Hello Money.

A simple payment application.

Contents

Introduction.

Online commerce is an ever growing facet of our daily lives. The opportunities for financial gain are enormous, and PayPal continues to be a frontrunner in the online payments business. With the development of the Adaptive Payments platform, PayPal has created a robust set of very flexible API's that enable the implementation of a variety of business models without needing to worry about the payment processing aspect of the projects.

In this document we will create a simple application called "Hello Money". This application enables you to send a one-time payment to a specific recipient. We will not focus exclusively on code, this is a comprehensive guide on how to create, test and deploy your first application on PayPal.

You will be walked through all the required steps in setting up your testing environment, creating a basic application that can serve as a base for further development, and how to properly submit your application to PayPal for approval. This guide should be accompanied by two sample files, (JSP and PHP). These files are complementary to the information contained here. They are a full working example of a simple Pay API call.

Furthermore, this guide will show you the recommended best practices for integrating your application and your business model.

About the testing environment.

Before we write the first line of code, we need to ensure that the testing environment is properly set up. PayPal's test environment is called **The Sandbox**.

This environment is best seen as a self-contained PayPal, populated with virtual accounts. Any credentials or accounts from a **Live** PayPal account do not exist in the Sandbox, and vice versa. Any calls made using live credentials will fail if used within the sandbox.

The main purpose of the sandbox is to allow you to try different scenarios, and to be able to test your implementation with virtual money and virtual accounts. Any live financial information (Such as credit cards numbers, bank accounts, etc) should not be used within the sandbox environment.

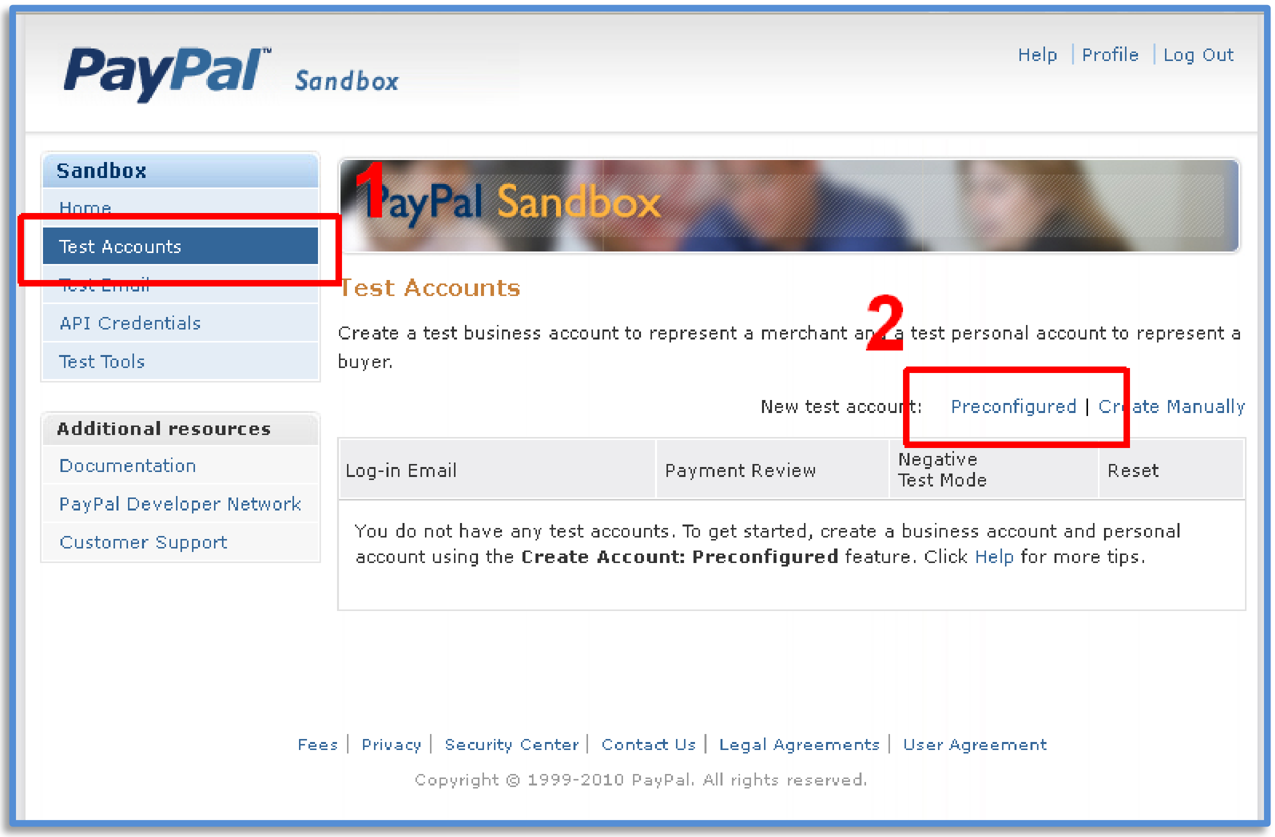
Setting up the testing environment.

1) Go to http://developer.paypal.com and create an account. This site allows you to control your sandbox environment.

2) Now we will create a few test accounts. These are virtual accounts that exist only in the sandbox, and will represent the senders and receivers in our test transactions. For this guide, we will create three accounts. One sender, who represents the person purchasing a good or service; one receiver, who represents the person/business that will get the money from the sender and finally the Caller. This account represents the owner of the Application.

   This last account is optional, as the receiver could also be the API owner, but for the purposes of this example, we will have the API caller be a third party that is not involved in the transaction.

To create the test accounts go to the “Test Accounts” menu, and then click on “Preconfigured”.



Test Accounts

Create a test business account to represent a merchant and a test personal account to represent a buyer.

New test account: Preconfigured | Create Manually

Log-in Email	Payment Review	Negative Test Mode	Reset
You do not have any test accounts. To get started, create a business account and personal account using the **Create Account: Preconfigured** feature. Click Help for more tips.			

This will bring up the pre-configured account options.
For this guide we will be working with United States based accounts using USD as the currency.

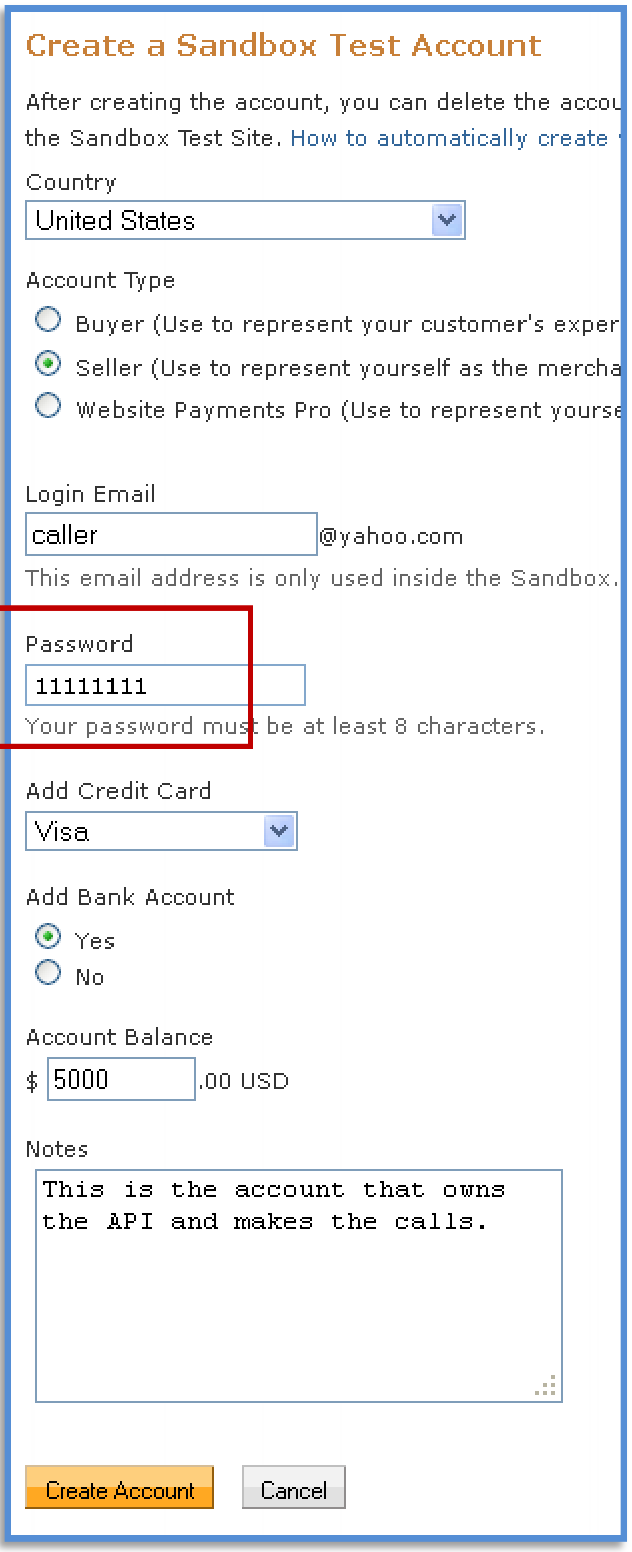
For testing purposes we will make sure that all accounts are verified. They will have a credit card, a bank account and plenty of balance for testing purposes.

The API caller and Receiver will be set as “Seller” account type, and the Sender will be set as a “Buyer” account type. Please refer to the following images for all the parameters that need to be set.

Tip:
In this guide we are only creating accounts that are fully qualified (They have a Credit Card, a verified bank account and PayPal Balance) in order to make getting started easier. However, when you go live, you will encounter many variations in how accounts are set up.

Once your first basic implementation is up and running, we highly recommend that you test your application under a number of different scenarios. Appendix A presents the recommended scenarios that should be tested to insure a robust application that can handle common real life cases.

Test Account 1: This account represents the API caller.



Create a Sandbox Test Account

After creating the account, you can delete the accou
the Sandbox Test Site. How to automatically create

Country
United States

Account Type
Buyer (Use to represent your customer's exper
[✓ Seller (Use to represent yourself as the mercha]
Website Payments Pro (Use to represent yourse

Login Email
caller @yahoo.com
This email address is only used inside the Sandbox.

Password
11111111
Your password must be at least 8 characters.

Add Credit Card
Visa

Add Bank Account
[✓ Yes]
No

Account Balance
$ 5000 .00 USD

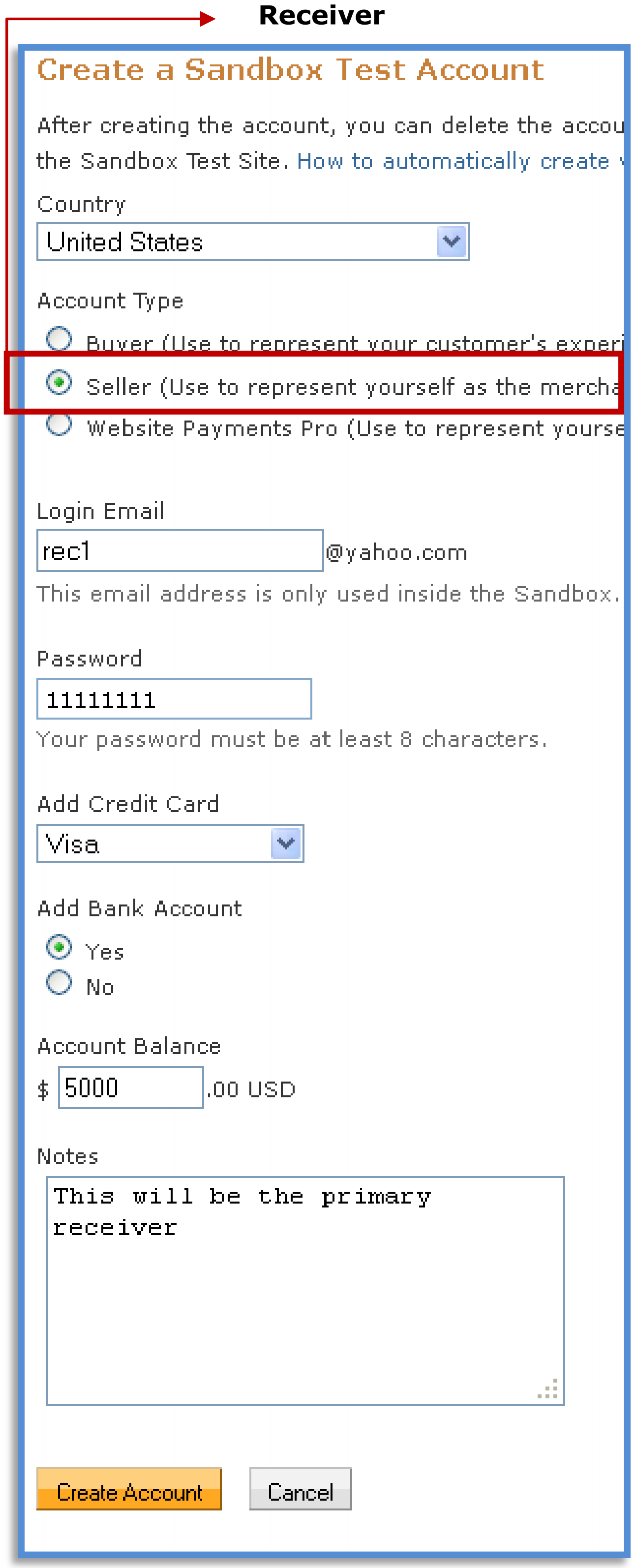
Notes
This is the account that owns the API and makes the calls.

***Tip*:**
Use the same password for all the test accounts.

This will simplify testing in later stages.

Repeat the steps above and create an additional two accounts with the following configuration. These will represent the sender and receiver in the transaction.

Receiver



Create a Sandbox Test Account

After creating the account, you can delete the accou
the Sandbox Test Site. How to automatically create

Country
United States

Account Type
Buyer (Use to represent your customer's experi
[✓ Seller (Use to represent yourself as the mercha]
Website Payments Pro (Use to represent yourse

Login Email
rec1 @yahoo.com
This email address is only used inside the Sandbox.

Password
11111111
Your password must be at least 8 characters.

Add Credit Card
Visa

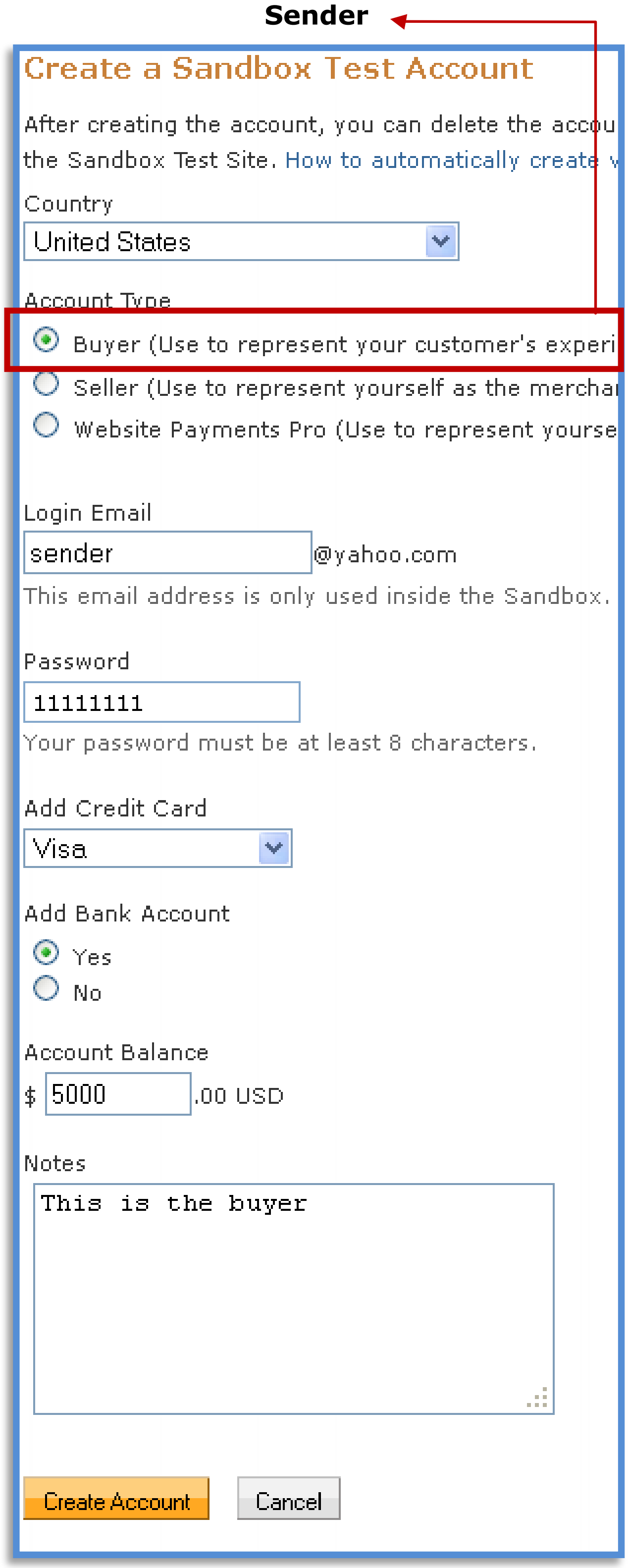
Add Bank Account
[✓ Yes] No

Account Balance
$ 5000 .00 USD

Notes
This will be the primary receiver

Create Account | Cancel

Sender



Create a Sandbox Test Account

After creating the account, you can delete the accou
the Sandbox Test Site. How to automatically create

Country
United States

Account Type
[✓ Buyer (Use to represent your customer's experi]
Seller (Use to represent yourself as the mercha
Website Payments Pro (Use to represent yourse

Login Email
sender @yahoo.com
This email address is only used inside the Sandbox.

Password
11111111
Your password must be at least 8 characters.

Add Credit Card
Visa

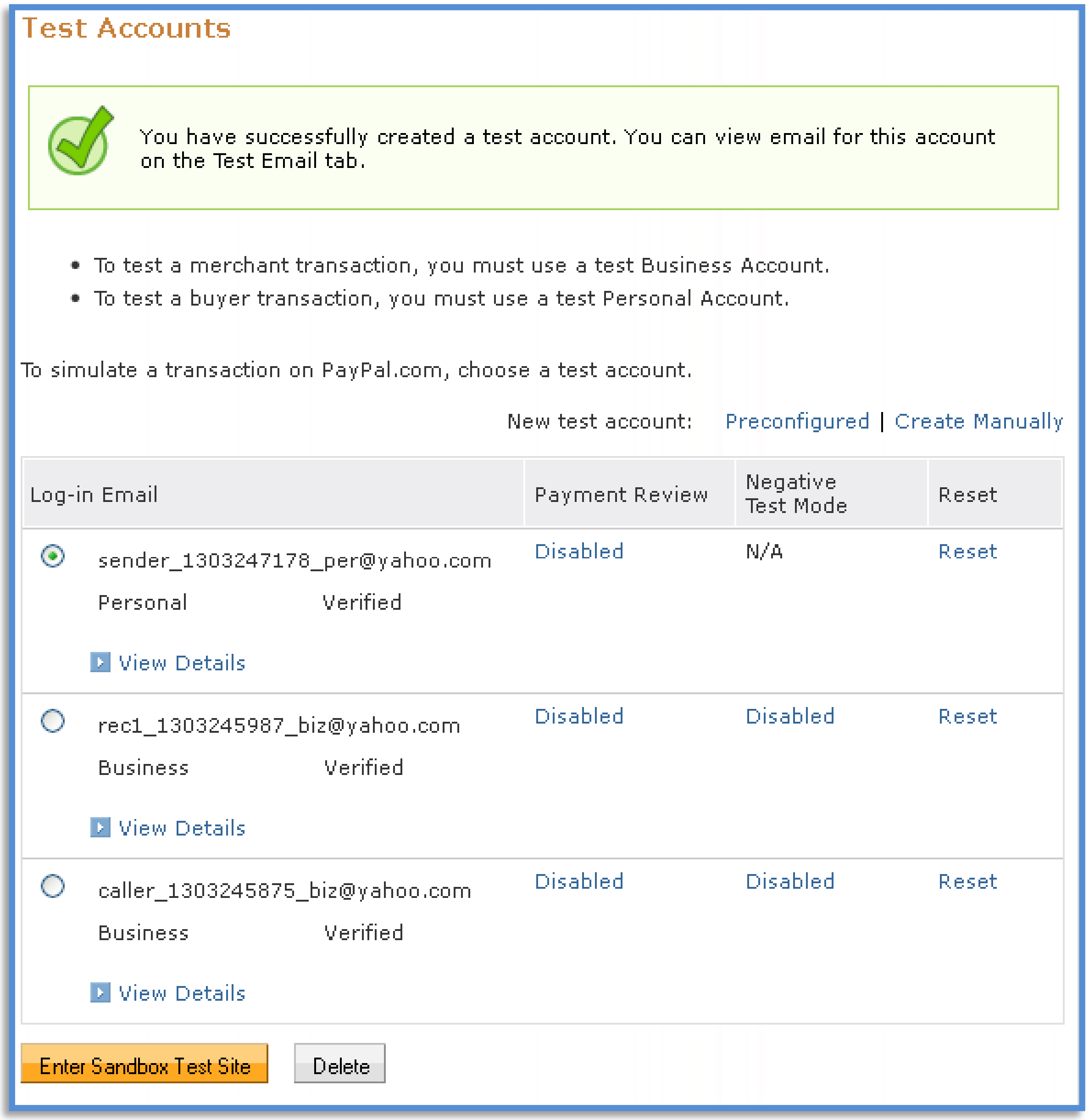
Add Bank Account
[✓ Yes] No

Account Balance
$ 5000 .00 USD

Notes
This is the buyer

Create Account | Cancel

Once you are done, you should have three accounts set up (Virtual account emails will differ slightly from the sample image)



Test Accounts

You have successfully created a test account. You can view email for this account on the Test Email tab.

- To test a merchant transaction, you must use a test Business Account.
- To test a buyer transaction, you must use a test Personal Account.

To simulate a transaction on PayPal.com, choose a test account.

New test account: Preconfigured | Create Manually

Log-in Email	Payment Review	Negative Test Mode	Reset
● sender_1303247178_per@yahoo.com Personal Verified View Details	Disabled	N/A	Reset
○ rec1_1303245987_biz@yahoo.com Business Verified View Details	Disabled	Disabled	Reset
○ caller_1303245875_biz@yahoo.com Business Verified View Details	Disabled	Disabled	Reset

Obtaining API credentials for testing.

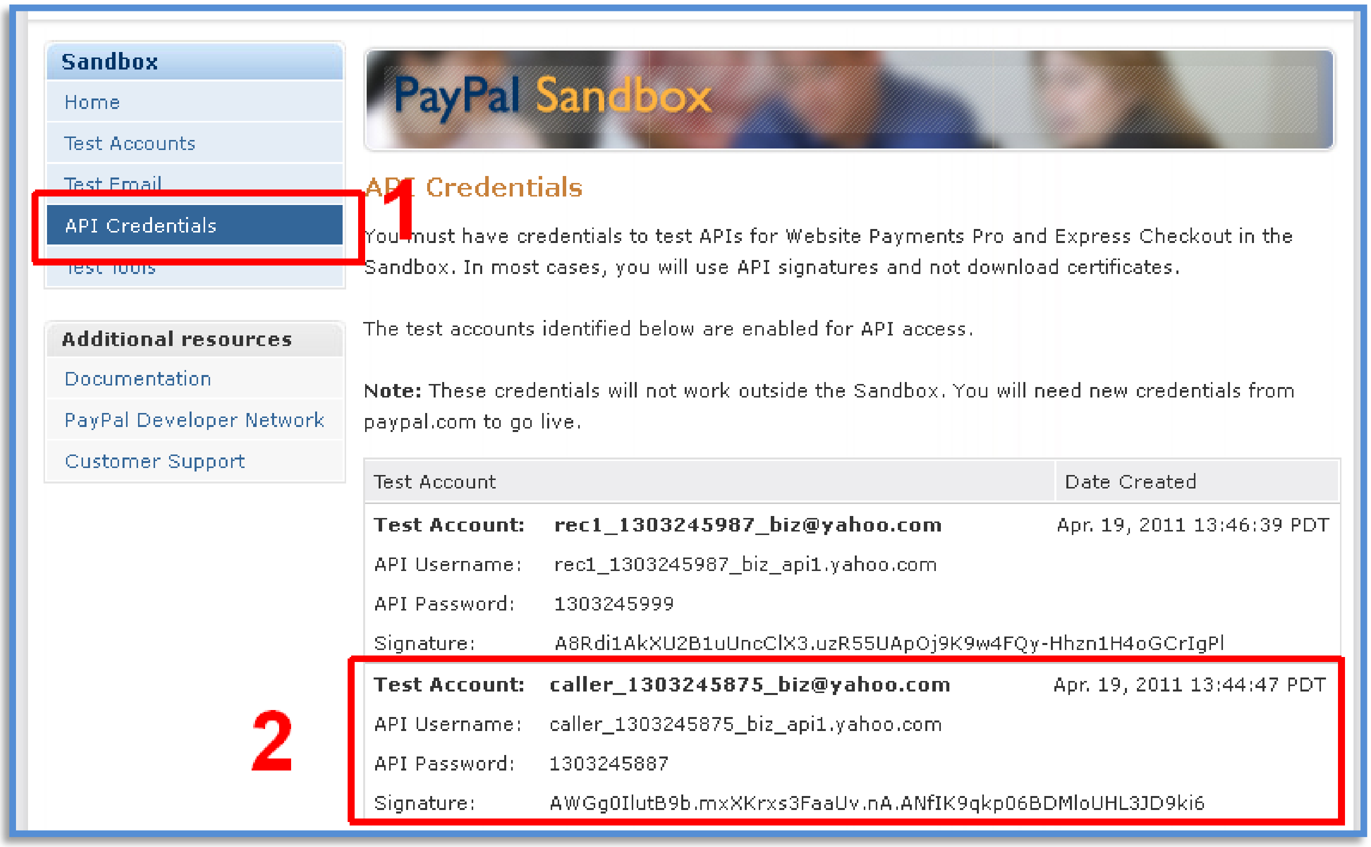
The following step is gathering the credentials you will need to make API calls in the sandbox. The first of these credentials is the APP-ID. All applications in the sandbox will use the following APP-ID: **APP-80W284485P519543T**

Like other sandbox credentials, this application ID works only within the sandbox. Once you are ready to move your application to a live environment, you will need to apply for your own App-Id (This is explained in more detail in chapter XX).

Next, we need to obtain the credentials for the account that will be making the API calls. This is the "Caller" account that we created in the previous step. The credentials needed are:

- *API Username* (This is different from the login email)
- *API Password*
- *Signature*.

To obtain these credentials go to the "API Credentials" menu, and take note of the ones for the caller's test account. These will be used in our sample application.



API Credentials

You must have credentials to test APIs for Website Payments Pro and Express Checkout in the Sandbox. In most cases, you will use API signatures and not download certificates.

The test accounts identified below are enabled for API access.

Note: These credentials will not work outside the Sandbox. You will need new credentials from paypal.com to go live.

Test Account		Date Created
Test Account:	**rec1_1303245987_biz@yahoo.com**	Apr. 19, 2011 13:46:39 PDT
API Username:	rec1_1303245987_biz_api1.yahoo.com	
API Password:	1303245999	
Signature:	A8Rdi1AkXU2B1uUncClX3.uzR55UApOj9K9w4FQy-Hhzn1H4oGCrIgPl	
Test Account:	**caller_1303245875_biz@yahoo.com**	Apr. 19, 2011 13:44:47 PDT
API Username:	caller_1303245875_biz_api1.yahoo.com	
API Password:	1303245887	
Signature:	AWGg0IlutB9b.mxXKrxs3FaaUv.nA.ANfIK9qkp06BDMloUHL3JD9ki6	

An Adaptive Payments Primer

Our testing environment is fully set up. We are now almost ready to start coding, but first, we will take a quick look at how an Adaptive Payments program is typically implemented, and how the technology works.

An Adaptive Payments API call is nothing more than an HTTP request. The request needs to have a few PayPal-specific headers that will identify the caller, as well as the format of the information contained. The body of the requests contains all the pertinent information to the transaction (e.g.: Who the receivers are, what amount is being transferred, etc).

Here is an example of the simplest **Pay** request, presented in NVP (Name-Value pair) format:

Request Headers:

X-PAYPAL-APPLICATION-ID	APP-80W284485P519543T
X-PAYPAL-SECURITY-USERID	caller_1303245875_biz_api1.yahoo.com
X-PAYPAL-SECURITY-PASSWORD	1303245887
X-PAYPAL-SECURITY-SIGNATURE	AWGg0IlutB9b.mxXKrxs3FaaUv.nA.ANfIK9qkp06BDMloUHL3JD9ki6
X-PAYPAL-REQUEST-DATA-FORMAT	NV
X-PAYPAL-RESPONSE-DATA-FORMAT	NV

*(Credentials here are obtained in the previous step)

Request Body:

*(Body formatted for readability. Actual request should not have unnecessary white space and parameter values should be URLencoded)

```
actionType=PAY &
currencyCode=USD &
receiverList.receiver(0).email=rec1_1303245987_biz@yahoo.com &
receiverList.receiver(0).amount=10.00 &
returnUrl=http://www.example.com/success.html &
cancelUrl=http://www.example.com/failure.html &
requestEnvelope.errorLanguage=en_US
```

Let’s break down that call into a line by line explanation.

The first four request headers identify *who* the caller is to PayPal. The last two indicate the format that the body of the request is in, and how PayPal should format the response. In this example we are using NV (Name-Value pair) format, which is nothing more than *param1=val1 & param2=val3 &…&paramN=valN.*

The parameters used in this call are the minimum required by PayPal. They are:

"actionType"
This parameter lets PayPal know want you want to do with this call. In this case the value "*PAY*" indicates, as its name implies, a payment.

"currencyCode"
This parameter indicates what currency this transaction will be performed in. For this guide we will be using U.S. Dollars (USD).

"receiverList.receiver(0).email"
This parameter indicates who will be receiving the funds. This email address corresponds to that of the virtual account receiver previously created.

"receiverList.receiver(0).amount"
This parameter indicates the amount being sent to the recipient. This amount should always have exactly two decimal places.

"requestEnvelope.errorLanguage"
Tells PayPal in what language to present the results of this operation.

"returnUrl"
Tells PayPal where to redirect the user after he has approved the payment.

"cancelUrl"
Tells PayPal where to redirect the user if the transaction is not completed.

Now that we have our request, we need to send it to PayPal. The endpoint that handles ***Pay*** calls is:

`https://svcs.`**`sandbox`**`.paypal.com/AdaptivePayments/Pay`

Note the bolded **Sandbox** keyword. This means that this call will be sent to the testing environment. Once your application is ready for production, this must be removed from the Endpoint URL.

If everything goes well we will receive a response from PayPal very similar to the following:

```
paymentExecStatus=CREATED &
payKey=AP-11C14830PD187845M &
responseEnvelope.timestamp=2011-04-20T15:30:45.337-07:00 &
responseEnvelope.correlationId=0ca5aa73b42d5 &
responseEnvelope.ack=Success &
responseEnvelope.build=1846084
```

In this response there are three important parameters that one should look for.

"responseEnvelope.ack"
This is the result of the call. It will be either "*Success*" or "*Failure*"

“payKey”
This is a unique identifier for this transaction. You will use this identifier whenever you need to reference the transaction.

“paymentExecStatus”
Tells you if the payment has been completed or not. In this case we received a status of *CREATED*. This means that the payment has not been completed, and further action must be taken.

This particular response tells us that the call was successful, and PayPal is now waiting for the sender of the money to approve the transaction.

This means that the sender must now be redirected to PayPal, so that he can review the details of the transaction, select the funding source, and click on “Pay”.

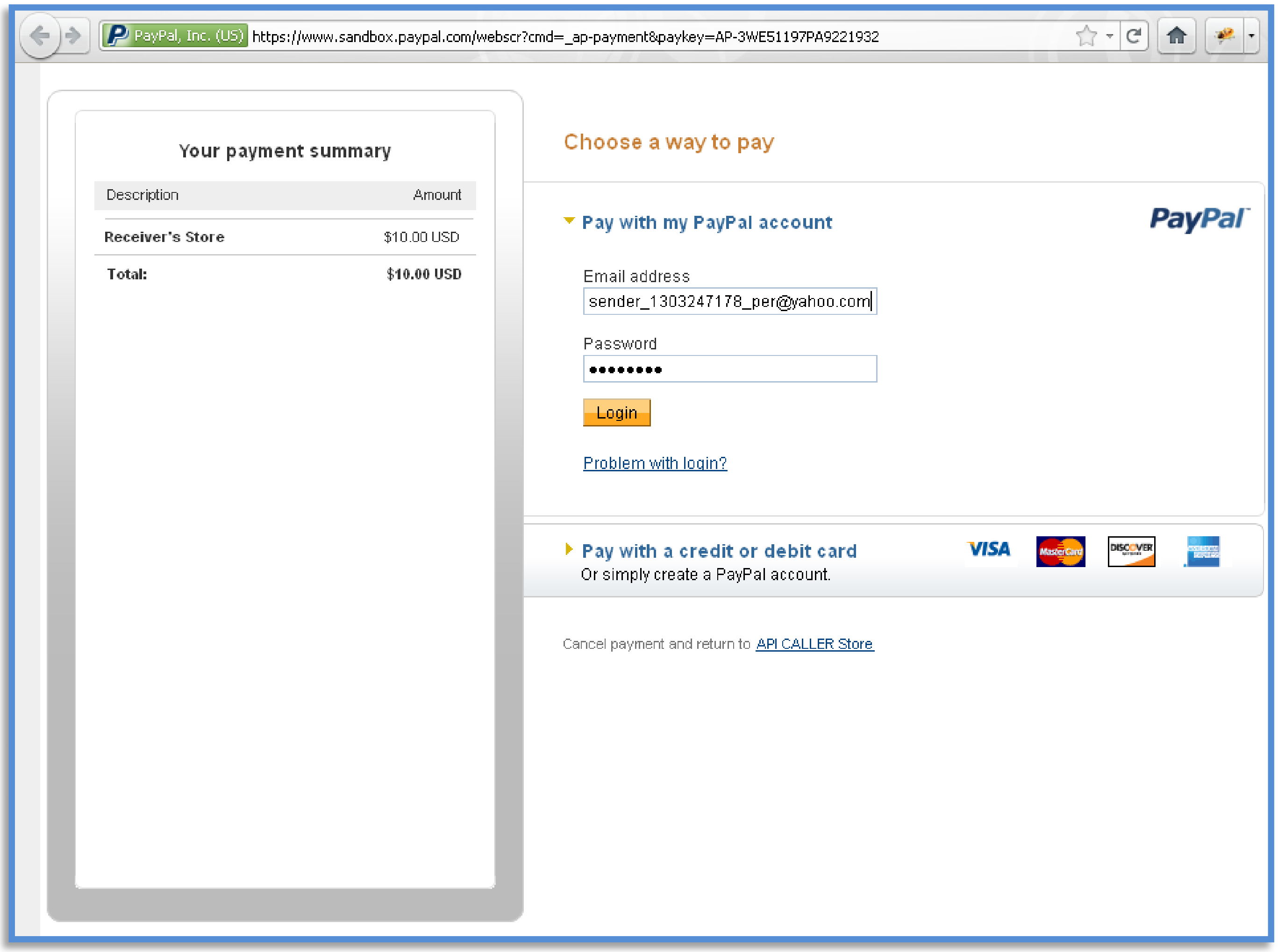
Once this happens, the funds will be transferred to the receiver, and the transaction will be completed. At this point, the user will be redirected to the website that was specified in the returnUrl parameter in the original call.

To redirect the user to PayPal, while referencing the transaction that he must complete, simply send the user to the following URL:

```
https://www.sandbox.paypal.com/webscr?cmd=_ap-payment&paykey=AP-11C14830PD187845M
```

Note the use of the sandbox keyword and also the use of the PayKey parameter.

Here, we will login and pay using the credentials for the “Sender” account we created:


PayPal, Inc. (US) https://www.sandbox.paypal.com/webscr?cmd=_ap-payment&paykey=AP-3WE51197PA9221932

Your payment summary

Description	Amount
Receiver's Store	$10.00 USD
Total:	$10.00 USD

Choose a way to pay

Pay with my PayPal account

Email address: sender_1303247178_per@yahoo.com

Password: ●●●●●●●●●

Login

Problem with login?

Pay with a credit or debit card

Or simply create a PayPal account.

Cancel payment and return to API CALLER Store

Once this payment has been completed, PayPal will redirect the user to the URL specified in the API call. The transaction is now complete!

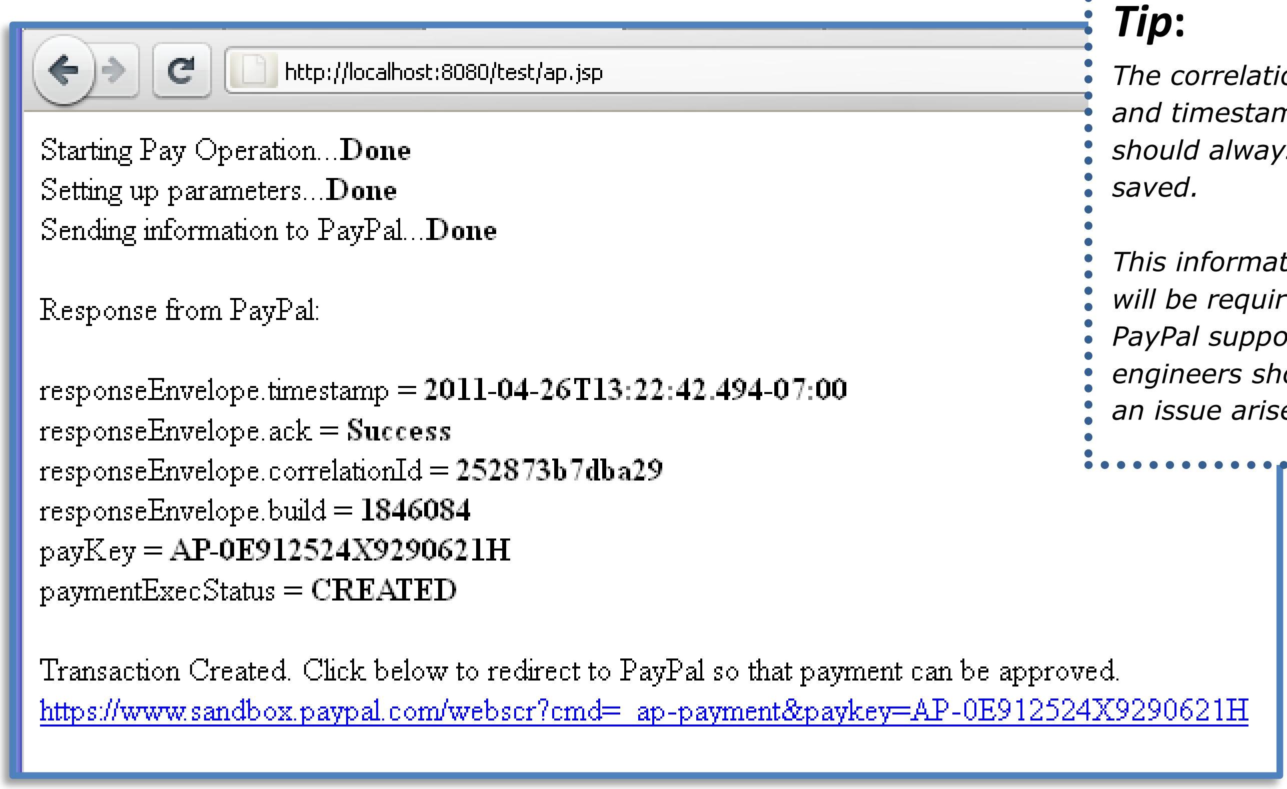
So in a nutshell, the steps to perform an Adaptive Payments Pay call are:

1) Create request with all payment parameters
2) Send request to PayPal
3) Receive response, and extract PayKey Parameter
4) Redirect buyer to PayPal using PayKey as a reference
5) PayPal returns buyer to your site after payment is complete.

Now that we have a test environment and we know how an Adaptive Payments call works; it’s time to write some code.

This guide contains examples in PHP and JAVA.

The sample code will execute the same example transaction that is shown in this guide.
If all goes well you should see the following:



http://localhost:8080/test/ap.jsp

```
Starting Pay Operation...Done
Setting up parameters...Done
Sending information to PayPal...Done

Response from PayPal:

responseEnvelope.timestamp = 2011-04-26T13:22:42.494-07:00
responseEnvelope.ack = Success
responseEnvelope.correlationId = 252873b7dba29
responseEnvelope.build = 1846084
payKey = AP-0E912524X9290621H
paymentExecStatus = CREATED

Transaction Created. Click below to redirect to PayPal so that payment can be approved.
https://www.sandbox.paypal.com/webscr?cmd=_ap-payment&paykey=AP-0E912524X9290621H
```

***Tip*:**
The correlationId and timestamp should always be saved.

This information will be required by PayPal support engineers should an issue arise.

Ideally, the user should be automatically redirected to PayPal, but we take this intermediate step so that the response from the API call can be studied.

The parameter **responseEnvelope.ack** will tell you if the transaction was accepted by PayPal

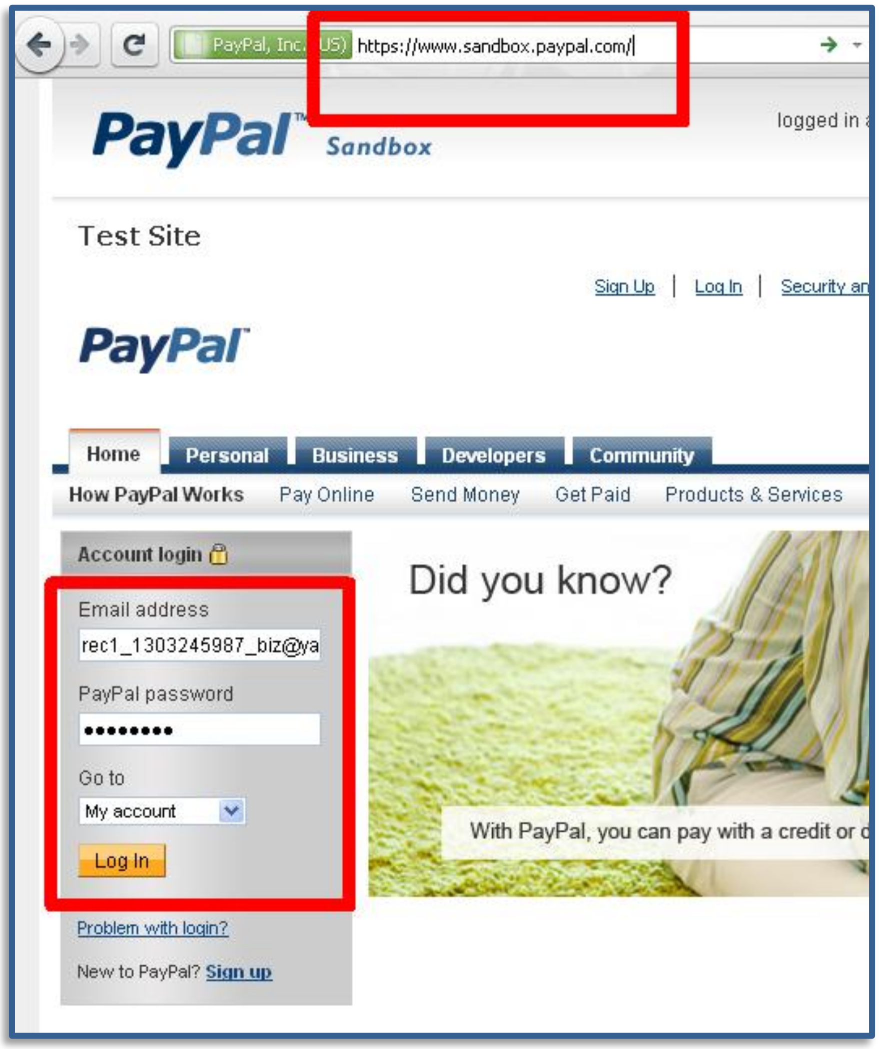
Parameters **correlationId** and **build** are log identifiers for this API call. This information should be saved as it will be useful in troubleshooting should any issue arise.

The **payKey** parameter is the transaction token that will be used to redirect the user to PayPal. This token should also be saved, as it becomes a permanent identifier for the transaction once the payment has been created; and will be necessary in a number of other calls related to the transaction (Such as getting transaction details, performing a refund, etc).

Finally the **paymentExecStatus** parameter shows you what the result of the payment request was. In this case, "CREATED" means that the payment is set up, but awaiting approval from the sender. This is why we must redirect the user to PayPal in order to complete the transaction.

Now that the transaction is completed, we will log into the sandbox as the receiver, so that we can verify our transaction. This login is done at: *http://www.sandbox.paypal.com*

Note the credentials belong to the receiver test account that we created.



https://www.sandbox.paypal.com/

PayPal Sandbox

logged in a

Test Site

Sign Up | Log In | Security an

Home | Personal | Business | Developers | Community

How PayPal Works | Pay Online | Send Money | Get Paid | Products & Services

Account login

Email address: rec1_1303245987_biz@ya

PayPal password: ••••••••

Go to: My account

Log In

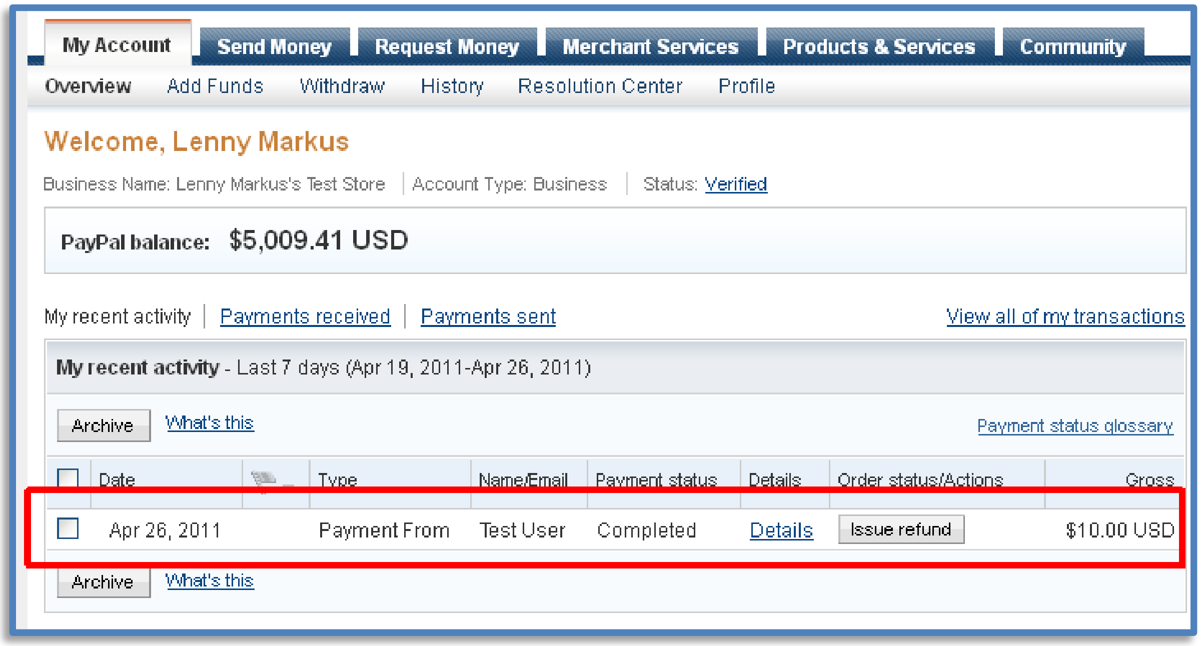
Problem with login?

New to PayPal? Sign up

Did you know?

With PayPal, you can pay with a credit or d

Here we can see that the transaction was completed successfully and that the money has been received.



My Account | Send Money | Request Money | Merchant Services | Products & Services | Community

Overview | Add Funds | Withdraw | History | Resolution Center | Profile

Welcome, Lenny Markus

Business Name: Lenny Markus's Test Store | Account Type: Business | Status: Verified

PayPal balance: $5,009.41 USD

My recent activity | Payments received | Payments sent | View all of my transactions

My recent activity - Last 7 days (Apr 19, 2011-Apr 26, 2011)

Archive | What's this | Payment status glossary

	Date		Type	Name/Email	Payment status	Details	Order status/Actions	Gross
☐	Apr 26, 2011		Payment From	Test User	Completed	Details	Issue refund	$10.00 USD

Archive | What's this

Congratulations! You've made your first transaction using Adaptive Payments!

By now you should have a more complete picture of how the PayPal API system works, how to make calls and how the responses from PayPal are structured.

Keep in mind that this guide only highlights a very simple example of sending money from point A to point B. However, the Adaptive Payments Platform is much more powerful and flexible, and can allow you to implement some very complex business models.

With Adaptive Payments you can create applications that split up payments between many recipients, you can create payments that travel down a chain of recipients; you can create subscriptions based businesses using Pre-Approvals, and much more.

Now would be a good time to do some more reading on the Adaptive Payments developer's guide, to get a taste of all that it offers. Documentation can be found here:

https://www.x.com/community/ppx/documentation#ap

So far we have only been working with virtual accounts. The next section shows you how to obtain live credentials, so that you can start using your application in a production environment, with actual users and of course, real money.

Going Live

Hopefully by now you have built a more complete application, and you are ready to launch your website.

At this point you will go from working with virtual accounts to real accounts, but there are a couple of steps that must be taken before this happens.

PayPal is a global leader in online payments and one of our main goals is offering a safe, secure and reliable environment for all out users. Before you are granted permissions to make Live API calls, you must submit your application for review by PayPal.

During the review process, two things will happen. First, a PayPal agent will use your application from a buyer's point of view to ensure the correct flow and utilization of the API's.

Second, your business model will be thoroughly evaluated to ensure the safety of our customers, as well as to verify compliance with all PayPal guidelines and terms of service.

The PayPal Developer agreement highlights all the key points that your application must comply with. This document can be found at the following link, and should be fully read and understood before submitting your application to PayPal for review.

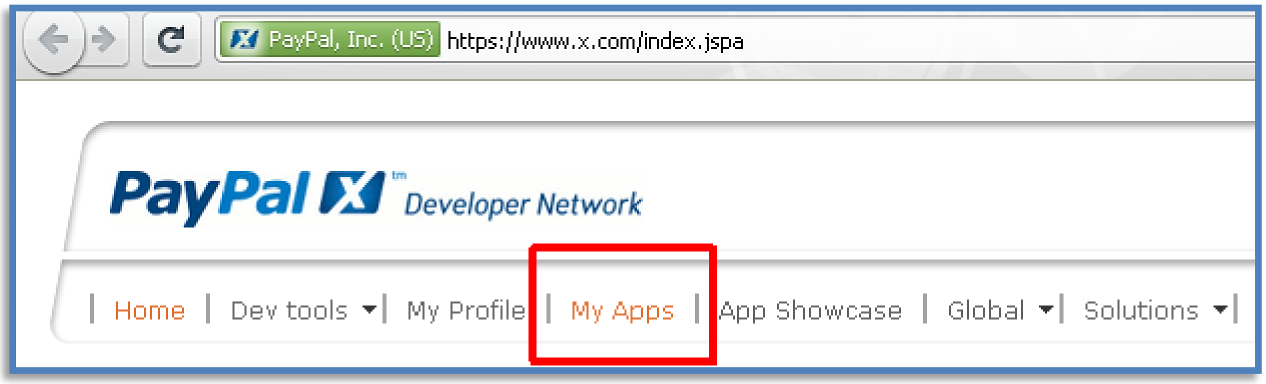
[https://www.x.com/docs/DOC-2161 **]**

Obtaining a live APP-ID

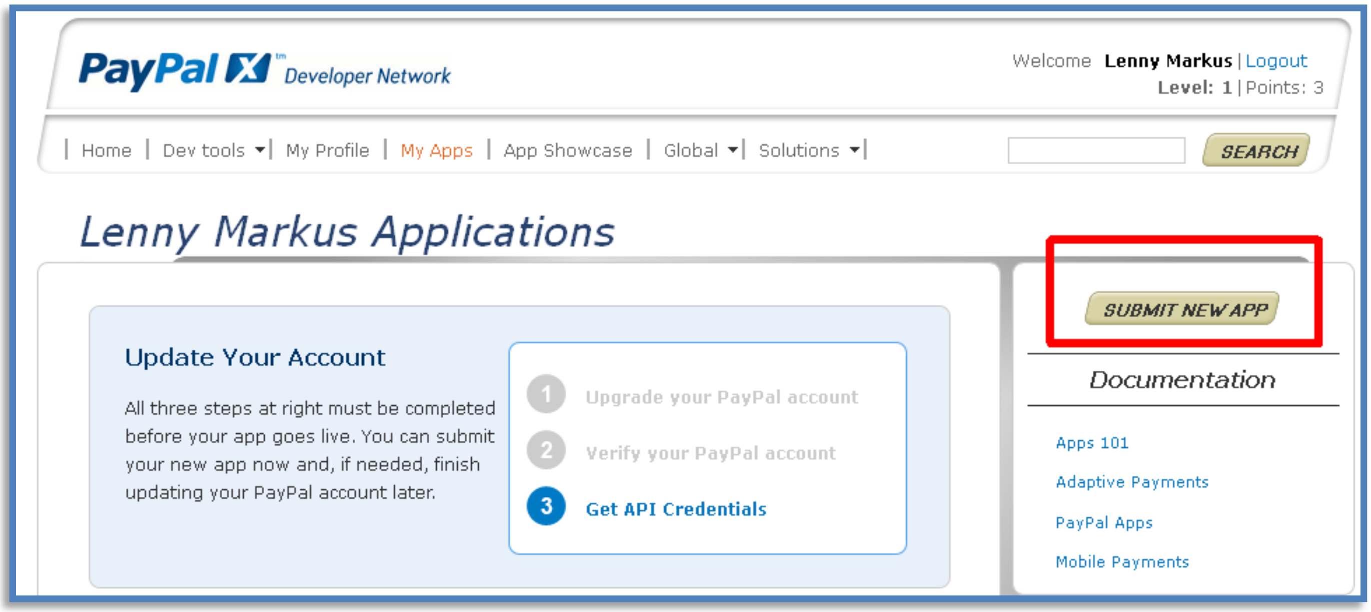
In order to submit your application for review, you must take the following steps:

- Log in to X.com **using the PayPal.com account that will own the application** and make the API calls. This account must be a Premier Verified or Business Verified account to be eligible for making API calls. If you are developing for a third party, make sure that the submission is performed under their name. Otherwise, ownership of the application may be necessary, causing delays.

- Go to the "My Apps" menu



- Click on "Submit New App"



Lenny Markus Applications

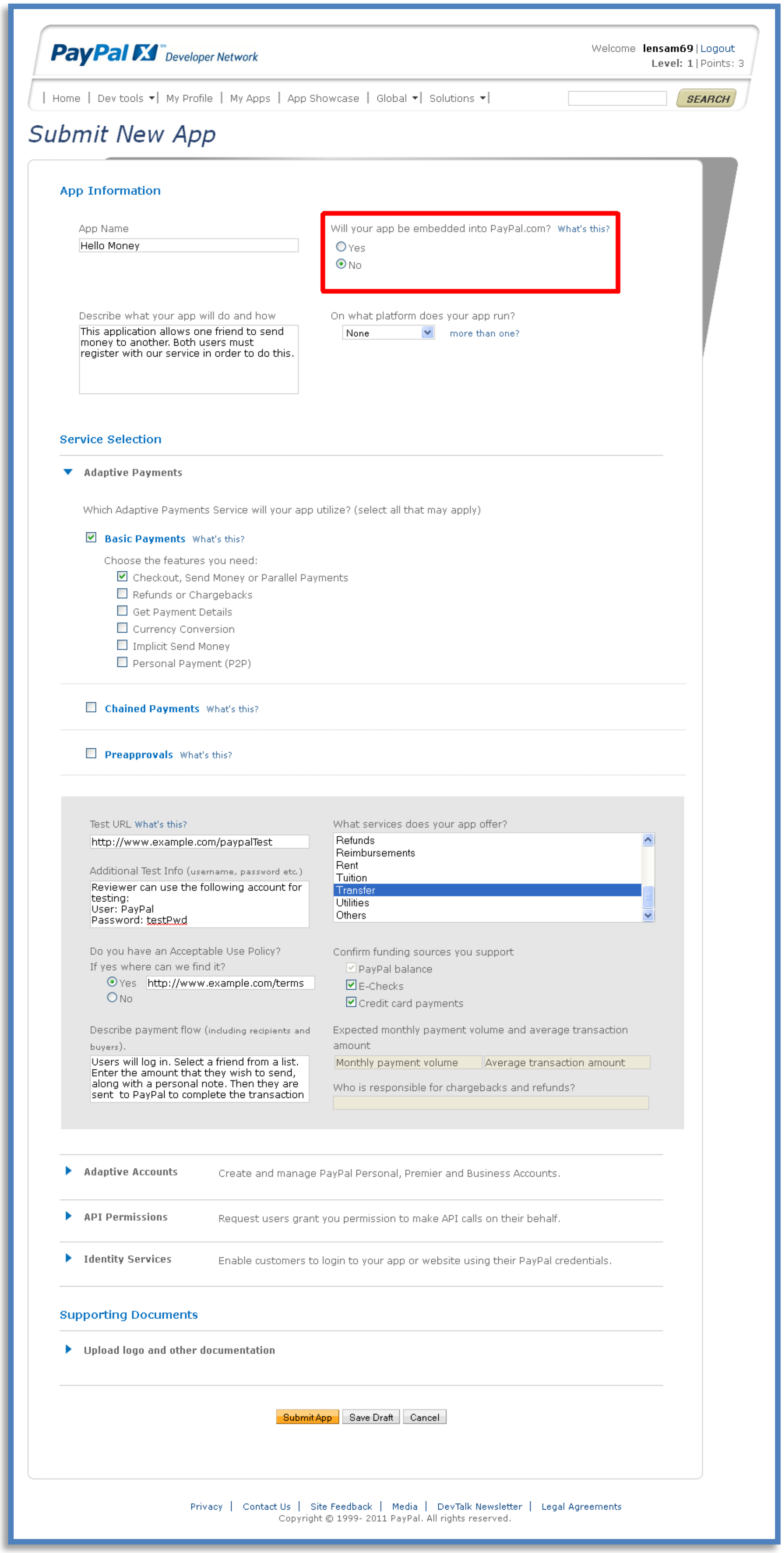
Update Your Account

All three steps at right must be completed before your app goes live. You can submit your new app now and, if needed, finish updating your PayPal account later.

1 Upgrade your PayPal account
2 Verify your PayPal account
3 Get API Credentials

At this point you will be presented with the submission form. This form should be filled out as completely as you can. The more information you present to the PayPal reviewers, the easier it will be for them to understand and approve your application.

Please note that the *"Will your app be embedded into PayPal.com"* option should be set to No.


PayPal X Developer Network
Welcome lensam69 | Logout
Level: 1 | Points: 3
| Home | Dev tools | My Profile | My Apps | App Showcase | Global | Solutions |
SEARCH

Submit New App

App Information

App Name
Hello Money

Will your app be embedded into PayPal.com? What's this?
Yes [✓ No]

Describe what your app will do and how
This application allows one friend to send money to another. Both users must register with our service in order to do this.

On what platform does your app run?
None
more than one?

Service Selection

Adaptive Payments

Which Adaptive Payments Service will your app utilize? (select all that may apply)

[✓ **Basic Payments**] What's this?

Choose the features you need:
- [✓ Checkout, Send Money or Parallel Payments]
- Refunds or Chargebacks
- Get Payment Details
- Currency Conversion
- Implicit Send Money
- Personal Payment (P2P)

Chained Payments What's this?

Preapprovals What's this?

Test URL What's this?
http://www.example.com/paypalTest

Additional Test Info (username, password etc.)
Reviewer can use the following account for testing:
User: PayPal
Password: testPwd

Do you have an Acceptable Use Policy?
If yes where can we find it?
[✓ Yes] http://www.example.com/terms
No

Describe payment flow (including recipients and buyers).
Users will log in. Select a friend from a list. Enter the amount that they wish to send, along with a personal note. Then they are sent to PayPal to complete the transaction

What services does your app offer?
Refunds
Reimbursements
Rent
Tuition
[✓ Transfer]
Utilities
Others

Confirm funding sources you support
- [✓ PayPal balance]
- [✓ E-Checks]
- [✓ Credit card payments]

Expected monthly payment volume and average transaction amount
Monthly payment volume | Average transaction amount

Who is responsible for chargebacks and refunds?

Adaptive Accounts Create and manage PayPal Personal, Premier and Business Accounts.

API Permissions Request users grant you permission to make API calls on their behalf.

Identity Services Enable customers to login to your app or website using their PayPal credentials.

Supporting Documents

Upload logo and other documentation

What to expect now

Now that you have submitted your application, a PayPal agent will perform a full review.

Finally, reviewers might require additional information, or have questions about your app. If so, they will post questions on the app submission page on X.com, which can only be accessed by the same account that submitted the application.

Until the application is approved, it is a good idea to check back often, as the review process will be stopped until these questions / requests are answered by you. If any questions are posted, a notification will be sent **to the email that was used to submit the application.** If you don't have access to this email, make sure to check back on X.com often. It is also a good idea to check your junk mail folder, to insure that legitimate PayPal messages are not being marked as spam.

How long will it take to approve my application?

All reviews are performed on a first-come, first-serve basis. There is no expedited process available. Review times can vary, and you should take this into account in your business planning. The review time varies depending on a number of factors, including the APIs your app uses, the uniqueness and complexity of the business model, if you are submitting a PayPal App, or if you are applying for Business Payments. We might ask a lot of questions or just a few, it all depends.

If you are using Standard APIs ONLY:

If your app uses Standard APIs like Send Payments or Parallel Payments, you can begin using the APP ID in a live environment as soon as you submit your application. PayPal will take a closer look at your app within 24 – 72 hours to ensure that your app meets all requirements for final approval. If there are any questions or concerns during this review period, you will be contacted. If you have not heard from us during or after this period, you can assume your app was reviewed without concern and you may continue using the live APP ID that was assigned to you.

If you are using Advanced APIs:

If your app uses Advanced APIs like Chained Payments, Pre-approved Payments, Personal Payments, Create Account (Business), you are submitting a PayPal App, applying for Business Payments, or you have a complex business model, you can expect the review to take approximately 5 – 15 days. On occasion some reviews will take longer, however this is due to how quickly you respond to our questions or how complex your business model may be.

Upgrades:

If you decide you would like to upgrade or add any Advanced APIs (Chained Payments, Pre-approved Payments, etc) to your already approved app, please contact Developer Technical Services at www.paypal.com/dts. Note that we may need to test the new payment model/flow and your app may be subject to the 5 – 15 day process for Advanced APIs.

Things to consider

Implementing your solution and receiving a Live App-Id is only part of the bigger picture. You are now running a live website and are handling money, as well as dealing with people. PayPal focuses on protecting its customer's money, as well as ensuring their satisfaction and providing a high quality user experience. You should strive for the same goals in your website.

Security

During the integration process, security is usually not a big concern, because the developer is acting as a trusted party; however, once your application is nearing completion; securing your checkout process should be a very high priority.

Particularly, when using pre-approvals, the burden of verifying that a user is who he claims to be is shifted to you, since PayPal will not request their login/password to complete a pre-approved transaction. We offer a document outlining our recommended information security guidelines, which is an excellent starting point towards securing your website. This document can be found at the following link: https://www.x.corn/cornmunity/ppx/apps101/info_security_guildlines

Quality Of Service

Offering a high quality user experience is another piece of the big picture that you must complete. Transactions will not always go as planned, and you may find yourself in unexpected situations.

You should be prepared to deal with customers who may be unhappy with their product / service, and request refunds, or initiate disputes. Having a clear Terms-of-Use policy, as well as a Returns policy readily available on your website is a very good way of protecting yourself, and it lends credibility to your business.

If you have a business with a high volume of transactions, you might benefit from integrating refunding functionality into your application. Besides the Pay API, PayPal also offers a number of API's that enable you to obtain transaction details, status issue refund's etc.

You should consider leveraging the full range of PayPal's API's in order to automate a large part of the accounts receivable aspect of your business. Using a tried and tested solutions from PayPal will provide you with a robust base on which to build your business, while at the same time allowing you to direct your time and energy to other areas.

Branding

The PayPal brand is globally trusted by millions of users, and by integrating it in your checkout process, you can take advantage of this. Making proper use of the PayPal brand during your checkout process enhances trust, and gives credibility to your site.

For proper use of PayPal logos and assets, please read the following document: https://www.x.com/docs/DOC-1506 . It outlines correct usage of PayPal logos and buttons in a way that is consistent with brand guidelines. This is important as it makes your website look legitimate.

Appendix A – Naming Conventions

In this guide we only created three accounts for testing purposes. To keep things simple we named them *caller, sender* and *rec1*. As you start creating more virtual accounts for testing purposes, it's important to keep a uniform naming convention to avoid clutter and streamline testing. You should be able to know exactly what an account represents just by looking at its name.

The sandbox only allows you to customize the first six characters of an account, so you need to maximize the information in them.

The following is an example of how to standardize your names:

- There are two categories of user accounts: **(P)**ersonal and **(B)**usiness.
- An account can be **(V)**erified if it has a BankAccount or **(U)**nverified if it does not.
- Finally, an account can have a **(C)**redit card associated to it or not.
- Use the remaining space for assigning a number to the account

Using these letters we can have the following naming convention for accounts:

Account Name	Personal	Business	Verified	Credit Card
PVC	✔		✔	✔
PV	✔		✔	
PC	✔			✔
P	✔			
BVC		✔	✔	✔
BV		✔	✔	
BC		✔		✔

So, at a glance , we can tell that the following account: **BV1_ 1296593066_biz@gmail.com** is a Business Verified account that does not have a credit card. **PC3_ 1296672107_per@gmail.com** is a Personal non-verified account, that only has a Credit card as a funding source.

This scheme is an example. Feel free to use one that works for you; that will help you keep a large number of accounts organized, and easily recognizable at a glance.

Appendix B – Further Reading