

# Bizpower CRM 二次开发手册

2013/07/10 by Bizpower

# 1. 简介

Bizpower CRM 是电企动力公司旗下服务于电商企业客户的 CRM 系统 基于 PHP5 技术采用 MVC 模式开发，系统经过大量优化与重构，在易用性、扩展性和性能方面不断改进，使系统在数据库操作、缓存机制、会话处理、安全访问、SEO 支持、大数据量访问与处理等方面不断完善，系统更加高效与稳定。

Bizpower CRM 可以支持 Windows/Unix 服务器环境，需要 PHP5.0 及以上版本支持，可运行于包括 Apache、IIS 和 Nginx 在内的多种 WEB 服务器，支持 MySQL 数据库。

Bizpower CRM 为广大电商企业客户保驾护航，我们一直在努力。

## 2. 架构设计

### 2.1 系统特性

Bizpower CRM 借鉴了国内外优秀的开源程序与开发模式，使用面向对象的开发结构、MVC 模式、单一入口、ORM 影射等。封装了 CURD 和一些常用操作，在项目配置、类库引入、查询语言、数据验证、模型处理、视图与布局、项目压缩、SEO 支持、分布式数据库支持及扩展性等方面有独特的表现。

#### ● MVC 设计

Bizpower CRM 融合了 MVC 模式进行开发，系统开发高效，各个节点结构更加清晰。

模型(M)：模型的定义由 Model 类来完成。

控制器(C)：由框架核心和 Action 共同完成。

视图(V)：由 tpl 类和模板文件组成。

MVC 作为一种模式只是提供了一种敏捷开发的手段，Bizpower CRM 系统融入 MVC 模式但不拘泥于 MVC 本身。

#### ● 框架压缩

Bizpower CRM 引入框架压缩机制，系统运行时直接加载压缩文件，有效减少 I/O 开销，只需在入口文件中定义

```
define('BUILDCORE',true);
```

即可将系统运行所需要的众多框架文件压缩至 cache/~bizpower.php 中，当再次运行系统时，系统只需

执行 `cache/~bizpower.php` 即可完成框架的加载任务。压缩任务由 `framework/function/build.php` 的 `build()` 方法来完成。

- **缓存机制**

商城系统支持包括文件缓存、数据表缓存以及从种内存缓存 ( APC、Memcache、eAccelerator 和 Xcache ), 用户可跟据实际运行环境自行设置。

- **调试模式**

系统提供了调试模式, 可用于开发过程的不同阶段, 包括开发、测试和演示等情况, 满足调试开发过程中的日志和分析需要, 确定将来系统以最佳的方式进行部署。

- **高效的 API 调用接口**

系统提供了便捷的 API 调用接口, API 调用不需要完全执行系统框架, 提高了响应速度。

- **多种数据库支持**

支持 MySQL、Oracle 数据库, 方便不同级别类型的客户使用。

- **查询机制丰富**

系统内建丰富的查询机制, 包括组合查询、复合查询、区间查询、统计查询、定位查询、多表查询、子查询和原生查询, 使用数据查询简洁高效。

- **动态模型**

系统中无需创建对应的模型类、即可轻松完成 CURD 操作, 使数据库操作更加简洁。

- **支持字段检测**

系统支持缓存字段信息, 支持非法字段过滤和字段类型强制转换, 确保数据操作的安全性。

- **高效的搜索机制**

系统使用文件缓存、数据表缓存、内存缓存、sphinx 搜索相结合, 最大化减少搜索给系统带来的影响。

- **支持数据库分布式部署**

系统支持对数据库分布式部署。

- **多语言支持**

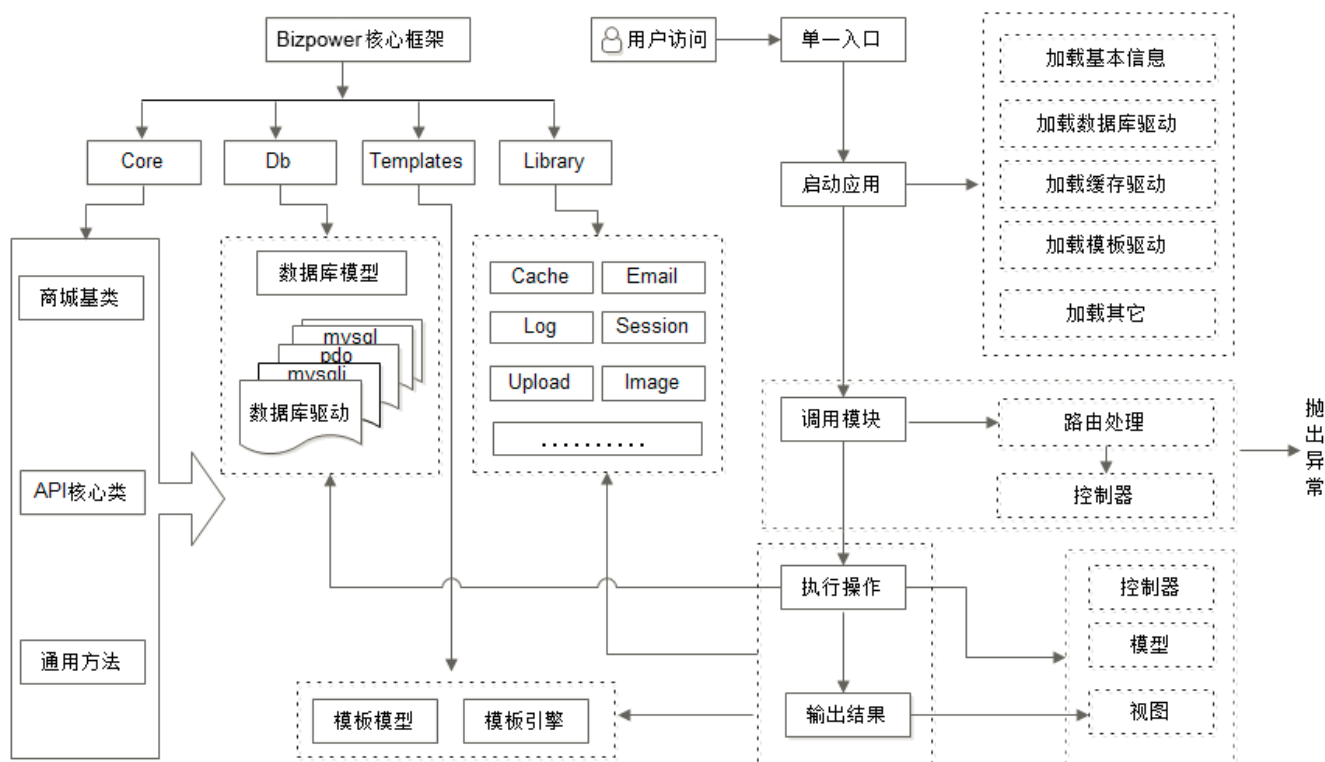
系统内置了简体中文与繁体中文语言包, 并可以跟据自己的运营需求自行扩展。

## 2.2 执行流程

系统采用 `index.php` 作为统一入口, 主要代码如下:

```
define('ProjectName','');           //此处定义为空
require(dirname(__FILE__).'/global.php');    //引入全局文件
require(dirname(__FILE__).'/config.ini.php'); //引入基本配置文件
```

```
require(BasePath.'/framework/core/runtime.php'); //开始执行框架
```



Bizpower CRM 框架执行流程如下：

## 2.3 目录结构

新版对原有目录进行了优化调整，结构更加清晰。系统主要目录：

- admin**      后台管理（可更改）
- api**        API 文件目录
- cache**      缓存文件
- control**    控制器文件
- data**        常用固定数据
- framework** 系统框架
- framework > cache**    缓存驱动
- framework > core**      核心文件
- framework > db**        数据层驱动
- framework > function** 系统方法

**framework > libraries**    类库  
**framework > tpl**        模板驱动  
**install**        安装文件目录  
**language**    语言包  
**model**        模型文件  
**resource**    外部资源  
**templates**   模板目录  
**upload**       存放附件  
**index.php**   入口文件

## 2.4 MVC 设计

Bizpower CRM 融入 MVC 模式进行开发，系统开发高效，各个节点结构更加清晰。

模型（ M ）：模型的定义由 Model 类来完成。

控制器（ C ）：由框架核心和 Action 共同完成。

视图（ V ）：由 Tpl 类和模板文件组成。

MVC 作为一种模式只是提供了一种敏捷开发的手段，Bizpower CRM 系统应用 MVC 但不拘泥于 MVC 本身。

## 2.5 控制器

Bizpower CRM 控制器类位于 control 目录，控制器调度由 framework/core/base.php 中 control() 方法依据 act 和 op 参数完成，如果 act 或 op 参数为空，系统会自动赋值 “index”。

控制器类文件名一般为业务名称，类名称一般为 业务名称 + “Control”，如品牌控制器类文件命名为 control/brand.php，类名为 brandControl。

<http://<siteurl>/index.php>

将会执行 control/index.php 中的 indexOp()方法

<http://<siteurl>/index.php?act=brand&op=list>

将会执行 control/brand.php 中的 listOp()方法

## 2.6 模型

Model 的使用非常灵活，可以无需进行任何模型定义，就可以完成对相关数据表的 CURD 操作，这

就是动态模型处理，不需要重复进行模型实例化即可实现对不同表的操作。新模型的处理支持向下兼容。

使用 Model()方法创建模型，可以创建一个空模型实例，然后使用该实例完成对不同表的操作，如：

```
$model = Model();  
$model->table('member')->find(5); // 查询主键 ID 为 5 的会员信息  
$model->table('brand')->delete(5); // 删除主键为 5 的品牌
```

也可以实例化一个非空模型实例，如：

```
$model = Model('member');
```

系统首先会查找 model/member.model.php 文件及内部的 memberModel 类是否存在：

- a) 如果存在，将实例化 member.model.php 中的 memberModel 类，如果需使用框架已封装的方法 ( select、find、delete、insert 等 )，需要在 memberModel 类中继承 Model 类并在构造方法中触发父类构造方法，

```
class memberModel extends Model{  
    public function __construct(){  
        parent::__construct('member');  
    }  
    //除使用系统提供方法以外，还可以自定义方法  
    //public function myfuc(){  
        //添加业务逻辑  
    }  
}
```

- b) 如果不存在，将实例化 framework/core/model.php 中的 Model 类，也就是只实例化框架提供的模型类（动态模型）

```
$model = Model('member');  
$model->find(5); // 查询主键为 5 的会员信息  
$model->table('brand')->delete(5); // 删除主键为 5 的品牌，即使创建 member 模型，还是可以使用该模型的 table()方法来操作其它表
```

## 2.7 数据库抽象层

Bizpower CRM 提供了多个数据库访问驱动，目前支持 mysql、mysqli、pdo\_mysql 和 oci8 驱动，支持 MySQL 和 ORACLE 数据库，系统会根据当前的数据库配置，自动调用相应的数据层驱动，同时系统还支持数据库的分布式配置，为企业客户保驾护航。

## 2.8 视图

视图功能主要由 Tpl 类 ( framework/tpl/nc.php ) 和模板文件组成 ( 位于 templates 目录下 ), Tpl 类完成控制器和模板文件的沟通, 控制器通过 Tpl 类将数据输送到模板, 然后由模板输出数据, Bizpower CRM 未使用特定的模板语言, 而是使用原生的 PHP 语法, 这样省去了解析模板语言的时间, 加快响应速度。

## 2.9 函数和类库

### 函数

系统函数均存放在 framework/function 目录中, 考虑到以后的扩展, 系统跟据功能将函数拆分进入不同的文件中,

core.php	存放系统的通用函数
goods.php	存放处理商品、订单、店铺相关的信息的函数
ftp.php	存放 ftp 操作的相关函数
seccode.php	存放验证码操作的函数
build.php	存放框架压缩的函数

### 类库

系统类库主要存放在 framework 下的 libraries、cache、core 和 tpl 目录中, librarites 存放的主要是完成专项功能的类, cache 下是存放多种缓存驱动类, core 下存放的是核心基类和模型处理类, tpl 只存放视图类 Tpl。

libraries/email.php	邮件发送类
libraries/ftp.php	ftp 处理类
libraries/gdimage.php	水印类
libraries/json.php	json 处理类
libraries/language.php	语言包处理类
libraries/log.php	日志类
libraries/page.php	分页类
libraries/resizeimage.php	图片裁切类
libraries/seccode.php	验证码处理类
libraries/security.php	字符过滤类
libraries/sphinx.php	sphinx 全文检索类

libraries/upload.php	上传类
libraries/excel.php	excel 导出类
cache/ cache.php	缓存统一处理类，它是每种缓存类的操作入口
cache/cache.file.php	文件缓存类
cache/ cache.apc.php	apc 缓存类
cache/cache.xcache.php	xcache 缓存类
cache/cache.memcache.php	memcache 缓存类
cache/ cache.eaccelerator.php	eaccelerator 缓存类
core/model.php	模型类
core/db.php	数据库操作中间层，介于模型和数据驱动中间的中间处理类
db/mysql.php	mysql 扩展 MySQL 数据库驱动
db/mysqli.php	mysqli 扩展 MySQL 数据库驱动
db/pdo_mysql.php	pdo_mysql 扩展 MySQL 数据库驱动
db/oci8.php	oci8 扩展 ORACLE 数据库驱动
tpl/nc.php	模板类(视图类)

## 3. 开发指南

### 3.1 配置文件

Bizpower CRM 的配置文件由 config.ini.php 和 cache/setting.php 组成，setting.php 内的配置项可进入系统后台进行设置，此处不介绍。config.ini.php 配置项需手动更改，配置项均存放在\$config 数组中，主要配置参数(下标)如下：

#### **site\_url**

网站地址，系统安装时自动生成（最后不要加斜杠 "/"）

#### **version**

版本号，该项为安装时自动生成，不需要更改

#### **setup\_date**

安装日期，安装时自动生成，不需要更改

#### **gip**

是否开启 gzip，此项需要服务器支持



**dbdriver**

数据库连接驱动 mysql(默认),mysql,pdo\_mysql,oci8

**tablepre**

数据表前缀

**db[1]**

主数据库配置

**db[slave]**

从数据库配置，可设置多台从数据库，如果只有一个数据库，只配置主数据库即可

**cache[file]**

缓存存储类型，支持类型为 file,memcache,xcache,apc,eaccelerator，默认为 file(文件缓存)

**debug**

是否开启调试模式，开启后将在底部显示系统运行状态信息

**system\_close**

系统是否关闭

**close\_reason**

系统关闭原因

**graphexportfile**

导出图表插件的文件名称

**graphfont**

统计图表字体

**crmbasepath**

服务器物理目录，无需设置

**exportfilefont**

导出文件字体，例如 Excel 中的字体

## 3.2 控制器

系统控制器类位于 control 目录，控制器调度由 framework/core/base.php 中 control()方法依据 act 和 op 参数完成，如果 act 或 op 参数为空，系统会自动赋值 “index”。

控制器类文件名一般为业务名称，类名称一般为 业务名称 + “Control”，例如系统的品牌控制器类文件为 control/brand.php，类名为 **brandControl**。

跟据商城业务需要，系统内置三个控制器父级类，**BaseHomeControl**、**BaseMemberControl** 和

BaseMemberStoreControl 分别适用于前台展示、会员中心、店铺中心三类控制器，品牌展示需要继承 BaseHomeController 类。

```
<?php

/**
 * 品牌展示
 *
 * 版权声明...
 */

defined('InBizpower CRM') or exit('Access Invalid!');
class brandControl extends BaseHomeController {
    public function indexOp(){
        //读取语言包
        Language::read('home_brand_index');
        //使用模型获得品牌列表
        $model = Model();
        $brand_list = $model->table('brand')->select();
        //向模板抛出内容
        Tpl::output('brand_list',$brand_list);
        //设置页面标题
        Tpl::output('html_title',Language::get('brand_index_brand_list'));
        //输出 SEO 设置信息
        Model('seo')->type('brand')->show();
        //调用模板展示
        Tpl::showpage('brand');
    }
    public function searchOp(){
        /**
         * 内容略...
         */
    }
}

?>
```

访问 <http://<siteurl>/index.php?act=brand>

将会执行 brandControl 类的 indexOp 方法

访问 <http://<siteurl>/index.php?act=brand&op=search>

将会执行 brandControl 类的 searchOp 方法

## 3.3 模型

### 3.3.1 实例化

使用 Model()方法创建模型，可以创建一个空模型实例，然后全用该实例完成对不同表的操作，如：

```
$model = Model();
```

也可以实例化一个非空模型实例，如：

```
$model = Model('member');
```

系统首先会查找 model/member.model.php 文件及内部的 memberModel 类是否存在，实例化该类，如果不存在，则实例化框架提供的模型类，实例化模型更详细的信息可查看 [2.6 模型](#)

### 3.3.2 内置方法

系统模型提供了一系列快捷操作的方法，可以大幅提高开发效率。目前已提供的方法主要有 select、find、limit、table、order、where、field、on、join、count、page、attr、showpage、insert、insertAll、delete、update、group、having、distinct、clear、query、execute、sum、avg、max、min、setInc、setDec、和动态方法 getby\_、getfby\_。

**1. Select 方法：**取得查询信息，返回结果为数组，如果未找到数据返回 null，select 一般在 where、order、table 等方法的后面，作为最后一个方法来使用。如：

```
$model = Model('member');
```

```
// 查询会员表全部信息
```

```
$model->select();
```

```
//取得性别为 1 的会员列表信息, 注意：select 方法需要在连贯操作中的最后一步出现
```

```
$model->where(array('member_sex'=>1))->select();
```

Select 方法可以传入主键 ID，系统会自动查找对应信息，如：

```
// 查询主键 ID 为 5 的会员信息
```

```
$model = Model('member');
```

```
$model->select(5);
```

2. **Find** 方法：取得一条记录信息,find 同 select 一样，一般作为最后一个方法来使用，如：

```
$model = Model('member');  
// 查询 ID 为 5 的会员信息  
$model->where(array('member_id'=>5))->find();
```

Find 方法可以传入主键 ID，系统会自动查找对应信息，如：

```
$model = Model('member');  
// 查询主键 ID 为 5 的会员信息  
$model->find(5);
```

3. **Limit** 方法：指定返回多少条记录数，

```
$model = Model('member');  
$model->limit(4)->select(); // 等同于 SELECT * FROM member LIMIT 4;  
$model->limit('4,10')->select(); // 等同于 SELECT * FROM member LIMIT 4,10;
```

4. **Table** 方法：指定要操作的数据表名称，返回模型实例本身，如：

```
$model = Model();  
// 查询主键 ID 为 5 的会员信息  
$model->table('member')->find(5);
```

多表联合查询时，可以传入多个表名称，如：

```
// 内链接查询 member 和 store 表,并返回前两条记录  
$on = 'store.member_id=member.member_id';  
$model->table('member,store')->join('inner')->on($on)->limit(2)->select();
```

如果实例化时指定了表名，则可以不使用 table 方法指定表名，如：

```
$model = Model('member');  
$model->limit(4)->select(); // 查询前条 4 会员记录
```

5. **Order** 方法：指定排序的参数，返回模型实例本身，如：

```
$model->table('member')->order('member_id desc')->limit(4)->select();
```

也可指定多个字段进行排序，如：

```
$model->table('member')->order('member_id desc,member_sex asc')->select();
```

6. **Where** 方法：指定 sql 执行的条件，返回模型实例本身，入可传入数组或字段串，如：

```
//传入数组条件  
$model->where(array('member_id'=>5))->find();  
//传入字符串条件
```

```
$model->where(array('member_id=5'))->find();  
//传入多表关联条件  
$model->table('member,store');  
$model->where('store.store_id=member.store_id and store.store_id=2')->find();
```

7. **Field** 方法：指定要查询的字段，不使用 field 方法时，默认查询所有字段，如：

```
$model->field('member_id,member_name')->select();
```

8. **On** 方法：指定多表联查时的字段关系。

9. **Join** 方法：指定多表联查时的链接类型，支持内链接、左链接(默认)、右链接。On 与 join 方法需要一起使用，如：

```
$model = Model();  
//内链接查询 member 和 store 表,返回会员 ID 为 6 的记录信息  
$field = 'member.member_name,store.store_name';  
$on = 'store.member_id=member.member_id';  
$model->table('member,store')->field($field);  
$model->join('inner')->on($on)->where(array('member.member_id'=>6))->find();
```

三表关联查询如下：

```
$model = Model();  
//内链接查询 member 和 store,然后左链接 store_class,查询会员 ID 为 6 的记录信息  
$field = 'member.member_name,store.store_name,store_class.sc_name';  
$on = 'store.member_id=member.member_id,store.sc_id=store_class.sc_id';  
$model->table('member,store,store_class')->field($field);  
$model->join('inner,left')->on($on)->where('member.member_id=6')->find();
```

10. **Count** 方法：返回记录总数量，如：

```
$model = Model('member');  
//返回会员表总行数  
$model->count();  
//返回会员 ID 大于 15 的记录数  
$model->where('member_id>15')->count();
```

11. **Page** 方法：实现记录分页，格式为 page(每页显示数，总记录数)，总记录数可以人为指定，也可以为空让系统自动去计算，如：

```
//每页显示 10 条数据  
$model = Model('member');
```

//系统会根据每页显示数和已知属性自动计算总记录数

```
$model->page(10)->order('member_id desc')->select();
```

//每页显示 10 条数据，指定总记录为 1000 条，系统将不再计算总记录数

```
$model->page(10, 1000)->order('member_id desc')->select();
```

注意：如果同时使用 where 和 page 方法时，where 方法要在 page 方法前面使用，如：

```
$model->where('id=1')->page(10)->select(); //正确
```

```
$model->page(10)->where('id=1')->select(); //错误
```

**12. Showpage** 方法：返回分页超链接，结合 page 方法完成分页，如：

//显示上一页下一下链接

```
$model->showpage(1); //样式 1
```

```
$model->showpage(2); //样式 2(默认)
```

**13. Insert** 方法：插入单行数据，并返回最新插入的主键 ID，如果插入失败返回 false,完整格式如下：

```
insert($data="", $replace=false, $options=array())
```

//向 link 表插入数据，并返回最新主键 ID

```
$model = Model('table');
```

```
$data = array(
```

```
    'link_title'=>'Bizpower CRM',
```

```
    'link_url'=>'http://www.Bizpower CRM.net',
```

```
    'link_sort'=>32,
```

```
);
```

```
$model->insert($data);
```

Insert 方法支持延迟插入，加入 \$options 参数即可，如：

```
$model->insert($data,false,array('priority'=>'DELAYED'));
```

Insert 方法同样支持 replace 操作，将第二个参数设置为 true 即可，如：

```
$model = Model();
```

```
$data = array(
```

```
    'link_title'=>'Bizpower CRM',
```

```
    'link_url'=>'http://www.Bizpower CRM.net',
```

```
    'link_sort'=>32,
```

```
    'link_id'=>30
```

```
);
```

```
$model->table('link')->insert($data,true);
```

**14. InsertAll** 方法：实现批量插入数据，如：

```
$model = Model('link');
$data = array(
    array(
        'link_title'=>'新浪',
        'link_url'=>'http://www.sina.com',
        'link_sort'=>32,
    ),
    array(
        'link_title'=>'百度',
        'link_url'=>'http://www.baidu.com',
        'link_sort'=>30,
    )
);
$model->insertAll($data);
```

**15. Delete** 方法：删除记录，如：

```
$model = Model('link');
//删除主键为 5 的记录
$model->delete(5);
//或者
$model->where(array('link_id'=>5))->delete();
```

**16. Update** 方法：数据更新，如果更新内容含有主键下标，自动以该主键为更新条件，如：

```
$model = Model();
//更新主键(link_id)为 37 的记录信息
$data = array(
    'link_title'=>'Bizpower CRM',
    'link_url'=>'http://www.shponc.net',
    'link_sort'=>32,
    'link_id'=>37
);
$model->table('link')->update($data);
//指定更新条件
```

```
$data = array(  
    'link_title'=>'Bizpower CRM',  
    'link_url'=>'http://www.shponc.net',  
    'link_sort'=>32  
);  
$model->table('link')->where(array('link_id'=>37))->update($data);
```

**17. Group** 方法：实现分组功能，如：

```
//查询每个店铺发布商品的数量  
$model = Model('goods');  
$model->field('store_id,count(*) as count')->group('store_id')->select();
```

**18. Having** 方法：结合 group 方法，进行条件过滤，传入参数为字符串形式，如：

```
//查找发布商品超过 500 的店铺 ID  
$model = Model('goods');  
$model->field('store_id,count(*) as nc_count')->group('store_id')->having('nc_count>500')->select();
```

**19. Distinct** 方法：可以去除列中相同的值，distinct 只接受一个参数值 true,如果不需要重复值筛选，不使用该方法即可。

```
//查找拥有商品的店铺主键  
$model = Model();  
$model->table('goods')->field('store_id')->distinct(true)->select();
```

**20. Clear** 方法：清空单个表中的内容，返回 true/false，如：

```
//清空 link 表  
$model = Model();  
$model->table('link')->clear();
```

**21. Query/execute** 方法，两个方法均用于直接执行 SQL 语句，query 方法用于查询，execute 方法用于更新、写入和删除操作，如：

```
Model()->query('SELECT * FROM `Bizpower CRM_member` LIMIT 10');  
Model()->execute('UPDATE `Bizpower CRM_goods` SET goods_click=1000 WHERE goods_id=2');
```

**22. Sum/Avg/Max/Min** 方法：求和、求平均值、取最大值、取最小值，如：

```
$model = Model();  
//返回所有商品总价格之和
```



```
$model->table('goods')->sum('price');  
//上面等同于 SQL : SELECT SUM(price) AS nc_sum FROM `goods`  
  
//取商品表中所有商品的平均价格  
$model->table('goods')->avg('price');  
//以上等同于 SQL : SELECT AVG(price) AS nc_avg FROM `goods` LIMIT 1  
  
//取商品的最高价  
$model->table('goods')->max('price');  
//以上等同于 SQL : SELECT MAX(price) AS nc_max FROM `goods` LIMIT 1  
  
//取商品的最低价  
$model->table('goods')->min('price');  
//以上等同于 SQL : SELECT MIN(price) AS nc_min FROM `goods` LIMIT 1
```

**23. 自增/自减：**系统使用 setInc 和 setDec 完成自增和自减，示例如下：

```
$model = Model();  
//使主键值为 2 的商品点击量加 1000  
$model->table('goods')->where(array('goods_id'=>2))->setInc('goods_click',1000);  
//等同于：UPDATE `goods` SET goods_click=goods_click+3 WHERE ( goods_id = '2' )  
  
//结合 exp 参数，使用该商品点击量减 1000  
$model = Model('goods');  
$data = array(  
    'goods_id' => 2,  
    'goods_click' =>array('exp','goods_click-1000'));  
$model->update($data);  
//等同于：UPDATE `goods` SET goods_click=goods_click-1000 WHERE ( goods_id = '2' )
```

**24. 动态方法：**系统内置 getby\_和 getfby\_两个动态方法，格式如下：

getby\_ + 字段名 ( 字段值 )

getfby\_ + 条件字段名 ( 条件字段值，返回字段名 )

结合示例来说明动态方法的使用

```
$model = Model('member');
```

//使用 getby\_动态方法，取得 member\_name 为 kevin 的会员信息

```
$model->getby_member_name('kevin');
```

//等同于 SQL : SELECT \* FROM `member` WHERE ( member\_name = 'kevin' ) LIMIT 1

//使用 getfby\_方法，取得 member\_id 为 6 的会员名

```
$a = $model->getfby_member_id(6,'member_name'); //返回 kevin
```

//等同于 SQL : SELECT member\_name FROM `Bizpower CRM\_member` WHERE ( member\_id = '6' ) LIMIT 1

**25. 设置 SQL 执行优先级：**系统支持使用 SQL 关键字 LOW\_PRIORITY、DELAYED、HIGH\_PRIORITY，格式如下：

attr ( 关键字 )

结合示例来说明动态方法的使用

```
$model = Model('goods');
```

```
$model->where(array('goods_id' => 100))->attr('LOW_PRIORITY');
```

```
$model->update(array('goods_click' => array('exp','goods_click+1')));
```

//等同于

```
//UPDATE LOW_PRIORITY `Bizpower
```

```
CRM_goods` SET goods_click=goods_click+1 WHERE ( goods_id = '100' )
```

### 3.3.3 CURD 操作

#### 3.3.3.1 读取数据

系统中可使用 select、find、query 方法完成查询操作。

使用 select 方法查询信息：

```
$model = Model('member');
```

// 查询会员表全部信息

```
$model->select();
```

//取得性别为 1 的会员列表信息，注意：select 方法需要在连贯操作中的最后一步出现

```
$model->where(array('member_sex'=>1))->select();
```

// 查询主键 ID 为 5 的会员信息

```
$model->select(5);
```

使用 find 方法查询信息：

```
$model = Model('member');  
// 查询 ID 为 5 的会员信息  
$model->where(array('member_id'=>5))->find();  
// 或者  
$model->find(5);
```

使用 query 方法取得查询信息：

```
Model()->query('SELECT * FROM `Bizpower CRM_member` LIMIT 10');
```

使用动态方法取得查询信息：

```
$model = Model('member');  
//使用 getby_动态方法，取得 member_name 为 kevin 的会员信息  
$model->getby_member_name('kevin');  
//等同于 SQL : SELECT * FROM `member` WHERE ( member_name = 'kevin' ) LIMIT 1  
  
//使用 getfby_方法，取得 member_id 为 6 的会员名  
$a = $model->getfby_member_id(6,'member_name'); //返回 kevin  
//等同于 SQL : SELECT member_name FROM `Bizpower CRM_member` WHERE ( member_id =  
'6' ) LIMIT 1
```

### 3.3.3.2 更新数据

系统可使用 update、execute 方法完成更新操作。

```
$model = Model();  
//更新主键(link_id)为 37 的记录信息  
$data = array(  
    'link_title'=>'Bizpower CRM',  
    'link_url'=>'http://www.shponc.net',  
    'link_sort'=>32,  
    'link_id'=>37  
);  
$model->table('link')->update($data); // 系统自动以主键 link_id 为更新条件  
//指定更新条件
```

```
$data = array(
    'link_title'=>'Bizpower CRM',
    'link_url'=>'http://www.bizpower.com',
    'link_sort'=>32
);
$model->table('link')->where(array('link_id'=>37))->update($data);
//使用 execute 方法执行更新
Model()->execute('UPDATE crm_goods` SET goods_click=1000 WHERE goods_id=2');
```

### 3.3.3.3 插入数据

系统可使用 insert、insertAll、execute 方法完成插入操作。

使用 insert 方法插入单行数据：

```
//向 link 表插入数据，并返回最新生成的主键 ID
$model = Model('table');
$data = array(
    'link_title'=>'Bizpower CRM',
    'link_url'=>'http://www.Bizpower CRM.net',
    'link_sort'=>32,
);
$model->insert($data);
```

Insert 方法支持延迟插入，加入 \$options 参数，如：

```
$model->insert($data,false,array('priority'=>'DELAYED'));
```

使用 Insert 方法执行 replace 操作，将第二个参数设置为 true，如：

```
$model = Model();
$data = array(
    'link_title'=>'Bizpower CRM',
    'link_url'=>'http://www.Bizpower CRM.net',
    'link_sort'=>32,
    'link_id'=>30
);
$model->table('link')->insert($data,true);
```

使用 insertAll 方法：实现批量插入数据：

```
$model = Model('link');  
$data = array(  
    array(  
        'link_title'=>'新浪',  
        'link_url'=>'http://www.sina.com',  
        'link_sort'=>32,  
    ),  
    array(  
        'link_title'=>'百度',  
        'link_url'=>'http://www.baidu.com',  
        'link_sort'=>30,  
    )  
);  
$model->insertAll($data);
```

也可以使用 execute 直接执行 SQL 语句来插入数据。

### 3.3.3.4 删除数据

系统可使用 delete、clear、execute 方法完成删除操作。

使用 delete 方法删除数据：

```
$model = Model('link');  
//删除主键为 5 的记录  
$model->delete(5);  
//或者  
$model->where(array('link_id'=>5))->delete();
```

使用 clear 方法清空数据：

```
//清空 link 表  
$model = Model();  
$model->table('link')->clear();
```

也可以使用 execute 直接执行 SQL 语句来删除数据。

### 3.3.4 运算符

系统对常用运算符的使用进行了二次封装，使用方便、快捷。

gt : 大于 ( > )

egt : 大于等于 ( >= )

lt : 小于 ( < )

elt : 小于等于 ( <= )

eq : 等于 ( = )

neq : 不等于 ( != )

notlike : NOT LIKE

like : 同 sql 中的 LIKE

between : 同 sql 中的 BETWEEN

[not] in : 同 sql 中的 [NOT] IN

示例：

//为便于演示，这里将所有运算符的使用均罗列出来，以下代码不可直接运行

```
$condition=array()
```

```
// uid > 5
```

```
$condition['uid'] = array('gt',5);
```

```
// uid < 5
```

```
$condition['uid'] = array('lt',5);
```

```
// uid = 5
```

```
$condition['uid'] = array('eq',5);
```

```
// uid >= 5
```

```
$condition['uid'] = array('egt',5);
```

```
// uid <= 5
```

```
$condition['uid'] = array('elt',5);
```

```
// uid 在 3,5,19 之间一个或多个
```

```
$condition['uid'] = array('in','3,5,19');
```

```
// uid 是 3,5,19 中的任何值
```

```
$condition['uid'] = array('not in','3,5,19');
```

```
// 5 <= uid <= 19
```

```
$condition['uid'] = array('between','5,19');
```

```
//product_name like 'a%'
$condition['product_name'] = array(array('like','a%'));
// product_name like 'a%' or product_name like 'b%'
$condition['product_name'] = array(array('like','a%'),array('like','b%'),'or');
//会员昵称或姓名有一个含有 Bizpower CRM 字样的即可满足
$condition['member_name|member_tname'] = array(array('like','%Bizpower CRM%'));
//会员昵称或姓名都必须含有 Bizpower CRM 字样的才可满足
$condition['member_name&member_tname'] = array(array('like','%Bizpower CRM%'));
//以上各条件默认均是 "AND" 关系，即每个条件都需要满足，如果想满足一个即可 ( "OR" 关系 ),
可增加以下条件
$condition['_op'] = 'or';
//最后将以上条件传入 where 方法
$list = Model(TABLE)->where($condition)->select();
```

### 3.4 视图

系统采用 MCV 模式，由视图类 Tpl 将变量抛到模板并进行输出，使用 `setDir` 设置模板目录，使用 `output` 方法抛出变量，使用 `showpage` 显示模板，抛出的变量会赋值到模板的 `$output` 数组中。

```
Tpl::setDir('home'); // 指定模板位于 templates/default/home
Tpl::output('name','Bizpower CRM'); // 向模板抛出变量
Tpl::showpage('index'); // 显示模板 /templates/default/home/index.php
echo $output['name']; // 在模板中使用$output[$var]直接输出
```

### 3.5 调试

开启调试模式可以看到更加详细的系统运行信息，调试模式的作用在于显示更多的运行日志信息，以便在项目开发过程中快速定位和解决问题。开启调试状态的效果如下：

#### 页面Trace信息

```
Array ( [当前页面] => /index.php?act=login
[请求时间] => 2012-10-12 20:56:20
[页面执行时间] => 0.065s
[占用内存] => 2.48MB
[请求方法] => GET
[通信协议] => HTTP/1.1
[用户代理] => Mozilla/5.0 (Windows NT 6.1; rv:15.0) Gecko/20100101 Firefox/15.0.1
[会话ID] => 7s733dr4beku2kvqfa1jjq2le7
[日志记录] => 无日志记录
[加载文件] => 36
[0] => D:\root\ShopNC_V2.3\trunk\index.php
[1] => D:\root\ShopNC_V2.3\trunk\global.php
```

开启调试模式需要在 config.ini.php 中设置

```
$config['debug'] = true;
```

对于 2.3 以前的版本，需要在系统后台 > 站点设置 中开启调试模式：

调试模式：



## 3.6 缓存

系统可以对数据进行多种形式缓存，包括文件方式、共享内存方式和数据库方式。目前已支持的缓存方式包括：file、apc、eAccelerator、memcache 和 xcache，开发者也可以开发更加适合自己的缓存。

系统由 Cache 缓存类进行缓存操作，由 `Cache::getInstance` 方法完成缓存类型实例化：

```
//取得 memcache 缓存实例
$obj_cache = Cache::getInstance('memcache');

//缓存赋值
$obj_cache->set('uid',1000);

//读取缓存
$obj_cache->get('uid');

//删除缓存
$obj_cache->rm('uid');
```

系统需要将部分常用表信息（如基本配置表、商品分类表、快递表、SEO 表等）生成缓存，为了便于操作，系统对这些表的缓存处理封装到专有的 cache 模型中，使用 `Model('cache')->call(表名)` 即可得到符合缓存格式的表信息。

为了方便对缓存的操作，系统新增了 H 和 F 方法，H 方法对缓存类操作再次进行封装。如取得系统 cache/setting.php 缓存信息：



```
H('setting');
```

内部执行过程为：首先取得 file 缓存实例

```
$obj_cache = Cache::getInstance('file')
```

判断操作类型（删除缓存？、缓存赋值？缓存读取？），然后执行

```
$obj_cache->get('setting');
```

取得缓存内容，至此执行完毕。

使用 H 方法将 setting 表信息生成到文件缓存：

```
H('setting',true,'file');
```

内部执行过程为：首先取得 file 类型缓存实例，

```
$obj_cache = Cache::getInstance('file')
```

判断操作类型（删除缓存？、缓存赋值？缓存读取？），然后执行

```
$obj_cache->rm('setting');
```

删除原有 setting 缓存，实例化 cache 模型并执行 \_setting 方法来取得 setting 表中的内容，

```
$list = Model('cache')->call($key);
```

使用 set 方法写入缓存

```
$obj_cache->set($key, $list, null, $expire);
```

最后返回 true，至此缓存生成成功。

F 方法也可以操作缓存，但它与 H 方法不同，F 只是一种快速度读、写文件缓存的方法，不可用于内存缓存的操作，F 方法不能直接将数据表内容生成缓存，只可将 PHP 中已经存的内容（如数组、字符串等）生成到文件缓存，在读取文件缓存时，推荐使用效率更高的 F 方法。

```
F('setting'); //取得 setting 缓存
```

```
F('setting',$data); //将$data 数组生成到 setting 文件缓存
```

## 3.7 安全

系统为保护目录及文件安全，在所有敏感的目录中放置一个 1 字节的 index.html 文件，内容为一个空格，以避免当 http 服务器的 Directory Listing 打开时，服务器文件被索引和列表。

为防止系统内文件被非法调用，系统会在所使用的.php 文件头部增加有合法性验证：

```
defined('InBizpower CRM') or exit('Access Invalid!');
```

## 4. 函数与类库

### 4.1 常量参考

[ProjectName] => 项目名称

[BUILDCORE] => 是否压缩框架 ( true/false )

[BasePath] => 系统所在基准目录 ( 如 /var/www/Bizpower CRM )

[DS] => 目录分隔符 /

[InBizpower CRM] => 合法调用判断标志 1

[StartTime] => 系统开始运行时间 ( 如 1350096132.6587 )

[TIMESTAMP] => 系统当前时间 ( 如 1350096132 )

[RUNCOREPATH] => 压缩后的框架所在目录 (如 /var/www/Bizpower CRM/cache/~Bizpower CRM.php)

[SiteUrl] => CRMURL 地址 ( http://www.xxx/com )

[CHARSET] => 系统编码 ( UTF-8/GBK )

[DBDRIVER] => 数据库访问驱动 ( mysqli、mysql、pdo\_mysql、oci8 )

[SESSION\_EXPIRE] => 缓存周期, 单位 ( 秒 )

[LANG\_TYPE] => 语言包, 默认简体中文 zh\_cn

[COOKIE\_PRE] => cookie 前缀, 系统自动生成

[CORE\_PATH] => 框架所在目录(如 /var/www/Bizpower CRM /framework)

[TPL\_NAME] => 模板风格, 默认 default

[BASE\_TPL\_PATH] => 模板文件目录(如 /var/www/Bizpower CRM /templates/default)

[RESOURCE\_PATH] => 外部资源目录地址 ( 如 http://www.Bizpower CRM.net/resource )

[DBPRE] => 数据库表前缀 默认 Bizpower CRM\_

[ATTACH\_PATH] => 附件目录, 默认 upload

[ATTACH\_COMMON] => 通用附件目录 默认 upload/common

[ATTACH\_AVATAR] => 头像目录 默认 upload/avatar

[ATTACH\_STORE] => 店铺信息目录 默认 upload/store

[ATTACH\_GOODS] => 店铺产品目录 默认 upload/store/goods

[ATTACH\_AUTH] => 店铺证件目录 默认 upload/auth

[ATTACH\_MOBILE] => 手机客户端内容目录 默认 upload/mobile

[ATTACH\_LINK] => 友情链接 logo 目录 默认 upload/link  
[ATTACH\_ARTICLE] => 文章附件目录 默认 upload/article  
[ATTACH\_BRAND] => 品牌附件目录 默认 upload/brand  
[ATTACH\_ADV] => 广告图片目录 默认 upload/adv  
[ATTACH\_ACTIVITY] => 促销活动目录 默认 upload/activity  
[ATTACH\_COUPON] => 优惠券目录 默认 upload/coupon  
[ATTACH\_WATERMARK] => 水印图片目录 默认 upload/watermark  
[ATTACH\_POINTPROD] => 积分商品图片目录 默认 upload/pointprod  
[ATTACH\_SPEC] => 自定义规格图片目录 默认 upload/spec  
[ATTACH\_GROUPBUY] => 团购图片目录 默认 upload/groupbuy  
[ATTACH\_SLIDE] => 店铺二维码目录 默认 upload/store/slide  
[ATTACH\_VOUCHER] => 代金券模板目录 默认 upload/voucher  
[TEMPLATES\_PATH] => 模板 URL 访问地址 ( 如 <http://www.BizpowerCRM.net/templates/default> )  
[MD5\_KEY] => MD5\_KEY 值 系统自动生成，加密解密时用

## 4.2 主要函数参考

### 4.2.1 系统函数

#### **cookie** 方法

作用：自动加上系统 cookie 前缀并返回 cookie 值

语法：string cookie(\$name= '')

参数：\$name 需要得到的 cookie 名称 ( 不含前缀 )

示例：

```
cookie('uid');
```

#### **encrypt** 方法

作用：加密字符串并返回加密结果

语法：string encrypt(\$txt, \$key = '')

参数：\$txt 待加密的字符串

\$key 密钥

示例：

```
encrypt('password','cb8d4e597fc751a56dc30258c4db1226');// 输出 KHEZ1IzRZpLV9Bg
```

### **decrypt** 方法

作用：解密字符串

语法：string decrypt(\$txt, \$key = '')

参数：\$txt 待解密的字符串

\$key 密钥

示例：

```
decrypt('KHEZ1IzRZpLV9Bg','cb8d4e597fc751a56dc30258c4db1226');// 输出 password
```

### **getIp** 方法

作用：取得 IP 地址

语法：getIp()

### **getReferer** 方法

作用：取得上一步来源地址

语法：string geReferer()

### **Model** 方法

作用：实例化模型

语法：object Model(\$model = null)

参数：\$model 待实例的模型，\$model 也可以为空，这样会实例化一个空模型

示例：

```
$model = Model();
```

//或

```
$model = Model('member');
```

### **redirect** 方法

作用：页面重定向

语法：redirect (\$url= '')

参数：\$url 需要重定向的地址，若为空，系统定向到上一页地址

示例：

```
redirect('http://www.Bizpower CRM.net')
```

### **readDirList** 方法

作用：读取同目录下的文件夹并返回数组

语法：array readDirList(\$path)

参数：\$path 需要读取的目录

示例：

```
readDirList('/var/www/Bizpower CRM');  
//输出  
Array  
(  
    [0] => admin  
    [1] => cache  
    [2] => control  
    [3] => framework  
    [4] => upload  
)
```

### **replaceSpecialChar** 方法

作用：转换特殊字符，将里面的 \r\n, \t, \n 去除

语法：string replaceSpecialChar (\$string)

参数：\$string 需要转换的字符

### **setNcCookie** 方法

作用：保存 cookie

语法：setNcCookie(\$name, \$value, \$expire='3600', \$path='', \$domain='', \$secure=false)

参数：\$name cookie 名称

\$value cookie 值

\$expire cookie 有效期，单位秒，默认 3600

\$path cookie 的服务器路径 默认为 /

\$domain cookie 的域名

\$secure 是否通过安全的 HTTPS 连接来传输 cookie,默认为 false

示例：

```
setNcCookie('uid',1000,2*3600); // 保存 2 小时  
setNcCookie('uid','',time()-3600); // 过期
```

### **showMessage** 方法

作用：输出提示信息

语法：

showMessage(\$msg,\$url='', \$show\_type='html', \$msg\_type='succ', \$is\_show=1, \$time=2000)

参数：

\$msg 待输出的信息

\$url 跳转地址 当\$url 为数组时，结构为 array('msg'=>'跳转连接文字','url'=>'跳转连接')

\$show\_type 输出格式 默认为 html

\$msg\_type 信息类型 succ 为成功，error 为失败/错误

\$is\_show 是否显示跳转链接，默认是为 1，显示

\$time 跳转时间，默认为 2 秒

示例：

```
showMessage('保存成功');
```

```
showMessage('保存失败','index.php?act=brand','html','error');
```

### **showDialog** 方法

作用：消息提示，只适用于页面 AJAX 提交的情况

语法：showDialog(\$message = "", \$url = "", \$alert\_type = 'error', \$extrajs = "", \$time = 2)

参数：

\$message 消息内容

\$url 提示完后的 URL 去向

\$alert\_type 提示类型 error/succ/notice 分别为错误/成功/警示

\$extrajs 扩展 JS 脚本

\$time 消息停留时间，默认为 2 秒

示例：

```
showDialog('保存成功','index.php?act=voucher&op=list');
```

### **setTimeZone** 方法

作用：设置时区

语法：showDialog(\$message = "", \$url = "", \$alert\_type = 'error', \$extrajs = "", \$time = 2)

参数： \$time\_zone 时区键值

示例：

```
setTimeZone(8); //设置东 8 区
```

### **showEditor** 方法

作用：输出编辑器

语法：

showEditor(\$id, \$value="", \$width='700px', \$height='300px',

\$style='visibility:hidden;',\$upload\_state="true", \$media\_open=false)

参数：

\$id 编辑器 id 名称，与 name 同名

\$value 编辑器内容

\$width 宽 带单位 px

\$height 高 带单位 px

\$style 样式内容

\$upload\_state 上传状态，默认开启

示例：

```
showEditor('content','这是内容','600px','400px','visibility:hidden;','false','false');
```

### **getDirSize** 方法

作用：获取目录大小

语法：numeric getDirSize(\$path, \$size=0)

参数：

\$path 目录

\$size 目录大小

示例：

```
$size = getDirSize('d:/root/ssi')/1024;  
echo number_format($size,2).' KB'; //输出 24.02KB
```

### **delCacheFile** 方法

作用：删除缓存目录下的文件或子目录文件，但不会删除默认 index.html 文件

语法：bool delCacheFile(\$dir)

参数：

\$dir 目录名或文件名

示例：

```
//删除 cache/fields 目录除 index.html 外的文件  
delCacheFile('fields');
```

### **readFileList** 方法

作用：获取文件列表(所有子目录文件)

语法：array readFileList(\$path,&\$file\_list,\$ignore\_dir=array())

参数：

\$path 目录

\$file\_list 存放所有子文件的数组

\$ignore\_dir 需要忽略的目录或文件

示例：

```
$file_list = array();

//罗列出商城 cache 目录内的所有文件全路径 ,adv、session 和 index 目录以及~Bizpower CRM.php
文件除外

readFileList(BasePath.'/cache/', $file_list, array('adv', 'session', 'index', '~Bizpower CRM.php'));
print_r($file_list);

//输出
Array
(
    [0] => D:/root/Bizpower CRM/trunk/cache/adv_change/index.html
    [1] => D:/root/Bizpower CRM/trunk/cache/area/index.html
    [2] => D:/root/Bizpower CRM/trunk/cache/brand/index.html
    [3] => D:/root/Bizpower CRM/trunk/cache/category/index.html
    [4] => D:/root/Bizpower CRM/trunk/cache/class_tag.php
    [5] => D:/root/Bizpower CRM/trunk/cache/fields/_pk.php
    [6] => D:/root/Bizpower CRM/trunk/cache/goods_class.php
    [7] => D:/root/Bizpower CRM/trunk/cache/link.php
    [8] => D:/root/Bizpower CRM/trunk/cache/index.html
)
```

### **ncPriceFormat** 方法

作用：价格格式化，返回两位小数位数的价格

语法：string ncPriceFormat(\$price)

参数：

\$price 待处理的价格

示例：

```
echo ncPriceFormat(100.256); // 输出 100.26
echo ncPriceFormat(100.2); // 输出 100.20
```

### **ncUrl** 方法

作用：组成 url 地址

语法：string ncUrl(\$param = array(), \$type = '', \$domain = '')

参数： \$param 参数

\$type url 类型



\$domain 二级域名

### **ncDomainUrl** 方法

作用：获取二级域名 url 地址

语法：ncDomainUrl(\$type,\$domain\_url,\$domain)

参数： \$type url 类型

\$domain\_url 原始地址

\$domain 二级域名

### **subdomain** 方法

作用：二级域名解析

语法：int subdomain()

### **ncReplaceText** 方法

作用：通知邮件/通知消息 内容转换函数

语法：string ncReplaceText(\$message,\$param)

参数： \$message 内容模板

\$param 内容参数数组

### **str\_cut** 方法

作用：字符串切割函数，一个字母算一个位置,一个字算 2 个位置

语法：str\_cut(\$string, \$length, \$dot = '')

参数： \$string 待切割的字符串

\$length 切割长度

\$dot 尾缀

### **request\_uri** 方法

作用：重写 \$\_SERVER['REQUEST\_URI'] 方法

### **get\_image\_type** 方法

作用：获取图片类型

语法：string get\_image\_type(\$str)

### **C** 方法

作用：取得系统配置信息

语法：C(\$key)

参数： string \$key 取得下标值

示例：

C('site\_url') 取得 \$config['site\_url'] 值

C('cache.type') 取得 \$config['cache']['type'] 值

### **defaultGoodsImage** 方法

作用：取得商品默认大小图片

语法： defaultGoodsImage(\$key)

参数： \$key 图片大小 small/tiny

### **import** 方法

作用 加载文件 ,只适用于加载框架内类库文件 如果文件名中包含"\_"使用"#"代替 默认加载 libraties

目录内的类库

语法： import(\$libname,\$file\_ext='.php')

参数： \$libname 要加载的文件

\$file\_ext 文件扩展名

示例：

```
import('cache');  
//相当于 require_once(BasePath.'/framework/libraries/cache.php');  
import('libraries.cache');  
//相当于 require_once(BasePath.'/framework/libraries/cache.php');  
import('function.core');  
//相当于 require_once(BasePath.'/framework/function/core.php');
```

### **random** 方法

作用：取得随机数

语法： random(\$length, \$numeric = 0)

参数： \$length 生成随机数的长度

\$numeric 是否只产生数字随机数 1 是 0 否

### **template** 方法

作用：返回模板文件所在完整目录

语法： template(\$tplpath,\$project='')

参数： \$tplpath 模板文件名（不含扩展名）

\$project 项目名称

### **chksubmit** 方法

作用：检测 FORM 表单是否是合法提交

语法： chksubmit()

### **check\_repeat** 方法

作用：检测重复提交

语法：boolean check\_repeat(\$key,\$ttl = 30)

参数： \$key 检测名称

\$ttl 提交间隔时间（秒）

### **log\_times** 方法

作用：记录操作次数

语法：int log\_times(\$key, \$op = 'cookie', \$expire='')

参数： \$key 记录操作的名称

\$op 数据存储类型，目前只支持 cookie

### **lazypage** 方法

作用：延时加载分页功能，判断是否有更多连接和 limitstart 值和经过验证修改的\$delay\_eachnum 值

语法：array lazypage(\$delay\_eachnum,\$delay\_page,

\$count,\$ispage=false,\$page\_nowpage=1,\$page\_eachnum=1,\$page\_limitstart=1)

参数： \$delay\_eachnum 延时分页每页显示的条数

\$delay\_page 延时分页当前页数

\$count 总记录数

\$ispage 是否在分页模式中实现延时分页(前台显示的两种不同效果)

\$page\_nowpage 分页当前页数

\$page\_eachnum 分页每页显示条数

\$page\_limitstart 分页初始 limit 值

### **F** 方法

作用：文件数据读取和保存 字符串、数组

语法：F(\$name, \$value="", \$path = 'cache',\$ext = '.php')

参数： \$name 文件名称（不含扩展名）

\$value 待写入文件的内容

\$path 写入 cache 的目录

\$ext 文件扩展名

示例：

**F('setting');** //取得 setting 缓存

**F('setting',\$data);** //将\$data 数组生成到 setting 文件缓存

### **write\_file** 方法

作用：写入文件操作

语法：boolean write\_file(\$filepath, \$data, \$mode = null)

参数： \$filepath 待写入内容的文件路径

\$data 待写入的内容

\$mode 写入模式，如果是追加，可传入 "append"

### **mk\_dir** 方法

作用：循环创建目录

语法：boolean mk\_dir(\$dir, \$mode = '0777')

参数： \$dir 待创建的目录

\$mode 权限

### **pagecmd** 方法

作用：封装分页操作的函数，方便调用

语法：pagecmd(\$cmd = "", \$arg = "")

参数： \$cmd 命令类型

\$arg 参数

\$cmd 命令类型允许的值如下：

seteachnum 设置每页数量

settotalnum 设置记录总数

setstyle 设置分页样式

show 返回分页链接

obj 返回分页对象本身

gettotalnum 取得记录总数

gettotalpage 取得总页数

### **throw\_exception** 方法

作用：抛出异常

语法：throw\_exception(\$error)

参数： \$error 异常信息

### **halt** 方法

作用：输出错误信息

语法：halt(\$error)

参数： \$error 错误信息

### **compress\_code** 方法

作用：去除代码中的空白和注释

语法：compress\_code(\$content)

参数：\$content 待压缩的内容

### **H** 方法

作用：读/写 缓存方法

语法：H(\$key, \$value="", \$cache\_type="", \$expire=null, \$args=null)

参数： \$key 缓存名称

\$value 缓存内容

\$type 缓存类型，允许值为 file, memcache, xcache, apc, eaccelerator，可以为空，默认为 file 缓存

\$expire 缓存周期

\$args 扩展参数

示例：

```
H('setting'); // 取得缓存
```

```
H('setting', true); // 生成 setting 缓存并返回缓存结果
```

```
H('setting', null); // 清空 setting 缓存
```

```
H('setting', true, 'file'); // 生成 setting 文件缓存
```

```
H('setting', true, 'memcache'); // 生成 setting 缓存到 memcache
```

### **rcache** 方法

作用：读取缓存信息（只适用于内存缓存）

语法：rcache(\$key = null, \$prefix = "", \$unserialize = true)

参数： \$key 要取得缓存 键

\$prefix 键值前缀

\$unserialize 是否需要反序列化

### **wcache** 方法

作用：写入缓存（只适用于内存缓存）

语法：boolean wcache(\$key = null, \$data = array(), \$prefix = "", \$ttl = 0, \$prefix = "", \$serialize = true)

参数： \$key 缓存键值

\$data 缓存数据

\$ttl 缓存周期

\$perfix 存入的键值前缀

\$serialize 是否序列化后保存

**rec** 方法

作用：调用推荐位

语法：string rec(\$rec\_id = null)

参数：\$rec\_id 推荐位 ID

**L** 方法

作用：快速调用语言包

语法：string L(\$key = "")

参数：\$rec\_id 推荐位 ID

## 4.2.2 验证码函数

**makeSeccode** 方法

作用：产生验证码

语法：string makeSeccode(\$nchash)

参数：\$nchash 哈希数

**checkSeccode** 方法

作用：验证验证码

语法：boolean checkSeccode(\$nchash,\$value)

参数： \$nchash 哈希数

\$value 待验证值

## 4.2.3 框架压缩函数

**build** 方法

作用：压缩框架文件

语法：build()

**compile** 方法

作用：过滤掉不需要压缩的内容

语法：string compile(\$filename)

参数：\$filename 待压缩文件

#### **compress\_code** 方法

作用：压缩 PHP 代码

语法：string compress\_code(\$content)

参数：\$content 压缩内容

## 4.3 主要类库参考

### 4.3.1 Model 类

#### **tableInfo** 方法

作用：生成表结构信息

语法：tableInfo(\$table)

参数：string \$table 待生成的表名

#### **table** 方法

作用：设置表名

语法：object table(\$table)

参数：string \$table 待设置的表名

#### **order** 方法

作用：设置排序规则

语法：object order(\$order)

参数：string \$order 待排序内容

#### **where** 方法

作用：设置 sql 条件

语法：object where(\$condition)

参数：string/array \$condition 待执行的条件

示例：where(array('member\_id'=>5)) 或 where('member\_id=5')

#### **on** 方法

作用：设置关联表

语法：object on(\$string)

参数：string \$string 待执行的条件

示例：on('store.member\_id=member.member\_id')

#### **join** 方法

作用：设置关联表，一般与 on 方法一起使用

语法：object join(\$string)

参数：string \$string 关联类型

示例：join('inner')

### **limit** 方法

作用：设置 limit 规则

语法：object limit(\$string)

参数：string \$string 待排序内容

示例：limit(3), limit('3,6')

### **group** 方法

作用：设置 group by 规则

语法：object limit(\$field)

参数：string \$field 待分组字段

示例：group('store\_id')

### **having** 方法

作用：设置 having 规则

语法：object having(\$condition)

参数：string \$condition 待筛选条件

示例：having('nc\_count>3')

### **distinct** 方法

作用：设置 distinct，去重复数据

语法：object distinct()

参数：boolean true/false

### **page** 方法

作用：设置每页显示数量

语法：object page(\$pagesize)

参数：integer \$pagesize

### **min** 方法

作用：取最小值

语法：integer min(\$field)

参数：string \$field 字段名



示例：`$model->table('member')->min('member_id')`

**max** 方法

作用：取最大值，用法同 min

**count** 方法

作用：取得记录总数

语法：`integer count()`

示例：`$model->where('member_id>23')->count();`

**sum** 方法

作用：取字段总和

语法：`integer sum($field)`

参数：`string $field` 字段名

示例：`$model->table('product')->sum('price')`

**avg** 方法

作用：取平均值

语法：`integer avg($field)`

参数：`string $field` 字段名

示例：`$model->table('product')->avg('price')`

**select** 方法：执行查询，详见 3.3 模型

**find** 方法：查询一条信息，详见 3.3 模型

**delete** 方法：执行删除，详见 3.3 模型

**update** 方法：执行更新，详见 3.3 模型

**insert** 方法：插入数据，详见 3.3 模型

**insertAll** 方法：批量插入，详见 3.3 模型

**execute** 方法：执行 sql，主要为增、删、改操作，详见 3.3 模型

**query** 方法：执行 sql，主要为查询操作，详见 3.3 模型

**clear** 方法

作用：清空表内容

语法：`clear()`

示例：`$model->table('product')->clear()`

**getLastID** 方法

作用：取得最新插入 ID

**getFields** 方法

作用：取得当前操作表结构信息

#### **setInc** 方法

作用：自增操作

语法：setInc(\$field,\$num)

参数：string \$field 字段名

Int \$num 增加数值

示例：\$model->where(array('link\_id'=>2))->setInc('link\_sort',3)

#### **setDec** 方法

作用：自增操作

语法：setDec(\$field,\$num)

参数：string \$field 字段名

Int \$num 增加数值

示例：\$model->where(array('link\_id'=>2))->setDec('link\_sort',3)

#### **cls** 方法

作用：清空模型对象中的表名、where 条件、排序等参数

语法：object cls()

示例：\$model->cls()

#### **showpage** 方法

作用：显示分页链接

语法：string showpage(\$style)

参数： string \$style 分页样式，可选值为 1，2（默认），3

示例：\$model->showpage();

### 4.3.2 ModelDb 类

select()：查询操作

buildSelectSql()：拼成 select 语句

parseValue()：格式化待保存到数据表的数据

parseTable()：处理表名

parseWhere()：组合 where 条件

parseWhereItem()：分析 where 条件

parseLimit()：处理 limit 条件

parseJoin() : 处理 join 条件  
delete() : 组合成 delete 语句  
update() : 组合成 update 语句  
parsePriority() : 设置 SQL 执行时, 延时、缓存等高级操作  
clear() : 清空表  
insert() : 插入操作  
getLastId() : 取得最后生成的主键 ID  
insertAll() : 批量插入数据

### 4.3.3 Tpl 类

getInstance() : 实例化类  
setDir() : 设置模板目录  
setLayout() : 设置布局  
output() : 抛出变量  
showpage() : 显示模板  
showTrace() : 显示 trace 信息

Tpl 类使用示例 :

```
//设置模板文件夹路径
Tpl::setDir('home');
//设置布局文件
Tpl::setLayout('home_layout');
//向模板抛出内容, 热门搜索
Tpl::output('hot_search','电脑,鼠标');
//显示模板
Tpl::showpage('member_profile.avatar');
```

### 4.3.4 Language 类

getGBK () : 得到数组变量的 GBK 编码  
getUTF8() : 得到数组变量的 UTF-8 编码  
get() : 取指定下标的数组内容

set() : 设置指定下标的数组内容

read() : 通过语言包文件设置语言内容

getLangContent() : 取语言包全部内容

appendLanguage() : 向语言包追加内容

Language 类使用示例 :

```
//读取语言包文件
```

```
Language::read('home_index');
```

```
//取得单个语言包
```

```
Language::get('pc_title')
```

```
//语言包转码
```

```
if (strtoupper(CHARSET) == 'GBK'){
```

```
$array = Language::getGBK($array);
```

```
}
```

### 4.3.5 Page 类

get () : 取得属性

set() : 设置属性

setPageName() : 设置 url 页码参数名

setNowPage() : 设置当前页码

setEachNum() : 设置每页数量

setStyle() : 设置输出样式

setTotalNum() : 设置信息总数

getNowPage() : 取当前页码

getTotalPage() : 取页码总数

getTotalNum() : 取信息总数

getEachNum() : 取每页信息数量

getLimitStart() : 取数据库 select 开始值

getLimitEnd() : 取数据库 select 结束值

setTotalPage() : 设置页码总数

show() : 输出页码链接

### 4.3.6 Security 类

getToken() : 取得令牌内容

checkToken() : 判断令牌是否正确

filterXss() : xss 过滤

filterHtmlSpecialChars() : 将字符 & " < > 替换

getAddslashesForInput() : 参数过滤

### 4.3.7 UploadFile 类

set() : 设置属性

get() : 读取属性

upload() : 上传操作

fileInputError() : 获取上传文件的错误信息

setPath() : 设置存储路径

setFileName() : 设置文件名称 不包括 文件路径

setError() : 设置错误信息

getSysSetPath() : 根据系统设置返回商品图片保存路径

UploadFile 类使用示例 :

```
//创建上传类
$upload = new UploadFile();

//删除原图
$upload->set('ifremove', true);

//设置上传目录
$upload->set('default_dir', 'upload/'.$upload->getSysSetPath());

//文件最大允许 1M
$upload->set('max_size', 1024*1024);

//生成两张缩略图，宽高分别为 30,300
$upload->set('thumb_width', '30,300');
$upload->set('thumb_height', '30,300');

//两个缩略图名称后面分别追加 "_tiny", "_mid"
$upload->set('thumb_ext', '_tiny_mid');
```

```
//开始上传
$result = $upload->upfile($v);
if($result){
//得到图片上传后的路径
$img_path = $upload->getSysSetPath().$upload->file_name;
}
```

### 4.3.8 Validate 类

Validate() : 验证数组中的值

Check() : 正则表达式运算

setValidate() : 需要验证的内容

getError() : 得到验证的错误信息

Validate 类使用示例 :

```
//创建验证类
$obj_validate = new Validate();
$obj_validate->validateparam = array(
array("input"=>$_POST["username"],"require"=>"true","message"=>'用户名不能为空'),
array("input"=>$_POST["password"],"require"=>"true","message"=>'密码不能为空'),
array("input"=>$_POST["email"], "require"=>"true","validator"=>"email","message"=>' 请
正确填入 Email'),
);
$error = $obj_validate->validate();
if ($error != ""){
//输出错误信息
showDialog($error);
}
```

### 4.3.9 ResizeImage 类

newImg() : 构造函数

init\_img() : 初始化图象

dst\_img() : 图象目标地址

### 4.3.10 Cache 类

connect() : 实例化缓存驱动

getInstance() : 取得实例

set() : 保存缓存

get() : 读取缓存

rm() : 删除缓存

inc() : 递增操作

dec() : 递减操作

Cache 类使用示例 :

```
//创建缓存对象 ( 这里以 memcache 为例 )
$obj_cache = Cache::getInstance('memcache');
//写入缓存, 缓存周期为 1 小时
$obj_cache->set('uid_100','这里是缓存内容', null, 3600);
//读取缓存内容
$value = $obj_cache->get('uid_100');
//删除缓存内容
$obj_cache->rm('uid_100');
//清空所有缓存内容
$obj_cache->clear();
//写入 100
$obj_cache->set('uid_100',100, null, 3600);
//增加 50
$obj_cache->inc('uid_100',50);
echo $obj_cache->get('uid_100');//输出 150
//减少 50
$obj_cache->dec('uid_100',50);
echo $obj_cache->get('uid_100');//输出 100
```

## 5. 结语

感谢选择 Bizpower , 可以登录 Bizpower 官网 <http://www.bizpower.com> 下载最新的 Bizpower 商城程序, 官方论坛 <http://bbs.bizpower.com> 可以帮助大家答疑解惑。