Documentação do Projeto: SWIFT

Alunos:

Natan Batalha de Araújo

Gustavo Horning

Ricardo Hammerschmidt

Lógica de Construção:

1. Criação de Protocolo ***Notificável*** com a propriedade mensagem e o método enviarNotificação().

**Modelagem de Struct para diferente tipos de canais: E-mail, SMS e “push notification”**

Cada um desses é um tipo de dados diferentes, mas todos devem seguir o mesmo protocolo, a mesma modelagem, o qual seria o notificável, e usarmos Struct para isso.

Ele organiza os dados que cada tipo precisa e permite definir o comportamento de cada tipo.

1. O Segundo passo foi dar continuidade as demais *struct* do canal:

Já temos o Email, continuaremos para SMS e “Push Notification”, os quais irão seguir o mesmo protocolo Notificável, cada um com sua propriedade exclusiva.



1. Aperfeiçoamento do sistema de mensagens usando ENUM:

Definimos o enum que deve representar os tipos possíveis de mensagens que podem ser enviadas. Os casos que definimos ajudam a caracterizar notificações de acordo com seu propósito

case promoção – Usado para mensagens de ofertas, descontos, ações de marketing

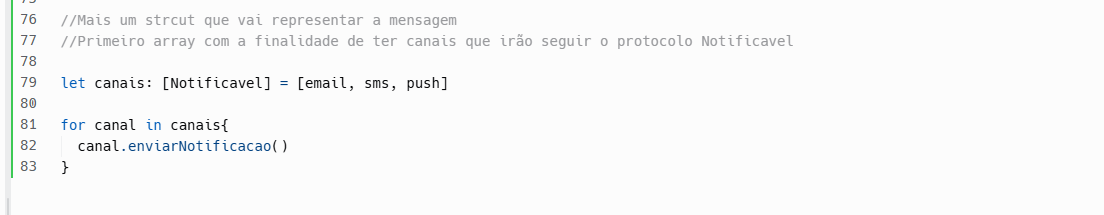
case lembrete – Notificações com lembretes importantes para o usuário

case alerta – Avisos urgentes ou críticos que exigem atenção imediata

O novo Struct que fizemos na imagem abaixo, representa o conteúdo de uma mensagem enviada por um canal de notificação, combinando o tipo da mensagem com o texto a ser enviado.



1. Definimos a importância de criar um Array que guarda vários canais de notificação diferentes (Email, SMS, Push), desde que todos conformem ao protocolo principal (“Notificável”) junto alinhando uma lógica de execução com loop



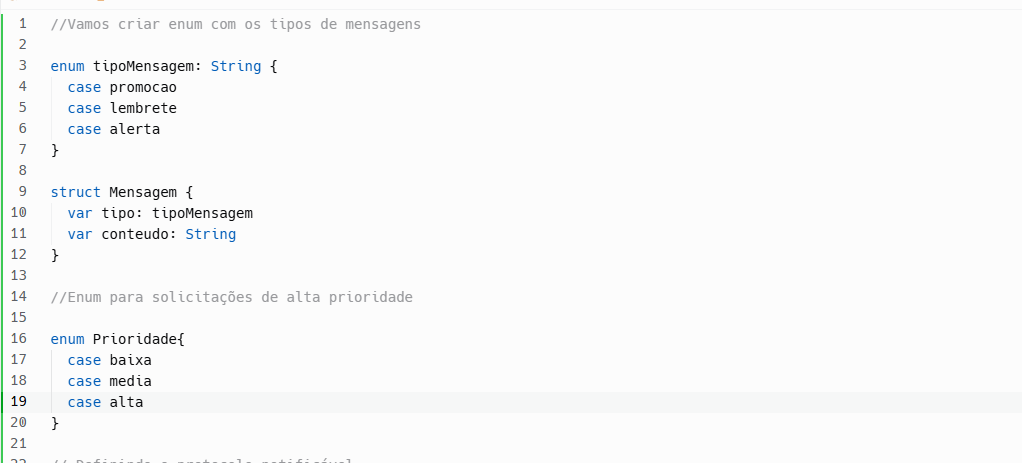
Como somos alunos dedicados, iremos entregar mais do que é pedido. Definimos os desafios adicionais.

1. Vamos definir as prioridades das notificações.

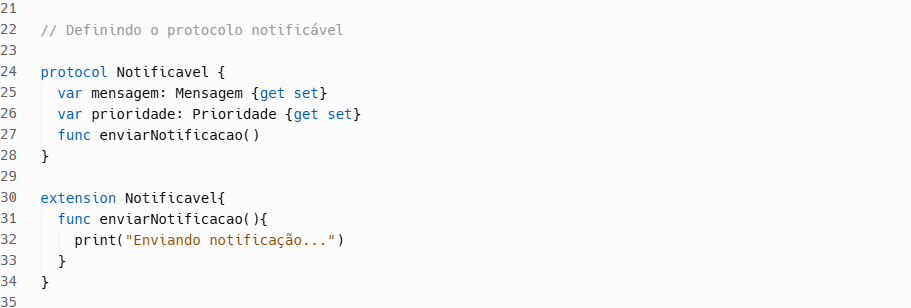
Nesse caso decidimos criar um novo Enum Prioridade com os casos: baixa, média e alta.

Depois não poderíamos não colocar no protocolo Notificável e ATUALIZAR os canais (Email, SMS e PushNotification) para considerar a prioridade na hora de enviar a notificação.

e como solicitado na atividade, fazer com que as solicitações de **alta prioridade** mostrem **URGENTE** no console.



Fizemos as modificações para incluir no protocolo notificável:



Por fim, iremos definir a lógica para o **Filtro por tipo de canal:**

Iremos criar uma Function que recebe um array de notificável e um tipo específico de canal e retorna os canais do tipo escolhido.

Essa parte mostra que entendemos os conceitos ensinados pelo professor sobre “protocol composition” e “type casting”

1. Por fim fizemos as demais filtragem para os outros tipos de canais e deixamos comentados caso o professor solicite

