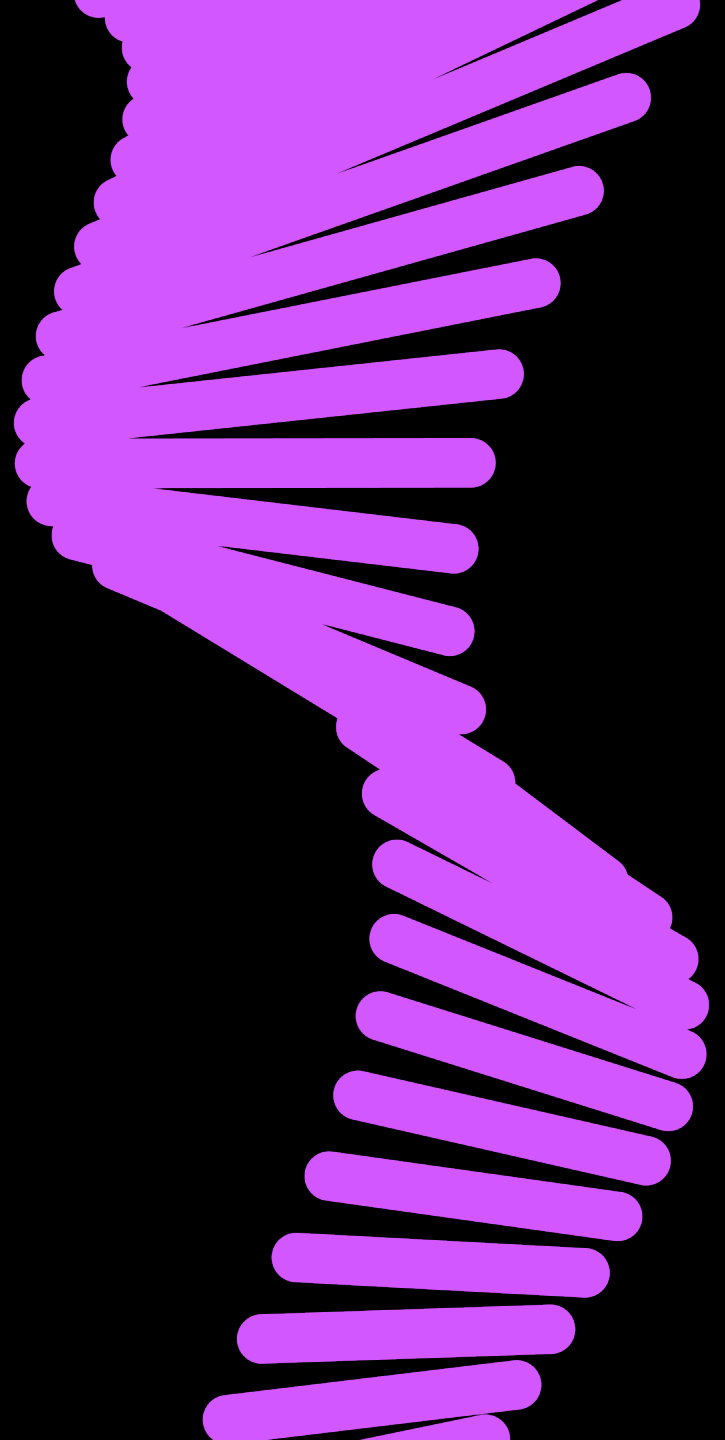
VK Клипы

Моментальный запуск

Как VK Клипы работают с плейбеком видео на Android



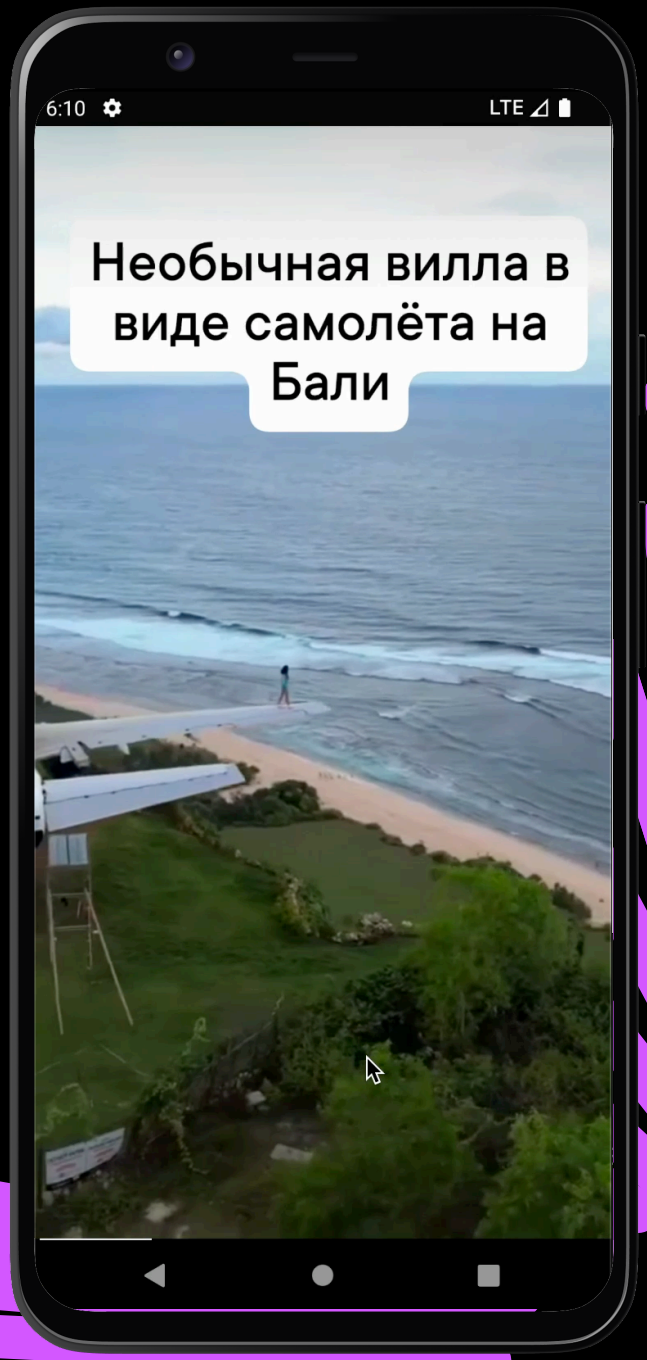
VK Клипы

Дмитрий Рычагов

- В Android-разработке уже 6 лет
- Руководжу разработкой в VK Клипах по направлению Зрителей

Проблема 😜

VK Клипы



100 мс

VK Клипы

Проблема – медленный старт видео

VK Клипы

Проблема – медленный старт видео

1

Воспринимается
как лаг
интерфейса

VK Клипы

Проблема – медленный старт видео

1

Воспринимается
как лаг
интерфейса

2

Пользователь
отвлекается
и уходит из раздела

VK Клипы

Проблема – медленный старт видео

1

Воспринимается
как лаг
интерфейса

2

Пользователь
отвлекается
и уходит из раздела

3

Меньше желания
возвращаться в
следующий раз

VK Клипы

Тернистый путь

VK Клипы

Тернистый путь

1. Проблема

VK Клипы

Тернистый путь

1. Проблема
2. Причины

VK Клипы

Тернистый путь

1. Проблема
2. Причины
3. Много плееров

VK Клипы

Тернистый путь

1. Проблема
2. Причины
3. Много плееров
4. Один плеер

VK Клипы

Тернистый путь

1. Проблема
2. Причины
3. Много плееров
4. Один плеер
5. Два плеера

VK Клипы

Тернистый путь

1. Проблема
2. Причины
3. Много плееров
4. Один плеер
5. Два плеера
6. One more thing

VK Клипы

Тернистый путь

1. Проблема
2. Причины
3. Много плееров
4. Один плеер
5. Два плеера
6. One more thing

VK Клипы

Причины медленного старта

VK Клипы

Причины медленного старта

Подгрузка первого чанка видео

VK Клипы

Причины медленного старта

Подгрузка первого чанка видео

Создание плеера и его инициализация

VK Клипы

Подготовка плеера к проигрыванию

Что в себя включает:

VK Клипы

Подготовка плеера к проигрыванию

Что в себя включает:

- Создание инстанса плеера

VK Клипы

Подготовка плеера к проигрыванию

Что в себя включает:

- Создание инстанса плеера
- Выделение кодека для проигрывания конкретного формата

VK Клипы

Подготовка плеера к проигрыванию

Что в себя включает:

- Создание инстанса плеера
- Выделение кодека для проигрывания конкретного формата
- Получение метаинформации по видео

VK Клипы

Подготовка плеера к проигрыванию

Что в себя включает:

- Создание инстанса плеера
- Выделение кодека для проигрывания конкретного формата
- Получение метаинформации по видео
- Скачивание первых чанков в буффер

VK Клипы

Подготовка плеера к проигрыванию

Что в себя включает:

- Создание инстанса плеера
- Выделение кодека для проигрывания конкретного формата
- Получение метаинформации по видео
- Скачивание первых чанков в буффер
- Связывание плеера с аутпутом кадров (TextureView)

VK Клипы

Тернистый путь

1. Проблема
2. Причины
3. Много плееров
4. Один плеер
5. Два плеера
6. One more thing

VK Клипы

Тернистый путь

1. Проблема
2. Причины
3. Много плееров
4. Один плеер
5. Два плеера
6. One more thing

VK Клипы

1 Item = 1 ExoPlayer

VK Клипы

1 Item = 1 ExoPlayer

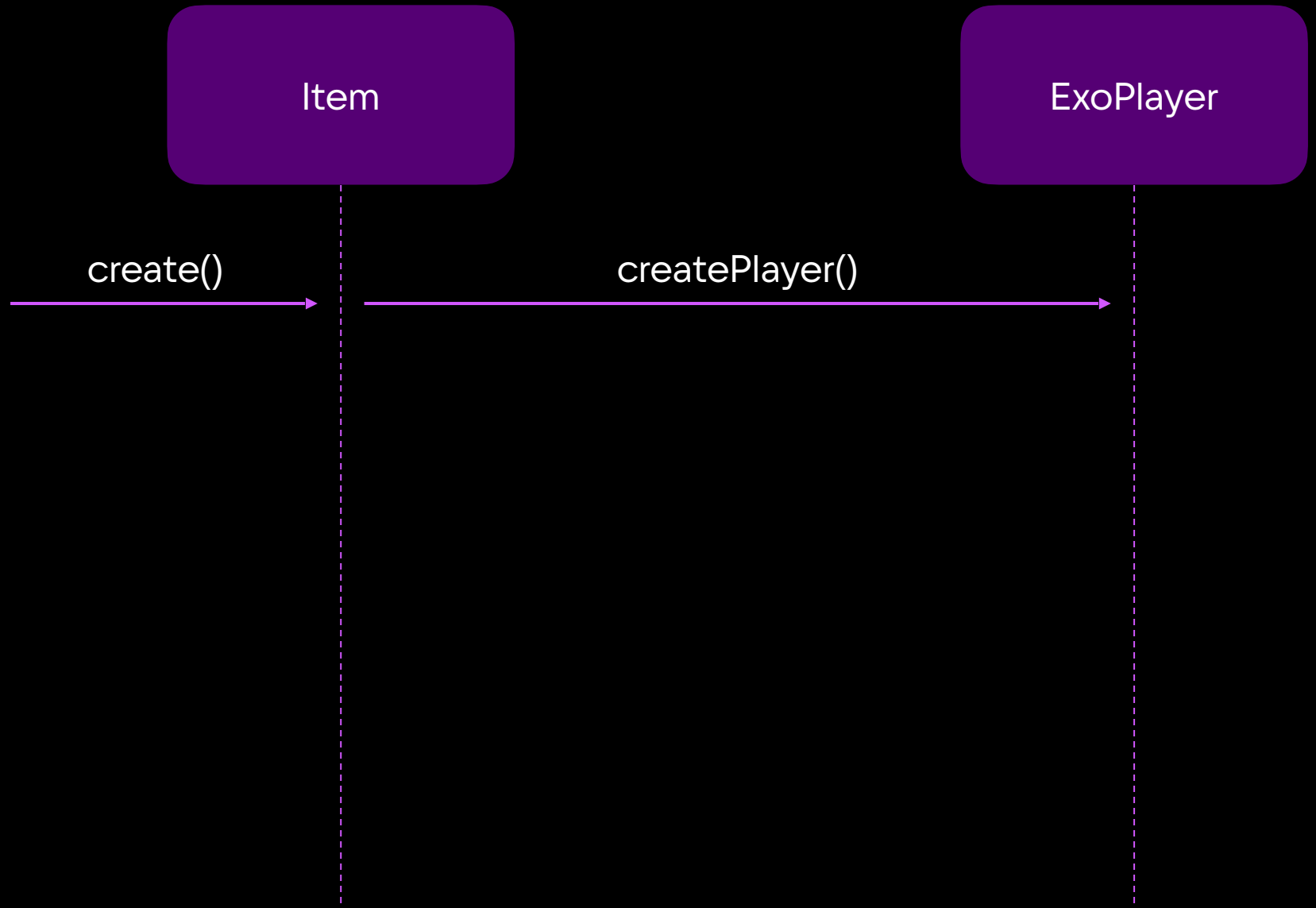
- Простая реализация

VK Клипы

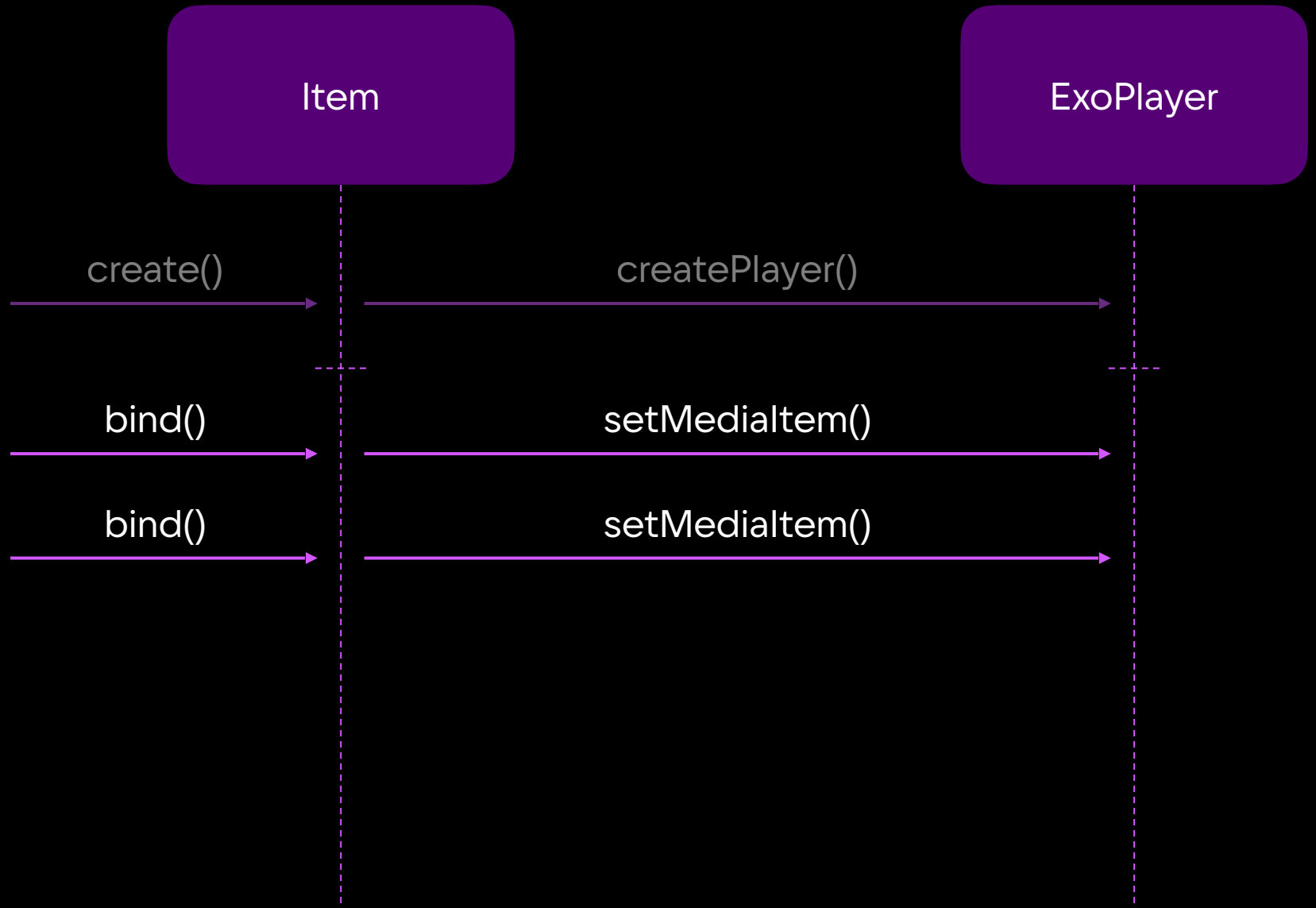
Item

ExoPlayer

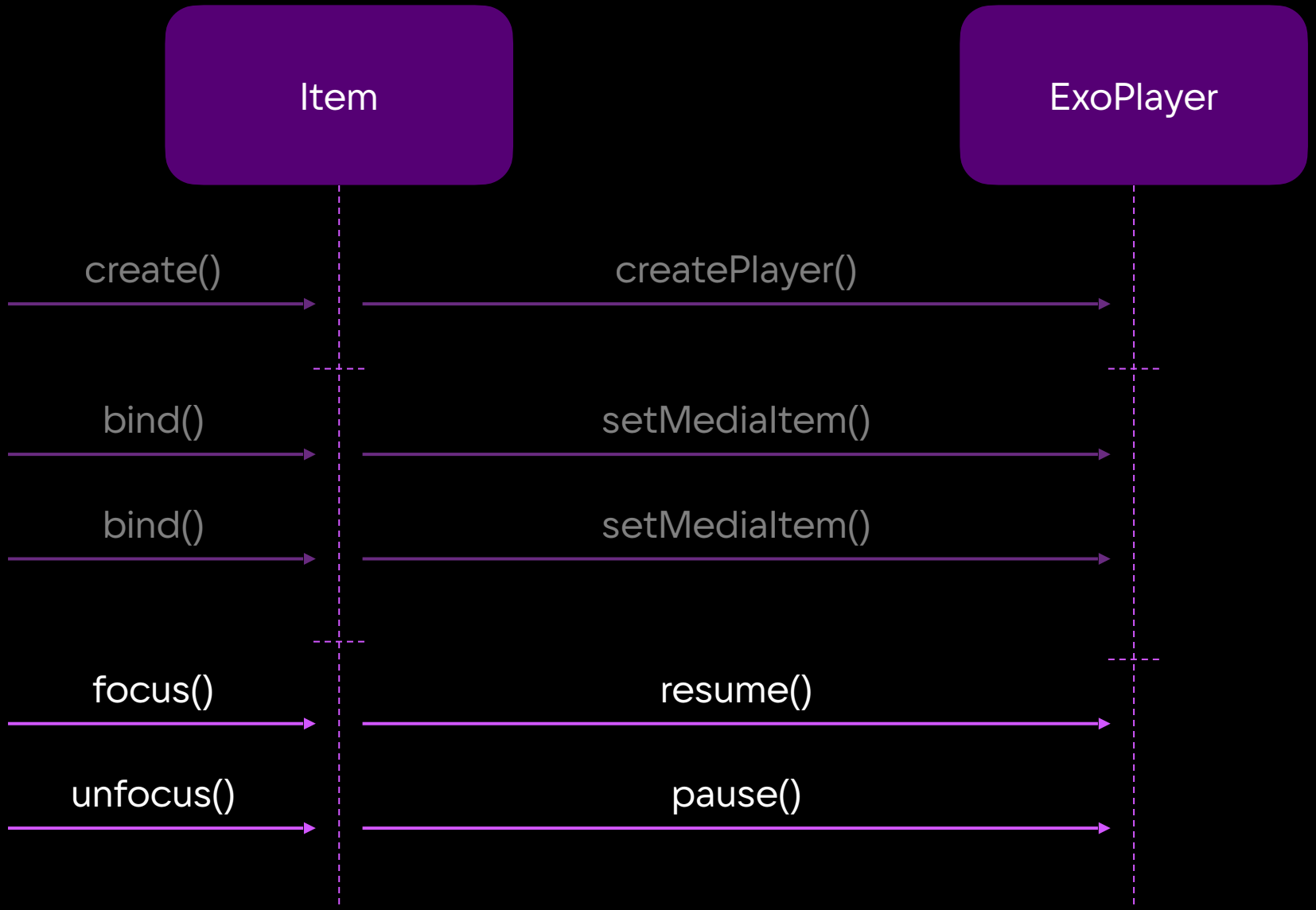
VK Клипы



VK Клипы



VK Клипы



VK Клипы

1 Item = 1 ExoPlayer

- Простая реализация
- На первый взгляд будет работать хорошо

VK Клипы

1 Item = 1 ExoPlayer

- Простая реализация
- На первый взгляд будет работать хорошо
- Но...

VK Клипы

```
E FATAL EXCEPTION: ExoPlayer:Playback
  Process: com.vk.clips.sdk.demo, PID: 9975
  java.lang.OutOfMemoryError: Failed to allocate a 64 byte allocation with 165352 free bytes and 161KB until OOM, target footprint 268435456, growth limit 268435456; giving up on allocation because <1% of heap free after GC.
      at android.media.MediaCodec.getBuffer(Native Method)
      at android.media.MediaCodec.getInputBuffer(MediaCodec.java:4076)
      at com.google.android.exoplayer2.mediacodec.AsynchronousMediaCodecAdapter.getInputBuffer(AsynchronousMediaCodecAdapter.java:212)
      at com.google.android.exoplayer2.mediacodec.MediaCodecRenderer.feedInputBuffer(MediaCodecRenderer.java:1193)
      at com.google.android.exoplayer2.mediacodec.MediaCodecRenderer.render(MediaCodecRenderer.java:784)
      at com.google.android.exoplayer2.ExoPlayerImplInternal.doSomeWork(ExoPlayerImplInternal.java:1007)
      at com.google.android.exoplayer2.ExoPlayerImplInternal.handleMessage(ExoPlayerImplInternal.java:502)
      at android.os.Handler.dispatchMessage(Handler.java:102)
      at android.os.Looper.loopOnce(Looper.java:201)
      at android.os.Looper.loop(Looper.java:288)
      at android.os.HandlerThread.run(HandlerThread.java:67)
```

VK Клипы

```
E Video codec error
    com.google.android.exoplayer2.mediacodec.MediaCodecRenderer$DecoderInitializationException: Decoder init failed: OMX.MTK.VIDEO.DECODER.VP9, Format(5, null, null, video/x-vnd.on2.vp9, vp9, 1852941, und, [1080, 1920, 30.0], [-1, -1])
        at com.google.android.exoplayer2.mediacodec.MediaCodecRenderer.maybeInitCodecWithFallback(MediaCodecRenderer.java:1021)
        at com.google.android.exoplayer2.mediacodec.MediaCodecRenderer.maybeInitCodecOrBypass(MediaCodecRenderer.java:537)
        at com.google.android.exoplayer2.mediacodec.MediaCodecRenderer.onInputFormatChanged(MediaCodecRenderer.java:1453)
        at com.google.android.exoplayer2.video.MediaCodecVideoRenderer.onInputFormatChanged(MediaCodecVideoRenderer.java:917)
        at com.google.android.exoplayer2.mediacodec.MediaCodecRenderer.readSourceOmittingSampleData(MediaCodecRenderer.java:954)
        at com.google.android.exoplayer2.mediacodec.MediaCodecRenderer.render(MediaCodecRenderer.java:769)
        at com.google.android.exoplayer2.ExoPlayerImplInternal.doSomeWork(ExoPlayerImplInternal.java:1007)
        at com.google.android.exoplayer2.ExoPlayerImplInternal.handleMessage(ExoPlayerImplInternal.java:502)
        at android.os.Handler.dispatchMessage(Handler.java:102)
        at android.os.Looper.loop(Looper.java:201)
        at android.os.HandlerThread.run(HandlerThread.java:65)
    Caused by: android.media.MediaCodec$CodecException: Failed to initialize OMX.MTK.VIDEO.DECODER.VP9, error 0xfffffff4
        at android.media.MediaCodec.native_setup(Native Method)
        at android.media.MediaCodec.<init>(MediaCodec.java:1811)
        at android.media.MediaCodec.createByCodecName(MediaCodec.java:1792)
        at com.google.android.exoplayer2.mediacodec.SynchronousMediaCodecAdapter$Factory.createCodec(SynchronousMediaCodecAdapter.java:74)
        at com.google.android.exoplayer2.mediacodec.SynchronousMediaCodecAdapter$Factory.createAdapter(SynchronousMediaCodecAdapter.java:49)
        at com.google.android.exoplayer2.mediacodec.DefaultMediaCodecAdapterFactory.createAdapter(DefaultMediaCodecAdapterFactory.java:113)
        at com.google.android.exoplayer2.mediacodec.MediaCodecRenderer.initCodec(MediaCodecRenderer.java:1099)
        at com.google.android.exoplayer2.mediacodec.MediaCodecRenderer.maybeInitCodecWithFallback(MediaCodecRenderer.java:1010)<10 more...>
```

VK Клипы

Сколько кодеков доступно?

```
private fun getMaxCodecInstances(mimeType: String): Int {
    return try {
        MediaCodecUtil
            .getDecoderInfo(mimeType, false, false)
            ?.maxSupportedInstances
            ?.takeIf { it != MediaCodecInfo.MAX_SUPPORTED_INSTANCES_UNKNOWN }
            ?.coerceAtMost(POOL_SIZE_MAX)
            ?: POOL_SIZE_MIN
    } catch (e: Exception) {
        POOL_SIZE_MIN
    }
}
```

```
private fun getMaxCodecInstances(mimeType: String): Int {
    return try {
        MediaCodecUtil
            .getDecoderInfo(mimeType, false, false)
            ?.maxSupportedInstances
            ?.takeIf { it != MediaCodecInfo.MAX_SUPPORTED_INSTANCES_UNKNOWN }
            ?.coerceAtMost(POOL_SIZE_MAX)
            ?: POOL_SIZE_MIN
    } catch (e: Exception) {
        POOL_SIZE_MIN
    }
}
```

- На Pixel 6 - по 32 кодека (vp9, hevc, h264, av1)

- На Pixel 6 - по 32 кодека (vp9, hevc, h264, av1)
- На среднем китайце - по 16 кодеков (для av1 вообще нет)

- На Pixel 6 - по 32 кодека (vp9, hevc, h264, av1)
- На среднем китайце - по 16 кодеков (для av1 вообще нет)
- На лоу-энд девайсе - ???

- На Pixel 6 - по 32 кодека (vp9, hevc, h264, av1)
- На среднем китайце - по 16 кодеков (для av1 вообще нет)
- На лоу-энд девайсе - ???

- Несколько активных приложений с видео

- На Pixel 6 - по 32 кодека (vp9, hevc, h264, av1)
- На среднем китайце - по 16 кодеков (для av1 вообще нет)
- На лоу-энд девайсе - ???

- Несколько активных приложений с видео
- Каждое приложение держит несколько кодеков

```
adb shell dumpsys media.resource_manager
```

```
adb shell dumpsys media.resource_manager

Pid: 26706
   Client:
     Id: -5476376657348616336
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
   Client:
     Id: -5476376657348598768
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
 Pid: 27080
 Pid: 27163
 Pid: 27394
 Pid: 27538
   Client:
     Id: -5476376657350010976
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
   Client:
     Id: -5476376657350007808
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
```

```
adb shell dumpsys media.resource_manager

Pid: 26706
   Client:
     Id: -5476376657348616336
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
   Client:
     Id: -5476376657348598768
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
 Pid: 27080
 Pid: 27163
 Pid: 27394
 Pid: 27538
   Client:
     Id: -5476376657350010976
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
   Client:
     Id: -5476376657350007808
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
```

```
adb shell dumpsys media.resource_manager

Pid: 26706
   Client:
     Id: -5476376657348616336
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
   Client:
     Id: -5476376657348598768
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
 Pid: 27080
 Pid: 27163
 Pid: 27394
 Pid: 27538
   Client:
     Id: -5476376657350010976
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
   Client:
     Id: -5476376657350007808
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
```

```
adb shell dumpsys media.resource_manager

Pid: 26706
   Client:
     Id: -5476376657348616336
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
   Client:
     Id: -5476376657348598768
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
 Pid: 27080
 Pid: 27163
 Pid: 27394
 Pid: 27538
   Client:
     Id: -5476376657350010976
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
   Client:
     Id: -5476376657350007808
     Name: c2.exynos.vp9.decoder
     Resources:
       non-secure-codec/video-codec:[]:1
```

VK Клипы

VK Клипы

А что по памяти?

VK Клипы

А что по памяти?

- Основное потребление памяти у ExoPlayer - это кодек и его буффер.

VK Клипы

А что по памяти?

- Основное потребление памяти у ExoPlayer - это кодек и его буффер.
- Величина буффера контролируется через LoadControl, а также зависит от выбранного видео-формата и его битрейта.

VK Клипы

Без переиспользования плееров

Словил крэш
на 11 итерации

VK Клипы

Без переиспользования плееров

Словил крэш
на 11 итерации

Переиспользуя 2 плеера

Не словил крэш
даже на 70 итерации

VK Клипы

Тернистый путь

1. Проблема
2. Причины
3. Много плееров
4. Один плеер
5. Два плеера
6. One more thing

VK Клипы

Тернистый путь

1. Проблема
2. Причины
3. Много плееров
4. Один плеер
5. Два плеера
6. One more thing

VK Клипы

1 ExoPlayer хватит всем

VK Клипы

1 ExoPlayer хватит всем

ПЛЮС

Оптимальный расход памяти
и кодеков

VK Клипы

1 ExoPlayer хватит всем

ПЛЮС

Оптимальный расход памяти
и кодеков

МИНУС

Необходимость менеджить жизненный цикл плеера + айтема, к которому плеер аттачится:

- Нельзя, чтобы плеер был приаттачен к нескольким айтемам
- Нужно правильно выбирать момент переключения фокуса

VK Клипы

Что подразумевается под переключением фокуса

VK Клипы

Что подразумевается под переключением фокуса

Смена MediaItem
внутри плеера

VK Клипы

Что подразумевается под переключением фокуса

Смена MediaItem
внутри плеера

Смена output
для кадров

VK Клипы

В какой момент менять фокус?

VK Клипы

В какой момент менять фокус?

```
private void setVideoOutputInternal(@Nullable Object videoOutput) {
  // Note: We don't turn this method into a no-op if the output is being replaced with itself so
  // as to ensure onRenderedFirstFrame callbacks are still called in this case.
  List<PlayerMessage> messages = new ArrayList<>();
  for (Renderer renderer : renderers) {
    if (renderer.getTrackType() == TRACK_TYPE_VIDEO) {
      messages.add(
          createMessageInternal(renderer)
              .setType(MSG_SET_VIDEO_OUTPUT)
              .setPayload(videoOutput)
              .send());
    }
  }
  boolean messageDeliveryTimedOut = false;
  if (this.videoOutput != null && this.videoOutput != videoOutput) {
    // We're replacing an output. Block to ensure that this output will not be accessed by the
    // renderers after this method returns.
    try {
      for (PlayerMessage message : messages) {
        message.blockUntilDelivered(detachSurfaceTimeoutMs);
      }
```

VK Клипы

В какой момент менять фокус?

```
private void setVideoOutputInternal(@Nullable Object videoOutput) {
  // Note: We don't turn this method into a no-op if the output is being replaced with itself so
  // as to ensure onRenderedFirstFrame callbacks are still called in this case.
  List<PlayerMessage> messages = new ArrayList<>();
  for (Renderer renderer : renderers) {
    if (renderer.getTrackType() == TRACK_TYPE_VIDEO) {
      messages.add(
          createMessageInternal(renderer)
              .setType(MSG_SET_VIDEO_OUTPUT)
              .setPayload(videoOutput)
              .send());
    }
  }
  boolean messageDeliveryTimedOut = false;
  if (this.videoOutput != null && this.videoOutput != videoOutput) {
    // We're replacing an output. Block to ensure that this output will not be accessed by the
    // renderers after this method returns.
    try {
      for (PlayerMessage message : messages) {
        message.blockUntilDelivered(detachSurfaceTimeoutMs);
      }
```

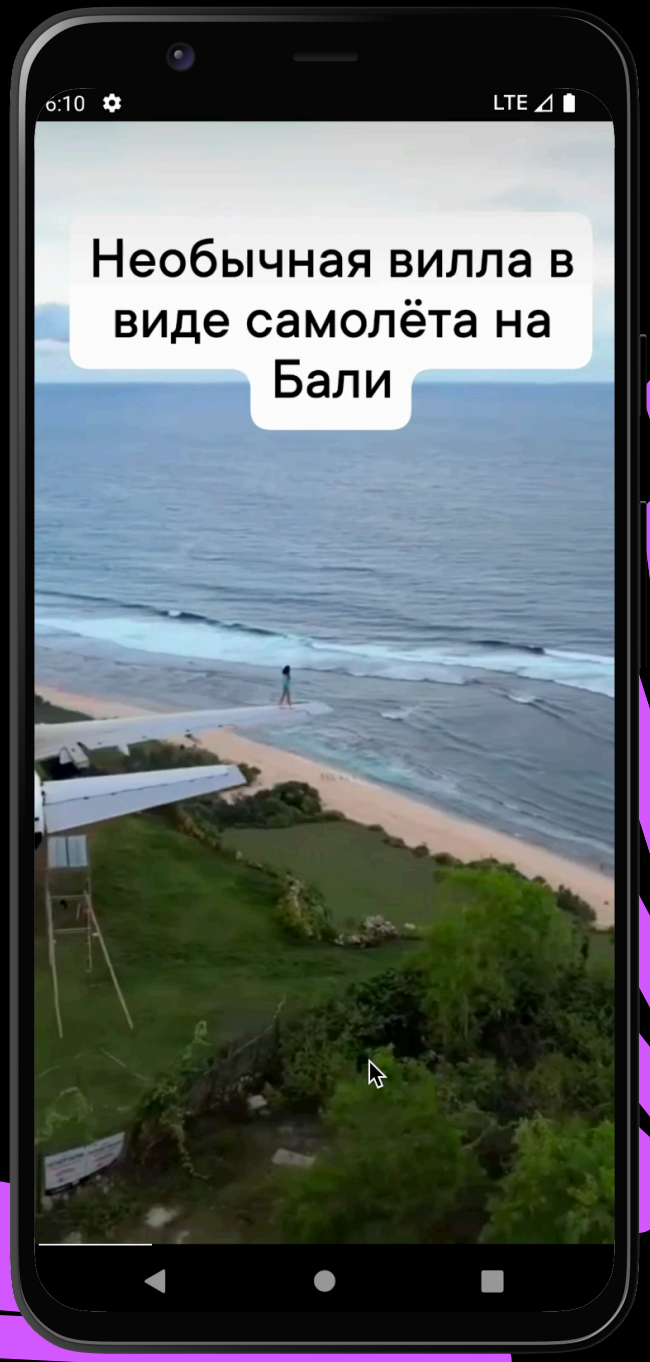
```
override fun onScrollStateChanged(recyclerView: RecyclerView, newState: Int) {
    if (newState == RecyclerView.SCROLL_STATE_IDLE) {
        val holder = getTargetPlayHolder(recyclerView) ?: return
        if (holder != currentPlayHolder) {
            currentPlayHolder?.pause()
            currentPlayHolder = holder
        }

        currentPlayHolder?.play()
    }
}
```

```
override fun onScrollStateChanged(recyclerView: RecyclerView, newState: Int) {
    if (newState == RecyclerView.SCROLL_STATE_IDLE) {
        val holder = getTargetPlayHolder(recyclerView) ?: return
        if (holder != currentPlayHolder) {
            currentPlayHolder?.pause()
            currentPlayHolder = holder
        }

        currentPlayHolder?.play()
    }
}
```

```
private fun getTargetPlayHolder(recyclerView: RecyclerView): ClipsRecyclerViewPlayer? {
    val layoutManager = recyclerView.layoutManager as? LinearLayoutManager ?: return null
    val positionFirst = layoutManager.findFirstVisibleItemPosition()
    val positionLast = layoutManager.findLastVisibleItemPosition()

    val targetPosition = if (positionFirst == positionLast) positionFirst else null

    return targetPosition?.let {
        recyclerView.findViewHolderForAdapterPosition(it) as? ClipsRecyclerViewPlayer
    }
}
```

VK Клипы

Что в итоге получаем?



VK Клипы

В чем дело?

VK Клипы

В чем дело?

В зависимости
от производителя,
ресайклер переходит в
стейт IDLE за некоторое
время до реального
конца скролла

VK Клипы

В чем дело?

В зависимости от производителя, ресайклер переходит в стейт IDLE за некоторое время до реального конца скролла

Из-за этого смена сюрфы плеера происходит во время скролла, а мы уже знаем, что эта операция требует синхронизации на главном потоке

VK Клипы

В чем дело?

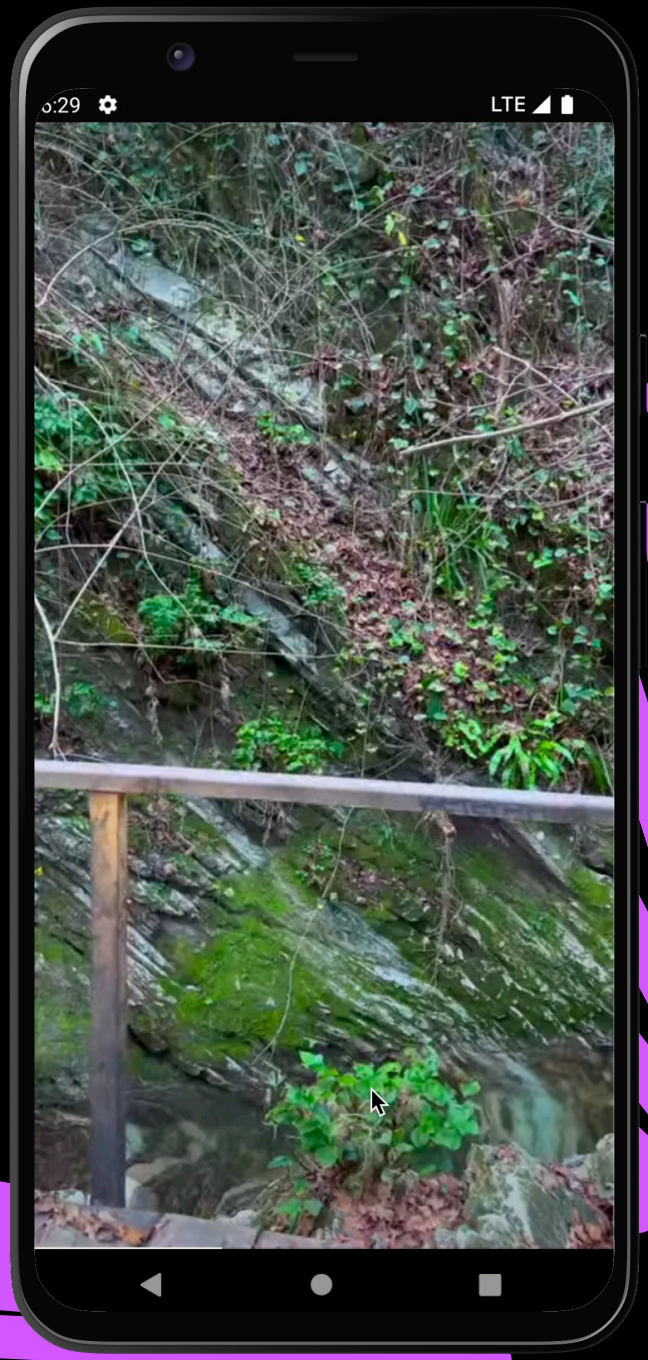
В зависимости от производителя, ресайклер переходит в стейт IDLE за некоторое время до реального конца скролла

Из-за этого смена сюрфы плеера происходит во время скролла, а мы уже знаем, что эта операция требует синхронизации на главном потоке

Решение - делаем post :)

VK Клипы

Уже лучше!



VK Клипы

Тернистый путь

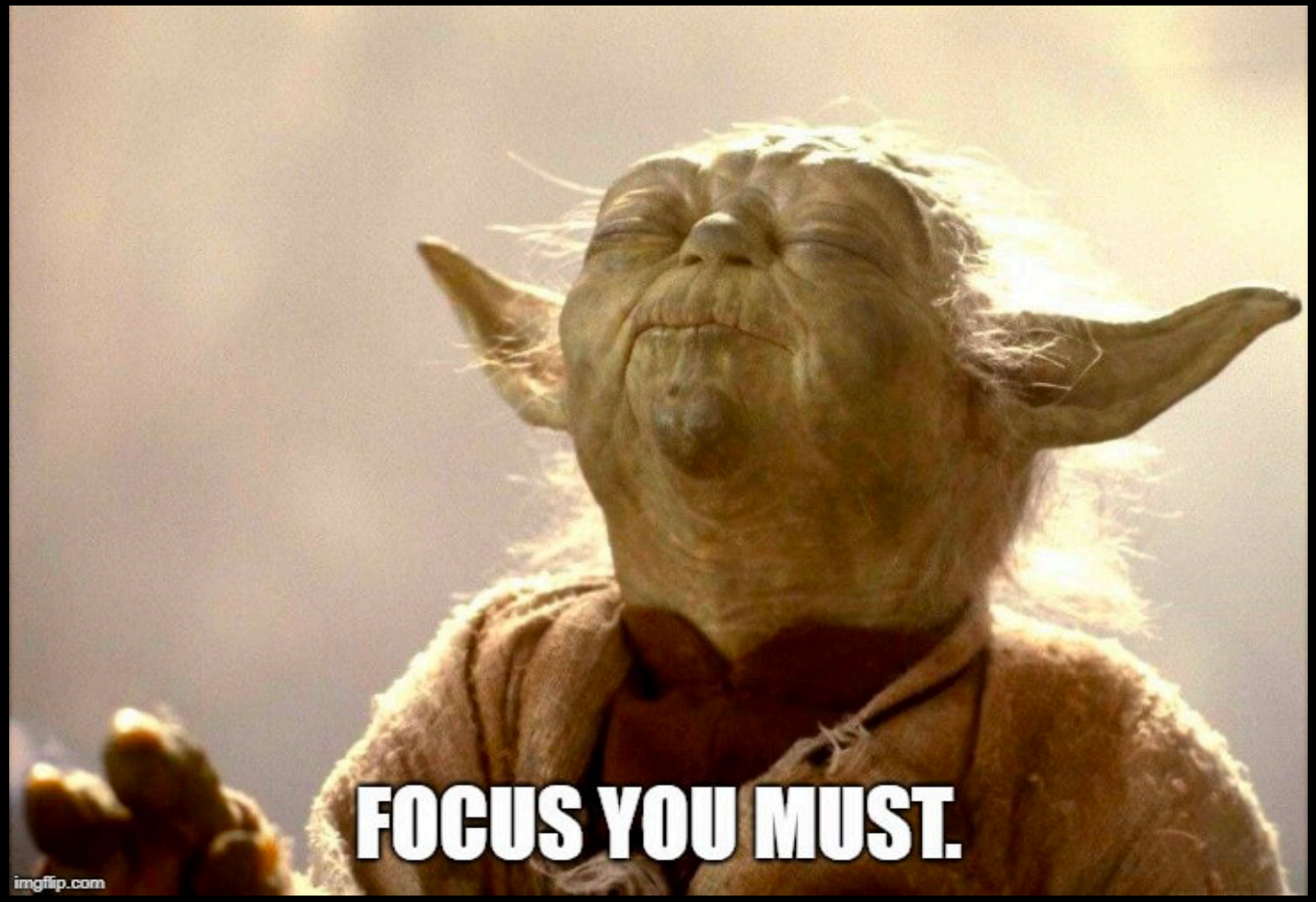
1. Проблема
2. Причины
3. Много плееров
4. Один плеер
5. Два плеера
6. One more thing

VK Клипы

Тернистый путь

1. Проблема
2. Причины
3. Много плееров
4. Один плеер
5. Два плеера
6. One more thing

VK Клипы



VK Клипы

Пул из двух плееров

VK Клипы

Пул из двух плееров

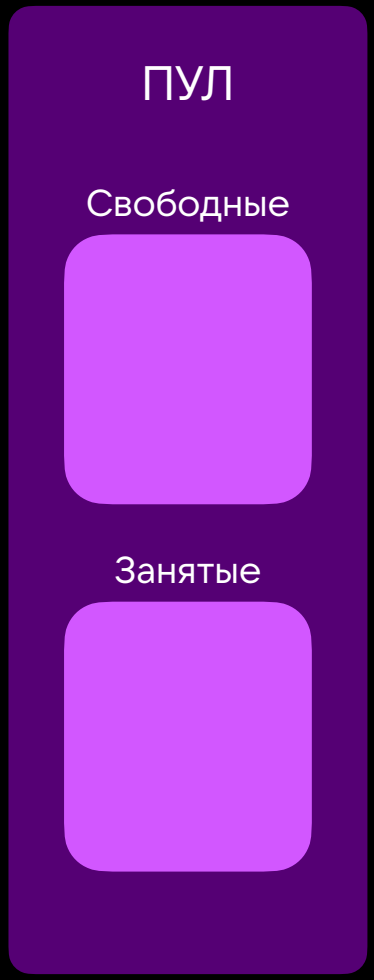
Правильно обрабатываем
жизненный цикл плееров

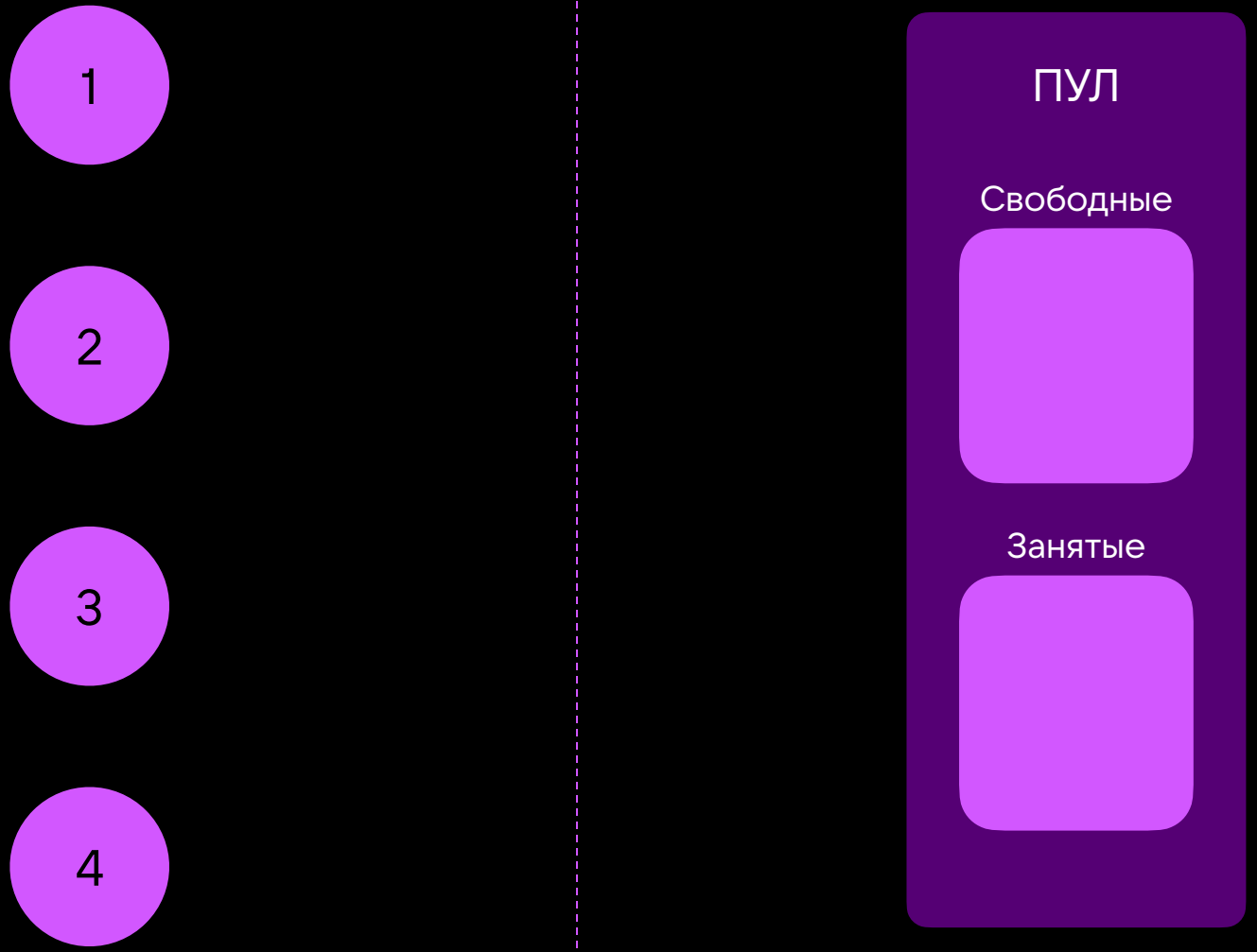
VK Клипы

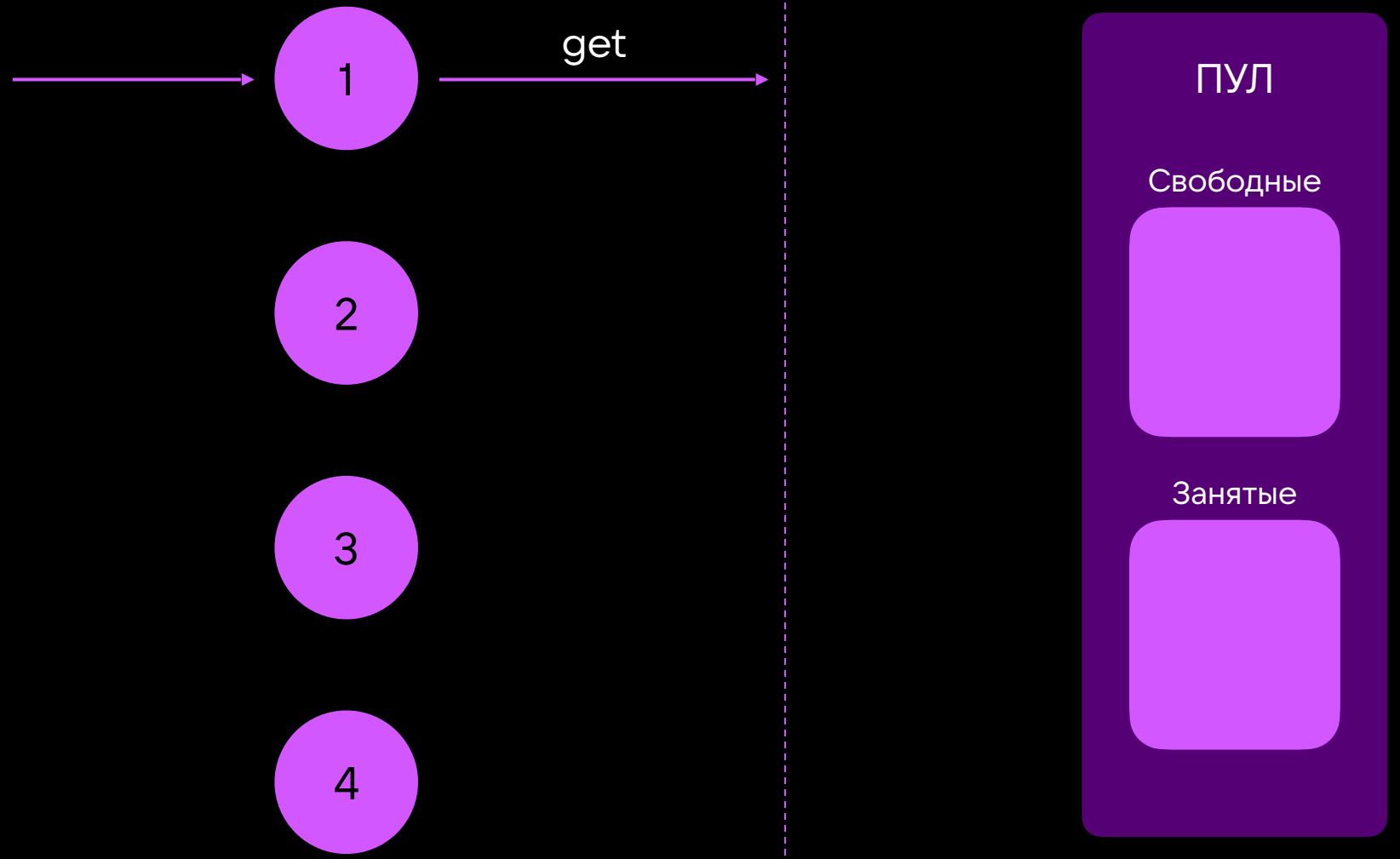
Пул из двух плееров

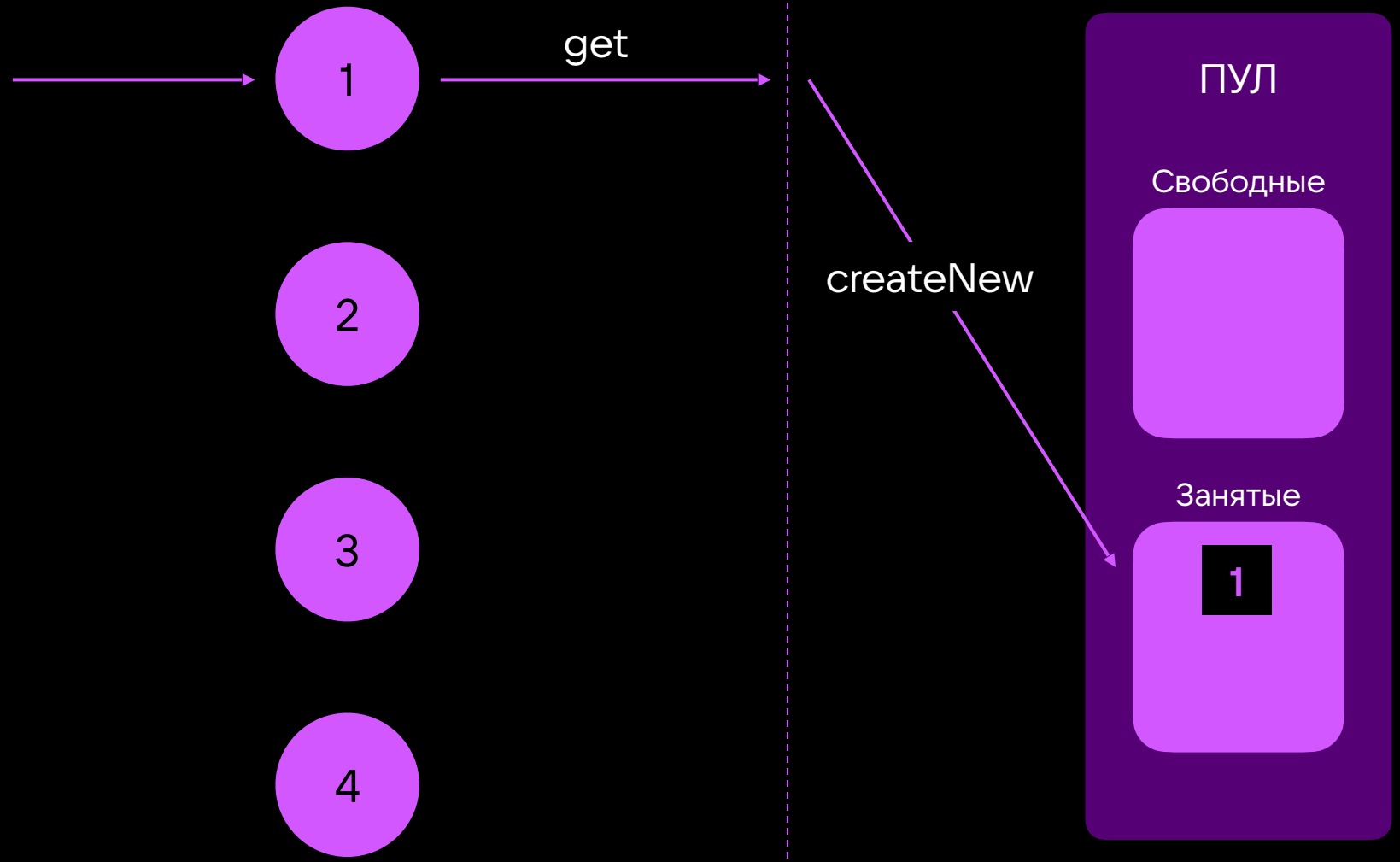
Правильно обрабатываем жизненный цикл плееров

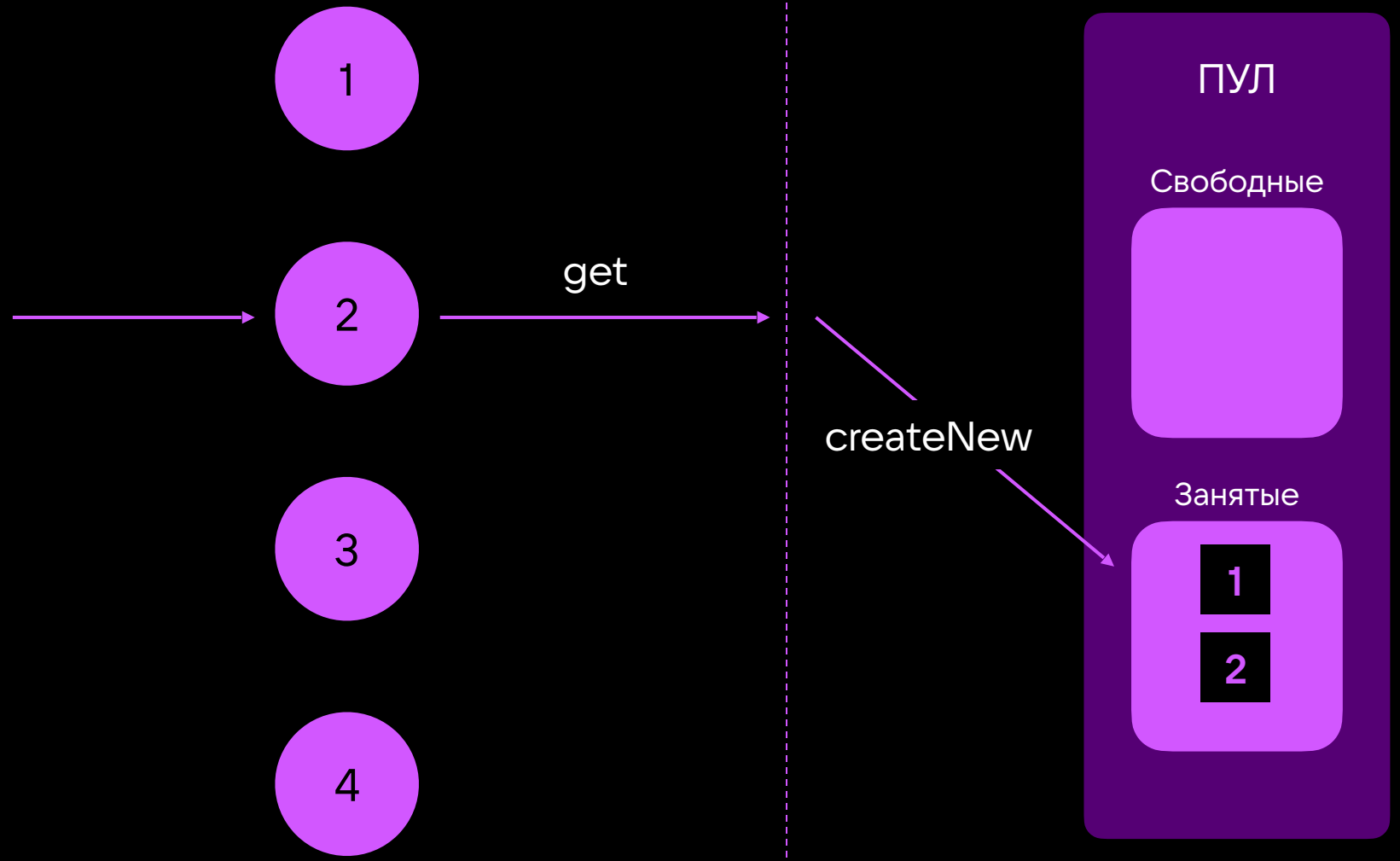
Попробуем добавить еще один?

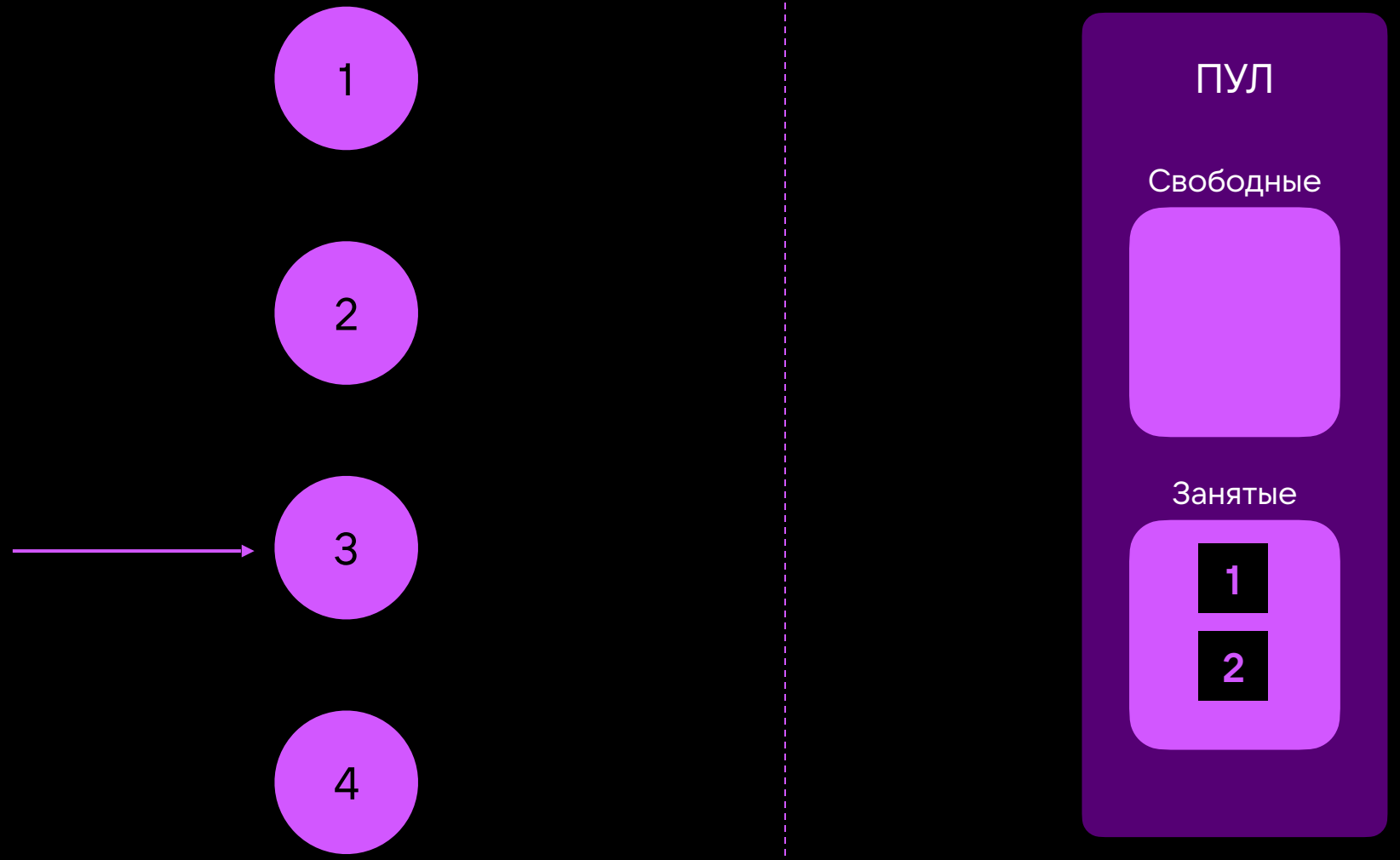

ПУЛ
Свободные
Занятые

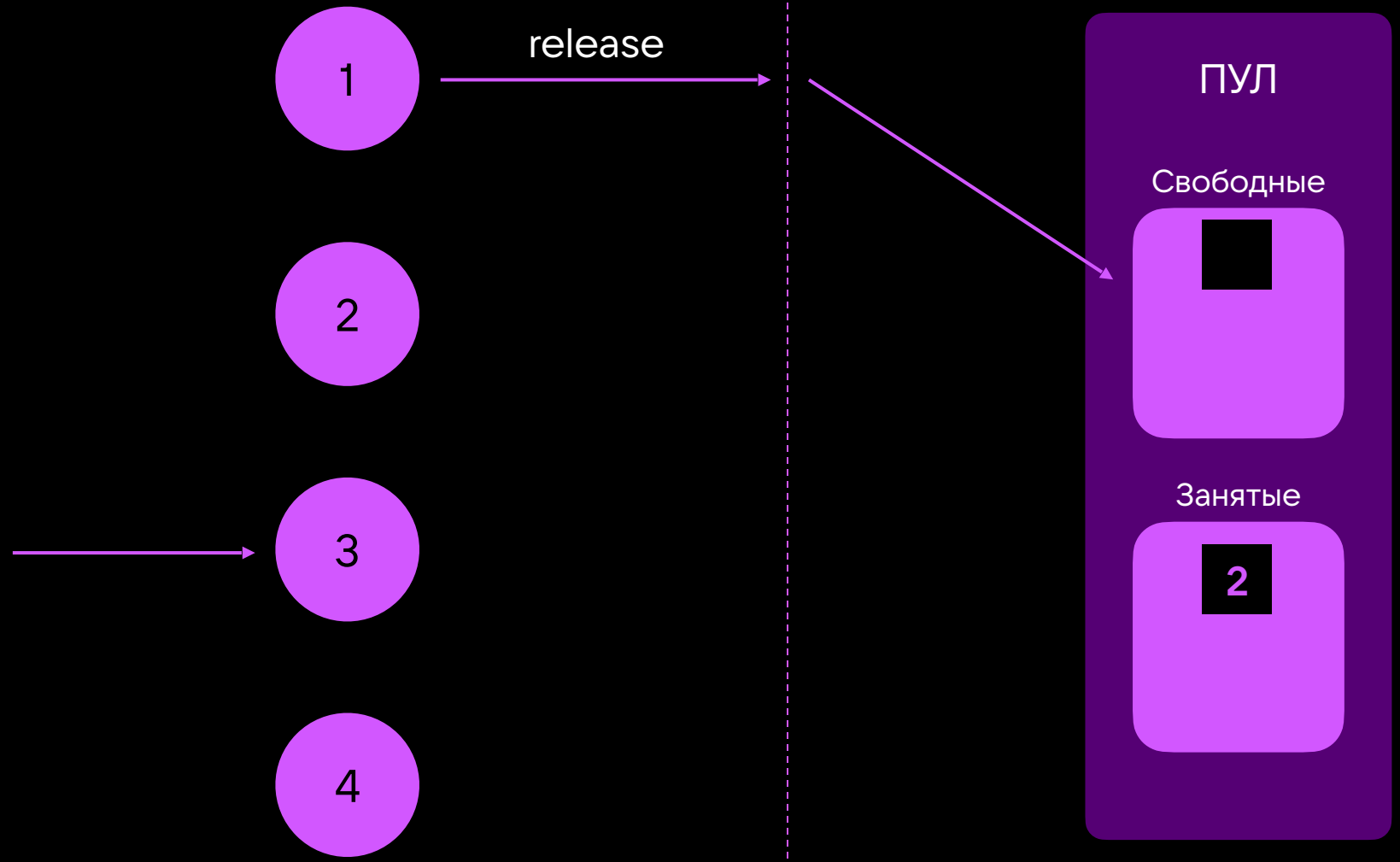

1
2
3
4
ПУЛ
Свободные
Занятые

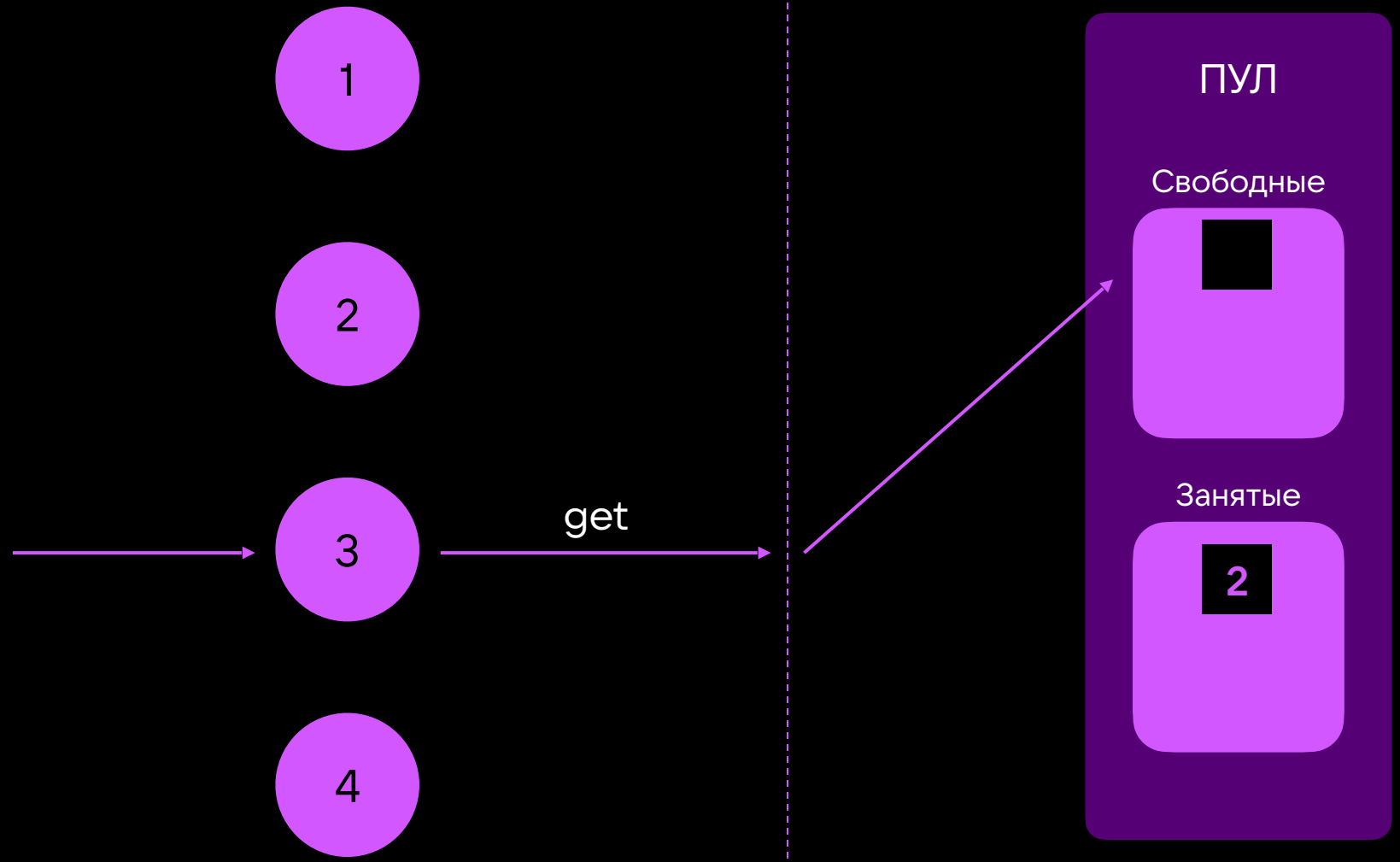

1
get
2
3
4
ПУЛ
Свободные
Занятые

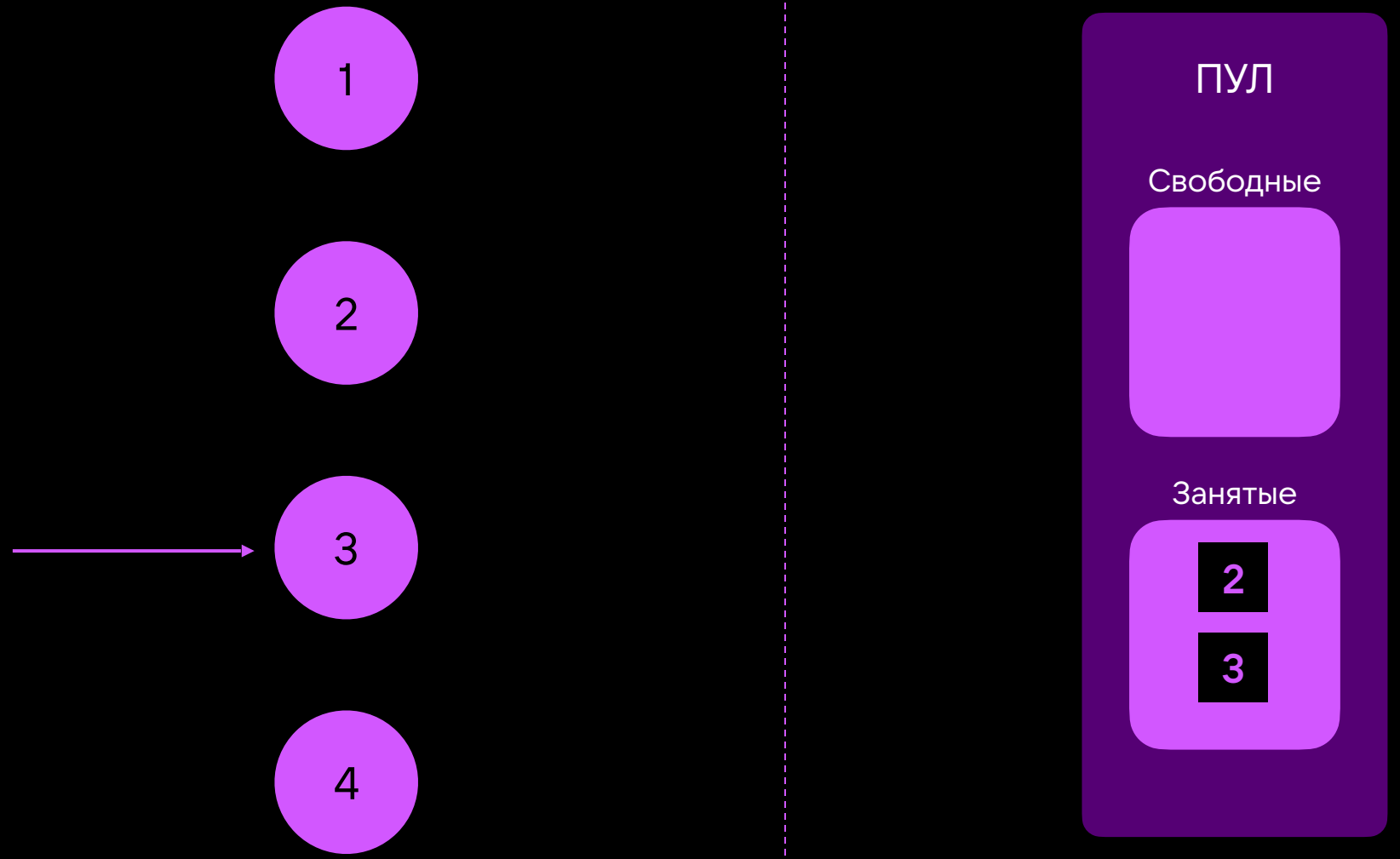

1
2
3
4
get
createNew
ПУЛ
Свободные
Занятые
1


1
2
3
4
get
createNew
ПУЛ
Свободные
Занятые
1
2


1
2
3
4
ПУЛ
Свободные
Занятые
1
2

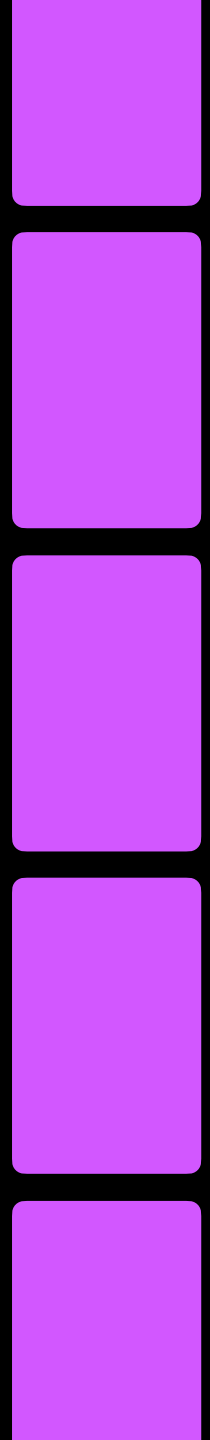

1
release
2
3
4
ПУЛ
Свободные
Занятые
2

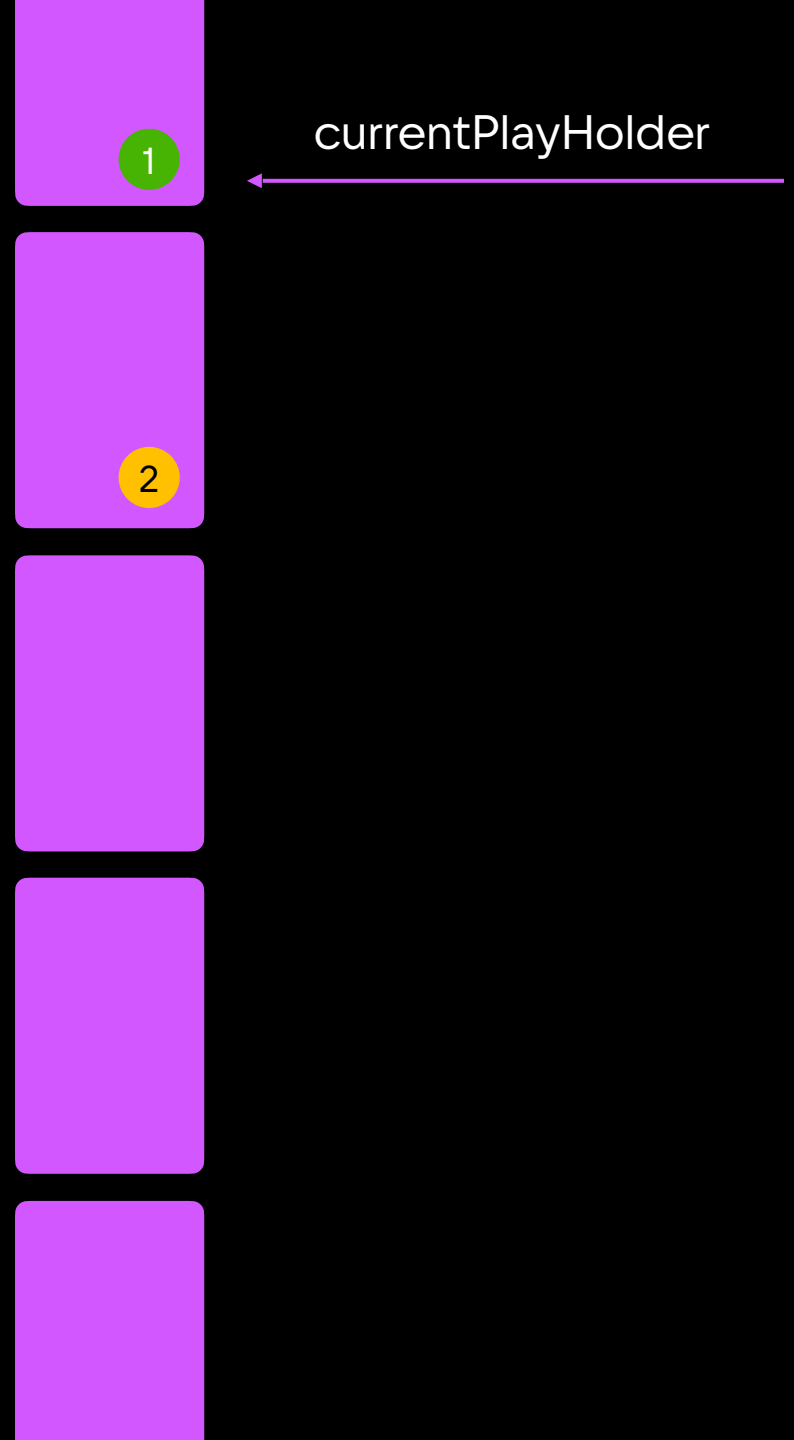

1
2
3
4
get
ПУЛ
Свободные
Занятые
2

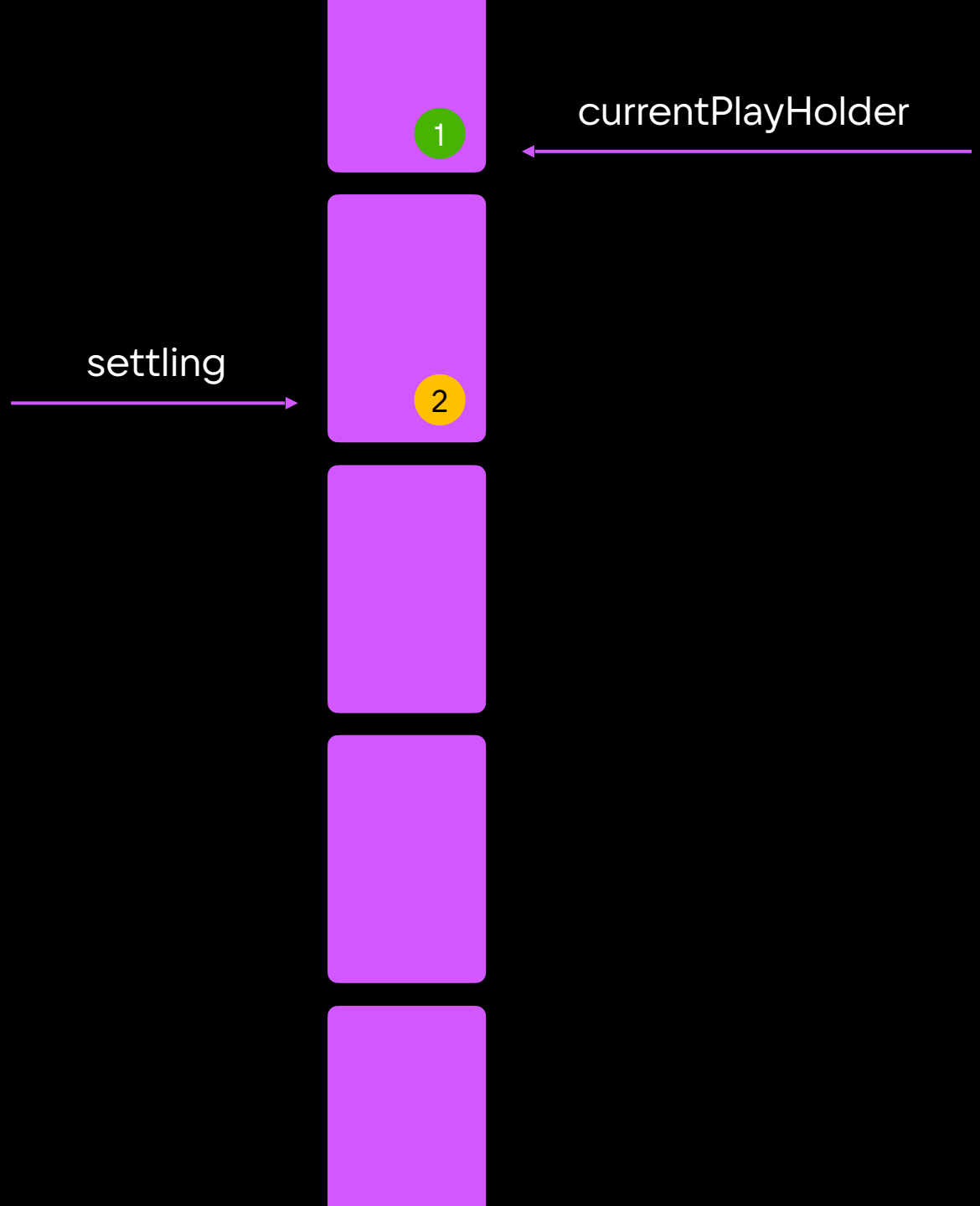

1
2
3
4
ПУЛ
Свободные
Занятые
2
3

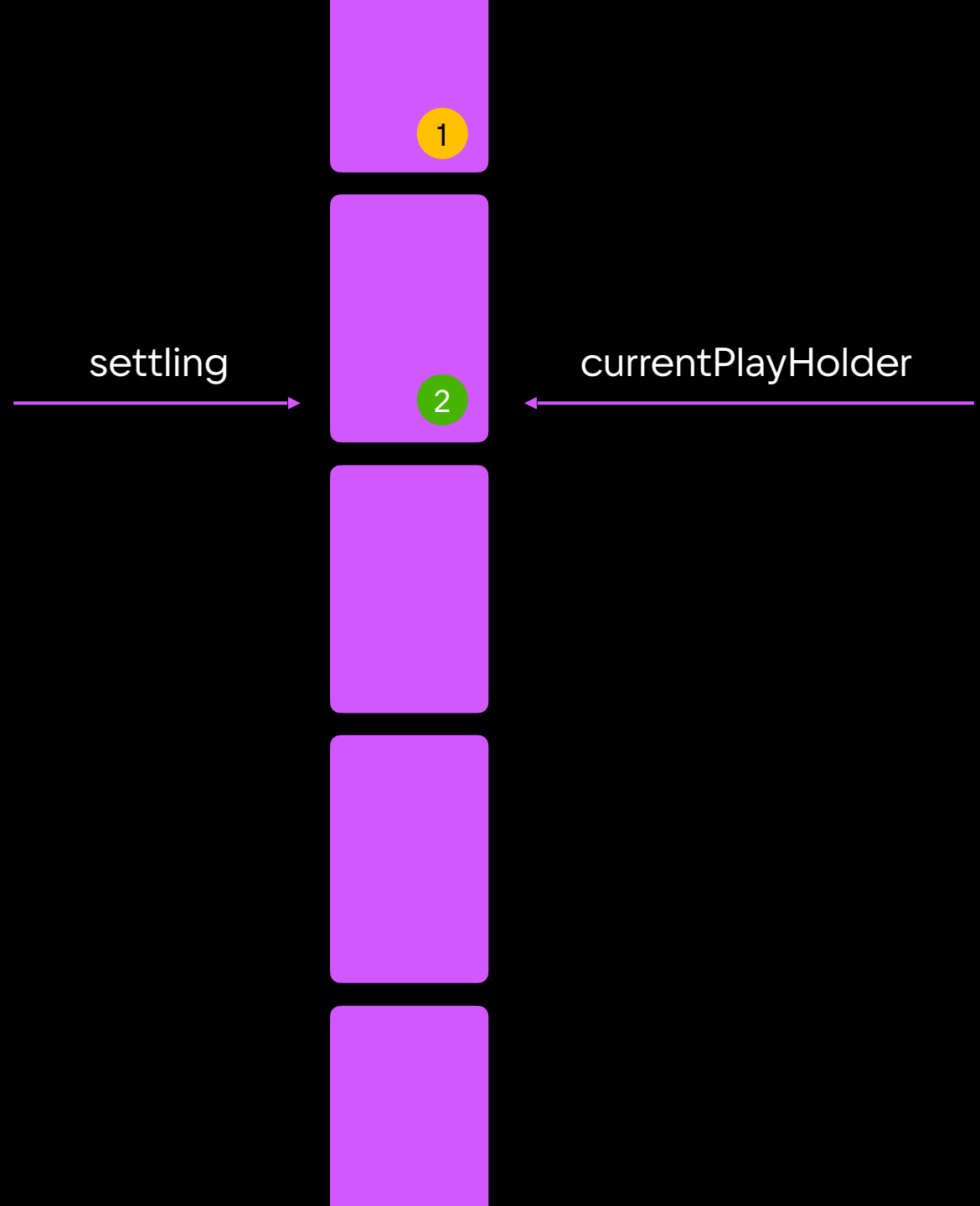
```
override fun onScrollStateChanged(recyclerView: RecyclerView, newState: Int) {
    if (newState != RecyclerView.SCROLL_STATE_IDLE) {
        cancelPrepareNext()
        cancelDelayedPlay()
    }

    when (newState) {
        RecyclerView.SCROLL_STATE_IDLE -> {
            startPlay(recyclerView, isSettling = false)
            schedulePrepareNext(recyclerView)
        }
        RecyclerView.SCROLL_STATE_SETTLING -> {
            startPlay(recyclerView, isSettling = true)
        }
    }
}
```

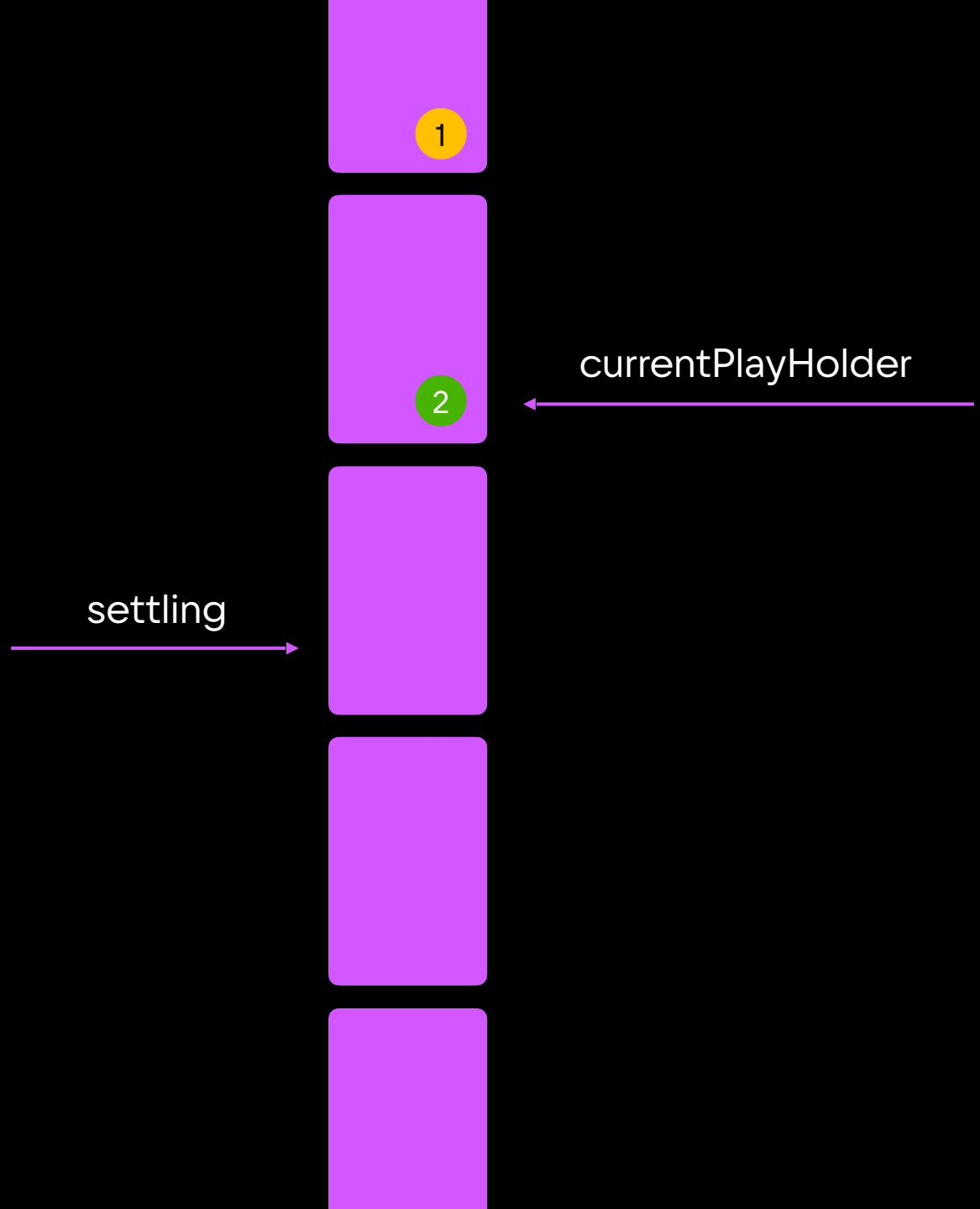
```
override fun onScrollStateChanged(recyclerView: RecyclerView, newState: Int) {
    if (newState != RecyclerView.SCROLL_STATE_IDLE) {
        cancelPrepareNext()
        cancelDelayedPlay()
    }

    when (newState) {
        RecyclerView.SCROLL_STATE_IDLE -> {
            startPlay(recyclerView, isSettling = false)
            schedulePrepareNext(recyclerView)
        }
        RecyclerView.SCROLL_STATE_SETTLING -> {
            startPlay(recyclerView, isSettling = true)
        }
    }
}
```

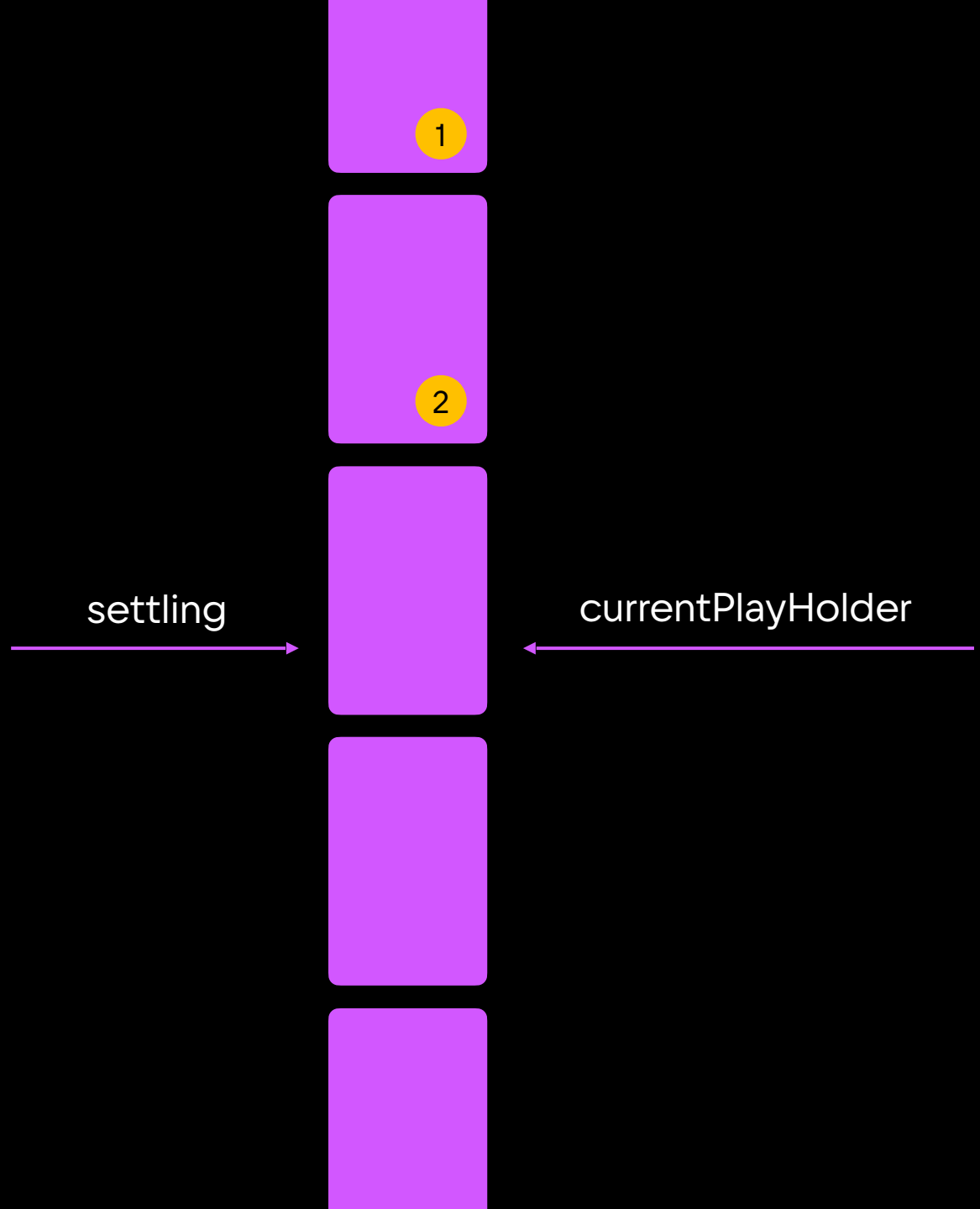


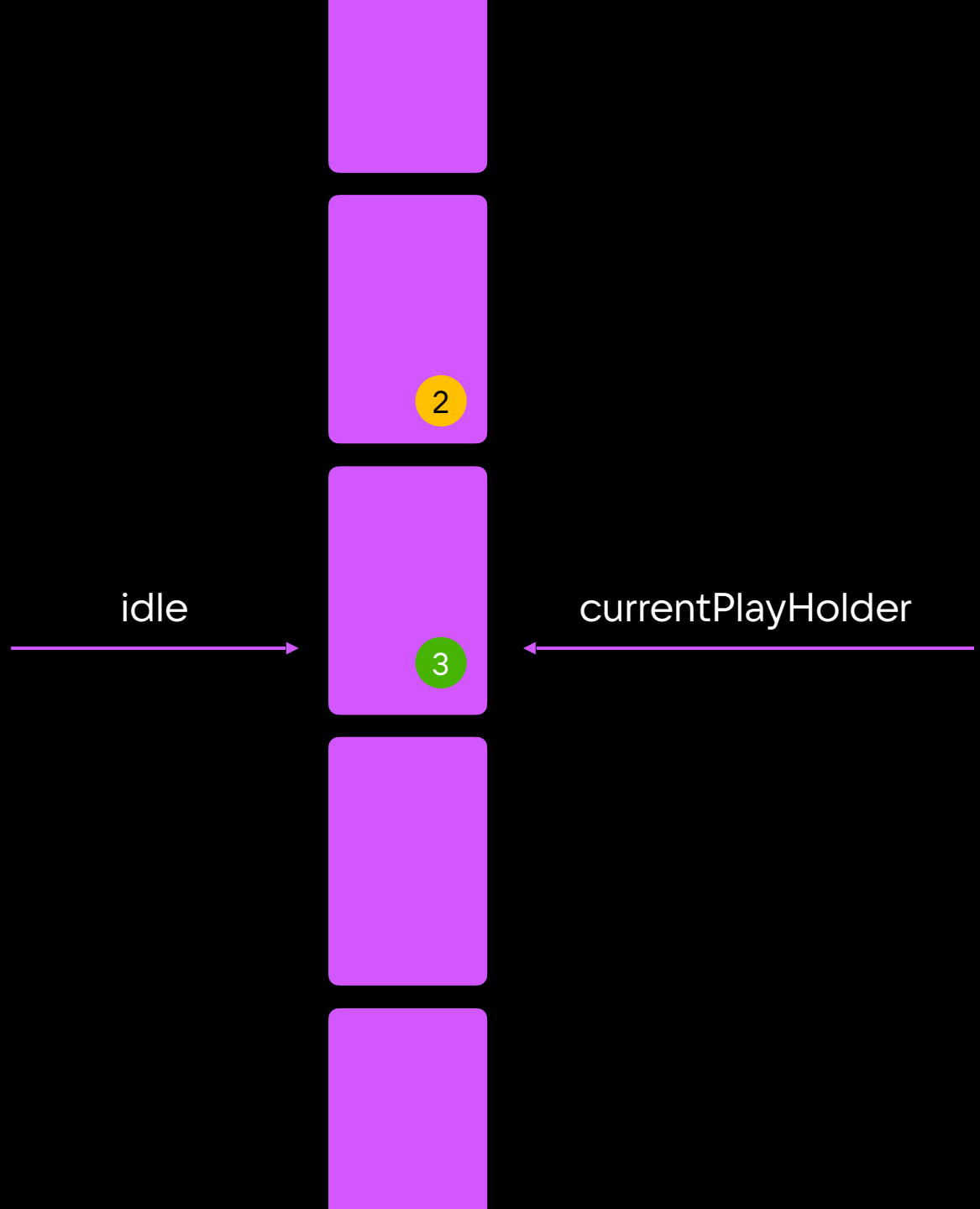

1
currentPlayHolder
2


currentPlayHolder
1
settling
2


1
settling
2
currentPlayHolder


1
2
currentPlayHolder
settling

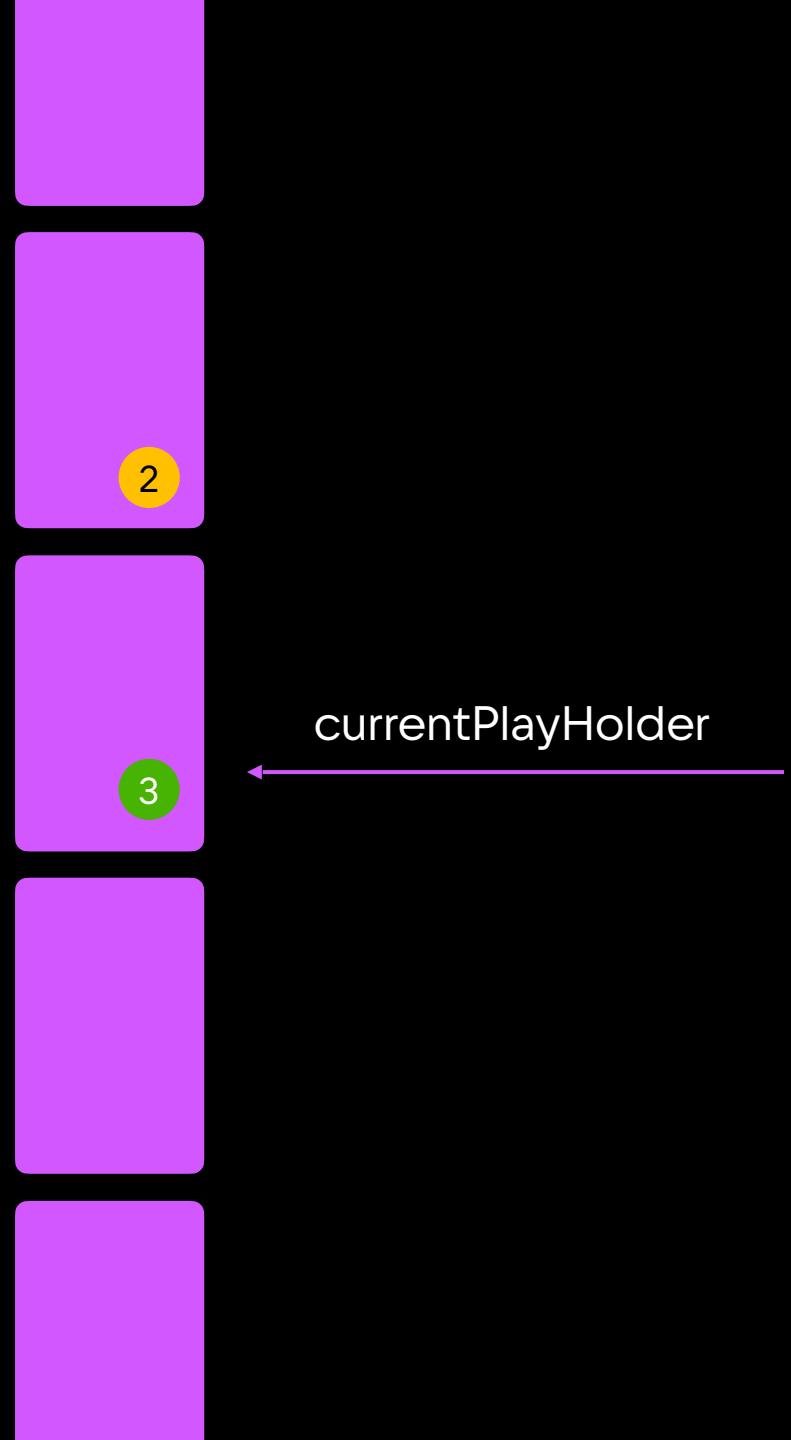

1
2
settling
currentPlayHolder

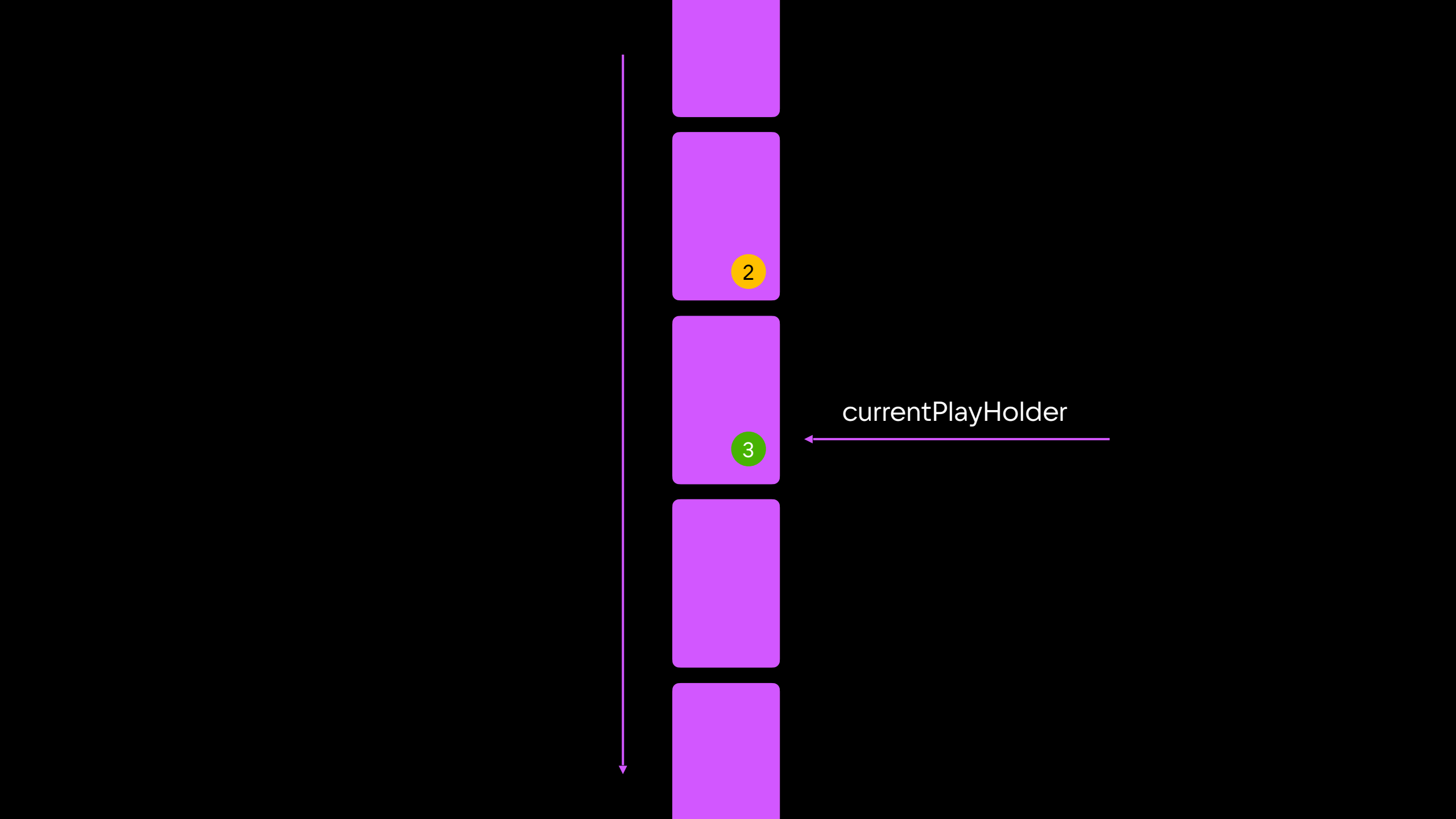

2
idle
currentPlayHolder
3

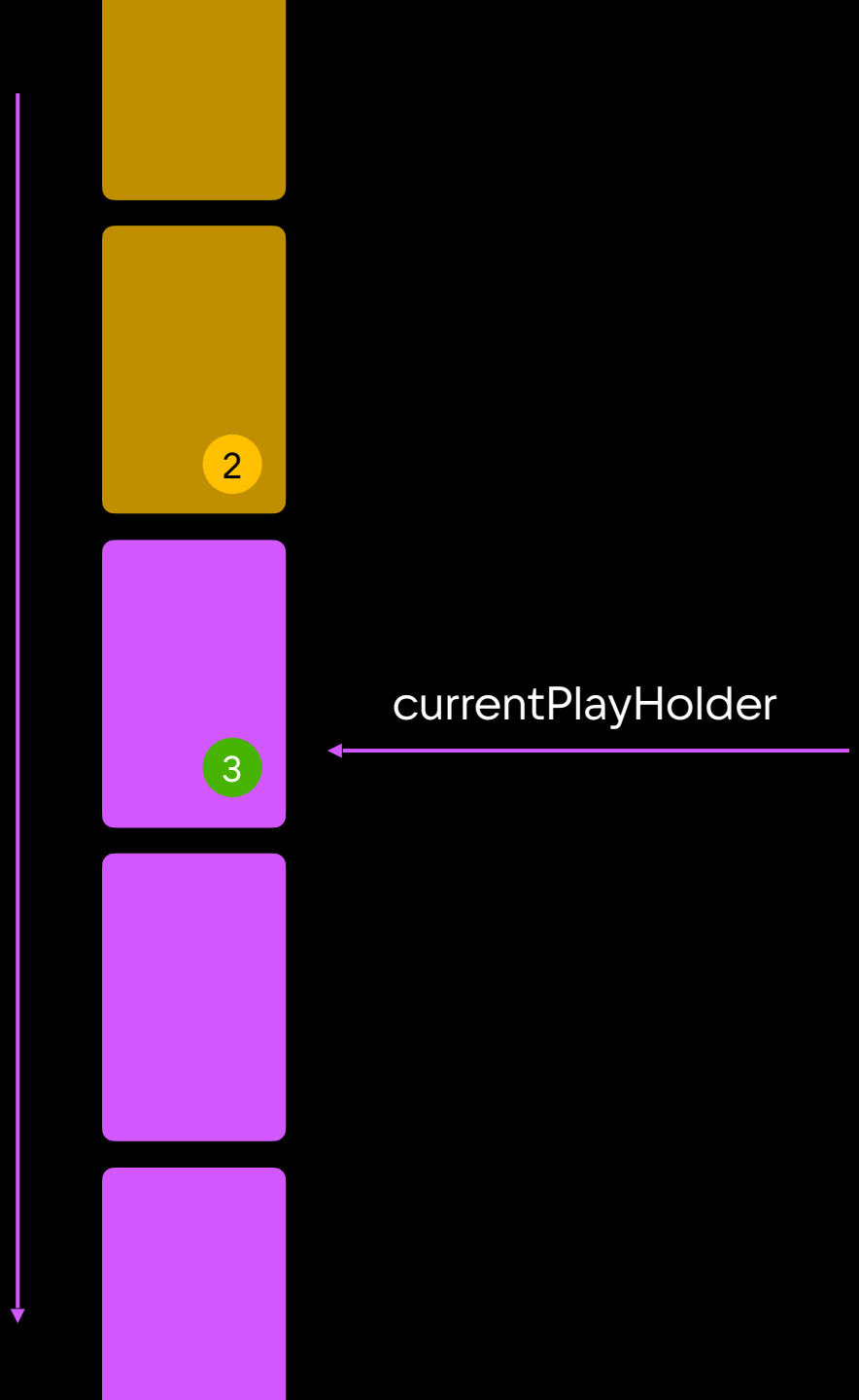
```
private fun startPlay(recyclerView: RecyclerView, isSettling: Boolean) {
    val holder = getTargetPlayHolder(recyclerView, isSettling) ?: return

    if (holder != currentPlayHolder) {
        currentPlayHolder?.pause()
        currentPlayHolder = holder
    }

    val isHolderPrepared = holder.isPrepared()

    when {
        isHolderPrepared -> currentPlayHolder?.play()
        isSettling -> return
        else -> playDelayed()
    }
}
```

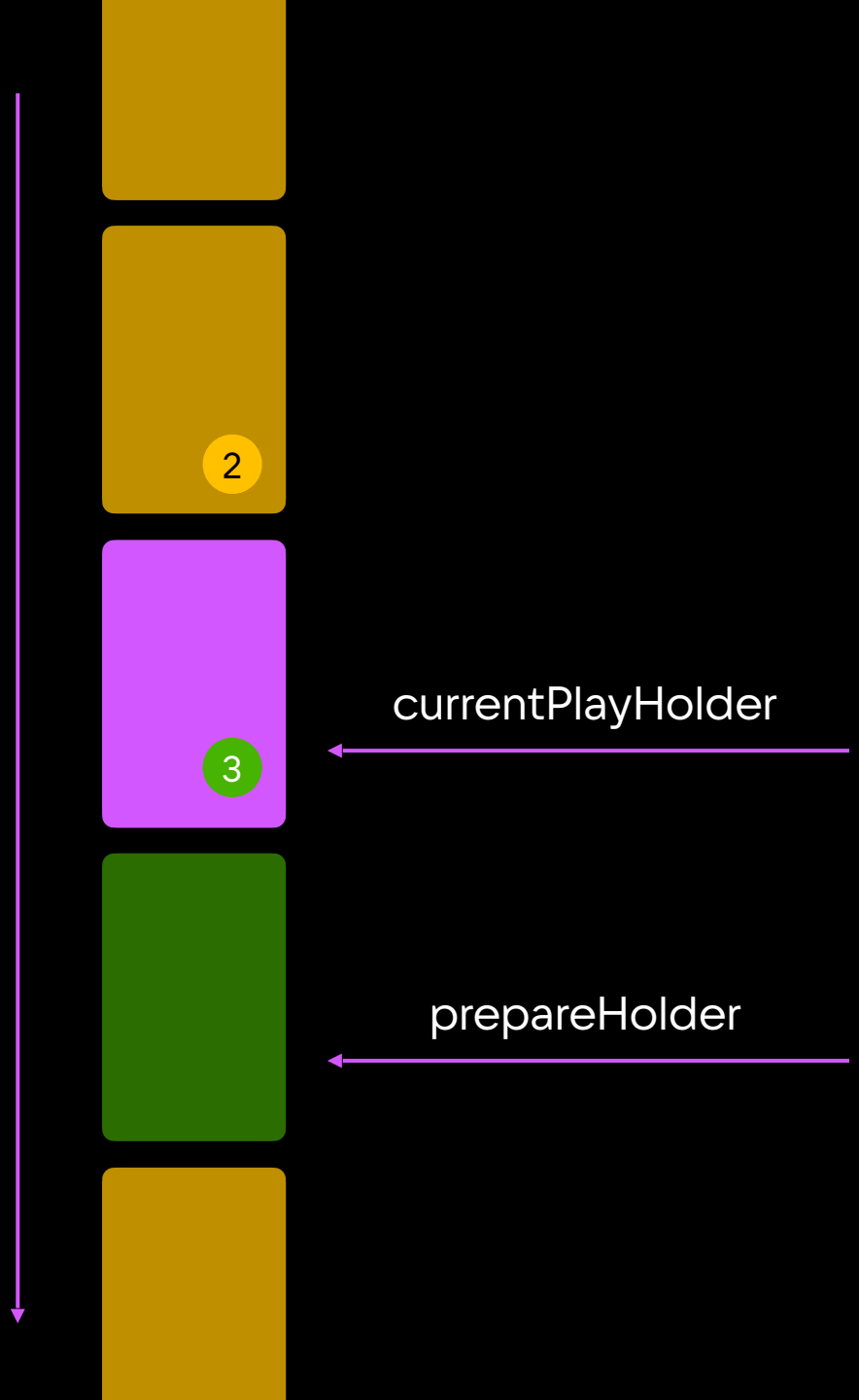
```
private fun startPlay(recyclerView: RecyclerView, isSettling: Boolean) {
    val holder = getTargetPlayHolder(recyclerView, isSettling) ?: return

    if (holder != currentPlayHolder) {
        currentPlayHolder?.pause()
        currentPlayHolder = holder
    }

    val isHolderPrepared = holder.isPrepared()

    when {
        isHolderPrepared -> currentPlayHolder?.play()
        isSettling -> return
        else -> playDelayed()
    }
}
```

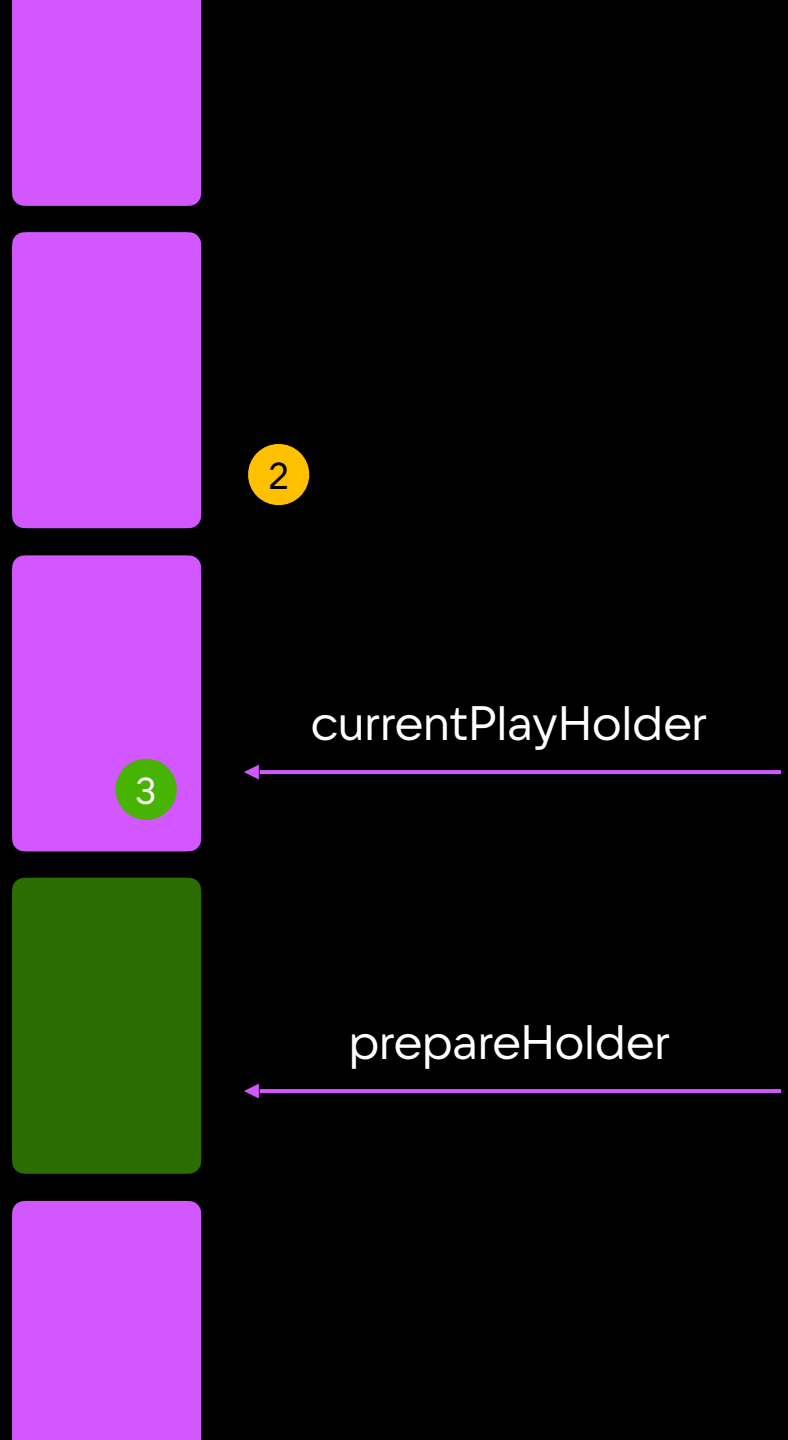
```
private fun startPlay(recyclerView: RecyclerView, isSettling: Boolean) {
    val holder = getTargetPlayHolder(recyclerView, isSettling) ?: return

    if (holder != currentPlayHolder) {
        currentPlayHolder?.pause()
        currentPlayHolder = holder
    }

    val isHolderPrepared = holder.isPrepared()

    when {
        isHolderPrepared -> currentPlayHolder?.play()
        isSettling -> return
        else -> playDelayed()
    }
}
```

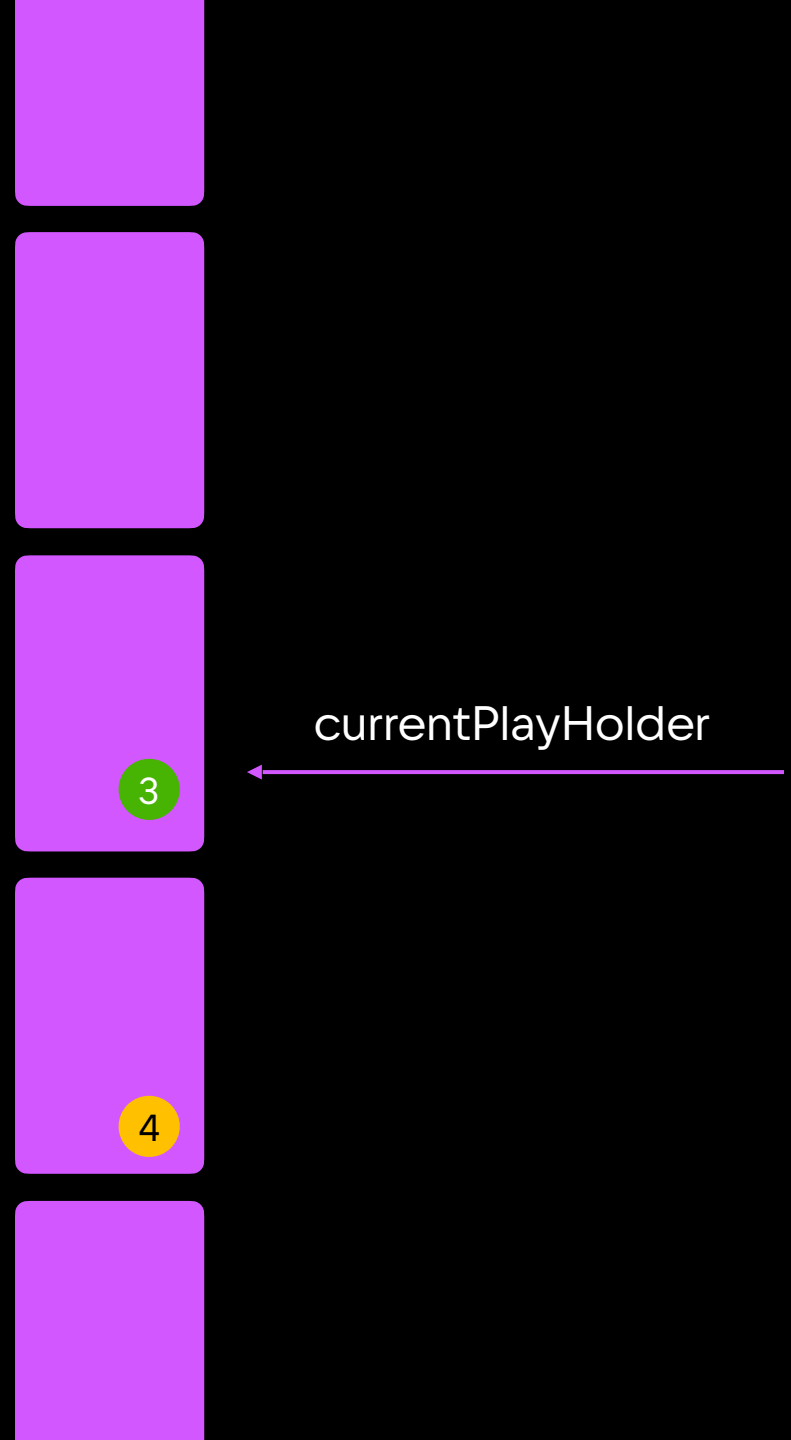

2
3
currentPlayHolder


2
3
currentPlayHolder


2
currentPlayHolder
3


2
3
currentPlayHolder
prepareHolder


2
currentPlayHolder
3
prepareHolder


currentPlayHolder
3
4

```
private fun schedulePrepareNext(recyclerView: RecyclerView) {
    val layoutManager = recyclerView.layoutManager as? LinearLayoutManager ?: return
    val playingView = (currentPlayHolder as? RecyclerView.ViewHolder)?.itemView ?: return
    val playingPosition = recyclerView.getChildAdapterPosition(playingView)

    val releaseViewHolders = mutableListOf<ClipsRecyclerViewPlayer>()
    var prepareViewHolder: ClipsRecyclerViewPlayer? = null

    recyclerView.children().forEach { child ->
        val childPosition = layoutManager.getPosition(child)
        val childViewHolder = recyclerView.getChildViewHolder(child)
        if (childViewHolder is ClipsRecyclerViewPlayer) {
            when {
                childPosition == playingPosition -> Unit
                childPosition > playingPosition && prepareViewHolder == null -> prepareViewHolder = childViewHolder
                else -> releaseViewHolders.add(childViewHolder)
            }
        }
    }

    val prepareRunnable = Runnable {
        releaseViewHolders.forEach { it.release() }
        prepareViewHolder?.prepare()
    }

    currentPrepareRunnable?.let { handler.removeCallbacks(it) }
    currentPrepareRunnable = prepareRunnable.also { handler.post(it) }
}
```

```
private fun schedulePrepareNext(recyclerView: RecyclerView) {
    val layoutManager = recyclerView.layoutManager as? LinearLayoutManager ?: return
    val playingView = (currentPlayHolder as? RecyclerView.ViewHolder)?.itemView ?: return
    val playingPosition = recyclerView.getChildAdapterPosition(playingView)

    val releaseViewHolders = mutableListOf<ClipsRecyclerViewPlayer>()
    var prepareViewHolder: ClipsRecyclerViewPlayer? = null

    recyclerView.children().forEach { child ->
        val childPosition = layoutManager.getPosition(child)
        val childViewHolder = recyclerView.getChildViewHolder(child)
        if (childViewHolder is ClipsRecyclerViewPlayer) {
            when {
                childPosition == playingPosition -> Unit
                childPosition > playingPosition && prepareViewHolder == null -> prepareViewHolder = childViewHolder
                else -> releaseViewHolders.add(childViewHolder)
            }
        }
    }

    val prepareRunnable = Runnable {
        releaseViewHolders.forEach { it.release() }
        prepareViewHolder?.prepare()
    }

    currentPrepareRunnable?.let { handler.removeCallbacks(it) }
    currentPrepareRunnable = prepareRunnable.also { handler.post(it) }
}
```

```
private fun schedulePrepareNext(recyclerView: RecyclerView) {
    val layoutManager = recyclerView.layoutManager as? LinearLayoutManager ?: return
    val playingView = (currentPlayHolder as? RecyclerView.ViewHolder)?.itemView ?: return
    val playingPosition = recyclerView.getChildAdapterPosition(playingView)

    val releaseViewHolders = mutableListOf<ClipsRecyclerViewPlayer>()
    var prepareViewHolder: ClipsRecyclerViewPlayer? = null

    recyclerView.children().forEach { child ->
        val childPosition = layoutManager.getPosition(child)
        val childViewHolder = recyclerView.getChildViewHolder(child)
        if (childViewHolder is ClipsRecyclerViewPlayer) {
            when {
                childPosition == playingPosition -> Unit
                childPosition > playingPosition && prepareViewHolder == null -> prepareViewHolder = childViewHolder
                else -> releaseViewHolders.add(childViewHolder)
            }
        }
    }

    val prepareRunnable = Runnable {
        releaseViewHolders.forEach { it.release() }
        prepareViewHolder?.prepare()
    }

    currentPrepareRunnable?.let { handler.removeCallbacks(it) }
    currentPrepareRunnable = prepareRunnable.also { handler.post(it) }
}
```

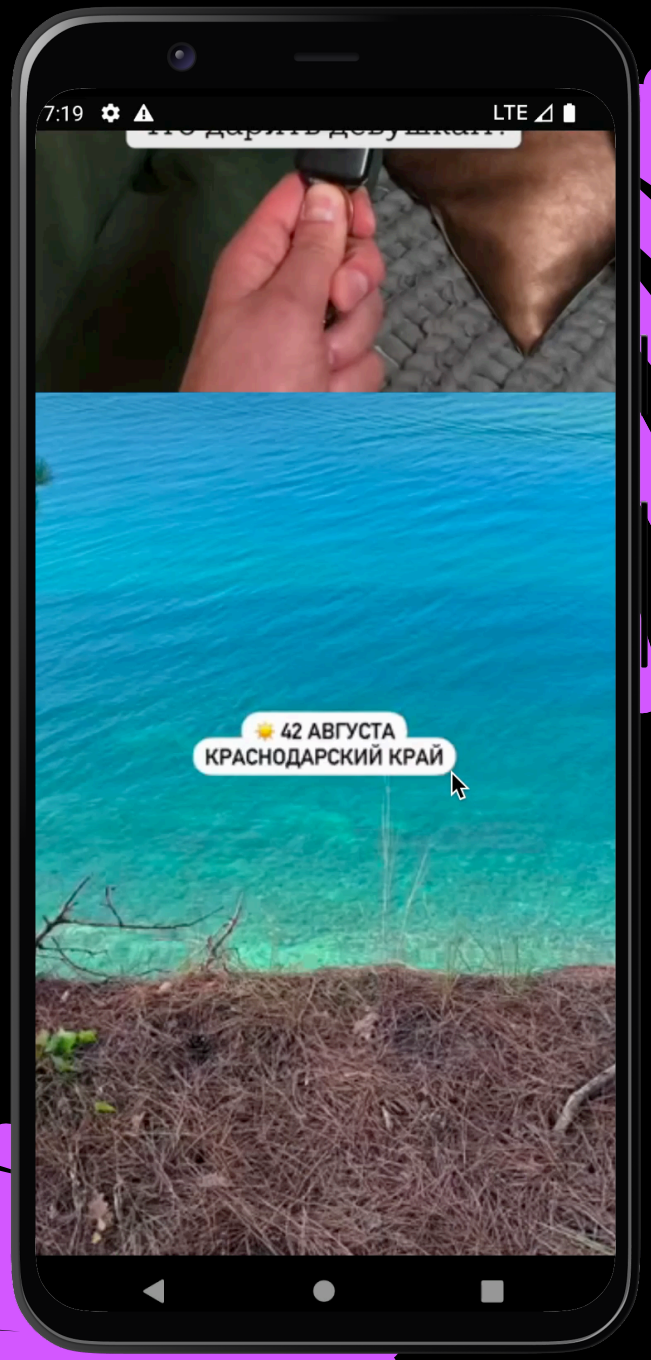
```kotlin
private fun schedulePrepareNext(recyclerView: RecyclerView) {
    val layoutManager = recyclerView.layoutManager as? LinearLayoutManager ?: return
    val playingView = (currentPlayHolder as? RecyclerView.ViewHolder)?.itemView ?: return
    val playingPosition = recyclerView.getChildAdapterPosition(playingView)

    val releaseViewHolders = mutableListOf<ClipsRecyclerViewPlayer>()
    var prepareViewHolder: ClipsRecyclerViewPlayer? = null

    recyclerView.children().forEach { child ->
        val childPosition = layoutManager.getPosition(child)
        val childViewHolder = recyclerView.getChildViewHolder(child)
        if (childViewHolder is ClipsRecyclerViewPlayer) {
            when {
                childPosition == playingPosition -> Unit
                childPosition > playingPosition && prepareViewHolder == null -> prepareViewHolder = childViewHolder
                else -> releaseViewHolders.add(childViewHolder)
            }
        }
    }

    val prepareRunnable = Runnable {
        releaseViewHolders.forEach { it.release() }
        prepareViewHolder?.prepare()
    }

    currentPrepareRunnable?.let { handler.removeCallbacks(it) }
    currentPrepareRunnable = prepareRunnable.also { handler.post(it) }
}
```

```
private fun schedulePrepareNext(recyclerView: RecyclerView) {
    val layoutManager = recyclerView.layoutManager as? LinearLayoutManager ?: return
    val playingView = (currentPlayHolder as? RecyclerView.ViewHolder)?.itemView ?: return
    val playingPosition = recyclerView.getChildAdapterPosition(playingView)

    val releaseViewHolders = mutableListOf<ClipsRecyclerViewPlayer>()
    var prepareViewHolder: ClipsRecyclerViewPlayer? = null

    recyclerView.children().forEach { child ->
        val childPosition = layoutManager.getPosition(child)
        val childViewHolder = recyclerView.getChildViewHolder(child)
        if (childViewHolder is ClipsRecyclerViewPlayer) {
            when {
                childPosition == playingPosition -> Unit
                childPosition > playingPosition && prepareViewHolder == null -> prepareViewHolder = childViewHolder
                else -> releaseViewHolders.add(childViewHolder)
            }
        }
    }

    val prepareRunnable = Runnable {
        releaseViewHolders.forEach { it.release() }
        prepareViewHolder?.prepare()
    }

    currentPrepareRunnable?.let { handler.removeCallbacks(it) }
    currentPrepareRunnable = prepareRunnable.also { handler.post(it) }
}
```

```
private fun schedulePrepareNext(recyclerView: RecyclerView) {
    val layoutManager = recyclerView.layoutManager as? LinearLayoutManager ?: return
    val playingView = (currentPlayHolder as? RecyclerView.ViewHolder)?.itemView ?: return
    val playingPosition = recyclerView.getChildAdapterPosition(playingView)

    val releaseViewHolders = mutableListOf<ClipsRecyclerViewPlayer>()
    var prepareViewHolder: ClipsRecyclerViewPlayer? = null

    recyclerView.children().forEach { child ->
        val childPosition = layoutManager.getPosition(child)
        val childViewHolder = recyclerView.getChildViewHolder(child)
        if (childViewHolder is ClipsRecyclerViewPlayer) {
            when {
                childPosition == playingPosition -> Unit
                childPosition > playingPosition && prepareViewHolder == null -> prepareViewHolder = childViewHolder
                else -> releaseViewHolders.add(childViewHolder)
            }
        }
    }

    val prepareRunnable = Runnable {
        releaseViewHolders.forEach { it.release() }
        prepareViewHolder?.prepare()
    }

    currentPrepareRunnable?.let { handler.removeCallbacks(it) }
    currentPrepareRunnable = prepareRunnable.also { handler.post(it) }
}
```

VK Клипы

Что получили? - То, что хотели!



VK Клипы

Тернистый путь

1. Проблема
2. Причины
3. Много плееров
4. Один плеер
5. Два плеера
6. One more thing

VK Клипы

Тернистый путь

1. Проблема
2. Причины
3. Много плееров
4. Один плеер
5. Два плеера
6. One more thing

VK Клипы

Можно было бы закончить, но...

VK Клипы

Можно было бы закончить, но...

Что если попробовать использовать 1 плеер и научить его переключаться между видео без фриза на главном потоке?

VK Клипы

Можно было бы закончить, но...

Что если попробовать использовать 1 плеер и научить его переключаться между видео без фриза на главном потоке?

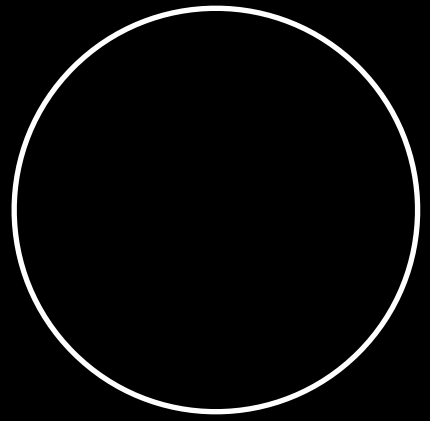
- Вариант 1 - фоновый поток для работы с плеером

VK Клипы

Можно было бы закончить, но...

Что если попробовать использовать 1 плеер и научить его переключаться между видео без фриза на главном потоке?

- Вариант 1 - фоновый поток для работы с плеером
- Вариант 2 - промежуточная сюрфа для плеера

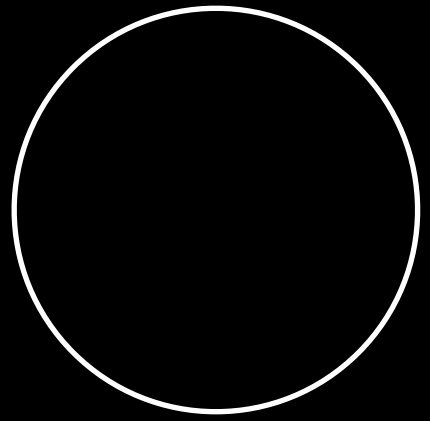


ExoPlayer

1

2

3



ExoPlayer

1

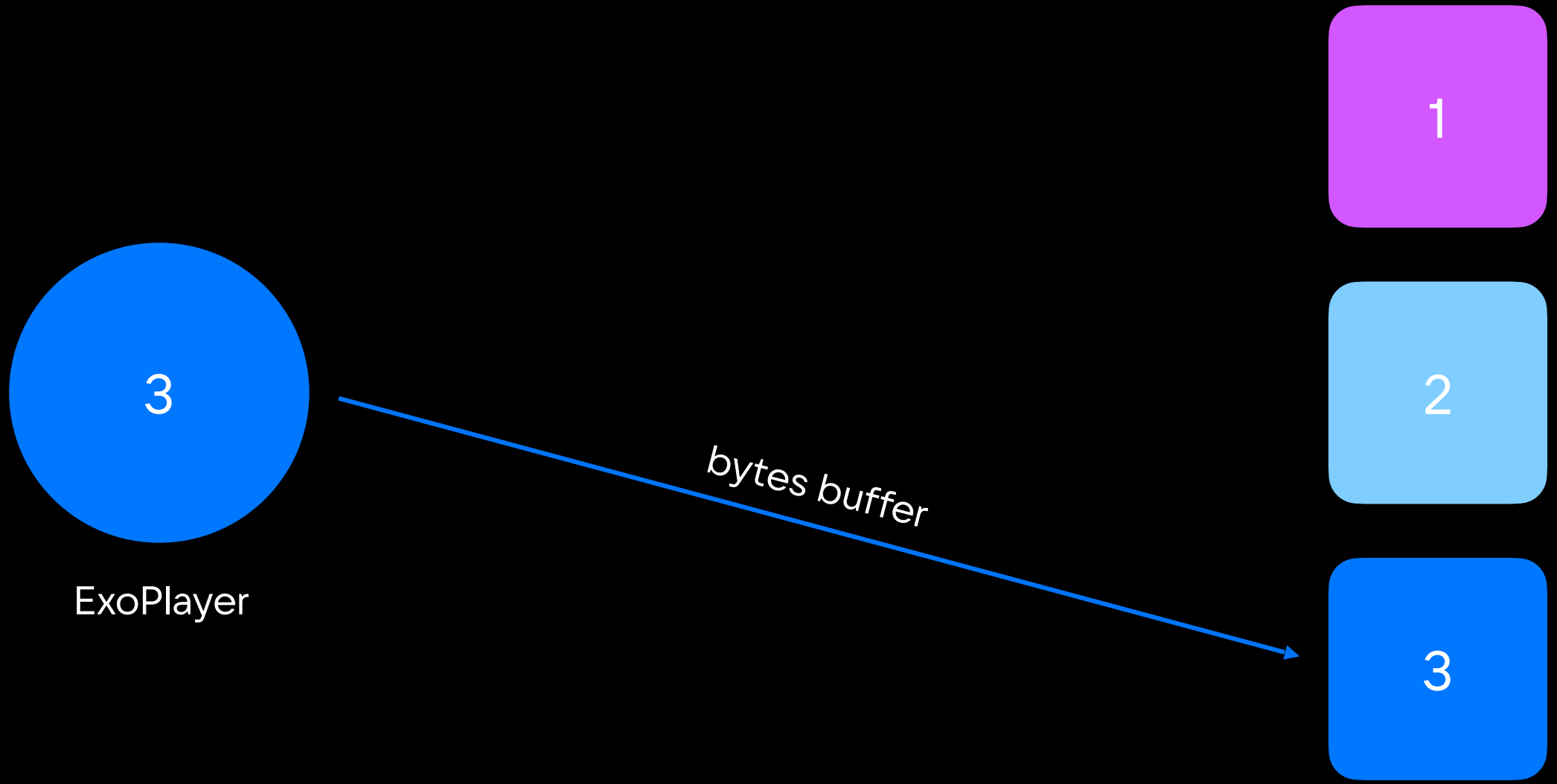
2

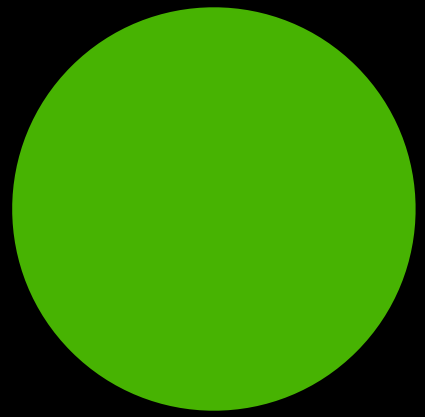
3

ExoPlayer

ExoPlayer

ExoPlayer


3
ExoPlayer
bytes buffer
1
2
3

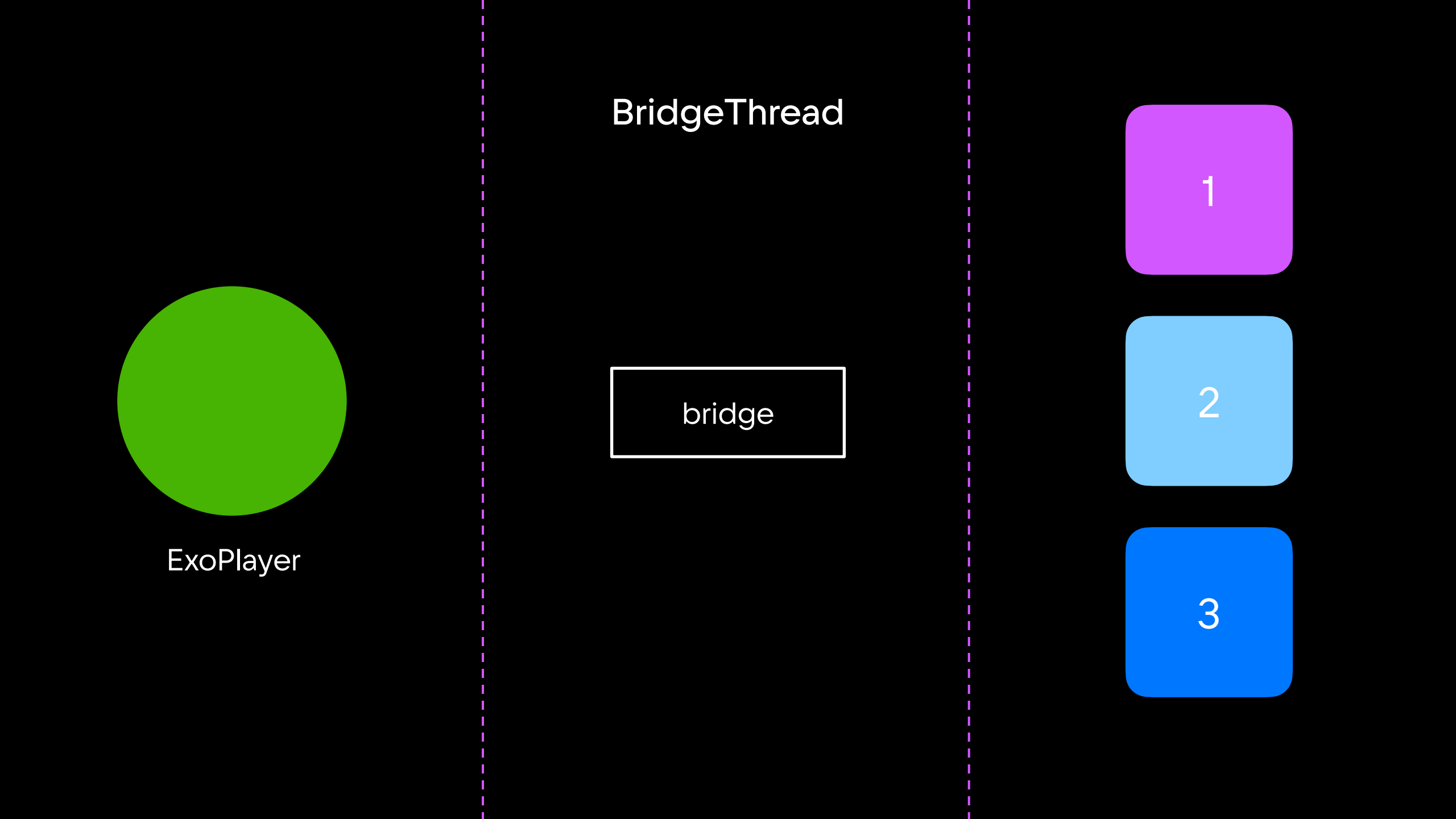


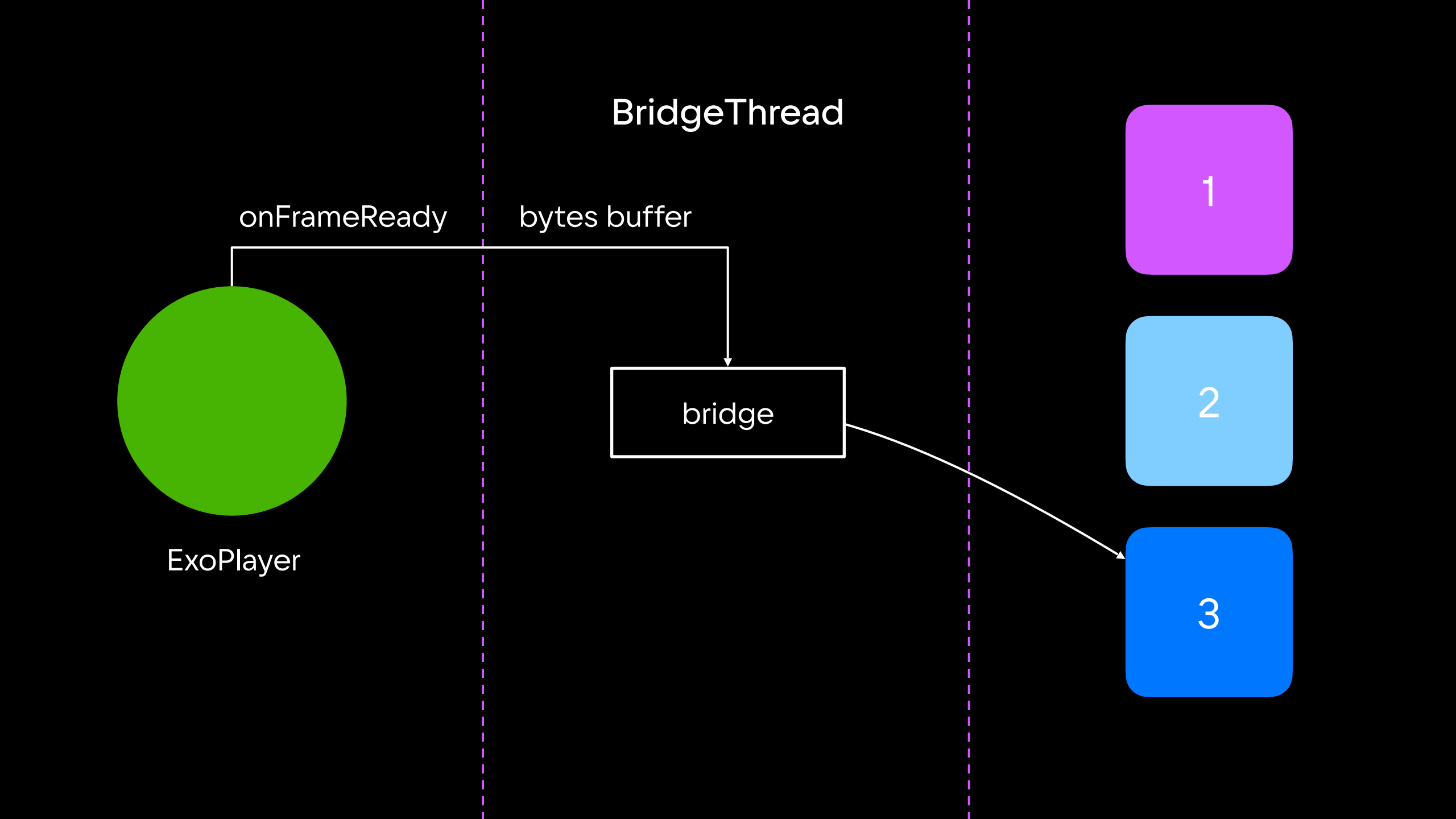
ExoPlayer

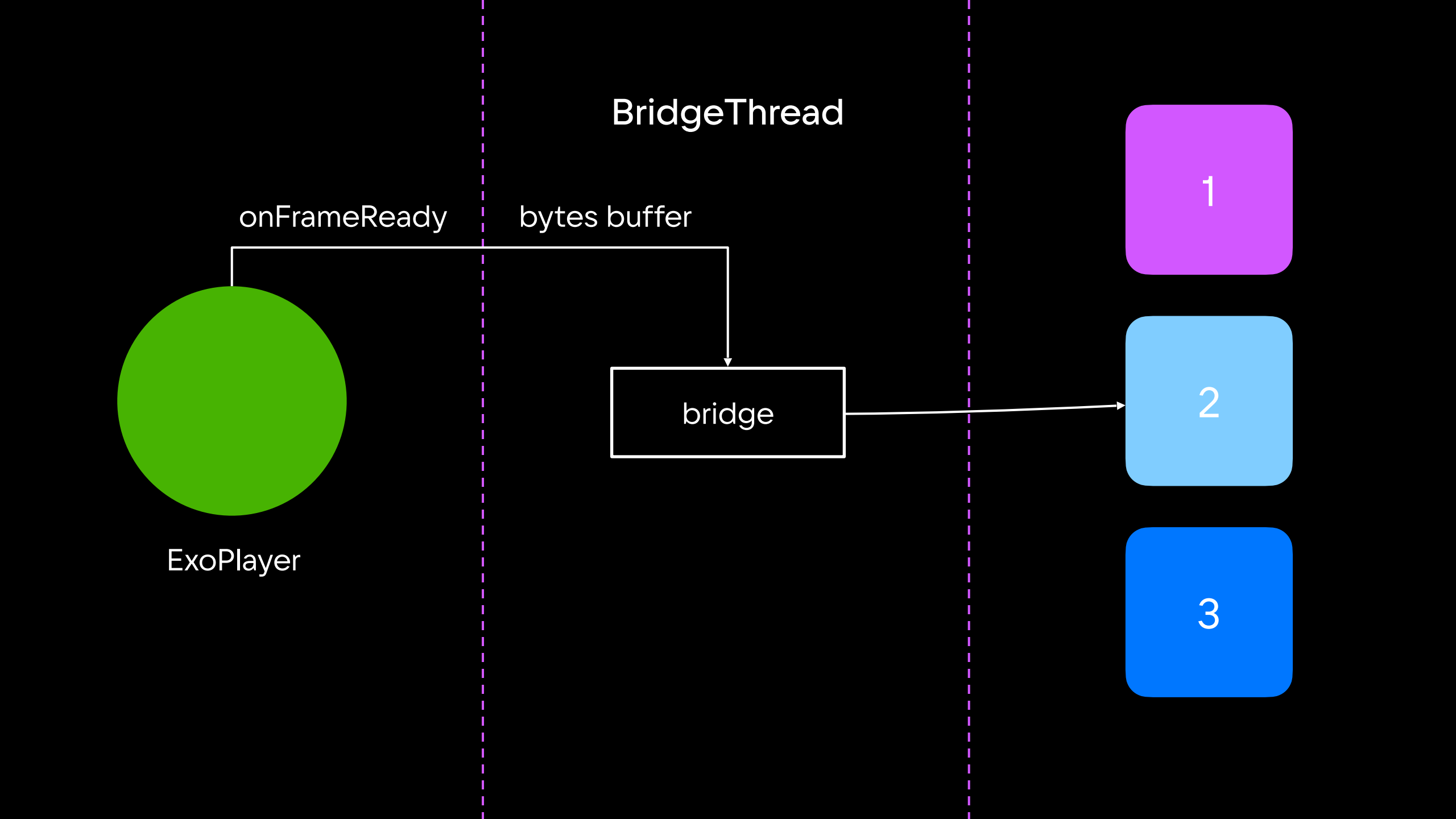
1

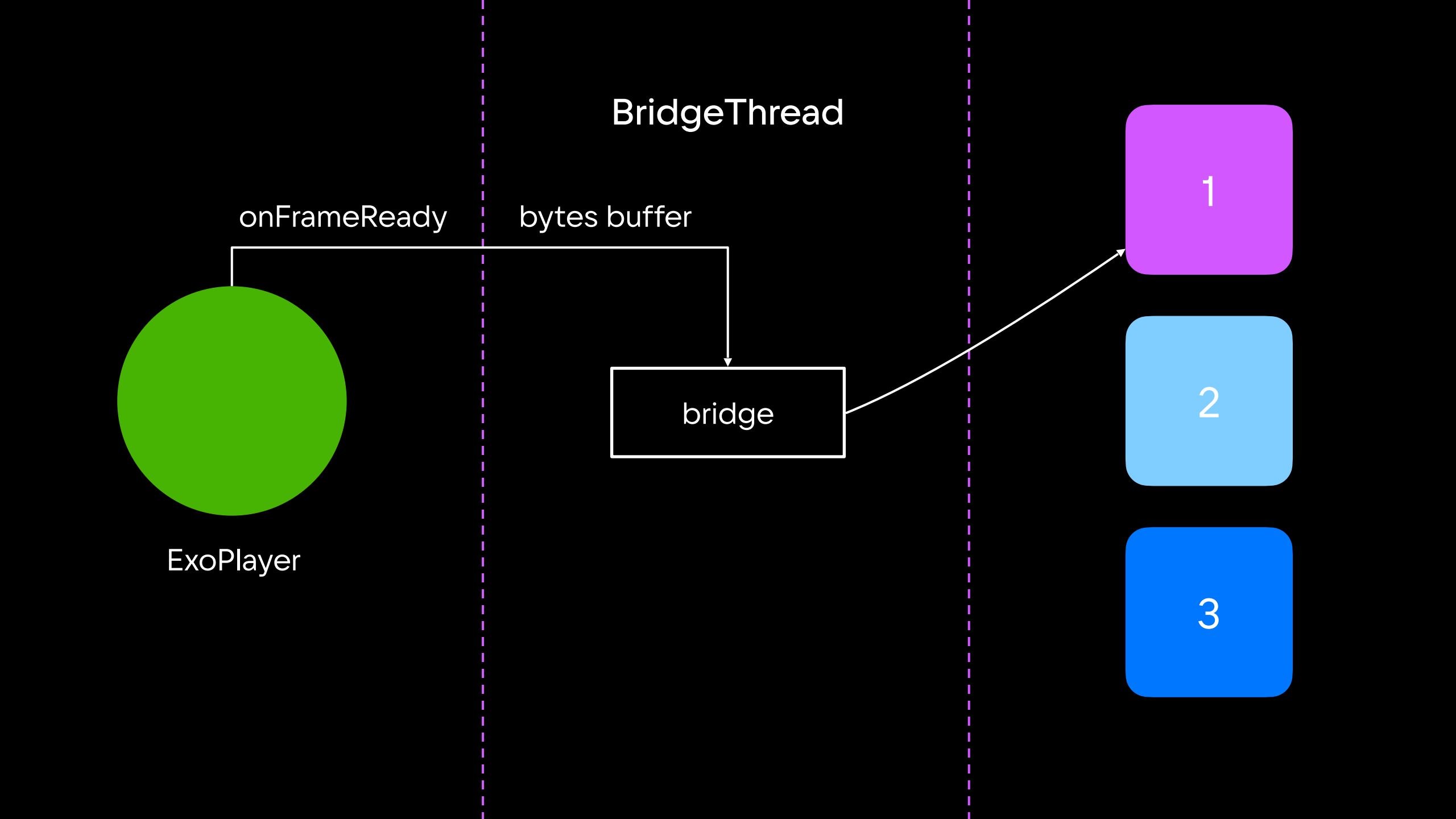
2

3


BridgeThread
bridge
ExoPlayer
1
2
3


BridgeThread
onFrameReady
bytes buffer
bridge
ExoPlayer
1
2
3


BridgeThread
onFrameReady
bytes buffer
ExoPlayer
bridge
1
2
3


BridgeThread
onFrameReady
bytes buffer
bridge
ExoPlayer
1
2
3

VK Клипы

Можно было бы закончить, но...

Что если попробовать использовать 1 плеер и научить его переключаться между видео без фриза на главном потоке?

- Вариант 1 - фоновый поток для работы с плеером
- Вариант 2 - промежуточная сюрфа для плеера

Но в целом на данный момент все крупные игроки используют 2 плеера

VK Клипы

Что еще можно попробовать?

VK Клипы

Что еще можно попробовать?

Использовать Playlist,
как советует Google

VK Клипы

Что еще можно попробовать?

Использовать Playlist,
как советует Google

Использовать кастомные
прекэширующие датасорсы

VK Клипы

О чем не рассказано?

VK Клипы

О чем не рассказано?

- Об архитектуре VP9 и том, где там можно выиграть время

О чем не рассказано?

- Об архитектуре VP9 и том, где там можно выиграть время
- О более современных кодеках (AV1/HEVC)

О чем не рассказано?

- Об архитектуре VP9 и том, где там можно выиграть время
- О более современных кодеках (AV1/HEVC)
- LoadControl для изменения размера буффера, необходимого для старта проигрывани - аккуратнее, можно нарваться на частые буфферизации

О чем не рассказано?

- Об архитектуре VP9 и том, где там можно выиграть время
- О более современных кодеках (AV1/HEVC)
- LoadControl для изменения размера буффера, необходимого для старта проигрывани - аккуратнее, можно нарваться на частые буфферизации
- О том, как прекэшировать видеопоток

О чем не рассказано?

- Об архитектуре VP9 и том, где там можно выиграть время
- О более современных кодеках (AV1/HEVC)
- LoadControl для изменения размера буффера, необходимого для старта проигрывани - аккуратнее, можно нарваться на частые буфферизации
- О том, как прекэшировать видеопоток
- И еще много всего – возможности кастомизации ExoPlayer поистине безграничны

VK Клипы

Итоги

VK Клипы

Итоги

Медленный старт видео – плохо

VK Клипы

Итоги

Медленный старт видео – плохо

Плееру нужно время на подготовку

VK Клипы

Итоги

Медленный старт видео – плохо

Плееру нужно время на подготовку

Готовьте плеер заранее

VK Клипы

Итоги

Медленный старт видео – плохо

Используйте пул из двух плееров

Плееру нужно время на подготовку

Готовьте плеер заранее

VK Клипы

Итоги

Медленный старт видео – плохо

Используйте пул из двух плееров

Плееру нужно время на подготовку

Оптимизация внутри плеера – промежуточная surface

Готовьте плеер заранее

VK Клипы

Итоги

Медленный старт видео – плохо

Используйте пул из двух плееров

Плееру нужно время на подготовку

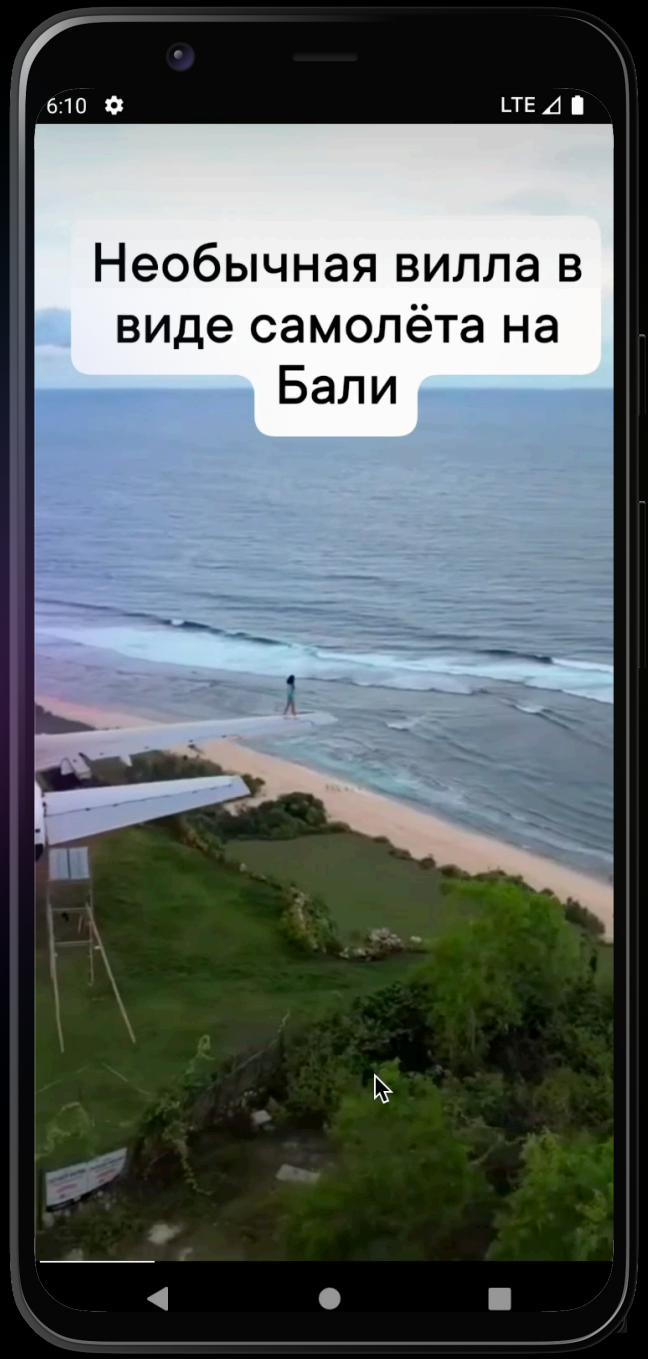
Оптимизация внутри плеера – промежуточная surface

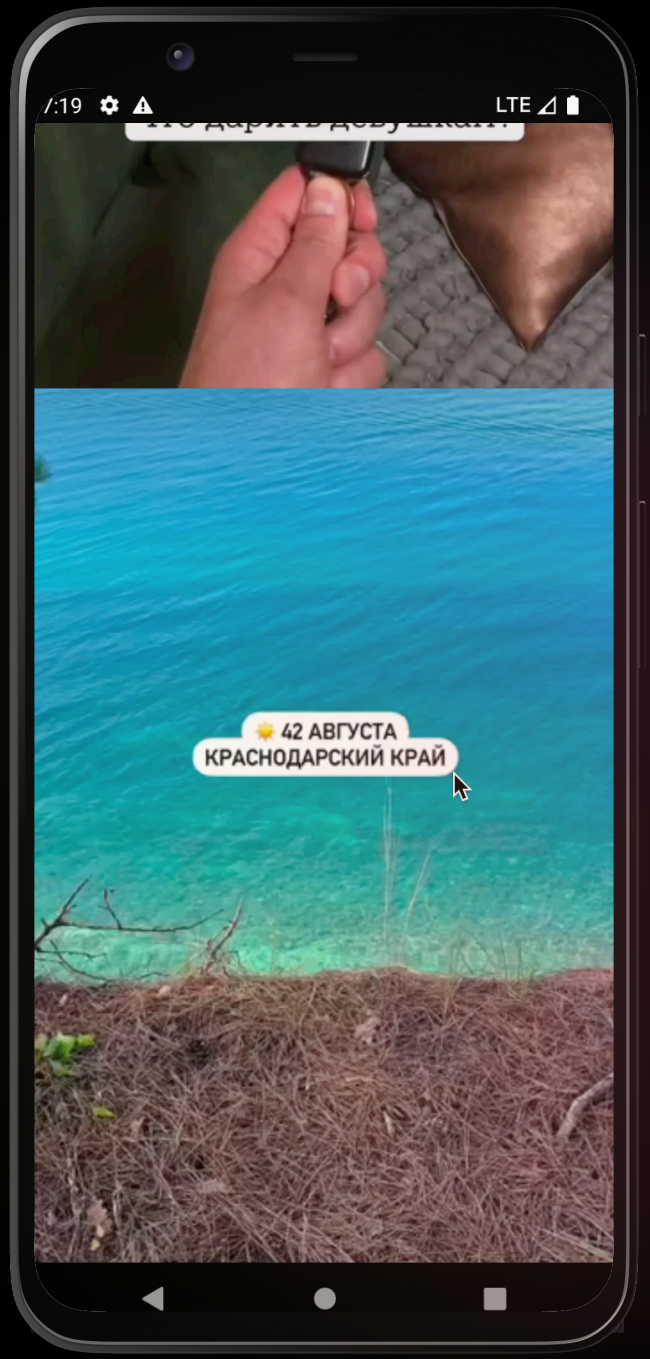
Готовьте плеер заранее

Playlist, кастомные датасорсы

VK Клипы

+210мс





-190мс

Будем ВКонтакте!