Class Diagram

- 설계 시 가장 많이 사용
- 객체의 타입인 Class를 표현하는 Diagram
- Name / Attributes / Operations
- 정적인 관계, 관계에 대한 제약 표
- Name // Name + Attributes // Name + Attributes + Operations // Name + Operations
- Visibility : +(public), #(protected), -(private), ~(package)
- 변수 기술 : name : type
- 함수 기술 : name(parameters):returnType, <constructor> <misc>등으로 분류
- 관련 클래스들의 적절한 분류를 패키지를 통해서 표현 가능

Boundary Class

- 사용자, **외부 시스템, 장비 등과 상호 작용**하는 클래스
- 초기에는 액터/유스케이스 한 쌍당 하나의 바운더리 클래스도 OK(이후 여러개로 분할)

Entity Class

- 시스템 관점에서의 **추상 개념**
- 유즈 케이스 기술서의 사건흐름으로부터 파악, **주요 추상 개념**을 참조
- 주로 명사
- **Wrapper**라는 스테레오타입을 정의해 DB연동에 사용

Control Class

- 유즈 케이스의 **행위를 조정**하는 클래스
- 하나의 유즈 케이스 - 하나의 컨트롤

Attributes

- [visibility] [/] name [:type] [multiplicity] [=default] [{property-string}]
- 밑줄 친 Attribute는 Static Attribute(Class의 모든 객체가 공유)

Associations

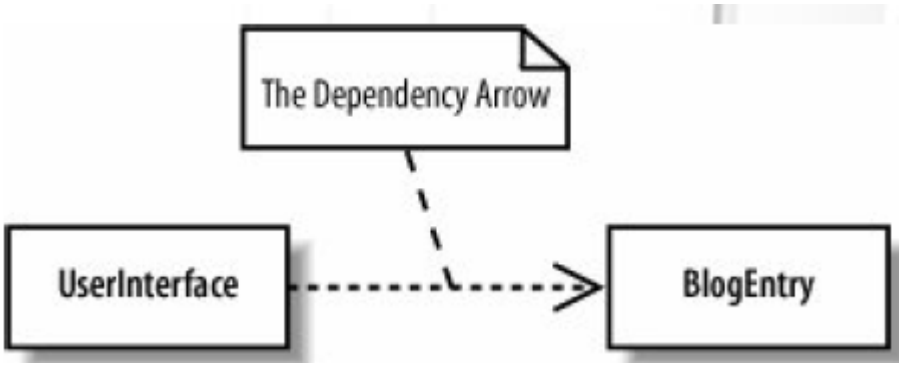
- Source / Target의 연결
- 속성의 이름은 역할으로 표현
- 연관관계의 양끝에 개수 표현

Operations

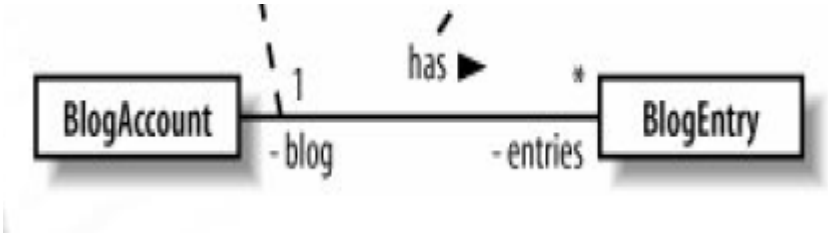
- [visibility] name ([parameter-list]) : [return-result] [{properties}]
- visibility는 private / public
- Static Operation은 밑줄

Relationships

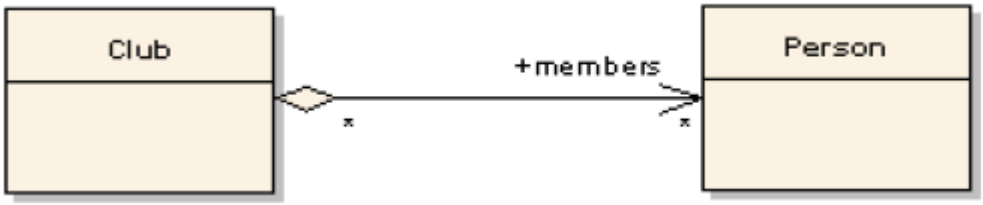
- 의존 관계 : 한 요소의 변화가 다른 요소에 영향을 미칠 경우 -> 가리키고 있는 클래스가 없으면 해당 클래스는 컴파일 불 가능



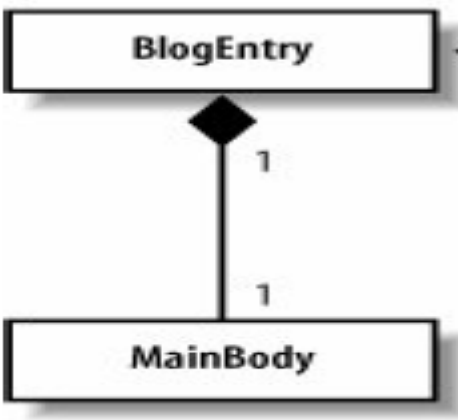
- 연관 관계 : 상대 객체와의 관계 표현, Has관계 등



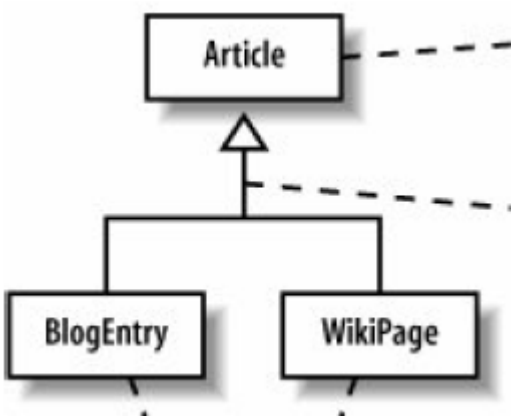
- 집합 관계 : Part-of 관계, 해당 객체쪽에 다이아몬드 그림, owns manages 등



- 구성 관계 : 전체-부분 관계(has a, contains, is part of) 분리되어 생각 될 수 없음(무조건 있어야 하는 것) -> 계층 구조를 가질 수 있으며, multiplicity를 가질 수 있음, Diagram 상으로 꽉찬 다이아몬드로 표현

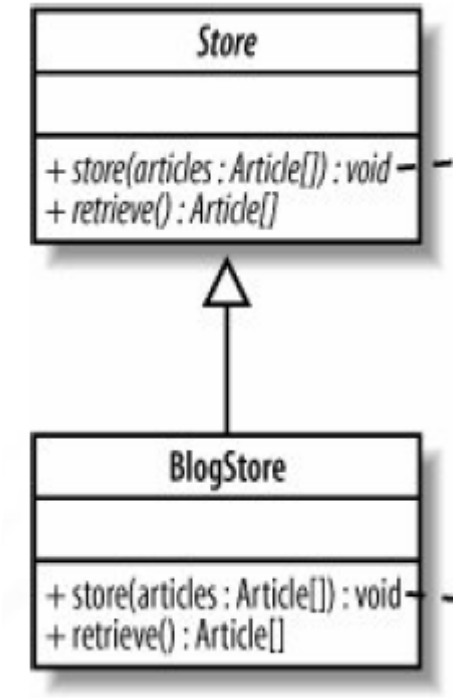


- 상속 관계 : superclass와 subclass 간의 논리적 추상화 제공(is a, is a kind of) -> subclass는 일반화된 suprclass의 모든 특성을 상속, 속이 빈 화살표로 표현(가리키는 쪽이 super)



추상 클래스

- 직접 인스턴스를 생성 할 수 없는 Class(하나 이상의 추상 오퍼레이션 보유) -> 유사품 추상 오퍼레이션 : 구현 없이 순수한 정의만을 가진 오퍼레이션
- Italic체로 표현



인터페이스

- 어떤 구현도 가지지 않는 클래스
- 모든 특성이 abstract
- 키워드 <<interface>>를 사용하여 표현
- 구현의 변경이 용이하고, 유지보수를 쉽게 함
- UML1 : 롤리팝으로 표현, 의존관계 사용
- UML2 : 의존관계 표현을 소켓 표기법으로 대체, class가 interface를 치환 가능

제약 조건

- Invariants (항상 true인 상수, class 속성에서 정의)
- Preconditions (메소드가 수행되기 전 만족하여야 하는 제약조건, 입력값의 유효성 검증에 사용)
- Postconditions (메소드가 수행 된 후 만족하여야 하는 제약조건, 출력값의 검증에 사용)