**UML(Unified Modeling Language)이란?**통합 모델링 언어라는 뜻으로**,** 즉 객체 지향 소프트웨어 엔지니어링 분야의   
표준화된 범용 모델링 언어를 말한다.

**UML**은 시스템의 모든 것을 문서화, 지정, 구축하는데 사용되는 표준언어이다**.**객체 지향 문제 해결에 적용할 수 있으며, 아홉 종류의 UML 다이어그램이 있다.

**UML의 정의**복잡한 소프트웨어 시스템 개발 모델링에 필요한 구성요소를 제시하고 이를 이용한 추상화 방법과 산출물들을 프로젝트 참여자들이 쉽게 이해할 수 있도록 소프트웨어 개발방법론(표현 및 기법)들이 통합된 객체지향개발 표준통합 모델링 언어

**UML의 궁극적 목적** - 사용자, 분석가, 설계자, 개발자 등의 개발을 위한 의사소통

**UML 필요 배경** - 시스템의 복잡성 → 표준적인 표기(notation)법 즉, 모델링 언어 필요

**​UML 관계 표기법** - 상속 (Inheritance)  
 - 구현 (Implements) - 의존관계 (Dependency) - 연관 (Association) - 합성 (Composition) - 집합(집약) (Aggregation)

**일반화 관계 Generalization**    - 부모와 자식 관계 (SuperClass 와 SubClass의 관계)    - 실선과 속이 비어있는 화살표로 표현

**연관 관계 Association**    - 구조적 관계, 한 쪽에서 다른 쪽을 사용하거나 참조할 수 있음    - 화살표가 있는 실선으로 표현, 역할을 가질 수 있음    - 다중성을 가질 수 있음    - 표현법  > 1    : 1개 **>** \*    : 0개 이상  
 > 1…\* : 1개 이상> 2,3  : 2개 or 3개  
    - 재귀적일 수 있음

**집합 연관 Aggregation**    - Association의 특수한 관계, 전체와 부분의 관계  
    - UML 2.0부터 사용하지 않음 -> Composition만 사용  
    - 다이아몬드와 실선

**합성 집합 연관 Composition**    - Association의 특수한 관계  
    - 전체와 부분의 관계 => 전체가 소멸되면 부분도 함께 소멸됨  
    - 채워진 다이아몬드와 실선

**연관 클래스 Association Class**    - 연관 자체가 여러 속성을 가질 때 그 속성을 간직할 연관 클래스를 둠  
    - 주로 두 클래스가 '다대다'의 관계로 맺어질 때 생성

**의존 관계 Dependency**    - 오퍼레이션의 매개 변수나 리턴 타입으로 사용될 때  
    - 점선의 화살표

**인터페이스 실체화 관계 Realization**  
    - 인터페이스와 그것을 구현한 클래스의 관계

**Javadoc이란?  
Java 소스에 문서화를 하는 방법**으로 클래스나 메서드에 주석으로 기술한 내용을 Javadoc 명령어나 또는 이를 이용한 빌드 툴을 사용하여 문서화할 수 있다. 문서화할 주석을 달 경우 /\*\*로 시작하고 \*/로 끝나야 한다. 특별한 키워드는 @keyword형식으로 작성한다.

**Javadoc 사용 이유**  
Javadoc을 통해 java 소스코드를 표준 주석등으로 문서화함으로써 **가독성이 높아지고 해당 코드(소스)에 대하여 파악하거나 분석하기 쉬워진다.** 이로써 시간을 아낄 수 있으며 분석을 잘못하는 경우도 없어진다.

**@author**-소스의 저자를 의미하며 이클립스에서는 기본 값으로 윈도우 계정id를 사용한다.

**@return**-메서드의 리턴(반환)값을 확인한다.

**@exception (or) throws**-메서드에서의 예외를 확인한다.

**@see**-다른 클래스나 메서드를 참고할 경우 사용한다.

**@param**-매개 변수에 대한 설명을 적습니다.

**@serial**-기본적으로 직렬화 할 수 있는 클래스의 멤버를 설명한다.