Implementacions

Existeixen diverses implementacions disponibles:

Squeak: gratuïta, open-source i multi-plataforma. Desenvolupada activament per una comunitat internacional.

VisualWorks: propietària i multi-plataforma, disponible gratuïtament per a ús no comercial.

Gemstone: propietària, inclou una base de dades d'objectes molt eficient.

I altres: GNU Smalltalk, Smalltalk/X, SyX, VA Smalltalk, Dolphin...

Aplicacions

Des de la seva creació a principis dels 80s, Smalltalk ha estat extensament utilitzat tant en recerca com en el món comercial. Exemples actuals d'aplicacions Smalltalk que contribueixen a fer avançar la tecnologia del programari.

Docència: EToys (Squeak), SqueakBot, BotsInc, Scratch...

Multimèdia: Sophie, OpenCroquet, Plopp...

Desenvolupament web: Seaside, Aida, Komanche, Swazoo...

Gestió de la persistència: bases de dades orientades a objectes (Magma, GemStone), bases de dades relacionals (MySQL, PostgreSQL), correspondència entre objectes i relacions (Glorp).

Una sessió de dibuix amb Plopp

Glossari

Imatge: L'entorn Smalltalk proporciona un magatzem persistent d'objectes, la imatge. Aquesta conté el codi de les aplicacions (classes i mètodes), objectes que mantenen l'estat de l'aplicació i fins i tot pot incloure eines de desenvolupament per inspeccionar i depurar un programa mentre s'executa.

Màquina Virtual: Una màquina virtual és un programa que és capaç d'executar altres programes. Facilita la portabilitat d'aplicacions.

Reflexió: Un llenguatge es diu que és reflexiu quan conté els mecanismes per inspeccionar i modificar el codi mentre s'està executant el programa corresponent.

Smalltalk

un llenguatge de programació
purament orientat a objectes
i un entorn dinàmic

EUROPEAN SMALLTALK USER GROUP
esug
www.esug.org

Conceptes Importants d'Smalltalk

Smalltalk és un llenguatge orientat a objectes, amb tipat dinàmic i una sintaxi senzilla que es pot aprendre en quinze minuts. El seu principal avantatge és ser molt consistent:

- tot és un objecte: classes, mètodes, nombres, etc.
- un nombre petit de regles, sense excepcions!

Smalltalk s'executa sobre una màquina virtual. El desenvolupament té lloc dins d'una imatge, en la que viuen i poden ser modificats tots els objectes.

Sintaxi d'Smalltalk

Paraules reservades	
`nil`	objecte no definit (valor per defecte)
`true` i `false`	objectes booleans
`self`	objecte receptor del missatge
`super`	objecte receptor del missatge (dins un context de super classe)
`thisContext`	pila d'execució del mètode actual

Caràcters reservats	
`:=` (o ←)	assignació
`^` (o ↑)	retorna un resultat des d'un mètode
`| var1 var2 var3 |`	declaració de tres variables temporals
`$a`	caràcter `a`
`#(abc 123)`	taula (array) que conté dos literals: el símbol `#abc` i el nombre `123`
`.` (punt)	fi d'expressió
`;`	missatges en cascada
`[ ]`	bloc de codi (és un objecte !)
`"comentari"`	
`'cadena'`	

Enviament de missatges

Un mètode és cridat enviant un missatge a un objecte, el receptor del missatge, el missatge retorna un objecte.

El missatge està basat en el llenguatge natural, amb subjecte, verb i complements. Hi ha tres tipus de missatges: unari, binari i paraula clau.

Missatges unaris. Un missatge unari no té arguments

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

El primer exemple crea i retorna una nova instància de la classe Array, enviant el missatge `new`. El segon exemple demana la mida d'aquesta taula (`array`), i retorna `0`.

Missatges binaris. Un missatge binari pren només un argument, el nom és un símbol i s'utilitza sovint per a expressions aritmètiques.

```
3 + 4.
'Hola', ' Món'.
```

El missatge `+` és enviat a l'objecte `3` amb `4` de paràmetre. En el segon cas, el missatge `,` és enviat a la cadena `'Hola'` amb `' Món'` de paràmetre.

Missatges de paraula clau. Un missatge de paraula clau pot prendre un o més arguments. Els arguments s'insereixen entre cada paraula clau, després dels dos punts.

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

El primer exemple crida el mètode `allButFirst:` sobre una cadena de caràcters i amb argument `5`. El mètode retorna la cadena de caràcters `'talk'`. El segon exemple retorna una col·lecció contenint els elements `3`, `5`, `7` i `9`.

Bloc

Els blocs són objectes que contenen codi que no és executat immediatament. Són la base d'estructures de control com els condicionals o les repeticions. Els blocs poden ser utilitzats per associar comportaments, ex. les opcions d'un menú.

```
#('Hola' ' Món')
       do: [:string | Transcript show: string].
```

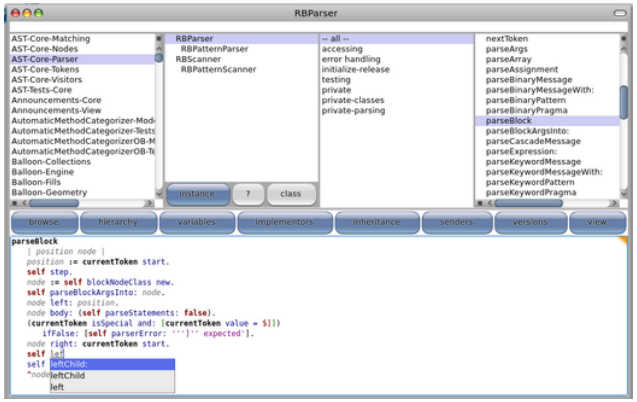
L'exemple envia el missatge `do:` a una taula de cadenes de caràcters amb un bloc com a paràmetre. El bloc és avaluat un cop per cada element de la taula. El paràmetre del bloc, `string`, conté cada element de la taula, un darrera l'altre. Com a resultat de tota l'expressió, les cadenes de caràcters `'Hola'` i després `' Món'` es mostren al Transcript.

Entorn de desenvolupament

La majoria de les implementacions d'Smalltalk proporcionen un entorn integrat que permet explorar el codi font i interaccionar amb objectes. Aquest entorn disposa de moltes eines, totes implementades en Smalltalk gràcies a la seva API de reflexió:

- explorador de classes i mètodes;
- eines de refactoring;
- inspectors d'objectes;
- un depurador;
- administrador i controlador de versions;
- i molt, molt més!

El codi pot ser inspeccionat i avaluat directament dins la imatge, amb menus i senzilles combinacions de tecles.



L'explorador de codi de Pharo