**Test Plan:**

The default constructors will be tested highlighting what the data type is, of the name, and the subject by Creating a ColorGrid Object with the default constructor and check the same.

Create a non-default constructor and with ColorGrid object and verify the results. Testing all the get methods (Accessor) and testing all set methods (mutators). Test the display method.

The mutator methods will be tested with both valid and invalid (relative to the assignment specification) data. These methods are methods that change the underlying value of an attribute of a class. The accessor methods are tested as well to highlight how inputted information relevant to the ColorGrid class can be retrieved and displayed.

**Test Strategy:**

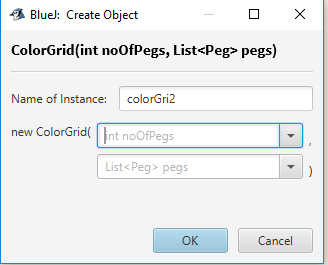
**Constructor Test:**

Create a ColorGrid object and inspect it.

**Test:** When the game starts, the constructor should ask for number of pegs which can be stored in the array list and the list of peg objects

**Expected Result:** The application asks for number of pegs and the list of Peg’s object

**Actual Result:**

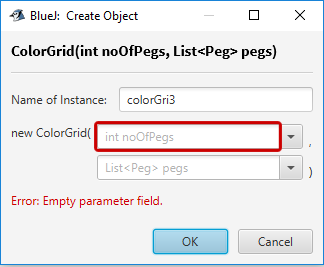
****

**Test:** If no value is provided to the constructor, the application should give an error for empty parameters

**Expected Result:**

The application should give an error of empty parameters because the expected value for the number of pegs is an integer and for the list of Peg’s object is an array list of type Peg.

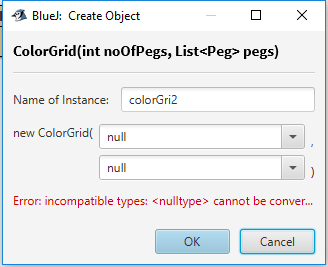
**Actual Result:**

****

**Test:** If a string is supplied for the number of pegs or the list of Peg’s object, the application should reject the string

**Expected Result:** The application should reject the string because the expected value for the number of pegs is an integer and for the list of Peg’s object is an arraylist of type Peg

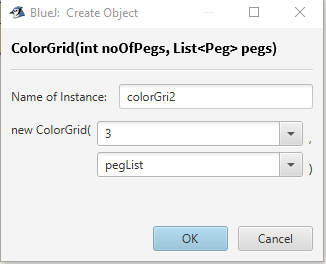
**Actual Result:**



**Test:** Supply the right parameters for the number of pegs as well as the list of Peg’s object

**Expected Result:** The application should create a new colorGrid object with the supplied parameters

**Actual Result:**



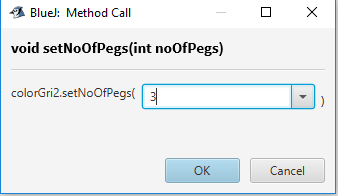
**Object has been created: **

**Here, we have used a list of Pegs created**: 

**Test:** Set number of Pegs

**Expected Result:** When the setNoOfPegs method is called the application should set the number of pegs for the object

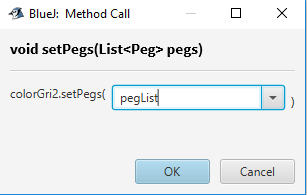
**Actual Result:**

****

**Test:** Set list of Peg’s object

**Expected Result:** When the setPegs method is called the application should set the list of Pegs for the object

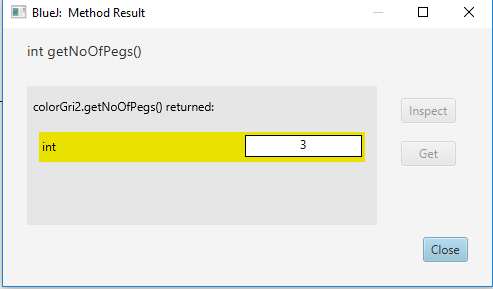
**Actual Result:**

****

**Test:** get number of pegs

**Expected Result:** When the getNoOfPegs method is called, the application should get the number of pegs

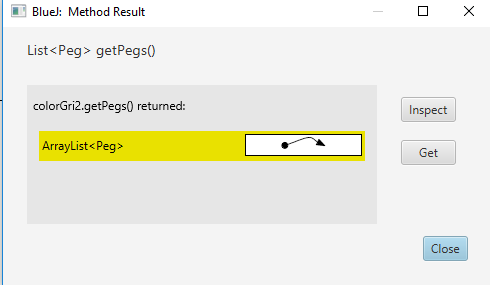
**Actual Result:**

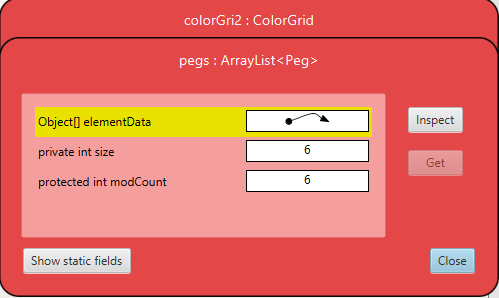
****

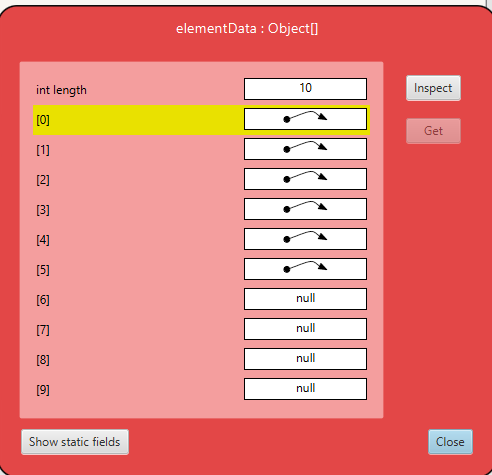
**Test:** get list of Peg’s object

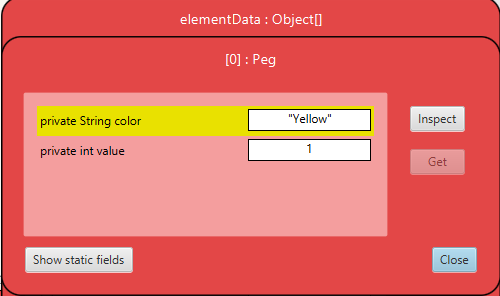
**Expected Result:** When the getPegs method is called, the application should get the list of Peg’s object

**Actual Result:**

****

****

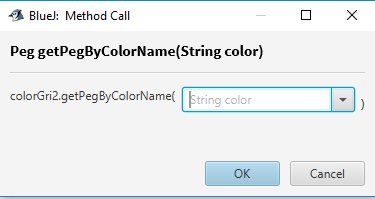
****

****

**Test:** get peg by the name of the color

**Expected Result:** When the getPegByColorName method is called it should ask a string parameter

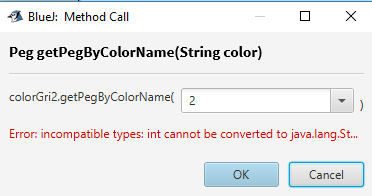
**Actual Result:**

****

**Test:** supply wrong parameters to the getPegByColorName method

**Expected Result:** When the getPegByColorName method is called with wrong parameters, let say an int value it should give error because a String value is expected

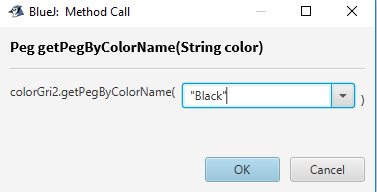
**Actual Result:**

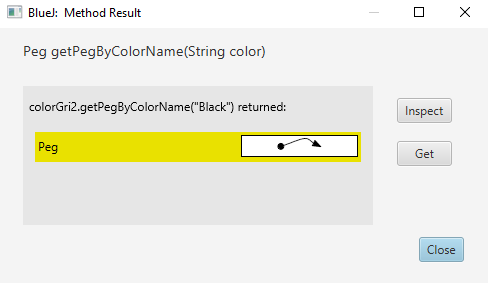
****

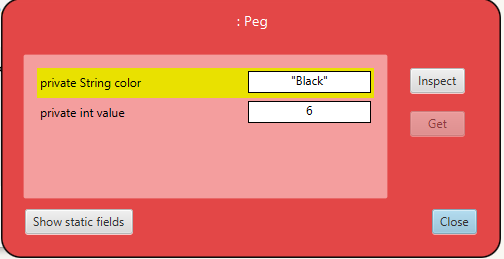
**Test:** supply String parameters to the getPegByColorName method

**Expected Result:** When the getPegByColorName method is called with correct data, it should give the value associated with the color

**Actual Result:**

****

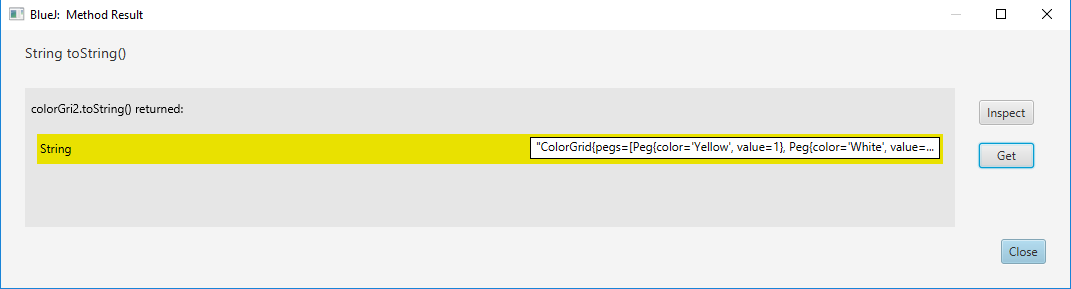
****

****

**Test:** Display Player

**Expected Result:** When the toString method is called the application should display the number of pegs and the list of Peg’s object

**Actual Result:**

****