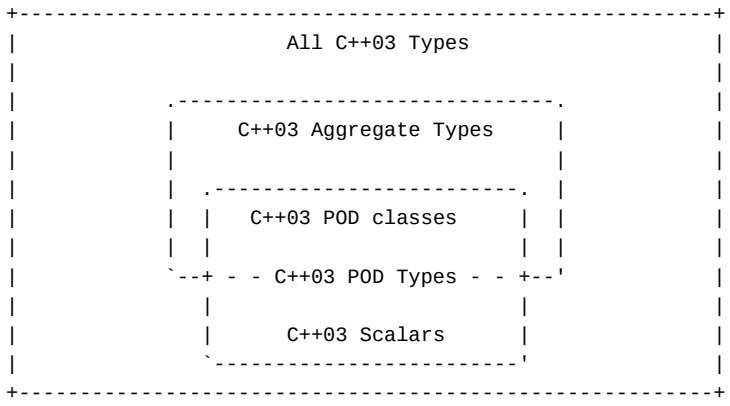
Now that we have classified what kinds of user-defined types are considered aggregate types in C++03, let's step back and appreciate the complete set of **C++03 POD types**. We begin by observing that all of the scalar types are POD types in every version of C++:



A *C++03 scalar* is a possibly **const** or **volatile arithmetic type**, enumeration type, pointer type, or pointer-to-member type:

```
          int         i    = { 0   };  // integer
    const short       cs   = { 0   };  // const short
          double      d    = { 0.0 };  // floating point
          enum E { }  e    = { E() };  // enumeration
          char*       p    = { 0   };  // pointer to char
    const char*       pc   = { 0   };  // pointer to const char
          char* const cp   = { 0   };  // const pointer to char
class C;  int   C::*pm     = { 0   };  // pointer to int member data
          void (C::*pmf)() = { 0   };  // pointer to void member function
```

As it turns out, scalars can also be initialized using the braced notation in C++03, just not with empty braces. A C++03 POD-**struct** is an **aggregate class**, declared with either the **struct** or **class class key**, that has no non-POD-**struct** members, no non-POD-**union** members, no array members having non-POD elements, no non**static** *reference* members, and no user-declared copy-assignment operator or destructor:

```
// Class declaration                Is a C++03 POD?
class  B0 { };                   // Yes, an empty (aggregate) class is a POD.
class  B1 { public: int x; };    // Yes, public data
class  B2 { int f(); };          // Yes, private non-virtual function
class  B3 { static B1 x; };      // Yes, static members don't matter
struct B4 { ~B4(); };            // No, a destructor cannot be declared.
struct B5 { B5& operator=(const B5&); };
                                 // No, copy assignment cannot be declared.
struct B6 { int* x; };           // Yes, pointers are allowed in PODs too.
struct B7 { B5   x; };           // No, any data members must also be PODs.
```