

Forum:[Hack Forums](#)

Thread: [Using the paymentwall API for Virtual Currency - UP-2-DATE API](#)

hellomen - February 27, 2015, 2:42 pm

Important Message: As there was an issue with posting it before as I timed out, I requested to remove both threads so I can re-post it correctly.

This is the correct thread.

Thank you.

..:PART OF THIS SOURCE IS FINDABLE IN THE API OF PAYMENTWALL.COM!:

Hello, this is an update thread of this [this](#) thread

First of all, thanks for stopping by to this thread.

I have had some people requestig to make a new thread with the updated version.

Just to make clear you will need to have the contents out of the LIB directory from [here](#).

place the lib directory with all it's contents on a location on your server.

as this update is here, it's a lot eassier to handle the pingback.

This is a almost straight copy paste from the API!:

Spoiler ([Click to View](#))

PHP Code

```
<?php
```

```
require_once('/path/to/paymentwall-php/lib/paymentwall.php');
Paymentwall_Base::setApiType(Paymentwall_Base::API_VC);
Paymentwall_Base::setAppKey('YOUR_SECRET_KEY');
// available in your Paymentwall merchant area
Paymentwall_Base::setSecretKey('YOUR_SECRET_KEY');
// available in your Paymentwall merchant area
```

```
$pingback = new Paymentwall_Pingback($_GET, $_SERVER['REMOTE_ADDR']);
```

```
if ($pingback->validate()) {
    $virtualCurrency = $pingback->getVirtualCurrencyAmount();
    $userid = $pingback->getUserId();
```

```
    // make a db connection
```

```
    if ($pingback->isDeliverable()) {
        // deliver the virtual currency
    } else if ($pingback->isCancelable()) {
        // withdraw the virtual currency
    }
}
```

```
    echo 'OK'; // Paymentwall expects response to be OK, otherwise the pingback will be resent
} else {
    echo $pingback->getErrorSummary();
}
```

// close db connection

I have added PHP Code

```
<?php
```

```
$userid = $pingback->getUserId();
```

and PHP Code

```
<?php
```

```
// write your db connection part here
```

and PHP Code

```
<?php
```

```
// close db connection
```

as we would like to know who we need to give/take the points from :-)

it's not that hard ?

now replace YOUR_SECRET_KEY with your secret key which can be found on your merchant area

In order to that you MUST create a "project" in your merchant area.
once you've done that you see this:

Shoot on that show button ? and TADAAA ? there is your secret key ?

Isn't that fancy? :-)

Let's go to the actual POINTS part now.

Assuming you have a table called users with a row called points which has it's datatype set to one of the int (int/bigint) preferred

it would be easy.

under:

PHP Code

```
<?php
```

```
// write your db connection part here
```

you add:

Spoiler ([Click to View](#))

PHP Code

```
<?php
```

```
$db = new PDO(
```

```
"mysql:host=YOUR_DB_HOST_NAME;port=YOUR_DB_PORT;dbname=YOUR_DB_NAME",
YOUR_DB_USERNAME, YOUR_DB_PASSWORD);
```

replace:

YOUR_DB_HOST_NAME :to: your actual DB Host Name {default: localhost}
YOUR_DB_PORT :to: your actual DB Port Number {default: 3306}
YOUR_DB_NAME :to: your actual DB Name
YOUR_DB_USERNAME :to: your actual DB Username
YOUR_DB_PASSWORD :to: your actual DB Password

This will create your connection ?

As PDO is OOP only we use -> to call something.

as we all know in OOP:

calling methods would be ->method_name();

calling PUBLIC variables would be ->\$variable_name;

Now add the following:

PHP Code

```
<?php
```

```
$db->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);
```

```
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

Both explains it selves note that ATTR stands for Attribute :-)

Now the real thing is coming together.

We will do the calculations inside on mysql as it works fine :-)

under:

PHP Code

```
<?php
```

```
// deliver the virtual currency
```

you add:

Spoiler ([Click to View](#))

PHP Code

```
<?php
```

```
$query = "UPDATE test SET points = (points + :points) WHERE id = :id;";
```

```
$prepared = $db->prepare($query);
```

```
$prepared->bindParam(':points', $virtualCurrency);
```

```
$prepared->bindParam(':id', $userid);
```

```
$prepared->execute();
```

for taking it away it's the same but then instead of (points + :points) we do (points - :points)

so to take away the points

do the following:

under:

PHP Code

```
<?php
```

```
// withdraw the virtual currency
```

you add:

Spoiler ([Click to View](#))

PHP Code

```
<?php
```

```
$query = "UPDATE test SET points = (points - :points) WHERE id = :id;";
```

```
$prepared = $db->prepare($query);
```

```
$prepared->bindParam(':points', $virtualCurrency);
```

```
$prepared->bindParam(':id', $userid);
```

```
$prepared->execute();
```

if bindParam(); doesn't work for you.

please use the following instead:

PHP Code

```
<?php
```

```
$bindings = Array(':points' => $virtualCurrency, ':id' => $userid);
```

```
$prepared->execute($bindings);
```

last but not least.

under:

PHP Code

```
<?php
```

```
// close db connection
```

You add:

Spoiler ([Click to View](#))

PHP Code

```
<?php
```

```
$db = NULL;
```

```
unset($db);
```

I do both just to make sure and because I am lame ?

Now test the pingback and see if it works ?

if it works, submit your project for a review and let it be accepted.

once you have it accepted you can sell your virtual currency ??

Please note:

I have written this without testing, I am sure the sql query will work but not if the Paymentwall api [I assume it does as it's their api]

If it works, I hope you can confirm this :-)

If you have questions, feel free to ask them.

If you have suggestions, feel free to post them.

I hope you will have a nice day AND have fun with all of this ?

Yours,
Mike M. ???