API
Developer Reference Manual

Version 1.0

Table of contents

Table of contents **2**

Overview **4**

What is the HiPay Mobile API 4

What is an API? *4*

What is a REST Web Service? *4*

Who may use this API? 4

Knowledge and skills *4*

How to read this documentation? 5

Starting Guide **6**

Requests *6*

Responses *7*

Character encoding *8*

DATES *9*

Available formats 10

How to specify the return format? *10*

XML *10*

JSON *10*

Merchant Authentication **12**

Overview 12

Get an API key 12

Signatures **13**

Make an authenticated request *14*

After payment is received (Forward URL) **17**

Overview 17

Response *17*

Notification of payment (notification URL) **18**

Overview 18

Response *18*

Status codes *20*

Notification Response (notification URL) **21**

Overview....21
Parameters....21
Signature verification....22
Check the signature of a response from the API....22
Check the signature of a notification....24
Signature verification example....24
Testing....26
Virtual bank card....26
Test codes....26
Free codes (only for One-Time Fixed products)....27
More questions ?....27

Overview

What is the HiPay Mobile API

Introduction	HiPay Mobile provides access to key features of its payment engine through an API or Web Service.
Web Services types	There are several types of Web Services, like • REST, • XML-RPC, • SOAP, • etc. HiPay Mobile offers an API based on the most common type, *REST*.

What is an API?

Description	An API (Application Programming Interface) is a library of functions or procedures to develop applications.

What is a REST Web Service?

Description	REST (Representational State Transfer) is a way to build Web services. This is neither a protocol nor a format but a style of architecture. Systems that follow the REST principles are based simply on the architectural style of the Web, mainly revolving around the HTTP protocol.

Who may use this API?

This SDK is opened to all merchants who wish to dynamically integrate the HiPay Mobile payment solution in their sites or applications.

This document is specifically aimed to developers.

Knowledge and skills

The HiPay Mobile API requires that the user is familiar with the following:

- Basic understanding of Web Services (http://www.w3schools.com/webservices),
- Manipulating an XML document (http://www.w3schools.com/xml)
- Mastery of a programming language allowing the use of Web Services.

How to read this documentation?

Description	This documentation is devoted to explain the basic concepts of the SDK: • How to create a request, • How to interpret a response, • The different formats available, • Authentication and data types used.
Reference	To get in detail the capabilities of each API methods, please refer to the “API Reference” document.

Starting Guide

Requests

Description

All requests to the API must be sent to the following base URL

```
https://api.allopass.com/rest
```

API accessibility

The API is accessible via both HTTP and HTTPS.

However, for security reasons, we strongly recommend using the HTTPS protocol in production environments.

Vocabulary

To retrieve information about a product, we use the GET verb and the "product" resource name

```
GET /rest/product
```

Example

For example, to retrieve information about the product identified by 123456, and to delete the same product, the requests would be:

```
GET /rest/product/123456
DELETE /rest/product/123456
```

Note

Later in this documentation, resources (URI: Uniform Resource Identifier) will be named using a path expressed after the base URL.

For example, resource "`/onetime/pricing`" maps to `https://api.allopass.com/rest/onetime/pricing`.

HTTPS

Description

Thus, when we want to create, edit, delete or retrieve a resource, it is useful to specify the action to perform.

This is done implicitly with the verb. The URL is only used to specify what resource is affected.

These four actions: Reading, Creating, Editing and Deleting (CRUD: Create Read Update Delete) are associated with the following HTTP methods (verbs):

HTTP Method	Description
GET	Retrieve information.
POST	Create a new resource.
PUT	Modify a resource.
DELETE	Delete a resource.

Responses

Description

The execution of an action on the server brings the return of formatted output, with an HTTP status code.

HTTPS

Description

The HiPay Mobile API can potentially return the following HTTP status codes:

HTTP Status	Description
200 OK	Everything went well.
201 CREATED	A new resource has been created correctly. For example, a transaction, product, etc.
304 NOT MODIFIED	The content of the response has been cached and has not changed since the last request.
400 BAD REQUEST	One of the parameters of the request is invalid.
401 UNAUTHORIZED	Authentication failed. The API key or signature is invalid.
403 FORBIDDEN	Access to resources is prohibited. For example, this status will be returned when a merchant attempts to retrieve information about a product that they don't own.
404 NOT FOUND	Resource not found.
405 METHOD	The HTTP method (GET, POST, PUT, DELETE) is invalid

HTTP Status	Description
NOT ALLOWED	for the requested resource. For example, requesting the deletion of an account is always a bad idea.
500 SERVER ERROR	An unexpected error occurred. Please report the incident.
503 SERVICE UNAVAILABLE	Access to a resource is temporarily unavailable. The user may have to try again later.

Response format

Responses are formatted in XML or JSON.

Each response consists of

- a numeric status code specific to the API,
- a message describing the status and
- a content described in the selected format (whenever information needs to be retrieved).

Request action

Performed correctly:

➔ the API will always return code "0" (zero) with the message "OK".

Performed with error:

➔ the API returns a status code greater than zero with a message describing the error

(For a list of error codes, see Appendix 1 "Error Codes" in the “API Reference document”).

Example

The following is a XML return example.

XML return

```
<response code="0" message="OK">
[...] CONTENT OF RESPONSE [...]
</response>
```

Character encoding

Important

All data must be encoded in UTF-8 (Unicode).

Verification

Verification of valid UTF-8 is performed. If an invalid sequence is found, it is automatically converted to UTF-8.

DATES

Description	All dates are converted to GMT and the merchants are responsible for the formatting in their preferred time zone.
Formats	Dates are returned by the API in two different formats as follows:

Format	Description
Timestamp	This is a UNIX timestamp, unsigned integer representing the number of seconds since January 1, 1970 GMT.
ISO-8601	The international standard ISO 8601 specifies numeric representations of date and time.

Example

The following is a API return example.

XML API return

```
<date timestamp="1258387836" date="2009-11-
16T16:10:36+00:00" />
```

Example in PHP

To get this date in PHP

```
<?php
date_default_timezone_set('UTC');

// Display the current date (ISO 8601)
print date(DATE_ISO8601);

// Display the date returned by the API
(ISO 8601)
$timestamp=1258387836;
print date(DATE_ISO8601, $timestamp);
```

Available formats

The HiPay Mobile API provides two formats: XML and JSON, each being provided for specific cases of programming.

How to specify the return format?

Return format

It is possible to specify the return format in two ways:

- With an HTTP header,
- With a *format* parameter.

The *format* parameter, if present, overrides the HTTP header.

Examples

Format parameter

Examples of requests with the *format* parameter explicitly stated:

```
http://api.allopass.com/rest/resource?format=json
http://api.allopass.com/rest/resource?format=xml
```

HTTP headers

Examples of HTTP headers to send with the request:

```
Content-type: application/json
Content-type: text/xml
```

XML

All responses are encapsulated in the `<response>` XML markup followed by two attributes *code* and *message* containing the code and status message returned by the API, as described under "Basic Concepts>Responses".

Example

XML Response

```
<?xml version="1.0" encoding="UTF-8" ?>
<response xmlns="https://api.allopass.com/rest"
code="0" message="OK">
<id>123456</id>
<name><![CDATA[PRODUCT NAME]]></name>
<purchase_url><![CDATA[http://localhost/purchase]
]></purchase_url>
<forward_url><![CDATA[http://localhost/product]]>
</forward_url>
</response>
```

JSON

JSON is a data format that reuses elements of JavaScript syntax. This greatly facilitates the use of content in an HTML page. This format is typically used to manipulate data in AJAX.

Example

Description	The JSON responses returned by the API are simple conversions of XML strings to JSON. This format is made available in order to provide greater comfort for AJAX application developers.

JSON Response	{"response": {
	"@attributes": {
	"code": "0",
	"message": "OK"
	},
	"id": "123456",
	"name": "PRODUCT NAME",
	"purchase_url": "http:\/\/localhost\/purchase",
	"forward_url": " http:\/\/localhost\/product"
	}

Merchant Authentication

Overview

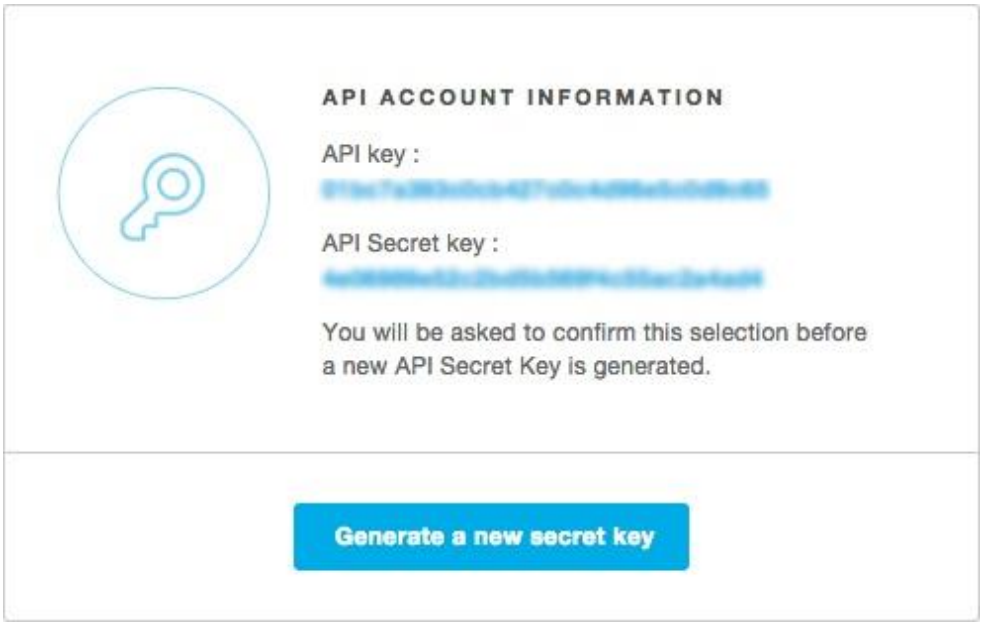
Many HiPay Mobile API methods require merchant authentication.

The authentication mechanism is based on a set of keys and hash functions (MD5 or SHA-1) to "sign" all exchanges with HiPay Mobile and ensure their authenticity.

Get an API key

Description

Merchants must have an API key to use the HiPay Mobile API methods.



This "public" key is unique and identifies each merchant.

In addition to the API key, merchants also have a secret key that allows them to calculate the signature of each request and verify the signature of each response returned by the API.

Keyset security

This keyset is available in your HiPay Mobile back office under the « Toolbox » category.

Merchants may always generate a new secret key clicking on the “Generate a secret key” button.

This key is randomly generated and published immediately.

To secure communication with the API, it is advisable to regularly change the secret key.

Warning

A merchant already using the HiPay Mobile API should be particularly careful in generating a new secret key. The API automatically takes new keys into account and any request is refused until the key has been updated on the applications and merchant sites.

Signatures

Authentication steps

Authentication happens at three steps while using the HiPay Mobile API

Steps	Description
1-Building Requests	Requests that require authentication by the merchant must be signed as explained in "*Make an authenticated request*". Array parameters are included by concatenating them along with their variables in standard URI ampersand format
2-Verification of responses	The response to a signed request is also signed. The API returns a signature calculated from merchant identifiers (API keys). Verifying the validity of the signature confirms that the response comes from HiPay Mobile. This verification is optional and is at the merchant's discretion. (see “Signature Verification”)
3-Verification of notifications	All events and payment notifications sent to a merchant are signed using the merchant's API keys. It is strongly advised to verify the signature and validate that HiPay Mobile is the source of the notification. (see “Signature Verification”)

Make an authenticated request

Description

To make an authenticated request ("sign" a request), merchants must use the following four parameters in URLs:

Status	Parameter	Type	Description
mandatory	api_key	string	Merchant API key
mandatory	api_sig	string	Signature
mandatory	api_ts	integer	UNIX Timestamp (GMT)
optional	api_hash	string	Hash function to use for signing: md5 or sha1. (SHA-1 is default)

Signed request URL

The URL of the signed request is as follows:

```
https://api.allopass.com/rest/resource?api_hash=[hash_function]&
api_ts=[timestamp]&api_key=[key]&api_sig=[signature]
```

Procedure

To create the signature, proceed as follow:

Steps	Action
1	Build the request parameters. `site_id=123456&product_id=654321`
2	Add the authentication settings (except *api_sig*, see step 6). `site_id=123456&product_id=654321&api_key=cfd3b9a6b7b309c06aa53f5527c96e67&api_hash=sha1&api_ts=1258387836`
3	Sort parameter/value pairs in ascending alphabetical order by maintaining the parameter/value association. `api_hash=sha1&api_key=cfd3b9a6b7b309c06aa53f5527c96e67&api_ts=1258387836&product_id=654321&site_id=123456`

Steps	Action
4	Create a string by concatenating the parameters and their values without adding delimiters and add the secret key last. `api_hashsha1api_keycfd3b9a6b7b309c06aa53f5527c96e67api_ts1258387836&product_id654321site_id123456ead9758399359a2bb3b32e240322a11e`
5	Hash the string result with the chosen hash function (in this case: SHA-1). `37d39beae276011bbb9e7d92e8585f9eeae3a42f`

6	Generate URL from step 3 by adding the parameter *api_sig* and its value obtained in step 5.
	
	`https://api.allopass.com/rest/onetime/pricing?api_hash=sha1`
	`&api_key=cfd3b9a6b7b309c06aa53f5527c96e67&api_ts=1258387836&product_id=654321`
	`&site_id=123456&api_sig=37d39beae276011bbb9e7d92e8585f9eeae3a42f`

Example in PHP

```
<?php
// Script to create a signature
define('API_BASE_URL', 'https://api.allopass.com/rest');
define('API_KEY', 'cfd3b9a6b7b309c06aa53f5527c96e67');
define('API_SECRET_KEY', 'ead9758399359a2bb3b32e240322a11e');
define('API_HASH_FUNCTION', 'sha1');

date_default_timezone_set('UTC');

// STEPS 1 and 2: Construction of query parameters
$queryParameters = array(
'site_id'    =>123456,
'product_id' =>654321,
'api_key'    => API_KEY,
'api_hash'   => API_HASH_FUNCTION,
'api_ts'     =>time()
 );

// STEP 3 : Sort parameters by ascending alphabetical order by name of
parameter
ksort($queryParameters);

/* STEP 4
 * Prepare a string to hash
 * with the hash function "API_HASH_FUNCTION"
*/
$stringToHash = '';

foreach ($queryParameters as $parameter =>$value) {
$stringToHash .= $parameter . (is_array($value) ? implode('', $value) :
$value);
 }
$stringToHash .= API_SECRET_KEY;

// STEP 5: Creation of signature
$signature = hash(API_HASH_FUNCTION, $stringToHash);

// STEP 6 : Generating URL
$queryParameters['api_sig'] = $signature;
$url =API_BASE_URL.'/onetime/pricing?'.http_build_query($queryParameters);
```

After payment is received (Forward URL)

Overview

Description	After a successful payment the Forward URL passes GET parameters that you can collect and use to query the */transaction* for verification.

Response

The following table lists the parameters returned to the merchant URL.

Parameter	Description	Example
data	Custom data that was initially passed to the *transaction/prepare* API.	Payment for 5 widgets
code[]	HiPay Mobile access code. If several codes were required for purchase, the list of codes is comma-delimited.	KFD45
transaction_id	HiPay Mobile ID of the transaction	0c92578d-3143-4bd8-aeae-72f2455e2499
merchant_transaction_id	Optional merchant transaction ID initially passed to the *transaction/prepare* API.	ABC123DEF456
DATAS	*Same as data [deprecated and only intended for backward compatibility]*	
RECALL	*Same as code [deprecated and only intended for backward compatibility]*	
codes[]	*Same as code [deprecated and only intended for backward compatibility]*	
trxid	*Same as transaction_id [deprecated and only intended for backward compatibility]*	

Notification of payment (notification URL)

Overview

Description	When the *url_notification* parameter is set in the product or in the transaction (*transaction/prepare*), then merchants are registered to receive acknowledgments of their transactions on a URL placed on their server.

Response

The following table lists the parameters returned to the merchant URL.

Parameter	Description	Example
action	Describes the type of event for which we notify (always *payment-confirm* for payments received)	payment-confirm
test	Allows merchant to identify test transactions.	*true* or *false*
transaction_id	Unique ID for the HiPay Mobile transaction.	0c92578d-3143-4bd8-aeae-72f2455e2499
status	Status of the transaction.	0
status_description	Description of the status of the transaction.	Success
access_type	Product type	onetime-dynamic
date	Transaction date	2010-12-15T16:09:57+00:00
code	HiPay Mobile code used	XXXXXXXX
pricepoint_id	Price point Id used	206
type	Payment method	premium-sms
data	Custom merchant data provided when making payment request.	Widget592
merchant_transaction_id	The transaction ID used by the merchant.	IN1237123
amount	Amount of the transaction. The *currency* is determined by the price	10.00

Parameter	Description	Example
	point identifier.	
paid	Amount actually paid by the customer in *currency*.	10.00
currency	Local currency for the transaction.	EUR
payout_amount	Amount of the merchant payout in payout currency	6.18
payout_currency	Payout currency for the transaction	EUR
reference_currency	Base currency (EUR is default).	USD
reference_amount	Amount of the transaction in base currency	14.79
reference_paid	Amount actually paid by the customer in base currency	14.79
reference_payout	Amount of the merchant payout in base currency	9.14
customer_country	Country code of the customer. This two-letter code complies with ISO-3166.	FR
site_id	Identifier of the merchant site.	123456
product_name	The name of the product where the code was used	My New Product
api_key	Merchant API key	cfd3b9a6b7b309c06aa53f5527c96e67
api_ts	UNIX Timestamp (GMT)	1258691527
api_hash	Hash function to use for signing: *md5* or *sha1*. (SHA-1 is default)	sha1 md5
api_sig	Signature	fb1bab50fd2c3751dab07b35…

Example

URL notification Example

> http://your-domain.com/allopass_notification.php?action=payment-confirm
> &transaction_id=0c92578d-3143-4bd8-aeae-72f2455e2499&status=0&status_description=success&data=
> &merchant_transaction_id=&amount=10.00&paid=10.00&currency=EUR&reference_currency=USD&reference_amount=14.79
> &reference_paid=14.79&reference_payout=9.14&payout_currency=EUR&payout_amount=6.18&customer_country=FR
> &site_id=123456&api_hash=sha1&api_ts=1258691527&api_key=cfd3b9a6b7b309c06aa53f5527c96e67&api_sig=1c90d5846d16f7f9fede3ff3d6769193fe5b0d1a

Status codes

Code	Description	Example
0	Success	Payment accepted

Notification Response (notification URL)

Overview

Description	Merchants may return an XML response to HiPay Mobile so as to explicitly acknowledge that the notification was successfully processed.
Notification response	The Notification Response is optional but recommended. In the absence of a response from the merchant, HiPay Mobile will interpret an HTTP 200 return code as a success. A response status of 0 is a failure, 1 is a success. After a failure, HiPay Mobile will attempt 4 more times.

Parameters

The following table lists the parameters utilized in code. They are only available for access through HiPay Mobile support.

Parameter	Description	Example
code	General purpose field	123
message	General purpose field	OK

Sample Response

URL notification Example

```
<?xml version="1.0" encoding="UTF-8"?>
<response status="1">
<code>123</code>
 <message>OK</message>
</response>
```

Signature verification

Check the signature of a response from the API

Description

The response to a signed request is also signed. This signature is returned in an HTTP header named “X-Allopass-Response-Signature”.

Example

X-Allopass-Response-Signature:
40b9f7897452a0d4494e42025b10fe31001f1f83

Procedure

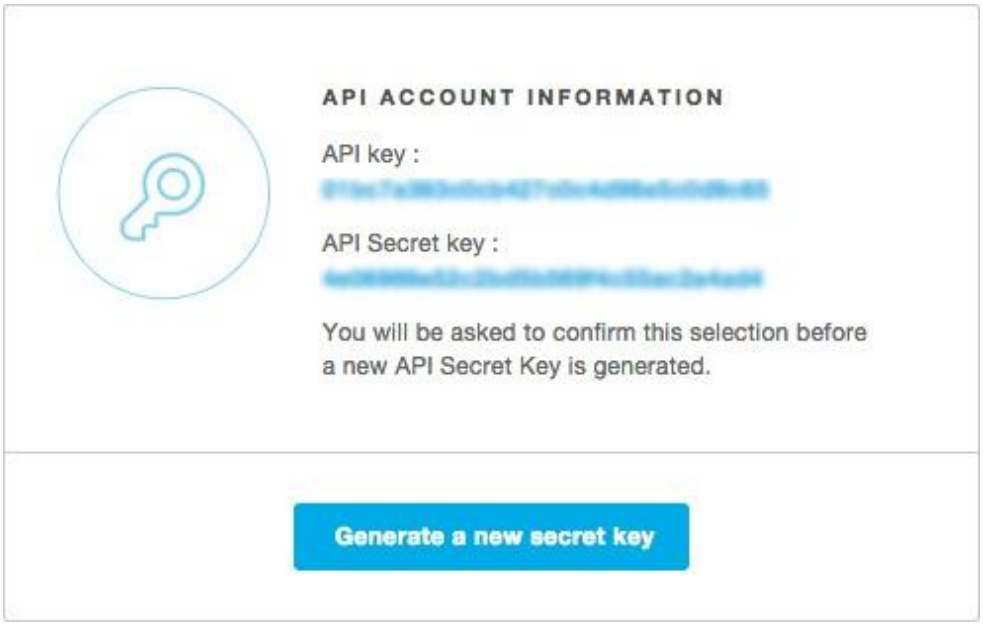
To verify the signature, proceed as follow:

Steps	Action
1	Get the body of the response (XML or JSON). <?xml version="1.0" encoding="UTF-8" ?> <response xmlns="https://api.allopass.com/rest" code="0" message="OK"> <id>123456</id> <name><![CDATA[PRODUCT NAME]]></name> <purchase_url><![CDATA[http://localhost/purchase]]></purchase_url> <forward_url><![CDATA[http://localhost/product]]></forward_url> </response>
2	Create a string by concatenating the body of the response and the secret key. <?xml version="1.0" encoding="UTF-8" ?> <response xmlns="https://api.allopass.com/rest" code="0" message="OK"> <id>123456</id> <name><![CDATA[PRODUCT NAME]]></name> <purchase_url><![CDATA[http://localhost/purchase]]></purchase_url> <forward_url><![CDATA[http://localhost/product]]></forward_url> </response>ead9758399359a2bb3b32e240322a11e
3	Hash the string result with the chosen hash function (in this case: SHA-1) 61434688f14cfdab252f2bf07d14f4ca39d30ff0
4	Verify that the signature obtained in step 3 is equal to the signature returned in the HTTP header “ X-Allopass-Response-Signature”.

Example in PHP

```php
<?php
/* Using the PHP example script in the category
 * "Merchant Authentication > Make an authenticated request", and assuming
that
 * the URL of the resource request is placed in the "$url" variable  */
$url = API_BASE_URL . '/onetime/pricing?' . http_build_query($parameters);

 $sock = curl_init($url);
 curl_setopt_array($sock, array(
    CURLOPT_HEADER         =>true,
    CURLOPT_RETURNTRANSFER =>true,
    CURLOPT_FOLLOWLOCATION =>false,
    CURLOPT_CONNECTTIMEOUT =>10,
    CURLOPT_LOW_SPEED_TIME =>10,
    CURLOPT_TIMEOUT        =>10
 ));
 $response = curl_exec($sock);

 if (0< ($curlErrno = curl_errno($sock))) {
trigger_error("CURL Error ($curlErrno): " . curl_error($sock),
E_USER_NOTICE);
     header('Location: /error/unavailable.php');
exit();
 }
 $httpStatusCode = curl_getinfo($sock, CURLINFO_HTTP_CODE);
 $httpHeaderSize = curl_getinfo($sock, CURLINFO_HEADER_SIZE);
curl_close($sock);

// Read the API response returned in the $response variable by the
curl_exec () function
$responseHeaders = array();
 $rawHeaders    = substr($response, 0, $httpHeaderSize - 4);
$responseBody = substr($response, $httpHeaderSize);

/* Build an associative array from HTTP headers
 * returned by the Allopass API. For example:
 *   Content-Type: text/xml
 *   X-Allopass-Response-Signature:
61434688f14cfdab252f2bf07d14f4ca39d30ff0
 *
 * becomes:
*   Array(
 *       'Content-Type'                  => 'text/xml',
 *       'X-Allopass-Response-Signature' =>
'61434688f14cfdab252f2bf07d14f4ca39d30ff0'
 *   )
*/
 foreach (explode("\r\n", $rawHeaders) as $header) {
list($name, $value) = explode(':', $header);
$responseHeaders[$name] = $value;
 }

 if (isset($responseHeaders['X-Allopass-Response-Signature'])) {
// STEPS 2 and 3: Calculation of the signature
$returnedResponseSignature = $responseHeaders['X-Allopass-Response-
Signature'];
$computedResponseSignature = hash(API_HASH_FUNCTION, $responseBody .
API_SECRET_KEY);

// STEP 4: Checking the signature returned by the API
if (trim($returnedResponseSignature) != trim($computedResponseSignature))
{
header('Location: /error/forbidden.php');
exit();
     }

}
```

Check the signature of a notification

Description	A unique signature is sent (api_sig) each time that HiPay Mobile contact a merchant page.
Verification of the signature	To verify this signature, you will need your API Secret Key.
Find the Secret Key	You will find this key in your HiPay Mobile back office under the « Toolbox » category. 

Signature verification example

Example in PHP

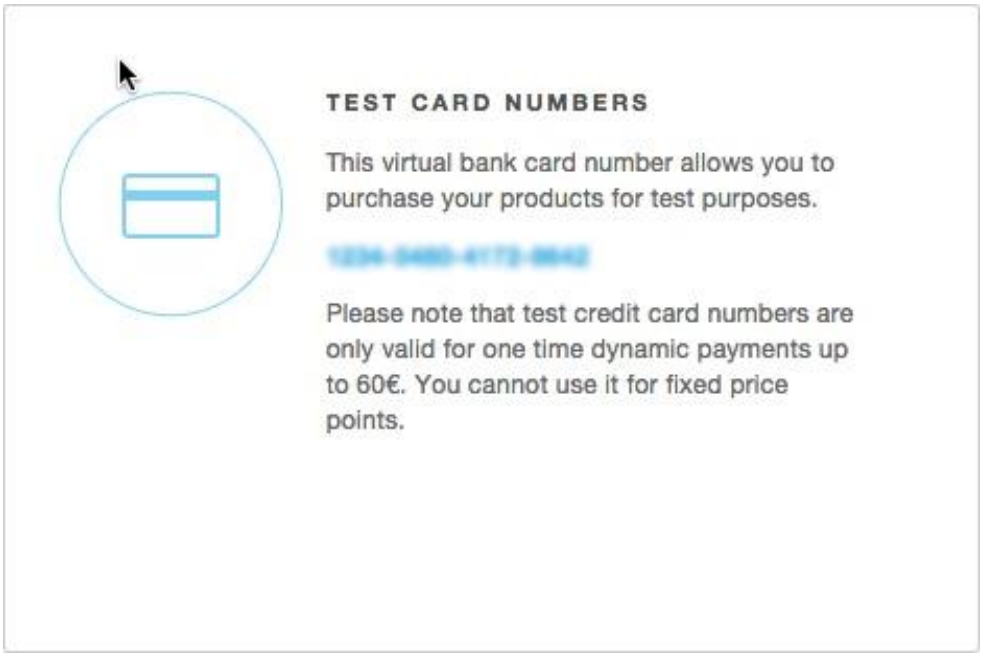
```php
<?php
$parameters = $_GET;

$signature = $parameters['api_sig'];
unset($parameters['api_sig']);
ksort($parameters);

$secretKey      = ''; // fill here with your personal secret key
$string2compute = '';

foreach ($parameters as $name => $value) {
    $string2compute .= $name . $value;
}

// true if OK, false if not
// if you are using md5 instead of sha1 please replace
if (sha1($string2compute . $secretKey) == $signature) {
    $code    = 0;
    $message = 'OK';
}
else {
    $code    = 1;
    $message = 'KO';
}
```

```
header('Content-Type: text/xml; charset=UTF-8');

echo '<?xml version="1.0" encoding="UTF-8" ?>';
?>
<response status="1">
 <code><?php echo $code; ?></code>
 <message><?php echo $message; ?></message>
</response>
```

Testing

Virtual bank card

Description	HiPay Mobile lets you use a virtual bank card to test your payments; you will find it in your HiPay Mobile back office under the « Toolbox » category:



This virtual bank card number allows you to purchase YOUR products for test purposes.

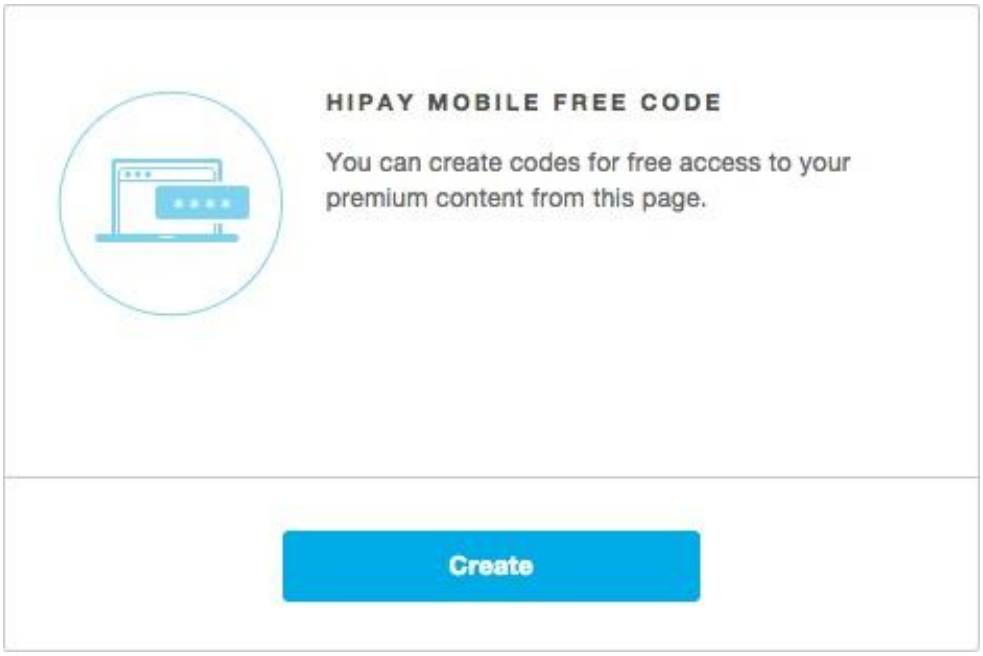
Note	**Test credit card number is only valid for discrete price points (payments up to 60€). You *cannot* use it for fixed price points.**

Test codes

Description	To facilitate the testing of your integration, for each product set up on the HiPay Mobile merchant account, you can specify a test code that will always lead to an unbilled, successful transaction.
Code security	**It is your responsibility to use strong enough test codes that won't be easily guessed or brute-forced by the end-user.**
Note	The ***Test codes*** don't send any notification to your notification URL, if you want to test the notifications, please test them with ***Free Codes.***

Free codes (only for One-Time Fixed products)

Description	Free codes are more sophisticated than test codes as they are meant to behave more like real codes. Their validity can be adjusted by duration or number of uses, so they can be used for customer support as well.
Free code	You can generate free codes on your HiPay Mobile back office under the « Toolbox » category:



Note	These free codes are tied to specific product IDs. When using the API to generate transactions that are not associated with product IDs, you need to select "*Generic Product*" in the drop-down list.

More questions ?

Contact our Merchant support at contact.mobile@hipay.com.