Mandando notificações pelo CloudKit

PARTE 1

A forma mais fácil de mandar notificação que você vai ver na vida

① Você já tem CloudKit no seu aplicativo?

Muito bom! Porque a proposta aqui não é ensinar como colocar cloudKit, e sim só as notificações.

② Beleza! E como que eu coloco as notificações?

Primeira coisa a fazer é habilitar Push Notifications ao projeto. Conseguimos fazer isso dentro do AppDelegate, na função de didFinishLauchingWithOptions (é uma função que já tem no AppDelegate, só coloca isso lá, eu espero aqui)

PERMISSÃO

```
// Pede permissão para mandar notificações
        UNUserNotificationCenter.current().requestAuthorization(options:
[.alert, .badge, .sound], completionHandler: { authorized, error in
          if authorized {
            DispatchQueue.main.async(execute: {
              application.registerForRemoteNotifications()
            })
          }
        })
```

③ OK! Tudo certo então? Posso continuar fazendo minhas coisas?

DELEGATE

Calma lá! Ainda tem alguns passos, como definir como serão nossas notificações, e em qual momento elas devem ser mandadas. Pra isso, devemos colocar o delegate do UNUserNotificationCenter. Então, também dentro do didFinishLauchingWithOptions, colocamos:

```
UNUserNotificationCenter.current().delegate = self
```

Ótimo! Agora vamos criar uma extensão do AppDelegate para conformar com o UNUserNotificationCenterDelegate

```
// Essa função é para quando recebemos notificação
    func userNotificationCenter(_ center: UNUserNotificationCenter, willPresent
notification: UNNotification, withCompletionHandler completionHandler: @escaping
(UNNotificationPresentationOptions) -> Void) {

        // show the notification alert (banner), and with sound
        completionHandler([.alert, .sound, .badge])
    }

    // Essa função é chamada quando o usuário clica na notificação
    func userNotificationCenter(_ center: UNUserNotificationCenter, didReceive
response: UNNotificationResponse, withCompletionHandler completionHandler: @escaping
() -> Void) {
        completionHandler()
    }
```

④ Tá, já to pronto para receber notificação, mas cadê a notificação?

Vamos criar ela agora mesmo! Usando o CloudKit, temos que fazer um "subscription" para ver alterações que foram feitas no CloudKit. Ou seja, quando algo é alterado na nuvem (seguindo alguns parametros) podemos receber uma notificação dessas alterações.

SUBSCRIPTION

```
//Função onde nos registramos para receber notificações, ainda dentro do AppDelegate
func application(_ application: UIApplication,
didRegisterForRemoteNotificationsWithDeviceToken deviceToken: Data) {

let subscription = CKQuerySubscription(recordType: "oNomeDoSeuRecord", predicate:
NSPredicate(format: "SeuPredicado"), options: .firesOnRecordCreation)

}
```

Agora vamos por partes:

recordType: Qual o record do CloudKit que desejamos observar.

predicate: O "filtro" que queremos usar na hora de receber notificações, ou seja, não precisamos receber tudo que vem do cloudKit, mas sim só aquilo que vem do predicate. Se quisermos TODAS as alterações, podemos usar o format: "TRUEPREDICATE"
Para entender mais de predicates, aí vai um site f*da:
https://nspredicate.xyz/#predicate-format-and-arguments

options: Diz respeito a ação executada no CloudKit para receber a notificação. No caso, .firesOnRecordCreation é disparado quando criamos um novo record no CloudKit. Outro caso famoso que podemos usar, é o .firesOnRecordUpdate, que é quando o record é alterado.

Agora, vamos criar o formato da nossa notificação. Criamos um NotificationInfo() para colocar o que desejamos na notificação, e depois passamos de volta para o subscription

EDITAR NOTIFICAÇÃO

```
let info = CKSubscription.NotificationInfo()
info.alertBody = "isso é um aviso" //Corpo da notificação
info.title = "isso é um titulo"    //Titulo da notificação

// Incrementa o número de badges
info.shouldBadge = true

// O som usado na notificação
info.soundName = "default"

//Colocar as informações da notificação na nossa subscription criada anteriormente
subscription.notificationInfo = info
```

Por fim, registramos nossa subscription para o CloudKit, e estamos prontos! :)

REGISTRAR

```
CKContainer.default().publicCloudDatabase.save(subscription, completionHandler: {
subscription, error in
            if error == nil {
                print("Deu certo! Você receberá notificações!")
                // Subscription saved successfully
            } else {
                print("PUTS Error: \(error)")

            }
        })
```

Pronto! Agora você já tem tudo que é necessário para lidar com notifications no iOS!

Referências

https://fluffy.es/push-notification-cloudkit/#capabilities

https://www.hackingwithswift.com/read/33/8/delivering-notifications-with-cloudkit-push-messages-ckquerysubscription

Felipe Kaça Petersen

iOS