

App Requisições http com Json

parte 1 - Desenhando e preparando a Interface

X-Code com Objective-C
Prof. Agesandro Scarpioni

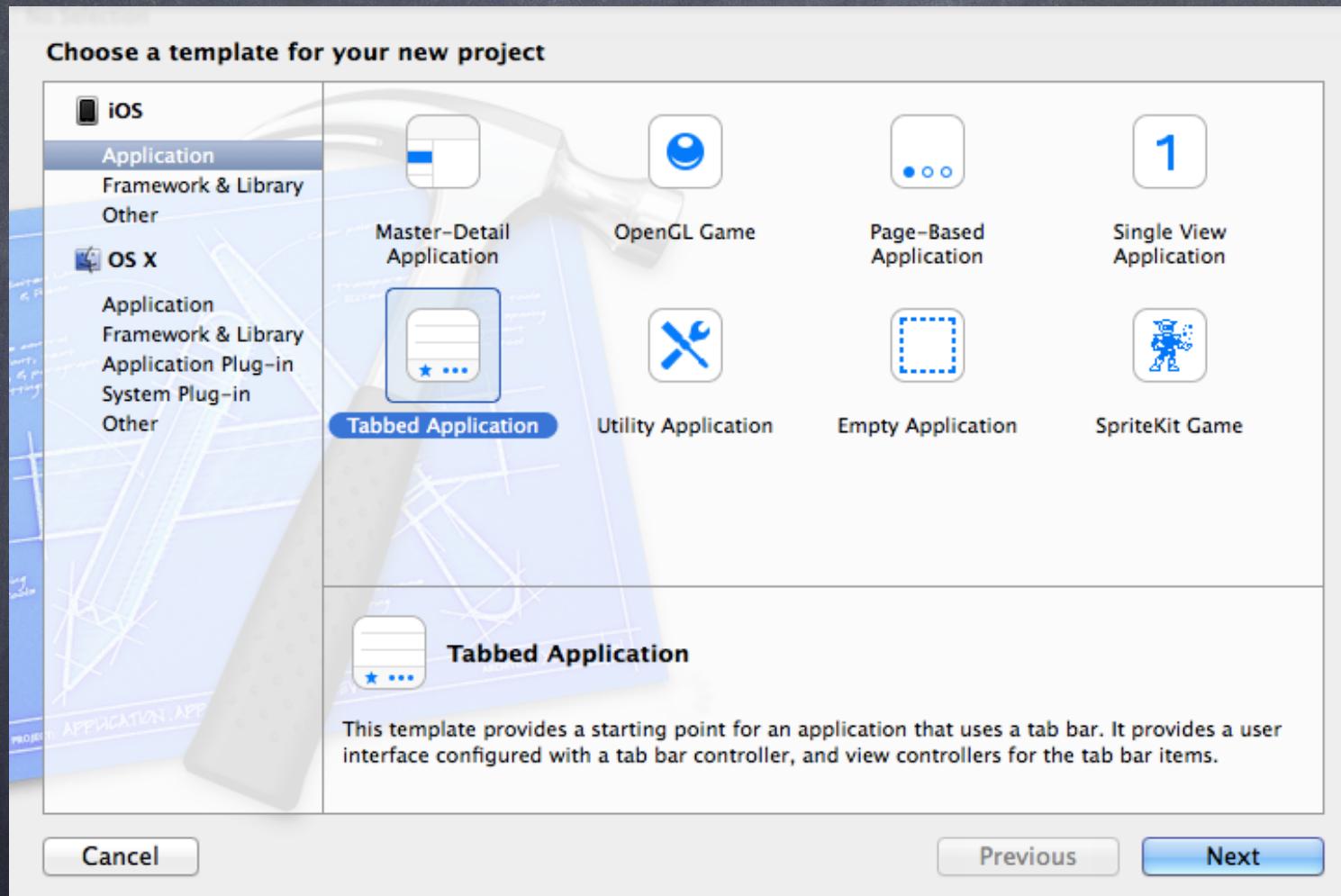
App com Table View e Web View

- Vamos criar um aplicativo para que uma primeira tela exiba alguns itens em um Table View, caso o usuário clique em um item, as informações deverão ser exibidas em uma segunda tela via WebView, se as informações não estiverem disponíveis no aparelho as mesmas deve ser requisitadas na Web e atualizadas no dispositivo.

Iniciando o Projeto

FIAP

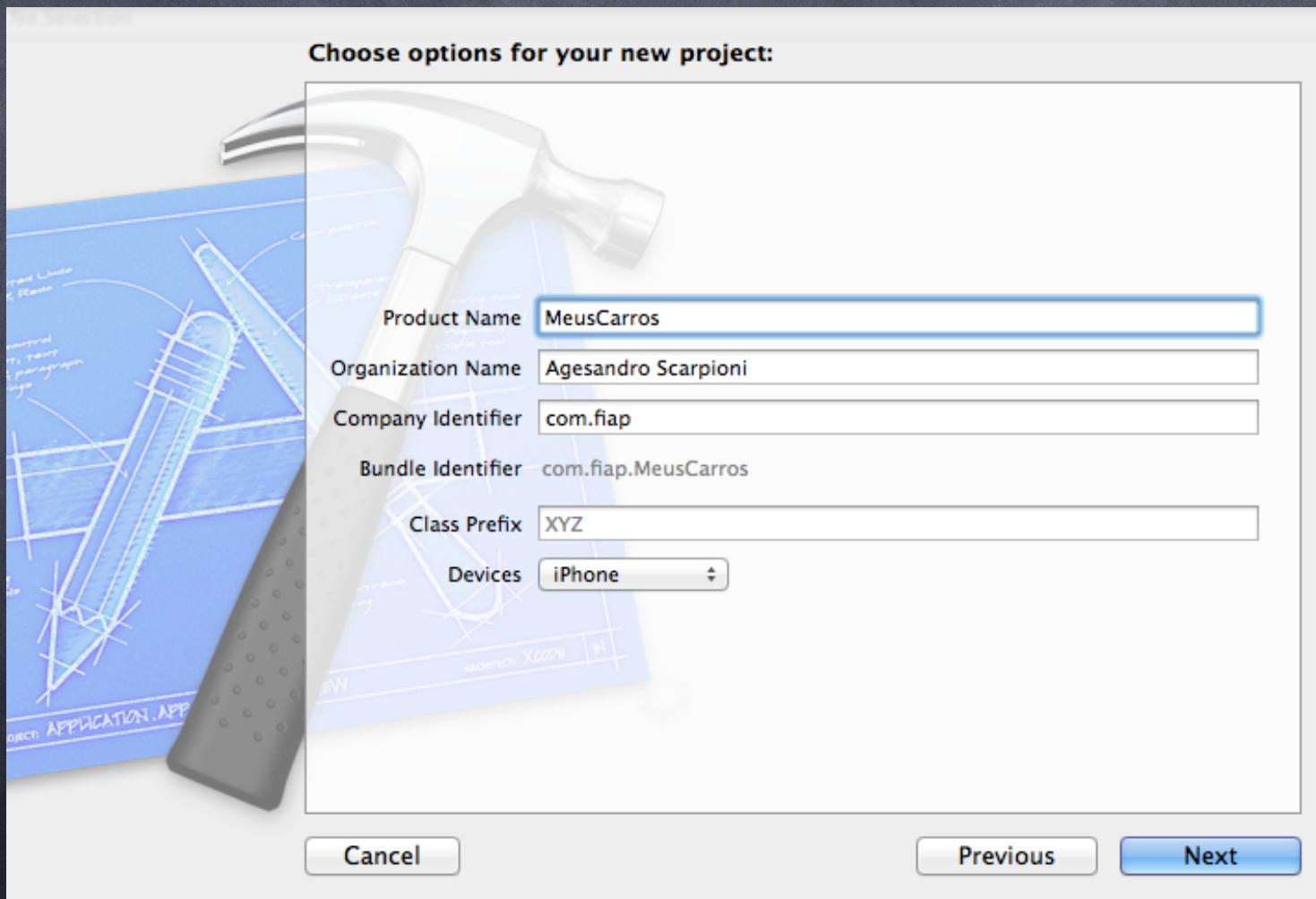
- Clique em File -> New Project -> IOS -> Application -> Tabbed Application.



OBS: Este é o template que já cria uma Tab Bar Controller com dois botões ligados a duas telas do tipo View Controller.

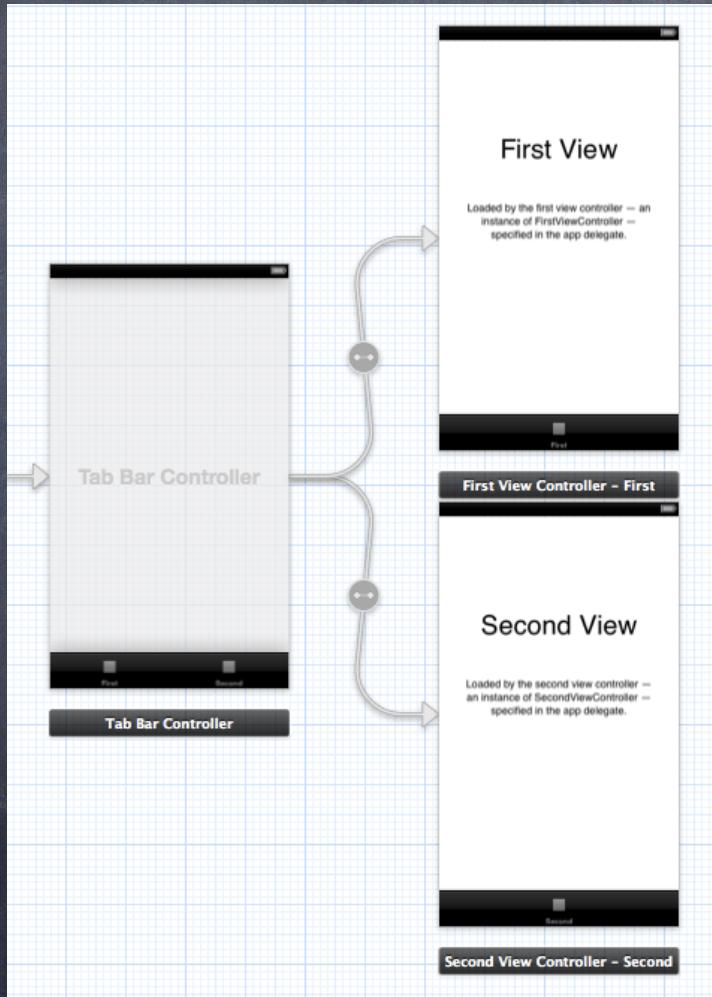
O App

- Preencha com os dados abaixo, lembre-se que o Company Identifier é como se fosse o pacote no Java ou o namespace do VB, marque o 1º e 2º check box. Em Devices selecione iPhone.



Tudo em seu lugar

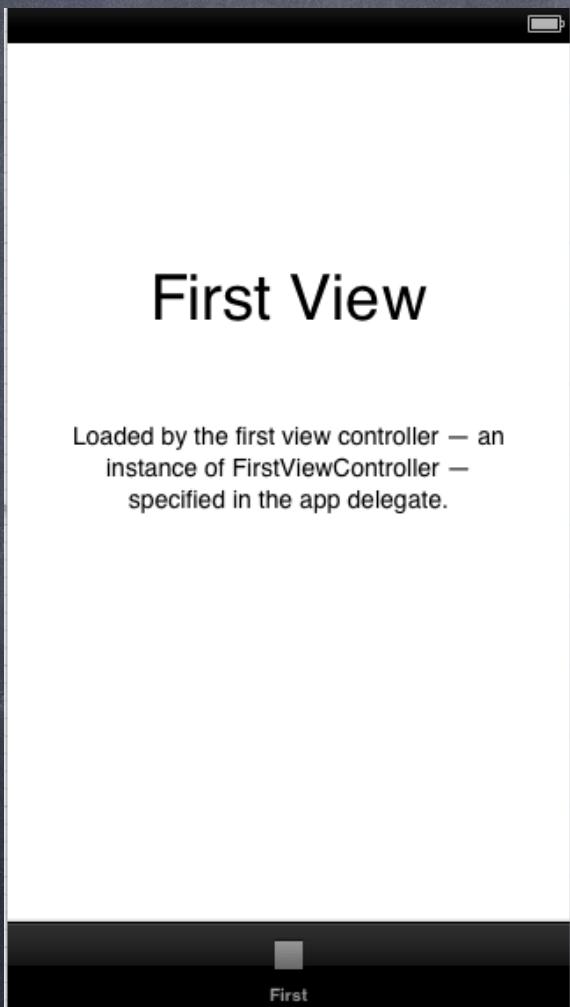
- Note que ao iniciar um objeto do tipo Tabbed Application, um storyboard como no exemplo abaixo será exibido.



Obs: Se você executar o programa agora, irá aparecer a tela de iPhone no simulador sem termos que programar ao menos uma linha. Para executar clique em Run no canto superior esquerdo ou use command + R.

Alterando os objetos

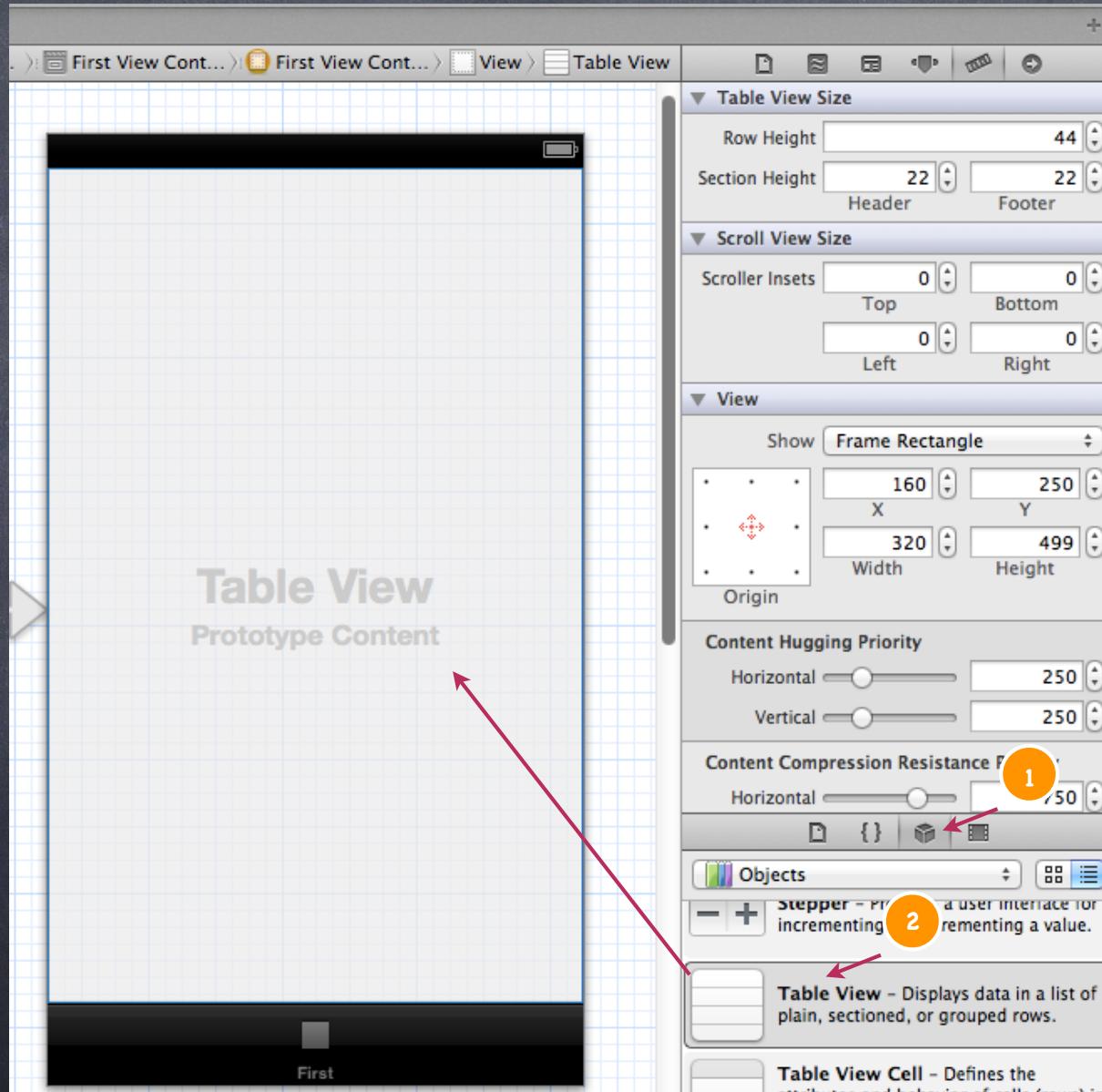
- Apague os dois objetos de dentro da ViewController abaixo e insira um objeto do tipo TableView como mostra no próximo slide.



Dica: Se você esqueceu como trocar os textos dos botões e labels veja no próximo slide.

Incluindo um Table View

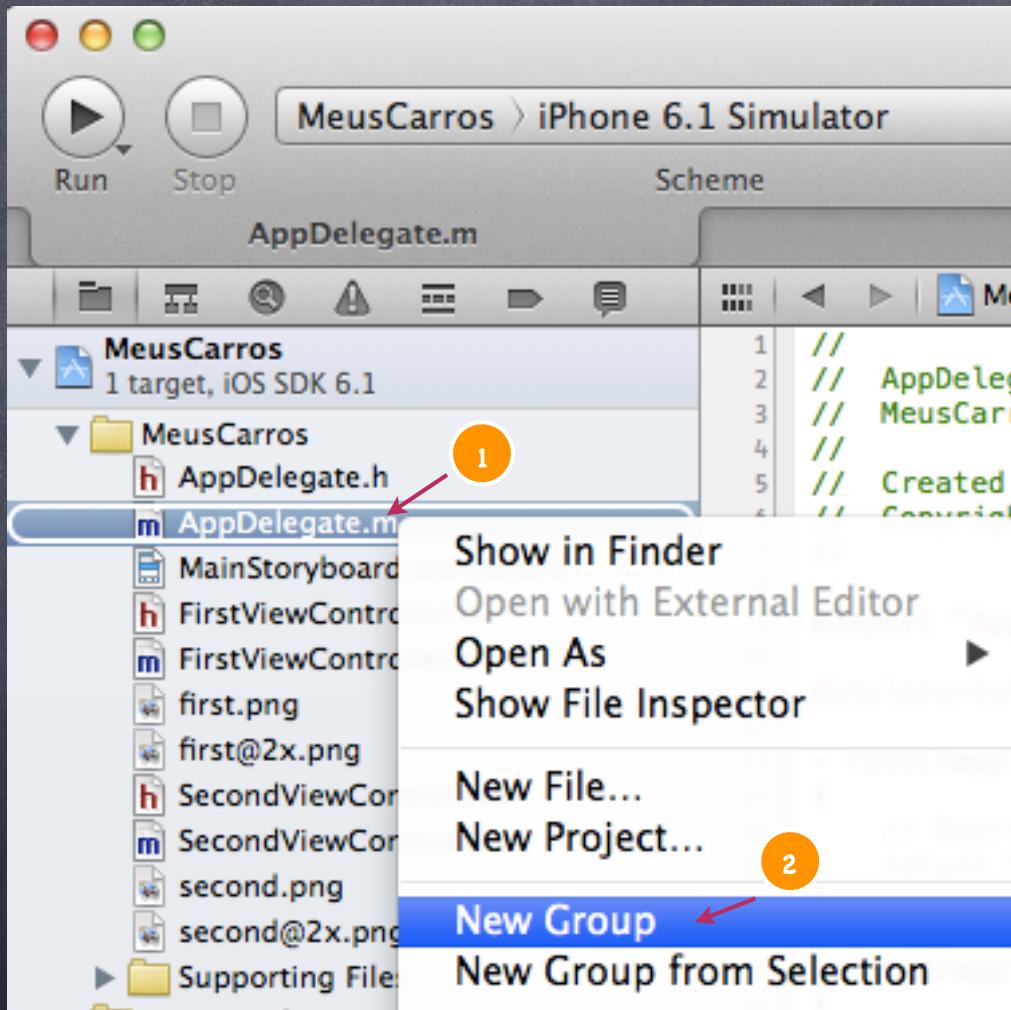
- Clique na sequência de botões para incluir um Table View



Organize seu Projeto

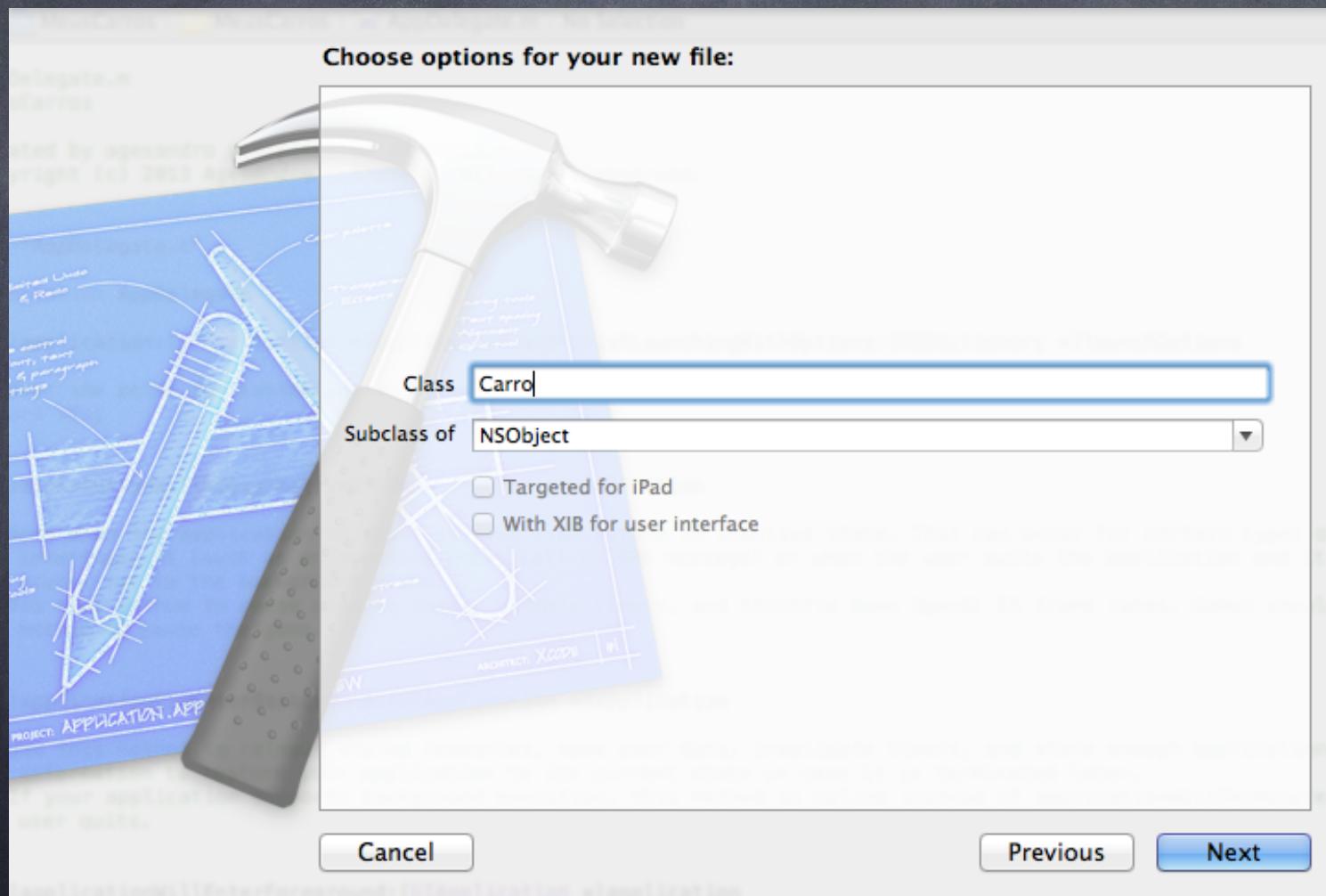
FIAP

- Crie uma pasta abaixo do AppDelegate.m, clique nesse item com o botão direito e escolha New Group, a nova pasta deve se chamar Classes.



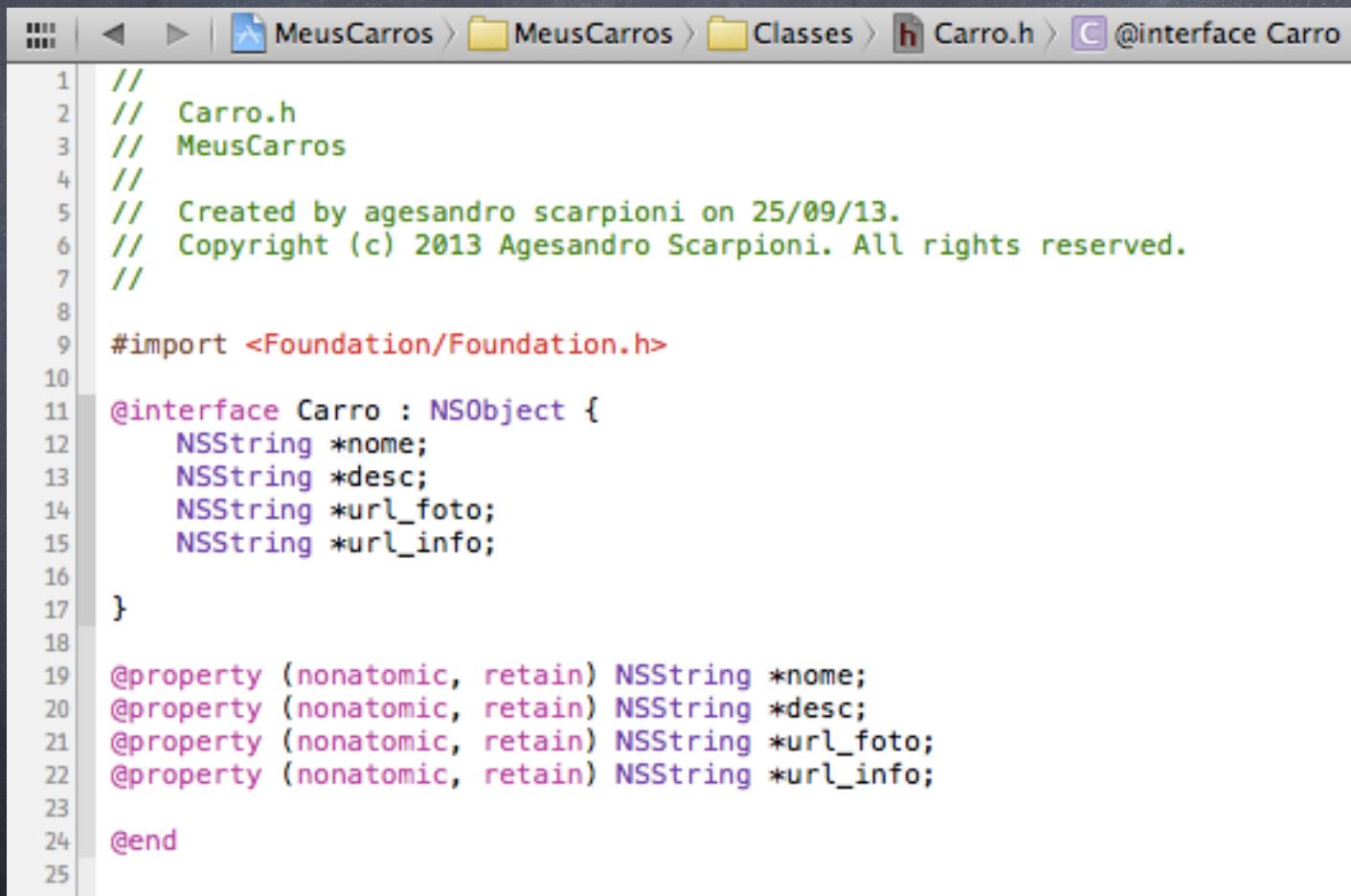
Classe Carro

- Criar uma Classe chamada Carro dentro da pasta Classe criada no slide anterior, filha de NSObject, para criar a classe use File --> New --> File ou Command + N.



Classe Carro.h

- Na classe carro declare as variáveis abaixo e seus atributos (@property) conforme os tipos indicados.



The screenshot shows the Xcode interface with the file 'Carro.h' selected in the navigation bar. The code editor displays the following content:

```
1 //  
2 //  Carro.h  
3 // MeusCarros  
4 //  
5 // Created by agesandro scarpioni on 25/09/13.  
6 // Copyright (c) 2013 Agesandro Scarpioni. All rights reserved.  
7 //  
8  
9 #import <Foundation/Foundation.h>  
10  
11 @interface Carro : NSObject {  
12     NSString *nome;  
13     NSString *desc;  
14     NSString *url_foto;  
15     NSString *url_info;  
16 }  
17  
18 @property (nonatomic, retain) NSString *nome;  
19 @property (nonatomic, retain) NSString *desc;  
20 @property (nonatomic, retain) NSString *url_foto;  
21 @property (nonatomic, retain) NSString *url_info;  
22  
23 @end  
24  
25
```

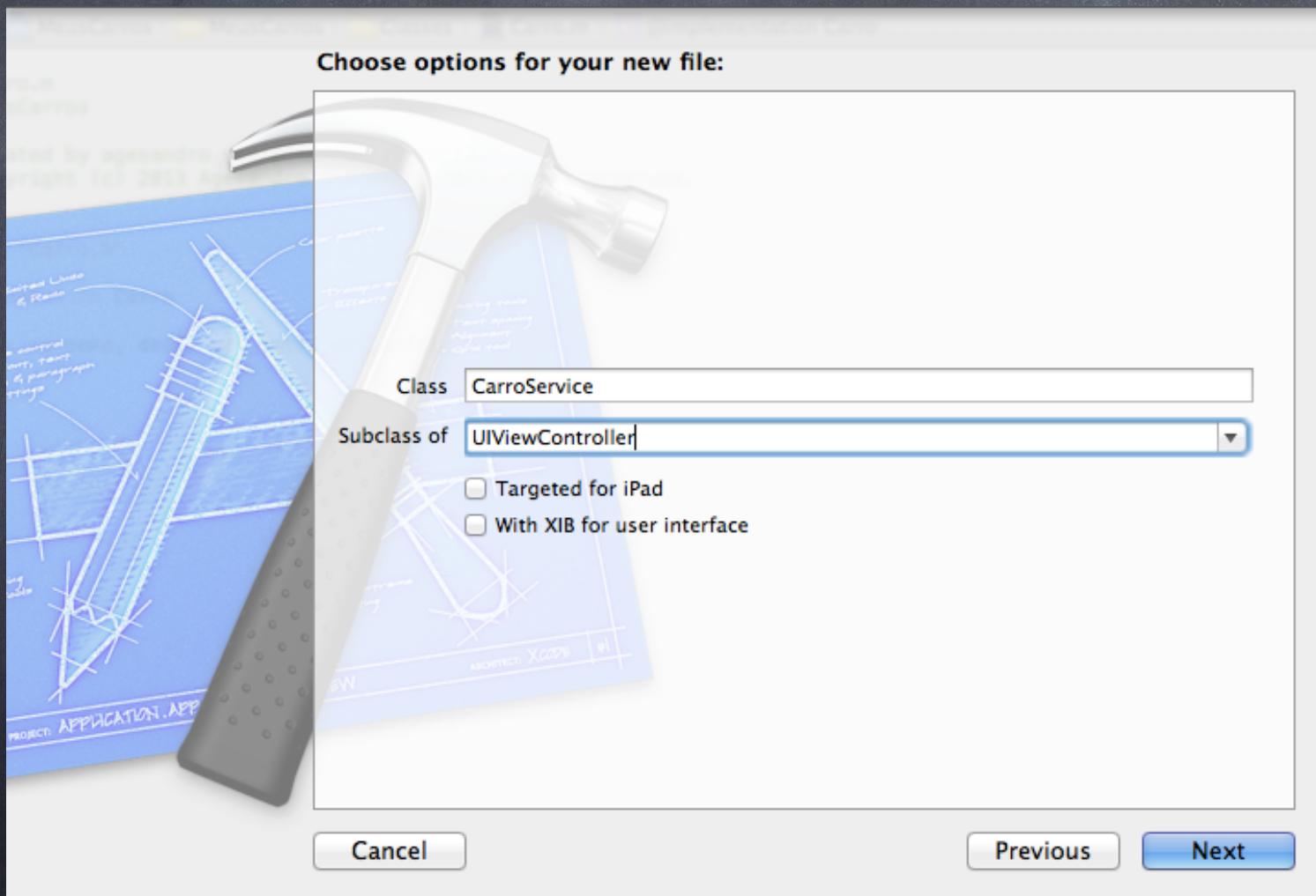
Classe Carro.m

- Chame o @synthesize para cada um dos atributos.

```
1 //  
2 // Carro.m  
3 // MeusCarros  
4 //  
5 // Created by agesandro scarpioni on 25/09/13.  
6 // Copyright (c) 2013 Agesandro Scarpioni. All rights reserved.  
7 //  
8  
9 #import "Carro.h"  
10  
11 @implementation Carro  
12  
13 @synthesize nome, desc, url_foto, url_info;  
14  
15 @end  
16
```

Classe CarroService

- Criar a classe CarroService como sub classe de UIViewController, também deve estar dentro da pasta Classe, CarroService vai possuir um método que retorna um array de carros para popularmos o TableView.



Classe CarroService.h

FIAP

- No arquivo .h, crie um método que retorne um array mutável com os carros.

The screenshot shows the Xcode interface with the project 'MeusCarros' selected. The scheme is set to 'iPhone 6.1 Simulator'. The 'Breakpoints' tab is active. The left sidebar shows the project structure with 'MeusCarros' target, 'MeusCarros' folder containing 'AppDelegate.h', 'AppDelegate.m', 'Classes' folder with 'Carro.h', 'Carro.m', and 'CarroService.h' (which is currently selected), and other files like 'MainStoryboard.storyboard', 'FirstViewController.h', 'FirstViewController.m', and 'first.png'. The right pane displays the code for 'CarroService.h':

```
// CarroService.h
// MeusCarros
//
// Created by agesandro scarpioni on 25/09/13.
// Copyright (c) 2013 Agesandro Scarpioni. All rights reserved.

#import <UIKit/UIKit.h>

@interface CarroService : UIViewController {
}

+ (NSMutableArray *) getCarros;
@end
```

Classe CarroService.h

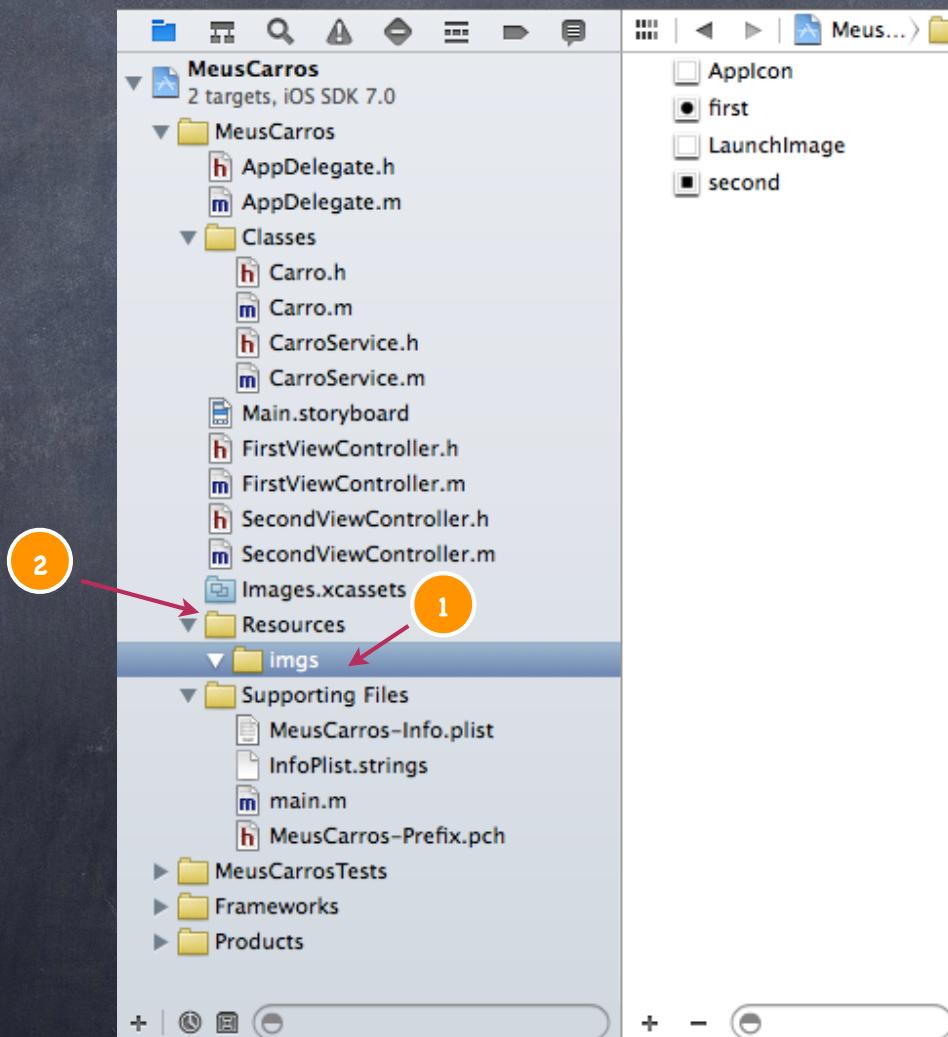
FIAP

- No arquivo.m import o carro.h, faça a implementação do método, adicionando cada item no objeto carro para depois retornar o array.

```
1 //  
2 // CarroService.m  
3 // MeusCarros  
4 //  
5 // Created by agesandro scarpioni on 25/09/13.  
6 // Copyright (c) 2013 Agesandro Scarpioni. All rights reserved.  
7 //  
8  
9 #import "CarroService.h"  
10 #import "Carro.h"  
11  
12 @interface CarroService()  
13  
14 @end  
15  
16 @implementation CarroService  
17  
18 + (NSMutableArray *) getCarros {  
19  
20     NSMutableArray *carros = [[NSMutableArray alloc] init];  
21  
22     for (int i = 0; i < 20; i++) {  
23         Carro *c = [[Carro alloc] init];  
24  
25         [c setName:[NSString stringWithFormat:@"Carro %d", i]];  
26         [c setDesc:[NSString stringWithFormat:@"Desc Carro %d", i]];  
27         [c setUrl_foto:@"1Ferrari_FF.png"];  
28         [c setUrl_info:@"http://www.google.com.br"];  
29  
30         [carros addObject:c];  
31     }  
32 }  
33  
34 return carros;  
35 }  
36 }
```

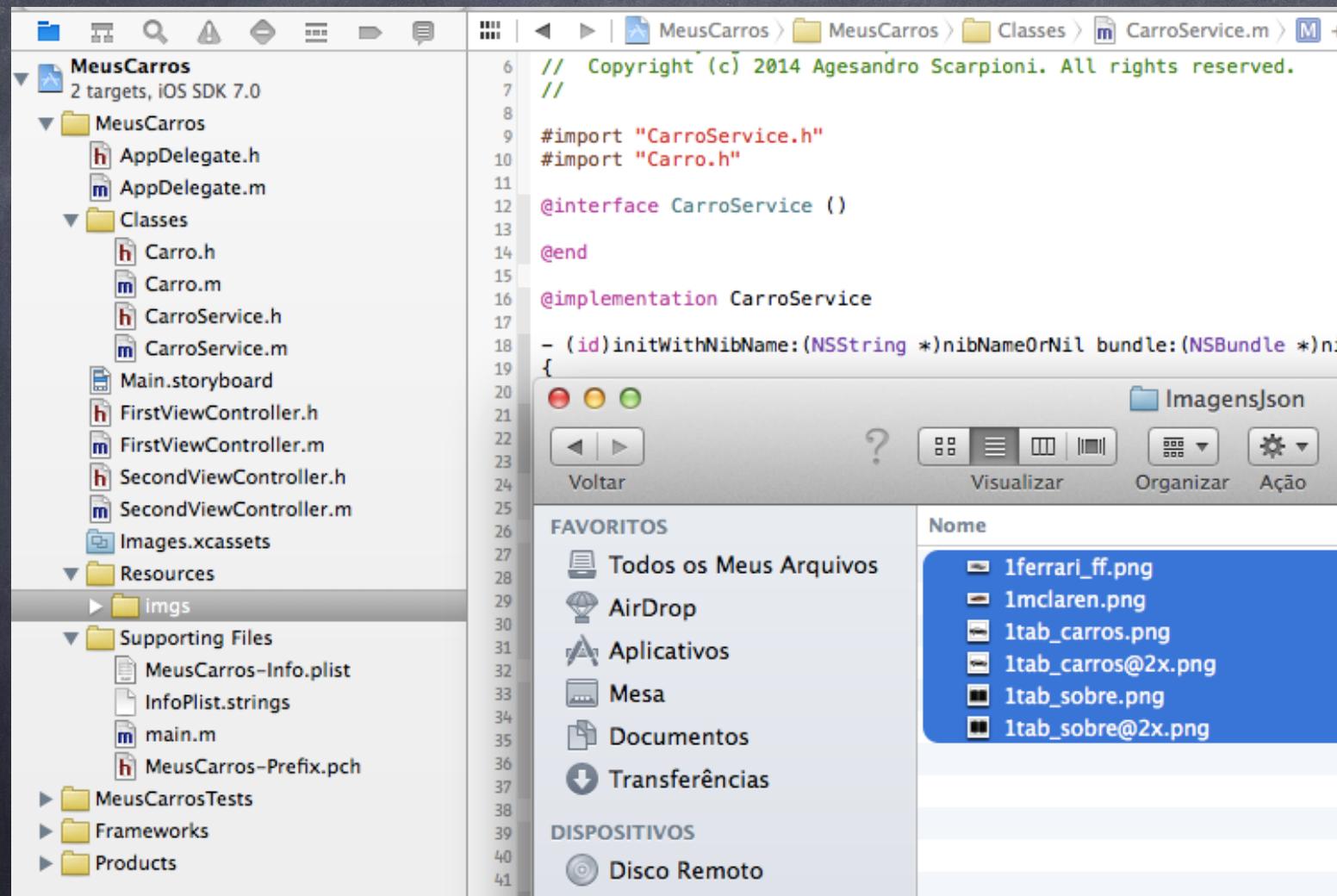
Mais uma Pasta

- Clique sobre o arquivo `images.xcassets` e com o botão direito escolha `New Group` para criar um grupo chamado `Resources` e dentro desse grupo inclua outro grupo chamado `imgs`.



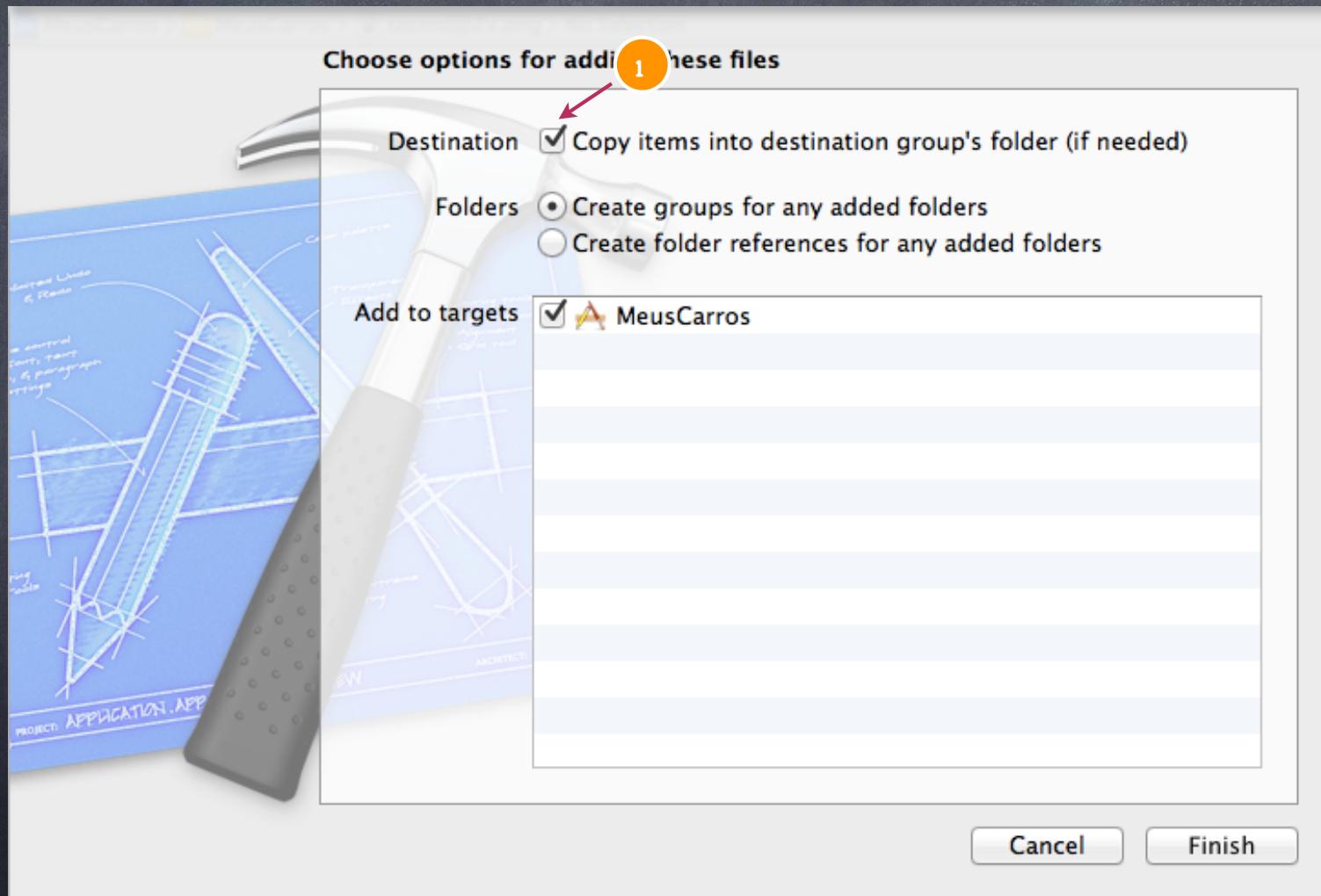
Carregando as Imagens

- Abra o Finder com as imagens do projeto e arraste para a pasta imgs no X-Code.



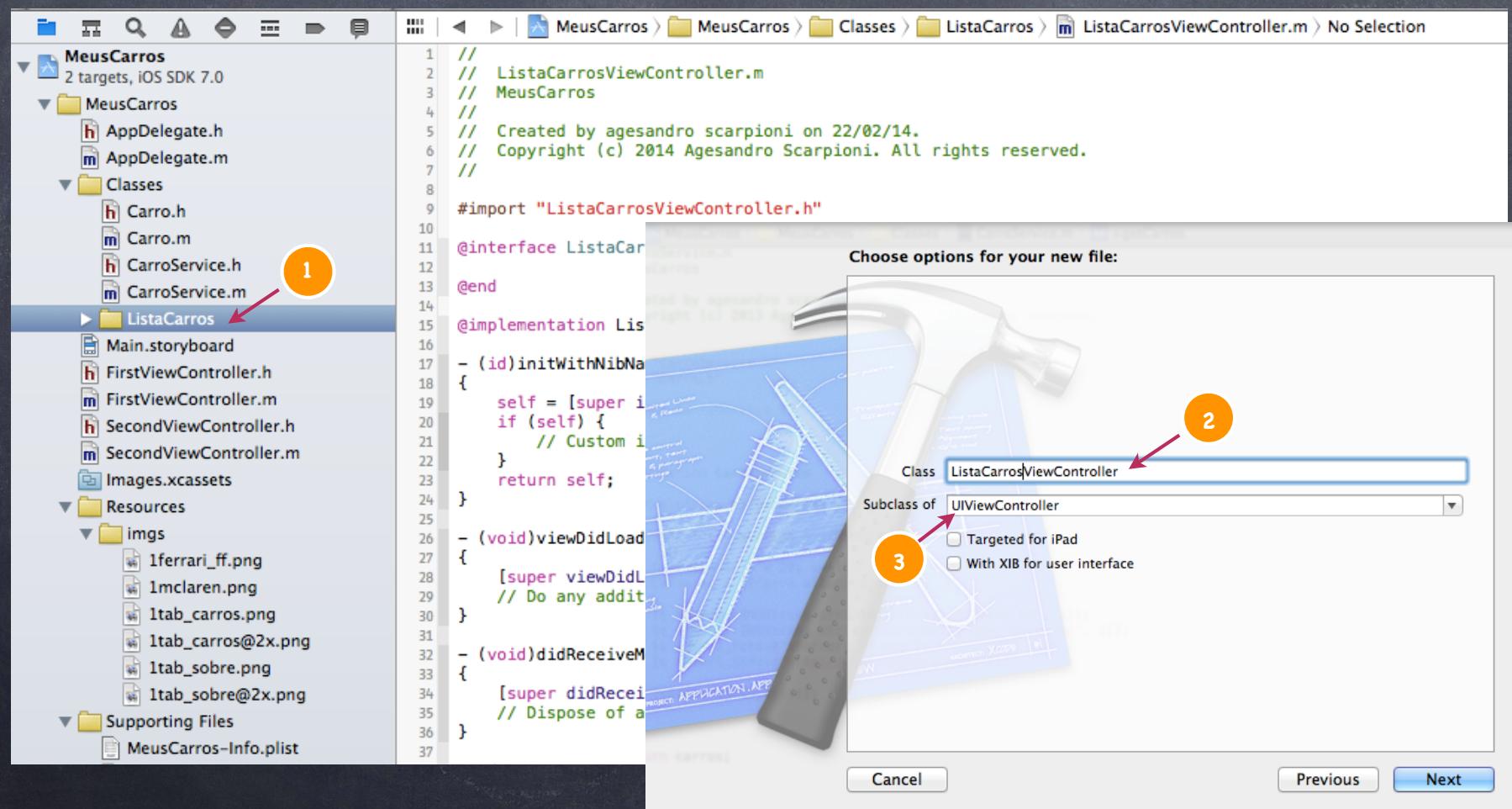
Carregando as Imagens

- Ao arrastar as imagens irá aparecer a tela abaixo, clique em Copy para que as imagens sejam copiadas para uma pasta do projeto, se o copy não for marcado teremos apenas um link das imagens.



Nova Pasta e nova Classe

- Clique em CarroService.m com o botão direito e escolha New Group, crie a pasta ListaCarros (1), dentro dessa pasta, crie uma classe (command + N) chamada ListaCarrosViewController (2), sub classe de UIViewController (3).



Protocolos do TableView

- Na classe `ListaCarrosViewController.h`, vamos implementar 2 protocolos do `TableView`, informe os protocolos datasource que cuida da exibição dos itens e delegate que é responsável pelo monitoramento dos eventos gerados na `TableView`, para declarar os protocolos use `< >` (1), ainda na classe e após as chaves crie um atributo do tipo `Array` mutável que represente o `get` e `set` do objeto carro(2).

The screenshot shows the Xcode interface with the project structure on the left and the code editor on the right.

Project Structure:

- MeusCarros (target, iOS SDK 6.1)
- MeusCarros (group)
 - AppDelegate.h
 - AppDelegate.m
 - Classes
 - Carro.h
 - Carro.m
 - CarroService.h
 - CarroService.m
 - ListaCarros
 - ListaCarrosViewController.h (selected)
 - ListaCarrosViewController.m
 - MainStoryboard.storyboard
 - FirstViewController.h
 - FirstViewController.m

Code Editor (ListaCarrosViewController.h):

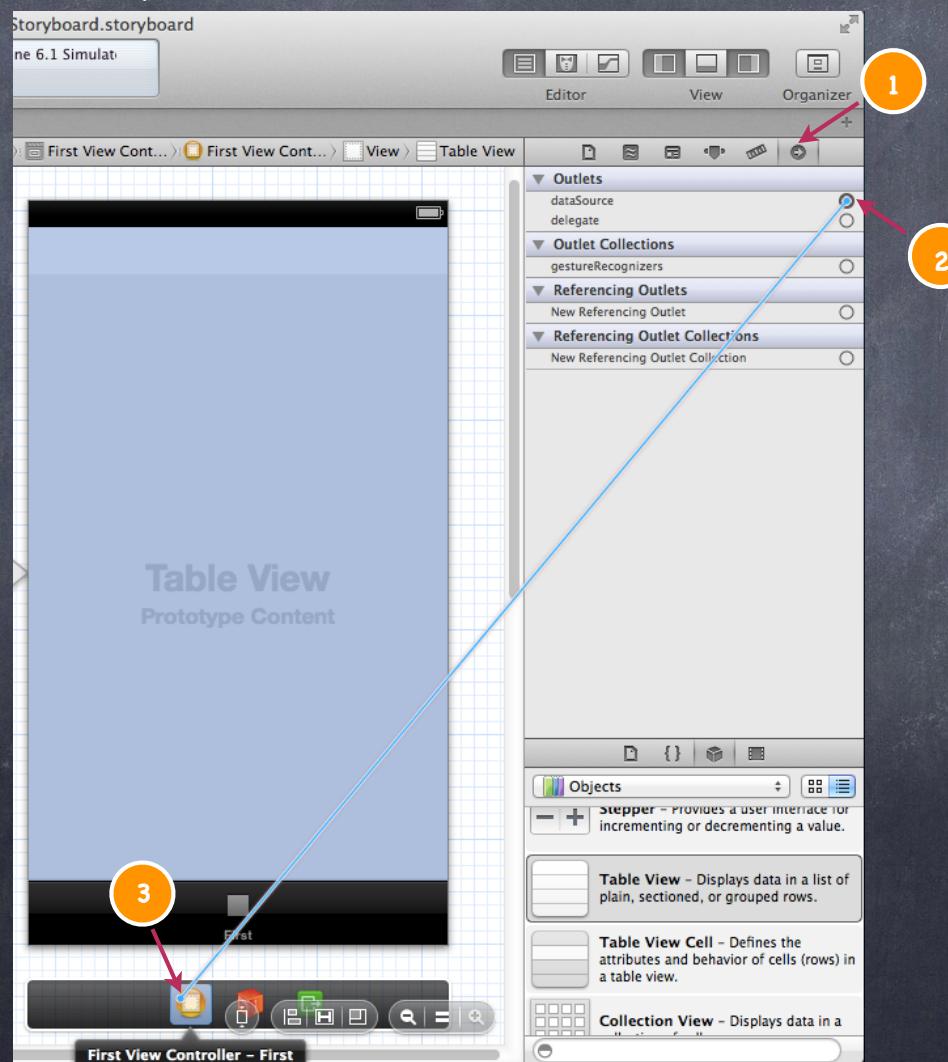
```
//  
//  ListaCarrosViewController.h  
//  MeusCarros  
//  
//  Created by agesandro scarpioni on 25/09/13.  
//  Copyright (c) 2013 Agesandro Scarpioni. All rights reserved.  
  
#import <UIKit/UIKit.h>  
  
@interface ListaCarrosViewController : UIViewController <UITableViewDataSource, UITableViewDelegate> {  
}  
  
// Criando o array para ser exibido no Tableview  
@property (nonatomic, retain) NSMutableArray *carros;  
  
@end
```

Annotations:

- A red circle with the number "1" has arrows pointing to the `<UITableViewDataSource, UITableViewDelegate>` part of the `@interface` line.
- A red circle with the number "2" has an arrow pointing to the `@property (nonatomic, retain) NSMutableArray *carros;` line.

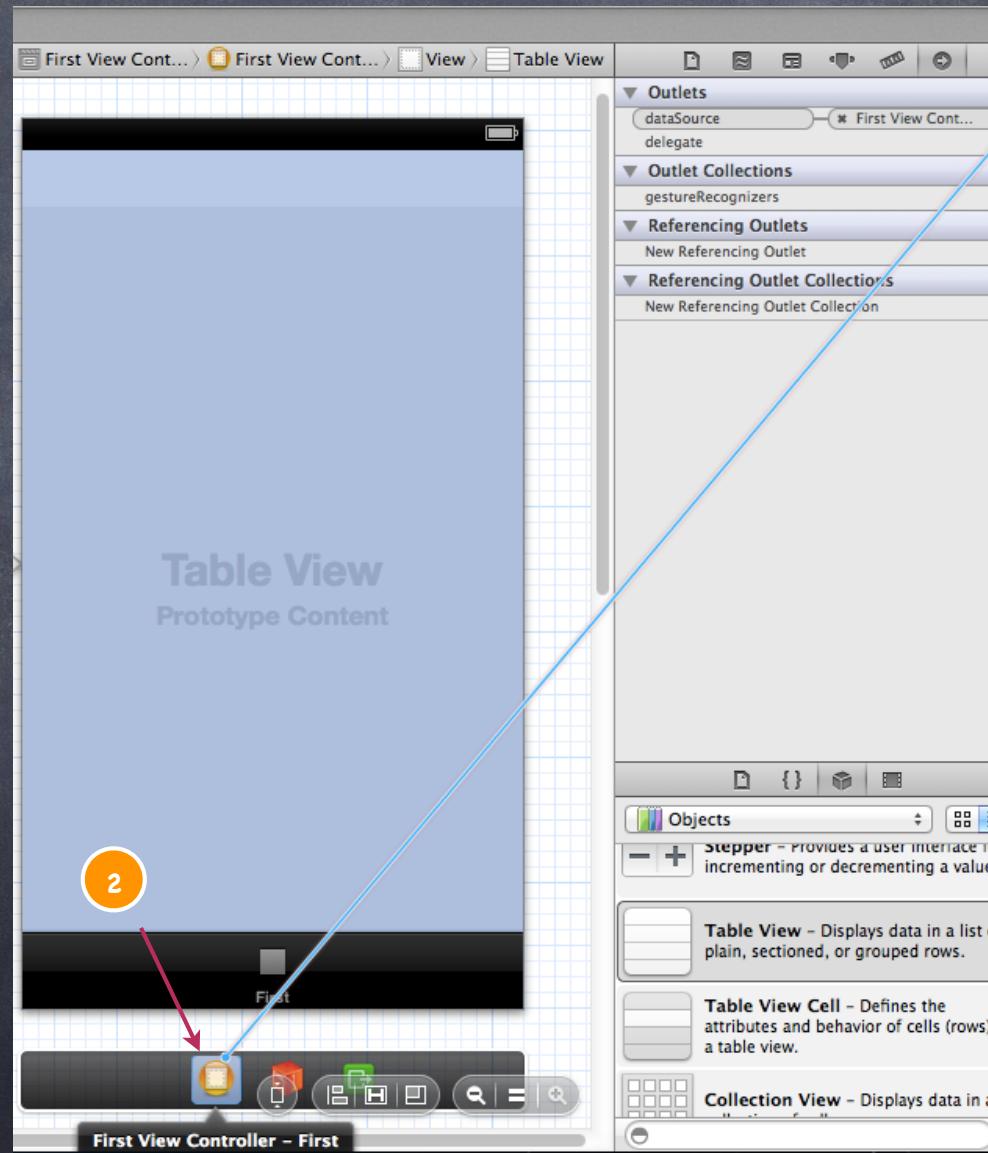
DataSource

- Precisamos ligar o Delegate e o DataSource para View Controller, selecione o Table View vá até o Connections Inspector(1), arraste o dataSource (2) até o ícone amarelo que representa a primeira View Controller (3).



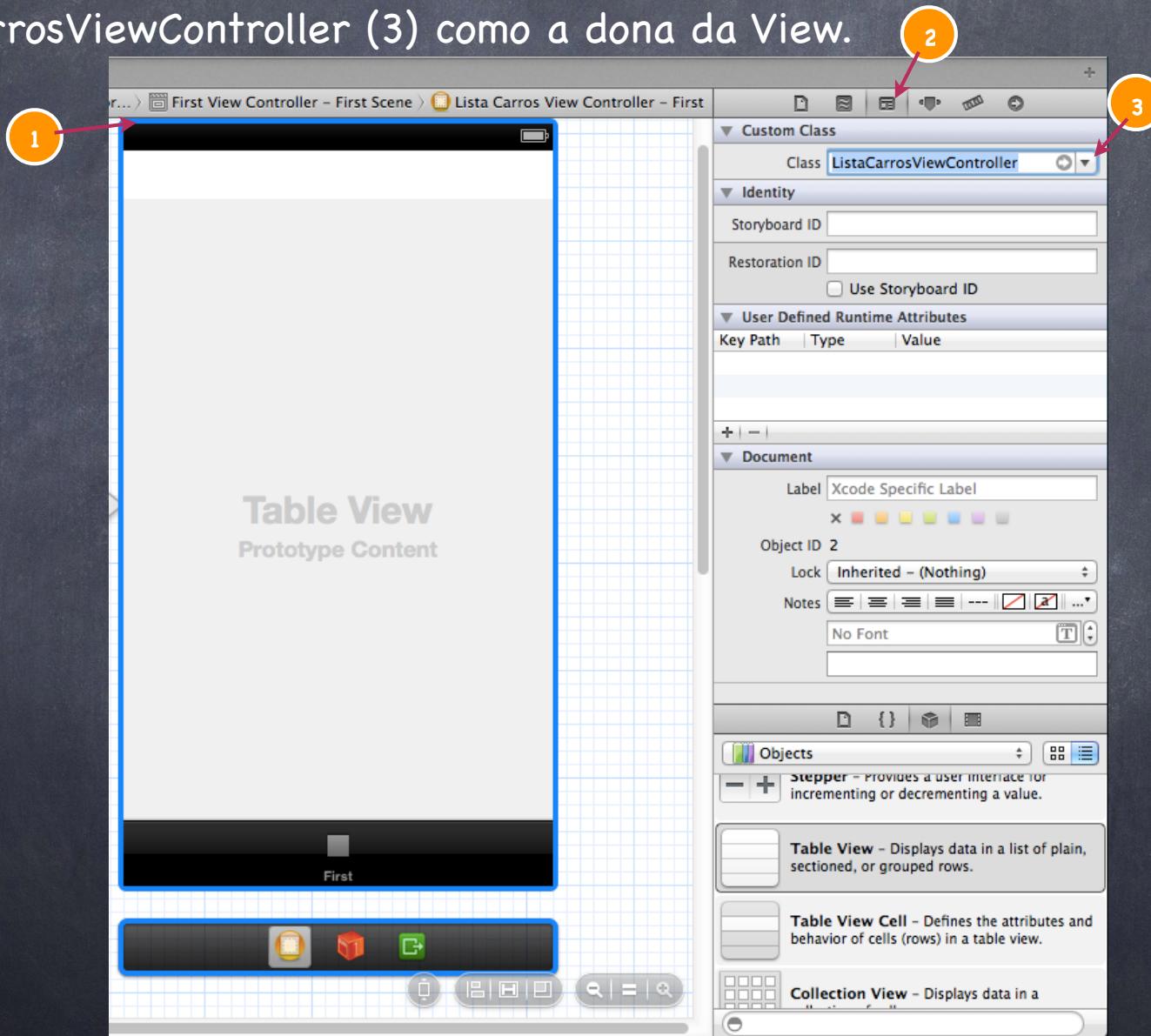
Delegate

- Execute os passos 1 e 2 para ligar o delegate ao DataSource



Declarando a “Dona” da View

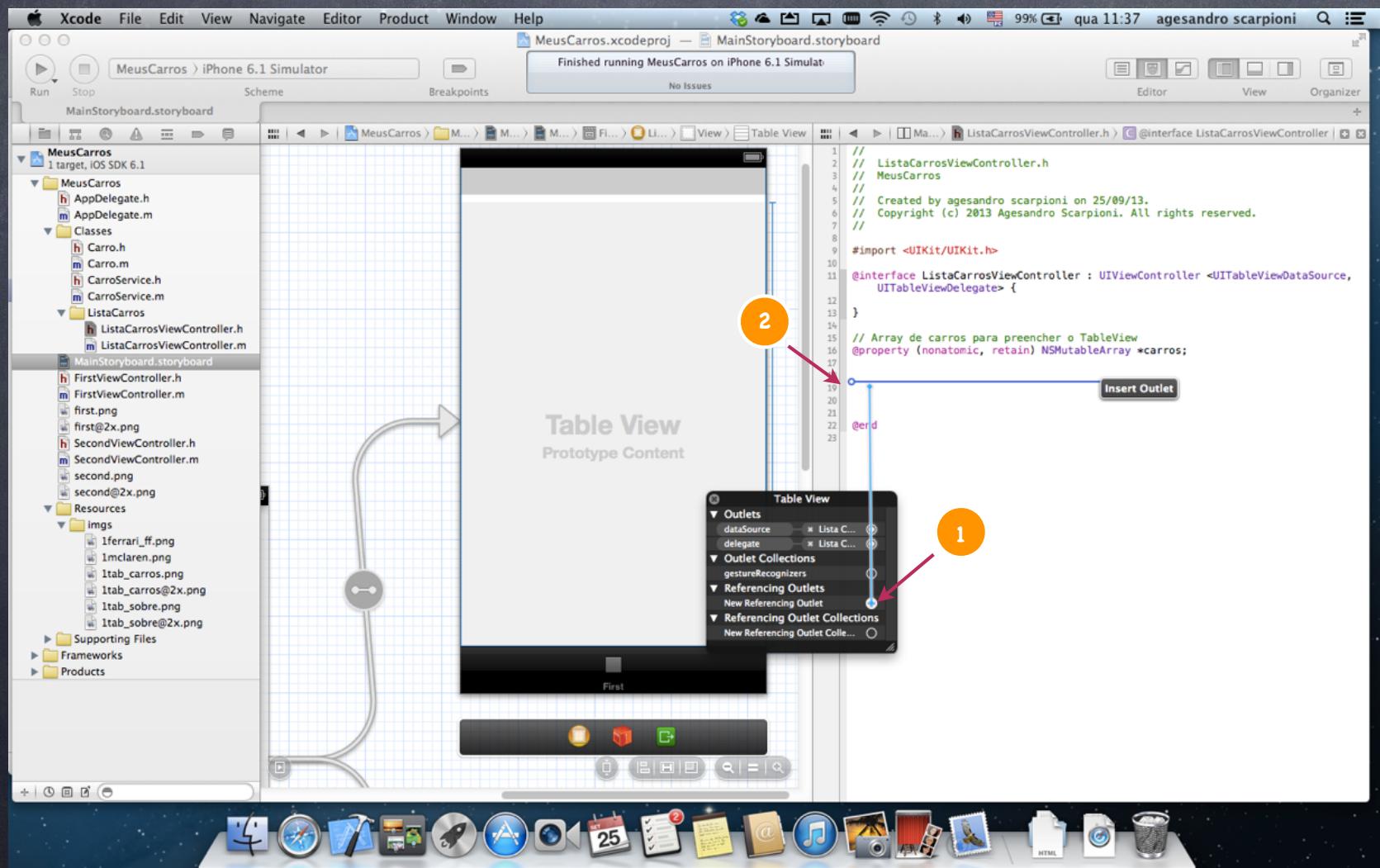
- Selecione a View (1), em identity inspector (2), identifique a classe ListaCarrosViewController (3) como a dona da View.



Declarando um atributo IBOulet

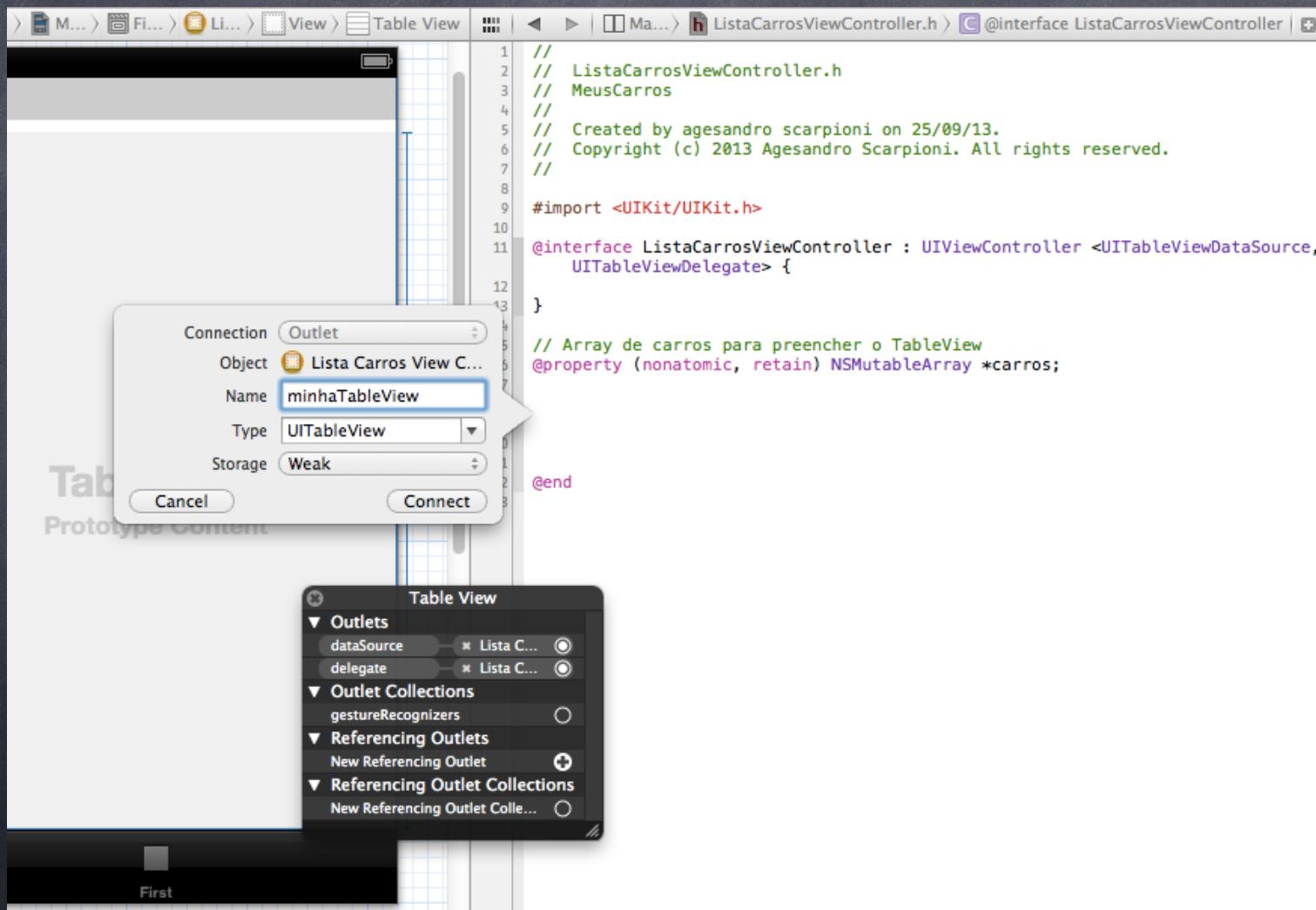
FIAP

- Com o botão direito sobre o TableView clique em New Referencing Outlet (1) e arraste até a área após as chaves (2).



Declarando um atributo FIA&P IBOutlet

- Ao soltar o botão irá aparecer um popup e informe o nome do Outlet como minhaTableView.



Declarando um atributo IBOulet

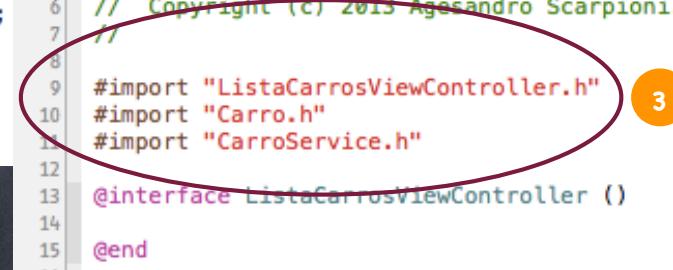
FIAP

- A linha 19 irá aparecer como um método get set (1) para a tableview, não esqueça de fazer o synthesize na ListaCarrosViewController.m (2), faça os imports (3)

```
1 // 1
2 // ListaCarrosViewController.h
3 // MeusCarros
4 //
5 // Created by agesandro scarpioni on 25/09/13.
6 // Copyright (c) 2013 Agesandro Scarpioni. All rights reserved.
7 //
8
9 #import <UIKit/UIKit.h>
10
11 @interface ListaCarrosViewController : UIViewController <UITableViewDataSource,
12     UITableViewDelegate> {
13 }
14
15 // Array de carros para preencher o TableView
16 @property (nonatomic, retain) NSMutableArray *carros;
17
18
19 @property (weak, nonatomic) IBOutlet UITableView *minhaTableView;
20
21
22 @end
23
24
```



```
1 // 3
2 // ListaCarrosViewController.m
3 // MeusCarros
4 //
5 // Created by agesandro scarpioni on 25/09/13.
6 // Copyright (c) 2013 Agesandro Scarpioni. All rights reserved.
7 //
8
9 #import "ListaCarrosViewController.h"
10 #import "Carro.h"
11 #import "CarroService.h"
12
13 @interface ListaCarrosViewController () 2
14
15 @end
16
17 @implementation ListaCarrosViewController
18
19 @synthesize carros;
20 @synthesize minhaTableView; 2
21
```



ViewDidLoad

- Vamos trocar o título do primeiro botão (1) e fazer com que a classe ListaCarrosViewController.m receba o array de carros (2).

```
1 //  
2 //  ListaCarrosViewController.m  
3 //  MeusCarros  
4 //  
5 //  Created by agesandro scarpioni on 25/09/13.  
6 //  Copyright (c) 2013 Agesandro Scarpioni. All rights reserved.  
7 //  
8  
9 #import "ListaCarrosViewController.h"  
10 #import "Carro.h"  
11 #import "CarroService.h"  
12  
13 @interface ListaCarrosViewController()  
14  
15 @end  
16  
17 @implementation ListaCarrosViewController  
18  
19 @synthesize carros;  
20 @synthesize minhaTableView;  
21  
22 |  
23 - (void)viewDidLoad  
24 {  
25     [super viewDidLoad];  
26     self.title = @"Carros"; // coloca nome "CARROS" no botão  
27     self.carros = [CarroService getCarros];  
28  
29     // Do any additional setup after loading the view.  
30 }  
31  
32 - (void)didReceiveMemoryWarning  
33 {  
34     [super didReceiveMemoryWarning];  
35     // Dispose of any resources that can be recreated.  
36 }  
37
```

1

2

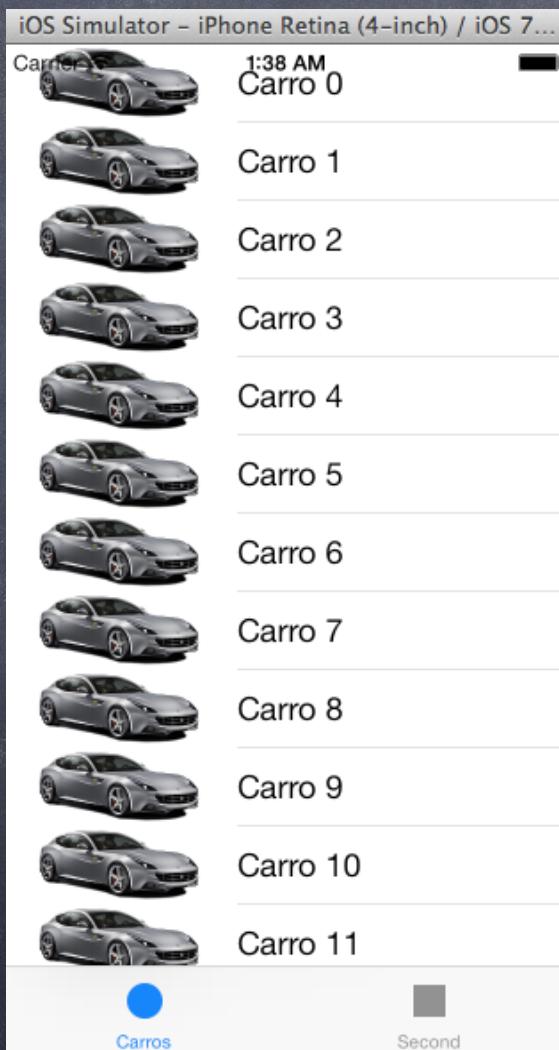
Implementando o Protocolo DataSource no ListaCarrosViewController.m

- Esta implementação já foi feita em outros programas de TableView, copie cole e altere as linhas 46, 62, 64, 65 para trabalhar com nosso objeto carro.

```
38 #pragma mark - tableview_protocolo_datasource
39
40 -(NSInteger) numberOfSectionsInTableView:(UITableView *)tableView{
41     return 1; // retorna o número de seções da tableview
42 }
43
44 -(NSInteger) tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section {
45     return [carros count]; // retorna a quantidade de linhas que terá o array de carros
46 }
47
48 -(UITableViewCell *) tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath {
49     //esta função é executada a quantidade de vezes informada no método acima
50     //e retorna uma célula que será o conteúdo do tableview
51
52     static NSString *CellIdentifier = @"Cell";
53
54     UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
55
56     if (cell==nil){
57         cell=[[UITableViewCell alloc]initWithStyle:UITableViewCellStyleDefault reuseIdentifier:CellIdentifier];
58     }
59
60
61
62     Carro *c = [carros objectAtIndex:indexPath.row];
63
64     cell.textLabel.text = c.nome;
65     cell.imageView.image = [UIImage imageNamed:c.url_foto];
66
67     return cell;
68 }
69 }
```

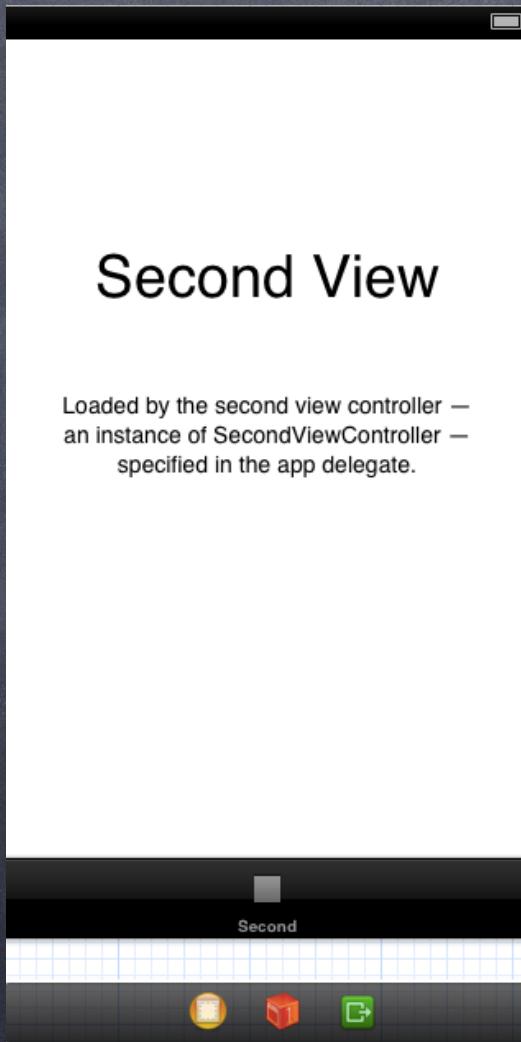
Resultado

- Execute seu programa, e observe a tela, veja que o botão 1 já possui outro nome.



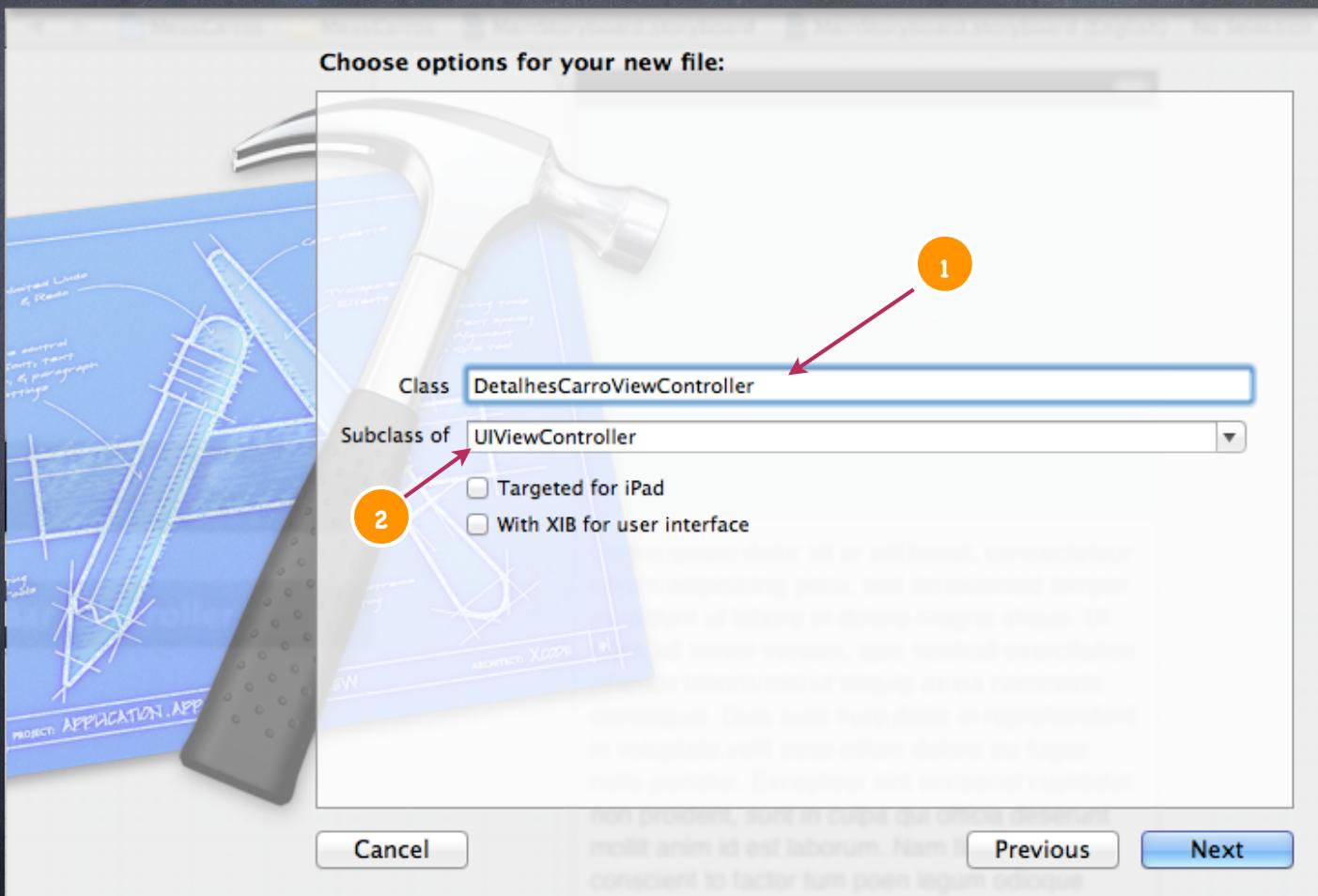
Segunda Tela do TabBar

- Na segunda tela do Tab Bar, apague os dois Labels.



Classe Detalhes

- Posicione o cursor em ListaCarrosViewController.m, dê um Command + N, crie uma nova Classe filha de UIViewController(2), para exibir os detalhes do carro chamada DetalhesCarroViewController(1).



3 Atributos no .h

- Faça o import da classe Carro.h(1), dentro das chaves crie um objeto carro do tipo Carro(2), e crie 3 propriedades como no exemplo abaixo, duas serão Outlet's.
- Optei por criar a codificação primeiro e depois a View.

```
1 //  
2 // DetalhesCarroViewController.h  
3 // MeusCarros  
4 //  
5 // Created by agesandro scarpioni on 29/09/13.  
6 // Copyright (c) 2013 Agesandro Scarpioni. All rights reserved.  
7 //  
8  
9 #import <UIKit/UIKit.h>  
10 #import "Carro.h" ← 1  
11  
12 @interface DetalhesCarroViewController : UIViewController {  
13     Carro *carro; ← 2  
14 }  
15  
16 @property (nonatomic, retain) Carro *carro;  
17 @property (nonatomic, retain) IBOutlet UIImageView *img;  
18 @property (nonatomic, retain) IBOutlet UITextView *tDesc;  
19 |  
20  
21 @end  
22
```

Obs: Depois ligaremos os 2 outlets aos objetos de uma View, outra forma de fazê-lo é criar a tela primeiro, clicar sobre o componente com o botão direito e arrastar até esta área, a vantagem é que já vem ligado.

Implementando o .m

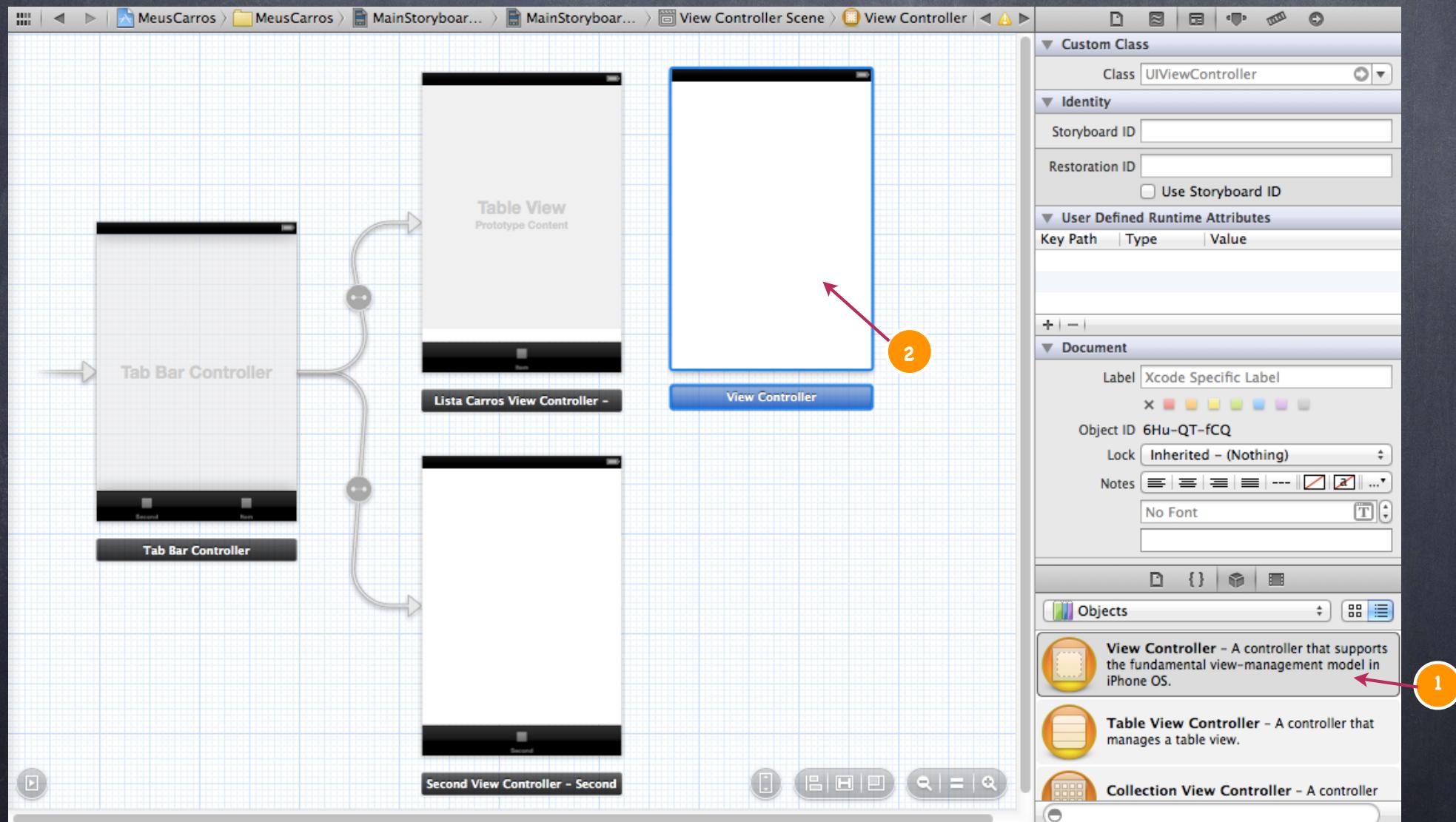
FIAP

- Faça o @synthesize para cada propriedade (1), e coloque a programação no DidLoad(2).

```
1 //  
2 // DetalhesCarroViewController.m  
3 // MeusCarros  
4 //  
5 // Created by agesandro scarpioni on 29/09/13.  
6 // Copyright (c) 2013 Agesandro Scarpioni. All rights reserved.  
7 //  
8  
9 #import "DetalhesCarroViewController.h"  
10  
11 @interface DetalhesCarroViewController ()  
12  
13 @end  
14  
15 @implementation DetalhesCarroViewController  
16  
17 @synthesize carro,img,tDesc; ← 1  
18  
19  
20 - (id)initWithNibName:(NSString *)NibNameOrNil bundle:(NSBundle *)nibBundleOrNil  
21 {  
22     self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];  
23     if (self) {  
24         // Custom initialization  
25     }  
26     return self;  
27 }  
28  
29 - (void)viewDidLoad  
30 {  
31     [super viewDidLoad];  
32  
33     self.title = carro.nome;  
34     self.tDesc.text = carro.desc;  
35     self.img.image = [UIImage imageNamed:carro.url_foto];  
36 }  
37  
38 - (void)didReceiveMemoryWarning  
39 {  
40     [super didReceiveMemoryWarning];  
41     // Dispose of any resources that can be recreated.  
42 }  
43  
44 @end  
45
```

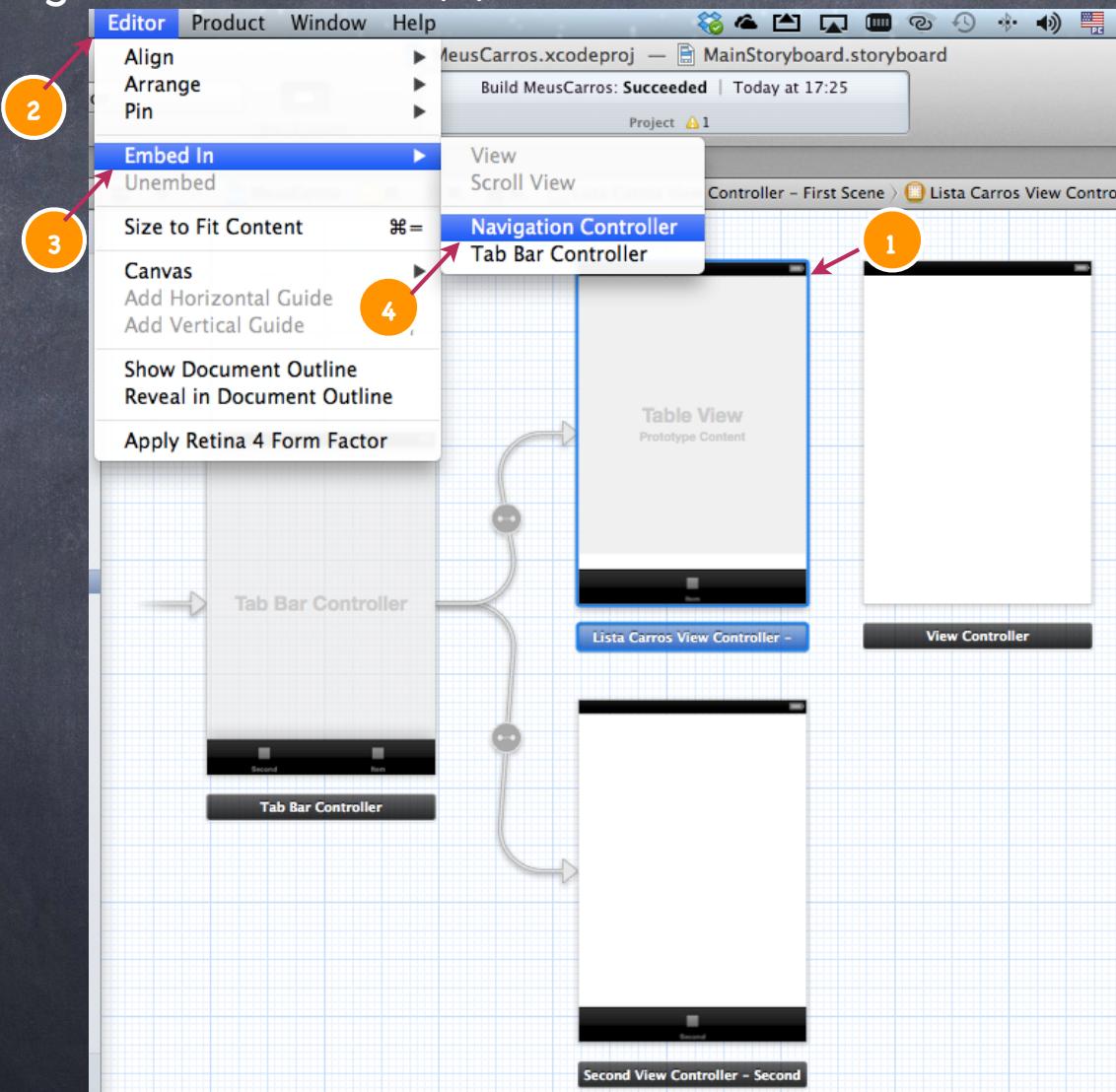
Mais uma View Controller

- Inclua uma View Controller em nossa StoryBoard.

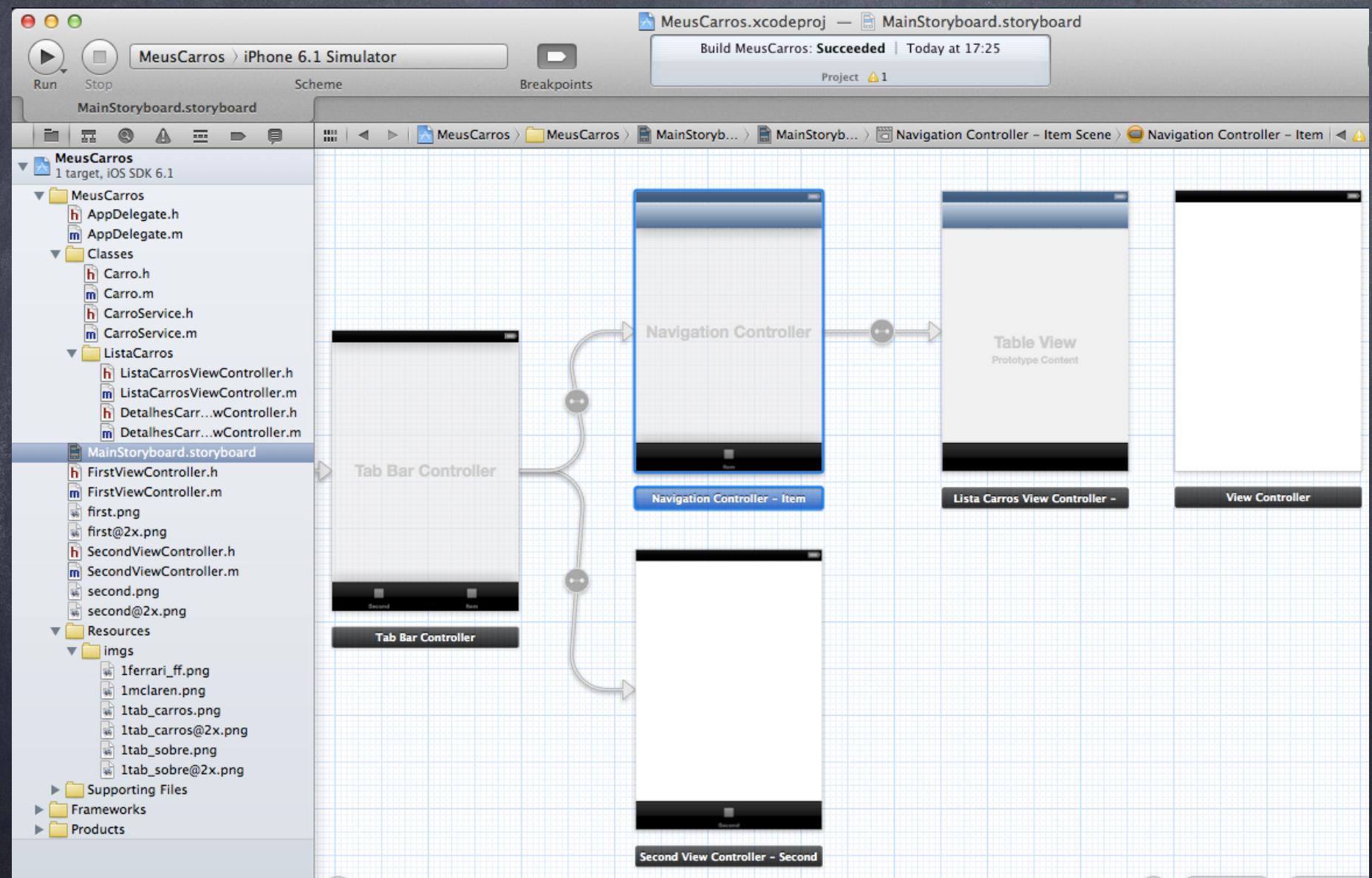


Incluindo um Navigation Controller

- Selecione a View Controller da Lista de Carros (1), clique em Editor (2), embed In (3) e selecione Navigation Controller (4).

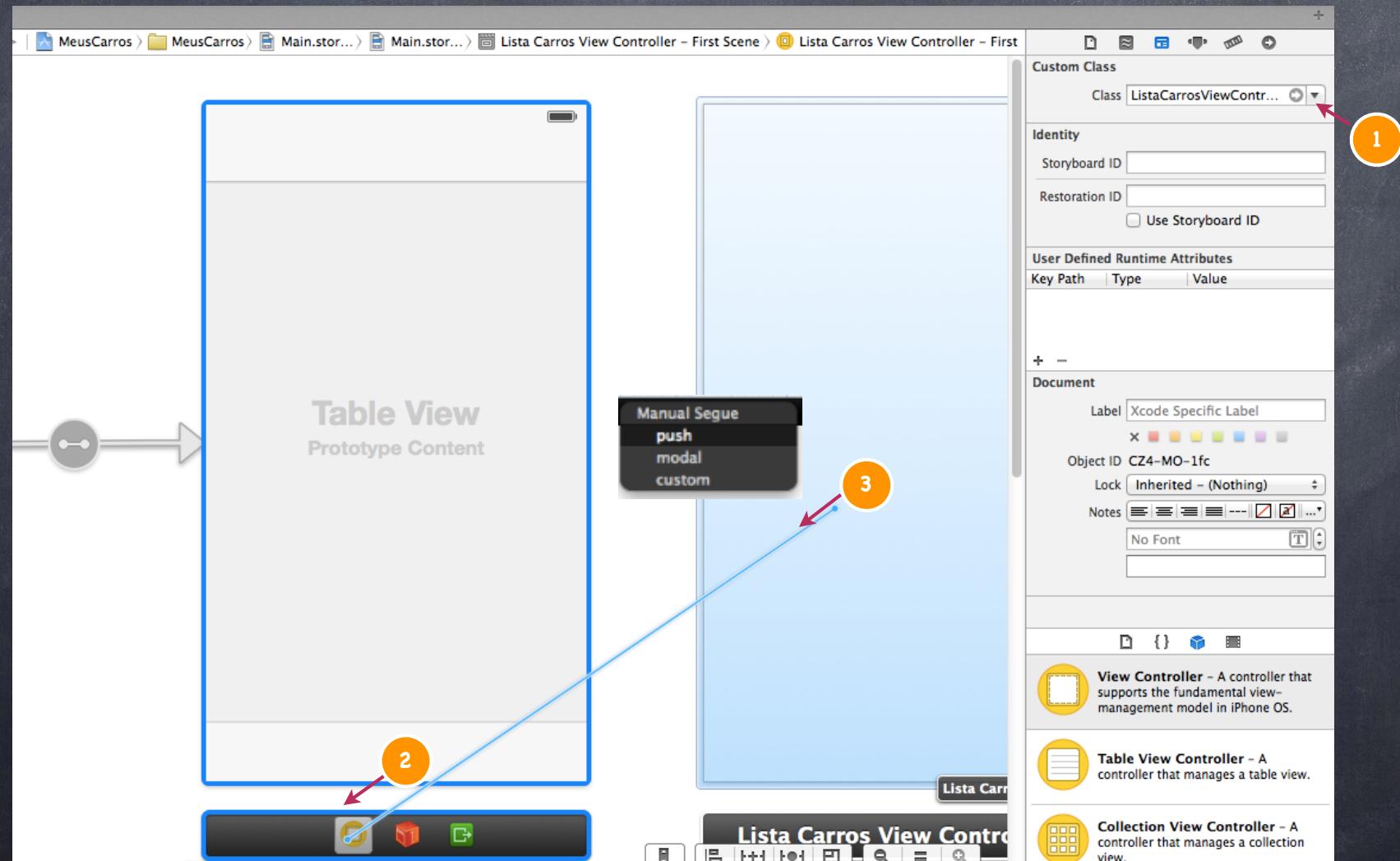


Resultado Final



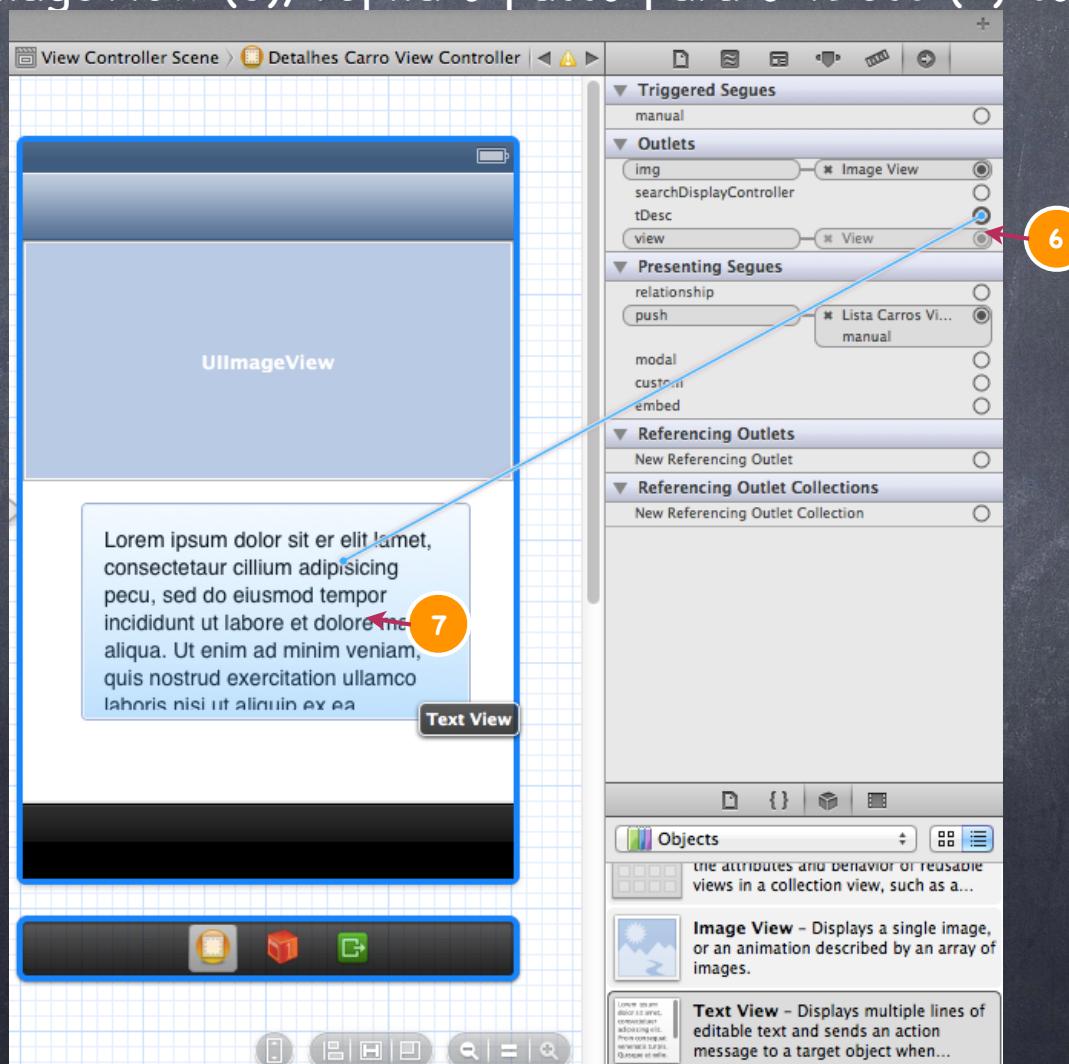
Classe dona da View e Push FIAP

- Selecione a View Controller e informe a classe relacionada a esta view (1), depois clique com botão direito sobre a view que possui a tableView (2) e arraste até a viewController (3), escolha Push.



Adicione Image View e Text View FIA&P

- Agora que já ligamos via Push as duas telas, inclua um componente Image View (1) e um Text View (2), após esta etapa selecione a ViewController (3) e faça a ligação do Outlet img (4) ao UIImageView (5), repita o passo para o tDesc (6) com TextView (7).



Implementando ListaCarro .m

FIAP

- Agora vamos implementar a classe ListaCarrosViewController.m, vá até o topo da classe e inclua o #import de DetalhesCarroViewController.h.

```
1 //  
2 //  ListaCarrosViewController.m  
3 //  MeusCarros  
4 //  
5 //  Created by agesandro scarpioni on 25/09/13.  
6 //  Copyright (c) 2013 Agesandro Scarpioni. All rights reserved.  
7 //  
8  
9 #import "ListaCarrosViewController.h"  
10 #import "Carro.h"  
11 #import "CarroService.h"  
12 #import "DetalhesCarroViewController.h"  
13  
14
```

Implementando Delegate

- Ainda na ListaCarrosViewController.m, implemente o método didSelectRowAtIndexPath do protocolo delegate como é mostrado abaixo, estas linhas servem para quando o usuário clicar em um item, passar os dados do carro para a view DetalhesCarroViewController.

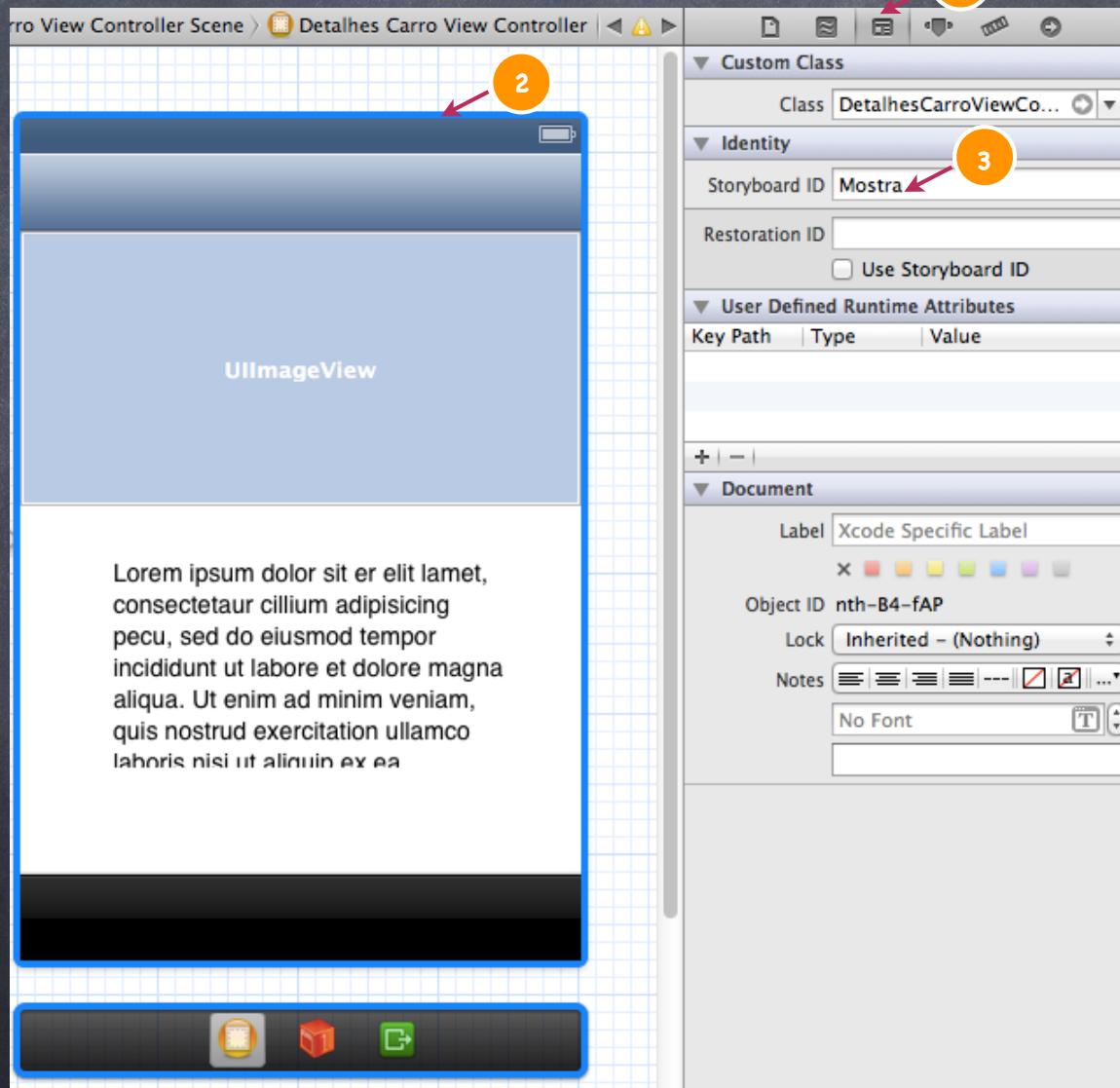
```
71 -(void) tableView:(UITableView *)tableView didSelectRowAtIndexPath:(NSIndexPath *)indexPath{  
72  
73     Carro *c = [carros objectAtIndex:indexPath.row];  
74     DetalhesCarroViewController *detalhes = [self.storyboard instantiateViewControllerWithIdentifier:@"Mostra"];  
75     detalhes.carro = c;  
76     [self.navigationController pushViewController:detalhes animated:YES];  
77 }  
78  
79 }
```

Obs: Falta identificar a view que contém o textView e o imageView com o nome de Mostra.

Identificando um Storyboard

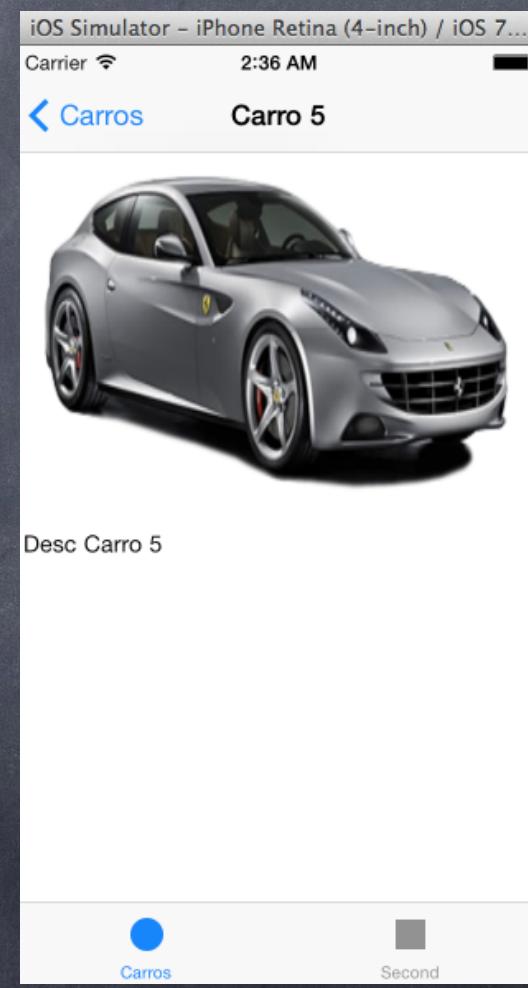
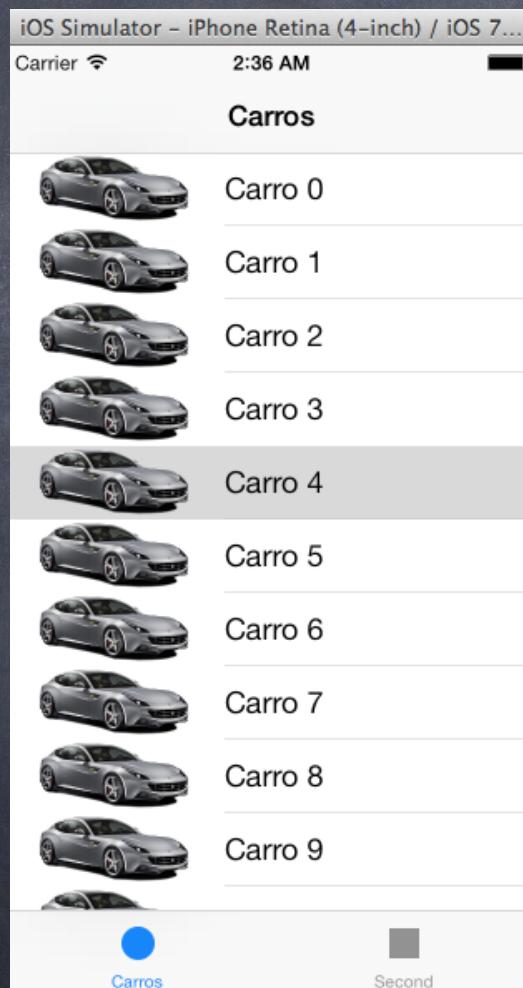
FIAP

- No Identity Inspector (1), selecione a View (2) e identifique o Storyboard com o nome Mostra (3).



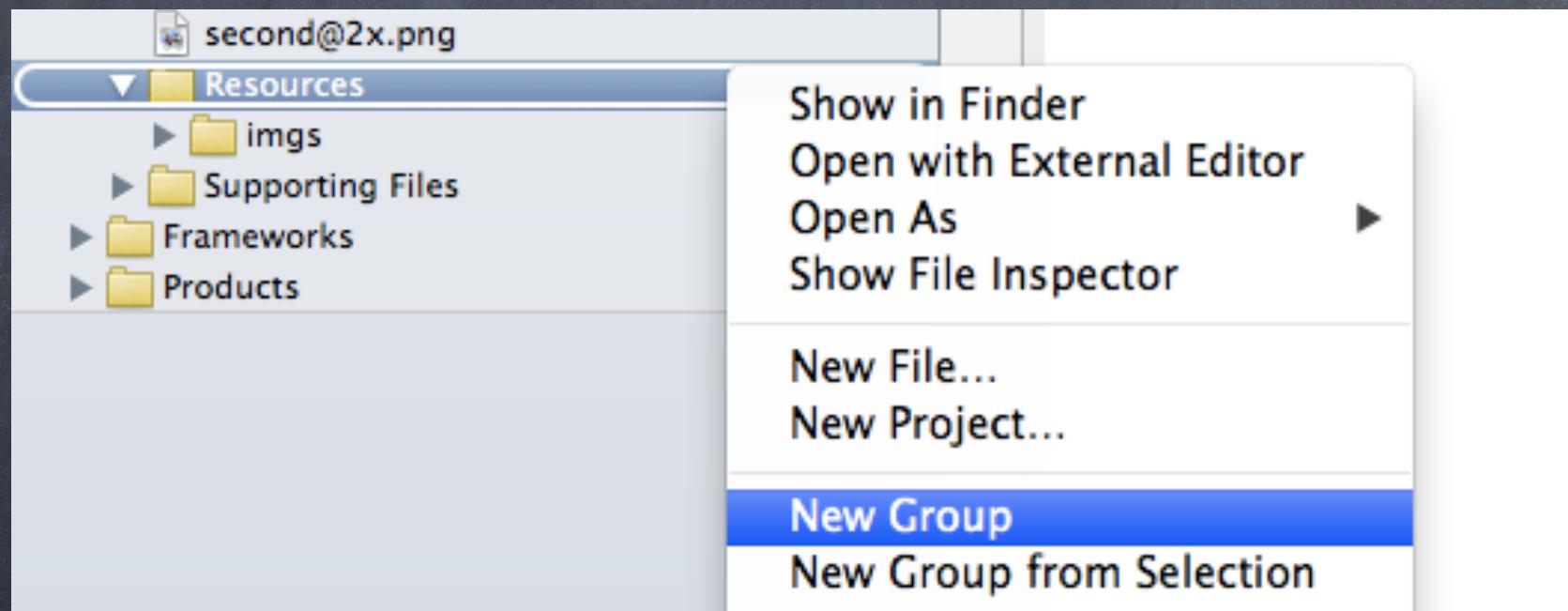
Execute

- Command+R para executar e observe o resultado.



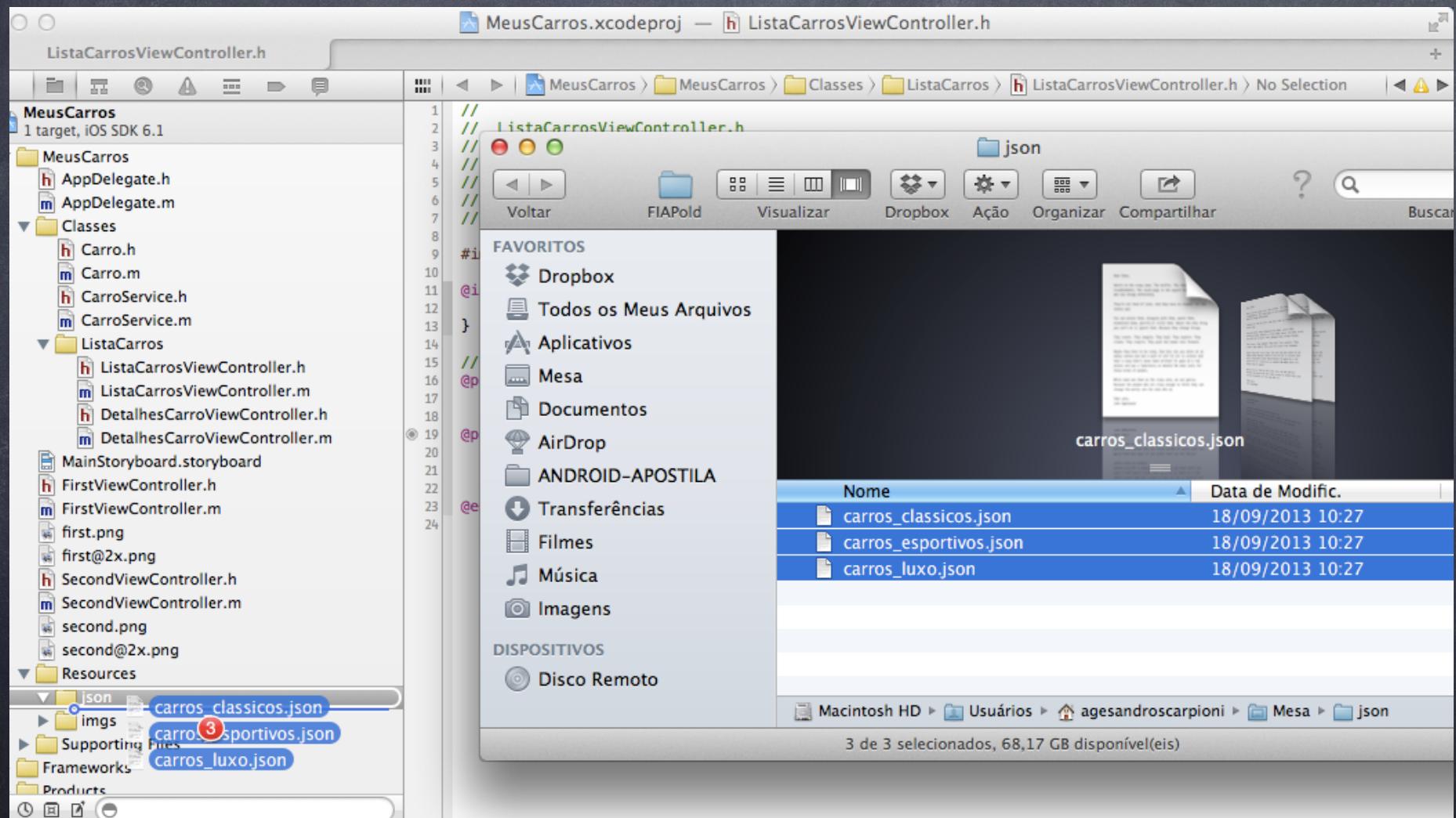
Json

- Crie uma pasta dentro de Resources (botão direito New Group) para guardarmos os arquivos Json, que inicialmente usaremos off line e depois os atualizaremos via Web Service.



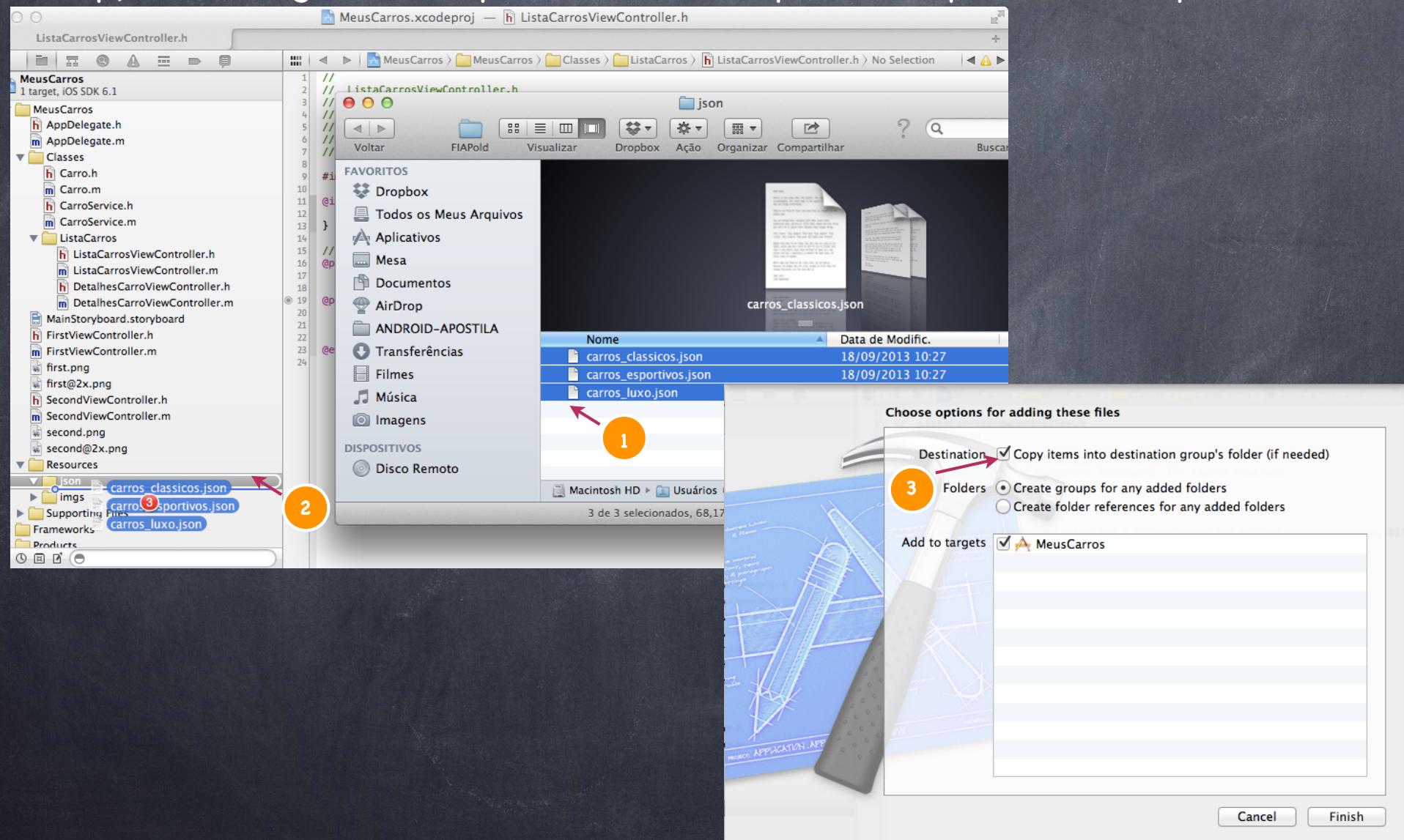
Json

- Abra o Finder e arraste os arquivos para a pasta json.



Json

- Abra o Finder (1) e arraste os arquivos para a pasta json (2), lembre-se de marcar copy (3) na segunda tela para criar uma cópia dos arquivos em seu projeto.



Json

- Abra a classe CarroService.h e vamos declarar dois novos métodos estáticos, getCarrosByTipo (1) e parserJSON (2).

```
1 //  
2 // CarroService.h  
3 // MeusCarros  
4 //  
5 // Created by agesandro scarpioni on 25/09/13.  
6 // Copyright (c) 2013 Agesandro Scarpioni. All rights reserved.  
7 //  
8  
9 #import <UIKit/UIKit.h>  
10  
11 @interface CarroService : UIViewController {  
12 }  
13  
14 + (NSMutableArray *) getCarros;  
15  
16 // Busca por tipo os carros, em principio vamos procurar  
17 // apenas os carros do tipo esporte.|  
18 +(NSMutableArray *) geCarrosByTipo:(NSString *) tipo; 1  
19  
20 // Parser de JSON  
21 +(NSMutableArray *) parserJSON:(NSData *)data; 2  
22  
23  
24 @end  
25
```

Json e Parser

- Parser é um programa ou componente, que serve para analisar a estrutura gramatical de uma entrada, manipulando os tokens, que são segmentos de texto ou símbolos que podem ser manipulados. Em XML ou Json o parser é o leitor que ajuda na conversão do arquivo para manipulação dos dados contidos no mesmo. A classe que faz isso para Json no OBJ-C é a NSJSONSerialization disponível a partir do IOS 5. Implemente o método abaixo na classe CarroService.m.

```
35
36 //faz o parse do JSON
37 +(NSMutableArray *) parserJSON:(NSData *)data{
38     if(!data || [data length]==0){
39         NSLog(@"Nenhum dado encontrado");
40         return nil;
41     }
42     NSMutableArray *carros = [[NSMutableArray alloc] init];
43
44     NSError* error;
45     //carrega o NSData em um dicionário
46     NSDictionary* json=[NSJSONSerialization JSONObjectWithData:data options:kNilOptions error:&error];
47     //dicionário para todos os carros
48     NSArray *jsonCarros = [[json objectForKey:@"carros"] objectForKey:@"carro"];
49     for (NSDictionary *dictCarro in jsonCarros){
50         Carro *c=[[Carro alloc]init];
51         c.nome = [dictCarro objectForKey:@"nome"];
52         c.desc = [dictCarro objectForKey:@"desc"];
53         c.url_foto = [dictCarro objectForKey:@"url_foto"];
54         c.url_info = [dictCarro objectForKey:@"url_info"];
55
56         [carros addObject:c];
57     }
58     return carros;
59 }
60 }
```

Leitura do Arquivo Json

FIAP

- O método abaixo deve ser implementado na classe CarroService.m, ele vai montar uma string com o nome carros_ + (esportivo ou classicos ou luxo) chegando via parâmetro, vai montar o caminho do arquivo json local e fazer a leitura do arquivo para um NSData, em seguida o NSData será convertido em um array de carros após passar pelo método parserJSON.

```
60
61
62 +(NSMutableArray *) getCarrosByTipo:(NSString *) tipo{
63     //cria um nome do arquivo conforme o tipo classico, esportivo ou luxo;
64     NSString *nomeArquivo= [[NSString alloc] initWithFormat:@"carros_%@",tipo];
65     //le o arquivo json;
66     NSString *caminho=[[NSBundle mainBundle] pathForResource:nomeArquivo ofType:@"json"];
67     //faz a leitura do arquivo local do projeto e retorna o NSData
68     NSData *data = [[NSData alloc] initWithContentsOfFile:caminho];
69     if (data.length ==0){
70         NSLog(@"arquivo vazio");
71         return nil;
72     }
73     //chama o método que converte o NSData em array de carros
74     NSMutableArray *carros=[self parserJSON:data];
75     return carros;
76 }
77 }
```

DetalhesCarroViewController.m

- Nessa classe DetalhesCarroViewController.m vá até o viewDidLoad, comente a linha onde a imagem do carro é carregada por um arquivo fixo (1) e implementa as 3 linhas onde a imagem é carregada via URL(2).

```
20 - (void)viewDidLoad
21 {
22     [super viewDidLoad];
23
24     self.title = carro.nome;
25     self.tDesc.text = carro.desc;
26     // self.img.image = [UIImage imageNamed:carro.url_foto];
27
28     // Foto de um caminho na internet
29     NSData *data = [NSData dataWithContentsOfURL:[NSURL URLWithString:carro.url_foto]];
30     UIImage *image = [[UIImage alloc] initWithData:data];
31     self.img.image = image;
32
33 }
34 }
```



ListaCarrosViewController.m

FIAP

- Nessa classe vá até o metodo cellForRowAtIndexPath, comente a linha onde a imagem do carro é carregada por um arquivo fixo (1) e implemente as 3 linhas onde a imagem é carregada via URL(2).

```
51 -(UITableViewCell *) tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath {  
52     //esta função é executada a quantidade de vezes informada no método acima  
53     //e retorna uma célula que será o conteúdo do tableview  
54  
55     static NSString *CellIdentifier = @"Cell";  
56  
57     UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];  
58  
59     if (cell==nil){  
60         cell=[[UITableViewCell alloc] initWithStyle:UITableViewCellStyleDefault reuseIdentifier:CellIdentifier];  
61     }  
62  
63  
64  
65     Carro *c = [carros objectAtIndex:indexPath.row];  
66  
67     cell.textLabel.text = c.nome;  
68     //cell.imageView.image = [UIImage imageNamed:c.url_foto];  
69  
70     // 1  
71     // Foto da internet  
72     NSData *data = [NSData dataWithContentsOfURL:[NSURL URLWithString:c.url_foto]];  
73     UIImage *image = [[UIImage alloc] initWithData:data];  
74     cell.imageView.image = image;  
75  
76     return cell;  
77 }
```

1

2

ListaCarrosViewController.m FIAP

- Na classe ListaCarrosViewController.m vá até o metodo viewDidLoad, comente a linha onde buscamos o carro de um array fixo (1) e coloque a linha que o carro é carregado por um método que recebe o tipo de carro e este mesmo método acessa o Json local(2).

```
24
25 - (void)viewDidLoad
26 {
27     [super viewDidLoad];
28     self.title = @"Carros"; // coloca nome "CARROS" no botão
29 // self.carros = [CarroService getCarros];
30     self.carros = [CarroService geCarrosByTipo:@"esportivos"];
31     // Do any additional setup after loading the view.
32 }
33
34 }
```



Resultado

Command + R

