

用Perl来创建可重用的Web应用程序 PDF转换可能丢失图片或格式，建议阅读原文

https://www.100test.com/kao_ti2020/167/2021_2022__E7_94_A8Perl_E6_9D_A5_c103_167232.htm Perl 是面向复杂 Web 应用程序开发的方便而有效的工具。然而，即使是经验丰富的程序员也会因为 Perl 的学习和使用难度太高而不愿使用。本文阐述了 Perl 面向对象的执行能简化工作，特别是因为其分布式设计及应用程序的功能性能使其比其他 Web 技术更为有效 当我尝试寻找一个好的留言簿脚本时发现，在 CGI 脚本中，面向对象的执行并不常见。我想要一个能通过改变式样、添加新功能等方式能进行轻易的修改，并能用来建立论坛的脚本。在 Web 上三十种免费留言簿中，没有一种符合我的需要。因此我转向了对象技术，作为面向基于 HTML 模板的、可重用的 Web 应用程序的解决方案。对象和类的关系 在设计对象模型前，最好检查一下现有的可能有用的 CPAN

(Comprehensive Perl Archive Network) 模块。主要问题是新创建的类与 CPAN 模块类之间的关系。这种关系可能会是：a) 一个标准类对象包含在新创建的类中（“包含”关系），或者不太常见的 b) 新创建的类从标准类继承而来（“等价”关系）。下面列出的代码是一个面向 My 类的构造程序示例，它从 BaseClass 继承而来。此外，My 类包括 AnotherClass 对象，该对象是私有的（其名称以下划线字符开始 -- 这是个并非由 Perl 本身实施的传统）。清单 1. My 类：执行关系

```
package My;
require BaseClass;
#required if BaseClass is present in
BaseClass.pm
@ISA=qw(BaseClass);
use AnotherClass;
sub new {
    my $package=shift;
    my $self=$package->SUPER::new($package);
```

#create object in BaseClass \$self->{_another_class_object}=new AnotherClass. \$self.} 执行特定的 CGI 类 Lincoln Stein 的 CGI 模块非常适合处理脚本参数，但不适合用来生成 HTML 元素（当 HTML 嵌在脚本内部时）。有意思的是，CGI 模块既能以面向函数的方式使用，也能以面向对象的方式使用，并保持了被继承的能力。例如，CGI 模块能被 CGI::Apache 和 CGI::Fast 模块所继承，它们分别是 Perl-Apache API 和 Open Market FastCGI 标准的 CGI 接口。大多数 Web 服务器保留有它们自己的 API，该 API 执行 CGI API，并以特定的功能对其予以增强。Apache::ASP 模块（移植到 Apache Web 服务器上的 Active Server Pages）使用了五个面向 CGI 和会话管理的对象，这种管理使 CGI 脚本的编写更加容易。对于留言簿脚本设计来说，可以使用 API 来创建一个抽象类。一个抽象类至少需要一个纯虚拟函数。在一般情况下，它可能包括 new() 构造程序函数和 get_param() 方法（没有为之定义执行程序）。它应该包括使用 CGI 所需要的所有函数：get_cookie(), set_cookie(), get_script_url(), 和甚至更为高级的函数，如 is_print_form_mode() 及 is_print_entries_mode()。该抽象类将被特定的 API 实现所继承。因此，当类被移植到另一个 API 上时，应该创建一个继承抽象类的新类。Perl 不直接支持抽象类。为此，我们需要用一些简单的技巧：禁止类对象的创建（让我们把类命名为 AbstractCGI，请参阅以下清单），并定义所有的纯虚拟函数，这样在每次函数调用时就会发生错误。在这种情况下，要从抽象类中创建一个新的类，就应只继承 init() 和 get_param() 方法。新类已经含有了其它（独立于 API 的）方法。清单 2. AbstractCGI 类：实现特定的 API

```
(CGI) package AbstractCGI.sub new { my $package=shift. my
$self={ _cgi_method=>undef, #simple class data _query=>undef }.
bless $self, ref $package || $package. $self->_init. $self.}sub _init
{die} #private methodsub get_param {die}sub is_print_form_mode
{ shift->get_param(mode) eq form }sub is_print_entries_mode {
shift->get_param(mode) eq entries }
```

使用 HTML 模板 让我们来看看用 HTML 模板来创建类。留言簿设计式样能存储在一个由脚本处理的普通 HTML 文件中，这样 HTML 编码人员可以对 CGI 一无所知，并能独立于 CGI 开发人员来设计 HTML 页。需要两个 HTML 文件：form.html（“添加输入项”页）以及 view.html（“查看留言簿”页）。在设计新类之前，让我们来看一下 CPAN。HTML::Template 模块允许在 CGI 脚本中使用 HTML 模板。Sam Tregar（模块的创建者）利用有意思的功能创建了一种有效的工具。模板本身是一个普通 HTML 文件，它含有由脚本处理的额外标签。当页面传递到客户端时，标签被脚本中定义的值所替代。此外，模块部分能根据脚本的值而被排除、包含或重复。这些关于模板的设想对于编写留言簿脚本来说特别有帮助。对 HTML::Template 的改进同时，该模块也有弱点。有两种类型的模板标签：和 执行依次打开及关闭一个额外标签（两者均可能为空）会变得更加有效。一些句法类型也能起到帮助作用。一部分 HTML 设计人员更喜欢用在 PHP 中的 标签，或用在 ASP 中的 标签。另外，_TMPL_TAG 能在 标签而不是 中被有效使用。可能还需要其它改进。当然，一个 HTML::Template 的新的和特定的派生类可能会有用。这就需要您复制并编辑巨大的 _parse() 方法，该方法是模块的核心。另一个更为灵活的解决方案是在

未来能得以简便、快速地升级的对象技术基础上创建一个新的模块。遗憾的是您不能在 HTML::Template 模块中实现这一步。用 CGI::Application 创建可重用的 Web 应用程序

CGI::Application 模块是专为创建可重用的 Web 应用程序而设计的，它使用了对象技术，并包括了 CGI 和 HTML::Template 模块。这一模块的观念简单而有效：应用程序驱动 Web 页，但反过来则不成立（就像 PHP、ASP、JSP 和 EmbPerl 等等）。

Web 应用程序代码能保存在单个文件（模块）中。此外，该文件能包含完整的控制代码（客户端与 Web 页以及数据库和操作系统界面之间的交互作用）。CGI::Application 模块对于大多数 Web 应用程序（包括留言簿）来说都是一流的解决方案。使用该模块能轻易地创建一个新的和特定的派生类，它带有新的方法和经过改进的基础类。还可以有一个继承下来的构造程序（有着留言簿专用的新参数）和一个继承下来的 load_tmpl() 方法（举例来说，若用特定的模板模块来代替 HTML::Template）。结论这种由受着单个 Perl 模块控制的 Web 应用程序来驱动 Web 页（模板）的设想对交互式 Web 应用程序特别有效。模板同样能驱动应用程序（在

CGI::Application 和 HTML::Template 模块中就不存在该功能）。

留言簿说明了这种执行：模块形式包括了“名称”、“电子邮件”和“消息正文”等字段。由于不必在脚本中进行直接的字段定义，这些字段能由脚本轻易地从模板中获得。因此，我们就有了一个有重用性的应用程序：要修改应用程序，只需编辑模板就行了。100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com