

通道对接文档

普通通道下单、查询、v1 支付通知与三语言示例。

文档 2.9 · 2026-10-21

UTF-8 表单 · MD5

内置账号 demo · 同一套正式接口

下载 PDF (与在线文档同步)

01 / GET STARTED

下载后直接联调

源码包已内置平台现有测试商户 **demo** 和现有签名密钥，直接走正式接口。无需另建账号、领取配置或切换环境；对接后换成自己的商户账号和密钥，接口与签名代码继续使用。

下载源码 → 运行 balance → 运行 demo → 接收自己的通知 → 更换商户账号 / 密钥

用途	地址 / 约定
下单	<code>https://payment.271458.xyz/?c=Pay</code>
查询	<code>https://payment.271458.xyz/?c=Pay&a=query</code>
余额查询	<code>https://payment.271458.xyz/?c=Pay&a=querybalance</code>
普通 v1 示例默认账号 / 通道	demo / 2601 (GPay)；示例金额 10000.00，随商户业务调整。
请求编码	POST， <code>application/x-www-form-urlencoded</code> ； <code>charset=UTF-8</code> 。format=json 控制响应，不是发送 JSON 请求体。
时间戳	UTC 秒级 Unix 时间戳，与服务器允许偏差最多 300 秒。

1. 下载源码包并解压，选择 Python、PHP 或 C#。
2. 运行 `selftest` 离线校验签名，运行 `balance` 验证正式接口身份。
3. 运行 `demo`，自动生成随机商户单号，下单后立即查询。
4. 对接自己的回调服务；完成后在 `merchant-config.json` 更换账号和密钥。

code=1 仅表示本次请求成功。下单成功不是支付成功；发货/入账前必须正确验签并核对通知，或通过查询确认订单状态。

下载 C# / Python / PHP 源码包

查看已内置的账号与密钥

下载 PDF

现有测试商户 · 同一套正式接口

后台试用：登录商户后台，账号 `demo`，密码 `demo`，填写页面图形验证码即可登录并查看此测试商户的交易记录。内部 ID 为 1001。API 签名仍使用示例配置中的 `PAY_KEY`，不能使用登录密码签名。

平台指定 `demo` 作为公开联调账号，它的现有 key 已包含在源码包的 `merchant-config.json`。三种语言读取同一份配置，不需要修改程序逻辑或启用额外开关。

配置项	默认值 / 使用方式
PAY_GATEWAY	https://payment.271458.xyz/，测试账号和正式商户使用同一网关。
PAY_MCH_ID	demo，联调完成后改成自己的商户登录账号。
PAY_KEY	已填入上述账号现有密钥；更换商户时同时改成自己的 key。
PAY_PTYPE / PAY_MONEY	2601 / 10000.00，是当前测试账号配置；自己的商户需开通对应通道，金额依实际限额。
PAY_NOTIFY_URL	默认平台测试通知入口；开始接入自己的业务通知时改成自己的公网 HTTPS 地址。

配置优先级：环境变量 > 可选私有 `merchant.env`（或 `PAY_ENV_FILE` 指定文件）> 源码包内置 JSON。直接运行不需要任何环境变量。数字通知商户 ID 可额外核对，但不是开始联调的必填配置。

共享测试账号的订单与余额用于联调，可能被其他联调者改变；自己的正式商户身份、订单与账务分别处理。
`demo` 自动生成随机订单号，避免复制固定例子造成撞单。

平台将该商户用于测试账务；它仍使用真实 API 和当前通道。当前 GPay 配置指向正式上游，代码没有按此商户强制切为上游虚拟环境的分支。示例不会自动付款、代付或充值；返回的收款账号/二维码可能接收真实资金，不能当作零资金效果的模拟码。验证账号和签名可先运行 `balance`，无需付款。

▼ 可选独立模拟环境（不属于默认接入流程）

之前的 `/sandbox.php`、`sandbox.env` 仍可显式使用，仅适合本地模拟订单、通知和幂等测试。它不代表真实通道验收；`simulate / notify / reset` 只在该环境有效，默认示例不使用。

下单接口

POST `/?c=Pay`

签名使用下表除 `sign` 以外的九项字段；可空字段也需要发送并参与签名。不要把路由 `c`、`a` 加入业务表单。

字段	传入	说明
mch_id	必填	商户登录账号；示例已填写平台现有测试账号 <code>demo</code> 。

字段	传入	说明
ptype	必填	通道编号。示例 2601；当前接口只接受大于 900 的通道，再按商户开通配置校验。
order_sn	必填	商户业务订单号，最多 32 字符；建议字母数字、下划线或短横线，全局唯一；共享测试账号也须使用随机单号。
money	必填	十进制金额字符串，如 1.00。币种、实际限额以商户开通约定为准；不要用浮点数拼接签名。
goods_desc	可空，必须传	商品描述，空值发送 goods_desc=。
client_ip	可空，必须传	下单用户 IP，空值发送 client_ip=。
format	必填	显式发送 json 或 page，参与签名。
notify_url	必填	商户接收通知的 HTTPS 地址；使用原始 URL 值签名，再进行表单编码。不是收银台地址。
time	必填	当前 UTC 秒级 Unix 时间戳，以字符串发送。
sign	必填	32 位小写 MD5；sign 自身不参与计算。

当前生产代码会对 notify_url 额外执行一次 URL 解码及 HTML 实体解码。请使用简单、稳定的通知路径，避免 URL 内嵌百分号编码、`+` 或 HTML 实体导致签名值变化；普通查询参数中的 `&` 应按表单正确编码。示例按照原始字符串签名。

生产普通明文接口的公共参数层还会 trim、引号转义、百分号转义和 HTML 转义。goods_desc 等字段含引号、`&`、`<`、`>`、`%` 或首尾空白时，值可能在验签前发生变化。示例默认使用空描述；生产特殊字符须单独联调。UTF-8 固定向量验证的是签名算法，不代表旧参数过滤对所有特殊字符已兼容。

04 / SIGNATURE

签名规则与固定样例

- 确定业务字段，保留空字符串，排除 `sign`。
- 参数名按 ASCII 字典序升序排列，拼接原始 `key=value`，使用 `&` 连接。
- 末尾追加 `&key=签名密钥`，不加空格、引号或换行。
- 以 UTF-8 计算 MD5，输出小写十六进制；再做表单编码发送。

离线签名固定向量

此 key 和历史时间戳仅用于离线算法校验，不能提交网关。所有语言的 selftest 都包含该向量。

```
client_ip=&format=json&goods_desc=&mch_id=demo_merchant&money=500.00&notify_url=https://merchant.example/notify&order_sn=DEM0202609170001&ptype=12&time=1789574400&key=test_key_only_20260917
```

预期结果： `1ff18ca5f270edfb71f29b2f7f3a5e11`

500 、 500.0 、 500.00 签名不同；空串、null 和省略不同；URL 原始值与已编码值不同。不要签名后再格式化金额、改写 URL，或对整段 JSON 计算 MD5。

请求	参与签名的字段
下单	原九项非 sign 字段；传入 crypto_callback_version 时额外包含该字段
订单查询	mch_id、out_order_sn、time
余额查询	mch_id、time
支付通知	mch_id、pt_order、sh_order、money、pay_money、goods_desc、status、time；当前 msg 不参与

密钥只保存在服务端。验签使用恒定时间比较；不要输出密钥或完整的含 key 签名原串到日志。

05 / RESPONSE

下单响应与收银台

format=json

```
{
  "code": "1", "msg": "下单成功",
  "data": {
    "order_sn": "2026101700010012", "ptype": "2601",
    "ptype_name": "GPay支付", "money": "10000.00",
    "pay_url": "https://通道返回的收银台或二维码地址"
  }
}
```

保存 data.order_sn（平台单号），优先使用通道返回的 data.pay_url。**pay_url 的具体形式取决于通道：可能是收银台页面，也可能是通道返回的二维码图片 URL。**当前 GPay 集成返回上游提供的二维码链接，不要统一假定它是 HTML 页面或本平台 /info 地址。它也不是商户通知地址。部分通道可能额外返回 extra，不保证所有通道返回相同收款字段。

format=page

成功返回 HTTP 302 和 Location，目标与通道下单返回的链接一致。后端调试需关闭自动跳转检查首跳；失败仍可能返回 HTTP 200 的 JSON。相对 Location 以网关地址解析。

```
{"code": "-1", "msg": "签名错误"}
```

同时检查 HTTP 状态、JSON 结构和 code。当前 JSON 可能标注 text/html，不要只按 Content-Type 判断，也不要把下单成功当作实际付款成功。

06 / QUERY

查询与订单号对应关系

```
POST /?c=Pay&a=query
```

发送 `mch_id`、`out_order_sn`、`time`，计算这三个字段的签名后加入 `sign`。`out_order_sn` 填下单时的商户订单号。

业务含义	下单请求	下单响应	查询请求	查询响应	通知
商户订单号	order_sn	—	out_order_sn	data.out_order_sn	sh_order
平台订单号	—	data.order_sn	—	data.orders[].order_sn	pt_order

```
{
  "code": "1", "msg": "ok",
  "data": {
    "mch_id": "demo", "out_order_sn": "MY_ORDER_001",
    "orders": [{
      "order_sn": "2026101700010012", "money": "10000.00", "pay_money": "10000.00",
      "order_time": "1792195200", "pay_time": "1792195260",
      "status": "9", "status_flag": "已支付",
      "notice_status": "4", "notice_status_flag": "回复成功", "notice_msg": "success"
    }]
  }
}
```

当前返回 `data.orders[]` 数组。 订单字段位于 `data.orders[]` 的每个元素内，不直接位于 `data`。统一类型后比较 `String(order.status) === "9"`，不要将 `code=1` 当作付款成功。生产可能返回多个子订单，需按商户业务规则逐单关联。

付款成功只看订单 `status=9`，其他中间/失败状态按平台配置处理。通知状态是投递结果，与付款状态分开：`4` 回复成功、`3` 回复异常、`2` 未成功送达。

超时与重复订单

发送前持久化商户订单号。网络超时可能已经创建成功，应先按原商户订单号查询，不要自动换单号重下。当前生产下单重复检查是全平台范围，建议使用带商户前缀的随机单号；重复提交会报错，不保证返回原订单。

余额查询

`a=querybalance`，签名字段为 `mch_id,time`，返回 `data.mch_id` 和 `data.balance`。共享测试账号余额用于联调，不能当成自己商户的可用资金。

07 / WEBHOOK

支付通知与业务处理

平台向下单时的 `notify_url` POST UTF-8 表单。以下是旧版 v1 字段；未选择 v2 的接入保持不变。数字货币 v1 仅通知最终成功，不通知确认中或超时。

字段	参与签名	说明
mch_id	是	数字商户用户 ID，不是请求中的商户登录账号。示例默认通过商户 key、订单号和金额核对身份；可选 PAY_CALLBACK_MCH_ID 额外校验该数字 ID。
pt_order	是	平台订单号，核对下单 / 查询时保存的单号。
sh_order	是	商户业务订单号。
money	是	订单金额，保持原始字符串验签。
pay_money	是	付款金额，调额场景须按约定处理；示例要求与本地金额一致。
goods_desc	是	商品描述，可能为空。
status	是	success 表示成功，fail 表示失败。
time	是	订单完成时间，秒级 Unix 时间戳；通知可能延迟重试，不套用请求的 300 秒窗口。
sign	否	签名本身。
msg	否	当前服务端在签名后追加的说明字段；不能用它判断资金状态。

v1 通知只对上表八项业务字段验签：mch_id、pt_order、sh_order、money、pay_money、goods_desc、status、time；排除 sign 和 msg，保留空字符串。三语言 v1 示例与本规则一致。数字货币 v2 通知使用另一套字段集合：所有非 sign 字段（包括 msg）均参与签名，详见[数字货币 v2](#)。不要混用两版规则。

- 1. 解析表单并保留原始字符串，拒绝重复键和结构异常。
- 2. 计算签名并恒定时间比较，失败时不处理业务。
- 3. 核对商户 ID、本地订单号、平台单号、金额和 status。
- 4. 在事务中幂等更新支付状态及相关业务；重复有效通知不可重复入账或发货。
- 5. 持久化成功后响应 HTTP 200，响应体仅为小写 `success`。不能返回 JSON、HTML 或额外文本。

示例对有效重复通知再次返回 `success`；验签、金额或本地订单校验失败返回 HTTP 400 `fail`。本地示例只保存支付状态，不替商户执行业务入账。正式环境应将发货、入账等业务写入同一个数据库事务。

正式接口由平台支付结果触发通知，投递和重试受后台队列配置影响。普通示例默认 notify_url 是平台原有 v1 测试验签入口，不支持 v2，不提供开发者通知记录查询；业务持久化、幂等与 v2 验收必须换成自己的公网 HTTPS 通知服务。

08 / SOURCE EXAMPLES

C# · Python · PHP 对接源码

提供完整客户端、签名函数、订单查询、余额查询和可启动的回调服务。下载包包含运行说明和固定测试向量；现有测试账号与 key 已随 merchant-config.json 内置，解压即可运行。

Python: 3.10+ 标准库; PHP: 8.1+ 与 cURL; C#: .NET 8+, 无需 NuGet 包。回调服务默认只监听本机, 联调时请在前面配置公网 HTTPS 反向代理。

三个示例的命令一致: `selftest`、`balance`、`demo`、`create`、`query ORDER`。先在一个语言示例中创建并保存订单, 再由对应示例回调服务处理。

文件锁存储用于单机示范, 不能替代生产数据库事务。示例支持普通 MD5 表单接入; 生产 RSA、代付与充值不在这三个示例范围内。不要将配置文件放到可公开下载的 Web 目录。

09 / OPTIONAL ENCRYPTION

RSA 可选加密

仅当商户开启 RSA 时使用; 当前示例测试商户未启用 RSA。加密规则如下:

1. 先计算 MD5 sign, 放入完整业务参数对象。
2. 将该对象转换为 UTF-8 JSON (中文不转义, 等价 PHP `json_encode(params, JSON_UNESCAPED_UNICODE)`), 再 Base64 编码。
3. 使用平台提供的 PEM 公钥 (已启用 RSA 的商户向管理员获取)、**RSA PKCS#1 v1.5** 填充, 按 `密钥位数 / 8 - 11` 字节分块加密; 拼接每块密文字节后整体 Base64 编码。
4. 以表单 `encrypted=密文` 提交原业务接口; 不要把密文当成签名。
5. RSA 开启时, 平台通知用商户上传的 PEM 公钥加密; 商户用自己的私钥按每块 `密钥位数 / 8` 字节解密, 拼接后 Base64 解码、JSON 解析, 最后按通知规则验证 MD5。

2048 位公钥对应 245 字节明文块、256 字节密文块。不要混用 OAEP, 不要对每块单独 Base64 后拼字符串。私钥永远留在自己服务端。三个下载示例尚不包含 RSA 联调; 正式启用前须单独验证密钥、分块和双向通知。

10 / TROUBLESHOOTING

排错与验收

现象	检查与处理
签名错误	先运行 <code>selftest</code> ; 检查空字段、金额尾零、原始 URL、时间戳及通知 <code>msg</code> 排除规则。发送表单, 不是 JSON。
缺少参数	检查字段名、Content-Type 及表单解析; <code>goods_desc</code> 和 <code>client_ip</code> 可以为空, 但仍发送。
通道不存在 / 未开通	示例测试商户当前使用 2601; 自己的商户使用已开通的大于 900 的通道。1、2、12 等历史编号不适用于当前接口; 离线向量中的 12 仅用于算法校验。
HTTP 200 但失败	检查 JSON <code>code/msg</code> ; <code>code=1</code> 也只表示本次操作成功。
page 返回空体 / HTML	关闭自动重定向, 检查首跳 302 和 Location; HTML 可能来自跟随后的收银台。

现象	检查与处理
下单超时	保留原商户订单号，先查询；不要自动换号或无条件重下。
通知验签失败	按八项签名字段验签，排除 msg；确认测试 / 生产 key 没有混用，保留原始字符串。
通知拒绝 / 未入账	核对数字商户 ID、本地订单、平台单号、两项金额、数据库事务及纯文本 success。先查询付款状态。
上游 get token fail	表示请求已进入通道认证，但当前 GPay 认证未成功；需平台核对上游账号、凭据或网络，不是让商户盲目换自己的签名算法。

上线前验收

- 固定签名向量一致，中文、空字段、金额尾零及表单编码一致。
- 创建订单、收银台、查询、余额和失败路径符合预期。
- 通知成功、重复投递、错误签名、金额不匹配和未知订单处理正确。
- 超时后能够按原单号查询并补齐平台单号。
- 正式配置、实际通道、RSA 要求、调额规则、结算及通知重试策略已单独确认。

排错提供发生时间及时区、接口、请求方法、脱敏参数、商户单号、HTTP 状态、code/msg 和 Location。不要提供生产密钥、私钥或完整含密钥的签名原串。

11 / VERSION & COMPATIBILITY

更新与兼容说明

2026-10-24 · 文档 3.1: 数字货币按 USDT 计价并使用独立服务费账户，新增整单累计阶梯、自动充值、主测隔离和法币币种设置；旧订单按原快照处理。

2026-10-21 · 文档 2.9: 公开 API 测试商户由 13122222222 改名为 demo，后台密码 demo。内部商户 ID 1001、原订单、通道及 API 签名密钥保持不变。新 API 请求使用 mch_id=demo，登录密码不能代替 API 签名密钥。

2026-10-19 · 文档 2.8: WP 插件安装说明独立为文档及 PDF，通道对接和数字货币文档不再重复收录安装正文。

2026-10-19 · 文档 2.7: 新增 WooCommerce 支付插件下载与配置、固定汇率及回调验收说明；保持现有商户接口协议不变。

2026-10-19 · 文档 2.6: 在线通道 / 数字货币文档与 PDF 统一从同一正文生成，下载不再提供 DOCX。同步当前网关、通道编号、data.orders[] 查询结构、v1 固定八字段验签与 v2 全部非 sign 字段验签，修正历史文档差异。文档同步不改变接口协议。

2026-10-19 · 文档 2.5: 新增数字货币 v2 独立三语言完整示例、SQLite 事务与 outbox、28 组离线通知、测试流隔离、三链真实下单/查询响应、6/18 位精度说明及默认 v1 通知地址能力边界。

2026-10-18 · 文档 2.4: 数字货币新订单必须配置有效费率并有足额人民币余额；新增固定汇率、预扣与退费、手续费查询字段及后台设置，默认 1 USDT = 7 CNY。

2026-10-18 · 文档 2.3: 数字货币新增可选 v2 三状态通知，包含签名、重试、幂等、乱序和确认宽限期规则；新增 PHP 验证函数，旧 v1 接入保持不变。

2026-10-17 · 文档 2.2: 按平台现有操作方式，改为直接使用测试商户 13122222222 及其现有 key、正式网关和 2601 通道。源码包内置配置；新增 demo 自动随机下单并查询，取消额外环境启用开关，数字回调商户 ID 改为可选额外核对。联调后替换账号和 key，保持同一接口协议。

沿用 2.1 对原 DOCX 的修订：查询为 `data.orders[]`；旧 v1 通知包含数字 `mch_id`、`pay_money`、`goods_desc`，`msg` 不签名（v2 的 `msg` 参与签名）；通道编号大于 900；时间偏差 300 秒；重复单号按当前接口全平台检查。补充 `pay_url` 可能为通道二维码图片链接，不能一律当成 HTML 收银台。

文档版本不等于 API 版本。账号验签、余额查询、上游下单、实际付款、通知和结算分别验收，不能用历史或模拟结果证明当前真实通道可用。

商户开发文档 · `payment.271458.xyz` · 2026-10-19

共享测试商户仅供联调；业务付款以正确验签及核对后的订单结果为准。