Проект Тип оборудования AcquiringTerminal

ОГЛАВЛЕНИЕ

1. Основные понятия и термины

Термин	Описание
Подключаемое оборудование	Любое торговое или промышленное оборудование, использующееся клиентским приложением
Тип оборудования	Регламентированное документацией множество, объединяющее устройства с одинаковым функциональным назначением
Драйвер оборудования, Драйвер	Программный компонент с возможностью отдельной установки или обновления в системе и предоставляющий интерфейс управления и взаимодействия с устройством определенной модели. Предоставляемый драйвером интерфейс должен строго соответствовать описанию типа оборудования по данной технологии, но может иметь расширения (дополнительные методы и/или параметры) для выполнения функций, не регламентированных в документах описания данной технологии
Модель, Модель оборудования	Множество устройств одного типа оборудования, одинаковых или близких по своим характеристикам, управляемых драйвером устройства. Модель оборудования определяется драйвером устройства. В случае, если один драйвер устройства позволяет единообразно управлять одной из нескольких сходных моделей физических устройств, для системы все эти устройства являются одной и той же моделью. Напротив, если для одного и того же физического устройства существует несколько разных драйверов, для системы они являются разными моделями.
Экземпляр оборудования	Совокупность файлов и данных о физическом устройстве и драйвере (идентификатор, наименование, список настроек и т. п.), хранимых и используемых для работы с данным устройством системой управления подключаемым оборудованием и приложением-клиентом.
Клиентское приложение	Любое приложение, использующее для управления подключаемым оборудованием описанные в документации по технологии интерфейсы и алгоритмы взаимодействия с драйверами устройств.
Менеджер оборудования	Часть клиентского приложения, в которой сосредоточены сервисные процедуры и регламентированные документацией интерфейсы взаимодействия с оборудованием.
Настройки устройства	Набор именованных значений, простых типов данных: String, Number, DateTime, Boolean, хранящийся вне клиентского приложения в файловом хранилище настроек. Для каждого определенного в системе устройства имеется свой набор настроек. Настройки устройства определяют параметры и режим функционирования устройства. Настройки доступны клиентскому приложению для чтения и записи с помощью специальных методов.
Уникальный идентификатор	В этом проекте – текстовое представление GUID. Идентификатор содержит 36 символов в верхнем регистре, не заключенных в фигурные скобки. Пример: 8A8910EC-78F5-41A5-9E18-7C28992CF580
Публикуемый метод	Один из определенных в описании интерфейса общих методов, либо методов типа. Все публикуемые методы имеют одинаковую структуру параметров. Входные и возвращаемые методом данные помещаются в два одномерных массива SafeArray
Функции обратного вызова	Регламентируемые данной документацией функции клиентского приложения, доступные для вызова через интерфейс IDIspatch драйвером оборудования. Функции обратного вызова предназначены для информирования приложения о событиях

Термин	Описание
	или передачи данных.
Событие	Событие возникает как результат изменения состояния устройства. Событие может быть вызвано внешним воздействием, либо внутренними процессами в оборудовании. Для передачи событий приложению драйвер устройства использует функции обратного вызова клиентского приложения
Объект-абстракт	Объект драйвера устройства, не ассоциированный с экземпляром оборудования. Любой объект драйвера является объектом-абстрактом до выполнения метода Open.У объекта-абстракта допустим вызов только тех методов, которые не требуют взаимодействия с физическим устройством. Объект-абстракт не может использовать функции обратного вызова, поскольку не имеет собственного идентификатора в системе.
Объект драйвера	Объект драйвера устройства, созданный клиентским приложением. Для каждого физического устройства создается отдельный экземпляр объекта драйвера

Термин	Описание
PIN	(англ. Personal Identification Number) - персональный идентификационный номер клиента.
Pin-pad, Pinpad, Pin pad	Устройство, оснащенное аппаратным Security модулем, предназначенное для выполнения различных криптографических операций, хранения ключей, чтения кредитных карт и для защищенного ввода PIN-кода клиента. Далее, под Pinpad подразумеваются также и устройства, относящиеся к классу POS (CAT), оснащенные принтером квитанций и имеющие расширенный функционал, по сравнению с Pinpad.
EFT	(англ. Electronic Funds Transfer) - платеж, осуществленный через электронную систему платежей.
CAT	(Credit Authorization Terminal) – банковский терминал для авторизации платежных карт
RRN, RN, Reference Number	Число, идентифицирующее транзакцию в системе расчетов владельца кредитной карточки. Каждый ссылочный номер транзакции напечатан на ежемесячном отчете, отсылаемом владельцу кредитной карточки. Номер ссылки используется, если необходим поисковый запрос для отмены или подтверждения транзакции
Settlement	Сверка итогов, закрытие смены. Удачное завершение данной операции говорит о том, что все данные, поступившие от терминала в течении смены, признаны корректными и учтены в базе данных платежной системы.
PAN	Первичный номер счета. Рельефное или закодированное число, которое идентифицирует выпускающего карты и счет.
Authorization Code	Код авторизации. Код, состоящий из букв и цифр, посылаемый банком-эмитентом (Card Issuer), подтверждающий авторизацию. Код авторизации обязательно включается в чек (Sales Draft), выписываемый продавцом.
Pre-Authorization	Блокирование суммы оплаты на счете кредитной карты. Заблокированная сумма впоследствии может быть либо списана со счета (подтверждение пре-авторизации) либо разблокирована (отмена пре-авторизации). Если в течение 32 дней не происходит подтверждения или отмены pre-authorization, сумма на счете автоматически разблокируется
EMV	EMV (Europay, MasterCard и VISA) — международный стандарт для операций по банковским картам с чипом. Этот стандарт разработан совместными усилиями компаниями Europay, MasterCard и Visa, чтобы повысить уровень безопасности финансовых операций.

2. Описание технологии

2.1. Наименование типа и синоним

Наименование типа англ.: AcquiringTerminal
Наименование типа рус.: ЭквайрингТерминал

Синоним англ.: Acquiring Terminal
Синоним рус.: Эквайринг-терминал

2.2. Краткое описание функциональности типа

Экземпляр типа Эквайринг-терминал (далее - драйвер) может выполнять все основные функции оплаты товаров и услуг с помощью платежной карты:

- Оплата товаров и услуг
- Возврат оплаты
- Отмена платежа
- Сверка итогов
- Пре-авторизация
- Подтверждение пре-авторизации
- Отмена пре-авторизации
- Запрос состояния счета (баланса)

Для реализации вышеперечисленных функций, драйвер взаимодействует с клиентским ПО платежной системы банка. Данное ПО требует подключения к системе Pin-pad, с помощью которого производится безопасное считывание карты и ввод PIN-кода. Считывание платежной карты (Visa, MasterCard и др.) на произвольном считывателе магнитных карт недопустимо из соображений безопасности.

3. Алгоритмы

3.1. Общие алгоритмы

В этой главе описаны общие для всех типов оборудования алгоритмы взаимодействия приложения с драйвером устройства.

Создание нового экземпляра оборудования

- Приложение создает объект драйвера устройства
- Следующим вызванным методом должен быть Init, инициализирующий драйвер. При этом в параметрах метода драйверу передается ссылка на интерфейс IDispatch клиентского приложения. Драйвер запоминает ссылку на интерфейс для обращения к функциям обратного вызова
- У созданного объекта вызывается метод метод GetDeviceInfo для получения массива DeviceInfo, содержащего константную информацию о драйвере (закладывается разработчиком при реализации драйвера)
- Для получения значений настроек по умолчанию, у драйвера вызывается метод GetSettings.
- Если драйвер имеет собственную форму для настройки оборудования (элемент IsSettingsDlgExist из структуры DeviceInfo), у него вызывается метод ShowSettingsDlg для её отображения. Если форма настроек отсутствует, приложение отображает свою универсальную форму, заполнив её значениями настроек по умолчанию
- Пользователь редактирует значения в форме настроек устройства и закрывает её
- Если пользователь отказался от сохранения настроек, нажав на форме кнопку «Отмена», считается, что он отказался от создания устройства. У драйвера вызывается метод Close и объект драйвера уничтожается. Процедура создания устройства прерывается.
- Если использовалась форма настройки приложения, то набор настроек передается в драйвер через метод SetSettings для проверки корректности их заполнения (валидации)
- Приложение генерирует уникальный идентификатор для нового экземпляра оборудования и создает соответствующий ему индивидуальный каталог в хранилище настроек
- У драйвера запрашивается набор его текущих (проверенных) настроек с помощью метода GetSettnigs. Набор настроек данного экземпляра оборудования сохраняется в индивидуальном каталоге хранилища настроек в файле Settings.xml (см. документ «Установка и обновление системы управления подключаемым оборудованием»)
- У данного объекта драйвера вызывается метод Close, после чего он выгружается из памяти приложения
- Клиентское приложение сохраняет идентификатор созданного устройства для последующего его использования (см. Использование экземпляра оборудования)

Использование экземпляра оборудования

В этом разделе описывается алгоритм работы с ранее созданным экземпляром оборудования.

- Приложение создает объект драйвера устройства. У созданного объекта вызывается метод Init, инициализирующий драйвер. При этом в параметрах метода драйверу передается ссылка на интерфейс IDispatch клиентского приложения (используется драйвером для обращения к функциям обратного вызова)
- Приложение вызывает метод драйвера SetSetiings, передавая в параметрах метода список значений настроек устройства. Драйвер применяет новые настройки.
- Приложение вызывает метод драйвера Open, в параметрах которого передается идентификатор устройства. Данный идентификатор драйвер при обращениях к функциям обратного вызова
- Начальным состоянием любого устройства после выполнения метода Open, является «Выключено». В этом состоянии допускается вызов следующих методов устройства: GetDeviceInfo, GetSettings, SetSettings, ShowSettingsDlg, CheckHealth, Enable, Disable, Close. Все остальные методы будут доступны после включения устройства.
- Перед началом реального использования оборудования приложение вызывает метод Enable. При выполнении метода Enable драйвер должен выполнить все необходимые операции по подготовке к работе: подгрузить необходимые ему внешние библиотеки, открыть, захватить, либо заблокировать необходимые для работы ресурсы: коммуникационные порты, файлы и т.п. Если метод Enable завершается успешно, то устройство переходит в состояние «Включено».
- У оборудования в состоянии «Включено» клиентское приложение последовательно вызывает необходимые ему публикуемые методы драйвера устройства.
- Драйвер устройства может порождать различные события посредством вызова у приложения функций обратного вызова.
- По окончании работы с драйвером устройства, приложение должно отключить его, вызвав метод Disable. При исполнении данного метода драйвер должен привести физическое устройство в исходное состояние, высвободить все захваченные им в процессе работы ресурсы (закрыть открытые им файлы, коммуникационные порты), выгрузить использовавшиеся сторонние библиотеки и т.п.
- Перед уничтожением объекта драйвера устройства, приложение вызывает метод-деструктор Close. Драйвер очищает все свои служебные структуры данных, в том числе очищает ссылку на интерфейс IDispatch клиентского приложения, переданную ему при вызове метода Init.

Удаление экземпляра оборудования

Для удаления экземпляра подключаемого оборудования из системы, достаточно удалить индивидуальный каталог экземпляра оборудования из хранилища настроек (см. описание в документе «Программа установки системы управления подключаемым оборудованием.doc») и очистить в настройках клиентских приложений, использовавших данный экземпляр, все ссылки на его идентификатор.

3.2. Специфические алгоритмы

Авторизация платежа требует получения номера карты. Карта может быть считана как на специальном устройстве – Pin-pad, так и с помощью отдельного считывателя магнитных карт, обычно совмещенного с клавиатурой кассира. В первом случае, функции запроса на считывание карты и получения данных от считывателя выполняет сама платежная система. Во втором случае, карта должна быть считана до отправки запроса на авторизацию платежной системе и номер карты должен быть передан в параметрах метода запроса. Однако в реальных банковских эквайринговых системах считывание карты на внешнем считывателе не допускается, поскольку это грубо нарушает существующие требования к безопасности таких систем. Однако, считывание карты на внешнем считывателе вполне допустимо для систем, работающих по алгоритму эквайринговых, но фактически не являющихся таковыми и не ограниченных стандартами безопасности (EMV).
Для осуществления авторизации платежа вызывается метод драйвера AuthorizeSales. Если входной параметр CustomerData – не заполнен (пустая строка), драйвер инициирует ввод данных карты на Pin-pad. В противном случае, в параметре CustomerData должны быть переданы «сырые» данные со второй дорожки магнитной карты или другие данные, необходимые системе, для идентификации плательщика.

В случае успешной авторизации, драйвер передает в выходных параметрах массив данных для печати квитанции. Клиентское приложение должно произвести печать на любом доступном принтере, назначенном в качестве принтера для печати квитанций, в случае, если свойство PrintReceiptOnPinpad равно Ложь. В случае ошибки в процессе печати квитанции, приложение должно вызвать метод отмены транзакции платежа. AuthorizeVoid.,
В случае возврата покупки, произведенной не в текущей смене, вызывается метод возврата AuthorizeRefund. Если же возврат покупки совершается, в текущей смене, вызывается метод отмены AuthorizeVoid.
Приложение должно сохранять параметры успешной транзакции (Amount, ReferenceNumber,ReceiptNumber, AuthorizationCode), возвращаемые методами AuthorizeSales, AuthorizeRefund, AuthorizePreSales и использовать их в качестве входных параметров при последующей отмене, возврате или подтверждении транзакции (методы AuthorizeVoid, AuthorizeRefund, AuthorizeVoidPreSales, AuthorizeCompletion). Это необходимо для поиска оригинальной отменяемой/подтверждаемой транзакции.
Для закрытия рабочей смены и сверки итогов предназначен метод Settlement. Обычно, эта операция производится одновременно с закрытием смены на кассовом аппарате в конце рабочего дня.
Метод AuthorizePreSales используется для пре-авторизации суммы на счете (см. «Основные понятия и термины»). Для последующего подтверждения пре-авторизации или её отмены предназначены, соответственно, методы AuthorizeCompletion и AuthorizeVoidPreSales.
С помощью необязательного метода GetBalance можно получить сумму остатка средств на счете. В зависимости от модели устройства, сумма может быть передана в параметре метода, тексте квитанции или отображена на экране Pin-pad.
В процессе выполнения транзакций, драйвер может вызывать метод TaskState, для информирования приложения о ходе выполнения операции. Использование TaskState крайне рекомендуется, поскольку процесс авторизации может быть достаточно длительным. При необходимости запроса дополнительных данных от пользователя в процессе выполнения транзакции, драйвер может вызывать у приложения (менеджера оборудования) методы MessageDlg, InputDataDlg. В любом случае, драйвер не должен самостоятельно отображать на экране какие-либо собственные окна.
Любые вышеописанные методы, кроме метода Сверки итогов, при их неудачном завершении могут передавать в массиве с информацией об ошибке параметры для печати квитанции в необязательном параметре ExtData. В этом случае, приложение должно напечатать квитанцию на принтере или фискальном регистраторе.
Параметры печати квитанции могут содержать как готовый текст квитанции в виде строки, так и набор параметов для печати квитнации по шаблону. В последнем случае, драйвер должен содержать в своём дистрибутиве шаблон квитанции для всех используемых методов. Шаблон должен иметь столько секций, сколько операций поддерживается драйвером. Имя каждой секции должно совпадать с именем соответствующего метода драйвера (AuthorizeSales, AuthorizeRefund и т. п.).
Если значение настройки PrintReceiptOnPinpad равно True, приложение игнорирует параметр ReceiptParameters и не печатает квитанции. Подразумевается, что в этом случае используется банковский терминал вместо Pinpad, оснащенный печатающим устройством на котором и печатаются тексты квитанций.
Если значение настройки PrintReceiptOnPinpad равно False и, при этом, после успешного выполнения методов AuthorizeSales, AuthorizeRefund и AuthorizePreSales, драйвер не вернул в том или ином виде данные для печати квитанции, это расценивается как ошибка. В этом случае транзакция должна быть немедленно отменена приложением (см. описание метода AuthorizeVoid).

4. Настройки

4.1. Общие настройки

В этом разделе описываются настройки общие для всех моделей данного типа оборудования. Данные настройки подлежат обязательной реализации в любом драйвере подключаемого оборудования.

DefaultTimeOut

Описание:

Значение таймаута выполнения метода по умолчанию в секундах. Если таймаут, преданный в параметрах какого-либо метода, равен нулю, драйвер использует в качестве таймаута значение данной настройки.

Тип	Допустимые значения	Значение по умолчанию	Представление
Number	10..65535	120	Таймаут выполнения метода по умолчанию (в сек.)

EventLogEnabled

Описание:

Данная настройка предназначена для включения режима записи диагностической информации в журнал событий, либо лог-файл. Используется в целях отладки и диагностики проблем с драйвером устройства. Если установлено значение True, драйвер вызывает функцию обратного вызова WriteLog в те моменты, когда это необходимо (определяется разработчиком). Значение по умолчанию для данной настройки – False

Тип	Допустимые значения	Значение по умолчанию	Представление
Boolean	True/False	False	Запись диагностической информации

4.2. Специфические настройки

В этом разделе документа описываются настройки, специфичные для описываемого типа оборудования.

TraningMode

Описание:

Если равно True, каждая операция выполняется в тренировочном режиме. Если равно False, все операции выполняются в нормальном режиме.

Тип	Допустимые значения	Значение по умолчанию	Представление
Boolean	True/False	False	Тренировочный режим

CutCode

Описание:

Код символа отреза бумаги. Если драйвер самостоятельно формирует текст квитанции, символ с данным кодом вставляется в места, где должен быть произведен отрез бумаги между копиями квитанции

Тип	Допустимые значения	Значение по умолчанию	Представление
Number	0..255	22	Код символа отреза

CurrencyCode

Описание:

Код валюты. Значение соответствует коду валюты по Общероссийскому классификатору валют. Значение по умолчанию – 643 (деноминированный рубль).

Тип	Допустимые значения	Значение по умолчанию	Представление
String	«000»..«999»	643	Код валюты

CapAuthorizePreSales

Описание:

Если равно True, поддерживаются методы пре-авторизации, её отмены и подтверждения.

Тип	Допустимые значения	Значение по умолчанию	Представление
Boolean	True/False	False	Поддерживается пре-авторизация

PrintReceiptOnPinpad

Описание:

Если равно True, квитанции печатает Pinpad и приложение не должно обрабатывать возвращаемый методами параметр ReceiptParameters и производить печать квитанций. В противном случае, приложение выполняет печать квитанций самостоятельно. Только для чтения

Тип	Допустимые значения	Значение по умолчанию	Представление
Boolean	True/False	False	Печать квитанций на Pinpad

DataInputMode

Описание:

Режим ввода данных клиента. Регламентирует место ввода данных клиента – в приложении, с последующей передачей данных драйверу в параметре CustomerData или непосредственно драйвером во время проведения транзакции

Тип	Допустимые значения	Значение по умолчанию	Представление
Number	0 – Автоматически, 1 – Передаются в параметре CustomerData, 2 – считываются драйвером с карты во время транзакции	0	Режим ввода данных. Если равно 0, драйвер сам определяет режим ввода

DataInputText

Описание:

Текст заголовка окна запроса ввода данных (см. выше). Данный текст должно отображать приложение в заголовке окна при запросе ввода данных клиента (CustomerData). Примеры: «Прокатайте карту», «Введите номер комнаты», «Считайте штрихкод»

Тип	Допустимые значения	Значение по умолчанию	Представление
String	Строка	Прокатайте карту	Текст запроса ввода данных

5. Методы

5.1. Служебные методы

В данном разделе описаны методы, обеспечивающие взаимодействие клиентского приложения и драйвера на начальных и конечных этапах цикла использования оборудования. Данные методы подлежат обязательной реализации в любом драйвере.

Init

Синтаксис:

Init (ApplicationRef: COM-object, ErrorDescription: String): Boolean

Описание:

Метод-конструктор объекта драйвера. При выполнении данного метода происходит начальная инициализация драйвера. Метод вызывается единожды, сразу после создания объекта. Любые другие методы объекта драйвера могут быть вызваны только после успешного выполнения метода Init. Если метод вернул значение False, объект драйвера должен быть уничтожен,

оставив систему в исходном состоянии. Если метод выполнен успешно, то с объектом драйвера можно работать как с объектом-абстрактом. Например получить список настроек по умолчанию, вызвав метод GetSettings.

Параметры:

ApplicationRef: COM-object

Ссылка на интерфейс IDispatch приложения-клиента. Используя данную ссылку, драйвер может вызывать у приложения функции обратного вызова (после выполнения метода Open).

ErrorDescription: String

Если метод возвращает False, в данном параметре содержится описание ошибки.

Возвращаемое значение:

True – метод выполнен успешно. False – в противном случае. В случае, если метод вернул результат False, дальнейшая работа с драйвером невозможна и данный объект драйвера должен быть уничтожен

Open

Синтаксис:

Open (DeviceID: String, ErrorDescription: String): Boolean

Описание:

Завершает инициализацию объекта драйвера и присваивает ему уникальный идентификатор экземпляра. В процессе выполнения данного метода, драйвер не должен захватывать какие-либо неразделяемые ресурсы (например, коммуникационные порты или файлы), и не должен выполнять никаких реальных взаимодействий с устройством. Драйвер проверяет наличие необходимых ему для работы ресурсов (файлов библиотек, их версий и т.п.) и возвращает Ложь, если проверки обнаружили невозможность полноценного функционирования драйвера. После успешного выполнения метода допускается вызов следующих методов драйвера: ShowSettingsDlg, CheckHealth, Enable, Disable..Все остальные методы будут доступны после включения устройства (метод Enable).
После выполнения метода драйвер может использовать функции обратного вызова.

Параметры:

DeviceID: String (GUID)

Уникальный идентификатор экземпляра оборудования. Идентификатор необходим при использовании функций обратного вызова.

ErrorDescription: String

Если метод вернул False, данный параметр содержит описание ошибки.

Возвращаемое значение:

True – метод выполнен успешно. False – в противном случае. В случае, если метод вернул результат False, дальнейшая работа с драйвером невозможна. Необходимо вызвать метод Close и уничтожить объект драйвера.

Close

Синтаксис:

Close (): Boolean

Описание:

Метод-деструктор драйвера. Метод должен вызываться непосредственно перед уничтожением объекта драйвера. При выполнении данного метода, драйвер устройства должен освободить все захваченные ресурсы, и очистить ссылку на интерфейс IDispatch приложения, переданную в параметрах метода Init. Никакие другие методы драйвера не могут быть вызваны после вызова Close

Параметры:

Нет

Возвращаемое значение:

True

GetDeviceInfo

Синтаксис:

GetDeviceInfo (InputParameters: Undefined, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:

Метод предназначен для получения статической информации о модели устройства. Допустимо вызывать данный метод в любой момент, даже у неинициализированного объекта (то есть до метода Init)

Параметры:

InputParameters: Undefined

Не используется. Необходимо передавать пустое значение.

ResultData: SafeArray

При успешном исполнении метода данный параметр содержит массив SafeArray с информацией о модели устройства. В случае ошибки выполнения метода, параметр содержит массив ErrorInfo

TimeOut: Number

Таймаут ожидания результата выполнения метода. (см. TimeOut)

Возвращаемое значение:

True – метод выполнен успешно. False – в противном случае

Массив ResultData:

Индекс SafeArray	Имя	Тип	Описание
0	TypeName	String	Наименование типа оборудования
1	ModelID	String	GUID модели оборудования
2	ModelName	String	Наименование модели
3	MajorVersion	Number	Мажор-версия системы (контроль совместимости на строгое равенство)
4	MinorVersion	Number	Минор-версия системы (совместимость снизу-вверх)
5	BuildVersion	Number	Текущая версия драйвера (билд), никак не контролируется системой
6	ProcessorName	String	Программный идентификатор драйвера. Используется для создания объекта драйвера
7	ProcessorType	Number	Тип драйвера: 1 = COM объект Win32, 2 = COM объект Win64, 3 = Native API Win32, 4 = Native API Win64, 5 = Native API Linux32, 6 = Native API Linux64
8	IsSettingsDlgExist	Boolean	Наличие собственной формы настроек (см. описание метода ShowSettingsDlg)
9	Description	String	Подробное описание модели
10	DeveloperName	String	Разработчик
11	CI_email	String	Контактная информация
12	CI_www	String	Контактная информация
13	CI_Phones	String	Контактная информация
14	CI_address	String	Контактная информация

5.2. Общие методы

Перечисленные в данном разделе методы являются обязательными для реализации в любом драйвере любого типа оборудования, использующего данную технологию. Совокупность общих и специфических методов называется «публикуемыми» методами. Публикуемые методы доступны для вызова из любого места приложения, в то время как служебные методы должен использовать только менеджер оборудования и строго по описанным алгоритмам взаимодействия. Вызов любого публикуемого метода возможен только у полностью инициализированного объекта драйвера (см. описание метода Open) за исключением методов GetSettings, GetDeviceInfo и ShowSettingsDlg, которые допустимо вызывать у объекта-абстракта. Входные и выходные параметры публикуемых методов всегда помещаются в одномерные массивы SafeArray. Все публикуемые методы всегда имеют три параметра:

InputParameters: SafeArray

Одномерный массив SafeArray с входными параметрами метода, либо Неопределено (Undefined) когда входные параметры отсутствуют.

ResultData: SafeArray

Одномерный массив SafeArray с результатами выполнения метода, либо Неопределенно (Undefined), если возвращаемые параметры отсутствуют. В случае, когда метод возвращает False, в данном параметре содержится массив, содержащий код и описание ошибки (см. описание массива ErrorInfo)

TimeOut: Number

Определяет максимальное время ожидания выполнения метода в секундах. Имеются два зарезервированных предопределенных значения:
0 – время исполнения метода не задано. Драйвер должен использовать значение настройки DefaultTimeOut.
-1 – время исполнения метода не ограничено.

Массив ErrorInfo:

Индекс в SafeArray	Имя	Тип	Описание
0	ErrorCode	Number	Регламентированный код ошибки (см. документ «Коды ошибок и их диапазоны.doc»)
1	ErrorDescription	String	Описание ошибки оборудования
2	ExtData	SafeArray / String / Number / DateTime / Boolean	Дополнительные данные или Undefined. Тип параметра определяется типом оборудования и вызываемым методом. Необязательный

GetSettings

Синтаксис:

GetSettings (InputParameters: Undefined, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:

Получение списка значений текущих настроек устройства. Если метод вызывается до первого вызова метода SetSettings, возвращаются значения настроек устройства по умолчанию, заданные разработчиком драйвера.

Параметры:

InputParameters: Undefined

Не используется. При вызове необходимо передавать значение Undefined.

ResultData: SafeArray

Массив значений настроек Settings. В случае ошибки выполнения метода, параметр содержит массив ErrorInfo

TimeOut: Number

Таймаут ожидания результата выполнения метода. (см. TimeOut)

Возвращаемое значение:

True – метод выполнен успешно. False – в противном случае.

Массив ResultData:

Индекс SafeArray	Имя	Тип	Описание
0	Settings	SafeArray	Массив значений настроек устройства

Массив Settings

Двумерный массив значений настроек устройства. Индекс старшего измерения (строки массива) изменяется в диапазоне от 0 до N (N = количество настроек - 1). Индекс младшего измерения (колонки массива) изменяется в диапазоне от 0 до 6

Индекс SafeArray	Имя	Тип	Описание
(0..N, 0)	ID	String	Идентификатор настройки. Идентификатор должен начинаться с буквы. Допустимо использование только латинских символов.
(0..N, 1)	Value	Number /	Значение настройки

		String / DateTime / Boolean	
(0..N, 2)	Type	String	Строковое представление типа значения настройки. Допустимые варианты: «Number» «String» «DateTime» «Boolean»
(0..N, 3)	Default	Number / String / DateTime / Boolean	Значение настройки по умолчанию
(0..N, 4)	ReadOnly	Boolean	Если равно True, то пользователям запрещается изменять значение данной настройки
(0..N, 5)	Name	String	Пользовательское представление настройки
(0..N, 6)	Description	String	Текстовое описание настройки, ее возможных значений и их воздействия на поведение драйвера

SetSettings

Синтаксис:
SetSettings (InputParameters: SafeArray, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Метод предназначен для установки новых значений настроек устройства. Допустимо вызывать данный метод до вызова метода Open. Метод не должен вызываться приложением, если устройство находится в состоянии «Включено» (см. метод Enable).

Параметры:
InputParameters: SafeArray
Массив значений настроек Settings.
ResultData: SafeArray либо Undefined
Возвращаемые параметры отсутствуют. В случае ошибки выполнения метода, параметр содержит массив ErrorInfo
TimeOut: Number
Таймаут ожидания результата выполнения метода. (см. TimeOut)

Возвращаемое значение:
True – метод выполнен успешно. False – в противном случае

Массив InputParameters:

Индекс SafeArray	Имя	Тип	Описание
0	Settings	SafeArray	Массив значений настроек устройства

ShowSettingsDlg

Синтаксис:
ShowSettingsDlg (InputParameters: Undefined, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
При выполнении данного метода, драйвер отображает в модальном режиме форму для интерактивного изменения настроек устройства. Допускается отсутствие собственной формы настроек при реализации драйвера. В этом случае, метод возвращает False и ошибку с кодом 1003 (см. ErrorInfo), а клиентское приложение отображает собственную универсальную форму для редактирования настроек устройства.

Параметры:
InputParameters: Undefined
Не используется. Необходимо передавать пустое значение.
ResultData: SafeArray
При отсутствии у драйвера формы настроек, возвращается ErrorInfo с кодом ошибки 1003. Если пользователь отказался от внесения изменений (нажал кнопку «Отмена») возвращается ErrorInfo с кодом ошибки 201. В случае, если пользователь нажал кнопку «ОК» и настройки были успешно применены, содержит значение Undefined
TimeOut: Number
Таймаут ожидания результата выполнения метода. Значение данного параметра игнорируется

Возвращаемое значение:
True – метод выполнен успешно, настройки изменены и применены новые значения настроек. False – изменения настроек не произошло.

CheckHealth

Синтаксис:
CheckHealth (InputParameters: Undefined, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Метод используется для проверки работоспособности устройства. Вызов данного метода может приводить к исполнению длительных операций, поэтому в клиентском приложении не рекомендуется использовать данный метод в обычном цикле использования оборудования. Технология никак не регламентирует действия драйвера при выполнении данного метода. Состав производимых проверок определяется разработчиком драйвера.

Параметры:
InputParameters: Undefined
Не используется. Необходимо передавать пустое значение.
ResultData: Undefined либо SafeArray
Возвращаемые параметры отсутствуют. В случае ошибки выполнения метода, параметр содержит массив ErrorInfo
TimeOut: Number
Таймаут ожидания результата выполнения метода. Значение данного параметра игнорируется

Возвращаемое значение:
True – устройство исправно и работоспособно. False – в противном случае

Enable

Синтаксис:
Enable (InputParameters: Undefined, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Включение устройства. Метод должен быть вызван перед началом использования оборудования, до вызова любого специфического метода. При выполнении данного метода драйвер проводит все возможные проверки на готовность устройства к работе и только в случае их успешного завершения, возвращает результат True. Если драйверу требуется для работы монопольный захват каких-либо ресурсов (например, коммуникационный порт), он должен выполнить его при выполнении данного метода.

Параметры:
InputParameters: Undefined
Не используется. Необходимо передавать пустое значение.
ErrorInfo: SafeArray либо Undefined
Возвращаемые параметры отсутствуют. В случае ошибки выполнения метода, параметр содержит массив ErrorInfo
TimeOut: Number
Таймаут ожидания результата выполнения метода. (см. TimeOut)

Возвращаемое значение:
True – устройство включено и готово к работе. False – в противном случае.

Disable

Синтаксис:
Disable (InputParameters: Undefined, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Выключение устройства. Если драйвером были монопольно захвачены какие-либо ресурсы (например, коммуникационный порт), он должен освободить их при выполнении данного метода.

Параметры:
InputParameters: Undefined
Не используется. Необходимо передавать пустое значение.
ErrorInfo: SafeArray
Возвращаемые параметры отсутствуют. В случае ошибки выполнения метода, параметр содержит массив ErrorInfo

TimeOut: Number
Таймаут ожидания результата выполнения метода. (см. TimeOut)

Возвращаемое значение:
True – устройство успешно выключено. False – в противном случае

5.3. Специфические методы

Перечисленные в данном разделе методы являются специфическими методами типа оборудования. Любой из этих методов может быть вызван только у включенного устройства (см. Enable).

AuthorizeSales

Синтаксис:
AuthorizeSales (InputParameters: SafeArray, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Производит авторизацию платежа на сумму, указанную в Amount. Если параметр CustomerData непустой (карта уже считана внешним считывателем) и DataInputMode рано 0 или 1, драйвер использует данный параметр, как номер карты и не инициирует запрос на считывание карты на Pin-pad. В случае, если авторизация карты была одобрена, возвращает True и в параметрах передает ссылочный номер проведенной транзакции, код авторизации, номер чека и параметры для печати квитанции. Приложение должно сохранять данные параметры в течение смены, на случай отмены платежа, иначе для возврата денежных средств необходимо будет использовать метод AuthorizeRefund, при этом деньги на счет клиента будут зачислены через несколько дней.

Параметры:
InputParameters: SafeArray
Массив входных параметров.
ResultData: SafeArray
Массив возвращаемых данных. В случае ошибки - стандартный массив ErrorInfo. Элемент массива ExtData может содержать необязательный параметр, соответствующий по структуре массиву ReceiptParameters. В этом параметре содержатся данные для печати квитанции. В случае, если метод вернул False и параметр ExtData непустой, приложение должно напечатать текст квитанции.
TimeOut: Number
Таймаут исполнения команды (в секундах). См. TimeOut

Возвращаемое значение:
True – метод выполнен успешно. В противном случае – False

Массив InputParameters

Индекс SafeArray	Имя	Тип	Описание
0	Amount	Number	Сумма оплаты
1	CustomerData	String	Данные, необходимые для идентификации плательщика (см. настройка DataInputMode). Необязательный

Массив ResultData

Индекс SafeArray	Имя	Тип	Описание
0	MaskedCardNumber	String	Маскированный номер карты
1	ReferenceNumber	String	Ссылочный номер проведенной транзакции (RRN). Необходим для операции отмены или возврата
2	ReceiptNumber	String	Номер чека проведенной транзакции. Необходим для операции отмены
3	ReceiptParameters	SafeArray	Таблица параметров квитанции. Таблица настроек состоит из двух полей – Имя параметра (строка) и Значение параметра (произвольный тип). В составе параметров чека может быть полный текст квитанции, сформированный драйвером. В этом случае, он помещается в первый элемент массива.

			В случае же, если драйвер не формирует текст квитанции самостоятельно, первый элемент массива должен содержать пустую строку. Имена параметров должны совпадать с подстановочными полями шаблона квитанции. Структура параметра описана ниже. В случае, если драйвер формирует текст квитанции, печать копий реализуется самим драйвером – т.е. возвращаемый им текст должен содержать необходимое количество копий.
4	AuthorizationCode	String	Код авторизации проведенной транзакции. Необходим для операции отмены или подтверждения

Массив ReceiptParameters

Индекс SafeArray	Имя	Тип	Описание
(0..N,0)	Name	Number	Имя параметра, как оно задано в шаблоне квитанции
(0..N,1)	Value	Number / String / DateTime / Boolean	Значение параметра

AuthorizeRefund

Синтаксис:
AuthorizeRefund (InputParameters: SafeArray, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Производит авторизацию возврата платежа на сумму, указанную в Amount. Если параметр CustomerData непустой (карта уже считана внешним считывателем) и DataInputMode рано 0 или 1, драйвер использует данный параметр, как номер карты и не инициирует запрос на считывание карты на Pin-pad.

Параметры:
InputParameters: SafeArray
Массив входных параметров.
ResultData: SafeArray
Массив возвращаемых данных. В случае ошибки - стандартный массив ErrorInfo. Элемент массива ExtData может содержать необязательный параметр, соответствующий по структуре массиву ReceiptParameters. В этом параметре содержатся данные для печати квитанции. В случае, если метод вернул False и параметр ExtData непустой, приложение должно напечатать текст квитанции.

TimeOut: Number
Таймаут исполнения команды (в секундах). См. TimeOut

Возвращаемое значение:
True – метод выполнен успешно. В противном случае – False

Массив InputParameters

Индекс SafeArray	Имя	Тип	Описание
0	Amount	Number	Сумма оплаты
1	CustomerData	String	Данные, необходимые для идентификации плательщика (см. настройка DataInputMode). Необязательный
2	ReferenceNumber	String	Ссылочный номер (RRN) оригинальной транзакции продажи. Необязательный.

Массив ResultData

Индекс SafeArray	Имя	Тип	Описание
0	MaskedCardNumber	String	Маскированный номер карты
1	ReferenceNumber	String	Ссылочный номер проведенной транзакции (RRN). Необходим для операции отмены

2	ReceiptNumber	String	Номер чека проведенной транзакции. Необходим для операции отмены
3	ReceiptParameters	SafeArray	Таблица параметров квитанции. См. описание метода AuthorizeSales
4	AuthorizationCode	String	Код авторизации проведенной транзакции. Необходим для операции отмены или подтверждения

AuthorizeVoid

Синтаксис:
AuthorizeVoid (InputParameters: SafeArray, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Производит отмену транзакции платежа или возврата платежа. Параметры ReferenceNimber, ReceiptNumber, AuthorizationCode должны быть заполнены соответствующими данными, полученными после выполнения транзакции, которая отменяется. Параметр Amount должен содержать сумму отменяемой транзакции. Часть входных параметров может не использоваться, в зависимости от требований используемой платежной системы.

Параметры:
InputParameters: SafeArray
Массив входных параметров, необходимых для поиска отменяемой транзакции.
ResultData: SafeArray
Массив возвращаемых данных. Содержит один элемент – таблицу параметров квитанции. В случае ошибки - стандартный массив ErrorInfo. Элемент массива ExtData может содержать необязательный параметр, соответствующий по структуре массиву ReceiptParameters. В этом параметре содержатся данные для печати квитанции. В случае, если метод вернул False и параметр ExtData непустой, приложение должно напечатать текст квитанции.

TimeOut: Number
Таймаут исполнения команды (в секундах). См. TimeOut

Возвращаемое значение:
True – метод выполнен успешно. В противном случае – False

Массив InputParameters

Индекс SafeArray	Имя	Тип	Описание
0	Amount	Number	Сумма оплаты
1	ReferenceNumber	String	Ссылочный номер отменяемой транзакции (RRN). Необязательный
2	ReceiptNumber	String	Номер чека отменяемой транзакции. Необязательный
3	AuthorizationCode	String	Код авторизации отменяемой транзакции. Необязательный

Массив ResultData

Индекс SafeArray	Имя	Тип	Описание
0	ReceiptParameters	SafeArray	Таблица параметров квитанции. См. описание метода AuthorizeSales

AuthorizePreSales

Синтаксис:
AuthorizePreSales (InputParameters: SafeArray, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Производит пре-авторизацию платежа на сумму, указанную в Amount (сумма блокируется, но не списывается со счета). Если параметр CustomerData непустой (карта уже считана внешним считывателем) и DataInputMode рано 0 или 1, драйвер использует данный параметр, как номер карты и не инициирует запрос на считывание карты на Pin-pad. В случае, если пре-авторизация карты была одобрена, возвращает True и в параметрах передает ссылочный номер проведенной транзакции и параметры для печати квитанции. Необязательный метод.

Параметры:
InputParameters: SafeArray
Массив входных параметров.
ResultData: SafeArray
Массив возвращаемых данных. В случае ошибки - стандартный массив ErrorInfo. Элемент массива ExtData может содержать необязательный параметр, соответствующий по структуре массиву ReceiptParameters. В этом параметре содержатся данные для печати квитанции. В случае, если метод вернул False и параметр ExtData непустой, приложение должно напечатать текст квитанции.

TimeOut: Number
Таймаут исполнения команды (в секундах). См. TimeOut

Возвращаемое значение:
True – метод выполнен успешно. В противном случае – False

Массив InputParameters

Индекс SafeArray	Имя	Тип	Описание
0	Amount	Number	Сумма оплаты
1	CustomerData	String	Данные, необходимые для идентификации плательщика (см. настройка DataInputMode). Необязательный

Массив ResultData

Индекс SafeArray	Имя	Тип	Описание
0	MaskedCardNumber	String	Маскированный номер карты
1	ReferenceNumber	String	Ссылочный номер проведенной транзакции (RRN). Необходим для операции отмены и подтверждения.
2	ReceiptNumber	String	Номер чека проведенной транзакции. Необходим для операции отмены или подтверждения
3	ReceiptParameters	SafeArray	Таблица параметров квитанции. См. описание метода AuthorizeSales
4	AuthorizationCode	String	Код авторизации проведенной транзакции. Необходим для операции отмены или подтверждения

AuthorizeCompletion

Синтаксис:
AuthorizeCompletion (InputParameters: SafeArray, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Производит подтверждение транзакции пре-авторизации (списывается сумма, заблокированная методом AuthorizePreSales). Параметры ReferenceNimber, ReceiptNumber, AuthorizationCode должны быть заполнены соответствующими данными, полученными после выполнения транзакции AuthorizePreSales.Часть входных параметров может не использоваться, в зависимости от требований используемой платежной системы. Необязательный метод

Параметры:
InputParameters: SafeArray
Массив входных параметров, необходимых для поиска подтверждаемой транзакции.
ResultData: SafeArray
Массив возвращаемых данных. Содержит один элемент – таблицу параметров квитанции. В случае ошибки - стандартный массив ErrorInfo. Элемент массива ExtData может содержать необязательный параметр, соответствующий по структуре массиву ReceiptParameters. В этом параметре содержатся данные для печати квитанции. В случае, если метод вернул False и параметр ExtData непустой, приложение должно напечатать текст квитанции.

TimeOut: Number
Таймаут исполнения команды (в секундах). См. TimeOut

Возвращаемое значение:

True – метод выполнен успешно. В противном случае – False

Массив InputParameters

Индекс SafeArray	Имя	Тип	Описание
0	Amount	Number	Сумма оплаты
1	AuthorizationCode	String	Код авторизации подтверждаемой транзакции. Необязательный
2	ReferenceNumber	String	Ссылочный номер подтверждаемой транзакции (RRN). Необязательный
3	ReceiptNumber	String	Номер чека подтверждаемой транзакции. Необязательный

Массив ResultData

Индекс SafeArray	Имя	Тип	Описание
0	ReceiptParameters	SafeArray	Таблица параметров квитанции. См. описание метода AuthorizeSales

AuthorizeVoidPreSales

Синтаксис:
AuthorizeVoidPreSales (InputParameters: SafeArray, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Производит отмену транзакции пре-авторизации (сумма на счете разблокируется). Параметры ReferenceNimber, ReceiptNumber, AuthorizationCode должны быть заполнены соответствующими данными, полученными после выполнения транзакции AuthorizePreSales. Часть входных параметров может не использоваться, в зависимости от требований используемой платежной системы. Необязательный метод

Параметры:
InputParameters: SafeArray
Массив входных параметров, необходимых для поиска отменяемой транзакции.
ResultData: SafeArray
Массив возвращаемых данных. Содержит один элемент – таблицу параметров квитанции. В случае ошибки - стандартный массив ErrorInfo. Элемент массива ExtData может содержать необязательный параметр, соответствующий по структуре массиву ReceiptParameters. В этом параметре содержатся данные для печати квитанции. В случае, если метод вернул False и параметр ExtData непустой, приложение должно напечатать текст квитанции.

TimeOut: Number
Таймаут исполнения команды (в секундах). См. TimeOut

Возвращаемое значение:
True – метод выполнен успешно. В противном случае – False

Массив InputParameters

Индекс SafeArray	Имя	Тип	Описание
0	Amount	Number	Сумма оплаты
1	AuthorizationCode	String	Код авторизации отменяемой транзакции. Необязательный
2	ReferenceNumber	String	Ссылочный номер отменяемой транзакции (RRN). Необязательный
3	ReceiptNumber	String	Номер чека отменяемой транзакции. Необязательный

Массив ResultData

Индекс SafeArray	Имя	Тип	Описание
0	ReceiptParameters	SafeArray	Таблица параметров квитанции. См. описание метода AuthorizeSales

Settlement

Синтаксис:
Settlement (InputParameters: SafeArray, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Функция сверки итогов дня (закрытия смены эквайринг-терминала). В случае, если данная модель эквайринг-терминала не поддерживает сверку итогов (не нуждается в ней), метод возвращает значение True, не выполняя никаких действий.

Параметры:
InputParameters: Undefined
Не используется. Необходимо передавать пустое значение.
ResultData: SafeArray
Массив возвращаемых данных. Содержит один элемент – таблицу параметров квитанции. В случае ошибки - стандартный массив ErrorInfo
TimeOut: Number
Таймаут исполнения команды (в секундах). См. TimeOut

Возвращаемое значение:
True – метод выполнен успешно. В противном случае – False

Массив ResultData

Индекс SafeArray	Имя	Тип	Описание
0	ReceiptParameters	SafeArray	Таблица параметров квитанции. См. описание метода AuthorizeSales

GetReport

Синтаксис:
GetReport (InputParameters: SafeArray, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Получение текста детального сменного отчета или отчета произвольного вида. Необязательный метод

Параметры:
InputParameters: SafeArray
Массив входных параметров. Содержит один параметр – вид отчета
ResultData: SafeArray
Массив возвращаемых данных. Содержит один элемент – таблицу параметров квитанции. В случае ошибки - стандартный массив ErrorInfo
TimeOut: Number
Таймаут исполнения команды (в секундах). См. TimeOut

Возвращаемое значение:
True – метод выполнен успешно. В противном случае – False

Массив InputParameters

Индекс SafeArray	Имя	Тип	Описание
0	ReportNum	Number	Вид отчета. Виды отчетов определяются моделью Эквайринг-терминала, банковским ПО. Если ровно 0 – печатается отчет по умолчанию. Вид отчета по умолчанию может быть задан в настройках драйвера.

Массив ResultData

Индекс SafeArray	Имя	Тип	Описание
0	ReceiptParameters	SafeArray	Таблица параметров квитанции (отчета). См. описание метода AuthorizeSales

GetReceiptCopy

Синтаксис:
GetReceiptCopy (InputParameters: SafeArray, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Получение текста копии квитанции по её номеру. Возможно только внутри открытой смены. Необязательный метод

Параметры:
InputParameters: SafeArray
Массив входных параметров. Содержит один параметр – номер квитанции
ResultData: SafeArray
Массив возвращаемых данных. Содержит один элемент – таблицу параметров квитанции. В случае ошибки - стандартный массив ErrorInfo
TimeOut: Number
Таймаут исполнения команды (в секундах). См. TimeOut

Возвращаемое значение:
True – метод выполнен успешно. В противном случае – False

Массив InputParameters

Индекс SafeArray	Имя	Тип	Описание
0	ReceiptNumber	String	Номер квитанции. Если пустая строка – возвращается копия последней квитанции

Массив ResultData

Индекс SafeArray	Имя	Тип	Описание
0	ReceiptParameters	SafeArray	Таблица параметров квитанции. См. описание метода AuthorizeSales

GetBalance

Синтаксис:
GetBalance (InputParameters: SafeArray, ResultData: SafeArray, TimeOut: Number): Boolean

Описание:
Получение остатка доступных средств (баланса) на счете карты. Необязательный метод

Параметры:
InputParameters: SafeArray
Массив входныех параметров.
ResultData: SafeArray
Массив возвращаемых данных. В случае ошибки - стандартный массив ErrorInfo
TimeOut: Number
Таймаут исполнения команды (в секундах). См. TimeOut

Возвращаемое значение:
True – метод выполнен успешно. В противном случае – False

Массив InputParameters

Индекс SafeArray	Имя	Тип	Описание
0	CustomerData	String	Данные, необходимые для идентификации плательщика (см. настройка DataInputMode). Необязательный
1	Mode	Number	Способ расчета баланса: 0 – Текущий баланс без учета кредита и без учета ограничений. Сумма на счету; 1 – Баланс с учетом глубины кредита, без учета ограничений транзакций; 2 – Баланс с учетом глубины кредита, с учетом ограничений транзакций. Максимально возможная

			сумма транзакции; Необязательный.

Массив ResultData

Индекс SafeArray	Имя	Тип	Описание
0	Amount	Number	Сумма баланса
1	ReceiptParameters	SafeArray	Таблица параметров квитанции. См. описание метода AuthorizeSales

6. Функции обратного вызова

В этом разделе описаны функции, подлежащие обязательной реализации в менеджере управления оборудованием любого клиентского приложения. Все функции предназначены для вызова их драйверами устройств.
При вызове метода Init, в его параметрах передается ссылка на интерфейс IDispatch приложения-клиента. Используя эту ссылку, драйвер может вызывать функции обратного вызова, реализованные в приложении.
Для удаленно используемого оборудования вызовы функций будут перенаправляться удаленным клиентским приложениям, через сетевой транспорт.

GetAppProperty

Синтаксис:
GetAppProperty (SourceID: String, Name: String, Value: String): Boolean

Описание:
Функция предназначена для получения свойств системы и приложения-клиента.

Параметры:
SourceID: String (GUID)
Идентификатор устройства, сгенерировавшего событие
Name: String
Наименование требуемого свойства. Допустимые значения:
- «ModelsFolder» - Путь к каталогу моделей оборудования
- «DevicesFolder» - Путь к каталогу экземпляров оборудования
- «Language» - Язык локализации приложения

Value: String
В этом параметре возвращается значение свойства. Для свойства «Language» возвращается двухсимвольный идентификатор языка («ru», «en», «ua» и т. д.)

Возвращаемое значение:
True – функция выполнена успешно, указанное свойство найдено. False – в противном случае

TaskState

Синтаксис:
TaskState (SourceID: String, Percent: Number, Description: String): Boolean

Описание:
Данная функция предназначена для использования только при выполнении драйвером неопределенно-длительных операций (печать большого количества этикеток, авторизация безналичного платежа и т. п.). Функция служит для уведомления приложения о ходе выполнения текущего метода и запроса на продление таймаута его исполнения. Если менеджер вернул True, отсчет таймаута выполнения метода необходимо начать заново. Если менеджер вернул False, драйвер должен прервать выполнение текущего метода. По умолчанию, менеджер оборудования возвращает True, позволяя драйверу продолжать выполнение метода. Рекомендуется вызывать эту функцию с интервалом от 1/10 до половины от значения таймаута исполнения метода. Это позволит приложению отображать на экране информацию о ходе выполнения команды (прогресс-бар, текст в строке стостояния и др.). Не рекомендуется вызывать данную функцию чаще одного раза в секунду, во избежание создания лишней нагрузки на систему.
В процессе выполнения транзакций драйвер может вызывать функцию TaskState, информируя приложение о текущей стадии выполнения транзакции, используя параметр функции Description. Приложение должно отображать полученный текст на экране (в строке состояния, отделном

окне и т. п.). Примеры текста: «Установка соединения», «Обработка запроса». Использование TaskState в данном случае крайне рекомендуется в связи с тем, что обмен данными с хостом, как правило – операция достаточно длительная.

Параметры:

SourceID: String

Идентификатор устройства, сгенерировавшего событие

Percent: Number

Значение от 0 до 100 – приблизительная степень выполнения текущего метода в процентах.

Description: String

Текстовое описание текущего статуса выполнения метода. Например: «Обработано 30% полученных данных».

Возвращаемое значение:

True – продолжение выполнения текущего метода разрешено. False – драйверу следует прервать выполнение метода

ErrorEvent

Синтаксис:

ErrorEvent (SourceID: String, ErrorName: String, Description: String, ExtData: SafeArray): Boolean

Описание:

Функция предназначена для передачи приложению информации о возникновении ошибочного состояния или аварийной ситуации в устройстве. Недопустим вызов данной функции во время выполнения драйвером устройства какого-либо метода. Клиентское приложение при вызове данной функции должно любым доступным образом сообщить пользователю о возникшей ошибке.

Параметры:

SourceID: String (GUID)

Идентификатор устройства, сгенерировавшего событие

ErrorName: String

Строка с наименованием типа ошибки. Типы ошибок регламентируются на уровне типов оборудования

Description: String

Строка с кратким описанием возникшей ошибки или аварийной ситуации в устройстве

ExtData: SafeArray либо Undefined

Массив, содержащий дополнительные данные об ошибке. Структура массива определяется типом ошибки и типом оборудования. Необязательный параметр.

Возвращаемое значение:

True – вызов обработан успешно. False – в противном случае

MessageDlg

Синтаксис:

MessageDlg (SourceID: String, Message: String, Type: Number, Buttons: Number, Timeout: Number): Number

Описание:

Функция предназначена для отображения приложением на экране формы диалога с произвольным текстом и набором кнопок. Текст сообщения диалога и набор кнопок задается драйвером устройства в параметрах функции. Функция возвращает индекс нажатой кнопки. Если в течение заданного таймаута пользователь не нажал ни одну из кнопок, функция возвращает 0. Так как данная функция вызывается синхронно и приводит к выводу модального диалога, это может привести к полному блокированию системы на время ожидания реакции пользователя.

Данную функцию рекомендуется использовать, если это необходимо, только в процессе выполнения драйвером какого-либо метода.

Параметры:

SourceID: String (GUID)

Идентификатор устройства, сгенерировавшего событие

Message: String

Текст сообщения или вопроса, обращенного к пользователю

Type: Number

Задает тип сообщения. Допустимые значения: 0 – для выдачи предупреждения; 1 – для вывода информационного сообщения; 2 – вопрос; 3 – ошибка. Клиентское приложение может использовать значение данного параметра для определения стиля оформления диалогового окна (например, снабжая его соответствующими пиктограммами)

Buttons: Number

Определяет количество кнопок диалога и их функциональное назначение. Допустимые значения:

1 - ОК (OK)
2 - ДА+НЕТ (YES+NO)
3 - ДА+НЕТ+ОТМЕНА (YES+NO+CANCEL)
4 - ПОВТОРИТЬ+ОТМЕНА (RETRY+CANCEL)
5 - ОК+ОТМЕНА (OK+CANCEL)
6 - ПОВТОРИТЬ+ОТМЕНА+ПРОПУСТИТЬ (RETRY+CANCEL+IGNORE)

Timeout: Number

Максимальное время ожидания реакции пользователя в секундах. Значение параметра должно быть больше нуля.

Возвращаемое значение:

Определяет нажатую пользователем кнопку. Допустимые значения:

1 - ДА (YES)
2 - НЕТ (NO)
3 - ОК (OK)
4 - ОТМЕНА (CANCEL)
5 - ПОВТОРИТЬ (RETRY)
6 - ПРОПУСТИТЬ (IGNORE)

Если функция вернула 0, значит пользователь за время таймаута не нажал ни одну из кнопок.

InputDataDlg

Синтаксис:

InputDataDlg (SourceID: String, Caption: String, DefaultData: String, Description: String, Mask: String, Password: Boolean, Timeout: Number): String

Описание:

Функция предназначена для организации интерактивного ввода дополнительных данных пользователем во время выполнения метода драйвером. Функция возвращает строку с введенными пользователем данными, либо пустую строку в случае если пользователь отказался от ввода или истек таймаут отображения диалога. Функцию следует использовать только в тех ситуациях, когда все необходимые данные невозможно сразу передать во входных параметрах метода драйвера (например, требуется ввод дополнительного параметра метода, не предусмотренного данной технологией). Так как данная функция вызывается синхронно и приводит к выводу модального диалога, это может привести к полному блокированию системы на время ожидания реакции пользователя.

Параметры:

SourceID: String (GUID)

Идентификатор устройства, сгенерировавшего событие

Caption: String

Краткий текст заголовка для формы диалога. Пример: «Введите сумму:»

DefaultData: String

Строка со значением по умолчанию или пустая строка, если такового нет. При открытии формы диалога, поле ввода должно быть заполнено значением данного параметра.

Description: String

Детальное описание запрашиваемой информации и порядка ее ввода. Переданный в данном параметре текст, приложение может отобразить рядом с полем ввода или в подсказке к нему. Необязательный для заполнения параметр (можно передать пустую строку)

Mask: String

Маска для поля ввода. Необязательный параметр. В строке маски допустимо использование следующих специальных символов:

- ! - любой введенный символ преобразуется в верхний регистр;
- 9 - допустимо ввести произвольный символ цифры;
- # - допустимо ввести произвольный символ цифры или - (знак минус) или + (знак плюс) или пробел;
- N - допустимо ввести любые алфавитно-цифровые символы (буквы или цифры);

- U - допустимо ввести любые алфавитно-цифровые символы (буквы или цифры) и любой введенный символ преобразуется в верхний регистр;
- X - допустимо ввести произвольный символ латинского алфавита;
- @ – допустимо ввести любые алфавитно-цифровые символы (буквы или цифры) в верхнем регистре или пробел.
- C – Выбор строки из списка. В параметре DefaultData передается многострочная строка, разделенная символами CR/LF. Приложение визуализирует в форме диалога данную строку как список строк, предлагая пользователю выбрать одну из них или несколько. Выбранные строки передаются драйверу в возвращаемом значении (если выбрано больше одной строки, они разделяются символами CR/LF). Маска «C» маской как таковой не является, она определяет способ ввода данных – выбор из списка.

При помещении значения из поля ввода с маской в текстовый реквизит, связанный с этим полем ввода, происходит следующее преобразование: на тех позициях, где в маске стоит символ "@", а в строке пробел – пробел удаляется. Если в маске из специальных символов используются только символы "@", то все символы текста, соответствующие символам маски, не являющимся специальными символами, удаляются после последнего непустого блока из символов "@". Например, при маске "@@.@@.@@." текст "41. 2. ." преобразуется в "41.2".
Для того, чтобы использовать в маске один из специальных символов, нужно использовать перед ним символ "\".

Password: Boolean

Если равно True, поле ввода предназначено для ввода пароля. При вводе пароля, все вводимые символы подменяются при отображении на экране символом пароля (обычно «*» или «•»), параметр *Mask* при этом игнорируется.

Timeout: Number

Максимальное время ожидания реакции пользователя в секундах. Значение параметра должно быть больше нуля.

Возвращаемое значение:

Введенные пользователем данные, либо пустая строка в случае, если пользователь отказался от ввода или истек таймаут ожидания

WriteLog

Синтаксис:

WriteLog (SourceID: String, Event: String, Text: String, Type: Number): Boolean

Описание:

Функция предназначена для передачи диагностической и отладочной информации приложению-клиенту. Приложение должно сохранять данную информацию в журнале (лог-файле). Данную функцию можно вызывать в любой момент – на усмотрение разработчика драйвера. Рекомендуется использовать её для диагностики проблем и ошибок при работе с устройством. Если настройка драйвера EventLogEnabled имеет значение False, функция вызываться не должна.

Параметры:

SourceID: String (GUID)

Идентификатор устройства, сгенерировавшего событие

Event: String

Наименование события. Произвольная строка. По наименованию событий можно фильтровать/группировать записи в журнале.

Text: String

Описание события. Произвольная строка

Type: Number

Тип события. Допустимые значения: 0 – Информация; 1 – Ошибка; 2 - Отладка

Возвращаемое значение:

True – функция выполнена успешно, информация записана приложением в журнал. False – в противном случае