Revolutionizing iOS Development

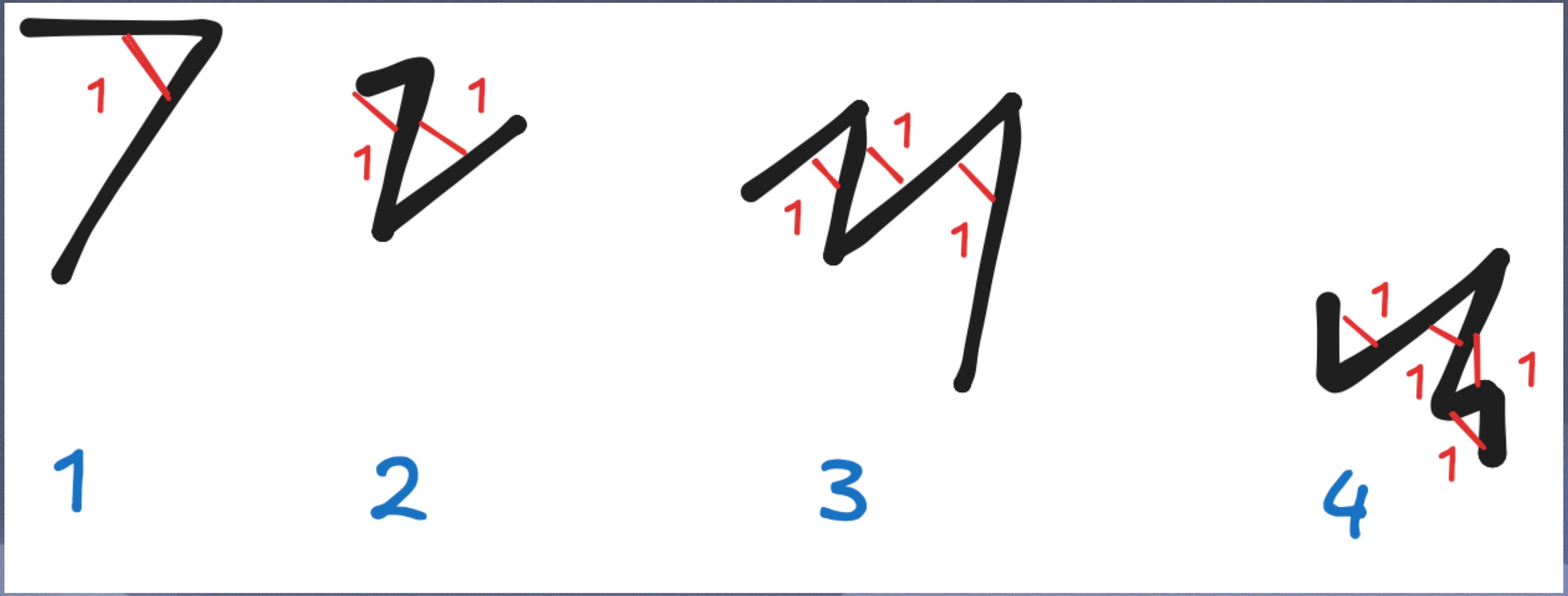
2013 → 2024

Dharmesh Avaiya | 10th Aug'24

Index

Why

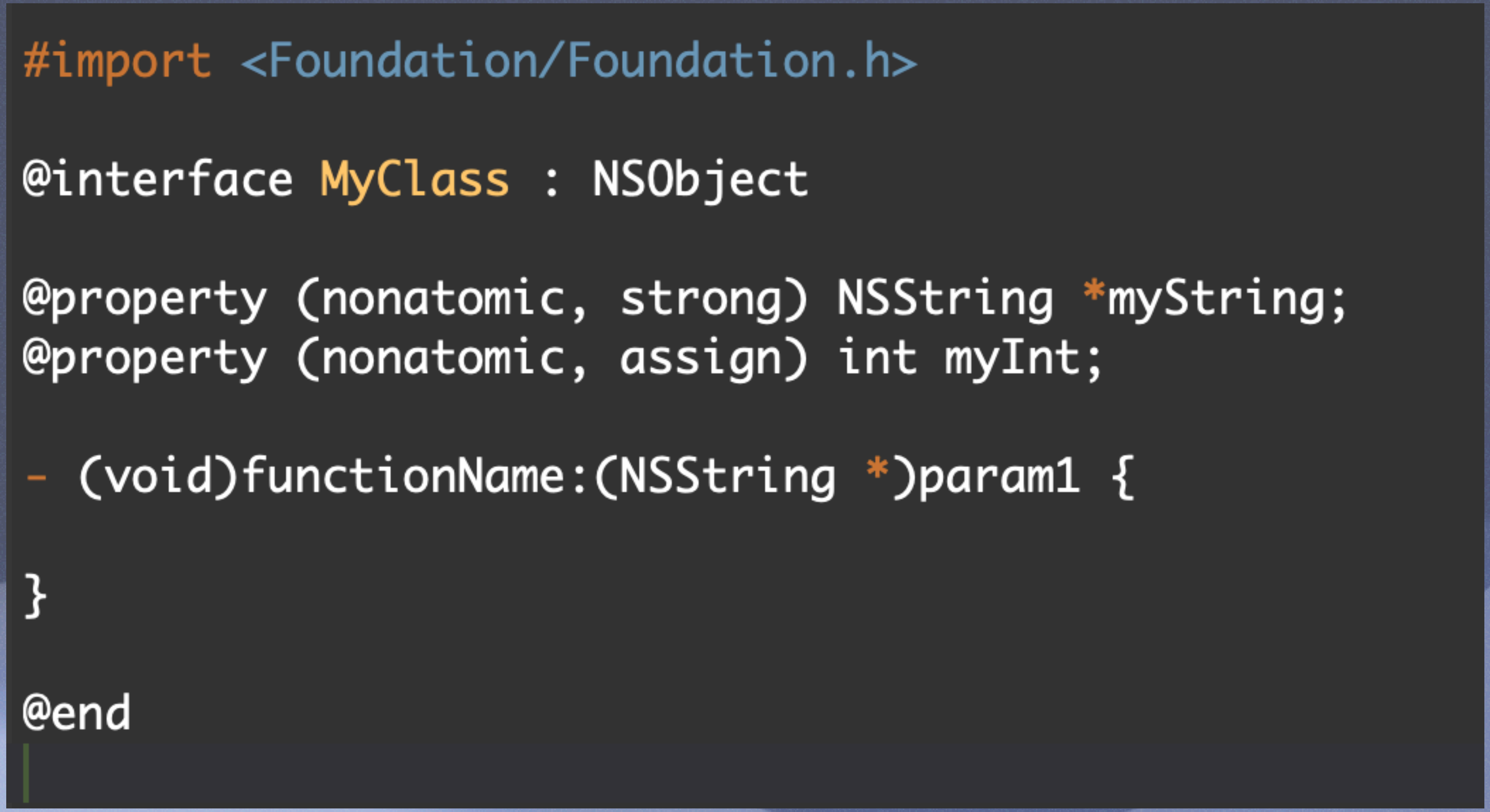
Why



Objective-C

- Two files .h and .m
- .h for declaration and .m for Implement
- Property (Nonatomic, Atomic, Strong, Weak, Assign, Readonly, Read-write, Copy, Getter, Setter etc.)
- XIBs / Storyboard (mid 2014)
- Without XIBs
- MacOS → AppKit/NSKit
- Mobile → UIKit

What are the challenges

- Complex Syntex → Developer Adaption
- Rapid Development → Cost
- 

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

@interface MyClass : NSObject

@property (nonatomic, strong) NSString *myString;
@property (nonatomic, assign) int myInt;

- (void)functionName:(NSString *)param1 {

}

@end
```

What are the challenges

- DRY Principle
- SOLID Principle → Not strongly supported, Objc is flexible & dynamic in nature
- Design Pattern Architecture → (MVC, MVP, MVP-C, MVVM, MVVM-C, VIPER, VIP) → Line of Code
- Server-side development
- WYSIWYG
- Default MVC Design pattern architecture

Swift

- Single file
- Simple Syntex
- Property (lazy, computed, willSet, didSet, optional, default, mutating etc.)
- XIBs/Without XIBs/Storyboard
- Service Side Swift Development

What we overcome with Swift

- Developer Adaption
- Cost
- DRY Principle
- SOLID Principle - Strongly Supported

What we still have not overcome with Swift

- AppKit/NSKit Cost
- Mobile → UIKit
- WYSIWYG
- Default MVC Design pattern architecture

Why SwiftUI

- Single UI Framework for all platforms (Including VisionPro)
- WYSIWYG
- RxSwift with Combine Framework

Give a Thought

- Actors
- Dependency Injection
- Associated Types
- CLEAN
- ONION
- HEXAGONAL

Have questions?

Thank you!!