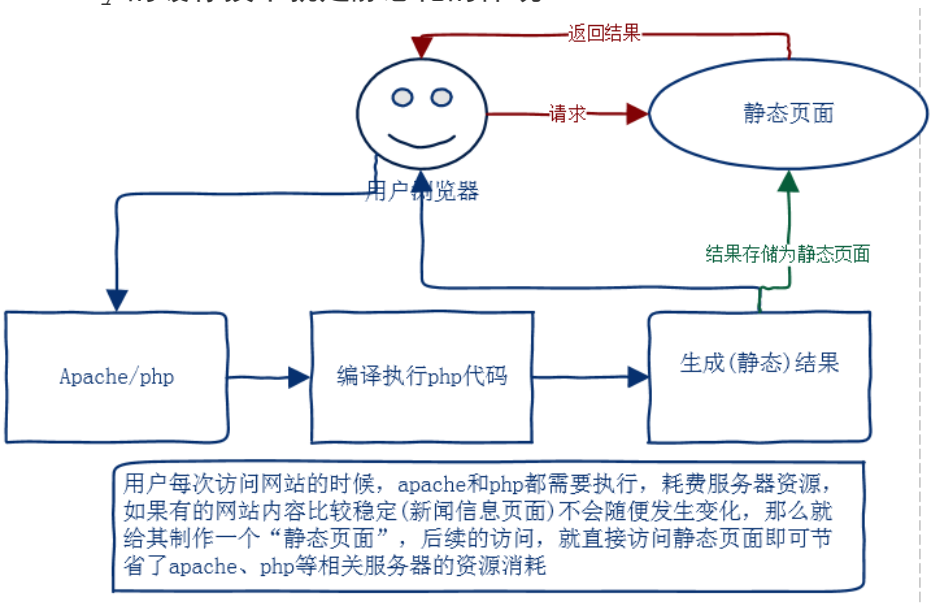
1 纯静态化

Smarty 的缓存技术就是静态化的体现。



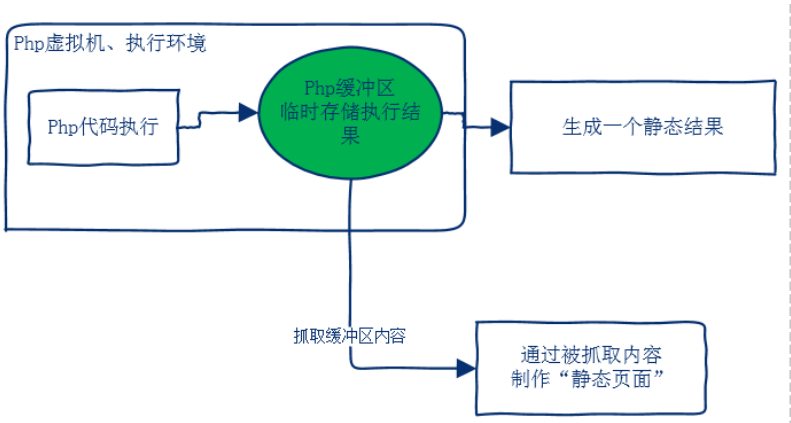
1.1 什么是纯静态化

把 php 执行、生成好的内容制作为一个“静态页面”，该制作过程就是静态化。

1.2 为什么使用静态化

节省 php、mysql 等服务器资源
节省用户等待时间，访问速度快
搜索引擎(百度)更喜欢收录“静态页面”

1.3 实现静态化



php 代码执行---->缓冲区---->被抓取----->生成静态页面
一个简答的 php 代码：

```php
<?php

//实现一个简单静态化过程
for($i=0; $i<10; $i++){
    echo $i."<hr />";
}
```

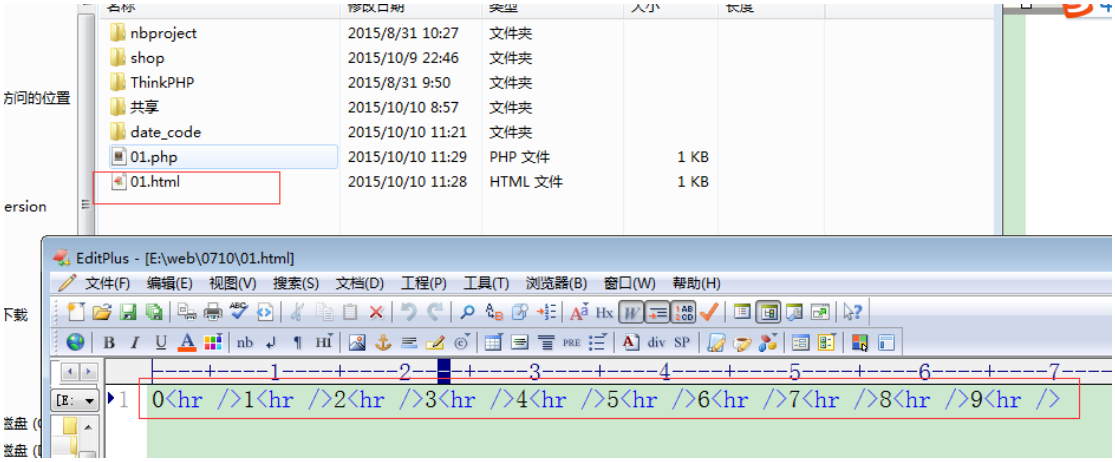
以上 php 代码对应的静态内容：

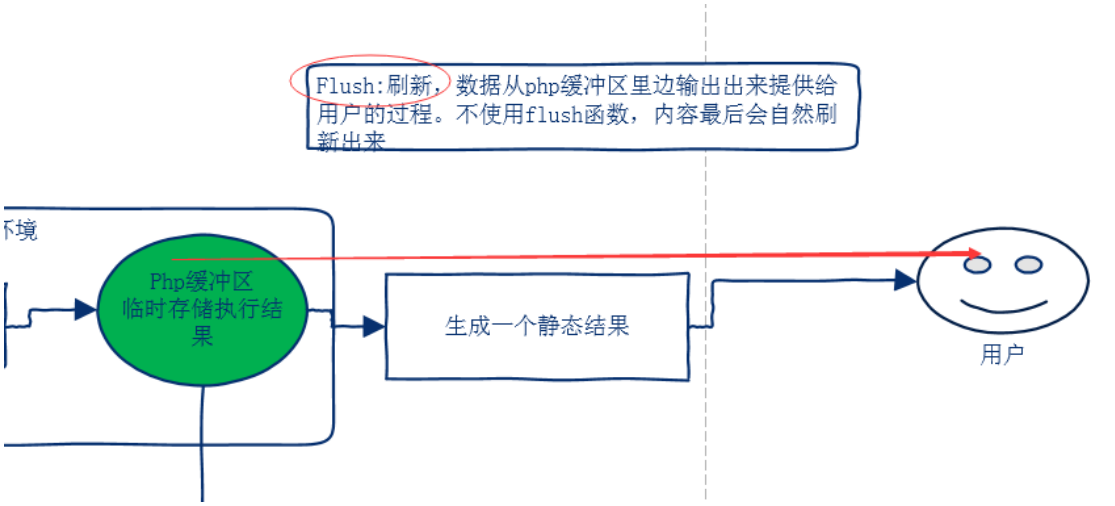
现在利用 php 把对应的静态内容从“php 缓冲区”里边给抓取出来，并制作静态页面：

```php
<?php

//① 开启php缓冲区(ob:output_buffering)
ob_start();

//实现一个简单静态化过程
for($i=0; $i<10; $i++){
    echo $i."<hr />";
}

//② 抓取php缓冲区的内容
//    我们在浏览器上看到的静态内容都会被该“php缓冲区”收集
$cont = ob_get_contents();

//③ 利用抓取到的内容制作静态页面(文件)
file_put_contents('./01.html',$cont);

//④ "删除"缓冲区内容  并"关闭"
ob_end_clean();

```

以上程序代码在执行的时候，浏览器页面就没有输出内容，因为 ob_end_clean 已经把内容从缓冲区删除了。(如果没有 ob_end_clean 浏览器会显示吗？--会的，调用 ob_end_flush)

生成的静态页面及对应的内容：

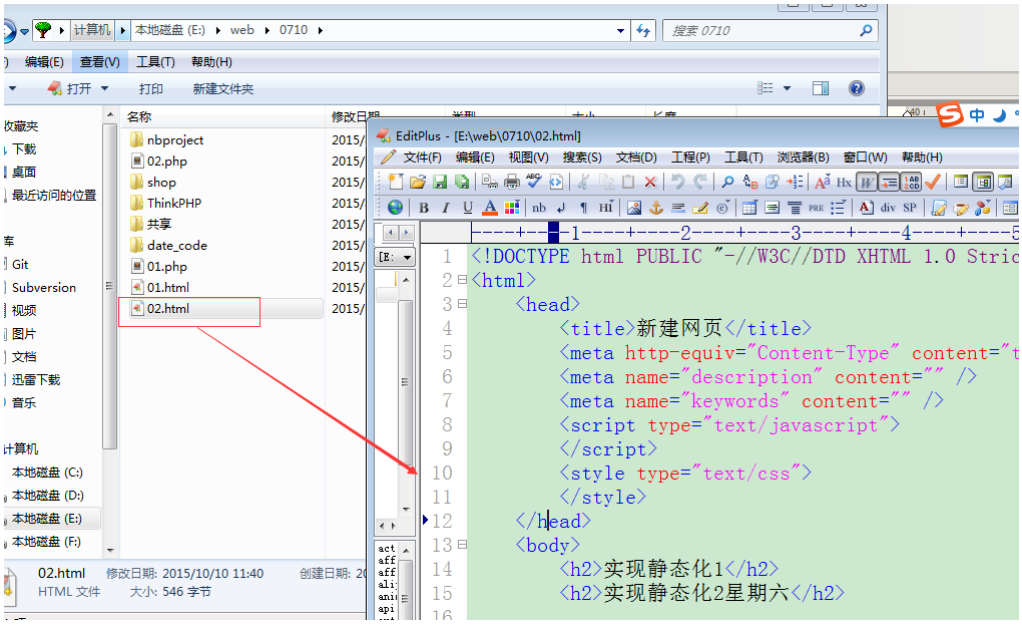





```php
<?php
ob_start();
$week = "星期六";
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/D
<html>
    <head>
        <title>新建网页</title>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
        <meta name="description" content="" />
        <meta name="keywords" content="" />
        <script type="text/javascript">
        </script>
        <style type="text/css">
        </style>
    </head>
    <body>
        <h2>实现静态化1</h2>
        <h2>实现静态化2<?php echo $week; ?></h2>
<?php
    //ob_get_contents()给收集当前行往上的输出内容(不给收集后续内容)
    $cont = ob_get_contents();
    file_put_contents('./02.html',$cont);
    ob_end_clean(); //"删除并关闭"缓冲区
?>
```

在静态化文件里边体现的是静态内容，而非php代码

静态化文件只给体现 ob_get_contents 之前收集的内容，之后输出 的内容不给体现：



1.4 相关函数

开启缓冲

```
ob_start()
```

php.ini　　output_buffering=4096;

```
262
263 output_buffering = 4096
264 ;output_buffering = Off
265
```

获取内容：

```
ob_get_contents();  //获取
ob_get_clean();     //获取后清空
ob_get_flush();     //获取刷新
```

清空

```
ob_clean()      //删除缓冲区内容
ob_get_clean(); //获取并删除缓冲区内容
ob_end_clean(); //清空并关闭缓冲区
```

刷新（缓冲区内容自然做输出，给用户的浏览器显示）

```
ob_flush()          //数据向下推送
ob_get_flush();     //获取内容并推送内容
ob_end_flush();     //推送内容并关闭缓冲区
```

（没有调用 flush 时，每个 php 脚本结束后会自动 flush，把内容呈现给用户）

关闭

```
ob_end_clean();     //清空关闭
ob_end_flush();     //刷新关闭
```

1.5 tp 项目对静态化的应用

在项目后台添加商品的时候　就给商品的详情生成静态页面
前台就直接访问商品的静态详情页面。
在后台展现前台模板页面：

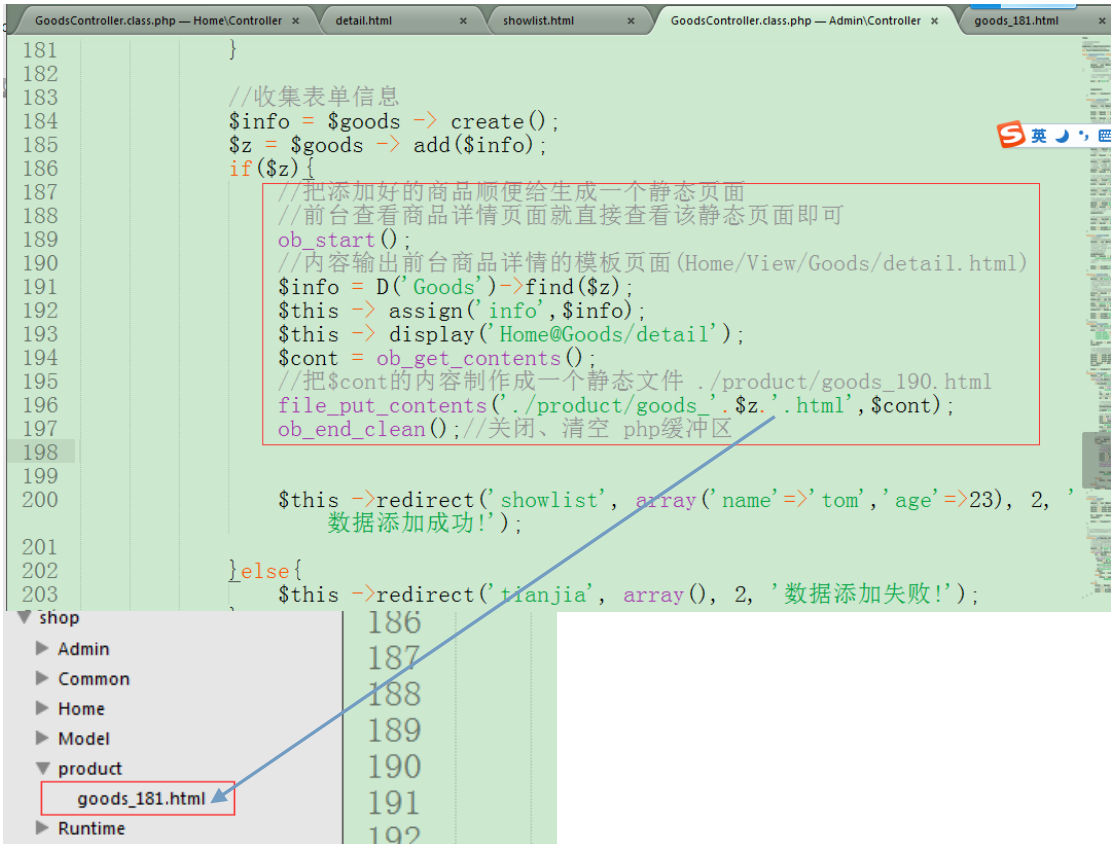
display('[模板文件]'['字符编码']['输出类型'])

模板文件的写法支持下面几种：

用法	描述
不带任何参数	自动定位当前操作的模板文件
[模块@][控制器:][操作]	常用写法，支持跨模块 模板主题可以和theme方法配合
完整的模板文件名	直接使用完整的模板文件名（包括模板后缀）

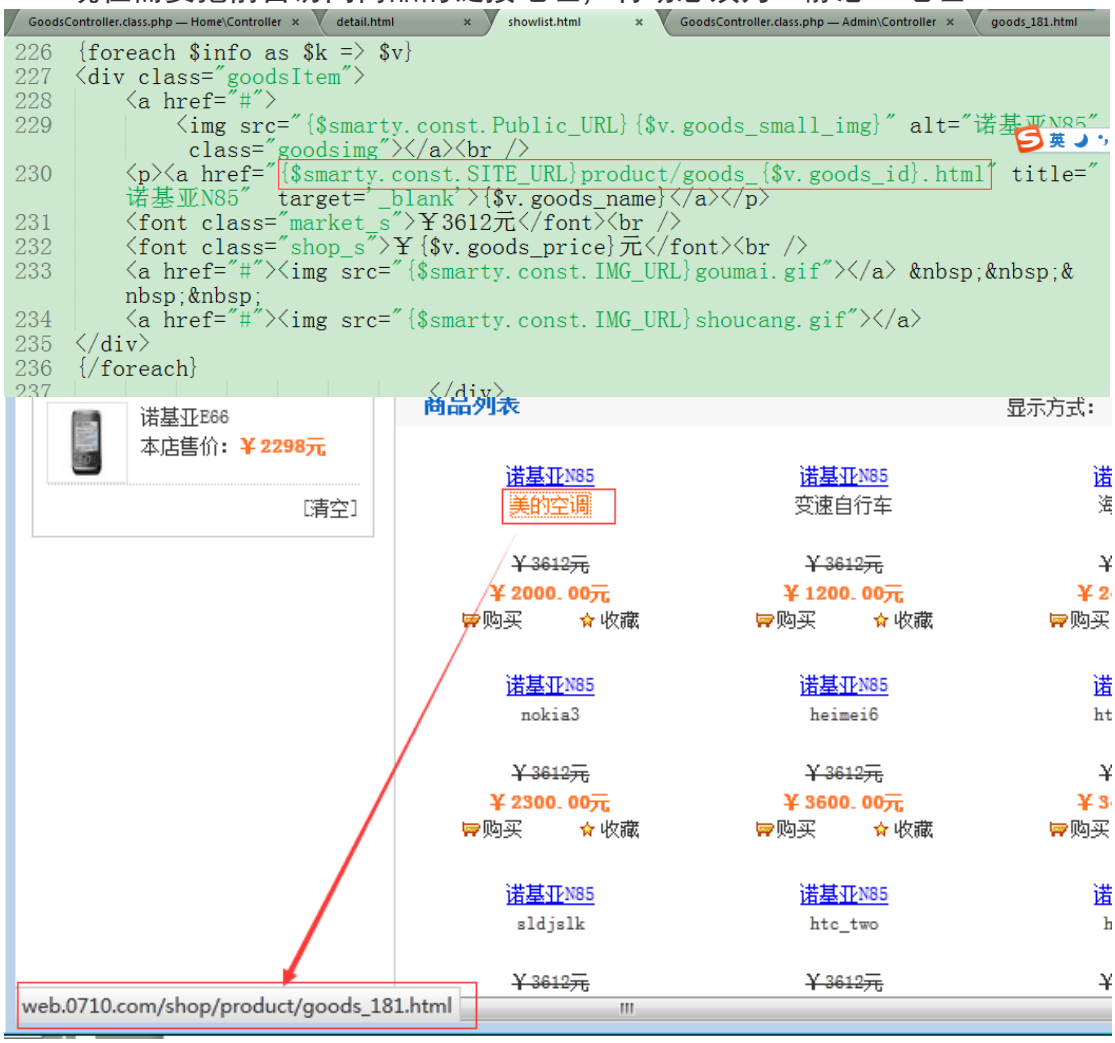
下面是一个最典型的用法，不带任何参数：

在项目后台添加商品成功后就顺便生成该商品对应的静态文件：



此时前台商品的链接地址还是“动态地址”：

现在需要把前台访问商品的链接地址，有动态改为“静态”地址：



前台访问商品信息，直接访问一个静态页面即可：

如果后期商品数据有修改，就根据修改后的信息重新生成静态页面就可以了。

利用一个私有方法，实现静态页面制作，这样各种操作(添加/修改)直接调用该 makehtml 方法即可，非常方便：

GoodsController.class.php — Home\Controller | detail.html | showlist.html | GoodsController.class.php — Admin\Controller | goods_181.html

```
187
188                 $this -> makehtml($z);
189
190                 $this ->redirect('showlist', array('name'=>'tom','age'=>23), 2, '
                        数据添加成功!');
191
192             }else{
193                 $this ->redirect('tianjia', array(), 2, '数据添加失败!');
194             }
195         }else{
196             //展示表单
197             $this -> display();
198         }
199     }
200     //给商品生成静态页面
201     private function makehtml($goods_id){
202         //把添加好的商品顺便给生成一个静态页面
203         //前台查看商品详情页面就直接查看该静态页面即可
204         ob_start();
205         //内容输出前台商品详情的模板页面(Home/View/Goods/detail.html)
206         $info = D('Goods')->find($goods_id);
207         $this -> assign('info',$info);
208         $this -> display('Home@Goods/detail');
209         $cont = ob_get_contents();
210         //把$cont的内容制作成一个静态文件 ./product/goods_190.html
211         file_put_contents('./product/goods_'.$goods_id.'.html',$cont);
212         ob_end_clean();//关闭、清空 php缓冲区
213     }
```

1.6 静态页面局部刷新

一个静态页面全部的内容都是固定的，但有的时候局部数据是随时需要变化的。可以利用 ajax 随时感知变化的信息再显示。

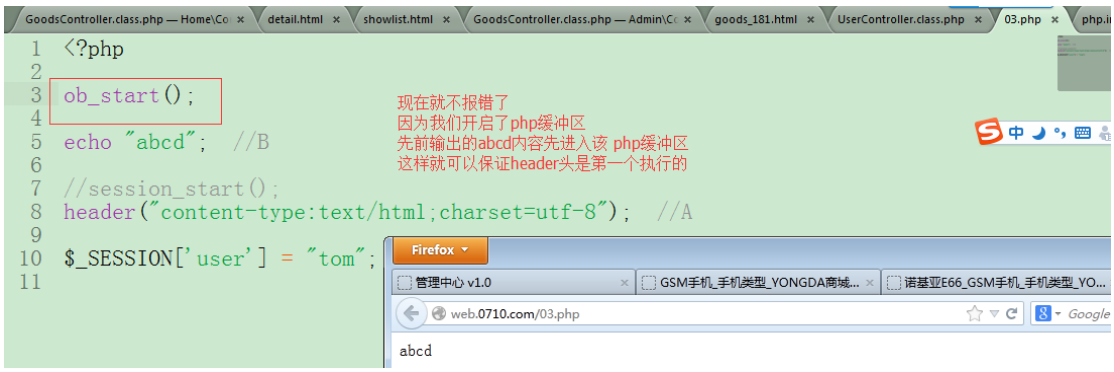
通过 ajax 给静态页面显示变化的信息(注意 ajax 访问的路径)：

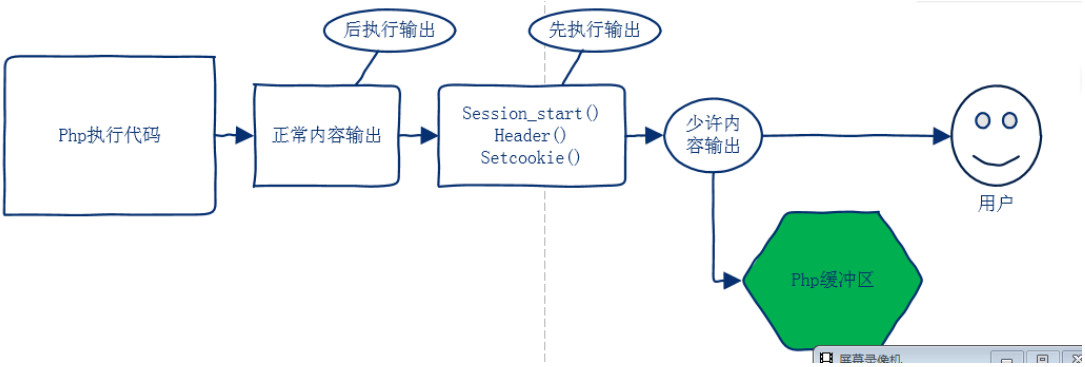
1.7 session_start/header()/setcookie()与 php 缓冲区的关系

缓存：缓存是可以看得见的，例如有缓存文件，数据较持久

缓冲：是一个临时存储区域，其数据都是运行在内存中，数据容易消失

session_start、header()函数、setcookie()设置 cookie 等语句在使用的时候前边不能有输出，否则系统要报错。





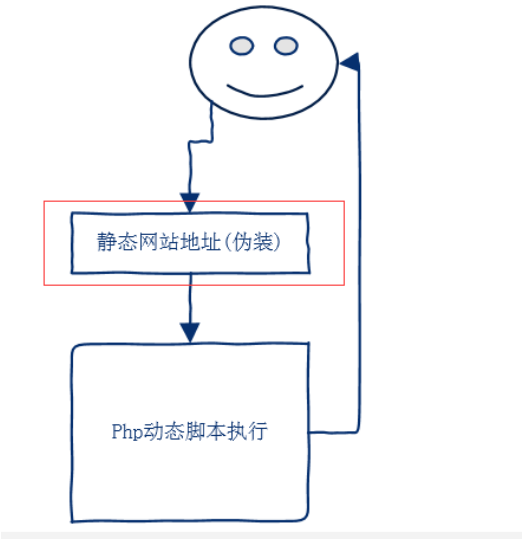
2 伪静态

我们在网站输入的 url 地址是静态的地址，但是内部走是的“动态”的程序。

伪静态就是一个伪装效果。

http://网址/product/goods.php

http://网址/product/goods.html



伪静态好处：

① 对搜索引擎(seo)的收录有好处

② 用户使用体验非常好

2.1 伪静态配置

apache(iis tomcat nginx)可以配置使用伪静态

修改 httpd.conf,开启伪静态的重写模块支持：

```
116 #LoadModule proxy_ftp_module modules/mod_proxy_ftp.so
117 #LoadModule proxy_http_module modules/mod_proxy_http.so
118 #LoadModule proxy_scgi_module modules/mod_proxy_scgi.so
119 #LoadModule reqtimeout_module modules/mod_reqtimeout.so
120 LoadModule rewrite_module modules/mod_rewrite.so
121 LoadModule setenvif_module modules/mod_setenvif.so
122 #LoadModule speling_module modules/mod_speling.so
123 #LoadModule ssl_module modules/mod_ssl.so
124 #LoadModule status_module modules/mod_status.so
```

虚拟主机配置，添加 AllowOverride All 项目：

```
443         Allow from all
444     </Directory>
445 </VirtualHost>
446
447 <VirtualHost *:80>
448     ServerName web.0710.com
449     DocumentRoot "E:/web/0710/"
450     <Directory "E:/web/0710/">
451         Options Indexes FollowSymLinks
452         AllowOverride All
453         Order Deny,Allow
454         Allow from all
455     </Directory>
456 </VirtualHost>
457
```

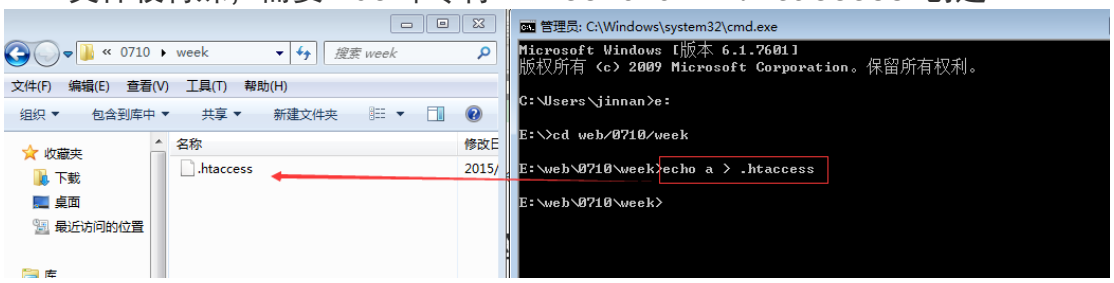
允许在主机中的任何目录设置伪静态效果

之后重启 apache

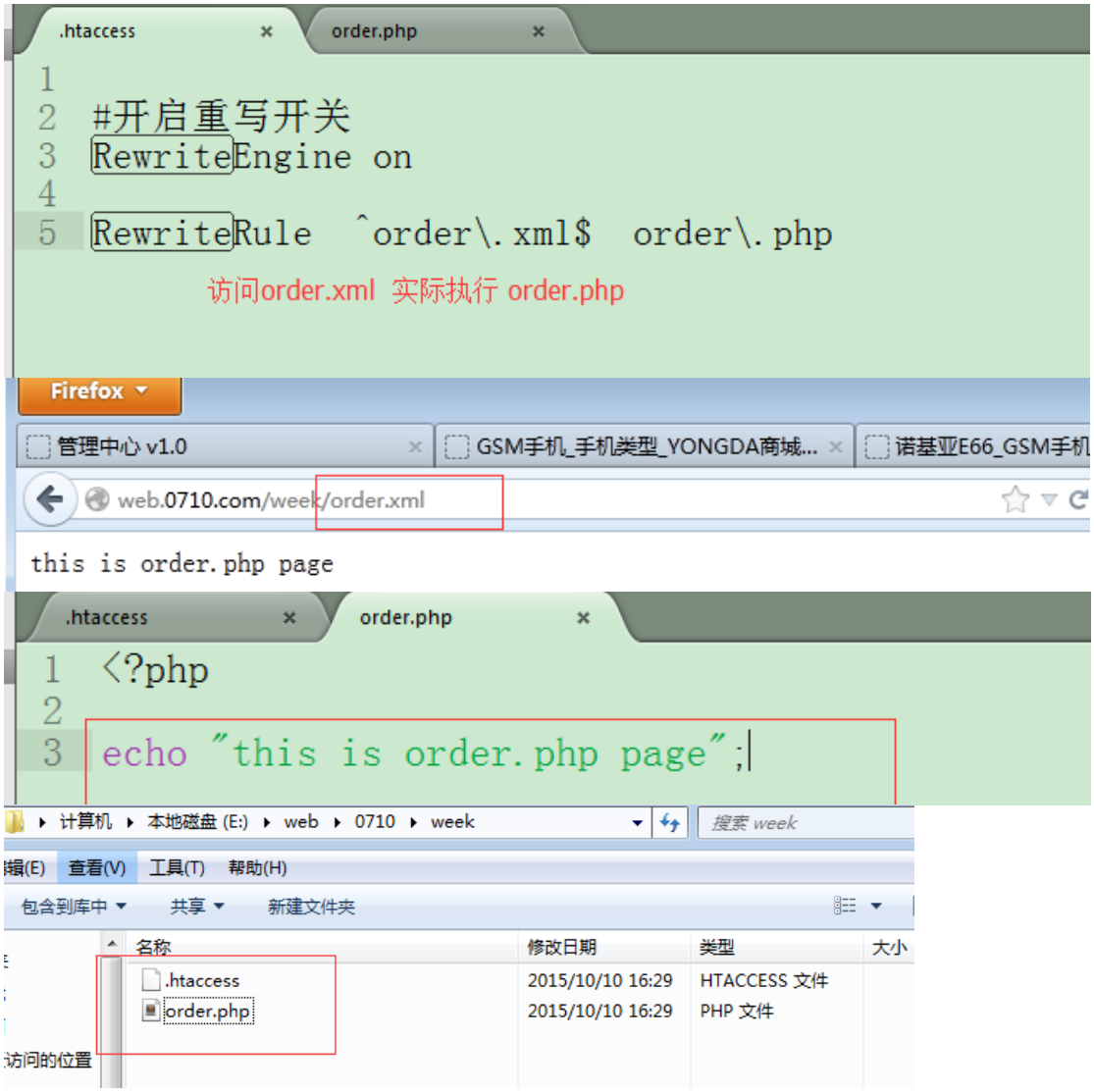
2.2 伪静态简单使用

访问：http://web.0710.com/week/order.xml
真实指向地址： http://web.0710.com/week/order.php

在被操作的“目录”下创建一个伪静态规则文件 “.htaccess”
文件较特殊，需要 DOS 命令行： >echo a > .htaccess 创建



在.htaccess 内部编辑伪静态规则：



2.3 带参数指向

访问：http://web.0710.com/week/cat_567.xml
真实指向地址： http://web.0710.com/week/cat.php?id=567

```
#cat_567.html  ==========>  cat.php?id=567
#() 是正则模式单元内容
# $1 表示重复使用第一个括号内容
RewriteRule  ^cat_(\d+)\.html$   cat\.php?id=$1
```

访问：http://web.0710.com/week/dog_567_wangcai_beijing.xml
真实指向地址：
http://web.0710.com/week/dog.php?id=567&name=wangcai&addr=beijing

```
#dog.html====>dog.php
#dog_567_wangcai_beijing.xml
RewriteRule ^dog_(\d+)_([a-z]+)_([a-z]+)\.xml$  dog\.php?id=$1&name=$2&addr=$3
```

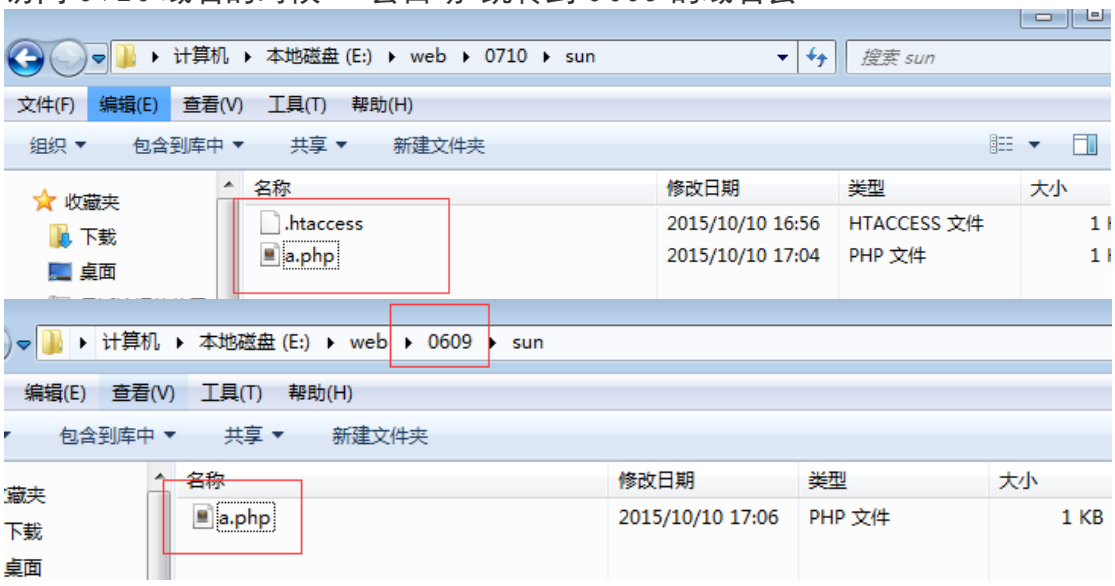
ecshop 提供的伪静态例子参考：

```
RewriteEngine On
#RewriteBase /

# direct one-word access
RewriteRule ^index\.html$    index\.php [L]
RewriteRule ^category$      index\.php [L]

# access any object by its numeric identifier
RewriteRule ^feed-c([0-9]+)\.xml$       feed\.php\?cat=$1 [L]
RewriteRule ^feed-b([0-9]+)\.xml$       feed\.php\?brand=$1 [L]
RewriteRule ^feed-type([^-]+)\.xml$       feed\.php\?type=$1 [L]
RewriteRule ^feed\.xml$                 feed\.php [L]

RewriteRule ^category-([0-9]+)-b([0-9]+)-min([0-9]+)-max([0-9]+)-attr([^-]*)-([0-9
]+)-(.+)-([a-zA-Z]+)(.*)\.html$  category\.php\?id=$1&brand=$2&price_min=$3&price_
max=$4&filter_attr=$5&page=$6&sort=$7&order=$8 [QSA,L]
RewriteRule ^category-([0-9]+)-b([0-9]+)-min([0-9]+)-max([0-9]+)-attr([^-]*)(.*)\.
html$                          category\.
```

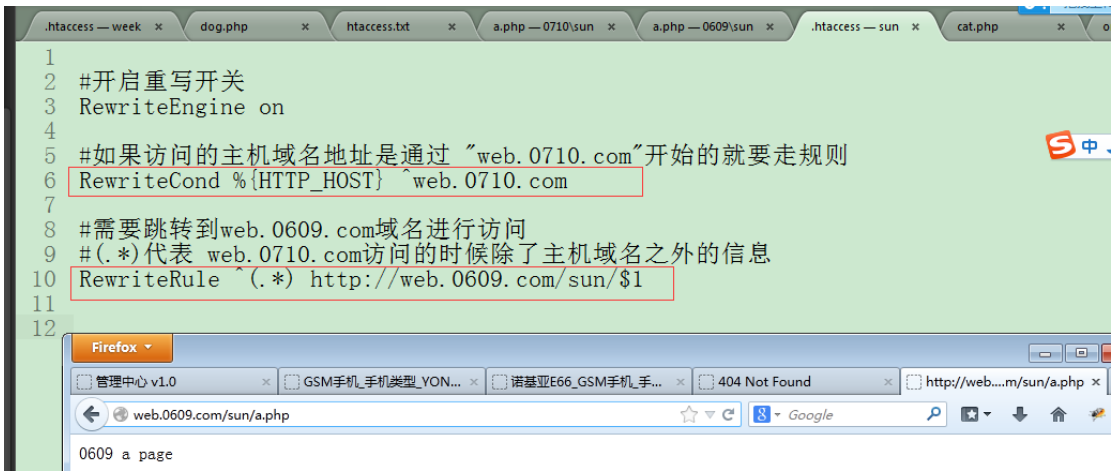
2.4 域名地址跳转

网站做升级，由旧域名升级使用新域名
但是老用户还习惯使用旧域名
我们要自动帮其跳转到新域名去

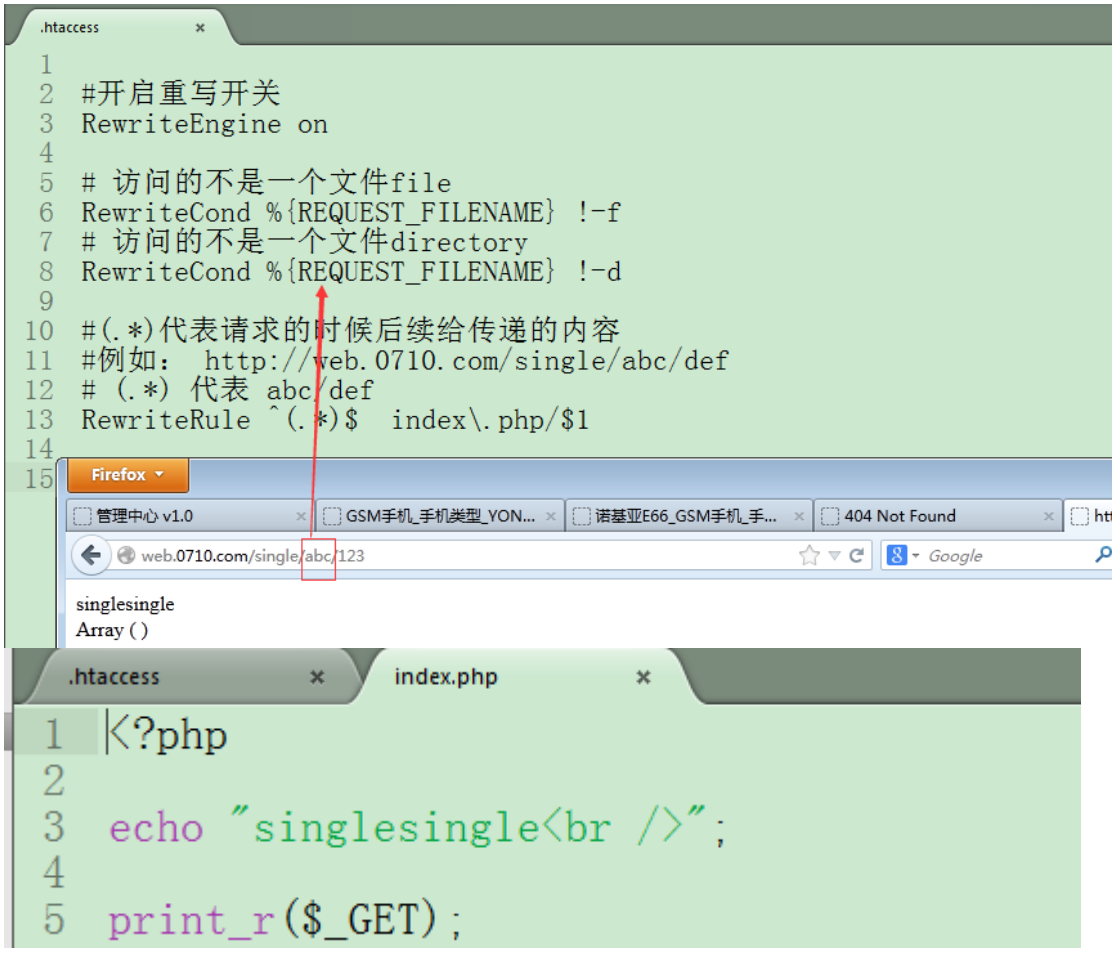
访问：http://web.0710.com/sun/a.php
真实指向地址：http://web.0609.com/sun/a.php
访问 0710 域名的时候　会自动 跳转到 0609 的域名去



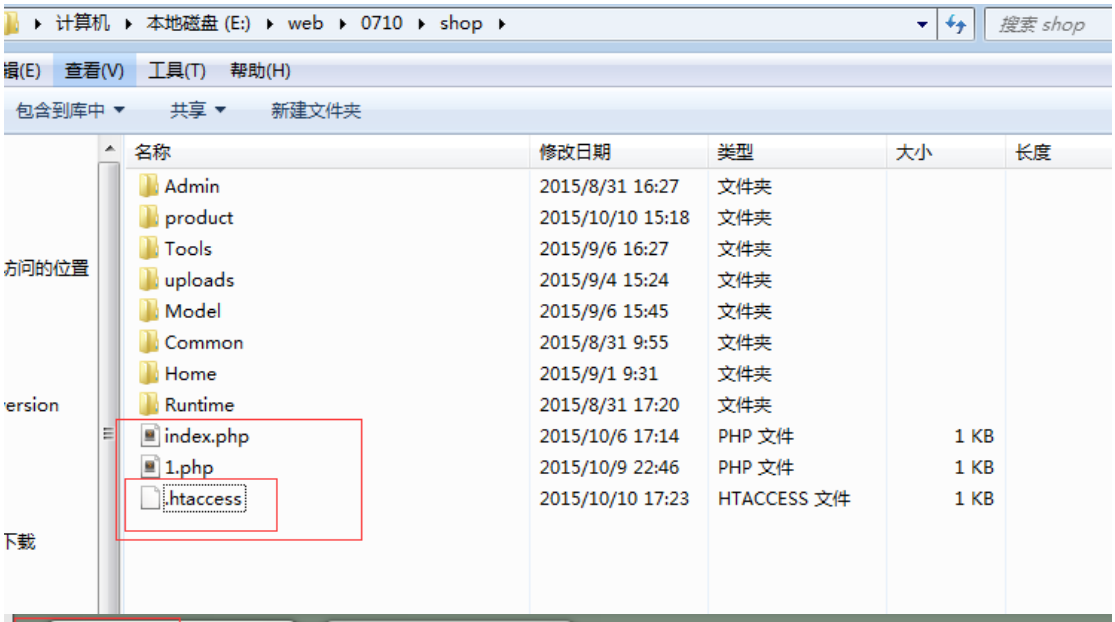
制作规则：



2.5 隐藏 index.php 入口文件



2.6 给 tp 框架隐藏 index.php 入口文件



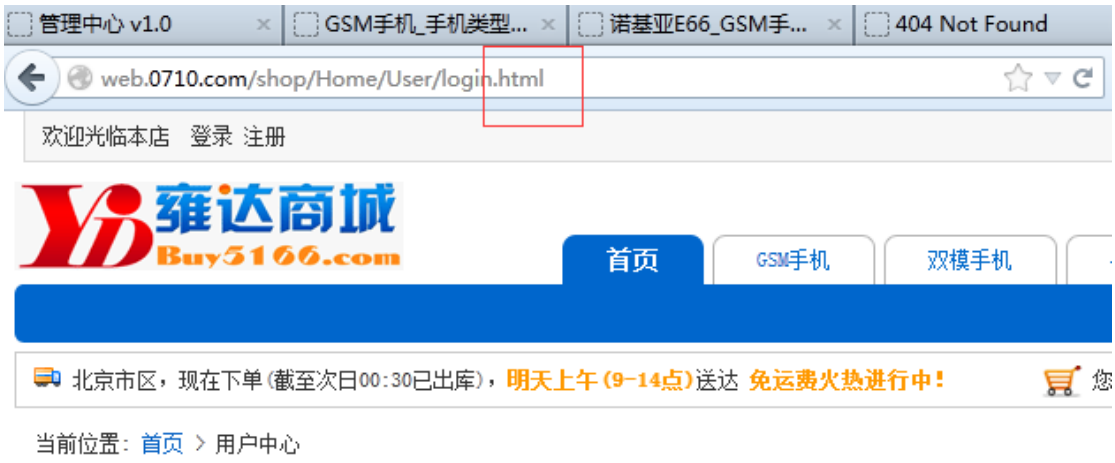
.htaccess

```
#开启重写开关
RewriteEngine on

# 访问的不是一个文件file
RewriteCond %{REQUEST_FILENAME} !-f
# 访问的不是一个文件directory
RewriteCond %{REQUEST_FILENAME} !-d

#(.*)代表请求的时候后续给传递的内容
#例如： http://web.0710.com/single/abc/def
# (.*) 代表 abc/def
RewriteRule ^(.*)$  index\.php/$1
```

2.7 tp 框架的伪静态

php 层面的伪静态

2.7.1 有伪装后缀：



2.7.2 pathinfo 路径模式是伪静态体现

2.7.3 路由设置伪静态

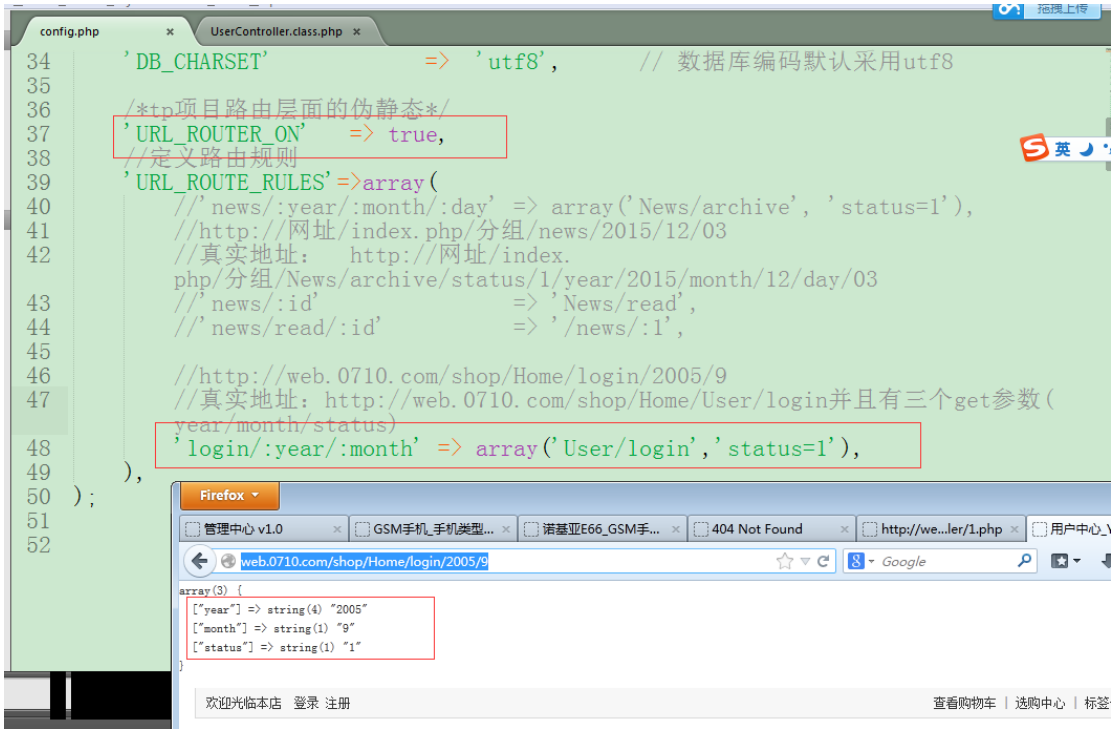
```
// 开启路由
'URL_ROUTER_ON'   => true,
```

给tp项目做伪静态设置

3.2的路由功能是针对模块设置的，所以URL中的模块名不能被路由，路由定义也通常是放在模块配置文件中。

然后就是配置路由规则了，在模块的配置文件中使用**URL_ROUTE_RULES**参数进行配置，配置格式是一个数组，每个元素都

```
'URL_ROUTE_RULES'=>array(
    'news/:year/:month/:day' => array('News/archive', 'status=1'),
    'news/:id'               => 'News/read',
    'news/read/:id'          => '/news/:1',
);
```



总结：

1．纯静态应用

tp 项目引用

在后台添加商品同时给前台生成一个静态页面

可以通过 ajax 随时更新静态页面的局部的变化信息

静态化好处：

节省服务器资源

提高请求速度

便于搜索引擎收录

一般新闻信息、文章信息 适合做纯静态。

2．伪静态

使用的地址是静态的，程序的执行是动态的

apache 层面

在 apache 的 httpd.conf 内部开启 rewrite 模块

虚拟主机开启 Allowoverride all(作用可以使得我们在任何一个 apache 主机文件目录下都可以使用.htaccess 文件)

具体使用：简单使用、带参数使用、隐藏 index.php 入口文件、域名跳转

php 层面(tp 框架、其他框架也有)

作业：

1．把项目的 index.php 入口文件给隐藏起来

2．把项目的不变化的信息(新闻、文章、简介信息)给制作静态页面，在前台访问