Scenario:

You need to build part of a game named Age of Villagers (AoV). The game is about people from different parts of the world trying to build their villages. The game has a lot of features, of which you need to focus only on the village creation part. A village can have several types of objects like house, tree and water source. Your main task is to create the house, tree or water sources by adding different types of simple shapes.

Procedure:

Composite pattern is used where one needs to treat a group of objects in a similar way as a single object. Composite pattern build objects in term of a tree structure to represent partial as well as whole hierarchy.

Let our village in the game is made of houses, trees, water-wells, and roads. We will make this using simple shapes like circle, plane, rectangle, triangle.

We will make a **Shape.java** class which will take name of an object ,color and its dimension. We will keep an **ArrayList** for keeping track of the simple shapes the object is made up of. The object can add any combinations of simple objects to make its own structure.We have a printf() method which will show the output of what village is consists of.

Demo.java class is our main class. Here we will add different objects that made the village of the game.

To make a **house** we will use a cube and pyramid shape. Again Cube and Pyramid shape is made using plane and triangle shape.

Snippet:

```
cube.add(plane);
pyramid.add(triangle);
shape house = new shape("House","Blue-Gray","3D");
house.add(cube);
house.add(pyramid);
```

To make a **tree** structure we will use circle as leaves and line shape as the stem.

Snippet:

```
shape tree = new shape("Tree","Green","3D");
tree.add(line);
tree.add(circle);
```

To make a **water-well** structure we can use circle shape only.

Snippet:

```
shape waterSource = new shape("Well","Tranparent","2D");
waterSource.add(circle);
```

Now, to make **roads** we can use rectangle shape as the road and line shape as the divider.

Snippet:

```
shape roads = new shape("roads","mixed","2D");
roads.add(rectangle);
roads.add(line);
```

Now, the **village** can be made of any number of these objects. We will take each of the objects once for demonstration.

Snippet:

```
shape village = new shape("Village","mixed","3D");
village.add(house);
village.add(tree);
village.add(waterSource);
village.add(roads);
```

We will use **print()** function to show the output.

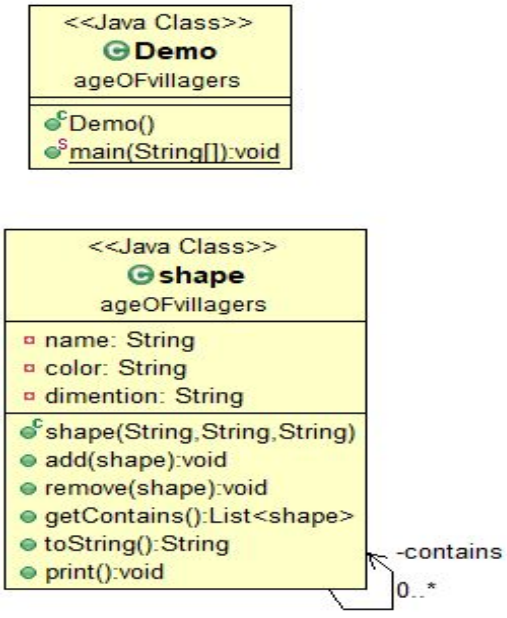
Snippet:

```
village.print();
house.print();
tree.print();
roads.print();
waterSource.print();
```

```
Village Consists of:
 Name: House, color: Blue-Gray, property:3D
 Name: Tree, color: Green, property:3D
 Name: Well, color: Tranparent, property:2D
 Name: roads, color: mixed, property:2D
-----------------------------------
House Consists of:
 Name: Cube, color: Blue, property:3D
 Name: Pyramid, color: Gray, property:3D
-----------------------------------
Tree Consists of:
 Name: Line, color: Brown, property:1D
 Name: Circle, color: Green, property:2D
-----------------------------------
Well Consists of:
 Name: Circle, color: Green, property:2D
-----------------------------------
roads Consists of:
 Name: Rectangle, color: Blue, property:2D
 Name: Line, color: Brown, property:1D
```

Fig: Output

UML Diagram:



Github Link : https://github.com/NMLami/Design_Pattern_Assignment .