esign Pattern
dapter

LLEUX KILLIAN
DORIAN
BOUYSSOU GATIEN
LE SAUX LOGAN

Plan

- I - Présentation
  - Design Pattern
  - Adapter
- II – Exemple
- Conclusion

Présentation

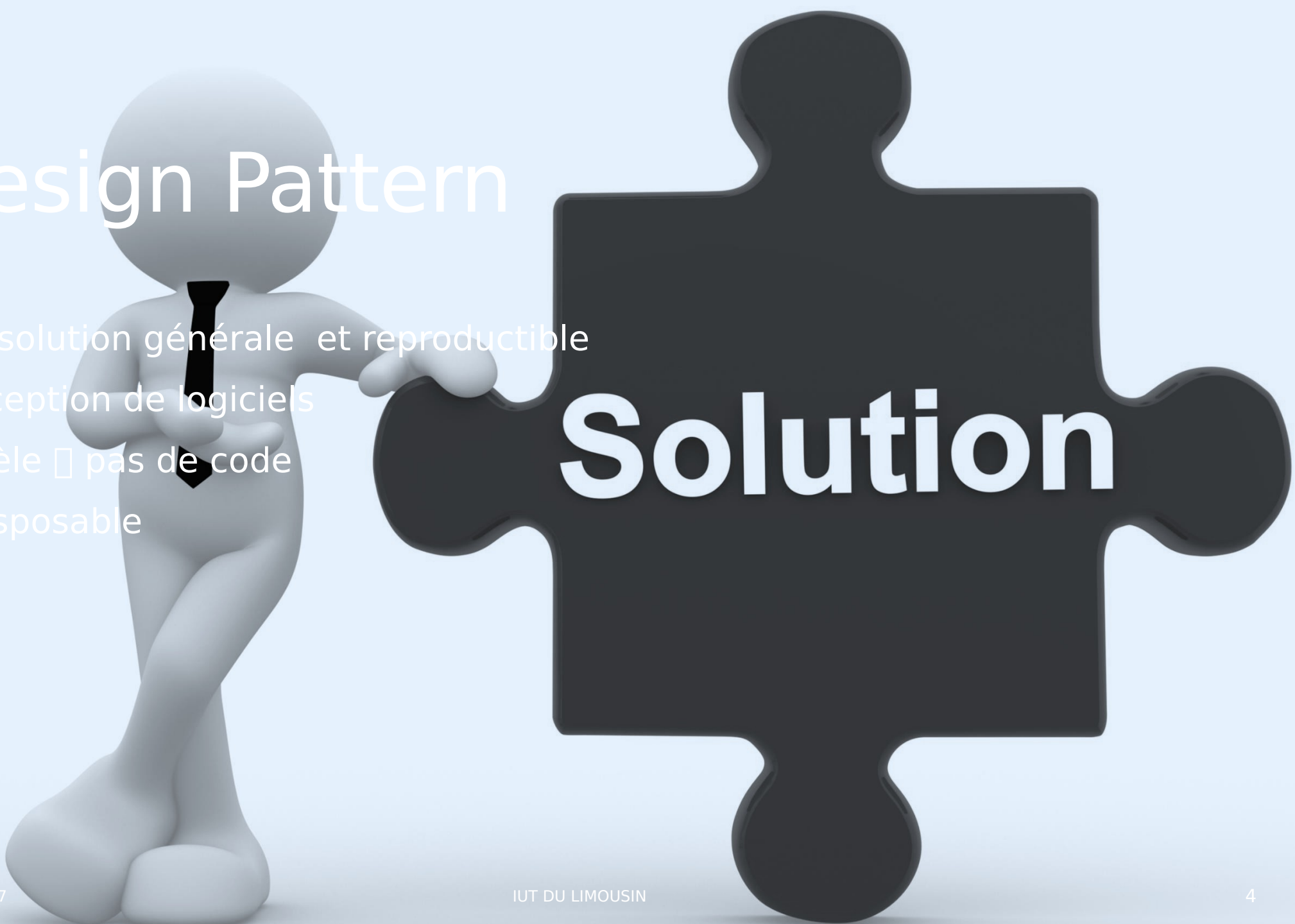
Design Pattern

Une solution générale et reproductible

Conception de logiciels

Modèle pas de code

Transposable



Problématique

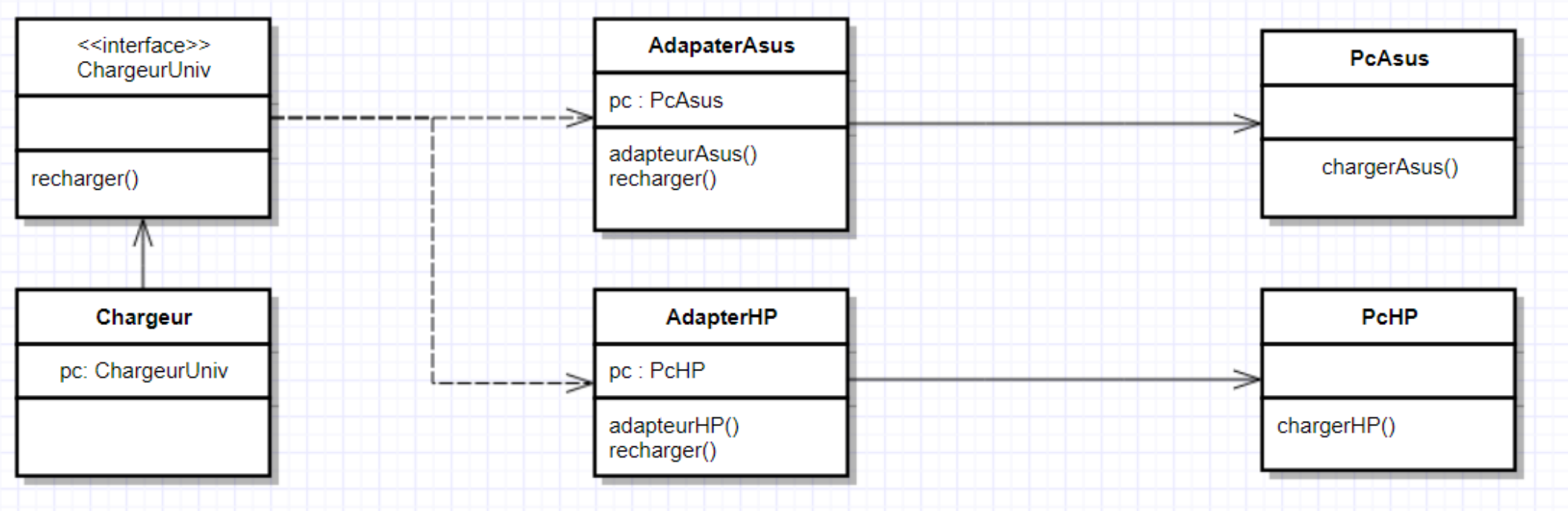
Faire fonctionner des objets incompatibles ensemble

Entreprise de production d’alimentations en tout genre

Nouveau marché : chargeurs PC

Chaque marque a son type de chargeur, comment faire ?

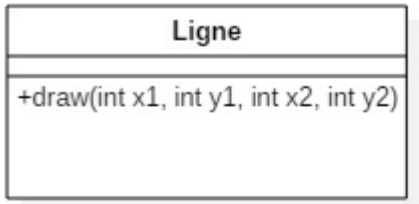
Représentation UML



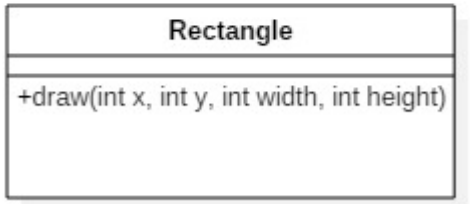
Exemple

II - Exemple

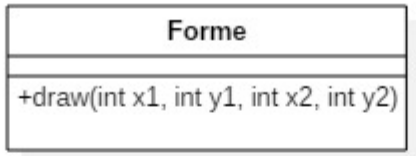
```java
public class Ligne {
    public void draw(int x1, int y1, int x2, int y2) {
        System.out.println("Ligne du Point A("
                + x1 + ";" + y1 + "),au point B("
                + x2 + ";" + y2 + ")");
    }
}
```



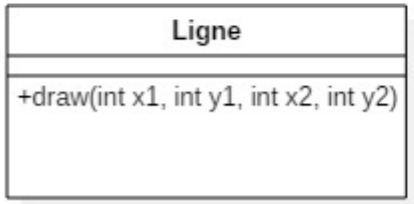

```java

public class Rectangle {
    public void draw(int x, int y, int width, int height) {
        System.out.println("Rectangle avec pour coordonnées ("
                + x + ";" + y + "), largeur: " + width
                + ", hauteur: " + height);
    }
}
```

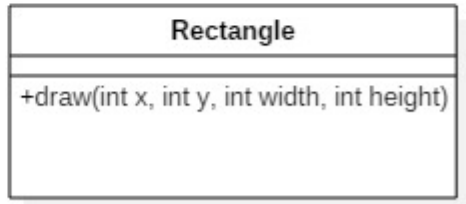


II - Exemple



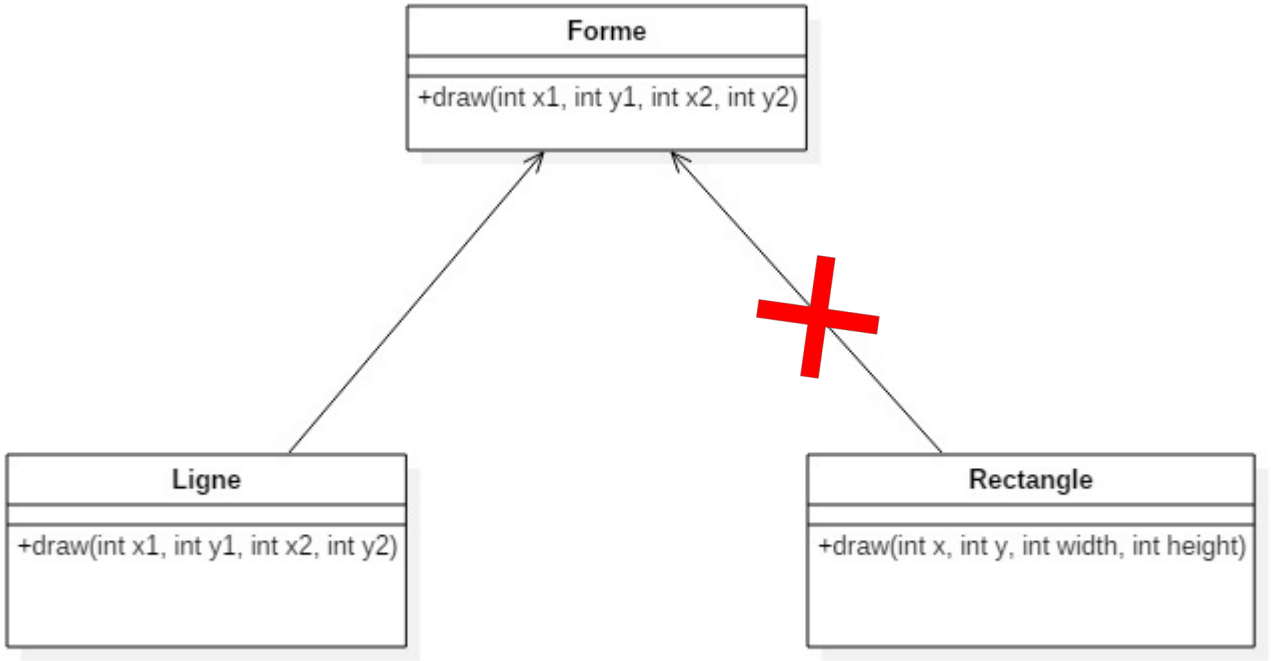

```java
public interface Forme {
    void draw(int x, int y, int z, int j);
}
```



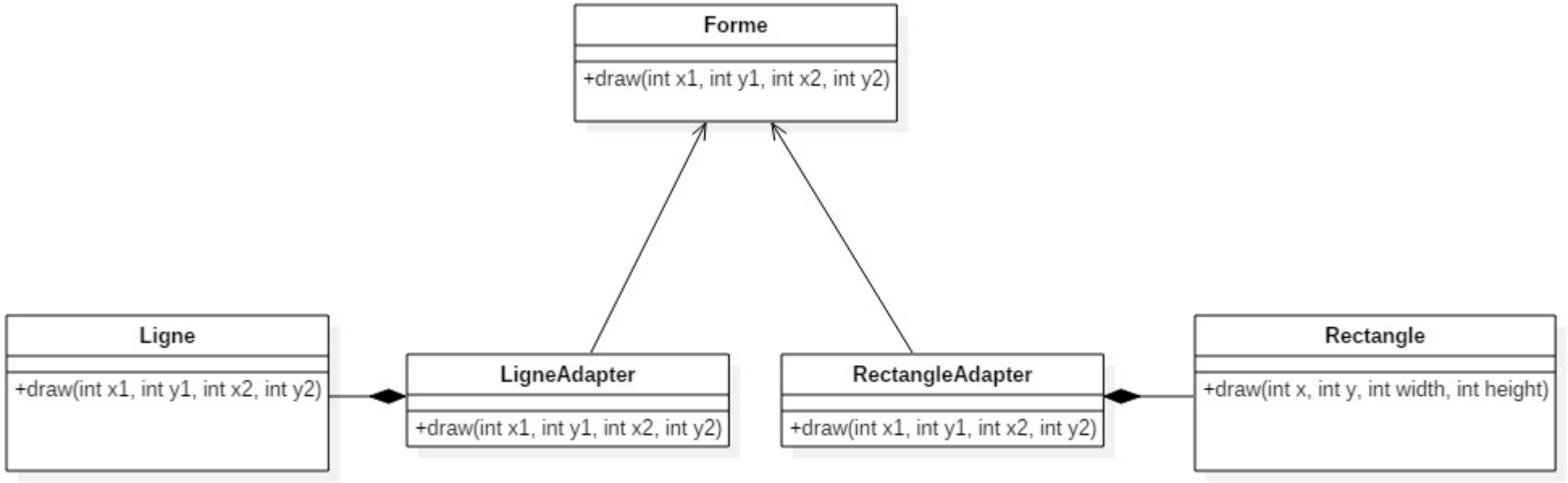


II - Exemple

```
1
2  public interface Forme {
3      void draw(int x, int y, int z, int j);
4  }
5
```



II - Exemple


Forme
+draw(int x1, int y1, int x2, int y2)
Ligne
+draw(int x1, int y1, int x2, int y2)
LigneAdapter
+draw(int x1, int y1, int x2, int y2)
RectangleAdapter
+draw(int x1, int y1, int x2, int y2)
Rectangle
+draw(int x, int y, int width, int height)

II - Exemple

```java
public class RectangleAdapter implements Forme{

    private Rectangle adapte;

    public RectangleAdapter(Rectangle rectangle) {
        this.adapte = rectangle;
    }

    @Override
    public void draw(int x1, int y1, int x2, int y2) {
        int x = Math.min(x1, x2);
        int y = Math.min(y1, y2);
        int width = Math.abs(x2 - x1);
        int height = Math.abs(y2 - y1);
        adapte.draw(x, y, width, height);
    }
}
```

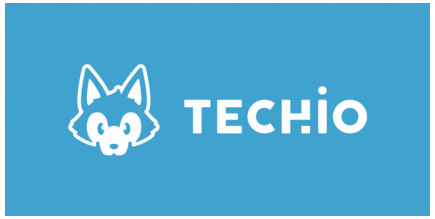
```java
public class LigneAdapter implements Forme{

    private Ligne adapte;

        public LigneAdapter(Ligne line) {
            this.adapte = line;
        }

        @Override
        public void draw(int x1, int y1, int x2, int y2) {
            adapte.draw(x1, y1, x2, y2);
        }
}
```

Conclusion

Faire communiquer deux classes sans les modifier

Faire collaborer de classes alors que leurs interfaces sont incompatibles

Quand l’utiliser ?



Sources

http://badger.developpez.com/tutoriels/dotnet/patterns/adaptateur/

https://fr.wikipedia.org/wiki/Adaptateur_(patron_de_conception)

https://sourcemaking.com/design_patterns/adapter