Simply Typed λ-calculus: $\lambda\!\to$

Types: $T ::= o \mid T \to T$ ←

o is the "base type"
it is unspecified
it contains no values

$T \to T$ type of functions

Typing Rules:

Variable
$$\frac{x : T \in \Gamma}{\Gamma \vdash x : T}$$

Abstraction
$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x{:}A.t : A \to B}$$

Application
$$\frac{\Gamma \vdash f : A \to B \qquad \Gamma \vdash a : A}{\Gamma \vdash f\,a : B}$$

Examples of types:

o, $o \to o$, $(o \to o) \to o$,

$o \to (o \to o)$, $(o \to o) \to (o \to$

also written $o \to o \to o$

also written $(o \to o) \to o \to$

arrows associate to the right

$((o \to o) \to o) \to (o \to o) \to o$

Reduction Rule: $(\lambda x{:}A.t)\,a \rightsquigarrow t[x{:}A]$

Explanation of the rules

Variable Rule

A variable has the type that is assigned to it by the context

ex:

$$x:o,\ y:(o\to o)\to o,\ z:o\to o \vdash y:(o\to o)\to o$$

Abstraction Rule

$$\frac{\Gamma, x:A \vdash t:B}{\Gamma \vdash \lambda x{:}A.t : B}$$

The variable x is in the context of the assumption not in context of conclusion

abstracted variables are explicitly typed

Application Rule

$$\frac{\Gamma \vdash f:A\to B \qquad \Gamma \vdash a:A}{\Gamma \vdash f\,a : B}$$

Same context for all three judgments

A term can be applied only if it has a function type

It can only be applied to arguments that have its domain type