Внимание!

Модуль распространяется как бесплатный и по приципу «как есть». Любая его продажа запрещена! Автор не несёт никакой ответственности за возможные потери с Вашей стороны при использовании данного кода. При использовании ссылка на страницу www.siseditor.at.ua обязательна!!!

Данная документация призвана облегчить понимание принципа работы с sis-файлами, так как файл имеет довольно запутанную структуру и кому-то может показаться сложным в понимании.

Для работы с sis необходимы:

- ***mainSISFields.pas*** - в нём реализованы классы для работы с SIS
- ***mainSISFunctions.pas*** - тут хранятся все необходимые функции для манипуляций с SIS-файлом
- ***mainDecriptKey.pas*** - модуль для дешифровки ключа *.key, необходимого для подписи sis-файла

Ещё мой модуль для работы с ASN1: **mainASN1.pas**

И **mainCRC.pas** для расчёта CRC16CCITT

Так же необходимы:

- **FGIntDSA.pas**, **FGIntRSA.pas**, **FGIntPrimeGeneration.pas**, **FGInt.pas** с сайта *http://www.submanifold.be* (необходимы для подписи приложений - Delphi 7 не умеет работать с большими числами)
- несколько модулей, которые были откуда-то скачаны и производилась их какая-то частичная модификация... Их я тоже выложу

Всё это было написано под Delphi 7, за работу в других версиях я не ручаюсь!

Перед использованием советую прочитать это: http://upwap.ru/543532

---

mainSISFields:

В этом модуле базовым классом является класс ***TSISBaseField.*** Сам по себе, этот класс ничего не умеет, в нём есть лишь общие для всех полей свойства - чтение типа поля, его длины, запись типа и длины, реализован доступ к вложенным полям, удаление/добавление их, контроль за корректностью чтения SIS-файла. У этого класса есть два абстрактных метода:

procedure ReadData(Source : TStream); Virtual; Abstract; - используется для чтения

специфичных данных каждого поля
procedure WriteData(Target : TStream); Virtual; Abstract; - используется для записи специфичных данных каждого поля

Эти процедуры должны быть обязательно реализованы (собственно, уже реализованы - без этого бы не работал siseditor) в классах-потомках.

Так же в этом классе есть:

constructor Create(Source:TStream; Parent:TSISBaseField; ReadType: boolean); OverLoad; Virtual; - метод для формирования класса по данным SIS файла, хранящихся в каком-либо потомке ***Source*** класса TStream (например, TMemoryStream или TFileStream). ***Parent*** - поле, внутри которого будет храниться создаваемое поле. ***ReadType*** - если True, то будет считываться тип поля (необходимо указывать True для всех полей, кроме SISArray)
constructor Create(Parent:TSISBaseField); Overload; Virtual; - создаёт структуру поля без опоры на какие-либо данные.
property TypeOfField : integer read FTypeOfField write FTypeOfField; - свойство с типом поля. Например, для массива (TSISArray) будет равно 2, для строки (TSISString) будет равняться 1, а для сжатого поля (TSISCompressed) будет равным 3. Все номера полей указаны в константах, начинающихся с **NSIS**, указанных в начале модуля.
property Parent : TSISBaseField read FParent write FParent; - думаю, уже понятно
property SIS : TSIS read FSIS write FSIS; - наш объект, который хранит в себе все данные (см. ниже)
procedure UpdateData(Target: TStream; WriteType, WriteLength: boolean); - используется для записи содержимого поля в поток **Target; WriteType** - если True, то пишется тип поля (полезно в массивах). **WriteLength** - если True, то записывается длина (полезно при получении данных для хэша контроллера при подписи).

На базе ***TSISBaseField*** реализован класс ***TSIS*** - класс, который как раз и занимается открытием, созданием и сохранением SIS.

TSIS	
Назначение:	Занимается открытием/сохранением sis-файлов
Расположение:	***Основной класс, без расположения***

Самое важное в этом классе:

constructor Create(FileName:string); overload; - конструктор класса, при вызове указывается имя открываемого файла *.sis или *.sisx. После выполнения этой команды происходит полное чтение SIS-файла с формированием структуры.
destructor Destroy; override; - уничтожает объект
function LoadFromPKG(PKGFilename:string):boolean; - позволяет создать SIS-файл по PKG-скрипту.
procedure SaveToFile(FileName:string); - сохраняет SIS
function PKGScript(Ctrl:TSISController):widestring; - позволяет получить PKG-скрипт по указанному контроллеру.
property SISContents : TSISContents read FSISContents write FSISContents; - свойство, позволяющее получить доступ к полю SISContents.
property Controller[Index : integer] : TSISController read Controllers; - позволяет получить доступ к любому контроллеру, лежащему внутри sis-файла. Число контроллеров

можно узнать с помощью свойства:
property ControllerCount : integer read GetControllerCount;
property FileName : string read FSISFileName; - позволяет узнать имя открытого sis-файла

После класса **TSIS** наиболее важным является класс ***TSISController***, в котором хранятся все основные данные о SIS-пакете.

TSISController	
Назначение:	Хранит в себе все данные о SIS, за исключением содержимого устанавливаемых файлов.
Расположение:	***SIS>SISContents>SISControllerCompressed>Controller*** ***или (для встроенных контроллеров)*** ***InstallBlock>EmbeddedSIS>Controller[N]***

procedure Sign(Key,Cer,Password:string); - с помощью этой процедуры происходит подпись с помощью сертификата. Key - имя файла ключа. Cer - имя файла сертификата. Password - пароль к файлу ключа. Если все переменные оставить пустыми, то контроллер будет подписан с помощью встроенного сертификата на 30 лет.
function SignVerify(var ReasonText:widestring):boolean; - обратная процедура - происходит проверка на корректность подписи. Если подпись корректна, то возвращает True. В переменную **ReasonText** заносится строка с причиной отказа (пока что не заносится)
property Certificates[Index:integer] : TSISSignatureCertificateChain read GetCertificate write SetCertificate; - позволяет получить доступ к сертификатам. Число сертификатов:
property CertificatesCount : integer read GetCertificatesCount;
procedure DeleteCertificate(Index:integer); - удаляет сертификат с индексом **Index**
procedure DeleteAllCertificate; - удаляет все сертификаты
property Info : TSISInfo read FSISInfo write FSISInfo; - поля с полной информацией о контроллере: название, версия, разработчик (продавец), дата создания и т.д.
property Options : TSISSupportedOptions read FSISSupportedOptions write FSISSupportedOptions; - список пунктов доп. компонентов, которые будут выводиться при установке.
property Languages : TSISSupportedLanguages read FSISSupportedLanguages write FSISSupportedLanguages; - список языков, которые поддерживает сис-файл (точнее, контроллер)
property Dependences : TSISPrerequisites read FSISPrerequisites write FSISPrerequisites; - список совместимостей с платформами и приложениями.
property InstallBlock : TSISInstallBlock read FSISInstallBlock write FSISInstallBlock; - то, что интересует большенство читающих это - список файлов хранится тут!
property DataIndex : TSISDataIndex read FSISDataIndex write FSISDataIndex; - а это поле с номером поля с данными файлов.

Перед тем, как узнать, что хранит в себе TSISInfo, необходимо познакомиться с другими классами. Итак,

TSISString - простейший класс, хранит в себе текст в формате Unicode. Полезное свойство одно:

property WString : WideString read FString write WriteString; - сам текст.

TSISArray - класс, являющийся, как следует из названия, массивом. Его свойства и методы:

property ArrayTypeOfField : integer read FArrayTypeOfField write FArrayTypeOfField;
- тип хранимых данных. Доступ к ним реализован с помощью свойств:

property Strings[Index : integer] : WideString read GetString write SetString;
property SupportedOptions[Index : integer] : TSISSupportedOption read GetOption write SetOption;
property Controller[Index : integer] : TSISController read GetCtrl write SetCtrl;
property IfBlock[Index : integer] : TSISIf read GetIf write SetIf;
property ElseIfBlock[Index : integer] : TSISElseIf read GetElseIf write SetElseIf;
property Language[Index : integer] : TSISLanguage read GetLng write SetLng;
property Dependency[Index : integer] : TSISDependency read GetDependency write SetDependency;
property FileDescription[Index : integer] : TSISFileDescription read GetFileDescription write SetFileDescription;
property SISProperty[Index : integer] : TSISproperty read GetProperty write SetProperty;
property DataUnit[Index : integer] : TSISDataUnit read GetDataUnit write SetDataUnit;
property FileData[Index : integer] : TSISFileData read GetFileData write SetFileData;
property Signature[Index : integer] : TSISSignature read GetSignature write SetSignature;

Соответственно, для доступа к элементу массива надо использовать какой-то один метод. Если мы знаем, что **ArrayTypeOfField=NSISString**, то используем для доступа **property Strings**, а если **ArrayTypeOfField=NSISController**, то используем для доступа **property Controller** и т.д.

Если вам не нравится такой подход, могу посоветовать так:
у класса **TSISBaseField** есть свойство

property Item[index:integer]: TSISBaseField read GetItem write SetItem;

Получаем с помощью него нужный элемент массива, например:

```
procedure Test(Arr:TSISArray);
var
Str:TSISString;
begin
if Arr.ArrayTypeOfField=NSISString then Str:=TSISString(Arr.Item[0]);
end;
```

TSISVersion - класс для работы с полем SISVersion. Сама версия хранится в записи:

```
CSISVersion = record
major : integer;
minor : integer;
build : integer;
end;
```

Методы класса:

property Version : CSISVersion read FVersion write WriteVersion; - доступ к FVersion : CSISVersion;
property StringVersion : string read SISVersionToString write WriteStringVersion; - позволяет работать с версией в виде стоки вида "major.minor(build)". Например, "1.0(345)"

TSISDate - для работы с датой.

Свойство одно:

property TDate : TDateTime read GetTDate write SetTDate; - позволяет получить или записать дату в стандартном формате TDateTime

TSISTime - аналогично TSISDate

property TTime : TDateTime read GetTTime write SetTTime;

TSISDateTime - внутри него хранятся поля SISDate и SISTime

property Date : TSISDate read FDate write FDate;
property Time : TSISTime read FTime write FTime; - с этим, я думаю, всё понятно уже;)
property DateTime : TDateTime read GetTDateTime write SetTDateTime; - упрощение небольшое при работе с вложенными полями. Вместо указания отдельно времени и отдельно даты просто задаём сюда полную дату и класс сам всё сделает.

TSISUID - работа с полем типа SISUID - позволяет получить UID приложения

property UID : integer read FValue write SetUID;

Итак, а теперь перейдём к классу ***TSISInfo:***

TSISInfo	
Назначение:	Хранит информацию о установочном файле: имя разработчика, название программы, версию, дату создания файла, тип пакета, флаги
Расположение:	***SIS>SISContents>SISControllerCompressed>Controller>Info*** ***или (для встроенных контроллеров)*** ***InstallBlock>EmbeddedSIS>Controller[N]>Info***

Свойства класса:

property UID:TSISUID read FUID write FUID; - UID приложения
property VendorName : TSISString read FVendorName write FVendorName; - так называемое "Уникальное имя продавца (Unique Vendor Name)"
property ProgramNames : TSISArray read FNames write FNames; - массив названий программы на разных языках
property VendorNames : TSISArray read FVendorNames write FVendorNames; - VendorName на разных языках
property Version : TSISVersion read FVersion write FVersion; - версия приложения
property CreationTime : TSISDateTime read FCreationTime write FCreationTime; - дата создания

property InstallType : byte read FInstallType write SetInstallType; - тип пакета. Может быть всего 5 значений:

SA (значение в InstallType: 0) - обычное приложение
SP (1)- патч, позволяет дополнить установленное приложение, но не умеет менять уже установленные компоненты. Не удаляется при переустановке приложения.
PU (2)- частичное обновление. Может заменить любой файл программы. Удаляется при переустановке приложения.
PI (3) - вроде как обновляет предустановленные программы
PP (4) - патч для предустановленных программ

property InstallFlags : byte read FInstallFlags write SetInstallFlags; - если 1, то закрываются все программы перед удалением.

TSISInstallBlock	
Назначение:	Хранит информацию о файлах, встроенных SIS-файлах, ветвлениях
Расположение:	***Controller>InstallBlock*** ***или*** ***InstallBlock>IfBlocks>IfBlock[N]>InstallBlock*** ***или*** ***InstallBlock>IfBlocks>IfBlock[N]>ElseIfs>ElseIfBlock[K]>InstallBlock*** ***и т.д.***

property Files : TSISArray read FFiles write FFiles; - массив с описанием файлов
property EmbeddedSIS : TSISArray read FEmmeddedFiles write FEmmeddedFiles; - массив со встроенными контроллерами (так называемые Embedded SIS)
property IfBlocks : TSISArray read FIfBlocks write FIfBlocks; - массив с ветвлениями

TSISFileDescription	
Назначение:	Хранит информацию о файле
Расположение:	***InstallBlock>Files>FileDescription[N]***

property FileIndex : integer read GetFileIndex; - порядковый номер файла в контроллере
property SourseName : string read FFileSourseName write FFileSourseName; - имя исходного файла.
property TargetName : TSISString read FTarget write FTarget; - имя файла назначения
property MIMEType : TSISString read FMIME write FMIME; - для графики можно указать MIME-тип, например “MIME/png”
property Hash : TSISHash read FHash write FHash; - хэш файла. Служит для защиты от подмены файлов и проверки целостности архива
property Capabilities : TSISCapabilities read FCapabilities write FCapabilities; - это поле есть только у файлов типа **.exe, *.dll*
property OtherData : CFileDescription read FFileDescription write FFileDescription; - дополнительная информация о файле. Для домтупа к ней есть свойства:

property Operation : integer read FFileDescription.operation write FFileDescription.operation; - операции, которые выполнятся при установке:

Operation=1 – просто установить файл
Operation=2 – запустить файл

Operation=4 – вывести файл на экран как текст
Operation=8 – файл «пустой», просто при установке создаётся на диске.

property OperationOptions : integer read FFileDescription.operationoptions write FFileDescription.operationoptions; - дополнительные опции для операций.

Операция	Бит, соответствующий флагу опции	Действие
2	1	Запустить файл при установке
2	2	Запустить файл при удалении
2	3	Запустить, используя MIME-тип
2	4	Запустить и подождать завершения
2	5	Закрыть после окончания установки
3	9	Вывести текст с кнопкой «Продолжить»
3	10	Вывести текст с кнопками «Да» и «Нет». Если пользователь выберет «Нет», то пропустить установку следующего файла.
3	11	Вывести текст с кнопками «Да» и «Нет». Если пользователь выберет «Нет», то отменить установку
3	12	Вывести текст с кнопками «Да» и «Нет». Если пользователь выберет «Нет», то выйти из установки.

property Index : integer read FFileDescription.index write FFileDescription.index; - порядковый номерданных файла в массиве TSISDataUnit

Продолжение следует...