Proposal/Wording for Bit-field Default Member Initializer Syntax

Document No.: R0187R1
Project: Programming Language C++ - Evolution
Author: Andrew Tomazos <andrewtomazos@gmail.com>
Date: 2016-06-28

Summary

We propose a new syntax for bit-fields that allows them to have default member initializers.

The consensus after discussing P0187R0 at EWG Oulu was to “add a new syntax for being able to provide both a bitfield-width and an initializer”, with 21 for and 0 against.

Further to this, we have designed a new syntax that is simple, easy to teach, requires no disambiguation rules, is easy to parse and requires only a one line addition to the grammar.

Motivation

The motivation for bit-field default member initializers is the same as for default member initializers for non-bit-field members. It can be argued the motivation for bit-fields is even stronger, as they usually occur in simple structs.

Design

Several alternatives were considered in the syntax design. In the first version of the proposal we offered allowing the ambiguous syntax by providing a set of disambiguation rules:

```
struct S { int x : 5 = 42; }  // not proposed
```

This approach was rejected at Oulu as the disambiguation rules were considered too difficult to teach and communicate.

We then offered a declarator-like syntax on the EWG reflector:

```
struct S { int x:[5] = 42; } // not proposed
struct S { int x[:5] = 42; } // not proposed
```

This approach was rejected as we explicitly do not want to make the bitfield width look like a declarator, so not to confuse it with a compound type.

There were several other syntaxes considered on the reflector.

Finally we settled on the proposed syntax:

```
struct S { int x : 5 : = 42; } // proposed
```

The reaction to this syntax was positive.

Example

```
struct S {
  int name : width;

  int name : width : = init;

  int name : width : { init };
};
```

Explanation

The syntax can be taught as follows: “To use a default member initializer for a bit-field, separate the initializer from the bit-field width with a second colon.”

Background

For background and motivation on the problem we are solving see P0187R0.

Wording

Add to grammar:

member-declarator:
    declarator virt-specifier-seq$_{opt}$ *pure-specifier*$_{opt}$
    declarator brace-or-equal-initializer$_{opt}$
    identifier$_{opt}$ *attribute-specifier-seq*$_{opt}$*: constant-expression*
    identifier attribute-specifier-seq$_{opt}$*: *
      constant-expression : brace-or-equal-initializer

Modify [class.bit]:

A *member-declarator* of one of the forms:

identifier$_{opt}$ *attribute-specifier-seq*$_{opt}$*: constant-expression*
identifier attribute-specifier-seq$_{opt}$*: *
constant-expression : brace-or-equal-initializer

specifies a bit-field; [...]