1. **구조체와 클래스**

* 구조체와 클래스는 프로그래머가 데이터를 용도에 맞게 묶어 표현하고자 할때 유용
* 스위프트에서 구조체와 클래스는 문법이 비슷함
* 구조체의 인스턴스는 값타입 / 클래스의 인스턴스는 참조타입
* 구조체 안에 구조체, 클래스안에 클래스등 중첩타입이 가능
* 공식문서 스위프트에서는 클래스의 객체를 인스턴스라 부름
* 스위프트의 기본 데이터 타입은 모두 구조체이다.

1. 구조체

* struct 키워드
* 내부에 프로퍼티를 정의
* let으로 선언하면 프로퍼티 값을 변경할수 없고, var로 선언하면 변경가능

1. 클래스

* class 키워드
* : 이후에 부모클래스를 입력하여 상속이 가능
* 구조체와는 다르게 참조타입이므로 클래스 인스턴스를 let으로 선언해도 내부 프로퍼티 값을 변경할 수 있음
* **디이니셜라이져 deinit**
* 클래스를 더이상 참조할 필요가 없을때 deinit 메서드가 호출됨
* deinit 메서드는 클래스당 하나만 구현 가능

1. **구조체와 클래스의 차이**

* 같은점
* 프로퍼티를 정의할 수 있음
* 메서드를 정의할 수 있음
* 서브스크립트 문법을 통해 구조체 또는 클래스가 갖는 값(프로퍼티)에 접근하도록 서브 스크립트를 정의할 수 있음
* 이니셜라이저를 사용
* 익스텐션을 통해 확장 가능
* 특정 프로토콜을 준수 가능
* 차이점
* 구조체는 상속 할수 없음
* 타입캐스팅은 클래스 인스턴스만 가능
* deinit은 클래스만 사용가능
* Reference Counting은 클래스 인스턴스에만 적용
* 클래스는 참조타입, 구조체는 값타입
* 구조체는 전달인자로 값타입을 넘긴다면 전달될 값이 복사 되어 전달, 클래스는 전달인자로 값타입을 넘긴다면 참조 주소가 전달

1. 구조체와 클래스 선택해서 사용하기

* 애플가이드라인에서 조건의 하나이상에 해당하면 구조체를 사용하길 권장
* 연관된 간단한 값의 집합을 캡슐화 하는것이 목적일때
* 캡슐화한 값을 참조하는것보다 복사하는것이 합당할때
* 구조체에 저장된 프로퍼티가 값 타입이며 참조하는 것보다 복사하는 것이 합당할때

다른 타입으로부터 상속받거나 자신을 상속할 필요가 없을때