

СРЕДСТВО РАЗРАБОТКИ АТЛАНТИС

Среда разработки Viper

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ



2018

АННОТАЦИЯ

Документ предназначен для программистов, разрабатывающих приложения с использованием средства разработки **Атлантис** (далее по тексту **Атлантис**). Данный документ содержит описание интегрированной среды разработки **Атлантис**-приложений **Viper**.

Среда разработки состоит из редактора текстов программ, средств управления проектом, интерактивного отладчика и других функциональных возможностей.

Издание 04.2018

Оглавление


1. Что нового.....	6
1.1. Издание 04.2018.....	6
1.2. Издание 11.2016.....	6
1.3. Издание 04.2015.....	6
1.4. Издание 03.2015.....	7
1.5. Издание 04.2014.....	9
1.6. Издание 02.2014.....	10
1.7. Издание 01.2014.....	11
1.8. Издание 05.2013.....	12
1.9. Издание 01.2013.....	13
1.10. Издание 06.2012.....	15
1.11. Издание 01.2012.....	17
2. Основы использования.....	20
2.1. Использование проекта.....	20
2.2. Встроенный компилятор.....	20
2.3. Интегрированный отладчик.....	21
2.4. Примеры использования.....	21
3. Интерфейс пользователя.....	22
3.1. Главное окно приложения.....	22
3.2. Меню верхнего уровня.....	24
3.2.1. Меню Файл.....	24
3.2.2. Меню Правка.....	27
3.2.2.1. Команда Специальная вставка.....	29
3.2.2.2. Меню Сворачивание кода.....	30
3.2.2.3. Команда Свернуть до N уровня.....	30
3.2.2.4. Меню Разворачивание кода.....	31
3.2.2.5. Меню Форматирование.....	31
3.2.2.6. Меню Изменение регистра.....	32
3.2.2.7. Меню Блок.....	32
3.2.2.8. Команда Сортировка.....	33
3.2.2.9. Меню Блок дополнительно.....	33
3.2.2.10. Команда Соединить строки.....	34
3.2.2.11. Команда Формат строки.....	35
3.2.2.12. Команда Орфография.....	35
3.2.3. Меню Поиск.....	36
3.2.3.1. Команда Найти.....	38
3.2.3.2. Команда Найти в файлах.....	39
3.2.3.3. Команда Заменить.....	41
3.2.3.4. Меню Навигация по коду.....	43
3.2.3.5. Команда Перейти.....	44
3.2.3.6. Меню Метка.....	44
3.2.4. Меню Вид.....	46
3.2.4.1. Меню Панели инструментов.....	47
3.2.4.2. Настройка инструментальных панелей и горячих клавиш.....	48
3.2.5. Меню Проект.....	50


3.2.6.	Меню Отладка	51
3.2.7.	Меню Скрипты	53
3.2.8.	Меню TortoiseSVN	53
3.2.9.	Меню Инструменты	54
3.2.10.	Меню Окно	54
3.2.11.	Меню Справка	55
3.3.	Инструментальные панели	55
3.4.	Инструментальные окна	57
3.4.1.	Управление окнами	57
3.4.2.	Окно менеджера проекта	58
3.4.3.	Окно структуры кода	61
3.4.4.	Окно Файловый проводник	63
3.4.5.	Окно меток	64
3.4.6.	Окно структуры ресурсов	65
3.4.7.	Окно дерева подключений	66
3.4.8.	Окно связей таблиц	67
3.4.9.	Окно вывода	68
3.4.10.	Окно сообщений компилятора	69
3.5.	Редактор	70
3.5.1.	Навигация	71
3.5.2.	Форматирование	73
3.5.3.	Сервис	73
3.5.4.	Подсказка кода	74
3.5.4.1.	Автодополнение	75
3.5.4.2.	Переход к описанию	77
3.5.4.3.	Вывод описания о текущей конструкции	78
3.5.4.4.	Подсказка параметров процедур и функций	78
3.5.5.	Функции рефакторинга	79
3.6.	Окно вкладок	79
3.7.	Статус-строка	80
4.	Проект	81
4.1.	Создание проекта	81
4.1.1.	Мастер настройки проекта	82
4.2.	Состав проекта	85
4.3.	Параметры проекта	86
4.3.1.	Окно Параметры проекта	87
4.3.1.1.	Вкладка Проект	87
4.3.1.2.	Вкладка Компилятор VIP	89
4.3.1.3.	Вкладка Отладчик VIP	100
4.3.2.	Макропеременные в параметрах проекта	101
5.	Среда Viper	103
5.1.	Исполняющая среда Viper	103
5.1.1.	Справочная система	103
5.1.2.	Диагностика исключительных ситуаций	104
5.2.	Настройка параметров среды	105
5.2.1.	Вкладка Общие	105
5.2.1.1.	Вкладка Общие - Просмотр	107
5.2.1.2.	Вкладка Общие - Ассоциации	108
5.2.1.3.	Вкладка Общие - Запуск	110
5.2.1.4.	Вкладка Общие - Скрипты	111
5.2.2.	Вкладка Редактор	112
5.2.2.1.	Вкладка Редактор - Цвет	114


5.2.2.2.	Вкладка Редактор - Автозамена.....	115
5.2.2.3.	Вкладка Редактор - Проверка правописания.....	115
5.2.2.4.	Вкладка Редактор - Параметры.....	116
5.2.2.5.	Вкладка Редактор - Служебное поле.....	118
5.2.2.6.	Вкладка Редактор - Сворачивание кода.....	118
5.2.2.7.	Вкладка Редактор - Подсказка кода.....	120
5.2.3.	Вкладка Синтаксис.....	121
5.2.3.1.	Синтаксические схемы.....	122
5.2.3.2.	Atlantis VIP.....	123
5.3.	Командная строка.....	125
5.4.	Использование регулярных выражений.....	127
6.	Компиляция программ.....	130
6.1.	Настройка компилятора.....	130
6.2.	Использование компилятора.....	130
7.	Отладка программ.....	132
7.1.	Настройка отладчика.....	132
7.2.	Процесс отладки.....	133
7.3.	Окно точек останова.....	135
7.4.	Окно стека.....	136
7.5.	Окно выражений и переменных.....	137
7.6.	Окно локальных переменных.....	138
7.7.	Окно интерфейсов.....	139
7.8.	Окно загруженных интерфейсов.....	140
7.9.	Окно таблиц в памяти.....	141
8.	Средства внешнего запуска и исполнения.....	143
8.1.	Инструменты.....	143
8.1.1.	Настройка инструмента.....	143
8.1.2.	Переменные в настройках инструмента.....	145
8.2.	Профили.....	147
8.2.1.	Настройка профиля.....	148
8.2.2.	Переменные в настройках профиля.....	149
8.3.	Интерпретатор скриптов.....	150
8.3.1.	Текст скрипта.....	150
8.3.2.	API редактора.....	151
8.3.2.1.	Объект "Главное окно".....	151
8.3.2.2.	Объект "Активный документ".....	152
8.3.2.3.	Объект "Проект".....	160
8.3.2.4.	Объект "Коллекция".....	161
8.3.2.5.	Объект "Блок".....	162
8.3.2.6.	Объект "Запись".....	163
8.3.3.	Пользовательские интерфейсы.....	164
8.3.4.	Примеры использования скриптов.....	165

1. Что нового


1.1. Издание 04.2018

 В параметрах компиляции для подключения к базе данных добавлена настройка **NTSecurity**. Она предназначена для идентификации пользователя в БД **Галактики ERP** средствами ОС Microsoft Windows.

 Начиная с дистрибутива **Viper 5.5.22.0** файлы **FReport3.dll** и **FREasyEdit.exe** поставляются с новыми именами **FReport.dll** и **FREditor.exe** в связи с их изменением в **Атлантисе 5.5.30**.


 Добавлена информация о возможности начать изучение процесса компиляции и отладки в **Viper** на примере готового проекта (**Sample.vpr**), который содержит несколько примеров кода на языке **VIP** и поставляется в составе дистрибутива интегрированной среды **Viper**.


1.2. Издание 11.2016


 Реализована возможность обмена с сервером аппаратного ключа по протоколу TCP, которая используется с версии компилятора **5.5.25.0**. Для данной возможности в настройках элементов сборки и мастере создания проекта **Viper** добавлены параметры подключения к серверу аппаратного ключа в зависимости от выбранного протокола: SMB-file или TCP.

1.3. Издание 04.2015


Механизм организации окон


 Для вкладок редактора добавлена всплывающая подсказка с полным именем открытого в ней файла, которая отображается по наведению указателя на данную вкладку.


 Доработан список инструментальных окон в окне выбора вкладок. В списке отображаются только пристыкованные окна к границам среды в текущем режиме рабочего стола **Viper**, при этом они могут быть раскрытыми, свернутыми или вложенными друг в друга.


 Список инструментальных панелей дополнен панелью *Режим рабочего стола*. В списке горячих клавиш переименованы команды *actCodeHintExecute*, *actGotoDeclaration* на *Подсказка описания* и *Подсказка кода*.

Доработки редактора


 Добавлена настройка для изменения фона выбранной записи во всех инструментальных окнах, кроме файлового проводника.

 В окно *Структура кода* добавлен разбор функции `char()` и типа `string[]`, содержащего идентификатор.

 Доработана функция *Заменить*. При включенном параметре *Начинать от курсора* поиск производится с текущего слова под курсором. При использовании одинаковых значений поиска и замены без *Учета регистра* замена предлагается для всех соответствий независимо от регистра.

 Исключено дублирование конфигураций отладки, добавляемых мастером создания проекта.


Доработки подсказчика


 Добавлена подсказка по функциям, описанным с ключом `external`.


1.4. Издание 03.2015


Начиная с **Viper 5.5.18** используется обновленный и оптимизированный механизм организации редакторов кода и инструментальных окон.


Механизм организации окон


 В **Viper** реализована возможность разделения настроек окон на два режима: *Отладка* и *По умолчанию*. Данные режимы включают в себя открытые инструментальные окна, их состояние и расположение. Пользователю необходимо единожды настроить каждый режим, чтобы потом в зависимости от выполняемых задач выбрать наиболее удобный. Для выбора режима на инструментальной панели среды добавлен параметр в виде выпадающего списка. При запуске/завершении отладки режим отображения окон меняется автоматически. Конфигурации режимов сохраняются в отдельных глобальных файлах настройки среды (`Default_userdock_debug.vpr`, `Default_userdock_default.vpr`) и являются общими для всех проектов.

 Переработан и оптимизирован механизм открепления и стыковки окон. При "перетаскивании" окна появляется специальный индикатор, отображающий возможные направления. Позиция индикатора зависит от зоны расположения указателя. Чтобы стыковать окно в нужной части необходимо навести указатель на соответствующую сторону индикатора, при этом зона стыковки будет подсвечиваться.


 Вкладки редактора также поддерживают открепление и стыковку. Для этого необходимо мышкой потянуть заголовок вкладки. Размещать редакторы можно по вертикали или горизонтали, но только в одинаковом направлении.

 На панели вкладок справа всегда расположены кнопки для движения панели, а также выпадающий список открытых вкладок для быстрого выбора активной.


 Вкладки редактора "перетаскиваются" в "Менеджер проекта" с помощью мыши при удерживании клавиши **Alt**.


 Изменен механизм активизирования вкладок. После выполнения функции *Заккрыть* для текущей вкладки активизируется та, что была предыдущей в состоянии "активная", и история активизирования очищается.


Функции подсказки кода


 Доработан разбор конструкций языка **VIP** из активного редактора, когда данный файл отсутствует в *Списке каталогов для поиска подключаемых файлов*.


 Добавлен разбор структуры файла, записанного в кавычках `#include "Имя_файла.inc"`.

 Реализована подсказка полей для таблиц с префиксом `'type$'`. Перечень полей определяется на основании данных, полученных из пользовательского скрипта для подсказки кода.


 Реализована возможность отключения подсказчика с помощью настройки среды *Использовать подсказчик кода*, при этом функции подсказки в редакторе становятся недоступными, в окне "Дерево подключений" отключается формирование структуры, а в окне "Связи таблиц" добавляются только связи по умолчанию, которые выгружены на основании актуальной сборки **Галактика 9.1**.


 В окне автодополнения доработана навигация по списку с помощью "мыши". Выбор записи указателем осуществляется при зажатой ЛКМ.


 Для функции перехода к описанию структуры изменилась комбинация горячих клавиш на **Alt+Space**.


 Исключен автоматический вызов всплывающей подсказки по параметрам процедуры/функции. Подсказка вызывается пользователем с помощью комбинации клавиш (**Ctrl+Shift+Space**).


Редактор


 Улучшена функция *Парная скобка*. Парные скобки добавляются в конце строки или перед пробелом. Удалить обе скобки можно клавишей **Backspace**, при этом курсор должен находиться между ними.


 Изменен параметр среды *Резервное копирование файлов*. Файл копии с расширением **'bak'** сохраняется в скрытом каталоге **'backup'**. Он создается в каталоге исходного документа или среды **Viper**.


 Доработана функция *Найти в файлах*. Скрытые каталоги в поиске не участвуют.


 Доработана функция *Заменить*. При одинаковых значениях поиска и замены регистр символов учитывается. Для закрытия окна подтверждения замены назначена клавиша **Esc**.


 При создании нового файла расширение подставляется автоматически. Тип расширения соответствует выбранной синтаксической схеме в диалоговом окне создания файла. В остальных случаях учитывается настройка синтаксиса по умолчанию в параметрах среды.


 В окне приветствия добавлена возможность горизонтальной/вертикальной прокрутки списка файлов и отображение иконок для них.


 Добавлены настройки для изменения шрифта и цвета фона во всех инструментальных окнах среды. Параметры внешнего вида окна вывода удалены. Также удалены настройки *Рисовать пиктограммы* и *Кнопки навигации справа* для панели вкладок. Иконки с типом файла в заголовке вкладки отсутствуют, кнопки навигации всегда расположены справа.

 В окне *=Связи таблиц=* доработан механизм наполнения списка. Теперь связи добавляются по таблицам, которые содержатся не только в секции **From** (структуры **Create view**), но и в пользовательском скрипте для подсказчика.


 Добавлена информация о загрузке связей таблиц. А именно, при открытии среды **Viper** в окно *=Связи таблиц=* автоматически загружаются связи актуальной сборки **Галактика ERP**.


 Добавлено описание функций *"С кем" связана таблица* и *"Кто" связывается с таблицей* для окна *=Связи таблиц=*.


 В *=Окне вывода=* реализована возможность фильтрации по вводу текста. Переход по **Ctrl+E/Shift+Ctrl+E** осуществляется с учетом выполненной фильтрации. Функции *Найти* и *Искать далее* удалены. Для команды *Очистить лист* назначена комбинация клавиш **Ctrl+Del**.

 В окне *=Сообщения=* доработан механизм переключения фильтров для результата компиляции нескольких файлов. При отключении фильтра скрывается весь узел файла компиляции, если в нем содержатся сообщения только отключенного типа. Количе-


ство сообщений компилятора формируется по последнему завершённому процессу компиляции в текущем проекте.


 В инструментальные окна с колонками добавлена возможность автоподбора ширины столбца с помощью двойного щелчка ЛКМ по линии границы.

 В параметрах компиляции для списков подключаемых файлов/каталогов добавлено автоматическое удаление дублирующих записей.

 Доработан перенос конфигураций отладчика **Vip** для **Viper 5.5.6.0** и младше. При добавлении конфигураций в файл **Default.vpr** дублирование параметров исключается.

Отладчик


 Добавлена возможность выбора приоритета подключаемого ресурсного файла. Пользователю предоставляются две функции: *Подключить ресурс с приоритетом* — позволяет при подключении ввести необходимый номер, *Подключить ресурс с приоритетом репозитория* — ресурс подключится с зарегистрированным номером.


 Реализован выход из текущего блока при выполнении трассировки. С помощью функции *Выйти из текущего блока* **Alt+F8** осуществляется возврат из описания процедуры/функции к месту ее вызова для продолжения пошаговой отладки.


1.5. Издание 04.2014

Начиная с **Viper 5.5.16** данные о параметрах отладчика **Vip** хранятся в файле настроек среды, реализовано новое окно «Связи таблиц».


Функции подсказки кода

 Реализовано окно «Связи таблиц», в котором содержится список возможных связей между таблицами БД. Список формируется на основе структур, описанных в разделах *where* и *bounds*, конструкции *Create view*. На основании найденных связей формируется список таблиц с полями, имеющими связи с другими таблицами. Коллекция может быть использована при проектировании sql-запросов. В список попадают лишь те связи, которые имеют явное описание имени таблицы и поля. В окне реализована возможность автоматического перехода от связи к таблице, функция сохранения перечня связей в csv-файл, а также фильтрация по вводу.


 Доработан механизм автоподстановки значений в параметрах процедур/функций. В первую очередь подсказчик предлагает записи, тип которых соответствует текущему параметру редактируемой функции. В функциях/процедурах, предназначенных для установки ограничений (*bounds*), в качестве параметров подсказчиком будут предложены идентификаторы ограничений из текущего контекста.


 Список подсказки дополнен функциями для работы с Excel (*xlOpenExcel*, *xlCreateExcel* и др.), использование данного списка в подсказке можно отключить. Также в подсказчик добавлены функции для работы с маркерами (*AddMarker*, *AtDeleteMarker* и др.).


 В подсказчик добавлена структура *Embedded* и список ограничений связей/узлов из секции *bounds*.

 Оптимизирован процесс индексации. Обновление дерева подключения происходит при изменении структуры сборки и внесении изменений в редактор.


Редактор


 В список сворачиваемых конструкций добавлены: *if*, *case*, *for*, *while*, *do...while*.


 Доработано открытие vpr-файла. Можно установить проект в качестве рабочего проекта либо открыть его как текстовый файл на редактирование.

 Доработана настройка **Резервное копирование каждые <количество> сек.** При отсутствии доступа к каталогу измененного файла, его копия сохраняется в каталоге **Viper.exe**.

Отладчик


 Изменен способ хранения конфигураций отладчика **Vip**. Список конфигураций сохраняется в глобальном файле параметров среды (**Default.vpr**) и является общим для всех проектов. Конфигурации из созданных ранее проектов будут перенесены в новое место хранения автоматически.


 В списках каталогов/файлов (параметры элемента сборки, мастер настройки проекта) добавлена возможность выбора нескольких элементов с помощью удержания клавиши **Ctrl**.


 В режиме отладки команда **Начать отладку** (меню **Отладка**) автоматически меняется на **Продолжить отладку** и назначением данной команды становится выполнение отлаживаемого приложения.

1.6. Издание 02.2014

Функции подсказки кода


 В окне подсказки по параметрам функции (**Ctrl+Shift+Space**) текущий параметр под курсором подсвечивается.


 В список подсказки по структуре кода добавлены методы родительских интерфейсов, таблицы с префиксом '#' и 'tn' из секции **from** и идентификаторы **FieldName** конструкции **Create View**, словарные типы **Set type**, макропеременные **#declare** и **#define**, включая их параметры.


 Доработана сортировка списка автодополнения. Первый вариант совпадает с текущим словом, не учитывая позицию каретки. Вторым следует соответствие части слова до позиции курсора, далее расположены остальные варианты подсказки.

 Вывод описания о текущей конструкции недоступен в режиме отладки.

Редактор


 Доработана функция **Открыть файл Ctrl+Enter** для конструкции **#make**. Конкретные файлы открываются в редакторе, для файлов, заданных маской открывается стандартное диалоговое окно выбора файла.

 Пункт главного меню **Отладка > Начать отладку** осуществляет запуск и выполнение отладки, список конфигураций удален.

 При выполнении функций **Искать слово далее/Искать слово назад** искомое значение добавляется в диалоговое окно функции **Найти текст** в выпадающий список поля **Найти**. История сохраняется в текущем проекте.

Компилятор

 Добавлен параметр компилятора **Загружать глобальные символы из ресурсов (LoadIds)**.


 В мастере настройки проекта для параметра **Каталог для поиска исходных файлов** доступно обозначение вложенных каталогов с помощью символов '*'. По за-

вершении настройки мастер создает дополнительную переменную $\$[Tmp]$ со значением $C:\Temp\Viper$ для создаваемых временных файлов.


 Параметр *Уровень сжатия секции в ресурсе* удален.


1.7. Издание 01.2014


Функции подсказки кода

 Реализован функционал подсказки кода, который включает в себя возможность автодополнения, перехода к описанию структуры, вывода описания о текущей конструкции, подсказки параметров процедур и функций. Подсказка формируется относительно "текущего" элемента сборки в *Менеджере проекта* на основании конструкций активного редактора и подключаемых файлов (`#include`, `#make`). В новом инструментальном окне *Дерево подключений* отображается иерархия подключения файлов на уровне кода.


Компилятор

 Добавлен мастер настройки проекта, позволяющий создать проект и заполнить параметры компиляции за несколько шагов.


 Добавлена возможность создавать собственные переменные в настройках проекта и использовать их в любых параметрах компиляции и отладки.


 В случае конвертации структуры проекта в новый формат параметры на вкладках *Отладчик VIP* и *Компилятор VIP* > *БД и лицензирование* сохраняются конфигурацией с именем "по умолчанию".


Отладчик


 Добавлен параметр отладчика *Автоматически запускать отладку после компиляции*. Параметр является общим для всех конфигураций и предназначен для автозапуска процесса отладки после завершения процесса компиляции, при условии отсутствия ошибок компиляции.

 Добавлена возможность запуска отлаживаемого приложения без отладки.

 Добавлена возможность создания копии конфигурации отладки. Конфигурация включает в себя все настройки вкладки *Отладчик VIP* за исключением параметра *Автоматически запускать отладку после компиляции*.

 Удален пункт меню *Запустить*. Когда процесс отладки не запущен, по **F9** выполняется запуск отладки. Когда отладчик переходит в состояние Останов данная кнопка снова становится активной и по ее нажатию осуществляется перемещение по ТО (при их наличии) и выполнение отлаживаемого приложения.

 В окне *Таблицы в памяти* в области записей выбранной таблицы с помощью двойного щелчка мыши выполняется автоподбор ширины данного столбца. По завершении процесса отладки содержимое окна *Таблицы в памяти* очищается.

 В списке окна *Выражения и переменные* доступна возможность выбора нескольких элементов с помощью удерживания клавиши **Ctrl**.

 Добавлено описание способов подключения отладочной информации.

Редактор

При выполнении ключей `-AddProject` или `-Script` из командной строки файл скрипта не открывается в редакторе. По ключу `-ClearProject` происходит удаление структуры проекта;

Изменения в окне функции *Найти*. Удален параметр *Пометить соответствия*.

Изменения в окне функции *Заменить*. Удален параметр *Пометить соответствия*.

По нажатию **F1** текущее слово (или выделенный текст) из редактора **Viper** помещается в буфер обмена и отображается в окне chm-справки на вкладке Указатель.

1.8. Издание 05.2013

Редактор

В **Viper** реализовано окно *=Таблицы в памяти=* для просмотра содержимого логических таблиц, загруженных в память. Окно позволяет просматривать список таблиц и их данные. Функция доступна с **Атлантис 5.5.14.0**.

Доработана функция *Умные скобки* — закрывающая фигурная скобка переносится на новую строку с отступом относительно открывающей скобки, при этом курсор позиционируется между ними с дополнительным отступом.

Во всех инструментальных окнах, кроме *=Окна вывода=* и *=Структуры ресурсов=*, реализована возможность фильтрации для поиска по списку.

Удален параметр *Отступ табуляцией*. По нажатию **Tab** — для нескольких выделенных строк добавляется отступ, в остальных случаях — добавляется символ табуляции. Комбинацией **Shift+Tab** удаляется отступ в начале строки.

Удален параметр *CTRL+Нажатие мыши выделяет слово* (вкладка *Редактор > Параметры*), слово не выделяется по **Ctrl+щелчок мыши**.

Удален параметр *Drop файлов* (вкладка *Редактор > Параметры*), теперь "перетаскивание" файлов в новый редактор из операционной системы доступно всегда.

Удален параметр *Автораспознавание URL* (вкладка *Редактор > Параметры*), по **Ctrl+щелчок мыши** в строке URL — документ не открывается.

Удален параметр *Отступ табуляцией*. Использование данного символа возможно при включенной настройке *Использовать табуляцию*. По клавише **Tab** осуществляется добавление символа табуляции в позицию курсора или вместо выделенных символов одной строки, а также добавление отступа (по **Shift+Tab** — удаление), равного размеру табуляции, в начало нескольких выделенных строк/вертикального блока.


В меню *Окно* удалены функции *Новое окно*, *Заккрыть*, *Заккрыть все*, *Выровнять*.


В настройках проекта реализована возможность создания нескольких конфигураций параметров БД. Настройки базы и лицензирования объединены.

Компилятор

Поиск подключаемых файлов производится в соответствии с их расположением относительно компилируемого элемента сборки. Доработан алгоритм поиска подключаемых файлов.

В параметр *Базовый каталог для системы репозитория* автоматически подставляется базовый каталог ресурсов из **Support**. Доработка доступна с версией **Атлантис 5.5.14**.

 Параметр *Сохранять исходные коды реализации в ресурсном файле* экспортируется в sfg-файл. Доработка доступна с **Атлантис 5.5.14.0**.

 В окне *=Сообщения=* при переключении фильтров скрывается и заголовок узла, содержимое которого не соответствует условию фильтрации.

1.9. Издание 01.2013


Начиная с **Viper 5.5.10.0** используется новый формат проекта и доступны окна *=Структура ресурсов=* и *=Менеджер проекта=*.


Справочная система


В издании 01.2013 раздел "Интерфейс пользователя" дополнен новыми подразделами ["Окно менеджера проекта"](#)⁵⁸ и ["Окно структуры ресурсов"](#)⁶⁵.


Удалены разделы: "Окно проекта", "Элементы сборки", "Файлы проекта", "Открытые файлы", "Вкладка Порядок сборки", "Вкладка Файлы".


Редактор


 Произведена замена *=Окна проекта=* на *=Менеджер проекта=*. Окно предназначено для организации файлов проекта. Состав проекта дополнен объектами: виртуальный каталог и пакет, который предоставляет возможность использования конфигурации параметров компилятора. В окне реализована многоуровневая иерархия объектов и множественный выбор узлов, а также группировка, сортировка и фильтрация узлов.


 В **Viper** интегрированы основные функции по работе с системой контроля версий (*Просмотр хранилища*, *Настройки TortoiseSVN*, *Обновить до ревизии*, *Зафиксировать*, *Различия* и др.).


 Реализована функция *Форматор кода* для приведения кода к стандартному виду. Команда доступна в контекстном меню редактора и применяется к активной вкладке или выделенному абзацу.


 Добавлена возможность "перетаскивания" конструкций по коду посредством окна *=Структура кода=*. Данная функция доступна, когда сортировка строк выполнена по возрастанию. Каждое перемещение автоматически проверяется на соответствие правилам вложенности. Вместе с конструкцией переносятся все точки останова, метки и закладки, содержащиеся в ней.


 Реализовано окно *=Структура ресурсов=* для просмотра содержимого ресурсных файлов.

 В контекстное меню редактора добавлены функции *Предыдущая* **Alt+UP**/*Следующая конструкция* **Alt+Down** для навигации по коду.


 Добавлена возможность удаления группы символов, в зависимости их расположения от курсора, с помощью комбинации клавиш **Ctrl+Delete**/**Ctrl+BackSpace**.

 При выполнении функции *Найти в файлах* с настройкой *Заменять* все несохраненные изменения в *Модифицированных* файлах остаются, и статус файла не изменяется. Файлы из скрытого каталога в поиске не участвуют.

 В окне *=Файловый проводник=* реализовано "перетаскивание" файлов в структуру *=Менеджера проекта=*, интегрировано меню системного проводника, а также добавлена возможность выбора текущего расположения (каталог проекта, *Мой компьютер*, *Сеть*) и функция *Сделать корнем*.


 При попытке открыть несуществующий файл из *=Менеджера проекта=* пользователю предоставляется возможность заменить этот файл.


 В функции *Формат строки* для поля *Разбить на* минимальное количество символов ограничено значением "2".


 Удален параметр *Перемещать курсор по нажатию правой кнопки мыши*. Теперь при вызове контекстного меню курсор всегда устанавливается в место вызова.


Компилятор


 Возможность выбора компилятора (*Атлантиса*). В *Параметры проекта* добавлена настройка *Каталог Атлантиса*.


 Добавлена новая макропеременная *\$(AtlPath)*. Она соответствует значению поля *Каталог Атлантиса* (в параметрах *Компилятора VIP*). При ее использовании настройки, в которых содержатся пути к файлам *Атлантиса*, становятся универсальными.


 При добавлении каталога в параметрах *Компилятора VIP* и *Отладчика* теперь открывается соответствующее стандартное диалоговое окно. Поля *Список каталогов для поиска подключаемых файлов* и *Дополнительный список каталогов* доступны для ввода текста и вместо списка вложенных каталогов теперь можно указывать символ "*".


 При [*Экспорте*] *Списка каталогов для поиска подключаемых файлов* символ "*" в *sfg*-файле заменяется списком с адресом каждого вложенного каталога.

 Добавлена возможность просмотра доступных *Атлантис*-систем. Для этого в настройке *Название системы в репозитории* предназначена кнопка [*Загрузить список доступных Атлантис-сисем*].


 *Уровень диагностики* в настройках *Компилятора VIP* дополнен параметром *Выдавать информацию о причине перекомпиляции файлов* (соответствует *Compilers.VerboseMake*). Данная настройка доступна при условии вывода *Всех сообщений* компилятора.


 При отсутствии подключения к базе данных или несоответствии версии словаря БД процесс компиляции выполняется, но работоспособность созданного ресурсного файла в таком случае не гарантируется.


 В процессе компиляции поиск подключаемых файлов первоначально производится в каталоге компилируемого файла (*.prj). Скрытые каталоги не учитываются при поиске подключаемых файлов.


 При выполнении процесса компиляции в *=Окне вывода=* отображается версия соответствующего компилятора (*ViperCompile.exe*).


Отладчик


 Реализована возможность создания нескольких конфигураций для запуска отлаживаемого приложения. Каждая конфигурация представляет собой набор параметров, которые хранятся на вкладке *Отладчик VIP*. Выбор необходимой конфигурации осуществляется перед выполнением отладки в списке команды *Начать отладку*. Отлаживаемое приложение создается в соответствии с выбранной конфигурацией.

 В окне *=Загруженные интерфейсы=* для списка файлов реализована возможность выбора нескольких файлов и их открытие в редакторе. Добавлен столбец *Компонент*.

 В контекстном меню окна *=Интерфейсы=* добавлена функция *Запуска внешнего интерфейса*.

 В процессе отладки по наведению указателя на код составного выражения или функции отображается всплывающая подсказка с соответствующим значением.


 В окно *=Выражения и переменные=* добавлена возможность фильтрации списка по мере ввода текста, при этом фильтруется каждый столбец.

 Доработана функция *Выполнить до курсора*, а именно, функция выполняется только при наличии отладочной информации в строке с курсором.

Дистрибутив


Начиная с комплекта поставки **Атлантис 5.5.11** и **Viper 5.5.10** (дистрибутив Viper5.5.10.0-vip5.5.11.0) рекомендуется каждую последующую установку интегрированной среды **Viper** производить в один каталог. Это позволит накапливать компиляторы **VIP** различных версий для возможности простого переключения между ними.


1.10. Издание 06.2012


 Начиная с **Атлантис 5.5** визуальная часть (в прошлом **Vipide.dll**) приложения **Viper** помещена в самостоятельное приложение **Viper.exe**, а функции компилятора и интеграция с **Атлантис** сконцентрированы в приложении **ViperCompiler.exe**.


Редактор


 Реализован механизм подсказки ввода для текущего редактора. Вызов функции осуществляется комбинацией клавиш **Ctrl+Space**, а вставка наименования из списка выполняется различными вариантами с помощью клавиш **Enter**, **Tab**, **Space** или клавишами знаков. Для языка **VIP** список формируется на основании дерева структуры кода, ключевых слов и стандартных процедур/функций языка, а для скриптовых языков (Visual Basic Script и JScript) на основании [API редактора](#)^[151]. Список подсказки фильтруется по мере ввода текста.


 Реализована возможность создания подсказчика на основе пользовательского словаря. Наполнение производится через интерпретатор скриптов, которые выполняются как при вызове через меню или "горячей" клавишей, так и после выполнения назначенного события. Пользовательская подсказка представляет собой запись блока в коллекции, причем в качестве расширения возможно наличие вложенных записей, которые становятся доступными после знака ".".

 В *Список событий* для запуска скриптов добавлено два новых события: *Изменение файла* (например, для автозаполнения либо автоформатирования) и *Вызов подсказки кода* (для наполнения списка подсказки пользовательскими конструкциями).


 Функция *Комментировать/раскомментировать* теперь применяет символы однострочного комментария к строкам, когда все они выделены целиком, в противном случае, добавляются символы многострочного комментария.

 Для *Комментирования/раскомментирования* (однострочным комментарием) функция *Отменить* выполняется над целой группой.


 В функциях *Добавить/Убрать отступ* размер отступа теперь равен *Ширине табуляции*.

 Функция *Добавить метку в Название* помещает выделенный текст или текущее слово, в *Комментарий* добавляется содержимое текущей строки.


 Функция *Установка закладки в Комментарий* добавляет содержимое текущей строки.


 В диалоговом окне функции *Специальная вставка* переключатель *С добавлением* переименован на *С добавлением текста*, наименование поля для ввода отсутствует.

 Файлы FastReport (*.fr3, *.fp3) из =Файлового проводника= и окна =Сообщения= открываются во внешней утилите FREasyEdit.


 В окне =Сообщения= переключение фильтров не влияет на состояние (свернутость/развернутость) узлов.


 В строке заголовка приложения **Viper** отображается версия файла **ViperCompile.exe**.


 Переработано окно =О версии=, а именно, добавлена колонка Путь, реализован механизм фильтрации записей по введенным символам, в локальном меню доступна возможность *Копировать* текущую запись и *Копировать все* записи списка.


 В *Параметры редактора* добавлена вкладка Подсказка кода с настройками стиля и размера списка, а также вариантами источников наполнения подсказки.


 В *Параметрах* редактора на вкладке Общие > Скрипты изменены кнопки управления списками, а также реализован множественный выбор элементов списка.


 Включенный *Параметр* редактора *Запускать только один экземпляр программы* отвечает за внешнее открытие файлов в одном экземпляре **Viper**.


 В *Параметрах* среды на вкладке Редактор > Параметры настройка *Прокрутка за конец файла* переименована на *Прокрутка за предел файла*.


 В *Параметрах* редактора отсутствует неактуальная вкладка Общие > Каталоги. Настройка *Шаблоны для создаваемых файлов* перемещена на вкладку Просмотр.


 В *Параметрах* среды на вкладке Редактор > Параметры отсутствует неактуальный параметр *Отключать режим вертикальной пометки*. Режим выделения вертикального блока доступен всегда.


 В *Параметрах* среды на вкладке Редактор > Проверка правописания отсутствует неактуальный параметр *Начинать от курсора*.


 В *Параметрах* среды на вкладке Редактор > Служебное поле отсутствует неактуальный параметр *Ширина* и *Авторазмер*.


 В *Параметрах* редактора отсутствует неактуальная вкладка Общие > Печать.


 Реализована возможность подключения к БД **PostgreSQL**. Драйвер БД **Oracle 7.3** теперь не поддерживается.

 В *Параметрах проекта* на вкладке Компилятор Vip > База данных добавлена настройка *Номер офиса* для аутентификации при подключении к БД.


 При сохранении измененных *Параметров проекта* на вкладке База данных, в условиях установленного подключения к ней, производится автоматическое отключение от БД.


 История *Параметров проекта* *Каталог аппаратного ключа* и *Имя лицензионного файла* сохраняется в выпадающем списке автоматически, а удаляется с помощью "горячих" клавиш **Ctrl+Del**.


 В *Параметрах проекта* на вкладке Каталоги отсутствует неактуальная настройка *Список каталогов для поиска исходных файлов*.

 Удалены свойства скриптовой машины `FileName` и `ProjectName` объекта `Project`.


Компилятор

 Процесс компиляции с ключом `-CompileAll` или `-BuildAll` останавливается в соответствии с параметрами компиляции текущего проекта. В сочетании ключей `-Exit -SkipError` **Viper** закрывается даже при наличии ошибок сборки.

 При запуске компиляции фильтр по введенным символам в окне `=Сообщения=` снимается.

 Теперь, когда `=Окно вывода=` или окно `=Сообщений=` не закреплено, оно автоматически всплывает. Когда оба окна скрыты, то окно `=Сообщения=` всплывает лишь при наличии *Ошибок* компиляции.

Отладчик


 В редакторе значение составной записи (поля таблицы) отображается во всплывающей подсказке при наведении курсора на выделенную запись отлаживаемого кода.

1.11. Издание 01.2012


Справочная система


Издание 01.2012 полностью переработано и дополнено. Изменения коснулись внешнего вида справки. А именно, в заголовках разделов кроме кнопок навигации добавлен полный путь раздела, по нажатию на который можно перейти к нужному родительскому разделу. Новая кнопка `[Сообщить о неточности в документации]` позволяет пользователю отправить свой отзыв или пожелание по дополнению справки на адрес электронной почты viper@galaktika.by. Пожелания будут учтены в будущих изданиях.

Редактор


 **Последние файлы.** Файлы, открытые через окно `=Файловый проводник=` тоже добавляются в список.


 Файлы, открытые в **Viper** способом Drag&Drop, теперь также добавляются в список последних открытых файлов.


 **Инкрементальный поиск вперед/обратно.**


 Добавлен выключатель функции **Вставки из буфера при нажатии средней кнопки мыши.**


 На панель инструментов добавлена кнопка вызова окна `=Сообщения=`.

 Количество сообщений компилятора по типам подсчитывается по последнему сеансу компиляции, при этом сообщения из истории в подсчете не участвуют.


 Дополнительно реализована возможность фильтрации сообщений по мере ввода символов с учетом установленных фильтров.


 Окно `=Сообщения=` и `=Окно вывода=` скрываются по нажатию клавиши **Esc**.


 **Сворачивание кода.** Реализовано сворачивание основных структур языка **VIP** (интерфейсы, обработчики событий, процедуры, операторные скобки и другие), пользовательских регионов (произвольные блоки, обрамленные конструкцией `// #region ... // #endregion`) и многострочных комментариев из любой позиции блока.


 Сворачивание определенных конструкций (события, процедуры/функции, пользовательские регионы).


 **Свернуть до N уровня.** Для управления функцией добавлено использование клавиш **Enter** и **Ecs**.


 Возможен выбор типов сворачиваемых структур.


 Пользовательский регион. Функция *ЗаклЮчить код в* обрамляет произвольный выделенный блок конструкцией `//#region ... //endregion` и применяется для использования функции сворачивания пользовательских регионов.


 Функция *Парная конструкция*. Используется для поиска парных элементов основных конструкций: `Interface|End`, `Begin|End`, `Case|End`, `{...}`, многострочные комментарии и другие. Переход осуществляется в прямой и обратной последовательности с подсветкой всей конструкции.

 Постраничное выделение вертикального блока. Функция позволяет постранично выделять вертикальный блок комбинацией клавиш **Alt+Shift+PgUp** или **Alt+Shift+PgDn**.


 Функции *Комментирования/Раскомментирования* объединены в одну команду с добавлением возможности многострочного комментирования и работы с вертикальным блоком.


 Функция навигации по открытым файлам улучшена диалоговым окном выбора вкладок, которое содержит список открытых файлов и инструментальных окон. Дополнительным преимуществом окна является возможность поиска.


 *Перекодировать*. Для данной команды реализована функция отмены.


 Для языка **VIP** реализована раскраска SQL-конструкций: *Sql Query*, *Sql_Loop*, *SqlQuery*.


 Улучшена функция поиска по регулярным выражениям. Исправлена работа с кириллицей.


 Изменен алгоритм работы с нумерованными закладками. Теперь они не редактируются, а повторная закладка перемещается сразу при установке.

 Внешний вид редактора. Редактированные строки помечаются полосой в служебном поле. Ее цвет меняется в зависимости от статуса изменений. Обновлено изображения отладочной стрелки, точек останова и меток, а также их функций.


 Параметры проекта. Теперь автоматическое сохранение проекта происходит при запуске компиляции.


 Окно *=Структура кода=*. Доработан алгоритм обновления списка и разбора отдельных конструкций.


 Создано функциональное окно *=Структура кода=* для отображения основных элементов языка **VIP** (интерфейсы, окна, обработчики, функции и др.) по тексту исходного кода, открытого в редакторе.

 Доработано окно уведомления об ошибке, возникающей при некорректной работе приложения **Viper**. С помощью кнопки [Отправить протокол в службу поддержки] можно отправить полную информацию о причинах ошибки на адрес viper@galaktika.by.


Компилятор

 Статус и этапы компиляции с индикатором процесса теперь отображаются в *=Окне вывода=*, а текущее время компиляции — в статусной строке редактора.


 Результат компиляции выводится в окно *=Сообщения=*, если были ошибки и/или предупреждения и в *=Окно вывода=* в остальных ситуациях. Скрыть окна можно с помощью клавиши **Esc**.

 Незакрепленное окно текущего этапа компиляции автоматически всплывает.


 Доработана функция завершения процесса компиляции.

 Реализована возможность отключения от базы данных в пункте меню *Проект > Отключиться от БД*. Подключение к БД производится автоматически в момент запуска компиляции.


Отладчик


 Доработана передача фокуса вводу окну **Viper** при переходе к точке останова.


 Средства для компиляции и отладки без остановки **Галактики**.


 В контекстное меню *=Окна проекта=*, для элементов сборки добавлен пункт *Подключить/отключить ресурс*. С помощью него можно отключить выбранный ресурс от целевого приложения и подключить обратно. Данный пункт активен только в режиме отладки.


 Информация обо всех отключениях/подключениях выводится на вкладке *Консоль =Окна вывода=*.

 В панели инструментов **Viper** реализованы кнопки [*Останавливаться на конструкторе*], [*Отлаживать все*] и [*Перехватывать исключения*].


 Доработаны и усовершенствованы инструменты отладки, *=Окно локальных переменных=* и *=Окно выражений и переменных=*.


 В окне *=Окне локальных переменных=* и *=Окне выражений и переменных=* отображение структур, массивов, буферов записей БД в виде древовидной структуры.


 В режиме отладки доступно интерактивное изменение значения, как обычной переменной, так и элемента массива или поля структуры.

 Используя команды выпадающего меню, добавлена возможность копировать имя или значение переменной в буфер обмена.

 Переменные, изменившиеся с прошлого шага, автоматически выделяются цветом.

 Окно *=Выражения и переменные=*. Реализована возможность множественного выбора выражений (с удерживанием клавиши **Ctrl** или **Shift**) для их удаления, а также полной очистки списка.

 Drag&Drop. Теперь переменную (выражение) можно "перетянуть" мышкой в окно *=Выражения и переменные=* непосредственно из области редактора кода.

 Реализована сортировка данных в окне *=Точки останова=*. Теперь список точек останова можно сортировать по номеру строки, имени файла и функции.

Дистрибутив

- Реализовано приложение для автоматической инсталляции интегрированной среды **Viper**.
- Дистрибутив **Viper**, а также архив для портативной установки доступен по адресу ftp://ftp.galaktika.ru/pub/support/galaktika/bug_fix/GAL810/Viper.

2. Основы использования

Viper предназначен для создания программного кода на языке **VIP** и компиляции ресурсных файлов.

Viper состоит из следующих функциональных модулей:

- редактор исходных текстов;
- компилятор;
- отладчик;
- средства внешнего запуска и выполнения.

Встроенные в **Viper** компилятор и отладчик позволяют минимизировать время разработки программного кода, не затрачивая его на другие инструментальные средства.

2.1. Использование проекта

Проект является главной рабочей единицей в интегрированной среде разработки **Атлантис**-приложений **Viper**. С ним связаны все основные функции и возможности как редактора, так и остальных функциональных модулей. Без использования проекта среда разработки **Viper** представляет собой лишь текстовый редактор.

Работа в **Viper** начинается с создания нового или открытия существующего проекта, который содержит описание рабочей среды и является центральным местом для хранения параметров:

- настройки компилятора;
- настройки отладчика;
- список модулей (проектных файлов);
- рабочие файлы исходных кодов;
- и т. д.

Кроме того, проект обеспечивает структурирование набора файлов по категориям их использования.

2.2. Встроенный компилятор

В редактор **Viper** встроен компилятор **VIP**. Функционал, связанный с компилятором, позволяет выполнить сборку любых исходных файлов в требуемый ресурсный модуль. Механизм использования проекта позволяет автоматизировать компиляцию группы файлов в отдельные выходные ресурсы. Существует возможность назначения индивидуальных параметров для каждого ресурса.

Принцип компиляции в **Viper** основан на использовании многопроцессности, что позволяет продолжать работу с редактором во время компиляции. Данный подход также защищает редактор от необработанных исключений компилятора, которые могут привести к аварийному завершению приложения.

Использование встроенного компилятора позволяет сократить время, затрачиваемое на пересборку ресурсов. Это реализовано за счет оптимизации инициализации компилятора. При повторных вызовах компилятора экономится время на создание процесса компиляции и подключение к базе.

Компиляция в среде позволяет быстро локализовать ошибки сборки. Для этого используются интерактивные сообщения, т. е. сообщения компилятора, поступающие на обработку пользователю непосредственно при их возникновении во время компиляции. Вывод сообщений осуществляется в окне «Сообщения» в виде древовидного списка.

2.3. Интегрированный отладчик

Встроенный в **Viper** отладчик обладает всеми основными функциональными возможностями, присущими отладчикам: механизм точек останова, использование методов трассировки и шага, просмотр значения переменных, навигация по элементам структуры кода и др.

Функционал отладчика имеет хорошо развитый механизм интерактивности с пользователем, что позволяет выполнять отладку почти ненастроенного проекта. При этом отладчик будет запрашивать недостающие параметры у пользователя.

2.4. Примеры использования

Для удобства ознакомления с возможностями компиляции и отладки в интегрированной среде **Viper** пользователю предоставляется готовый проект **Sample.vpr** с несколькими примерами разработки на языке **VIP**. Данный проект доступен в папке **Samples**, которая добавляется в каталог интегрированной среды **Viper** при его установке.

3. Интерфейс пользователя

При первом запуске приложения **Viper** отображается окно «Приветствие».

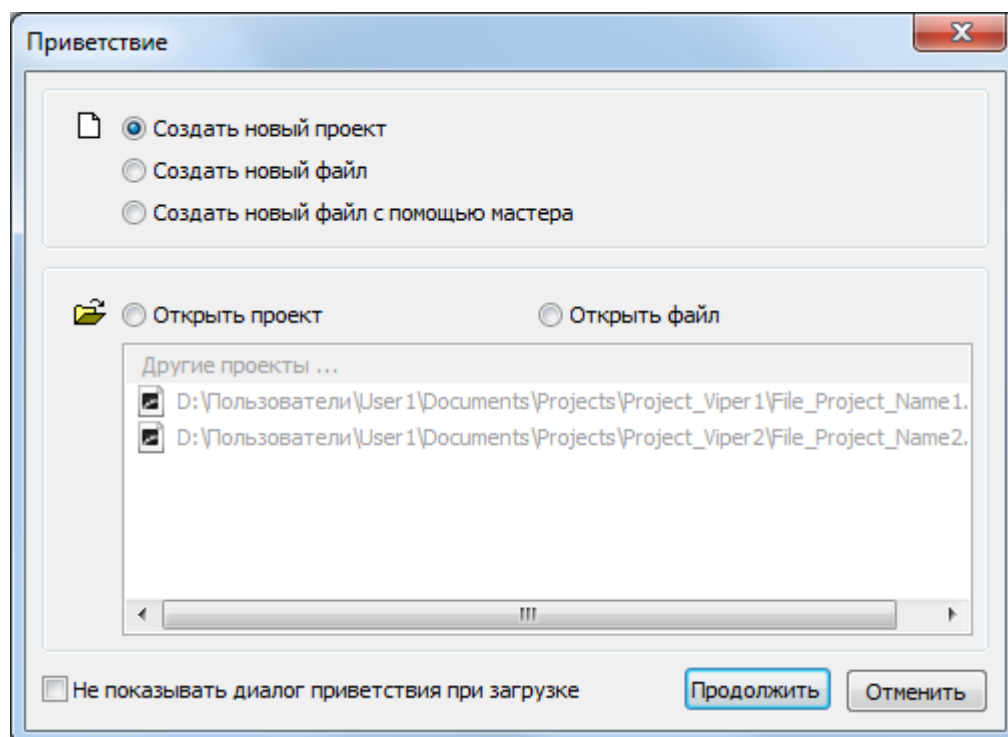


Рис. 1. Окно приветствия

Возможности окна предоставляют выбор одного из действий:

- **Создать новый проект.** Подробнее о команде см. в разделе "[Меню Проект](#)⁵⁰".
- **Создать новый файл** — открытие нового редактора кода, заголовок которого обозначен как *Безымянный* с порядковым номером. При сохранении изменений открывается диалоговое окно команды *Сохранить как* (см. "[Меню Файл](#)²⁴").
- **Создать новый файл с помощью мастера** — выполнение команды *Создать новый файл* с использованием диалогового окна настроек.
- **Открыть проект** — выбор необходимого проекта из предложенного списка (дополнительно см. [предупреждение](#)⁵⁰).
- **Открыть файл** — выбор необходимого файла из предложенного списка.

Параметр *Не показывать диалог приветствия при загрузке* позволяет скрыть данный диалог при запуске среды. В дальнейшем отобразить данное окно можно с помощью параметров среды (меню *Вид > Параметры > вкладка Общие > Просмотр*).

[[Продолжить](#)] — выполнить указанную команду.

[[Отменить](#)] — закрыть окно «Приветствие» и перейти в главное окно приложения **Viper**.

3.1. Главное окно приложения

Главное окно приложения имеет вид, типичный для приложений с графическим интерфейсом. В верхней границе окна расположена строка заголовка. Она состоит из имени открытого [проекта](#)⁸¹ (при наличии), имени приложения — **Viper**, версии файла

ViperCompile.exe (см. "[Меню Справка](#)⁵⁵"), а также имени открытого файла (при наличии), причем в настройках среды имеется возможность отображать *имя файла с путем в заголовке окна* (см. "[Вкладка Общие - Просмотр](#)¹⁰⁷").

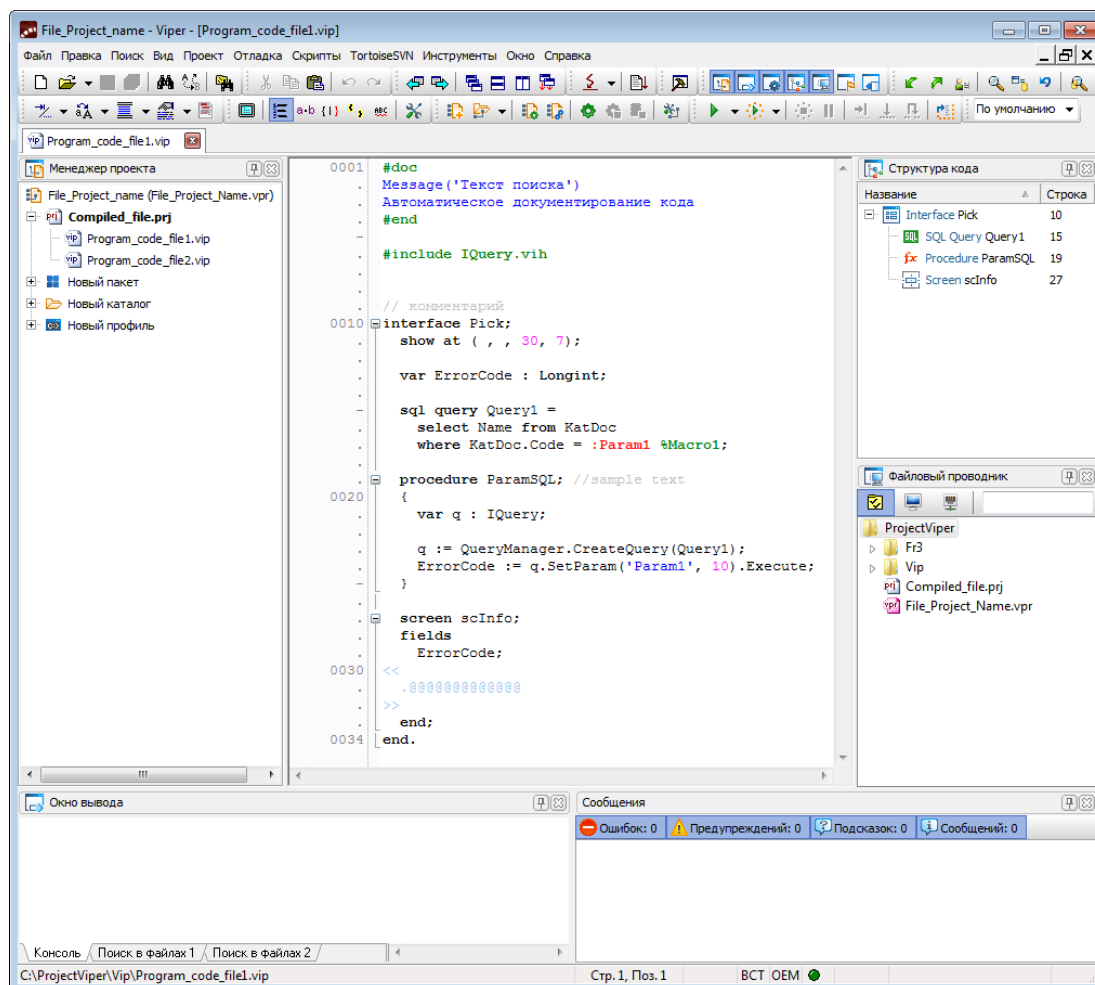


Рис. 2. Главное окно приложения

Под строкой заголовка расположено меню и инструментальные панели. В нижней части приложения — статус-строка. Центральную часть главного окна занимают текстовый редактор и инструментальные окна:

- =[Менеджер проекта](#)⁵⁸;
- =[Файловый проводник](#)⁶³;
- =[Окно вывода](#)⁶⁸;
- =[Сообщения](#)⁶⁹;
- =[Окно стека](#)¹³⁶;
- =[Выражения и переменные](#)¹³⁷;
- =[Локальные переменные](#)¹³⁸;
- =[Точки останова](#)¹³⁵;
- =[Таблицы в памяти](#)¹⁴¹;
- =[Интерфейсы](#)¹³⁹;
- =[Загруженные интерфейсы](#)¹⁴⁰;
- =[Структура кода](#)⁶¹;

- =[Метки](#)⁶⁴=;
- =[Окно структуры ресурсов](#)⁶⁵=;
- =[Дерево подключений](#)⁶⁶=.

Для каждого инструментального окна можно изменять размер, состояние и расположение (подробнее см. "[Управление окнами](#)"⁵⁷). Причем данные настройки можно сохранить в качестве режимов, чтобы в дальнейшем выбирать наиболее удобный для выполнения определенных задач. Выбор режима осуществляется в выпадающем списке на инструментальной панели.

3.2. Меню верхнего уровня

Основное меню расположено в верхней части окна. Каждый его элемент содержит подменю из определенного набора команд.

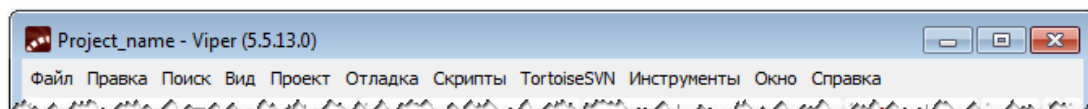


Рис. 3. Меню верхнего уровня среды разработки

Файл — команды управления файлом (см. "[Меню Файл](#)"²⁴).

Правка — набор команд для редактирования текста в окне редактора (см. "[Меню Правка](#)"²⁷).

Поиск — команды поиска, замены текста, управления маркерами в редакторе кода или во внешних файлах (см. "[Меню Поиск](#)"³⁶).

Вид — команды управления видимой частью среды разработки (состав отображаемых окон и их настройка, см. "[Меню Вид](#)"⁴⁶).

Проект — команды для работы с проектом (см. "[Меню Проект](#)"⁵⁰).

Отладка — команды интерактивного отладчика (см. "[Меню Отладка](#)"⁵¹).

Скрипты — настройка и выполнение скриптов (см. "[Меню Скрипты](#)"⁵³).

TortoiseSVN — основные команды для работы с приложением TortoiseSVN (см. "[Меню TortoiseSVN](#)"⁵³).

Инструменты — настройка и выполнение инструментов (отдельных программ или их комбинаций, см. "[Меню Инструменты](#)"⁵⁴).

Окно — управление окнами открытых файлов среды разработки (см. "[Меню Окно](#)"⁵⁴).

Справка — получение справочной информации и назначение "горячих" клавиш (см. "[Меню Справка](#)"⁵⁵).

3.2.1. Меню Файл

Меню *Файл* содержит набор команд для работы с файлами.

Создать — создание нового файла одним из способов:

- без предварительного сохранения, при этом открывается редактор с заголовком *Безымянный* и порядковым номером. Номер формируется при создании файла без указания имени в текущем сеансе работы в *Viper*;

- с вызовом диалогового окна для сохранения файла. Для этого необходимо включить параметр *При создании нового файла открывать диалог настройки* (см. "[Вкладка Общие — Просмотр](#)¹⁰⁷").

После вызова команды *Создать* открывается диалоговое окно.

В открывшемся окне необходимо выбрать формат файла на одной из вкладок:

- *Файл* — новый файл;
- *Шаблон* — файл на основе шаблона.

Шаблон представляет собой набор текста в файле. Каталог шаблонов указывается в главном меню *Вид > Параметры* (см. "[Вкладка Общие — Просмотр](#)¹⁰⁷").

На вкладке *Файл* можно выбрать оформление синтаксической схемы для файла (см. "[Вкладка Atlantis VIP — Подсветка](#)¹²³"):

- *Без подсветки синтаксиса* — отображение текста в редакторе одним цветом без выделения синтаксических конструкций. При использовании данного параметра файлу с полным именем присваивается расширение *txt*, при отсутствии полного имени — используется настройка синтаксиса по умолчанию в параметрах среды (см. "[Вкладка Синтаксис](#)¹²¹").
- *С подсветкой синтаксиса по правилам языка* — выделение синтаксических конструкций текста в редакторе с использованием различных цветов, шрифтов и начертаний. Стиль раскраски зависит от выбранной синтаксической схемы в предложенном списке. Также от выбранной синтаксической схемы зависит и расширение файла, которое автоматически присвоится файлу при сохранении.

Настройки подсветки синтаксиса содержатся в параметрах среды, при необходимости их можно изменить.

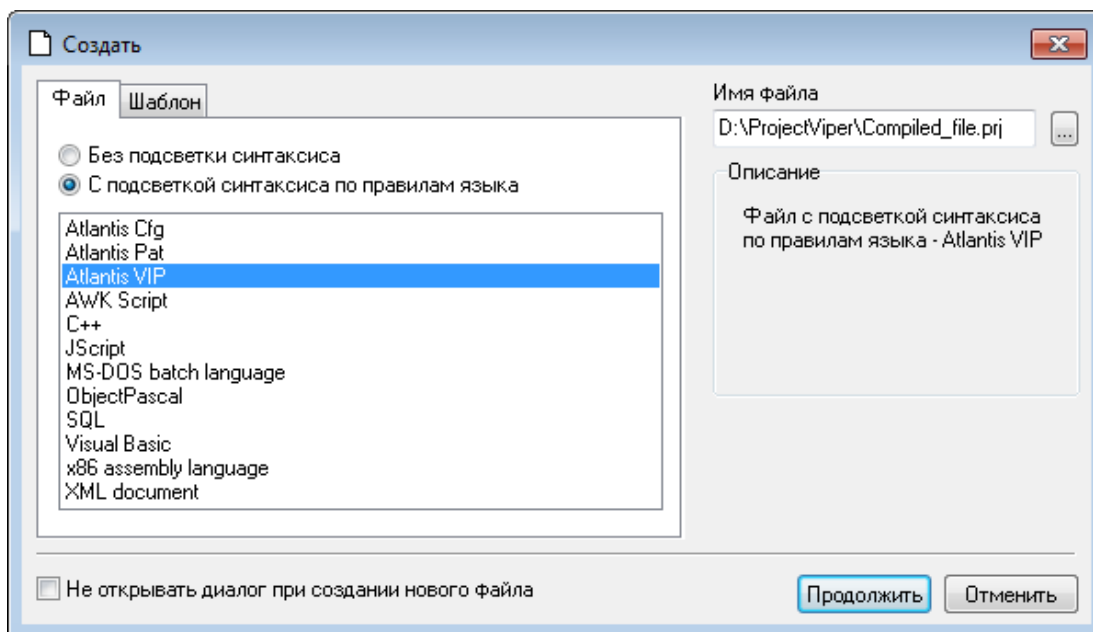



Рис. 4. Диалоговое окно создания файла

Имя файла — имя нового файла. По умолчанию файл не имеет пути и не сохраняется в файловой системе. Для его сохранения необходимо указать полный путь с помощью кнопки . По нажатию на нее, откроется стандартное диалоговое окно для выбора расположения файла. В данном диалоговом окне расширение файла можно ввести в ручную, при отсутствии расширения — оно будет присвоено автоматически после нажатия кнопки *[Сохранить]*. Расширение зависит от выбранного оформления синтаксической схемы (см. параметры выше).

Если в строке **Имя файла** содержится полный путь без расширения файла, то файл будет сохранен без него.

Параметр **Не открывать диалог при создании нового файла** позволяет скрыть диалоговое окно параметров при дальнейшем использовании функции **Создать**.

Открыть — открытие существующего файла, при этом осуществляется проверка его содержимого на нулевые символы (#0). При наличии таких символов пользователю предлагается заменить их пробелами. В случае положительного ответа все данные символы заменяются пробелами и для файла устанавливается статус *Модифицирован*. При отрицательном ответе файл в редакторе не открывается и выводится соответствующее предупреждение.

Открыть повторно — переоткрытие активного файла. При этом все его несохраненные изменения будут утеряны.

Последние файлы — список последних открытых файлов в редакторе, в том числе, открытых способом Drag&Drop. **Максимальное количество файлов** определяется в параметрах среды (см. "[Вкладка Общие](#)"¹⁰⁵).

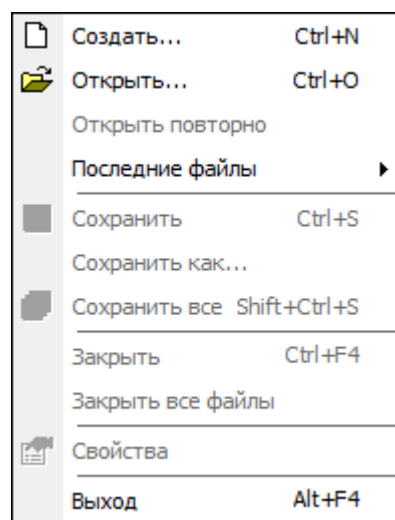


Рис. 5. Меню Файл

Сохранить — сохранение изменений активного файла.

Сохранить как — сохранение текста редактора в новом файле. Расширение файла по умолчанию остается прежним, при необходимости его можно изменить.

При сохранении файла с полным именем, уже существующим в проекте, формируется запрос пользователю о замене файла, добавленного ранее.

При попытке заменить файл с атрибутом *Только чтение* сохранение не происходит, имя файла не изменяется.

Сохранить все — сохранение всех открытых файлов.

Закрывать — закрытие активного файла. При наличии несохраненных изменений сформируется запрос на их сохранение.

Закрывать все файлы — закрытие всех вкладок редактора, при этом выполняется [проверка на наличие несохраненных изменений](#)²⁶.

При обнаружении несохраненных файлов открывается диалоговое окно, предоставляющее пользователю выбор нужного файла и выполнение необходимого действия:

- **[Да]** — сохранить только выделенные файлы.
- **[Нет]** — удалить несохраненные изменения во всех файлах.

- [Отменить] — не сохранять файлы и вернуться на этап, предшествующий активизации данного окна.
- [Сохранить все] — сохранить все файлы из списка.

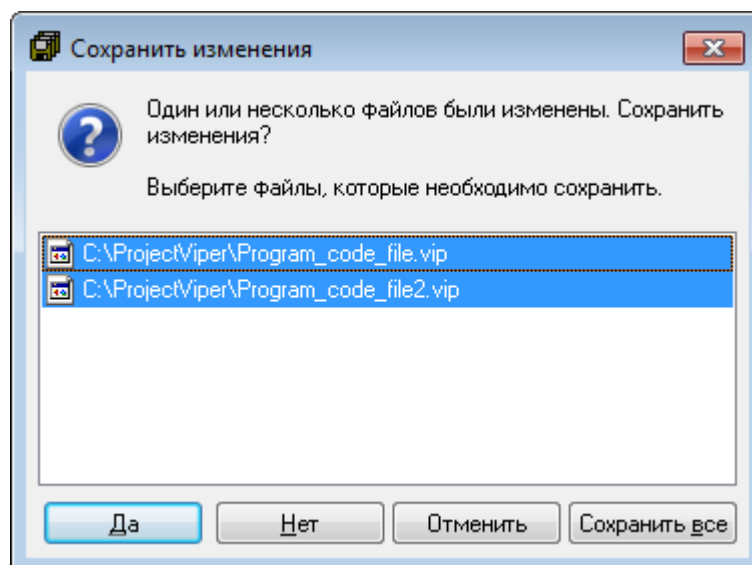


Рис. 6. Диалоговое окно сохранения измененных файлов

Свойства — открытие окна свойств активного файла. Стандартные свойства отображаются на вкладке *Общие*. Вкладка *Дополнительно* содержит информацию о количестве **Строк** и **Символов**, а также сведения о синтаксической схеме **Подсветки синтаксиса** и **Длине табуляции** с возможностью их изменения при необходимости.

Выход — закрытие программы, при этом выполняется [проверка на наличие несохраненных изменений](#)²⁶.

3.2.2. Меню Правка

Меню **Правка** содержит команды, выполняющие редактирование текста в окне редактора. Данные команды позволяют изменять целые блоки, импортировать содержимое внешних файлов, выполнять автоматическую проверку орфографии и удобно работать со структурой кода при помощи инструментов сворачивания конструкций.

Отменить — отмена (откат) последней команды редактирования. Возможность **Очищать очередь отмены после сохранения** доступна в параметрах среды (см. "[Вкладка Редактор — Параметры](#)"¹¹⁶).

Вернуть — возврат последней отмененной команды редактирования.

Вырезать — удаление выделенного текста с добавлением его в буфер обмена.

Вырезать в конец буфера — удаление выделенного текста с добавлением его в конец буфера обмена.

Копировать — копирование выделенного текста в буфер обмена.

Копировать в конец буфера — копирование выделенного текста с добавлением его в конец буфера обмена.

Вставить — вставка содержимого буфера обмена в позицию курсора. Выделенный текст при этом заменяется.

Специальная вставка — вставка содержимого буфера обмена в позицию курсора с использованием дополнительных параметров (см. "[Команда Специальная вставка](#)"²⁹).

Удалить — удаление выделенного текста.

Выделить все — выделение всего текста.

Сворачивание кода — команды управления состоянием развернутых блоков структуры кода (см. "[Меню Сворачивание кода](#)^[30]").

Разворачивание кода — команды управления состоянием свернутых блоков структуры кода (см. "[Меню Разворачивание кода](#)^[31]").

Заключить код в — обрамление выделенного текста конструкцией пользовательского региона `// #region...// #endregion`, что позволяет управлять состоянием произвольного блока.

Форматирование — группа команд форматирования файла (см. "[Меню Форматирование](#)^[31]").

Изменение регистра — изменение регистра выделенного текста (см. "[Меню Изменение регистра](#)^[32]").

Блок — команды редактирования блока текста (см. "[Меню Блок](#)^[32]").

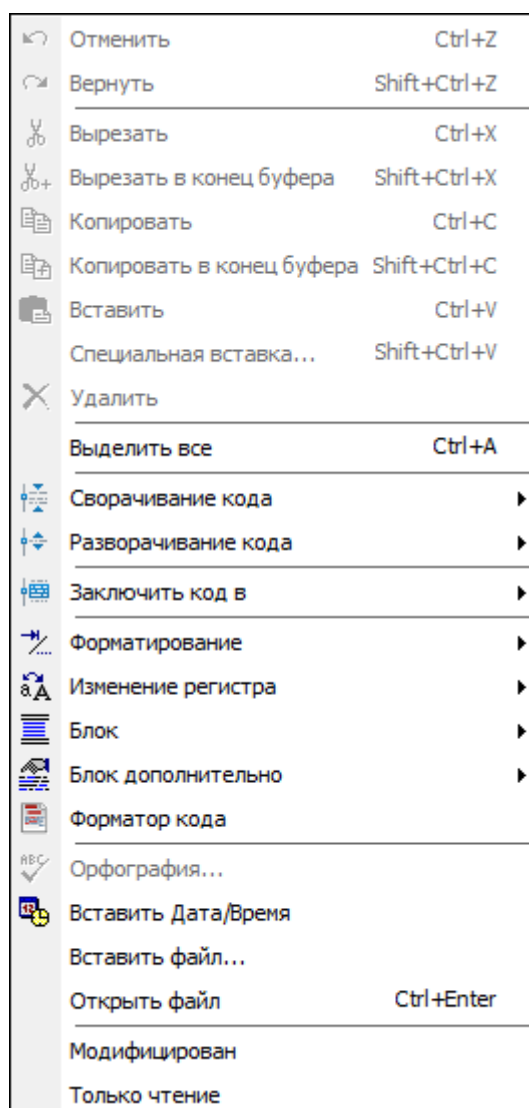


Рис. 7. Меню Правка

Блок дополнительно — расширенные команды редактирования блока текста (см. "[Меню Блок дополнительно](#)^[33]").

Форматор кода — функция приведения кода к стандартному виду (см. "[Функции ре-факторинга](#)"⁷⁹).

Орфография — проверка активного файла на наличие орфографических ошибок (см. "[Команда Орфография](#)"³⁵).

Вставить Дата/Время — добавление текущей даты и времени в позицию курсора. Формат их представления устанавливается в параметрах среды (см. "[Вкладка Общие](#)"¹⁰⁵).

Вставить файл — вставка текста из указанного файла в текущую позицию курсора. Вызов функции осуществляет открытие стандартного диалогового окна для выбора файла.

Открыть файл — открытие файла, имя которого указано в позиции курсора конструкции `#include`.

Модифицирован — применение статуса, определяющего наличие изменений в файле. По умолчанию статус автоматически включается при изменении текста и учитывается для сохранения изменений.

Только чтение — применение атрибута, запрещающего изменять активный файл. Функция меняет атрибут не только в редакторе, но и в файловой системе.

3.2.2.1. Команда Специальная вставка

Команда **Специальная вставка** позволяет вставить текст из буфера обмена с использованием дополнительных параметров.

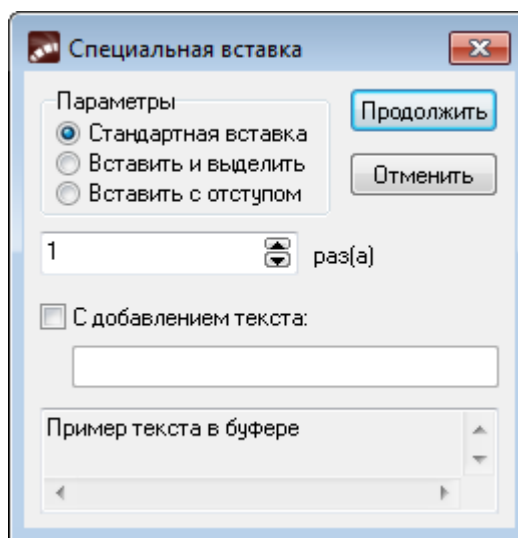


Рис. 8. Диалоговое окно команды Специальная вставка

Режим вставки определяется в диалоговом окне данной команды.

Стандартная вставка — вставка текста.

Вставить и выделить — вставка текста и выделение его в окне редактора.

Вставить с отступом — вставка текста с добавлением отступа.

Раз(a) — количество повторного выполнения операции вставки.


С добавлением текста — текст, добавляемый в начало буфера обмена. Текст, который на данный момент содержится в буфере обмена, отображается в нижней панели диалогового окна.

[[Продолжить](#)] — выполнить вставку текста в соответствии с указанными параметрами.

[[Отменить](#)] — отменить вызов команды специальной вставки.

3.2.2.2. Меню Сворачивание кода

В меню *Сворачивание кода* содержатся команды, позволяющие свернуть основные структуры языка **VIP** (интерфейсы, обработчики событий, процедуры, операторные скобки и другие, см. "[Сворачивание кода](#)¹¹⁸"), пользовательские регионы (см. "[Меню Правка](#)²⁸") и многострочные комментарии (см. "[Многострочные комментарии](#)³²") из любой позиции блока в активном окне редактора. При сворачивании основной структуры — дочерние сворачиваются автоматически.

В заголовке свернутого блока отображается значок , при наведении на который во всплывающей подсказке выводится содержимое данного блока.

Свернуть блок — сворачивание текущего блока кода или нескольких выделенных блоков.

Свернуть все блоки — сворачивание всех блоков.

Свернуть конструкции — сворачивание всех конструкций типа `cmEvent`, `Procedure/Function` или `Region`.

Свернуть до N уровня — сворачивание всех блоков до определенного уровня вложенности (см. "[Команда Свернуть до N уровня](#)³⁰").

Статус свернутых блоков сохраняется в проекте.

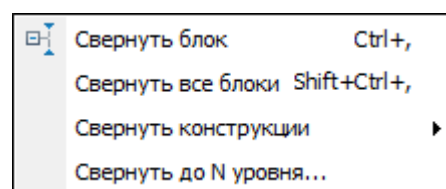


Рис. 9. Меню функций сворачивания кода

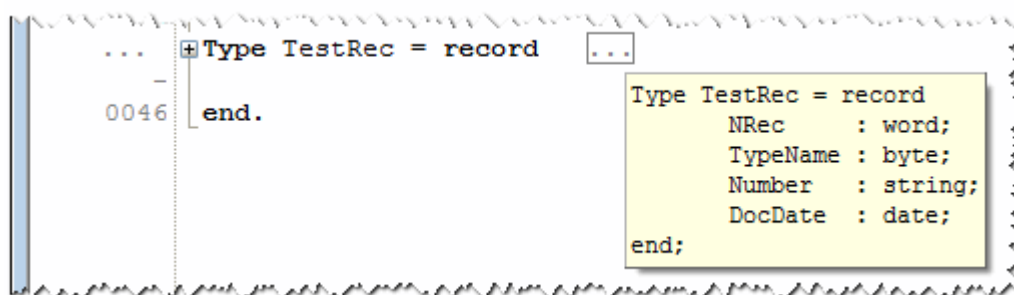


Рис. 10. Всплывающая подсказка свернутого блока

3.2.2.3. Команда Свернуть до N уровня

Команда **Свернуть до N уровня** выполняет сворачивание всех блоков в активном окне редактора до указанного уровня.

Уровень вложенности — глубина свернутых блоков. Блоки указанных уровней принимают развернутое состояние, остальные сворачиваются.

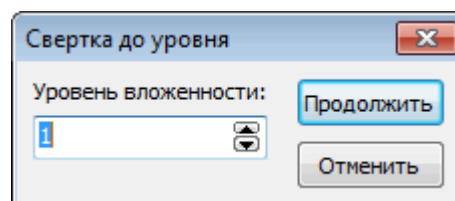


Рис. 11. Диалоговое окно сворачивания блоков

[Продолжить] — применить состояние ко всем блокам в соответствии с указанными параметрами.

[Отменить] — отменить вызов команды сворачивания блоков.

3.2.2.4. Меню Разворачивание кода

В меню *Разворачивание кода* содержатся команды, изменяющие состояние свернутых блоков в активном окне редактора. При редактировании заголовка свернутого блока производится его автоматическое разворачивание.

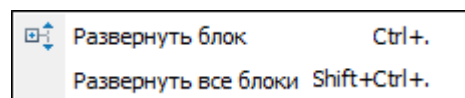


Рис. 12. Меню Разворачивание кода

Развернуть блок — разворачивание текущего блока кода или нескольких выделенных блоков.

Развернуть все блоки — разворачивание всех блоков.

3.2.2.5. Меню Форматирование

В меню *Форматирование* содержатся команды, позволяющие изменить в активном окне редактора кодировку символов и формат отступа между ними.

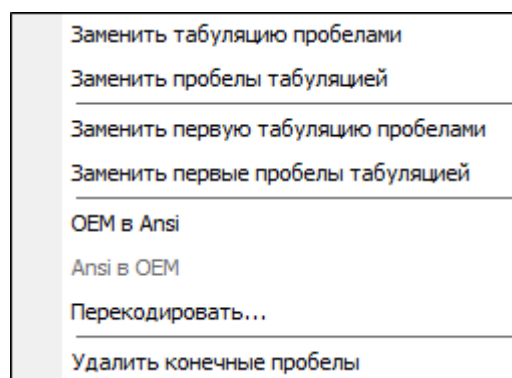


Рис. 13. Меню Форматирование

Заменить табуляцию пробелами — замена каждого имеющегося символа табуляции соответствующим количеством пробелов. Число пробелов, равное одной табуляции, определяет **Ширина табуляции** в настройках редактора (см. "[Вкладка Редактор](#)¹¹²").

Заменить пробелы табуляцией — замена каждого ряда пробелов символом табуляции. Описание ширины табуляции указано в предыдущей команде.

Заменить первую табуляцию пробелами — замена символов табуляции, расположенных в начале строки, заданным количеством пробелов.

Заменить первые пробелы табуляцией — замена каждого ряда пробелов, расположенных в начале строки, символом табуляции.

OEM в Ansi — перевод текста из OEM-кодировки в ANSI.

Ansi в OEM — перевод текста из ANSI-кодировки в OEM.

Перекодировать — изменение кодировки текста. Текущая и результирующая кодировки задаются в диалоговом окне, при необходимости выполнение данной команды можно отменить.

Удалить конечные пробелы — удаление пробелов, имеющих в конце строк.

3.2.2.6. Меню Изменение регистра

В меню *Изменение регистра* содержатся команды, выполняющие преобразование между строчными и заглавными символами в выделенном тексте.

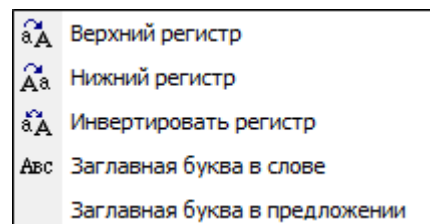


Рис. 14. Меню Изменение регистра

Верхний регистр — преобразование всех букв в прописные.

Нижний регистр — преобразование всех букв в строчные.

Инвертировать регистр — преобразование прописных букв в строчные, а строчных в прописные.

Заглавная буква в слове — преобразование первой буквы каждого слова в прописную, остальных в строчные.

Заглавная буква в предложении — преобразование первой буквы каждого предложения в прописную, остальных в строчные. В качестве конца предложения считается знак препинания (.?!,: и т. д.) и следующий за ним пробел, табуляция, конец строки и т. п.

3.2.2.7. Меню Блок

В меню *Блок* содержатся команды, позволяющие управлять комментариями, изменять размер отступа в выделенном блоке, а также сортировать выделенные строки в определенной последовательности и сохранять фрагмент текста из редактора во внешнем файле.

Добавить отступ — добавление знака пробела в начало выделенного блока или строки. Количество пробелов определяется в настройке *Ширина табуляции*. Добавить отступ можно с помощью клавиши **Tab**, для этого необходимо включить параметр *Использовать табуляцию* (см. "[Вкладка Редактор - Параметры](#)"^[116]).

Убрать отступ — уменьшение отступа в начале строки или выделенной области вертикального блока. Вызов функции по **Shift+Tab** возможен при включенном параметре *Использовать табуляцию*, как и в предыдущей функции.

Размер отступа для описанных выше команд определяется в настройке *Ширина табуляции* (см. "[Вкладка Редактор](#)"^[112]), независимо от статуса параметра *Использовать табуляцию*.

Комментировать/Раскомментировать — добавление/удаление символов комментария для выделенной области (вертикальный блок, часть строки) или текущей строки в соответствии с синтаксической схемой. Настройка синтаксиса комментариев выполняется в параметрах среды (см. "[Вкладка Atlantis VIP - Подсветка](#)"^[123]).

В закомментированной области выполняется функция *Раскомментировать*, в незакомментированной — *Комментировать*.

Функция добавляет символы однострочного комментария только к текущей строке или к нескольким строкам, если все они выделены целиком, а при выделенной части строки добавляются/удаляются символы многострочного комментария.

Сортировка — сортировка выделенных строк в соответствии с указанными критериями. Вызов команды осуществляет открытие диалогового окна с параметрами сортировки (подробнее см. в разделе "[Команда Сортировка](#)"^[33]).

Сохранить — сохранение текста выделенного блока в файл. Вызов команды осуществляет открытие стандартного диалогового окна для выбора файла.

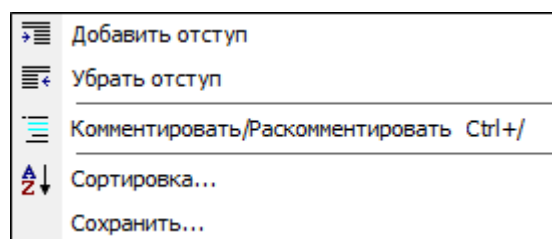


Рис. 15. Меню Блок

3.2.2.8. Команда Сортировка

Команда **Сортировка** выполняет упорядочение выделенных строк исходя из заданных условий.

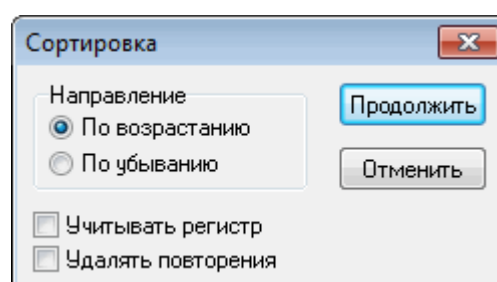


Рис. 16. Диалоговое окно Сортировка

Направление — порядок сортировки:

- **По возрастанию** — в порядке алфавита;
- **По убыванию** — в обратном порядке.

Учитывать регистр — выполнение сортировки с учетом разницы в размерах символов.

Удалять повторения — удаление повторных строк, в которых совпадает содержание. Регистр символов не учитывается.

[Продолжить] — выполнить сортировку строк.

[Отменить] — отменить вызов команды сортировки строк.

3.2.2.9. Меню Блок дополнительно

В меню **Блок дополнительно** содержатся команды, позволяющие оперативно работать с текстом на уровне строк и слов.

Удалить строки — удаление текущей строки или нескольких выделенных строк. Удаляется вся строка независимо от того, выделена она полностью или частично. При этом нижние строки перемещаются на соответствующий уровень выше.

Удалить текст до курсора — удаление текста от начала строки до позиции курсора. При этом текст после курсора перемещается в начало строки.

Удалить текст после курсора — удаление текста от позиции курсора до конца текущей строки.

Удалить слово — удаление слова, на котором установлен курсор.

Выделить строку — выделение строки (включая символ конца строки), в которой установлен курсор.

Выделить текст до курсора — выделение текста от начала текущей строки до позиции курсора.

Выделить текст после курсора — выделение текста от позиции курсора до конца текущей строки. Символ конца строки в выделение не входит.

Выделить слово — выделение слова, на котором установлен курсор.

Соединить строки — соединение выделенных строк в одну. При активизации функции открывается диалоговое окно с параметрами соединения строк (см. "[Команда Соединить строки](#)"³⁴).

Разорвать строку — разделение одной строки на две в позиции курсора. Новая строка размещается без отступа от левого края.

Дублировать строку — копирование выделенных строк или строки с курсором и выполнение их вставки со следующей строки.

Формат строки — изменение длины текущей строки или нескольких выделенных строк. Данная команда открывает диалоговое окно с параметрами ограничения строки (см. "[Команда Формат строки](#)"³⁵).

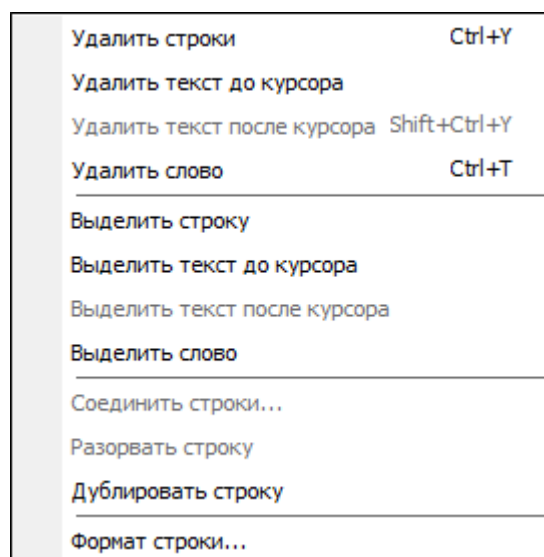


Рис. 17. Меню Блок дополнительно


3.2.2.10. Команда Соединить строки

Команда **Соединить строки** позволяет соединить выделенные строки или элементы вертикального блока в одну строку. При этом символы конца строк удаляются.

Дополнительные параметры соединения задаются в диалоговом окне.

Варианты соединения:

- Несколько выделенных строк соединяются целиком независимо от границ выделения. При этом нижние строки перемещаются на соответствующий уровень выше.
- Элементы вертикального блока перемещаются в позицию начала выделения, последовательно соединяясь одной строкой. При этом текст, расположенный справа от позиции курсора, сдвигается.

Разделитель — символ, который устанавливается в позиции соединения строк. Кнопка  [Выбор] открывает список метасимволов, используемых для построения регулярного выражения. Подробнее о регулярных выражениях см. в разделе "[Использование регулярных выражений](#)"¹²⁷.

Параметр *Использовать управляющую последовательность* автоматически включается при выборе макроопределения из списка (при необходимости его можно отключить) и указывает на то, что разделитель состоит из метасимволов.

[Продолжить] — выполнить соединение строк.

[Отменить] — отменить вызов команды соединения строк.

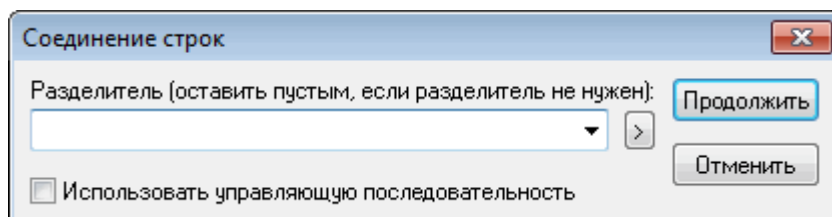


Рис. 18. Диалоговое окно Соединение строк

3.2.2.11. Команда Формат строки

Команда *Формат строки* изменяет размер выделенных строк или строки с курсором в текущем окне редактора. Параметры форматирования строк задаются в диалоговом окне.

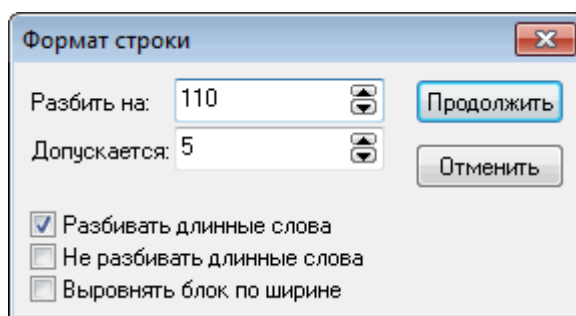


Рис. 19. Диалоговое окно Формат строки

Разбить на — максимальное количество символов в строке. При этом текст, расположенный за пределами данной границы, будет перемещен на следующую строку с учетом параметров переноса слов (минимальное количество символов 2).

Допускается — количество символов, которое допускается использовать для завершения слова при выходе за границу форматирования.

Разбивать длинные слова — часть слова, которая выходит за дополнительную границу, переносить на следующую строку.

Не разбивать длинные слова — слово, которое пересекает дополнительную границу, переносить на следующую строку целиком.

Выровнять блок по ширине — одновременное выравнивание содержимого строк по левому и правому краю за счет добавления пробелов между словами.

[Продолжить] — выполнить команду форматирования строк в соответствии с указанными параметрами.

[Отменить] — отменить вызов команды форматирования строки.

3.2.2.12. Команда Орфография

Команда *Орфография* выполняет грамматическую проверку в активном редакторе. При этом первое слово после курсора, отсутствующее в словаре, выделяется, после чего открывается диалоговое окно. В открывшемся окне содержится проверяемое слово и список предлагаемых вариантов замены.

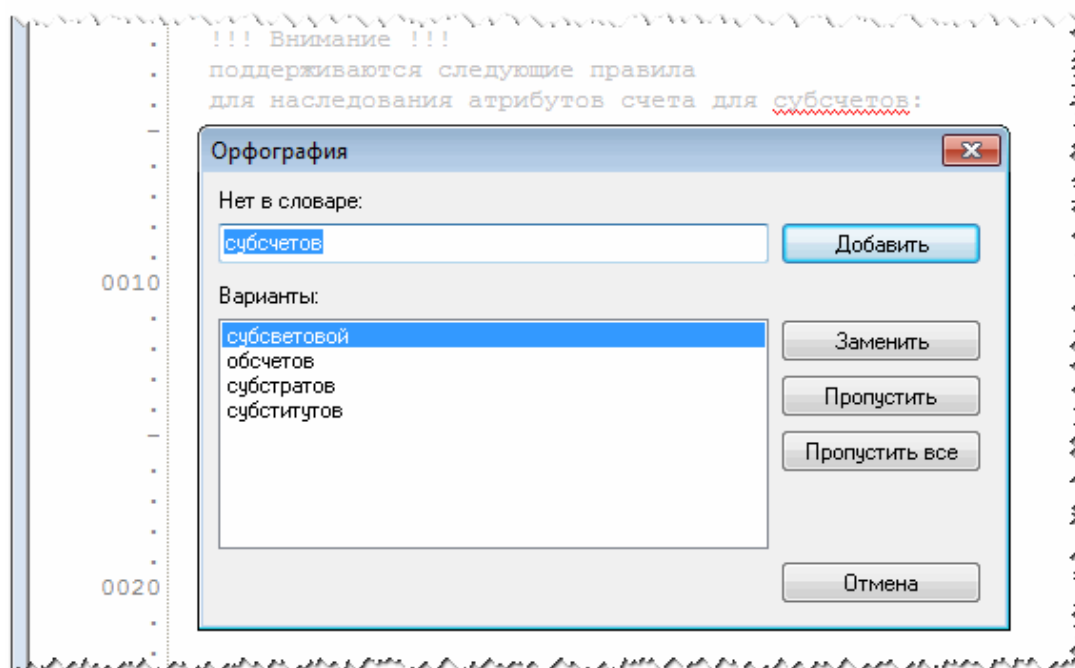


Рис. 20. Диалоговое окно Орфография

Нет в словаре — текущее слово, которое отсутствует в словаре.

Файл пользовательского словаря **User.dic** размещается в каталоге <системный диск>:\Documents and Settings\<имя пользователя>\Application Data\Viper.

[**Добавить**] — добавление текущего слова в пользовательский словарь, используемый в процессе проверки.

[**Заменить**] — замена текущего слова выбранным **Вариантом** из списка или отредактированным в поле **Нет в словаре**, при этом заменяемое слово, отсутствующее в словаре, автоматически добавится в него. В момент редактирования слова активизируется кнопка [**Отменить правку**], которая остается доступной до осуществления замены.

Варианты — однокоренные слова и словоформы, которые содержатся в словаре.

[**Пропустить**] — оставление текущего слова без изменений и участия в дальнейших проверках активного редактора.

[**Пропустить все**] — аналогично [**Пропустить**], но применяется одновременно ко всем ошибкам активного редактора.

[**Отмена**] — закрыть окно =*Орфография*= и прекратить проверку.

Дополнительная возможность исправления грамматических ошибок доступна в пункте *Орфография* контекстного меню слова с ошибкой (см. "[Сервис](#)"⁷³).

Использование функции проверки орфографии возможно только с включенным параметром **Подчеркивать слова с ошибками** (см. "[Вкладка Редактор - Проверка правописания](#)"¹¹⁵). *Орфография* доступна для активного редактора (см. "[Меню Правка](#)"²⁷) при снятом атрибуте *Только чтение* и наличии в редакторе грамматической ошибки.

3.2.3. Меню Поиск

Меню *Поиск* содержит команды поиска и замены, а также навигации по тексту открытых файлов.

Найти — поиск соответствий в активном файле. При активизации функции открывается диалоговое окно с параметрами поиска (см. "[Команда Найти](#)"³⁸).

Искать далее — поиск следующего соответствия в активном файле, при этом используются параметры последнего поиска по направлению от курсора к концу файла.

Искать назад — поиск предыдущего соответствия в активном файле, при этом используются параметры последнего поиска по направлению от курсора к началу файла.

Искать слово далее — поиск следующего соответствия выделенному слову в активном файле. При отсутствии выделения используется текущее слово. Поиск выполняется по направлению от текущего выделения к концу файла, при этом диалоговое окно параметров поиска не открывается, но учитываются *Параметры* последней выполненной команды *Найти*. История поиска сохраняется в диалоговом окне функции *Найти*.

Искать слово назад — поиск предыдущего соответствия выделенному слову в активном файле. При отсутствии выделения используется текущее слово. Поиск выполняется по направлению от текущего выделения к началу файла, при этом учитываются параметры, описанные в команде *Искать слово далее*.

Инкрементальный поиск вперед — переключатель инкрементального поиска. Поиск выполняется по мере ввода текста, при этом в строке статуса включается пометка *Поиск* >:, после которой отображается набранный текст. В процессе поиска в редакторе помечаются все соответствия по направлению от курсора к концу файла, текущим выделяется первое соответствие от курсора. Для выхода из режима инкрементального поиска предназначена клавиша **Esc** или **Enter**.

Инкрементальный поиск обратно — принцип выполнения аналогичен предыдущей функции, при этом в строке статуса включается пометка *Поиск* <: и поиск соответствия выполняется по направлению от курсора к началу файла.

Найти в файлах — поиск в файлах. При активизации функции открывается диалоговое окно с параметрами поиска (см. "[Команда Найти в файлах](#)"³⁹).

Заменить — замена текста. Вызов команды осуществляет открытие диалогового окна с параметрами замены (см. "[Команда Заменить](#)"⁴¹).

Навигация по коду — группа команд оперативного перемещения курсора по элементам структуры кода (см. "[Меню Навигация по коду](#)"⁴³).

Перейти назад в окне вывода — переход к предыдущей записи активной вкладки =[Окна вывода](#)⁶⁸=.

Перейти дальше в окне вывода — переход к следующей записи активной вкладки =[Окна вывода](#)⁶⁸=.

Установка закладки — управление нумерованными закладками, которые используются для пометки строки.

В группу команды входит список из 10 закладок, обозначенных номерами от 0 до 9. При выборе свободной закладки из списка она устанавливается в служебном поле текущей строки. В случае, если закладка занята, то она перемещается в текущую строку или снимается с нее.

Запись об установленной закладке отображается в окне =[Метки](#)⁶⁴= с информацией об имени файла, приоритете, порядковом номере закладки и текущей строке, а в качестве комментария добавляется содержимое текущей строки. Информация о закладках сохраняется в файле проекта.

Команда *Удалить закладки* снимает все закладки в активном редакторе.

Переход к закладке — перемещение курсора к строке с указанной закладкой.

В группу команды входит список из 10 закладок, обозначенных номерами от 0 до 9. Выбор соответствующей закладки из списка активизирует перемещение курсора к строке с указанной закладкой.

Метка — управление объектами, которые используются для пометки строки (см. "[Меню Метка](#)"⁴⁴).

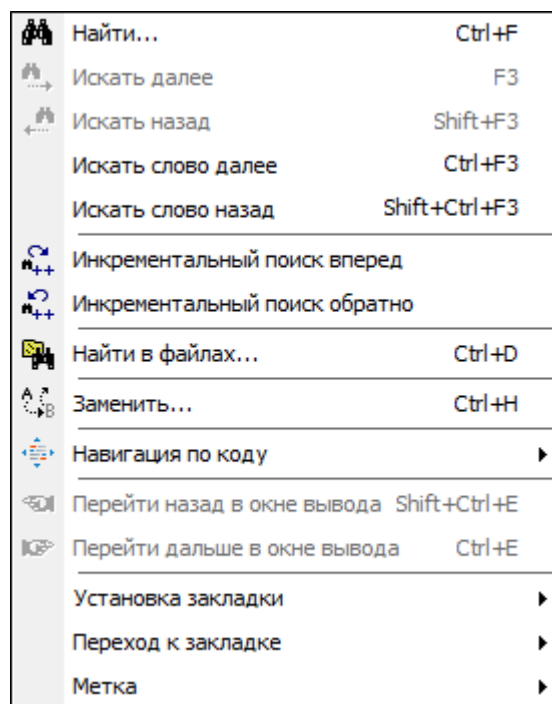



Рис. 21. Меню Поиск

3.2.3.1. Команда Найти

Команда *Найти* позволяет выполнить поиск текста с дополнительными параметрами в активном редакторе. После выбора команды открывается диалоговое окно с параметрами поиска.

Найти — текст поиска. По умолчанию используется выделенный фрагмент в активном редакторе, текст под курсором или последнее найденное соответствие. Кнопка  открывает список метасимволов, используемых для построения регулярного выражения (см. "[Использование регулярных выражений](#)"¹²⁷). История поиска текста сохраняется в выпадающем списке.

[Дополнительно] — отображение панели дополнительных параметров.

Параметры — опции поиска:

- **Учитывать регистр** — поиск текста с учетом регистра символов.
- **Только слово целиком** — поиск полного слова.
- **Регулярное выражение** — использование для поиска регулярного выражения. Данный параметр включается автоматически при выборе макроопределения из списка.

Начинать — начальная позиция для выполнения функции поиска.

- **С начала** — поиск с первой строки редактора.
- **От курсора** — поиск с позиции курсора в редакторе.

Направление — направление поиска:

- **Вперед** — выполнение поиска от начала файла (курсора, при использовании параметра *От курсора*) и до конца.

- **Назад** — выполнение поиска от конца файла (курсора, при использовании параметра **От курсора**) к его началу.

Границы — область поиска.

- **По всему файлу** — поиск по текущему редактору.
- **В выделенном тексте** — поиск в границах выделенного блока.

[**Продолжить**] — выполнить поиск по указанным параметрам.

При этом первое найденное соответствие выделяется текущим, остальные подсвечиваются по всему файлу. Возможность изменения **Подсветки при поиске** доступна в параметрах среды (см. "[Вкладка Редактор - Цвет](#)"¹¹⁴).

Параметры поиска сохраняются для использования в следующих вызовах данной функции.

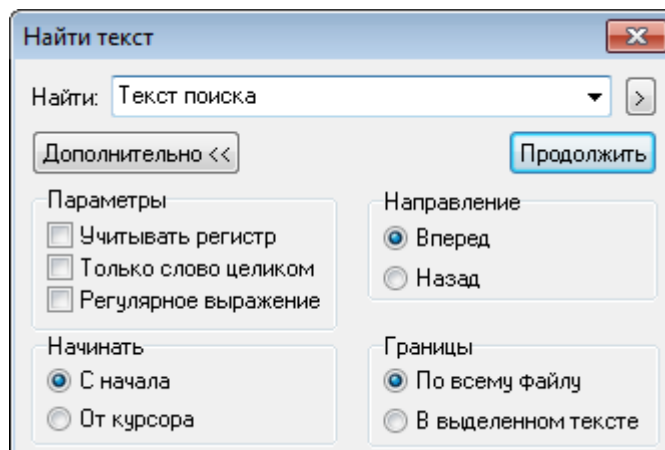


Рис. 22. Диалоговое окно параметров поиска в активном файле

3.2.3.2. Команда **Найти в файлах**

Команда **Найти в файлах** позволяет выполнить поиск текста по файлам. При активации функции открывается диалоговое окно с параметрами поиска.

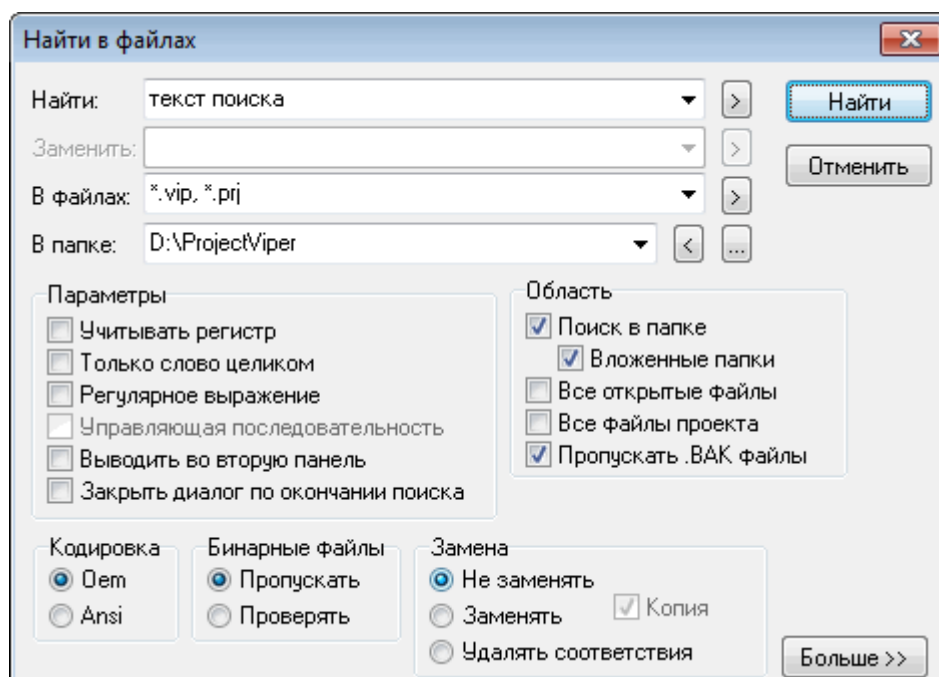







Рис. 23. Диалоговое окно параметров поиска в файлах

Найти — текст поиска. По умолчанию используется выделенный фрагмент в активном файле, текст под курсором или последнее найденное соответствие. Кнопка  раскрывает список метасимволов, используемых для построения регулярного выражения (см. "[Использование регулярных выражений](#)"¹²⁷").

Заменить — текст для замены найденных соответствий. Нетекстовые файлы будут автоматически пропущены. При нажатии кнопки  раскрывается список допустимых управляющих последовательностей.

В файлах — шаблон расширений файлов, допускаемых для поиска. Скрытые файлы в поиске не участвуют. При нажатии кнопки  раскрывается меню типовых наборов шаблонов.

В папке — каталог поиска. Скрытые каталоги в поиске не участвуют. Для выбора необходимого каталога предназначена кнопка .

Она открывает окно со списком каталогов. С помощью кнопки  автоматически добавляется адрес активного файла.

Параметры — опции поиска:

- **Учитывать регистр** — поиск текста с учетом регистра символов.
- **Только слово целиком** — поиск полного слова.
- **Регулярное выражение** — использование для поиска регулярного выражения. Данный параметр включается автоматически при выборе макроопределения из списка.
- **Управляющая последовательность** — использование управляющих последовательностей (`\\`, `\\n`, `\\t`, `\\f`, `\\r`) для замены.
- **Выводить во вторую панель** — вывод списка найденных соответствий в «Окне вывода» на вкладке Поиск в файлах 2. По умолчанию используется вкладка Поиск в файлах 1.
- **Закрыть диалог по окончании поиска** — скрытие окна с процессом поиска после его завершения.

Область — установить границы поиска файлов:

- **Поиск в папке** — поиск в каталоге из поля **В папке**.
 - **Вложенные папки** — поиск во вложенных папках.
- **Все открытые файлы** — поиск в файлах, открытых в редакторе, при этом поиск производится по содержимому в редакторе.
- **Все файлы проекта** — поиск в файлах текущего проекта.
- **Пропускать .BAK файлы** — исключение из поиска файлов с расширением *bak*.

Кодировка — кодировка текста поиска:

- **Oem** — поиск соответствия в кодировке OEM.
- **Ansi** — поиск соответствия в кодировке ANSI.

Бинарные файлы — обработка двоичных файлов:

- **Пропускать** — исключение из поиска двоичных файлов.
- **Проверять** — обработка двоичных файлов.

Замена — режим изменения найденных соответствий:

- **Не заменять** — поиск.
- **Заменять** — поиск и замена. В результате выполнения замены несохраненные изменения в редакторе остаются, и статус такого файла не изменяется.

- **Удалить соответствия** — удаление найденных соответствий в файлах.





Параметр **Копия** позволяет в случае замены или удалении соответствий сохранить копию изначального варианта файла.

[Найти] — выполнить поиск по указанным параметрам.

[Отменить] — закрыть окно параметров поиска в файлах.

[Больше] — дополнительный список каталогов для поиска.

Команды управления списком:

-  — дополнительные каталоги поиска. Вызов команды осуществляет открытие стандартного диалогового окна для выбора каталога из древовидного списка;
-  — удаление текущей записи из списка;
-  — перемещение текущей записи вверх на одну позицию;
-  — перемещение текущей записи вниз на одну позицию.

Свернуть панель можно с помощью кнопки [Меньше].

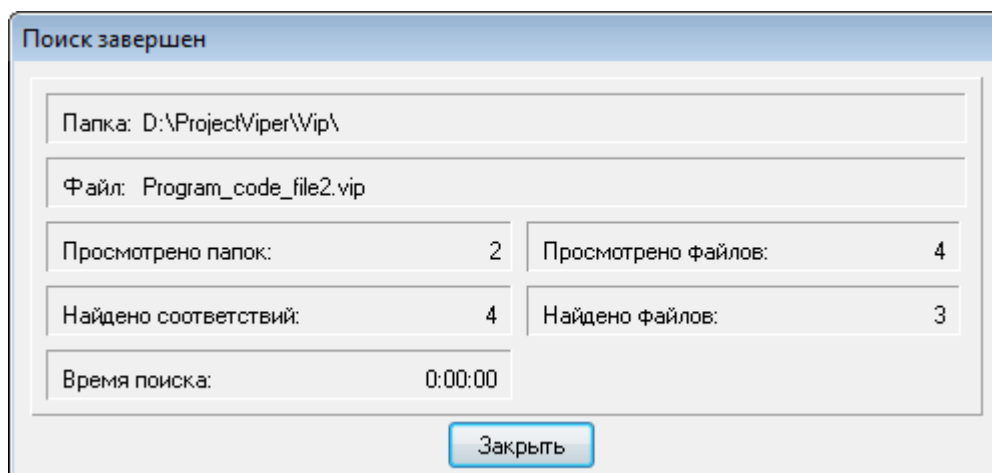


Рис. 24. Информационное окно процесса поиска в файлах

В процессе поиска отображается информационное окно с количеством просмотренных каталогов, файлов и найденных соответствий (см. рис. "[Информационное окно процесса поиска в файлах](#)"⁴¹).


[Остановить] — завершить поиск, не дожидаясь его окончания.


[Закрыть] — закрыть информационное окно. Команда доступна после завершения поиска.

Положительный результат поиска отображается в «Окне вывода» (см. рис. "[Вкладка результатов поиска в файлах](#)"⁶⁸).

3.2.3.3. Команда Заменить

Команда **Заменить** позволяет найти соответствия в активном файле и автоматически заменить их. После выбора команды открывается диалоговое окно параметров замены текста.

Найти — текст поиска. По умолчанию используется выделенный фрагмент в активном файле, текст под курсором или последнее найденное соответствие. Кнопка  раскрывает список метасимволов, используемых для построения регулярного выражения (см. "[Использование регулярных выражений](#)"¹²⁷).

Заменить — текст для замены найденных соответствий. Нетекстовые файлы будут автоматически пропущены. Кнопка  открывает список допустимых управляющих последовательностей.

[**Дополнительно**] — отображение панели дополнительных параметров.

[**Заменить все**] — найти и заменить все соответствия.

[**Заменить**] — начать поиск соответствий. При отсутствии соответствий отобразится соответствующее сообщение, при нахождении первого соответствия откроется диалоговое окно для подтверждения замены.

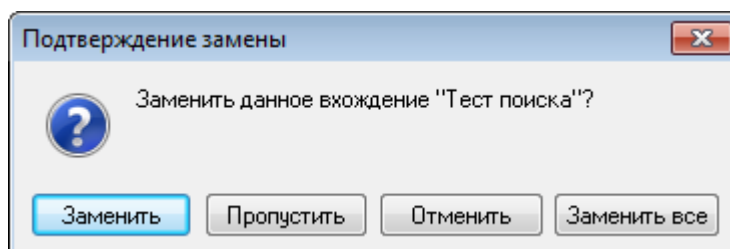


Рис. 25. Диалоговое окно подтверждения замены

Функции окна подтверждения замены:

- [**Заменить**] — заменить текущее соответствие.
- [**Пропустить**] — не заменять текущее соответствие и перейти к следующему.
- [**Отменить**] — закрыть окно подтверждения замены и остановить выполнение функции *Заменить*. Также для отмены можно нажать клавишу **Esc**.
- [**Заменить все**] — заменить все найденные соответствия в активном файле.

Параметры — опции поиска:

- **Учитывать регистр** — поиск текста с учетом регистра символов.
- **Только слово целиком** — поиск полного слова.
- **Регулярное выражение** — использование для поиска регулярных выражений. По умолчанию параметр включается при выборе макроопределения из списка.
- **Управляющая последовательность** — использование управляющих последовательностей (`\\`, `\\n`, `\\t`) для замены.
- **Вывести количество** — вывод сообщения о количестве замен после завершения команды [**Заменить все**].

Начинать — начальная позиция для выполнения функции поиска и замены.

- **С начала** — поиск с первой строки редактора.
- **От курсора** — поиск с текущего слова под курсором.

Направление — направление поиска:

- **Вперед** — выполнение поиска от начала файла (курсора, при использовании параметра *От курсора*) и до конца.
- **Назад** — выполнение поиска от конца файла (курсора, при использовании параметра *От курсора*) к его началу.

Границы — область поиска.

- **По всему файлу** — поиск по текущему редактору.
- **В выделенном тексте** — поиск в границах выделенного блока текущего редактора.

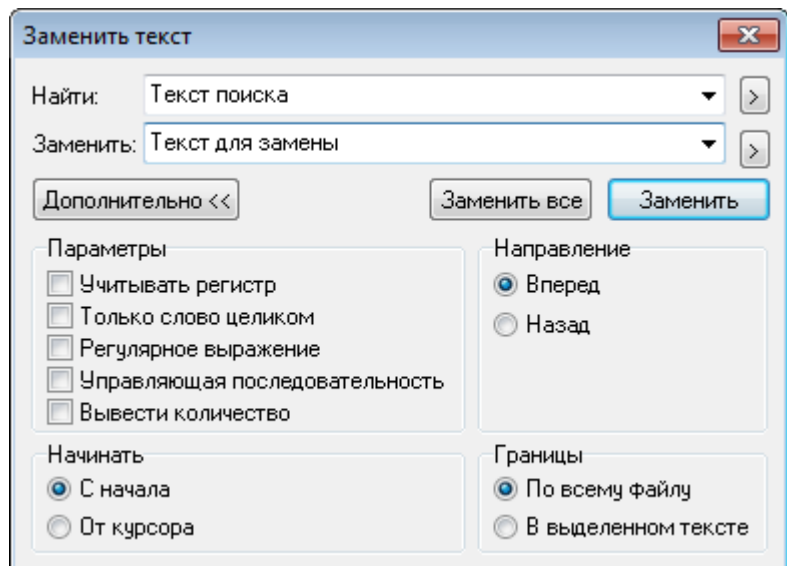


Рис. 26. Диалоговое окно параметров замены в активном файле

3.2.3.4. Меню Навигация по коду

В меню *Навигация по коду* содержатся команды навигации по структуре кода, парным элементам, измененным позициям, а также для перехода к курсору или установке его в определенной позиции.

Перейти к курсору	Shift+Ctrl+F5
Перейти...	Ctrl+G
Парная скобка	Shift+Ctrl+B
Парная конструкция	Alt+[
Позиция в окне структуры	Alt+F
Перейти к последнему изменению	Shift+Ctrl+G
Перейти к предыдущему изменению	Shift+F5
Перейти к следующему изменению	Shift+F6

Рис. 27. Меню Навигация по коду

Перейти к курсору — прокрутка строк активного файла таким образом, чтобы текущая позиция курсора оказалась в пределах видимости.

Перейти — перевод курсора на заданную позицию в активном файле. Данная команда открывает диалоговое окно с параметрами перехода (см. "[Команда Перейти](#)"⁴⁴).

Парная скобка — перевод курсора к символу, который является парным для текущего.

Парная конструкция — перевод курсора в начало или в конец программной конструкции. При этом фон блока данной конструкции выделяется. Изменить цвет выделения можно в параметрах редактора (см. "[Вкладка Редактор - Цвет](#)"¹¹⁴). Команда предназначена для основных конструкций языка **VIP** и многострочных комментариев.

Позиция в окне структуры — перевод курсора к соответствующей программной конструкции в списке структуры кода (см. "[Окно структуры кода](#)"⁶¹). Заголовок данной структуры выделяется.

Перейти к последнему изменению — перевод курсора к позиции последнего изменения в активном файле.

Перейти к предыдущему изменению — перевод курсора к предыдущей отредактированной строке.

Перейти к следующему изменению — перевод курсора к следующей отредактированной строке.

3.2.3.5. Команда *Перейти*

Команда *Перейти* выполняет перемещение курсора в первую позицию указанной строки.

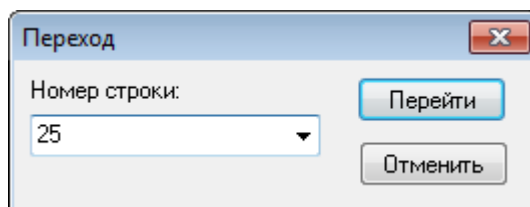


Рис. 28. Диалоговое окно параметров перехода

Вызов команды осуществляет открытие диалогового окна, в котором следует добавить необходимый **Номер строки**. История последних 10 перемещений сохраняется в выпадающем списке.

[Перейти] — переместить курсор в начало соответствующей строки.

[Отменить] — закрыть окно перехода без выполнения данной команды.

3.2.3.6. Меню *Метка*

В меню *Метка* содержатся команды, управляющие метками строк.

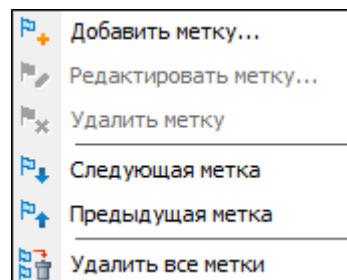


Рис. 29. Меню *Метка*

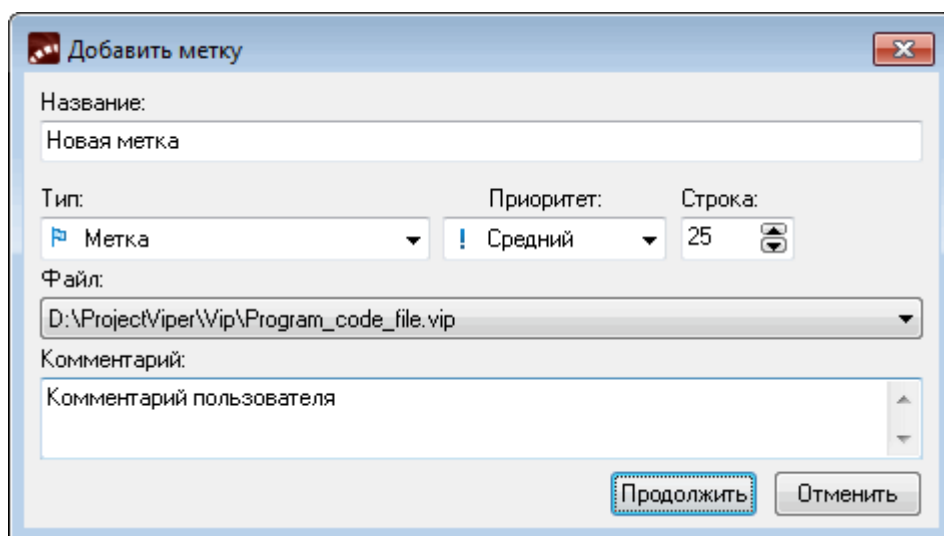


Рис. 30. Диалоговое окно добавления метки

Используемые метки активного файла отображаются в служебном поле редактора, в неограниченном количестве для каждой строки. Запись об установленной метке помещается в соответствующее окно (см. "[Окно меток](#)"⁶⁴) и сохраняется в файле проекта (*.vpr).

Добавить метку — добавление метки для строки. После выбора команды открывается диалоговое окно, в котором задаются параметры для создаваемой метки.

Название — имя метки. По умолчанию отображается выделенный текст или слово в позиции курсора, при необходимости название можно изменить.

Тип — тип метки. По умолчанию установлен тип *Метка*, при необходимости его можно изменить с помощью выпадающего списка.

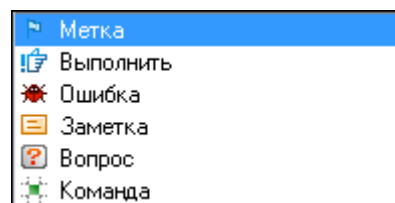


Рис. 31. Типы меток

Приоритет — степень значимости. По умолчанию установлен *Средний* приоритет, при необходимости его можно изменить с помощью выпадающего списка.

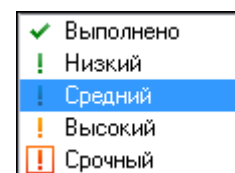


Рис. 32. Приоритеты меток

Строка — номер строки для размещения метки. По умолчанию используется текущая строка.

Файл — файл для размещения метки. По умолчанию установлен адрес активного файла, при необходимости выбрать другой открытый в редакторе файл можно с помощью выпадающего списка. В команде *Редактировать метку* данный параметр заблокирован от изменений.

Комментарий — описание метки. По умолчанию отображается содержимое текущей строки.

[[Продолжить](#)] — установить метку в служебное поле редактора в соответствии с указанными параметрами.

[[Отменить](#)] — закрыть окно создания метки без установки метки.

Редактировать метку — изменение параметров текущей метки. При выборе данной команды курсор должен быть установлен в строке с меткой. Команда вызывает диалоговое окно аналогичное окну из команды добавления метки.

Удалить метку — удалить текущую метку. При выборе данной команды курсор должен быть установлен в строке с меткой.

Следующая метка — перевод курсора к строке со следующей меткой по направлению к концу файла.

Предыдущая метка — перевод курсора к строке с предыдущей меткой по направлению к началу файла.

Удалить все метки — удаление всех меток активного файла.

3.2.4. Меню Вид

Меню *Вид* содержит команды управления видимой частью среды разработки. Они позволяют изменить состав отображаемых окон и их настройку.

Команды, управляющие наличием [инструментальных окон](#)^[57].

- *Окно менеджера проекта*;
- *Окно вывода*;
- *Окно сообщений*;
- *Окно структуры кода*;
- *Проводник*;
- *Окно меток*;
- *Окно структуры ресурсов*;
- *Окно дерева подключений*.

Синтаксическая схема — выбор синтаксической схемы для подсветки текста в редакторе. Для программирования на языке **VIP** предназначена схема *Atlantis VIP*. О ее настройке см. в разделе "[Вкладка Atlantis VIP — Подсветка](#)"^[123].

Для открываемых файлов текущая синтаксическая схема устанавливается автоматически, исходя из расширения файла.

Автоматическое определение синтаксической схемы можно настроить на вкладке *Синтаксис* в функции главного меню *Вид > Параметры*.

Полноэкранный режим — переключение рабочей области в полноэкранный режим. Для возврата в нормальный режим можно использовать горячую клавишу (по умолчанию **F11**). Автоматический выход из полноэкранного режима происходит при вызове функций, не связанных с редактором кода, в том числе функции компиляции и отладки.

Увеличить масштаб и *Уменьшить масштаб* — изменение масштаба редактора кода, выполняется также комбинацией **Ctrl+<колесико мышки>**. Текущий размер шрифта отражается в параметрах среды (см "[Вкладка Редактор](#)"^[112]).

Номера строк — отображение порядкового номера строки в служебном поле редактора кода. Причем, постоянно отображаются только круглые порядковые номера строк и номер текущей строки.

Непечатаемые символы — отображение непечатаемых символов, таких как пробел и табулятор. Символ табуляция в OEM-кодировке представлен точкой, в ANSI-кодировке — в виде стрелки.

Парная скобка — автоматическое добавление парной закрывающей скобки при вводе открывающей.

Подсветка скобок — подсветка соответствующей пары скобок при позиционировании курсора на одной из них.

Проверка орфографии — автоматическая проверка орфографии. При этом слово с ошибкой в окне редактора выделяется подчеркиванием.

Проверке подвергается слово, которое содержит символ русского алфавита. Ошибочным является слово, которое содержит грамматическую ошибку или другие символы в сочетании с русскими буквами. В контекстном меню слова содержатся варианты для замены, а также возможность *Добавить в словарь* или *Пропустить* (без добавления в словарь).

Параметры — открытие окна параметров редактора (см. "[Настройка параметров среды](#)"¹⁰⁵).

Панель вкладок — отображение панели с вкладками открытых файлов (см. "[Окно вкладок](#)"⁷⁹).

Статус-строка — отображение [статус-строки](#)⁸⁰.

Панели инструментов — отображение инструментальных панелей (см. "[Меню Панели инструментов](#)"⁴⁷).

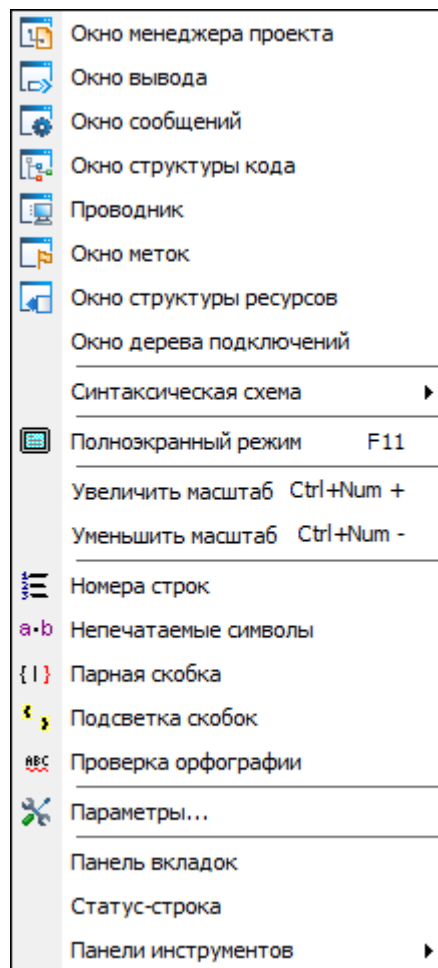


Рис. 33. Меню Вид

3.2.4.1. Меню Панели инструментов

В меню **Панели инструментов** содержится список инструментальных панелей и настройка для вызова команд.

С помощью переключателей, расположенных в левой части списка, можно управлять отображением инструментальных панелей. Подробное описание см. в разделе "[Инструментальные панели](#)"⁵⁵.

Настройка — управление элементами инструментальной панели и настройка комбинаций "горячих" клавиш для вызова команд меню. При активизировании команды открывается диалоговое окно с параметрами (см. "[Настройка инструментальных панелей и горячих клавиш](#)"⁴⁸).

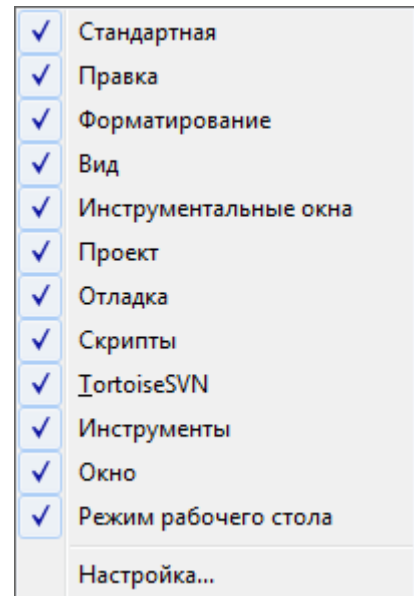


Рис. 34. Меню Панели инструментов

3.2.4.2. Настройка инструментальных панелей и горячих клавиш

Пункт меню *Настройка* позволяет управлять набором инструментальной панели и назначать параметры для вызова команд с помощью клавиатуры или мыши.

Вызов команды осуществляет открытие диалогового окна, которое содержит следующие вкладки:

- *Панель инструментов* — список панелей инструментов, соответствующих пунктам главного меню *Вид* (см. "[Меню Панели инструментов](#)"⁴⁷).

Панель *Режим рабочего стола* позволяет пользователю выбрать один из режимов, которые автоматически меняют настройки окон.

- *Клавиатура* — настройки "горячих" клавиш для вызова команд. Список *Категория* аналогичен [меню верхнего уровня](#)²⁴ приложения *Viper* и содержит полный список соответствующих команд.

Категория — группы команд среды *Viper*.

Команды — список команд, соответствующих выбранной *Категории*, для назначения "горячих" клавиш.

Ключ — комбинация клавиш.

Описание — подробная информация о текущей команде.

[Удалить ключ] — удалить комбинацию "горячих" клавиш текущей команды.

[Назначить] — связать комбинацию клавиш из поля *Ключ* с текущей *Командой*. При повторном назначении комбинации клавиш отображается соответствующее предупреждение с именем команды, которой они были назначены ранее.

Для управления клавишами вызова следует выбрать необходимую Категорию и *Команду*.

Добавить желаемую комбинацию можно, установив курсор в поле *Ключ* и нажав соответствующие клавиши на клавиатуре.

[Удалить ключ] можно с помощью соответствующей кнопки. Завершить настройку текущей команды следует подтверждением с помощью кнопки [Назначить].

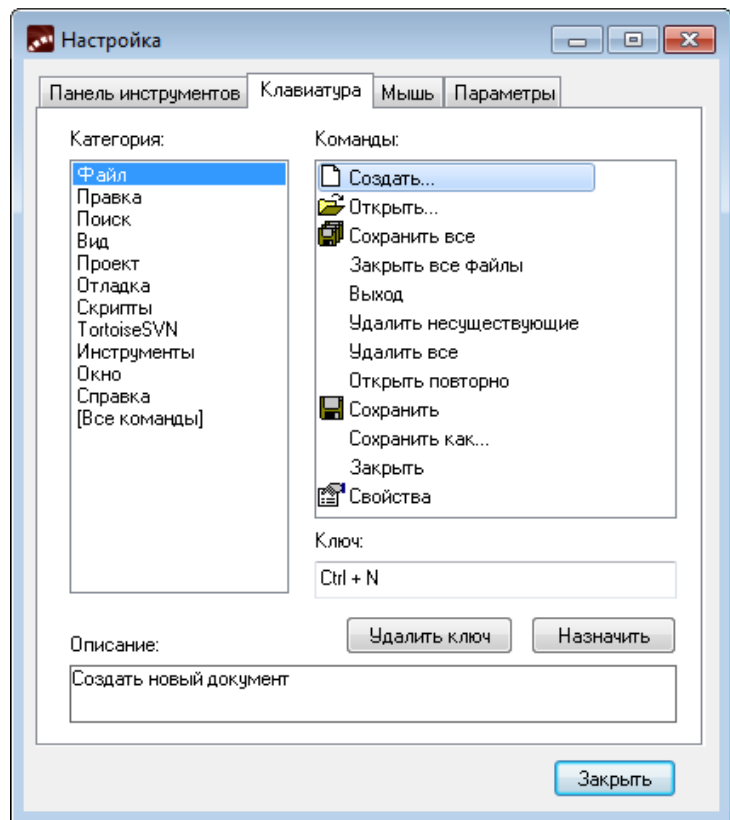


Рис. 35. Настройка горячих клавиш

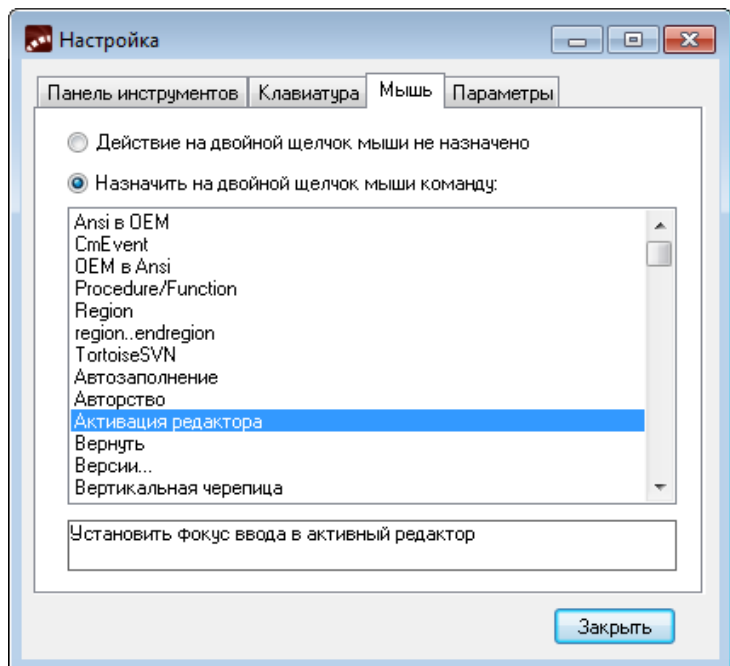


Рис. 36. Настройка для двойного щелчка мыши

- **Мышь** — назначение действия для двойного щелчка мыши. По умолчанию *Действие на двойной щелчок мыши не назначено*.

Для указания действия необходимо *Назначить на двойной щелчок мыши команду* и выбрать команду из списка.

- **Параметры** — настройки вывода подсказок для элементов инструментальной панели:
 - **Отображать подсказку в панели инструментов** — вывод всплывающей подсказки инструмента при наведении на него мышкой.
 - **Отображать комбинацию клавиш в панели инструментов** — вывод соответствующей комбинации клавиш во всплывающей подсказке. Выполняется при включенном параметре **Отображать подсказку в панели инструментов**.


[Заккрыть] — закрыть диалоговое окно настроек с сохранением измененных параметров.

3.2.5. Меню Проект

В меню *Проект* содержится набор команд для управления проектом и процессом компиляции.

Новый — создание нового проекта. Вызов команды открывает диалоговое окно для сохранения файла нового проекта (*.vpr). При выполнении данной команды предоставляется возможность использования [мастера настройки проекта](#)^[82].

Открыть — открытие существующего проекта. При этом установленный атрибут файла *Только чтение* автоматически снимается на уровне файловой системы.

 В среде **Viper**, начиная с версии 5.5.11, структура проекта изменена. Поэтому при открытии проекта, созданного в одной из предыдущих версий **Viper**, предлагается возможность конвертировать проект в новый формат.

Последние проекты — отображение списка истории последних используемых проектов. Максимальное количество элементов в списке можно задать в параметрах редактора (см. "[Настройка параметров среды](#)"^[105]).

Сохранить — сохранение изменений в файле проекта.

Сохранить как — сохранение проекта в новом файле. Вызов команды открывает стандартное окно для сохранения файла.

Заккрыть — закрытие проекта. Для измененных файлов проекта формируется запрос на их сохранение.

Команды *Компилировать проект* и *Пересобрать проект* последовательно компилируют (выполняют) все элементы сборки проекта (см. "[Состав проекта](#)"^[85]).

При выборе команды *Компилировать проект* в зоне действия **#make** компилируются только измененные файлы, а по команде *Пересобрать проект* компилируются все файлы. Для профилей вызова внешних компиляторов разницы между командами *Компилировать проект* и *Пересобрать проект* нет.

Компилировать файл/Выполнить профиль — компиляция (выполнение) текущего элемента сборки. Назначение элемента *текущим* осуществляется в контекстном меню *=Менеджера проекта=*.

Остановить компиляцию/выполнение профиля — прерывание процесса компиляции или выполнения профиля.

Отключиться от БД — отключение от БД. Подключение к БД происходит при запуске компиляции. В процессе компиляции команда недоступна.

Параметры проекта — настройка компиляции и отладки текущего проекта. При запуске команды открывается диалоговое окно с настройками проекта (см. "[Окно Параметры проекта](#)"^[87]).

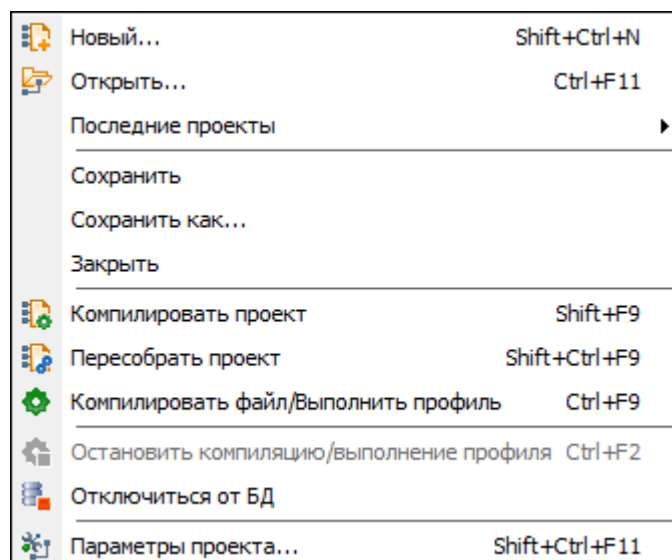


Рис. 37. Меню Проект

3.2.6. Меню Отладка

В меню *Отладка* содержится набор команд для управления процессом отладки приложения, загруженными интерфейсами, просмотра значений переменных и стека вызовов методов.

Начать отладку — управление процессом отладки. Команда используется в двух режимах:

- когда отладка отключена, данная команда осуществляет запуск процесса отладки (см. "[Отладка программ](#)"¹³²) для текущей конфигурации. Выбор конфигурации осуществляется на инструментальной панели *Отладка*. Добавление конфигураций осуществляется в главном меню *Проект > Параметры проекта* (см. "[Вкладка Отладчик VIP](#)"¹⁰⁰). При необходимости можно включить настройку *Автоматически запускать отладку после компиляции* в параметрах проекта.

Если на момент выполнения данной команды приложение уже открыто, то появится диалоговое окно для подтверждения повторного запуска: по кнопке **[Да]** — приложение откроется повторно, **[Нет]** — запуск будет отменен.

- в процессе отладки с помощью данного пункта меню можно *Продолжить отладку*. При вызове данной команды происходит выполнение отлаживаемого приложения.

Запустить без отладки — запуск текущей конфигурации приложения в обычном режиме (без отладки). Выбор конфигурации и повторный запуск приложения описан в пункте *Начать отладку*.

Точка останова — установка точки останова для текущей строки. Точка останова предназначена только для исходных файлов, написанных на языке **VIP** (перечень *Типов файлов* см. в разделе "[Вкладка Atlantis VIP - Подсветка](#)"¹²³), с ограничением в одну точку на строку. Управление установленными точками осуществляется в окне *=[Точки останова](#)*¹³⁵.

Шаг — переход к следующему оператору в текущем блоке. При этом выполнение программы приостановлено и окна приложения находятся в заблокированном состоянии. Курсор позиционируется в строке с оператором и программа выполняется построчно. Вызов процедуры или функции интерпретируется как простой оператор, и поэтому после его выполнения управление передается на следующую строку.

Трассировка — переход к следующему оператору. Аналогично команде *Шаг*, за исключением вызываемых подпрограмм.

Данный режим позволяет выполнять построчно код каждой команды. При этом вызов процедуры или функции перемещает курсор в данные подпрограммы для их построчного выполнения, по окончании которого возвращается к продолжению построчного выполнения самой программы.

	Начать отладку	F9
	Запустить без отладки	F10
	Точка останова	F2
	Шаг	F8
	Трассировка	F7
	Выполнить до курсора	F4
	Выйти из текущего блока	Alt+F8
	Приостановить выполнение	Ctrl+F7
	Закончить отладку	Ctrl+F2
	К текущей позиции	Ctrl+Alt+O
	Отлаживать все	
	Перехватывать исключения	
	Останавливаться на конструкторе	
	Добавить выражение	Ctrl+F5
	Окно стека	
	Окно выражений и переменных	
	Окно локальных переменных	
	Окно точек останова	
	Окно таблиц в памяти	
	Все интерфейсы	
	Загруженные интерфейсы	

Рис. 38. Меню Отладка

Выполнить до курсора — выполнение (прогон) кода до позиции курсора без пошаговой отладки. Функция выполняется при наличии отладочной информации в строке с курсором.

Выйти из текущего блока — прерывание трассировки процедуры/функции и возврат из описания к месту вызова данной процедуры/функции для продолжения пошаговой отладки.

Приостановить выполнение — прерывание выполнения кода интерфейса.

Закончить отладку — завершение процесса отладки.

К текущей позиции — переход к текущей позиции. Курсор перемещается в текущую позицию останова. Если она находится в другом отлаживаемом интерфейсе, то курсор спозиционируется на нее.

Отлаживать все — подключение отладчика ко всем интерфейсам.

Перехватывать исключения — продолжение выполнения программы построчно при возникновении в ней необработанных исключений.

Останавливаться на конструкторе — остановка на инициализации интерфейса.

Добавить выражение — добавление выделенного элемента или слова под курсором из редактора кода в окно *=Выражения и переменные=*.

Окно стека — открытие/закрытие окна *=Стек¹³⁶=*.

Окно выражений и переменных — открытие/закрытие окна *=Выражения и переменные¹³⁷=*.

Окно локальных переменных — открытие/закрытие окна *=Локальные переменные¹³⁸=*.

Окно точек останова — открытие/закрытие окна *=Точки останова¹³⁵=*.

Окно таблиц в памяти — открытие/закрытие окна *=Таблицы в памяти¹⁴¹=*.

Все интерфейсы — открытие/закрытие окна *=Интерфейсы¹³⁹=*.

Загруженные интерфейсы — открытие/закрытие окна *=Загруженные интерфейсы¹⁴⁰=*.

3.2.7. Меню Скрипты

В меню *Скрипты* содержатся функции для выполнения скриптов. Подробнее о правилах их написания см. в разделе *"Интерпретатор скриптов¹⁵⁰"*.

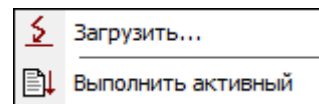


Рис. 39. Меню Скрипты

Загрузить — выполнение скрипта из файла. Вызов данной команды активизирует стандартное окно для выбора файла. По команде открытия файла выполняется скрипт, содержащийся в нем, при этом сам файл в редакторе не открывается.

Выполнить активный — запустить на выполнение код активного файла. В качестве скрипта используется текст редактора кода, поэтому предварительно сохранять его необязательно.

Также меню *Скрипты* содержит список добавленных скриптов в параметрах среды (см. *"Вкладка Общие - Скрипты¹¹¹"*).

3.2.8. Меню TortoiseSVN


Меню содержит команды для работы с основными функциями приложения TortoiseSVN. Данные команды могут применяться как к проекту, так и к отдельному файлу.

Для доступа к данным функциям необходимо наличие установленного приложения TortoiseSVN и Command line client tools.

Проект — функции, предназначенные для работы с файлами, расположенными в каталоге проекта (*.vpr). Данный пункт включает в себя следующие команды: *Обновить до реверзии*, *Зафиксировать*, *Журнал*, *Проверить наличие изменений*.

Остальные команды применяются к активному файлу в редакторе. Состояние файлов рабочей копии в *Viper* автоматически отслеживается и помечается соответствующими иконками в окнах *=Менеджер проекта=* и *=Файловый проводник=*, а также в окне выбора вкладок.

Функции для работы с *TortoiseSVN* доступны в контекстном меню редактора, окнах *=Менеджер проекта⁵⁸=* и *=Файловый проводник⁶³=*.

 Подробное описание функций см. в [документации для TortoiseSVN](#).

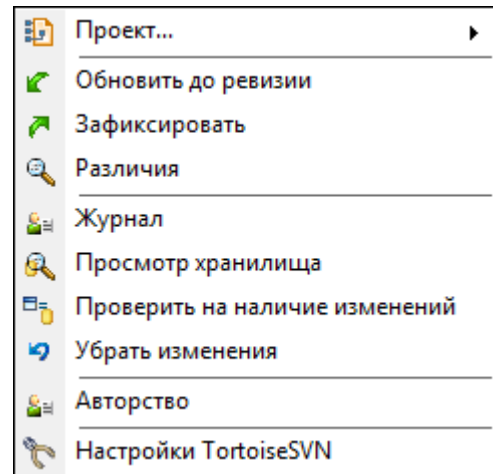


Рис. 40. Меню TortoiseSVN

3.2.9. Меню Инструменты

Меню *Инструменты* предоставляет возможность настройки необходимого инструмента, а также содержит уже созданные пользовательские инструменты, автоматически добавленные в данный список (см. "[Инструменты](#)¹⁴³").

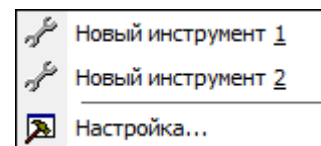


Рис. 41. Меню Инструменты

3.2.10. Меню Окно

В меню *Окно* содержатся команды управления вкладками редактора кода. С их помощью можно осуществлять навигацию по открытым редакторам, изменять их выравнивание и состояние.

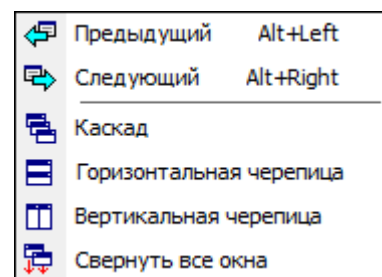


Рис. 42. Меню Окно

Предыдущий — активация предыдущей развернутой вкладки редактора.

Следующий — активация следующей развернутой вкладки редактора.

Каскад — компактное расположение развернутых окон в виде ступени, при этом видна строка заголовка и окна приводятся к единому размеру.

Горизонтальная черепица — компактное расположение открытых редакторов на экране в горизонтальной последовательности, при этом они заполняют всю область главного окна среды, приводясь к единому размеру и не перекрывая друг друга.

Вертикальная черепица — компактное расположение открытых редакторов на экране в вертикальной последовательности, при этом они заполняют всю область главного окна среды, приводясь к единому размеру и не перекрывая друг друга.

Свернуть все окна — сворачивание открытых редакторов.

3.2.11. Меню Справка

В меню *Справка* содержится набор команд для получения справочной информации (см. "[Справочная система](#)"¹⁰³).

Содержание — открытие справочной информации "Среда разработки Viper".

Документация на Атлантис — открытие комплекта документации, поставляемой с установочными файлами **Атлантис**.

Настройка клавиш — список команд и их "горячие" клавиши. Вызов данной функции открывает диалоговое окно со списком команд. Изменить "горячие" клавиши можно с помощью двойного щелчка по команде или функции [**Редактировать**] в инструментальной панели.

Рекомендуется для назначения "горячих" клавиш использовать главное меню *Вид > Панели инструментов > Настройка* (см. "[Настройка инструментальных панелей и горячих клавиш](#)"⁴⁹).

Версии — информация о версии файлов, используемых программой. При активизации данного пункта открывается диалог со списком файлов (*.exe, *.dll, *.rtf каталога **Exe**) и информацией по ним (*Файл*, *Версия продукта*, *Версия файла*, *Путь*).

💡 При использовании **Атлантис** до версии **5.5 Viper.exe** соответствует версии компилятора (**Атлантис**), а **Vipide.dll** — версии редактора. В последующем, новая версия **Атлантис** (компилятора) содержится в файле **ViperCompile.exe**, а версия редактора — в **Viper.exe**.

Подробнее о хранении версий компилятора см. "[Издание 01.2013](#)"¹⁵.

В диалоге *=О версии=* доступна возможность сортировки списка по каждому столбцу, фильтрации по мере ввода текста, а также копирование в буфер обмена текущей записи **Ctrl+C** и всех записей **Ctrl+Alt+C**.

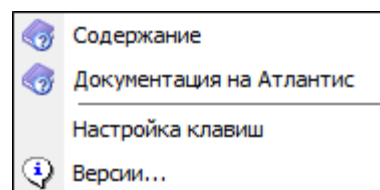


Рис. 43. Меню Справка

3.3. Инструментальные панели

Инструментальные панели обеспечивают быстрый доступ к основным командам редактора.

Все инструментальные панели содержат подмножество или полный набор команд соответствующего меню.

Функции инструментальных панелей обозначены смысловыми иконками, в некоторых дополнительно отображается пометка в виде треугольника острием вниз. При нажатии на него открывается подменю со списком однотипных команд.

При наведении курсора мыши на пункт панели инструментов отображается всплывающая подсказка с названием функции и комбинацией клавиш для быстрого ее запуска. Данное описание отображается при соответствующих настройках панели инструментов. Более детальное описание выводится в строке состояния. Управление инструментальными панелями и изменение их настроек осуществляется с помощью главного меню Вид > Панели инструментов (см. "[Меню Вид](#)"⁴⁶).

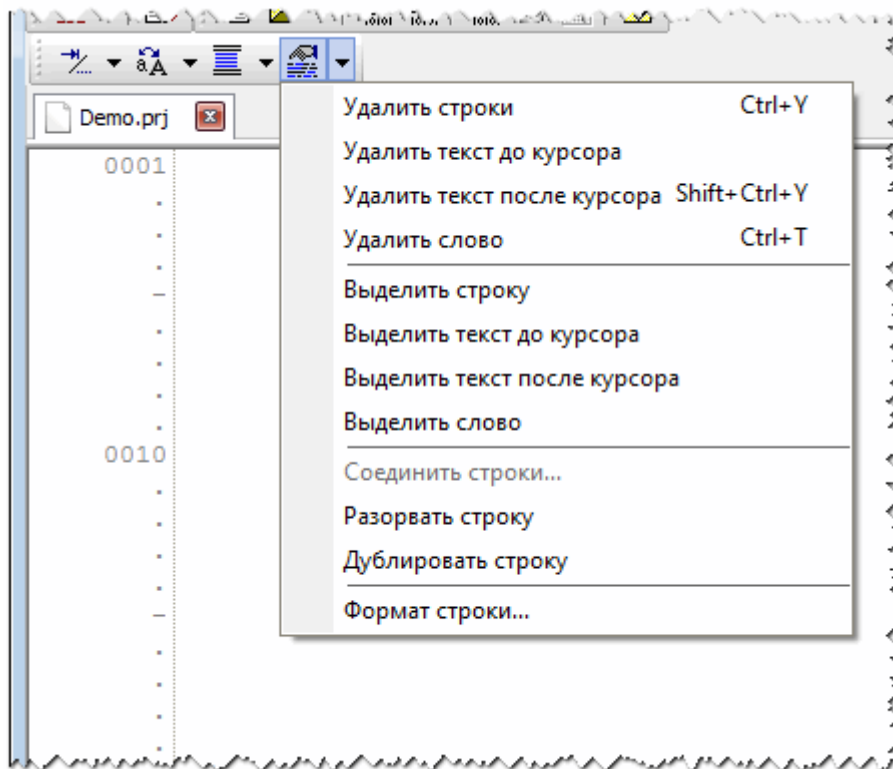


Рис. 44. Группа команд инструмента Блок дополнительно

Инструментальные панели:

- **Стандартная** — группа часто используемых команд управления файлами, редактирования и поиска (*Создать, Открыть, Сохранить, Сохранить все, Найти, Заменить, Найти в файлах*).
- **Правка** — функции для работы с буфером (*Вырезать, Копировать, Вставить, Отменить, Вернуть*).
- **Форматирование** — набор функций меню *Правка > Форматирование* для перестроения, объединения или выравнивания выделенного абзаца и вертикального блока (*Форматирование, Изменение регистра, Блок, Блок дополнительно*).
- **Вид** — функции для настройки среды (*Полноэкранный режим, Номера строк, Непечатаемые символы, Парная скобка, Подсветка скобок, Проверка орфографии, Параметры*).
- **Инструментальные окна** — пункты меню *Вид*, управляющие видимостью окон (*Окно менеджера проекта, Окно вывода, Окно сообщений, Окно структуры кода, Проводник, Окно меток, Окно структуры ресурсов*).
- **Проект** — команды для выбора проекта и управления процессом компиляции (*Новый проект, Открыть проект, Компилировать проект, Пересобрать проект, Компилировать файл/Выполнить профиль, Отключиться от БД, Остановить компиляцию, Параметры проекта*).
- **Отладка** — команды для управления процессом отладки (*Запустить без отладки, Начать отладку, Закончить отладку, Запустить, Приостановить выполнение*).

ние, Выполнить до курсора, Трассировка, Шаг, Отлаживать все, Перехватывать исключения, Останавливаться на конструкторе). В выпадающем списке пиктограмм Запустить без отладки и Начать отладку предоставляется возможность выбора необходимой для запуска конфигурации.

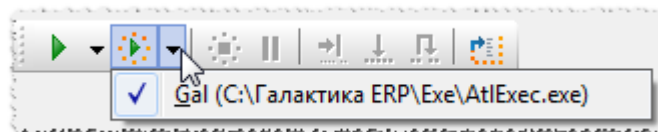


Рис. 45. Выбор конфигурации для запуска

- **Скрипты** — функции для выполнения скрипта (*Загрузить, Выполнить активный*).
- **TortoiseSVN** — команды для работы с основными функциями приложения TortoiseSVN (*Обновить до ревизии, Зафиксировать, Журнал, Различия, Проверить на наличие изменений, Убрать изменения, Обозреватель хранилища*).
- **Инструменты** — функции для настройки инструмента и его активации (*Настройка, пиктограммы созданных инструментов*).
- **Окно** — команды управления вкладками редактора кода (*Предыдущий, Следующий, Каскад, Горизонтальная черепица, Вертикальная черепица, Свернуть все окна*).
- **Режим рабочего стола** — режимы настроек инструментальных окон (см. "[Управление окнами](#)"⁵⁷). В выпадающем списке содержатся варианты: *Отладка* и *По умолчанию*.

3.4. Инструментальные окна

3.4.1. Управление окнами

Для всех инструментальных окон (включая окна отладчика, доступные в пункте оконного меню *Отладка*) имеется возможность изменять геометрию и положение относительно других окон.

Для изменения размера окна необходимо потянуть левой кнопкой "мыши" линию границы окна в нужном направлении.

Для изменения положения окна в **Viper** существует функция открепления и стыковки к границам главного окна приложения. Чтобы открепить окно необходимо потянуть левой кнопкой "мыши" его заголовок, а затем отпустить клавишу "мыши". В результате окно окажется отсоединенным от главного окна приложения. Для стыковки окна при "перетаскивании" появляется специальный индикатор, отображающий возможные направления. Позиция индикатора зависит от зоны расположения указателя. Чтобы стыковать окно в нужной части необходимо в момент "перетаскивания" навести указатель на соответствующую сторону индикатора, при этом зона стыковки будет подсвечиваться.

Также можно "склеивать" окна между собой или вкладывать друг в друга с появлением вкладок переключения между ними. Для вкладывания окон друг в друга необходимо навести дочернее окно в центр окна-приемника. Для закрытия и отстыковки вложенного окна можно воспользоваться контекстным меню, которое вызывается на вкладке окна.

Существует возможность свернуть окно, при этом оно будет представлять собой выплывающую вкладку у границы главного окна приложения. Для сворачивания окна необходимо нажать на изображение "гвоздика", которое находится в заголовке окна,


приклеенного к границе. Окно, которое перемещается на имеющуюся вкладку, формируется отдельной выплывающей закладкой. При этом внутренние вкладки окна также разделяются.

Состав отображаемых окон настраивается в главном меню *Вид* или на панели инструментов *Инструментальные окна*.

В окне, которое содержит колонки существует возможность автоподбора ширины столбца по содержимому. Вызов данной функции осуществляется двойным щелчком ЛКМ на линии правой границы столбца.

В *Viper* существует возможность использовать несколько конфигураций среды: *Отладка* и *По умолчанию*. Каждая конфигурация хранит информацию о расположении окон, их состоянии и размере (в т. ч. ширина столбцов). Пользователю необходимо единожды настроить каждый режим, чтобы потом в зависимости от выполняемых задач выбрать наиболее удобный. Выбор необходимого режима осуществляется на инструментальной панели с помощью выпадающего списка. Сохранение конфигураций происходит при закрытии приложения *Viper* или смене режима.

При запуске/завершении отладки режим отображения окон меняется автоматически.

 Конфигурация режима *Отладка* хранится в файле `Default_userdock_debug.vpr`, режима *По умолчанию* — в файле `Default_userdock_default.vpr`. Данные файлы располагаются в каталоге [настроек среды](#)^[105].

При первой загрузке среды *Viper* на компьютере открываемые инструментальные окна по умолчанию приклеиваются к одной из трех границ главного окна приложения.

Переход в инструментальные окна (в т. ч. в активное окно редактора) производится путем нажатия комбинации клавиш, закрепленной за вызовом конкретного окна. При этом закрытое окно автоматически раскрывается.

Настройка комбинации клавиш производится на вкладке *Клавиатура* главного меню *Вид* > *Панели инструментов* > *Настройка* (см. "[Меню Панели инструментов](#)"^[47]).

3.4.2. Окно менеджера проекта

=*Менеджер проекта*= предназначен для отображения структуры проекта в виде дерева. Проект может включать в себя добавленные пользователем ссылки на файлы, файлы сборки, виртуальные каталоги, пакеты и профили. Подробное описание данных объектов см. в разделе "[Проект](#)"^[81]. Такими объектами могут быть узлы следующих типов: ссылка на файл, файл сборки, профиль, пакет и виртуальный каталог (см. "[Состав проекта](#)"^[85]). Каждый объект может содержать вложенную структуру, при этом глубина иерархии неограничена.

Для открытия данного окна предназначен пункт главного меню *Вид* > *Окно менеджера проекта*.

Дерево проекта состоит из узлов, расположенных на различных уровнях вложенности. В зависимости от уровня расположения, объекты узлов различаются по назначению и функциональности:

- **нулевой уровень** (верхний) — узел, который ссылается на объект проекта. Состоит из названия проекта и имени файла (см. "[Проект](#)"^[81]);
- **первый уровень** — элементы сборки и виртуальные каталоги. В отличие от элементов сборки виртуальные каталоги не подлежат компиляции/выполнению. Они предназначены для группировки файлов и построения иерархии, которая помогает упорядочить структуру файлов проекта и вносит удобство в работу пользователя;

- **дочерние уровни** — объекты, не предназначенные для компиляции. На данных уровнях располагаются ссылки на файлы, которые используются при работе с проектом, и виртуальные каталоги, выполняющие функцию группировки ссылок. С помощью дочерних узлов можно создавать любое количество уровней в структуре проекта.

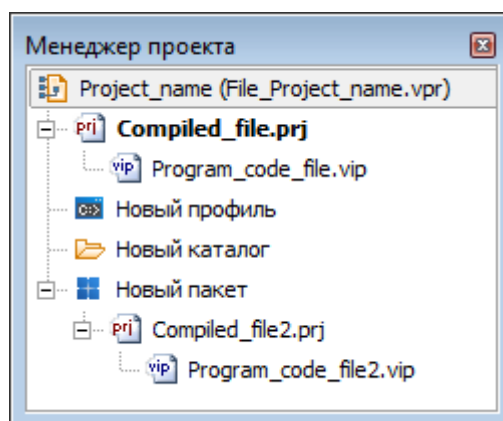


Рис. 46. Окно менеджера проекта

В контекстном меню корневого узла проекта доступны следующие команды:

- **Компилировать проект Shift+F9** — последовательная компиляция элементов сборки. Процесс компиляции выполняется только для измененного исходного кода.
- **Пересобрать проект Ctrl+Shift+F9** — полная компиляция всех элементов сборки проекта. Процесс компиляции выполняется для всего исходного кода независимо от времени модификации файлов.

Существует возможность компилировать только выбранные элементы сборки.

- **Добавить файл** — добавление файла в структуру проекта. В качестве элементов сборки могут быть добавлены проектные файлы (*.prj) и исходные файлы (*.vip, *.frm, *.cnf).
- **Добавить профиль** — создание профиля для вызова внешнего компилятора (см. "[Профили](#)¹⁴⁷").
- **Создать пакет** — добавление объекта для создания конфигурации параметров компиляции (см. "[Состав проекта](#)⁸⁵").
- **Создать каталог** — добавление каталога для группировки ссылок на файлы.
- **Сортировать** — упорядочение вложенных узлов по названию в алфавитном порядке.
- **TortoiseSVN** — команды для работы с функциями приложения TortoiseSVN (см. "[Меню TortoiseSVN](#)⁵³").
- **Параметры Ctrl+Shift+F11** — настройки параметров компиляции и отладки проекта.

Для объекта "виртуальный каталог" доступны следующие команды:

- **Добавить файл** — добавление файла в элементы сборки проекта.
- **Создать каталог** — добавление каталога, в котором могут содержаться любые элементы, не участвующие в сборке.
- **Переименовать F2** — изменение имени.
- **Удалить Ctrl+Del** — удаление выбранного каталога, при этом из списка удаляются и его дочерние элементы. Несохранные файлы остаются открытыми в редакторе кода.
- **Сортировать** — упорядочение списка на дочернем уровне выбранного элемента.

Команды контекстного меню для элементов сборки:

- *Открыть файл* — открытие выбранного файла в редакторе кода.
- *Выбрать текущим* — выбор элемента сборки, над которым будут выполняться функции *Компилировать/Выполнить* и *Пересобрать* в случае, если другой элемент не будет выбран. При этом его имя выделяется в списке жирным шрифтом. По умолчанию *текущим* назначается первый добавленный элемент или выбранный указателем при запуске компиляции.
- *Компилировать/Выполнить Ctrl+F9* — компиляция (выполнение) *текущего* элемента сборки.
- *Пересобрать* — функция для файла/пакета, аналогичная команде *Компилировать/Выполнить*, но отличается тем, что в компиляции участвуют не только измененные, а все файлы.
- *Добавить файл* — добавление дочернего файла.
- *Создать каталог* — добавление дочернего каталога, в который можно помещать любые элементы, не участвующие в сборке.
- *Добавить профиль* — функция для пакета, аналогичная одноименной команде в контекстном меню корневого элемента сборки.
- *Переименовать F2* — функция для пакета, позволяющая изменить его имя.
- *Удалить Ctrl+Del* — удаление выбранного элемента в структуре проекта, при этом из списка удаляются и его дочерние объекты. Несохранные файлы остаются открытыми в редакторе кода.
- *Сортировать* — упорядочение списка на следующем дочернем уровне выбранного элемента.
- *TortoiseSVN* — команды главного меню *TortoiseSVN* (см. "[Меню верхнего уровня](#)"^[24]).
- *Параметры Ctrl+Shift+F11* — настройка компиляции выбранного элемента сборки.
- *Отключить ресурс* — отключение ресурса от отлаживаемого приложения. Пункт активен в режиме отладки только для подключенных ресурсов (см. "[Процесс отладки](#)"^[133]). При выполнении данной команды в "[Окне вывода](#)"^[68] отображается информация об изменении статуса интерфейса.
- *Подключить ресурс с приоритетом репозитория* — подключение ресурса к отлаживаемому приложению, при этом используется приоритет, который указан для данного ресурса в репозитории.
- *Подключить ресурс с приоритетом* — подключение ресурса к отлаживаемому приложению с возможностью выбора приоритета подключаемого ресурсного файла. При вызове данной функции открывается диалоговое окно, в котором необходимо ввести желаемый номер приоритета и нажать кнопку [[Продолжить](#)]. С помощью кнопки [[Отменить](#)] окно ввода приоритета закрывается без подключения ресурса.

Команды *Подключить ресурс с приоритетом* репозитория и *Подключить ресурс с приоритетом* доступны в режиме отладки только для отключенных ресурсов. При подключении в "[Окне вывода](#)" отображается информация об изменении статуса интерфейса, а также номер приоритета ресурса.

В контекстном меню дочерних файлов первого узла содержатся команды, аналогичные описанным выше, за исключением: *Выбрать текущим*, *Компилировать/Выполнить*, *Пересобрать*, *Добавить профиль*, *Переименовать*, *Параметры*, *Подключить/Отключить*.

Функция добавления файла в "[Менеджере проекта](#)" выполняется командами контекстного меню и способом Drag&Drop. Для этого необходимо "перетащить" один или несколько выбранных файлов в структуру проекта из файлового проводника сре-

ды **Viper** или операционной системы, а также из окна *Загруженные интерфейсы*. Для "перетаскивания" панели вкладок редактора необходимо удерживать клавишу **Alt**. При добавлении файла в *Менеджер проекта* осуществляется проверка на соответствие данного типа файла выбранному **уровню** структуры проекта (подробнее об уровнях см. выше).

Дополнительно в *Менеджере проекта* существует возможность добавления узлов, ссылающихся на один и тот же файл, при этом создание дублирующих ссылок в одной ветке запрещено. Изменение содержимого такого файла и его переименование автоматически отражаются во всех узлах, где он содержится. В случае замены файла — узлы, ссылающиеся на него, будут автоматически удалены.

Способы группировки объектов проекта:

- добавление объекта с помощью контекстного меню, вызванного для конкретного элемента списка;
- операция Drag&Drop (т. е. "перетаскивание"), которая поддерживает возможность перемещения узлов внутри дерева;
- зарезервированные комбинации клавиш:
 - **Up/Down** — перемещение выбранного объекта по древовидному списку вверх/вниз;
 - **Left/Right, +/-** — сворачивание/разворачивание активного узла;
 - **Shift+Ctrl+минус/Shift+Ctrl+плюс** — сворачивание/разворачивание всех узлов;
 - **Ctrl+Up/Down** — перемещение активного узла на позицию выше/ниже;
 - **Ctrl+Left/Right** — перемещение активного узла на уровень выше/ниже.

Для выбора нескольких объектов структуры проекта необходимо удерживать клавишу **Ctrl**.

Для быстрого поиска в *Менеджере проекта* доступна возможность фильтрации списка по мере ввода символов. Фильтр распространяется на все узлы, при этом свернутые — разворачиваются. Отмена фильтра выполняется по клавише **Delete** или автоматически при закрытии данного закрепленного окна. В условиях выполненной фильтрации изменение структуры в *Менеджере проекта* недоступно.

В случае, если файл, включенный в проект, входит в состав рабочей копии TortoiseSVN, то **Viper** отслеживает его актуальное состояние и отображает статус соответствующей иконкой.

Ссылки в *Менеджере проекта*, указывающие на несуществующие на диске файлы, помечаются соответствующим значком. При попытке открыть несуществующий файл пользователю предоставляется возможность указать новое расположение файла либо выбрать другой с помощью стандартного диалогового окна. Для файла сборки во всплывающей подсказке отображается полное имя ресурсного файла, который формируется в результате его компиляции.

3.4.3. Окно структуры кода

Окно *Структура кода* предназначено для отображения основных конструкций языка **VIP** (интерфейсы, окна, обработчики, функции и др.), которые содержатся в активном редакторе кода.

При наличии некорректных участков кода в окне *Структура кода* выводится сообщение с описанием ошибки и ее позиции в файле.

Для открытия окна предназначен пункт главного меню *Вид > Окно структуры кода*.

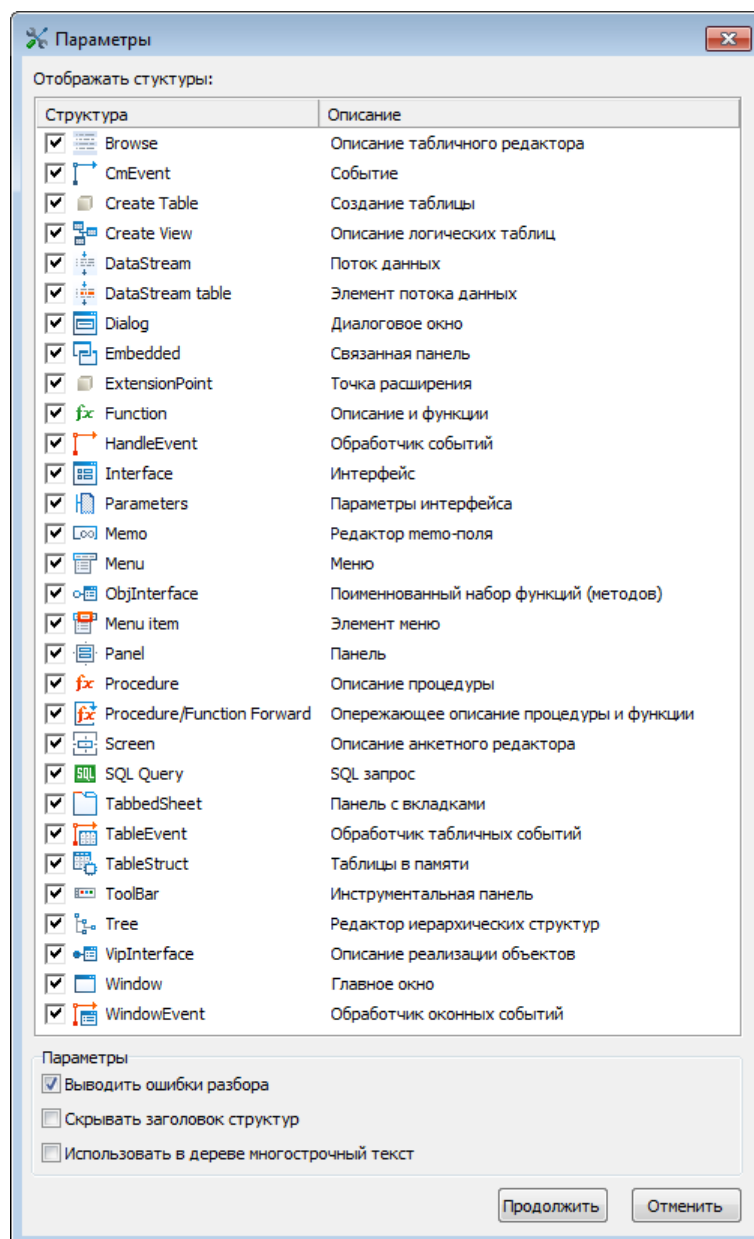


Рис. 47. Параметры окна структуры кода

Наполнение окна *=Структура кода=* происходит автоматически во время работы с кодом, при этом разбор исходного кода для построения его структуры осуществляется сверху вниз.

Разбор конструкций доступен только с использованием подсветки синтаксиса *Atlantis Vip* (меню Вид > Параметры > [вкладка](#)¹²¹ [Синтаксис](#)¹²¹).

Кроме перехода к выбранной конструкции из окна *=Структура кода=* предусмотрено и обратное позиционирование на текущей конструкции в данном окне (команда *Переход в окно структуры* подменю *Навигация по коду*, см. "[Меню Поиск](#)³⁶"). В *=Структуре кода=* реализована возможность перехода к *Предыдущей/Следующей конструкции* (см. "[Навигация](#)⁷¹"). Для элементов окна предусмотрена возможность сортировки (по названию, номеру строки). При обновлении дерева сортировка, а также состояние узлов дерева сохраняется.

В окне *=Структура кода=* реализована возможность изменять порядок конструкций в коде, перемещая их по дереву разбора (см. "[Функции рефакторинга](#)⁷⁹").

В контекстном меню окна содержатся команды:

- *Перейти к* — перевод курсора к выбранной структуре из окна в редактор кода.
- *Параметры* — набор отображаемых элементов структуры и параметры их отображения.

Выводить ошибки разбора — вывод информации об ошибке в окне структуры.

Скрывать заголовок структур — скрывание заголовка структуры, при этом отображается только ее идентификатор, а в случае его отсутствия — заголовок структуры.

Использовать в дереве многострочный текст — использование переноса на следующую строку в тексте названия структуры.

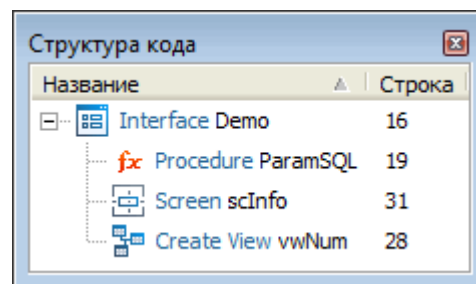


Рис. 48. Окно Структура кода

3.4.4. Окно Файловый проводник

Инструментальное окно «Файловый проводник» содержит древовидную структуру файловой системы компьютера.

Для открытия окна предназначен пункт главного меню *Вид > Проводник*.

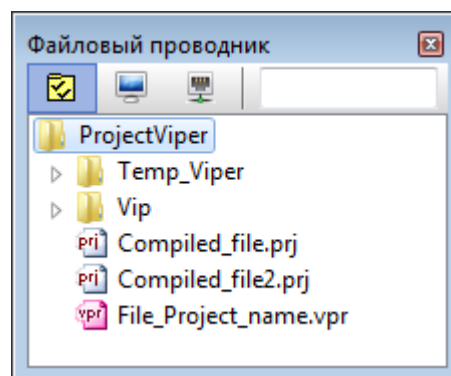


Рис. 49. Инструментальное окно Файловый проводник

В верхней части окна расположены переключатели для выбора расположения:

- *проект* — содержимое каталога текущего проекта;
- *компьютер* — перечень локальных дисков и подключенных накопителей;
- *сеть* — перечень компьютеров и устройств сети.

В дереве файлового проводника доступна возможность поиска по первым символам. Для этого необходимо выбрать объект списка, который будет считаться началом поиска, и ввести символы. В результате выделится первое найденное соответствие. Дополнительно имеется возможность фильтрации списка по ряду символов. Она осуществляется в строке ввода, расположенной в верхней части окна. В результате фильтрации список прокручивается к первому найденному соответствию, остальные — подсвечиваются. Файлы, не соответствующие условию, скрываются. При выполнении по-

иска/фильтрации состояние узлов не меняется. При закрытии закрепленного окна *=Файловый проводник=* фильтр отменяется.

Файл из проводника можно "перетаскивать" в структуру *=Менеджера проекта=*.

В контекстном меню элемента списка содержатся команды:

- *Открыть* — отображение содержимого выбранного каталога или открытие текущего файла в редакторе кода, причем файлы приложения FastReport (*.fr3, *.fp3) автоматически открываются утилитой FREditor, расположенной в *Каталоге компилятора* проекта.



Ранее для открытия **FastReport**-файлов использовалась утилита **FREasyEdit.exe**. Начиная с **Атлантуса 5.5.30** она переименована в **FREditor.exe**.

- *Показать файл в проводнике* — автоматический поиск файла из активного редактора в дереве проводника. Данная функция также доступна и в редакторе.
- *Проводник* — основные команды проводника файловой системы.
- *Скрыть* — закрытие окна *=Файловый проводник=*.

Для каталогов существует дополнительная команда *Сделать корнем*, которая устанавливает выделенный каталог в качестве текущего расположения.

3.4.5. Окно меток

Инструментальное окно *=Метки=* отображает информацию об установленных в редакторе метках и закладках в течении сеанса текущего проекта, а также содержит набор команд для управления ими.

Для открытия окна предназначен пункт главного меню *Вид > Окно меток*.

Запись о размещенных метках/закладках помещается в данное окно с информацией о наименовании метки, ее приоритете, имени файла и номере строки.

В окне предусмотрена возможность сортировки списка по любой из колонок таблицы.

В контекстном меню окна содержатся команды:

- *Перейти к* — открытие соответствующего файла в редакторе кода и позиционирование курсора в строке с текущей меткой (закладкой).
- *Группировать по файлам* — объединение меток с древовидной группировкой по файлам.
- *Фильтр* — фильтрация списка по типу меток (см. "[Меню Метка](#)⁴⁴"), а также по элементу закладка (см. "[Меню Поиск](#)³⁶").
- *Добавить* — добавление метки в соответствии с указанными параметрами.
- *Редактировать* — изменение параметров текущей метки.
- *Удалить* — удаление текущей метки.
- *Удалить все* — очистка списка меток.
- *Скрыть* — закрытие окна *=Метки=*.

В инструментальной панели дублируются все команды контекстного меню окна *=Метки=*, кроме команды *Редактировать*.

В окне *=Метки=* доступна возможность фильтрации списка по ряду введенных символов. Фильтр распространяется на все столбцы. В результате фильтрации отображаются только узлы и записи, соответствующие условию. При включенной группировке по файлам свернутые узлы разворачиваются. Посимвольная фильтрация отменяется клавишей **Del** и при изменении статуса переключателей фильтра на панели инструментов или в контекстном меню, а также при закрытии данного закрепленного окна.

Возможность управления метками также доступна в главном меню *Поиск > Метка* и в контекстном меню редактора кода.

Информация об установленных метках (закладках) хранится в файле проекта.

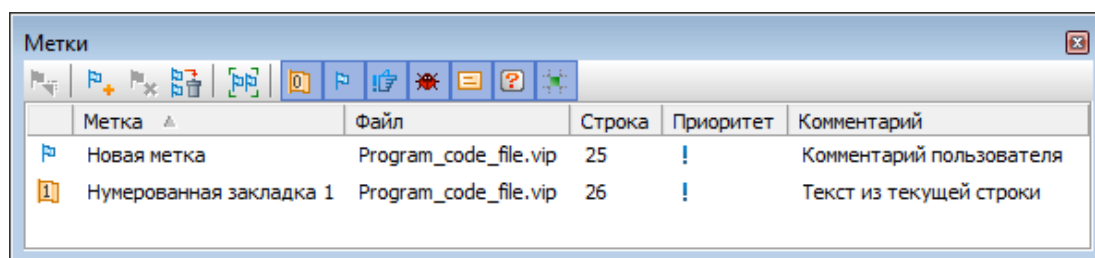


Рис. 50. Инструментальное окно меток

3.4.6. Окно структуры ресурсов

Окно *=Структура ресурсов=* предназначено для просмотра содержимого ресурсных файлов. Объектами ресурса являются: интерфейсы, переменные, процедуры и функции, доступные типы, константы, логические таблицы (с полями), параметры, обработчики, потоки данных (с полями), свойства и обработчики событий.

Окно *=Структура ресурсов=* доступно с версии **Viper 5.5.10** и компилятора **5.5.13**, указанного в параметрах проекта (см. "[Вкладка Компилятор VIP⁸⁹](#)").

Для открытия данного окна предназначен пункт главного меню *Вид > Окно структуры ресурсов*.

Для *Добавления ресурса* предназначена соответствующая команда в контекстном меню окна. Вызов команды осуществляет открытие стандартного диалогового окна выбора файлов (*.res). После выбора ресурса происходит подключение к базе данных и в списке появляется узел с именем ресурса. При загрузке ресурса в *=Окне вывода=* появляется полоса индикатора и информация о версии компилятора. По завершении загрузки в данном окне отображается информация о результате.

При загрузке содержимого ресурсного файла для подключения к БД используется текущая конфигурация БД в настройках проекта, при этом **Параметры лицензии** не учитываются.

Структура ресурсного файла отображается в иерархическом виде. Дочерними узлами ресурса являются: *Items*, интерфейсы и их объекты, которые содержат вложенную иерархическую структуру своих значений.

Для объектов ресурсного файла доступна возможность выгрузки исходного кода из соответствующего ресурса. Данный код хранится в ресурсе, не влияет на его работоспособность и может быть получен только в окне *=Структура ресурсов=*. Наличие исходного кода в ресурсе для объекта обозначается "галочкой" рядом с соответствующим узлом в колонке *Код*. Получить код из ресурса можно с помощью команды **Выгрузить исходный код**, в результате которой код добавится в текущую позицию активного редактора. При отсутствии открытого редактора — создается новый.

Упаковка исходного кода в ресурс выполняется на этапе компиляции при включенном параметре **Сохранять исходные коды реализации в ресурсном файле** (см. "[Вкладка Компилятор VIP⁸⁹](#)").

При необходимости содержимое выбранного ресурса из списка окна можно **Обновить** с помощью соответствующей функции в контекстном меню. В случае, если файл, до-

бавленный в окно *=Структура ресурсов=* был изменен, то при попытке его загрузки автоматически формируется запрос на обновление его содержимого.

Добавленные ресурсы хранятся в списке окна на протяжении текущей сессии **Viper**.

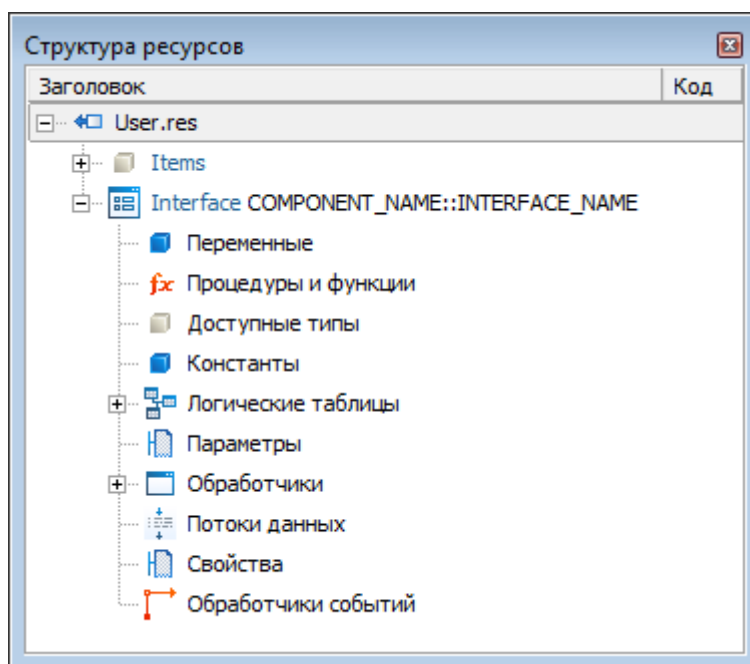


Рис. 51. Окно структуры ресурсов

3.4.7. Окно дерева подключений

Окно *=Дерево подключений=* предназначено для отображения иерархии подключения файлов на уровне кода.

Для открытия окна предназначен пункт главного меню *Вид > Окно дерева подключений*. Поиск и формирование структуры подключения — индексация, выполняется автоматически при загрузке и изменении структуры проекта, а также изменении файлов.

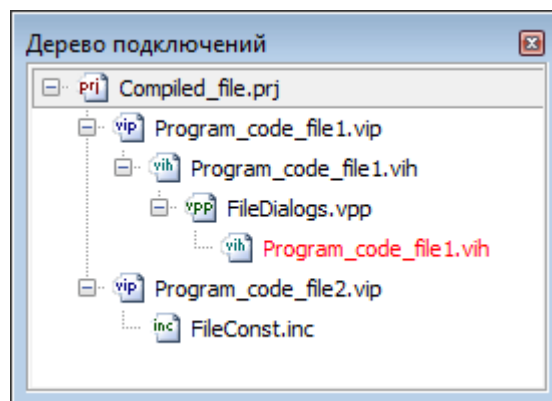



Рис. 52. Дерево подключений

 Красным цветом в дереве выделяется файл, который подключен не только в дочерней ветке, но и в родительской.

Поиск файлов производится в *Списке каталогов для поиска подключаемых файлов* из параметров "текущего" элемента сборки (см. команду [Выбрать текущим](#)⁵⁸).

В «Дереве подключений» на корневом уровне содержится файл сборки (*.prj, *.vip), а дочерними узлами располагаются файлы, подключаемые с помощью `#include`, `#make`. Открыть файл, выбранный в дереве подключений, можно двойным щелчком ЛКМ или по команде **Открыть файл** **Enter** в контекстном меню. Для быстрого поиска в окне «Дерево подключений» доступна возможность фильтрации списка по мере ввода символов.

3.4.8. Окно связей таблиц

Окно «Связи таблиц» позволяет просматривать список возможных связей между таблицами БД. Список формируется автоматически при открытии среды **Viper** на основании актуальной сборки **Галактика ERP**. В дальнейшем загрузка связей — индексация источников, выполняется при редактировании файлов и изменении структуры сборки проекта на основании структур, описанных в разделах `where` и `bounds`, конструкции `Create view`.

Для открытия окна предназначен пункт главного меню **Вид > Окно связей таблиц**.

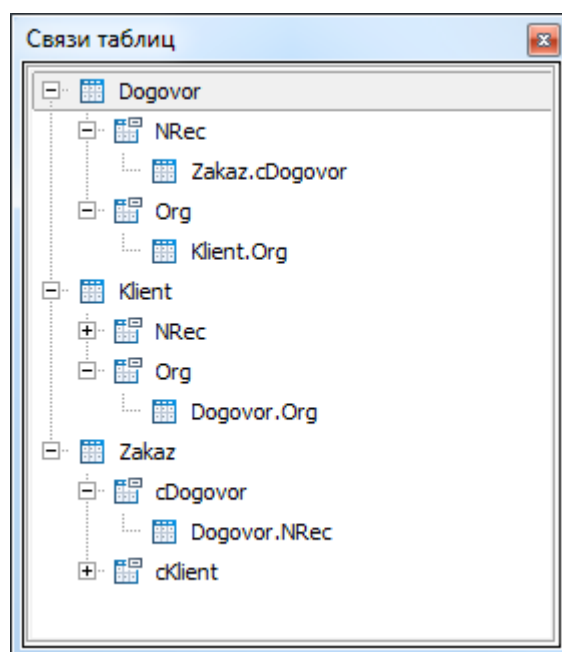



Рис. 53. Связи таблиц

Поиск связей производится в подключаемых файлах, которые содержатся в **Каталогах для поиска подключаемых файлов** соответствующего файла сборки.

 В список попадают лишь те связи, которые имеют явное описание имени таблицы и поля, при этом имя таблицы должно содержаться в разделе `From` структуры `Create view` или в пользовательском скрипте для подсказчика.

Структура связей формируется в виде дерева. На корневом уровне содержатся таблицы, для которых установлена "подцепка" (ограничение, соответствие) с другими таблицами. Во вложенных уровнях — поле корневой таблицы и его связь с полем "подцепляемой" таблицы (в формате *ИмяТаблицы.ИмяПоля*).

Список отсортирован по алфавиту. Навигация по записям осуществляется мышкой или клавишами **Up/Down**. Для удобства перемещения по связям в контекстном меню узла с именем таблицы содержатся команды:

"С кем" связана таблица — показать "подцепки", которые установлены для текущей таблицы. Функция отобразит поля выбранной таблицы, для которых установлены связи ("подцепки").

"Кто" *связывается с таблицей* — показать "подцепки", которым установлена связь с текущей таблицей. Функция отобразит поля выбранной таблицы и связи, для которых "подцеплены" данные поля.

Вернуться к полному списку таблиц можно с помощью функции *Все связи таблиц*.

Список связей таблиц можно *Сохранить в файл Ctrl+S*. Данная команда доступна контекстном меню списка таблиц. При ее активации открывается диалоговое окно для сохранения полного списка связей в csv-файл (имя файла по умолчанию **TableBounds.csv**). Дополнительно в окне *=Связи таблиц=* существует возможность фильтрации по вводу.

3.4.9. Окно вывода

=Окно вывода= предназначено для отображения этапов выполнения процесса компиляции и профиля, статусов подключения/отключения отлаживаемых ресурсов, а также результата выполнения функции поиска и импорта/экспорта параметров проекта.

Открытие окна осуществляется с помощью главного меню *Вид > Окно вывода* и автоматически при выполнении процесса компиляции (см. "[Использование компилятора](#)"¹³⁰) или команды поиска (см. "[Команда Найти в файлах](#)"³⁹).

Окно состоит из нескольких вкладок:

- *Консоль* — предназначена для вывода этапов процесса компиляции. Для данной вкладки существуют дополнительные параметры (см. "[Вкладка Общие - Запуск](#)"¹¹⁰).
- *Поиск в файлах 1* и *Поиск в файлах 2* — предназначены для отображения результата выполнения команды *Найти в файлах*. По умолчанию результат поиска выводится на вкладке *Поиск в файлах 1*, при необходимости можно назначить вывод на вкладку *Поиск в файлах 2* с помощью параметра *Выводить во вторую панель* (см. "[Команда Найти в файлах](#)"³⁹).

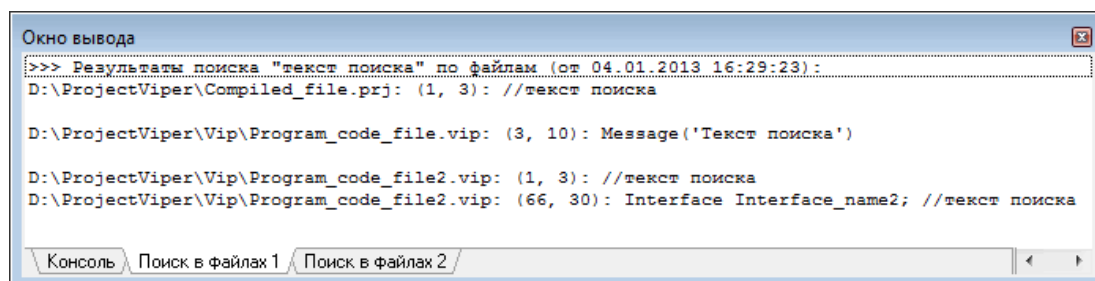


Рис. 54. Вкладка результатов поиска в файлах

В контекстном меню вкладок *Консоль* и *Поиск в файлах* содержатся команды:

- *Перейти к строке* — открытие файла из текущей строки *=Окна вывода=* в редакторе кода и позиционирование курсора в соответствии с отображенными координатами. Функция также выполняется по двойному щелчку левой кнопкой мыши.
- *Перейти назад в окне вывода* — переход к предыдущей записи на текущей вкладке.
- *Перейти дальше в окне вывода* — переход к следующей записи на текущей вкладке.
- *Очистить лист* — удаление всех записей текущей вкладки.
- *Копировать все* — копирование всех записей текущей вкладки в буфер обмена.
- *Копировать строку* — копирование текущей строки в буфер обмена.
- *Сохранить в файл* — сохранение в файле всех записей текущей вкладки. При активации команды отображается диалоговое окно выбора файла.

- **Скрыть** — скрытие =Окна вывода=. В случае, если окно закреплено — оно закрывается, незакрепленное — свернется. Функция выполняется с помощью клавиши **Esc** как из =Окна вывода=, так и из редактора кода.

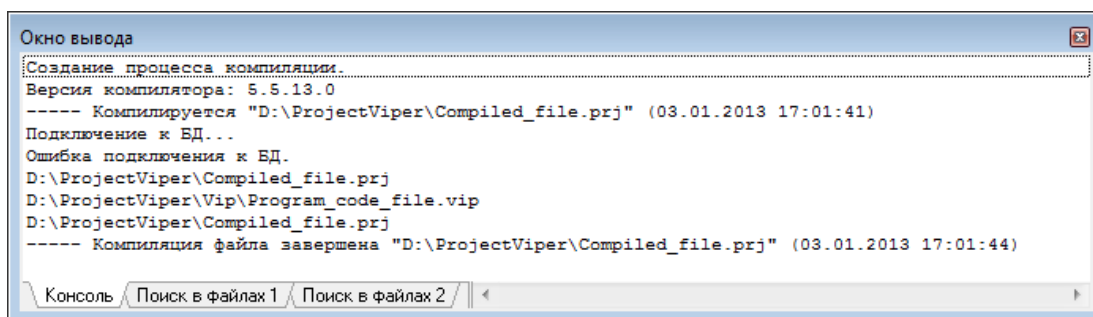


Рис. 55. Вкладка Консоль окна вывода

Для быстрого поиска в =Окне вывода= доступна возможность фильтрации по мере ввода символов. Фильтр применяется к активной вкладке, отменяется по клавише **Delete**, а также автоматически при обновлении вкладки (например, в процессе компиляции). Функции *Перейти назад/Перейти дальше в окне вывода* выполняются с учетом выполненной фильтрации.

3.4.10. Окно сообщений компилятора

Окно =Сообщения= предназначено для отображения состояния процесса компиляции и вывода ошибок и предупреждений компилятора.

Открытие окна осуществляется с помощью главного меню *Вид > Окно сообщений* и автоматически при обнаружении *Ошибки* в процессе компиляции.

Структура результата компиляции:

- 1-й уровень: узел процесса компиляции;
- 2-й уровень: сообщение компилятора;
- 3-й уровень: стек сообщения.

При компиляции элемента сборки в окне =Сообщения= создается соответствующий узел первого уровня, который содержит имя компилируемого файла и время начала сборки. В данный узел добавляются все поступающие от компилятора сообщения. Для каждого элемента сборки в текущей сессии компиляции создается новый узел первого уровня. Узлы прошлых сессий компиляции являются историей. По умолчанию при запуске компиляции история удаляется, возможность **Сохранять историю сообщений компилятора** доступна при использовании соответствующей настройки в параметрах проекта (см. "[Вкладка Компилятор VIP — Сообщения](#)⁹⁵").

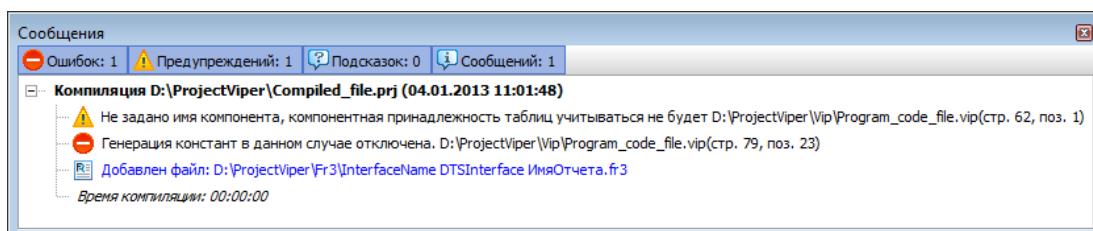


Рис. 56. Окно сообщений

В контекстном меню содержатся команды:

- **Перейти к** — открытие данного файла в редакторе кода и установка курсора в строке, соответствующей указанной позиции в окне =Сообщения=. Данная ко-

манда также выполняется двойным нажатием. Файлы приложения FastReport (*.fr3) открываются во внешней утилите FREditor.

- **Удалить запись** — удаление текущего сообщения из списка.
- **Удалить все** — удаление всех сообщений.
- **Редактировать сообщение** — перевод текущей строки в состояние редактирования с возможностью изменения, выделения или копирования текста в буфер. При выходе из данного состояния модификация данной строки не сохраняется.
- **Копировать сообщение** — копирование текущей строки в буфер обмена.
- **Сохранить сообщения** — сохранение в файле всех записей окна. Вызов команды отображает диалоговое окно выбора файла.
- **Фильтры** — разделение сообщений по характеру, соответствующему правилам диагностики. С помощью фильтров можно просматривать сообщения определенного типа. Фильтры дублируются на инструментальной панели окна сообщений с отображением количества сообщений из последнего сеанса компиляции в текущем проекте. Фильтр применяется ко всем сессиям компиляции и доступен на любом ее этапе. Для отображения сообщений определенного типа необходимо включить соответствующий фильтр:
 - **Ошибок** — сообщения об ошибках кодирования.
 - **Предупреждений** — информация, позволяющая предотвратить нежелательные последствия.
 - **Подсказок** — предложения по улучшению.
 - **Сообщений** — информация о выполненной операции.При отключении фильтра скрывается весь узел компиляции, если все сообщения в нем соответствуют данному фильтру.
- **Скрыть** — скрытие окна «Сообщения». Функция аналогична выполнению команды скрытия «[Окна вывода](#)⁶⁸».

Дополнительно реализована возможность фильтрации сообщений по мере ввода символов. Фильтр применяется ко всем сессиям компиляции. При фильтрации свернутые узлы разворачиваются и переключатели на верхней панели окна автоматически включаются. Фильтр отменяется по клавише **Delete**, а также автоматически в процессе компиляции.

В условиях посимвольной фильтрации функция **Удалить все** недоступна.

3.5. Редактор

Встроенный редактор интегрированной среды представляет собой область ввода со стандартным набором функционала, присущим любому редактору исходных текстов.

В возможности редактора входит:

- редактирование;
- работа с буфером;
- синтаксический раскрасчик;
- подсказка кода и т. д.

Кроме того, встроенный редактор включает ряд дополнительных функций:

- работа с вертикальным блоком;
- сворачивание кода;

- проверка орфографии;
- навигация по структурам и т. д.

Редактор поддерживает синтаксис практически всех основных языков программирования, в т. ч. синтаксис языка **VIP**. Настройка синтаксиса осуществляется в параметрах среды (см. "[Вкладка Синтаксис](#)"¹²¹). Для каждого типа файлов применяется раскраска синтаксических конструкций, поиск парных скобок и элементов структуры кода и т. д.

Количество открытых редакторов неограничено. Для удобства перемещения между ними существует панель вкладок и **=Окно вкладок=** с функцией быстрого поиска. На панели вкладок справа располагается выпадающий список, который содержит перечень открытых редакторов. С его помощью также можно быстро активизировать необходимую вкладку. Для движения панели вкладок предназначены специальные кнопки, расположенные справа от нее. Для удобства пользователю предоставляется возможность открепления и стыковки вкладок. Для этого необходимо "потянуть" заголовок вкладки левой кнопкой "мыши" (подробнее см. "[Управление окнами](#)"⁵⁷).



*Стыковать вкладки редактора можно в области самого редактора по вертикали или горизонтали, но **только** в одинаковом направлении.*

В статусной строке **Viper** отображается полный путь файла, который открыт в активном редакторе. При необходимости его можно отражать и в заголовке приложения, для этого необходимо включить соответствующую настройку среды.

По наведению указателя на заголовок вкладки редактора появляется всплывающая подсказка с полным именем файла соответствующей вкладки.

На панели вкладок в контекстном меню содержатся следующие функции:

- **Заккрыть** — закрытие активного файла. При наличии несохраненных изменений сформируется запрос на их сохранение. После закрытия текущей вкладки активизируется та, что была предыдущей в состоянии "активная", и история активизирования очищается.
- **Заккрыть все кроме текущего** — закрытие всех открытых файлов, кроме активного. При наличии несохраненных изменений формируется запрос, как и в предыдущей функции.
- **Копировать имя файла** — добавление полного имени активного файла в буфер обмена.
- **Показать файл в проводнике** — автоматический поиск файла из активного редактора в дереве проводника.
- **Окно выбора вкладок** — открытие окна, позволяющего быстро переключиться между открытыми файлами и окнами.
- **Сохранить/Сохранить как/Сохранить все** — сохранение изменений в активном файле/в новом/во всех измененных файлах (подробнее см. "[Меню Файл](#)"²⁴).
- **Tortoise SVN** — функции приложения TortoiseSVN (подробнее см. "[Меню TortoiseSVN](#)"⁵³).
- **Модифицирован** — применение статуса, определяющего наличие изменений в файле (подробнее см. "[Меню Правка](#)"²⁷).
- **Только чтение** — применение атрибута, запрещающего изменять активный файл (подробнее см. в разделе из предыдущей функции).

Пользователю предоставляется возможность настройки редактора и панели вкладок, для этого предназначено главное меню **Вид > Параметры**.

3.5.1. Навигация

В редакторе существуют различные способы навигации по тексту, которые позволяют в процессе редактирования использовать элементы кода целиком.

Способы навигации по тексту при помощи клавиш:

- **Left/Right** — посимвольное перемещение курсора;
- **Ctrl+Left/Right** — перемещение курсора в начало/конец следующего/предыдущего идентификатора или символа-разделителя в строках;
- **Ctrl+Up/Down** — построчная прокрутка редактора кода в зависимости от направления, соответствующего используемой клавиши;
- **Home** — перемещение курсора в начало строки. Дополнительно существует возможность *Улучшить клавишу Home* (см. "[Вкладка Редактор — Параметры](#)^[116]");
- **End** — перемещение курсора в конец строки;
- **PageUp** — постраничная прокрутка редактора кода по направлению к началу файла. При этом вертикальная позиция курсора не меняется;
- **PageDown** — постраничная прокрутка редактора кода по направлению к окончанию файла. При этом вертикальная позиция курсора не меняется;
- **Ctrl+Home** — перемещение курсора в начало файла;
- **Ctrl+End** — перемещение курсора в конец файла.

Для удобства перемещения по структуре программного кода реализована группа команд *Навигация по коду* (см. "[Меню Поиск](#)^[36]"), выполняющих перемещение курсора:

- по парным скобкам **Ctrl+Shift+B** — курсор, расположенный перед скобкой, перемещается к ее соответствующей парной скобке.
- по парным конструкциям **Alt+[** — курсор перемещается от начала текущей конструкции в конец, и наоборот.
- к предыдущей/следующей конструкции **Alt+Up/Alt+Down** — курсор последовательно перемещается по конструкциям кода в редакторе с соответствующей фокусировкой в окне *=Структура кода=*.

Дополнительно существует возможность навигации по измененным строкам с помощью команды *Перейти к предыдущему (следующему) изменению* (см. "[Меню Навигация по коду](#)^[43]").

Для выделения текста предназначена комбинация клавиш **Shift+<клавиши навигации по строкам/PageUp/PageDown>**.

Для блочного выделения с клавишей **Shift** следует удерживать клавишу **Alt**:

- **Shift+Alt+<клавиши со стрелками/указатель>** — выделение указанного блока;
- **Shift+Alt+Home** — выделение блока до начала строки;
- **Shift+Alt+End** — выделение блока до конца строки;
- **Shift+Alt+PageUp** — выделение указанного блока, размером в одну страницу, по направлению к началу файла;
- **Shift+Alt+PageDown** — выделение указанного блока, размером в одну страницу, по направлению к окончанию файла.

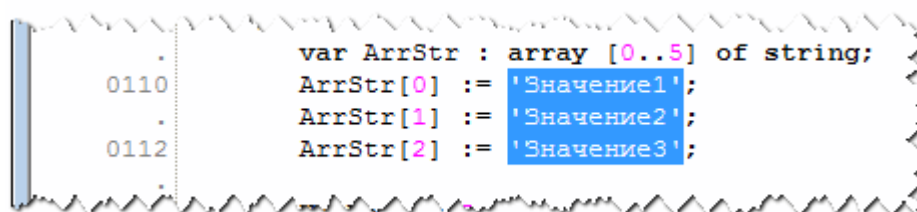


Рис. 57. Выделение вертикальным блоком

Работа с буфером обмена выделенных блоков осуществляется с учетом их типа.

В редакторе существуют дополнительные функции модификации текста:

- по **Ctrl+Del** удаляется последовательность символов (буквы и цифры), расположенных справа от курсора до пробела/знака, а также группа пробелов или знаков;
- по **Ctrl+BackSpace** удаляется последовательность символов (буквы, цифры, знаки, пробелы) от начала текущего слова до курсора.

Навигация по вкладкам редактора осуществляется с помощью комбинации клавиш **Alt+Left/Right**.

3.5.2. Форматирование

В редакторе существует набор элементов для оформления и смыслового выделения текста. В качестве таких элементов используются параметры редактора.

Параметр *Автоотступ* применяется автоматически при добавлении новой строки по нажатию кнопки **Enter** и выполняет функцию автоматического выравнивания (см. "[Вкладка Редактор — Параметры](#)"¹¹⁶).

Параметр *"Умные" скобки* доступен только при включенном параметре *Автоотступ* и применяется для выравнивания скобок (см. "[Вкладка Редактор — Параметры](#)"¹¹⁶).

Параметр *Парная скобка* применяется автоматически при вставке открывающей скобки ("{" , "(" , "[") и выполняет добавление парной закрывающей (см. "[Вкладка Редактор — Параметры](#)"¹¹⁶).

Параметр подсветки синтаксиса определяет цветовое оформление и стиль ключевых слов и контекстов, таких как комментарии, строковые константы и т. д. (см. "[Вкладка Atlantis VIP — Подсветка](#)"¹²³). Схемы цветовой раскраски разделены на группы форматов файлов и доступны для изменения.

3.5.3. Сервис

Сервисные функции редактора обеспечивают удобство в процессе кодирования за счет автоматизации некоторых процессов.

В редакторе содержится функция проверки орфографии. Поиск орфографических ошибок производится в рамках заданных конструкций языка **VIP**. Параметр *Подчеркивать слова с ошибками* в настройках редактора обрабатывает слова с ошибками аналогично популярным офисным приложениям (подробнее см. "[Вкладка Редактор — Проверка правописания](#)"¹¹⁵). Для исправления орфографических ошибок предназначена команда *Орфография* (см. "[Меню Правка](#)"²⁷).

Существует возможность масштабирования текста в редакторе посредством изменения размера шрифта в активном редакторе. Для изменения масштаба используются команды *Увеличить масштаб* и *Уменьшить масштаб* (см. "[Меню Вид](#)"⁴⁶) или удержание клавиши **Ctrl** с прокручиванием колеса мыши. Для облегчения контроля за внесенными изменениями реализована функция автопометки модифицированных строк: измененные строки обозначаются вертикальной чертой в служебном поле. До сохранения изменений — черта **красного** цвета, а после — **зеленого**. Функция доступна при использовании параметра *Отображать* и *Автопометка редактированных строк* в настройках редактора (см. "[Вкладка Редактор — Служебное поле](#)"¹¹⁸).

Функция *Автозаполнение* применяется при необходимости автоматической вставки данных на основании шаблона. Создание шаблона данных выполняется в параметрах редактора (см. "[Вкладка Atlantis VIP — Автозаполнение](#)"¹²⁴). Вызов списка шаблонов

из редактора осуществляется комбинацией "горячих" клавиш **Ctrl+J**. При необходимости их можно изменить.

Функция **Автозамена** применяется автоматически для замены одного или ряда символов готовым шаблоном. Замена выполняется при добавлении новой строки или символа, следующими за введенным рядом. Создание шаблона автозамены выполняется в параметрах синтаксиса (см. "[Вкладка Atlantis VIP — Автозамена](#)¹²⁵"). Параметры использования функции содержатся в настройках редактора (см. "[Вкладка Редактор — Автозамена](#)¹¹⁵").

Функция **Контекстная справка** применяется при необходимости в подсказке по синтаксису языка **VIP** (см. "[Вкладка Atlantis VIP — Справка](#)¹²⁵"), при этом вызов справки осуществляется непосредственно из редактора кода с помощью клавиши **F1**, а слово под курсором используется в качестве ключевого.

Функции сворачивания/разворачивания блока позволяют управлять состоянием видимости программного блока (см. "[Меню Правка](#)²⁷"), что облегчает работу с объемными конструкциями кода. Данная возможность доступна для файлов синтаксической схемы *Atlantis VIP*, при этом должна быть включена подсветка кода для данной схемы (меню *Вид* > *Параметры* > вкладка *Синтаксис*).

В **Viper** существует возможность "перетаскивания" файлов. Подробнее см. в разделе "[Окно менеджера проекта](#)⁵⁸". "Перетаскивать" файлы можно не только в среде **Viper**, но и из операционной системы. В случае перемещения файла в область редактора кода, он открывается без добавления в проект. Чтобы добавить файл в проект необходимо "перетащить" его в *Менеджер проекта*. При открытии файла проекта (*.vpr) с помощью проводника пользователю будет сформирован запрос "Открыть проект?". По нажатию кнопки [ОК] — проект установится в качестве рабочего, по кнопке [Отмена] — файл откроется в редакторе.

В среду интегрированы команды для работы с функциями **Subversion** (*Просмотр хранилища, Настройки TortoiseSVN, Обновить до ревизии, Зафиксировать, Различия* и др.). Функции доступны в главном меню, редакторе кода, окнах *Менеджер проекта* и *Файловый проводник*. В **Viper** можно изменять и отслеживать текущее состояние файлов рабочей копии. Подробнее см. в разделе "[Меню TortoiseSVN](#)⁵³".

Функция **Открыть файл Ctrl+Enter** осуществляет открытие файла, имя которого содержится в редакторе под курсором в директиве `#include` или `#make`. Для файлов, заданных маской (*.<расширение файла>) открывается стандартное диалоговое окно выбора файла.

Командой **Открыть предыдущий файл Shift+Ctrl+Enter** будет открыт файл, указанный в предыдущей позиции курсора. Последовательность поиска файла см. в разделе "[Использование компилятора](#)¹³¹".


3.5.4. Подсказка кода

В редакторе существует возможность подсказки кода, которая включает в себя следующие функции:

- автодополнение;
- переход к описанию;
- вывод описания о текущей конструкции;
- подсказка параметров процедур и функций.

Возможности подсказки доступны при включенном параметре [Использовать подсказчик кода](#)¹²⁰ и включенной подсветке соответствующей синтаксической схе-

мы, для которой будет вызываться подсказка, в параметрах среды (меню Вид > Параметры > вкладка Синтаксис).

 Следует учесть, что формирование списка глобальных переменных, констант, интерфейсов и их методов выполняется на основании "текущего" элемента сборки в =Менеджере проекта= (дополнительно см. команду [Выбрать текущим](#)⁵⁸).

В среде существует дополнительная возможность просматривать иерархию подключения файлов проекта на уровне кода. Для этого предназначено окно =[Дерево подключений](#)⁶⁶=.

3.5.4.1. Автодополнение

Функция представляет собой выпадающий список ключевых слов и идентификаторов, который позволяет выбрать необходимую запись для подстановки в процессе кодирования.

Вызов подсказки кода осуществляется комбинацией клавиш **Ctrl+Space**, при необходимости ее можно изменить (см. "[Настройка инструментальных панелей и горячих клавиш](#)"⁴⁸). Также подсказка может быть отображена автоматически во время написания кода (например, после символа ".").

В открывшемся списке подсказки отображается перечень ключевых слов и идентификаторов, доступных в текущей области видимости. Записи подсказки формируются по шаблону: <тип> <название> <параметры>. Первый вариант в списке совпадает с текущим словом в редакторе, не учитывая позицию каретки. Вторым следует соответствие до позиции курсора, далее расположены остальные варианты подсказки.

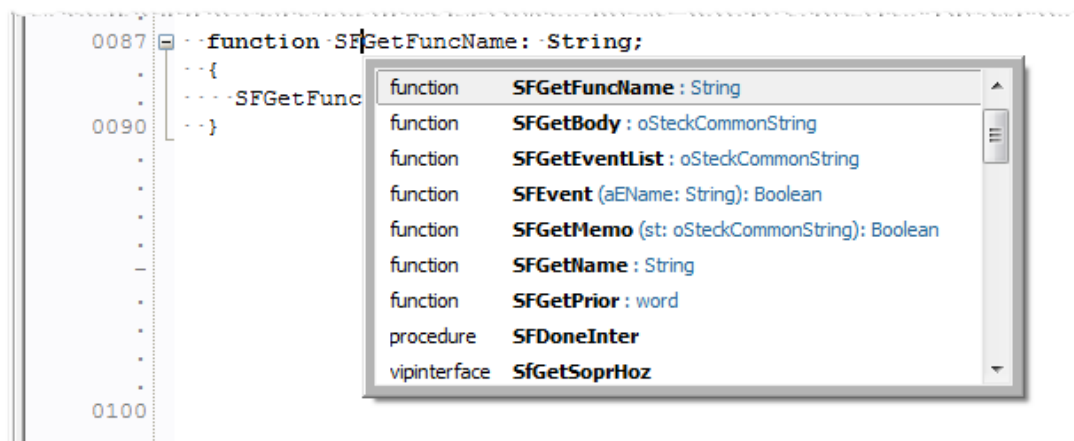


Рис. 58. Вызов функции автодополнения

В списке автодополнения для параметров процедур/функций в первую очередь подсказчик предлагает записи, тип которых соответствует текущему параметру редактируемой функции.

В функциях/процедурах, предназначенных для установки ограничений (bounds), в качестве параметров подсказчиком будут предложены идентификаторы ограничений из текущего контекста.

По умолчанию подсказка формируется на основании **структуры кода**, которой являются конструкции языка **VIP** из активного редактора (см. "[Окно структуры кода](#)"⁶¹) и подключаемых файлов (#include, #make). Формирование списка глобальных переменных, констант, интерфейсов и их методов выполняется на основании "текущего" элемента сборки в окне =Менеджер проекта= (см. команду [Выбрать текущим](#)⁵⁸).

Подсказка по структуре кода выполняется для следующих конструкций:

- `VipInterface`, включая разбор секции `implements`;
- `ObjInterface`, включая набор описываемых методов;
- методы родительских интерфейсов;
- глобальные и локальные переменные/константы;
- макропеременные `#declare` и `#define`, включая их параметры;
- `Create View`, включая секцию `var`, идентификаторы `FieldName`, а также список таблиц и синонимов из секции `from`;
- таблицы с префиксом `'#'`, `'tn'` и `'type$'` из секции `from` конструкции `Create View`;
- список ограничений и связей узлов из секции `bounds` конструкции `Create View`;
- `Procedure/Function` (в т. ч. с ключом `external`), включая разбор параметров;
- словарные типы `Set type`;
- `Table Struct`;
- `Record`;
- `CmEvent`;
- `Embedded`.

Для скриптовых языков (Visual Basic Script и JScript) список подсказки формируется на основании [API редактора](#)¹⁵¹, который можно использовать при выполнении скриптов, кроме того, для API-объектов подсказчик предоставляет список методов и свойств.

Дополнительно существует настраиваемый список подсказки в параметрах среды. Он формируется на основании следующих источников:

- **Ключевые слова и функции языка `vir`**;
- **Функции для работы с Excel**;
- **Пользовательский скрипт** — возможность самостоятельного наполнения подсказчика (см. "[Примеры использования скриптов](#)"¹⁶⁵) с помощью интерпретатора скриптов.

Скрипты, предназначенные для наполнения пользовательской подсказки можно вызвать через меню [Скрипты](#)⁵³ либо установленной комбинацией "горячих" клавиш, либо назначить на выполнение определенного события, например, начало программы, открытие проекта, вызов подсказки и т. д. (см. "[Вкладка Общие - Скрипты](#)"¹¹¹). Возможность наполнения подсказчика через интерпретатор скриптов позволяет пользователю самому определить контент наполнения в зависимости от ситуации. Записи пользовательских подсказок объединены в специальные контейнеры — блоки. Блок представляет собой коллекцию записей и предназначен для управления видимостью записей в зависимости от активного редактора. Блоки доступны и хранятся в [объекте "Коллекция"](#)¹⁶¹ (`CodeCompleteUser`), который доступен в скрипте.

[Объект "Блок"](#)¹⁶² (`CodeCompleteBlock`) содержит коллекцию записей подсказки и настройки видимости в зависимости от редактируемого файла. Видимость блока может быть настроена на конкретный файл по его имени, по типу используемого языка, по расширению файла. Если ограничения не заданы, то записи подсказки будут доступны для всех редакторов.

[Объект "Запись"](#)¹⁶³ подсказки (`CodeCompleteItem`) хранится в коллекции конкретного блока и отображается в списке подсказки (вызов **Ctrl+Space**) в зависимости от его настроек. Запись подсказки может содержать иерархию вложенных за-

писей, которые будут доступны в качестве расширений после использования символа ".".

Процесс добавления пользовательской подсказки включает выполнение следующих шагов:

- 1) добавление нового блока в объект `CodeCompleteUser`;
- 2) настройка области видимости для блока;
- 3) наполнение блока иерархией записей подсказок.

Настройка списка и возможность выбора *Источника данных* доступна в параметрах среды (см. "[Вкладка Редактор - Подсказка кода](#)"¹²⁰).

Навигация по списку подсказки осуществляется с помощью клавиш **Up/Down/End/Home** или указателем мыши с зажатой ЛКМ. Вставка текущего варианта выполняется следующими способами:

- клавишей **Tab** или с помощью ЛКМ текущее слово в редакторе заменяется целиком без добавления пробела, табуляции, переноса каретки;
- клавишей **Enter** или с помощью мыши заменяется текст слева от курсора до разделителя без добавления пробела, табуляции, переноса каретки;
- клавишей знака (+-*/=():;.,) заменяется текст слева от курсора до разделителя, при этом в конце добавляется соответствующий знак. После нажатия клавиши "." подсказка не закрывается, если добавленная запись содержит вложенные, а отображает соответствующие подзаписи;
- клавишей **Space** заменяется текст слева от курсора до разделителя с добавлением пробела.

Для закрытия списка подсказки предназначена клавиша **Esc**.

Записи в списке фильтруются на основании введенных символов до позиции курсора и обновляются по мере ввода. Набор допустимых идентификаторов подсказки формируется относительно текущей области видимости структуры кода.

В случае обнаружения подсказчиком кода одноименных идентификаторов предоставляется возможность выбора необходимой структуры кода. В таком случае в окне подсказчика отобразится список конструкций. Для выбора необходимого варианта из списка следует воспользоваться одним из способов:

- продолжить ввод идентификатора, при этом автоматически отобразится полный список подсказки с соответствующими вхождениями;
- с помощью клавиш **Up/Down/End/Home** выбрать конструкцию из списка подсказчика, соответствующую контексту ввода. Затем отобразить записи выбранной конструкции с помощью мыши или клавиш **Left/Tab/Space/Enter**, или же ввести первые символы нужной записи. Вернуться к списку групп можно клавишей **Left/BackSpace**.

3.5.4.2. Переход к описанию

Функция предназначена для быстрого перемещения от места вызова/использования идентификатора к его описанию, а также от реализации структур к их описанию.

Переход осуществляется по всем конструкциям кода, для которых доступна функция [Автодополнение](#)⁷⁵. Выполнить переход можно следующими способами:

- установить курсор на нужном идентификаторе и нажать комбинацию клавиш **Alt+Space**;
- навести указатель на нужный идентификатор и осуществить переход по **Ctrl+ЛКМ**.

При обнаружении одноименных идентификаторов предоставляется возможность выбора необходимой структуры кода. В таком случае при выполнении функции перехода откроется список подсказки, в котором следует выбрать нужную конструкцию с помощью клавиш навигации **Up/Down/End/Home** и осуществить переход клавишей **Left/Tab/Space/Enter** или мышью. Возврат к предыдущей конструкции осуществляется по **Ctrl+Z**.

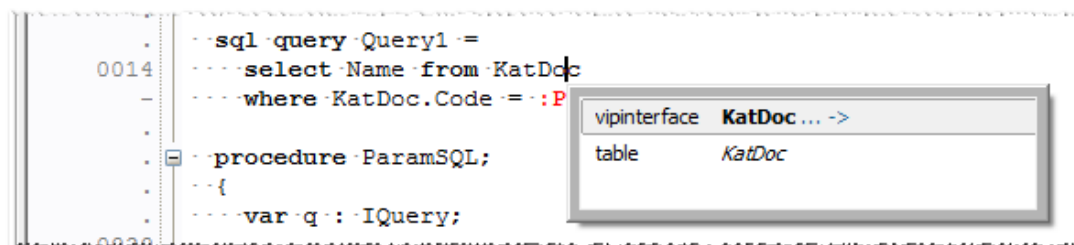


Рис. 59. Переход к описанию

Для функций, имеющих *forward*-описание, переход выполняется в обе стороны. Для *VipInterface* предусмотрена возможность перехода от описания к его реализации, поиск одноименной реализации осуществляется относительно "текущего" элемента сборки.

3.5.4.3. Вывод описания о текущей конструкции

Данная возможность позволяет просматривать описание конструкции во всплывающей подсказке. Для получения информации необходимо навести курсор мыши на нужный идентификатор, при этом в списке отобразятся все имеющиеся одноименные конструкции.

Вывод описания осуществляется для тех конструкций, которым доступна функция [Автодополнение](#)⁷⁵.

В режиме отладки описание по текущей конструкции недоступно, т. к. активируется подсказка о значении переменных в текущем контексте отладки.

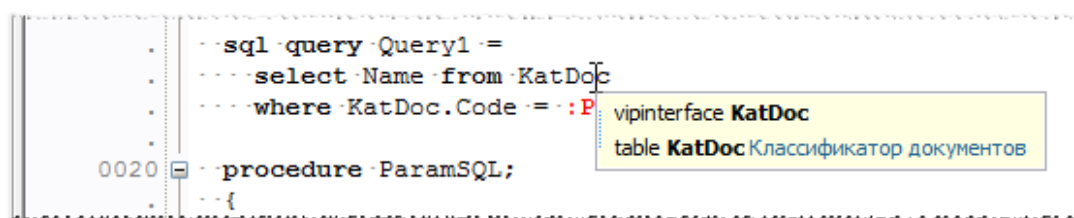


Рис. 60. Вывод описания о текущей конструкции

3.5.4.4. Подсказка параметров процедур и функций

Функция позволяет просматривать параметры процедур и функций, содержащихся в списке автодополнения, во всплывающей подсказке.

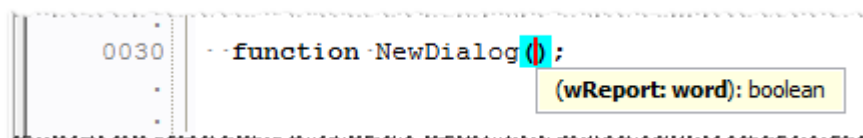


Рис. 61. Подсказка параметров процедур и функций

Для вызова подсказки следует установить курсор в параметрах вызываемой функции и активизировать комбинацию клавиш **Ctrl+Shift+Space**. В окне подсказки текущий параметр под курсором подсвечивается.

3.5.5. Функции рефакторинга

Функция *Форматор кода* выполняет приведение кода к стандартному виду, может применяться к активному редактору или выделенному абзацу. Активизировать функцию можно с помощью главного меню *Правка* и контекстного меню редактора кода.

В окне *Структура кода* существует возможность изменять расположение конструкций в коде путем их перемещения по дереву разбора. Переместить выбранную конструкцию можно способом Drag&Drop ("перетаскивание"), при этом *Строки* в окне *Структура кода* должны быть отсортированы по возрастанию. Для каждого перемещения автоматически производится проверка на соответствие правилам вложенности. Вместе с конструкцией переносятся и все маркеры (точки останова, метки и закладки), содержащиеся в ней.

3.6. Окно вкладок

Окно выбора вкладок предназначено для быстрого переключения между открытыми в окне редактора файлами и инструментальными окнами.

Данная возможность доступна с помощью команды *Окно выбора вкладок* контекстного меню на вкладке редактора кода или комбинации клавиш **Ctrl+Tab**.

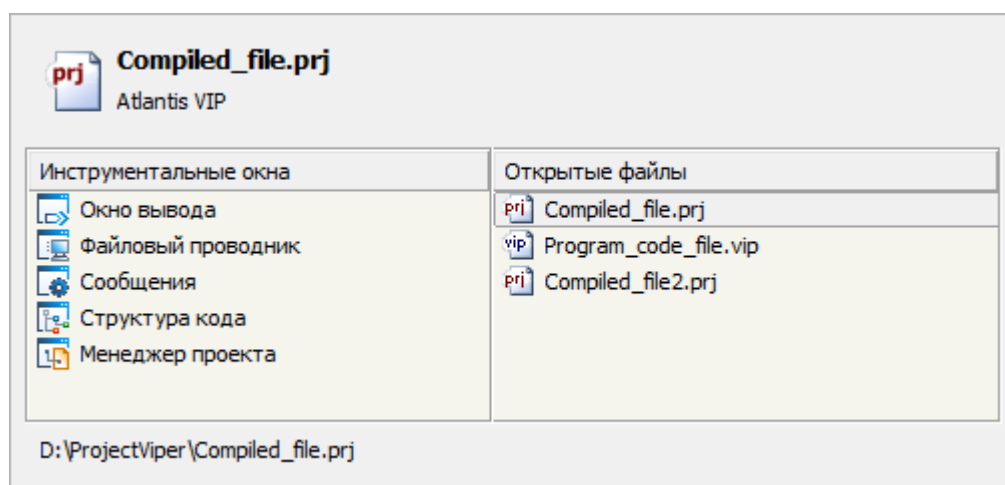


Рис. 62. Окно выбора вкладок

С помощью списка *Инструментальные окна* можно перемещать фокус по открытым окнам текущего режима рабочего стола *Viper*, которые пристыкованы к границам среды. По клавише **Del** в списке инструментальных окон отображаются все существующие окна среды, кроме отстыкованных.

Элементы списка *Открытые файлы* отсортированы в порядке открытия, причем последний открытый файл содержится в конце списка.

При наведении указателя на файл в нижней части окна выбора отображается полное имя файла.

Переход между списками может осуществляться клавишами направлений **Left/Right**.

Выбор одного из элементов осуществляет перевод фокуса в него, при этом окно выбора вкладок закрывается.

В списке *Открытые файлы* доступна возможность переключения между редакторами кода без закрытия окна вкладок. Для этого используются клавиши:

- **Ctrl+Tab** — вперед по списку;
- **Ctrl+Shift+Tab** — назад по списку.

Закрыть окно вкладок можно клавишей **Esc**.

В окне выбора вкладок дополнительно существует возможность фильтрации списка по мере ввода символов. Это способствует быстрому переключению между окнами и редакторами кода без использования мыши. Сбросить условия фильтра можно нажатием клавиши **Delete**.

Файл, открытый из рабочей копии (см. "[Меню TortoiseSVN](#)"⁵³), помечается соответствующим значком TortoiseSVN, обозначающим актуальность его состояния.

3.7. Статус-строка

Строка статуса предназначена для отражения информации о текущем состоянии среды разработки.

Для отображения статусной строки предназначен пункт главного меню *Вид > Статус-строка*.

Строка состоит из нескольких панелей:

- Первая панель — краткая подсказка о текущей команде, на которой находится указатель мыши (кнопка инструментальной панели, элемент меню). В остальных случаях в первой панели отображается путь на редактируемый файл.
- Вторая панель — текущая позиция курсора (строка и столбец). По двойному щелчку на данной панели открывается диалог перехода (см. "[Команда Перейти](#)"⁴⁴).
- Третья панель — статус активного файла. При изменении файла отображается статус *Модифицирован*, снять данный статус можно двойным щелчком на соответствующей панели.
- Четвертая панель — режим вставки *ВСТ* или замещения *ЗАМ* для редактируемого файла. Переключение режима закреплено за клавишей **Ins**.
- Пятая панель — используемая кодовая страница (*OEM*, *ANSI*). Двойной щелчок на данной панели меняет кодировку. Переключение кодовой страницы закреплено за клавишей **F6**.
- Шестая панель — состояние отслеживания внешних изменений, которое имеет вид шарика и в зависимости от цвета означает:
 - зеленый — отслеживание включено;
 - серый — отслеживание выключено;
 - красный — отслеживание включено, но не выполняется (например, когда файл не существует).

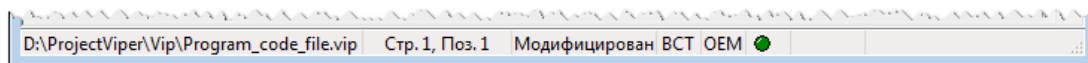


Рис. 63. Статус-строка

4. Проект

Проект является главной рабочей единицей в интегрированной среде разработки **Viper**. С ним связаны все основные задачи и возможности как редактора, так и остальных функциональных модулей.

Состав проекта определяет пользователь. Основные функции для этого содержатся в окне ["Менеджер проекта"](#)^[20].

В параметрах проекта определяются настройки компилятора и отладчика. Кроме этого, в проекте хранится состояние параметров отладчика: *Отлаживать все*, *Останавливаться на конструкторе*, *Перехватывать исключения*, а также информация о добавленных в файлы метках, закладках (см. ["Меню Поиск"](#)^[36]) и точках останова (см. ["Меню Отладка"](#)^[51]).

Работу в **Viper** следует начинать с открытия существующего проекта или создания нового, в случае разработки нового приложения (см. ["Создание проекта"](#)^[81]), т. к. без использования проекта **Viper** представляет собой лишь текстовый редактор.

Перейти к процессу компиляции и отладки можно только после выполнения настройки проекта (см. ["Параметры проекта"](#)^[86]).


4.1. Создание проекта

В интегрированной среде разработки **Viper** существует два способа создания проекта:

- При открытии среды **Viper** выбрать функцию *Создать новый проект* в окне *"Приветствие"*. Наличие окна определяется настройкой *Показывать диалог приветствия при старте* (см. ["Вкладка Общие — Просмотр"](#)^[107]).
- В открывшейся среде **Viper** вызвать команду главного меню *Проект > Новый*.

При создании проекта требуется указать название и директорию хранения файла с описанием проекта.

Данные о проекте хранятся в файле формата XML, имя которого состоит из имени проекта и расширения *vpr*.

 *Файл проекта (*.vpr) рекомендуется размещать в корневой папке самого проекта. Тогда все пути к его элементам будут автоматически сохраняться относительно данного каталога. Это позволяет использовать проект без изменения настроек в случае перемещения его каталога со всей структурой.*

При создании проекта предоставляется возможность заполнения параметров компиляции за несколько шагов в окне ["Мастер настройки проекта"](#)^[82].

Добавление файла в проект зависит от назначения файла и осуществляется несколькими вариантами (см. ["Состав проекта"](#)^[85]).

Сохранение изменений в проекте выполняется одним из способов:

- Автоматическое сохранение изменений в проекте при его закрытии либо закрытии приложения **Viper**.
- Сохранение в процессе работы с помощью главного меню *Проект > Сохранить* или *Сохранить как*.

Открытие созданного ранее проекта выполняется одним из способов:

- Команда главного меню *Проект > Открыть*.

- Команда главного меню *Проект > Последние проекты*, которая раскрывает список с историей последних используемых проектов.
- Функция *Открыть проект* в окне *Приветствие*, которая отображает список последних открытых проектов.

При отсутствии необходимого варианта в списке следует использовать параметр *Другие проекты*, который осуществляет открытие диалогового окна для выбора проекта на диске.

4.1.1. Мастер настройки проекта

Мастер создания проекта предоставляет упорядоченный процесс настройки проекта с помощью нескольких шагов.

Данная возможность позволяет быстро настроить обязательные параметры, необходимые для выполнения процесса компиляции.

Мастер доступен при включенной настройке *Использовать мастер настройки проекта* (см. "[Вкладка Общие - Просмотр](#)"¹⁰⁷) и активируется после сохранения файла нового проекта.

Не показывать мастер при создании проекта — отключение возможности использования мастера настройки проекта (подробнее см. "[Вкладка Компилятор VIP](#)"⁸⁹). При включенном параметре *Не показывать мастер при создании проекта* для закрытия окна появляется кнопка *[Завершить]*.

[Продолжить] — перейти к первому этапу настройки проекта.

[Отменить] — закрыть окно мастера.

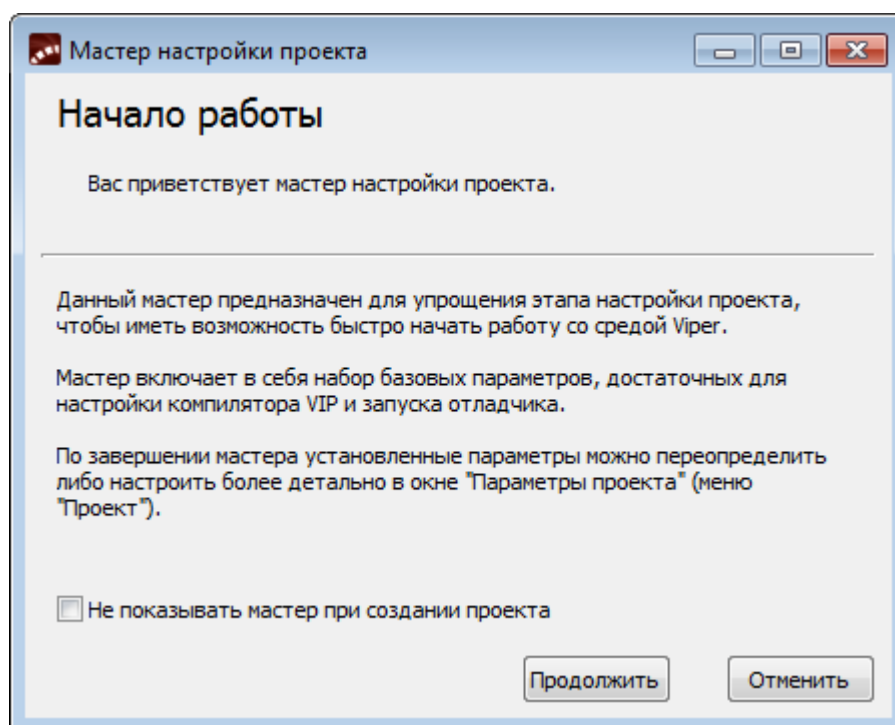


Рис. 64. Окно мастера настройки проекта

Каталог компилятора — каталог, в котором содержится встроенный компилятор (*ViperCompile.exe*) соответствующей версии. Подробнее о версиях см. в разделе "[Справка](#)"⁵⁵. Для изменения параметра предназначена кнопка . Значение данного параметра будет присвоено макропеременной *\$(AtlPath)*.

[Далее] — перейти к следующему этапу настройки проекта.

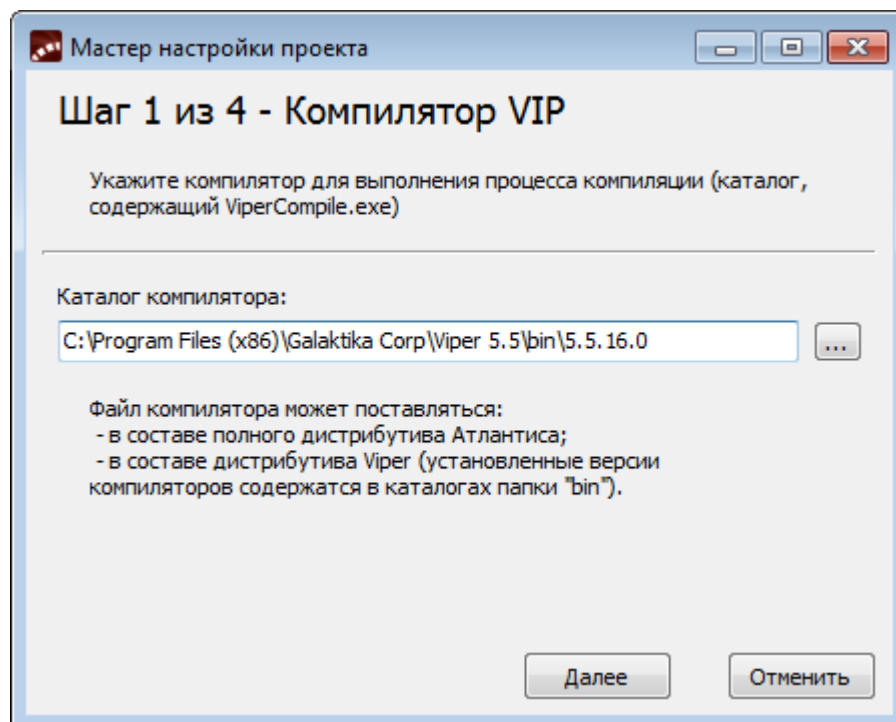


Рис. 65. Этап настройки компилятора VIP

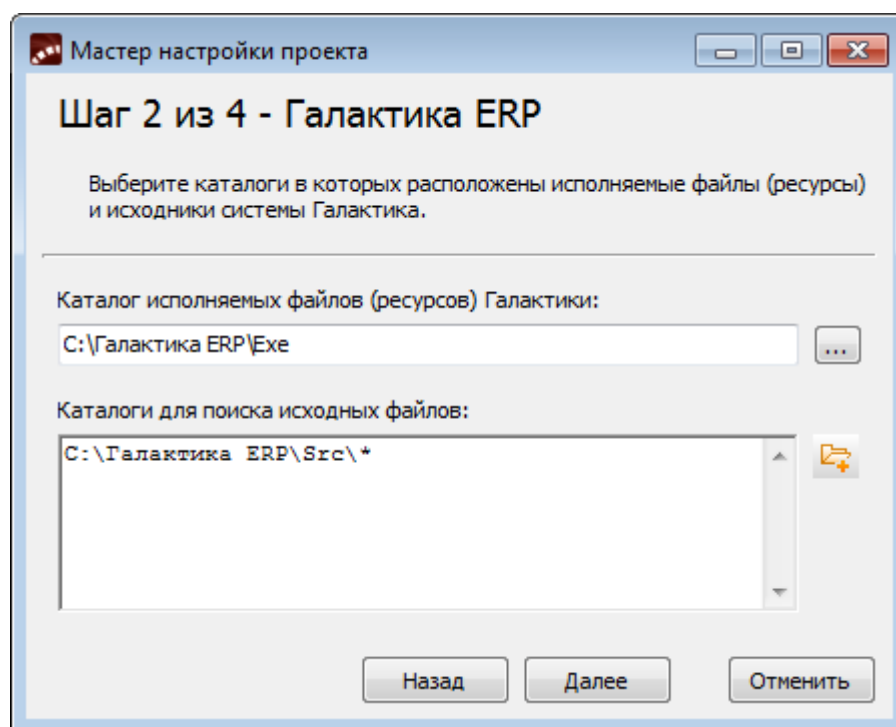


Рис. 66. Этап настройки компилятора VIP

Каталог исполняемых файлов (ресурсов) Галактики — каталог размещения ресурсного файла (*.res). Значение данного параметра будет присвоено переменной $[GalExe]$. Данная переменная по умолчанию используется и для поиска отлаживаемого приложения (AtIExec.exe) и конфигурационного файла.

Каталоги для поиска исходных файлов — каталоги подключаемых файлов приложения (подробнее см. "[Вкладка Компилятор VIP - Каталоги](#)"⁹¹). В данном параметре в

качестве обозначения вложенных папок можно добавить '*'. По завершении настройки проекта дополнительно автоматически добавится путь на ресурсы **Атлантиса**.

[Назад] — вернуться к предыдущему этапу настройки.

[Далее] — перейти к этапу настройки базы данных.

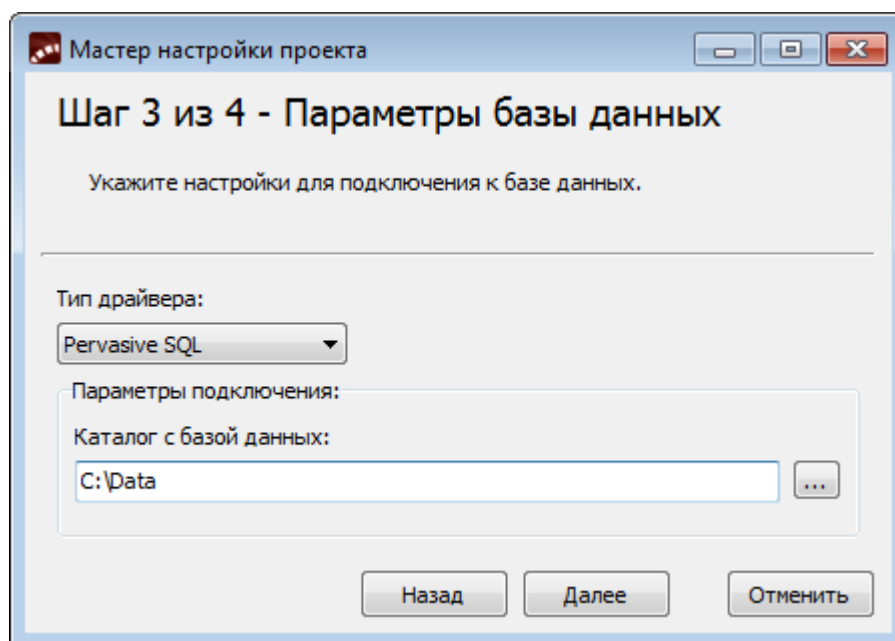


Рис. 67. Этап настройки компилятора VIP

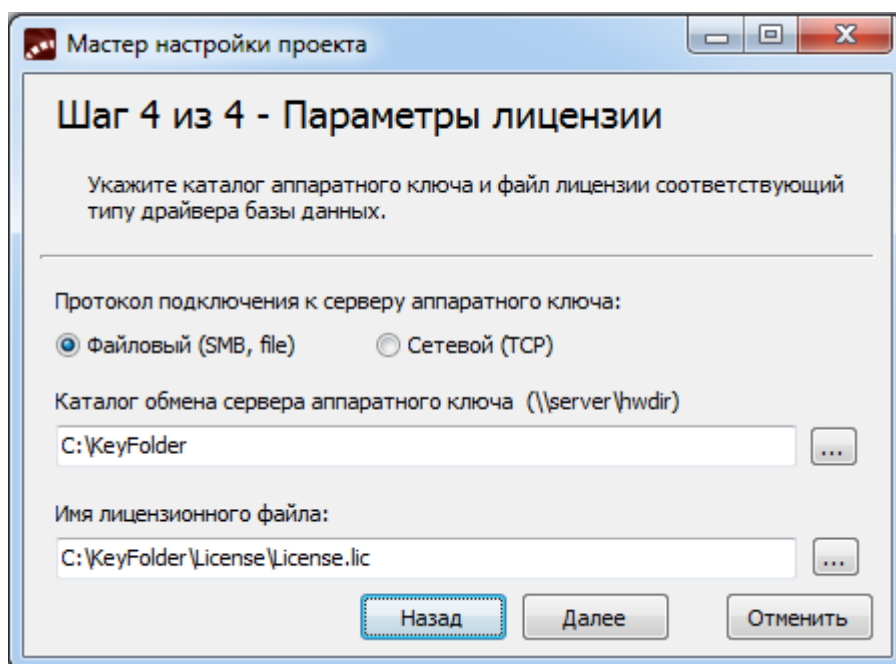


Рис. 68. Этап настройки параметров лицензии

На третьем этапе следует выбрать из выпадающего списка используемый **Тип драйвера** и заполнить соответствующие параметры базы данных (подробнее см. "[Вкладка Компилятор VIP - БД и лицензирование](#)⁹⁶").

[Назад] — вернуться к предыдущему этапу настройки.

[Далее] — перейти к настройке лицензионных файлов системы **Галактика ERP**.

На шаге 4 следует выбрать используемый **Протокол подключения к серверу аппаратного ключа** из предложенных вариантов и заполнить соответствующие параметры (подробнее см. "[Вкладка Компилятор VIP - БД и лицензирование](#)"⁹⁶).

[Назад] — вернуться к предыдущему этапу настройки.

[Далее] — перейти к завершающему этапу настройки проекта.

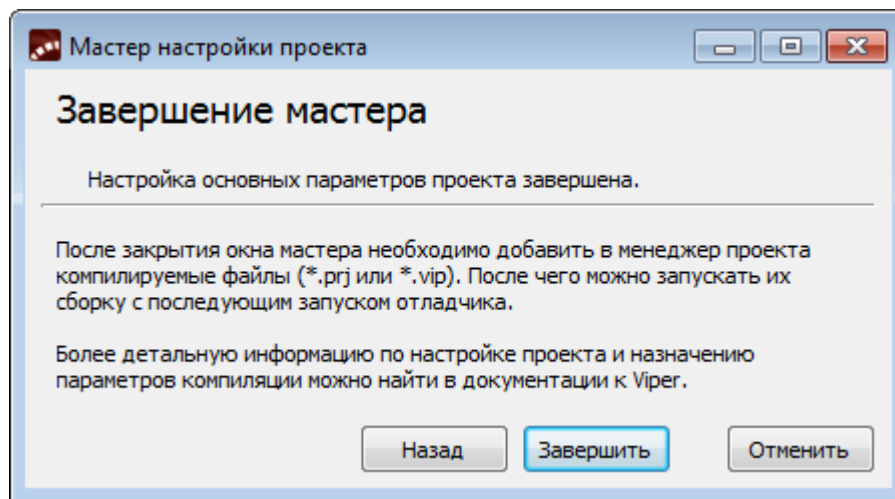


Рис. 69. Завершение работы с мастером настройки проекта

[Завершить] — закрыть окно мастера с сохранением параметров проекта.

[Отменить] — закрыть окно мастера без сохранения параметров проекта.

Мастер настройки проекта автоматически создает конфигурацию отлаживаемого приложения. По умолчанию имя конфигурации *Gal*. При наличии одноименной конфигурации к создаваемому имени *Gal* добавляется порядковый номер, при существовании идентичных параметров повторная конфигурация не создается. Для настройки **Имя исполняемого файла приложения** добавляется значение `[$GalExe]\AtlExec.exe`, в параметр **Текущий каталог** добавляется значение `[$GalExe]`.

Для создаваемых временных файлов мастер настройки автоматически добавит переменную `[$Tmp]` со значением `C:\Temp\Viper\` в соответствующие параметры проекта.

4.2. Состав проекта

Состав проекта определяется пользователем в окне =[Менеджер проекта](#)⁵⁸=.

В проект могут входить следующие объекты:

- файл — представляет собой ссылку на файл, расположенный на диске;
- элементы сборки — предназначены для компиляции и выполнения следующих элементов:
 - файл компиляции — файл, содержащий параметры для вызова встроенного компилятора. Такими объектами могут быть только ссылки на файлы с расширениями `*.prj`, `*.vip`, `*.frm`, `*.cnf`;
 - [профиль](#)¹⁴⁷ — список задач с набором параметров, предназначенных для запуска внешних приложений/компиляторов;
 - пакет — отдельная конфигурация параметров компиляции, позволяющая объединить несколько элементов проекта.

Пакет предназначен для создания группы элементов сборки с определенными параметрами компиляции отличных от проекта.

- Виртуальный каталог — элемент, объединяющий ссылки на файлы и другие каталоги для удобного представления структуры файлов.

Элементы сборки могут содержать вложенную иерархию объектов (виртуальные каталоги, ссылки на файлы) неограниченного уровня вложенности. Созданная пользователем структура проекта сохраняется в проектном файле (*.vpr). При открытии проекта загрузка пользовательской структуры файлов выполняется в соответствии с параметром *Загружать как в последнем сеансе* (см. "[Вкладка Проект](#)").

Проект содержит настройки параметров компилятора, влияющих на вложенные элементы (см. "[Параметры проекта](#)").

Элементы сборки (файлы компиляции и пакеты) по умолчанию наследуют настройки компилятора из параметров проекта, но при необходимости их можно изменить для каждого отдельного элемента сборки. В свою очередь элементы, вложенные в пакет наследуют его параметры.

Схематически наследование параметров компиляции представлено на рисунке ниже.

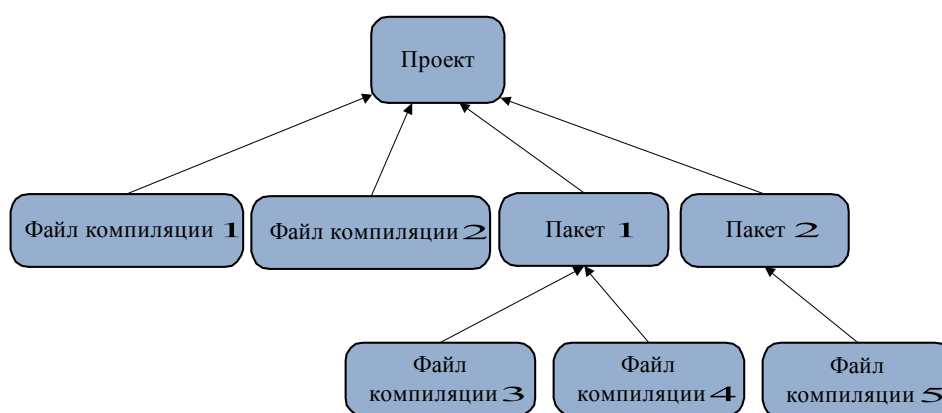


Рис. 70. Схема наследования параметров компиляции

4.3. Параметры проекта

Параметры проекта предназначены для настройки компилятора и отладчика. Настройка заключается в описании путей на файлы с исходными текстами, подключаемые файлы, ресурсные и т. д (см. "[Окно Параметры проекта](#)").

Настройка проекта выполняется в окне *Параметры проекта*.

Для вызова окна предоставляется выбор одного из способов:

- Команда главного меню *Проект > Параметры проекта*.
- Команда контекстного меню *Параметры* в *Менеджере проекта*.

Настройки компилятора по умолчанию наследуются каждым элементом сборки из параметров проекта (пакета). При необходимости их можно переопределить в параметрах необходимого файла компиляции или пакета.

Параметры проекта, за исключением данных отладчика Vip, хранятся в файле текущего проекта.

Начиная с версии **Viper 5.5.16**, параметры вкладки *Отладчик VIP* сохраняются в глобальном файле параметров среды **Default.vpr** (см. "[Настройка параметров среды](#)") и являются общими для всех проектов. Конфигурации из созданных ранее проектов переносятся в новое место хранения автоматически при открытии данных проектов. Повторные параметры не импортируются.


4.3.1. Окно Параметры проекта


Окно «Параметры проекта» включает группы настроек, необходимых для выполнения процесса компиляции и отладки. Для вызова окна существует несколько способов (см. «Параметры проекта»⁸⁶).

Окно содержит страницы с параметрами, сгруппированными в следующие разделы:

- Проект;
- Компилятор VIP;
- Отладчик VIP.

Настройки из разделов Проект и Компилятор VIP предназначены для компиляции. Параметры раздела Отладчик VIP используются во время отладки приложения.

Для гибкости настройки предусмотрен набор макропеременных¹⁰¹ (макросов). Список стандартных макропеременных среды можно выбрать в списке, открываемом по нажатию кнопки , пользовательские переменные добавляются путем ввода в поле настройки.

 В параметрах, содержащих список каталогов (например, Список каталогов для поиска подключаемых файлов⁹¹), имеется возможность указания вложенных каталогов символом "*".

В списках подключаемых файлов/каталогов (Список каталогов для поиска подключаемых файлов, Список дополнительно подключаемых ресурсных файлов, Дополнительный список каталогов для поиска файлов) доступна возможность множественного выбора при удерживании клавиши **Ctrl**. Также в данных параметрах существует проверка на наличие дублирующих записей. Повторные записи автоматически удаляются при перемещении фокуса.

[Продолжить] — сохранить измененные параметры, при этом выполняется автоматическое отключение от БД, если настройки базы данных⁹⁶ были изменены в условиях установленного подключения к ней.


[Отменить] — закрыть окно параметров проекта без сохранения изменений.

[Справка] — отобразить справочную документацию «Среда разработки Viper».

4.3.1.1. Вкладка Проект

Вкладка Проект позволяет определить имя проекта и особенность загрузки его файлов.

Имя проекта — имя проекта.

Имя файла — полное имя файла для хранения настроек проекта. Для изменения файла проекта предназначена кнопка .

Загрузка файлов проекта — режим загрузки файлов в редактор кода при открытии проекта:

- **Не загружать** — редактор кода в закрытом состоянии;
- **Загружать все** — загрузка всех файлов из структуры проекта;
- **Загружать как в последнем сеансе** — загрузка всех файлов, которые были загружены в предыдущей сессии.

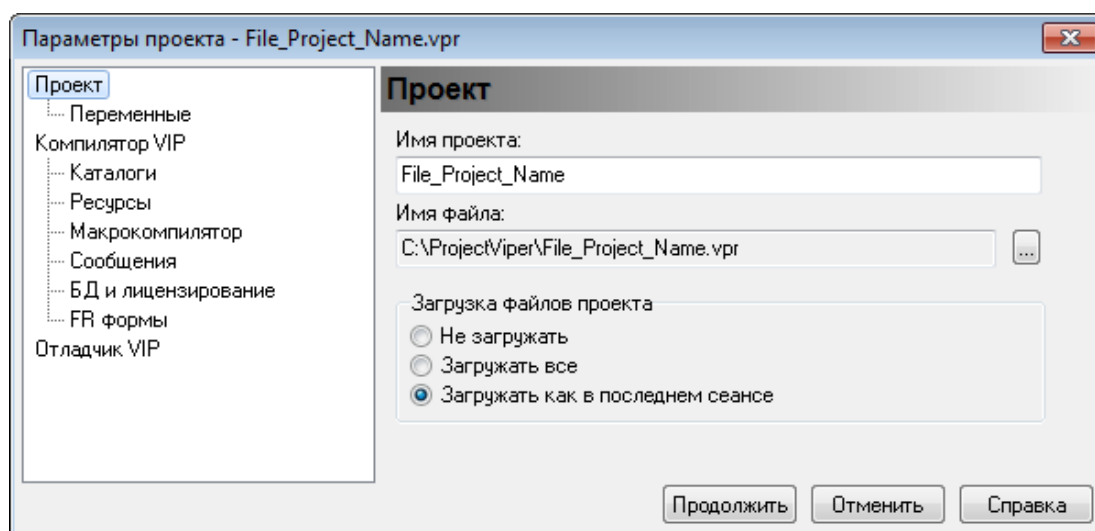


Рис. 71. Страница настроек "Проект"

4.3.1.1.1. Вкладка Проект - Переменные

Вкладка *Проект* > *Переменные* позволяет создавать собственные переменные, для их использования в параметрах элементов сборки текущего проекта.

Список пользовательских переменных — переменные, созданные пользователем или =Мастером настройки проекта=, которые можно использовать в параметрах данного проекта.

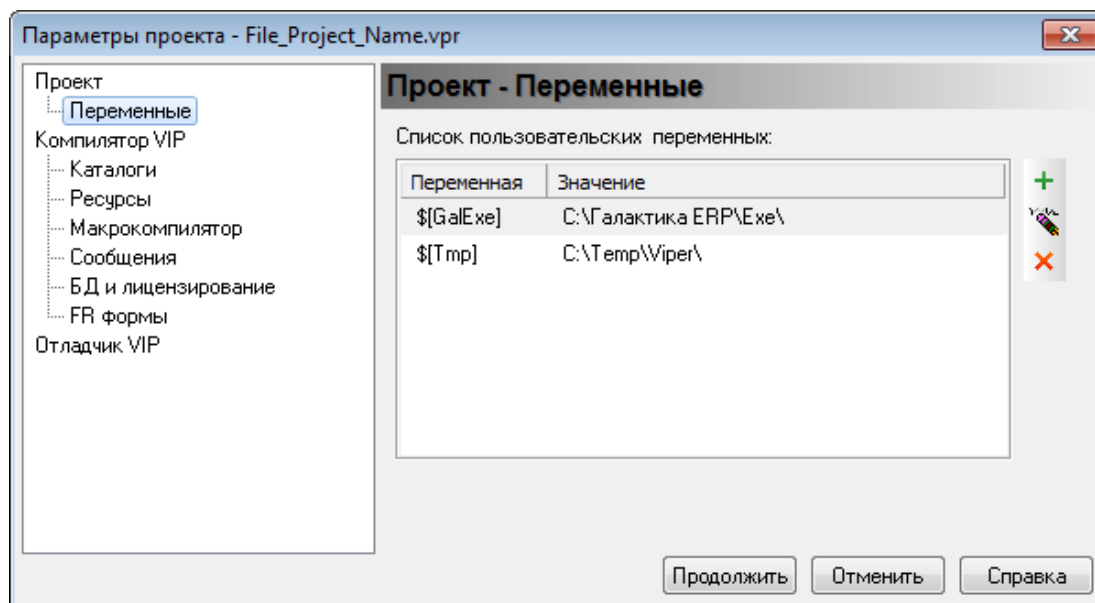





Рис. 72. Страница настроек "Проект-Переменные"

Функции изменения пользовательских переменных:

-  — создание новой пользовательской переменной;
-  — изменение параметров выбранной переменной из списка;
-  — удаление выбранной переменной из списка.

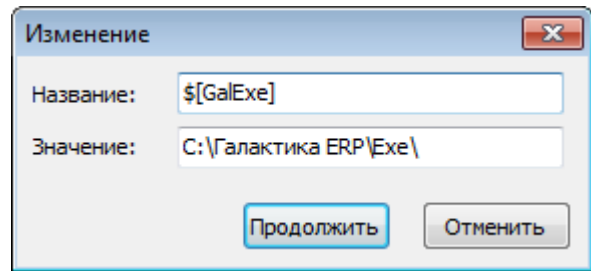


Рис. 73. Окно редактирования переменной параметров проекта

Название — имя переменной, например *GalExe*. При сохранении оно автоматически обрамляется символами '[' и ']', в таком формате (т. е. *\$[GalExe]*) следует добавлять переменную в необходимые настройки проекта.

Значение — путь к каталогу.


[Продолжить] — сохранить измененные параметры в списке пользовательских переменных.

[Отменить] — закрыть окно параметров переменной без сохранения.

Навигация по списку переменных осуществляется с помощью клавиш **Up/Down/End/Home** или **ЛКМ**.

4.3.1.2. Вкладка Компилятор VIP

Вкладка *Компилятор VIP* позволяет определять настройки компилятора для управления отладочной информацией и состоянием ресурса, а также использовать конфигурационный файл настроек проекта.

Имя текущего компонента — имя текущего компонента (соответствует параметру *VIP.ComponentName*). По нажатию кнопки  отображается список макропеременных для добавления в состав имени компонента: *Имя проекта*, *Имя файла проекта*, *Путь проекта*, *Имя текущего файла*, *Путь текущего файла*, *Путь на Атлантис* (см. "[Макропеременные в параметрах проекта](#)"¹⁰¹").

Использование макропеременных упрощает процесс настройки, т. к. компилируемых единиц в проекте может быть множество, а изменять этот параметр для каждого элемента в отдельности нецелесообразно. В таком случае, в имени текущего компонента рекомендуется указать макрос *\$[FileName]*. Тогда для всех подключенных в сборку файлов компиляции имя компонента будет назначаться по имени файла.

Отладочная информация — добавление отладочной информации в результат компиляции. Данная настройка необходима при планировании использования интерактивно отладчика и содержит следующие параметры:

- **Формировать отладочную информацию** (соответствует параметру *VIP.Debug*).
- **Включить отладочную информацию для локальных переменных** (соответствует параметру *VIP.LocalDebug*).

Использовать информацию о сборке из любых ресурсных файлов — перекомпиляция файлов, изменившихся со времени прошлой компиляции. Для этого используется информация из всех подключенных файлов ресурсов (соответствует параметру *Compilers.MakeUsingAllResourceFiles*).

Очищать рабочие ресурсы перед компиляцией — удаление ресурсных файлов перед компиляцией. К ним относятся: ресурсный файл, служебный ресурсный файл и рабочий ресурсный файл конфигулятора.

Сжимать ресурс — использование режима сжатия ресурса.

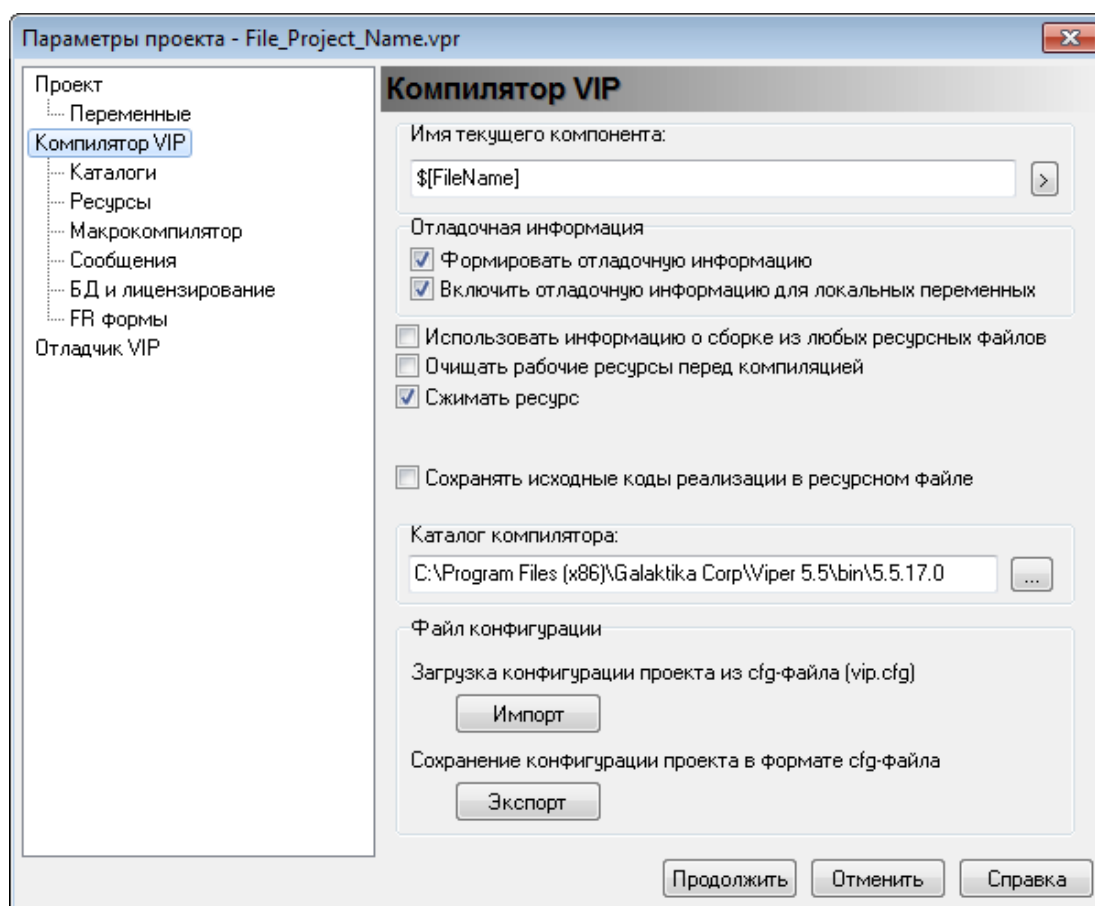



Рис. 74. Страница настроек "Компилятор VIP"

Уровень сжатия секции в ресурсе — степень сжатия в диапазоне уровня от 0 до 9, которая учитывается при включенном параметре **Сжимать ресурс**.

Сохранять исходные коды реализации в ресурсном файле — упаковка исходного кода в ресурс (соответствует параметру *Compilers.SaveSRC*). Параметр учитывается при пересборке ресурса или компиляции с включенной настройкой **Очищать рабочие ресурсы перед компиляцией**.

 Параметр учитывается с версией компилятора 5.5 (настройка **Каталог компилятора**) для элемента сборки.

Каталог компилятора — каталог, в котором расположен используемый компилятор (каталог **Атлантиса**). По умолчанию таким каталогом назначается папка исполняемого файла среды **Viper**. Данный параметр может быть перекрыт в настройках элементов сборки, и тогда при компиляции проекта для каждого элемента будет использована собственная версия **Атлантиса**.

При открытии проекта выполняется проверка на соответствие версии текущего **Атлантиса** (расположенного в папке открытой среды **Viper**) с версией из параметра проекта **Каталог компилятора**. Когда они не совпадают, пользователю предлагается заменить версию компилятора в проекте. Такая проверка выполняется для каждого проекта лишь единожды.

Файл конфигурации — использование конфигурационного файла настроек проекта:

- **[Импорт]** — загрузка конфигурации проекта из csg-файла (**Vip.csg**). Вызов данной команды открывает диалоговое окно для выбора файла конфигурации.

Перед выполнением импорта конфигурации текущие настройки автоматически сбрасываются.

- [Экспорт] — сохранение конфигурации проекта в формате sfg-файла. При активации данной команды открывается диалоговое окно сохранения файла конфигурации.

Результат импорта/экспорта параметров отображается в =[Окне вывода](#)⁶⁸=.

Дополнительные параметры, используемые при компиляции, расположены во вложенных узлах вкладки Компилятор VIP.

4.3.1.2.1. Вкладка Компилятор VIP - Каталоги

Вкладка Компилятор VIP > Каталоги содержит адреса к каталогам, используемым при компиляции.

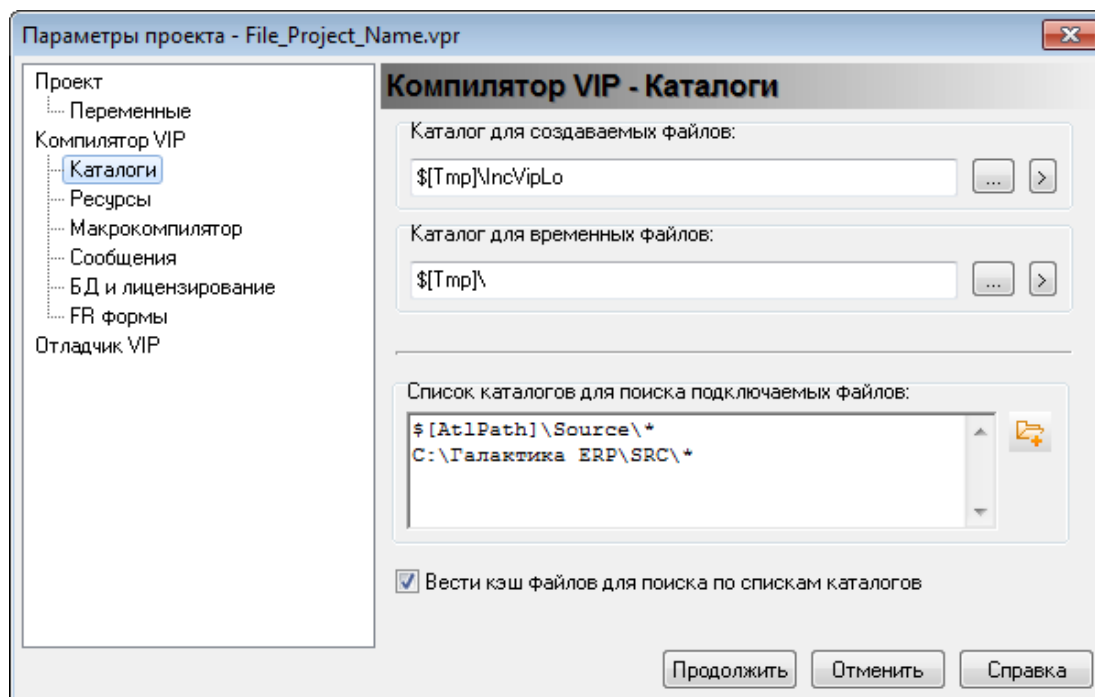






Рис. 75. Страница настроек "Компилятор VIP-Каталоги"

Каталог для создаваемых файлов — адрес каталога для хранения файлов с описанием констант, типов, элементов отчетов и других данных, которые используются в ходе компиляции (соответствует параметру *Files.OutputFilesDirectory*). По нажатию кнопки  отображается список макропеременных для добавления в состав пути: *Имя проекта*, *Имя файла проекта*, *Путь проекта*, *Имя текущего файла*, *Путь текущего файла*, *Путь на Атлантис*.

Каталог для временных файлов — адрес каталога для хранения временных файлов. Они необходимы для наполнения данными, которые используются в результирующем ресурсе (соответствует параметру *Files.TmpFilesDirectory*). В данном параметре доступен список макропеременных, описанный в параметре выше. Для изменения адреса каталога предназначена кнопка , которая осуществляет открытие стандартного диалогового окна.

Список каталогов для поиска подключаемых файлов — путь к каталогу подключаемых файлов. Он используется компилятором при поиске файлов, которые подключаются в исходном коде директивой `#include` (соответствует параметру *Compilers.IncludePath*). Изменить список можно с помощью клавиатуры в области ввода или кнопкой , которая активизирует стандартное диалоговое окно для выбора каталога. Дополнительно см. "[Окно Параметры проекта](#)"⁸⁷.


Для автоматического подключения/использования вложенных каталогов предназначен специальный символ `"*"`. При выполнении [Экспорта] конфигурации проекта (см. "[Вкладка Компилятор VIP](#)⁸⁹") в `cfg`-файле символ `"*"` будет автоматически "разворачиваться" в полный список каталогов.


 При выполнении компиляции поиск подключаемых файлов в первую очередь производится в каталоге файла компиляции (`*.prj`). Скрытые каталоги компилятором не учитываются.


Вести кэш файлов для поиска по спискам каталогов — кэширование имен файлов в каталогах (соответствует параметру `Compilers.IncludeCache`).

4.3.1.2.2. Вкладка Компилятор VIP - Ресурсы

Вкладка *Компилятор VIP* > *Ресурсы* содержит адреса ресурсных файлов, используемых при компиляции.

Имя ресурсного файла — полное имя ресурсного файла для сохранения результата компиляции (соответствует параметру `Compilers.OutputResourceName`). Для изменения файла предназначена кнопка , которая осуществляет открытие диалогового окна.

 Каталогом создаваемого ресурсного файла рекомендуется указывать каталог исполняемой части (`EXE`) приложения.

В качестве имени создаваемого ресурса можно использовать макроопределение `$(FileName)`, тогда, при компиляции для всех подключенных в сборку файлов имя ресурса будет назначаться по имени файла. При нажатии кнопки  отображается список макропеременных: *Имя проекта*, *Имя файла проекта*, *Путь проекта*, *Имя текущего файла*, *Путь текущего файла*, *Путь на Атлантис*, *Расширение ресурса (.res)* (см. "[Макропеременные в параметрах проекта](#)¹⁰¹").


Например, в приведенном ниже примере каждый элемент сборки компилируется в ресурс с именем единицы сборки и со стандартным расширением для ресурсного файла:

```
exe\$(FileName)$(ExtRes)
```

Таким же образом следует настроить следующие параметры ресурсных файлов.

Имя служебного ресурсного файла — полное имя служебного ресурсного файла (соответствует параметру `Compilers.SubServientResource`), который содержит имена, адреса и другие параметры методов, используемых в исходном коде. В данном параметре доступен список макропеременных, описанный в параметре выше.

Имя рабочего ресурсного файла конфигулятора — полное имя рабочего ресурса, содержащего изменения, произведенные пользователем посредством конфигулятора (соответствует параметру `Configurator.Resource`). В данном параметре доступен список макропеременных, описанный в параметре выше.

 В качестве каталога в данных параметрах рекомендуется указывать временную папку для выполнения ее периодической очистки, т. к. предыдущие версии данных ресурсов могут влиять на результаты компиляции.

Название системы в репозитории — имя текущей системы из репозитория (соответствует ключу `/env` компилятора **VIP**), которая используется компилятором для подключения списка дополнительных ресурсов. По нажатию кнопки [Загрузить список доступных Атлантис-сисем] производится подключение к БД и существующие системы добавляются в выпадающий список. После сохранения выбранной системы остальной список очищается.

Базовый каталог для ресурсов репозитория — адрес каталога, содержащего ресурсы приложения (соответствует макропеременной `%StartPath%` в описании пути на ресурс в репозитории). Автоматически загружает путь, указанный в Support, при необходимости его можно изменить. Параметр не учитывается, когда для **Названия системы в репозитории** значение `<не задано>`.

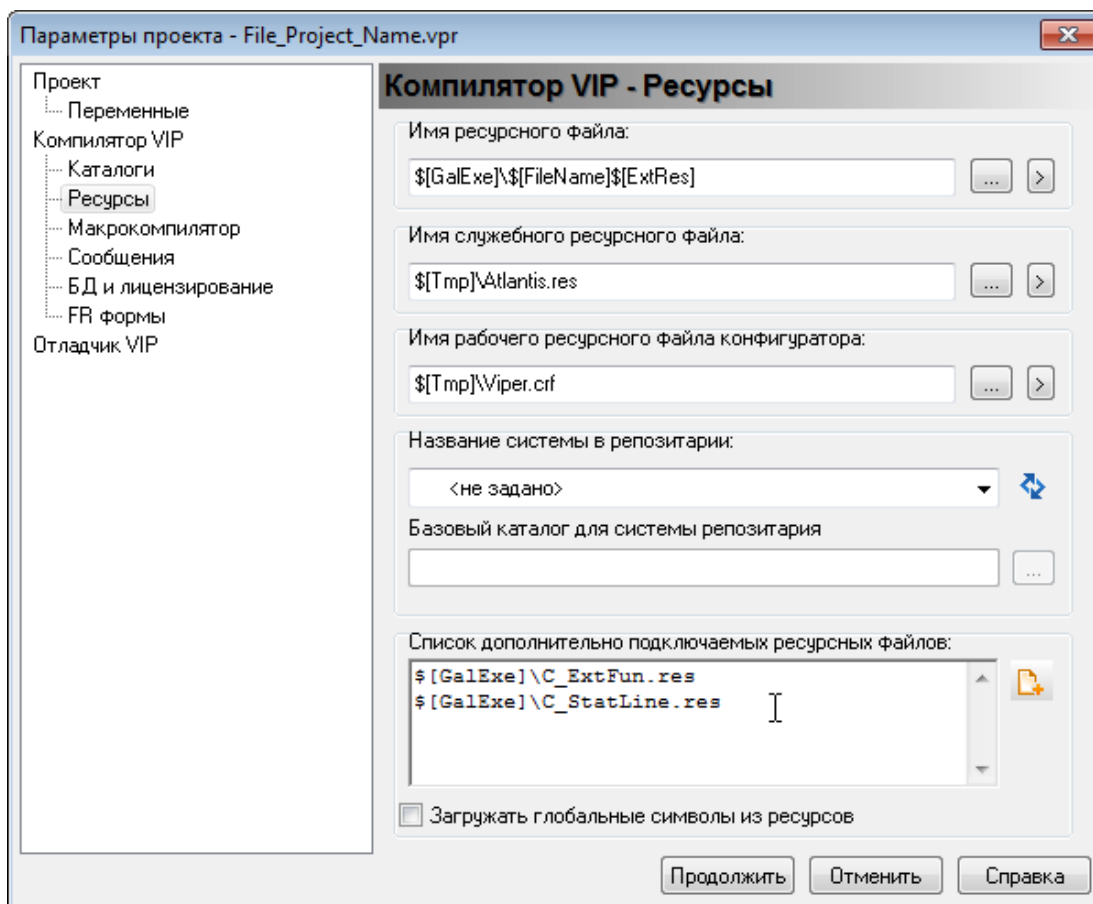



Рис. 76. Страница настроек "Компилятор VIP-Ресурсы"

Список дополнительно подключаемых ресурсных файлов — полные имена ресурсов, автоматически загружаемых при запуске приложения (соответствует параметру `System.OpenResources`). Список учитывается, когда в поле **Название системы в репозитории** установлено значение `<не задано>`. Добавить подключаемый ресурс в список можно путем набора пути с помощью клавиатуры в области ввода или кнопкой , которая активизирует стандартное диалоговое окно для выбора файла. Дополнительно см. "[Окно Параметры проекта](#)"⁸⁷".

Загружать глобальные символы из ресурсов — загрузка глобальных идентификаторов из ресурсов (соответствует параметру `Compilers.LoadIds`). При включенном параметре в процессе компиляции создается файл глобальных идентификаторов, по умолчанию `atlantis.idf`, и сохраняется по адресу, указанному в параметре **Каталог для временных файлов**.

4.3.1.2.3. Вкладка Компилятор VIP - Макрокомпилятор

Вкладка *Компилятор VIP > Макрокомпилятор* содержит параметры, влияющие на обработку исходного кода перед компиляцией.

Дополнительные макропеременные — константы или арифметические выражения (содержащие константу и имена макропеременных ранее определённых), которые используются для управления процессом компиляции.

Учитывать регистр символов в идентификаторах — чувствительность макропроцессора к регистру символов в идентификаторах (соответствует параметру *Macro.CaseSense*).

Проверять вложенность директив в подключаемых файлах — проверка вложенности директив в подключаемых файлах.

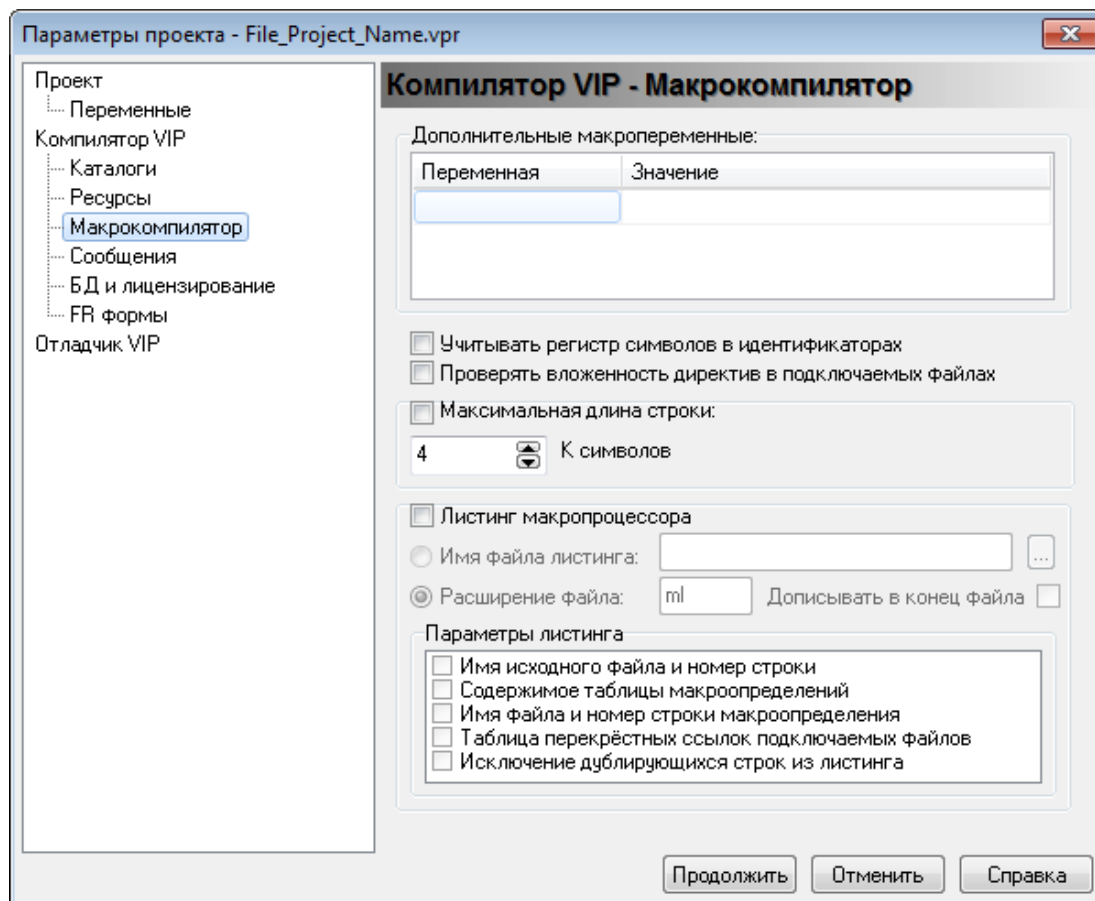



Рис. 77. Страница настроек "Компилятор VIP-Макрокомпилятор"

Максимальная длина строки — максимальная обрабатываемая длина строки входного файла (1024..32767, параметр *Compilers.MaxLineLength*).

Листинг макропроцессора — формирование листинга макропроцессора (соответствует параметру *Macro.ListEnable*). Данный параметр предоставляет выбор варианта формирования файла листинга:

- **Имя файла листинга** — полное имя файла листинга макропроцессора (соответствует параметру *Macro.ListFileName*). Для выбора файла предназначена кнопка , которая осуществляет открытие диалогового окна.
- **Расширение файла** — расширение для файла листинга макропроцессора (соответствует параметру *Macro.ListExtension*).

Дописывать в конец файла — добавление листинга к существующему файлу (соответствует параметру *Macro.ListAppend*).

Параметры листинга:

- **Имя исходного файла и номер строки** — вывод имени исходного файла и номера строки, в нем, в начало каждой записи файла листинга (соответствует параметру *Macro.ShowSource*);
- **Содержимое таблицы макроопределений** — вывод содержимого таблицы макроопределений в конец файла листинга (соответствует параметру *Macro.ListDefines*);
- **Имя файла и номер строки макроопределения** — при формировании таблицы макроопределений для каждого элемента выводить имя файла и номер строки, в которой был описан этот элемент (соответствует параметру *Macro.DefineSource*);
- **Таблица перекрестных ссылок подключаемых файлов** — вывод, в конце файла листинга, таблицы перекрестных ссылок между исходным и подключаемыми файлами (соответствует параметру *Macro.SourceCrossRef*);
- **Исключение дублирующихся строк из листинга** — исключение из файла листинга дублирующихся строк, формирующихся из-за повторной обработки компилятором одного и того же фрагмента текста (соответствует параметру *Macro.ListNoDup*).

4.3.1.2.4. Вкладка Компилятор VIP - Сообщения

Вкладка *Компилятор VIP > Сообщения* содержит параметры оповещений, которые формируются в процессе компиляции, при обнаружении нарушений правил языка **VIP** в исходном коде.

Компилировать до первой ошибки — остановка процесса компиляции при обнаружении первой "ошибки" (соответствует параметру *Compilers.Full*), которая отобразится в окне =Сообщения=.

Уровень диагностики — тип отображаемых сообщений компилятора (соответствует параметру *Compilers.ErrorLevel*):

- **Все сообщения** — сообщения любого типа. Данный вид диагностики позволяет использовать настройку **Выдавать информацию о причине перекомпиляции файлов** (соответствует *Compilers.VerboseMake*).
- **От подсказок и выше** — от сообщений с предложением по улучшению до ошибок, включительно;
- **Предупреждения и ошибки** — информация о нежелательном отражении на будущих событиях и сообщения об ошибках кодирования;
- **Только ошибки** — сообщения об ошибках кодирования.

Параметры компилятора — расширенные параметры сообщений компилятора:

- **Усилить "строгость" языка** — проверка соблюдения максимального количества правил (соответствует параметру *Vip.StrictVip*).
- **Отключить автоматическую генерацию констант** — отключение генерации констант для неизвестных идентификаторов (соответствует параметру *Vip.IgnoreNewConstants*).
- **Реагировать на пустые параметры** — оповещение о пустых параметрах.
- **Отдавать предпочтение константам** — при поиске идентификаторов оказание предпочтения константам (соответствует параметру *Compilers.ConstantPreference*).

Предупреждения — виды информационных сообщений:

- **Нет описания лицензии интерфейса** — отсутствие описания лицензии в интерфейсах (соответствует параметру *Compilers.DisableIfcLicWarnings*);

- **В компоненте отсутствуют таблицы** — отсутствие таблиц в компоненте (соответствует параметру *Vip.NoTablesInComponent*);
- **Одинаковые имена областей ввода** — наличие одинаковых имен конструкций *screen/browse* (соответствует параметру *Vip.DupNamesWarning*);
- **Требуется явное преобразование типа** (соответствует параметру *Vip.BaseTypesWarning*) для:
 - целых чисел;
 - вещественных чисел;
 - строк;
 - даты и времени.

Сохранять историю сообщений компилятора — сохранение информации по обнаруженным компилятором ошибкам, в окне «Сообщения».

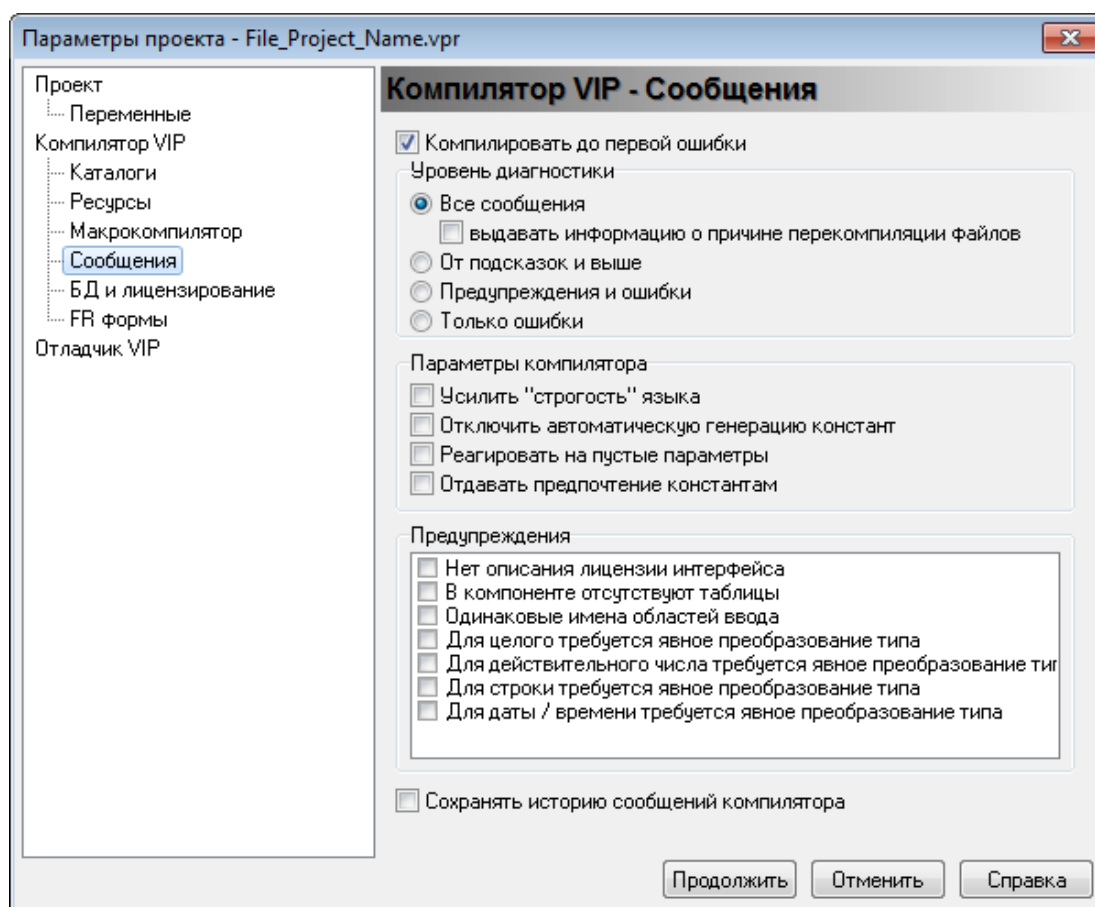



Рис. 78. Страница настроек "Компилятор VIP-Сообщения"

4.3.1.2.5. Вкладка Компилятор VIP - БД и лицензирование

Вкладка *Компилятор VIP > БД и лицензирование* содержит параметры подключения к базе данных и настройку лицензионных файлов, определяющих доступ к программному обеспечению.

Конфигурация — группа, содержащая набор параметров для подключения к базе данных. Добавленные конфигурации текущего проекта содержатся в выпадающем списке данного поля. Выбранная конфигурация, в списке параметров проекта, по умолчанию наследуется для каждого элемента сборки.

Создать конфигурацию можно только в параметрах проекта, но при необходимости ее настройки можно переопределить в параметрах необходимого файла компиляции или пакета.

 В случае конвертации структуры проекта в новый формат (см. [предупреждение](#)⁵⁰) или при выполнении импорта параметров из *sfg*-файла, настройки базы данных и лицензии сохраняются в конфигурацию "по умолчанию".

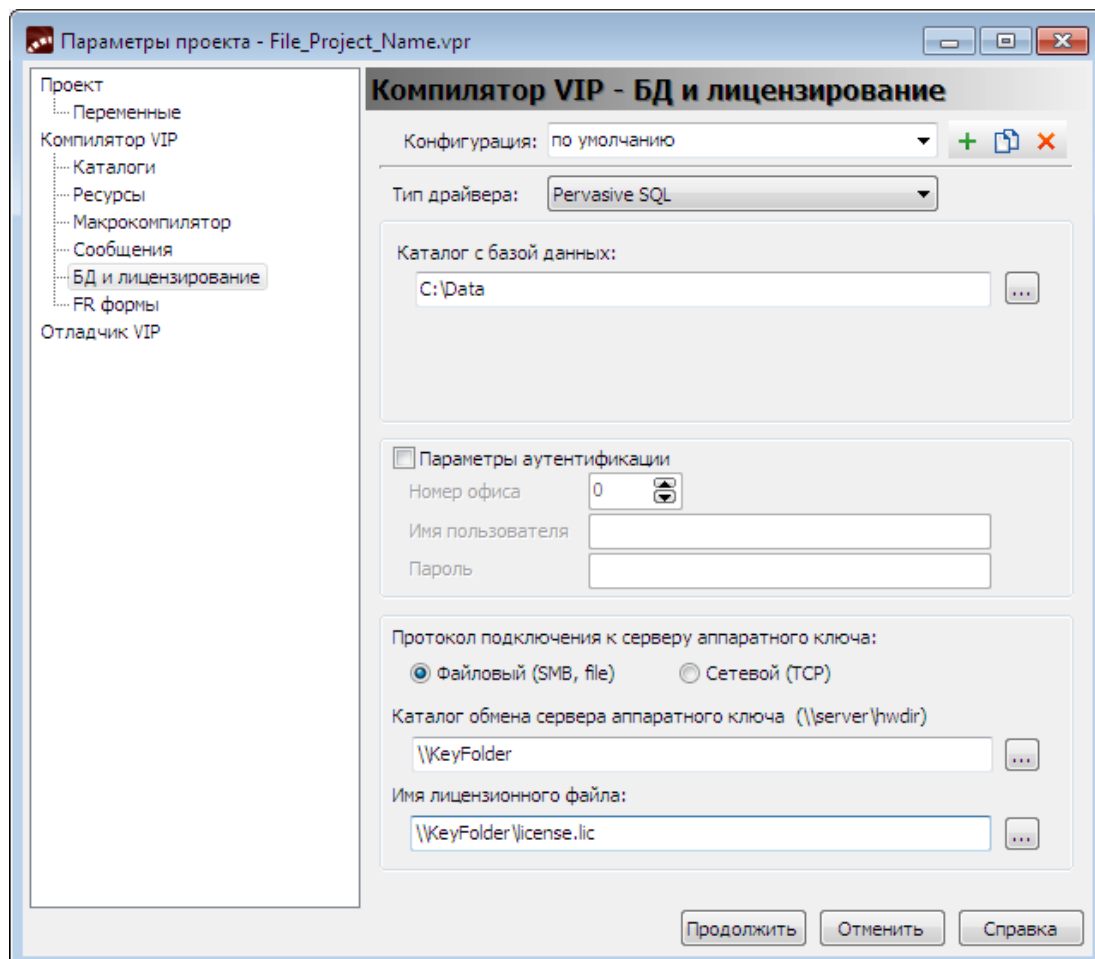






Рис. 79. Страница настроек "Компилятор VIP-БД и лицензирование"

Функции управления конфигурациями:

-  **Ctrl+Enter** — добавление конфигурации, при этом создается новая запись с именем *Безымянная <порядковый номер>*. Для изменения имени выбранной конфигурации необходимо ввести нужное название в поле и переместить фокус.
-  — копирование параметров выбранной конфигурации в новую.
-  **Ctrl+Del** — удаление выбранной конфигурации в списке.

Сортировать конфигурации в списке можно с помощью комбинации клавиш **Ctrl+Up/Down**.

Тип драйвера — драйвер базы данных, используемой при компиляции (соответствует параметру *DataBase.DataBaseDriver*). В параметре представлен выбор необходимого драйвера из выпадающего списка: *Pervasive SQL*, *Oracle 9.0 и выше*, *MS SQL* или *PostgreSQL*. Для каждого драйвера доступны определенные настройки.

Каталог с базой данных — адрес каталога базы данных (соответствует параметру *DataBase.DataBaseName*). Выбрать каталог можно с помощью кнопки . Настройка доступна при использовании драйвера *Pervasive SQL*.

Имя схемы базы данных — имя базы данных (соответствует параметру *DataBase.DatabaseName*). Настройка доступна при использовании драйвера *Oracle* и *PostgreSQL*.

Имя источника данных — имя экземпляра СУБД (соответствует параметру *SQLDriver.SQLServer*). Настройка доступна при использовании драйвера *Oracle*, *PostgreSQL* и *MS SQL*.

Параметры подключения к службе — адрес SQL-сервера (соответствует параметру *SQLDriver.SQLServer*). Настройка доступна при использовании драйвера *MS SQL*.

Использовать имя базы данных как префикс — поиск таблиц по имени базы данных (аналог *FullLoginName*). Настройка доступна при использовании драйвера *MS SQL*, *PostgreSQL* и *Oracle*.

Параметры аутентификации — дополнительные параметры для подключения к БД при компиляции, которые можно задать одним из способов:

- Вручную задать следующие идентификационные данные:
 - **Номер офиса**, к которому относится пользователь БД.
 - **Имя пользователя**.
 - **Пароль**.


Следует отметить, что **Имя пользователя** и **Пароль** сохраняются в файл проекта в открытом виде.

В случае отсутствия необходимых параметров аутентификации при запуске компиляции отобразится диалоговое окно, в котором следует ввести соответствующую информацию.


- Включить параметр **NTSecurity** для автоматической идентификации пользователя в БД средствами ОС Microsoft Windows. Данный параметр доступен, когда значением настройки **Тип драйвера** является *MS SQL*, *Oracle 9.0 и выше*, *PostgreSQL*.


Параметры **Номер офиса**, **Имя пользователя** и **Пароль** недоступны при включенной настройке **NTSecurity**.


Когда подключение к базе данных установлено, при сохранении изменений вышеуказанных параметров выполняется автоматическое отключение от БД.

 При отсутствии подключения к БД или несоответствии версии словаря БД процесс компиляции выполняется, но корректная работоспособность созданного ресурсного файла в таком случае не гарантируется.


Протокол подключения к серверу аппаратного ключа — возможность выбора протокола, используемого для обмена с сервером аппаратного ключа. Следует выбрать один из предложенных вариантов протокола и заполнить соответствующие параметры, представленные ниже.

- **Файловый (SMB, file)** — используется, если запросы и ответы сервера аппаратного ключа будут формироваться в виде файлов в сетевой папке, на которую должны быть предоставлены соответствующие права доступа требуемым пользователям.
 - **Каталог обмена сервера аппаратного ключа (\\server\hwdir)** — адрес расположения ключа (соответствует параметру *HardwareKey.SharedPath*), предназначенного для защиты ПО. Для выбора каталога предназначена кнопка , которая осуществляет открытие стандартного диалогового окна;

- **Имя лицензионного файла** — полное имя лицензионного файла (соответствует параметру *LicParam.LicFileName*). По нажатию кнопки  активизируется стандартное диалоговое окно для выбора файла. В зависимости от выбранного Типа драйвера устанавливается соответствующий фильтр типа файла (*.lic, *.btr, *.sql, *.ora или *.pgs), при необходимости можно выбрать фильтр *Все файлы*.
- **Сетевой (TCP)** — используется, если запросы и ответы сервера аппаратного ключа будут передаваться бинарным массивом данных через протокол TCP.
 - **Имя компьютера и порт сервера аппаратного ключа (server:port)** — настройки, необходимые для работы протокола (порт по умолчанию имеет значение 55555).
 - **Имя лицензионного файла** — используется аналогично такому же параметру для протокола **Файловый (SMB, file)**.

 Протокол **Сетевой (TCP)** поддерживается с версии компилятора **5.5.25.0** и выше.

 Следует учесть, что параметр **Сетевой (TCP)** автоматически отключается после открытия проекта (*.vpr) в среде **Viper** ниже версии **9.1.20.0**.

 При отсутствии ключа или лицензии компилятор будет запущен в режиме демо-версии, т. е. процесс компиляции отработает стандартно, но ресурс не сформируется.

4.3.1.2.6. Вкладка Компилятор VIP - FR формы

Вкладка Компилятор VIP > FR формы содержит параметры сборки форм отчетов.

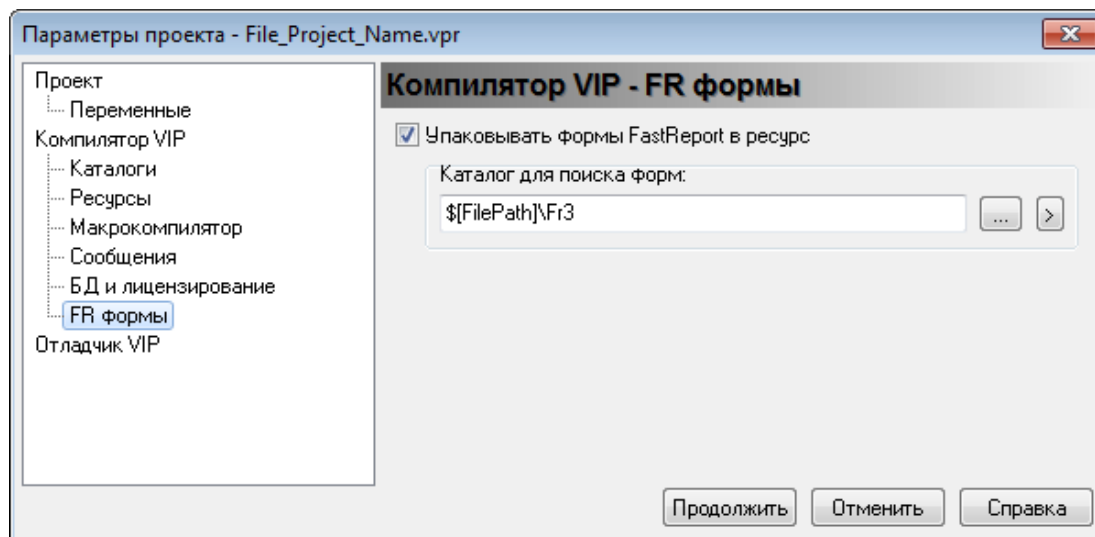



Рис. 80. Страница настроек "Компилятор VIP - FR формы"

Упаковывать формы FastReport в ресурс — выполнение автоматической упаковки FR-отчетов при выполнении компиляции/сборки ресурса. При использовании данной опции необходима настройка следующих параметров упаковки форм FastReport:

- **Утилита упаковки форм** — полное имя утилиты, с помощью которой осуществляется упаковка отчетов.
- **Каталог для поиска форм** — адрес каталога, содержащего формы отчетов. В процессе упаковки все формы отчетов из данного каталога загружаются в ресурс. По


нажатую кнопки  отображается список макропеременных для добавления в состав пути: *Имя проекта, Имя файла проекта, Путь проекта, Имя текущего файла, Путь текущего файла, Путь на Атлантис*.

Информация из файла **frf_res.log**, формирующегося в результате выполнения утилиты, выводится в окне «Сообщения», сам файл при этом удаляется.


4.3.1.3. Вкладка Отладчик VIP

Вкладка *Отладчик VIP* содержит параметры запуска отлаживаемого приложения с возможностью хранения нескольких конфигураций.

Конфигурация — группа, содержащая набор параметров, влияющих на запуск отлаживаемого приложения. Созданные конфигурации текущего проекта содержатся в выпадающем списке данного поля.


 Конфигурация отладчика **VIP** включает в себя все настройки вкладки *Отладчик VIP* за исключением параметра **Автоматически запускать отладку после компиляции**.

Функции управления конфигурациями отладчика **VIP** см. в разделе "[Вкладка Компилятор VIP - БД и лицензирование](#)^[97]". Перечень существующих конфигураций отладчика **VIP** доступен в выпадающем списке инструментальной панели *Отладка*^[55].


 При конвертации структуры проекта в новый формат (см. [предупреждение](#)^[50]), параметры *Отладчика VIP* сохраняются конфигурацией с именем по умолчанию. При создании проекта с помощью мастера автоматически создается конфигурация с именем *Gal*.

Имя исполняемого файла приложения — полное имя исполняемого файла приложения. Например:


- *Atlexec.exe* — для отладки **Галактики** или другого **Атлантис**-приложения;
- *Vip.exe* — для отладки отдельного интерфейса (необходимо добавить ключ */r* в **Параметрах приложения**).

По нажатую кнопки  отображается список стандартных макропеременных среды для добавления в состав пути: *Имя проекта, Имя файла проекта, Путь проекта, Путь на Атлантис* (см. "[Макропеременные в параметрах проекта](#)^[101]").

Текущий каталог — адрес каталога, содержащий файл конфигурации отлаживаемого приложения (например, **Galnet.cfg** для **Галактики**). В данном параметре можно использовать макропеременные, описанные в параметре выше.

 Адрес текущего каталога может не совпадать с путем отлаживаемого приложения.

Параметры приложения — ключи, влияющие на параметры загрузки приложения и установку некоторых настроек. Например, для запуска отлаживаемого приложения посредством **Atlexec.exe** может потребоваться ключ */client.application*. Подробнее об используемых параметрах следует смотреть документацию по запуску приложения. В данном параметре можно использовать макропеременные, описанные в параметре выше.

 Использование компилятора **VIP** и утилиты **ATLEXEC** см. в документе "*Средство разработки Атлантис. Инструменты и утилиты*".

Дополнительный список каталогов для поиска файлов — список каталогов, в которых расположены файлы, используемые в процессе отладки. Параметр предназначен

для поиска отлаживаемого файла, когда он отсутствует в *Списке каталогов для поиска подключаемых файлов* (см. рис. "[Вкладка Компилятор VIP — Каталоги](#)⁹¹"). Добавление каталога в данных параметрах производится одинаково. Дополнительно см. "[Окно Параметры проекта](#)⁸⁷". При отсутствии файла в списках каталогов в момент его отладки отображается диалоговое окно для выбора файла на диске (см. "[Настройка отладчика](#)¹³³").

Автоматически запускать отладку после компиляции — авто-выполнение команды [Начать отладку](#)⁵¹ после завершения процесса компиляции без *Ошибок* (см. "[Окно сообщений компилятора](#)⁶⁹").

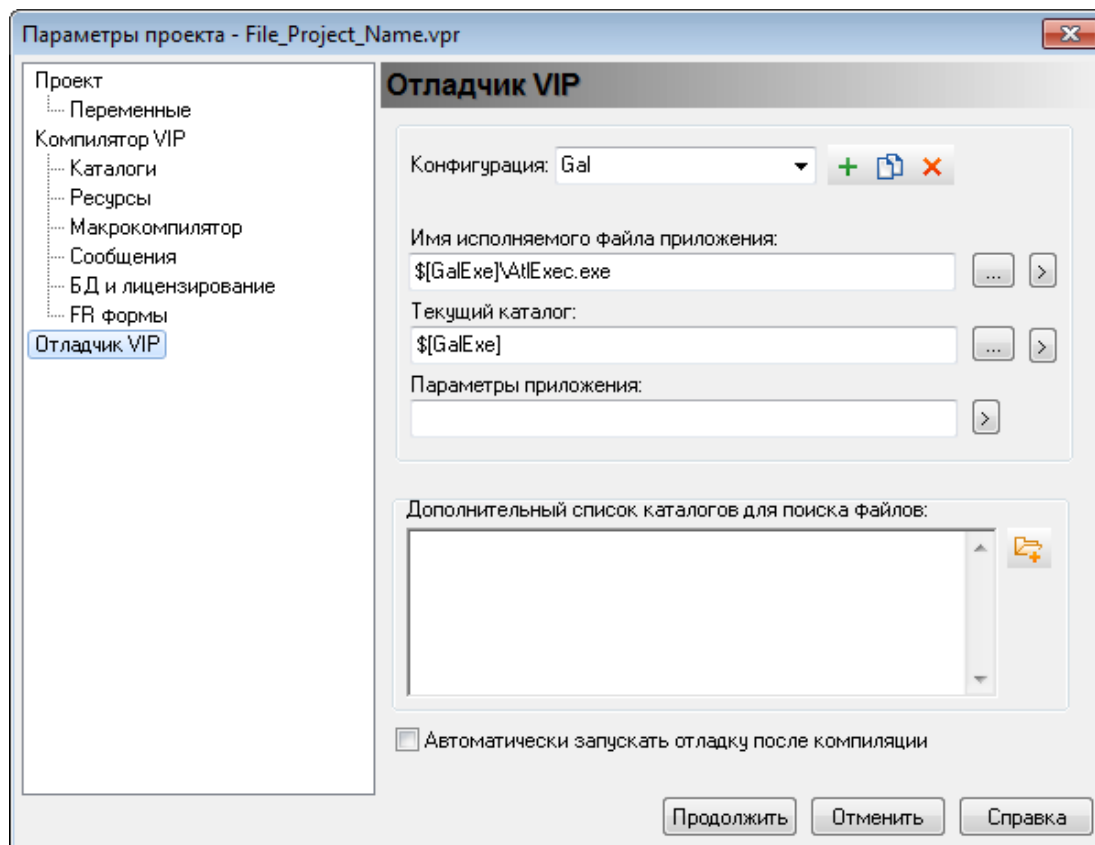



Рис. 81. Страница настроек "Отладчик VIP"

4.3.2. Макропеременные в параметрах проекта

Для универсальности настройки проекта в параметрах компиляции рекомендуется использовать переменные. Макросы позволяют описать шаблоны значений отдельных полей. Расшифровка макросов доступна во всплывающей подсказке.

В настройках проекта по нажатию кнопки  открывается стандартный набор следующих переменных:

- *[\$[ProjectName]* — имя проекта;
- *[\$[ProjectFile]* — имя файла проекта;
- *[\$[ProjectPath]* — путь проекта;
- *[\$[FileName]* — имя текущего файла;
- *[\$[FilePath]* — путь текущего файла;
- *[\$[AtlPath]* — путь **Каталога компилятора** (параметр элемента сборки на вкладке Компилятор VIP);

- $\$[ExtRes]$ — расширение ресурса (.res).

Существует возможность использования собственных макропеременных. Для их создания предназначена вкладка [Переменные](#)⁸⁸.

Рекомендуемый список настроек компилятора проекта с использованием макропеременных:

- Вкладка Компилятор VIP параметр **Имя текущего компонента** — $\$[FileName]$.
- Вкладка Компилятор VIP > Ресурсы параметр **Имя ресурсного файла** — <адрес каталога ресурсного файла> $\backslash \$[FileName] \$[ExtRes]$.
- Вкладка Компилятор VIP > FR формы параметр **Каталог для поиска форм** — $\$[FilePath]$ <имя каталога форм отчетов>.

5. Среда Viper

5.1. Исполняющая среда Viper

Среда разработки **Viper** поставляется в составе средства разработки **Атлантис**. Файл **Viper.exe** находится в папке **EXE** средства **Атлантис**.

Существует возможность одновременного использования нескольких копий **Viper**. Ею управляет параметр *Запускать только один экземпляр программы* (см. "[Вкладка Об-
щие](#)¹⁰⁵").

5.1.1. Справочная система

Организация справочной системы в среде **Viper** предоставляет пользователю возможность получения информации в интерактивном режиме. Среда разработки **Viper** содержит справки двух видов:

- вызов контекстной справки по **Атлантис** (операторы языка **VIP**, описание методов и т. д.) осуществляется по нажатию клавиши **F1** в редакторе кода. Данная справка содержится в файле **AtIDoc.chm**;
- справка о среде разработки **Viper**. Она содержится в файле **VIPER_NNNN.chm** (где **NNNN** — версия).

Файлы **AtIDoc.chm** и **VIPER_NNNN.chm** входят в состав документации на **Атлантис**. Данная документация поставляется в составе дистрибутива и обычно помещается в папку **AtIDoc**, рядом с папкой **EXE** средства разработки **Атлантис** (или внутри папки **EXE**).

При вызове контекстной справки **AtIDoc.chm** в окне просмотра активируется вкладка Указатель, при этом текущее слово (или выделенный текст) в редакторе **Viper** автоматически добавляется в буфер обмена и отображается в окне справки в строке ключевого слова для поиска.

Файл **AtIDoc.chm** содержит всю документацию по **Атлантис**, поэтому использованный алгоритм обеспечивает доступ ко всей имеющейся информации по данному слову. Для поиска всех вхождений данного слова (например, всех примеров программ, где был использован данный оператор) необходимо добавить его в поле ввода вкладки Поиск и нажать кнопку Разделы. Для просмотра найденного соответствия необходимо выбрать раздел из списка результата поиска. Возможность переназначить файл контекстной справки доступна в главном меню *Вид > Параметры* (см. "[Вкладка Atlantis VIP — Справка](#)¹²⁵").

Вызов справки "Среда разработки Viper" осуществляется по нажатию кнопки Справка в диалоговых окнах команд **Viper**, где она предусмотрена. Например, в окне *=Параметры=* (см. "[Настройка параметров среды](#)¹⁰⁵").

Также доступ к справочной документации возможен при помощи главного меню *Справка*:

- *Содержание* — открытие файла **VIPER_NNNN.chm**;
- *Документация на Атлантис* — открытие файла **AtIDoc.chm**.

5.1.2. Диагностика исключительных ситуаций

Функция диагностики исключительных ситуаций позволяет выявить причину их возникновения для последующего устранения.

В **Viper** существуют следующие режимы диагностики:


- 1) Автоматическое формирование диалогового окна с протоколом ошибки.

При возникновении ошибки среды, отображается окно **=Viper — Ошибка приложения=** с текстом сообщения. В строке **Протокол ошибки** отображается полное имя файла, содержащего детальный отчет об ошибке.

Как правило файл **ViperError.log** формируется в папке **EXE** приложения.

Для продолжения работы в **Viper** предоставляются следующие кнопки:

- **[Подробности]** — отобразить детальный отчет об ошибке. В нем содержится информация об адресе исключения, состоянии стека, список загруженных модулей и др., а также доступна возможность копировать текст отчета в буфер обмена.
- **[Отправить протокол в службу поддержки]** — формирование электронного письма в адрес поддержки среды (viper@galaktika.by), к которому автоматически прикрепляется протокол об ошибке. Данное письмо содержит несколько пунктов, которые необходимо максимально подробно заполнить для предоставления более детальной информации о возникшей проблеме. Для формирования письма используется почтовая программа, установленная по умолчанию в операционной системе.
- **[Завершить процесс]** — выполнить завершение работы приложения. При этом среда **Viper** закрывается аварийно, из-за чего все несохраненные изменения в редакторе будут утеряны.
- **[Продолжить]** — закрыть окно об ошибке и продолжить работу с приложением. Данная возможность выполнима при возникновении некритичных ошибок и предназначена, к примеру, для того, чтобы сохранить измененные в редакторе файлы.

 Функция диагностики не гарантирует 100% обработку всех возможных исключительных ситуаций. А также возможны случаи формирования стандартного окна об ошибке для **Атлантис**-приложений. При этом формируется файл **AtlError.log**.

- 2) Формирование детального диагностического протокола при работе в **Viper**.

Использование данного режима доступно на любом этапе работы в среде. При этом в каталоге приложения формируется файл **ViperLog.sil**.

Для активации диагностического протокола существует несколько способов:

- Через параметры командной строки. Ключ **-logtofile** включает режим записи диагностических данных в протокол.
- Функцией **Диагностика** в системном меню приложения **Viper** или с помощью комбинации клавиш **Alt+Shift+F4**.
- При использовании программного обеспечения SmartInspect Resistriutable Console (<http://www.gurock.com/smartinspect/resources/>). Ключ **-logtopipe** включает режим записи диагностических данных непосредственно в консоль SmartInspect по протоколу **pipe**.

Приложение SmartInspect Resistriutable Console также используется для просмотра файлов ***.sil**.

После активации протокола следует выполнить действия, приводящие к ошибке приложения **Viper**.

Завершить формирование протокола можно закрытием среды разработки или командой Диагностика **Alt+Shift+F4**.

Сформированный файл протокола следует отправить в службу поддержки **Viper** (viper@galaktika.by).

5.2. Настройка параметров среды

Параметры среды **Viper** позволяют управлять поведением функций приложения и оформлением рабочей области на локальном компьютере.

Для доступа к настройкам среды предназначен пункт главного меню *Вид > Параметры*.

Окно *Параметры* содержит следующие группы настроек:

- *Общие*;
- *Редактор*;
- *Синтаксис*.

Раздел *Общие* позволяет определить параметры загрузки среды **Viper**, настройки *Окна вывода* и панели вкладок, ассоциации расширений файлов, параметры сохранения файла и информации в *Окне вывода* при запуске компиляции, настройки скриптов и шаблонов файлов.

Раздел *Редактор* влияет на параметры редактора кода и дополнительные функции среды, в том числе позволяет использовать функцию подсказки кода, автозамены, проверки орфографии, сворачивания кода и др.

Раздел *Синтаксис* предоставляет настройки раскраски и стиля синтаксической схемы для конкретных типов файлов, а также возможность использования функции автозаполнения и автозамены.

 Данные о параметрах среды **Viper** хранятся в файле **Default.vpr**, который расположен по адресу *<Системный диск>\<Имя пользователя>\AppData\Roaming\Viper*.

[Продолжить] — сохранить измененные настройки и закрыть окно параметров, без использования диалога параметров среды измененные настройки сохраняются при закрытии проекта или приложения **Viper**.

[Отменить] — закрыть окно параметров без сохранения изменений.

[Справка] — открыть справочную информацию о среде разработки **Viper**.

5.2.1. Вкладка Общие

Вкладка позволяет управлять параметрами основных команд среды. В том числе команды по созданию файлов, открытию, сохранению и формату страниц.

Список последних открытых файлов — настройки списка:

- *Максимальное количество файлов* — максимальная длина списка *Последние файлы* (см. "[Меню Файл](#)"²⁴).
- *Максимальное количество проектов* — максимальная длина списка *Последние проекты* (см. "[Меню Проект](#)"⁵⁰).
- *Показывать имена файлов с путем* — полные имена файлов в списках последних файлов/проектов.

Отслеживать внешние изменения — автоматическая проверка открытых в редакторе файлов на изменения, сделанные внешними программами.

При этом выполняется одна из команд:

- **Только предупреждать** — вывод сообщения об измененном файле с указанием его полного имени.
- **Предлагать обновить** — вывод предложения о повторной загрузке измененного файла с указанием его полного имени.
- **Обновлять без предупреждения** — повторная загрузка измененного файла. При использовании данной функции все его несохраненные изменения удалятся.

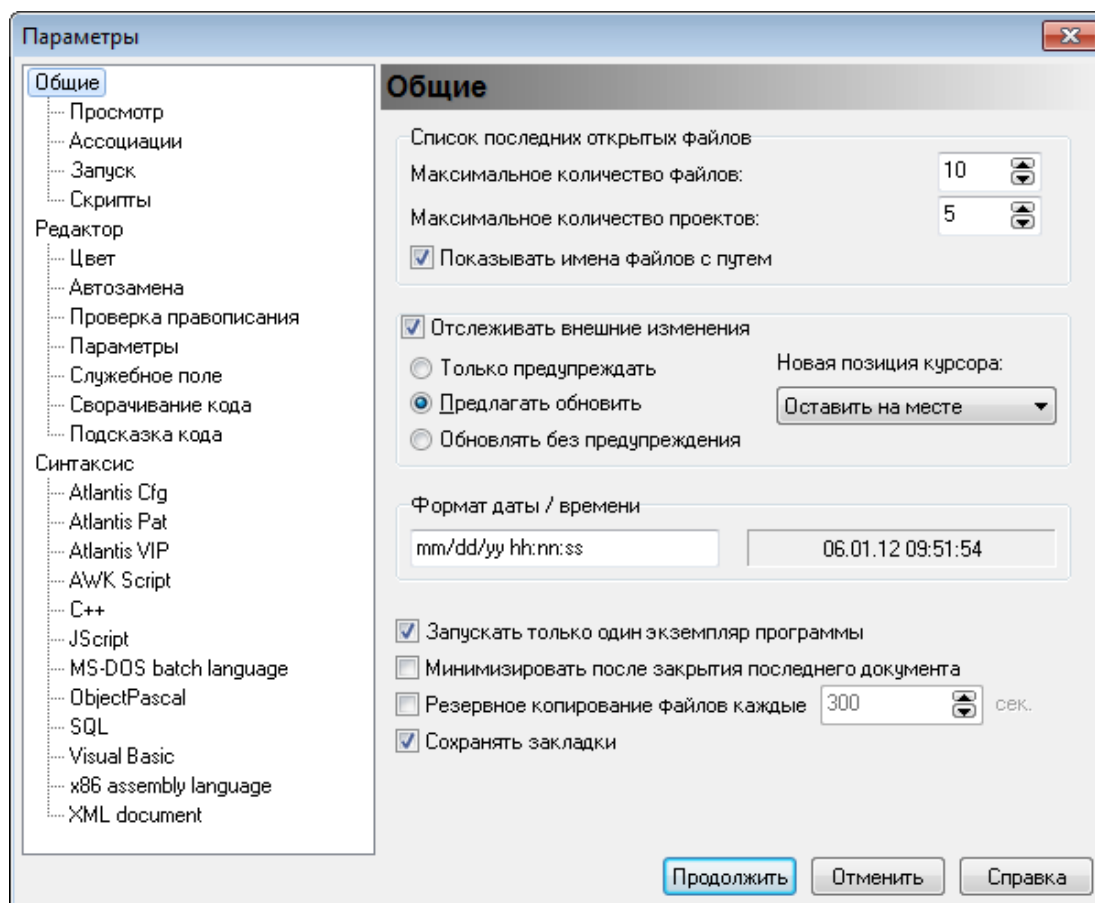


Рис. 82. Страница настроек "Общие"

Новая позиция курсора — перемещение курсора в редакторе кода после повторной загрузки файла. При помощи стрелки, направленной вниз, осуществляется выбор позиции: *Начало файла*, *Конец файла*, *Файл закладок*, *Оставить на месте*, *Предположить*.


Формат даты/времени — формат даты и времени, используемые командой *Вставить Дата/Время* (см. "[Меню Правка](#)"²⁷). Управляющие символы:

- *y* — год;
- *m* — месяц;
- *d* — день;
- *h* — часы;
- *n* — минуты;
- *s* — секунды.

Запускать только один экземпляр программы — запрет на запуск второго экземпляра среды **Viper**. Ассоциированные файлы будут открываться в последний запущенный экземпляр **Viper**.

Минимизировать после закрытия последнего документа — сворачивание приложения **Viper** в панель задач при закрытии последнего редактора кода.

Резервное копирование файлов каждые — создание резервной копии файла, содержащего несохраненные изменения, с указанной периодичностью времени (сек.). Копия файла сохраняется с расширением **'bak'** в скрытом каталоге **'backup'**, который создается в папке исходного документа или среды **Viper**. При корректном закрытии файла его резервная копия удаляется.

 При отсутствии доступа к каталогу измененного файла, его копия сохраняется в каталоге активного приложения **Viper.exe**.

Сохранять закладки — сохранение установленных закладок (см. "[Меню Поиск](#)"³⁶) в файлах проекта.

5.2.1.1. Вкладка Общие - Просмотр

Вкладка содержит параметры загрузки среды **Viper**, а также оформления окон.

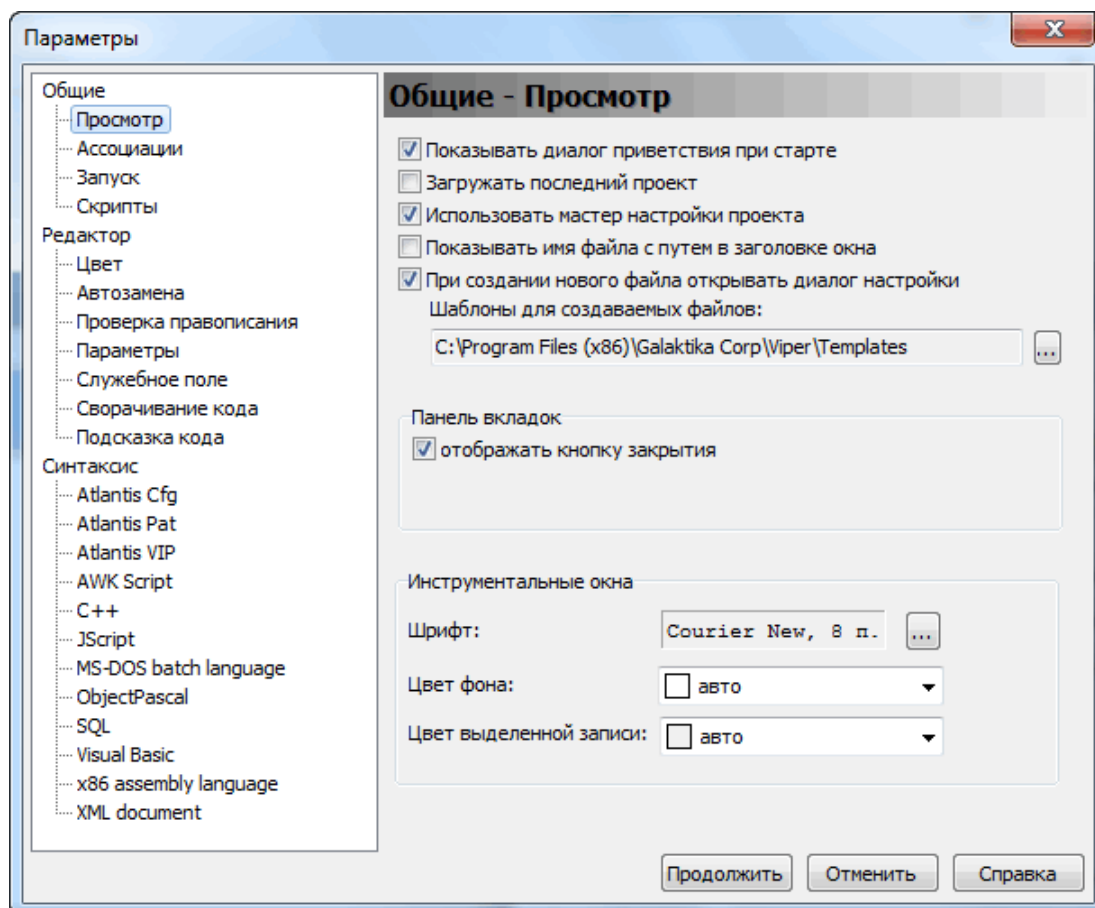


Рис. 83. Страница настроек "Общие-Просмотр"


Показывать диалог приветствия при старте — отображение диалогового окна =Приветствие= (см. "[Интерфейс пользователя](#)"²²) при открытии среды **Viper**.

Загружать последний проект — загрузка проекта из предыдущей сессии при открытии среды **Viper**.

Использовать мастер настройки проекта — отображение диалогового окна =Мастер настройки проекта= при выполнении команды [создания проекта](#)⁸¹.

Показывать имя файла с путем в заголовке окна — отображение в заголовке приложения **Viper** пути активного файла в редакторе. По умолчанию отображается только имя файла.

При создании нового файла открывать диалог настройки — отображение диалогового окна при выполнении команды **Создать** (см. "[Меню Файл](#)²⁴").

Шаблоны для создаваемых файлов — адрес каталога с файлами шаблонов. Параметр используется вместе с предыдущей настройкой. Для добавления адреса предназначена кнопка , которая осуществляет открытие диалогового окна.

Панель вкладок — атрибуты для вкладок редактора кода:

- **Отображать кнопку закрытия** — возможность закрывать вкладку редактора по нажатию на "крестик" в строке заголовка.

Инструментальные окна — единые настройки стиля для инструментальных окон среды **Viper**:

Шрифт — стиль текста в окне (**Шрифт**, **Начертание**, **Размер**, **Видоизменение**, **Цвет**, **Набор символов**).

Цвет фона — цвет, являющийся фоном для содержимого в области окна.

Цвет выделенной записи — цвет, являющийся фоном для выбранной записи в области окна.

5.2.1.2. Вкладка **Общие - Ассоциации**

Вкладка позволяет связывать определенные типы файлов со средой **Viper**, а также назначать для них иконки.

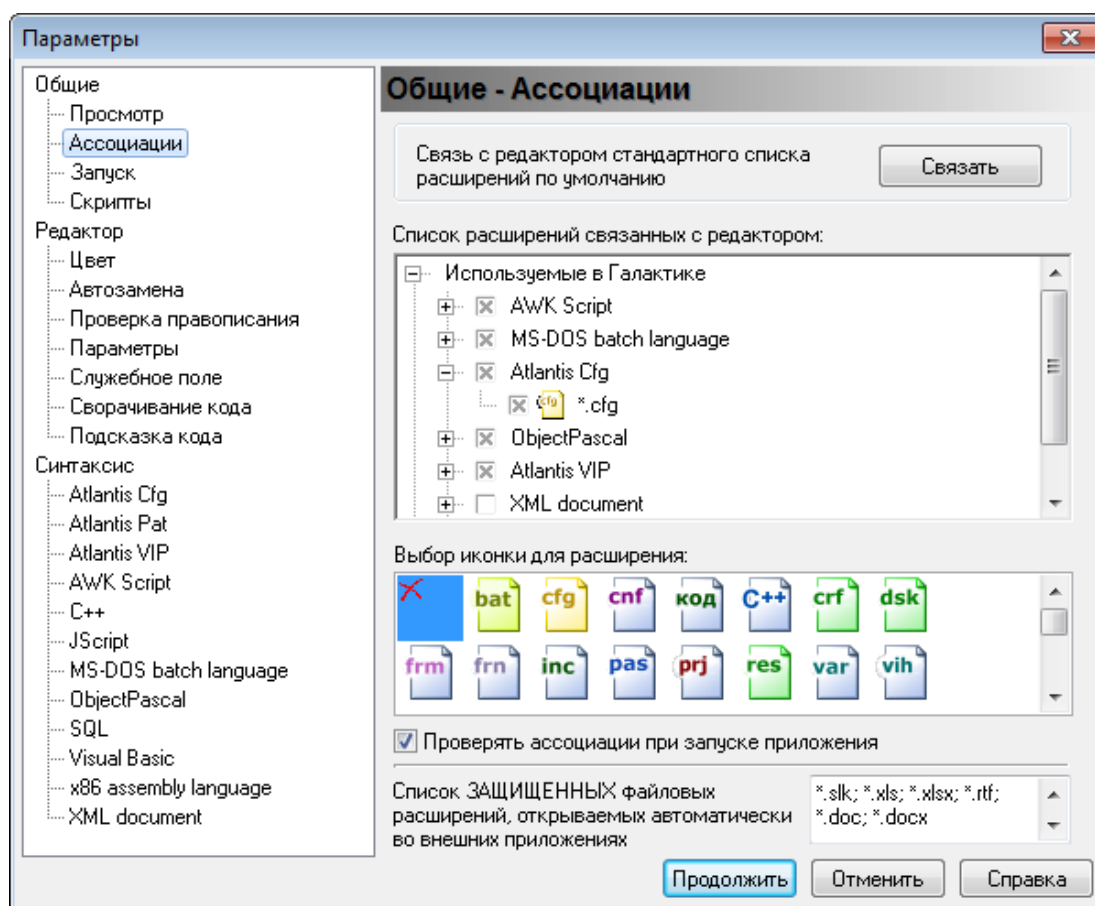


Рис. 84. Страница настроек "Общие-Ассоциации"

Список расширений связанных с редактором — список типов файлов. Структура списка аналогична синтаксическим схемам с соответствующими расширениями файлов. Для назначения ассоциации следует пометить галочкой необходимый тип файла.

Состав расширений определяется содержимым параметра **Типы файлов** в настройках синтаксиса (см. "[Вкладка Atlantis VIP — Подсветка](#)¹²³").

Выбор иконки для расширения — список иконок. Выбор иконки для указанного расширения осуществляется двойным нажатием. Изображение, назначенное основному узлу, применяется к его вложенным типам файлов.

Список иконок хранится в каталоге <Системный диск>\Documents and Settings\<Имя пользователя>\Application Data\Viper\Icon и первоначально формируется путем загрузки из Viper.exe.

Проверять ассоциации при запуске приложения — проверка актуальности установленных ассоциаций при запуске **Viper**. При отсутствии связи с расширением, которое было ассоциировано в предыдущей сессии **Viper**, отображается запрос о восстановлении связи с ним.

Дополнительно реализован **Список ЗАЩИЩЕННЫХ файловых расширений, открываемых автоматически во внешних приложениях** операционной системы. По умолчанию список содержит: *.slk; *.xls; *.xlsx; *.rtf; *.doc; *.docx. При добавлении расширения в список, в начале следует указывать символы [*].

[Связать] — назначить связь для стандартных типов файлов со средой **Viper**. В том числе типы файлов синтаксической схемы AtlantisVIP, а также *.pas, *.cfg и *.bat.



Параметры вкладки *Ассоциации* доступны для изменения в среде разработки *Viper*, запущенной с правами администратора.

5.2.1.3. Вкладка *Общие - Запуск*

Вкладка управляет изменениями в файле и сообщениями *=Окна вывода=* при выполнении компиляции.

Сохранение перед началом компиляции — режим сохранения измененных файлов перед запуском компиляции файла: *Не сохранять*, *Сохранять с запросом*, *Сохранять без запроса*. При включенном параметре *Сохранять с запросом* пользователю предоставляется возможность выбора варианта сохранения в диалоговом окне *=Сохранить изменения=* (см. "[Меню Файл](#)"²⁶).

Сохранение перед выполнением профиля — режим сохранения измененных файлов перед выполнением профиля. В данном параметре содержатся режимы, аналогичные описанным в параметре *Сохранение перед началом компиляции*.

Окно вывода — параметры отображения информации в *=Окне вывода=*:

- **Очищать перед выполнением компиляции** — удаление истории предыдущей компиляции;
- **Выводить командную строку и другую информацию** (статус и время процесса, адрес каталога при компиляции профиля);
- **Прокручивать список автоматически**.

Класс приоритета — приоритет выполняемой программы:

- **Ожидание** — выполнение при простое системы;
- **Стандартный** — режим без специальных требований;
- **Высокий** — критичные ко времени задания, которые для корректной работы должны выполняться немедленно;
- **Реальное время** — задания с наивысшим приоритетом. Такие потоки перекрывают выполнение всех остальных процессов, включая другие важные процессы операционной системы.



Следует крайне осторожно использовать приоритеты *Высокий* и *Реальное время*, т. к. их задачи могут занять все время центрального процессора и перекрыть другие задания, включая системные.

Время простоя для консоли — временное ограничение (секунды) на ожидание выполнения, после которого выдается сообщение с предложением завершить выполнение запущенного задания.

Сохранить результат выполнения в файл — сохранение сообщений перехватываемой консоли в файле протокола, который автоматически создается в каталоге проекта. Имя файла состоит из имени проекта с расширением *log*.

Для обеспечения уникальности к имени добавляется дата и время.

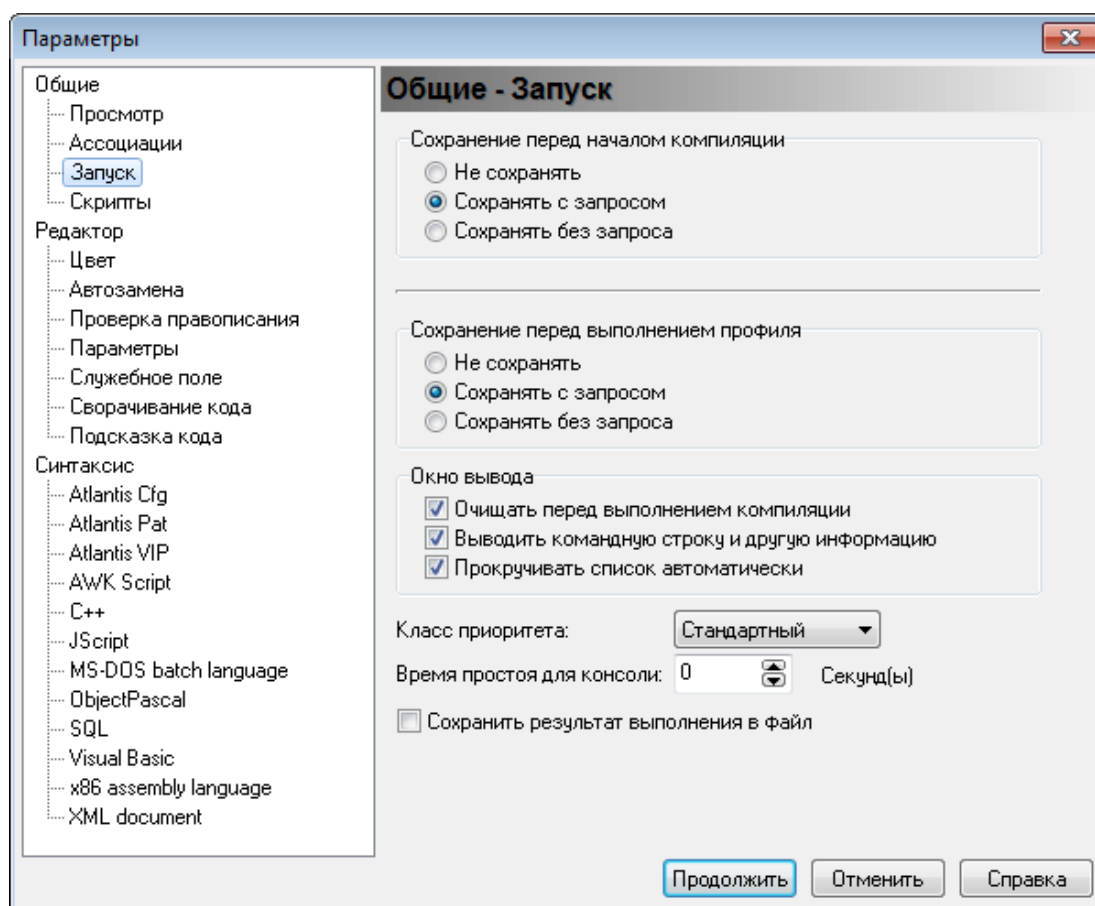


Рис. 85. Страница настроек "Общие-Запуск"

5.2.1.4. Вкладка Общие - Скрипты

Вкладка позволяет добавлять скрипты на панель главного меню для их последующего использования, а также назначать их выполнение на определенные события среды.

Для добавления скрипта предназначены следующие вкладки: *Список*, *На событие*.

- *Список* — выполнение скрипта по запросу пользователя. Список скриптов отображается в главном меню [Скрипты](#)^[53], на их вызов можно назначить "горячую" клавишу (см. "[Настройка инструментальных панелей и горячих клавиш](#)"^[48]).
- *На событие* — выполнение скрипта при возникновении определенного события:
 - начало или конец программы;
 - создание файла;
 - активирование или деактивирование файла;
 - открытие или закрытие файла;
 - сохранение файла или сохранение файла как;
 - изменение файла — событие активируется с секундной задержкой после последнего действия пользователя, связанного с изменением текста в редакторе. На данное событие рекомендуется назначать скрипты, предназначенные для модификации введенного текста (например, автозаполнение или автоформатирование);
 - вызов подсказки кода — событие активируется перед вызовом функции *Подсказка кода* (см. "[Сервис](#)"^[73]). На данное событие рекомендуется назначать

скрипты наполняющие список подсказки пользовательскими конструкциями (см. "[Примеры использования скриптов](#)"¹⁶⁵");

- открытие или закрытие проекта.

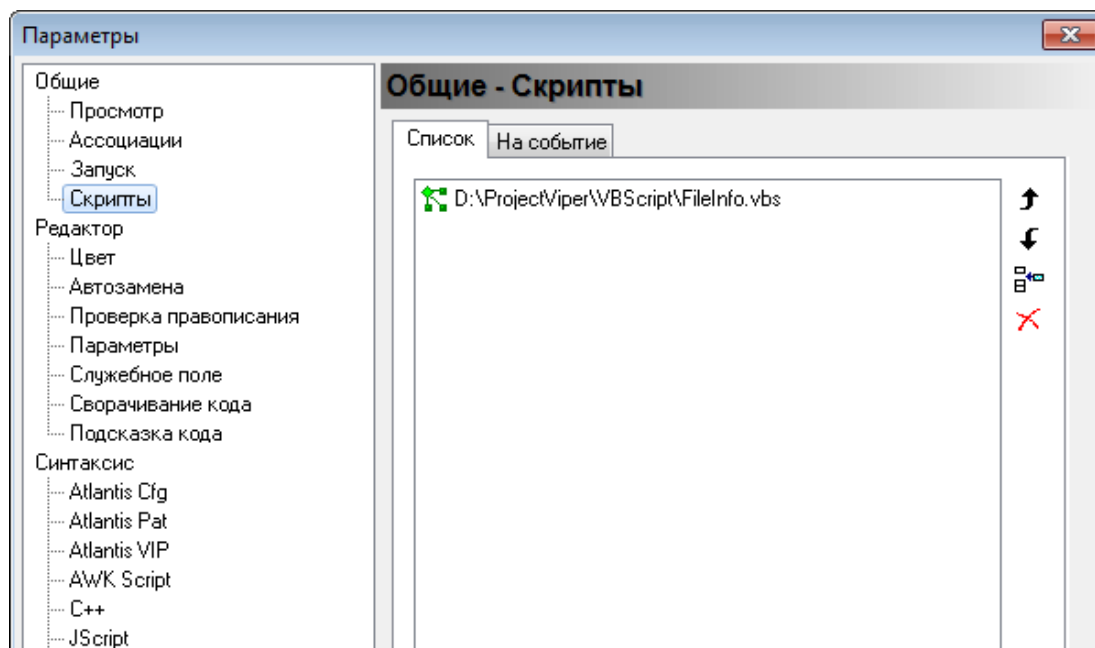






Рис. 86. Страница настроек "Общие-Скрипты"

Команды управления списком скриптов:

-  **Ctrl+Up** — перемещение текущей записи вверх на одну позицию;
-  **Ctrl+Down** — перемещение текущей записи вниз на одну позицию;
-  **Ins** — добавление файла с текстом скрипта. При вызове команды осуществляется открытие диалогового окна для выбора файла;
-  **Del** — удаление текущей записи из списка.

Выбор нескольких элементов списка осуществляется при удержании клавиши **Ctrl**.

5.2.2. Вкладка Редактор

Вкладка содержит параметры для настройки редактора кода.

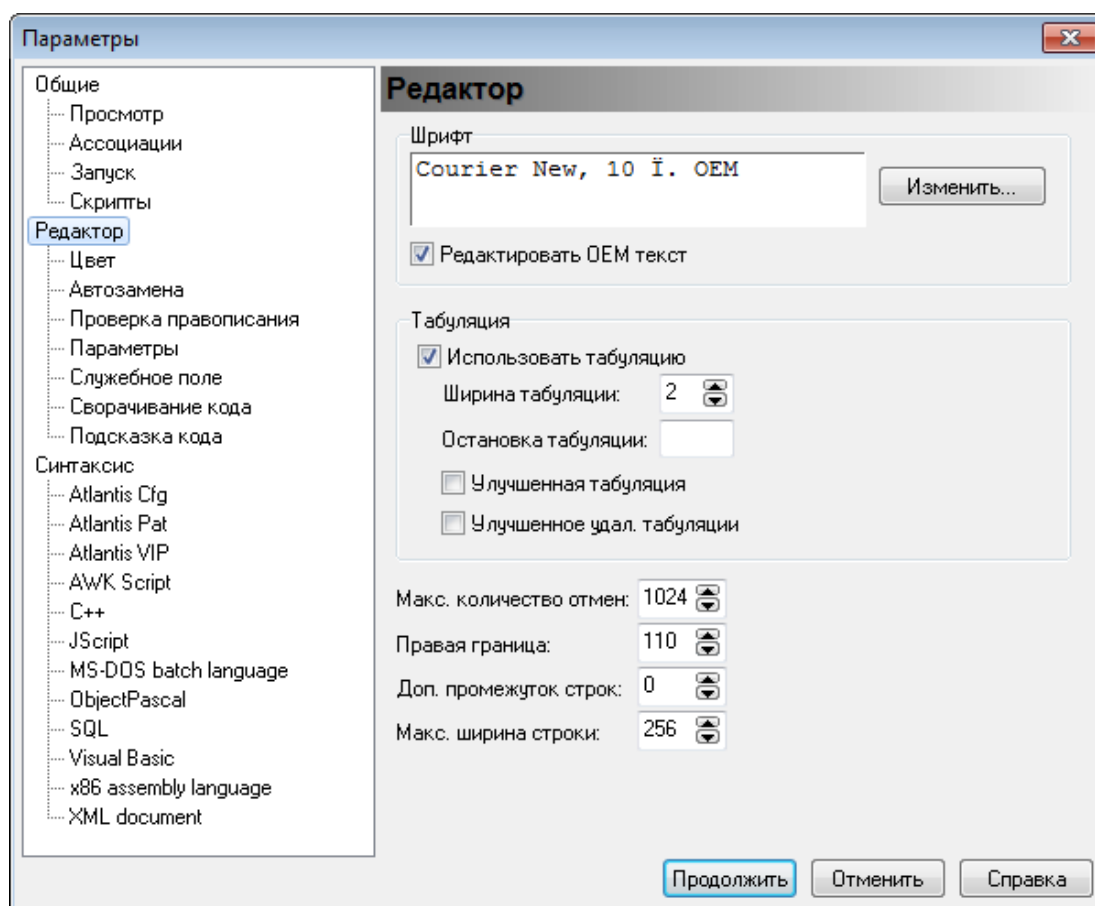


Рис. 87. Страница настроек "Редактор"

Шрифт — параметры шрифта:

- **[Изменить]** — изменение формата шрифта. Вызов команды осуществляет открытие диалогового окна с параметрами: шрифт, начертание, размер, цвет и т. д. Атрибуты — цвет, начертание и видоизменение шрифта применяются, когда параметр подсветки синтаксиса в среде не используется (см. "[Вкладка Синтаксис](#)"¹²¹).
- **Редактировать OEM текст** — перевод текста в OEM-кодировку при открытии файла.

Табуляция — параметры использования символа табуляции:

- **Использовать табуляцию** — возможность использования клавиши **Tab** для добавления символа табуляции или отступа, в зависимости от ситуации:
 - символ табуляции добавляется в позицию курсора или вместо выделенных символов одной строки;
 - отступ добавляется (по **Shift+Tab** — удаляется) в начало нескольких выделенных строк/вертикального блока. Размер отступа равен значению параметра **Ширина табуляции**.
- **Ширина табуляции** — размер знака табуляции и отступа (функция **Добавить отступ** в разделе "[Меню Блок](#)"³²). Измеряется количеством пробелов;
- **Остановка табуляции** — позиции табуляторов (например, 3,8,10);
- **Улучшенная табуляция** — выравнивание табуляции по вертикали;
- **Улучшенное удаление табуляции** — выравнивание удаляемой табуляции по вертикали.

Параметры ограничений:

- **Максимальное количество отмен** — количество выполненных действий по редактированию, которые можно отменить (команда *Отменить* меню *Правка*).
- **Правая граница** — рекомендуемая длина строки в символах. По умолчанию 110 символов. Граница обозначается серой вертикальной линией в редакторе кода, но не блокирует ввод текста за ее пределами.
- **Дополнительный промежуток строк** — межстрочный интервал.
- **Максимальная ширина строки** — размер строки (количество символов) в пределах видимой области редактора, определяющий появление скроллинга, когда данное условие не выполняется.

5.2.2.1. Вкладка Редактор - Цвет

Вкладка содержит параметры цветового оформления редактора кода, а также подсветки парных элементов и выделенных фрагментов кода.

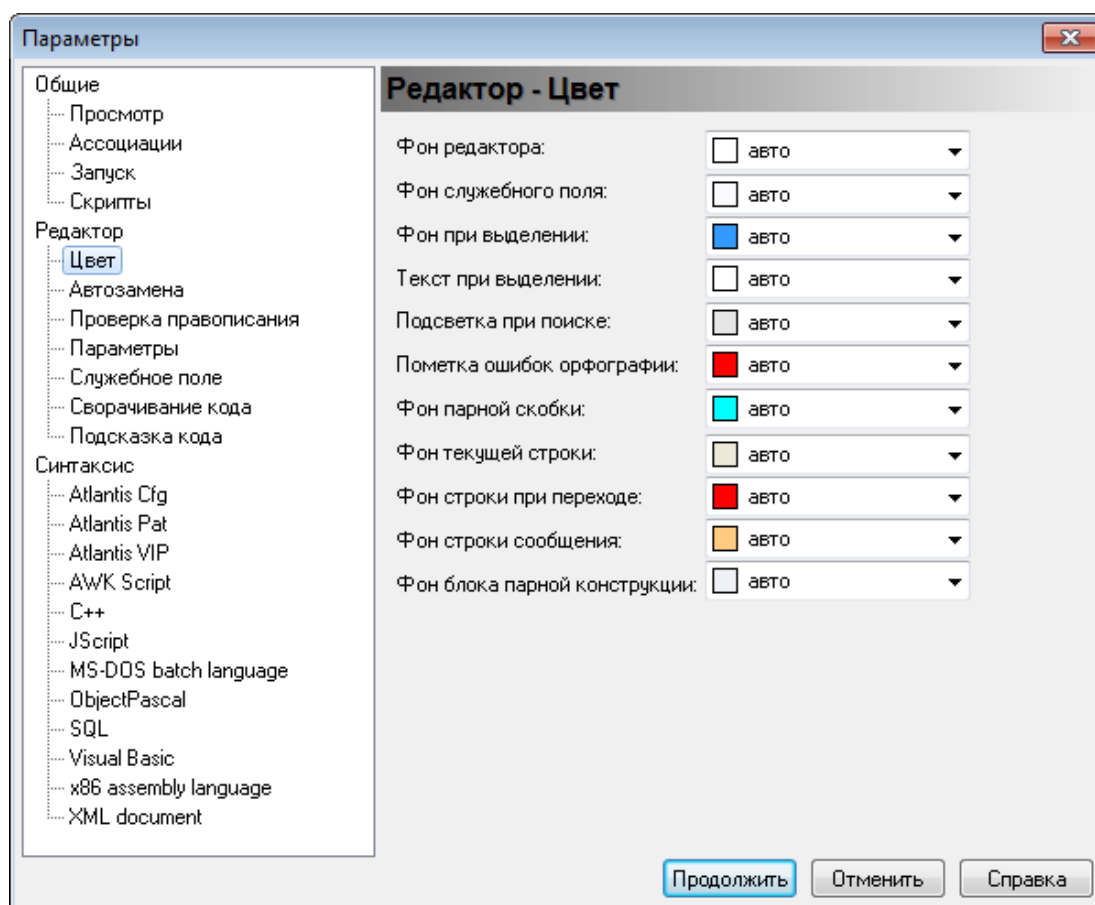


Рис. 88. Страница настроек "Редактор-Цвет"

Изменения цвета осуществляется с помощью выпадающего списка по нажатию стрелки, направленной вниз, в следующих параметрах:

- **Фон редактора** — фон области редактора кода;
- **Фон служебного поля** редактора кода;
- **Фон при выделении** — фон выделенной области в редакторе кода;
- **Текст при выделении** — цвет текста в выделенной области;
- **Подсветка при поиске** — фон найденных соответствий в редакторе кода по результатам выполнения функции поиска;

- **Пометка ошибок орфографии** — цвет подчеркивания при использовании параметра *Подчеркивать слова с ошибками* (см. "[Вкладка Редактор — Проверка правописания](#)¹¹⁵");
- **Фон парной скобки** при использовании функции *Подсветка скобок* (см. "[Меню Вид](#)⁴⁶");
- **Фон текущей строки** при использовании параметра *Подсвечивать текущую строку* (см. "[Вкладка Редактор — Параметры](#)¹¹⁶");
- **Фон строки при переходе** — фон строки редактора кода при переходе к ней по ссылке из *=Окна вывода=*;
- **Фон строки сообщения** — фон строки редактора кода при переходе к ней по ссылке из окна *=Сообщения=*;
- **Фон блока парной конструкции** — фон блока текущей конструкции при выполнении функции *Парная конструкция* (см. "[Меню Навигация по коду](#)⁴³").

5.2.2.2. Вкладка Редактор - Автозамена

Вкладка позволяет использовать функцию автозамены. Ее параметры для каждой синтаксической схемы настраиваются индивидуально (см. "[Вкладка Atlantis VIP — Автозамена](#)¹²⁵").

Возможность и параметры использования автозамены:

- **Включено** — использование функции автоматической замены текста из шаблона;
- **Не учитывать регистр** — выполнение замены без учета регистра текста;
- **Сохранять регистр** — выполнение замены с применением регистра заменяемого текста.

Функция автозамены выполняется при добавлении следующего символа или строки.

5.2.2.3. Вкладка Редактор - Проверка правописания

Вкладка содержит параметры проверки орфографии в редакторе кода.

Возможность и параметры проверки орфографии:

- **Подчеркивать слова с ошибками** — проверка грамматических ошибок в русскоязычном тексте, при этом слова с ошибками выделяются подчеркиванием. Возможность изменения цвета **Пометки ошибок орфографии** доступна в настройках среды (см. "[Вкладка Редактор — Цвет](#)¹¹⁴").
- **Пропускать одиночные символы** — игнорирование слов, состоящих из одного символа.
- **Пропускать слова с цифрами** — игнорирование слов, содержащих цифры.
- **Проверять области** — основные синтаксические конструкции языка **VIP**, подлежащие проверке орфографии.

Проверяемые конструкции следует отметить в представленном списке. По умолчанию проверка назначена для области комментариев (*Comment*), строковых констант (*String*), описания экранов (*Screen*), текста документации (*Text*).

Для исправления ошибок предназначен пункт меню *Орфография* (см. "[Меню Правка](#)²⁷").

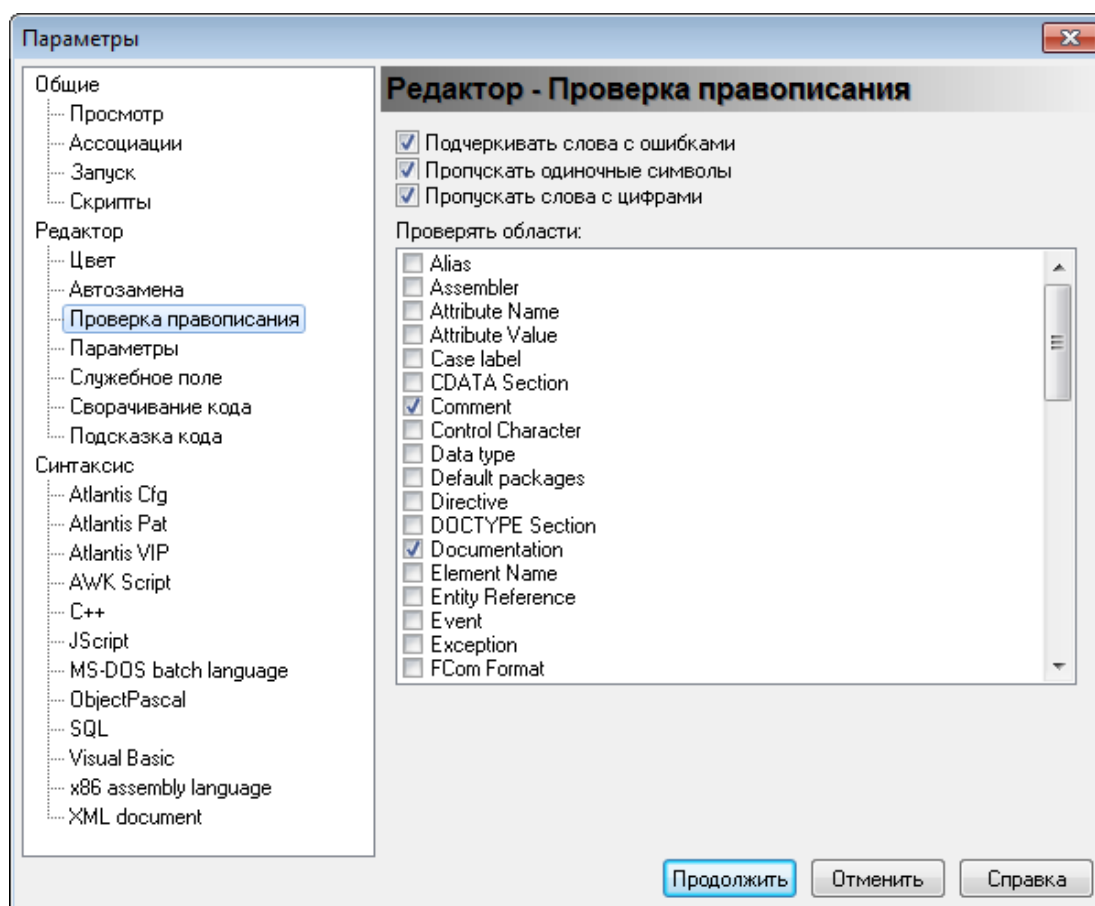


Рис. 89. Страница настроек "Редактор-Проверка правописания"

5.2.2.4. Вкладка Редактор - Параметры

Вкладка предоставляет набор параметров для использования дополнительных возможностей среды **Viper**.

Вставка из буфера при нажатии средней кнопки мыши, в позицию указателя.

Очищать очередь отмены после сохранения — очистка буфера обмена, содержащего данные для отмены выполненных команд редактирования, при сохранении изменений.

Группировать параметры отмены — выполнение отмены одной командой для ряда последовательно введенных изменений.

Drag and Drop текста — перемещение выделенного текста в необходимую позицию с помощью указателя.

Улучшить клавишу Home — перемещение курсора в позицию первого символа (не пробела и табуляции) строки относительно отступа. При этом повторное нажатие клавиши **Home** перемещает курсор в первую позицию строки.

Прокрутка по полстраницы — прокрутка активного редактора и перемещение курсора на полстраницы в заданном направлении с помощью клавиш **Page Up/Page Down**.

Прокрутка за предел файла — прокрутка строк в редакторе за пределы окончания файла.

Прокрутка за конец строки — сдвиг строки за пределы ее границы с возможностью установки курсора в любой позиции.

Подсказка прокрутки — отображение всплывающей подсказки с номером первой строки на странице при пролистывании с помощью полосы прокрутки.

Заменять табуляцию пробелами — замена вводимых символов табуляции пробелом. Данная настройка применяется только с параметром **Использовать табуляцию** в соответствии с ее настройками (см. "[Вкладка Редактор](#)¹¹²").

Удаление конечных пробелов — удаление пробелов в конце строк.

Автоотступ — режим автоматического выравнивания строк с учетом отступа предыдущей строки. При этом начало новой строки автоматически сдвигается вправо на ширину отступа предыдущей.

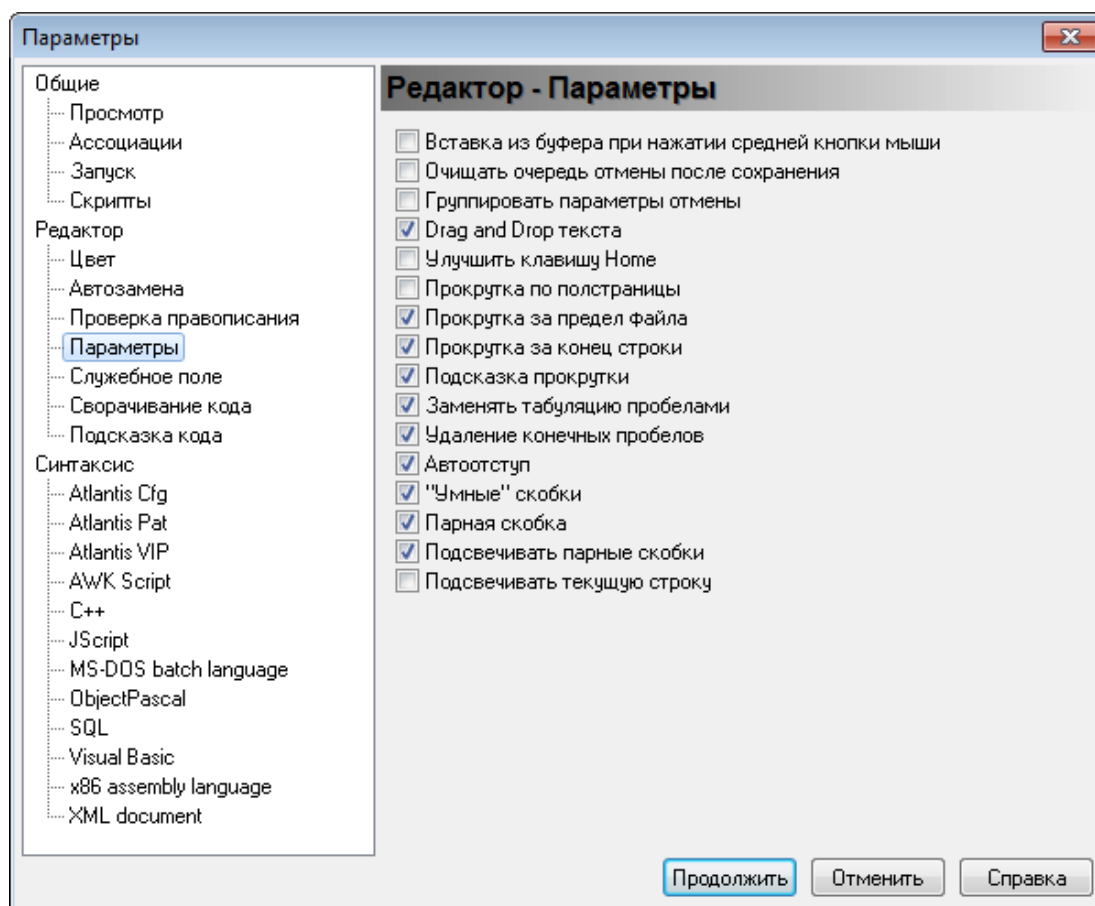


Рис. 90. Страница настроек "Редактор-Параметры"

"Умные" скобки — выравнивание скобок. Применяется автоматически в следующих случаях:

- код, расположенный после открывающей скобки, переносится на новую строку с дополнительным отступом вправо относительно первого символа предыдущей строки. Отступ соответствует **Ширине табуляции** (см. "[Вкладка Редактор](#)¹¹²");
- закрывающая скобка переносится на новую строку с вертикальным выравниванием относительно первого символа предыдущей строки.

Парная скобка — автоматическое добавление парной закрывающей скобки при вводе открывающей. Парные скобки добавляются в конце строки или перед пробелом. Удалить обе скобки можно клавишей **Backspace**, при этом курсор должен находиться между ними.

Подсвечивать парные скобки — выделение цветом парных скобок, на одной из которых установлен курсор.

Возможность изменения **Фона парной скобки** доступна в разделе "[Вкладка Редактор — Цвет](#)¹¹⁴".

Подсвечивать текущую строку — выделение текущей строки цветом.

Возможность изменения **Фона текущей строки** доступна в разделе "[Вкладка Редактор — Цвет](#)¹¹⁴".

5.2.2.5. Вкладка Редактор - Служебное поле

Вкладка предоставляет параметры служебного поля, расположенного в левой части редактора кода.

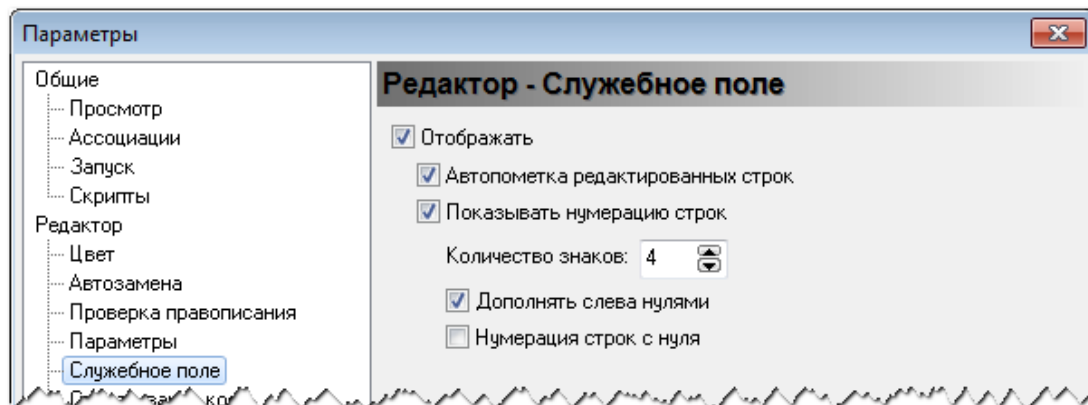


Рис. 91. Страница настроек "Редактор-Служебное поле"

Отображать — показывать служебное поле у левой границы редактора кода.

Существует возможность изменения **Фона служебного поля** (см. "[Вкладка Редактор — Цвет](#)¹¹⁴").

- **Автопометка редактированных строк** — вертикальная линия в служебном поле для обозначения измененных строк. Когда изменения не сохранены — линия оранжевого цвета, после сохранения — зеленого.
- **Показывать нумерацию строк** — отображение в служебном поле номера строки. Номер каждой пятой строки отображается по умолчанию, другие отображаются при позиционировании курсора.

Для использования нумерации доступны следующие настройки:

- **Количество знаков в номере строки;**
- **Дополнять слева нулями** — отображение лидирующих нулей в номере строки;
- **Нумерация строк с нуля** — цифра начала отсчета 0.

5.2.2.6. Вкладка Редактор - Сворачивание кода

Вкладка управляет использованием функции сворачивания кода и ее параметрами.

Использовать сворачивание кода — использование функций подпунктов **Сворачивание** и **Разворачивание кода** (см. "[Меню Правка](#)²⁷").

Сворачивать структуры — конструкции, подлежащие сворачиванию.

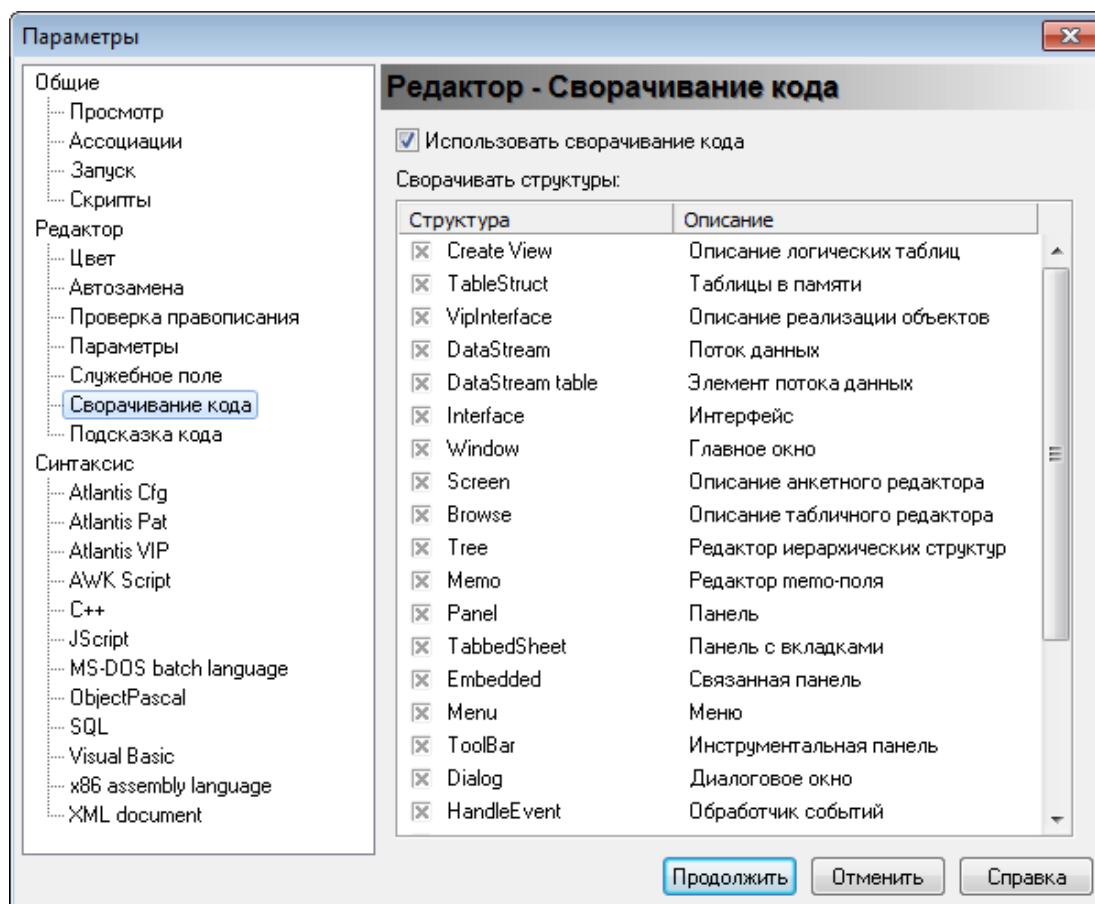


Рис. 92. Страница настроек "Редактор-Сворачивание кода"

Перечень конструкций:

- Create view;
- TableStruct;
- VipInterface;
- DataStream;
- DataStream table;
- Interface;
- Window;
- Screen;
- Browse;
- Tree;
- Memo;
- Panel;
- TabbedSheet;
- Embedded;
- Menu;
- ToolBar;
- Dialog;

- HandleEvent;
- TableEvent;
- WindowEvent;
- CmEvent;
- Procedure/Function;
- Var;
- #Region..#EndRegion;
- Parameters;
- Case;
- For;
- If;
- While;
- Do..while.

5.2.2.7. Вкладка Редактор - Подсказка кода

Вкладка содержит настройки списка [подсказки кода](#)⁷⁴, в том числе возможность выбора источников для наполнения списка.

Использовать подсказчик кода — возможность выполнения функций [подсказки кода](#)⁷⁴, а также просмотра структуры подключаемых файлов в окне *=Дерево подключений=* и связей в окне *=Связи таблиц=*, полученных на основании подключаемых исходных файлов.

Источник данных — допустимые источники для наполнения списка подсказки:

- **Ключевые слова и функции языка viper** — стандартные конструкции языка **VIP**;
- **Функции для работы с Excel** (см. "Галактика ERP. Прикладные модули. Библиотека функций для работы с MS Excel.");
- **Скрипт** — пользовательские конструкции, наполняемые список подсказки с помощью скрипта (см. "[Примеры использования скриптов](#)"¹⁶⁷).

Такой скрипт можно выполнить в любой момент времени из меню [Скрипты](#)⁵³ или назначить его автоматическое выполнение определенному [Событию](#)¹¹¹, например, *Начало программы* или *Вызов подсказки кода*.

Параметры выпадающего списка:

- **Количество видимых строк** (по умолчанию 6);
 - **Стиль Windows-окна** — стиль локальной ОС.
При использовании данного стиля доступна возможность изменения размера списка мышью в текущем экземпляре **Viper**;
 - **Использовать иконки** — обозначение константных типов идентификаторов иконкой (см. "[Объект "Запись"](#)"¹⁶³).
- Другие типы обозначаются соответствующим для них наименованием.

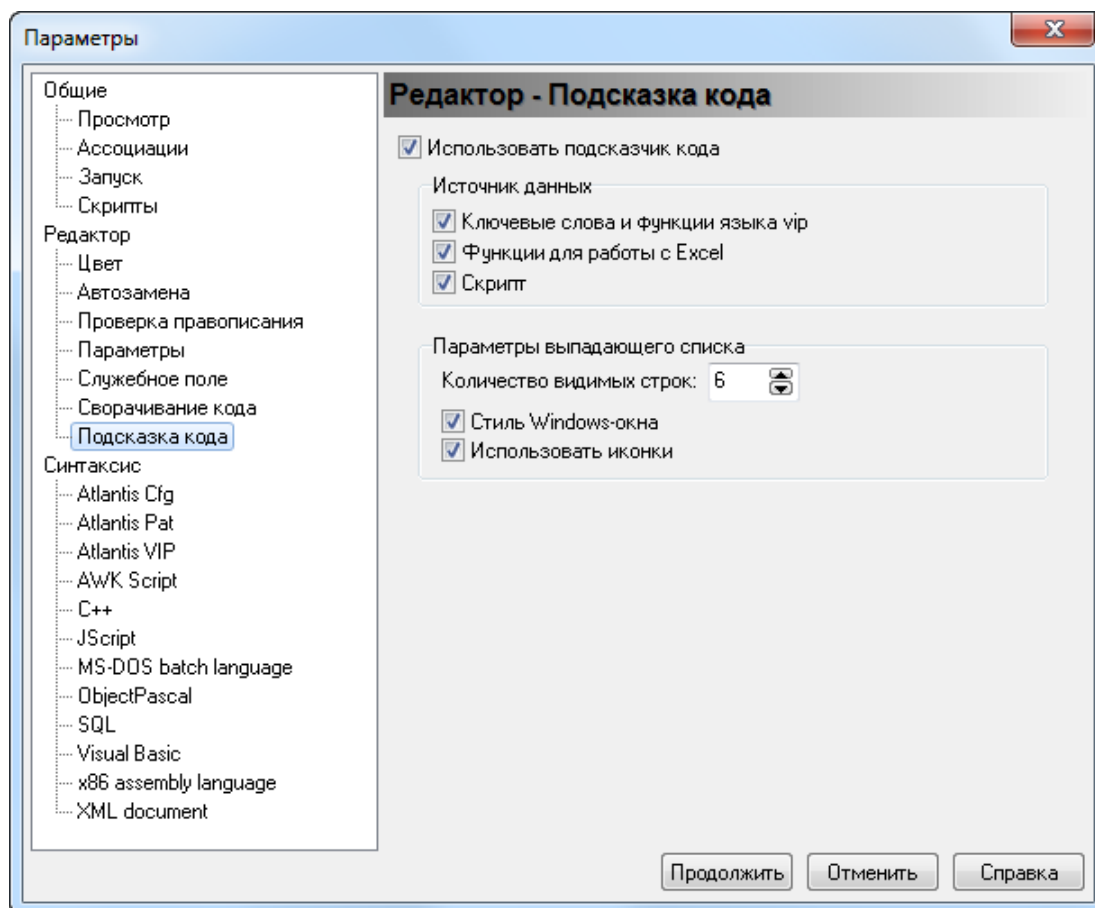


Рис. 93. Страница настроек "Подсказка кода"

5.2.3. Вкладка Синтаксис

Вкладка позволяет настроить подсветку синтаксической схемы в редакторе кода.

Выбор подсветки синтаксиса — варианты использования синтаксической раскраски:

- **По расширению файла** — применение синтаксической схемы на основании расширения файла. **Типы файлов** определяются в настройках синтаксических схем (см. "[Вкладка Atlantis VIP — Подсветка](#)¹²³").
 - **Для файлов без расширения использовать** — тип схемы, используемый для неизвестных типов файлов.
- **Использовать всегда** — применение подсветки синтаксической схемы, выбранной из списка, для файлов любого типа.
 При включении перечисленных выше настроек активируется автоматический разбор синтаксических конструкций языка **VIP** в окне ["Структура кода"](#)⁶¹.
- **Не использовать** — отключение подсветки синтаксиса. Тем не менее, применение подсветки синтаксиса возможно:
 - для активного редактора кода при помощи функции **Синтаксическая схема** (см. "[Меню Вид](#)"⁴⁶);
 - при использовании команды **Создать** (см. "[Меню Файл](#)"²⁴).

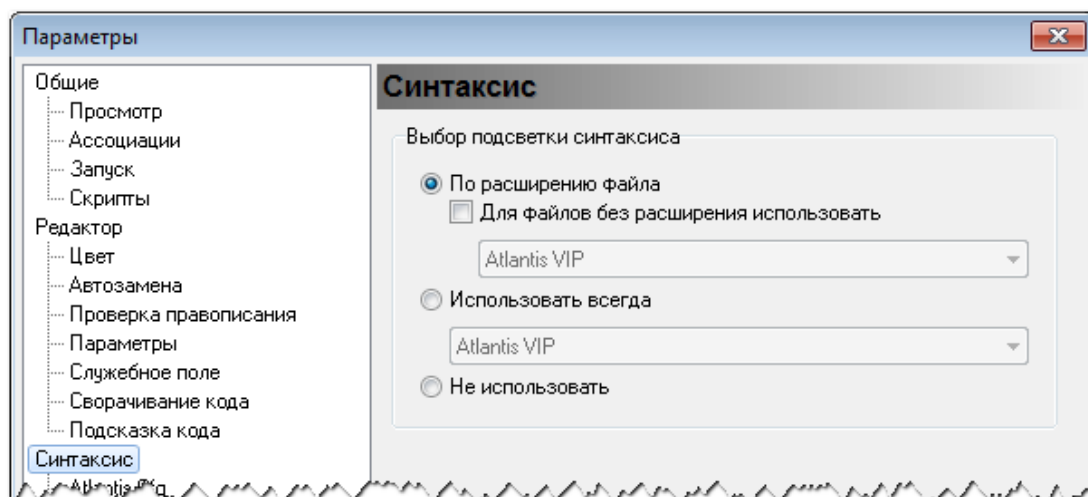


Рис. 94. Страница настроек "Синтаксис"

Настройки вкладки *Синтаксис* также учитываются для автоматического присвоения расширения файлу, который был создан без него. Тип расширения зависит от настройки, которая являлась установленной на момент создания файла:

- При выбранной настройке **По расширению файла** (и отключенной настройке **Для файлов без расширения использовать**) или **Не использовать** — присваивается расширение ***.txt**.
- В остальных случаях присваивается расширение, соответствующее синтаксической схеме, которая выбрана на вкладке *Синтаксис* из выпадающего списка (*Atlantis CFG* — **cfg**; *Atlantis Pat* — **pat**; *Atlantis VIP* — **vip**; *AWK Script* — **awk**; *C++* — **c++**; *JScript* — **js**; *MS-DOS batch language* — **bat**; *ObjectPascal* — **pas**; *SQL* — **sql**; *VisualBasic* — **vbs**; *x86 assembly language* — **ASM**, *XML document* — **xml**).

5.2.3.1. Синтаксические схемы

Синтаксическая схема содержит перечень типов файлов и предоставляет возможность упрощенной настройки данной коллекции. Вкладка *Синтаксис* включает в себя следующие синтаксические схемы с группой типов файлов:

- *Atlantis Cfg* (*.cfg);
- *Atlantis Pat* (*.pat);
- *Atlantis VIP*¹²³ (*.vip; *.vpp; *.gd; *.cp; *.def; *.dic; *.vih; *.vil; *.inc; *.var; *.mnu; *.mnh; *.rc0; *.buh; *.dlg; *.fld; *.prj; *.gcd; *.pan; *.han; *.frm; *.frn; *.cnf; *.ver; *.ccm; *.csc);
- *AWK Script* (*.awk);
- *C++* (*.c; *.cpp; *.h; *.hpp; *.hh);
- *JScript* (*.java);
- *MS-DOS batch language* (*.bat; *.cmd);
- *ObjectPascal* (*.pas; *.dpr; *.dpk; *.inc);
- *SQL* (*.sql);
- *Visual Basic* (*.bas);
- *x86 assembly language* (*.asm);
- *XML document* (*.xml; *.xsd; *.xsl; *.xslt; *.dtd).

При необходимости перечень типов файлов можно изменить.

Общий принцип настройки синтаксических схем одинаковый, и поэтому в данной документации будут рассмотрены параметры на основании схемы *Atlantis VIP*¹²³.

5.2.3.2. Atlantis VIP

Настройка синтаксической схемы *Atlantis VIP* состоит из следующих разделов:

- Подсветка;
- Автозаполнение;
- Автозамена;
- Справка.

5.2.3.2.1. Вкладка Atlantis VIP - Подсветка

Вкладка Подсветка определяет типы файлов для данной синтаксической схемы и управляет параметрами ее подсветки, для этого используются стиль шрифта, цвет символов и фона.

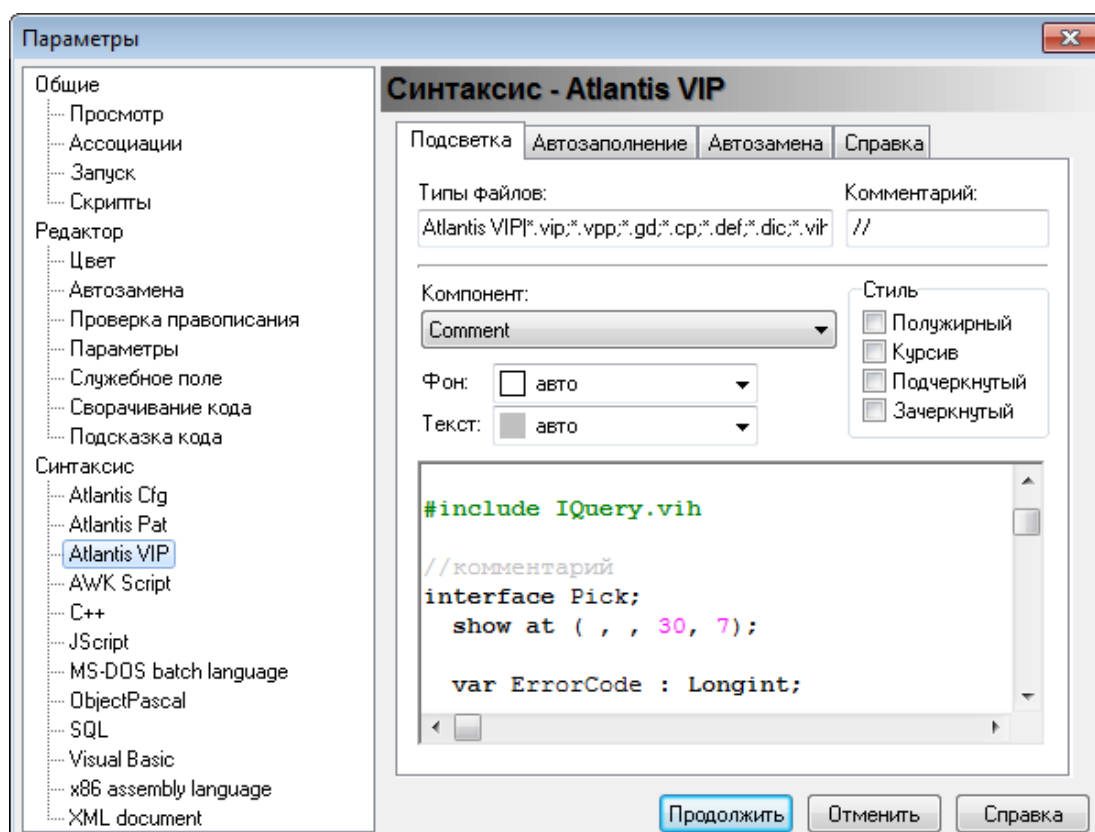


Рис. 95. Вкладка "Подсветка"

Типы файлов — расширения файлов, используемых в данной схеме.


Комментарий — символы комментария, используемые в данной синтаксической схеме.

Компонент — элементы программного кода, такие как числа, строки, идентификаторы и т. п., в том числе sql-конструкции. Каждый элемент содержит индивидуальные настройки стиля шрифта (*Полужирный*, *Курсив*, *Подчеркнутый*, *Зачеркнутый*), цвета фона и символов. Для выбора элемента следует нажать стрелку, направленную вниз.

Область предварительного просмотра позволяет просматривать настройки синтаксической раскраски и стиля на примере кода. Причем, по щелчку левой кнопкой мыши в данной области определяется соответствующий данному тексту компонент.

5.2.3.2.2. Вкладка Atlantis VIP - Автозаполнение

Вкладка Автозаполнение позволяет создавать и использовать в дальнейшем шаблоны синтаксиса базовых конструкций языка. Шаблоны автозаполнения содержатся в списке, который вызывается в редакторе по **Ctrl+J** (см. "[Сервис](#)⁷³").

Имя файла — полное имя файла, в котором хранятся шаблоны автозаполнения. Для выбора файла предназначена кнопка .

Символы разделители — символы, разделяющие синтаксические единицы.

Таблица предназначена для отображения списка с **Именем** и **Описанием** шаблонов.

[**Добавить**] — создание нового шаблона. Вызов команды осуществляет открытие диалогового окна, в котором следует ввести **Имя** и **Описание** шаблона.

[**Редактировать**] — редактирование текущего шаблона. Вызов команды открывает диалоговое окно, в котором можно изменить **Имя** и **Описание** текущего шаблона.

[**Удалить**] — удаление текущего шаблона из файла шаблонов.

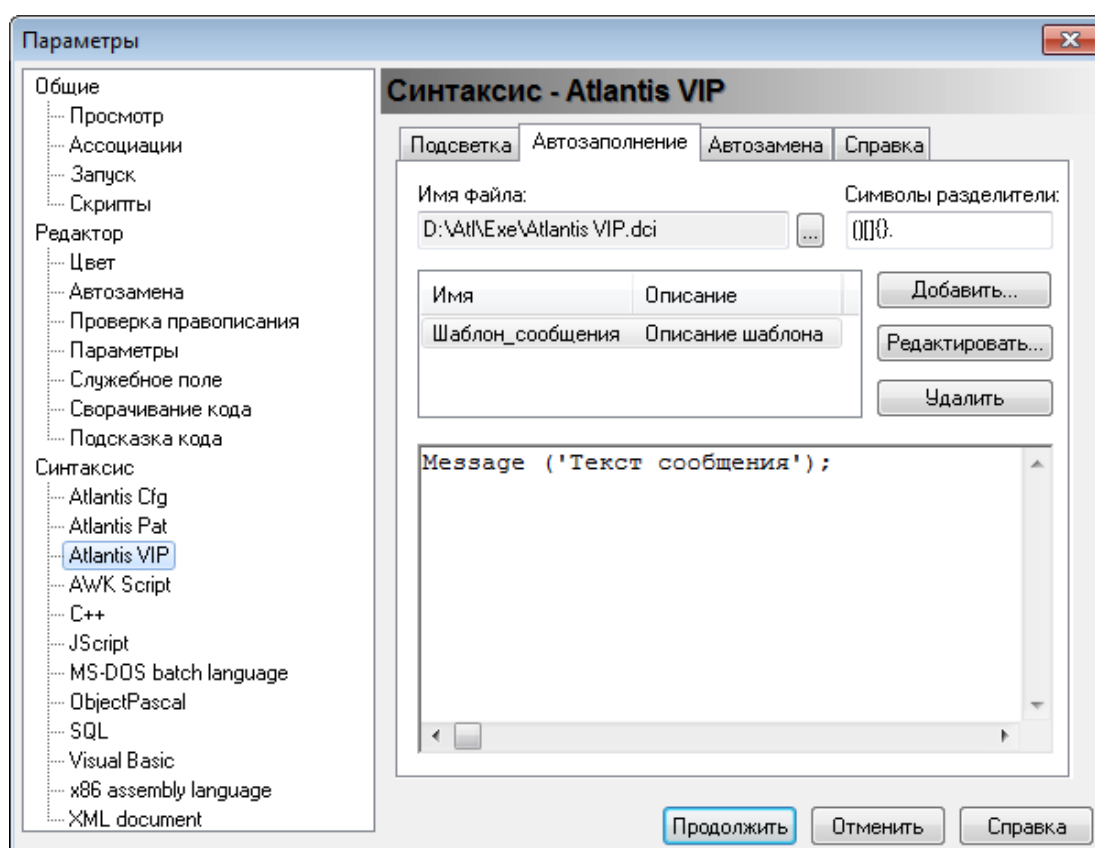


Рис. 96. Вкладка "Автозаполнение"

5.2.3.2.3. Вкладка Atlantis VIP - Автозамена

Вкладка Автозамена содержит настройки шаблонов для автозамены. Они ускоряют набор текста в редакторе кода за счет использования сокращений (см. "[Сервис](#)⁷³").


Файл автозамены — полное имя файла, в котором хранится список автозамен. Для выбора файла предназначена кнопка .

Таблица отображает список существующих шаблонов замен, которые содержатся в файле автозамены.

Для создания нового шаблона нужно заполнить следующие параметры:

- **Заменить** — вводимый текст (сокращение) в редактор кода, подлежащий замене.
- **На** — вариант замены введенного текста.

[Добавить] — создание нового шаблона, на основании указанных параметров.

[Заменить] — изменение текущего шаблона в списке, в соответствии с указанными параметрами.

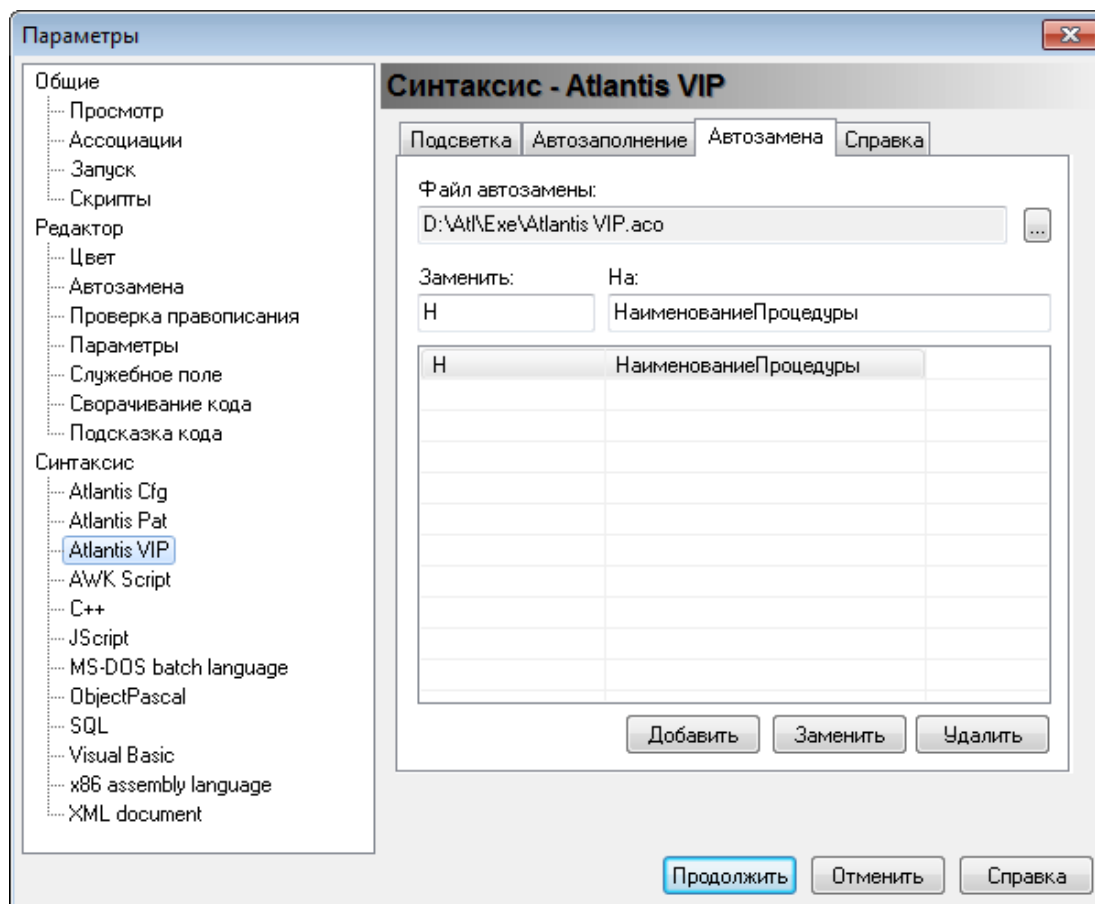



Рис. 97. Вкладка "Автозамена"

5.2.3.2.4. Вкладка Atlantis VIP - Справка

Вкладка Справка содержит полное имя файла справочной информации. Для изменения имени предназначена кнопка . Клавиша **Backspace** в этом поле очищает указанный адрес.

Данный параметр используется при вызове контекстной справки (см. "[Сервис](#)⁷³") и позволяет настроить вызов контекстной справки для различных синтаксических схем, а следовательно языков программирования.

5.3. Командная строка

Среда разработки **Viper** может быть запущена с дополнительными параметрами и ключами командной строки. Параметры командной строки позволяют выполнить некоторые действия без участия пользователя. В том числе предоставляют возможность использования **Viper** в пакетном режиме, например, для компиляции программы. Допол-

нительные параметры и ключи могут быть заданы при запуске через командную строку либо через параметры запуска, расположенные в настройках ярлыка приложения.

Существуют следующие ключи и параметры командной строки:

<файл-проекта> — имя загружаемого проектного файла, при пустом параметре среда загружается в соответствии с настройкой *Загрузка файлов проекта* (см. "[Вкладка Проект](#)"⁸⁷);

-ClearProject — удаление структуры проекта;

-CompileAll — запуск компиляции проекта;

-BuildAll — запуск пересборки проекта;

Процесс компиляции с ключом -CompileAll или -BuildAll останавливается в случае обнаружении ошибок, при этом учитывается настройка *Компилировать до первой ошибки* (см. "[Вкладка Компилятор VIP - Сообщения](#)"⁹⁵).

<файл> -AddBuild — открытие файла в редакторе;

<файл-проекта> -AddProject — добавление файла в сборку проекта, при этом сам файл в редакторе не открывается;

<элемент-компиляции> -Compile — запуск компиляции файла или выполнения профиля (при наличии такового в элементах сборки);

<элемент-компиляции> -Build — запуск пересборки файла или выполнения профиля (при наличии такового в элементах сборки);

-SkipError — игнорирование ошибок компиляции. Используется для безостановочной компиляции в случаях обнаружения ошибок.

-Exit — закрытие приложения **Viper** по завершении компиляции, при наличии ошибок флаг игнорируется. В сочетании с ключом -SkipError **Viper** закрывается даже при наличии ошибок сборки;

-Line <номер-строки> — установка курсора в строке указанного файла, открытого в редакторе;

-Col <номер-колонки> — установка курсора в колонке указанного файла, открытого в редакторе;

-Highl — выделение подсветкой строки, указанной с помощью ключа -Line или -Col;

<файл-скрипт> -Script — исполнение файла скрипта, при этом сам файл не открывается в редакторе;

-LogToFile — включение протоколирования в файл результатов работы среды (см. "[Диагностика исключительных ситуаций](#)"¹⁰⁴);

-LogToPipe — включение протоколирования в поток результатов работы среды (дополнительно см. в разделе, указанном в описании ключа -LogToFile).

Параметры командной строки могут быть использованы в комплексе. При разборе они обрабатываются последовательно слева направо, и неверное расположение параметров может вызвать различную реакцию при их обработке. Если есть запущенный экземпляр **Viper** и в среде включен параметр *Запускать только один экземпляр программы* (см. "[Вкладка Общие](#)"¹⁰⁵), то действия, заданные параметрами командной строки, будут перенаправлены запущенному экземпляру.

Примеры

Пример 1.

```
viper.exe D:\Test1.vpr -ClearBuild D:\Test2\Test2.prj -AddBuild D:\Test3\Test3.prj -AddBuild -BuildAll -Exit
```

Алгоритм разбора и выполнения командной строки:

- запуск **Viper**;
- загрузка проекта **D:\Test1.vpr**;
- удаление из проекта всех элементов сборки;
- добавление в проект двух *prj*-файлов;
- запуск пересборки всего проекта, в случае возникновения ошибок компиляции процесс будет остановлен;
- закрытие среды по завершении сборки, в случае обнаружения ошибок компиляции закрытие не произойдет.

Пример 2.

```
viper.exe D:\Test1.vpr D:\Test2\Test2.prj -Compile -SkipError  
-Exit
```

Алгоритм выполнения:

- запуск **Viper**;
- загрузка проекта **D:\Test1.vpr**;
- запуск компиляции указанного проектного файла (при его наличии в проекте);
- при возникновении ошибок компиляции процесс не будет остановлен;
- закрытие среды по завершении сборки даже в случае обнаружения ошибок компиляции.

Пример 3.

```
viper.exe D:\Samples\LotDemo.vip -AddProject -Line 42 -Col 38  
-Highl
```

Алгоритм выполнения:

- запуск **Viper**;
- загрузка последнего проекта в соответствии с параметром *Загрузка файлов проекта*;
- открытие в редакторе *VIP*-файла и добавление его в структуру проекта;
- в текущем редакторе установка курсора в указанной строке и колонке;
- подсветка строки под курсором.

5.4. Использование регулярных выражений

Регулярные выражения используются в качестве шаблона для поиска, подстановки символов или отображения определенной информации (например, при использовании команды *Найти*, *Заменить* или в параметрах профиля).

В состав регулярного выражения входят метасимволы. Они представлены в виде обычных символов, но обозначают определенное условие с учетом их последовательности.

Для определения подвыражений регулярного выражения могут использоваться круглые скобки.

Представление символов в регулярных выражениях:

- Отмена значения метасимвола. Обратная косая черта перед метасимволом отменяет его специальное значение. Например, символ "^" соответствует началу строки, а сочетание символов "\^" соответствует "^". Аналогично, "\\" соответствует символу "\" и т. д.
- Любой один символ задается с помощью метасимвола ".".

- **Группа символов.** Заключение списка символов в квадратные скобки означает, что на данном месте в строке может находиться один из перечисленных символов. Добавление символа "^" после "[" указывает на то, что группа не соответствует символу в строке.
- **Интервал группы символов.** Для указания интервала внутри списка используется символ "-". Так, "a-z" представляет все символы между "a" и "z" включительно. Символ "-" обозначает собственное значение в списке, если он установлен в начале или в конце списка, а также при использовании обратного следа. Например, три следующих группы определяют один и тот же класс из трех символов: "[az]", "[az-]" и "[a-z]". И все они отличаются от группы "[a-z]", которая задает группу, содержащую 26 символов от a до z. Если необходимо включить в группу символ "]", то его достаточно поместить в начало списка. Например, "[]-a]" соответствует любому символу в интервале от "]" до "a".
- **Специальные символы:** "\n" соответствует новой строке, "\t" — табулятору, "\r" — переводу каретки, "\f" — переводу формата (form feed) и т. д.
- **Альтернативные значения.** Существует возможность использования альтернативных значений, разделяя их символом "|". Например, "fee|fie|foe" соответствует любой из последовательностей символов "fee", "fie" или "foe" в анализируемой строке. Такой же результат даст запись "f(e|i|o)e". Первая альтернатива включает все от последнего ограничителя (начало шаблона, "(" или "[") до первого символа "|". Последняя альтернатива содержит все от последнего "|" до следующего разделителя. Альтернативы принято заключать в скобки, чтобы улучшить читаемость выражения и снизить вероятность ошибки.

Альтернативы обрабатываются слева направо. Используется первая, для которой найдено соответствие. Например, при разборе строки "barefoot" с использованием шаблона "foo|foot" соответствие будет установлено только с "foo", т. к. данное значение проверялось первым и оказалось соответствующим анализируемой строке.

Следует также помнить, что "|" в квадратных скобках интерпретируется как литерал. Так что шаблон "[fee|fie|foe]" эквивалентен шаблону "[feio]".

- **Фигурные скобки** можно использовать после любой единицы (символ, группа, подвыражение и т. п.) регулярного выражения в виде {n,m}. Где n указывает минимальное, а m — максимальное количество повторений данной единицы. Форма {n} эквивалентна {n,n} и соответствует n повторениям. Форма {n,} соответствует n или более повторениям.


Фигурные скобки в любом другом контексте трактуются как обычный символ. Для значений n и m нет ограничений, но большие значения могут вызвать проблемы с оперативной памятью и скоростью обработки.

Модификатор "*" эквивалентен "{0,}", модификатор "+" эквивалентен "{1,}" а модификатор "?" эквивалентен "{0,1}".

Ниже в таблице содержатся определения основных элементов регулярных выражений:

Метасимвол	Описание
^	начало строки
\$	конец строки
.	любой символ
\	следующий символ трактуется в своем собственном значении
*	предыдущий символ соответствует 0 или более повторений
+	предыдущий символ соответствует 1 или более повторений
?	предыдущий символ соответствует 0 или 1 повторений

Метасимвол	Описание
{n}	соответствует ровно n повторений
{n,}	соответствует по меньшей мере n повторений
{n,m}	соответствует не менее n и не более m повторений
[aeiou0-9]	соответствует одной из букв списка "a", "e", "i", "o", "u" или цифре от "0" до "9"
[^aeiou0-9]	соответствует любым символам кроме букв "a", "e", "i", "o", "u" и цифр от "0" до "9"
\w	соответствует цифрам и буквам (включая "_")
\W	соответствует всем символам, кроме цифр и букв (включая "_")
\d	соответствует цифрам
\D	соответствует нецифровым символам
\s	соответствует пробельным символам ([\t\n\r\f])
\S	соответствует непробельным символам

В функциях **Viper**, где разрешено использование регулярных выражений, при нажатии кнопки  открывается меню для выбора основных метасимволов. Меню содержит следующие варианты:

Метасимвол	Пункт меню
.	Любой символ
[]	Символ, входящий в ряд
[^]	Символ, не входящий в ряд
^	Начало строки
\$	Конец строки
*	Ноль или более соответствий
+	Одно или более соответствий
{}	n соответствий
{,}	от n до m соответствий
\w	Буквенно-цифровой символ
\W	Не буквенно-цифровой символ
\d	Цифровой символ
\D	Не цифровой символ
\s	Пробел
\S	Не пробел

Примеры

Действительное число (например, '13.88e-4' или '-7E2'):

```
([+|-]? \d+ (\. \d+)? ([eE] [+|-]? \d+)?
```

Адрес e-mail (например, 'viper@galaktika.by'):

```
([_a-zA-Z\d\-\.\.]+@[_a-zA-Z\d\-\.\.]+(\. [_a-zA-Z\d\-\.\.]+)+)
```

Адрес интернет (например, 'http://forum.galaktika.ru'):

```
([Ff] [Tt] [Pp] | [Hh] [Tt] [Tt] [Pp]) :// ([_a-zA-Z\d\-\.\.]+(\. [_a-zA-Z\d\-\.\.]+) ) ((/[_a-zA-Z\d\-\.\.\.]+)+) *
```

6. Компиляция программ

В среде **Viper** функции компиляции доступны только в [проекте](#)^[81].

Компиляция может выполняться с помощью:

- встроенного компилятора **ViperCompile.exe** (см. "[Настройка компилятора](#)"^[130]);
- любого внешнего компилятора. Для этого необходимо использовать [профиль](#)^[147].

Также возможности компилятора предусматривают одновременное использование одного проекта в различных копиях **Viper**. Это позволяет выполнять параллельную компиляцию нескольких элементов проекта для ускорения его сборки. Данная возможность зависит от параметра *Запускать только один экземпляр программы* (см. "[Вкладка Общие](#)"^[105]). При попытке одновременной компиляции в разных копиях **Viper** осуществляется проверка на доступ/запись к общим ресурсам:

- ресурсный файл;
- служебный ресурсный файл;
- рабочий ресурсный файл конфигулятора.

Для корректного выполнения параллельной компиляции необходимо, чтобы ресурсы в разных экземплярах **Viper** имели разные имена (см. "[Вкладка Компилятор VIP — Ресурсы](#)"^[92]). Например, включить в имя ресурса имя компилируемого файла:

- `[$FileName]$[ExtRes]` — ресурсный файл;
- `[$FileName]Atl$[ExtRes]` — служебный ресурсный файл.

6.1. Настройка компилятора

Для использования встроенного компилятора **ViperCompile.exe** следует настроить необходимые параметры (подробнее см. "[Вкладка Компилятор VIP](#)"^[89]).

В среде **Viper** реализована возможность самостоятельного выбора компилятора необходимой версии (подробнее о версиях см. "[Меню Справка](#)"^[55]), для этого предназначен параметр *Каталог компилятора*. В рамках одного проекта для файлов компиляции и пакетов могут одновременно использоваться разные версии компилятора.

Дополнительные настройки, используемые при запуске компиляции, см. в разделе "[Вкладка Общие - Запуск](#)"^[110].

6.2. Использование компилятора

При использовании **ViperCompile.exe** компиляция осуществляется в одном из двух режимов:

- Полная пересборка всех файлов с исходным кодом, независимо от даты их исправления. Для этого предназначена команда *Пересобрать* (см. "[Меню Проект](#)"^[50]).
- Перекомпиляция только измененных файлов исходного кода, после предыдущей компиляции. Это позволяют выполнить команды *Компилировать проект* и *Компилировать/Выполнить*.

Компиляцию [элементов сборки](#)^[85] в **Viper** можно выполнить двумя способами:

- Компиляция *текущего* элемента сборки. Для этого предназначены команды *Компилировать/Выполнить* и *Пересобрать* контекстного меню элемента сборки, а

также *Компилировать файл/Выполнить профиль* в главном меню (см. "[Меню Проект](#)⁵⁰").

Текущим является элемент сборки, помеченный жирным шрифтом. Выбрать элемент *Текущим* можно щелчком средней кнопки мыши или с помощью соответствующего контекстного меню.

Существует возможность компилировать только некоторые элементы сборки, для этого нужно выбрать элементы, удерживая клавишу **Ctrl**.

- Компиляция всего проекта, т. е. всех элементов сборки осуществляется последовательно. Для этого предназначены команды *Компилировать* и *Пересобрать проект* главного [Меню Проект](#)⁵⁰ и в контекстном меню корневого узла *=Менеджера проекта=*. При выполнении данных команд каждый компилируемый элемент поочередно выделяется в списке *=Менеджера проекта=*.

Поиск подключаемого файла с исходным кодом осуществляется в следующем порядке:

- 1) при наличии пути для данного файла в исходном коде поиск производится в соответствующем расположении на диске;
- 2) при отсутствии пути для данного файла в исходном коде поиск производится в каталоге соответствующего файла сборки;
- 3) *Список каталогов для поиска подключаемых файлов* (см. "[Вкладка Компилятор VIP — Каталог](#)⁹¹").

Выполнению компиляции выделен отдельный физический процесс, что позволяет при компиляции продолжать работу с редактором. Основной процесс всегда опрашивает процесс компилятора на его работоспособность, в случае задержки ответа или аварийного завершения процесса компиляция будет прервана, а в *=Окне вывода*⁶⁸ отобразится соответствующая информация.

Для досрочной остановки компиляции предназначен пункт в главном меню *Проект > Остановить компиляцию/выполнение профиля*. Превышение процесса остановки более 5 секунд завершает его аварийно (процесс уничтожается).

Информация о выполнении процесса компиляции и версия соответствующего компилятора отображается в *=Окне вывода=* (см. "[Меню Вид](#)⁴⁶"). Если данное окно скрыто, то в начале/в конце компиляции элемента сборки оно раскроется. Когда приложение *Viper* неактивно, об окончании процесса компиляции оповещает мигание вкладки приложения на панели задач. Предупреждения/ошибки, обнаруженные при компиляции, отображаются в окне *=Сообщения*⁶⁹. Если данное окно скрыто, то автоматически оно раскроется только после завершения процесса компиляции всех участвующих элементов сборки при наличии записи *Ошибка* в списке сообщений.

Скрыть данные окна можно одним из способов:

- Закрыть командой главного меню *Вид > Окно сообщений (Окно вывода)*.
- Свернуть с помощью кнопки в правом верхнем углу окна, при этом, чтобы свернуть окно в процессе вывода сообщений, достаточно установить курсор в редакторе и нажать клавишу **Esc**.

Существует возможность настройки оповещений (см. "[Вкладка Компилятор VIP — Сообщения](#)⁹⁵") и сохранения истории сообщений, а также фильтрация оповещений компилятора по типам или по мере ввода символов в списке окна *=Сообщения=*.

7. Отладка программ

7.1. Настройка отладчика

Параметры отладчика определяются в группе настроек проекта *Отладочная информация* (см. "[Вкладка Компилятор VIP](#)^[89]").

Для выбора программы, используемой для запуска приложения в режиме отладки, следует указать полное имя файла в поле *Имя исполняемого файла приложения* (см. "[Вкладка Отладчик VIP](#)^[100]").

Например, для отладки *Галактики*:

```
atlexec.exe
```

или отдельного интерфейса:

```
vip.exe /r
```


Отладка может выполняться параллельно компиляции или одновременно с другим сеансом отладки в различных копиях *Viper*. При этом использование одного и того же отлаживаемого приложения возможно, но из разных рабочих каталогов.

Настройки из раздела *Компилятор VIP* отлаживаемым приложением не используются. Поэтому необходимые ресурсы должны быть зарегистрированы в репозитории, а для приложения настроен файл конфигурации (параметры базы данных, лицензия и т. д.). Его адрес указывается в параметре *Текущий каталог*. При этом файл конфигурации должен иметь имя по умолчанию либо подключаться явно через параметр */c*.

Автоматический поиск отлаживаемого файла с исходным кодом производится последовательно в следующих расположениях:

- 1) при наличии пути для данного файла в исходном коде поиск производится в соответствующем расположении на диске;
- 2) при отсутствии пути для данного файла в исходном коде поиск производится в каталоге соответствующего файла сборки;
- 3) одноименный файл, открытый в редакторе;
- 4) *Список каталогов для поиска подключаемых файлов* следующего элемента сборки:
 - a) текущий элемент (см. функцию *Выбрать текущим* в разделе "[Окно менеджера проекта](#)^[58]");
 - b) родитель текущего элемента сборки.
- 5) параметры проекта:
 - a) *Список каталогов для поиска подключаемых файлов*;
 - b) *Дополнительный список каталогов для поиска файлов* текущей конфигурации отладчика *VIP*.

При отсутствии файла в результате автоматического поиска формируется диалоговое окно для указания файла пользователем. Окно содержит имя отсутствующего файла и позволяет указать *Путь к файлу*. По кнопке *[Обзор]* открывается стандартное диалоговое окно выбора файла на диске.

 В поле *Путь к файлу* можно указать путь каталога с добавлением *'*'*, тогда поиск необходимого файла, а также дальнейших отлаживаемых файлов будет выполняться во всех вложенных каталогах по данному пути.

Добавить в дополнительный список каталогов для поиска файлов — добавление указанного пути в **Дополнительный список каталогов для поиска файлов** (см. "[Вкладка Отладчик VIP¹⁰⁰](#)").

[**Продолжить**] — продолжить отладку с использованием указанного пути к файлу.

[**Отменить**] — продолжить отладку без отсутствующего файла.

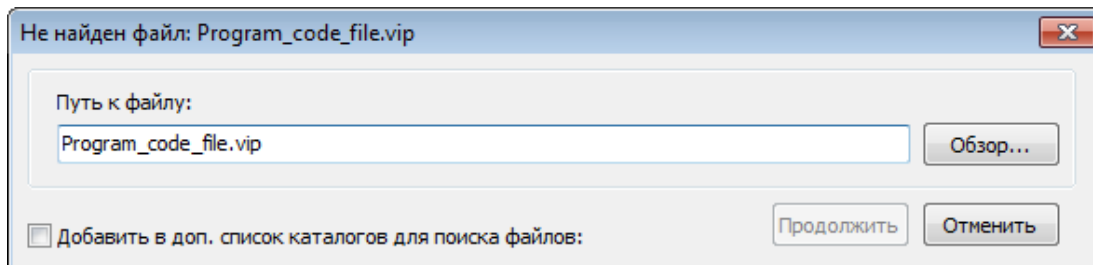



Рис. 98. Диалоговое окно для выбора файла

7.2. Процесс отладки

Перед началом отладки следует выбрать (пометить) необходимую конфигурацию. Для этого необходимо на инструментальной панели **Отладка** нажать стрелку пиктограммы **Начать отладку**  и в открывшемся списке с помощью ЛКМ пометить нужную конфигурацию. Существующие конфигурации хранятся в параметрах проекта на вкладке [Отладчик VIP¹⁰⁰](#), при необходимости настройки конфигурации можно изменить или создать новую. После выбора конфигурации для отладки отлаживаемое приложение будет создано в соответствии с данной конфигурацией. Для запуска процесса отладки необходимо вызвать функцию главного меню [Отладка⁵¹](#) > **Начать отладку**. На время создания процесса, появляется модальное окно визуализации — редактор становится недоступным до завершения данного этапа. В случае длительности создания процесса более одной минуты, каждые следующие 10 секунд осуществляется проверка на его работоспособность.

На начальном этапе создание процесса отладки можно отменить по нажатию кнопки [**Отмена**].

Закончить отладку во время ее исполнения можно с помощью главного меню **Отладка** > **Закончить отладку**, при этом отлаживаемому приложению будет послан сигнал о закрытии. В случае, если отлаживаемое приложение не сможет закрыться самостоятельно (например, при выполнении длительного процесса либо отображении модального окна), пользователю будет предложен запрос о его принудительном завершении.

Функции отладки доступны при условии наличия отладочной информации для необходимого отлаживаемого интерфейса (см. "[Вкладка Компилятор VIP⁸⁹](#)"). Подключение отладочной информации происходит следующими способами:

- Автоматически, при наличии точки останова, добавленной перед запуском интерфейса.
- Функция **Отладить** в окне **=Все интерфейсы=** предназначена для интерфейсов, еще не открытых в **Галактике**, и выполняет подключение отладочной информации независимо от наличия точек останова. Функцию нужно использовать в режиме отладки перед открытием самого интерфейса. В дальнейшем после открытия интерфейса устанавливаемые ТО будут подхвачены без необходимости переподключения данного интерфейса.
- Функция **Отладить** в окне **=Загруженные интерфейсы=** предназначена для интерфейсов, открытых в **Галактике**, и выполняет подключение отладочной информации.

мации независимо от наличия точек останова. По команде *Отладить* для выбранной записи в окне *Интерфейсы* добавится обозначение "зеленый шарик". В дальнейшем устанавливаемые ТО будут подхвачены без необходимости переоткрытия данного интерфейса.

Отладчик может находиться в одном из двух режимов, который одинаково отражается в окне *Загруженные интерфейсы* и *Все интерфейсы*:

- В состояние *Останов* (красный шарик) он попадает при достижении точки останова или по завершении команды *Шаг*, *Трассировка*, *Выйти из текущего блока*, *Выполнить до курсора*. При этом отлаживаемое приложение находится в заблокированном состоянии, а фокус переводится в окно редактора.
- В состоянии *Выполнение* (зеленый шарик) сразу после старта, а также в момент выполнения команд *Запустить*, *Шаг*, *Трассировка*, *Выйти из текущего блока*, *Выполнить до курсора*. При этом состояние отлаживаемого приложения разблокируется.

Команды управления процессом отладки, а также список инструментальных окон содержатся в главном меню *Отладка*^[51].

При выполнении команд *Шаг*, *Трассировка*, *Выйти из текущего блока* или *Продолжить отладку* осуществляется открытие файла с исходным кодом отлаживаемой программы, для вкладки файла устанавливается соответствующий статус, а в служебном поле появляются отметки напротив строк, для которых в ресурсе есть отладочная информация. Помимо этого значок файла в *Менеджере проекта* заменяется на оранжевую стрелку. Указатель позиционируется на отлаживаемой строке в редакторе, которая тоже помечается оранжевой стрелкой в служебном поле.

При выполнении трассировки существует возможность выйти из текущего блока, т. е. вернуться из описания процедуры/функции к месту ее вызова. Для этого необходимо вызвать функцию в главном меню *Отладка* > *Выйти из текущего блока* или нажать соответствующую комбинацию клавиш (**Alt+F8**).

В процессе отладки можно изменять код программы и устанавливать точки останова, отслеживать значение переменных, элементов структуры (например, функций) и составных записей (например, поле таблицы) по наведению указателя на соответствующее выражение в редакторе, а также с помощью окна *Локальные переменные*^[138] или *Выражения и переменные*^[137].

Щелчок левой кнопки мыши в служебном поле управляет точками останова для соответствующей строки. Точки останова применяются в строках с исполняемым кодом и соответствуют одному из режимов: включенные, выключенные, с отладочной информацией или без нее. Информация о точках останова помещается в соответствующее инструментальное окно (см. *Окно точек останова*^[135]).

В процессе отладки доступна возможность изменения файла с исходным кодом и перекомпиляция ресурса. При этом процесс отладки завершать необязательно, а выполнить компиляцию можно, используя один из двух режимов:

- **Автоматический** — режим, в котором контроль за отлаживаемым приложением возлагается на среду *Viper*. В данном режиме команда компиляции может быть активирована как самим пользователем (подробнее о компиляции см. в разделе *Использование компилятора*^[131]), так и автоматически по мере изменения исходного кода, соответствующего отлаживаемому ресурсу. Т. е. после того, как отлаживаемый исходный код был изменен, но не скомпилирован, на очередном отладочном шаге формируется запрос о пересборке соответствующего ресурса. При выполнении компиляции необходимые ресурсы автоматически отключаются для выполнения компиляции, и по завершении ее выполнения будут автоматически подключены. При выполнении компиляции производится автоматическое сохранение изменений в файлах компилируемого ресурса.

- **Интерактивный** — режим, когда возможность управления ресурсами предоставляется пользователю. Данный режим позволяет подключать/отключать ресурсы, отсутствующие в репозитории или по каким-то причинам не отключенные при выполнении компиляции. Для управления ресурсами предназначены команды контекстного меню элемента сборки *Подключить* ресурс к приложению и *Отключить от приложения*.

После подключения ресурса к отлаживаемому приложению в *=Окне вывода=* отображается приоритет данного ресурса. При необходимости пользователь может изменить приоритет подключаемого ресурса. Для ввода приоритета необходимо использовать функцию *Подключить ресурс с приоритетом*, которая содержится в контекстном меню окна *=Менеджер проекта=*.

Подробнее о возможностях редактора **Viper** в процессе отладки описано в следующих разделах.

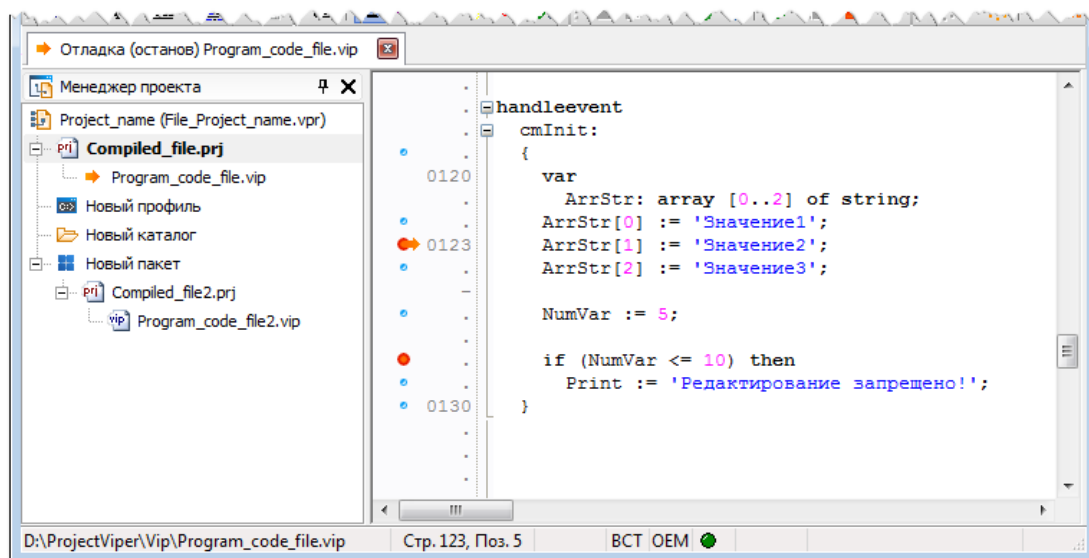


Рис. 99. Окно редактора исходного текста

7.3. Окно точек останова


Точка останова — это выделенная инструкция программы, на которой автоматически останавливается выполнение программы. Точки останова предназначены для принудительной остановки программы в нужном месте. Информация о точках и возможность управления ими содержится в окне *=Точки останова=*. Добавить/удалить точку останова можно с помощью команды *Точка останова* (см. "[Меню Отладка](#)"⁵¹) или щелчком мыши в служебном поле необходимой строки.

Использование точек возможно только в файлах типа **Atlantis VIP** (см. "[Вкладка Atlantis VIP — Подсветка](#)"¹²³) при наличии открытого проекта *.vpr (см. "[Меню Проект](#)"⁵⁰). Установка точек осуществляется на любом этапе работы в среде.

В режиме отладки строки отлаживаемого кода помечаются определенным значком в зависимости от наличия в них отладочной информации:

- красный шарик — точка останова с отладочной информацией. При достижении данной точки отладчиком выполнение программы приостанавливается. Управление переходит к отладчику **Viper**, а указатель позиционируется в строке с данной точкой. Программист получает возможность посмотреть значения переменных, изменить их, назначить новые точки останова или выполнить другие действия по отладке программы;

- серый шарик — точка останова в выключенном состоянии или без отладочной информации;
- синий шарик — наличие отладочной информации для данной строки в ресурсе;
- пусто — отсутствие отладочной информации для данной строки в ресурсе.

 *Отладчик останавливается на точках останова, имеющих отладочную информацию и находящихся в состоянии Включена.*

В окне редактора точка смещается вместе с назначенной строкой и удаляется при выполнении команд по удалению строки (см. "[Меню Блок дополнительно](#)"³³").

Запись о размещенных точках помещается в окно *Точки останова* с информацией об имени файла и номером строки.

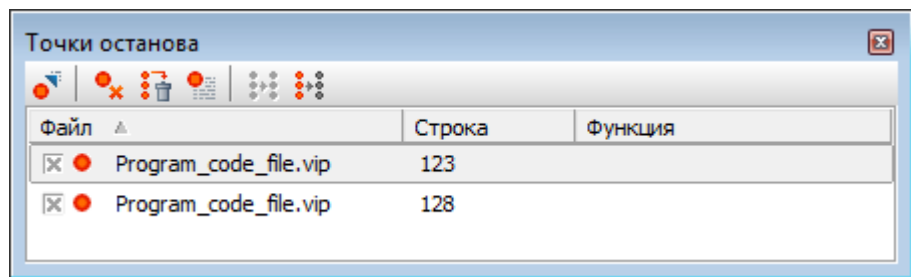


Рис. 100. Окно "Точки останова"

Предусмотрена возможность сортировки списка по любой из колонок таблицы.

На инструментальной панели окна содержатся команды:

- *Перейти* — открытие соответствующего файла в редакторе кода и отображение курсора в строке с текущей точкой останова. При этом поиск файла осуществляется в каталогах параметров проекта (см. "[Использование компилятора](#)"¹³¹").
- *Удалить* — удаление текущей точки останова из проекта.
- *Удалить все* — удаление всех точек останова из проекта.
- *Свойства точки останова* — дополнительные параметры использования точки останова.

Для назначения состояния всем точкам в списке используются команды *Включить* или *Выключить*. *Переключить состояние* текущей точки можно с помощью галочки или в контекстном меню соответствующей строки списка, при этом выключенная точка в процессе отладки не используется. Информация об установленных точках хранится в файле проекта.

В контекстном меню окна *Точки останова* дублируются команды инструментальной панели.

Для поиска точки останова в списке существует возможность фильтрации по введенным символам. Фильтр распространяется на все столбцы. Отмена фильтрации выполняется по клавише **Delete** и автоматически при закрытии данного закрепленного окна.

7.4. Окно стека

Окно *Стек* предназначено для отображения очереди вызовов подпрограмм и предоставляет возможность отслеживать позицию возврата из подпрограмм (функций, обработчиков прерываний) в программу.

Для раскрытия окна предназначен пункт главного меню *Отладка > Окно стека*.

В процессе отладки информация о вызовах программы помещается в список.

При запуске интерфейса и вызове подпрограммы или возникновении прерывания в окно добавляется адрес возврата. Последующие вложенные вызовы заносят в стек очередную информацию для возврата.

В списке окна содержится информация об имени функции, файла и интерфейса, а также номере позиции и уровне. После завершения выполнения подпрограммы, адрес соответствующей подпрограммы удаляется из списка.

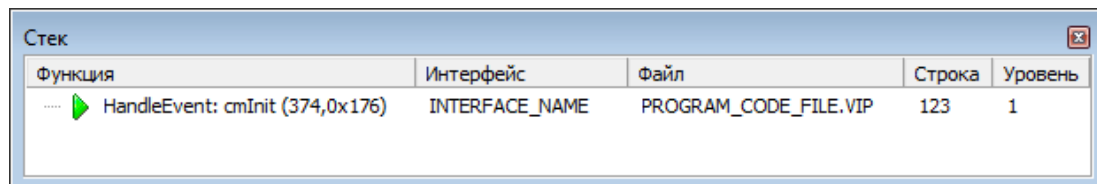


Рис. 101. Окно стека вызовов

Возможность перехода в место вызова осуществляется двойным нажатием на строку соответствующего адреса, при этом исходный файл открывается в редакторе кода с позиционированием курсора в строке из адреса.


В окне «Стек» существует возможность фильтрации по введенным символам. Фильтр распространяется на все столбцы. Отмена фильтрации выполняется по клавише **Delete** или автоматически при закрытии данного закрепленного окна.

По завершении процесса отладки список окна «Стек» очищается.

7.5. Окно выражений и переменных

Окно «Выражения и переменные» позволяет управлять значениями необходимых выражений в процессе отладки.

Для открытия окна предназначен пункт главного меню *Отладка > Окно выражений и переменных*. Возможности окна позволяют добавлять переменные и выражения, чтобы отслеживать и изменять их значения в текущем контексте отладки.

 Окно «Выражения и переменные» позволяет просматривать буфер таблиц (при указании имени таблицы и добавления `.buffer`) и содержимое структур `record`, `array` и `vipinterface`.

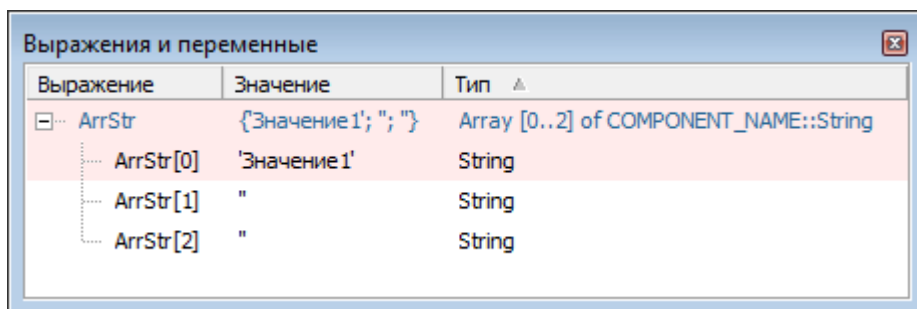


Рис. 102. Окно выражений и переменных

Команды управления списком окна выражений:

- **Добавить** — добавление нового выражения в список. При этом если окно «Выражения и переменные» закрыто, оно открывается. В качестве выражения используется выделенный текст в редакторе кода или идентификатор, на котором установлен курсор.

Также добавить выражение можно с помощью главного меню *Отладка > Добавить выражение* или "перетаскиванием" из редактора в окно *=Выражения и переменные=*.

- *Изменить* — редактирование имени или значения основного элемента списка. Также вызов команды осуществляется двойным щелчком левой кнопки мыши на элементе списка, при этом открывается диалоговое окно *=Вычислить и модифицировать=*.

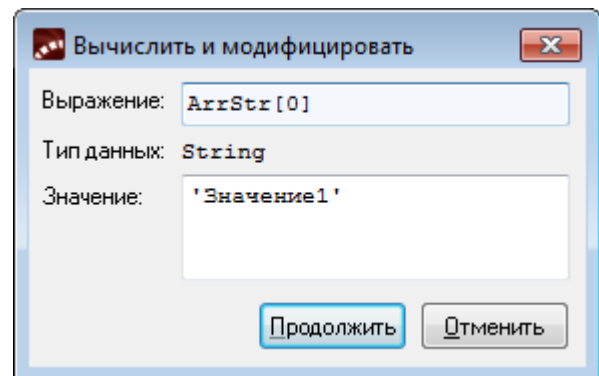


Рис. 103. Окно редактирования переменной

- *Удалить выбранное* — удаление текущего выражения из списка. Выбор нескольких элементов списка осуществляется при удержании клавиши **Ctrl**.
- *Удалить все* — очистка списка выражений.
- *Копировать имя* — копирование имени текущего выражения в буфер обмена.
- *Копировать значение* — копирование значения текущего выражения в буфер обмена.
- *Переместить вверх/вниз* — перемещение текущей записи в соответствующем направлении по списку. Функция доступна при включенной сортировке столбцов.
- *Обновить* — обновление значений в выражениях текущего контекста отладки.
- *Короткие имена* — скрытие имени родительского выражения в его вложенных элементах.

В процессе отладки оранжевым цветом подсвечиваются записи, значения которых изменились последними. Выбранное выражение всегда находится в области видимости, при этом список окна не прокручивается, что позволяет наблюдать за изменениями данного выражения.

Записи в окне *=Выражения и переменные=* можно сортировать: по наименованию выражения, значению или типу. Настройка сортировки сохраняется в параметрах среды и устанавливается при последующих запусках приложения.

Дополнительно в окне *=Выражения и переменные=* существует возможность фильтрации списка по мере ввода текста. Фильтр распространяется на все столбцы. Отмена фильтрации происходит автоматически при обновлении списка выражений, а так же с помощью клавиши **Delete**.

В списке доступна возможность выбора нескольких выражений при удерживании клавиши **Ctrl**.

Добавленные выражения хранятся в файле проекта.

7.6. Окно локальных переменных

Окно *=Локальные переменные=* позволяет управлять значениями выражений текущего сеанса отладки.

Для открытия окна предназначен пункт главного меню *Отладка > Окно локальных переменных* (см. "[Меню Отладка](#)"⁵¹). В процессе отладки окно позволяет просматривать список переменных загруженного интерфейса и изменять их значения.

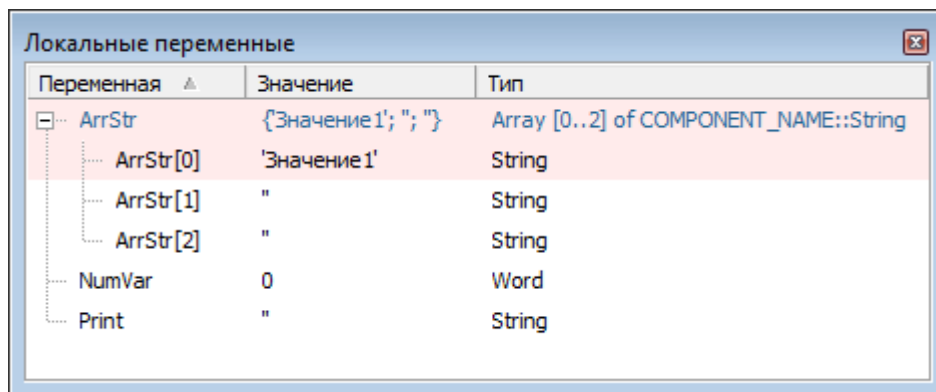


Рис. 104. Окно локальных переменных

Список выражений формируется и изменяется в процессе отладки автоматически. При этом оранжевым цветом подсвечивается группа элементов списка, значения которых изменились последними.

Также возможна сортировка списка по его заголовкам: *Переменная*, *Значение*, *Тип*. Настройка сортировки сохраняется в параметрах среды и устанавливается при последующих запусках приложения.

Команды управления списком окна локальных переменных:

- *Изменить значение* — редактирование значения основного элемента списка. Вызов команды также осуществляется двойным щелчком левой кнопки мыши на элементе списка, при этом открывается диалоговое окно *Вычислить и модифицировать*.
- *Копировать имя* — копирование в буфер обмена имени текущего выражения.
- *Копировать значение* — копирование в буфер обмена значения текущего выражения.
- *Обновить* — обновление значений в выражениях списка.
- *Короткие номера* — скрытие имени родительского выражения в его вложенных элементах.

В окне *Локальные переменные* существует возможность фильтрации списка по мере ввода текста. Фильтр распространяется на все столбцы и уровни вложенности. Отмена фильтрации происходит автоматически при обновлении списка выражений или закрытии данного закрепленного окна, а также с помощью клавиши **Delete**.

7.7. Окно интерфейсов

Окно *Интерфейсы* отображает список интерфейсов всех подключенных ресурсных файлов и позволяет изменять их состояние отладки. Для открытия окна предназначен пункт главного меню *Отладка > Все интерфейсы* (см. "[Меню Отладка](#)"⁵¹).

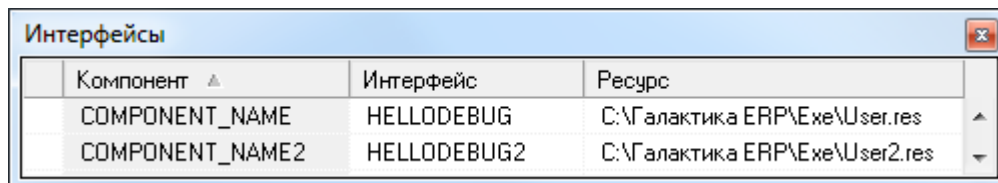


Рис. 105. Окно всех загруженных интерфейсов

Список окна формируется при запуске процесса отладки (см. "[Меню Отладка](#)"⁵¹) и изменяется при выполнении команд отключения/подключения ресурса. Процесс формирования

списка отмечается индикатором и соответствующим сообщением в «Окне вывода». В окне содержится информация о подключенных ресурсных файлах к отлаживаемому приложению (*Компонент, Интерфейс, Ресурс*).

В первой колонке отражается состояние интерфейса по отношению к отладчику. Оно определяется наличием значка в виде зеленого, красного или желтого шарика, который добавляется по команде *Отладить* (см. ниже).

В контекстном меню окна содержатся команды:

- *Отладить* — автоматическое подключение отладочной информации закрытого интерфейса в *Галактике* для его выполнения в режиме отладки. Функция предназначена для интерфейсов, которые еще не открыты в *Галактике*. С ее помощью можно выполнять интерфейс в режиме отладки без точек останова. По команде *Отладить* в окне «Интерфейсы» для выбранной записи добавится обозначение «желтый шарик». Помеченный таким образом интерфейс автоматически подключит отладочную информацию после его открытия. Когда интерфейс переходит в состояние *выполнения*, он пометится «зеленым» шариком, в режиме *останов* — «красным».
- *Остановить* — отменить автоматическое подключение отладочной информации при отсутствии точек останова, при этом для выбранного интерфейса удаляется отметка «желтый шарик».
- *Остановить все* — отменить автоматическое подключение отладочной информации при отсутствии точек останова, при этом для всех интерфейсов удаляется имеющаяся отметка «желтый шарик».
- *Запуск внешнего интерфейса* — открытие интерфейса в отлаживаемом приложении.
- *Обновить* — получить актуальный список подключенных интерфейсов.
- *Очистить* — удалить список интерфейсов.

В окне имеется возможность поиска с помощью фильтрации по введенным символам. Фильтр распространяется на все столбцы и отменить его можно с помощью клавиши **Delete**, а также автоматически при закрытии данного закрепленного окна.

7.8. Окно загруженных интерфейсов

Окно «Загруженные интерфейсы» содержит список загруженных интерфейсов и принадлежащий им список файлов с исходным кодом, отражает состояние интерфейсов в отладке, а также указывает количество загруженных экземпляров интерфейса. Функции окна позволяют установить под отладку необходимый загруженный интерфейс.

Для открытия окна предназначен пункт главного меню *Отладка > Загруженные интерфейсы* (см. «[Меню Отладка](#)^[51]»).

В верхней части окна содержится информация о загруженных интерфейсах (*Компонент, Интерфейс, Экземпляр, Счетчик ссылок*, отметка о наличии отладочной информации компонента и его локальных переменных, а также отметка видимости интерфейса). Записи формируются в процессе отладки и обновляются автоматически.

В контекстном меню окна содержатся команды:

- *Отладить* — подключение отладочной информации открытого интерфейса в *Галактике* для его выполнения в режиме отладки. По команде *Отладить* в окне «Загруженные интерфейсы» для выбранной записи добавится обозначение «зеленый шарик». Помеченный таким образом интерфейс автоматически подключит отладочную информацию после его открытия. Когда интерфейс переходит в состояние *выполнения*, он пометится «зеленым» шариком, в режиме *останов* — «красным».

- **Остановить** — остановить режим отладки для выбранного интерфейса, при этом удаляется отметка о состоянии отладчика.
- **Остановить все** — остановить режим отладки всех интерфейсов, при этом для всех интерфейсов удаляется отметка о состоянии отладчика.
- **Обновить** — получить актуальный список загруженных интерфейсов.
- **Очистить** — удалить весь список.
- **Очистить историю** — удалить список закрытых интерфейсов.

В нижней части окна *=Загруженные интерфейсы=* содержится перечень файлов, которые принадлежат соответствующему интерфейсу. Список файлов формируется после установки интерфейса под отладку, при этом отладчик совершает поиск необходимого файла с исходным кодом для его открытия. В случае отсутствия файла формируется диалоговое окно (см. "[Настройка отладчика](#)"¹³³).

Состояние отладчика отражается в первой колонке списка интерфейсов и определяется типом значка:

- зеленый шарик — интерфейс в состоянии выполнения;
- красный шарик — интерфейс в состоянии отладки.

Подробное описание режимов отладчика см. в разделе "[Процесс отладки](#)"¹³³.

Для открытия выбранного файла интерфейса предназначена команда *Перейти* в его контекстном меню. Добавить файл в *=Менеджер проекта=* можно способом Drag&Drop ("перетаскивание"). Для выбора нескольких записей необходимо удерживать **Ctrl**.

В окне *=Загруженные интерфейсы=* имеется возможность поиска с помощью фильтрации. Фильтр выполняется при вводе символов в списке окна и распространяется на все столбцы. Отмена фильтрации выполняется по клавише **Delete**, а также автоматически при закрытии данного закрепленного окна.

Загруженные интерфейсы					
Компонент	Интерфейс	Экземпляр	Счетчик ссылок		
	MARKER	0B46031C	1		
C_COMMON	SMH_GALNET	0B444F7C	1		
C_EXTCLASS	ATTRTABLEPICK	0B5C4EDC	1		
F_FPBDUDGET	OBJFPSEMAPHOR	0B5A9C84	1		
Z_STAFF	USEREVENTS	0B446D70	1		
COMPONENT_NAME	INTERFACE_NAME	0C2054AC	1		
Файл	Путь	Дата/Время			
PROGRAM_CODE_FILE.VIP	D:\PROJECTVIPER\VIP\	04.01.2013 13:24:10			

Рис. 106. Окно загруженных интерфейсов


7.9. Окно таблиц в памяти

Окно *=Таблицы в памяти=* предназначено для просмотра содержимого логических таблиц, загруженных в память при выполнении отладки. В окне отображается список таблиц с их данными.

Для открытия окна предназначен пункт главного меню *Отладка > Окно таблиц в памяти* (см. "[Меню Отладка](#)"⁵¹).

Таблицы в памяти					
Таблицы	Префикс	NREC	SSTRING	WDOUBLE	DDATE
C_TUNE::MFILIALS		1	строка 1	1.1000000000000000	01/05/2013
C_TUNE::TUNESSTATISTICS		2	строка 2	1.2000000000000000	02/05/2013
ITMPTABLE=>TMPTABLE	0F582DB8	3	строка 3	1.3000000000000000	03/05/2013

Рис. 107. Окно таблиц в памяти

 Просмотр таблиц в памяти доступен, когда в текущей конфигурации отладчика выбран **Атлантис** не ниже версии **5.5.14.0**.

Окно разделено на две части. В первой — содержится список загруженных таблиц, во второй — записи соответствующей выбранной таблицы. Наполнение окна производится автоматически после открытия интерфейса в отлаживаемом приложении.

Для перемещения по спискам окна предназначены клавиши направлений (**Left/Right/Up/Down**). Сортировка записей осуществляется щелчком левой кнопкой мыши по заголовку столбца. Также в окне существует возможность фильтрации списка по мере ввода текста. Фильтр распространяется на все столбцы. Отмена фильтрации происходит автоматически при обновлении списка, а также с помощью клавиши **Delete**.

По завершении процесса отладки содержимое окна «Таблицы в памяти» очищается.

8. Средства внешнего запуска и исполнения

8.1. Инструменты

Инструмент предназначен для быстрого выполнения пользовательской команды и состоит из определенных параметров.

Типы пользовательских команд:

- вызов внешней программы;
- выполнение команды редактора;
- вставка строки в активный редактор кода.

Для создания инструмента предназначен пункт главного меню *Инструменты* (см. "[Настройка инструмента](#)"¹⁴³).

Инструмент может быть вызван на исполнение одним из способов:

- из списка главного меню *Инструменты*;
- из панели инструментов. Пиктограмма инструмента автоматически добавляется на панель инструментов (см. "[Меню Панели инструментов](#)"⁴⁷);
- с помощью "горячей" клавиши. Комбинация по умолчанию не назначается, но при необходимости ее можно выбрать в настройках клавиатуры (см. "[Настройка инструментальных панелей и горячих клавиш](#)"⁴⁸).


Ограничений на количество используемых инструментов нет.



8.1.1. Настройка инструмента


Для создания инструмента предназначен пункт главного меню *Инструменты > Настройка*. Вызов команды осуществляет открытие диалогового окна *Настройка инструментов*.

Данное окно содержит список инструментов и их параметры. Для каждого элемента списка они индивидуальны.

Для управления списком *Инструментов* предназначена инструментальная панель окна. Она содержит команды копирования, удаления, перемещения вверх/вниз, аналогичные командам для управления подключаемыми ресурсами (см. "[Вкладка Компилятор VIP - Ресурсы](#)"⁹²).

С помощью кнопки  можно создать копию выбранного элемента списка.

Для добавления нового инструмента следует нажать соответствующую пиктограмму  на верхней панели окна настройки инструмента. После ее нажатия в списке *Инструменты* добавляется имя по умолчанию, при необходимости его можно изменить после нажатия соответствующей кнопки . Имя инструмента отображается в названии соответствующего пункта меню и во всплывающей подсказке на панели инструментов.

Пиктограмма — полное имя файла с иконкой, которая отобразится для данного инструмента в главном меню и в панели инструмента. Для выбора файла предназначена кнопка . В качестве параметра необходимо указать файл *.ico.

Тип команды — задача, выполняемая инструментом. Варианты задач отображаются при нажатии стрелки, направленной вниз. Каждая задача имеет свои настраиваемые параметры.

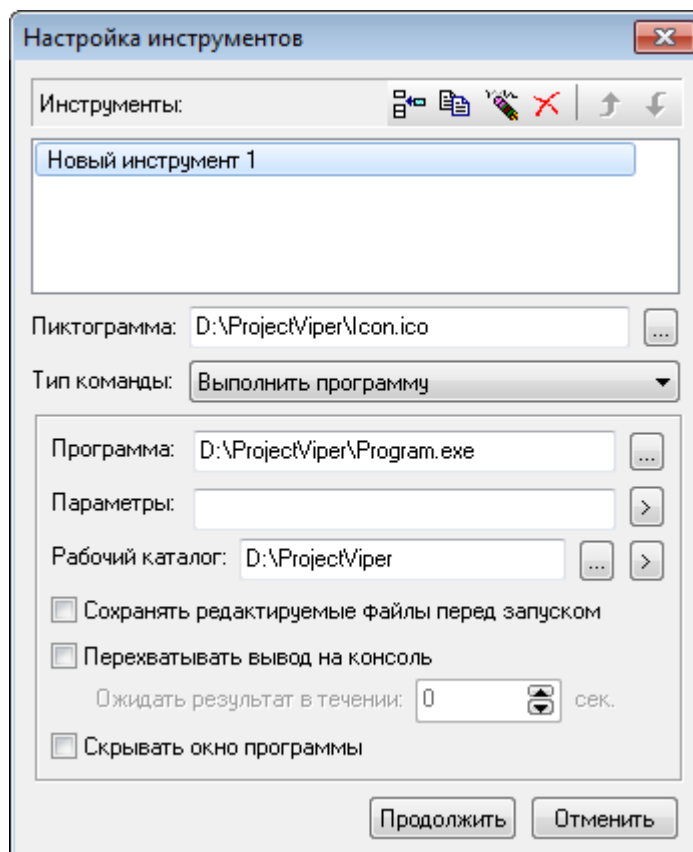



Рис. 108. Настройка инструмента

Настройка некоторых параметров может быть расширена путем использования макропеременных. Список макропеременных раскрывается с помощью кнопки  (Подробнее см. "[Переменные в настройках инструмента](#)"¹⁴⁵).

Параметры команды типа *Выполнить программу*:

- **Программа** — полное имя запускаемого приложения.
- **Параметры** — параметры запуска приложения. В настройке могут быть использованы переменные.
- **Рабочий каталог** — адрес каталога запускаемого приложения. В настройке могут быть использованы переменные.
- **Сохранять редактируемые файлы перед запуском** — сохранение измененных файлов в редакторе перед запуском приложения.
- **Перехватывать вывод на консоль** — отображение результата выполняемой программы на вкладке *Консоль* = *Окна вывода*⁶⁹. Параметр используется для консольных приложений и предоставляет настройку **Ожидать результат в течении**, с помощью которой автоматически формируется запрос о завершении процесса, если время его запуска превысило допустимое.
- **Скрывать окно программы** — скрытие графического пользовательского интерфейса программы.

Команда типа *Вставить строку* добавляет определенный текст в текущую позицию курсора в редакторе. Для выполнения данной команды необходимо заполнить параметр **Вставляемый текст**, при этом возможно использование **[Переменных]**.


Для инструмента типа *Выполнить команду редактора* следует выбрать необходимую **Команду** из существующего списка, который раскрывается при нажатии стрелки, направленной вниз.

[Продолжить] — закрыть окно *=Настройка инструментов=* с сохранением измененных параметров.

[Отменить] — закрыть окно *=Настройка инструментов=* без сохранения изменений.

8.1.2. Переменные в настройках инструмента

Переменные в параметрах инструмента позволяют упростить процесс его настройки.

Список переменных содержится в локальном меню кнопки  соответствующего параметра. Пути на директории хранятся без "/" в конце.

Допустимые переменные в **Параметрах** команды типа *Выполнить программу*:

Пункт меню	Переменная	Пояснение
Активный документ (короткое)	\$(ActiveDocShort]	Короткое имя файла активного документа.
Активный документ	\$(ActiveDocLong]	Полное имя файла активного документа.
Активный документ без расширения (короткое)	\$(ActiveDocShortNoExt]	Короткое имя файла активного документа без расширения.
Активный документ без расширения	\$(ActiveDocLongNoExt]	Полное имя файла активного документа без расширения.
Название активного документа (короткое)	\$(ActiveDocTitleShort]	Короткое имя файла активного документа без пути.
Название активного документа	\$(ActiveDocTitleLong]	Полное имя файла активного документа без пути.
Название активного документа без расширения (короткое)	\$(ActiveDocTitleShortNoExt]	Короткое имя файла активного документа без пути и расширения.
Название активного документа без расширения	\$(ActiveDocTitleLongNoExt]	Полное имя файла активного документа без пути и расширения.
Файловое расширение активного документа	\$(ActiveDocExt]	Расширение имени файла активного документа.
Имя файла проекта (короткое)	\$(ProjectFileShort]	Короткое имя файла текущего проекта.
Имя файла проекта	\$(ProjectFileLong]	Полное имя файла текущего проекта.
Выбрать имя файла (короткое)	\$(FileNameShort:?)	Открытие диалога выбора файла, возвращает короткое имя файла.

Пункт меню	Переменная	Пояснение
Выбрать имя файла (длинное)	\$(FileNameLong:?0]	Открытие диалога выбора файла, возвращает полное имя файла.
Модифицировать файлы	\$(ModFiles]	Список измененных файлов, имена файлов разделяются пробелом.
Путь документа (короткий)	\$(ActiveDocPathShort]	Короткое имя пути на активный документ.
Путь документа	\$(ActiveDocPathLong]	Полное имя пути на активный документ.
Путь программы (короткий)	\$(ProgramPathShort]	Короткое имя пути на программу.
Путь программы	\$(ProgramPathLong]	Полное имя пути на программу.
Путь проекта (короткий)	\$(ProjectPathShort]	Короткое имя пути на текущий проект.
Путь проекта	\$(ProjectPathLong]	Полное имя пути на текущий проект.
Параметры проекта — Имя текущего элемента сборки (короткое)	\$(PrjCurFileNameS]	Короткое имя текущего файла сборки.
Параметры проекта — Имя текущего элемента сборки	\$(PrjCurFileNameL]	Полное имя текущего файла сборки.
Параметры проекта — Путь текущего элемента сборки (короткий)	\$(PrjCurPathS]	Короткое имя пути текущего файла сборки.
Параметры проекта — Путь текущего элемента сборки	\$(PrjCurPathL]	Полное имя пути текущего файла сборки.
Параметры проекта — Путь базы данных (короткий)	\$(PrjDatabasePathS]	Короткий адрес к каталогу базы данных.
Параметры проекта — Путь базы данных	\$(PrjDatabasePathL]	Полный адрес к каталогу базы данных.
Системные папки — Каталог Windows	\$(WinDirShort]	Директория Windows.
Системные папки — Системный каталог	\$(SysDirShort]	Системная директория.
Системные папки — Временный каталог	\$(TempDirShort]	Временная директория.
Текущее слово	\$(CurWord]	Слово в текущей позиции курсора.
Номер строки	\$(LineNumber]	Номер строки.
Номер позиции	\$(ColNumber]	Номер позиции в строке.
Подсказка	\$(Prompt:0]	Подсказка на ввод слова.

Допустимые переменные в параметре **Рабочий каталог** для команды типа *Выполнить программу*:

Пункт меню	Переменная
Путь документа	\$(ActiveDocPathShort)
Путь программы	\$(ProgramPathShort)
Путь проекта	\$(ProjectPathShort)
Каталог Windows	\$(WinDirShort)
Системный каталог	\$(SysDirShort)
Временный каталог	\$(TempDirShort)

Допустимые переменные в параметре **Вставить текст** для команды типа *Вставить строку*:

Пункт меню	Переменная	Пояснения
Выделенный текст	{ \$Sel }	
Дата/время	{ \$DateTime }	
Имя пользователя	{ \$UserName }	
Название файла	{ \$FileTitle }	
Название файла без расширения	{ \$FileTitleNoExt }	
Имя файла	{ \$FileName }	
Путь к файлу	{ \$FilePath }	
Файл	{ \$File:<полное имя файла> }	вставить содержимое указанного файла

8.2. Профили

Профиль — элемент сборки, состоящий из задач с набором параметров, используемых для запуска внешних приложений/компиляторов. Результат выполнения профиля перехватывается и выводится в `=Окно вывода68=`. Профиль состоит из набора программ, выполняемых последовательно. Программа представляет собой набор параметров для вызова внешнего приложения.

В элементы сборки проекта может одновременно входить несколько профилей. При этом один профиль может содержать несколько программ, которые выполняются одна за другой.

Для создания нового профиля предназначена команда **Добавить профиль**, расположенная в контекстном меню проекта или *Пакета* (см. "[Окно менеджера проекта](#)⁵⁸"). Изменить *Параметры* существующего профиля можно с помощью соответствующей команды в его контекстном меню.

Для выполнения профиля предназначен пункт главного меню *Проект > Компилировать файл/Выполнить профиль* или команда контекстного меню профиля *Компилировать/Выполнить* (см. "[Использование компилятора](#)¹³⁰").

Удалить профиль можно соответствующей командой в его контекстном меню.

Ограничений на количество используемых профилей нет.

8.2.1. Настройка профиля

При *Добавлении профиля* с помощью контекстного меню в *=Менеджере проекта=* — осуществляется открытие диалогового окна с параметрами профиля.

Окно содержит список задач с параметрами для данного профиля.

Наименование из параметра **Профиль** отображается в списке элементов сборки *=Окна проекта=*. При необходимости его можно изменить. Для управления списком **Программы** предназначена инструментальная панель окна. Она содержит команды копирования, изменения, удаления, перемещения вверх/вниз, аналогичные командам инструментальной панели в окне *=Настройка инструмента¹⁴³=*.

Чтобы [**Добавить**] новую программу профиля существует кнопка на панели инструментов диалогового окна. При этом в списке **Программы** добавляется новая программа.

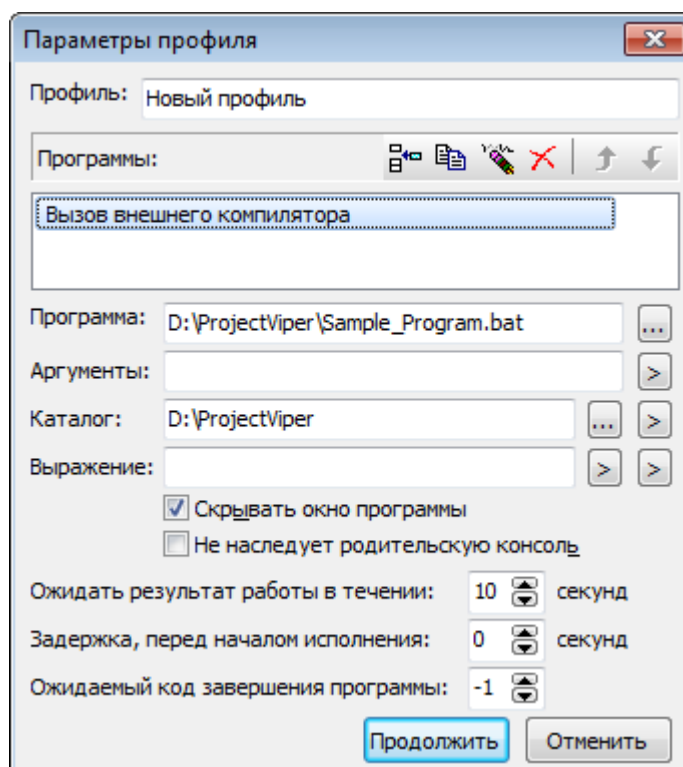




Рис. 109. Окно настройки профиля

Каждый элемент списка программ содержит следующие параметры:

- **Программа** — полное имя вызываемого приложения. Для выбора исполняемого файла предназначена кнопка .
- **Аргументы** — параметры запуска приложения. В настройке могут быть использованы переменные, список переменных раскрывается с помощью кнопки .
- **Каталог** — рабочий каталог вызываемого приложения. В настройке могут быть использованы переменные.
- **Выражение** — переменные и регулярные выражения, предназначенные для разбора строки с результатами выполнения (см. "[Переменные в настройках профиля¹⁴⁹](#)"). Разбор вывода выражения позволяет *Перейти к строке* сообщения из *=Окна вывода⁶⁸=* в окно редактора при использовании имени файла, номера строки и позиции.
- **Скрывать окно программы** — скрытие графического пользовательского интерфейса исполняемого приложения.

- **Не наследует родительскую консоль** — вызываемое приложение не является дочерним процессом.
- **Ожидать результат работы в течении** — время ожидания после последнего вывода, по истечению которого автоматически формируется запрос о завершении процесса. При значении 0 время ожидания не учитывается.
- **Задержка, перед началом исполнения** — время ожидания перед запуском программы, по истечению которого задача профиля выполняется. При значении 0 ожидания не происходит.
- **Ожидаемый код завершения программы** — продолжение работы профиля, если программа имеет указанный код завершения. Код "-1" продолжает профиль в любом случае.

[Продолжить] — закрыть окно параметров профиля с сохранением измененных настроек.

[Отменить] — закрыть окно параметров профиля без сохранения изменений.

Примеры

Пример 1. Разбор логов строчного компилятора для перехвата сообщений =Окном вывода= с возможностью перехода.

```
Программа — cmd
Аргументы — /c find /i "Ошибка" *.log
Каталог — D:\Gal810\src\!Sborka\!logs
Выражение — .*\\(c\т\p\.$[LineNumber].*\п\о\з\.$[ColNumber]
\s\в\s$[FileName]\\)
```

Пример 2. Сборка паскаль *dll* и анализ логов.

Программа 1. Сборка.

```
Программа — D:\Gal\G810_5425\Src\!Sborka\63_Fin.bat
Каталог — D:\Gal\G810_5425\Src\!Sborka\
Скрыть — да
```

Программа 2. Анализ логов.

```
Программа — cmd
Аргументы — /c find /i "Error" *.log
Каталог — D:\Gal\G810_5425\Src\!Sborka\!logs\DLL
Выражение — $[FileName]\($[LineNumber])\)
```

8.2.2. Переменные в настройках профиля

Переменные и регулярные выражения в настройках профиля используются в качестве шаблонов и применяются для выполнения программ, а также для разбора информации в лог-файлах.

Список переменных, допустимых в качестве *Аргументов*, совпадает с переменными в настройках инструмента. А именно, в *Параметрах* для инструмента типа *Выполнить программу* (см. "[Переменные в настройках инструмента](#)"¹⁴⁵).

Список переменных параметра *Каталог*, аналогичен переменным в настройках инструмента. А именно, в параметре *Рабочий каталог* для инструмента типа *Выполнить программу*.

В настройке *Выражение* возможно использование регулярных выражений со следующими переменными:

Пункт меню	Переменная
Имя файла	[\$[FileName]
Номер строки	[\$[LineNumber]
Номер позиции	[\$[ColNumber]

Правила записи регулярных выражений приведены в разделе "[Использование регулярных выражений](#)"¹²⁷.

Примеры

Строка для разбора лог-файла строчного компилятора **Delphi**:

```
$[FileName]\\($[LineNumber]\\)\s(Error|Fatal|Warning).*
```

Строка для разбора лог-файла строчного компилятора **C++ Builder**:

```
Error\s\S*\s$[FileName]\s$[LineNumber]\:.*
```

8.3. Интерпретатор скриптов

Основное предназначение интерпретатора скриптов — это расширение функциональности **Viper** и автоматизация выполнения каких либо действий в редакторе. Скрипты могут быть назначены для выполнения стандартных операций редактора (открытие, закрытие и т. д.) (см. "[Вкладка Общие - Скрипты](#)"¹¹¹). Могут вызываться по запросу пользователя из панели инструментов или из меню [Скрипты](#)⁵³.

Встроенный в **Viper** механизм исполнения скриптов поддерживает языки:

- VBScript;
- JScript.

Перечень объектов, методов и свойств, доступных скриптовой машине, описан в разделе "[API редактора](#)"¹⁵¹.

8.3.1. Текст скрипта

Текст скрипта должен быть сохранен в файле на диске. Следует использовать расширение **.vbs** имени файла для кода на VBScript, и расширение **.js** для кода на JScript. Текст скрипта должен содержать функцию с именем **Main**. Для функции **Main** должен быть описан один параметр **FileName**. Функция **Main** является точкой входа для выполнения скрипта, параметр **FileName** на вход получает имя файла скрипта. Кроме того текст скрипта может содержать любое количество функций, комментариев и прочих конструкций, допустимых в языках VBScript и JScript.

Пример

Простейший скрипт — вывод текста в *=Окно вывода=* редактора **Viper**.

Код на JScript (**Sample.js**)

```
function Main(FileName)
{
    Echo('Hello world!', 'Script file: ' + FileName);
}
```

Код на VBScript (**Sample.vbs**)

```
sub Main(FileName)
    Echo "Hello world!", "Script file: " + FileName
end sub
```

Ссылки

[Руководство по JScript от Microsoft](#) (рус.)

[JScript, Материал из Википедии — свободной энциклопедии](#) (рус.)

[Руководство по VBScript от Microsoft](#) (англ.)

[VBScript, Материал из Википедии — свободной энциклопедии](#) (англ.)

8.3.2. API редактора

Для использования во внешних вызовах доступны объекты:

- ["Главное окно"](#)¹⁵¹;
- ["Активный документ"](#)¹⁵²;
- ["Проект"](#)¹⁶⁰;
- ["Коллекция"](#)¹⁶¹;
- ["Блок"](#)¹⁶²;
- ["Запись"](#)¹⁶³.

8.3.2.1. Объект "Главное окно"

Объект `ViperMainForm` представляет главное окно редактора. Является корневым объектом. К его методам и свойствам можно общаться без указания ссылки на объект либо через переменную `this`.

Свойства

`Path` — Строка, путь и имя файла приложения **Viper**. Доступ только по чтению.

`CommandLine` — Строка, параметры приложения **Viper**. Доступ только по чтению.

`CurDir` — Строка, текущий каталог. Доступно по чтению и на запись.

`ClipBrdText` — Строка, содержимое буфера обмена. Доступно по чтению и на запись.

`ActiveDocument` — содержит ссылку на [Объект "Активный документ"](#)¹⁵², доступ только по чтению.

`Project` — содержит ссылку на [Объект "Проект"](#)¹⁶⁰, доступ только по чтению.

`EventId` — Строка, Id события. При вызове скрипта без события значением является пустая строка. Доступ только по чтению.

Значения событий:

- 'eProgramStart' — начало программы;
- 'eProgramEnd' — конец программы;
- 'eNewDocument' — создание файла;
- 'eActivateDocument' — активирование файла;
- 'eDeactivateDocument' — деактивирование файла;
- 'eOpenDocument' — открытие файла;
- 'eCloseDocument' — закрытие файла;
- 'eSaveDocument' — сохранение файла;
- 'eSaveDocumentAs' — сохранение файла как;
- 'eChangeDocument' — изменение файла;
- 'eExecCodeComplete' — вызов подсказки кода;
- 'eOpenProject' — открытие проекта;
- 'eCloseProject' — закрытие проекта.

Методы

`Echo(Line1 [, Line2, ...])`

Вывод строки `Line1`, `Line2` и т. д. в `=Окно вывода=` редактора.

`ClearOutput`

Очистка `=Окна вывода=` редактора.

`Exec(Path, CommandLine, CurDir, Capture, Hidden, Save)`

Запуск внешней программы или приложения.

Параметры:

`Path` — *Строка*, путь и имя файла приложения;

`CommandLine` — *Строка*, параметры приложения;

`CurDir` — *Строка*, рабочий каталог;

`Capture` — *Логический*, перехват вывода на консоль;

`Hidden` — *Логический*, скрыть окно приложения;

`Save` — *Логический*, сохранить текст в открытых редакторах.

`Close`

Закрытие приложения.

8.3.2.2. Объект "Активный документ"

Объект `ActiveDocument` представляет активное окно редактора. Для доступа к его методам и свойствам необходимо указывать ссылку на данный объект.

Свойства

`FileName` — *Строка*, имя файла. Доступно по чтению и на запись.

`ReadOnly` — *Флаг*, атрибут редактора *Только чтение*. Доступно по чтению и на запись.

`Modified` — *Флаг*, атрибут редактора *Модифицировано*. Доступно по чтению и на запись.

`FilePath` — *Строка*, путь файла. Доступно по чтению и на запись.

`TypeName` — *Строка*, название синтаксической схемы. Доступно по чтению и на запись.

`Text` — *Строка*, текст редактора. Доступно по чтению и на запись.

`Lines` — *Список*, текст редактора построчно. Доступно по чтению и на запись.

`WordAtCaret` — *Строка*, слово в позиции курсора. Доступ только по чтению.

`PaintLocked` — *Флаг*, блокировка прорисовки. Доступно по чтению и на запись.

`SelText` — *Строка*, выделенный текст. При записи замена выделенного текста. Доступно по чтению и на запись.

`BlockBeginX` — *Число*, начальная позиция выделенного блока. Доступно по чтению и на запись.

`BlockBeginY` — *Число*, номер строки начала выделенного блока. Доступно по чтению и на запись.

`BlockEndX` — *Число*, конечная позиция выделенного блока. Доступно по чтению и на запись.

`BlockEndY` — *Число*, номер строки конца выделенного блока. Доступно по чтению и на запись.

`CaretX` — *Число*, позиция каретки. Доступно по чтению и на запись.

`CaretY` — *Число*, номер строки с кареткой. Доступно по чтению и на запись.

`TopLine` — *Число*, номер первой видимой строки. Доступно по чтению и на запись.

`BottomLine` — *Число*, номер последней видимой строки. Доступно по чтению и на запись.

`CurLine` — *Строка*, текст в текущей строке. Доступно по чтению и на запись.

Методы

`Close`

Заккрытие файла.

`Save`

Сохранение файла.

`SaveAs (ShowDialog, FileName)`

Сохранение в новом файле.

Параметры:

`ShowDialog` — *Логический*, отображение диалога выбора файла;

`FileName` — *Строка*, имя файла.

`Activate`

Активация документа.

`Reopen`

Повторное открытие файла.

`ExecCommand (Command)`

Выполнение команды приложения.

`Command` — код команды (см. раздел "[Команды метода ExecCommand](#)¹⁵⁴").

`ExecEditCommand (Command)`

Выполнение команды редактора.

`Command` — код команды (см. раздел "[Команды метода ExecEditCommand](#)¹⁵⁵").

`ClearBookmarks`

Удаление закладок.

`GotoBookmark (Index)`

Переход к закладке.

`Index` — *Число*, номер закладки.

`ToggleBookmark (Index, x, y)`

Переключение закладки.

Параметры:

`Index` — *Число*, номер закладки;

`x` — *Число*, позиция в строке;

`y` — *Число*, номер строки.

`ShowCaret`

Переход к курсору.

`GetWordAt (x, y)`

Получение слова, расположенного в указанной позиции.

Параметры:

`x` — *Число*, позиция в строке;

`y` — *Число*, номер строки.

`GotoXY(x, y)`

Перевод курсора в указанную позицию.

Параметры:

x — Число, позиция в строке;

y — Число, номер строки.

`SelectRange(BX, BY, EX, EY)`

Выделение блока.

Параметры:

BX — Число, начальная позиция блока;

BY — Число, начальная строка блока;

EX — Число, конечная позиция блока;

EY — Число, конечная строка блока.

`HighlightLine(ALine, AColor)`

Выделение строки.

Параметры:

ALine — Число, номер строки;

AColor — цвет.

`AddToCompile`

Добавление файла в элементы сборки.

`Compile`

Компиляция текущего элемента сборки.

8.3.2.2.1. Команды метода *ExecCommand*

Код команды	Значение
scEInsertDateTime	Вставка даты/времени.
scEPaste	Вставка текста.
scEInsertFile	Вставка файла.
scESelectLine	Выделение строк.
scESelectAll	Выделение текста.
scEIndent	Добавление отступа.
scEDuplicateLines	Дублирование строк.
scECapitalize	Заглавная буква в слове.
scETabsToSpace	Замена табуляции пробелами.
scELeadingTabsToSpace	Замена первой табуляции пробелами.
scELeadingSpaceToTabs	Замена первых пробелов табуляцией.
scSReplace	Замена текста.
scEFormatLines	Изменение формата строк.
scSIncrSearchForward	Инкрементальный поиск вперед.
scSIncrSearchBackward	Инкрементальный поиск обратно.
scEComment	Комментирование.
scECopy	Копирование текста.

Код команды	Значение
scEAppenCopy	Копирование текста в конец буфера.
scEUndo	Отмена.
scERedo	Возврат.
scSShowCaret	Переход к курсору.
scSGoto	Переход к позиции.
scSMatchBracket	Переход к парной скобке.
scSGotoLastChange	Переход к последнему изменению.
scSGotoPrevOutput	Переход вперед в окне вывода.
scSGotoNextOutput	Переход назад в окне вывода.
scFReopen	Повторное открытие файла.
scSFind	Поиск.
scSFindNext	Поиск вперед.
scSFindPrev	Поиск назад.
scSFindNextWord	Поиск слова вперед.
scSFindPrevWord	Поиск слова назад.
scEEncoding	Перекодирование.
scEAnsiToOem	Преобразование кодировки текста из Ansi в Oem.
scEOEMToAnsi	Преобразование кодировки текста из Oem в Ansi.
scECheckSpelling	Проверка орфографии.
scEJoinLines	Соединение строк.
scEBlockSort	Сортировка строк.
scFSave	Сохранение файла.
scFSaveAs	Сохранение в новом файле.
scEBlockSave	Сохранение выделенного блока в файл.
scEDelete	Удаление текста.
scEDeleteToBOL	Удаление текста до курсора.
scEDeleteToEOL	Удаление текста после курсора.
scECut	Удаление текста с помещением в буфер.
scEAppendCut	Удаление текста с помещением в конец буфера.
scEDeleteWord	Удаление слова.
scEDeleteLine	Удаление строки.
scEOutdent	Удаление отступа.

8.3.2.2.2. Команды метода *ExecEditCommand*

Идентификатор	Выполняемое действие
ecNone	Пустое действие.
ecLeft	Перемещение курсора на одну позицию влево.
ecRight	Перемещение курсора на одну позицию вправо.

Идентификатор	Выполняемое действие
ecUp	Перемещение курсора на одну строку вверх.
ecDown	Перемещение курсора на одну строку вниз.
ecWordLeft	Перемещение курсора на одно слово влево.
ecWordRight	Перемещение курсора на одно слово вправо.
ecLineStart	Перемещение курсора в начало строки.
ecLineEnd	Перемещение курсора в конец строки.
ecPageUp	Перемещение курсора на одну страницу вверх.
ecPageDown	Перемещение курсора на одну страницу вниз.
ecPageLeft	Перемещение курсора на одну страницу влево.
ecPageRight	Перемещение курсора на одну страницу вправо.
ecPageTop	Перемещение курсора к верхнему краю страницы.
ecPageBottom	Перемещение курсора к нижнему краю страницы.
ecEditorTop	Перемещение курсора в начало текста.
ecEditorBottom	Перемещение курсора в конец текста.
ecGotoXY	Перемещение курсора в указанные координаты, Data = PPoint.
ecLineTextStart	Перемещение курсора к началу текста в строке.
ecSelection	Модификатор для добавления к команде свойства выделения. Текст между позициями курсора до и после выполнения команды будет помечен как выделенный.
ecSelLeft	Перемещение курсора на одну позицию влево с выделением текста.
ecSelRight	Перемещение курсора на одну позицию вправо с выделением текста.
ecSelUp	Перемещение курсора на одну строку вверх с выделением текста.
ecSelDown	Перемещение курсора на одну строку вниз с выделением текста.
ecSelWordLeft	Перемещение курсора на одно слово влево с выделением текста.
ecSelWordRight	Перемещение курсора на одно слово вправо с выделением текста.
ecSelLineStart	Перемещение курсора в начало строки с выделением текста.
ecSelLineTextStart	Перемещение курсора к началу текста строки, с выделением текста.
ecSelLineEnd	Перемещение курсора в конец строки с выделением текста.
ecSelPageUp	Перемещение курсора на одну страницу вверх с выделением текста.
ecSelPageDown	Перемещение курсора на одну страницу вниз с выделением текста.
ecSelPageLeft	Перемещение курсора на одну страницу влево с выделением текста.
ecSelPageRight	Перемещение курсора на одну страницу вправо с выделением текста.

Идентификатор	Выполняемое действие
ecSelPageTop	Перемещение курсора к верхней границе страницы с выделением текста.
ecSelPageBottom	Перемещение курсора к нижней границе страницы с выделением текста.
ecSelEditorTop	Перемещение курсора в начало текста с выделением текста.
ecSelEditorBottom	Перемещение курсора в конец текста с выделением текста.
ecSelGotoXY	Перемещение курсора в указанные координаты с выделением текста, Data = PPoint.
ecSelectAll	Выделение текста с перемещением курсора в конец текста.
ecCopy	Копирование выделенного текста в буфер обмена.
ecScrollUp	Прокрутка на одну линию вверх без изменения позиции курсора.
ecScrollDown	Прокрутка на одну линию вниз без изменения позиции курсора.
ecScrollLeft	Прокрутка на одну позицию влево без изменения позиции курсора.
ecScrollRight	Прокрутка на одну позицию вправо без изменения позиции курсора.
ecInsertMode	Режим вставки.
ecOverwriteMode	Режим замены.
ecToggleMode	Переключение режима вставки/замены.
ecNormalSelect	Обычный режим выделения текста.
ecColumnSelect	Выделение текста вертикальным блоком.
ecColSelLeft	Перемещение курсора на одну позицию влево с выделением вертикального блока.
ecColSelRight	Перемещение курсора на одну позицию вправо с выделением вертикального блока.
ecColSelUp	Перемещение курсора на одну строку вверх с выделением вертикального блока.
ecColSelDown	Перемещение курсора на одну строку вниз с выделением вертикального блока.
ecColSelWordLeft	Перемещение курсора на одно слово влево с выделением вертикального блока.
ecColSelWordRight	Перемещение курсора на одно слово вправо с выделением вертикального блока.
ecColSelLineStart	Перемещение курсора в начало строки с выделением вертикального блока.
ecColSelLineEnd	Перемещение курсора в конец строки с выделением вертикального блока.
ecColSelPageUp	Перемещение курсора на одну страницу вверх с выделением вертикального блока.
ecColSelPageDown	Перемещение курсора на одну страницу вниз с выделением вертикального блока.
ecColSelPageLeft	Перемещение курсора на одну страницу влево с выделением вертикального блока.

Идентификатор	Выполняемое действие
ecColSelPageRight	Перемещение курсора на одну страницу вправо с выделением вертикального блока.
ecColSelPageTop	Перемещение курсора к верхней границе страницы с выделением вертикального блока.
ecColSelPageBottom	Перемещение курсора к нижней границе страницы с выделением вертикального блока.
ecColSelEditorTop	Перемещение курсора в начало текста с выделением вертикального блока.
ecColSelEditorBottom	Перемещение курсора в конец текста с выделением вертикального блока.
ecColSelLineTextStart	Перемещение курсора к началу текста в строке с выделением вертикального блока.
ecLineSelect	Выделение текста по строкам.
ecBlockSetBegin	Установка блока в начало.
ecBlockSetEnd	Установка блока в конец.
ecBlockToggleHide	Скрыть переключатель блока.
ecBlockHide	Скрыть блок.
ecBlockShow	Показать блок.
ecBlockMove	Переместить блок.
ecBlockCopy	Копировать блок.
ecBlockDelete	Удалить блок.
ecBlockGotoBegin	Переход в начало блока.
ecBlockGotoEnd	Переход в конец блока.
ecMatchBracket	Переход к парной скобке.
ecGotoMarker0	Переход к маркеру 0.
ecGotoMarker1	Переход к маркеру 1.
ecGotoMarker2	Переход к маркеру 2.
ecGotoMarker3	Переход к маркеру 3.
ecGotoMarker4	Переход к маркеру 4.
ecGotoMarker5	Переход к маркеру 5.
ecGotoMarker6	Переход к маркеру 6.
ecGotoMarker7	Переход к маркеру 7.
ecGotoMarker8	Переход к маркеру 8.
ecGotoMarker9	Переход к маркеру 9.
ecSetMarker0	Установка маркера 0, Data = PPoint - X, Y Pos.
ecSetMarker1	Установка маркера 1, Data = PPoint - X, Y Pos.
ecSetMarker2	Установка маркера 2, Data = PPoint - X, Y Pos.
ecSetMarker3	Установка маркера 3, Data = PPoint - X, Y Pos.
ecSetMarker4	Установка маркера 4, Data = PPoint - X, Y Pos.
ecSetMarker5	Установка маркера 5, Data = PPoint - X, Y Pos.

Идентификатор	Выполняемое действие
ecSetMarker6	Установка маркера 6, Data = PPoint - X, Y Pos.
ecSetMarker7	Установка маркера 7, Data = PPoint - X, Y Pos.
ecSetMarker8	Установка маркера 8, Data = PPoint - X, Y Pos.
ecSetMarker9	Установка маркера 9, Data = PPoint - X, Y Pos.
ecToggleMarker0	Переключение маркера 0. Иначе установить маркер, Data = PPoint - X, Y Pos. При наличии данного маркера в текущей позиции - удалить его.
ecToggleMarker1	Переключение маркера 1.
ecToggleMarker2	Переключение маркера 2.
ecToggleMarker3	Переключение маркера 3.
ecToggleMarker4	Переключение маркера 4.
ecToggleMarker5	Переключение маркера 5.
ecToggleMarker6	Переключение маркера 6.
ecToggleMarker7	Переключение маркера 7.
ecToggleMarker8	Переключение маркера 8.
ecToggleMarker9	Переключение маркера 9.
EcFoldLevel1	Закрытие каталогов с уровнем вложенности не менее 1.
EcFoldLevel2	Закрытие каталогов с уровнем вложенности не менее 2.
EcFoldLevel3	Закрытие каталогов с уровнем вложенности не менее 3.
EcFoldLevel4	Закрытие каталогов с уровнем вложенности не менее 4.
EcFoldLevel5	Закрытие каталогов с уровнем вложенности не менее 5.
EcFoldLevel6	Закрытие каталогов с уровнем вложенности не менее 6.
EcFoldLevel7	Закрытие каталогов с уровнем вложенности не менее 7.
EcFoldLevel8	Закрытие каталогов с уровнем вложенности не менее 8.
EcFoldLevel9	Закрытие каталогов с уровнем вложенности не менее 9.
EcFoldLevel0	Закрытие каталогов с уровнем вложенности не менее 10.
EcFoldCurrent	Закрытие текущей папки.
EcUnFoldCurrent	Раскрытие текущей папки.
ecContextHelp	Вызов контекстной справки, Data = Word.
ecViewCommandLast	Просмотр последней команды.
ecDeleteLastChar	Удаление последнего символа (backspace).
ecEditCommandFirst	Редактирование первой команды.
ecDeleteChar	Удаление символа под курсором (delete).
ecDeleteWord	Удаление символов от курсора до конца слова.
ecDeleteLastWord	Удаление символов от начала слова до курсора.
ecDeleteBOL	Удаление символов от начала строки до курсора.
ecDeleteEOL	Удаление символов от курсора до конца строки.
ecDeleteLine	Удаление текущей строки.
ecClearAll	Очистка.

Идентификатор	Выполняемое действие
ecLineBreak	Разрыв строки в текущей позиции с перемещением курсора на новую строку.
ecInsertLine	Разрыв строки в текущей позиции без перемещения курсора.
ecChar	Вставка символа в текущей позиции.
ecSmartUnindent	Улучшенное удаление табуляции
ecUndo	Выполнение функции "Отменить", если возможно.
ecRedo	Выполнение функции "Повторить", если возможно.
ecCut	Удаление выбранного текста с помещением его в буфер обмена.
ecPaste	Вставка текста из буфера обмена в текущую позицию.
ecBlockIndent	Выравнивание текста.
ecBlockUnindent	Отмена выполненного выравнивания текста.
ecTab	Добавление отступа.
ecShiftTab	Удаление отступа.
ecUpperCase	Преобразование текста к верхнему регистру.
ecLowerCase	Преобразование текста к нижнему регистру.
ecToggleCase	Инвертирование регистра текста.
ecTitleCase	Преобразование первой буквы в заглавную.
ecUpperCaseBlock	Преобразование выбранного текста или текущего символа к верхнему регистру.
ecLowerCaseBlock	Преобразование выбранного текста или текущего символа к нижнему регистру.
ecToggleCaseBlock	Инвертирование регистра выбранного текста или текущего символа.
ecString	Вставка целой строки.
ecAutoCompletion	Выполнение функции "Автозаполнение".
ecEditorFocus	Установка фокуса в активный редактор кода.
ecGotFocus	Получение фокуса.
ecLostFocus	Забытие фокуса.
ecMouseDown	Нажатие мыши.
ecEditCommandLast	Редактирование последней команды.
ecUserFirst	Начало определяемых пользователем команд.
ecPluginFirst	Плагин первых.

8.3.2.3. Объект "Проект"

Объект `Project` представляет текущий проект. Для доступа к его методам и свойствам необходимо указывать ссылку на данный объект.

Свойство

`IsLoad` — Флаг, наличие загруженного проекта. Доступ только по чтению.

Методы

`New (ShowDialog, FileName)`

Создание нового проекта.

Параметры:

`ShowDialog` — *Логический*, отображение диалога выбора файла проекта;

`FileName` — *Строка*, имя файла проекта.

`Open (ShowDialog, FileName)`

Открытие проекта.

Параметры:

`ShowDialog` — *Логический*, отображение диалога выбора файла проекта;

`FileName` — *Строка*, имя файла проекта.

`Close`

Закрытие проекта.

8.3.2.4. Объект "Коллекция"

Объект `CodeCompleteUser` обеспечивает доступ и хранение записей пользовательских подсказок — блоков.

Методы

`Add (AName: String)`

Предназначен для создания нового блока. На входе задается имя нового блока (имена блоков уникальны), если имя не указано, либо блок с таким именем существует, то имя будет сгенерировано автоматически по шаблону `CodeCompleteBlock_<номер>`. На выходе метод возвращает ссылку на новый объект блока.

`AName` — *Строка*, имя блока.

`Delete (AIndex: integer)`

Удаление блока из коллекции, на входе задается индекс блока.

`AIndex` — *Число*, индекс блока.

`Items (AIndex: integer)`

Получение ссылки на объект блока.

На входе — индекс блока, на выходе — ссылка.

`AIndex` — *Число*, индекс блока.

`IndexOf (AValue: TCodeCompleteBlock)`

Получение индекса блока. На входе — ссылка на блок, на выходе — индекс.

`AValue` — *Число*, ссылка на блок.

`Count`

Количество блоков.

На выходе — количество блоков.

`Find (AName: AnsiString)`

Поиск блока по имени.

На вход — имя блока. На выходе — ссылка на блок.

`AName` — *Строка*, имя блока.

`Clear`

Удаление всех блоков.

8.3.2.5. Объект "Блок"

Объект `CodeCompleteBlock` представляет собой коллекцию записей пользовательской подсказки.

Свойства

`Name` — *Строка*, имя блока. Доступно по чтению и на запись. При попытке задания нового имени блоку будет произведена проверка на уникальность, в противном случае имя останется прежним.

`FileLang` — *Строка*, имя используемого языка. Доступно по чтению и на запись. В параметр может быть присвоено несколько языков, для этого необходимо проинициализировать это свойство для каждого требуемого языка в отдельности.

В качестве ограничения по языку можно указывать полное имя языка, как в [синтаксической схеме](#)¹²² раскрыва, либо использовать псевдоним языка:

- `'Pascal'`;
- `'C++'`;
- `'VBScript'`;
- `'Bat'`;
- `'JScript'`;
- `'XML'`;
- `'Cfg'`;
- `'VIP'`;
- `'Pat'`.

`FileName` — *Строка*, имя файла. Обозначает конкретный файл, в котором возможно использование записи. Доступно по чтению и на запись.

`FileExt` — *Строка*, расширения файлов. Обозначает типы файлов, в которых возможно использование записи. Доступно по чтению и на запись.

Методы

`Add(AName, AType, AHint, AInsert)`

Предназначен для создания новой записи. На выходе метод возвращает ссылку на новую запись.

Параметры:

`AName` — *Строка*, имя идентификатора. Обязательный параметр;

`AType` — *Строка*, тип идентификатора;

`AHint` — *Строка*, подсказка;

`AInsert` — *Строка*, дополнительно вставляемый текст.

`Delete(AIndex)`

Удаление записи из коллекции.

`AIndex` — *Число*, индекс записи.

`Items(AIndex)`

Получение ссылки на объект записи. На входе — индекс записи, на выходе — ссылка.

`AIndex` — *Число*, индекс записи.

`IndexOf(AValue: TCodeCompleteItem)`

Получение индекса записи. На входе — ссылка на запись, на выходе — индекс.

`AValue` — Число, ссылка на запись.

`Count`

Количество записей. На выходе — количество записей.

`Clear`

Удаление всех записей.

8.3.2.6. Объект "Запись"

Объект `CodeCompleteItem` представляет собой запись подсказки и может содержать вложенные записи.

Свойства

`Name` — Строка, имя/текст идентификатора. Доступно по чтению и на запись.

`TypeName` — Строка, тип идентификатора. Доступно по чтению и на запись.

В качестве типа идентификатора может применяться любое наименование, а для следующих константных типов существует соответствующая иконка:

- `'Identifier'`;
- `'Object'`;
- `'Property'`;
- `'Key word'`;
- `'Procedure'`;
- `'Function'`;
- `'Table'`;
- `'Field'`;
- `'Browse'`;
- `'CmEvent'`;
- `'CreateViewStatement'`;
- `'DataStream'`;
- `'DataStreamTable'`;
- `'Dialog'`;
- `'Embedded'`;
- `'HandleEvent'`;
- `'Interface'`;
- `'InterfaceParameters'`;
- `'MemoEdit'`;
- `'Menu'`;
- `'MenuItem'`;
- `'MenuPopup'`;
- `'Panel'`;
- `'ProcFunc'`;
- `'Screen'`;
- `'TabbedSheet'`;

- *'TableEvent'*;
- *'TableStruct'*;
- *'ToolBar'*;
- *'Tree'*;
- *'VipInterface'*;
- *'Window'*;
- *'WindowEvent'*.

Hint — *Строка*, дополнительная информация по идентификатору, отображаемая в выпадающем списке подсказки (в редактор не добавляется).

Insert — *Строка*, дополнительно вставляемый текст после идентификатора (может содержать символ установки каретки "|").

Методы

Add(AName, AType, AHint, AInsert)

Предназначен для создания вложенной записи. На выходе метод возвращает ссылку на новую запись.

Параметры:

AName — *Строка*, имя идентификатора. Обязательный параметр;

AType — *Строка*, тип идентификатора;

AHint — *Строка*, подсказка;

AInsert — *Строка*, дополнительно вставляемый текст.

Delete(AIndex)

Удаление вложенной записи из коллекции, на входе задается индекс записи.

AIndex — *Число*, индекс записи.

Items(AIndex)

Получение ссылки на вложенную объект записи. На входе — индекс записи, на выходе — ссылка.

AIndex — *Число*, индекс записи.

IndexOf(AValue: TCodeCompleteItem)

Получение индекса вложенной записи. На входе — ссылка на запись, на выходе — индекс.

AValue — *Число*, ссылка на запись.

Count

Количество вложенных записей. На выходе — количество записей.

Clear

Удаление всех вложенных записей.

8.3.3. Пользовательские интерфейсы

Для создания GUI-интерфейсов, во время исполнения скрипта, доступны компоненты из палитры Delphi. Свойства данных компонент можно использовать по аналогии с Delphi, доступные методы описаны ниже.

GUI-компоненты и методы:

- TForm;
ShowModal

Открытие окна в модальном режиме.

- TColorDialog;

[Execute](#)

Запуск диалога.

- TFontDialog;

[Execute](#)

Запуск диалога.

- TSaveDialog;

[Execute](#)

Запуск диалога.

- TOpenDialog;

[Execute](#)

Запуск диалога.

- TStatusBar;

- TDateTimePicker;

- TUpDown;

- TTrackBar;

- TPageControl;

- TTabControl;

- THotKey;

- TGroupBox;

- TComboBox;

- TRadioButton;

- TCheckBox;

- TMemo;

- TEdit;

- TListBox;

- TPanel;

- TButton;

[Click](#)

Нажатие кнопки.

- TLabel;

- TRadioGroup;

- TImage;

- TCheckListBox;

- TSpinEdit.

8.3.4. Примеры использования скриптов

Примеры следующих скриптов можно найти в папке **Samples\Scripts** дистрибутива **Viper**.

Функции выполнения скриптов см. в разделе "[Меню Скрипты](#)⁵³".

Пример1

Функции создания окна сообщения на JScript (*.js)

```
function MessageBox(cap, prompt)
{
    var dlg = Create('TForm', Self);
    with (dlg)
    {
        Caption = cap;
        Position = 'poOwnerFormCenter';
        BorderStyle = "bsDialog";
        Height = 110;
        Width = 220;
    }

    with (Create('TLabel', dlg))
    {
        Parent = dlg;
        AutoSize = false;
        WordWrap = true;
        Caption = prompt;
        Left = 10;
        Top = 10;
        Width = 200;
        Height = 60;
    }

    with (Create('TButton', dlg))
    {
        Parent = dlg;
        Caption = 'OK';
        ModalResult = 1;
        Left = 70;
        Top = 50;
        Default = true;
    }
    dlg.ShowModal;
}

function Main()
{
    MessageBox('Hello', 'World')
}
```

Пример 2

Скрипт для быстрой компиляции компонента на VBScript (*.vbs)

```
' Проверка существования файла
function FileExists(FileName)
    dim objFs
    set objFs = CreateObject("Scripting.FileSystemObject")
    FileExists = objFs.FileExists(FileName)
end function

' Проверка существования папки
function DirExists(DirName)
    dim objFs
    set objFs = CreateObject("Scripting.FileSystemObject")
    DirExists = objFs.FolderExists(DirName)
end function

' Добавление backslash
function AddBackslash(DirName)
    if Right(DirName, 1) <> "\" then
        AddBackslash = DirName & "\"
    end if
end function
```



```

else
    AddBackslash = DirName
end if
end function

sub ShowMessage(Text)
    MsgBox Text, vbCritical, "Ошибка скрипта"
end sub

' Точка входа скрипта
sub Main(FileName)
    ' Расположение исходников
    const sPathCompSrc = "D:\Gal\Src\CompSrc"

    if not DirExists(sPathCompSrc) then
        ShowMessage("Папка с исходниками не найдена")
        exit sub
    end if

    dim sFile
    sFile = InputBox("Введите имя компоненты",
        "Компиляция компоненты", "")

    if Trim(sFile) = "" then
        ShowMessage("Неверный ввод")
        exit sub
    end if

    dim sPrj
    sPrj = AddBackslash(sPathCompSrc) & Mid(sFile, 1, 1) &
        "\" & sFile & "\" & sFile & ".prj"

    if not FileExists(sPrj) then
        ShowMessage("Не найден файл:" & vbCrLf & sPrj)
        exit sub
    end if

    Documents.Open false, sPrj, false 'открытие файла
    ActiveDocument.AddToCompile       'добавление в сборку
    ActiveDocument.Compile             'запуск компиляции
end sub

```

Пример 3

Скрипт для вставки из буфера преобразованного текста на VBScript (*.vbs)

```

sub Main(FileName)
    Dim s
    s = ClipBrdText
    ActiveDocument.SelText = "message(" + s + ");"
end sub

```

Пример 4

Скрипт для наполнения списка [подсказки кода](#)⁷⁴ вариантами (которые отображаются в подсказке после ввода Identifier. и Table.) на VBScript (*.vbs)

```

sub Main(FileName)
    CodeCompleteBlock = CodeCompleteUser.find("PointComplete")
    if not IsNull(CodeCompleteBlock) then
        index = CodeCompleteUser.indexOf(CodeCompleteBlock)
        CodeCompleteUser.Delete (index)
    end if

    dim sText
    sText = ActiveDocument.Lines(ActiveDocument.CaretY - 1)

```

```

with CodeCompleteUser.Add("PointComplete")
  .FileLang = "VIP"

  if Right(sText, 1) = "." then
    if Right(sText, 2) = "Identifier." then
      with .Add ("Identifier")
        .Add ("Property1")
        .Add ("Property2")
        .Add ("Property3")
      end with
    else
      if Right(sText, 2) = "Table." then
        with .Add ("Table")
          .Add ("Field1")
          .Add ("Field2")
          .Add ("Field3")
        end with
      end if
    end if
  end if
end with
end sub

```

Пример 5

Скрипт формирования подсказки по таблице и ее полям для языка **VIP** на JScript (*.js)

```

function Main()
{
  with (CodeCompleteUser.Add("Dict91"))
  {
    FileLang = "VIP";
    with (Add ("AktSver", "Table", "Акты сверки по договору"))
    {
      Add ("cDogovor", "Field", "tNRec - Ссылка на договор");
      Add ("cNote", "Field", "tNRec - Ссылка на пояснение по статусу");
      Add ("cVal", "Field", "tNRec - Валюта расч <копия из Dogovor
cValRas>");
      Add ("dBeg", "Field", "Date - Дата начала проведения взаимозач");
      Add ("dDoc", "Field", "Date - Дата документа");
      Add ("dEnd", "Field", "Date - Дата окончания проведения взаимозач");
      Add ("Descr", "Field", "tDescr - Дескриптор менеджера");
      Add ("DesGr", "Field", "tDesGr - Дескриптор группы менеджеров");
      Add ("dInput", "Field", "Date - Дата ввода документа");
      Add ("NoDoc", "Field", "tDocNumber - Номер документа");
      Add ("NoDoc_Ext", "Field", "tDocNumber - Номер документа внешний");
      Add ("NRec", "Field", "tNRec - Номер записи");
      Add ("SaldoIn", "Field", "tSumma - Входящее сальдо взаимозачетов");
      Add ("SaldoOut", "Field", "tSumma - Исходящее сальдо взаимозачетов");
      Add ("Status", "Field", "Word - Статус документа");
      Add ("SumD01", "Field", "tSumma - Сумма выставленных счетов
контр 1 за период");
      Add ("SumD02", "Field", "tSumma - Сумма выставленных счетов
контр 2 за период");
      Add ("SumOtgr1", "Field", "tSumma - Сумма отгрузок контр 1 за
период");
      Add ("SumOtgr2", "Field", "tSumma - Сумма отгрузок контр 2 за
период");
      Add ("SumPlat1", "Field", "tSumma - Сумма платежей контр 1 за
период");
      Add ("SumPlat2", "Field", "tSumma - Сумма платежей контр 2 за
период");
      Add ("TiDk", "Field", "Word - Тип документа");
    }
  }
}

```