Stack Overflow is a question and answer site for professional and enthusiast programmers. It's 100% free. Take the 2-minute tour ×

# Correct way to initialize vector member variable

```
// Method One
class ClassName
{
public:
    ClassName() : m_vecInts() {}

private:
    std::vector<int> m_vecInts;
}

// Method Two
class ClassName
{
public:
    ClassName() {} // do nothing

private:
    std::vector<int> m_vecInts;
}
```

Question> What is the correct way to initialize the vector member variable of the class? Do we have to initialize it at all?

c++

edited Jun 21 '14 at 18:36
Drew Noakes
93.3k 62 313 404

asked Jul 30 '12 at 16:16
q0987
8,723 21 105 219

1 No, not unless you want to specify an initial capacity (or initialize it's content which you can do with initializer lists in C++11 or boost's `list_of` I *think*). – hmjd Jul 30 '12 at 16:18

1 Both way's vector gets constructed using default constructor , but 1st is preferred – Mr.Anubis Jul 30 '12 at 16:18

1 @Mr.Anubis: Not by me. – Benjamin Lindley Jul 30 '12 at 16:18

If you don't mention it at all in your initializer list, the default constructor will be called. Otherwise, call whichever constructor is appropriate (e.g. maybe you want it to have 100 default constructed elements). – BoBTFish Jul 30 '12 at 16:19

## 3 Answers

See http://en.cppreference.com/w/cpp/language/default_initialization

> Default initialization is performed in three situations:
>
> 1. when a variable with automatic storage duration is declared with no initializer
> 2. when an object with dynamic storage duration is created by a new-expression without an initializer
> 3. when a base class or a **non-static data member is not mentioned in a constructor initializer list and that constructor is called**.
>
> The effects of default initialization are:
>
> - If T is a **class type, the default constructor is called to provide the initial value for the new object**.
> - If T is an array type, every element of the array is default-initialized.
> - Otherwise, nothing is done.

Since `std::vector` is a class type its default constructor is called. So the manual initialization isn't needed.

answered Jul 30 '12 at 16:19
Zeta
**44.1k** 4 66 105

surprise to me that `class T2 { int mem; public: T2() {} // "mem" not in initializer list };` `const T2 t2; // ok, calls the user-provided default ctor // t2.mem is default-initialized` – q0987 Jul 30 '12 at 16:40

I just tested the above code in VS2010 and it is WRONG. the t2.mem is NOT default-initialized! – q0987 Jul 30 '12 at 16:52

@q0987: `mem` is an `int` , it's neither a class, nor an array type, so it won't get default initialized. The code provided on cppreference is almost always verified under the C++11 standard, but I don't know whether this changes anything for the given case (given the `const` qualifier). However, even if the example is inconsistent and maybe even wrong, the explanation and standard itself aren't. – Zeta Jul 30 '12 at 16:54

It depends. If you want a size 0 vector, then you don't have to do anything. If you wanted, say, a size N vector fill of 42s then use the constructor initializer lists:

```
ClassName() : m_vecInts(N, 42) {}
```

answered Jul 30 '12 at 16:18
juanchopanza
**143k** 12 160 279

You do not have to initialise it explcitly, it will be created when you create an instance of your class.

answered Jul 30 '12 at 16:18
mathematician1975
**14.6k** 3 22 53