

Rest e Json no iOS

X-Code com Objective C
Prof. Agesandro Scarpioni

App's com label's, image's e button's

- Criar dois aplicativos para que seja feito um request, depois um parser do Json em um dicionário para exibir os dados.

O que é Rest

- REST, abreviação de "REpresentational State Transfer" (Transferência de Estado Representativo) é uma técnica de engenharia de Software para sistemas hipermídia distribuídos como World Wide Web. REST é um estilo de se projetar aplicativos da Web fracamente acoplados que contam com recursos nomeados em forma de Localizador Uniforme de Recursos (URL), Identificador Uniforme de Recursos (URI) e Nome de Recurso Uniforme (URN), e não com mensagens. Engenhosamente, o REST transporta a infra estrutura já validada e bem sucedida da Web, HTTP, ou seja, o REST alavanca aspectos do protocolo HTTP como pedidos GET e POST. Esses pedidos são perfeitamente mapeados para necessidades de aplicativo de negócios padrão, como create read, update, and delete (CRUD).
- Na arquitetura REST os clientes (páginas web, dispositivos iOS, Android) efetuam requisições (request) através do protocolo HTTP e recebem respostas (response) dos servidores. Dessa forma é feita a transmissão de dados.
- Utilizar um Webservice é uma das maneiras mais comuns de se integrar aplicações diferentes. Existem diferentes tipos de arquiteturas para web services, e o RESTful é mais simples em comparação aos outros web services, que geralmente utilizam o SOAP.

Dica: Saiba mais sobre Rest em: <http://www.ibm.com/developerworks/br/library/j-rest/>

O que é Rest

- Dada essa simplicidade, isso faz com que a arquitetura RESTful seja uma escolha popular principalmente para serviços abertos ao público. Por exemplo, o Twitter possui uma API Restful. Além do Twitter, o Flickr também possui uma API que segue os princípios da arquitetura REST. De certa forma é uma tendência que os serviços conhecidos como a “Web 2.0” disponibilizem uma API (geralmente REST), pois é cada vez maior a necessidade que esses serviços sejam integrados com diversos tipos aplicações.

Dica: Saiba mais sobre Rest em: <http://imasters.com.br/desenvolvimento/definicao-restricoes-e-beneficios-modelo-de-arquitetura-rest/>

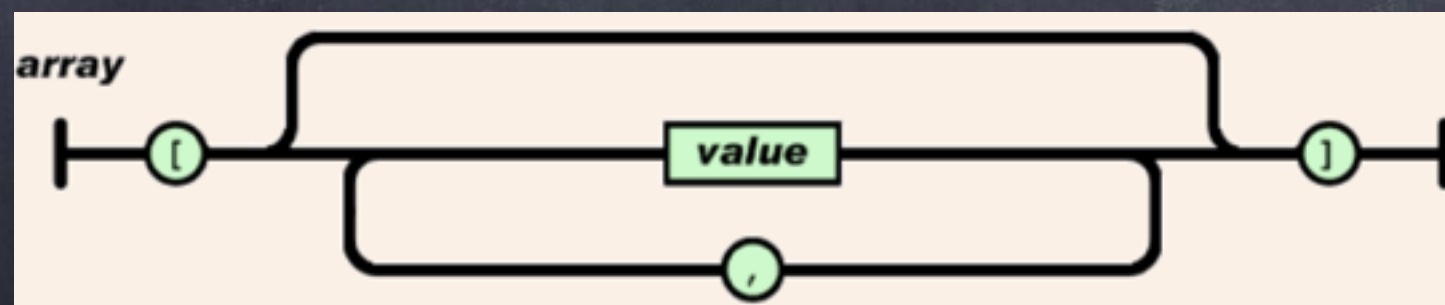
O que é JSON

- **JSON** É um acrônimo para **JavaScript Object Notation**, é um formato leve para intercâmbio de dados computacionais, é um subconjunto da notação de objeto de JavaScript, mas seu uso não requer JavaScript exclusivamente. A simplicidade de JSON tem resultado em seu uso difundido, especialmente como uma alternativa para XML em AJAX. Uma das vantagens reivindicadas de **JSON** sobre XML como um formato para intercâmbio de dados, é o fato de ser muito mais fácil escrever um analisador JSON. Na prática esses arquivos Json são retornados de um Web Service. Veja abaixo um exemplo de um objeto JSON.

```
{ "Alunos" : [  
  { "nome": "João", "notas": [ 8, 9, 7 ] },  
  { "nome": "Maria", "notas": [ 8, 10, 7 ] },  
  { "nome": "Pedro", "notas": [ 10, 10, 9 ] }  
]
```

The image shows a JSON object with a key "Alunos" and a value that is an array of three objects. Each object has a "nome" key and a "notas" key. The "notas" values are arrays of three numbers. Annotations include an orange oval labeled "Array" pointing to the outer array, and another orange oval labeled "Array" pointing to the inner arrays of numbers.

Um array Json é uma coleção de valores ordenados. O array começa com [e termina com], os valores são separados por vírgula, veja a representação gráfica abaixo:



Dica: Saiba mais sobre JSON em: <http://json.org/json-pt.html>

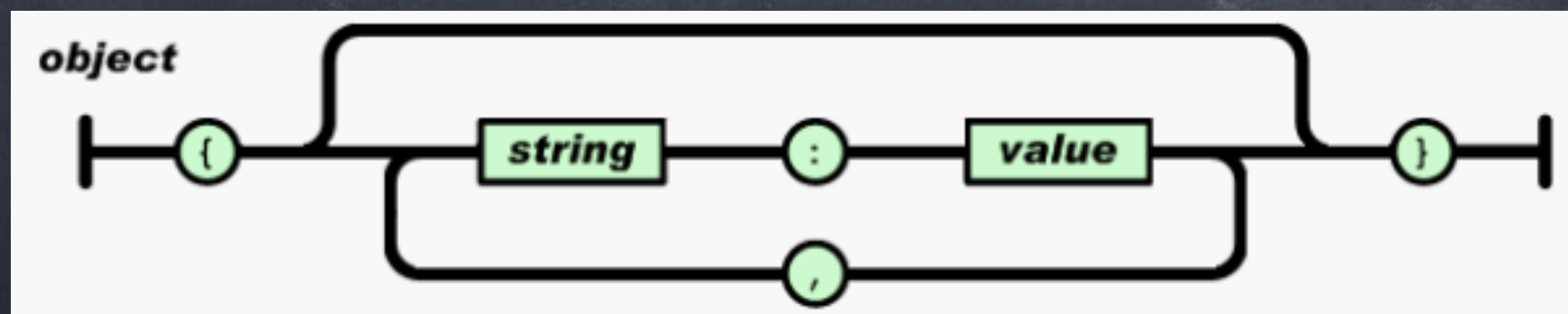
O que é JSON

- JSON é um formato de intercâmbio de dados leve. É fácil para os seres humanos a ler e escrever. É fácil para máquinas para analisar e gerar. É baseado em um subconjunto da linguagem de programação JavaScript. JSON é um formato de texto que é completamente independente do idioma, mas usa convenções que são familiares aos programadores das linguagens da família C, incluindo C, C++, C#, Java, JavaScript, Perl, Python, e muitos outros. Estas propriedades fazem JSON uma linguagem de intercâmbio de dados ideal. Quando você observa um JSON, ele vai ser muito semelhante a um outro exemplo abaixo:

```
{  
  "id": 123,  
  "name": "Jordan G",  
  "age": 17  
}
```

Um objeto JSON é um conjunto não ordenado de pares nome / valor. Um objeto começa com a chave esquerda { e termina com a chave direita }. Cada nome é seguido por dois pontos: e os pares nome / valor são separados por uma vírgula.

Abaixo está uma representação gráfica da forma JSON acima



Dica: Saiba mais sobre JSON em: <http://www.ios-blog.co.uk/tutorials/swift/parse-json-deserialization/#references>

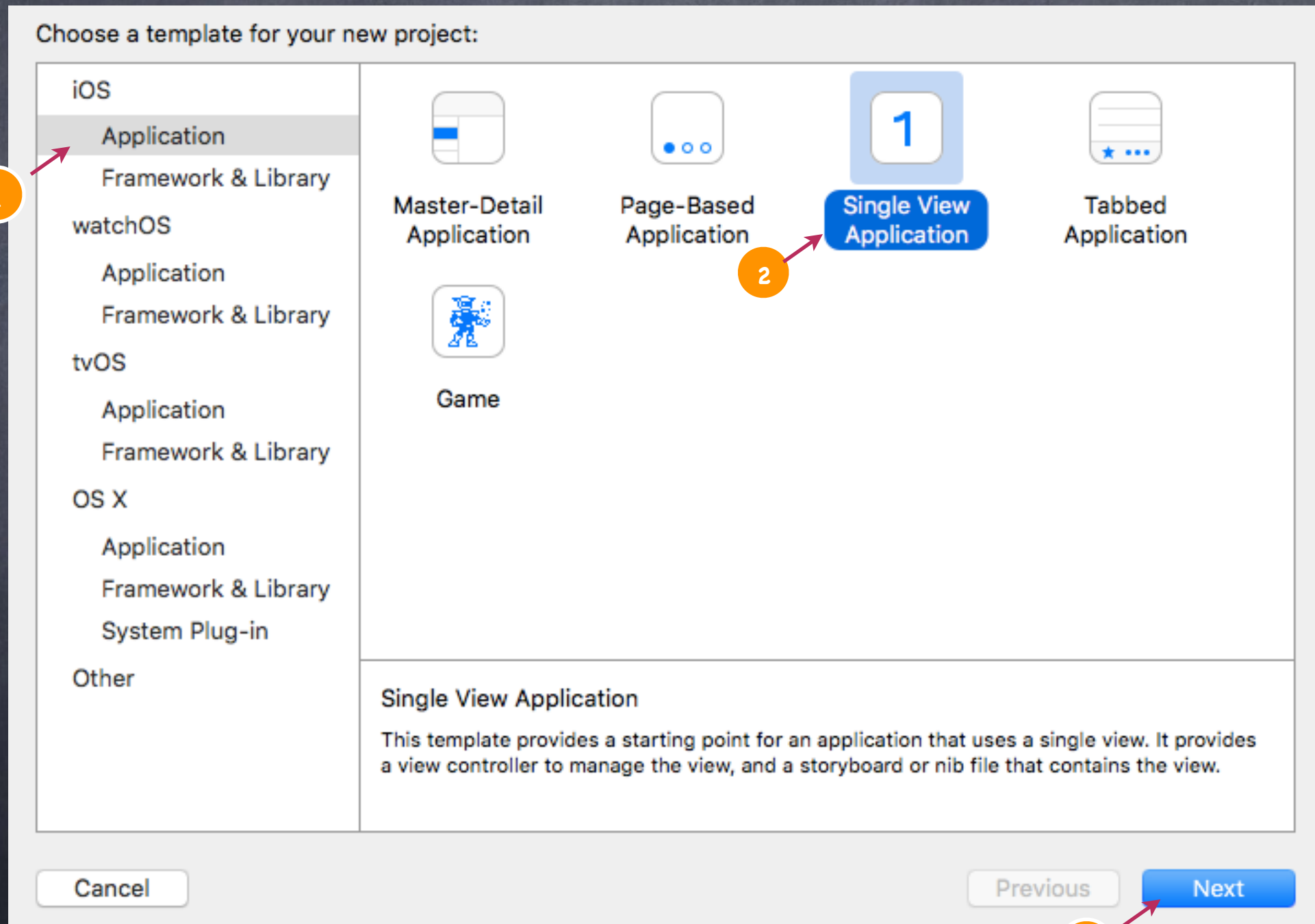
O que é Parser

- Um Parser é um programa de computador ou apenas um componente de um programa 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, o parser pode ser um leitor que ajuda na conversão do arquivo para manipulação dos dados contidos no mesmo.
- Para trafegar informações entre o servidor e o aparelho usamos arquivos XML ou JSON, existem dois tipos de Parser de XML: O SAX e o DOM o SAX é o mais conhecido e consome poucos recursos, o SAX é mais trabalhoso que o DOM, O DOM lê o arquivo inteiro em memória criando uma árvore que pode ter suas tags do XML lidas com muita facilidade, O SAX vai navegando pela estrutura do XML linha a linha, e no momento que as tags são encontradas um objeto da aplicação é notificado para ler os dados. O SAX utiliza menos memória que o DOM.
- Pela simplicidade vamos abordar apenas o PARSER do JSON.

Dica: Mais exemplos de JSON em: <http://json.org/example.html>

Iniciando o Projeto

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



Os dados do Projeto

- 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 escolha Objective-C, em Devices selecione iPhone.

Choose options for your new project:

4 → Product Name: Exemplo2_Rest_iOS

Organization Name: Agesandro Scarpioni

Organization Identifier: com.scarpioni

Bundle Identifier: com.scarpioni.Exemplo2-Rest-iOS

5 → Language: Objective-C

6 → Devices: iPhone

☐ Use Core Data

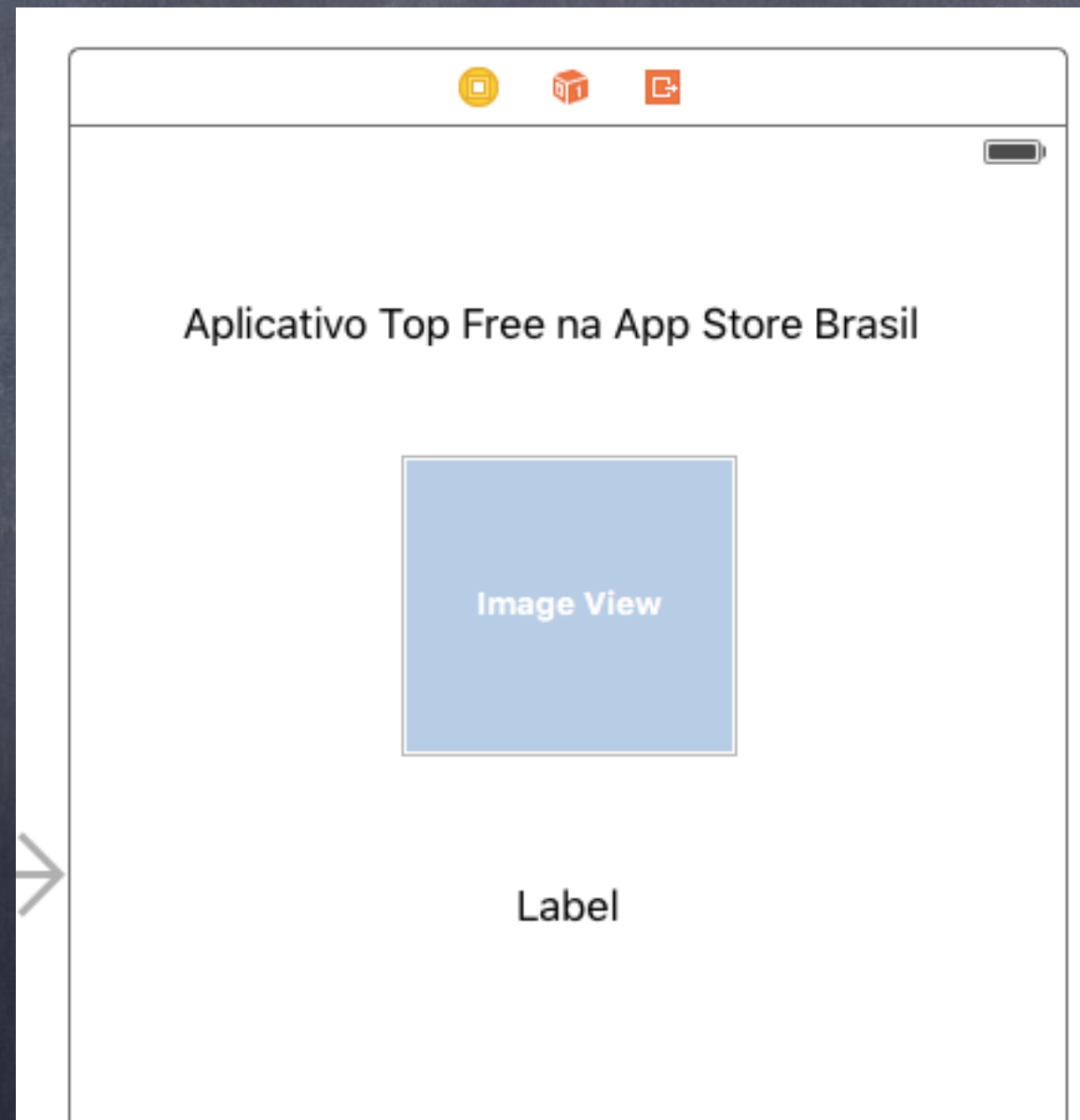
☐ Include Unit Tests

☐ Include UI Tests

Cancel Previous Next

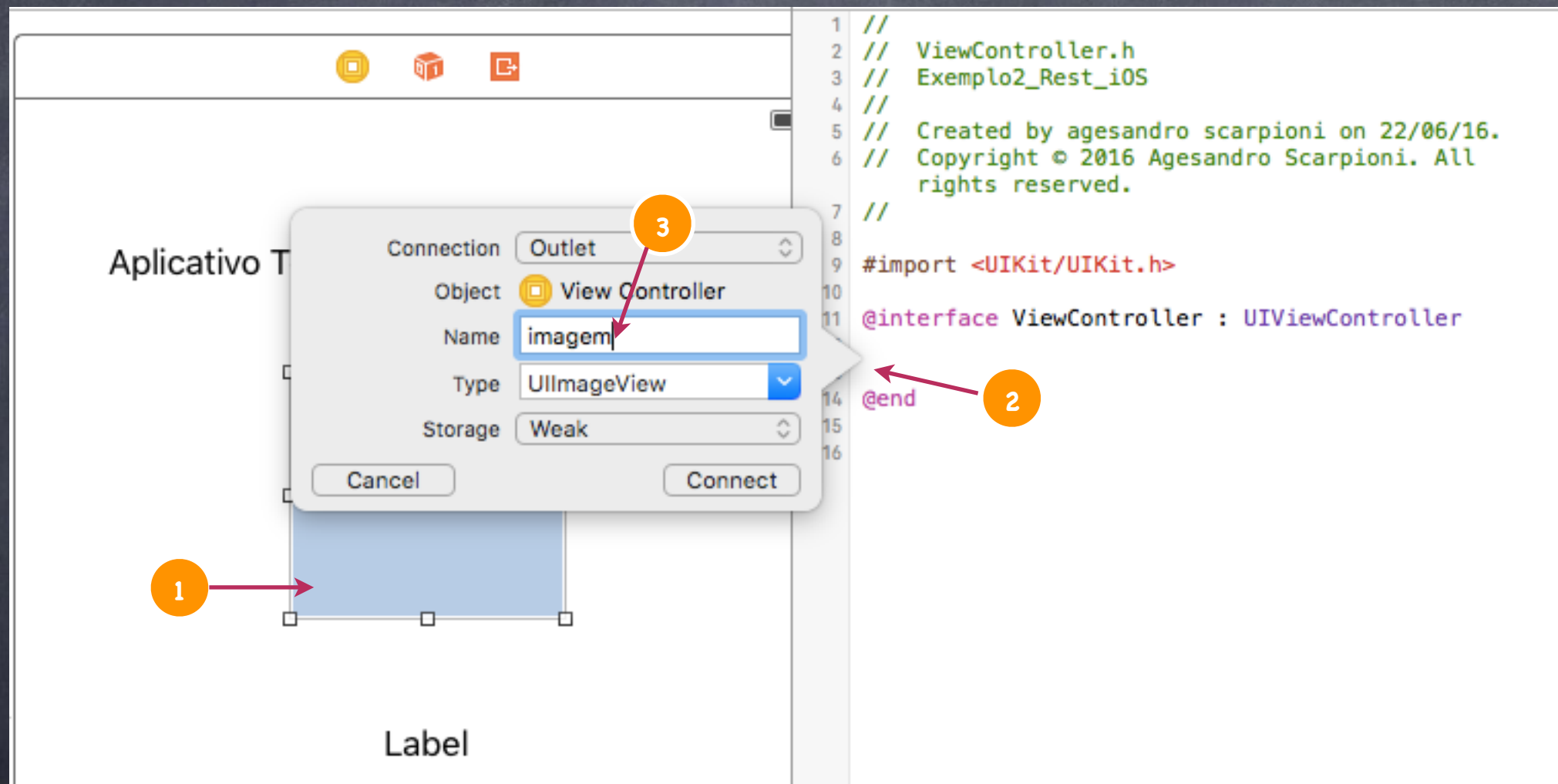
A Interface

- Desenhe a tela abaixo com 2 labels e 1 ImageView.



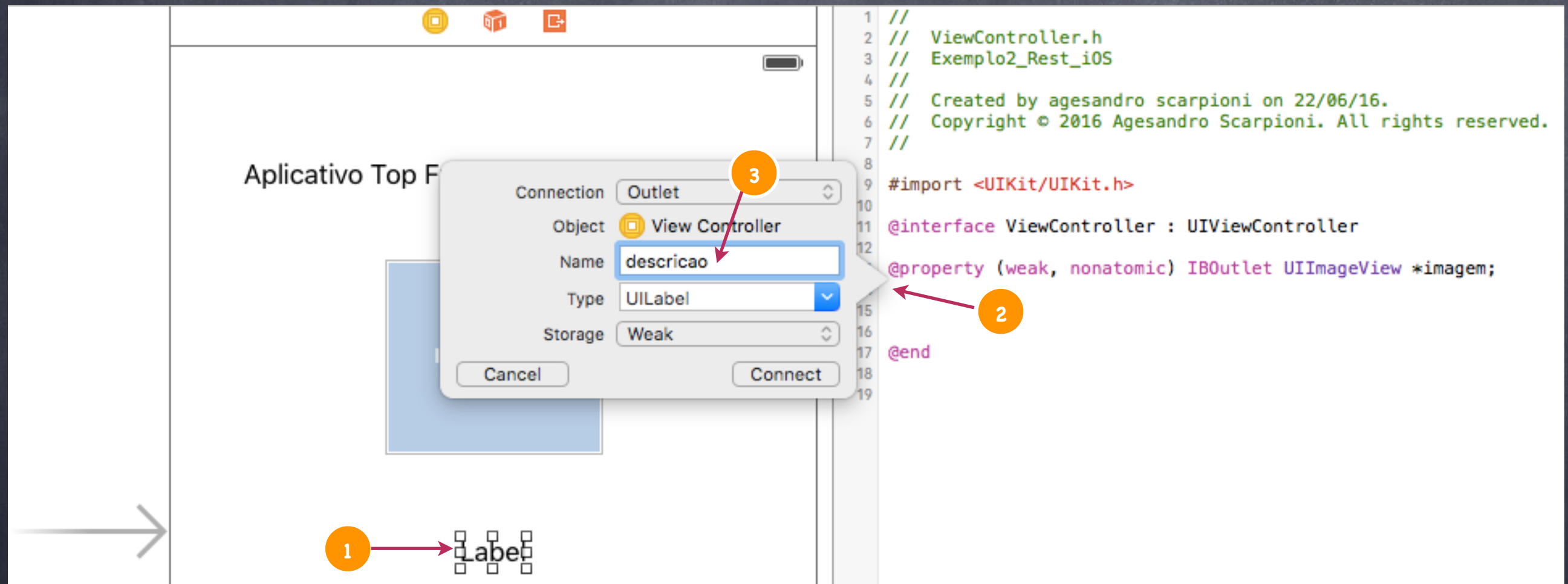
Definindo os IBOutlet's

- Criar no .h os Outlet's do Label e do Image. Selecione o objeto Image (1) e arraste até a área indicada (2), nomeie como imagem (3).



Definindo os IBOutlet's

- Selecione o objeto Label (1) e arraste até a área indicada (2), nomeie como descricao (3).



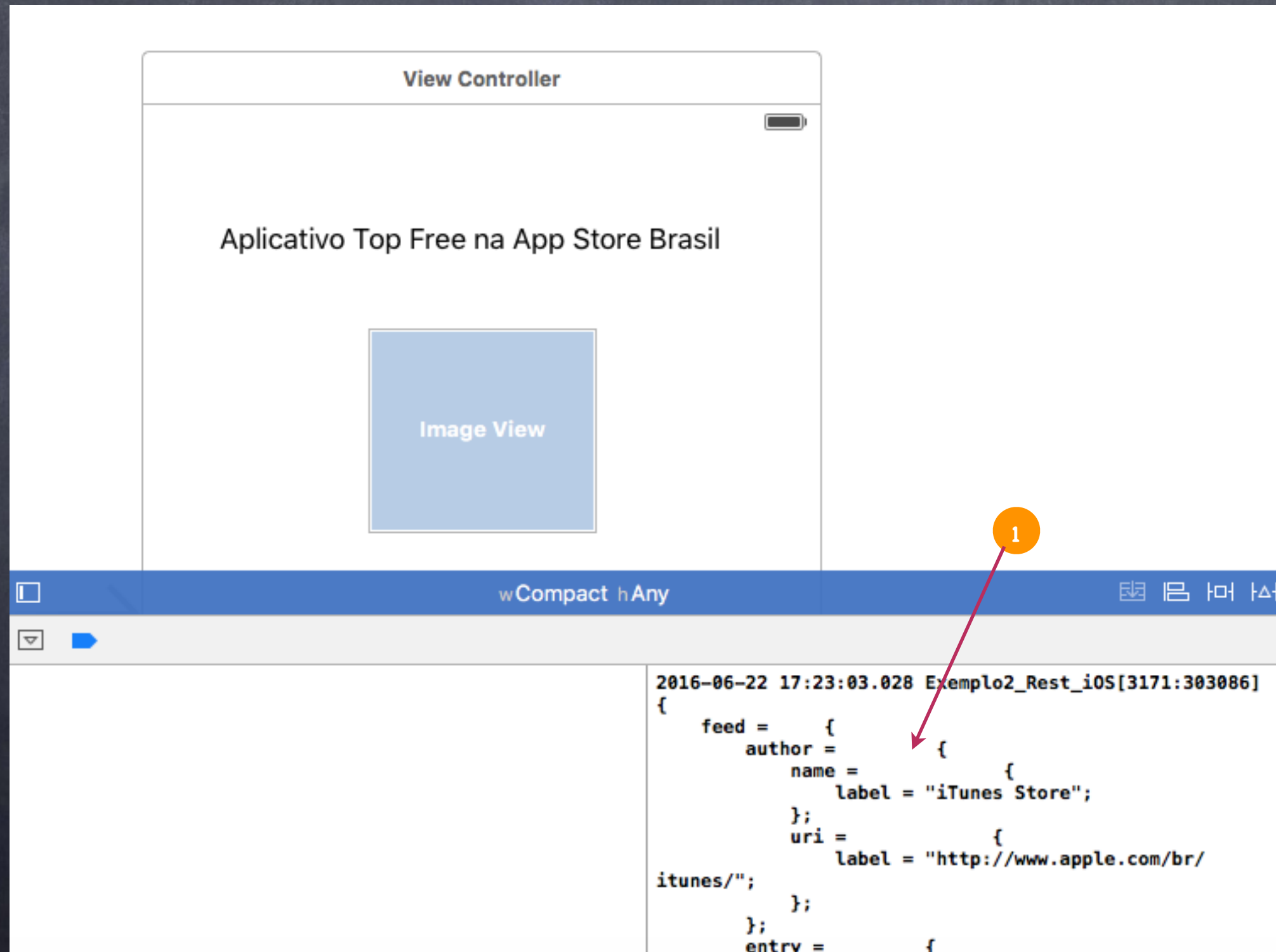
Implementando o viewDidLoad

- Nas linhas abaixo é feito o request e o parser do Json para um NSDictionary, a url abaixo é da própria App Store, o termo limit=1 irá trazer o aplicativo top 1 da loja, caso troque o limit para 10 será montado um dicionário com os App's top 10, porém, para exibir os dados seria necessário uma lista.

```
1 //
2 // ViewController.m
3 // Exemplo2_Rest_iOS
4 //
5 // Created by agesandro scarpioni on 22/06/16.
6 // Copyright © 2016 Agesandro Scarpioni. All rights reserved.
7 //
8
9 #import "ViewController.h"
10
11 @interface ViewController ()
12
13 @end
14
15 @implementation ViewController
16
17 - (void)viewDidLoad {
18     [super viewDidLoad];
19     NSURL *url=[NSURL URLWithString:@"https://itunes.apple.com/br/rss/topfreeapplications/limit=1/json"];
20
21     //criando o NSData e fazendo um request
22     NSData *data = [NSData dataWithContentsOfURL:url];
23
24     //transformando o NSData para NSDictionary, o NSJSONSerialization faz o Parser do Json
25     NSError *error;
26     NSDictionary *json =[NSJSONSerialization JSONObjectWithData:data options: kNilOptions error:&error];
27
28     //caso precise, isto vai exibir o conteúdo do json, assim é possível identificar os nomes das chaves do dicionário.
29     NSLog(@"%@", json);
30
31 }
```


Implementando o viewDidLoad

- Ao executar o programa, irá aparecer no console o conteúdo do Json (1).



Estrutura do Json da App Store

```
2016-07-11 13:05:08.974 Exemplo2_Rest_iOS[3160:186332] {
  feed = {
    author = {
      name = {
        label = "iTunes Store";
      };
      uri = {
        label = "http://www.apple.com/br/itunes/";
      };
    };
    entry = {
      category = {
        attributes = {
          "im:id" = 6003;
          label = Viagens;
          scheme = "https://itunes.apple.com/br/genre/ios-viagens/id6003?mt=8&uo=2";
          term = Travel;
        };
      };
      id = {
        attributes = {
          "im:bundleId" = "com.ubercab.UberClient";
          "im:id" = 368677368;
        };
        label = "https://itunes.apple.com/br/app/uber/id368677368?mt=8&uo=2";
      };
      "im:artist" = {
        attributes = {
          href = "https://itunes.apple.com/br/developer/uber-technologies-inc./id368677371?mt=8&uo=2";
        };
        label = "Uber Technologies, Inc.";
      };
      "im:contentType" = {
        attributes = {
          label = Aplicativo;
          term = Application;
        };
      };
      "im:image" = (
        [0] {
          attributes = {
            height = 53;
          };
          label = "http://is2.mzstatic.com/image/thumb/Purple18/v4/d3/62/52/d36252b8-4d49-cf57-df8f-6a03fc18e520/mzl.vtoredrs.png/53x53bb-85.png";
        },
        [1] {
          attributes = {
            height = 75;
          };
          label = "http://is4.mzstatic.com/image/thumb/Purple18/v4/d3/62/52/d36252b8-4d49-cf57-df8f-6a03fc18e520/mzl.vtoredrs.png/75x75bb-85.png";
        }
      )
    }
  }
}
```


Continuação

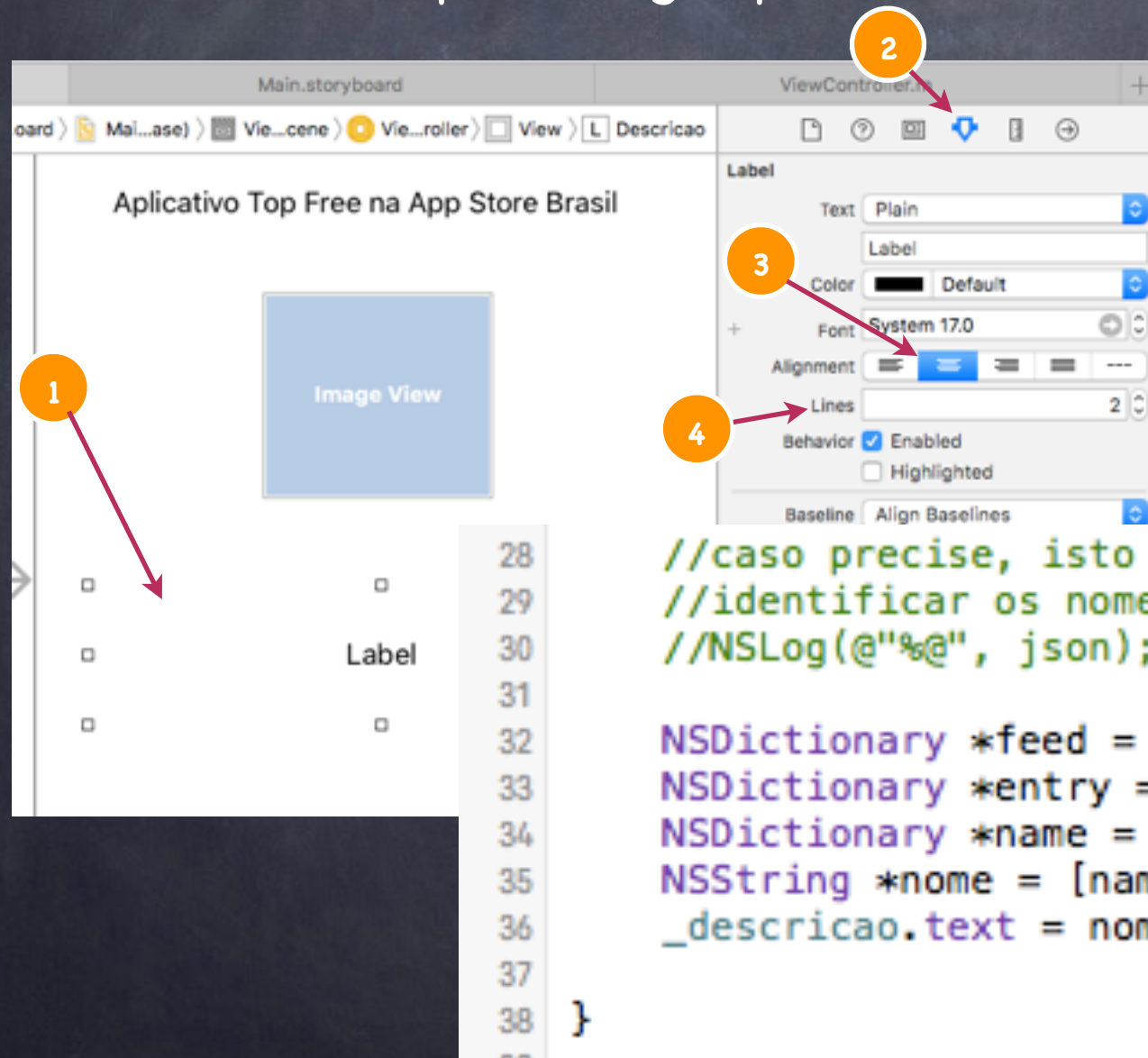
```
    },
    {
      [2] attributes =
            {
              height = 100;
            };
      label = "http://is4.mzstatic.com/image/thumb/Purple18/v4/d3/62/52/d36252b8-4d49-cf57-df8f-6a03fc18e520/mzl.vtoredrs.png/100x100bb-85.png";
    }
  );
  "im:name" =
    {
      label = Uber;
    };
  "im:price" =
    {
      attributes =
        {
          amount = "0.00000";
          currency = USD;
        };
      label = 0bter;
    };
  "im:releaseDate" =
    {
      attributes =
        {
          label = "20/05/2010";
        };
      label = "2010-05-20T20:11:23-07:00";
    };
  link =
    {
      attributes =
        {
          href = "https://itunes.apple.com/br/app/uber/id368677368?mt=8&uo=2";
          rel = alternate;
          type = "text/html";
        };
    };
  rights =
    {
      label = "\U00a9 Uber Technologies Inc.";
    };
  summary =
    {
      label = "Consiga uma viagem confi\U00elvel em minutos com o aplicativo Uber \U2014 sem reservas ou filas de espera de t\U00elxi. \n
      \nTodas as op\U00e7\U00f5es, desde a viagem de baixo custo at\U00e9 a premium, funcionam como uma atualiza\U00e7\U00e3o aos sistemas comuns. \n\nCrie
      sua conta para explorar o aplicativo. Adicione um cart\U00e3o de cr\U00e9dito ou vincule seu PayPal, que sua tarifa ser\U00el cobrada automaticamente no
      final da sua viagem. Voc\U00ea tamb\U00e9m pode pagar com dinheiro em algumas cidades. Depois de sua viagem, vamos enviar um recibo a voc\U00ea por e-
      mail. \n\nVeja se o Uber est\U00el dispon\U00edvel em sua cidade em https://www.uber.com/cities\nSiga-nos no Twitter em https://twitter.com/uber\nCurta-
      nos no Facebook em https://www.facebook.com/uber\n\nD\U00favidas? Acesse https://help.uber.com/";
    };
};
```


Continuação

```
        label = "\U00a9 Uber Technologies Inc.";
    };
    → summary =
        {
            label = "Consiga uma viagem confi\U00elvel em minutos com o aplicativo Uber \U2014 sem reservas ou filas de espera de t\U00elxi. \n
\nTodas as op\U00e7\U00f5es, desde a viagem de baixo custo at\U00e9 a premium, funcionam como uma atualiza\U00e7\U00e3o aos sistemas comuns. \n\nCrie
sua conta para explorar o aplicativo. Adicione um cart\U00e3o de cr\U00e9dito ou vincule seu PayPal, que sua tarifa ser\U00el cobrada automaticamente no
final da sua viagem. Voc\U00ea tamb\U00e9m pode pagar com dinheiro em algumas cidades. Depois de sua viagem, vamos enviar um recibo a voc\U00ea por e-
mail. \n\nVeja se o Uber est\U00el dispon\U00edvel em sua cidade em https://www.uber.com/cities\nSiga-nos no Twitter em https://twitter.com/uber\nCurta-
nos no Facebook em https://www.facebook.com/uber\n\nD\U00favidas? Acesse https://help.uber.com/";
        };
    → title =
        {
            label = "Uber - Uber Technologies, Inc.";
        };
    };
    → icon =
        {
            label = "http://itunes.apple.com/favicon.ico";
        };
    → id =
        {
            label = "https://itunes.apple.com/br/rss/topfreeapplications/limit=1/json";
        };
    → link =
        (
            {
                attributes =
                    {
                        href = "https://itunes.apple.com/WebObjects/MZStore.woa/wa/viewTop?cc=br&id=132006&popId=27";
                        rel = alternate;
                        type = "text/html";
                    };
            },
            {
                attributes =
                    {
                        href = "https://itunes.apple.com/br/rss/topfreeapplications/limit=1/json";
                        rel = self;
                    };
            }
        );
    → rights =
        {
            label = "Copyright 2008 Apple Inc.";
        };
    → title =
        {
            label = "iTunes Store: Top apps gr\U00eltis";
        };
    → updated =
        {
            label = "2016-07-11T08:39:55-07:00";
        };
    };
}
```


Exibir o nome Top Free FIAP

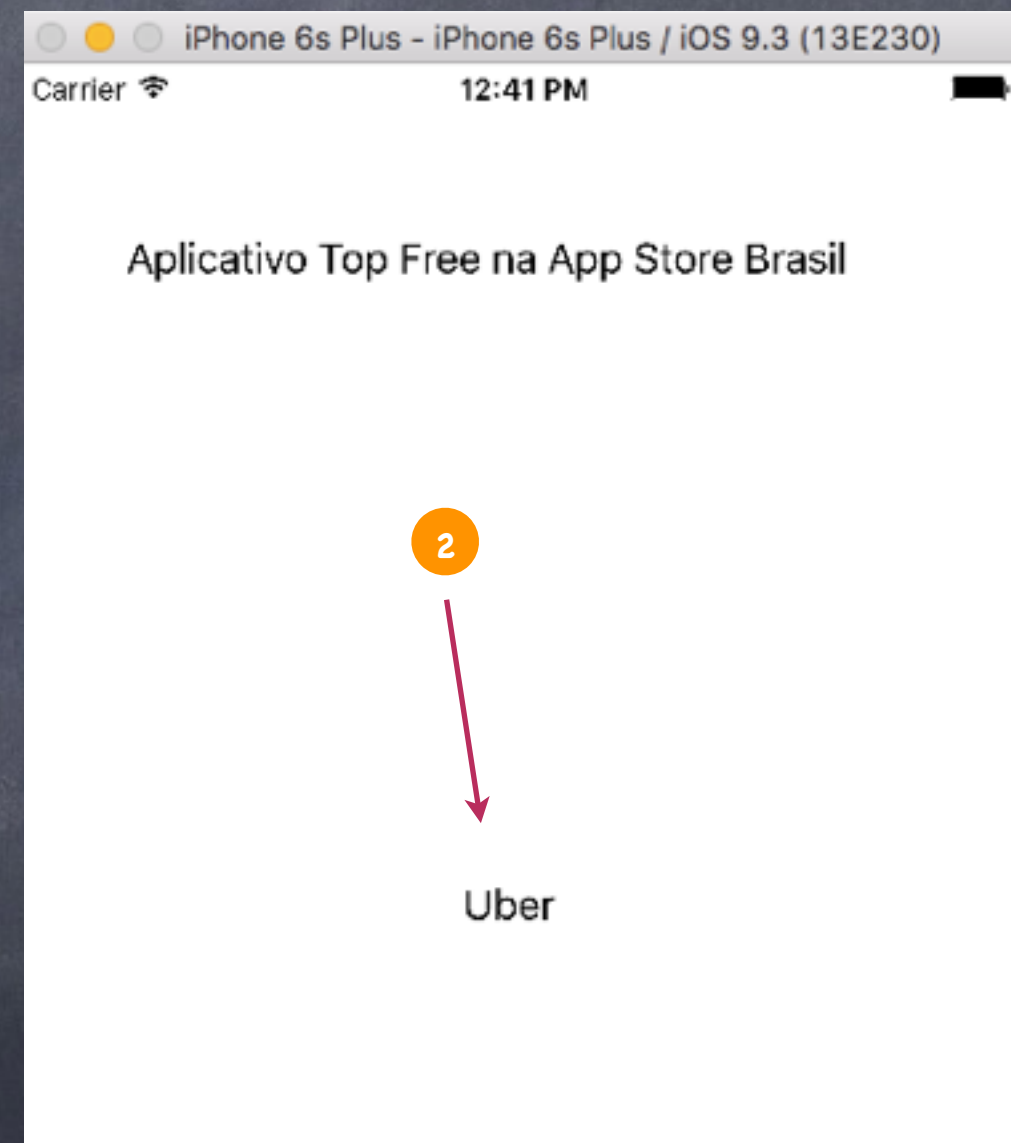
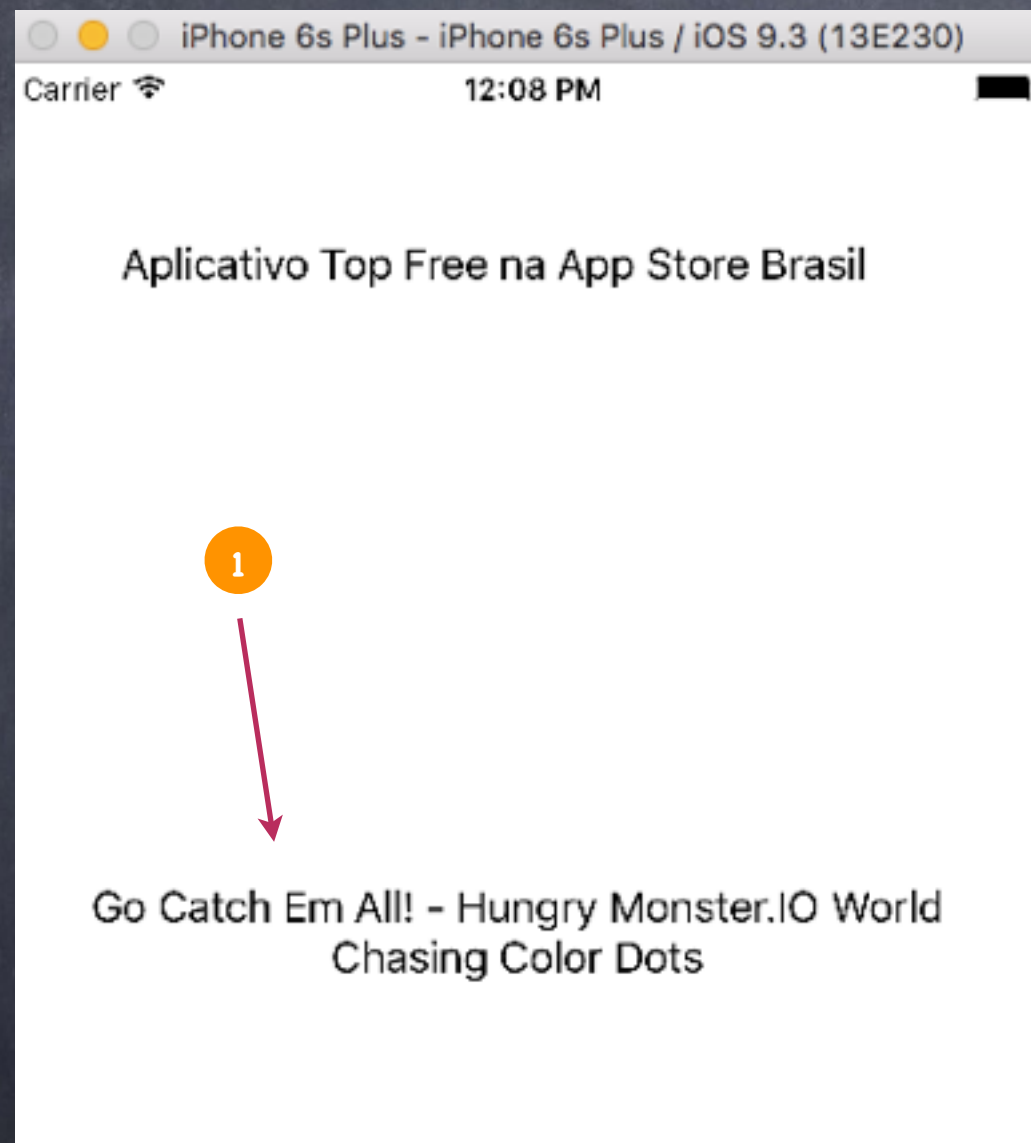
- Selecione e aumente a área do Label (1), em propriedades (2), alinhe o texto ao centro (3) e deixe formatado para o texto aparecer em 2 linhas (4). Comente a linha 30 e crie os dicionários das linhas 32,33,34 para que seja possível penetrar na estrutura do Json e acessar o nome do aplicativo, descarregue esse nome em uma variável do tipo String e posteriormente exiba a variável no outlet do Label.



```
28 //caso precise, isto vai exibir o conteúdo do json, assim é possível
29 //identificar os nomes das chaves do dicionário.
30 //NSLog(@"%@", json);
31
32 NSDictionary *feed = [json objectForKey:@"feed"];
33 NSDictionary *entry = [feed objectForKey:@"entry"];
34 NSDictionary *name = [entry objectForKey:@"im:name"];
35 NSString *nome = [name objectForKey:@"label"];
36 _descricao.text = nome;
37
38 }
```


Resultado

- Ao executar o programa irá aparecer o nome do Top Free da Apple Store no dia 11/07/2016, exemplo de dois momentos às 12h08 (1) e às 12:41 (2)



Exibindo a imagem

- As imagens ficam em um array dentro de um dicionário chamado "im:image", faça a programação da linha 38 criando um array mutável chamado img, na linha 39 descarregue nesse array o conteúdo do dicionário "im:image", na linha 40 exiba o array com NSLog execute o programa e veja o resultado (1)

```
35 NSString *nome = [name objectForKey:@"label"];
36 _descricao.text = nome;
37
38 NSMutableArray *img = [[NSMutableArray alloc] init];
39 img = [entry objectForKey:@"im:image"];
40 NSLog(@"%@", img);
```

1

```
2016-07-11 12:39:58.006 Exemplo2_Rest_iOS[2925:171158] (
    {
        attributes = {
            height = 53; [0]
        };
        label = "http://is2.mzstatic.com/image/thumb/Purple18/v4/d3/62/52/d36252b8-4d49-cf57-df8f-6a03fc18e520/mzl.vtoredrs.png/53x53bb-85.png";
    },
    {
        attributes = {
            height = 75; [1]
        };
        label = "http://is4.mzstatic.com/image/thumb/Purple18/v4/d3/62/52/d36252b8-4d49-cf57-df8f-6a03fc18e520/mzl.vtoredrs.png/75x75bb-85.png";
    },
    {
        attributes = {
            height = 100; [2]
        };
        label = "http://is4.mzstatic.com/image/thumb/Purple18/v4/d3/62/52/d36252b8-4d49-cf57-df8f-6a03fc18e520/mzl.vtoredrs.png/100x100bb-85.png";
    }
)
```


Exibindo a imagem

- Observe que existem 3 tamanhos de imagem, uma em cada índice, o índice 0 tem a imagem com altura 53, o índice 1 tem a imagem com altura 75 e o índice 2 tem a imagem com altura 100, acesse a url da imagem escolhendo o índice apropriado (0,1 ou 2) e utilize o dicionário "label" para acessar a URL da imagem.

```
35 NSString *nome = [name objectForKey:@"label"];
36 _descricao.text = nome;
37
38 NSMutableArray *img = [[NSMutableArray alloc] init];
39 img = [entry objectForKey:@"im:image"];
40 NSLog(@"%@", img);
41
42 }
43
44 - (void)didReceiveMemoryWarning {
45     [super didReceiveMemoryWarning];
46 }
47
48
49 @end
50
51
```

Exemplo2_Rest_iOS

Array

```
2016-07-11 12:39:58.006 Exemplo2_Rest_iOS[2925:171158] (
    {
        attributes = {
            height = 53;
        };
        label = "http://is2.mzstatic.com/image/thumb/Purple18/v4/d3/62/52/d36252b8-4d49-cf57-df8f-6a03fc18e520/mzl.vtoredrs.png/53x53bb-85.png";
    },
    {
        attributes = {
            height = 75;
        };
        label = "http://is4.mzstatic.com/image/thumb/Purple18/v4/d3/62/52/d36252b8-4d49-cf57-df8f-6a03fc18e520/mzl.vtoredrs.png/75x75bb-85.png";
    },
    {
        attributes = {
            height = 100;
        };
        label = "http://is4.mzstatic.com/image/thumb/Purple18/v4/d3/62/52/d36252b8-4d49-cf57-df8f-6a03fc18e520/mzl.vtoredrs.png/100x100bb-85.png";
    }
)
```

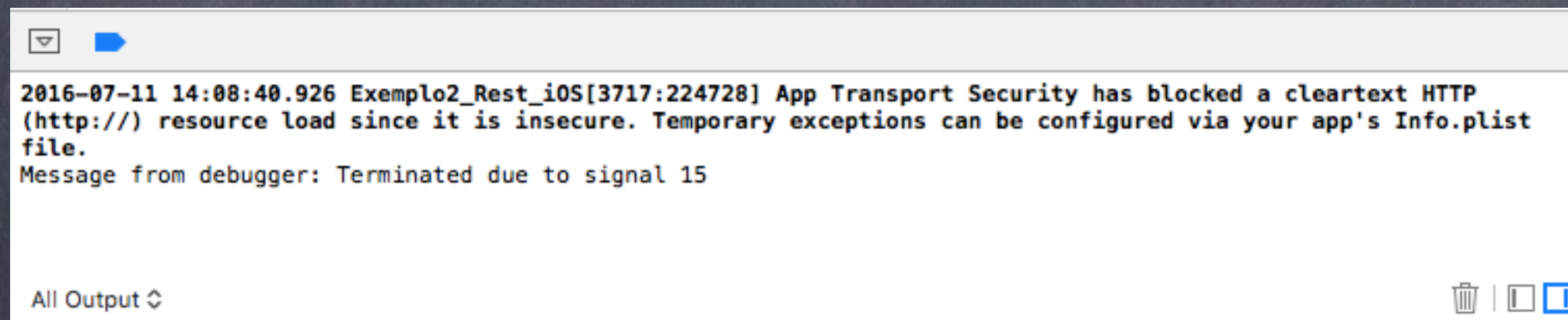

Exibindo a imagem

- Comente a linha 40, crie um NSString para receber o dicionário "label" do índice 2 do array, no índice 2 está armazenada a URL da maior imagem do App.

```
37
38 NSMutableArray *img = [[NSMutableArray alloc] init];
39 img = [entry objectForKey:@"im:image"];
40 //NSLog(@"%@",img);
41 NSString *urlFoto = [img[2] objectForKey:@"label"];
42 data = [NSData dataWithContentsOfURL:[NSURL URLWithString:urlFoto]];
43 UIImage *icone = [[UIImage alloc] initWithData:data];
44 _imagem.image = icone;
```


Exibindo a imagem

- Ao executar observe o erro que ocorre quando você tenta exibir uma imagem utilizando o http ao invés de https, sobre essas informações de segurança reveja o slide aula_13_2016_WebView_ObjC página 19 até a 23.



```
2016-07-11 14:08:40.926 Exemplo2_Rest_iOS[3717:224728] App Transport Security has blocked a cleartext HTTP (http://) resource load since it is insecure. Temporary exceptions can be configured via your app's Info.plist file.  
Message from debugger: Terminated due to signal 15
```

All Output ↕

- Para continuar o exercício veja na próxima página uma das soluções que está disponível na página 22 do slide aula_13_2016_WebView_ObjC.

Info.plist (forma 1) FIAP

- Abra o arquivo info.plist(1), clique no símbolo + (2), role as opções para encontrar: App Transport Security Settings (3), clique no símbolo do triângulo indicando-o para baixo (4), dessa forma é possível incluir um sub item, role as opções para encontrar: Allow Arbitrary Loads (5), selecione YES (6).

The screenshot illustrates the process of adding a new key to the Info.plist file in Xcode. The interface is divided into three main sections: the Project Navigator on the left, the Properties Inspector in the center, and the Keychain Access window on the right.

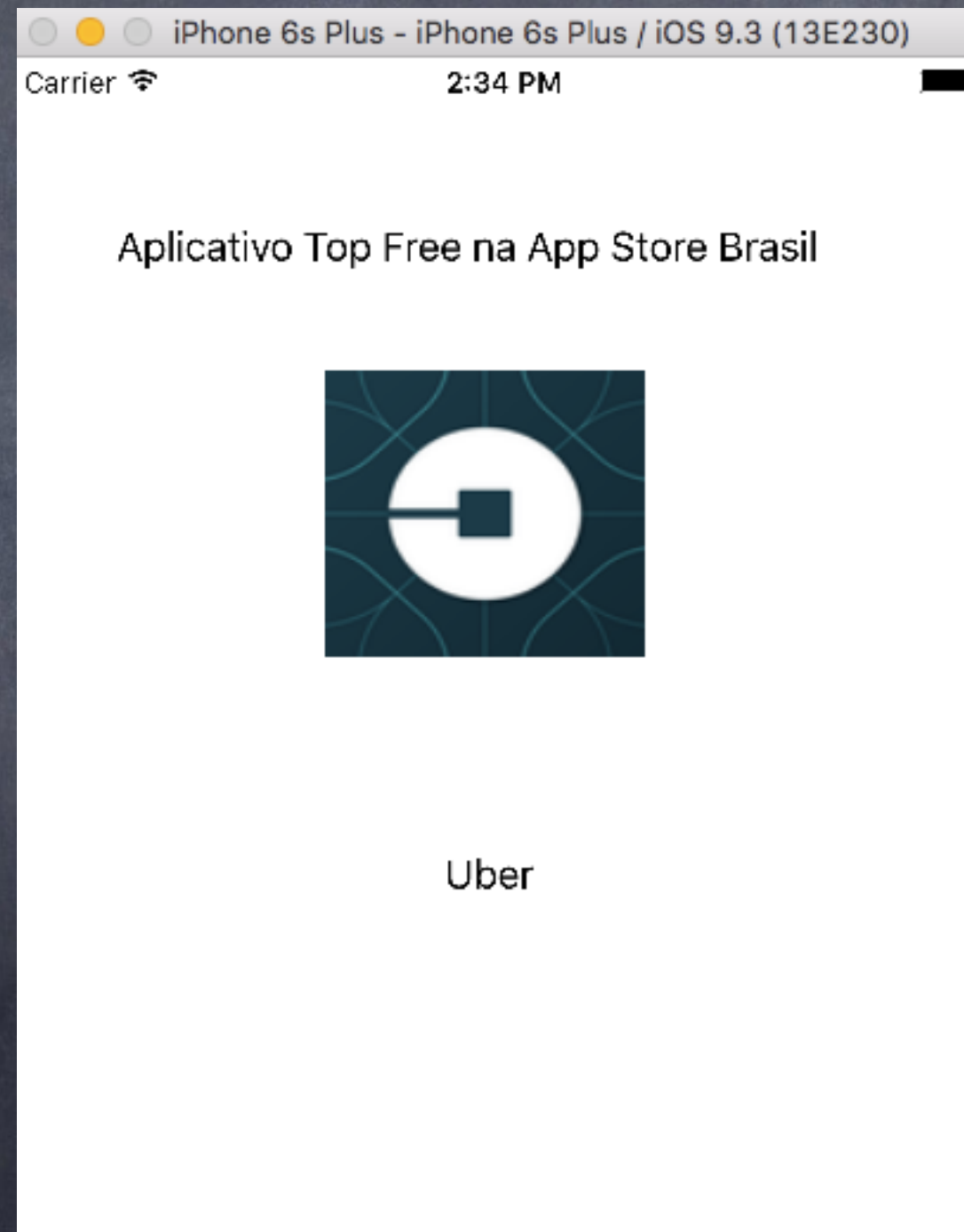
Project Navigator (Left): Shows the project structure for 'Exemplo2_Rest_iOS'. The 'Info.plist' file is highlighted with a red arrow and a yellow circle labeled '1'.

Properties Inspector (Center): Displays the 'Key' and 'Type' columns for the 'Information Property List'. The 'Supported interface orientations' key is highlighted with a red arrow and a yellow circle labeled '2'. A red arrow points to the '+' button next to it.

Keychain Access Window (Right): Shows the 'App Transport Security Settings' dictionary. The 'App Transport Security Settings' key is highlighted with a red arrow and a yellow circle labeled '3'. A red arrow points to the triangle icon next to it, labeled '4'.

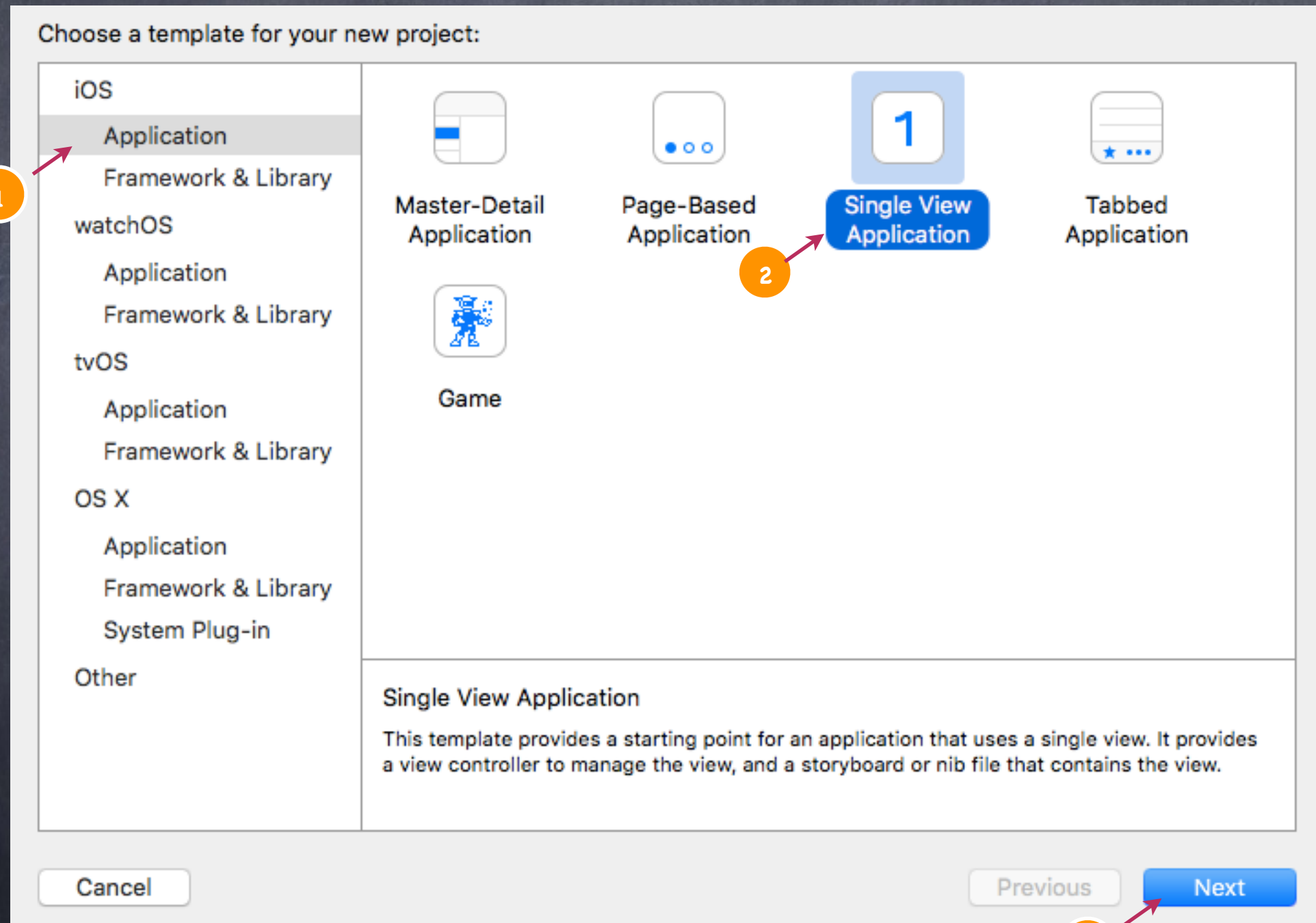
Keychain Access Window (Bottom): Shows the 'App Transport Security Settings' dictionary with one item, 'Allow Arbitrary Loads', which is highlighted with a red arrow and a yellow circle labeled '5'. The value 'YES' is selected, indicated by a red arrow and a yellow circle labeled '6'.

Exibindo a imagem



Iniciando outro Projeto

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



Os dados do Projeto

- 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 escolha Objective-C, em Devices selecione iPhone.

Choose options for your new project:

Product Name:

4 Organization Name:

Organization Identifier:

Bundle Identifier:

Language:

5 Devices:

6

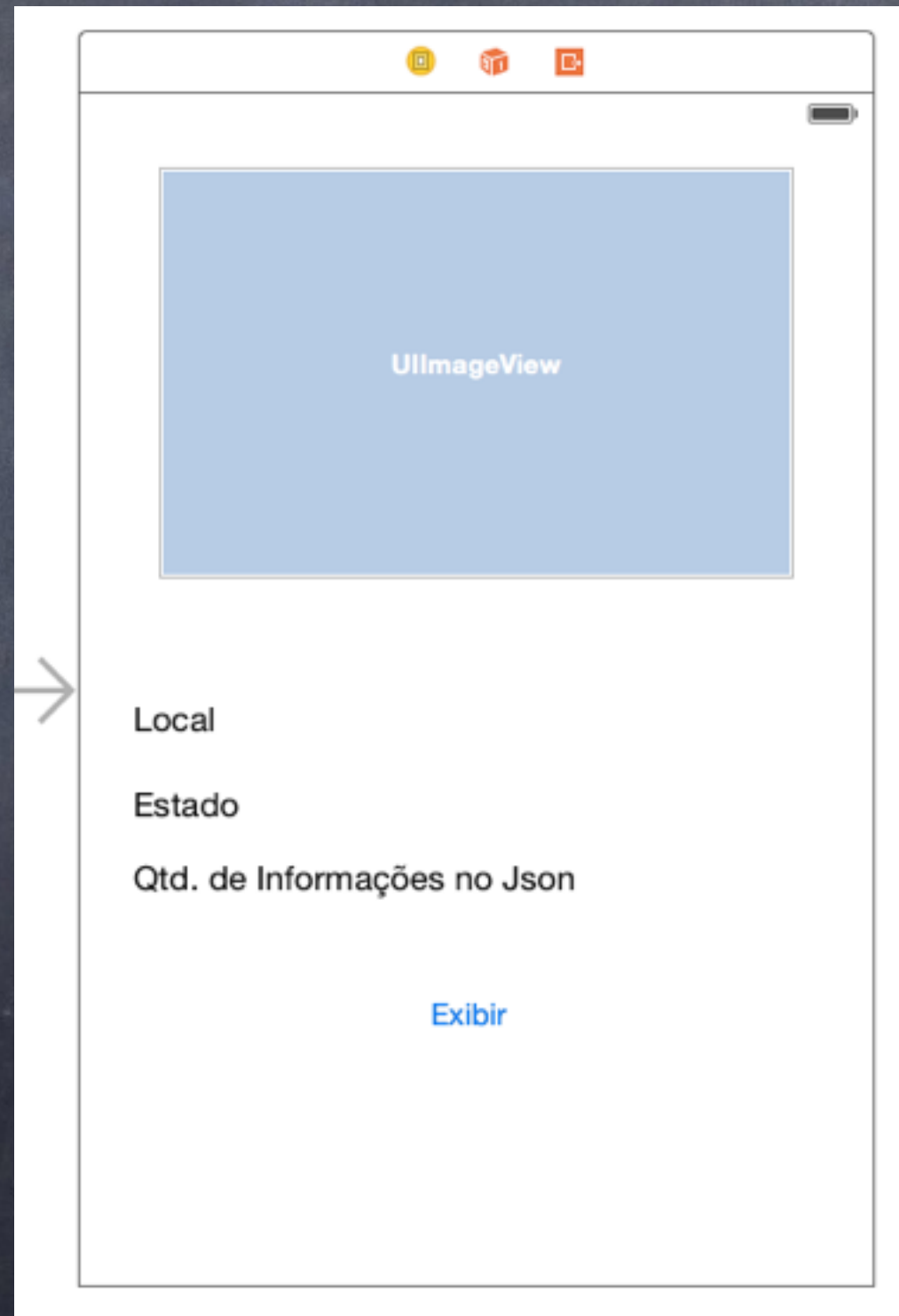
☐ Use Core Data

☐ Include Unit Tests

☐ Include UI Tests

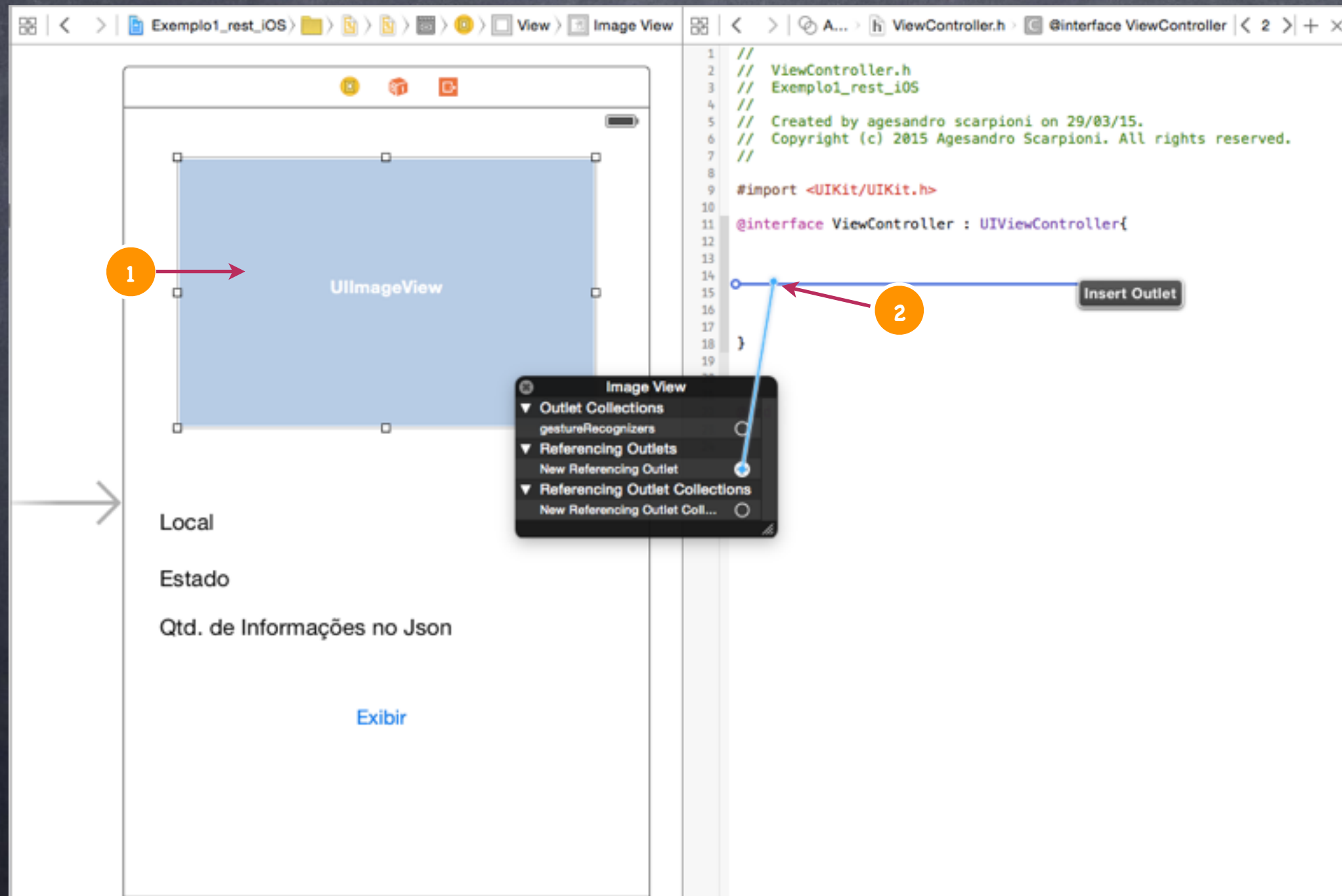
A Interface

- Desenhe a tela abaixo com 3 label's, 1 button e 1 UIImageView.



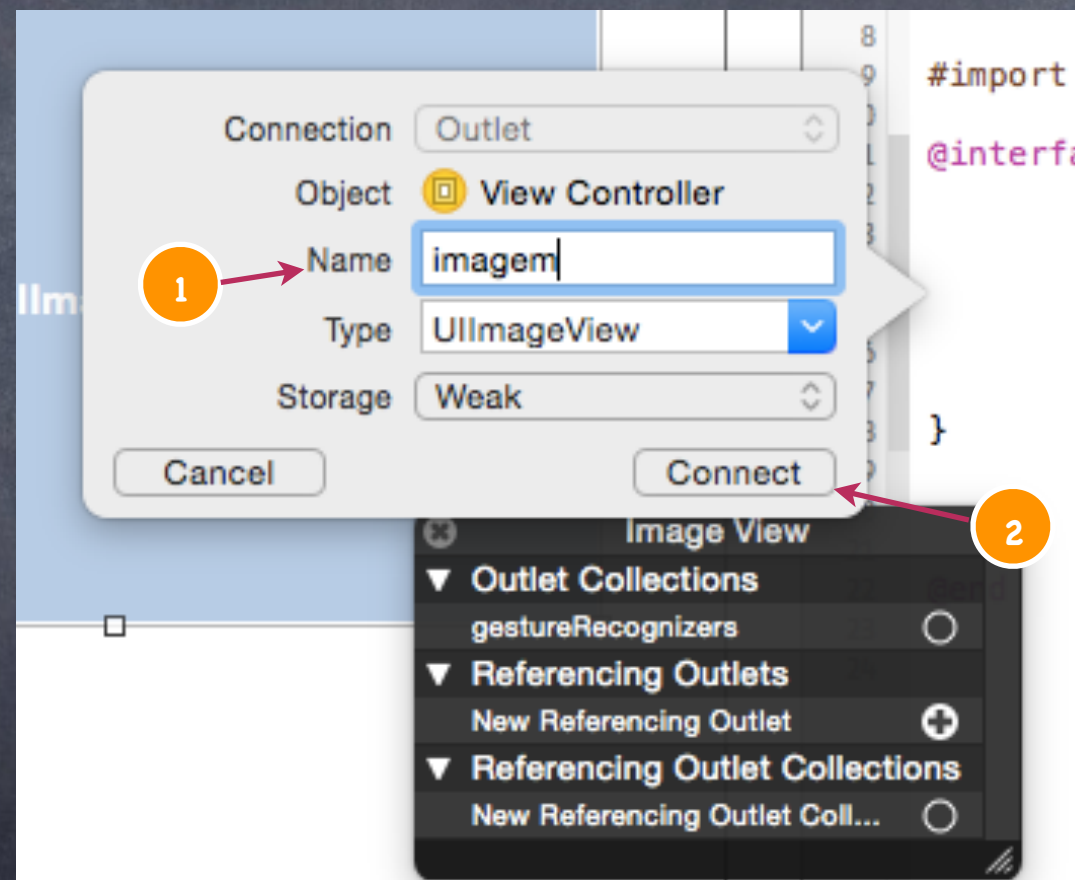
Definindo os IBOutlet's

- Abra duas telas simultâneas e vamos criar no .h os Outlet's dos Labels e do Image. Atenção coloque as chaves { } e com botão direito sobre o objeto (1), escolha New Referencing Outlet e arraste dentro da área das chaves (2).

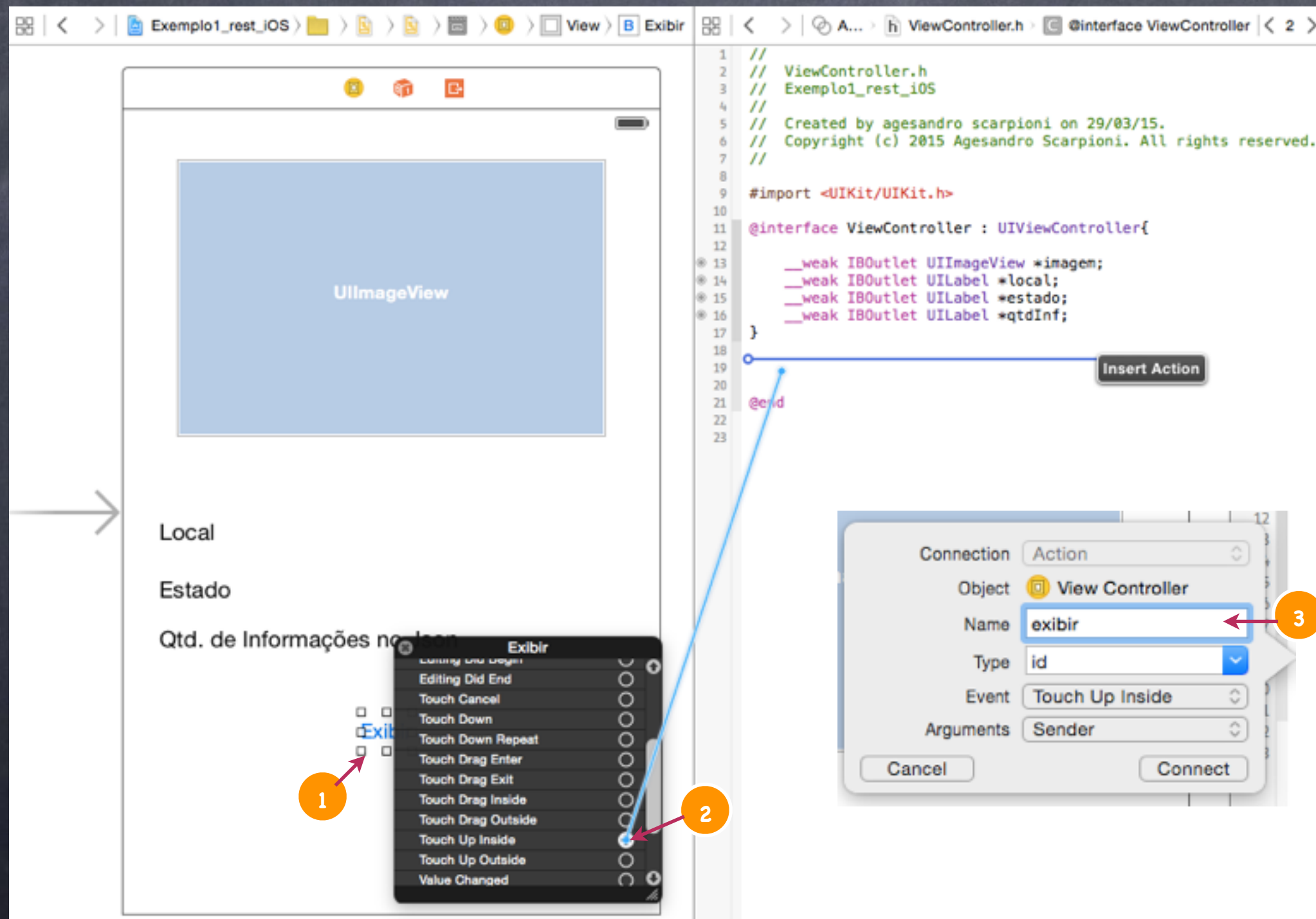


Definindo os IBOutlet's

- Ao soltar o botão um nome para o Outlet é requisitado, informe "imagem" e clique em Connect. Faça o mesmo para os labels, veja a sugestão dos nomes no próximo slide.




- Clique com o botão direito sobre o Exibir, escolha Touch Up Inside e arraste até a área fora das chaves { }, Nomeie o IBAction como exhibir



Definindo os IBAction's

- Note que ao fazer a ligação do IBAction além da declaração do método no arquivo .h, também a área da implementação (1) aparece no arquivo .m.

```
1 //
2 // ViewController.m
3 // Exemplo1_rest_iOS
4 //
5 // Created by agesandro scarpioni on 29/03/15.
6 // Copyright (c) 2015 Agesandro Scarpioni. All rights reserved.
7 //
8
9 #import "ViewController.h"
10
11 @interface ViewController ()
12
13 @end
14
15 @implementation ViewController
16
17 - (void)viewDidLoad {
18     [super viewDidLoad];
19     // Do any additional setup after loading the view, typically from a nib.
20 }
21
22 - (void)didReceiveMemoryWarning {
23     [super didReceiveMemoryWarning];
24     // Dispose of any resources that can be recreated.
25 }
26
27 - (IBAction)exibir:(id)sender {
28 }
29 @end
30
```



Implementando o Exibir

- Nas linhas abaixo estamos fazendo o request e o parser do Json para um NSDictionary, a url scarpioni.com é de um webservice em php hospedado no meu servidor GoDaddy acessando banco Mysql, a outra url é de um webservice em Node.JS hospedado no Heroku acessando banco MongoDB.

```
27 - (IBAction)exibir:(id)sender {
28     //acesso a API feita em PHP criada em meu servidor na Godaddy
29     //NSURL *url=[NSURL URLWithString:@"https://www.scarpioni.com/webservices/local.php/?id=1"];
30     //acesso a API feita em Node.JS criada no heroku
31     NSURL *url=[NSURL URLWithString:@"https://parks-api.herokuapp.com/parks/577024e4a44821110001ee93"];
32
33     //criando o NSData e fazendo um request
34     NSData *data = [NSData dataWithContentsOfURL:url];
35
36     //transformando o NSData para NSDictionary, o NSJSONSerialization faz o Parser do Json
37     NSError *error;
38     NSDictionary *json =[NSJSONSerialization JSONObjectWithData:data options: kNilOptions error:&error];
39
40     //caso precise, isto vai exibir o conteúdo do json, assim é possível
41     //identificar os nomes das chaves do dicionário.
42     NSLog(@"%@", json);
43 }
```

- Ao executar o App e clicar no botão Exibir, os dados do Json irá aparecer no console como mostra a figura abaixo:



The screenshot shows the Xcode console window for the project 'Exemplo1_rest_iOS'. The console output displays a JSON object with the following fields: `__v`, `_id`, `cidade`, `datacriacao`, `estado`, `nome`, `urlfoto`, and `usuario`. The values are: `__v` is 0, `_id` is 577024e4a44821110001ee93, `cidade` is "S\u00e3o Paulo", `datacriacao` is "2016-06-26T18:54:28.872Z", `estado` is SP, `nome` is "Parque Villa Lobos", `urlfoto` is "http://msalx.vejasp.abril.com.br/2012/10/08/1103/jcclk/parque-villa-lobos-3.jpeg", and `usuario` is 5779ae8c59f2471f13651ede.

```
2016-07-14 02:26:01.545 Exemplo1_rest_iOS[9015:436199] {
    "_v" = 0;
    "_id" = 577024e4a44821110001ee93;
    cidade = "S\u00e3o Paulo";
    datacriacao = "2016-06-26T18:54:28.872Z";
    estado = SP;
    nome = "Parque Villa Lobos";
    urlfoto = "http://msalx.vejasp.abril.com.br/2012/10/08/1103/jcclk/parque-villa-lobos-3.jpeg";
    usuario = 5779ae8c59f2471f13651ede;
}
```

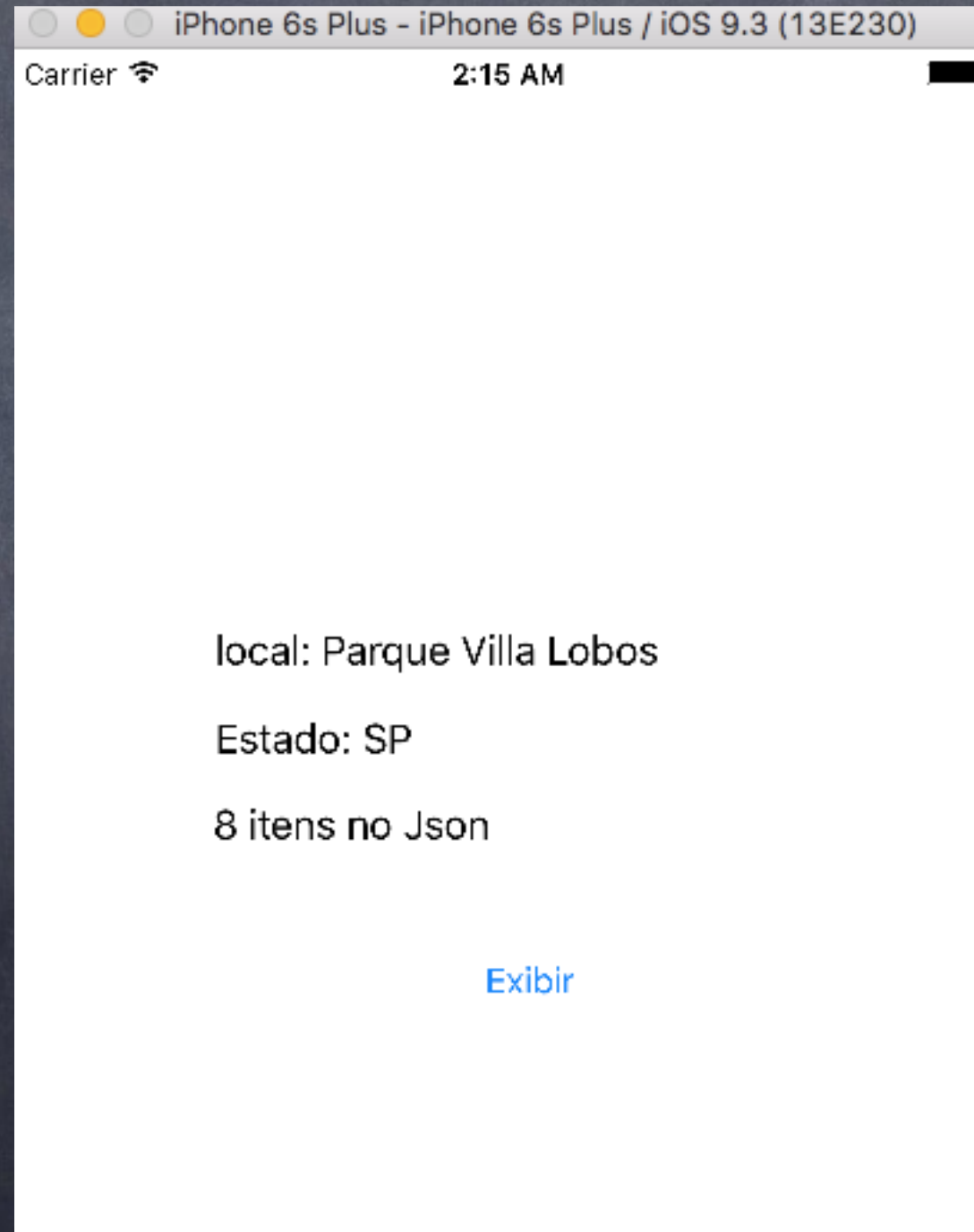

Implementando o Exibir

- Comente o NSLog(1), e implemente o trecho abaixo (2) faz a leitura do dicionário acessando os dados via chave e os passando para os respectivos outlets.

```
27 - (IBAction)exibir:(id)sender {
28     //acesso a API feita em PHP criada em meu servidor na Godaddy
29     //NSURL *url=[NSURL URLWithString:@"https://www.scorpioni.com/webservices/local.php/?id=1"];
30     //acesso a API feita em Node.JS criada no heroku
31     NSURL *url=[NSURL URLWithString:@"https://parks-api.herokuapp.com/parks/577024e4a44821110001ee93"];
32
33     //criando o NSData e fazendo um request
34     NSData *data = [NSData dataWithContentsOfURL:url];
35
36     //transformando o NSData para NSDictionary, o NSJSONSerialization faz o Parser do Json
37     NSError *error;
38     NSDictionary *json =[NSJSONSerialization JSONObjectWithData:data options: kNilOptions error:&error];
39
40     //caso precise, isto vai exibir o conteúdo do json, assim é possível
41     //identificar os nomes das chaves do dicionário.
42     //NSLog(@"%@", json); ← 1
43
44     //fazendo a leitura do dicionário
45     local.text = [NSString stringWithFormat:@"local: %@", [json objectForKey:@"nome"]];
46     estado.text = [NSString stringWithFormat:@"Estado: %@", [json objectForKey:@"estado"]];
47     NSString *texto = [NSString stringWithFormat:@"%lu itens no Json", (unsigned long) [json count]];
48     qtdInf.text = texto;
49 }
```


Implementando o Exibir

- Command + R e clique em Exibir para carregar o local e o estado.



Implementando o Exibir

- As linhas abaixo servem para carregar a URL de onde vem a imagem e exibi-la no imageView.

```
49
50 //imagem.image = [UIImage imageNamed:@"corinthians.png"];
51 NSString *urlFoto = [json objectForKey:@"urlfoto"]; //foto de um caminho na internet
52 NSData *data2 = [NSData dataWithContentsOfURL:[NSURL URLWithString:urlFoto]];
53 UIImage *foto = [[UIImage alloc] initWithData:data2];
54 imagem.image = foto;
55
```


Info.plist (forma 1) FIAP

- Antes de executar configure o ATS para poder abrir imagens em caminho http:, para isso abra o arquivo info.plist(1), clique no símbolo + (2), role as opções para encontrar: App Transport Security Settings (3), clique no símbolo do triângulo indicando-o para baixo (4), dessa forma é possível incluir um sub item, role as opções para encontrar: Allow Arbitrary Loads (5), selecione YES (6).

The screenshot illustrates the steps to configure App Transport Security (ATS) in an Xcode project. It shows the project navigator, the Info.plist file, and the settings pane.

Step 1: Select the `Info.plist` file in the project navigator.

Step 2: Click the `+` button at the bottom of the `Supported interface orientations` row in the `Info.plist` table.

Step 3: Select `App Transport Security Settings` from the dropdown menu.

Step 4: Click the triangle icon to expand the `App Transport Security Settings` dictionary.

Step 5: Select `Allow Arbitrary Loads` from the list of settings.

Step 6: Select `YES` for the `Allow Arbitrary Loads` setting.

Key	Type
Information Property List	Dictionary
Localization native development re...	String
Executable file	String
Bundle identifier	String
InfoDictionary version	String
Bundle name	String
Bundle OS Type code	String
Bundle versions string, short	String
Bundle creator OS Type code	String
Bundle version	String
Application requires iPhone enviro...	Boolean
Launch screen interface file base...	String
Main storyboard file base name	String
Required device capabilities	Array
Supported interface orientati...	Array

App Transport Security Settings (1 item)

Key	Type	Value
Allow Arbitrary Loads	Boolean	YES

Resultado

- Execute seu programa, e observe as telas, a segunda tela é do primeiro webservice em php a terceira tela é do segundo webservice em Node.JS.

