

[首页](#) [资讯](#) [精华](#) [论坛](#) [问答](#) [博客](#) [专栏](#) [群组](#) [更多](#) ▼  
[您还未登录!](#) [登录](#) [注册](#)

## 横行青海夜带刀

- [博客](#)
- [微博](#)
- [相册](#)
- [收藏](#)
- [留言](#)
- [关于我](#)



### C语言插件机制(上)

- 博客分类:
- [UNIX相关](#)

[CC#C++LinuxEmacs](#)

## 前言

插件机制可以使得应用程序在发布之后, 在不经重新编译的情况下修改应用程序的行为, 这种形式使得应用的框架比较小巧, 也可以给用户一些自由(不是完全的自由, 有一定的限制)。Java中, 使用\*.jar或者其他的脚本引擎都可以完成这样的工作, 如Eclipse那样。在C语言中, 当然可以使用脚本引擎来实现, 比如emacs, 内置一个lisp的引擎, 用户可以自己为emacs写脚本, 访问emacs环境的一些组件, 从而定制emacs. 这里要讨论的无需使用脚本引擎, 而是用C语言访问动态链接库来实现。

## Linux下的动态库

Linux环境中, 与windows下一样, 函数库有两种方式: 静态库和动态库, 静态库参与连接, 由于要将目标代码.o与lib中的函数符号合并在一起, 所以最终生成的可执行文件较大。一般以.a结尾。如libxxx.a。而动态库(共享库)则不参与编译, 只是在运行时才加载如内存, 且仅加载一次, 因此最终的可执行文件较小。事实上, 当一个可执行文件需要运行动态库中的函数时, 系统会在内存中查找, 如果已经加载, 则直接调用, 否则才做一次加载, 动态库的结尾一般为.so, 如libxxx.so。

Linux为动态库的访问提供了4个API, 分别为dlopen, dlopen, dlsym和dlclose。这些API的原型定义在文件dlfcn.h中, 其实现则分别对应有两个库文件(静态库libdl.a和动态库libdl.so)。

- dlopen加载动态库, 并返回句柄
- dlopen如果加载, 访问符号出错, 可以通过此接口获得详细的描述
- dlsym返回一个动态库中的符号, 即通过函数名获得此函数的指针
- dlclose完成之后, 释放dlopen返回的句柄

动态库的使用较为简单, 比如动态库的名称为plugin.so, 其中包含这样的函数原型:

C代码 ☆

```
1. int func(int a, int b);
```

可以看下一个例子：

C代码 ☆

```
1. #include <dlfcn.h>
2.
3. //句柄
4. void *flib;
5.
6. //入口函数原型
7. int (*pfunc)(int a, int b);
8.
9. //错误信息字符串
10. char *error_message;
11.
12. int plugin_test(){
13.     int a = 1, b = 4;
14.     int result = 0;
15.
16.     //加载plugin.so, 以RTLD_LAZY方式
17.     flib = dlopen("/home/juntao/.libs/plugin.so", RTLD_LAZY);
18.     error_message = dlerror();
19.
20.     if(error_message){
21.         return (-1);
22.     }
23.
24.     //找到函数名为func的函数, 返回其指针
25.     *(void **)&pfunc = dlsym(flib, "func");
26.     error_message = dlerror();
27.     if(error_message){
28.         return (-1);
29.     }
30.
31.     //调用pfunc指向的指针, 及func函数
32.     result = pfunc(a, b);
33.
34.     //释放
35.     int code = dlclose(flib);
36.     error_message = dlerror();
37.
38.     if(error_message){
39.         return (-1);
40.     }
41.
42.     return 0;
43. }
```

## 编译运行

假设plugin.so的源文件为plugin.c，内容为：

C代码 ☆

```
1. //file plugin.c
2.
3. int func(int a, int b){
4.     int c = 0;
5.     c = 3*a + 4*b + 6;
6.
7.     return c;
8. }
```

我们将这个.c文件编译为.so，命令如下：

```
//生成plugin.o
$gcc -c -fpic plugin.c
```

```
//生成plugin.so
$gcc -shared -lc -o plugin.so plugin.o
```

将动态库访问部分的代码存为plugintest.c，然后使用下列命令编译：

C代码 ☆

```
1. $gcc -o plugintest plugintest.c -ldl
```

-ldl意思是，使用库libdl.so，linux下访问搜索路径内的库文件无需加lib前缀。

将生成的plugin.so放入路径/home/juntao/.libs/，然后运行plugintest。

```
[juntao@juntao plugintest]$ make
gcc -c -fpic plugina.c
gcc -shared -lc -o plugina.so plugina.o
[juntao@juntao plugintest]$ ls
Makefile  plugina.c  plugina.o  plugina.so  plugintest  plugintest.c
[juntao@juntao plugintest]$ cp plugina.so ~/.libs/
[juntao@juntao plugintest]$ ./plugintest
result = 25
[juntao@juntao plugintest]$ cat Makefile
all: plugintest plugina.so
plugina.so: plugina.o
        gcc -shared -lc -o plugina.so plugina.o
plugina.o: plugina.c
        gcc -c -fpic plugina.c
plugintest: plugintest.c
        gcc -o plugintest plugintest.c -ldl

clean:
        rm -f *.o
        rm -f *.so
[juntao@juntao plugintest]$
```

./plugintest


result = 25


好了，这一次先介绍一些基础知识，相信在此基础上，很多朋友都可以自己设计出一些简单实用的支持“插件”的应用来了，我们下一次详细讨论一个更实际一些的例子，一个计算器的实现，这个计算器只有简单的框剪，所有的运算都通过插件来实现。用户可以通过配置文件来定制插件的路径，入口等信息。



- [查看图片附件](#)

ハルジオンプログラム講座

知識ゼロ 初心者のための Web プログラマー 養成講座





分享到:  

[C语言插件机制\(下\)](#) | [JavaScript内核系列 第9章 函数式的Java ...](#)

- 2010-08-17 23:08
- 浏览 4024
- [评论\(2\)](#)
- 论坛回复 / [浏览](#) (2 / 3555)
- 分类:[编程语言](#)
- [相关推荐](#)

评论

- 2 楼 [生活小丑](#) 2010-09-01  
不错~没试过，回去试试~
- 1 楼 [shuiguozheng](#) 2010-09-01  
最近爱上c 了

发表评论



您还没有登录,请您登录后再发表评论



abruzzi

- 浏览: 253923 次
- 性别:
- 来自: 西安
- 我现在离线

最近访客 [更多访客>>](#)



[max20120](#)



[橙臣1314](#)



[lymanloo](#)



[yghjoe](#)

文章分类

- [全部博客 \(84\)](#)
- [计算机科学与技术 \(48\)](#)
- [生活&文化 \(4\)](#)
- [其他 \(5\)](#)
- [UNIX相关 \(8\)](#)
- [编译技术 \(5\)](#)
- [算法 \(1\)](#)
- [Web2.0 \(12\)](#)
- [\[发布至博客园首页\] \(0\)](#)
- [\[随笔分类\] 计算机科学 \(2\)](#)
- [\[网站分类\] java \(0\)](#)
- [\[随笔分类\] java技术 \(0\)](#)
- [\[随笔分类\] 脚本语言 \(0\)](#)
- [JavaScript内核 \(8\)](#)

社区版块

- [我的资讯](#) (0)
- [我的论坛](#) (340)
- [我的问答](#) (49)

存档分类

- [2013-05](#) (1)
- [2012-02](#) (4)
- [2012-01](#) (1)
- [更多存档...](#)

最新评论

- [进退取舍](#): 谢谢, 这个用上了!!  
[Java 一个线程池的示例](#)
- [pb\\_water](#): 感谢楼主, 打算买楼主的书, 支持一下, 楼主功德无量  
[JavaScript内核系列第0版整理稿下载](#)
- [lancezhcj](#): 有图会直观的多呢, 再摸索摸索  
[有限自动机与建模](#)
- [hsmsyy](#): 这里应该是原创了吧, 楼主我觉得闭包的作用: 实现面向对象。有待商 ...  
[JavaScript内核系列 第7章 闭包](#)
- [wll52](#): 在应用退出之前, 需要释放连接 con.disconnect() ...  
[使用smack与GTalk通信](#)

---

声明: ITeye文章版权属于作者, 受法律保护。没有作者书面许可不得转载。若作者同意转载, 必须以超链接形式标明文章原始出处和作者。  
© 2003-2015 ITeye.com. All rights reserved. [ 京ICP证110151号 京公网安备110105010620 ]