giannis_tsakiris

mostly programming-related stuff

C++: calling a member function poin

Posted on September 7, 2012 by giannis

This article is about calling a member function (method) througl
tricked me for a while, and I think this may be useful to others.

I will start with a small introduction and gradually get to the poi
article to see how it's done!

Calling a "global" function pointer

In good old C, the following code is probably familiar to most s

```
#include <stdio.h>

void func1( void (*func)() ) {
func();
}

void func2() {
printf("Goodbye world!n");
}

int main() {
func1(func2);
}
```

Now if we compile this and execute it we'll get of course the fol

```
giannis@goofy:~$ gcc fpointer.c
giannis@goofy:~$ ./a.out
Goodbye world!
```

The slightly strange syntax *void (*func)()* in func1()'s declarati
specifically declares that "func is pointer to a function with no a

Similarly, *void (*func)(int)* declares a pointer to a function that t
() declares a pointer to a function that takes no ar guments and r

Function pointers are like data pointers, except the point to exec
way in C to encapsulate and delegate behaviors or logic.

Now, let's take the same scenario only this time within the scop

```
#include <stdio.h>

class MyClass {
```

```
private:

void func1( void (*func)() ) {
func();
}

void func2() {
printf("Goodbye world!n");
}

public:

void method() {
func1(func2);
}
};

int main() {
MyClass myObject;
myObject.method();
}
```

At least for me, this made sense… But what happens if we try to

```
giannis@goofy:~$ g++ fpointer.cpp
fpointer.cpp: In member function void MyClass::method():
fpointer.cpp:19: error: no matching function for call to MyCla
fpointer.cpp:8: note: candidates are: void MyClass::func1(voi
```

What the error message says, more or less, is that that func1() ex
however we tried to pass an "unresolved overloaded function typ

Okay, that's weird. func1() expects an ar gument of type *void (*

Well, not exactly. func2 is actually a *MyClass::void (*)()*, that i
func1() was expecting a *void (*)()*, which is a pointer to a functi

To fix this, we need to need to slightly modify the code in severa

First of all, the declaration of func1(). As we said the function w
Therefore it has to be altered accordingly:

```
void func1( void (MyClass::*func)() ) {
func();
}
```

Furthermore we need to change the way the function pointer is d
func(). Remember that all the compiler knows is the address of a
class. When dereferencing a member function pointer , we also n
called, in that case *this* holds a reference to this object, which is
now:

```
void func1( void (MyClass::*func)() ) {
(*this.*func)();
}
```

The *(*this.*func)();* statement just instructs the compiler to call

And we're almost done, we need to make one final small change
makes clear that func2 is a method of MyClass and not a "stray"

```
void method() {
func1(&MyClass::func2);
}
```

Now it should compile and work as expected:

```
giannis@goofy:~$ g++ fpointer.cpp
giannis@goofy:~$ ./a.out
Goodbye world!
```

Let's put it all together:

```
#include <stdio.h>

class MyClass {

private:

void func1( void (MyClass::*func)() ) {
(*this.*func)();
}

void func2() {
printf("Goodbye world!n");
}

public:

void method() {
func1(&MyClass::func2);
}
};

int main() {
MyClass myObject;
myObject.method();
}
```

I hope this helped 🙂

This entry was posted in Uncategorized. Bookmark the permalink.

4 Responses to *C++: calling a member function po*

Steve *says:*
May 21, 2013 at 1:07 am

Wow! This saved me! What a convoluted way of getting it to worl
will study this syntax more closely and try to understand why it w

Reply

Steve Duff *says:*
April 28, 2014 at 7:00 am

Excellent! Thanks much for this: it's as clear an explanation as is p

The short story is really basic semantics: the first ("obvious") way
function expects to have a class context to work in; a "this". The r

I would say the error message could be a little less obtuse IMO. So
overloaded function type".

Reply

Anna *says:*
August 21, 2014 at 4:47 pm

Best explanation possible. I was forgetting the & and all of today I
all makes sense!

Reply

Niccola *says:*
March 29, 2015 at 10:05 pm

Fantastic explanation. I was struggling with this for hours. Thank y

Reply

giannis_tsakiris
Proudly powered by WordPress.