

SSS

201111383 전훈

201311259 권오승

201311292 유효상

201511266 배율희

Index

1. Implement Class & Method Definitions
2. Implement Windows
3. Write Unit Test Code
4. Unit Testing
5. System Testing
6. Testing Traceability Analysis
7. 프로그램 실행 화면

Implement Class & Method Definitions

Type	Class
Name	MainController
Purpose	DataController와 Calculator를 통해 프로그램의 전체 조작을 다루는 클래스
Overview	N/A
Cross Reference	Function: All Use Case: All
Exceptional Courses of Events	N/A

Type	Class
Name	Category
Purpose	Value와 DataValue를 함께 다루는 클래스
Overview	N/A
Cross Reference	Function: R2.1, R2.2, R2.3, R3.1, R3.2, R3.3, R3.4 Use Case: Add Value, Delete Value, Edit Value, Check Single, Check Error, Add Property, Add If-Property
Exceptional Courses of Events	N/A

Type	Class
Name	DataController
Purpose	Category 및 Value의 입력을 다루는 클래스
Overview	N/A
Cross Reference	Function: R1.1, R1.2, R1.3, R2.1, R2.2, R2.3, R3.1, R3.2, R3.3, R3.4 Use Case: Add Category, Delete Category, Edit Category, Add Value, Delete Value, Edit Value, Check Single, Check Error, Add Property, Add If-Property
Exceptional Courses of Events	N/A

Type	Class
Name	Value
Purpose	데이터를 관리하기 쉬운 형태로 Value를 다루는 클래스
Overview	N/A
Cross Reference	Function: R3.1, R3.2, R3.3, R3.4 Use Case: Check Single, Check Error, Add Property, Add If-Property
Exceptional Courses of Events	N/A

Implement Class & Method Definitions

Type	Class
Name	DataValue
Purpose	계산하기 원활한 형태로 Value를 다루는 클래스
Overview	N/A
Cross Reference	Function: R2.1, R2.2, R2.3, R3.1, R3.2, R3.3, R3.4 Use Case: Add Value, Delete Value, Edit Value, Check Single, Check Error, Add Property, Add If-Property
Exceptional Courses of Events	N/A

Type	Class
Name	Calculator
Purpose	DataValue를 참조하여 Test Case 개수를 계산하는 클래스
Overview	N/A
Cross Reference	Function: R4.1 Use Case: Calculate
Exceptional Courses of Events	N/A

Implement Class & Method Definitions

Type	Method
Name	addCategory()
Purpose	datavalue에 카테고리 추가
Abstract operation	datavalue에 categoryName을 사용하여 category를 추가한다
Input	String categoryName
Output	N/A
Cross Reference	Function: addCategory Use Case: addCategory
Exceptional Courses of Events	N/A

Type	Method
Name	editCategory()
Purpose	datavalue에서 카테고리 이름 편집
Abstract operation	datavalue에서 categoryName을 사용하여 그 이름을 찾아 category를 편집할 수 있게한다
Input	String oldname, String newname
Output	void
Cross Reference	Function: editCategory Use Case: editCategory
Exceptional Courses of Events	N/A

Type	Method
Name	deleteCategory()
Purpose	datavalue에서 카테고리 삭제
Abstract operation	datavalue에서 categoryName을 사용하여 그 이름을 찾아 category를 삭제한다
Input	String categoryName
Output	void
Cross Reference	Function: deleteCategory Use Case: deleteCategory
Exceptional Courses of Events	N/A

Type	Method
Name	getCategoryIndex()
Purpose	categoryName에 따른 categoryIndex를 가져온다
Abstract operation	현재 category의 categoryIndex를 반환한다
Input	String categoryName
Output	int
Cross Reference	Function: N/A Use Case: N/A
Exceptional Courses of Events	N/A

Implement Class & Method Definitions

Type	Method
Name	addValue()
Purpose	Category에 따른 Value를 추가한다
Abstract operation	datavalue에 categoryName, valueName를 사용하여 추가한다
Input	String categoryName, String valueName
Output	void
Cross Reference	Function: addValue Use Case: All
Exceptional Courses of Events	N/A

Type	Method
Name	displayCategoryList()
Purpose	categoryList를 출력한다
Abstract operation	category의 크기만큼 categoryName을 가져와 categoryList를 출력한다.
Input	N/A
Output	void
Cross Reference	Function: displayCategoryList Use Case: displayCategoryList
Exceptional Courses of Events	N/A

Type	Method
Name	deleteValue()
Purpose	Category에 따른 Value를 삭제한다
Abstract operation	datavalue에 categoryName, valueName를 사용하여 value를 찾아 삭제한다
Input	String categoryName, String valueName
Output	void
Cross Reference	Function: deleteValue Use Case: All
Exceptional Courses of Events	N/A

Type	Method
Name	addCategory()
Purpose	categoryList를 호출하여 카테고리를 추가한다
Abstract operation	headerCategory를 기준으로 categoryName을 사용하여 카테고리를 추가 시킨다
Input	String categoryName
Output	void
Cross Reference	Function: addCategory Use Case: addCategory
Exceptional Courses of Events	N/A

Implement Class & Method Definitions

Type	Method
Name	deleteCategory()
Purpose	카테고리 리스트를 호출하여 카테고리를 삭제한다
Abstract operation	headerCategory에서 categoryName을 사용하여 카테고리를 삭제 시킨다
Input	String categoryName
Output	void
Cross Reference	Function: deleteCategory Use Case: deleteCategory
Exceptional Courses of Events	N/A

Type	Method
Name	editCategory()
Purpose	카테고리 리스트를 호출하여 카테고리를 편집한다
Abstract operation	headerCategory에서 categoryName을 사용하여 카테고리를 찾고 편집할 수 있게 한다
Input	String oldName, String newName
Output	void
Cross Reference	Function: editCategory Use Case: editCategory
Exceptional Courses of Events	N/A

Type	Method
Name	addValue()
Purpose	category에서 value값을 추가시킨다.
Abstract operation	headerCategory에서 headerValue에 valueName을 사용하여 value를 추가한다
Input	String categoryName, valueName, type, prevName
Output	void
Cross Reference	Function: addValue Use Case: addValue
Exceptional Courses of Events	N/A

Type	Method
Name	deleteValue()
Purpose	category의 value를 삭제한다
Abstract operation	headerCategory에서 headerValue에 valueName을 사용하여 value를 찾고 삭제한다
Input	String categoryName, valueName
Output	void
Cross Reference	Function: deleteValue Use Case: deleteValue
Exceptional Courses of Events	N/A

Implement Class & Method Definitions

Type	Method
Name	editValue()
Purpose	category의 value를 편집한다
Abstract operation	headerCategory에서 headerValue에 valueName을 사용하여 value를 찾고 편집 한다
Input	String oldname, newname
Output	void
Cross Reference	Function: editValue Use Case: editValue
Exceptional Courses of Events	N/A

Type	Method
Name	getCategoryIndex()
Purpose	categoryIndex를 반환한다
Abstract operation	categoryIndex를 반환한다
Input	N/A
Output	int
Cross Reference	Function: displayCategoryList Use Case: displayCategoryList
Exceptional Courses of Events	N/A

Type	Method
Name	addValue()
Purpose	category에 value를 추가한다
Abstract operation	headercategory를 기준으로 앞 valueName를 찾고 valueName을 사용하여 value를 추가한다
Input	String name, type, pname
Output	void
Cross Reference	Function: addValue Use Case: addValue
Exceptional Courses of Events	N/A

Type	Method
Name	deleteValue()
Purpose	category의 value를 value이름을 사용하여 삭제한다
Abstract operation	headercategory를 기준으로 앞 valueName를 찾고 valueName을 사용하여 value를 삭제한다
Input	String name
Output	void
Cross Reference	Function: deleteValue Use Case: deleteValue
Exceptional Courses of Events	N/A

Implement Class & Method Definitions

Type	Method
Name	getDv()
Purpose	DataValue를 반환한다
Abstract operation	DataValue를 반환한다
Input	void
Output	DataValue
Cross Reference	Function: N/A Use Case: N/A
Exceptional Courses of Events	N/A

Type	Method
Name	getDv()
Purpose	DataValue를 반환한다
Abstract operation	DataValue를 반환한다
Input	void
Output	DataValue
Cross Reference	Function: N/A Use Case: N/A
Exceptional Courses of Events	N/A

Type	Method
Name	getValueIndex()
Purpose	valueIndex를 반환한다
Abstract operation	valueIndex를 반환한다
Input	N/A
Output	int
Cross Reference	Function: N/A Use Case: N/A
Exceptional Courses of Events	N/A

Type	Method
Name	getPropertyArr()
Purpose	property를 반환한다
Abstract operation	property를 반환한다
Input	N/A
Output	String[]
Cross Reference	Function: N/A Use Case: N/A
Exceptional Courses of Events	N/A

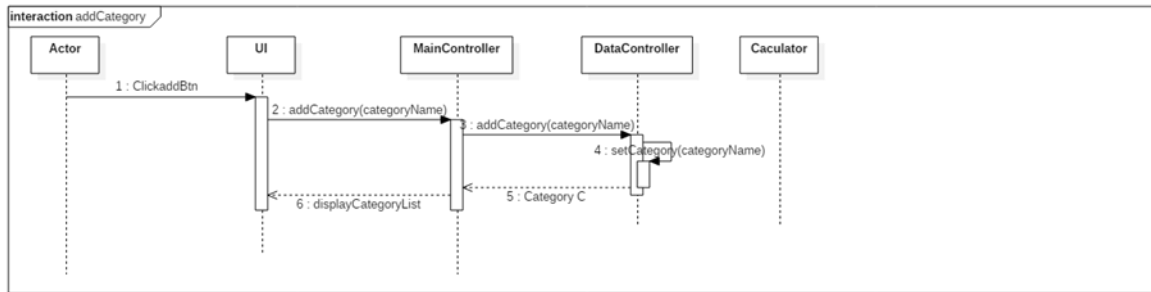
Implement Class & Method Definitions

Type	Method
Name	getIfpropertyArr()
Purpose	IfpropertyArr를 반환한다
Abstract operation	IfpropertyArr를 반환한다
Input	N/A
Output	String[]
Cross Reference	Function: N/A Use Case: N/A
Exceptional Courses of Events	N/A

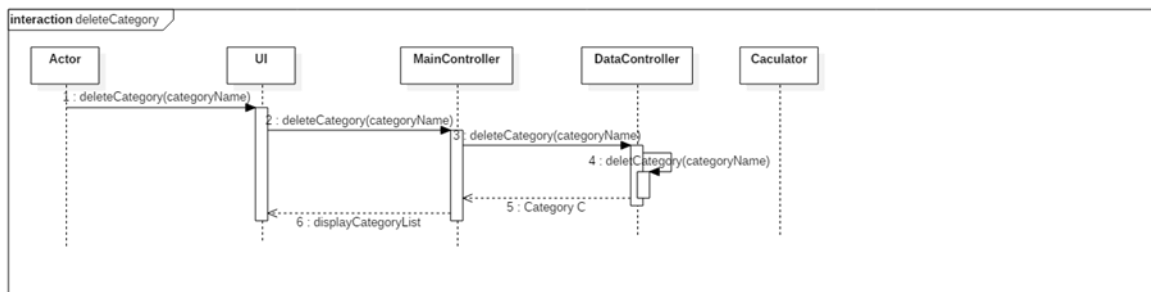
Type	Method
Name	addDataValue()
Purpose	value의 속성의 결과를 탐색하여 추가시킨다
Abstract operation	value의 속성을 순회하며 탐색하여 dataValue에 해당하는 속성에 value를 추가시킨다
Input	Value
Output	void
Cross Reference	Function: N/A Use Case: N/A
Exceptional Courses of Events	N/A

Type	Method
Name	calculate()
Purpose	category와 value들을 받아서 testcase 수를 계산한다
Abstract operation	Category c에 연결된 모든 category를 size만큼 탐색하여 property, if property, single, error, valid를 모두 고려하여 계산한 testcase수를 반환한다
Input	Category c
Output	int
Cross Reference	Function: calculate Use Case: calculate
Exceptional Courses of Events	N/A

Implement Windows

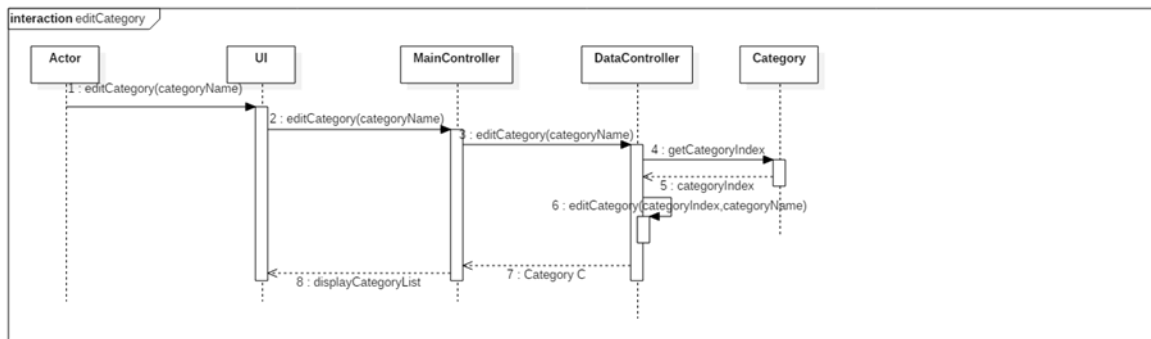


Name	ClickaddBtn
Responsibilities	GUI의 Add 버튼을 누른다.
Type	GUI
Cross Reference	R1.1
Notes	GUI 왼쪽 하단의 '+' 버튼을 누른다.
Pre-Conditions	N/A
Post-Donditions	추가된 Category를 화면에 보여준다.



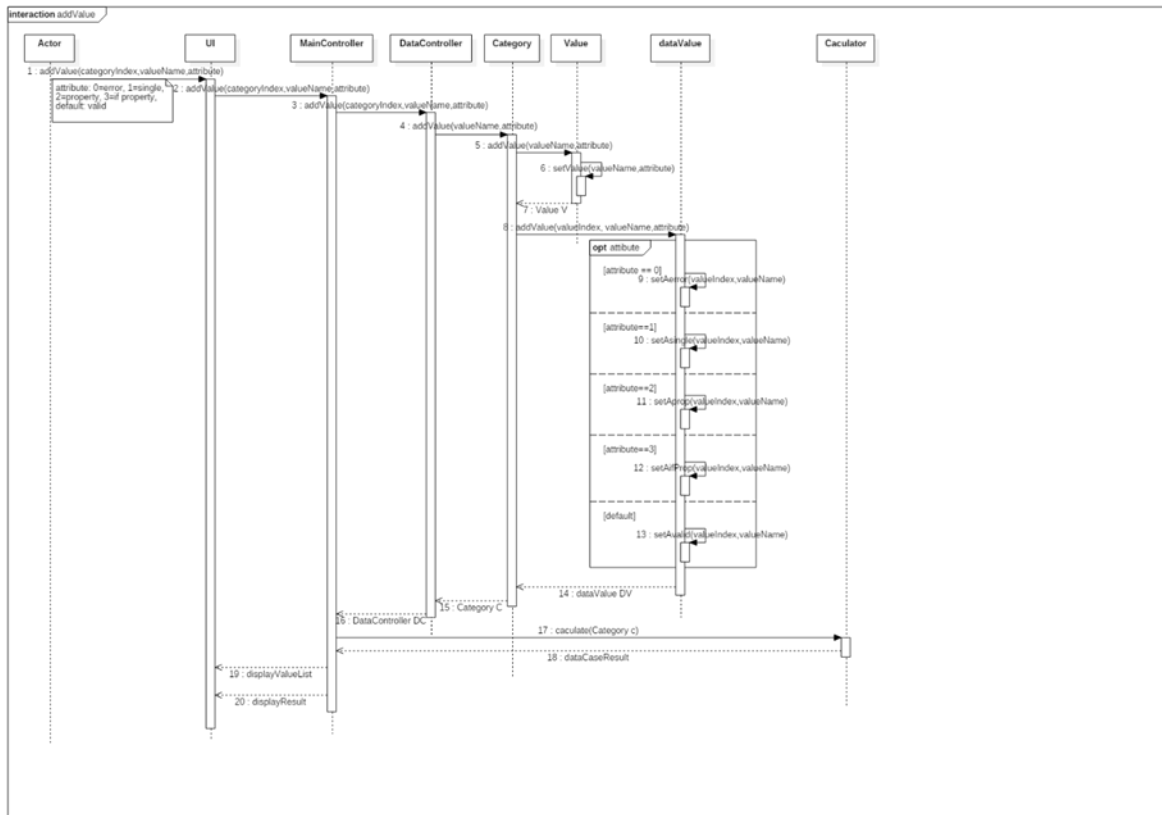
Name	deleteCategory
Responsibilities	GUI의 Delete 버튼을 누른다.
Type	GUI
Cross Reference	R1.2
Notes	삭제할 Category를 선택하고 화면 왼쪽 하단의 '-' 버튼을 눌러 Category를 삭제한다.
Pre-Conditions	N/A
Post-Donditions	선택한 Category를 화면에서 제거한다.

Implement Windows



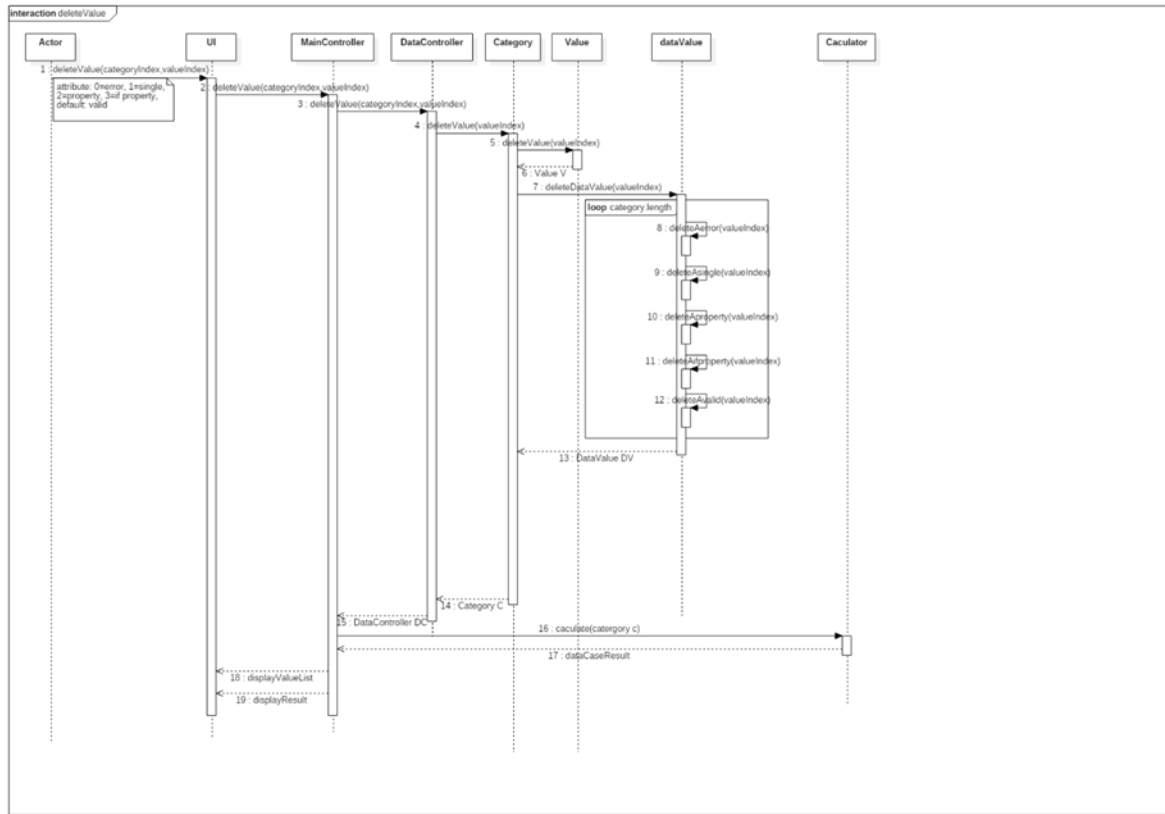
Name	editCategory
Responsibilities	GUI의 Edit 버튼을 누른다.
Type	GUI
Cross Reference	R1.3
Notes	Category를 선택하고 Text Field에 이름을 입력하여 Category의 이름을 변경한다.
Pre-Conditions	N/A
Post-Conditions	바뀐 Category 이름이 화면에 표시된다.

Implement Windows



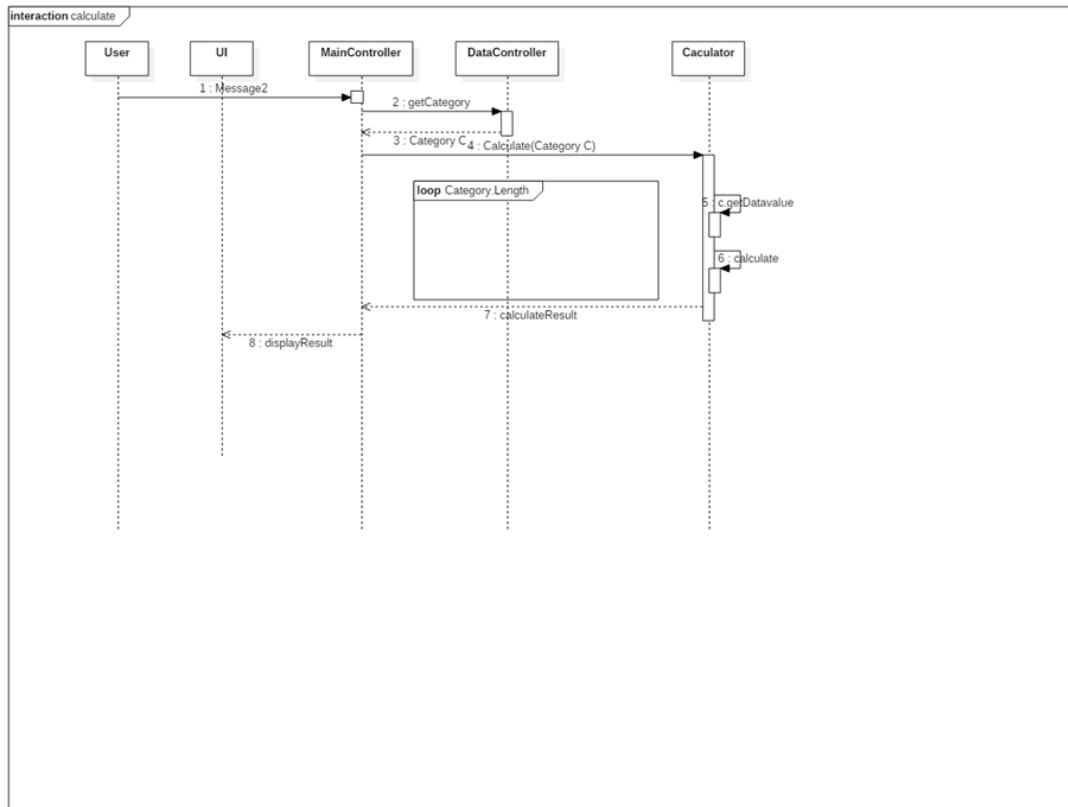
Name	addValue
Responsibilities	GUI의 Add 버튼을 누른다.
Type	GUI
Cross Reference	R2.1
Notes	GUI 오른쪽 하단의 '+' 버튼을 눌러 Value를 추가한 후, Value의 이름, Single, Error, Property 등의 Constraint를 설정한 후 Add 버튼을 눌러 Value를 추가한다.
Pre-Conditions	N/A
Post-Conditions	추가된 Value가 화면에 표시된다.

Implement Windows



Name	deleteValue
Responsibilities	GUI의 Delete 버튼을 누른다.
Type	GUI
Cross Reference	R2.2
Notes	삭제할 Value를 선택하고 화면 오른쪽 하단의 ‘ ‘ 버튼을 눌러 Value를 삭제한다.
Pre-Conditions	N/A
Post-Conditions	선택한 Value를 화면에서 제거한다.

Implement Windows



Name	Message2
Responsibilities	Value를 추가하거나, Category 혹은 Value를 제거한다.
Type	GUI
Cross Reference	R4.1
Notes	Value를 추가하거나, Category 혹은 Value를 제거할 때 시스템 내부적으로 Calculate가 동작한다.
Pre-Conditions	N/A
Post-Conditions	계산 결과를 화면에 출력한다.

Write Unit Test Code

```
DataController dt = new DataController();
String expected = "category";
String changed = "newName";

@Test
public void testAddCategory() {
    dt.addCategory(expected);
    assertEquals(0, dt.getCategoryIndex(expected));
}

@Test
public void testEditCategory() {
    int index = dt.getCategoryIndex(expected);
    dt.editCategory(expected, "newName");
    assertEquals(index, dt.getCategoryIndex("newName"));
}

@Test
public void testDeleteCategory() {
    dt.deleteCategory(changed);
    assertEquals(0, dt.getCategoryIndex(changed));
}
```

```
@Test
public void testSetName() {
    Category c = new Category();
    String expected = "category";
    c.setName(expected);
    assertEquals(c.getName(), expected);
}

@Test
public void testAddValue() {
    Category c = new Category();
    c.addValue("value", "valid", null);
    assertEquals(c.getValue().size(), 1);
}

@Test
public void testDeleteValue() {
    Category c = new Category();
    c.addValue("value", "valid", null);
    c.deleteValue("valid");
    assertEquals(c.getValue().size(), 0);
}
```


Write Unit Test Code

```
@Test
public void testSetName() {
    Category c = new Category();
    String expected = "category";
    c.setName(expected);
    assertEquals(c.getName(), expected);
}

@Test
public void testAddValue() {
    Category c = new Category();
    c.addValue("value", "valid", null);
    assertEquals(c.getValue().size(), 1);
}

@Test
public void testDeleteValue() {
    Category c = new Category();
    c.addValue("value", "valid", null);
    c.deleteValue("valid");
    assertEquals(c.getValue().size(), 0);
}
```

```
@Test
public void testAddSingle() {
    DataValue dv = new DataValue();
    dv.addSingle("single");
    assertEquals(1, dv.getSingleCount());
}


@Test
public void testAddError() {
    DataValue dv = new DataValue();
    dv.addError("error");
    assertEquals(1, dv.getErrorCount());
}

@Test
public void testAddValid() {
    DataValue dv = new DataValue();
    dv.addValid("valid");
    assertEquals(1, dv.getValidCount());
}

@Test
public void testAddPropertyValue() {
    DataValue dv = new DataValue();
    Value v = new Value();
    dv.addProperty(v);
    assertEquals(1, dv.getPropertyCount());
}
```

Unit Testing

Runs: 3/3 Errors: 1 Failures: 0




Controller.DataControllerTest [Runner: JUnit 4] (0.000 s)

- testAddCategory (0.000 s)
- testEditCategory (0.000 s)
- testDeleteCategory (0.000 s)

Finished after 0.022 seconds

Runs: 3/3 Errors: 0 Failures: 0




Model.CategoryTest [Runner: JUnit 4] (0.000 s)

- testDeleteValue (0.000 s)
- testAddValue (0.000 s)
- testSetName (0.000 s)

Finished after 0.02 seconds

Runs: 2/2 Errors: 0 Failures: 0




Model.ValueTest [Runner: JUnit 4] (0.000 s)

- testSetIfPropertyArr (0.000 s)
- testSetPropertyArr (0.000 s)

Finished after 0.016 seconds

Runs: 4/4 Errors: 0 Failures: 0



Model.DataValueTest [Runner: JUnit 4] (0.000 s)

- testAddPropertyValue (0.000 s)
- testAddError (0.000 s)
- testAddValid (0.000 s)
- testAddSingle (0.000 s)

System Testing

Test Number	Test 항목	Description	Use-Case	System Function
1	Add Category 버튼 시험	User가 Add한 Category가 List에 Add되는지 확인한다.	1. Add Category	R1.1
2	Delete Category 버튼 시험	User가 Delete한 Category가 List에서 Delete되는지 확인한다.	2. Delete Category	R1.2
3	Edit Category 버튼 시험	User가 선택한 Category의 이름이 바뀌었는지 확인한다.	3. Edit Category	R1.3
4	Add Value 버튼 시험	User가 선택한 Category에 Value가 추가되었는지 확인한다.	5. Add Value	R2.1
5	Delete Value 버튼 시험	User가 선택한 Category에 선택한 Value가 Delete되었는지 확인한다.	6. Delete Value	R2.2
6	Edit Value 버튼 시험	User가 선택한 Value의 이름이 바뀌었는지 확인한다.	7. Edit Value	R2.3
7	Check Single 버튼 시험	User가 선택한 Category에 선택한 Value에 Single이 부여되었는지 확인한다.	9. Check Single	R3.1
8	Check Error 버튼 시험	User가 선택한 Category에 선택한 Value에 Error가 부여되었는지 확인한다.	10. Check Error	R3.2
9	Add Property 버튼 시험	User가 선택한 Category에 선택한 Value에 Property가 부여되었는지 확인한다.	11. Add Property	R3.3
10	Add If-Property 버튼 시험	User가 선택한 Category에 선택한 Value에 If-Property가 부여되었는지 확인한다.	12. Add If-Property	R3.4
11	Calculate 시험	매 입력마다 Test Case 계산이 제대로 일어나는지 확인한다.	14. Calculate	R4.1

Q&A