

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

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

Metaclasses

- **Classes which instances are also classes**
 - Allow defining class properties
[Cointe87/90, Briot89, Danforth94-96, Ledoux96]:
 - Abstract, Singleton,
 - Multiple Inheritance, Final,
 - Lazy memory allocation, Persistent instances,...
- **Implicit in Smalltalk**
 - Solution for safe usage...
 - But, no reuse/composition

Outline

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

Need for Mixin-Based Inheritance

- **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

① inherits from mixin Colored

② inherits from mixin 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

- **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 W)

Class level: W (bar), Z (bar)

Instance level: aZ (instance of Z)

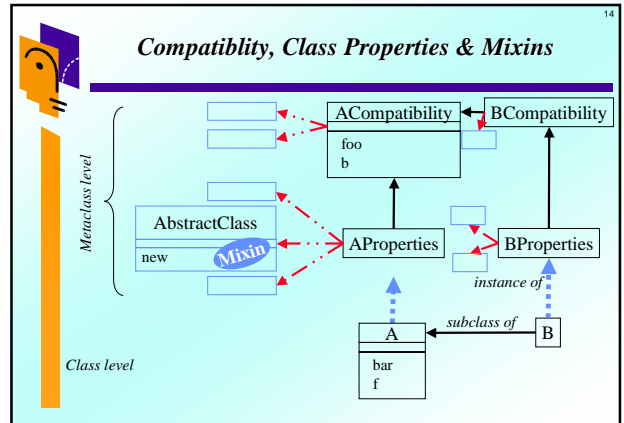
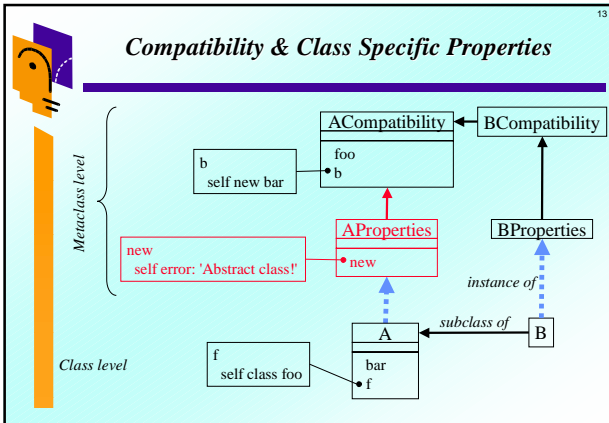
No Class Specific Properties in Smalltalk

Unwanted property propagation B becomes abstract!


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

Class level: A (bar), B (bar)

Instance level: a (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>