OTUS

Android Developer. Basic

Services

otus.ru

Проверить, идет ли запись

Меня хорошо видно && слышно?

Ставим “+”, если все хорошо
“-”, если есть проблемы

Тема вебинара

Service

Максим Качинкин

Android Tech Lead, Dodo Engineering

Об опыте:
10 лет опыта в Android разработке

telegram: **@mobilefiction**

Правила вебинара

Активно участвуем

Пишите вопросы в чат

Буду делать паузы на ответы

Маршрут вебинара

- Service
- История ограничений фоновой работы в Android
- Live coding

Цели вебинара

К концу занятия вы сможете

1.	Создавать Background сервисы
2.	Создавать Bounded сервисы
3.	Создавать Foreground сервисы

Смысл

Зачем вам это уметь

1. Правильно подбирать инструмент фоновой работы для вашей задачи

Кто работал с Background Service, напишите 1
Кто работал с Bound Service, напишите 2
Кто работал с Foreground Service, напишите 3

Фоновая работа

Что такое фоновая работа в Android

Это когда пользователь не видит Activity приложения, а работа какая-то идёт

Инструменты в Android

- Service
- WorkManager
- Alarm Manager
- Broadcast Receivers
- JobScheduler
- Sync Adapter
- Download Manager
- Firebase Job Dispatcher, GCMNetworkManager - Deprecated

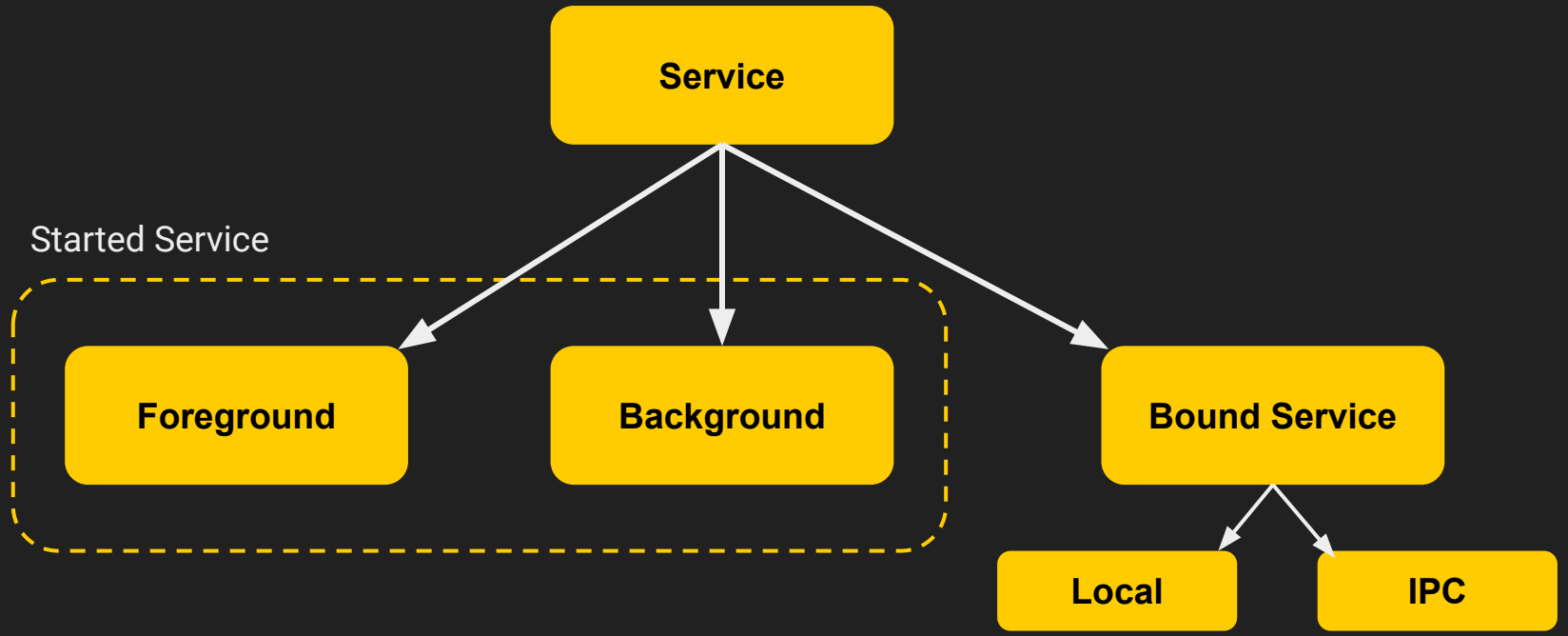
Service

Что такое сервис

Service – компонент приложения, позволяющий выполнять длительные операции в фоне. Не имеет собственного UI*, но работает в главном потоке приложения.

*** кроме уведомлений в случае foreground service**

Типы сервисов



Started service

Started service

- Запускается при помощи `startService`
- Останавливается вызовом `stopService` или `stopSelf`

```
Intent(this, MyService::class.java).also { intent ->
   startService(intent)
}

...

Intent(this, MyService::class.java).also { intent ->
   stopService(intent)
}
```

Started service

- Запускается при помощи `startService`
- Останавливается вызовом `stopService` или `stopSelf`

```
class MyService : Service() {

   override fun onBind(intent: Intent): IBinder {
       throw UnsupportedOperationException()
   }
   override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {
       return START_STICKY
   }

}
```

Started service

Режимы запуска `onStartCommand` в случае смерти процесса:

- `START_STICKY` - перезапускать сервис
- `START_NOT_STICKY` - не перезапускать сервис
- `START_REDELIVER_INTENT` - перезапускать сервис с последним полученным Intent’ом

Foreground service

- Добавить пермишен `FOREGROUND_SERVICE` в манифест
- Запустить сервис при помощи `startForegroundService` вместо `startService`
- В течение 5 секунд после запуска сервиса вызвать в нем `startForeground` для показа уведомления

Есть ли вопросы?

Bound service

Bound Service

1. Имплементировать binder и вернуть его в методе onBind
   a. Наследник Binder
   b. Используя Messenger
   c. AIDL
2. Подключиться к сервису при помощи `bindService`
   a. И не забыть отключить в `unbindService`

Bound Service

```
class MyService : Service() {

    override fun onBind(intent: Intent): IBinder = LocalBinder()

    inner class LocalBinder : Binder() {
        fun doJob() { }
    }
}
```

Bound Service

```
class MainActivity : AppCompatActivity() {
   private val connection = object : ServiceConnection {

       override fun onServiceConnected(className: ComponentName, service: IBinder) {
           val binder = service as MyService.LocalBinder
           binder.doJob()
       }

       override fun onServiceDisconnected(className: ComponentName) {}
   }
   override fun onStart() {
       super.onStart()
       bindService(Intent(this, MyService::class.java), connection, Context.BIND_AUTO_CREATE)
   }
   override fun onStop() {
       unbindService(connection)
       super.onStop()
   }
}
```

AIDL

AIDL - Android Interface Definition Language

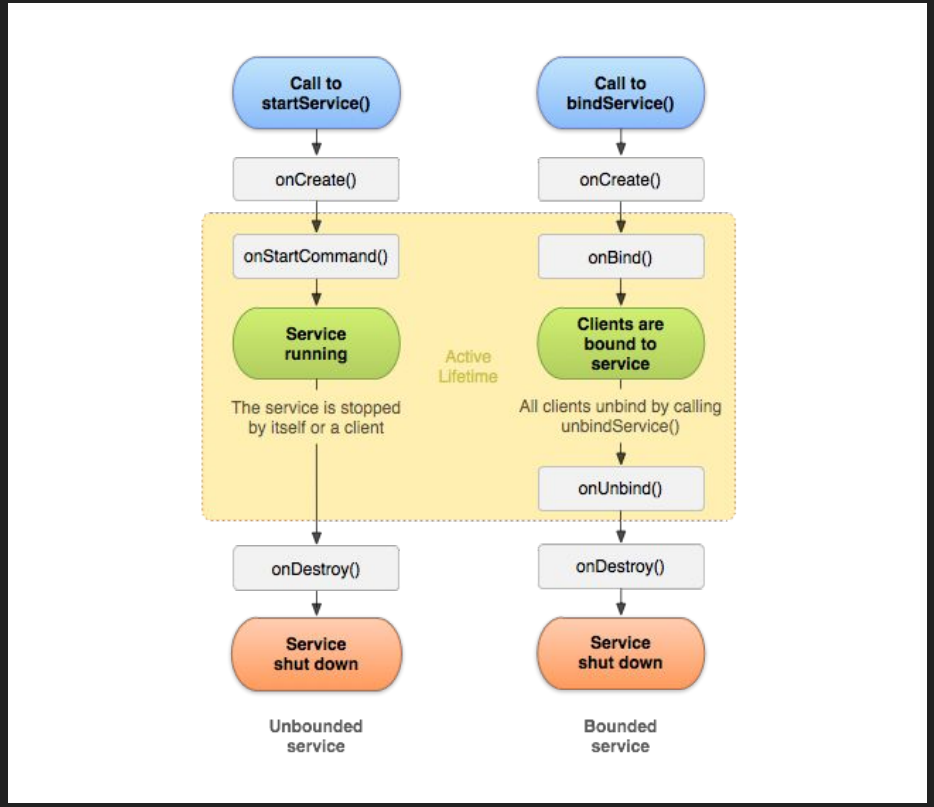
YourService.aidl

```
interface IRemoteService {
   /** Request the process ID of this service. */
   int getPid();

   /** Demonstrates some basic types that you can use as parameters
    * and return values in AIDL.
    */
   void basicTypes(int anInt, long aLong, boolean aBoolean, float aFloat, double
aDouble, String aString);
}
```

Жизненный цикл сервисов

Жизненный цикл



В какой момент сервис завершится, если сервис запустить как Started, а потом прибиндится к нему и отбиндится?

```
startService(Intent(this, MyService::class.java))

...

bindService(Intent(this, MyService::class.java), connection, Context.BIND_AUTO_CREATE)

...

unbindService(connection)
```

Приоритеты процессов

1. Foreground process – взаимодействует с пользователем
2. Visible process – пользователь видит, но не взаимодействует
3. Service process – делает невидимую для пользователя работу
4. Cached process – не делает работы и не видим пользователю

Документация: приоритеты процессов

Приоритеты процессов

1. Foreground process – взаимодействует с пользователем
2. Visible process – пользователь видит, но не взаимодействует (Foreground service)
3. Service process – делает невидимую для пользователя работу
4. Cached process – не делает работы и не видим пользователю

Документация: приоритеты процессов

Android 5

Android 5:

- JobScheduler API
- https://developer.android.com/about/versions/lollipop/android-5.0#JobScheduler

Android 6:

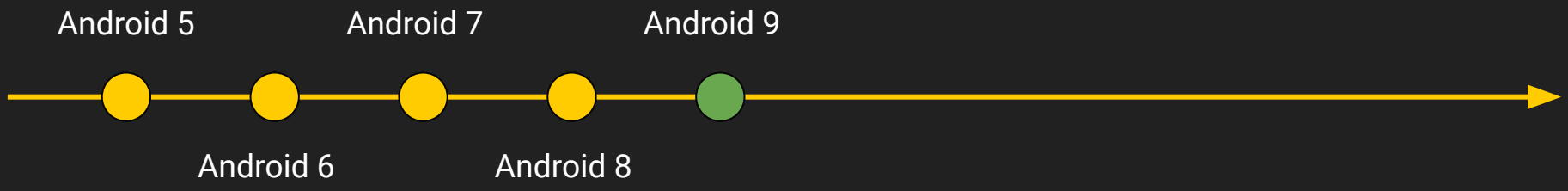
- Doze Mode
- App Standby
- https://developer.android.com/about/versions/marshmallow/android-6.0-changes#behavior-power

Android 7:

- Doze Mode продолжение
- Нельзя подписываться на 3 неявных broadcast:
  - CONNECTIVITY_ACTION
  - ACTION_NEW_PICTURE
  - ACTION_NEW_VIDEO

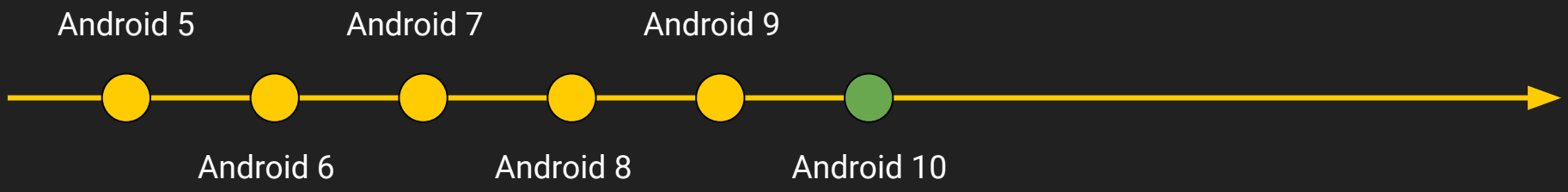
Android 8:

- Нельзя регистрировать broadcast receivers для неявных broadcasts https://developer.android.com/guide/components/broadcast-exceptions
- **Ввели разделение на foreground и background**. Пока приложение в foreground, оно может создавать и foreground и background сервисы. Если приложение в background, оно имеет окно в несколько минут, в течение которого может запускать сервисы. По окончанию - не может.
  https://developer.android.com/about/versions/oreo/background#services



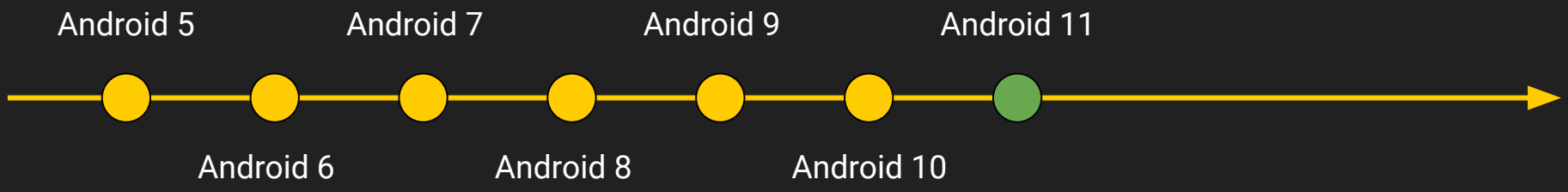
Android 9:

- Для foreground service нужно объявлять FOREGROUND_SERVICE разрешение (SecurityException)
- App StandBy Buckets



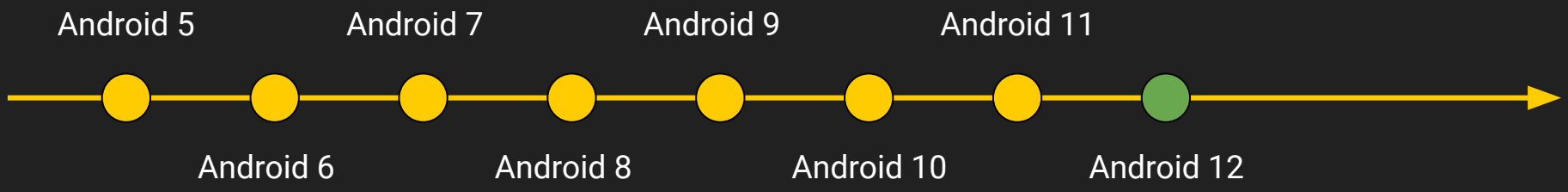
Android 10:

- foreground service types, список: https://developer.android.com/guide/topics/manifest/service-element#foregroundservicetype
- Чтобы получить доступ к локации в foreground service нужно объявить тип location для foreground service
- Запрещено запускать activity из background



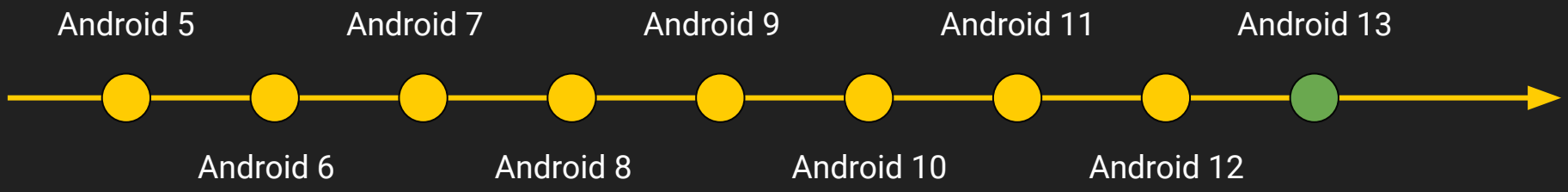
Android 11:

- IntentService deprecated
- Если нужен доступ к камере или микрофону в foreground service, то надо задать camera или microphone тип для foreground service types
- Если вы запустили foreground service из состояния background:
  - нельзя получить доступ к локации за исключением если нет ACCESS_BACKGROUND_LOCATION разрешения
  - нельзя получить доступ к микрофону или камере
  - но есть исключения: https://developer.android.com/guide/components/foreground-services#bg-access-restriction-exemptions



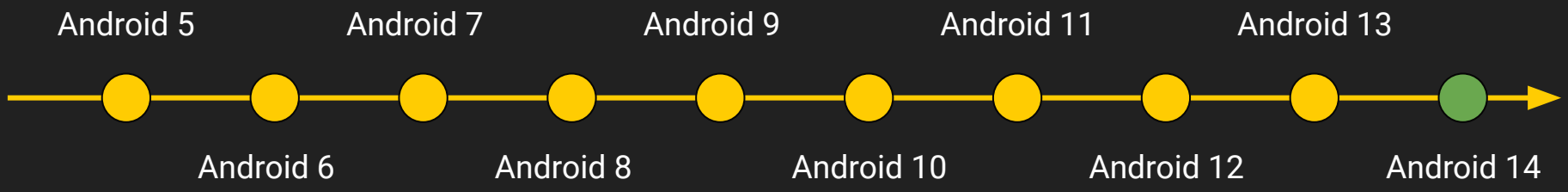
Android 12:

- Короткоживущие foreground services: система ждет 10 секунд, перед тем как показать уведомление. Но есть исключения: https://developer.android.com/guide/components/foreground-services#notification-immediate
- Нельзя запустить foreground services если приложение в background (ForegroundServiceStartNotAllowedException), но есть исключения https://developer.android.com/guide/components/foreground-services#background-start-restriction-exemptions



Android 13:

- пользователь может смахнуть уведомление от foreground service
- чтобы этого избежать надо установить флаг через метод `setOngoing()`
- если пользователь отклонил разрешение на показ уведомлений, то он всеравно увидит уведомление о foreground service в Task Manager (но есть исключения)
- пользователь может остановить приложение (включая foreground service) через Task Manager (но есть исключения)
- система не пришлет нам колбэк о том, что нас отменили



Android 14:

- Новые типы сервисов: *health*, *remoteMessaging*, ***shortService***, *specialUse*, *systemExempted*
- обязательно объявлять специальные разрешения для соответствующего типа foreground service
- кроме Short Service
- https://developer.android.com/about/versions/14/changes/fgs-types-required#use-cases

Service это не SystemService

- Service — это компонент Андроид приложения
  - Started/Bounded
  - Foreground/Background
- SystemService — это компонент Андроид ОС
  - получаем доступ через Context, через метод getSystemService
  - WindowManager, ActivityManager, PackageManager, LocationManager, etc

Есть ли вопросы?

WorkManager

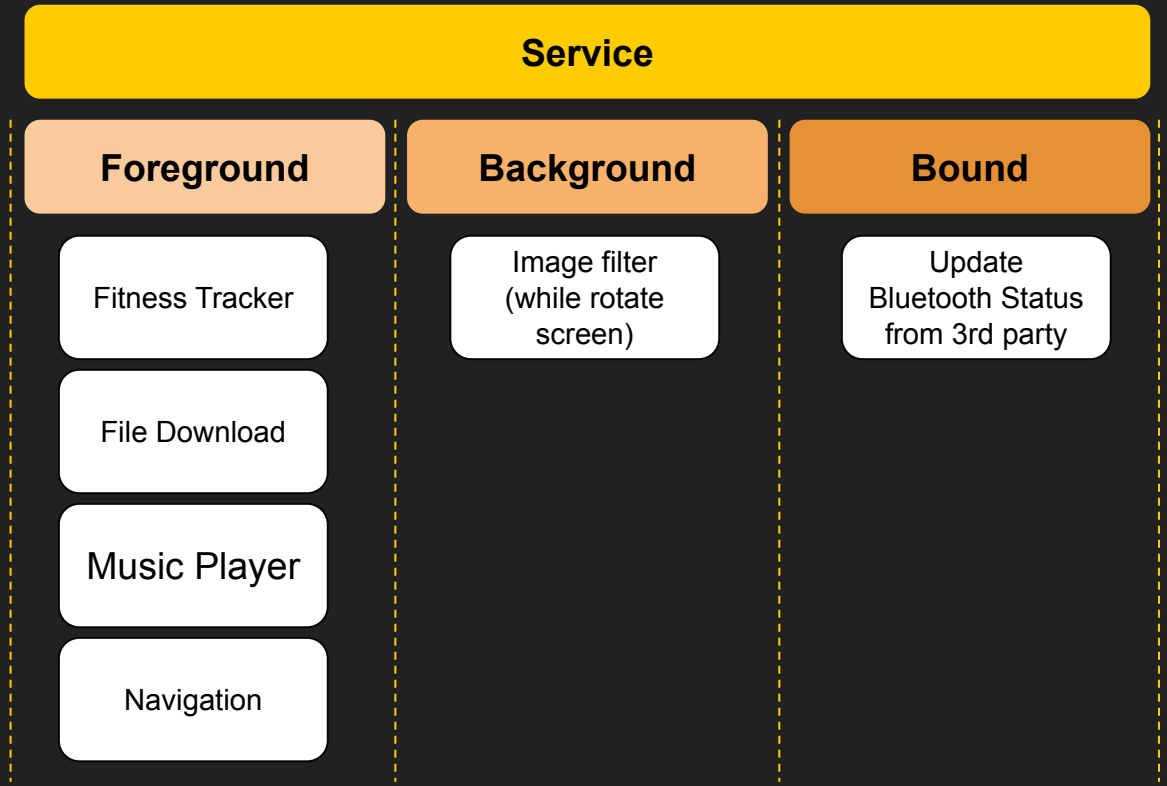
WorkManager – библиотека, являющаяся часть Android Jetpack, предоставляющая возможность для выполнения одноразовой или повторяющейся фоновой работы.

Но это на следующем уроке…

LIVE

https://github.com/makzimi/otus-basic-service-lesson

Что где использовать?



Рефлексия

Цели вебинара

Проверка достижения целей

1.	Создавать Background/Foreground/Bounded сервисы
2.	Правильно подбирать инструмент фоновой работы для вашей задачи

Заполните, пожалуйста, опрос о занятии по ссылке в чате

Спасибо за внимание!

Приходите на следующие вебинары

Максим Качинкин

Android Tech Lead, Dodo Engineering

telegram: @mobilefiction