Inventory Management Junit documentation

Package: inventory.management

Test Class: FileOperationsTest

The **FileOperationsTest** class is a JUnit test class designed to validate the functionality of the **FileOperations** class. It contains test methods to ensure that the methods within **FileOperations** perform as expected in various scenarios.

Test Methods:

1. **public void saveDataAndLoadData()**
   This test method verifies the correct functionality of saving and loading data using the **FileOperations** class.
   - **Arrange:**
     - Creates a list of test data using the **createTestData()** method.
   - **Act:**
     - Calls **FileOperations.saveData()** to save the test data to a test file.
     - Calls **FileOperations.loadData()** to load the data from the test file.
   - **Assert:**
     - Compares the size of the original test data list with the size of the loaded data list.
     - Iterates through each product in the test data and asserts that it is equal to the corresponding product in the loaded data.
2. **public void loadDataFromNonexistentFile()**
   This test method checks the behavior of the **FileOperations.loadData()** method when attempting to load data from a nonexistent file.
   - **Act:**
     - Calls **FileOperations.loadData()** with the path of a nonexistent file.
   - **Assert:**
     - Verifies that the returned list is not null.
     - Verifies that the returned list is empty.

1. Cleanup Method

- **public void cleanup()**
  This method is annotated with **@After** and is responsible for cleaning up resources after each test execution. It deletes the test file created during the test process.

2. Supporting Methods

- **private List<Product> createTestData()**
  This method creates and returns a list of **Product** objects for use in the test methods.
- **private void assertProductEquals(Product expected, Product actual)**
  This method asserts that the properties of two **Product** objects are equal.
- **private void deleteTestFile()**
  This method deletes the test file created during testing if it exists.

Test Class: ProductTest

The **ProductTest** class is a JUnit test class designed to validate the functionality of the **Product** class within the "inventory.management" package. It includes test methods that assess the correct initialization and behavior of the **Product** class, focusing on its constructor and various setter methods.

Test Methods:

1. **testProductInitialization:**
    - **Purpose:** Validates the proper initialization of a **Product** instance with the provided parameters.
    - **Arrange:**
        - Create a **Product** instance with predefined values.
    - **Act:**
        - Check if the getters return the expected values.
    - **Assert:**
        - Ensure that the getters return the values set during initialization.
2. **testSetName:**
    - **Purpose:** Verifies the functionality of the **setName** method in updating the name of a **Product**.
    - **Arrange:**
        - Create a **Product** instance with an initial name.
    - **Act:**
        - Set a new name using the **setName** method.
    - **Assert:**
        - Confirm if the name has been successfully updated.
3. **testSetSku:**
    - **Purpose:** Validates the functionality of the **setSku** method in updating the SKU of a **Product**.
    - **Arrange:**
        - Create a **Product** instance with an initial SKU.
    - **Act:**
        - Set a new SKU using the **setSku** method.
    - **Assert:**
        - Ensure that the SKU has been successfully updated.
4. **testSetCategory:**
    - **Purpose:** Ensures the proper execution of the **setCategory** method in updating the category of a **Product**.
    - **Arrange:**
        - Create a **Product** instance with an initial category.
    - **Act:**
        - Set a new category using the **setCategory** method.
    - **Assert:**
        - Validate if the category has been successfully updated.
5. **testSetQuantity:**
    - **Purpose:** Validates the functionality of the **setQuantity** method in updating the quantity of a **Product**.
    - **Arrange:**
        - Create a **Product** instance with an initial quantity.
    - **Act:**
        - Set a new quantity using the **setQuantity** method.
    - **Assert:**
        - Verify if the quantity has been successfully updated.

Test Class: ProductTableModelTest

Test Methods:

1. **testAddProduct:**
   - **Purpose:** Verifies the proper addition of a product to the **ProductTableModel**.
   - **Arrange:**
     - Create an instance of **ProductTableModel**.
     - Create a **Product** instance for testing.
   - **Act:**
     - Call **addProduct** method on the **ProductTableModel**.
   - **Assert:**
     - Ensure the table row count increases by 1.
     - Confirm the added product matches the retrieved product.
2. **testRemoveProduct:**
   - **Purpose:** Validates the correct removal of a product from the **ProductTableModel**.
   - **Arrange:**
     - Create an instance of **ProductTableModel**.
     - Add a **Product** instance for testing.
   - **Act:**
     - Call **removeProduct** method on the **ProductTableModel**.
   - **Assert:**
     - Ensure the table row count decreases by 1.
3. **testGetProduct:**
   - **Purpose:** Verifies the accurate retrieval of a product from the **ProductTableModel**.
   - **Arrange:**
     - Create an instance of **ProductTableModel**.
     - Add a **Product** instance for testing.
   - **Act:**
     - Call **getProduct** method on the **ProductTableModel**.
   - **Assert:**
     - Confirm the retrieved product matches the added product.
4. **testGetAllProducts:**
   - **Purpose:** Validates the proper retrieval of all products from the **ProductTableModel**.
   - **Arrange:**
     - Create an instance of **ProductTableModel**.
     - Add multiple **Product** instances for testing.
   - **Act:**
     - Call **getAllProducts** method on the **ProductTableModel**.
   - **Assert:**
     - Ensure the list size matches the number of added products.
     - Confirm each product in the list matches the corresponding added product.
5. **testIsCellEditable:**
   - **Purpose:** Ensures that cells are not editable directly from the table view.
   - **Arrange:**
     - Create an instance of **ProductTableModel**.
   - **Act & Assert:**
     - Verify that **isCellEditable** returns false for any cell.