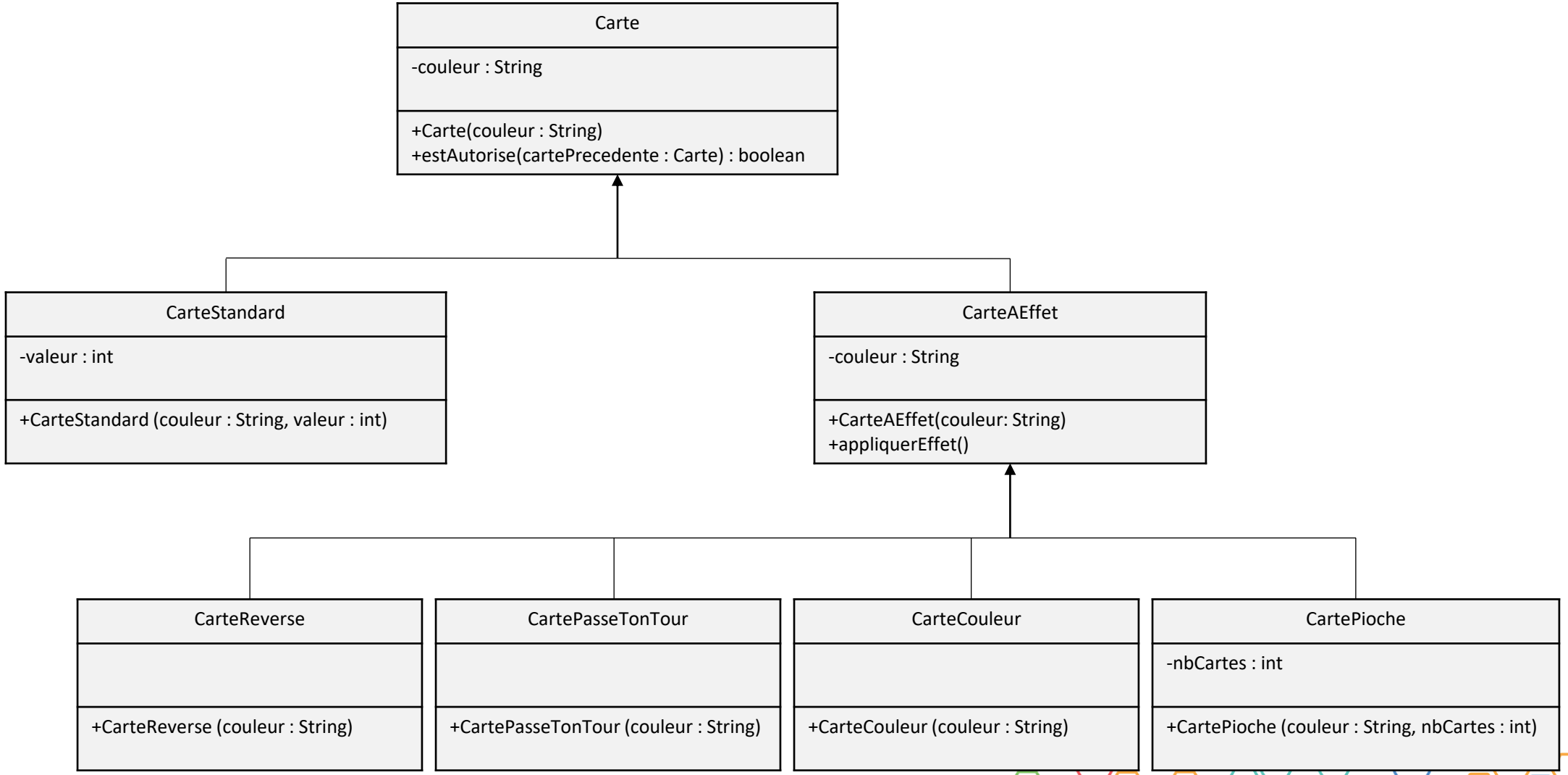
Java
Classe abstraite et interface

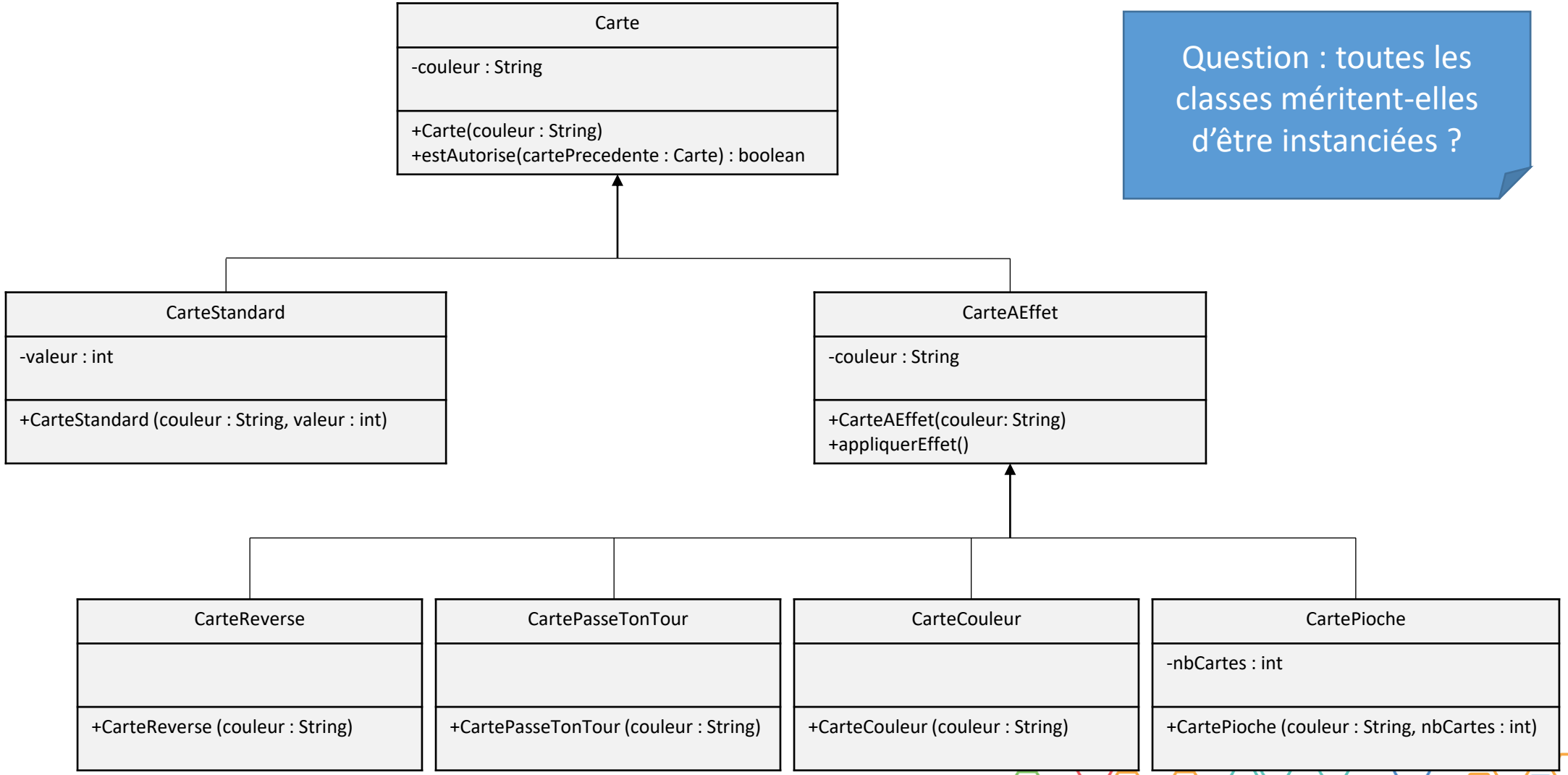
Objectifs

- Découvrir la notion d'abstraction
- Comprendre la différence entre une classe concrète et une classe abstraite
- Savoir coder des classes abstraites contenant éventuellement des méthodes abstraites
- Comprendre la notion d'interface et son intérêt

Les classes abstraites



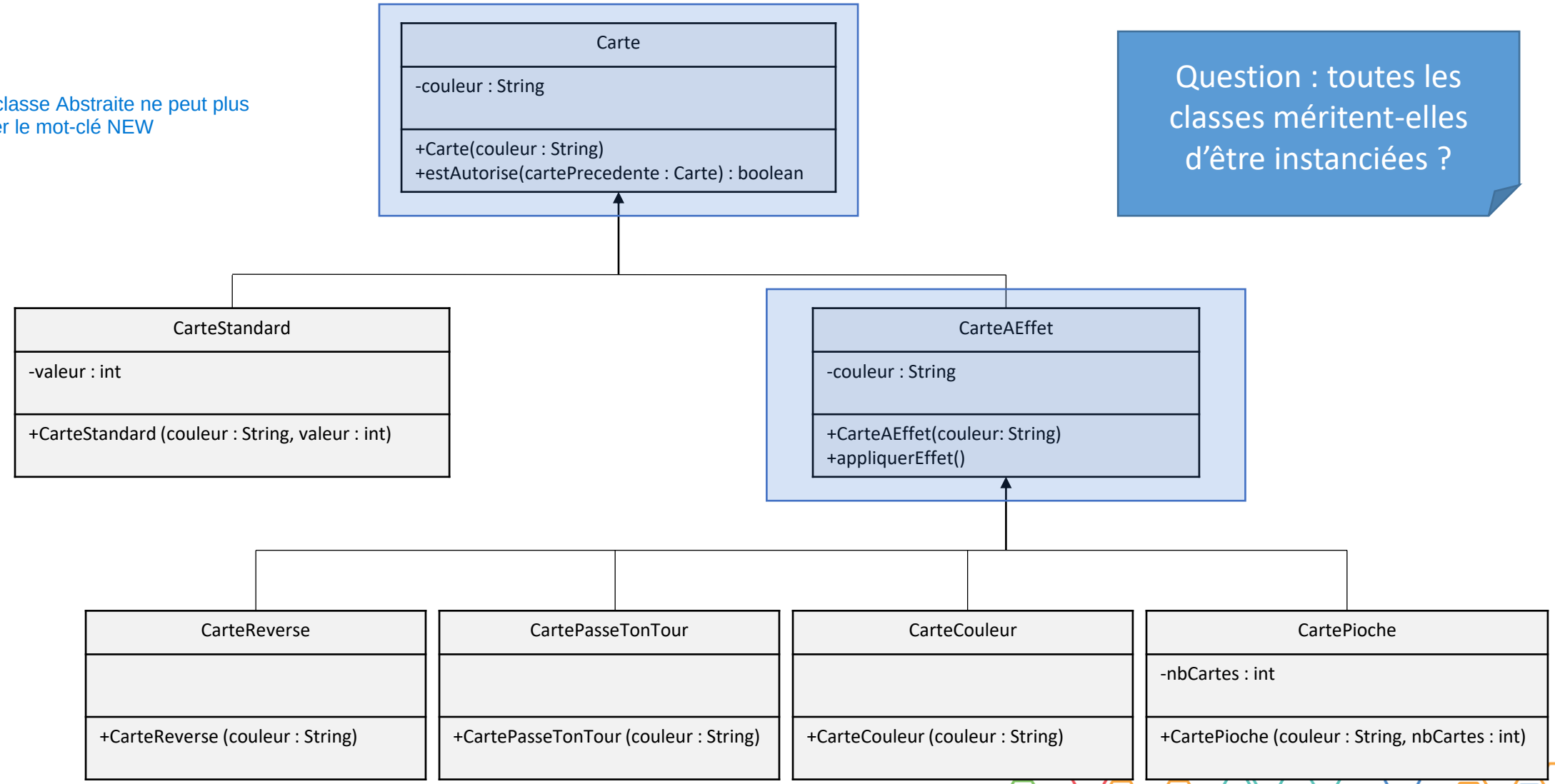
Les classes abstraites



Question : toutes les classes méritent-elles d'être instanciées ?

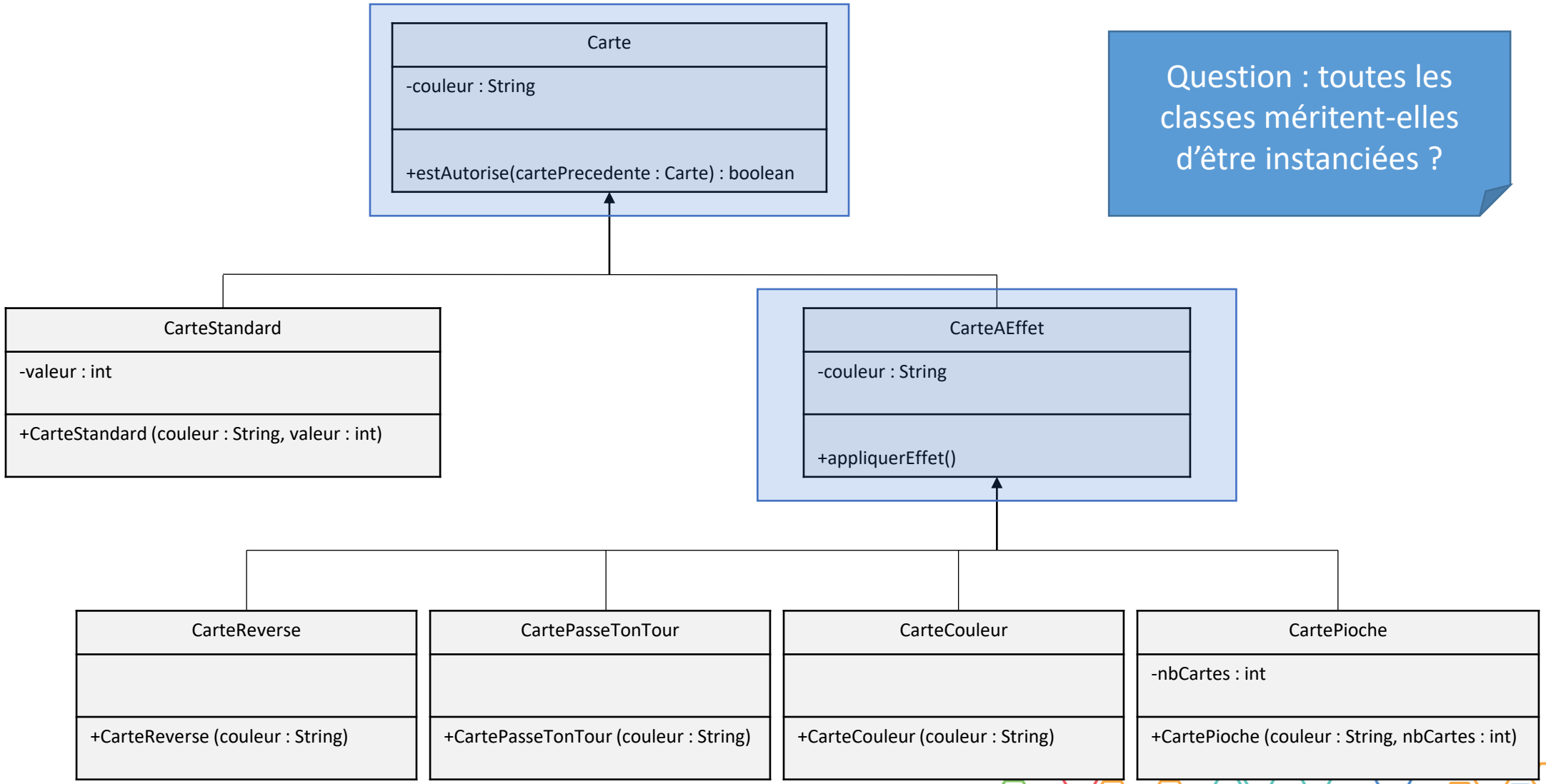
Les classes abstraites

Une classe Abstraite ne peut plus utiliser le mot-clé NEW

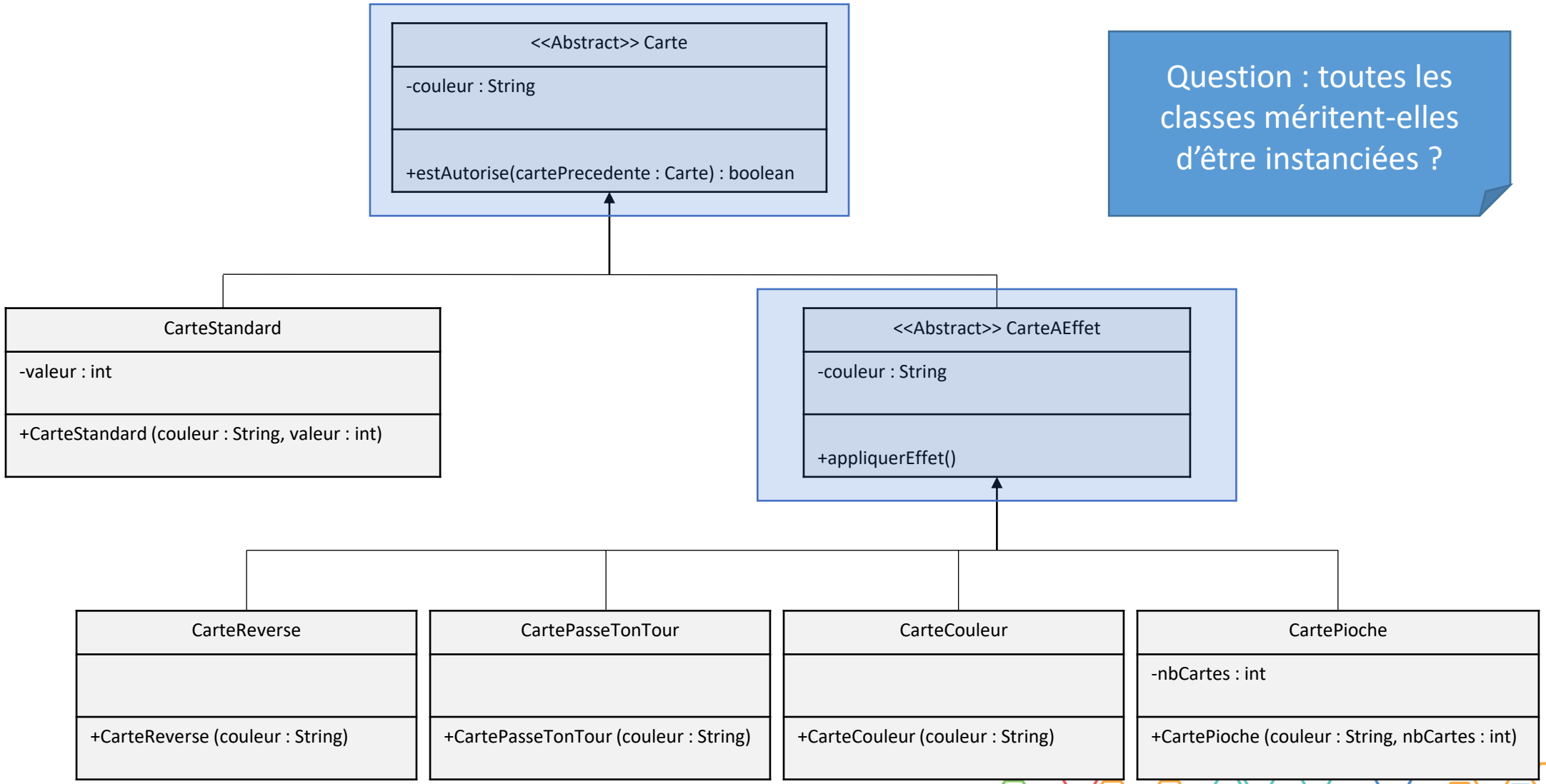


Question : toutes les classes méritent-elles d'être instanciées ?

Les classes abstraites



Les classes abstraites



Les classes abstraites

```
public abstract class Carte {
    protected String couleur;

    public Carte(String couleur) {
        this.couleur = couleur;
    }
```

Les classes abstraites

```
public abstract class Carte {
    protected String couleur;

    public Carte(String couleur) {
        this.couleur = couleur;
    }
```

Le mot clé abstract permet d'empêcher l'instanciation d'une classe

Interdiction d'un:
Carte carte1= new Carte;

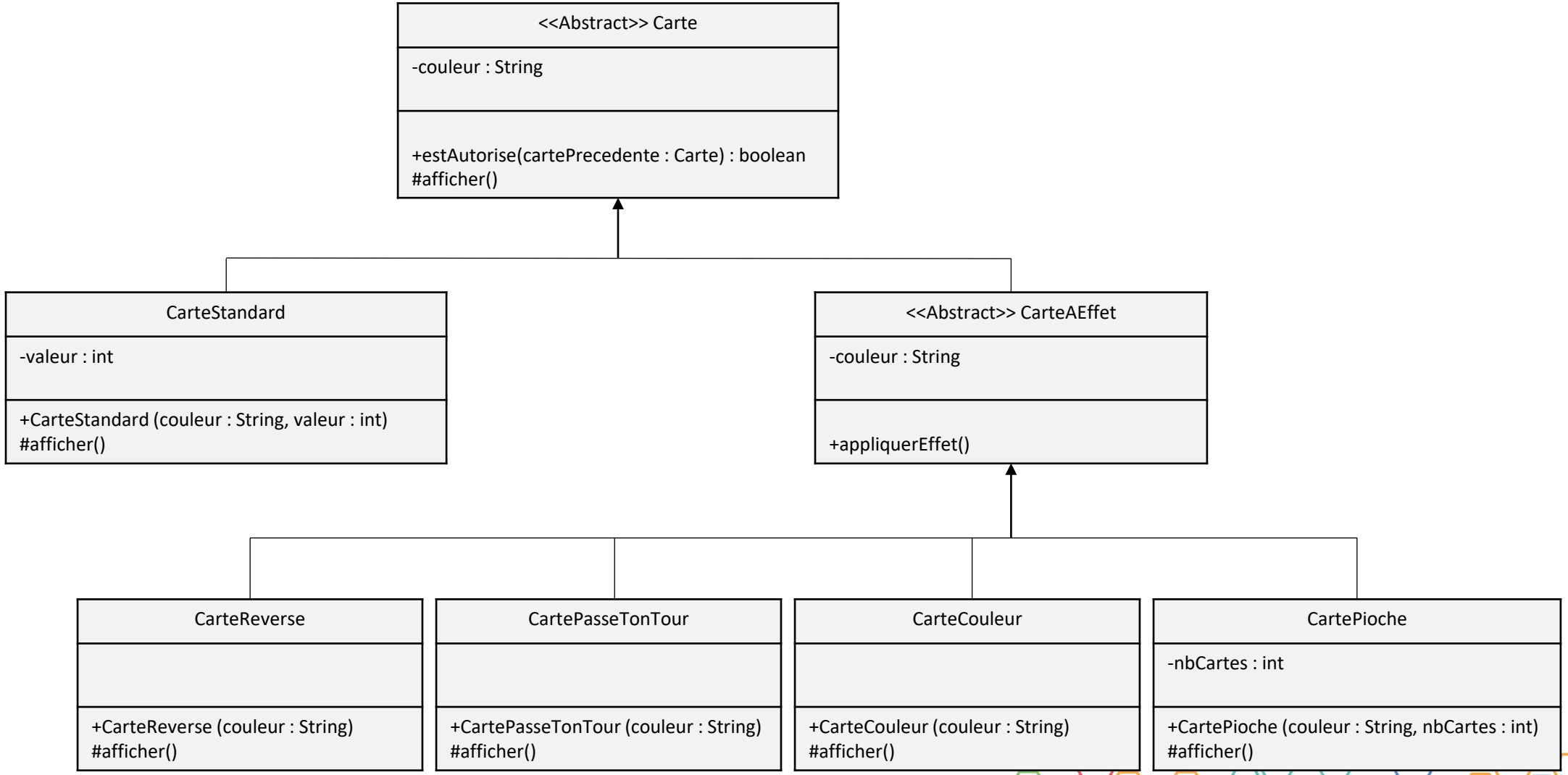
Les classes abstraites

```
public abstract class Carte {
    protected String couleur;

    public Carte(String couleur) {
        this.couleur = couleur;
    }
```

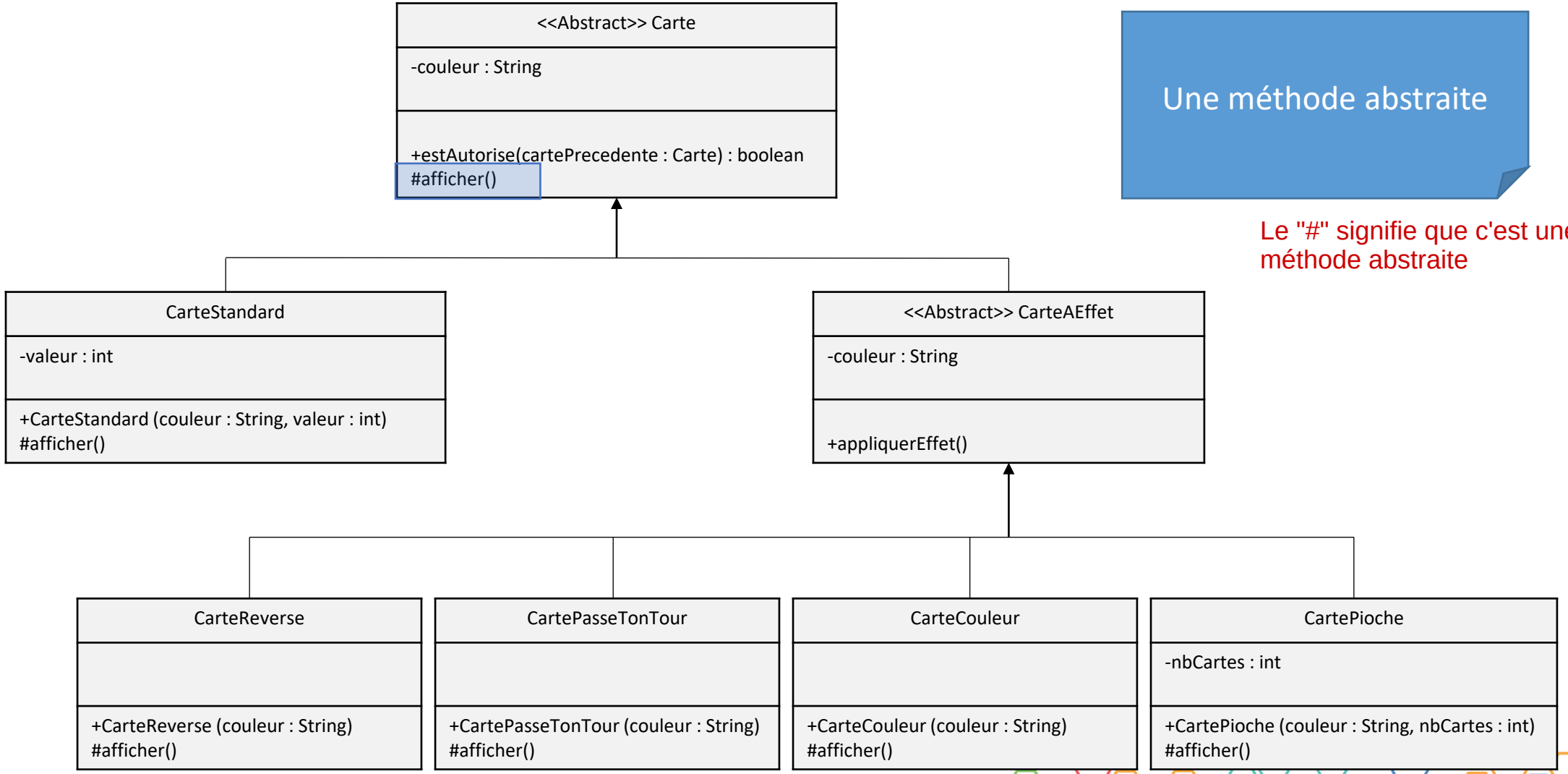
Le mot clé abstract permet d'empêcher l'instanciation d'une classe

```
public class TestCarte {
    public static void main(String[] args) {

        Carte maCarte = new Carte("jaune");
```

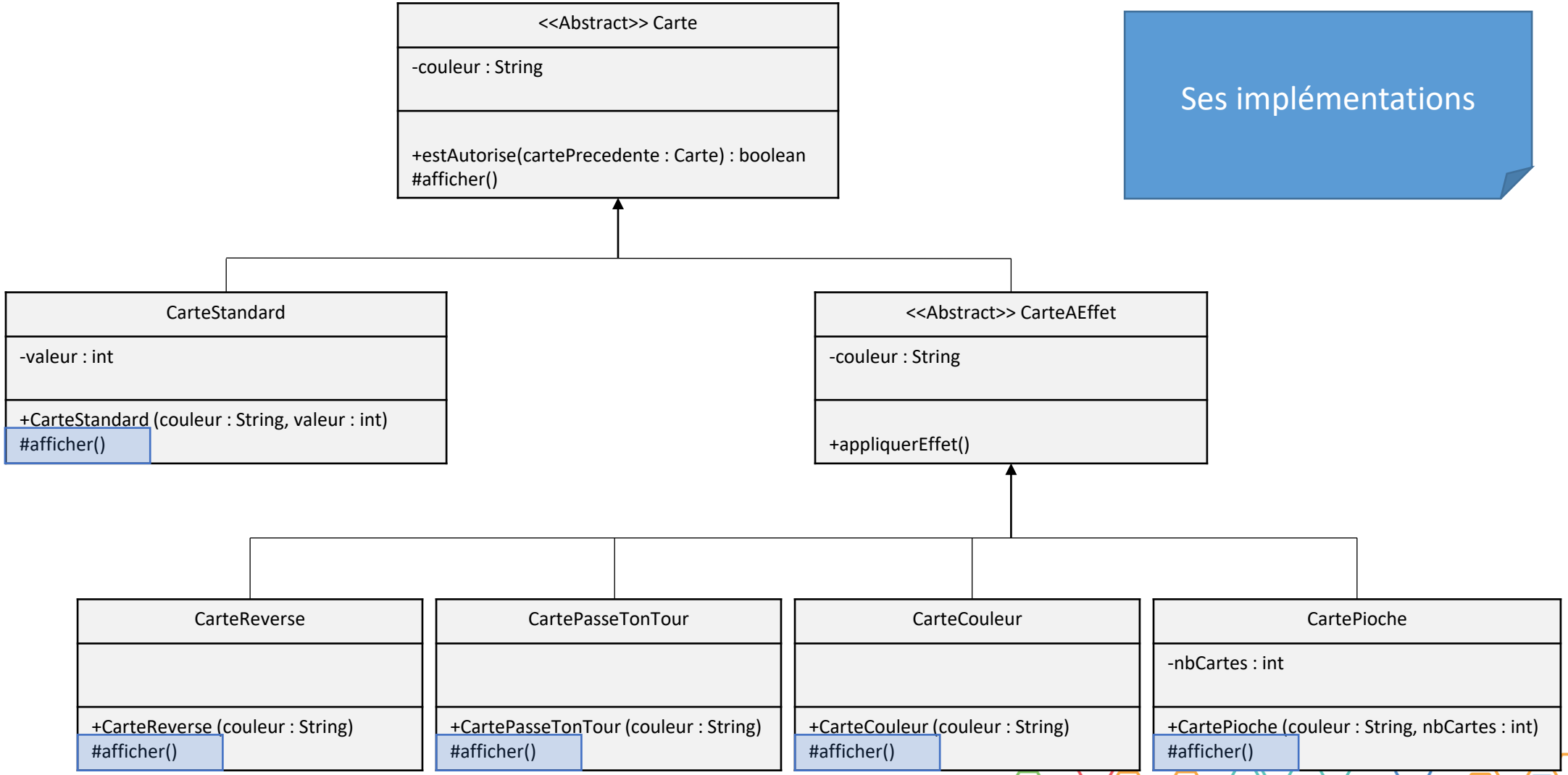
Les classes abstraites



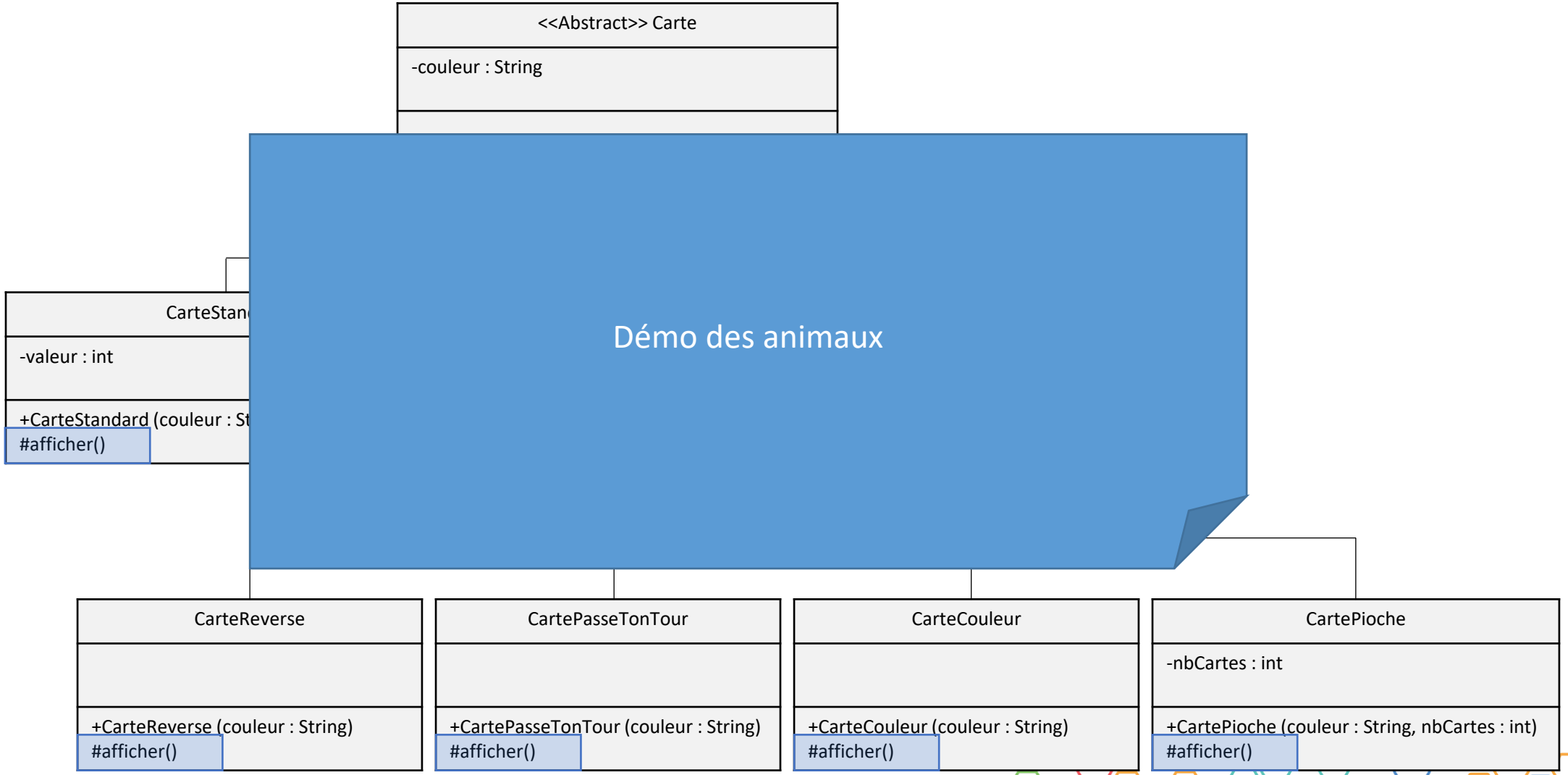
Les classes abstraites



Les classes abstraites



Les classes abstraites

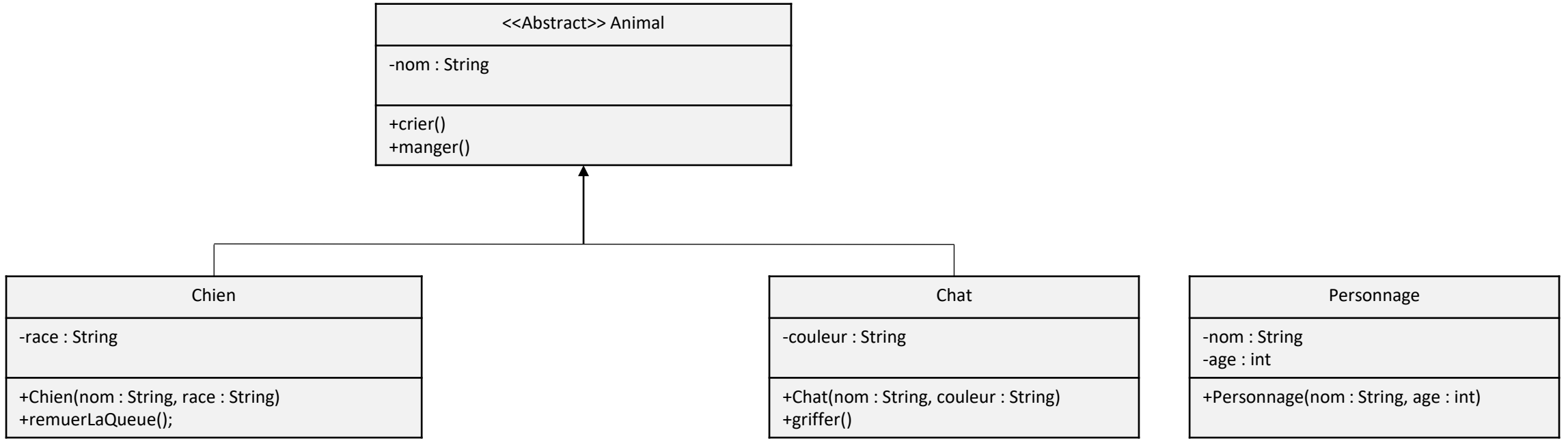


Les interfaces

- Les interfaces sont une forme de contrat que l'on peut imposer aux classes.

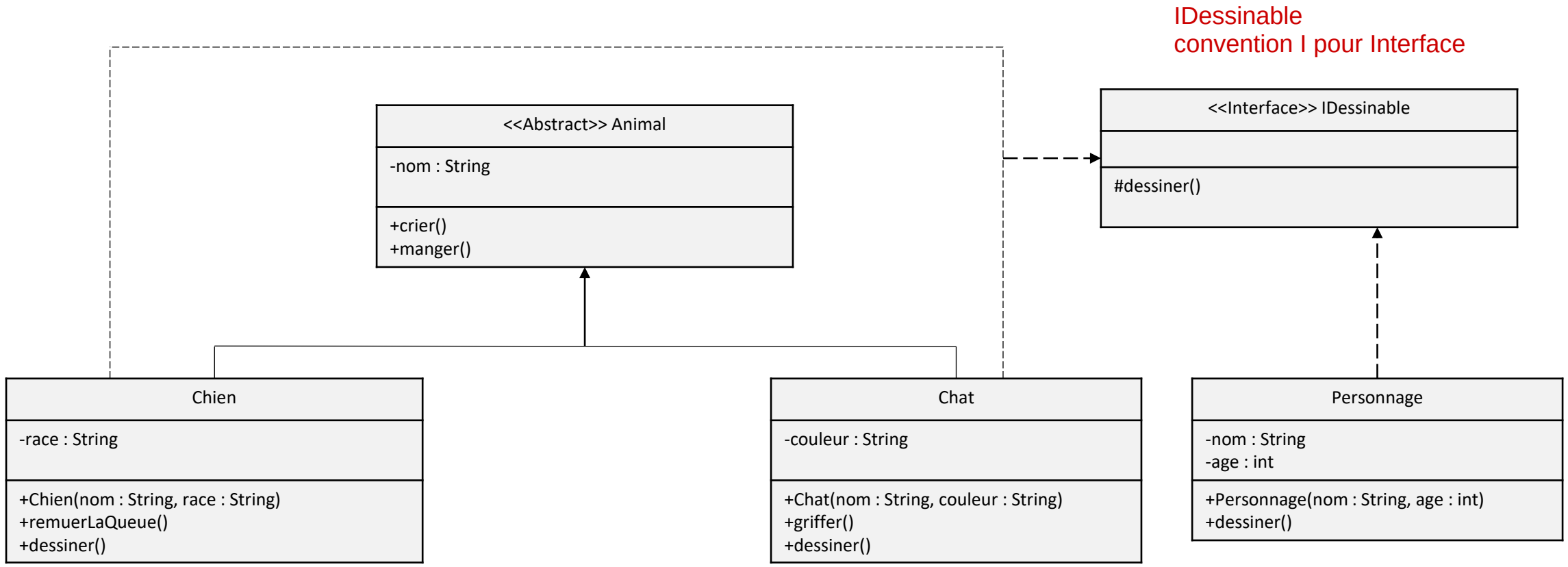
Les interfaces

- Les interfaces sont une forme de contrat que l'on peut imposer aux classes.



Les interfaces

- Les interfaces sont une forme de contrat que l'on peut imposer aux classes.

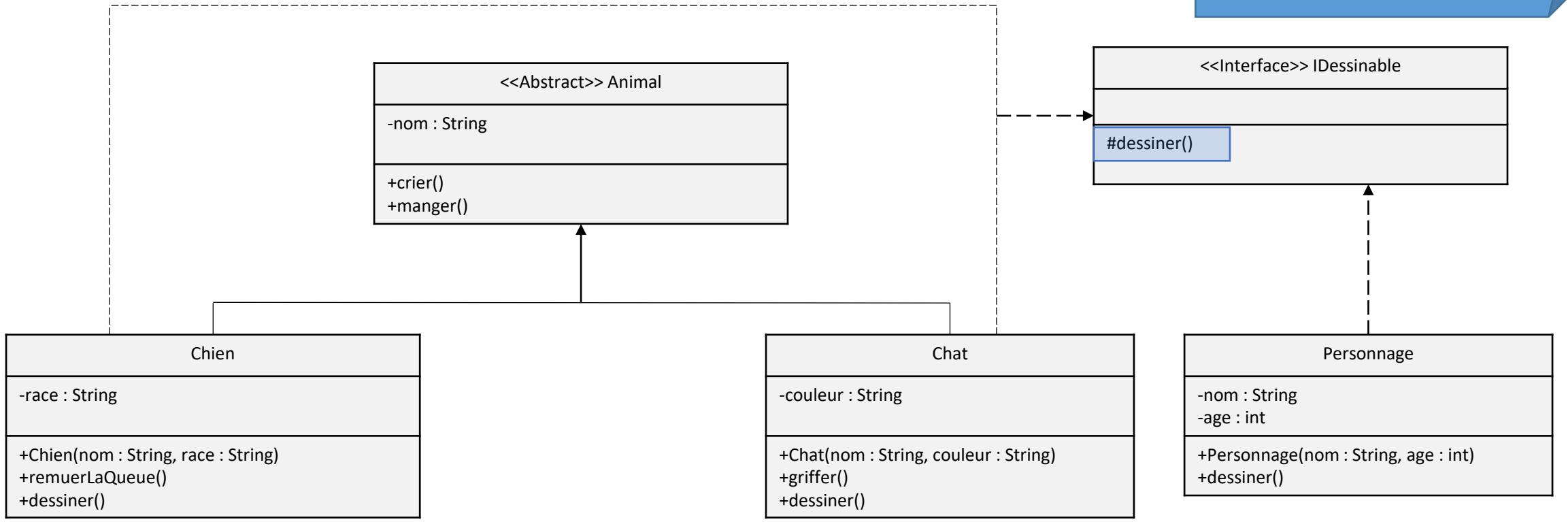


Les interfaces

- Les interfaces sont une forme de contrat que l'on peut imposer aux classes.

Méthodes abstraites

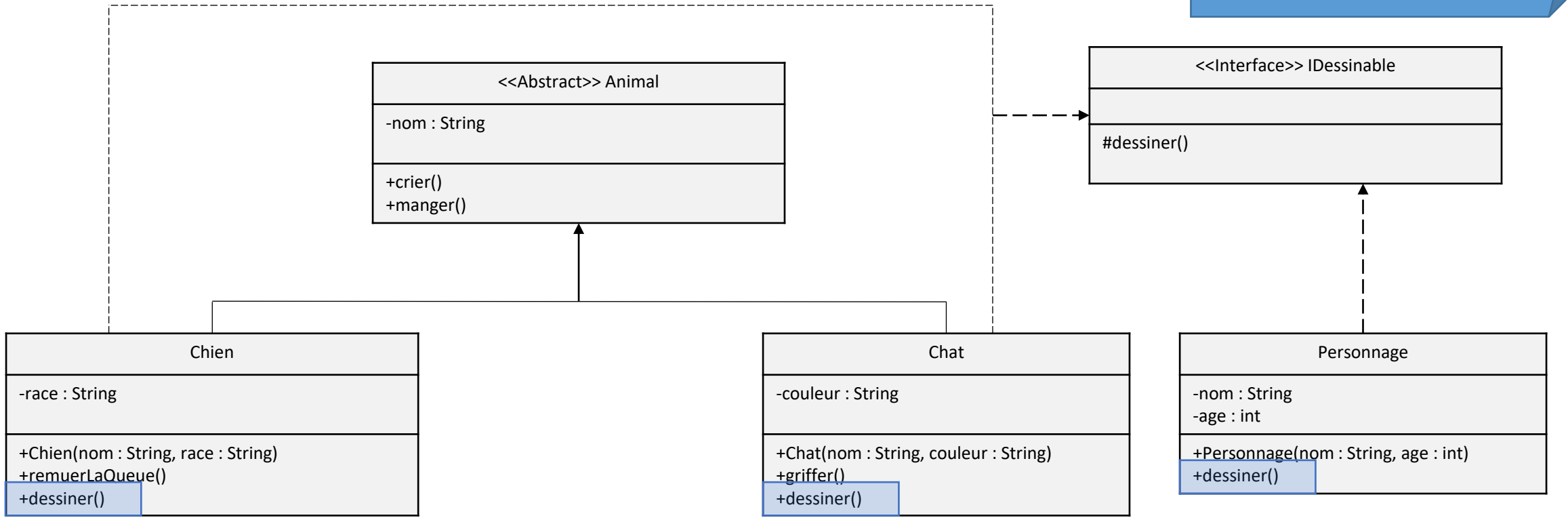
On aurait pu mettre IDessinable dans la class abtraite animal
=>tous les animaux doivent avoir la méthode dessiner



Les interfaces

- Les interfaces sont une forme de contrat que l'on peut imposer aux classes.

Implémentation des méthodes



Les interfaces

- Les interfaces sont une forme de contrat que l'on peut imposer aux classes.

```
public interface IDessinable {
    void dessiner();
}
```

*Toutes les méthodes d'une interface sont **public abstract** par défaut.*

Ici:
void dessiner();
<=>
public abstract void dessiner();
(par défaut pour une interface)

Héritage entre interface très rare

Une IF n'a que des méthodes, jamais d'attribut.

Les interfaces

- Les interfaces sont une forme de contrat que l'on peut imposer aux classes.

```
public interface IDessinable {
    void dessiner();
}
```

*Toutes les méthodes d'une interface sont **public abstract** par défaut.*

```
public class Chat extends Animal implements IDessinable {
    @Override
    public void dessiner() {
        System.out.println(" /\\_/\\");
        System.out.println("( o.o )");
        System.out.println(" > ^ <");
    }
}
```

On peut implémenter autant d'interfaces (IF) que l'on souhaite sur une class

Les interfaces ne sont pas très fréquentes dans le code

Si comportement par defaut: on fait une class abstraite
Sinon: on fait une IF

Les interfaces

- Les interfaces sont une forme de contrat que l'on peut imposer aux classes.

```
public interface IDessinable {
    void dessiner();
}
```

*Toutes les méthodes d'une interface sont **public abstract** par défaut.*

```
public class Chat extends Animal implements IDessinable {
    @Override
    public void dessiner() {
        System.out.println(" /\\_/\\");
        System.out.println("( o.o )");
        System.out.println(" > ^ <");
    }
}
```

```
public class Chien extends Animal implements IDessinable {
    @Override
    public void dessiner() {
        System.out.println(",-.___,-.");
        System.out.println("\\_/_ _\\_/");
        System.out.println("  )O_O(");
        System.out.println(" { (_) }");
        System.out.println("  `-^-'");
    }
}
```

Souvent une interface permet une action (clique souris par exemple)

Les interfaces

- Les interfaces sont une forme de contrat que l'on peut imposer aux classes.

```
public interface IDessinable {
    void dessiner();
}
```

*Toutes les méthodes d'une interface sont **public abstract** par défaut.*

```
public class Chat extends Animal implements IDessinable {
    @Override
    public void dessiner() {
        System.out.println(" /\\_/\\");
        System.out.println("( o.o )");
        System.out.println(" > ^ <");
    }
}
```

```
public class Bonhomme implements IDessinable {
    @Override
    public void dessiner() {
        System.out.println("¯\\_(ツ)_/¯");
    }
}
```

```
public class Chien extends Animal implements IDessinable {
    @Override
    public void dessiner() {
        System.out.println(",-.___,-.");
        System.out.println("\\_/_ _\\_/");
        System.out.println("  )O_O(");
        System.out.println(" { (_) }");
        System.out.println("  `-^-'");
    }
}
```

Bilan

Dans la définition de	Classe concrète	Classe abstraite	Interface
Déclaration	`class`	`abstract class`	`interface`
Attributs	✓	✓	✗
Constantes	✓	✓	✓
Constructeur	✓	✓	✗
Méthode concrète	✓	✓	✗
Méthode abstraite (mot-clef)	✗	✓ (`abstract`)	✓ (Ø)

Dans l'utilisation de	Classe concrète	Classe abstraite	Interface
Déclaration	`extends`	`extends`	`implements`
Minimum	0	0	0
Maximum	1	1	∞

nb class hérités max

Atelier 6
Balade à vélo

Java
Classe abstraite et interface