

Payeezy.JS Integration Guide

April 2015

Contents

Quickly integrate your applications with Payeezy JS	3
Minimum technical requirements	3
Payeezy.js initialization	3
Downloadable information	8
Submitting/Generating transaction (method of payments) with example	8
Additional capabilities API	8

Quickly integrate your applications with Payeezy JS

If you want to enable secure and convenient payments in your payment applications, this guide will get you up and running quickly. Payeezy handles all the heavy lifting of the complex tokenization and protects your customers' transactions. It is simple to create a developer test account and apply for a merchant account through our developer portal.

PayeezySM, from First Data, is part of the Small Business Solutions Suite, which includes CloverTM, InsighticsSM, PerkaTM and TransArmor[®]. The Payeezy eCommerce Solution empowers SMBs to expand their horizons and easily grow their business online or via mobile by reaching new customers no matter where they are.

This document refers to the Payeezy.JS integration method included within the overall Payeezy eCommerce Solution. Henceforth, all references to Payeezy are in relation to Payeezy APIs.

Minimum technical requirements

1. TransArmor must be enabled on your merchant account to do token based transactions. Contact your merchant account representative or call 1-855-448-3493 for more details on how to enable this.
2. Compatible browsers:

Browser	Version
Firefox	v33.0
Chrome	v39.x
Internet Explorer	IE6+
Safari	v5.1.7

3. Sign up for a test account and begin the integration process!

Payeezy.js initialization

Step by step integration

1. Look at the prerequisites link to get an overview of our developer portal. Link for prerequisites https://github.com/payeezy/get_started_with_payeezy/raw/master/get_started_with_payeezy042015.pdf
2. Developer portal setup:
Provision your developer account on developer.payeezy.com for Payeezy.JS. On the developer portal, select the API you want to entitle, and add Entitlements using the Entitlements Tab

developer.payeezy.com webpage: 1

My APIs

Add more, edit or delete them as you like.

[+ ADD A NEW API](#)

new salesfast

SANDBOX

KEYS

PRODUCTS

ENTITLEMENTS

DELETE "SALESFAST"

EDIT "SALESFAST"

ANALYTICS

Add more, edit or delete them as you like. [+ Add Entitlements](#)

developer.payeezy.com webpage: 2

Edit API Entitlements

Select Merchant *

ACME SOCK (SANDBOX)

Acme Sock (sandbox)

Merchant Identifier

e2eace387097bd24

Referer (Optional)

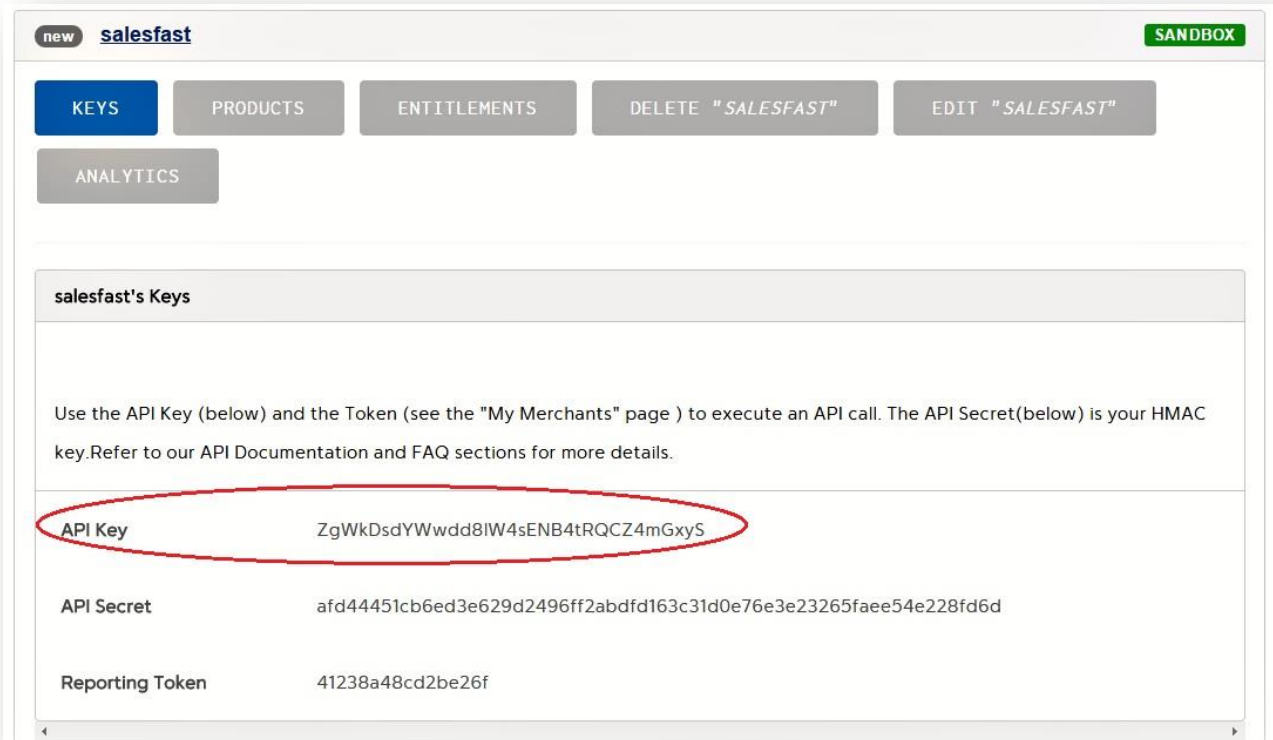
Supported Tokens *

☒ payeezy

[SAVE ENTITLEMENTS](#)

- Once you've picked the merchant to enable, we will create a unique merchant identifier for you. You will need the API Key and the merchant identifier (highlighted below) to initialize Payeezy.js.

developer.payeezy.com webpage: 3



new salesfast SANDBOX

KEYS PRODUCTS ENTITLEMENTS DELETE "SALESFAST" EDIT "SALESFAST"

ANALYTICS

salesfast's Keys

Use the API Key (below) and the Token (see the "My Merchants" page) to execute an API call. The API Secret(below) is your HMAC key.Refer to our API Documentation and FAQ sections for more details.

API Key	ZgWkDsdYWdd8IW4sENB4tRQCZ4mGxyS
API Secret	afd44451cb6ed3e629d2496ff2abdfd163c31d0e76e3e23265faee54e228fd6d
Reporting Token	41238a48cd2be26f

4. Initializing the Payeezy.JS :

Payeezy JS implementation can be initialized by placing the `<script>` tag inside the `<head>` tag of the Payment page. For security reasons, link directly to payeezy.js hosted on Payeezy over https as shown below; do not make a local copy of the js file.

```
<script src="https://developer.payeezy.com/v1/payeezy.js" type="text/javascript"></script>
```

5. Initialize Payeezy.js by the apiKey and the Merchant Identifier inside the `<script>` tag as shown below. For every merchant, api key and merchant identifier will be unique and the same can be retrieved from the developer portal (sample values shown below)

```
<script>
Payeezy.setApiKey('eHcvPV9zheE3wZAvE7a9FrIb6KtcBliq');
Payeezy.setMerchantIdentifier('5edf46316f6b');
</script>
```

6. Payment Information Set Up:

- a. Payeezy.js has to be initialized for setting up the Payment information page. All Personally Identifiable Information (PII) fields on the form should have an attribute “payeezy-data”. It is important that these fields do not have the “name” attribute. By not having the “name” attribute, the browser does not send that information back to the merchant server. Please look at the highlighted attribute names below that are required to generate the payment token. All the fields that are not PII will be submitted to the Merchant server.

```
<form action="{call to merchant server}" method="post" id="payment -info-form" >
<span class="payment-errors"></span>
<div class="row">
<label>Card Type : </label>
<select payeezy-data="card_type">
<option value="visa">Visa</option>
```

```
<option value="mastercard">Master Card</option>
<option value="American Express" >American Express</option>
<option value="discover" >Discover</option>
</select>
</div>
<div class="row">
<label>Cardholder Name : </label>
<input type="text" payeezy-data="cardholder_name" value="Tom Eck"/>
</div>
<div class="row">
<label>Card Number : </label>
<input type="text" payeezy-data="cc_number" value="4788250000028291"/>
</div>
<div class="row">
<label>CVV Code : </label>
<input type="text" payeezy-data="cvv_code" value="123"/>
</div>
<div class="row">
<label>Expiry Date : </label>
<select payeezy-data="exp_month">
<option value="01">01</option>
<option value="02">02</option>
<option value="03">03</option>
<option value="04">04</option>
<option value="05">05</option>
<option value="06">06</option>
<option value="07">07</option>
<option value="08">08</option>
<option value="09">09</option>
<option value="10">10</option>
<option value="11">11</option>
<option value="12" selected>12</option>
</select>
<select payeezy-data="exp_year">
<option value="14" selected>2014</option>
<option value="15">2015</option>
<option value="16">2016</option>
<option value="17">2017</option>
<option value="18">2018</option>
</select>
</div>
```

(Code continues....)

```
<div class="row">
<label>Merchant Ref : </label>
<input type="text" name="merchant_ref" id="merchant_ref" value="Token Testing"/>
</div>
<div class="row">
<label>Transaction Type : </label>
<select id="transaction_type" name="transaction_type" >
<option value="purchase" >Purchase</option>
<option value="authorize" selected="selected" >Authorize</option>
</select>
</div>
<div class="row">
<label>Amount : </label>
<input type="text" name="amount" id="amount" value="101"/>
</div>
<button type="submit">Make Payment</button>
</form>
```

7. **Creating the Payment Token:** Below code snippet can be used to capture the submit event and then create the payment token using the Payeezy.JS implementation. It's not required to use jQuery, the same functionality can be accomplished with native javascript.

```
jQuery(function($) {
$('#payment-info-form').submit(function(e) {
var $form = $(this);
$form.find('button').prop('disabled', true);
Payeezy.createToken(responseHandler);
return false;
});
});
```

“responseHandler” is a merchant defined JS function, which will be called when the creation of payment token is successful. Example of this function is provided in the next section.

8. Sending the Payment Information to Merchant Server:

“responseHandler” method take two arguments status and response(a javascript object). If the payment token creation is successful, then a status code of “201” is returned. Anything other than a “201”, the error message can be

```
var responseHandler = function(status, response)
{
    var $form = $('#payment-info-form');
    if (status != 201) {
        if (response.error) {
            var errorMessages = response.error.messages;
            var allErrors = '';
            for (i=0; i<errorMessages.length;i++) {
                allErrors = allErrors + errorMessages[i].description;
            }
            $form.find('.payment-errors').text(allErrors);
        }
        $form.find('button').prop('disabled', false);
    } else {
        var token = response.token.value;
        $form.append($('
```

got from the response object. Highlighted code shows how to get the error and the token value(if successful).

The token value will be put in a hidden field (payeezyToken in this case), and then the form will be submitted to the Merchant server for making the payment with the “Token” instead of credit card information.

9. Make a Payment (server side) using the payment token:

Please refer to the Developer portal documentation link <https://developer.payeezy.com/payeezy-api-reference/apis> on how to make a payment ("authorize" or "purchase") with the token.

10. Deploy the Payeezy.js and dependency java script files in the specified folder structure in your website JavaScript deployment folder.

Downloadable information

The sample code for payeezy.js can be found on GitHub. Following is the GitHub link for the sample code.

Click on GitHub link <https://github.com/payeezy/payeezy.js> and look for JavaScript folder to download Payeezy.js file.

Submitting/Generating transaction (method of payments) with example

Payeezy supports the following method of payments

- Credit Card Payments
- PayPal Transactions
- Gift Card (via ValueLink) Transactions
- eCheck (via TeleCheck) Transactions
- 3D Secure Transactions

For API processing details and examples, click here [Method of Payments](#)

Or click on the link https://developer-qa.payeezy.com/payeezy_ref_docs/apis

Additional capabilities API

- Partner Reporting API - Use our powerful query engine to retrieve payment records. Supports complex filtering, sorting, pagination and more. This is exclusively for Third Party Partners and applicable for a **live** environment only.

For Reporting API processing details, click here https://developer-qa.payeezy.com/payeezy_ref_docs/apis/get/transactions-0