

Classes IOS – ObjC

Parte 3 – Exceptions e Protocolos

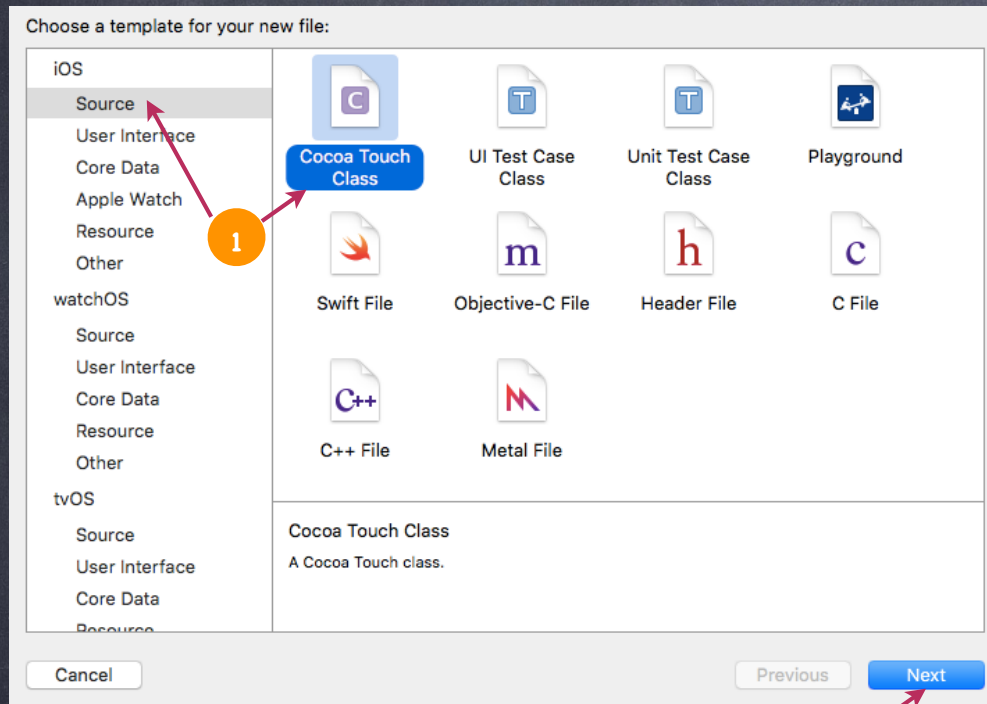
X-Code

Prof. Agesandro Scarpioni

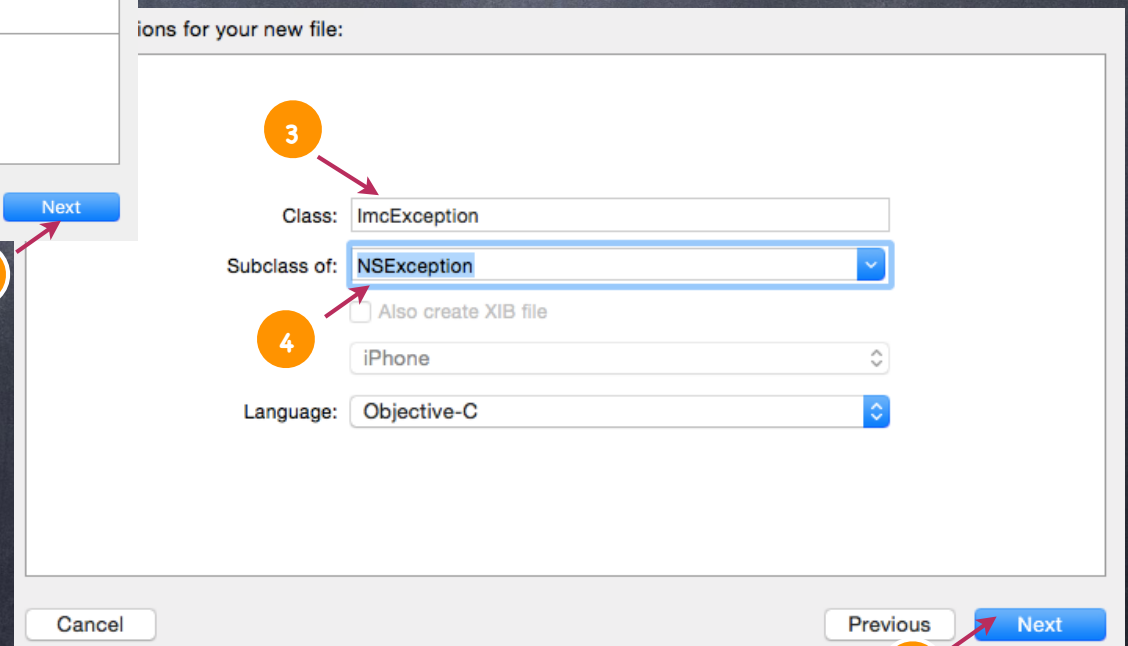
Exceptions

- A forma de criar uma exception em Obj-C é muito semelhante a forma de outras linguagens. Abra nosso programa da classe Atleta.
- Inclua mais uma Classe chamada ImcException filha de NSErrorException.

- Vamos criar uma classe "ImcException". Clique em File->New->File-> iOS -> Cocoa Touch -> Objective-C class.



- Nomeie a classe como ImcException subclasse de NSError



Exceptions

- Na classe Atleta.m faça o import da classe ImcException.h

```
1 //
2 // Atleta.m
3 // ClasseExemplo
4 //
5 // Created by agesandro scarpioni on 08/03/15.
6 // Copyright (c) 2015 Agesandro Scarpioni. All rights reserved.
7 //
8
9 #import "Atleta.h"
10 #import "ImcException.h"
11
12 @implementation Atleta
13
14 -(void) setName: (NSString *)_nome{
15     nome=_nome;
16 }
17 -(NSString *) getName{
18     return nome;
19 }
```


Exceptions

- Ainda na classe Atleta.m faça o lançamento da exception dentro do método CalcularImcComPesoeAltura.

```
29
30 - (void) calcularImcComPeso:(float) peso eAltura:(float) altura{
31     float imc;
32     if (altura > 2){
33         //throw exception
34         NSString *motivo=@"Altura não pode ser maior que 2 metros";
35         NSError *e = [ImcException exceptionWithName:@"ImcException" reason:motivo userInfo:nil];
36         @throw e;
37     }
38     imc = peso / (altura * altura);
39     NSLog(@" O IMC de %@ é %0.2f", self.getNome, imc);
40 }
41
```


Exceptions

- O método `exceptionWithName` serve para criar uma exception e iniciá-la com um nome e um reason, o reason será uma mensagem que explica o erro.
- O parâmetro `userInfo` é um `NSDictionary` do Foundation que representa uma estrutura de chave e código e pode conter mais informações sobre o erro.

```
29
30 - (void) calcularImcComPeso:(float) peso eAltura:(float) altura{
31     float imc;
32     if (altura > 2){
33         //throw exception
34         NSString *motivo=@"Altura não pode ser maior que 2 metros";
35         NSException *e = [ImcException exceptionWithName:@"ImcException" reason:motivo userInfo:nil];
36         @throw e;
37     }
38     imc = peso / (altura * altura);
39     NSLog(@" O IMC de %@ é %0.2f", self.getNome, imc);
40 }
41
```


Exceptions

- Na classe main.m faça o import da classe ImcException.h

```
1 //
2 // main.m
3 // ClasseExemplo
4 //
5 // Created by agesandro scarpioni on 08/03/15.
6 // Copyright (c) 2015 Agesandro Scarpioni. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10 #import "Atleta.h"
11 #import "ImcException.h"
12
13 1 int main(int argc, const char * argv[]) {
14     @autoreleasepool {
15
```


Exceptions

- Ainda na classe main.m vamos chamar um método com uma altura não permitida, por exemplo: 2.20m.

```
23
24 Atleta *a2 = [[Atleta alloc] initWithNome:@"Maria Mendes" andIdade:18];
25 NSLog(@"Iron Man %@ %d anos", [a2 getNome], [a2 getIdade]);
26 [a2 calcularImcComPeso:83.8 eAltura:1.87];
27 NSLog(@"%@", [a2 calcularSeuRendimentonaAguawithTempoemHoras:1.5 andMetros:8000]);
28
29
30 @try {
31     [a2 calcularImcComPeso:83.8 eAltura:2.20];
32 }
33 @catch (ImcException *exception) {
34     NSLog(@"Erro: %@", [exception reason]);
35 }
36 @finally {
37
38 }
39
```


Exceptions

- Utilizando o bloco try/catch/finally, chamamos o método com uma altura inválida e uma exceção será lançada.

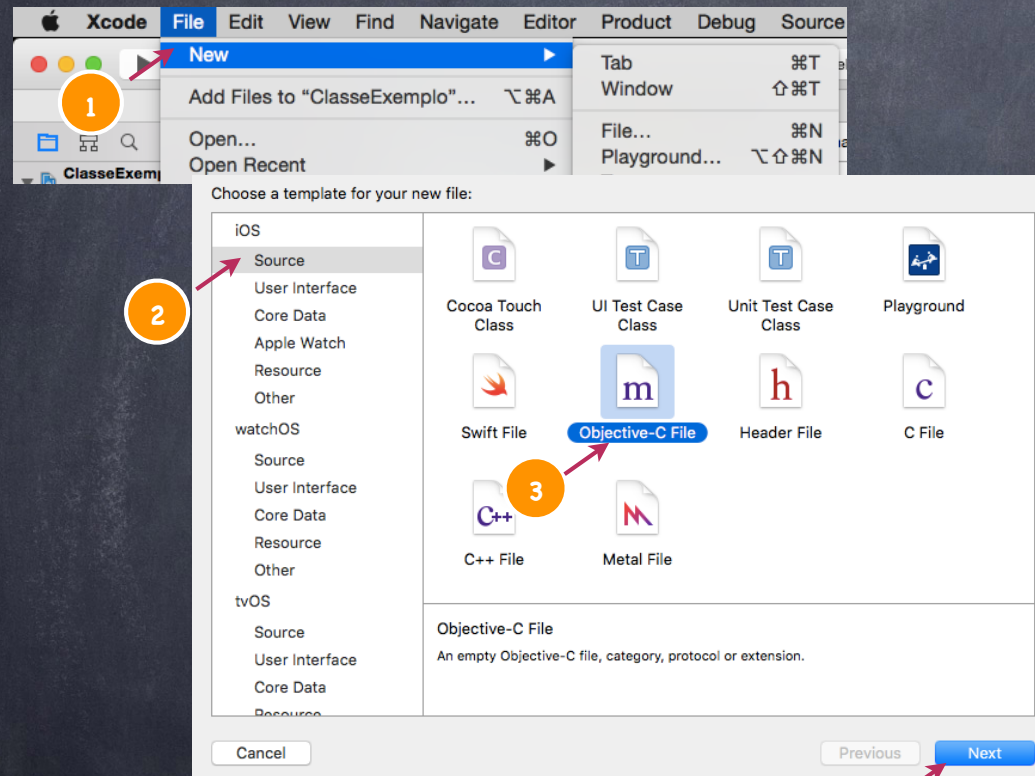
```
@try {  
    [a2 calcularImcComPeso:83.8 eAltura:2.20];  
}  
@catch (ImcException *exception) {  
    NSLog(@"Erro: %@", [exception reason]);  
}  
@finally {  
}
```


Protocolos

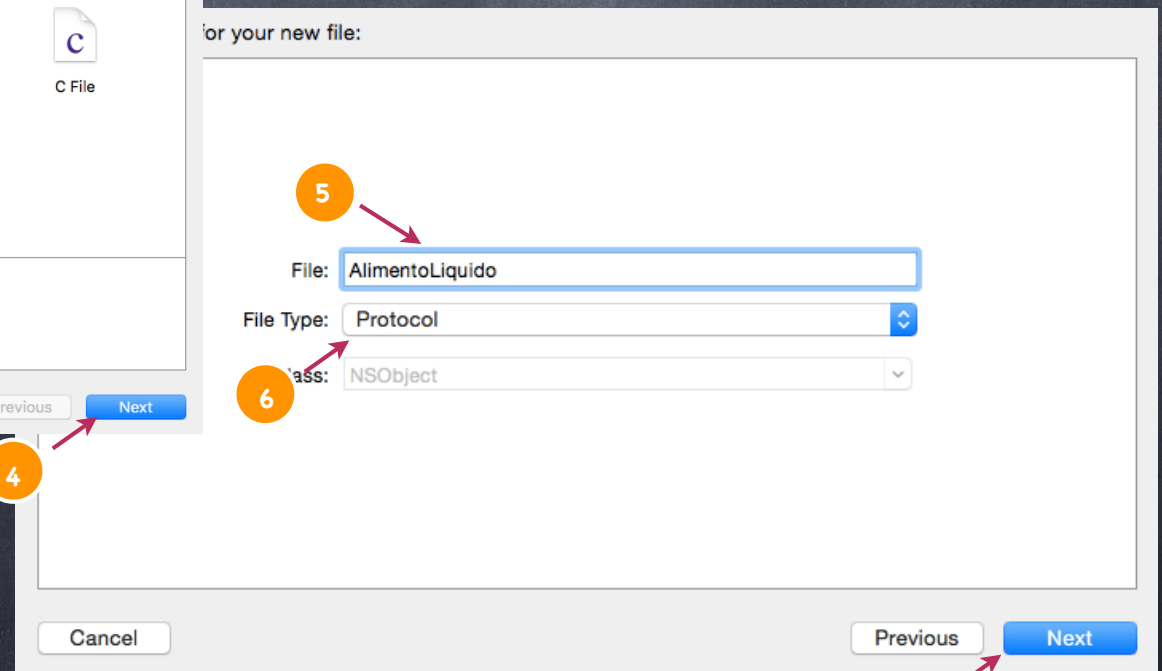
- O protocolo funciona como as interfaces do Java ou do VB ou aos métodos virtuais do C++, é declarado com a notação @protocol e uma classe pode implementar um ou mais protocolos, desde que quando tivermos mais que um protocolo, a declaração entre um protocolo e outro deve ser separado por vírgula, exemplo: <protocolo1,protocolo2, protocolo3>.
- O protocolo necessita apenas do arquivo .h, vamos criar 3 protocolos para a alimentação do atleta.

Protocolos

- Vamos criar 3 protocolos para alimentação do atleta, AlimentoLiquido, AlimentoSolido, AlimentoLiquidoeSolido, este último vai herdar os outros dois protocolos. Clique em File->New->File->IOS->Source->Objective-C File.

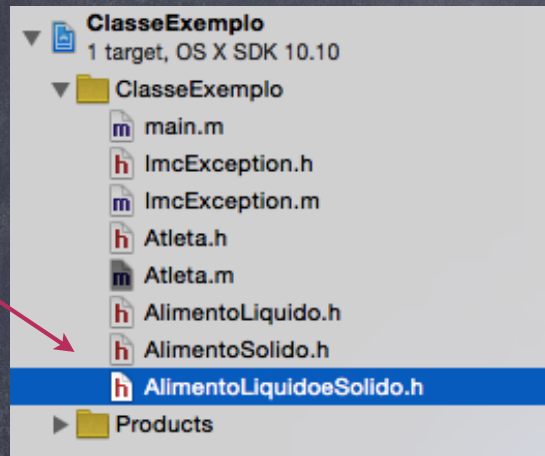


- Nomeie o primeiro como AlimentoLiquido escolha Protocol e repita o mesmo passo para os outros.



Protocolos

- Com os 3 protocolos criados como mostra na imagem 1, vamos declarar os métodos destacados nos pontos 2 e 3.



```

1 //
2 // AlimentoLiquido.h
3 // ClasseExemplo
4 //
5 // Created by agesandro scarpioni on 08/03/15.
6 // Copyright (c) 2015 Agesandro Scarpioni. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10
11 @protocol AlimentoLiquido <NSObject>
12
13 -(void) beberIsotonico;
14
15 @end
16

```

```

1 //
2 // AlimentoSolido.h
3 // ClasseExemplo
4 //
5 // Created by agesandro scarpioni on 08/03/15.
6 // Copyright (c) 2015 Agesandro Scarpioni. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10
11 @protocol AlimentoSolido <NSObject>
12
13 -(void) comerCarboidrato;
14
15 @end
16

```

```

1 //
2 // AlimentoLiquidoSolido.h
3 // ClasseExemplo
4 //
5 // Created by agesandro scarpioni on 08/03/15.
6 // Copyright (c) 2015 Agesandro Scarpioni. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10 #import "AlimentoLiquido.h"
11 #import "AlimentoSolido.h"
12
13 @protocol AlimentoLiquidoSolido <AlimentoLiquido, AlimentoSolido>
14
15
16 @end
17

```

Obs: Não esqueça dos dois imports no ponto 4. O protocolo AlimentoLiquidoSolido herda os métodos dos 2 protocolos, veja que no ponto 5 para implementar mais de um protocolo em uma classe separamos os nomes por virgula < __ , __ > .

Protocolos

- Para ser mais rápido vamos apenas implementar o protocolo: AlimentoLiquidoeSolido, vá para a classe Atleta.h. faça o import (1) e declare a interface AlimentoLiquidoeSolido (2).

```
1 //
2 // Atleta.h
3 // ClasseExemplo
4 //
5 // Created by agesandro scarpioni on 08/03/15.
6 // Copyright (c) 2015 Agesandro Scarpioni. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10 #import "AlimentoLiquidoeSolido.h"
11
12 @interface Atleta : NSObject <AlimentoLiquidoeSolido> {
13     NSString *nome;
14     int idade;
15 }
16
17
18 -(void) setName: (NSString *)_nome;
19 -(NSString *) getName;
20 -(void) setIdade: (int) _idade;
21 -(int) getIdade;
22
23 -(void) calcularImcComPeso:(float) peso eAltura:(float) altura;
```


Protocolos

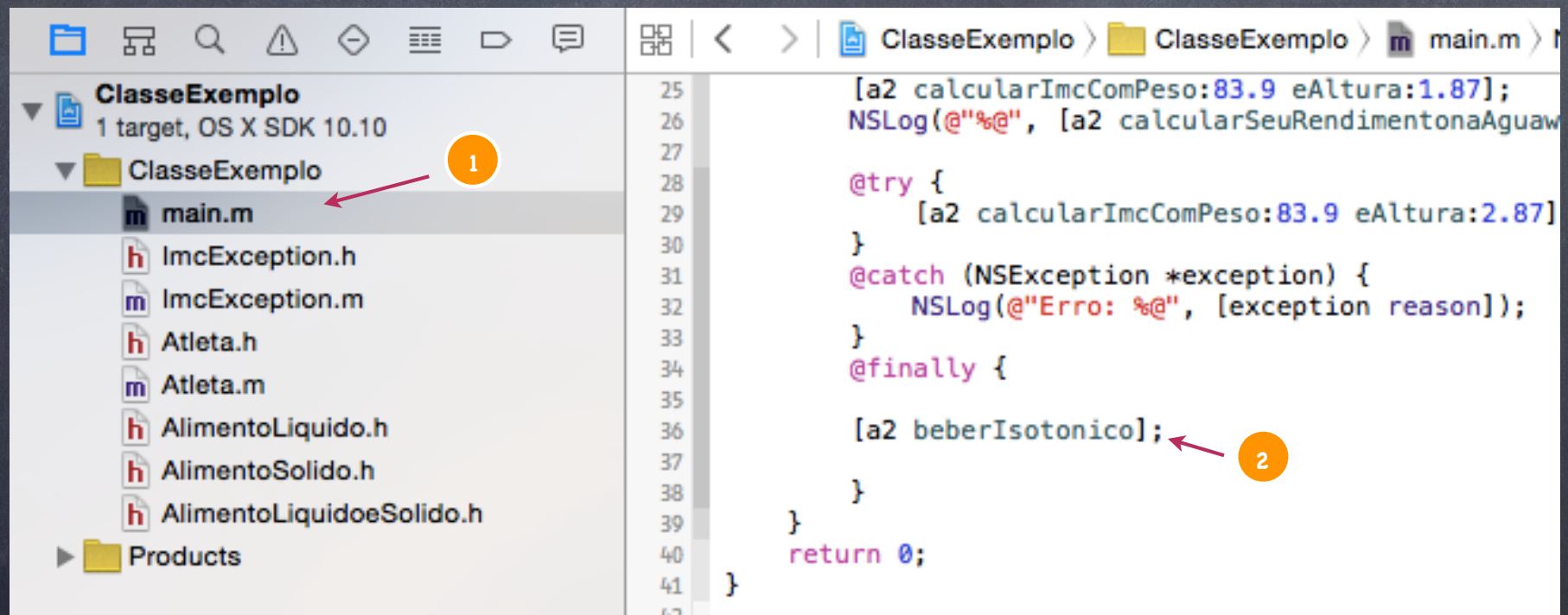
- Na classe `Atleta.m` vamos implementar os dois métodos do protocolo `AlimentoLiquidoeSolido`.

The screenshot shows the Xcode interface with the project structure on the left and the code editor on the right. The project structure includes a folder named 'ClasseExemplo' with files: 'main.m', 'ImcException.h', 'ImcException.m', 'Atleta.h', 'Atleta.m' (highlighted with a red arrow and a yellow circle with the number 1), 'AlimentoLiquido.h', 'AlimentoSolido.h', 'AlimentoLiquidoeSolido.h', and a 'Products' folder. The code editor shows the implementation of the 'Atleta' class in 'Atleta.m'. The code includes a header file, an initialization method, and two methods implementing the 'AlimentoLiquidoeSolido' protocol. Red arrows and yellow circles with numbers 2 and 3 point to the implementation of the 'comerCarboidrato' and 'beberIsotonico' methods, respectively.

```
44 }
45
46 -(Atleta *) initWithNome:(NSString *)_nome andIdade:(int)_idade{
47     self=[super init];
48     if (self) { // Se a inicialização foi ok
49         [self setName:_nome];
50         [self setIdade:_idade];
51     }
52     return self;
53 }
54
55 -(void) comerCarboidrato{
56     NSLog(@"Coma macarrão");
57 }
58
59 -(void) beberIsotonico{
60     NSLog(@"Beba água de coco para repor sais");
61 }
62
63 @end
64
```

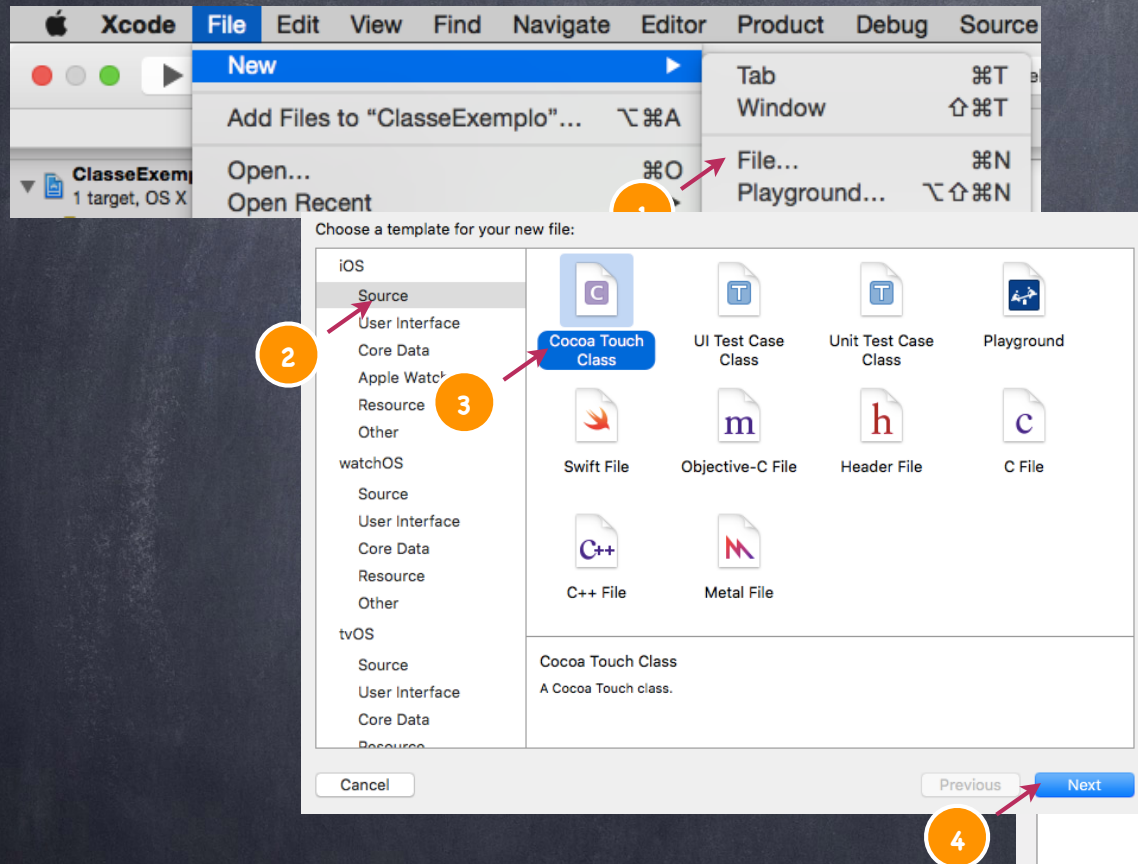

Protocolos

- Na main.m vamos testar nosso protocolo chamando um dos métodos.

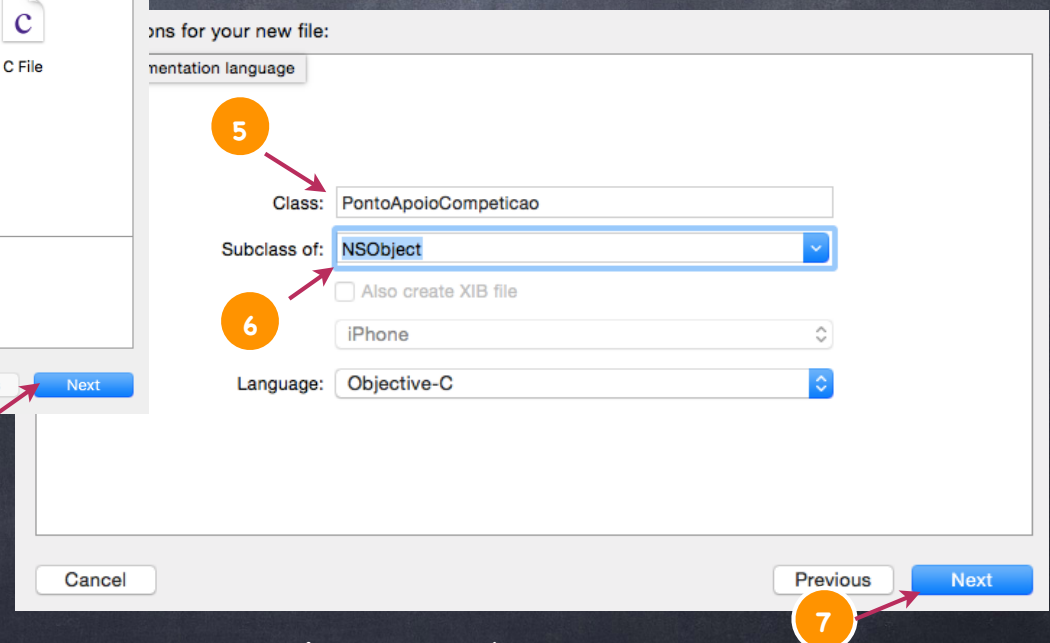


Classes + POO

- Vamos criar uma classe "PontoApoioCompeticao". Clique em File->New->File-> IOS -> Source -> Cocoa Touch Class -> Next.



- Nomeie a classe PontoApoioCompeticao como subclasse de NSObject



Dica: Esta classe será responsável por entregar um alimento do tipo líquido ou sólido para o atleta.

Classes + POO

- O símbolo + indica que são métodos da classe, ou seja, métodos estáticos.
- Na classe PontoApoioCompeticao.h vamos importar os dois protocolos (1) e declarar os dois métodos estáticos (2).

```
1 //
2 // PontoApoioCompeticao.h
3 // ClasseExemplo
4 //
5 // Created by agesandro scarpioni on 08/03/15.
6 // Copyright (c) 2015 Agesandro Scarpioni. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10 #import "AlimentoLiquido.h"
11 #import "AlimentoSolido.h"
12
13 @interface PontoApoioCompeticao : NSObject
14
15 + (void) entregarParaAtletaLiquidos:(NSObject <AlimentoLiquido> *) tipoLiquido;
16 + (void) entregarParaAtletaSolidos:(NSObject <AlimentoSolido> *) tipoLiquido;
17
18 @end
19
```


Classes + POO

- Na classe **PontoApoioCompeticao.m** vamos implementar os dois métodos estáticos. Abaixo, a forma de declaramos um parâmetro do tipo protocolo `NSObject <protocolo> *`

```
8
9  #import "PontoApoioCompeticao.h"
10
11 1 @implementation PontoApoioCompeticao
12
13  + (void) entregarParaAtletaLiquidos:(NSObject <AlimentoLiquido> *) tipoLiquido{
14      // delega para o método do protocolo
15      [tipoLiquido beberIsotonicos];
16  }
17
18  + (void) entregarParaAtletaSolidos:(NSObject <AlimentoSolido> *) tipoSolido{
19      // delega para o método do protocolo
20      [tipoSolido comerCarboidratos];
21  }
22  @end
23
```


Classes + POO

- A classe chamada PontoApoioCompeticao vai receber atletas com necessidades de alimentação dos tipos líquidos (isotônicos, vitaminas, proteínas, etc) ou sólidos (barras de cereais, barras de proteínas, carboidratos, etc) ou até mesmo dos dois tipos, lembre-se que apenas implementamos isotônico e carboidrato.
- A implementação dos métodos de PontoApoioCompeticao recebe objetos do tipo AlimentoSolido ou AlimentoLiquido (que são os protocolos) e apenas delega a chamada para os métodos comerCarboidrato e bebeIsotonico.

Classes + POO


- No main.m vamos importar a classe pontoApoioCompeticao (1), e chamar o método estático entregarParaAtletaSolidos(2).

```
1 //
2 // main.m
3 // ClasseExemplo
4 //
5 // Created by agesandro scarpioni on 08/03/15.
6 // Copyright (c) 2015 Agesandro Scarpioni. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10 #import "Atleta.h"
11 #import "ImcException.h"
12 #import "PontoApoioCompeticao.h" ← 1
13
14 int main(int argc, const char * argv[]) {
15     @autoreleasepool {
16
17         Atleta *a = [[Atleta alloc] init];
18         [a setName:@"José da Silva"];
19         [a setIdade:25];
20         NSLog(@"Iron Man %@ %d anos", [a getName], [a getIdade]);
21         [a calcularImcComPeso:83.9 eAltura:1.87];
22         NSLog(@"%@", [a calcularSeuRendimentonaAguawithTempoemHoras:1.5 andMetros:8000]);
23
24         Atleta *a2 = [[Atleta alloc] initWithNome:@"Maria Mendes" andIdade:18];
25         NSLog(@"Iron Man %@ %d anos", [a2 getName], [a2 getIdade]);
26         [a2 calcularImcComPeso:83.9 eAltura:1.87];
27         NSLog(@"%@", [a2 calcularSeuRendimentonaAguawithTempoemHoras:1.5 andMetros:8000]);
28
29         @try {
30             [a2 calcularImcComPeso:83.9 eAltura:2.87];
31         }
32         @catch (NSEException *exception) {
33             NSLog(@"Erro: %@", [exception reason]);
34         }
35         @finally {
36
37             ← 2
38             [a2 beberIsotonico];
39             [PontoApoioCompeticao entregarParaAtletaSolidos:a2];
40
41         }
42     }
43     return 0;
44 }
45
```


Classes + POO

- Veja o resultado dos dois últimos métodos.

```
2015-03-08 14:50:54.716 ClasseExemplo[1876:127011] Iron Man José da Silva 25 anos
2015-03-08 14:50:54.717 ClasseExemplo[1876:127011] O IMC de José da Silva é 23.99
2015-03-08 14:50:54.717 ClasseExemplo[1876:127011] O meu rendimento na água é 5333.33 metros por hora
2015-03-08 14:50:54.718 ClasseExemplo[1876:127011] Iron Man Maria Mendes 18 anos
2015-03-08 14:50:54.718 ClasseExemplo[1876:127011] O IMC de Maria Mendes é 23.99
2015-03-08 14:50:54.718 ClasseExemplo[1876:127011] O meu rendimento na água é 5333.33 metros por hora
2015-03-08 14:50:54.719 ClasseExemplo[1876:127011] Erro: Altura não pode ser maior que 2 metros
2015-03-08 14:50:54.719 ClasseExemplo[1876:127011] Beba água de coco para repor sais
2015-03-08 14:50:54.719 ClasseExemplo[1876:127011] Coma macarrão
Program ended with exit code: 0
```



Classes + POO

- Continuaremos com o tema @property e os exercícios práticos no próximo conjunto de slides.