在PHP中实现单链表

# 开发环境

1、环境搭建：Windows 7+Apache 2.4.18+MySQL 5.7.11+PHP 7.1.0。

2、文本编辑器：Sublime 3。

# 二、主要技术

本案例主要在PHP中通过面向对象的程序设计思想来实现单链表的数据结构算法。

单链表是一种链式存取的数据结构，用一组地址任意的存储单元存放线性表中的数据元素。

单链表是最简单的链表表示。用它来表示线性表时，每一个数据元素占用一个结点(node)。一个结点一般由两个域组成，一个域存放数据元素data；另一个域存放一个指向链表中下一个结点的指针link，它指出下一个结点的开始存储地址。而最后一个结点的指针为空。单链表中数据元素之间的逻辑关系是由结点中的指针指示的，换句话说，指针为数据元素之间的逻辑关系的映像，则逻辑上相邻的两个元素其存储的物理位置不要求紧邻，因此，这种存储结构为非顺序映像或链式映像。

# 三、效果图展示

实现效果图，如图1所示。

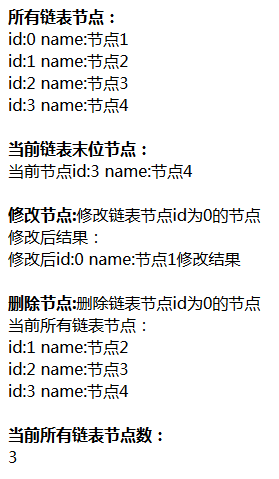


图1 效果图（单链表操作结果）

# 四、具体步骤

## 4.1、创建PHP文件

创建一个名为“index.php”的PHP文件。

## 4.2、创建链表节点类

// 链表节点

class node {

public static $count = -1; //节点id

public $name; //节点名称

public $next; //下一节点

public $id;

// 构造函数

public function \_\_construct($name) {

$this->id = self::$count;

$this->name = $name;

$this->next = null;

self::$count += 1;

}

}

## 4.3、创建单链表类

### （1）声明单链表对象

class singelLinkList {

private $header; // 头结点

private $current; // 当前节点

public $count; // 节点数

...

}

### （2）创建构造函数

// 构造方法

public function \_\_construct($data = null) {

$this->header = new node ($data);

$this->current = $this->header;

$this->count = 0;

}

### （3）添加节点数据

public function addLink($node) {

if($this->current->next != null)

$this->current = $this->current->next;

$this->count++;

$node->next = $this->current->next;

$this->current->next = $node;

}

### （4）删除链表节点

public function delLink($id) {

$current = $this->header;

$flag = false;

while ( $current->next != null ) {

if ($current->next->id == $id) {

$flag = true;

break;

}

$current = $current->next;

}

// 判断是否删除成功

if ($flag) {

$this->count--;

$current->next = $current->next->next;

} else {

echo "未找到id=" . $id . "的节点！<br>";

}

}

### （5）获取所有链表节点

public function getLinkList() {

$this->checkNull(); // 判断链表是否为空

$current = $this->header;

while ( $current->next != null ) {

echo 'id:' . $current->next->id . ' name:' .

$current->next->name . "<br>";

if ($current->next->next == null) {

break;

}

$current = $current->next;

}

}

### （6）获取链表长度

public function getLinkLength()

{

echo $this->count;

}

### （7）获取当前节点

public function getCurrent()

{

$this->checkNull();

echo '当前节点id:' . $this->current->next->id . '

name:' . $this->current->next->name . "<br>";

}

### （8）判断链表是否为空

public function checkNull()

{

if ($this->header->next == null) {

echo "链表为空!";

exit;

}

}

### （9）获取节点名字

public function getLinkById($id) {

$this->checkNull();

$current = $this->header;

while ( $current->next != null ) {

if ($current->id == $id) {

break;

}

$current = $current->next;

}

echo '修改后id:' . $current->id . ' name:' .

$current->name . "<br>";

}

### （10）更新节点名称

public function updateLink($id, $name) {

$this->checkNull();

$current = $this->header;

while ( $current->next != null ) {

if ($current->id == $id) {

break;

}

$current = $current->next;

}

return $current->name = $name;

}

## 4.4、创建并操作单链表

### （1）初始化单链表对象

$list = new singelLinkList ();

### （2）添加节点数据

$list->addLink ( new node ('节点1' ) );

$list->addLink ( new node ('节点2' ) );

$list->addLink ( new node ('节点3' ) );

$list->addLink ( new node ('节点4' ) );

### （3）输出所有链表节点

echo "<b>所有链表节点：</b><br/>";

$list->getLinkList();

echo "<br/>";

### （4）输出当前链表末位节点

echo "<b>当前链表末位节点：</b><br/>";

$list->getCurrent();

echo "<br/>";

### （5）修改链表节点

echo "<b>修改节点:</b>修改链表节点id为0的节点<br/>";

$list->updateLink(0, '节点1修改结果');

echo "修改后结果：<br/>";

$list->getLinkById(0); // 输出查找结果

echo "<br/>";

### （6）删除节点

echo "<b>删除节点:</b>删除链表节点id为0的节点<br/>";

$list->delLink(0);

### （7）输出当前所有链表节点

echo "当前所有链表节点：<br/>";

$list->getLinkList();

echo "<br/>";

### （8）输出当前链表节点数

echo "<b>当前所有链表节点数：</b><br/>";

$list->getLinkLength ();

# 五、网页测试

开启Apache服务器后，打开浏览器，在浏览器地址栏中输入文件地址，例如：“localhost:8888/PHParticle/071/Codes/”，可以看到程序运行结果如图2所示。

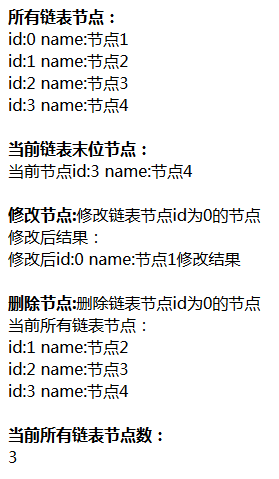


图2 案例结果图（单链表操作结果）