Protocols

-Devendranath

For Offline/ Online Training, reach me@ iPhoneDev1990@gmail.com

- **Definition: A protocol defines a blueprint of methods, properties, and other requirements that suit a particular task or piece of functionality.**
- **The protocol can then be adopted by a class, structure, or enumeration to provide an actual implementation of those requirements.**
- **Any type that satisfies the requirements of a protocol is said to conform to that protocol.**

Syntax:

```
protocol SomeProtocol {
    // protocol definition goes here
}

//Protocol Adoption
class ClassName: SampleProtocol
{
    // Regular methods implementation
    // Protocol methods implementation
}
```

- **Protocols does contain Properties**
- **Protocol doesn't specify the property is stored or computed, It just specifies the property is gettable or gettable and settable**
- **Protocol does contain type properties. Those are preceded with static keyword**

```
protocol SomeProtocol {
    var mustBeSettable: Int { get set }
    var doesNotNeedToBeSettable: Int { get }
    static var someTypeProperty: Int { get set }
}
```

Example

```
protocol FullyNamed {
    var fullName: String { get }
}

struct Person: FullyNamed {
    var fullName: String
}
let john = Person(fullName: "John Limkhan")
print(john.fullName) //John Limkhan
```

Example

```
protocol FullyNamed {
    var fullName: String { get }
}

class Starship: FullyNamed {
    var prefix: String?
    var name: String
    init(name: String, prefix: String? = nil) {
        self.name = name
        self.prefix = prefix
    }
    var fullName: String {
        return (prefix != nil ? prefix! + " " : "")
+ name
    }
}
var org = Starship(name: "Enterprise", prefix:
"USS")
print(org.fullName) // USS Enterprise
```

Methods in Protocols

- Protocol declares set of methods without curly braces or a method definition
- Protocol methods can be instance or type methods
- Use static when declaring type methods

Example:

```
protocol SomeProtocol {
    static func someTypeMethod()
    func someInstanceMethod()
}
```

Methods in Protocols

```
protocol SomeProtocol {
    static func someTypeMethod()
    func someInstanceMethod()
}

class ProtocolAdoptor: SomeProtocol
{
    static func someTypeMethod() {
        print("This is some Required method")
    }

    func someInstanceMethod() {
        print("This is some instanc method")
    }

    func regularInstanceMethod()
    {
        print("This is regular instanc method")
    }
}

var anInstance = ProtocolAdoptar()
anInstance.someInstanceMethod() //This is some instanc method

anInstance.regularInstanceMethod() //This is regular instanc method

ProtocolAdoptar.someTypeMethod()//This is some Required method
```

Protocols can be inherited as follows

```
protocol InheritingProtocol: SomeProtocol, AnotherProtocol
{
    // protocol definition goes here
}
```

- **By default all properties and methods are required in Protocol**
- **You can declare optional properties and methods using @objc as follows**
- **The protocol which has optional properties or methods are only adopted by Classes.**
- **Optional Protocol can not be adopted by structures and Enums**

```
@objc protocol CounterDataSource {
    @objc optional func increment(forCount count: Int) -> Int
    @objc optional var fixedIncrement: Int { get }
}
```

Thanks

For Offline/ Online Training, reach me@ iPhoneDev1990@gmail.com