早わかりSmalltalk

Ruby勉強会

2008 Masashi Umezawa

これが読めればOK

```
exampleWithNumber: x
|y|
true & false not & (nil isNil) ifFalse: [self halt].
y := self size + super size.
#($a #a "a" 1 1.0) do: [:each |
  Transcript show: (each class name); show: ‘hi'
].
^ x < y
```

http://www.esug.org/whyusesmalltalktoteachoop/smalltalksyntaxonapostcard/ より引用

一時変数の定義

- 縦棒で囲む
- 例:
  - |x y z|
- 代入しない場合、nilが入っている

変数への代入

- := または _ (アンダースコア)
  - := がANSI標準
  - Squeakでは_を使う人も多い(環境内では←で表示)
- 例:
  - x := 1.
  - x := y := 2.

値を返す

- ^ (キャレット)を使う
  - Squeakでは ↑ で表示される
- 例:
  - ^1
  - ^3+4

メッセージ式

- オブジェクトにメッセージを送る
- 例:
  - obj name.
  - obj size.
- メッセージ式の終わりにピリオドを打つ

2項メッセージ式

- 演算子っぽいもの
  - 1 + 3.
  - 10 / 4.
  - 10 // 3.
  - 'aaa', 'bbb'.

キーワードメッセージ式

- 引数の区切りに任意の文字列を使える
  - string copyFrom: 1 to: 3.
    - string.copy(1,3)
  - point setX: 10 setY: 20
    - point.set(10,20)
- それぞれの引数の意味がメソッド定義部を見なくともなんとなくわかる

式の優先順位

- 単項 > 2項 > キーワード
  - 3 factorial + 4 factorial between: 10 and: 100.
    - の場合
  - 6 + 24 between: 10 and: 100.
    - となって
  - 30 between: 10 and: 100.
    - となる
  - 30は10と100の間なので、true
- ()で括るとそこの優先度が最も高くなる
  - ('hello' copyFrom: 1 to: 3) reverse.

リテラル

- 文字
  - $a
- 文字列
  - ' hello'
- シンボル (変更不可能な文字列)
  - #hello
- 配列
  - #(1 2 3 $a 'hello')
- 動的配列
  - {1+2. 'hello‘ reverse}
- コメント
  - ""で囲む

予約オブジェクト

- self
  - 自身を表す
- super
  - selfと同じだがメソッドの検索はスーパークラスのものを使う
- true
- false
- nil
- thisContext
  - 実行中のコンテキスト(システムの実行状態)を表す

グローバルオブジェクト

- Transcript
  - 標準出力っぽく使う窓
- Smalltalk
  - 名前空間を提供
  - グローバル的な便利メソッドもいくつか持つ
- ActiveWorld
  - Squeak限定
  - 現在のルートウィンドウとなっているオブジェクト

カスケード

- 先頭のオブジェクトに連続的にメッセージを送るときの記法
  - ; (セミコロン)を使う
  - Transcript cr; space; show: 'hello'.
- カスケードを使わないと...
  - Transcript cr.
  - Transcript space.
  - Transcript show: 'hello'.

ブロック

- あとで実行したいコードの固まり
- []で囲む
  - 引数は :arg1 :arg2 ... | のように指定
  - [:x | x + 1].
    - func(x) {x + 1} のようなもの
  - 実行したいときに引数の数に応じて value、value:、value:value:などを送る
- 制御構造やイテレータで多用される
  - 10>3 ifTrue: [self inform: 'hello'].
  - #(1 2 3) do: [:each | Transcript show: each].

ではもう一度

```
exampleWithNumber: x
|y|
true & false not & (nil isNil) ifFalse: [self halt].
y := self size + super size.
#($a #a "a" 1 1.0) do: [:each |
  Transcript show: (each class name); show: ‘hi'
].
^ x < y
```

http://www.esug.org/whyusesmalltalktoteachoop/smalltalksyntaxonapostcard/ より引用

まとめ

- 奇妙で複雑に見えていたSmalltalkが大体読めるようになった
- あとは開発環境に慣れて下さい
  - 予告:
    - 「驚き最小限Smalltalk」
      - Rubyなどに慣れている人がなんとなく使うことが出来るSmalltalkの環境
      - どうすれば良いのか、ヒントをお願いします