FOL S-Expression Syntax

S-Expressions for untyped FOL

Expression Type	Fitch Syntax	S-Expression Syntax
Propositional Statements	P	P
Not	$\neg P$	(not P)
And	$P \wedge Q$	(and P Q)
Or	$P \vee Q$	(or P Q)
Material Conditional	$P \to Q$	(implies P Q)
Bi-conditional	$P \leftrightarrow Q$	(iff P Q)
XOR	$P \oplus Q$	(xor P Q)
Variable/constant	x, a	x, a
Predicate	$P(a)$	(P a)
Function	$s(a)$	(s a)
Universal Quantifier	$\forall x P(x)$	(forAll x (P x))
Existential Quantifier	$\exists x P(x)$	(exists x (P x))

S-Expressions for sorted/typed FOL

If you're writing software for sorted FOL, you can extend the S-expression syntax as follows in order to allow the user to define sorts/types, define constants under those sorts/types, define functions returning specific sorts/types and accepting arguments of fixed sorts/types, and finally express statements quantified over specific sorts/types.

Expression Type	Expression Structure	Example
Type/Sort Declarations	(declare-sort *new-sort supersort*)	(declare-sort Dog Animal)
Constant Declarations	(*sort object-name*)	(Object a)
Function Declarations	(*return-type function name argument-types* ...)	(Number *s* Number)
Universal Quantifier	(forAll (*sort* x) *quantified-statement*)	(forAll (Object x) (P x))
Existential Quantifier	(exists (*sort* x) *quantified-statement*)	(exists (Object x) (P x))