Jumping Into Smalltalk

I hear from many Java and C# programmers that they can't read Smalltalk code. Smalltalk is actually very easy to learn since it has very little syntax. The true power of Smalltalk lies in the environment and the class libraries.

This article is intended to give Java and C# programmers an extremely rapid plunge into Smalltalk. I'll do that by using side-by-side code comparisons.

Temporary Variables

Smalltalk doesn't require type declarations on any variables. Temporary variables are defined within vertical bars.

Java	Smalltalk

int a;
char b;
float c;
	| a b c |

Assignment

Smalltalk uses := for assignment.

Java	Smalltalk
a = 5;	a := 5

Messages

Smalltalk has three kinds of messages.

Type	Form	Parameters	Examples
unary	alphanumeric starting with a lowercase letter	0	squared

binary	punctuation marks	1	+
keyword	multiple colon terminated alphanumeric words	1 or more	do: between: and:

To pass one or more parameters, you would usually use a keyword message. Each parameter is preceded by a keyword. Smalltalk doesn't use brackets and commas to separate the parameters.

Examples

Java	**Smalltalk**
myAccount.getBalance();	myAccount getBalance
myAccount.setBalance (10);	myAccount setBalance: 10
myAccount.transfer (20, anotherAccount)	myAccount transfer: 20 to: anotherAccount
myAccount.equals (anotherAccount);	myAccount = anotherAccount

Order of operations is:

Unary (evaluate first)
Binary (evaluate second)
Keyword (evaluate last)

Within each priority level, evaluate left to right.

Java	**Smalltalk**
3 + 5 * 6 // answer: 33	3 + 5 * 6 "answer: 48"

Notice that in the Smalltalk version, this expression is actually two messages:

Message 1
  receiver: 3
  message: +
  parameter: 5

result: 8

Message 2
receiver: 8
message: *
parameter: 6

result: 48

Statements

Smalltalk uses a period (.) as a statement separator. You don't need a period on the last statement.

Java	Smalltalk

myAccount.deposit(20);
myAccount.transfer(20, anotherAccount);
	myAccount deposit: 20.
	myAccount transfer: 20 to: anotherAccount

Literals

In Smalltalk, integers, characters, strings, booleans, floats and doubles are all first class objects. Integers are infinite precision and automatically grow as needed without overflow. As such, there's no equivalent to char, byte, short, int, or long. They're all just integers.

Java	Smalltalk
5	5
01230	8r1230
0x7f	16r7f
<no equivalent>	3r21012 (you can use any base you like)
200L	<no equivalent>
2e-5	2e-5
2e-5d	2d-5
'h'	$h

'\u03A9'	Character value: 16r3A9
"hello"	'hello'
"can't"	'can''t'
{"a","b","c"}	#('a' 'b' 'c')
<no equivalent>	#($a 234 #hello)

Special Words

In Smalltalk, nil refers to a real object. It's an instance of the class UndefinedObject. The word true refers to an instance of the class True and false refers to an instance of the class False.

Java	**Smalltalk**
this	self
null	nil
true	true
false	false
super base (C#)	super

Returning From Methods

Java	**Smalltalk**
return value;	^value

Cascades

Smalltalk uses a semicolon (;) to separate multiple messages sent to the same object.

Java	**Smalltalk**

<no equivalent>	myAccount
	  deposit: 20;
	  transfer: 20 to: anotherAccount

Comments

Java	Smalltalk
/* comment */ // another comment	"comment"

Instance Creation

In Smalltalk, classes are real objects. To create an instance, just send new to the class. Methods for a class are called class methods (similar to Java static methods).

Java	Smalltalk
new Reservation();	Reservation new

Constructors

Smalltalk has no constructors. If you want to perform instance initialization, you can redefine the "new" class method to initialize the instance.

Java	Smalltalk

Reservation()
{
  startTime =
    new GregorianCalendar().getTime();
  endTime =
    new GregorianCalendar().getTime();
}
	Reservation class method:
	new
	  ^super new initialize
	
	Reservation instance method:
	initialize
	  startTime := Timestamp now.
	  endTime := Timestamp now

Methods

Java	Smalltalk

class Room {
 	
  void book (DateTime start, DateTime end)
  {
    reservations.add (
      new Reservation(start, end));
  }
 	
}
	bookFrom: startTime to: endTime
	  reservations add:
	    (Reservation from: startTime to: endTime)

Blocks

Smalltalk has an object called a block. It's an object that contains executable code. The closest thing Java has is an anonymous inner class. In C# 2.0, there are anonymous delegates that are similar.

To execute a block with no parameters, you send it a **value** message.

Smalltalk
| block | block := [3 + 4]. block value   "answer is 7"

Blocks can have parameters. Each block parameter declaration starts with a colon (:). A vertical bar (|) denotes the end of the parameter list and the start of the code for the block.

Smalltalk
| block | block := [:x :y | x * 2 + y]. block value: 5 value: 3   "answer is 13"

End of the Syntax

At this point, we've covered all of the syntax of Smalltalk. Everything else is part of the class library. Have you noticed anything missing? How about if-then-else or while loops? Smalltalk just uses blocks and ordinary message sends.

Control Structures

Smalltalk has no control structures like if built into the language. Instead, Smalltalk uses messages sent to the true or false objects.

Java	Smalltalk

if (tries > 5)
  return "Too many tries";
else
  return "Trying again";
	tries > 5
	  ifTrue: [^'Too many tries']
	  ifFalse: [^'Trying again']

Notice that the ^ returns from the method, not just the block.

Loops

Smalltalk uses blocks to do looping. Since blocks are just objects, we can send messages to them.

Java	Smalltalk

int tries = 0;
while (tries <= 5) {
  tryAgain();
  tries++;
  }
	| tries |
	tries := 0.
	[tries <= 5] whileTrue: [
	  self tryAgain.
	  tries := tries + 1]
	
	alternatively:
	
	5 timesRepeat: [self tryAgain]

Notice that timesRepeat: is a message understood by Integers. It simply evaluates the block the proper number of times.

Things left to learn

This concludes this quick introduction to Smalltalk. There are still many things to learn but there is no more syntax. Everything else is part of the class library.

If you'd like to learn more, download Smalltalk try it out. Wilf Lalonde has a good article on learning Smalltalk for Java and C++ programmers. For lots of Smalltalk resources including links to online Smalltalk books, visit the Why Smalltalk site. If you'd be interested in formal training contact Simberon and ask about Smalltalk training