Paypal, the PHP approach

By Joel De Gan

June 3, 2003

Intended audience

This tutorial is intended for PHP programmers of all levels, who wish to add Paypal functionality to their website. The programmer needs only a PHP enabled website and Paypal account.

Overview

This tutorial offers some solutions to issues common when using Paypal for website instant payments.

This tutorial expects that you have a firm grasp of PHP and MySQL. Aside from functions explained in this article, all functions used are standard for a typical PHP compile with MySQL, and so the examples given should work on any Apache/PHP host.

Learning Objectives

In this tutorial you will create a website payment gateway for content. In this example we will assume that you are selling access to a portion of a website. We will create a database to hold our login information and the payment data that Paypal returns. We will cover activation of an account within a very small system, which is a trimmed down example. To keep this tutorial brief we will not cover Paypal subscriptions or auction payments.

Background Information

Payment solutions for websites can be difficult and costly for micro payments and for new commercial websites. Many systems have been developed in an attempt to solve this problem, ranging from full-blown banks to small credit trading systems. Undeniably Paypal have emerged as the world leader in this market. Their system is known as safe and efficient for website owners, auction sellers, and buyers.

When choosing Paypal with PHP, whether for cost reasons or for simplicity and ease of use, you must keep in mind that PHP does not support Paypal in functions as it does other payment processors such as PayflowPro.

This is because there is no API for us to develop these functions for. All functions for dealing with Paypal through PHP are done via the Paypal website, emails and a system that Paypal has developed called IPN. IPN stands for “Instant Purchase Notification” and will make a HTTP post of all the information we could need about a purchase to a web page that you specify on the Paypal website.

Prerequisites

In order to set up your website to accept Paypal payments you yourself need a Paypal account in good standing. You will also need to make sure that you change the IPN information for your Paypal account to post IPN data to your website. For this example, if your website is http://example.com we would set the IPN post to go to http://example.com/members/ipn.php.

How it Works

To start we will need to clarify our goals.

Let’s assume we have a site which has a members only section and we would like to charge access to this area. We could be offering downloads, latest digital artwork, templates or simply advanced features to a busy bulletin board system.

Putting it all together

First, create a database (and tables) that will house information on users, and another database to keep records of what Paypal has sent you.

Create a database named “paypal_tutorial” and in it create the following table.

```
CREATE TABLE users (
    uid bigint(20) NOT NULL auto_increment,
    username varchar(30) NOT NULL default '',
    password varchar(30) NOT NULL default '',
    created timestamp(14) NOT NULL,
    paid enum('N','Y') NOT NULL default 'N',
    PRIMARY KEY  (uid),
    UNIQUE KEY username (username)
) TYPE=MyISAM COMMENT='Store user paid y/n information';
```

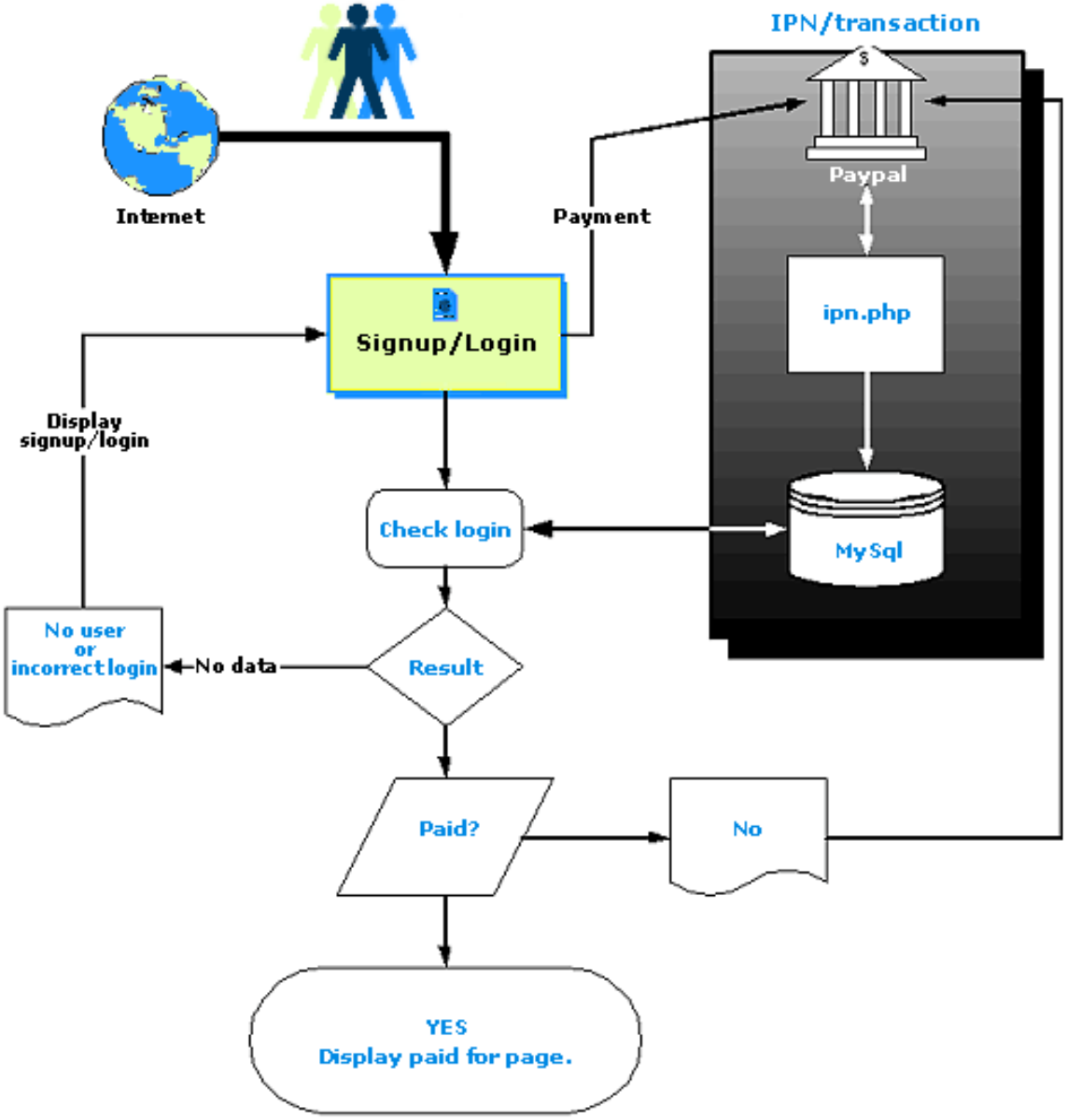
Now make a table which stores data from Paypal, for reference and for future reports. This table will hold all the information you need about each post from Paypal.

```
CREATE TABLE accounting_paypal (
    uid bigint(20) NOT NULL auto_increment,
    date timestamp(14) NOT NULL,
    item_name varchar(130) NOT NULL default 'joi',
    receiver_email varchar(125) default NULL,
    item_number varchar(130) NOT NULL default '0',
    quantity smallint(6) NOT NULL default '0',
    invoice varchar(25) NOT NULL default '0',
    custom varchar(60) default NULL,
    payment_status set('Completed','Pending','Failed','Denied') NOT NULL default
'Failed',
```

```
    pending_reason set
('echeck','intl','verify','address','upgrade','unilateral','other') NOT NULL
default 'other',
    payment_gross float NOT NULL default '0',
    payment_fee float NOT NULL default '0',
    payment_type set('echeck','instant') NOT NULL default 'instant',
    payment_date varchar(50) NOT NULL default '0',
    txn_id varchar(20) NOT NULL default '0',
    payer_email varchar(125) default NULL,
    payer_status set('verified','unverified','intl_verified') NOT NULL default
'unverified',
    txn_type set
('web_accept','cart','send_money','subscr_signup','subscr_cancel','subscr_failed',
'subscr_payment','subscr_eot') NOT NULL default 'subscr_payment',
    first_name varchar(35) default NULL,
    last_name varchar(60) default NULL,
    address_city varchar(60) default NULL,
    address_street varchar(60) default NULL,
    address_state varchar(60) default NULL,
    address_zip varchar(15) default NULL,
    address_country varchar(60) default NULL,
    address_status set('confirmed','unconfirmed') NOT NULL default 'unconfirmed',
    subscr_date varchar(50) NOT NULL default '0',
    period1 varchar(20) NOT NULL default 'UNK',
    period2 varchar(20) NOT NULL default 'UNK',
    period3 varchar(20) NOT NULL default 'UNK',
    amount1 float NOT NULL default '0',
    amount2 float NOT NULL default '0',
    amount3 float NOT NULL default '0',
    recurring tinyint(4) NOT NULL default '1',
    reattempt tinyint(4) NOT NULL default '0',
    retry_at varchar(50) default NULL,
    recur_times smallint(6) NOT NULL default '0',
    username varchar(25) default NULL,
    password varchar(20) default NULL,
    subscr_id varchar(20) default NULL,
    entirepost text,
    paypal_verified set('VERIFIED','INVALID') NOT NULL default 'INVALID',
    verify_sign varchar(125) default NULL,
    PRIMARY KEY  (uid),
    KEY txn_type (txn_type),
    KEY payment_status (payment_status),
    KEY pending_reason (pending_reason),
    KEY payer_status (payer_status),
    KEY payment_type (payment_type),
    KEY retry_at (retry_at),
    KEY receiver_email (receiver_email),
    KEY date (date)
) TYPE=MyISAM COMMENT='Recieve posts FROM paypal servers';
```

Now the tables are ready, let’s dive in and examine what you need to do in order for this system to work.

The following diagram shows the basic system flow.



You need to have a directory that is protected from unauthorized access, you need to collect payments, and you need to change the “paid” status in the users table to “Y” once payment is received.

Let’s assume the directory in question is called “members”. In your members directory, create a blank file called ipn.php and another named index.php.

The Script

Create a simple script at the top of index.php that will not control data display based on the user’s “paid” status in the system, it should look like the following, and follows the logic laid out in the diagram above.

```
<?php
### LISTING OF index.php
### first some definitions we will be using.
define ("DBHOST", "localhost");
define ("DBNAME", "paypal_tutorial");
```

```php
define ("DBUSER", "root");
define ("DBPASS", "");

define("PAYPAL_USER", "you@youremail.com");
define("PPLINK", "https://www.paypal.com/xclick/business=".
    PAYPAL_USER.
    "&item_name=members_payment&item_number=1".
    "&amount=10.00&no_note=1&currency_code=USD");

// our login form for user logins
$SHOW_LOGIN_FORM = <<<ENDFORM
    <br /><br />
    <center><form method='post' action='$PHP_SELF'><table>
    <tr>
        <td>Username: </td>
        <td><input name='username' type='text' value=''></td>
    </tr>
    <tr>
        <td>Password: </td>
        <td><input name='password' type='password' value=''></td>
    </tr>
    <tr>
        <td colspan='2' align='center'>
            <input type='submit' value='log in'>
        </td>
    </tr>
    </table>
    </form></center>
ENDFORM;

// a function to handle setting cookies.
function sec_setcookie($var, $val, $modify=3600)
{
    $exp = gmstrftime("%A, %d-%b-%Y %H:%M:%S", time() + $modify);
    $dom = $GLOBALS["HTTP_HOST"];
    if (preg_match("/^(.*):(.*)$/", $dom, $arr)) {
           print_r($arr);
        $dom = $arr[1];
    }
    $parts = explode(".", $dom);
    $dom = ".". $parts[count($parts)-2]. ".". $parts[count($parts) - 1];
    setcookie($var, $val, time() + $modify,"/", $dom, 0);
    ${$var} = $val;

    global ${$var};
} //end function

### CONNECT TO THE DATABASE
function DatabaseConnect()
{
    if (!($mylink = mysql_connect(DBHOST, DBUSER, DBPASS))) {
```

```
        echo mysql_error();
        exit;
    } //fi
    mysql_select_db(DBNAME) or die(mysql_error());
} // end function
DatabaseConnect(); // this will automatically connect us


### NOW THE LOGIC
// first see if we have a post
if ($HTTP_POST_VARS['username'] && $HTTP_POST_VARS['password']) {
    $sql = "
        SELECT *
        FROM users
        WHERE username = '$username'
            AND password = '$password'
    ";
    $result  = mysql_db_query(DBNAME, $sql);

    if (mysql_num_rows($result) > 0) {
        $info = mysql_fetch_assoc($result);
        if ($info[paid] == "Y") {
            sec_setcookie("username", $username);
            sec_setcookie("password", $password);
        } else {
            echo "<center><font color=red><b>ERROR, ACCOUNT NOT PAID</b></
font><br>
            <a href=".PPLINK.">CLICK HERE</a> to pay for service.</center>";
            die();
        } //fi
    } else {
        sec_setcookie("count", $count + 1);
        echo "<center><font color=red><b>ERROR IN LOGIN - SIGN UP FOR AN ACCOUNT
FIRST</b></font></center>";
        if ($count > 3) {
            echo "<center><font color=red><b>TOO MANY ATTEMPTS, TRY LATER</b></
font></center>";
        } else {
            echo SHOW_LOGIN_FORM;
        } //fi
        die();
    } //fi
} //fi

if($_COOKIE['username'] && $_COOKIE['password']) {
    $sql = "
        SELECT *
        FROM users
        WHERE username = '$username'
            AND password = '$password'
    ";
    $result  = mysql_db_query(DBNAME, $sql);
```

```
    if (mysql_num_rows($result) == 0) {
        # clear the cookies
        sec_setcookie("username", "");
        sec_setcookie("password", "");
        echo SHOW_LOGIN_FORM;
        die();
    } //fi
} else {
    echo SHOW_LOGIN_FORM;
    die();
} //fi
?>
HERE IS THE PAID FOR PAGE.
```

The above script makes some assumptions: one is that the username is the user's email address, another is that this email is the same email address used in their Paypal account. The listing above is an extremely trimmed down version of a way to protect a page based on the "paid" status in a database.

The script ipn.php to where Paypal posts IPN data should look like the following:

```
<?php
### LISTING OF ipn.php
define ("DBHOST", "localhost");
define ("DBNAME", "paypal_tutorial");
define ("DBUSER", "root");
define ("DBPASS", "");

### CONNECT TO THE DATABASE
function DatabaseConnect() {
    if (!($mylink = mysql_connect(DBHOST, DBUSER, DBPASS))) {
        echo mysql_error();
        exit;
    } //fi
    mysql_select_db(DBNAME) or die(mysql_error());
} // end function

DatabaseConnect(); // this will automatically connect us

// below supported vals that paypal posts to us, this list is exhaustive.. but
// without notify_version and verify_sign NOTE: if in is not in this array, it
// is not going in the database.

$paypal_vals = array("item_name", "receiver_email", "item_number",
    "invoice", "quantity", "custom", "payment_status",
    "pending_reason", "payment_date", "payment_gross", "payment_fee",
    "txn_id", "txn_type", "first_name", "last_name", "address_street",
    "address_city", "address_state", "address_zip", "address_country",
    "address_status", "payer_email", "payer_status", "payment_type",
    "subscr_date", "period1", "period2", "period3", "amount1",
    "amount2", "amount3", "recurring", "reattempt", "retry_at",
    "recur_times", "username", "password", "subscr_id", "option_name1",
```

```
    "option_selection1", "option_name2", "option_selection2",
    "num_cart_items"
);

// build insert statement
while (list ($key, $value) = each ($HTTP_POST_VARS)) {
    if (in_array ($key, $paypal_vals)) {
        if (is_numeric($value)) {
            $addtosql .= " $key=$value,";
        } else {
            $newval = urlencode($value);
            $topost .= "&$key=$newval"; //used later in reposting
            $value = addslashes($value);
            $addtosql .= " $key='$value',";
        } //fi
    } //fi
    $entirepost .= "[$key]='$value',";
} //wend

$entirepost = addslashes($entirepost); // just in case..

$addtosql = substr("$addtosql", 0, -1).";"; //chop trailing "," replace with ";"

$sql1 = "
    INSERT INTO accounting_paypal
    SET date=now(), entirepost='$entirepost',". $addtosql;
mysql_db_query(DBNAME, $sql1) or die($sql1);

// We could use this in a log, or to track which users have which payment.
$paypal_id = mysql_insert_id();

if ($HTTP_POST_VARS['payment_status'] == "Completed"
    || $HTTP_POST_VARS['payment_status'] == "Pending")
{
    $username = $HTTP_POST_VARS['payer_email'];
    $sql = "
        UPDATE user
        SET paid = 'Y'
        WHERE username = '$username'
    ";
    $result = mysql_db_query(DBNAME, $sql) or die($sql);
} //fi
### END LISTING OF ipn.php
?>
```

One great thing you can do with the above ipn.php script is that when you pass the item number on the Paypal URL you can also pass in your system's ID number for the user. You should also verify the price of the service being paid for, since Paypal end users can easily change this value. (I will leave those concepts for the reader to explore.)

Now let's see the scripts in action. First open up a web browser to the "members" directory on your web server and try entering in a random username and password., The request will be rejected. Next use phpMyAdmin (or go directly in via MySQL) and add a username (email) and password into the database "users" table, and then attempt to log in as the new

user. This time you will see the “not paid” error.

Go back to the MySQL table, change your user’s “paid” value to “Y”, and then log in again. This time you will see the lower portion of the HTML where you would place your protected page. You are now logged into your paid page and are viewing what a paid user would see.

In the ipn.php script you will notice that we always keep a log of the entire post that Paypal sends to us. This is very useful in creating logs for viewing, and generating payment history for users.

The entire set of variables passed back from Paypal is an ever-growing set that unfortunately cannot all be referenced in this article.

Conclusion

I hope that this tutorial will give you a very good start for setting up your Paypal enabled website. There are many types of payment systems that Paypal supports, so the sky is the limit for development.

As you can see from the above scripts, integrating Paypal is a simple process that you can use to sell services easily from your web page. Obviously the scripts should be added upon. Obviously in their current state the scripts are not a complete application, and I think a user would be justifiably disappointed if he paid ten dollars and got in return nothing more than the words “SECRET PAGE”.

I have often seen questions in message boards about sending Paypal payments using PHP. The following is a function I wrote that does just that. It uses the Paypal WAP interface (cellular telephone), which is a minimal version of Paypal for use in handheld devices.

```
<?php
## this uses a file on disk called "cook" which is
## used for temporary storage of session variables.

define("PAYPAL_EMAIL", "YOUR@PAYPAL.COM");
define("PAYPAL_PASSW", "your-password-here");
define("DEBUG", 1);

if (file_exists('cook')) {
    unlink('cook');
}

function GetCurlPage($pageSpec)
{
    $agent = "up.b";
    $header = array("Accept: text/vnd.wap.wml,*.*");
    $ch = curl_init($pageSpec);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
    curl_setopt($ch, CURLOPT_USERAGENT,      $agent);
    curl_setopt($ch, CURLOPT_HTTPHEADER,     $header);
    curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
    curl_setopt($ch, CURLOPT_COOKIEJAR,      "cook");
    curl_setopt($ch, CURLOPT_COOKIEFILE,     "cook");
    $tmp = curl_exec($ch);
    curl_close($ch);
```

```
    return $tmp;
}

function PostCurlPage ($pageSpec, $data)
{
    $agent = "up.b";
    $header = array("Accept: text/vnd.wap.wml,*.*");
    $ch = curl_init($pageSpec);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
    curl_setopt($ch, CURLOPT_USERAGENT,      $agent);
    curl_setopt($ch, CURLOPT_HTTPHEADER,     $header);
    curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
    curl_setopt($ch, CURLOPT_COOKIEJAR,      "cook");
    curl_setopt($ch, CURLOPT_COOKIEFILE,     "cook");
    curl_setopt($ch, CURLOPT_POST,           1);
    curl_setopt($ch, CURLOPT_POSTFIELDS,     $data);

    $tmp = curl_exec ($ch);
    curl_close ($ch);
    return $tmp;
}

/*
Here is a function to send money to a person via your paypal account.
you will need to have you paypal email and password defined.
  example:
    sending $4.55 to some@body.com
    $send = sendPaypalMoney("some@body.com", 4, 55);
*/
function sendPaypalMoney($email, $dollars, $cents)
{
    $first = GetCurlPage("https://www.paypal.com/cgi-bin/phscr?cmd=login");
       $exp = explode("=", $first);
       $id = explode("\"", $exp[16]);
       $second = PostCurlPage("https://www.paypal.com/cgi-bin/phscr?rs=". $id[0],
        "email=". PAYPAL_EMAIL.
        "&pass=". PAYPAL_PASSW.
        "&cmd=login-submit-pass");
       if (DEBUG == 1) {
        echo $second ."\n";
    }

       $out = explode("<", $second);
       $b1 = explode(":", $out[5]);

       if (DEBUG == 1) {
        echo "Balance: ". $b1[1] ."\n";
    }

       $send1 = explode("\"", $out[8]);
       $send2 = explode("?", $send1[1]);
```

```
       $send3 = explode("\"", $out[10]);
       $sendvals = "cmd=beam-email&auth=".$send3[3];
    $sendpage = PostCurlPage("https://www.paypal.com/cgi-bin/phscr?rs=".
        $send2[1], $sendvals);

       $sendem = explode("<", $sendpage);
    # snag the rs= value for posting
    $rspart = explode("\"", $sendem[12]);
    $rsportion = explode("=", $rspart[1]);
    $rsval = $rsportion[1];

    # snag the auth value for posting
    $authpart = explode("\"", $sendem[17]);
    $authval = $authpart[3];

       $buildpost = "cmd=beam-email-confirm&email=$email&dollars=$dollars".
        "&cents=$cents&auth=$authval";
       $sendpage = PostCurlPage("https://www.paypal.com/cgi-bin/phscr?rs=".
$rsval,
        $buildpost);

       if (DEBUG == 1) {
        echo $sendpage ."\n";
    }

       $sendem = explode("<", $sendpage);
    # snag the rs= value for posting
    $rspart = explode("\"", $sendem[9]);
    $rsportion = explode("=", $rspart[1]);
    $rsval = $rsportion[1];

    # snag the auth value for posting
    $authpart = explode("\"", $sendem[13]);
    $authval = $authpart[3];

       $buildpost = "cmd=beam-email-submit&email=$email&dollars=$dollars".
        "&cents=$cents&auth=$authval";
       $sendpage = PostCurlPage("https://www.paypal.com/cgi-bin/phscr?rs=".
$rsval,
        $buildpost);

       if (DEBUG == 1) {
        echo $sendpage ."\n";
    }

       if (substr_count($sendpage, "Successful") > 0) {
        return true;
       } else {
        if (DEBUG == 1) {
            echo "FAILED: ". $sendpage ."\n";
        }
```

```
        return false;
    }
}// end function
?>
```

Links

The main Paypal website: http://www.paypal.com

A testing script to test posts to your ipn.php: http://www.paypal.com/cgi-bin/webscr?cmd=p/pdn/3p-solutions-ipntools-outside

About the author

Joel De Gan is a seasoned developer currently working for a high-profile domain name registrar. He is obsessed with PHP and in addition to working with it on a daily basis, he often continues late into the night to sit up and program.

Joel is the author of JoiPal the Paypal IPN logging and notification system (http://ipn.joihost.com) and is a member of the Paypal developer network. He most recently finished his work integrating Paypal for escrow services on Listbid (http://listbid.com).