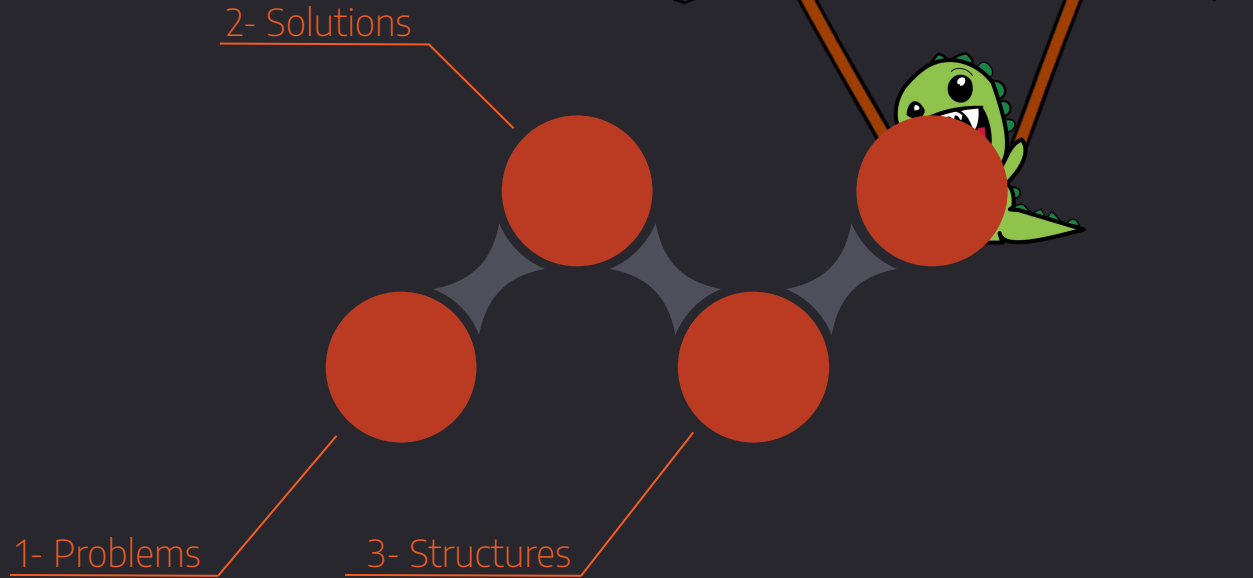


Design Pattern

Facade



Summary





Ok.



Sommaire

Problèmes

Nous sommes nous voiler la face ?

Solutions

Un nouveau Jurassic park ?

(je signe direct)

Structure

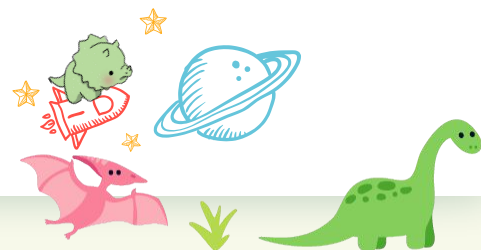
Comment c'est qu'on fait ?

Implementation

Ah!! Alors c'est comme ca qu'on fait bien ?

Pour et contre

Pour ou contre le retour des dinos sur Terre ?



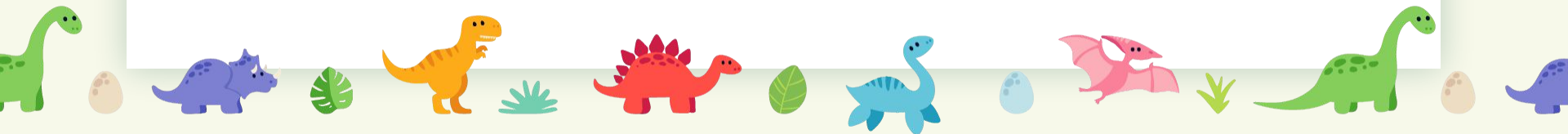
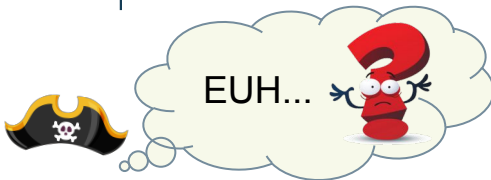
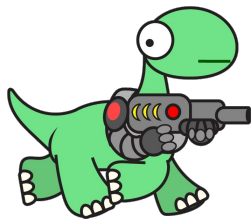


1. Problèmes



Problèmes

- Que se passe-t-il si vous avez plein de bibliothèques ?
- Que se passe-t-il si vous avez plein de classes ?

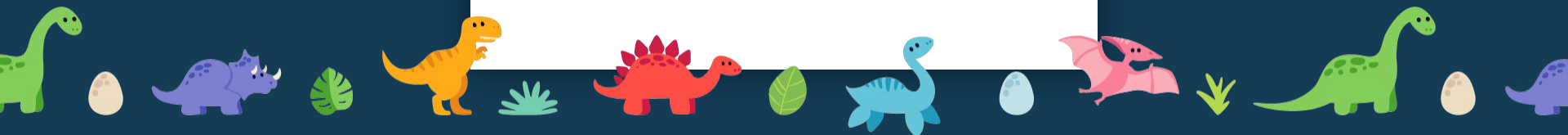


2. Solution





La façade est un modèle de
conception structurelle qui
fournit une interface
simplifiée à une
bibliothèque, un framework
ou tout autre ensemble
complexe de classes.

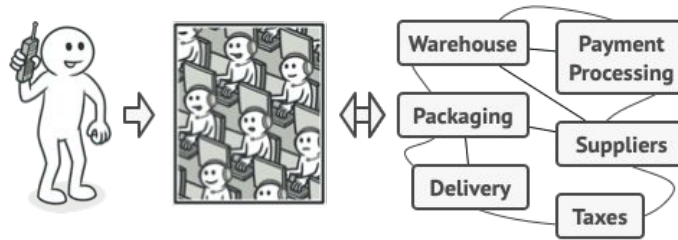




Solution

La façade fournit une interface simple à un sous système complexe

Elle limite l'utilisation des frameworks aux besoins du client

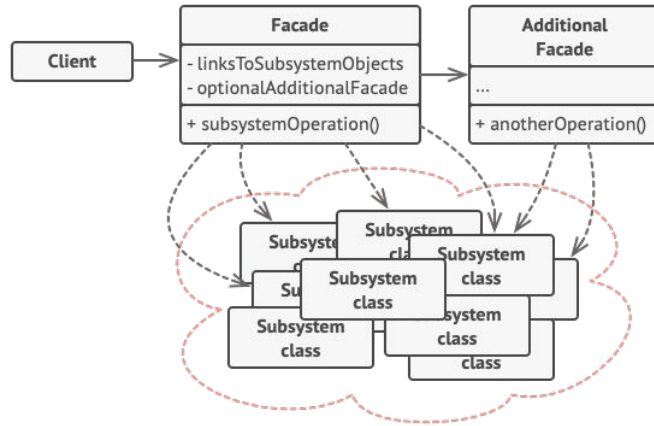


The background is a dark blue field filled with various colorful, stylized dinosaur illustrations. These include a green long-necked dinosaur, a purple Triceratops, an orange T-Rex, a red Stegosaurus, a blue long-necked dinosaur, a pink Pterosaur, and a green long-necked dinosaur. There are also several light blue eggs and green plants scattered throughout. A large, white rectangular box is centered on the page, containing the text "3. Structures".

3. Structures



Structure



1- Accéder facilement à une partie particulière des fonctionnalités du sous-système.

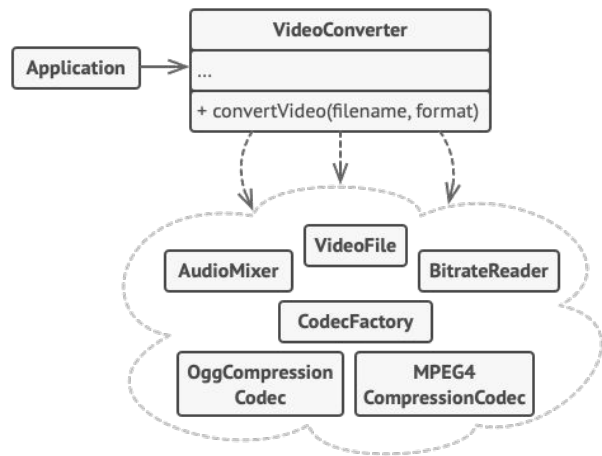
2- Créer une classe Façade supplémentaire pour éviter une Façade principale complexe

3- Les classes du sous-système ne connaissent pas l'existence de la façade.

4- Le client utilise la façade au lieu d'appeler directement les objets du sous-système



Pseudocode



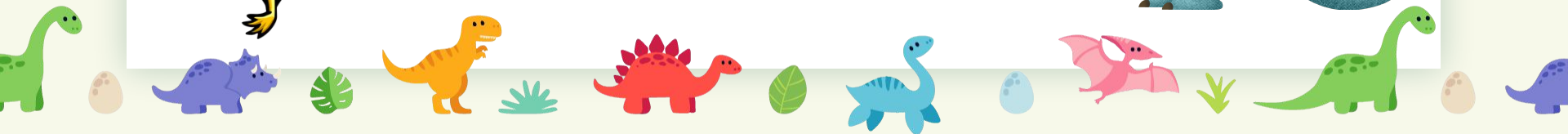


Pseudocode



```

// These are some of the classes of a complex 3rd-party video
// conversion framework. We don't control that code, therefore
// can't simplify it.
class VideoFile
// ...
class OggCompressionCodec
// ...
class MPEG4CompressionCodec
// ...
class CodecFactory
// ...
class BitrateReader
// ...
class AudioMixer
// ...
```

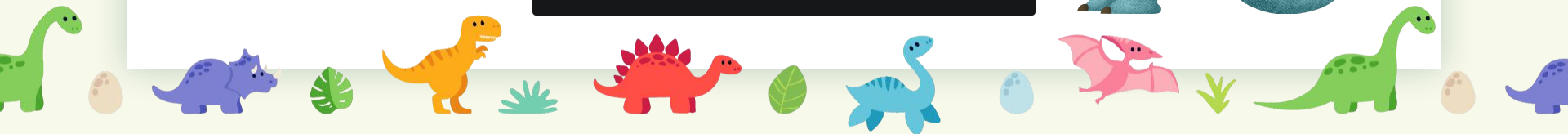




Pseudocode


```
// We create a facade class to hide the framework's complexity
// behind a simple interface. It's a trade-off between
// functionality and simplicity.
class VideoConverter is
    method convert(filename, format):File is
        file = new VideoFile(filename)
        sourceCodec = new CodecFactory.extract(file)
        if (format == "mp4")
            destinationCodec = new MPEG4CompressionCodec()
        else
            destinationCodec = new OggCompressionCodec()
        buffer = BitrateReader.read(filename, sourceCodec)
        result = BitrateReader.convert(buffer,
            destinationCodec)
        result = (new AudioMixer()).fix(result)
        return new File(result)

// Application classes don't depend on a billion classes
// provided by the complex framework. Also, if you decide to
// switch frameworks, you only need to rewrite the facade
// class.
class Application is
    method main() is
        converter = new VideoConverter()
        mp4 = converter.convert("funny-cats-video.ogg", "mp4")
        mp4.save()
```

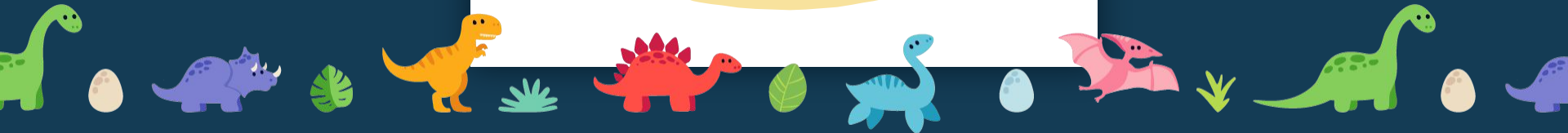
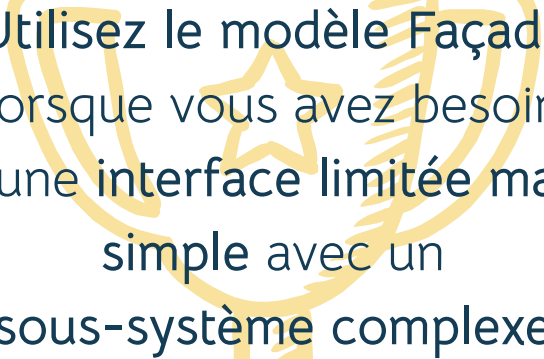




4. Implémentation



Utilisez le modèle Façade
lorsque vous avez besoin
d'une interface limitée mais
simple avec un
sous-système complexe

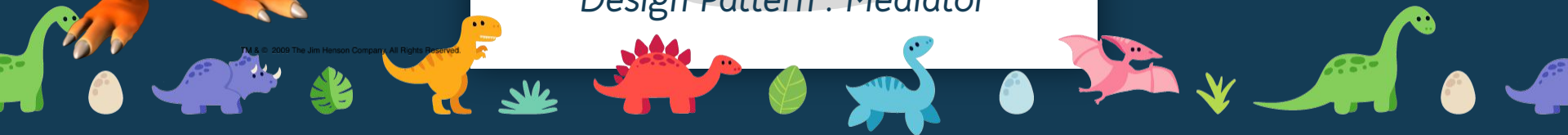




Utilisez la façade lorsque
vous souhaitez structurer un
sous-système en couches

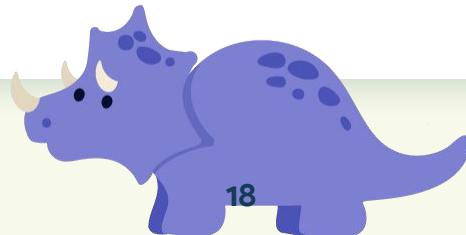
Design Pattern : Mediator

TM & © 2009 The Jim Henson Company. All Rights Reserved.



Implémentation

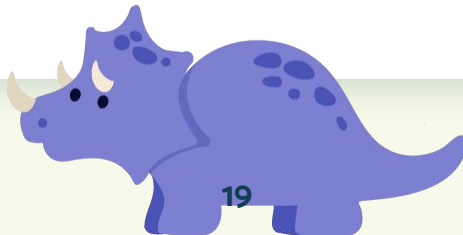
1. Vérifiez s'il est possible de fournir une interface plus simple que celle fournie par un sous-système existant.
2. Déclarez et implémentez cette interface dans une nouvelle classe de façade. La façade doit rediriger les appels du code client vers les objets appropriés du sous-système.



Implémentation

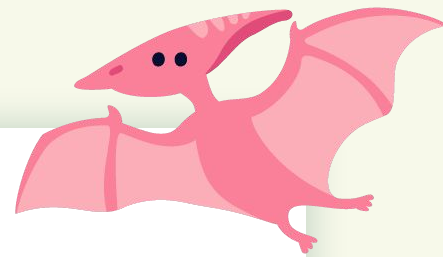
3. Pour tirer pleinement parti du modèle, tous les codes clients doivent communiquer avec le sous-système uniquement par la façade.

4. Si la façade devient trop grande, pensez à extraire une partie de son comportement vers une nouvelle classe de façade.





5. Pour et Contre



Pour

Vous pouvez isoler votre code de la complexité d'un sous-système

Contre

Une façade peut devenir un *god object* couplé à toutes les classes d'une application.



Alliances favorables

Mediator

Façade et Mediator ont un travail similaire : ils essaient d'organiser la collaboration entre de nombreuses classes étroitement liées.

Singleton

Une classe Façade peut souvent être transformée en Singleton puisqu'un seul objet de façade suffit dans la plupart des cas.



The background is a dark blue gradient. In the top left and top right corners, there are stylized green leaves and ferns. In the bottom left, there are green bushes and a small blue dinosaur. In the bottom right, there is a large orange dinosaur with a speech bubble pointing towards the center. The speech bubble is white and contains the text.

Merci!

Avez vous des questions ?

Source : <https://refactoring.guru/design-patterns/facade>

Slides : <https://slidescarnival.com/>