



[Home](#) / [Design Patterns](#) / [Facade](#) / [Java](#)



Facade in Java

Facade is a structural design pattern that provides a simplified (but limited) interface to a complex system of classes, library or framework.

While Facade decreases the overall complexity of the application, it also helps to move unwanted dependencies to one place.

[Learn more about Facade →](#)

Navigation

- [Intro](#)
- [Simple interface for a complex video conversion library](#)
 - [some_complex_media_library](#)
 - [VideoFile](#)
 - [Codec](#)
 - [MPEG4CompressionCodec](#)
 - [OggCompressionCodec](#)
 - [CodecFactory](#)
 - [BitrateReader](#)
 - [AudioMixer](#)
 - [facade](#)
 - [VideoConversionFacade](#)
 - [Demo](#)
 - [OutputDemo](#)

Complexity: ★☆☆



Usage examples: The Facade pattern is commonly used in apps written in Java. It's especially handy when working with complex libraries and APIs.

Here are some Facade examples in core Java libs:

- `javax.faces.context.FacesContext` uses `LifeCycle`, `ViewHandler`, `NavigationHandler` classes under the hood, but most clients aren't aware of that.
- `javax.faces.context.ExternalContext` uses `ServletContext`, `HttpSession`, `HttpServletRequest`, `HttpServletResponse` and others inside.

Identification: Facade can be recognized in a class that has a simple interface, but delegates most of the work to other classes. Usually, facades manage the full life cycle of objects they use.

Simple interface for a complex video conversion library

In this example, the Facade simplifies communication with a complex video conversion framework.

The Facade provides a single class with a single method that handles all the complexity of configuring the right classes of the framework and retrieving the result in a correct format.

📁 **some_complex_media_library: Complex video conversion library**

📄 **some_complex_media_library/VideoFile.java**

```
package refactoring_guru.facade.example.some_complex_media_library;

public class VideoFile {
    private String name;
    private String codecType;

    public VideoFile(String name) {
        this.name = name;
        this.codecType = name.substring(name.indexOf(".") + 1);
    }

    public String getCodecType() {
        return codecType;
    }
}
```



```
public String getName() {  
    return name;  
}  
}
```

some_complex_media_library/Codec.java

```
package refactoring_guru.facade.example.some_complex_media_library;  
  
public interface Codec {  
}
```

some_complex_media_library/MPEG4CompressionCodec.java

```
package refactoring_guru.facade.example.some_complex_media_library;  
  
public class MPEG4CompressionCodec implements Codec {  
    public String type = "mp4";  
}
```

some_complex_media_library/OggCompressionCodec.java

```
package refactoring_guru.facade.example.some_complex_media_library;  
  
public class OggCompressionCodec implements Codec {  
    public String type = "ogg";  
}
```

some_complex_media_library/CodecFactory.java

```
package refactoring_guru.facade.example.some_complex_media_library;  
  
public class CodecFactory {  
    public static Codec extract(VideoFile file) {  
        String type = file.getCodecType();  
        if (type.equals("mp4")) {
```



```
    }  
    else {  
        System.out.println("CodecFactory: extracting ogg audio...");  
        return new OggCompressionCodec();  
    }  
}  
}
```

some_complex_media_library/BitrateReader.java

```
package refactoring_guru.facade.example.some_complex_media_library;  
  
public class BitrateReader {  
    public static VideoFile read(VideoFile file, Codec codec) {  
        System.out.println("BitrateReader: reading file...");  
        return file;  
    }  
  
    public static VideoFile convert(VideoFile buffer, Codec codec) {  
        System.out.println("BitrateReader: writing file...");  
        return buffer;  
    }  
}
```

some_complex_media_library/AudioMixer.java

```
package refactoring_guru.facade.example.some_complex_media_library;  
  
import java.io.File;  
  
public class AudioMixer {  
    public File fix(VideoFile result){  
        System.out.println("AudioMixer: fixing audio...");  
        return new File("tmp");  
    }  
}
```



```
package refactoring_guru.facade.example.facade;

import refactoring_guru.facade.example.some_complex_media_library.*;

import java.io.File;

public class VideoConversionFacade {
    public File convertVideo(String fileName, String format) {
        System.out.println("VideoConversionFacade: conversion started.");
        VideoFile file = new VideoFile(fileName);
        Codec sourceCodec = CodecFactory.extract(file);
        Codec destinationCodec;
        if (format.equals("mp4")) {
            destinationCodec = new MPEG4CompressionCodec();
        } else {
            destinationCodec = new OggCompressionCodec();
        }
        VideoFile buffer = BitrateReader.read(file, sourceCodec);
        VideoFile intermediateResult = BitrateReader.convert(buffer, destinationCodec);
        File result = (new AudioMixer()).fix(intermediateResult);
        System.out.println("VideoConversionFacade: conversion completed.");
        return result;
    }
}
```

Demo.java: Client code

```
package refactoring_guru.facade.example;

import refactoring_guru.facade.example.facade.VideoConversionFacade;

import java.io.File;

public class Demo {
    public static void main(String[] args) {
        VideoConversionFacade converter = new VideoConversionFacade();
        File mp4Video = converter.convertVideo("youtubevideo.ogg", "mp4");
        // ...
    }
}
```



SPRING SALE 

```
VideoConversionFacade: conversion started.  
CodecFactory: extracting ogg audio...  
BitrateReader: reading file...  
BitrateReader: writing file...  
AudioMixer: fixing audio...  
VideoConversionFacade: conversion completed.
```

[Home](#)



[Refactoring](#)



[Design Patterns](#)




[Premium Content](#)

[Forum](#)


[Contact us](#)

© 2014-2023 Refactoring.Guru. All rights reserved.

 Illustrations by Dmitry Zhart

Ukrainian office:

 FOP Olga Skobeleva


 Abolmasova 7

Kyiv, Ukraine, 02002


 Email: support@refactoring.guru

Spanish office:

 Oleksandr Shvets

 Avda Pamplona 63, 4b

Pamplona, Spain, 31010

 Email: spain@refactoring.guru

[Terms & Conditions](#)

[Privacy Policy](#)

[Content Usage Policy](#)

[About us](#)



