1. 소개

본 문서는 학생이 수강 신청 과정을 원활하게 이용할 수 있도록 하기 위한 Smart Time Table System 개발 과정 중 분석, 설계 단계에서의 결과물을 기반으로 한 소스 코드 구현 과정에 대하여 기술하고 있다.

1.1 목적

본 문서는 Smart Time Table System의 구현 과정을 상세히 기술한다. 설계 과정 중에서 다루지 않았던 GUI 관련 Class가 포함된 Sequence 및 Smart Time Table System의 핵심이라 할 수 있는 Smart 기능에 대한 Trace 과정을 설명한다.

특히 본 문서는 구현 단계에서의 생성물과 이전 분석 및 설계 단계에서의 생성물 사이에서의 차이점을 집중적으로 분석하여 이 분석 결과를 차후 Smart Time Table System의 유지 보수 및 신규 System개발에 적용함으로써 Software Product의 완성도를 높일 목적으로 작성되었다.

1.2 개요

본 문서의 2장에서는 구현 과정 중 사용한 UML Tool에 대한 설명을 다루고 있으며, 3장에서는 GUI 관련 Class를 포함한 전체 System의 Sequence에 대하여 기술한다. 4장에서는 본 Smart Time Table System의 핵심인 Smart 기능에 대한 분석, 설계, 구현 각 단계에서의 Trace를 기술하며, 5장에서는 구현 단계에서 최종적으로 완성된 System의 Class 구조와 기존 설계 단계에서의 Class 구조 사이에서 발생한 차이점에 대해 기술하고 이러한 차이점이 발생하게 된 요인을 각 Class별로 설명한다.

2. 사용 UML Tool

2.1 Visual Paradigm

2조는 무료 UML 툴을 검색하다가 적절한 프로그램을 찾아 내었다. UML은 기본이고, 디비 아키텍처 등 지원하는 것이 많이 있고 자료를 정리하기가 쉽게 되어 있어서 한번 익혀두면 두고 두고 써먹을 것 같은 Tool이다. 참고로, 무료로 사용하기 위해선 설치할 때, community 버전을 선택해야 한다. 그러지 않으면 시간제한이 있기 때문에 만료가 되면 이용하지 못한다.



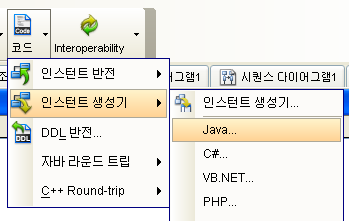
- Visual Paradigm 로고 -

2.2 Code Generator한 클래스

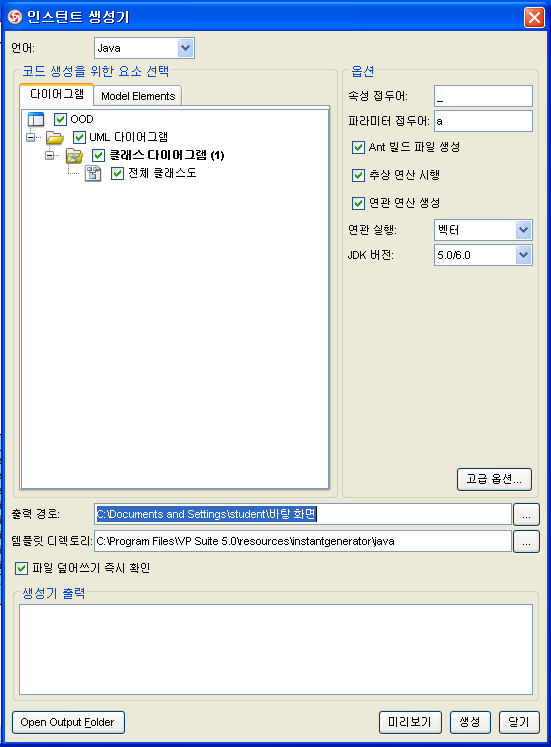
1. 메뉴 바에서 코드 부분을 클릭

3.bmp

2. 인스턴트 생성기 -> Java



3. 출력 경로를 설정

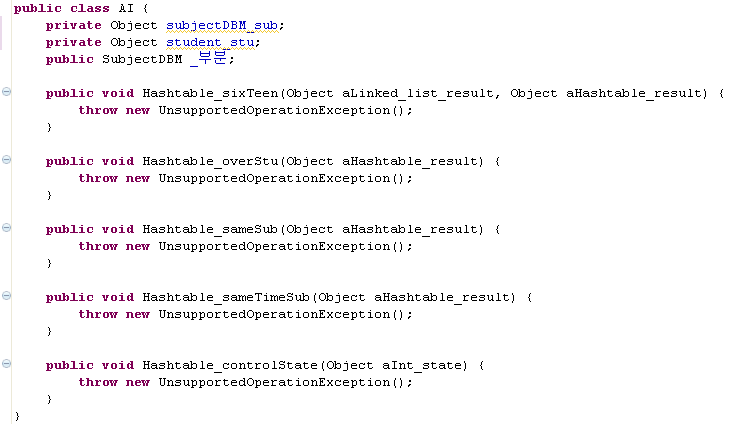


4. 파일이 생성됨

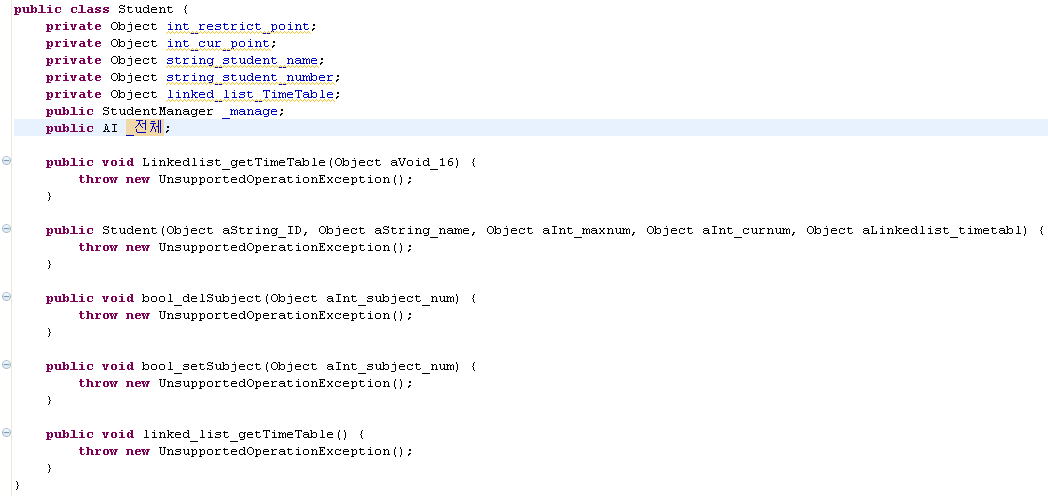


2.3 Code Generator한 Java 파일

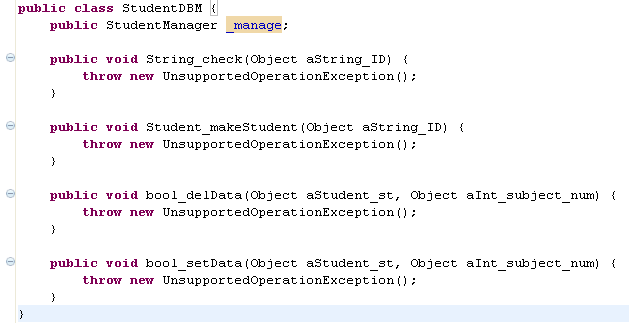
1. AI.java



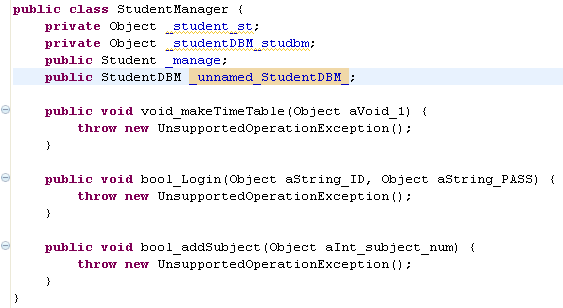
2. Student.java



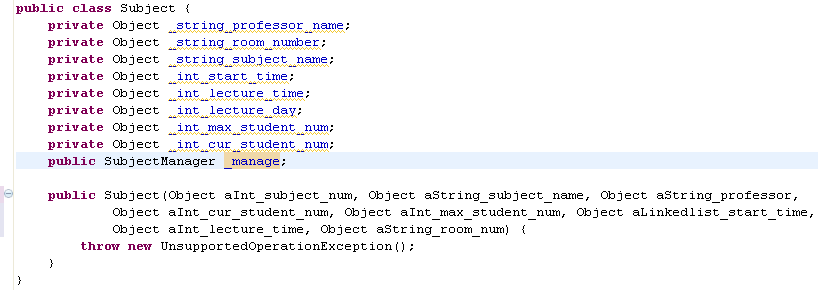
3. StudentDBM.java



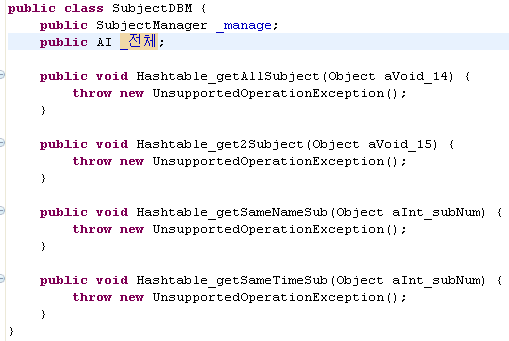
4. StudentManager.java



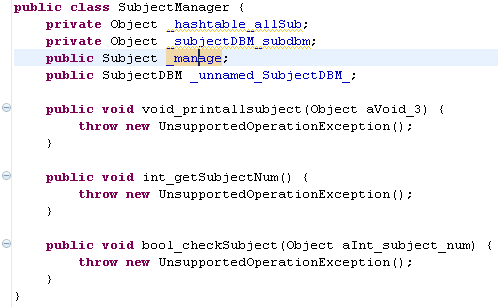
5. Subject.java



6. SubjectDBM.java



7. SubjectManager.java

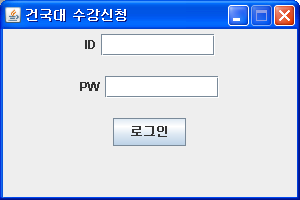


8. UI\_.java

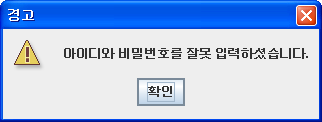


3. GUI 관련 Sequence

3.1 Class Login extends JFrame



- 로그인 실패시



- 로그인 성공시 sugang 화면

3.2 Class sugang extends Jframe



- 개인 시간표 보기 클릭시 Private 화면

- 수강 신청 클릭시 수강신청 실패 상태에 따라 Propose 화면

3.3 Class Propose extends Jframe



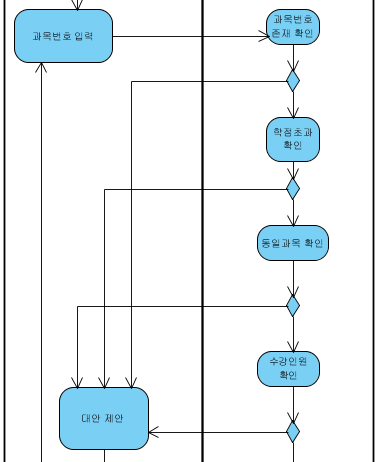
3.4 Class Private extends Jframe



4. Smart 기능의 Trace

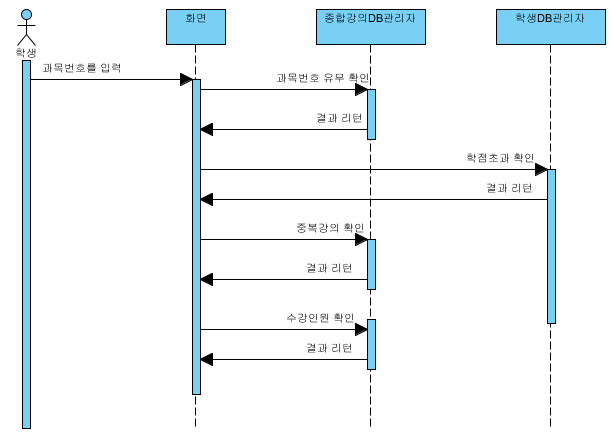
4.1 분석 단계

SRS 문서의 **<4.2.2. 수강신청>**에서 처음으로 언급이 되었다.



액티비티 다이어그램에서 AI(대안 제안)으로 가는 흐름에 대해 언급하였다.

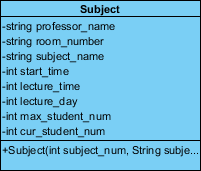
그 후 **<4.3 Sequence Diagram>**에서 AI의 진입 요건에 대해 막연하게나마 생각을 하게 되었다.



다이어그램에 명시되어 있지는 않지만 결과값 리턴에 따라서 대안 제안을 하는 AI을 호출할 수 있다고 팀원끼리 간략하게나마 상의를 하였다.

4.2 설계 단계

SDD **<4.2.2 Subject Class>**에서는 Subject class는 강의에 대한 정보를 포함하는 class이다. AI 기능은 예외 상황이 발생했을 시 수강 신청이 가능한 과목을 검색하여 추천해주는 기능이다. 그러한 예외 상황을 알 수 있게 하는 요소들은 Subject class의 변수들이다.



-String Subject\_name : 과목명을 저장.

-String room\_number : 강의실을 저장.

-int start\_time : 강의 시작 시간을 저장.

-int lecture\_time : 강의 시간을 저장.

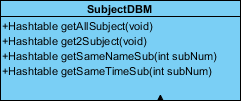
-int lecture\_day : 강의 날짜를 저장.

-int max\_student\_num : 최대 수강 인원을 저장.

-int cur\_student\_num : 현재 수강 인원을 저장.

위의 정보를 토대로 개인 강의 시간표를 구성하게 되며, 시간표와 겹치는 경우, 최대 수강 인원이 다 꽉 찬 경우 등 예외 상황을 계산하므로, 프로그램 실행을 위한 변수이지만, AI를 위한 변수들이기도 하다.

**<4.2.3 SubjectDBM Class>**에서는 AI 기능 중 후보 목록 강의들을 뽑아주는 함수들에 대해 언급되었다.



+Hashtable get2Subject(void) : 2학점인 강의 정보를 Hashtable 타입으로 return한다.

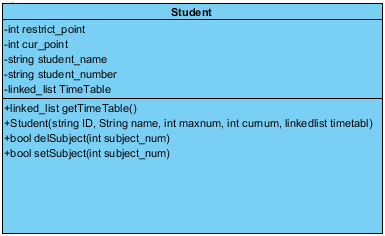
+Hashtable getSameNameSub(int subNum) : Parameter로 입력받는 int 타입의 변수 subNum를 key값으로 DB를 검색하여 동일한 과목이지만 다른 과목번호인 강의를 Hashtable 타입으로 return한다.

+Hashtable getSameTimeSub(int subNum) : Parameter로 입력받는 int 타입의 변수 subNum를 key값으로 DB를 검색하여 동일 시간대의 다른 과목들의 정보를 Hashtable 타입으로 return한다.

AI 기능은 수강 신청 학점이 18학점 일 경우, 현재 수강 신청 학점이 16학점이고 3학점 신청 시

2학점 강의를 추천해주므로 **<4.2.5. Student Class>**의 학생의 현재 수강 학점에 대해 관심이 있

을 것이다.



-int cur\_point : 해당 학생의 현재 수강신청 학점을 저장.

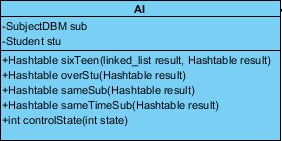
-int restrict\_point : 해당 학생의 제한 학점을 저장.

-string student\_name : 해당 학생의 이름을 저장.

-string student\_number : 해당 학생의 학번을 저장.

-linked\_list TimeTable : 해당 학생이 수강신청한 과목들을 저장.

**<4.2.7. AI Class>**에서는 본격적으로 AI의 설계를 시작하였다.



-SubjectDBM sub : SubjectDBM 클래스의 객체

-Student stu : Student 클래스의 객체

+Hashtable sixTeen(linked\_list result, Hashtable result) : Operator의 Parameter로 전달받는 linked\_list result는 학생이 개인 시간표로 출력할 때 쓰이는 Parameter이고, Hashtable result는 2학점인 강의 정보가 담겨있다. 이 두 개를 Parameter로 받아 sixTeen( ) Operator는 수강신청이 불가능한 시간대에 위치한 2학점 과목을 제외한 2학점 과목 목록을 Hash Table로 return한다.

+Hashtable overStu(Hashtable result) : Parameter로 받는Hashtable result는 사용자가 입력한 과목번호의 과목과 동일한 과목명을 가진 과목 중에서 수강 인원이 초과되지 않은 과목의 정보를 가지게 된다. 수강 인원이 초과되지 않은 과목 목록을 Hash Table로 return한다.

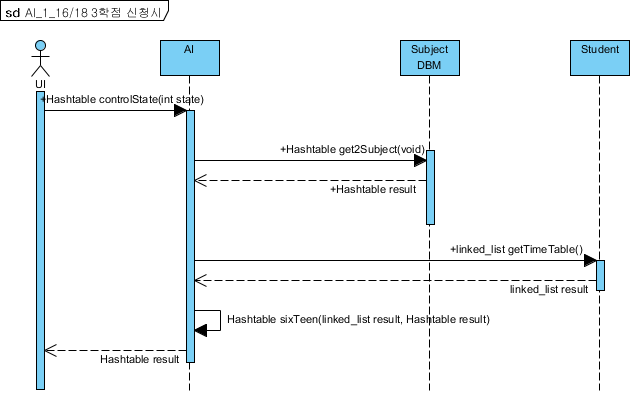
+Hashtable sameSub(Hashtable result) : Parameter로 받는 Hashtable result는 동일 시간대의 다른 과목의 정보를 담고있으며, sameSub( )Operator는 같은 시간대의 다른 과목의 정보를 Hash Table로 return한다.

+Hashtable sameTimeSub(Hashtable result) : Parameter로 받는 Hashtable result는 사용자가 입력한 과목번화의 과목명과 동일한 다른 시간대의 과목 정보를 담고 있으며, sameTimeSub( )Operator는 다른 시간대의 동일 과목의 정보를 Hash Table로 return한다.

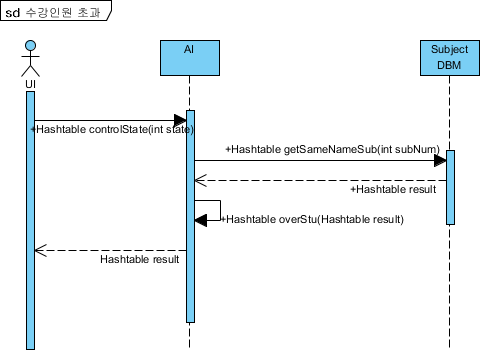
+Hashtable controlState(int state) : Parameter로 받는 int타입 변수 state는 controlState( ) Operator가 호출되기 이전의 상황에 따라 구분되어 받게 된다. 받은 Parameter 값에 따라서 위의 4가지 operation을 실행시키는 Operator가 되며 이 Operator들의 return 값을 다시 return해주는 역할을 한다.

**<5.2.1.6. AI의 Sequence Diagram>**에서는 AI의 4가지 경우에 대한 시퀀스 다이어그램에 대해 언급되었다.

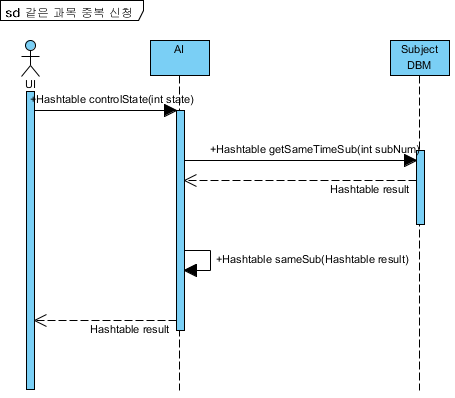
A. 학생 개인의 제한 학점이 초과된 상태에서 수강신청 시



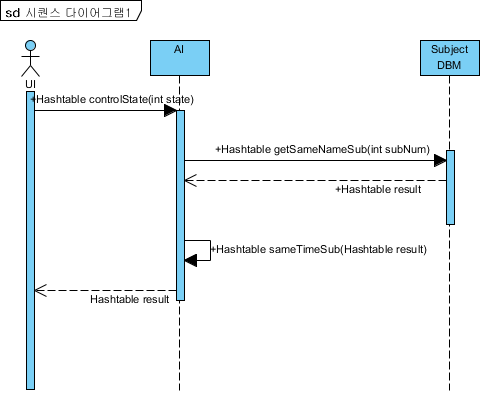
B. 과목의 수강 인원 한도가 초과된 상태에서 수강신청 시



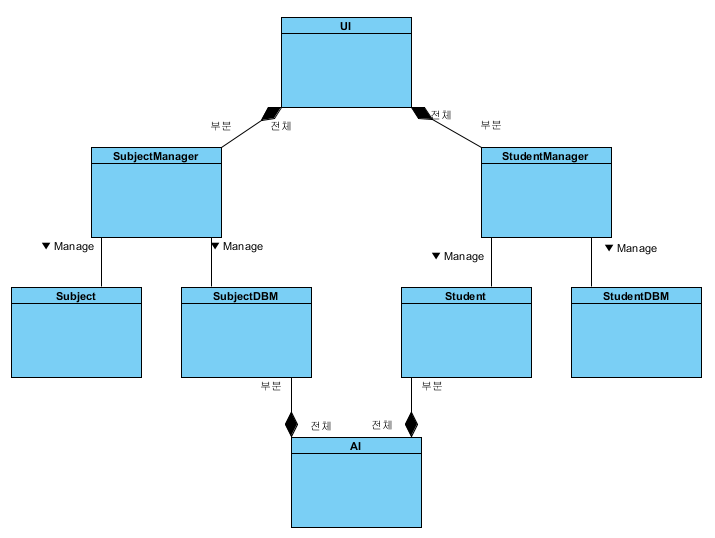
C. 과목 중복 신청 시



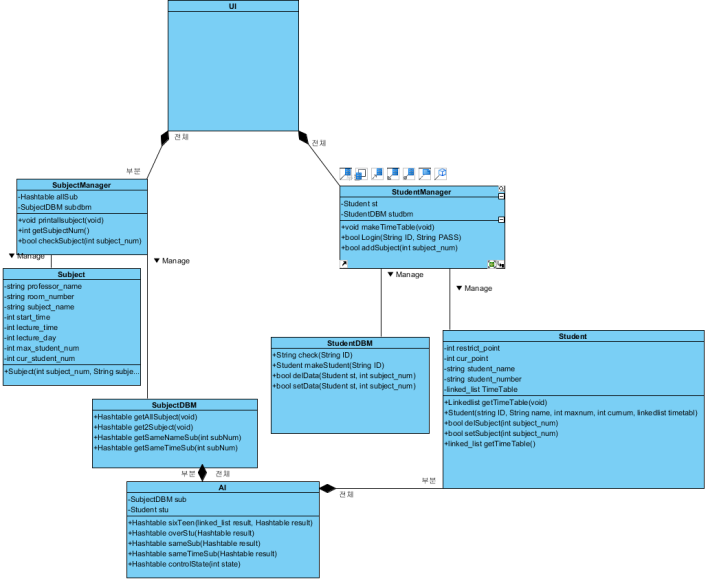
D. 동일한 시간대의 서로 다른 두 과목을 중복하여 수강신청 시



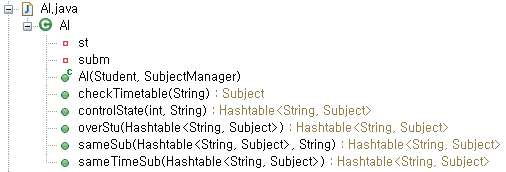
**<5.2.2 Class Dependency>**에서는 AI 클래스와 다른 클래스 간의 관계에 대해 나타나고 있다.



마지막으로 설계에 결과물이라 할 수 있는 **<5.3.3 Class Diagram>**에서 언급되었다.



4.3 구현 단계



+ControlState(int, String) method는 문제의 상태를 구분한다.

1. +overStu(hashtable<String, Subject>) 해당 과목의 수강인원이 초과된 상태

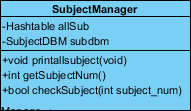
2. +sameSub(Hashtable <String,Subject>,String) 이미 수강된 과목을 다시 수강하는 상태

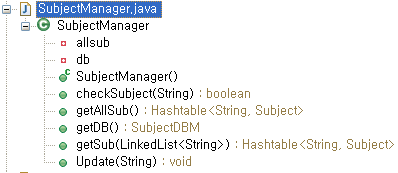
3. +sameTimeSub(Hashtable<String,Subject) 이미 수강된 시간에 다시 수강을 하는 상태

\* +checkTimetable(String)은 현재 자기 시간표와 제안해주는 Subject자료들과 겹치는지를 검사한다.

5. 설계-구현 단계 간 Class 구조의 차이점

5.1 Class SubjectManager



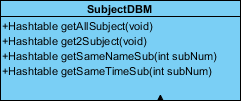


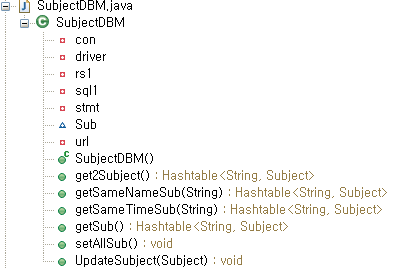
+Get~ method들은 캡슐화의 결과로 나온 부산물들이다.

+Update(String)의 해당 과목번호(string)의 내용을 수정을 하는 메소드가 추가되었다

+checkSubject(int)는 +checkSubject(string)으로 데이터베이스의 데이터 형식이 int에서 string으로 바뀜에 따라 method도 변경되었다.

5.2 Class SubjectDBM



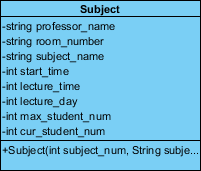


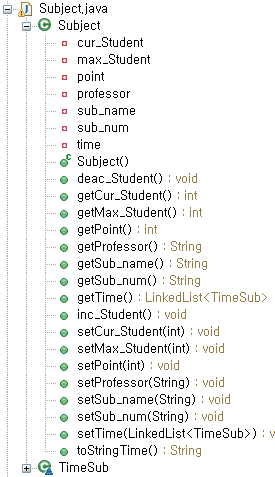
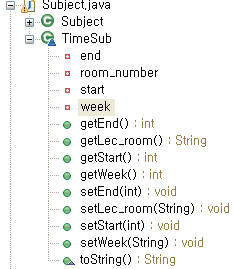
con, driver, rs1, sql1, stmt, url은 JAVA에서 JDBC를 다루기 위한 변수들이다.

+Set~ method, Get~method 들은 캡슐화의 결과로 나온 부산물들이다.

+UpdateSubject(Subject)는 SubjectManager class에서 +Update(string) method 호출시 사용되며, 해당 과목번호의 내용을 수정하는 부분이다.

5.3 Class Subject





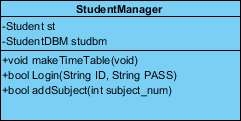
설계 단계에서의 Suject class가 Subject, TimeSub class로 나뉘었다. 기존의 Subjec class에서 시간과 강의실에 관련된 멤버들을 TimeSub class에서 다루게 하였고, Lecture\_time(강의 시간)과 start\_time(강의 시작 시간)을 변수를 사용하는 대신에 start(강의 시작시간)와 end(강의 종료시간)를 사용하였다. 요일을 저장하는 변수 Day의 이름을 week로 변경하였다.

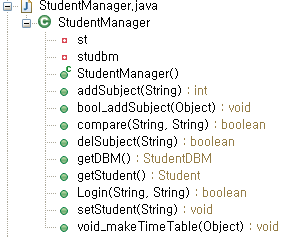
해당 과목의 학점을 저장하기 위한 point 변수와 Timesub Object의 Linked list인 time 변수를 추가하였다.

수강 시 해당 과목의 수강인원을 변경하기 위한 +inc\_Student(), +deac\_Student() method를 추가하였다.

+Set~, +Get~ method들은 캡슐화의 결과로 나온 부산물들이다.

5.4 Class StudentManager

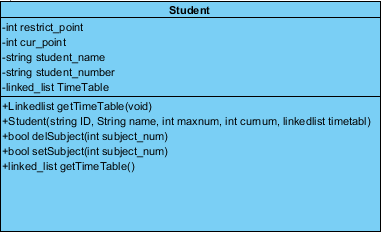


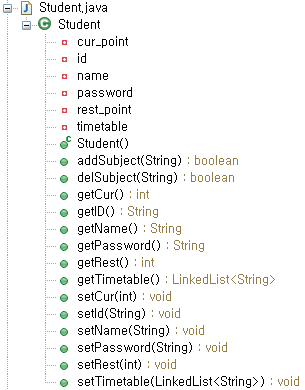


+Compare(string,string)은 단순히 두 string을 비교하는 method이다.

+delSubject(String)는 과목을 삭제하기 위한 method이다.

5.5 Class Student



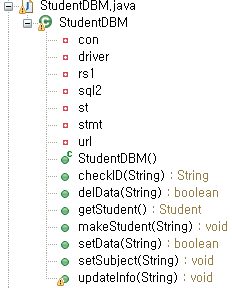


Restrict\_point 변수의 이름을 rest\_point로 변경하였다.

Subject를 추가, 삭제를 위한 +addSubject(String), +delSubject(String) method를 추가하였다.

5.6 ClassStudentDBM





con, driver, rs1, sql1, stmt, url 변수들은 JAVA에서 JDBC를 다루기 위한 변수들이다.

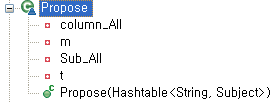
+Set~, +Get~ method들은 캡슐화의 결과로 나온 부산물들이다

5.7 Class private



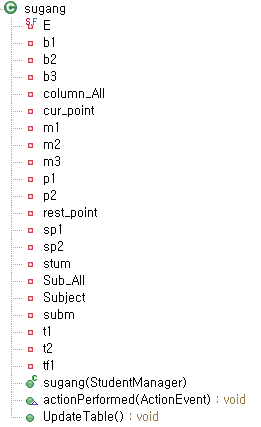
GUI를 위해 추가된 개인 시간표를 출력하는 Class이다.

5.8 Class propose



GUI를 위해 추가된 적절한 과목을 제안해주는 Class이다.

5.9 Class Sugang



GUI를 위해 추가된 로그인 성공시 메인화면을 보여주는 Class이다.