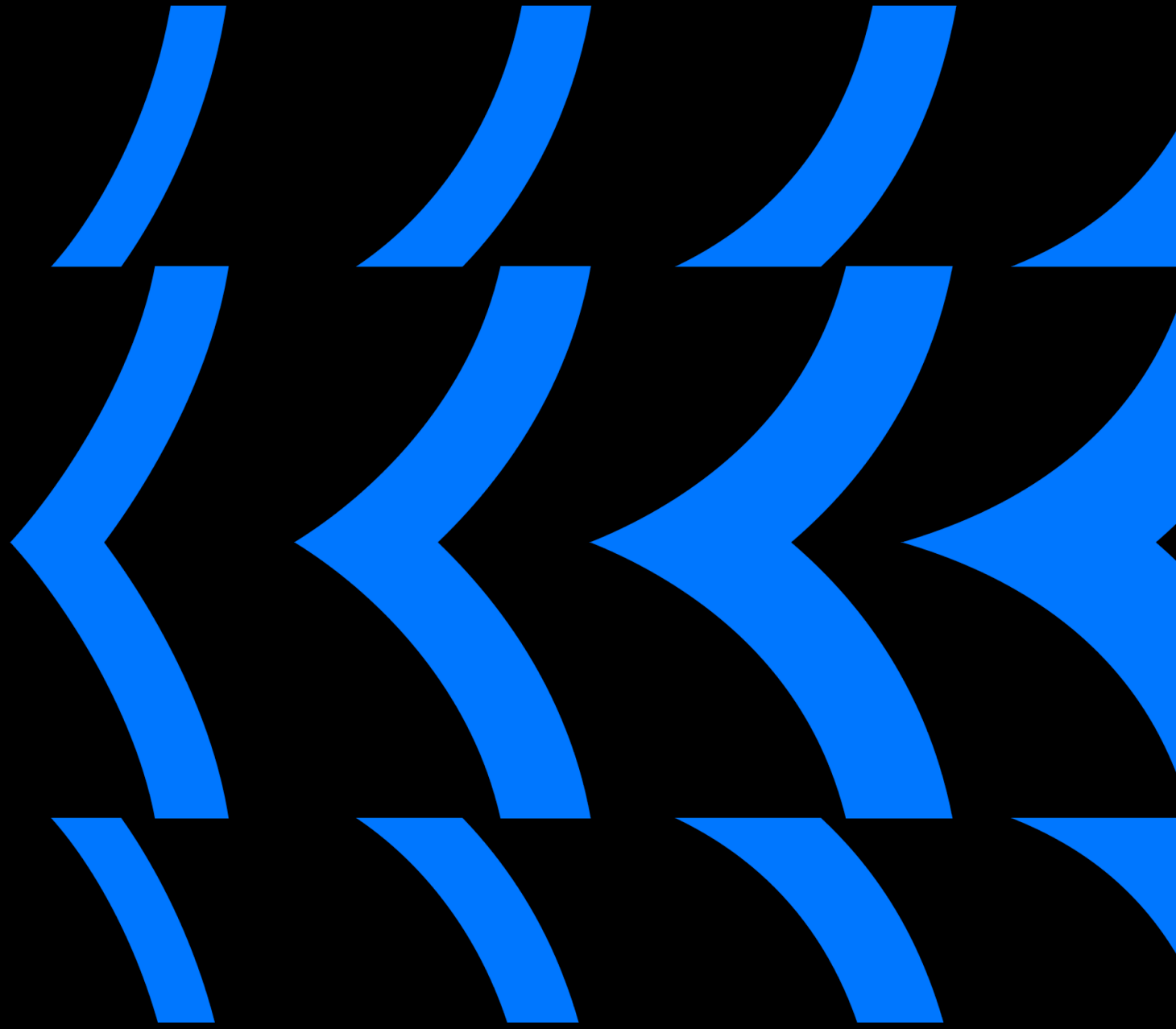
Групповые звонки на WebRTC из коробки не заработают

- Оптимизируем аудио
- Оптимизируем видео

Групповые звонки на WebRTC из коробки не заработают

- Оптимизируем аудио
- Оптимизируем видео
- Оптимизируем сеть

Аудио



Групповое аудио

- Для каждого пользователя есть аудиотрек

Групповое аудио

- Для каждого пользователя есть аудиотрек
- По треку определяем voice activity

Групповое аудио

- Для каждого пользователя есть аудиотрек
- По треку определяем voice activity
- По треку определяем качество сети

Что если все начнут одновременно говорить?

- много суммарного входящего трафика — порядка 1 Мбит/с для десяти участников и кодирования аудио в высоком качестве

Что если все начнут одновременно говорить?

- много суммарного входящего трафика — порядка 1 Мбит/с для десяти участников и кодирования аудио в высоком качестве
- N-1 декодеров на клиенте

Что если все начнут одновременно говорить?

- много суммарного входящего трафика — порядка 1 Мбит/с для десяти участников и кодирования аудио в высоком качестве
- N-1 декодеров на клиенте
- WebRTC начинает умирать на большом количестве аудио стримов

Audio Mix

- Один трек на всех участников

Audio Mix

- Один трек на всех участников
- Просим сервер присылать нотификации об активности конкретных участников

Audio Mix

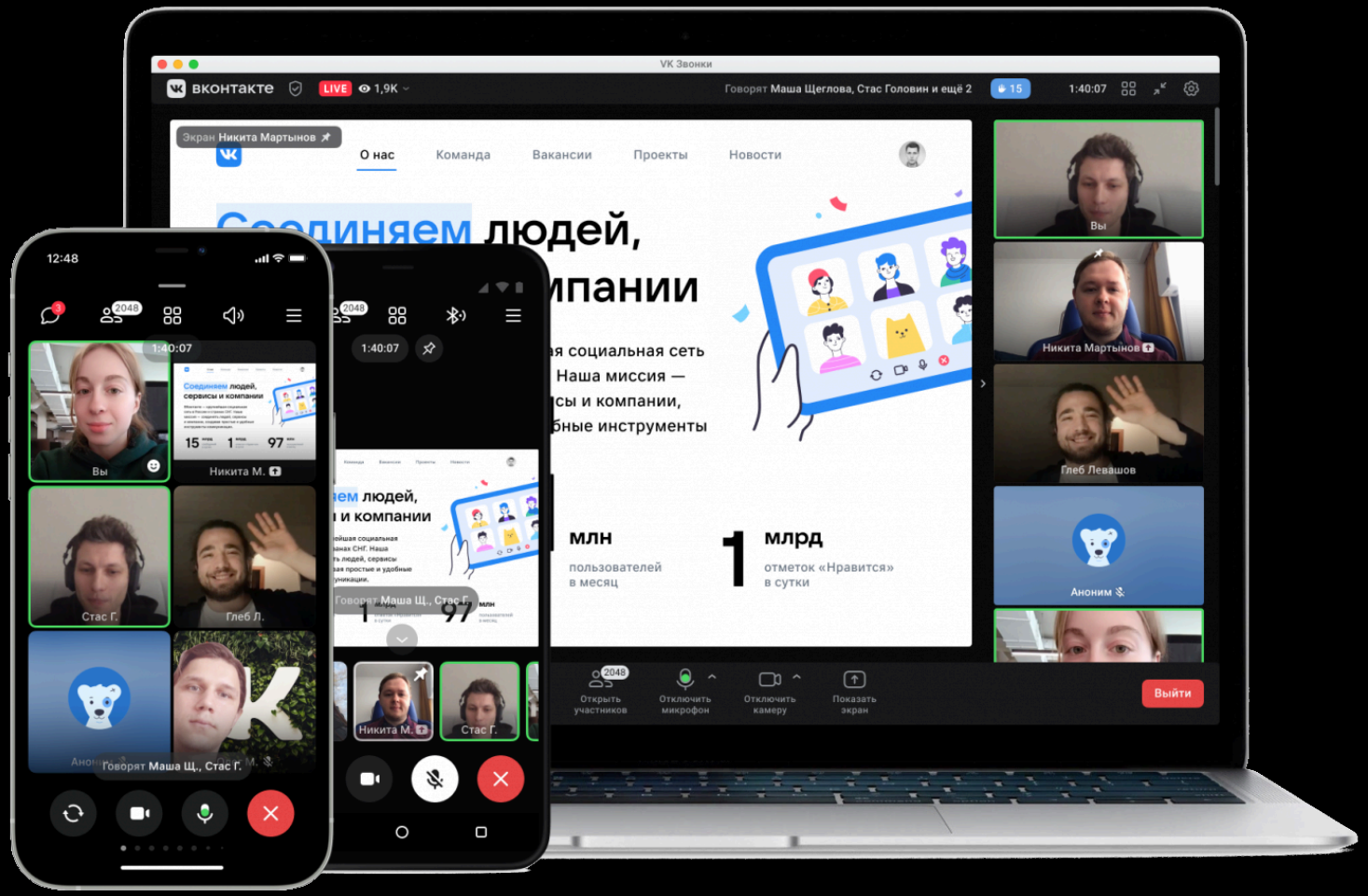
- Один трек на всех участников
- Просим сервер присылать нотификации об активности конкретных участников
- Для оптимизации задержек переносим нотификации на DataChannel

Групповое аудио: итоги

- 1 декодер на любое количество говорящих
- Нотификации о “говорении” с низкими задержками
- Прогнозируемая нагрузка на сеть

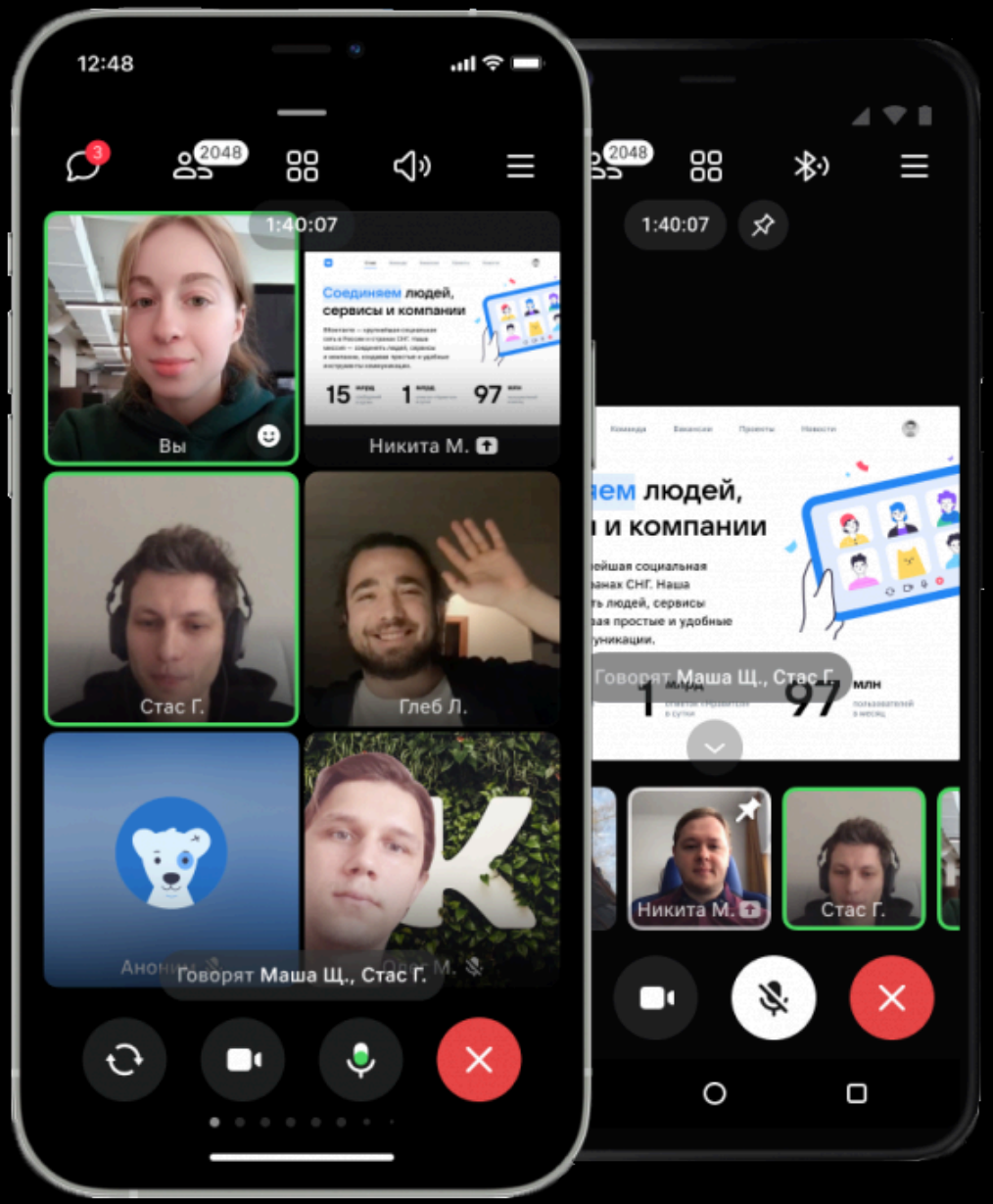
Видео

Много видео долго декодировать



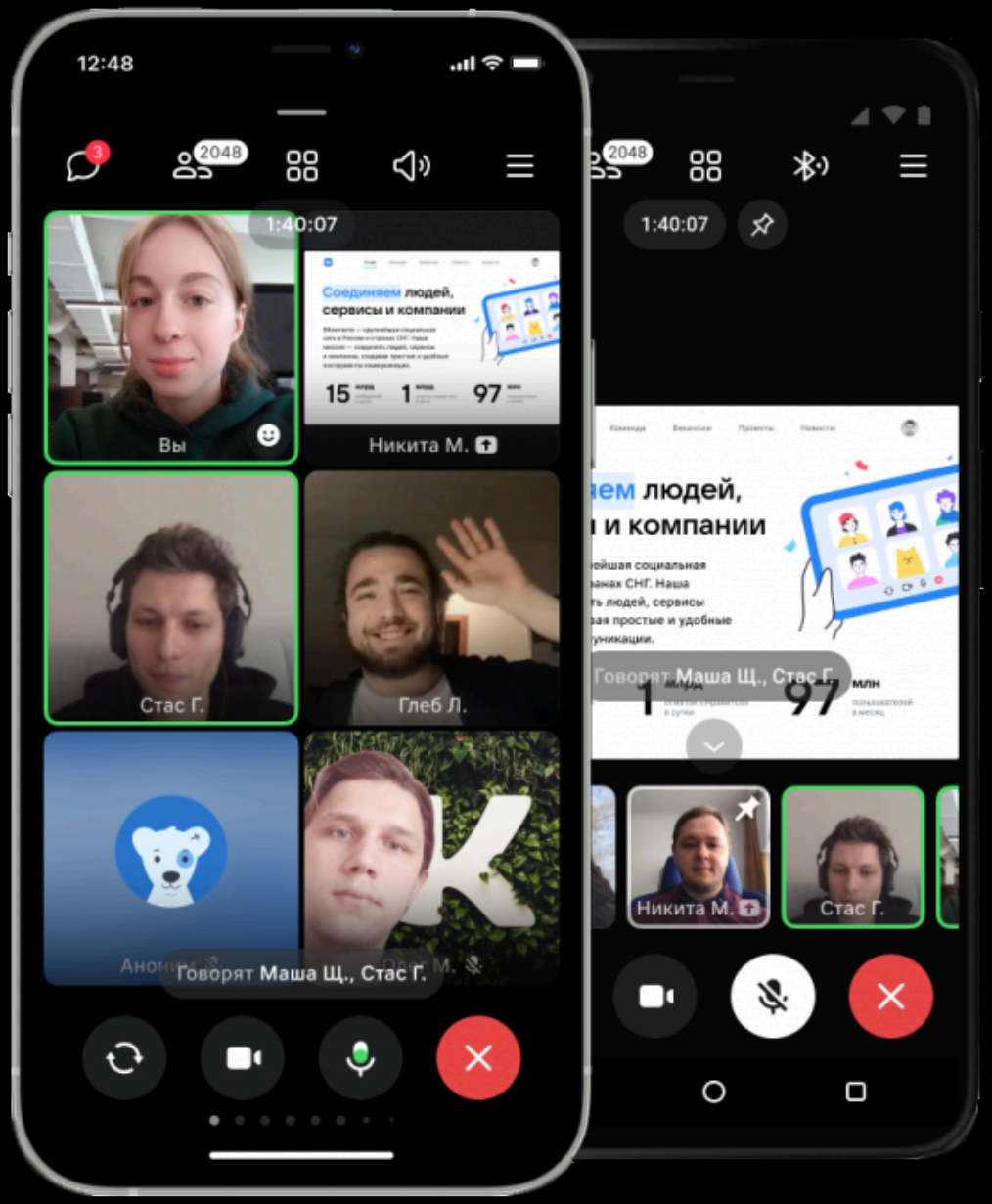
Клиенту все видео не нужны

- Рисовать людей за экраном не надо



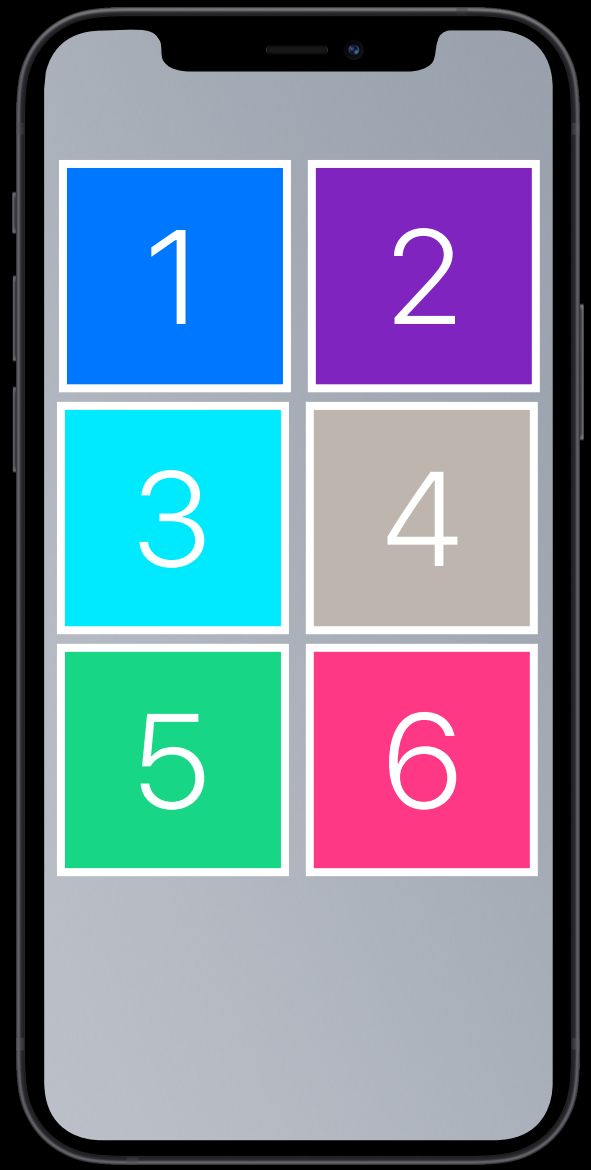
Клиенту все видео не нужны

- Рисовать людей за экраном не надо
- Получать 4K качество для маленького превью не надо



Просим сервер присылать только видимых участников в нужном разрешении

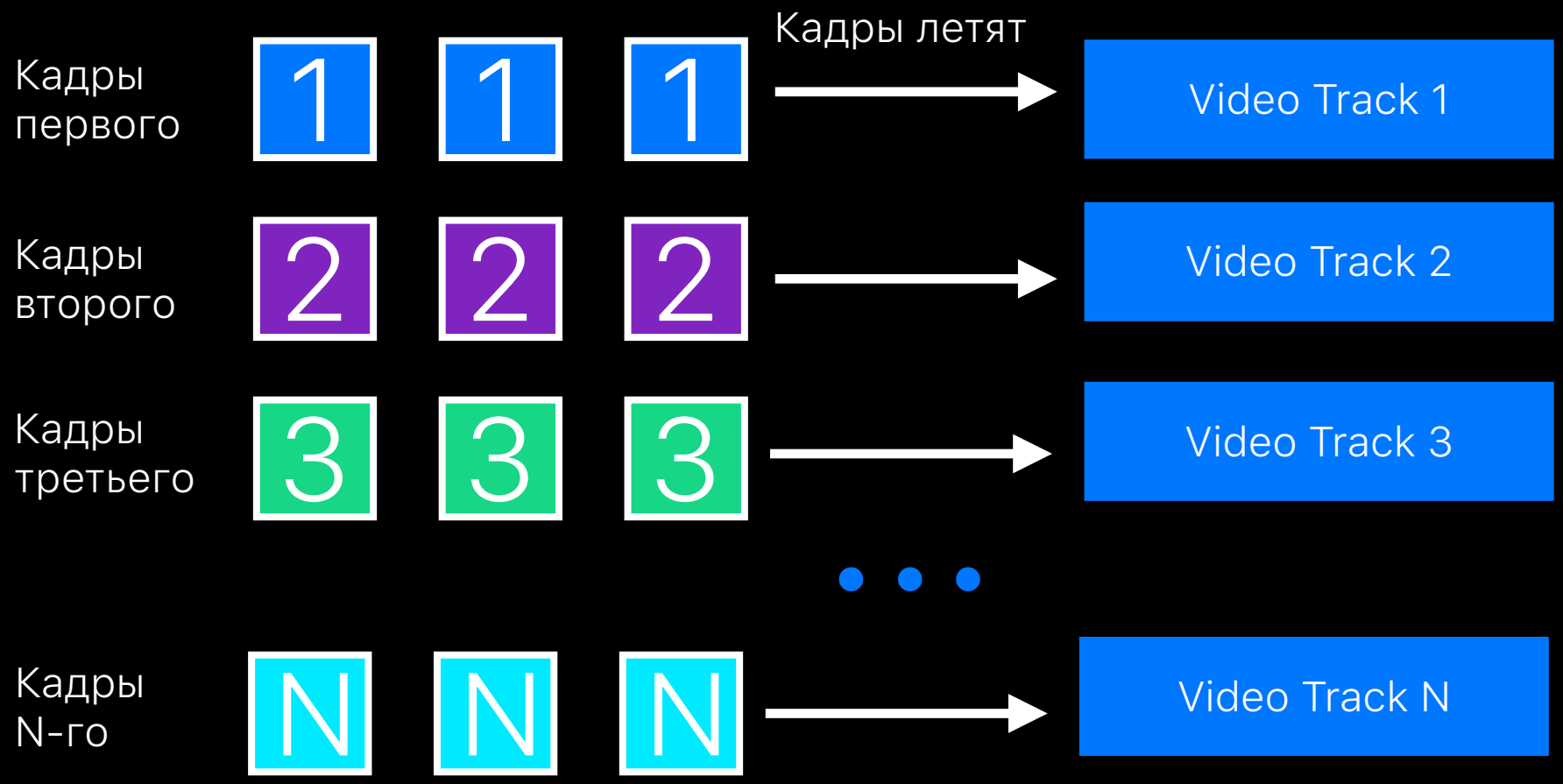
```
[
  {"id": "1", "size":"100x100"},
  {"id": "2", "size":"100x100"},
  {"id": "3", "size":"100x100"},
  {"id": "4", "size":"100x100"},
  {"id": "5", "size":"100x100"},
  {"id": "6", "size":"100x100"}
]
```



Просим сервер присылать только видимых участников в нужном разрешении

```
[
  {"id": "1", "size":"200x400"},
  {"id": "2", "size":"50x50"},
  {"id": "3", "size":"50x50"},
  {"id": "4", "size":"50x50"},
  {"id": "5", "size":"50x50"}
]
```

По умолчанию WebRTC создаёт видеотрек на каждого участника



Проблемы видеотреков

- На каждого участника заводится отдельная сущность

Проблемы видеотреков

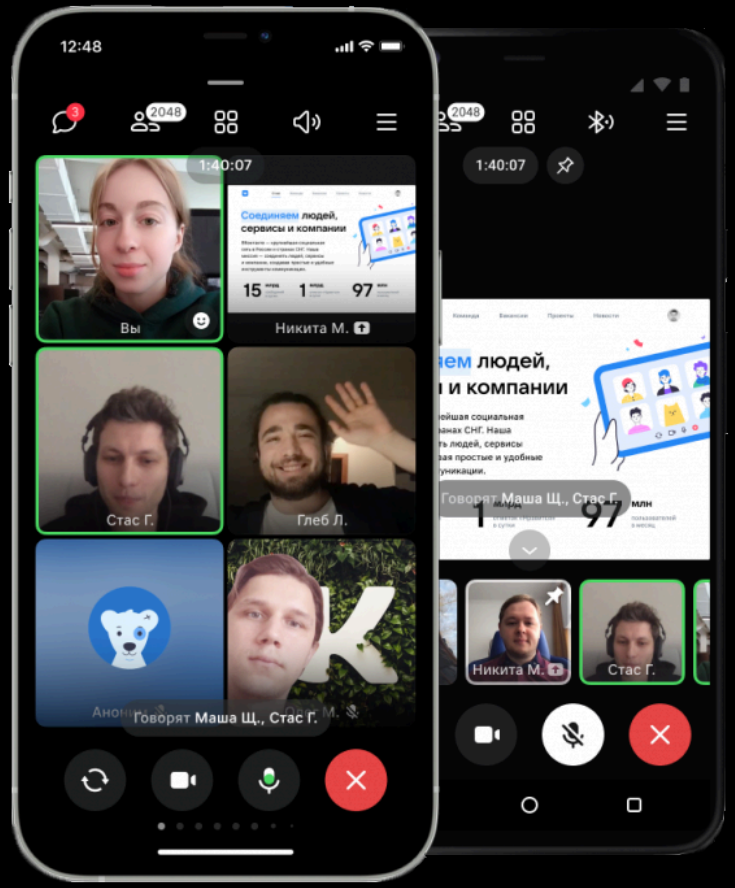
- На каждого участника заводится отдельная сущность
- Треки нужно поддерживать живыми

Проблемы видеотреков

- На каждого участника заводится отдельная сущность
- Треки нужно поддерживать живыми
- Нужно переслать SDP всем при входе или выходе одного участника

```
v=0
o=jdoe 2890844526 2890842807 IN IP4 10.47.16.5
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.example.com/seminars/sdp.pdf
e=j.doe@example.com (Jane Doe)
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
a=recvonly
m=audio 49170 RTP/AVP 0
m=video 51372 RTP/AVP 99
a=rtpmap:99 h263-1998/90000
```

Фиксируем число видеотреков максимальным количеством видео на экране



Кадры летят

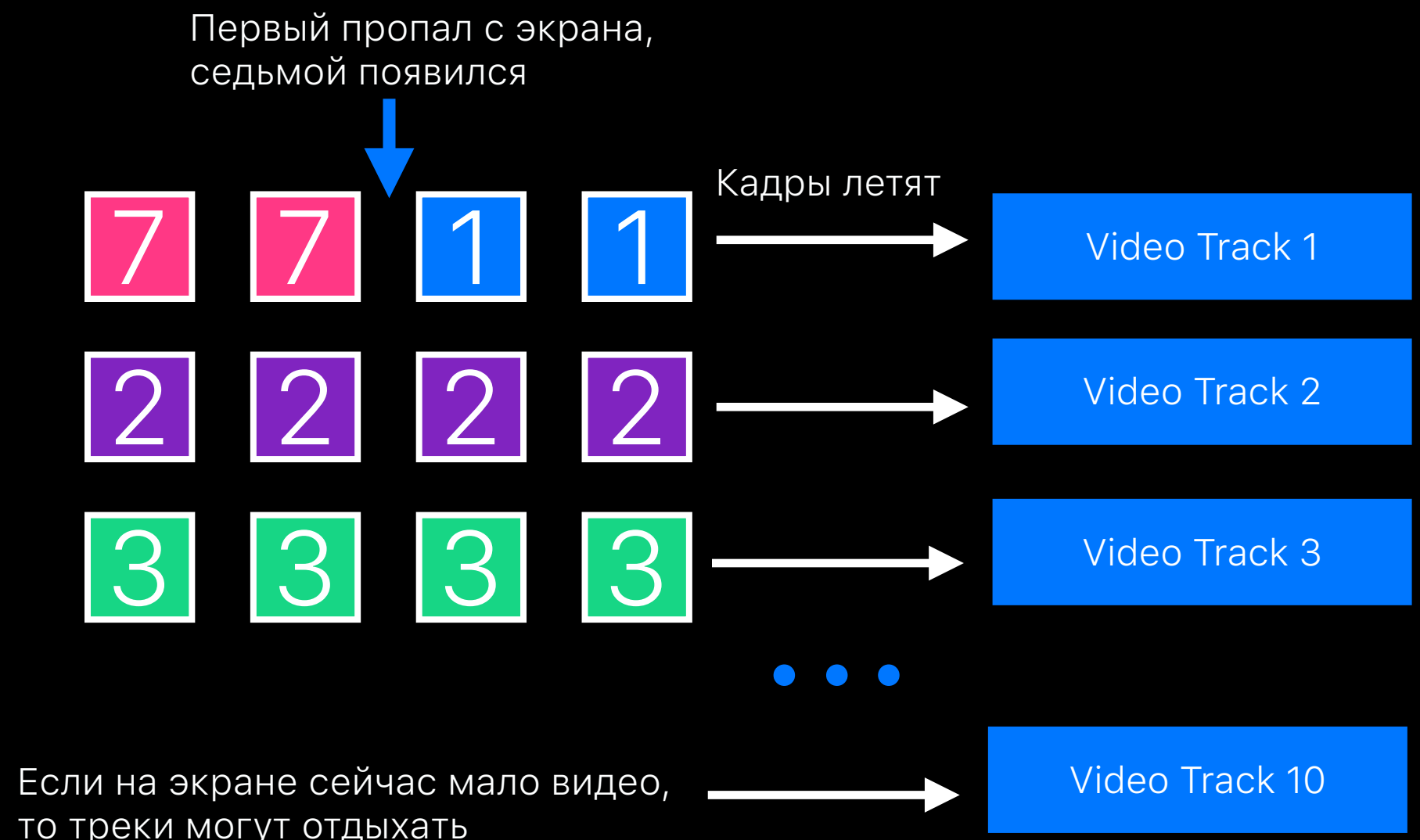
Video Track 1

Video Track 2

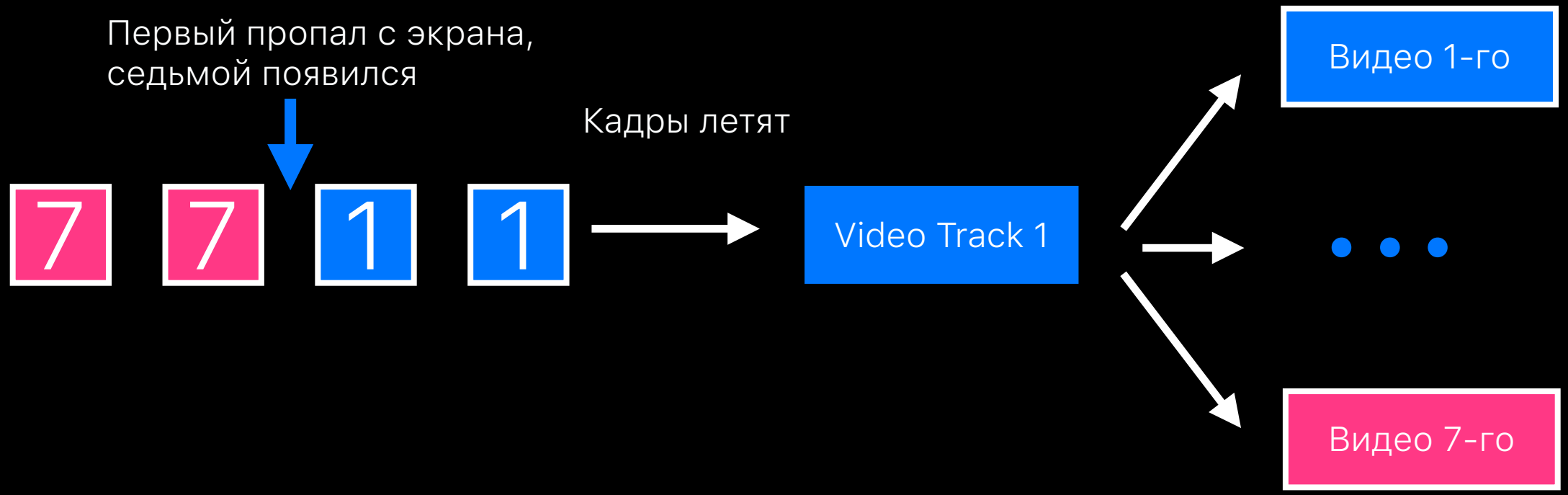
Video Track 3

Video Track 10

Теперь кадры не привязаны к видеотрекам

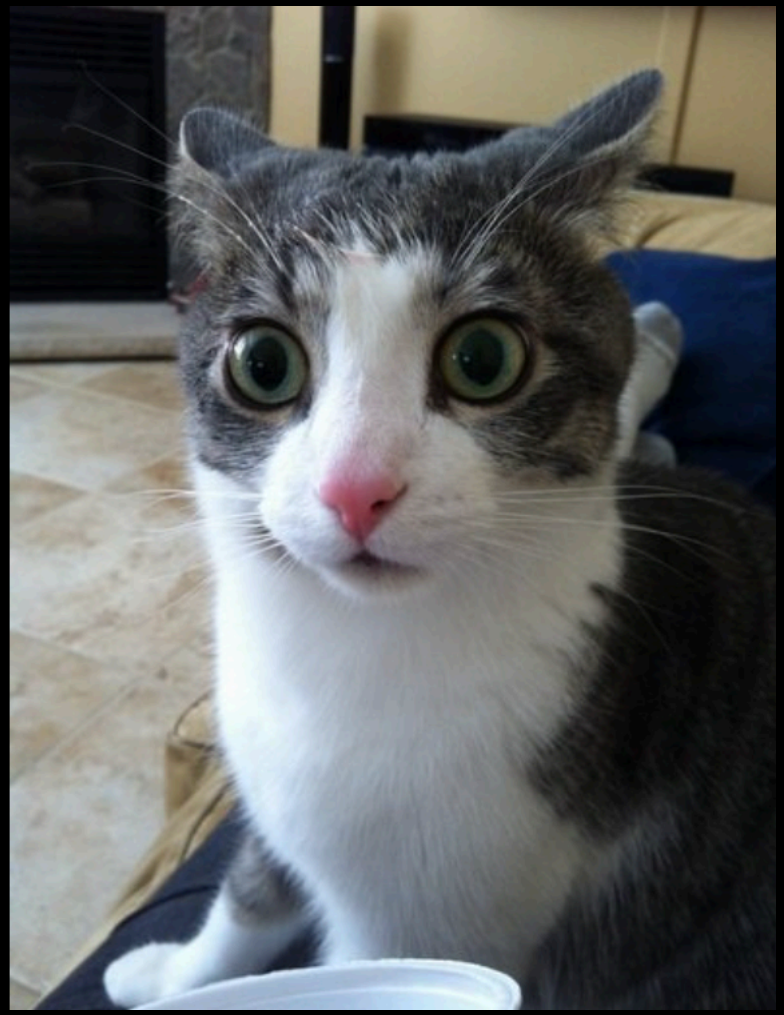


Реализуем прокси,
который раскидывает кадры по видеоэлементам



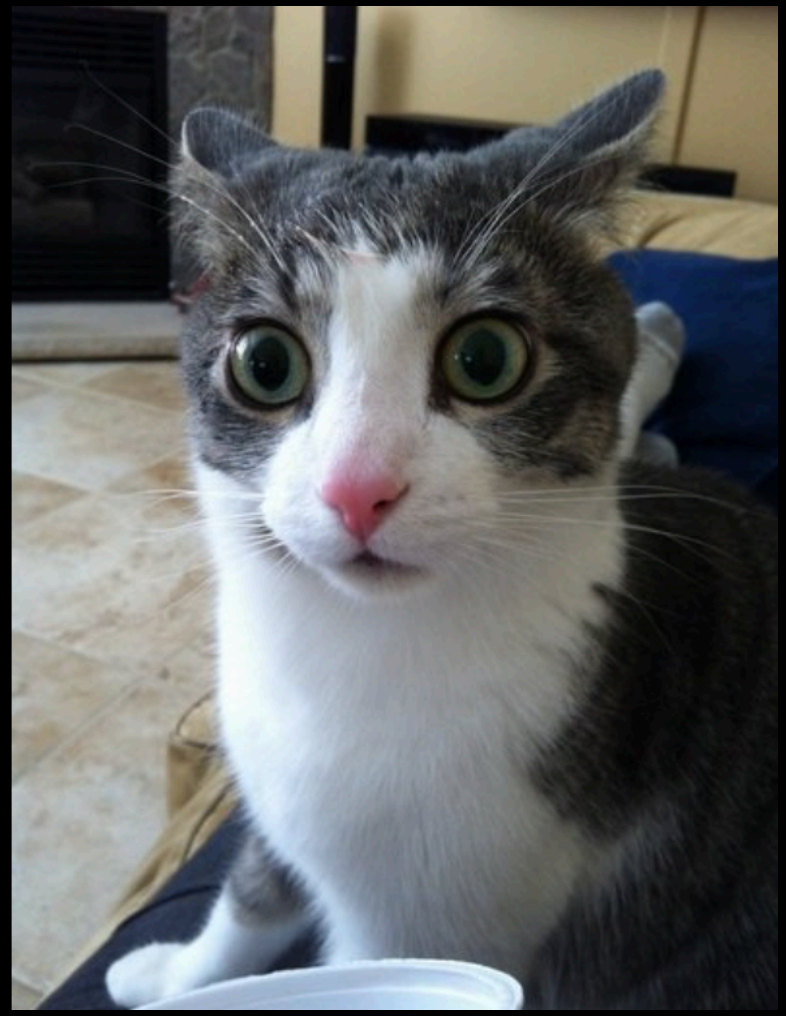
А теперь подумаем про экономию отправки видео

- А что если не отправлять видео, если нас никто не смотрит?



А теперь подумаем про экономию отправки видео

- А что если не отправлять видео, если нас никто не смотрит?
- А что если отправлять видео в меньшем разрешении, если нас видят только в превью?



Видео, видео, видео

- Договариваемся с сервером о требуемых видео и размерах

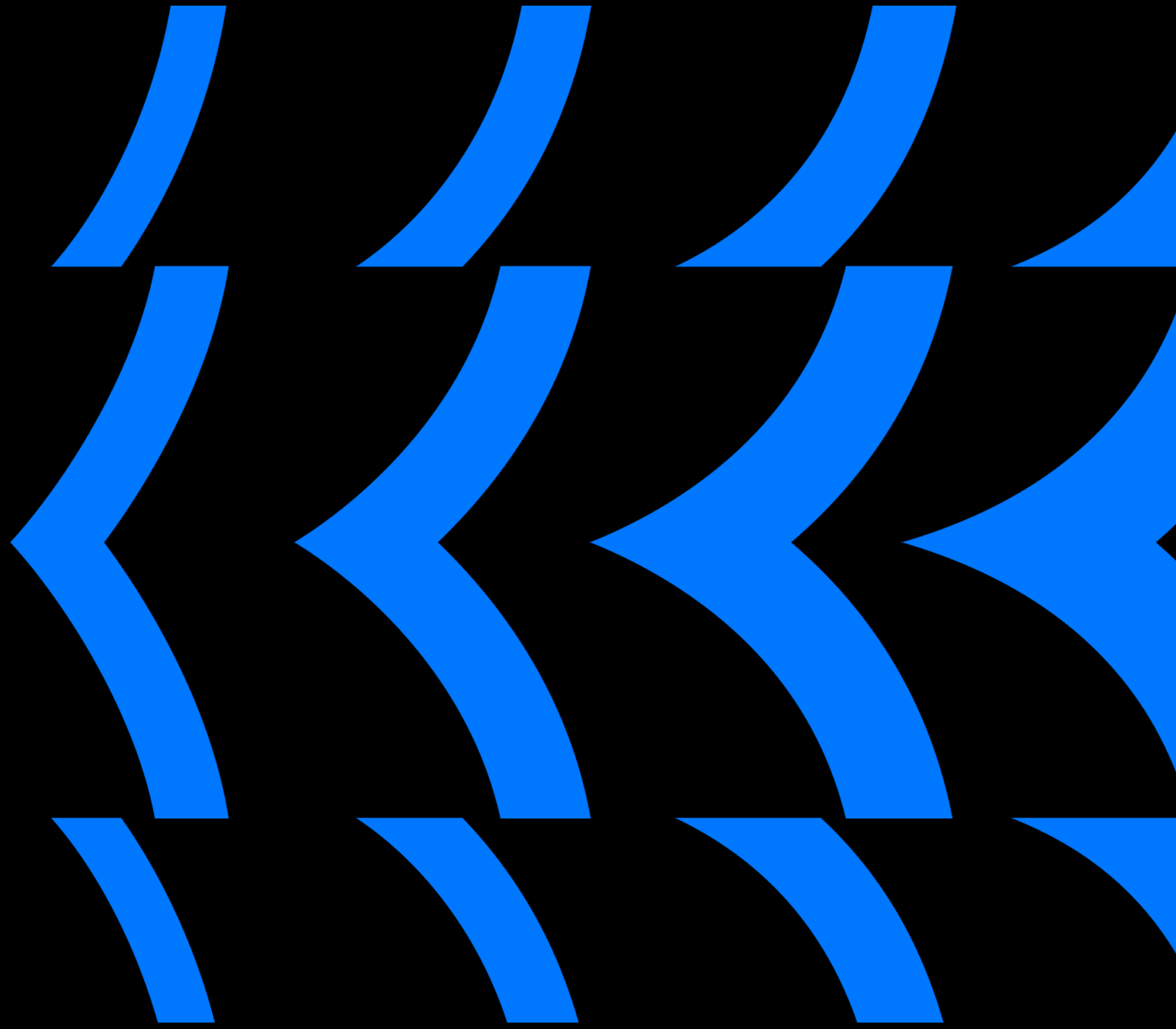
Видео, видео, видео

- Договариваемся с сервером о требуемых видео и размерах
- Убираем связь между участниками и видео треками

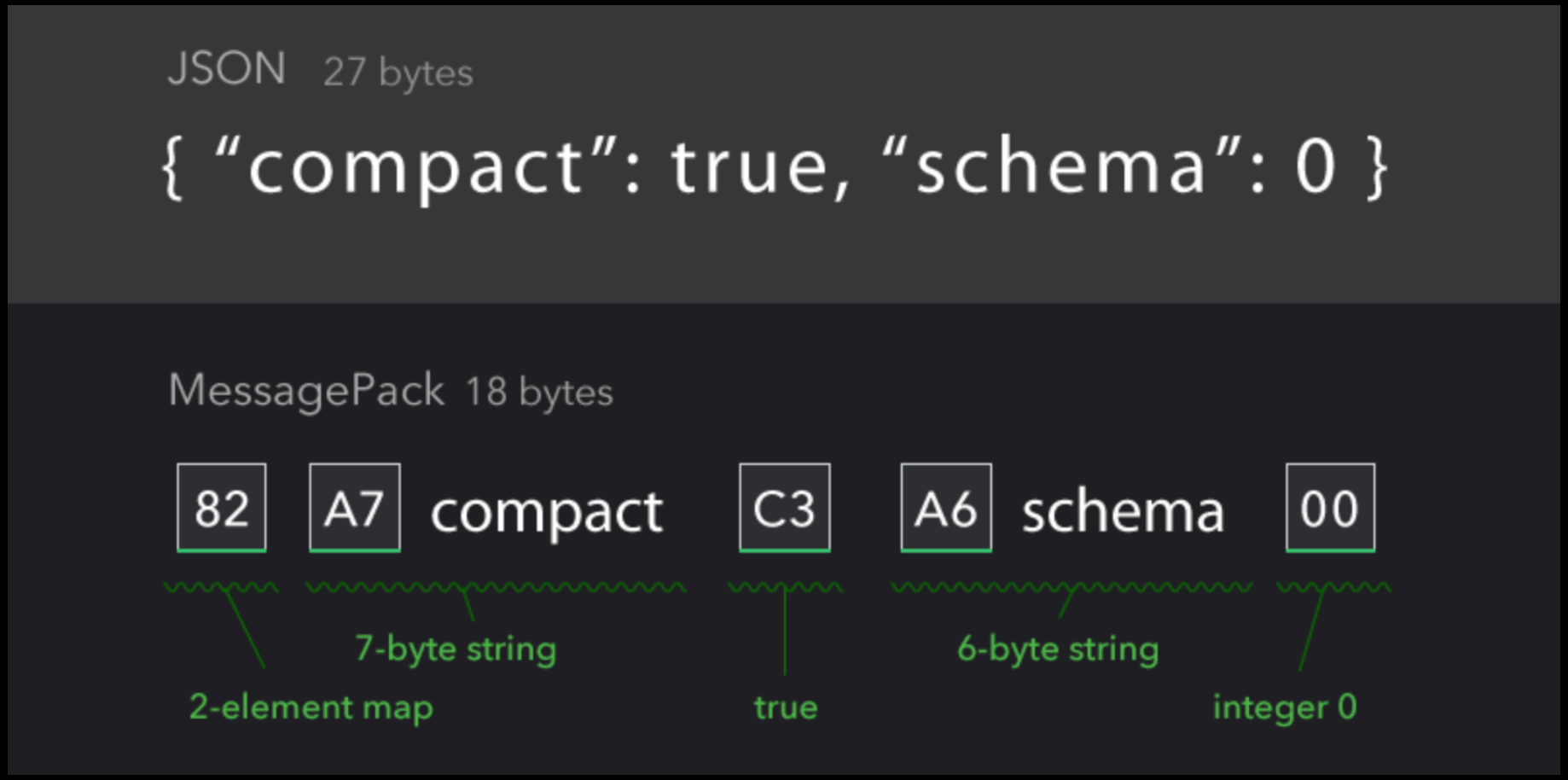
Видео, видео, видео

- Договариваемся с сервером о требуемых видео и размерах
- Убираем связь между участниками и видео треками
- Не отправляем видео, если нас никто не видит

Сеть



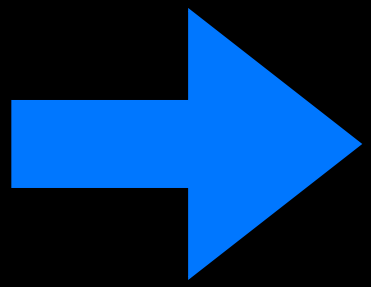
Переводим частые сообщения на MessagePack через DataChannel



https://msgpack.org/

Убираем значения по умолчанию

```
{
  "audio": true,
  "video": false,
  "screen_share": false
}
```



```
{
  "audio": true
}
```

Проверяем включённость permessage-deflate у WebSocket

- В Android начиная с 4.5 OkHttp

Проверяем включённость permessage-deflate у WebSocket

- В Android начиная с 4.5 OkHttp
- В iOS зависит от библиотеки
- В iOS 15 в системном API WebSocket

Сеть итоги

- Уменьшаем размер и частотность частых сообщений
- Проверяем настройки WebSocket
- Оптимизации аудио и видео значительно уменьшают нагрузку

Подведём итоги

Резюме

- WebRTC отличный проект для создания звонков и не только

Резюме

- WebRTC отличный проект для создания звонков и не только
- В 1 на 1 звонке желательно
  - Починить скриншару
  - Настроить выбор кодеков

Резюме

- WebRTC отличный проект для создания звонков и не только
- В 1 на 1 звонке желательно
  - Починить скриншару
  - Настроить выбор кодеков
- В 1 на 1 звонке можно
  - Улучшать аудио пайплайн
  - Добавлять виртуальные фоны и маски

Резюме

- WebRTC отличный проект для создания звонков и не только
- В 1 на 1 звонке желательно
  - Починить скриншару
  - Настроить выбор кодеков
- В 1 на 1 звонке можно
  - Улучшать аудио пайплайн
  - Добавлять виртуальные фоны и маски
- В больших групповых звонках обязательно
  - Перейти от N к 1 или нескольким аудио
  - Принимать только видео на экране

Будем ВКонтакте!

Никита Разумный
VK: @xc

Иван Шафран
VK: @ivan.shafran
twitter: @NotShafran