Languages-beta: OC-L-03-Names *

The PLanCompS Project

`OC-L-03-Names.cbs` | PLAIN | PRETTY

OUTLINE

Language "OCaml Light"

3 Names

Naming objects

Syntax *VN* : value-name ::= lowercase-ident
| '(' operator-name ')'
ON : operator-name ::= prefix-op | infix-op
PO : prefix-op ::= prefix-symbol

```
// infix-op ::= infix-symbol
//            | '*' | '+' | '-' | '-.' | '=' | '<' | '>' | '||' | '&' | '&&'
//            | '!=' | 'or' | ':=' | 'mod'
//            | 'land' | 'lor' | 'lxor' | 'lsl' | 'lsr' | 'asr'
```

*Suggestions for improvement: `plancomps@gmail.com`.
Reports of issues: `https://github.com/plancomps/CBS-beta/issues`.

Infix operator precedence

Syntax *IO* : infix-op ::= infix-op-1 | infix-op-2 | infix-op-3 | infix-op-4
| infix-op-5 | infix-op-6 | infix-op-7 | infix-op-8

Lexis *IO-1* : infix-op-1 ::= '**' operator-char* | 'lsl' | 'lsr' | 'asr'

IO-2 : infix-op-2 ::= '*'
| '*' operator-char-not-asterisk operator-char*
| ('/' | '%') operator-char*
| 'mod' | 'land' | 'lor' | 'lxor'

IO-3 : infix-op-3 ::= ('+' | '-') operator-char*

IO-4 : infix-op-4 ::= ('@' | '^') operator-char*

IO-5 : infix-op-5 ::= ('=' | '<' | '>' | '$') operator-char*
| '|' (operator-char-not-bar operator-char*)?
| '||' operator-char+
| '&' operator-char-not-ampersand operator-char*
| '&&' operator-char+
| '!='

IO-6 : infix-op-6 ::= '&' | '&&'

IO-7 : infix-op-7 ::= 'or' | '||'

IO-8 : infix-op-8 ::= ':='

Lexis *CN* : constr-name ::= capitalized-ident

TCN : typeconstr-name ::= lowercase-ident

FN : field-name ::= lowercase-ident

MN : module-name ::= capitalized-ident

Referring to named objects

Syntax *VP* : value-path ::= value-name

CSTR : constr ::= constr-name

TCSTR : typeconstr ::= typeconstr-name

F : field ::= field-name

Semantics value-name⟦ _ : value-path ⟧ : ⇒ ids

Rule value-name⟦ *LI* ⟧ = "*LI*"

Rule value-name⟦ '(' *PS* ')' ⟧ = string-append("(", "*PS*", ")")

Rule value-name⟦ '(' *IO-1* ')' ⟧ = string-append("(", "*IO-1*", ")")

Rule value-name⟦ '(' *IO-2* ')' ⟧ = string-append("(", "*IO-2*", ")")

Rule value-name⟦ '(' *IO-3* ')' ⟧ = string-append("(", "*IO-3*", ")")

Rule value-name⟦ '(' *IO-4* ')' ⟧ = string-append("(", "*IO-4*", ")")

Rule value-name⟦ '(' *IO-5* ')' ⟧ = string-append("(", "*IO-5*", ")")

Rule value-name⟦ '(' *IO-6* ')' ⟧ = string-append("(", "*IO-6*", ")")

Rule value-name⟦ '(' *IO-7* ')' ⟧ = string-append("(", "*IO-7*", ")")

Rule value-name⟦ '(' *IO-8* ')' ⟧ = string-append("(", "*IO-8*", ")")

Semantics constr-name⟦ _ : constr ⟧ : ⇒ ids

Rule constr-name⟦ *CN* ⟧ = "*CN*"

Semantics typeconstr-name⟦ _ : typeconstr ⟧ : ⇒ ids

Rule typeconstr-name⟦ *TCN* ⟧ = "*TCN*"

Semantics field-name⟦ _ : field ⟧ : ⇒ ids

Rule field-name⟦ *FN* ⟧ = "*FN*"