版本号：**V1.2**

快钱安全支付网关

商户接口规范

上海快钱信息服务有限公司

目　录

1. 概要

1.1 目的

帮助商户接入快钱安全支付网关，快速掌握支付平台提供的各种功能，便于尽快投入使用。

1.2 使用对象

快钱安全支付网关商户的网上应用开发人员、维护人员和管理人员。

他们应具备以下基本知识：

- 了解 Microsoft Windows/NT、Windows9x、Windows 2000、HP-UX、AIX、SUN Solaris、Linux、BSD 等操作系统的其中一种；
- 了解上述系统上的网站设置和网页制作方法；
- 了解 HTML 语言以及 CGI（Common Gateway Interface）或 ASP（Active Server Pages）或 ISAPI 的开发方法或 PHP 或 JAVA 等开发语言；
- 了解信息安全的基本概念。

1.3 技术支持

一般事务咨询：请访问快钱网站或联系快钱客服

技术支持邮箱：support@99bill.com

技术支持热线：(021)58777299-864/876

技术支持时间：9:00-18:00

2 商户申请开通流程

第一步：注册成为快钱用户
第二步：申请成为高级帐户
第三步：登录快钱网站，获取商户编号、设置商户密钥、商户网址等信息

注：快钱全面开通实时注册、实时开通功能，只需要在线注册并申请成为高级帐户即可实时开始使用快钱安全支付网关。目前初级用户也可以试用快钱网关。

3 商户管理系统操作说明

3.1 商户登录界面

商户可以在以下地址登录商户管理系统 **http://www.99bill.com** ，正确输入用户名、及验证码后，点击“登录”进入欢迎界面，输入密码后即进入到快钱帐户管理系统。

3.2 支付网关商户管理

在用户登录快钱帐户管理后，在“支付网关”一栏中，商户可以选择“历史交易查询”、“下载开发包”、“商家信息设定”管理相关信息。

3.3 交易查询

用户在“支付网关”中选择“历史交易查询”后，将看到支付网关历史交易查询界面，用户可以根据交易状态和交易时间来查询交易历史。如下图。

3.4 开发包下载

用户在“支付网关”中选择“下载开发包”后，即可下载开发包。开发包中包含有快钱支付网关规范文档、接口示例以及快钱支付 Logo.

3.5 修改商户信息

用户在“支付网关”中选择“商家信息设定”后，即可获得商户编号，修改包括商户密钥、商户名称、商户网址等商户信息。

4 支付接口开发

4.1 商户端系统环境要求

1.系统要求：Windows 9x、Windows NT、Windows 2000、Windows XP、Windows 2003、UNIX 及 Linux 以上一种系统。

2.WEB 服务器要求：IIS、Tomcat、weblogic 等。

4.2 商户开发步骤和详细说明

当消费者在商户处完成购物过程，形成最终订单且消费者选择快钱安全支付网关付款方式时，该接口程序将消费者订单中有关支付的信息提交到快钱交易平台，消费者在该平台完成网上支付后，将通过 URL 方式将支付结果返回给商户。

4.3 商户提交付款订单接口

（商户-->快钱）

用途：用来接收商户提交的订单信息

1、FORM 表单设置

商户必须以表单的 **POST** 方式将交易数据提交到快钱提供的支付接口 **URL: https://www.99bill.com/webapp/receiveMerchantInfoAction.do**，表单数据字段名与格式如下表：

变量名称	变量命名	长度定义	说明	举例
商户编号*	**merchant_id**	**20**	商户在快钱的唯一身份标识。	88990412099278681
订单编号*	**orderid**	50	商户网站形成的订单号,同一家商户订单编号不能有重复，建议使用流水号。	2005101513223232353 3
订单金额*	**amount**		不可为空。货币型或整型数字。	2321.13
货币种类	**currency**	1	货币种类，缺省为 1。 1 为 人民币； 3 为 预付费卡。	1

商户 URL	**merchant_url**	256	商户接受快钱交易结果的 **URL**，请尽量使用 **http** 的 **80** 端口接受。如使用 **https** 地址，请确保所使用的证书为系统内嵌证书，比如：**VeriSign**	http://domain/reply.cgi 错误的用法： http://domain/rcv.asp?pn=pv
订货人姓名	**pname**	64	中文或英文字符。建议填写。 （如果包含中文字符，请 url 编码后提交，避免页面显示乱码）	CoCo 可可
商品信息	**commodity_info**	512	商品的描述。建议填写。 （如果包含中文字符，请 url 编码后提交，避免页面显示乱码）	Beer 啤酒
订货人 Email	**pemail**	64	订货人的有效 **email**。由商家网站传递到快钱支付页面并显示。	username@domain.com
合作伙伴商户编号	**pid**	20	快钱的代理以及特殊合作伙伴的商户编号。可为空值。	
附加信息	**merchant_param**	512	商户需要传递的参数组成的字符串（比方:收货人的相关信息）。	V_1|V_2
是否校验*	**isSupportDES**	1	快钱是否校验商户订单数据的签名。必填项目 **1** 为 不校验 2 为 校验	2
MAC 校验域	mac	32	如 **isSupportDES** 为 **2** 则必填，详情见 md5 检验串生成方法。	

注：以上参数值中不能包含以下特殊字符 ‘’ ， “ ” &<>()，否则将会导致校验码匹对失败，商户得到“校验码错误”的提示！

当 currency 值为 3 时,系统自动切换到预付费卡支付界面。商户快钱帐户上的预付费卡余额可以结算为人民币余额。结算的比率请参见快钱网站公布的资料。

Md5 校验串生成方法：当消费者在商户端生成最终订单的时候，将订单中的 merchant_id ，orderid，amount，merchant_url， key 五个参数的 value 值根据：

“merchant_id=merchant_id 值 &orderid=orderid 值 &amount=amount 值 &merchant_url=merchant_url 值&merchant_key= key 值”

的规则拼成一个无间隔的字符串(char 型，顺序不要改变)。其中参数 key 为商户自行在快钱帐户管理里设置的 16 位密钥值。例如：

merchant_id=88990412099278681&orderid=20051015132232323533&amount=2321.13&merchant_url=http://domain/reply.cgi&merchant_key= 99bill0123456789

注意：以上拼凑值不要有空格！

使用标准 MD5 算法对该字符串进行加密，加密结果全部转换成大写后，即为我们所需的订单 MD5 校验码，将其写入 mac 字段即可。

我们在开发包中提供了各种开发语言工具下的 MD5 校验码生成方法和以及订单生成范

例。

4.4 订单支付结果页面返回接口

（快钱-->商户）

支付完成以后，快钱支付网关将订单支付结果数据返回到商户提交表单时所设定的**merchant_url**，数据名称与格式如下表：

变量名称	变量命名	长度定义	说明	举例
商户编号*	**merchant_id**	**20**	商户编号。	88990412099278681
订单编号*	**orderid**	50	商户订单编号	200510151322323235 33
订单金额*	**amount**		用户在快钱的实际支付金额。货币型或整型数字	2321.13
交易日期	**date**	8	快钱系统产生，格式 YYYYMMDD	20050512
附加信息	**merchant_para m**	512	商户自定义的参数	V_1|V_2
交易结果 *	**succeed**	1	**“Y”:**支付成功，**“N”:**支付失败	Y
优惠券代码	**couponid**		支付人使用的优惠券代码	
优惠券面额	**couponvalue**		支付人使用的优惠券面额	
MAC 校验域	mac	32	根据交易结果数据及商户密钥产生的签名	

说明：

1.快钱返回同一笔订单不限次数，请商户判断，进行可能的重复记录的处理。

2.由于快钱采用服务器远程打开 **merchant_url** 并同通知用户浏览器方式，因此，商户 **merchant_url** 页面中对路径引用请采用绝对路径（如图片，链接等）。

商户验证交易结果的方法：将订单中的 merchant_id ，orderid，amount，date，succeed， key 六个参数的 value 值根据：

“merchant_id=merchant_id 值&orderid=orderid 值&amount=amount 值& date = date 值 &succeed=succeed 值&merchant_key= key 值”

的规则拼成一个无间隔的字符串(char 型，顺序不要改变)。其中参数 key 为商户自行在快钱帐户管理里设置的 16 位密钥值。例如：

“merchant_id=88990412099278681&orderid=2005101513223232353 3&amount=2321.13& date= 20050512&succeed=Y&merchant_key=99bill0123456789”

注意：以上拼凑值不要有空格！

使用标准 MD5 算法对该字符串进行加密，加密结果全部转换成大写后即为校验码，把它和收到的 mac 字段进行比较，相同的话即为交易结果数据验证成功。

我们在开发包中提供了各种主要开发语言工具下的 MD5 校验码生成方法和以及接收订单结果的范例。

5 自动对账接口

5.1 方式一、根据交易日期对账(批量对账)

自动对账/冲正请求由商户通过调用快钱提供的 web service 远程方法主动发起，接口定义如下：

访问 webService 的 url: http://www.99bill.com/webapp/services/OrderManager?wsdl

```
String payCheck(
        String startdate,
        String enddate,
        String userID,
        String password,
    );
```

参数说明：

startdate	对账开始日期(日期格式为 8 位数字)
enddate	对账结束日期(日期格式为 8 位数字)
userID	商户在快钱的商户编号
password	商户在快钱的登录密码

返回值说明：

返回值是一个采用 XML 进行描述的文本信息(OrderResult)，包括所提交日期范围内，所有成功交易返回结果的信息。

其中包含字段为：

Msg	信息 值为 0 正常，其为他失败（包括个各种失败情况，登录失败，日期不正确等）

orderid 商户订单编号。

orderdate 交易日期

amount 交易金额。货币型或整型数字。

signature 签名值，该值为对每笔交易中的 orderid， amount，orderdate 三个字段用“|”组合后的字符串再用”|”与商户密钥组合，然后使用 MD5 算法进行的签名值。请商户接收到此参数后，使用同样的规则生成加密串进行对比验证。商户签名字符串格式为：orderid1|amount1| orderdate1|orderid2|amount2|orderdate2|orderid3|amount3|orderdate3|……|key 值

快钱返回的结果 XML 描述文本样例如下：

```
<OrderResult startdate="20050401" enddate="20050401">
            <Msg>0</Msg>
            <Order orderid="1000001" amount="10" orderdate="20050401" />
            <Order orderid ="1000002" amount="20" orderdate="20050401" />
            <Order orderid ="1000003" amount="10" orderdate ="20050401" />
            <Order orderid ="1000004" amount="30" orderdate ="20050401" />
      <signature> 96B406C29AD80CB0D4D71AF242C43AA9</signature>
</OrderResult>
```

5.2 方式二、根据交易订单号对账（单笔对账）

由商户通过以WEB表单形式提交对账请求，对账接口定义如下：

<form method=”post” action=”https://www.99bill.com/webapp/settleDealAction.do”>

<input name=”orderid”>

<input name=”merchant_id”>

<input name=”merchant_url”>

<input name=”mac”>

</form>

参数说明：

orderid　　商户订单编号

merchant_id　商户编号

merchant_url　商户接收查账结果 URL

mac　　商户根据merchant_id=merchant_id 值&orderid=orderid 值& merchant_url = merchant_url值&merchant_key= key值的拼成字符串,使用MD5 所进行的加密后全部转换成大写，其中参数key 为商户在快钱帐户管理里设置的 16 位商户密钥值。

快钱服务器收到对账请求后，首先验证商户数据的真实性，然后将以表单以GET的方式将订单支付结果数据返回到商户网站，表单数据名称及格式与网上支付时的返回格式非常类似，区别是多了**order**字段，如下表：

变量名称	变量命名	长度定义	说明	举例
查询情况	order	1	“0”表示没有查到，”1”表示查到	1
商户编号*	**merchant_id**	**20**	商户编号。	88990412099278681
订单编号*	**orderid**	50	商户订单编号	20051015132232323533
订单金额*	**amount**		实际交易金额。货币型或整型数字	2321.13
交易日期	**date**	8	格式 YYYYMMDD	20050512
附加信息	**merchant_param**	512	商户自定义参数。	V_1|V-2
交易结果 *	**succeed**	1	“**Y**”:支付成功，“**N**”:支付失败	Y
MAC 校验域	mac	32	根据交易结果数据及商户密钥产生的签名	

商户验证交易结果的方法：将订单中的merchant_id ，orderid，amount，date，succeed，key六个参数的value值根据：
“merchant_id=merchant_id 值&orderid=orderid 值&amount=amount 值& date = date 值&succeed=succeed值&merchant_key= key值”
的规则拼成一个无间隔的字符串(char 型，顺序不要改变)。其中参数key 为商户自行在快钱帐户管理里设置的 16 位商户密钥值。例如：
“merchant_id=88990412099278681&orderid=20051015132232323533&amount=2321.13&date= 20050512&succeed=Y&merchant_key=99bill0123456789”
注意：以上拼凑值不要有空格！

使用标准 MD5 算法对该字符串进行加密，加密结果即为我们所需的订单 MD5 校验码，将其写入 mac 字段即可。

我们在支付网关开发包中提供了各种开发语言工具下的 MD5 校验码生成方法。

由支付人直接发起对账请求：
对于一些需要实时开通服务或数字产品购买的交易订单（比如电影，音乐下载），如果因为

网络故障或其他因素商户没有收到快钱支付订单的情况，利用本接口可以由支付者本人通过商户网站发起单笔交易对账请求。对账成功后，商户可以立即为用户开通服务。

附录 MD5 算法简介。

MD5 的全称是 Message-Digest Algorithm 5（信息-摘要算法），在 90 年代初由 MIT Laboratory for Computer Science 和 RSA Data Security Inc 的 Ronald L. Rivest 开发出来，经 MD2、MD3 和 MD4 发展而来。它的作用是让大容量信息在用数字签名软件签署私人密匙前被"压缩"成一种保密的格式（就是把一个任意长度的字节串变换成一定长的大整数）。不管是 MD2、MD4 还是 MD5，它们都需要获得一个随机长度的信息并产生一个 128 位的信息摘要。虽然这些算法的结构或多或少有些相似，但 MD2 的设计与 MD4 和 MD5 完全不同，那是因为 MD2 是为 8 位机器做过设计优化的，而 MD4 和 MD5 却是面向 32 位的电脑。这三个算法的描述和 C 语言源代码在 Internet RFCs 1321 中有详细的描述 http://www.ietf.org/rfc/rfc1321.txt），这是一份最权威的文档，由 Ronald L. Rivest 在 1992 年 8 月向 IEFT 提交。

Rivest 在 1989 年开发出 MD2 算法。在这个算法中，首先对信息进行数据补位，使信息的字节长度是 16 的倍数。然后，以一个 16 位的检验和追加到信息末尾。并且根据这个新产生的信息计算出散列值。后来，Rogier 和 Chauvaud 发现如果忽略了检验和将产生 MD2 冲突。MD2 算法的加密后结果是唯一的--既没有重复。

为了加强算法的安全性，Rivest 在 1990 年又开发出 MD4 算法。MD4 算法同样需要填补信息以确保信息的字节长度加上 448 后能被 512 整除（信息字节长度 mod 512 = 448）。然后，一个以 64 位二进制表示的信息的最初长度被添加进来。信息被处理成 512 位 Damg?rd/Merkle 迭代结构的区块，而且每个区块要通过三个不同步骤的处理。Den Boer 和 Bosselaers 以及其他人很快的发现了攻击 MD4 版本中第一步和第三步的漏洞。Dobbertin 向大家演示了如何利用一部普通的个人电脑在几分钟内找到 MD4 完整版本中的冲突（这个冲突实际上是一种漏洞，它将导致对不同的内容进行加密却可能得到相同的加密后结果）。毫无疑问，MD4 就此被淘汰掉了。

尽管 MD4 算法在安全上有个这么大的漏洞，但它对在其后才被开发出来的好几种信息安全加密算法的出现却有着不可忽视的引导作用。除了 MD5 以外，其中比较有名的还有 SHA-1、RIPE-MD 以及 HAVAL 等。

一年以后，即 1991 年，Rivest 开发出技术上更为趋近成熟的 MD5 算法。它在 MD4 的基础上增加了"安全-带子"（Safety-Belts）的概念。虽然 MD5 比 MD4 稍微慢一些，但却更为安全。这个算法很明显的由四个和 MD4 设计有少许不同的步骤组成。在 MD5 算法中，信息-摘要的大小和填充的必要条件与 MD4 完全相同。Den Boer 和 Bosselaers 曾发现 MD5 算法中的假冲突（Pseudo-Collisions），但除此之外就没有其他被发现的加密后结果了。Van Oorschot 和 Wiener 曾经考虑过一个在散列中暴力搜寻冲突的函数（Brute-Force Hash Function），而且他们猜测一个被设计专门用来搜索 MD5 冲突的机器（这台机器在 1994 年的制造成本大约是一百万美元）可以平均每 24 天就找到一个冲突。但单从 1991 年到 2001 年这 10 年间，竟没有出现替代 MD5 算法的 MD6 或被叫做其他什么名字的新算法这一点，我们就可以看出这个瑕疵并没有太多的影响 MD5 的安全性。上面所有这些都不足以成为 MD5 的在实际应用中的问题。并且，由于 MD5 算法的使用不需要支付任何版权费用的，

所以在一般的情况下（非绝密应用领域。但即便是应用在绝密领域内，MD5 也不失为一种非常优秀的中间技术），MD5 怎么都应该算得上是非常安全的了。