Модуль "Script execute"
РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Содержание

Раздел 1 Введение 1

Раздел 2 Системные требования 1

Раздел 3 Установка Script execute 2

Раздел 4 Глоссарий 3

Раздел 5 Настройка 4

Раздел 6 Список функций 5

1 Отсутствующий функционал 5
2 Типы данных 6
3 Математические 6
4 Строковые 7
5 Дата и время 7
6 Прочее 7
7 Специальные 8

Раздел 7 Проблемы? 10

1 Возможные проблемы 10

1 Введение

Модуль фильтрации данных "Script execute" для наших логгеров (например, Advanced Serial Data Logger или Advanced TCP/IP Data Logger) включает в себя простую реализацию скриптовых движков для наиболее популярных языков программирования. Уникальная возможность это модуля это поддержка языков PascalScript, C++Script, JScript или BasicScript в одном модуле.

Модуль не использует Microsoft Scripting Host, встроенный в Windows, поэтому может выполняться на любом компьютере с Windows.

Скриптовой движок "Script execute" работает быстро и включает в себя богатый набор функций на все случаи жизни. Это позволяет подстроить приложение практически под любую задачу.

Если вы знаете любой из поддерживаемых языков программирования, то вы очень быстро напишете нужный вам скрипт. Внутри ваших скриптов вы можете оперировать с переменными парсера, добавлять новые переменные и отправлять данные в источник данных (например, последовательный порт).

Возможности:
- Поддержка несколько языков программирования: PascalScript, C++Script, BasicScript, JScript.
- Стандартный набор для каждого языка: переменные, константы, процедуры, функции (включая вложенные функции, каждая из которых может иметь внутренние переменные, константы и параметры по умолчанию), все стандартные операторы и выражения (включая case, try/finally/except, with), типы данных (int, float, bool, char, string, многомерные массивы, enum, variant), классы (с методами, событиями, свойствами и т.п.).
- Проверка совместимости типов данных.
- Доступ к стандартным классам (например, TStrings, TFileStream).
- Возможность сохранения значений между каждым выполнением скрипта.
- Позволяет отправлять строки, байты, массивы данных.
- Может использоваться для нескольких источников и в нескольких конфигурациях одновременно.

2 Системные требования

Для установки Script execute должны быть выполнены следующие требования:

Операционная система: Windows 2000 SP4 и выше, включая версии для 32-х и 64-х битных систем, а также серверные версии операционных систем.
Наличие установленного последнего сервис-пака желательно.

Свободное дисковое пространство: Рекомендуется не менее 5 MB свободного дискового пространства.

Специальные требования для доступа: Вам потребуются права Администратора для установки модуля. Для некоторых модулей могут потребоваться права администратора для выполнения заданных функций.

Необходимо наличие главного приложения (ядра), например Advanced Serial Data Logger.

3 Установка Script execute

1. Закройте главное приложение (например, Advanced Serial Data Logger), если оно запущено;
2. Скопируйте программу на ваш жесткий диск;
3. Запустите программу установки модуля, кликнув два раза на имени файла в проводнике Windows;
4. Следуйте инструкциям программы установки. Обычно достаточно нажать несколько раз кнопку "Дальше";
5. Запустите главное приложение. В случае успешной установки название модуля появится в окне настройки, на закладке "Модули".

Если модуль совместим с программой, то его название и номер версии будет отображаться в списке модулей. Примеры установленных модулей можно посмотреть на рис. 1-2. Некоторые типы модулей требуют дополнительной настройки. Для этого достаточно выбрать модуль из списка и нажать кнопку "Настроить" рядом со списком. Процедура настройки модуля описана в следующих главах.

Некоторые типы модулей видны на закладке "Лог-файл". Для их настройки необходимо выбрать модуль из списка "Тип файла" и нажать кнопку "Дополнительно".

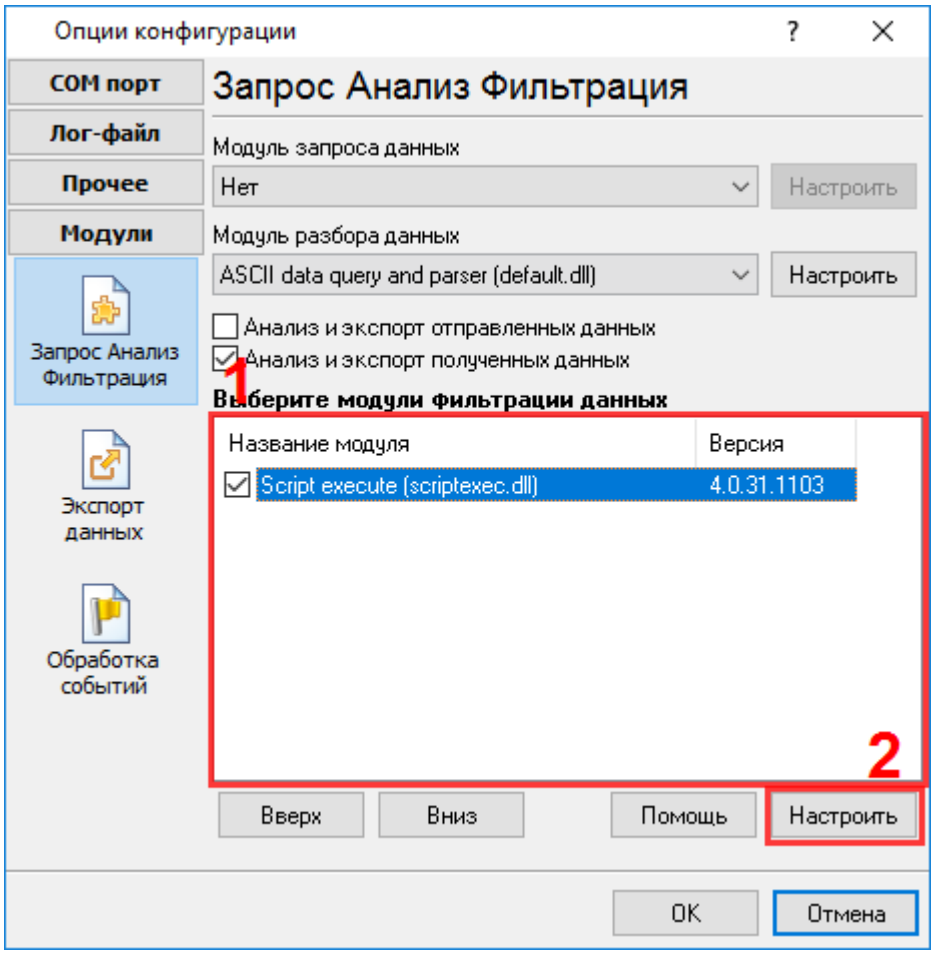

Рис. 1. Пример установленного модуля

4 Глоссарий

Plug-in - модуль

Главная программа - программная оболочка, которая использует данный модуль. Например: Advanced Serial Data Logger

Основная программа - см. "Главная программа".

Парсер - модуль, который обрабатывает поток данных, выделяя из него пакеты данных и переменные из пакетов данных. Затем эти переменные используются в модулях экспорта данных.

Ядро - см. "Главная программа".

5 Настройка

Настройка модуля очень проста (рис.1). Вы должны выбрать тип языка программирования скрипта и набрать текст скрипта в редакторе. Вы можете добавляться функции в позицию курсора дважды кликнув на ее имени в дереве справа.

Основные функции описаны в следующих разделах. Описание функции, отсутствующей в файле справки можно найти в Гугле по запросу "Pascal Имя_Функции"

Когда вы завершили редактирование, то кликните на кнопке "Проверить" и проверьте правильность составления скрипта. Вы можете кликнуть "OK" и сохранить все изменения.

Вы можете загрузить скрипт из файла воспользовавшись соответствующей кнопкой.

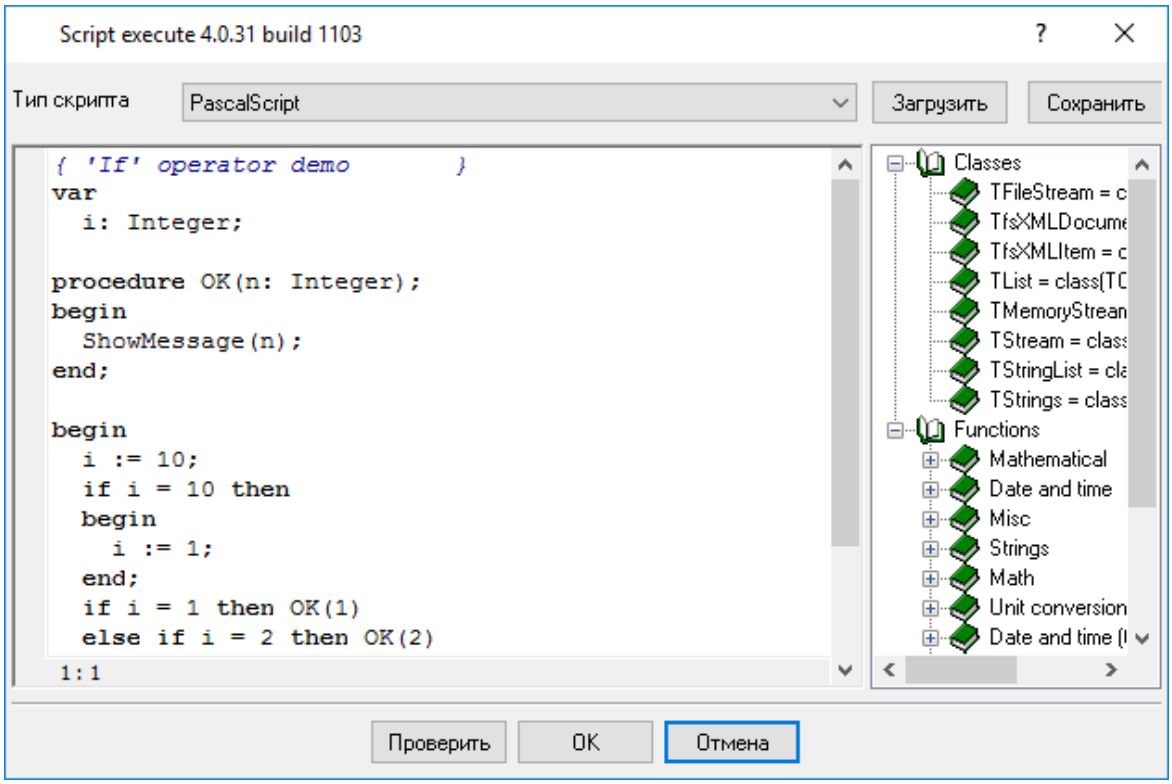

Рис.1. Окно настройки

Горячие клавиши редактора

Key	Action
Cursor arrow	Cursor moving
PgUp, PgDn,	Page Up / Page Down
Ctrl+PgUp	Move to the begin of text
Ctrl+PgDn	Move to the end of text

Home	Move to the begin of line
End	Move to the end of line
Enter	Move to the next line
Delete	Delete symbol at right or selected text
Backspace	Delete symbol at left
Ctrl+Y	Delete current line
Ctrl+Z	Undo last change
Shift+Стрелки курсора	Select the text block
Ctrl+A	Select all text
Ctrl+U	Unindent selected block
Ctrl+I	Indent selected block
Ctrl+C, Ctrl+Insert	Copy to clipboard
Ctrl+V, Shift+Insert	Paste from clipboard
Ctrl+X, Shift+Delete	Cut to clipboard
Ctrl+Shift+<цифра>	Set bookmark
Ctrl+<цифра>	Goto bookmark
Ctrl+F	Search text
F3	Continue search

6 Список функций

6.1 Отсутствующий функционал

- Нельзя объявлять свои типы данных (записи и классы) в теле скрипта; нельзя использовать указатели, записи и sets (но можно использовать оператор 'IN' для конструкций вида "a in ['a'..'c','d']"),
- Нельзя использовать тип shortstring
- Не поддерживается оператор GOTO.
- C++Script: нет восьмеричных констант; нет 'break' в операторе SWITCH (SWITCH работает как CASE в Pascal); '++' и '--' могут использоваться только после переменных, (нельзя написать '++i'); Операторы '--', '++' и '=' не возвращают значения поэтому нельзя написать 'if(i++)'; Все идентификаторы регистронезависимые.

6.2 Типы данных

Внутри себя модуль оперирует типом Variant для всех значений, поэтому допустимы только типы, производные от него:

```
Byte      | Тип Integer
Word      |
Integer   |
Longint   |
Cardinal  |

Boolean   | Тип Boolean

Real      | Тип Extended
Single    |
Double    |
Extended  |
TDate     |
TTime     |
TDateTime |

Char      | Тип Char

String    | Тип String

Variant   | Тип Variant
Pointer   |

Array     | Массивы
```

C++Script приводит стандартные типы как:

int, long = Integer
void = Integer
bool = Boolean
float = Extended

JScript не имеет типов, все значения имеют тип variants.

BasicScript может иметь типы данных (например, dim i as Integer), или может не использовать типы и объявлять переменные. Тогда для переменной будет использоваться тип Variant.

Не все вышеперечисленные типы могут быть присвоено друг другу. Например, вы не можете присвоить типы Object Pascal Extended или String переменной типа Integer. Только тип Variant может принимать значения всех типов.

6.3 Математические

```
Abs(e: Extended): Extended
ArcTan(X: Extended): Extended
Cos(e: Extended): Extended
Exp(X: Extended): Extended
```

```
Frac(X: Extended): Extended
Int(e: Extended): Integer
Ln(X: Extended): Extended
Pi: Extended
Round(e: Extended): Integer
Sin(e: Extended): Extended
Sqrt(e: Extended): Extended
Tan(X: Extended): Extended
Trunc(e: Extended): Integer
```

6.4 Строковые

```
Chr(i: Integer): Char
CompareText(s, s1: String): Integer
Copy(s: String; from, count: Integer): String
Delete(var s: String; from, count: Integer)
Insert(s: String; var s2: String; pos: Integer)
Length(s: Variant): Integer
Lowercase(s: String): String
NameCase(s: String): String
Ord(ch: Char): Integer
Pos(substr, s: String): Integer
SetLength(var S: Variant; L: Integer)
Trim(s: String): String
Uppercase(s: String): String
```

6.5 Дата и время

```
Date: TDateTime
DayOfWeek(aDate: TDateTime): Integer
DaysInMonth(nYear, nMonth: Integer): Integer
DecodeDate(Date: TDateTime; var Year, Month, Day: Word)
DecodeTime(Time: TDateTime; var Hour, Min, Sec, MSec: Word)
EncodeDate(Year, Month, Day: Word): TDateTime
EncodeTime(Hour, Min, Sec, MSec: Word): TDateTime
IsLeapYear(Year: Word): Boolean
Now: TDateTime
Time: TDateTime
```

6.6 Прочее

Other

```
CreateOleObject(ClassName: String): Variant
Dec(var i: Integer; decr: Integer = 1)
Inc(var i: Integer; incr: Integer = 1)
InputBox(ACaption, APrompt, ADefault: string): string
InputQuery(ACaption, APrompt: string; var Value: string): Boolean
MessageDlg(Msg: string; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons; HelpCtx: Longint): Integer
RaiseException(Param: String)
```

```
Random: Extended
Randomize
ShowMessage(Msg: Variant)
ValidDate(cDate: String): Boolean
ValidFloat(cFlt: String): Boolean
ValidInt(cInt: String): Boolean
VarArrayCreate(Bounds: Array; Typ: Integer): Variant
VarType(V: Variant): Integer
```

Conversion

```
DateTimeToStr(e: Extended): String
DateToStr(e: Extended): String
FloatToStr(e: Extended): String
IntToStr(i: Integer): String
StrToDate(s: String): Extended
StrToDateTime(s: String): Extended
StrToFloat(s: String): Extended
StrToInt(s: String): Integer
StrToTime(s: String): Extended
TimeToStr(e: Extended): String
VarToStr(v: Variant): String
```

Formatting

```
Format(Fmt: String; Args: array): String
FormatDateTime(Fmt: String; DateTime: TDateTime): String
FormatFloat(Fmt: String; Value: Extended): String
FormatMaskText(EditMask: string; Value: string): string
```

6.7 Специальные

```
procedure SetVariable(Name: String; Value: Variant)
```

Процедура изменяет или добавляет переменную парсера в пакет данных. Далее эта переменная может использоваться в модуля экспорта данных.

```
function GetVariable(Name: String):Variant
```

Эта функция возвращает значение переменной из пакета данных. Если переменная с указанным именем отсутствует, то будет возвращено значение Null.

```
procedure PushVariable(Name: String; Value: Variant)
```

Процедура сохраняет значение во внутренней памяти модуля. Значение этой переменной не попадает в пакет данных, но может быть получено при следующем выполнении скрипта. Если переменная уже есть во внутреннем хранилище, то ее значение будет обновлено. Внутреннее хранилище очищается при перезапуске программы или изменении конфигурации модуля.

```
function PopVariable(Name: String):Variant
```

Функция возвращает ранее сохраненное значение переменной с помощью процедуры PushVariable. Если переменная с указанным именем отсутствует, то будет возвращено значение Null.

```
procedure DeleteVariable(Name: String)
```

Процедура удаляет переменную из пакета данных.

```
procedure DiscardDataPacket
```

Процедура полностью отбрасывает пакет. Дальнейшая обработка этого пакета другими модулями прекращается.

```
function IsVariableDefined(Name: String):boolean
```

Эта функция позволяет проверить, если переменная с указанными именем в пакете данных или нет.

```
function IsVariableStored(Name: String):boolean
```

Эта функция позволяет проверить, если переменная с указанными именем во внутренней памяти модуля или нет.

```
function SendData(Data: String):boolean
```

Данная функция отправляет блок данных в источник данных (например, последовательный порт). Например, Data = #01+'Test'+#02

```
function SendString(Data: String):boolean
```

Данная функция отправляет строку в источник данных. Перед отправкой логгер заменит все вхождения вида #0D на ASCII символ с соответствующим шестнадцатеричным кодом.

```
function SendByte(Data: Byte):boolean
```

Данная функция отправляет один байт.

```
function SendByte(Data: Byte):boolean
```

Данная функция отправляет один байт.

```
function SendByte(Data: Byte):boolean
```

Данная функция отправляет один байт.

```
function SendDataToDataSource(DataSource, Data: String):boolean
```

Аналогична функции `SendString`, но можно указать источник данных, в который будут отправлены данные. Значение `DataSource` можно увидеть в главном окне, в выпадающем списке.

```
procedure DbgLog(Msg: String; LogLevel: Byte = 0)
```

Процедура отправляет сообщение Msg в лог сообщений программы. LogLevel может принимать значения: 0 - ошибка, 1 - предупреждение, 2 - информация.

```
procedure SendEvent(Name: String)
```

Процедура генерирует событие с указанным идентификатором.

```
procedure SendEventEx(Name: String; Args: array of const)
```

Процедура генерирует событие с указанным идентификатором и списком параметров.

Пример: `SendEventEx('EVENT-ID', ['ValueName1', 0, 'ValueName2', 'Test'])`

```
procedure RedirectData(TargetConfig: String)
```

Процедура перенаправляет переменные в другую конфигурацию, с указанным именем.

```
function GetVariableCount:integer
```

Функция возвращает количество переменных в обрабатываемом пакете данных. Эту и следующие две функции можно использовать для перебора всех переменных в пакете.

```
function GetVariableByIndex(AIndex: integer):Variant
```

Функция возвращает значение переменной по ее индексу в обрабатываемом пакете данных.

```
function GetVariableNameByIndex(AIndex: integer):string
```

Функция возвращает имя переменной по ее индексу.

```
procedure NewRow()
```

Процедура начинает формирование нового списка переменных для нового пакета данных. Все дальнейшие операции будут осуществляться только над переменными в этом новом пакете.

7 Проблемы?

7.1 Возможные проблемы

Модуль отсутствует в списке или помечен как "не установлен" - убедитесь, что модуль был установлен нужную папку. Все модули должны располагаться в подпапке "Plugins" в папке с программой. Также необходимо убедиться, что модуль совместим с вашей версией логгера. Если модуль несовместим, то сообщение об этом должно появиться в логе сообщений программы в главном окне.

Нет данных для публикации (экспорта) – никаких данных не было передано для экспорта. Решение: настройте парсер, убедитесь, что в парсере объявлена одна или несколько переменных.

Ошибка при присваивании значения переменной или параметра %s [%s] – ошибка обычно возникает, если данные не соответствуют указанному формату. Например, формат даты или времени не соответствует данным.

Ошибка при формировании параметра (%s) – программа не может преобразовать переменную одного формата в другой. Это проблема возникает, когда вы пытаетесь произвести экспорт переменной одного типа данных в столбец базы данных, который имеет другой

несовместимый тип данных. Для исправления ошибки измените тип данных в парсере или в модуле экспорта данных.

Не удалось соединиться %s. (%s) – ошибка возникает, когда модуль не может подсоединиться к базе данных. Проверьте ваши настройки соединения в модуле экспорта данных.

Не удалось отсоединиться %s. (%s) – ошибка возникает, когда модуль не может отсоединиться от базы данных. Обычно, ошибка возникает при нарушении связи с базой данных (нет сети, база данных недоступна).

Если у вас возникли другие проблемы, то пишите на support@aggsoft.ru. Все решим в кратчайшие сроки.

Замечание: в тексте ошибки выше, выражение "%s" будет заменено на дополнительные данные.