Traveler Digital Watch
OOPT 3rd Cycle

소프트웨어 모델링 및 실습
[19. 06. 10]

201411295 이상훈
201711394 민하은
201711395 박성준
201711423 정종화

Index

1. Requirement Revision
2. System Test Response
3. Static Analysis Response
4. OOPT Review

1. Requirement Revision

Stage 2040

R.2.1 Start Timer
- Problem
1. (A) 타이머 시작버튼을 누른다. 2. (S) 타이머에 저장된 시간을 1초씩 줄이며 흐르게 한다. 3. (S) 타이머 시간이 0이 되면 알람을 10초 울린다. 4. (S) 타이머 시간 화면에 초과된 시간을 1초씩 흐르게 한다. - Timer가 종료된 시점에서도 알람이 울리지 않는다. 소리가 나지 않을 뿐만 아니라 알람 표시조차 없다.
- Fixed
- Timer가 종료되면 알람이 울리도록 변경하고, 알람 표시가 나도록 수정.

1. Requirement Revision

Stage 2040

R.2.1 Start Timer
- Problem
- 시계의 초에 해당하는 부분이 invalid 한 값이 등장한다. (00 ~ 59가 아닌 74가 나오는 문제 발생.
- Fixed
- 시계 범위에 맞는 값이 나오도록 조정 : 00 ~ 59

2. System Test Response

1. Set time 모드에서 유효한 시간으로 변경 후 오른쪽 위 버튼을 길게 눌러 바꾼 시간을 적용한다.
- Problem
- 시간을 바꿀 수는 있으나 오른쪽 위 버튼을 길게 눌러도 바꾼 시간이 정상적으로 적용되지 않음.
- Fixed(x) -> 문제없음
Set time에서 조정하고 싶은 시간으로 맞춘 뒤, 저장 버튼을 누른 후 set Time 모드를 종료시켜야 결과가 저장됨. 저장되지 않은 경우, 조정된 시간을 무시하고 처음에 설정 되어있던 time을 보여주므로 저장 후 mode를 종료시켜야 한다.

2. System Test Response

2. Global Time모드에서 왼쪽 위 버튼을 짧게 눌러 Global Time을 15분 씩 증가시킨다.

- Problem

- 시간을 증가시킬 수는 있으나 '15분 -> 30분 -> 45분'이 아닌 '15분 -> 45분 -> 1시간 30분'으로 증가함. 즉 더해지는 폭이 15분씩 증가함

- Fixed

```
public GlobalTime() {
    this.gTime = super.time;
    this.modifiedMinute = 15;
    this.modifiedHour = 1;
    this.simpleDateFormat = new SimpleDateFormat( pattern: "yy-MM-dd HH:mm:ss");
    this.stringGTime = this.simpleDateFormat.format(this.gTime.getTime());
    this.timer.scheduleAtFixedRate(this.task,  delay: 0L,  period: 1000L);
}
```

```
public void saveMinutesAdd() { this.gTime.add( field: 12, this.modifiedMinute); }
```

- modifiedMinute 변수를 통해 분이 단위에 맞게 증가할 수 있도록 수정하였다.

2. System Test Response

7. timer 모드에서 시간을 00:00:00으로 세팅하고 시작 버튼을 눌러 timer를 실행시킨다.

- Problem

- 프로그램이 동작하지 않음

- Fixed

```
private float time = 0.0F;
private boolean isTimerStart = false;
private boolean isCountDownZero = false;
private boolean setBuzzerRing = false;
```

- 변수 초기값을 제대로 전달하여 프로그램이 동작하도록 수정하였다.

8. stopwatch 모드에서 시간이 00:00:00일 때 리셋 버튼을 눌러 리셋 후, 시작 버튼을 눌러 stopwatch를 실행한다.

- Problem

☝ 프로그램이 동작하지 않음.

- Fixed

```
public void resetStopwatch() {
    this.startStopwatch = false;
    this.stopwatchTime = 0L;
}
```

- resetStopwatch()로 분리하여 리셋이 올바르게 적용되고, 프로그램이 동작하도록 수정하였다.

2. System Test Response

10. 동일한 alarm 2개 이상 중복하여 설정 한 후, 설정한 시간이 되었을 때 alarm이 제대로 울리는지 확인한다.

- Problem

- 리셋이 되긴 하나 가끔 씩 프로그램이 멈춰버린다. 안정성 문제 발생

- Fixed

```
public boolean isAlarmActivated() {
    return this.alarm0.getIsAlarmActivated() || this.alarm1.getIsAlarmActivated() || this.alarm2.getIsAlarmActivated() || this.alarm3.getIsAlarmActivated();
}

public boolean isThisAlarmActivated() {
    switch(this.alarmIndex) {
    case 0:
        return this.alarm0.getIsAlarmActivated();
    case 1:
        return this.alarm1.getIsAlarmActivated();
    case 2:
        return this.alarm2.getIsAlarmActivated();
    case 3:
        return this.alarm3.getIsAlarmActivated();
    default:
        return false;
    }
}
```

- switch문이 중첩되지 않도록 하여 안정성을 향상시켰다.

2. System Test Response

12. alarm을 1개 설정한 후에, timer 모드 화면으로 전환되고 기다렸다가 설정한 시간에 alarm이 제대로 울리는지 확인한다.
- Problem
- alarm이 제 시간에 제대로 울리지 않고, 설정한 시간에서 6초 정도 시간이 더 흐른 후에 울림.
- Fixed
- exception in thread "AWT-EventQueue-0" java.lang.NullPointerException 에러 수정 완료 -> 알람 제시간에 울린다.

15. timer를 1개 설정한 후에, time 모드 화면으로 전환되고 기다렸다가 설정한 시간에 timer alarm이 제대로 울리는지 확인한다.
- Problem
- timer alarm이 설정한 시간이 만료되어도 울리지 않는다.
- Fixed
- 알람 울림 기능 추가.

3. Static Analysis Response

1. bug

- Problem

```
[TravelerDigitalWatchJDKError/src/Alarm.java]
    1) Either re-interrupt this method or rethrow the "InterruptedException". Bug 해결 요망.
        → thread에게 interrupt를 발생시켜 주어야 함.
```

- Fixed

```
try {
    Thread.sleep( millis: 9);
} catch (InterruptedException e) {
    String exception= e.getMessage();
    System.out.println("UI Interrupted Exception occured : "+exception);
    Thread.currentThread().interrupt();
}
```

- thread에게 interrupt를 발생시켜주었다.

3. Static Analysis Response

1. Vulnerability

- Problem

```
[TravelerDigitalWatchJDKError/src/NumberGenerator.java]
    1) The use of java.lang.Math.random() is predictable. 해결 요망.
      → 난수 생성 시 그냥 random() method 사용 시 예측이 가능할 수 있으므로 secureRandom 사용을 권
```

- Fixed

```
SecureRandom random;
```

```
try{
    random = SecureRandom.getInstance("SHA1PRNG");
}catch(NoSuchAlgorithmException e){
    String exception = e.getMessage();
    System.out.println("Random Number Generator Exception occurred : "+exception);
}
```

- 수정 완료

4. OOPT Review

- 소프트웨어의 단계적 개발의 중요성
- 반복적인 작업을 통한 객체 지향 의 중요성
- OOAD 방법론을 통한 Stage를 진행하다 실제 코드 구현을 시작할 때 클래스 간의 관계를 한 눈에 파악할 수 있어 쉽게 진행할 수 있었음
- 소프트웨어 검증을 통해 프로젝트의 결함을 파악하고 수정하는 단계를 거치면서 완벽한 소프트웨어를 만드는 것이 얼마나 많은 인력과 시간을 필요로 하는 일인지 실감할 수 있었음

- 소모팀에서 찾아내지 못하는 오류 발견에 큰 도움이 됨
- CTIP 환경 구축단계 어려움 -> 소검팀의 빠른 피드백이 있었음

감사합니다

Q & A