

Section 2.1 C++11

Braced Init

```

int i;           // default initialization
int j();        // Oops, function declaration
int k = int();  // copy initialization of k with a value-initialized temporary

int* pd = new int; // default initialization of dynamic int object
int* pv = new int(); // value      "      "      "      "

```

For **scalar types**, *default* initialization does not actually initialize an object, and *value* initialization will initialize the object as if by the literal \emptyset . Note that the representation of this value is not necessarily all zero bits, as some platforms use distinct trap values for the null pointer value for pointers and for pointer-to-member objects.

For class having an accessible **user-provided** default constructor, default initialization and value initialization behave identically, calling the default constructor. If there is no accessible default constructor, both forms are ill formed. For objects of class types having an implicitly defined default constructor, each base and member subobject will be default initialized or value initialized according to the form of initialization indicated for the complete object; if any of those initializations produces an error, then the program is ill formed. Note that when a union with an implicitly defined default constructor is value initialized, the first member of the union will be value initialized as the active member of that union:

```

struct B
{
    int i;
    B() : i() { } // User-provided default constructor: i is value initialized.
};

struct C
{
    int i;
    C() { } // User-provided default constructor: i is default initialized.
};

struct D : B { int j; }; // derived class with no user-provided constructors

int* pdi = new int; // default init of dynamic int, *pdi is uninitialized
int* pvi = new int(); // value  init of dynamic int, *pvi is 0
B* pdb = new B; // default init of dynamic B, pdb->i is 0
B* pvb = new B(); // value  init of dynamic B, pvb->i is 0
C* pdc = new C; // default init of dynamic C, pdc->i is uninitialized
C* pvc = new C(); // value  init of dynamic C, pvc->i is uninitialized
D* pdd = new D; // default init of dynamic D, pdd->i is uninitialized
D* pvd = new D(); // value  init of dynamic D, pvd->i is 0

```

In the case of an object of type B, both default and value initialization will invoke the user-provided default constructor, which initializes the subobject *i* to \emptyset . In the case of an object of type C, both default and value initialization will invoke the user-provided default constructor, which does not initialize the subobject *i*. In the case of an object of type D, which