An Introduction Page for paypal-ipn.php Script

First off, teaching about PayPal IPN is beyond the scope of this documentation, but we will provide a bit of insight into it.

What Is PayPal IPN? Instant Payment Notification [https://developer.paypal.com/api/nvp-soap/ipn/] is a message service that automatically notifies merchants of events related to PayPal transactions. Merchants can use it to automate back-office and administrative functions, including automatically fulfilling orders and providing customers with order status.

Beyond the above, if you are unfamiliar with IPN, use Google to direct you to many pages within PayPal's Sandbox (aka developer.paypal.com).

Some General Notes:

Oct 2nd, 2022: This is still a very new application. It has been tested locally and on the Paypal sandbox. However, it does need much more testing to consider it stable.

Scripts required for IPN: paypal-ipn.php and PaypayIPN.php

Most likely installation should be in your theme child directory.

CURRENTLY custom functions used will be in the PaypalIPN.php script.

CURRENT FEATURES INCLUDED: Ability to process any and all messages sent by PayPal. Each valid message will be saved to a unique text file.

PLANNED FEATURE ENHANCEMENT: Write any and all messages sent by PayPal to MySQL database table (wp_paypal_transactions). Table schema will be posted soon.

This DB table represents my view of PayPal data. One key field is wp_user.ID (if transaction by a logged in Wordpress user, else value = 0). Much more information will be released, once the DB interface is functional (originally written in 2020).

MORE THOUGHTS ON MESSAGES: PayPal sends many different types of transactions (complete, pending, canceled and so on). Handling these transactions will be left to individual developers.

OTHER CURRENT FEATURES: Reports, reports, reports. Maybe an API for Quickbooks.

Document for paypal-ipn.php

This script calls Class Script, which does all the heavy lifting.

I try to do a fair amount of documentation.

Some of the code in here is R&D, but it has been left in place.

No template required for this script!

```
<?php namespace Listener;
/**
* Template Name: Paypal Ipn
* script_name: paypal-ipn.php
* parent_script_name: none
* page_name: Paypal Ipn
* application_name: PayPal IPN Validation for PayPal IPN
* business_use: WpApps PayPal Ipn Solutions
* re-authored: Dave Van Abel
* dev_site: wpappsforthat.com
* re-create_date: 2022-08-22
* last_update_date: 2022-09-26
* base_note: Source = https://dopehacker.com/paypal-integration-
website-php/ & now from Github.com
* Github 08-28-22 DVA: https://github.com/paypal/ipn-code-
samples/blob/master/php/example_usage_advanced.php
 * A list of variables is available here:
 *  https://developer.paypal.com/webapps/developer/docs/classic/
ipn/integration-guide/IPNandPDTVariables/
* Last Committed: Latest commit 3b21502 on May 4, 2017
* status: Work-in-progress
* WithDBCode: paypal-ipn-08-28-22-withDBsavedCode.php
* Listener MUST BE 1ST THING (excluding comments) in the file
*/

/**
* 09-27-22 - echo print lines have been left in for local
debugging & do not affect PP Simulator.
* 09-27-22 - DVA re-testing locally, tested Paypal Simulator.
Both Good!
* 09-26-22 DVA retested locally.
* 09-09-22 @8:31pm - DVA This is SIMULATOR USE - SANDBOX
* 09-08-22 DVA - Added if ($le) { code and also had to pass $le
+ retrieve on some functions (PaypalIPN.php).
*                As of now, can not pass to "public function
usePHPCerts()" as not called from paypal-ipn.php script
* GENERAL RULLE ON pass to function & retrieve:
* Pass to: $ipn = new PaypalIPN($le);
* Retrieve from: public function useSandbox($le)
* UNABLE TO MANAGE:
*       public function usePHPCerts(), as not called from
paypal-ipn.php
*       but never called
* IF EVER NECESSARY !!
 * How to create cert files?
 * Use this command from a terminal on the Mac Mini:
```

```
 * $openssl req -newkey rsa:2048 -new -nodes -x509 -days 3650
-keyout key.pem -out cert.pem<cr>
 * Follow all commands and two files created for the cert!
 * Outputs:
 * 1) Location = /Users/davevanabel
 * 2) cert.pem and key.pem
 * 3) Now more out each file and
 * 4) Save to 0-wpappsforthat.com/cert
 * 5) Copy over to wpappsforthat.com website: <wp-content>
 * 6) Now should have valid certs for PaypalIPN use!
 */
//
if ( ! defined( 'ABSPATH' ) ) {die( '-1' );}

?>
    <br>PP->TOP OF LOCAL IPN TEST PROCESS! -> <a
href="index.php/local-test/"><b>IPN Local Test</b></a><br><br>
<?php
```

Used in local output to provide easy navigation in local testing.

```
$enable_sandbox = true; // Set this to true to use the sandbox
endpoint during testing

//$my_email_addresses = array("my_email_address@gmail.com",
"my_email_address2@gmail.com", "my_email_address3@gmail.com");
$my_email_addresses = array("yourseller@yourdomain.com");

require('PaypalIPN.php');
use PaypalIPN;
$ipn = new PaypalIPN();
```

Used for Papal

Controls Local vs. Paypal Use

```
//SIMULATOR LOCATION //
$test_local = 1;   // 1= Local, 0 = Paypal Simulator
echo "PP-1a1->Before New PaypalIPN Function defined<br>";

$location = new PaypalIPN();

echo "PP-1a2->After New PaypalIPN Function defined<br>";
echo "PP-2a1->Before set_name Function call<br>";

$location->set_name($test_local);

echo "PP-2a2->After set_name Function call<br>";

list ($local_ver, $local_inv, $sim_ver, $sim_inv) = $location-
>get_name($test_local);
```

```
echo "<br>PP-3a1->Local Valid = $local_ver<br>";
echo "PP-3b1->Local InValid = $local_inv<br>";
echo "PP-3c1->Sim Valid = $sim_ver<br>";
echo "PP-3d1->Sim InValid = $sim_inv<br>";

if ($enable_sandbox) {
    $ipn->useSandbox();
}

list($verified, $logfile, $data_text) = $ipn->verifyIPN();

if ($verified) {
    echo "<br>PP-Worked<br>Logfile = $logfile<br>";   // No
content for SIMULATOR VERSION
} else {
    echo "<br>PP End -> FAILED<br>";
}
?>
<br>PP->BOTTOM OF LOCAL IPN TEST PROCESS! -> <a href="index.php/
local-test/"><b>IPN Local Test</b></a><br><br>
<?php
if ($verified) {echo "$data_text<br>";}
// END //
?>
```