Alarm Customizing Digital Watch

Stage 2050 & 2060 Implementation & Unit Test

201511172 컴퓨터공학부 강민호
201511257 컴퓨터공학부 남관우
201511271 컴퓨터공학부 신윤섭
201810502 컴퓨터공학부 전현지

Index

2051. Implement Class & Method Definitions

Type	Class
Name	System
Purpose	시계의 메인 시스템으로 사용자의 반응에 따라 적절한 기능을 수행하는 클래스
Overview(Class)	GUI 입력에 따라 적절하게 Model 값을 변경하고, 변경된 내용을 GUI에 반영한다.
Cross Reference	Function : All Use Cases : All
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	startCheckTimeOut
Purpose	Time out을 관찰하기 위한 쓰레드
Cross Reference	Function : R7.2 Use Cases : Time Out
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	modeBtnLongPressed
Purpose	Mode 버튼이 2초 눌렸을 때를 처리한다.
Cross Reference	Function : R7.3 Use Cases : Cancel
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	selectBtnLongPressed
Purpose	Select 버튼이 2초 눌렸을 때를 처리한다.
Cross Reference	Function : R3.6, R4.6, R5.4, R6.1 Use Cases : Control Stopwatch Record, Control Alarm List, Delete D-day, Control Alarm Custom List
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	startBtnPressed
Purpose	Start 버튼이 눌렸을 때를 처리한다.
Cross Reference	Function : R1.1, R1.3, R2.1, R2.2, R2.5, R2.6, R3.1, R3.2, R3.6, R4.1, R4.4, R4.6, R5.1, R5.3, R6.1, R6.2, R6.3 Use Cases : Set Time, Set Display, Set Timer, Start Timer, Pause Timer, Stop Timer Buzzer, Start Stopwatch, Pause Stopwatch, Control Stopwatch Record, Set Alarm, Stop Alarm Buzzer, Control Alarm List, Set D-day, Stop D-day Border, Control Alarm Custom List, Set Alarm Interval, Set Alarm Volume
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	resetBtnPressed
Purpose	Reset 버튼이 눌렸을 때를 처리한다.
Cross Reference	Function : R1.1, R1.3, R2.1, R2.2, R2.4, R2.5, R2.6, R3.1, R3.2, R3.3, R3.6, R4.1, R4.4, R4.6, R5.1, R5.3, R6.1, R6.2, R6.3 Use Cases : Set Time, Set Display, Set Timer, Start Timer, Reset Timer, Pause Timer, Stop Timer Buzzer, Start Stopwatch, Pause Stopwatch, Reset Stopwatch, Control Stopwatch Record, Set Alarm, Stop Alarm Buzzer, Control Alarm List, Set D-day, Stop D-day Border, Control Alarm Custom List, Set Alarm Interval, Set Alarm Volume
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	selectBtnPressed
Purpose	Select 버튼이 눌렸을 때를 처리한다.
Cross Reference	Function : R1.1, R1.3, R2.1, R2.6, R3.4, R4.1, R4.2, R4.4, R5.1, R5.3, R6.1, R6.2, R6.3 Use Cases : Set Time, Set Display, Set Timer, Stop Timer Buzzer, Record Stopwatch, Set Alarm, Delete Alarm, Stop Alarm Buzzer, Set D-day, Stop D-day Border, Control Alarm Custom List, Set Alarm Interval, Set Alarm Volume
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	modeBtnPressed
Purpose	Mode 버튼이 눌렸을 때를 처리한다.
Cross Reference	Function : R1.1, R1.3, R2.1, R2.6, R4.1, R4.4, R5.1, R5.3, R6.2, R6.3, R7.1 Use Cases : Set Time, Set Display, Set Timer, Stop Timer Buzzer, Set Alarm, Stop Alarm Buzzer, Set D-day, Stop D-day Border, Set Alarm Interval, Set Alarm Volume, Change Screen
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestFunctionSettingMode
Purpose	6개 기능 중 4개 기능을 선택하는 모드로 전환한다.
Cross Reference	Function : R1.3 Use Cases : Set Display
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	setFunction
Purpose	사용자가 선택한 4개 기능을 저장한다.
Cross Reference	Function : R1.3 Use Cases : Set Display
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	beepBuzzer
Purpose	버저를 울린다.
Cross Reference	Function : R2.3, R 4.3 Use Cases : Beep Timer, Beep Alarm
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	updateState
Purpose	현재 Border 혹은 Buzzer가 실행중일 경우, 우선순위에 맞게 종료하고 상태를 갱신한다.
Cross Reference	Function : R2.6, R4.4, R5.2, R5.3, Use Cases : Stop Timer Buzzer, Stop Alarm Buzzer, Border D-day, Stop D-day Border,
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	nextFunction
Purpose	다음 화면으로 넘어간다.
Cross Reference	Function : R7.1 Use Cases : Change Screen
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Class
Name	TimeKeeping
Purpose	시계의 현재 시간을 보여주는 기본 화면. 흐르는 시간을 화면에 표시한다.
Overview(Class)	
Cross Reference	Function : R1.1 Use Cases : Set Time
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	setDayOfTheWeek
Purpose	현재 날짜에 맞는 요일을 설정한다.
Cross Reference	Function : R1.1 Use Cases : Set Time
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Class
Name	Time
Purpose	시간의 흐름을 제어하는 클래스.
Overview(Class)	시간을 1초씩 늘리거나, 줄이는 쓰레드를 제어하는 클래스.
Cross Reference	Function : R2.2, R2.3, R2.4, R2.5, R3.1, R3.2, R3.3, R4.3, R7.2 Use Cases : Start Timer, Beep Timer, Reset Timer, Pause Timer, Start Stopwatch, Pause Stopwatch, Reset Stopwatch, Beep Alarm, Time Out
Exceptional Courses of Event	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	startTime
Purpose	시간 흐름을 시작한다.
Cross Reference	Function : R2.2, R3.2, R7.2 Use Cases : Start Timer, Start Stopwatch, Time Out
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	pauseTime
Purpose	시간 흐름을 일시정지한다.
Cross Reference	Function : R2.4, R2.5, R3.2, R3.3, R7.2 Use Cases : Reset Timer, Pause Timer, Pause Stopwatch, Reset Stopwatch, Time Out
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	clearTime
Purpose	Time 클래스 내 시, 분, 초를 0으로 초기화한다.
Cross Reference	Function : R2.4, R3.3, Use Cases : Reset Timer, Reset Stopwatch
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Class
Name	Date
Purpose	날짜 정보를 저장하는 클래스.
Overview(Class)	현재 날짜를 저장, Time과 상호작용하여 날짜를 증가시킨다.
Cross Reference	Function : R1.1, R5.1, R5.2, R5.4 Use Cases : Set Time, Set D-day, Border D-day, Delete D-day
Exceptional Courses of Event	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	raiseDate
Purpose	현재 시간이 23:59:59에서 00:00:00으로 전환될 때, 일을 증가시킨다.
Cross Reference	Function : R5.2 Use Cases : Border D-day
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	deleteDday
Purpose	현재 설정된 Dday를 삭제한다.
Cross Reference	Function : R5.4 Use Cases : Delete D-day
Input	void
Output	void
Abstract operation	N/A
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Class
Name	Time
Purpose	시간 정보를 저장하는 클래스.
Overview(Class)	시간을 저장하고, 현재 시간의 경우 초마다 시간이 증가한다.
Cross Reference	Function : R1.1, R2.1, R2.2, R2.4, R2.5, R3.1, R3.2, R3.3, R3.4 Use Cases : Set Time, Set Timer, Start Timer, Reset Timer, Pause Timer, Start Stopwatch, Pause Stopwatch, Reset Stopwatch, Record Stopwatch
Exceptional Courses of Event	N/A

2051. Implement Class & Method Definitions

Type	Class
Name	Timer
Purpose	타이머와 관련된 값을 저장, 관리하고 타이머와 관련된 적절한 기능을 수행하는 클래스
Overview(Class)	설정된 타이머의 시간을 가지고 있다.
Cross Reference	Function : R2.1, R2.2, R2.3, R2.4, R2.5, R2.6 Use Cases : Set Timer, Start Timer, Beep Timer, Reset Timer, Pause Timer, Stop Timer Buzzer
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestTimerSettingMode
Purpose	Timer의 시간을 조정하는 모드로 전환해준다.
Cross Reference	Function : R2.1 Use Cases : Set Timer
Input	
Output	void
Abstract operation	시간을 조정할 수 있는 모드(1)로 변경해준다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestSave
Purpose	사용자가 입력한 Timer의 시간을 저장한다.
Cross Reference	Function : R2.1 Use Cases : Set Timer
Input	
Output	void
Abstract operation	입력받은 시간을 timer의 시간에 저장한 후 모드 0으로 돌아온다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestStartTimer
Purpose	Timer의 시간이 흐르기 시작하도록 요청한다.
Cross Reference	Function : R2.2 Use Cases : Start Timer
Input	
Output	void
Abstract operation	저장되었던 시간에서부터 1초 단위로 시간이 줄어든다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestResetTimer
Purpose	Timer의 시간을 초기화하도록 요청한다.
Cross Reference	Function : R2.4 Use Cases : Reset Timer
Input	
Output	void
Abstract operation	Timer의 시간을 0시간 0분 0초로 바꾼다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestPauseTimer
Purpose	Timer의 시간이 흐르던 것을 일시정지하도록 요청한다.
Cross Reference	Function : R2.5 Use Cases : Pause Timer
Input	
Output	void
Abstract operation	Timer의 시간이 줄어드는 것을 멈춘다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Class
Name	Stopwatch
Purpose	스톱워치와 관련된 값을 저장, 관리하고 스톱워치와 관련된 적절한 기능을 수행하는 클래스
Overview(Class)	스톱워치에서 흐른 시간을 가지고있다.
Cross Reference	Function : R3.1, R3.2, R3.3, R3.4, R3.5, R3.6 Use Cases : Start Stopwatch, Pause Stopwatch, Reset Stopwatch, Record Stopwatch, Display Stopwatch Record, Control Stopwatch Record
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestStartStopwatch
Purpose	Stopwatch의 시간이 흐르기 시작하도록 요청한다.
Cross Reference	Function : R3.1 Use Cases : Start Stopwatch
Input	
Output	void
Abstract operation	Stopwatch의 시간이 1초 단위로 증가한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestPauseStopwatch
Purpose	Stopwatch의 시간이 흐르던 것을 멈추도록 요청한다.
Cross Reference	Function : R3.2 Use Cases : Pause Stopwatch
Input	
Output	void
Abstract operation	Stopwatch의 시간이 증가하던 것을 멈춘다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestResetStopwatch
Purpose	Stopwatch의 초기화를 요청한다.
Cross Reference	Function : R3.3 Use Cases : Reset Stopwatch
Input	
Output	void
Abstract operation	Stopwatch의 시간을 0시 0분 0초로 초기화한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	clearList
Purpose	Stopwatch의 기록 초기화를 요청한다.
Cross Reference	Function : R3.3 Use Cases : Reset Stopwatch
Input	
Output	void
Abstract operation	Stopwatch의 시간을 기록했던 것을 초기화한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestSaveRecord
Purpose	Stopwatch의 현재 시간 기록을 요청한다.
Cross Reference	Function : R3.4 Use Cases : Record Stopwatch
Input	
Output	void
Abstract operation	Stopwatch의 현재 시간을 stopwatchRecord에 저장한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	record
Purpose	Stopwatch의 현재 시간을 기록한다.
Cross Reference	Function : R3.4 Use Cases : Record Stopwatch
Input	time: String
Output	void
Abstract operation	stopwatchRecord에 Input값을 저장한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestRecordCheckMode
Purpose	Stopwatch에 저장했던 기록들을 볼 수 있는 모드로 변경을 요청한다.
Cross Reference	Function : R3.6 Use Cases : Control Stopwatch Record
Input	
Output	void
Abstract operation	Stopwatch의 기록들을 볼 수 있는 모드(2)로 변경해준다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Class
Name	AlarmData
Purpose	알람과 관련된 값을 저장하는 클래스
Overview(Class)	알람으로 저장한 시간, 간격, 볼륨을 가지고 있다.
Cross Reference	Function : R4.1, R4.2, R4.3, R4.4, R4.5, R4.6 Use Cases : Set Alarm, Delete Alarm, Beep Alarm, Stop Alarm Buzzer, Display Alarm List, Control Alarm List
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Class
Name	Alarm
Purpose	알람과 관련된 값을 저장, 관리하고 알람과 관련된 적절한 기능을 수행하는 클래스
Overview(Class)	알람으로 저장한 시간을 갖고 있다.
Cross Reference	Function : R4.1, R4.2, R4.3, R4.4, R4.5, R4.6 Use Cases : Set Alarm, Delete Alarm, Beep Alarm, Stop Alarm Buzzer, Display Alarm List, Control Alarm List
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestAlarmSettingMode
Purpose	Alarm의 시간을 조정하는 모드로 전환해준다.
Cross Reference	Function : R4.1 Use Cases : Set Alarm
Input	
Output	void
Abstract operation	알람을 조정할 수 있는 모드(1)로 변경해준다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	addTimeToAlarmList
Purpose	저장했던 alarm 시간을 alarmList에 추가한다.
Cross Reference	Function : R4.1 Use Cases : Set Alarm
Input	alarmTime:Time
Output	void
Abstract operation	alarmList에 알람을 추가한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestSave
Purpose	설정했던 alarm의 시간을 저장한다.
Cross Reference	Function : R4.1 Use Cases : Set Alarm
Input	
Output	void
Abstract operation	alarm의 시간을 저장한 후 다시 모드 0으로 돌아온다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestDeleteAlarm
Purpose	설정했던 alarm의 시간 삭제를 요청한다.
Cross Reference	Function : R4.2, R4.6 Use Cases : Delete Alarm, Control Alarm List
Input	
Output	void
Abstract operation	alarmPointer에 해당하는 알람 시간을 삭제한 후 모드를 0으로 바꾼다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	deleteAlarm
Purpose	설정했던 alarm의 시간을 삭제한다.
Cross Reference	Function : R4.2, R4.6 Use Cases : Delete Alarm, Control Alarm List
Input	alarmIdx:Int
Output	void
Abstract operation	선택한 알람 시간을 삭제한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestStopAlarmBuzzer
Purpose	울리고 있는 alarm buzzer를 멈춘다.
Cross Reference	Function : R4.4 Use Cases : Stop Alarm Buzzer
Input	
Output	void
Abstract operation	buzzer를 멈춘다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestAlarmSelectMode
Purpose	알람을 선택할 수 있는 모드로 진입하는 것을 요청한다.
Cross Reference	Function : R4.6 Use Cases : Control Alarm List
Input	
Output	void
Abstract operation	알람을 선택할 수 있는 모드(2)로 변경한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Class
Name	D_Day
Purpose	d-day와 관련된 값을 저장, 관리하고 d=day와 관련된 적절한 기능을 수행하는 클래스
Overview(Class)	설정된 타이머의 시간을 가지고 있다.
Cross Reference	Function : R5.1, R5.2, R5.3, R5.4 Use Cases : Set D-day, Stop D-day Border, Delete D-day, Border D-day
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestDdaySettingMode
Purpose	d-day를 제어하는 모드를 요청한다.
Cross Reference	Function : R5.1 Use Cases : Set D-day
Input	
Output	void
Abstract operation	시간을 조정할 수 있는 모드로 초기화한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestSave
Purpose	설정한 d-day를 저장한다.
Cross Reference	Function : R5.1 Use Cases : Set D-day
Input	
Output	void
Abstract operation	d-day 날짜를 기록하고 d-day까지의 일 수를 기록한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	setDate
Purpose	설정한 d-day를 저장한다.
Cross Reference	Function : R5.1 Use Cases : Set D-day
Input	
Output	void
Abstract operation	d-day 날짜를 기록하고 d-day까지의 일 수를 기록한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestDeleteDday()
Purpose	현재 D-day를 초기화한다.
Cross Reference	Function : R5.4 Use Cases : Delete D-day
Input	
Output	void
Abstract operation	d-day까지의 일 수 역시 초기화한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Class
Name	AlarmCustom
Purpose	타이머와 관련된 값을 저장, 관리하고 타이머와 관련된 적절한 기능을 수행하는 클래스
Overview(Class)	설정된 타이머의 시간을 가지고 있다.
Cross Reference	Function : R6.1, R6.2, R6.3 Use Cases : Control Alarm Custom List, Set Alarm Interval, Set Alarm Volume
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestAlarmSelectMode
Purpose	알람 선택 모드로 전환한다.
Cross Reference	Function : R5.1 Use Cases : Set D-day
Input	void
Output	void
Abstract operation	AlarmCustom 클래스의 mode 변수를 1로 설정하고, interval 및 volume 설정에 사용할 변수를 초기화한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestIntervalSettingMode
Purpose	간격 설정 모드로 전환한다.
Cross Reference	Function : R6.2 Use Cases : Set Alarm Interval
Input	void
Output	void
Abstract operation	AlarmCustom 클래스의 mode 변수를 2로 설정한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestVolumeSettingMode
Purpose	볼륨 설정 모드로 전환한다.
Cross Reference	Function : R6.2 Use Cases : Set Alarm Volume
Input	void
Output	void
Abstract operation	AlarmCustom 클래스의 mode 변수를 2로 설정한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	setCustom
Purpose	사용자가 선택한 알람의 간격, 볼륨을 저장한다.
Cross Reference	Function : R6.2, R6.3 Use Cases : Set Alarm Interval, Set Alarm Volume
Input	void
Output	void
Abstract operation	Alarm 인스턴스를 참조하여 사용자가 선택한 알람에 사용자가 조절한 간격, 볼륨을 반영한다.
Exceptional Courses of Events	N/A

2051. Implement Class & Method Definitions

Type	Method
Name	requestSave
Purpose	사용자가 조정한 간격, 볼륨을 저장하기를 요청한다.
Cross Reference	Function : R6.2, R6.3 Use Cases : Set Alarm Interval, Set Alarm Volume
Input	void
Output	void
Abstract operation	내부적으로 setCustom() 을 호출하여 현재 조절한 간격, 볼륨을 반영한다. 이후 AlarmCutom의 기본모드로 되돌린다.
Exceptional Courses of Events	N/A

2052. Implements Windows

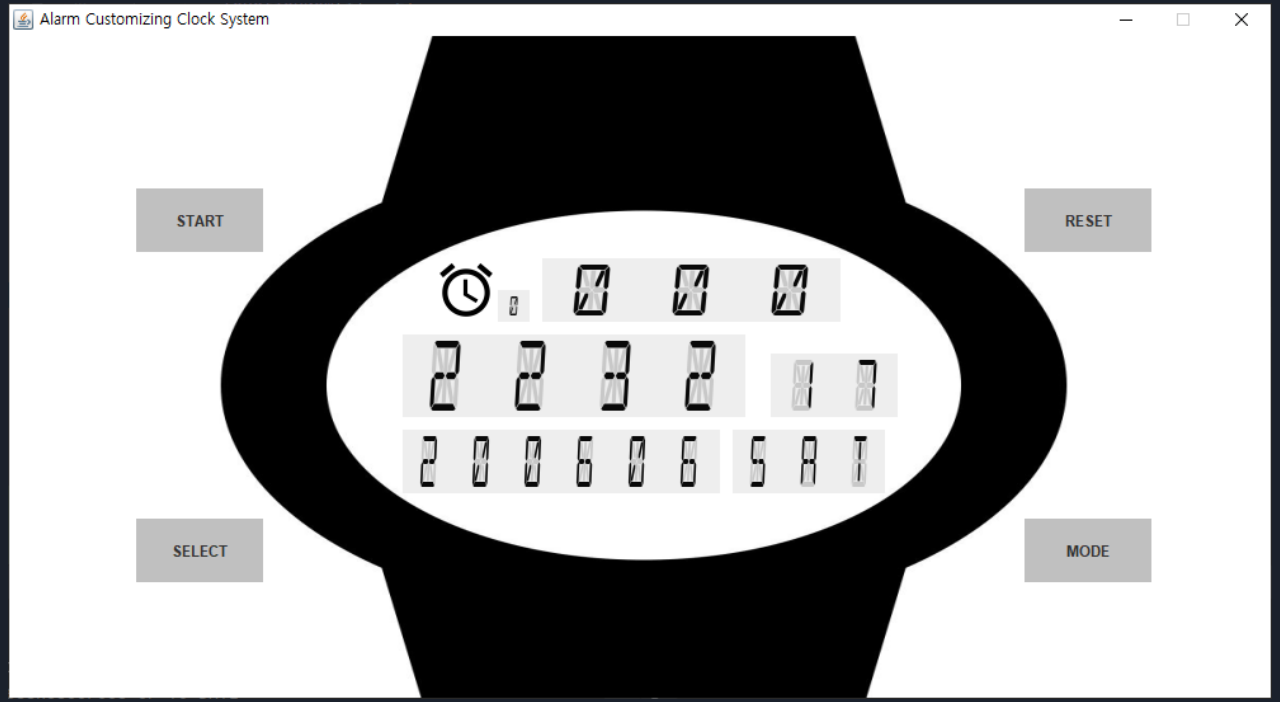
- 비고

해당 프로그램에서 클래스 System이 사실상 Controller와 GUI의 역할을 하고 있으므로 Implements Window의 시퀀스는, System에서 호출하는 자신의 함수를 빼고 기존 시퀀스 다이어그램의 연장선이 된다.

따라서 기존 사용자와 System간의 메시지, 함수 호출, 리턴 값을 유지한 채로 그 사이에 GUI가 추가된다.

2052. Implements Windows

- TimeKeeping

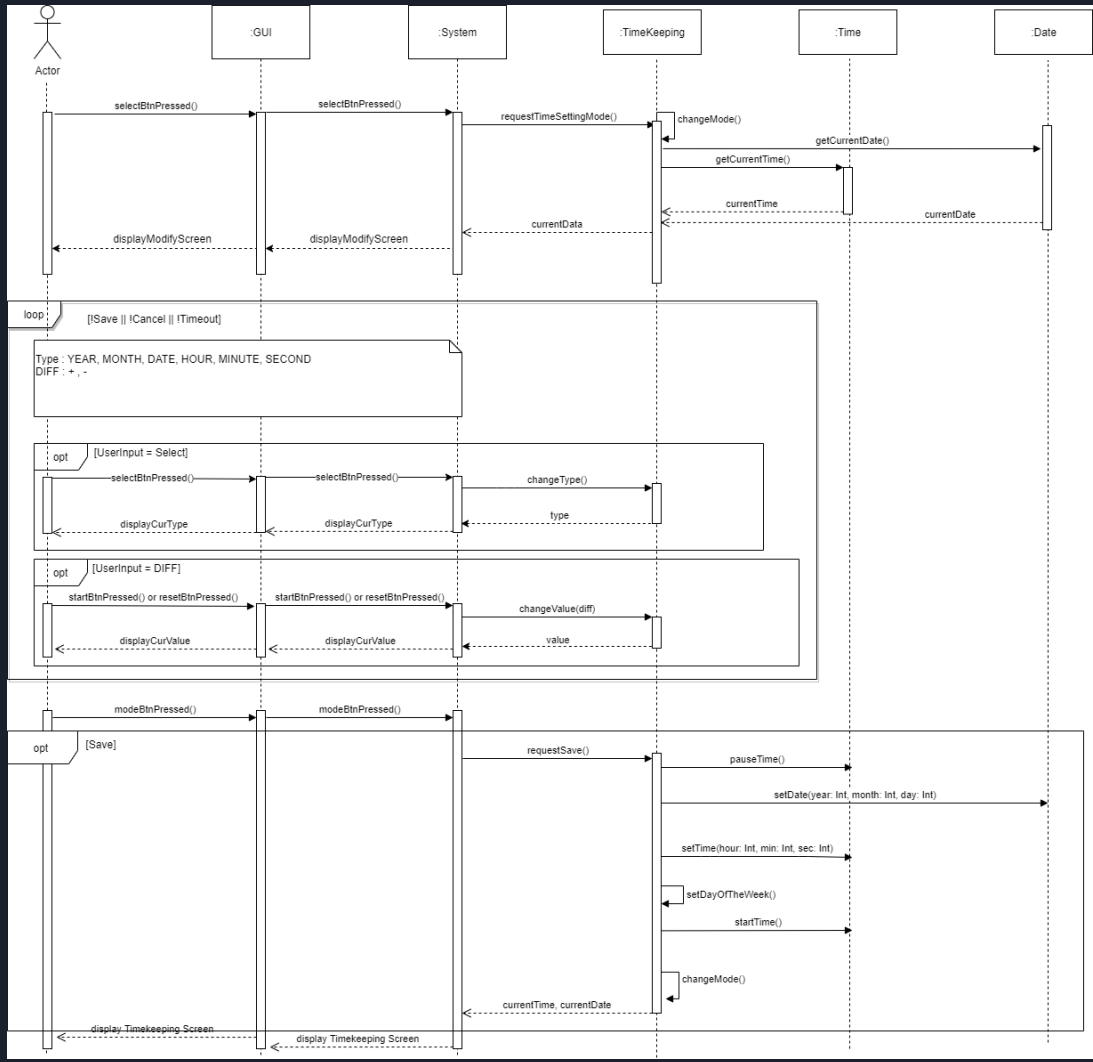


2052. Implements Windows

- TimeKeeping

1. Set Time

Type	GUI
Responsibilites	현재 시각을 설정한다.
Cross Reference	Function : R1.1
Note	입력한대로 현재 시간이 설정된다.
Pre-Conditions	TimeKeeping 화면이어야 한다.
Post-Conditions	설정한 현재시간에서부터 1초씩 시간이 흐른다.

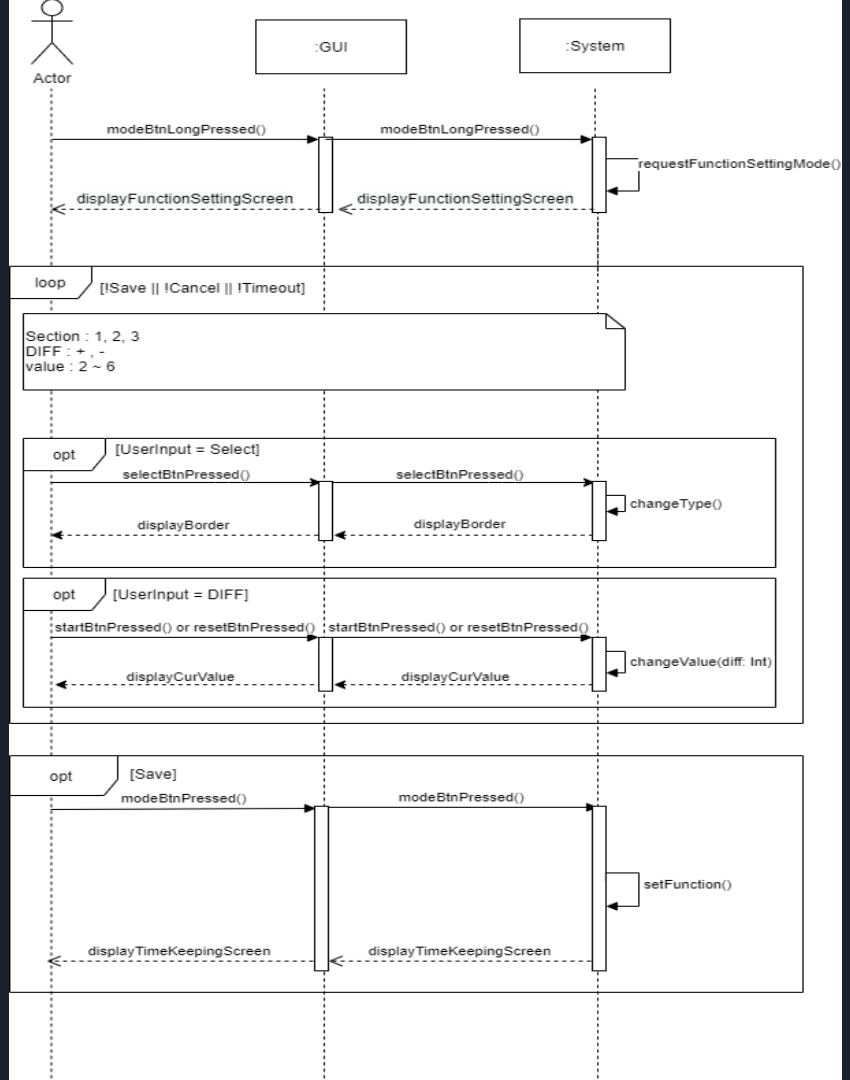


2052. Implements Windows

- TimeKeeping

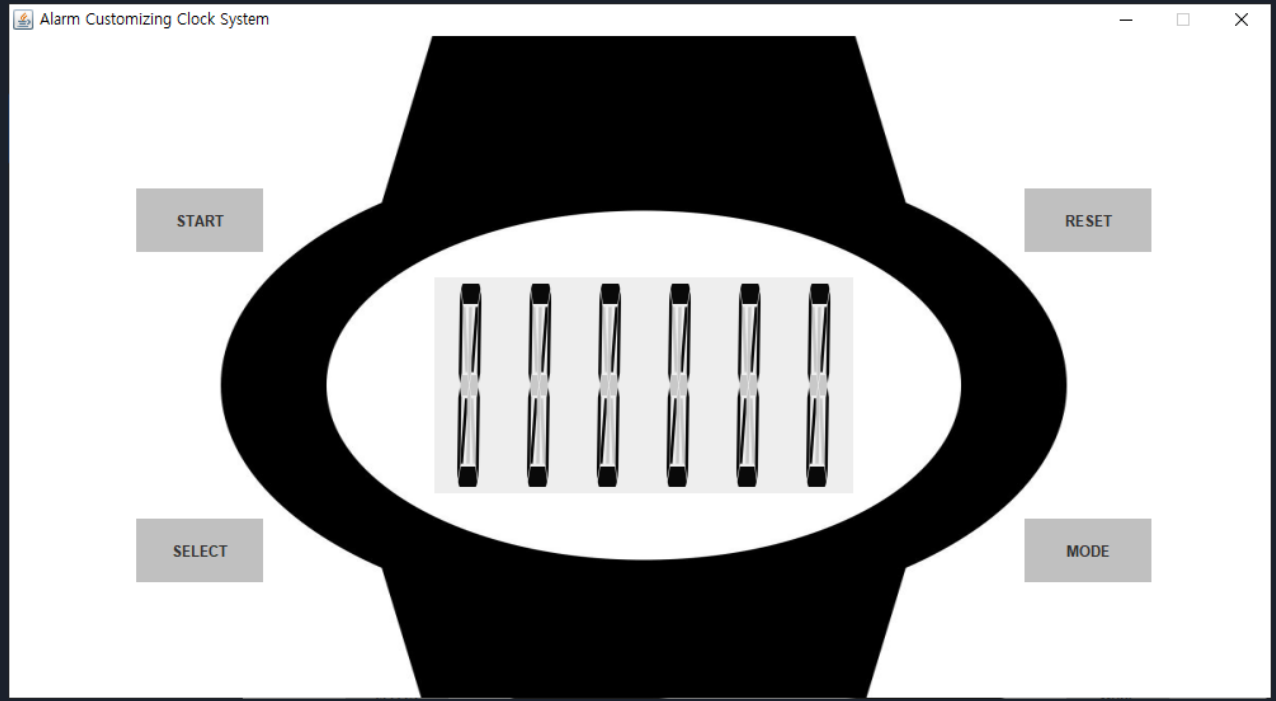
3. Set Display

Type	GUI
Responsibilites	4가지 기능을 입력한다.
Cross Reference	Function : R1.3
Note	6가지 기능들 중 설정한 4가지 기능을 저장한다.
Pre-Conditions	TimeKeeping 화면이어야 한다.
Post-Conditions	mode전환 버튼을 누를 때마다 설정했던 기능이 순서대로 나온다.



2052. Implements Windows

- Timer

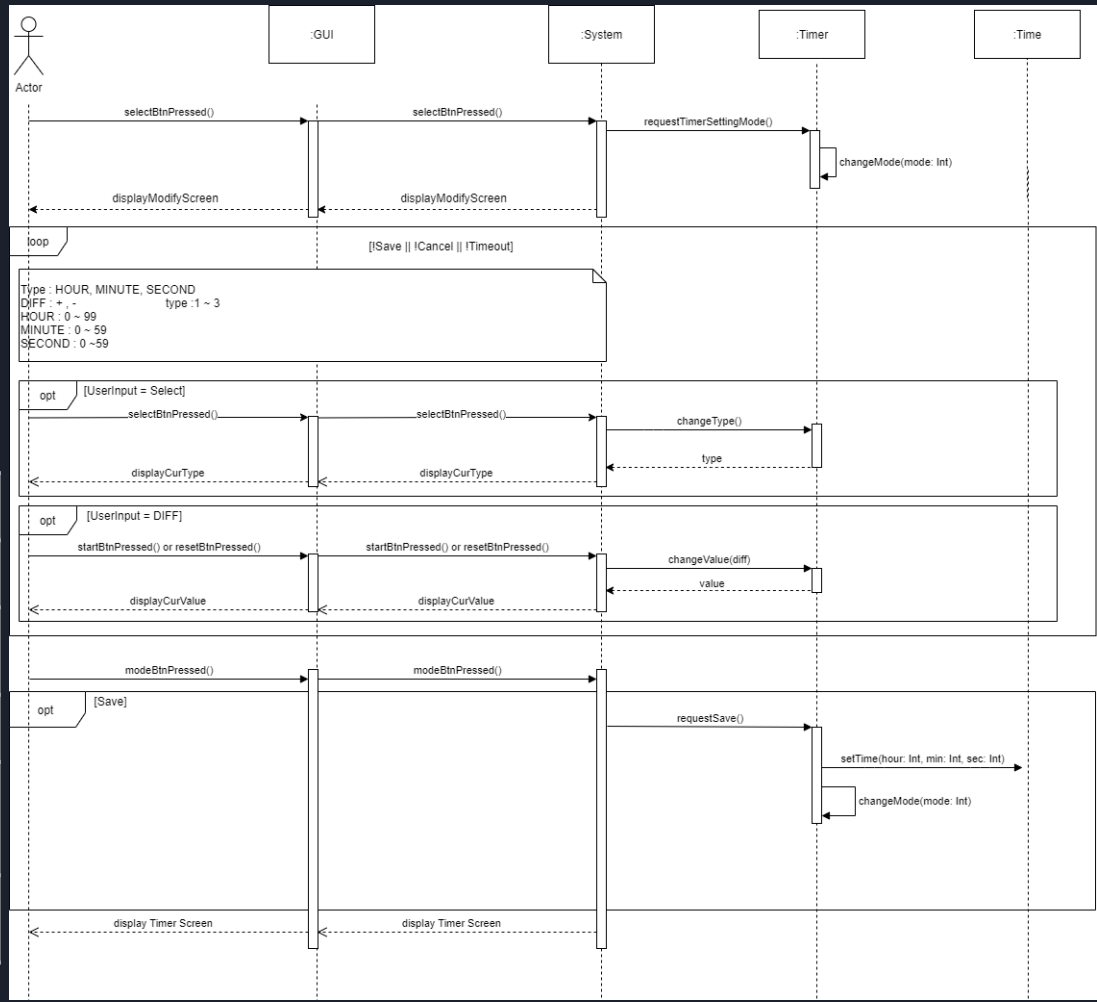


2052. Implements Windows

- Timer

4. Set Timer

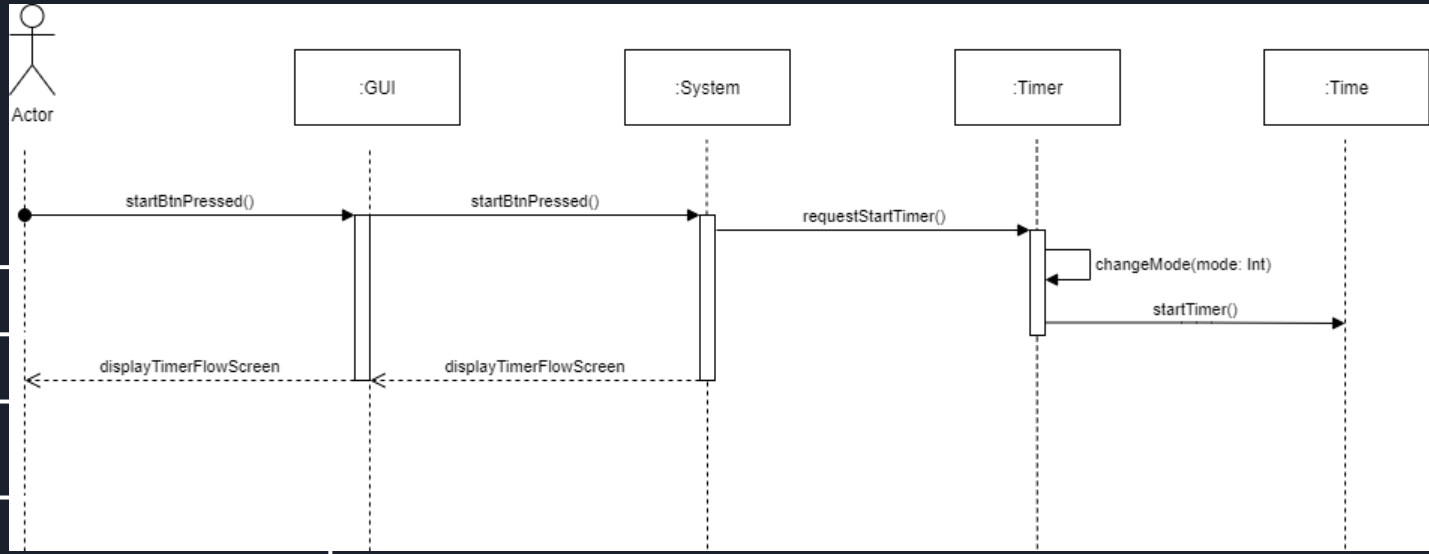
Type	GUI
Responsibilites	타이머를 설정한다.
Cross Reference	Function : R2.1
Note	입력한 대로 타이머가 설정된다.
Pre-Conditions	타이머 화면이어야 한다.
Post-Conditions	설정된 타이머값이 화면에 표시된다.



2052. Implements Windows

- Timer

5. Start Timer

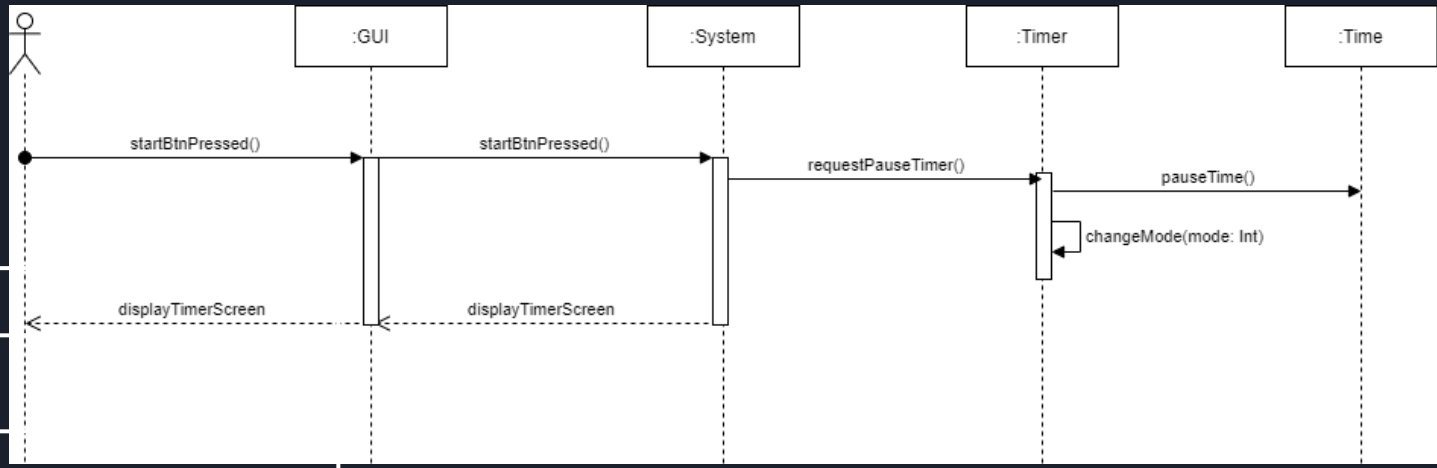


Type	GUI
Responsibilites	타이머를 시작한다.
Cross Reference	Function : R2.2
Note	Start버튼을 눌러 시작한다.
Pre-Conditions	타이머 화면이어야 한다. 타이머 값이 설정되어야한다.
Post-Conditions	설정값으로부터 1초마다 카운트다운이 표시된다.

2052. Implements Windows

- Timer

7. Reset Timer

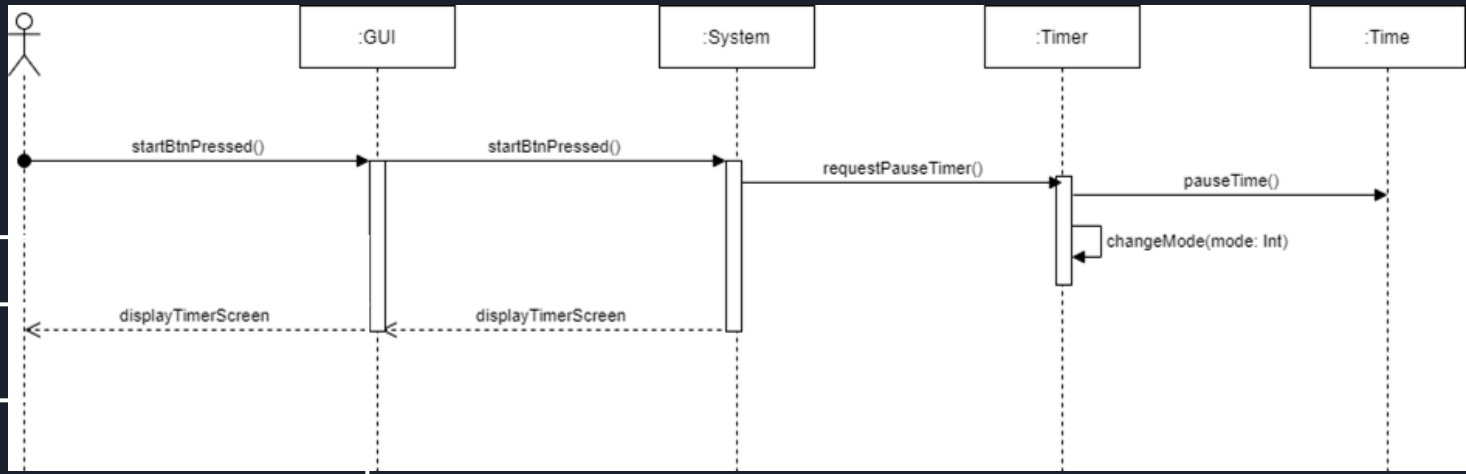


Type	GUI
Responsibilites	타이머를 초기화 한다.
Cross Reference	Function : R2.3
Note	Reset버튼을 눌러 초기화한다.
Pre-Conditions	타이머 화면이어야한다
Post-Conditions	00:00:00으로 초기화된 타이머가 표시된다.

2052. Implements Windows

- Timer

8. Pause Timer

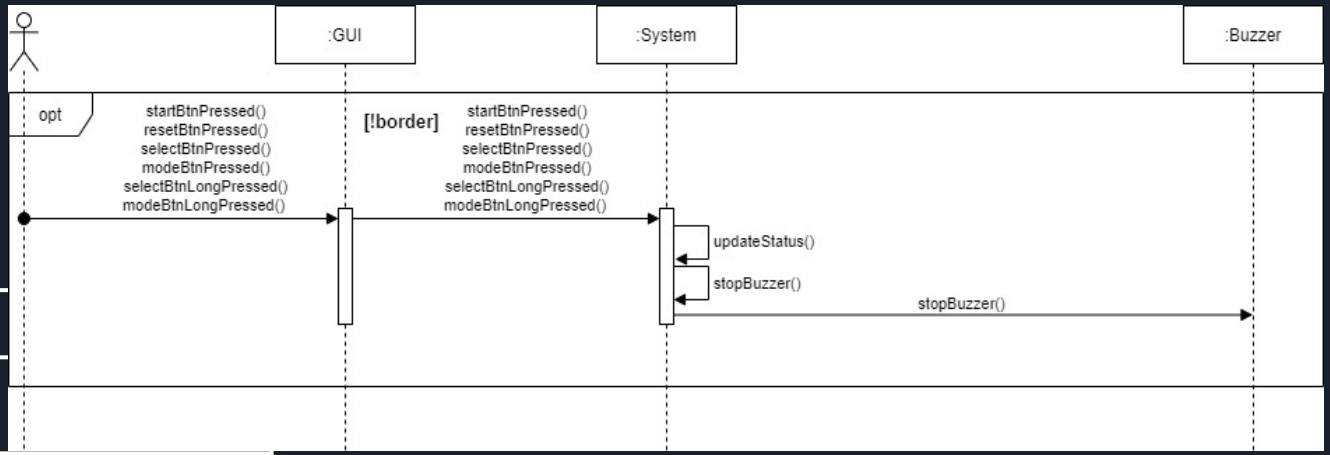


Type	GUI
Responsibilites	타이머를 일시정지 한다.
Cross Reference	Function : R2.5
Note	다시 Start버튼을 눌러 일시정지 한다.
Pre-Conditions	타이머 화면이어야한다. 타이머가 Start된 상태여야한다.
Post-Conditions	일시정지한 순간의 타이머 값이 표시된다.

2052. Implements Windows

- Timer

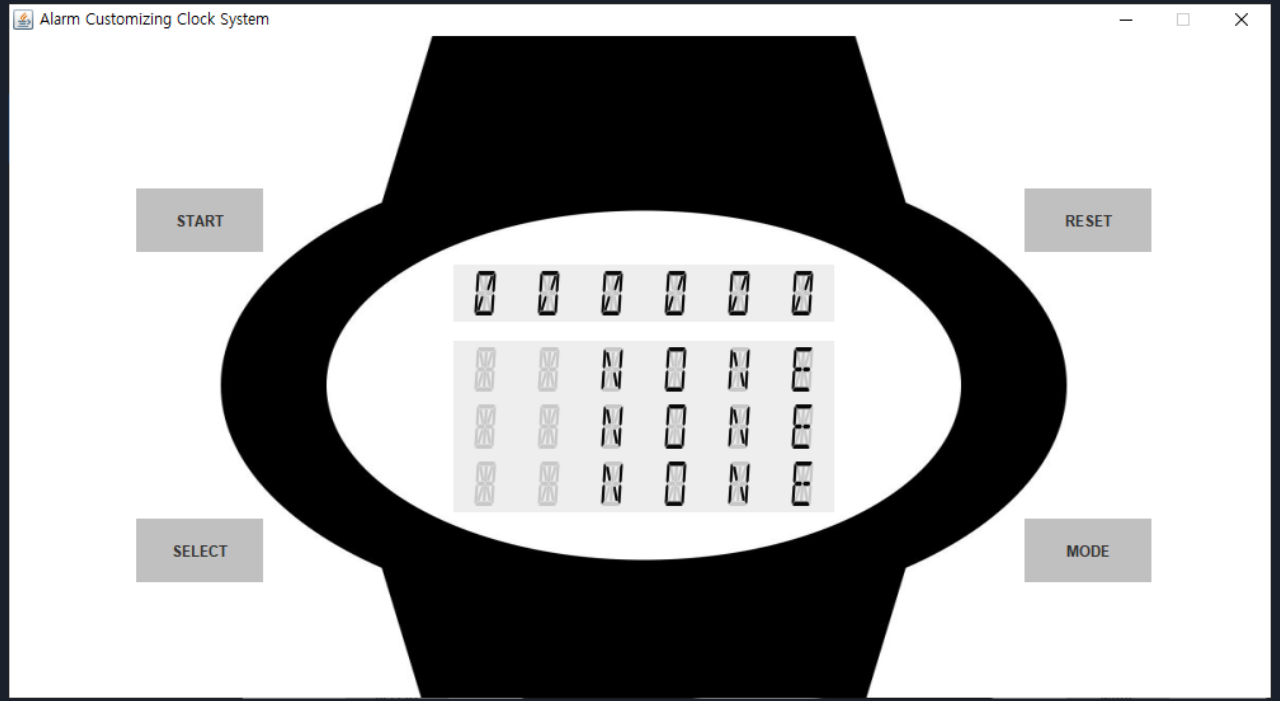
9. Stop Timer Buzzer



Type	GUI
Responsibilites	타이머로 울리는 버저를 멈춘다.
Cross Reference	Function : R2.6
Note	아무 버튼이나 눌러 버저를 멈춘다.
Pre-Conditions	타이머 화면이어야 한다. 버저가 울리는 상태이어야 한다.
Post-Conditions	울리던 버저가 멈춘다.

2052. Implements Windows

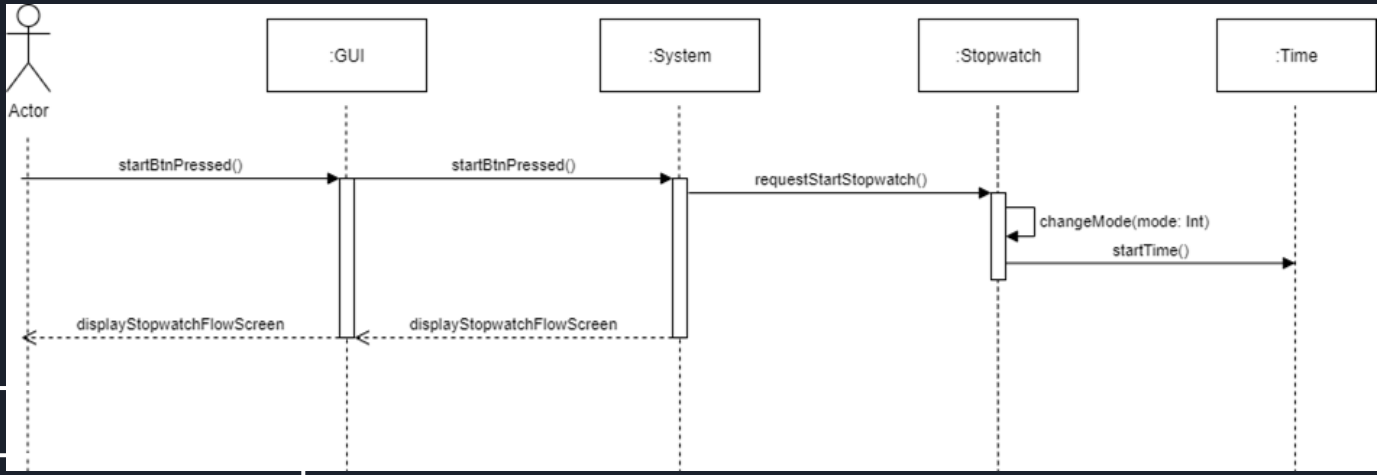
- Stopwatch



2052. Implements Windows

- Stopwatch

10. Start Stopwatch

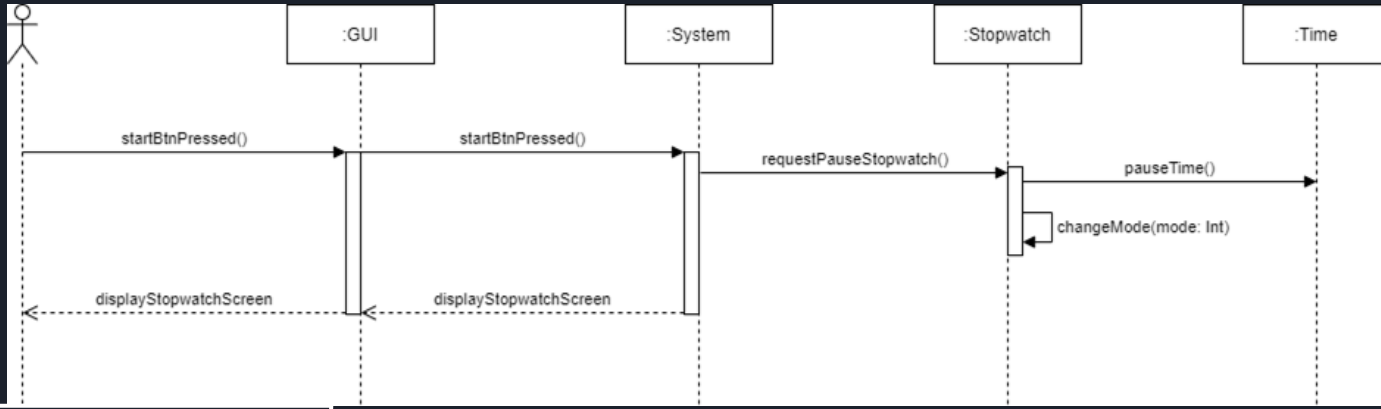


Type	GUI
Responsibilites	버튼을 눌러 스톱워치를 시작시킨다.
Cross Reference	Function : R3.1
Note	스톱워치의 시간이 흐른다.
Pre-Conditions	스톱워치 화면이어야 한다.
Post-Conditions	해당 시간에서부터 1초씩 시간이 흐르는 것이 화면에 보인다.

2052. Implements Windows

- Stopwatch

11. Pause Stopwatch

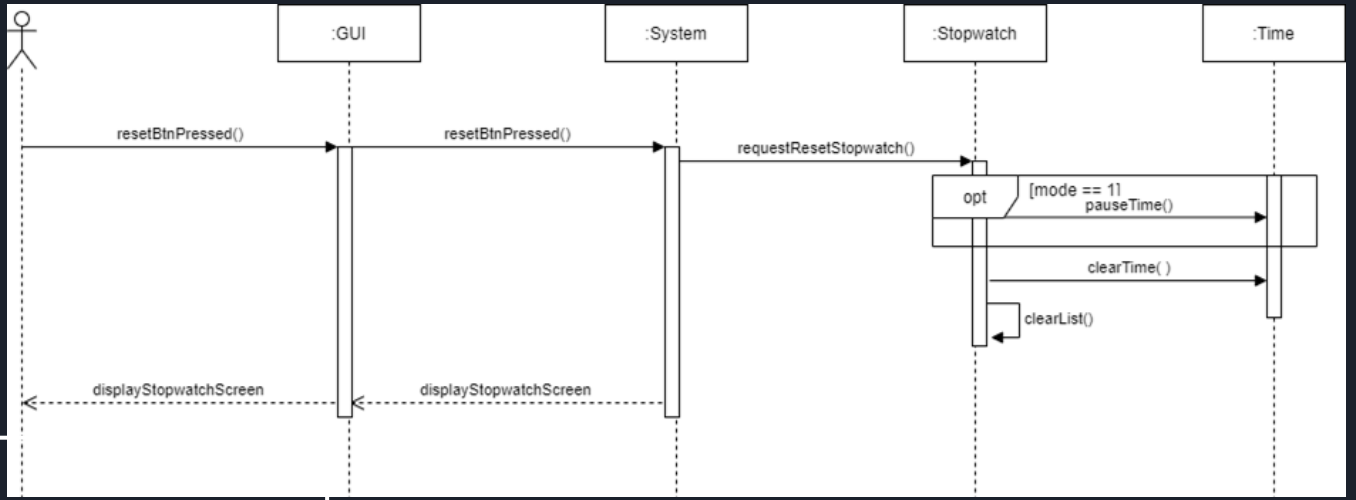


Type	GUI
Responsibilites	버튼을 눌러 스톱워치를 일시정지한다.
Cross Reference	Function : R3.2
Note	스톱워치의 시간이 멈춘다.
Pre-Conditions	스톱워치 화면이어야 한다. 스톱워치가 진행중이어야 한다.
Post-Conditions	버튼을 누른 시점에서 스톱워치의 시간이 멈춘 것이 화면에 보인다.

2052. Implements Windows

- Stopwatch

12. Reset Stopwatch

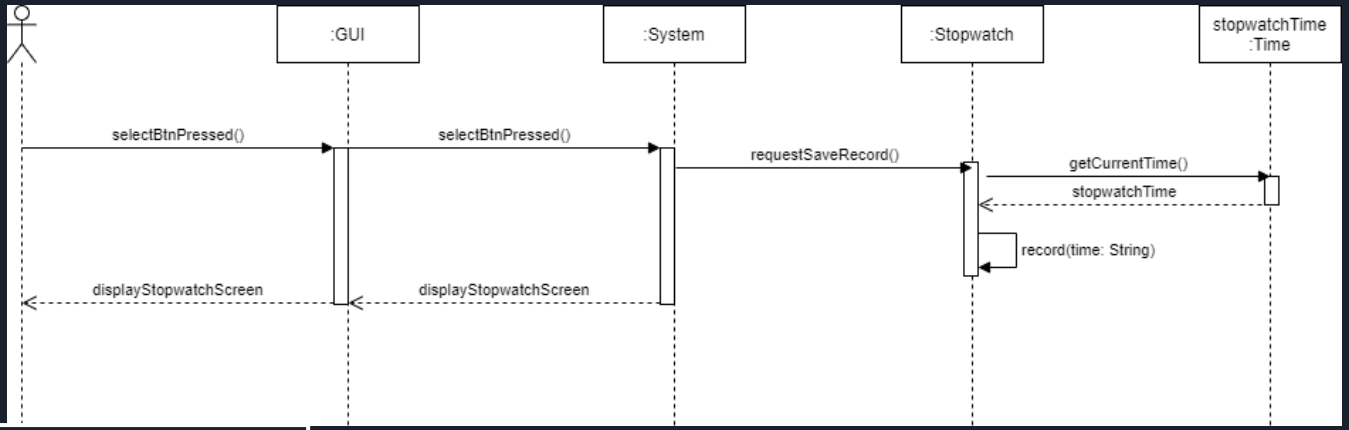


Type	GUI
Responsibilites	버튼을 눌러 스톱워치를 초기화한다.
Cross Reference	Function : R3.3
Note	스톱워치가 0시 0분 0초로 초기화된다.
Pre-Conditions	스톱워치 화면이어야 한다.
Post-Conditions	스톱워치가 초기화된 것이 화면에 나타난다.

2052. Implements Windows

- Stopwatch

13. Record Stopwatch



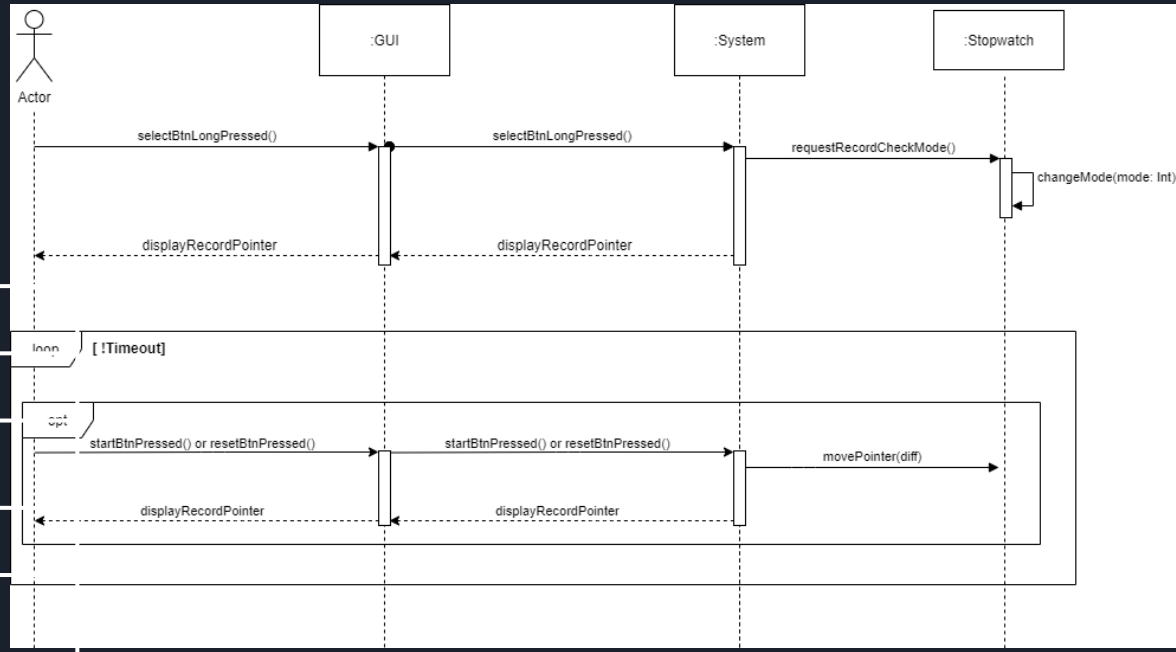
Type	GUI
Responsibilites	버튼을 눌러 스톱워치의 현재 기록을 저장한다.
Cross Reference	Function : R3.4
Note	스톱워치 기록 리스트에 현재 기록이 저장된다.
Pre-Conditions	스톱워치 화면이어야한다.
Post-Conditions	저장한 스톱워치 기록이 화면에 나타난다.

2052. Implements Windows

- Stopwatch

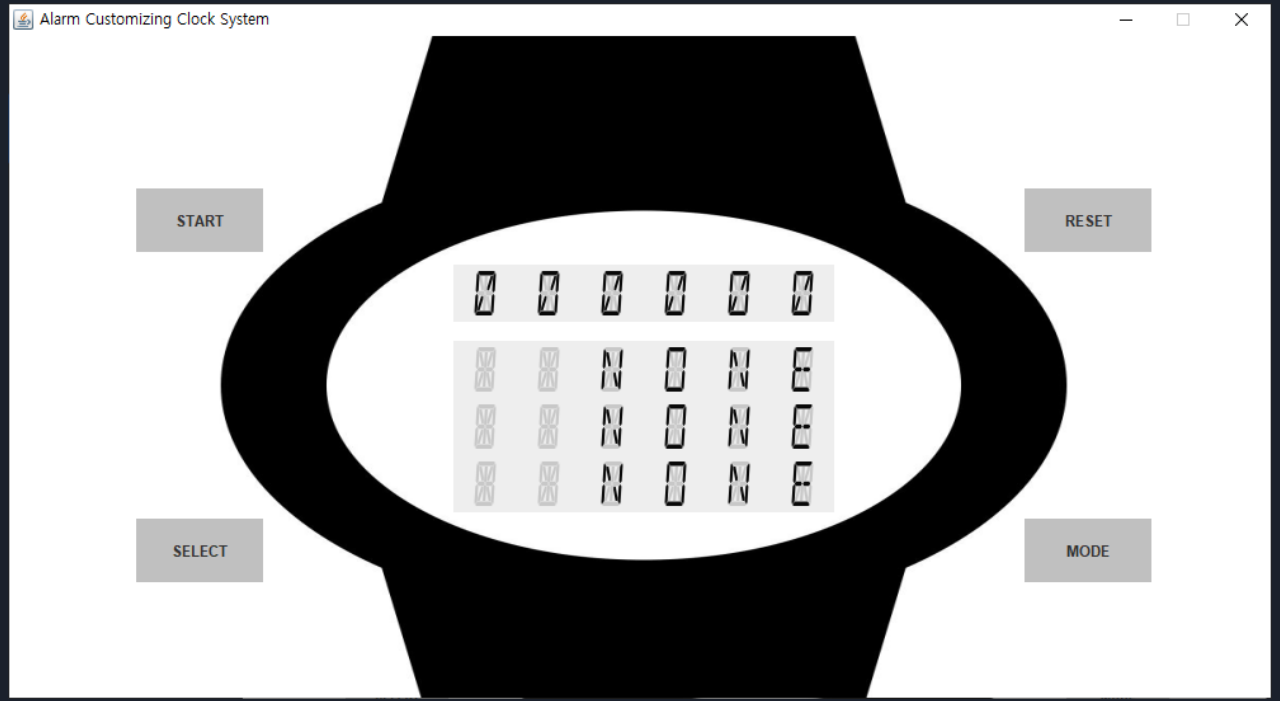
15. Control Stopwatch Record

Type	GUI
Responsibilites	버튼을 눌러 스톱워치 기록을 확인한다.
Cross Reference	Function : R3.6
Note	스톱워치 기록을 보여준다.
Pre-Conditions	스톱워치 화면이어야 한다. 스톱워치 기록이 있어야 한다.
Post-Conditions	포인터에 해당하는 기록을 화면에 보여준다.



2052. Implements Windows

- Alarm

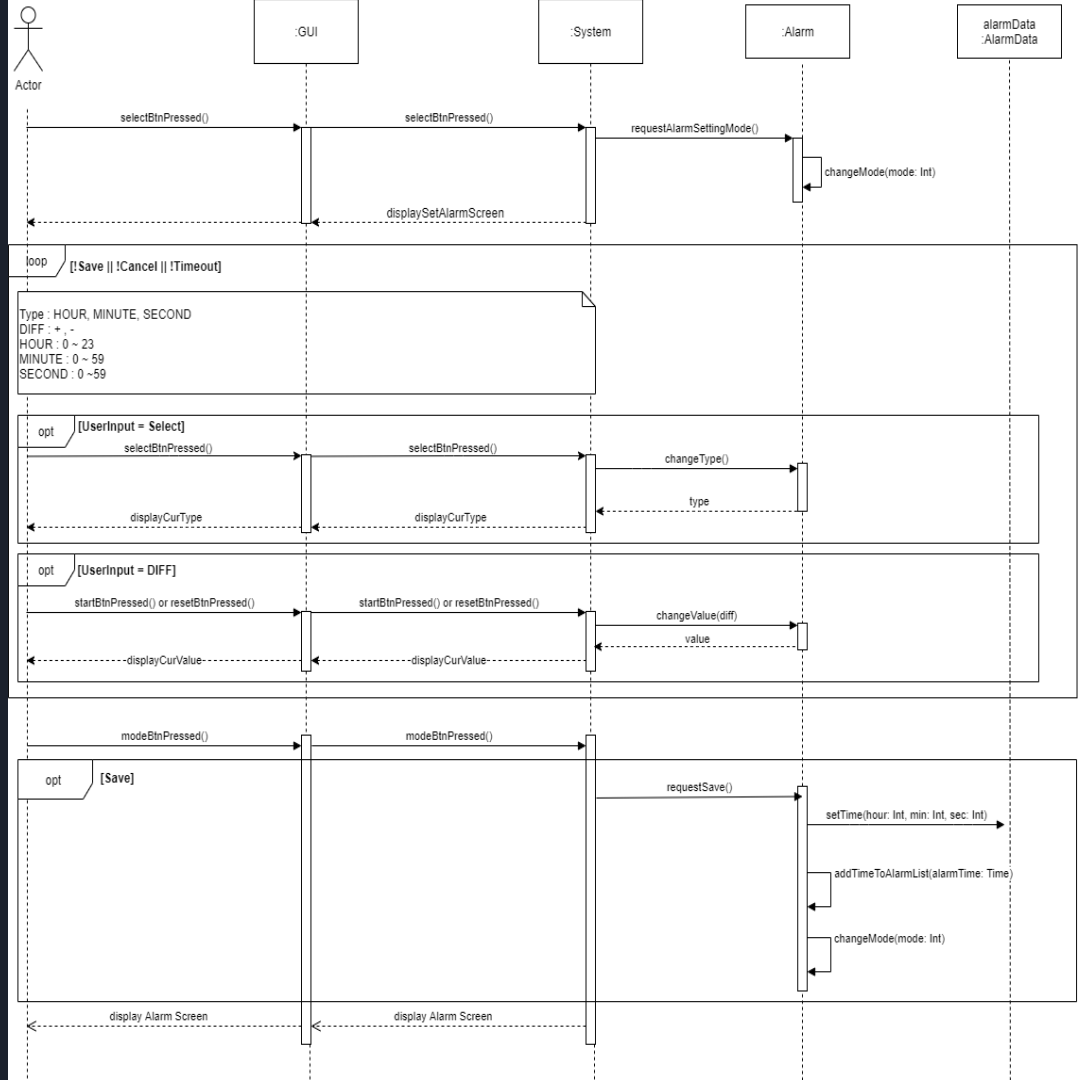


2052. Implements Windows

- Alarm

16. Set Alarm

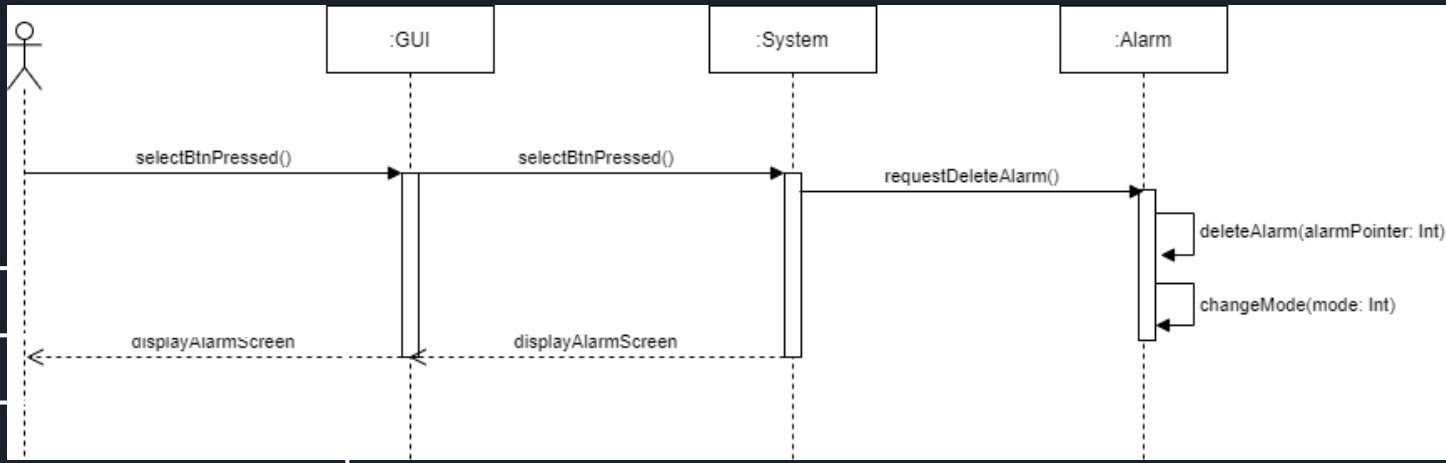
Type	GUI
Responsibilites	알람을 설정한다.
Cross Reference	Function : R4.1
Note	입력한 대로 새로운 알람이 설정된다.
Pre-Conditions	알람 화면이어야한다.
Post-Conditions	설정된 알람을 알람리스트에 반영하여 표시한다.



2052. Implements Windows

- Alarm

17. Delete Alarm

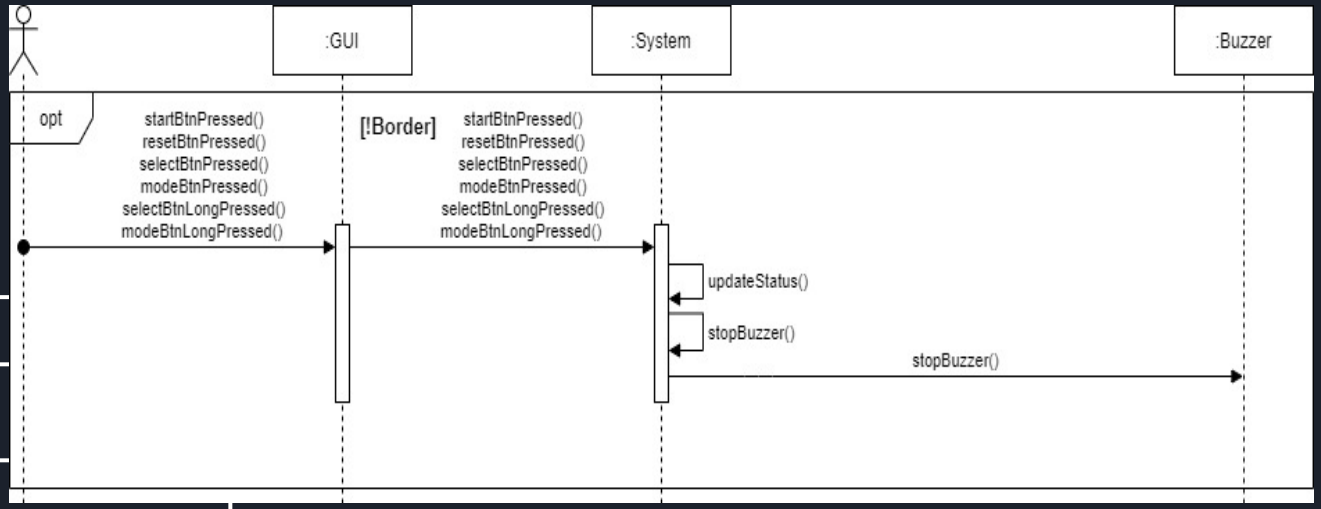


Type	GUI
Responsibilites	알람을 삭제한다.
Cross Reference	Function : R4.2
Note	포인터가 가리키는 알람을 삭제한다.
Pre-Conditions	알람 화면이면서 알람 선택 모드여야 한다.
Post-Conditions	삭제된 알람을 반영하여 알람리스트를 표시한다.

2052. Implements Windows

- Alarm

19. Stop Alarm Buzzer



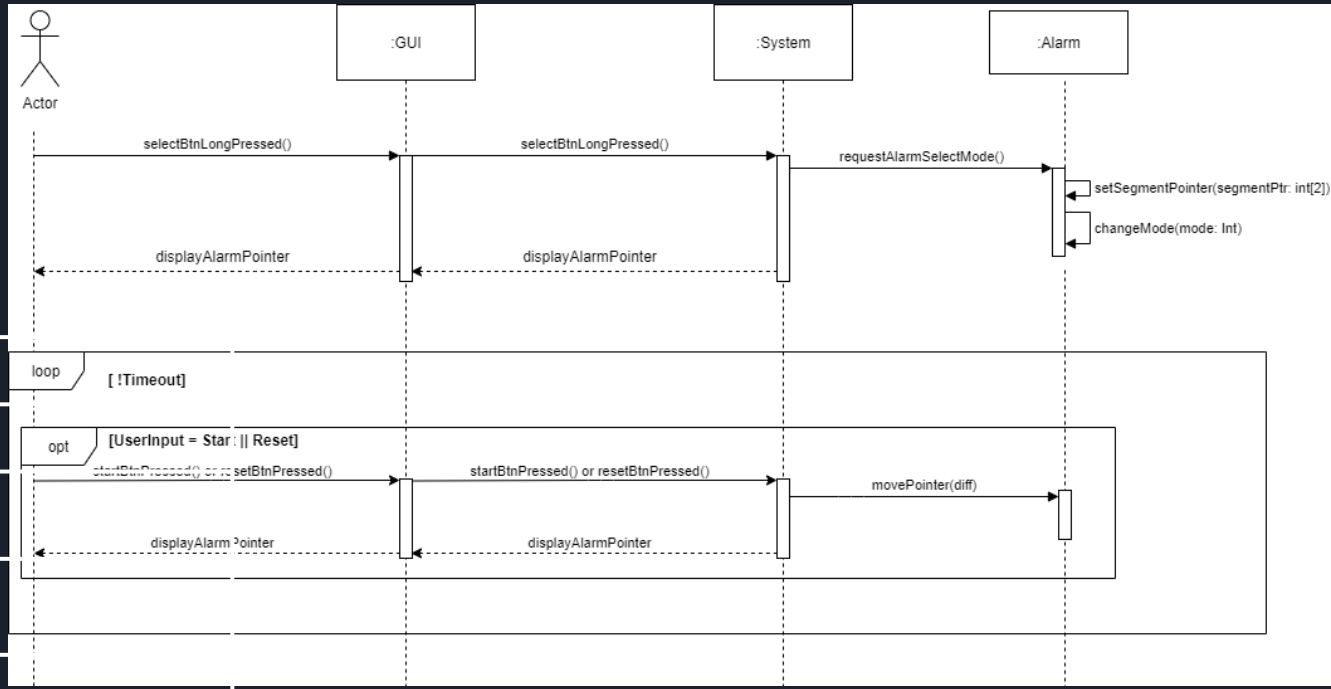
Type	GUI
Responsibilites	알람으로 인해 울리는 버저를 멈춘다.
Cross Reference	Function : R4.3
Note	아무 버튼이나 눌러 울리는 버저를 멈춘다.
Pre-Conditions	버저가 울리는 상태여야한다.
Post-Conditions	울리던 알람이 멈춘다.

2052. Implements Windows

- Alarm

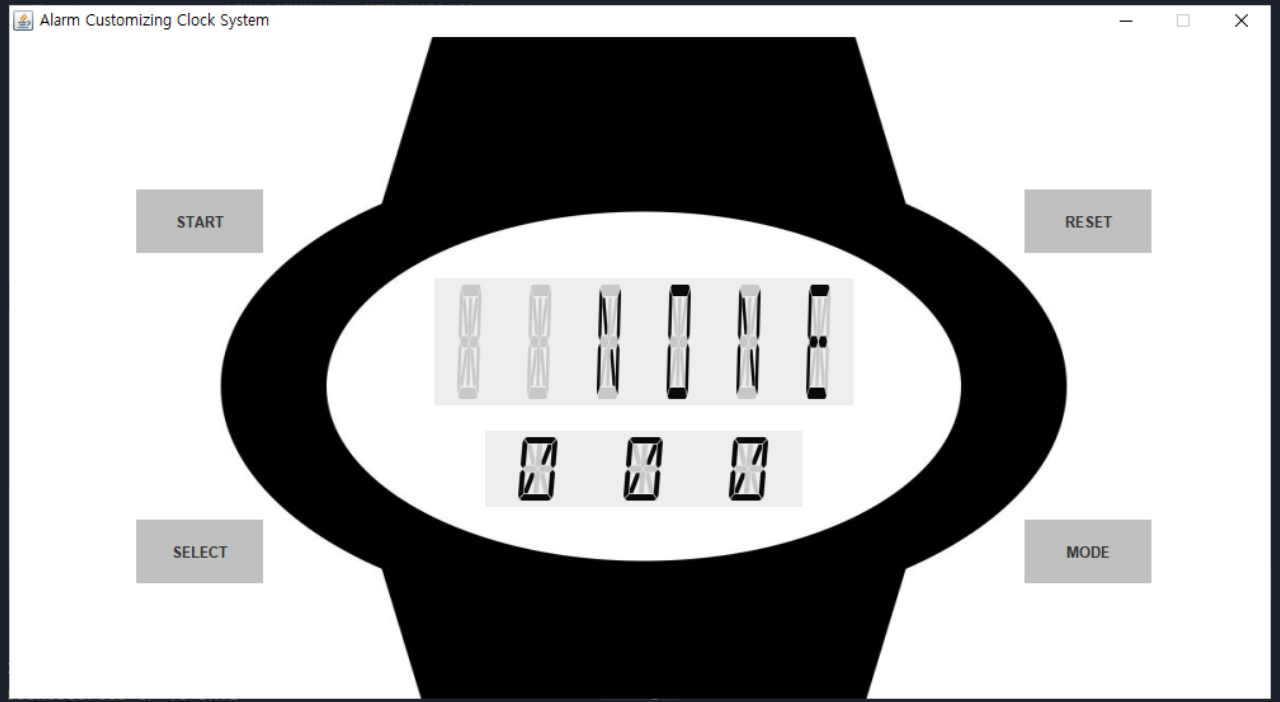
21. Control Alarm List

Type	GUI
Responsibilites	알람 포인터를 조작한다.
Cross Reference	Function : R4.4
Note	Start, Reset 버튼으로 포인터값을 조작한다.
Pre-Conditions	알람 화면이면서 알람 리스트에 알람이 존재해야 한다.
Post-Conditions	조작할 때마다 알람 리스트에서 선택하고 있는 알람을 표시한다. 또한 그 위치에 따라 표시되는 알람리스트도 바뀌어진다.



2052. Implements Windows

- D-day

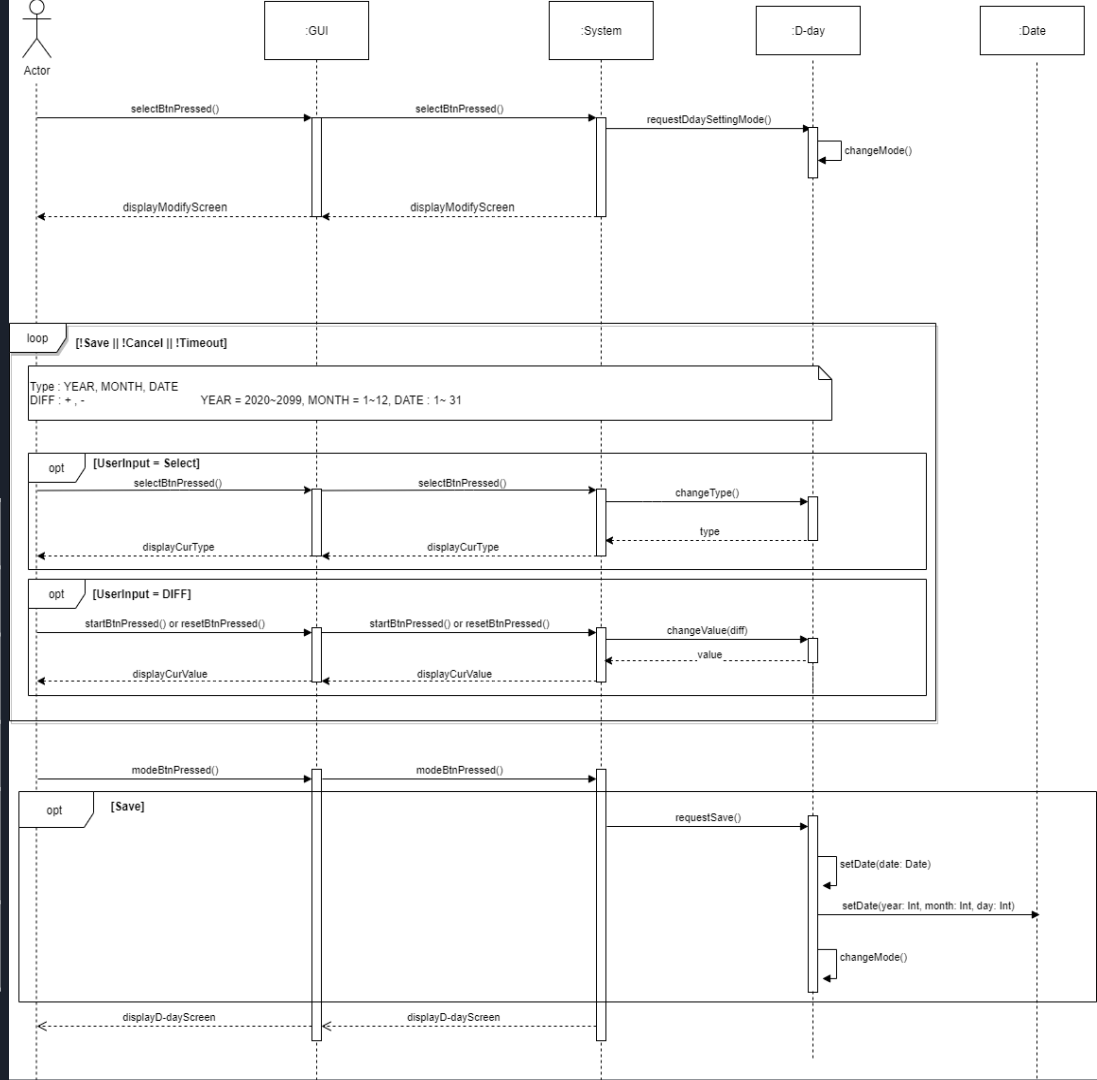


2052. Implements Windows

- D-day

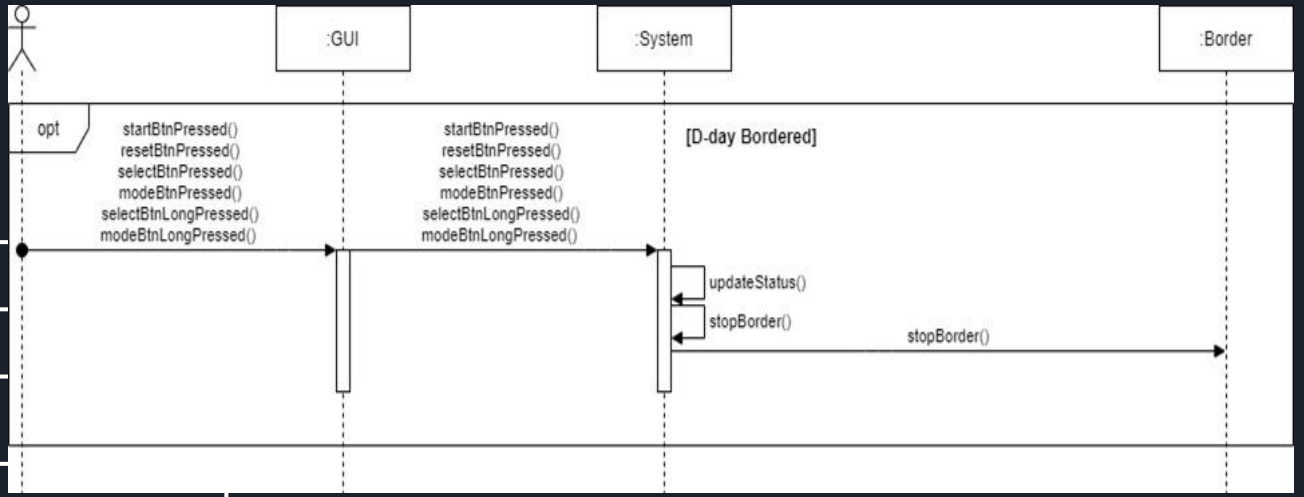
22. Set D-day

Type	GUI
Responsibilites	D-day를 설정한다.
Cross Reference	Function : R5.1
Note	설정한 대로 d-day가 저장된다.
Pre-Conditions	D-day 화면이어야 한다.
Post-Conditions	설정 한 D-day의 날짜와 남은 일수가 표시된다.



2052. Implements Windows

- D-day
24. Stop D-day Border

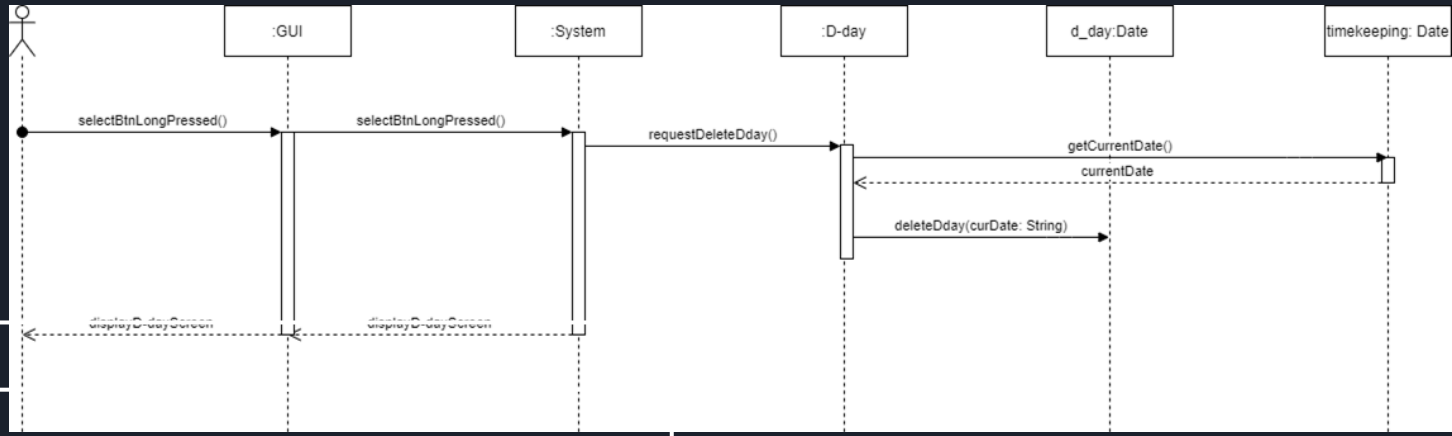


Type	GUI
Responsibilites	버튼을 눌러 깜빡임을 멈춘다.
Cross Reference	Function : R5.3
Note	blink를 멈춘다.
Pre-Conditions	시계 가장자리가 깜빡이는 상태여야 한다.
Post-Conditions	화면 가장자리의 깜빡임이 멈춘다.

2052. Implements Windows

- D-day

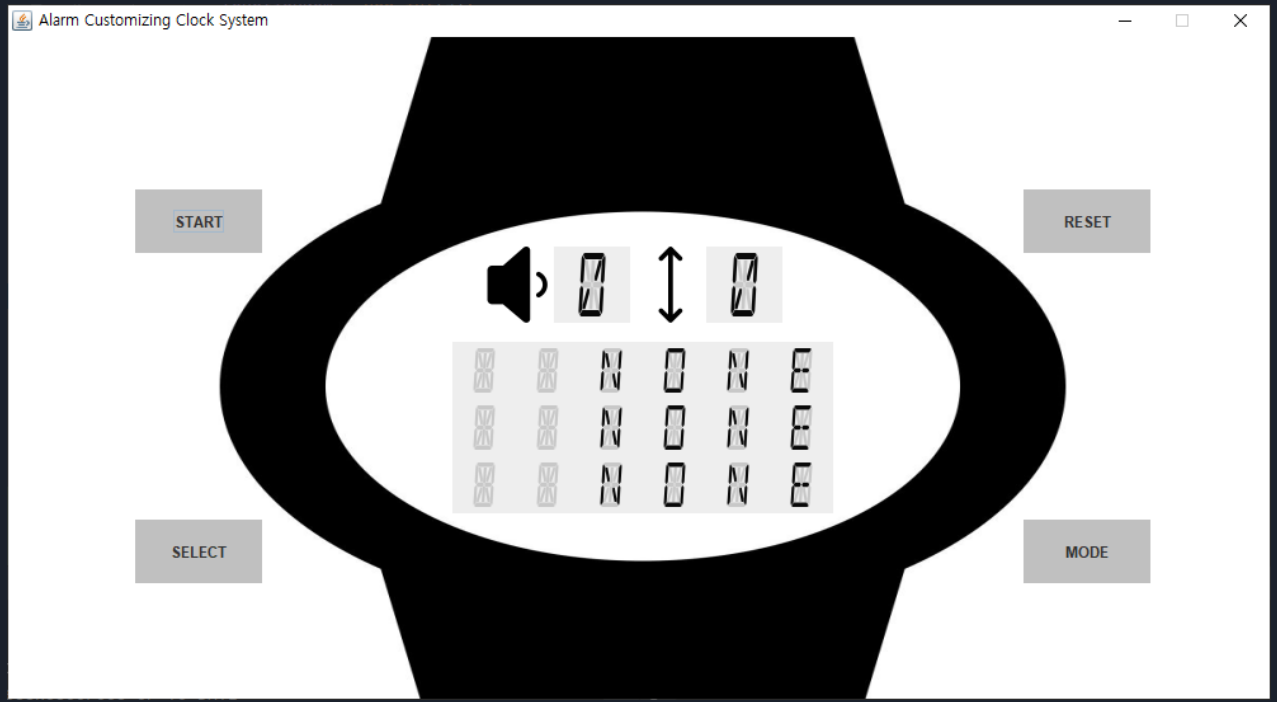
25. Delete D-day



Type	GUI
Responsibilites	설정한 D-day를 삭제 한다.
Cross Reference	Function : R5.4
Note	Select버튼을 길게 눌러 설정한 D-day값을 삭제한다.
Pre-Conditions	D-day화면이어야 한다.
Post-Conditions	D-day가 삭제되고, 화면에는 ‘NONE’이 출력된다.

2052. Implements Windows

- AlarmCustom

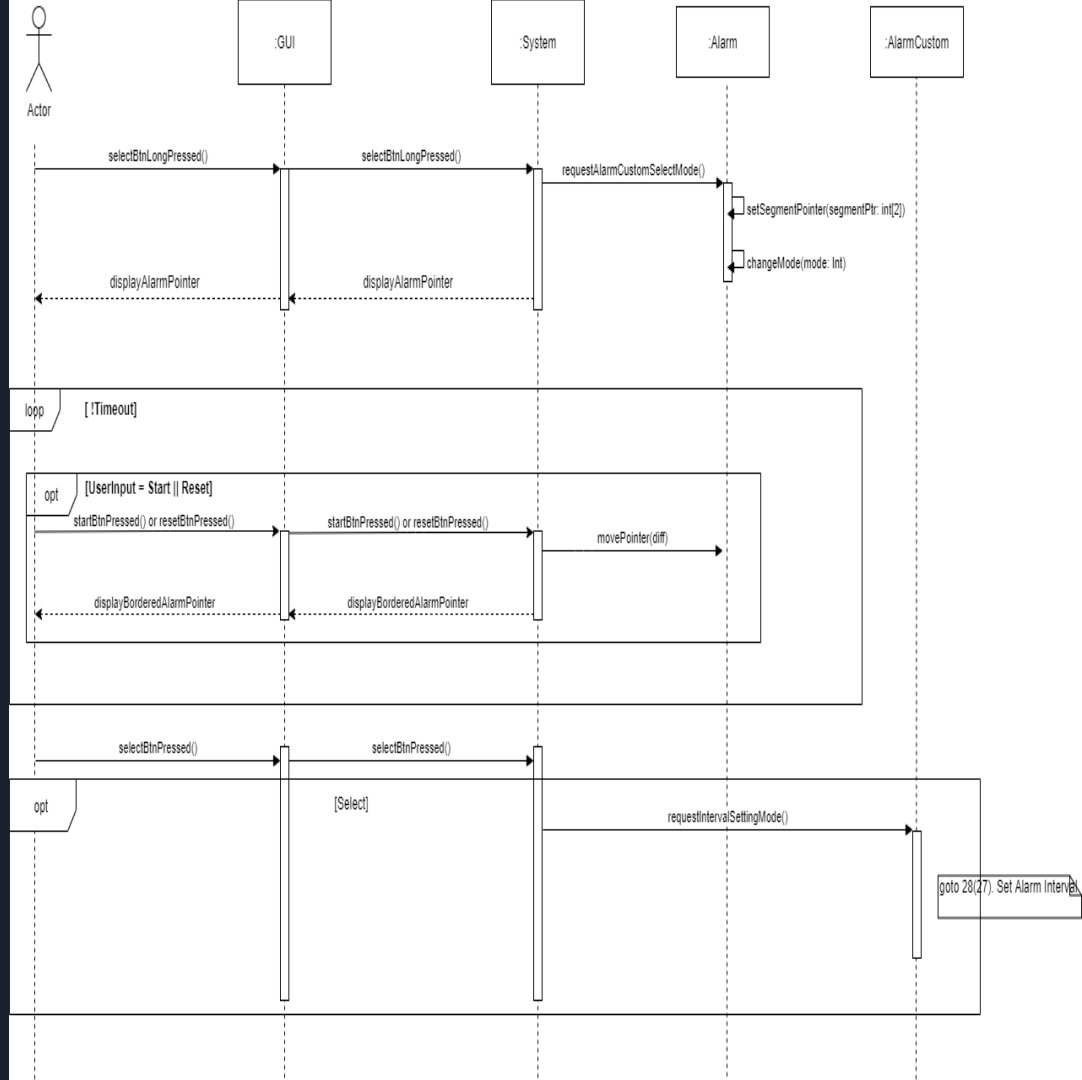


2052. Implements Windows

- AlarmCustom

26. Control
Alarm Custom List

Type	GUI
Responsibilites	알람리스트의 포인터를 조작한다.
Cross Reference	Function : R6.2, R6.3
Note	알람 선택
Pre-Conditions	알람커스텀 화면이면서 알람 조작 모드이어야 한다.
Post-Conditions	선택하고 있는 알람이 표시된다.

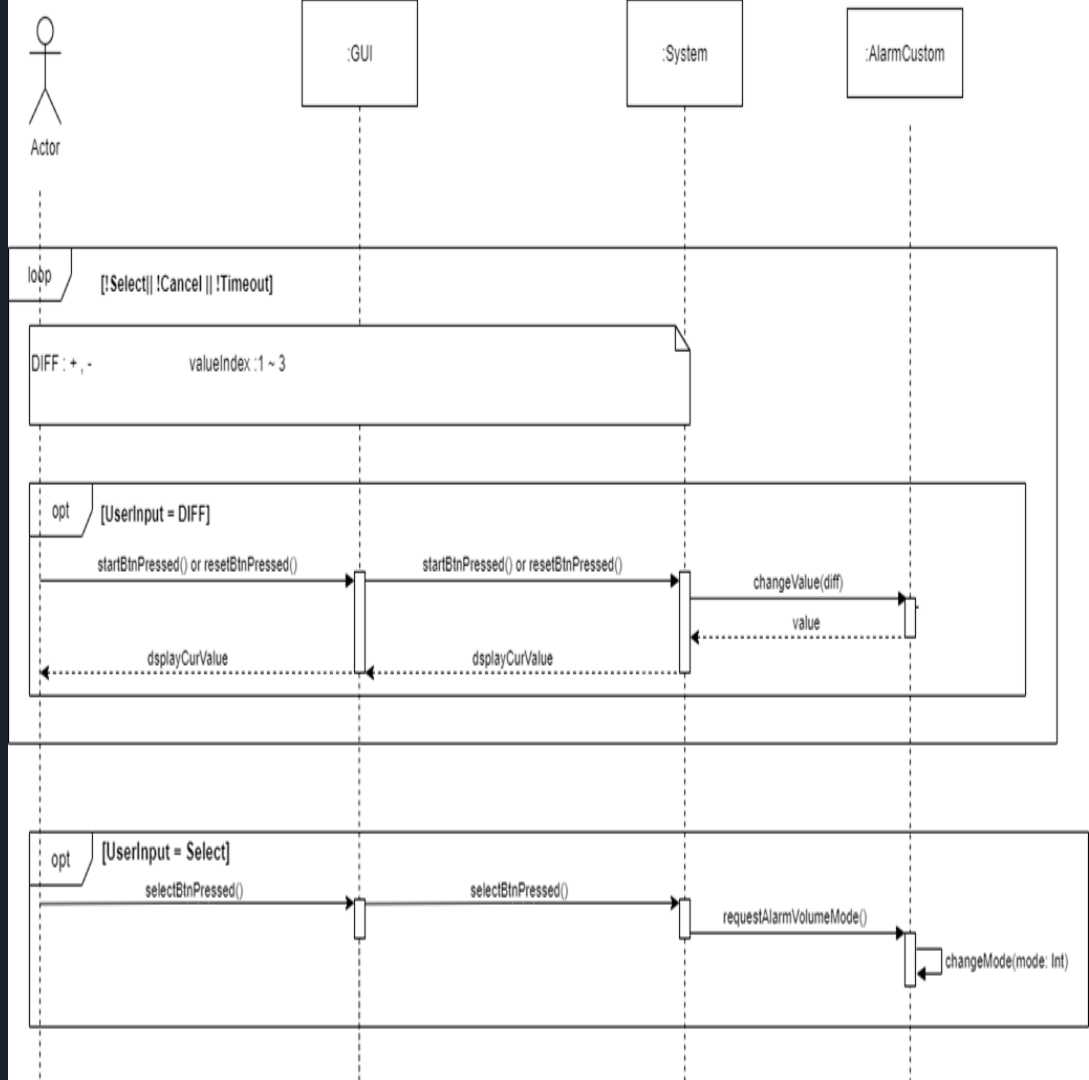


2052. Implements Windows

- AlarmCustom

27. Set Alarm Interval

Type	GUI
Responsibilites	현재 시각을 설정한다.
Cross Reference	Function : R6.2, R6.3
Note	설정한 대로 알람 간격이 설정된다.
Pre-Conditions	알람 선택 모드에서 알람을 선택해야한다.
Post-Conditions	설정값이 화면에 표시된다. 알람 볼륨 설정 모드로 넘어간다.

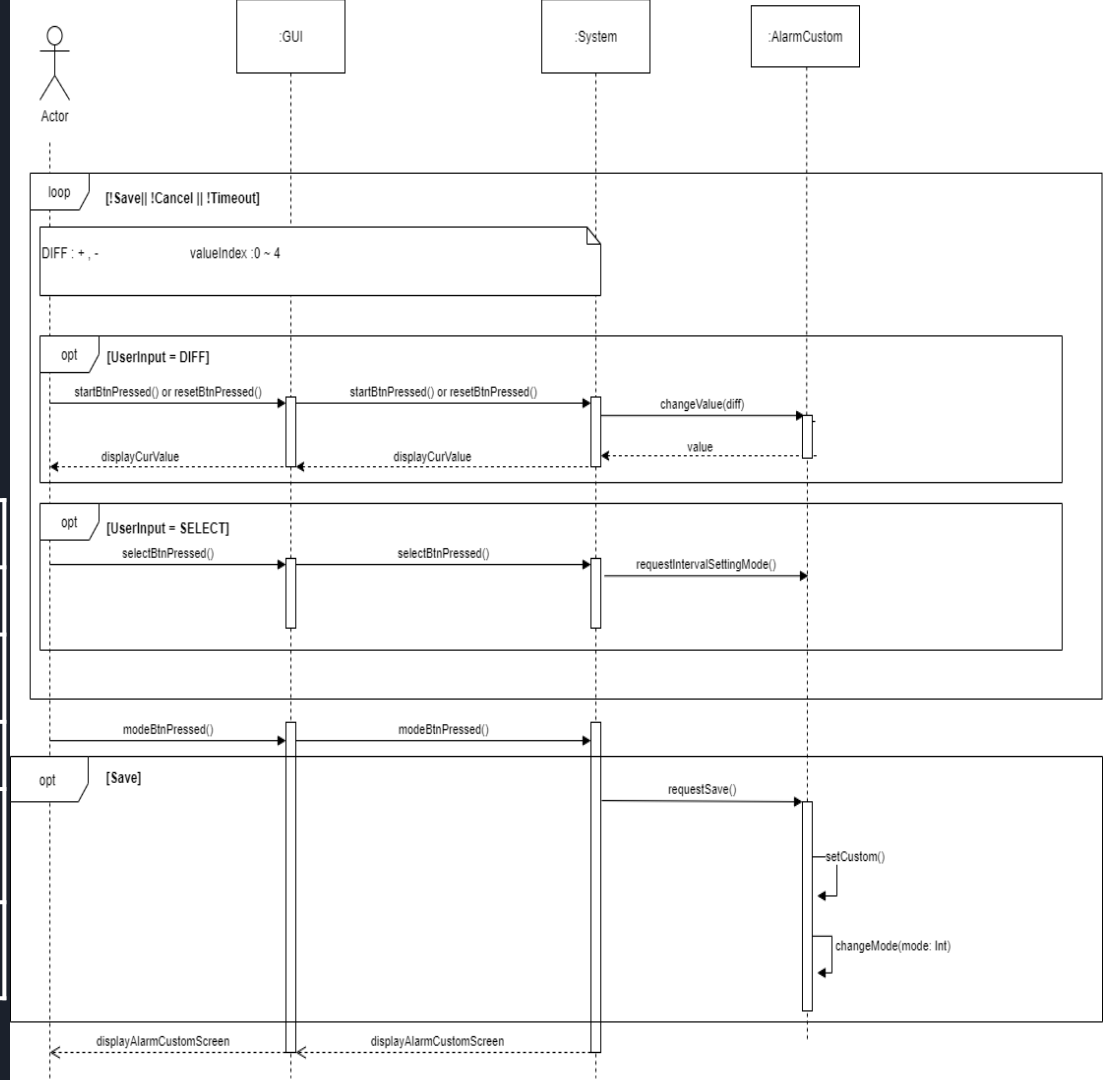


2052. Implements Windows

- AlarmCustom

28. Set Alarm Volume

Type	GUI
Responsibilites	알람 볼륨을 설정한다.
Cross Reference	Function : R6.4
Note	설정한 값으로 알람 볼륨이 설정된다.
Pre-Conditions	알람 커스텀 화면이면서, 알람볼륨 설정 모드여야 한다.
Post-Conditions	커스텀한 알람이 저장되고, 기본 화면으로 돌아간다.

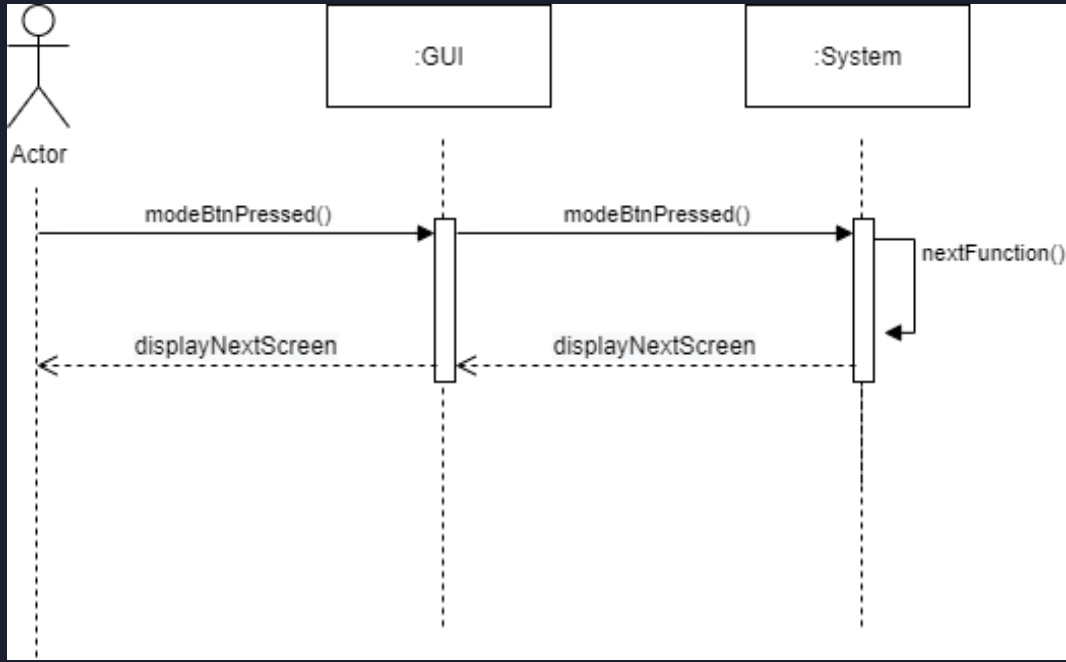


2052. Implements Windows

- System

29. Change Screen

Type	GUI
Responsivilites	화면을 전환한다.
Cross Reference	Function : R7.1
Note	Mode 버튼으로 다음화면으로 넘긴다.
Pre-Conditions	각 화면의 기본모드여야한다.
Post-Conditions	3. 에서 설정된 화면 순서대로 화면이 전환된다.

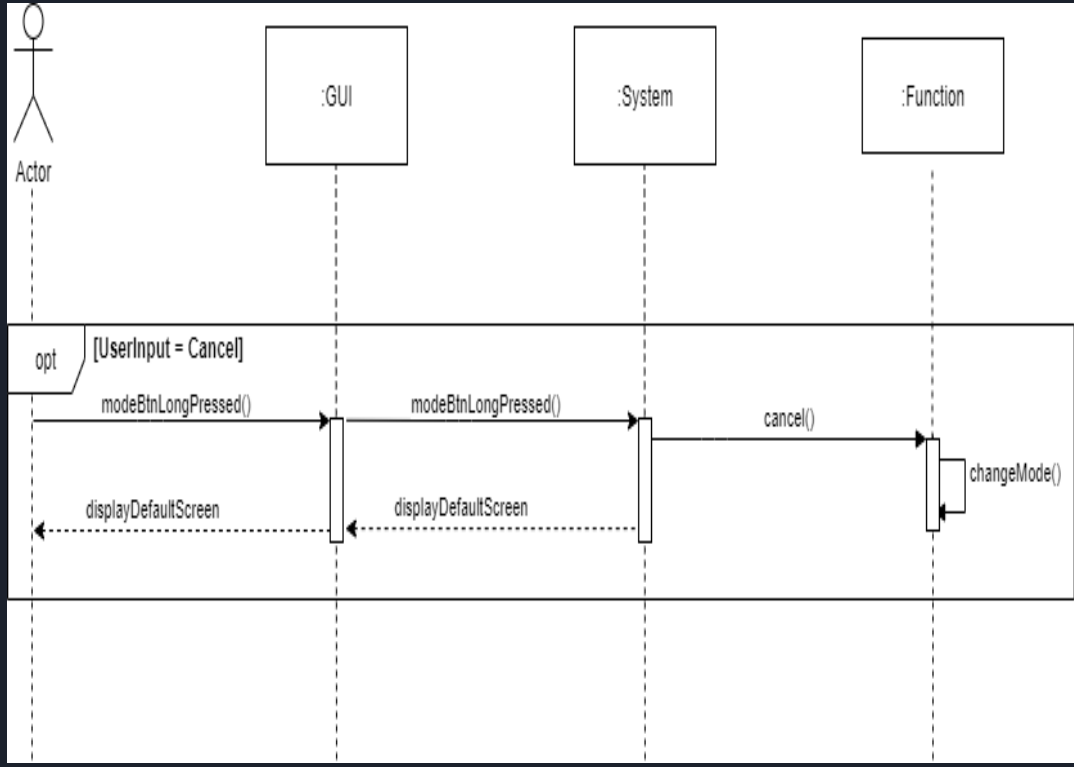


2052. Implements Windows

- System

31. Cancel

Type	GUI
Responsivilites	진행중이던 기능을 취소한다.
Cross Reference	Function : R7.3
Note	Mode 버튼을 길게 눌러 기본 모드로 돌아간다.
Pre-Conditions	N/A
Post-Conditions	진행중이던 기능을 취소하고, 해당 화면의 기본모드로 돌아가 화면을 표시한다.



2055. Write Unit Test Code

- Timekeeping

```
@Test
public void setTimeTest() {
    TimeKeeping timekeeping = system.timeKeeping;
    timekeeping.requestTimeSettingMode();
    String time = timekeeping.getCurTime().getCurrentTime();
    String date = timekeeping.getCurDate().getCurrentDate();
    int dayOfTheWeek;
    for (int i = 0; i < 5; i++) {
        timekeeping.changeValue( diff: 2);
        timekeeping.changeType();
    }
    timekeeping.changeValue( diff: 2);
    timekeeping.requestSave();
```

```
String splitedTime[] = time.split( regex: " ");
int splitedTimeInt[] = new int[3];
for (int i = 0; i < 3; i++)
    splitedTimeInt[i] = Integer.parseInt(splitedTime[i]) + 2;
String splitedDate[] = date.split( regex: " ");
int splitedDateInt[] = new int[3];
for (int i = 0; i < 3; i++)
    splitedDateInt[i] = Integer.parseInt(splitedDate[i]) + 2;

time = splitedTimeInt[0] + " " + splitedTimeInt[1] + " " + splitedTimeInt[2];
date = splitedDateInt[0] + " " + splitedDateInt[1] + " " + splitedDateInt[2];

Calendar calendar = Calendar.getInstance();
DateFormat dateFormat = new SimpleDateFormat( pattern: "yyyy MM dd");
try {
    java.util.Date date1 = dateFormat.parse(date);
    calendar.setTime(date1);
} catch (ParseException e) {
    e.printStackTrace();
```

```
dayOfTheWeek = calendar.get(Calendar.DAY_OF_WEEK);

assert (timekeeping.getCurTime().getCurrentTime().equals(time));
assert (timekeeping.getCurDate().getCurrentDate().equals(date));
assert (timekeeping.getDayOfTheWeek() == dayOfTheWeek);
```

2055. Write Unit Test Code

- Timekeeping

```
@Test
public void setDisplayTest() {
    system.nextFunction();
    assert (system.getSelectedFid() == 2);
}
```

2055. Write Unit Test Code

- Stopwatch

```java
@Test
public void startStopwatchTest() {
    Stopwatch stopwatch = system.stopwatch;

    stopwatch.requestStartStopwatch();

    try {
        Thread.sleep( millis: 3100);
    } catch(InterruptedException e) {
        e.printStackTrace();
    }

    Time time = stopwatch.getStopwatch();

    String timeStr = time.getCurrentTime();
    String splitedTime[] = timeStr.split( regex: " ");

    assert(splitedTime[2].equals("3"));
}
```

2055. Write Unit Test Code

- Stopwatch

```
@Test
public void pauseStopwatchTest() {
    Stopwatch stopwatch = system.stopwatch;
    stopwatch.requestStartStopwatch();
    try {
        Thread.sleep( millis: 3100);
    } catch(InterruptedException e) {
        java.lang.System.out.println(e.getMessage());
    }
    stopwatch.requestPauseStopwatch();
    try {
        Thread.sleep( millis: 2100);
    } catch(InterruptedException e) {
        java.lang.System.out.println(e.getMessage());
    }
    Time time = stopwatch.getStopwatch();
    String timeStr = time.getCurrentTime();
    String splitedTime[] = timeStr.split( regex: " ");
    assert(splitedTime[2].equals("3"));
}
```

2055. Write Unit Test Code

- Stopwatch

```java
@Test
public void resetStopwatchTest() {
    Stopwatch stopwatch = system.stopwatch;
    stopwatch.requestStartStopwatch();
    try {
        Thread.sleep( millis: 3100);
    } catch(InterruptedException e) {
        java.lang.System.out.println(e.getMessage());
    }
    Time time = stopwatch.getStopwatch();
    String timeStr = time.getCurrentTime();
    String splitedTime[] = timeStr.split( regex: " ");
    assert(splitedTime[2].equals("3"));
    stopwatch.requestResetStopwatch(); //reset
    timeStr = time.getCurrentTime();
    splitedTime = timeStr.split( regex: " ");
    assert(splitedTime[2].equals("0"));
}
```

2055. Write Unit Test Code

- Stopwatch

```java
@Test
public void recordStopwatchTest() {
    Stopwatch stopwatch = system.stopwatch;
    stopwatch.requestStartStopwatch();
    for(int i=0; i<5; i++) {
        try {
            Thread.sleep( millis: 1100);
        } catch(InterruptedException e) {
            java.lang.System.out.println(e.getMessage());
        }
        stopwatch.requestSaveRecord();
    }
    String[] rec = stopwatch.getStopwatchRecord();
    for(int i=0; i<5; i++)
        assert(rec[i].equals("0 0 " + (i+1)));
}
```

2055. Write Unit Test Code

- Stopwatch

```
@Test
public void controlStopwatchRecord() {
    Stopwatch stopwatch = system.stopwatch;

    stopwatch.movePointer( diff: 1);
    assert(stopwatch.getRecordPointer()==1);
}
```

2055. Write Unit Test Code

- Stopwatch

```
@Test
public void startStopwatchTest() {
    Stopwatch stopwatch = system.stopwatch;

    stopwatch.requestStartStopwatch();

    try {
        Thread.sleep( millis: 3100);
    } catch(InterruptedException e) {
        e.printStackTrace();
    }

    Time time = stopwatch.getStopwatch();

    String timeStr = time.getCurrentTime();
    String splitedTime[] = timeStr.split( regex: " ");

    assert(splitedTime[2].equals("3"));
}
```

2055. Write Unit Test Code

- Timer

```java
@Test
public void setTimerTest() {
    Timer timer = system.timer;

    timer.requestTimerSettingMode();

    for(int i = 0; i<2; i++) {
        timer.changeValue( diff: i+1);
        timer.changeType();
    }
    timer.changeValue( diff: 3);
    timer.requestSave();

    Time time = timer.getTimer();

    String timeStr = time.getCurrentTime();
    String splitedTime[] = timeStr.split( regex: " ");

    assert(splitedTime[0].equals("1"));
    assert(splitedTime[1].equals("2"));
    assert(splitedTime[2].equals("3"));
}
```

```java
@Test
public void startTimerTest() {
    Timer timer = system.timer;
    timer.requestTimerSettingMode();

    timer.changeType();
    timer.changeType();
    timer.changeValue( diff: 5); // Timer를 5초로 Setting
    timer.requestSave();

    timer.requestStartTimer();

    try {
        Thread.sleep( millis: 3100); // 3초가 흐른게 되네요
    } catch(InterruptedException e) {
        java.lang.System.out.println(e.getMessage());
    }

    Time time = timer.getTimer();

    String timeStr = time.getCurrentTime();
    String splitedTime[] = timeStr.split( regex: " ");

    assert(splitedTime[2].equals("2"));

}
```

```java
@Test
public void beepTimerTest() {
    Timer timer = system.timer;
    timer.requestTimerSettingMode();

    timer.changeType();
    timer.changeType();
    timer.changeValue( diff: 3); // Timer를 3초로 Setting
    timer.requestSave();

    timer.requestStartTimer();

    try {
        Thread.sleep( millis: 3100); // 3초가 흐른게 되네요
    } catch(InterruptedException e) {
        java.lang.System.out.println(e.getMessage());
    }

    assertTrue(timer.system.buzzer.isBuzzerState());
}
```

2055. Write Unit Test Code

- Timer

```
@Test
public void resetTimerTest() {
    Timer timer = system.timer;

    timer.requestTimerSettingMode();

    timer.changeType();
    timer.changeType();
    timer.changeValue( diff: 5); // Timer를 5초로 Setting
    timer.requestSave();

    timer.requestStartTimer();

    Time time = timer.getTimer();

    String timeStr = time.getCurrentTime();
    String splitedTime[] = timeStr.split( regex: " ");

    assert(splitedTime[2].equals("5"));

    timer.requestResetTimer();

    time = timer.getTimer();

    timeStr = time.getCurrentTime();
    splitedTime = timeStr.split( regex: " ");

    assert(splitedTime[0].equals("0"));
    assert(splitedTime[1].equals("0"));
    assert(splitedTime[2].equals("0"));
}
```

```
@Test
public void pauseTimerTest() {
    Timer timer = system.timer;

    timer.requestTimerSettingMode();

    timer.changeType();
    timer.changeType();
    timer.changeValue( diff: 5); // Timer를 5초로 Setting
    timer.requestSave();

    timer.requestStartTimer();

    try {
        Thread.sleep( millis: 1100); // 1초가 흐른게 되네요
    } catch(InterruptedException e) {
        java.lang.System.out.println(e.getMessage());
    }

    timer.requestPauseTimer();

    try {
        Thread.sleep( millis: 1100); // 1초가 흐른게 되네요
    } catch(InterruptedException e) {
        java.lang.System.out.println(e.getMessage());
    }

    Time time = timer.getTimer();

    String timeStr = time.getCurrentTime();
    String splitedTime[] = timeStr.split( regex: " ");

    assert(splitedTime[2].equals("4"));
}
```

2055. Write Unit Test Code

- Timer

```java
@Test
public void stopTimerBuzzer() {
    Timer timer = system.timer;

    timer.system.buzzer.beepBuzzer( interval: 1,  volume: 1);
    timer.system.buzzer.stopBuzzer();

    assertFalse(timer.system.buzzer.isBuzzerState());
}
```

2055. Write Unit Test Code

- D-day

```
@Test
public void setDdayTest() {
    D_day d_day = system.d_day;

    d_day.requestDdaySettingMode();

    String curDate = d_day.getD_dayDate().getCurrentDate();
    String splited[] = curDate.split( regex: " ");

    StringTokenizer st = new StringTokenizer(curDate, delim: " ");
    Calendar curDateCal = Calendar.getInstance();
    curDateCal.set(Integer.parseInt(st.nextToken()), month: Integer.parseInt(st.nextToken()) - 1, Integer.parseInt(st.nextToken()));

    //1년 뒤, 2개월 뒤, 3일 뒤로 저장
    d_day.changeValue( diff: 1);
    d_day.changeType();
    d_day.changeValue( diff: 2);
    d_day.changeType();
    d_day.changeValue( diff: 3);
    d_day.requestSave();
```

```
    Date date = d_day.getD_dayDate();
    int dday = d_day.getD_day();

    Calendar d_dayDateCal = Calendar.getInstance();
    d_dayDateCal.set(date.getYear(), month: date.getMonth()-1, date.getDay());

    assert(date.getYear() == (Integer.parseInt(splited[0]) + 1));
    assert(date.getMonth() == (Integer.parseInt(splited[1]) + 2));
    assert(date.getDay() == (Integer.parseInt(splited[2]) + 3));
    assert(dday == (d_dayDateCal.getTimeInMillis() - curDateCal.getTimeInMillis()) / (60*60*24*1000));
}
```

2055. Write Unit Test Code

- D-day

```java
@Test
public void borderDdayTest() {
    D_day d_day = system.d_day;

    Date date = new Date();
    date.setDate( y: 2020, m: 6, d: 9);
    d_day.setDate(date);
    java.lang.System.out.println(d_day.getD_day());

    system.timeKeeping.getCurTime().setTime( hour: 23, min: 59, sec: 59);

    try {
        Thread.sleep( millis: 2000); // 1초가 흐른게 되네요
    } catch(InterruptedException e) {
        java.lang.System.out.println(e.getMessage());
    }

    assertTrue(system.border.getBorderState());
}
```

```java
@Test
public void stopDdayBorderTest() {
    D_day d_day = system.d_day;

    system.startBorder();
    assertTrue(system.border.getBorderState());

    system.stopBorder();
    assertFalse(system.border.getBorderState());
}
```

2055. Write Unit Test Code

- D-day

```
@Test
public void deleteDdayTest() {
    D_day d_day = system.d_day;

    String curDate = d_day.getD_dayDate().getCurrentDate();
    String splited[] = curDate.split( regex: " ");

    d_day.requestDdaySettingMode();

    //1년 뒤, 2개월 뒤, 3일 뒤로 저장
    d_day.changeValue( diff: 1);
    d_day.changeType();
    d_day.changeValue( diff: 2);
    d_day.changeType();
    d_day.changeValue( diff: 3);
    d_day.requestSave();

    d_day.requestDeleteDday();

    java.lang.System.out.println(d_day.getD_dayDate().getCurrentDate());

    assert(d_day.getD_dayDate().getYear() == Integer.parseInt(splited[0]));
    assert(d_day.getD_dayDate().getMonth() == Integer.parseInt(splited[1]));
    assert(d_day.getD_dayDate().getDay() == Integer.parseInt(splited[2]));
    assert(d_day.getD_day() == -1);
}
```

2055. Write Unit Test Code

- Alarm

```java
@Test
public void SetAlarmTest() {
    Alarm alarm = system.alarm;

    alarm.requestAlarmSettingMode(); // mode = 1;

    // 순서대로 시, 분, 초 입력
    alarm.changeValue( diff: 2); // 시
    alarm.changeType();
    alarm.changeValue( diff: 2); // 분
    alarm.changeType();
    alarm.changeValue( diff: 2); // 초

    //request Save
    alarm.requestSave();

    // 테스트 비교용
    int alarmSettingValue[] = {-1,-1,-1};
    Time time = new Time( timeFlag: 2);

    // 순서대로 시, 분, 초 입력
    for (int i = 0 ; i < 3 ; i++)
        alarmSettingValue[i] = 2;


    time.setTime(alarmSettingValue[0], alarmSettingValue[1], alarmSettingValue[2]);

    // 잘 저장 되었는 지 확인.
    assert(alarm.getAlarmList()[0].getAlarmTime().getCurrentTime().equals(time.getCurrentTime()));
}
```

```java
@Test
public void DeleteAlarmTest() {
    Alarm alarm = new Alarm(system);

    alarm.movePointer( diff: 1);
    int i = alarm.getAlarmPointer();
    assertEquals( expected: 0, i);

    Time time = new Time( timeFlag: 2);
    time.setTime( hour: 1, min: 1, sec: 1);
    alarm.addTimeToAlarmList(time);
    Time time2 = new Time( timeFlag: 2);
    time2.setTime( hour: 2, min: 2, sec: 2);
    alarm.addTimeToAlarmList(time2);
    Time time3 = new Time( timeFlag: 2);
    time3.setTime( hour: 3, min: 3, sec: 3);
    alarm.addTimeToAlarmList(time3);

    alarm.requestDeleteAlarm(); // " 2 2 2" 삭제

    assert(alarm.getAlarmList()[1].getAlarmTime().equals(time3)); // "3 3 3"
    assert(alarm.getAlarmList()[0].getAlarmTime().equals(time2));
}
```

2055. Write Unit Test Code

- Alarm

```
@Test
public void ControlAlarmListTest() {

    Alarm alarm = system.alarm;

    // size == 0
    alarm.movePointer( diff: 1);
    assertEquals( expected: 0, alarm.getAlarmPointer()); // 바로 리턴되서 변동 없음.

    // 중복 테스트
    Time time = new Time( timeFlag: 2);
    time.setTime( hour: 1, min: 1, sec: 1);
    alarm.addTimeToAlarmList(time);
    Time time2 = new Time( timeFlag: 2);
    time2.setTime( hour: 1, min: 1, sec: 1);
    alarm.addTimeToAlarmList(time2);
    assertEquals( expected: 1, alarm.getSize()); // 바로 리턴되서 변동 없음.

    // size > 0
    time2.setTime( hour: 2, min: 2, sec: 2);
    alarm.addTimeToAlarmList(time2);
    Time time3 = new Time( timeFlag: 2);
    time3.setTime( hour: 3, min: 3, sec: 3);
    alarm.addTimeToAlarmList(time3);
    Time time4 = new Time( timeFlag: 2);
    time4.setTime( hour: 4, min: 4, sec: 4);
    alarm.addTimeToAlarmList(time4);

    assertEquals( expected: 0, alarm.getAlarmPointer());
    alarm.movePointer( diff: 1);
    assertEquals( expected: 1, alarm.getAlarmPointer());
    alarm.movePointer( diff: 1);
    assertEquals( expected: 2, alarm.getAlarmPointer());
    alarm.movePointer( diff: 1);
    assertEquals( expected: 3, alarm.getAlarmPointer());
    alarm.movePointer( diff: -1);
    assertEquals( expected: 2, alarm.getAlarmPointer());
    alarm.movePointer( diff: -1);
    assertEquals( expected: 1, alarm.getAlarmPointer());
    alarm.movePointer( diff: -1);
    assertEquals( expected: 0, alarm.getAlarmPointer());
}
```

2055. Write Unit Test Code

- Alarm

```
@Test
public void BeepAlarmTest() {
    Alarm alarm = system.alarm;
    TimeKeeping timeKeeping = system.timeKeeping;
    Buzzer buzzer = system.buzzer;

    Time time = new Time( timeFlag: 2);
    time.setTime( hour: 23, min: 59, sec: 59);

    alarm.addTimeToAlarmList(time);

    // 알람 커스텀된 간격, 볼륨 설정
    alarm.getAlarmList()[0].setInterval(2);
    alarm.getAlarmList()[0].setVolume(2);

    //현재 시간 임의 설정
    timeKeeping.requestTimeSettingMode();

    // 시간, 분, 초, 순으로 설정
    for(int i = 0 ; i < 5 ; i++) {
        timeKeeping.changeValue( diff: 60); // 알맞게 변경하세요
        timeKeeping.changeType();
    }
    timeKeeping.changeValue( diff: 60);
    timeKeeping.requestSave(); // "23 59 59"

    java.lang.System.out.println(timeKeeping.getCurTime().getCurrentTime());
    java.lang.System.out.println(alarm.getAlarmList()[0].getAlarmTime().getCurrentTime());

    // 현재 시각과 알람 시간이 같으면
    if(timeKeeping.getCurTime().getCurrentTime().equals(alarm.getAlarmList()[0].getAlarmTime().getCurrentTime()))
    {
        system.beepBuzzer(alarm.getAlarmList()[0].getInterval(), alarm.getAlarmList()[0].getVolume()); // 버저 울리기.
        assertEquals( expected: 1, actual: system.getStatus() & 1);
        // buzzer에서 interval과 volume을 get할 방법이 없음. -> getter로 신규 함수 추가해야함.
        assertEquals( expected: 1000, system.buzzer.getInterval());
        assertEquals( expected: 0.07, system.buzzer.getVolume());
    }
}
```

2055. Write Unit Test Code

- Alarm

```
@Test
public void StopAlarmBuzzerTest() {
    Alarm alarm = system.alarm;
    system.beepBuzzer( interval: 1,  volume: 1);
    system.updateStatus();

    assertEquals( expected: 0,  actual: system.getStatus() & 0);
}
```

2055. Write Unit Test Code

- AlarmCustom

```
public AlarmCustomTest() {
    int[] alarmFuction = {1, 2, 5, 6};
    system.setFunctionNum(alarmFuction);
    system.alarm = new Alarm(system);
    system.alarmCustom = new AlarmCustom(system);
}
```

2055. Write Unit Test Code

- AlarmCustom

```
@Test
public void ControlAlarmListTest() {
    // alarm 리스트 설정
    Time time = new Time( timeFlag: 2);
    time.setTime( hour: 1, min: 2, sec: 3);
    system.alarm.addTimeToAlarmList(time);

    Time time2 = new Time( timeFlag: 2);
    time2.setTime( hour: 1, min: 2, sec: 4);
    system.alarm.addTimeToAlarmList(time2);

    Time time3 = new Time( timeFlag: 2);
    time3.setTime( hour: 1, min: 2, sec: 5);
    system.alarm.addTimeToAlarmList(time3);

    Time time4 = new Time( timeFlag: 2);
    time4.setTime( hour: 1, min: 2, sec: 6);
    system.alarm.addTimeToAlarmList(time4);


    system.alarmCustom.requestAlarmSelectMode();
    java.lang.System.out.println(system.alarmCustom.getAlarm().getAlarmPoi
    assertEquals( expected: 0, system.alarmCustom.getAlarm().getAlarmPointe

    java.lang.System.out.println("size: " + system.alarm.getSize());
```

```
// 알람 리스트 잘 확인 되는지 확인.
system.alarmCustom.changeValue2( diff: 1);
java.lang.System.out.println(system.alarmCustom.getAlarm().getAlarmPointer());
java.lang.System.out.println(system.alarmCustom.getAlarm().getSegmentPointer());
assertEquals( expected: 1, system.alarmCustom.getAlarm().getAlarmPointer()); // 0에서 포인터가 하나 증가하는 지 확인

system.alarmCustom.changeValue2( diff: 1);
java.lang.System.out.println(system.alarmCustom.getAlarm().getAlarmPointer());
java.lang.System.out.println(system.alarmCustom.getAlarm().getSegmentPointer());
assertEquals( expected: 2, system.alarmCustom.getAlarm().getAlarmPointer()); // 1에서 포인터가 하나 증가하는 지 확인

system.alarmCustom.changeValue2( diff: 1);
java.lang.System.out.println(system.alarmCustom.getAlarm().getAlarmPointer());
java.lang.System.out.println(system.alarmCustom.getAlarm().getSegmentPointer());
assertEquals( expected: 3, system.alarmCustom.getAlarm().getAlarmPointer()); // 2에서 포인터가 하나 증가하는 지 확인

system.alarmCustom.changeValue2( diff: 1);
java.lang.System.out.println(system.alarmCustom.getAlarm().getAlarmPointer());
java.lang.System.out.println(system.alarmCustom.getAlarm().getSegmentPointer());
assertEquals( expected: 3, system.alarmCustom.getAlarm().getAlarmPointer()); // 3에서 포인터가 하나 증가하는 지 확인 (마지막이라 안 올라감)
```

```
system.alarmCustom.changeValue2( diff: -1);
java.lang.System.out.println(system.alarmCustom.getAlarm().getAlarmPointer());
java.lang.System.out.println(system.alarmCustom.getAlarm().getSegmentPointer());
assertEquals( expected: 2, system.alarmCustom.getAlarm().getAlarmPointer()); // 3에서 포인터가 하나 감소하는 지 확인

system.alarmCustom.changeValue2( diff: -1);
java.lang.System.out.println(system.alarmCustom.getAlarm().getAlarmPointer());
java.lang.System.out.println(system.alarmCustom.getAlarm().getSegmentPointer());
assertEquals( expected: 1, system.alarmCustom.getAlarm().getAlarmPointer()); // 2에서 포인터가 하나 감소하는 지 확인

system.alarmCustom.changeValue2( diff: -1);
java.lang.System.out.println(system.alarmCustom.getAlarm().getAlarmPointer());
java.lang.System.out.println(system.alarmCustom.getAlarm().getSegmentPointer());
assertEquals( expected: 0, system.alarmCustom.getAlarm().getAlarmPointer()); // 1에서 포인터가 하나 감소하는 지 확인

system.alarmCustom.changeValue2( diff: -1);
java.lang.System.out.println(system.alarmCustom.getAlarm().getAlarmPointer());
java.lang.System.out.println(system.alarmCustom.getAlarm().getSegmentPointer());
assertEquals( expected: 0, system.alarmCustom.getAlarm().getAlarmPointer()); // 0에서 포인터가 하나 감소하는 지 확인 (0 미만으로 안 내려감)
```

2055. Write Unit Test Code

- AlarmCustom

```
@Test
public void SetAlarmIntervalTest() {

    AlarmCustom alarmCustom = system.alarmCustom;

    // alarm 리스트 설정
    Time time = new Time( timeFlag: 2);
    time.setTime( hour: 1, min: 2, sec: 3);
    system.alarm.addTimeToAlarmList(time);


    alarmCustom.requestAlarmSelectMode(); // 알람 선택 모드
    alarmCustom.getAlarm().movePointer( diff: 1); // 포인터

    alarmCustom.requestIntervalSettingMode(); // 알람 인터
    alarmCustom.changeType(); // type  0: 기본 -> 1 : 인터
```

```
assertEquals( expected: 1, alarmCustom.getType());
assertEquals( expected: 2, alarmCustom.getCustomSettingValue()[1]); // 초기값 2인지 확인

//인터벌이 범위 내에서 잘 바뀌는 지 확인.
alarmCustom.changeValue( diff: 1);
assertEquals( expected: 3, alarmCustom.getCustomSettingValue()[1]); // 1이 올라갔는지 확인

alarmCustom.changeValue( diff: 1);
assertEquals( expected: 3, alarmCustom.getCustomSettingValue()[1]); // 한계값 3에 막혀 안 올라가는지 확인

alarmCustom.changeValue( diff: 1);
assertEquals( expected: 3, alarmCustom.getCustomSettingValue()[1]); // 또 한계값 3에 막혀 안 올라가는지 확인

alarmCustom.changeValue( diff: -1);
assertEquals( expected: 2, alarmCustom.getCustomSettingValue()[1]); // 3에서 값이 1 내려가는지 확인

alarmCustom.changeValue( diff: -1);
assertEquals( expected: 1, alarmCustom.getCustomSettingValue()[1]); // 2에서 값이 1 내려가는지 확인

alarmCustom.changeValue( diff: -1);
assertEquals( expected: 1, alarmCustom.getCustomSettingValue()[1]); // 한계값 1에 막혀 값이 안 내려가는지 확인

alarmCustom.changeValue( diff: -1);
assertEquals( expected: 1, alarmCustom.getCustomSettingValue()[1]); // 또 한계값 1에 막혀 값이 안 내려가는지 확인
```

```
alarmCustom.changeValue( diff: 1);
assertEquals( expected: 2, alarmCustom.getCustomSettingValue()[1]); // 1에서 값이 1 올라가는지 확인

alarmCustom.changeValue( diff: 1);
assertEquals( expected: 3, alarmCustom.getCustomSettingValue()[1]); // 2에서 값이 1 올라가는지 확인
```

2055. Write Unit Test Code

- AlarmCustom

```
@Test
public void SetAlarmVolumeTest() {
    AlarmCustom alarmCustom = system.alarmCustom;

    Time time = new Time( timeFlag: 2);
    time.setTime( hour: 1, min: 1, sec: 2);
    system.alarm.addTimeToAlarmList(time);

    alarmCustom.requestAlarmSelectMode(); // 알람 선택 모드
    alarmCustom.getAlarm().movePointer( diff: 1); // 포인터 +1

    alarmCustom.requestIntervalSettingMode(); // 알람 인터벌 설정 모드
    alarmCustom.changeType(); // type  0: 기본 -> 1 : 인터벌
    alarmCustom.changeValue( diff: 1);

    alarmCustom.requestVolumeSettingMode(); // 알람 볼륨 설정 모드
    alarmCustom.changeType(); // type  1: 인터벌 -> 2 : 볼륨


    assertEquals( expected: 2, alarmCustom.getCustomSettingValue()[2]); // 초기 볼륨값 2인지 확인
```

2055. Write Unit Test Code

- AlarmCustom

```
// 범위에 맞게 볼륨값이 잘 바뀌는 지 확인
alarmCustom.changeValue( diff: 1);
assertEquals( expected: 3, alarmCustom.getCustomSettingValue()[2]); // 2에서 값이 1 올라가는지 확인

alarmCustom.changeValue( diff: 1);
assertEquals( expected: 4, alarmCustom.getCustomSettingValue()[2]); // 3에서

alarmCustom.changeValue( diff: 1);
assertEquals( expected: 4, alarmCustom.getCustomSettingValue()[2]); // 한계값

alarmCustom.changeValue( diff: -1);
assertEquals( expected: 3, alarmCustom.getCustomSettingValue()[2]); // 4에서

alarmCustom.changeValue( diff: -1);
assertEquals( expected: 2, alarmCustom.getCustomSettingValue()[2]); // 3에서
```

```
alarmCustom.changeValue( diff: -1);
assertEquals( expected: 1, alarmCustom.getCustomSettingValue()[2]); // 2에서 값이 1 내려가는지 확인

alarmCustom.changeValue( diff: -1);
assertEquals( expected: 0, alarmCustom.getCustomSettingValue()[2]); // 1에서 값이 1 내려가는지 확인

alarmCustom.changeValue( diff: -1);
assertEquals( expected: 0, alarmCustom.getCustomSettingValue()[2]); // 한계값 0에 막혀 값이 안 내려가는지 확인

alarmCustom.changeValue( diff: 1);
assertEquals( expected: 1, alarmCustom.getCustomSettingValue()[2]); // 0에서 값이 1 올라가는지 확인

alarmCustom.changeValue( diff: 1);
assertEquals( expected: 2, alarmCustom.getCustomSettingValue()[2]); // 1에서 값이 1 올라가는지 확인

// 저장이 잘 되는지 확인
alarmCustom.setCustom();
```

2055. Write Unit Test Code

- System

```
@Test
public void ChangeScreenTest() {
    int[] functionNum = system.getFunctionNum();
    int functionNumIdx = system.getFunctionNumIdx();
    int selectedFid;

    functionNumIdx = (functionNumIdx + 1) % 4;
    selectedFid = functionNum[functionNumIdx];
    system.nextFunction();
    assert(selectedFid == system.getSelectedFid());
    functionNumIdx = (functionNumIdx + 1) % 4;
    selectedFid = functionNum[functionNumIdx];
    system.nextFunction();
    assert(selectedFid == system.getSelectedFid());
}
```

2055. Write Unit Test Code

- System

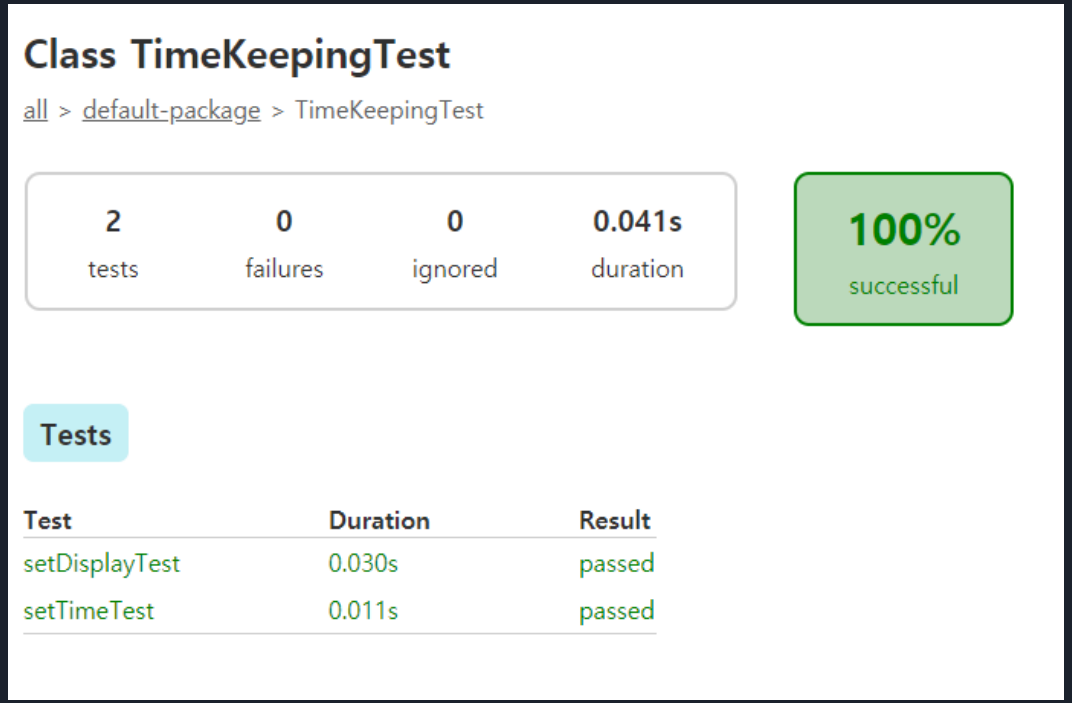
```
@Test
public void CancelTest() {
    TimeKeeping timeKeeping = system.timeKeeping;
    Stopwatch stopwatch = system.stopwatch;
    Timer timer = system.timer;
    D_day d_day = system.d_day;
    Alarm alarm = null;
    AlarmCustom alarmCustom = null;

    //타임 키핑
    timeKeeping.requestTimeSettingMode();
    timeKeeping.cancel();
    assert(timeKeeping.getMode() == 0);

    // 스톱워치
    stopwatch.requestRecordCheckMode();
    stopwatch.cancel();
    assert(stopwatch.getMode() == 0);
```

```
    //타이머
    timer.requestTimerSettingMode();
    timer.cancel();
    assert(timer.getMode() == 0);

    //디데이
    d_day.requestDdaySettingMode();
    d_day.cancel();
    assert(d_day.getMode() == 0);

    int[] alarmFunction = {1, 3, 5, 6};
    system.setFunctionNum(alarmFunction);
    system.alarm = new Alarm(system);
    system.alarmCustom = new AlarmCustom(system);
    alarm = system.alarm;
    alarmCustom = system.alarmCustom;
```

```
    // 알람
    alarm.requestAlarmSettingMode();
    alarm.cancel();
    assert(alarm.getMode() == 0);

    // 알람 커스텀
    alarmCustom.requestIntervalSettingMode();
    alarmCustom.cancel();
    assert(alarmCustom.getMode() == 0);

    alarmCustom.requestVolumeSettingMode();
    alarmCustom.cancel();
    assert(alarmCustom.getMode() == 0);

    alarmCustom.requestAlarmSelectMode();
    alarmCustom.cancel();
    assert(alarmCustom.getMode() == 0);
}
```

2061. Unit Testing

- TimeKeeping



Class TimeKeepingTest

all > default-package > TimeKeepingTest

2	0	0	0.041s
tests	failures	ignored	duration

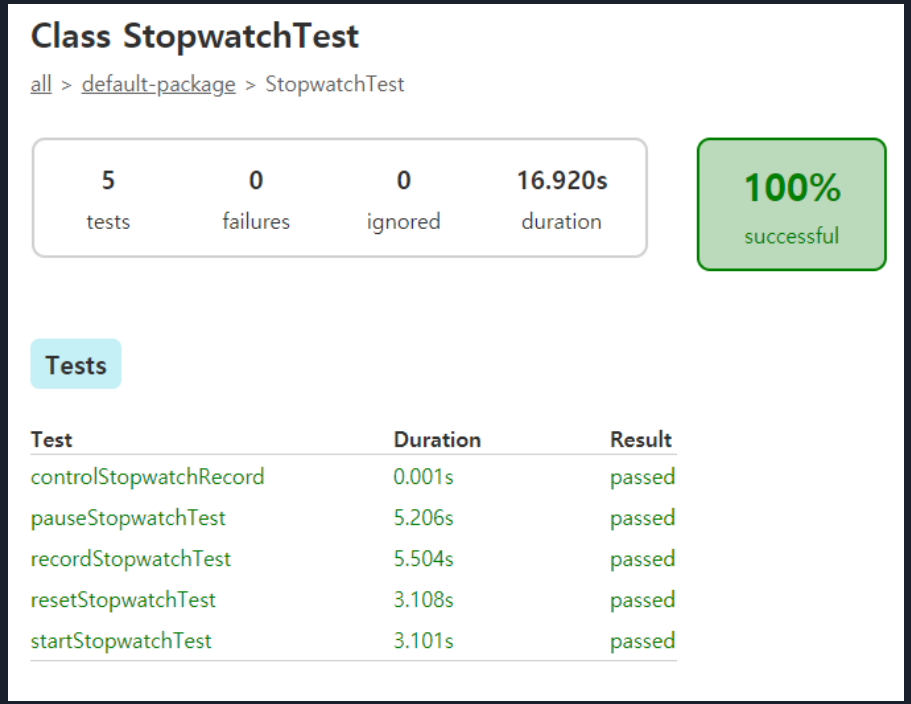
100% successful

Tests

Test	Duration	Result
setDisplayTest	0.030s	passed
setTimeTest	0.011s	passed

2061. Unit Testing

- Stopwatch



Class StopwatchTest

all > default-package > StopwatchTest

5	0	0	16.920s
tests	failures	ignored	duration

100% successful

Tests

Test	Duration	Result
controlStopwatchRecord	0.001s	passed
pauseStopwatchTest	5.206s	passed
recordStopwatchTest	5.504s	passed
resetStopwatchTest	3.108s	passed
startStopwatchTest	3.101s	passed

2061. Unit Testing

- Timer

Class TimerTest

all > default-package > TimerTest

6	0	0	8.411s
tests	failures	ignored	duration

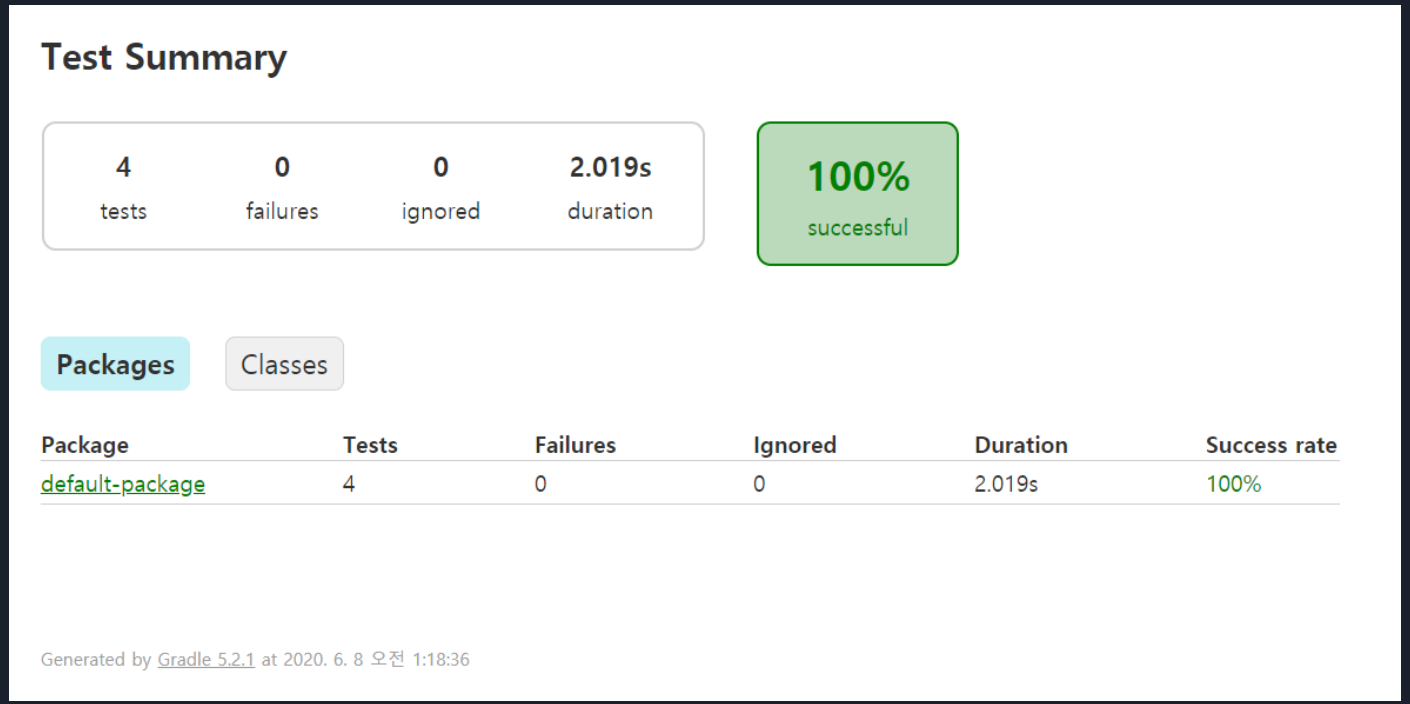
100%
successful

Tests

Test	Duration	Result
beepTimerTest	3.101s	passed
pauseTimerTest	2.204s	passed
resetTimerTest	0.002s	passed
setTimerTest	0s	passed
startTimerTest	3.102s	passed
stopTimerBuzzer	0.002s	passed

2061. Unit Testing

- D-day



Test Summary

4	0	0	2.019s
tests	failures	ignored	duration

100% successful

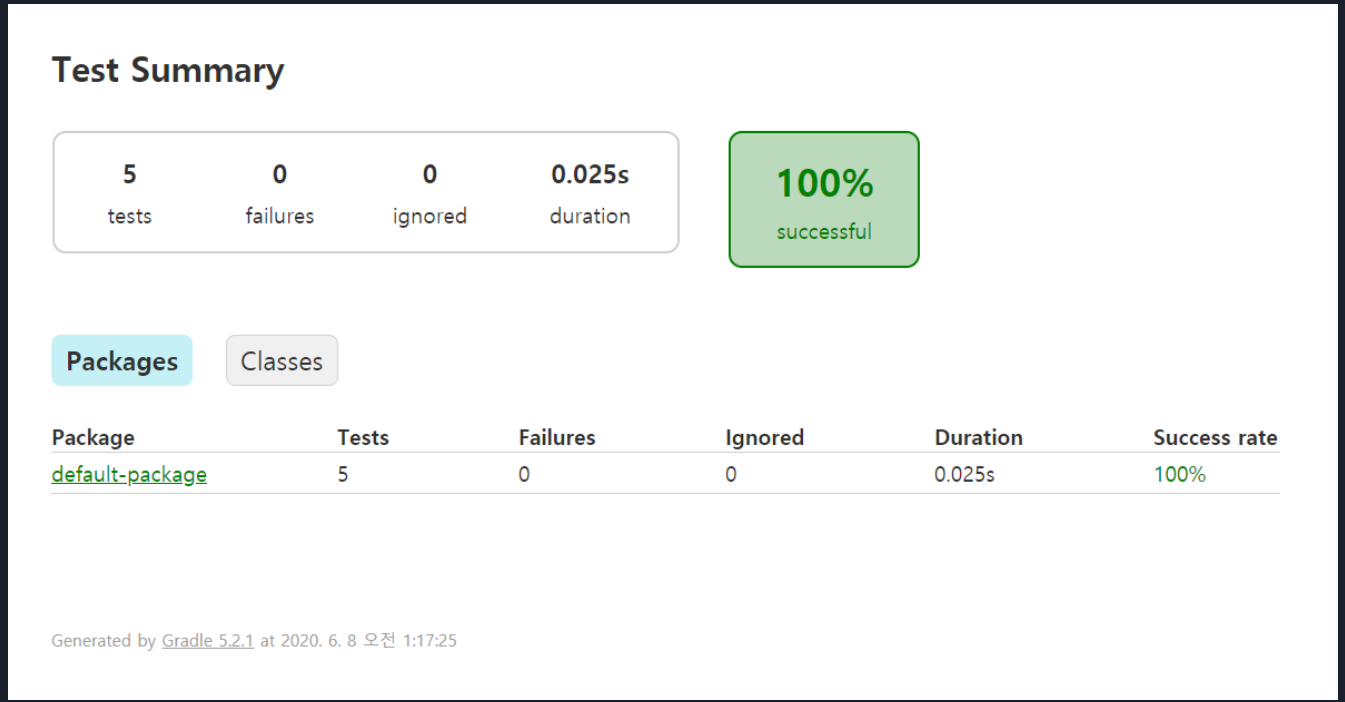
Packages | Classes

Package	Tests	Failures	Ignored	Duration	Success rate
default-package	4	0	0	2.019s	100%

Generated by Gradle 5.2.1 at 2020. 6. 8 오전 1:18:36

2061. Unit Testing

- Alarm



Test Summary

5	0	0	0.025s
tests	failures	ignored	duration

100%
successful

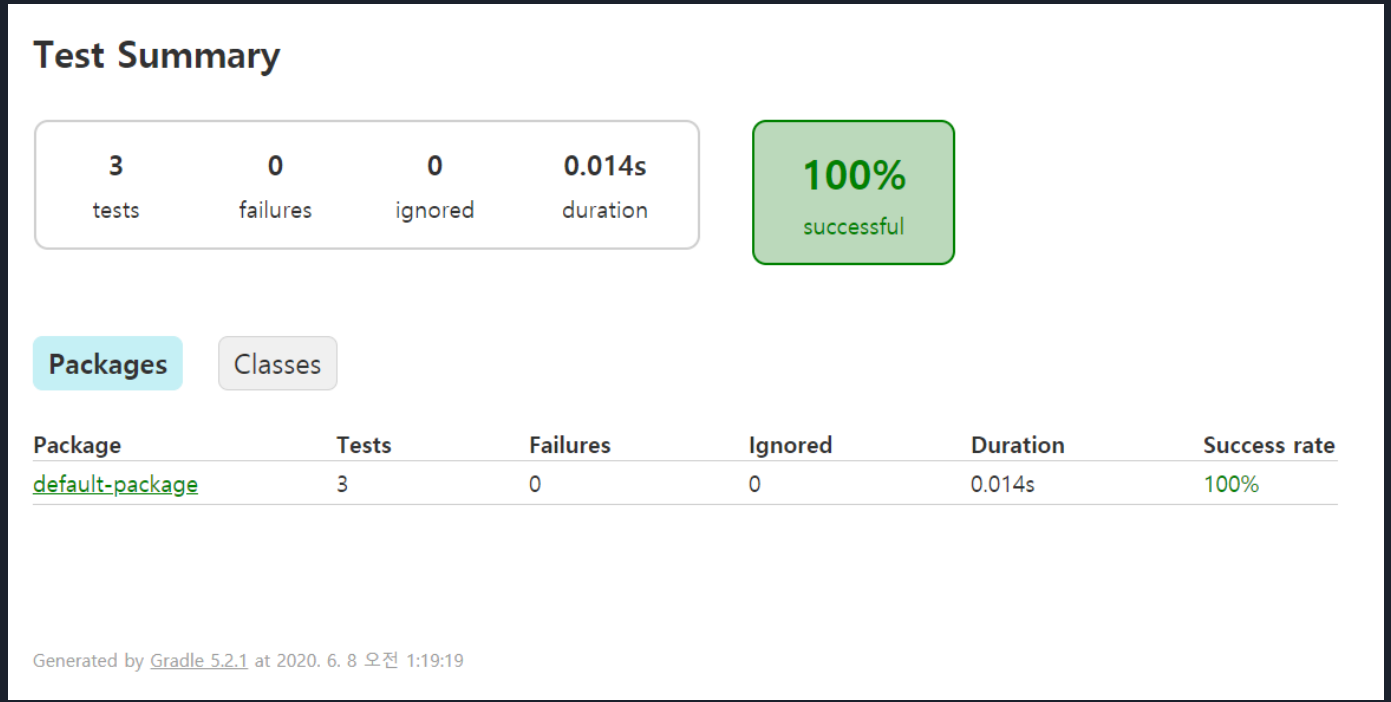
Packages | Classes

Package	Tests	Failures	Ignored	Duration	Success rate
default-package	5	0	0	0.025s	100%

Generated by Gradle 5.2.1 at 2020. 6. 8 오전 1:17:25

2061. Unit Testing

- AlarmCustom



Test Summary

3	0	0	0.014s
tests	failures	ignored	duration

100%
successful

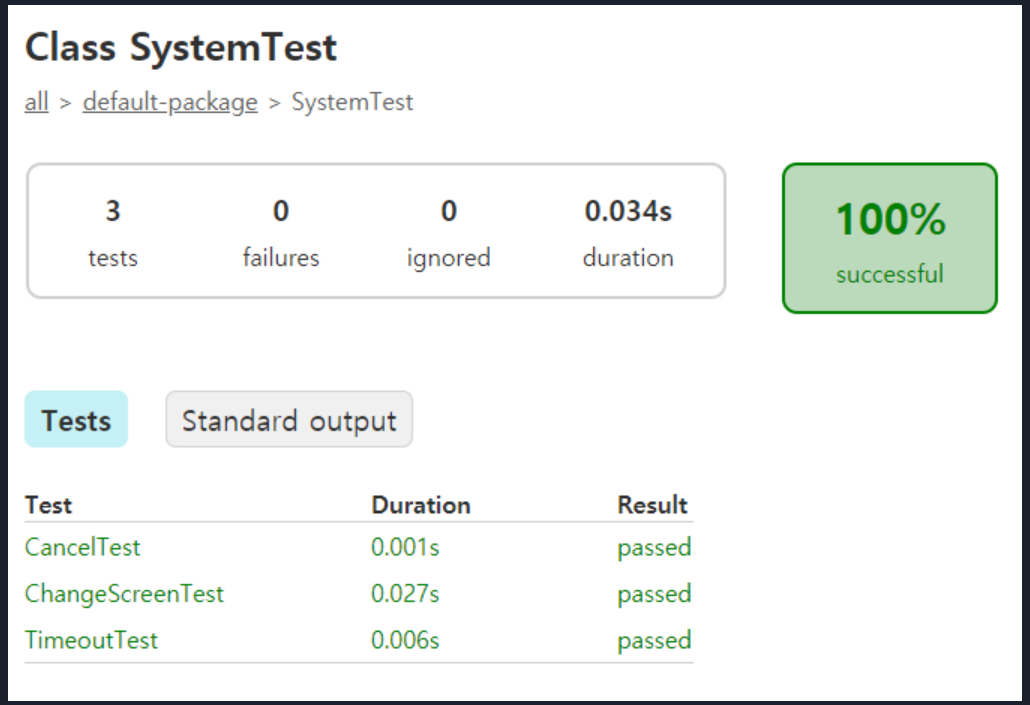
Packages | Classes

Package	Tests	Failures	Ignored	Duration	Success rate
default-package	3	0	0	0.014s	100%

Generated by Gradle 5.2.1 at 2020. 6. 8 오전 1:19:19

2061. Unit Testing

- System



Class SystemTest

all > default-package > SystemTest

3	0	0	0.034s
tests	failures	ignored	duration

100% successful

Tests | Standard output

Test	Duration	Result
CancelTest	0.001s	passed
ChangeScreenTest	0.027s	passed
TimeoutTest	0.006s	passed

2063. System Testing

Test No.	Test 항목	Description	Use Case	System func
1-1	시간, 날짜 설정 시험	년, 월, 일, 시간, 분을 설정할 수 있는지 확인한다. 설정이 잘 되었는지 확인한다.	Set Time	R1.1
1-2	시간, 날짜 저장 시험	설정한 내용이 잘 적용(저장)되어 표시되었는지 확인한다.	Set Time	R1.1
1-3	요일 계산 시험	년, 월, 일에 따른 요일이 정상적으로 계산되어 설정되었는지 확인한다.	Set Time	R1.1
1-4	월별 범주 시험	각 월에 따른 일의 범주를 넘어서지 않는지 확인한다. (ex. 2월 30일은 존재하지 않는다.)	Set Time	R1.1
1-5	설정값 저장 시험	설정값을 저장하고 TimeKeeping 화면으로 돌아가는지 확인한다.	Set Time	R1.1

2063. System Testing

Test No.	Test 항목	Description	Use Case	System func
2-1	시간 출력 시험	시간이 정상적으로 출력되는지 확인한다.	Display Time	R1.2
2-2	시간 변화 시험	시간이 초 단위로 변화하는지 확인한다.	Display Time	R1.2
2-3	정보 표시 시험	설정된 현재 시각, 날짜, 요일, D-day, 알람 개수, 아이콘이 잘 표시되는지 확인한다.	Display Time	R1.2
3-1	기능 전환 시험	선택한 3개의 기능이 설정대로 Change Screen를 통해 전환 가능한지 확인한다.	Set Display	R1.3
3-2	Timekeeping 고정 시험	TimeKeeping 기능은 1번에 고정인지 확인한다.	Set Display	R1.3

2063. System Testing

Test No.	Test 항목	Description	Use Case	System func
4-1	타이머 시간 설정 시험	시, 분, 초를 각각 설정할 수 있는지 확인한다.	Set Timer	R2.1
4-2	타이머 설정 저장 시험	타이머가 잘 설정이 되었는지 확인한다.	Set Timer	R2.1
5-1	타이머 카운트다운 시험	정상적으로 타이머의 시간이 카운트 다운되는 지 확인한다.	Start Timer	R2.2
6-1	타이머 버저 시험	타이머가 끝났을 때 버저가 울리는 지 확인한다.	Beep Timer	R2.3
7-1	타이머 초기화 시험	정상적으로 타이머가 0으로 초기화하는지 확인한다.	Reset Timer	R2.4

2063. System Testing

Test No.	Test 항목	Description	Use Case	System func
8-1	타이머 멈춤 시험	일시정지한 시점에 타이머가 잘 멈추는 지 확인한다.	Pause Timer	R2.5
8-2	타이머 재시작 시험	다시 누르면 타이머가 해당 시각에서부터 움직이는 지 확인한다.	Pause Timer	R2.5
9-1	타이머 버저 멈춤 시험	Timer기능 사용 중 울리는 Buzzer가 멈춰지는지 확인한다.	Stop Timer Buzzer	R2.6
10-1	스톱워치 시작 시험	스톱워치가 정상적으로 시작하는지 확인한다.	Start Stopwatch	R3.1
11-1	스톱워치 정지 시험	스톱워치가 일시정지 하는지 확인한다.	Pause Stopwatch	R3.2
11-2	스톱워치 재시작 시험	다시 누르면 스톱워치가 해당 시각부터 움직이는지 확인한다.	Pause Stopwatch	R3.2

2063. System Testing

Test No.	Test 항목	Description	Use Case	System func
12-1	스톱워치 초기화 시험	스톱워치의 시간을 0으로 정상적으로 초기화하는지 확인한다.	Reset Stopwatch	R3.3
12-2	스톱워치 기록 초기화 시험	스톱워치 기록 리스트가 초기화 되었는지 확인한다.	Reset Stopwatch	R3.3
13-1	스톱워치 기록 시험	정확하게 스톱워치의 해당 시각을 기록하는 지 확인한다.	Record Stopwatch	R3.4
14-1	스톱워치 기록 저장 시험	스톱워치의 기록들이 정상적인지 확인한다.	Display Stopwatch Record	R3.5
14-2	스톱워치 출력 시험	최대 3개까지 보여주는지 확인한다.	Display Stopwatch Record	R3.5

2063. System Testing

Test No.	Test 항목	Description	Use Case	System func
14-3	스톱워치 기록 None 시험	기록이 없으면 None을 띄우는지 확인한다.	Display Stopwatch Record	R3.5
15-1	스톱워치 기록 포인터 시험	스톱워치 기록 포인터가 잘 작동하는지 확인한다.	Control Stopwatch Record	R3.6
15-2	스톱워치 기록 예외처리 시험	범위를 벗어나는 숫자에 대해서 적절하게 예외처리를 하였는지 확인한다.	Control Stopwatch Record	R3.6
15-3	스톱워치 모드 전환 시험	기록이 없으면 기록 확인 모드로 전환되지 않는지 확인한다.	Control Stopwatch Record	R3.6

2063. System Testing

Test No.	Test 항목	Description	Use Case	System func
16-1	알람 저장 시험	사용자가 설정한 알람이 제대로 저장되었는지 확인한다.	Set Alarm	R4.1
16-2	알람 개수 시험	알람이 10개가 넘어가지 않는지 확인한다.	Set Alarm	R4.1
17-1	알람 삭제 시험	설정한 알람이 삭제되었는지 확인한다.	Delete Alarm	R4.2
18-1	알람 Beep 시험	설정된 시간에 알람이 울리는지 확인한다.	Beep Alarm	R4.3
18-2	알람 볼륨, 간격 시험	알람이 커스텀된 볼륨과 간격에 맞게 울리는지 확인한다.	Beep Alarm	R4.3

2063. System Testing

Test No.	Test 항목	Description	Use Case	System func
19-1	알람 버저 멈춤 시험	알람의 버저가 정상적으로 멈추는지 확인한다.	Stop Alarm Buzzer	R4.4
20-1	알람 리스트 출력 시험	알람 리스트가 정상적으로 3개까지 출력되는지 확인한다.	Display Alarm List	R4.5
20-2	알람 None 출력 시험	알람이 없으면 None을 출력하는지 확인한다.	Display Alarm List	R4.5
21-1	알람 선택 모드 시험	알람 선택 모드가 정상적으로 작동하는지 확인한다.	Control Alarm List	R4.6
21-1	알람 모드 예외 시험	알람이 없으면 알람 선택 모드로 진입하지 않는지 확인한다.	Control Alarm List	R4.6

2063. System Testing

Test No.	Test 항목	Description	Use Case	System func
22-1	과거 D-day 시험	과거 날짜를 D-day로 설정할 수 없는지 확인한다.	Set D-day	R5.1
22-2	D-day 범주 시험	각 월에 따른 일의 범주를 넘어서지 않는지 확인한다.	Set D-day	R5.1
22-3	D-day 저장 시험	사용자가 저장 요청한 D-day가 잘 저장되었는지 확인한다.	Set D-day	R5.1
23-1	D-day 테두리 시험	D-day가 0일이면 시계 가장자리에 테두리가 생기는지 확인한다.	Border D-day	R5.2
24-1	D-day 테두리 종료 시험	시계의 테두리 표시가 잘 종료되는지 확인한다.	Stop D-day Border	R5.3

2063. System Testing

Test No.	Test 항목	Description	Use Case	System func
25-1	D-day 삭제 시험	설정한 D-day가 정상적으로 삭제되는지 확인한다.	Delete D-day	R5.4
26-1	알람 선택 모드 시험	알람 선택 모드가 정상적으로 작동하는지 확인한다.	Control Alarm CustomList	R6.1
26-2	알람 선택 모드 예외 시험	알람이 없으면 알람 선택 모드로 진입하지 않는지 확인한다.	Control Alarm CustomList	R6.1
27-1	알람 간격 적용 시험	알람의 간격 설정이 정상적으로 적용되는지 확인한다.	Set Alarm Interval	R6.2
27-2	알람 볼륨 설정 모드 시험	알람 볼륨 설정 모드로 정상적으로 넘어가는지 확인한다.	Set Alarm Interval	R6.2

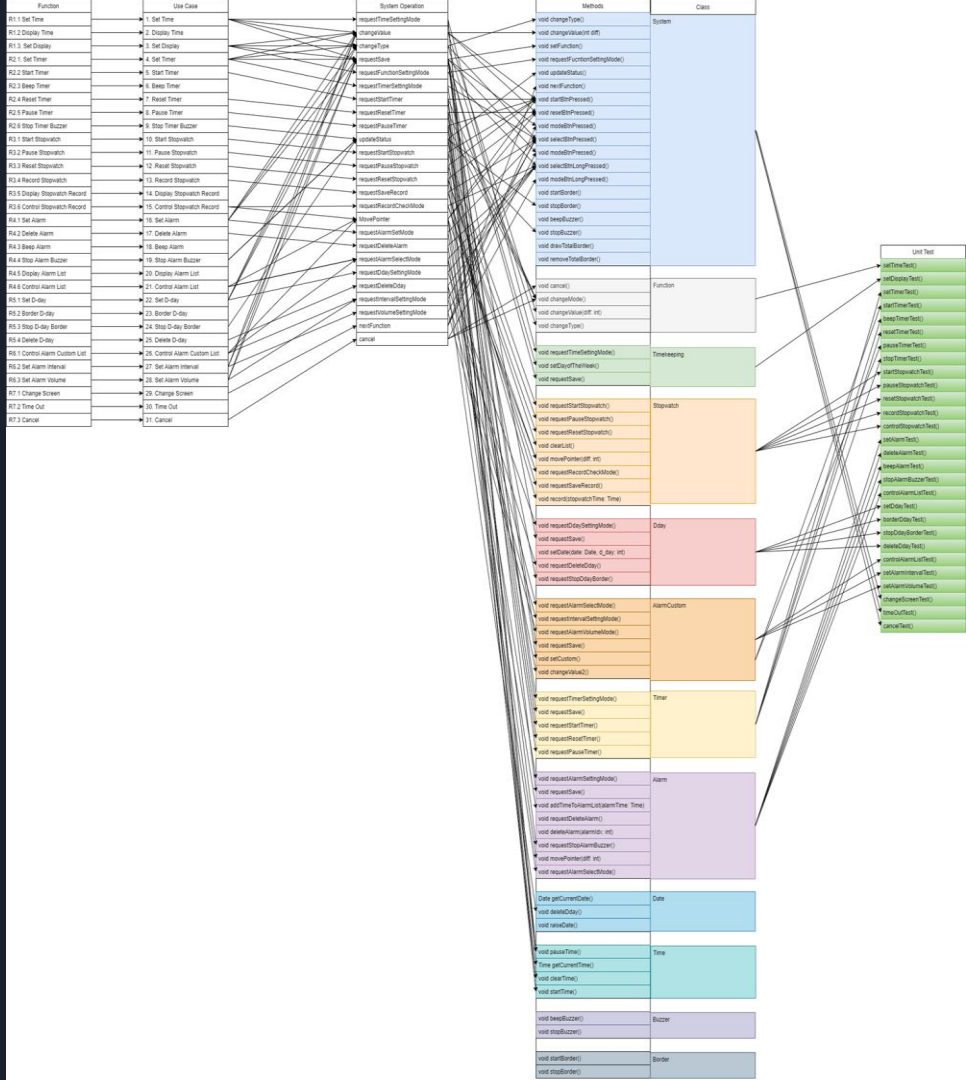
2063. System Testing

Test No.	Test 항목	Description	Use Case	System func
28-1	알람 볼륨 설정 시험	알람의 볼륨 설정이 정상적으로 적용되는지 확인한다.	Set Alarm Volume	R6.3
28-2	알람 커스텀 기본 모드 시험	저장 후 알람 커스텀 기본 모드로 돌아오는지 확인한다.	Set Alarm Volume	R6.3
29-1	화면 전환 시험	화면이 Set Display에서 설정한 순서대로 정상적으로 전환되는지 확인한다.	Change Screen	R7.1
30-1	타임 아웃 시험	Timeout이 발생했을 때 정상적으로 TimeKeeping 화면의 기본 모드로 돌아가는지 확인한다.	Time Out	R7.2
30-2	타임 아웃 시점 시험	정확히 아무런 동작을 하지 않은 지 10분이 지났을때 작동하는 지 확인한다.	Time Out	R7.2

2063. System Testing

Test No.	Test 항목	Description	Use Case	System func
30-3	정보 미저장 시험	입력하던 정보가 저장되지 않았는지 확인한다.	Time Out	R7.2
31-1	취소 화면 기본 모드 시험	해당 화면의 기본 모드로 돌아가는지 확인한다.	Cancel	R7.3
31-2	취소 정보 미저장 시험	입력하던 정보가 저장되지 않았는지 확인한다.	Cancel	R7.3

2066. Testing Traceability Analysis



QnA