Grammaire

Voici une description EBNF de la grammaire supportée par SINS. Au moment d'écrire le rapport de la première remise, SINS ne supportait pas encore les abbréviations complétement mais nous avons choisi de l'inclure dans la description malgré tout. Cette grammaire évoluera probablement d'ici la fin du projet.

datum

⟨*datum*⟩ ⟶ ⟨*simple datum*⟩
| ⟨*compound datum*⟩

⟨*simple datum*⟩ ⟶ ⟨*boolean*⟩
| ⟨*number*⟩
| ⟨*character*⟩
| ⟨*string*⟩
| ⟨*symbol*⟩

⟨*symbol*⟩ ⟶ ⟨*identifier*⟩

⟨*compound datum*⟩ ⟶ ⟨*list*⟩

⟨*list*⟩ ⟶ (⟨*datum*⟩*)
| (⟨*datum*⟩+ . ⟨*datum*⟩)
| ⟨*abbreviation*⟩

⟨*abbreviation*⟩ ⟶ ⟨*abbrev prefix*⟩ ⟨*datum*⟩

⟨*abbrev prefix*⟩ ⟶ ´
| `
| ,
| ,@

program

⟨*program*⟩ ⟶ ⟨*command or definition*⟩*

⟨*command or definition*⟩ ⟶ ⟨*command*⟩
| ⟨*definition*⟩
| (begin ⟨*command or definition*⟩+)

⟨*definition*⟩ ⟶ (define ⟨*variable*⟩ ⟨*expression*⟩)
| (define (⟨*variable*⟩ ⟨*def formals*⟩) ⟨*body*⟩)
| (begin ⟨*definition*⟩*)

⟨*def formals*⟩ ⟶ ⟨*variable*⟩*
| ⟨*variable*⟩* . ⟨*variable*⟩

expression

⟨*expression*⟩ ⟶ ⟨*variable*⟩
| ⟨*literal*⟩
| ⟨*procedure call*⟩
| ⟨*lambda expression*⟩
| ⟨*conditional*⟩
| ⟨*assignment*⟩
| ⟨*derived expression*⟩

⟨*literal*⟩ ⟶ ⟨*quotation*⟩
| ⟨*self-evaluating*⟩

⟨*self-evaluating*⟩ ⟶ ⟨*boolean*⟩
| ⟨*number*⟩
| ⟨*character*⟩
| ⟨*string*⟩

⟨*quotation*⟩ ⟶ '⟨*datum*⟩
| (quote ⟨*datum*⟩)

⟨*procedure call*⟩ ⟶ (⟨*operator*⟩ ⟨*operand*⟩*)

⟨*operator*⟩ ⟶ ⟨*expression*⟩

⟨*operand*⟩ ⟶ ⟨*expression*⟩

⟨*lambda expression*⟩ ⟶ (lambda ⟨*formals*⟩ ⟨*body*⟩)

⟨*formals*⟩ ⟶ (⟨*variable*⟩*)
| ⟨*variable*⟩
| (⟨*variable*⟩+ . ⟨*variable*⟩)

⟨*body*⟩ ⟶ ⟨*definition*⟩* ⟨*sequence*⟩

⟨*sequence*⟩ ⟶ ⟨*command*⟩* ⟨*expression*⟩

⟨*command*⟩ ⟶ ⟨*expression*⟩

⟨*conditional*⟩ ⟶ (if ⟨*test*⟩ ⟨*consequent*⟩ ⟨*alternate*⟩)

⟨*test*⟩ ⟶ ⟨*expression*⟩

⟨*consequent*⟩ ⟶ ⟨*expression*⟩

⟨*alternate*⟩ ⟶ ⟨*expression*⟩
| ⟨*empty*⟩

⟨*assignment*⟩ ⟶ (set ! ⟨*variable*⟩ ⟨*expression*⟩)

⟨*derived expression*⟩ ⟶ (cond ⟨*cond clause*⟩+)
| (cond ⟨*cond clause*⟩* (else ⟨*sequence*⟩))
| (and ⟨*test*⟩*)
| (or ⟨*test*⟩*)
| (let (⟨*binding spec*⟩*) ⟨*body*⟩)
| (let ⟨*variable*⟩ (⟨*binding spec*⟩*) ⟨*body*⟩)
| (let* (⟨*binding spec*⟩*) ⟨*body*⟩)
| (letrec (⟨*binding spec*⟩*) ⟨*body*⟩)
| (begin ⟨*sequence*⟩)
| ⟨*quasiquotation*⟩

⟨*cond clause*⟩ ⟶ (⟨*test*⟩ ⟨*sequence*⟩)
| (⟨*test*⟩)
| (⟨*test*⟩ => ⟨*recipient*⟩)

⟨*recipient*⟩ ⟶ ⟨*expression*⟩

⟨*binding spec*⟩ ⟶ (⟨*variable*⟩ ⟨*expression*⟩)

quasiquotation

⟨*quasiquotation*⟩ ⟶ `⟨*expression*⟩
| (quasiquote ⟨*expression*⟩)