Vencendo a LGPD com Windows 💻

ou sendo derrotado...

Sobre a LGPD

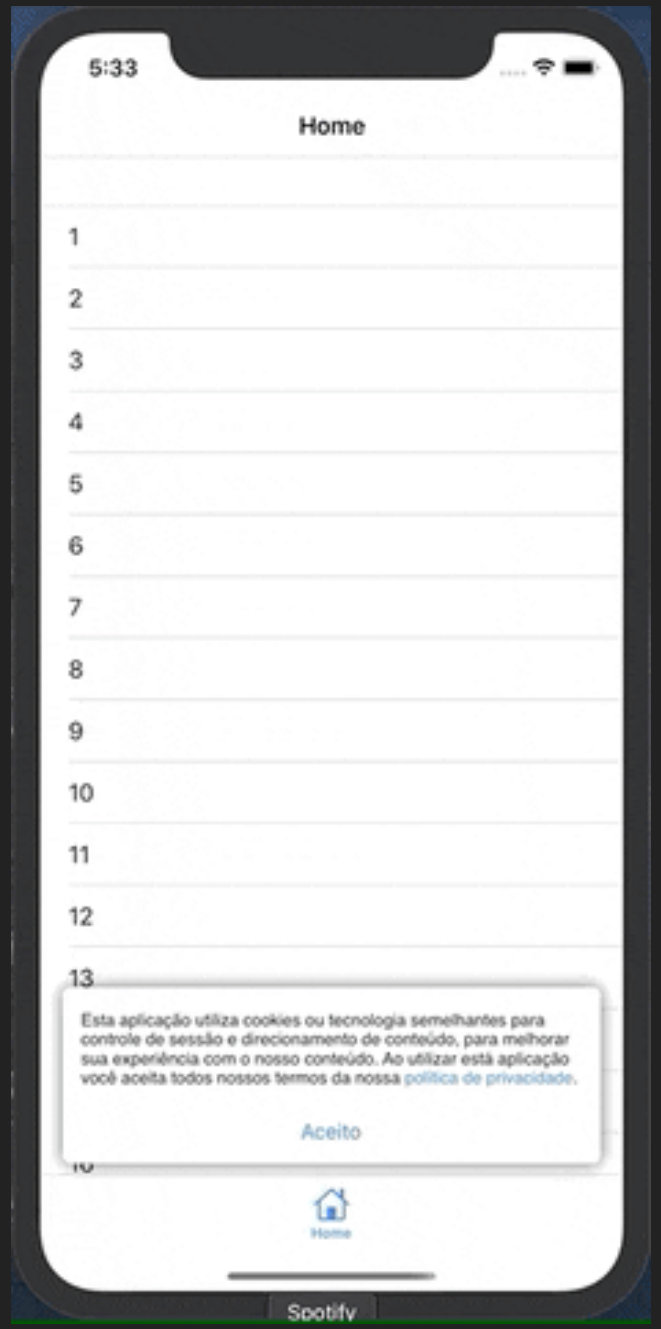
Com a chegada da LGPD as aplicações tem que informar de forma clara e objetiva com qual finalidade os dados coletados do usuário serão utilizados, obtendo o consentimento do usuário por meio de um opt-in.

Uma forma de fazer isso é utilizando o famoso **Cookie Banner**

O desafio...

Apresentar uma `view` por cima de toda a aplicação

- mantendo a interação do app
- com **link** para redirecionamento
- botão de aceite



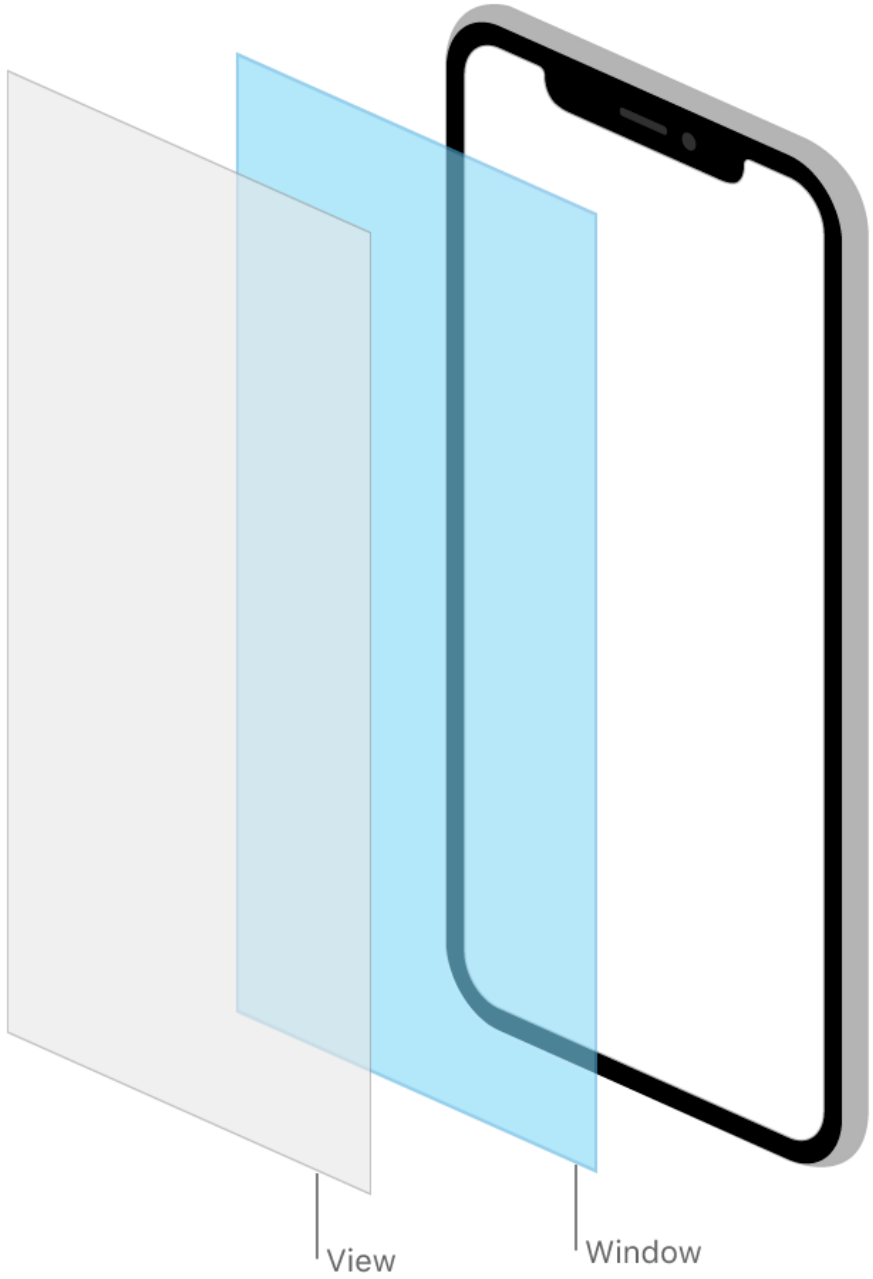
Pensando no problema...

Se eu fosse apresentar isso como filho de um *ViewController* eu teria que praticamente replicar essa integração em cada controller da minha aplicação. Eu quero que eu **instâncie uma única vez** e resolva para toda aplicação, ficando transparente para aplicação e que eu tenha certeza que não irá mexer na estrutura atual do app.

O que é a UIWindow

A primeira view criada na aplicação é uma **UIWindow**, logo depois os ViewControllers e depois a view dos controllers é adicionada a window e então esse é apresentado na tela.

Podemos dizer que é por causa da **UIWindow** que podemos ver a nossa aplicação na tela.



Como eu acesso as windows da aplicação?

```
// A window chave da sua aplicação
UIApplication.shared.keyWindow //UIWindow?

// todas as windows da sua aplicação
UIApplication.shared.windows //[UIWindow]
```

O que podemos fazer com isso?

```
extension UIApplication {

    class func topViewController(base: UIViewController? =
                                 UIApplication.shared.keyWindow?.rootViewController) -> UIViewController? {

        if let nav = base as? UINavigationController {
            return topViewController(base: nav.visibleViewController)
        }

        if let tab = base as? UITabBarController {
            if let selected = tab.selectedViewController {
                return topViewController(base: selected)
            }
        }

        if let presented = base?.presentedViewController {
            return topViewController(base: presented)
        }

        return base
    }

}
```

Como eu posso garantir que uma *UIWindow* vai sobre por outra?

Window.Level

Você já percebeu que dentro da controller da sua aplicação é impossível você colocar uma view acima da `statusBar`, de um `alerta` ou do `keyboard`?

Isso porque existem três níveis distintos de hierarquia da window:

- **normal**: esse é o nível da window principal da aplicação
- **statusBar**: neste nível, ela será apresentada por cima da statusBar mas abaixo dos `alertas` ou `keyboard`
- **alert**: esse é maior nível, sendo apresentado por cima de `keyboard` e `alertas`

Com isso podemos garantir que algo sobreponha nossa aplicação e até mesmo views do sistema.

Demo

- apresentando `window` no iOS 13+
- como criar e apresentar uma `window`
- visual debugging
- retirando uma `window` da stack

Como passar a interação do usuário para uma hierarquia abaixo da minha?

isUserInteractionEnabled = false

Demo

isUserInteractionEnabled

> If set to NO, this view (along with its subviews) is excluded from receiving touches. Touches on this view or one of its subviews "fall through" to a view behind it.
>
> > *Matt Neuburg - Programming iOS 5, p. 467*

Como inibir o toque em uma `view` específica da hierarquia?

hitTest:withEvent:

> The window object uses hit-testing and the responder chain to find the view to receive the touch event. In hit-testing, a window calls hitTest:withEvent: on the top-most view of the view hierarchy; this method proceeds by recursively calling pointInside:withEvent: on each view in the view hierarchy that returns YES, proceeding down the hierarchy until it finds the subview within whose bounds the touch took place. That view becomes the hit-test view.
>
> > *Apple Events Programming Guide*

```swift
import UIKit

final class UntouchableView: UIView {

    override func hitTest(_ point: CGPoint, with event: UIEvent?) -> UIView? {
        let result = super.hitTest(point, with: event)
        if result == self { return nil }
        return result
    }

}
```

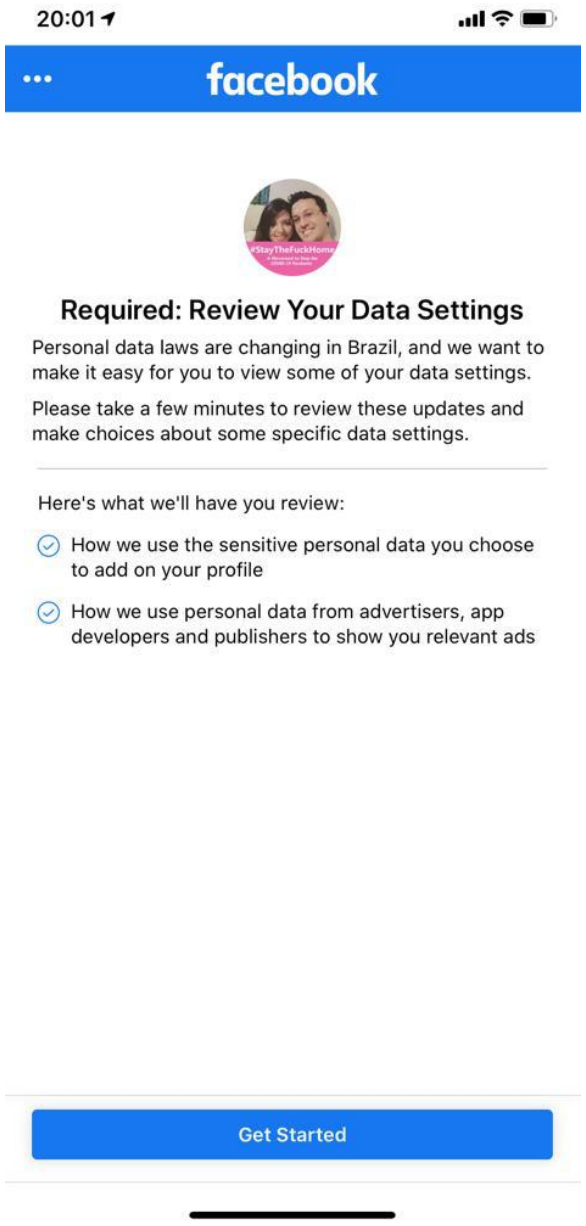
Demo

Recapitulando

- utilizar `UIWindow` quando quero algo apresentado sobre a aplicação:
  - banner, toast, loading, custom alert, etc...
- **isUserInteractionEnabled** tira a interação da `view` e `subviews`
- **hitTest:withEvent:** para tratar a interação em uma `view` específica

Mas essa é a única maneira de se apresentar essa info da LGPD?

A LGPD não especifica de forma clara como deve ser apresentada essa informação para o usuário, e o **Cookie Banner** é uma das maneiras de apresentar essa informação. Por exemplo, o Facebook apresenta uma tela que bloqueia a interação do usuário até que seja feita a revisão dessas informações pelo usuário.



Referências

Blog - Creating context menu with highlight

Stackoverflow - about isUserInteractionEnabled and HitTest

Stackoverflow - avoiding HitTest in a view

PDF - Apple events programming guide

Blog - UITextEffectsWindow

Gist - Sample, how to use marp to build keynote

Contato

tulio.bazan@dextra-sw.com