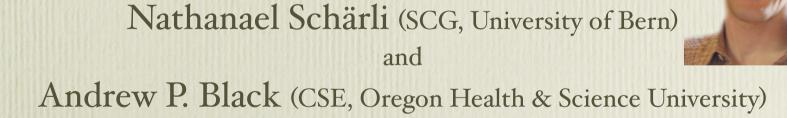
## A Browser for Incremental Programming

### Nathanael Schärli (SCG, University of Bern) and Andrew P. Black (CSE, Oregon Health & Science University)

## A Browser for Incremental Programming



# A Browser for Incremental Programming

### Nathanael Schärli (SCG, University of Bern)

and

Andrew P. Black CSE, Oregon Health & Science University)

me

## What is Incremental Programming?

- What we do in Smalltalk!
- One of the Extreme Programming (XP) practices
- Characterized by some patterns of work that should be familiar to you:

- Programming with limited knowledge
- Working in multiple contexts
- Refactoring
- Testing
- Understanding how classes collaborate
- Understanding what is still missing

- Programming with limited knowledge

   Generic protocols, absence of declarations

   Working in multiple contexts
- Refactoring
- Testing
- Understanding how classes collaborate
- Understanding what is still missing

- Programming with limited knowledge

   Generic protocols, absence of declarations

   Working in multiple contexts

   Multiple windows, tiling browsers
- Refactoring
- Testing
- Understanding how classes collaborate
- Understanding what is still missing

- Programming with limited knowledge

   Generic protocols, absence of declarations

   Working in multiple contexts
  - Multiple windows, tiling browsers
- Refactoring
  - Refactoring Browser
- Testing
- Understanding how classes collaborate
- Understanding what is still missing

- Programming with limited knowledge – Generic protocols, absence of declarations
- Working in multiple contexts – Multiple windows, tiling browsers
- Refactoring
  - Refactoring Browser
- Testing
  - SUnit
- Understanding how classes collaborate
- Understanding what is still missing

- Programming with limited knowledge – Generic protocols, absence of declarations
- Working in multiple contexts – Multiple windows, tiling browsers
- Refactoring
  - Refactoring Browser
- Testing
  - SUnit
- Understanding how classes collaborate
   -?
- Understanding what is still missing

- Programming with limited knowledge – Generic protocols, absence of declarations
- Working in multiple contexts

   Multiple windows, tiling browsers
- Refactoring
  - Refactoring Browser
- Testing
  - SUnit

-?

- Understanding how classes collaborate
   -?
- Understanding what is still missing

### How should we support Incremental Programming?

- Provide information about completeness of classes and collaborations between classes that is
  - statically computed,
  - always accessible, and
  - always up-to-date
- Why not? This information is in the code
  - My PowerBook is 50 times faster than a Dorado

• Virtual Categories

### • Virtual Categories

X 🖼 Traits Browser: Boolean										
Kernel-Objects		Boolean 💻			-		all	-	å	-
Kernel-Classes		DependentsArray 🍧					logical operations	-	and:	-
Kernel-Methods		Even	EventMessageSet				controlling		ifFalse:	
Kernel-Processes		Even	EventModel				copying		ifFalse:ifTrue:	
Kernel-Magnitudes		False					printing		ifTrue:	
Kernel-Numbers		MessageSend					misc		ifTrue:ifFalse:	
Kernel-ST80 Remnar	4	Model					-requires-		not	
TraitsPaperExample		Morp	MorphObjectOut				-overrides-		or:	
Collections-Abstract		Ohiec	Object 💌							
Collections-Unordere	-	<-	inst.	?	class			-		-
									·	_
(-own-)									[ -su]	per- ]

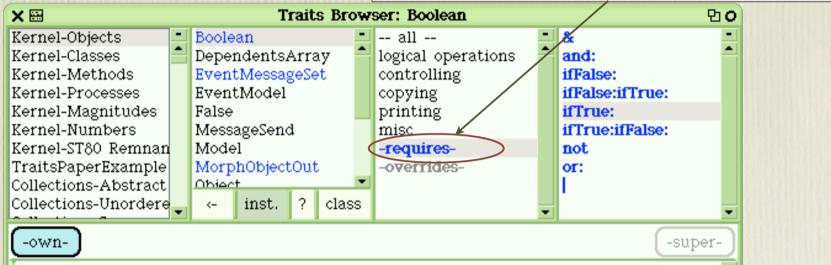
#### ifTrue: alternativeBlock

"If the receiver is false (i.e., the condition is false), then the value is the false alternative, which is nil. Otherwise answer the result of evaluating the argument, alternativeBlock. Create an error notification if the receiver is nonBoolean. Execution does not actually reach here because the expression is compiled in-line."

self subclassResponsibility

### • Virtual Categories

categorization of methods by the browser, based on their characteristics; always up-to-date



#### ifTrue: alternativeBlock

"If the receiver is false (i.e., the condition is false), then the value is the false alternative, which is nil. Otherwise answer the result of evaluating the argument, alternativeBlock. Create an error notification if the receiver is nonBoolean. Execution does not actually reach here because the expression is compiled in-line."

self subclassResponsibility

## Four Categories:

### -requires-

• all messages sent to this class for which there is no method defined or inherited

-supplies-

• all messages required by some other class for which methods are provided in this class

-overrides-

methods defined in this class that override inherited methods

-sending super-

• methods that perform super sends

X 🖽 Traits Browser: Boolean											
Kernel-Objects		Boole	an		-	•	all	-	å	-	
Kernel-Classes		Deper	ndents,	Arra	y 🕯		logical operations	-	and:	-	
Kernel-Methods		Event	EventMessageSet				controlling		ifFalse:		
Kernel-Processes		EventModel					copying		ifFalse:ifTrue:		
Kernel-Magnitudes		False					printing		ifTrue:		
Kernel-Numbers		Messa	MessageSend				misc		ifTrue:ifFalse:		
Kernel-ST80 Remnar	1	Mode	Model				-requires-		not		
TraitsPaperExample		Morpi	MorphObjectOut				-overrides-		or:		
Collections-Abstract		Ohiect	F		-	-					
Collections-Unordere	-	<-	inst.	?	class			-		-	
								-		_	
[-own-]									-sup	er-	

#### ifTrue: alternativeBlock

"If the receiver is false (i.e., the condition is false), then the value is the false alternative, which is nil. Otherwise answer the result of evaluating the argument, alternativeBlock. Create an error notification if the receiver is nonBoolean. Execution does not actually reach here because the expression is compiled in-line."

self subclassResponsibility

🗙 🖼 🛛 Traits Browser: True 🛛 🗠 😋											
Kernel-Objects 🛛	Morp	hObject	tOut	-	all	-	Å. 💻				
Kernel-Classes	Objec	:t		-	logical operations	-	and:				
Kernel-Methods	Objec	ObjectOut			controlling		ifFalse:				
Kernel-Processes	Objec	ObjectTracer			printing		ifFalse:ifTrue:				
Kernel-Magnitudes	Objec	:tViewe	≥r		-supplies-		ifTrue:				
Kernel-Numbers	Prote	ProtoObject			-overrides-		ifTrue:ifFalse:				
Kernel-ST80 Remnan	True						not				
TraitsPaperExample	Unde	finedOt	oject				or:				
Collections-Abstract											
Collections-Unordere	<-	<- inst. ? class				-	-				
-own-											
ifTrue: alternativeBlock											
"Answer the value of alternativeBlock. Execution does not actually											
reach here because the expression is compiled in-line."											

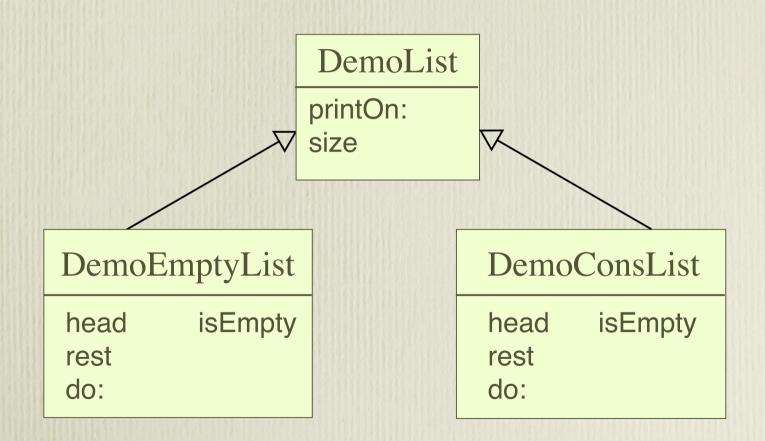
↑alternativeBlock value

🗙 🖼 🛛 Traits Browser: True 🛛 🖸 O											
Kernel-Objects Kernel-Classes Kernel-Methods Kernel-Processes Kernel-Magnitudes Kernel-Mumbers Kernel-ST80 Remnan TraitsPaperExample Collections-Abstract Collections-Unordere	Object Object Object Object Proto True		r		all logical operations controlling printing - <b>supplies</b> - - <b>overrides</b> -	•	printOn:	•			
-own- printOn: aStream aStream nextPutA	All: 'tru	le'			P		P	-super-			

🗙 🖼 🛛 Traits Browser: True 🛛 🖸 O											
Kernel-Objects Kernel-Classes Kernel-Methods Kernel-Processes Kernel-Magnitudes Kernel-Numbers Kernel-ST&O Remnan TraitsPaperExample Collections-Abstract Collections-Unordere	MorphObjectOut Object ObjectOut ObjectTracer ObjectViewer ProtoObject True UndefinedObject <- inst. ? ci	- - - -	all logical operations controlling printing <b>-supplies-</b> -overrides-	-	printOn:						
-own- Object» printOn: aStread "Append to the argonidentifies the recent the recent of self printNameOn	gument, aStream, : iver."	a sec	uence of characters	th	-super- at						

### Extended Example/Demonstration

• Creating a new sub-tree of classes



### Understanding and Modifying Existing Hierarchies

- Feedback from the browser helps us find:
  - which methods are "core" and which are "support"
  - how the sub- and superclasses depend on each other
- When extending code, the browser helps avoid:
  - introducing inter-level errors
  - accidentally incomplete classes

# The Core/Support Split

- A common pattern used to increase reuse in a data type implementation [Black ECOOP inh wk 2002]
  - *e.g.*, the abstract superclass Collection defines 110 *support* methods
    - they don't access the state of any collection directly
    - instead, they depend on 4 core methods
      - add:, atRandom:, do: and remove: if Absent:
        - 3 are defined as self subclassResponsibility
        - atRandom: is not defined at all
    - The browser finds these 4 required methods amongst the 110!

# The Core/Support Split (2)

- When we look at a subclass of Collection, e.g., Bag, we can distinguish:
  - the 4 supplied methods,
  - 10 methods that override the inherited methods, either to disable them or to improve their efficiency, and
  - 7 additional methods that widen the interface of Bag beyond that of Collection

### Accidentally Abstract Classes

- The browser tells us some surprising things about Squeak's core classes:
  - Fraction is abstract
    - it implements the support method printOn:, whereas it should implement the core method printOn:Base:
  - **Bitmap** is abstract
    - the programmer sends an error message primitiveFail, which he forgot to define
  - **Debugger**, **CharacterSet**, **Morph** (and nearly all of its subclasses): all are abstract

### Accidentally Abstract Classes (2)

- Why are these errors present in a code base that has been used by thousands of users for many years?
  - It is *not* because a bad programmer wrote BitMap
- It is because even good programmers will make mistakes unless they have good tools

### Accidentally Abstract Classes (2)

- Why are these errors present in a code base that has been used by thousands of users for many years?
  - It is *not* because a bad programmer wrote BitMap

 Iap W1 decompress:fromByteArray:at:

 m size
 ar 2/3/2001 16:11

 hileTrt
 what to show...

 next
 browse full (b)

 at
 browse hierarchy (h)

• It is because even good programmers will make mistakes unless they have good tools

### Implementation

### -sending super- is easy

• look for the bytecode for super sends

### -overrides- is easy

- compare this class's selectors with its superclass's protocol
- -supplies- is easy once one knows requires
  - compare this class's selectors with the requires set of other classes
- -requires- is most definitely not easy
  - implementing requires in real-time required a lot of careful thinking and more careful programming!

• Pseudocode:

Behavior >> requires self reachableMethods selfMessages difference: (self allReallyImplementedSelectors)

we had to invent and formalize a definition of reachability

• Pseudocode:

Behavior >> requires

self reachable Methods self Messages

difference:

(self allReallyImplementedSelectors)

• Pseudocode:

Behavior >> requires self reachableMethods selfMessages difference: (self allReallyImplementedSelectors)

cannot infer this from bytecode.

• Pseudocode:

Behavior >> requires

self reachableMethods selfMessages

difference:

(self allReallyImplementedSelectors)

• Pseudocode:

Behavior >> requires self reachableMethods selfMessages difference: (self allReallyImplementedSelectors)

• Pseudocode:

Behavior >> requires

self reachableMethods selfMessages

difference:

(self allReallyImplementedSelectors)

allSelectors \ those not really implemented (subclass-Responsibility, shouldNotImplement, requirement, etc.)

# Recognizing self-sends

- Recognizing self-sends requires a full parse of the method text
- A change in, say, Object, might change the required methods of every class in the system!
- Squeak images contain > 60 000 methods
- We decided that we needed to cache the selfsends for every method when it is compiled

#### A problem of scale

• Even with these caches for self- and super sends, the first implementation took over 3 minutes to ascertain the required methods of a class!

# Two key insights

- The caches should be arranged "backwards"
  - for each message, cache the methods that self-send it
- We don't need to know the requires set, all we need to know is whether it is empty
  - Does a subclass override *all* of the methods that selfsend a message required by the superclass?
    - if not, we *immediately* know that it is also required in the subclass

## The Complete Algorithm...

- is far too complex to put on a slide
  - that's what the paper is for!
- Computing the required set now takes less than 100 ms

   fast enough to provide
   "real-time" feedback

## The Complete Algorithm...

- is far too complex to put on a slide
  - that's what the paper is for!
- Computing the required set now takes less than 100 ms — fast enough to provide "real-time" feedback

	cremental Programming
for IN	crementar
A Browser for A	
	Low P. Black
1 50	härli and Andrew V. n Group, University of Bern, Switzerland nerring, Oregon Health and Science University, USA seering, Oregon Health and Science University.
Nathanael Sc	härft attes zw. n Group, University of Bern, Switzerland n Group, University of Bern, Switzerland seering, Oregon Health and Science University, USA seering, Oregon Health and Science University, USA
- nume Compositio	n Group. Oregon Health and Science
Software & Engli	mering, 0110
OGI School of the	
Abstract Much of the elegance and power of First introduced more than 20 year on particular methods using a hier sification scheme says a lot about sification scheme says a lot about	Stallishk some form its programming environment also some of some for Smallask browser endsternes byte name, to taxo den andre of analysis and andre of the source of the source of the source of the source of the source of the source of the source and and source of the source of the source of the source and the source of the source of the source of the source and commonly improve and emanates for dependent used to and commonly sources.

> The most important of the Smalltalk programming tools is the Brows most important of the Smallash programming toole is the *Browser* which as the programmer to examine, modely and extend the code of applications and by system itself. The Browser was revolutionary when it was first introduced, we are the interference of the system 1 Introduction n usen. Lue prowser was revonationary when it was nist introduced, intervening years it has been improved in several ways. For exame intervening years it has been improved in several ways, for exam-tantated reflectoring has been added, leading to a tool known as the nated relacioning has oven auded, reading to a loost known as the wser [10], and in many Smalltalk dialects some form of package proviser (10), and in many smallialik unalects some form of package subsumed the original primitive categorization of classes. However, se has sursame ane original primaire caregorization of casees. Frowever, i in which today's browser organizes the methods of a class is essentially the ау на чинса ималу з инимост игданиzes цие metnons of a crass из as it was in 1980: a hierarchy of manually assigned "protocols" the concept of the "Integrated Development Environment"-for In the meanine, the concept of the "Integrated Development Environment"—(rot that is what the Smallialk toolnet would note be called – has proved to be so suc-courds the simultaneous have have been ensured for robust recommension have hat is what the Smalltalk toolser would now be called—has proved to be so starting lan-earding that similar environments have been created for other programming lan-neares for another that is the starting of the term of the second starting of the starting of the second starting of the starting of th cessful that similar environments have been created for other programming lan-guages. For example, TBM's VisualAge for Java [7] was essentially a re-targeting of Romin address: ochaerlisian.unibe.ch., blackscese.ogi.edu (Nathamut Scharh Markers P. Black)

and Andrew P. Black). Proprint submitted to ESU/G 2003 Academic Track 12 August 2003

### **Related Work**

- To do lists
  - Trellis's "grass catcher" was also the product of changing a single method
  - More commonly, as with Eclipse's "Tasks" window, to do lists are updated only on global recompilation.
- Browser extensions
  - decoration of names to indicate *local* properties such as overrides or <u>sends to super</u>, *e.g.*, in VisualWorks
  - Star Browser allows the definition of intentional classifications that are recomputed when necessary

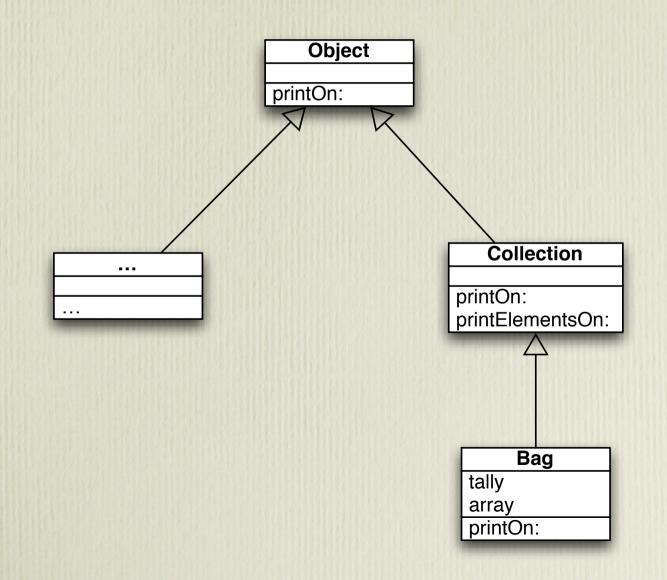
#### Future Work

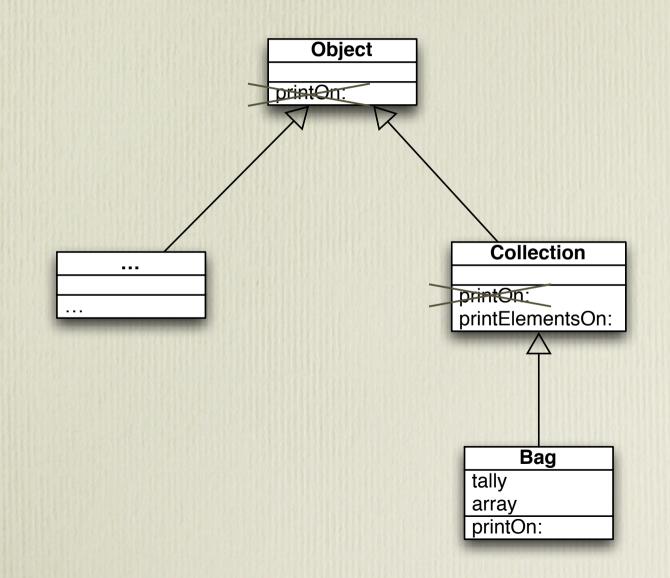
- Other visualizations of the self-send information
  - e.g., Blueprint-like diagrams
- Two directions for extension:
  - Help in understanding other kinds of collaboration
    - e.g., delegation, aggregation, Mudpie's package dependencies
  - A pluggable browser framework
    - what are the key features?

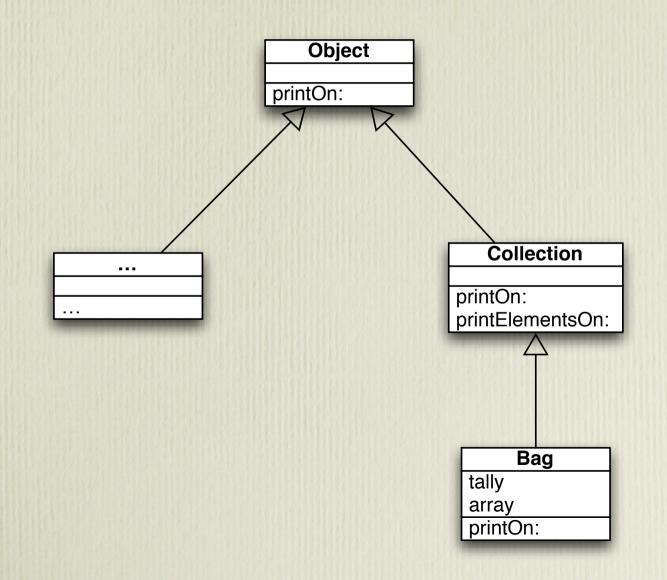
#### Conclusion

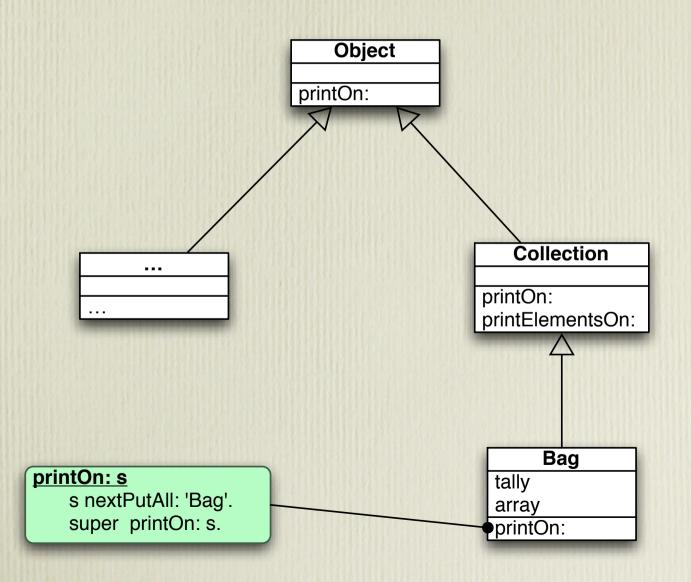
- The Browser is Feasible
  - with careful design and implementation, it *is* feasible to provide real-time feedback even for global properties such as required methods
- The Browser is Useful
  - Simplifies Intentional Programming
  - Makes it easier to understanding existing classes
  - Clarifies the relationship between sub- and superclasses
  - Exposes many bugs in existing code

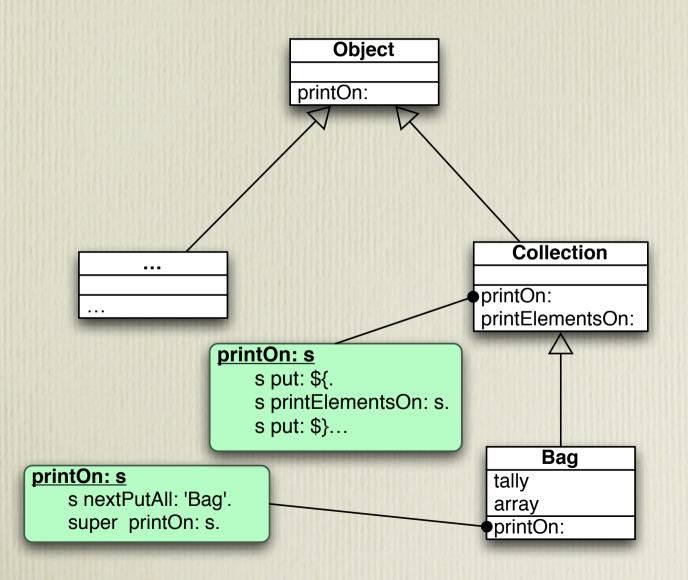




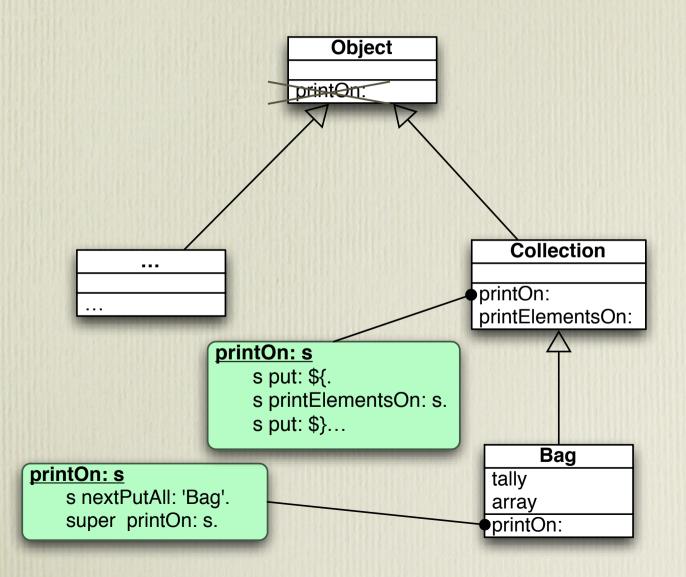


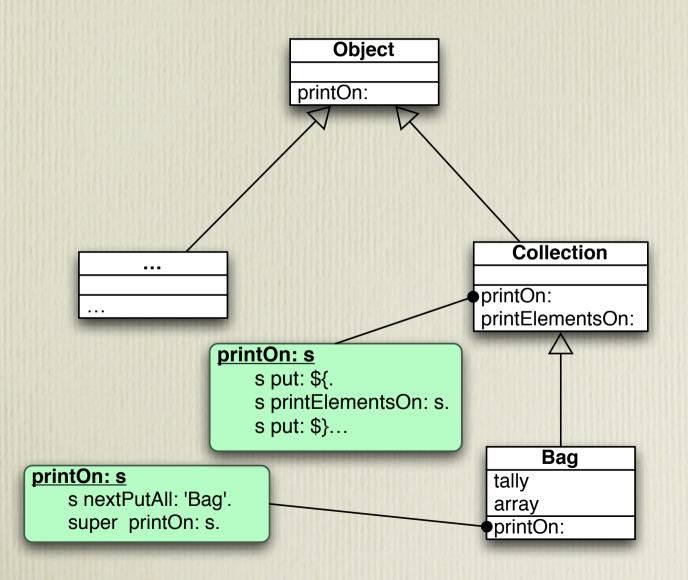




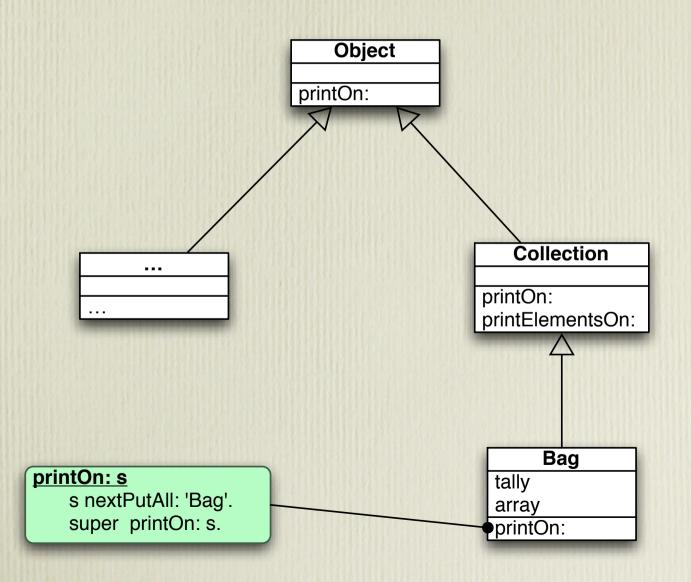


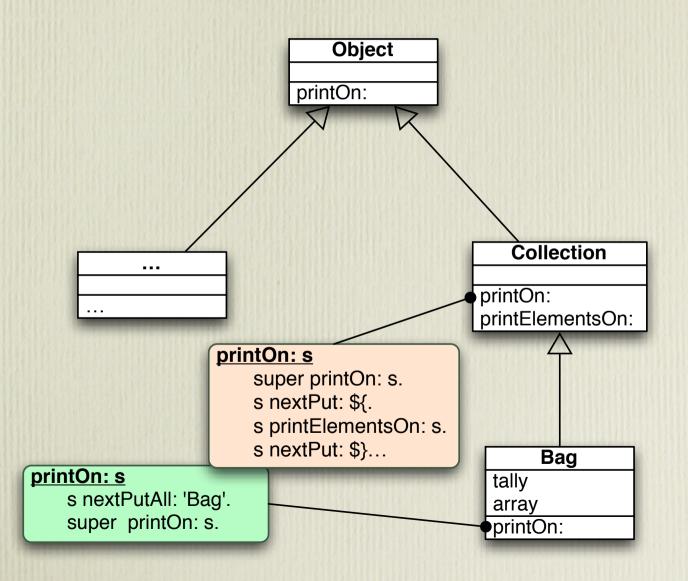
24

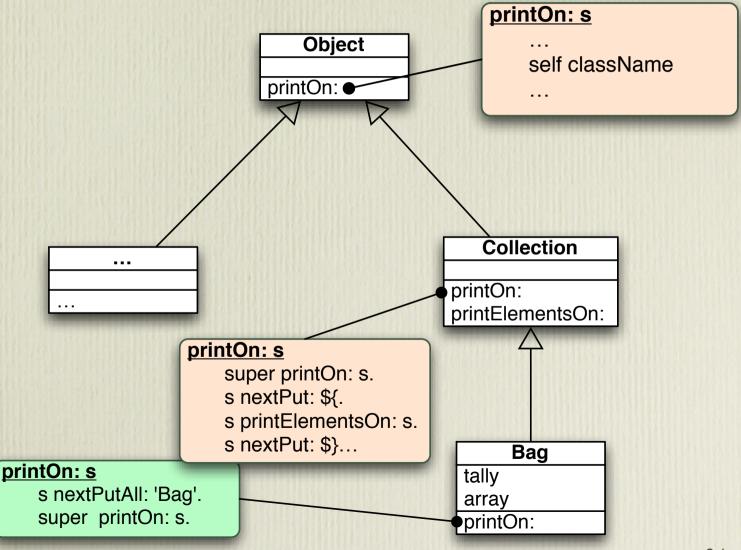




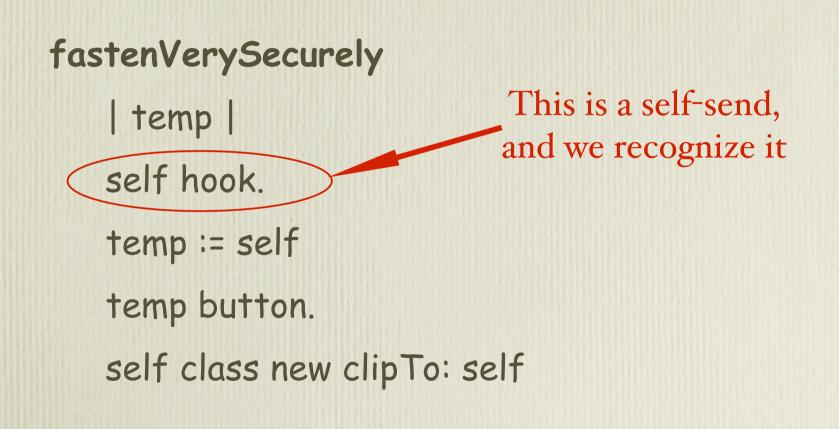
24







fastenVerySecurely | temp | self hook. temp := self temp button. self class new clipTo: self



fastenVerySecurely | temp | self hook. temp := self temp button. self class new clipTo: self

fastenVerySecurely temp self hook. This is a self-send, but temp := self we don't recognize it temp button. self class new clipTo: self

