Utilisation du SDK Android

Contenu

Plus d'informations

Impossible de produire un rendu {enfants}. Page introuvable : SDKMOBILE:Guide d'utilisation Android.

Initialisation

Pour l'initialisation du SDK au sein de l'application, il faut tout d'abord instancier un `*PaymentController()*` et un `*WalletController()*` et ensuite les associer à leur listener qui seront décrits par la suite. Habituellement, cela est fait dans le `*onCreate()*` de l'activité :

```
private var paymentController = PaymentController()
paymentController.registerListener(listener, context)

private var walletController = WalletController()
walletController.registerListener(listener, context)
```

La méthode d'initialisation du paiement requiert deux paramètres : un "PaymentControllerListener" et le "Context".

La méthode d'initialisation du portefeuille requiert deux paramètres : un "WalletControllerListener" et le "Context".

Cependant, il faut aussi dissocier le listener lorsque l'activité est détruite. Habituellement, cela est fait dans le `*onDestroy()*` de celle-ci :

```
paymentController.unregisterListener()
walletController.unregisterListener()
```

Pour que votre activité agisse comme un listener, vous devez implémenter les interfaces `*PaymentControllerListener*` et/ou `*WalletControllerListener*` :

```
class MainActivity : AppCompatActivity(), PaymentControllerListener, WalletControllerListener
```

Configuration

La méthode `*showPaymentForm*` est utilisée pour afficher la page des moyens de paiement.

```
private val paymentController = PaymentController()
paymentController.showPaymentForm(uri)
```

La méthode `*showManageWallet*` est utilisée pour afficher la page de gestion du portefeuille.

```
private val walletController = WalletController()
walletController.manageWebWallet(uri)
```

Ces deux méthodes requièrent l'uri de la page vers laquelle nous devons être redirigés. La récupération du paramètre `*uri*` se fera selon vos choix d'implementation.

Pour plus d'informations, veuillez vous référer à la documentation Payline en cliquant : PW - Intégration Widget

PaymentController

Une fois que la page des moyens de paiement a été affichée, plusieurs méthodes sont accessibles :

```
fun updateWebPaymentData(data: JSONObject)
```

`*updateWebPaymentData*` permet de mettre à jour les informations de session de paiement après l'initialisation du widget et avant la finalisation du paiement.

```
fun getIsSandbox()
```

`*getIsSandbox*` permet de savoir si l'environnement est une production ou une homologation.

```
fun endToken(handledByMerchant: Boolean, additionalData: JSONObject?)
```

`*endToken*` permet de mettre fin à la vie du jeton de session web.

```
fun getLanguageCode()
```

`*getLanguageCode*` permet de connaitre la clé du language du widget.

```
fun getContextInfo(key: ContextInfoKey)
```

`getContextInfo` permet de connaitre l'information dont la clé est passée en paramètre.

Les différentes clés disponibles pour cette fonction sont les suivantes :

- AMOUNT_SMALLEST_UNIT("PaylineAmountSmallestUnit")
- CURRENCY_DIGITS("PaylineCurrencyDigits")
- CURRENCY_CODE("PaylineCurrencyCode")
- BUYER_FIRST_NAME("PaylineBuyerFirstName")
- BUYER_LAST_NAME("PaylineBuyerLastName")
- SHIPPING_ADDRESS_STREET2("PaylineBuyerShippingAddress.street2")
- SHIPPING_ADDRESS_COUNTRY("PaylineBuyerShippingAddress.country")
- SHIPPING_ADDRESS_NAME("PaylineBuyerShippingAddress.name")
- SHIPPING_ADDRESS_STREET1("PaylineBuyerShippingAddress.street1")
- SHIPPING_ADDRESS_CITYNAME("PaylineBuyerShippingAddress.cityName")
- SHIPPING_ADDRESS_ZIPCODE("PaylineBuyerShippingAddress.zipCode")
- BUYER_MOBILE_PHONE("PaylineBuyerMobilePhone")
- SHIPPING_ADDRESS_PHONE("PaylineBuyerShippingAddress.phone")
- BUYER_IP("PaylineBuyerIp")
- FORMATTED_AMOUNT("PaylineFormattedAmount")
- ORDER_DATE("PaylineOrderDate")
- ORDER_REF("PaylineOrderRef")
- ORDER_DELIVERY_MODE("PaylineOrderDeliveryMode")
- ORDER_DELIVERY_TIME("PaylineOrderDeliveryTime")
- ORDER_DETAILS("PaylineOrderDetails")

PaymentControllerListener

Le `*PaymentControllerListener*` est une interface qui définit la communication entre l'application et le PaymentController. Il va permettre d'avertir la classe qui l'implémente lorsque des données ou des actions sont reçues. Il contient différentes méthodes :

```
fun didShowPaymentForm()
```

`*didShowPaymentForm*` est la méthode appelée lorsque la liste des moyens de paiement a été affichée.

```
fun didFinishPaymentForm(state: WidgetState)
```

`*didFinishPaymentForm*` est la méthode appelée lorsque le paiement a été terminé. Elle reçoit en paramètre un objet de type `*widgetState*` qui correspond aux différents états possibles lors de la fin du paiement. Cet objet peut prendre les valeurs suivantes :

- PAYMENT_METHODS_LIST
- PAYMENT_CANCELED
- PAYMENT_SUCCESS
- PAYMENT_FAILURE
- PAYMENT_FAILURE_WITH_RETRY
- TOKEN_EXPIRED
- BROWSER_NOT_SUPPORTED
- PAYMENT_METHOD_NEEDS_MORE_INFOS
- PAYMENT_REDIRECT_NO_RESPONSE
- MANAGE_WEB_WALLET
- ACTIVE_WAITING
- PAYMENT_CANCELED_WITH_RETRY
- PAYMENT_ONHOLD_PARTNER
- PAYMENT_SUCCESS_FORCE_TICKET_DISPLAY

```
fun didGetIsSandbox(isSandbox: Boolean)
```

`*didGetIsSandbox*` est la méthode appelée lorsque l'environnement a été reçu. `*isSandbox*` vaudra true si l'environnement est une production et false si c'est une homologation.

```
fun didGetLanguageCode(language: String)
```

`*didGetLanguageCode*` est la méthode appelée lorsque la clé du language du widget est connue. `*language*` peut avoir des valeurs comme "fr", "en" ...

```
fun didGetContextInfo(key: ContextInfoKey)
```

`*didGetContextInfo*` est la méthode appelée lorsque l'information du contexte est connue. Le paramètre `*key*` est de type `*ContextInfoResult*`. Il s'agit d'une classe qui va être utilisée pour traiter le résultat obtenu par la webView. Trois types de données pourront être reçus : "Int", "String" ou "ObjectArray".

WalletControllerListener

Le `*WalletControllerListener*` est une interface qui définit la communication entre l'application et le WalletController. Il va permettre d'avertir la classe qui l'implémente lorsque des données ou des actions sont reçues. Il contient une méthode :

```
fun didShowManageWebWallet()
```

`*didShowManageWebWallet*` est la méthode appelée lorsque la page de gestion du portefeuille a été affiché.