# C++库向上兼容的方法

王斌 2010-08-26

这里的兼容是指接口相同。要求给旧库打个补丁库，使之与新库兼容。当然尽量充分利用旧库，否则可能导致补丁库过大。如果新库相对于旧库的系统框架发生了巨变，而不仅仅是接口问题，那么可能工作量会增大不少，比如Qt2.X、Qt3.X打上Qt4的补丁库，其中极其重要的QObject改动很大，而她正是几乎所有其他类的基类。

打上补丁后可以直接使用用新库API编写的代码来编译程序，当然有可能需要改动很少一部分代码。

最关键的是利用名字空间。不同名字空间中可以有同名的成员。

如果只是库中很少的几个API名字发生了变化，那么可以用宏定义方便的解决。而对于一个庞大繁杂的库，事情没那么简单了。这里我只是提个思想，接口实现的细节不是这里的重点。

以一个例子来说明问题。不知何人曰过：Codes say everything.

假设原库为libloli.so，库中有个类Loli，头文件如下：

//loli.h

#ifndef LOLI\_H

#define LOLI\_H

#include "global.h"

class Loli {

public:

Loli();

virtual ~Loli();

};

#endif // Loli\_H

// global.h

#ifndef GLOBAL\_H

#define GLOBAL\_H

#include <stdio.h>

#define xdebug(fmt,args...) printf("line: %d, function: %s, file: %s... \t"fmt"\n", \

\_\_LINE\_\_,\_\_FUNCTION\_\_,\_\_FILE\_\_,##args)

#endif // GLOBAL\_H

现在新库中新增了一个接口bool cute() const，那么我的做法是先新建些目录：compatible/src，compatible/include，在compatible/src下建立新的头文件和原代码，其中包含Loli的派生类Loli\_X，派生类中当然要加入所有的新接口，实现不急，无法实现的先不实现。然后在头文件中新增一个名字空间，下面有2种方法：

1. 名字空间中类的名字同新类，继承派生类，而该类不用写任何接口，所有接口在其父类中已写好。这是我用写Qt程序时从uic生成的头文件中得到的灵感。但是有很多问题，比如继续继承会造成更多的资源浪费，多个构造函数会影响效率。更重要的是继承后还是需要写一些接口，比如父类中没有无参数的构造函数，那么子类中要重写一个父类参数相同的构造函数，或者自己加个无参数构造函数，反正就是一定要有个构造函数。对于Qt，最重要的是名字空间中的类若需要moc编译，可能无法生成正确的源代码。
2. 使用强大的typedef功能。简单易操作。新的名字空间中只要

Namespace { typedef 父类 与新库中类同名的子类; }

compatible/include中是新的头文件，可以与原库的头文件名相同，此时你在make时要注意INCPATH这个环境变量的设置。头文件内容仅仅是包含src中的头文件。

再上代码：

// compatible/src/loli\_x.h

#ifndef LOLI\_X\_H

#define LOLI\_X\_H

#include "loli.h"

class Loli\_X : public Loli {

public:

Loli\_X();

virtual ~Loli\_X();

const bool cute() const;

};

namespace X {

typedef Loli\_X Loli;

/\* Not good enough

class Loli :public Loli\_X { };

\*/

}

#endif // LOLI\_X\_H

// compatible/src/loli\_x.cpp

#include "loli\_x.h"

Loli\_X::Loli\_X() {

xdebug();

}

Loli\_X::~Loli\_X() {

xdebug();

}

const bool Loli\_X::cute() const {

xdebug("Qt Loli");

return true;

}

//a new header file: compatible/include/Loli

#include "loli\_x.h"

## Qt2.x and Qt3.x API升级到Qt4的方法

只需在main.cpp或其它头文件中加入

#include <QApplicatiom> //注意与一下宏定义的顺序。QApplication在Qt3中是你自己新加的

#if (\_\_GNUC\_\_ < 4)

# if QT\_VERSION >= 0x040000

# define LIB\_QT4

# endif

#else

# define LIB\_QT4 (QT\_VERSION >= 0x040000)

#endif

#ifdef LIB\_QT4

#define X

#endif

在main函数中的适当位置加入

#ifndef LIB\_QT4

using X::QApplication;

#endif

using namespace X; 不管用，不知为什么

这样，同样的Qt4代码可以在加入新库的Qt2.x、Qt3.x环境中编译了。

TODO:

融入Qt4的框架、宏定义等

Qt4EZX

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*LICENSE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Copyright (C) 2010 Wangbin <wbsecg1@gmail.com>

(aka. nukin in ccmove & novesky in http://forum.motorolafans.com)

This program is free software; you can redistribute it and/or modify

it under the terms of the GNU General Public License as published by

the Free Software Foundation; either version 2 of the License, or

(at your option) any later version.

This program is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

GNU General Public License for more details.

You should have received a copy of the GNU General Public License along

with this program; if not, write to the Free Software Foundation, Inc.,

51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/