**Different coat application Support and maintenance information**

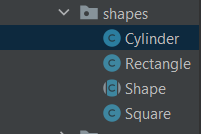
This documentation describes the different coat application whose purpose is to calculate paint coverage for the customers of different coat based on the customer type (General customer/trade customer). A general customer is required to input additional information for calculating the paint coverage including room shape (Cylinder, Square or rectangle currently), and room width and height so that the surface area can be calculated for the customer before calculating the paint coverage as it is assumed that the general customer does not have the knowledge required to accurately make this surface area calculation which returns the paint required in litres, however trade Customers only have to enter the surface area as they are assumed to know how to make this calculation.

Graphical user interface, text

Description automatically generatedThe application name for this project is Different coat and the main package name is followed by org. differentCoat. These packages are shapes, customers, demos, and support figure 1 below shows the file structure of the project.

Figure

Each of the packages contains their relevant classes this helps to organise code in a structures and easily understood manner making it more maintainable when adding code in futures updates. Each package and its classes are explained in more detail below.

**Text

Description automatically generatedText

Description automatically generatedshapes**: This package contains four classes including Shape, Rectangle, Cylinder and square. The Shape class is an abstract class which defines basic properties and functions like the height and width fields a several getters and a get Perimeter function which will be used in calculating the surface Area of the shape later in the calculation. Shape, Rectangle and Cylinder add further implementation through inheriting from shape, Cylinder has the biggest differences as it overrides the functionality of get Perimeter within the from the super class shape to use a different perimeter calculation.

Figure 3

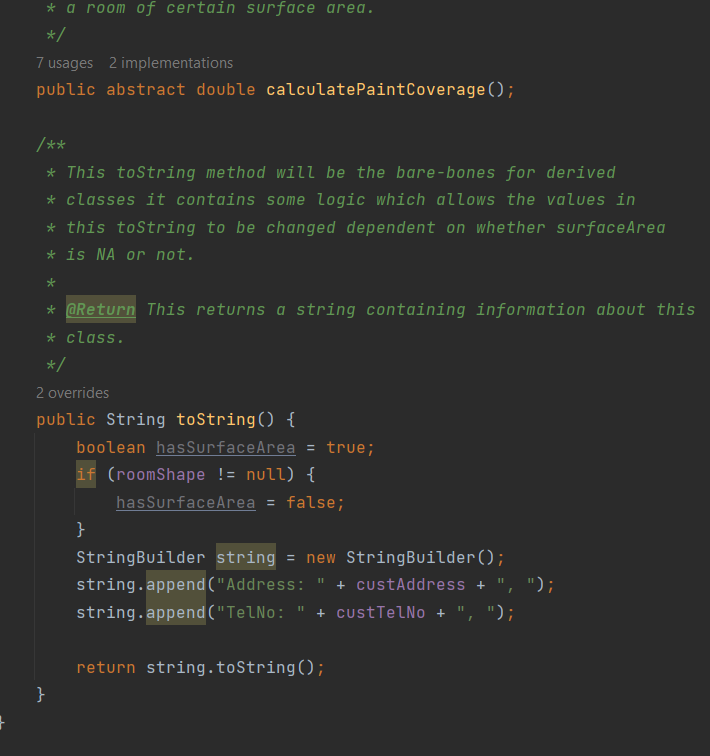
Figure 2

The left screenshot shows an image of the Shape class is the super class which all the shape classes extend from like in the example in the right screenshot where Cylinder extends from Shape.

Graphical user interface, text, application, chat or text message

Description automatically generated**Customers:** Within the customers package there are three classes customer, Telecaster and General Customer. The customer is an abstract class which defines basic functionality and fields expected of classes that inherit from this which currently are General Public and Trade Customer.

**Text

Description automatically generated**

Text

Description automatically generatedGeneral Public class extends from the customer class and provides further implementation. A general customer is a normal customer with assumed no previous experience in calculating paint coverage therefore when we create this customer with additional information to other types of customers like Trade Customer such as height, width and room shape of the room they want to paint to calculate surface are for the room and then calculate the paint coverage in litres.

Text

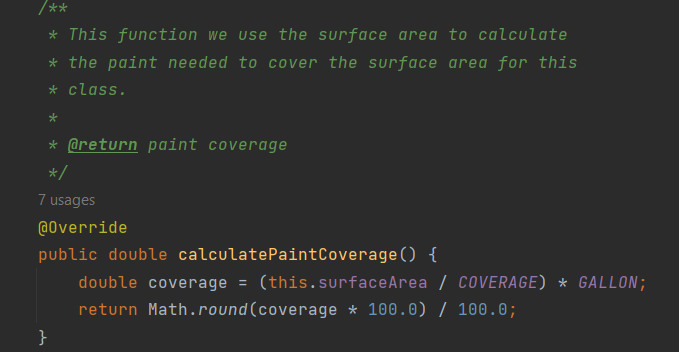
Description automatically generatedBelow shows the calculateSurfaceArea function which used the roomShape height and width to calculate the surface area of a room.

Text

Description automatically generatedThis is the calculatePaintCoverage function is used the surface area created in the function above to calculate how much paint in liters in needed to cover the entirety of the room defined by the GeneralCustomer

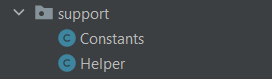
Trade Customer is the other class which extends from customer this type of customer is assumed to know how to make an accurate measurement of surface area for a room that hey are painting therefore this customer we do not take room shape, height, and width to calculate the surface area rather we let the customer input the surface are in the constructor.

Below is the calculatePaintCoverage function for the TradeCustomer:



**Support**

Within the support package there are two classes Helper and Constants the purpose of this class is the aid with the general functionality of the application.

****

**Helper**

The helper class reads and writes customer information relating to the paint coverage to the paint\_coverage\_information.csv file, Within this class there is two public functions; add Item which takes a customer object and adds its values to the pain\_coverage\_information.csv and read file which reads everything in the paint\_coverage\_information.csv file and puts it into an Array List of Customers which is then returned.

Helper fields

**Text

Description automatically generated**

**Helper add to file function:**

**Text

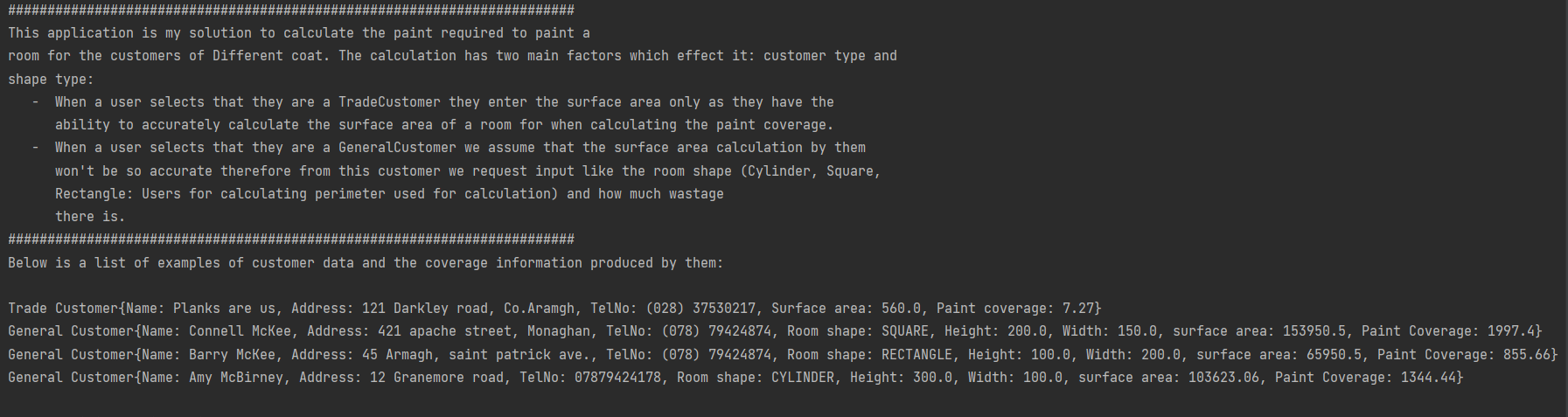
Description automatically generatedGetFileAsArrayList function:**

**Text

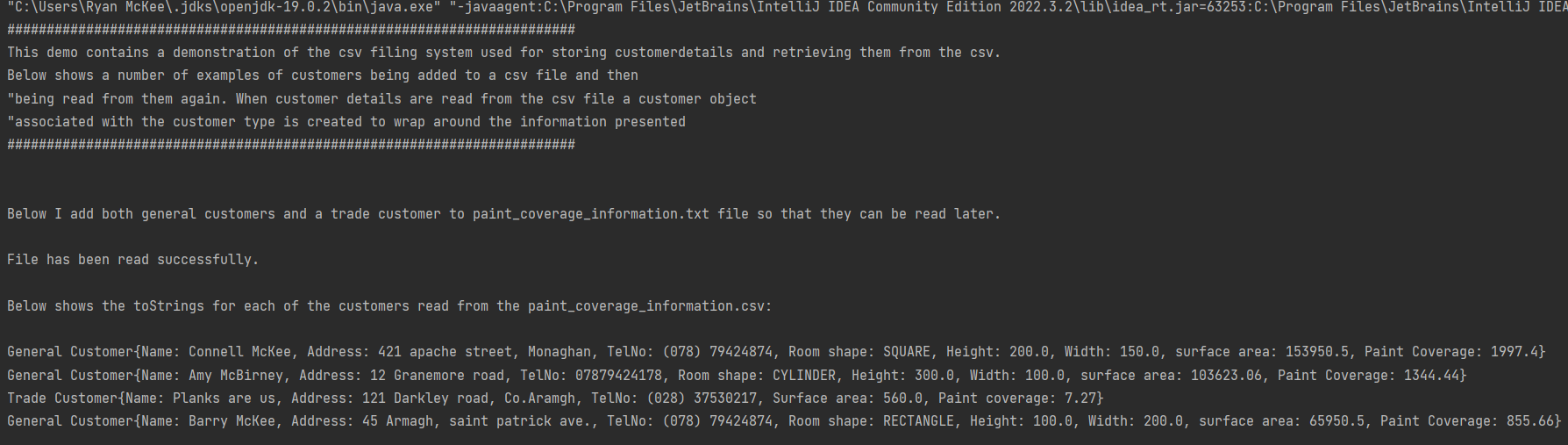
Description automatically generated**

**Demos**

**Demo 1**: To showcase the functionality of the application, Demo classes are used. In Demo 1, a general customer is created and the get coverage function is shown within the toString function displayed in the console. As well an example of the TradeCustomer class toString.

****

**Demo 2** shows the further functionality of the application including the file storage system using the csv file. In this demo 2 customer objects a trade Customer and General Customer are written to the paint\_coverage\_information.csv file then they are read and converted back to customer objects and the toString is finally printed.

****

Self-Evaluation

Generally, I believe there are positives and weaknesses to my applications. My application I believe is well commented and easy to read it also fulfils the requirements set by different coat it has a lot of automation is a very rigid system. I believe that my system uses the OOP principles very well and there is little redundancy in the code because of the use of parent child classes where the parent classes would have common functionality of the child classes implemented in them. I have also annotated everything very well like overridden functions for example and thoroughly tested the system using unit tests.

However, I do believe that there is some work that could be further done. I believe that there is more input validation which could but added for things like the customer telephone numbers and the width and height entry. I also believe I could make my helper class better by creating a parser to separate out certain functionality for example converting the string values from the csv file into a customer object. In the future I believe a better UI could be created for different coats this would make it simpler to interact with the applications and finally I believe that I spent longer on the application than I would have otherwise if I had of better done better design within my UML.

In conclusion I believe my application fulfils it purpose for the task set by different coat however there are improvements that can be made in the future.