64.3

# PayPal Express Checkout Integration Guide

Last updated: December 2010

*PayPal Express Checkout Integration Guide*

Document Number: 100010.en_US-201012

# Contents

# 1 Preface

## About This Guide

This document describes basic Express Checkout integration.

## Intended Audience

This document is for merchants and developers who want to get started implementing Express Checkout.

## Revision History

Revision history for *PayPal Express Checkout Integration Guide*.

| Date Published | Description |
|---|---|
| 12/20/10 | Replaced deprecated field names in examples. |
| 11/15/10 | Revised for version 65.3, with updates to the locales supported by Express Checkout on mobile devices. |
| 10/26/10 | Added the chapter “Express Checkout on Mobile Devices.” |
| 08/11/10 | Divided the *Express Checkout Integration Guide* into 2 guides: the *Express Checkout Integration Guide* and the *Express Checkout Advanced Features Guide*. |
| 05/11/10 | Added details for integrating parallel payments using the NVP and SOAP API, including use with airlines. Added new Immediate Payment functionality. Updated billing agreements with functionality to obtain the latest billing address, skip billing agreement creation, and clarify use of the BAUpdate API. |
| 03/10/10 | Added support for parallel payments. |
| 01/21/2010 | Added new Express Checkout fields to provide the buyer contact information, gift options, promotions, and a survey question on the PayPal pages. Added a new callback response API field providing no-shipping details. |

| Date Published | Description |
|---|---|
| 10/05/2009 | Added Immediate Payment.<br>Edited for technical accuracy.<br>Removed PayPal placement guidelines. |
| 06/30/2009 | Added a section on payment review. |
| 06/04/2009 | Added a chapter on pre-populating the PayPal review page. Updated PayPal Review pages. Moved some customization topics out of this guide. They are now in the *Merchant Setup and Administration Guide*. |
| 04/30/2009 | Created first edition for Enterprise-level merchants and added chapter on reference transactions. |
| 04/08/2009 | Added a chapter describing the Instant Update Callback API. |
| 03/03/2009 | Updated to allow useraction=continue for eBay incentives. |
| 11/13/2008 | Added information about integrating dynamic images and added information about order details that can be displayed on the PayPal Review page. |
| 06/30/2008 | Complete revision. |

## Where to Go for More Information

- Express Checkout Advanced Features Guide
- Merchant Setup and Administration Guide

## Documentation Feedback

Help us improve this guide by sending feedback to:

`documentationfeedback@paypal.com`

# 1 Introducing Express Checkout

Express Checkout is PayPal's premier checkout solution, which streamlines the checkout process for buyers and keeps them on the merchant's site after making a purchase.

- The Express Checkout Experience
- Express Checkout Integration Steps
- Express Checkout Flow
- Express Checkout Building Blocks

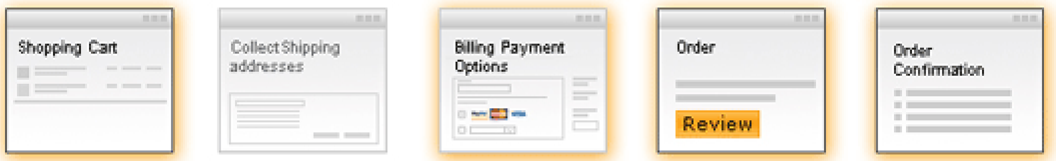
**NOTE:** For information about administrative tasks you can perform from your PayPal account such as adding users, setting up custom page styles, and managing multiple currency balances, see the Merchant Setup and Administration Guide.

## The Express Checkout Experience

Express Checkout makes it easier for a buyers to pay online. It also enables you to accept PayPal while retaining control of the buyer and the overall checkout flow.

Consider your buyers' experience before implementing Express Checkout. A generic flow probably has the following sequence of pages:

***A generic checkout flow***



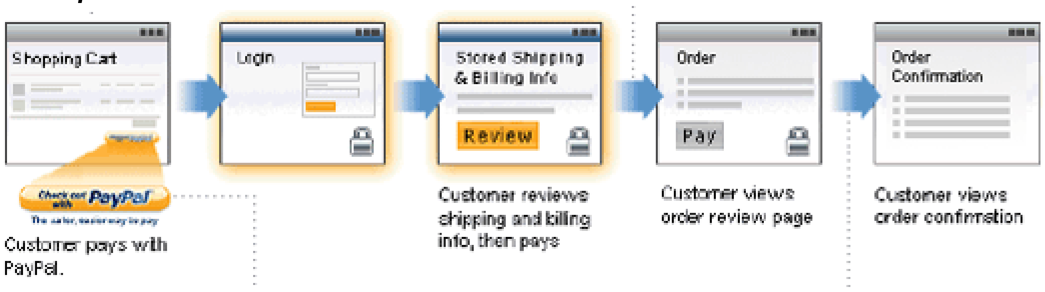
In a typical checkout flow, a buyer:

1. Checks out from the shopping cart page
2. Provides shipping information
3. Chooses a payment option and provides billing and payment information
4. Reviews the order and pays
5. Receives an order confirmation

In an Express Checkout flow, a buyer still checks out at the beginning of the flow. However, the buyer does not enter shipping, billing, or payment information, because PayPal provides the stored information. This simplifies and expedites the checkout process.

The following diagram shows the Express Checkout flow:

***Express Checkout flow***


Shopping Cart
Check out with PayPal
Customer pays with PayPal.
Login
Stored Shipping & Billing Info
Review
Customer reviews shipping and billing info, then pays
Order
Pay
Customer views order review page
Order Confirmation
Customer views order confirmation

In the Express Checkout flow, the buyer:

1. Chooses Express Checkout by clicking **Check out with PayPal**
2. Logs into PayPal to authenticate his or her identity
3. Reviews the transaction on PayPal

   **NOTE:** Optionally, (not shown in the diagram), the buyer can then proceed to review the order on your site. You can also include other checkout steps, including upselling on your Review Order page.

4. Confirms the order and pays from your site
5. Receives an order confirmation

## Express Checkout Integration Steps

You can implement Express Checkout in 4 steps:

1. Place PayPal checkout buttons and PayPal payment mark images in your checkout flow.
2. For each PayPal button that you place, modify your page to handle the button click.

   Use a PayPal Express Checkout API operation to set up the interaction with PayPal and redirect the browser to PayPal to initiate buyer approval for the payment.

3. On your Order page, use PayPal Express Checkout API operations to obtain the shipping address and accept the payment.
4. Test your integration using the PayPal Sandbox before taking your pages live.

Because PayPal offers you the flexibility to control your checkout flow, you should first understand how your current checkout flow works, then, become familiar with the Express Checkout flow. Start by reviewing Express Checkout Flow. For additional background information to help you get started, see Express Checkout Building Blocks.

## Configuring and Customizing the Express Checkout Experience

After you implement and test your basic Express Checkout integration, you should configure the additional features of Express Checkout to customize it to meet your needs.

Carefully evaluate each feature because the more you streamline the checkout process and make Express Checkout seamless to buyers, the more likely your sales will increase.

At a minimum, you should:

- Set your logo on the PayPal site and provide order details in the transaction history.
- Use the PayPal confirmation page as your Order Review page to further streamline the user experience when you do not need the benefits associated with paying on your site. This strategy can lead to a better order completion rate, also known as a *conversion rate*.

Configure the look and feel of PayPal pages to match the look and feel of your site by specifying the:

- Logo to display
- Colors for the background and border
- Language in which PayPal content is displayed

You should include:

- Order details, including shipping and tax, during checkout

  **IMPORTANT:** Not displaying this information is a major cause of shopping cart abandonment during checkout.

- Shipping information for non-digital goods, which can be your address information for the buyer or the address on file with PayPal; if you use the address on file with PayPal, you can specify whether or not it must be a confirmed address

You can also activate additional features, including:

- Associate a payment with an eBay auction item
- Assign an invoice number to a payment
- Accept payments with giropay (Germany only)

## Additional PayPal API Operations

You can use PayPal API operations to include advanced processing and back-office processes with Express Checkout. You can:

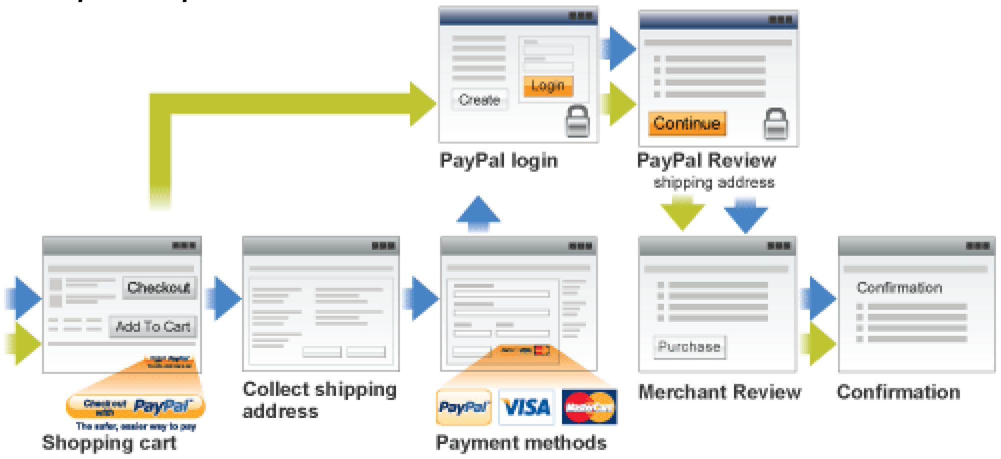
- Capture payments associated with authorizations and orders
- Process recurring payments
- Issue refunds, search transactions using various criteria, and provide other back-office operations

## Express Checkout Flow

To implement Express Checkout, you must offer it both as a checkout option and as a payment method. Typically, you initiate the Express Checkout flow on your shopping cart page and on your payment options page.

You add Express Checkout to your existing flow by placing the **Checkout with PayPal** button on your **Shopping Cart** page and by placing the **PayPal** mark on your **Payment Methods** page. The following diagram shows the complete flow:

***Complete Express Checkout flow***



Make the following changes to implement the complete Express Checkout flow:

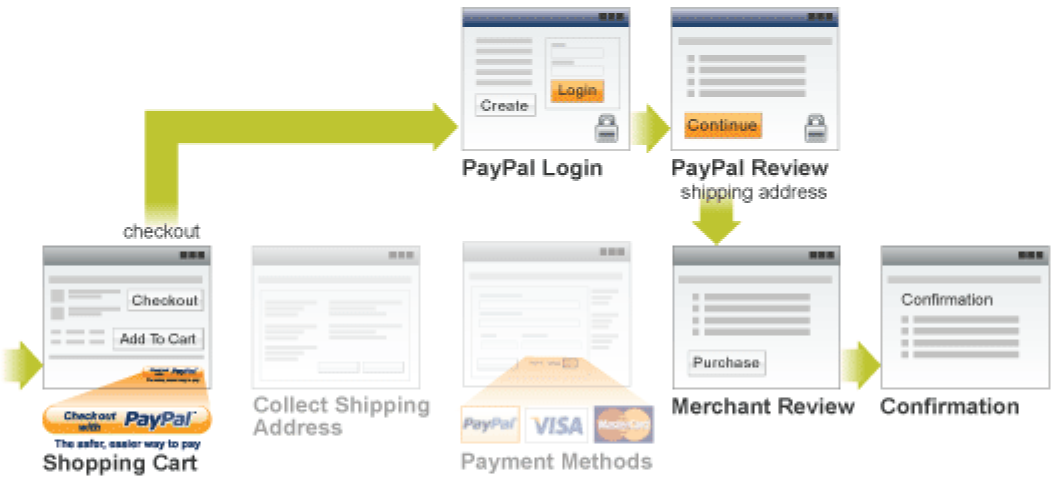
- On your **Shopping Cart** page, place the **Checkout with PayPal** button and respond to a click by setting up the Express Checkout request and redirecting your buyer's browser to PayPal.
- On your **Payment Methods** page, associate the **PayPal** mark with an option. Handle selection of the **PayPal** mark by setting up the Express Checkout request and redirecting your buyer's browser to PayPal.
- On the page your buyer returns to, obtain shipping information from PayPal and accept the payment to complete the Express Checkout transaction.

**NOTE:** You also can allow the buyer to pay on the PayPal Review page. In this case, your checkout flow can omit the Merchant Review page and proceed directly to your Confirmation page.

### Checkout Entry Point

The checkout entry point is one of the places where you must implement Express Checkout. Buyers initiate the Express Checkout flow on your shopping cart page by clicking the **Checkout with PayPal** button.

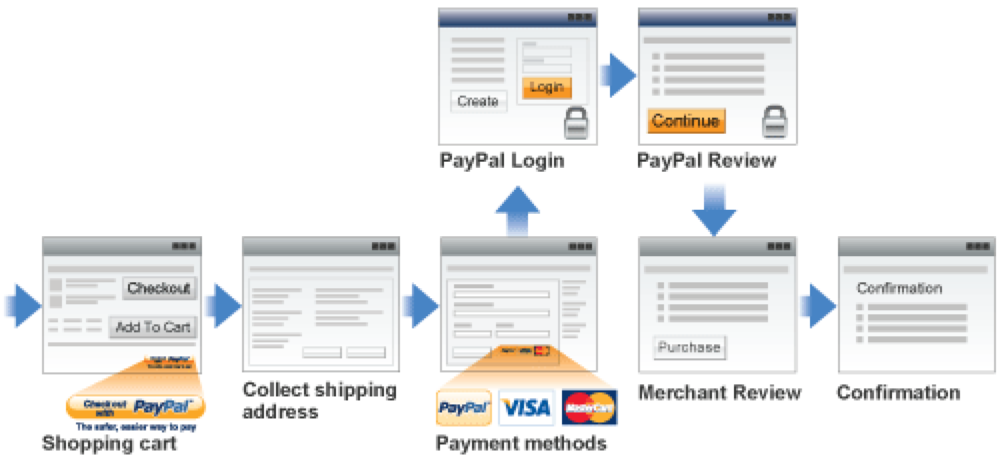
The following diagram shows how Express Checkout integrates with a typical checkout flow:



## Payment Option Entry Point

The payment option entry point is one of the places where you must implement Express Checkout. Buyers initiate the Express Checkout flow on your payment methods page by selecting PayPal as the default option.

The following diagram shows how to integrate Express Checkout from your payment methods page:



## PayPal Button and Logo Images

To inform buyers that PayPal is accepted on your website, you must place PayPal button and logo images in your checkout flow. PayPal recommends that you use dynamic images.

PayPal requires that you use the **Check out with PayPal** and the **PayPal** mark images hosted on secure PayPal servers. When the images are updated, the changes appear automatically in your application. Do not host copies of the PayPal images locally on your servers. Outdated PayPal images reduces buyer confidence in your site.

## Express Checkout Image Flavors

The **Check out with PayPal** button and the **PayPal** mark images are available in two flavors:

- Dynamic image
- Static image

The dynamic images enable PayPal to change their appearance dynamically. If, for example, you have signed up to participate in a PayPal campaign, PayPal can change the appearance of the image dynamically for the duration of that campaign based on parameter information you append to the image URL. By default, the Express Checkout images appears as shown above.

The static images cannot be changed dynamically. To participate in a PayPal campaign, you would have to manually update the image code to change the image displayed and restore the default image when the campaign is over. The only way you can have image management taken care of for you is to replace static images in your implementation with dynamic images.

## Express Checkout Images

The **Check out with PayPal** button is the image you place on your shopping cart page. The US version of the image looks like this.

To create an Express Checkout button, see https://www.paypal.com/us/cgi-bin/webscr?cmd=xpt/Merchant/merchant/ExpressCheckoutButtonCode-outside. PayPal also provides buttons for other countries. To locate a page for another country, replace the country abbreviation in the link with another country abbreviation. For example, replace `us` with `uk` for United Kingdom, as follows: https://www.paypal.com/uk/cgi-bin/webscr?cmd=xpt/Merchant/merchant/ExpressCheckoutButtonCode-outside. PayPal hosts images for the countries:

### Country-specific buttons and images

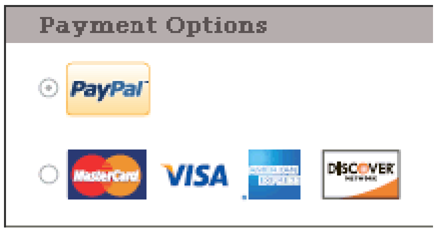
| Country | URL Change | Country | URL Change | Country | URL Change | Country | URL Change |
|---|---|---|---|---|---|---|---|
| Australia | au | Austria | at | Belgium | be | Canada | ca |
| China | cn | France | fr | Germany | de | Italy | it |
| Japan | j1 | Netherlands | nl | Poland | pl | Spain | es |
| Switzerland | ch | United Kingdom | uk | United States | us | | |

**NOTE:** URL changes are case sensitive. The abbreviation in the URL may not be a country code.

## Payment Mark

The **PayPal** mark is the image you place on your payment methods page. It looks like this:

To implement PayPal as a payment option, which is part of the Express Checkout experience, associate the **PayPal** mark image with your payment options. PayPal recommends using radio buttons for payment options:



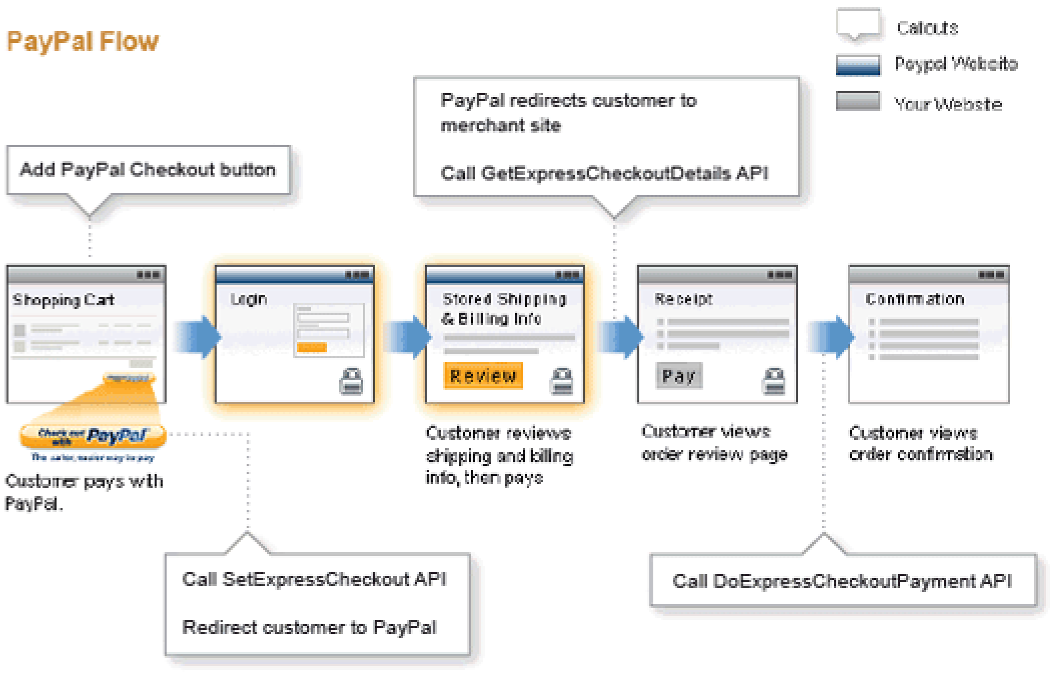
To create a **PayPal** mark, see https://www.paypal.com/cgi-bin/webscr?cmd=xpt/Marketing/general/OnlineLogoCenter-outside.

## Express Checkout Building Blocks

You implement Express Checkout flows with Express Checkout buttons, PayPal API operations, PayPal commands, and tokens.

The following conceptual diagram identifies the building blocks that you use to integrate Express Checkout on your website:

***Express Checkout Integration***


PayPal Flow
Callouts
Paypal Website
Your Website
Add PayPal Checkout button
PayPal redirects customer to merchant site
Call GetExpressCheckoutDetails API
Shopping Cart
Login
Stored Shipping & Billing Info
Review
Receipt
Pay
Confirmation
Customer pays with PayPal.
Customer reviews shipping and billing info, then pays
Customer views order review page
Customer views order confirmation
Call SetExpressCheckout API
Redirect customer to PayPal
Call DoExpressCheckoutPayment API

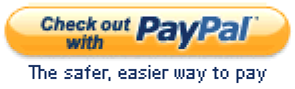
A *token* is a value assigned by PayPal that associates the execution of API operations and commands with a specific instance of a user experience flow.

**NOTE:** Tokens are not shown in the diagram.

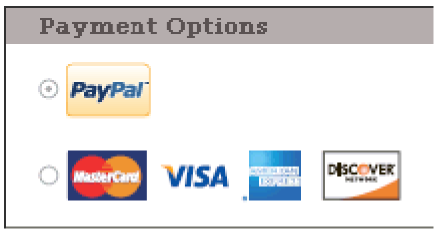
## Express Checkout Buttons

PayPal provides buttons and images for you to place on your website.

To implement the Express Checkout shopping cart experience, place the following button on your Shopping Cart page:



To implement PayPal as a payment option, which is part of the Express Checkout experience, associate the **PayPal** mark image with your payment options. PayPal recommends using radio buttons for payment options:



## Express Checkout API Operations

The PayPal API provides three API operations for Express Checkout, which sets up the transaction, obtains information about the buyer, and handles the payment and completes the transaction.

| API Operation | Description |
|---|---|
| `SetExpressCheckout` | Sets up the Express Checkout transaction. You can specify information to customize the look and feel of the PayPal site and the information it displays. You must include the following information:<br>● URL to the page on your website that PayPal redirects to after the buyer logs into PayPal and approves the payment successfully.<br>● URL to the page on your website that PayPal redirects to if the buyer cancels.<br>● Total amount of the order or your best estimate of the total. It should be as accurate as possible. |
| `GetExpressCheckout` | Obtains information about the buyer from PayPal, including shipping information. |
| `DoExpressCheckoutPayment` | Completes the Express Checkout transaction, including the actual total amount of the order. |

## Express Checkout Command

PayPal provides a command that you use when redirecting your buyer's browser to PayPal. This command enables your buyer to log into PayPal to approve an Express Checkout payment.

When you redirect your buyer's browser to PayPal, you must specify the `_ExpressCheckout` command for Express Checkout. You also specify the token that identifies the transaction, which was returned by the `SetExpressCheckout` API operation.

**NOTE:** To enable PayPal to redirect back to your website, you must have already invoked the `SetExpressCheckout` API operation, specifying URLs that PayPal uses to redirect back to your site. PayPal redirects to the *success* URL when the buyer pays on PayPal; otherwise, PayPal redirects to the *cancel* URL.

If the buyer approves the payment, PayPal redirects to the success URL with the following information:

- The token that was included in the redirect to PayPal
- The buyer's unique identifier (Payer ID)

If the buyer cancels, PayPal redirects to the cancel URL with the token that was included in the redirect to PayPal.

## Express Checkout Token Usage

Express Checkout uses a token to control access to PayPal and execute Express Checkout API operations.

The `SetExpressCheckout` API operation returns a token, which is used by other Express Checkout API operations and by the `_ExpressCheckout` command to identify the transaction. The life of the token is approximately 3 hours.

# 2 Getting Started With Express Checkout

To implement Express Checkout, start with the simplest Express Checkout integration and test it. Then you can decide the kind of payment settlement actions you want to support.

- Implementing the Simplest Express Checkout Integration
- Testing an Express Checkout Integration
- Handling Payment Settlements With Express Checkout
- Issuing Refunds

## Implementing the Simplest Express Checkout Integration

The simplest Express Checkout integration requires the following PayPal API operations: `SetExpressCheckout`, `DoExpressCheckoutPayment`, and optionally, `GetExpressCheckoutDetails`.

- Setting Up the Express Checkout Transaction
- Obtaining Express Checkout Transaction Details
- Completing the Express Checkout Transaction

### Setting Up the Express Checkout Transaction

To set up an Express Checkout transaction, you must invoke the `SetExpressCheckout` API operation to provide sufficient information to initiate the payment flow and redirect to PayPal if the operation was successful.

This example assumes that you have set up the mechanism you will use to communicate with the PayPal server and have a PayPal business account with API credentials. It also assumes that the payment action is a final sale.

When you set up an Express Checkout transaction, you specify values in the `SetExpressCheckout` request and then call the API. The values you specify control the PayPal page flow and the options available to you and your buyers. You should start by setting up a standard Express Checkout transaction, which can be modified to include additional options.

To set up the simplest standard Express Checkout transaction

**1.** Specify the amount of the transaction; include the currency if it is not in US dollars.

Specify the total amount of the transaction if it is known; otherwise, specify the subtotal. Regardless of the specified currency, the format must have decimal point with exactly two digits to the right and an optional thousands separator to the left, which must be a comma.

For example, EUR 2.000,00 must be specified as 2000.00 or 2,000.00. The specified amount cannot exceed USD $10,000.00, regardless of the currency used.

```
PAYMENTREQUEST_0_AMT=amount
PAYMENTREQUEST_0_CURRENCYCODE=currencyID
```

**2.** Specify the return URL.

The return URL is the page to which PayPal redirects your buyer's browser after the buyer logs into PayPal and approves the payment. Typically, this is a secure page (`https://...`) on your site.

**NOTE:** You can use the return URL to piggyback parameters between pages on your site. For example, you can set your Return URL to specify additional parameters using the `https://www.`*yourcompany*`.com/`*page*`.html?`*param*`=`*value*... syntax. The parameters become available as request parameters on the page specified by the Return URL.

```
RETURNURL=return_url
```

**3.** Specify the cancel URL.

The cancel URL is the page to which PayPal redirects your buyer's browser if the buyer does not approve the payment. Typically, this is the secure page (`https://...`) on your site from which you redirected the buyer to PayPal.

**NOTE:** You can pass `SetExpressCheckout` request values as parameters in your URL to have the values available, if necessary, after PayPal redirects to your URL.

```
CANCELURL=cancel_url
```

**4.** Specify the payment action.

Although the default payment action is a `Sale`, it is a best practice to explicitly specify the payment action as one of the following values:

```
PAYMENTREQUEST_0_PAYMENTACTION=Sale
PAYMENTREQUEST_0_PAYMENTACTION=Authorization
PAYMENTREQUEST_0_PAYMENTACTION=Order
```

**5.** Execute the `SetExpressCheckout` API operation to set up the Express Checkout transaction.

6. Test that the response to the `SetExpressCheckout` API operation was successful.
7. If calling the `SetExpressCheckout` API was successful, redirect the buyer's browser to PayPal and execute the `_express-checkout` command using the token returned in the `SetExpressCheckout` response.

**NOTE:** The following example uses the PayPal Sandbox server:

```
https://www.sandbox.paypal.com/webscr
       ?cmd=_express-checkout&token=tokenValue
```

## Obtaining Express Checkout Transaction Details

To obtain details about an Express Checkout transaction, you can invoke the `GetExpressCheckoutDetails` API operation.

This example assumes that PayPal redirects to your buyer's browser with a valid token after the buyer reviews the transaction on PayPal.

Although you are not required to invoke the `GetExpressCheckoutDetails` API operation, most Express Checkout implementations take this action to obtain information about the buyer. You invoke the `GetExpressCheckoutDetails` API operation from the page specified by return URL, which you set in your call to the `SetExpressCheckout` API. Typically, you invoke this operation as soon as the redirect occurs and use the information in the response to populate your review page.

To obtain a buyer's shipping address and Payer ID

1. Specify the token returned by PayPal when it redirects the buyer's browser to your site.

   PayPal returns the token to use in the `token` HTTP request parameter when redirecting to the URL you specified in your call to the `SetExpressCheckout` API.

```
TOKEN=tokenValue
```

2. Execute the `GetExpressCheckoutDetails` API to obtain information about the buyer.
3. Access the fields in the `GetExpressCheckoutDetails` API response.

   **NOTE:** Only populated fields are returned in the response.

## Completing the Express Checkout Transaction

To complete an Express Checkout transaction, you must invoke the `DoExpressCheckoutPayment` API operation.

This example assumes that PayPal redirects your buyer's browser to your website with a valid token after you call the `SetExpressCheckout` API. Optionally, you may call the

`GetExpressCheckoutDetails` API before calling the `DoExpressCheckoutPayment` API.

In the simplest case, you set the total amount of the order when you call the `SetExpressCheckout` API. However, you can change the amount before calling the `DoExpressCheckoutPayment` API if you did not know the total amount when you called the `SetExpressCheckout` API.

This example assumes the simplest case, in which the total amount was specified in the return URL when calling the `SetExpressCheckout` API. Although you can specify additional options, this example does not use any additional options.

To execute an Express Checkout transaction

**1.** Specify the token returned by PayPal when it redirects the buyer's browser to your site.

PayPal returns the token to use in the `token` HTTP request parameter when redirecting to the URL you specified in your call to the `SetExpressCheckout` API.

```
TOKEN=tokenValue
```

**2.** Specify the Payer ID returned by PayPal when it redirects the buyer's browser to your site.

PayPal returns the Payer ID to use in the `token` HTTP request parameter when redirecting to the URL you specified in your call to the `SetExpressCheckout` API. Optionally, you can obtain the Payer ID by calling the `GetExpressCheckoutDetails` API.

```
PAYERID=id
```

**3.** Specify the amount of the order including shipping, handling, and tax; include the currency if it is not in US dollars.

Regardless of the specified currency, the format must have decimal point with exactly two digits to the right and an optional thousands separator to the left, which must be a comma; for example, EUR 2.000,00 must be specified as 2000.00 or 2,000.00. The specified amount cannot exceed USD $10,000.00, regardless of the currency used.

```
PAYMENTREQUEST_0_AMT=amount
PAYMENTREQUEST_0_CURRENCYCODE=currencyID
```

**4.** Specify the payment action.

Although the default payment action is a `Sale`, it is a best practice to explicitly specify the payment action as one of the following values:

```
PAYMENTREQUEST_0_PAYMENTACTION=Sale
PAYMENTREQUEST_0_PAYMENTACTION=Authorization
PAYMENTREQUEST_0_PAYMENTACTION=Order
```

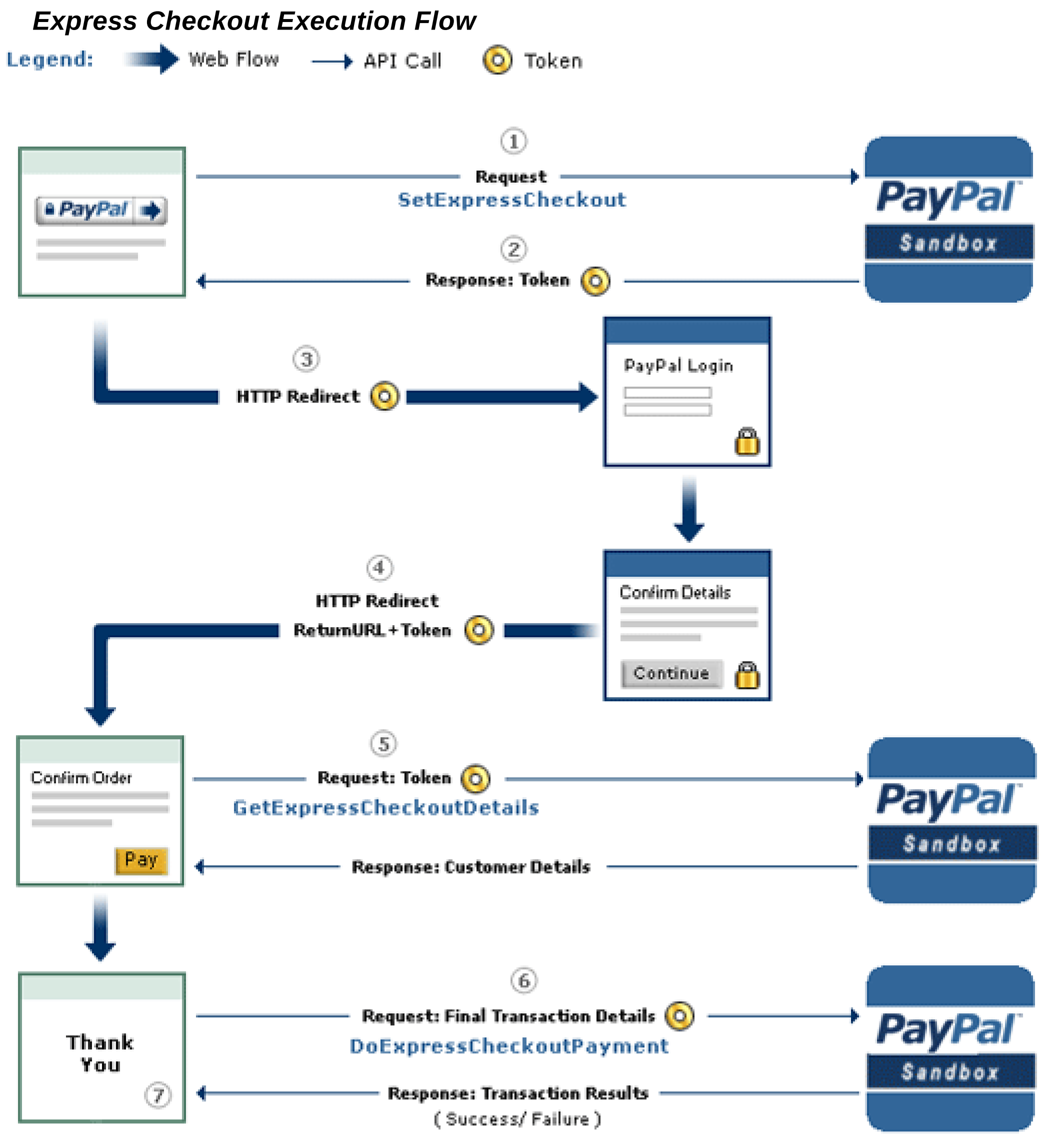
5. Execute the `DoExpressCheckoutPayment` API to complete the Express Checkout transaction.
6. Examine the values returned by the API if the transaction completed successfully.

## Testing an Express Checkout Integration

You can test your Express Checkout integration in the Sandbox.

This example shows how to simulate your web pages using HTTP forms and supplying the values for API operations from these forms. You can use this strategy for your initial testing; however, for more complete testing, you will want to replace these forms with your web pages containing actual code.

The following diagram shows the Express Checkout execution flow, which uses the Sandbox as the API server. The pages on the left represent your site.


Express Checkout Execution Flow
Legend: Web Flow API Call Token
1 Request SetExpressCheckout
2 Response: Token
PayPal Sandbox
3 HTTP Redirect
PayPal Login
Confirm Details
Continue
4 HTTP Redirect ReturnURL + Token
Confirm Order
Pay
5 Request: Token GetExpressCheckoutDetails
Response: Customer Details
6 Request: Final Transaction Details DoExpressCheckoutPayment
Response: Transaction Results (Success/ Failure)
Thank You
7

The following steps match the circled numbers in the diagram. Perform the actions in each step to test Express Checkout.

**1.** Invoke a form on your site that calls the `SetExpressCheckout` API on the Sandbox.

To invoke the API, set form fields whose names match the NVP names of the fields you want to set, specify their corresponding values, and then post the form to a PayPal Sandbox server, such as `https://api-3t.sandbox.paypal.com/nvp`, as shown in the following example:

```
<form method=post action=https://api-3t.sandbox.paypal.com/nvp>
   <input type=hidden name=USER value=API_username>
   <input type=hidden name=PWD value=API_password>
   <input type=hidden name=SIGNATURE value=API_signature>
   <input type=hidden name=VERSION value=XX.0>
   <input type=hidden name=PAYMENTREQUEST_0_PAYMENTACTION
      value=Authorization>
   <input name=PAYMENTREQUEST_0_AMT value=19.95>
   <input type=hidden name=RETURNURL
      value=http://www.YourReturnURL.com>
   <input type=hidden name=CANCELURL
      value=http://www.YourCancelURL.com>
   <input type=submit name=METHOD value=SetExpressCheckout>
</form>
```

**NOTE:** The API username is a Sandbox business test account for which a signature exists. See the Test Certificates tab of the Sandbox to obtain a signature. If you are not using a signature, you must use a different Sandbox server.

**2.** Review the response string from the `SetExpressCheckout` API operation.

PayPal responds with a message, such as the one shown below. Note the status, which should include `ACK` set to `Success`, and a token that is used in subsequent steps.

```
TIMESTAMP=2007%2d04%2d05T23%3a23%3a07Z
&CORRELATIONID=63cdac0b67b50
&ACK=Success
&VERSION=XX%2e000000
&BUILD=1%2e0006
&TOKEN=EC%2d1NK66318YB717835M
```

**3.** If the operation was successful, use the token and redirect your browser to the Sandbox to log in, as follows:

```
https://www.sandbox.paypal.com/cgi-bin/webscr?
cmd=_express-checkout
&token=EC-1NK66318YB717835M
```

You may need to replace hexadecimal codes with ASCII codes; for example, you may need to replace `%2d` in the token with a hyphen ( `-` ).

You must log in to `https://developer.paypal.com` before you log in to a Sandbox test account. You then log in to the test account that represents the buyer, not the API_username business test account that represents you as the merchant.

**4.** After logging into the buyer test account, confirm the details.

When you confirm, the Sandbox redirects your browser to the return URL you specified when invoking the `SetExpressCheckout` API operation, as in the following example:

```
http://www.YourReturnURL.com/
                    ?token=EC-1NK66318YB717835M&PayerID=7AKUSARZ7SAT8
```

**5.** Invoke a form on your site that calls the `GetExpressCheckoutDetails` API operation on the Sandbox:

```
<form method=post action=https://api-3t.sandbox.paypal.com/nvp
   <input type=hidden name=USER value=API_username>
   <input type=hidden name=PWD value=API_password>
   <input type=hidden name=SIGNATURE value=API_signature>
   <input type=hidden name=VERSION value=XX.0>
   <input name=TOKEN value=EC-1NK66318YB717835M>
   <input type=submit name=METHOD value=GetExpressCheckoutDetails>
</form>
```

If the operation was successful, the `GetExpressCheckoutDetails` API returns information about the payer, such as the following information:

```
TIMESTAMP=2007%2d04%2d05T23%3a44%3a11Z
&CORRELATIONID=6b174e9bac3b3
&ACK=Success
&VERSION=XX%2e000000
&BUILD=1%2e0006
&TOKEN=EC%2d1NK66318YB717835M
&EMAIL=YourSandboxBuyerAccountEmail
&PAYERID=7AKUSARZ7SAT8
&PAYERSTATUS=verified
&FIRSTNAME=...
&LASTNAME=...
&COUNTRYCODE=US
&BUSINESS=...
&PAYMENTREQUEST_0_SHIPTONAME=...
&PAYMENTREQUEST_0_SHIPTOSTREET=...
&PAYMENTREQUEST_0_SHIPTOCITY=...
&PAYMENTREQUEST_0_SHIPTOSTATE=CA
&PAYMENTREQUEST_0_SHIPTOCOUNTRYCODE=US
&PAYMENTREQUEST_0_SHIPTOCOUNTRYNAME=United%20States
&PAYMENTREQUEST_0_SHIPTOZIP=94666
&PAYMENTREQUEST_0_ADDRESSID=...
&PAYMENTREQUEST_0_ADDRESSSTATUS=Confirmed
```

**6.** Invoke a form on your site that invokes the `DoExpressCheckoutPayment` API operation on the Sandbox:

```
<form method=post action=https://api-3t.sandbox.paypal.com/nvp>
   <input type=hidden name=USER value=API_username>
   <input type=hidden name=PWD value=API_password>
   <input type=hidden name=SIGNATURE value=API_signature>
   <input type=hidden name=VERSION value=XX.0>
   <input type=hidden name=PAYMENTREQUEST_0_PAYMENTACTION
      value=Authorization>
   <input type=hidden name=PAYERID value=7AKUSARZ7SAT8>
   <input type=hidden name=TOKEN value= EC%2d1NK66318YB717835M>
   <input type=hidden name=PAYMENTREQUEST_0_AMT value= 19.95>
   <input type=submit name=METHOD value=DoExpressCheckoutPayment>
</form>
```

7. Review the response string from the `DoExpressCheckoutPayment` API operation.

   If the operation was successful, the response should include `ACK` set to `Success`, as follows:

```
TIMESTAMP=2007%2d04%2d05T23%3a30%3a16Z
&CORRELATIONID=333fb808bb23
ACK=Success
&VERSION=XX%2e000000
&BUILD=1%2e0006
&TOKEN=EC%2d1NK66318YB717835M
&PAYMENTREQUEST_0_TRANSACTIONID=043144440L487742J
&PAYMENTREQUEST_0_TRANSACTIONTYPE=expresscheckout
&PAYMENTREQUEST_0_PAYMENTTYPE=instant
&PAYMENTREQUEST_0_ORDERTIME=2007%2d04%2d05T23%3a30%3a14Z
&PAYMENTREQUEST_0_AMT=19%2e95
&PAYMENTREQUEST_0_CURRENCYCODE=USD
&PAYMENTREQUEST_0_TAXAMT=0%2e00
&PAYMENTREQUEST_0_PAYMENTSTATUS=Pending
&PAYMENTREQUEST_0_PENDINGREASON=authorization
&PAYMENTREQUEST_0_REASONCODE=None
```

## Handling Payment Settlements With Express Checkout

You can use PayPal API operations to handle the capture of payments authorized using Express Checkout and manage Express Checkout authorization and order payment actions.

- Sale Payment Action for Express Checkout
- Authorization Payment Action for Express Checkout
- Order Payment Action for Express Checkout

## Sale Payment Action for Express Checkout

A *sale* payment action represents a single payment that completes a purchase for a specified amount.

A sale is the default Express Checkout payment action; however, you can also specify the following action in your `SetExpressCheckout` and `DoExpressCheckoutPayment` requests:

```
PAYMENTREQUEST_n_PAYMENTACTION=Sale
```

A sale is the most straightforward payment action. Choose this payment action if the transaction, including shipping of goods, can be completed immediately. To use this payment action:

- The final amount of the payment must be known when you invoke the `DoExpressCheckoutPayment` API operation
- You should intend to fulfill the order immediately, such as would be the case for digital goods or for items you have in stock for immediate shipment

After you execute the `DoExpressCheckoutPayment` API operation, the payment is complete and further action is unnecessary. You cannot capture a further payment or void any part of the payment when you use this payment action.

## Authorization Payment Action for Express Checkout

An *authorization* payment action represents an agreement to pay and places the buyer’s funds on hold for up to three days.

To set up an authorization, specify the following payment action in your `SetExpressCheckout` and `DoExpressCheckoutPayment` requests:

```
PAYMENTREQUEST_n_PAYMENTACTION=Authorization
```

An authorization enables you to capture multiple payments up to 115% of, or USD $75 more than, the amount you specify in the `DoExpressCheckoutPayment` request. Choose this payment action if you need to ship the goods before capturing the payment or if there is some reason not to accept the payment immediately.

The *honor period*, for which funds can be held, is three days. The *valid period*, for which the authorization is valid, is 29 days. You can reauthorize the 3-day honor period at most once within the 29-day valid period.

You can void an authorization, in which case, the uncaptured part of the amount specified in the `DoExpressCheckoutPayment` request becomes void and can no longer be captured. If no part of the payment has been captured, the entire payment becomes void and nothing can be captured.

**API operations associated with Authorization payment action in Express Checkout**

| API Operation | Description |
|---|---|
| `DoCapture` | Capture an authorized payment |
| `DoReauthorization` | Reauthorize a payment |
| `DoVoid` | Void an order or an authorization |

## Order Payment Action for Express Checkout

An *order* payment action represents an agreement to pay one or more authorized amounts up to the specified total over a maximum of 29 days.

To set up an order, specify the following payment action in your `SetExpressCheckout` and `DoExpressCheckoutPayment` requests:

```
PAYMENTREQUEST_n_PAYMENTACTION=Order
```

An order enables you to create multiple authorizations over the 29 days; each authorization you create places the buyer’s funds on hold for up to three days. You can capture multiple payments for each authorization, up to 115% of, or USD $75 more than, the amount you specify in the `DoExpressCheckoutPayment` request.

**NOTE:** The default number of child authorizations in your PayPal account is 1. To do multiple authorizations please contact PayPal to request an increase.

This payment action provides the most flexibility and should be used when either a sale or one authorization plus one reauthorization do not meet your needs. Situations in which orders are appropriate include the handling of

- Back orders, in which available merchandise is sent immediately and the remaining merchandise is sent when available, which may include more than two shipments
- Split orders, in which merchandise is sent in more than one shipment, perhaps to different addresses, and you want to collect a payment for each shipment
- Drop shipments, which are shipments from other vendors for which you accept the payment

You cannot reauthorize an authorization. You handle the need to reauthorize, for example when the hold period or valid period of an authorization expires, simply by creating another authorization.

You can void an order or an authorization created from the order. If you void an order, the uncaptured part of the amount specified in the `DoExpressCheckoutPayment` request becomes void and can no longer be captured. If no part of the payment has been captured, the entire payment becomes void and nothing can be captured.

If you void an authorization associated with the order, the uncaptured part of the amount specified in the authorization becomes void and can no longer be captured. If no part of the authorization has been captured, the entire authorized payment becomes void.

**API operations associated with Order payment action in Express Checkout**

| API Operation | Description |
|---|---|
| `DoAuthorization` | Authorize a payment |
| `DoCapture` | Capture an authorized payment |
| `DoVoid` | Void an order or an authorization |

## Issuing Refunds

You can use the `RefundTransaction` PayPal API operation to issue refunds.

Use the `RefundTransaction` API to issue one or more refunds associated with a transaction, such as a transaction created by a capture of a payment. The transaction is identified by a transaction ID that PayPal assigns when the payment is captured.

**NOTE:** You cannot make a refund if the transaction occurred after the refund period has passed; typically, the refund period is 60 days.

You can refund amounts up to the total amount of the original transaction. If you specify a a full refund, the entire amount is refunded. If you specify a partial refund, you must specify the amount to refund, the currency, and a description of the refund, which is called a *memo*.

When you call the `RefundTransaction` API, PayPal responds with another transaction ID, which is associated with the refund (not the original transaction), and additional information about the refund. This information identifies

- the gross amount of the refund, which is returned to the payer
- the amount of the refund associated with the original transaction fee, which is returned to you
- the net amount of the refund, which is deducted from your balance

To issue a refund

1. In the `RefundTransaction` request, specify the transaction ID of the transaction whose payment you want to refund.

   `TRANSACTIONID=`*transaction_id*

2. Specify the kind of refund, which is either `Full` or `Partial`.

```
REFUNDTYPE=Full
```

or

```
REFUNDTYPE=Partial
```

3. For a partial refund, specify the refund amount, including the currency.

```
AMT=amount
CURRENCYCODE=currencyID
```

4. For a partial refund, specify the memo description.

```
NOTE=description
```

5. Execute the `RefundTransaction` operation.
6. Check the acknowledgement status in the `RefundTransaction` response to ensure that the operation was successful.

# 3 Handling Recurring Payments

Set up a recurring payment to handle subscription and other payments that occur on a fixed schedule.

## How Recurring Payments Work

When you support recurring payments for a buyer, you create a *recurring payments profile*. The profile contains information about the recurring payments, including details for an optional trial period and a regular payment period. Both periods contain information about the payment frequency and payment amounts, including shipping and tax, if applicable.

After a profile is created, PayPal automatically queues payments based on the billing start date, billing frequency, and billing amount. Payments reoccur until the profile expires, there are too many failed payments to continue, or you cancel the profile.

**NOTE:** When using Express Checkout, the buyer can also cancel a recurring payments profile.

When a buyer uses Express Checkout, queued payments are funded using the normal funding source hierarchy within the buyer's PayPal account.

After you create a recurring payments profile, you can view recurring payments details or cancel the recurring payments profile from your PayPal account.You can also access recurring payments reports from the PayPal **Business Overview** page.

Also, after creating a recurring payments profile, you can use the Recurring Payments API to do the following:

- Get information details about a recurring payments profile.
- Change the status of a recurring payments profile.

- Update the details of the recurring payments profile.
- Bill the outstanding amount of the recurring payments profile.

### Limitations

The current release of the Recurring Payments API has the following limitations:

- A profile can have at most one optional trial period and a single regular payment period.
- The profile start date may not be earlier than the profile creation date.

Recurring payments with Express Checkout also has the following limitations:

- To be able to create a recurring payments profile for the buyer, the buyer's PayPal account must include an active credit card.
- At most ten recurring payments profiles can be created during checkout.
- You can only increase the profile amount by 20% in each 180-day interval after the profile is created.

## Recurring Payments Terms

Some terms are commonly used by PayPal in the context of recurring payments.

**Recurring payments terms**

| Term | Definition |
|---|---|
| Recurring payments profile | Your record of a recurring transaction for a single buyer. The profile includes all information required to automatically bill the buyer a fixed amount of money at a fixed interval. |
| Billing cycle | One payment is made per billing cycle. Each billing cycle is made up of two components.<br>• The billing period specifies the unit to be used to calculate the billing cycle (such as days or months).<br>• The billing frequency specifies the number of billing periods that make up the billing cycle.<br>For example, if the billing period is Month and the billing frequency is 2, the billing cycle will be two months. If the billing period is Week and the billing frequency is 6, the payments will be scheduled every 6 weeks. |
| Regular payment period | The main subscription period for this profile, which defines a payment amount for each billing cycle. The regular payment period begins after the trial period, if a trial period is specified for the profile. |
| Trial period | An optional subscription period before the regular payment period begins. A trial period may not have the same billing cycles and payment amounts as the regular payment period. |

| Term | Definition |
|---|---|
| Payment amount | The amount to be paid by the buyer for each billing cycle. |
| Outstanding balance | If a payment fails for any reason, that amount is added to the profile's outstanding balance. |
| Profile ID | An alphanumeric string (generated by PayPal) that uniquely identifies a recurring profile. You can specify the Profile ID in the `TransactionSearch` API operation to obtain all payments associated with the identified profile. |

## Options for Creating a Recurring Payments Profile

You can create a recurring payments profile that allows a regular payment period, an optional trial period, an initial payment, and other options.

### Specifying the Regular Payment Period

Each recurring payments profile has a regular payment period that defines the amount and frequency of the payment. The following table lists the required fields for specifying the regular payment period.

**Required fields for specifying a regular payment period**

| NVP | SOAP | Description |
|---|---|---|
| `PROFILESTARTDATE` | `RecurringPaymentsProfileDetails.BillingStartDate` | The date when billing for this profile begins.<br>**NOTE:** The profile may take up to 24 hours for activation. |
| `BILLINGPERIOD` | `ScheduleDetails.PaymentPeriod.BillingPeriod` | The unit of measure for the billing cycle. Must be one of:<br>● `Day`<br>● `Week`<br>● `SemiMonth`<br>● `Month`<br>● `Year` |
| `BILLINGFREQUENCY` | `ScheduleDetails.PaymentPeriod.BillingFrequency` | Number of billing periods that make up one billing cycle.<br>**NOTE:** The combination of billing frequency and billing period must be less than or equal to one year.<br>**NOTE:** If the billing period is `SemiMonth`., the billing frequency must be 1. |

| NVP | SOAP | Description |
|---|---|---|
| `AMT` | `ScheduleDetails.PaymentPeriod.Amount` | Amount to bill for each billing cycle. |

You can optionally include a value for `TOTALBILLINGCYCLES` (SOAP field `ScheduleDetails.PaymentPeriod.TotalBillingCycles`), which specifies the total number of billing cycles in the regular payment period. If no value is specified or if the value is 0, the payments continue until the profile is canceled or suspended. If the value is greater than 0, the regular payment period will continue for the specified number of billing cycles.

You can also specify an optional shipping amount or tax amount for the regular payment period.

## Including an Optional Trial Period

You can optionally include a trial period in the profile by specifying the following fields in the `CreateRecurringPaymentsProfile` request. The following table lists the required fields for creating an optional trial period.

### Required fields for specifying a trial period

| NVP | SOAP |
|---|---|
| `TRIALBILLINGPERIOD` | `ScheduleDetails.TrialPeriod.BillingPeriod` |
| `TRIALBILLINGFREQUENCY` | `ScheduleDetails.TrialPeriod.BillingFrequency` |
| `TRIALAMT` | `ScheduleDetails.TrialPeriod.Amount` |
| `TRIALTOTALBILLINGCYCLES` | `ScheduleDetails.TrialPeriod.TotalBillingCycles` |

## Specifying an Initial Payment

You can optionally specify an initial non-recurring payment when the recurring payments profile is created by including the following fields in the `CreateRecurringPaymentsProfile` request:

### Required fields for specifying an initial payment

| NVP | SOAP |
|---|---|
| `INITAMT` | `ScheduleDetails.ActivationDetails.InitialAmount` |
| `FAILEDINITAMTACTION` | `ScheduleDetails.ActivationDetails.FailedInitAmountAction` |

By default, PayPal will not activate the profile if the initial payment amount fails. You can override this default behavior by setting the `FAILEDINITAMTACTION` field to `ContinueOnFailure`, which indicates that if the initial payment amount fails, PayPal should add the failed payment amount to the outstanding balance due on this recurring payment profile.

If this field is not set or is set to `CancelOnFailure`, PayPal will create the recurring payment profile, but will place it into a pending status until the initial payment is completed. If the initial payment clears, PayPal will notify you by IPN that the pending profile has been activated. If the payment fails, PayPal will notify you by IPN that the pending profile has been canceled.

The buyer will receive an email stating that the initial payment cleared or that the pending profile has been canceled if the profile was created using Express Checkout.

### Maximum Number of Failed Payments

By including the `MAXFAILEDPAYMENTS` field in the `CreateRecurringPaymentsProfile` request, you set the number of failed payments allowed before the profile is automatically suspended. You receive an IPN message when the number of failed payments reaches the maximum number specified.

### Billing the Outstanding Amount

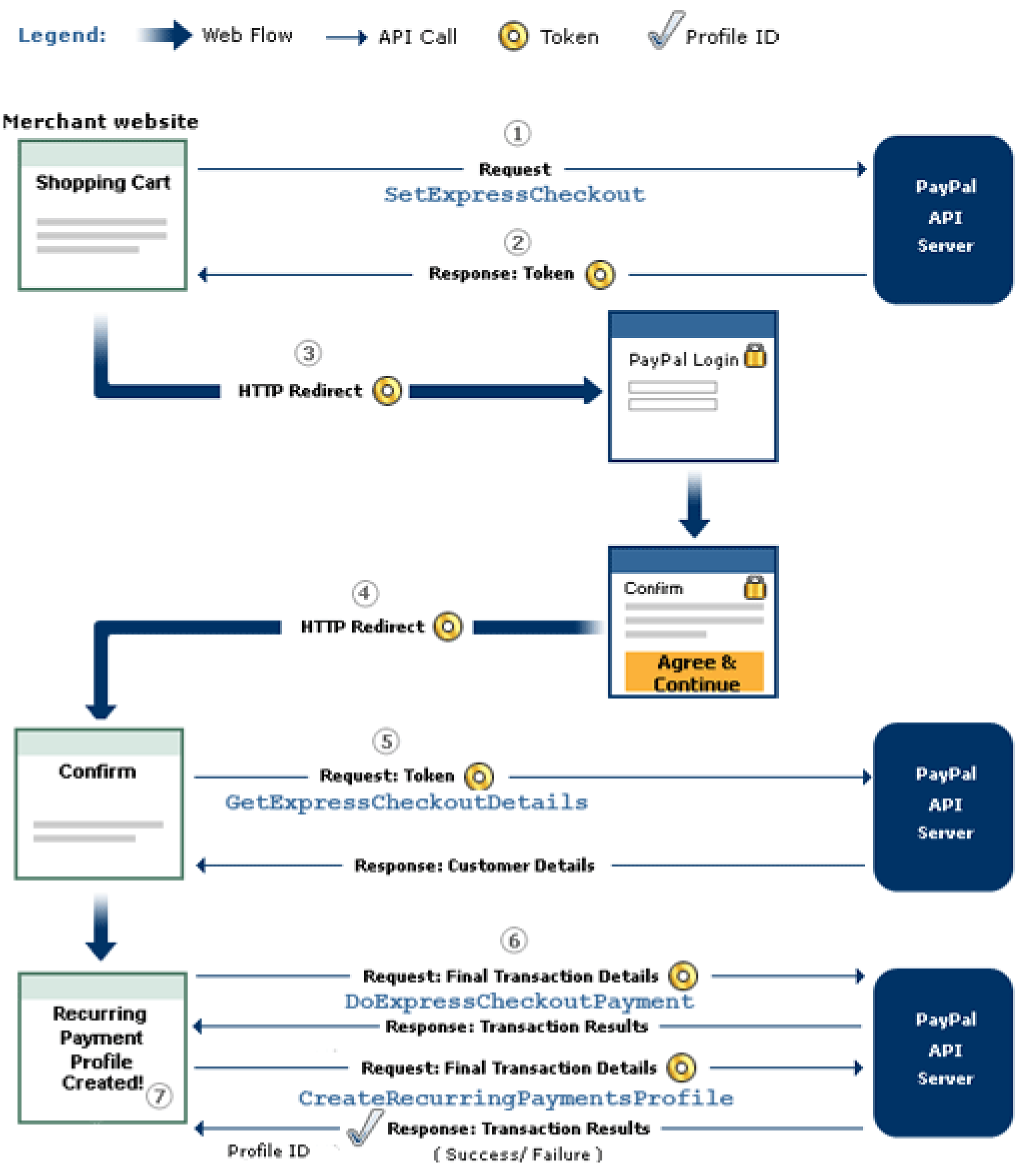
If a payment fails due to any reason, the amount that was to be billed (including shipping and tax, if applicable) is added to the profile's outstanding balance. Use the `AUTOBILLOUTAMT` field in the `CreateRecurringPaymentsProfile` request to specify whether or not the outstanding amount should be added to the payment amount for the next billing cycle.

Whether or not you choose to include the outstanding amount with the payment for the next billing cycle, you can also use the `BillOutstandingAmount` API to programmatically collect that amount at any time.

## Recurring Payments With Express Checkout

During the Express Checkout flow, you can create one or more recurring payments and mix recurring payments with other purchases.

The following diagram illustrates the typical processing flow to create recurring payments during checkout.



The circled numbers in the diagram correspond to the following steps:

## Recurring payments processing flow

| Step | Merchant... | PayPal... |
|---|---|---|
| 1 | Calls `SetExpressCheckout` with one or more billing agreement details in the request | |
| 2 | | Returns a token, which identifies the transaction, to the merchant. |

| Step | Merchant... | PayPal... |
|---|---|---|
| 3 | Redirects buyer's browser to: `https://www.paypal.com/cgi-bin/webscr?cmd=_express-checkout&token=<`*`token returned by`*` SetExpressCheckout>` | |
| | | Displays login page. Allows user to select payment options and shipping address. |
| 4 | | Redirects buyer's browser to `returnURL` passed to `SetExpressCheckout` if buyer agrees to payment description. |
| 5 | Calls `GetExpressCheckoutDetails` to get buyer information (optional). | |
| | | Returns `GetExpressCheckoutDetails` response. |
| | Displays merchant review page for buyer. | |
| 6 | Calls `DoExpressCheckoutPayment` if the order includes one-time purchases as well as a recurring payment. Otherwise, skip this step. | |
| | | Returns `DoExpressCheckoutPayment` response |
| | Calls `CreateRecurringPaymentsProfile` one time for each recurring payment item included in the order. | |
| | | Returns `ProfileID` in `CreateRecurringPaymentsProfile` response for each profile successfully created. |
| 7 | Displays successful transaction page. | |

## Initiating the Processing Flow With SetExpressCheckout

As in the Express Checkout flow, the `SetExpressCheckout` request notifies PayPal that you are:

- Initiating an order that can be either a one-time purchase, up to ten recurring payments, or a mixture of a one-time purchase and recurring payments
- Initiating the processing flow to create one or more billing agreements for recurring payments with no associated one-time purchase or recurring payment

**NOTE:** You can also initiate the processing flow using `SetCustomerBillingAgreement` for orders that contain only a single recurring payment.

Typically, you set the amount of the payment for an Express Checkout transaction when you call the `SetExpressCheckout` request and confirm the amount in the `DoExpressCheckoutPayment` request. If, however, you set the amount of the payment to 0 in the `SetExpressCheckout` request, you can create a billing agreement without initiating a payment.

**NOTE:** To create a billing agreement without purchase, use Version 54.0 or higher, of the PayPal API.

To set up one or more billing agreements for recurring payments, modify the `SetExpressCheckout` request as follows:

1. Add an `L_BILLINGTYPE`*n* field for each billing agreement you want to create; *n* is a value in the range of `0` to `9`, inclusive. Set the value of each field to `RecurringPayments`.

   ```
   L_BILLINGTYPE0=RecurringPayments
   ```

2. Add an `L_BILLINGAGREEMENTDESCRIPTION`*n* field to correspond to each `L_BILLINGTYPE`*n* field you pass; *n* is a value in the range of `0` to `9`, inclusive. Set the value of each field to the description of the goods or services associated with that billing agreement, for example:

   ```
   L_BILLINGAGREEMENTDESCRIPTION0=Time Magazine subscription
   ```

3. If there is no associated purchase, set `AMT` to `0`.

   ```
   AMT=0
   ```

4. (Optional) Set `MAXAMT` to the average expected transaction amount.

   PayPal uses the value you pass to validate the buyer's funding source for recurring payments. If you do not specify a value, the default is 25.00.

**NOTE:** When creating the recurring payments profile, pass the same `L_BILLINGTYPE`*n* value and `L_BILLINGAGREEMENTDESCRIPTION`*n* string describing the goods or service in the call to `CreateRecurringPaymentsProfile` that you passed in the call to `SetExpressCheckout`.

The `SetExpressCheckout` response provides a token that uniquely identifies the transaction for subsequent redirects and API calls.

## Redirecting the Buyer's Browser to PayPal

After you receive a successful response from `SetExpressCheckout`, add the `TOKEN` from the `SetExpressCheckout` response as a name/value pair to the following URL, and redirect your buyer's browser to it:

```
https://www.paypal.com/cgi-bin/webscr?cmd=_express-checkout&
token=<value_from_SetExpressCheckoutResponse>
```

For redirecting the buyer's browser to the PayPal login page, PayPal recommends that you use the HTTPS response 302 "Object Moved" with the URL above as the value of the `Location` header in the HTTPS response. Ensure that you use an SSL-enabled server to prevent browser warnings about a mix of secure and insecure graphics.

### Getting Buyer Details Using GetExpressCheckoutDetails

The `GetExpressCheckoutDetails` method returns information about the buyer, including name and email address stored on PayPal. You can optionally call this API after PayPal redirects the buyer's browser to the `ReturnURL` you specified in the `SetExpressCheckout` request.

The `GetExpressCheckoutDetails` request has one required parameter, `TOKEN`, which is the value returned in the `SetExpressCheckout` response.

The values you specified for the following parameter fields are not returned in the `GetExpressCheckoutDetails` response unless the transaction includes a purchase. The fields are ignored if you set up a billing agreement for a recurring payment that is not immediately charged.

- `DESC`
- `CUSTOM`
- `INVNUM`

### Creating the Profiles With CreateRecurringPaymentsProfile

After your buyer has agreed to the recurring payments billing agreement on your confirmation page, you must call `CreateRecurringPaymentsProfile` to create the profile. If you are creating multiple recurring payments profiles, you must call `CreateRecurringPaymentsProfile` once for each profile to be created.

If the transaction includes a mixture of a one-time purchase and recurring payments profiles, call `DoExpressCheckoutPayment` to complete the one-time purchase transaction, and then call `CreateRecurringPaymentsProfile` for each recurring payment profile to be created.

**IMPORTANT:** The recurring payments profile is not created until you receive a success response from the `CreateRecurringPaymentsProfile` call.

The `CreateRecurringPaymentsProfile` response contains a Profile ID, which is an encoded string that uniquely identifies the recurring payments profile.

## Recurring Payments Profile Status

The recurring payments actions you may take, depend on the status of the profile.

A recurring payments profile can have one of the following status values:

- `ActiveProfile`
- `PendingProfile`

- `ExpiredProfile`
- `SuspendedProfile`
- `CancelledProfile`

If the profile is successfully created, it has an `ActiveProfile` status. However, if a non-recurring initial payment fails and `FAILEDINITAMTACTION` is set to `CancelOnFailure` in the `CreateRecurringPaymentsProfile` request, the profile is created with a status of `PendingProfile` until the initial payment either completes successfully or fails.

A profile has a status of `ExpiredProfile` when both the total billing cycles for both the optional trial period and the regular payment period have been completed.

You can suspend or cancel a profile by using the `ManageRecurringPaymentsProfileStatus` API. You can also reactivate a suspended profile. If the maximum number of failed payments has already been reached, however, you will need to increase the number of failed payments before reactivating the profile.

**NOTE:** You can also suspend, cancel, or reactive a recurring payments profile through the PayPal website.

For recurring payments profiles created with Express Checkout, the buyer receives an email about the change in status of their recurring payment.

## Getting Recurring Payments Profile Information

Use the `GetRecurringPaymentsProfileDetails` API to get information about a profile.

**NOTE:** You can also get information about recurring payments profiles from the PayPal website.

Along with the information that you specified in the `CreateRecurringPaymentsProfile` request, `GetRecurringPaymentsProfileDetails` also returns the following summary information about the profile:

- Profile status
- Next scheduled billing date
- Number of billing cycles completed in the active subscription period
- Number of billing cycles remaining in the active subscription period
- Current outstanding balance
- Total number of failed billing cycles
- Date of the last successful payment received
- Amount of the last successful payment received

## Modifying a Recurring Payments Profile

Use the `UpdateRecurringPaymentsProfile` API to modify a recurring payments profile.

**NOTE:** You can also modify recurring payments profiles from the PayPal website.

You can only modify the following specific information about an active or suspended profile:

- Subscriber name or address
- Past due or outstanding amount
- Whether to bill the outstanding amount with the next billing cycle
- Maximum number of failed payments allowed
- Profile description and reference
- Number of additional billing cycles
- Billing amount, tax amount, or shipping amount

**NOTE:** You cannot modify the billing frequency or billing period of a profile. You can modify the number of billing cycles in the profile.

**NOTE:** For recurring payments with Express Checkout, certain updates, such as billing amount, are not allowed within 3 days of the scheduled billing date, and an error is returned.

You can modify the following profile information during the trial period or regular payment period.

- Billing amount
- Number of billing cycles

The profile changes take effect with the next payment after the call to update the profile. Say, for example, the buyer has made one trial payment out of a total of three. `UpdateRecurringPaymentsProfile` is called to increase the number of billing cycles to five. As a result, the buyer will have four additional trial payments to make. If the call to `UpdateRecurringPaymentsProfile` is made during the regular payment period, the changes take effect with the buyer's next scheduled regular payment.

For complete details, see the *Name-Value Pair Developer Guide and Reference* or the *SOAP API Reference*.

### Updating Addresses

When you update the subscriber shipping address, you must enter all of address fields, not just those that are changing:

For example, if you want to update the subscriber's street address, you must specify all of the address fields listed in the *Name-Value Pair Developer Guide and Reference* or *SOAP API Reference*, not just the field for the street address.

### Updating the Billing Amount

For profiles created using Express Checkout, the total amount of a recurring payment can only be increased 20% in a fixed 180-day interval after the profile is created. The 20% maximum is based on the total amount of the profile at the beginning of the 180-day interval, including any shipping or tax amount.

For example, if a profile is created on March 10 with a total amount of $100, then during the 180-day interval from March 10 to September 6, you can increase the payment amount to a maximum of $120 (120% of $100).

Suppose that during the first 180-day interval, you increased the payment amount to $110. Then during the next 180-day interval (starting on September 7 in this example), you can only increase the amount of the payment to a maximum of $132 (120% of $110).

## Billing the Outstanding Amount of a Profile

Use the `BillOutstandingAmount` API to immediately bill the buyer for the current past due or outstanding amount for a recurring payments profile.

**NOTE:** You can also bill the buyer for the current past due or outstanding amount for a recurring payments profile from the PayPal website.

To bill the outstanding amount:

- The profile status must be active or suspended.

  **NOTE:** The `BillOutstandingAmount` API does not reactivate a suspended profile. You need to call `ManageRecurringProfileStatus` to do this.

- The profile must have a non-zero outstanding balance.
- The amount of the payment cannot exceed the outstanding amount for the profile.
- The `BillOutstandingAmount` call cannot be within 24 hours of a regularly scheduled payment for this profile.

**NOTE:** If another outstanding balance payment is already queued, an API error is returned.

You will be informed by IPN about the success or failure of the outstanding payment. For profiles created using Express Checkout, the buyer will receive an email notification of the payment.

## Recurring Payments Notifications

You are notified of recurring payments events through IPN and email; however, using `GetTransactionDetails` to obtain the information you need is typically sufficient.

You are notified of certain events through IPN. For recurring payments profiles created using Express Checkout, buyers are also notified of specific events by email. The following table indicates when IPN and emails are generated.

### Recurring payments IPN messages and email

| Event | IPN | Buyer Email |
|---|---|---|
| Profile successfully created | Yes | Yes |
| Profile creation failed | Yes | Yes |
| Profile canceled from paypal.com interface | Yes | Yes |
| Profile status changed using API | No | Yes |
| Profile update using API | No | Yes |
| Initial payment either succeeded or failed | Yes | Yes |
| Payment either succeeded or failed (during either trial period or regular payment period) | Yes | Yes |
| Outstanding payment either succeeded or failed | Yes | Yes |
| Maximum number of failed payments reached | Yes | No |

**NOTE:** API transactions such as `ManangeRecurringPaymentsProfileStatus` do not trigger IPN notification because the success or failure of the call is provided immediately by the API response.

# 4 Express Checkout on Mobile Devices

Express Checkout on mobile devices runs in mobile browsers with pages optimized for smaller mobile screens and mobile keyboards. If you have an Express Checkout implementation, minimal programming changes let you take advantage of the mobile checkout experience.

- About the Express Checkout Experience on Mobile Devices
- How to Invoke the Express Checkout Experience on Mobile Devices
- Mobile Platforms Supported by Express Checkout
- Limitations of Express Checkout on Mobile Devices

## About the Express Checkout Experience on Mobile Devices

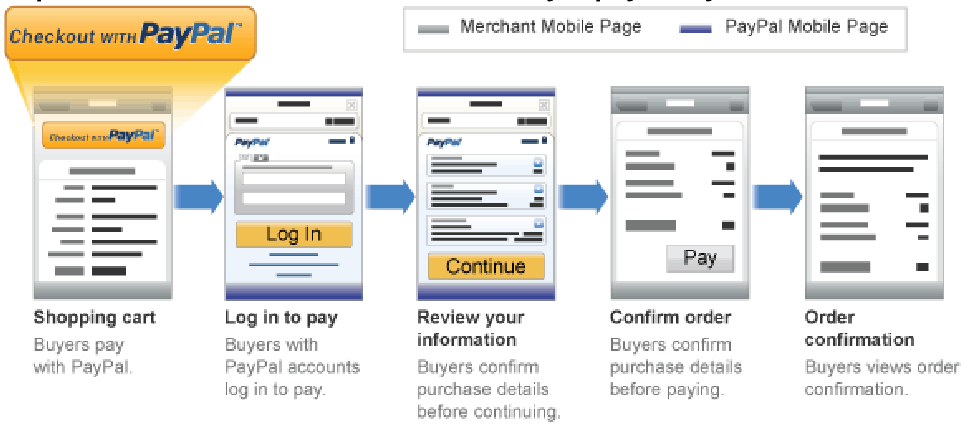
On mobile devices, Express Checkout provides payment pages tailored for faster checkout and for smaller mobile screens and keyboards.

- Express Checkout on Mobile Devices
- Express Checkout on Mobile Devices – Buyer Pays on PayPal
- Express Checkout Embedded in Mobile Applications

### Express Checkout on Mobile Devices

The Express Checkout experience on mobile devices begins on your mobile website when a buyer is ready to pay you.

***Express Checkout on mobile devices – buyer pays on your mobile website***



With Express Checkout on mobile devices, the buyer:

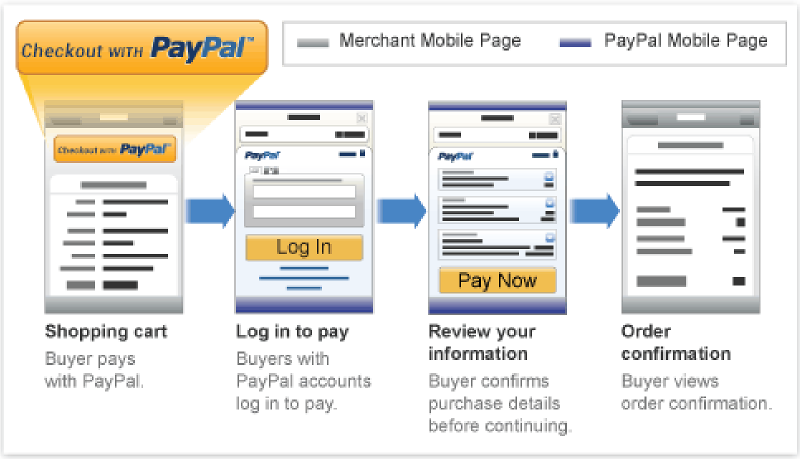
1. Presses **Checkout with PayPal** on your mobile website
2. Enters PayPal log-in credentials on the mobile PayPal log-in page, and then presses **Log in**
3. Reviews payment details on the mobile PayPal review page, and then presses **Continue**
4. Confirms the order and pays on your mobile website
5. Views order confirmation on your mobile site

**NOTE:** Only buyers with existing PayPal accounts can check out with the mobile experience. To pay with debit or credit cards, buyers click a link and enter their billing information on the full PayPal website.

## Express Checkout on Mobile Devices – Buyer Pays on PayPal

You can shorten the Express Checkout experience on mobile devices by letting buyers pay on PayPal.

***Express Checkout on mobile devices – buyer pays on PayPal***



When buyers pay on PayPal, the buyer:

1. Presses **Checkout with PayPal** on your mobile website
2. Enters PayPal log-in credentials on the mobile PayPal log-in page, and then presses **Log in**
3. Reviews payment details on the mobile PayPal review page, and then presses **Pay Now**.
4. Views order confirmation on your mobile site

To let your mobile buyers pay on PayPal, include the `useraction` parameter set to `commit` when you redirect buyers to PayPal. For example:

`https://www.paypal.com/cgi-bin/webscr?cmd=_express-checkout-mobile&`**`useraction=commit`**`&token=`*valueFromSetExpressCheckoutResponse*

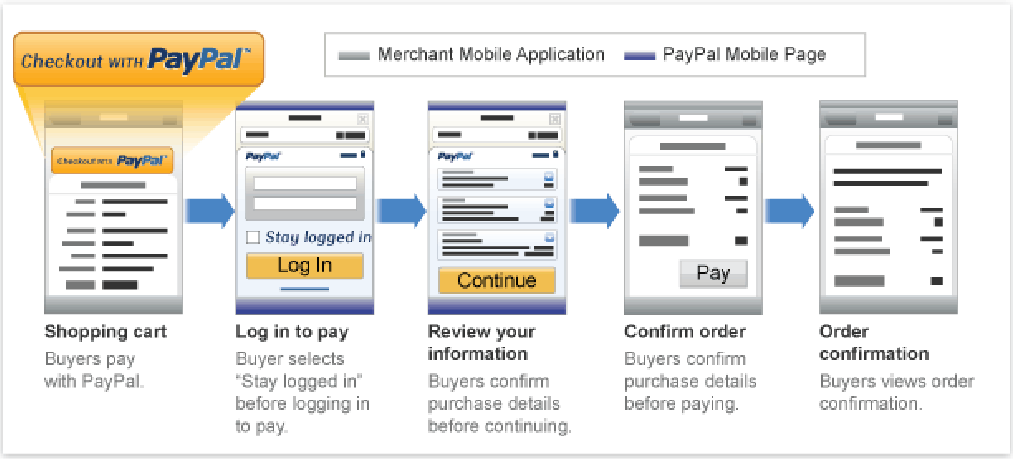
After PayPal redirects buyers to your site, call `GetExpressCheckoutDetails` and `DoExpressCheckoutPayment` to have PayPal complete the payment successfully. Call `DoExpressCheckoutPayment` without waiting for buyer interaction. Use information in the `GetExpressCheckoutDetails` response to fill out your order confirmation page.

## Express Checkout Embedded in Mobile Applications

You can embed your Express Checkout integration in your mobile applications by using the PayPal Mobile Express Checkout Library. When embedded in mobile applications, Express Checkout lets PayPal buyers stay logged in on a mobile device and skip the log-in page when approving purchases.

Buyers stay logged in on the current device only. The buyer's choice applies to all PayPal merchants on the device who also embed Express Checkout in their mobile applications. To log out of a device, buyers go to their PayPal accounts on the web.

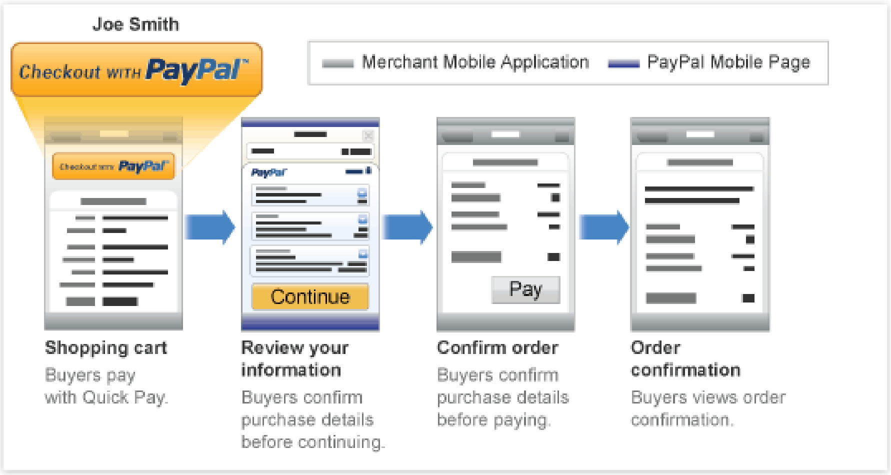
***Express Checkout in mobile applications – before buyers stay logged in***


Checkout WITH PayPal
Merchant Mobile Application
PayPal Mobile Page
Stay logged in
Log In
Continue
Pay
Shopping cart
Buyers pay with PayPal.
Log in to pay
Buyer selects "Stay logged in" before logging in to pay.
Review your information
Buyers confirm purchase details before continuing.
Confirm order
Buyers confirm purchase details before paying.
Order confirmation
Buyers views order confirmation.

When buyers choose to stay logged in on a mobile device, the buyer:

1. Presses **Checkout with PayPal** in your mobile application
2. Selects the "Keep me logged in on this device checkbox" on the mobile PayPal log-in page, and then presses **Log in**
3. Reviews payment details on the mobile PayPal review page, and then presses **Continue**
4. Completes the order and payment in your mobile application
5. Views order confirmation in your mobile application

***Express Checkout in mobile applications – after buyers stay logged in***


Joe Smith
Checkout with PayPal
Merchant Mobile Application
PayPal Mobile Page
Continue
Pay
Shopping cart
Buyers pay with Quick Pay.
Review your information
Buyers confirm purchase details before continuing.
Confirm order
Buyers confirm purchase details before paying.
Order confirmation
Buyers views order confirmation.

After buyers stay logged in on a mobile device, PayPal displays the buyer's name above the **Checkout with PayPal** button. The button displays the buyer's name wherever you and other merchants show the button in mobile applications on the device. The name reminds buyers that they are logged in to PayPal on the mobile device.

While buyers stay logged in on a mobile device, the buyer:

1. Presses **Checkout with PayPal** in your mobile application
2. Reviews payment details on the mobile PayPal review page, and then presses **Continue**
3. Completes the order and payment in your mobile application
4. Views order confirmation in your mobile application

## How to Invoke the Express Checkout Experience on Mobile Devices

You can invoke the mobile experience for Express Checkout in the following ways:

- **Mobile buyer pays on your mobile website:** use the mobile checkout command in place of the express checkout command when you redirect the buyer to PayPal. For example:

  `https://www.paypal.com/cgi-bin/webscr?`**`cmd=_express-checkout-mobile`**`&token=`*valueFromSetExpressCheckoutResponse*

  Make sure that you specify URLs for cancel and return pages on your mobile website in the `SetExpressCheckout` API request.

- **Mobile buyer pays on PayPal:** add "commit" as the `useraction` when you redirect the buyer to PayPal. For example:

  ```
  https://www.paypal.com/cgi-bin/webscr?cmd=_-express-checkout-
  mobile&useraction=commit&token=valueFromSetExpressCheckoutResponse
  ```

  After PayPal redirects buyers to your site, call `GetExpressCheckoutDetails` and `DoExpressCheckoutPayment` to have PayPal complete the payment successfully. Call `DoExpressCheckoutPayment` without waiting for buyer interaction. Use information in the `GetExpressCheckoutDetails` response to fill out your order confirmation page.

  Make sure that you specify URLs for cancel and return pages on your mobile website in the `SetExpressCheckout` API request.

- **Mobile buyer pays with Express Checkout embedded in your mobile application:** add the token from the Mobile Express Checkout Library as the `drt` variable when you start the webview and redirect the buyer to PayPal. For example:

  ```
  https://www.paypal.com/cgi-bin/webscr?cmd=_express-checkout-mobile
  &drt=valueFromMobileExpressCheckoutLibrary&token=valueFromSetExpressCheckoutRespons
  e
  ```

  Make sure that you get a new token from the library each time you start a webview and redirect the buyer to PayPal.

  Make sure that you specify cancel and return URLs. Monitor events in the webview for the redirect from PayPal, and use the URL value to determine whether to proceed with calls to `GetExpressCheckoutDetails` and `DoExpressCheckoutPayment`.

  For more information, see the Mobile Express Checkout Library Developer Guide and Reference - iOS Edition.

## Mobile Platforms Supported by Express Checkout

Express Checkout supports the mobile checkout experience on specific mobile devices and their embedded mobile browsers only.

| Supported Mobile Devices | Software Versions |
|---|---|
| Android-based devices | 2.0, 2.1 |
| iPhone® (2), 3G, 3GS, 4 | iPhone OS 2.x, 3.x; iOS 4.0 |
| iPod touch® 2G, 3G | iPhone OS 2.x, 3.x |

## Limitations of Express Checkout on Mobile Devices

Express Checkout on mobile devices limits some Express Checkout features.

- Locales Supported by Express Checkout on Mobile Devices
- Features Not Supported by Express Checkout on Mobile Devices
- Request Fields Supported by Express Checkout on Mobile Devices

## Locales Supported by Express Checkout on Mobile Devices

You can set the locale for Express Checkout pages on mobile devices to any of the countries generally supported by PayPal, with a few exceptions. PayPal displays the log-in page in the default language for the country that you specify. The default locale is United States.

### Locale Codes Not Supported by Express Checkout on Mobile Devices

The following locale codes are not supported by Express Checkout on mobile devices.

| Country | Code |
|---|---|
| Australia | `AU` |
| Canada | `CA` |
| France | `FR` |
| Italy | `IT` |
| Malaysia | `MY` |
| Netherlands | `NL` |
| United Kingdom | `GB` |
| United States | `US` |

### Handling Locales Not Supported by Express Checkout on Mobile Devices

For countries included in the list of unsupported locales, the following occurs:

- **Merchants:** If you set the locale code to an country, PayPal switches buyers to the full website checkout experience.
- **Buyers:** If a buyer logs in with a PayPal account in an unsupported country, PayPal stops the mobile checkout experience so the buyer cannot pay.

## Features Not Supported by Express Checkout on Mobile Devices

Express Checkout does not support the following features when it runs on mobile devices:

- **PayPal Account Optional:** lets buyers pay with credit and debit cards in addition to PayPal accounts.

- **PayPal account signup at end of checkout:** lets buyers without PayPal accounts enter billing information, make their payments, and then sign up for PayPal.
- **SMS security keys for mobile PayPal login:** buyers can sign up for an extra layer of security with a secure token. Buyers add a one-time secure code to their password or mobile PIN when they log in to PayPal. Buyers can use secure codes from hardware tokens but not from software tokens sent by SMS to the buyer's mobile phone.
- **Forgotten email or password:** buyers can request their email address or password during checkout, before logging in to PayPal.
- **Parallel payments:** lets buyers pay multiple merchants in a single checkout session.
- **Dynamic images:** lets PayPal update the images on your website for the PayPal button and acceptance marks automatically to coincide with PayPal campaigns.
- **Buyer experience enhancements:** lets you offer gift wrap, ask a survey question, or display your customer service number during checkout.
- **Custom payment page styles:** lets you change the overall appearance of the PayPal review page with a custom page style or with individual page style characteristics.
- **Instant Update API:** a callback that sends you the buyer's shipping address during checkout on PayPal and let's you respond with your actual shipping charges and handling charges.
- **International addresses:** lets buyers add international addresses during checkout after logging in to PayPal.
- **Promotional offers:** lets buyers pay with coupons, incentives, and buyer credit
- **Dynamic currency conversion:** lets you list an item in one currency and then accept payment in a different currency.
- **Recurring payments:** lets buyers checkout for subscriptions and other payments that occur on a fixed, recurring schedule.
- **Reference transactions:** lets you create a payment based on the transaction ID from a previous payment.

## Request Fields Supported by Express Checkout on Mobile Devices

Express Checkout on mobile devices supports a subset of the API request fields that Express Checkout supports when it runs on personal computers. PayPal ignores API request fields for features that Express Checkout does not support when it runs on mobile devices.

### NVP Request Fields Supported by Express Checkout on Mobile Devices

Express Checkout on mobile devices supports these NVP fields only.

### NVP SetExpressCheckout Request Fields Supported on Mobile Devices

| | | | |
|---|---|---|---|
| AMT | RETURNURL | CANCELURL | TOKEN |
| MAXAMT | DESC | CUSTOM | INVNUM |
| REQCONFIRMSHIPPING | NOSHIPPING | ADDROVERRIDE | LOCALECODE<br>The following values are not allowed:<br>● AU – Australia<br>● CA – Canada<br>● FR – France<br>● IT – Italy<br>● MY – Malaysia<br>● NL – Netherlands<br>● GB – United Kingdom<br>● US – United States |
| PAYMENTACTION | EMAIL | SOLUTIONTYPE<br>value must be mark | CHANNELTYPE<br>value must be merchant |
| BRANDNAME | | | |

### NVP Address Type Fields Supported by Express Checkout on Mobile Devices

| | | |
|---|---|---|
| STREET | STREET2 | CITY |
| STATE | COUNTRYCODE | ZIP |
| SHIPTOPHONENUM | | |

### NVP Payment Details Type Fields Supported by Express Checkout on Mobile Devices

| | | |
|---|---|---|
| PAYMENTREQUEST_*n*_AMT | AMT *(deprecated)* | PAYMENTREQUEST_*n*_CURRENCYCODE |
| CURRENCYCODE *(deprecated)* | PAYMENTREQUEST_*n*_ITEMAMT | ITEMAMT *(deprecated)* |
| PAYMENTREQUEST_*n*_SHIPPINGAMT | SHIPPINGAMT *(deprecated)* | PAYMENTREQUEST_*n*_INSURANCEAMT |
| INSURANCEAMT *(deprecated)* | PAYMENTREQUEST_*n*_SHIPDISCAMT | SHIPPINGDISCAMT *(deprecated)* |
| PAYMENTREQUEST_*n*_HANDLINGAMT | HANDLINGAMT *(deprecated)* | PAYMENTREQUEST_*n*_TAXAMT |
| TAXAMT *(deprecated)* | PAYMENTREQUEST_*n*_DESC | DESC *(deprecated)* |
| PAYMENTREQUEST_*n*_CUSTOM | CUSTOM *(deprecated)* | PAYMENTREQUEST_*n*_INVNUM |
| INVNUM *(deprecated)* | BUTTONSOURCE | PAYMENTREQUEST_*n*_NOTIFYURL |
| NOTIFYURL *(deprecated)* | PAYMENTREQUEST_*n*_NOTETEXT | NOTETEXT *(deprecated)* |

| | | |
|---|---|---|
| PAYMENTREQUEST_*n*_TRANSACTIONID | TRANSACTIONID *(deprecated)* | PAYMENTREQUEST_*n*_ALLOWEDPAYMENTMETHOD |
| ALLOWEDPAYMENTMETHOD *(deprecated)* | PAYMENTREQUEST_*n*_PAYMENTACTION | PAYMENTACTION *(deprecated)* |
| PAYMENTREQUEST_*n*_PAYMENTREQUESTID | PAYMENTREQUESTID *(deprecated)* | |

### NVP Payment Details Item Type Fields Supported by Express Checkout on Mobile Devices

| | | |
|---|---|---|
| L_PAYMENTREQUEST_*n*_NAME*m* | L_NAME*n* *(deprecated)* | L_PAYMENTREQUEST_*n*_DESC*m* |
| L_DESC*n* *(deprecated)* | L_PAYMENTREQUEST_*n*_AMT*m* | L_AMT*n* *(deprecated)* |
| L_PAYMENTREQUEST_*n*_NUMBER*m* | L_NUMBER*n* *(deprecated)* | L_PAYMENTREQUEST_*n*_QTY*m* |
| L_QTY*n* *(deprecated)* | L_PAYMENTREQUEST_*n*_TAXAMT*m* | L_TAXAMT*n* *(deprecated)* |
| L_PAYMENTREQUEST_*n*_ITEMWEIGHTVALUEm and L_PAYMENTREQUEST_*n*_ITEMWEIGHTUNIT*m* | L_ITEMWEIGHTVALUE*n* and L_ITEMWEIGHTUNIT*n* *(deprecated)* | L_PAYMENTREQUEST_*n*_ITEMURL*m* |
| L_ITEMURL*n* *(deprecated)* | | |

## SOAP Request Fields Supported by Express Checkout on Mobile Devices

Express Checkout on mobile devices supports these SOAP fields only.

### SOAP SetExpressCheckout Request Fields Supported on Mobile Devices

| | | | |
|---|---|---|---|
| OrderTotal | ReturnURL | CancelURL | Token |
| MaxAmount | OrderDescription | Custom | InvoiceID |
| Address | ReqConfirmShipping | NoShipping | AddressOverride |
| LocaleCode<br>allowable values are:<br>• AU – Australia<br>• CA – Canada<br>• FR – France<br>• IT – Italy<br>• MY – Malaysia<br>• NL – Netherlands<br>• GB – United Kingdom<br>• US – United States | PaymentAction | BuyerEmail | SolutionType<br>value must be mark |
| ChannelType<br>value must be merchant | BrandName | | |

### SOAP AddressType Fields Supported by Express Checkout on Mobile Devices

| | | |
|---|---|---|
| `Street1` | `Street2` | `CityName` |
| `StateOrProvince` | `Country` | `PostalCode` |
| `Phone` | | |

### SOAP PaymentDetailsType Fields Supported by Express Checkout on Mobile Devices

| | | |
|---|---|---|
| `OrderTotal` | `ItemTotal` | `ShippingTotal` |
| `InsuranceTotal` | `ShippingDiscount` | `HandlingTotal` |
| `TaxTotal` | `OrderDescription` | `Custom` |
| `InvoiceID` | `ButtonSource` | `NotifyURL` |
| `ShipToAddress` | `PaymentDetailsItem` | `NoteText` |
| `TransactionId` | `AllowedPaymentMethodType` | `PaymentAction` |
| `PaymentRequestID` | | |

### SOAP PaymentDetailsItemType Fields Supported by Express Checkout on Mobile Devices

| | | |
|---|---|---|
| `Name` | `Description` | `Amount` |
| `Number` | `Quantity` | `Tax` |
| `ItemWeight` | `ItemURL` | |

# 5 PayPal Name-Value Pair API Basics

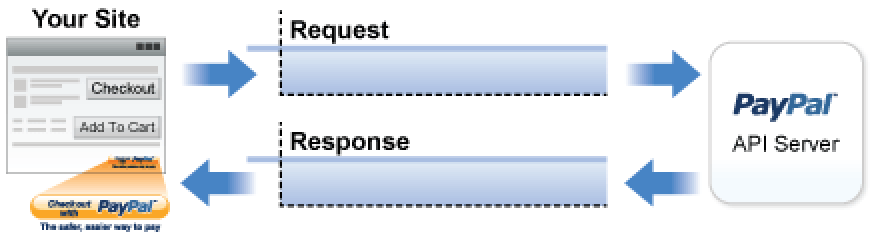
The Name-Value Pair (NVP) API provides parameter-based association between request and response fields of a message and their values. The request message is sent via the API from your website and a response message is returned by PayPal using a client-server model in which your site is a client of the PayPal server.

**NOTE:** The PayFlow API also uses name-value pairs to provide parameter-based association between request and response fields of a message and their values; however, the PayFlow API is not the same as the NVP API; for more information about the PayFlow API, see Website Payments Pro Payflow Edition Developer Guide.

## PayPal API Client-Server Architecture

The PayPal API uses a client-server model in which your website is a client of the PayPal server.

A page on your website initiates an action on a PayPal API server by sending a request to the server. The PayPal server responds with a confirmation that the requested action was taken or or indicates that an error occurred. The response might also contain additional information related to the request. The following diagram shows the basic request-response mechanism.

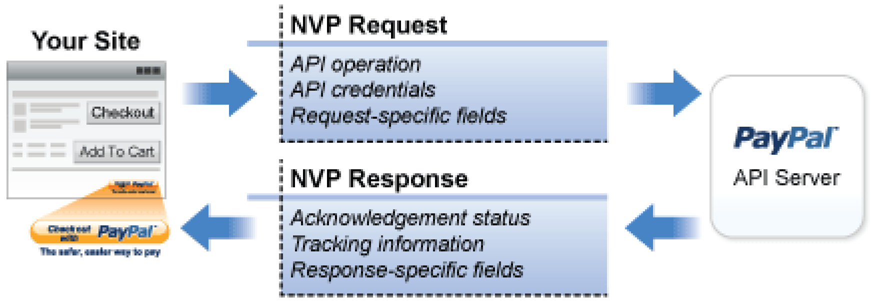


For example, you might want to obtain the buyer’s shipping address from PayPal. You can initiate a request specifying an API operation that gets buyer details. The response from the PayPal API server contains information about whether the request was successful. If the operation succeeds, the response contains the requested information. In this case, the response contains the buyer’s shipping address. If the operation fails, the response contains one or more error messages.

## PayPal Name-Value Pair API Requests and Responses

To perform a PayPal NVP API operation, you send an NVP-formatted request to a PayPal NVP server and interpret the response.

In the following diagram, your website generates a request. The request is executed on a PayPal server and the response is returned to your site.



The request identifies

- The name of the API operation to be performed and its version
- Credentials that identify the PayPal account making the request
- Request-specific information that controls the API operation to be performed

A PayPal API server performs the operation and returns a response. The response contains

- An acknowledgement status that indicates whether the operation was a success or failure and whether any warning messages were returned
- Information that can be used by PayPal to track execution of the API operation
- Response-specific information required to fulfill the request

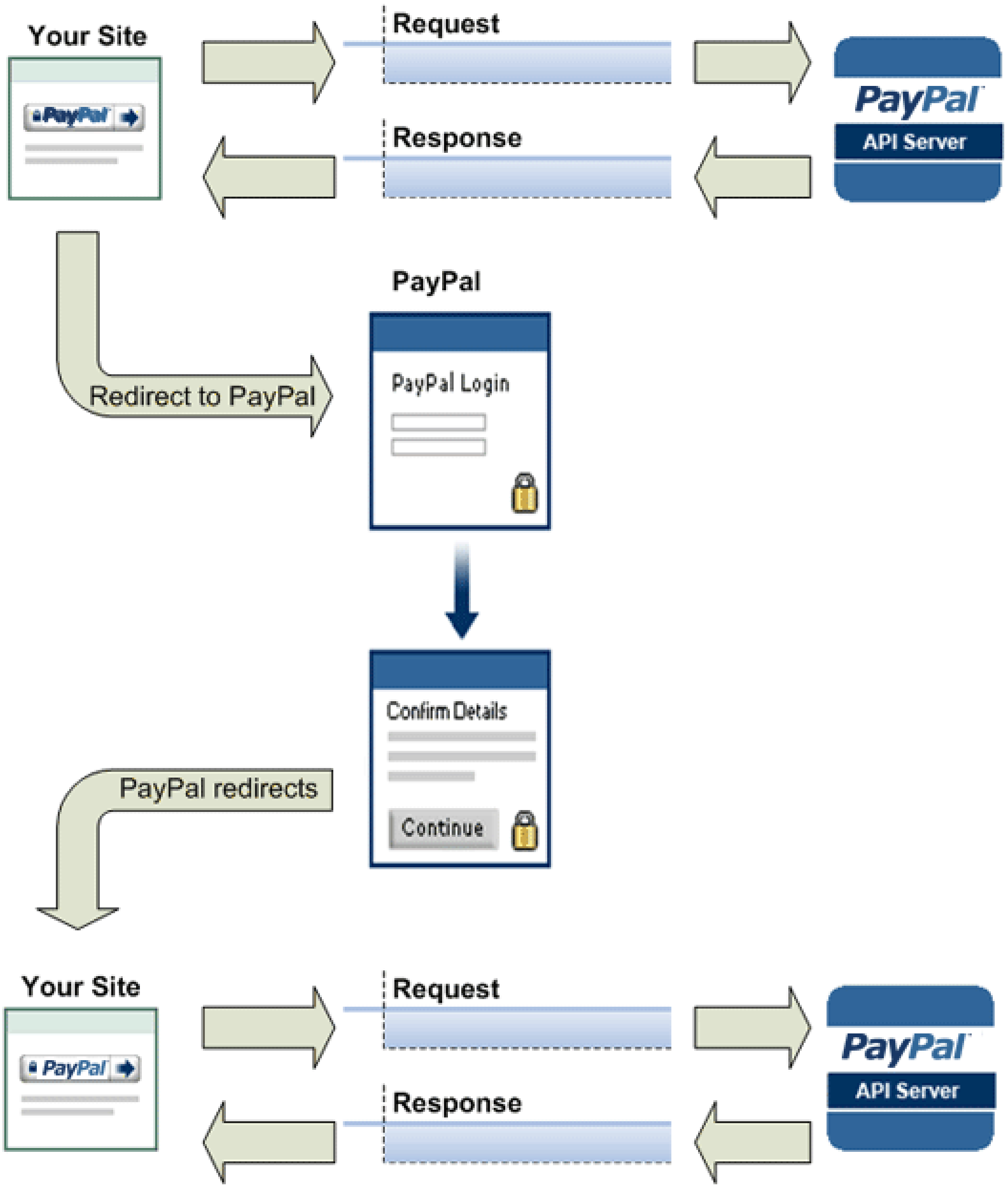
## Multiple API Operations

Some of the features, such as Express Checkout, require you to call multiple API operations.

Typically, these features require you to

1. Invoke an API operation, such as `SetExpressCheckout`, that sets up the return URL to which PayPal redirects your buyer's browser after the buyer finishes on PayPal. Other setup also can be performed by this API operation.
2. Invoke additional API operations after receiving the buyer's permission on PayPal, for example, `GetExpressCheckoutDetails` or `DoExpressCheckoutPayment`.

The following diagram shows the execution flow between your site and PayPal:



## Token Usage

Typically, the API operation that sets up a redirection to PayPal returns a token. This token is passed as a parameter in the redirect to PayPal. The token also might be required in related API operations.

## Obtaining API Credentials

To use the PayPal API, you must have API credentials that identify you as a PayPal Business account holder who is authorized to perform various API operations. Although you can use either an API *signature* or a *certificate* for credentials, PayPal recommends you use a signature.

**IMPORTANT:** Although you can have both a signature and certificate, you cannot use both at the same time.

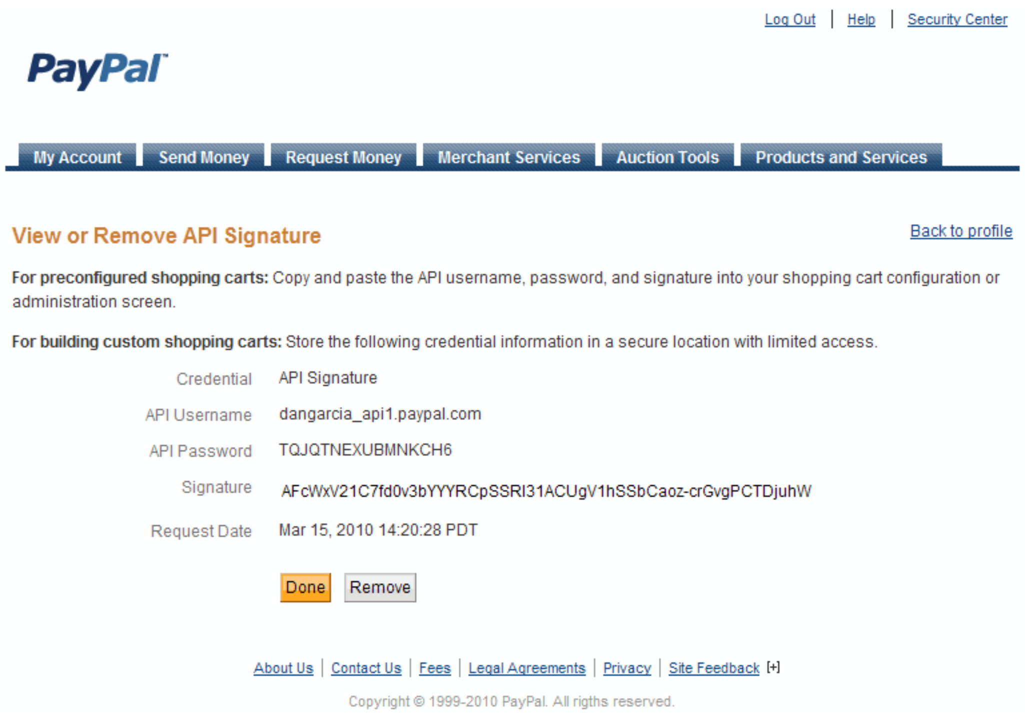
### Creating an API Signature

An API signature consists of an API username along with an associated API password and signature, all of which are assigned by PayPal. You need to include this information whenever you execute a PayPal API operation.

You must have a PayPal Business account to create a signature.

To create an API signature:

1. Log into PayPal, then click **Profile** under **My Account**.
2. Click **API Access**.
3. Click **Request API Credentials**.
4. Check **Request API signature** and click **Agree and Submit**.



View or Remove API Signature

**For preconfigured shopping carts:** Copy and paste the API username, password, and signature into your shopping cart configuration or administration screen.

**For building custom shopping carts:** Store the following credential information in a secure location with limited access.

| | |
|---|---|
| Credential | API Signature |
| API Username | dangarcia_api1.paypal.com |
| API Password | TQJQTNEXUBMNKCH6 |
| Signature | AFcWxV21C7fd0v3bYYYRCpSSRI31ACUgV1hSSbCaoz-crGvgPCTDjuhW |
| Request Date | Mar 15, 2010 14:20:28 PDT |

Copyright © 1999-2010 PayPal. All rigths reserved.

5. Click **Done to complete the process**.

## Creating an API Certificate

Create an API certificate only if your website requires it. Typically, you want to create an API signature for your credentials instead.

If you do need a certificate, follow the instructions at https://www.paypal.com/IntegrationCenter/ic_api-certificate.html.

**NOTE:** The certificate for API credentials is not the same as an SSL certificate for your website; they are not related to each other.

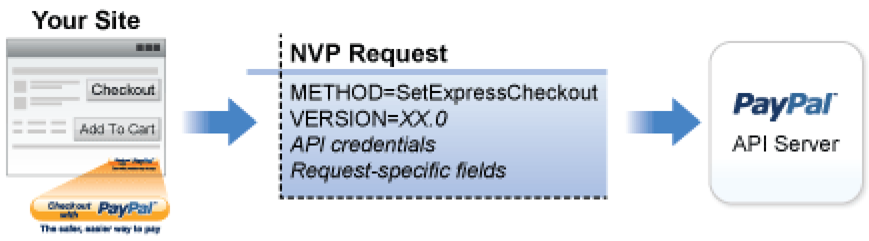
## Creating an NVP Request

The Name-Value Pair request format specifies the API operation to perform, credentials that authorize PayPal to access your account, and fields containing additional information to be used in the request.

## Specifying the PayPal API Operation

For the NVP version of the PayPal API, you must specify the name of the PayPal API operation to execute in each request along with the version of the API operation.

The following diagram shows the API operation part of an NVP request:



A *method* specifies the PayPal operation you want to execute and each method is associated with a *version*. Together, the method and version define the exact behavior of the API operation. Typically, the behavior of an API operation does not change between versions; however, you should carefully retest your code whenever you change a version.

To specify a method and version number:

**1.** Choose the PayPal API operation you want to use.

`METHOD=`*operation*

**2.** Choose the appropriate version.

In most cases, you should use the latest version of the API operation.

`VERSION=`*version_number*

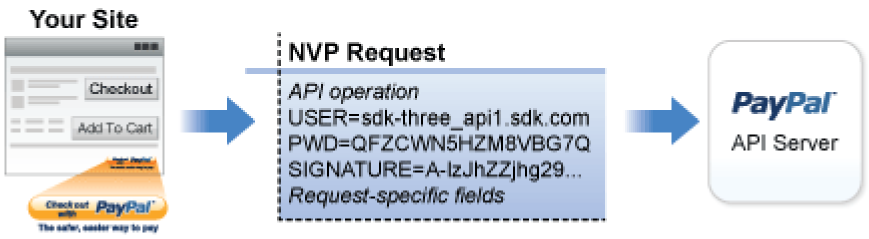
### Example of setting the API operation and version using PHP

```
function PPHttpPost($methodName_, $nvpStr_) {
  ...
  $version = urlencode('XX.0');  // NVPRequest for submitting to server
  $nvpreq ="METHOD=$methodName_&VERSION=$version...$nvpStr_";
  ...
}
```

## Specifying an API Credential

You must specify API credentials in each request to execute a PayPal API operation.

When you execute a PayPal API operation, you use credentials, such as a signature, to authenticate that you are requesting the API operation. The following diagram shows the API credentials part of an NVP request:



To enable PayPal to authenticate your request

1. Specify the API user name associated with your account.

USER=*API_username*

2. Specify the password associated with the API user name.

PWD=*API_password*

3. If you are using an API signature and not an API certificate, specify the API signature associated with the API username.

SIGNATURE=*API_signature*

### Specifying Credentials using cURL

The following example shows one way to specify a signature using cURL:

```
curl --insecure https://api-3t.sandbox.paypal.com/nvp -d ^
"METHOD=DoDirectPayment^
&VERSION=XX.0^
&USER=API_username^
&PWD=API_password^
&SIGNATURE=API_signature^
&..."
```

**NOTE:** This example does not establish a secure connection and should not be used live on paypal.com.

## URL Encoding

All requests to execute PayPal API operations sent via HTTP must be URL encoded.

The PayPal NVP API uses the HTTP protocol to send requests and receive responses from a PayPal API server. You must encode all data sent using the HTTP protocol because data that is not encoded could be misinterpreted as part of the HTTP protocol instead of part of the request. Most programming languages provide a way to encode strings in this way. You should consistently URL encode the complete API request; otherwise, you may find that unanticipated data causes an error.

**NOTE:** An HTTP form is automatically URL encoded by most browsers.

### List Syntax for Name-Value Pairs

The PayPal API uses a special syntax for NVP fields defined as lists.

The NVP interface to the PayPal API requires a unique name for each field. In the API, lists are prefixed by `L_`. To identify an element within the list, use the offset from the beginning of the list, starting with 0 as the first element. For example, `L_DESC0` is the first line of a description, `L_DESC1`, is the second line, and so on.

**NOTE:** Not all lists follow the `L_` prefix convention; however, all lists start with 0 as the first element.

## Executing NVP API Operations

You execute an PayPal NVP API operation by submitting an HTTP POST request to a PayPal API server.

### Specifying a PayPal Server

You execute a PayPal API operation by submitting the request to a PayPal API server.

To execute a PayPal NVP API operation, submit your complete request to one of the following end points:

| Server end point | Description |
|---|---|
| `https://api-3t.sandbox.paypal.com/nvp` | Sandbox server for use with API signatures; use for testing your API |
| `https://api-3t.paypal.com/nvp` | PayPal “live” production server for use with API signatures |
| `https://api.sandbox.paypal.com/nvp` | Sandbox server for use with API certificates; use for testing your API |
| `https://api.paypal.com/nvp` | PayPal “live” production server for use with API certificates |

**NOTE:** You must use different API credentials for each server end point. Typically, you obtain API credentials when you test in the Sandbox and then obtain another set of credentials for the production server. You must change each API request to use the new credentials when you go live.

### Logging API Operations

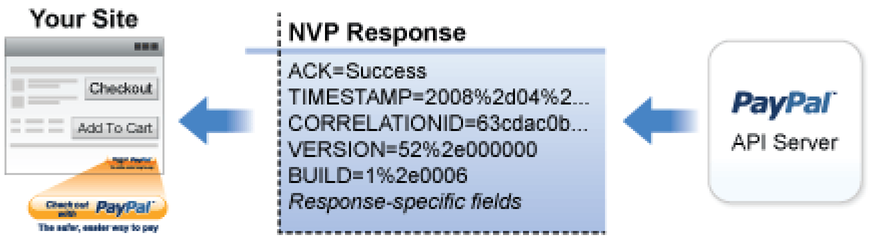
You should log basic information about each PayPal API operation you execute.

All responses to PayPal API operations contain information that may be useful for debugging purposes. You should log the Correlation ID, which identifies the API operation to PayPal, and response-specific information, such as the transaction ID, which you can use to review a transaction on the PayPal website or through the API. You can log other information that may be useful, such as the timestamp. You could implement a scheme that logs the entire request and response in a "verbose" mode; however, you should never log the password from a request.

## Responding to an NVP Response

The Name-Value Pair response consists of the answer to the request as well as common fields that identify the API operation and how it was executed.

The following diagram shows fields in the response to a PayPal NVP API operation:



```
ACK=Success
TIMESTAMP=2008%2d04%2...
CORRELATIONID=63cdac0b...
VERSION=52%2e000000
BUILD=1%2e0006
Response-specific fields
```

### Common Response Fields

The PayPal API always returns common fields in addition to fields that are specific to the requested PayPal API operation.

A PayPal API response includes the following fields:

| Field | Description |
|---|---|
| `ACK` | Acknowledgement status, which is one of the following values:<br>• `Success` indicates a successful operation.<br>• `SuccessWithWarning` indicates a successful operation; however, there are messages returned in the response that you should examine.<br>• `Failure` indicates the operation failed; the response also contains one or more error messages explaining the failure.<br>• `FailureWithWarning` indicates that the operation failed and that there are messages returned in the response that you should examine |
| `CORRELATIONID` | Correlation ID, which uniquely identifies the transaction to PayPal |

| Field | Description |
|---|---|
| `TIMESTAMP` | The date and time that the requested API operation was performed |
| `VERSION` | The version of the API |
| `BUILD` | The sub-version of the API |

## URL Decoding

All responses to HTTP POST operations used by the PayPal NVP API must be decoded.

The PayPal NVP API uses the HTTP protocol to send requests and receive responses from a PayPal API server. You must decode all data returned using the HTTP protocol so that it can be displayed properly. Most programming languages provide a way to decode strings.

**NOTE:** Most browsers decode responses to HTTP requests automatically.