1 Terms, Types and Kinds

For programming language, three levels: ***terms***, ***types***, and ***kinds***, have proved sufficient .

1.1 Terms

Sometimes, the word *term* and *expression* are used interchangeably.

Recap term-level *abstraction* and *application* in the λ-calculus:

`t`	::=	`x`	*Variable*
	|	`λx.t`	*Abstraction*
	|	`t, t`	*Application*

The symbol `t` in the left-hand side of the rules is called a *metavariable*. It is a place-holder for some particular term.

1.2 Types

1.3 Kinds

To treat type-level functions, collectively called *type operators* more formally, it is required to:

1. Add a collection of rules of *kinding* which specify how type expressions can be combined to yield new type expressions.
2. Whenever a type T appears in a term (λx : T.t), check whether T is well formed.
3. Add a collection of rules for the definitonal equivalence relations between types.

Γ ⊢ T :: K is read as "type T has kind K in context Γ".

kinding is a well-formedness relation.