ООО «Компания «АвтоДилер»

Плагины (модули расширения)

РУКОВОДСТВО РАЗРАБОТЧИКА

2014

Плагины

Плагины (*модули расширения*) предназначены для добавления функционала в программу путем установки модулей, которые позволяют выполнять различные задачи, не затрагивая при этом логику основной программы.

При помощи плагинов, в программе имеется возможность добавлять кнопки на панель инструментов «Плагины» на вкладке «Параметры» главного окна программы, создавать дополнительные пункты контекстного меню большинства окон со списками документов, а так же пункты в контекстном меню «Плагины», которое является выпадающим для кнопки «Менеджер плагинов».

При помощи менеджера плагинов, в программе существует возможность включать и отключать не используемые плагины, тем самым выбирая необходимый набор плагинов, в зависимости от потребностей пользователя.

Плагинами можно реализовать как простые, так и достаточно сложные по объему задачи, в связи с чем, введено понятие «Статуса» плагина. Плагины имеют три статуса: бесплатный, условно-бесплатный и платный. Если плагин имеет статус условно-бесплатный, то для обеспечения его работы, система «АвтоДилер» должна иметь электронный USB-ключ защиты и наличие хотя бы одного активного модуля. В случае, если плагин платный, то необходимо наличие лицензии на плагин.

Ещё одним неотъемлемым условием, обеспечивающим работу плагинов, является наличие у каждого плагина электронной подписи. Электронная подпись должна быть у каждого плагина, независимо от его статуса.

Установка плагинов

Специальной процедуры установки плагинов не требуется, достаточно скопировать модуль плагина в корень папки «Plugins», которая расположена в корне папки установки системы «АвтоДилер». Однако, папка «Plugins» может отсутствовать в силу того, что поддержка плагинов появилась в 7.0 версии системы «АвтоДилер», тогда ее необходимо создать вручную. Помимо копирования модуля плагина, необходимо в корне папки «Plugins» создать одноименную с файлом плагина папку для размещения в ней различного рода файлов, обеспечивающих работу плагина. В частности, такими файлами являются файл цифровой подписи и файл лицензии (*при необходимости*).

По умолчанию, путь до папки «Plugins»: “C:\Program Files\AutoDealer\AutoDealer\Plugins\”

Например, если модуль плагина имеет имя «ClientInfo.dll», то в корне папки «Plugins» необходимо создать папку «ClientInfo», в которую скопировать файлы, обеспечивающие работу плагинов.

Файлы лицензий и электронных подписей необходимо получить в Компании «АвтоДилер». Детали вопросов, связанных с получением этих файлов, обсуждаются при обращении в Компанию. Нужно лишь отметить, что для разработки плагина необходимо так же обратиться в Компанию «АвтоДилер» для получения файла подписи для разработки, который выдается на определенный срок и по его истечении становится недействительным. Полученный файл подписи позволит полнофункционально тестировать работу плагина в программе.

Описание интерфейса

Каждый плагин должен реализовать следующий интерфейс:

```
IPluginObject = interface
    ['{24E6DD8B-39F9-42B3-9F31-AE381FFEE697}']
    function GetDescription: PWideChar; stdcall;
    function GetVendor: PWideChar; stdcall;
    function GetVersion: PWideChar; stdcall;
    function GetPluginType: LongWord; stdcall;
    procedure ShowAbout; stdcall;
    function GetVerb: PWideChar; stdcall;
    function GetVerbWindowClass: PWideChar; stdcall;
    function ExecuteVerb(Params: PWideChar): PWideChar; stdcall;
    function GetVerbState: TPluginState; stdcall;
    function ErrorMessage: PWideChar; stdcall;
    function GetVerbStatus: Integer; stdcall;
    function GetDataFolder: PWideChar; stdcall;
  end;
```

Для описания функций и методов плагина, объявим класс, реализующий интерфейс плагина.

```
TPluginClass = class(TInterfacedObject, IPluginObject)
  private
    function GetDescription: PWideChar; stdcall;
    function GetVendor: PWideChar; stdcall;
    function GetVersion: PWideChar; stdcall;
    function GetPluginType: LongWord; stdcall;
    procedure ShowAbout; stdcall;
    function GetVerb: PWideChar; stdcall;
    function GetVerbWindowClass: PWideChar; stdcall;
    function ExecuteVerb(Params: PWideChar): PWideChar; stdcall;
    function GetVerbState: TPluginState; stdcall;
    function ErrorMessage: PWideChar; stdcall;
    function GetDataFolder: PWideChar; stdcall;
    function GetVerbStatus: Integer; stdcall;
  end;
```

Функции и процедуры

❖ GetDescription

Функция возвращает в программу описание плагина в виде строки.

Пример реализации:

```
function TPluginClass.GetDescription: PWideChar;
begin
  Result := 'Описание плагина';
end;
```

❖ GetVendor

Функция возвращает в программу название автора, написавшего плагин в виде строки.

Пример реализации:

```
function TPluginClass.GetVendor: PWideChar;
begin
  Result := 'ООО «Компания «АвтоДилер»';
end;
```

❖ GetVersion

Функция возвращает в программу версию плагина в виде строки.

Пример реализации:

```
function TPluginClass.GetVersion: PWideChar;
begin
  Result := '1.0.0.0';
end;
```

❖ GetPluginType

Функция возвращает в программу тип плагина и может принимать одно или несколько значений:

- AD_PLUGIN_MENUITEM – будет создан пункт в контекстном меню для вызова плагина.
- AD_PLUGIN_MDIFORM – в программе будет создано MDI-child окно-контейнер, в котором можно поместить разработанное окно плагина.
- AD_PLUGIN_MAINFORMBUTTON – будет создана кнопка на панели инструментов «Плагины» для вызова плагина в главной форме программы на вкладке «Параметры».

Пример реализации:

```
function TPluginClass.GetPluginType: LongWord;
begin
  Result := AD_PLUGIN_MENUITEM or AD_PLUGIN_MAINFORMBUTTON;
end;
```

❖ ShowAbout

Процедура (метод) вызывается из программ. Например, из «Менеджера плагинов», может быть использована для реализации окна «О плагине» или показа диалогового сообщения, как в примере ниже.

Пример реализации:

```
procedure TPluginClass.ShowAbout;
begin
  MessageDlg('ООО «Компания «АвтоДилер»' + sLineBreak +
    'Информация о плагине.', mtInformation, [mbOK], 0);
end;
```

❖ GetVerb

Функция возвращает в программу название плагина в виде строки. Если результатом функции будет пустая строка или nil, то плагин будет отвергнут при загрузке в программу.

Пример реализации:

```
function TPluginClass.GetVerb: PWideChar;
begin
  Result := 'Название плагина';
end;
```

❖ GetVerbWindowClass

Функция возвращает в программу список классов окон программы (*см. раздел «Классы окон программы»*), у которых в контекстном меню основного списка будет создан пункт меню для вызова плагина (в случае если тип плагина включает в себе константу AD_PLUGIN_ MENUITEM). В остальных случаях возвращается пустая строка или nil. Список возвращается в виде строки, где имена классов окон разделены точкой с запятой.

Если среди списка классов будет класс главной формы TfmMain, то будет так же создан пункт в выпадающем меню «Менеджер плагинов – Плагины» на вкладке «Параметры» главного окна программы.

Пример реализации:

```
function TPluginClass.GetVerbWindowClass: PWideChar;
begin
  Result := 'TfmShopDocument;TfmServiceDocument;TfmDirClient';
end;
```

❖ ExecuteVerb

Функция запускает плагин на исполнение. В ней, как правило, реализуется создание и показ главного окна плагина. Во входном параметре Params плагин принимает от программы переданные ей параметры (*см. раздел «Параметры окон программы»*) в виде строки разделенной сочетанием символов возврата каретки и перевода строки. Каждый параметр имеет конструкцию вида: <имя параметра>=<значение>.
Если тип плагина включает в себя константу AD_PLUGIN_MDIFORM, то результатом функции должен быть указатель на дескриптор главного окна плагина. В остальных случаях в качестве результата функции возвращается пустая строка.

Пример реализации:
Вы можете посмотреть реализацию этой функции в примерах плагинов, входящих в поставку программы.

❖ GetVerbState

Функция возвращает в программу набор состояний плагина:

- psEnabled – означает, что плагин доступен.

На текущий момент, в программе обрабатывается только это состояние, если результатом функции GetVerbState будет пустой набор [], то это будет означать, что плагин недоступен.

Пример реализации:

```
function TPluginClass. GetVerbState: TPluginState;
begin
  Result := [psEnabled];
end;
```

❖ ErrorMessage

Функция возвращает в программу сообщение в виде строки, которое, например, может указывать причину недоступности плагина.

Пример реализации:

```
function TPluginClass.ErrorMessage: PWideChar;
begin
  Result := 'Сообщение о причине недоступности плагина';
end;
```

❖ GetDataFolder

Функция возвращает в программу полный путь до папки с данными плагина. Папка с данными плагина представляет собой одноименную с файлом плагина папку, в которой располагаются различные файлы, необходимые для работы плагина, в частности, там должен располагаться файл подписи и файл лицензии (*в случае если плагин платный*).

Пример реализации:

```
function TPluginClass.GetDataFolder: PWideChar;
var
  PluginFile, PluginPath: string;
begin
  PluginPath := GetModuleName(HInstance);
  PluginFile := ExtractFileName(PluginPath);
  PluginPath := ExtractFilePath(PluginPath) +
                ChangeFileExt(PluginFile, PathDelim);

  Result := PWideChar(PluginPath);
end;
```

❖ GetVerbStatus

Функция возвращает в программу статус плагина. Существует четыре статуса плагина. Каждому статусу соответствует константа:

- plsFree – плагин бесплатный;
- plsShare – плагин условно-бесплатный (*необходимо наличие USB-ключа защиты программы и хотя бы одного активного модуля*);
- plsCommerce – платный плагин. Для работы плагина необходимо наличие файла лицензии (*для плагина*) в папке с данными;
- plsDevelopment – указывается в случае, когда плагин находится в стадии разработки (*чтобы обеспечить тестирование плагина в программе*). При этом необходимо иметь файл подписи для разработки, который выдается на определенный срок, после чего становится недействительным.

Примечание: файл лицензии и файл подписи, в том числе на разработку, запрашиваются в Компании «АвтоДилер».

Пример реализации:

```
function TPluginClass.GetVerbStatus: Integer;
begin
  Result := Integer( plsFree );
end;
```

Экспортируемые функции

Помимо реализации интерфейса, каждый плагин должен реализовать следующие экспортируемые функции и процедуры:

❖ GetAdPluginObject

Функция используется программой для получения экземпляра основного класса плагина, который является наследником класса TInterfacedObject и реализует интерфейс IPluginObject.
В примере реализации ниже переменная PluginObject типа IPluginObject содержит экземпляр основного класса плагина.

Пример реализации:

```
function GetAdPluginObject: IPluginObject; stdcall;
begin
  Result := PluginObject;
end;
```

❖ LoadDoneAdPlugin

Процедура вызывается программой по окончанию процесса загрузки плагина. При вызове процедуры, программа передает плагину дескрипторы приложения, главного окна программы и дескриптор соединения с базой данных. Данная процедура в плагине может быть использована, например, для определения состояния доступности плагина или для создания модуля данными, как в примере приведенном ниже.

Пример реализации:

```
procedure LoadDoneAdPlugin(APPHandle, MainWindow: THandle;
  DBHandle: TISC_DB_HANDLE); stdcall;
begin
  ApplicationHandle := APPHandle;
  MainWindowHandle := MainWindow;
  { Создание модуля данных }
  dmDataModule := TdmDataModule.Create(Application);
  dmDataModule.dbMain.Handle := DBHandle;
  dmDataModule.dbMain.Connected := True;
end;
```

❖ UnloadAdPlugin

Процедура вызывается программой в момент завершения ее работы, когда происходит выгрузка плагинов, загруженных в программу. Данная процедура в плагине может быть использована для освобождения созданных при загрузке или при работе плагина объектов.

Пример реализации:

```
procedure UnloadAdPlugin; stdcall;
begin
  PluginObject := nil;
end;
```

Классы окон программы

В таблице №1 приведен перечень окон в программе, которые поддерживают работу с плагинами.

Таблица №1 – перечень окон

Имя класса окна	Описание
TfmMain	Главное окно программы
TfmDirClient	Справочник по клиентам
TfmDirEmployee	Справочник по сотрудникам
TfmInsuranceOsago	Контракты ОСАГО
TfmInsuranceDocument	Бланки строгой отчетности в страховании
TfmDiagnosticCard	Диагностические карты
TfmServiceCalculation	Калькуляции
TfmServiceDocument	Заказ-наряды
TfmDiagnosticDocument	Бланки талонов техосмотра
TfmShopWaybill_out	Расходные накладные (магазин)
TfmShopInList	Приходные накладные (магазин)
TfmShopDocument	Все документы
TfmPlanning	Планирование (окно планировщика)
TfmPlanningDocument	Планирование (список документов)
TfmOrderProviderDocument	Заказы поставщикам
TfmOrderDocument	Заказы клиентов
TfmMotorShowBlank	Бланки строгой отчетности: транзитные номера
TfmMotorShowDocumentIn	Приходные накладные (салон)
TfmMotorShowDocumentOut	Расходные накладные (салон)

Параметры окон программы

Все параметры передаются из окон программы в плагин в виде строки разделенной сочетанием символов возврата каретки и перевода строки.
Каждый параметр имеет конструкцию вида: <имя параметра>=<значение>.
В таблице №2 приведен перечень параметров, которые передаются из окон программы при запуске плагинов. Описание параметров приведено в таблице №3.

Таблица №2 – Параметры, передаваемые окнами для плагинов

Имя класса окна	Имя параметра
TfmDirClient	CLIENT
TfmDirEmployee	EMPLOYEE
TfmInsuranceOsago	OSAGO_CONTRACT
	OSAGO_POLICY
	MODEL_LINK
	ORGANIZATION
	AGENT
	INSURER
	DOCUMENT_REGISTRY

TfmInsuranceDocument	DOCUMENT_REGISTRY
TfmDiagnosticCard	DIAGNOSTIC_CARD
	DOCUMENT_REGISTRY
	ORGANIZATION
	DIAGNOSTIC_DOCUMENT
TfmServiceCalculation	CLIENT
	MODEL_LINK
	SERVICE_CALCULATION_ITEM
TfmServiceDocument	CLIENT
	DOCUMENT_REGISTRY
	ORGANIZATION
	DOCUMENT_TYPE
	DOCUMENT_OUT
TfmDiagnosticDocument	DOCUMENT_REGISTRY
	ORGANIZATION
	DIAGNOSTIC_DOCUMENT
TfmShopWaybill_out	CLIENT
	DOCUMENT_REGISTRY
	DOCUMENT_TYPE
	ORGANIZATION
	DOCUMENT_OUT
	MANAGER
TfmShopInList	ORGANIZATION
	DOCUMENT_REGISTRY
	DOCUMENT_IN
	PROVIDER
TfmShopDocument	CLIENT
	DOCUMENT_REGISTRY
	DOCUMENT_TYPE
	DOCUMENT_OUT_HEADER
TfmPlanning	DOCUMENT_REGISTRY
TfmPlanningDocument	DOCUMENT_REGISTRY
TfmOrderProviderDocument	ORGANIZATION
	DOCUMENT_REGISTRY
	PROVIDER
	ORDER_PROVIDER_DOCUMENT
TfmOrderDocument	CLIENT
	DOCUMENT_REGISTRY
	DOCUMENT_TYPE
	ORGANIZATION
	MANAGER
	DOCUMENT_OUT
TfmMotorShowBlank	DOCUMENT_REGISTRY
	ORGANIZATION
TfmMotorShowDocumentIn	ORGANIZATION
	DOCUMENT_REGISTRY
	PROVIDER
	MOTORSHOW_DOCUMENT_IN
TfmMotorShowDocumentOut	ORGANIZATION
	DOCUMENT_REGISTRY
	MOTORSHOW_DOCUMENT_OUT

Таблица №3 – Описание параметров, возвращаемых окнами в программы

Имя параметра	Описание
CLIENT	Идентификатор клиента. Ссылка на первичный ключ CLIENT_ID в таблице CLIENT
EMPLOYEE	Идентификатор сотрудника. Ссылка на первичный ключ EMPLOYEE_ID в таблице EMPLOYEE
OSAGO_CONTRACT	Идентификатор контракта ОСАГО. Ссылка на первичный ключ OSAGO_CONTRACT_ID в таблице OSAGO_CONTRACT
OSAGO_POLICY	Идентификатор полиса ОСАГО. Ссылка на первичный ключ OSAGO_POLICY_ID в таблице OSAGO_POLICY
MODEL_LINK	Идентификатор отношения «Автомобили» - «Владельцы» . Ссылка на первичный ключ MODEL_LINK_ID в таблице MODEL_LINK
ORGANIZATION	Идентификатор предприятия. Ссылка на первичный ключ ORGANIZATION_ID в таблице ORGANIZATION
AGENT	Идентификатор страхового агента. Ссылка на первичный ключ AGENT_ID в таблице AGENT
INSURER	Идентификатор страховщика. Ссылка на первичный ключ INSURER_ID в таблице INSURER
DOCUMENT_REGISTRY	Идентификатор реестрового номера документа. Ссылка на первичный ключ DOCUMENT_REGISTRY_ID в таблице DOCUMENT_REGISTRY
DIAGNOSTIC_CARD	Идентификатор диагностической карты. Ссылка на первичный ключ DIAGNOSTIC_CARD_ID в таблице DIAGNOSTIC_CARD
DIAGNOSTIC_DOCUMENT	Идентификатор диагностического бланка. Ссылка на первичный ключ DIAGNOSTIC_DOCUMENT_ID в таблице DIAGNOSTIC_DOCUMENT
SERVICE_CALCULATION_ITEM	Идентификатор калькуляции. Ссылка на первичный ключ SERVICE_CALCULATION_ITEM_ID в таблице SERVICE_CALCULATION_ITEM
DOCUMENT_TYPE	Идентификатор типа документа. Ссылка на первичный ключ DOCUMENT_TYPE_ID в таблице DOCUMENT_TYPE
DOCUMENT_OUT	Идентификатор документа (если передан параметр DOCUMENT_TYPE, то это позволяет понять ещё и тип документа). Ссылка на первичный ключ DOCUMENT_OUT_ID в таблице DOCUMENT_OUT
MANAGER	Идентификатор менеджера в документе. Ссылка на первичный ключ MANAGER_ID в таблице MANAGER
DOCUMENT_IN	Идентификатор приходной накладной (Магазин). Ссылка на первичный ключ DOCUMENT_IN_ID в таблице DOCUMENT_IN
PROVIDER	Идентификатор поставщика в документе. Ссылка на первичный ключ PROVIDER_ID в таблице PROVIDER
DOCUMENT_OUT_HEADER	Идентификатор «шапки» документа. Ссылка на первичный ключ DOCUMENT_OUT_HEADER_ID в таблице DOCUMENT_OUT_HEADER
ORDER_PROVIDER_DOCUMENT	Идентификатор заказа поставщику. Ссылка на первичный ключ ORDER_PROVIDER_DOCUMENT_ID в таблице ORDER_PROVIDER_DOCUMENT
MOTORSHOW_DOCUMENT_IN	Идентификатор приходной накладной (Салон). Ссылка на первичный ключ MOTORSHOW_DOCUMENT_IN_ID в таблице MOTORSHOW_DOCUMENT_IN
MOTORSHOW_DOCUMENT_OUT	Идентификатор расходной накладной (Салон). Ссылка на первичный ключ MOTORSHOW_DOCUMENT_OUT_ID в таблице MOTORSHOW_DOCUMENT_OUT

Техническая поддержка

Интернет: http://www.autodealer.ru/support

E-mail: support@autodealer.ru