

[空间](#)

[博客](#)

[好友](#)

[相册](#)

[留言](#)

用户操作

[\[留言\]](#) [\[发消息\]](#) [\[加为好友\]](#)

订阅我的博客



will2ni的公告

文章分类

- [DIY](#)
- [Linux/HP-UX](#)
- [Perl](#)
- [Shell](#)
- [批处理](#)
- [软件测试](#)
- [随笔](#)

存档

[2010年01月\(2\)](#)
[2009年11月\(5\)](#)
[2009年10月\(1\)](#)



Perl读写Excel

[收藏](#)

Perl读写Excel

2009年02月(3)
2008年05月(1)

版 本: Perl 5.10.0

操作系统: Windows Vista

2009-02-05

- 1. 背景
- 2. 准备
 - 2.1 安装Perl
 - 2.2 安装Perl 模块
- 3. 实现
 - 3.1 Win32::OLE
 - 3.1.1 使用Win32::OLE
 - 3.2 Examples
 - 3.2.1 九九乘法表
 - 3.2.2 系统性能分析
- 4. 附录

背景

测试过程中对测试数据、测试结果处理时，经常会遇到如下几种情况：

- 输入为txt、Excel等文件时，需要造一系列测试数据，写入文件中。
- 对保存在txt、Excel等文件中的用例执行结果进行验证。需要验证的数据很多时，往往是对其中的一部分进行验证。
- 将测试结果保存到Excel中，或绘制成图表。数据量很大时，要花费大量的时间处理数据，如果这个任务需要频繁的去执行，效率可想而知。

针对以上情况，为提高工作效率，最好使用工具来完成上面的任务。该工具应该可以：

- 读、写txt、Excel等文件。
- 按照一定规则生成测试数据，写入文件。
- 从文件中读取用例执行结果，与预期结果进行比较。
- 将测试结果按照一定格式或绘制成图表写入文件中。

其中，读、写txt、Excel等文件是最基本的，本文就是要讲述如何使用Perl语言及Win32::OLE模块来实现读、写Excel（至于txt等其他文本文件的解析，可以使用Perl+正则表达式来实现，不在本文中讲述）。

准备

安装Perl

在<http://www.perl.com>下载perl的最新版本。本例中使用Perl v5.10.0。
操作系统为Windows Vista。

安装Perl 模块

在<http://search.cpan.org/>上搜索并下载以下模块（下载最新版本）：

- Win32::OLE

➤ 手动安装步骤：

- 安装VC。
- 下载模块均为.tar.gz包，需要使用7-zip解压。
- 解压后，进入到目录。
- \$ Perl Makefile.PL
- \$ nmake
- \$ nmake test
- \$ nmake install

➤ 网络安装:

- `$ ppm install Win32::OLE 或 perl -MCPAN -e "install Win32::OLE"`

安装完成后, 可以输入`perldoc Win32::OLE`查看使用说明。

实现

OLE (Object Linking and Embedding) 是一种基于COM的技术。OLE允许应用程序使用其他应用程序提供的通用接口来访问其组件和功能。而在Perl里使用Win32::OLE模块, 来实现对OLE对象的支持。

实现Automation接口的应用程序 (或者DLL动态链接库) 被称作 Automation Server。而创建并使用Automation接口的应用程序被称做 Automation Controller 或者 Automation Client。许多应用程序, 比如 Microsoft Word, Excel, Access, 以及其他 Office 应用程序等等, 都已经实现了 Automation 接口。Automation 被广泛使用在 ASP, CGI 脚本中, 通过 ADO 技术来存取某个 Data Respository。

为了创建一个 Automation 对象, Automation Server 需要在系统中注册一下自己。一般来说, 一个应用程序在安装时, 已经帮你做了这一步, 当然啦, 即便没有, 那么你也可以使用 `regsvr32.exe` 这样的工具来手动注册你的应用。这将在系统的注册表中插入一个条目, 来告诉COM如何找到这个组件, 它提供了什么类型的接口, 它是什么类型的 Automation Server。你能在一个应用程序的文档中找到有关它的 Automation 对象, 以及相关信息。使用Perl语法同样能够创建这样的对象并访问它的组件。

3.1.1 使用Win32::OLE

➤ 创建一个OLE Server对象

首先, 我们要创建一个可用来表示OLE Server的Perl对象。它的意思是: 如果我们想控制其他应用程序的OLE对象, 比如说Excel的, 我们必须创建一个Perl对象用来表示Excel。所以, 尽管我们的程序只是个OLE控制器, 它应该包含用来描述OLE Server的对象。

你可以使用**Win32::OLE->new**创建一个新的OLE Server对象。它接收程序ID (类似“Speech.VoiceText”) 参数, 并且返回一个Server对象:

```
my $server = Win32::OLE->new('Excel.Application', 'Quit');
```

当你的程序不再使用这些Server对象时, 有一些Server对象 (特别是那些Microsoft Office应用程序) 不会自动终止, 它们需要一些Quit方法, 这就是上面我们使用第二个参数的原因。

你也可以直接将你的程序挂接到已经在运行的OLE Server上。

```
my $server = Win32::OLE->GetActiveObject('Excel.Application');
```

如果该Server不存在或因为一些原因拒绝连接，上面语句会执行失败（返回undef）。也可以使用固定的对象名（通常是文件名）来启动合适的Server并且把该对象加载到内存中。

```
my $server = Win32::OLE->GetObject('MyDocument.Doc');
```

➤ 调用方法

一旦创建了一个Server对象，你需要调用它的方法使它工作。调用OLE的方法就像普通的Perl对象方法一样：

```
$server->Foo(@arguments);
```

这是Perl方法的调用----但是它也会触发OLE对象的方法调用。当程序执行这个语句时，\$server对象会执行它的Foo()方法。在应用程序的对象模型文档中找到可以使用的方法。

➤ 参数

默认情况下，所有的参数是按位置的（e.g. foo(\$first, \$second, \$third)），而不是按名称的（e.g. foo(-name)=>"Yogi", foo(-title)=>"Coach"）。必须输入的参数排在前面，接着是可选参数；如果你想让可选参数为假，使用undef。

如果一个方法接收很多位置参数，这会变得很麻烦。可以使用名称参数来代替。

➤ 属性

一个对象的属性可以使用如下方式存取：

```
$server->{Bar} = $value;  
$value = $server->{Bar};
```

这个例子是设置和查询Bar属性的值。也可以使用对象的Bar()方法来达到同样的效果：

```
$value = $server->Bar;
```

然而你不能将第一行写为\$server->Bar = \$value，因为你不能改变一个方法的返回值。

3.2.1 九九乘法表

将九九乘法表写在Excel中。

```
#!/ perl -w

use strict;
use Win32::OLE;
use Win32::OLE::Const 'Microsoft Excel';

$Win32::OLE::Warn = 3;                                # die on erros . . .

my $excel = Win32::OLE->GetActiveObject('Excel.Application') ||
    Win32::OLE->new('Excel.Application', 'Quit');      # get already active # excel
                                                         application
                                                         # or open new

my $book = $excel->Workbooks->Add;                     # open excel file
my $sheet = $book->Worksheets(1);                     # select worksheet number 1

foreach my $i (1..9) {
    my $j = 1;

    while ($j <= $i) {
        my $result = $j * $i;
        my $cell_content = "$j"." X ".$i"." = ".$result";

        $sheet->Cells($i, $j)->{Value} = $cell_content;    # set value of cells

        $j ++;
    }
}

$book->SaveAs('D:\MyProject\Perl\读写Excel\test.xls');
$book->Close;
```

3.2.2 系统性能分析

在Linux服务器上，记录应用程序app-for-test在9：20---15：35时间段内的CPU、内存使用情况，保存到log中。读取log中的时间、CPU使用率和内存使用率，并写入Excel中，方便绘制图表。

每隔3秒记录一次CPU、内存使用率，并保存到日志。

```
#!/bin/bash
```

```
PID=`ps aux | grep -v grep | grep app-for-test | awk '{print $2}'` # get the pid
```

```
top -b -d3 -p $PID > cpu_mem_status.log # record the %CPU %MEM every 3 seconds
```

日志内容如下:



图片审核中...

图3.2-1 日志内容

图中标红线的是需要提取出的数据，且第一行的时间和后面第一次出现的%CPU、%MEM为一组有效数据。提取这些数据主要是利用正则表达式，这里不再详述。接下来，将分析如何把这些数据写入Excel及绘制图表。

```
#!/ perl -w
```

```
use strict;
```

```
use Win32::OLE qw(in with);
```

```
use Win32::OLE::Const 'Microsoft Excel';
```

```
$Win32::OLE::Warn = 3; # die on erros
```

```
#####78
```

```
# Read from "cpu_mem_status.log" and store the data in hash
```

```
my %data = (rows => 6,
```

```
columns => 3,
```

```
1 => ['Time', '%CPU', '%MEM'],
```

```
2 => ['09:24:57', 0.3, 1.0],
```

```

3           => ['09:25:00', 1.2, 2.0],
4           => ['09:25:03', 1.3, 2.0],
5           => ['09:25:06', 1.4, 2.1],
6           => ['09:25:09', 1.5, 2.2],
    );

my $excel = Win32::OLE->GetActiveObject('Excel.Application') ||
    Win32::OLE->new('Excel.Application', 'Quit');

my $book  = $excel->Workbooks->Open('D:\MyProject\Perl\读写Excel\test.xls');
my $sheet = $book->Worksheets(2);
my $range = $sheet->Range('A1:C5');

foreach my $i ( 1..$data{'rows'} ) {
    foreach my $j (1..$data{'columns'}) {
        $sheet->Cells($i, $j)->{'Value'} = $data{$i}[$j-1];
    }
}

my $chart = $excel->Charts->Add;          # Add chart
$chart->{'ChartType'} = xlLineMarkers; # Set chart type
$chart->SetSourceData( {Source => $range,
                       PlotBy  => xlColumns} );
$chart->{'HasTitle'} = 1;
$chart->ChartTitle->{'Text'} = 'Cpu, Memory status'; # Set chart title


$book->Save;
$book->Close;

```

Excel的内容及生成的图表如下:

图片审核中...

图3.2-2 Excel内容



图片审核中...

图3.2-3 图表样式

附录

参考资料:

<http://baike.baidu.com/view/118545.htm>

<http://dev.csdn.net/article/45/45678.shtm>

<http://www.xav.com/perl/faq/Windows/ActivePerl-Winfaq12.html>

http://www.foo.be/docs/tpj/issues/vol3_2/tpj0302-0008.html

[lftlying](#) □□□Friday, March 06, 2009 21:57:49 [□□](#) [□□](#)



□□□□□2□□□□□□□□□□□□□□□□VC□

```
1.perl -MCPAN -e "install Win32::OLE"
```

2. my_excel.pl my_excel.pl

□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

```
E:\StudyData\Perl>my_excel.pl
```

Win32::OLE(0.1709) error 0x800401f3: "" at E:\StudyData\Perl\my_excel.pl line 8

```
eval {...} called at E:\StudyData\Perl\my_excel.pl line 8
```

[will2ni](#) □□□Saturday, March 07, 2009 19:12:33 [□□](#) [□□](#)



Itflying□□□□□□□□□□□□□□

my_excel.pl8

[lrflying](#) □□□Saturday, March 07, 2009 19:43:37 [□□](#) [□□](#)



□8□□□□□□

```
my $excel = Win32::OLE->GetActiveObject('Excel.Application') || Win32::OLE->new('Excel.Application', '  
# get already active # excel application
```

[will2ni](#) □□□Saturday, March 07, 2009 19:50:08 [□□](#) [□□](#)



□□□□□□□□□□□□□□□□□□Excel□□□

□□□□□Office 2003(Excel)□□

[lrflying](#) □□□Saturday, March 07, 2009 20:00:58 [□□](#) [□□](#)

☐Office 2003(Excel)☐

□□□-->□□Microsoft Office Excel(A)□□□□□□□□□

Microsoft Excel(11.8169.8172) SP3

[will2ni](#) □□□Saturday, March 07, 2009 20:09:01 [□□](#) [□□](#)

[illegible]

use diagnostics;

[illegible]





Itflyng ☐☐☐ Saturday, March 07, 2009 20:15:33 ☐☐ ☐☐



Uncaught exception from user code:





Win32::OLE(0.1709) error 0x800401f3: "" at E:\StudyData\Perl\my_excel.pl

```
line 9
eval {...} called at E:\StudyData\Perl\my_excel.pl line 9
at E:\StudyData\Perl\my_excel.pl line 9
```

[will2ni](#)   Saturday, March 07, 2009 21:16:03  



google
Perl Python

[songdiandong](#)   Thursday, July 23, 2009 00:12:41  



array Formula ?

发表评论 [“评论王争夺赛”第3期活动开始啦!](#)

表情:



评论内容:

用户名: 匿名用户

[登录](#) [注册](#)

验证码:  [重新获得验证码](#)

发表评论