Type Checking

To do type checking a compiler needs to assign a type expression to each component of the source program. The compiler must then determine that these type expressions conform to a collection of logical rules that is called the type system for the source language

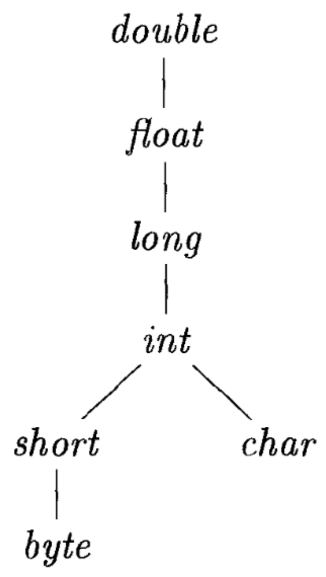
Rules

Two forms: synthesis and inference.
Type synthesis builds up the type of an expression from the types of its subexpressions . It requires names to be declared before they are used.
Type inference determines the type of a language construct from the way it is used. Mostly used on ML with check types but do not require names to be declared

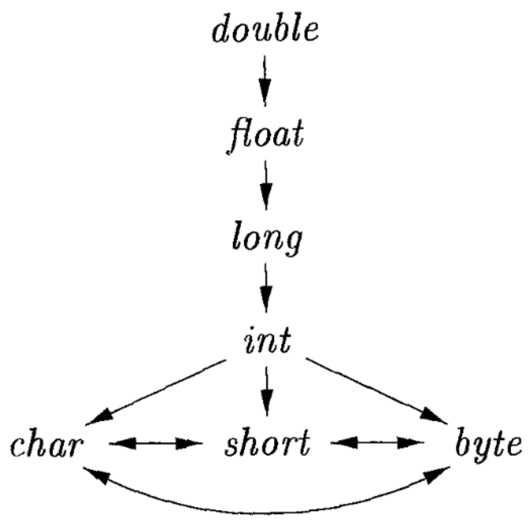
Type conversions

The widening rules: any type lower in the hierarchy can be widened into a higher type.

The narrowing rules: a type s can be narrowed to a type t if there is a path from s to t.



(a) Widening conversions



(b) Narrowing conversions

Conversion is said to be explicit if the programmer must write something to cause the conversion (casts).