

集成 Ping++ SDK 与 CodeIgniter 框架

以下是一个完整的、逐步示例，展示如何在使用提供的 `BaseController` 类的基础上，将 **Pingpp** (Ping++) 支付处理 SDK 集成到 PHP 应用程序中，使用 CodeIgniter 框架。本指南假设你已经有一个基本的 CodeIgniter 设置，并专注于集成 Pingpp 进行支付处理，例如启用用户通过支付宝 PC 直连支付（尽管你可以将其适配到其他渠道）。

前提条件

1. **Pingpp 账户**：在 Ping++ 注册并获取你的 **API Key** 和 **App ID**。你将有一个测试密钥 (`sk_test_...`) 用于开发，和一个生产密钥 (`sk_live_...`) 用于生产。
2. **Pingpp PHP SDK**：通过 Composer 安装 Pingpp PHP SDK：

```
composer require pingplusplus/pingpp-php
```

确保 SDK 在你的 CodeIgniter 应用中自动加载（例如，将其放在 `application/libraries` 或配置 Composer 自动加载）。

3. **CodeIgniter 设置**：你应该有一个工作的 CodeIgniter 应用程序，其中包含用户和费用的控制器和数据库模型。
-

逐步集成

1. 设置控制器 创建一个控制器（例如 `PaymentController.php`），它扩展了类似于 `BaseController` 的基础控制器。在构造函数中根据环境（开发或生产）配置 Pingpp API 密钥。

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

require_once APPPATH . '/libraries/REST_Controller.php';
require_once APPPATH . '/vendor/autoload.php'; // 假设 Composer 自动加载

class PaymentController extends REST_Controller {

    public $userDao;
    public $chargeDao;

    public function __construct() {
```

```

parent::__construct();

// 根据环境设置 Pingpp API 密钥
$isLocalDebug = ENVIRONMENT === 'development'; // CodeIgniter 环境
if ($isLocalDebug) {
    \Pingpp\Pingpp::setApiKey('sk_test_your_test_key_here'); // 替换为你的测试密钥
} else {
    \Pingpp\Pingpp::setApiKey('sk_live_your_live_key_here'); // 替换为你的生产密钥
}

// 加载模型
$this->load->model('UserDao');
$this->userDao = new UserDao();
$this->load->model('ChargeDao');
$this->chargeDao = new ChargeDao();
}
}

```

- 替换 'sk_test_your_test_key_here' 和 'sk_live_your_live_key_here' 为你的实际 Pingpp API 密钥。
- 确保 UserDao 和 ChargeDao 已实现以处理用户认证和费用存储（见步骤 2）。

2. 创建支持模型 你需要模型来管理用户会话和存储费用详情。

UserDao.php（简化示例）：

```

<?php
class UserDao extends CI_Model {
    public function findUserBySessionToken($token) {
        $query = $this->db->get_where('users', ['session_token' => $token]);
        return $query->row(); // 返回用户对象或 null
    }
}

```

ChargeDao.php（简化示例）：

```

<?php
class ChargeDao extends CI_Model {
    public function add($orderNo, $amount, $userId, $ipAddress) {
        $data = [
            'order_no' => $orderNo,
            'amount' => $amount,

```

```

        'user_id' => $userId,
        'client_ip' => $ipAddress,
        'status' => 'pending',
        'created_at' => date('Y-m-d H:i:s')
    ];
    $this->db->insert('charges', $data);
}
}

```

创建相应的数据库表（users 和 charges），并包含适当的字段。

3. 实现费用创建方法 添加一个方法来创建 Pingpp 费用并将其返回给客户端。此示例使用支付宝 PC 直连作为支付渠道。

```

public function createChargeThenResponse($amount, $subject, $body, $metaData, $user) {
    // 生成唯一订单号
    $orderNo = $this->generateOrderNo();

    // 根据环境设置 App ID
    $isLocalDebug = ENVIRONMENT === 'development';
    $appId = $isLocalDebug ? 'app_your_test_app_id' : 'app_your_live_app_id';

    // 获取客户端 IP 地址
    $ipAddress = $this->input->ip_address();
    if ($ipAddress === ':::1') { // 处理本地调试情况
        $ipAddress = '127.0.0.1';
    }

    // 创建费用
    try {
        $charge = \Pingpp\Charge::create([
            'order_no' => $orderNo,
            'app' => ['id' => $appId],
            'channel' => 'alipay_pc_direct', // 更改此处以使用其他渠道（例如，'wx' 表示微信）
            'amount' => $amount, // 以分为单位（例如，1000 = 10 CNY）
            'client_ip' => $ipAddress,
            'currency' => 'cny',
            'subject' => $subject,
            'body' => $body,
            'metadata' => $metaData,

```

```

        'extra' => [
            'success_url' => 'http://yourdomain.com/payment/success' // 替换为你的成功 URL
        ]
    });

    // 将费用详情存储在数据库中
    $this->chargeDao->add($orderNo, $amount, $user->id, $ipAddress);

    // 将费用对象作为 JSON 返回
    $this->output
        ->set_status_header(200)
        ->set_content_type('application/json', 'utf-8')
        ->set_output(json_encode($charge));
} catch (\Pingpp\Error\Base $e) {
    log_message('error', 'Pingpp Charge Failed: ' . $e->getMessage());
    $this->response(['error' => '支付创建失败'], 500);
}
}

private function generateOrderNo() {
    return uniqid(); // 简单唯一 ID; 如果需要, 替换为更强大的生成器
}

```

- 替换 'app_your_test_app_id' 和 'app_your_live_app_id' 为你的 Pingpp App ID。
- 调整 'success_url' 为你应用程序的成功页面 URL。

4. 创建结账方法 添加一个公共方法来处理用户结账，调用费用创建方法。

```

public function checkout_post() {
    // 检查用户是否登录
    $user = $this->getSessionUser();
    if (!$user) {
        $this->response(['error' => '用户未登录'], 401);
        return;
    }

    // 示例订单详情
    $amount = 1000; // 10 CNY (示例; 在实际应用中动态计算)
    $subject = '订单支付';
    $body = '订单 #123 的支付';
}

```

```

$metadata = ['order_id' => '123']; // 附加订单特定数据

// 创建费用并响应
$this->createChargeThenResponse($amount, $subject, $body, $metadata, $user);
}

private function getSessionUser() {
    $token = $this->input->get_request_header('Authorization', TRUE); // 根据你的认证方法调整
    if ($token) {
        return $this->userDao->findUserBySessionToken($token);
    }
    return null;
}

```

5. 处理客户端响应 createChargeThenResponse 方法返回一个 JSON charge 对象。对于 alipay_pc_direct，它包括一个包含支付 URL 的 credential 字段。在客户端（例如，使用 JavaScript）处理此内容。

示例（前端 JavaScript）：

```

fetch('/payment/checkout', {
    method: 'POST',
    headers: { 'Authorization': 'your-session-token' }
})
.then(response => response.json())
.then(charge => {
    if (charge.credential && charge.credential.alipay_pc_direct) {
        window.location.href = charge.credential.alipay_pc_direct; // 重定向到支付宝
    } else {
        console.error('支付错误:', charge);
    }
})
.catch(error => console.error('错误:', error));

```

6. 成功页面 创建一个成功页面（例如 payment/success），在支付后显示确认。如果需要，可以在此处手动更新订单状态，尽管建议使用 webhook 以确保可靠性（见步骤 7）。

```

public function success_get() {
    // 可选：在此处验证支付状态
    $this->load->view('payment_success'); // 加载视图
}

```

7. (可选) 处理 Webhook Pingpp 通过 webhook 发送异步支付状态更新。设置一个端点来接收这些通知并更新你的数据库。

```
public function webhook_post() {  
    $rawData = file_get_contents('php://input');  
    $event = json_decode($rawData, true);  
  
    if ($event['type'] === 'charge.succeeded') {  
        $charge = $event['data']['object'];  
        $orderNo = $charge['order_no'];  
        $this->chargeDao->updateStatus($orderNo, 'paid');  
    }  
  
    $this->response(['status' => 'ok'], 200);  
}
```

- 向 ChargeDao 添加一个 updateStatus 方法以更新费用状态。
- 在 Pingpp 仪表板中配置你的 webhook URL (例如, <http://yourdomain.com/payment/webhook>)。

完整示例使用

1. **用户操作**: 用户登录并通过 POST 请求到 `/payment/checkout` 启动结账。
2. **服务器响应**: 服务器创建费用并返回包含支付详情的 JSON 对象。
3. **客户端处理**: 前端将用户重定向到支付宝支付页面。
4. **支付完成**: 支付后, 用户被重定向到成功 URL。
5. **Webhook 更新**: Pingpp 通知你的 webhook 端点, 你更新费用状态。

附加说明

- **支付渠道**: 将 `'alipay_pc_direct'` 替换为其他渠道 (例如, `'wx'` 表示微信) 以满足需要。查看 Pingpp 文档以获取渠道特定的 `extra` 参数。
- **错误处理**: 增强生产环境的错误处理 (例如, 重试逻辑、详细错误消息)。
- **安全性**: 验证用户输入 (`$amount` 等) 并保护你的 API 端点。

此示例为将 Pingpp 集成到你的 CodeIgniter 应用程序中提供了一个坚实的基础, 可以适应各种用例, 如电子商务或捐赠。