

# print, Dump

```
import UIKit

struct BasicInformation {
    let name: String
    var age: Int
}

var hoonInfo: BasicInformation = BasicInformation(name: "hoon", age: 99)

class Person {
    var height: Float = 0.0
    var weight: Float = 0.0
}

let hoon: Person = Person()
hoon.height = 175.2
hoon.weight = 73.3

print(hoonInfo)
dump(hoonInfo)

BasicInformation(name: "hoon", age: 99)
└─ __lldb_expr_9.BasicInformation
   └─ name: "hoon"
      └─ age: 99
         └─ __lldb_expr_9.Person
            └─ __lldb_expr_9.Person #0
               └─ height: 175.2
                  └─ weight: 73.3
```

스위프트 표준 라이브러리에는 `print()` 함수 외에도 `dump()`라는 함수가 있습니다. `print()` 함수는 디버깅 콘솔에 간단한 정보를 출력해주는 반면, `dump()` 함수는 조금 더 자세한 정보를 출력해줍니다. `print()` 함수는 출력하려는 인스턴스의 `description` 프로퍼티에 해당하는 내용을 출력해주고, `dump()` 함수는 출력하려는 인스턴스의 자세한 내부 콘텐츠까지 출력해줍니다. 필요에 따라 `print()` 함수 대신에 `dump()` 함수를 적절히 사용하는 것도 좋습니다. `print()`와 `dump()` 함수는 '부록 B. 스위프트의 주요 함수'에서 더 살펴볼 수 있습니다.

← 결과

dump는 더 자세하게 출력된다.

## Int, UInt

— 정수 type으로 Int는 +, - 부호를 포함 하는 정수를 뜻하고, 이 중 - 부호를 포함 하지 않는 0을 포함한 양의 정수는 UInt로 표현됨

— 각각 8, 16, 32, 64 비트의 형태가 있음.

— 시스템 아키텍처에 따라 Int와 UInt의 type이 달라짐.

↳ 32bit : Int32 → Int, UInt32 → UInt  
64bit : Int64 → Int, UInt64 → UInt

Int와 UInt의 한 변형에 사용되기

물리적으로 모든 Int의 저장공간(비트)은 항상 32비트인 Int32와 64비트 저장 공간 이상 Int64의 저장 공간을 사용하는 것은 불가능하다. 예를 들어 64비트 저장 공간과 Int64의 저장 공간을 사용하는 것은 32비트 저장 공간을 64비트로 늘려야 할 것이고, 이는 더 이상을 사용하는 것이 불가능하다. 또, 양수만 사용하는 것(즉, 양의 Int)을 고려할 때는 Int32와 UInt32는 대략 2의 제곱에 동등하게 동등하다. 같은 양의 Int32와 UInt32의 Int64를 변환하기 다른 타입으로 인식된다. 따라서, 32비트 Int와 UInt 두 타입을 모두 사용하는 경우 변환의 무효는 다뤄지지만 양수 타입의 Int와 UInt도 같은 크기를 가지는 Int64로 사용될 수 있다.

그림 3-1 Int와 UInt의 한 변형에 사용되기

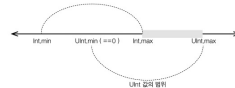


그림 3-1은 Int와 UInt 값의 범위를 나타내 보여줍니다. 그림에서 붉은 표시된 영역은 Int로 표현할 수 있는 값의 범위를 보여줍니다. Int의 저장공간은 작은 값의 범위입니다. 이 범위에 해당하는 값을 사용하고자 할 때 UInt를 사용하고, 그 외의 경우에는 Int를 사용해야 합니다.

```
1 import UIKit
2
3 var integer: Int = -100
4 let unsignedInteger: UInt = 50 // UInt 타입에는 음수값을 할당할 수 없음.
5 print("Integer 값: \(integer), unsignedInteger 값: \(unsignedInteger)")
6 print("Int 최댓값: \(Int.max), Int 최솟값: \(Int.min)")
7 print("UInt 최댓값: \(UInt.max), UInt 최솟값: \(UInt.min)")
8
9 let largeInteger: Int64 = Int64.max
10 let smallUnsignedInteger: UInt8 = UInt8.max
11 print("Int64 최댓값: \(largeInteger), UInt8 최댓값: \(smallUnsignedInteger)")
12
13 let tooLarge: Int = Int.max + 1 // Int의 표현 범위를 초과하므로 오류를 냄.
14 let cannotBeNegative: UInt = -5 // UInt는 음수가 될 수 없으므로 오류를 냄.
15
16 integer = unsignedInteger // 오류! 스위프트에서 Int와 UInt는 다른 타입임.
17 integer = Int(unsignedInteger) // Int 타입의 값으로 할당해야 할 수 있음.
18
19
```

integer 값: -100, unsignedInteger 값: 50  
Int 최댓값: 9223372036854775807, Int 최솟값: -9223372036854775808  
UInt 최댓값: 18446744073709551615, UInt 최솟값: 0  
Int64 최댓값: 9223372036854775807, UInt8 최댓값: 255

error: MyPlayground.playground:16:11: error: cannot assign value of type 'UInt' to type 'Int'  
integer = unsignedInteger // 오류! 스위프트에서 Int와 UInt는 다른 타입임.