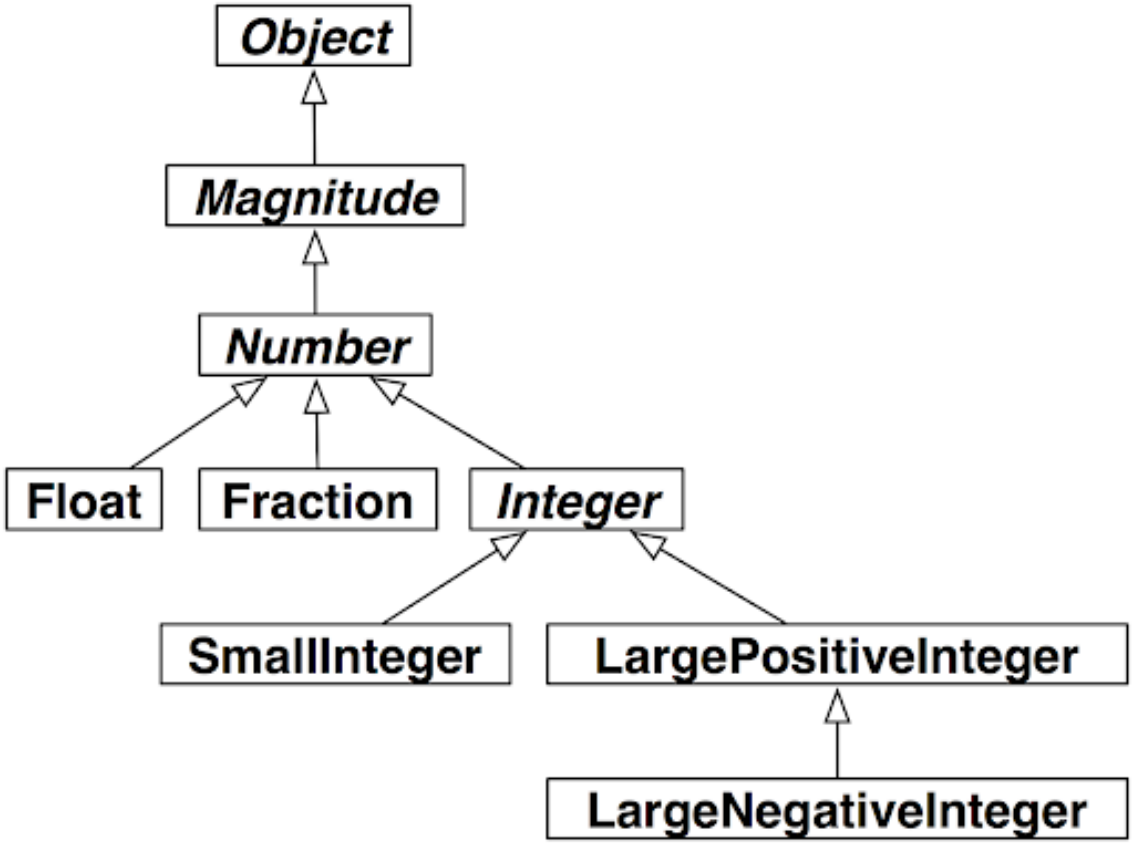
Smalltalk

- Object-oriented
- Dynamically typed
- Reflective

Everything is an object



GNU Smalltalk

- `ssh username@cs-parallel.ua.edu`
- Interactive Mode
- `gst`

```
st> 4+5
9
st> 4+5*6
54
st > ObjectMemory quit
```

- Or type in <CTRL-D> to quit

Literals

- Numbers: 12, 23.4
- Characters: $a, $A
- Strings: ‘This is a string’
- Symbols (strings used for names): #foo
- Arrays: #(‘three’ ‘four’ 5 6 $Z)
- Blocks: [:x | x + 2]

Naming

- A sequence of letters and digits beginning with a letter.
- Global variables, class variables, pool dictionaries, and class names should start with an uppercase letter. Instance variables, methods, block arguments, and temporary variables start with a lowercase letter.
- Only six "keywords" are reserved: `true, false, nil, self, super, thisContext`

Variables

- Temporary variables
- Instance variables
- Class variables
- Pool variables: A pool variable is a variable whose scope is a defined subset of classes.
- Global variables (`Smalltalk` dictionary)

Expression

- A variable name
- A literal
- A message expression
- A block expression

Messages

- Unary, keyword, and binary messages.
- Message chaining, evaluated from left to right
- Unary messages take precedence over binary messages.
- Binary messages take precedence over keyword messages.
- All binary messges have the same precedence.
- Parentheses changes the precedence.

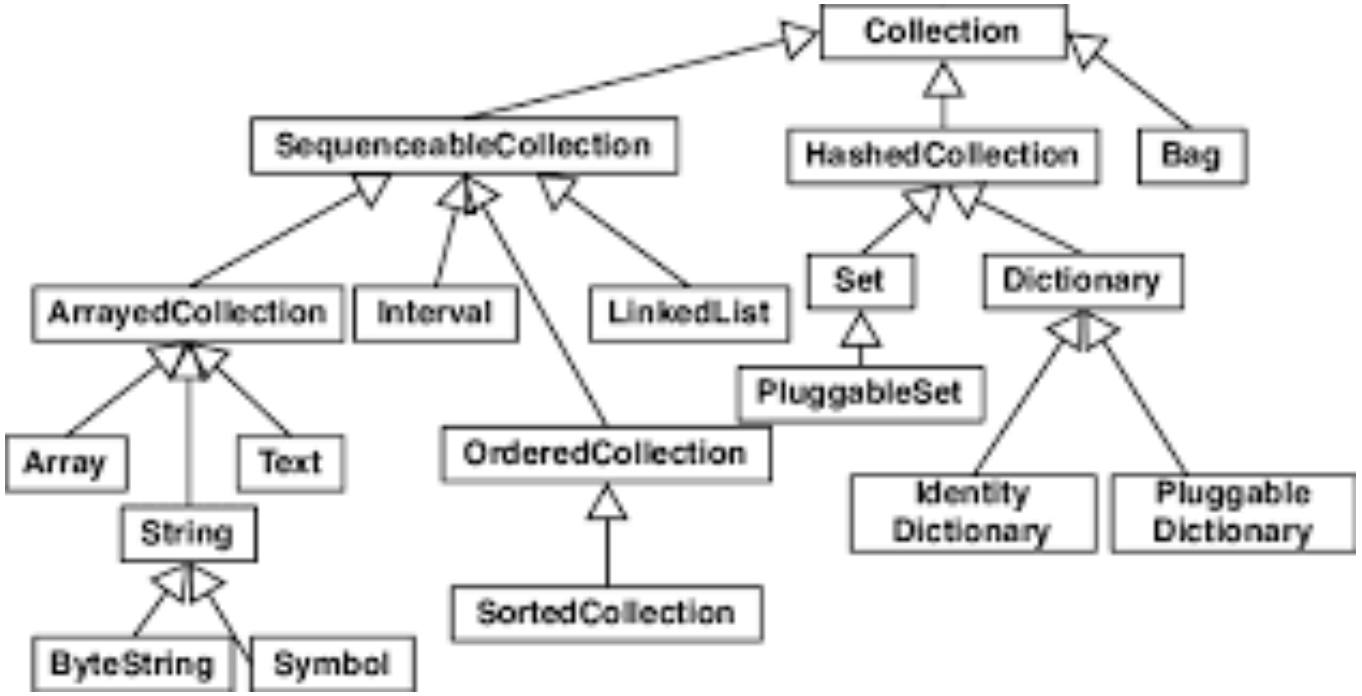
Assignment Statement

- variable := expression
- quantity := 19
- index := initialIndex
- index := index + 1
- y := quantity sqrt
- z := 1 + 2 * 3
- f := [:x | x+1]

Collections

- Arrays
- Sets
- Dictionary

Collection Class Hierarchy



Array

- x := Array new: 20
- x at: 1
- x at: 1 put: 99

Set

- x := Set new
- x add: 5. x add: 7. x add: 'foo’
- x add: 5; add: 7; add: 'foo’

  (message cascading)
- x remove: 5
- x includes: 7

Dictionary

- `y := Dictionary new`
- `y at: 'name' put: 'John Smith'`
- `y at: 'age' put: 25`
- `y at: 'name’`
- `y at: 'age'`

Control Structures

- Selection
- Iteration

Selection

- *aBoolean* ifTrue: *aBlock*
  evaluates aBlock if aBoolean is true
- *aBoolean* ifFalse: *aBlock*
  evaluates aBlock if aBoolean is false
- *aBoolean* ifTrue:*trueBlock* ifFalse: *falseBlock*
  evaluates trueBlock if aBoolean is true, falseBlock if false
- *aBoolean* ifFalse: *falseBlock* ifTrue:*trueBlock*
  evaluates trueBlock if aBoolean is true, falseBlock if false

Selection Examples

- index <= limit

  ifTrue: [total := total + (list at: index)]

- index <= limit

  ifTrue: [total := total + (list at: index)]

  ifFalse: []

Interation

- *aBooleanBlock* whileTrue: *loopBlock*
  as long as aBooleanBlock evaluates to true, loopBlock is evaluated
- *aBooleanBlock* whileFalse: *loopBlock*
  as long as aBooleanBlock evaluates to false, loopBlock is evaluated
- *aBooleanBlock* whileTrue
  repeats evaluating aBooleanBlock until it returns false
- *aBooleanBlock* whileFalse
  repeats evaluating aBooleanBlock until it returns true

Iteration Example

- ```
  index := 1.
   [index <= list size]
   whileTrue: [list at: index put: 0.
               index := index + 1]
  ```
  ```

Repeating multiple times

- ```
  n timesRepeat:[
      ...
      repeated statements
      ...
  ]
  ```
- `5 timesRepeat:[ 100 printNl ]`
  ```

Interval and Iteration

- ```
  |anArray|
  anArray := #( 'one' 'deux' 'drei' 'quatro' 5 6.0 ).

  1 to: 6 do: [:idx | (anArray at: idx) printNl].
  ```
  ```

Collection and Iteration

- *aCollection* do: *aOneArgBlock*
-
```
|anArray|
   anArray := #( 'one' 'deux' 'drei'
'quatro' 5 6.0 ).

   anArray do:[:eachElement | eachElement
printNl ].
```

Collection Methods

- detect

```
#( 4 7 10 3 17) detect: [ :each | each > 7]
```

- select, reject

```
'now is the time' select: [ :each | each
isVowel ]
```

- collect

```
#( 1 2 3 4 5 ) collect: [:i | i * i ]
```

Non-interactive Mode

```
"first.st"
"A program to print 'Hello World!' to the
terminal."

'Hello World!' printNl
```

- gst first.st

References

- Bluebook: http://stephane.ducasse.free.fr/FreeBooks/BlueBook/Bluebook.pdf
- GNU Smalltalk User's Guide: https://www.gnu.org/software/smalltalk/manual/gst.html
- GNU Smalltalk Library Reference: https://www.gnu.org/software/smalltalk/manual-base/gst-base.html