Chargify SDK for PHP Documentation

Release 0.1.0

Crucial Web Studio

Sep 19, 2017

Contents

Chargify SDK for PHP is a PHP package that makes it easy to interact with the Chargify API. It has been used in production for years on your flagship product, Chargely, a hosted billing portal for Chargify.

- Abstracts away underlying HTTP requests to the Chargify API
- Supports Chargify API v1 and Chargify Direct (v2)
- Well documented
- Unit tested

```php
<?php
require 'vendor/autoload.php';

use Crucial\Service\Chargify;

$chargify = new Chargify([
    'hostname'   => 'yoursubdomain.chargify.com',
    'api_key'    => '{{API_KEY}}',
    'shared_key' => '{{SHARED_KEY}}'
]);

// Crucial\Service\Chargify\Customer
$customer = $chargify->customer()
    // set customer properties
    ->setFirstName('Dan')
    ->setLastName('Bowen')
    ->setEmail('dan@mailinator.com')
    // send the create request
    ->create();

// check for errors
if ($customer->isError()) {
    // array of errors loaded during the transfer
    $errors = $customer->getErrors();
 } else {
    // the transfer was successful
    $customerId = $customer['id']; // Chargify customer ID
    $firstName  = $customer['first_name'];
    $lastName   = $customer['last_name'];
    $email      = $customer['email'];
 }
```

CHAPTER 1

User Guide

Overview

Requirements

1. PHP >= 5.5.0

Installation

Using Composer is the recommended way to install the Chargify SDK for PHP. Composer is a dependency management tool for PHP that allows you to declare the dependencies your project needs and installs them into your project. In order to use the SDK with Composer, you must do the following:

Download and install Composer, if you don't already have it.

```
curl -sS https://getcomposer.org/installer | php
```

Add `chargely/chargify-sdk-php` as a dependency in your project's composer.json file.

```
php composer.phar require chargely/chargify-sdk-php:~0.1
```

Alternatively, you can specify Chargify SDK for PHP as a dependency in your project's existing composer.json file:

```
{
  "require": {
    "chargely/chargify-sdk-php": "~0.1"
  }
}
```

Install your dependencies, including your newly added Chargify SDK for PHP.

```
php composer.phar install
```

After installing, you need to require Composer's autoloader:

```
require 'vendor/autoload.php';
```

You can find out more on how to install Composer, configure autoloading, and other best-practices for defining dependencies at getcomposer.org.

Bleeding edge

During your development, you can keep up with the latest changes on the master branch by setting the version requirement for Chargify SDK for PHP to `~0.1@dev`.

```
{
   "require": {
      "chargely/chargify-sdk-php": "~0.1@dev"
   }
}
```

License

Licensed using the Apache 2.0 license.

> Copyright (c) 2016 Crucial Web Studio, LLC
>
> Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at
>
> http://www.apache.org/licenses/LICENSE-2.0
>
> Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Contributing

We work hard to provide a high-quality and useful SDK for Chargify services, and we greatly value feedback and contributions from our community. Please submit your issues or pull requests through GitHub.

Guidelines

1. The SDK is released under the Apache 2.0 license. Any code you submit will be released under that license. For substantial contributions, we may ask you to sign a Contributor License Agreement (CLA).
2. The SDK has a minimum PHP version requirement of PHP 5.5. Pull requests must not require a PHP version greater than PHP 5.5 unless the feature is only utilized conditionally.
3. We follow all of the relevant PSR recommendations from the PHP Framework Interop Group. Please submit code that follows these standards. The PHP CS Fixer tool can be helpful for formatting your code.
4. We maintain a high percentage of code coverage in our unit tests. If you make changes to the code, please add, update, and/or remove tests as appropriate.

5. If your code does not conform to the PSR standards or does not include adequate tests, we may ask you to update your pull requests before we accept them. We also reserve the right to deny any pull requests that do not align with our standards or goals.
6. If you would like to implement support for a significant feature that is not yet available in the SDK, please talk to us beforehand to avoid any duplication of effort.

In order to contribute, you'll need to checkout the source from GitHub and install Chargify SDK fpr PHP's dependencies using Composer:

```
git clone https://github.com/chargely/chargify-sdk-php.git
cd chargify-sdk-php && curl -s http://getcomposer.org/installer | php && ./composer.
↪phar install --dev
```

Running the tests

The SDK is unit tested with PHPUnit. Run the tests using the following commands:

```
cd tests/phpunit
../../vendor/bin/phpunit

# with coverage report
../../vendor/bin/phpunit --coverage-html artifacts/coverage
```

API v1 Quickstart

This page provides a quick introduction to Chargify SDK for PHP and introductory examples for v1 of the Chargify API. If you have not already installed the SDK, head over to the *Installation* page.

Making a Request

The first step to sending a request with the SDK is to create a `Crucial\Service\Chargify` object.

Creating an API Client

```
<?php
use Crucial\Service\Chargify;

$chargify = new Chargify([
    'hostname'   => 'yoursubdomain.chargify.com',
    'api_key'    => '{{API_KEY}}',
    'shared_key' => '{{SHARED_KEY}}'
]);
```

The client constructor accepts an associative array of options:

`hostname` (string) The hostname of the Chargify site you want to interact with.

`api_key` (string) API key from your Chargify site.

`shared_key` (string) Shared key from your Chargify site.

See the Chargify documentation for help finding your API v1 credentials.

Resource Objects

The Chargify API v1 is divided into resources such as subscriptions, customers, products, etc. The SDK follows this same pattern and provides a simple interface for selecting the resource you want to operate on.

```php
<?php
use Crucial\Service\Chargify;

$chargify = new Chargify([
    'hostname'   => 'yoursubdomain.chargify.com',
    'api_key'    => '{{API_KEY}}',
    'shared_key' => '{{SHARED_KEY}}'
]);

// Crucial\Service\Chargify\Customer
$customer = $chargify->customer();

// Crucial\Service\Chargify\Subscription
$subscription = $chargify->subscription();

// Crucial\Service\Chargify\Product
$product = $chargify->product();

// Crucial\Service\Chargify\Adjustment
$adjustment = $chargify->adjustment();

// Crucial\Service\Chargify\Charge
$charge = $chargify->charge();

// Crucial\Service\Chargify\Component
$component = $chargify->component();

// Crucial\Service\Chargify\Coupon
$coupon = $chargify->coupon();

// Crucial\Service\Chargify\Transaction
$transaction = $chargify->transaction();

// Crucial\Service\Chargify\Refund
$refund = $chargify->refund();

// Crucial\Service\Chargify\Statement
$statement = $chargify->statement();

// Crucial\Service\Chargify\Event
$event = $chargify->event();

// Crucial\Service\Chargify\Webhook
$webhook = $chargify->webhook();

// Crucial\Service\Chargify\Stats
$stats = $chargify->stats();
```

Sending Requests

Now that you have a resource object you're ready to send a request. The resource objects provide a fluent interface for setting properties to send in your request. The example below shows how to create a new customer.

```php
<?php
use Crucial\Service\Chargify;

$chargify = new Chargify([
    'hostname'   => 'yoursubdomain.chargify.com',
    'api_key'    => '{{API_KEY}}',
    'shared_key' => '{{SHARED_KEY}}'
]);

// Crucial\Service\Chargify\Customer
$customer = $chargify->customer()
    // set customer properties
    ->setFirstName('Dan')
    ->setLastName('Bowen')
    ->setEmail('dan@mailinator.com')
    // send the create request
    ->create();
```

Using Responses

In the previous example, calling `->create()` on the `Crucial\Service\Chargify\Customer` object will send the request to Chargify and load it with the newly created customer response.

You can access the response data as you would any normal array:

```
$customerId = $customer['id']; // Chargify customer ID
$firstName  = $customer['first_name'];
$lastName   = $customer['last_name'];
$email      = $customer['email'];
```

Error Handling

The SDK loads errors on the resource object for any errors that occur during a transfer.

- You can test for an error using the `->isError()` method of the resource object.

```
if ($customer->isError()) {
   // handle errors
} else {
   // the transfer was successful
   $customerId = $customer['id']; // Chargify customer ID
   $firstName  = $customer['first_name'];
   $lastName   = $customer['last_name'];
   $email      = $customer['email'];
}
```

- You can get the loaded errors, if any, using the `->getErrors()` method of the resource object.

```
if ($customer->isError()) {
   // array of errors loaded during the transfer
   $errors = $customer->getErrors();
}
```

API v1 Resources

Adjustments

coming soon

Charges

coming soon

Components

coming soon

Coupons

coming soon

Customers

coming soon

Events

coming soon

Products

coming soon

Refunds

coming soon

Statements

coming soon

Stats

coming soon

Subscriptions

The `Crucial\Service\Chargify\Subscription` object provides the following methods.

Chargify API documentation for subscriptions

Subscription Output Attributes

Many of the examples below will result in the following standard attributes on the `$subscription` object.

```
{
     "id":[@subscription.id],
     "state":"active",
     "previous_state":`auto generated`,
     "balance_in_cents":0,
     "total_revenue_in_cents":1000,
     "product_price_in_cents": 1000,
     "product_version_number": 1,
     "current_period_started_at":`auto generated`,
     "current_period_ends_at":`auto generated`,
     "next_assessment_at":`auto generated`,
```

```
    "activated_at":`auto generated`,
    "trial_ended_at":`auto generated`,
    "trial_started_at":`auto generated`,
    "expires_at":`auto generated`,
    "created_at":`auto generated`,
    "updated_at":`auto generated`,
    "cancellation_message":null,
    "canceled_at":`your value`,
    "cancel_at_end_of_period":false,
    "delayed_cancel_at":null,
    "coupon_code":`your value`,
    "signup_payment_id":`auto generated`,
    "signup_revenue":`your value`,
    "payment_collection_method":"automatic",
    "current_billing_amount_in_cents": "1000",
    "customer":{
      "id":`auto generated`,
      "first_name":`your value`,
      "last_name":`your value`,
      "email":`your value`,
      "organization":`your value`,
      "reference":`your value`,
      "address":`your value`,
      "address_2":`your value`,
      "city":`your value`,
      "state":`auto generated`,
      "zip":`auto generated`,
      "country":`auto generated`,
      "phone":`auto generated`,
      "updated_at":`auto generated`,
      "created_at":`auto generated`
    },
    "product":{
      "id":`auto generated`,
      "name":`your value`,
      "handle":`your value`,
      "description":`your value`,
      "price_in_cents":`your value`,
      "accounting_code":`your value`,
      "interval":`your value`,
      "interval_unit":`your value`,
      "initial_charge_in_cents":null,
      "trial_price_in_cents":null,
      "trial_interval":null,
      "trial_interval_unit":null,
      "expiration_interval_unit":null,
      "expiration_interval":null,
      "return_url":null,
      "update_return_url":null,
      "return_params":null,
      "require_credit_card":true,
      "request_credit_card":true,
      "created_at":`auto generated`,
      "updated_at":`auto generated`,
      "archived_at":null,
      "product_family":{
        "id":`auto generated`,
        "name":`your value`,
```

```
        "handle":`your value`,
        "accounting_code":`your value`,
        "description":`your value`
      }
    },
    "credit_card":{
      "id":`auto generated`,
      "first_name":`your value`,
      "last_name":`your value`,
      "masked_card_number":`your value`,
      "card_type":`auto generated`,
      "expiration_month":`your value`,
      "expiration_year":`your value`,
      "billing_address":`your value`,
      "billing_address_2":`your value`,
      "billing_city":`your value`,
      "billing_state":`your value`,
      "billing_zip":`your value`,
      "billing_country":`your value`,
      "current_vault":`your value`,
      "vault_token":`your value`,
      "customer_vault_token":`your value`,
      "customer_id":`auto generated`
    }
  }
```

Create

Create a new subscription in Chargify.

```
$subscription = $chargify->subscription()
    // product ID being signed up for
    ->setProductId(123)

    // alternatively, set the product by handle
    //->setProductHandle('my-product-handle')

    // customer attributes
    ->setCustomerAttributes([
        'first_name'   => '{{FIRST_NAME}}',
        'last_name'    => '{{LAST_NAME}}',
        'email'        => '{{EMAIL}}',
        'organization' => '{{ORGANIZATION}}',
        'phone'        => '{{PHONE}}',
        'address'      => '{{ADDRESS}}',
        'address_2'    => '{{ADDRESS_2}}',
        'city'         => '{{CITY}}',
        'state'        => '{{STATE}}',
        'zip'          => '{{ZIP}}',
        'country'      => '{{COUNTRY}}',
    ])

    // alternatively, set customer ID or reference if the new subscription is for an␣
↪existing customer
    //->setCustomerId(1234)
    //->setCustomerReference('customer-reference')
```

```
    // payment profile attributes
    ->setPaymentProfileAttributes([
        'first_name'       => '{{FIRST_NAME}}',
        'last_name'        => '{{LAST_NAME}}',
        'full_number'      => '{{CC_NUMBER}}',
        'expiration_month' => '{{EXPIRY_MONTH}}',
        'expiration_year'  => '{{EXPIRY_YEAR}}',
        'cvv'              => '{{CVV}}',
        'billing_address'  => '{{ADDRESS}}',
        'billing_city'     => '{{CITY}}',
        'billing_state'    => '{{STATE}}',
        'billing_zip'      => '{{ZIP}}',
        'billing_country'  => '{{COUNTRY}}'
    ])

    // (Optional, used for Subscription Import)
    //->setNextBillingAt('8/6/2010 11:34:00 EDT')

    // send the request
    ->create();
```

Create: Output Attributes

Standard subscription output attributes.

Read

Read a single existing subscription.

```
$subscription = $chargify->subscription()
     ->read($existingSubscriptionId);
```

Read: Output Attributes

Standard subscription output attributes.

List

List all subscriptions for the Chargify site you are working with.

Listing subscriptions is paginated, 2000 at a time, by default. They are listed most recently created first. You may control pagination using the `->setPage()` and `->setPerPage()` methods.

```
$subscription = $chargify->subscription()
     ->setPage(1)
     ->setPerPage(100)
     ->listSubscriptions();
```

List: Output Attributes

A zero-indexed array of subscriptions, each with the standard subscription output attributes.

List By Customer

List all subscriptions for a given customer.

```
$subscription = $chargify->subscription()
     // list subscriptions for customer ID 1234
     ->listByCustomer(1234);
```

List By Customer: Output Attributes

A zero-indexed array of subscriptions, each with the standard subscription output attributes.

Update

Update a subscription's product, customer attributes, or payment profile attributes.

```
$subscription = $chargify->subscription()
    // changing the product on an existing subscription will result in a non-prorated␣
↪migration.
    ->setProductId(123)

    // alternatively, set the product by handle
    //->setProductHandle('my-product-handle')

    // new customer attributes
    ->setCustomerAttributes([
        'first_name'   => '{{FIRST_NAME}}',
        'last_name'    => '{{LAST_NAME}}',
        'email'        => '{{EMAIL}}',
        'organization' => '{{ORGANIZATION}}',
        'phone'        => '{{PHONE}}',
        'address'      => '{{ADDRESS}}',
        'address_2'    => '{{ADDRESS_2}}',
        'city'         => '{{CITY}}',
        'state'        => '{{STATE}}',
        'zip'          => '{{ZIP}}',
        'country'      => '{{COUNTRY}}',
    ])

    // new payment profile attributes
    ->setPaymentProfileAttributes([
        'first_name'       => '{{FIRST_NAME}}',
        'last_name'        => '{{LAST_NAME}}',
        'full_number'      => '{{CC_NUMBER}}',
        'expiration_month' => '{{EXPIRY_MONTH}}',
        'expiration_year'  => '{{EXPIRY_YEAR}}',
        'cvv'              => '{{CVV}}',
        'billing_address'  => '{{ADDRESS}}',
        'billing_city'     => '{{CITY}}',
        'billing_state'    => '{{STATE}}',
        'billing_zip'      => '{{ZIP}}',
        'billing_country'  => '{{COUNTRY}}'
    ])

    // send the request
    ->create();
```

Update: Output Attributes

Standard subscription output attributes.

Migrate

Perform a prorated migration on a subscription. See Chargify Documentation for more details.

```
$subscription = $chargify->subscription()
     // set new product ID
     ->setProductId(1234)

     // alternatively, set new product by handle
     //->setProductHandle('product-handle')

     // (optional) Include trial in migration. 1 for yes, 0 for no. default: 0
     //->setIncludeTrial(1)

     // (optional) Include initial charge in migration. 1 for yes, 0 for no. default:␣
↪0
     //->setIncludeInitialCharge(1)

     // send the migration request
     ->migrate();
```

Migrate: Output Attributes

Standard subscription output attributes.

Cancel Immediately

Cancel a subscription immediately in Chargify.

```
$subscription = $chargify->subscription()

     // (optional) Set cancellation message.
     //->setCancellationMessage('No longer using the service')

     // cancel subscription ID 1234 immediately
     ->cancelImmediately(1234);
```

Cancel Immediately: Output Attributes

Standard subscription output attributes.

Cancel Delayed

Cancel a subscription at the end of the current billing period.

```
$subscription = $chargify->subscription()

     // (optional) Set cancellation message.
     //->setCancellationMessage('No longer using the service')

     // cancel subscription ID 1234 immediately
     ->cancelDelayed(1234);
```

Cancel Delayed: Output Attributes

Standard subscription output attributes.

Reactivate

Reactivate an inactive subscription.

```
$subscription = $chargify->subscription()

     // (optional) Include trial period (if any) when the subscription is re-activated
     //->setIncludeTrial(true)

     // reactivate subscription ID 1234
     ->reactivate(1234);
```

Reactivate: Output Attributes

Standard subscription output attributes.

Reset Balance

Reset the balance of a subscription to zero.

```
$subscription = $chargify->subscription()
     // reset balance to zero on subscription ID 1234
     ->resetBalance(1234);
```

Reset Balance: Output Attributes

Standard subscription output attributes.

Transactions

coming soon

Webhooks

coming soon

API v2 Quickstart

This page provides a quick introduction to Chargify SDK for PHP and introductory examples for v2 of the Chargify API. If you have not already installed the SDK, head over to the *Installation* page.

Making a Request

The first step to sending a request with the SDK is to create a `Crucial\Service\ChargifyV2` object.

Creating an API Client

```php
<?php
use Crucial\Service\ChargifyV2;

$chargifyV2 = new ChargifyV2([
    'api_id'       => '{{API_ID}}',
    'api_password' => '{{API_PASSWORD}}',
    'api_secret'   => '{{API_SECRET}}'
]);
```

The client constructor accepts an associative array of options:

api_id (string) Chargify Direct API ID.

api_password (string) Chargify Direct API password.

api_secret (string) Chargify Direct API secret.

See the Chargify documentation for help finding your API v2 credentials.

Resource Objects

The Chargify API v2 is divided into resources such as call, signups, card update, etc. The SDK follows this same pattern and provides a simple interface for selecting the resource you want to operate on.

```php
<?php
use Crucial\Service\ChargifyV2;

$chargifyV2 = new ChargifyV2([
    'api_id'       => '{{API_ID}}',
    'api_password' => '{{API_PASSWORD}}',
    'api_secret'   => '{{API_SECRET}}'
]);

// Crucial\Service\ChargifyV2\Direct
$direct = $chargifyV2->direct();

// Crucial\Service\ChargifyV2\Call
$call = $chargifyV2->call();
```

Sending Requests

Now that you have a resource object you're ready to send a request. The resource objects provide a fluent interface for setting properties to send in your request. The example below shows how to fetch a call object.

```php
<?php
use Crucial\Service\ChargifyV2;

$chargifyV2 = new ChargifyV2([
    'api_id'       => '{{API_ID}}',
    'api_password' => '{{API_PASSWORD}}',
    'api_secret'   => '{{API_SECRET}}'
]);

// Crucial\Service\ChargifyV2\Call
$call = $chargifyV2->call()
    // send request to fetch call object by ID
    ->readByChargifyId(1234);
```

Using Responses

In the previous example, calling `->readByChargifyId()` on the `Crucial\Service\ChargifyV2\Call` object will send the request to Chargify and load it with the call response.

You can access the response data as you would any normal array:

```
$callId   = $call['id']; // Chargify call ID
$request  = $call['request'];
$response = $call['response'];
$success  = $call['success'];
```

Error Handling

The SDK loads errors on the resource object for any errors that occur during a transfer.

- You can test for an error using the `->isError()` method of the resource object.

```
if ($call->isError()) {
   // handle errors
} else {
   // the transfer was successful
   $callId   = $call['id']; // Chargify call ID
   $request  = $call['request'];
   $response = $call['response'];
   $success  = $call['success'];
}
```

- You can get the loaded errors, if any, using the `->getErrors()` method of the resource object.

```
if ($call->isError()) {
   // array of errors loaded during the transfer
   $errors = $call->getErrors();
}
```

Note: In the above example, `->isError()` and `->getErrors()` only loads errors with the SDK's request to the Chargify API. It does not load errors from the call object itself, such as might be present in `$call['response']['result']['errors']`.

API v2 Resources

Calls

coming soon

Chargify Direct

The `Crucial\Service\ChargifyV2\Direct` object provides the following methods.

Chargify Direct documentation

Signup

Create a Chargify Direct signup form. See the Chargify Documentation for more information.

```
$direct = $chargifyV2->direct();

// set redirect
$direct->setRedirect('https://example.local');

// set tamper-proof data
$direct->setData([
    'signup' => [
        'product'  => [
            'id' => 1234
    ],
    'customer' => [
        'first_name' => 'Dan',
        'last_name'  => 'Bowen',
        'email'      => 'foo@mailinator.com'
    ]
]]);
```

Now use `$direct` object to help create your HTML signup form.

```
<form accept-charset="utf-8" method="post" action="<?php echo $direct->
↪getSignupAction() ?>">
     <?php echo $direct->getHiddenFields(); ?>

     <!-- the rest of your form goes here -->

</form>
```

Card Update

Create a Chargify Direct card update form. See the Chargify Documentation for more information.

```
$direct = $chargifyV2->direct();

// set redirect
$direct->setRedirect('https://example.local');

// set tamper-proof data
$direct->setData([
    'subscription_id' => 1234
]);
```

Now use `$direct` object to help create your HTML card update form.

```
<!-- card update form for subscription ID 1234 -->
<form accept-charset="utf-8" method="post" action="<?php echo $direct->
↪getCardUpdateAction(1234) ?>">
    <?php echo $direct->getHiddenFields(); ?>

    <!-- the rest of your form goes here -->

</form>
```

Validate Response Signature

The `->isValidResponseSignature()` method will test for a valid redirect to your site after a Chargify Direct signup request.

```
$isValidResponse = $chargifyV2->direct()
    ->isValidResponseSignature($_GET['api_id'], $_GET['timestamp'], $_GET['nonce'],
↪$_GET['status_code'], $_GET['result_code'], $_GET['call_id']);

// $isValidResponse will be a boolean true\false.
if ($isValidResponse) {
    // Do post-signup (or post card-update) tasks such as updating your database,␣
↪sending a thank you email, etc.
} else {
    // display an error message
}
```