Diagram Test

1. public void addRelationship()
    a. Testing that our addRelationship() method in Diagram.java adds a Relation r to the list of all relationships in the diagram
    b. Passes
2. public void deleteRelationship()
    a. Testing that our deleteRelationship() method in Diagram.java deletes a Relation r from the list of all relationships in the diagram
    b. Passes
3. public void addClass()
    a. Testing that our addClass() method in Diagram.java adds a ClassComponent c to the list of all classes in the diagram
    b. Passes
4. public void deleteClass()
    a. Testing that our deleteClass() method in Diagram.java deletes a ClassComponent c from the list of all classes in the diagram
    b. Passes
5. public void getAllClasses()
    a. Testing that our getAllClasses() method in Diagram.java returns the list of all classes in the diagram
    b. Passes
6. public void getAllRelationships()
    a. Testing that our getAllRelationships() method in Diagram.java returns the list of all relationships in the diagram
    b. Passes

Relation Test

1. Public Void construct()
    a. Testing that after a relation is added, the source and destination components of the relation are assigned correctly.
    b. Passes
2. public void componentTest1()
    a. Testing that when a Relation of type Subtyping is added that's it's type really is.
    b. Passes
3. public void componentTest2()
    a. Testing that when a Relation of type Delegation is added that's it's type really is.
    b. Passes
4. public void componentTest3()
    a. Testing that when a Relation of type Containment is added that's it's type really is.

b. Passes

Class Test

1. public void testName()
    a. Testing with getter if String classname can be added for DesignClass and DesignInterface
    b. Passes
2. public void testPosition()
    a. Testing with with getter if Point position can be added for DesignClass and DesignInterface
    b. Passes
3. public void testSter()
    a. Testing with getter if String stereotype can be added for DesignClass and DesignInterface
    b. Passes
4. public void testMethodsInterface()
    a. Testing with getter if String method can be added for DesignClass and DesignInterface
    b. Passes
5. public void testInsvar()
    a. Testing with getter if String instanceVariables can be added for DesignClass
    b. Passes

ThemeCreator Test

1. public void setOutlineColor()
    a. Testing to see if setOutlineColor sets the theme's outline color
    b. Passes
2. public void setBackgroundColor()
    a. Testing to see if setBackgroundColor sets the theme's background color
    b. Passes
3. public void setBoxFillColor()
    a. Testing to see if setBoxFillColor sets the theme's box fill color
    b. Passes
4. public void setFontColor()
    a. Testing to see if setFontColor sets the theme's font color
    b. Passes
5. public void setArrowColor()
    a. Testing to see if setArrowColor sets the theme's arrow color
    b. Passes

SpyTest Test - Uses Spy Test to test the Diagram class

1. public void testAddRelationToList()
    a. Tests if when addRelationList is called the listeners are actually updated and the relation is added to the list of relationships
    b. Passes
2. public void testDeleteRelationship()
    a. Tests if when deleteRelationship is called the listeners are actually updated and the relationship is deleted from the list of relationships
    b. Passes
3. public void testAddClass()
    a. Tests if when addClass is called the listeners are actually updated and the class is added to the list of classes
    b. Passes
4. public void testDeleteClass()
    a. Tests if when deleteClass is called the listeners are actually updated and the relationship is deleted from the list of classes
    b. Passes

DiagramView overview

1. We coded the DiagramView in small chunks and used the "gradle run" command very often to check how DiagramView was working
2. Used Zoom's share screen feature to show each other what the DiagramView class looked like in the editor after making local changes
3. By doing those two things we know that our diagram editor successfully:
    a. Displays the top menu bar with "file" → "open", "save as", and "export" options
    b. Displays the menu option "view" with "change theme option", then "dark mode", light mode" and 'custom" options
        i. When user clicks "dark mode" the theme changes to dark mode
        ii. When user clicks "light mode" the theme changes to light mode
        iii. When user clicks "custom" they have the option to change the background color, class box outline color, class box fill color, font color and arrow color
    c. Displays the right-hand side toolbar with "add class", "add interface"
        i. When a user selects "add class" or "add design class" a pop up window appears where they can enter the name of the class and then the class box appears
    d. The user can drag the class/interface anywhere in the window
    e. The user can right click on a class or interface and select options to add a method, instance variable, relationship or delete a class box all together