# FAST CAMPUS ios dev school 과제 8

박상욱

**절차지향과 객체지향의 차이점**

**절차지향의 정의** : 순차적인 처리를 중요시하여 프로그램 전체가 유기적으로 연결이 될 수 있도록 만드는 프로그래밍 기법이다. -> 컴퓨터의 작업 방식과 비슷하다.
**객체지향의 정의**: 실제 세계를 모델링해 소프트웨어를 개발하는 프로그래밍 기법이다. 개발하고자 하는 것들을 모듈화 하여 언제든지 꺼내 사용할 수 있도록 함.

| 객체지향 | 절차지향 |
| --- | --- |
| 신뢰성 있는 소프트웨어를 쉽게 작성할 수 있음. (개발자가 만든 데이터 사용으로 신뢰 가능)<br><br>코드 재사용이 가능함<br><br>업그레이드 하기 쉬움<br><br>디버깅이 쉬움 | 데이터에 대한 접근이 객체지향 보다 빠르게 접근이 가능함<br><br>코드 사이즈를 최소화하고, 빠르게 동작하는게 최우선으로 작업하는 팀에 적합한 방법<br><br>c 로 구성하여 사용할때 주로 사용함 |

**클래스 vs 객체 vs 인스턴스 차이점**

클래스와 클래스 내 존재하고 있는 인스턴스의 차이점은 실체화다. 클래스는 항상 존재하지만, 인스턴스를 사용하기 위해선 별도의 인스턴스 식을 작성해야 하고, 그 식이 바로 클래스가 되는 꼴이다.

**Format Specifier (NSInterger, CGFloat, Nsstring) 조사하기**

**아직 제대로 알아보지 못함...**

String Format Specifiers

This article summarizes the format specifiers supported by string formatting methods and functions.

## Format Specifiers

The format specifiers supported by the NSString formatting methods and CFString formatting functions follow the IEEE printf specification; the specifiers are summarized in Table 1. Note that you can also use the "n$" positional specifiers such as %1$@ %2$s. For more details, see the IEEE printf specification. You can also use these format specifiers with the NSLog function.

**Table 1** Format specifiers supported by the NSString formatting methods and CFString formatting functions

**Table 2** Length modifiers supported by the NSString formatting methods and CFString formatting functions

## Platform Dependencies

OS X uses several data types—NSInteger, NSUInteger,CGFloat, and CFIndex—to provide a consistent means of representing values in 32- and 64-bit environments. In a 32-bit environment, NSInteger and NSUInteger are defined as int and unsigned int, respectively. In 64-bit environments, NSInteger and NSUInteger are defined as long and unsigned long, respectively. To avoid the need to use different printf-style type specifiers depending on the platform, you can use the specifiers shown in Table 3. Note that in some cases you may have to cast the value.

**Table 3** Format specifiers for data types

The following example illustrates the use of %ld to format an NSInteger and the use of a cast.

In addition to the considerations mentioned in Table 3, there is one extra case with scanning: you must distinguish the types for float and double. You should use %f for float, %lf for double. If you need to use scanf (or a variant thereof) with CGFloat, switch to double instead, and copy the double to CGFloat.

It is important to remember that %lf does not represent CGFloat correctly on either 32- or 64-bit platforms. This is unlike %ld, which works for long in all cases.