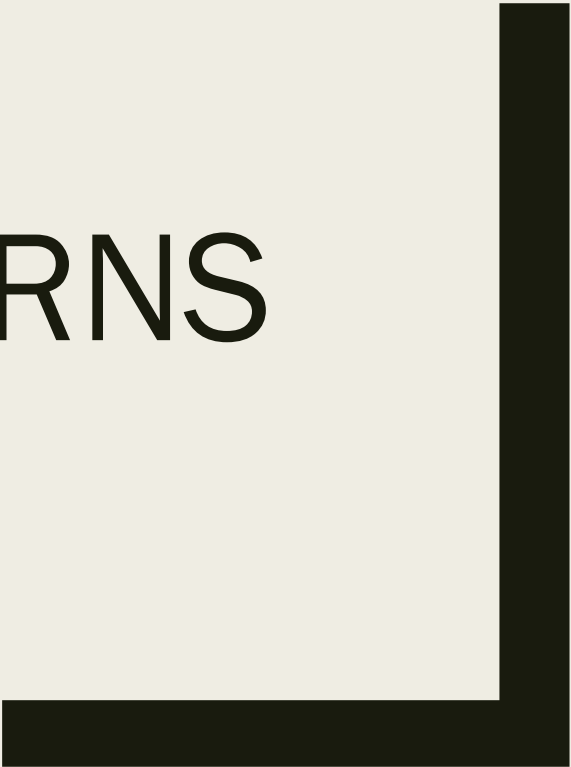




DESIGN PATTERNS

Adapter Pattern
By Kishore Muralitharan



What are they?

- Design patterns are best practices that are used by object oriented developers
- They are solutions to general problems that occur during development
- Developed by trial and error by numerous developers over an extended period
- 4 Major Classifications:
 - *Creational Patterns*
 - *Structural Patterns*
 - *Behaviour Patterns*
 - *J2EE Patterns*

Adapter Pattern

- Classified as a Structural Pattern because it combines the abilities of two separate interfaces
- Adapter Pattern acts a bridge, allowing communication between two incompatible interfaces
- A single class which is used to join the functionality of separate or incompatible interfaces
- An Real world example:
 - *Travel Adapter, Allows the use of U.S Standard devices in other regions like Europe*
 - *SD Card Reader, Acts as a bridge between the SD card and the laptop*

Java Implementati on

- Audio Player uses the Media Adapter which uses the Adv. Media Player
- Both Classes implement Media Player Interface
- Depending on the file type the adapter can be used to use Adv. Media Player
- If not needed AudioPlayer can play the given file

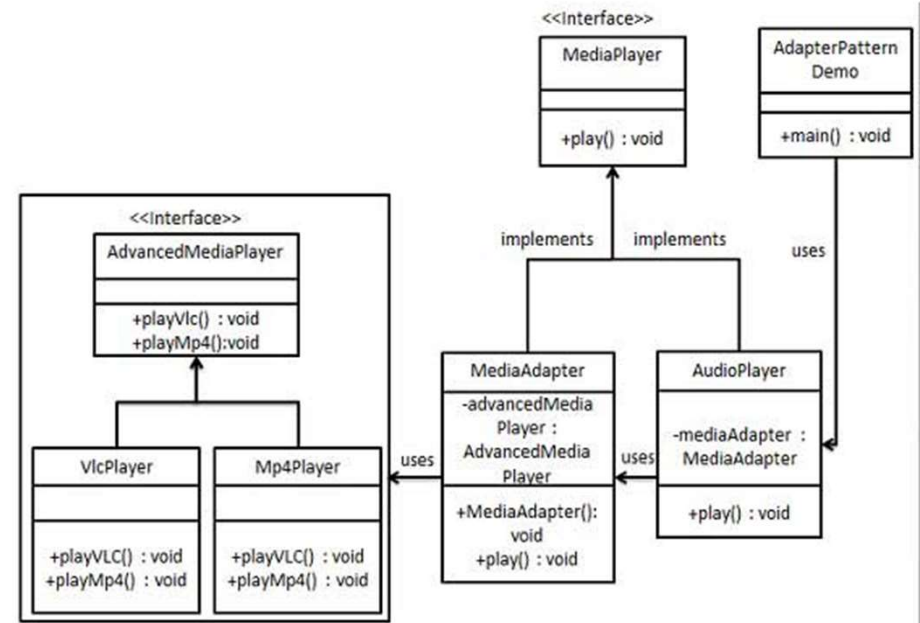


Fig 1. UML Diagram of an Adapter Pattern

Interfaces

MediaPlayer.java

```
public interface MediaPlayer {  
    public void play(String audioType, String fileName);  
}
```

AdvancedMediaPlayer.java

```
public interface AdvancedMediaPlayer {  
    public void playVlc(String fileName);  
    public void playMp4(String fileName);  
}
```

Adv. Media Player Classes

VlcPlayer.java

```
public class VlcPlayer implements AdvancedMediaPlayer{
    @Override
    public void playVlc(String fileName) {
        System.out.println("Playing vlc file. Name: "+ fileName);
    }

    @Override
    public void playMp4(String fileName) {
        //do nothing
    }
}
```

Mp4Player.java

```
public class Mp4Player implements AdvancedMediaPlayer{

    @Override
    public void playVlc(String fileName) {
        //do nothing
    }

    @Override
    public void playMp4(String fileName) {
        System.out.println("Playing mp4 file. Name: "+ fileName);
    }
}
```

Adapter Class

```
MediaAdapter.java

public class MediaAdapter implements MediaPlayer {

    AdvancedMediaPlayer advancedMusicPlayer;

    public MediaAdapter(String audioType){

        if(audioType.equalsIgnoreCase("vlc") ){
            advancedMusicPlayer = new VlcPlayer();

        }else if (audioType.equalsIgnoreCase("mp4")){
            advancedMusicPlayer = new Mp4Player();
        }
    }

    @Override
    public void play(String audioType, String fileName) {

        if(audioType.equalsIgnoreCase("vlc")){
            advancedMusicPlayer.playVlc(fileName);
        }
        else if(audioType.equalsIgnoreCase("mp4")){
            advancedMusicPlayer.playMp4(fileName);
        }
    }
}
```

Primary Audio Player

AudioPlayer.java

```
public class AudioPlayer implements MediaPlayer {
    MediaAdapter mediaAdapter;

    @Override
    public void play(String audioType, String fileName) {

        //inbuilt support to play mp3 music files
        if(audioType.equalsIgnoreCase("mp3")){
            System.out.println("Playing mp3 file. Name: " + fileName);
        }

        //mediaAdapter is providing support to play other file formats
        else if(audioType.equalsIgnoreCase("vlc") || audioType.equalsIgnoreCase("mp4")){
            mediaAdapter = new MediaAdapter(audioType);
            mediaAdapter.play(audioType, fileName);
        }

        else{
            System.out.println("Invalid media. " + audioType + " format not supported");
        }
    }
}
```

AdapterPatternDemo.java

```
public class AdapterPatternDemo {
    public static void main(String[] args) {
        AudioPlayer audioPlayer = new AudioPlayer();

        audioPlayer.play("mp3", "beyond the horizon.mp3");
        audioPlayer.play("mp4", "alone.mp4");
        audioPlayer.play("vlc", "far far away.vlc");
        audioPlayer.play("avi", "mind me.avi");
    }
}
```


References

- *Design Patterns - Adapter Pattern*. Tutorials Point. (n.d.). Retrieved February 25, 2022, from https://www.tutorialspoint.com/design_pattern/adapter_pattern.htm