# pp空间技术文档 by Phoenix

目录

[pp空间技术文档 by Phoenix 1](#_Toc500530815)

[命名规范 2](#_Toc500530816)

[1． 数据库 2](#_Toc500530817)

[2. Asp.net 前台 2](#_Toc500530818)

[1. Asp.net后台 2](#_Toc500530819)

[4.类 2](#_Toc500530820)

[5. 注意事项 2](#_Toc500530821)

[类简介 3](#_Toc500530822)

[1. Safe 3](#_Toc500530823)

[2. Sql 3](#_Toc500530824)

[3. User 3](#_Toc500530825)

[后台重要代码思路简介 3](#_Toc500530826)

[1. 访问他人空间 3](#_Toc500530827)

[2. 自动登入 3](#_Toc500530828)

[3. 联表查询 4](#_Toc500530829)

[数据库字段解释 5](#_Toc500530830)

[1. album 用于储存用户相册的图片内容 5](#_Toc500530831)

[2. applyfriends 用于储存申请者信息 5](#_Toc500530832)

[3. Comments 动态评论（passage） 5](#_Toc500530833)

[4. diary 日志 6](#_Toc500530834)

[5. diarycomment 日志评论 6](#_Toc500530835)

[6. friend 好友列表 6](#_Toc500530836)

[7. message 留言 7](#_Toc500530837)

[8. messagecomment 留言回复 7](#_Toc500530838)

[9. passages 动态文章 7](#_Toc500530839)

[10. users 用户信息表 8](#_Toc500530840)

[注意事项 9](#_Toc500530841)

[1. Ueditor 9](#_Toc500530842)

[2.数据库 9](#_Toc500530843)

[Web.config 9](#_Toc500530844)

[后台代码 10](#_Toc500530845)

## 命名规范

### 数据库

#### 表名：小写，并且不超过三个单词，不产生歧义

#### 列名：驼峰命名法，并且开头单词首字母大写

#### 特殊字段类型

id列：出现id的都以小写，并为int型

Time列：以datetime为数据类型，用于存储上传时间

### Asp.net 前台

页面命名：全部小写，不超过三个单词

控件命名：开头字母大写的驼峰命名

### Asp.net后台

变量命名：开头字母大写的驼峰命名法

Session传值命名：开头字母大写的驼峰命名法

函数命名：开头字母大写的驼峰命名法

### 4.类

1. 类名：开头大写，单个单词构成

2. 方法：开头为动词首字母小写，否则首字母大写

### 5. 注意事项

1. 由于开发过程中的频繁改动，有部分命名并不规范，若发现可进行纠正

2. 为了页面间方便调整，部分页面中某些相同功能的控件具有相同名称

3. 某些函数名字为了防止命名冲突使用特殊的命名规则

4. 重要变量

1. AccountName：用来储存当前登入者的名字

2. Friend：用来储存显示当前页面所展示的用户信息的名字

## 类简介

### Safe

安全检测，加密解密及字符替换类2

### Sql

基础sql语句的封装，直接对数据库进行操作的位置

### User

各类对用户的操作，包括各个页面对用户信息的引用修改或添加

每个页面所使用的方法由注释分开并注明页面名称

**具体方法在注释中有明确说明功能**

## 后台重要代码思路简介

### 访问他人空间

通过判别Friend与AccountName是否一致实现

AccountName为当前登入者，Friend为被访问者（可以为AccountName自己）

若一致则不显示CloseSomedivForOwn中的内容（如返回自己主页按钮）；

如不一致则不显示CloseSomedivForVisit中的内容（如新增日志）

以此来实现对他人空间访问时无法修改，可在函数中添加新的内容

两个函数最终都在Test\_username函数中调用（该函数用于检测是否登入，由于各个页面都有，因而设计为该名字防止命名冲突

### 自动登入

在login页面建立了两个cookie值并且分别赋值

cookie变量是存储加密后的用户名，而cookieAccountName则是存储用户名

cookie的加密方式是以当前日期的MD5加密后的值的前八位作为密匙通过DES加密方式将用户名进行加密

重新点击登入页面是若反向解密后用户名与解密用户名一致则登入

由于以时间为密匙，因此该cookie值只能维持一天登入状态，可以根据后期用户需求更改密匙设定

### 联表查询

在有评论的地方为了将某一用户的备注昵称及由账户名一一查出

在User类中的查询函数使用了连表查询的sql语句

如 getCommet， 方法等

### 模板页显示用户姓名

在母版页的cs代码中写入了一个Name函数

母版页的a标签内写入<%=Name()%>

使用函数而不直接使用session传到界面上的原因是因为直接传可能会出现无该变量（用户直接从非login的界面登入）而导致的错误，用try catch避免了

### 相册封面设置与删除

处理方法封装与User类中的updateAlbumCover 和deleteImage 函数中

封面可设置为该相册中任意图片，设置后会在数据库中对该图片进行标记

一旦删除标记图片，相册封面会变回默认封面

默认封面路径 /album/Cover.jpg

创建相册时也是利用默认封面

笔者曾经尝试过通过查询获得该相册的封面路径然后在该相册中的图片中寻找是否有路径重合来判断是否删除了作为封面的图片，但这种方法比起直接标记一旦数据量增加会大幅降低速度，因此笔者认为通过牺牲一部分储存（bit大小）来提升速度是可取的

## 数据库字段解释

### album 用于储存用户相册的图片内容

#### id 唯一标识

#### ImagePath 图片路径

#### ImageName 图片名称

#### Time 上传时间

#### Albumid 所属相册的id

#### IsCoverOrNot 是否为封面的标记，若为是，为1，否则为null

### albumlist 相册列表（所有相册的信息）

#### id 唯一标识

#### AccountName 相册拥有者账户

#### AlbumName 相册名称

#### CoverPath 封面图片路径

### applyfriends 用于储存申请者信息

#### AccountName 收到申请的账户

#### ApplyFriend 申请者

### Comments 动态评论（passage）

#### id 唯一标识

#### Comment 评论

#### CommentUser 评论者账户

#### Pageid 所属的文章的id

#### Time 写入时间

### diary 日志

#### id 唯一标识

#### AccountName 所属账户名称

#### DiaryTitle 日志名称

#### Diary 日志内容

#### 6.Time 写入时间

### diarycomment 日志评论

#### id 唯一标示

#### Diaryid 所属日志id

#### Comment 评论

#### CommentUser 评论者

#### Time 写入时间

### friend 好友列表

#### AccountName 账户名

#### Friend 朋友账户名

#### Remark 对该朋友的备注

### message 留言

#### id 唯一标识

#### AccountName 留言所属账户名

#### Message 留言内容

#### MessageUser 留言者

#### Time 留言时间

### messagecomment 留言回复

#### id 唯一标识

#### MessageComment 留言评论内容

#### MessageCommentUser 留言评论者

#### Messageid 所属留言的id

#### Time 写入时间

### passages 动态文章

#### AccountName 所属用户名

#### Passage 文章内容

#### Title 文章标题

#### Time 写入时间

### users 用户信息表

#### AccountName 账户名

#### AccountPwd 账户密码（MD5加密后的结果）

#### UserNickName 账户昵称

#### Question 账户密保问题

#### Answer 账户密保问题答案

#### Age 年龄

#### Sex 性别

#### BornYear 出生年份

#### BornMonth 出生月份

#### BornDay 出生日

#### Profession 职业

#### ProfileImage 头像图片路径

## 注意事项

### Ueditor

#### Ueditor中controller.ashx 被排除出系统

#### Newston.json 包已安装

### 2.数据库

#### 1. 数据库中有些表间的关系未建立连接，否则可能出现循环引用等问题，因而无法建立 如meesage中的MessageComment 与Messageid

#### 2. 除了id以外大部分数据字段用nvarchar的原因是因为sql的参数化查询封装后对查询内容设定为nvarchar，这样使得封装的函数更具有普遍使用价值

### Web.config

#### 修改为<httpRuntime targetFramework="4.6.1" requestValidationMode="2.0" maxRequestLength="102400"/>

#### 其中requestValidationMode="2.0"为解除对<> 等符号的拦截，防止网站停止工作

#### maxRequestLength="102400"指最大上传大小，为100M（图片上传后台设定为10M）

### 后台代码

#### Sql类中多已经写入try catch 错误统一返回为数据库连接失败