Implémentations

Il existe différentes implémentations de Smalltalk :

Squeak & Pharo : implémentations libres, gratuites et multi-plateformes. Activement développés.

VisualWorks : implémentation propriétaire et multiplateforme, disponible gratuitement pour une utilisation personnelle.

Gemstone : implémentation propriétaire qui intègre une base de données objet à hautes performances.

Et d'autres : GNU Smalltalk, Smalltalk/X, SyX, VA Smalltalk, Dolphin. . .

Applications

Éducation : EToys (Squeak pour OLPC), SqueakBot, BotsInc, Scratch, iStoa.net, Edusim, DrGeoII...

Multimédia : Sophie, OpenCroquet, Plopp...

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

Gestion de la persistance : base de données objets (Magma, GemStone), relationnelles (MySQL, PostgreSQL), mapping objet relationnel (Glorp...).



Etoys et DrGeo sur un OLPC

Glossaire

Image mémoire : L'environnement Smalltalk contient un entrepôt persistant d'objets, l'image. Elle contient le code de l'application (les classes et les méthodes), les objets qui constituent l'état de l'application et peut même contenir l'environnement de programmation pour inspecter et déboguer le programme pendant qu'il s'exécute.

Machine virtuelle : Une machine virtuelle est un programme qui est capable d'exécuter d'autres programmes. Cela permet de faciliter la portabilité des applications que l'on développe.

Réflexion: On dit qu'un langage est réflexif s'il dispose de mécanismes permettant d'inspecter et de modifier du code pendant l'exécution d'un programme.

Smalltalk

un langage de programmation purement orienté objet et un environnement dynamique



RIO

Les blocs sont des objets qui contiennent du code non exécuté immédiatement. Ils sont à la base des structures de contrôles comme les conditionnelles et les boucles.

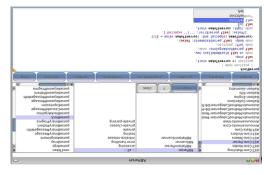
```
#('Hello ' 'World')
do: [:string | Transcript show: string].
```

Dans cet exemple, le message do: est envoyé à un tableau de chaînes de caractères avec un bloc en paramètre. Le bloc est évalué une fois pour chaque élément du tableau. Le paramètre string du bloc contient chaque élément du tableau, l'un après l'autre. Le résultat de l'expression complète est l'affichage des chaînes 'Hello ' puis 'World' dans le transcript.

Environnement de développement

La plupart des implémentations de Smalltalk sont fournies avec un environnement de développement intégré qui permet de naviguer dans le code et d'interagir avec les objets. De nombreux outils sont disponibles, tous implémentés en Smalltalk grâce à une API de réflexion

- -un navigateur de classes et de méthodes (+ refactorisation);
- un inspecteur d'objets ;
- : məngoqəp un -
- et beaucoup plus! L'environnement permet d'évaluer du code par un simple raccourci clavier et de voir immédiatement le



Le navigateur de code de Pharo

L'appel de méthode se fait par envoi de message. Le message se construit sur la base du langage naturel avec un sujet, un verbe et des compléments. Tout envoi de messages retourne un objet. Tous les messages sont envoyés à un objet que l'on appelle le receveur du message. Il existe trois types de messages : unaire, binaire sage. Il existe trois types de messages : unaire, binaire

Messages unaires. Un message unaire n'a pas d'argument.

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

et à mots-clés.

Envoi de messages

Le premier exemple crée et retourne une nouvelle instance de la classe Array en lui envoyant le message new. Le deuxième exemple demande la taille du tableau ce qui retourne 0.

Messages binaires. Un message binaire ne prend qu'un argument, se nomme par un symbole et est souvent utilisé pour des opérations arithmétiques.

```
3 + 4.
```

Le message + est envoyé à l'objet 3 avec comme paramètre 4. Dans le second cas, le message, est envoyé à la chaîne de caractères 'Hello' avec ' World' en paramètre.

Messages à mots-clès. Un message à mots-clès peut prendre un ou plusieurs arguments. Les arguments sont insérés entre chaque mot-clé, après les deux-points.

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

Le premier exemple appelle la méthode allButFirst: sur une chaîne de caractères et passe l'argument 5. La méthode retourne la chaîne de caractères 'talk'. Le deuxième exemple retourne une collection contenant les élèments 3, 5, 7 et 9.

Concepts importants de Smalltalk

Smalltalk est un langage orienté objet, à typage dynamique dont la syntaxe est minimale et peut s'apprendre en quinze minutes.

Sa principale force vient du fait qu'il soit très cohérent .

tout est objet : les classes, les méthodes, les nombres,

 très peu de règles et aucune exception.
 Smalltalk fonctionne sur le principe d'une machine virtuelle. Le développement se fait dans une image

mémoire dans laquelle se trouve l'ensemble des objets du système avec lesquels il est possible d'interagir.

Syntaxe Smalltalk

Mots réservés

noitetaeffe	(-) 110) = •
	Caractères réservés
correspondant à la méthode courante	
contexte de la super classe partie de la pile d'exécution	txətnoOsidt
faut des variables) objets booléens l'objet courant l'objet courant	true et false self super
objet indéfini (valeur par dé-	Ţţū

	chaîne de caractères,
	"enistrammoo"
(i	
bloc de code (c'est un objet	[]
cascade de messages	.
termine toute expression	(triod).
nombre 123	
raux : le symbole #abc et le	
tableau contenant deux litté-	#(sbc 123)
le caractère a	2 \$
poraires	
déclaration de variables tem-	Figu Sigu Iigu
thode	
retour du résultat d'une mé-	(1 no) -
affectation	(→ no) =: