Принцип декодирования с использованием FastScript

Также как в любом другом случае декодирование на стороне TCPCardReader'а включается строкой.

```
useDecodeCards=1
```

в TCPCardReader.ini в настройках конкретного считывателя:

```
[READER1]
COMPORT=6
COMPORTPARAMS=baud=9600 data=8 parity=N stop=1
Prefix=;
Postfix=?
useDecodeCards=1
StartTermChars=i
FinishTermChars=?#13;#10;#0;
```

Таким образом при включении декодирования TCPCardReader отправляет часть трека начинающегося с символа следующего за *StartTermChars* и заканчивающимся символом стоящим перед одним из *FinishTermChars в* Decodecards*.*

Итак Decodecards получает только эту часть и декодирует (преобразовывает/выделяет идентификатор) её.

В **decodecards.ini** В секции **[mask]** перечисляются маски по префиксам которых можно разделять декодирование разных карт

[mask]
card_em = ne*
card_mifare = fare*
card = *

В данном примере если пришедший от TCPCardReader'а трек содержит префикс "ne" ,то обработка такого трека будет описана в секции [cards.**card_em**]. Соответственно если трек содержит префикс "fare" то обработка такого трека описана в секции [cards.card_mifare]. Символ "*" в данном случае означает любое количество символов.

`Чтение данной секции происходит последовательно, то есть при первом совпадении пришедшего трека с маской дальнейшей просмотр секции не ведётся`

Далее в секциях с названием "cards." и левой частью до знака равенства в секции [mask] конкретного типа трека настраивается декодирование.

```
[cards.Card]
code=2
CardDecodeType = FastScript
DecodeTypeEx = 0
```

Опцией CardDecodeType = FastScript выбираем алгоритм декодирования с использованием FastScript.

В таком случаем алгоритм декодирования программируется в функции decodecard во внешнем файле, задаваемым параметром fsUnit в секции [frf].

```
[frf]
fsUnit=fsdecodecard.upas
```

Пример внешнего файла fsdecodecard.upas

```
function fsUnitVersion:integer;
begin
  result := 1;
end;

function DecodeCard(Track:string; var Code:integer):string;
begin
  if ( Length(track) >= 4 ) then
    result :1= copy( track, length(track)-3, 4 )
  else
    result := track;
end;

begin

end.
```

Т.е. при считывании идентификатора, для его декодирования будет вызвана функция DecodeCard() из данного файла.

Входные параметры функции:
Track - полный трек полученный от считывателя (типа string),
Code - код типа идентификатора (типа integer, подлежит изменению в теле функции);
Выходные параметры функции:
Result - номер идентификатора представленный строкой, после получения должен безошибочно конвертироваться в Int64 (типа string). Т.е результатом работы функции DecodeCard должна быть строка которая должна однозначно приводиться к 10-тиричному числу.

В файле скрипта доступны все функции доступные в печатных формах абонемента, что позволяет облегчить написание и отладку скрипта.
Пример печатной формы для тестирования разрабатываемого скрипта в приложении.

Вся обработка происходит в функции DecodeCard в которую передаются трек без *ExcludedPrefix* и *code* указанный в соответствующей секции файла *decodecards.ini.* Чаще всего это *code=2* как в данном примере. В функции можно использовать другие функции реализованные в этом же файле.

Далее TCPCardReader обрамит этот трек префиксом "Prefix=;" и суффиксом "Postfix=?" и отправит клиентам.

Третий закон Чизхолма[1].

Даже если вы считаете, что сделали прекрасную вещь, за которую вам все будут благодарны, всё равно найдётся человек, которому это не понравится, и он даже может рассердиться на вас

Версия #18
Игорь Докутович создал 31 August 2021 15:24:23
Игорь Докутович обновил 6 October 2022 12:23:41