

- Zongxiong Chen

# The Most Vexing Parse

---

# Outline

- root cause
- function declarations
- cases study
- how to fix it

## Root cause

- In C++, pretty much anything can be parsed as a function declaration

# function declarations

- `int foo(double d);`
- `int foo(double (d));`      `// same as above; parens around d are ignored`
- `int foo(double);`      `// same as above; parameter name is omitted`
- `int bar(double (*pf)());`      `// takes a pointer to a function a parameter`
- `int bar(double pf()) ;`      `// same as above; pf is implicitly a pointer`
- `int bar(double ());`      `// same as above; parameter name is omitted`

```
64#include <iostream>
65int f1(double a);
66int f2(double (a));
67int f3(double);
68
69int f1(double a) {
70    std::cout << "f1(a): " << a << std::endl;
71    return 0;
72}
73int f2(double (a)) {
74    std::cout << "f2(a): " << a << std::endl;
75    return 0;
76}
77int f3(double a) {
78    std::cout << "f3(a): " << a << std::endl;
79    return 0;
80}
81int main() {
82    std::cout << "Function Declarations" << std::endl;
83    f1(1.0);
84    f2(2.0);
85    f3(3.0);
86    return 0;
87}
```

```
77~/emacs.d/notes $ ./ex1
78Function Declarations
79f1(a): 1
80f2(a): 2
81f3(a): 3
```

## A degenerated case

- `int foo(); // no paramters and return an integral`

# Case 1

```
45#include <iostream>
46
47class A {
48public:
49    A(){ std::cout << "A() is called" << std::endl; }
50    A(int) { std::cout << "A(int) is called" << std::endl; }
51};
52
53int main() {
54    std::cout << "The Most Vexing Parse" << std::endl;
55    A a1();
56    std::cout << "-----" << std::endl;
57    A a2(5);
58    std::cout << "-----" << std::endl;
59    A a3;
60    std::cout << "-----" << std::endl;
61    return 0;
62}
```

# clang

```
109 ~/.emacs.d/notes $ g++ -v
110 Configured with: --prefix=/Applications/Xcode.app/Contents/Developer/usr --with-gxx-include-dir=/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX10.14.sdk/usr/include/c++/4.2.1
111 Apple LLVM version 10.0.0 (clang-1000.11.45.5)
112 Target: x86_64-apple-darwin18.2.0
113 Thread model: posix
114 InstalledDir: /Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefault.xctoolchain/usr/bin
115 ~/.emacs.d/notes $ make
116 g++ ex1.cpp -o ex1
117 ex1.cpp:55:7: warning: empty parentheses interpreted as a function declaration [-Wvexing-parse]
118     A a1();
119         ^~
120 ex1.cpp:55:7: note: remove parentheses to declare a variable
121     A a1();
122         ^~
123 1 warning generated.
124 ~/.emacs.d/notes $ ./ex1
    The Most Vexing Parse
    -----
    A(int) is called
    -----
    A() is called
    -----
```



# gcc

```
12~ $ g++ -v
13Using built-in specs.
14COLLECT_GCC=/usr/bin/g++
15COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/5/lto-wrapper
16Target: x86_64-linux-gnu
17Configured with: ../src/configure -v --with-pkgversion='Ubuntu 5.4.0-6ubuntu1~16.04.11' --with-bugurl=file:///usr/share/doc/gcc-5/README.Bug\
s --enable-languages=c,ada,c++,java,go,d,fortran,objc,obj-c++ --prefix=/usr --program-suffix=-5 --enable-shared --enable-linker-build-id --l\
ibexecdir=/usr/lib --without-included-gettext --enable-threads=posix --libdir=/usr/lib --enable-nls --with-sysroot=/ --enable-clocale=gnu --\
enable-libstdcxx-debug --enable-libstdcxx-time=yes --with-default-libstdcxx-abi=new --enable-gnu-unique-object --disable-vtable-verify --ena\
ble-libmpx --enable-plugin --with-system-zlib --disable-browser-plugin --enable-java-awt=gtk --enable-gtk-cairo --with-java-home=/usr/lib/jv\
m/java-1.5.0-gcj-5-amd64/jre --enable-java-home --with-jvm-root-dir=/usr/lib/jvm/java-1.5.0-gcj-5-amd64 --with-jvm-jar-dir=/usr/lib/jvm-expo\
rts/java-1.5.0-gcj-5-amd64 --with-arch-directory=amd64 --with-ecj-jar=/usr/share/java/eclipse-ecj.jar --enable-objc-gc --enable-multiarch --\
disable-werror --with-arch-32=i686 --with-abi=m64 --with-multilib-list=m32,m64,mx32 --enable-multilib --with-tune=generic --enable-checking=\
release --build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=x86_64-linux-gnu
18Thread model: posix
19gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.11)
20~ $ g++ 1.cpp -o ex1
21~ $ ./ex1
The Most Vexing Parse
-----
A(int) is called
-----
A() is called
-----
```

## solution 1

```
45#include <iostream>
46
47class A {
48public:
49    A(){ std::cout << "A() is called" << std::endl; }
50    A(int) { std::cout << "A(int) is called" << std::endl; }
51};
52
53int main() {
54    std::cout << "The Most Vexing Parse" << std::endl;
55    A a1 {};
56    std::cout << "-----" << std::endl;
57    A a2(5);
58    std::cout << "-----"
141~/.emacs.d/notes $ ./ex1
142The Most Vexing Parse
143A() is called
144-----
145A(int) is called
146-----
147A() is called
148-----
60    std::cout << "-----"
61    return 0;
62}
63
```

## Case 2

```
89#include <iostream>
90class B;
91class A {
92public:
93    explicit A() { std::cout << "A() is called" << std::endl; }
94    explicit A(const B &b) { std::cout << "A(const B &) is called" << std::endl; }
95    void foo() { std::cout << "A.foo is called" << std::endl; }
96};
97class B {
98public:
99    B() { std::cout << "B() is called " << std::endl; }
100};
101
102
103int main() {
104    A a(B());
105    a.foo();
106    return 0;
107}
```

# Compiling messages

```
~/emacs.d/notes $ make
```

```
g++ -std=c++11 ex1.cpp -o ex1
```

```
ex1.cpp:104:6: warning: parentheses were disambiguated as a function declaration [-Wvexing-parse]
```

```
    A a(B());
```

^~~~~

```
ex1.cpp:104:7: note: add a pair of parentheses to declare a variable
```

```
    A a(B());
```

^

( )

```
ex1.cpp:105:4: error: member reference base type 'A (B (*)())' is not a structure or union
```

```
    a.foo();
```

~^~~~

```
1 warning and 1 error generated.
```

```
make: *** [make] Error 1
```



## Case 2

```
89#include <iostream>
90class B;
91class A {
92public:
93    explicit A() { std::cout << "A() is called" << std::endl; }
94    explicit A(const B &b) { std::cout << "A(const B &) is called" << std::endl; }
95    void foo() { std::cout << "A.foo is called" << std::endl; }
96};
97class B {
98public:
99    B() { std::cout << "B() is called " << std::endl; }
100};
101
102
103int main() {
104    A a(B());
105    a.foo();
106    return 0;
107}
```

Compared with function declaration:  
It takes a parameter of type B and return an object of type A

# Solution

```
89#include <iostream>
90class B;
91class A {
92public:
93    explicit A() { std::cout << "A() is called" << std::endl; }
94    explicit A(const B &b) { std::cout << "A(const B &) is called" << std::endl; }
95    void foo() { std::cout << "A.foo is called" << std::endl; }
96};
97class B {
98public:
99    B() { std::cout << "B() is called " << std::endl; }
100};
101
102
103int main() {
104    A a(B{});
105    a.foo();
106    return 0;
107}
```

using curly brace instead

```
06~/.emacs.d/notes $ ./ex1
B() is called
A(const B &) is called
A.foo is called
~/.emacs.d/notes $
```

## Solution 2

```
88
89#include <iostream>
90class B;
91class A {
92public:
93    explicit A() { std::cout << "A() is called" << std::endl; }
94    explicit A(const B &b) { std::cout << "A(const B &) is called" << std::endl; }
95    void foo() { std::cout << "A.foo is called" << std::endl; }
96};
97class B {
98public:
99    B() { std::cout << "B() is called " << std::endl; }
100};
101
102
103int main() {
104    A a((B()));
105    a.foo();
106    return 0;
107}
```

add extra parentheses

## Solution 3

```
89#include <iostream>
90class B;
91class A {
92public:
93    explicit A() { std::cout << "A() is called" << std::endl; }
94    explicit A(const B &b) { std::cout << "A(const B &) is called" << std::endl; }
95    void foo() { std::cout << "A.foo is called" << std::endl; }
96};
97class B {
98public:
99    B() { std::cout << "B() is called " << std::endl; }
100};
101
102
103int main() {
104    B b;
105    A a(b);
106    a.foo();
107    return 0;
108}
```

introduce a helper variable



## How to fix

- In C++11, we can use curly braces {} instead of parentheses ()
- Before C++11
  - add extra parentheses around the constructor
  - introduce a helper variable

# Reference

- Effective STL (Item 6)
- <https://www.fluentcpp.com/2018/01/30/most-vexing-parse/>
- [https://en.wikipedia.org/wiki/Most\\_vexing\\_parse](https://en.wikipedia.org/wiki/Most_vexing_parse)
- <https://mbevin.wordpress.com/2012/11/16/uniform-initialization/>