APACS 3000

Подсистема автоматизации и SDK

Руководство пользователя

Содержание

1 Введение

Подсистема автоматизации ПК APACS 3000 позволяет администратору настроить нестандартное поведение комплекса и автоматизировать реакции системы на часто возникающие ситуации. Подсистема позволяет настраивать как последовательность действий, которые исполняются в комплексе без участия дежурного оператора (в фоновом режиме), так и действия, которые исполняются по решению оператора.

При помощи подсистемы автоматизации могут быть реализованы следующие типичные задачи:

- взаимодействие между различными аппаратными системами (например, при предъявлении карты на считывателе Apollo поставить на охрану сигнализацию VertX),
- уведомление администратора о поступлении определенных сообщений (например, рассылка сообщений по электронной почте),
- создание собственного пользовательского интерфейса,
- передача поступающих сообщений из ПК APACS 3000 во внешние системы учета рабочего времени,
- исполнение в системе каких–либо функций по расписанию.

Также подсистема автоматизации позволяет инициировать запись видео на поступление определенных сообщений и связывать записанный видеофрагмент и его сообщение–инициатор (подробно см. раздел «Просмотр видео в ПК APACS 3000»).

Таким образом, с помощью подсистемы автоматизации администратор может легко настроить комплекс под требования конкретной системы управления и контроля доступом.

Расположение подсистемы автоматизации

Подсистема автоматизации может быть настроена и запущена в разных частях комплекса:

- на сервере APACS 3000 (реализовано в составе службы «Автоматизация»),
- в рамках разных клиентских приложений APACS 3000, как правило, это приложения «Консоль» и «Дежурный режим» (реализовано в виде модуля ***Клиентская автоматизация***),
- на разных рабочих станциях.

На сервер APACS 3000 должны быть вынесены безусловные задачи автоматизации, которые должны исполняться в фоновом режиме и не требуют участия оператора.

Ориентированные на пользователя задачи автоматизации требуется настраивать в рамках конкретного приложения.

Макрокоманда

Макрокоманда позволяет задавать действие или последовательность действий с объектами или группой объектов, в системе макрокоманды выполняется по решению оператора.

Заданные администратором макрокоманды могут отображаться в меню «Макрокоманды» окна ***Основная панель*** в приложении «Дежурный режим». В зависимости от ситуации оператор выбирает нужную ему макрокоманду, и команда выполняется в системе.

Например, в меню «Макрокоманды» могут находится следующие пункты:

- Включить тревогу на складе,
- Поставить офис на охрану,
- Снять офис с охраны,
- Разблокировать все двери в случае пожара.

Макрокоманда может использоваться в качестве составной части при конфигурировании *программ*.

Программа

Программа позволяет задать последовательность действий системы на поступающие сообщения. Настроенные один раз программы исполняются в системе автоматически и не требуют участия дежурного оператора.

Обратите внимание: далее в документации приводятся примеры скриптов. Эти и другие примеры скриптов находятся в папке Samples корневого каталога APACS 3000.

Способы конфигурирования подсистемы автоматизации

Подсистема автоматизации может быть сконфигурирована следующим образом:

- с помощью объектов автоматизации,
- с помощью сценариев, написанных на языке VBScript.

Интерфейс APACS3000 позволяет быстро и наглядно настроить программы и макрокоманды, но имеет ограниченный набор действий.

При написании сценариев на языке VBScript администратор может использовать все возможности VBScript: массивы, операторы управления, перечисления свойств объектов, функции работы со строками, обработка ошибок.

VBScript предоставляет доступ к COM–объектам из скрипта, что позволяет использовать возможности других приложений. Например, используя возможности приложения Microsoft Office, можно составлять интерактивные отчеты о сообщениях и объектах системы в MS Word и таблицах MS Excel.

Администратор комплекса может выбрать подходящий ему способ настройки подсистемы автоматизации в зависимости от поставленной задачи.

2 Конфигурирование подсистемы автоматизации

Подсистема автоматизации конфигурируется в окне ***Проводник*** с помощью объектов автоматизации (см. «Apc: Глава 3 Консоль 3.2 Клиентский модуль Проводник»).

Далее мы рассмотрим последовательность конфигурирования автоматизации на сервере и клиенте APACS 3000. Настройки объектов автоматизации смотрите далее в п. «2.5 Объекты автоматизации».

2.1 Создание макрокоманды

Макрокоманда может быть создана следующими способами:

- с помощью интерфейса APACS 3000 – создайте в окне ***Проводник*** объект типа *Простая макрокоманда*.
- с помощью языка VBScript:
  - напишите макрокоманду на языке VBScript,
  - создайте объект типа *Макрокоманда VBScript* и загрузите в него написанную на VBScript макрокоманду.

2.2 Создание программы

Программа может быть создана следующими способами:

- с помощью интерфейса APACS 3000 – создайте в окне ***Проводник*** объект типа *Простая программа*.
- с помощью языка VBScript:
  - напишите программу на языке VBScript,
  - создайте объект типа *Программа VBScript* и загрузите в него написанную на VBScript программу.

Рисунок Способы конфигурирования программы и макрокоманды

2.3 Конфигурирование списка макрокоманд для приложения

Для того чтобы в приложении дежурного оператора в меню «Макрокоманды» окна ***Основная панель*** присутствовал список доступных макрокоманд, выполните следующее:

- Создайте необходимое количество объектов типа *Простая макрокоманда* и/или *Макрокоманда VBScript*.
- В рамках этого приложения выберите пункт меню «Клиентская автоматизация / Список макрокоманд» окна ***Основная панель***. Откроется диалоговое окно ***Список макрокоманд***. С помощью кнопок **Добавить** и **Удалить** сформируйте список макрокоманд для данного приложения.

  Последовательность макрокоманд в меню «Макрокоманды» определяется последовательностью в окне ***Список макрокоманд***. Чтобы поменять порядок следования макрокоманд, выделите макрокоманду и используйте кнопки **Переместить вверх** и **Переместить вниз**.

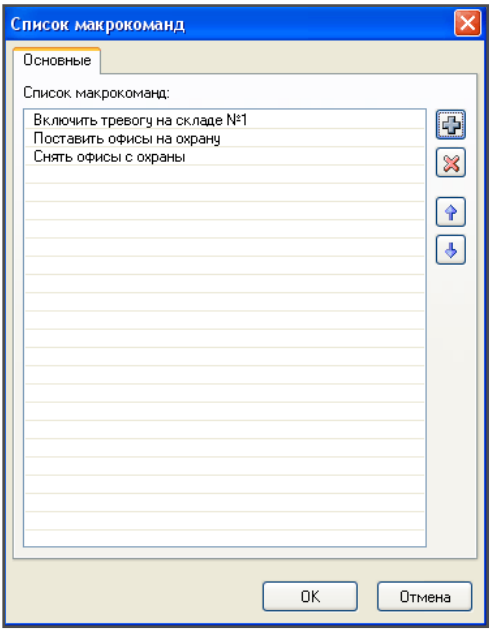


Рисунок Окно ***Список макрокоманд***

- Чтобы подсистема автоматизации работала в рамках этого приложения, в схему приложения включите модуль ***Клиентская автоматизация*** (объект типа *Схема клиентского приложения* вкладка **«Основные»**).

2.4 Конфигурирование автоматизации

Чтобы сконфигурировать автоматизацию, выполните следующее:

- Создайте необходимое количество объектов типа *Простая программа* и/или *Программа VBScript*.
- Спланируйте выполнение программ в системе с помощью объекта типа *Настройки автоматизации*.
- Если автоматизация должна выполняться в рамках приложения, выполните следующее:
  - Укажите объект, содержащий описание автоматизации для данного приложения. Для этого пунктом меню «Клиентская автоматизация / Настройки» окна ***Основная панель*** откройте диалоговое окно ***Выбор настроек клиентской автоматизации*** и укажите объект.
  - В схему этого приложения включите модуль ***Клиентская автоматизация*** (объект типа *Схема клиентского приложения* вкладка **«Основные»**).
  - Запустите выполнение автоматизации пунктом меню «Клиентская автоматизация / Перезапустить» окна ***Основная панель***. Остановить выполнение автоматизации можно пунктом меню «Клиентская автоматизация / Остановить».

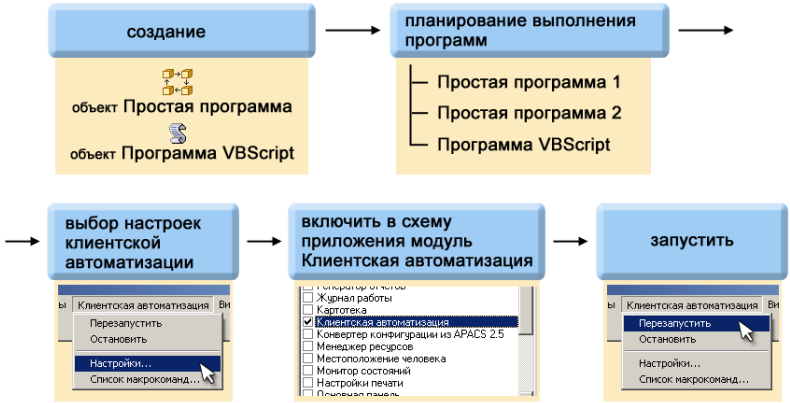


Рисунок Схема конфигурирования клиентской автоматизации

- Если автоматизация должна выполняться на сервере APACS 3000, выполните следующее:
  - В настройках системного объекта *Настройки служб* укажите объект типа *Настройки автоматизации*, содержащий описание автоматизации, которая должна выполняться на сервере APACS 3000 (см. «Арс: Глава 3 Консоль 3.4.4 Объект Настройки служб»).
  - Чтобы настройки серверной автоматизации вступили в силу, на объекте *Настройки служб* выполните команду ***Перезапустить***

автоматизацию.

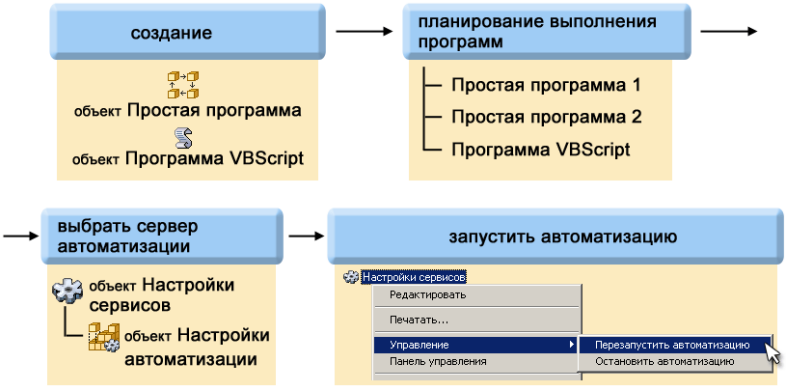


Рисунок Схема конфигурирования серверной автоматизации

2.5 Объекты автоматизации

Далее рассмотрим конфигурирование объектов автоматизации.

Обратите внимание: чтобы измененные настройки объектов автоматизации вступили в силу, требуется:

- для серверной автоматизации — выполнить команду ***Перезапустить автоматизацию*** объекта *Настройки служб*,
- для клиентской автоматизации — выбрать пункт меню «Клиентская автоматизация / Перезапустить» окна ***Основная панель***.

2.5.1 *Простая макрокоманда*

Файл типа *Простая макрокоманда* позволяет задать список объектов и команд, которые должны быть последовательно выполнены на этих объектах.

Файл создается путем добавления к объектам типа *Папка*.

На вкладке **«Основные»** можно сформировать список объектов и команд, которые должны быть выполнены на этих объектах.

Список объектов и команд формируется с помощью кнопок **Добавить**, **Редактировать** и **Удалить**.

При выполнении макрокоманды, на объектах выполняются команды в той последовательности, в которой они заданы на вкладке **«Основные»**. Чтобы изменить последовательность, выделите строку и используйте кнопки **Переместить вверх** и **Переместить вниз**.

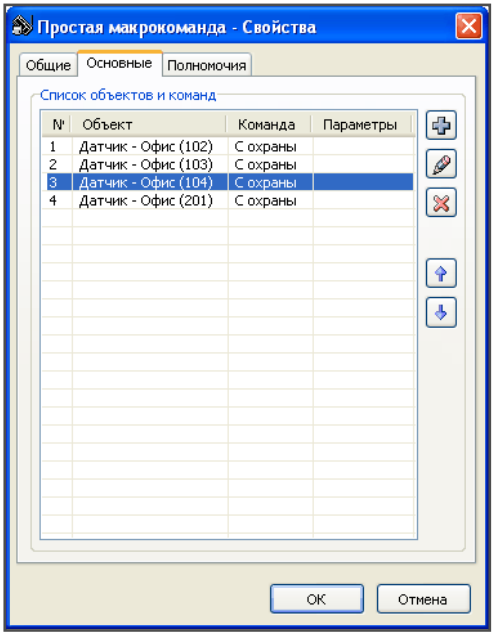


Рисунок Вкладка **«Основные»** окна редактирования свойств файла *Простая макрокоманда*

Добавление объекта и команды

Чтобы добавить в список объект и команду, которая должна быть выполнена на этом объекте, нажмите кнопку **Добавить**.

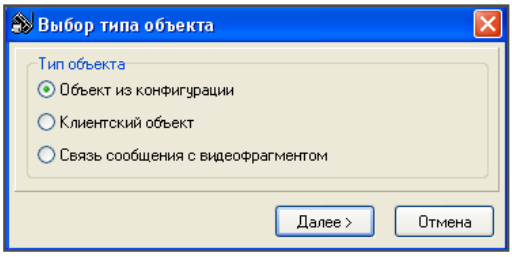


Рисунок Окно ***Выбор типа объекта***

Откроется диалоговое окно ***Выбор типа объекта***, где можно выбрать объекты следующих типов:

- *Объект из конфигурации* — позволяет выбрать объект на сервере APACS 3000 и указать для этого объекта команду управления,
- *Клиентский объект* — позволяет выбрать клиентские объекты Client и Script и использовать их методы (подробно см. далее п. «3 Объектная

модель, доступная из макрокоманд и программ VBScript».

- *Связь сообщения с видеофрагментом* — объект позволяет инициировать запись видео на указанной видеокамере при поступлении определенных сообщений. В момент записи видео создается сообщение типа *Видеофрагмент связан с сообщением*, связанное со своим сообщением–инициатором (подробно см. п. «3.2.6 Интерфейс IApcVideoLinkerWrap»).

Объекты	Клиентская автоматизация	Серверная автоматизация
Объект из конфигурации	✓	✓
Client	✓	✗
Script	✓	✓
Связь сообщения с видеофрагментом	✓	✓

Таблица показывает, где могут быть использованы объекты простой макрокоманды

Обратите внимание: для работы макрокоманды с объектом *Связь сообщения с видеофрагментом* требуется фильтрация поступающих сообщений, поэтому такую макрокоманду имеет смысл использовать в составе простой программы или программы VBScript.

Выберите тип объекта и нажмите кнопку **Далее**.
Дальнейшие действия зависят от типа выбранного объекта.

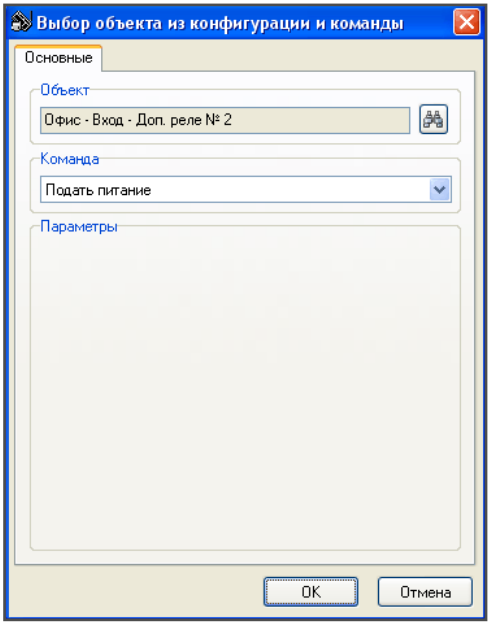


Рисунок Окно ***Выбор объекта из конфигурации и команды***

- Если выбран тип *Объект из конфигурации*, откроется диалоговое окно ***Выбор объекта из конфигурации и команды***, где требуется указать:
  - **Объект**,
  - **Команда** — команда, которая должна быть выполнена на этом объекте,
  - **Параметры** — параметры, с которыми должна быть выполнена команда.

  Команды управления, которые могут быть применены к объектам, можно посмотреть в документации на соответствующий драйвер оборудования.

- Если выбран тип *Клиентский объект*, откроется диалоговое окно ***Выбор клиентского объекта и команды***, где требуется указать:
  - **Объект**,
  - **Команда** — команда, которая должна быть выполнена на этом объекте,
  - **Параметры** — параметры, с которыми должна быть выполнена команда.

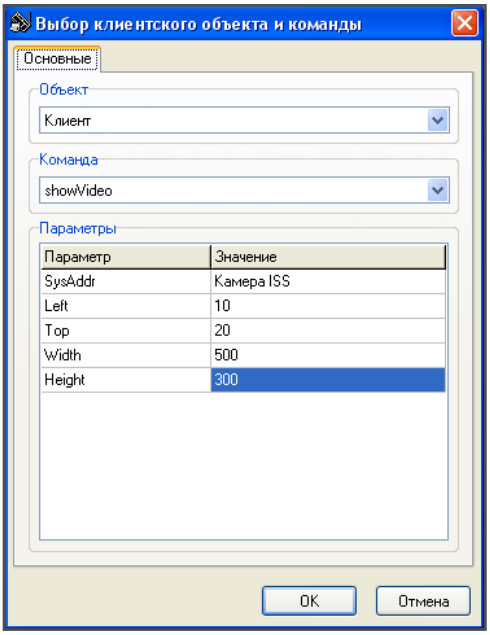


Рисунок Окно ***Выбор глобального объекта и команды***

- Если выбран тип *Связь видеофрагмента с сообщением*, откроется диалоговое окно ***Связь сообщения с видео***, где требуется указать:
  - **Камера** — камера, на которой должна быть инициирована запись видео,
  - **Длительность записи, с** — продолжительность записи видео.

Обратите внимание: простая макрокоманда на связь сообщения с видео срабатывает только как реакция на определённое событие. Для записи видео без инициирующих событий используйте команды ***Начать запись видео*** и ***Остановить запись видео*** для выбранной камеры.

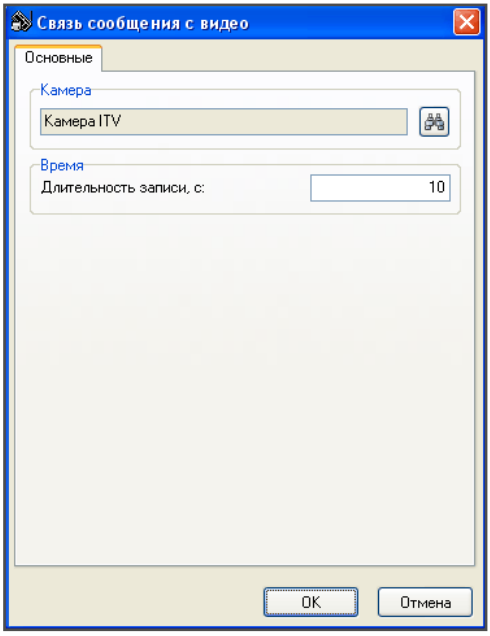


Рисунок Окно ***Связь сообщения с видео***

Команды управления

Объект *Простая макрокоманда* поддерживает команду ***Выполнить***, с помощью которой макрокоманда может быть выполнена в рамках клиентского приложения.

2.5.2 *Фильтр событий*

Файл типа *Фильтр событий* позволяет создавать условия фильтрации поступающих сообщений, которые будут использоваться в работе простой программы.

Файл создается путем добавления к объектам типа *Папка*.

Все настройки объекта находятся на вкладках: **«Основные»**,**«Инициатор сообщения»** и **«Общие поля сообщений»**.

На вкладке **«Основные»** окна ***Фильтр событий-Свойства*** можно задать следующие условия фильтрации сообщений:

- **Время возникновения** — в этой группе параметров можно задать условие фильтрации по времени возникновения сообщения. В полях **с** и **до** можно указать начало и конец интервала. Также интервал поиска можно ограничить сроком ранее / позднее какого–либо времени. При помощи кнопки **Установить текущее время** можно указать в поле текущее время.

- **Время регистрации сообщения на сервере** — в этой группе параметров можно задать условие фильтрации по времени регистрации сообщения на сервере. В полях **с** и **до** можно указать начало и конец интервала. Также интервал поиска можно ограничить сроком ранее / позднее какого–либо времени. При помощи кнопки **Установить текущее время** можно указать в поле текущее время.
- **Типы событий** — в этом поле укажите нужные типы сообщений. Для отображения типов сообщений используются следующие режимы:
  - o группировка типов сообщений на логические группы,
  - o группировка в зависимости от наличия в сообщении информации о владельце карты,
  - o группировка по типам объектов, инициирующих сообщения,
  - o список типов сообщений без группировки.

  Режимы группировки позволяют быстро выбрать необходимые типы сообщений. Задать режим группировки можно при помощи кнопок фильтра. Изменение режима не ведет к изменению типов сообщений, выбранных в предыдущем режиме.

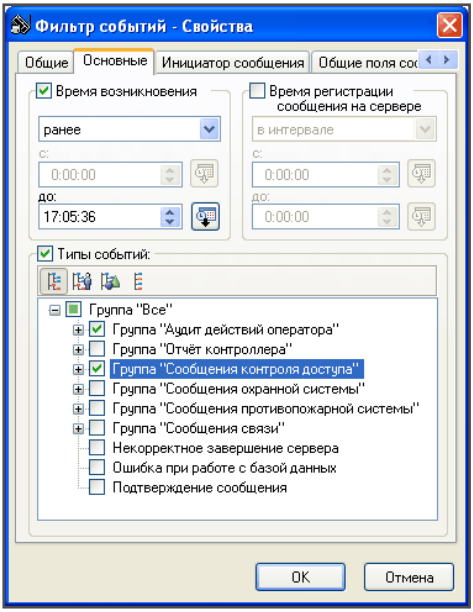


Рисунок Вкладка **«Основные»** окна редактирования свойств файла *Фильтр событий*

На вкладке **«Инициатор сообщения»** окна ***Фильтр событий-Свойства*** можно выбирать сообщения в зависимости от их объекта–инициатора.

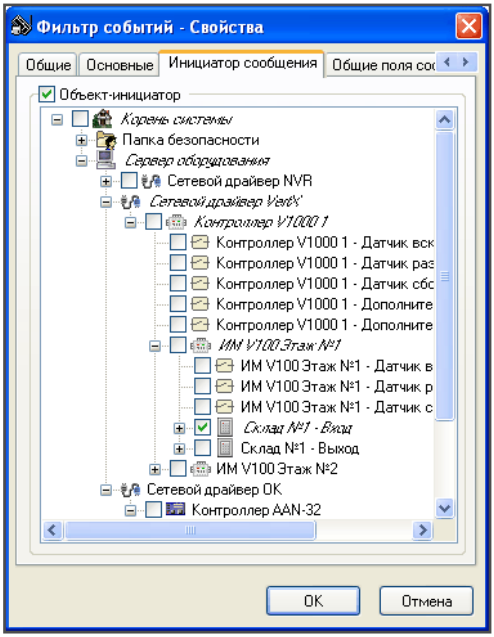


Рисунок Вкладка «**Инициатор сообщения**» окна редактирования свойств файла *Фильтр событий*

В поле **Объект–инициатор** находится иерархический список объектов, от которых могут поступать сообщения. Для выбора объектов удобно использовать контекстное меню, чтобы:

- выделить подэлементы выбранного объекта,
- отменить выделение подэлементов выбранного объекта,
- выделить все объекты, от которых могут поступать сообщения,
- отменить выделение всех выделенных объектов.

На вкладке **«Общие поля сообщений»** окна ***Фильтр событий-Свойства*** можно задать условия фильтрации по полям сообщений.

В поле **Доступные поля** находятся поля, общие для выбранных ранее типов сообщений. Выберите поле сообщений и перенесите его в поле **Активные поля для фильтрации** кнопкой **Добавить в список активных**.

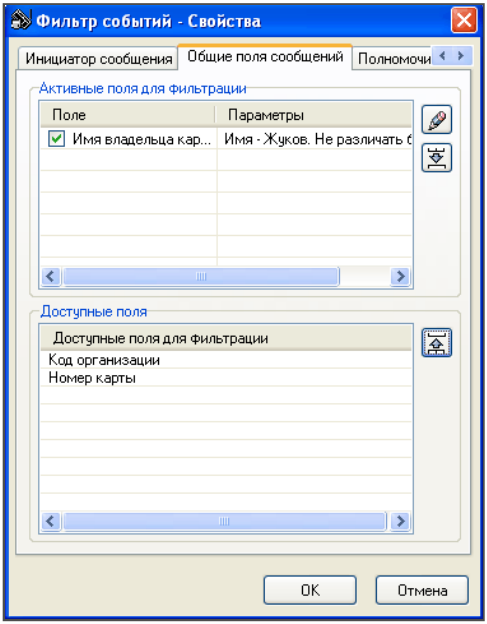


Рисунок Вкладка **«Общие поля сообщений»** окна редактирования свойств файла *Фильтр событий*

Дальнейшие действия зависят от выбранного поля:

- Если выбрано поле **Номер карты**, откроется диалоговое окно ***Фильтр по номеру карты***, где требуется указать условия выбора сообщений по номеру карты. Выберите способ фильтрации и задайте интервал поиска в полях **С** и **По**.

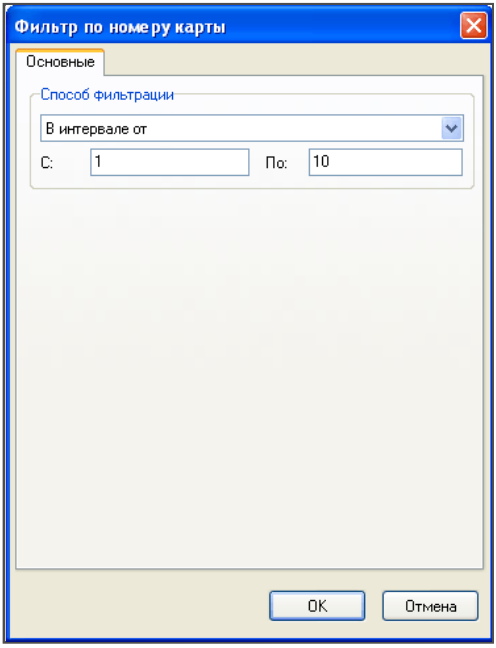


Рисунок Окно ***Фильтр по номеру карты***

- Если выбрано поле **Имя владельца карты**, откроется диалоговое окно ***Фильтр по имени владельца карты***, где требуется указать условия выбора сообщений по имени владельца карты.
  Поля **Учитывать регистр** и **Искать полное совпадение** задают дополнительные параметры поиска введенного слова.
  - **Учитывать регистр** — если стоит этот флажок, при поиске слов учитывается разница между прописными и строчными буквами.
  - **Искать полное совпадение** — если стоит этот флажок, будут выбраны слова, полностью совпадающие по написанию с введенным Вами в фильтр словом. Если этот флажок не стоит, будут выбраны слова, включающие в себя введенное слово.

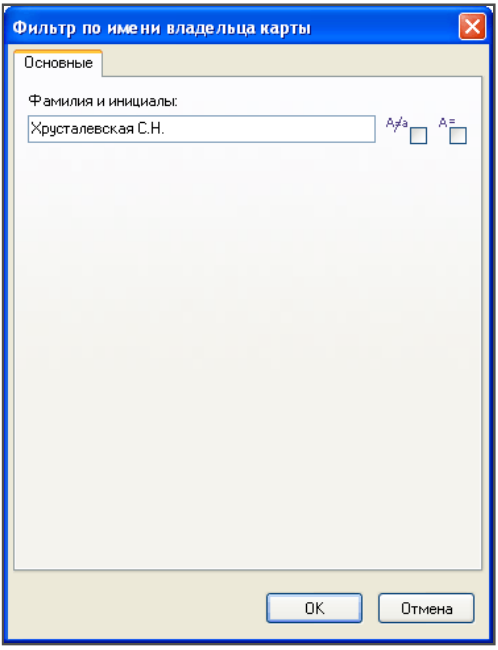


Рисунок Окно ***Фильтр по имени владельца карты***

- Если выбрано поле **Код организации**, откроется диалоговое окно ***Фильтр по коду организации***, где требуется указать условия выбора сообщений по коду организации.

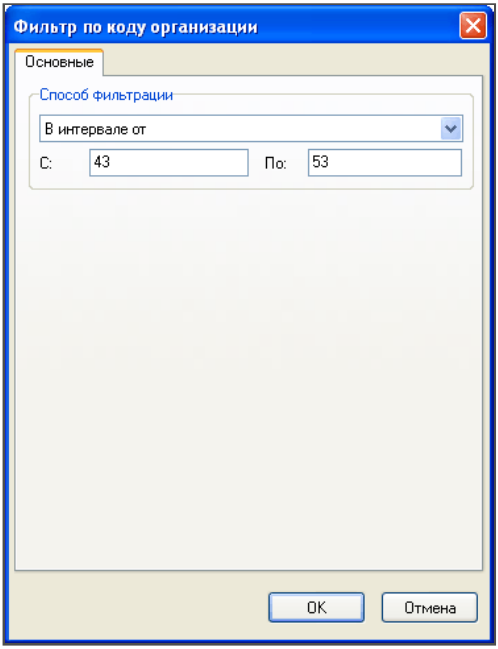


Рисунок Окно ***Фильтр по коду организации***

После того как для поля было указано условие фильтрации, поле появится в поле **Активные поля для фильтрации**.

Участвующие в фильтрации поля сообщений отмечаются флажками в поле **Активные поля для фильтрации**. Чтобы отменить фильтрацию по этому полю, снимите флажок.

Чтобы изменить условие фильтрации по этому полю, нажмите кнопку **Изменить**.

Чтобы удалить условие фильтрации по этому полю, нажмите кнопку **Удалить из списка активных**.

2.5.3 *Простая программа*

Файл типа *Простая программа* позволяет с помощью фильтров событий и макрокоманд задать реакции системы на поступающие сообщения.

Файл создается путем добавления к объектам типа *Папка*.

На вкладке **«Основные»** требуется сформировать список соответствий между фильтрами сообщений и макрокомандами, которые должны быть выполнены при поступлении выбранных в фильтре сообщений.

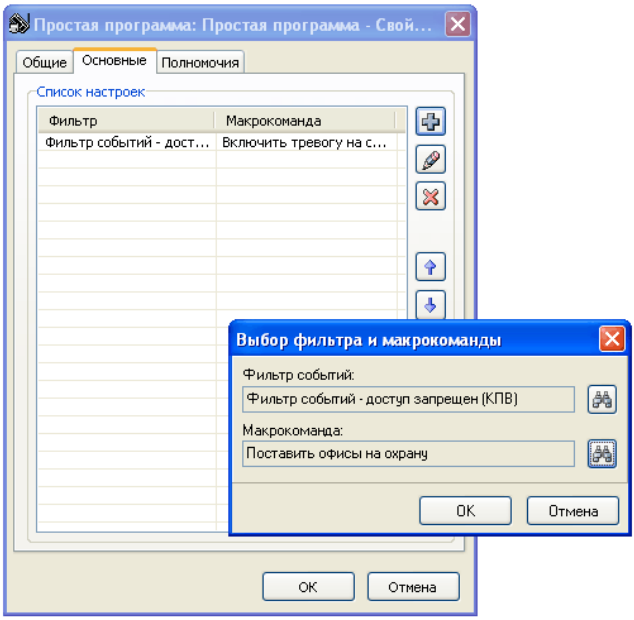


Рисунок Вкладка **«Основные»** окна редактирования настроек файла *Простая программа*

Список формируется с помощью кнопок **Добавить**, **Редактировать** и **Удалить**.

При работе программы в системе проверка поступающих сообщений и выполнение заданных действий будет происходит в той последовательности, которая задана на вкладке **«Основные»**. Чтобы изменить последовательность, выделите строку и используйте кнопки **Переместить вверх** и **Переместить вниз**.

Добавление фильтра сообщений и макрокоманды

Чтобы добавить в список фильтр сообщений и макрокоманду, которая должна выполняться при поступлении этих сообщений, нажмите кнопку **Добавить**. Откроется диалоговое окно ***Выбор фильтра и макрокоманды***, где укажите:

- фильтр событий, в котором указаны условия фильтрации сообщений,
- макрокоманду, которая должна быть вызвана при выполнении условий фильтра.

2.5.4 Макрокоманда VBScript

Файл типа *Макрокоманда VBScript* содержит написанный на языке VBScript сценарий управления объектами, который можно запустить на выполнение вручную.

Файл создается путем добавления к объектам типа *Папка*.

Чтобы сконфигурировать объект, выполните на объекте команду ***Редактировать***. Откроется диалоговое окно ***Редактор макрокоманд VBScript***.

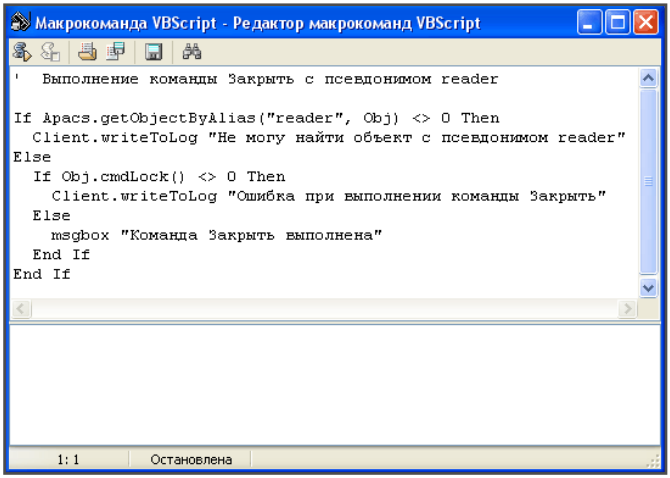


Рисунок Окно ***Редактор макрокоманд VBScript***

Окно ***Редактор макрокоманд VBScript*** поделено на две части:

- верхняя часть предназначена для текста макрокоманды, написанной на VBScript,
- нижняя предназначена для отображения информации об исполнении макрокоманды.

Внизу окна ***Редактор макрокоманд VBScript*** располагается строка состояния со следующей информацией:

- позиция курсора,
- сведения о состоянии макрокоманды (например, *Остановлена*).

Для работы с окном используется панель инструментов.

Создание макрокоманды

Текст макрокоманды можно написать непосредственно в верхней части окна, либо поместить туда макрокоманду, написанную и сохраненную ранее в файлах формата *.txt,*.vbs.

Для загрузки макрокоманды воспользуйтесь кнопкой **Загрузить из внешнего файла**.

Чтобы сохранить макрокоманду, нажмите кнопку **Сохранить скрипт**.

С помощью кнопки **Сохранить во внешний файл** можно сохранить текст макрокоманды в файле формата *.vbs.

Выполнение макрокоманды

Чтобы запустить макрокоманду, нажмите кнопку **Выполнить**. Выполнение макрокоманды можно прекратить кнопкой **Прервать**.

Команды управления

Объект *Макрокоманда VBScript* поддерживает команду ***Выполнить***, с помощью которой макрокоманда может быть выполнена в рамках клиентского приложения.

2.5.5 *Программа VBScript*

Файл типа *Программа VBScript* содержит написанный на языке VBScript сценарий, где описана последовательность действий системы на поступающие сообщения.

Чтобы сконфигурировать объект, выполните на объекте команду ***Редактировать***. Откроется диалоговое окно ***Редактор программ VBScript***. Работа с этим окном аналогична работе с окном ***Редактор макрокоманд VBScript*** (см. п. «2.5.4 Макрокоманда VBScript»).

2.5.6 *Настройки автоматизации*

Файл типа *Настройки автоматизации* позволяет спланировать выполнение программ в системе APACS 3000.

Чтобы сконфигурировать объект, выполните на объекте команду ***Редактировать***. На вкладке **«Основные»** этого объекта укажите следующие настройки:

- **Планирование программ** — в этом поле с помощью кнопок **Добавить поток**, **Добавить программу** и **Удалить** сформируйте распределение программ по потокам выполнения.

  В рамках одного потока программы обрабатывают сообщения последовательно, в том порядке, в каком они указаны в списке. Только после того как сообщение было обработано последней программой, начинается обработка следующего поступившего сообщения в первой программе.

  Программы из разных потоков исполняются параллельно, независимо друг от друга.

  В рамках одного объекта *Настройки автоматизации* максимально может использоваться пять потоков. В рамках одного потока максимально может использоваться десять программ.
- **Длина очереди сообщений** — укажите размер буфера сообщений (по умолчанию *1000*). Возникающие в системе сообщения передаются в

спланированные программы. Если программы обрабатывают сообщения медленно, то сообщения накапливаются в буфере. Если буфер переполнен, новые поступающие сообщения игнорируются и не обрабатываются. Поэтому при планировании работы программ избегайте длительных действий.

Для каждого потока в программе используется отдельный буфер сообщений. Программу, которая работает медленно, можно выделить в отдельный поток.

- **Время ожидания завершения, с** — при остановке клиентской автоматизации в течение этого времени система ожидает корректного завершения программы (по умолчанию *5 с*). В противном случае программа будет останавлена принудительно, и потребуется перезагрузить приложение.

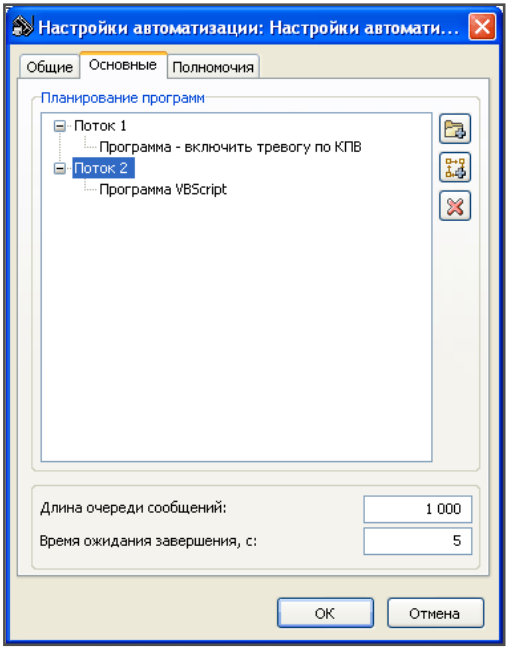


Рисунок Вкладка **«Основные»** окна редактирования свойств файла *Настройки автоматизации*

3 Объектная модель, доступная из макрокоманд и программ VBScript

Объектная модель описывает объекты, методы и их параметры, которые могут быть использованы при написании макрокоманд и программ на языке VBScript (далее также скрипты).

Методы программ и макрокоманд VBScript носят универсальный характер по отношению к командам, объектам и сообщениям, которые на данный момент поддерживает комплекс.

Возможности макрокоманд и программ VBScript

Макрокоманды и программы VBScript предоставляют следующие возможности:

- получение текущих настроек объектов,
- редактирование настроек объектов,
- создание новых объектов,
- регистрация сообщений от объектов,
- удаление объектов,
- управление объектами (выполнение команд),
- получение текущих сообщений,
- получение уведомлений об удалении, изменении, добавлении объектов,
- получение сообщений из базы данных по фильтру.

Получение доступа к серверным объектам APACS 3000 из скрипта

Для получения из скрипта доступа к объектам, расположенным на сервере APACS 3000, можно воспользоваться следующими способами:

- использовать метод getObjectByAlias(). Для использования этого метода в свойствах объекта обязательно требуется указать *псевдоним* – уникальный текстовый идентификатор объекта (окно редактирования свойств объекта, вкладка **«Общие»** поле **Псевдоним**). Псевдоним требуется задавать буквами латинского алфавита.
- вызвать метод getRootObject, который позволяет получить ссылку на *Корневой элемент системы*, и далее использовать методы:
  - getChildrenObjsByTypes – позволяет получить список дочерних объектов указанного типа, существующих на текущий момент в конфигурации системы у данного объекта,
  - getChildrenObjs — позволяет получить список всех дочерних объектов данного объекта, существующих на данный момент в конфигурации системы.
- использовать метод getObjectsByFilter, который позволяет получить объекты определенного типа в соответствии с поставленным условием.

Получение доступа к сообщениям APACS 3000

Чтобы разработать скрипт, в который поступают сообщения APACS 3000, используйте следующее:

- для получения текущих сообщений переопределите тело функции

Apacs_Event(PropEvt),

- для получения доступа к сообщениям из базы данных используйте метод getEvents.

Получение уведомлений

Для получения уведомлений об изменении объктов APACS 3000 используйте следующие методы:

- set_onNotifyAdd – метод позволяет установить обработчик, который будет принимать уведомления о добавлении объектов,
- set_onNotifyChange – метод позволяет установить обработчик, который будет принимать уведомления об изменении настроек объектов,
- set_onNotifyDelete – метод позволяет установить обработчик, который будет принимать уведомления об удалении объектов.

Для получения уведомлений о сообщениях используйте метод set_onEvent.

Подробно о методах и функциях см. далее.

Представление параметров объектов и свойств сообщений APACS 3000 в скрипте

Для идентификации параметров объектов и сообщений APACS 3000 в скрипте используется список свойств следующего вида: имя (идентификатор) – значение.

Параметры могут быть следующих типов:

- **Числовые**, могут иметь следующие префиксы:
  - **n** — числовые параметры типа Integer (целочисленный, 32 бита, знаковый). Например, параметр **Номер объекта** объекта *Корневой элемент системы* имеет идентификатор nSiteID.
  - **b** — числовые параметры типа Byte (8 бит, беззнаковый). Например, параметр **Адрес связи через COM–порт** объекта *Контроллер AAN–100* имеет идентификатор bCommAddress.
  - **dw** — числовые параметры типа DWord (32 бита, беззнаковый). Например, параметр **Максимальное количество карт** объекта *Контроллер AAN–100* имеет идентификатор dwMaxNumCards.
  - **w** — числовые параметры типа Word (16 бит, знаковый). Например, параметр **Длительность импульса** объекта *Дополнительное реле считывателя* имеет идентификатор wPulseTime.
- **Строковые**, имеют префикс **str** (например, общее для всех сообщений свойство **Тип сообщения** имеет идентификатор strEventTypeID).
- **Бинарные массивы**, имеют префикс **blob** (например, параметр **Выберите термы с начальным значением ИСТИНА** объекта *Внутренняя переменная* имеет идентификатор blobTermsValue).

Обратите внимание: массивы, использующиеся в системе APACS 3000, не доступны из скриптов. Для работы с массивами необходимо использовать метод Script.toVBArray(cppArray) для преобразования массивов из формата APACS 3000 в формат, типичный для VBScript.

- **Дата и время**, имеют префикс **dt** (например, общий для всех объектов параметр **Дата создания** имеет идентификатор dtCreateTime).
- **Ссылка на объект конфигурации**, имеют префикс **SysAddr** (например, параметр **Первый формат** объекта *Список форматов карт* имеет идентификатор SysAddrCardFormats1).
  При этом значения могут являться:
  - ссылкой на объект системы (например, в поле **Первый формат** объекта *Список форматов карт* выбрано *Формат карт №1*),
  - NULL — в том случае, если данный параметр объекта не используется (в окне редактирования свойств объекта для параметра указано *не используется*).

Обратите внимание: идентификаторы объектов и сообщений, их свойств и команд, которые могут быть к ним применены, можно посмотреть в окне ***Просмотр метаданных*** (см. п. «Арс: Глава 5 Подсистемы, расширяющие возможности приложений «Консоль» и «Дежурный режим» 5.9 Клиентский модуль Просмотр метаданных).

3.1 Предопределенные функции в программах VBScript

В программах VBScript предопределены следующие функции:

- Apacs_Event(PropEvt),
- Script_Load(),
- Script_Close().

3.1.1 Функция Apacs_Event(PropEvt)

Основной функцией программы VBScript является функция Apacs_Event(PropEvt), где PropEvt — список текущих свойств сообщения. После того как функция была определена, она будет вызываться каждый раз когда в системе APACS 3000 регистрируется новое сообщение.

В теле функции Apacs_Event(PropEvt) можно задать последовательность действий системы и условия, при которых эти действия должны выполнятся.

Пример обработчика на поступление сообщения:

```
Sub Apacs_Event(Obj)
   MsgBox("Событие " + Obj.strEventTypeID + " на объекте " +
    Obj.strInitObjName)
End Sub
```

Пример реакции системы на сообщение *Доступ запрещен, необходимо предъявить карту (не только ПИН)*

```
Sub Apacs_Event(Evt)
```

```
'Выберем считыватель ReaderOnAIM4SL_A_One
'на сообщение "Доступ запрещен, необходимо предъявить карту (не
'только ПИН)" и выполним на считывателе команду Закрыть
If Evt.strEventTypeID = "TApISCEvAccReqNoCard_DeniedOnlyPIN" And
   Evt.strInitObjName = "ReaderOnAIM4SL_A_One" Then
    Evt.SysAddrInitObj.lockReader()
    Client.writeToLog("ReaderOnAIM4SL_A_One:lockReader")
 End If
End Sub
```

3.1.2 Функция Script_Load()

Функция Script_Load() позволяет получать оповещения о загрузке скрипта.

Оповещение о загрузке может быть использовано:

- для оповещения о запуске приложения (только для клиентской автоматизации),
- для установки связи с COM–объектами (например, для своевременной загрузки MS Word),
- для установки начальных значений глобальных переменных (например, может быть задана переменная, в которую сохраняется общее число сообщений за время работы клиентского приложения).

Пример оповещения о загрузке программы:

```
Sub Script_Load
  MsgBox("Скрипт загружен")
End Sub
```

3.1.3 Функция Script_Close()

Функция Script_Close() позволяет получать уведомление о том, что скрипт будет сейчас выгружен.

Оповещение о завершении работы может быть использовано:

- для завершения связи с COM–объектами,
- для освобождения ресурсов.

Пример оповещения о завершении работы программы:

```
Sub Script_Close
  MsgBox("Скрипт выгружается")
End Sub
```

3.2 Предопределенные объекты в программах и макрокомандах VBScript

В программах и макрокомандах VBScript предопределены следующие объекты:

- Основные:
  - *Apacs* — методы объекта *Apacs* позволяют получать доступ к объектам на сервере APACS 3000. Объект поддерживает интерфейс IApcServerWrap.

- *Log* — метод объекта *Log* позволяет выводить отладочную информацию в окна утилит «Блокирующий просмотр логов» и «Просмотр логов» (см. раздел «Утилиты»). Объект поддерживает интерфейс IApcLogWrap.
- *Client* — методы объекта *Client* предоставляют доступ к локальным ресурсам компьютера, на котором запущено клиентское приложение APACS 3000. Объект реализован только на клиенте APACS 3000, и не поддерживается на сервере APACS 3000. Объект поддерживает интерфейс IApcClientWrap.
- *Script* — поддерживает интерфейс IApcScriptWrap.
- *VideoLinkerWrap* — поддерживает интерфейс IApcVideoLinkerWrap, имеет ряд вспомогательных функций, предоставляет интерфейс к связыванию видеофрагментов.

• Вспомогательные:
- *Объекты системы* — поддерживают интерфейс IApcObjectWrap.
- класс *TApcEQUALObjFilter* — экземпляры этого класса позволяют задать фильтр на эквивалентность, реализуют интерфейс IApcEQUALObjFilter.
- класс *TApcANDObjFilter* — экземпляры этого класса позволяют объеденить фильтры по логическому *И*, реализуют интерфейс IApcANDObjFilter.

Объекты	Клиентская автоматизация	Серверная автоматизация
Apacs	✓	✓
Log	✓	✓
Client	✓	✗
Script	✓	✓
VideoLinker	не рекомендуется	✓

Таблица показывает, где могут быть использованы предопределенные объекты VBScript

3.2.1 Интерфейс IApcServerWrap

Интерфейс сервера APACS 3000. Предоставляет следующие методы:

• getObjectByAlias

- getRootObject
- getObjectByUID
- getObjectsByFilter
- getEvents
- set_onEvent
- set_onNotifyAdd
- set_onNotifyChange
- set_onNotifyDelete

3.2.1.1 Метод IApcServerWrap.getObjectByAlias

Метод предназначен для получения ссылки на объект по его псевдониму.

VBScript

Apacs.getObjectByAlias(strAlias, Obj),
где:
strAlias — псевдоним объекта,
Obj — возвращаемое значение, ссылка на объект системы.
В случае успешного выполнения метод возвращает *0*.
Возможные причины невыполнения:

- нет связи с сервером,
- объекта с таким псевдонимом не существует в системе,
- нет прав на просмотр этого объекта (о правах см. п. «Apa: Глава 2 Права и аудит»).


Пример доступа к объекту системы из скрипта:

```
'Получим объект с псевдонимом sample и выведем его имя
If Apacs.getObjectByAlias("sample", obj) <> 0 Then
  Client.writeToLog "Не могу получить объект с псевдонимом sample"
Else
  If obj.getCurrentSettings(prop) <> 0 Then
    Client.writeToLog "Ошибка получения свойств объекта"
  Else
    msgbox "Имя объекта: " & prop.strName
  End If
End If
```

C++

HRESULT **getObjectByAlias**(BSTR astrObjAlias/*[in]*/,
LPDISPATCH* appObject/*[out]*/,
int* apnResult/*[out,retval]*/),

Параметры
[in] astrObjAlias — псевдоним объекта,
[out] appObject — указатель на объект, поддерживающий интерфейс IApcObjectWrap,
[out] apnResult — успешное выполнение — *0*, иначе — код ошибки.
Возвращает значение HRESULT.

3.2.1.2 Метод IApcServerWrap.getRootObject

Метод предназначен для получения ссылки на *Корневой элемент системы*.

VBScript

Apacs.getRootObject(RootObj),

где RootObj — возвращаемое значение, ссылка на *Корневой элемент системы*.

В случае успешного выполнения метод возвращает *0*.

Возможные причины невыполнения:

- нет связи с сервером.

Пример изменения имени корня системы:

```
If Apacs.getRootObject(objRoot) <> 0 Then
  Client.writeToLog "Не могу получить корневой объект"
Else
  If objRoot.getCurrentSettings(prop) <> 0 Then
    Client.writeToLog "Ошибка получения свойств объекта"
  Else
    prop.strName = "Новое имя"
    If objRoot.applySettings(prop) <> 0 Then
      Client.writetoLog "Ошибка применения свойств"
    Else
      msgbox "Имя корня изменено"
    End If
  End If
End If
```

C++

HRESULT **getRootObject**(LPDISPATCH* appObject/*[out]*/,
int* apnResult/*[out,retval]*/)

Параметры

[out] appObject — указатель на объект, поддерживающий интерфейс IApcObjectWrap,

[out] apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.1.3 Метод IApcServerWrap.getObjectByUID

Метод предназначен для получения ссылки на объект по его строковому идентификатору в системе.

VBScript

getObjectByUID(strUID, Obj),

где:

strUID — строковое представление идентификатора объекта,

Obj — возвращаемое значение, ссылка на объект системы.

Формат строкового представления идентификатора объекта (strUID) имеет следующий вид: SA XXXX.YYYYYYYY, где XXXX – первая часть

системного адреса в шестнадцатеричном формате, а YYYYYYYY – вторая часть системного адреса в шестнадцатеричном формате.

Например, строковое представление идентификатора объекта может быть таким: SA 0000.0000000A

В случае успешного выполнения метод возвращает *0*.
Возможные причины невыполнения:

- нет связи с сервером,
- объекта с таким strUID не существует в системе.

Пример

```
'Получим объект с UID SA 0000.00000001 и выведем его имя
If Apacs.getObjectByUID("SA 0000.00000001", obj) <> 0 Then
  Client.writeToLog "Не могу получить объект с UID SA 0000.00000001"
Else
  If obj.getCurrentSettings(prop) <> 0 Then
    Client.writeToLog "Ошибка получения свойств объекта"
  Else
    msgbox "Имя объекта: " & prop.strName
  End If
End If
```

C++

HRESULT **getObjectByUID**(BSTR astrUID/*[in]*/,
LPDISPATCH* appObject/*[out]*/,
int* apnResult/*[out,retval]*/)

Параметры

[in] astrUID — идентификатор объекта,
[out] appObject — указатель на объект, поддерживающий интерфейс IApcObjectWrap,
[out] apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.1.4 Метод IApcServerWrap.getObjectsByFilter

Метод позволяет получить объекты указанного типа в соответствии с условием. В процессе выполнения метод формирует SQL–запрос.

VBScript

getObjectsByFilter (astrObjType, apFilter, appObjs),
где:

astrObjType — тип объектов для фильтрации,
apFilter — условие фильтрации, в качестве фильтра могут выступать объекты, поддерживающие интерфейсы IApcEQUALObjFilter и IApcANDObjFilter,
если в качестве значения параметра apFilter задано nothing, то возвращаться будут все записи без применения фильтра,

appObjs — возвращаемое значение, список объектов, удовлетворяющих фильтру.

Возвращаемое значение

В случае успешного выполнения метод возвращает *0*.

Возможные причины невыполнения:

- нет связи с сервером.

Пример получения владельцев карт по условию фильтра:

```
'Найдем фамилии владельцев карт с именем Сергей и отчеством
    Александрович
'Условие на имя
set objFirstNameCondition = CreateObject("ApcSrvSDK.TApcEQUALObjFilter")
objFirstNameCondition.strName = "strFirstName"
objFirstNameCondition.Value = "Сергей"
'Условие на отчество
set objMiddleNameCondition = CreateObject("ApcSrvSDK.TApcEQUALObjFilter")
objMiddleNameCondition.strName = "strMiddleName"
objMiddleNameCondition.Value = "Александрович"
'Объединение условий
set objANDFilter = CreateObject("ApcSrvSDK.TApcANDObjFilter")
objANDFilter.addCondition objFirstNameCondition
objANDFilter.addCondition objMiddleNameCondition
'Получаем объекты с помощью фильтра
If Apacs.getObjectsByFilter("TApcCardHolder", objANDFilter, listObjs) <> 0 Then
  Client.writeToLog "Ошибка получения объектов по фильтру"
Else
  first = true
  strRes = ""
  For i = 0 To UBound (listObjs)
    If listObjs(i).getCurrentSettings(prop) = 0 Then
      If first Then
        strRes = prop.strLastName
        first = false
      Else
        strRes = strRes + vbNewLine + prop.strLastName
      End If
    End If
  Next
    MsgBox "Фамилии владельцев карт : " + vbNewLine + strRes
End If
```

Пример получения владельцев карт по фильтру без условия

```
if Apacs.getObjectsByFilter("TApcCardHolder", nothing, listH) <> 0 then
  msgError "getObjectsByFilter fail"
```

```
    exit function
  End if
```

C++

```
HRESULT getObjectsByFilter(BSTR astrObjType/*[in]*/,
                           LPDISPATCH apFilter/*[in]*/,
                           LPSAFEARRAY* appObjs/*[out]*/,
                           int* apnResult/*[out,retval]*/)
```

Параметры:

[in] astrObjType — тип объектов для фильтрации,

[in] apFilter — условие фильтрации, в качестве фильтра может выступать объекты, поддерживающие интерфейсы IApcEQUALObjFilter и IApcANDObjFilter,

[out] appObjs — возвращаемое значение, список объектов, удовлетворяющих фильтру,

[out] apnResult — успешное выполнение — 0, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.1.5 Метод IApcServerWrap.getEvents

Метод позволяет получать сообщения из базы данных.

VBScript

```
getEvents (arrEventTypes, dtFrom, dtTo, arrEvents ),
```

где:

arrEventTypes — список типов сообщений,

adtFrom — начиная с этого момента времени будем получать сообщения,

adtTo — момент окончания получения сообщений,

arrEvents — возвращаемое значение, список сообщений.

Обратите внимание: временем прихода сообщения считается время его возникновения, а не время регистрации на сервере.

В случае успешного выполнения метод возвращает *0*.

Пример получения общего количества сообщений типа *Доступ разрешен* за день:

```
If Apacs.getEvents(Array("TApcCardHolderAccess_Granted"), DateAdd("d", -1,
    Now()), Now(), evtProps) <> 0 Then
  Client.writeToLog "Ошибка получения сообщений"
Else
  msgbox "Общее число сообщений за день: " & (UBound(evtProps) + 1)
End If
```

C++

```
HRESULT getEvents(LPSAFEARRAY aEventTypes/*[in]*/,
                  DATE adtFrom/*[in]*/,
```

DATE adtTo/*[in]*/, LPSAFEARRAY* appEvents/*[out]*/,
int* apnResult/*[out,retval]*/)

Параметры

[in] aEventTypes — массив строк типов сообщений,

[in] adtFrom — начиная с этого момента времени будем получать сообщения,

[in] adtTo — момент окончания получения сообщений,

[out] appEvents — список сообщений,

[out] apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.1.6 Метод IApcServerWrap.set_onEvent

Метод позволяет установить обработчик, который будет принимать сообщения, происходящие в системе APACS 3000. При возникновении сообщения в системе для объекта, переданного в функцию, будет вызван метод Invoke с DispID=0 и с параметром «настройки сообщения».

Метод рекомендуется использовать в рамках SDK.

VBScript

set_onEvent (subRef),

где subRef — указатель на функцию, которая будет вызвана при поступлении сообщения в системе.

В случае успешного выполнения метод возвращает *0*.

Пример установки обработчика, принимающего сообщения системы:

```
Apacs.set_onEvent getref("Apacs_Event")
sub Apacs_Event(Evt)
  Client.writeToLog "Получено сообщение: " & Evt.SysAddrEventID & ", тип
    сообщения: " & Evt.strEventTypeID
end sub
```

Обратите внимание: если Вы установили обработчик событий, перед завершением использования интерфейса IApcServerWrap, чтобы избежать зацикливания объектных ссылок, требуется установить обработчик в nothing.

C++

HRESULT **set_onEvent**(LPDISPATCH apEventHandler/*[in]*/)

Параметры

[in] apEventHandler — обработчик сообщений.

Возвращает значение HRESULT.

Обратите внимание: если Вы установили обработчик событий, перед завершением использования интерфейса IApcServerWrap, чтобы избежать зацикливания объектных ссылок, требуется установить обработчик в NULL.

3.2.1.7 Метод IApcServerWrap.set_onNotifyAdd

Метод позволяет установить обработчик, который будет принимать уведомления о добавлении объектов в системе APACS 3000. При добавлении объекта в системе для объекта, переданного в функцию, будет вызван метод Invoke с DispID=0 и с параметром «добавленный объект».

VBScript

set_onNotifyAdd (subRef),

где subRef — указатель на функцию, которая будет вызвана при добавлении объекта в системе.

В случае успешного выполнения метод возвращает *0*.

Пример установки обработчика, принимающего уведомления о добавлении объектов в системе:

```
Apacs.set_onNotifyAdd getref("APACS 3000_NotifyAdd")
Sub APACS 3000_NotifyAdd(Obj)
  Obj.getUID strUID
  Obj.getApacsType strType
  Client.writeToLog "Добавлен объект: " + strUID + ", тип объекта: " + strType
End Sub
```

Обратите внимание: если Вы установили обработчик событий, перед завершением использования интерфейса IApcServerWrap, чтобы избежать зацикливания объектных ссылок, требуется установить обработчик в nothing.

C++

HRESULT **set_onNotifyAdd**(LPDISPATCH
apEventHandler/*[in]*/)

Параметры

[in] apEventHandler — обработчик уведомлений о добавлении объектов.

Возвращает значение HRESULT.

Обратите внимание: если Вы установили обработчик уведомления о добавлении объектов, перед завершением использования интерфейса IApcServerWrap, чтобы избежать зацикливания объектных ссылок, требуется установить обработчик в NULL.

3.2.1.8 Метод IApcServerWrap.set_onNotifyChange

Метод позволяет установить обработчик, который будет принимать уведомления об изменении настроек объектов в системе APACS 3000. При изменении настроек объекта в системе для переданного в функцию объекта будет вызван метод Invoke с DispID=0, с первым параметром «измененный объект» и со вторым параметром «измененные настройки объекта».

VBScript

set_onNotifyChange (subRef, changedProp),

где subRef — указатель на функцию, которая будет вызвана при изменении настроек объектов в системе,

changedProp — список изменившихся настроек объекта.

В случае успешного выполнения метод возвращает *0*.

 Пример установки обработчика, принимающего уведомления об изменении настроек объектов в системе:

```
Apacs.set_onNotifyChange getref("APACS 3000_NotifyChange")
   Sub APACS 3000_NotifyChange(Obj, ChangedProp)
     Obj.getUID strUID
     Obj.getApacsType strType
     Client.writeToLog strType
   End Sub
```

 Обратите внимание: если Вы установили обработчик событий, перед завершением использования интерфейса IApcServerWrap, чтобы избежать зацикливания объектных ссылок, требуется установить обработчик в nothing.

C++

HRESULT **set_onNotifyChange**(LPDISPATCH
apEventHandler/*[in]*/)

Параметры

[in] apEventHandler — обработчик уведомлений об изменении настроек объектов.

Возвращает значение HRESULT.

 Обратите внимание: если Вы установили обработчик уведомления об изменении настроек объектов, перед завершением использования интерфейса IApcServerWrap, чтобы избежать зацикливания объектных ссылок, требуется установить обработчик в NULL.

3.2.1.9 Метод IApcServerWrap.set_onNotifyDelete

Метод позволяет установить обработчик, который будет принимать уведомления об удалении объектов в системе APACS 3000. При удалении объекта в системе для переданного в функцию объекта будет вызван метод Invoke с DispID=0 и с параметром «удаленный объект».

 Обратите внимание: так как объект уже будет удален в системе, для этого объекта будет успешно выполнена только функция getUID.

VBScript

set_onNotifyDelete (subRef),

где subRef — указатель на функцию, которая будет вызвана при удалении объекта в системе.

В случае успешного выполнения метод возвращает *0*.

Пример установки обработчика, принимающего уведомления об удалении объектов в системе:

```
Apacs.set_onNotifyDelete getref("APACS 3000_NotifyDelete")
Sub APACS 3000_NotifyDelete(Obj)
  Obj.getUID strUID
  Obj.getApacsType strType
  Client.writeToLog "Удален объект: " + strUID + ", тип объекта: " + strType
End Sub
```

Обратите внимание: если Вы установили обработчик событий, перед завершением использования интерфейса IApcServerWrap, чтобы избежать зацикливания объектных ссылок, требуется установить обработчик в nothing.

C++

HRESULT **set_onNotifyDelete**(LPDISPATCH
apEventHandler/*[in]*/)

Параметры

[in] apEventHandler — обработчик уведомлений об удалении объекта.

Возвращает значение HRESULT.

Обратите внимание: если Вы установили обработчик уведомления об удалении объектов, перед завершением использования интерфейса IApcServerWrap, чтобы избежать зацикливания объектных ссылок, требуется установить обработчик в NULL.

3.2.2 Интерфейс IApcLogWrap

Интерфейс объекта *Log*. Предоставляет метод logBAD.

3.2.2.1 Метод IApcLogWrap.logBAD

Метод позволяет отправлять сведения в подсистему вывода отладочной информации (см. раздел «Утилиты»).

VBScript

logBAD(strLog),

где strLog — строка с информацией.

Пример вывода строки в окно утилиты «просмотр логов»:

```
Log.logBAD("Пример вывода в лог")
```

3.2.3 Интерфейс IApcClientWrap

Интерфейс объекта *Client*. Доступен только на клиенте APACS 3000. Предоставляет следующие методы:

- playSound,
- showVideo,
- hideVideo,
- writeToLog.

3.2.3.1 Метод IApcClientWrap.playSound

Метод предназначен для проигрывания звукового файла. Поддерживаются файлы в формате *.wav.

VBScript

playSound(strFile),
где strFile — путь к внешнему файлу.

Обратите внимание: путь к внешнему файлу должен быть указан относительно того компьютера, на котором будет проигрываться файл.

Пример проигрывания файла:

```
'Проиграть звуковой файл C:\Windows\Media\Tada.wav
Client.playSound "C:\Windows\Media\Tada.wav"
```

3.2.3.2 Метод IApcClientWrap.showVideo

Метод позволяет открыть окно с изображение от указанного видеоисточника.

VBScript

showVideo(objCam, dwLeft, dwTop, dwWidth, dwHeight),
где:

objCam — объект системы, содержащий информацию о камере,
dwLeft — положение окна с видеоизображением, отступ от левого края монитора (пиксели),
dwTop — окна с видеоизображением, отступ от верхнего края монитора (пиксели),
dwWidth — ширина окна с видеоизображением (пиксели),
dwHeight — высота окна с видеоизображением (пиксели).

Пример показа видео с камеры с псевдонимом cam:

```
If Apacs.getObjectByAlias("cam", objCam) <> 0 Then
  Client.writeToLog "Не могу найти объект с псевдонимом cam"
Else
  Client.showVideo objCam, 50, 50, 640, 480
End If
```

3.2.3.3 Метод IApcClientWrap.hideVideo

Метод позволяет закрыть окно с видеоизображением.

VBScript

hideVideo(objCam),
где objCam — объект системы, содержащий информацию о камере.

Пример завершения показа видео с камеры с псевдонимом cam:

```
If Apacs.getObjectByAlias("cam", objCam) <> 0 Then
  Client.writeToLog "Не могу найти объект с псевдонимом cam"
Else
  Client.hideVideo objCam
End If
```

3.2.3.4 Метод IApcClientWrap.writeToLog

Метод предназначен для вывода строки в окно ***Журнал работы*** (см. «Apc: Глава 2 Общие модули приложений»).

VBScript

writeToLog(strLog),
где strLog — строка с информацией.

Пример вывод строки в окно ***Журнал работы***:

```
Client.writeToLog("Скрипт в работе!")
```

3.2.4 Интерфейс IApcScriptWrap

Интерфейс объекта *Script*. Использует следующие методы:

- sleep,
- работа с таймерами:
  - setTimeout,
  - clearTimeout,
  - setInterval,
  - clearInterval,
- работа с массивами
  - toVBArray,
  - toCPPArray,
  - loadCPPArray,
  - saveCPPArray.

3.2.4.1 Метод IApcScriptWrap.sleep

Метод позволяет приостанавливать выполнение скрипта на указанное время.

VBScript

sleep(dwMillesecond),
где dwMillesecond — время в милисекундах.

Пример:

```
' Приостановить выполнение программы
Client.writeToLog "Начало: " + CStr(Now())
Script.sleep(1000)
Client.writeToLog "Конец: " + CStr(Now())
```

Обратите внимание: избегайте длительной приостановки выполнения скрипта, так как сообщения, поступившее за это время, будут не обработаны подсистемой автоматизации.

3.2.4.2 Работа с таймерами

Метод IApcScriptWrap.setTimeout

Метод предназначен для однократного вызова функции через указанное время. В случае успешного выполнения метод возвращает идентификатор, который можно использовать для отмены вызова функции при помощи метода clearTimeout (см. далее).

VBScript

setTimeout(subRef, dwMillesecond),
где:

subRef — указатель на функцию, которая будет однократно вызвана по прошествии указанного времени;
dwMillesecond — время в милисекундах.

Метод IApcScriptWrap.clearTimeout

Метод предназначен для сброса таймера, созданного ранее при помощи метода setTimeout.

VBScript

clearTimeout(id),
где id — идентификатор, возвращенный при выполнении метода setTimeout.

Пример работы с методами setTimeout и clearTimeout:

```
'Установить два таймера, срабатывающих однократно через
  установленный интервал времени,
'первый из них установим, и он сработает, другой установим и сбросим,
  и он не сработает
Script.setTimeout getref("timer1"), 1000
dim id2
id2 = Script.setTimeout(getref("timer2"), 1000)
Script.clearTimeout id2
Sub timer1()
  msgbox "Сработал таймер1"
End Sub
Sub timer2()
```

```
  msgbox "Сработал таймер2"
End Sub
```

Метод IApcScriptWrap.setInterval

Метод предназначен для многократного вызова функции через указанный интервал времени. В случае успешного выполнения метод возвращает идентификатор, который можно использовать для отмены вызова функции при помощи метода clearInterval (см. далее).

VBScript

setInterval(subRef, dwMillesecond)
где:

subRef — указатель на функцию, которая будет многократно вызываться через указанный интервал времени;

dwMillesecond — время в милисекундах.

Метод IApcScriptWrap.clearInterval

Метод предназначен для сброса таймера, созданного ранее при помощи метода setInterval.

VBScript

clearInterval(id),
где id — идентификатор, возвращенный при выполнении метода setInterval.

Пример работы с методами setInterval и clearInterval:

```
'Установим таймер, срабатывающий многократно через установленный
    интервал времени,
  'таймер сработает два раза, и мы его сбросим
dim Count
Count = 0
dim id
id = Script.setInterval(getref("timer"), 1000)
Sub timer()
  If Count = 0 Then
    msgbox "Таймер сработал первый раз"
  End If
  If Count = 1 Then
    msgbox "Таймер сработал второй раз"
    Script.clearInterval id
  End If
  If Count > 1 Then
    msgbox "Таймер сработал более одного раза"
  End If
  Count = Count + 1
End Sub
```

3.2.4.3 Работа с массивами

Метод IApcScriptWrap.toVBArray

Метод предназначен для преобразования массивов, используемых в системе APACS 3000, в формат массивов, принятых в VBScript.

VBScript

toVBArray(cppArray),
где cppArray — массив в формате, принятом в системе APACS 3000.
Возвращаемое значение
В случае успешного выполнения метод возвращает массив в формате, принятом в VBScript.

Метод IApcScriptWrap.toCPPArray

Метод предназначен для преобразования массивов, используемых в VBScript, в формат массивов, принятых в системе APACS 3000.

VBScript

toCPPArray(vbArray),
где vbArray — массив в формате, принятом в VBScript.
Возвращаемое значение
В случае успешного выполнения метод возвращает массив в формате, принятом в APACS 3000.


Пример работы с методами toVBArray и toCPPArray:

```
' Выставление всех термов ВП с псевдонимом iv в значение ложь
If Apacs.getObjectByAlias("iv", iv) <> 0 Then
  Client.writeToLog "Не могу найти объект с псевдонимом iv"
Else
  If iv.getCurrentSettings(prop) <> 0 Then
    Client.writeToLog "Ошибка получения свойств ВП"
  Else
    terms = Script.toVBArray(prop.blobTermsValue)
    For i = 0 To UBound(terms)
      terms(i) = 0
    Next
    prop.blobTermsValue = Script.toCPPArray(terms)
    If iv.applySettings(prop) <> 0 Then
      Client.writeToLog "Ошибка изменения свойств объекта"
    Else
      MsgBox "Внутренняя переменная успешно изменена"
    End If
  End If
End If
```

Метод IApcScriptWrap.loadCPPArray

Метод предназначен для загрузки в переменную APACS 3000 массива из указанного файла.

VBScript

loadCPPArray(strFile, cppArray),
где:
strFile — путь к внешнему файлу,
cppArray — массив в формате, принятом в системе APACS 3000.
В случае успешного выполнения метод возвращает *0*.
Возможные причины невыполнения:
- ошибка работы с файловой системой.

Метод IApcScriptWrap.saveCPPArray

Метод предназначен для сохранения в указанный файл массива в формате APACS 3000.

VBScript

saveCPPArray(strFile, cppArray),
где:
strFile — путь к внешнему файлу,
cppArray — массив в формате, принятом в системе APACS 3000.
В случае успешного выполнения метод возвращает *0*.
Возможные причины невыполнения:
- ошибка работы с файловой системой.

Пример работы с методами saveCPPArray и loadCPPArray:

```
' Сохранение фотографии владельца на диск и загрузка другой фотографии
' Для работы скрипта необходим владелец карты с псевдонимом hld и фотографией типа jpeg
If Apacs.getObjectByAlias("hld", hld) <> 0 Then
  Client.writeToLog "Не могу найти объект с псевдонимом hld"
Else
  If hld.getChildrenObjsByTypes(Array("TApcCHMainPhoto"), photos) <> 0 Then
   Client.writeToLog "У владельца нет фотографии"
  Else
   If UBound(photos) < 0 Then
    Client.writeToLog "У владельца нет фотографии"
   Else
    If photos(0).getCurrentSettings(prop) <> 0 Then
     Client.writeToLog "Ошибка получения фотографии"
    Else
     If Script.saveCPPArray("C:\Documents and Settings\All Users\Application
     Data\photosrc.jpg", prop.binBufPhoto) <> 0 Then
       Client.writeToLog "Ошибка сохранения в файл"
     Else
      If Script.loadCPPArray("C:\Documents and Settings\All Users\Application
     Data\photodst.jpg", blobPhoto) <> 0 Then
         Client.writeToLog "Ошибка чтения из файла"
```

```
            Else
              prop.binBufPhoto = blobPhoto
              If photos(0).applySettings(prop) <> 0 Then
                Client.writeToLog "Ошибка изменения фотографии"
              Else
                MsgBox "Фотография изменена успешно"
              End If
            End If
          End If
        End If
      End If
    End If
End If
```

3.2.5 Интерфейс IApcObjectWrap

Интерфейс объекта конфигурации в системе APACS 3000. Предоставляет следующие методы:

- getCurrentSettings
- applySettings
- getChildrenObjsByTypes
- getApacsType
- getChildrenObjs
- getParentObject
- getUID
- getChildSettingsForAdd
- addChildWithSettings
- deleteObject
- getEventSettingsForRegister
- registerEventWithSettings
- <команда>

3.2.5.1 Метод IApcObjectWrap.getCurrentSettings

Метод позволяет получить текущие настройки объекта.

VBScript

getCurrentSettings(PropObj),
где PropObj — возвращаемое значение, текущие параметры объекта.
В случае успешного выполнения метод возвращает *0*.
Возможные причины невыполнения:

- нет связи с сервером,
- объект удален.


Пример получения имени корня системы

```
If Apacs.getRootObject(objRoot) <> 0 Then
  Client.writeToLog "Не могу получить корневой объект"
Else
  If objRoot.getCurrentSettings(prop) <> 0 Then
    Client.writeToLog "Ошибка получения свойств объекта"
```

```
    Else
      msgbox "Имя корня: " + prop.strName
    End If
  End If
```

C++

```
HRESULT getCurrentSettings(LPDISPATCH*
                           appSettings/*[out]*/,
                           int* apnResult/*[out,retval]*/)
```

Параметры

[out] appSettings — текущие настройки объекта,

[out] apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.5.2 Метод IApcObjectWrap.applySettings

Метод позволяет сохранить на сервере APACS 3000 переданный список измененных параметров объекта.

VBScript

applySettings(PropObj),

где PropObj — список новых параметров объекта.

В случае успешного выполнения метод возвращает *0*.

Возможные причины невыполнения:

- нет связи с сервером,
- объект удален,
- введены некорректные значения параметров,
- нет прав на редактирование объекта (о правах см. п. «Apa: Глава 2 Права и аудит»)

Пример изменения свойства объекта:

```
' Изменение имени корня системы
If Apacs.getRootObject(objRoot) <> 0 Then
  Client.writeToLog "Не могу получить корневой объект"
Else
  If objRoot.getCurrentSettings(prop) <> 0 Then
    Client.writeToLog "Ошибка получения свойств объекта"
  Else
    prop.strName = "Новое имя"
    If objRoot.applySettings(prop) <> 0 Then
      Client.writetoLog "Ошибка применения свойств"
    Else
      msgbox "Имя корня изменено"
    End If
  End If
End If
```

C++

```
HRESULT applySettings(LPDISPATCH apSettings/*[in]*/,
```

int* apnResult/*[out,retval]*/)

Параметры

[in] apSettings — новые настройки объекта,

[out] apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.5.3 Метод IApcObjectWrap.getChildrenObjsByTypes

Метод предназначен для получения списка дочерних объектов указанного типа, существующих на текущий момент в конфигурации системы у данного объекта.

VBScript

getChildrenObjsByTypes(arrType, arrObjs),

где:

arrType — список типов дочерних объектов,

arrObjs — возвращаемое значение, список дочерних объектов указанного типа, существующих на текущий момент в конфигурации системы у данного объекта.

В случае успешного выполнения метод возвращает *0*.

Возможные причины невыполнения:

- нет связи с сервером,
- объект удален.

Пример получения имен всех папок, прикрепленных к корню:

```
If Apacs.getRootObject(objRoot) = 0 Then
  Client.writeToLog "Не могу получить корневой объект"
  If objRoot.getChildrenObjsByTypes(Array("TApcFolder"), chlds) <> 0 Then
    Client.writeToLog "Ошибка получения дочерних элементов корня"
  Else
    first = true
    strChlds = ""
    For i = 0 To UBound (chlds)
      If chlds(i).getCurrentSettings(prop) = 0 Then
        If first Then
          strChlds = prop.strName
          first = false
        Else
          strChlds = strChlds + vbNewLine + prop.strName
        End If
      End If
    Next
    MsgBox "Дочерние объекты корня типа Папка: " + vbNewLine + strChlds
  End If
End If
```

C++

HRESULT **getChildrenObjsByTypes**(LPSAFEARRAY aObjTypes/*[in]*/,
LPSAFEARRAY* apChildrenObjs/*[out]*/,

int* apnResult/*[out,retval]*/)

Параметры

[in] aObjTypes — список типов дочерних объектов,

[out apChildrenObjs — массив объектов, поддерживающих интерфейс IApcObjectWrap,

[out apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.5.4 Метод IApcObjectWrap.getApacsType

Метод позволяет получить строковый идентификатор типа данного объекта.

VBScript

getApacsType(strType),

где strType — возвращаемое значение, тип объекта в системе.

В случае успешного выполнения метод возвращает 0.

Возможные причины невыполнения:

- нет связи с сервером;
- объект удален.

Пример определения типа объекта:

```
If Apacs.getRootObject(objRoot) <> 0 Then
  Client.writeToLog "Не могу получить корневой объект"
Else
  If objRoot.getApacsType(strType) <> 0 Then
    Client.writeToLog "Ошибка получения типа объекта"
  Else
    msgbox "Тип корня = " + strType
  End If
End If
```

C++

HRESULT **getApacsType**(BSTR* astrApacsType/*[out]*/,
int* apnResult/*[out,retval]*/)

Параметры

[out astrApacsType — тип объекта,

[out apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.5.5 Метод IApcObjectWrap.getChildrenObjs

Метод позволяет получить список всех дочерних объектов данного объекта, существующих на данный момент в конфигурации системы.

VBScript

getChildrenObjs(arrObjs),

где arrObjs — возвращаемое значение, список всех дочерних объектов данного объекта, существующих на данный момент в конфигурации системы.

В случае успешного выполнения метод возвращает *0*.

Возможные причины невыполнения:

- нет связи с сервером;
- объект удален.

Пример получения имен дочерних элементов корня:

```
If Apacs.getRootObject(objRoot) <> 0 Then
  Client.writeToLog "Не могу получить корневой объект"
Else
  If objRoot.getChildrenObjs(chlds) <> 0 Then
    Client.writeToLog "Ошибка получения дочерних элементов корня"
  Else
    first = true
    strChlds = ""
    For i = 0 To UBound (chlds)
      If chlds(i).getCurrentSettings(prop) = 0 Then
        If first Then
          strChlds = prop.strName
          first = false
        Else
          strChlds = strChlds + vbNewLine + prop.strName
        End If
      End If
    Next
    MsgBox "Дочерние объекты корня : " + vbNewLine + strChlds
  End If
End If
```

C++

```
HRESULT getChildrenObjs(LPSAFEARRAY*
                        apChildrenObjs/*[out]*/,
                        int* apnResult/*[out,retval]*/),
```

Параметры

[out apChildrenObjs — массив объектов, поддерживающих интерфейс IApcObjectWrap,

[out apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.5.6 Метод IApcObjectWrap.getParentObject

Метод предназначен для получения родительского объекта указанного объекта.

VBScript

```
getParentObject(parentObj),
```

где parentObj — возвращаемое значение, родительский объект указанного объекта.

В случае успешного выполнения метод возвращает *0*.

Возможные причины невыполнения:

- нет связи с сервером,
- объект удален,
- объект является *Корневым элементом системы*.

 Пример получения имени родительского объекта для объекта с псевдонимом contr

```
If Apacs.getObjectByAlias("contr", obj) <> 0 Then
  Client.writeToLog "Не могу получить объект с псевдонимом contr"
Else
  If obj.getParentObject(objParent) <> 0 Then
    Client.writeToLog "Ошибка получения родительского объекта"
  Else
    If objParent.getCurrentSettings(prop) = 0 Then
      msgbox "Имя родительского объекта: " + prop.strName
    End If
  End If
End If
```

C++

HRESULT **getParentObject**(LPDISPATCH* appObject/*[out]*/,
int* apnResult/*[out,retval]*/)

Параметры

[out appObject — указатель на родительский объект, поддерживающий интерфейс IApcObjectWrap,

[out apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.5.7 Метод IApcObjectWrap.getUID

Метод позволяет получить строковое представление идентификатора данного объекта.

VBScript

getUID(strUID),

где strUID — возвращаемое значение, идентификатор объекта.

Формат возвращаемого значения (strUID) имеет следующий вид: SA XXXX.YYYYYYYY, где XXXX – первая часть системного адреса в шестнадцатеричном формате, а YYYYYYYY – вторая часть системного адреса в шестнадцатеричном формате.

В случае успешного выполнения метод возвращает *0*.

Возможные причины невыполнения:

- нет связи с сервером,
- объект удален.

 Пример получения идентификатора объекта

```
If Apacs.getRootObject(objRoot) <> 0 Then
  Client.writeToLog "Не могу получить корневой объект"
Else
```

```
If objRoot.getUID(strUID) <> 0 Then
  Client.writeToLog "Ошибка получения идентификатора объекта"
Else
  msgbox "Идентификатор корня = " + strUID
End If
End If
```

C++

```
HRESULT getUID(BSTR* astrUID/*[out]*/,
               int* apnResult/*[out,retval]*/),
```

Параметры

[out astrUID — идентификатор объекта,

[out apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.5.8 Метод IApcObjectWrap.getChildSettingsForAdd

Метод позволяет получить настройки по умолчанию дочернего объекта для его добавления в систему.

VBScript

```
getChildSettingsForAdd (strObjType, propObj),
```

где:

strObjType — тип добавляемого объекта,

propObj — возвращаемое значение, настройки нового объекта по умолчанию.

В случае успешного выполнения метод возвращает *0*.

Возможные причины невыполнения:

- нет связи с сервером,
- объектов такого типа больше нельзя добавить в конфигурацию системы (существуют ограничения по конфигурированию либо ограничения в лицензии).

Пример добавления объекта типа *Папка*:

```
If Apacs.getRootObject(ObjRoot) <> 0 Then
  Client.writeToLog "Не могу получить корневой объект"
Else
  If objRoot.getChildSettingsForAdd("TApcFolder", prop) <> 0 Then
    Client.writeToLog "Ошибка получения свойств объекта для добавления"
  Else
    prop.strName = "Новая папка"
    If objRoot.addChildWithSettings(prop, objFld) <> 0 Then
      Client.writeToLog "Не могу добавить папку к корню"
    Else
      msgbox "Папка добавлена к корню"
    End If
  End If
End If
```

C++

HRESULT **getChildSettingsForAdd**(BSTR aObjType/*[in]*/,
LPDISPATCH* apSettings/*[out]*/,
int* apnResult/*[out,retval]*/),

Параметры

[in] aObjType — тип объекта,

[out apSettings — настройки нового объекта по умолчанию,

[out apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.5.9 Метод IApcObjectWrap.addChildWithSettings

Метод позволяет к данному объекту добавить дочерний объект с указанными настройками.

VBScript

addChildWithSettings(propObj, newObj),

где:

propOb — настройки добавляемого объекта,

newObj — возвращаемое значение, добавленный объект.

В случае успешного выполнения метод возвращает *0*.

Возможные причины невыполнения:

- нет связи с сервером,
- нет прав на добавление объекта,
- некорректные значения свойств объекта.

Пример добавления объекта типа *Папка*:

```
If Apacs.getRootObject(ObjRoot) <> 0 Then
  Client.writeToLog "Не могу получить корневой объект"
Else
  If objRoot.getChildSettingsForAdd("TApcFolder", prop) <> 0 Then
    Client.writeToLog "Ошибка получения свойств объекта для добавления"
  Else
    prop.strName = "Новая папка"
    If objRoot.addChildWithSettings(prop, objFld) <> 0 Then
      Client.writeToLog "Не могу добавить папку к корню"
    Else
      msgbox "Папка добавлена к корню"
    End If
  End If
End If
```

C++

HRESULT **addChildWithSettings**(LPDISPATCH apSettings/*[in]*/,
LPDISPATCH* appObject/*[out]*/,
int* apnResult/*[out,retval]*/),

Параметры

[in] apSettings — настройки нового объекта,

[out] appObject — указатель на новый объект, поддерживающий интерфейс IApcObjectWrap,

[out] apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.5.10 Метод IApcObjectWrap.deleteObject

Метод позволяет удалить объект из конфигурации системы.

VBScript

deleteObject()

В случае успешного выполнения метод возвращает *0*.

Возможные причины невыполнения

- нет связи с сервером,
- нет прав на удаление объекта,
- объект не существует в системе.

Пример удаления объекта с псевдонимом obj4del

```
If Apacs.getObjectByAlias("obj4del", obj) <> 0 Then
  Client.writeToLog "Не могу найти объект с псевдонимом obj4del"
Else
  If obj.deleteObject() <> 0 Then
    Client.writeToLog "Ошибка удаления объекта"
  Else
    msgbox "Объект упешно удален"
  End If
End If
```

C++

HRESULT **deleteObject**(int* apnResult/*[out,retval]*/),

Параметры

[out] apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.5.11 Метод IApcObjectWrap.getEventSettingsForRegister

Метод позволяет получить настройки по умолчанию для регистрации нового сообщения в системе.

VBScript

getEventSettingsForRegister(strEventType, propEvent),

где:

strEventType — тип регистрируемого сообщения,

propEvent — возвращаемое значение, настройки нового сообщения по умолчанию.

В случае успешного выполнения метод возвращает *0*.

Возможные причины невыполнения

- нет связи с сервером.

Пример регистрации сообщения *Доступ разрешен*

```
' Для выполнения скрипта необходимо иметь:
'  1. Считыватель с псевдонимом rdr
'  2. Карта с номером 1000 и псевдонимом crd
'  3. Владелец карты Иванов Петр Сергеевич с спевдонимом hld и
'      с выданной ему картой № 1000
If Apacs.getObjectByAlias("rdr", rdr) <> 0 Then
  Client.writeToLog "Не могу найти объект с псевдонимом rdr"
Else
  If Apacs.getObjectByAlias("crd", crd) <> 0 Then
    Client.writeToLog "Не могу найти объект с псевдонимом crd"
  Else
    If Apacs.getObjectByAlias("hld", hld) <> 0 Then
      Client.writeToLog "Не могу найти объект с псевдонимом hld"
    Else
      If rdr.getEventSettingsForRegister("TApcCardHolderAccess_Granted", prop)
     <> 0 Then
      Client.writeToLog "Ошибка получения настроек сообщения для регистрации"
      Else
        prop.isOffLineEvent = false
        prop.dwCardNumber = 1000
        set prop.SysAddrCard = crd
        prop.strHolderName = "Иванов П. С."
        set prop.SysAddrHolder = hld
        If rdr.registerEventWithSettings(prop) <> 0 Then
          Client.writeToLog "Ошибка регистрации сообщения"
        Else
          msgbox "Новое сообщение зарегестрировано"
        End If
      End If
    End If
  End If
End If
```

C++

HRESULT **getEventSettingsForRegister**(BSTR aEventType/*[in]*/,
LPDISPATCH* appSettings/*[out]*/,
int* apnResult/*[out,retval]*/),

Параметры

[in] aEventType — тип регистрируемого сообщения,

[out] appSettings — настройки нового сообщения по умолчанию,

[out] apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.5.12 Метод IApcObjectWrap.registerEventWithSettings

Метод позволяет зарегистрировать в системе новое сообщение от данного объекта.

VBScript

registerEventWithSettings(propEvent),
где propEvent — настройки регистрируемого сообщения.
В случае успешного выполнения метод возвращает *0*.
Возможные причины невыполнения:

- нет связи с сервером.

 Пример регистрации сообщения *Доступ разрешен*

```
' Для выполнения скрипта необходимо иметь:
'  1. Считыватель с псевдонимом rdr
'  2. Карта с номером 1000 и псевдонимом crd
'  3. Владелец карты Иванов Петр Сергеевич с спевдонимом hld и
'     с выданной ему картой № 1000
If Apacs.getObjectByAlias("rdr", rdr) <> 0 Then
  Client.writeToLog "Не могу найти объект с псевдонимом rdr"
Else
  If Apacs.getObjectByAlias("crd", crd) <> 0 Then
    Client.writeToLog "Не могу найти объект с псевдонимом crd"
  Else
    If Apacs.getObjectByAlias("hld", hld) <> 0 Then
      Client.writeToLog "Не могу найти объект с псевдонимом hld"
    Else
      If rdr.getEventSettingsForRegister("TApcCardHolderAccess_Granted", prop)
     <> 0 Then
      Client.writeToLog "Ошибка получения настроек сообщения для регистрации"
      Else
        prop.isOffLineEvent = false
        prop.dwCardNumber = 1000
        set prop.SysAddrCard = crd
        prop.strHolderName = "Иванов П. С."
        set prop.SysAddrHolder = hld
        If rdr.registerEventWithSettings(prop) <> 0 Then
          Client.writeToLog "Ошибка регистрации сообщения"
        Else
          msgbox "Новое сообщение зарегестрированно"
        End If
      End If
    End If
  End If
End If
```

C++

HRESULT **registerEventWithSettings**(LPDISPATCH appSettings/*[in]*/,
int* apnResult/*[out,retval]*/),

Параметры

[in] appSettings — настройки нового сообщения,

[out] apnResult — успешное выполнение — *0*, иначе — код ошибки.

Возвращает значение HRESULT.

3.2.5.13 Метод IApcObjectWrap.<команда>

Метод предназначен для выполнения команд управления, поддерживаемых объектом.

VBScript

<команда> (параметр1, ... параметрN),
где <команда> — идентификатор команды,
параметр1, ... параметрN — параметры выполнения команды.
В случае успешного выполнения метод возвращает *0*.
Возможные причины невыполнения:

- нет связи с сервером;
- объект удален;
- нет возможности выполнить команду;
- нет прав на выполнение команд этого объекта (о правах см. п. «Apa: Глава 2 Права и аудит»).

Пример выполнения команды ***Закрыть*** на объекте с псевдонимом reader:

```
If Apacs.getObjectByAlias("reader", Obj) <> 0 Then
  Client.writeToLog "Не могу найти объект с псевдонимом reader"
Else
  If Obj.cmdLock() <> 0 Then
    Client.writeToLog "Ошибка при выполнении команды Закрыть"
  Else
    msgbox "Команда Закрыть выполнена"
  End If
End If
```

3.2.6 Интерфейс IApcVideoLinkerWrap

Интерфейс объекта *VideoLinker*. Предоставляет метод linkRec.

3.2.6.1 Метод IApcVideoLinkerWrap.linkRec

Метод позволяет инициировать запись на видеокамере и связать видеофрагмент с поступившим сообщением. В момент записи видеофрагмента в системе регистрируется дочернее сообщение типа *Видеофрагмент связан с сообщением*.

VBScript

linkRec (Event, CamObj, dwTime),
где:
Event — сообщение, для которого была инициирована запись видео,
CamObj — камера, производящая запись видеофрагмента,
dwTime — продолжительность записи видеофрагмента в секундах.
Возвращаемое значение
В случае успешного выполнения метод возвращает *0*.
Возможные причины невыполнения

- нет связи с сервером,

- нет прав на просмотр объекта *Камера*.

Пример записи видеофрагмента

```
Sub Apacs_Event (evt)
  If Apacs.getObjectByAlias( "Camera", obj ) = 0 Then
    ' запись видеофрагмента на сообщение evt с помощью камеры obj продолжительностью 5 секунд
    VideoLinker.linkRec evt, obj, 5
  End If
End Sub
```

3.2.7 Интерфейс IApcEQUALObjFilter

Интерфейс IApcEQUALObjFilter, предназначен для создания фильтра, проверяющего эквивалентность.

Имеет следующие свойства.

- strName — свойство типа string, строка, определяющая свойства объекта или события в APACS 3000,
- Value — свойство типа variant, значение свойства, имя которого определено в strName.

Пример использования интерфейса IApcEQUALObjFilter смотрите в описании метода getObjectsByFilter (см. п. «3.2.1.4 Метод getObjectsByFilter»).

3.2.8 Интерфейс IApcANDObjFilter

Интерфейс IApcANDObjFilter, предназначен для объединения фильтров по логическому *И*.

Использует следующие методы:

- addCondition,
- getConditions.

Пример использования интерфейса IApcANDObjFilter смотрите в описании метода getObjectsByFilter (см. п. «3.2.1.4 Метод getObjectsByFilter»).

3.2.8.1 Метод IApcANDObjFilter.addCondition

Метод addCondition позволяет добавить фильтр.

VBScript

```
addCondition (Condition)
```

где Condition — экземпляр объекта, реализующего интерфейс IApcANDObjFilter или IApcEQUALObjFilter.

C++

```
HRESULT addCondition(LPUNKNOWN apCondition/*[in]*/)
```

Параметры:

[in] apCondition — экземпляр объекта, реализующего интерфейс IApcANDObjFilter или IApcEQUALObjFilter.

Возвращает значение HRESULT.

3.2.8.2 Метод IApcANDObjFilter.getConditions

Метод позволяет получить список фильтров, которые объединяются по логическому *И*.

VBScript

getConditions

В случае успешного выполнения метод возвращает список фильтров, объединенных по логическому условию.

C++

HRESULT **getConditions**(LPSAFEARRAY* apConditions/*[out,retval]*/)

Параметры:

[out,retval] apConditions — возвращаемое значение, список фильтров, объединенных по логическому условию.

Возвращает значение HRESULT.

4 Клиентский модуль *HTML обозреватель*

Клиентский модуль ***HTML обозреватель*** позволяет создавать пользовательский интерфейс на основе загруженных HTML документов, объектная модель которых расширена объектами APACS 3000.

Объектная модель APACS 3000 доступна из объекта DHTML window.external:

- *Apacs* — window.external.Apacs,
- *Log* — window.external.Log,
- *Client* — window.external.Client,
- *Script* — window.external.Script,
- *VideoLinker* — window.external.VideoLinker.

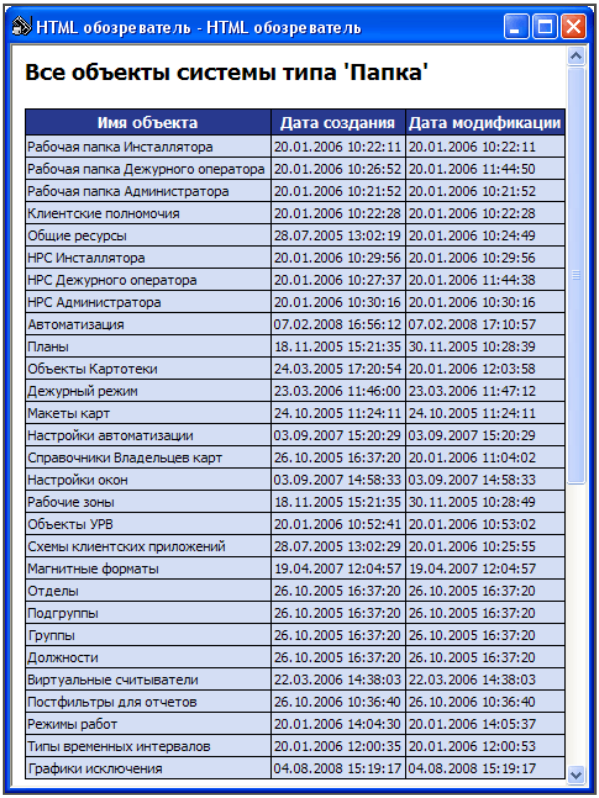
Таким образом, администратор комплекса может написать собственные HTML документы, используя весь доступный из макросов и реакций функционал. С помощью HTML документов могут быть реализованы такие функции, как:

- отчет об объектах и сообщениях системы,
- пропуск людей, предъявивший карту на считывателе, по решению оператора,
- интерактивное управление оборудованием и др.

В результате комплекс легко настраивается под требования конкретной системы управления и контроля доступом.

Обратите внимание: для корректной работы модуля ***HTML обозреватель*** в настройках Вашего MS Internet Explorer Browser требуется разрешить выполнение сценариев (скриптов).

Для загрузки в систему HTML документов используются объекты типа *HTML обозреватель*.



Имя объекта	Дата создания	Дата модификации
Рабочая папка Инсталлятора	20.01.2006 10:22:11	20.01.2006 10:22:11
Рабочая папка Дежурного оператора	20.01.2006 10:26:52	20.01.2006 11:44:50
Рабочая папка Администратора	20.01.2006 10:21:52	20.01.2006 10:21:52
Клиентские полномочия	20.01.2006 10:22:28	20.01.2006 10:22:28
Общие ресурсы	28.07.2005 13:02:19	20.01.2006 10:24:49
НРС Инсталлятора	20.01.2006 10:29:56	20.01.2006 10:29:56
НРС Дежурного оператора	20.01.2006 10:27:37	20.01.2006 11:44:38
НРС Администратора	20.01.2006 10:30:16	20.01.2006 10:30:16
Автоматизация	07.02.2008 16:56:12	07.02.2008 17:10:57
Планы	18.11.2005 15:21:35	30.11.2005 10:28:39
Объекты Картотеки	24.03.2005 17:20:54	20.01.2006 12:03:58
Дежурный режим	23.03.2006 11:46:00	23.03.2006 11:47:12
Макеты карт	24.10.2005 11:24:11	24.10.2005 11:24:11
Настройки автоматизации	03.09.2007 15:20:29	03.09.2007 15:20:29
Справочники Владельцев карт	26.10.2005 16:37:20	20.01.2006 11:04:02
Настройки окон	03.09.2007 14:58:33	03.09.2007 14:58:33
Рабочие зоны	18.11.2005 15:21:35	30.11.2005 10:28:49
Объекты УРВ	20.01.2006 10:52:41	20.01.2006 10:53:02
Схемы клиентских приложений	28.07.2005 13:02:29	20.01.2006 10:25:55
Магнитные форматы	19.04.2007 12:04:57	19.04.2007 12:04:57
Отделы	26.10.2005 16:37:20	26.10.2005 16:37:20
Подгруппы	26.10.2005 16:37:20	26.10.2005 16:37:20
Группы	26.10.2005 16:37:20	26.10.2005 16:37:20
Должности	26.10.2005 16:37:20	26.10.2005 16:37:20
Виртуальные считыватели	22.03.2006 14:38:03	22.03.2006 14:38:03
Постфильтры для отчетов	26.10.2006 10:36:40	26.10.2006 10:36:40
Режимы работ	20.01.2006 14:04:30	20.01.2006 14:05:37
Типы временных интервалов	20.01.2006 12:00:35	20.01.2006 12:00:53
Графики исключения	04.08.2008 15:19:17	04.08.2008 15:19:17

Рисунок Окно HTML документа с отчетом обо всех объектах типа *Папка*

Конфигурирование объекта HTML обозреватель

Конфигурирование объекта *HTML обозреватель* осуществляется в окне ***Проводник***:

- В окне ***Проводник*** добавьте объект типа *HTML обозреватель* к объекту типа *Папка*. Откроется окно ***HTML обозреватель – Свойства***. Укажите название объекта и нажмите кнопку **OK**. Объект появится в дереве объектов окна ***Проводник***.
- Сконфигурируйте объект *HTML обозреватель*. Для этого выполните на объекте команду ***Редактировать***. Откроется диалоговое окно ***HTML обозреватель***, где на вкладке **«Основные»** в поле **Адрес страницы** требуется указать путь к локальному HTML документу или URL HTML страницы.

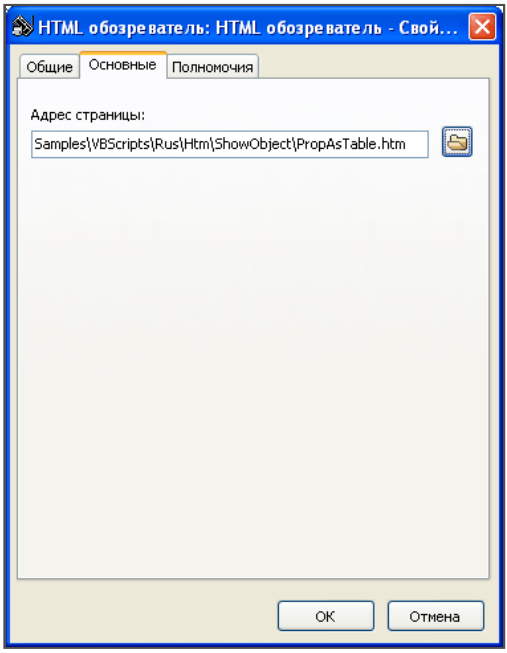


Рисунок Вкладка **«Основные»** окна редактирования свойств объекта ***HTML обозреватель***

Просмотр HTML документа

Чтобы открыть на экране окно HTML документа, выделите объект в окне ***Проводник*** и воспользуйтесь командой ***Показать***.

5 SDK (комплект для разработки программного обеспечения)

SDK (комплект для разработки программного обеспечения) представляет собой библиотеку, с помощью которой сторонние разработчики могут вызывать функции серверных объектов ПК APACS 3000 и, таким образом, осуществлять интеграцию с комплексом или же разрабатывать собственные клиентские приложения. Если при помощи подсистемы автоматизации создаются скрипты, которые исполняются внутри приложений APACS 3000, то SDK предоставляет возможность вызывать функции и прослушивать сообщения APACS 3000 из внешних программ.

SDK предоставляет следующие возможности:

- авторизация на сервере APACS 3000,
- доступ к любым объектам конфигурации,
- получение текущих настроек объектов,
- редактирование настроек объектов,
- создание новых объектов,

- удаление объектов,
- получение сообщений от объектов в режиме online,
- получение уведомлений об удалении, изменении, добавлении объектов в режиме online,
- получение сохраненных сообщений из базы данных по фильтру,
- регистрация сообщений от объектов,
- получение уведомления в случае потери связи с сервером.

Функции SDK реализованы через COM–объект, находящийся в библиотеке ApcSrvConnection.dll. Регистрация COM–объекта происходит автоматически при инсталляции комплекса. Библиотека предоставляет все необходимые COM–объекты для прямого вызова функций сервера.

Так как SDK построен на основе технологии COM, возможно использовать SDK APACS 3000 из любых сред программирования, которые поддерживают работу с COM (например, MS Visual C++, MS Visual C#, C++ Builder, Delphi и др.). Также работу с SDK возможно реализовать через стандартный *.vbs файл, который может быть разработан в любом редакторе и запущен из командной строки операционной системы.

В составе комплекса поставляются примеры на языке C++ Builder, VBScript и Visual C# для платформы .Net Framework 1.1 (находятся в папке Samples\SDK корневого каталога APACS 3000). Примеры демонстрируют все основные возможности SDK.

SDK APACS 3000 имеет следующие ограничения:

- SDK предоставляет возможность вызова серверных команд комплекса, но не позволяет встраивать вызов какого–либо дополнительного функционала «внутрь» уже существующих клиентских приложений (например, нельзя дописать функционал модулей ***Редактор планов, Просмотр планов***, ***Проходная*** и т.д.).
- Для получения сообщений из базы данных доступны только запросы с фильтрацией по дате начала, дате окончания и типу событий.
- Получение списка объектов типа *Владелец карты*, *Группа доступа, Идентификатор* возможно только в несортированном списке, без возможности задания фильтрации (или же по псевдонимам, указанным для каждого объекта).
- Через SDK недоступен объект «Метаданные» (информация о типах, полях и т.д.).
- Через SDK нельзя получить переводы для информации в «Метаданных» (например, стандартные названия типов считывателей, входов и т.д.).
- При помощи SDK нельзя читать/редактировать настройки полномочий для объектов конфигурации.
- При помощи SDK нельзя разработать серверный драйвер оборудования, работающий в составе сервера APACS 3000 (так называемый DDK).

Предопределенные классы SDK

В составе SDK предопределены следующие класс:

- TApcConnection – экземпляры этого класса позволяют установить соединение с сервером APACS 3000. Класс реализует интерфейс IApcConnection.
- TApcSession – экземпляры этого класса предоставляют сеанс связи с сервером APACS 3000 и реализуют интерфейс IApcSession.

Принципы работы

Для того чтобы начать работу с сервером APACS 3000, необходимо установить с ним соединение. Для этого предназначен класс TApcConnection. Класс TApcConnection реализует метод createSession, позволяющий Вам создать новую сессию соединения с с сервером APACS 3000.

Класс TApcSession предоставляет сеанс связи с сервером APACS 3000. Из него доступен объект с интерфейсом IApcServerWrap, который отображает объектную модель сервера APACS 3000.

Для получения подробной информации об SDK APACS 3000 смотри описание интерфейсов IApcConnection, IApcSession, IApcServerWrap и IApcObjectWrap.

5.1 Интерфейс IApcConnection

Позволяет установить новое соединение с сервером APACS 3000. Реализует метод createSession.

5.1.1 Метод IApcConnection.createSession

Метод создает новое соединение с сервером APACS 3000.

C++

```
HRESULT createSession(BSTR astrLoginName/*[in]*/,
                      BSTR astrPasswd/*[in]*/,
                      IApcSession** apSession/*[out]*/,
                      int* apnResult/*[out,retval]*/
```

Параметры

[in] astrLoginName — имя оператора в системе APACS 3000,
[in] astrPasswd — пароль оператора в системе APACS 3000,
[out] apSession — сессия соединения с сервером APACS 3000 (IApcSession),
[out] apnResult — *0* — если соединение с сервером установлено, иначе — код ошибки.

Возвращает значение HRESULT.

Пример установления соединения с сервером APACS 3000.

```
// Инициализируем COM
 CoInitializeEx(NULL, COINIT_APARTMENTTHREADED);
 // Создаем объект, создающий новое соединение
 CComPtr<IApcConnection> spCon;
 HRESULT hRes = spCon.CoCreateInstance(CLSID_TApcConnection,
   NULL,
   CLSCTX_INPROC);
```

```
if(hRes == S_OK)
{
  // Создаем новое соединение
  CComPtr<IApcSession> spSes;
  int nResult = -1;
  CComBSTR login(L"Inst");
  CComBSTR pswrd(L"");
  hRes = spCon->createSession(login,
    pswrd,
    &spSes,
    &nResult);
  if(hRes == S_OK && nResult == 0)
  {
    // Получаем доступ к серверу Apacs 3000
    CComPtr<IApcServerWrap> spApacs;
    CComPtr<IDispatch> spDispApacs;
    hRes = spSes->getServer(&spDispApacs);
    if(hRes == S_OK)
    {
      hRes = spDispApacs.QueryInterface(&spApacs);
      if(hRes == S_OK)
      {
        // Работаем с сервером Apacs 3000
      }
    }

    // Закрываем соединение
    spSes->close();
  }
}
```

5.2 Интерфейс IApcSession

Сессия соединения с сервером APACS 3000.

Для доступа к объектной модели сервера APACS 3000 используйте метод getServer.

Для завершения сеанса работы с сервером APACS 3000 — метод close.

Для получения уведомления об обрыве соединения с сервером APACS 3000 — метод set_onDisconnect.

Смотрите пример установления соединения с сервером APACS 3000 (п. «5.1.1 Метод createSession»).

5.2.1 Метод IApcSession.getServer

Метод позволяет получить объект, поддерживающий интерфейс IApcServerWrap.

C++

HRESULT **getServer**(LPDISPATCH* appServer/*[out,retval]*/),

Параметры

[out] appServer — указатель на объект, поддерживающий интерфейс IApcServerWrap.

Возвращает значение HRESULT.

5.2.2 Метод IApcSession.close

Метод завершает сеанс связи с сервером APACS 3000.

C++

HRESULT **close**(void)
Возвращает значение HRESULT.

5.2.3 Метод IApcSession.set_onDisconnect

Метод позволяет установить обработчик на обрыв с соединения с сервером APACS 3000. При обрыве соединения будет вызван метод Invoke с DispID=0,без параметров.

C++

HRESULT STDMETHODCALLTYPE **set_onDisconnect**(LPDISPATCH apDisHandler/*[in]*/),

Параметры

[in] apDisHandler — указатель на объект, который будет принимать уведомления о разрыве соединения с сервером.

Возвращает значение HRESULT.