객체지향개발방법론

# OOAD 2050 (Implementation)

201212519 컴퓨터공학과 김선우
201510624 컴퓨터공학과 김용현
201614157 컴퓨터공학과 김도연
201614158 컴퓨터공학과 장다혜

# Refine Class Diagram

**Use case 변화**

2. show current Time 제거

3. show current Date 제거

22,23 Select SW와 Change SW를 Change SW로 병합

## DWS controller

Getter, Setter, Unit test를 위한 함수는 클래스다이어그램에 표현하지 않았습니다.

StopwatchTime 클래스 제거

1. 메소드 변경
   A. checkState(), checkCurrentTime() 제거
   B. reqSelectAlarmNum() 제거
   C. turnOnAlarmIndicator,turnOffAlarmIndicator 제거

2. 변수 변경
   A. button 제거
   B. defaultTime 제거
   C. currentTime, Date 제거

## Timekeeping Mode

1. 메소드 변경
   A. changeMode(currentState: int) 추가
   B. changeValue의 파라미터로 button 추가
   C. previousWorld, nextWorld의 파라미터와 리턴값 제거
   D. Unit test를 위한 setValue() 함수 추가

## Stopwatch Mode

1. 메소드 변경
   A. calStopwatchTime() 제거
   B. start,stop,reset,lapStopwatch()
   리턴 Time 추가

## Timer Mode

1. 메소드 변경
    A. changeMode(currentState: int) 추가
    B. startTimer, stopTimer 리턴 Time 추가
    C. saveTimerTime(int) 제가
2. 변수 변경
    A. currentTime 변수 제거

## Worldtime Mode

1. 메소드 변경
    A. ChangeWorldtime() 추가
2. 변수 변경
    A. String[] worldTimezone 추가
    B. Calendar worldTime 추가

## BrightcontrolMode

1. 메소드 변경
    A. Daylong에 리턴값과 인풋 파라미터 추가

## Ring(미구현)

## SWMode

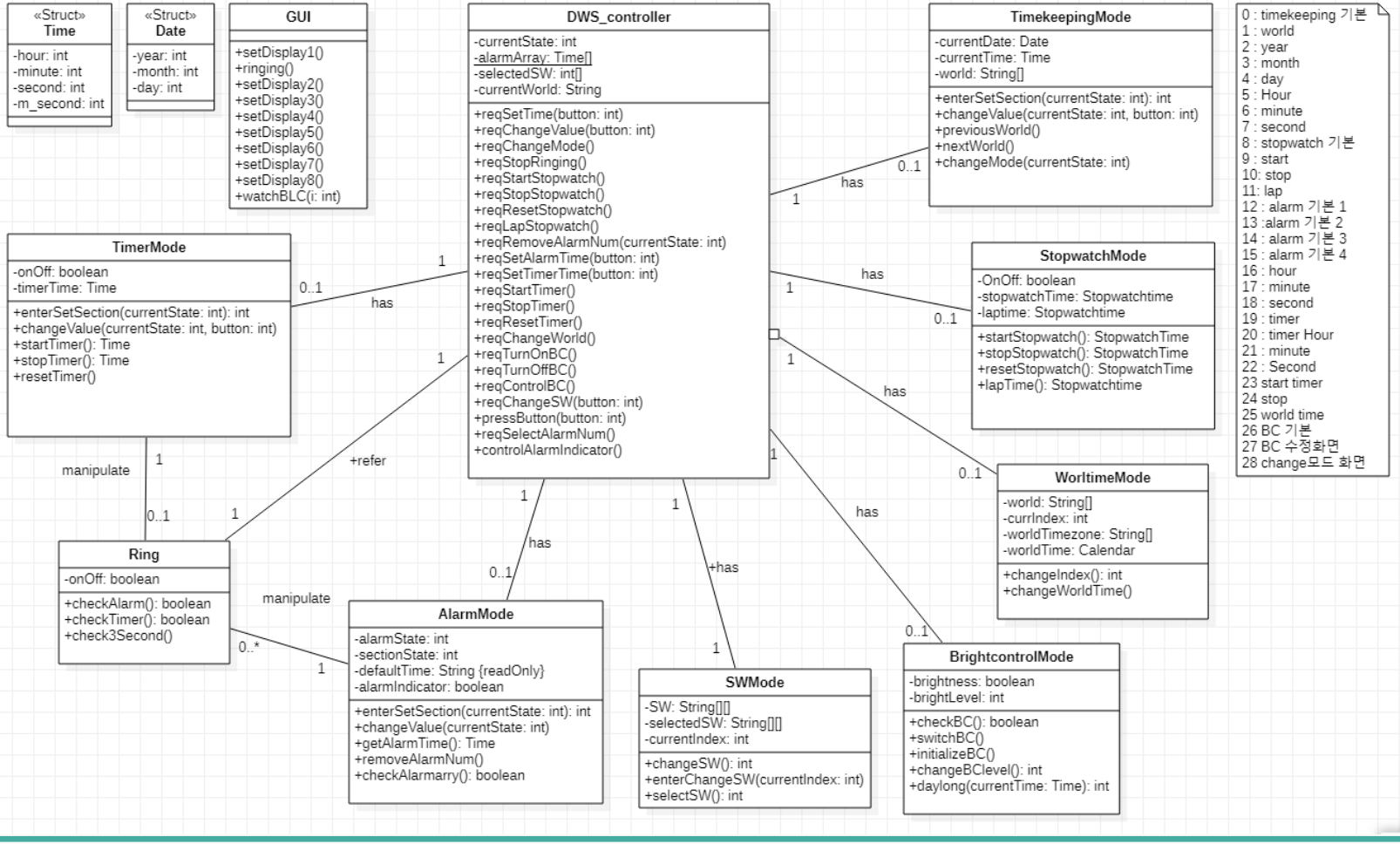
1. 메소드 변경
    A. enterChangeSW()에 인풋 파라미터 추가
2. 변수 변경
    A. Int currentIndex 추가
    B. swMode String[] -> String[][] 2차원배열로 변경
    C. String[][] selectedSW 추가

# UI → GUI

1. 메소드 변경
   A. display1(), display2(), displayMain()

   → setDisplay1~8()로 변경 및 추가

   B. watchBLC() 추가


«Struct»
Time
-hour: int
-minute: int
-second: int
-m_second: int
«Struct»
Date
-year: int
-month: int
-day: int
GUI
+setDisplay1()
+ringing()
+setDisplay2()
+setDisplay3()
+setDisplay4()
+setDisplay5()
+setDisplay6()
+setDisplay7()
+setDisplay8()
+watchBLC(i: int)
DWS_controller
-currentState: int
-alarmArray: Time[]
-selectedSW: int[]
-currentWorld: String
+reqSetTime(button: int)
+reqChangeValue(button: int)
+reqChangeMode()
+reqStopRinging()
+reqStartStopwatch()
+reqStopStopwatch()
+reqResetStopwatch()
+reqLapStopwatch()
+reqRemoveAlarmNum(currentState: int)
+reqSetAlarmTime(button: int)
+reqSetTimerTime(button: int)
+reqStartTimer()
+reqStopTimer()
+reqResetTimer()
+reqChangeWorld()
+reqTurnOnBC()
+reqTurnOffBC()
+reqControlBC()
+reqChangeSW(button: int)
+pressButton(button: int)
+reqSelectAlarmNum()
+controlAlarmIndicator()
TimekeepingMode
-currentDate: Date
-currentTime: Time
-world: String[]
+enterSetSection(currentState: int): int
+changeValue(currentState: int, button: int)
+previousWorld()
+nextWorld()
+changeMode(currentState: int)
0 : timekeeping 기본
1 : world
2 : year
3 : month
4 : day
5 : Hour
6 : minute
7 : second
8 : stopwatch 기본
9 : start
10: stop
11: lap
12 : alarm 기본 1
13 :alarm 기본 2
14 : alarm 기본 3
15 : alarm 기본 4
16 : hour
17 : minute
18 : second
19 : timer
20 : timer Hour
21 : minute
22 : Second
23 start timer
24 stop
25 world time
26 BC 기본
27 BC 수정화면
28 change모드 화면
TimerMode
-onOff: boolean
-timerTime: Time
+enterSetSection(currentState: int): int
+changeValue(currentState: int, button: int)
+startTimer(): Time
+stopTimer(): Time
+resetTimer()
StopwatchMode
-OnOff: boolean
-stopwatchTime: Stopwatchtime
-laptime: Stopwatchtime
+startStopwatch(): StopwatchTime
+stopStopwatch(): StopwatchTime
+resetStopwatch(): StopwatchTime
+lapTime(): Stopwatchtime
WorltimeMode
-world: String[]
-currIndex: int
-worldTimezone: String[]
-worldTime: Calendar
+changeIndex(): int
+changeWorldTime()
Ring
-onOff: boolean
+checkAlarm(): boolean
+checkTimer(): boolean
+check3Second()
AlarmMode
-alarmState: int
-sectionState: int
-defaultTime: String {readOnly}
-alarmIndicator: boolean
+enterSetSection(currentState: int): int
+changeValue(currentState: int)
+getAlarmTime(): Time
+removeAlarmNum()
+checkAlarmarry(): boolean
SWMode
-SW: String[][]
-selectedSW: String[][]
-currentIndex: int
+changeSW(): int
+enterChangeSW(currentIndex: int)
+selectSW(): int
BrightcontrolMode
-brightness: boolean
-brightLevel: int
+checkBC(): boolean
+switchBC()
+initializeBC()
+changeBClevel(): int
+daylong(currentTime: Time): int
has
manipulate
+refer
+has

# Activity 2050 : OOI
# (Object Oriented Implementation)

| **Type** | **Class** |
| --- | --- |
| Name | DWS_Controller |
| Purpose | 시계의 메인 시스템으로 사용자의 반응에 따라 기능을 조절하는 시스템 |
| Overview | • UI에서 받은 버튼과 currentState을 가지고 DWS를 control한다.<br>• 각 Mode에서 받은 리턴값이나 getter를 이용하여 얻은 정보를 DWS화면에 display되도록 UI에게 전달한다. |
| Exceptional Courses of Events | N/A |

| **Type** | **Class** |
| --- | --- |
| Name | AlarmMode |
| Purpose | 시계의 알람을 관리하는 클래스 |
| Overview | • 알람은 총 4개를 저장할 수 있으며 4개 중 1개의 알람이 저장되어 있으면 alarmIndicator가 켜진다. 만약 알람 4개가 비어 있다면 alarmIndicator는 자동적으로 꺼지게 된다.<br>• 사용자가 알람을 설정하거나 리셋할 수 있으며 알람이 정해진 시간이 되었을 때 울리게 해준다. |
| Exceptional Courses of Events | N/A |

| **Type** | **Class** |
| --- | --- |
| Name | BrightcontrolMode |
| Purpose | 시계의 Bright를 관리하는 클래스 |
| Overview | • Bright를 turn on / turn off 할 수 있다.<br>• Bright의 level을 관리하여 사용자가 원하는 밝기를 보여줄 수 있다.<br>• 시간에 따라 자동적으로 밝기에 변화를 준다. |
| Exceptional Courses of Events | N/A |

| **Type** | **Class** |
| --- | --- |
| Name | StopWatchMode |
| Purpose | 시계의 StopWatch를 관리하는 클래스 |
| Overview | • Stopwatch를 start 하면 1/100초로 타임이 올라간다.<br>• Stopwatch를 stop 하면 stopwatchTime은 멈춘다.<br>• Stopwatch를 reset 하면 stopwatchTime을 초기화 시켜준다.<br>• Stopwatch를 lap 하면 lapTime을 보여주고 stopwatchTime은 계속해서 돌아간다. |
| Exceptional Courses of Events | N/A |

| **Type** | **Class** |
| --- | --- |
| Name | SWMode |
| Purpose | 시계의 SWMode를 변경하는 클래스 |
| Overview | • 사용자는 총 5개의 SW MODE 를 선택해 변경할 수 있다.<br>• Default 모드는 (기본)TimeKeeping, Timer, Stopwatch, Worldtime으로 정해져 있다.<br>• SW MODE 선택 시, 선입선출을 통해 모드가 저장된다. |
| Exceptional Courses of Events | N/A |

| **Type** | **Class** |
| --- | --- |
| Name | TimeKeepingMode |
| Purpose | 시계의 Timekeeping을 관리하는 클래스 |
| Overview | • 사용자가 시간을 설정하여 TimeKeeping을 자동으로 흐를 수 있게 한다.<br>• world, year, month, day, hour, minute, second 순으로 시간을 설정할 수 있다. |
| Exceptional Courses of Events | N/A |

| **Type** | **Class** |
| --- | --- |
| Name | TimerMode |
| Purpose | 시계의 Timer를 관리하는 클래스 |
| Overview | • Timer 시간을 시, 분, 초로 설정할 수 있다.<br>• Timer를 start 하면 사용자가 지정한 시간이 1초씩 감소한다. 만약 00:00:00이 된다면 감소되는 것을 멈추고 ring을 울려준다.<br>• Timer를 stop 하면 Timer의 시간은 멈춘다.<br>• Timer를 reset 하면 Timer의 시간은 초기화된다. |
| Exceptional Courses of Events | N/A |

| **Type** | **Class** |
| --- | --- |
| Name | WorldTimeMode |
| Purpose | WorldTime을 관리하는 클래스 |
| Overview | • 설정한 나라의 시간들을 가져와서 보여주는 클래스 이다.<br>(한국->미국->영국->중국->러시아->싱가포르 순) |
| Exceptional Courses of Events | N/A |

| **Type** | **Method** |
| --- | --- |
| Name | displayTask |
| Purpose | 1초마다 시계의 currentState의 상태에 따라 맞는 화면을 띄워주기 위한 메소드 |
| Overview | • currentTime의 값을 리턴받는 일종의 getter 함수 |
| Exceptional Courses of Events | N/A |

| **Type** | **Method** |
| --- | --- |
| Name | calculateTask |
| Purpose | 사용자가 시계를 시작한 다음부터 Timekeeping의 시간을 활성화하기 위한 메소드로 1초마다 Timekeeping의 시간을 1초씩 올려준다. |
| Overview | • 시계를 시작하면 1970년 1월1일로 시작하며 시간은 00시 00분 00초로 시작한다. 1초마다 1초씩 calculateTask를 통해 Timekeeping의 시간을 증가시킨다. |
| Exceptional Courses of Events | N/A |

| **Type** | **Method** |
| --- | --- |
| Name | selectSW |
| Purpose | 사용자가 선택한 SW를 바꿔주기 위한 메소드 |
| Overview | • 사용자가 선택한 SW를 시스템에 저장된 배열에 저장해주면 선입선출을 통해 배열에 맨 앞에 저장된 Mode를 제거하고 선택된 Mode를 배열에 저장한다. |
| Exceptional Courses of Events | N/A |

| **Type** | **Method** |
| --- | --- |
| Name | changeWorldTime |
| Purpose | 사용자가 선택한 World에 따라 시간을 바꿔주기 위한 메소드 |
| Overview | • 사용자가 선택한 World에 따라 그 나라에 맞는 timezone을 선택한 이후 timezone에 맞는 시간을 설정한다. |
| Exceptional Courses of Events | N/A |

| **Type** | **Method** |
| --- | --- |
| Name | tempTask // timerTask |
| Purpose | stopwatch의 시간을 계산하기 위한 메소드<br>timer의 시간을 계산하기 위한 메소드 |
| Overview | • 사용자가 stopwatch에서 B버튼을 누르면 stopwatch의 시간을 계속 증가시켜주며 laptime시에는 정지된 시간을 보여주지만 뒤에서는 계속 stopwatch의 시간이 흐르고 있기 때문에 계속 증가시켜 주며 resetStopwatch시에 stopwatch의 시간을 초기화 한다.<br>• 사용자가 timer에서 B버튼을 누르면 timer의 시간을 계속 감소시켜주며 resetTimer시에 timer의 시간을 초기화 한다. |
| Exceptional Courses of Events | N/A |

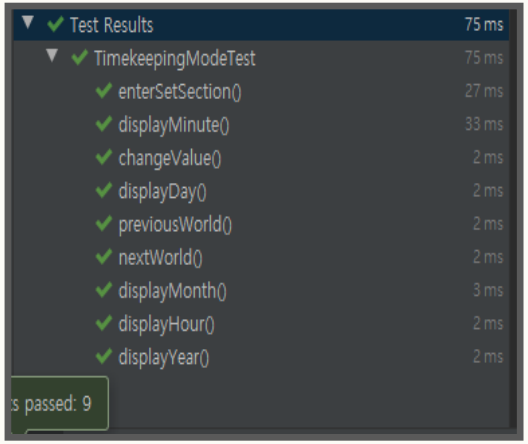
| **Type** | **Method** |
| --- | --- |
| Name | daylong |
| Purpose | 사용자가 BrightControl을 켯을 때 현재시간에 맞춰 밝기를 자동 조절하기 위한 메소드 |
| Overview | • 사용자가 BrightControl을 켜게 되면 매시 정각 현재시간에 맞춰 화면의 밝기(색)을 조절한다.<br>• 하루를 3분할로 나누어 각 시간에 맞는 밝기를 보여준다. |
| Exceptional Courses of Events | N/A |

# Activity 2060 :
# Unit Test

# 미구현 사항 / Unit test를 진행하지 못한 부분

| | |
|---|---|
| Timekeeping mode | 사용자의 incorrect한 입력값에 반응하는지 test |
| Stopwatch mode | 1/100초 단위로 측정되는지 test<br>내부적으로 onOff 변수 값의 변화 test |
| Alarm / Timer Mode | Ringing과 관련된 test(Ring클래스 미구현) |
| Worldtime mode | 세계시간의 완전한 비교 test |
| Brightcontrol mode | BRC에 의한 UI 변화 test(내부 변수 test 완료) |
| SW select mode | UI 변화 test(내부 변수 test 완료) |

# TimeKeeping Mode

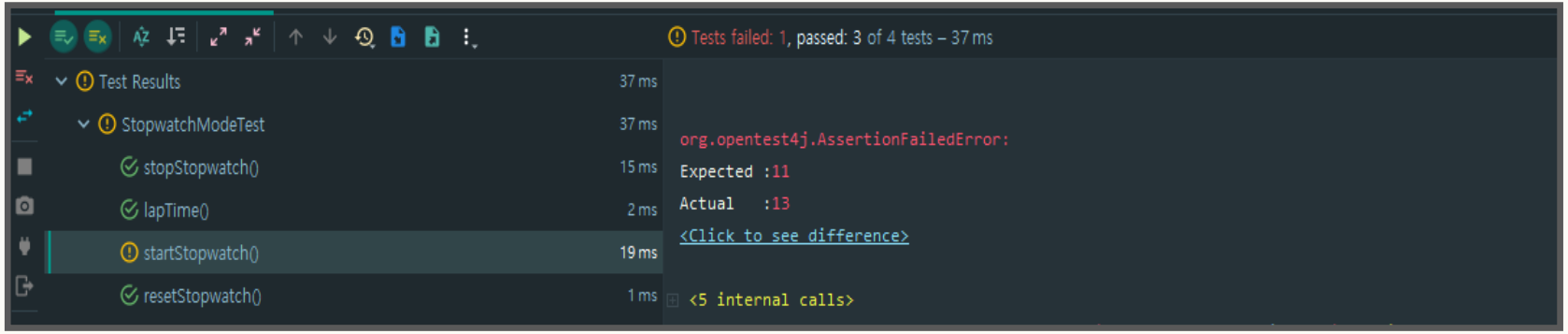



```java
@Test
void changeValue() {
    TimekeepingMode tkm = new TimekeepingMode();

    tkm.setValue( currentState: 2, value: 1970);
    tkm.changeValue( currentState: 2, button: 3);
    //Year의 값이 1970아래로 내려가는지
    assertEquals( expected: "1970 ", tkm.displayYear());

    tkm.setValue( currentState: 3, value: 11);
    tkm.changeValue( currentState: 3, button: 1);
    //Month의 값이 1로 되돌아 가는지
    assertEquals( expected: "01 ", tkm.displayMonth());

    tkm.setValue( currentState: 4, value: 31);
    tkm.changeValue( currentState: 4, button: 1);
    //Day의 값이 1로 되돌아 가는지
    assertEquals( expected: "01", tkm.displayDay());

    tkm.setValue( currentState: 5, value: 23);
    tkm.changeValue( currentState: 5, button: 1);
    //Hour의 값이 0로 되돌아가는지
    assertEquals( expected: "00 ", tkm.displayHour());

    tkm.setValue( currentState: 6, value: 59);
    tkm.changeValue( currentState: 6, button: 1);
    //Minute의 값이 0으로 되돌아가는지
    assertEquals( expected: "00 ", tkm.displayMinute());

    tkm.setValue( currentState: 7, value: 59);
    tkm.changeValue( currentState: 7, button: 1);
    //Second의 값이 0으로 되돌아가는지
    assertEquals( expected: "00 ", tkm.displaySecond());

}
```

```java
@Test
void previousWorld() {
    TimekeepingMode tkm = new TimekeepingMode();
    tkm.previousWorld();
    assertEquals( expected: "SINGAPORE", tkm.displayWorld());
}
```

```java
@Test
void nextWorld() {
    TimekeepingMode tkm = new TimekeepingMode();
    tkm.nextWorld();
    assertEquals( expected: "AMERICA", tkm.displayWorld());
}

@Test
void displayYear() {
    TimekeepingMode tkm = new TimekeepingMode();
    assertEquals( expected: "1970 ", tkm.displayYear());
}

@Test
void displayMonth() {
    TimekeepingMode tkm = new TimekeepingMode();
    assertEquals( expected: "01 ", tkm.displayMonth());
}

@Test
void displayDay() {
    TimekeepingMode tkm = new TimekeepingMode();
    assertEquals( expected: "01", tkm.displayDay());
}

@Test
void displayHour() {
    TimekeepingMode tkm = new TimekeepingMode();
    assertEquals( expected: "00 ", tkm.displayHour());
}

@Test
void displayMinute() {
    TimekeepingMode tkm = new TimekeepingMode();
    assertEquals( expected: "00 ", tkm.displayMinute());
}
```

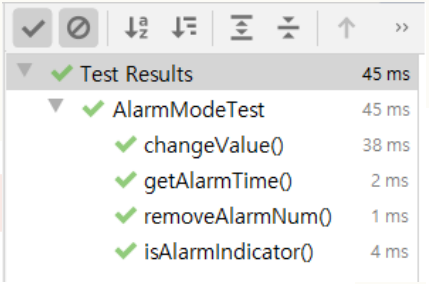
## Stopwatch mode




```java
@Test
void stopStopwatch() {
    // 시작 상태에서 B 버튼에 의해서 정지되는지 test
    dws.setCurrentState(9);
    dws.pressButton(1);
    assertFalse(stw.isOnOff());

}
```

```java
@Test
void resetStopwatch() {
    // reset 버튼으로 stopwatch timeOI 0:0:0으로 초기화되는지
    dws.setCurrentState(10);
    dws.pressButton(3);
    assertEquals( expected: 0,stw.resetStopwatch().getMinute());
    assertEquals( expected: 0,stw.resetStopwatch().getSecond());
    assertEquals( expected: 0,stw.resetStopwatch().getM_second());
}
```

# Alarm Mode




```java
@Test
void changeValue() {
    AlarmMode alm = new AlarmMode();

    alm.getAlarmTime( state: 12).setHour(24);
    alm.changeValue( currentstate: 16,  button: 1 );
    assertEquals( expected: 1, alm.getAlarmTime( state: 12).getHour());
    // hour의 값이 24를 초과하는지

    alm.getAlarmTime( state: 12).setMinute(59);
    alm.changeValue( currentstate: 17,  button: 1 );
    assertEquals( expected: 0, alm.getAlarmTime( state: 12).getMinute());
    // minute의 값이 59를 초과하는지
}
```

```java
@Test
void isAlarmIndicator() {
    AlarmMode alm = new AlarmMode();
    assertFalse(alm.isAlarmIndicator());
    // 알람 설정 안된 상태에서 Indicator가 off(false) 상태인지

    alm.getAlarmTime( state: 12).setHour(24);
    alm.getAlarmTime( state: 12).setMinute(59);
    alm.getAlarmTime( state: 13).setHour(24);
    alm.getAlarmTime( state: 13).setMinute(59);
    alm.checkAlarmArray();
    assertTrue(alm.isAlarmIndicator());
    // 알람이 하나라도 켜지면 Indicator가 on(true) 상태인지

    alm.removeAlarmNum( currentstate: 12);
    alm.checkAlarmArray();
    assertTrue(alm.isAlarmIndicator());
    // 설정된 알람 2개 중 1개가 삭제되었을때 Indicator가 on(true) 상태인지
}
```

```java
@Test
void removeAlarmNum() {
    AlarmMode alm = new AlarmMode();
    alm.getAlarmTime( state: 12).setHour(24);
    alm.getAlarmTime( state: 12).setMinute(59);
    alm.removeAlarmNum( currentstate: 12);

    assertEquals( expected: 0, alm.getAlarmTime( state: 12).getHour());
    assertEquals( expected: 0, alm.getAlarmTime( state: 12).getMinute());
} // 현재 currentState에서 선택된 알람이 삭제되는지
```

```java
@Test
void getAlarmTime() {
    AlarmMode alm = new AlarmMode();
    for(int i = 0; i < 4 ; i++){
        assertEquals( expected: 0, alm.getAlarmTime( state: 12+i).getHour());
        assertEquals( expected: 0, alm.getAlarmTime( state: 12+i).getMinute());
    }
    // 현재 currentState 에서 알람의 hour, minute 값을 가져오는지
}
```

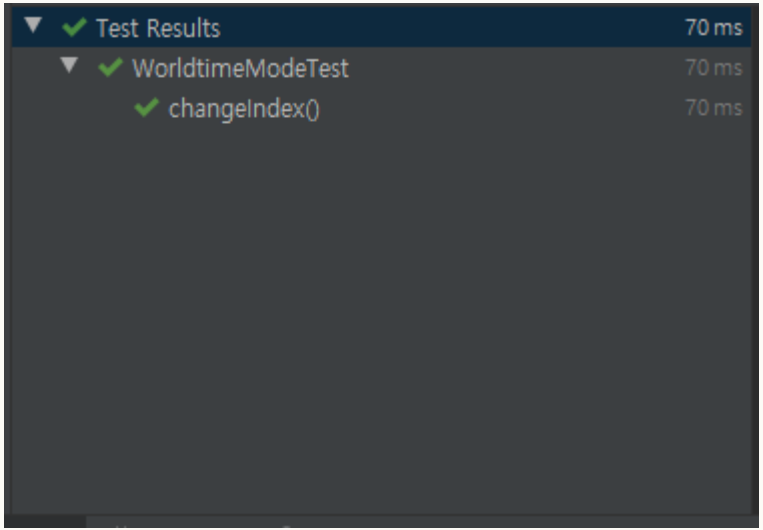
# Timer mode


```java
@Test
void resetTimer() {
    // 리셋버튼에 의해 0:0:0으로 초기화되는가
    TimerMode timer = new TimerMode();
    timer.setTimerTime( hour: 0 , min: 0, sec: 4);
    timer.resetTimer();
    assertEquals( expected: 0,timer.getTimerTime().getSecond());
    assertEquals( expected: 0,timer.getTimerTime().getMinute());
    assertEquals( expected: 0,timer.getTimerTime().getHour());
}
```

```java
@Test
void startTimer() {
    // timerTime의 값이 음수 값 이하로 측정되는지 가지는지
    TimerMode timer = new TimerMode();
    timer.setTimerTime( hour: 0 , min: 0, sec: 0);
    try {
        Thread.sleep( millis: 10);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
    assertEquals( expected: 0,timer.startTimer().getHour());
    assertEquals( expected: 0,timer.startTimer().getMinute());
    assertEquals( expected: 0,timer.startTimer().getSecond());

}
```

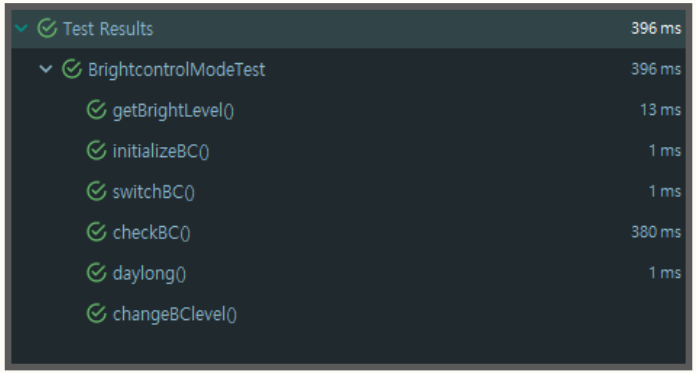
```java
@Test
void changeValue() {
    TimerMode timer = new TimerMode();
    // 최대/최소 값에서 증가/감소 버튼을 눌렀을 때, 최소값/최대값으로 변경되는지
    timer.setValue( currentState: 20, value: 99);
    timer.changeValue( currentState: 20, button: 1);
    assertEquals( expected: 0,timer.getTimerTime().getHour());

    timer.setValue( currentState: 21, value: 59);
    timer.changeValue( currentState: 21, button: 1);
    assertEquals( expected: 0,timer.getTimerTime().getMinute());

    timer.setValue( currentState: 22, value: 59);
    timer.changeValue( currentState: 22, button: 1);
    assertEquals( expected: 0,timer.getTimerTime().getSecond());

    timer.setValue( currentState: 20, value: 0);
    timer.changeValue( currentState: 20, button: 3);
    assertEquals( expected: 99,timer.getTimerTime().getHour());

    timer.setValue( currentState: 21, value: 0);
    timer.changeValue( currentState: 21, button: 3);
    assertEquals( expected: 59,timer.getTimerTime().getMinute());

    timer.setValue( currentState: 22, value: 0);
    timer.changeValue( currentState: 22, button: 3);
    assertEquals( expected: 59,timer.getTimerTime().getSecond());

    // timerTime이 잘 저장되는지
    timer.setTimerTime( hour: 1, min: 1, sec: 1);
    assertEquals( expected: 1,timer.getTimerTime().getHour());
    assertEquals( expected: 1,timer.getTimerTime().getMinute());
    assertEquals( expected: 1,timer.getTimerTime().getSecond());
}
```

## WorldtimeMode




```java
//"KOR", "USA", "UK", "CHI", "RSA", "SGP"
@Test
void changeIndex() {
    WorldtimeMode WTM = new WorldtimeMode( currIndex: 5);
    WTM.changeIndex();
    assertEquals( expected: "KOR",WTM.displayWorld());
    WTM.changeIndex();
    assertEquals( expected: "USA",WTM.displayWorld());
    WTM.changeIndex();
    assertEquals( expected: "UK",WTM.displayWorld());
    WTM.changeIndex();
    assertEquals( expected: "CHI",WTM.displayWorld());
    WTM.changeIndex();
    assertEquals( expected: "RSA",WTM.displayWorld());
    WTM.changeIndex();
    assertEquals( expected: "SGP",WTM.displayWorld());

}
```

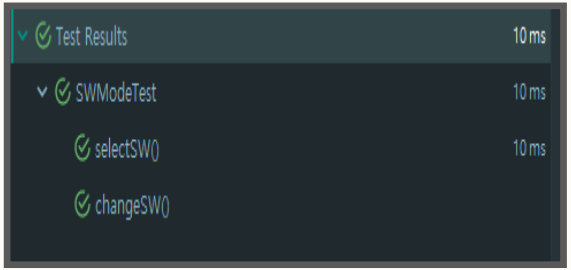
## Brightcontrol mode




```java
@Test
void checkBC() {
    // BC on/off가 한 버튼에 의해 적절히 토글되는가
    BrightcontrolMode blc = new BrightcontrolMode();
    DWS_controller dws = new DWS_controller();
    GUI gui =new GUI(dws);
    blc.setBrightness(true);
    assertEquals( expected: false,blc.checkBC());
}
```

```java
@Test
void changeBClevel() {
    // 밝기 레벨이 0-3 이외의 값을 가지는지
    BrightcontrolMode blc = new BrightcontrolMode();
    blc.setBrightness(true);
    blc.setBrightLevel(3);
    blc.changeBClevel();
    assertEquals( expected: 0, blc.getBrightLevel());
}

@Test
void daylong() {
    // 시간에 따라 적절히 BC Level이 변화되는지
    BrightcontrolMode blc = new BrightcontrolMode();
    blc.setBrightness(true);
    Time time = new Time();
    time.setHour(15);
    assertEquals( expected: 1,blc.daylong(time));

}
```

## SW select mode




```java
@Test
void changeSW() {
    // 1. 마지막 기능창에서 처음 기능창으로 넘어가는지
    SWM.setCurrentIndex(4);
    assertEquals( expected: 0,SWM.changeSW());
}

@Test
void selectSW() {
    // 2. 이미 선택된 index는 선택되지 않는지 확인
    String[][] setSW = {{"8","STM"},{"25","WLT"},{"19","Timer"}};
    SWM.setSelectedSW(setSW);
    SWM.setCurrentIndex(0);
    assertEquals( expected: -1,SWM.selectSW());
    // 3. 선택된 SW가 마지막 인덱스에 제대로 저장되는지 테스트
    SWM.setCurrentIndex(1);
    SWM.selectSW();
    assertEquals( expected: 12,Integer.parseInt(SWM.getSelectedSWIdx(2)));

}
```

# System Test

| No | Test 항목 | Description | Use case | System function | Pass |
|---|---|---|---|---|---|
| 1. | Set Time test | currentState=1일 때, 사용자의 incorrect한 입력에 반응하는지 | Set Time | R.1 | P |
| | | 년도가 1970년 아래로 내려가는지 | | | P |
| | | 월 값이 12월 이후에 1월도 돌아가는지 | | | P |
| | | month 값에 맞는 day 범위가 설정되는지 | | | P |
| | | Hour의 값이 24시간 이후에는 1로 돌아가는지 | | | P |
| | | Minute의 값이 60 이후에 00으로 돌아가는지 | | | P |
| | | KOREA 이전 나라 설정시 SINGAPORE로 가는지 | | | P |
| | | KOREA 이후 나라 설정시 AMERICA로 가는지 | | | P |
| | | 년도의 display가 잘 나오는지 | | | P |
| | | 월의 display가 잘 나오는지 | | | P |
| | | 일의 display가 잘 나오는지 | | | P |
| | | 시의 display가 잘 나오는지 | | | P |
| | | 분의 display가 잘 나오는지 | | | P |

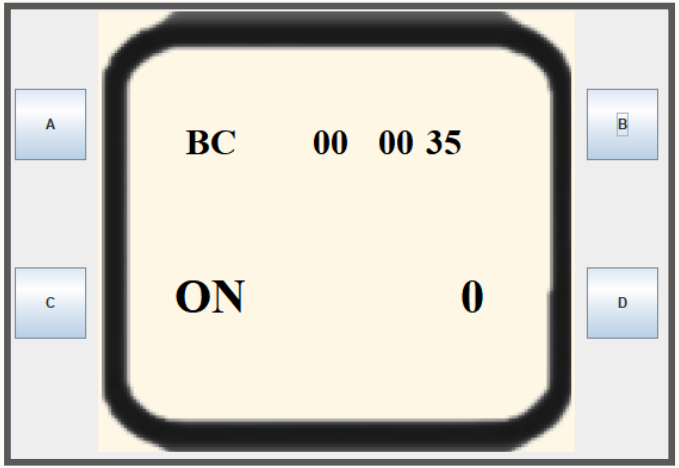
| | | | | | |
|---|---|---|---|---|---|
| 2 | Change mode test | 모드 전환이 순환형태로 잘 이루어지는지 | Change mode | R 1.3.1 | P |
| 3 | Stop ringing test | 사용자가 아무 버튼을 눌렀을 때, 소리가 꺼지는지 | Stop ringing | R 1.3.2 | F |
| | | 타 기능에서의 버튼 기능보다 우선적으로 기능하는지 | | | F |
| | | 알람이 울릴 때, 3초 뒤에 알람이 자동으로 OFF 되는지 | | | F |
| 4 | Start stopwatch test | 사용자가 B 버튼을 통해 스탑워치 시작 되는지 | Start stopwatch | R 2.1 | P |
| | | (Stop 모드) 일시정지된 상태에서 재개 되는지 | | | P |
| | | 1/100초 단위로 측정되는지 | | | F |
| | | 표시할 수 있는 범위 (59:59.99) 이상으로 측정이 계속될 때 멈추는 지 | | | P |
| 5 | Stop stopwatch test | (+) 사용자가 정지버튼을 통해 스탑워치가 정지 되는지 | Stop stopwatch | R 2.2 | P |
| | | (-) 지정된 버튼 이외의 입력으로 스탑워치가 정지 되는지 | | | P |
| | | (stop 모드) B 버튼을누르면  스탑워치 재개되는지 | | | P |

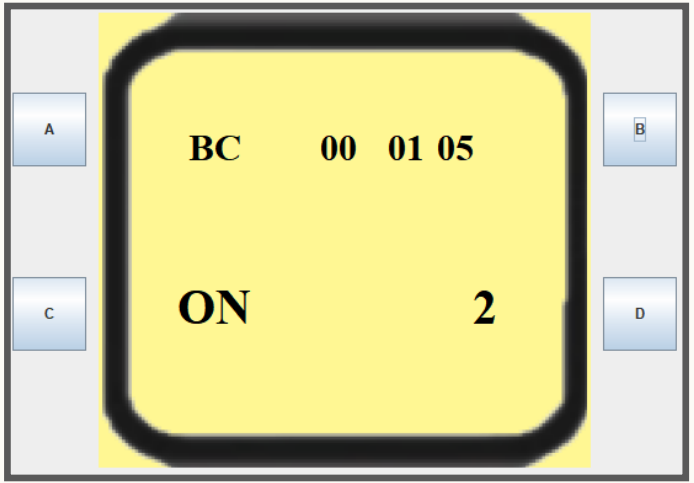
| | | | | | |
|---|---|---|---|---|---|
| 6 | Reset stopwatch test | 사용자가 리셋버튼을 통해 스탑워치 리셋(0:0:00)되는지 | Reset stopwatch | R 2.3 | P |
| 7 | Lap stopwatch test | 사용자가 랩 버튼을 눌렀을 때, 현재 lap time을 보여주는지 | Lap stopwatch | R 2.4 | P |
| | | (lap 모드) 시작버튼을 눌렀을 때, 현재 스탑워치 time을 보여주는지 | | | P |
| 8 | Select alarm number test | 4가지 알람  중 선택된 알람의 화면을 보여주는지 | Select alarm number | R 3.1 | P |
| 9 | Remove alarm number test | 4가지 알람  중 선택된 알람이 삭제되는지 | Remove alarm number | R 3.2 | P |
| 10 | set alarm time test | 사용자가 설정한 시간이 저장되는지 | Set alarm time | R 3.3 | P |
| | | 설정한 시간이 (23:59)내에서 설정이 되는지 | | | P |
| 11 | Turn on alarm test(indicator test) | 4가지 알람 중 하나라도 저장되어 있다면, indicator 가 켜지는지 | Turn on alarm | R 3.4 | P |
| 12 | Turn off alarm test | 4가지 알람 중 저장된 알람이 없다면, indicator 가 꺼지는지 | Turn off alarm | R 3.5 | P |

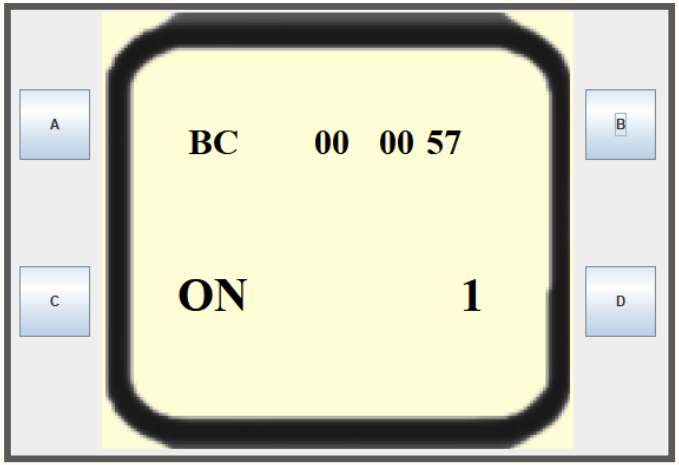
| | | | | | |
|---|---|---|---|---|---|
| 13 | Set timer time test | 사용자가 지정한 타이머 시간이 설정되는지 | set timer time | R 4.1 | P |
| 14 | Start timer test | 타이머의 시간이 감소되는지 | Start timer | R 4.2 | P |
| | | 다른 모드에서도 타이머가 작동하고 있는지 | | | F |
| | | 시간이 다 되었을때 감소하지 않는지 | | | P |
| 15 | Stop timer test | 타이머의 시간이 타이머를 정지했을 때 시간이 감소하지 않는지 | Stop timer | R 4.3 | P |
| | | 타이머의 시간이 정지 상태에서 다른 모드로 바꿨을 때 상태가 유지되는지 | | | F |
| 16 | Reset timer test | 타이머의 시간이 00:00:00 으로 바뀌는지 | Reset timer | R 4.4 | P |
| 17 | Ringing test | 사용자의 알람이 정시에 울리는지 | Ringing | R 4.5 | F |

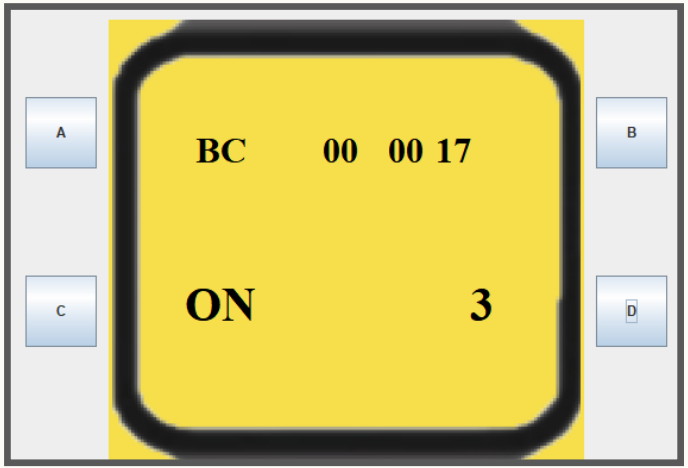
| | | | | | |
|---|---|---|---|---|---|
| 18 | ChangeWorld | B 버튼 시 나라순에 맞는 시간으로 바뀌는 지 | ChangeWorld | R 5.1 | P |
| 19 | Turn on bright control test | incorrect한 입력에 의해 켜지지 않는지 | Turn on brightness control | R 6.1 | P |
| 20 | Turn off bright control test | incorrect한 입력에 의해 꺼지지 않는지 | Turn off brightness control | R 6.2 | P |
| 21 | Control brightness test | 밝기 레벨 3단계에서 0단계로 바뀌는지 | Control brightness | R 6.3 | P |
| 22 | Change brightness test | 시간에 따른 밝기조절기능이 제대로 작동하는지 | Change brightness | | F |
| 23 | Change SW test | 이미 사용 중인 SW가 재선택되지 않는지 | Change SW | R 7.1 | P |
| | | 소프트웨어 선택이 선입선출의 원칙을 따르는지 | | | P |
| | | 마지막 SW 화면에서 C 버튼을 눌렀을 때, 첫 번째 SW 창으로 돌아가는지 | | | P |

# GUI


A
B
C
D
BC 00 00 35
ON 0


A
B
C
D
BC 00 01 05
ON 2


A
B
C
D
BC 00 00 57
ON 1


A
B
C
D
BC 00 00 17
ON 3

# Activity 2060 :
# Traceability

| Essential UseCase | S-Link | SID | Operation in Sequence Diagram | M-Link | MID | Method | Class | Unit Test |
|---|---|---|---|---|---|---|---|---|
| Set time | S1, S3, S21 | S1 | reqSetTime(button: int) | M1-8, M11 | M1 | setDisplay1() : void | **GUI** | |
| Change mode | S2, S21 | S2 | reqChangeMode() | M1-8 | M2 | setDisplay2() : void | | |
| Stop ringing | S4, S21 | S3 | reqChangeValue(button: int) | M1-8, M14, M24, M32 | M3 | setDisplay3() : void | | |
| Start stopwatch | S5, S21 | S4 | reqStopRinging() | M1-8, M25 | M4 | setDisplay4() : void | | |
| Stop stopwatch | S6, S21 | S5 | reqStartStopwatch() | M1-8, M16 | M5 | setDisplay5() : void | | |
| Reset stopwatch | S7, S21 | S6 | reqStopStopwatch() | M1-8, M17 | M6 | setDisplay6() : void | | |
| Lap stopwatch | S8, S21 | S7 | reqResetStopwatch() | M1-8, M18 | M7 | setDisplay7() : void | | enterSetSection() |
| Select alarm number | S9, S21 | S8 | reqLapStopwatch() | M1-8, M19 | M8 | setDisplay8() : void | | displayMinute() |
| Remove alarm number | S10, S21 | S9 | reqSelectAlarmNum() | M1-8 | M9 | watchBLC(int i): void | | changeValue() |
| Set Alarm Time | S3, S11, S21 | S10 | reqRemoveAlarmNum(currentState: int) | M1-8, M21 | M10 | ringing () : void | | displayDay() |
| Turn on alarm | | S11 | reqSetAlarmTime(button : int) | M1-8, M22 | M11 | enterSetSection(currentState: int) : int | **TimekeepingMode** | previousWorld() |
| Turn off alarm | | S12 | reqSetTimerTime(button : int) | M1-8, M28 | M12 | nextWorld(): void | | nextWorld() |
| Set timer time | S3, S12, S21 | S13 | reqStartTimer() | M1-8,M29 | M13 | previousWorld() void | | displayMonth() |
| Start timer | S13, S21 | S14 | reqStopTimer() | M1-8, M30 | M14 | changeValue(currentState: int, button: int) : void | | displayHour() |
| Stop timer | S14, S21 | S15 | reqResetTimer() | M1-8, M31 | M15 | changeMode(currentState: int) : void | | displayYear() |
| Reset timer | S15, S21 | S16 | reqChangeWorld() | M1-8, M33 | M16 | startStopwatch() : StopwatchTime | **StopwatchMode** | startStopwatch() |
| Ringing | | S17 | reqTurnOnBC() | M1-8, M9, M35, M39 | M17 | stopStopwatch() : StopwatchTime | | stopStopwatch() |
| Change World | S16, S21 | S18 | reqTurnOffBC() | M1-8, M9, M35, M39 | M18 | resetStopwatch() : StopwatchTime | | resetStopwatch() |
| Turn on brightness control | S17, S21 | S19 | reqControlBC() | M1-8, M9, M38 | M19 | lapTime() : StopwatchTime | | lapTime() |
| Turn off brightness control | S18, S21 | S20 | reqChangeSW(button: int) | M1-8, M41 | M20 | getAlarmTime() : Time | **AlarmMode** | changeValue() |
| Control brightness | S19, S21 | S21 | pressButton(button : int) | M1-8 | M21 | removeAlarmNum() : void | | removeAlarmNum() |
| Change brightness | S22 | S22 | controlAlarmIndicator() | M1-8, M23 | M22 | enterSetSection(currentState : int) : int | | isAlarmIndicator() |
| Change SW | S20, S21 | | | | M23 | checkAlarmArray() : boolean | | getAlarmTime() |
| | | | | | M24 | changeValue(currentState: int) : void | | |
| | | | | | M25 | check3Second() : boolean | **Ring** | |
| | | | | | M26 | checkAlarm() : boolean | | |
| | | | | | M27 | checkTimer() : boolean | | |
| | | | | | M28 | enterSetSection(currentState : int) : int | **TimerMode** | resetTimer() |
| | | | | | M29 | startTimer() : Time | | changeValue() |
| | | | | | M30 | stopTimer() : Time | | startTimer() |
| | | | | | M31 | resetTimer() : Time | | stopTimer() |
| | | | | | M32 | changeValue(currentState: int) : void | | |
| | | | | | M33 | changeIndex() : int | **WorldtimeMode** | changeIndex() |
| | | | | | M34 | changeWorldTime() : void | | changeBCLevel() |
| | | | | | M35 | checkBC() : boolean | **BrightcontrolMode** | getBrightLevel() |
| | | | | | M36 | switchBC() : void | | initializeBC() |
| | | | | | M37 | initializeBC() : void | | switchBC() |
| | | | | | M38 | changeBClevel : int | | checkBC() |
| | | | | | M39 | daylong(currentTime: Time) : int | | dayLong() |
| | | | | | M40 | changeSW() : int | **SWMode** | selectSW() |
| | | | | | M41 | enterChangeSW() : void | | changeSW() |
| | | | | | M42 | selectSW() : int | | |