**TP开发文档**

作者：ZLQ 日期：2017/5/16

一、前台多语言切换

1. 在app\_init行为处加载CheckLang判断是否存在语言lang缓存

在应用程序tags.php文件中添加如下代码

'app\_init' => ['app\\common\\behavior\\CheckLang'],

1. 通过是否存在lang变量，判断用户是否点击语言变换按钮，在应用程序中新建common/behavior目录并新建CheckLang类，首先判断入口文件，如果是index.php或者域名进入，判断是否存在lang变量并正确跳转到相应模块

namespace app\common\behavior;

use think\Request;

class CheckLang {

function run(){

$request= Request::instance();

if (strpos( $request->baseUrl(),'index.php') !== false || $request->baseUrl() == '/') {

//如果用户通过index.php/admin/index/index来访问网站，因为没有lang变量，所以不跳转，另验//证码不能正常显示，故如法登录后台，只能通过admin.php登陆后台

switch ($request->get('lang')) {

case 'zh-cn':

header('location:/index.php/index/');

exit;

break;

case 'en-us':

header('location:/index.php/en/');

exit;

break;

default:

header('location:/index.php/index/');

break;

}

}

}

}

1. 通过获取当前模块名来判断所需加载的语言包，如果是index模块则加载中文语言包zh-cn.php文件，如果是en模块则加载英文语言包en-us.php文件；

namespace app\common\behavior;

use think\Config;

use think\Request;

use think\Lang;

class LoadLang{

function run(){

$request= Request::instance();

switch ($request->module()) {

case 'index':

Lang::load(APP\_PATH . 'common\lang\zh-cn.php');

break;

case 'en':

Lang::load(APP\_PATH . 'common\lang\en-us.php');

break;

default:

break;

}

}

}

1. 在应用common/behavior文件下新建LoadLang行为类，并在应用tags.php文件中设置模块加载时调用。

'module\_init' => ['app\\common\\behavior\\LoadLang'],

1. 在网站或者模块首页设置语言变换链接

<div>

<a href='index.php?lang=zh-cn'>简体中文</a>

<a href="index.php?lang=en-us">English</a>

</div>

二、后台多语言切换

注：因整个后台均为一个模块，所有语言切换均发生在模块内，故不能在模块开始时加载语言包，只能通过控制器进行加载，初步设想为，建立一个公共基类，所有控制器均继承该基类。基类的初始化函数判断是否存在input(‘lang’)变量，若存在，则加载相应的语言包文件，并设置语言包cookie（即cookie(‘lang’,$val)）；若不存在则判断语言包是否存在，若存在则加载相应的语言包，否在加载默认语言包；

1. 建立公共基类，并设置\_initliaze()函数

namespace app\admin\controller;

use think\Controller;

use think\Lang;

use think\Cookie;

use think\Request;

use think\Config;

class Base extends Controller

{

function \_initialize(){

$lang = '';

$default\_lang = Config::get('default\_lang');

if (input('lang')) {

$lang = input('lang');

} elseif (cookie('lang')) {

$lang = cookie('lang');

} else {

$lang = $default\_lang;

}

switch ($lang) {

case 'zh-cn':

Lang::load(ADMIN\_PATH . 'lang/zh-cn.php');

cookie('lang', 'zh-cn');

break;

case 'en-us':

Lang::load(ADMIN\_PATH . 'lang/en-us.php');

cookie('lang', 'en-us');

break;

default:

Lang::load(ADMIN\_PATH . 'lang/' . $default\_lang . '.php');

cookie('lang', $default\_lang);

break;

}

}

1. 控制器继承基类，并在模板设置选择语言内容

class Index extends Base

{

public function index()

{

return $this->fetch();

}

}

三、后台多语言网站信息配置（如：配置网站信息）

1、建立setting表,用字段language\_id分别存储中英文网站信息（1为中文，2为英文）

CREATE TABLE `mlcms\_setting` (

`id` int(11) NOT NULL COMMENT '网站ID',

`site\_name` varchar(100) NOT NULL COMMENT '网站名称',

`site\_domain` varchar(50) NOT NULL COMMENT '网站域名',

`site\_switch` tinyint(4) NOT NULL DEFAULT '1' COMMENT '网站开启',

`site\_keywords` varchar(150) NOT NULL COMMENT '网站关键词',

`site\_desciption` varchar(150) NOT NULL COMMENT '网站描述',

`site\_templet` varchar(100) NOT NULL DEFAULT 'default' COMMENT '网站模板',

`upload\_size` int(11) NOT NULL DEFAULT '2048' COMMENT '上传文件大小',

`upload\_type` varchar(150) NOT NULL COMMENT '上传文件类型',

`GD\_test` tinyint(4) NOT NULL COMMENT 'GD库是否可用',

`GD\_test\_switch` tinyint(4) NOT NULL DEFAULT '1' COMMENT '水印开关',

`watermart\_condition` varchar(150) NOT NULL COMMENT '水印条件',

`watermark\_image` varchar(100) NOT NULL COMMENT '水印图片',

`watermark\_transparency` smallint(6) NOT NULL DEFAULT '50' COMMENT '水印透明度',

`watermark\_position` tinyint(4) NOT NULL DEFAULT '9' COMMENT '水印位置',

`site\_type` varchar(20) NOT NULL COMMENT '网站类型：ch为中文网站；en为英文网站，language\_id=0时为中文，1时为英文',

`language\_id` tinyint(4) NOT NULL COMMENT '语言类型ID'

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='网站设置' ROW\_FORMAT=COMPACT;

2、在admin/view/common/left文件中新建两个链接均指向Setting控制器的edit操作，并传递language\_id参数以区别是编辑中文网站还是英文网站信息。

href="{:url('setting/edit',array('language\_id'=>'2'))}"

1. 在setting控制器的edit操作中接收传递过来的language\_id参数，如果是其它方式跳转过来的，判断是否有Session的language\_id值,如果有则接受，否则通过公共函数的get\_language\_id()函数获取，：（注意：通常在用户登陆时会检测系统默认语言并通过公共函数common.php的set\_language\_id()进行设置。必须用Session，不可以用Cookie因为Cookie无法在服务器端存储）

$language\_id= get\_language\_id();

switch ( $language\_id){

case '1':

$or\_data= db('setting')->where(array($language\_id =>'1'))->find();

break;

case '2':

$or\_data= db('setting')->where(array($language\_id =>'2'))->find();

break;

}

1. 如果是post方法提交的数据，则更新相应数据表，更新后跳转回原界面：

if (request()->isPost()) {

$data = array();

$data['site\_switch'] = 0;

$data['GD\_test'] = 0;

$data['GD\_test\_switch'] = 0;

foreach (input() as $k => $v) {

if ($k == 'site\_switch' || $k == 'GD\_test' || $k == 'GD\_test\_switch') {

$data[$k] = 1;

} else {

if ($v != '') {

$data[$k] = $v;

}

}

}

if (db('setting')->update($data)!==false) {

$this->success('操作成功', 'edit');

} else {

$this->error('操作失败', 'edit');

}

}四、语言切换时返回原操作界面

1、在admin模块的Base类的 \_initialize初始化函数中，利用Resquest类获取当前页面的控制器和操作名，并传递给模板：

$request= Request::instance();

$current\_action=$request->controller().'/'.$request->action();

$this->assign("current\_action", $current\_action);

1. 在admin模块的view/common/header.html文件中，修改中英文切换链接的url值：

{:url($current\_action,array('lang'=>'en-us'))}

五、建立多语言表（目前无效，方便后期扩展）

建立多语言表

CREATE TABLE `mlcms\_language` (

`id` int(11) NOT NULL COMMENT '语言ID',

`brief\_name` varchar(10) CHARACTER SET utf8 NOT NULL COMMENT '语言简写，用于在首页切换语言种类',

`detail\_name` varchar(20) CHARACTER SET utf8 NOT NULL COMMENT '语言全称',

`en\_name` varchar(20) NOT NULL COMMENT '英文名称',

`lang\_file\_name` varchar(10) CHARACTER SET utf8 NOT NULL COMMENT '语言文件名称'

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

六、栏目添加删除操作

1、建立栏目表

CREATE TABLE `mlcms\_category` (

`id` int(11) NOT NULL COMMENT '栏目ID',

`cate\_name` varchar(20) CHARACTER SET utf8 NOT NULL COMMENT '栏目名称',

`language\_id` int(11) NOT NULL DEFAULT '0' COMMENT '语言类型ID',

`pid` int(11) NOT NULL DEFAULT '0' COMMENT '上级栏目ID',

`sort` int(11) NOT NULL DEFAULT '0' COMMENT '排序',

`type` tinyint(4) NOT NULL DEFAULT '0' COMMENT '0表示文章列表，1表示图片列表'

`status` tinyint(4) NOT NULL DEFAULT '1' COMMENT '0表示栏目禁用，1表示栏目启用'

) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW\_FORMAT=COMPACT;

1. 在left.html中建立到list.html的连接，并传递参数language\_id;

在list函数中首先调用公共函数set\_language\_id()设置当前使用语言，其次通过get\_language\_id()获取当前语言，并根据语言的不同获取不同的栏目序列；

1. 在\_initialize() 函数中取到语言和目录变量，lang和cate，并进行排序，方便在显示时调用

$language\_id= get\_language\_id();

//添加栏目时，用于显示栏目所属语种

$lang= db('language')->select();

$lang\_arr='';

if($lang){

foreach ($lang as $key => $value) {

$lang\_arr[$value['id']]=$value;

}

}

unset($lang);

$this->assign('lang',$lang\_arr);

$cates= CategoryModel::name('category')->where(array('language\_id'=>$language\_id))->order('sort asc, id asc')->select();

//方便list.html中显示上级栏目名称

$cat\_arr='';

if($cates){

foreach ($cates as $key => $value) {

$cat\_arr[$value['id']]=$value;

}

}

$this->assign('cates',$cat\_arr);

//对娶到栏目进行分层列表

$sort\_cates=$this->sort($cates,0,0);

unset($cates);

$this->assign('sort\_cates',$sort\_cates);

1. 添加栏目时，需要对栏目按照父子关系及sort排序进行排序，在category中建立sort及paixu函数，方便添加栏目时选择上级栏目使用。

protected function sort($datas,$pid, $level){

static $arr='';

static $k=0;

foreach ($datas as $key => $value) {

if($value['pid']==$pid){

$arr[$k]=$value;

$arr[$k]['level']=$level;

$k++;

$this->sort($datas, $value['id'], $level+1);

}

}

return $arr;

}

1. 在公共函数库中建立set\_languange\_id(),get\_language\_id()两个函数，方便后期使用,
2. 对栏目进行编辑时，同4，
3. 对栏目删除时，需要判断是否有子栏目，若有子栏目则一并删除（需要提示用户）；

public function del(){

$id=input('id');

$language\_id= get\_language\_id();

$cates=db('category')->where(array('language\_id'=>$language\_id))->select();

$child\_arr= $this->getChilds($cates, $id);

$delete=TRUE;

foreach ($child\_arr as $key => $value) {

if(db('category')->delete($value)){

} else {

$delete=FALSE;

break;

};

}

if(db('category')->delete(array('id'=>$id))&&$delete)

{

$this->success(\think\Lang::get('operate\_success'), url('category/lst',array('language\_id'=>input('language\_id'))));

}else{

$this->success(\think\Lang::get('operate\_failure'), url('category/lst',array('language\_id'=>input('language\_id'))));

}

}

//获取子栏目函数

protected function getChilds($datas,$pid){

static $arr='';

static $k=0;

foreach ($datas as $key => $value) {

if($value['pid']==$pid){

$arr[$k]=$value['id'];

$k++;

$this->getChilds($datas, $value['id']);

}

}

return $arr;

}

1. 对栏目进行排序

function sort\_arr(){

$("form").attr("action","{:url('resort',array('language\_id'=>$language\_id))}");

$("#myform").submit();

}

public function resort(){

$sort\_arr= input();

$language\_id= get\_language\_id();

if(array\_key\_exists('sort',$sort\_arr)){

$datas='';

foreach ($sort\_arr['sort'] as $key => $value) {

static $k=0;

$datas[$k]['sort']=$value;

$datas[$k]['id']=$key;

$k++;

}

$is\_success=true;

foreach ($datas as $key => $value) {

if(db('category')->update($value)!==false){

$is\_success=true;;

}else{

$is\_success=FALSE;

break;;

}

}

if($is\_success){

$this->success(\think\Lang::get('operate\_success'), 'lst');

}else{

$this->error(\think\Lang::get('operate\_failure'), 'lst');

}

}

}

1. 对栏目删除，点击父栏目的checkbox则子栏目也要被选中
2. 从数据库中取出栏目数据
3. 对取出的数据进行排序

protected function sort\_tree($datas, $pid, $level, $f\_dataid) {

static $sort\_datas = array();

static $key = 0;

$data\_id = '';

//改的

foreach ($datas as $key => $data) {

$dataid = $f\_dataid;

if ($data['pid'] == $pid) {

$sort\_datas[$key] = $data;

$sort\_datas[$key]['level'] = $level;

if ($pid == 0) {

$dataid = $data['id'];

} else {

$dataid .= '-' . $data['id'];

}

$sort\_datas[$key]['dataid'] = $dataid;

$key++;

$this->sort\_tree($datas, $data['id'], $level + 1, $dataid);

}

}

/\* 原来程序，有点小问题

static $dataid='';

foreach ($datas as $key=> $data){

if($data['pid']==$pid){

$sort\_datas[$key]=$data;

$sort\_datas[$key]['level']=$level;

if($pid==0){

$dataid=$data['id'];

}else{

$dataid.='-'.$data['id'];

}

$sort\_datas[$key]['dataid']=$dataid;

$key++;

$this->sort\_tree($datas, $data['id'], $level+1);

}

}

\*

\*/

return $sort\_datas;

}

1. 建立ajax函数用于改变栏目状态，首先在list模板中引入按钮，并设置按钮的点击函数，由点击函数判断后提交到后台进行修改更新

<input class="checkbox-slider colored-success " style="left: inherit" id='status\_{$vo.id}' onclick="set\_status({$vo.id},this)" name='status' {if condition="$vo.status == 1"} checked="checked" {/if}type="checkbox">



function set\_status(id,obj){

var status=false;

if($('#'+obj.id).attr('checked')=='checked'){

status=true;

}

if(status){

alert('0');

$('#'+obj.id).removeAttr('checked');

set\_remote\_status(id,obj.name,0);

}else{

alert('1');

$('#'+obj.id).attr('checked','checked');

set\_remote\_status(id,obj.name,1);

}

}

function set\_remote\_status(id,field\_name,field\_value){

$.ajax({

url:"{:url('updateStatus')}",

data:{'id':id,'field\_name':field\_name,'field\_value':field\_value},

success:function(msg){

alert(msg);

},

error:function (XMLHttpRequest, textStatus, errorThrown) {

alert(errorThrown);

},

});

}

1. 在Category控制器中加入更新函数

public function updateStatus(){

$data['id']= input('id');

if(input('field\_name')&&(input('field\_value')!==false)){

$field\_name=input('field\_name');

$field\_value=input('field\_value');

$data[$field\_name]=$field\_value;

}else{

}

if(db('category')->update($data)!==false){

echo 'name'.input('field\_name').'value'.input('field\_value');

}else{

echo 'error';

}

}

其它栏目类似

七、权限认证