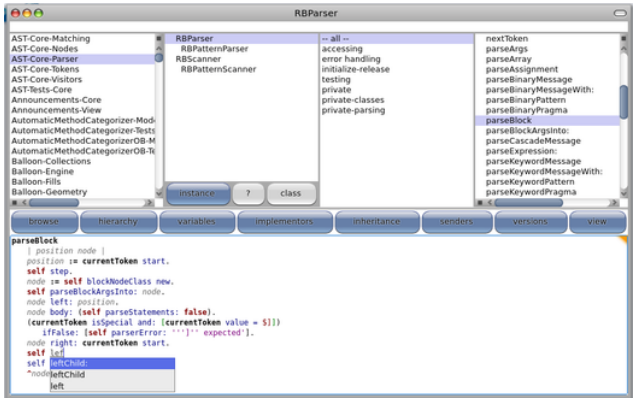

O navegador de código do Pharo

Implementações

Existem várias implementações de Smalltalk disponíveis:

Squeak: gratuita, de código aberto e multi-plataforma. Desenvolvida ativamente por uma comunidade internacional.

VisualWorks: proprietaria e multi-plataforma, disponível gratuitamente para uso não-comercial.

Gemstone: implementação proprietária que inclui um banco de dados altamente eficiente.

And others: GNU Smalltalk, Smalltalk/X, SyX, VA Smalltalk, Dolphin...

Aplicações

Desde sua criação no início dos anos 80, Smalltalk tem sido amplamente usado tanto em pesquisa acadêmica com em aplicações comerciais. Aqui estão algumas aplicações em Smalltalk atuais que estão evoluindo a tecnologia do software.

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

Multimídia: Sophie, OpenCroquet, Plopp...

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

Persistência: bancos de dados orientados a objeto (Magma, GemStone), relacionais (MySQL, PostgreSQL) e mapeamentos objeto-relacional (Glorp).


Uma sesso de desenho no Plopp

Glossário

Imagem: O ambiente do Smalltalk contém um repositório de objetos persistente, a imagem. Esta contém código de aplicações (classes and métodos), objetos contendo o estado das aplicações e até inclui as ferramentas de desenvolvimento para inspecionar e depurar o programa enquantoeventos ele está executando.

Máquina virtual: Uma máquina virtual é um programa que é capaz de executar outros programas. Ele facilita a portabilidade de aplicações.

Reflexão computacional: Um linguagem é dita reflexiva quando contem mecanismos para inspecionar e modificar o código durante a execução do programa.

Tipagem dinâmica: Algumas linguagens forçam o desenvolvedor a indicar o tipo de cada variável (inteiro,

Smalltalk

uma linguagem de programação
puramente orientada a objetos
e um ambiente dinâmico

EUROPEAN SMALLTALK USER GROUP
esug
www.esug.org

Conceitos importantes de Smalltalk

Smalltalk é uma linguagem orientada a objetos, dinamicamente tipada, com uma sintaxe simples que pode ser aprendida em quinze minutos. Sua maior vantagem é ser muito consistente:

- tudo é um objeto: classes, métodos, números, etc.
- um pequeno número de regras, sem excessões!

Smalltalk roda numa máquina virtual. O desenvolvimento acontece numa imagem onde todos os objetos vivem e são modificados.

A sintaxe do Smalltalk

Palavras reservadas	
`nil`	objeto indefinido (valor inicial das variáveis)
`true` e `false`	objetos boleanos
`self`	objeto atual
`super`	objeto atual no contexto da superclasse
`thisContext`	pilha de execução do método atual

Caracteres reservados	
`:=` (ou ←)	atribuição
`^` (ou ↑)	retorna um resultado dum método
`| var1 var2 var3 |`	declaração de três variáveis temporárias
`$a`	charactere `a`
`#(abc 123)`	array contendo dois literais: o símbolo `#abc` e o número `123`
`.` (ponto)	fim da expressão
`;`	messagem em cascata
`[ ]`	bloco de código (é um objeto !)
`"comentário"`	
`'cadeia de caracteres'`	

Envio de mensagens

Um método é chamado pelo envio de uma mensagem a um objeto, o receptor da mensagem; a mensagem retorna um objeto. A mensagem é baseada em linguagem natural, tem um sujeito, um verbo e complementos. Existem três tipos de mensagem: unária, binária e palavra-chave.

Mensagens unárias. Não têm argumentos.

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

O primeiro exemplo cria e retorna uma nova instancia da classe Array ao lhe enviar a mensagem `new`. O segundo exemplo pede o tamanho desta, o que retorna `0`.

Mensagens binárias. Têm só um argumento, são chamadas por um símbolo e normalmente usadas para expressões matemáticas.

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

A mensagem `+` é enviada para o objeto `3` tendo `4` como parâmetro. No segundo caso a mensagem `,` é envidada para a cadeia de caracteres `'Hello'` tendo `' World'` como parâmetro.

Mensagens de palavra-chave. Podem ter um ou mais argumentos, que são inseridos entre cada palavra-chave, depois dos dois pontos.

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

O primeiro exemplo chama o método `allButFirst:` numa cadeia de caracteres e passa o argumento `5`. O método retorna a cadeia de caracteres `'talk'`. O segundo exemplo retorna uma coleção contendo os elementos `3`, `5`, `7` e `9`.

Bloco

Blocos são objetos contendo código que não é executado imediatamente. Eles são a base das estruturas de controle como condicionais ou laços. Inclusive blocos podem ser usados para acrescentar comportamento, p. ex., a itens de menu.

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

O exemplo envia a mensagem `do:` para uma array de cadeia de caracteres com um bloco como parâmetro. O bloco é avaliado uma vez para cada elemento da array. O parâmetro do bloco, `string`, contém cada elemento da array um após o outro. Como resultado da expressão inteira as cadeias de caracteres `'Hello '` e então `'World'` são mostradas.

Ambiente de desenvolvimento

A maioria de suas implementações vem com um ambiente de desenvolvimento integrado que permite navegar no código e interajir com os objetos. Muitas ferramentas estão disponíveis, todas implementadas em Smalltalk graças à API de reflexão:

- um navegador de classe e método;
- ferramentas de refatoração;
- inspetores de objetos;
- um depurador;
- ferramentas de controle de releases e controle de versão;
- e muito, muito mais!

O código pode ser inspecionado e avaliado diretamente na imagem, usando simples combinações de teclas e menus.