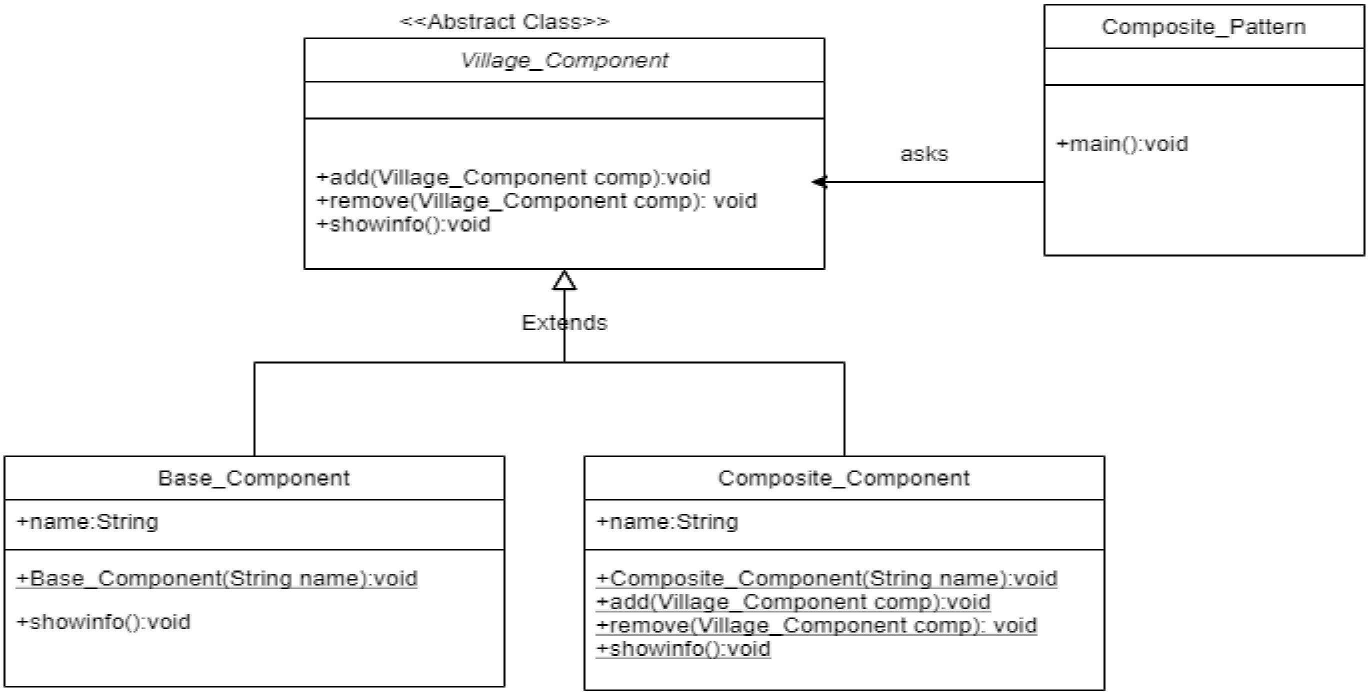
Composite Pattern

Imtiaj Ahmed Chowdhury
ID: 160041048
Section: B

We have two types of components, one is ***Base_Component*** that consists of smaller objects like roof, foundation, leaves, root etc and another is ***Composite_Component*** which consists of the Base_Components. Both types of components inherit the same abstract class ***Village_Component***. Base_Component objects can be added and removed to and from the Composite_Component objects through the *add()* and *remove()* methods respectively. This actually implements the composite pattern where complex objects are made up of several simpler objects.

When the *showinfo()* method is called on a Composite_Component object, its traversed recursively and the Base_Component objects' *showinfo()* methods are called. Finally, a Composite_Component object can be made up of other Composite_Component objects which is shown by adding *house*, *tree* and *water_source* to the village objects. Following is the class diagram of the project:


<<Abstract Class>>
Village_Component
+add(Village_Component comp):void
+remove(Village_Component comp): void
+showinfo():void
Composite_Pattern
+main():void
asks
Extends
Base_Component
+name:String
+Base_Component(String name):void
+showinfo():void
Composite_Component
+name:String
+Composite_Component(String name):void
+add(Village_Component comp):void
+remove(Village_Component comp): void
+showinfo():void