C++模板(template)

2015年12月6日 星期日　　下午3:17

一，简介

类型不同但是功能基本相同的函数，在代码实现中，可以用模板代替那些不同的类型
一般是用模板代替类型/类，一般是在函数、类的实现中使用。
使用在函数中，一般是把函数的输入输出参数的变量类型 用模板代替。
使用在类中，是一种类级别的类型(而不是成员函数级别的)，作用域覆盖整个类，可

二，原理及实现

1. 定义

模板定义：模板就是实现代码重用机制的一种工具，它可以实现类型参数化，
版可以分为两类，一个是函数模版，另外一个是类模版。

2. 模板函数

2.1 原理与实现：

A 函数声明的时候，用template声明语句，声明一个"泛的类型"的代号
B 函数实现的时候，如果你预设的类型可以支持，那只写一套实现就行了
C 函数调用的时候，传入你希望用的类型，他会自动调整，会调用你实现
含具体类型"的实现。

2.2 示例代码及运行结果

```cpp
#include <iostream>

template<typename T1>
T1 max(T1 a, T1 b);

int main(){
    max(0, 1);
    //max(0.6, 5); // complie error: no matching function for call to `max(double, int)'
    max(0.6, -0.9);
    max("0.4", "asafsf");
    max("bbbbb", "asafsf");
}

template<typename T1>
T1 max(T1 a, T1 b){
    if (a > b)
    {
        std::cout << "max(" << a << ",:" << b << ")="  << a << std::endl;
        return a;
    }
    else if (b >= a)
    {
        std::cout << "max(" << a << ",:" << b << ")=" << b << std::endl;
        return b;
    }
}
```

```
max(0,:1)=1
max(0.6,:-0.9)=0.6
```

型，达到复用代码的目的。

可以用模板代替 成员函数输入输出参数、成员变量的类型。

即把类型定义为参数， 从而实现了真正的代码可重用性。模

，如 T1；
了
现阶段写的代码，但是根据具体的类型，编译器会自动生成"包

```
max(0.4,:asafsf)=0.4
max(bbbbb,:asafsf)=bbbbb
```

2.3 示例代码解读

A template<typename T1> 是声明这种"泛的类型"代号的语句，比

B template<typename T1> 这个语句必须

C 示例代码中使用了前向声明，是为了让读者区分出"声明"与"实现"

D 5个testcase中，第二个编译不通过，会报错。这里调用函数的过
float和int。然而我们的实现并不支持这种类型。

3. 类模板

3.1 简介

类模板指的是类级别的类型(而不是成员函数级别的)，作用域覆盖整个类
型。

当然，如果不想对整个类全局的使用模板，也可以仿照"模板函数"中的方
样的，详见实验环境D 。

3.2 原理与实现

A 在类的声明之前，加上template<typename T1> 语句，做类级别

B 如果成员函数需要用模板类型，

a）如果成员函数在类声明之内进行实现，正常实现即可。（如示

b）如果成员函数在类声明之外进行实现（我们经常用这种），要

（如示例代码中，test函数、max函数）

要在函数实现前，加上类似template<typename T1>，
你加T1之后的类。

3.3 示例代码及运行结果

```
1 #include <iostream>
2
3 template <typename T1>
4 class Tzl{
5     public:
6     T1 max(T1 a, T1 b);
7     T1 num;
8     int test();
9     int printf_test(T1 num){
10        std::cout << "printf_test " << num << std::endl;
11        return 0;
12    }
13 };
14
15 template <typename T1>
16 T1 Tzl<T1>::max(T1 a, T1 b){
17    if (a > b)
18    {
19        std::cout << "max(" << a << ",:" << b << ")="  << a << std::en
```

如string、int、float都是可以T1这种类型。

两个部分的实现方法，不要理解为template只能前向声明
程中，编译器会做"sh实参推演"的过程，会把0.6和5认为是

类，可以用模板代替 成员函数输入输出参数、成员变量的类

方式，只对一两个成员函数使用模板，使用方法和"模板函数"一

的声明。

例代码中，printf_test函数）
加点语法。

在类::改成类<T1>::。不然实现部分编译的时候，编译器不认识

```
20          return a;
21      }
22      else if (b >= a)
23      {
24          std::cout << "max(" << a << ",:" << b << ")=" << b << std::end
25          return b;
26      }
27  }
28
29  template <typename T1> int Tzl<T1>::test()
30  {
31      std::cout << "test" << std::endl;
32      return 0;
33  }
34
35  /*template <typename T1>
36  int Tzl<T1>::printf_test(T1 num){
37      std::cout << "printf_test " << num << std::endl;
38      return 0;
39  }*/
40
41  int main(){
42      Tzl<int> tzl_int;
43      tzl_int.max(0,1);
44
45      Tzl<float> tzl_float;
46      tzl_float.max(1.11,1);
47
48      tzl_float.printf_test(5.5);
49      tzl_float.test();
50  }
```

```
max(0,:1)=1
max(1.11,:1)=1.11
printf_test 5.5
test
```

3.4 示例代码解释

A test函数和max函数的实现部分，函数名的写法不一样，其实含义都是

三、拓展的用法

1、模板类和重载函数一起使用

其实很简单：

模板可以对类型不同但实现相同的函数，实现代码复用。对于类型不同、
名，一样的方式调用。这样调用方便。

博客中的描述"重载函数"，我觉得这里他说的有问题，其实不是重载函数，
数。

不过他的做法还是可以借鉴一下的。

详见参考资料C 中"模板类和重载函数一起使用"部分

```
l;
```

一样的（详见参考资料D）

实现稍微不同、但功能一致的函数，我们可以起一样的函数

就是写了一个函数名相同、函数参数类型不一样的两个函

2、template<typename A，typename B>

template是声明各模板的关键字，表示声明一个模板，模板参数可以是一个，

例如：template<typename A，typename B> 相当于template<typenar

在实现和使用的时候，要加个myClass<T1,T2>:: 例如：（详见参考资料c）

```
template<typename T1,typename T2>

class myClass{

private:

    T1 I;

    T2 J;

public:

    myClass(T1 a, T2 b);//Constructor

    void show();

};

//这是构造函数

//注意这些格式

template <typename T1,typename T2>

myClass<T1,T2>::myClass(T1 a,T2 b):I(a),J(b){}

//这是void show();

template <typename T1,typename T2>

void myClass<T1,T2>::show()

{

    cout<<"I="<<I<<", J="<<J<<endl;

}
```

四、实验环境：

work@nj01-nlp-test01.nj01.baidu.com:/home/work/tianzhiliang/test/cpp

A 模板函数：func/

B 类模板：class/

C 类模板(分到class.cpp class.h中实现，编译没过，有时间看看原因)：class_

D 类中对成员变量使用模板函数：class_func/

五，参考资料

也可以是多个。
me A>；template <typename B>

/template

separate/

A wikipedia http://zh.wikipedia.org/wiki/%E6%A8%A1%E6%9D%BF_(C%2B%2
B 博客不错 简单 http://www.360doc.com/content/09/0403/17/799_3011262.
C 博客，演示了"模板类和重载函数一起使用" http://blog.csdn.net/hackbute
D 详细、系统 http://www.cnblogs.com/gw811/archive/2012/10/25/2738

2B)
.shtml
eer1/article/details/6735704
3929.html