Smartkasa PosAPI .net sdk

Smartkasa PosAPI .net sdk дозволяє інтегрувати термінал Smartkasa зі сторонньою касою. Нижче наведені опис та приклади роботи зі Smartkasa PosAPI за допомогою PosAPI .net sdk.

Початок роботи

Для того, щоб підімкнути sdk до Вашого .net проєкту потрібно спочатку додати до залежностей файл з розширенням `.dll` (має надаватися разом з цією документацією). Далі необхідно ініціалізувати sdk. Для цього виконайте наступні дії:

Задайте бажану швидкість порту. Ця дія не є обов'язковою, адже sdk за замовчуванням використовує швидкість порту 19200.

```
PosApiSdk.GetInstance().SetPortSpeed(port_speed);
```

Далі скористайтесь одним зі способів для того, щоб почати роботу з sdk:

```
PosApiSdk.GetInstance().ConnectAutomatically((api) => {
    // Використовуйте об'єкт api для подальшої роботи
}, (error) => {
    // Виникла помилка
});
```

Якщо ж Ви хочете власноруч задавати ім'я порту для підключення - використайте наступний метод:

```
PosApiSdk.GetInstance().Connect("port_name", (api) => {
    // Використовуйте об'єкт api для подальшої роботи
}, (error) => {
    // Виникла помилка
});
```

Ви також можете отримати посилання на об'єкт `api` використовуючи `PosApiSdk.GetApi()`. Використовуйте цей метод тільки після ініціалізації sdk.

Перевірка зв'язку

Якщо Вам потрібно перевірити, чи є зв'язок з терміналом, Ви можете використати метод `Ping`:

```
api.Ping(() => {
    // Зв'язок з терміналом є
}, (error) => {
    // Помилка зв'язку або інша помилка
});
```

Друк логів

Ви можете скористатись вбудованою системою логів - так Ваші логи будуть відображатись у такому ж форматі, що й логи PosAPI. Для цього зробіть наступне:

```
Log.Write("ваш лог тут"); // Ваші логи будуть містити рядок "ECR", що означає "стороння каса"
```

Проведення транзакцій

Наразі PosAPI підтримує наступні транзакції:
1. Покупка
2. Покупка з додатковою сумою
3. Повернення коштів
4. Відміна
5. Преавторизація
6. Завершення преавторизації
7. Копія чеку
8. Звіт X
9. Звіт Z
10. Отримання інформації про транзакці(ю/ї)
11. Отримання інформації про продавц(я/ів)

Увага! Перед тим, як проводити транзакцію - переконайтесь, що попередню транзакцію було завершено. Проведення більш, ніж однієї транзакції одночасно може призвести до нестабільної роботи PosAPI та/або PosAPI sdk. Транзакція вважається проведеною, якщо була отримана будь-яка відповідь (про успішне завершення або про помилку).

Увага! Деякі операції, такі як: покупка, покупка з ддатковою сумою, преавторизація, завершення преавторизації, повернення коштів, відміна дають можливість використовувати додаткові дані. Задати додаткові дані можна таким чином:

```
List<PackageField> additonalData = new List<PackageField>();
additonalData.Add(new PackageField("62", "value"));
additonalData.Add(new PackageField("63", "value"));
```

Якщо Ви не бажаєте використовувати додаткові дані - можете передати `null` замість `additionalData`.

Покупка

Для покупки викличте метод `Purchase` у об'єкта `api`:

```
api.Purchase(100 /*сума*/, "merchant_id" /*id продавця*/, additonalData /*додаткові дані*/, (status, data) => {
    // Використовуйте параметри data та status дял отримання інформації про транзакцію
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Якщо транзакцію було проведено, Ви можете отримати дани про неї, використовуючи параметри `status` та `data`. Параметр `status` містить в собі інформацію про те, чи була транзакція проведена успішно: `status.isSuccessfull`, а також код помилки, якщо виникли помилки під час проведення: `status.errorCode`. Параметр `data` містить наступну інформацію про транзакцію: `data.action`, `data.amount`, `data.approvalCode`, `data.date`, `data.maskedPan`, `data.merchant`, `data.rrn`.

Покупка з додатковою сумою

Для здійснення покупки з додатвовою сумою, виконайте наступну дію:

```
api.PurchaseAdditional(100 /*сума*/, 150 /*додаткова сума*/, "merchant_id" /*id продавця*/, additonalData /*додаткові дані*/, (statu
    // Використовуйте параметри data та status дял отримання інформації про транзакцію
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Отримати інформацію про помилку або про транзакцію можна ідентичним шляхом до транзакції *Покупка*.

Повернення коштів

Щоб здійснити повернення коштів, необхідно виконати вказане нижче:

```
api.Refund(100 /*сума*/, "merchant_id" /*id продавця*/, additonalData /*додаткові дані*/, (status, data) => {
    // Використовуйте параметри data та status дял отримання інформації про транзакцію
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Отримати інформацію про помилку або про транзакцію можна ідентичним шляхом до транзакції *Покупка*.

Відміна

Щоб здійснити відміну, виконайте наступну дію:

```
api.Reversal("rrn" /*код транзакції*/, "merchant_id" /*id продавця*/, additonalData /*додаткові дані*/, (status, data) => {
    // Використовуйте параметри data та status дял отримання інформації про транзакцію
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Отримати інформацію про помилку або про транзакцію можна ідентичним шляхом до транзакції *Покупка*.

Преавторизація

Для здійснення преавторизації, скористайтесь цим методом:

```
api.PreAuthorize(100 /*сума*/, "merchant_id" /*id продавця*/, additonalData /*додаткові дані*/, (status, data) => {
    // Використовуйте параметри data та status дял отримання інформації про транзакцію
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Отримати інформацію про помилку або про транзакцію можна ідентичним шляхом до транзакції *Покупка*.

Завершення преавторизації

Щоб завершити преавторизацію, використайте цей метод:

```
api.PreAuthorizeFinish("rrn" /*код транзакції*/, "merchant_id" /*id продавця*/, additonalData /*додаткові дані*/, (status, data) =>
    // Використовуйте параметри data та status дял отримання інформації про транзакцію
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Отримати інформацію про помилку або про транзакцію можна ідентичним шляхом до транзакції *Покупка*.

Копія чеку

Скористайтесь нижче наведеним методом щоб отримати копію чеку:

```
api.Receipt("rrn" /*код транзакції*/, (status, data) => {
    // Використовуйте параметри data та status дял отримання інформації про транзакцію
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Отримати інформацію про помилку або про транзакцію можна ідентичним шляхом до транзакції *Покупка*.

Звіт X

Щоб зробити X-звіт, виконайте наступне:

```
api.ReportX("merchant_id" /*id продавця*/, (status, fields) => {
    // Використовуйте параметри fields та status дял отримання інформації про транзакцію
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Використайте параметр `fields`, який є списком об'єктів класу `PackageField`, щоб отримати дані звіту. Клас `PackageField` має два параметри: `name`, `data`. Використайте їх для того, щоб отримати ім'я та значення з кожного об'єкту.

Звіт Z

Щоб зробити Z-звіт, виконайте наступну дію:

```
api.ReportZ("merchant_id" /*id продавця*/, (status, fields) => {
    // Використовуйте параметри fields та status дял отримання інформації про транзакцію
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Отримати інформацію про помилку або про транзакцію можна ідентичним шляхом до транзакції *Звіт X*.

Отримання інформації про транзакцію

Ви можете отримати інформацію про транзакцію, що вже була здійснена. Для цього Вам потрібно зробити наступне:

```
api.GetTransaction("rrn" /*код транзакції*/, "merchant_id" /*id продавця*/, (status, data) => {
    // Використовуйте параметри data та status дял отримання інформації про транзакцію
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Отримати інформацію про помилку або про транзакцію можна ідентичним шляхом до транзакції *Покупка*.

Отримання інформації про транзакції

Ви також можете отримати інформацію про усі транзакції певного продавця. Для цього здійсніть дану операцію:

```
api.GetTransactions("merchant_id" /*id продавця*/, (status, data) => {
    // Використовуйте параметри data та status дял отримання інформації про транзакції
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Параметр `data` в даному випадку є списком транзакцій. Кожну транзакцію зі списку Ви можете опрацювати таким самим способом, як і в описаних вище операціях.

Отримання інформації про продавця

Для того, щоб отримати інформацію про конкретного продавця - використайте наступний метод:

```
api.GetMerchant("merchant_id" /*id продавця*/, (status, merchant) => {
    // Використовуйте параметри merchant та status дял отримання інформації про продавця
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Параметр `merchant` є об'єктом класу `Merchant` і містіть у собі параметри `name` - назва продавця, `mid` - id продавця, `tid` - id терміналу.

Отримання інформації про продавців

Ви можете отримати інформацію про усіх продавців, присутніх на терміналі, Для цього зробіть наступне:

```
api.GetMerchants((status, merchants) => {
    // Використовуйте параметри merchants та status дял отримання інформації про продавців
}, (error) => {
    // Використовуйте параметр error для отримання інформації про помилку
});
```

Параметр `merchants` є списком об'єктів класу `Merchant` , Кожен елемент списку може бути опрацьований за аналогією до транзакції *Отримання інформації про продавця*.