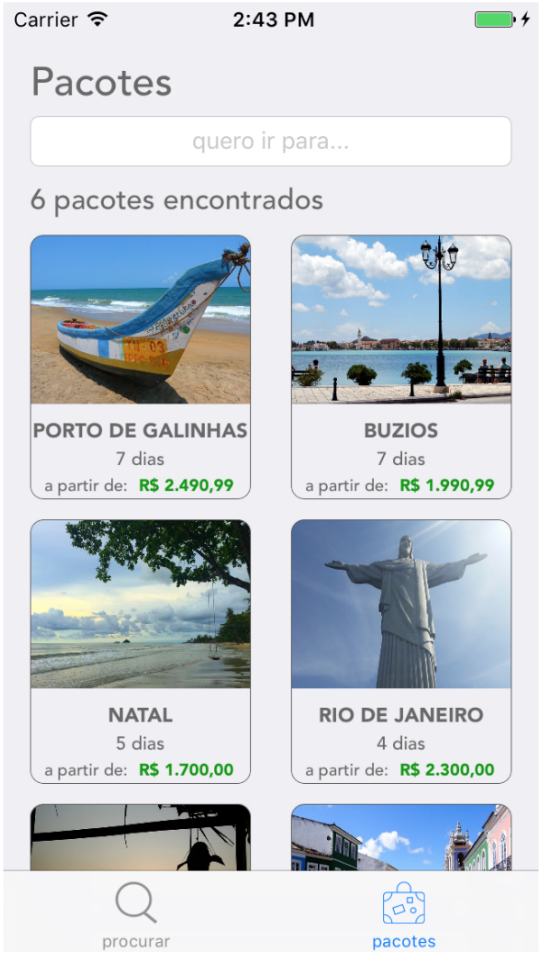
03

Iniciando a implementação com UIStackView

> Dica: Começando daqui ? baixe (https://github.com/alura-cursos/Alura-Viagens-parte1/archive/master.zip) o projeto do curso anterior

Agora que já temos a primeira tela do aplicativo 'Alura Viagens' implementada, é hora de dar continuidade ao projeto implementando novas 'features'.

Vamos analisar a próxima tela:

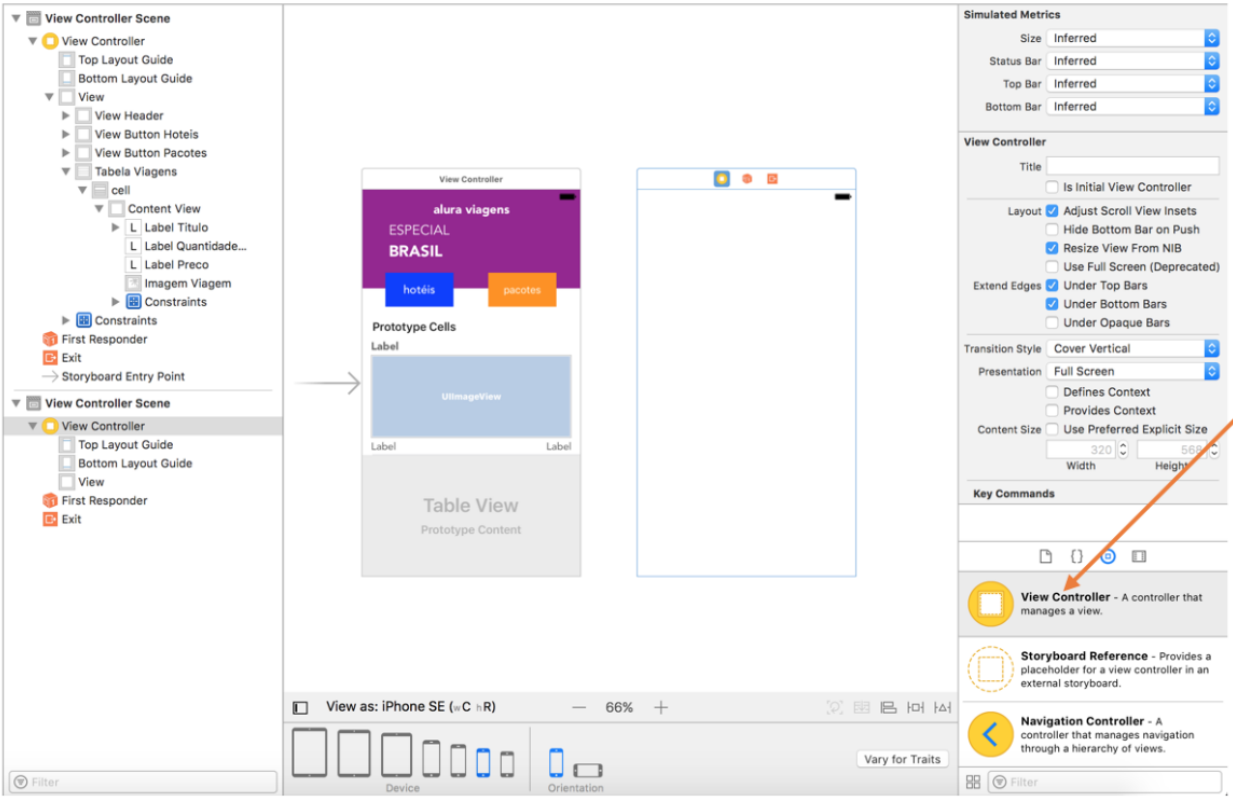


Essa tela apresenta diversos pacotes de viagens.

Assim como implementamos a primeira tela do aplicativo, vamos continuar a implementação utilizando a visualização de iPhone SE no storyboard.

Então vamos abrir o storyboard e selecionar a opção iPhone SE

Agora, precisamos criar uma nova tela na nossa aplicação. Como já estudamos no curso de Swift aqui da Alura, para criar uma nova tela no app basta arrastarmos um objeto de 'ViewController' do object library para o storyboard:



Pronto, já temos uma nova tela no nosso aplicativo.

Essa tela, assim como a primeira, deve se adaptar para todos os tamanhos de iPhone e iPad.

Agora é sua vez: vamos praticar colocando os elementos na tela:

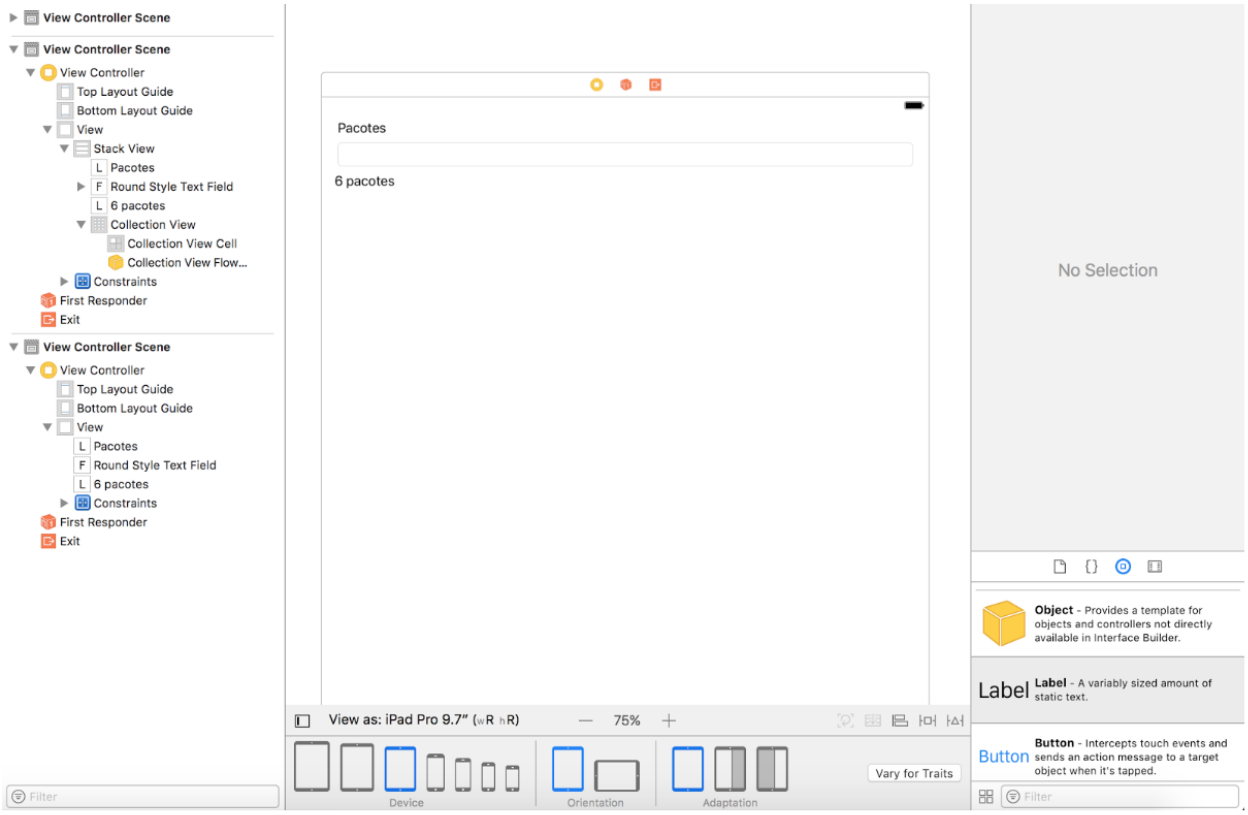
1- Label

2- Textfield

3- Label

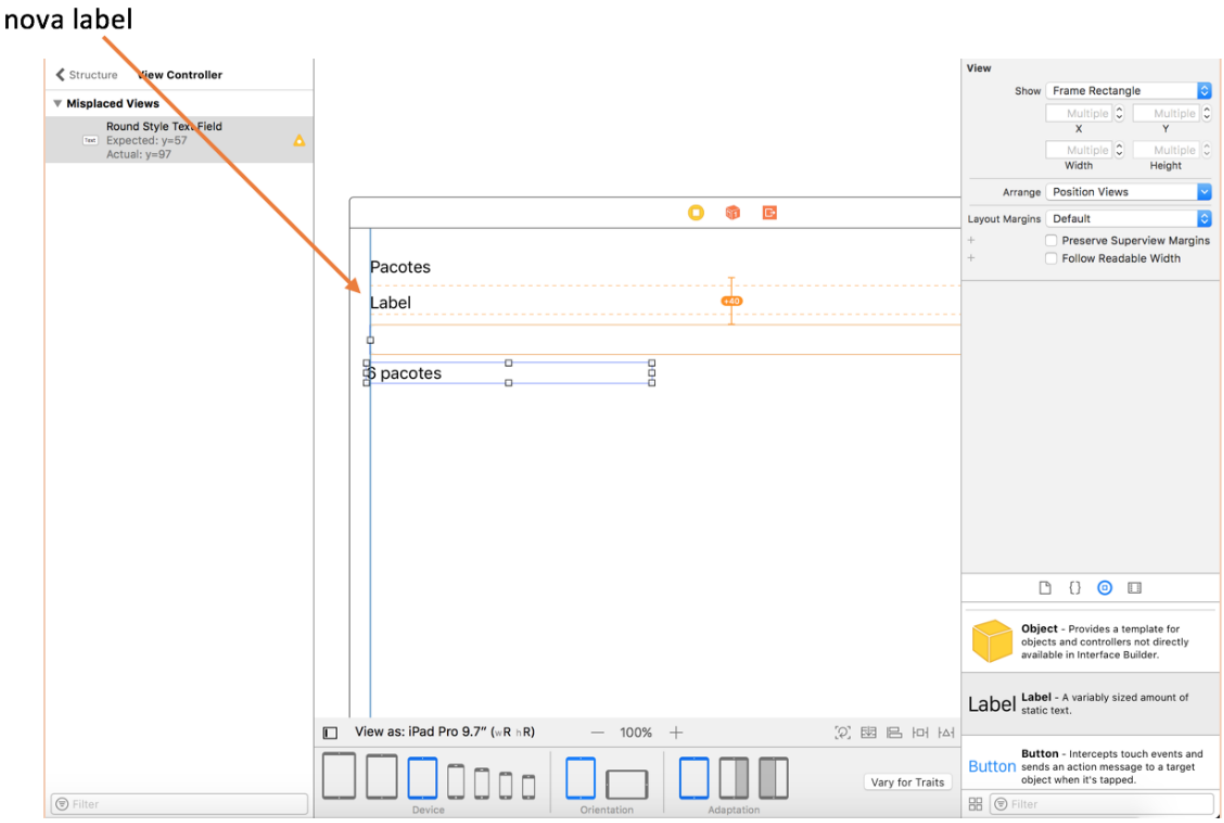
Aplique as constraints necessárias nos elementos.

Para testar, vamos mudar a visualização no storyboard para iPad:

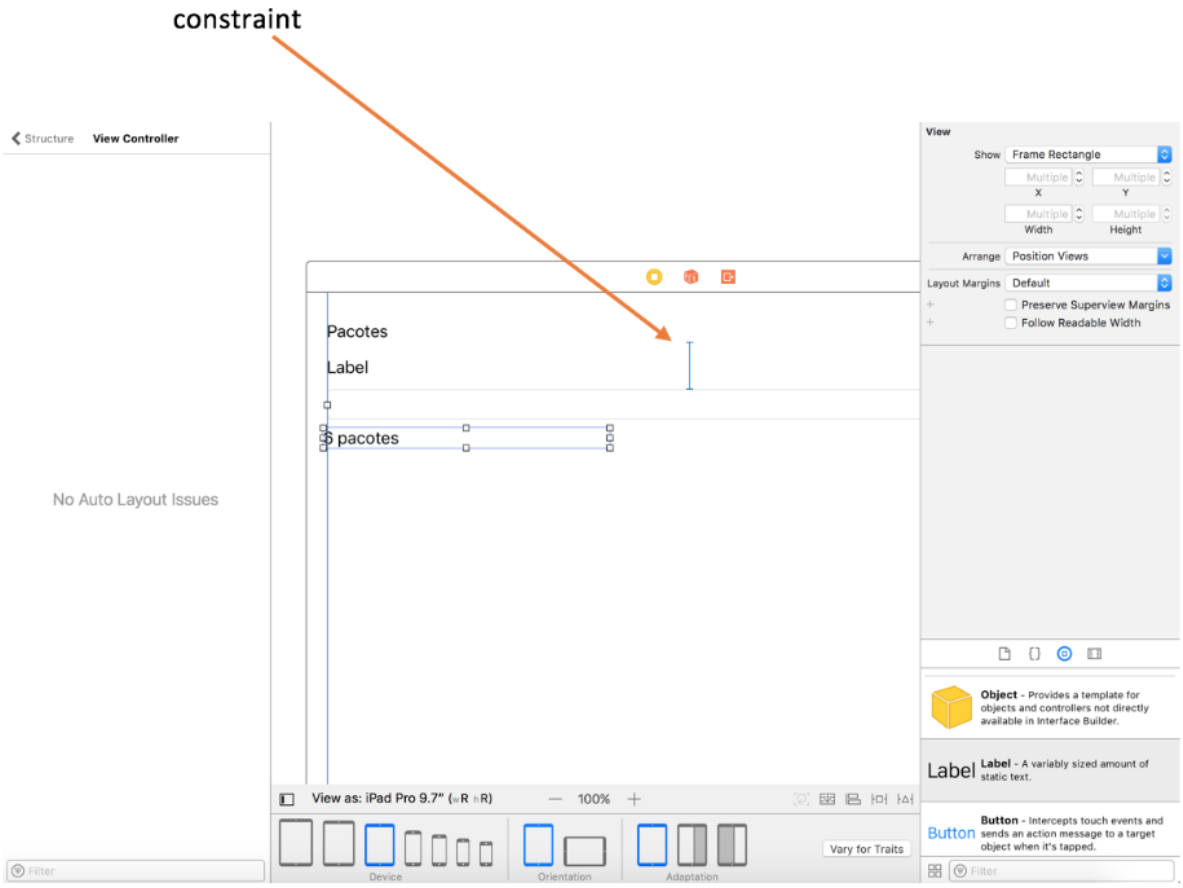


E se precisássemos colocar mais uma label entre a label 'pacotes' e o 'textfield'?

Uma das formas de resolver seria arrastar o textfield e a label '6 pacotes' para baixo, e aí colocar a nova label. Porém, precisaríamos apagar algumas constraints para recolocá-la novamente com base na posição da nova label.

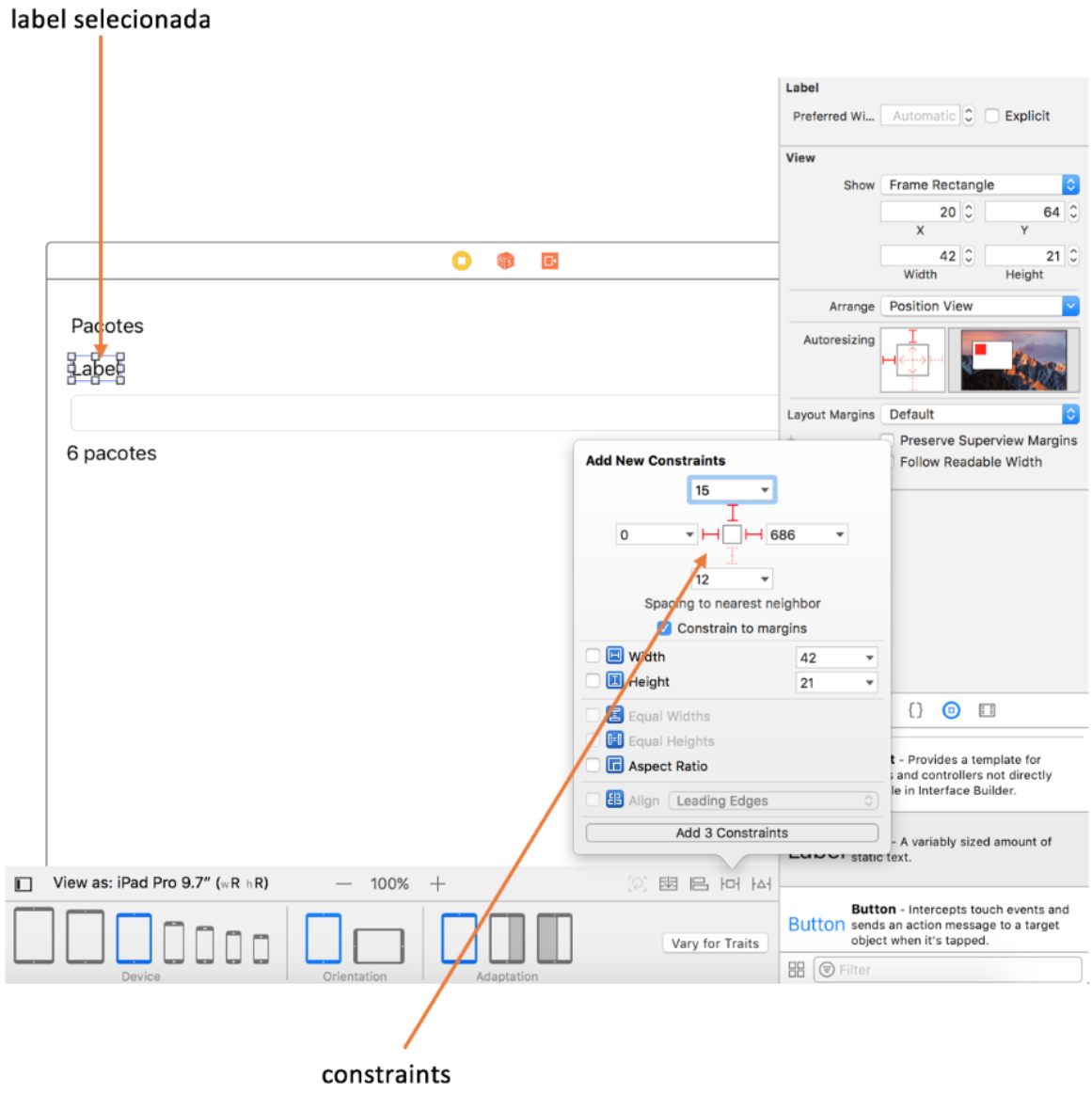


Atualizando a posição das constraints:

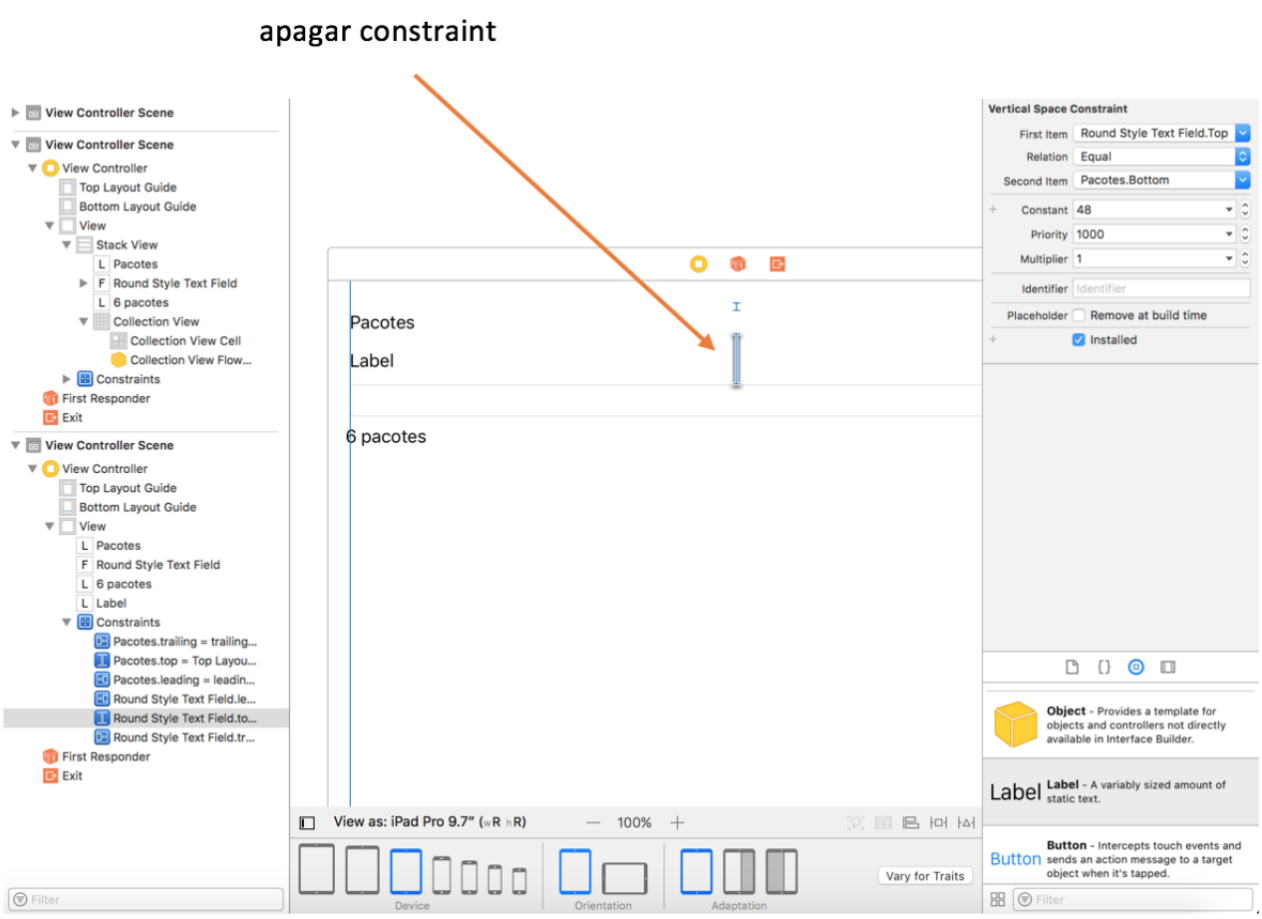


Repare que a constraint do topo do textfield está presa em relação a 'label pacotes'. Agora o correto seria prender a constraint de 'top' do textfield em relação a nova label.

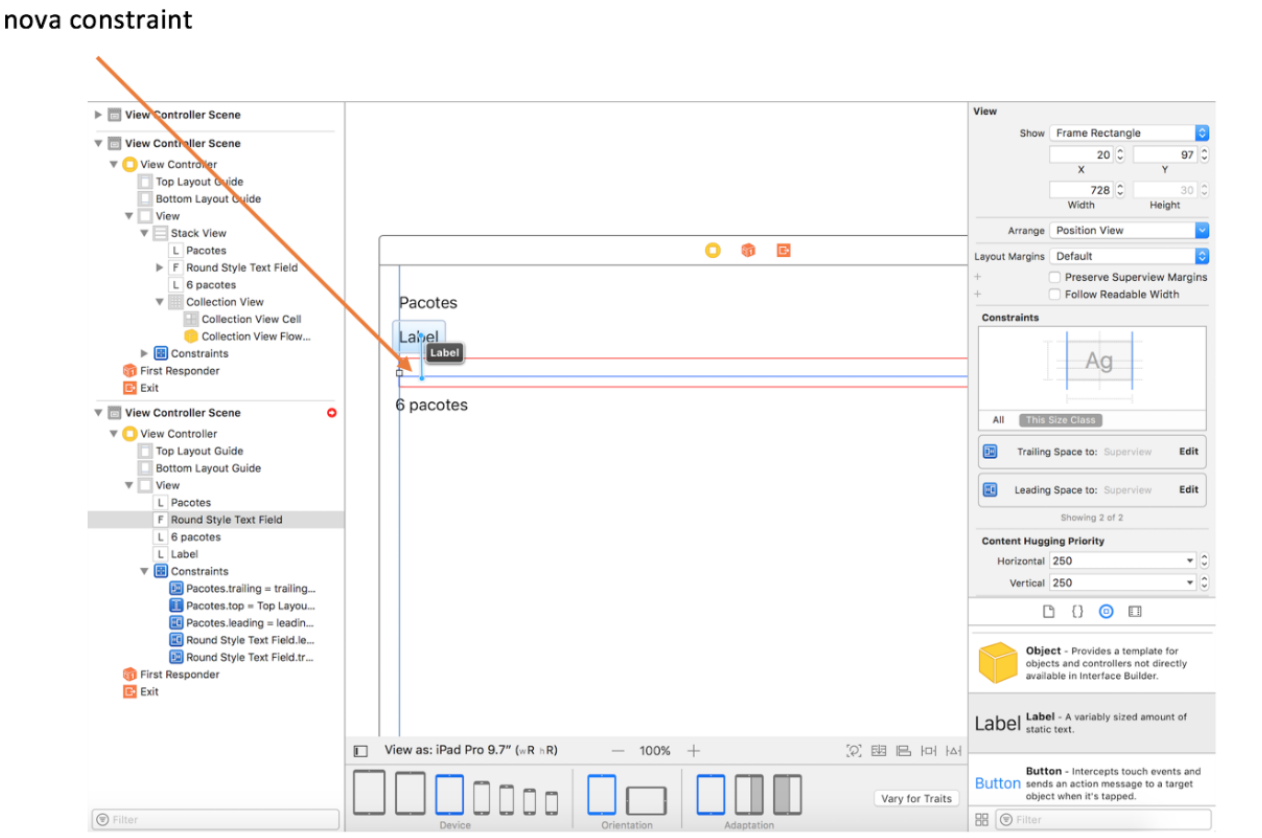
Para isso vamos colocar constraint na label que acabamos de colocar no ViewController:



Agora vamos apagar a constraint de 'top' do textfield e recolocá-la em relação a última label que criamos:



Aplicando a nova constraint do textfield:



Agora sim todos os elementos estão com as constraints corretas.

Repare que tivemos bastante trabalho para adicionar apenas uma label no layout do app. Isso porque o layout dessa tela está bem simples, imagine o trabalho que daria para mexer em elementos com constraints em uma tela complexa.

É muito comum no processo de desenvolvimento o layout do aplicativo sofrer mudanças. Se todas as vezes tivermos esse trabalho para adicionar um novo elemento, daria muito trabalho para corrigir as constraints não acha?

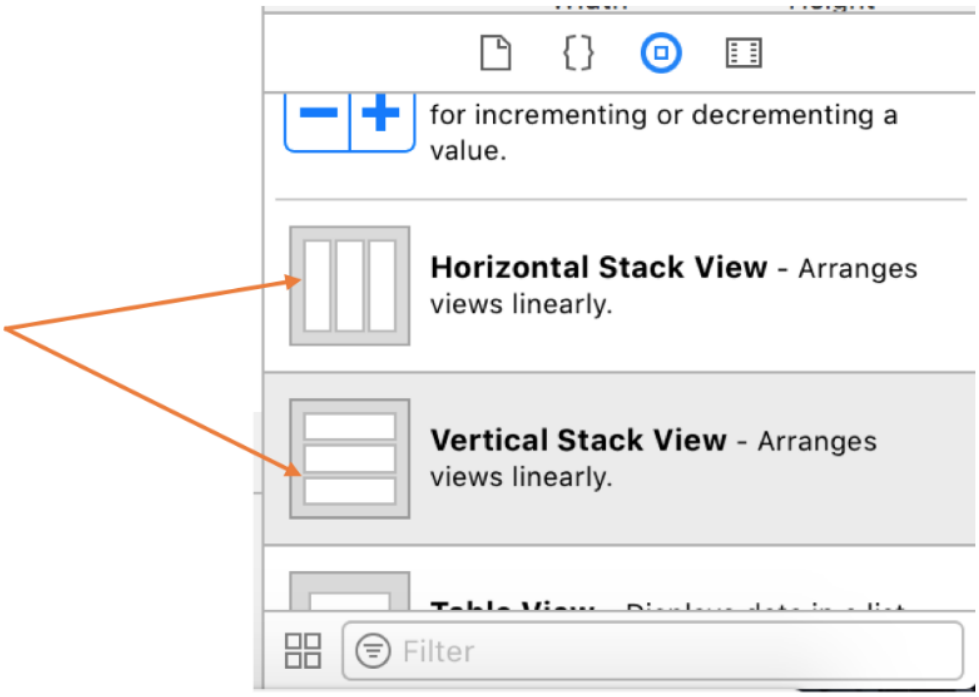
Há casos ainda em que é mais fácil apagar todas as constraints e começar a aplicar novamente de tão complexo que pode ser.

Por isso, há uma forma mais fácil de trabalhar com auto layout no iOS. A partir de agora começaremos a estudar sobre 'stackview'

Então vamos apagar os elementos que utilizamos no ViewController para começarmos uma implementação utilizando stackview.

O 'stackview' empilha os elementos tanto na horizontal quanto na vertical. A grande vantagem é que se precisarmos adicionar ou remover algum objeto de dentro do stackview, basta remove-lo e ele reposiciona os elementos automaticamente baseado na 'distribuição' setada.

Se procurarmos por stackview no object library, há duas opções:

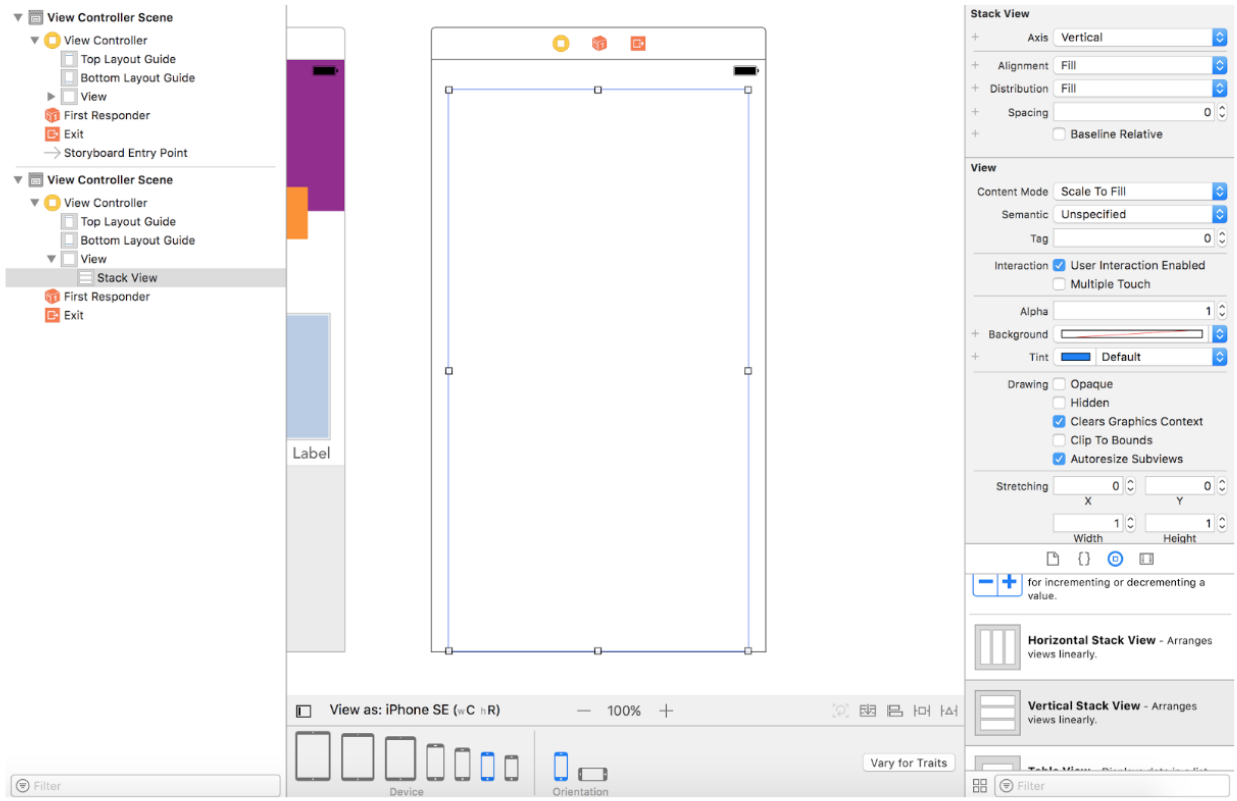


A primeira opção empilha os elementos na horizontal (lado a lado). A segunda, faz o contrário, empilha os elementos na vertical (um embaixo do outro).

Para saber qual deles utilizar, basta analisar a UI (interface) do aplicativo:

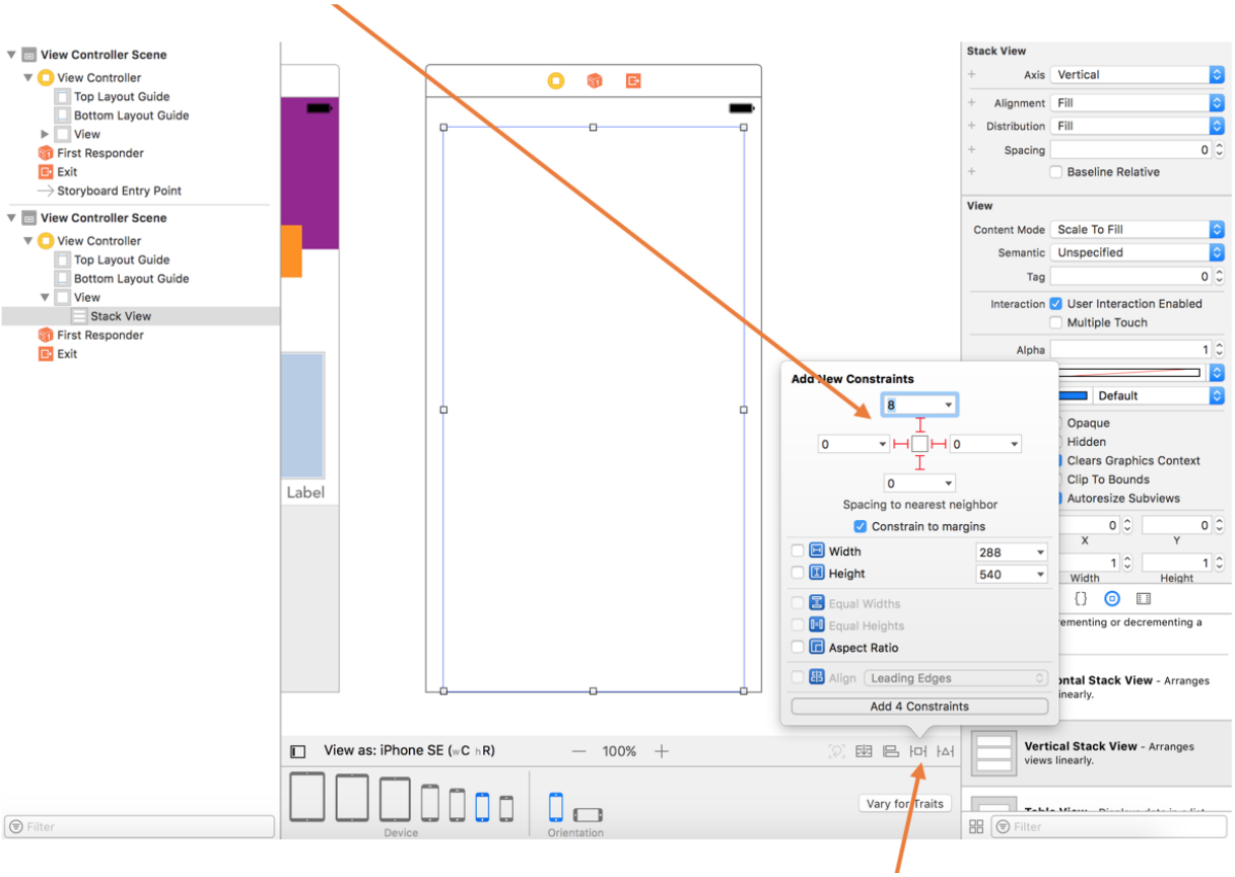
Repare que temos os elementos: labelPacotes, textfield, labelPacotesEncontrados e uma collectionView um embaixo do outro. Então uma boa escolha seria utilizarmos um 'Vertical StackView'.

Vamos então arrastar um 'Vertical StackView' para o ViewController e posicioná-lo da seguinte forma:



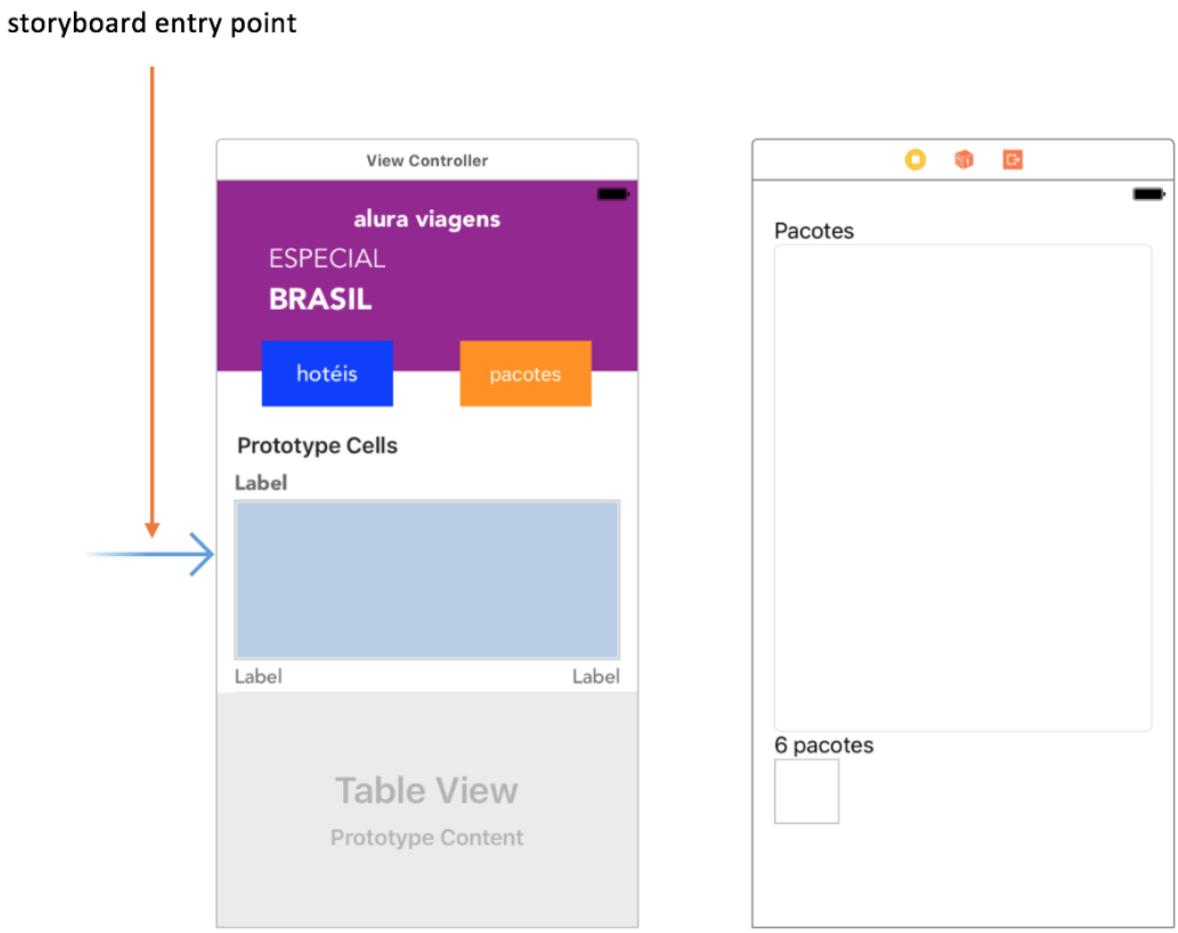
Vamos deixar uma margem à esquerda, à direita e no topo, para que as informações não atrapalhem por exemplo a exibição do relógio na parte de cima do app, e para que as labels não fiquem grudadas nas margens.

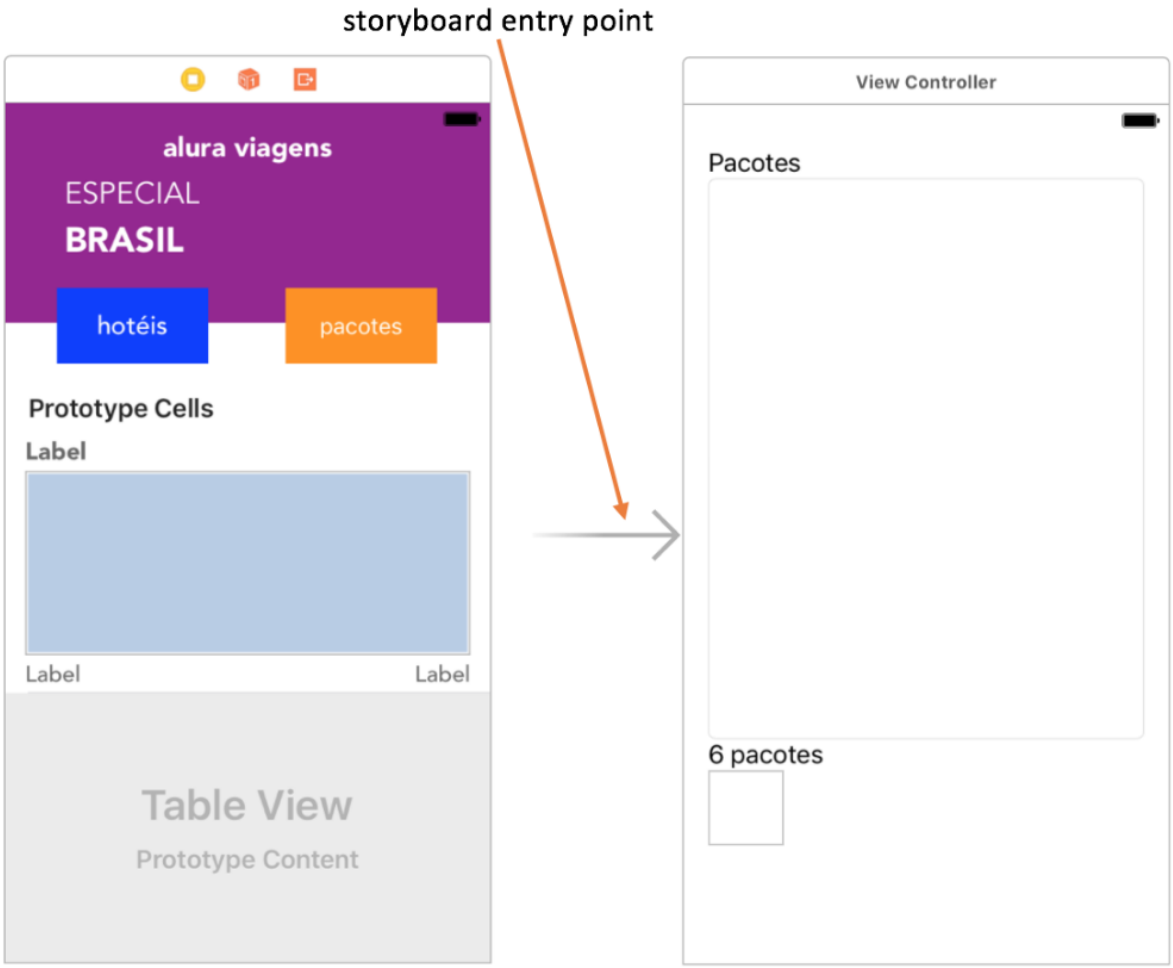
Vamos prender essas margens através de constraints:



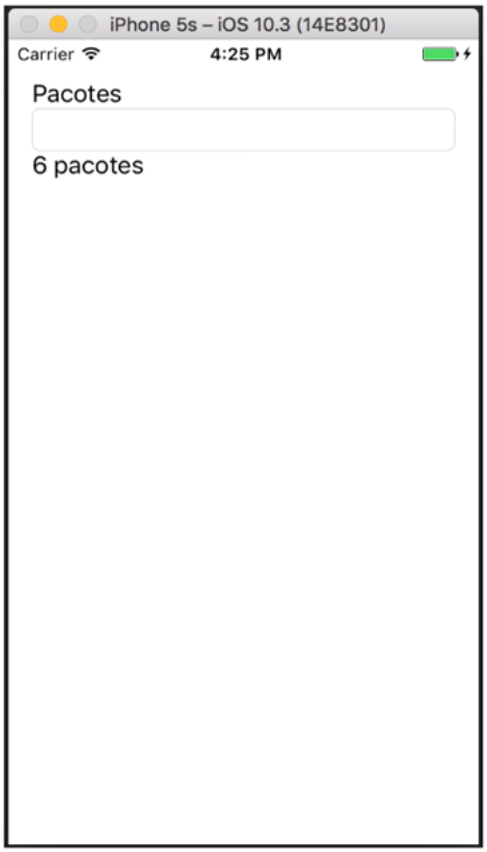
Agora é sua vez: com o stackview criado, vamos arrastar novamente os elementos para dentro do stackview

Para testar vamos setar o 'Storyboard Entry Point' na segunda tela:

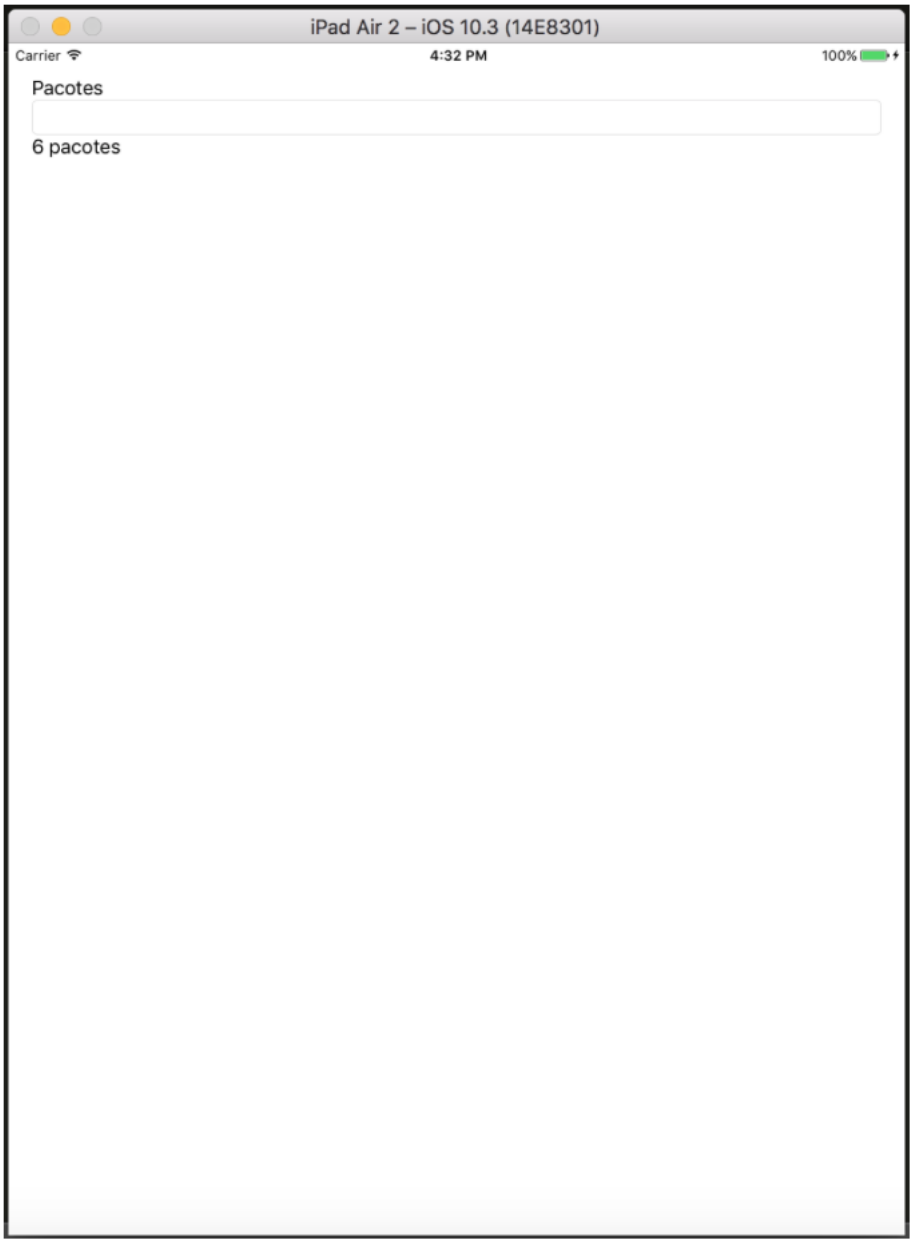




Vamos começar testando no simulador do iPhone 5:



Vamos agora rodar o app no simulador do iPad:



Repare que como implementamos essa tela utilizando o UIStackView, ao rodar o app no simulador do iPad não precisamos setar constraints de margem em nenhum elemento, pois quando o stackview se redimenciona os elementos são reposicionados automaticamente.

A seguir continuaremos a implementação, melhorando o layout do nosso app.