FNCore2 версия 3.0 Работа с фискальным накопителем на устройстве МКАССА-9000

Оглавление

Принцип работы

FNCore2 предоставляет API для работы с фискальным накопителем. С помощью него вы можете проводить все фискальные операции, описанные в ФФД 1.2. Для многих объектов (таких как чек, коррекция, предмет расчета) предоставляются объекты-абстракции, которые снимают с программиста работу с тегами, описанными в ФФД напрямую. Хотя, такой вариант работы тоже является допустимым. FNCore2 является отдельным предустановленным приложением, связь с которым осуществляется через Android IPC с использованием ServiceConnection. Все методы работы с ФН рекомендуется выполнять в отдельном потоке, методы помеченные [UI] можно выполнять в основном потоке приложения.

Данный документ не описывает структуру объектов-оберток, они описаны в сопутствующем javadoc, здесь описаны только методы класса FiscalService и пояснения. Данный документ ссылается на приложение FNCoreSample, в котором даны примеры работы с FNCore2.

Соединение с сервисом

Для соединения с сервисом используется интент `rs.fncore3.FiscalStorage` для пакета `rs.fncore3`. Так же вы можете использовать готовый интент `FISCAL_STORAGE` из класса `rs.fncore.Const.`

Так же для удобства работы с фискальным хранилищем предлагается класс `rs.utils.app.AppCore` который унаследован от `Application`, и предлагает средства для работы с FNCore и некоторые дополнительные функции, такие как очередь сообщений. Если вы используете AppCore, то для инициализации предназначены методы `public boolean initialize()` и `public boolean initialize(long waitFNTimeoutMs).` Они устанавливают связь с сервисом и инициализируют класс FiscalStorage, который доступен через метод `protected FiscalStorage getStorage().` Пример инициализации находится в классе Main проекта FNCoreSample.

Методы FiscalStorage

Ниже описаны методы предоставляемые классом FiscalStorage и их назначение. Все методы в случае успеха возвращают 0 или код ошибки, описанный в классе rs.fncore.Errors.

```
int waitFnReady(in long waitFNtimeoutMs);
```
Метод предназначен для ожидания готовности ФН (физического подключения).

```
int readKKMInfo(out KKMInfo info);
```
Чтение настроек и данных фискализации ККТ.

```
int resetFN();
```
Произвести сброс МГМ.

```
int cancelDocument();
```
Отмена незавершенного документа. При выполении любой фискальной операции незавершенный документ отменяется автоматически

```
int doFiscalization(in KKMInfo.FiscalReasonE reason, in OU operator, in KKMInfo info, out KKMInfo signed, String template);
```
Выполнение процедуры фискализации/изменения параметров ККТ. В параметре template вы можете передать свой шаблон для печати документа.

```
int toggleShift(in OU operator, out Shift shift, String template);
```
Переключение состояния смены. Если смена была закрыта, то она откроется и наоборот.В параметре template вы можете передать свой шаблон для печати документа.

```
int updateOfdStatistic(out OfdStatistic statistic);
```
Получить статистику по неотправленным в ОФД документам

```
int requestFiscalReport(in OU operator, out FiscalReport report, String template);
```
Сформировать отчет о состоянии расчетов. В параметре template вы можете передать свой шаблон для печати документа.

```
int doSellOrder(in SellOrder order, in OU operator, out SellOrder signed, boolean doPrint, String header, String item, String footer, String footerEx);
```
Сформировать чек продаж. Вы можете передать в параметрах header, item,footer свои шаблоны для печати чека, а в параметре footerEx — текст, который будет напечатан после чека.

```
int doArchive(in OU operator, out ArchiveReport report, String template);
```
Выполнить перевод ККТ в постфискальный режим. Перед этой операцией смена должна быть закрыта.

```
DocServerSettings getOFDSettings();
```
Получить сетевые настройки ОФД

```
void setOFDSettings(in DocServerSettings settings);
```
Установить сетевые настройки ОДФ. Эта операция может быть выполнена в любой момент.

```
PrintSettings getPrintSettings();
```
Получить настройки печати.

```
void setPrintSettings(in PrintSettings settings);
```
Установить настройки печати (размер шрифта, размер бумаги и т.д.)

```
void doPrint(String text);
```
Напечатать произвольный текст. Для форматирования используйте формат описанный в разделе Шаблоны печати.

```
void pushDocuments();
```
Начать отправку документов в ОФД немедленно.

```
int openTransaction();
```
Открыть соединение с ФН по UART. Возвращает ИД транзакции, 0 в случае неудачи

```
int writeB(int transaction, in byte [] data, int offset, int size);
```
Записать пакет байт в открытое соединение. Возвращает количество записанных байт

```
int readB(int transacrion, out byte [] data, int offset, int size);
```
Прочитать байты из открытого соединения. Возвращает количество прочитанных байт

```
void closeTransaction(int transaction);
```
Закрыть ранее открытое соединение

```
double getCashRest();
```
Получить остаток наличных в кассе

```
int putOrWithdrawCash(double v, in OU operator, String template);
```
Выполнить нефискальную операцию внесения/изъятия наличных. Если v меньше 0 то выполняется изъятие, иначе внесение. В template можно передать свой шаблон для печати

```
void setCashControl(boolean  val);
```
Включить/выключить контроль наличности. По умолчанию выключено. Если контроль наличности включен, то операции расхода при оплате наличности будут заблокированы если остаток наличности меньше чем сумма операции.

```
boolean isCashControlEnabled();
```
Проверить, включен ли контроль наличности.

```
int restartCore();
```
Перезапустить фискальное ядро. В этом случае соединене с ФН устанавливается по-новой

```
int openShift(in OU operator, out Shift shift, String template);
```
Открыть новую смену. В template можно передать свой шаблон для печати

```
int closeShift(in OU operator, out Shift shift, String template);
```
Закрыть смену. В template можно передать свой шаблон для печати

```
boolean isMGM();
```
Признак, является ли ФН МГМ (Массо-габаритным макетом)

```
int updateOismStatistic(out OismStatistic statistic);
```
Получить статистику по документам, подлежащим отправке в ОИСМ

`DocServerSettings getOismSettings();`
Получить сетевые настройки сервера ОИСМ

`void setOismSettings(in DocServerSettings settings);`
Установить сетевые настройки сервера ОИСМ

`DocServerSettings getOKPSettings();`
Получить сетевые настройки сервера ОКП

`void setOKPSettings(in DocServerSettings settings);`
Установить сетевые настройки сервера ОКП

`int checkMarkingItem(inout SellItem item);`
Проверить валидность маркировки предмета расчета

`int confirmMarkingItem(inout SellItem item, boolean accepted);`
Подтвердить маркировку предмета расчета

`int getExistingDocument(int docNo, out rs.fncore.data.Tag doc);`
Прочитать документ из ФН по номеру

```
int doCorrection2(in Correction correction, in OU operator, out Correction
signed, String header, String item, String footer, String footerEx);
```
Сформировать чек коррекции. Вы можете передать в параметрах header, item,footer свои шаблоны для печати чека, а в параметре footerEx — текст, который будет напечатан после чека.

`String getPF(in Tag tag);`
Получить встроенную печатную форму для документа

`int getFNCounters(out FNCounters counters, boolean shiftCounters);`
Получить счетчики итогов ФН. Если значение параметра `shiftCounters` – true, то возвращаются сменные счетчики, иначе счетчики ФН.

`int exportMarking(String file);`
Выгрузить уведомления о продаже маркированного товара. Этот режим доступен если ККТ фискализирована в режиме «ККТ оффлайн». Передается имя файла, в который осуществлять выгрузку.

`long getPaperConsume();`
Получить значение счетчика расхода бумаги в миллиметрах

`void resetPaperCounter();`
Сбросить текущее значение счетчика расхода бумаги в 0

Работа с маркированным товаром

Если ККТ сконфигурирована в режиме работы с маркированным товаром, то прежде чем провести чек, необходимо произвести проверку кода маркировки. Для это требуется

- добавить значение кода маркировки в SellItem через вызов `setMarkingCode(String codeStr, SellOrder.OrderTypeE type).` В качестве параметра type передается тип чека (SellOrder)
- вызвать метод `checkMarkingItem()` FNCore. Вызов надо осуществлять в потоке, т. к. проверка средствами ОИСМ может занять некоторое время. Если результат метода 0, то надо вызвать `confirmMarkingItem,` со вторым параметром (accepted) true, тем

самым подтвердив маркированный товар. Если проверка не выполнена но пользователь все равно хочет включить товар в чек, то так же надо вызвать confirmMarkingItem

Шаблоны документов

Для форматирования используется табличноориентированый язык разметки. Теки определяются следующим образом

тег ::= {\имя[параметр:значение][;параметр:значение][;..]\данные}

Для всех тегов допустимы следующие общие параметры:

style:**тип** — стиль текста, где **тип** один или несколько атрибутов, разделенных запятыми: *normal* — нормальный, *bold* — жирный текст, *italic* — наклонный текст, *underline* — подчернутый текст, *strkieout* — зачеркнутый текст.
style:bold,underline - подчеркнутый жирный текст.

fontsSize:*размер*[%] - размер шрифта. Если после размера шрифта указан символ % то размер будет считаться в процентах от размера шрифта по умолчанию (установленного через метод setPrinterSettings).

fontName:*имя шрифта* — печатать указанным шрифтом. Можно использовать все шрифты, которые входят в поставку ОС Android.

if:*условие* — печатать блок, если *условие* является истинным. *Условие* записывается как *lv* **сравнение** *rv,* где сравнение может быть = != > < >= <= (равно, не равно, больше, меньше, больше или равно, меньше или равно). Операции = != применимы как к строчным аргументам, так и к числовым, остальные применимы только к числовым аргументам. Строчные аргументы для валидности надо брать в ". Примеры параметра:
if:"Да" = "Да" - истина
if:"Да" = "Нет" - ложь
if:5!=10 - истина
if:5 > 10 - ложь

align:*выравнивание* — выравнивание текста по горизонтали. Значение *выравнивание* может быть **left, center, right**

valign:*выравнивание* — выравнивание текста по вертикали в пределах блока. Значение может быть **top, center, bottom**

Теги.

Для форматирования используются следующие теги

{**s**\текст} — определяет стиль для текста. Не имеет дополнительных параметров. Тег может быть вложеным.
{\s style:bold;\Этот жирный {\s style:italic\наклонный} текст}

{**p**\текст} — перенос текста на новую строку. Не имеет дополнительных параметров
Текст следующая строка{\p\будет автоматически перенесена}

{**table**\тело таблицы} — определение таблицы. Дополнительные параметры:
width:*ширина*[%] - указание ширины таблицы. Если указан символ %, то ширина считается от размера предыдущего блока (если table первый тег — то от размера страницы печати (384 точки - поля слева и справа).

border:*лево*[,*верх*[,*право*[,*низ*]]] — рамка вокруг таблицы. Указывается толщина линий рамки в пикселях

{**tr**\тело строки} — определяет строку таблицы. Может использоваться только внутри тега **table**. Дополнительных параметров нет. Внутри тега весь текст, кроме тега **td** игнорируется.

{**td**\текст} — определяет ячейку таблицы. Может использоваться только внутри тега **tr**. Дополнительные параметры:
width:*ширина*[%] - указание ширины ячейки. Если указан символ %, то ширина считается от размера предыдущего блока. Если в качестве значения использован символ * то ширина берется как оставшаяся.
padding:*лево*[,*верх*[,*право*[,*низ*]]] — отступы в ячейке.
border:*лево*[,*верх*[,*право*[,*низ*]]] — рамка вокруг ячейки. Указывается толщина линий рамки в пикселях

Таблица может содержать разное количество ячеек в разных строках. Пример таблицы

```
{\table\
  {\tr\{\td width:40;\Один}{\td width:*;align:right;\Два}}
  {\tr\{\td width:*;align:center;style:bold\Три}}
}
```

{**barcode**\текст} — сформировать штрихкод. Дополнительные параметры:
width:*ширина*[%] - указание ширины баркода. Если указан символ %, то ширина считается от размера предыдущего блока.
height:*высота* — высота баркода в пикселях. Если не указана, то для линейных баркодов используется 1/5 от ширины, для QR, DataMatrix — ширина
type:*code128*|*ean13*|*ean8*|*code39*|*code93*|*qr*|*dm* — тип баркода. По умолчанию используется Code128

{**image**\base64} — вывод изображения. Изображение передается в теле тега в формате JPG или PNG закодированное в base64. Дополнительные параметры
width:*ширина* — ширина изображения в пикселях.
height:*высота* — высота изображения в пикселях

Передача параметров фискальных документов в шаблоны печати

Для передачи параметров фискального документа в шаблон печати используется подстановка переменных. Переменная текст, ограниченный символами $. Пример: $signatrue.Date$, $owner.Name$

Для всех документов (кроме счетчиков ФН, класс FNCounters) доступны следующие общие переменные:

signature.Date — дата проведения документа
signature.Number — фискальный номер документа
signature.sign — фискальная подпись документа
device.Number — заводской номер ККТ
device.regNo — регистрационный номер ККТ
device.FN — номер ФН
FFD.KKT — версия ФФД поддерживаемая ККТ (всегда 1.2)
FFD.FN — версия ФФД поддерживаемая ФН
FFD.VER — используемая версия ФФД
device.Version — версия FNCore (всегда 003)
operator.Name — имя оператора, выполнившего операцию
operator.INN — ИНН оператора, выполнившего операцию (если указан)
Address — адрес проведения операции
Location — место проведения операции
owner.Name — имя владельца ККТ
owner.INN — ИНН владельца ККТ
warning.3days — предупреждение об окончании ресурса ФН (Да/Нет)
warning.full — предупреждение об исчерпании ресурса ФН (Да/Нет)
warning.30days — предупреждение об окончании действия ФН (Да/Нет)
automateNumber — номер автомата (если установлен режим «Принтер в автомате», иначе пустая строка)
sender_email — адрес отправителя чеков
fns_url — адрес сайта ФНС
T_xxx — значение произвольного тега в виде строки, xxxx - номер тега согласно ФФД. Можно использовать цепочку тегов разделяя их символом «.». Т.е. $T_1223.1005$ получить тег 1223 в виде STLV, а из него взять значение тега 1005 как строку.

Документ фискализации/изменения параметров ККТ

reason.Type — коды причин перегистрации ККТ
reason.Name — причина перегистрации/регистрации
bso — использование БСО (Да/Нет)
encryption — использование шифрования (Да/Нет)
isInternetMode — ККТ для Интернета (Да/Нет)
isServiceMode — оказание услуг (Да/Нет)
isExcisesMode — подакцызные товары (Да/Нет)
isCasinoMode — проведение азартных игр (Да/Нет)
isLotteryMode — проведение лотерей (Да/Нет)
isMarking — работа с маркированными товарами (Да/Нет)
isPawnShop — услуги ломбарда (Да/Нет)
isInsurance — страховые услуги (Да/Нет)

AgentType — тип агентских услуг (перечисление)
offline — ККТ оффлайн (Да/Нет)
automation — Принтер установлен в автомате (Да/Нет)
ofd.INN — ИНН ОФД
ofd.Name — наименование ОФД

Keys.Days.Remaning — ресурс ключей ОКП
Registation.Reason — причина регистрации
counters.total — счетчики ФН (заполняется таблицей)
is.counters.total — признак присутствия счетчиков ФН в документе

Документ открытие/закрытие смены

shift.NumDocuments — количество документов за смену
shift.Number — номер смены
shift.IsOpen — признак открытой смены (Да/Нет)
shift.NumChecks — количество чеков за смену
ofd.NumUnsent — количество неотправленных в ОФД документов
ofd.DateUnsent — дата первого неотправленного документа
ofd.FirstUnsentNo — номер первого неотправленного документа
oism.NumUnsent — количество неотправленных запросов в ОИСМ
mark.Incorrect — количество неверных кодов маркировки
mark.Incorrect.InFiscal — количество неверных кодов маркировки в ФН
counters.total — счетчики итогов ФН (таблица)
counters.shift — счетчики итогов смены (таблица)
is.counters.total — наличие счетчиков итогов ФН (Да/Нет) в документе
is.counters.shift — наличие счетчиков итогов смены (Да/Нет) в документе
mark.OKP.Update.No.Need — признак, что обновление ключей ОКП не требуется (Да/Нет)
mark.OKP.Update.Passed — признак, что обновление ключей ОКП прошло успешно (Да/Нет)
mark.OKP.Update.Error — признак, что обновление ключей ОКП прошло с ошибкой (Да/Нет)
mark.OKP.Update.Error.Fatal — признак, что обновление ключей ОКП невозможно (Да/Нет)
Keys.Days.Remaning — ресурс ФН в днях

Чек

Чек состоит из трех шаблонов — заголовка, шаблона предмета расчета (повторяющегося) и подвала.

Для заголовка и подвала доступны следующие переменные

order.Type — тип чека
order.Number — номер чека
order.Refund — сдача наличными
order.Barcode — QR код (текстовое значение)
order.AgentType — типа агентских услуг
Total — общая сумма по чеку
taxMode — система налогообложения
shift.Number — номер смены
order.Vat_xxx — значение НДС по чеку, где xxx — 20, 10, 10_110, 20_120,0, NONE
order.Sum.xxx — значение оплаты по типу, где xxx — CASH,

CARD,PREPAYMENT,CREDIT,AHEAD

В строке предмета расчета не действуют общие для всех документов переменные, кроме T_xxx. Для строки предмета расчета допустимы следующие переменные

item.name — наименование предмета расчета
item.qtty — количество предмета расчета
item.measure — единица измерения
item.price — цена за единицу
item.sum — стоимость
item.Vat.Name — наименование ставки НДС
item.Vat.Value — величина ставки НДС
item.PaymentType — способ расчета
item.ItemType — тип предмета расчета
item.AgentType — тип агентской услуги
item.MarkCode — признак кода маркировки
item.CheckCode — контрольная сумма кода маркировки (тег 2115)

Чек коррекции

Чек коррекции является обычным чеком, к которому добавляются следующие переменные

correction.Type — тип коррекции
correction.baseDocument — номер документа-основания
correction.baseDocument.Date — дата документа-основания

Отчет о состоянии расчетов

shift.Number — номер смены
shift.IsOpen — признак открытой смены (Да/Нет)
ofd.NumUnsent — количество неотправленных в ОФД документов
ofd.DateUnsent — дата первого неотправленного документа
ofd.FirstUnsentNo — номер первого неотправленного документа
offline — признак ККТ оффлайн (Да/Нет)
oism.NumUnsent — количество запросов не переданных в ОИСМ
counters.total — счетчики итогов ФН (таблица)

Документ перевода в постфискальный режим

automation — признак установки в автомате (Да/Нет)
shift.Number — номер последней смены
counters.total — счетчики итогов ФН (таблица)

Счетчики итогов

Для таблицы счетчики итогов используются следующие переменные

is.Total.Counters — является ли счетчик счетчиком итогов ФН (Да) или смены (Нет)
total.Bills — количество чеков
Income.xxx — суммы и количества по чекам прихода, где xxx: *count* — количество чеков,

totalSum — общая сумма, *totalSumCash* — сумма наличными, *totalSumCard* — сумма безналичными, *totalSumPrepayment* — сумма предоплат, *totalSumCredit* — сумма кредитов, *totalSumAhead* — сумма встречными, *totalSumVat_20* — сумма НДС по ставке 20%, *totalSumVat_10* - сумма НДС по ставке 10%, *totalSumVat_10_110* — сумма НДС по ставке 10/110, *totalSumVat_20_120* — сумма НДС по ставке 20/120, *totalSumVat_0* — суммы с НДС 0%, *totalSumvat_none* — суммы не облагаемые НДС

Outcome.xxx - суммы и количества по чекам расхода, значение xxx см. выше

ReturnIncome.xxx - суммы и количества по чекам возврата прихода, значение xxx см. выше

ReturnOutcome.xxx - суммы и количества по чекам возврата расхода, значение xxx см. выше

Correction.xxx - суммы и количества по чекам коррекции, значение xxx см. выше