

# App Nome e Idade

X-Code com Swift

Prof. Agesandro Scarpioni  
[agesandro@fiap.com.br](mailto:agesandro@fiap.com.br)

# Nome e Idade

- ⦿ Vamos criar o aplicativo de uma forma diferente:
  - ⦿ Nesta aula vamos criar um projeto utilizando templates para otimizar o processo de desenvolvimento do App, tudo de forma automática.
  - ⦿ Em um segundo momento iremos incluir um IBAction e um IBOutlet de forma manual.
  - ⦿ A intenção é demonstrar as funcionalidades da ferramenta vista de dois ângulos diferentes.

# Iniciando Forma 1

The image shows the Xcode welcome screen on the left and the project template selection dialog on the right.

**Welcome to Xcode**  
Version 9.2 (9C40b)

**1** Get started with a playground  
Explore new ideas quickly and easily.

**1** Create a new Xcode project  
Create an app for iPhone, iPad, Mac, Apple Watch, tvOS, or macOS.

**1** Clone an existing project  
Start working on something from an SCM repository.

**2** Choose a template for your new project:

**iOS** watchOS tvOS macOS Cross-platform **Filter**

**Application**

- 3** Single View App
- Game
- Augmented Reality App
- Document Based App
- Master-Detail App

**Framework & Library**

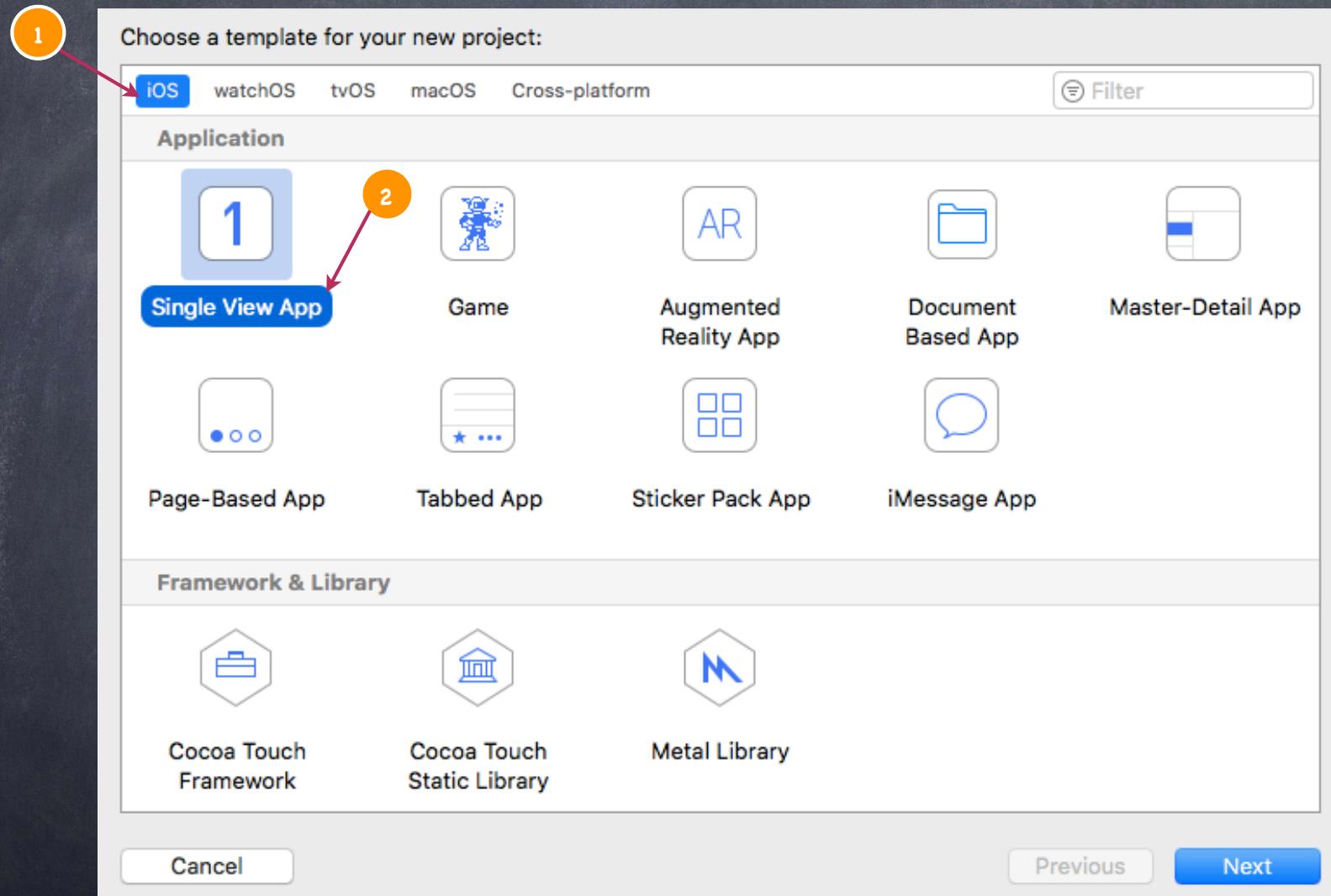
- Cocoa Touch Framework
- Cocoa Touch Static Library
- Metal Library

Cancel Previous Next

Detailed description: The screenshot captures the initial setup of Xcode. On the left, the 'Welcome to Xcode' screen is visible, showing version 9.2 and three main options: 'Get started with a playground', 'Create a new Xcode project', and 'Clone an existing project'. A large orange circle labeled '1' points to the 'Create a new Xcode project' option. On the right, the 'Choose a template for your new project' dialog is open. It features tabs for iOS, watchOS, tvOS, macOS, and Cross-platform, with 'iOS' selected. Under the 'Application' section, the 'Single View App' template is highlighted with a blue border and a large orange circle labeled '3' pointing to it. Other application templates shown include Game, Augmented Reality App, Document Based App, and Master-Detail App. Below the application section, there's a 'Framework & Library' section with icons for Cocoa Touch Framework, Cocoa Touch Static Library, and Metal Library. At the bottom of the dialog are 'Cancel', 'Previous', and 'Next' buttons.

# Iniciando Forma 2

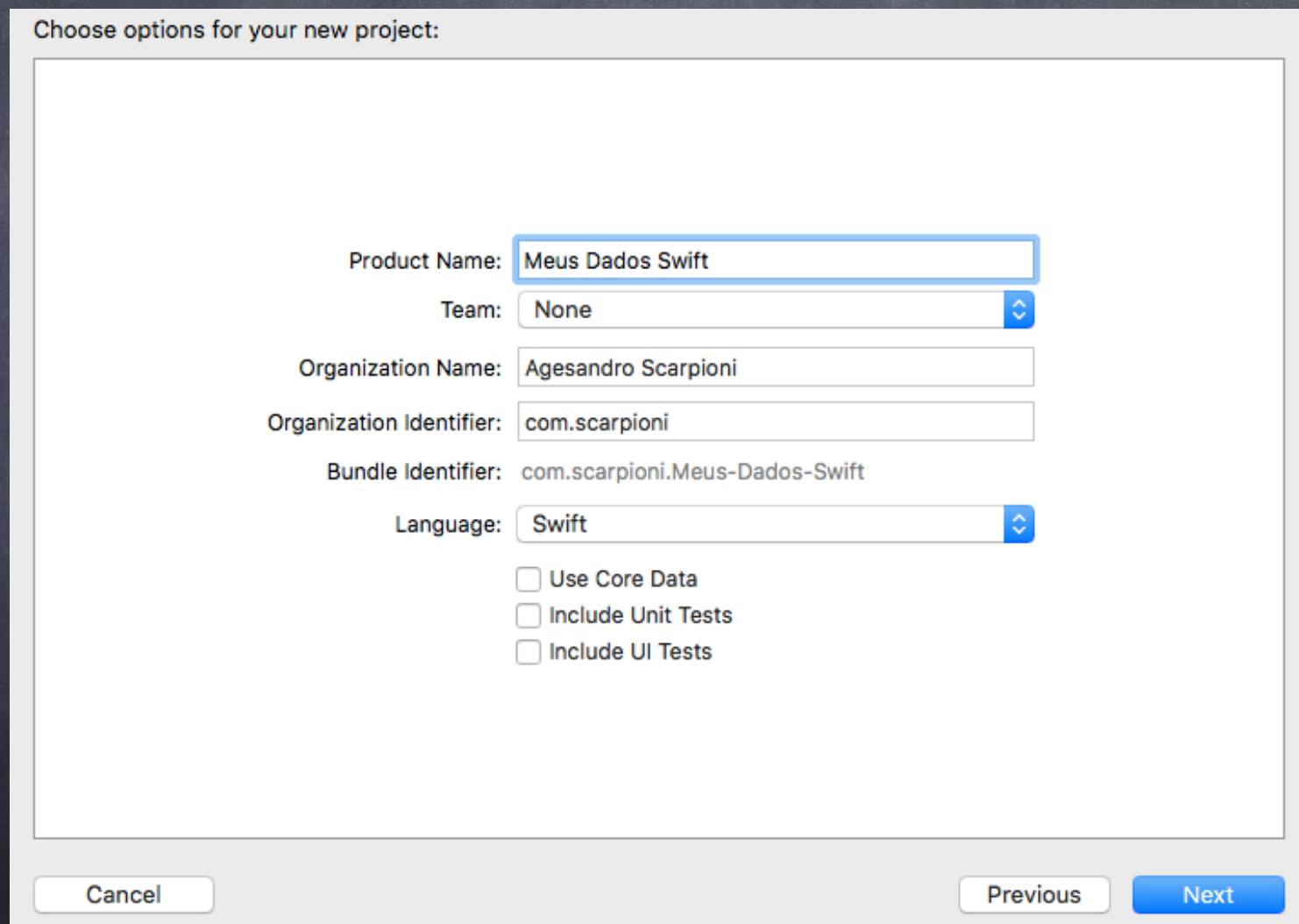
Ou Clique em File -> New Project -> Application -> Single View Application.



OBS: Single View Application é o template que já cria uma viewController e uma classe com o arquivo .swift, inclusive o delegate já tem a referencia dessa classe.

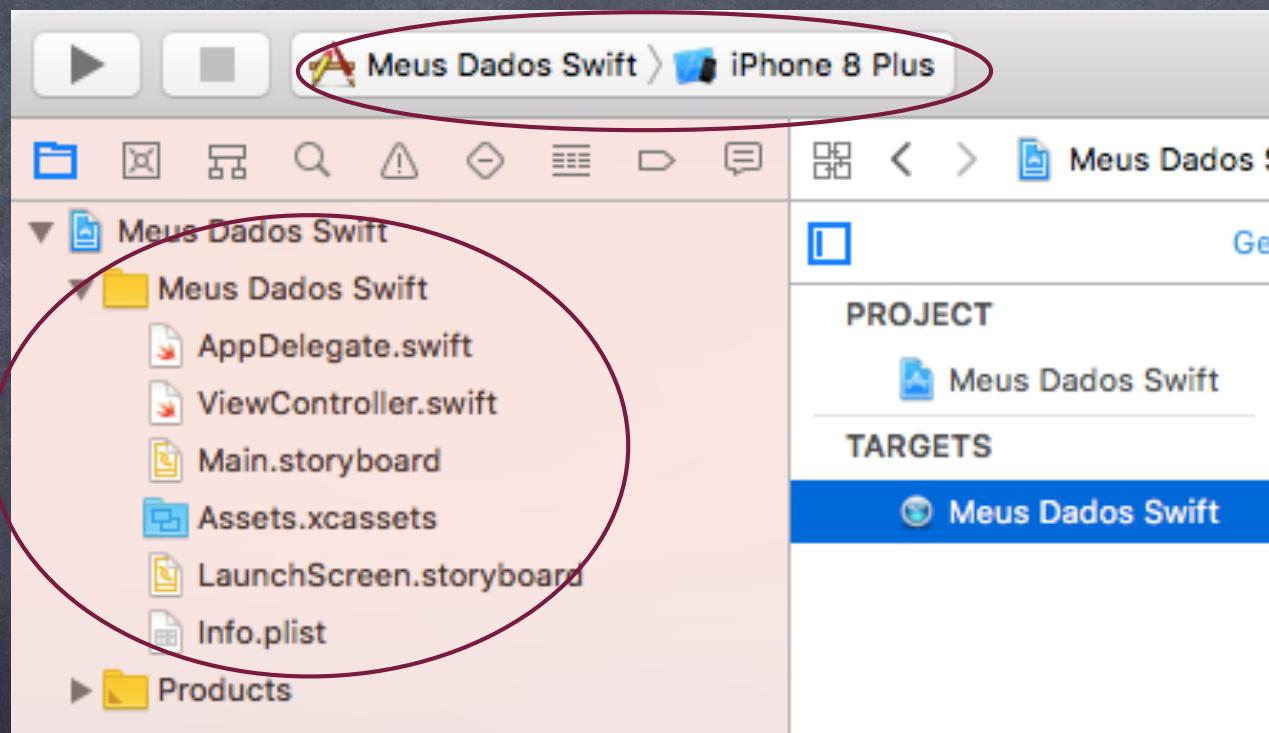
# O App

- Preencha com os dados abaixo, lembre-se que o Organization Identifier é como se fosse o pacote no Java ou o namespace do VB, em language use Swift.



# Tudo em seu lugar

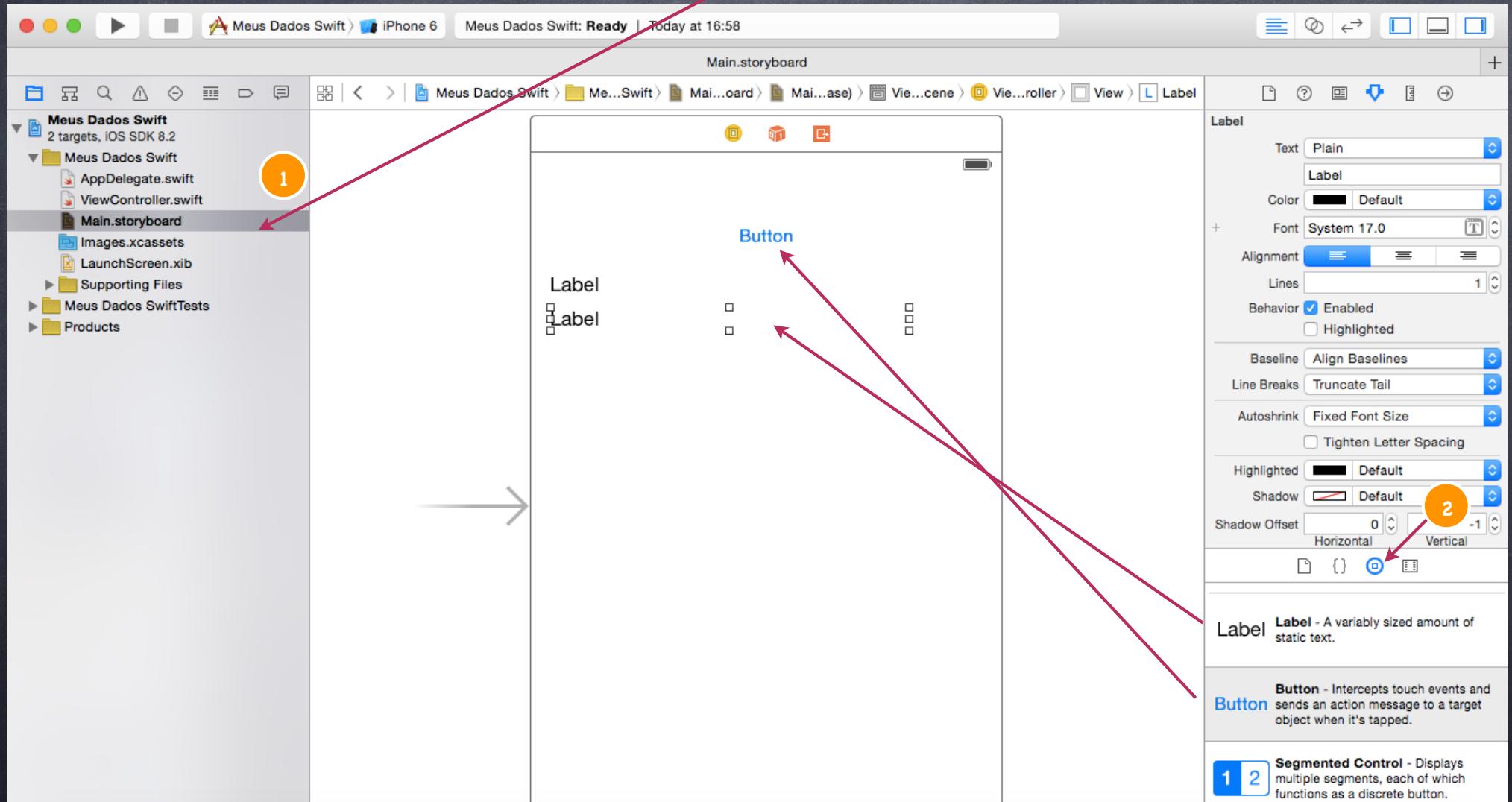
- Note que utilizando este template foi incluída uma classe chamada ViewController.swift, o delegate que já possui a classe referenciada e o Storyboard. Você pode escolher o device que irá rodar no simulador, no exemplo o app irá rodar para iPhone 8 Plus.



**Obs:** Se você executar o programa agora, irá aparecer a tela de iPhone no simulador sem termos que programar ao menos uma linha.

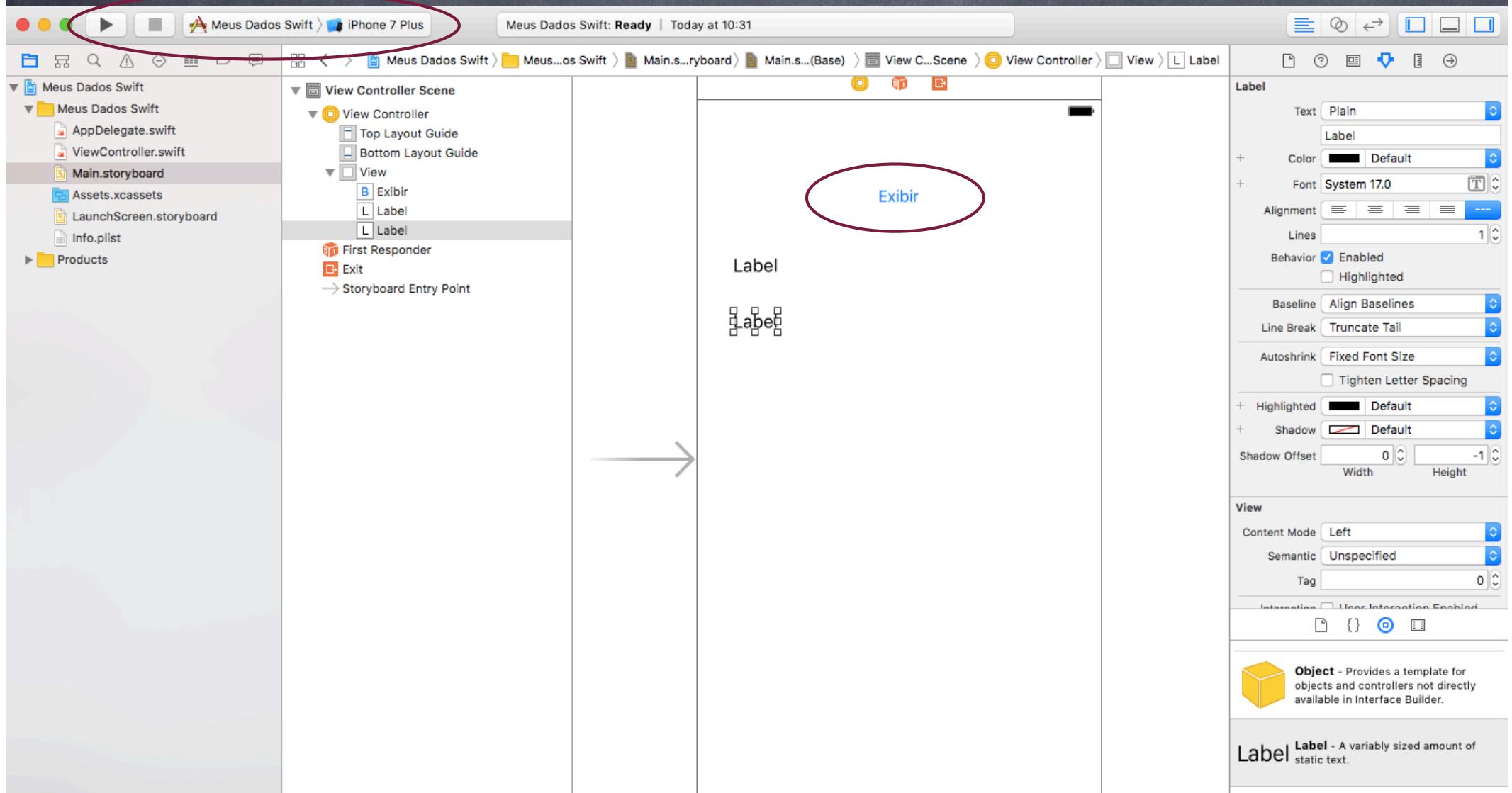
# Adicionando objetos

- Inclua dois labels e um botão no Main.Storyboard, altere o texto do botão para Exibir.



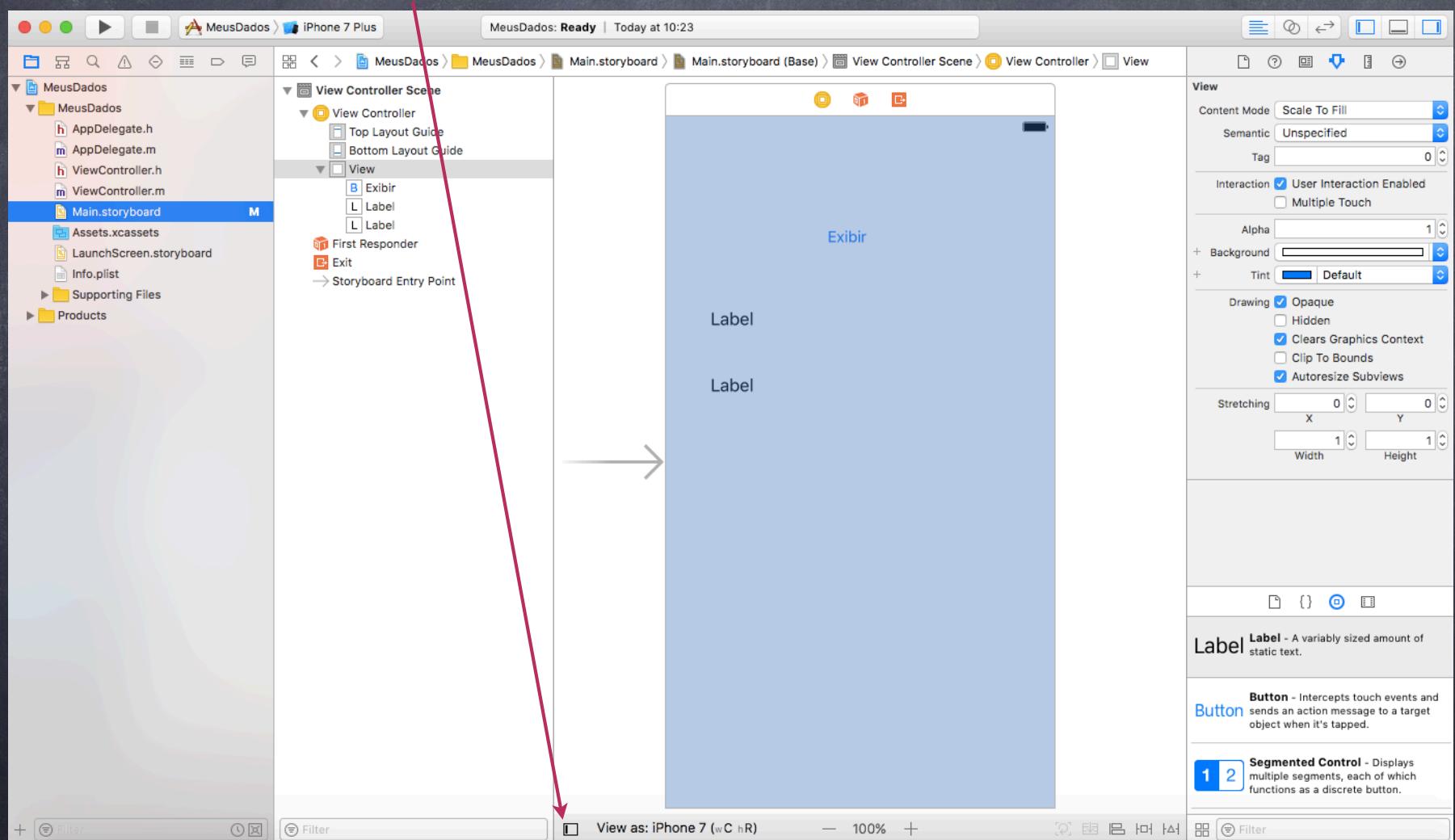
# Executando

- Clique duas vezes sobre o botão e digite "Exibir" no texto, escolha um modelo de simulador ou deixe o modelo sugerido pelo Xcode e execute seu App clicando em Run ou utilizando o atalho command + R.



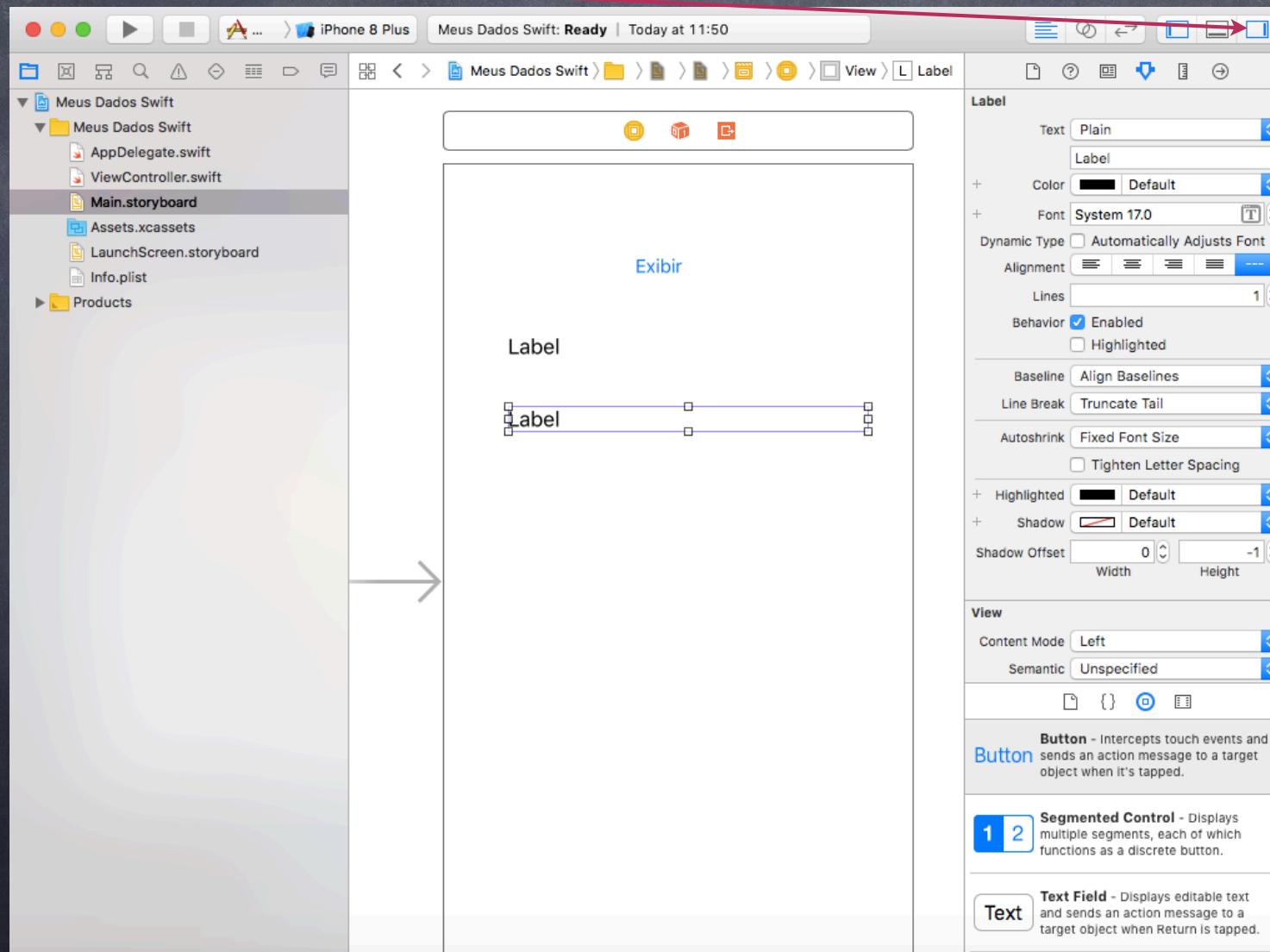
# Continuando

- Primeiramente vamos organizar nosso ambiente escondendo a janela da direita clicando neste ícone.



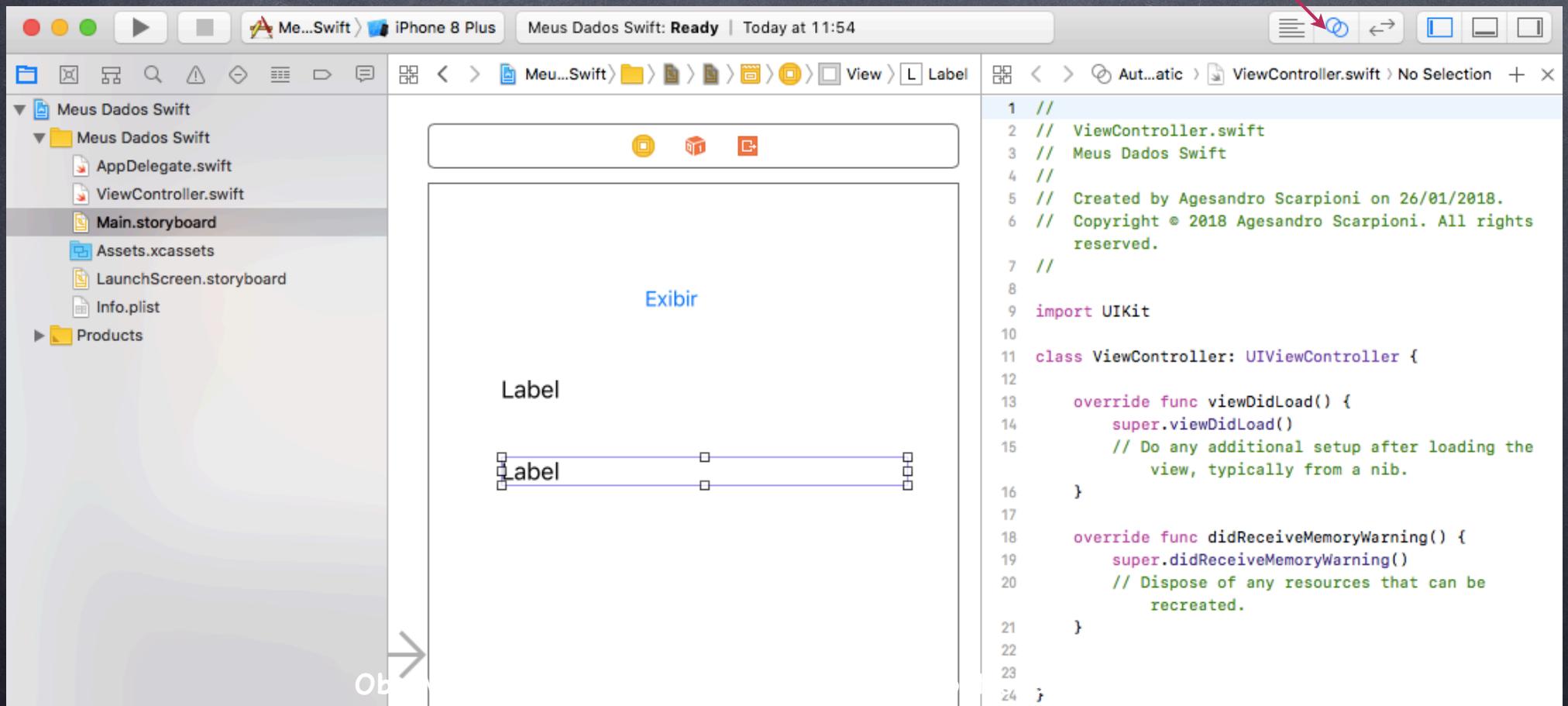
# Declarando os outlet's no .swift

- Depois vamos organizar nosso ambiente escondendo a janela da direita clicando neste ícone.



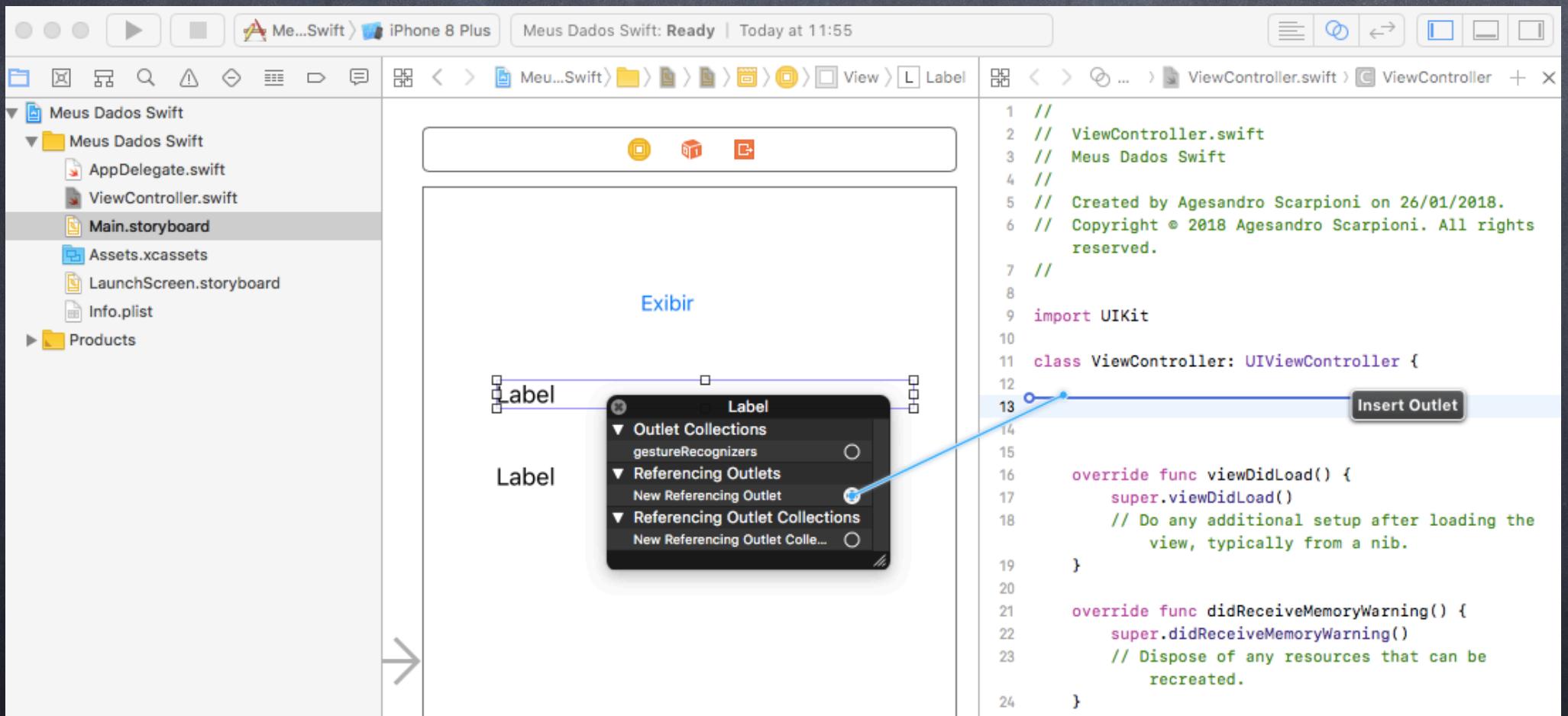
# Declarando os outlet's no .swift

- Depois vamos compartilhar a tela de Storyboard com a viewController obtendo as duas simultaneamente clicando neste ícone.



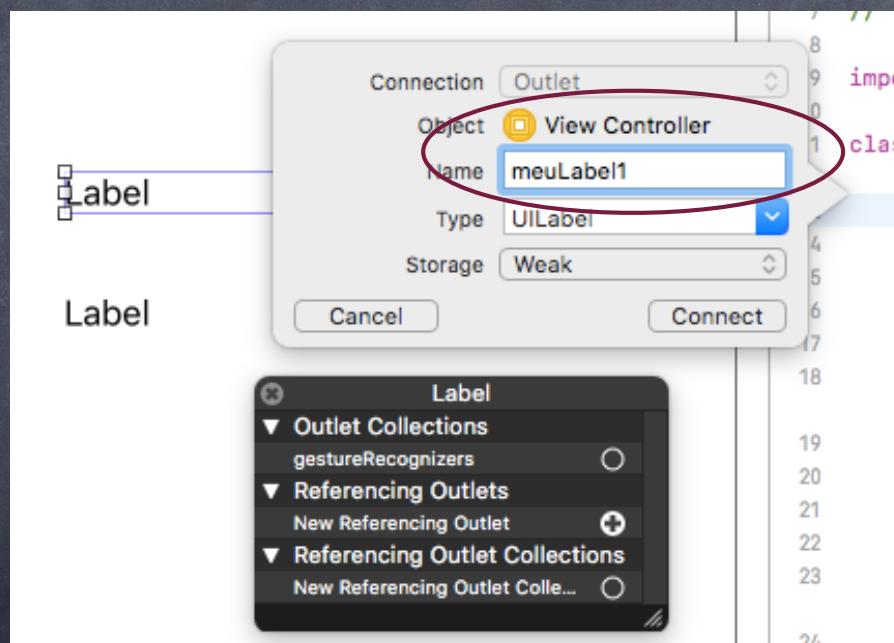
# Declarando os outlet's automaticamente

- Clique com o botão direito sobre o 1º label e escolha "New Referencing Outlet's" clicando no local indicado na figura.



# Declarando os outlet's automaticamente

- Quando você soltar o botão do mouse aparecerá a janela abaixo, nomeie o outlet como "meuLabel1" e clique em connect. Repita os mesmos passos para o Label2.



# Declarando os outlet's automaticamente

- Ao final essas duas linhas ( IBOutlets ) são declaradas automaticamente no arquivo e os dois label's da view já estão relacionados aos dois outlet's.

```
8
9 import UIKit
10
11 class ViewController: UIViewController {
12
13
14    @IBOutlet weak var meuLabel1: UILabel!
15
16    @IBOutlet weak var meuLabel2: UILabel!
17
18
19    override func viewDidLoad() {
20        super.viewDidLoad()
21        // Do any additional setup after loading the view, typically
22        // from a nib.
23    }
24
25    override func didReceiveMemoryWarning() {
26        super.didReceiveMemoryWarning()
27        // Dispose of any resources that can be recreated.
28
29
30
31
32 }
```

# Implementando

- Na Classe ViewController.swift, existe um método chamado viewDidLoad que é o primeiro método que é executado quando a classe é chamada. Vamos carregar uma frase em cada label.

The screenshot shows the Xcode interface. On the left, the Project Navigator lists files: AppDelegate.swift, ViewController.swift, Main.storyboard (circled in red), Assets.xcassets, LaunchScreen.storyboard, Info.plist, and Products. In the center, the Storyboard Editor shows a single view controller with a label labeled "Label". A blue arrow points from the storyboard to the code editor on the right. The code editor displays ViewController.swift:

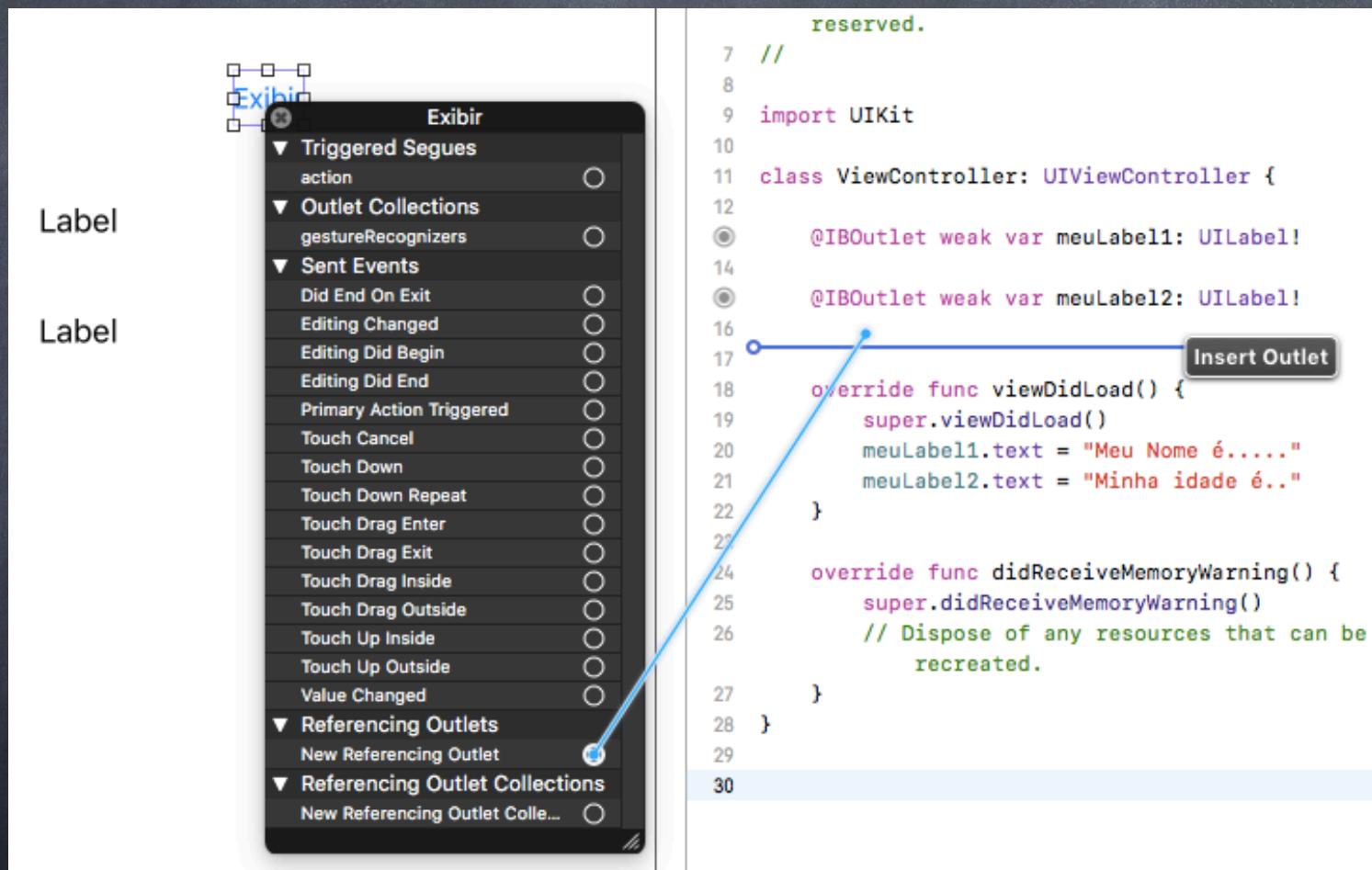
```
4 //  
5 // Created by Agesandro Scarpioni on 26/01/2018.  
6 // Copyright © 2018 Agesandro Scarpioni. All rights  
reserved.  
7 //  
8  
9 import UIKit  
10  
11 class ViewController: UIViewController {  
12  
13     @IBOutlet weak var meuLabel1: UILabel!  
14  
15     @IBOutlet weak var meuLabel2: UILabel!  
16  
17  
18     override func viewDidLoad() {  
19         super.viewDidLoad()  
20         meuLabel1.text = "Meu Nome é....."  
21         meuLabel2.text = "Minha idade é.."  
22     }  
23  
24     override func didReceiveMemoryWarning() {  
25         super.didReceiveMemoryWarning()  
26         // Dispose of any resources that can be  
27         // recreated.  
28     }  
}
```

A red oval highlights the viewDidLoad() method and its content.

Obs: viewDidLoad equivale ao Form\_Load do VB e ao windowOpened do JFrame no JAVA.

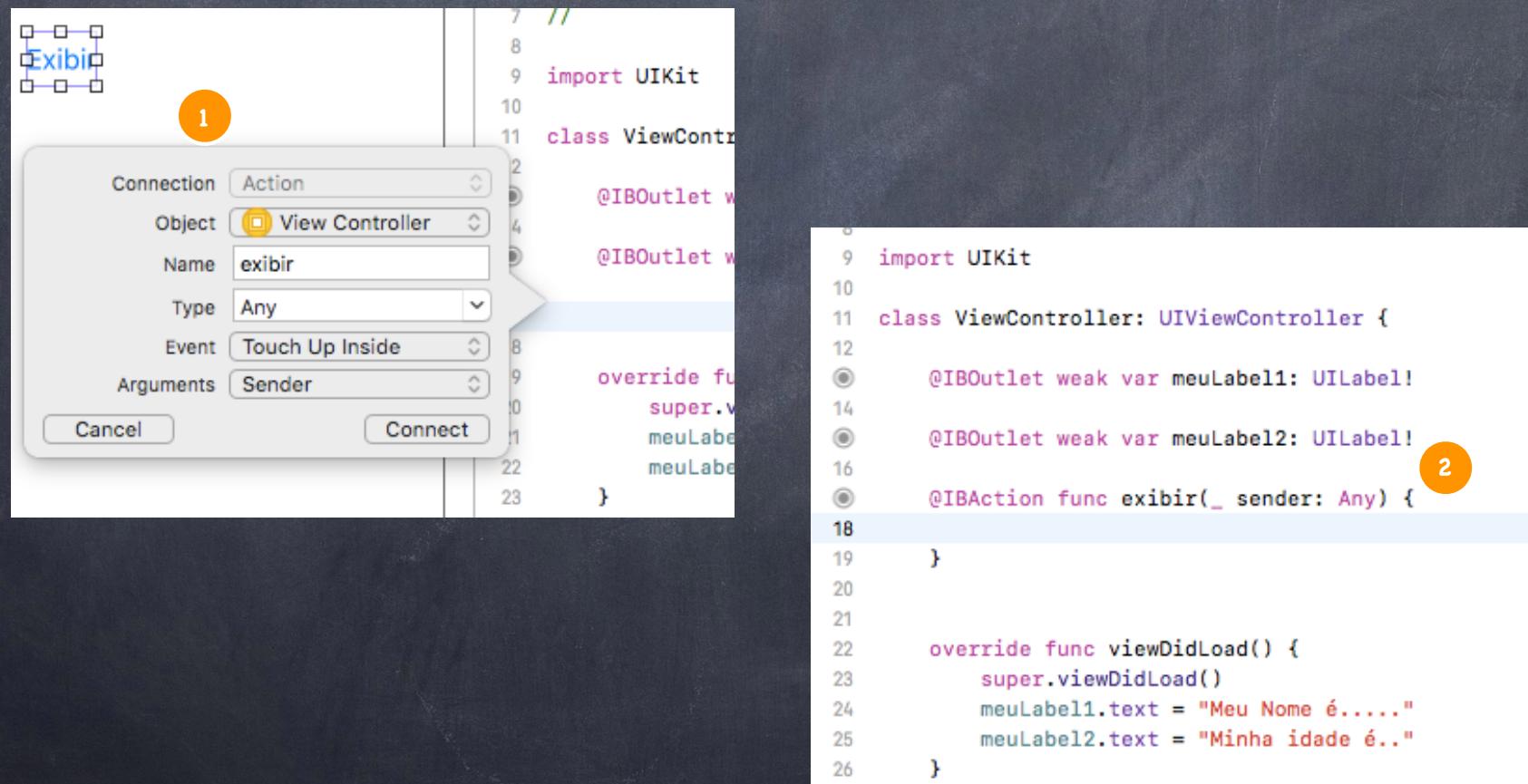
# Declarando IBAction automaticamente

- Clique no storyboard, clique com o botão direito do mouse sobre o Exibir, escolha o evento “Touch Up Inside” e arraste para o ViewController.swift.



# Declarando IBAction automaticamente

- Preencha o IBAction com o nome “exibir” (1) e clique em Connect, a declaração irá aparecer automaticamente como mostra a figura 2 e já estará pronta para implementação.



# Implementando IBAction

- Escreva as 2 linhas de comando.

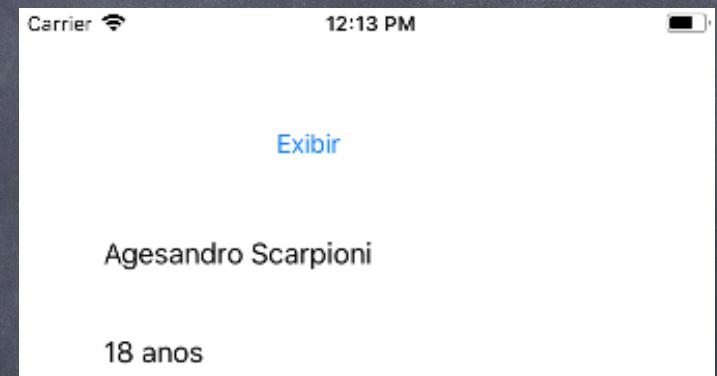
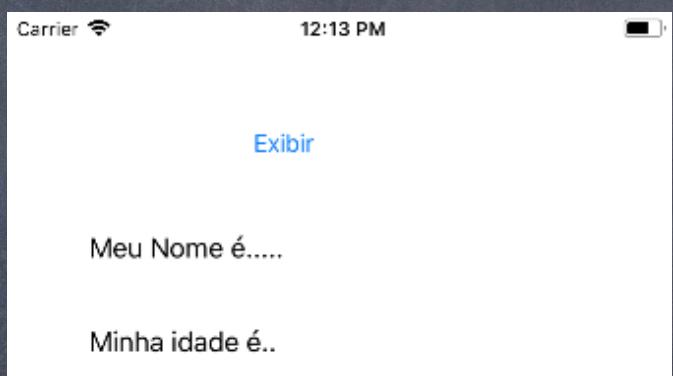
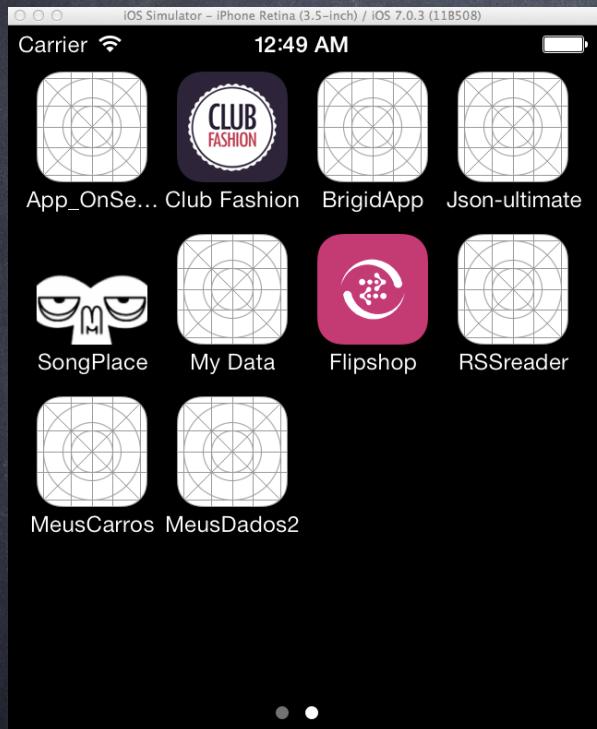
```
9 import UIKit
10
11 class ViewController: UIViewController {
12
13     @IBOutlet weak var meuLabel1: UILabel!
14
15     @IBOutlet weak var meuLabel2: UILabel!
16
17     @IBAction func exibir(_ sender: Any) {
18         meuLabel1.text = "Agesandro Scarpioni"
19         meuLabel2.text = "18 anos"
20     }
21
22
23     override func viewDidLoad() {
24         super.viewDidLoad()
25         meuLabel1.text = "Meu Nome é....."
26         meuLabel2.text = "Minha idade é.."
27     }
28
29     override func didReceiveMemoryWarning() {
30         super.didReceiveMemoryWarning()
31         // Dispose of any resources that can be
32         // recreated.
33     }
34 }
```

# O App no simulador

Quando executamos.

Antes de clicar no botão com informações carregadas pelo viewDidLoad.

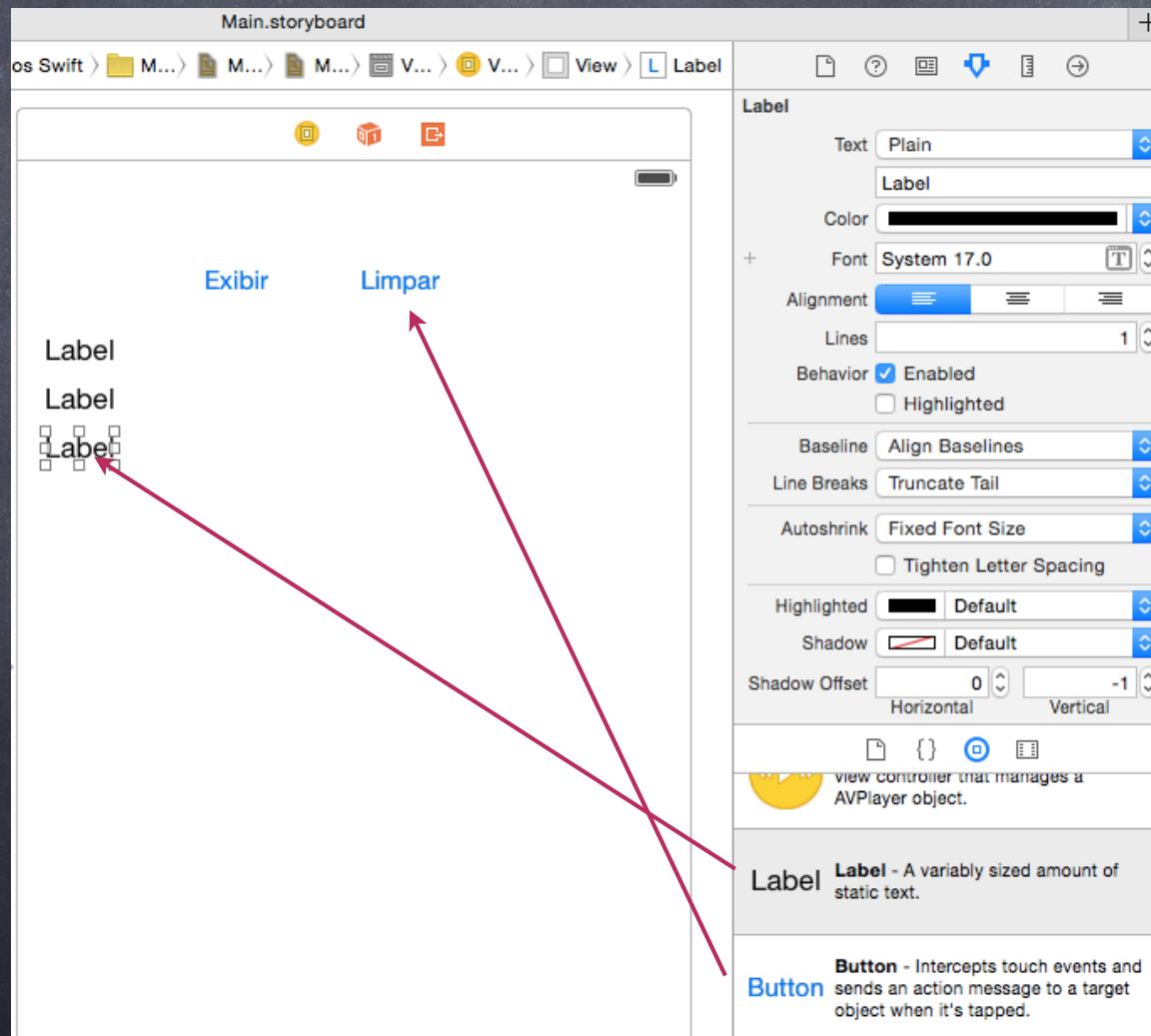
Após clicar no botão.



Dica: O Command + R é o atalho para executarmos nosso App.

# Implementando IBAction e IBOulet de forma manual

- Inclua um botão Limpar e mais um label em seu Storyboard.



# Implementando IBAction e IBOulet de forma manual

- Digite a linha do Outlet e a linha do IBAction abaixo:

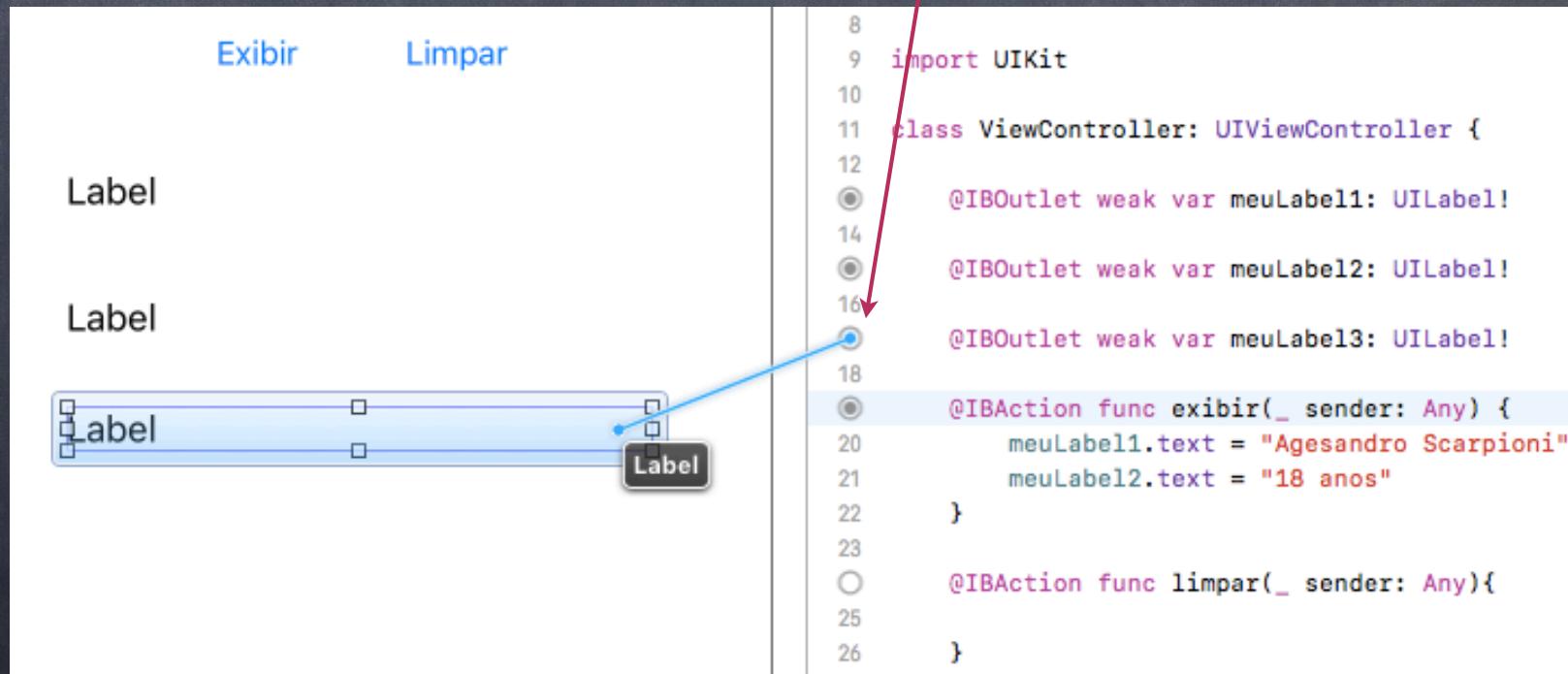
- Note que os mesmos não estão conectados com a View, observe que os círculos brancos em frente aos números das linhas não estão preenchidos.

```
9 import UIKit
10
11 class ViewController: UIViewController {
12
13     @IBOutlet weak var meuLabel1: UILabel!
14
15     @IBOutlet weak var meuLabel2: UILabel!
16
17     @IBOutlet weak var meuLabel3: UILabel!
18
19     @IBAction func exibir(_ sender: Any) {
20         meuLabel1.text = "Agesandro Scarpioni"
21         meuLabel2.text = "18 anos"
22     }
23
24     @IBAction func limpar(_ sender: Any){
25
26     }
27
28     override func viewDidLoad() {
29         super.viewDidLoad()
30         meuLabel1.text = "Meu Nome é....."
31         meuLabel2.text = "Minha idade é.."
32     }
33 }
```

Dica: Para exibir os números de linha clique em Xcode -> Preferences, ou (Command + , ) e em Text Editing, clique no primeiro check box "Show Line Numbers".

# Implementando IBAction e IBOutlet de forma manual

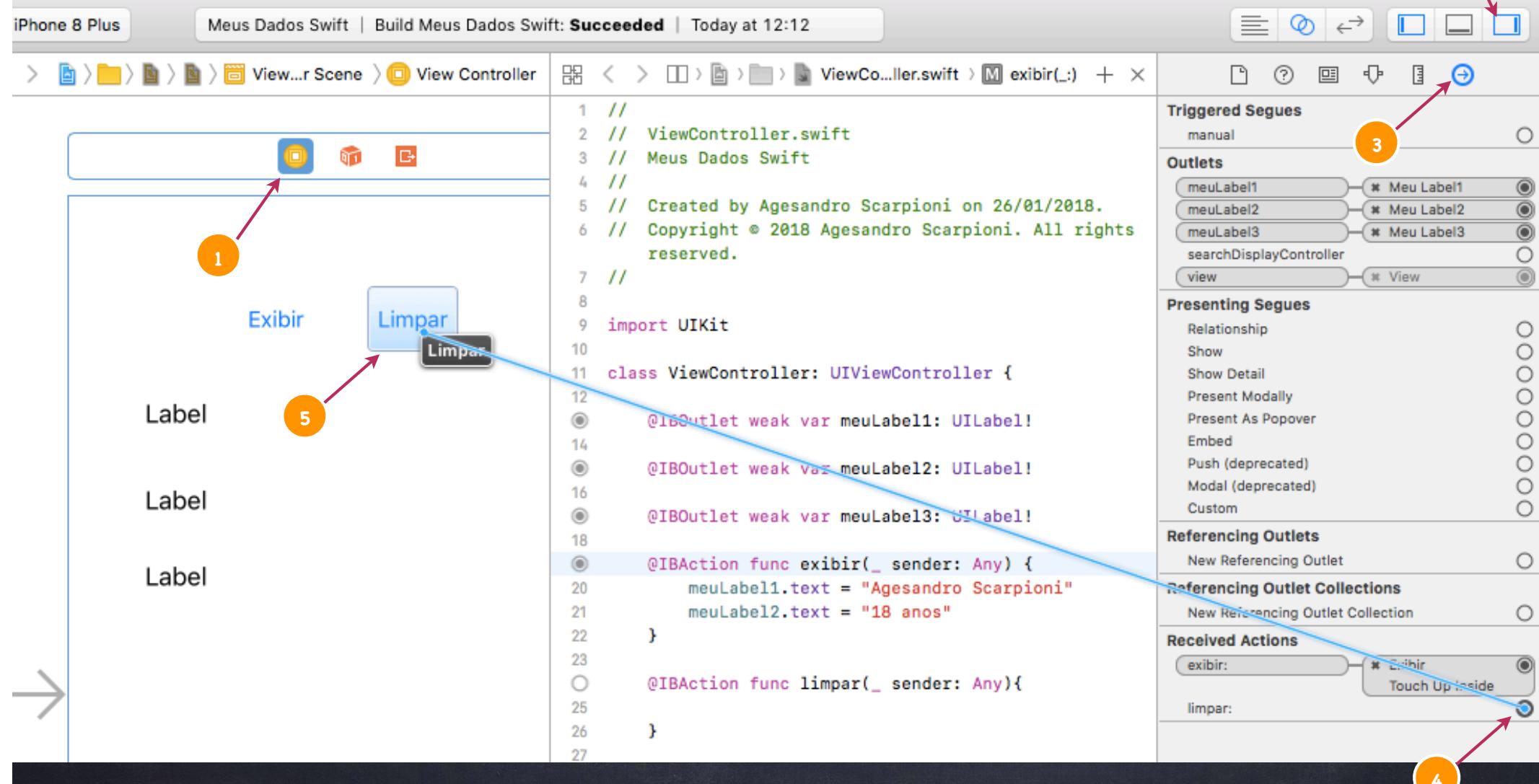
- Trace uma linha do ponto indicado do Outlet até o Label para criarmos um link entre o Label no controller ao Label da View.



# Implementando IBAction e IBOulet de forma manual

FIAP

- Existe uma outra forma para fazer a ligação dos outlet's e action's, siga os passos:



# Implementando IBAction e IBOulet de forma manual

- Note que após essas ligações nosso novo IBOulet e IBAction estão conectados com a View.

```
9 import UIKit
10
11 class ViewController: UIViewController {
12
13     @IBOutlet weak var meuLabel1: UILabel!
14
15     @IBOutlet weak var meuLabel2: UILabel!
16
17     @IBOutlet weak var meuLabel3: UILabel!
18
19     @IBAction func exibir(_ sender: Any) {
20         meuLabel1.text = "Agesandro Scarpioni"
21         meuLabel2.text = "18 anos"
22     }
23
24     @IBAction func limpar(_ sender: Any){
25
26     }
27
28     override func viewDidLoad() {
29         super.viewDidLoad()
30         meuLabel1.text = "Meu Nome é....."
31         meuLabel2.text = "Minha idade é.."
32     }
33 }
```

# Implementando IBAction e IBOulet de forma manual

- Inclua as linhas do Label3 no DidLoad e no Exibir.

- No limpar escreva as linhas indicadas.

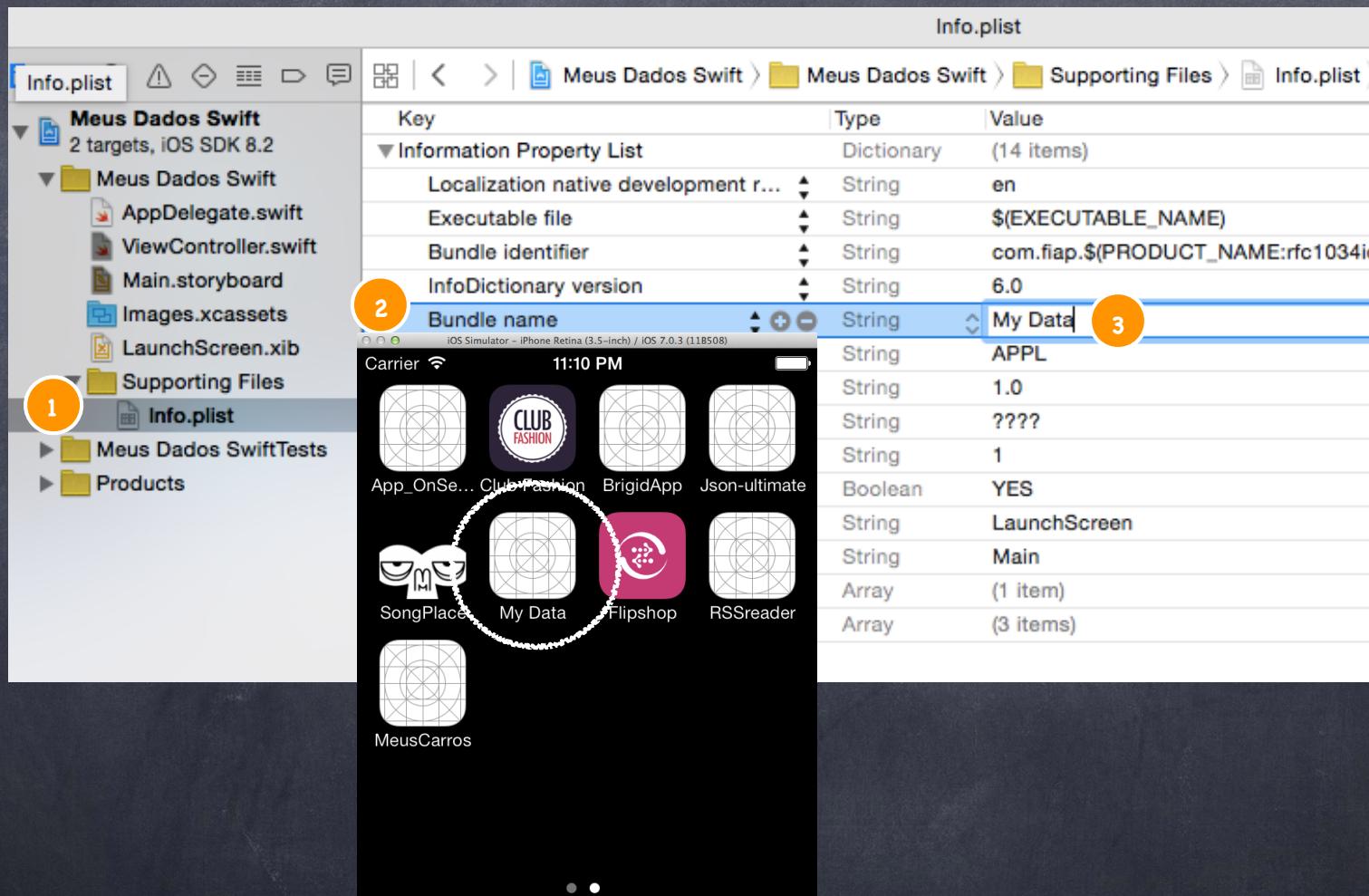
```
9 import UIKit
10
11 class ViewController: UIViewController {
12
13     @IBOutlet weak var meuLabel1: UILabel!
14
15     @IBOutlet weak var meuLabel2: UILabel!
16
17     @IBOutlet weak var meuLabel3: UILabel!
18
19     @IBAction func exibir(_ sender: Any) {
20         meuLabel1.text = "Agesandro Scarpioni"
21         meuLabel2.text = "18 anos"
22         meuLabel3.text = "São Paulo"
23     }
24
25     @IBAction func limpar(_ sender: Any){
26         meuLabel1.text = ""
27         meuLabel2.text = ""
28         meuLabel3.text = ""
29     }
30
31     override func viewDidLoad() {
32         super.viewDidLoad()
33         meuLabel1.text = "Meu Nome é....."
34         meuLabel2.text = "Minha idade é.."
35         meuLabel3.text = "Minha cidade é."
36     }
37
```

- Outra possibilidade no limpar é chamar o método viewDidLoad.

```
25
26     @IBAction func limpar(_ sender: Any){
27         // meuLabel1.text = ""
28         // meuLabel2.text = ""
29         // meuLabel3.text = ""
30         viewDidLoad()
31     }
```

# Como trocar o nome do App

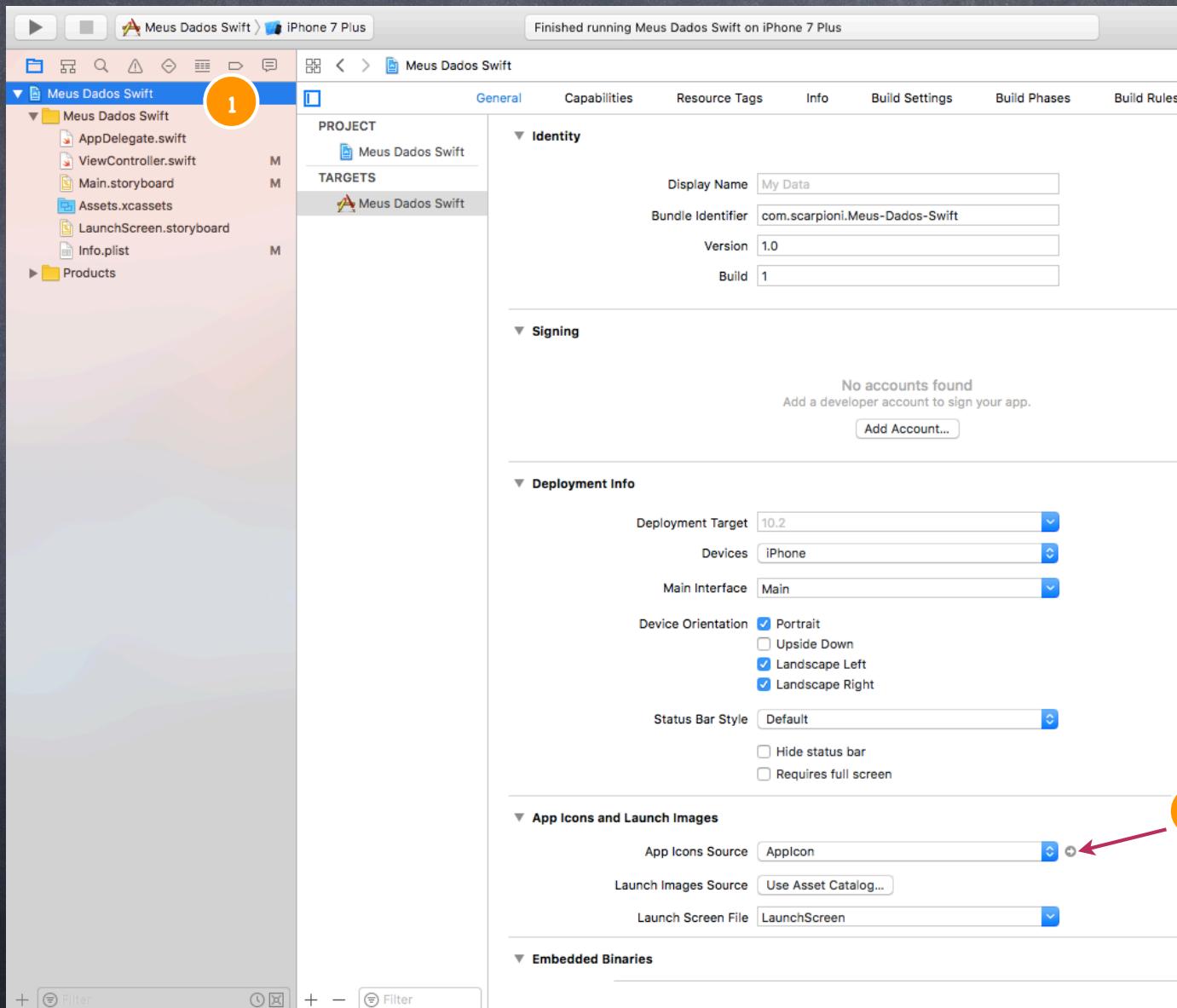
- Encontre o arquivo Info.plist e altere o campo Bundle name (2), digite My Data(3), veja o próximo slide.



Dica: Shift + Command + H → Aciona o botão Home no Simulador

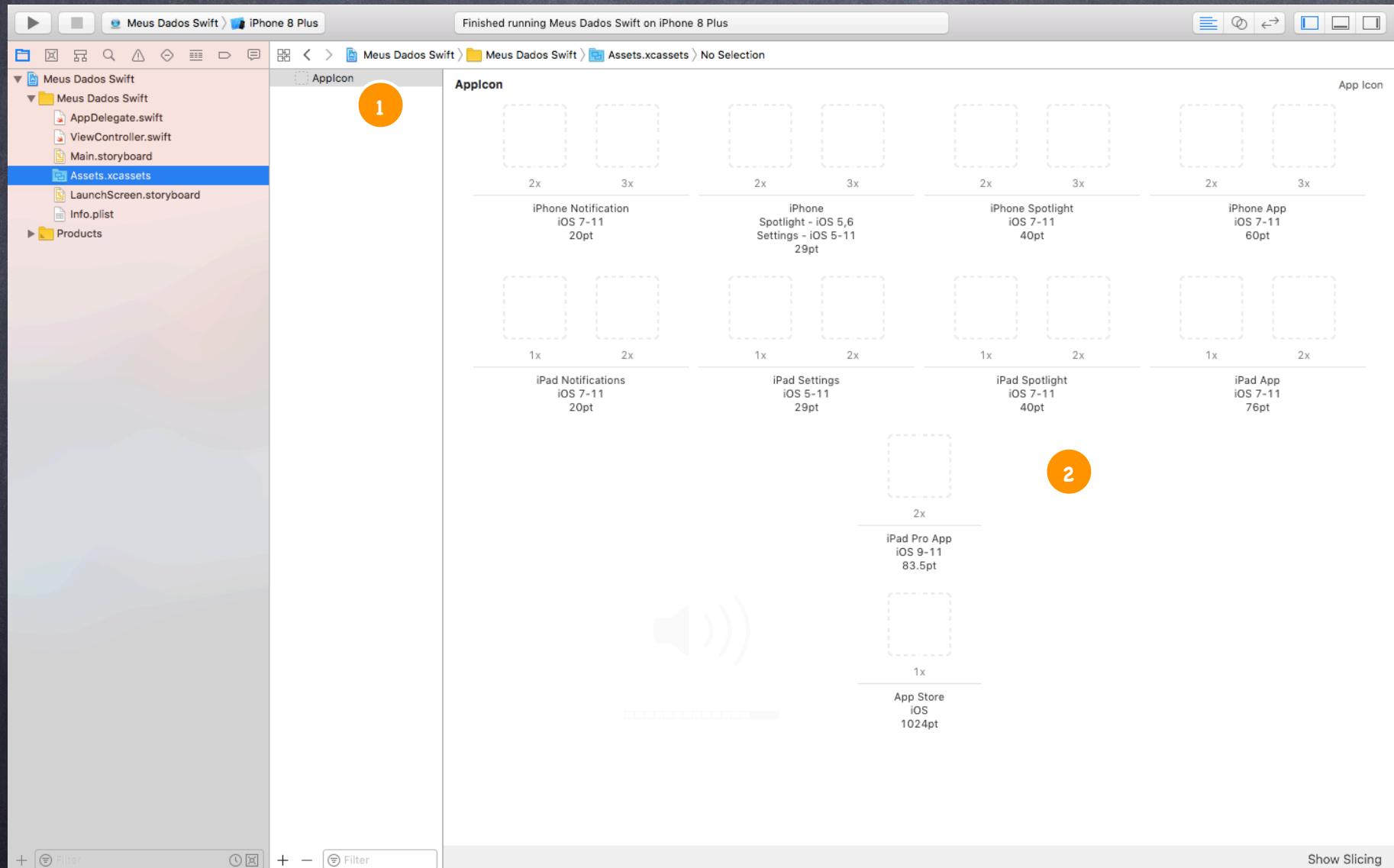
# Alterando o Ícone do App

- ➊ Clique em seu projeto (1), clique na seta do passo 2



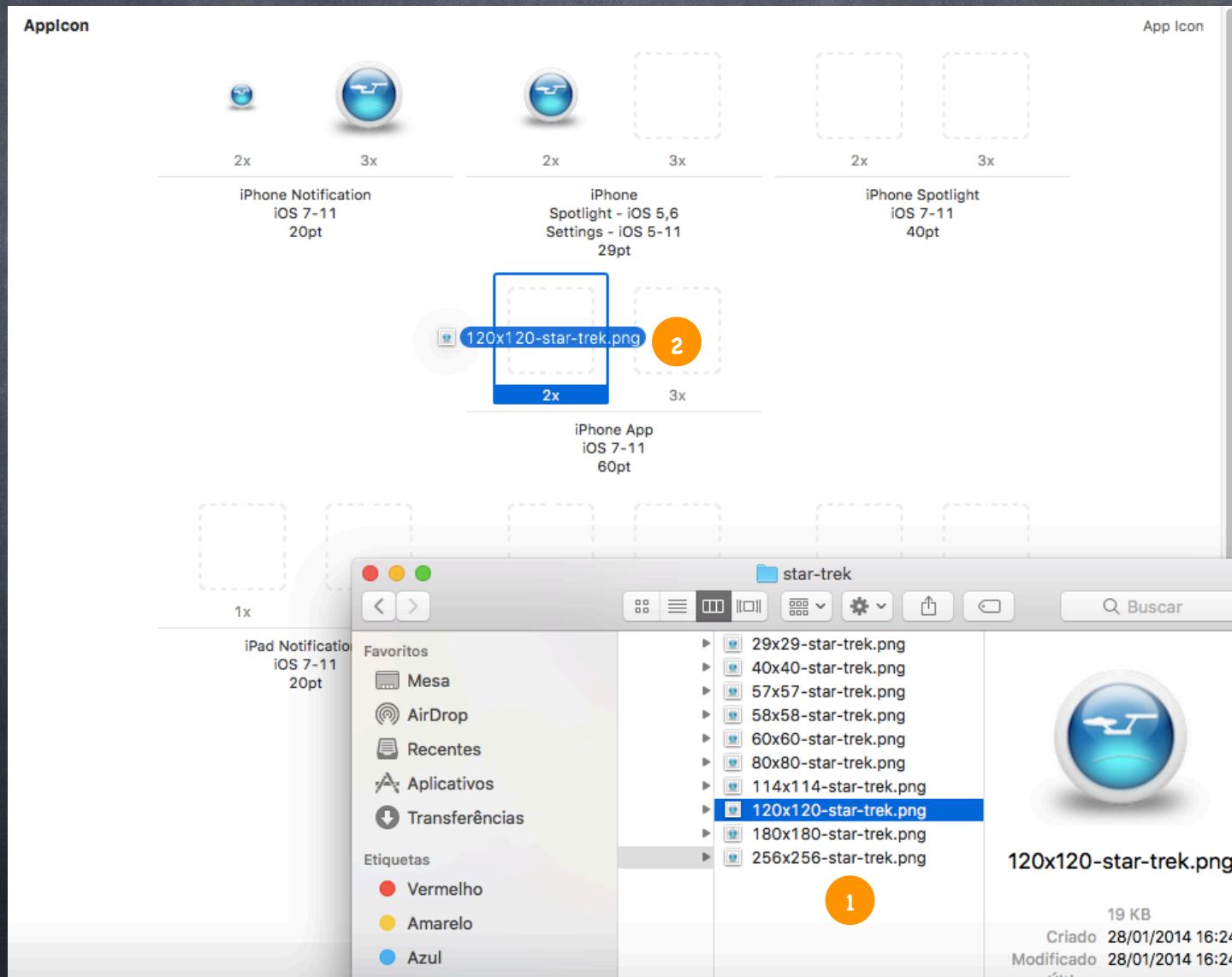
# Alterando o Ícone do App

- A tela abaixo será exibida. O item (1) exibe o catálogo de imagens, o ítem (2) exibe os padrões de imagens e suas respectivas visualizações.



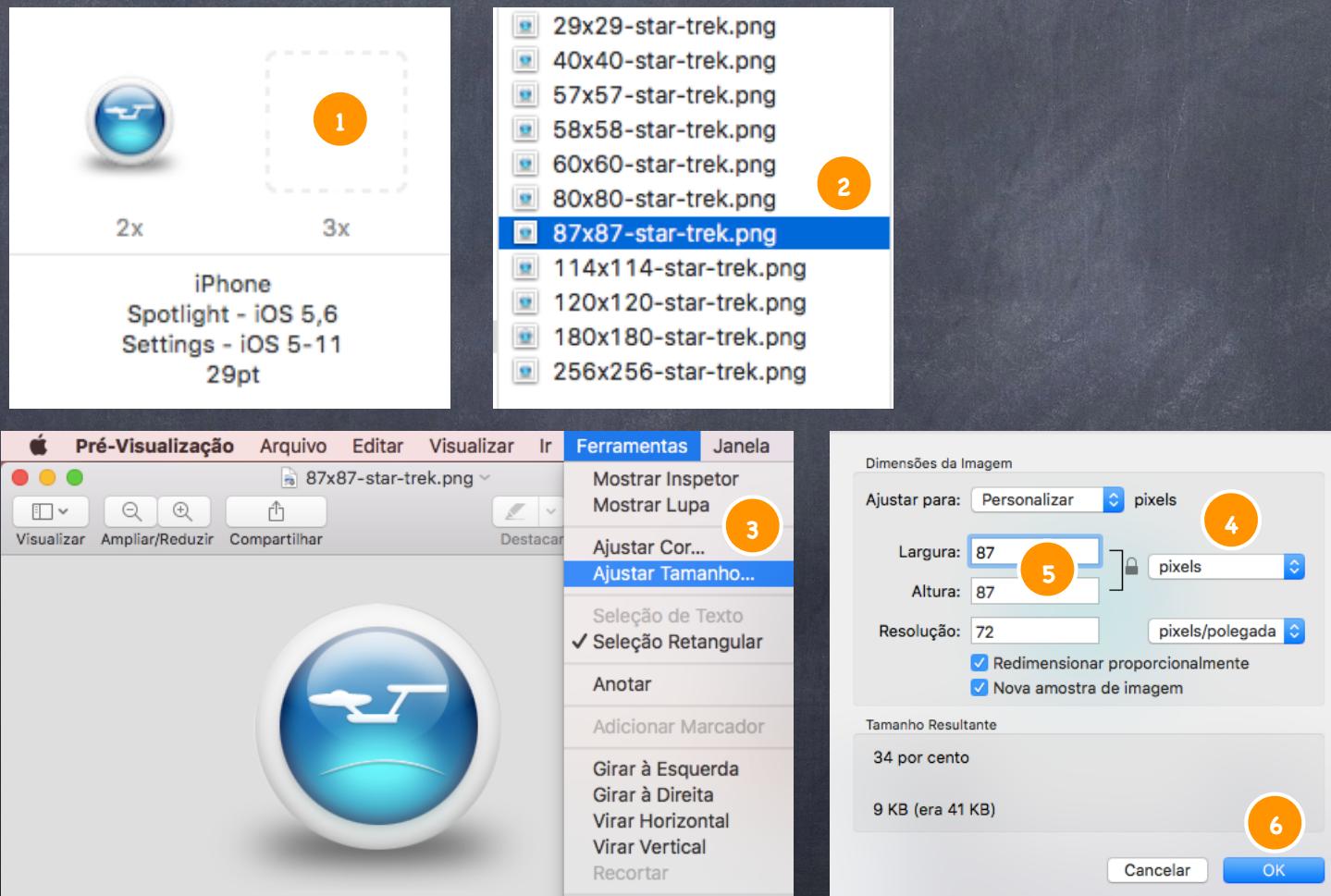
# Alterando o Ícone do App

- Abra uma das pastas disponibilizadas pelo professor com o finder e arraste a imagem apropriada para o espaço exclusivo da imagem escolhida, exemplo: A imagem 120x120 entra no 2x - 60pt, passos (1) e (2).



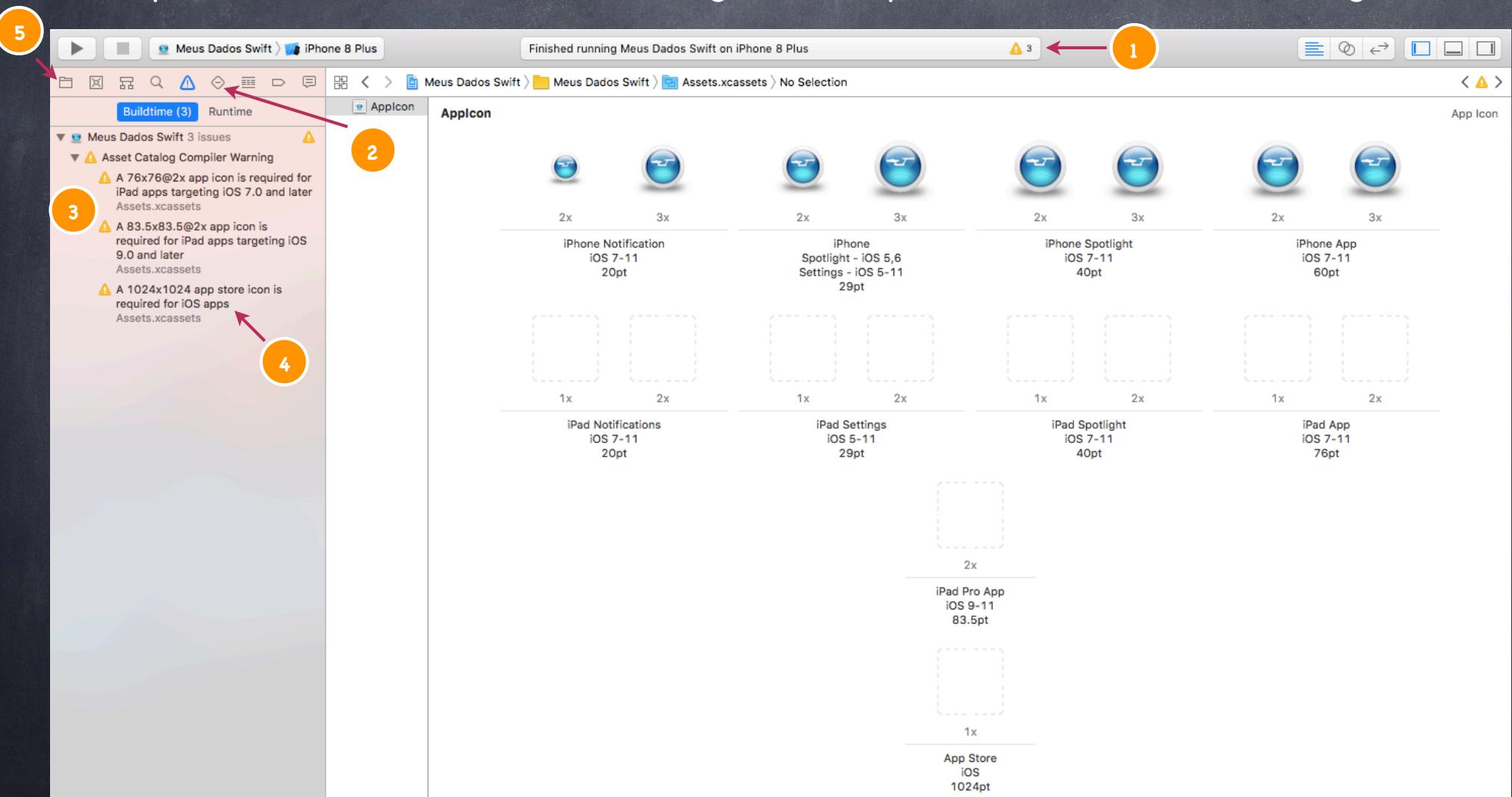
# Ajustando um ícone

- Note que para a imagem 3x 29pt, precisamos de uma imagem 87x87 (1) que no momento não temos ainda, uma possibilidade é duplicar a imagem 260x260 e renomear para 87x87 (2), dê um duplo clique na imagem 87x87 dessa forma irá abrir a pré-visualização, clique no menu ferramentas -> Ajustar tamanho (3), troque o tipo de imagem para pixels (4), altere 260x260 para 87x87 (5), clique em Ok e feche a Pré-Visualização a imagem é salva automaticamente, pronto você já tem um ícone 87x87.



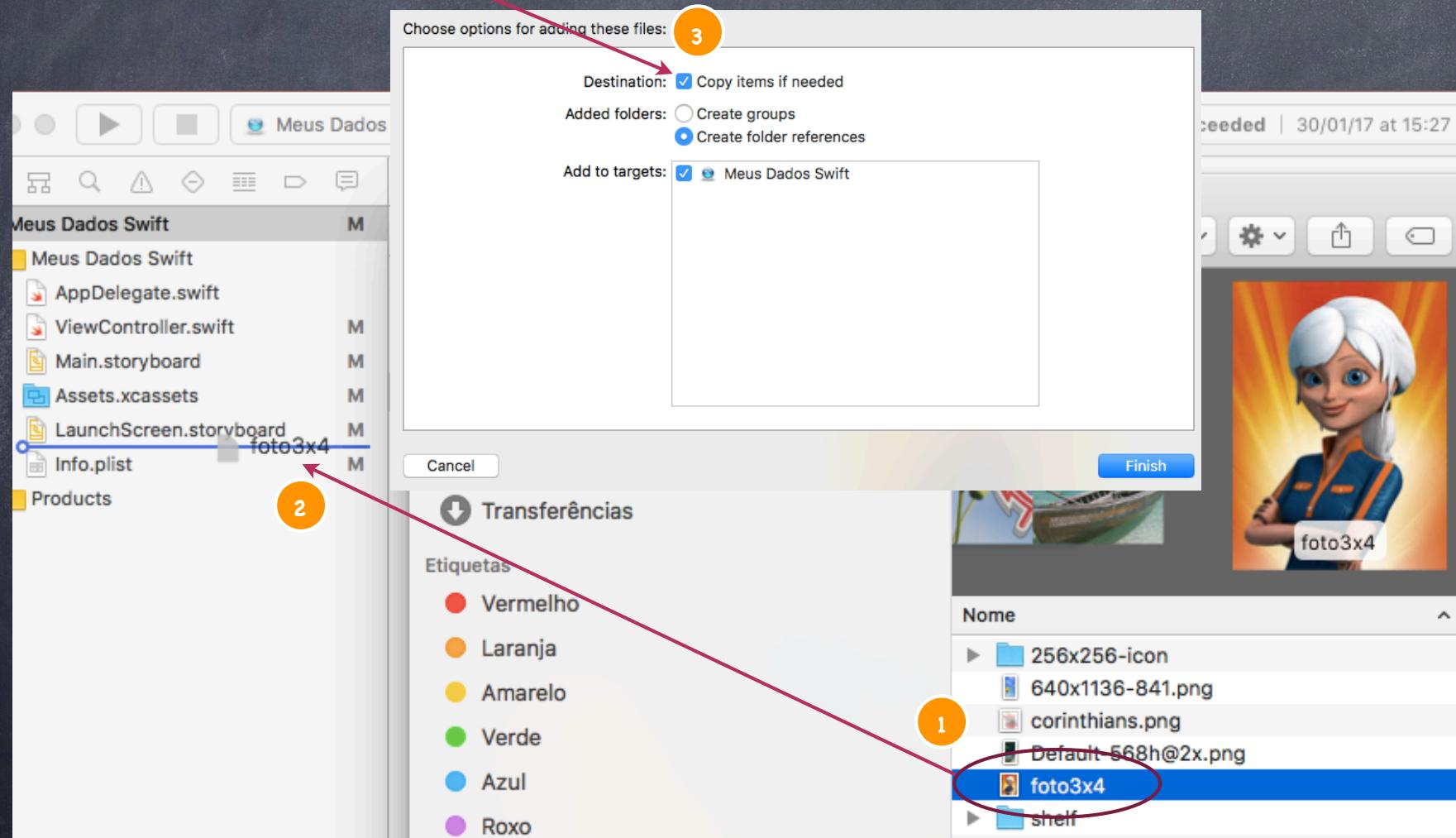
# Warning's

- Se você não errou nenhum tamanho irá aparecer apenas 3 Warning's (1), clique em Show the Issue navigator (2), aqui você verá que os avisos aparecem pela falta dos ícones de iPad (3) e o ícone da Apple Store (4), isso não impede de seu App funcionar, você poderá cuidar disso mais tarde, agora volte para Show the Project navigator (5).



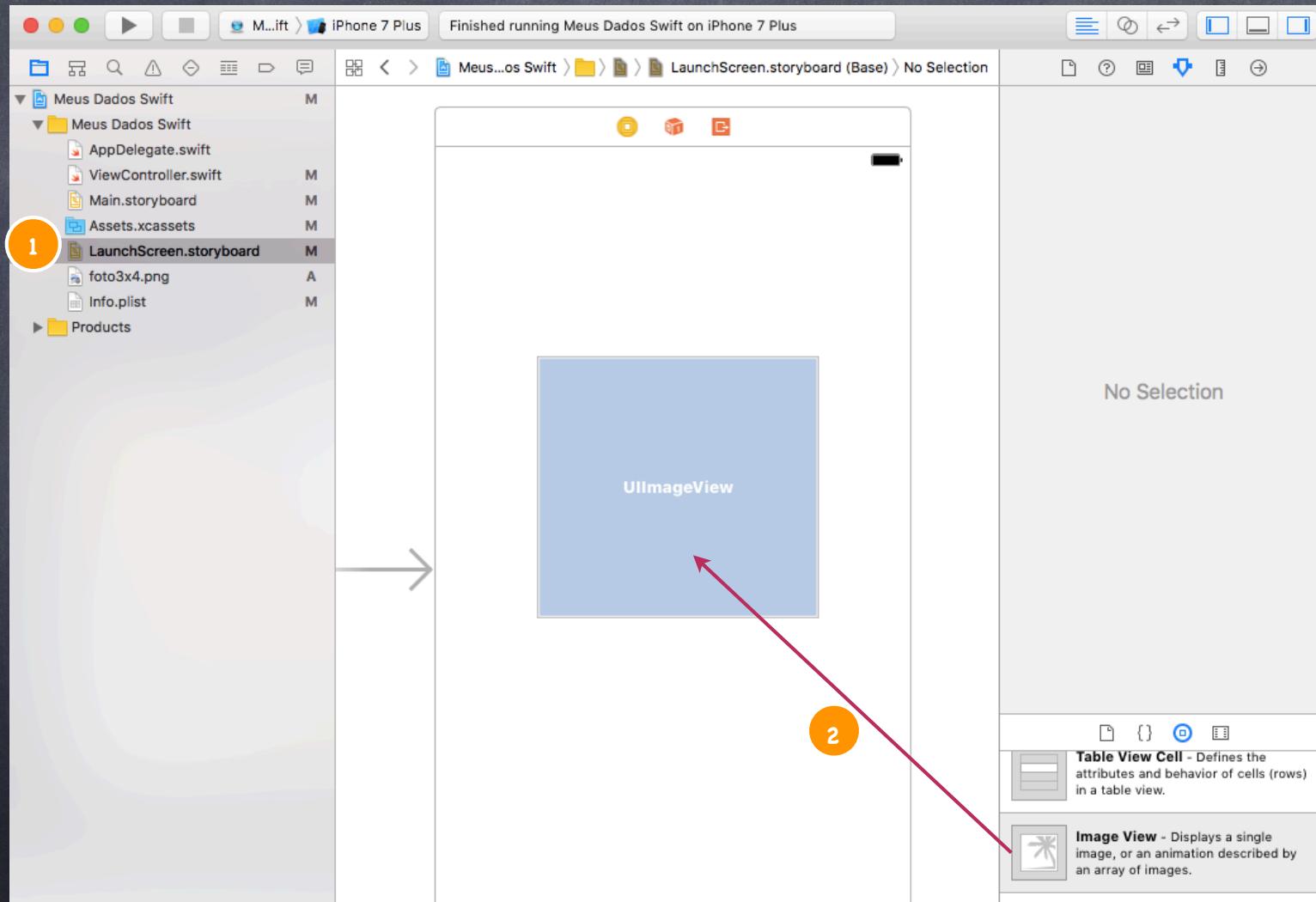
# Splash Screen – Forma 1

- Abra a pasta de imagem disponibilizada e arraste a imagem foto3x4 para o projeto, clicando aqui você gera uma cópia da imagem e não apenas uma referência. Se preferir você pode criar uma pasta com todas as imagens que vai utilizar e arrastar para o projeto uma única vez.



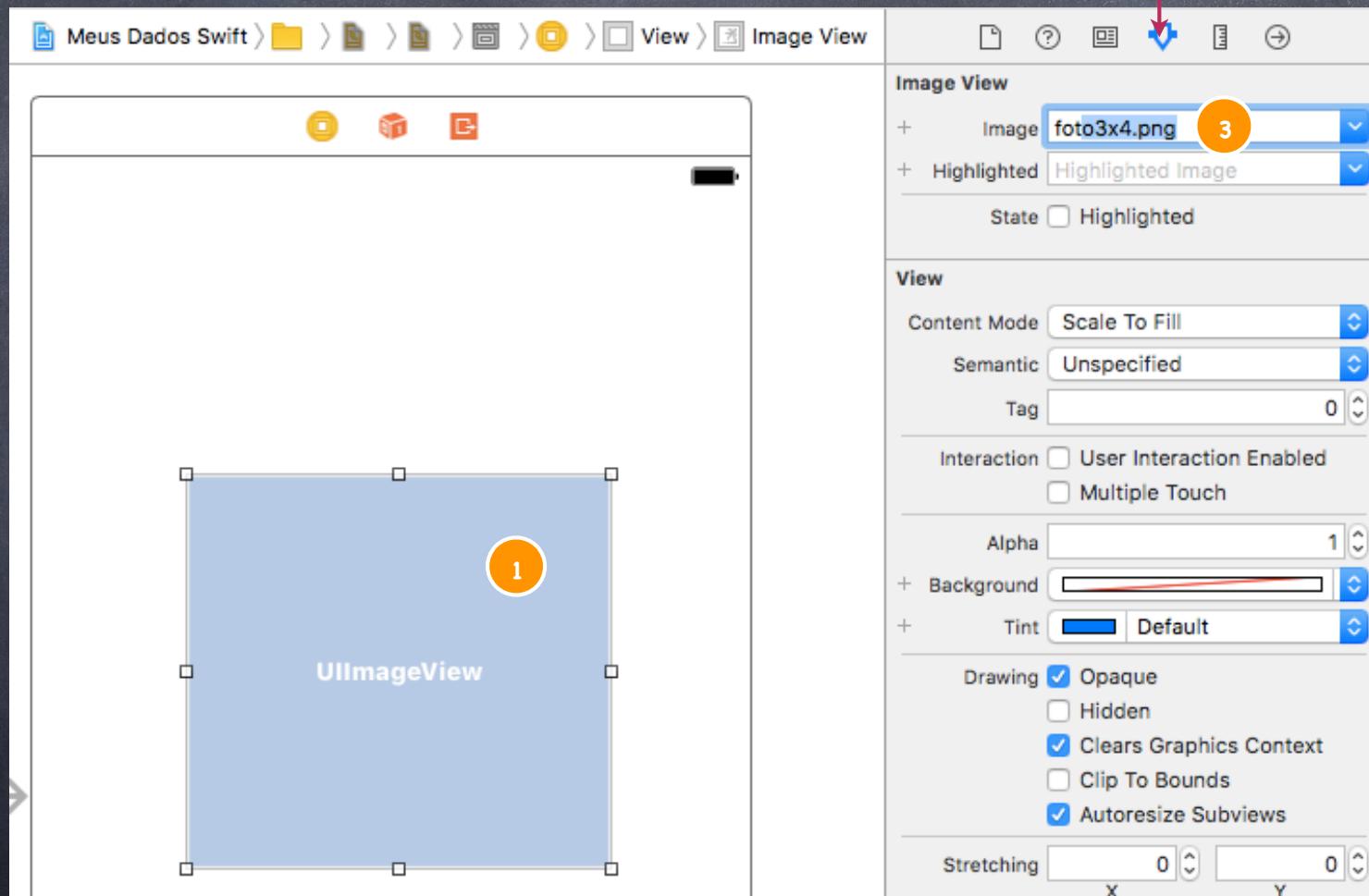
# Splash Screen – Forma 1

- Clique no LaunchScreen.storyboard (1), arraste um Image View para a View (2)



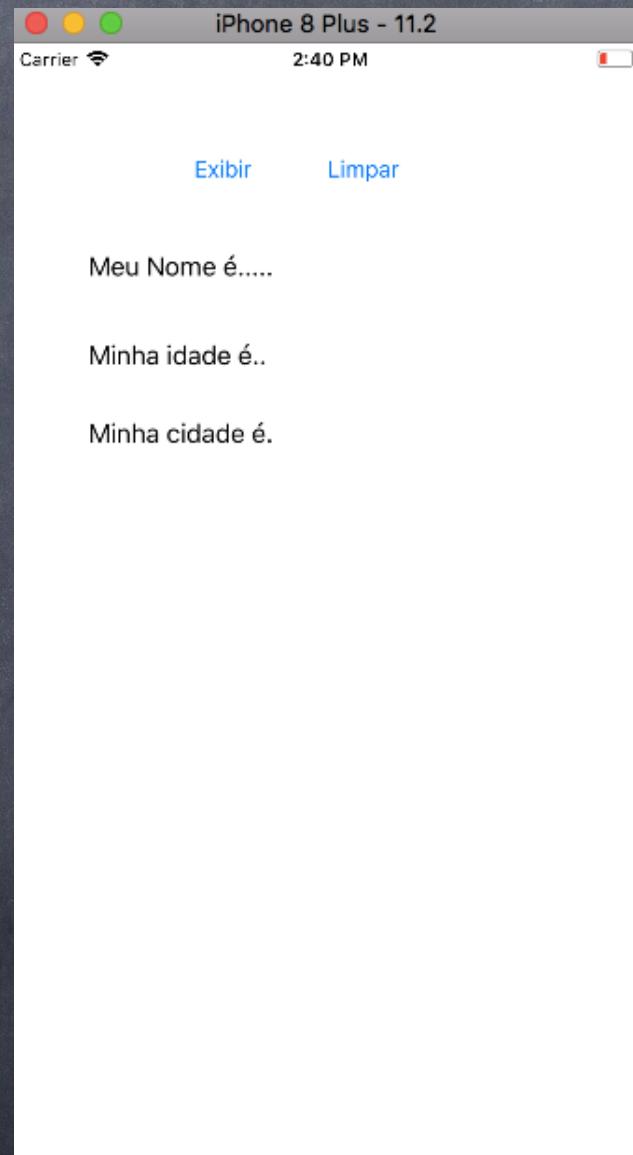
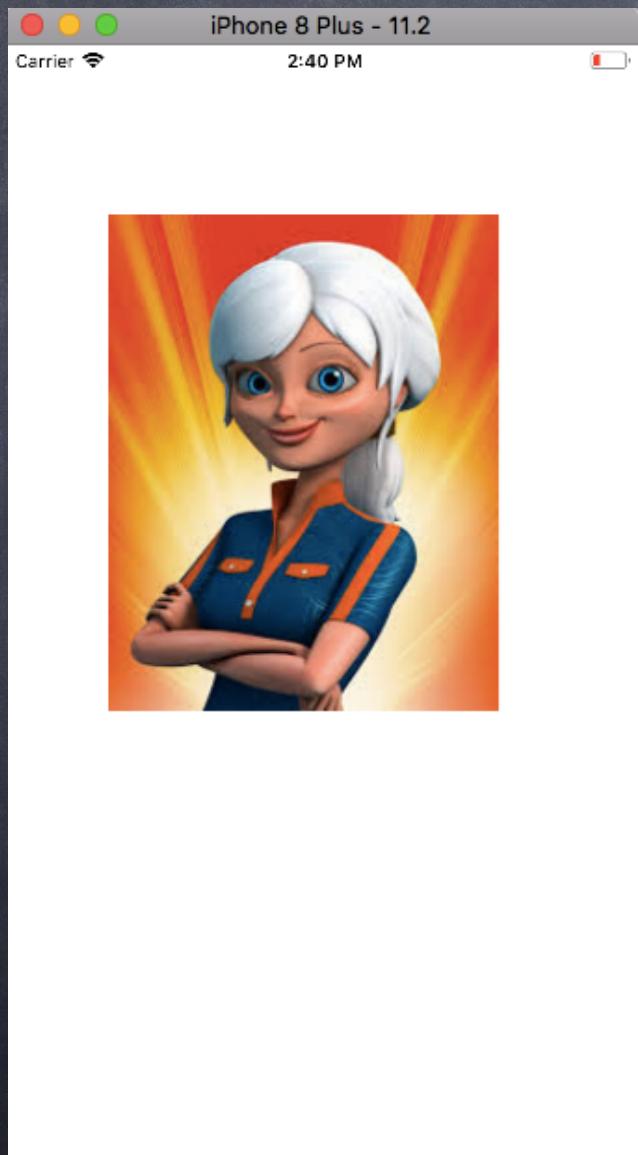
# Splash Screen – Forma 1

- Selecione a imagem (1), em Attributes Inspector (2), digite o nome da imagem e dê um enter (3).



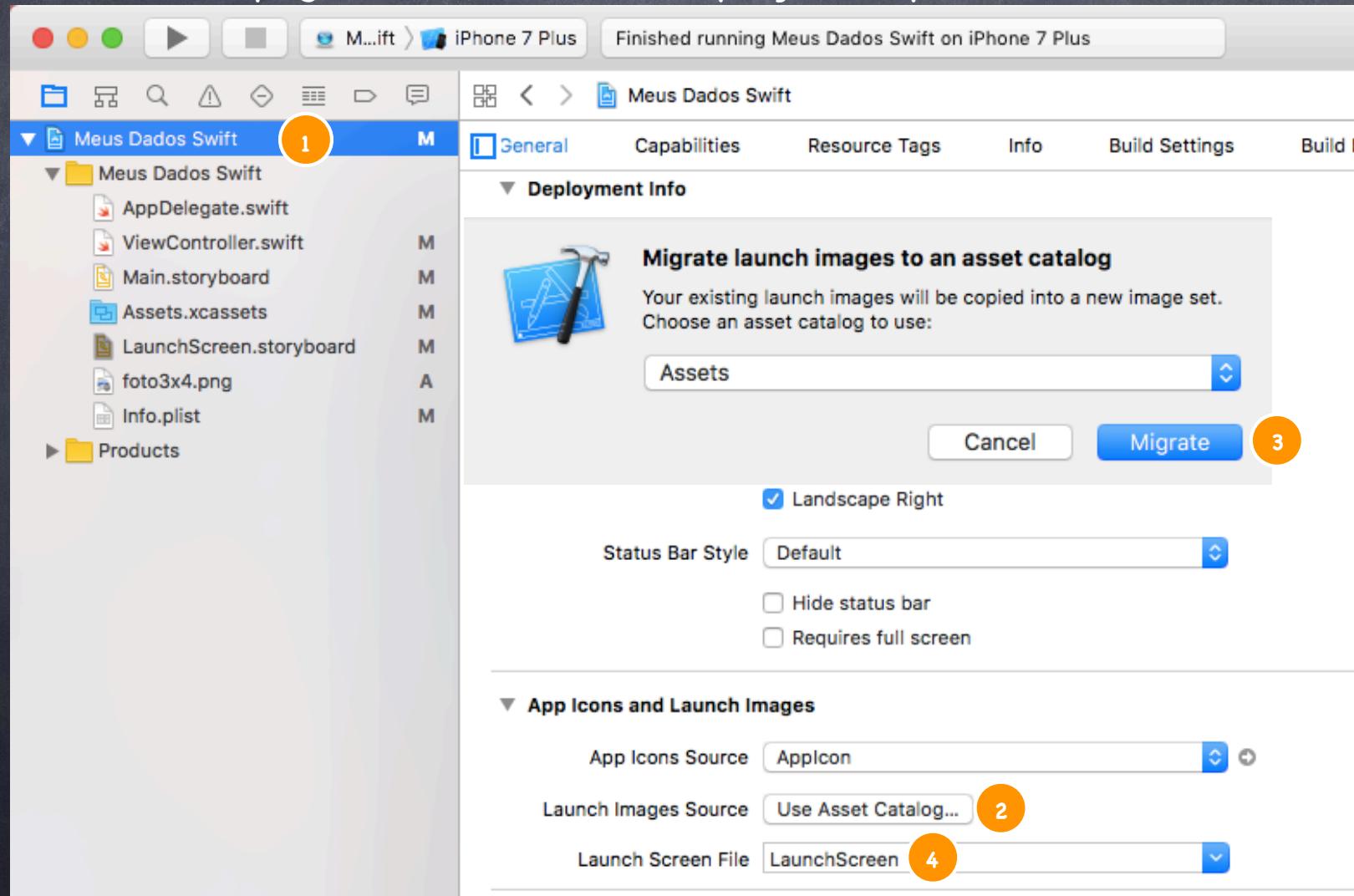
# Splash Screen – Forma 1

Execute (Cmd + R).



# Splash Screen – Forma 2

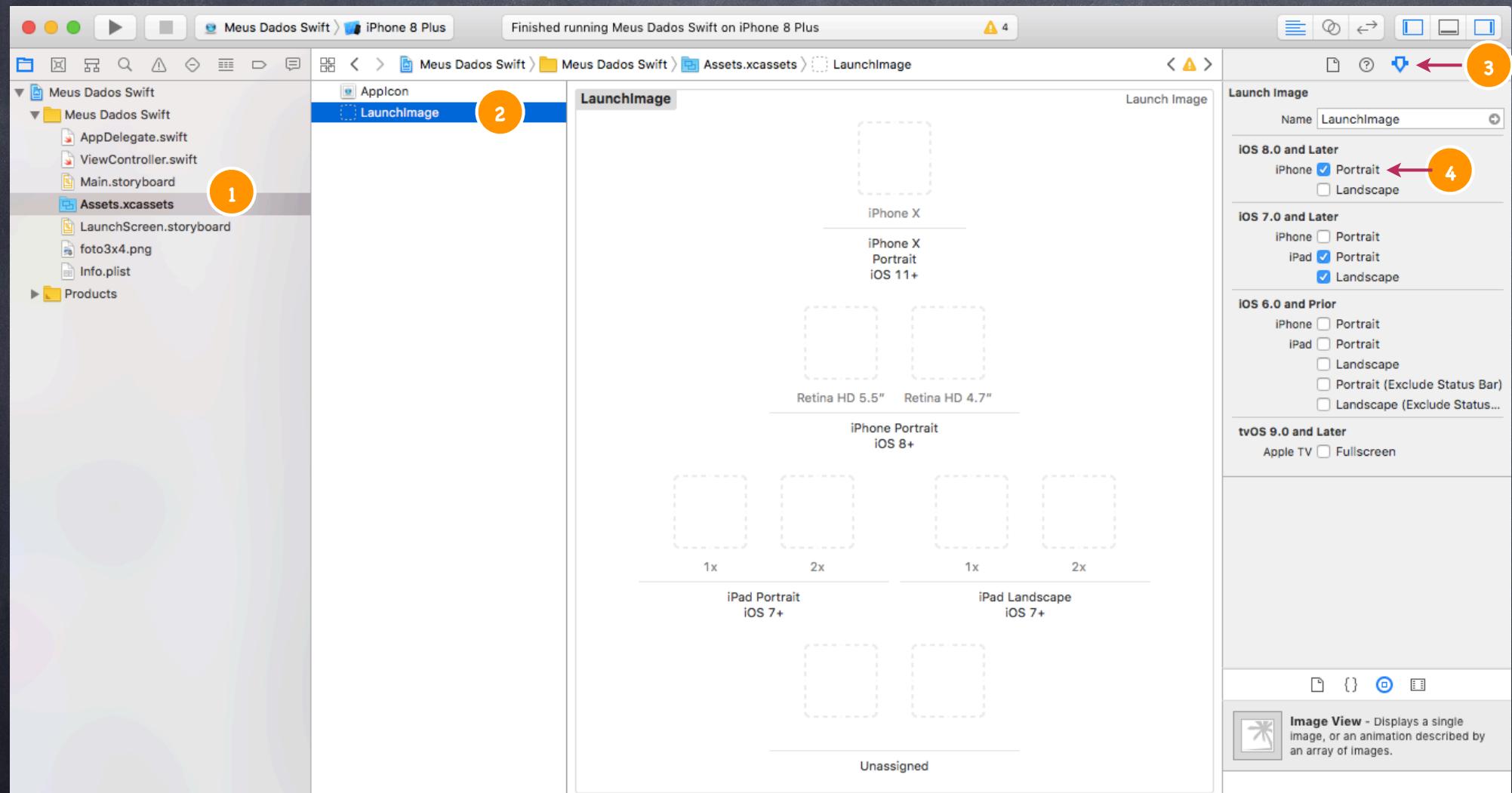
- Clique no Projeto (1) depois clique em Use Asset Catalog (2), ao aparecer a caixa de diálogo escolha Migrate (3), para não ser utilizado o LauncScreen.storyboard com a imagem anterior apague o item (4), não esqueça de apertar a tecla return/enter.



# Splash Screen - Forma 2

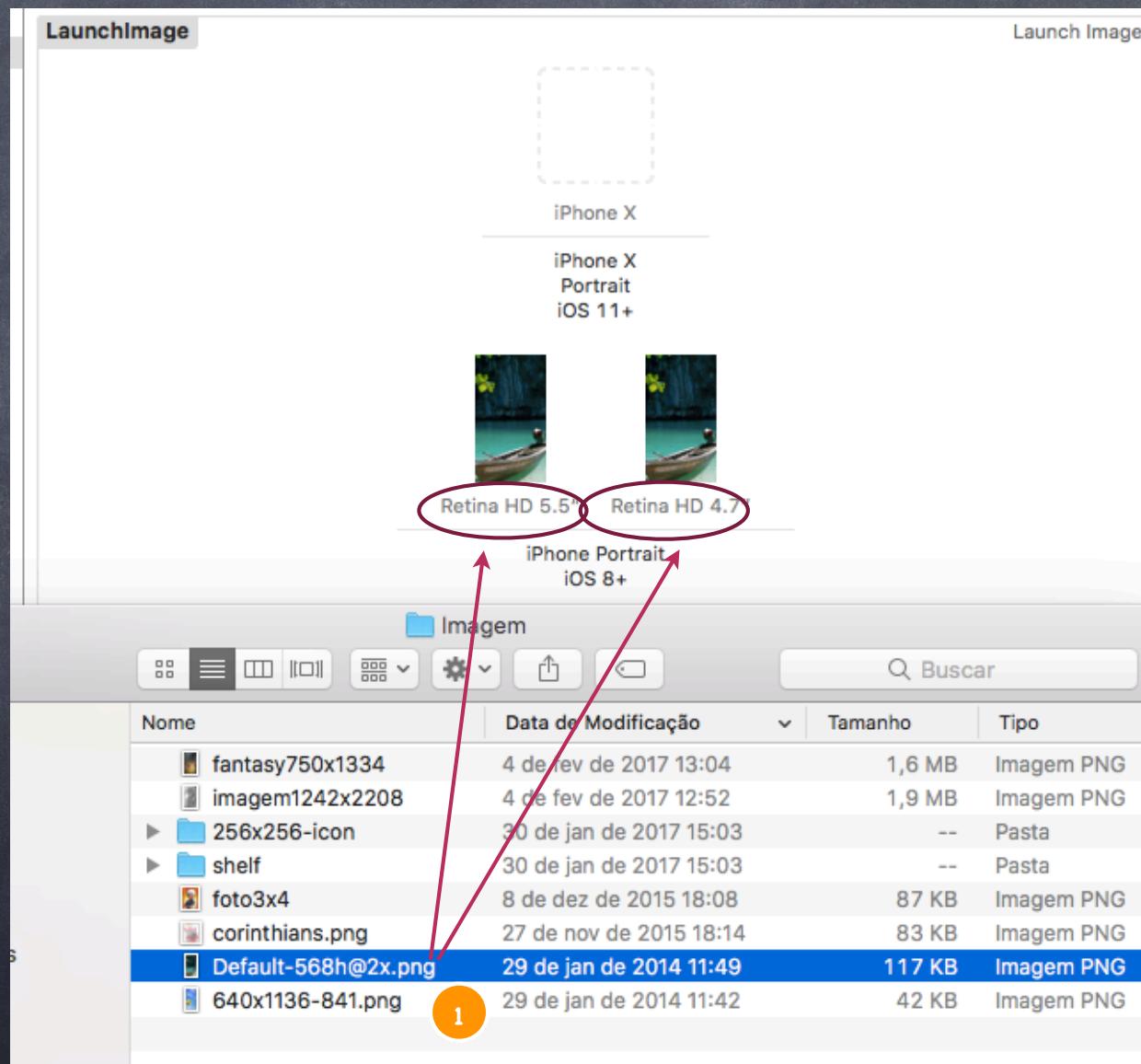
FIAP

- Clique em Assets.xcassets (1), clique em LaunchImage (2), clique em Attributes Inspector (3), depois clique no CheckBox iOS 8.0 and Later iPhone Portrait (4).



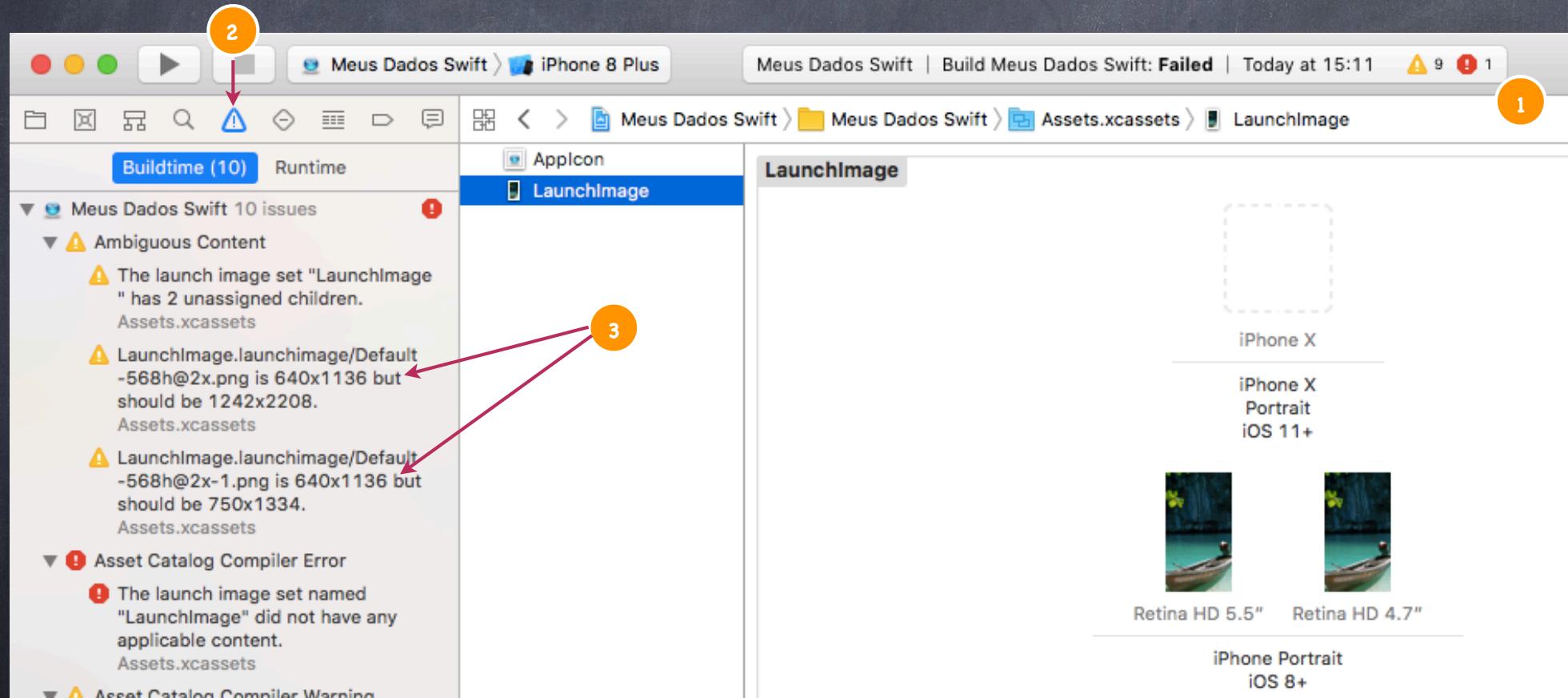
# Splash Screen – Forma 2

- Arraste a imagem disponibilizada chamada Default-568h@2x.png para cada imagem do iPhone Portrait (1).



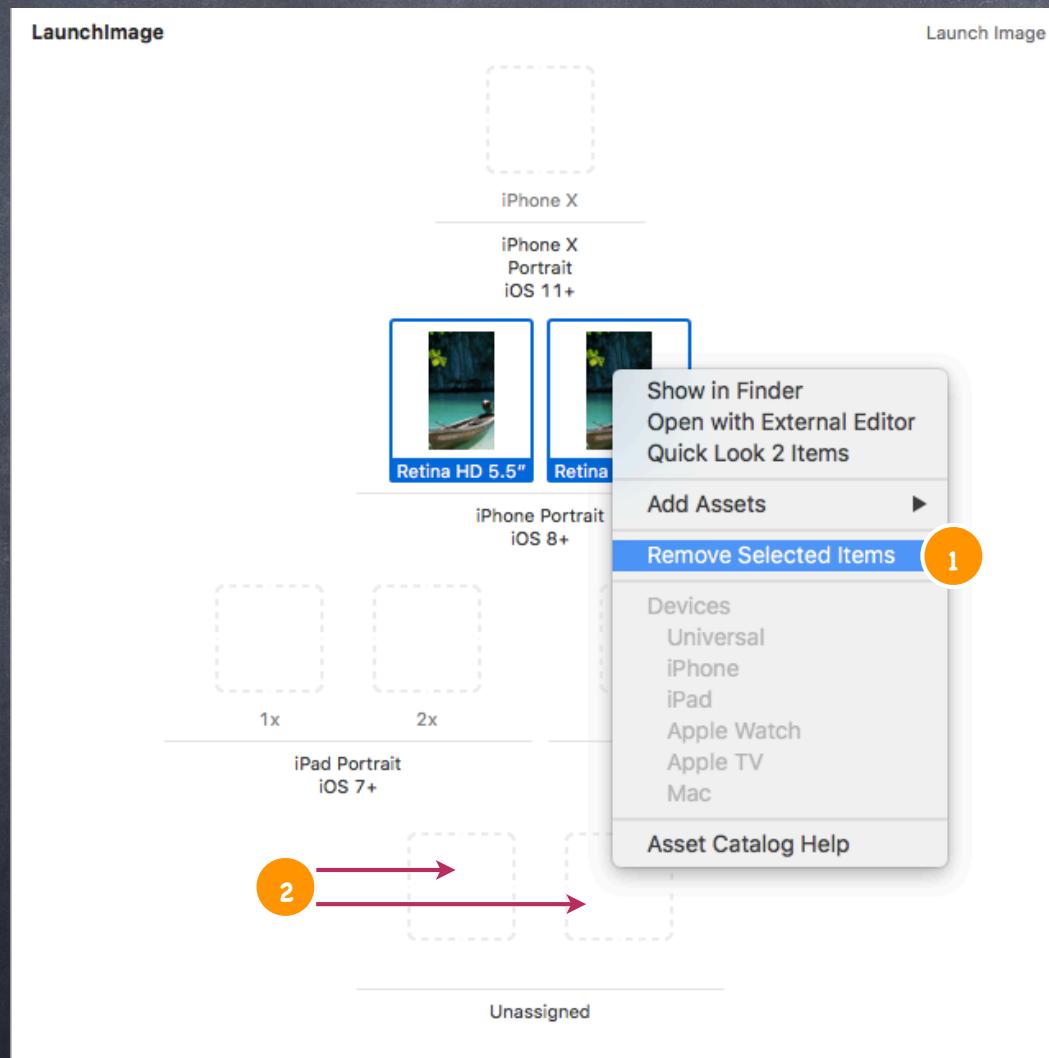
# Splash Screen - Forma 2

- Execute seu programa - Command + R, note que irá aparecer erro de compilação (1), isso ocorre porque as imagens utilizadas não estão no padrão adequado (2), para Iphone 6 Plus, 6s Plus e 7 Plus que é 1242x2208 e 750x1334 para os Iphones 6 e 6s em diante (3)



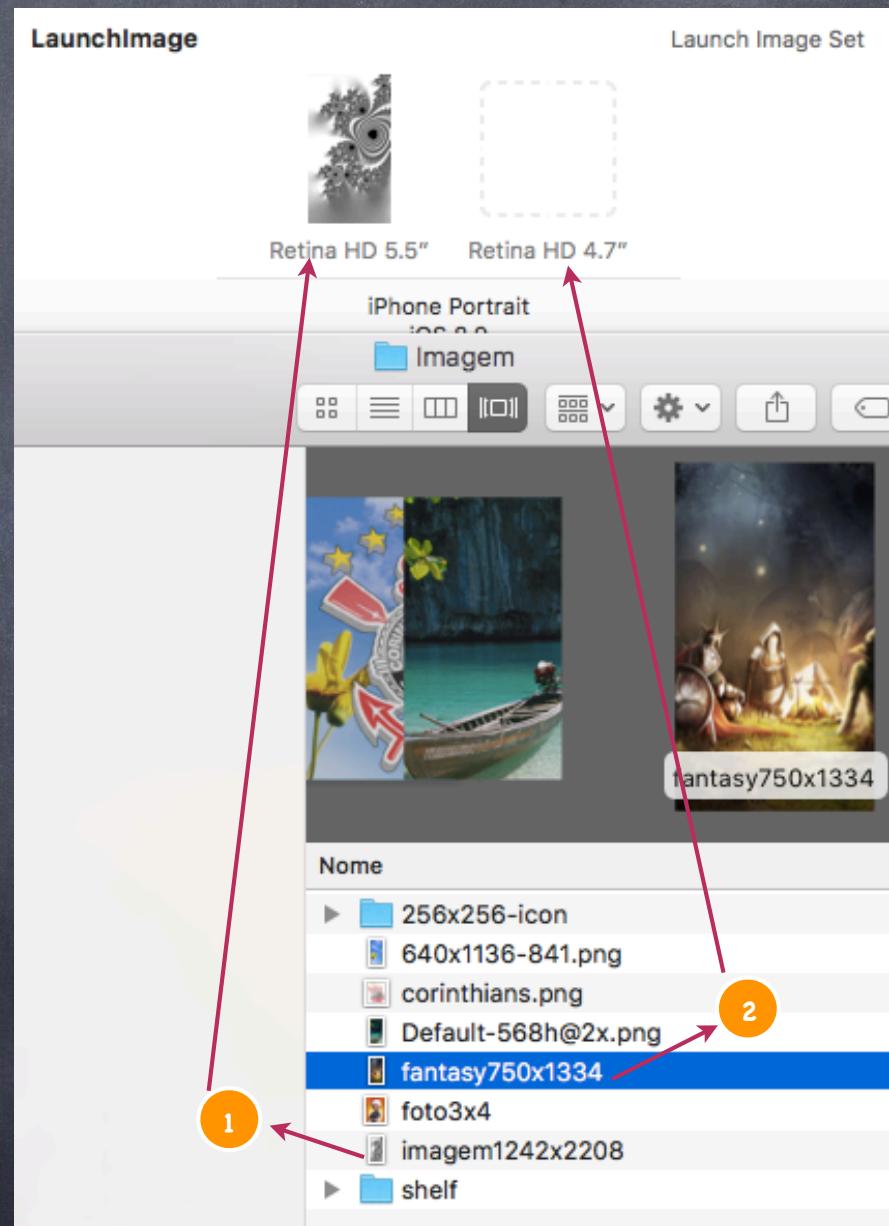
# Splash Screen – Forma 2

- Para resolver esse problema selecione as duas imagens, clique com o botão direito e escolha Remove Selected Items (1), aproveite para remover os dois templates logo abaixo. (2)



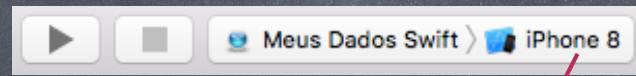
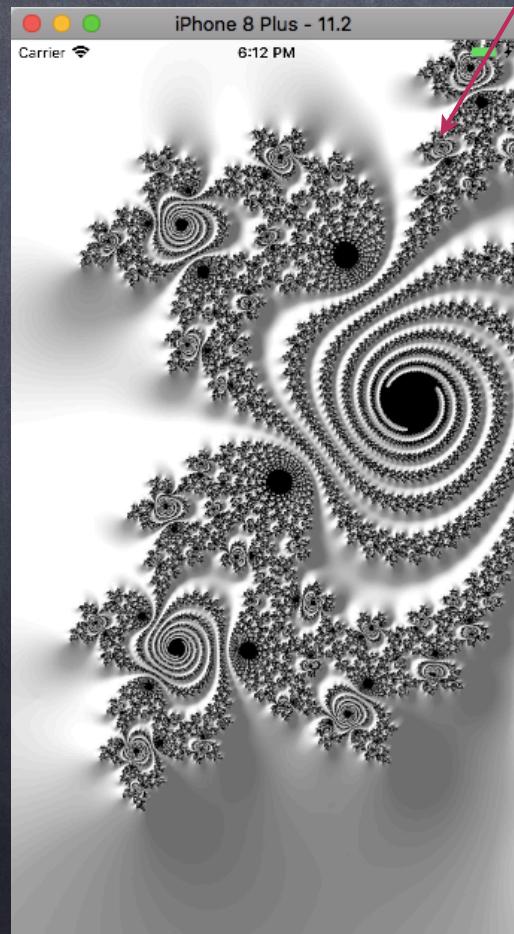
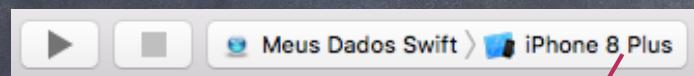
# Splash Screen – Forma 2

- Arraste a imagem 1242x2208 para a retina HD 5.5" (1) em seguida repita o processo arrastando a imagem Fantasy 750x1334 para o espaço Retina HD 4.7" (2).



# Splash Screen - Forma 2

- Execute o programa para iPhone 8Plus e para Iphone 8 e note que aparecerá telas distintas de Splash Screen.



# Imagen de splash screen

- Existe um padrão para os nomes das imagens para todos os tipos de devices da Apple (iPhone, iPad, iPhone com tela de retina, etc.), veja a tabela dos padrões de nomes por dispositivo nos próximos slides tanto para splash quanto para ícones, se você tiver uma pasta com estes nomes em formato padrão elas são úteis para iOS anterior ao 7.

# Tabela de padrões de nomes de imagem

- A tabela abaixo é para imagens splash screen.

Default.png	320x480 - iPhone, iPod
Default@2x.png	640x960 - iPhone retina 3,5 polegadas
Default-568h@2x.png	640x1136 - iPhone retina 4 polegadas
Default-Landscape~ipad.png	1024x768 - iPad
Default-Portrait~ipad.png	768x1024 - iPad
Default-Landscape@2x~ipad.png	2048x1536 - iPad retina
Default-Portrait@2x~ipad.png	1536x2048 - iPad retina

# Tabela de padrões de nomes de imagem para ícones

## • Dimensões dos ícones para iOS 6.1 ou anterior

icon.png	57x57 ícone principal iPhone, iPod Touch
icon@2x.png	114x114 – ícone principal iPhone, iPod Touch (retina)
icon-72.png	72x72 – ícone principal iPad
icon-72@2x.png	144x144 – ícone principal iPad (retina)
icon-Small.png	29x29 iPhone – ícones para resultados de pesquisa e configurações do app
icon-Small@2x.png	58x58 iPhone – ícones para resultados de pesquisa e configurações do app (retina)
icon-Small-50.png	50x50 iPad – ícones para resultados de pesquisa e configurações do app
icon-Small-50@2x.png	100x100 iPad – ícones para resultados de pesquisa e configurações do app (retina)

# Tabela de padrões de ícones e resoluções

## • Dimensões dos ícones para iOS 7 ou posterior

60x60 - ícone principal iPhone, iPod Touch
120x120 - ícone principal iPhone, iPod Touch (retina)
76x76 - ícone principal iPad
152x152 - ícone principal iPad (retina)
40x40 - Todos os dispositivos - ícones para resultados de pesquisa
80x80 - Todos os dispositivos - ícones para resultados de pesquisa (retina)
29x29 - Todos os dispositivos - ícones para configurações do app
58x58 - Todos os dispositivos - ícones para configurações do app (retina)

**Obs:** O uso de nomes de arquivos fixos para os seus ícones de aplicativos é somente para compatibilidade com versões anteriores do iOS.

# Veja os Links para Gerar Icons e Telas Splash

- ☞ <http://makeappicon.com>
- ☞ <http://ticons.fokkezb.nl>

# Utilizando break points

- Ao clicar nessa área é inserido um break point - Azul escuro break point ativo

- Ao clicar novamente em um break point ativo, ele fica azul claro e se torna inativo.

The screenshot shows the Xcode interface during a debugging session. In the top navigation bar, the title is "MeusDados" and the subtitle is "Running MeusDados on iPhone 7 Plus". A warning icon with the number "1" is visible. The left sidebar shows the project structure and the current thread: "Thread 1 Queue: com.apple.main-thread (Serial)" with a stack trace starting from "ViewController.viewDidLoad". The main editor area displays the code for ViewController.m. A red arrow points to the line "meuLabel1.text = @"Meu nome é .....";" where a blue breakpoint is active. Another red arrow points to the same line after it has been cleared, turning it light blue. The bottom toolbar shows various debug buttons, and the bottom right corner shows the console output with the message "2017-02-23 09:48:36.521 sDados[2383:129343] Atenção".

```

// ViewController.m
//
// Created by Agesandro Scarpioni on 30/01/17.
// Copyright © 2017 Agesandro Scarpioni. All rights reserved.

#import "ViewController.h"

@interface ViewController : UIViewController

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    NSLog(@"Atenção");
    meuLabel1.text = @"Meu nome é .....";
    meuLabel2.text = @"Minha idade é.....";
    meuLabel3.text = @"Minha cidade é ....." ;
}

- (IBAction)limpar:(id)sender{
    meuLabel1.text = @"";
    meuLabel2.text = @"";
    meuLabel3.text = @"";
}

- (void) didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

- (IBAction)exibir:(id)sender {
    meuLabel1.text = @"Agesandro Scarpioni";
    meuLabel2.text = @"18 anos";
    meuLabel3.text = @"São Palo";
}
@end

```

Para apagar todos os break point's, use esse atalho e com o botão direito na classe de seu interesse, escolha delete break point.

Dica: Utilize o botão Step Over para avançar por entre as linhas após o break point, use a janela Console para ver as informações "printadas" com NSLog.

# Prática 1

Criação de um programa para testarmos todos os conceitos deste tópico, tente fazer sem olhar os slides.

- Crie um projeto novo com 4 labels e 2 botões, faça aparecer nestes label's as seguintes informações:
- Botão 1: seu nome completo, sua cidade de nascimento, respectivamente nos 2 primeiros label's
- Botão 2: seu email e a data de nascimento nos outros dois label's.
- Altere o nome e o ícone do aplicativo, altere a imagem de splash screen.

# Próxima aula

- Novos objetos em novas interfaces.