Desenvolvimento para Dispositivos Móveis

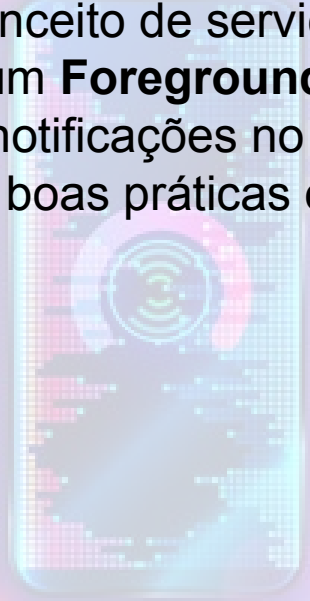
Aula 10

Notificações e Serviços em Background

Prof. Msc. Francisco de Assis de Souza Rodrigues

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

- Entender o conceito de serviços em background e sua importância.
- Implementar um **ForegroundService**.
- Criar e exibir notificações no Android.
- Compreender boas práticas e limitações para background tasks (API 26+).



UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

🧩 O que são Serviços no Android?

Serviços (Services) são componentes Android usados para executar operações em background, sem interação direta com o usuário. Eles são ideais para tarefas que precisam continuar mesmo se o usuário sair da aplicação, como:

- Reproduzir música
- Sincronizar dados com um servidor
- Registrar localização periodicamente

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

🧩 O que são Serviços no Android?

Tipos de Serviços

Service

- É o tipo mais básico.
- Roda no **mesmo processo da aplicação**.
- Ideal para **tarefas curtas ou controladas**, mas **não cria threads automaticamente**. Se for algo demorado, deve-se criar uma thread dentro dele.

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

🧩 O que são Serviços no Android?

```
class MeuServico : Service() {
    override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {
        // Código da tarefa
        return START_NOT_STICKY
    }

    override fun onBind(intent: Intent?): IBinder? = null
}
```

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

🧩 O que são Serviços no Android?

ForegroundService

- Um Service que **deve exibir uma notificação obrigatória** enquanto está rodando.

- É ideal para tarefas que **o usuário deve estar ciente**, como:
  - Gravação de áudio
  - GPS ativo
  - Uploads de arquivos grandes

- **Menos suscetível a ser encerrado pelo sistema**, mesmo em economia de bateria.

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

🧩 O que são Serviços no Android?

IntentService (Obsoleto)

- Extensão de Service que já **cria uma thread separada automaticamente**.
- Executa uma tarefa por vez e **para automaticamente** ao final.
- Substituído por abordagens modernas como WorkManager.

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Comparação:
Service vs
JobIntentService vs
WorkManager

Característica	Service	JobIntentService	WorkManager
Criação de thread própria	❌ Manual	✅ Automática	✅ Automática
Compatível com API < 26	✅ Sim	✅ Sim	✅ Sim
Compatível com API ≥ 26	⚠️ Limitado	⚠️ Limitado (deprecado)	✅ Recomendado
Persistência (após reboot)	❌ Não	❌ Não	✅ Sim (opcional)
Condições de execução	❌ Nenhuma	❌ Nenhuma	✅ Com condições (rede, bateria, etc)
Cancelamento e Retry	❌ Difícil	❌ Difícil	✅ Fácil

✳ Resumo das boas práticas:

- **Tarefa rápida e imediata?** Service
- **Tarefa que o usuário precisa saber?** ForegroundService
- **Tarefa agendada ou que deve persistir mesmo após reboot?** WorkManager

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

🚫 Por que o Android impõe restrições ao background?

A partir do Android 8 (API 26), o sistema impôs **restrições severas** ao uso de serviços em background por três motivos principais:

1. **Bateria**: Serviços rodando sem controle drenam a bateria.

2. **Desempenho**: Muitos apps rodando simultaneamente em background degradam a performance do dispositivo.

3. **Experiência do usuário**: Usuários ficavam com apps "invisíveis" rodando sem saber.

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

🚫 Por que o Android impõe restrições ao background?

🔐 **Restrições importantes:**

- **Services não podem ser iniciados em background sem notificação** (exceto WorkManager ou Broadcasts específicos).
- Serviços **não-foreground são encerrados rapidamente** se o app estiver em segundo plano.
- O Android força o uso de ForegroundService, JobScheduler, ou WorkManager para tarefas de longa duração.

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

🚫 Por que o Android impõe restrições ao background?

📌 Exemplo de erro comum:

```
IllegalStateException: Context.startService() not allowed while app is in background
```

✅ Soluções modernas:

- Para tarefas agendadas ou que não precisam de interação direta: **WorkManager**
- Para tarefas imediatas, visíveis ao usuário: **ForegroundService**

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Notificações no Android

📢 O que são Notificações?

Notificações são **mensagens fora da interface da aplicação** que informam o usuário sobre algo importante. Elas aparecem na área de notificação do Android e podem:

- Exibir informações ao usuário.
- Lançar uma atividade ao ser tocada.
- Executar ações diretamente (como responder uma mensagem ou cancelar um download).

Notificações no Android

🧱 Estrutura de uma Notificação

Uma notificação básica tem:



Elemento	Função
Ícone pequeno	Exibido na barra de status
Título (title)	Frase curta que resume a notificação
Texto (content text)	Informação mais detalhada
Ação (opcional)	Botões para interagir sem abrir o app
Canal (NotificationChannel)	Onde a notificação será agrupada

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Notificações no Android

```
val builder = NotificationCompat.Builder(context, CHANNEL_ID)
    .setSmallIcon(R.drawable.ic_notification)
    .setContentTitle("Título")
    .setContentText("Mensagem exibida ao usuário")
    .setPriority(NotificationCompat.PRIORITY_DEFAULT)
```

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

🔗 NotificationChannel (API 26+)

A partir do Android 8.0 (API 26), todas as notificações devem estar associadas a um canal (NotificationChannel). Isso permite que o usuário gerencie o comportamento das notificações por tipo (vibração, som, importância, etc.).



UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

🔗 NotificationChannel (API 26+)

🧪 Criando um canal:

```
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
    val name = "Canal Padrão"
    val descriptionText = "Usado para notificações gerais"
    val importance = NotificationManager.IMPORTANCE_DEFAULT
    val channel = NotificationChannel("meu_canal_id", name, importance).apply {
        description = descriptionText
    }

    val notificationManager: NotificationManager =
        context.getSystemService(Context.NOTIFICATION_SERVICE) as NotificationManager
    notificationManager.createNotificationChannel(channel)
}
```

🔗 NotificationChannel (API 26+)

📝 **Parâmetros importantes:**

- **ID**: Identificador único do canal ("meu_canal_id")
- **Nome**: Visível para o usuário nas configurações
- **Importância**: Determina como a notificação se comporta (som, alerta, etc.)

🔧 Criando notificações com NotificationCompat

A classe NotificationCompat é usada para garantir compatibilidade com versões antigas do Android. É parte da biblioteca AndroidX.

Exemplo completo:

```
val builder = NotificationCompat.Builder(this, "meu_canal_id")
    .setSmallIcon(R.drawable.ic_notification)
    .setContentTitle("Nova mensagem")
    .setContentText("Você tem uma nova mensagem.")
    .setPriority(NotificationCompat.PRIORITY_DEFAULT)

with(NotificationManagerCompat.from(this)) {
    notify(1, builder.build()) // 1 é o ID da notificação
}
```

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

🧠 Dicas Importantes

- Cada **canal deve ser criado apenas uma vez**, geralmente na inicialização do app.
- As notificações com o mesmo ID são **atualizadas**, não duplicadas.
- É possível adicionar **ações** e **intenções pendentes (PendingIntent)** para interação.

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

🧠 Dicas Importantes

🔁 Exemplo com clique:

```
val intent = Intent(this, MinhaActivity::class.java)
val pendingIntent = PendingIntent.getActivity(this, 0, intent, PendingIntent.FLAG_IMMUTABLE)

val builder = NotificationCompat.Builder(this, "meu_canal_id")
    .setSmallIcon(R.drawable.ic_notification)
    .setContentTitle("Clique aqui")
    .setContentText("Essa notificação abre uma Activity")
    .setContentIntent(pendingIntent)
    .setAutoCancel(true) // fecha a notificação ao clicar
```

📌 Conclusão

- Toda notificação a partir do Android 8 **precisa de um canal**.
- NotificationCompat é a forma recomendada de criar notificações compatíveis com várias versões.
- Notificações são essenciais para apps que usam **serviços em background**, como ForegroundService.

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Foreground Services

🛠️ O que é um ForegroundService?

Um **ForegroundService** é um tipo especial de Service que:

- Executa tarefas em segundo plano.
- **Permanece ativo mesmo quando o app está em background.**
- Exige **uma notificação visível ao usuário** durante toda sua execução.

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Foreground Services

👉 **Exemplos de uso:**

- Reprodução de áudio (Spotify, YouTube Music)
- Gravação de vídeo ou voz
- Localização em tempo real (Waze, Google Maps)
- Upload ou download de arquivos longos

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Foreground Services

O Android **impõe essa exigência a partir da API 26+** para que o sistema:

1. **Evite abusos**: Aplicativos mal-intencionados não podem rodar “escondidos” em segundo plano sem que o usuário saiba.

2. **Melhore a autonomia da bateria**: Apenas serviços essenciais e visíveis têm permissão para rodar continuamente.

3. **Ofereça transparência**: O usuário vê o que está sendo executado e pode tomar decisões (fechar o app, restringir permissões etc).

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Foreground Services

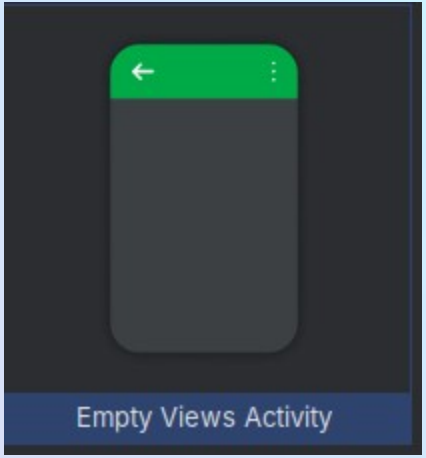
🔐 Se um serviço estiver rodando em primeiro plano **sem uma notificação ativa**, o Android o **interrompe imediatamente** e lança um erro como:

```
ForegroundService did not call startForeground()
```

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Prática

Crie um Projeto Android: **Empty Views Activity**



UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Prática

No **AndroidManifest.xml** adicione:

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission android:name="android.permission.FOREGROUND_SERVICE" />
    <uses-permission android:name="android.permission.FOREGROUND_SERVICE_DATA_SYNC" />
    <uses-permission android:name="android.permission.POST_NOTIFICATIONS" />
```

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Prática

No **AndroidManifest.xml** adicione o serviço dentro de **<application/>**

```
    <service
        android:name=".MyService"
        android:foregroundServiceType="dataSync"
        android:exported="false" />

</application>
```

Outros valores possíveis:

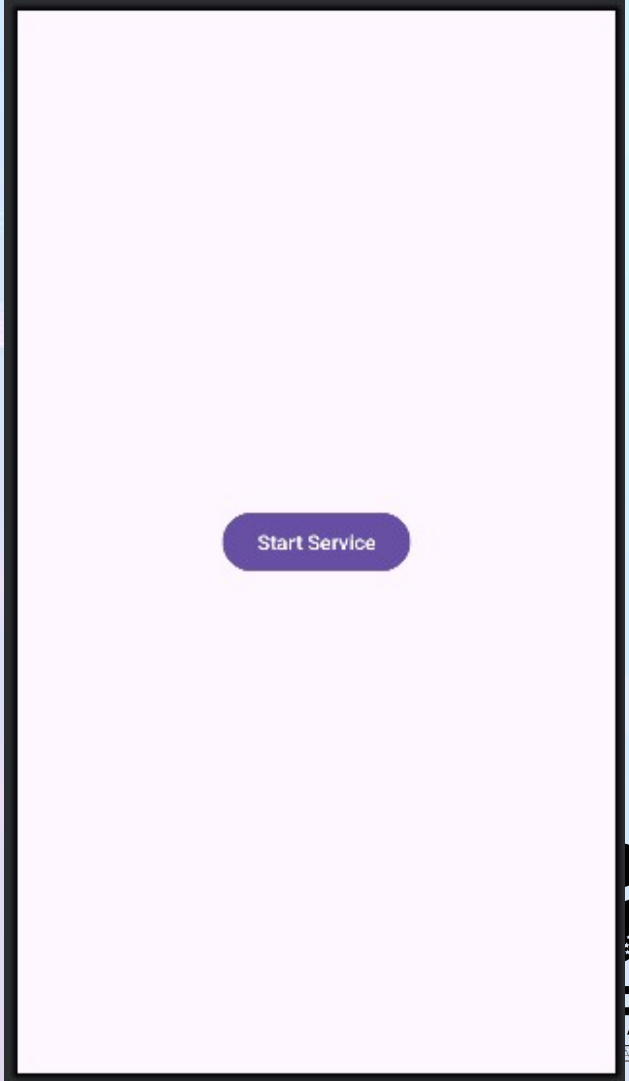
- dataSync → Sincronização de dados
- mediaPlayback → Reprodução de mídia
- location → Uso de GPS
- microphone → Gravação de áudio
- camera → Uso da câmera
- connectedDevice → Comunicação com dispositivos Bluetooth/USB
- mediaProjection → Gravação de tela
- shortService → Serviços curtos (menos de 10 segundos)

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Prática

No Arquivo **activity_main.xml** adicione botão:

```
<Button
    android:id="@+id/btnStartservice"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:text="Start Service"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```



Prática

Crie uma classe **MyService**

```
class MyService : Service (){
    override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {
        val notification = NotificationCompat.Builder( context: this,  channelId: "notification_example")
            .setContentTitle("Service run")
            .setContentText("The service is running in the background.")
            .setSmallIcon(R.drawable.ic_notifications)
            .build()

        startForeground( id: 1, notification)

        Thread {
            for (i in 1 ≤ .. ≤ 10) {
                Log.d( tag: "Service",  msg: "Cont: $i")
                Thread.sleep( millis: 1000)
            }
            stopSelf()
        }.start()

        return START_NOT_STICKY
    }

    override fun onBind(intent: Intent?): IBinder? = null
}
```

Prática No arquivo principal **MainActivity**

```
class MainActivity : AppCompatActivity() {
    private val channelId = "notification_example"

    private val requestPermissionLauncher = registerForActivityResult(
        ActivityResultContracts.RequestPermission()
    ) { isGranted ->
        if (isGranted) {
            Toast.makeText(this, "Permission granted", Toast.LENGTH_SHORT).show()
        } else {
            Toast.makeText(
                this,
                "Notification permission denied. Some notifications may not appear.",
                Toast.LENGTH_LONG
            ).show()
        }
    }
```

Prática No arquivo principal **MainActivity**

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    enableEdgeToEdge()
    setContentView(R.layout.activity_main)

    createChannelNotification()
    checkNotificationPermission()

    val btnStartservice = findViewById<Button>(R.id.btnStartservice)
    btnStartservice.setOnClickListener {
        Toast.makeText(this, "Clicked", Toast.LENGTH_SHORT).show()
        val intent = Intent(this, MyService::class.java)
        startService(intent)
    }
}
```

Prática No arquivo principal **MainActivity**

```
private fun createChannelNotification(){
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
        val name = "Chanel Notification"
        val descriptionText = "Description Chanel"
        val importance = NotificationManager.IMPORTANCE_DEFAULT
        val channel = NotificationChannel(channelId, name, importance).apply {
            description = descriptionText
        }
        val notificationManager: NotificationManager =
            getSystemService(Context.NOTIFICATION_SERVICE) as NotificationManager
        notificationManager.createNotificationChannel(channel)

        val downloadChannelId = "download_channel"
        val downloadChannelName = "Download Channel"
        val downloadChannel = NotificationChannel(
            downloadChannelId,
            downloadChannelName,
            NotificationManager.IMPORTANCE_LOW
        )
        notificationManager.createNotificationChannel(downloadChannel)
    }
}
```

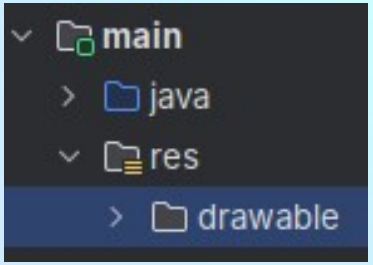
Prática No arquivo principal **MainActivity**

```
private fun checkNotificationPermission() {
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.TIRAMISU) {
        when {
            ContextCompat.checkSelfPermission(
                this,
                Manifest.permission.POST_NOTIFICATIONS
            ) == PackageManager.PERMISSION_GRANTED -> {}
            shouldShowRequestPermissionRationale(Manifest.permission.POST_NOTIFICATIONS) -> {
                Toast.makeText(
                    this,
                    "Notification permission is required to show progress",
                    Toast.LENGTH_LONG
                ).show()
                requestPermissionLauncher.launch(Manifest.permission.POST_NOTIFICATIONS)
            }
            else -> {
                requestPermissionLauncher.launch(Manifest.permission.POST_NOTIFICATIONS)
            }
        }
    }
}
```

Prática

Gerar o ícone ic_notification com Android Studio

1. Clique com o **botão direito** na pasta res > drawable
2. Vá em: **New > Image Asset**
3. Em **Asset Type**, selecione: Notification Icons
4. Escolha um ícone como:
   - ic_notifications
5. Clique em **Next > Finish**

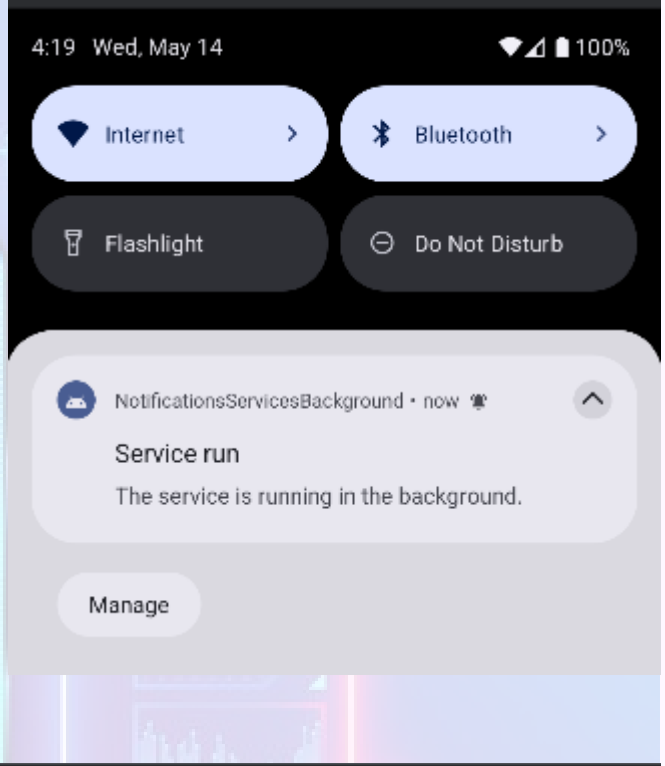


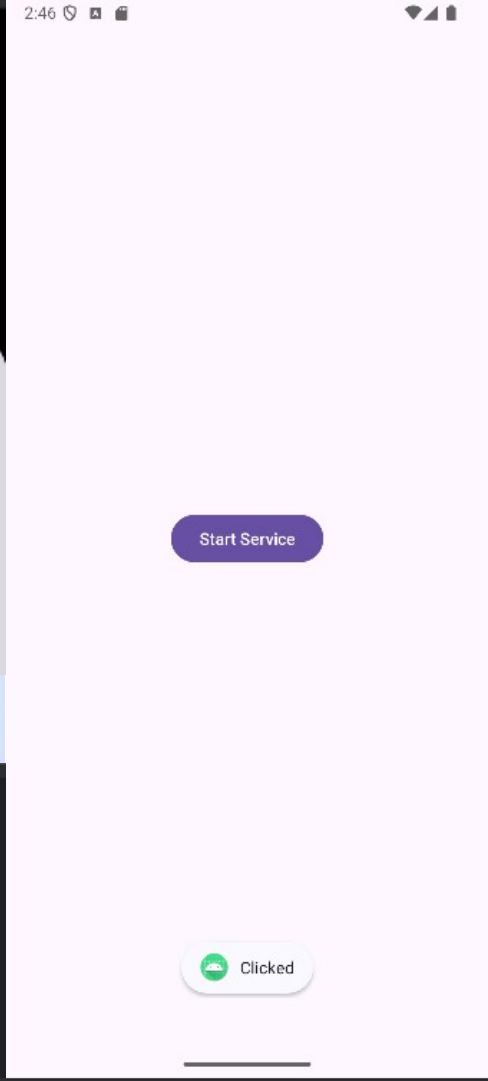
👉 Isso vai gerar um arquivo ic_notification.xml ou .png automaticamente em res/drawable.

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Prática

Run



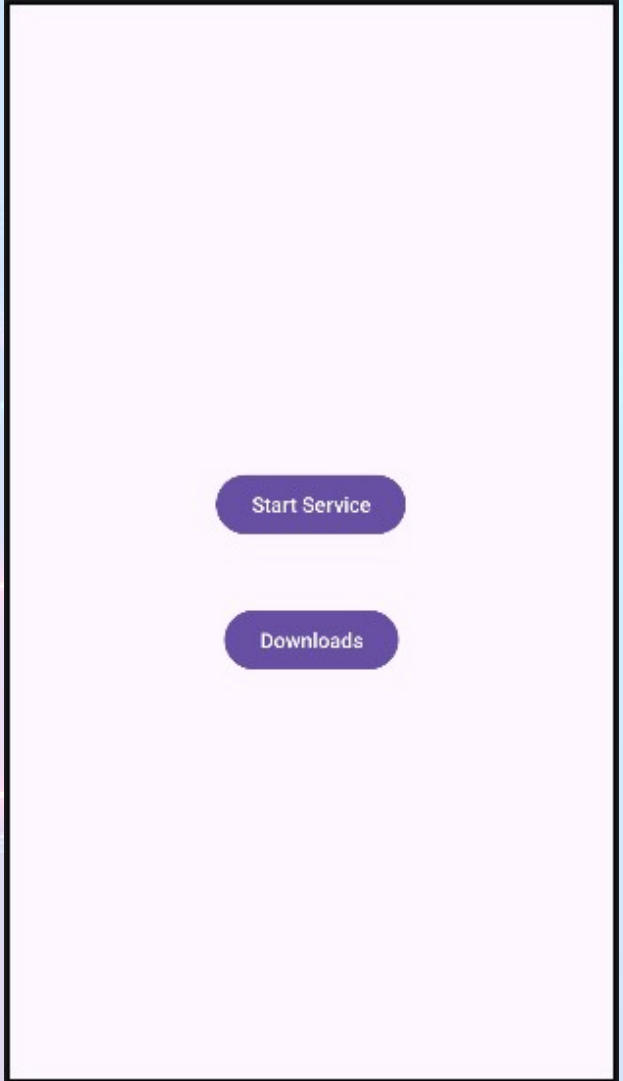



```
2025-05-14 14:46:54.351 12371-12441 Service            br....tificationsservicesbackground  D  Cont: 2
2025-05-14 14:46:55.352 12371-12441 Service            br....tificationsservicesbackground  D  Cont: 3
2025-05-14 14:46:56.353 12371-12441 Service            br....tificationsservicesbackground  D  Cont: 4
2025-05-14 14:46:57.353 12371-12441 Service            br....tificationsservicesbackground  D  Cont: 5
2025-05-14 14:46:58.354 12371-12441 Service            br....tificationsservicesbackground  D  Cont: 6
2025-05-14 14:46:59.355 12371-12441 Service            br....tificationsservicesbackground  D  Cont: 7
2025-05-14 14:47:00.356 12371-12441 Service            br....tificationsservicesbackground  D  Cont: 8
2025-05-14 14:47:01.356 12371-12441 Service            br....tificationsservicesbackground  D  Cont: 9
2025-05-14 14:47:02.357 12371-12441 Service            br....tificationsservicesbackground  D  Cont: 10
```

Foreground Services

Adicione mais um botão

Esse botão será utilizado para simular um download.



Foreground Services

Criar o Serviço (classe Kotlin) **DownloadService**

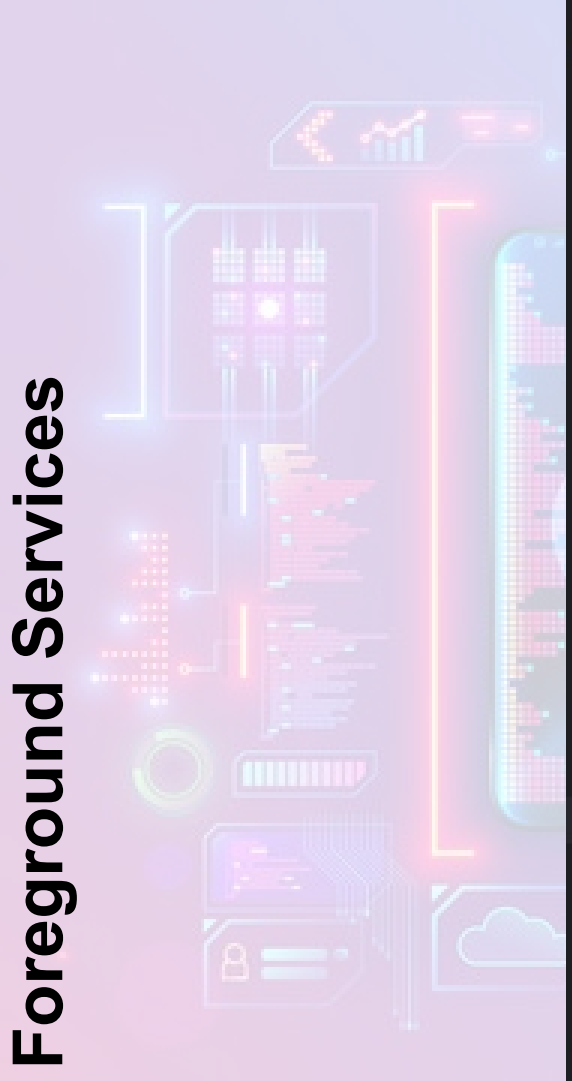
```kotlin
class DownloadService : Service() {
    private val channelId = "download_channel"

    override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {
        createNotificationChannel()

        val notification = NotificationCompat.Builder(this, channelId)
            .setContentTitle("Downloading file...")
            .setContentText("Progress: 0%")
            .setSmallIcon(R.drawable.ic_download)
            .setPriority(NotificationCompat.PRIORITY_LOW)
            .build()

        // Use the appropriate foreground service type for Android 13+
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.Q) {
            startForeground(1, notification, ServiceInfo.FOREGROUND_SERVICE_TYPE_DATA_SYNC)
        } else {
            startForeground(1, notification)
        }
```

Foreground Services



```
override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {
    ...
    Thread {
        for (i in 1 ≤ .. ≤ 100 step 10) {
            Thread.sleep(1000)
            val updated = NotificationCompat.Builder(this, channelId)
                .setContentTitle("Downloading file...")
                .setContentText("Progress: $i%")
                .setSmallIcon(R.drawable.ic_download)
                .setPriority(NotificationCompat.PRIORITY_LOW)
                .build()

            if (checkNotificationPermission()) {
                NotificationManagerCompat.from(this).notify(1, updated)
            }
        }

        stopForeground(true)
        stopSelf()
    }.start()

    return START_NOT_STICKY
}
```

Foreground Services

```
private fun createNotificationChannel() {
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
        val channel = NotificationChannel(
            channelId,
            "Download Channel",
            NotificationManager.IMPORTANCE_LOW
        )
        val manager = getSystemService(Context.NOTIFICATION_SERVICE) as NotificationManager
        manager.createNotificationChannel(channel)
    }
}
```

Foreground Services

```
private fun checkNotificationPermission(): Boolean {
    return if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.TIRAMISU) {
        ActivityCompat.checkSelfPermission(
            this,
            android.Manifest.permission.POST_NOTIFICATIONS
        ) == PackageManager.PERMISSION_GRANTED
    } else {
        true // For Android 12 and below
    }
}

override fun onBind(intent: Intent?): IBinder? = null
```

Foreground Services

Adicionar ao AndroidManifest.xml

```
        <service
            android:name=".MyService"
            android:foregroundServiceType="dataSync"
            android:exported="false" />

        <service
            android:name=".DownloadService"
            android:enabled="true"
            android:exported="false"
            android:foregroundServiceType="dataSync" />

    </application>

</manifest>
```

Foreground Services

Iniciar o serviço a partir da Activity

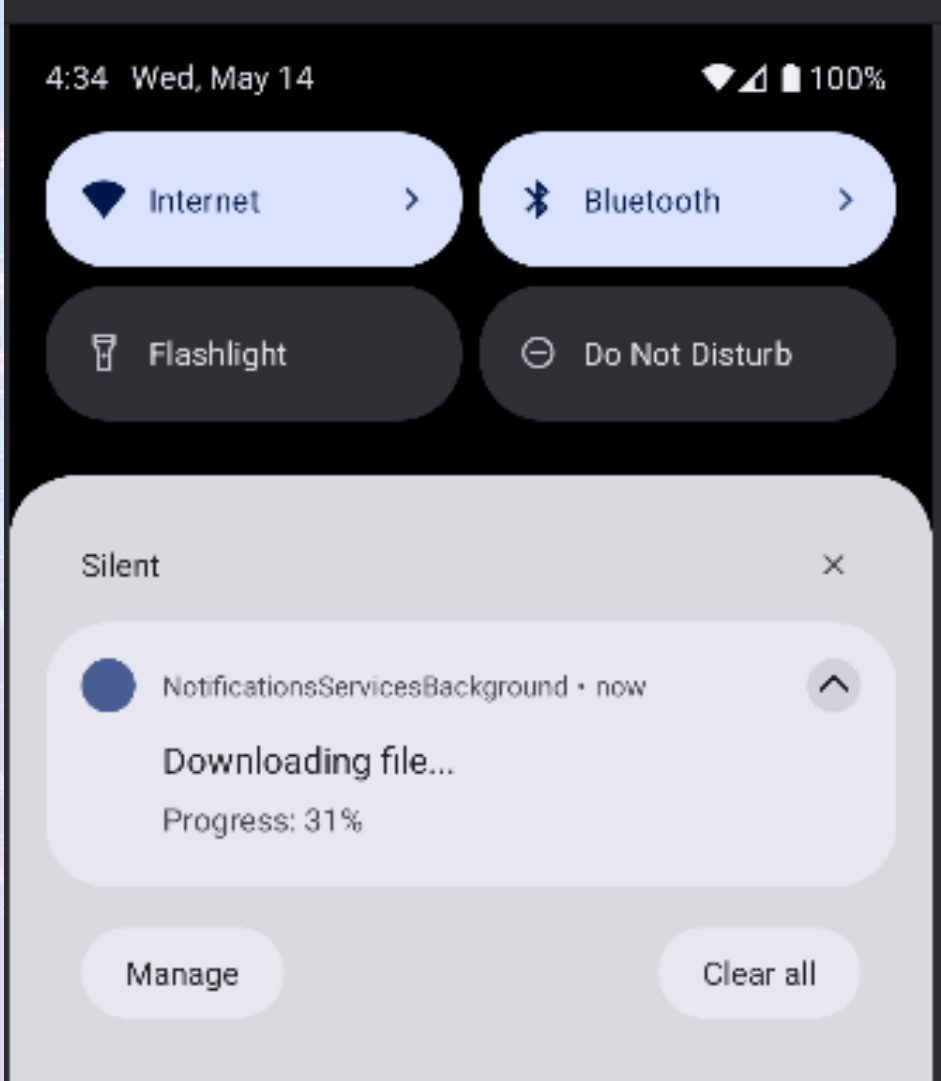
```
override fun onCreate(savedInstanceState: Bundle?) {
    ...

    val btnDownloads = findViewById<Button>(R.id.btnDownloads)
    btnDownloads.setOnClickListener {
        Toast.makeText(this, "Clicked", Toast.LENGTH_SHORT).show()
        val intent = Intent(this, DownloadService::class.java)
        startService(intent)
    }
}
```

Foreground Services

Run App - Simulation downloading file …

Show, notification …



Foreground Services

🔎 foregroundServiceType informa ao sistema o propósito do serviço.

Exemplos:

- location
- mediaPlayback
- dataSync
- connectedDevice

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Foreground Services

🧠 Boas práticas com ForegroundService

- Use **apenas quando for realmente necessário** manter algo rodando continuamente.
- **Encerre o serviço assim que a tarefa for concluída** para liberar recursos.
- Mantenha a notificação informativa, clara e objetiva.
- Nunca use como substituto de tarefas agendadas (para isso, prefira WorkManager).

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Foreground Services

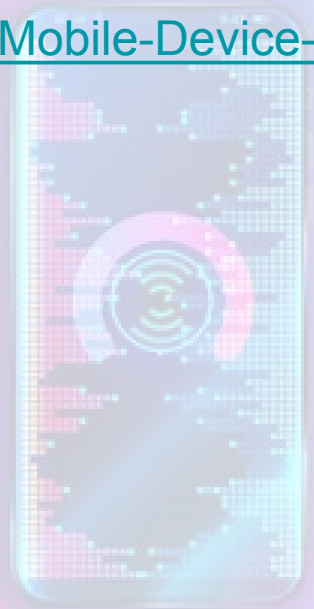
📌 Conclusão

- ForegroundService é ideal para tarefas essenciais e contínuas.
- Ele é mais confiável do que um Service comum, mas **deve sempre ser acompanhado de uma notificação**.
- É amplamente usado em apps de música, mapas, monitoramento e backup.

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Github

https://github.com/Mobile-Device-Development/NotificationsServicesBackground



UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Bibliografia

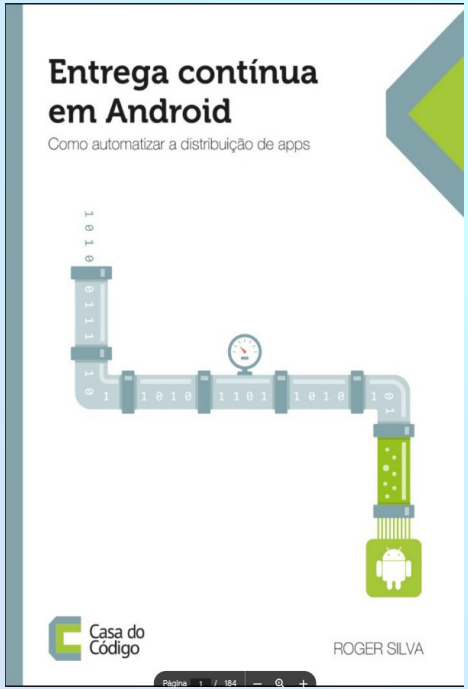
RESENDE, Kassiano. Kotlin com android: crie aplicativos de maneira fácil e divertida. São Paulo, SP: Casa do Código, 2018. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 26 fev. 2025.

https://plataforma.bvirtual.com.br/Acervo/Publicacao/213016

SILVA, Roger. Entrega contínua em android: como automatizar a distribuição de apps. São Paulo, SP: Casa do Código, 2016. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 26 fev. 2025.

https://plataforma.bvirtual.com.br/Acervo/Publicacao/212670

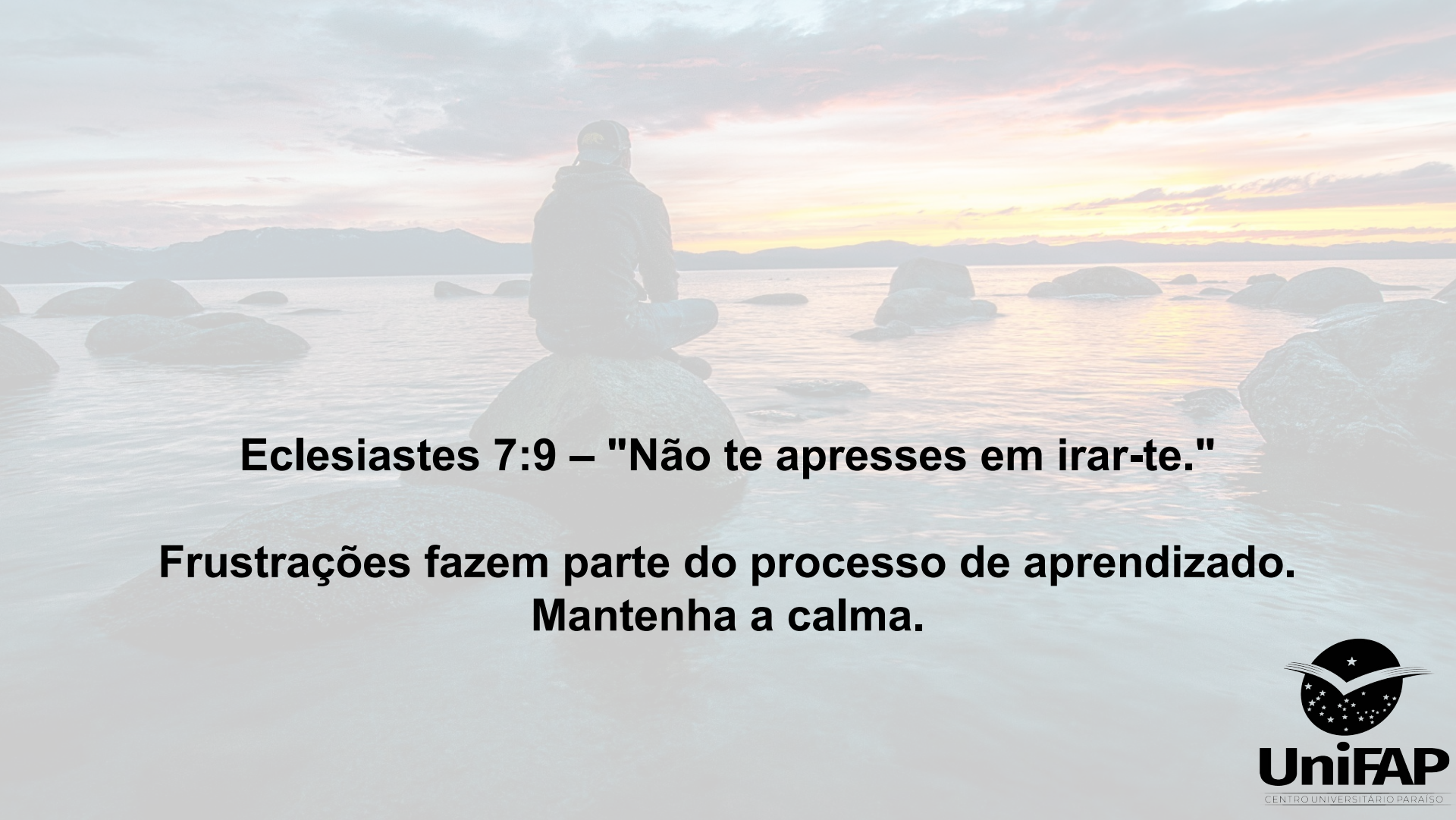




Perguntas?!



UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO



Eclesiastes 7:9 – "Não te apresses em irar-te."

Frustrações fazem parte do processo de aprendizado.
Mantenha a calma.

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO



Obrigado!

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO