Subject: Re: THISBACK and function-overloading
Posted by mrjt on Fri, 25 Jul 2008 09:49:24 GMT
View Forum Message <> Reply to Message

I've had this problem, these are two solutions that I know of:
1) Obvious. Resolve the ambiguity by naming one of you functions something else.
2) Resolve the ambiguity caused by the compiler not being able to determine which version of the callback function to use. You could do this by casting the function pointer correctly, but this would be very ugly.

My prefered solution is to define callback0 somewhere (it's identical to callback, except for the name):

```
template <class OBJECT, class METHOD>
Callback callback0(OBJECT *object, void (METHOD::*method)()) {
 return callback(object, method);
}

template <class OBJECT, class METHOD>
Callback callback0(const OBJECT *object, void (METHOD::*method)() const) {
 return callback(object, method);
}

inline Callback callback0(void (*fn)()) {
 return callback(fn);
}
#define THISBACK0(x)              callback0(this, &CLASSNAME::x)
```

And then it is available to use for resolving this sort of ambiguity.
Perhaps there is a better solution available that I haven't thought of?