Trường ĐH Khoa Học Tự Nhiên Tp. Hồ Chí Minh
TRUNG TÂM TIN HỌC

Lập trình iOS

Bài 4. *Protocol và Delegate*

Ngành Mạng & Lập trình thiết bị di động

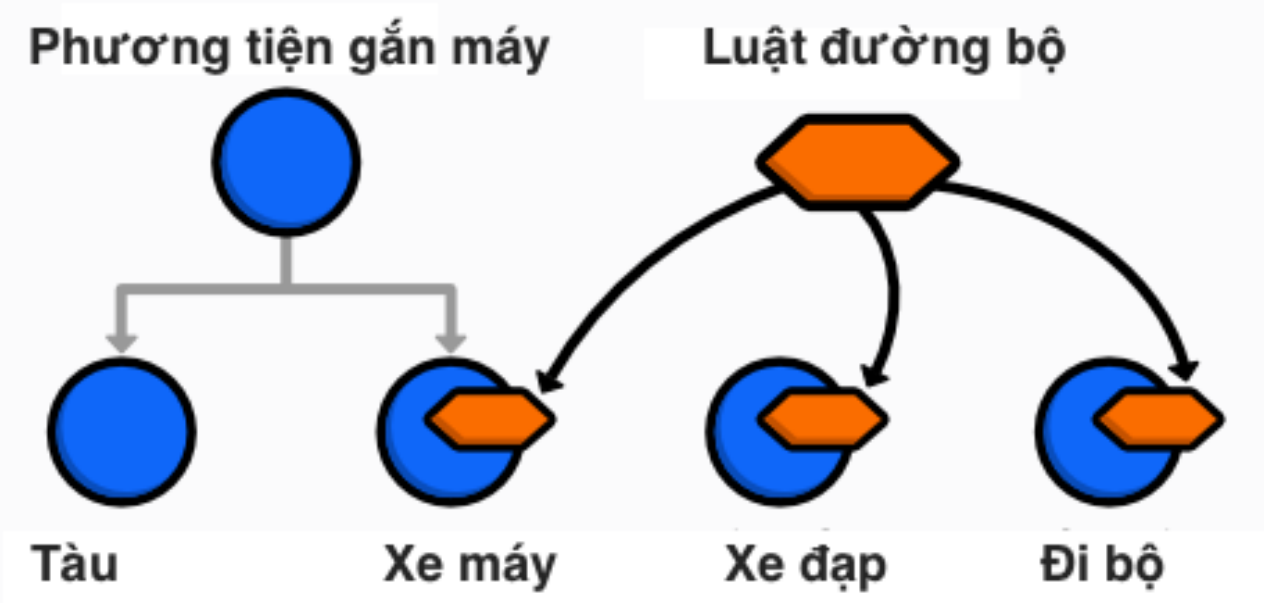
2015

Nội dung

1. **Protocol**
   - Khái niệm
   - Sử dụng Protocol trong lớp
   - Kiểm tra lớp sử dụng
2. **Delegate**

1.1 Khái niệm

- **Protocol được dùng để khai báo các thuộc tính và phương thức không phụ thuộc vào bất kỳ lớp nào. Tuy nhiên các lớp áp dụng protocol thì có thể phải định nghĩa các phương thức mà protocol đã khai báo:**
  - Ví dụ thực tế:



1.1. Khái niệm

- **Cú pháp để khai báo protocol:**

```
@protocol <Tên Protocol>
// khai báo các phương thức và thuộc tính tương tự như với lớp
@end
```

- **Protocol có hai khóa đó là @optional và @required để quy định các phương thức bắt buộc hoặc không bắt buộc triển khai ở các lớp áp dụng nó.**

- Ví dụ:

```
@protocol LuatDuongBo
@required
- (void)tinHieuDung;
@optional
- (void)tinHieuReTrai;
- (void)tinHieuRePhai;
@end
```

1.2 Sử dụng Protocol trong lớp

Ví dụ áp dụng protocol LuatDuongBo vào lớp XeDap:

```
// XeDap.h

#import "LuatDuongBo.h"

@interface XeDap : NSObject <LuatDuongBo>


@end
```

```
// XeDap.m

#import "XeDap.h"

@implementation XeDap

- (void)tinHieuDung {
    NSLog(@"Tín hiệu dừng");
}

- (void)tinHieuRePhai {
    NSLog(@"Tín hiệu được rẽ phải");
}

- (void)tinHieuReTrai {
    NSLog(@"Tín hiệu được rẽ trái");
}

@end
```

1.3 Kiểm tra lớp sử dụng

- **Có thể kiểm tra một lớp có sử dụng protocol hay không thông qua phương thức conformsToProtocol.**
  - Ví dụ: kiểm tra đối tượng phuongTien trước khi thực thi phương thức trong protocol.

```
id phuongTien = [[PhuongTien alloc]init];

[phuongTien tinHieuReTrai];

phuongTien = [[XeDap alloc]init];

if ([phuongTien conformsToProtocol:@protocol(LuatDuongBo) ]) {
    [phuongTien tinHieuDung];
    [phuongTien tinHieuRePhai];
}
```

Nội dung

1. Protocol

2. **Delegate**
   - Khái niệm
   - Sử dụng Delegate kết hợp Protocol

2.1 Khái niệm Delegate

- **Delegate trong Objective-C là một pattern để đối tượng A ủy nhiệm đối tượng B làm hộ việc gì thông qua các phương thức của protocol mà đối tượng B phải tuân thủ (adopt protocol).**

- **Có thể minh hoạ Delegate qua các bước sau:**
  - A uỷ thác đối tượng cho B.
  - B thực hiện tạo tham chiếu đến A.
  - A thực hiện phương thác uỷ thác khai báo trong B.
  - B thông báo kết quả thông qua phương thức uỷ thác

2.2 Sử dụng Delegate kết hợp Protocol

- Ví dụ tạo delegate cho lớp Nguoi, delegate là một đối tượng nào đó mà có áp dụng protocol DongVat

Protocol DongVat

```
#import <Foundation/Foundation.h>

@protocol DongVat <NSObject>
@required
- (void)diChuyen;
@end
```

```
#import <Foundation/Foundation.h>
#import "DongVat.h"

@interface Nguoi : NSObject

@property (nonatomic, weak) id<DongVat> delegate;
- (void)thucHienDiChuyen;

@end
```

Lớp Nguoi

2.2 Sử dụng Delegate kết hợp Protocol

- Triển khai phương thức **thucHienDiChuyen** cho lớp Nguoi như sau:

```
- (void)thucHienDiChuyen{
  //Nếu có đối tượng nhận ủy thác và đối tượng này có triển khai phương thức diChuyen
  if (self.delegate && [self.delegate respondsToSelector:@selector(diChuyen)]) {
    //Gọi đối tượng nhận ủy thác này thực hiện phương thức diChuyen
    [self.delegate diChuyen];
  }else{
    //nếu không có đối tượng nhận ủy thác thì xử lý như bình thường
    NSLog(@"Bằng hai chân");
  }
}
```

2.2 Sử dụng Delegate kết hợp Protocol

- Ta có thể thử sử dụng delegate tại phương thức viewDidLoad của lớp ViewController như sau:

```
18  - (void)viewDidLoad {
19      Nguoi *nguoi = [[Nguoi alloc]init];
20      Meo *meo = [[Meo alloc]init];
21      [meo diChuyen];
22
23      nguoi.delegate = self;
24  }
```

⚠ Assigning to 'id<DongVat>' from incompatible type 'ViewController *const __strong'

➔ Xcode sẽ nhắc nhở delegate này cần một đối tượng áp dụng protocol DongVat

2.2 Sử dụng Delegate kết hợp Protocol

- Ta sẽ áp dụng protocol DongVat cho lớp ViewController:

```
#import "ViewController.h"
#import "Nguoi.h"
#import "Meo.h"
@interface ViewController ()<DongVat>

@end

@implementation ViewController

- (void)viewDidLoad {
    Nguoi *nguoi = [[Nguoi alloc]init];
    Meo *meo = [[Meo alloc]init];
    [meo diChuyen];

    nguoi.delegate = self;
}
```

2.2 Sử dụng Delegate kết hợp Protocol

- Triển khai phương thức **diChuyen** trong lớp **ViewController** và gọi phương thức **thucHienDiChuyen** thông qua đối tượng **nguoi**

```
@implementation ViewController

- (void)viewDidLoad {
    Nguoi *nguoi = [[Nguoi alloc]init];
    Meo *meo = [[Meo alloc]init];
    [meo diChuyen];
    //self sẽ nhận ủy thác từ đối tượng nguoi
    nguoi.delegate = self;
    [nguoi thucHienDiChuyen];
}

- (void)diChuyen{
    NSLog(@"Bằng xe máy");
}
```

Thảo luận