Objects and Design in Smalltalk

Smalltalk Basics: Collections

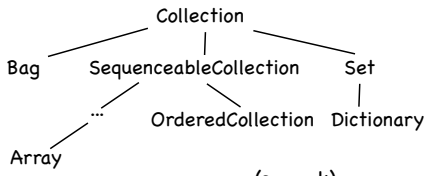
Éric Tanter
etanter@dcc.uchile.cl





Containers

- Indispensable everyday objects
  - the patterns that occur among software objects should reflect the ones we observe among everyday objects
- For a programmer
  - queues, stacks, arrays, sets, etc.
  - aka. collections



(squeak)

Some Container Classes

Class	Characteristics
Array	fixed size, consecutive slots
Bag	unordered, possibly duplicate
Set	unordered, no duplicate
Dictionary	lookup key -> element
OrderedCollection	variable size array
String	0 or more characters
ByteArray	array of bytes
SortedCollection	ordered collection + sorting criteria

Heterogeneity

- A typical Smalltalk container does not check that all its content is the same type
  - attractive but dangerous

```
OC:= OrderedCollection new.
OC add: 'a' ; add: 'c' ; add: Whale new.
OC ~> an OrderedCollection ('a' 'c' a Whale)
```

```
P := 't'.
OC do: [ :elem | P := elem , P ]
```


Sorted Collection

```
X := SortedCollection new.
X add: 3 ; add: 2 ; add: 5.
X ~> a SortedCollection(2 3 5)
```

- What defines the order?
  - a sort block that defines <=

```
X sortBlock: [ :a :b | a >= b ].
X ~> a SortedCollection(5 3 2)
```

- What about heterogeneity?
  - any added element should be comparable to the first

Dictionaries

- Dictionary
  - X at: <key> put: <value>
  - X at: <key> -> <value>
- Dictionary and IdentityDictionary
  - Dictionary uses equality test (=)
  - IdentityDictionary uses identity test (==)
- == is not late-bound
  - otherwise object identity could be corrupted