

编程论坛
bbs.bccn.net

软件需要验证码识别？

登录 | 注册 | 用户控制面板 | 搜索 | 道具 | 短消息 | 帮助

编程论坛 → 开发语言 → 『 C语言论坛 』 → 支持插件的程序设计

我的收件箱(0)

欢迎加入我们，一同切磋技术。 登录后广告即消失

用户名：

[注册](#) [忘记密码](#)

密 码：

登录

简约中式装修

c++程序设计

c语言入门

青岛招聘

c语言编程基础

c++编程基础

c语言编程入门

青岛黄岛房价

ios开发工程师

c++教程

c++游戏开发

我要找工作

编程入门教程

艺术漆

ee留学

发表文章

回复文章

共有 921 人关注过本帖

标题：支持插件的程序设计

[只看楼主] [收藏]

dorainm





等级：新手上路

帖子：17

专家分：0

注册：2007-7-31

 短信  好友  信息  博客

问题点数：0 回复次数：2 楼主

支持插件的程序设计

支持插件的程序设计

作者：dorainm

看到很多软件，尤其老外的，大型软件，异常支持插件功能
主程序简洁了不说，灵活性，
而且貌似老外闲人很多，帮忙编写支持其它功能的新插件
逐渐，该软件的功能就变得完善异常

插件有2种存在方式：

1、脚本
一个插件，就是一段脚本，主程序读取某个脚本，需要解释它，
然后再有选择性、逻辑性地执行早已编译主程序内的相对应的二进制代码！
脚本不许要系统的任何支持(系统：哼，我不给你 I/O 操作)，也跟系统环境没有关系
编写这种脚本只需要一个文本编辑器，例如 vim 或 emacs 就可以了
但执行效率会稍稍逊色，而且脚本的设计代码容易暴露
针对陌生的脚本插件，如果主程序拥有足够的容错能力，还是能够避免程序崩溃
(dorainm：设计个逻辑无限循环的恶意插件，让主程序瞎耗去吧)

2、二进制

这种脚本，是一种链接库，由各种函数的已经编译好的二进制字符组成而且必须要求操作系统能够支持一个程序调用链接库的功能（类似win32的dll动态链接库，*nix的.o/.so共享对象）还需要针对不同操作系统，软、硬件环境编译出相应的二进制代码，当然，编写插件的时候，需要额外的编译器，如 gcc 这种插件执行效率相对比较高，源码相对也比较不透明因为它本身就是已经被编译了的可执行二进制代码但是一旦遇到恶意或者劣质的插件，主程序就会崩溃！

下面我们来介绍第二种插件的实现思想与方法，一则，学会了第二种，第一种方法无非是编译器的工作变成了主程序的任务二来，我们可以学习动态链接库的程序写法它即能减少程序的大小，也能提高程序编译过程中的时间花费，更能提高程序灵活性，比如升级、补丁；不可能我们的软件只有一个可执行的二进制文件，（比如cs，就一个 1.3G 大的 .exe，升级方面不说，如果作者觉得 AWP 的甩枪需要修正，修改源码，让电脑编译，我 阿富汗 旅游！！旅游回来，汗，修改源码的时候，不小心多加了个逗号，重新修正，再编译，得，这回去 伊拉克 旅游...）

关于插件的思想，是dorainm自己想的不知道与当前各大软件的设计是否相似，如果有冗余或错误的地方，还望斧正

环境：
系统 linux 2.6.15
编译器 gcc 4.1.0
文本编辑 emacs 21.4.1

关于动态链接库的介绍，引用 雨亦奇 的 <LINUX系统中动态链接库的创建与使用> 大家都知道，在WINDOWS系统中有很多的动态链接库（以.DLL为后缀的文件，DLL即Dynamic Link Library）。这种动态链接库，和静态函数库不同，它里面的函数并不是执行程序本身的一部分，而是根据执行程??要按需装入，同时其执行代码可在多个执行程序间共享，节省了空间，提高了效率，具备很高的灵活性，得到越来越多程序员和用户的青睐。那么，在LINUX系统中有无这样的函数 库呢？答案是肯定的，LINUX的动态链接库不仅有，而且为数不少。在/lib目录下，就有许多以.so作后缀的文件，这就是LINUX系统应用的动态链接库，只不过与WINDOWS叫法不同，它叫so，即Shared Object，共享对??。（在LINUX下，静态函数库是以.a作后缀的）X-WINDOW作为LINUX下的标准图形窗口界面，它本身就采用了很多的动态链接库（在/usr/X11R6/lib目录下），以方便程序间的共享，节省占用空间。著名的APACHE网?服务器，也采用了动态链接库，以便扩充程序功能。你只需将PHP动态链接库拷到其共享目录，修改一下配置，APACHE就可以支持PHP网页了。如果你愿意，可以自己编写动态链接库，让APACHE支持你己定义的网页格式。这就是动态链接的好处。

我们是做插件的！！

现在我们明白，主程序是一个可执行的二进制，但是它并不是完整的它的某些部分与功能，需要插件来补充！！

对于插件来说，它们实现各自特殊的功能，但是它们的设计，要针对主程序的要求，给主程序调用，提供统一意义的函数与接口（比如主程序读取某个插件名字可能调用 get_plug_name 函数，读取作者，可能调用 get_plug_author，每个插件必须拥有这些函数，做相同意义的事情，get_plug_author就提供作者，而不是插件的名字）

现在，我们这里提供一个最最简单的插件头文件，任何插件编写者，可以根据这个头文件，填充自己的功能，实现自己的插件

```
/*d_plug.h*/
#ifndef __D_PLUG_H
#define __D_PLUG_H
```

```
#ifdef SHARED
int ( *d_plug )( int x, int y, int *res );
#else
int d_plug( int x, int y, int *res );
#endif

#endif
```

这个头文件说明，插件应该有一个 `d_plug`函数，它的参数是这样这样的
然后在说明文档中着重解释一下主程序怎么使用它：
比如，主程序提供`x,y`两个数字，让插件来运算，返回给`res`

任何插件的编写者都明白来！！
那作为一员，我们现在编写自己的第一个插件

```
/*d_add.c*/
#include <stdio.h>
#include "d_plug.h"

int d_plug( int x, int y, int *res )
{
    *res = x+y;
    return 0;
};
```

够简单，而且让读者觉得简单得莫名其妙！
我们把 `x`和 `y` 相加后，交给了要作为返回的 `res`

如果觉得有些迷茫，我们看看主程序

```
/*main.c*/
#include <stdio.h>
/*Linux下使用动态链接库，源程序需要包含dlfcn.h头文件，此文件定义了调用动态链接库的函数的原型*/
#include <dlfcn.h>

#define SHARED

/*插件的头文件*/
#include "d_plug.h"

/*显示程序信息*/
void disp_logo( void )
{
    printf(" a plug-in simple, by dorainm, dorainm@gmail.com\n");
    return;
};

/*显示语法信息*/
void disp_usage( char *app_name )
{
    printf( "usage : %s [plug-in file name (*.so) ]\n", app_name );
    return;
};

int main( int argc, char *argv[] )
```

```
{
int res; /*用于装运算返回值*/
int x=5; /*第一个运算数,这里固定为5*/
int y=3; /*第二个运算数,当然我们可以让它们也用实参输入*/

char *so_filename; /*插件的文件名*/

void *dp; /*调用.so的句柄*/
char *error; /*错误信息*/

disp_logo();
if( argc<2 )
{
disp_usage( argv[0] );
return 1;
}

/*在实参中获取插件文件名*/
so_filename = argv[1];
printf("use shared object file : [ %s ]\n", so_filename );

/*装载该插件,动态链接库*/
dp = dlopen( so_filename, RTLD_LAZY );
/*返回为NULL,则装载失败*/
if ( dp==NULL )
{
fprintf( stderr, dlerror() );
return 2;
}

printf( "load plug [ %s ] successfully.\n", so_filename );

/*获取动态链接库中的 d_plug 函数地址,即获取插件的功能代码*/
d_plug = dlsym( dp, "d_plug" );
error = dlerror();
/*判断是否出错,比如该“不明插件”没有这函数*/
if ( error )
{
fprintf( stderr, error );
return 3;
}

/*执行插件的功能*/
d_plug( x, y, &res );
/*显示结果*/
printf(" the result to %d, %d : %d.\n", x, y, res );

/*卸载插件*/
dlclose( dp );

return 0;
};
```

我们可以看到,主程序通过第一个实参,装载该动态链接库(插件)
获取插件中的 d_plug 函数,然后根据d_plug头文件里面的规定
把两个运算数和保存结果值的地址 交与 d_plug 函数
然后显示结果!!

我们来编译主函数

编译时要采用-rdynamic选项与-ldl选项
以产生可调用动态链接库（插件）的执行代码

现在，我们编译我们可爱的第一个插件

```
[dorainm@localhost plug_simple]$ ls
d_add.c d_plug.h main.c
[dorainm@localhost plug_simple]$ cc -rdynamic -s -o simple main.c -ldl
[dorainm@localhost plug_simple]$ ls
d_add.c d_plug.h main.c simple
[dorainm@localhost plug_simple]$
```

编译插件源程序时选用-shared选项即可创建动态链接库

我们现在来运行我们的主程序

```
[dorainm@localhost plug_simple]$ ls
d_add.c d_plug.h main.c simple
[dorainm@localhost plug_simple]$ cc -shared -o d_add.so d_add.c
[dorainm@localhost plug_simple]$ ls
d_add.c d_add.so d_plug.h main.c simple
[dorainm@localhost plug_simple]$
```

我们根据语法，加入我们第一个参数：我们的 d_add.so 插件

```
[dorainm@localhost plug_simple]$ ./simple
a plug-in simple, by dorainm, dorainm@gmail.com
usage : ./simple [plug-in file name (*.so) ]
[dorainm@localhost plug_simple]$
```

```
[dorainm@localhost plug_simple]$ ./simple ./d_add.so
a plug-in simple, by dorainm, dorainm@gmail.com
use shared object file : [ ./d_add.so ]
load plug [ ./d_add.so ] successfully.
the result to 5, 3 : 8.
[dorainm@localhost plug_simple]$
```

看，主程序根据第一个插件，已经成功实现来加法运算！

信心！！

再接再厉，我们来编写第二个插件：实现减法
其实很简单，修改加法插件源码中的一个字符就可以

```
/*d_sub.c*/
#include <stdio.h>
#include "d_plug.h"

int d_plug( int x, int y, int *res )
{
    *res = x-y;
    return 0;
};
```

我们编译这个插件，试让主程序来接纳这个新伙伴

```
[dorainm@localhost plug_simple]$ ls
d_add.c d_add.so d_plug.h d_sub.c main.c simple
[dorainm@localhost plug_simple]$ cc --shared -o d_sub.so d_sub.c
[dorainm@localhost plug_simple]$ ls
```

```
d_add.c d_add.so d_plug.h d_sub.c d_sub.so main.c simple
[dorainm@localhost plug_simple]$ ./simple ./d_sub.so
a plug-in simple, by dorainm, dorainm@gmail.com
use shared object file : [ ./d_sub.so ]
load plug [ ./d_sub.so ] successfully.
the result to 5, 3 : 2.
[dorainm@localhost plug_simple]$
```

成功实现来第二个插件提供的功能，减法！

乘法、除法插件的源码，想必就不需要提供来，也是修改一个字符！
下面看它们实现的效果

```
[dorainm@localhost plug_simple]$ ls
d_add.c d_div.c d_plug.h d_sub.so simple
d_add.so d_mul.c d_sub.c main.c
[dorainm@localhost plug_simple]$ cc --shared -o d_mul.so d_mul.c
[dorainm@localhost plug_simple]$ cc --shared -o d_div.so d_div.c
[dorainm@localhost plug_simple]$ ls
d_add.c d_div.c d_mul.c d_plug.h d_sub.so simple
d_add.so d_div.so d_mul.so d_sub.c main.c
[dorainm@localhost plug_simple]$ ./simple ./d_mul.so
a plug-in simple, by dorainm, dorainm@gmail.com
use shared object file : [ ./d_mul.so ]
load plug [ ./d_mul.so ] successfully.
the result to 5, 3 : 15.
[dorainm@localhost plug_simple]$ ./simple ./d_div.so
a plug-in simple, by dorainm, dorainm@gmail.com
use shared object file : [ ./d_div.so ]
load plug [ ./d_div.so ] successfully.
the result to 5, 3 : 1.
[dorainm@localhost plug_simple]$
```

乘法、除法插件也实现来！

当然，我们还可以编写，
比如把 x当作成绩，y当作及格标准，res返回该学生成绩是否合格的判断成绩的插件

这样，一个主程序，不修改自己任何代码，就可以实现了各种各样的功能：）

真实的软件

各种插件可能由一些插件列表维护，
比如一个插件配置的文本文件 plug.conf，里面有插件对应的位置列表
或者把插件都放置在某个文件夹中，
比如plug-in文件夹里，就丢着为主程序提供各种功能的.so插件

主程序运行起来时，需要建立一张插件的表，存储插件列表
再读取相应的信息，给用户显示：
看，当前有这些这些插件
当用户选择其一，进行操作时，软件就调入该动态链接库
圆满完成用户预先想要的功能

这就是插件的奥秘！

	文件：plug_simple.tar.gz
	大小：5KB
	下载：下载

搜索更多相关主题的帖子: [程序设计](#) [插件](#)



北风网
1BEIFENG.COM
IT在线教育培训


包就业、包跳槽的编程开发课程来了!


10000月薪
轻松搞定!


 2007-07-31 03:00:18

 引用  回复  帖子操作 





anlogo







等级: 论坛游民
威望: 1
帖子: 293
专家分: 20
注册: 2007-7-20

 短信  好友  信息  博客


得分:0 第 2 楼

强,顶下~~~~~

 2007-07-31 08:34:56

 引用  回复  帖子操作 





gavin1024











等级: 新手上路
帖子: 1
专家分: 0
注册: 2012-6-7

 短信  好友  信息  博客

讲的很全，收了~~

 引用  回复  帖子操作 

2012-06-07 12:32:29



【推荐】全球顶尖大型工控、仿真、组态、CAD与GIS 100% VC++ 源码提供，无任何保留！

➡ 大型VC++工控仿真组态源码

➡ 大型VC++图形建模与仿真软件源码

➡ 大型CAD制图与打印VC++源码

➡ 大型GIS地理信息软件VC++源码

免费下载！

源码

[关于我们](#) | [广告合作](#) | [编程中国](#) | [清除Cookies](#) | [TOP](#) | [手机版](#)

编程中国 版权所有，并保留所有权利。
Powered by Discuz, Processed in 0.028336 second(s), 9 queries.
Copyright©2004-2015, BCCN.NET, All Rights Reserved

第8页 共8页

2015年11月10日 13:53