Composite is a structural design pattern that allows composing objects into a tree-like structure and work with the it as if it was a singular object.

It is one of the Structural design pattern.

It is used when we have to represent a part-whole hierarchy.

Composite Pattern consists of following objects.

*Base Component* - Base component is the interface for all objects in the composition, client program uses base component to work with the objects in the composition. It can be an interface or an abstract class with some methods common to all the objects.

*Leaf* - Defines the behaviour for the elements in the composition. It is the building block for the composition and implements base component. It doesn’t have references to other Components.

*Composite* - It consists of leaf elements and implements the operations in base component.

Example : File system

*Base Component*

public interface FileSystem {  
 public void ls() ;  
  
}

*Leaf*

public class File implements FileSystem{  
  
 String fileName ;  
  
 public File(String fileName ){  
 this.fileName = fileName ;  
 }  
  
 @Override  
 public void ls() {  
 System.*out*.println("File name : " + fileName );  
 }  
  
}

*Composite*

public class Directory implements FileSystem{  
  
 String directoryName ;  
  
 List<FileSystem> fileSystemList ;  
  
 public Directory(String directoryName) {  
 this.directoryName = directoryName;  
 fileSystemList = new ArrayList<>() ;  
 }  
  
 public void add(FileSystem fileSystem){  
 fileSystemList.add(fileSystem) ;  
 }  
  
 @Override  
 public void ls() {  
 System.*out*.println("Directory name : " + directoryName );  
  
 for( FileSystem fileSystem : fileSystemList ){  
 fileSystem.ls();  
 }  
 }  
  
}

public class CompositeDesignPattern {  
  
 public static void main(String[] args) {  
  
 Directory movieRootDirectory = new Directory("Movies") ;  
  
 FileSystem subtitleFile = new File("english\_subtitles.srt") ;  
 movieRootDirectory.add(subtitleFile);  
  
 Directory comedyFilmDirectory = new Directory("Comedy") ;  
  
 FileSystem ben10Film = new File("Ben 10") ;  
  
 comedyFilmDirectory.add(ben10Film);  
  
 Directory actionFilmDirectory = new Directory("Action") ;  
  
 FileSystem terminatorFilm = new File("Terminator") ;  
  
 actionFilmDirectory.add(terminatorFilm);  
  
 movieRootDirectory.add(comedyFilmDirectory);  
  
 movieRootDirectory.add(actionFilmDirectory);  
  
 movieRootDirectory.ls();  
  
 }  
}

Output :

Directory name : Movies

File name : english\_subtitles.srt

Directory name : Comedy

File name : Ben 10

Directory name : Action

File name : Terminator

Process finished with exit code 0