SDK

SberPay

Exported on 08/31/2022

Table of Contents

1 Клиентский сценарий 3
2 Реализация на iOS 4
2.1 Основной сценарий 4
2.2 Последовательность выполнения сценария SDK на Swift 4
2.3 Последовательность выполнения сценария SDK на Objective C 5
3 Реализация на Android 7
3.1 Подключение SDK (Java & Kotlin) 7
3.2 Добавление кнопки "Оплатить SberPay" (Java & Kotlin) 7
3.3 Видимость кнопки "Оплатить SberPay" 8
3.4 Реализация обработчика кнопки "Оплатить SberPay" 8

1 Клиентский сценарий

Клиент в мобильном приложении мерчанта совершает покупку. После формирования корзины переходит к экрану оплаты. У клиента должна отобразиться кнопка оплаты SberPay. После тапа на нее открывается мп сбол, клиент аутентифицируется в мп сбол и видит флоу оплаты App2App SberPay.

2 Реализация на iOS

2.1 Основной сценарий

Для возможности проверять наличие установленного МП СБОЛ на устройстве, необходимо поправить Info.plist вашего приложения, добавив следующее:

```
<key>LSApplicationQueriesSchemes</key>
<array>
    <string>sberbankonline</string>
    <string>sbolpay</string>
</array>
```

2.2 Последовательность выполнения сценария SDK на Swift

1. Проверка установленного МП СБОЛ. Требуется вызвать метод **isSberbankAppInstalled**. Если проверка успешна (мп сбол установлен), то метод **isSberbankAppInstalled** вернет **true**, и кнопку оплаты следует отображаеть / делаеть активной. Если ни одна из поддерживаемых версий МП СБОЛ не установлена, метод вернет **false**,

Проверка установленного мп сбол на SWIFT

```
SberPay.isSberbankAppInstalled
```

2. Для старта сценария SDK, нужно создать модель PaymentRequest. Она может содержать обязательный параметр invoiceId (идентификатор платежа).

Пример создания модели на SWIFT

```
let requestModel = PaymentRequest(invoiceId: "12345678") или
let requestModel = PaymentRequest(invoiceId: "12345678", redirectUri: "https://google.com|merchantapp://")
SberPay.pay(with: requestModel) { result in
    // check result
}
```

3. Создание кнопки оплаты.

Кнопка оплаты на SWIFT

```
let payButton = PayButton()
payButton.tapAction = {
 // action
}
```

4. Результат выполнения сценария SDK типа Result.
    a. SDK успешно отработал сценарий и успешно запустился мп сбол, SDK передает статусный код ***success***.
    b. Если длина параметра (идентификатора заказа), полученного для формирования универсальной ссылки, превышает 128 символов или пустое, SDK передает статусный код ***invalidParams***.
    c. При невозможности запуска мп сбол по сформированной универсальной ссылке, SDK передает статусный код ***launchError***.
    d. В иных ситуациях исключительных ситуациях, SDK передает статусный код ***unknown***.

2.3 Последовательность выполнения сценария SDK на Objective C

1. Проверка установленного МП СБОЛ. Требуется вызвать метод **isSberbankAppInstalled**. Если проверка успешна (мп сбол установлен), то метод **isSberbankAppInstalled** вернет **true**, и кнопку оплаты следует отображаеть / делаеть активной. Если ни одна из поддерживаемых версий МП СБОЛ не установлена, метод вернет **false**.

Проверка установленного мп сбол на Objective C

```
SBPSberPay.isSberbankAppInstalled
```

2. Для старта сценария SDK, нужно создать модель PaymentRequest. Она может содержать обязательный параметр invoiceId (идентификатор платежа) и необязательный параметр redirectUri (ссылка / диплинк для возврата в магазин мерчанта). redirectUri является обязательным, если при регистрации операции он не передается в АС Банка. Использование параметра redirectUri возможно только для iOS.

Пример создания модели на Objective C

```
SBPPaymentRequest *requestModel = [[SBPPaymentRequest alloc] initWithInvoiceId:@"123456578"]
 или
SBPPaymentRequest *requestModel = [[SBPPaymentRequest alloc] initWithInvoiceId:@"123456578"
 redirectUri:@"https://google.com|merchantapp://"]
[SBPSberPay payWithRequest:requestModel completionHandler:^(SBPResult result) {
    // check result
}];
```

3. Создание кнопки оплаты.

Кнопка оплаты на Objective C

```
SBPPayButton *payButton = [SBPPayButton new];
payButton.tapAction = ^{
 // action
};
```

4. Результат выполнения сценария SDK типа **SBPResult**:
    a. SDK успешно отработал сценарий и успешно запустился мп сбол, SDK передает статусный код ***SBPResultSuccess***.
    b. Если длина параметра (идентификатора заказа), полученного для формирования универсальной ссылки, превышает 128 символов или пустое, SDK передает статусный код invalidParams ***SBPResultInvalidParams***.
    c. При невозможности запуска мп сбол по сформированной универсальной ссылке, SDK передает статусный код ***SBPResultLaunchError***.
    d. В иных ситуациях исключительных ситуациях, SDK передает статусный код unknown ***SBPResultunknown***.

3 Реализация на Android

3.1 Подключение SDK (Java & Kotlin)

В данный момент SDK поставляется в виде файла .aar. SDK поддерживает версии Android 4.2 и выше. Для подключение SDK к проекту нужно

1. В файле *build.gradle* проекта в подсекцию *repositories* секции *allprojects* добавить инструкцию *flatDir* (возможность подключать зависимости локально). Секция будет выглядеть, примерно, следующим образом:

```
allprojects {
    repositories {
        google()
        jcenter()
        // прочие репозитории
        flatDir { // вот эти
            dirs 'libs' // строки
        } // нужно вставить
    }
}
```

2. В корне модуля, к которому SDK будет подключаться как зависимость, нужно создать директорию libs (если ее нет) и положить туда aar-архив SDK.
3. В файле *build.gradle* модуля, к которому SDK будет подключаться как зависимость, требуется эту зависимость добавить:

```
dependencies {
    implementation fileTree(dir: "libs", include: ["*.jar"])
    implementation "org.jetbrains.kotlin:kotlin-stdlib:$kotlin_version"
    implementation 'androidx.core:core-ktx:1.3.1'
    implementation 'androidx.appcompat:appcompat:1.2.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.0.1'
    // прочие зависимости
    implementation(name:'sberpaysdk', ext:'aar') // вот эту строку нужно добавить (файл с sdk - sberpaysdk.aar)
}
```

3.2 Добавление кнопки "Оплатить SberPay" (Java & Kotlin)

Для добавления кнопки в файле верстки макета можно воспользоваться следующим прототипом:

```
<sberpay.sdk.sberpaysdk.view.SberButton
    android:id="@+id/submit_button_large"
    app:buttonCornerRadius="16dp"
    app:buttonType="white_type"
    android:layout_width="300dp"
    android:layout_height="80dp" />
```

Используйте *app:buttonCornerRadius* для того, чтобы задать радиус округления углов кнопки. Если данный атрибут не задать, углы округляться не будут.

Используйте *app:buttonType* для того, чтобы задать стиль кнопки. Если данный атрибут не задать, то будет использован стиль по умолчанию (зеленый фон, белый цвет шрифта и логотипа).

Также, можно указывать стандартные атрибуты Android View.

3.3 Видимость кнопки "Оплатить SberPay"

Кнопку нужно отображать только в том случае, если на мобильном устройстве клиента установлено мобильное приложение Сбербанк Онлайн. В противном случае кнопка должна быть скрыта. Реализовать данное поведение можно используя следующий код:

Kotlin

```
1        val sberbankOnlineManager = SberbankOnlineManager() // рекомендуется вынести данную
переменную в свойство
2        findViewById<View>(R.id.submit_button).visibility = if
 (sberbankOnlineManager.isSberbankOnlineInstalled(context)) View.VISIBLE else View.GONE
```

Java

```
1        SberbankOnlineManager sberbankOnlineManager = new SberbankOnlineManager(); //
рекомендуется вынести данную переменную в свойство
2        View submitButton = findViewById(R.id.submit_button);
3        if (sberbankOnlineManager.isSberbankOnlineInstalled(getContext())) {
4            submitButton.setVisibility(View.VISIBLE);
5        } else {
6            submitButton.setVisibility(View.GONE);
7        }
```

3.4 Реализация обработчика кнопки "Оплатить SberPay"

При тапе клиента на кнопку "Оплатить" SberPay, мерчант должен зарегистрировать заказ в платежном шлюзе и получить от него уникальный идентификатор операции bankInvoiceId. "bankInvoiceId" требуется передать в соответствующей функции для последующего перехода в Сбербанк Онлайн для оплаты заказа.

Kotlin

```
1       val sberbankOnlineManager = SberbankOnlineManager() // рекомендуется вынести данную
переменную в свойство
2       sberbankOnlineManager.openSberbankOnline(context, bankInvoiceId)
```

Java

```
1       SberbankOnlineManager sberbankOnlineManager = new SberbankOnlineManager(); //
рекомендуется вынести данную переменную в свойство
2       sberbankOnlineManager.openSberbankOnline(getContext(), bankInvoiceId)
```