GAT 108 – Automação Avançada

Marcus Vinícius Oliveira Pacheco

AV1
Relatório de desenvolvimento de aplicativo para monitoramento da velocidade de um veículo em tempo real

LAVRAS - MG
JUNHO DE 2023

INTRODUÇÃO

Nesse relatório, será abordado sobre como foi feito o desenvolvimento do aplicativo *SpeedMonitoring*, que funciona como uma calculadora de velocidade ideal. Sua base é através do fornecimento de dados do gps do dispositivo. Dessa forma, o usuário entra com a distância que irá percorrer e o tempo máximo que quer chegar no destino, então o aplicativo monitora em tempo real a distância percorrida, a distância restante, a velocidade média, o tempo de deslocamento e calcula o tempo estimado de chegada ao destino, o consumo de combustível e uma velocidade recomendada para chegar ao destino no tempo desejado e gastando menos combustível.

O aplicativo foi todo desenvolvido em Java através da IDE Android Studio, utilizando da máquina virtual disponibilizada para os testes e demonstrações de funcionamento. Todo o código pode ser acessado pelo repositório no Github através desse link, bem como um vídeo de explicação de seu funcionamento no YouTube.

INTERFACE

A interface do aplicativo é bem simples e consta com apenas um layout, feito em XML na ferramenta de design do Android Studio no formato LinearLayout. No mesmo contém um título de função do app, duas entradas de texto para o usuário, um botão de início/parada e seis textos correspondentes às informações fornecidas. O Layout pode ser conferido na imagem:

CÓDIGO

As bibliotecas utilizadas são em maioria padrões ou para facilitar o uso de elementos da interface, como o *android.widget.TextView* e o *android.widget.EditText*, por exemplo. Porém, as bibliotecas *android.location.Location, android.location.LocationListener e android.location.LocationManager*, que servem para utilizar do serviço de localização do Android.

O código se resume a basicamente uma classe infelizmente, pois o uso dessas bibliotecas só foi possível dentro do *MainActivity*, principalmente nos métodos de request. Algumas tentativas foram feitas usando um thread separado (utilizando um Listener para interface e tentar resolver o problema das chamadas das funções do Location que requeriam um objeto de contexto/atividade), a biblioteca recomendada da maddevs e outras versões dos serviços de localização, como o FusedLocation, mas nenhum dos esforços funcionou adequadamente, sendo mais por conta de fechamento do aplicativo e sem problemas de compilação.

No *MainActivity(),* são primeiramente declarados os atributos de forma privada relacionados a cada elemento da interface, as variáveis que recebem os valores calculados, uma variável de estado para saber se o monitoramento está ativo e algumas variáveis para a biblioteca Location.

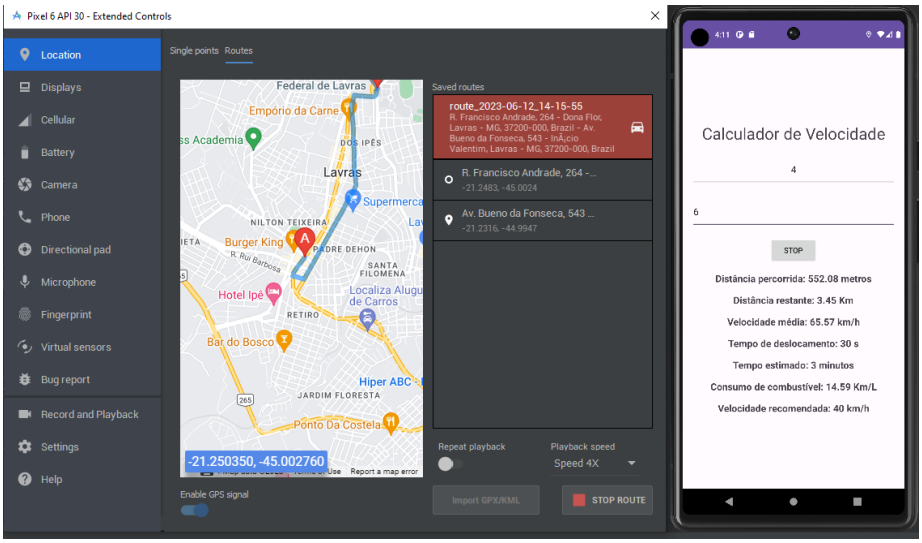
O construtor basicamente correlaciona os atributos com os elementos do Layout e inicializa o LocationManager, também fazendo o request de permissão de uso da localização do dispositivo. O método *startStopTracking()* é chamado a partir do clique no botão. Caso o monitoramento esteja desativado, ele muda a variável de estado para indicar o inicio do serviço e chama a função startTracking(). Caso já esteja ativo, ele muda a variável de estado para indicar que o monitoramento está desligado e chama a função stopTracking().

No startTracking(), é conferido se foi dada a permissão de acesso à localização do celular, inicia algumas variáveis utilizadas para as contas e faz o request de updates do gps. Quando ocorre um update, é gerado um evento ouvido pelo LocationListener que dispara a função onLocationChanged(). Ela pega a localização recebida e compara com a última através do método distanceTo(), atualizando a distância percorrida e calculando o consumo de combustível em litros a partir da velocidade instantânea fornecida. Apesar do artigo da Quatro Rodas mostrar um melhor consumo nos 80Km/h, como para simulação não é possível chegar nessa velocidade, por fins de teste foi adotado que veículos em velocidades até 40Km/h fazem em média 21Km/L, até 70Km/h fazem 15Km/L e a partir disso fazem 10Km/L. Por fim, é também calculado o tempo de atividade do monitoramento, a velocidade média e são chamadas as funções de update da interface;

No StopTracking(), as variáveis utilizadas para os cálculos são resetadas e textos da interface têm os valores limpos. Ainda sobram alguns métodos que precisam ser sobrescritos da biblioteca Location, mas que não são utilizados aqui e por isso ficam vazios. Nas funções de update, são feitos os cálculos das variáveis a serem exibidas e suas respectivas atualizações na tela. Por fim, a função onRequestPermissionsResult() que trata o pedido de permissão ao acesso da localização do dispositivo

TESTE

Os testes foram feitos com um dispositivo virtual que emula um Google Pixel 6, visto que nele é possível simular velocidades mais adequadas ao trabalho que são mais próximas de um veículo. A imagem a seguir mostra a interface em utilização e a rota simulada para testar o aplicativo em tempo real.



CONCLUSÕES

O aplicativo funciona corretamente, atendendo os requisitos de ter uma interface gráfica para a visualização dos resultados e para acoleta de dados importantes para a execução, a leitura dos dados de localização disponível no smartphone, além de gerar e apresentar na tela osdados consolidados de Velocidade Média, Tempo de Deslocamento, Distância Percorrida, Consumo de Combustível, Tempo para chegar ao destino e velocidade recomendada.

Porém, não foi possível o uso de Thread para a coleta de dados conforme já foi explicado anteriormente. Ainda consta o arquivo Recon.java no repositório, que nada mais é que a implementação da reconciliação de dados para o cálculo da velocidade recomendadaa partir de um Thread, contando 10 nós para o trajeto medidos em tempo. O código compila e funciona com o MainActivity.java (que ainda contém os trechos utilizados para o funcionamento comentados no código) , ocorrendo tudo certo nos primeiros momentos. Depois da atualização dos nós, o aplicativo simplesmente fecha e mostra um erro na ação de Clique no botão, dificultando a localização real do problema para a solução. De qualquer forma, está disponível para a consideração da tentativa bem como recomendações e dicas para o funcionamento correto do Thread.