

Глава 5. Взаимодействие с внешними объектами

Внешние источники данных	1
ОткрытьDSN	1
Запрос	2
Закреть – закрытие DSN	3
Следующий – для запроса	3
Закреть – закрытие запроса	4
СОМ – объект	4
ВнешнийОбъект	4
ПросмотрМетодов	5
ВыполнитьМетод	6
Последовательный порт	7
ПоследовательныйПорт	7
СписокПоследовательныхПортов	7
Открыть – открытие порта	8
Закреть – закрытие порта	9
Кодировка – для порта	9
ЗаписатьСтроку – для порта	10
Работа с реестром операционной системы	11
ЗначениеКлючаРеестра	11
СписокИменКлючейРеестра	11
РеестрСписокЗначенийКлючей	12
Оборудование	13
ИмяКомпьютера	13
НайтиОборудование	13
СообщитьРабочееМесто	14
СписокПринтеров	15
ДляВсех(Устройств)	16
Прочие функции	17
Запустить	17
Заснуть	19

Внешние источники данных

ОткрытьDSN

Открывает объект типа DSN.

Синтаксис

`ОткрытьDSN(ИмяDSN, Пользователь, Пароль[, ТаймАут]`

Параметры



ИмяDSN (текст) - название источника данных.

Пользователь (текст) - имя пользователя, зарегистрированного в базе данных DSN. Если ни один пользователь не зарегистрирован, то в качестве параметра следует указать пустую строку - «».

Пароль (текст) - пароль пользователя.

ТаймАут (целое) - количество секунд, через которое запрос о соединении прекратит выполняться, если не завершился ранее.

Возвращает

Объект типа DSN.

Комментарии

DSN (Data Source Name) - имя источника данных (логическое имя, которое используется приложениями, чтобы получить связь с источником данных через ODBC - открытый интерфейс доступа к базам данных, встроенный в Windows и Windows NT, определяет набор функций, которые можно использовать для доступа к любой реляционной СУБД).

См. также:

- [Запрос](#)
- [Закреть – закрытие DSN](#)

Запрос

Возвращает объект типа запрос.

Синтаксис

`ОбъектDSN.Запрос (ЗапросSQL)`

Парамтры

ЗапросSQL (текст) - строка запроса на языке SQL.

Возвращает

Объект типа запрос.



См. также:

- [Следующий – для запроса](#)
- [Закрыть – закрытие запроса](#)

Закрыть – закрытие DSN

Закрывает соединение с источником данных.

Синтаксис

Объект DSN . **Закрыть** ()

Параметры

Не указываются.

Возвращает

Ничего не возвращает.

См. также:

- [Открыть DSN](#)

Следующий – для запроса

Извлекает очередную запись.

Синтаксис

oЗапрос . **Следующий** ()

Парамтры

Не указываются.

Возвращает

- 1 – удалось извлечь очередную запись.
- 0 – не удалось извлечь очередную запись.

См. также:

- [Запрос](#)



- [Закреть – для запроса](#)

Закреть – закрытие запроса

Закрывает запрос.

Синтаксис

oЗапрос . **Закреть** ()

Параметры

Не указываются.

Возвращает

Ничего не возвращает.

См. также:

- [Запрос](#)
- [Следующий – для запроса](#)

COM – объект

ВнешнийОбъект

Позволяет обращаться к внешним объектам операционной системы из программы СБИС.

Синтаксис

ВнешнийОбъект (ИмяОбъекта)

Параметры

ИмяОбъекта (текст) – имя Com – объекта (например, Excel.Application, Word.Application и т.д.)

Возвращает

Объект - Com – объект с имеющимися свойствами и методами.

Примеры

С помощью программы изменим в книге Excel значение ячейки, не открывая приложения:

```
перем экс = ВнешнийОбъект("Excel.Application"); #  
создаем Com-объект  
перем books = экс.GetWorkbooks(); # создаем объект  
книги  
перем book = books.FuncOpen(СпроситьФайл("",  
"*.*xls")); # открываем нужную книгу  
перем sheets = book.GetWorksheets(); # создаем листы  
перем ws = sheets.GetItem( 1 ); # выбираем первый  
лист  
перем cls = экс.GetCells(); # считываем ячейки из  
выбранного листа  
cls.GetItem( 1, 1 ).PutValue( "test" ); # в первую  
ячейку вносим значение «test»  
экс.PutVisible( Да ); # делаем книгу видимой для  
пользователя  
экс.FuncQuit(); # закрываем книгу, с сохранением  
изменений.
```

ПросмотрМетодов

Выводит список свойств и методов Com – объекта.

Синтаксис

```
Com-Объект.ПросмотрМетодов ( )
```

Параметры

Не указываются.

Комментарии

При обращении в программе к свойству Com – объекта необходимо написать «Get» (например, «GetCells»), при к методу – «Func» (например, «FuncOpen»). Для присвоения свойству значения необходимо приписать «Put» (например, «PutValue("test")»). При написании метода или свойства обращать внимание на регистры.



Возвращает

Да (логическое значение) – список со свойствами и методами возможно вывести.

Нет (логическое значение) – список со свойствами и методами невозможно вывести.

Примеры

Необходимо вывести список свойств и методов Сом-объекта «Excel.Application»:

```
перем экс = ВнешнийОбъект("Excel.Application"); #  
создание Сом-объекта;  
exs.ПросмотрМетодов();
```

ВыполнитьМетод

Позволяет вызывать методы внешнего объекта (ActiveX и COM-объекты), поддерживающего интерфейс IDispatch.

Синтаксис

```
Сом-Объект.ВыполнитьМетод(НазваниеМетода [ ,  
Параметр1] . . . )
```

Параметры

НазваниеМетода (текст) - имя метода внешнего объекта, который требуется выполнить.

Параметр1, Параметр2, ... (текст) - параметры метода внешнего объекта.

Возвращает

Значение любого типа, в зависимости от реализации вызываемого метода внешнего объекта.

Примеры

```
перем v = ВнешнийОбъект("V77.Application"); #  
создаем ActiveX-объект 1с 7.7  
перем rm = v.ВыполнитьМетод("RMTrade"); #  
инициализация RMTrade  
перем init_res = v.ВыполнитьМетод("Initialize", rm,
```



"" , ""); # Создается COM-объект 1С 7.7, работающий из под СБиС++

Последовательный порт

ПоследовательныйПорт

Возвращает запись объекта - последовательный порт. Функция используется для работы с дисплеем покупателя.

Синтаксис

ПоследовательныйПорт ()

Параметры

Не указываются.

Возвращает

Объект – запись объекта - последовательный порт.

См. также:

- [Открыть – открытие порта](#)
- [Закрыть – закрытие порта](#)
- [Кодировка – для порта](#)
- [ЗаписатьСтроку – для порта](#)

СписокПоследовательныхПортов

Возвращает список последовательных портов.

Синтаксис

СписокПоследовательныйПорт ()

Параметры

Не указываются.

Возвращает



Массив, содержащий строки с названиями включенных последовательных портов в системе. Отображаются все включенные порты, в том числе и уже занятые устройствами.

Пример

Необходимо узнать названия включенных последовательных портов в операционной системе Windows:

```
мСписок = СписокПоследовательныхПортов ( ) ;  
перем пНом = 0 ;  
Пока (пНом++<Размер (мСписок) )  
    Сообщить (мСписок [пНом] ) ;
```

См. также:

- [Открыть – открытие порта](#)
- [Закреть – закрытие порта](#)
- [Кодировка – для порта](#)
- [ЗаписатьСтроку – для порта](#)

Открыть – открытие порта

Открывает доступ к порту для передачи данных.

Синтаксис

Порт.**Открыть** (**ИмяПорта**, **СкоростьПередачи**,
РазмерБайта, **КонтрольЧет**, **СтоповыеБиты**)

Параметры

ИмяПорта (текст) – имя порта. Возможные варианты имени: "COM1", "COM2", "COM3", "COM4".

СкоростьПередачи (число) – скорость передачи данных. Возможные варианты скорости: 110, 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 56000, 57600, 115200, 128000, 256000.

РазмерБайта (число) - количество бит в байте при передаче и приеме данных.



КонтрольЧет (логическое значение) – контроль четности. Если указано значение параметра «Да»

СтоповыеБиты (число) – стоповые биты. Возможные варианты при указанном количестве бит в байте: 1 бит - 0, 1.5 бита - 1, 2 бита – 2.

Возвращает

Да (логическое значение) – последовательный порт открыт.

Нет (логическое значение) – последовательный порт закрыт.

См. также:

- [ПоследовательныйПорт](#)
- [Закреть – закрытие порта](#)

Закреть – закрытие порта

Закрывает доступ к порту.

Синтаксис

Порт . **Закреть** ()

Параметры

Не указываются.

Возвращает

Да (логическое значение) – если доступ к порту закрыт.

Нет (логическое значение) – если доступ к порту не закрыт.

См. также:

- [ПоследовательныйПорт](#)
- [Открыть – открытие порта](#)

Кодировка – для порта

Устанавливает кодовую страницу для выводимых в порт символов. Эта функция действует на функцию «ЗаписатьСтроку».



Синтаксис

Порт . Кодировка (КодоваяСтраница)

Параметры

КодоваяСтраница (текст) – желаемый вид кодировки. Для Windows - WIN, 1251; для DOS - ДОС, DOS, 866.

Возвращает

Да (логическое значение) – если кодовая страница установлена;

Нет (логическое значение) – если кодовая страница не установлена.

См. также:

- [ПоследовательныйПорт](#)

ЗаписатьСтроку – для порта

Выводит указанную строку в порт.

Синтаксис

Порт . ЗаписатьСтроку (Строка)

Параметры

Строка (текст) - текст, который будет выводиться в порт.

Возвращает

Целое – количество выводимых в порт символов.

Примеры

```
порт1 = ПоследовательныйПорт ();  
порт1.Открыть ("COM1", 9600, 8, нет, 0);  
порт1.Кодировка (866);  
порт1.ЗаписатьСтроку ("Приветствие для покупателя");  
порт1.Закрыть ();
```

См. также:

- [ПоследовательныйПорт](#)



Работа с реестром операционной системы

ЗначениеКлючаРеестра

Возвращает значение указанного ключа раздела реестра операционной системы Windows.

Синтаксис

ЗначениеКлючаРеестра (ИмяРаздела , ИмяКлюча)

Параметры

ИмяРаздела (текст) – имя раздела, в котором находится ключ.

ИмяКлюча (текст) – имя ключа.

Возвращает

Текст – значение указанного ключа.

Пример

Необходимо узнать значение ключа «Галочки»:

```
пВетка = "HKEY_CURRENT_USER\\Control  
Panel\\Patterns";  
Сообщить (ЗначениеКлючаРеестра (пВетка , "Галочки") ) ;
```

См. также:

- [СписокИменКлючейРеестра](#)
- [РеестрСписокЗначенийКлючей](#)

СписокИменКлючейРеестра

Возвращает список имен ключей указанного раздела реестра операционной системы Windows.

Синтаксис

СписокИменКлючейРеестра (ИмяРаздела)



Параметры

ИмяРаздела (текст) – имя раздела с списком ключей. Необходимо указывать полный путь к разделу реестра.

Возвращает

Массив – список имен ключей раздела реестра операционной системы.

Пример

Необходимо узнать имена ключей указанного раздела реестра операционной системы. Можно воспользоваться следующим примером:

```
пВетка = "HKEY_CURRENT_USER\\Control  
Panel\\Patterns";  
мСписок = СписокИменКлючейРеестра (пВетка);  
перем пИмя=0;  
Пока (пИмя++<Размер (мСписок))  
    Сообщить (мСписок [пИмя]);
```

См. также:

- [ЗначениеКлючаРеестра](#)
- [РеестрСписокЗначенийКлючей](#)

РеестрСписокЗначенийКлючей

Возвращает объект с ключами указанного раздела реестра операционной системы Windows.

Синтаксис

```
РеестрСписокЗначенийКлючей (ИмяРаздела)
```

Параметры

ИмяРаздела (текст) – имя раздела с списком ключей. Необходимо указывать полный путь к разделу реестра.

Возвращает

Объект, содержащий ключи указанного раздела реестра. Каждый ключ имеет соответствующее значение.



Пример

Необходимо узнать значения ключей в указанном разделе реестра операционной системы:

```
пВетка = "HKEY_CURRENT_USER\\Control  
Panel\\Patterns";  
оРеестр = РеестрСписокЗначенийКлючей (пВетка) ;  
ДляВсех (Переменных (оРеестр , пИмя) )  
Сообщить (пИмя+ " - "+оРеестр [пИмя] ) ;
```

См. также:

- [ЗначениеКлючаРеестра](#)
- [СписокИменКлючейРеестра](#)

Оборудование

ИмяКомпьютера

Возвращает системное имя компьютера.

Синтаксис

```
ИмяКомпьютера ( ) ;
```

Параметры

Не указываются.

Возвращает

Текст – системное имя компьютера.

НайтиОборудование

Возвращает запись устройства, подключенное к рабочему месту и активированное (включен флаг «Устройство активно»).

Синтаксис

```
НайтиОборудование (ТипУстройства [ , ВидУстройства ] )
```

Параметры



ТипУстройства (текст) - тип устройства (ККМ, Весы, и т.п.) Название типа должно точно совпадать с названием раздела 1-го уровня таблицы «Список устройств».

ВидУстройства (текст) - конкретный вид (марка) устройства данного типа (для ККМ - ШтрихФР, Феликс и т.п.). Название вида должно точно совпадать с названием устройства в таблице «Список устройств», но может и не совпадать с названием устройства, подключенного к рабочему месту. Если вид устройства не указан и есть несколько устройств одного типа, то вернет первое встретившееся.

Возвращает

Объект – запись таблицы «Список устройств».

Примеры

Допустим у вас на рабочем месте в разделе «Сканеры» находятся сканеры. Хотим найти первый активный (включен флаг «Устройство активно») сканер для текущего рабочего места:

```
пОборудование = НайтиОборудование ("Сканеры") ;
```

```
Если (пОборудование)
```

```
    Сообщить (пОборудование.Название) ;
```

```
Иначе
```

```
    Сообщить ("Оборудование не найдено.") ;
```

```
Если результат вас не устраивает и надо найти именно «Сканер документов», нужно написать так:
```

```
пВидОборудования = "Сканер документов" ;
```

```
пОборудование = НайтиОборудование ("Сканеры" ,
```

```
пВидОборудования) ;
```

```
Если (пОборудование)
```

```
    Сообщить (пОборудование.Название) ;
```

```
Иначе
```

```
    Сообщить ("Оборудование не найдено.") ;
```

См. также:

- [ДляВсех\(Устройств\)](#)
- [СписокПринтеров](#)

СообщитьРабочееМесто



Возвращает запись таблицы «Рабочие места».

Синтаксис

`СообщитьРабочееМесто ([Идентификатор])`

Параметры

Идентификатор (текст) - идентификатор рабочего места. Если параметр не указан, возвращает запись текущего рабочего места.

Возвращает

Объект – запись таблицы «Рабочие места».

Примеры

Сообщает название и идентификатор текущего рабочего места:

```
пТекущееРабочееМесто = СообщитьРабочееМесто ( ) ;  
Сообщить ("Название рабочего места :  
"+пТекущееРабочееМесто.Название , "Идентификатор  
рабочего места : " +  
пТекущееРабочееМесто.Идентификатор) ;
```

Сообщает название по заданному идентификатору рабочего места:

```
пИдентификатор = ИмяКомпьютера ( ) ; # инициализируем  
переменную, взяв значение имени компьютера  
Спросить ("Укажите идентификатор для поиска  
компьютера" , пИдентификатор) ;  
пРабочееМесто =  
СообщитьРабочееМесто (пИдентификатор) ;  
Если (пРабочееМесто)  
Сообщить ("Название рабочего места :  
"+пРабочееМесто.Название) ;  
Иначе  
Сообщить ("Рабочее место с идентификатором  
\ "+пИдентификатор+ " \" не найдено") ;
```

Список Принтеров

Возвращает список принтеров, установленных в операционной системе Windows.

Синтаксис

`СписокПринтеров ()`



Параметры

Не указываются.

Возвращает

Массив, содержащий строки с названиями принтеров, установленных в системе.

Пример

Необходимо узнать названия принтеров, установленных в операционной системе Windows:

```
мСписок = СписокПринтеров ( ) ;  
перем пНом = 0 ;  
Пока (пНом++<Размер (мСписок) )  
    Сообщить (мСписок [пНом] ) ;
```

См. также:

- [НайтиОборудование](#)
- [ДляВсех\(Устройств\)](#)

ДляВсех(Устройств)

Перебирает устройства указанного типа подключенные к указанному рабочему месту.

Синтаксис

```
ДляВсех (Устройств ( [РабМесто] [ , ТипУстройства ] )
```

Параметры

РабМесто (объект) – запись таблицы «Рабочие места». Если параметр не определен, перебираются устройства текущего рабочего места.

ТипУстройства (текст) – тип устройства, как он указан в списке устройств (раздел верхнего уровня). Если параметр не указан, то функция перебирает все устройства подключенные к рабочему месту.

Возвращает

Ничего не возвращает.

Комментарии

Функция устанавливает контекстный объект с именем "Устройство", содержащий запись выборки "Оборудование рабочего места". Обращаться к полям этого объекта можно без его уточнения.

Пример

Нужно узнать название всех неактивных устройств типа "Клавиатуры" на текущем рабочем месте:

```
ДляВсех (Устройств ("Клавиатуры"))
{
    Если (!Флаг (Свойства, "Устройство активно"))
        Сообщить (Название);
}
```

Необходимо вызывать окно выбора рабочих мест и для выбранного места сообщить названия всех устройств идентификации:

```
оОкноВыбора = Окно ("Список рабочих мест");
Если (оОкноВыбора.ВыполнитьВыбор ())
{
    пЗапись = оОкноВыбора.Запись ();
    ДляВсех (Устройств (пЗапись, "Устройства
идентификации"))
        Сообщить (Название);
}
```

См. также:

- [НайтиОборудование](#)
- [СписокПринтеров](#)

Прочие функции

Запустить

Запускает выполнение внешней программы.

Синтаксис

```
Запустить (Имя [, Действия [, Каталог]])
```



Параметры

Имя (текст) – имя файла с программой. Если в имени не указан диск и каталог, в котором этот файл находится, то просматриваются последовательно каталог, откуда был запущен «sbis.exe», и каталоги, перечисленные в системной переменной «PATH». При указании имени каталога учтите, что знак «\» является служебным, поэтому, чтобы поставить в строке собственно символ «\», его нужно указывать *дважды* (смотрите пример). Если в имени файла присутствуют пробелы, то в вызове необходимо добавлять двойные кавычки. Символ «"» тоже является служебным по этому перед ним надо ставить «\».

В операционных системах MS Windows допускается запускать не только исполняемые файлы, но и документы, просто указав в качестве параметра имя открываемого документа.

Для запуска файлов может потребоваться программа «start.exe», входящая в MS Windows. В случае возникновения проблем с функцией «Запустить», проверьте, выполняется ли команда «start» из командной строки.

Действия (текст) – строка, в которой можно указать через запятую любые из четырех параметров: «**ждать**» - ждать окончания программы; «**мин**» - запустить в минимизированном виде; «**макс**» - запускать в максимизированном виде; «**спрятать**» - запускать скрыто.

Каталог (текст) – рабочий каталог программы.

Возвращает

Число – код возврата выполняемой программы.

Примеры

Вызвать калькулятор Windows:

```
Запустить ("calc") ;
```

Вызвать программу «Pcc.exe» (Pervasive.SQL Control Center) из каталога «C:\PVSW\Bin»:

```
Запустить ("C:\\PVSW\\Bin\\Pcc.exe") ;
```

Обратите внимание, что поскольку обратная косая черта рассматривается программой как специальный символ, её необходимо продублировать.

Учтите, что Windows не будет дожидаться окончания выполнения запущенной программы. Поэтому, чтобы запущенное действие обязательно было выполнено до выхода из функции «Запустить», нужно написать так:

```
Запустить ("C:\PVSW\BIN\Pcc.exe" , "ждать") ;
```

При выполнении bat-файла под Windows удобнее всего сделать на этот bat-файл ярлык и запускать именно ярлык:

```
Запустить ("run_my_bat.lnk") ;
```

Следует учесть, что из-за особенностей Windows режим ожидания окончания выполнения программы в этом случае работать не будет.

Чтобы просто запустить bat-файл придётся воспользоваться конструкцией «cmd.exe /c ИмяФайла» (для Windows 2000) или «command.com /c ИмяФайла» (для Windows 95/98) или

```
Запустить ("Имяфайла") .
```

В последнем случае окно запуска не будет закрываться по завершению работы. Его можно закрыть принудительно, прописав в самом bat-файле команду «exit».

Ещё несколько применений функции «Запустить».

Открыть документ, созданный в MS Word:

```
Запустить ("\"Список функций.doc\"") ;
```

Или с указанием пути к документу:

```
Запустить ("C:\Мои документы\Список функций.doc") ;
```

Открыть интернет-страницу:

```
Запустить ("http://sbis.ru") ;
```

Отправить e-mail письмо:

```
Запустить ("mailto:sbis@tensor.ru") ;
```

Открыть определённый каталог:

```
Запустить ("C:\SBIS") ;
```

Заснуть

Заставляет программу некоторое время не реагировать на любые действия пользователя.

Синтаксис

```
Заснуть (Время)
```



Параметры

Время (целое) – время бездействия программы. Указывается в миллисекундах.

Возвращает

Ничего не возвращает.

Пример

Чтобы заставить программу подождать 1 секунду от выполнения дальнейших действий пользователя, нужно всего лишь написать:

Заснуть (1000)