PayPal™

PayPal Apps Developer Guide

Last updated: October 2010

PayPal Apps Developer Guide

Document Number: 10124.en_US-201010

Contents

Preface

PayPal App Developer Guide Revision History

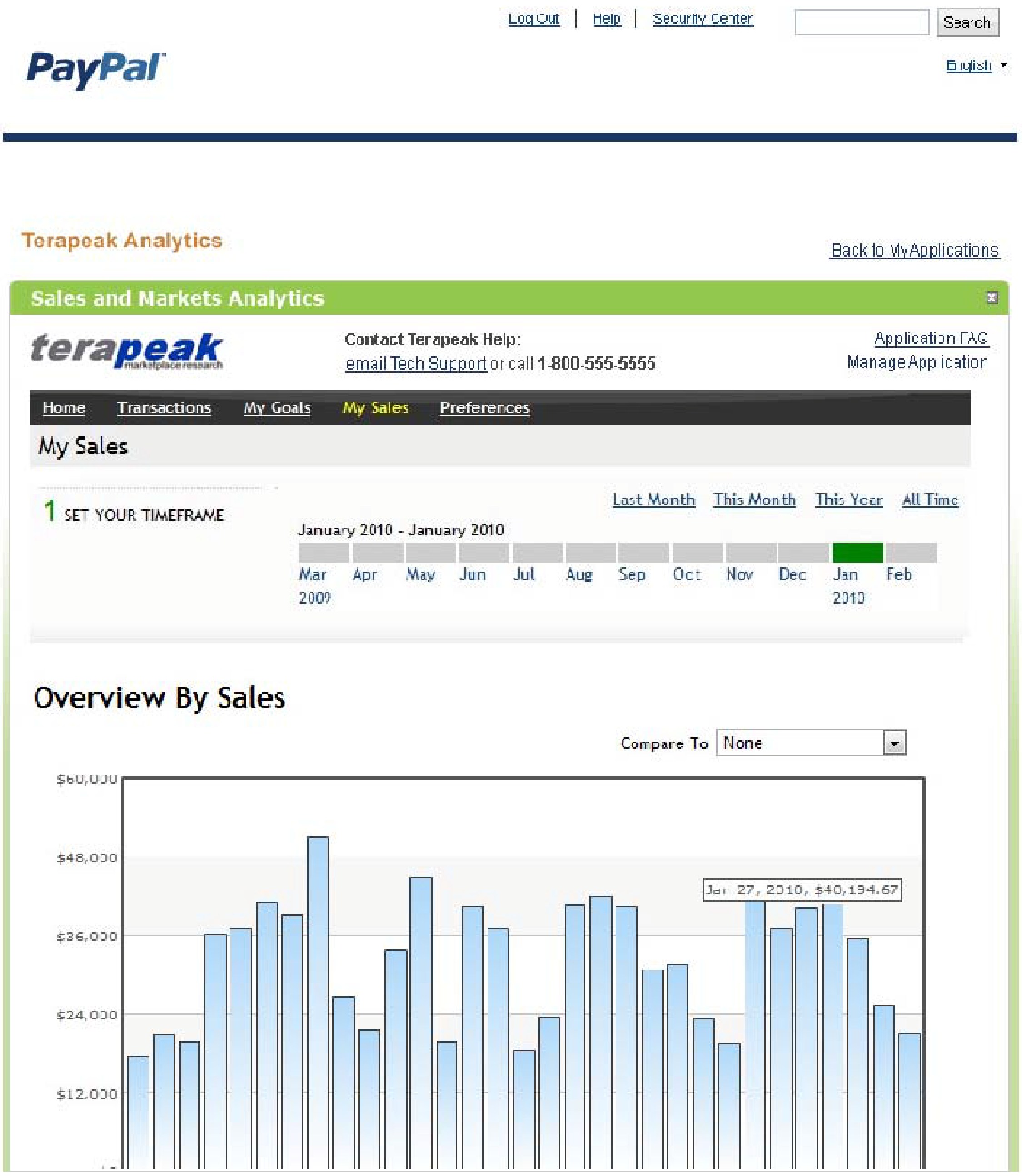
Date	Description
10/22/10	Initial draft for version 1.0.0.

1 Introducing PayPal Apps

PayPal Apps are applications you write and PayPal serves to its account holders when they are logged into paypal.com. PayPal Apps are innovative, often relevant to payments or financial services, and they give you a prominent presence on PayPal while enhancing a PayPal account holder's experience.

A Sample Full Page View of a PayPal App

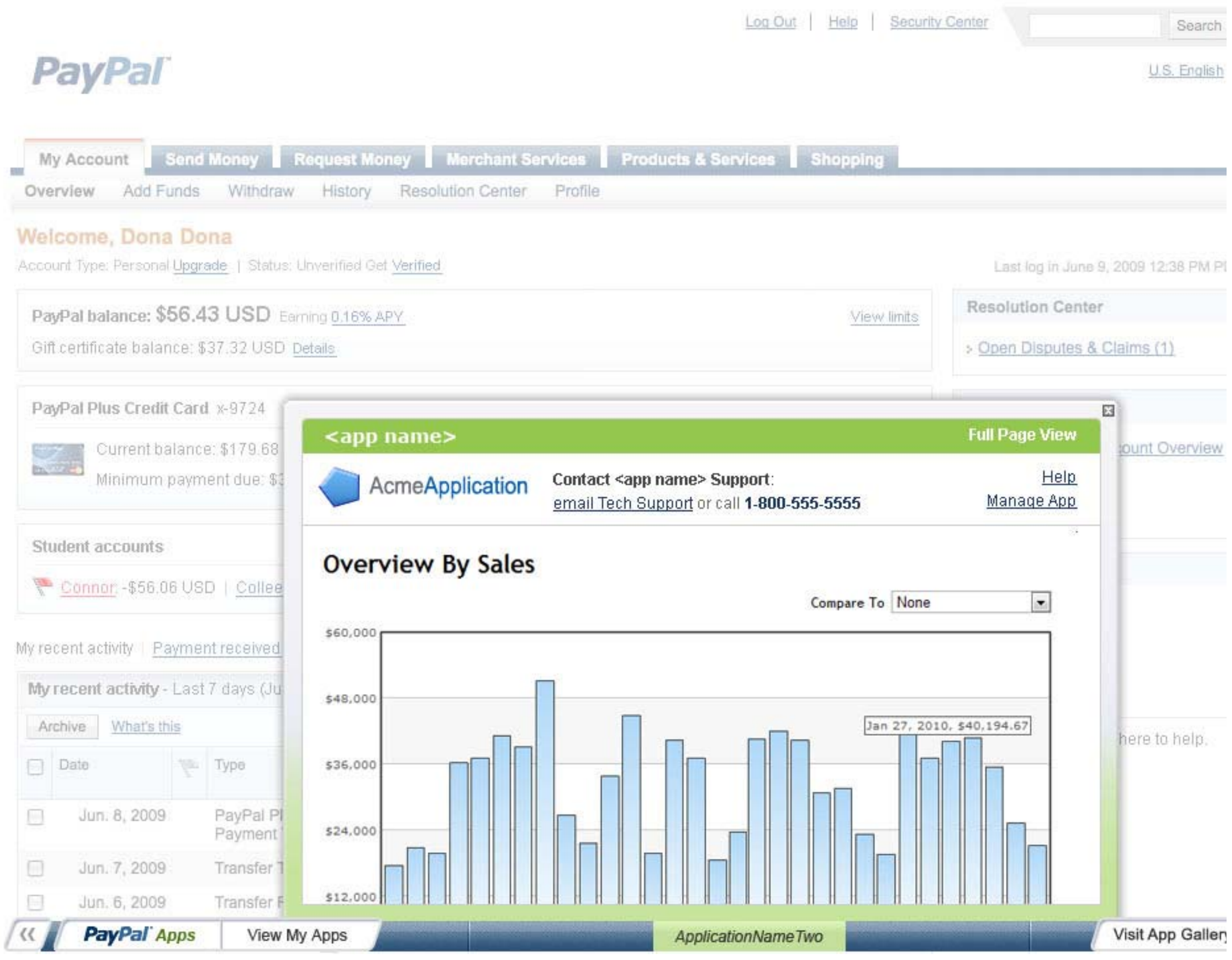
When an account holder launches your PayPal App, it covers the PayPal page from which it was launched.



A Widget View of a PayPal App

A widget view of a PayPal App is simply the full page version of the PayPal App scaled to display in a smaller frame. The PayPal account holder can expand the frame to become a full page.

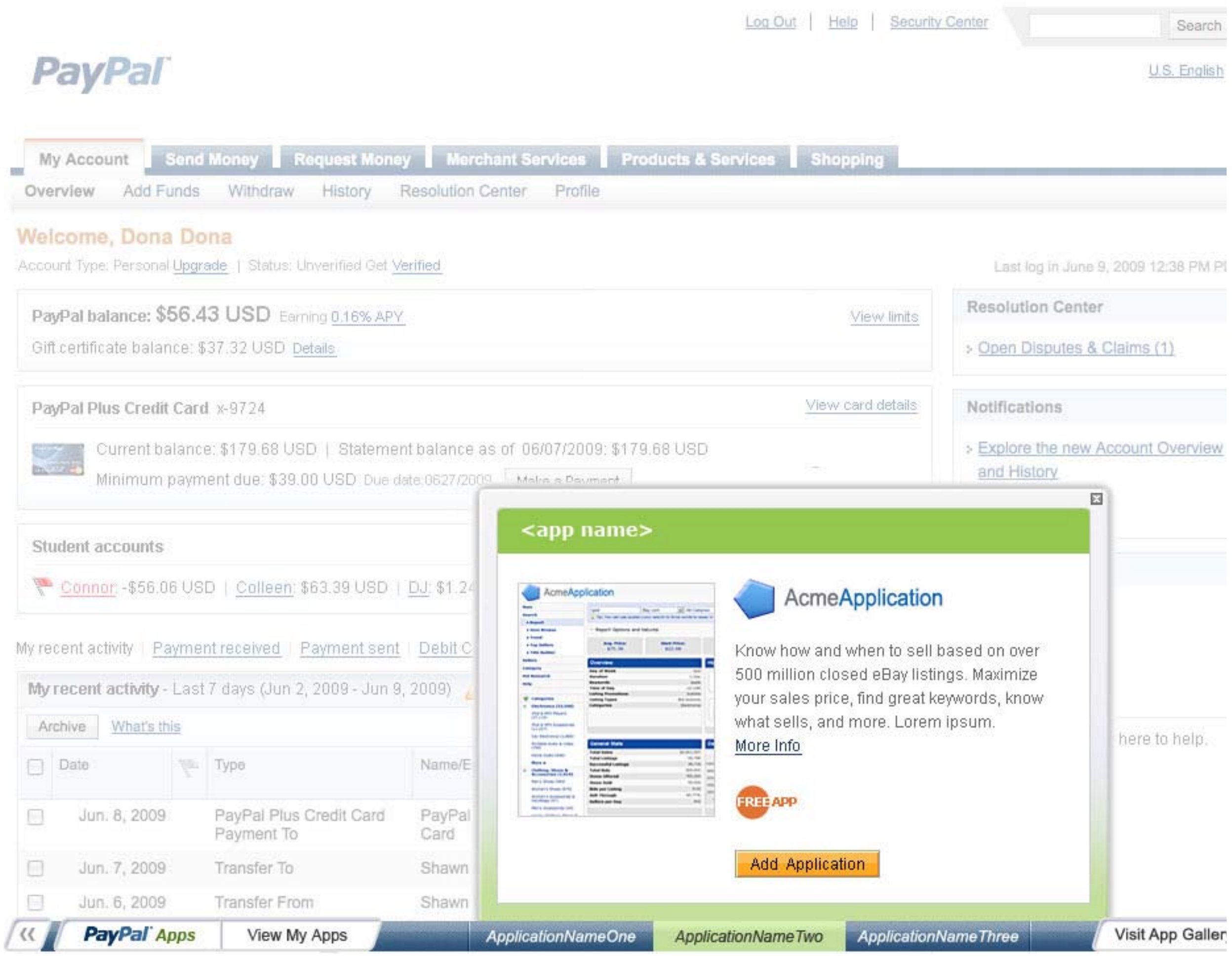
The following example shows a widget view of a PayPal App:



PayPal App Discovery

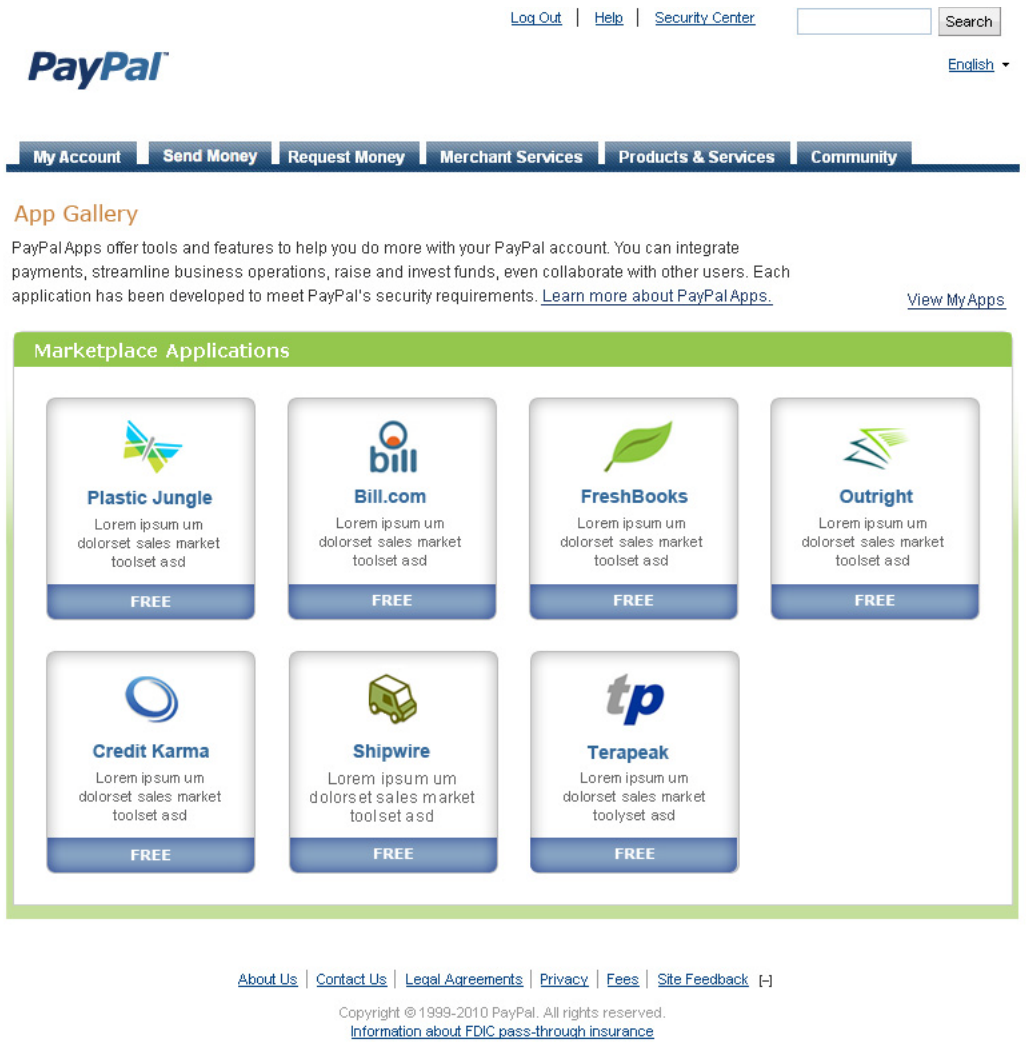
Information about your PayPal App can be reviewed by a PayPal account holder from either the App Tray at the bottom of a PayPal page or from the App Gallery. You provide PayPal with your logo and descriptive information, which PayPal formats in a consistent way for all PayPal Apps.

The following example shows the App Tray, from which your PayPal App can be discovered, activated, and launched.


Log Out | Help | Security Center
Search
PayPal
U.S. English
My Account
Send Money
Request Money
Merchant Services
Products & Services
Shopping
Overview
Add Funds
Withdraw
History
Resolution Center
Profile
Welcome, Dona Dona
Account Type: Personal Upgrade | Status: Unverified Get Verified
Last log in June 9, 2009 12:38 PM PI
PayPal balance: $56.43 USD Earning 0.16% APY
View limits
Gift certificate balance: $37.32 USD Details
Resolution Center
Open Disputes & Claims (1)
PayPal Plus Credit Card x-9724
View card details
Current balance: $179.68 USD | Statement balance as of 06/07/2009: $179.68 USD
Minimum payment due: $39.00 USD Due date 0627/2009
Make a Payment
Notifications
Explore the new Account Overview and History
Student accounts
Connor: -$56.06 USD | Colleen: $63.39 USD | DJ: $1.24
My recent activity | Payment received | Payment sent | Debit C
My recent activity - Last 7 days (Jun 2, 2009 - Jun 9, 2009)
Archive
What's this
Date
Type
Name/E
Jun. 8, 2009
PayPal Plus Credit Card Payment To
PayPal Card
Jun. 7, 2009
Transfer To
Shawn
Jun. 6, 2009
Transfer From
Shawn
here to help.
<app name>
AcmeApplication
More Info
FREE APP
Add Application
PayPal Apps
View My Apps
ApplicationNameOne
ApplicationNameTwo
ApplicationNameThree
Visit App Galler

Know how and when to sell based on over 500 million closed eBay listings. Maximize your sales price, find great keywords, know what sells, and more. Lorem ipsum.

The following example shows the App Gallery, by which your PayPal App can be discovered, activated, and launched.



Log Out | Help | Security Center

Search

English

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

App Gallery

PayPal Apps offer tools and features to help you do more with your PayPal account. You can integrate payments, streamline business operations, raise and invest funds, even collaborate with other users. Each application has been developed to meet PayPal's security requirements. Learn more about PayPal Apps.

View My Apps

Marketplace Applications

Plastic Jungle
Lorem ipsum um dolorset sales market toolset asd
FREE

Bill.com
Lorem ipsum um dolorset sales market toolset asd
FREE

FreshBooks
Lorem ipsum um dolorset sales market toolset asd
FREE

Outright
Lorem ipsum um dolorset sales market toolset asd
FREE

Credit Karma
Lorem ipsum um dolorset sales market toolset asd
FREE

Shipwire
Lorem ipsum um dolorset sales market toolset asd
FREE

Terapeak
Lorem ipsum um dolorset sales market toolyset asd
FREE

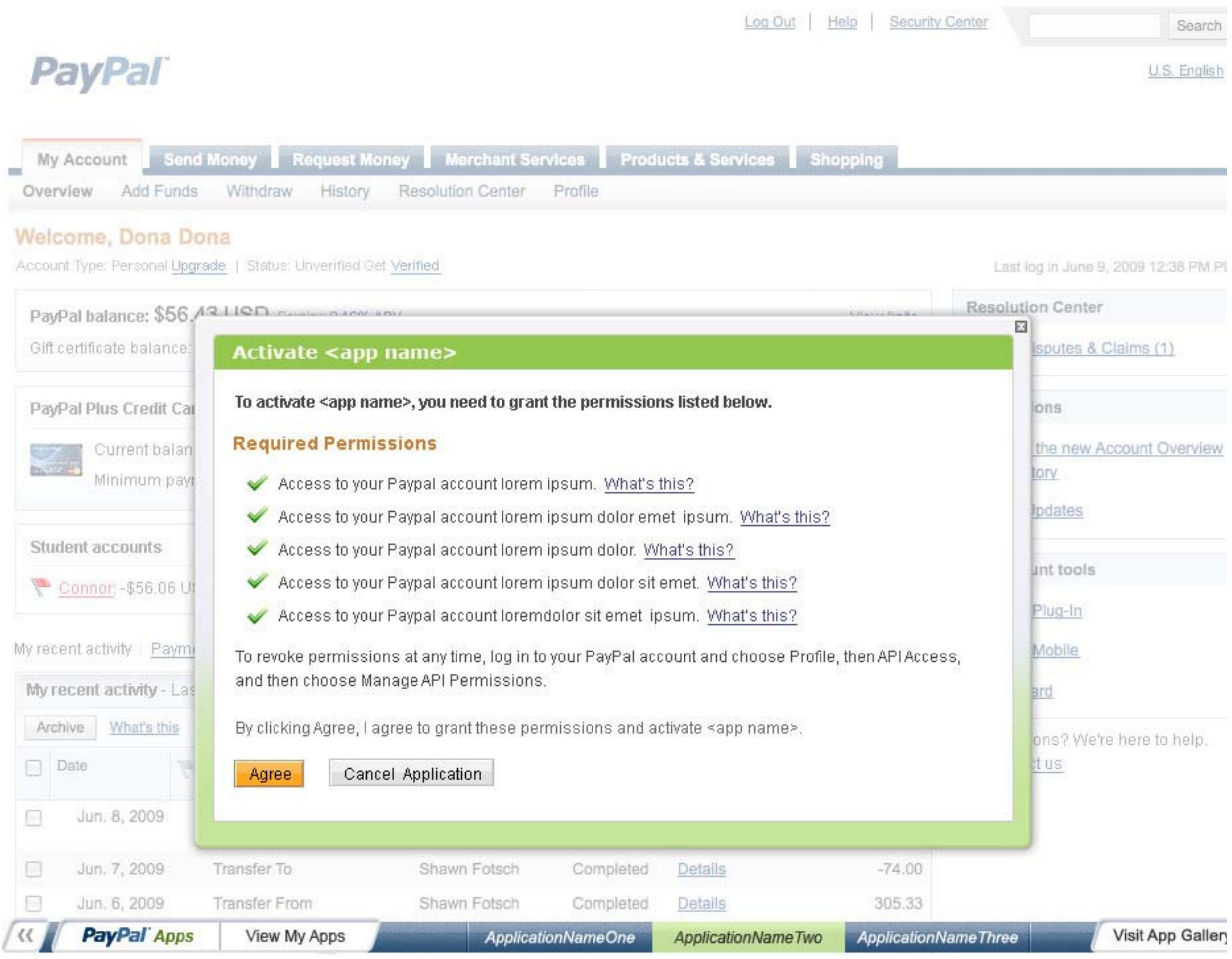
About Us | Contact Us | Legal Agreements | Privacy | Fees | Site Feedback [-]

Copyright © 1999-2010 PayPal. All rights reserved.
Information about FDIC pass-through insurance

PaPal App Activation

PayPal manages the PayPal account holder’s activation experience with your PayPal App. The activation should not require user input. PayPal handles granting of permissions for you. These permissions allow you to execute PayPal API operations on the account holder’s behalf.

The following example shows an the initial activation screen:



Activate <app name>

To activate <app name>, you need to grant the permissions listed below.

Required Permissions

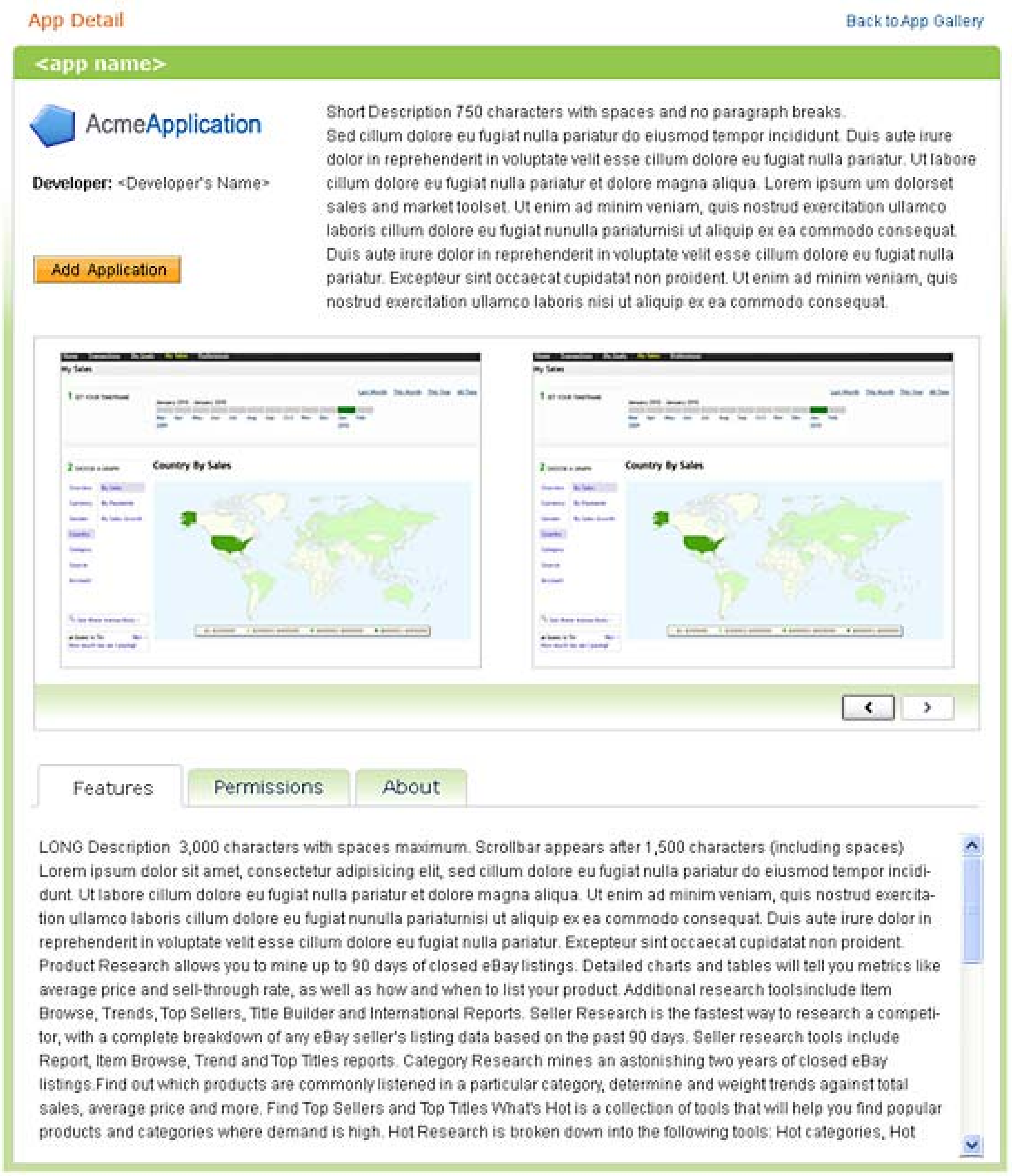
- ✓ Access to your Paypal account lorem ipsum. What's this?
- ✓ Access to your Paypal account lorem ipsum dolor emet ipsum. What's this?
- ✓ Access to your Paypal account lorem ipsum dolor. What's this?
- ✓ Access to your Paypal account lorem ipsum dolor sit emet. What's this?
- ✓ Access to your Paypal account loremdolor sit emet ipsum. What's this?

To revoke permissions at any time, log in to your PayPal account and choose Profile, then API Access, and then choose Manage API Permissions.

By clicking Agree, I agree to grant these permissions and activate <app name>.

If you are calling PayPal APIs on behalf of the account holder, you specify the permissions you need when you submit your PayPal App to PayPal. When requesting permissions, PayPal provides a non-technical description for each of its API operations or groups of related API operations

PayPal notifies the account holder when the PayPal App is available to launch:



Developer: <Developer's Name>

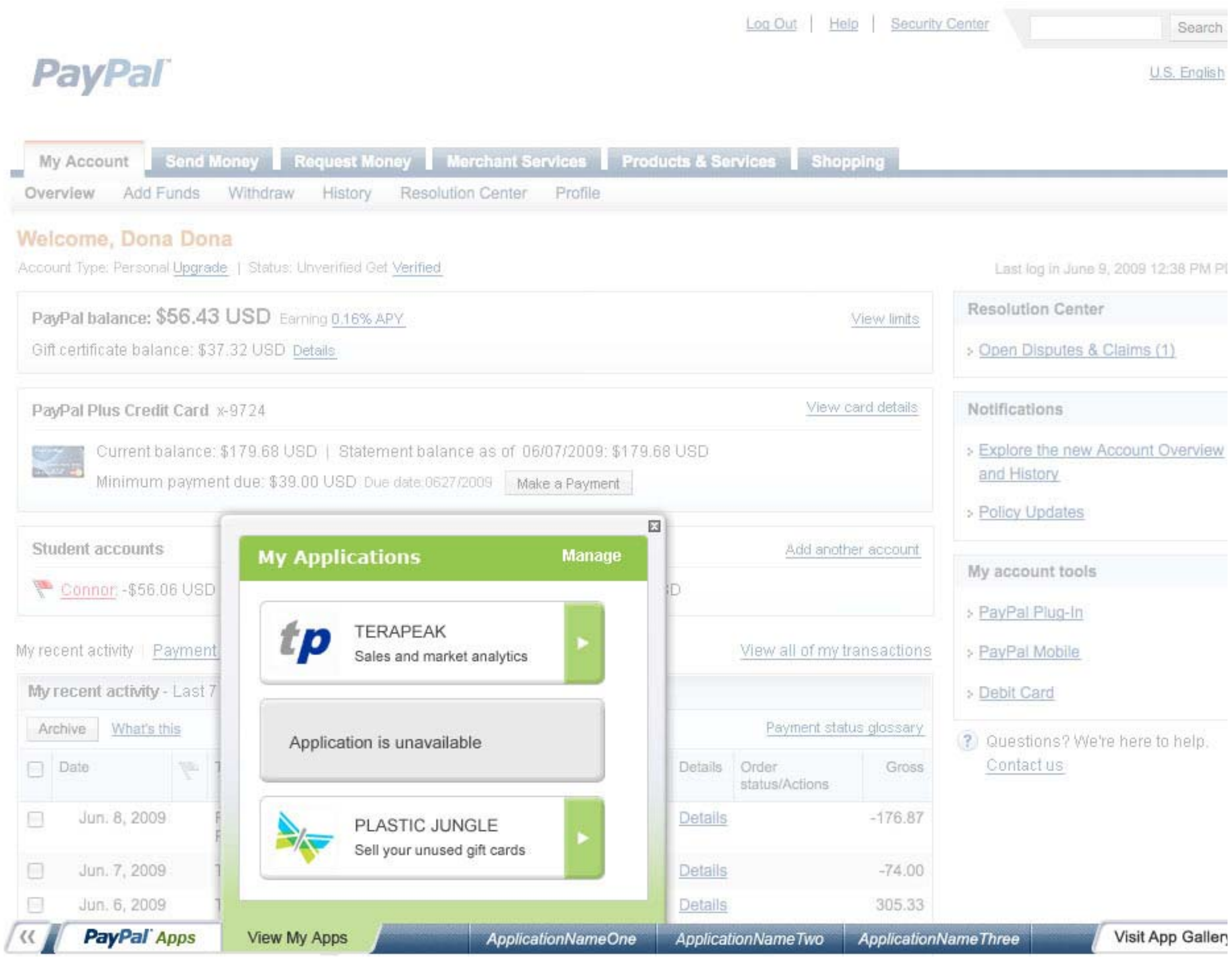
Short Description 750 characters with spaces and no paragraph breaks.
Sed cillum dolore eu fugiat nulla pariatur do eiusmod tempor incididunt. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Ut labore cillum dolore eu fugiat nulla pariatur et dolore magna aliqua. Lorem ipsum um dolorset sales and market toolset. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris cillum dolore eu fugiat nunulla pariaturnisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

LONG Description 3,000 characters with spaces maximum. Scrollbar appears after 1,500 characters (including spaces) Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed cillum dolore eu fugiat nulla pariatur do eiusmod tempor incididunt. Ut labore cillum dolore eu fugiat nulla pariatur et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris cillum dolore eu fugiat nunulla pariaturnisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident. Product Research allows you to mine up to 90 days of closed eBay listings. Detailed charts and tables will tell you metrics like average price and sell-through rate, as well as how and when to list your product. Additional research toolsinclude Item Browse, Trends, Top Sellers, Title Builder and International Reports. Seller Research is the fastest way to research a competitor, with a complete breakdown of any eBay seller's listing data based on the past 90 days. Seller research tools include Report, Item Browse, Trend and Top Titles reports. Category Research mines an astonishing two years of closed eBay listings.Find out which products are commonly listened in a particular category, determine and weight trends against total sales, average price and more. Find Top Sellers and Top Titles What's Hot is a collection of tools that will help you find popular products and categories where demand is high. Hot Research is broken down into the following tools: Hot categories, Hot

PayPal App Management

A PayPal account holder can view his or her PayPal Apps from the App Tray at the bottom of most pages. You provide PayPal with your logo and descriptive information, which PayPal formats in a consistent way for all PayPal Apps.

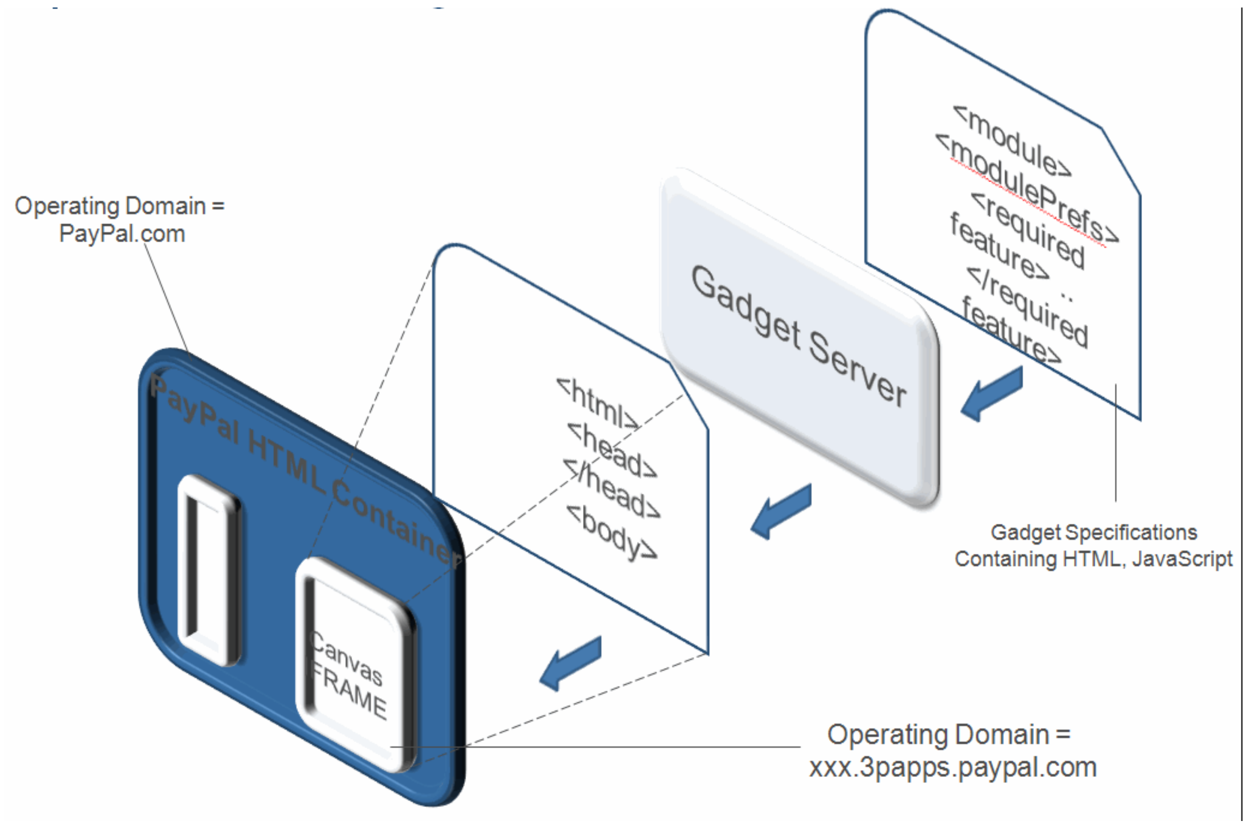
The following example shows the PayPal Apps that have been activated by a PayPal account holder.



2 PayPal App Execution Architecture

The PayPal execution architecture for PayPal Apps supports HTML gadgets. You can create a PayPal App by packaging it inside the appropriate XML elements, called a *deployment descriptor* or *deployment spec*.

The following diagram shows a conceptual model of an HTML gadget:



Operating Domain = PayPal.com
PayPal HTML Container
Canvas FRAME
<html> <head> </head> <body>
Gadget Server
<module> <modulePrefs> <required feature> .. </required feature>
Gadget Specifications Containing HTML, JavaScript
Operating Domain = xxx.3papps.paypal.com

The following example is the most basic PayPal App:

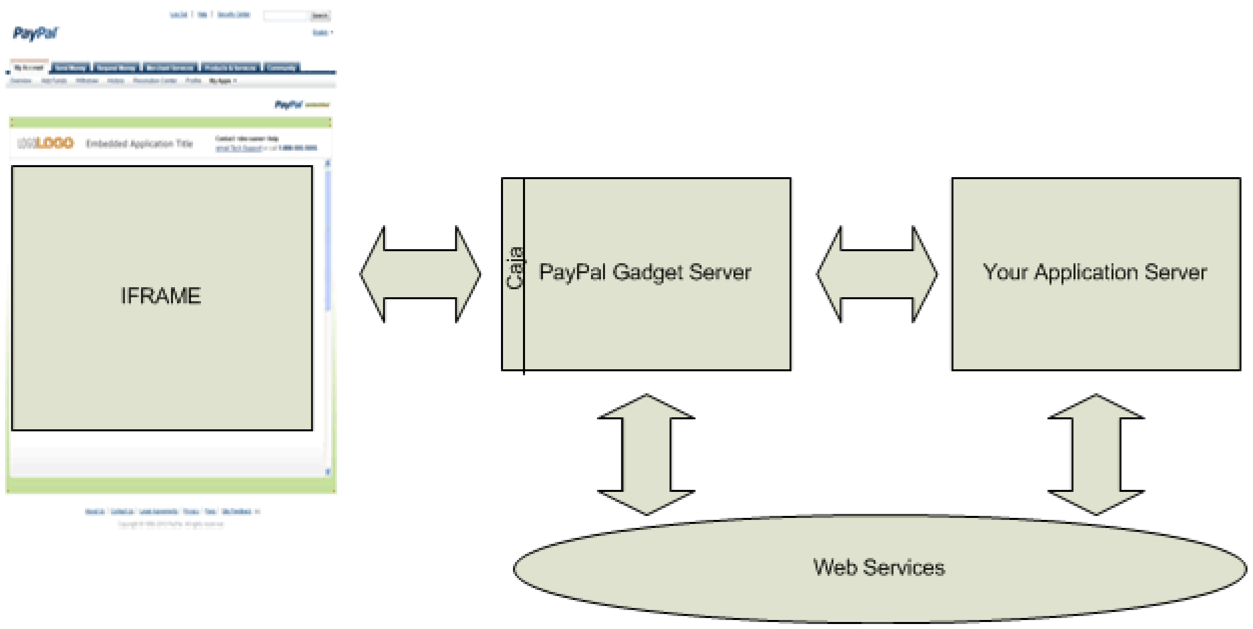
```
<?xml version="1.0" encoding="UTF-8" ?>
 <Module>
   <ModulePrefs title="Hello PayPal App World!" />
   <Content type="html">
     <![CDATA[
       Hello, PayPal App World!
     ]]>
   </Content>
 </Module>
```

In this example, the PayPal App is an HTML page. If it were an actual web page, the page would look something like this:

```
<!DOCTYPE html ...>
<html ...>
<head>
<meta ... />
<title>Hello World!</title>
</head>
<body>
       Hello, PayPal App World!
</body>
</html>
```

For more information about the Open Social framework, see http://www.opensocial.org/. For more information about the Google Gadgets API, see http://code.google.com/apis/gadgets/.

The PayPal execution architecture for HTML gadgets places a gadget server between PayPal pages and the pages in a PayPal App. The gadget server uses Caja technology to protect account holders from unauthorized access to information. The following diagram shows the relationship between PayPal, your PayPal App, and web services:



Your PayPal App displays in an HTML IFRAME on the PayPal page. Your code is rendered by the PayPal gadget server. The gadget output is sanitized for security using Caja. Caja sanitization rewrites HTML, CSS, and JavaScript. It uses static analysis techniques to preclude unsafe DOM or cookie access to the gadget content, while maintaining expressiveness nearly equivalent to content running simply inside an IFRAME. For more information about Caja, see http://code.google.com/p/google-caja/.

Your PayPal App can obtain access to external data using web services, either directly through the PayPal Gadget Server or indirectly using your application server.

Typically, you will use an HTML gadget, which might require you to modify existing applications in the following ways:

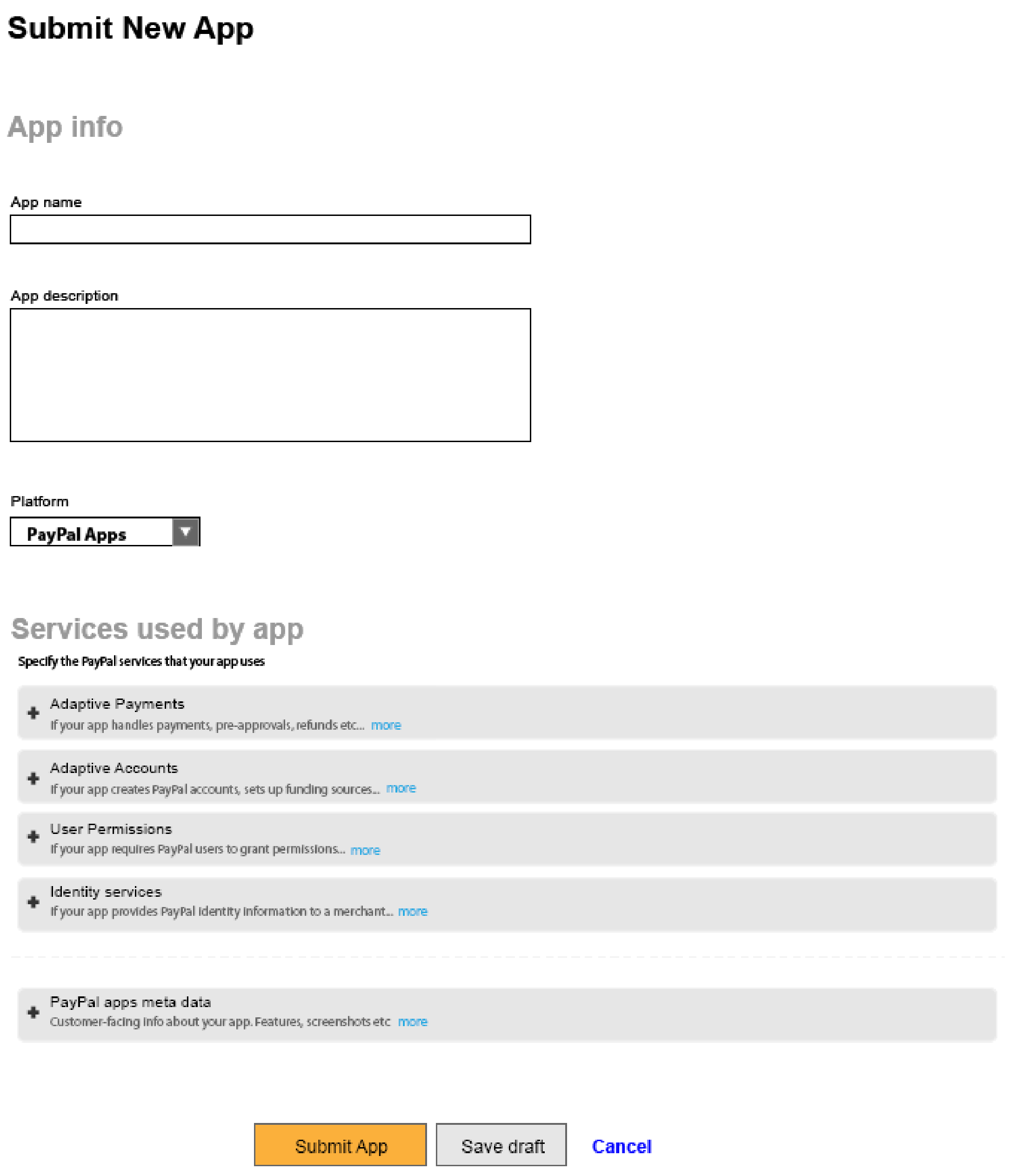
- Inline JavaScript and cascading style sheets (CSS) inside the body of an HTML page
- Perform authentication by calling PayPal API operations

3 Testing and Submitting Your PayPal App

You submit your PayPal App from x.com, which allows you to test you PayPal App in the PayPal Sandbox. From the Sandbox, you can confirm submission, after which PayPal reviews and approves your PayPal App and allows it to go live.

NOTE: In the event that your PayPal App does not render in the Sandbox, you may wish to test the gadget spec in a Shindig 1.1 container. For more information, see http://shindig.apache.org/getting-started.html

When you submit your PayPal App on x.com, choose **PayPal Apps** from the **Platform** drop down menu:



Submit New App

App info

App name

App description

Platform
PayPal Apps

Services used by app
Specify the PayPal services that your app uses

Adaptive Payments
If your app handles payments, pre-approvals, refunds etc... more

Adaptive Accounts
If your app creates PayPal accounts, sets up funding sources... more

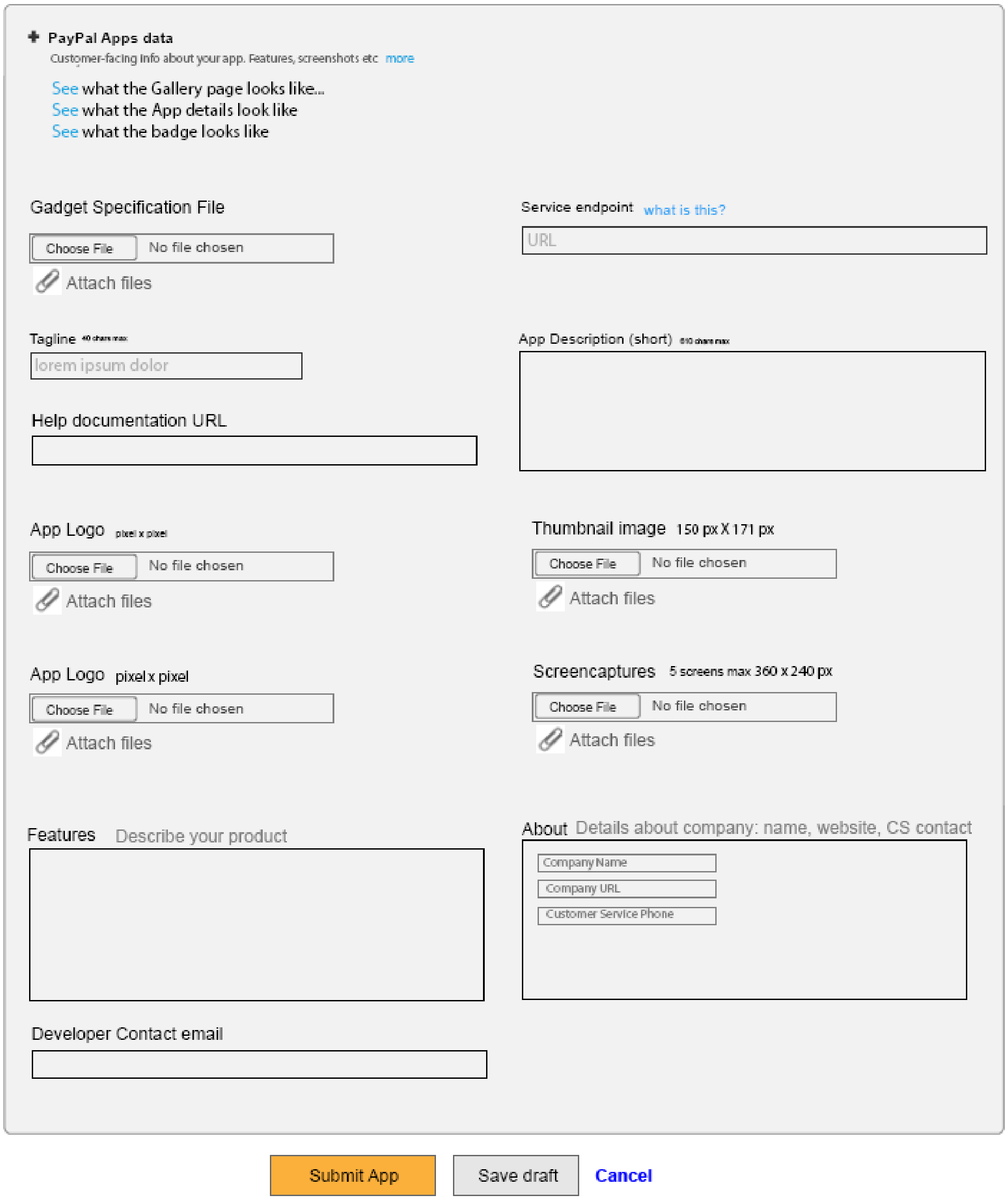
User Permissions
If your app requires PayPal users to grant permissions... more

Identity services
If your app provides PayPal identity information to a merchant... more

PayPal apps meta data
Customer-facing info about your app. Features, screenshots etc more

On the first **Submit New App** page, you provide basic information about your PayPal App to allow PayPal to set up automated testing. You also specify the PayPal API permissions that you need from the account holder that runs your PayPal App. These permissions are called *third-party permissions* because you intend execute the APIs for which permissions are granted as a third-party to the account holder and PayPal.

You then upload the text and graphics to be displayed in your PayPal App, on the App Tray and in the App Gallery. You also upload the gadget metadata that contains the HTML to render your PayPal App on `paypal.com`. The following **Submit New App** page shows the items to upload:



✚ PayPal Apps data

Customer-facing info about your app. Features, screenshots etc more

See what the Gallery page looks like...
See what the App details look like
See what the badge looks like

Gadget Specification File

Choose File No file chosen

Attach files

Service endpoint what is this?

URL

Tagline 40 chars max

lorem ipsum dolor

App Description (short) 610 chars max

Help documentation URL

App Logo pixel x pixel

Choose File No file chosen

Attach files

Thumbnail image 150 px X 171 px

Choose File No file chosen

Attach files

App Logo pixel x pixel

Choose File No file chosen

Attach files

Screencaptures 5 screens max 360 x 240 px

Choose File No file chosen

Attach files

Features Describe your product

About Details about company: name, website, CS contact

Company Name

Company URL

Customer Service Phone

Developer Contact email

Submit App Save draft Cancel

Before you click **Submit App**, you can examine the items you upload, which allows you to change them if you are not satisfied.

4 PayPal App User Interface Guidelines

Follow the PayPal App user interface guidelines to provide the best experience for your users when they execute your PayPal App. Failure to follow these guidelines could require corrective action before it can be hosted.

General User Interface Guidelines

Follow these guidelines on each page in your PayPal App:

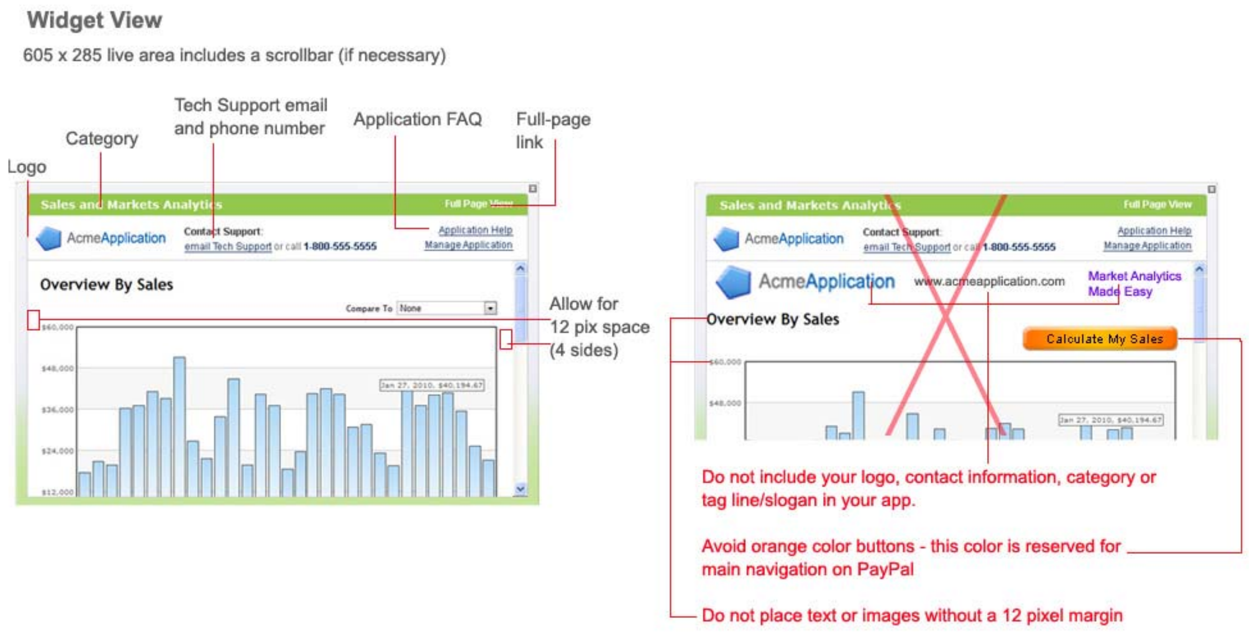
- Avoid orange buttons within your pages unless they initiate the preferred action to conform with the way that PayPal buttons initiate actions.

Widget View Guidelines

Follow these guidelines for the widget view:

- Your widget view must be 605 pixels wide by 285 pixels high, including space for a vertical scroll bar if necessary.

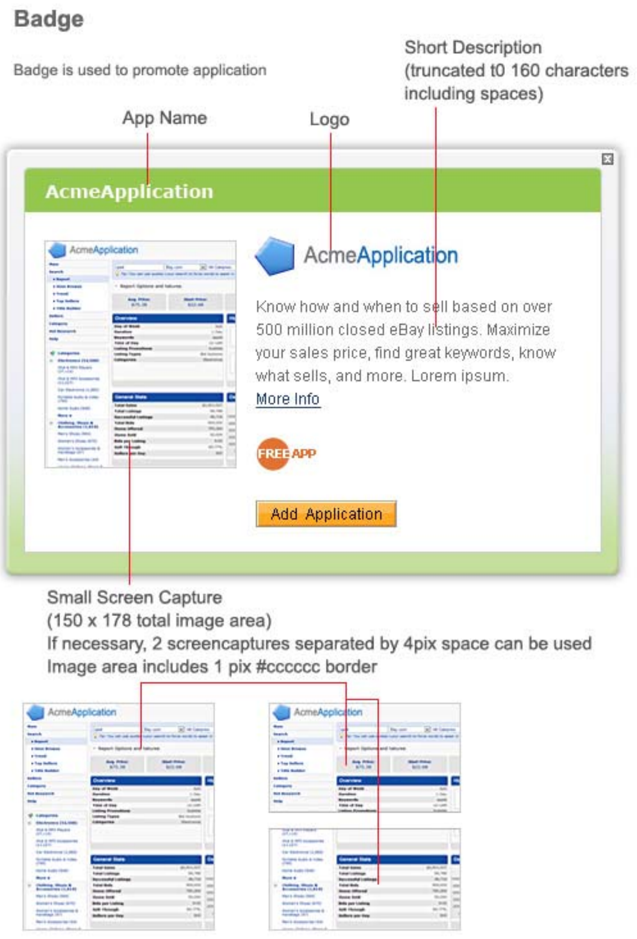
  NOTE: Horizontal scrolling is not allowed.
- If you support vertical scrolling, reserve 12 pixels for a scroll bar on the right-hand-side of your page.
- Reserve 4 pixels on each side of your page as white space.
- Do not place your company logo, company icon, contact information, category, and tag line or slogan in your pages. These items will be rendered by PayPal outside of the IFRAME that displays your page. Including them will make them appear redundant.


Widget View
605 x 285 live area includes a scrollbar (if necessary)
Logo
Category
Tech Support email and phone number
Application FAQ
Full-page link
Allow for 12 pix space (4 sides)
Sales and Markets Analytics
Full Page View
AcmeApplication
Contact Support:
email Tech Support or call 1-800-555-5555
Application Help
Manage Application
Overview By Sales
Compare To None
www.acmeapplication.com
Market Analytics Made Easy
Calculate My Sales
Do not include your logo, contact information, category or tag line/slogan in your app.
Avoid orange color buttons - this color is reserved for main navigation on PayPal
Do not place text or images without a 12 pixel margin

Badge View Guidelines

Follow these guidelines for the badge view:

- Provide 1 or 2 graphics whose total size is exactly 150 pixels wide by 171 pixels high.
- If you provide 1 graphic, you must provide a 1-pixel border on the left and right sides; thus, the width is the sum of a 1-pixel border on the left, 148 pixels for the graphic, and a 1-pixel border on the right.
- If you provide 2 graphics, the total height of both must not equal 166 pixels because the graphics will be separated by 5 pixels of white space between them.


Badge
Badge is used to promote application
App Name
Logo
Short Description (truncated t0 160 characters including spaces)
AcmeApplication
AcmeApplication
Know how and when to sell based on over 500 million closed eBay listings. Maximize your sales price, find great keywords, know what sells, and more. Lorem ipsum.
More Info
FREE APP
Add Application
Small Screen Capture
(150 x 178 total image area)
If necessary, 2 screencaptures separated by 4pix space can be used
Image area includes 1 pix #cccccc border
AcmeApplication
AcmeApplication

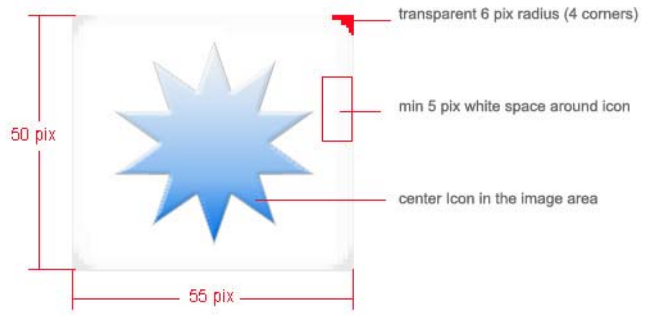
Icon Guidelines

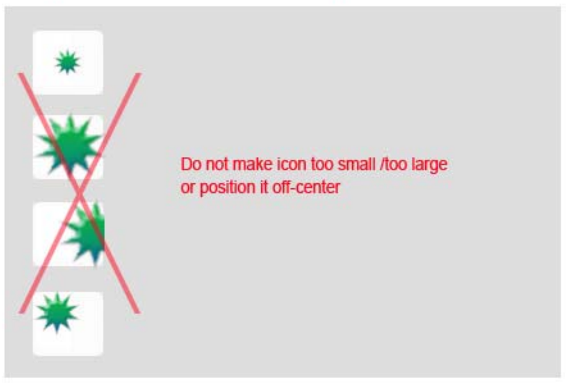
Follow these guidelines for your company icon:

- Your icon must be a transparent GIF on a white background, exactly 55 pixels wide by 50 pixels high.
- The transparency radius must be 6 pixels.
- The graphic image inside the icon must be centered left-to-right and top-to bottom.
- You must provide 5 pixels of white space on each side of the icon.

ICON

All icons must be supplied as TRANSPARENT GIF files on white background
Size is 55 pix by 50pix
Minimum 5 pix white space on all sides
Radius (transparency) 6 pixels


transparent 6 pix radius (4 corners)
min 5 pix white space around icon
50 pix
center Icon in the image area
55 pix


Do not make icon too small /too large
or position it off-center

Logo Guidelines

Follow these guidelines for your company logo:

- Your logo must be a transparent GIF on a white background, exactly 30 pixels high.
- The logo must be trimmed so that it exactly fits the width of the image, which must be 160 pixels.

Logo

All logos must be supplied as non-transparent GIF files on white background
160 pix by 30pix
Position logo Left Justified. Do not trim to fit!

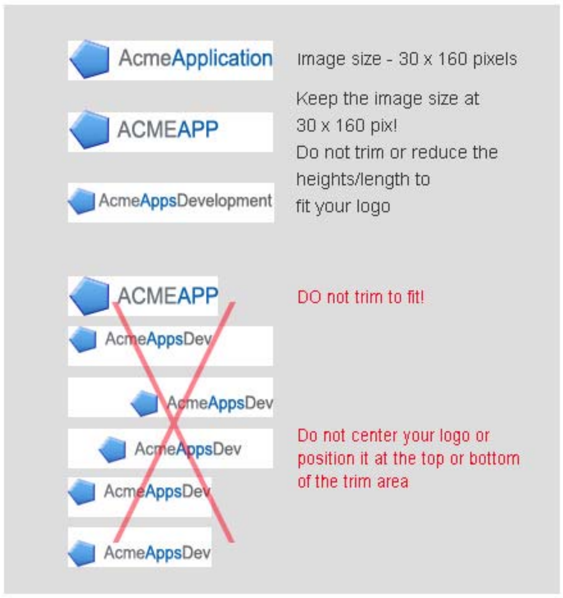


Image size - 30 x 160 pixels

Keep the image size at
30 x 160 pix!
Do not trim or reduce the
heights/length to
fit your logo

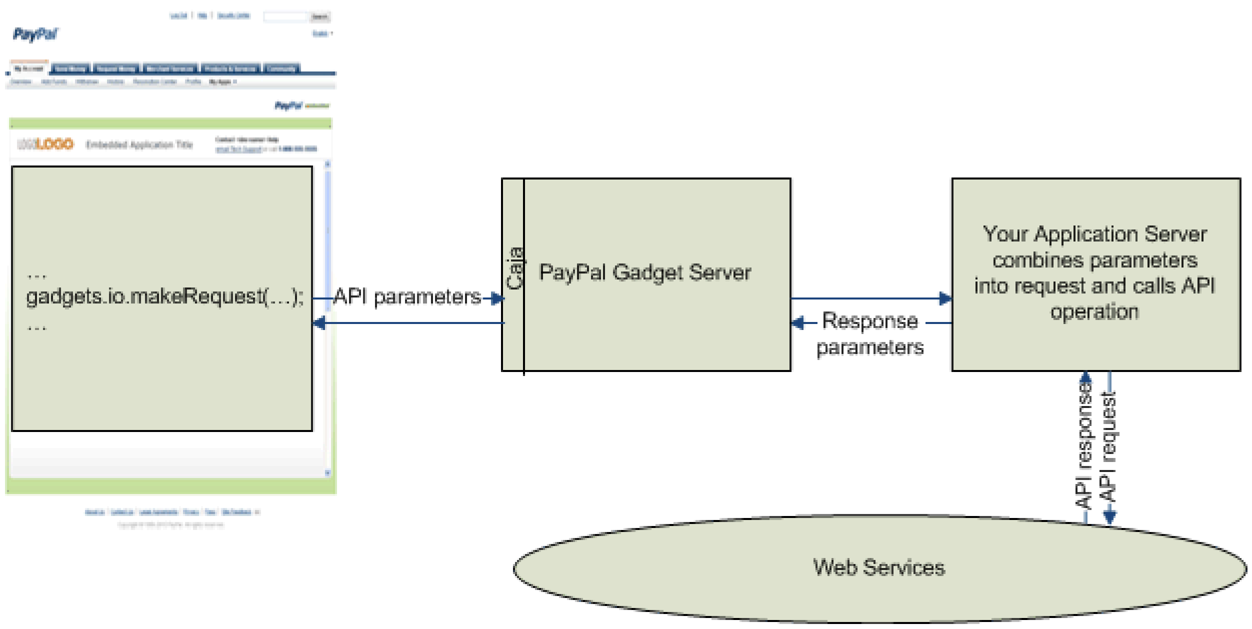
DO not trim to fit!

Do not center your logo or
position it at the top or bottom
of the trim area

5 Invoking PayPal APIs From Your PayPal App

To call a PayPal API operation from your PayPal App, you must collect parameters needed by the API and invoke the PayPal gadget server to deliver the parameters to your server. Your server then makes the actual API call and returns the response to your PayPal App.

The following diagram shows the general flow of your API request from your PayPal App to your server, which makes the request and delivers the response to your PayPal App.


...
gadgets.io.makeRequest(...);
...
API parameters
Caja
PayPal Gadget Server
Response parameters
Your Application Server combines parameters into request and calls API operation
API response
API request
Web Services

Set parameters associated with the API operations request message and pass them to your server, which executes the API operation, using the `gadgets.io.makeRequest` API operation.

IMPORTANT: You must use the JSON data model in requests in responses to `gadgets.io.makeRequest`; Caja does not allow you to use XML.

The PayPal gadget server also provides you with the account holder’s email address and Payer ID to identify the account holder making the invocation.

Supported JavaScript Components

You should use the Caja-supported YUI 2.8 JavaScript Library to implement JavaScript within your HTML pages. Some JavaScript constructs are restricted.

The following components are supported:

- Core components
  - Yahoo! Global Object (base requirement for all YUI components)
  - DOM Collection (convenience methods for DOM interactions)
  - Event Utility (event normalization and custom events)
- Controls and widgets
  - AutoComplete
  - Button
  - Container, including Module, Overlay, Panel, Tooltip, Dialog, and SimpleDialog
  - Menu
  - TreeView
- Utilities
  - Animation Utility
  - Connection Manager (for XHR / AJAX)
  - DataSource Utility
  - Drag and Drop Utility
  - Element Utility
  - ImageLoader Utility
  - Resize Utility
  - Selector Utility

For information about the YUI Library, see http://developer.yahoo.com/yui/.

The following JavaScript constructs cannot be used:

- `this` if it promotes to global object
- DOM events (must be elements in DOM)
- `with`
- `eval()`
- `javascript:void(0)`
- `delete` variable
- `aFunction`.caller or `aFunction`.arguments
- `arguments`.caller or `arguments`.callee
- `try`/`catch` with scope leakage
- `Function()` constructors
- Identifiers ending in __
- Reflective attribute control
- Joining arguments with parameter list

Content Restrictions for PayPal Apps

Some HTML constructs cannot be used, because they could compromise security. Caja enforces client-side sanitization, which may strip out or inline HTML code, such as style sheets and scripts. There are also additional restrictions.

Do not use the following HTML constructs:

- targets except for `<a target=...>`, `target=_blank`, and `target=_top`.
- `<embed>` For Flash, use JavaScript instead; requires permission from PayPal.
- `<head>` Use plain HTML starting from inside the body.
- `<iframe>` (temporary restriction)
- `<link rel=stylesheet>` (Temporary restriction; inline the style sheet)
- `<object>` For Flash, use JavaScript with `swfobject`; other objects, such as ActiveX are not allowed.
- `<script>` on the client side; using inline scripts on the server-side is allowed. (Caja inlines them on your site.)
- `<script src=...>` (temporary restriction).
- `<style>` on the client side; its use on the server-side is allowed. (Caja inlines them on your site.)

In addition to HTML restrictions, the PayPal Apps are restricted in the following ways:

- jQuery is not supported.
- Charting is not supported.
- Flash is supported only with permission from PayPal.
- Remote calls are restricted to only your domain.
- Opening windows to a predefined URL.

NOTE: Caja currently supports only Shindig 1.1.

PayPal Invocation Parameters For Your PayPal App

When a PayPal account holder invokes your PayPal App, PayPal uses the Open Authorization (OAuth) protocol to provide sufficient information for you to verify that the requestor is PayPal. PayPal also provides you with the account holder's email address and Payer ID to identify the account holder making the invocation.

Parameter	Description	Example Value
`paypal_payer_id`	PayPal account holder's Payer ID	PP12345678
`opensocial_app_id`	Your PayPal App ID	terapeak12345

Parameter	Description	Example Value
`opensocial_app_url`	Reserved for future use	`https://gs.paypal3papps.com`
`opensocial_viewer_id`	PayPal account holder's email address	`foo@paypal.com`
`oauth_consumer_key`	A uniform resource indicator (URI) that specifies the PayPal callback name for the PayPal App	`paypal3papps.com`
`oauth_signature_method`	Open Authorization signature type	`RSA-SHA1`
`oauth_timestamp`	A timestamp expressed in the number of seconds since January 1, 1970 00:00:00 GMT; used to limit	`1276046551`
`oauth_nonce`	A random string, uniquely generated for each authorization request with the same timestamp	`1052638374415123`
`oauth_version`	The Open Authorization version; must be `1.0`	`1.0`
`oauth_signature`	PayPal generated signature	`abYtkh/O3uehYgK/c...DBq2Ij5sDcxGU=`

For example, PayPal invokes your URL-based gadget with the these parameters, as follows:

```
http://your_app.your_domain.com?paypal_payer_id=PP12345678&opensocial_app_id=
terapeak12345&opensocial_app_url=https%3A%2F%2Fgs.paypal3papps.com&opensoci
al_viewer_id=foo%40paypal.com&oauth_consumer_key=paypal3papps.com&oauth_sig
nature_method=RSA-
SHA1&oauth_timestamp=1276046551&oauth_nonce=1052638374415123&oauth_version=
1.0&oauth_signature=abYtkh%2FO3uehYgK%2FcVH5fxSdJ3uXgl7LttxJyIQwVxx%2Ft6dpi
vn9%2FLXxRzwzm2P%2FfvJLb0yuh5CL6gksU5vOzO%2BrzxmgSHs3Jb7KJNOpM09%2B1LaJHGlz
U4zW7YTpoHd2AVj2pcPz4H89I4p5R0PqQol8syxUEDBq2Ij5sDcxXGU%3D
```

To implement the Open Authentication protocol, you must

- verify that the signature in `oauth_signature` is valid using the protocol specified in `oauth_signature_method` and a public key downloaded from PayPal. You should compute your own signature and compare it with the signature in `oauth_signature`. A match indicates that the parameters have not been altered.
- verify that the timestamp is recent; for example, within 5 minutes.
- verify that the invocation request is not being replayed by comparing the value in `oauth_nonce` with a database of values; a match indicates that the request has already occurred.

In addition, you should verify the consistency of the request; for example, you should check that the application ID in `opensocial_app_id` matches your PayPal-assigned PayPal App ID.

PayPal APIs Available From PayPal Apps

PayPal APIs consist of the Adaptive Payments API, the Adaptive Accounts API, the Website Payments Pro API, which includes the Express Checkout API, Button Manager API, and the Payflow API. Not all APIs are available from PayPal Apps.

IMPORTANT: You can use PayPal APIs in your PayPal App if they do not redirect to PayPal or require the transmission of sensitive information, such as credit card numbers.

Your PayPal App can use the following PayPal APIs:

API Operation	Description
Adaptive Payments API operations:	
`Pay`	Transfers funds from a sender's PayPal account to one or more receivers' PayPal accounts (up to 6 receivers)
`PaymentDetails`	Obtains information about a payment set up with the `Pay` API operation
`ExecutePayment`	Executes a payment
`GetPaymentOptions`	Obtain the settings specified with the `SetPaymentOptions` API operation
`SetPaymentOptions`	Sets payment options
`CancelPreapproval`	Cancels a preapproval
`Refund`	Refunds all or part of a payment
`ConvertCurrency`	Obtains the current foreign exchange (FX) rate for a specific amount and currency
`GetFundingPlans`	Determines the funding sources that are available for a specified payment
`GetShippingAddresses`	Obtains the selected shipping address
Adaptive Accounts API operations:	
`AddBankAccount`	Link bank accounts to PayPal accounts as funding sources.
`AddPaymentCard`	Link payment cards to PayPal accounts as funding sources
`SetFundingSourceConfirmed`	Set the funding source to confirmed; they may set the account to PayPal Verified status.
`GetVerifiedStatus`	Verify PayPal accounts by matching account holder criteria such as the account holder's email address.
`GetUserAgreement`	Obtains the PayPal End User License Agreement (EULA) associated with the country of an account. (mobile only)
Common API operations: (Direct Payment and Express Checkout only)	
`GetTransactionDetails`	Obtain information about a specific transaction.

API Operation	Description
`ManagePendingTransactionStatus`	Accept or deny a pending transaction held by Fraud Management Filters.
`TransactionSearch`	Search transaction history for transactions that meet the specified criteria.
Recurring Payment API operations:	
`ManageRecurringPaymentsProfileStatus`	Cancel, suspend, or reactivate a recurring payments profile.
`BillOutstandingAmount`	Bill the buyer for the outstanding balance associated with a recurring payments profile.
`UpdateRecurringPaymentsProfile`	Update a recurring payments profile.
`DoReferenceTransaction`	Process a payment from a buyer's account, which is identified by a previous transaction.
Recurring Payment Billing Agreement API operations: (Express Checkout only)	
`BAUpdate`	Update or delete a billing agreement.
`GetBillingAgreementCustomerDetails`	Obtain information about a billing agreement's PayPal account holder.
`SetCustomerBillingAgreement`	Initiates the creation of a billing agreement.
Other Express Checkout API operations: (Express Checkout only)	
`AddressVerify`	Confirms whether a postal address and postal code match those of the specified PayPal account holder. (Express Checkout only)
`GetBalance`	Obtain the available balance for a PayPal account. (Express Checkout only)
`GetPalDetails`	Obtain your Pal ID, which is the PayPal-assigned merchant account number, and other information about your account.
`MassPay`	Make a payment to one or more PayPal account holders.
Button Manager API operations:	
`BMCreateButton`	Create button code
`BMUpdateButton`	Update a hosted button
`BMManageButtonStatus`	Delete a button
`BMGetButtonDetails`	Obtain information about the parameters associated with a button
`BMButtonSearch`	Obtain a list of all buttons
`BMGetInventory`	Obtain the inventory levels associated with a button
`BMSetInventory`	Specify the inventory levels associated with a button

6 Setting Up Your PayPal App to Enable Payments

This example shows how to set up your PayPal App to enable payments using the Adaptive Payments Pay API operation.

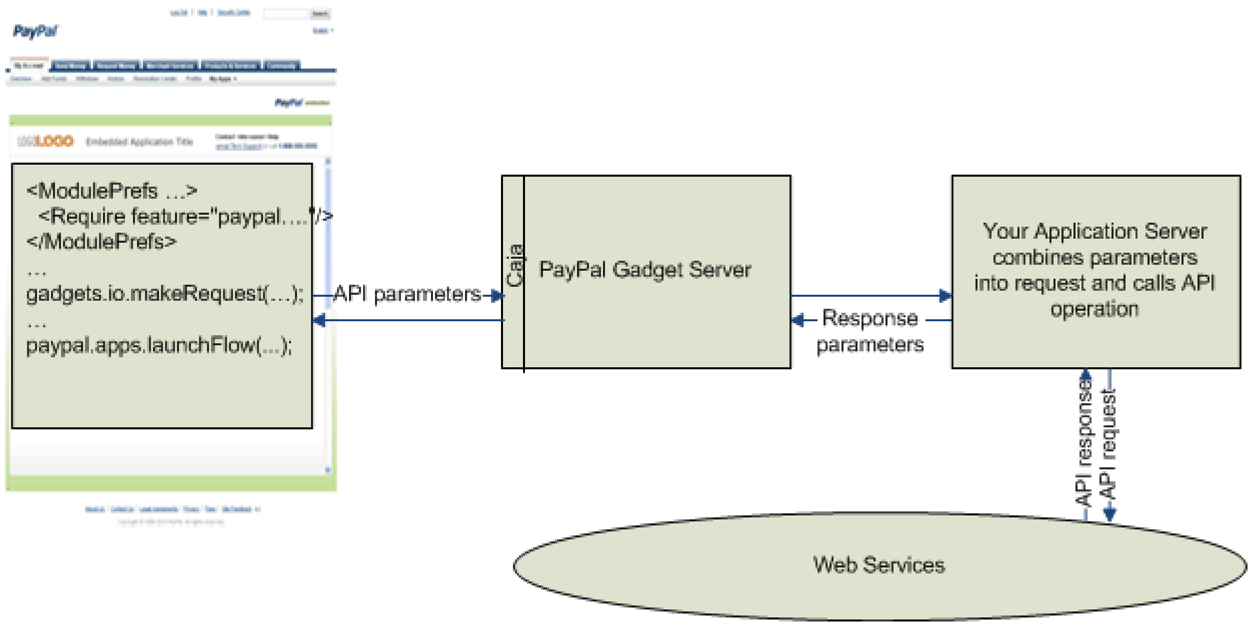
This example assumes that you have a working server that responds to your payment requests. These requests are passed to your server via the PayPal gadget server using the gadgets.io.MakeRequest method.

These instructions only describe the steps to include JavaScript code in your PayPal App for invoking the PayPal gadget server and responding to the response. It does not show how to set up the Pay call on your server; however, there are several important requirements for calling the Pay API operation from your App:

- You cannot specify `senderEmail`.
- You cannot use a preapproval.
- You cannot make an implicit payment.
- You cannot use a funding constraint.

For more information about Adaptive Payments and the Pay API operation, see https://cms.paypal.com/cms_content/US/en_US/files/developer/PP_AdaptivePayments.pdf.

The following diagram shows the general flow of your API request from your PayPal App to your server, which makes the request and delivers the response to your App.



<ModulePrefs ...>
<Require feature="paypal...."/>
</ModulePrefs>
...
gadgets.io.makeRequest(...);
...
paypal.apps.launchFlow(...);
API parameters
Caja
PayPal Gadget Server
Response parameters
Your Application Server combines parameters into request and calls API operation
API response
API request
Web Services

Your PayPal App must perform the following actions:

- Set parameters associated with the `Pay` API request message and pass them to your server, which executes the request.

  NOTE: Not shown in the diagram, you must call the PayPal-supplied `getReturnUrl` and `getCancelUrl` JavaScript functions to obtain the correct URLs to pass to the `Pay` API operation.

- Call the `gadgets.io.makeRequest` method to initiate the request on your server.
- Invoke the PayPal-supplied `launchFlow` JavaScript function to initiate an Adaptive Payments inline payment flow. Typically, your callback to the `gadgets.io.makeRequest` method calls `launchFlow`.

To set up your PayPal App to enable payments, follow these steps:

1. Specify the required library associated with the `Pay` API operation.

```
<ModulePrefs title="...">
  <Require feature="opensocial-0.8" />
  ...
  <Require feature="paypal.pay"/>
</ModulePrefs>
```

2. Obtain the Return and Cancel URLs.

The Return and Cancel URLs determine whether `success` or `cancel`, respectively, is returned to your PayPal App. See Step 6 for an example.

```
var payReturnUrl = paypal.apps.getReturnUrl();
var payCancelUrl = paypal.apps.getCancelUrl();
```

3. Set up all parameters required to create the `Pay` API request.

This example shows the parameters required in the request being set for the call to `gadgets.io.makeRequest`:

```
var params = {};
params[gadgets.io.RequestParameters.CONTENT_TYPE] =
   gadgets.io.ContentType.JSON;
params[gadgets.io.RequestParameters.METHOD] =
       gadgets.io.MethodType.GET;
var queryUrl = externalServer + "?receiverEmail=" +
               document.getElementById("receiverEmail").value;
queryUrl = queryUrl + "&amount=" +
           document.getElementById("amount").value;
queryUrl = queryUrl + "&returnUrl=" + payReturnUrl;
queryUrl = queryUrl + "&cancelUrl=" + payCancelUrl;
});
```

4. Call `gadgets.io.makeRequest`, specifying the URL of your service endpoint, a callback function that completes the request (see Step 5), and the parameters required to form the request.

```
gadgets.io.makeRequest(
    https://your_domain:port/your_server/your_servlet,
    ppCallback, params);
```

NOTE: You must specify the same service endpoint that you specified when you submitted your App to PayPal.The service endpoint includes the protocol, server name, and port; for example, `https://your_domain.com:port/your_server/...`.

5. Obtain the pay key, which will be used to launch the payment flow.

This example shows a callback function associated with `gadgets.io.makeRequest` that will be invoked when the gadget server responds. The callback obtains the payment key and uses it in the payment flow's URL.

```
function ppCallback(obj){
    var jsondata = obj.data;
    var payKey = jsondata["payKey"];
    paypal.apps.launchFlow(
        "https://paypal.com/webapps/adaptivepayment/flow/pay?paykey="
        + payKey + "&source=ppa",
        popupCallback);
}
```

6. Define a callback function that will be invoked at the end of the payment flow.

   The callback function receives either `success` or `cancel`, depending respectively on whether the Return URL or Cancel URL redirected the browser to your App.

```
function popupCallback(data){
    if (data == 'success') {
        html = "<font color=green><b>Payment Successfully Completed
                </b></font>";
    } else {
        html = "<font color=red><b>Payment Cancelled</b></font>";
    }
    document.getElementById('statusMessage').innerHTML = html;
}
```

7. Call the PayPal `launchFlow` JavaScript function, specifying the URL of the flow and the callback function to execute after the flow terminates.

```
paypal.apps.launchFlow(
"https://paypal.com/webapps/adaptivepayment/flow/pay?source=ppa&paykey="
+ payKey, popupCallback);
```

JavaScript Functions for PayPal Apps

PayPal provides JavaScript functions that you use within your PayPal App to send and receive parameters used in PayPal API request and response messages. All functions must be prefixed with the `paypal.apps` namespace.

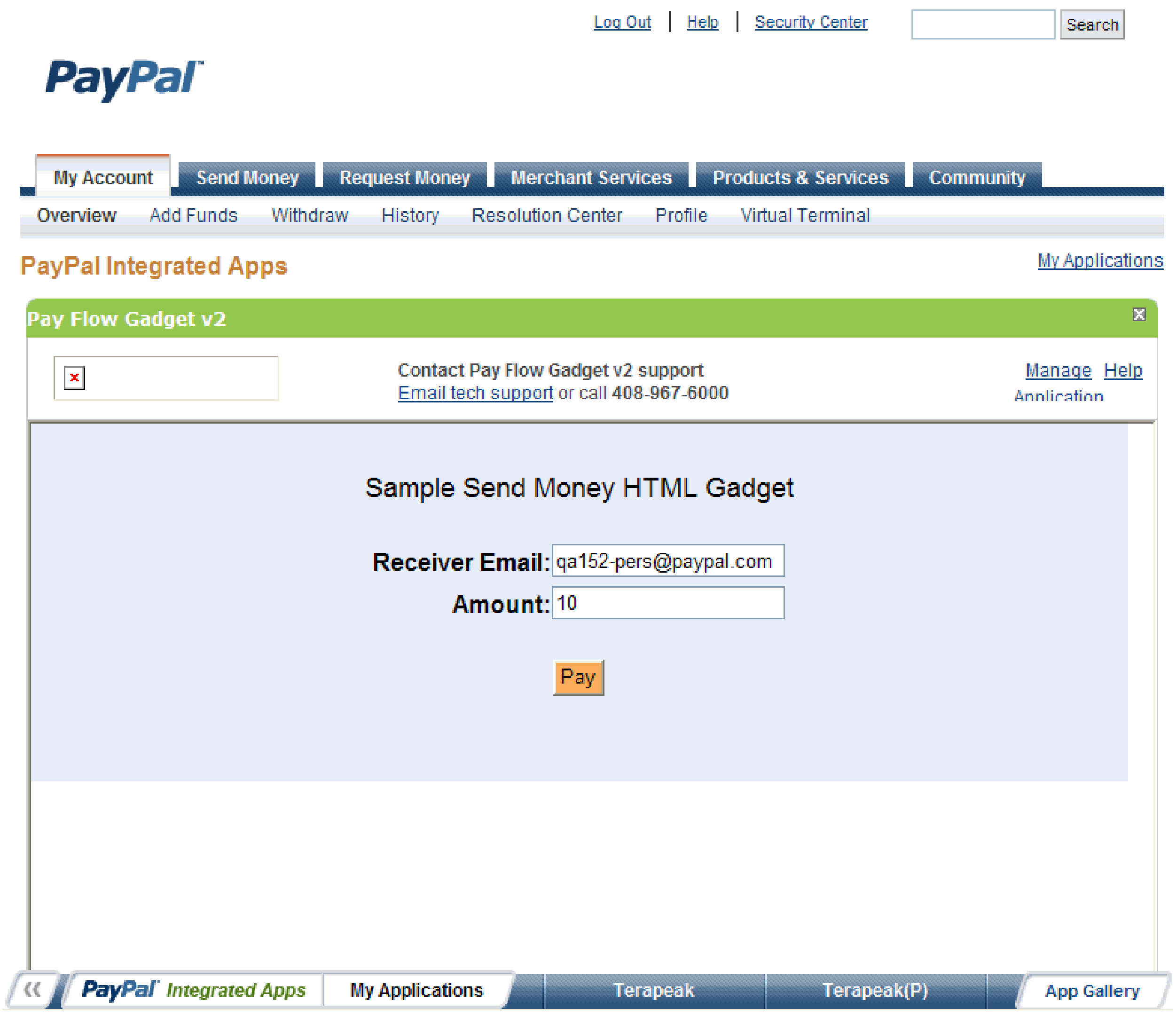
Function and Signature	Description
`launchFlow` `= function (`*`url`*`, `*`callback`*`)`	Opens a minibrowser window to initiate the flow at an endpoint specified by *`url`*. On successful completion or cancellation of the flow, it invokes the specified *`callback`* function.
`getReturnUrl` `: function ()`	Obtain the return URL.
`getCancelUrl` `: function ()`	Obtain the cancel URL.

Example Send Money PayPal App

This example PayPal App presents a form for sending money. When a PayPal account holder submits the form, a Pay API operation executes on your server and the response is sent back to your App. You call `launchFlow` to confirm the payment and transfer money.

Payment Flow

This PayPal App enables a PayPal account holder to specify the receiver’s email address and the amount of the payment. When the account holder clicks **Pay**, the request is sent to your server:

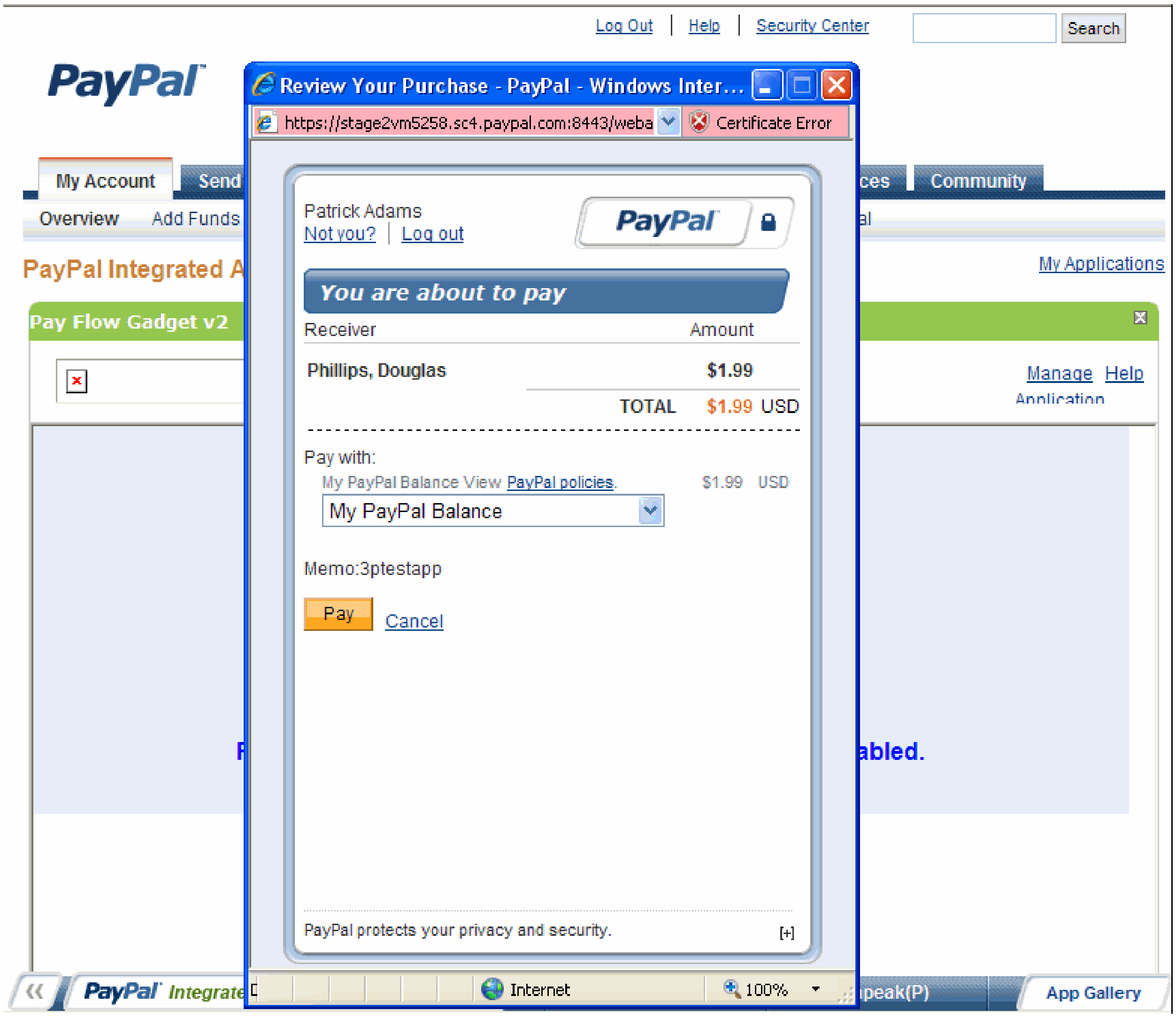


Contact Pay Flow Gadget v2 support
Email tech support or call 408-967-6000

Sample Send Money HTML Gadget

Receiver Email: qa152-pers@paypal.com
Amount: 10

When your server responds. the `launchFlow` JavaScript function initiates the payment flow and presents a review page in the PayPal minibrowser:



Patrick Adams
Not you? | Log out

You are about to pay

Receiver	Amount
Phillips, Douglas	$1.99
TOTAL	$1.99 USD

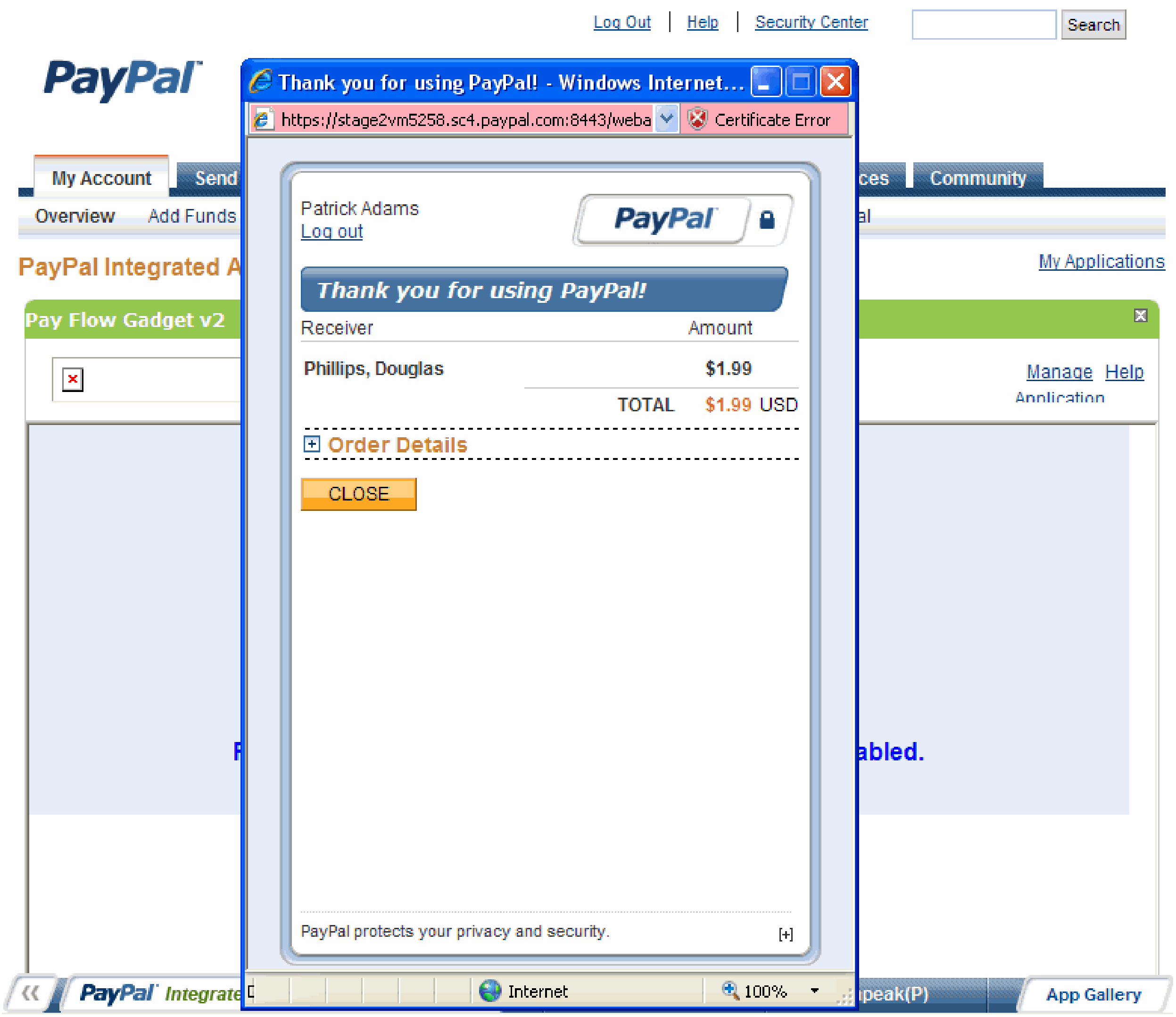
Pay with:
My PayPal Balance View PayPal policies. $1.99 USD
My PayPal Balance

Memo:3ptestapp

Pay Cancel

PayPal protects your privacy and security. [+]

When the account holder clicks the **Pay** button in the minibrowser, the flow continues with a summary page:



Patrick Adams
Log out

Thank you for using PayPal!

Receiver	Amount
Phillips, Douglas	$1.99
TOTAL	$1.99 USD

Order Details

CLOSE

PayPal protects your privacy and security.

When the account holder clicks **Close**, PayPal automatically closes the minibrowser, and redirects the browser to your PayPal App.

Sample Code

```
<?xml version="1.0" encoding="utf-8"?>
<Module>
 <ModulePrefs title="Simple Pay Gadget" author="..."
author_email="...@ebay.com">
   <Require feature="paypal.pay"/>
 </ModulePrefs>
 <Content type="html">
  <![CDATA[
<style type="text/css">
 body {
  font-family: arial, sans-serif;
 }
 #headerDiv {
  padding: 10px;
  margin-bottom: 20px;
  background-color: #e5ecf9;
  color: #3366cc;
  font-size: larger;
  font-weight: bold;
  text-align: center;
 }
 .subTitle {
  font-size: smaller;
  text-align: center;
 }
 .mainTitle {
  font-size: larger;
  text-align: center;
 }
 .gadgets-gadget-chrome {
  width: 60%;
  margin: auto;
 }
 .buttonDisplay {
  background-color: #faac58;
 }
 .gadgets-gadget {
  width: 100%;
 }
</style>

   <center> <h2>Sample Pay Gadget</h2> </center>

<!-- Capture Payment Details -->
 <div id="headerDiv">
 <table align="center">
 <tr align="center"><td><b> Receiver Email:</b><input type="text"
 id="receiverEmail" size=30 value="...@yahoo.com"></td></tr
 <tr align="center"><td><b> Amount:</b><input type="text" id="amount"
 value="10"></td></tr>
 <tr align="center"><td><input type="button" class="buttonDisplay"
 value="Pay" onclick="makeJSONRequest();"></td></tr>
 <tr align="center"><td> <div id="statusMessage"> </div></td></tr>
 </table>
 </div>

<!-- Include URLs for your extenal server and Adaptive Payment confirmation
-->
<script type="text/javascript">
  var externalServer = "https://host@domain:port/payserver/PayServlet";
  var adaptivePaymentConfirmation =
      "https://paypal.com/webapps/adaptivepayment/flow/pay?paykey=";
  var payReturnUrl = paypal.apps.getReturnUrl();
  var payCancelUrl = paypal.apps.getCancelUrl();
</script>

<!-- Send payment details to your external server using
gadgets.io.makeRequest -->
<script type="text/javascript">
  function makeJSONRequest() {
     var params = {};
     var queryUrl = externalServer + "?receiverEmail=" +
                    document.getElementById("receiverEmail").value;
     queryUrl = queryUrl + "&amount=" +
                document.getElementById("amount").value;
     queryUrl = queryUrl + "&returnUrl=" + payReturnUrl;
     queryUrl = queryUrl + "&cancelUrl=" + payCancelUrl;
     var html = "<font color=blue><b>Payment Being Processed -
                 Make sure  pop-ups are enabled.</b></font>";
     document.getElementById('statusMessage').innerHTML = html;
     params[gadgets.io.RequestParameters.CONTENT_TYPE] =
     gadgets.io.ContentType.JSON;
     params[gadgets.io.RequestParameters.METHOD] =
     gadgets.io.MethodType.GET;
     gadgets.io.makeRequest(queryUrl, makeRequestCallback, params);
   }
</script>

<!-- Create a callback function to read the paykey and launch the mini-
browser -->
<script type="text/javascript">
  function makeRequestCallback(obj){
    var jsondata = obj.data;
    var payKey = jsondata["payKey"];
    paypal.apps.launchFlow(adaptivePaymentConfirmation + payKey,
    popupCallback);
  }
</script>
```

```
<!-- Create a callback function to update the status -->
<script type="text/javascript">
  function popupCallback(data){
    var html = "";
    if (data == 'success') {
        html =
        "<font color=green><b>Payment Successfully Completed </b></font>";
    } else {
        html = "<font color=red><b>Payment Cancelled</b></font>";
    }
        document.getElementById('statusMessage').innerHTML = html;
  }

</script>
   ]]>
 </Content>
</Module>
```

7 Open Source Resources for PayPal Apps

For more information about the Google Gadgets API, see http://code.google.com/apis/gadgets/.

For more information about the Open Social framework, see http://www.opensocial.org/.

For more information about Caja, see google-caja

For information about Shingdig, see Shingdig Getting Started

For information about the YUI Library, see http://developer.yahoo.com/yui/.

For more information on the Open Authorization (OAuth) protocol as it used with Open Social applications, see the following links:

- Introduction To Signed Requests
- Validating Signed Requests
- OAuth Enabled Gadgets on OpenSocial enabled Social Networks