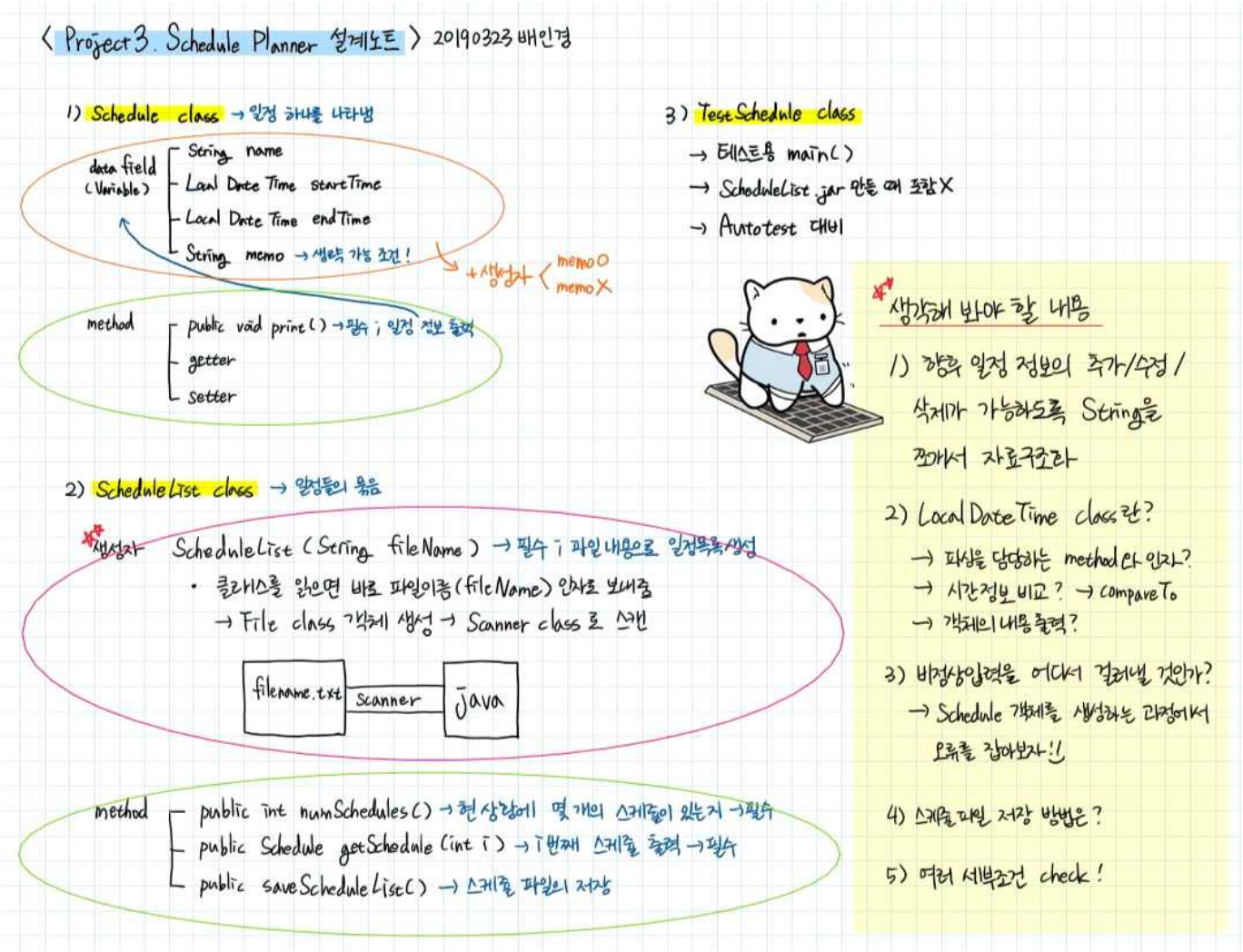
소프트웨어프로젝트
Project3 레포트

Schedule Planner

소프트웨어프로젝트 02분반
월5 수 5,6교시 박창윤 교수님

소프트웨어학부
20190323 배인경



▲ 프로젝트 진행 전 생각할 점을 정리해본 설계 노트

1. Schedule Class (* 로 작성된 문장은 구현 시 고민 했던 사항)

*Schedule.java | *ScheduleList.java | TestSchedule.java

```
1  import java.time.LocalDateTime;
2  import java.time.format.DateTimeFormatter;
3
4  public class Schedule {
5      private String name;
6      private LocalDateTime startTime;
7      private LocalDateTime endTime;
8      private String memo;
9
10     Schedule (String name2, LocalDateTime sTime,
11             LocalDateTime eTime) {
12         name = name2;
13         startTime = sTime;
14         endTime = eTime;
15     } //생성자 memo없을 때
16
17     Schedule (String name2, LocalDateTime sTime,
18             LocalDateTime eTime, String memo2) {
19         this(name2, sTime, eTime);
20         memo = memo2;
21     } //생성자 memo있을 때
22
```

line 1 ~ 2
외부 패키지의 클래스를 불러 사용하고자 import문 작성
* C언어와 다른 java의 장점이라고 생각

line 5 ~ 8
Schedule 클래스에서 사용 될 변수 선언; data field
Schedule App에 저장될 일정(1-1)조건에 맞춤)

line 10 ~ 15
memo가 생략되었을 때의 생성자
(1-1)조건에 따라 memo가 생략될 수 있음)
* 매개값을 다양하게 입력받아 처리할 수 있도록 하기 위해 method overloading 사용

line 17 ~ 21
memo가 생략되지 않았을 때의 생성자
* 매개값을 다양하게 입력받아 처리할 수 있도록 하기 위해 method overloading 사용

```
23
24    public String getName() {
25        return name;
26    }
27    public void setName(String name) {
28        this.name = name;
29    }
30    public LocalDateTime getStartTime() {
31        return startTime;
32    }
33    public void setStartTime(LocalDateTime startTime) {
34        this.startTime = startTime;
35    }
36    public LocalDateTime getEndTime() {
37        return endTime;
38    }
39    public void setEndTime(LocalDateTime endTime) {
40        this.endTime = endTime;
41    }
42    public String getMemo() {
43        return memo;
44    }
45    public void setMemo(String memo) {
46        this.memo = memo;
47    }
48    public void print() {
49        if (memo == null)
50            System.out.println(name + "//"
51            + startTime.format(DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm"))
52            + "// " + endTime.format(DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm")));
53        else
54            System.out.println(name + "//"
55            + startTime.format(DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm"))
56            + "// " + endTime.format(DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm"))
57            + "//" + memo);
58    } //출력
59 }
```

line 24 ~ 47
각 변수에 대한 getter와 setter method들

line 48 ~ 58
일정 정보를 콘솔로 출력하여 주는 print method
if문으로 memo의 유무에 따라 구분
(1-1)조건에 따르면 필수 print method)
* 객체의 내용 출력 : LDT 형태의 객체를 프린트하면 String으로 표현하기 위해 format method 사용
: 변환을 위해 포맷을 직접 설정한 Formatter 생성

2. ScheduleList Class (* 로 작성된 문장은 구현 시 고민 했던 사항)

Schedule.java | ScheduleList.java | *TestSchedule.java

```
1  import java.util.Scanner;
2  import java.util.regex.Pattern;
3  import java.io.File;
4  import java.io.FileOutputStream;
5  import java.time.LocalDateTime;
6  import java.time.format.DateTimeFormatter;
7  import java.io.BufferedReader;
8  import java.io.IOException;
9  import java.io.InputStreamReader;
10 import java.io.PrintStream;
11
12 public class ScheduleList {
13
14     private Schedule[] array = new Schedule[100];
15     private int num = 0;
16     private static String pattern = "(^\\d{4}-\\d{2}-\\d{2} \\d{2}:\\d{2}$)";
17
18
19     ScheduleList(String fileName) {
20         File file = new File(fileName);
21         try {
22
23             Scanner input = new Scanner(file);
24             while(input.hasNext()) {
25                 String line = input.nextLine();
26                 line =  line.trim();
27
28                 if (line.isBlank() || line.charAt(0) == ';')
29                     continue;
30
31                 String[] div = line.split("//");
32
33                 String name, startTime2, endTime2, memo;
34                 LocalDateTime startTime, endTime;
35
```

line 1 ~ 10
외부 패키지의 클래스를 불러 사용하고자 import문 작성
* C언어와 다른 java의 장점이라고 생각

line 14
스케쥴 목록을 작성하기 위한 배열
line 16
LDT형식이 맞는 지 비교하기 위해 선언

line 19 ~ 110
ScheduleList 생성자
* ScheduleList 생성 시 생성자를 실행한 후 ScheduleList에 있는 Schedule 객체를 출력하도록 구상함 + 비정상 입력이 있다면 Schedule 객체가 생성되지 않음
line 26
String method인 trim을 사용해 앞뒤의 공백문자 무시
(조건 1-3)
line 28 ~ 29
빈줄, 또는 처음을 ';'로 시작하면 comment 처리(조건 1-3)
line 31
String으로 주어진 시간 정보를 '//'을 구분자로 시간 정보를 분해해서 LDT형태의 객체로 만들어주기 위한 과정

```
36              name = div[0].trim();
37              startTime2 = div[1].trim();
38              endTime2 = div[2].trim();
39              if(div.length > 3) {
40                  memo = div[3].trim();
41              }
42              else {
43                  memo = null;
44              }
45
46
47              if (name.isBlank()) {
48                  // 오류메시지
49                  System.out.print("No Schedule Title  ; Skip the input line : ");
50                  System.out.println(line);
51                  break;
52              } //이름이 비었을 때
53
54              if (!Pattern.matches(pattern, startTime2)) {
55                  // 오류메시지
56                  System.out.print("Wrong Date Format ; Skip the input line : ");
57                  System.out.println(line);
58                  break;
59              }
60              else {
61                  startTime = LocalDateTime.parse(startTime2, DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm"));
62              }
63
64              if (!Pattern.matches(pattern, endTime2)) {
65                  // 오류메시지
66                  System.out.print("Wrong Date Format ; Skip the input line : ");
67                  System.out.println(line);
68                  break;
69              }
70              else {
71                  endTime = LocalDateTime.parse(endTime2, DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm"));
72              }
73
74                if (startTime.compareTo(endTime) > 0) {
75                    // 오류메시지
76                    System.out.print("Start Time is later than End Time ; Skip the input line : ")
77                    System.out.println(line);
78                    break;
79                }
80
81                if(div.length < 3) {
82                    // 오류메시지
83                    System.out.print("Put too little information ; Skip the input line : ");
84                    System.out.println(line);
85                    break;
86                }
87
88                if(div.length > 4) {
89                    // 오류메시지
90                    System.out.print("Put too much information ; Skip the input line : ");
91                    System.out.println(line);
92                    break;
93                }
94                //입력값이 '이름+시작시간+종료시간+메모 +알파로 더 있으면 오류
95
96                if (memo == null) {
97                    Schedule schedule = new Schedule(name, startTime, endTime);
98                    array[num++] = schedule;
99                } else {
100                   Schedule schedule = new Schedule(name, startTime, endTime, memo);
101                   array[num++] = schedule;
102               }
103           }
104           input.close();
105
106       }
107       catch(Exception e) {
108           System.out.println("Unknown File");
109       }
110   }
111
```

line 36 ~ 44

String method인 trim을 사용해 분해한 시간 정보들의 앞뒤의 공백문자 무시 (조건 1-3)

line 47 ~ 52 비정상 일정 정보 ①

이름이 비었을 경우

콘솔창에 잘못된 이유와 잘못된 문장 보고해 줌

line 54 ~ 72 비정상 일정 정보 ② & ③

(시작, 종료) 시간을 나타내는 스트링이 정상 포맷에 맞는지를 스트링을 체크해서 검사하고, 검사를 통과한 것만 parse를 하고, 통과 못한 것은 비정상으로 처리

* LocalDateTime.parse() method로 파싱
* LDT형태의 객체를 String으로 표현하기 위해 format method 사용
* C와 달리 java특성을 이용하기 위해 line을 읽어와서 파씽할 때 생성자 이용

콘솔창에 잘못된 이유와 잘못된 문장 보고해 줌

line 74 ~ 79 비정상 일정 정보 ④

시작시간이 종료시간보다 느린 경우

* 시간 정보 비교를 위해 compareTo method 사용

콘솔창에 잘못된 이유와 잘못된 문장 보고해 줌

line 81 ~ 86 비정상 일정 정보 ⑤

일정 정보의 개수가 3개 미만일 경우

콘솔창에 잘못된 이유와 잘못된 문장 보고해 줌

line 88 ~ 93 비정상 일정 정보 ⑥

일정 정보의 개수가 4개를 초과할 경우

콘솔창에 잘못된 이유와 잘못된 문장 보고해 줌

line 96 ~ 102

비정상적이지 않은 일정 정보일 경우 Schedule 객체 생성

* method overloading 활용해 메모가 있는 경우와 생략된 경우 나눠 줌

line 107 ~ 109

파일을 읽는 과정에서의 오류

콘솔창에 잘못된 이유 보고해 줌

* 이 과정에서 비정상 입력을 걸러내는 것을 처음에 고려하였으나 Schedule 객체를 생성하는 과정에서 오류를 잡는 것이 맞다는 것을 시행착오를 통해 판단

```
112    public int numSchedules() {
113        return num;
114    }
115
116    public Schedule getSchedule(int i) {
117        return array[i];
118    }
119
120
121    public void saveScheduleList(ScheduleList s_list, String fileName) {
122       PrintWriter newFile = null;
123       try {
124           newFile = new PrintWriter(fileName);
125       }
126       catch(Exception e) {
127           System.out.println("error");
128       }
129       for(int i=0; i<s_list.numSchedules();i++)
130           newFile.println(s_list.getSchedule(i));
131       newFile.close();
132    }
133
134 }
135
```

line 112 ~ 114

현 상황에 몇 개의 스케쥴이 있는지 나타내는 method

line 116 ~ 118

i번째 스케쥴 출력 method

line 121 ~ 142

향후 프로젝트를 위한 saveScheduleList method

main에서 fileName을 기존 파일과 똑같은 위치와 똑같은 이름으로 인자로 넣어주면 기존 파일이 결과에 맞춰 변경됨

* 처음엔 Buffer를 생성하는 거에 대해 인터넷 서칭을 해보다가 교수님께서 제안해 주신 PrintWriter 기능을 이용하여 다시 도전하였더니 성공함.

3. TestSchedule Class

Schedule.java | ScheduleList.java | *TestSchedule.java

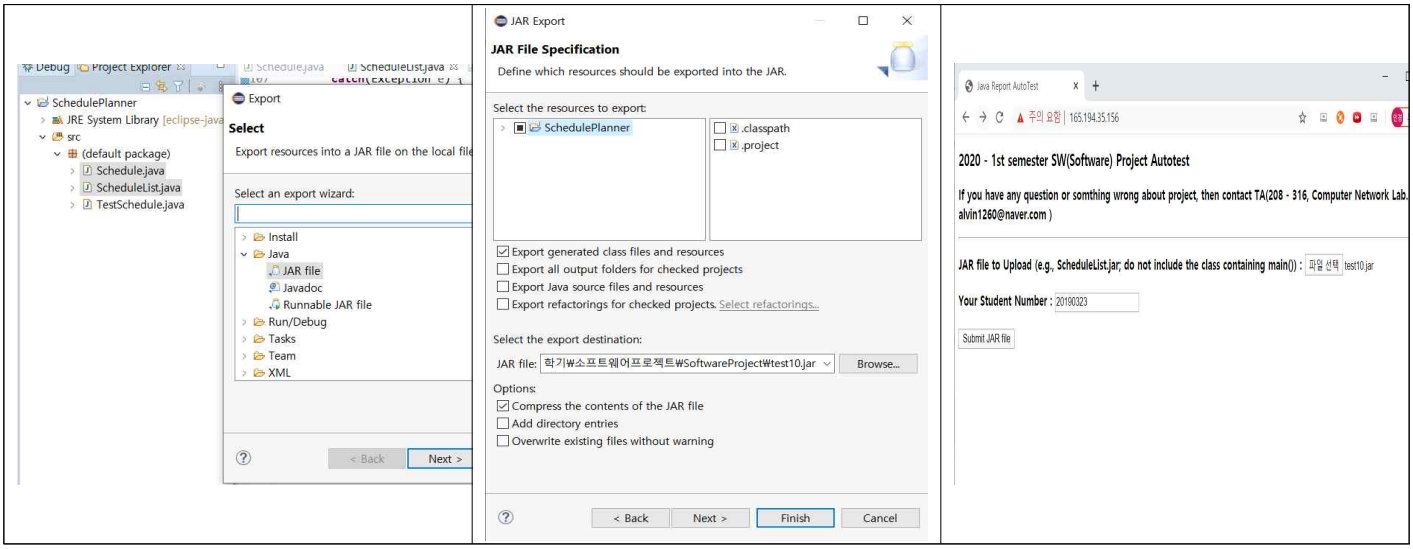
```
1  public class TestSchedule {
2
3      public static void main(String[] args) {
4
5          ScheduleList list = new ScheduleList("C:\\" + "schedule-file.data");
6          for (int i = 0; i < list.numSchedules(); i++)
7              list.getSchedule(i).print();
8
9      }
10
11 }
12
```

line 1 ~ 10

프로젝트 3 관련 유인물에 따라 자신의 프로그램 정상 동작 확인 방법을 참고하여 작성한 main()

(조건 1-2)에 따라 main을 통한 테스트는 TestSchedule Class를 따로 작성하여 수행함)

(조건에 따라 'jar'을 이용하여 ScheduleList.jar을 만들 때는 TestSchedule class는 제외 시킴)



▲ jar 파일 제작 후 autotest 사용하는 과정

```
Sun May 10 05:39:42 KST 2020 by 20190323
Case1: Normal Input
Java Meeting//2020-03-25 13:00// 2020-03-25 13:50//Test
Java Meeting 2//2020-03-25 15:01// 2020-03-25 15:50//Test - 2
New Year//2020-01-01 00:01// 2020-01-01 01:00//Happy
EIRIC//2020-01-02 00:00// 2020-01-02 00:01
End of Case1

Case2: Check String Trimming
The result should be the same as the normal case above.
Java Meeting//2020-03-25 13:00// 2020-03-25 13:50//Test
Java Meeting 2//2020-03-25 15:01// 2020-03-25 15:50//Test - 2
New Year//2020-01-01 00:01// 2020-01-01 01:00//Happy
EIRIC//2020-01-02 00:00// 2020-01-02 00:01

End of Case2


*** From now on, Test Reactions on Abnormal Inputs ***
Your Answer (your program's reaction) should be similar to the Correct Answer

Case3-1: No Schedule Title
  Correct Answer: No Schedule Title  ; Skip the input line : //2019-12-25 00:01//2019-12-25 01:00//Test
    Your  Answer : No Schedule Title  ; Skip the input line : //2019-12-25 00:01//2019-12-25 01:00//Test
End of Case3-1

Case3-2: Check Date Format Error  ->
  Correct Answer: Wrong Date Format ; Skip the input line : Java Meeting//02-01-2020-12-25 00:01//2019-12-25 01:00//Test
    Your  Answer : Wrong Date Format ; Skip the input line : Java Meeting//02-01-2020-12-25 00:01//2019-12-25 01:00//Test
End of Case3-2

Case3-3: Time Conflict Error  ->
  Correct Answer: Start Time is later than End Time ; Skip the input line : Wrong Timing //2020-01-25 00:01//2019-01-25 01:00//Time Conflict
    Your  Answer : Start Time is later than End Time ; Skip the input line : Wrong Timing //2020-01-25 00:01//2019-01-25 01:00//Time Conflict
End of Case3-3

Case4: Unknown Schedule File
  Correct Answer: Unknown File
    Your  Answer : Unknown File
End of Case4
```

▲ AutoTest 결과 화면 (20190323 배인경)

평 가 표

평가 항목	학생 자체 평가 (리포트 해당 부분 표시 및 간단한 의견)	평가 (빈칸)	점수 (빈칸)
기본 동작 - autotest 결과화면	1) 인터페이스 요구사항을 준수하였는가? 1-1) print 메소드 포함, 정보 변수들 + 생성자 1-2) num · get schedule 메소드 포함 + 생성자 + main 사용X 1-3) '//' 주석자로 사용, 메모생략 고려 (by_method overloading) LDT class 사용, trim 공백처리, 빈줄·'i' comment처리 2) 기능적 요구사항을 준수하였는가? 1-4) 오류에 대해 콘솔 보고		
	1-5) main을 포함하는 별도 testSchedule 클래스 생성 3) autotest 결과화면 보고서에 포함시킴		
설계 사항 - 설계 착안점 - 사용한 클래스 - 시행 착오 및 해결책	· 향후 일정 정보의 추가/수정/삭제가 가능하도록 String을 쪼개서 자료구조화함 · 일정 관리 프로그램의 각 class가 있어야 할 공간/기능정의 · 사용한 클래스 총 3개 (test 포함 → jar 포함X) · 시행착오 → 레포트에 기술함 해결책 및 해결과정에서 사용하게 된 함수나 방법 → 레포트에 기술함 (* … 파란글씨로 기술)		
본인 인증	· 리포트에 최종 autotest 결과화면 (학번기재), 작업 시 설계노트, 소스프로그램+설명, 과제평가표 첨부 · 캡처화면에 본인이 지정한 파일경로 or 이름 노출		
기타	향후 프로젝트를 위해 스케줄 파일을 저장하는 public saveScheduleList() 메소드를 제작		
총평/계	이번 프로젝트3을 수행하며 JAVA에 대한 전반적 이해도가 올라가고 실력을 향상시킴을 스스로 느낌.		

* 학생 자체 평가는 점수에 반영되지 않음.
* 학생 스스로 자신의 보고서를 평가하면서, 체계적으로 프로젝트를 마무리하도록 유도하는 것이 목적임.