Extended friend Declarations (Rev. 2)

I. Introduction

The two previous versions of this document (J16/03-0103 = WG21 N1520, J16/04-0056 = WG21 N1616) described the need to extend the current language to support a wider range of friend declarations and proposed specific wording to be incorporated into the C++ Standard. At the Redmond (October, 2004) meeting, the Evolution Working Group approved the proposal for recommendation to the Committee and referred it to the Core Language Working Group for review of the wording.

The Core Language Working group suggested one change to the wording in the previous document. Core language issue 298, adopted as a Defect Report by the Committee in April, 2003, allows use of typedefs of cv-qualified class names as nested-name-specifiers. For consistency with that change, the proposal was amended to allow cv-qualified types to specified as friends. The wording below reflects that change but is otherwise identical to that of the previous document.

II. Proposed Wording

1. Change 9.2¶7 from

   The member-declarator-list can be omitted only after a class-specifier, an enum-specifier, or a decl-specifier-seq of the form friend elaborated-type-specifier.

   to

   The member-declarator-list can be omitted only after a class-specifier or an enum-specifier or in a friend declaration (11.4).

2. Delete the following wording from 11.4¶2:

   An elaborated-type-specifier shall be used in a friend declaration for a class. [Footnote: The class-key of the elaborated-type-specifier is required.]

3. Add the following as a new paragraph following 11.4¶2:

   A friend declaration that does not declare a function shall have one of the following forms:
friend elaborated-type-specifier;
friend simple-type-specifier;
friend typename-specifier;

[Note: a friend declaration may be the declaration in a template-declaration (clause 14, 14.5.3).] If the type specifier in a friend declaration designates a (possibly cv-qualified) class type, that class is declared as a friend; otherwise, the friend declaration is ignored. [Example:

class C;
typedef C Ct;

class X1 {
    friend C; // OK: class C is a friend
};

class X2 {
    friend Ct; // OK: class C is a friend
    friend D; // error: no type-name D in scope
    friend class D; // OK: elaborated-type-specifier declares new class
};

template <typename T> class R {
    friend T;
};

R<C> rc; // class C is a friend of R<C>
R<int> Rint; // OK: "friend int;" is ignored

—end example]