Patrick GIRARD
University of Poitiers

Object-Oriented Programming in Java™

Advanced Inheritance

Chapter 6 - Section 3

Table of contents

Abstract classes: interest

- What can we do when the behaviour cannot be known in the parent class?
- Ex. to fold up the sunroof of a convertible car
  - 2CV: to roll up the canvas
  - C3 Pluriel : folding up the canvas, and removing the supporting archs

Folding up the sunroof

Unfold : déplier

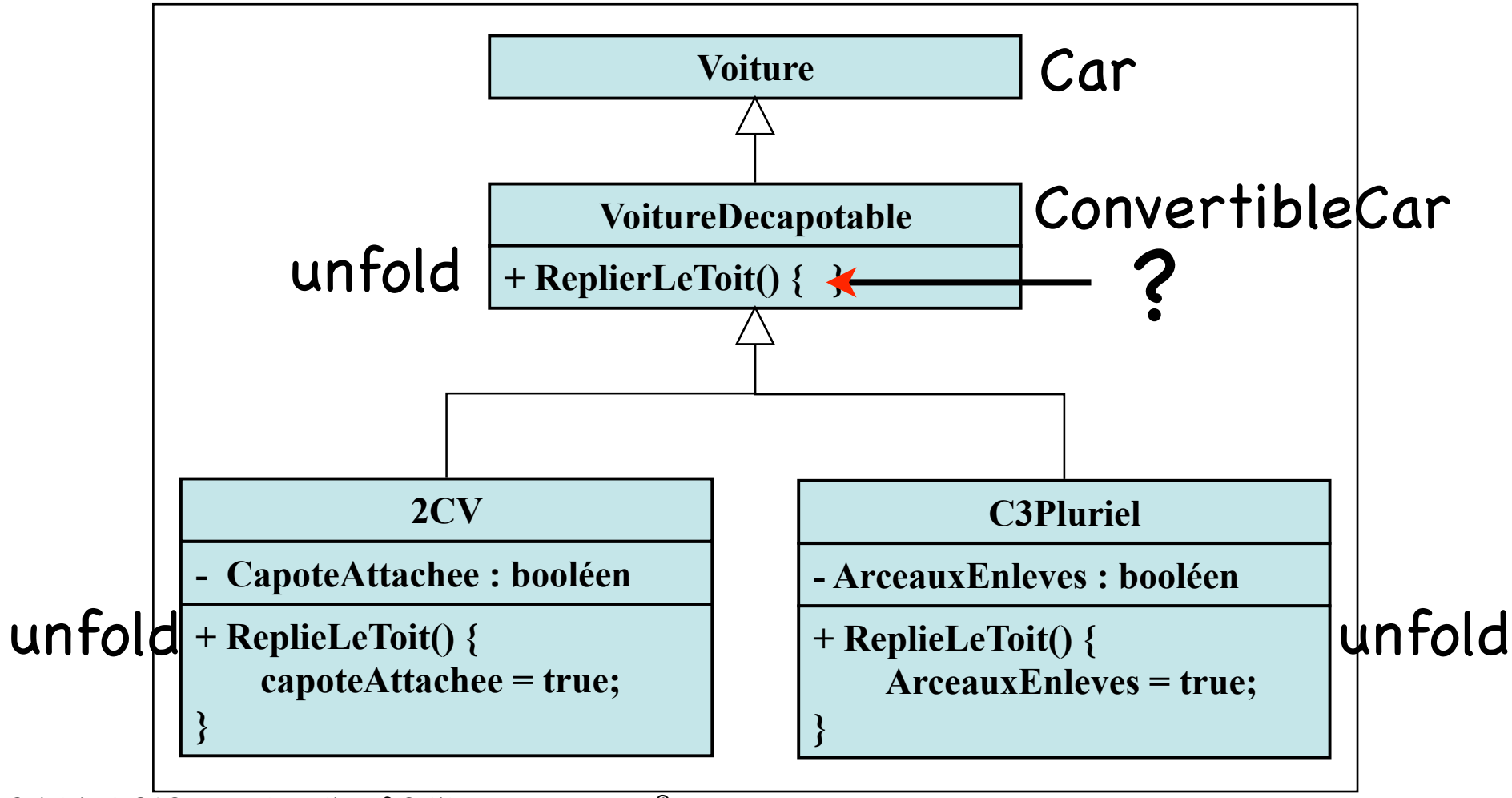
Fold up : replier

Sunroof : toît ouvrant

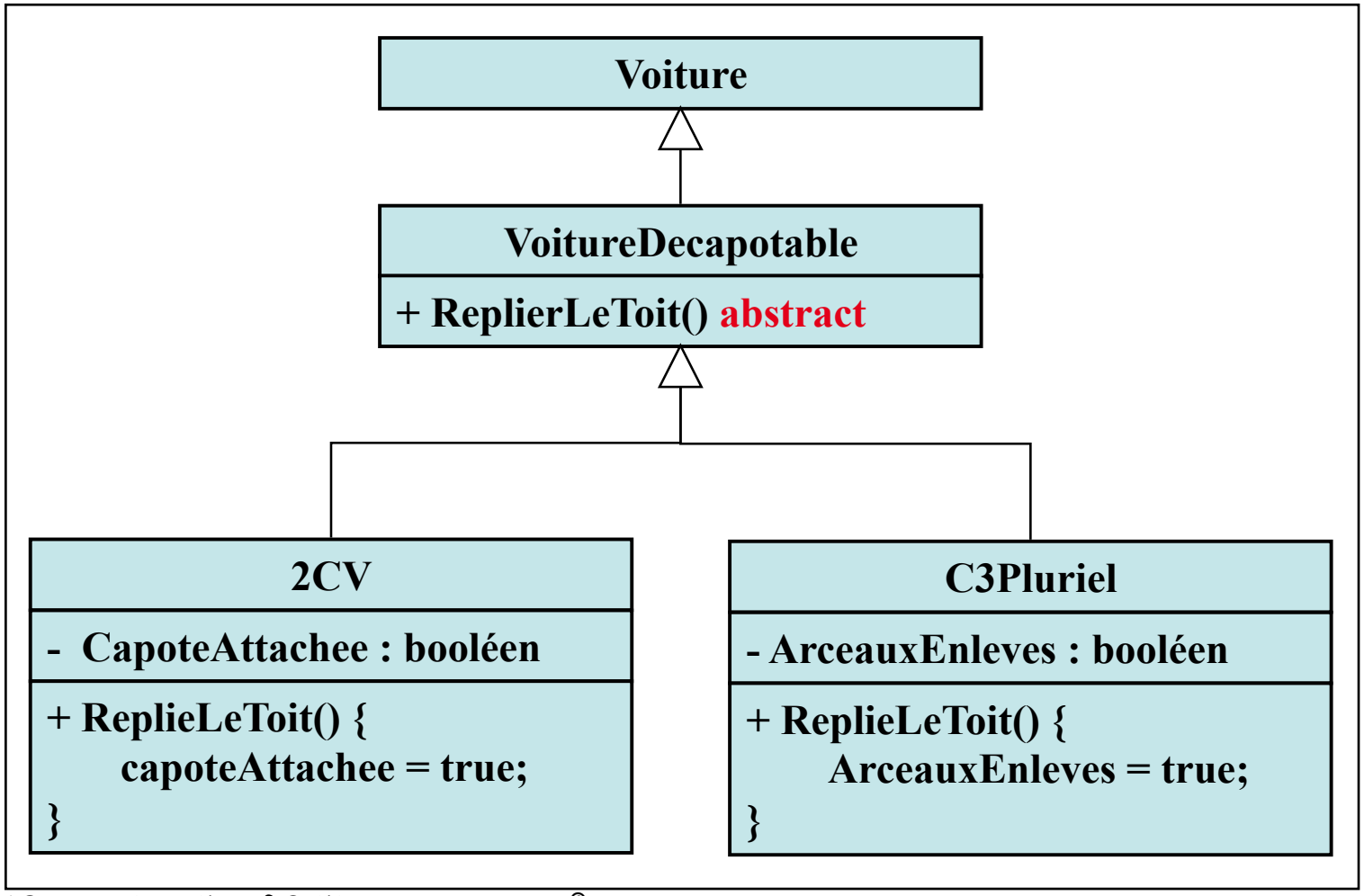
Canvas : toile

Convertible car : voiture décapotable

Abstract classes: the problem



Abstract: the solution



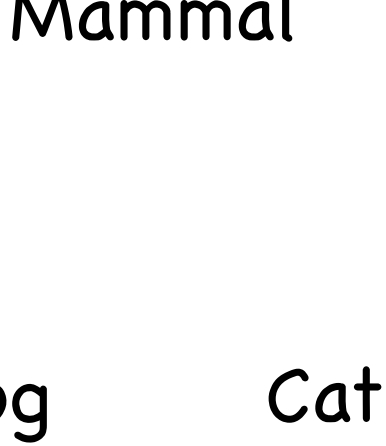
Abstract classes: the solution

- Declare the method (and the class) abstract
  - No required implementation

```
public abstract class VoitureDecapotable
   extends Voiture {
      public abstract void replieLeToit();
}
```

Abstract classes: justification

- Abstract classes are natural in inheritance trees
  - A mammal does not exist
  - A dog exists
  - A cat exists



Abstract classes: the solution

- Implement in the child classes

```
public class DeuxChevaux extends VoitureDecapotable {
   private boolean capote_attachee;
   public void replieLeToit() {
         this.capote_attachee = true;
   }
}
```

not overriding...

```
public class C3Pluriel extends VoitureDecapotable {
   private boolean arceaux_retires;
   public void replieLeToit() {
      this.arceaux_retirés = true;
   }
}
```

... but implementing

Abstract classes: summary

- Abstract class declaration
  - public abstract class nom ...
- Abstract method declaration
  - public abstract void meth1 (...);
  - public abstract Object meth2 (...);

Abstract classes: rules

- Instanciation is not possible

```
public abstract class Exemple1 {
   public abstract void MethodeAbstraite();
   public static void main(String args[]){
      Exemple1 e = new Exemple1()
   }
}
```

```
Exemple1 is abstract; cannot be instantiated
  Exemple1 e = new Exemple1();
                           ^
1 error
```

Abstract classes: rules

- If only one abstract method in a class, the class becomes abstract

```
class Exemple3 {
   public abstract void MethodeAbstraite();
}
```

```
Exemple3 is not abstract and does not override  abstract method MethodeAbstraite() in Exemple3
class Exemple3 {
^
1 error
```

Abstract classes: rules

- Childs classes are required to implement abstract methods

```
class abstract Exemple2 {
   public abstract void MethodeAbstraite();
}
```

```
public class Concrete extends Exemple2
}
```

```
Concrete is not abstract and does not override abstract method MethodeAbstraite() in Exemple2
public class Concrete extends Exemple2 {
       ^
1 error
```

Abstract classes: usage

```
public class Utilise {
  public static void main (String[] argv) {
    // Déclaration et création d'une DeuxChevaux
    DeuxChevaux voiture_ancienne = new DeuxChevaux();
    // Envoi de message
    voiture_ancienne.replieLeToit();

    // Déclaration d'une VoitureDecapotable (pas de création)
    VoitureDecapotable voiture_decapotable ;
    // voiture_decapotable = new VoitureDecapotable(); // Impossible
    voiture_decapotable = voiture_ancienne ;
    voiture_decapotable.replieLeToit();

    // Déclaration et création d'une C3Pluriel
    VoitureDecapotable voiture_recente = new C3Pluriel();
    // Envoi de message
    voiture_recente.replieLeToit();
    }
}
```

Impossible

Ok: upcasting

Ok: using