# **Applications**

Since its creation in the early 1980s, Smalltalk has been widely used in academic research as well as in commercial applications. Here are some current Smalltalk applications that are advancing the state of software.

**Teaching:** EToys (Squeak), SqueakBot, BotsInc, Scratch...

Multimedia: Sophie, OpenCroquet, Plopp...

Web development: Seaside, Aida, Komanche, Swazoo...

Persistence management: object oriented databases (Magma, GemStone), relational databases (MySQL, PostgreSQL), object relational mapping (Glorp).



Etoys and DrGeo on an OLPC

# Glossary

Image: The Smalltalk environment contains a persistent object store, the image. This contains application code (classes and methods), objects holding application state and can even include the development tools to inspect and debug the program while it is executing.

**Virtual Machine:** A virtual machine is a program which is capable of executing other programs. It eases application portability.

**Reflection:** A language is said to be reflective when it contains mechanisms to inspect and modify code during a program execution.

**Dynamic typing:** Some languages force the developer to indicate the type of each variable (integer, string...); this is called static typing. Dynamic typing does not impose this constraint, and so makes programs more reusable and easier to change.

#### Books

 Numerous free books: http://stephane.ducasse.free.fr/FreeBooks. html

- Smalltalk in general
  - Smalltalk with Style (Edward Klimas, Suzanne Skublics and David A. Thomas, free)
  - Smalltalk by Example: the Developer's Guide (Alec Sharp, free)
- Squeak in particular
  - Squeak by Example (2007, free)
  - Powerful Ideas in the Classroom (BJ Allen-Conn and Kim Rose)

#### **Events**

• European Smalltalk User Group conferences (ESUG). Since 1993, industrial and academic Smalltalkers meet in an European country.

http://www.esug.org/conferences

 Annual conference, organised in North America by the STIC (http://www.stic.st), an association with industrial actors and Smalltalk editors.

http://www.smalltalksolutions.com/

#### Internet

Official Squeak website: http://www.squeak.org

• Wiki:

http://wiki.squeak.org

News:

http://news.squeak.org

# Smalltalk

a programming language purely **object oriented** and a **dynamic** environment



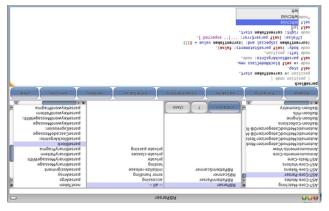
# Development environment

Most Smalltalk implementations come with an integrated development environment that allows you to browse the source code and to interact with objects. Lots of tools are available, all implemented in Smalltalk thanks to a reflection API:

- a class and method browser;
- refactoring tools;
- object inspectors;

and much, much more!

- a debugger;
- release management and version control tools;
- Code can be inspected and evaluated directly in the image, using simple key combinations and full menus.



The Pharo code browser

## **enoitatnemelqml**

There are a number of Smallfalk implementations available: Squeak & Pharo: free, open-source and multi-platform implementations. Actively developed.

VisualWorks: proprietary and multi-platform, freely available for non commercial use.

Gemstone: proprietary implementation which includes an highly

efficient object database. And others: GNU Smalltalk, Smalltalk, Sughin...

```
array := Array new.
array size.
```

The first example creates and returns a new instance of the Array class, by sending it the message  $\pi ew$ . The second example asks for the size of this array which returns 0.

**Binary messages.** A binary message takes only one argument, is named by a symbol and is often used for arithmetic expressions.

```
. Hello', ' World'.
3 + 4.
```

The + message is sent to the object 3 with 4 as a parameter. In the second case, the message , is sent to the string 'Hello' with ' World' as a parameter.

**Keyword messages.** A keyword message can take one or more arguments. The arguments are inserted between each keyword, after each colon.

```
'Smalltalk' allButFirst: 5.
3 to: 10 by: 2.
```

The first example calls the method allButFirst: on a string and pass the argument 5. The method returns the string 'talk'. The second example returns a collection containing elements 3, 5, 7 and 9.

#### Block

Blocks are objects containing code that is not executed immediately. They are the basis for control structures like conditionals or loops. Also, blocks can be used to attach behavior, e.g., to menu items.

```
#('Hello ' 'World')
#('Hello ' 'World')
```

The example sends the message do: to an array of strings with a block as a parameter. The block is evaluated once for each element in the array. The block parameter strings contains of the whole expression, the strings 'Hello ' then 'World' are displayed in the transcript.

## Smalltalk important concepts

Smalltalk is an object-oriented, dynamically-typed language, with a simple syntax which can be learnt in fifteen minutes. Its main advantage comes from the fact that it is very consistent:

- everything is an object: classes, methods, numbers, etc.
- a small number of rules, and no exceptions!

Smalltalk runs in a *virtual machine*. Development takes place in an *image* in which all objects live and are modified.

## Smalltalk syntax

a autocacio	~φ
declaration of three temporary variables	var1 var2
return a result from a method	(† 10) ^
tnəmnpizza	(→ 10) =:
Reserved characters	
run-time stack of the current method	txətmoSzidt
current object in the super class context	anber
toejdo tnemu	llea
boolean objects	true and false
undefined object (default variable values)	Ţţu
	Reserved words

```
#(abc 123)
#(abc 123)
# (abc 123)
# (abc 123)
# ferminate expressions
# ferminate expressions
# message cascade

| code block (it's an object !)
| code block (it's an object !)
```

### Message sending

' gnirte '

A method is called by sending a message to an object, the message receiver; the message returns an object, a verb and is based on natural language, having a subject, a verb and complements. There are three message types: unary, binary and keyword.

Unary messages. A unary message is one with no arguments.