Kalpay Gateway Doc

Version 0.1

Tobin Frost

mars 31, 2022

Table des matières

SuperSDK est une librairie javascript qui permet d'intégrer la passerelle de paiement Kalpay dans votre site e-commerce afin de proposer des moyens de paiement.

Cette librairie est flexible et offre une grande capacité de customisation.

Grâce à **SuperSDK** de Kalpay vous pourrez définir la charte graphique de votre passerelle de paiement de sorte à s'accorder à votre site, choisir vos moyens de paiement acceptés.

Parmi les fonctionnalités nous avons
- Customisation graphique
  - Couleur (bouton, label)
  - Logo d'entreprise
  - Item d'achat
- Définition des moyens de paiement
- Paramètrage de navigation (redirections après transaction)
- Facturation automatique (inclue)

Consultez la section *Usage* pour plus d'informations, y compris comment *Installation* le sdk.

Note : This project is under active development.

CHAPITRE 1

Table des matières

1.1 Usage

1.1.1 Installation

Pour utiliser le *SuperSDK* la première chose à faire est de l'ajouter à votre page html.

Pour ajouter un script ou bibliothèque javascript tier, il y'a deux école :

— La manière synchrone

```
<!doctype html>
<html>
<head>
   <meta charset="utf-8">
   <title>KALPAY-Demo</title>
</head>
<body>
   <script src="kalpay-gateway-sdk-X.X.X.js"></script>
   <script>

   // Your code goes here.

   </script>
</body>
</html>
```

— La manière asynchrone

```
<!doctype html>
<html>
<head>
   <meta charset="utf-8">
```

(suite sur la page suivante)

(suite de la page précédente)

```
   <title>KALPAY-Demo</title>
</head>
<body>
   <script>
   (function(){
        window.superSDK_ready = window.superSDK_ready || [];
        var script = document.createElement('script');
        script.src = 'kalpay-gateway-sdk-X.X.X.js';
        script.async = true;
        var entry = document.getElementsByTagName('script')[0];
        entry.parentNode.insertBefore(script, entry);
    })();

    window.superSDK_Async = function() {

      // Your code goes here.

    }
   </script>
</body>
</html>
```

Note : Nous recommandons la méthode *asynchrone* et d'inserer le code à la fin de votre code html pour une meilleure expérience utilisateur

1.1.2 Commencer

1. Initialisation

Pour commencer l'expérience Kalpay, il faudra *obligatoirement* activer la passerelle grâce à la fonction `superSDK.init()`.

la fonction prend _deux_ paramètres : un `identifiant` et une `fonction de callback`.

le paramètre `identifiant` doit être un objet du type :

, le paramètre `fonction de callback` sera du type :

```
function(response){
   // Your code goes here.
}
```

L'initialisation de la gateway dans une page html ressemblera à ceci :

```
superSDK.init({accessKey: [ACCESS_KEY], secretKey: [SECRET_KEY]},function(response){});
```

Avertissement : le `ACCESS_KEY` et `SECRET_KEY` sont disponibles qu'après souscription à un compte **Kalpay Business**

2. Paramètrage (Customisation)

Après *initialisation* ici initialisation, la seconde étape est de paramètrer votre passerelle.

Le paramètrage se fait grâce à la fonction `superSDK.setup()`.

la fonction attend un `objet de customisation`.

le paramètre `objet de customisation` ressemble à ceci :

```
{
   store_name : [PAGE_NAME],
   triggerer: "render_zone",
   store_logo_url : [LOGO_URL],
   frame_color: [CUSTOM_COLOR]
}
```

Le paramètrage de la gateway ressemblera à ceci :

```
superSDK.setup(
   {
      store_name : [PAGE_NAME],
      triggerer: "render_zone",
      store_logo_url : [LOGO_URL],
      frame_color: [CUSTOM_COLOR]
   }
);
```

Note : `LOGO_URL` n'est pas obligatoire et doit être un lien externe (http://lien_vers_image) `CUSTOM_COLOR` est un code hexadecimal (#03fc7b)

1.1.3 Exemple

Voici un exemple de code *javascript* intégrant la Gateway de Kalpay

```
<script>
    (function(){
        window.superSDK_ready = window.superSDK_ready || [];
        var script = document.createElement('script');
        script.src = 'kalpay-gateway-sdk-1.0.4.js';
        script.async = true;
        var entry = document.getElementsByTagName('script')[0];
        entry.parentNode.insertBefore(script, entry);
    })();

    window.superSDK_Async = function() {

        superSDK.init({accessKey: [ACCESS_KEY], secretKey: [SECRET_KEY]},
↪function(response){});
        superSDK.setup({
            store_name : [PAGE_NAME],
            triggerer: "render_zone",
            store_logo_url : [LOGO_URL],
            frame_color: [CUSTOM_COLOR]
```

(suite sur la page suivante)

(suite de la page précédente)

```
        });
        superSDK.addItem({
            name : [ITEM_NAME],
            quantity: [ITEM_QUANTITY],
            unit_price : [ITEM_PRICE],
            item_photo : [ITEM_PHOTO_URL]
        });
        // define superSDK navigation
        superSDK.setNavigation({
            success_url : [TRANSACTION_SUCCESS_PAGE_URL],
            failed_url : [TRANSACTION_FAILED_PAGE_URL],
            back_url : window.location.href
        })
    }
</script>
```

Pour afficher la gateway, inserer le code suivant dans un code *html*

Note : La méthode `superSDK.proceed()` lance la gateway avec les données précédemments renseignées.

1.2 API

1.2.1 superSDK.init

la méthode `init(credential, callback_fn)` active la gateway grâce aux paramètres d'authentification, vérifie l'identité du compte associé et s'assure de son authenticité.

Paramètres	détails
credential (Object)	objet d'authentification
callback_fn (function)	fonction de callback pour capter à la réponse de l'initialisation de la gateway

credential (obligatoire)

```
{
   accessKey: [ACCESS_KEY],
   secretKey: [SECRET_KEY]
}
```

callback_fn

```
function(response){

}
```

1.2.2 superSDK.setup

la méthode `setup(custom_settings)` permet de définir les éléments de customisation de la gateway.

Paramètre	détails
custom_settings (Object)	objet de customisation

custom_settings (obligatoire)

```
{
   store_name : [PAGE_NAME], // le nom du site ou page, utilisé pour la facturation
   triggerer: "render_zone", // le sélecteur (CSS) vers le bouton ou l'élément déclacheur␣
↪du paiement
   store_logo_url : [LOGO_URL], // le logo du site ou page
   frame_color: [CUSTOM_COLOR] // la couleur du ui (hex) de la gateway : bouton, labels,␣
↪etc..
}
```

1.2.3 superSDK.addItem

la méthode `addItem(item)` permet d'ajouter un article à la liste des articles à afficher dans la gateway pour paiement.

Paramètre	détails
item (Object)	article à facturer

item (obligatoire)

```
{
   name : [ITEM_NAME], // nom de l'article
   quantity: [ITEM_QUANTITY], // quantité de l'article
   unit_price : [ITEM_PRICE], // prix unitaire de l'article
   item_photo : [ITEM_PHOTO_URL] // photo de l'article
}
```

1.2.4 superSDK.setNavigation

la méthode `setNavigation(navigation_settings)` permet de définir la politique de redirections de la gateway à la fin de la transaction. Les redirections se feront `3s` après fin de la transaction.

Paramètre	détails
navigation_settings (Object)	urls à suivre pour chaque scenari de transactions

navigation_settings

```
{
   success_url : [TRANSACTION_SUCCESS_PAGE_URL], // url à laquelle se rediriger au␣
↪succès de la transaction
   failed_url : [TRANSACTION_FAILED_PAGE_URL], // url à laquelle se rediriger à l'échec␣
↪de la transaction
   back_url : window.location.href // url de retour après annulation
}
```

1.2.5 superSDK.setOperators

la méthode `setOperators(operators_list)` permet de définir la liste des opérateurs de paiement acceptés dans la gateway. La liste des opérateurs est composée du code des opérateurs, exemple : « OrangeMoney », »EMoney »

Paramètre	détails
operators_list (Array<String>)	liste des opérateurs de paiement à afficher dans votre gateway

operators_list

```
const operators_list = ["OrangeMoney","EMoney","FreeMoney"]
```