模板预定义变量:

$appdir:工程目录

$curl:当前请求的url

$ccontroller:当前请求的controller

$entry:入口文件地址

$webroot:项目根目录

预定义常量:

SMVC\_VERSION:版本

SMVC\_BASEDIR:smvc的SmallMVC.php所在的目录绝对路径

SMVC\_CONFIGDIR:框架配置文件所在路径

SMVC\_COREDIR:框架core路径

SMVC\_PLUGINDIR:

SMVC\_ENTRYSCRIPT: 项目入口文件路径

APP\_INCLUDE\_PATH: 项目路径环境变量串

APPDIR:项目路径,即在入口文件中定义的文件夹,相对于smvc

WEB\_ROOT:入口文件所处的路径

DS: 目录用的分隔符linux为/ ，windows为\

PS: 路径变量所用分隔符，linux为: , windows为;

约定：

所有框架文件，以SmallMVC开头，以.php结尾。

Controller需要继承自SmallMVCController,Model需要继承自SmallMVCModel.

所有类都要以大写开头文件名为类名，Model类或Controll类需要在类名后面加上Model或Controller。

类文件，必须以.php结尾，除结尾的.php中的字符’.’之外，不可以再出现字符’.’，因为该字符被用作include文件时的目录分隔符，如import(‘A.b.c’),其实是包含的A/b/c.php文件。

SmallMVCModel类的构造函数原型为\_\_construct(‘$table\_name\_in\_db’, ‘$pool\_name’),所以如果自己定义的Model有构造函数，且需要通过构造函数接受参数，需要如下定义:YourModel::\_\_construct(‘$table’, ‘$params\_array’){parent::\_\_construct(‘$table’, ‘$pool\_name\_from\_params\_if\_you\_need’)}

模板相关函数:

display(‘view下面的目录/目录下的模板文件’);

layout($html\_file,$template = ‘layout’)

$html\_file,$template为模板文件相对于工程目录下的view文件夹的位置,不需要”.html”后缀.

Layout.html文件第一行必须是{\_\_LAYOUT\_\_},且此行会被替换成DHTML的标准头部,此外还需要{\_\_CONTENT\_\_}标示,以标明内容的位置.{\_\_CONTENT\_\_}标示可以出现多次,但是每次出现的地方会被替换成同样的内容.

文件会以路径md5的文件名存放于工程的cache目录下.

快速方法:

M($model\_name,array($params)))，$model\_name大小写敏感,如果$params不是array，会被放入array数组中。注意：

1．只有一个数组参数，且需要用到该数组的key索引，则必须以如下规则调用

M(‘SomeModel’,array($array\_param))

2．如果使用的为SmallMVC框架自带的SmallMVCModel，则$params的第一个元素必须为null或者一个已经在config.php中定义的数据库连接池(pool)名称。

C($controller\_name, mix $params\_pass\_to\_controller)，$model\_name大小写敏感

import($name):如果$name以’@.’开头，则只在本工程目录下搜索，否则出了在工程目录下搜索外，还会再环境变量中搜索。如果名字中有’.’则视为目录分割符号。注意：import的文件建议放在项目的plugins目录下。

数据库配置文件：

当模型M($modelName, $poolName)的时候，此函数首先会实例化一个modelName的Model出来（如果该model不存在，则强制使用SmallMVCModel.php中的类）。然后，会根据Model中传入的数据库连接池（$poolName）名连接数据库，可以传入不通的poolName，实现向不同的数据库连接。连接用的驱动由$config[$poolName][‘plugin’]制定。

每一个数据库连接池存放于SMvc对象的$dbs数组变量中

第三方模板引擎相关:

模板引擎需要有的几个public函数:layout display assign

框架自动载入功能：

注意：目前有命名规则检测机制，由于命名规则还不成熟，故命名规则，请查看core/SmallMVCLoader.php下面相应函数的名称检测部分。本文档不做额外说明。

框架可以自动载入3种形式的对象：script/model/library

Script:script只能包含类/函数，不能包含变量，如需变量，请将其封装在特定的对象之中。Script不会自动被实例化。可配置$config[‘autoloads’][‘scripts’]。一旦script被加载进来，就可以直接调用其函数，或实例化其对象。

Model:只能包含类，会自动实例化对象，可以配置config.php中的$config[‘autoloads’][‘models’] = array(‘model\_name’) 或 = array( ‘model\_name’ => arry(‘model\_params’)),这里的model\_name（如果不为空）会被用于当作存储该model对象的变量名存放在controller对象

的$model\_name中。

Library:只能包含类，会自动实例化对象。如果以”@.”开头，则只在项目目录下面寻找该类库（目前不支持子目录）。可以配置config.php的$config[‘autoloads’][‘libraries’] = array(‘library\_name’) 或 = array( ‘library\_name’ => array(‘library\_params’)),该对象会被存储于SMvc::instance($library, $library\_name)中。

函数（不完整）

SMVCController::redirect($url = 跳转url, $time = 跳转延迟时间默认0, $message = 跳转提示信息)

SmallMVCModel.php类的记录（非文档，自己记录用的）：

Function where($clause, $args)

$clause 为条件，例如 id = ?, id, id < 都为合法内容，当为id 这种形式时，会自动追加 = ?。当包含?的时候，$args数组中为替换?的内容，有几个问号，$args就要有几个元素