

Forum:[Hack Forums](#)

Thread: [How do i get the address of a member function from inside a class?](#)

jammy47 - March 16, 2017, 1:46 pm

Hi guys, I am trying to get the address of a member function in a class and store it a void pointer within the class. How do i do that?

aeiro - March 16, 2017, 8:47 pm

maybe this could help. I didn't actually read it over but it sounds similar.

<http://stackoverflow.com/questions/12988320/store-function-pointers-to-any-member-function>

jammy47 - March 17, 2017, 9:35 am

(Mar 16, 2017 08:47 PM)aeiro Wrote: [\[CITE\]](#) maybe this could help. I didn't actually read it over but it sounds similar.

<http://stackoverflow.com/questions/12988320/store-function-pointers-to-any-member-function>

Well, thats not exactly the thing i was looking for, but i got my answer from stackexchange.

It seems you can get the address of a member function like this.....

```
void * a;
```

```
a = &A::func();
```

This gets the address of func from within class A. So yeah, my problem was solved.

Goodies - March 17, 2017, 9:40 am

Well, you kind of can't.

<http://www.dietmar-kuehl.de/mirror/c++-faq/pointers-to-members.html#faq-33.7>

What are you trying to do? The general way of getting a member function is by typedefing a member function pointer (or using auto if you're brave enough).

PHP Code

```
<?php
#include <iostream>
#include <functional>
#include <string>

class Foo
{
public:
    typedef void (Foo::*vfuncptr)(void);
    Foo(int i) : m_i{i}
    {}

    void print(void)
    {
        std::cout << m_i << std::endl;
```

```

    }

private:
    int m_i;
};

int main(int argc, char **argv)
{
    Foo f1(7);
    Foo f2(3);

    Foo::vfuncptr ptr = &Foo::print;
    // converting 'ptr' to a (void*) is a no-no.

    (f1.*ptr)();
    (f2.*ptr)();

    return 0;
}

```

There's probably a better way of doing whatever you're trying to do... like, for example, `std::function`. This would be way better.

PHP Code

```

<?php
int main(int argc, char **argv)
{
    Foo f1(7);
    Foo f2(3);

    std::vector<std::function<void()>> vec;
    vec.push_back(std::bind(&Foo::print, &f1));
    vec.push_back(std::bind(&Foo::print, &f2));

    for(auto func : vec)
    {
        func();
    }
    return 0;
}

```

or lambdas <3

PHP Code

```

<?php
std::vector<std::function<void()>> vec;
vec.push_back([&f1]() { f1.print(); });
vec.push_back([&f2]() { f2.print(); });

```