1. [Установка соединения](#соединение)
   1. Установка соединения с ЭККР

1. [Формирование чеков](#Формчеков)
   1. Открытие чека
   2. Открытие чека возврата
   3. Переключение текущего чека в режим “возврат товара”
   4. Определение номера возвратного чека
   5. Ввод фискальных данных чека
   6. Дополнительная информация о товаре (услуге)
   7. Открытие новой группы фискальных позиций в пределах чека
   8. Закрытие группы фискальных позиций в пределах чека
   9. Служебная информация в чеке
   10. Отмена служебной информации в чеке
   11. Добавление оплаты в фискальный чек
   12. Указание типа выплаты
   13. Закрытие чека
   14. Ввод данных по слипу(после транзакции по платежному терминалу)

1. [Служебные операции](#Служоперации)
   1. Служебное внесение-изъятие наличных средств
   2. Открытие новой смены без регистрации оборотов
2. [Графика, QR, штрих-коды](#QRштрихкоды)
   1. Штрихкодовые символы «Код128» или «EAN-13» в чеке
   2. Загрузка QR-кода
   3. Печать QR-кода
   4. Загрузка логотипа из файла в ЭККА
   5. Отмена печати логотипа на чеках

1. [Печать отчетов](#печатьотчетов)
   1. Реализация товаров и услуг в разрезе артикулов
   2. Примененные скидки и надбавки
   3. Периодический отчет ФП (по номерам)
   4. Печать X-отчёта
   5. Печать Z-отчёта
   6. Периодический отчет ФП (по датам)
   7. Состояние аппаратуры

1. [Управление устройствами](#упрустройствами)
   1. Выдача информации на выносной индикатор
   2. Управление работой обрезчика чековой ленты и звуковым сигналом
   3. Прогон чековой ленты
   4. Открытие кассового ящика
2. [Настройка параметров](#Настрпараметров)
   1. Определение режима печати информации о наложенных налогах
   2. Управление определением налогооблагаемого оборота при применении двух налогов
   3. Программирование необязательной заключительной строки на чеке
   4. Программирование необязательной заключительной строки на чеке
   5. Программирование нескольких необязательных заключительных строк на чеке
   6. Программирование заголовочной информационной строки на чеке
   7. Программирование мнемоники торгового отдела
   8. Установка наименования налога
   9. Программирование схемы налогообложения
   10. Управление печатью информации о скидках-надбавках в чеке
   11. Установка наименования итогов по скидкам-надбавкам в чеке
   12. Определение набора служебных сообщений на чеке
   13. Управление видом представления фискальной информации на чеке
   14. Установка режима «построчной» печати чека
   15. Корректировка часов реального времени
   16. Корректировка часов реального времени – плюс 1 час
   17. Корректировка часов реального времени – минус 1 час

1. [Информация](#Информация)
   1. Запрос регистра учета реализации артикула
   2. Определение состояния фискальной смены ЭККА
   3. Запрос информации о движении средств по кассе
   4. Запрос текущего состояния дневных фискальных регистров.
   5. Запрос номеров чеков, документов, идентификатора транзакции
   6. Запрос даты-времени часов реального времени ЭККР
   7. Запрос списка запрограммированных схем налогообложения
   8. Запрос внутреннего состояния ЭККР
   9. Время до блокировки ЭККА в часах
   10. Проверка состояния канала связи «Внешнее Устройство - ЭККР»
   11. Запрос состояния батареи
   12. Запрос количества пакетов данных, ожидающих отправку на сервер эквайера

1. [Бонусы, карты](#Бонусыкарты)
   1. Прием платежа на бонусный счет/подарочный сертификат/клубную карту
   2. Списание средств с бонусного счета/подарочного сертификата/клубной карты
2. [Авансовые платежи](#Авансовыеплатежи)
   1. Открытие чека “заказа”. Прием авансового платежа.
   2. Печать описания одной заказываемой позиции товара(услуги) в пределах чека заказа
   3. Регистрация оборота приема аванса и закрытие чека заказа
   4. Открытие чека “выполнения заказа”. Расчет по авансовому платежу
   5. Учет авансового платежа (уменьшение оборота реализации)
   6. Открытие чека “возврата аванса”. Расчет по авансовому платежу.
   7. Регистрация оборота возврата полной суммы ранее принятого аванса.
   8. Учет сумм отпущенных товаров-услуг при возврате аванса
3. [Печать билетов](#Печатьбилетов)
   1. Открытие чека продажи билета (багажной квитанции)
   2. Открытие чека возврата билета (багажной квитанции)
   3. Ввод описания маршрута
   4. Ввод сумм составляющих стоимости билета (квитанции)
   5. Ввод суммы удержания по билету
   6. Закрытие чека билета (багажной квитанции)
4. [Прочее](#Прочее)
   1. Принудительно передать данные на эквайер

Общая информация

MariaAndroidAPI — API библиотека, предоставляемая для разработки приложений на платформе Android

Предназначена для работы с ЭККА Мария-301MTM T7-T11, Мария-M304.

Включает : Эмулятор для тестирования методов библиотеки и непосредственно библиотеку API

Функциональное назначение: инициация и обеспечение транспортного уровня между пользовательским приложением и ЭККА. Позволяет упростить разработку управляющего приложения в виду того, что большинство операций по обработке и формированию чеков обеспечивают ресурсы библиотеки.

Обеспечивает возможность подключения к ЭККА посредством Bluetooth или USB.

Минимальные требования к ПО: **Android 4.2-4.2.2** – Jelly Bean

|  |
| --- |
| **Установка соединения** |

**Установка соединения с ЭККР**

***@параметр*** *<CashierName> до 10 или более 10 символов идентификатор кассира****@параметр*** *<*EkkaPassword *> 10 символов текущий пароль****@результат*** *0/1 успех/ошибка, генерирует исключение MariaException*

**int** Init(String CashierName, String EkkaPassword) **throws** MariaException

|  |
| --- |
| **Формирование чеков** |

**Открытие чека**

*Для упрощения алгоритмов функционирования ПО ВУ перед открытием нового  
чека рекомендуется вызов функции AbortCheck для гарантированной отмены не  
закрытого документа или не закрытого чека после аварийных ситуаций в канале  
связи или в ПО ВУ – отпадает необходимость контролировать состояние чека  
(открыт/закрыт) по «признаку ожидаемой команды создания документов».****@параметр*** *departmentName String до 35-ти символов идентификатор наименование)*

*торгового отдела.****@результат*** *0/1 успех/ошибка, генерирует исключение MariaException***int** OpenCheck(String departmentName) **throws** MariaException

**Открытие чека возврата**

***@параметр departmentName*** *название торгового отдела****@параметр NumberCheckReturn*** *№ чека, по которому производится возврат  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)***int** OpenReturnCheck(String departmentName, String NumberCheckReturn) **throws** MariaException

**Переключение текущего чека в режим “возврат товара”**

*После регистрации продаж, чек переключается на регистрацию возврата(”Смешаный чек”). @результат**0/1 успех/ошибка*

**int** SetCurrentCheckForReturn()

**Определение номера возвратного чека**

*Имеет значение для наборов сообщений '0','1' (SetServiceMessagesKit). Переданные в  
команде номера будут отпечатаны за словами "ПОВЕРНЕННЯ ПО ЧЕКУ N" и на  
следующих двух строках (в зависимости от длины numbersList). Используйте команду  
перед открытием чека возврата, т.е. до команды openCheck.****@параметр numbersList*** *String 1 от 1-го до 86-ти символов список номеров*

*чеков.****@результат*** *0/1 успех/ошибка, генерирует исключение MariaException*

**int** SetReturnCheckNumber(String numbersList) **throws** MariaException

**Ввод фискальных данных чека**

*Применяется при открытой вызовом функции OpenCheck*

***@параметр nameTov*** *до 24 символов наименование товара****@параметр count*** *количество единиц товара****@параметр сost*** *стоимость единицы товара (услуги)****@параметр isDivisible*** *признак делимости, где'0' - делимый, при этом значение*

*<count> интерпретируется как xxx,xxx.  
 '1' - неделимый, при этом значение <count>*

*интерпретируется как xxxxxx.****@параметр tax1*** *1 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается****@параметр tax2*** *2 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается****@параметр goodsNumber*** *9-ти символьный номер артикула****@результат*** *0/1/2/3 успех/ошибка протокола/чек не открыт/неправильные*

*параметры (генерирует исключение MariaException)*

**int** FiscalLine(String nameTov, **double** count, **double** сost, **int** isDivisible, **int** tax1,

**int** tax2, **int** goodsNumber) **throws** MariaException

**Ввод фискальных данных чека**

*Применяется при открытой вызовом функции OpenCheck*

***@параметр nameTov*** *до 24 символов наименование товара****@параметр count*** *количество единиц товара****@параметр сost*** *стоимость единицы товара (услуги)****@параметр isDivisible*** *признак делимости, где'0' - делимый, при этом значение*

*<count> интерпретируется как xxx,xxx.  
 '1' - неделимый, при этом значение <count>*

*интерпретируется как xxxxxx.****@параметр tax1*** *1 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается****@параметр tax2*** *2 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается*

***@параметр goodsNumber*** *9-ти символьный номер артикула****@параметр isDiscountOrExtraCharge*** *0/1/2 – нет скидки(надбавки)/скидка/надбавка****@параметр nameDiscount*** *название скидки****@параметр discountOrExtraChargeSum*** *cумма скидки****@результат*** *0/1/2/3 успех/ошибка протокола/чек не открыт/неправильные*

*параметры (генерирует исключение MariaException)*

**int** FiscalLine(String nameTov, **double** count, **double** cost, **int** isDivisible, **int** tax1, **int**

tax2, **int** goodsNumber, **int** isDiscountOrExtraCharge, String nameDiscount, **double** discountOrExtraChargeSum) **throws** MariaException

**Дополнительная информация о товаре (услуге)**

*Переданный в качестве info текст будет отпечатан во время исполнения  
следующей команды FiscalLine*

***@параметр info*** *String до 86-ти символов текстовой информации.****@результат*** *0/1 успех/ошибка, генерирует исключение MariaException***int** LongName(String info) **throws** MariaException

**Открытие новой группы фискальных позиций в пределах чека**

*Применяется при не открытой группе в любой момент после открытия чека  
вызовом функции openCheck, независимо от количества уже созданных фискальных позиций в чеке.  
На чеке будет отпечатано наименование группы позиций, указанное в groupName.****@параметр groupName*** *String от 1 до 21-го символа наименование новой группы*

*позиций.****@результат*** *0/1 успех/ошибка(генерирует исключение MariaException)***int** OpenGoodsGroup(String groupName) **throws** MariaException

**Закрытие группы фискальных позиций в пределах чека**

*Применяется при открытой вызовом функции* OpenGoodsGroup *группе  
 в любой момент после создания необходимого количества фискальных позиций.  
 На чеке будет отпечатана итоговая информация по закрываемой группе:  
 - итог по скидкам,  
 - итог по надбавкам,  
- сумма операций с учетом скидок-надбавок.****@параметр discountsTotalsName*** *String 22 символа - наименование итога по скидкам в*

*пределах закрываемой группы.****@параметр surchargesTotalsName*** *String 22 символа - наименование итога по надбавкам в*

*пределах закрываемой группы.****@результат*** *0/1 успех/ошибка(генерирует исключение MariaException)***int** CloseGoodsGroup (String discountsTotalsName, String surchargesTotalsName) **throws**

MariaException

**Служебная информация в чеке**

*Каждый следующий вызов с одинаковым значением параметра position  
программирует следующую по счету сверху вниз строку для печати.  
Информация, заданная вызовом* FreeTextLine*, сохраняется до того момента,  
пока она не будет отпечатана либо пока не будет выполнена команда* FreeTextLine*.****@параметр position*** *int - 1 символ 0/1 признак расположения текста*

*до/после фискальной информации.****@параметр fontStyle*** *int - 1 символ [0..3] признак печати строки*

*модифицированным шрифтом:  
 “0” – нет модификации,  
 “1” - c удвоенной шириной,  
 “2” – с удвоенной высотой,  
 “3” - c удвоенными шириной и высотой.****@параметр text*** *String - до 43-х символов собственно строка для*

*печати.****@результат*** *0/1 успех/ошибка, генерирует исключение MariaException***int** FreeTextLine(**int** position, **int** fontStyle, String text) **throws** MariaException

***Отмена служебной информации в чеке***

*Очищает массивы и счетчики строк, подготовленных вызовами* FreeTextLine***@результат*** *0/1 успех/ошибка, генерирует исключение MariaException***int** ClearFreeTextLines() **throws** MariaException

**Добавление оплаты в фискальный чек**

***@параметр cash*** *сумма наличными****@параметр noncash3*** *сумма безналичных 3****@параметр noncash2*** *сумма безналичных 2****@параметр noncash1*** *сумма безналичных 1****@параметр NameTransaction*** *описание  
@результат**0/1 успех/ошибка*

**int** AddPayment(**double** cash, **double** noncash3, **double** noncash2, **double** noncash1, String NameTransaction)

**Добавление оплаты в фискальный чек**

***@параметр sum*** *сумма****@параметр paymentType*** *0/1/2/3 наличные/безналичные 1/безналичные2/безналичные3  
@результат**0/1 успех/ошибка*

**int** AddPayment ( **double** sum , **int** paymentType )

**Указание типа выплаты**

*Для указания типа выплаты фискальной позиции секции возврата.  
По умолчанию для каждой фискальной позиции секции возврата  
установлен тип выплаты «0». При необходимости изменить тип выплаты  
непосредственно перед FiscalLine*

***@параметр paymentID*** *int 1 символ [0..3] идентификатор типа выплаты:  
 0 – возврат товара  
 1 – рекомпенсация услуги  
 2 – прием ценностей под залог  
 3 – выплата выигрыша  
  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)***int** SetPaymentType(**int** paymentID) **throws** MariaException

**Закрытие чека**

***@результат*** *0/1 успех/ошибка, генерирует исключение MariaException*

**int** CloseCheck()**throws** MariaException

***Организация работы с чеками*:**

* Вариант 1.

**Открытие чека:** OpenCheck(…);

**Печать штрих-кода:**PrintBarcode(…);

**Ввод фискальных данных:** FiscalLine(…);

**Дополнительная информация о товаре (услуге):** LongName(…);

**Ввод фискальных данных:** FiscalLine(…);

**Ввод данных по слипу(*если используется платежный терминал*):** ReceiptPaymantTerminal(…)

**Добавление оплаты в фискальный чек:** AddPayment(…);

**Закрытие чека:** CloseCheck();

* Вариант 2.

**Открытие чека:** OpenCheck(String);

**Открытие новой группы фискальных позиций:** OpenGoodsGroup(…);

**Ввод фискальных данных:** FiscalLine(…);

**Ввод фискальных данных:** FiscalLine(…);

**Закрытие группы фискальных позиций:** CloseGoodsGroup(…);

**Ввод фискальных данных:** FiscalLine(…);

**Ввод фискальных данных:** FiscalLine(…);

**Закрытие чека:** CloseCheck();

* Вариант 3.

**Открытие чека:** OpenCheck(…);

**Формируем QR-код:** LoadQR(…)

**Печать QR-кода:** PrintQR(…)

**Ввод фискальных данных:** FiscalLine(…);

**Ввод фискальных данных:** FiscalLine(…);

**Переключаем текущий чек в режим “возврат товара”:** SetCurrentCheckForReturn();

**Ввод фискальных данных:** FiscalLine(…);

**Ввод фискальных данных:** FiscalLine(…);

**Закрытие чека:** CloseCheck();

* Вариант 4.

**Открытие чека возврата:** OpenReturnCheck (…);

**Указание типа выплаты:** SetPaymentType(…);

**Ввод фискальных данных:** FiscalLine(…);

**Ввод фискальных данных:** FiscalLine(…);

**Закрытие чека:** CloseCheck()

***Комбинирование команд и вариантов при формировании чека может быть множество. Рекомендации и ограничения детально описаны в протоколе информационного обмена РРО*** ([ЗАО"Резонанс"](http://support.ekka.com.ua/)).

**Отмена чека**

*Используйте эту команду для отмены операции регистрации продажи после  
вызова openCheck и до вызова closeCheck или для гарантированной отмены не  
закрытого чека после аварийных ситуаций в канале связи или в ПО ВУ.****@результат*** *0/1 успех/ошибка, генерирует исключение MariaException*

**int** AbortCheck() **throws** MariaException

**Создание копии чека**

*После успешного создания оригинала фискального чека доступна команда  
создания копии. Допустимо отпечатать только одну копию****@результат*** *0/1 успех/ошибка, генерирует исключение MariaException*

**int** CheckCopy() **throws** MariaException

***Печать “нулевого чека”***

***@результат*** *0/1 успех/ошибка, генерирует исключение MariaException*

**int** NullCheck() **throws** MariaException

***Ввод данных по слипу после успешной транзакции по платежному терминалу.***

*Вызывается перед CloseCheck() с безналичной формой оплаты (AddPayment(…)).Заполняется данными от платежного терминала*

***@параметр*** *LenIdAcquirer, //2 символа цифры (числовое значение от 0 до 99)*

*наличие и длина поля «ID эквайра, торгiвця»****@параметр*** *IdAcquirer, //от 1 до 99 символов в соответствии со значением*

*«ID эквайра, торгiвця».****@параметр*** *LenIdDevice, //2 символа цифры (числовое значение от 0 до 32) наличие*

*и длина поля «ID пристрою»****@параметр*** *IdDevice, //от 1 до 32 символов в соответствии со значением <п3>*

*«ID пристрою».****@параметр*** *LenTypeOperation, //2 символа цифры (числовое значение от 0 до 32)*

*наличие и длина поля «Вид операцiї»****@параметр*** *TypeOperation, //от 1 до 32 символов в соответствии со значением*

*«Вид операцiї».****@параметр*** *LenPaymentDevice, //2 символа цифры (числовое значение от 0 до 32)*

*наличие и длина поля «Електронного Платiжного Засобу»****@параметр*** *PaymentDevice, //от 1 до 32 символов в соответствии со значением*

*«ЕПЗ».****@параметр*** *LenCodeAvtorization, //2 символа цифры (числовое значение от 0 до 32) наличие*

*и длина поля «Код авторизации»****@параметр*** *CodeAvtorization, //от 1 до 32 символов в соответствии со значением <п9>*

*«Код авторизации».****@параметр*** *SummKomission, //10 символов цифры сумма комиссии в копейках****@параметр*** *CashierSignature, //1 символ «1» или «0» присутствует или нет место для*

*подписи кассира.****@параметр*** *OwnerSignature //1 символ «1» или «0» присутствует или нет место для*

*подписи владельца «ЕПЗ».)*

***@результат*** *0/1 успех/ошибка, генерирует исключение MariaException*

**int** ReceiptPaymantTerminal(**int** LenIdAcquirer, String IdAcquirer, **int** LenIdDevice,

String IdDevice,**int** LenTypeOperation, String TypeOperation,

**int** LenPaymentDevice, String PaymentDevice,

**int** LenCodeAvtorization, String CodeAvtorization,

**double** SummKomission,**boolean** CashierSignature,

**boolean** OwnerSignature) **throws** MariaException

|  |
| --- |
| **Служебные операции** |

***Служебное внесение-изъятие наличных средств***

*Соответствующим образом изменяются внутренние регистры с информацией о  
движении средств – увеличиваются суммы внесения-изъятия и корректируется исходящий остаток.  
Проверяется корректность (достаточность) исходящих остатков для операций изъятия.  
В случае нехватки средств операция отменяется с сообщением об  
ошибке ‘SOFTNEED’.  
В случае переполнения регистров учета по операции внесения - операция  
отменяется с сообщением об ошибке ‘SOFTOVER’.*

***@параметр movedirection*** *0/1 - изъятие/внесение****@параметр cashsum*** *int - 10-ти разрядная сумма внесения-изъятия****@параметр description*** *String (необязательный параметр)- до 120 символов*

*текстовое описание операции  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)***int** MoveCash(**int** movedirection, **double** cashsum, String description) **throws** MariaException

***Открытие новой смены без регистрации оборотов***

*Увеличивает номер Z-отчета на 1 и сбрасывает признак выполненного  
Z-отчета.  
Команда доступна к выполнению только на зарегистрированном ФМ в состоянии  
 «Z-отчет выполнен». После применения команды возможна регистрация оборотов.  
 Применяется с целью записи в ФП и печати нового Z-отчета с нулевыми  
 суммами по вызову createZReport.  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** OpenBusinessDay() **throws** MariaException

|  |
| --- |
| **QR, штрих-коды** |

***Штрихкодовые символы «Код128» или «EAN-13» в чеке***

*После печати графического образа кода будет автоматически отпечатано  
символьное содержимое кодированной информации в виде последовательности  
пар символов. Знаки с номерами 101 и 103 отображаются в виде «>A». Знаки  
с номерами 100 и 104 отображаются в виде «>B». Знак с номером 102 отоб-  
ражаются в виде «>F». Знак с номером 105 отображаются в виде «>C». Кон-  
трольный и знак «STOP» не отображаются. Остальные информационные знаки с  
номерами от 0 до 99 отображаются соответственно парами символов от «00»  
до «99». Применение команды создает строку с графической информацией плюс от 1-й  
до 2-х строк текстовой информации (содержимое кода) в соответствии с  
требованиями и ограничениями, аналогичными команде ‘TEXT’.****@параметр position*** *int - 1 символ 0/1 признак расположения текста до/после*

*фискальной информации.****@параметр codeType*** *String - 1 символ указатель на вид кода. Символ*

*‘E’(69dec) – печать EAN13,  
 любые другие символы – печать «Код128» по ДСТУ 3776-98.****@параметр code*** *String - от 4-х до 17-ти символов – номера знаков*

*Код128. Первый символ* ***обязательно*** *должен находиться в*

*диапазоне [103dec..105dec] – один из  
 3-х возможных стартовых знаков в соответствии с ДСТУ*

*3776-98. Остальные символы - с кодами из диапазона  
 [0dec..102dec]. Контрольный знак и знак «STOP» будут*

*добавлены автоматически.  
 Таким образом, строка символов code начинается с*

*требуемого знака «START» (CODEA, CODEB или CODEC), вслед*

*за которым может следовать от 3-х  
 до 16-ти информационных знаков кода (без контрольного*

*знака и без знака «STOP»).  
 После печати графического образа кода будет*

*автоматически отпечатано  
 символьное содержимое кодированной информации в виде*

*последовательности пар символов. Знаки с номерами 101 и*

*103 отображаются в виде «>A». Знаки  
 с номерами 100 и 104 отображаются в виде «>B». Знак с*

*номером 102 отображаются в виде «>F». Знак с номером 105*

*отображаются в виде «>C». Контрольный и знак «STOP» не*

*отображаются. Остальные информационные знаки с номерами*

*от 0 до 99 отображаются соответственно парами символов*

*от «00» до «99».  
 Применение команды создает строку с графической*

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** PrintBarcode(**int** position, String codeType, String code) **throws** MariaException

***Загрузка QR-кода***

***@параметр scale*** *множитель масштаба изображения 2..10*

***@параметр LevelCorrection*** *Символ идентификатор уровня коррекции ошибок  
 [‘L’, ‘M’, ‘Q’, ‘H’]. Значение параметра вне указанного*

*множества автоматически заменяется на ‘L’* ***@параметр StringForQR*** *Строка данных до 240 символов для кодирования  
 @результат возвращает 0-успех,1-ошибки протокола, 2-неверные параметры*

**int** LoadQR (**int** scale, String ***LevelCorrection,*** String StringForQR) **throws** MariaException

***Печать QR-кода***

***@параметр placeForPrint*** признак расположения текста *0/1* ***-*** до/после фискальной

*информации*

***@return*** *возвращает 0-успех,1-ошибки протокола, 2-неверные параметры* **int** PrintQR (**int *placeForPrint***) **throws** MariaException

***Загрузка логотипа из файла в ЭККА***

***@параметр path*** *Путь к файлу****@параметр sizeOption*** *0/1/2 без изменений/растянуть/подогнать под допустимый @результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetLogoFromFile (String path, **int** sizeOption) **throws** MariaException

***Отмена печати логотипа на чеках***

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** ClearLogo () **throws** MariaException

|  |
| --- |
|  |
| **Печать отчетов** |

***Реализация товаров и услуг в разрезе артикулов****.*

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** ArticleReport () **throws** MariaException

***Примененные скидки и надбавки****.*

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** DiscountReport() **throws** MariaException

***Периодический отчет ФП (по номерам)***

*В случае, если между Z-отчетами с номерами startZReport и endZReport производилась  
смена валюты ЭККР, более 300 изменений схем налогообложения, замена фискального  
(регистрационного) номера ЭККР, то печатается несколько  
отчетных чеков, соответствующих периодам с одинаковой валютой, 300 наборами  
схем налогообложения, с одним и тем же регистрационным номером ЭККР.****@параметр startZReport*** *int – до 4 знаков номер начального Z-отчета.****@параметр endZReport*** *int – до 4 знаков номер конечного Z-отчета.****@параметр print\_full\_report*** *boolean false/true сокращенный/полный  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** PeriodicalFiscalReport(String startZReport, String endZReport, **boolean** print\_full\_report) **throws** MariaException

***Печать X-отчёта***

***@параметр printFundsFlow*** *0/1 признак печати в X- отчете сведений о движении средств*

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** XReport(**boolean** printFundsFlow) **throws** MariaException

***Печать Z-отчёта***

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**public int** ZReport() **throws** MariaException{

***Периодический отчет ФП (по датам)***

*В случае, если между датами dateBegin и dateEnd производилась  
смена валюты ЭККР, более 300 изменений схем налогообложения, замена фискального  
(регистрационного) номера ЭККР, то печатается несколько  
отчетных чеков, соответствующих периодам с одинаковой валютой, 300 наборами  
схем налогообложения, с одним и тем же регистрационным номером ЭККР.****@параметр dateBegin*** *String - 8 символов дата начала отчета в формате*

*ггггммдд;****@параметр dateEnd*** *String - 8 символов дата конца отчета в формате*

*ггггммдд;****@параметр print\_full\_report*** *boolean false/true сокращенный/полный  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** PeriodicalFiscalReportDate(String dateBegin, String dateEnd, **boolean** print\_full\_report) **throws** MariaException

***Состояние аппаратуры***

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** GetHardwareStatus() **throws** MariaException

|  |
| --- |
| **Управление устройствами** |

***Выдача информации на выносной индикатор***

*@параметр symbols String до 200 произвольных символов с соблюдением*

*ограничений транспортного протокола – данные для*

*передачи по каналу связи на выносной индикатор.  
 В качестве данных ВУ формирует последовательности*

*символов в соответствии с протоколом конкретного*

*индикатора.*

*@результат 0/1 успех/ошибка(генерирует исключение MariaException)*

int PutToExternalDisplay(String symbols) throws MariaException{

***Обеспечение совместимости выдачи информации на встроенный  
10-ти разрядный LED (или 16-ти разрядный LCD) индикатор ЭККР предыдущих моделей.***

***@параметр messageNumber*** *1 символ [’1’,'2',’3’]:  
 - номер зарезервированного сообщения на первой строке*

*Дисплея («Цiна», «Сума» или «Здача» соответственно).  
 При значении параметра равном ‘0’ ни одно сообщение не*

*выводится.****@параметр sum*** *сумма отображается на второй строке индикатора.  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** PutSumToDisplay(String messageNumber, **double** sum) **throws** MariaException

***Управление работой обрезчика чековой ленты и звуковым сигналом***

***@параметр isCutterEnabled*** *включает(1) или выключает(0) автоматическую обрезку*

*чековой ленты после завершения печати чека****@параметр isBeepEnabled*** *включает(1) или выключает(0) функцию звукового сигнала*

*после завершения печати чека****@параметр partialCutOn*** *включает(1) или включает(0) функцию неполной обрезки*

*чековой ленты.  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** CutAndBeep(**boolean** isCutterEnabled, **boolean** isBeepEnabled, **boolean** partialCutOn)

**throws** MariaException

***Прогон чековой ленты***

***@параметр*** *steps**int до 4-х символов число от 0 до 65535 –  
 количество шагов двигателя протяжки чековой ленты (шаг 0,125 мм).****@результат*** *0/1 успех/ошибка, генерирует исключение MariaException*

**int** Feed(**int** steps) **throws** MariaException

***Открытие кассового ящика***

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** OpenCashBox() **throws** MariaException

|  |
| --- |
| **Настройка параметров** |

***Определение режима печати информации о наложенных налогах***

*@параметр on 1 - наложенные налоги печатаются после каждой товарной строки в*

*чеке, 0 - сумма наложенных налогов единожды печатается перед*

*строкой "РАЗОМ"*

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetOnSaleTaxes ( **int** on ) **throws** MariaException{

***Управление определением налогооблагаемого оборота при применении двух налогов***

*В обычном режиме налогооблагаемый оборот для двух схем налогообложения  
определяется независимо от порядка применения таких схем следующим образом:  
 Пусть процентная ставка одного налога A%, второго налога B%,  
 Тогда оборот для налога A равен S-((S\*B)/(100+A+B)), оборот для налога B равен  
 S - ((S\*A)/(100+A+B)).  
Значение schemaId1 соответствует идентификатору одного из налогов (далее «Налог1»),  
Значение schemaId2 соответствует идентификатору другого  
Для изменения указанного выше порядка определения налогооблагаемого оборота****@параметр schemaId1*** *String 1 символ идентификатор (номер) первой*

*применяемой схемы налогообложения оборотов по реализации*

*(возврата) товаров (услуг) ['А'..'Ж'] (КИРИЛЛИЦА).****@параметр schemaId2*** *String 1 символ идентификатор (номер) второй*

*применяемой схемы налогообложения  
 оборотов по реализации (возврата) товаров (услуг)*

*['А'..'Ж'] (КИРИЛЛИЦА).  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetDoubledTaxCalcMode (String schemaId1, String schemaId2) **throws** MariaException{

***Программирование необязательной заключительной строки на чеке***

*Настройка сохраняется независимо от наличия электропитания до явного ее изменения вызовом функции SetCheckBottomString.****@параметр bottom*** *String до 43-х символов. Символы, переданные в bottom,  
 печатаются заключительной строкой на чеке  
 перед информацией о дате/времени печати.  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetCheckBottomLine(String bottom) **throws** MariaException

***Программирование нескольких необязательных заключительных строк на чеке***

Детальнее: описание команды протокола: [**BOTm]**

***@параметр lastStringNumber*** *String 1 символ [“0”..”9”] номер заключительной строки*

*сверху вниз.****@параметр fontModifier*** *String (не обязателтный параметр) 1 символ [“0”..”3”] признак печати строки модифицированным шрифтом:  
 “0” – нет модификации,  
 “1” - c удвоенной шириной,  
 “2” – с удвоенной высотой,  
 “3” - c удвоенными шириной и высотой.* ***@параметр strToPrint*** *String (не обязателтный параметр) до 43-х символов*

*собственно строка для печати.  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetCheckBottomLineEx(String lastStringNumber,String fontModifier, String

strToPrint) **throws** MariaException

***Программирование заголовочной информационной строки на чеке***

***@параметр header*** *String до 43-х символов Символы, переданные в header,*

*печатаются первой строкой на чеке.  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetCheckHeadLine(String header) **throws** MariaException

***Программирование мнемоники торгового отдела***

*По умолчанию после строки с номером чека на документе печатается строка  
вида "Вiдд.XXXXXXXXXXXXXXXX Касир YYYYYYYYY". В зависимости от специфики  
торгового предприятия командой 'DEPT' измените символы "Вiдд." на требуемые ("Окно", "Терм" и т.д.).****@параметр MnemonicValue*** *String до 5-ти символов.  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetDeptAlias(String MnemonicValue) **throws** MariaException

***Установка наименования налога***

***@параметр TaxSchemeNumber*** *int номер схемы налогообложения [1..7] соответственно*

*cхем налогообложения [A..Ж]*

***@параметр TaxSchemeName*** *String до 19-ти символов наименование налога*

*соответственно*

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int int** SetTaxName(**int** TaxSchemeNumber,String TaxSchemeName)**throws** MariaException

***Программирование cхемы налогообложения***

***@параметр TaxSchemeNumber*** *int номер схемы налогообложения[1..7] соответственно*

*cхем налогообложения [A..Ж]*

***@параметр TaxType*** *int 0/1/2 соответственно вложенный/наложенный с*

*добавлением/наложенный с вычитанием (“подоходный”)*

***@параметр TaxRate*** *double процентная ставка налога (например 20.00)*

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetTaxScheme(**int** TaxSchemeNumber,**int** TaxType, **double** TaxRate) **throws** MariaException

***Управление печатью информации о скидках-надбавках в чеке***

*Применяется до открытия чека командой OpenCheck.****@параметр value*** *int 1 символ  
 ‘0’ – обычная печать информации о скидке-надбавке  
 после каждой фискальной позиции в чеке,  
 ‘1’ - печать информации об итоговой скидке-надбавке*

*только в итоге чека.  
 @результат String возвращает 1 символ  
 ‘0’ – обычная печать информации о скидке-надбавке*

*после каждой фискальной позиции в чеке,  
 ‘1’ - печать информации об итоговой скидке-надбавке*

*только в итоге чека.*String SetDiscountPrintMode (**int** value) **throws** MariaException

***Установка наименования итогов по скидкам-надбавкам в чеке***

***@параметр discountTotalsName*** *String 22 символа - наименование итога по скидкам в*

*пределах закрываемого чека.****@параметр surchargesTotalsName*** *String 22 символа - наименование итога по надбавкам в*

*пределах закрываемого чека.  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetDiscountUpcountTotalsName(String discountTotalsName, String

surchargesTotalsName) **throws** MariaException

***Определение набора служебных сообщений на чеке***

*Используйте команду перед открытием чека, т.е. до вызова OpenCheck.  
В зависимости от специфики работы предприятия, реализующего товары  
(услуги), меняется набор служебных сообщений на чеке, печатаемых перед каж-  
 дой фискальной частью (реализация и возврат):  
 +----------------------------------------+----------------------------------------------------------+  
 | Номер набора (специфика) | Сообщение при РЕАЛИЗАЦИИ | Сообщение при ВОЗВРАТЕ |  
 +----------------------------------------+----------------------------------------------------------+  
 | '0' торговые ( розница, опт) | ---- | "ПОВЕРНЕННЯ ПО ЧЕКУ N..." |  
 +----------------------------------------+--------------------------+-------------------------------+  
 | '1' автостанции, вокзалы | ---- | "ПОВЕРНЕННЯ ПО ЧЕКУ N..." |  
 +----------------------------------------+--------------------------+-------------------------------+  
 | '2' услуги по кредитованию, | "ОДЕРЖАНО:" | "ВИДАНО:" |  
 | ломбарды, игорный бизнес | | |  
 | (только «Возврат»: операции возмещения | | |  
 | по фишкам и выплаты выигрыша) | | |  
 +----------------------------------------+--------------------------+-------------------------------+  
 Настройка сохраняется независимо от наличия электропитания до явного  
 ее изменения новым вызовом setServiceMessagesKit.  
 Заводская установка режима – «'0' - торговые».****@параметр messagesKitNumber*** *int 1 символ в диапазоне [0..2] номер набора сообщений.  
  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetServiceMessagesKit(**int** messagesKitNumber) **throws** MariaException

***Управление видом представления фискальной информации на чеке***

*Имеется возможность представлять фискальную информацию на чеке в табличном виде, когда цена, количество и сумма по каждой позиции чека печатаются в соответствующих колонках таблицы.*

***@параметр isTableView*** *true/false Включение/Выключение табличного режима  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetCheckAsTable(**boolean** isTableView) **throws** MariaException

***Установка режима «построчной» печати чека***

*Применяется до открытия чека вызовом openCheck.  
Действует только в пределах одного чека.*

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetLinePrintingOn() **throws** MariaException

***Корректировка часов реального времени***

*можно выполнить 1 раз не более +/- 90 минут оттекущего времени после Z-отчета с обнулением****@параметр hours  
@параметр minutes  
@параметр seconds****@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetTime(**int** hours, **int** minutes, **int** seconds) **throws** MariaException

***Корректировка часов реального времени – плюс 1 час***

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetTimePlusOneHour() **throws** MariaException

***Корректировка часов реального времени – минус 1 час***

*@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** SetTimeMinusOneHour() **throws** MariaException

|  |
| --- |
| **Информация** |

***Запрос регистра учета реализации артикула***

***@параметр articleNumber*** *int - номер артикула в диапазоне 1-15516 или  
 - номер артикула по бухгалтерской (внутрисистемной)  
 кодировке в диапазоне 1..999999999 - только в режиме*

*работы артикульной таблицы «Регистрация новых по*

*бухгалтерским кодам»****@результат*** *возвращает HashMap<String, String>:  
 - Ключ: articleNumberRes, значение: номер запрошенного*

*артикула;  
 - Ключ: goodName, значение: наименование товара(24*

*символа);  
 - Ключ: isDivisible, значение: 0 - делимый, 1 –*

*неделимый  
 - Ключ: taxingState, значение: 8 символов состояние*

*налогообложения. Каждая позиция слева  
 направо соответствует схемам налогообложения от ‘А’ до*

*‘З’. Если данная  
 схема налогообложения не используется, в*

*соответствующей позиции  
 находится символ ‘0’. В противном случае в*

*соответствующей позиции  
 находится соответствующий символ [‘А’..‘З’].  
 Схема 'З' не программируется по setUpSchemesOfTaxation,  
 всегда активна со значениями:  
 тип «вложенный» (0), ставка 0,00% (0000).  
 Указание на использованиесхемы ‘З’ подразумевает*

*регистрацию операции такой,  
 что не являетсяобъектом налогообложения;  
 - Ключ: quantityOfGoodsSoldByArticle, значение: 10*

*символов в формате XXXXXX.XXXX:  
 количество реализованного товара с данным артикулом;  
 - Ключ: totalGoodsSoldByArticle, значение: 10 символов*

*общая сумма реализации товара  
 с данным артикулом в копейках;  
 - Ключ: cashlessPayment, значение: 9 символов код*

*артикула по бухгалтерской (внутрисистемной)  
 кодировке в диапазоне ["000000001".."999999999"].*

*Передается только в режиме работы артикульной таблицы  
 «Регистрация новых по бухгалтерским кодам» \*  
 генерирует исключение MariaException в случае неудачи*HashMap<String, String> GetArticleInfo(**int** articleNumber) **throws** MariaException

***Определение состояния фискальной смены ЭККА***

***@результат*** *0 (произошла ошибка), 1 (смена закрыта), 2 (смена*

*открыта) (генерирует исключение MariaException)***int** GetBusinessDayState ()**throws** MariaException

***Запрос информации о движении средств по кассе***

*Показывает актуальное (при currentOrPrev == 0) или предыдущее (при currentOrPrev == 1) состояние регистров учета движения денежных средств, изменяемых операциями реализации-возврата или служебным внесением-изъятием денежных средств.*

***@параметр currentOrPrev*** *int - 0 или 1 – признак запроса данных «по текущей*

*смене» или «по предыдущей смене» соответственно.****@результат*** *возвращает HashMap<String, String>:  
 - Ключ: initialBalance, значение: "Початковий Залишок";  
 - Ключ: serviceEntry, значение: "Службове Внесення";  
 - Ключ: serviceRemoval, значение: "Службове Вилучення";  
 - Ключ: obtained, значение: "Одержано";  
 - Ключ: issued, значение: "Видано";  
 - Ключ: finalBalance, значение: "Кiнцевий залишок";  
 - Ключ: cashlessPayment, значение: "Безготiвкова*

*оплата";  
 - Ключ: cashlessRefund, значение: "Безготiвкове*

*повернення";  
   
 генерирует исключение MariaException в случае неудачи*HashMap<String, String> GetCashInfo(**int** currentOrPrev) **throws** MariaException

***Запрос текущего состояния дневных фискальных регистров****.*

***@результат*** *возвращает HashMap<String, String>:  
 - Ключ: totalSalesTurnover, значение: общий оборот*

*реализации;  
 - Ключ:*

*salesAmountTurnoverBySchemesTaxati*

*salesAmountTurnoverBySchemesTaxation7,*

*значение: суммы оборотов реализации по схемам*

*налогообложения, соответствующих номерам ['А'..'З'];  
 - Ключ: nonTaxableSales, значение: не облагаемый налогом*

*оборот реализации;*

*- Ключ: totalRefundTurnover, значение: общий оборот*

*возврата;*

*- Ключ: refundAmountTurnoverBySchemesTaxation1.*

*.refundAmountTurnoverBySchemesTaxation7,  
 назначение: 12-ти символьные суммы оборотов возврата по*

*схемам налогообложения,*

*соответствующих номерам ['А'..'З'];*

*- Ключ: totalRefundTurnover, значение: не облагаемый*

*налогом оборот возврата;  
 Генерирует исключение MariaException в случае неудачи*

HashMap<String, String> GetFiscalInfo() **throws** MariaException

***Запрос номеров чеков, документов, идентификатора транзакции***

***@результат*** *возвращает HashMap<String, String>:  
 - Ключ: lastCheck, значение: 10 символов – номер*

*последнего закрытого (или текущего открытого)*

*чека реализации-возврата (продажи-покупки);  
 - Ключ: lastDocDevice, значение: 10 символов – номер*

*последнего закрытого или текущего открытого*

*служебного документа, созданного (создаваемого)*

*последовательностью команд (вызовов) от Android устройства;  
 - Ключ: lastDocMaria, значение: 10 символов – номер*

*последнего закрытого служебного документа,  
 созданного ЭККР самостоятельно, как результат:  
 - вызова performCashDepositOrWithdraw  
 - выполнения служебных отчетов  
 - самостоятельной печати служебных сообщений (включение*

*ЭККР, смена времени и т.п.),*

*содержащих строку «СЛУЖБОВИЙ ДОКУМЕНТ»  
 Больший из номеров lastDocDevice,lastDocMaria является*

*последним сквозным номером документа.  
 - Ключ: articleTableMode, значение: 1 символ ‘0’, ‘1’ или*

*‘2’ - режим артикульной таблицы:  
 «Регистрация новых», «Использование запрограммированных»*

*или«Регистрация новых» соответственно;  
 - Ключ: isFiscalReportPerformed, значение: 1 символ - признак*

*выполнения фискального отчета с обнулением  
 "0" или "1" - не выполнен/выполнен;  
 - Ключ: fiscalCheckNumber, значение: 4 символа - номер*

*фискального отчетного чека(в зависимости от значения признака*

*выполненного Z- отчета: либо номер последнего Z-*

*отчета("выполнен"), либо номер следующего Z- отчета ("не*

*выполнен")).  
 генерирует исключение MariaException в случае неудачи*

HashMap<String, String> GetDocumentsInfo() **throws** MariaException

***Запрос даты-времени часов реального времени ЭККР***

*Используйте для контроля состояния часов реального времени  
ЭККР и для получения информации о времени для ВУ, не имеющих своих часов.****@результат*** *возвращает HashMap<String, String>:  
 - Ключ: currentDate, значение: 8 символов - текущая дата  
 (по системным часам реального времени) в формате*

*ггггммдд;  
 - Ключ: currentTime, значение: 6 символов - текущее*

*время  
 (по системным часам реального времени) в формате ччммсс;  
 генерирует исключение MariaException в случае неудачи*HashMap<String, String> GetDate() **throws** MariaException

***Запрос списка запрограммированных схем налогообложения***

***@параметр format***

***@результат*** *Если параметр format == true*

*возвращает HashMap<String, String>:*

*- Ключ: progdate , значение: ггггммдд - дата программирования  
 - Ключ: taxnum, значение: ['1'..'8'] - номер схемы*

*- Ключ:taxType, значение: т ['0'..’1’] -тип вложенный/наложенный.*

*- Ключ:taxValue, значение: сссс - ставка в процентах с*

*сотыми долями без десятичной точки.*

*Если параметр format == false ответ на команду содержит информацию:*

*-Ключ: taxName[1..8], значение:19-ти символьные наименования*

*налогов, позиционно соответствующие схемам от ‘А’ до ‘З’.*

HashMap<String, String> GetTaxesInfo(**boolean** format) **throws** MariaException

***Запрос внутреннего состояния ЭККР****.*

***@результат*** *возвращает HashMap<String, String>:  
 - Ключ: number, значение: 10 символов – последние 10*

*символов заводского номера(задан при изготовлении*

*устройства);  
 - Ключ: ownerNumber, значение: 10 символов –*

*регистрационный номер(программируется вызовом*

*registerOwnerInfo);  
 - Ключ: enterpriseInfo, значение: 36 символов –*

*наименование и адрес предприятия(программируется вызовом*

*registerOwnerInfo);  
 - Ключ: currentDate, значение: 8 символов - текущая дата  
 (по системным часам реального времени) в формате*

*ггггммдд;  
 - Ключ: currentTime, значение: 6 символов - текущее*

*время(по системным часам реального времени) в формате*

*ччммсс;  
 - Ключ: keyPosition, значение: 1 символ - положение*

*cистемного ключа - зависит от положения ключа:  
 “ОТКЛЮЧЕН” (О) 0  
 “РАБОТА” (Р) 1  
 “X- ОТЧЕТ” (X) 2  
 “Z-ОТЧЕТ” (Z) 4  
 “ПРОГРАММИРОВАНИЕ” (П) 8  
 - Ключ: expectedCommand, значение: 1 символ - признак*

*ожидаемого вызова функции создания документов  
 (зависит от последней вызваной функции создания*

*документов)*

*- Ключ: isCashierRegistered, значение: 1 символ –*

*признак зарегистрированного кассира  
 ("0"/"1" - не зарегистриран/зарегистрирован);  
 - Ключ: cashierId, значение: 4 символа - идентификатор*

*зарегистрированного кассира;  
 - Ключ: isReported, значение: 1 символ - признак*

*выполнения фискального отчета с обнулением  
 (0/1 - не выполнен/выполнен, меняет свое значение на  
 0 после подачи первой команды openCheck после*

*выполнения Z- отчета);  
 - Ключ: fiscalCheckNumber, значение: 12 символов - номер*

*фискального отчетного чека  
 (в зависимости от значения признака выполненного Z-*

*отчета либо номер последнего Z- отчета  
 ("выполнен"), либо номер следующего Z- отчета ("не*

*выполнен")).  
 - Ключ: lastCheckNumber, значение: 12 символов - номер*

*последнего успешно созданного  
 (или открытого по openCheck в данный момент) чека.  
 - Ключ: lastSuccessCommand, значение: 4 символа –*

*идентификатор последней успешно исполненной команды;  
 - Ключ: versionId, значение: 4 символа - идентификатор*

*версии ПО ЭККР;  
 - Ключ: versionDate, значение: 8 символов - дата*

*создания версии ПО ЭККР в формате ггггммдд;  
 - Ключ: currentHeader, значение: 18 символов - текущая информационная строка чека(совпадает с первыми 18-ю*

*символами значения header, переданного в последней*

*успешно исполненной команде* SetCheckHeadLine*);  
 - Ключ: setupCurrencyDate, значение: 8 символов - дата*

*программирования валюты ЭККР в формате ггггммдд  
 (совпадает со значением даты системных часов в момент*

*программирования валюты вызовом setupCurrency);  
 - Ключ: decimal, значение: 1 символ - количество знаков*

*после десятичной точки в изображении  
 сумм (совпадает со значением decimal, переданного при*

*последнем успешном вызове setupCurrency);  
 - Ключ: currencyName, значение: 3 символа - сокращенное*

*наименование валюты ЭККР   
генерирует исключение MariaException в случае неудачи*HashMap<String, String> GetPrinterConfig() **throws** MariaException

***Время до блокировки ЭККА в часах***

***@результат*** *Возвращает время до блокировки ЭККА в часах*

*(сбрасывается на значение '72 часа' после  
 каждого успешного сеанса связи с сервером ДПА).*

*Отрицательное значение означает, что ЭККА  
 заблокирован. Значение 999999999 означает, что ЭККА не*

*поддерживает блокировку (старые модели)***int** GetTimeToPendingLock () **throws** MariaException

***Проверка состояния канала связи «Внешнее Устройство - ЭККР»***

***@параметр randomData*** *String до 252 произвольных символов;****@результат*** *String копия randomData  
генерирует исключение MariaException в случае неудачи*String checkConnection(String randomData) **throws** MariaException

***Запрос состояния батареи***

***@результат*** *(-1) ошибка, заряд аккумулятора в процентах*

int GetBatteryStatus()**throws** MariaException

***Запрос количества пакетов данных, ожидающих отправку на сервер эквайера.***

***@результат***  *возвращает HashMap<String, Integer>:  
 - Ключ: TOTAL\_PKG, значение: Всего документов к отправке  
 - Ключ: TOTAL\_CHK, значение: Количество чеков, ожидающих*

*отправку  
 - Ключ: TOTAL\_SDOC, значение: Количество служ.*

*документов, ожидающих отправку  
 - Ключ: TOTAL\_ZRREP, значение: Количество отчетов,*

*ожидающих отправку  
  
 пробрасывает исключение MariaException в случае неудачи*

HashMap<String,Integer> GetCountPendingDocs () **throws** MariaException

|  |
| --- |
| **Бонусы, карты** |

***Прием платежа на бонусный счет/подарочный сертификат/клубную карту***

***@параметр summOp*** *сумма****@параметр TypeOp*** *1/2/3 бонусный счет/подарочный сертификат/клубная*

*карта****@параметр DescribeOp*** *128 символов дополнительная информация****@параметр Tax1*** *1 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается****@параметр Tax2*** *2 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** PutAccountingPayments (**double** summOp, **int** TypeOp, String DescribeOp, **int** Tax1, **int**

Tax2) **throws** MariaException

***Пример чека с начислением бонусов, продажи подарочного сертификата.***

OpenCheck(…)

FiscalLine(“Костюм черный”,…);

PutAccountingPayments(0.10,1,0,0,”Номер 123456”);

PutAccountingPayments(100,2,0,0,”СЕРТИФIКАТ 'КРАЩИЙ ПОДАРУНОК'”);

CloseCheck();

***Списание средств с бонусного счета/подарочного сертификата/клубной карты***

***@параметр summOp*** *сумма****@параметр TypeOp*** *1/2/3/4 бонусный счет/подарочный сертификат/клубная*

*Карта/скидка на общий оборот****@параметр DescribeOp*** *128 символов дополнительная информация****@параметр Tax1*** *1 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается****@параметр Tax2*** *2 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** TakeAccountingPayments (**double** summOp, **int** TypeOp, String DescribeOp, **int** Tax1,

**int** Tax2) **throws** MariaException

***Пример чека с учетом бонусов, подарочного сертификата и скидки на общий оборот***

OpenCheck(“ПРОМТОВАРЫ”)

FiscalLine(“Костюм черный”,…);

TakeAccountingPayments(0.10,1,1,0,”Номер 123456”);

TakeAccountingPayments(200,2,1,0,”Номер 123456789”);

TakeAccountingPayments(100,4,1,0,”ЗА НАПОЛЕГЛИВIСТЬ”);

CloseCheck();

|  |
| --- |
| **Авансовые платежи** |

***Открытие чека “заказа”. Прием авансового платежа****.*

*@параметр departmentName до 15-ти символов идентификатор (наименование) торгового*

*отдела.  
@результат**0/1 успех/ошибка(генерирует исключение MariaException)*

**int** OpenOrderCheck (String departmentName) **throws** MariaException

***Печать описания одной заказываемой позиции товара(услуги) в пределах чека заказа***

*@параметр <nameTov> до 32 символов наименование (услуги)  
@параметр <count> количество единиц  
@параметр <singleCost> стоимость единицы товара (услуги)  
@параметр <isDivisible> символ признак делимости, где  
 '0' - делимый, при этом значение <п2> интерпретируется как*

*xxx,xxx.  
 '1' - неделимый, при этом значение <п2> интерпретируется*

*как xxxxxx.****@результат*** *0/1/2-3 успех/ошибка протокола/ошибки параметров функции***int** OrderLine(String nameTov, **double** count, **double** Cost, **int** isDivisible) **throws**

MariaException

***Регистрация оборота приема аванса и закрытие чека заказа***

***@параметр NameOrder*** *до 128 символов дополнительная информация об авансе****@параметр Tax1*** *1 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается****@параметр Tax2*** *2 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается****@результат*** *0/1/2 успех/ошибка протокола/ошибки параметров функции***int** CloseOrder(String NameOrder, **int** Tax1, **int** Tax2 )**throws** MariaException

***Пример чека заказа***

OpenOrderCheck(“БАКАЛЕЯ”);

OrderLine(“НАИМЕНОВАНИЕ ТОВАРА 1”,456.78,19.63,0);

OrderLine(“НАИМЕНОВАНИЕ ТОВАРА 2”,56.78,237.8,0);

OrderLine(“НАИМЕНОВАНИЕ ТОВАРА 3”,0.011,10,0);

CloseOrder(“%”,1,0)

***Открытие чека “выполнения заказа”. Расчет по авансовому платежу***

*@параметр departmentName до 15-ти символов идентификатор (наименование) торгового*

*отдела.****@результат*** *0/1 успех/ошибка протокола*

**int** OpenAdvancePaymentCheck (String departmentName) **throws** MariaException

***Учет авансового платежа (уменьшение оборота реализации)***

*@параметр <sum> сумма учитываемого аванса  
@параметр <Tax1> 1 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается  
@параметр <Tax2> 2 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается  
@параметр <ExtendedInfo> до 128 символов дополнительная информация об авансе,****@результат*** *0/1/2-3 успех/ошибка протокола/ошибки параметров функции***int** AdvancePaymentLine(**double** sum, **int** Tax1, **int** Tax2, String ExtendedInfo) **throws**

MariaException

***Пример чека выполнения заказа***

OpenAdvancePaymentCheck(“гастрономия”)

OpenGoodsGroup(“GRBGПО ЗАМОВЛЕНЮ”)

FiscalLine(“ЗАКАЗ ТОВАР1”…);

FiscalLine(“ЗАКАЗ ТОВАР2”…);

FiscalLine(“ЗАКАЗ ТОВАР3”…);

CloseGoodsGroup(“”,””)

AdvancePaymentLine(“ПО ЧЕКУ 123456”,…)

AdvancePaymentLine(“ПО ЧЕКУ 1234567”,…)

CloseCheck();

***Открытие чека “возврата аванса”. Расчет по авансовому платежу.***

*@* ***параметр*** *departmentName до 15-ти символов идентификатор (наименование) торгового*

*отдела.  
@* ***результат*** *0/1 успех/ошибка***int** OpenReturnAdvance(String departmentName) **throws** MariaException

***Регистрация оборота возврата полной суммы ранее принятого аванса.***

*@****параметр*** *sum сумма возвращаемого аванса****@параметр Tax1*** *1 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается****@параметр Tax2*** *2 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается  
@* ***параметр*** *ExtendedInfo до 128 символов дополнительная информация об авансе,  
@* ***результат*** *0/1/2-3 успех/ошибка протокола/ошибки параметров функции***int** RegistrationReturnAcceptedAdvance(**double** sum, **int** Tax1, **int** Tax2, String ExtendedInfo) **throws** MariaException

***Учет сумм отпущенных товаров-услуг при возврате аванса***

*@****параметр*** *sum сумма возвращаемого аванса****@параметр Tax1*** *1 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается****@параметр Tax2*** *2 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается  
@* ***параметр*** *ExtendedInfo до 128 символов дополнительная информация об авансе,  
@* ***результат*** *0/1/2-3 успех/ошибка протокола/ошибки параметров функции***int** RegistrationSummByAcceptedAdvance(**double** sum, **int** Tax1, **int** Tax2, String ExtendedInfo) **throws** MariaException{

***Пример чека возврата аванса:***

OpenReturnAdvance(“гастрономия-БАК”);

RegistrationReturnAcceptedAdvance(24.64,1,0,” По чеку 123456”);

RegistrationReturnAcceptedAdvance(24.64,1,0,” По чеку 456”);

RegistrationSummByAcceptedAdvance(24.64,1,0,” По ЧЕКУ NNN”);

RegistrationSummByAcceptedAdvance(24.60,1,0,” По ЧЕКУ XXX”);

CloseCheck();

|  |
| --- |
| **Печать билетов** |

***Открытие чека продажи билета (багажной квитанции)***

*@****параметр*** *TicketOrBaggage 0/1 билет/багажная квитанция  
@****параметр*** *TypeTicket [0..4] «ПОЛНЫЙ», «ДЕТСКИЙ», «ЛЬГОТНЫЙ»,*

*«АБОНЕМЕНТНЫЙ», «БЕСПЛАТНЫЙ»*

*Если TypeTicket из ряда {«ДЕТСКИЙ», «ЛЬГОТНЫЙ», «БЕСПЛАТНЫЙ»}-  
 нижеследующие параметры являются обязательными:  
@****параметр*** *SysNumberDoc 0 до 20 символов Системный номер документа*

*(билета/квитанции)  
@****параметр*** *IdentDocExemption 0 до 40 символов Идентификация документа на право*

*на льготы  
@****параметр*** *RateExemption 0 до 10 символов Идентификатор льготной ставки  
@****результат*** *0/1/2-3 успех/ошибка протокола/ошибки параметров функции***int** OpenTicketForSale(**int** TicketOrBaggage, **int** TypeTicket, String SysNumberDoc, String IdentDocExemption, String RateExemption) **throws** MariaException

***Открытие чека возврата билета (багажной квитанции)***

*@****параметр*** *TicketOrBaggage 0/1 билет/багажная квитанция  
@****параметр*** *TypeOperation 1/2 «возврат билета» /«возврат как погашение*

*испорченного документа»  
@****параметр*** *TypeTicket [0..4] «ПОЛНЫЙ», «ДЕТСКИЙ», «ЛЬГОТНЫЙ»,*

*«АБОНЕМЕНТНЫЙ», «БЕСПЛАТНЫЙ»  
 Если TypeTicket из ряда {«ДЕТСКИЙ», «ЛЬГОТНЫЙ», «БЕСПЛАТНЫЙ»}-  
 нижеследующие параметры являются обязательными:  
@****параметр*** *SysNumberDoc 0 до 20 символов Системный номер документа*

*(билета/квитанции)  
@****параметр*** *IdentDocExemption 0 до 40 символов Идентификация документа на право на*

*льготы  
@****параметр*** *RateExemption 0 до 10 символов Идентификатор льготной ставки  
@****результат*** *0/1/2 успех/ошибка протокола/ошибки параметров функции***int** OpenTicketForReturn(**int** TicketOrBaggage, **int** TypeOperation, **int** TypeTicket, String SysNumberDoc, String IdentDocExemption, String RateExemption) **throws** MariaException

***Ввод описания маршрута***

*@****параметр*** *NameStationDeparture от 1 до 99 символов Наименование станции отправления  
@****параметр*** *NameStationDestination от 0 до 40 символов Наименование станции назначения  
@****параметр*** *NameNodePoint от 0 до 40 символов Наименование узлового пункта.  
@****параметр*** *NameTransplantPoint от 0 до 11 символов Наименование пункта пересадки  
@****параметр*** *DateExpired 10 символов граничная дата действия документа  
@****результат*** *0/1/2 успех/ошибка протокола/ошибки параметров функции***int** TicketRouteDescriptions(String NameStationDeparture,String NameStationDestination,String NameNodePoint,String NameTransplantPoint,String DateExpired) **throws** MariaException

***Ввод сумм составляющих стоимости билета (квитанции)***

***составляющая билета «ТАРИФ» «БАГАЖ»****@****параметр*** *CntPlaceOrPsngrs (числовое значение от 1 до 9999) количество*

*пассажирских мест или мест багажа  
@****параметр*** *CostPlaceOrPsngrs стоимость одного места (стоимость составляющей*

*«ТАРИФ» или «БАГАЖ»)  
@параметр TaxSchemePlaceOrPsngrs 0/1 применение налогов по обычной схеме / по*

*приоритетной схеме  
@параметр Tax1PlaceOrPsngrs 1 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается  
@параметр Tax2PlaceOrPsngrs 2 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается* ***составляющая билета «СТРАХОВОЙ ПЛАТЕЖ»****@параметр CntInsurancePayment (числовое значение от 1 до 9999) количество единиц*

*составляющей стоимости «СТРАХОВОЙ ПЛАТЕЖ»  
@параметр CostInsurancePayment стоимость составляющей стоимости «СТРАХОВОЙ ПЛАТЕЖ»  
@параметр TaxSchemeInsurance 0/1 применение налогов по обычной схеме / по*

*приоритетной схеме  
@параметр Tax1Insurance 1 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается  
@параметр Tax2Insurance 2 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается* ***составляющая билета «КОМИССИОННЫЙ СБОР»****@параметр CntCommissionFee (числовое значение от 1 до 9999) ) количество единиц*

*составляющей стоимости «КОМИССИОННЫЙ СБОР»  
@параметр CostCommissionFee стоимость составляющей стоимости «КОМИССИОННЫЙ СБОР»  
@параметр TaxSchemeCommission 0/1 применение налогов по обычной схеме / по*

*приоритетной схеме  
@параметр Tax1Commission 1 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается  
@параметр Tax2Commission 2 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается  
  
@****результат*** *0/1/2 успех/ошибка протокола/ошибки параметров функции***int** SumComponentsOfTicket(**int** CntPlaceOrPsngrs, **double** CostPlaceOrPsngrs, **int** TaxSchemePlaceOrPsngrs, **int** Tax1PlaceOrPsngrs, **int** Tax2PlaceOrPsngrs,  
**int** CntInsurancePayment , **double** CostInsurancePayment, **int** TaxSchemeInsurance, **int** Tax1Insurance, **int** Tax2Insurance, **int** CntCommissionFee, **double** CostCommissionFee, **int** TaxSchemeCommission, **int** Tax1Commission, **int** Tax2Commission ) **throws** MariaException

***Ввод суммы удержания по билету***

*@параметр SumRetention сумма удержания  
@параметр TaxScheme 0/1 применение налогов по обычной схеме / по*

*приоритетной схеме  
@параметр Tax1 1 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается  
@параметр Tax2 2 идентификатор(номер) схемы налогообложения оборота*

*[1..8]['А'..'З'] 0 - налог не учитывается  
@параметр NameRetention от 0 до 128 символов наименование удержания.  
@результат 0/1/2 успех/ошибка протокола/ошибки параметров функции***int** TicketRetention(**double** SumRetention,**int** TaxScheme,**int** Tax1,**int** Tax2,String NameRetention) **throws** MariaException

***Закрытие чека билета (багажной квитанции)***

*@параметр FormOfPayment код формы оплаты  
@параметр SummPaymentCash сумма полученная от клиента. Имеет значение только*

*для чека продажи и формы оплаты наличные.  
@параметр IdPaymentSystem до 20-ти символов идентификатор транзакции платежной*

*системы. Имеет смысл для указанных форм оплаты «не*

*наличные»;  
@результат 0/1/2 успех/ошибка протокола/ошибки параметров функции***int** TicketClose(**int** FormOfPayment,**double** SummPaymentCash,String IdPaymentSystem) **throws** MariaException

***Организация работы с билетами*:**

**Очистить массивы нефискальных строк:** ClearFreeTextLines()

**Загрузить необходимое количество нефискальных строк:** FreeTextLine(**int** , **int** ,String),

LoadQR(**int**, String, String),

PrintQR(**int**)

**Открыть чек:** OpenTicketForSale(**int**, **int**, String, String, String)

или OpenTicketForReturn(**int**, **int,** **int**, String, String, String)

**Внести данные по маршруту:** TicketRouteDescriptions(String, String,String, String,String)

**Ввести суммы составляющих стоимости билета:** SumComponentsOfTicket(**int**, **double**, **int**, **int**,

**int**, **int**, **double**, **int**, **int**, **int**, **int**, **double**, **int**, **int**, **int**)

**Для чека возврата ввести необходимое количество сумм удержаний:**

TicketRetention(**double**,**int**,**int**,**int**,String)

**Закрыть чек:** TicketClose(**int**,**double**,String)

|  |
| --- |
| **Прочее** |

***Принудительно передать данные на эквайер***

***@результат*** *0/1 успех/ошибка***int** TransferForTaxPurposes () **throws** MariaException