

**Safe Metaclass Composition  
Using  
Mixin-Based Inheritance**

**Noury Bouraqadi**  
Computer Science Laboratory (CSL)  
Ecole des Mines de Douai  
France



**Outline**

- **Mixin Based Inheritance**
- **Metaclass Composition Using Mixins**
- **Conclusion**

- **Context**
  - Single inheritance *inherits from class*
  - Unrelated hierarchies
  - Same Properties
- **Goal**
  - Reuse shared properties
  - Avoid code duplication
  - Alternative to multiple inheritance

**Mixin-Based Inheritance**

**Single Inheritance Behind the scene**

**Mixin = Subclass Generator**  
[Bracha & Cook 90]

### Example of Inheritance from Different Mixins

Point

ColoredBoundedPoint inherits from class Point

Colored (Mixin): color, printOn: color, color

Bounded (Mixin): boundsRectangle, printOn: move: bounds: bounds

ColoredBoundedPoint mixes in: {Colored, Bounded}

- Explicit Linearization on Definition:**
  - ColoredBoundedPoint mixins: {Colored, Bounded}
- Lookup list**
  - ColoredBoundedPoint, Colored, Bounded, Point

### Outline

- Mixin Based Inheritance
- Metaclass Composition Using Mixins
- Conclusion

### Main Idea and Issues

Singleton

LazyMemoryAllocation

MetaC (instance of C)

Singleton and LazyMemoryAllocation inherit from MetaC

- But, we need also:**
  - Compatibility (inheritance + inter-level messages)
  - Class specific properties

### The Upward Compatibility issue

foo is NOT understood by B

Metaclass level: MetaA (foo), MetaB (MetaX)

Class level: A (self class foo), B (subclass of A)

Instance level: aB (instance of B)

### The Downward Compatibility issue

bar is NOT understood by aZ

Metaclass level: MetaW (self new bar), MetaZ (subclass of MetaW)

Class level: W (bar), Z (instance of Y)

Instance level: aZ (instance of Z)

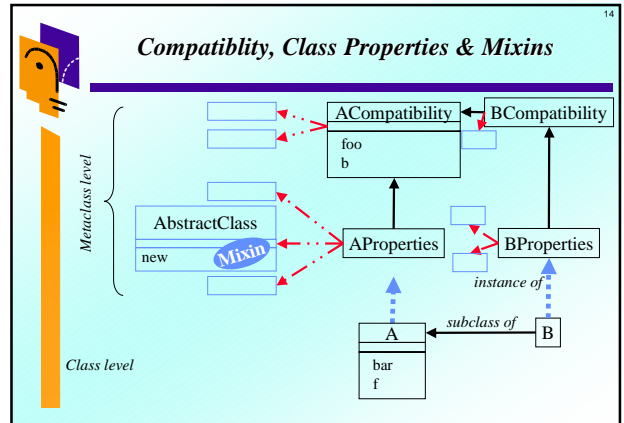
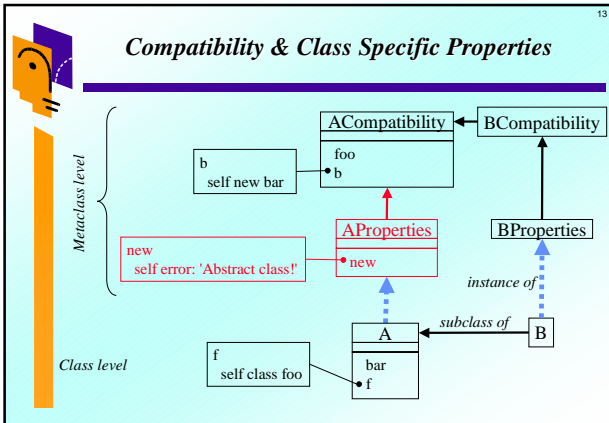
### No Class Specific Properties in Smalltalk

Unwanted property propagation B becomes abstract!


Metaclass level: MetaA (self new bar), MetaB (subclass of MetaA)

Class level: A (bar, f), B (subclass of A)

Instance level: f (instance of A)



- ### Outline
- **Mixin Based Inheritance**
  - **Metaclass Composition Using Mixins**
  - **Conclusion**

- ### Summary
- **Metaclasses are useful**
    - Class properties = new "kinds" of classes
    - e.g. Mixin-based inheritance (3 class properties)
  - **Metaclass Composition using Mixins**
    - Class-Metaclass compatibility
    - No undesirable class properties propagation
  - **Implementation on top of Squeak**
    - Mixin-Based Inheritance = 3 metaclasses
    - No Performance Loss!
- 

- ### Some Future Works
- **Extend mixin-based inheritance**
    - Traits approach for methods composition
    - Instance variables composition
  - **OO Programming without "traditional" inheritance**
    - Mixin-based inheritance only!
  - **Refactoring Squeak/Smalltalk libraries**
    - Explicit metaclasses + Mixins
    - New kernel (bootstrap)

### Thanks for your attention Questions? Comments?

Documents & Download  
<http://csl.enscm-douai.fr/MetaclassTalk>