Федеральное агентство по образованию

Государственное образовательное учреждение высшего профессионального образования "Санкт-Петербургский государственный политехнический университет"

КОМПАНИЯ "Schneider Electric"

В.Г. Давыдов

Система супервизорного управления Vijeo Citect 7.30 SP1. Базовый курс

Учебное пособие

Санкт-Петербург
2013

Содержание

Предисловие

Учебное пособие обеспечивает курс *"Вычислительные системы"*. Курс предназначен для подготовки магистров по направлению 230100 "Информатика и вычислительная техника" (профиль "Встраиваемые системы управления"). В качестве средства проектирования компьютерных систем управления в учебном пособии используется система супервизорного управления Vijeo Citect 7.30 SP1 (фирмы Schneider Electric, Франция и Citect, Австралия) и ее инструментальный язык программирования Cicode. Изложение материала в учебном пособии соответствует программе базового курса по системе супервизорного управления Vijeo Citect (модуль AUT20) и языку программирования Cicode (модуль AUT24), изучаемого в центрах обучения компании Schneider Electric. Поэтому пособие может быть использовано не только студентами старших курсов в рамках магистерской подготовки, но и инженерным персоналом, проходящим техническое обучение в центрах компании Schneider Electric.

Для удобства пользователей учебное пособие содержит более 90 упражнений, выполнение которых способствует более успешному освоению изучаемого материала, и более 30 сквозных демонстрационных проекта по основным разделам курса, которые служат удобным средством получения справочного материала. Это позволяет использовать учебное пособие и для *самостоятельного* изучения материала. Прилагаемый материал содержит установочные файлы для инсталляции SCADA-системы Vijeo Citect v7.30 SP1, демонстрационные проекты и др. (приложение 3).

Используемые обозначения

Исходные тексты программ и результаты их работы, приводимые в учебном пособии, для удобства читателей печатаются с использованием моноширинного шрифта `Courier New`. Названия окон, полей окон, меню, команд, акселераторов, клавиш, кнопок и т. п. в тексте книги выделяются **полужирным шрифтом**.

Курсивом в тексте выделяются определяющие вхождения новых понятий, а также отдельные слова или выражения, на которые следует обратить внимание.

Имена папок, файлов и их расширения пишутся без кавычек и с выделением **полужирным шрифтом**.

Кроме шрифтовых выделений, используется три типа специальных абзацев: советы, замечания и примечания.

> ***Совет***
> Наряду с данным учебным пособием пользуйтесь и другими общедоступными документами, указанными в списке использованной литературы [1-5]. В частности, прежде чем продолжить работу самостоятельно изучите вводный материал ([2], темы справки Help Overview, What's New in v7.x, Upgrading to Vijeo Citect v 7.10; [3], слайды 1, 2, 7 — 26).

Замечание

При использовании Process Analyst, Vijeo Citect Web Client или Vijeo Citect Web Server следует пользоваться обозревателем Internet Explorer версии 6.0 или выше.

Примечание

При использовании Process Analyst рекомендуется применение видеокарты с объемом собственной видеоОЗУ не менее 128 Мбайт.

Ваши отзывы об учебном пособии, конструктивные замечания и критику направляйте по адресу: **davydov@aivt.ftk.spbstu.ru**.

Система супервизорного управления (SCADA-система) Vijeo Citect 7.30 SP1. Базовый курс (модуль AUT20)

Введение

Цель данного учебного пособия — предметно продемонстрировать основные возможности SCADA-системы **Vijeo Citect v7.30 SP1** путем изучения ее основных возможностей, что иллюстрируется цепочкой сквозных постепенно наращиваемых демонстрационных примеров. Эти примеры в виде архивных файлов содержатся в папке **..\Демонстационные примеры и сопутствующие файлы**, а файлы называются **OvenTraining*.ctz**. Для более быстрого и простого освоения изучаемого материала рекомендуем в процессе упражнений параллельно с преподавателем повторять учебные примеры. При этом примеры **OvenTraining*** можно использовать в качестве справочного материала. В качестве названия повторяемого проекта используйте **Training***, а его резервную копию называйте и размещайте в папке **..\Training\Training*.ctz**.

> ***Совет***
> Наряду с данным учебным пособием пользуйтесь и другими общедоступными документами, указанными в списке использованной литературы [1-5]. В частности, прежде чем продолжить работу самостоятельно изучите вводный материал ([2], темы справки Getting Started; [3], слайды 1, 2, 7 — 26).

Систему супервизорного управления и сбора данных Vijeo Citect можно настроить на работу с любым промышленным предприятием [5, 6]. Поскольку при разработке Vijeo Citect большое внимание уделялось гибкости, то она позволяет построить систему, удовлетворяющую конкретным требованиям. Система супервизорного управления Vijeo Citect пригодна для решения как малых, так и больших задач. Vijeo Citect можно использовать для наблюдения за производством и оборудованием и для управления в различных отраслях промышленности, в энергетике, при обработке и транспортировке нефти и газа, в жилищно-коммунальной сфере и др. Доля рынка Vijeo Citect составляет более 60% в Австралии и более 11% в мире. По всему миру продано свыше 160000 лицензий. Первая версия SCADA-системы Citect была разработана в 1987 году. Первая версия Vijeo Citect вышла в 1992 году. В 2006 году появилась версия 6.10, в 2007 году — версия 7.00, в 2009 году — версия 7.10, в сентябре 2010 года — версия 7.20, а в декабре 2012 – версия 7.30. Сильными сторонами системы супервизорного управления Vijeo Citect являются поддержка контроллеров практически всех фирм-производителей, присутствующих на мировом рынке, мощная среда разработки, относительная простота разработки приложений разной сложности, интеграция в коммерческое предложение компании Schneider Electric и т. п. Благодаря гибкости, систему Vijeo Citect можно расширять в соответствии с технологическими и информационными требованиями производства. Система Vijeo Citect легка в обучении и использовании. Такие функции, как шаблоны, genies (джины), мастера, включаемые проекты и др. сокращают время и усилия, необходимые для настройки и повышения эффективности системы Vijeo Citect.

Совет

Подробнее о сказанном см. в файле **Новые описания и презентации \ Vijeo Citect \ Presentations \ Vijeo Citect Overview.pdf**. Настоятельно рекомендуем посмотреть этот файл

Перед установкой версии 7.30 системы Vijeo Citect убедитесь, что компьютер удовлетворяет *минимальным аппаратным и программным требованиям*, предъявляемым со стороны системы Vijeo Citect. При этом может понадобиться обновление компьютерного оборудования.

Версия 7.30 системы Vijeo Citect может работать в следующих программных средах (ОС):

- Windows 8 - 32 Bit (в настоящее время пока не поддерживается);
- Windows 8 - 64 Bit (в настоящее время пока не поддерживается);
- Windows Server 2012 (в настоящее время пока не поддерживается);
- Windows 7 - 32 Bit;
- Windows 7 - 64 Bit;
- Windows Server® 2008 R2;
- Windows Server 2008 - 32 Bit;
- Windows Server 2008 - 64 Bit;
- Windows Vista® - 32 Bit;
- Windows Vista® - 64 Bit;
- Windows Server 2003 - 32 Bit;
- Windows Server 2003 - 64 Bit;
- Windows Server 2003 R2 - 32 Bit;
- Windows Server 2003 R2 - 64 Bit;
- Windows XP - 32 Bit;
- Windows XP - 64 Bit.

Совет

Подробнее о сказанном см. в файле **Новые описания и презентации \ Vijeo Citect \ Vijeo Compatibility and Interoperability Matrix (Ru).pdf**. Настоятельно рекомендуем посмотреть этот файл

Версия 7.30 Vijeo Citect может работать при следующих минимальных параметрах аппаратуры:

- Windows XP: процессор Pentium 2GHz с 1GB ОЗУ;
- Windows Server 2003: процессор Pentium 3GHz с 2GB ОЗУ;
- Windows Vista, Windows 7 or Windows Server 2008: процессор Pentium 3.2GHz Dual Core с 3GB ОЗУ;
- 5GB свободного пространства на жестком диске;

- Графическая карта по крайней мере с 128MB VRAM;
- VRAM не должна разделяться главной памятью.

После установки системы Vijeo Citect необходимо выполнить *лицензирование*. Лицензия системы Vijeo Citect запрограммирована в устройстве ***Ключ аппаратной защиты*** (ключ защиты), которое можно подключить, например, к USB. Если проект системы Vijeo Citect выполняется без ключа защиты, то он может работать только в *демонстрационном режиме*. Демонстрационный режим позволит нормально использовать все возможности системы Vijeo Citect, в том числе в мультипроцессорном режиме, но с ограниченным временем работоспособного состояния и ограниченным количеством входов/выходов. *В демонстрационном режиме не поддерживаются сетевые возможности Vijeo Citect.*

Доступны следующие демонстрационные режимы:

- Непрерывная работа в течение 15 минут с максимальным количеством 50 000 реальных входов выходов.
- Непрерывная работа в течение 10 часов максимально с одним динамическим вводом/выводом реального времени. Это подходит для демонстрационного режима с использованием дискового ввода/вывода. Система Vijeo Citect начнет работу с этого режима, если отсутствует ключ защиты.
- Непрерывная работа в течение 8 часов с максимальным количеством 42 000 реальных вводов / выводов. **Эта возможность доступна только через системных интеграторов Vijeo Citect. В России порядка 150, а в мире боде 2000 системных интеграторов используют Vijeo Citect в своих решениях.**

Демонстрационный режим полезен для конфигурирования, выполняемого вне реальной системы, и тестирования проекта. Тем не менее, если Vijeo Citect будет использоваться в реальной системе, то понадобится лицензия.

Ключи защиты системы Vijeo Citect должны обновляться при *критических обновлениях программного обеспечения*. Критическое обновление обозначается приростом первой цифры после десятичной точки в номере версии. Например, если вы выполняете обновление с версии V7.1 до версии V7.2, то нужно будет обновить ключ защиты. Обновление же версии V7.10 до версии V7.11 или обновление до сервисного пакета не требует обновления ключа защиты. Вы можете получить или обновить лицензию системы Vijeo Citect с помощью факса, телефона, электронной почты или интерактивно через Интернет. Чтобы иметь возможность обновить ключ защиты, нужна программа **CiUSAFE.exe**, инсталлированная на том же компьютере (находится в папке ..\Program Files\ Schneider Electric\ Vijeo Citect 7.30\Bin или может быть загружена с сайта www.citect.com). Как правило, доступ к ней можно получить из среды приложения **Проводник Citect** с помощью команды **Справка | Диспетчер лицензий**.

Если вы выберете пересылку информации с помощью электронной почты, факса или телефона, то нужно будет выслать или сообщить серийный номер и идентификатор, имеющиеся на наклейке ключа.

Примечание

Обновление ключа защиты должно быть выполнено как операция "проталкивание". Это означает, что вы должны инсталлировать версию 7.30 системы Vijeo Citect на компьютер, а затем выполнить обновление ключа, используя последнюю версию программы CiUSAFE.exe, поставляемую с версией 7.30.

Если вы имеете доступ к Интернету, то можете использовать **Online Authorisation Code Generator** (интерактивный генератор кода авторизации), который можно найти в разделе **Customer Service Area** сайта Citect.

Совет

Подробнее о сказанном см. в файле **Новые описания и презентации \ Vijeo Citect \ Catalog \ 36400-ru (web).pdf**. Настоятельно рекомендуем посмотреть этот файл

Глава 1. Среда конфигурирования Vijeo Citect

Среда конфигурирования Vijeo Citect состоит из четырех отдельных программ. Это модули (интегрированные среды разработки) **Проводник Citect**, **Редактор проектов Citect**, **Построитель графики Citect** и **Редактор Cicode**.

1.1. Проводник Citect

Приложение *Проводник Citect* является проводником проектов и позволяет создавать проекты Vijeo Citect и управлять ими. Проводник служит также управляющим конфигурационным приложением, из которого можно запустить **Редактор проектов Citect**, **Построитель графики Citect** и **Редактор Cicode**. Для запуска **Проводника Citect** выполните команду **Start(Пуск) | Programs (Программы) | Schneider Electric | Vijeo Citect 7.30 | Vijeo Citect Проводник**, но удобнее это делать с помощью ярлыка, расположенного на рабочем столе. При запуске приложения **Проводник Citect** автоматически запускаются приложения **Редактор проектов Citect** и **Построитель графики Citect**. При закрытии **Проводника Citect** указанные приложения автоматически закрываются.

> ***Совет***
> Для получения дополнительной информации о приложении **Проводник Citect** в среде этого приложения выполните команду **Справка | Справка по проводнику**.

Упражнение 1.1

Запустите приложение **Проводник Citect**, изучите имеющиеся команды меню и кнопки на панели инструментов (снабжены всплывающими подсказками). Посмотрите справку о приложении **Проводник Citect**.

1.2. Редактор проектов Citect

Приложение *Редактор проектов Citect* применяется для создания баз данных Vijeo Citect и управления ими. Базы данных содержат информацию о конфигурации проекта Vijeo Citect, которая не относится к графическим страницам. С помощью приложения **Редактор проектов Citect** можно просматривать все записи базы данных проекта Vijeo Citect. Приложение **Редактор проектов Citect** запускается одновременно с приложением **Проводник Citect**. В среду **Редактора проектов Citect** можно перейти из сред **Проводника Citect** и **Построителя графики Citect** с помощью кнопки **Редактор проектов** на панели инструментов или с помощью команды **Средства | Редактор проектов**.

Совет
Для получения дополнительной информации о приложении **Редактор проектов Citect** в среде этого приложения выполните команду **Помощь | Справка по Редактору проектов**.

Упражнение 1.2

Перейдите в среду приложения **Редактор проектов Citect**, изучите имеющиеся команды меню и кнопки на панели инструментов (снабжены всплывающими подсказками). Посмотрите справку о приложении **Редактор проектов Citect**.

1.3. Построитель графики Citect

Приложение *Построитель графики Citect* служит для создания и редактирования графических страниц, в т. ч. объектов, входящих в состав графических страниц. Приложение **Построитель графики Citect** запускается одновременно с приложением **Проводник Citect**. В среду **Построителя графики Citect** можно перейти из сред **Проводника Citect** и **Редактора проектов Citect** с помощью кнопки **Построитель графики** на панели инструментов или с помощью команды **Средства | Построитель Графики**.

Совет
Для получения дополнительной информации о приложении **Построитель графики Citect** в среде этого приложения выполните команду **Спровка | Справка по построителю графики**.

Упражнение 1.3

Перейдите в среду приложения **Построитель графики Citect**, изучите имеющиеся команды меню и кнопки на панели инструментов (снабжены всплывающими подсказками). Посмотрите справку о приложении **Построитель графики Citect**.

1.4. Редактор Cicode

Приложение *Редактор Cicode* применяется для написания и редактирования программ, написанных на языке Cicode, и представляет собой интегрированную среду разработки (ИСР). Это приложение можно также использовать в качестве отладчика на этапе выполнения для трассировки Cicode-программы и навигации ошибок программирования. В среде **Редактора Cicode** можно получить справку по любой Cicode-функции. Для этого следует щелкнуть правой кнопкой мыши по имени функции и выполнить команду **Справка** контекстного меню. В отличие от других, ранее названных приложений, ИСР **Редактор Cicode** не запускается одновременно с приложением **Проводника Citect**. Для запуска ИСР **Редактор Cicode** достаточно в любой из сред **Проводник Citect**, **Редактор проектов Citect** или **Построитель графики Citect** нажать кнопку **Редактор Cicode** на панели инструментов или выполнить команду **Средства | Редактор Cicode**.

Упражнение 1.4

Запустите приложение **Проводник Citect** и выберите в нем проект **Example**. Нажмите кнопку **Запустить** на панели инструментов, чтобы запустить этот проект (в ответ на запрос нажмите кнопку **OK**). Перемещайтесь по проекту, переходя на новые страницы с помощью команд меню **Example**. Окончив ознакомление, закройте проект, нажав кнопку **Закрыть**.

Глава 2. Управление проектами

Как указывалось ранее, **Проводник Citect** является приложением, с помощью которого выполняется *управление проектами* и запуск конфигурационных приложений и исполняемых программ. В среде **Проводника Citect** выполняются основные задачи, к числу которых относятся создание, удаление, сохранение и восстановление проектов. Далее эти задачи будут последовательно рассмотрены.

2.1. Создание нового проекта. Включение проектов

Первое, что нужно сделать для конфигурирования нового Citect –проекта, это *создать новый проект*, в котором будет храниться информация о проекте. Каждый проект имеет собственную папку в папке **\User** установочного каталога Vijeo Citect. Папка проекта создается при создании проекта и ей присваивается то же самое имя, что и проекту.

> ***Замечание***
> В Vijeo Citect нельзя использовать для папок (проектов) длинные имена. Длина имени папки или проекта не должна превышать 64 символа, а имя может содержать любые символы, кроме точки с запятой и одиночной кавычки. Если в дальнейшем возникнут проблемы с просмотром списка тегов, то сократите длину имени проекта *с учетом пути*.

Для создания нового проекта в среде **Проводника Citect** нажмите кнопку **Новый** на панели инструментов, или выполните команду **Файл | Новый проект…**, или в поле **Список проектов** выберите любой элемент и выполните команду **Новый проект…** его контекстного меню (рис. 2.1).

Упражнение 2.1. Создание проекта Training

Создайте новый проект. С этой целью откройте приложение **Проводник Citect**, нажмите кнопку **Новый** на панели инструментов и настройте параметры проекта в соответствии с рис. 2.2. В поле **Название:** укажите имя **Training**, а проект **OvenTraining** используйте, при необходимости, в качестве справочного проекта. Проект должен быть расположен в папке

..\User,

местоположение которой было указано при инсталляции SCADA-системы Vijeo Citect. Нажмите кнопку **OK**. Теперь Vijeo Citect создаст базы данных конфигурации. Обратите внимание на элемент **Training**, появившийся в списке проектов.

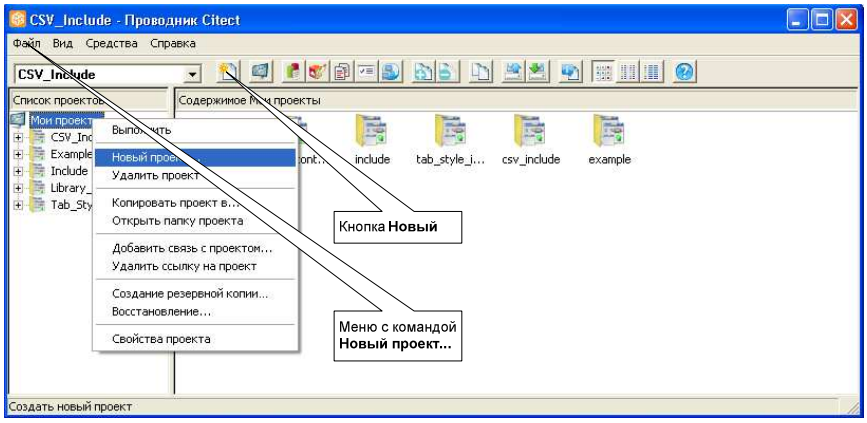


Рис. 2.1. Создание нового проекта

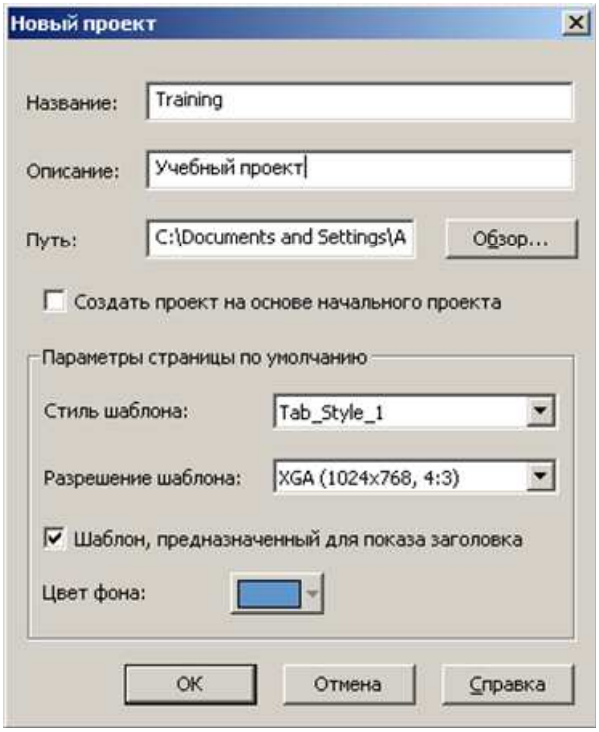


Рис. 2.2. Создание проекта **Training**

Совет

При заполнении диалогового окна, представленного ранее на рис. 2.2, для перехода в следующее поле окна пользуйтесь курсором мыши или клавишей **Tab**. Для возврата в предыдущее поле воспользуйтесь акселератором **Shift+Tab**.

При этом в качестве включаемого проекта по умолчанию используется включаемый проект **Tab_Style_Include**. Посмотреть включаемые проекты можно из среды приложения **Редактор проектов Citect** с помощью команды **Система | Включенные проекты** (рис. 2.3).

Включаемый проект Tab_Style_Include является предварительно сконфигурированным проектом и все его возможности наследуются новым проектом. Он содержит ряд шаблонов, которые можно использовать для создания новых графических страниц в стиле ОС Windows XP. Проект включает предопределенные страницы отображения трендов и сигналов тревог, страницу средств проектирования и др.

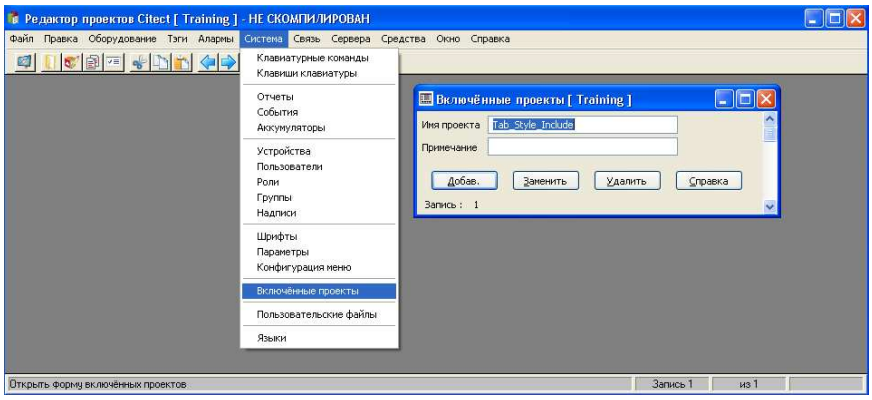


Рис. 2.3. Включаемый проект нового проекта

В больших системах удобнее разрабатывать проект не как один большой проект, а как ряд маленьких проектов. Например, можно создать отдельные, небольшие проекты для каждого участка завода. Тогда перед включением в основной проект можно разработать и проверить каждый небольшой проект (декомпозиция). Проект Vijeo Citect не включается в компиляцию какого-либо другого проекта, если он преднамеренно не включен в этот проект. Для включения другого проекта в текущий проект удобнее всего перейти в среду **Редактора проектов Citect** и выполнить команду **Система | Включенные проекты**.

В каждую систему Vijeo Citect входят несколько стандартных включаемых проекта, содержащие предопределенные записи баз данных. Проект **Tab_Style_Include** по умолчанию автоматически включается в каждый создаваемый проект.

> ***Замечание***
> Более подробные сведения о включаемых проектах содержатся в [2], тема справки Using Vijeo Citect | Administering Projects | Including Projects; [3], слайды 21, 40 — 42.

2.2. Архивирование, удаление и восстановление проекта

Проекты Vijeo Citect можно резервировать в архивных файлах, занимающих, при необходимости, намного меньше места, нежели исходные папки проекта. Во время разработки проекта резервирование следует выполнять регулярно на случай непреднамеренного удаления или разрушения файлов. Архивную копию можно сохранять на дискете, локальном диске, съемном диске или в сети. Очень важно вести историю архивных файлов, чтобы всегда можно было вернуться к предыдущей версии проекта (особенно это важно при внесении изменений в уже работающую систему).

Для архивирования проекта в среде **Проводника Citect** выберите проект, подлежащий резервированию, нажмите кнопку **Резервная копия** на панели инструментов, или выполните команду **Средства | Создание резервной копии…**, или выполните команду **Создание резервной копии…** контекстного меню выбранного проекта. Для восстановления проекта в среде **Проводника Citect** нажмите кнопку **Восстановить** на панели инструментов, или выполните команду **Средства | Восстановление…**, или выполните команду **Восстановление…** контекстного меню выбранного проекта. Для удаления проекта в среде **Проводника Citect** выберите проект, подлежащий удалению, выполните команду **Файл | Удалить проект**, или выполните команду **Удалить проект** контекстного меню выбранного проекта.

Совет

Не забывайте периодически сохранять проект на магнитном диске. Для сохранения вновь созданного проекта в среде **Проводника Citect** выберите в окне **Список проектов** проект **Training** и выполните команду **Создание резервной копии…** его контекстного меню. В появившемся окне **Созданеие резервной копии проекта** укажите требуемые параметры (в поле **Название:** введите **Training**, а в поле **Файл резервной копии: — Backup\Training.ctz**) и нажмите кнопку **OK** (рис. 2.4). Более подробные сведения о сохранении проекта содержатся в [2], тема Using Vijeo Citect | Administering Projects | Archiving projects | Backing up a project; [3], слайды 28, 34 — 37.

Примечание

По умолчанию, если не указано другое расширение, архивным файлам присваивается расширение **.ctz**. Эти файлы записываются в стандартном формате **zip**. Это значит, что для открытия такого файла можно воспользоваться любым средством извлечения zip-файлов.

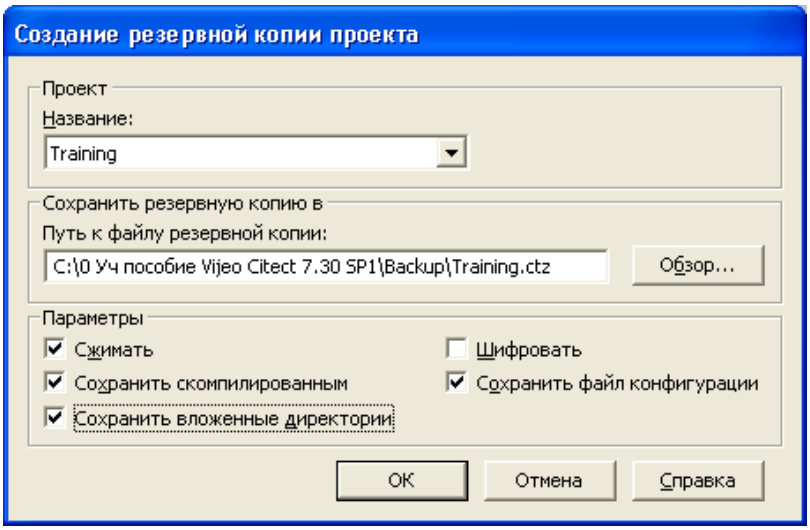


Рис. 2.4. Сохранение проекта

Упражнение 2.2. Удаление и восстановление проекта Training

Удалите проект **Training**, который был ранее архивирован в папке **Backup**. Для этого в среде **Проводника Citect** выберите проект, подлежащий удалению, выполните команду **Удалить проект** контекстного меню выбранного проекта. Появится диалоговое окно с предупреждением. Для подтверждения удаления проекта нажмите кнопку **Да**. Восстановите проект **Training**. Для этого в среде **Проводника Citect** нажмите кнопку **Восстановить** на панели инструментов. Появится диалоговое окно. С помощью кнопки **Обзор…** в папке **Backup** выберите архивный файл **Training.ctz,** радиокнопку **Новый проект** и нажмите кнопку **OK**. Появится следующая подсказка **Восстановление**. Нажмите кнопку **Да**. Когда появится подсказка **Восстановление завершено**, нажмите кнопку **OK**. Проект **Training** вновь появится в списке проектов.

2.3. Кластеры и серверы. Мастер конфигурирования компьютера

Кластеризация дает возможность группировать в пределах одного проекта независимые наборы серверных компонентов Vijeo Citect, что позволяет одновременно контролировать несколько систем и управлять ими. Для каждого проекта Vijeo Citect нужно иметь хотя бы по одному из следующих компонентов — кластер, сервер ввода-вывода, сервер отчетов, сервер сигналов тревог и сервер

трендов. Эти компоненты можно распределить между несколькими компьютерами, но в простейшем проекте Vijeo Citect эти компоненты размещены на одном компьютере. Такая конфигурация называется *автономной*. В данном учебном пособии будет рассматриваться автономная конфигурация проектов.

Таким образом, для проекта с автономной конфигурацией следует определить один кластер и по одному серверу ввода-вывода, отчетов, сигналов тревог и трендов в этом кластере. Вначале рассмотрим добавление в новый проект кластера и сервера ввода-вывода.

Упражнение 2.3. Добавление в проект Training кластера и сервера ввода-вывода

В среде **Проводника Citect** выберите проект **Training** и перейдите в среду **Редактора проектов Citect**. Для определения кластера выполните команду **Сервера | Кластеры**, сконфигурируйте кластер аналогично рис. 2.5 и нажмите кнопку **Добав**.

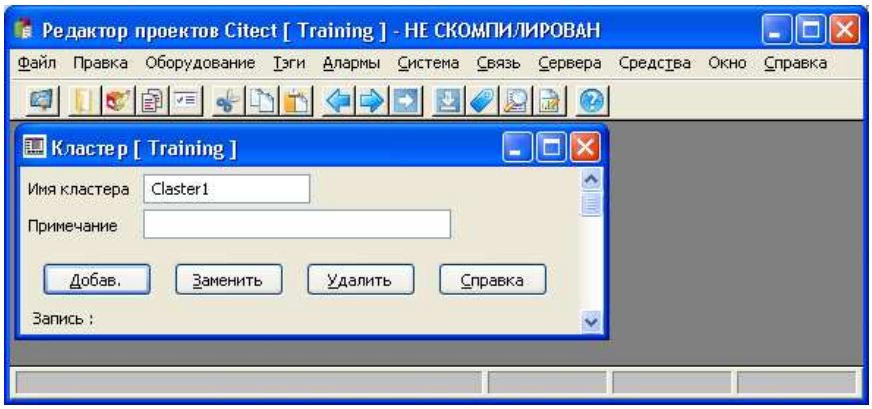


Рис. 2.5. Включение кластера

Для определения сервера ввода-вывода выполните команду **Сервера | Сервера ввода/вывода**, сконфигурируйте сервер аналогично рис. 2.6а и нажмите кнопку **Добав**.

Мастер конфигурирования компьютера позволяет быстро настроить компьютер в соответствии с требованиями проекта Vijeo Citect и является приложением, которое определяет коммуникации SCADA-системы и аппаратные средства компьютера. С его помощью можно при экспресс-установке определить, является ли компьютер клиентом только для просмотра, клиентом сервера и управления или же клиентом управления. При полной установке он задает возможность использования сигналов тревог, отчетов, трендов и событий. **Мастер конфигурирования компьютера**

указывает также опции и параметры, которые определяют поведение SCADA-приложения.

> ***Совет***
> Очень важно!!! При выборе другого проекта не забывайте перенастроить компьютер с помощью **мастера конфигурирования компьютера**.

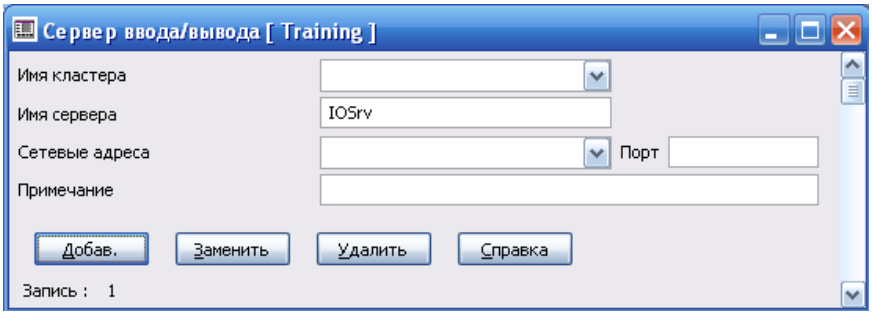


Рис. 2.6a. Включение сервера ввода-вывода

Мастер конфигурирования компьютера можно запустить из среды **Проводника Citect** тремя различными способами — нажатием кнопки **Мастер конфигурирования компьютера** на панели инструментов, с помощью команды **Средства | Мастер настройки компьютера** и, наконец, в поле **Список проектов** можно щелкнуть левой кнопкой мыши по **Мои проекты** и в поле **Содержимое Мои проекты** дважды щелкнуть левой кнопкой мыши по компоненте **Настройка компьютера**.

Перед настройкой компьютера необходимо сконфигурировать роли (хотя бы одну) и пользователей (хотя бы одного).

2.3.1. Роли, группы, привилегии и пользователи

Роли создаются для отдельных операторов или групп операторов чтобы установить для них разрешения — что они могут делать в пределах проекта. Для добавления роли в среде **Редактора проектов Citect** выполните команду **Система | Роли**, сконфигурируйте роль в соответствии с рис. 2.6b (пример такой роли вы найдете в предустановленном демонстрационном проекте **Example**) и нажмите кнопку **Добав**.

В поле **Имя роли** указывается идентификатор длиной не более 16 символов. Поле **Группа окон** используется при конфигурировании политики безопасности. В поле **Права** задан максимальный диапазон глобальных привилегий (подробнее об этом см. далее в подразд. 3.4). Привилегии также являются элементом политики безопасности и определяют разрешения для оперативного персонала.

Роль связывается с определенным оператором (пользователем). Для создания пользователя в среде **Редактора проектов Citect** выполните команду **Система | Пользователи**, сконфигурируйте пользователя в соответствии с рис. 2.6c

(пример такого пользователя вы также найдете в предустановленном демонстрационном проекте **Example**, пароль совпадает с именем пользователя) и нажмите кнопку **Добав**.

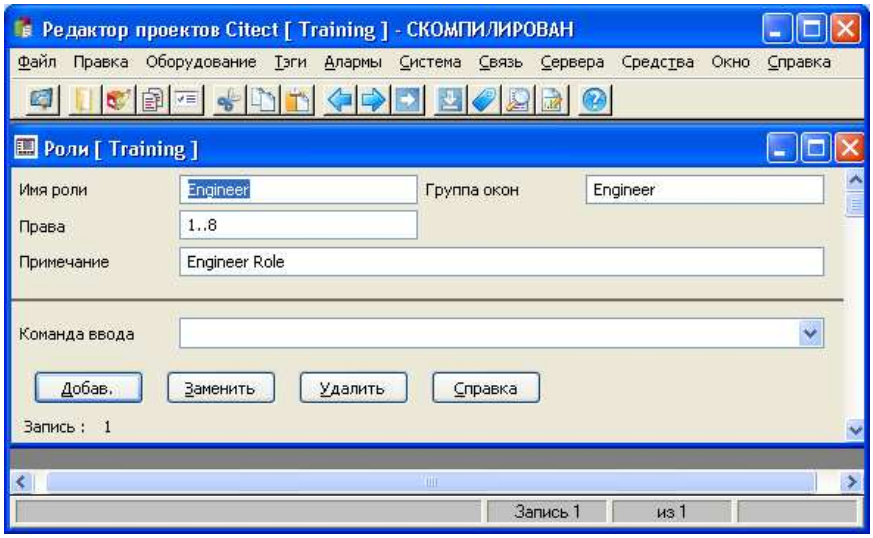


Рис. 2.6b. Добавление роли

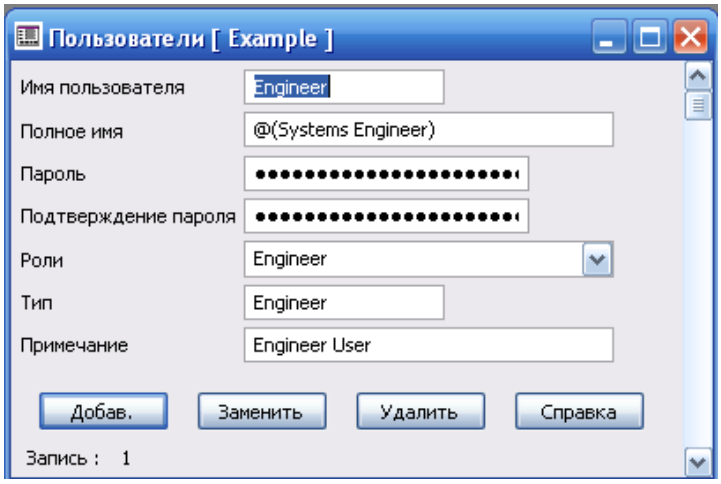


Рис. 2.6c. Добавление пользователя

Упражнение 2.4. Настройка компьютера для проекта Training

Перед настройкой компьютера выполните компиляцию проекта **Training**. Для этого в среде **Проводника Citect** выберите проект **Training**, перейдите в среду приложения **Редактор проектов Citect** и выполните команду **Файл | Компилировать**, или активизируйте акселератор **Alt+F10**, или нажмите кнопку **Компилировать проект** на панели инструментов. В конце компиляции появится предупреждающее сообщение (рис. 2.7), означающее, что устройства ввода-вывода еще не определены (это будет сделано далее). Нажмите кнопку **OK**.

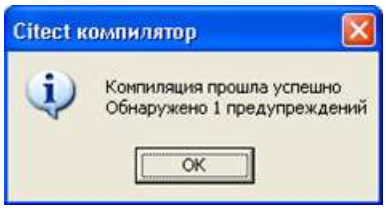


Рис. 2.7. Предупреждение об отсутствии в проекте устройств ввода-вывода

Для запуска **Мастера конфигурирования компьютера** выполните команду **Средства | Мастер настройки компьютера**. Появляется окно, представленное на рис. 2.8.

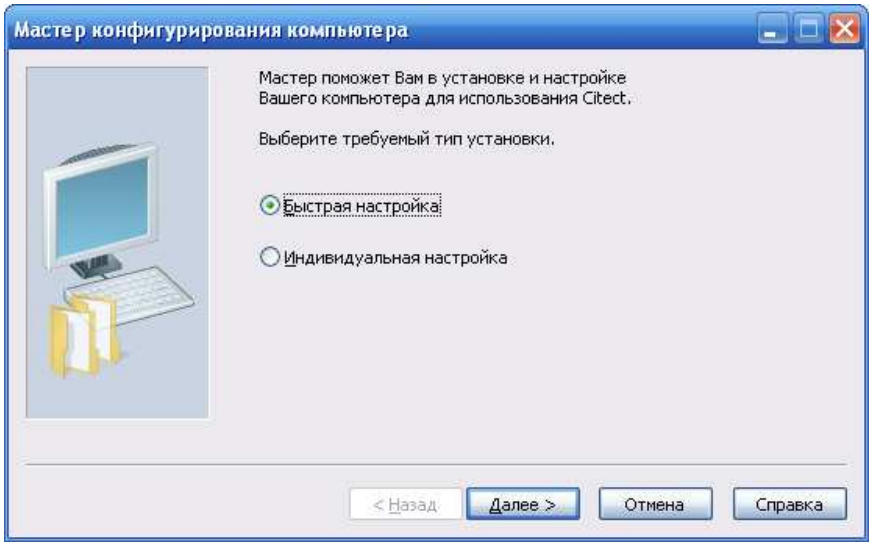


Рис. 2.8. Окно Мастера конфигурирования компьютера

Выберите **Быстрая настройка** и нажмите кнопку **Далее**. В результате появится окно **Настройка проекта**, которое следует настроить в соответствии с рис. 2.9 (в поле **Название проекта:** выбрать **Training**) и нажать кнопку **Далее**. Это диалоговое окно позволяет выбрать проект для выполнения. Затем последовательно появятся еще три окна настройки компьютера. Окно **Настройка зщли компьютера** настройте в соответствии с рис. 2.10. Поскольку работа в сети не была настроена, то по умолчанию выбран вариант **Клиент сервера и управления**, а остальные варианты не доступны. Нажмите кнопку **Далее**.

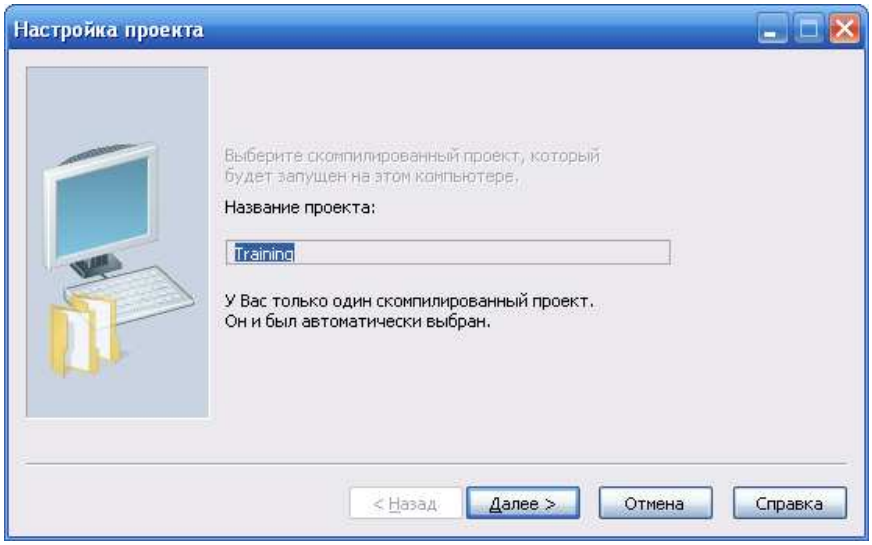


Рис. 2.9. Окно **Настройка проекта** мастера конфигурирования компьютера

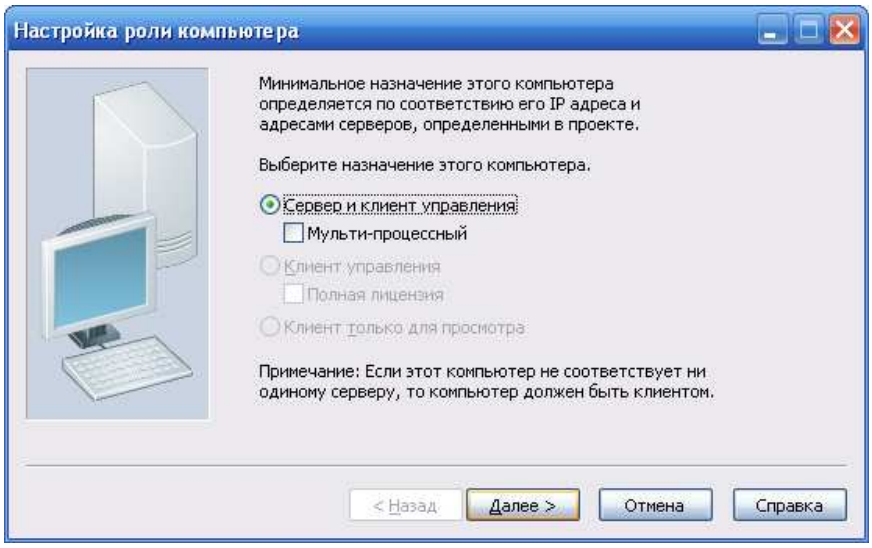


Рис. 2.10. Окно **Настройка назначения компьютера** мастера конфигурирования компьютера

Три последующих окна настройки компьютера **Настройка сети, Проверка подлинности сервера** и **Citect конфигурирование компьютера** настройте в соответствии с рис. 2.11, 2.12 и 2.13 и нажмите последовательно кнопки **Далее, Далее** и **Готово**. Диалоговое окно **Настройка сети** позволяет выбрать тип построения сети, используемый в проекте. Поскольку данный проект будет работать в автономном режиме, выбран вариант **Автономный**.

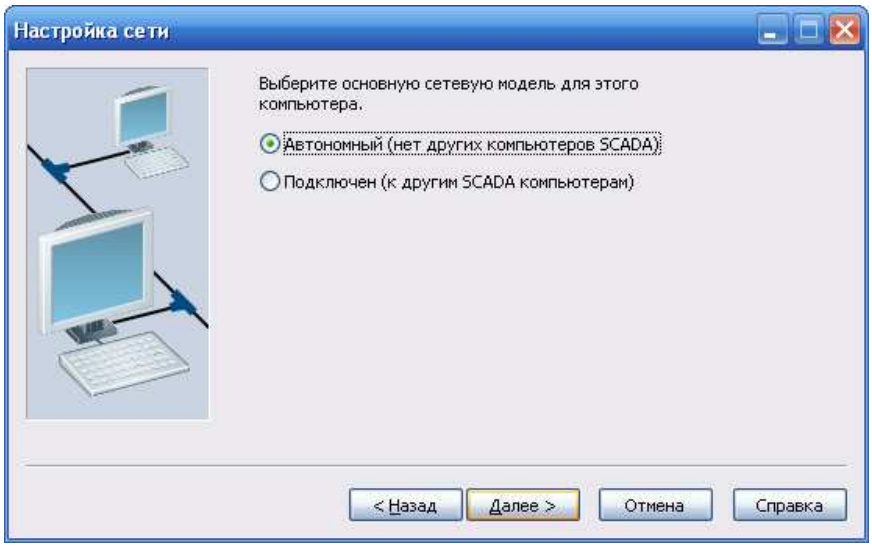


Рис. 2.11. Окно **Настройка сети** мастера конфигурирования компьютера

Замечание

Более подробные сведения о мастере настройки компьютера (Computer Setup Wizard) содержатся в [2], тема справки Using Vijeo Citect | Configuring Your System | Running the Computer Setup Wizard; [3], слайды 31 — 32.

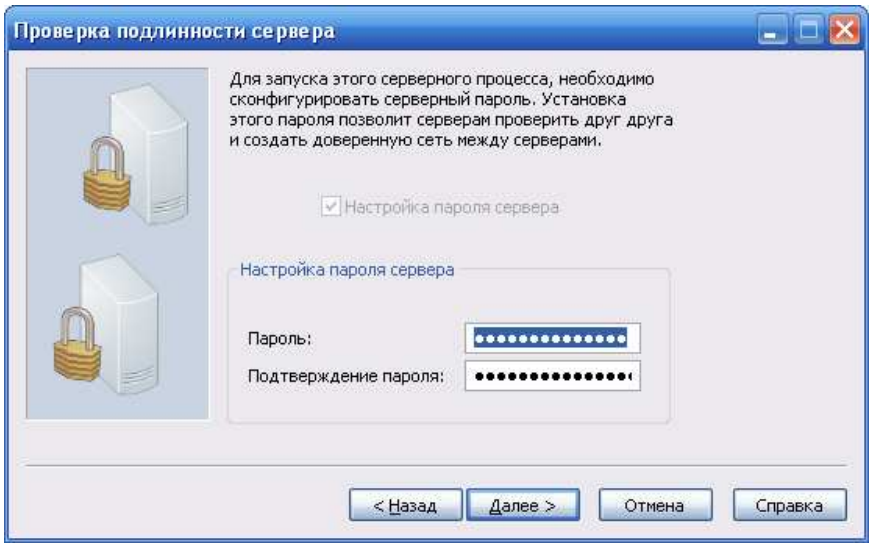


Рис. 2.12. Окно **Проверка подлинности сервера** (параль **dvg**)

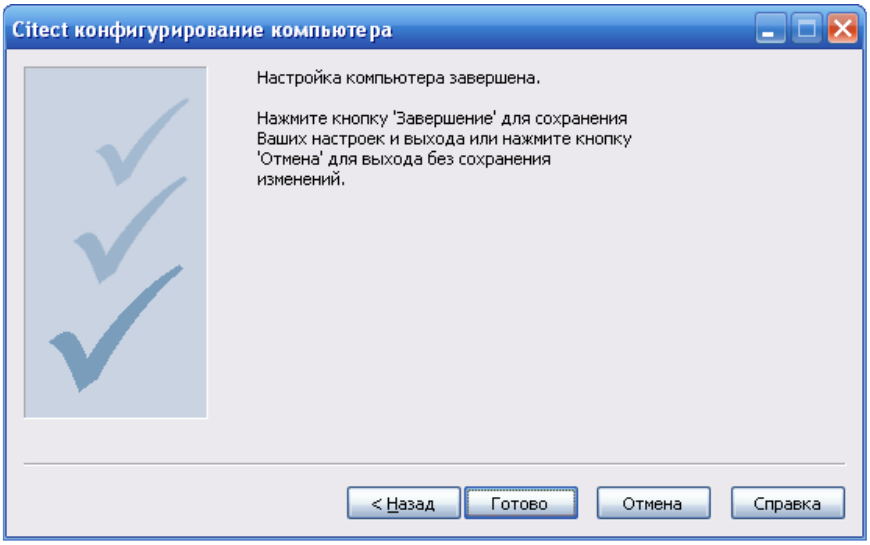


Рис. 2.13. Окно **Citect конфигурирование компьютера**

Глава 3. Настройка связи и работа с тегами

SCADA-система Vijeo Citect может связываться со многими типами устройств ввода-вывода систем контроля и управления — программируемыми логическими контроллерами, регуляторами, распределенными системами управления и т. п. Это позволяет обмениваться данными с устройствами и выполнять супервизорное управление системой.

Обмен информацией с устройствами ввода/вывода в SCADA-системе Vijeo Citect иллюстрирует рис. 3.1.

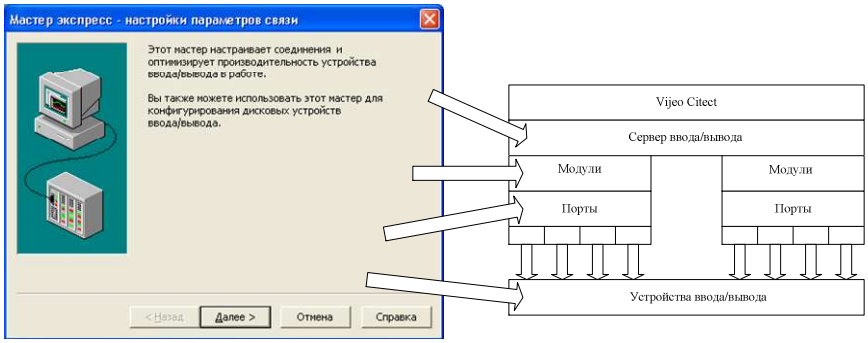


Рис. 3.1. Организация ввода/вывода в Vijeo Citect

В процессе информационного обмена, в общем случае, используются **Сервер ввода/вывода** (таких серверов в общем случае может быть несколько), **Модули**, **Порты** и **Устройства ввода/вывода**. Модули представляют собой интерфейсные платы компьютера, а порты — порты связи на этих платах. Настроить перечисленные компоненты удобнее всего и проще с помощью **Мастера экспресс-настройки параметров связи**.

3.1. Мастер быстрой настройки параметров связи

Мастер быстрой настройки параметров связи позволяет быстро установить соединение с новым или существующим устройством ввода-вывода. В каждом проекте Vijeo Citect должен быть хотя бы один сервер ввода-вывода, предназначенный для связи с устройствами ввода-вывода. Каждое устройство ввода-вывода следует настроить в Vijeo Citect в соответствии с протоколом и параметрами связи. Следует также определить все интерфейсные платы и порты связи в компьютере.

Совет

Прежде, чем продолжить работу, внимательно прочтите материал, относящийся к коммуникациям и **Мастеру быстрой настройки параметров связи** в [2], темы справки Using Vijeo Citect |Communicating with I/O Devices, Using Vijeo Citect | Implementing Clastering; [3], слайды 44 — 57.

Во время работы **Мастер быстрой настройки** определяет имена серверов ввода/вывода, интерфейсные платы компьютера, порты связи на этих платах и устройства ввода-вывода, подключенные к портам (см. приведенный ранее рис. 3.1).

Примечание

Каждый компьютер Vijeo Citect может выполнять функции только одного сервера ввода-вывода. При выполнении полного проекта сетью компьютеров Vijeo Citect, в которой к устройству ввода-вывода подключены несколько компьютеров, для проекта можно определить несколько серверов ввода-вывода.

Мастер быстрой настройки можно запустить двумя различными способами — из среды **Редактора проектов Citect** с помощью команды **Связь | Мастер быстрой настройки** или из среды **Проводника Citect**, если в поле **Список проектов** открыть требуемый проект (например, проект **Training**), щелкнуть левой кнопкой мыши по компоненте **Связь** и в поле **Содержимое Связь** дважды щелкнуть левой кнопкой мыши по компоненте **Экспресс-настройка устройств ввода/вывода**.

Упражнение 3.1. Настройка связи для проекта Training

С помощью **Мастера быстрой настройки** параметров связи настройте систему ввода/вывода для учебного проекта **Training**. Для этого запустите приложение **Проводник Citect** и выберите проект **Training**. Для перехода в среду приложения **Редактор проектов Citect** нажмите кнопку **Редактор проектов** на панели инструментов. Командой **Связь | Мастер Быстрой настройки** запустите **Мастер быстрой настройки**. В появившемся окне (рис. 3.2) нажмите кнопку **Далее** для продолжения. В следующем диалоговом окне задайте параметры **Сервера ввода/вывода** в соответствии с рис. 3.3 и нажмите кнопку **Далее**.

Появится диалоговое окно для конфигурирования **Устройства ввода/вывода**. Настройте его в соответствии с рис. 3.4 и нажмите кнопку **Далее**. Задайте тип **Устройства ввода/вывода** (рис. 3.5) и также нажмите кнопку **Далее**. В следующем диалоговом окне задайте для **Устройства ввода/вывода** изготовителя, тип и метод коммуникации (рис. 3.6) и нажмите кнопку **Далее**. В появившемся диалоговом окне ничего не меняйте (рис. 3.7) и для продолжения нажмите кнопку **Далее**. В последнем диалоговом окне содержатся результирующие сведения (рис. 3.8). Нажмите кнопку **Готово**.

Настройка связи для **Модулей** и **Портов** пока не требуется и мы ее не будем выполнять. Результаты настройки связи представлены на рис. 3.9. Показанные на этом рисунке диалоговые окна связи визуализированы с помощью команд **Сервера | Кластеры**, **Сервера | Сервера ввода/вывода** и **Связь | Устройства ввода/вывода**.

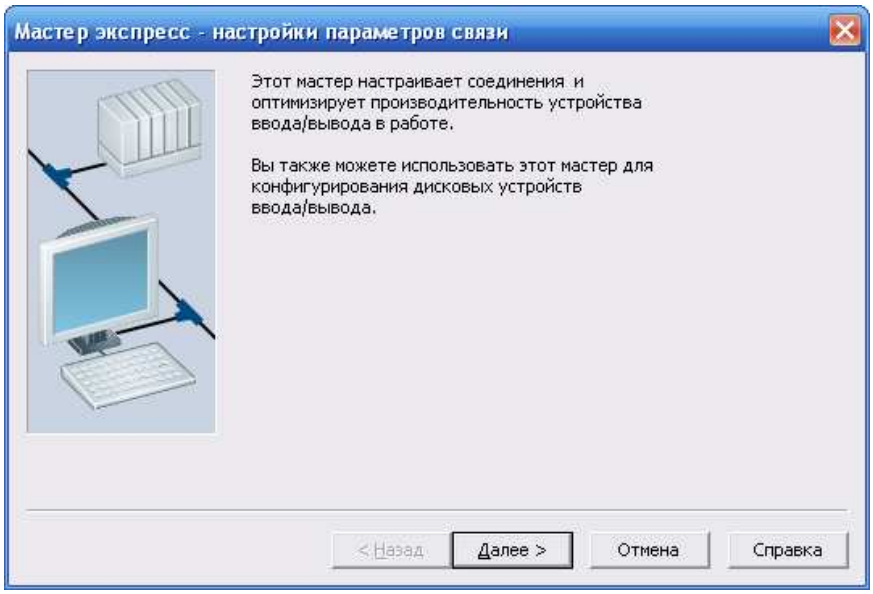


Рис. 3.2. Окно **Мастера быстрой настройки** параметров связи

Все *диалоговые окна (формы)* в системе Vijeo Citect имеют одинаковый интерфейс (см. рис. 3.9 и табл. 3.1). Полоса прокрутки диалогового окна позволяет прокручивать список от записи к записи. Записи содержатся в базе данных в порядке ввода.

> ***Совет***
> Чтобы найти конкретную запись, при открытом диалоговом окне выполните команду **Найти | Найти…** и ищите в текущей форме. Тогда окно диалога отфильтрует только совпадающие записи. Если будет найдено более одной соответствующей записи, то воспользуйтесь полосой прокрутки.

3.2. Тестирование связи (на примере проекта Training)

Тестирование связи очень важно выполнить на ранних этапах разработки проекта. Без надежной связи проект Vijeo Citect не будет работать эффективно. Далее последовательно рассматриваются выполненная ранее настройка связи и ее тестирование путем создания тега, используемого в графической странице.

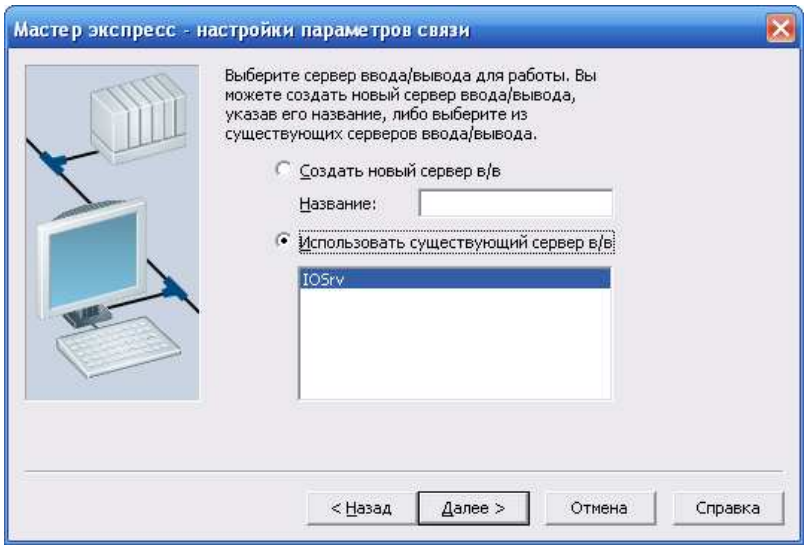


Рис. 3.3. Окно **Мастера быстрой настройки** для конфигурирования **Сервера ввода/вывода**

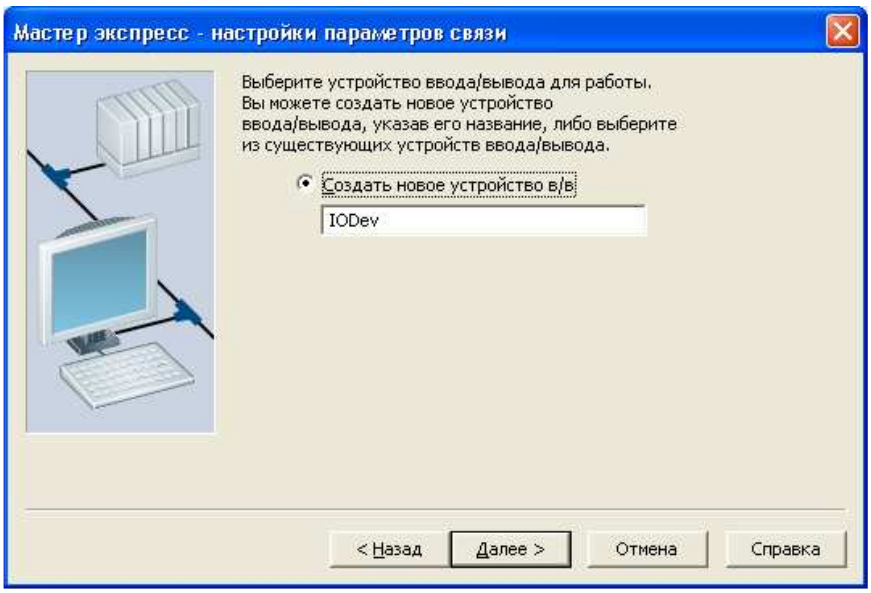


Рис. 3.4. Окно **Мастера быстрой настройки** для конфигурирования **Устройства ввода/вывода**

Теги (дескрипторы) переменных определяют данные, передаваемые между устройством ввода-вывода и сервером ввода-вывода Vijeo Citect. Каждый тег переменной определяется уникальным именем, типом данных, адресом и связанным с переменной устройством ввода-вывода ([3], слайды 56, 57).

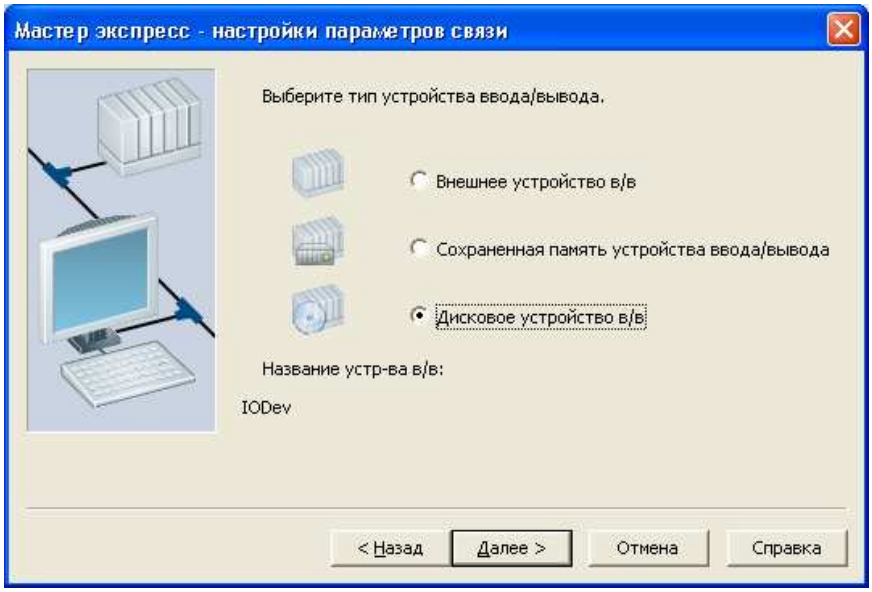


Рис. 3.5. Окно **Мастера быстрой настройки** для задания типа **Устройства ввода/вывода**

Определить тег переменной можно тремя различными способами — из среды приложения **Редактор проектов Citect** нажатием кнопки **Переменные теги** на панели инструментов, с помощью команды **Теги | Переменные теги** и, наконец, из среды приложения **Проводник Citect** в поле **Список проектов** можно открыть требуемый проект (например, проект **Training**), щелкнуть левой кнопкой мыши по компоненте **Теги** и в поле **Содержимое Теги** дважды щелкнуть левой кнопкой мыши по компоненте **Переменные теги** (рис. 3.10).

Совет

Прежде, чем продолжить работу, внимательно прочтите материал, относящийся к созданию тегов, графических станиц и использованию в графических страницах простейших объектов, таких как кнопки и текст ([2], темы справки Using Vijeo Citect | Tagging Process Variables, Using Vijeo Citect | Defining and Drawning Graphics Pages, [3], слайды 59 — 64).

Упражнение 3.2. Создание дискетного тега

Для тестирования созданных связей создайте тег переменной с именем **Test**, который может принимать нулевое или единичное значения. Для этого запустите приложение **Проводник Citect** и выберите проект **Training**. Для перехода в среду приложения **Редактор проектов Citect** нажмите кнопку **Редактор проектов** на панели инструментов. Из среды **Редактор проектов Citect** выполните команду **Теги | Переменные теги**, в появившемся диалоговом окне укажите параметры тега в соответствии с рис. 3.11 и нажмите кнопку **Добавить**. Используемый в проекте **Training** протокол **Generic**, являющийся внутренним протоколом Vijeo Citect, использует следующие удобные и естественные соглашения адресации, представленные в табл. 3.2.

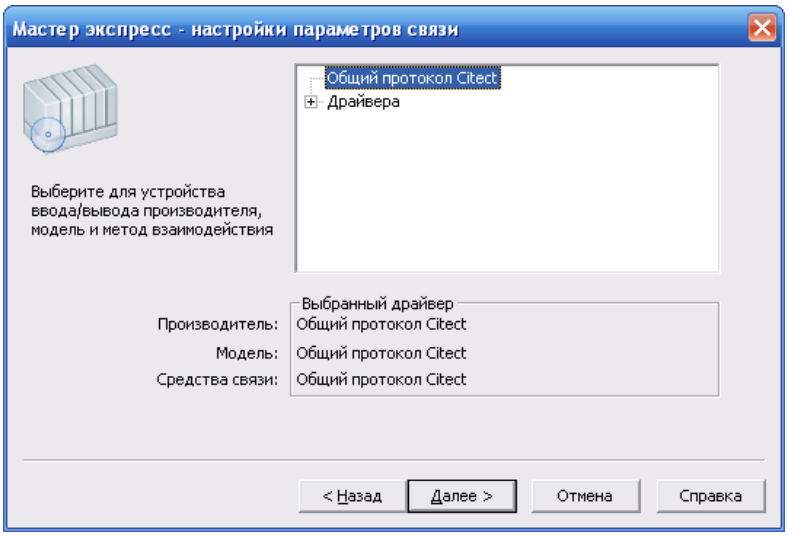


Рис. 3.6. Окно **Мастера быстрой настройки** для задания производителя, модели и метода коммуникации **Устройства ввода/вывода**

Более подробные сведения об адресации тегов для протокола **Generic** можно получить путем просмотра файла **..\Program Files\Schneider Electric\ Vijeo Citect 7.30\Bin\generic.dbf** с помощью приложения **Microsoft Excel** (рис. 3.12). Аналогичным образом можно получить информацию об адресации тегов и для протоколов других фирм. Например, для протокола **Omron** аналогичную информацию можно получить из файла **Omron.dbf**.

Создание новой графической станицы выполняется в среде приложения **Построитель графики Citect** с помощью кнопки **Новый** или команды **Файл | Новый…** (рис. 3.13).

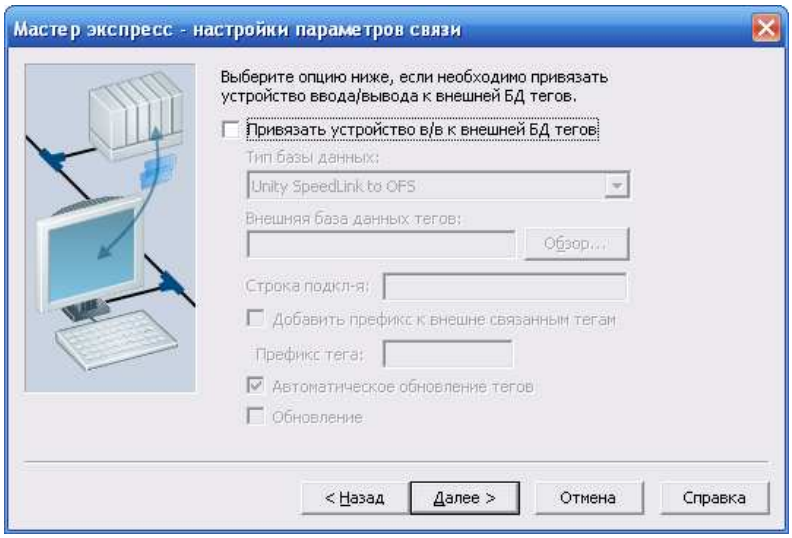


Рис. 3.7. Окно **Мастера экспресс-настройки** (используйте предложенные значения) для **Устройства ввода/вывода**

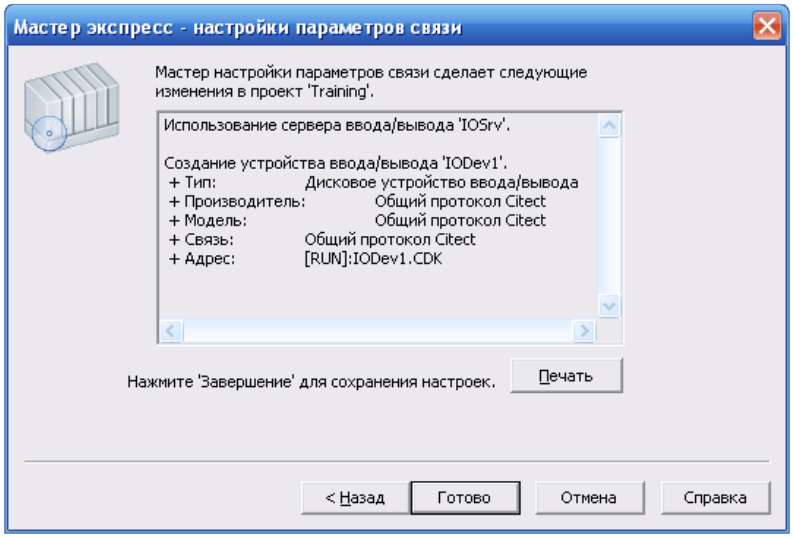


Рис. 3.8. Результирующее окно **Мастера экспресс-настройки**

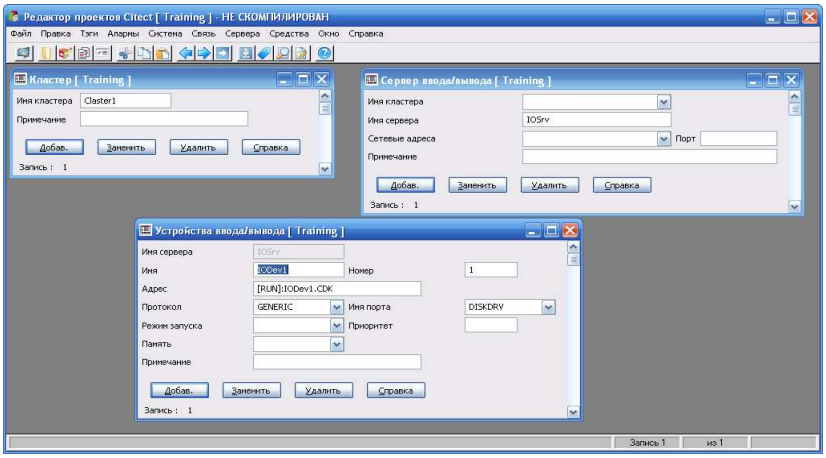


Рис. 3.9. Результирующие настройки коммуникаций проекта **Training**

Таблица 3.1. *Назначение кнопок диалоговых окон (форм)*

Кнопка	**Назначение кнопки**
Добав	Добавление отображаемой в данный момент информации к базе данных в качестве новой записи
Заменить	Замена текущей записи отображаемой в данный момент информацией
Удалить	Удаление текущей записи (отметка текущей записи для удаления)
Справка	Открытие окна **Vijeo Citect Help** для параметров в текущей форме

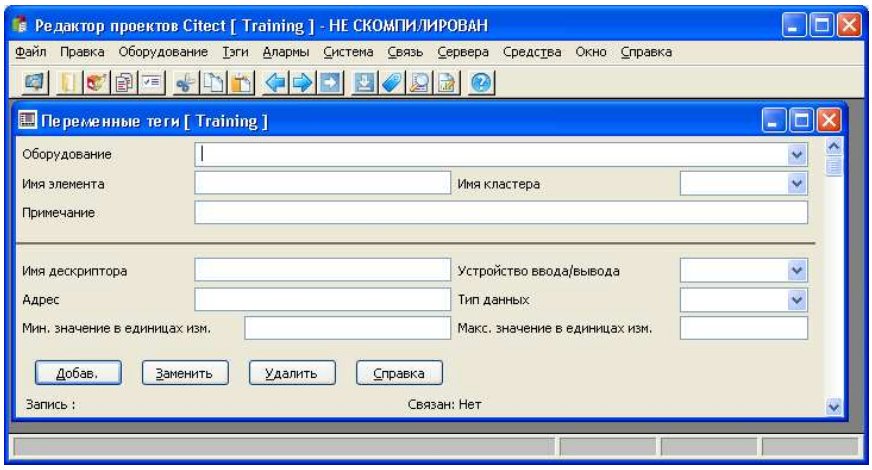


Рис. 3.10. Диалоговое окно для определения тега переменной проекта **Training**

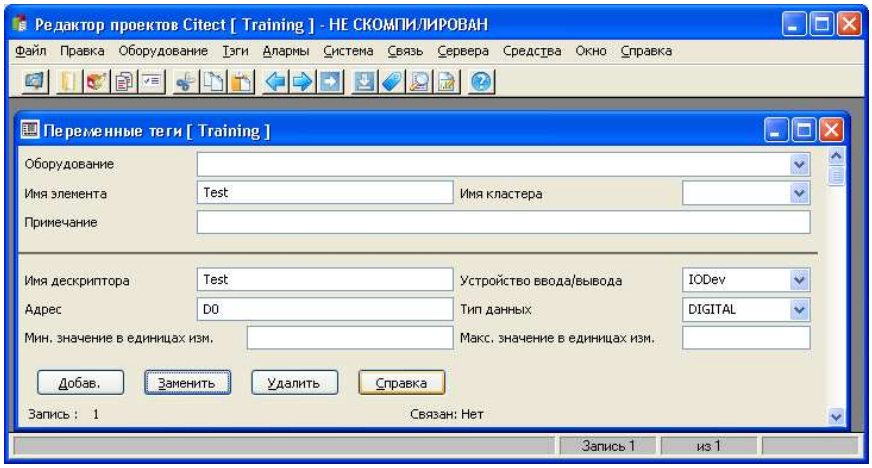


Рис. 3.11. Определение тега переменной **Test** проекта **OvenTraining**

Таблица 3.2. *Адресация тегов для протокола Generic*

Data Type	**Adress**
INT, UINT	Iномер
DIGITAL	Dномер
STRING	Sномер
REAL	Rномер

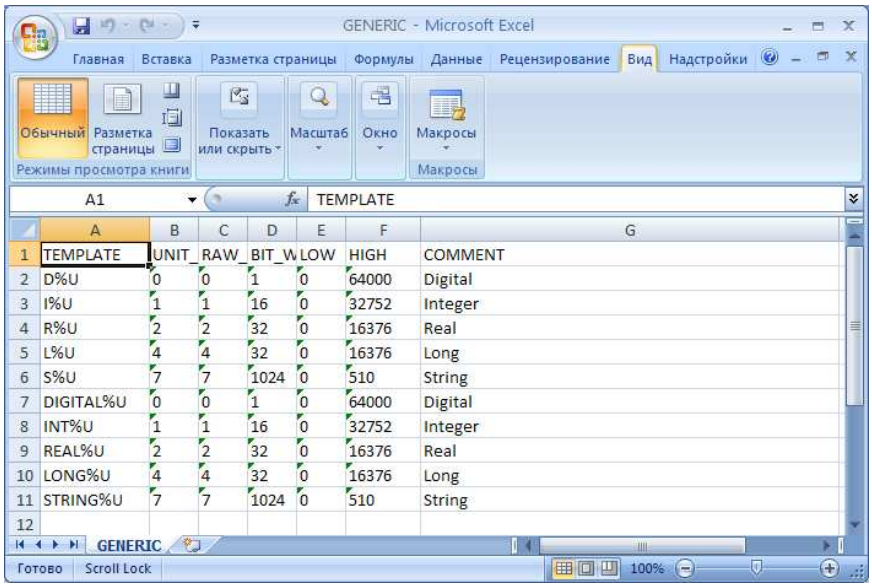
Data Type	Adress
LONG, LONGBCD	Lномер или Iномер
BCD, BYTE	Не поддерживаются

Упражнение 3.3. Создание графической страницы

Для *создания новой графической страницы* с использованием предопределенного шаблона в появившемся окне **Новый** следует нажать кнопку **Страница**, в результате чего появится следующее окно **Использовать шаблон**, представленное на рис. 3.14. Используйте параметры шаблона графической страницы, предлагаемые по умолчанию, и нажмите кнопку **OK**. Появится новая графическая страница. Для ее конфигурирования выполните команду **Свойства страницы…** ее контекстного меню (вызывается с помощью правой кнопки мыши), выберите вкладку **Представление**, в поле **Цвет страницы:** задайте серый цвет фона страницы и нажмите кнопку **OK**. С помощью команды **Файл | Сохранить как…** (рис. 3.15) сохраните страницу под именем **NewPage** (выберите вкладку **Страница**, в поле **Страница** укажите имя **NewPage** и нажмите кнопку **OK**). Графическая страница приобретет вид, показанный на рис. 3.16.

Совет

Никогда не русифицируйте имена графических страниц. В этом случае в дальнейшем могут возникнуть проблемы.



	A	B	C	D	E	F	G
1	TEMPLATE	UNIT_	RAW_	BIT_W	LOW	HIGH	COMMENT
2	D%U	0	0	1	0	64000	Digital
3	I%U	1	1	16	0	32752	Integer
4	R%U	2	2	32	0	16376	Real
5	L%U	4	4	32	0	16376	Long
6	S%U	7	7	1024	0	510	String
7	DIGITAL%U	0	0	1	0	64000	Digital
8	INT%U	1	1	16	0	32752	Integer
9	REAL%U	2	2	32	0	16376	Real
10	LONG%U	4	4	32	0	16376	Long
11	STRING%U	7	7	1024	0	510	String
12							

Рис. 3.12. Адресация тегов для протокола **Generic**

Упражнение 3.4. Добавление в страницу графических объектов и их настройка

Для тестирования созданных связей поместите в созданную графическую страницу **NewPage** три графических объекта — две **Кнопки** и объект **Текст**. Предварительно посмотрите в [3] слайды 74, 75 и 79. Продемонстрируйте использование действий, перечисленных в [3] на слайде 79. Настройте свойства объектов **Кнопка** таким образом, чтобы при их нажатии созданный тег **Test** принимал соответственно нулевое и единичное значения, а объект **Текст** настройте так, чтобы при изменении значения тега происходила анимация текста и его цвета.

Далее рассмотрим *добавление объекта Кнопка* в созданную графическую страницу **NewPage**. Это следует выполнять в среде **Построитель графики Citect** (см. приведенный ранее рис. 3.16).

Примечание

Для загрузки требуемой графической страницы достаточно нажать кнопку **Открыть** на панели инструментов, или выполнить команду **Файл | Открыть…**, или же активизировать акселератор **Ctrl+O**. Далее в появившемся окне **Открыть** следует выбрать во вкладке **Страница** имя нужного проекта и страницы (см. приведенный ранее рис. 3.15) и нажать кнопку **OK**.

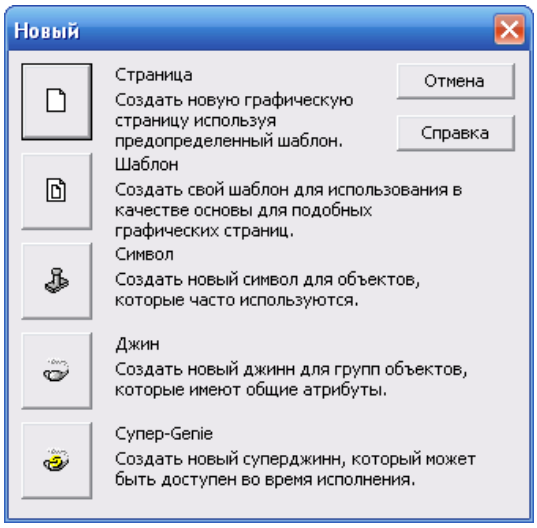


Рис. 3.13. Создание новой графической страницы

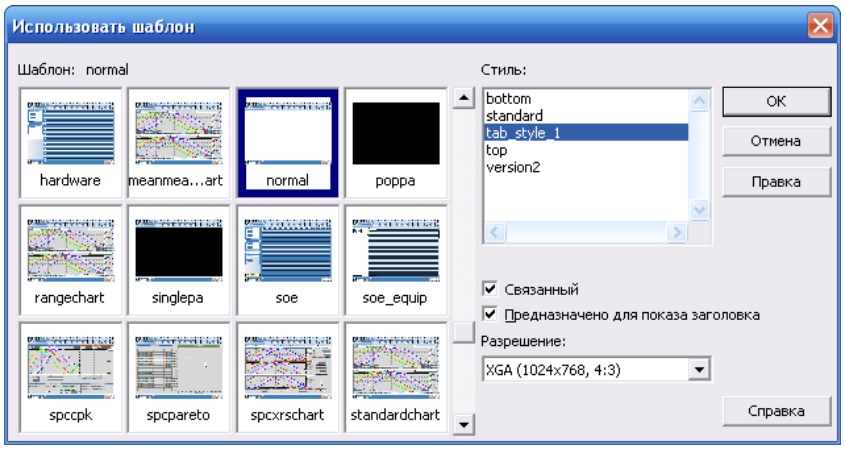


Рис. 3.14. Задание параметров шаблона графической станицы

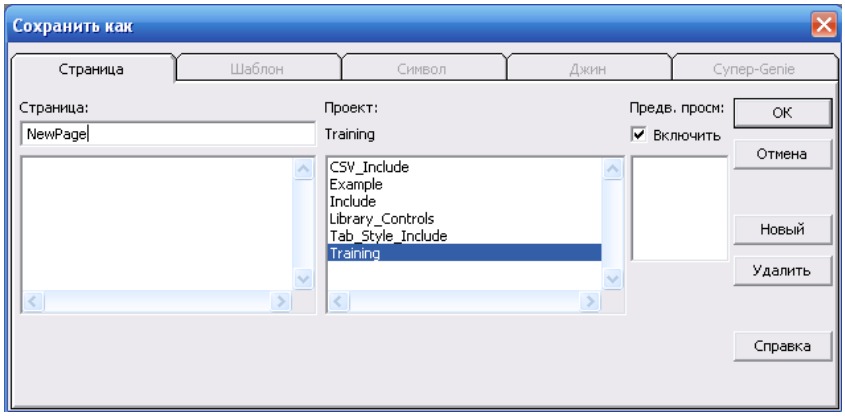


Рис. 3.15. Задание имени новой графической страницы

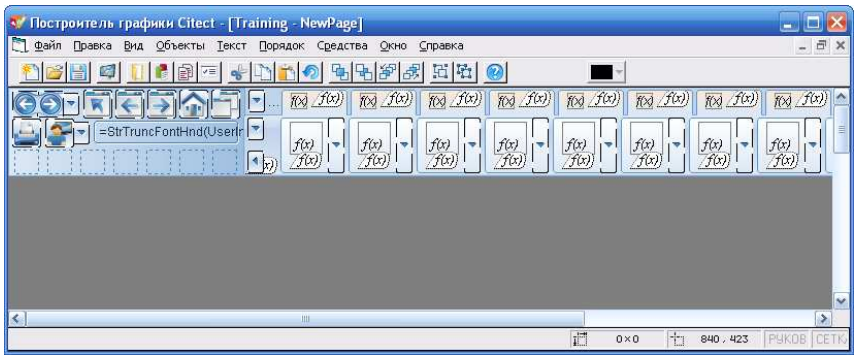


Рис. 3.16. Вид новой графической страницы после ее создания и конфигурирования

Для размещения объекта **Кнопка** в графической странице достаточно в окне объектов (рис. 3.17) выбрать нажатием и отпусканием левой кнопки мыши объект **Кнопка**, переместить мышь в нужное место графической страницы, нажатием левой кнопки и перемещением мыши определить размеры и местоположение кнопки и отпустить левую кнопку мыши. В результате появится окно **Свойства: Кнопка**, которое пока закройте. Аналогичным образом добавьте в графическую страницу еще один объект **Кнопка**, чтобы графическая страница приобрела вид, представленный на рис. 3.18.

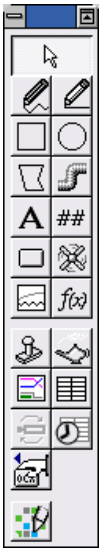


Рис. 3.17. Окно объектов

Рис. 3.18. Графическая страница **NewPage** с добавленными кнопками

Предварительно посмотрите материал, представленный в [3] на слайдах 76, 82 и 84. Для определения требуемых свойств объекта **Кнопка** следует ее выбрать (см. приведенный ранее рис. 3.18) и выполнить для кнопки команду **Свойства…** контекстного меню, вызываемого правой кнопкой мыши. Появится окно конфигурирования кнопки, показанное на рис. 3.19. В этом окне имеются четыре основные (постоянные) горизонтальные закладки (**Представление**, **Движение**, **Ввод**, **Доступ, Метаданные**) и по несколько вертикальных, вспомогательных закладки, варьируемых в зависимости от выбранной горизонтальной закладки (**Общие, 3D эффекты** и **Видимость** при выборе **Представление**; **По горизонтали** и **По вертикали** при выборе **Движение**; **Касание** и **Клавиатурные команды** — при выборе **Ввод**; **Общие** и **Запрещен** — при выборе **Доступ, Общие** — при выборе **Метаданные**). С помощью указанных закладок можно выбирать любую из десяти вкладок, с помощью которых задаются требуемые свойства кнопки.

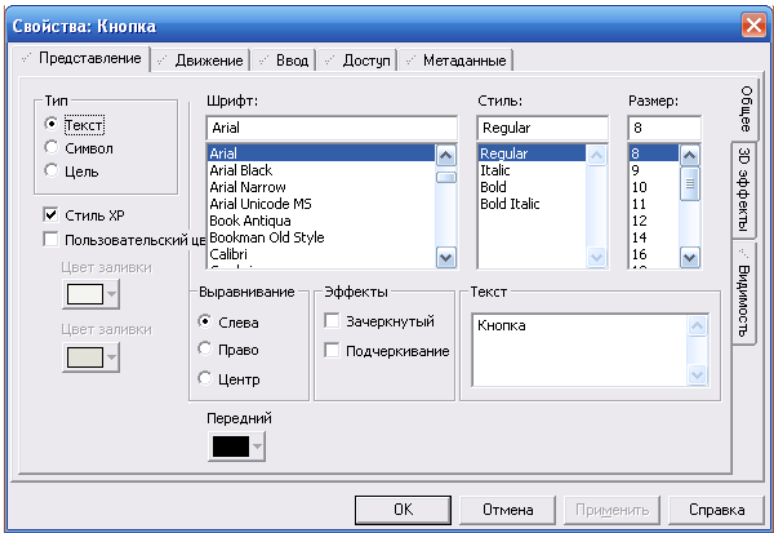


Рис. 3.19. Окно конфигурирования кнопки: вкладка **Представление (Общие)**

Примечание [A1]: Окно свойств объекта вместо вкладки «Вид» содержит вкладку «Представление»

Свойства кнопки, обеспечивающей задание нулевого значения тега **Test** при нажатии кнопки, иллюстрируют рис. 3.20. Аналогично, свойства другой кнопки, обеспечивающей задание единичного значения тега **Test** при нажатии кнопки, представлены на рис. 3.21 и 3.22.

Для размещения *объекта Текст* в графической странице достаточно в окне объектов (см. приведенный ранее рис. 3.17) выбрать нажатием и отпусканием левой кнопки мыши объект **Текст**, переместить мышь в нужное место графической страницы, ввести первую букву текста и нажать левую кнопку мыши. В результате появится окно **Свойства: Текст**, в котором задайте свойства объекта в соответствии с рис. 3.23 и 3.24.

Для сохранения модифицированной страницы **NewPage** нажмите кнопку **Сохранить** на панели инструментов, или выполните команду **Файл | Сохранить**, или же активизируйте акселератор **Ctrl+S**. Для проверки работы созданных кнопок и текста командой **Файл | Компилировать** или с помощью акселератора **Alt+F10** выполните компиляцию проекта.

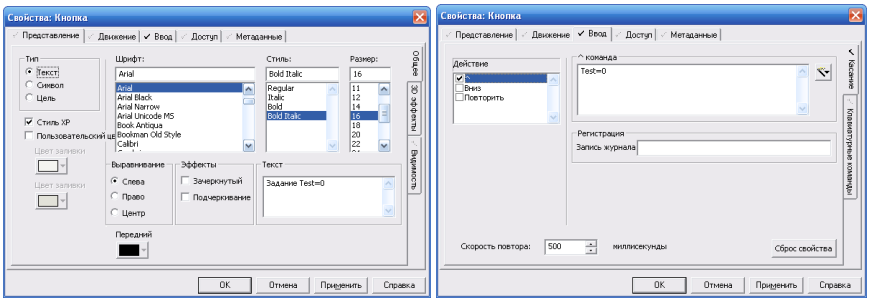


Рис. 3.20. Конфигурирование кнопки: обратите внимание на символы ✓, появившиеся в закладках модифицированных вкладок

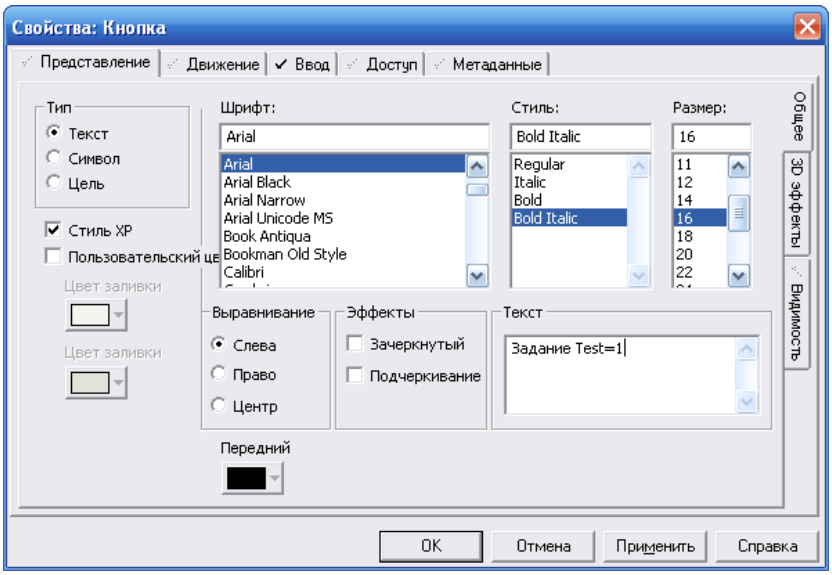


Рис. 3.21. Конфигурирование кнопки (свойство **Вид (Общие)**)

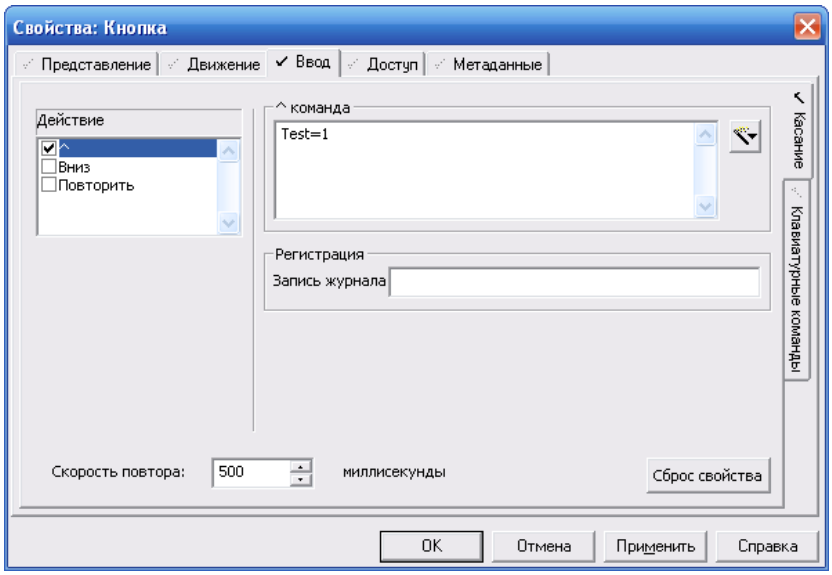


Рис. 3.22. Конфигурирование кнопки (свойство **Ввод (Касание)**)

Примечание

При компиляции проекта Vijeo Citect проверяет наличие ошибок. Если ошибки выявлены, выберите **Перейти**, чтобы отобразить место ошибки. Исправив все ошибки, сохраните исправленные графические ошибки и снова выполните компиляцию проекта.

Упражнение 3.5. Настройка компьютера, запуск проекта и тестирование связей

Настройте компьютер с помощью **Мастера конфигурирования компьютера** (см. подробнее приведенное ранее упражнение 2.4). Запустите его нажатием кнопки **Запустить проект**, или с помощью команды **Файл | Запустить**, или же нажатием клавиши **F5**.

Если Вы работаете с демонстрационной версией SCADA-системы, то появляется окно приглашения, представленное на рис. 3.25, в котором для перехода в демонстрационный режим следует нажать кнопку **OK**. Когда проект запустится, отобразится графическая страница **Start**.

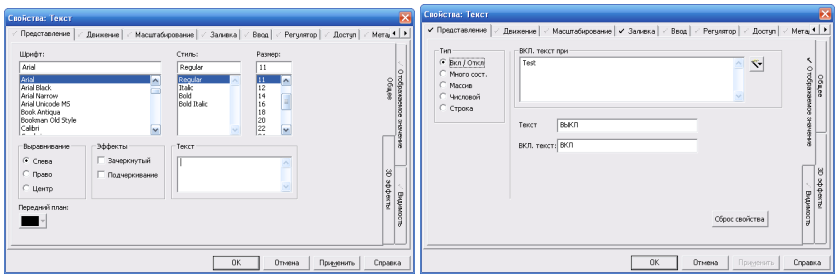


Рис. 3.23. Конфигурирование текста (свойства **Представление (Общие)** и **Представление (Отображаемое значение)**)

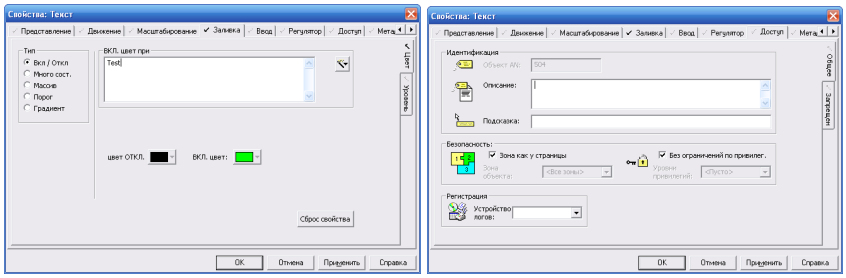


Рис. 3.24. Конфигурирование текста (свойства **Заливка (Цвет)** и **Доступ (Общие)**)

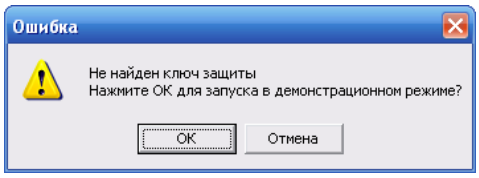


Рис. 3.25. Окно приглашения при работе с демонстрационной версией SCADA-системы

В появившемся окне для тестирования созданной страницы достаточно выполнить команду **Pages | NewPage** (рис. 3.26 и 3.27).

> ***Замечание***
> Если сервер ввода-вывода не имеет связи с устройством ввода-вывода, то на месте текстового объекта появится сообщение **#COM**.

Для завершения работы приложения достаточно нажать кнопку **Закрыть**, расположенную в правом верхнем углу, или выполнить команду **Завершение** системного меню приложения.

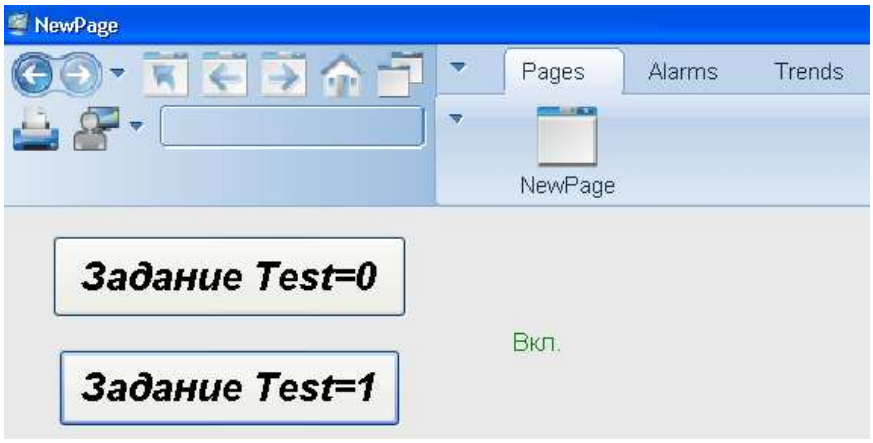


Рис. 3.26. Активизация созданной страницы и ее вид при нажатии кнопки **Задание Test=1**

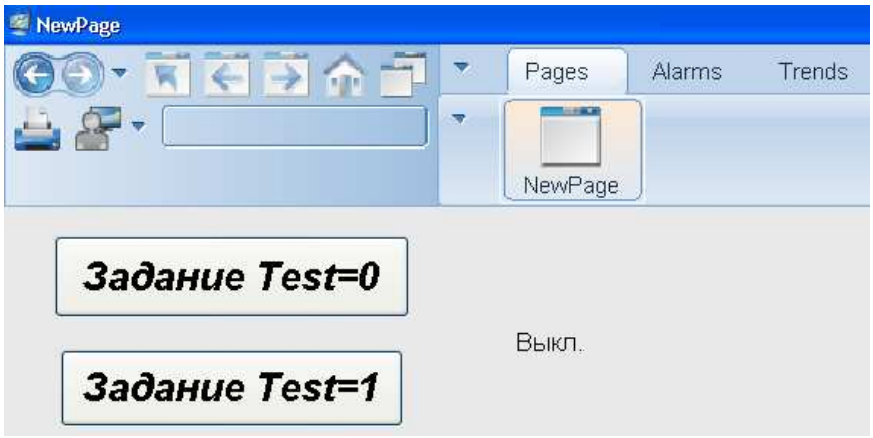


Рис. 3.27. Активизация созданной страницы и ее вид при нажатии кнопки **Задание Test=0**

Совет

Поэкспериментируйте, задавая различные параметры объектов **Кнопка** и **Текст** — это очень полезно. Проанализируйте результаты внесенных изменений в процессе работы приложения. После завершения экспериментов остановите работу приложения так, как это было указано ранее.

3.3. Структурированные имена тегов

Система Vijeo Citect налагает некоторые ограничения на *имена тегов переменных*. Благодаря этому облегчается и ускоряется разработка, настройка и запуск проекта, а также сокращается время на последующее его сопровождение. Соблюдение правил именования тегов переменных особенно полезно при использовании таких средств, как джины и суперджины (о них пойдет речь далее).

Каждое имя тега может содержать до 79 символов. Для соблюдения правил именования символы имени следует разделять на секции, описывающие характеристики тега. Например, зона размещения тега, тип тега и любые конкретные признаки. В правилах об именовании, принятых в Vijeo Citect, предполагается наличие четырех основных секций:

```
Зона_Тип_Местонахождение_Атрибут
```

> ***Примечание***
> Дополнительная информация о структурных именах тегов имеется в [2], тема Using Vijeo Citect | Tagging Process Variables | Tag Naming | Using structured tag names и в [3], слайды 60 — 62.

Секция **Зона** определяет зону, номер или название установки. Например, при наличии трех пастеризаторов с одинаковым управлением можно сконфигурировать теги для пастеризатора №1 и скопировать их на пастеризаторы №№2 и 3. Тогда нужно просто изменить секцию зоны в именах тегов на зоны второго и третьего пастеризаторов (табл. 3.3). Остальные теги остаются неизменными. Если эта возможность не нужна, секцию зоны в имени тега можно опустить, уменьшив тем самым количество символов в имени.

Таблица 3.3. *Использование секции зоны в именах тегов*

Участок	Имя тега
Пастеризатор 1	**P1**_TIC_101_PV
Пастеризатор 2	**P2**_TIC_101_PV
Пастеризатор 3	**P3**_TIC_101_PV

В секции **Тип** указывается тип параметра, технологического оборудования или средства управления (табл. 3.4). Рекомендуется система именования по стандарту ISA.

Таблица 3.4. *Использование секции типа в именах тегов*

Имя тега	Значение
P1_**TIC**_101_PV	**Регулятор, показывающий температуру**
P1_**FIC**_101_PV	**Регулятор, показывающий расход**
P1_**PUMP**_101_PV	**Насос**
P1_**VALVE**_101_PV	**Вентиль**

В секции **Местонахождение** указывается номер оборудования (табл. 3.5).

Таблица 3.5. *Использование секции местонахождения в именах тегов*

Имя тега	Значение
P1_TIC_**101**_PV	**Регулятор, показывающий температуру №101**
P1_TIC_**102**_PV	**Регулятор, показывающий температуру №102**
P1_PUMP_**101**_PV	**Насос №101**
P1_PUMP_**102**_PV	**Насос №102**

В секции **Атрибут** указывается атрибут конкретного параметра, связанный с оборудованием (табл. 3.6).

Таблица 3.6. *Использование секции атрибутов в именах тегов*

Имя тега	Значение
P1_TIC_101_**PV**	**Переменная процесса**
P1_TIC_101_**SP**	**Заданное значение**
P1_TIC_101_**OP**	**Выходное данное**
P1_TIC_101_**P**	**Коэффициент пропорциональности**
P1_TIC_101_**I**	**Коэффициент интегрирования**
P1_TIC_101_**CMD**	**Управляющий сигнал**
P1_TIC_101_**V**	**Значение**

3.4. Добавление тегов и их редактирование с помощью приложения Microsoft Excel

Теперь, настроив и протестировав связь между сервером ввода-вывода и устройством ввода-вывода, можно определить теги переменных, необходимые для дальнейшего использования. Теги переменных определяются точно так же, как ранее был определен тег **Test**.

Теги переменных можно сконфигурировать быстро, т. к. в каждом теге есть много повторяющейся информации. Если два тега переменных похожи друг на друга, отобразите информацию для уже созданного тега, измените нужные поля и нажмите кнопку **Добавить**. Для изменения параметров существующего тега отобразите его и введите изменения, а затем нажмите кнопку **Заменить**.

Упражнение 3.6. Добавление тегов в проект Training. Проект Training1

Добавьте в проект **Training** аналогично тегу **Test** еще три тега переменных с атрибутами, показанными в табл. 3.7, и, для проверки, выполните компиляцию проекта.

Таблица 3.7. *Свойства добавленных тегов проекта Training*

Имя дескриптора	**Oven_Temp**	**Gas_Valve**	**Burner_Stat**
Устройства ввода/вывода		IODev	
Адрес	I0	D1	D2
Тип данных	INT	DIGITAL	
Мин. значение в единицах изм.	0		
Макс. значение в единицах изм.	130		
Единица измерения	Гр. Ц		
Формат	###		
Нечувствительность			
Примечание	Значение температуры в градусах Цельсия	Состояние задвижки подачи газа	Состояние горелки

Замечание

Информация об адресации тегов переменных приведена ранее в подразд. 3.2.

Совет

Не забывайте периодически сохранять проект на магнитном диске. Для этого в среде приложения **Проводник Citect** достаточно выбрать в окне **Список проектов** проект **Training** и выполнить команду **Создание резервной копии…** его контекстного меню. В появившемся окне **Создание резервной копии проекта** следует указать требуемые параметры и нажать кнопку **OK** (рис. 3.28). Проект сохраните под именем **Training1.** В дальнейшем работайте с проектом **Training1**. Для этого достаточно выбрать **Мои проекты** в окне **Список проектов** и выполнить команду **Восстановление…** его контекстного меню. В появившемся окне **Восстановить проект** следует указать требуемые параметры и нажать кнопку **OK** (рис. 3.29).

Созданные в проекте Vijeo Citect теги можно просматривать и редактировать в процессе исполнения проекта. Указанная возможность доступна только привилегированному пользователю.

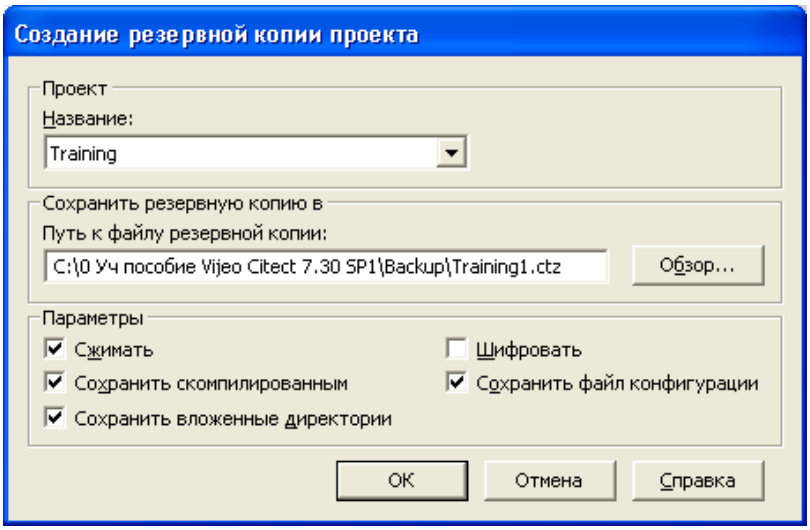


Рис. 3.28. Архивирование проекта **Training** под именем **Training1**

Привилегии (рис. 3.30) могут быть установлены в проекте таким образом, чтобы только определенные пользователи имели доступ к соответствующим командам и средствам управления.

Для этого необходимо, с одной стороны, предоставить пользователю определенный уровень привилегий (см. приведенный ранее рис. 2.6а), а с другой стороны, определить доступность команд и средств управления (рис. 3.31 и 3.32).

Упражнение 3.7. Просмотр и модификация тегов. Проект Training2

Просмотр и модификацию тегов модно выполнить с использованием графической страницы **CSV_AdminTools** из предустановленного включаемого проекта **CSV_Include**. Чтобы сделать эту страницу доступной, достаточно подключить к проекту **Training1** в качестве предустановленного включаемого проекта **CSV_Include** (рис. 3.33) и нажать кнопку **Добавить**.

Для тестирования работы с тегами выполните компиляцию, с помощью **Мастера конфигурирования компьютера** задайте к качестве стартовой (домашней) страницы страницу **CSV_AdminTools**, запустите проект **Training1** и "кликнув" левой кнопки мыши по любому управляющему элементу страницы **AdminTools** убедитесь, что из-за ограничений доступа элементы страницы недоступны.

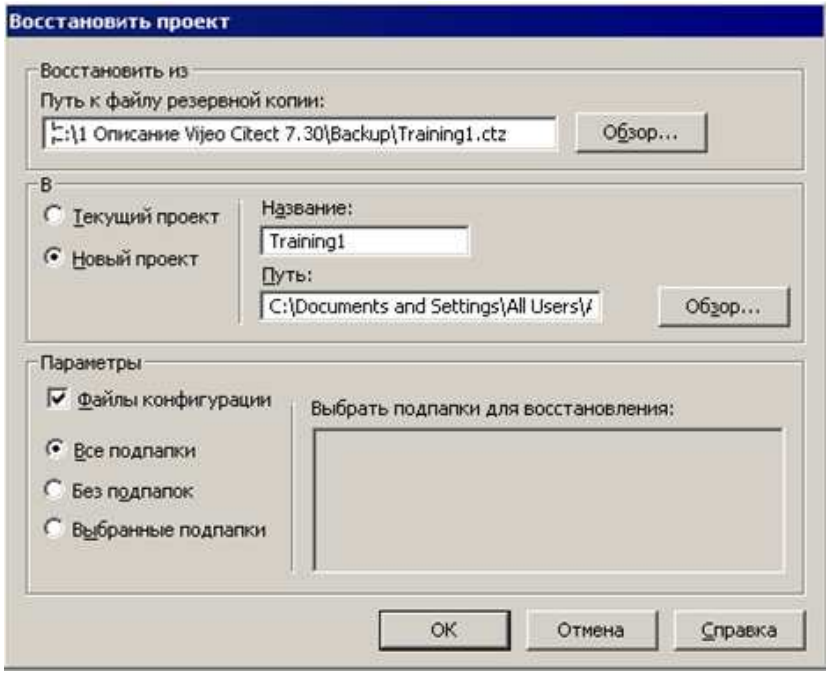


Рис. 3.29. Восстановление проекта **Training1**

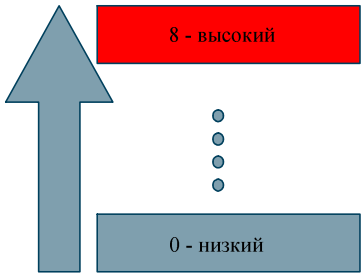


Рис. 3.30. Уровни привилегий в системе Vijeo Citect

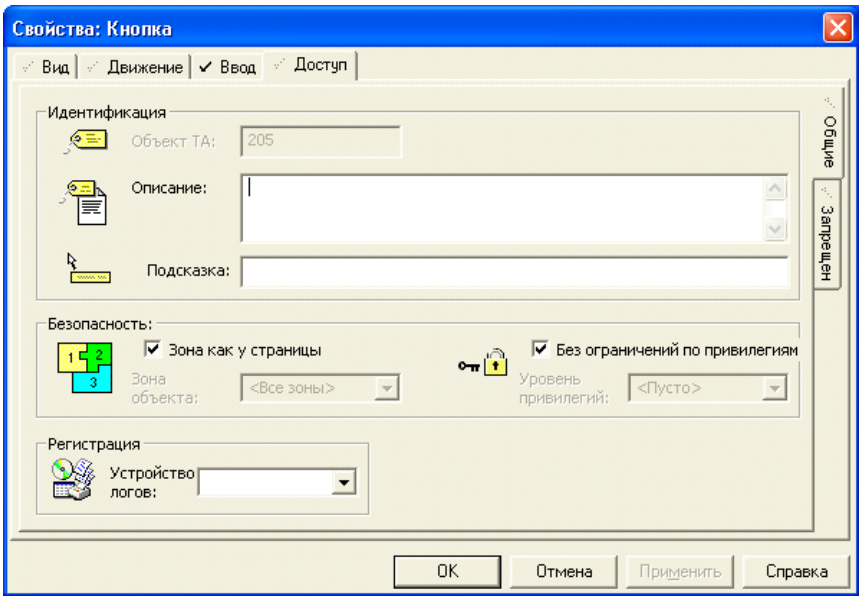


Рис. 3.31. Привилегии управляющего элемента по умолчанию

С помощью кнопки **Login\Logout** на панели инструментов зарегистрируйте ранее созданного привилегированного пользователя с именем **Engineer** и паролем **Engineer** (рис. 3.34) и нажмите кнопку **OK**. Теперь все управляющие элементы страницы доступны. Убедитесь в этом, нажав, например, кнопку **Tag Debug** (рис. 3.35). С помощью появившегося диалогового окна выберите нажатием кнопки **Browse** требуемый тег (рис. 3.36), используя кнопки **Read**, **Write** и окно **Tag Value** посмотрите или измените значение выбранного тега (рис. 3.37). Закройте проект.

Совет

Не забывайте периодически сохранять проект на магнитном диске. Для этого в среде приложения **Проводник Citect** достаточно выбрать в окне **Список проектов** проект **Training1** и выполнить команду **Создание резервной копии…** его контекстного меню. В появившемся окне **Создание резервной копии проекта** следует указать требуемые параметры и нажать кнопку **OK** (см. указанное ранее в упр. 3.6). Проект сохраните под именем **Training2.** В дальнейшем работайте с проектом **Training2**. Для этого достаточно выбрать **Мои проекты** в окне **Список проектов** и выполнить команду **Восстановление…** его контекстного меню. В появившемся окне **Восстановить проект** следует указать требуемые параметры и нажать кнопку **OK** (см. указанное ранее в упр. 3.6).

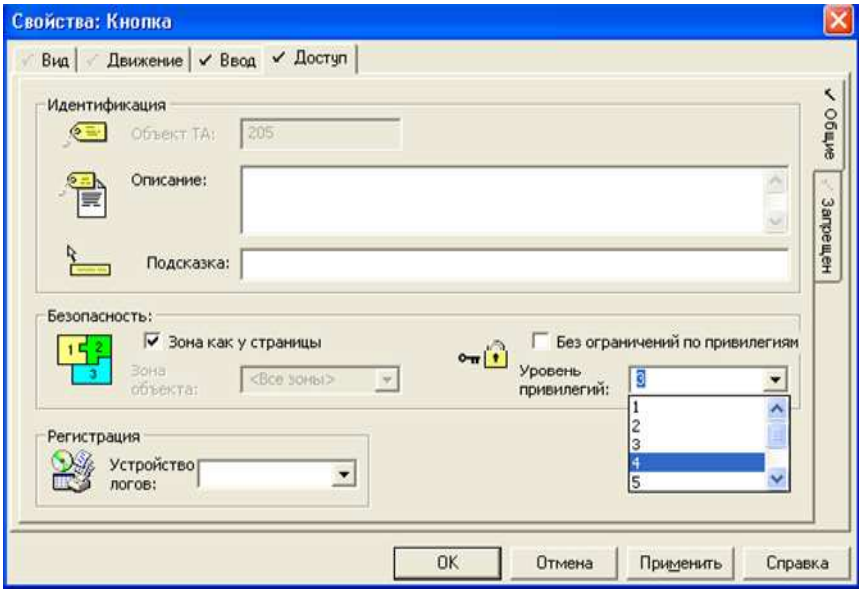


Рис. 3.32. Задание привилегий управляющего элемента

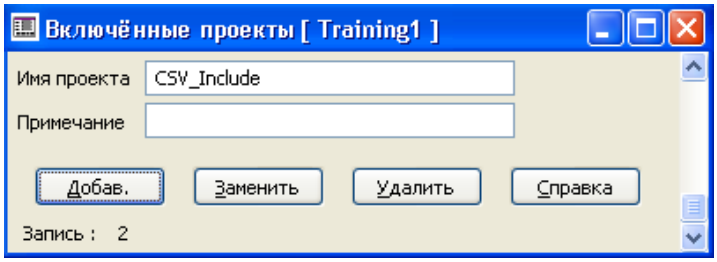


Рис. 3.33. Подключение предустановленного проекта **CSV_Include** к проекту **Training1**

Во многих случаях добавление или модификация тега переменной может быть трудоемкой задачей, особенно при наличии сотен или тысяч тегов со схожими именами. Поскольку все диалоговые окна Vijeo Citect созданы на основе файлов DBF, то возможно непосредственное редактирование тегов переменных в файле DBF с помощью приложения Microsoft Excel.

Совет

Прежде, чем продолжить работу, внимательно посмотрите материал, относящийся к редактированию тегов с помощью приложения Microsoft Excel в [3], слайды 65 — 67.

Рис. 3.34. Регистрация привилегированного пользователя

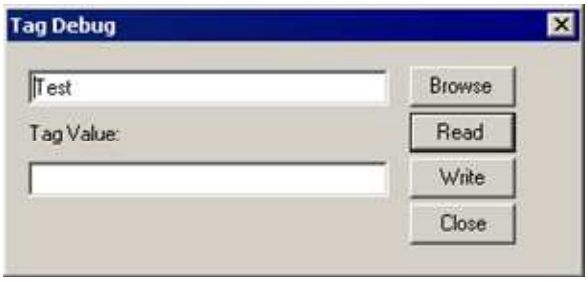


Рис. 3.35. Окно отладки тегов

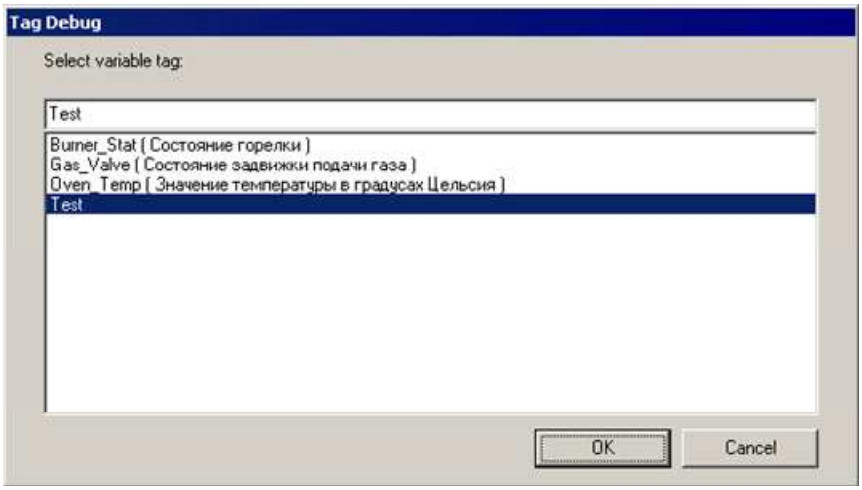


Рис. 3.36. Выбор тега для просмотра или модификации

Рис. 3.37. Просмотр или модификация значения тега

Упражнение 3.8. Редактирование тегов с помощью приложения Microsoft Excel

Выполните редактирование тегов с помощью приложения Microsoft Excel. В качестве проекта для демонстрации этой возможности используйте копию проекта **Training2**. С этой целью сохраните проект под именем **EditTagExcel1**, восстановите проект под этим именем и работайте с ним далее. В папке ..\Program Files\Schneider Electric\ Vijeo Citect 7.30\Bin "кликните" дважды левой кнопкой мыши по файлу **save_dbf.xls**. Запустится приложение Microsoft Excel. Таким образом будет загружен макрос Excel, необходимый для надлежащего сохранения файлов нужного формата.

> ***Примечание***
> Редактирование файлов за пределами Vijeo Citect может привести к невозможности их считывания приложениями Vijeo Citect. Поэтому перед редактированием файла Variable.dbf или любого другого dbf-файла всегда создавайте его копию. Тогда, в случае возникновения проблемы, можно восстановить базу данных из резервной копии.

Откройте файл **variable.dbf**, расположенный в папке проекта **EditTagExcel1** (..\User, тип файла — файлы dBase). Выполните команду **Сервис | Параметры**, выберите вкладку **Безопасность**, нажмите кнопку **Безопасность макросов**, во вкладке **Уровень безопасности** выберите **Низкая…** и дважды нажмите кнопку **ОК**. Скопируйте строку **Test** в свободную строку (отредактируйте поле **Name** скопированной строки на **TestExcel**; поле **ADDR** на **D3**, выполните команду **Формат | Ячейки…**, выберите закладку **Число**, в поле **Числовые форматы** выберите **Текстовый** и нажмите кнопку **ОК**, **D3** прижмется к левой границе поля; поле **OID** очистите). В идентификаторе тега цифры не используйте. Выполните команду **Сервис | Макрос**, выберите **Макросы…**. В окне **Макрос** в поле **Находится в:** должно быть **Все открытые книги**. Выберите **Save_dbf.xls!SaveDB** и нажмите кнопку **Выполнить**. Закройте приложение Microsoft Excel (на запросы последовательно нажмите кнопку **Да**, кнопку **Сохранить** и еще два раза нажмите кнопку **Да**).

Замечание
Важно! В среде **Редактора проектов Citect** выполните команду **Файл | Упаковать** (база данных будет проиндексирована, а все помеченные к удалению записи будут физически удалены) и обязательно выполните компиляцию.

На графической странице **NewPage** в настройках графических объектов замените теги **Test** на теги **TestExcel**. Сохраните графическую страницу, выполните компиляцию, запуск проекта, протестируйте графическую страницу и завершите работу проекта. Архивируйте проект.

Замечание
Важно! Редактирование тегов без проблем можно выполнить с помощью свободно распространяемого продукта OpenOffice.

3.5. OPC Factory Server (OFS). Связь контроллера Twido со SCADA-системой Vijeo Citect

Замечание
Важно! Материал, изложенный в этом подразделе, ориентирован на операционные системы Microsoft Windows XP/7. Для операционной системы Microsoft Windows 7 отличие состоит лишь в использовании другого инсталлятора драйвера кабеля связи с контроллером Twido (см. приложение 3, файлы **Инсталлятор\Communication Drivers\ SchneiderModbusDriverSuite.exe** и **Инсталлятор\Communication Drivers\ EngSetupDrv.pdf**)

Последовательно рассмотрим инсталляцию драйверов связи, их конфигурирование и тестирование, среды разработки программ для контроллера Twido, импорт и экспорт программ между средами разработки и контроллером и организацию связи между контроллером и SCADA-приложением Vijeo Citect.

Совет
Материал, относящийся к данной теме можно найти в [3], слайды 157 — 164.

OPC представляет собой OLE для управления процессами. *OPC* — это сокращение от *OLE for Process Control* (связывание и встраивание объектов для управления процессами), предназначенное для предоставления бизнес-приложениям доступа к данным объекта контроля единым, с точки зрения интерфейса образом. Стандарт OPC разработан фондом OPC Foundation и изложен в документе OPC Specification Version 1.0a и 2.0. Для совместимости с OPC, приложение должно быть реализовано с интерфейсом COM, описанным в документе OPC Specification.

Продукт *OFS (OPC Factory Server)* — это многоконтроллерный OPC-сервер, способный связываться с ПЛК фирмы Schneider Electric для поддержки OPC-клиентов, таких, как Vijeo Citect. Одной из младших моделей ряда ПЛК фирмы Schneider Electric является контроллер Twido.

OPC действует подобно общему языку. Поэтому разработчики аппаратного и программного обеспечения могут разрабатывать свои продукты и знать, что другие

продукты смогут взаимодействовать с ними. Целая армия передовых разработчиков программного и аппаратного обеспечения объединились с корпорацией Microsoft в работе над этим стандартом взаимодействия. Этот стандарт называется OPC, а организация, которая осуществляет управление этим стандартом, называется OPC Foundation. OPC основан на технологиях Microsoft: OLE (теперь ActiveX), COM (объектная модель компонентов) и DCOM (объектная модель распределенных компонентов). OPC состоит из стандартного набора интерфейсов, свойств и методов, используемых в приложениях для управления процессами и автоматизации производства. Технологии ActiveX/COM определяют, как отдельные компоненты приложения взаимодействуют и совместно используют данные. Говоря иначе, OPC используется для предоставления данных объекта управления из одной программы в другую, написанную кем-то другим.

Сервер OFS действует как шлюз между устройствами ввода-вывода (ПЛК фирмы Schneider Electric) и приложением (Vijeo Citect), которое хочет использовать их значения. В архитектуре COM каждый вид сервера получает уникальный идентификатор, который обозначается как ClassID (128-битное число). Для удобства, каждое такое число заменяется строковой ссылкой вида **Производитель.Приложение** с номером версии в качестве дополнения. Для сервера OFS строковая ссылка — **Schneider-Aut.OFS**.

5.5.1. Инсталляция драйвера кабеля связи с контроллером Twido

Примечание
Важно! Материал, изложенный в этом подразделе, ориентирован на операционную систему Microsoft Windows XP

Прежде чем начать установку драйвера связи контроллера и компьютера, необходимо произвести *установку драйвера для кабеля связи USB (TSXPCX3030),* с помощью которого контроллер подключается к локальному компьютеру. С этой целью соединяем *кабелем USB (TSXPCX3030)* контроллер и компьютер. При этом переключатель на кабеле связи д. б. установлен в положение 2. В правом нижнем углу экрана компьютера появится сообщение о найденном новом оборудовании, и появится окно мастера обновления оборудования (рис. 3.38).

Так как драйвер для кабеля находится на инсталляционном CD ROM **TwidoSoft v3.5**, то для отмены автоматического поиска операционной системой нужного драйвера выбираем **Нет, не в этот раз** и нажимаем кнопку **Далее.** В следующем окне мастера, выбираем **Установка из указанного места** (рис. 3.39) и нажимаем кнопку **Далее**.

Затем выбираем то место на установочном диске, где находится драйвер (рис. 3.40). С этой целью снимаем галочку около **Поиск на сменных носителях (дискетах, компакт-дисках...)**, устанавливаем галочку около **Включить следующее место поиска:**, нажатием кнопки **Обзор** выбираем местоположение драйвера и нажимаем последовательно на кнопки **ОК** и **Далее** (см. приведенный ранее рис. 3.40).

Далее начинается процесс установки драйвера, во время которого появляется окно, предупреждающее о том, что устанавливаемое программное обеспечение не тестировалось на совместимость с операционной средой, в данном случае с ОС Windows XP (рис. 3.41). Нажимаем на кнопку **Всё равно продолжить**. По окончании установки нажимаем на кнопку **Готово.**

После установки драйвера повторно появится мастер установки нового оборудования и необходимо повторить изложенные ранее действия ещё раз, начиная с приведенного ранее рис. 3.38.

Для проверки правильности установки драйвера можно выполнить следующее. Для ярлыка **Мой компьютер** на рабочем столе выполнить команду **Свойства** контекстного меню. Появится окно **Свойства системы** (рис. 3.42), в нем нужно выбрать вкладку **Оборудование** и нажать кнопку **Диспетчер устройств**. В результате появится окно с одноименным названием, в котором нужно раскрыть пункт **Контроллеры универсальной последовательной шины USB** (рис. 3.43). *После установки драйвера компьютер следует перегрузить.*

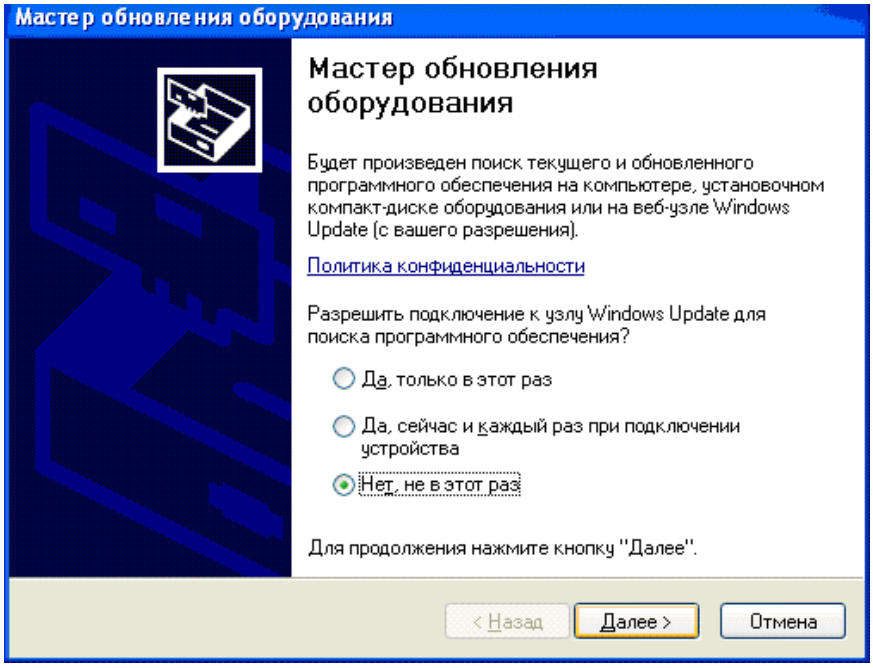


Рис. 3.38. Мастер обновления оборудования

Замечание

Если *драйвер связи контроллера Twido и персонального компьютера* (**Schneider Modbus Serial Driver**) еще не был установлен, то по окончании установки

драйвера кабеля связи необходимо его установить. Методика инсталляции драйвера связи контроллера Twido и компьютера и тестирования связи приведена в приложении 1.

3.5.2. Конфигурирование драйвера связи персонального компьютера с контроллером Twido

Для *конфигурирования драйвера связи* выполняем следующее. Если **MODBAS Driver** еще не был запущен, то его следует *запустить* командой **Пуск | Программы | Schneider Electric | Communication Drivers | MODBAS Driver**. Выбираем **Schneider Modbas Serial Driver** и выполняем команду **Configure** его контекстного меню. Во вкладке **Configuration** появившегося окна **MODBAS Driver - MODBUS01**, устанавливаем параметры соединения в соответствии с рис. 3.44. Остальные вкладки окна оставляем без изменений. Нажимаем на кнопку **OK**.

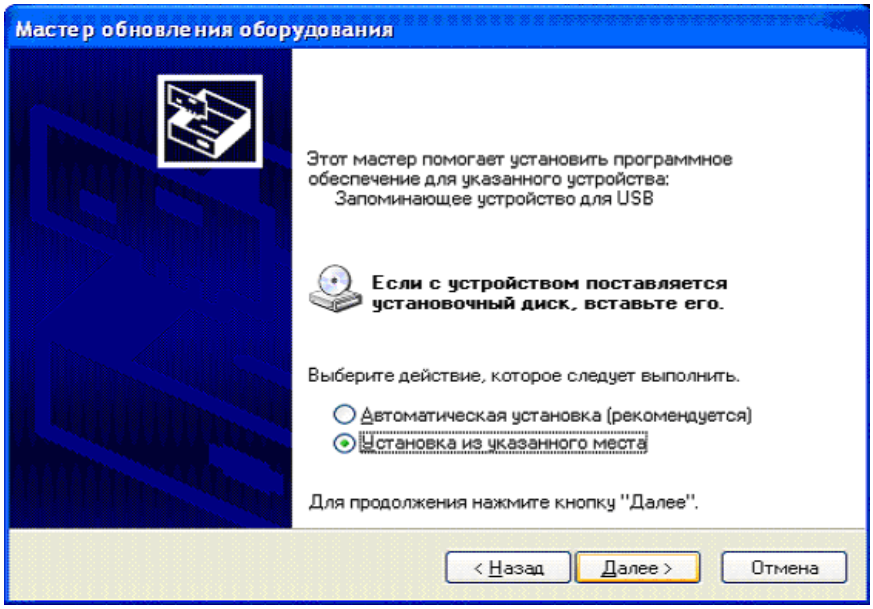


Рис. 3.39. Мастер обновления оборудования

Совет

В приложении 2 приведены краткие сведения о среде разработки программ для контроллера **Twido — TwidoSoft**, конфигурировании контроллера Twido, создании новой программы и экспорте-импорте программ между средой разработки и контроллером Twido. Настоятельно советуем рассмотреть указанный материал — это очень полезно.

3.5.3. Конфигурирование OFS, создание клиента OFS и тестирование связи с контроллером Twido (на примере программы Example для контроллера)

Далее последовательно рассмотрим конфигурирование OFS, создание клиента OFS и тестирование связи с контроллером Twido как с помощью клиента OFS, так и с помощью SCADA-приложения Vijeo Citect (на примере программы Example, см. приведенное далее приложение 2).

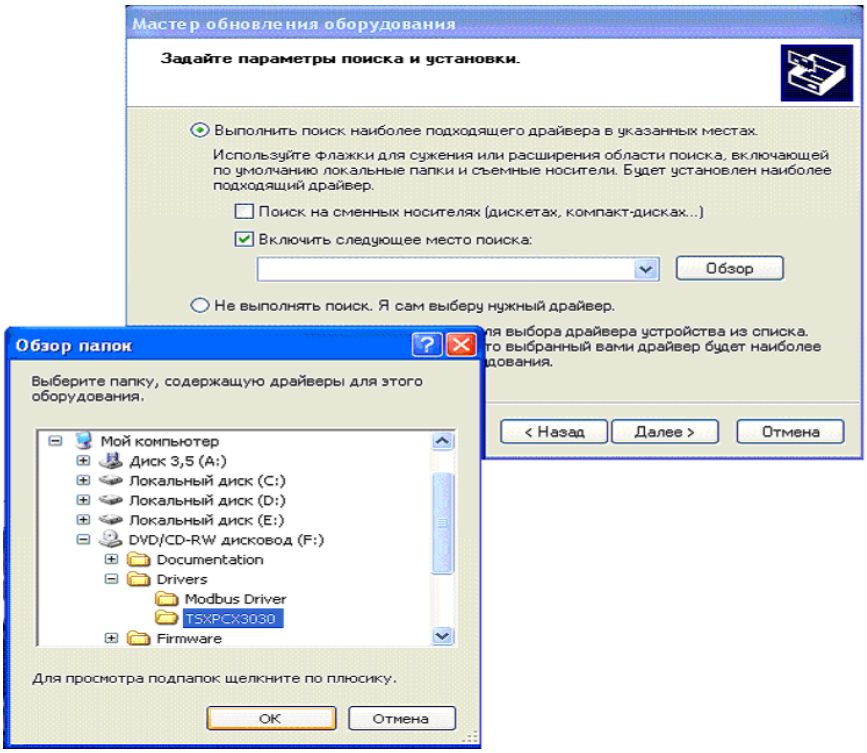


Рис. 3.40. Поиск папки драйвера кабеля USB

3.5.3.1. Конфигурирование OFS

Для запуска приложения **OFS Configuration Tool** выполняем команду **Пуск | Программы | Schneider Electric | SoCollaborative | Ofs | OFS Configuration Tool**. Появившееся окно настраиваем в соответствии с рис. 3.45. В поле **Device name** указываем **TwidoOFS**. Для заполнения поля **Device address** нажимаем кнопку просмотра**,** расположенную в правой части поля. Появившееся окно

Device address wizard заполняем в соответствии с рис. 3.46 и нажимаем кнопку **OK**. Для заполнения поля **Symbol table file**, задающего символьный файл описания переменных проекта, также используем кнопку просмотра. Как указано далее в приложении 2, этот файл был создан при настройке проекта и модифицирован для использования в сервере OFS.

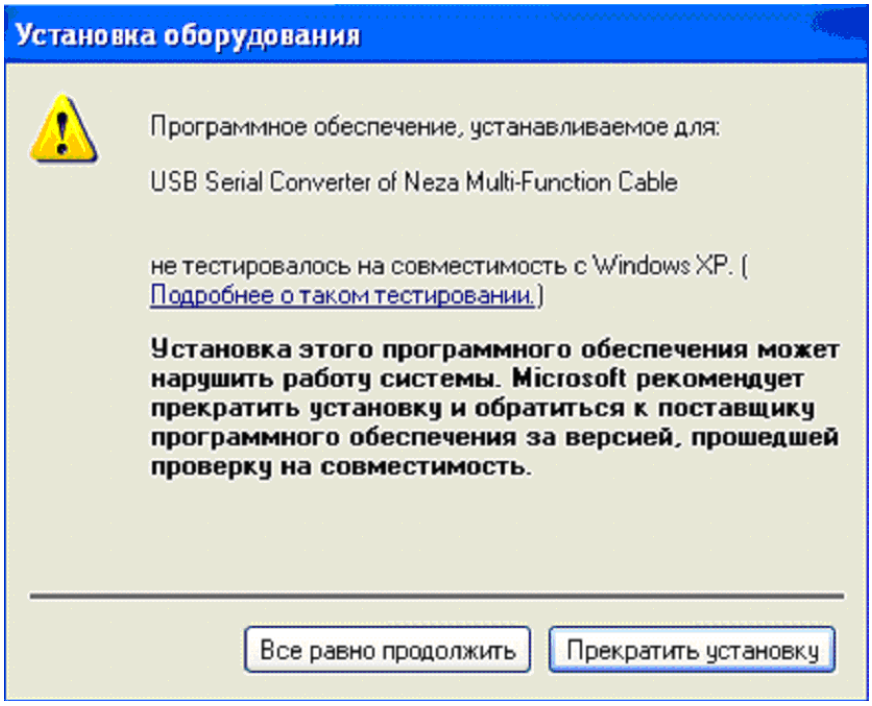


Рис. 3.41. Предупреждение

Конфигурации для остальных вкладок, расположенных в левом поле главного окна приложения **OFS Configuration Tool**, должны соответствовать рис. 3.47 – 3.51. Для сохранения заданных параметров выполняем команду **File | Save Configuration,** после чего откроется окно (рис. 3.52), говорящее о том, что для вступления в силу внесённых изменений обходимо перегрузить OFS-сервер, если к этому моменту сервер уже был запущен. Нажимаем кнопку **OK** и закрываем окно приложения **OFS Configuration Tool.**

3.5.3.2. Создание клиента OFS

Перед созданием клиента OFS нужно определиться будет ли клиентом использоваться приложение **OPC Factory Server Simulation** или **OPC Factory Server**. В первом случае подключение контроллера не требуется, будет

происходить симуляция работы OFS-сервера, а во втором случае требуется организация связи инструментальной ЭВМ и контроллера. В первом случае приложение **OPC Factory Server Simulation** необходимо запустить до начала создания клиента с помощью команды **Пуск | Программы | Schneider Electric | SoCollaborative | Ofs | OFS Factory Server Simulation**. Во втором случае приложение **OPC Factory Server** запустится автоматически.

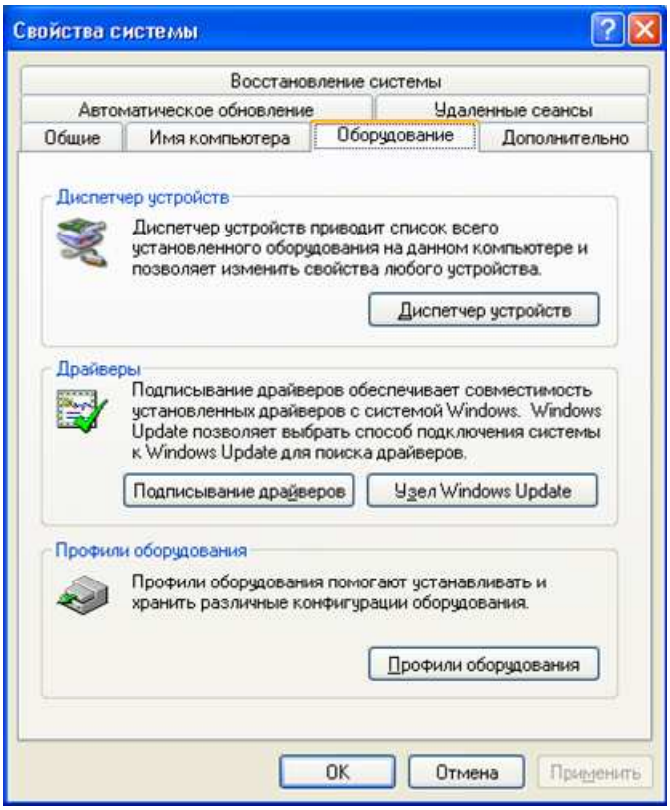


Рис. 3.42. Окно **Свойства системы**

Для запуска приложения **OFS Client** выполняем команду **Пуск | Программы | Schneider Electric | SoCollaborative | Ofs | OFS Testing Client | OFS Client**. В появившемся окне (рис. 3.53) выбираем сервер **Schneider-Aut.OFS** и нажимаем кнопку **OK**. Далее в окне клиента необходимо создать новую группу и добавить в неё переменные из символьного файла описания переменных. Для этого в окне клиента выполняем команду **Group | New Group** (рис. 3.54) и нажимаем кнопку **OK**. Для добавления новых элементов в группу выполняем команду **Item | New…**

(рис. 3.55), выбираем **TwidoOFS** и появится список символьных переменных, который был создан при создании проекта **Example.twd**. В списке переменных последовательно добавляем переменную **M1**, нажимаем кнопку **ОК**, переменную **MW1** и нажимаем **ОК** (рис. 3.56).

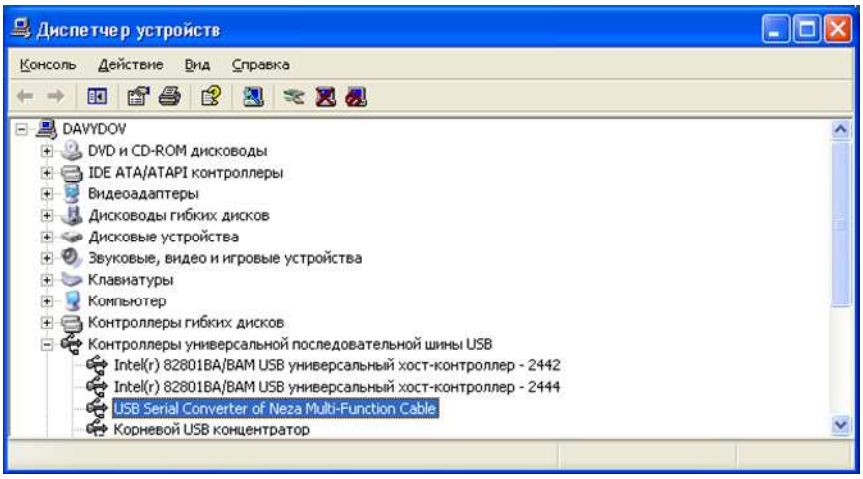


Рис. 3.43. USB Serial Converter of Neza Multi-Function Cable в окне **Диспетчер устройств**

Правильность проделанных действий можно проверить двумя способами, что зависит от предварительного выбора приложения **OPC Factory Server** или **OPC Factory Server Simulation**. При использовании приложения **OPC Factory Server** в контроллер должен быть загружен и запущен проект **Example.** Для проверки переменной **M1** переключаем дискретный вход, моделируемый первым тумблером слева на линейке тумблеров и смотрим в окне клиента на последнюю запись в поле **Value** (для **M1** чередуются значения **False**-**True**, а для переменной **MW1** отображается значение 2 после установки второго слева тумблера во включенное состояние). При использовании приложения **OPC Factory Server Simulation** можно видеть периодические изменения значений переменных (для **M1** чередуются значения **False**-**True**, а для переменной **MW1** значение увеличивается на единицу до 99 с последующим сбросом в ноль**)**. Вид окна клиента при контроле изменения значений переменных контроллера представлен на рис. 3.57.

3.5.3.3. Сервер OFS. Связь SCADA-системы Vijeo Citect с контроллером Twido

Для тестирования связи Vijeo Citect с контроллером Twido создаём в среде приложения **Проводника Citect** новый проект. Для этого используем кнопку **Новый** на панели инструментов, или команду **Файл | Новый проект…**, или же команду

Новый проект… контекстного меню любой компоненты окна **Список проектов**. Сохраняем созданный проект под именем **TwidoOFS**.

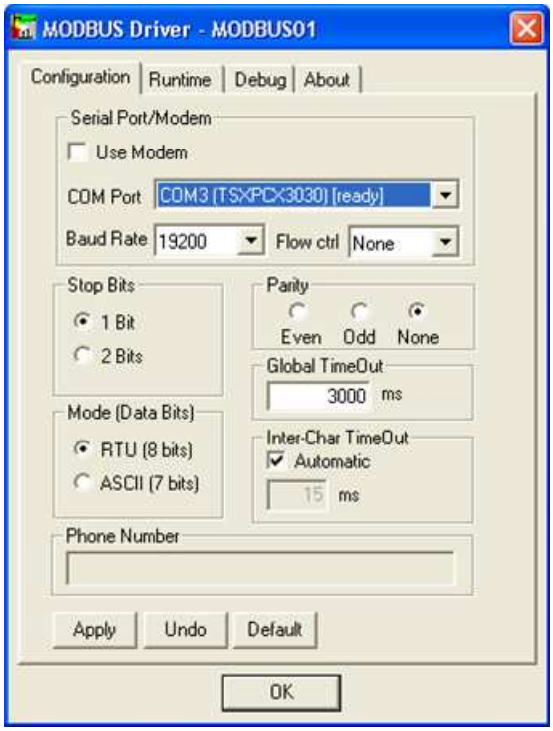


Рис. 3.44. Окно конфигурирования драйвера связи

В среде **Редактора проектов Citect** с помощью меню **Сервера** конфигурируем **Кластеры и Сервера ввода/вывода** в соответствии с рис. 3.58. С помощью команды **Связь | Мастер быстрой настройки** настраиваем связь контроллера с сервером ввода-вывода (табл. 3.8).

В среде **Проводника Citect** выполняем команду **Средства | Импорт тегов…**, настраиваем окно **Импорт переменных тегов** в соответствии с рис. 3.59 и нажимаем кнопку **Импорт**.

Таблица 3.8. *Настройка связи контроллера с сервером ввода-вывода*

Номер сценария, окно	**Параметры и действия**
1. Мастер экспресс-настройки параметров связи	Нажать кнопку **Далее**

Номер сценария, окно	Параметры и действия
2. Мастер экспресс-настройки параметров связи	I/O Server Name: **IOSvr**. Нажать кнопку **Далее**
3. Мастер экспресс-настройки параметров связи	I/O Device Name: **IODev**. Нажать кнопку **Далее**
4. Мастер экспресс-настройки параметров связи	Выбрать **Внешнее устройство в/в**. Нажать кнопку **Далее**
5. Мастер экспресс-настройки параметров связи	Производитель: **OPC Foundation**. Модель: **OPC DA Client**. Средства связи: **OPC**. Нажать кнопку **Далее**
6. Мастер экспресс-настройки параметров связи	В поле **Адрес:** ввести **Schneider-Aut.OFS** и нажать кнопку **Далее**
7. Мастер экспресс-настройки параметров связи	Нажать кнопку **Далее**
8. Мастер экспресс-настройки параметров связи	Нажать кнопку **Готово**

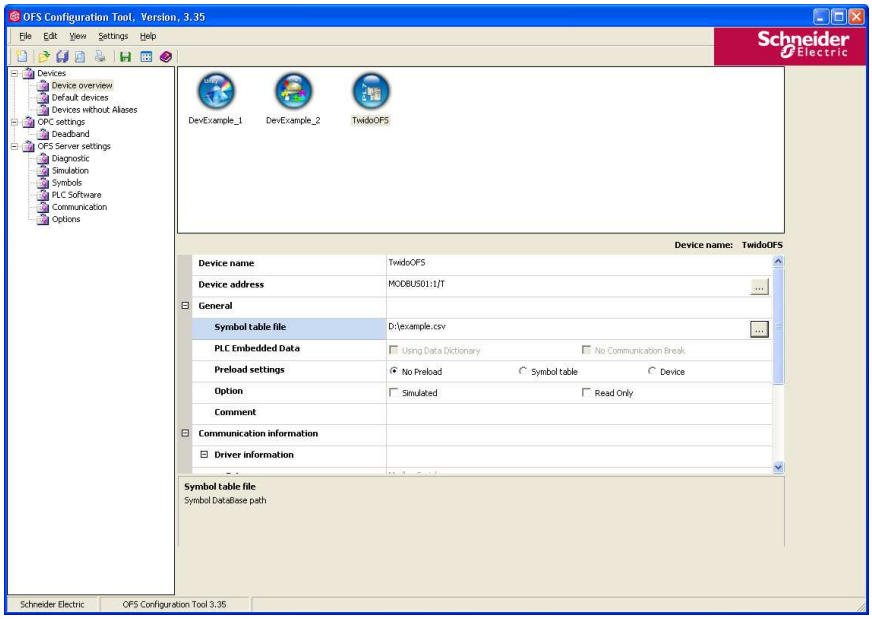


Рис. 3.45. Вид приложения **OFS Configuration Tool** после запуска и конфигурирования вкладки **Devices | Device overview**

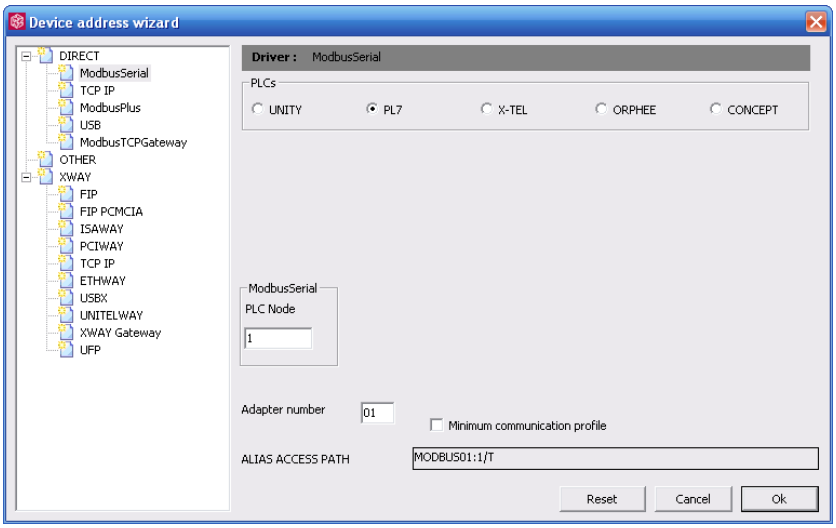


Рис. 3.46. Конфигурирование окна **Device address wizard**

Создаем новую графическую станицу в среде приложения **Построитель графики Citect** с помощью кнопки **Новый** на панели инструментов или команды **Файл | Новый…**. В появившемся окне **Новый** нажимаем на кнопку **Страница**. В результате появится следующее окно **Использовать шаблон**, представленное на рис. 3.60.

Используем параметры шаблона графической страницы, показанные на рис. 3.60, и нажимаем кнопку **OK**. Для сохранения созданной страницы под именем **Test** нажимаем кнопку **Сохранить** на панели инструментов, или выполняем команду **Файл | Сохранить,** или же активизируем акселератор **Ctrl+S**. В появившемся окне **Сохранить как** в поле **Страница:** вкладки **Страница** указываем имя графической страницы **Test** и нажимаем кнопку **OK**.

Помещаем в созданную страницу графический объект **Эллипс** и конфигурируем его в соответствии с рис. 3.61. Аналогично помещаем в созданную страницу графический объект **Число** и конфигурируем его в соответствии с рис. 3.62.

Сохраняем графическую страницу, в среде **Редактора проектов Citect** выполняем команды **Файл | Упаковать**, **Файл | Компилировать**, **Файл | Выполнить** и тестируем проект (рис. 3.63). Для проверки переменной **M1** переключаем дискретный вход, моделируемый первым тумблером слева на линейке тумблеров. При этом чередуются цвета заполнения эллипса и текст под ним отображает значение **2**, что свидетельствует о правильном обмене информацией с контроллером.

Сохраняем созданный проект под прежним именем **TwidoOFS**.

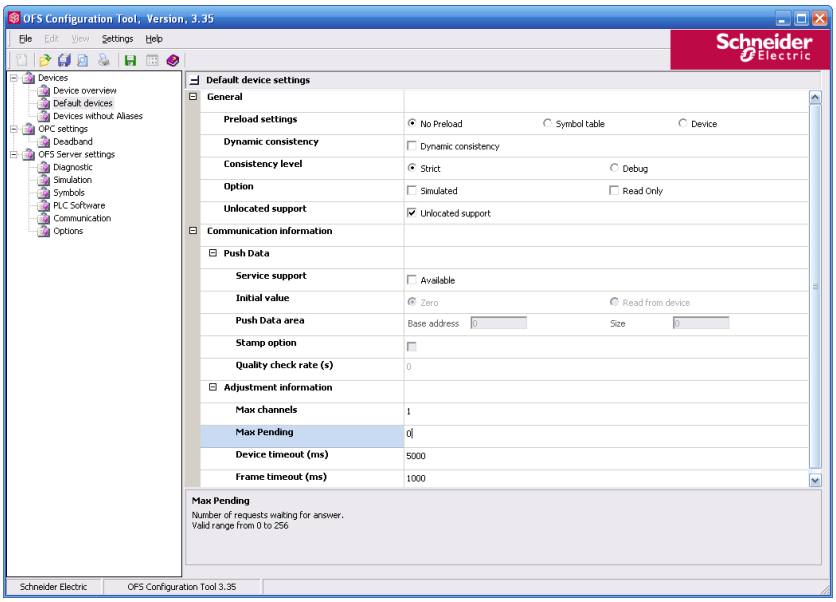


Рис. 3.47. Вкладка **Devices | Default Devices**

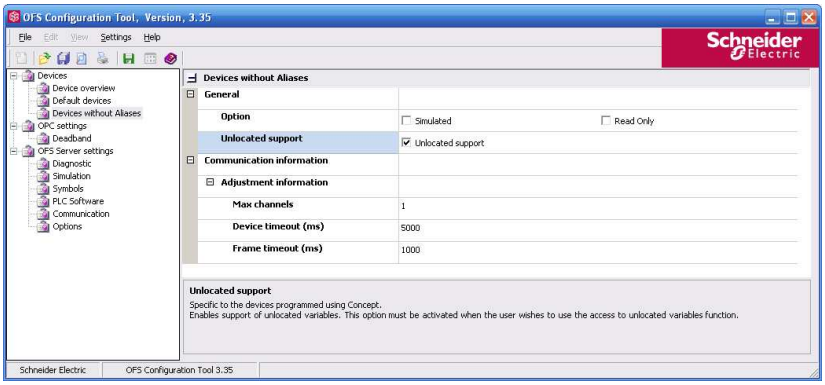


Рис. 3.48. Вкладка **Devices | Devices without Aliases**

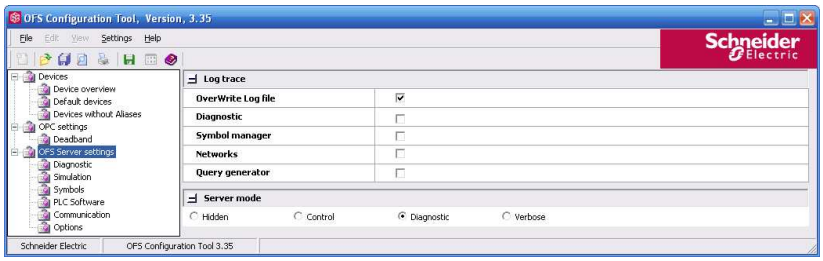


Рис. 3.49. Вкладка **OFS Server settings**

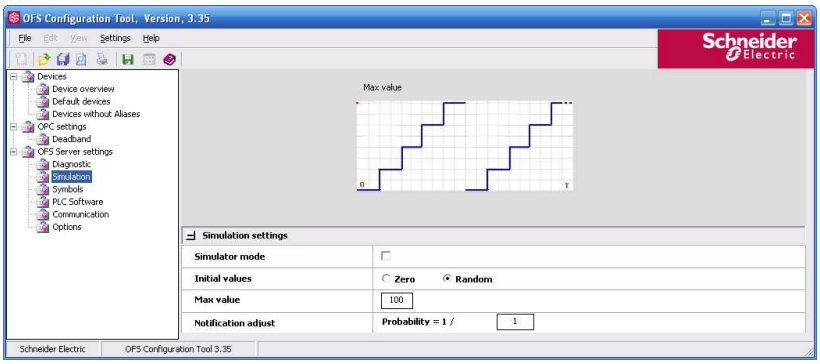


Рис. 3.50. Вкладка **OFS Server settings | Simulation**

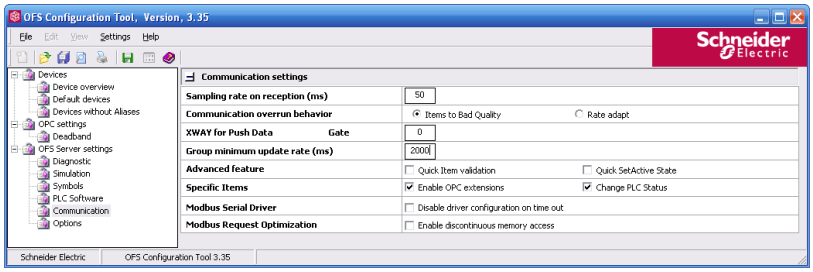


Рис. 3.51. Вкладка **OFS Server settings | Communication**

Глава 4. Графика

Графические страницы являются одним из основных компонентов системы Vijeo Citect. Они являюся интерфейсным средством операторов и могут предусматривать как отображение данных, так и ввод данных от оператора. Графическая страница имеет шаблон, объекты, нарисованные на странице, и присущие графической странице свойства.

Рис. 3.52. Сообщение необходимости перегрузки OFS-сервера

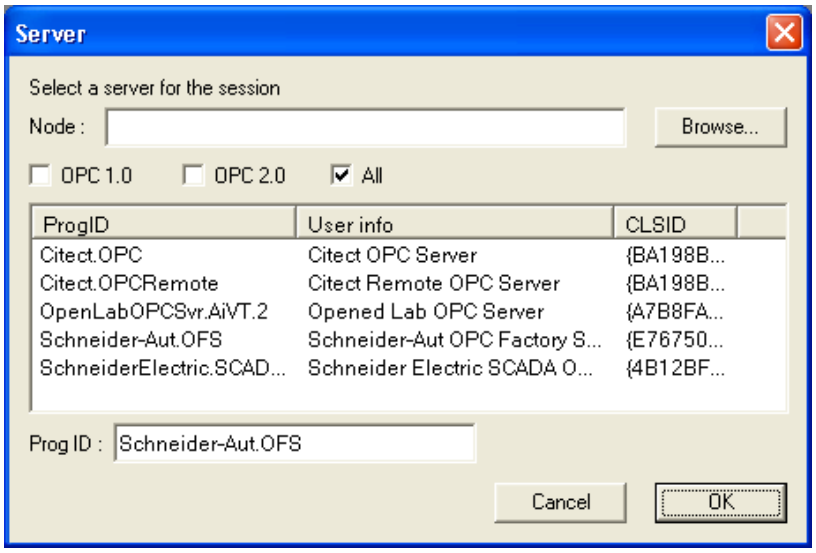


Рис. 3.53. Выбор сервера для приложения **OFS Client**

Совет

Прежде, чем продолжить работу, внимательно прочтите материал, относящийся к графическим объектам и их использованию в [2], темы Using Vijeo Citect | Defining and Drawning Graphics Pages | Using Objects, Using Vijeo Citect | Defining and Drawning Graphics Pages | Understanding Object Types, Using Vijeo Citect | Defining and

Drawning Graphics Pages | Defining Common Object Properties, Using Vijeo Citect | Defining and Drawning Graphics Pages | Defining Commands And Controls, Using Vijeo Citect | Genie And Super Genie; [3], слайды 71 — 118, 125 — 136, 140 — 146.

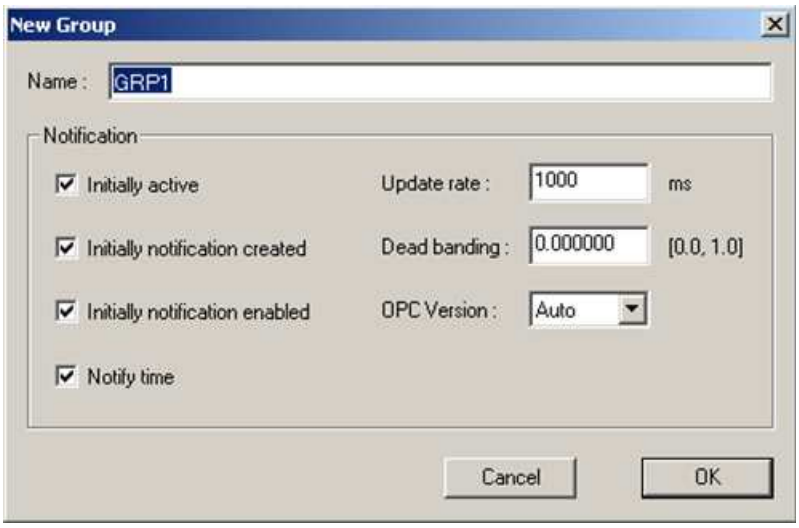


Рис. 3.54. Создание новой группы

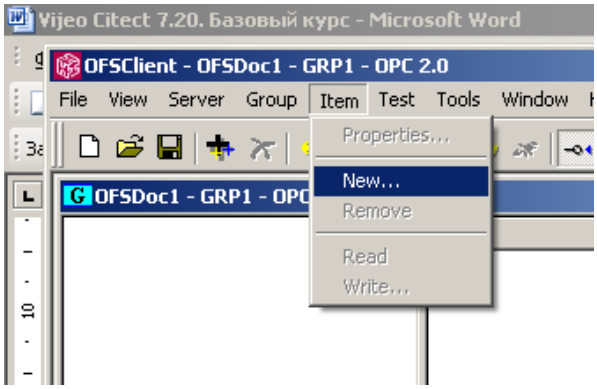


Рис. 3.55. Выбор **Aliase**

При создании проекта новые страницы можно разрабатывать сообразно требуемому стилю. В основе простейшей страницы лежит шаблон **Blank**, представляющий пустое окно. Разработчик может добавлять в это окно объекты и функциональность и разрабатывать новые шаблоны для страниц своего проекта. Для разработчиков,

ограниченных сжатыми сроками проектирования, а также для неопытных пользователей системы Vijeo Citect, существуют предварительно разработанные шаблоны, которые позволяют быстро создавать страницы.

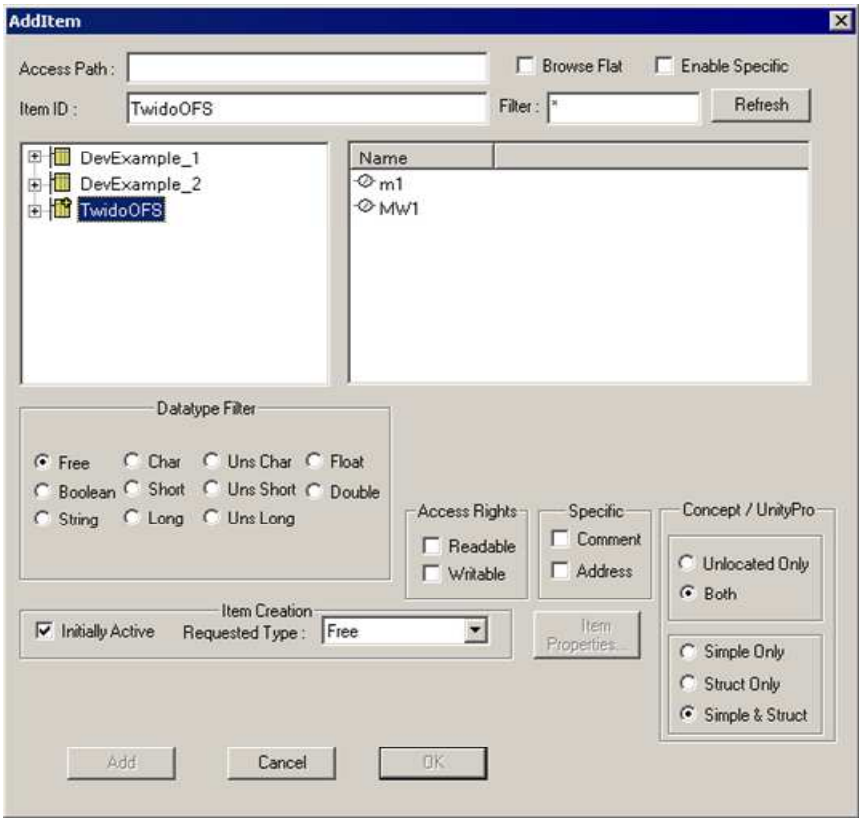


Рис. 3.56. Добавление переменных в созданную группу

Как указывалось ранее, включаемые проекты **Tab_Style_Include** и **CSV_Include** являются предварительно сконфигурированными проектами. Они предназначены для сокращения времени, необходимого для настройки нового проекта, и содержат шаблоны и страницы, выполненные в стиле операционной системы Windows XP. При создании нового проекта проект **Tab_Style_Include** автоматически входит в него как включаемый проект. Это означает, что все шаблоны проекта и его другие средства доступны для использования при создании графических страниц.

Наряду с шаблонами стандартной графики, для создания технологических графических страниц, в проект входят предопределенные страницы трендов и

сигналов тревог, страница средств администрирования, страницы файлов для демонстрации текстовых файлов, а также разнообразные всплывающие окна. Все они оснащены обычными меню и панелями инструментов для навигации, панелями инструментов для навигации сигналов тревог, которые обеспечивают единство функциональности и внешнего вида графических страниц в пределах всего проекта (рис. 4.1 и приведенный ранее рис. 3.26).

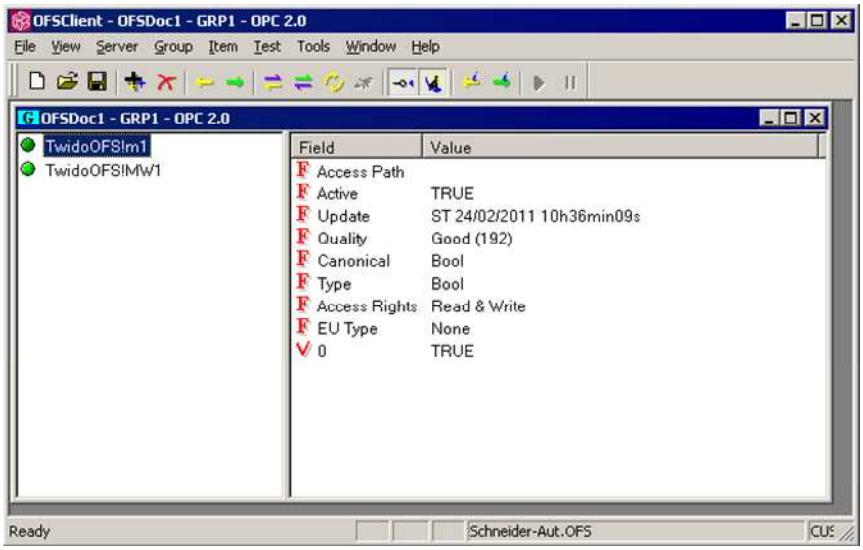


Рис. 3.57. Окно клиента OFS после конфигурирования

Специализированная строка меню содержит ряд предопределенных меню, каждое их которых содержит предопределенные команды. На этапе исполнения строка меню может модифицироваться в соответствии с потребностями конкретного проекта, о чем будет сказано далее.

Панель инструментов навигации содержит навигационные кнопки и кнопки прямого доступа к основным страницам (трендов, сигналов тревог, администрирования и др.).

Панель инструментов сигналов тревог обеспечивает доступ к страницам сигналов тревог и отображает, по крайней мере, три активных страницы сигналов тревог.

В большинстве проектов создаются шаблоны, разработанные под конкретный объект.

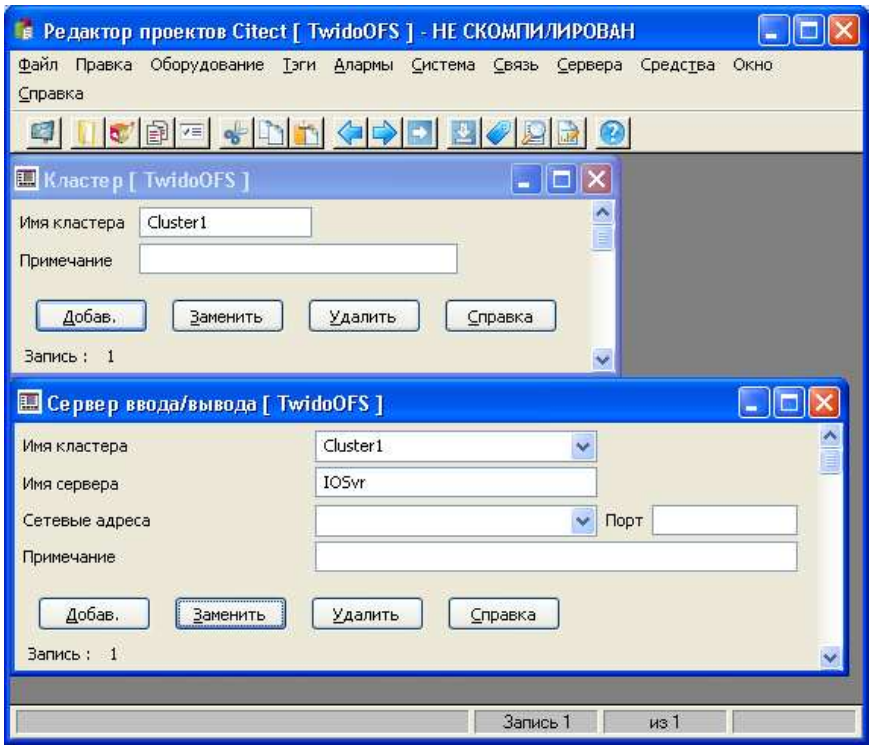


Рис. 3.58. Конфигурирование кластера и серверов

Совет

Более подробная информация о шаблонах графических страниц имеется в [2], тема Using Vijeo Citect | Defining and Drawing Graphics Pages | Using Page Templates. Познакомьтесь с этой информацией.

Создание новой графической страницы возможно одним из трех способов. В среде приложения **Проводник Citect** в окне **Список проектов** выбрать и раскрыть проект, раскрыть компонент **Графика**, выбрать компонент **Страницы**, в окне **Содержимое Страницы** "кликнуть" два раза левой кнопкой мыши по компоненту **Создать новую страницу**. В среде **Построителя графики Citect** нажать кнопку **Новый** на панели инструментов или выполнить команду **Файл | Новый…**.

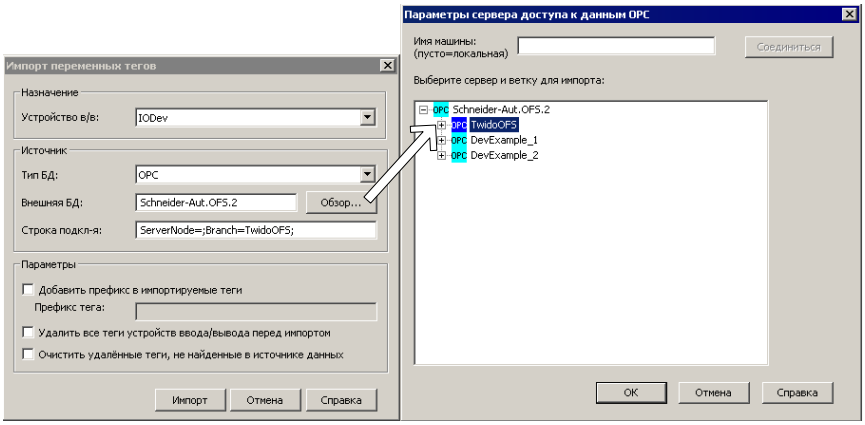


Рис. 3.59. Импорт тегов переменных

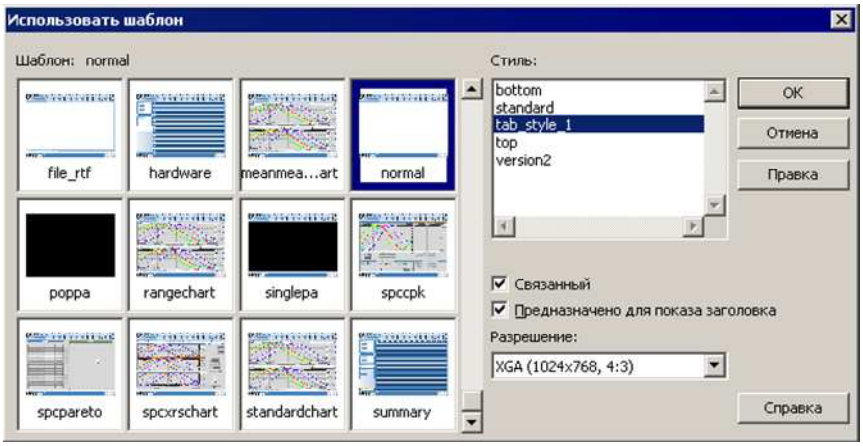


Рис. 3.60. Задание параметров шаблона графической станицы

Объекты на странице можно рисовать, выбрав объекты из панели инструментов (см. приведенный ранее рис. 3.17) или из меню **Объекты**. Действия по рисованию (размещению в графическом окне) каждого объекта несколько отличаются. Конкретную информацию о том, как рисовать каждый объект можно найти в [2], тема
Using Vijeo Citect | Defining and Drawing Graphics Pages | Understanding Object Types.

Совет

Для изменения размеров графических объектов страницы пользуйтесь маркерами (прямоугольными метками), появляющимися по контуру объекта после его выбора.

Маркеры можно переместить в новую позицию, используя левую кнопку мыши. Ряд полезных советов при работе с графическими объектами приведен в [3] на слайде 79.

Объектами можно манипулировать с помощью меню **Правка**, **Вид**, **Текст** и **Порядок**. Как и в других пакетах для рисования, графические объекты можно вращать, масштабировать и выравнивать.

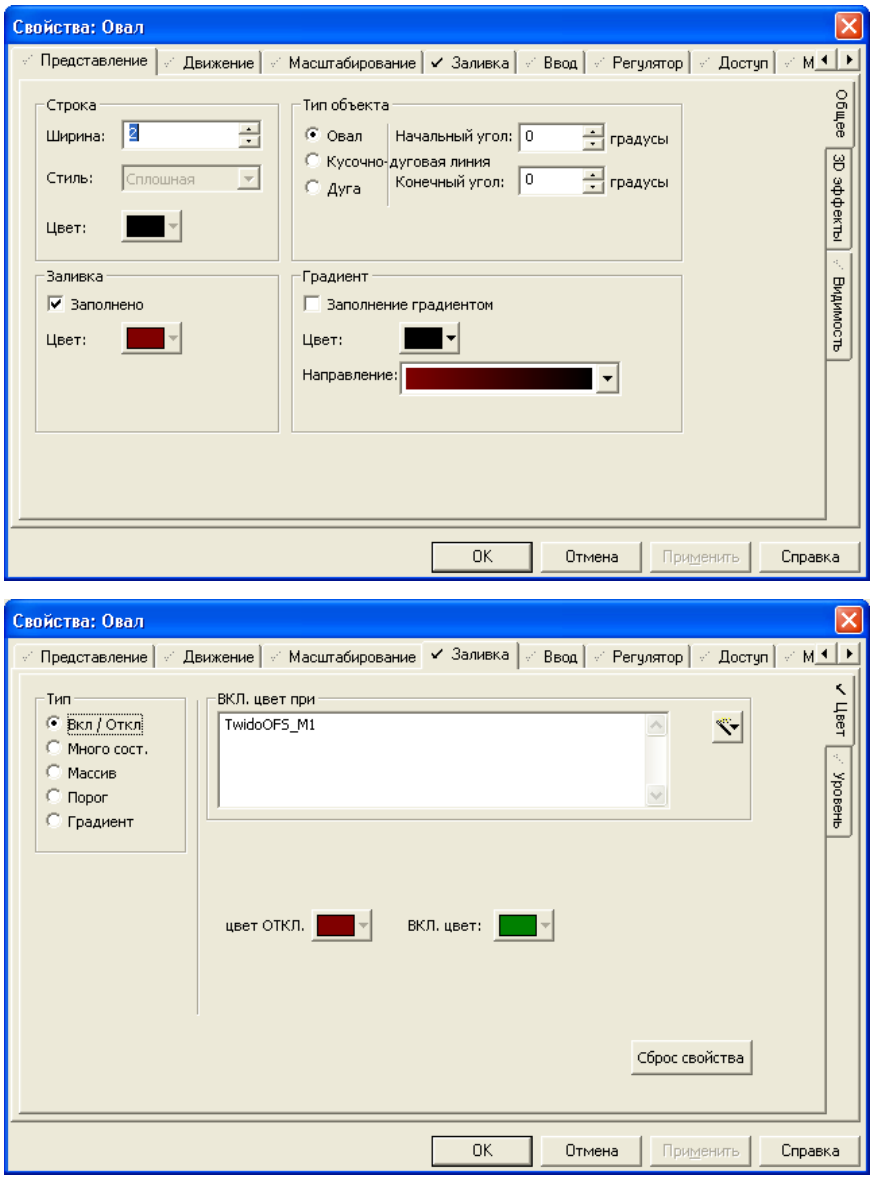


Рис. 3.61. Свойства графического объекта **Эллипс**

Примечание
Диалоговые окна свойств объектов рассмотрены ранее в упражнении 3.4. Более подробные сведения о свойствах графических объектов содержатся в [2], тема Using Vijeo Citect | Defining and Drawing Graphics Pages | Using Objects | Objects Properties.

4.1. Рисование графической страницы проекта Training2

Первым этапом настройки графической страницы после создания страницы и определения ее свойств является рисование графических объектов на странице. Графические объекты, которые следует нарисовать, и их примерное расположение на новой странице показаны на рис. 4.2. Эти объекты представляют температурную печь (духовку, oven).

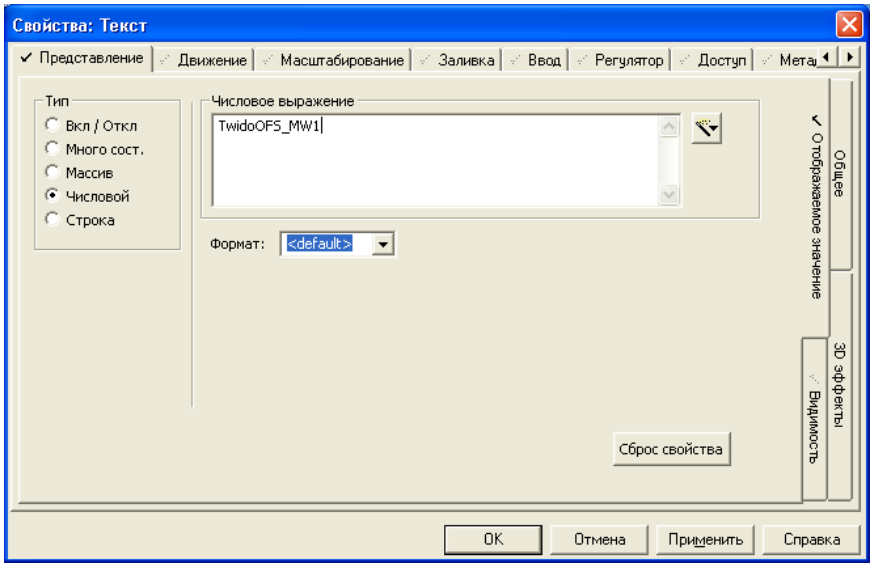


Рис. 3.62. Свойство графического объекта **Number**

Упражнение 4.1. Создание новой графической страницы Oven

В качестве проекта используйте проект **Training2**. Аналогично упр. 3.3 создайте новую графическую страницу. При создании новой графической страницы используйте шаблон страницы с параметрами, указанными в табл. 4.1. Сохраните страницу под именем **Oven**.

Совет

Регулярно сохраняйте создаваемую страницу — не ждите, пока страница будет полностью создана! Пользуйтесь меню **Порядок** приложения **Построитель графики Citect** — его команды часто оказываются весьма полезными. Если нужно отменить последнее выполненное действие, то нажимайте кнопку **Откат** на панели инструментов или выполняйте команду **Правка | Откат**. Для удобства работы с графическими объектами установите в странице режим показа сетки — выполните команду **Вид | Настройка сетки…**, в окне **Настройка сетки** установите свойство **Отобразить сетку** и нажмите кнопку **OK**.

Упражнение 4.2. Размещение в графической странице Oven символа Tank

Поместите символ **Tank** в графическую страницу. Для этого в **Окне объектов** приложения **Построитель графики Citect** (см. приведенный ранее рис. 3.17) нажмите кнопку **Вставить символ**, в появившемся окне **Вставить символ** выберите библиотеку **tanks_conb_30** и символ **tank** (рис. 4.3), нажмите кнопку **OK** и поместите символ в нужном месте страницы. Для того, чтобы повернуть символ на 180 градусов, в среде приложения **Построитель графики Citect** выполните команду **Порядок | Повернуть…**, в появившемся окне **Вращение** выберите дважды **По часовой стрелке** и нажмите кнопку **OK**. Снабдите добавленный символ поясняющим текстом **Gas Tank** (см. приведенный ранее рис. 4.2), аналогично тому, как это делалось в упр. 3.4.

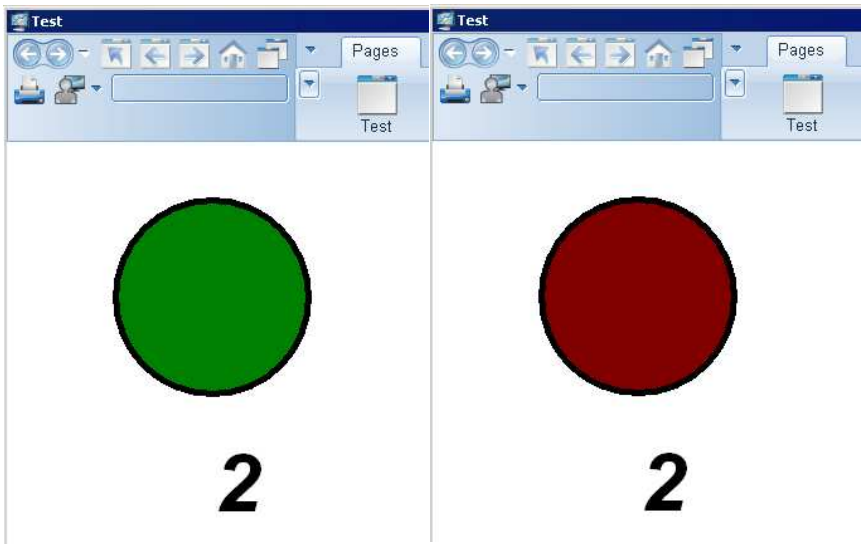


Рис. 3.63. Тестирование приложения

Рис. 4.1. Типовой внешний вид графической страницы, основанной на шаблоне **Normal** из включаемого проекта **CSV_Include**

Таблица 4.1. *Параметры графической страницы Oven*

Параметр	**Значение**
Стиль	tab_style_1
Разрешение	XGA
Шаблон	normal
Связанный	✓
Предназначено для показа заголовка	✓
Цвет фона	Серый (задайте в свойствах страницы)

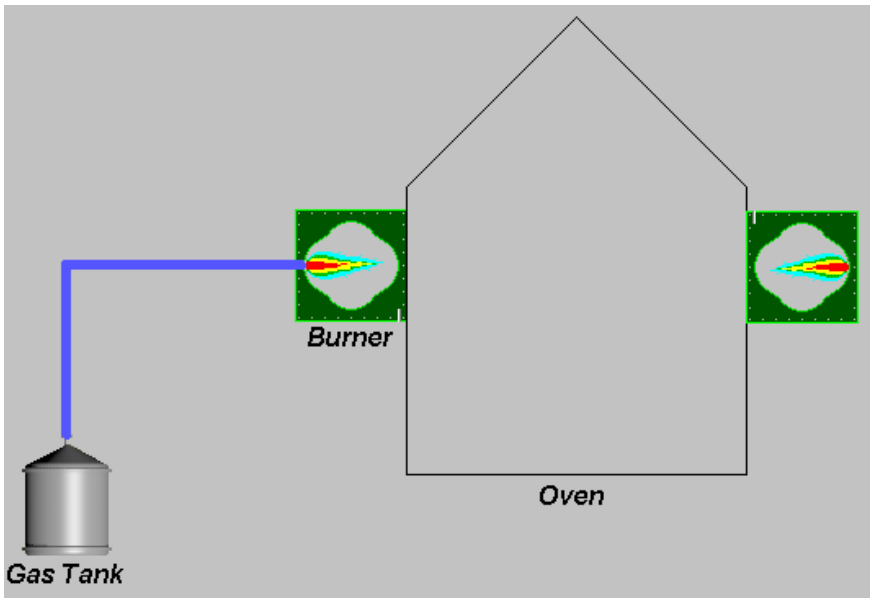


Рис. 4.2. Примерный вид графической страницы

Совет

Не забудьте еще раз сохранить страницу — не ждите, пока страница будет полностью создана! Если сразу же после размещения графического объекта открывается диалоговое окно настройки его свойств, просто нажмите кнопку **OK**, если не требуется изменять свойства, задаваемые по умолчанию. Чтобы легче было рисовать, включите режим **Привязать к сетке**. Для его включения выполните команду **Вид | Настройка сетки**, в появившемся окне отметьте **Привязать к сетке** и нажмите кнопку **OK**. Если при выборе действия в меню **Правка**, **Вид**, **Текст** или **Порядок** открывается диалоговое окно, то нажмите кнопку **Справка**, чтобы получить дополнительную информацию по этому действию.

Упражнение 4.3. Размещение в графической странице Oven набора символов Burner

Поместите набор символов **Burner** в графическую страницу. Для этого в **Окне объектов** приложения **Построитель графики Citect** (см. приведенный ранее рис. 3.17) нажмите кнопку **Набор образов**, переместите курсор в нужное место страницы (см. приведенный ранее рис. 4.2), нажмите левую кнопку мыши, в появившемся окне **Свойства: Набор образов** нажмите кнопку **Установить**, в окне **Выбрать образ** выберите библиотеку **misk2** и символ **flame001** (рис. 4.4).

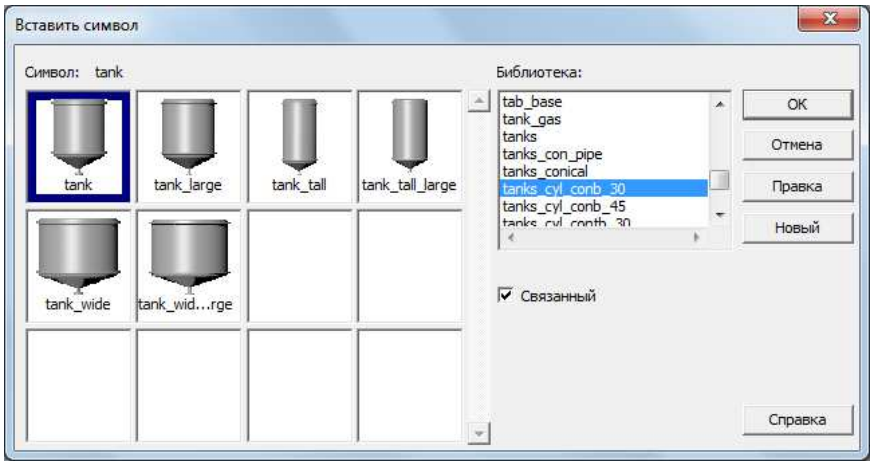


Рис. 4.3. Выбор объекта (символа) **tank**

Чтобы в этом символе убрать изображение пламени, нажмите кнопку **Правка**, в появившемся окне выберите символ, выполните команду **Средства | Редактор растровых изображений**, в появившемся окне **Редактор растровых изображений** уберите изображение пламени, нажмите кнопку ✓ на панели инструментов и окно **Редактор растровых изображений** закроется. Для сохранения откорректированного изображения символа нажмите кнопку **Сохранить** на панели инструментов и закройте текущее окно. Далее в окне **Oven** приложения **Построитель Графики Citect** выполните команду **Свойства…** контекстного меню объекта **Набор образов**, нажмите кнопку **Задать…** справа от поля **Символ ОТКЛ**, выберите библиотеку **misk2** и символ **flame001** (см. приведенный ранее рис. 4.4)**,** дважды нажмите кнопку **OK** и поместите символ в нужном месте страницы. Снабдите добавленный набор символов поясняющим текстом **Burner** (см. приведенный ранее рис. 4.2), аналогично тому, как это делалось в упр. 3.4.

> ***Совет***
> Не забудьте снова сохранить страницу — не ждите, пока страница будет полностью создана!

Упражнение 4.4. Размещение в графической странице многоугольника Oven

Поместите многоугольник **Oven** в графическую страницу (см. приведенный ранее рис. 4.2). Для этого в **Окне объектов** приложения **Построитель графики Citect** (см. приведенный ранее рис. 3.17) нажмите кнопку **Полигон**, отпустите кнопку мыши, переместите мышь в узел (место) страницы, где должна находиться вершина многоугольника. Нажмите левую кнопку мыши и, не отпуская ее, переместите мышь в узел следующей вершины и отпустите кнопку мыши. Аналогично постройте

остальные стороны многоугольника, а в последней вершине дважды "щелкните" левой кнопкой мыши. Появится окно **Свойства: Полигон**, которое следует сразу же закрыть, нажав кнопку **OK**. Снабдите добавленный многоугольник поясняющим текстом **Oven** (см. приведенный ранее рис. 4.2), аналогично тому, как это делалось в упр. 3.4.

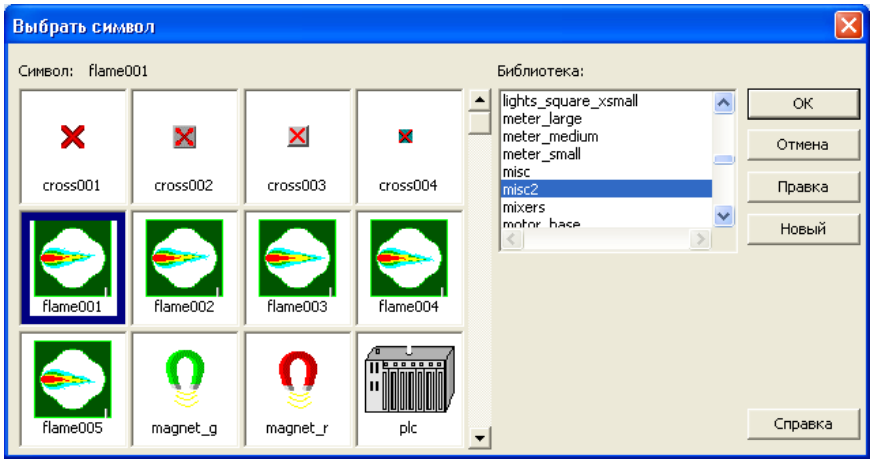


Рис. 4.4. Выбор набора символов **Burner**

Совет

Не забудьте снова сохранить страницу — не ждите, пока страница будет полностью создана!

Упражнение 4.5. Размещение трубопровода в графической странице Oven

Поместите трубопровод в графическую страницу. Для этого в **Окне объектов** приложения **Построитель графики Citect** (см. приведенный ранее рис. 3.17) нажмите кнопку **Труба**, отпустите кнопку мыши, переместите мышь в узел (место) страницы, где должно находиться начало трубопровода, нажмите левую кнопку мыши и, не отпуская ее, переместите мышь в узел следующего изгиба трубопровода и отпустите кнопку мыши. Аналогично выполните остальные участки трубопровода, а в конце трубопровода дважды "щелкните" левой кнопкой мыши. Появится окно **Свойства: Труба**, которое следует настроить в соответствии с рис. 4.5 и нажать кнопку **OK**.

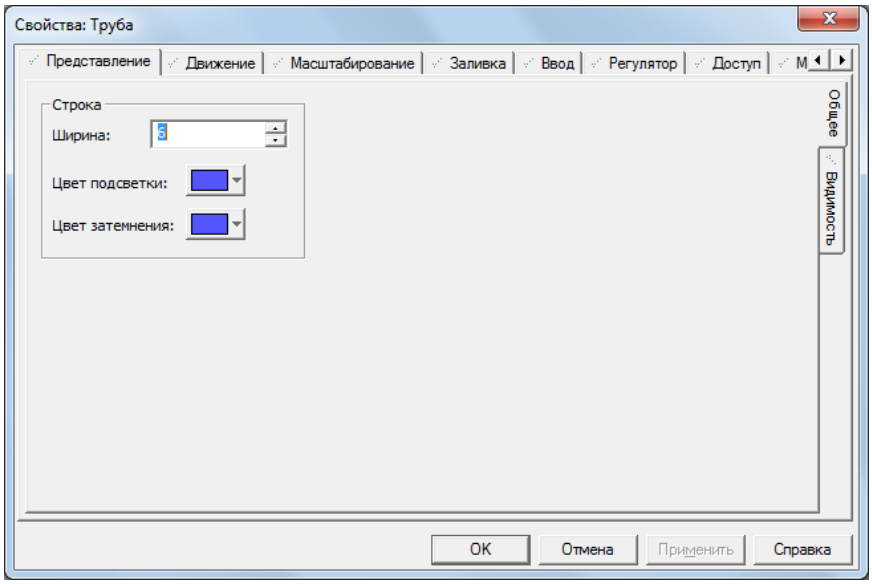


Рис. 4.5. Свойства трубопровода

Рассмотренные ранее графические объекты **Полигон**, **Труба**, а также графические объекты **Прямоугольник** и **Эллипс** можно перемещать, изменять их размеры и форму, переносить на передний план и т. д. Каждый из указанных графических объектов состоит из непрерывной серии линий, соединяющих вершины. Каждая вершина изображается в виде маленького квадратика. Формы рассмотренных ранее графических объектов можно изменять многими способами.

Для выделенного графического объекта вершины можно выбирать по отдельности или группой и перемещать в другое место, изменяя таким образом форму графического объекта. Для добавления вершины достаточно выбрать графический объект, установить курсор в требуемое место линии и нажать клавишу **Insert**. Для удаления вершины достаточно выбрать графический объект, выбрать вершину и нажать клавишу **Delete**.

Упражнение 4.6. Размещение в графической странице Oven еще одного набора символов Burner

Поместите еще один набор символов **Burner** в графическую страницу с другой стороны многоугольника. Для этого выделите существующий набор символов **Burner**, скопируйте его и вставьте в страницу с помощью кнопок **Копировать** и **Вставить** на панели инструментов приложения **Построитель графики Citect**, поверните копию на 180 градусов так, это делалось для символа **Gas Tank** и разместите его так, как показано ранее на рис. 4.2.

Совет
Не забудьте снова сохранить страницу — не ждите, пока страница будет полностью создана!

Упражнение 4.7. Просмотр созданной графической страницы Oven. Проект Training3

Сохраните, компилируйте и запустите проект. Чтобы новую страницу можно было увидеть, выполните команду **Pages | Oven**. Протестируйте выполненную модификацию страницы **Oven** и завершите работу проекта.

Совет
Не забывайте периодически сохранять проект на магнитном диске. Сохраните проект **Training2** под именем **Training3.** В дальнейшем работайте с проектом **Training3**.

4.2. Цветовая анимация графических объектов во время выполнения проекта

Графические объекты страницы могут иметь динамические свойства (например, цвет, размер и положение), которые могут изменяться во время выполнения проекта, отражая изменения условий управляемого процесса. В частности, цвет заполнения полигона **Духовки** (**Oven**) был бы идеальным способом передачи оператору значения температуры **Духовки**. Этот аспект вначале и будет рассмотрен далее.

Упражнение 4.8. Цветовая анимация графического объекта

Определите свойства объекта **Oven** так, чтобы при изменении температуры менялся цвет заполнения полигона.

В качестве проекта используйте проект **Training3**. В среде **Построитель графики Citect** на странице **Oven** двойным щелчком левой кнопки мыши на объекте (полигоне) **Oven** откройте диалог **Свойства: Полигон**, во вкладке **Представление (Общий)** отметьте кнопку **Заполнено**. Во вкладке **Заливка (Цвет)** выберите тип анимации **Массив**, в поле **Выражение массива** задайте выражение **Oven_Temp/25**, определяющее индекс элемента массива, в поле **Цвета массивов** определите цвета заполнения многоугольника, соответствующие различным значениям индекса **Oven_Temp/25**. Во вкладке **Доступ (Общие)** в поле **Подсказка** задайте текст всплывающей подсказки **Цвет зап. мен. в завис. от знач. тега Oven_Temp** и нажмите кнопку **OK** (рис. 4.6).

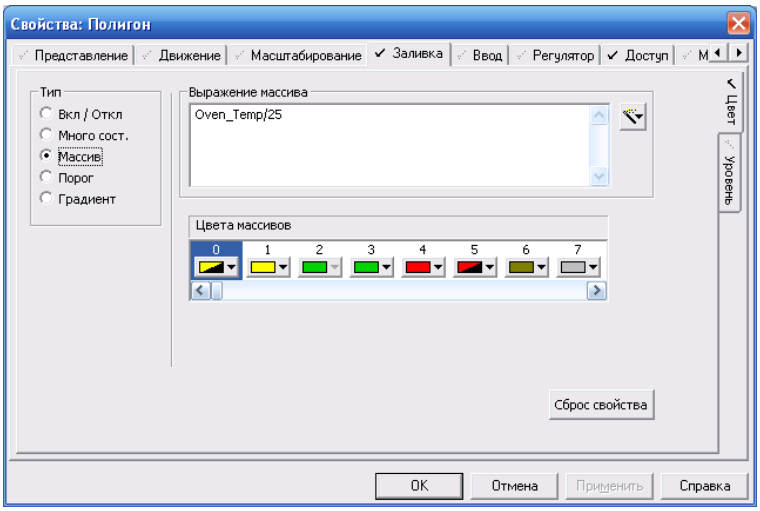


Рис. 4.6. Анимация объекта **Oven** (полигона) цветом заполнения

Замечание

Поскольку значение тега переменной **Oven_Temp** изменяется в диапазоне от 0 до 130 градусов, то значение индекса **Oven_Temp/25** изменяется в диапазоне от 0 до 5. Следовательно, в поле **Цвета массивов** (см. приведенный ранее рис. 4.6) следует определить цвета заполнения для первых шести элементов массива цветов. При значениях тега **Oven_Temp** между 0 и 12 градусами будет использован мерцающий черно-желтый цвет заполнения, между 13 и 37 градусами — желтый, между 38 и 87 градусами — зеленый, между 88 и 112 градусами — красный и при значении более 113 градусов — мерцающий черно-красный цвета заполнения.

Совет

При записи выражения в поле **Выражение массива** (см. приведенный ранее рис. 4.6) пользуйтесь **Мастером выражений** (кнопка его активизации расположена справа). Это позволит избежать опечаток в записи идентификаторов тегов или функций (процедур). Использование **Мастера Выражений** иллюстрирует рис. 4.7. Сохраните модифицированную страницу **Oven**, выполните компиляцию и запустите проект. Зарегистрируйтесь как привилегированный пользователь, откройте страницу **CSV_AdminTools**, командой **Tag Debug** откройте диалог **Tag Debug**, с его помощью задавайте различные значения тега **Oven_Temp**, предварительно вернувшись на страницу **Oven**, и наблюдайте за изменениями цветов заполнения многоугольника **Oven**. Завершите работу проекта.

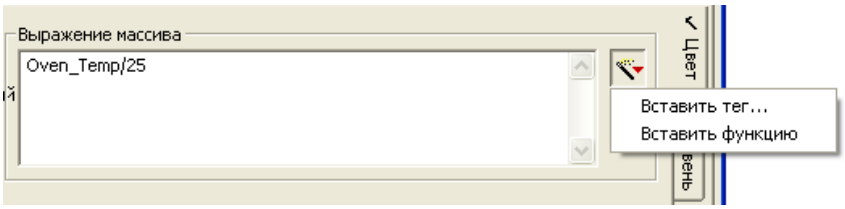


Рис. 4.7. Использование **Мастера Выражений**

4.3. Отображение гистограммы во время выполнения проекта

Одним из динамических свойств графических объектов является свойство, задаваемое во вкладке **Заливка (Уровень)** диалогового окна свойств графического объекта. Если заполнение применяется для прямоугольного объекта, то можно создать гистограмму для отображения изменяемой величины. Например, можно нарисовать гистограмму и использовать свойство **Заливка (Уровень)** для отображения температуры печи.

Упражнение 4.9. Использование гистограммы и предопределенного объекта Джин (Genie)

Создайте **Гистограмму** для отображения температуры **Духовки** в виде уровня заполнения прямоугольника.

Поместите объект **Прямоугольник** в графическую страницу. Для этого в **Окне объектов** приложения **Построитель графики Citect** (см. приведенный ранее рис. 3.17) нажмите кнопку **Прямоугольник** и разместите объект **Прямоугольник** в графической странице **Oven** в соответствии с рис. 4.8. Сразу же появится окно **Свойства: Прямоугольник**, которое следует настроить в соответствии с рис. 4.9 и нажать кнопку **OK**.

Снабдите добавленную гистограмму поясняющим текстом **Oven_Temp** (см. рис. 4.8), аналогично тому, как это делалось ранее.

> ***Совет***
> Не забудьте снова сохранить страницу — не ждите, пока страница будет полностью создана!

Для изменения значения температуры **Духовки** добавьте в графическую страницу предопределенный графический объект **Джин**. Для этого в **Окне объектов** приложения **Построитель графики Citect** (см. приведенный ранее рис. 3.17) нажмите кнопку **Вставить джин**. В появившемся диалоге **Вставить джин** в поле **Библиотека** выберите библиотеку **controls**, в поле **Джин — Ramp Up…tn2** и нажмите кнопку **OK.** В следующем диалоге **Genie — Ramp Up Down Button**

выберите тег переменной **Oven_Temp** и нажмите кнопку **OK** (рис. 4.10). Разместите объект **Джин** в графической странице **Oven** в соответствии с рис. 4.11.

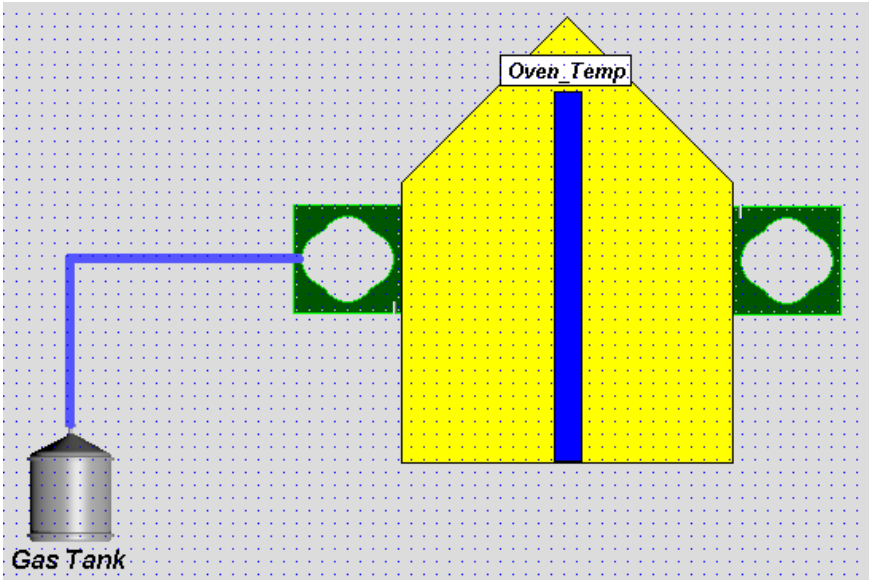


Рис. 4.8. Размещение объекта-гистограммы в графической странице **Oven**

Совет

Сохраните модифицированную страницу **Oven**, выполните компиляцию и запустите проект. С помощью **Genie — Ramp Up Down Button** задавайте различные значения тега **Oven_Temp** и наблюдайте за изменениями цветов заполнения многоугольника **Oven**. Убедитесь в наличии всплывающих подсказок для графических объектов **Духовка**, **Гистограмма** и **Джин.** Завершите работу проекта.

4.4. Отображение числовых значений во время выполнения проекта

Вы можете отображать значение любого тега или выражения во время выполнения проекта как число. Если значение тега или выражения изменяется, то отображающее его число на графической странице автоматически обновляется.

Упражнение 4.10. Отображение числовых значений

Используйте графический объект **Число** для отображения значения тега переменной **Oven_Temp** на странице **Oven** проекта **Training3**. В **Окне объектов** приложения **Построитель графики Citect** (см. приведенный ранее рис. 3.17) нажмите кнопку

Число и разместите объект **Число** в графической странице **Oven** в соответствии с рис. 4.12. Сразу же появится диалоговое окно **Свойства: Текст**.

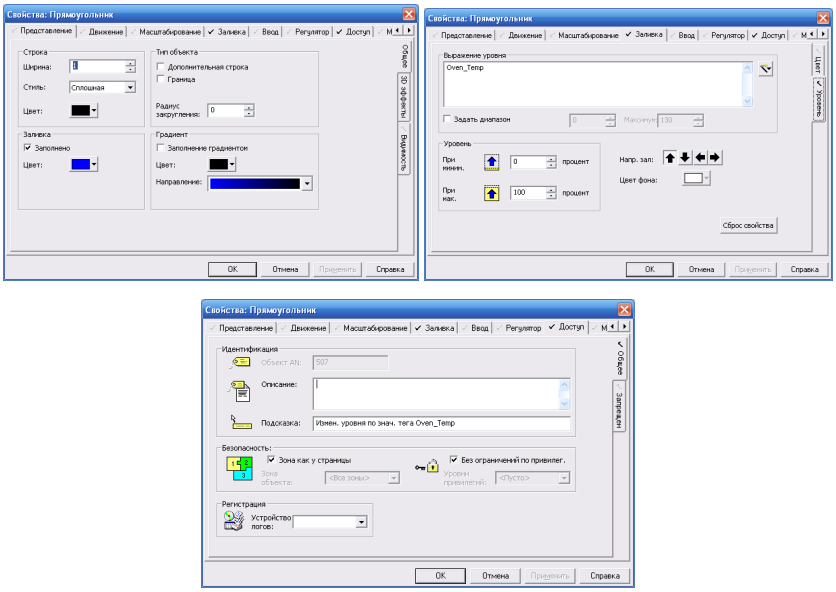


Рис. 4.9. Настройка объекта-гистограммы в графической странице **Oven**

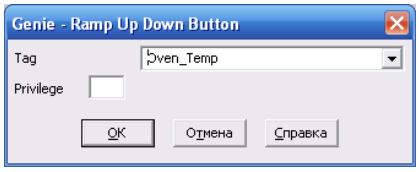


Рис. 4.10. Настройка объекта **Джин — Ramp Up Down Button** в графической странице **Oven**

Выберите вкладку **Представление (Отображаемое значение)** диалогового окна свойств и вставьте с помощью **Мастера выражений** тег **Oven_Temp** в окно **Числовое выражение** (рис. 4.13). Выберите вкладку **Доступ (Общие)** диалога свойств и введите текст всплывающей подсказки в соответствии с приведенным ранее рис. 4.13. Выберите вкладку **Вид (Общие)** диалога свойств и установите желаемый шрифт, цвет, выравнивание, эффекты или воспользуйтесь значениями, предлагаемыми по умолчанию. Для завершения диалога нажмите кнопку **OK**.

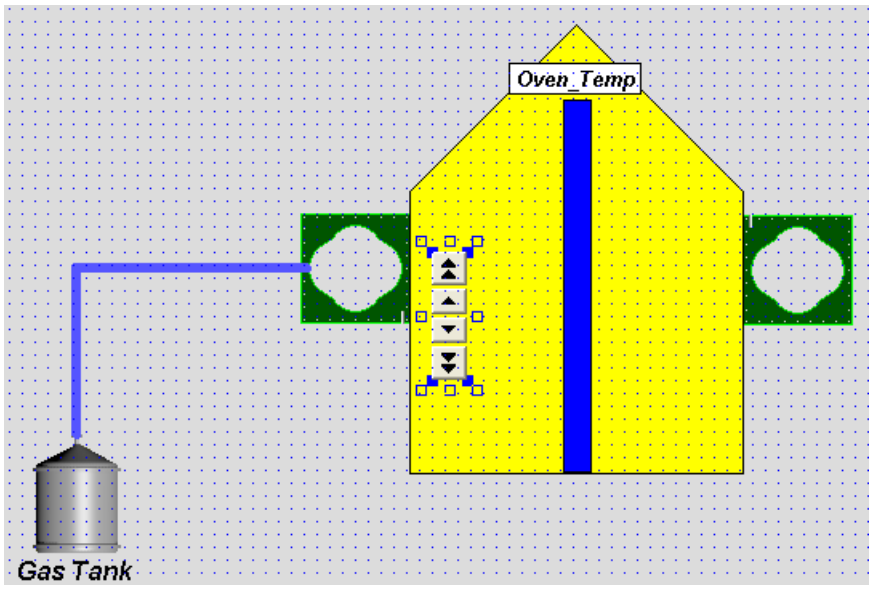


Рис. 4.11. Размещение объекта **Джин — Ramp Up Down Button** в графической странице **Oven**

> ***Совет***
> Для более удобного отображения числового значения температуры поместите в графическую страницу прямоугольник с границей черного цвета и белым цветом заполнения, переместите в него графический объект **Число**, сгруппируйте их и поместите сгруппированный объект на прежнее место внутрь многоугольника.

Сохраните графическую страницу, выполните компиляцию проекта, запустите проект, протестируйте сделанные изменения и завершите работу проекта.

4.5. Отображение текста во время выполнения проекта

Вы можете отображать на графической странице различные текстовые сообщения, зависящие от состояния некоторого дискретного тега или выполнения некоторого условия. Например, вы можете отображать слово **Вкл.** рядом с изображением мотора во время его работы или на том же месте слово **Выкл.** в ином случае.

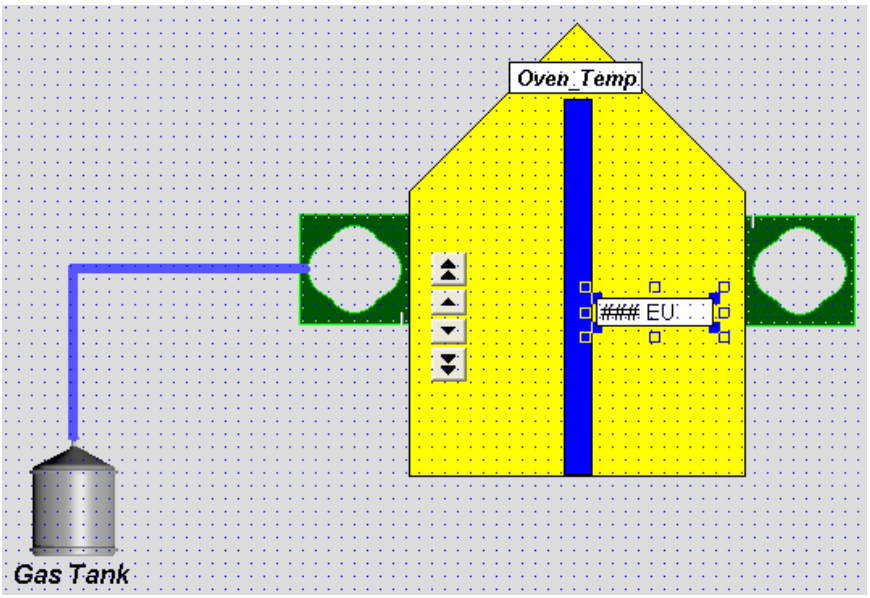


Рис. 4.12. Размещение объекта **Число** в графической странице **Oven**

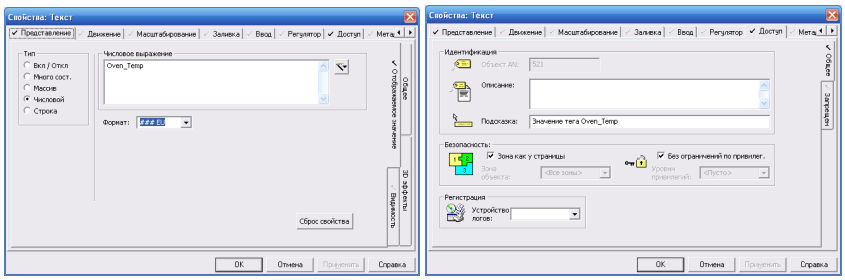


Рис. 4.13. Вид вкладок **Доступ (Общие)** и **Представление (Отображаемое значение)** объекта **Число** в графической странице **Oven**

Упражнение 4.11. Отображение текста и его анимация. Проект Training4

На странице **Oven** проекта **Training3** используйте графический объект **Текст** для отображения состояния тега переменной **Burner_Stat** в виде соответствующего текста, изменения цвета текста в зависимости от состояния тега и переключения тега.

В **Окне объектов** приложения **Построитель графики Citect** (см. приведенный ранее рис. 3.17) щелкните левой кнопкой мыши на кнопке **Текст**, переместите указатель

мыши в требуемое место графической страницы (рис. 4.14), введите первый символ текста и нажмите левую кнопку мыши. Сразу же появится диалоговое окно **Свойства: Текст**.

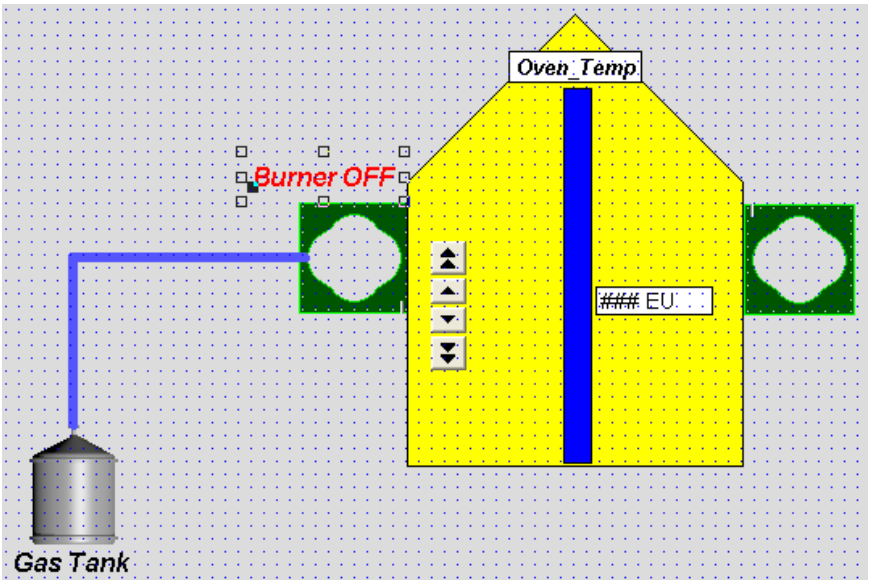


Рис. 4.14. Размещение объекта **Текст** в графической странице **Oven**

Задайте в поле **Текст** начальный отображаемый текст, а в поле **Передний план:** — его цвет (красный) и нажмите кнопку **Применить** (рис. 4.15).

Задайте тег, состояние которого нужно отобразить, и текст, соответствующий возможным состояниям тега. Для этого выберите вкладку **Представление (Отображаемое значение)** диалогового окна свойств, настройте ее в соответствии с рис. 4.16 и нажмите кнопку **Применить**. Во вкладке **Заливка (Цвет)** задайте цвета, которыми будет отображаться текст в зависимости от состояния тега (рис. 4.17) и нажмите кнопку **Применить**. Во вкладке **Ввод (Касание)** задайте переключение тега **Burner_Stat** при щелчке левой кнопкой мыши по объекту **Текст** (рис. 4.18). Для переключения состояния дискретного тега используйте функцию **Toggle**, которую выбирайте с помощью **Мастера алгебраических выражений и функций**. Нажмите кнопку **Применить**. В поле **Подсказка:** вкладки **Доступ (Общие)** задайте текст всплывающей подсказки, которая будет появляться во время выполнения, если курсор мыши поместить над объектом (рис. 4.19). Для закрытия диалогового окна нажмите кнопку **OK**. Сохраните графическую страницу, выполните компиляцию проекта, запустите проект, протестируйте сделанные изменения. Завершите работу проекта.

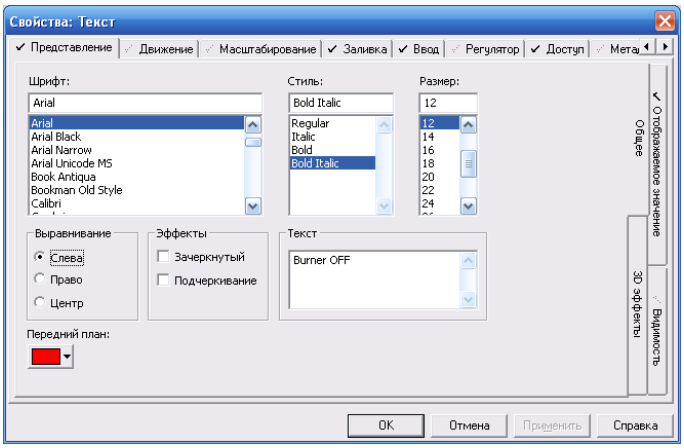


Рис. 4.15. Задание начальных параметров отображаемого текста объекта **Текст** в графической странице **Oven**

Совет

Не забудьте проверить переключение цвета, текста, изменение значения дискретного тега и появление всплывающей подсказки для графических объектов страницы **Oven**. Сохраните проект на магнитном диске. Для этого в среде приложения **Проводник Citect** достаточно выбрать в окне **Список проектов** проект **Training3** и выполнить команду **Создание резервной копии…** его контекстного меню. В появившемся окне **Создание резервной копии проекта** следует указать требуемые параметры и нажать кнопку **OK**. Проект сохраните под именем **Training4.** В дальнейшем работайте с проектом **Training4**. Для этого достаточно выбрать **Мои проекты** в окне **Список проектов** и выполнить команду **Восстановление…** его контекстного меню. В появившемся окне **Восстановить проект** следует указать требуемые параметры и нажать кнопку **OK** (новый проект, имя проекта **Training4**).

4.6. Использование наборов образов (Symbols Sets)

Наборы образов можно использовать на графической странице для отображения последовательности изображений, выбираемых в зависимости от значения тега переменной или некоторого выражения. Это позволяет моделировать движение (динамическое изображение). В учебных проектах **Training — Training4** уже использовался объект **Набор символов (Burner)** для изображения **Горелки** на графической странице **Oven**. Заменим далее статическое изображение пламени **Горелки** динамическим изображением пламени.

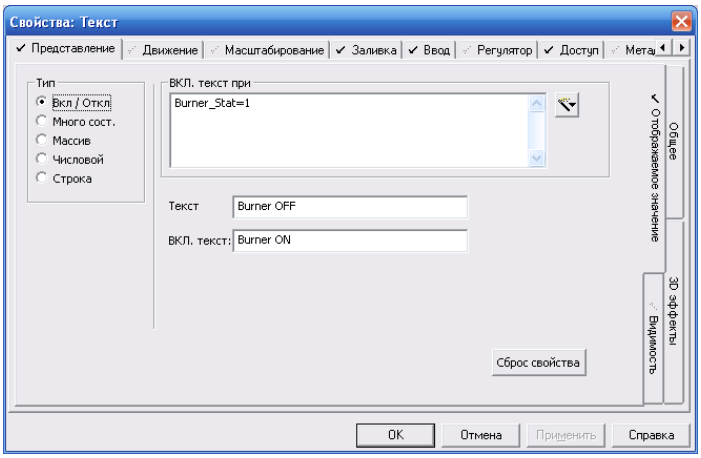


Рис. 4.16. Задание тега и отображаемого текста объекта **Текст** в графической странице **Oven**

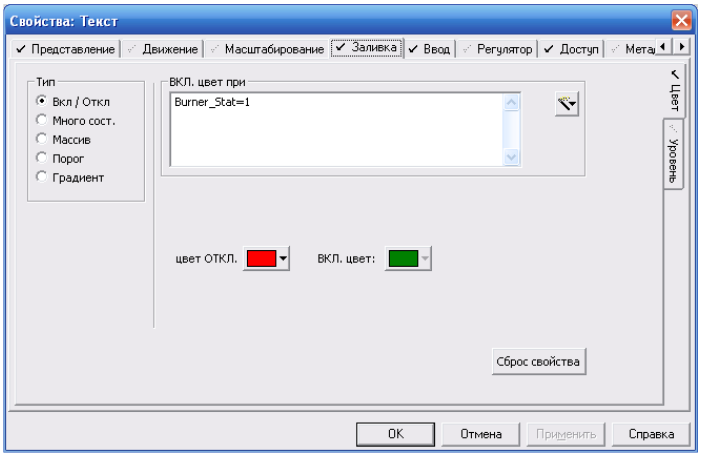


Рис. 4.17. Задание цветов отображаемого текста объекта **Текст** в графической странице **Oven**

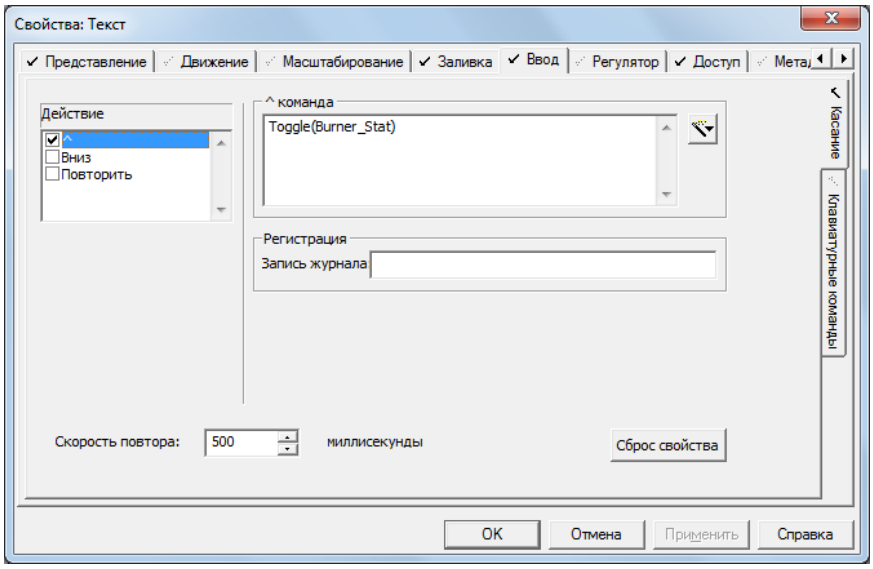


Рис. 4.18. Задание условия переключения дискретного тега при нажатии объекта **Текст** в графической странице **Oven**

Упражнение 4.12. Динамическая анимация

На странице **Oven** проекта **Training4** модифицируйте графический объект **Burner** для отображения пламени в виде статического или динамического изображения в зависимости от состояния тега переменной **Burner_Stat**. На графической странице **Oven** дважды щелкните по **Burner** и появится диалоговое окно задания свойств этого объекта. Для анимации пламени горелки выберите вкладку **Вид (Общие)**, в поле **Тип** выберите **Анимированный**, в поле **Кадры анимации** выберите **Frame1**, нажмите кнопку **Задать…** и появится диалог **Выбрать символ**. В нем в поле **Библиотека:** выберите **misc2**, в поле **Символ — flame002** и нажмите кнопку **OK**. Аналогичным образом, в поле **Кадры анимации** заполните **Frame2 — Frame4** символами **flame003 — flame005** и нажмите кнопку **Применить** (рис. 4.20). Для задания всплывающей подсказки выберите вкладку **Доступ (Общие)** диалогового окна свойств, в поле **Подсказка:** введите **Анимация пламени при Burner_Stat=1** и для сохранения диалога нажмите кнопку **OK**. Аналогичным образом сконфигурируйте второй экземпляр объекта **Горелка**, расположенный на графической странице **Oven** справа.

Рис. 4.19. Задание текста всплывающей подсказки для объекта **Текст** в графической странице **Oven**

> ***Замечание***
> Этой же цели можно достичь и иным способом. Удалите второй экземпляр объекта **Горелка**, скопируйте оставшийся экземпляр, поверните его на 180 градусов так, как это указывалось ранее, и поместите в требуемое место.

Сохраните графическую страницу, выполните компиляцию проекта, запустите проект, протестируйте сделанные изменения.

> ***Совет***
> Не забудьте проверить появление всплывающей подсказки и анимацию изображения пламени. Для проверки анимации изображения пламени нажмите последовательно два раза на объект **Текст** (**Burner OFF-ON**), обращая каждый раз внимание на изображение пламени.

Завершите работу проекта.

Рассмотрим еще один пример анимации изображений — дополним трубопровод в графической странице **Oven** клапаном и выполним анимацию его изображения.

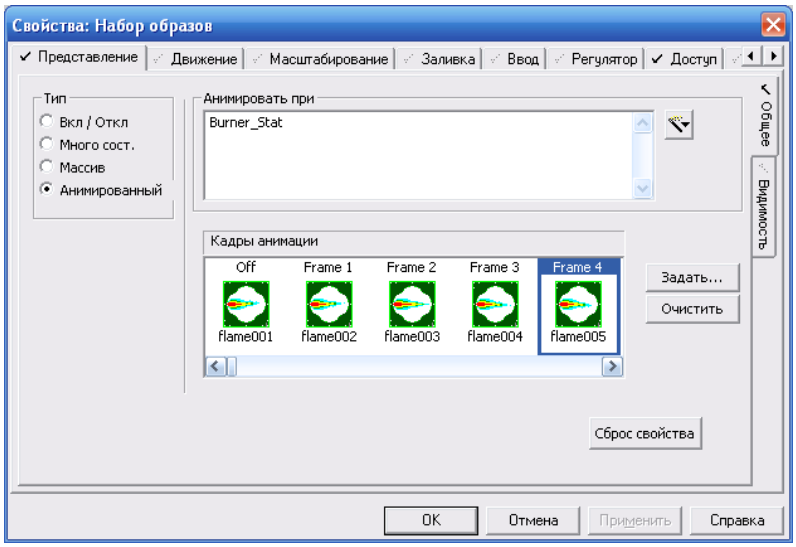


Рис. 4.20. Анимация пламени горелки для объекта **Burner** в графической странице **Oven**

Упражнение 4.13. Анимация клапана трубопровода. Проект Training5

На странице **Oven** проекта **Training4** добавьте графический объект для отображения клапана трубопровода и его состояния. При щелчке левой кнопкой мыши по клапану предусмотрите изменение изображения клапана и изменение значения дискретного тега, связанного с клапаном. В **Окне объектов** приложения **Построитель графики Citect** (см. приведенный ранее рис. 3.17) щелкните левой кнопкой мыши на кнопке **Набор образов**, переместите указатель мыши в требуемое место графической страницы (рис. 4.21) и нажмите левую кнопку мыши. Сразу же появится диалоговое окно **Набор образов Properties**. Для изменения изображения клапана в зависимости от значения дискретного тега **Gas_Valve** выберите вкладку **Вид (Общие)** диалога свойств, настройте ее в соответствии с рис. 4.22. При заполнении поля **ВКЛ. символ при** используйте **Мастер алгебраических выражений и функций**, а поля **Символ ОТКЛ.** и **ВКЛ. символ:** заполняйте с помощью соответствующих кнопок **Задать…**в соответствии с рис. 4.23. Нажмите кнопку **Применить**. Для изменения значения дискретного тега **Gas_Valve**, связанного с клапаном, при щелчке левой кнопкой мыши по клапану выберите вкладку **Ввод (Касание)** диалогового окна свойств, настройте ее в соответствии с рис. 4.24 и нажмите кнопку **Применить**. Обратите внимание, что в поле **^ команда** использована стандартная функция языка **Cicode Toggle()**, действие которой состоит в изменении значения дискретного тега,

заданного аргументом. При выборе этой функции целесообразно пользоваться **Мастером алгебраических выражений и функций**.

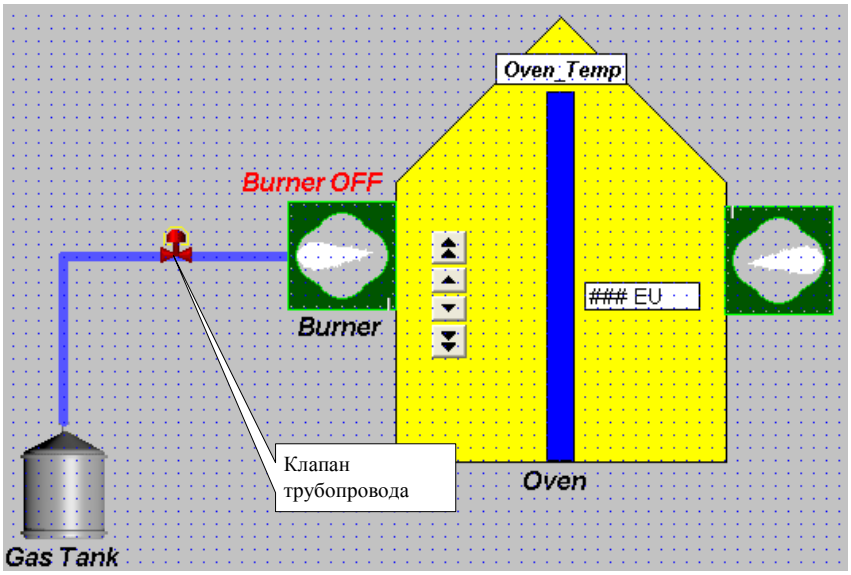


Рис. 4.21. Вид графической страницы **Oven** после добавления клапана трубопровода

Для задания для клапана трубопровода всплывающей подсказки выберите вкладку **Доступ (Общие)** заполните поле **Подсказка:** в соответствии с рис. 4.25 и завершите диалог нажатием кнопки **OK**. Сохраните графическую страницу, выполните компиляцию проекта, запустите проект, протестируйте сделанные изменения. При тестировании последовательно щелкните левой кнопкой мыши по изображению клапана. После каждого щелчка обратите внимание на изменение изображения клапана и значение дискретного тега **Gas_Valve**. Не забудьте проверить функционирование всплывающей подсказки. Завершите работу проекта.

Совет

Сохраните проект на магнитном диске. Для этого в среде приложения Citect Explorer достаточно выбрать в окне **Список проектов** проект **Training4** и выполнить команду **Создание резервной копии…** его контекстного меню. В появившемся окне **Создание резервной копии проекта** следует указать требуемые параметры и нажать кнопку **OK**. Проект сохраните под именем **Training5.** В дальнейшем работайте с проектом **Training5**. Для этого достаточно выбрать **Мои проекты** в окне **Список проектов** и выполнить команду **Восстановление…** его контекстного меню. В появившемся окне **Восстановить проект** следует указать требуемые параметры и нажать кнопку **OK** (новый проект, имя проекта **Training5**).

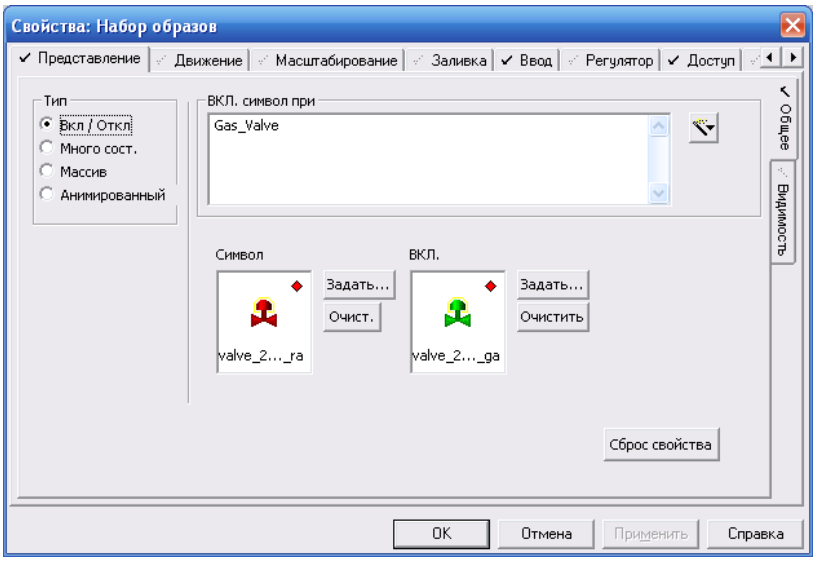


Рис. 4.22. Задание изменения цвета клапана в зависимости от состояния дискретного тега **Gas_Valve** графической страницы **Oven**

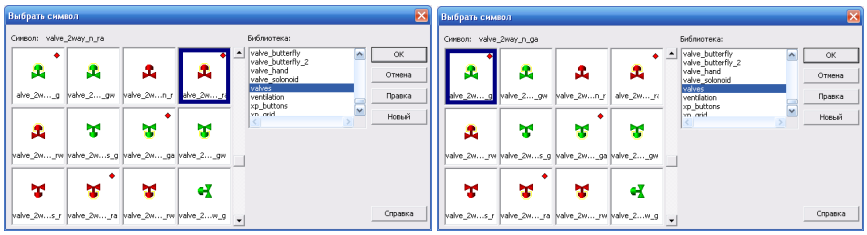


Рис. 4.23. Выбор изображения клапана трубопровода

Примечание

Вставив символ из библиотеки на графическую страницу, можно перемещать его, изменять его размер и форму, переносить на передний план, редактировать его свойства точно так же, как и с другими типами объектов. Символ из библиотеки можно вставить на графическую страницу как непривязанный (unlinked) или как привязанный (linked) символ. Непривязанный символ, в отличие от привязанного, не обновляется при изменении символа в библиотеке. Разорвать связь символа с библиотекой можно командой **Правка | Разорвать связь**.

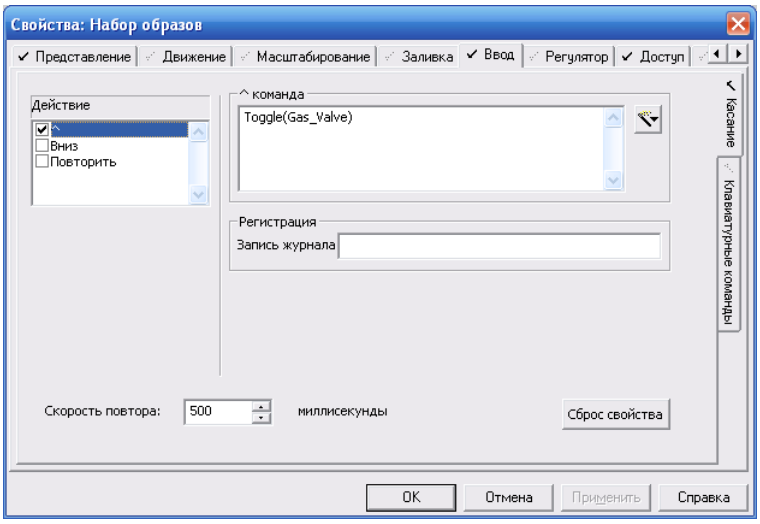


Рис. 4.24. Задание изменения значения дискретного тега при щелчке по клапану трубопровода

Примечание

Продолжение рассмотрения других, более "продвинутых" средств графики Vijeo Citect с несколько иных позиций будет продолжено далее в главе 9 после изучения сигналов тревог, трендов, команд, управляющих элементов и анализатора процессов.

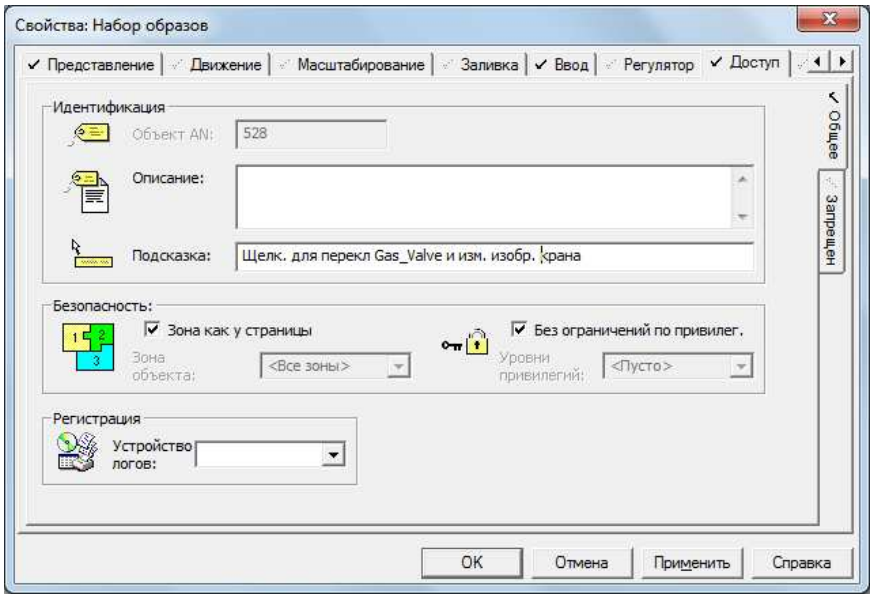


Рис. 4.25. Задание всплывающей подсказки для клапана трубопровода

Глава 5. Сигналы тревог (Alarms)

Совет

Материал, относящийся к **Сигналам тревог** (**Alarms**) и их использованию можно найти в [2], темы Using Vijeo Citect | Using Alarms in SCADA; [3], слайды 190 — 214.

Защита ценного оборудования предприятия является одной из центральных особенностей системы Vijeo Citect. Сигналы тревог обеспечивают постоянный мониторинг и другие требуемые операции при возникновении любых ошибок оборудования. Vijeo Citect поддерживает два типа тревог — сигналы тревог аппаратных средств и конфигурируемые сигналы тревог.

Сигналы тревог аппаратных средств. Система Vijeo Citect непрерывно управляет диагностическими подпрограммами, чтобы проверить все периферийное оборудование, в том числе, устройства ввода/вывода. Обо всех ошибках автоматически формируются сообщения оперативному персоналу. Тревоги аппаратных средств полностью интегрированы с Vijeo Citect — никакого конфигурирования их не требуется.

Конфигурируемые сигналы тревог. В отличие от тревог аппаратных средств, вы должны конфигурировать сигналы тревог в виде сообщений об ошибочных ситуациях на предприятии (например, когда уровень жидкости в резервуаре слишком высок или когда перегревается двигатель). Конфигурируемые сигналы тревог вводятся в базу данных с помощью форм (диалоговых окон), аналогичных формам для конфигурирования тегов переменных. Каждый тип сигналов тревог имеет отличающиеся условия возникновения и параметры и, соответственно этому, свою форму для конфигурирования тревог.

Эта группа сигналов включает цифровые сигналы тревог, сигналы тревог с временной меткой, аналоговые сигналы тревог и расширенные сигналы тревог.

Дискретные алармы (Digital Alarms) формируются в зависимости от значений одного или двух двоичных тегов (возможно использование в проверяемом выражении операции логического умножения AND).

Хронологические алармы (Time Stamped Alarms) похожи на цифровые — сигнал включается при изменении состояния цифрового тега. В отличие от обычных сигналов тревог, эти сигналы снабжаются отметкой времени, полученной из программируемого логического контроллера (ПЛК).

Аналоговые алармы (Analog Alarms) формируются, когда значение аналоговой переменной выходит за заданные пределы. Аналоговый сигнал тревоги может формироваться как любая комбинация следующих типов тревог — **High** (**Высокое** значение сигнала) или **High High** (**Высокое Высокое** значение сигнала) тревоги, **Low** (**Низкое** значение сигнала) или **Low Low** (**Низкое Низкое** значение сигнала) тревоги, **Deviation** (**Отклонение** значения сигнала от уставки превышает заданное значение) тревога и **Rate of Change** (**Скорость Изменения** значения сигнала) тревога.

Расширенные алармы (Advanced Alarms) формируются, когда значение функции языка Cicode изменяется от значения **FALSE** к значению **TRUE**.

5.1. Конфигурирование сигналов тревог

Для *добавления нового сигнала тревоги* имеются две возможности. В рамках первой возможности откройте приложение **Проводник Citect**, выберите требуемый проект (например, **Training5**), откройте узел **Алармы** и дважды щелкните по иконе, соответствующей типу добавляемого сигнала тревоги (рис. 5.1). В рамках второй возможности откройте приложение **Редактор проектов Citect**, выберите меню **Алармы** и выполните подходящую команду этого меню, соответствующую типу добавляемого сигнала тревоги (рис. 5.2).

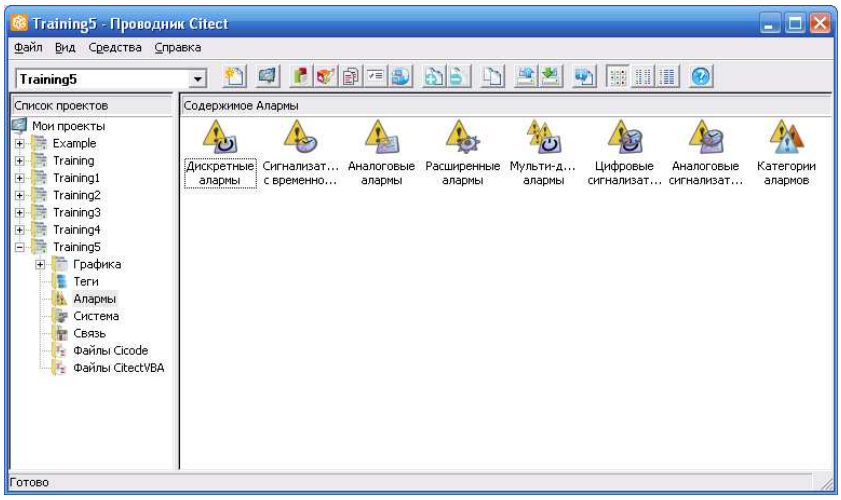


Рис. 5.1. Добавление сигнала тревоги

Упражнение 5.1. Добавление сигналов тревог

В проект **Training5** добавьте один аналоговый и два битовых сигнала тревоги. Откройте приложение **Редактор проектов Citect** , выполните команду **Сервера | Серверы алармов**, настройте параметры сервера в соответствии с рис. 5.3a) и нажмите кнопку **Добавить**. Далее для конфигурирования аналоговой тревоги выберите меню **Алармы** и выполните команду **Аналоговые алармы** (см. приведенный ранее рис. 5.2). Настройте параметры аналогового сигнала тревоги в соответствии с рис. 5.3b) и нажмите кнопку **Добав**.

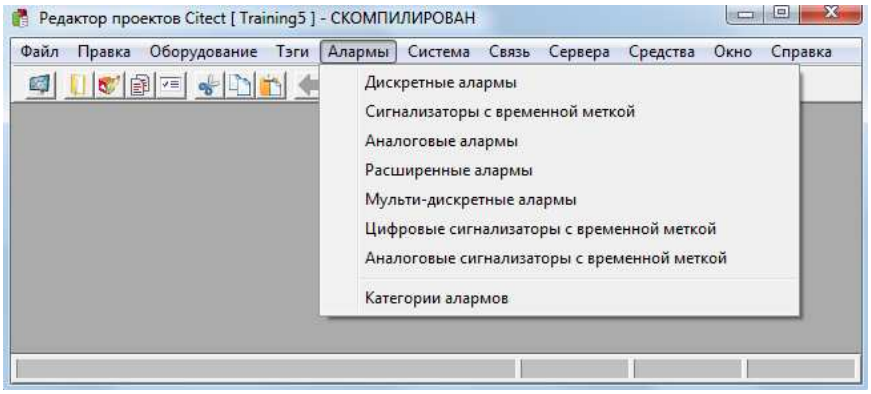
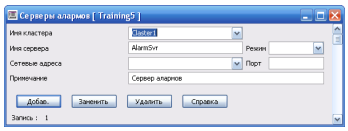
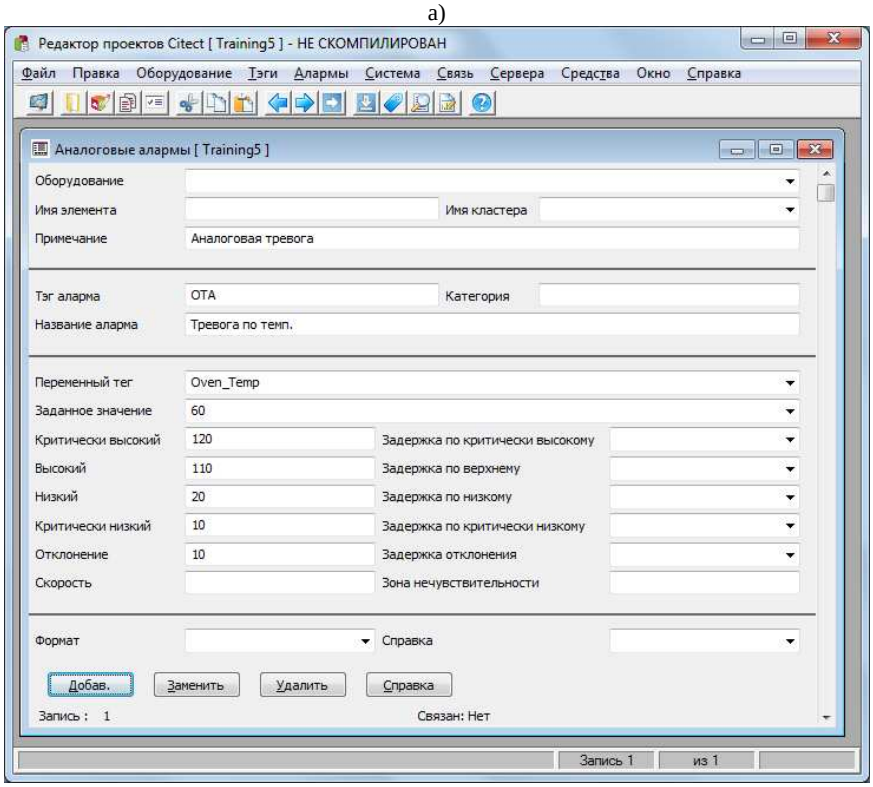


Рис. 5.2. Добавление сигнала тревоги (вариант 2)



a)



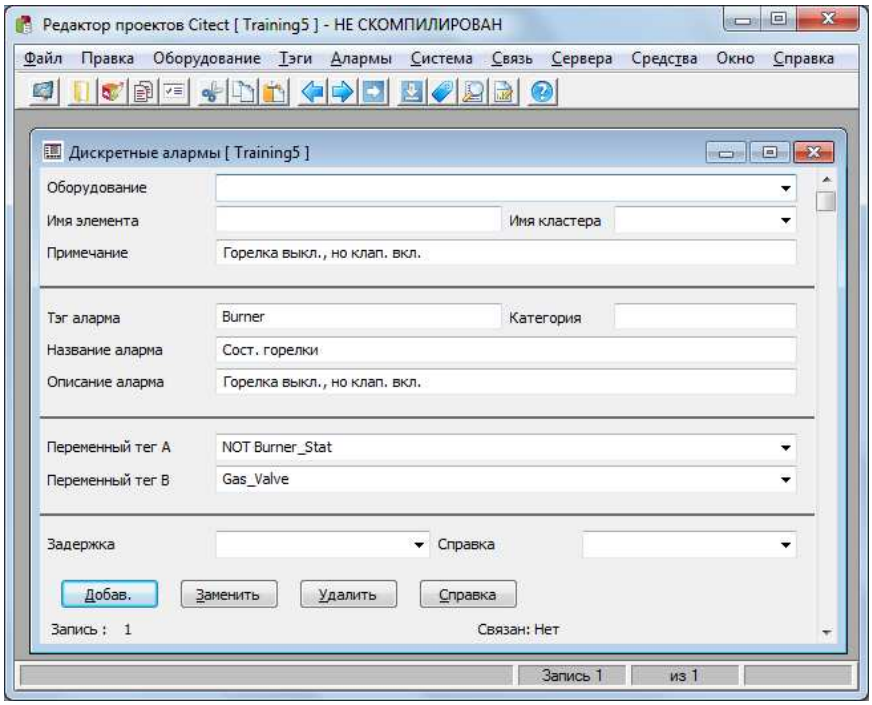
b)

Рис. 5.3. Конфигурирование сервера тревог и аналогового сигнала тревоги

Для создания дискретных тревог в приложении **Редактор проектов Citect** выберите меню **Алармы** и выполните команду **Дискретные алармы** (см. приведенный ранее рис. 5.2). Настройте параметры битовых сигналов тревоги в соответствии с рис. 5.4, каждый раз нажимая кнопку **Добавить**. Выполните компиляцию проекта.

5.2. Отображение сигналов тревог

Во включаемых проектах имеется несколько стандартных страниц, которые можно использовать для отображения сигналов тревог различных типов. При использовании включаемого проекта **CSV_Include** конфигурируемые тревоги можно отображать на странице **Alarms** (активные тревоги), тревоги аппаратных средств — на странице **Hardware Alarms**, историю появления сообщений тревог из файла регистрации событий — на странице **Alarm Summary** и сигналы тревоги, запрещенные вручную, — на странице **Disabled Alarms**. Все перечисленные страницы базируются на шаблонах тревог включаемого проекта **CSV_Include**.



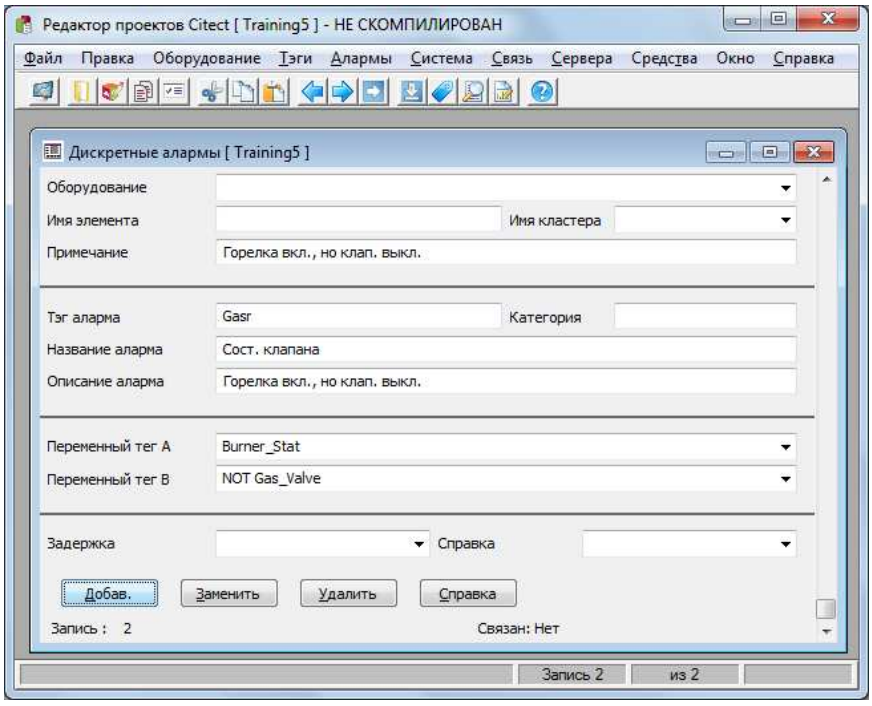


Рис. 5.4. Конфигурирование битовых сигналов тревог

При использовании включаемого проекта **Tab_Style_Include** конфигурируемые тревоги можно отображать на странице **Alarm** (активные тревоги, шаблон **alarm**), тревоги аппаратных средств — на странице **Hardware** (шаблон **hardware**), историю появления сообщений тревог из файла регистрации событий — на странице **Summary** (шаблон **summary**) и сигналы тревоги, запрещенные вручную, — на странице **Disabled** (шаблон **disabled**). Все перечисленные страницы базируются на шаблонах тревог включаемого проекта **Tab_Style_Include** и должны быть предварительно созданы в проекте для использования их с целью отображения сигналов тревог.

Упражнение 5.2. Отображение (просмотр) сигналов тревог. Проект Training6

Выберите проект **Training5** для включения и просмотра различных сигналов тревог. Для просмотра тревог средствами включаемого проекта **Tab_Style_Include** предварительно включите в проект графические страницы **Alarm** (шаблон **alarm** стиля **Tab_Style_1**), **Hardware** (шаблон **hardware** стиля **Tab_Style_1**), **Summary**

(шаблон **summary** стиля **Tab_Style_1**) и **Disabled** (шаблон **disabled** стиля **Tab_Style_1**). Запустите проект **Training5**, зарегистрируйтесь как привилегированный пользователь. Перейдите в графическую страницу **Oven** и смоделируйте аналоговую (выполните изменение значения тега **Oven_Temp** во всем диапазоне от минимума до максимума с помощью **Genie — Ramp Up Down Button**) и дискретную тревоги (переключите по нескольку раз подряд состояние клапана трубопровода и состояние графического объекта **Burner ON-OFF**).

Вначале выполните просмотр сигналов тревог средствами включаемого проекта **CSV_Include**. С этой целью воспользуйтесь командами меню **Pages**. Для перехода в страницу активных тревог выполните команду **Pages | CSV_Alarm**. Возможный вид появившейся страницы иллюстрирует рис. 5.5. Выберите в верхнем или нижнем списке страницы **Alarms** требуемый сигнал тревоги, например, **OTA**, и с помощью правой кнопки мыши вызовите его контекстное меню (рис. 5.6). Верхняя команда контекстного меню **OTA** предназначена для отображения свойств выбранного сигнала тревоги (рис. 5.7), команда **Acknowledge** служит для квитирования (подтверждения) сигнала тревоги, команда **Disable** — для отключения, а команда **Enable** — для подключения сигнала тревоги.

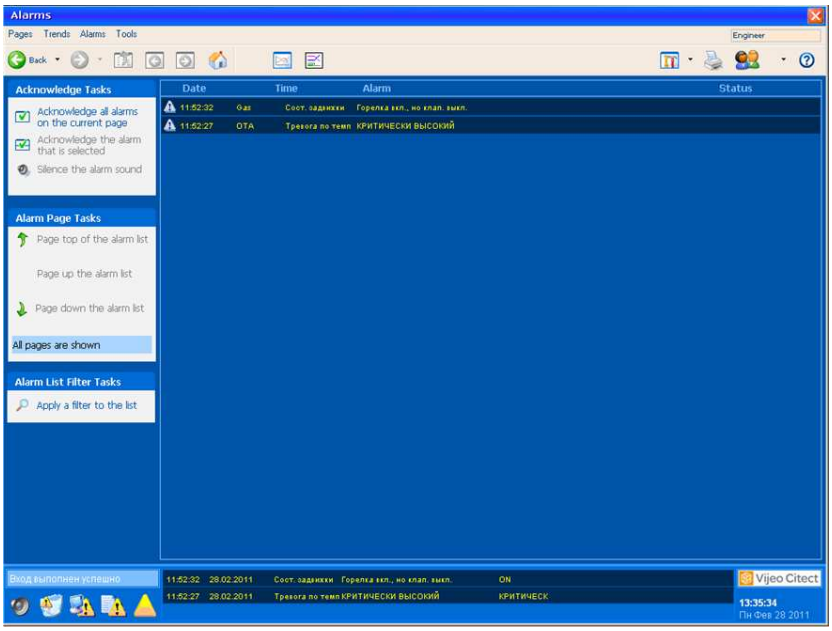


Рис. 5.5. Страница отображения активных конфигурированных сигналов тревог

Замечание

Обратите внимание, что команда **Enable** не доступна в странице **Alarms**, а доступна только в странице **Disabled Alarms**. И наоборот, команда **Disable** доступна в странице **Alarms** и не доступна в странице **Disabled Alarms**.

Для упражнения отключите сигнал тревоги **OTA** и снова подключите его. Для переключения между страницами сигналов тревог пользуйтесь командами **Alarms | Active Alarms**, **Alarms | Alarm Summary**, **Alarms | Disabled Alarms** и **Alarms | Hardware Alarms**. Вместе с тем, обратите внимание на то, что кнопки **Alarm Page**, **Alarm Summary Page**, **Hardware Alarm Page** и **Disable Alarm Page**, расположенные на панели инструментов в нижней части окна (см. приведенный ранее рис. 5.5) производят аналогичные действия. Перейдите на страницу **Alarm Summary** и внимательно изучите ее (рис. 5.8).

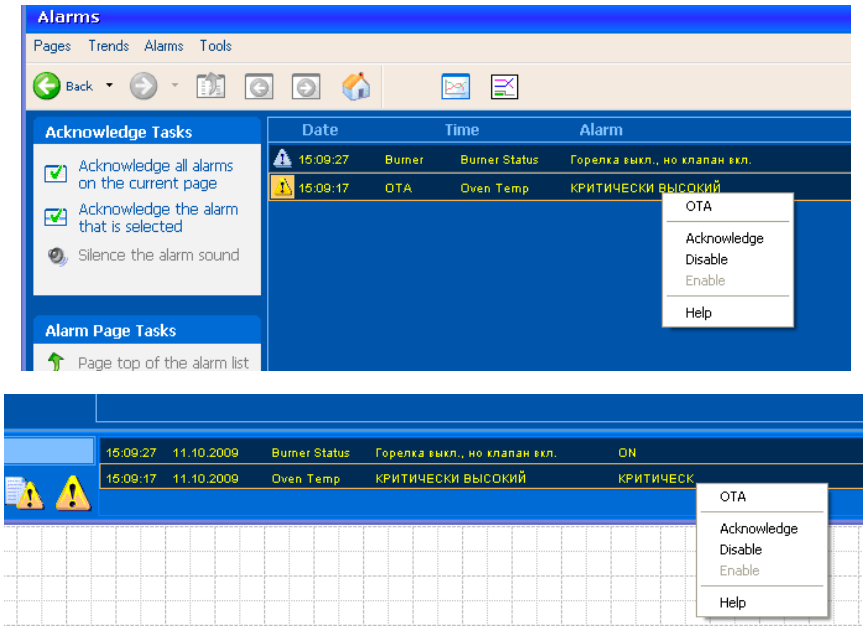


Рис. 5.6. Контекстные меню выбранных сигналов тревог

Для просмотра сигналов тревог средствами включаемого проекта перейдите на страницу **Oven** и воспользуйтесть командами меню **Alarms** и/или кнопками на панели инструментов, расположенной в левой нижней части графической страницы. Работа с графическими страницами отображения информации о тревогах выполняется аналогично и несколько отличается лишь в части интерфейса.

Завершите работу проекта.

Совет

Сохраните проект на магнитном диске. Для этого в среде приложения **Проводник Citect** достаточно выполнить команду **Создание резервной копии…** его контекстного меню. В появившемся окне **Создание резервной копии проекта** следует указать требуемые параметры и нажать кнопку **OK**. Проект сохраните под именем **Training6.** В дальнейшем работайте с проектом **Training6**. Для этого достаточно выбрать **Мои проекты** в окне **Список проектов** и выполнить команду **Восстановление…** его контекстного меню. В появившемся окне **Восстановить проект** следует указать требуемые параметры и нажать кнопку **OK** (новый проект, имя проекта **Training6**).

Замечание

Рассмотрение других аспектов сигналов тревог будет продолжено далее в главе 12.

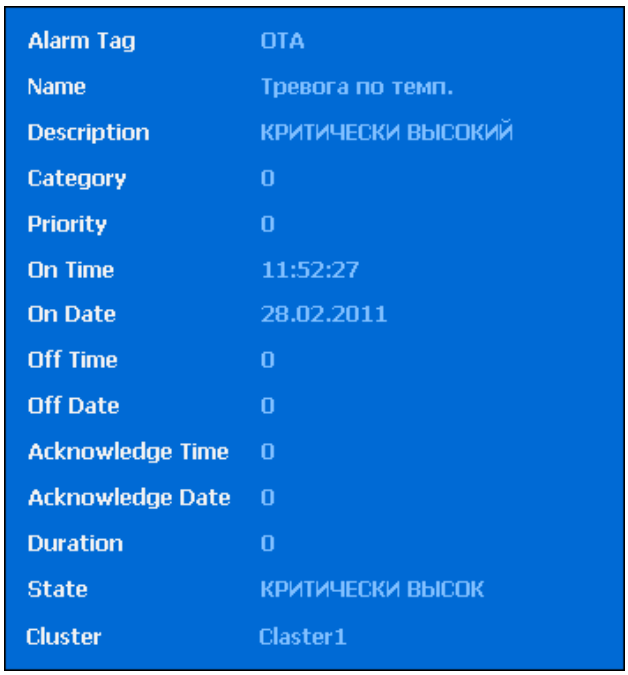


Рис. 5.7. Свойства выбранного сигнала тревоги

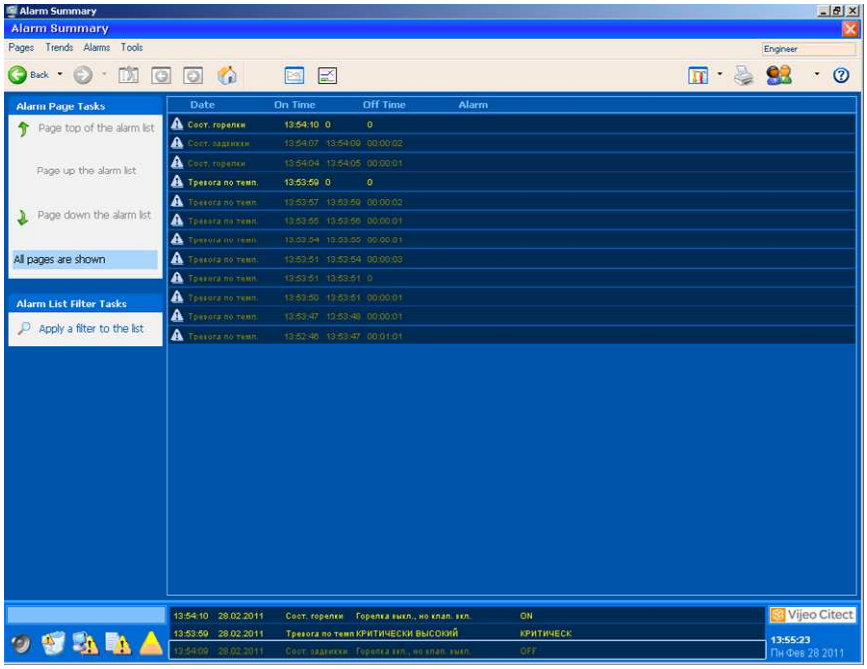


Рис. 5.8. Вид страницы **Alarm Summary**

Глава 6. Графики тегов — тренды (Trends)

Совет

Рассмотрите материал, относящийся к **Трендам** (**Trends**) и их использованию в [2], темы Using Vijeo Citect | Logging and Trending Data; [3], слайды 225 — 245.

Визуальное представление прошлой и текущей деятельности, основанное на использовании **Трендов** (графиков времени), улучшает понимание работы предприятия. С помощью **Трендов** вы можете представить значения переменных (или процесса) в графическом формате. Поскольку эти значения изменяются во времени, то график перемещается по странице так, чтобы последние значения всегда отображались.

Вы можете также прокручивать архивные данные назад, чтобы отображать прошлые значения переменных или параметров процесса. Сбор архивных данных продолжается даже тогда, когда графический показ не активен. Вы можете переключаться между страницами, не мешая сбору архивных данных. Вы можете отображать в виде графика значение любой одиночной переменной или значение выражения, записанного на языке Cicode. Вы можете отображать на экране любое количество трендов одновременно и на каждом тренде до восьми временных графиков.

Система Vijeo Citect поддерживает три типа трендов — **Периодические** (**Periodic**, графики строятся постоянно на основе заданного интервала времени), **Событийные** (**Event**, очередные точки графика строятся каждый раз, когда происходит заданное событие, т. е. оказывается справедливым заданное условие) и **Смешанные** (**Periodic Event**, построение графиков с заданным периодом происходит только в случае выполнения заданного условия).

Тренды могут быть добавлены в систему Vijeo Citect путем создания **Тренд-тегов** (**Trend Tags**). Каждый тренд-тег должен иметь один или несколько отдельных файлов, в которых сохраняются данные. Vijeo Citect непрерывно сохраняет данные тренда, независимо от того, будет ли отображаться тренд на графической странице.

Для регистрации значений тега переменной с помощью тренда создайте тренд-тег и определите один или более файлов для регистрации данных тренд-тега. Для отображения тренда создается страница тренда и назначается перо для отображения значений тренд-тега в окне тренда.

Для сохранения регистрируемых данных система Vijeo Citect использует ротацию непрерывной цепочки из нескольких архивных файлов, вместо того, чтобы сохранять их в одном большом файле. По умолчанию, Vijeo Citect использует два файла, каждый из которых сохраняет данные на протяжении одной недели, начиная с полуночи воскресенья. По умолчанию, имя файла совпадает с именем тренд-тега. Пользователь может изменить период регистрации данных и число архивных файлов. Для просмотра дополнительных параметров настройки тренда откройте диалоговое

окно (форму) свойств тренд-тега и нажмите клавишу **F2** для вывода расширенных параметров.

6.1. Конфигурирование тренд-тегов

Для добавления нового тренда имеются две возможности. Во первых, в среде приложения **Проводник Citect**, выберите требуемый проект (например, **Training6**), откройте папку **Теги** и дважды щелкните по иконке **Дескрипторы тренда** (рис. 6.1). В рамках второй возможности откройте приложение **Редактор проектов Citect** и выполните команду **Теги | Дескрипторы тренда** (рис. 6.2).

Примечание
В прдыдущей версии (Vijeo Citect 7.20) команда **Дискрипторы трендов** меню **Тэги** называлось **Теги трендов**.

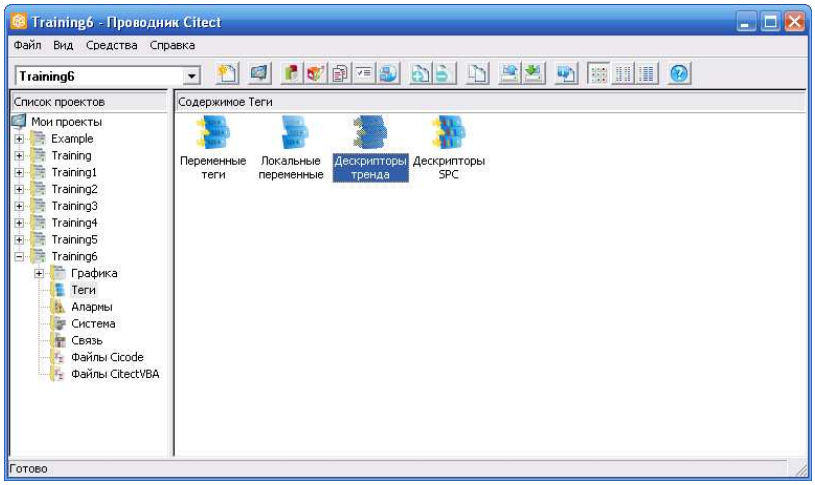


Рис. 6.1. Первый способ добавления **Дескрипторов тренда**

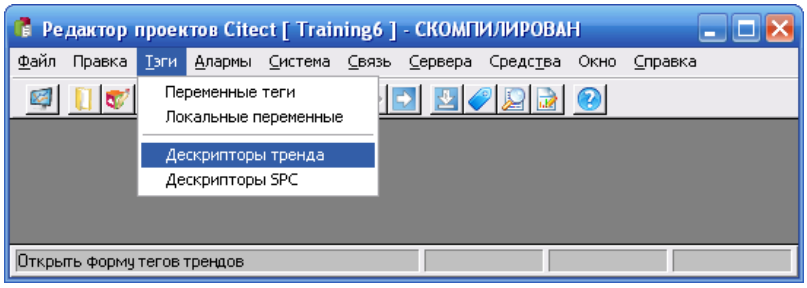
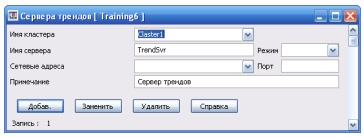


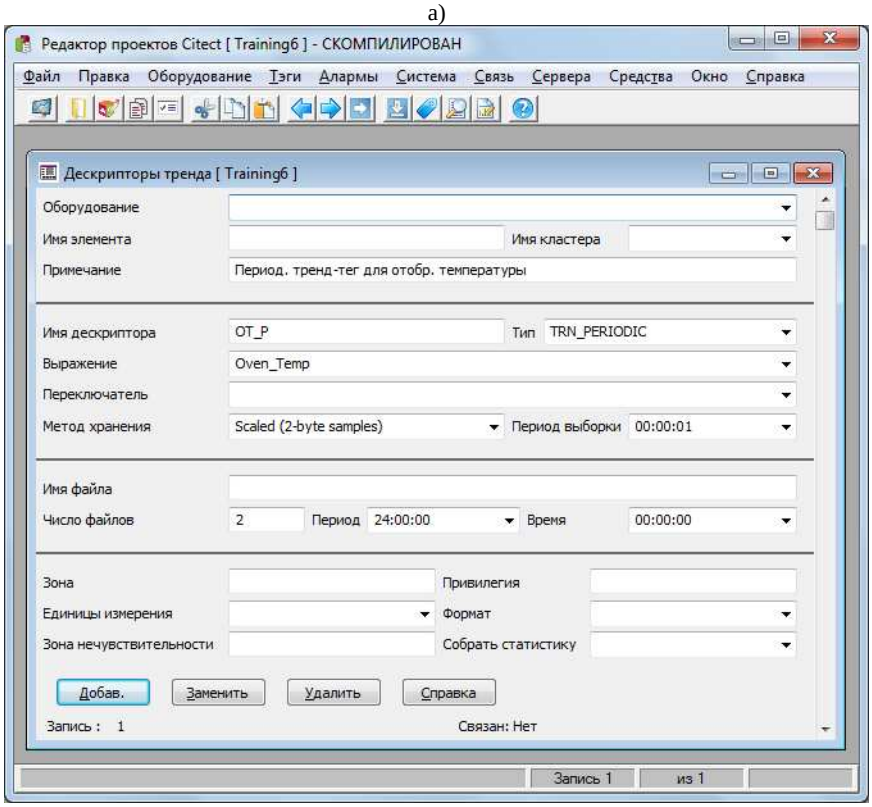
Рис. 6.2. Второй способ добавления **Дескрипторов тренда**

Упражнение 6.1. Создание и конфигурирование дескриптора тренда

Добавьте в проект **Training6** новый **Дескриптор тренда** и сконфигурируйте его. Откройте приложение **Редактор проектов Citect**, выполните команду **Сервера | Сервера трендов**, настройте параметры тренд-сервера в соответствии с рис. 6.3a) и нажмите кнопку **Добавить**. Выберите меню **Теги** и выполните команду этого меню **Теги | Дескрипторы тренда** (см. приведенный ранее рис. 6.2). Сконфигурируйте добавляемый **Тренд-тег** в соответствии с рис. 6.3b) и нажмите кнопку **Добавить**. Выполните компиляцию проекта.



a)



b)

Рис. 6.3. Конфигурирование **Сервера трендов** и **Дескриптора тренда**

Примечание

Для отображения расширенного диалогового окна (формы) конфигурирования тренд-тега достаточно нажать клавишу **F2**. Если поля в появившейся нижней половине формы оставить пустыми, то для настройки тренд-тега будут использованы умалчиваемые значения. Для возврата к отображению формы в нормальном виде достаточно еще раз нажать клавишу **F2**.

6.2. Отображение трендов

Во включаемом проекте **CSV_Include** имеется несколько стандартных страниц, которые можно использовать для отображения трендов. Все стандартные страницы базируются на шаблонах трендов включаемого проекта **CSV_Include**. Этот включаемый проект содержит следующие шаблоны трендов. Шаблон **trend** — отображает тренд с восемью перьями. Предварительно созданная страница **CSV_Trend** создана на основе этого шаблона. Шаблон **doubletrend** — отображает два тренда каждый с восемью перьями, разделенные на экране. Предварительно созданная страница **CSV_TrendDouble** создана на основе этого шаблона. Шаблон **PopTrend** — отображает всплывающий тренд с четырьмя перьями, который можно вызвать из любой графической страницы. Использование этих шаблонов и основанных на них графических страниц считается устаревшей практикой, наследованной из предыдущих версий Vijeo Citect. Тем не менее эту возможность мы также рассмотрим.

> ***Замечание***
> Тренд-теги разных типов (периодические, событийные и периодически-событийные) могут не отображаться на одном и том же тренде.

В приложении **Построитель графики Citect** с помощью команды **Объекты | Тренд** (рис. 6.4) на любой графической странице можно создать полностью настроенный тренд.

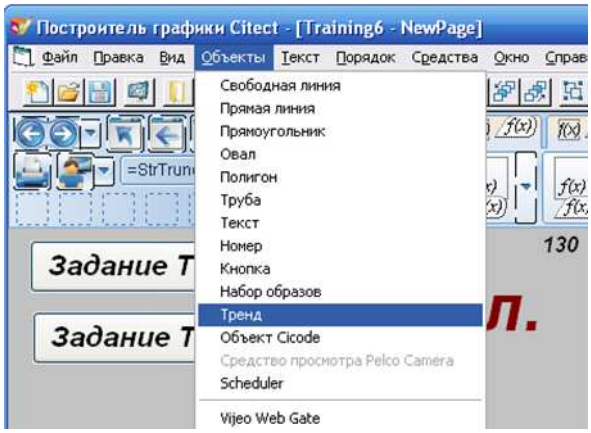


Рис. 6.4. Создание объекта **Тренд** на любой графической странице

Упражнение 6.2. Создание, конфигурирование и отображение тренд-тега на обычной графической странице

Создайте тренд на существующей странице **NewPage** проекта **Training6**. В окне свойств тренда выберите для отображения созданный ранее тренд-тег и задайте цвет графика. Протестируйте сконфигурированный тренд.

Упражнение 6.3. Отображение тренд-тега средствами включаемого проекта CSV_Include (устаревшая практика)

В проекте **Training6** протестируйте различными способами отображение созданного тренд-тега. Запустите проект **Training6** и перейдите в графическую страницу **Oven**. Выполните изменение значения тега **Oven_Temp** во всем диапазоне от минимума до максимума и обратно с помощью **Genie — Ramp Up Down Button**. Изменение значения этих тегов будут отображаться далее в виде трендов различными способами. Перейдите на графическую страницу **CSV_AdminTools** и выполните команду **Trend | Single Trend**. Появится графическое окно с полноэкранным трендом (рис. 6.5). В этом окне выполните команду **Select Trend Pen1** контекстного меню для кривой зеленого цвета, в появившемся диалоге **Trend Selection** выберите тренд-тег **OT_P** и нажмите кнопку **Add** (рис. 6.6). Вид окна после этого показан на рис. 6.7. Перейдите в архивный режим и поэкспериментируйте с кнопками управления архивным режимом.

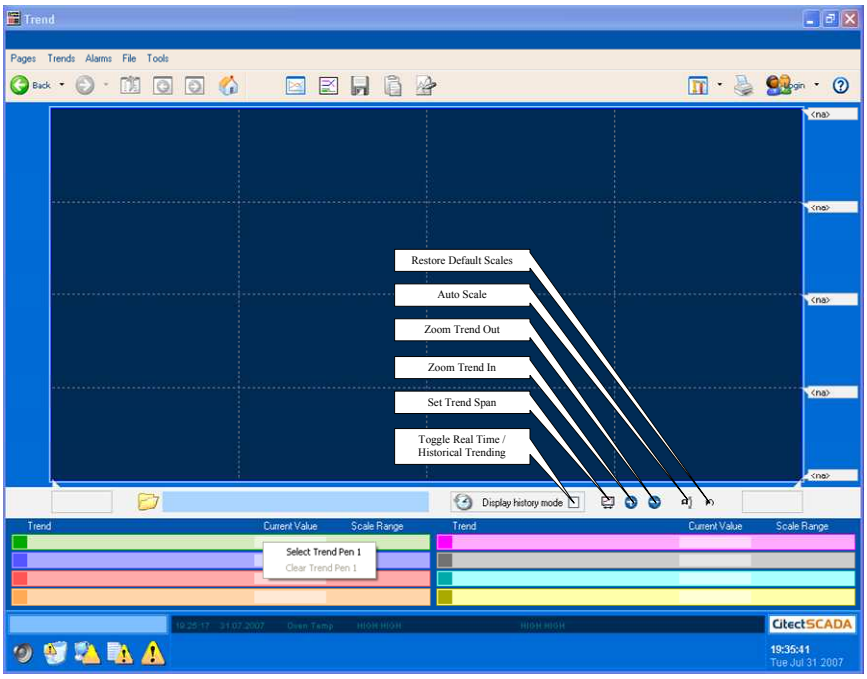


Рис. 6.5. Вид окна **Single Trend** для отображения тренда

Выполните команду **Trend | Double Trend**. Появится графическое окно с двумя полноэкранными трендами. Настройте оба тренда на отображение тренд-тега **OT_P**. Вид окна **Double Trend** после этого показан на рис. 6.8. Выполните команду **Trend | Popup Trend**. Появится всплывающее окно **Popup Trend**. Аналогично тому, как это делалось ранее, настройте тренд на отображение тренд-тега **OT_P**. Вид окна **Popup Trend** после этого показан на рис. 6.9.

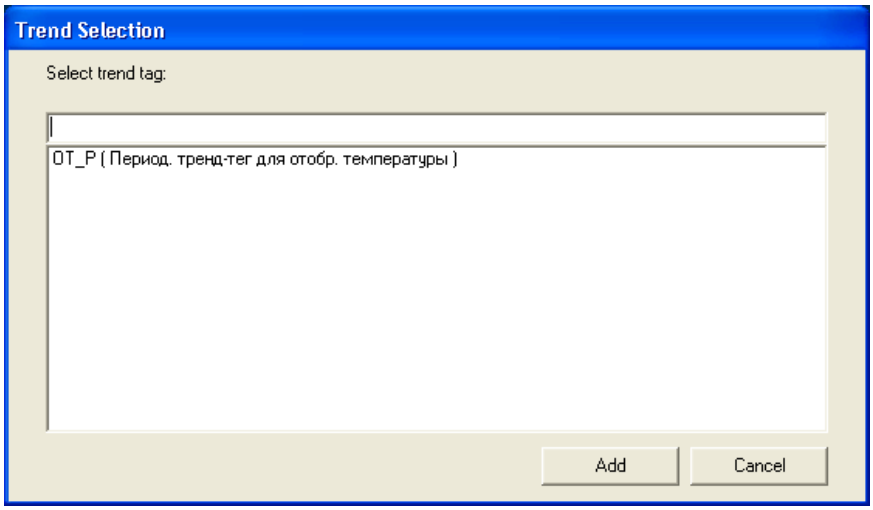


Рис. 6.6. Конфигурирование тренда **Single Trend** для отображения созданного тренд-тега **OT_P**

Совет

Сохраните проект **Training6** на магнитном диске. Для этого в среде приложения **Проводник Citect** достаточно выбрать в окне **Список проектов** этот проект и выполнить команду **Создание резервной копии…** его контекстного меню. В появившемся окне **Создание резервной копии проекта** следует указать требуемые параметры и нажать кнопку **OK**. Проект сохраните под именем **Training7.** В дальнейшем работайте с проектом **Training7**. Для этого достаточно выбрать **Мои проекты** в окне **Список проектов** и выполнить команду **Восстановление…** его контекстного меню. В появившемся окне **Восстановить проект** следует указать требуемые параметры и нажать кнопку **OK** (новый проект, имя проекта **Training7**).

6.2.1. Настройка отображения тренда. Архивные файлы трендов

Когда тренд отображается в окне первый раз, значения отображаются *динамически*, в режиме реального времени. Расположение линий графика представляет значения каждого тренд-тега. График сдвигается вдоль окна тренда при отображении новых значений.

Поскольку все данные тренда записываются на диск, то можно использовать архивный режим для обратной прокрутки и просмотра прошлых значений. В архивном режиме данные тренда отображаются *статически* и отображаются только значения за определенный период. Включать или выключать архивный режим можно с помощью отмечаемой кнопки в поле **Display histoty mode** (см. приведенный ранее рис. 6.5). Кнопки, расположенные слева от отмечаемой кнопки, позволяют прокручивать тренд назад к архивным данным или вперед к данным реального

времени. Щелкнув по изображению часов, можно ввести конечную дату для отображения архивных данных.

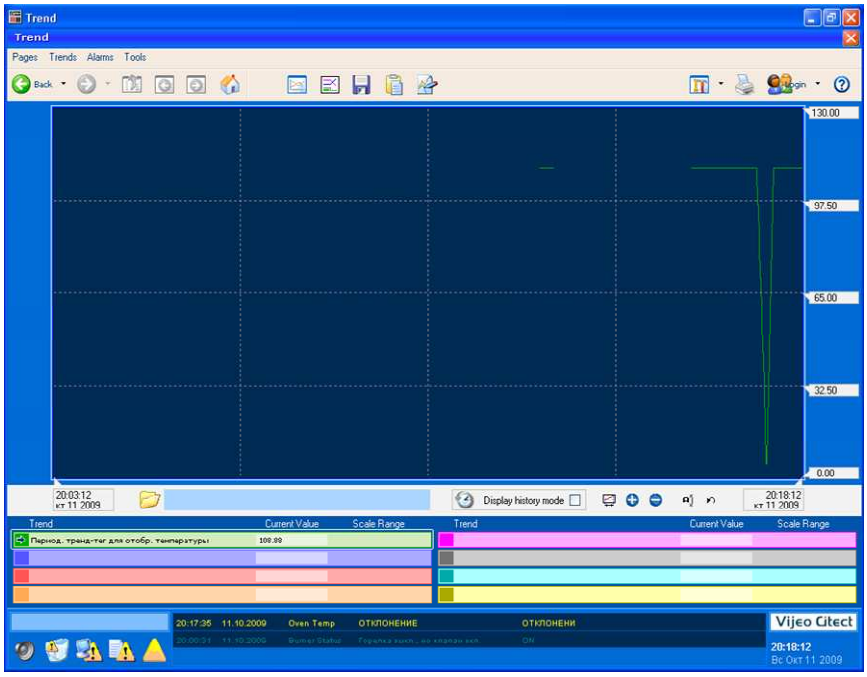


Рис. 6.7. Отображение созданного тренд-тега **OT_P**

Использование включаемого проекта **CSV_Include** позволяет использовать **Trend Groups** (группы трендов) для отображения определенного набора тренд-тегов. Одна группа включает в себя до восьми тегов, которые могут быть автоматически загружены в окно отображения тренда, исключая необходимость отдельного выбора каждого тега. Группирование трендов доступно только привилегированному пользователю.

Каждый элемент данных тренда, кроме трендов переменных с плавающей точкой, занимает при хранении два байта. С учетом сказанного, можно рассчитать для каждого тренд-тега место, занятое на магнитном диске за определенный период:

$$Q = 464 * N + 176 + (\frac{T * N * 2}{P})$$

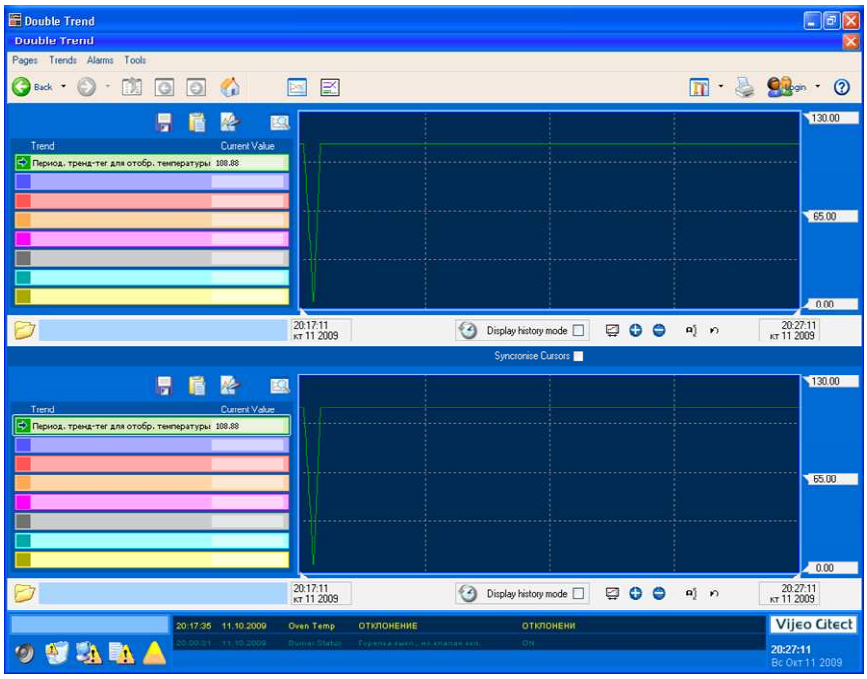


Рис. 6.8. Отображение созданного тренд-тега **OT_P** одновременно двумя трендами

Здесь Q — требуемый объем места на диске, байт; N — количество архивных файлов; T — период регистрации в одном архивном файле, секунд; P — интервал между соседними моментами регистрации, секунд.

Например, если регистрация данных тренда происходит каждые десять секунд в течение недели и используются пять файлов данных (пять недель), то необходимый объем места на диске будет составлять

$$Q = 464 * 5 + 176 + (\frac{(7 * 24 * 60 * 60) * 5 * 2}{10}) = 607296 \text{ байт}$$

Тренды переменных с плавающей точкой (восемь байт) требуют каждый в четыре раза больше места. Поэтому можно для них использовать предыдущую формулу и умножить результат на четыре.

Примечание

Запрещается удалять с жесткого диска архивные файлы, которые создает система Vijeo Citect, во время ее работы.

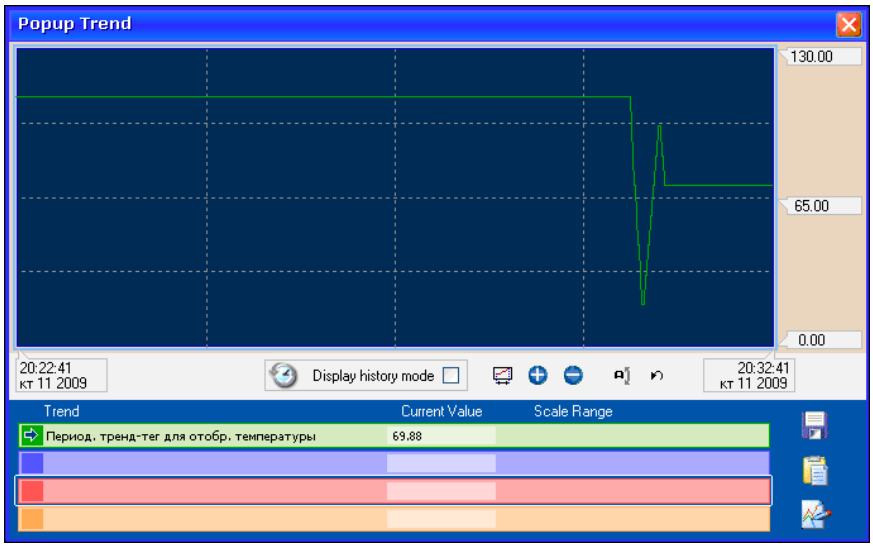


Рис. 6.9. Отображение тренд-тега **OT_P** во всплывающем окне

В большинстве случаев разумно создавать копии архивных файлов, например, для последующего анализа или сохранении достаточного пространства на сервере трендов.

Другой, более удобной формой отображения трендов является использование **Анализатора процессов**, который дает еще ряд дополнительных возможностей, таких, например, как отображение тревог, тегов переменных и локальных переменных. Анализатор процессов будет рассмотрен далее.

Глава 7. Команды и средства управления (Commands and Controls)

> ***Совет***
> Материал, относящийся к **Командам и средствам управления** (**Commands and Controls**) и их использованию можно найти в [2], темы Using Vijeo Citect | Defining and Drawing Graphics Pages |Defining Commands and Controls; [3], слайды 110 — 119.

Команды и средства управления позволяют операторам взаимодействовать с системой Vijeo Citect во время выполнения. Существуют три типа команд и средств управления — **Ползунковый переключатель (Регулятор)**, который позволяет изменять или отображать значение аналоговой переменной; **Команды ввода с помощью мыши** (**Touch commands**) и **Команды ввода с помощью клавиатуры** (**Keyboard commands**). Вы можете назначить привилегии и области действия для любой команды или элемента управления, посылать сообщения регистрации при всяком выполнении команды и записывать в журнал каждую команду оператора.

7.1. Ползунковый переключатель (Регулятор)

Ползунковый переключатель (**Регулятор**) позволяет оператору изменять значение аналоговой переменной путем перемещения графического объекта со свойством ползункового переключателя на графической странице. Положение ползункового переключателя будет также автоматически обновляться при изменении значения аналоговой переменной каким-либо иным способом.

Ползунковый переключатель можно передвигать горизонтально, вертикально или вращательно (Rotationally), что определяется его свойствами. Функция ползункового переключателя имеется у большинства графических объектов и задается с помощью вкладки **Регулятор**.

Упражнение 7.1. Создание и тестирование ползункового переключателя

В графической странице **Oven** проекта **Training7** для изменения значения аналогового тега **Oven_Temp** создайте ползунковый переключатель и протестируйте его работу. В среде приложения **Проводник Citect** выберите проект **Training7**, активизируйте приложение **Построитель графики Citect** и откройте графическую страницу **Oven**. В **Окне объектов** приложения **Построитель графики Citect** (см. приведенный ранее рис. 3.17) щелкните левой кнопкой мыши на кнопке **Вставить символ**, сразу же появится диалоговое окно **Вставить символ**. Настройте этот диалог в соответствии с рис. 7.1 и нажмите кнопку **OK**. Разместите ползунковый переключатель на графической странице **Oven** так, как указано на рис. 7.2 и

выполните команду **Свойства…** его контекстного меню. В результате появится диалоговое окно **Образ Properties** конфигурирования ползункового переключателя.

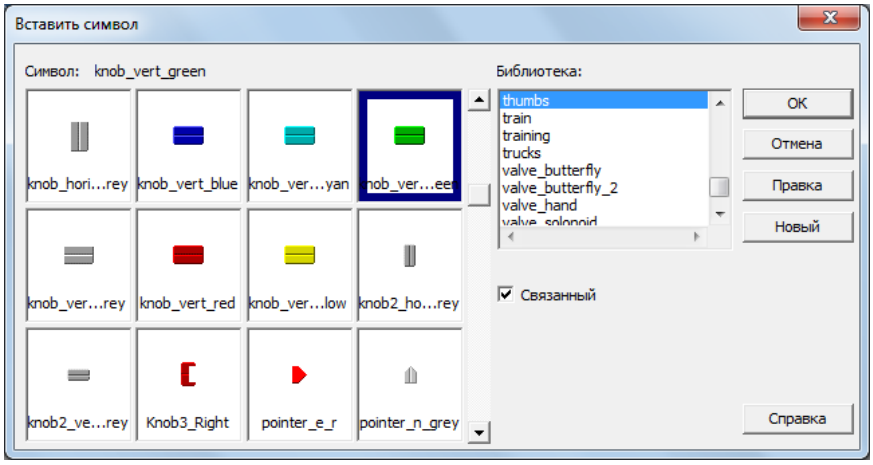


Рис. 7.1. Выбор графического объекта для ползункового переключателя

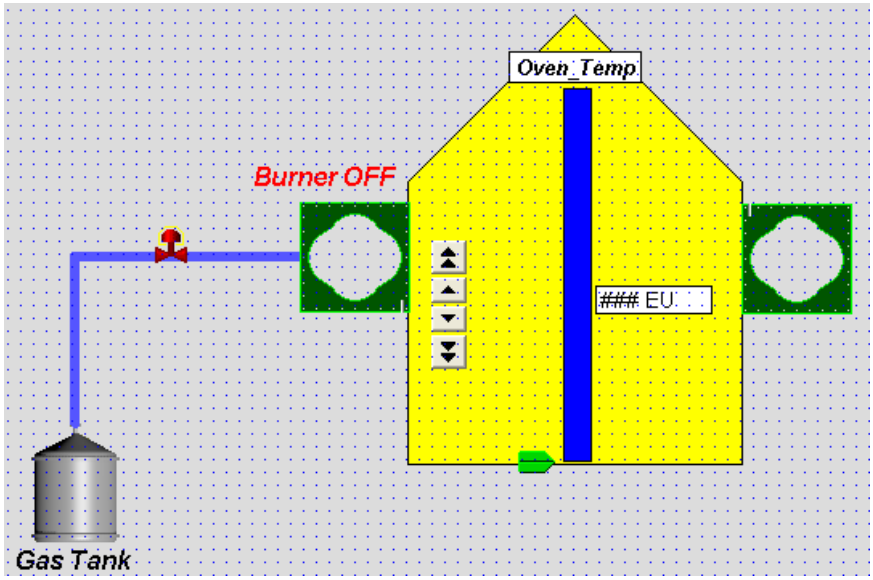


Рис. 7.2. Размещение ползункового переключателя на графической странице **Oven**

В окне диалога выберите вкладку **Регулятор** (**Вертикальный**), настройте ее в соответствии с рис. 7.3 и нажмите кнопку **Применить**. Обратите внимание, что в поле **При мак.:** указано значение **233**, соответствующее высоте в пикселях прямоугольника, отображающего столбиковую диаграмму. Теперь имеется возможность изменять значение аналогового тега как с помощью ползункового переключателя, так и с помощью управляющего элемента **Genie — Ramp Up Down Button**. Добавьте для ползункового переключателя всплывающую подсказку (рис. 7.4) и нажмите кнопку **OK**.

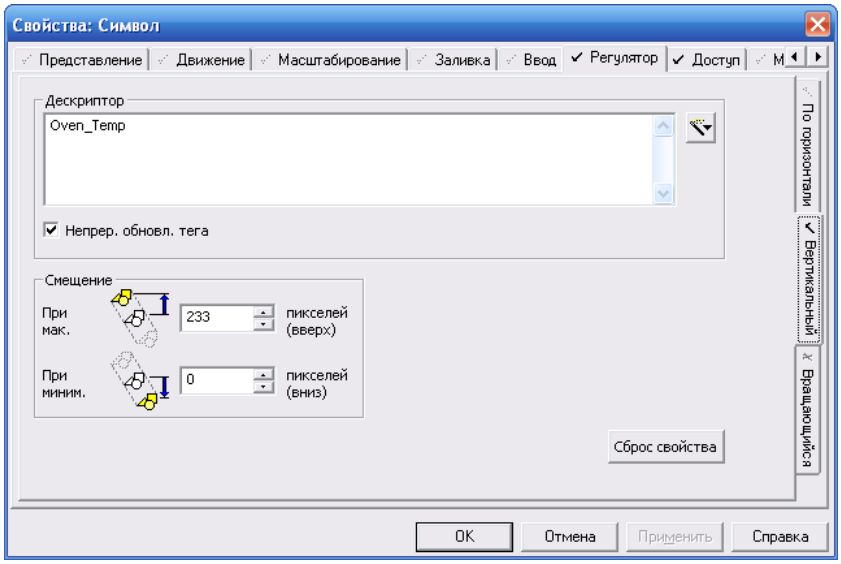


Рис. 7.3. Конфигурирование ползункового переключателя

Сохраните графическую страницу **Oven**, выполните компиляцию, запустите проект и протестируйте сделанные изменения. Обратите внимание, что при перемещении ползункового переключателя соответствующим образом изменяются высота закрашенного столбца слайдера и значение цифрового индикатора, отображающие значения аналогового тега **Oven_Temp**. При этом крайнее верхнее положение ползункового переключателя соответствует верхней границе столбиковой диаграммы. И еще проделайте не менее важный эксперимент. С помощью элемента **Genie — Ramp Up Down Button** изменяйте значение аналогового тега **Oven_Temp** и наблюдайте за соответствующим изменением положения ползункового переключателя. Убедитесь в наличии всплывающей подсказки. Завершите работу приложения.

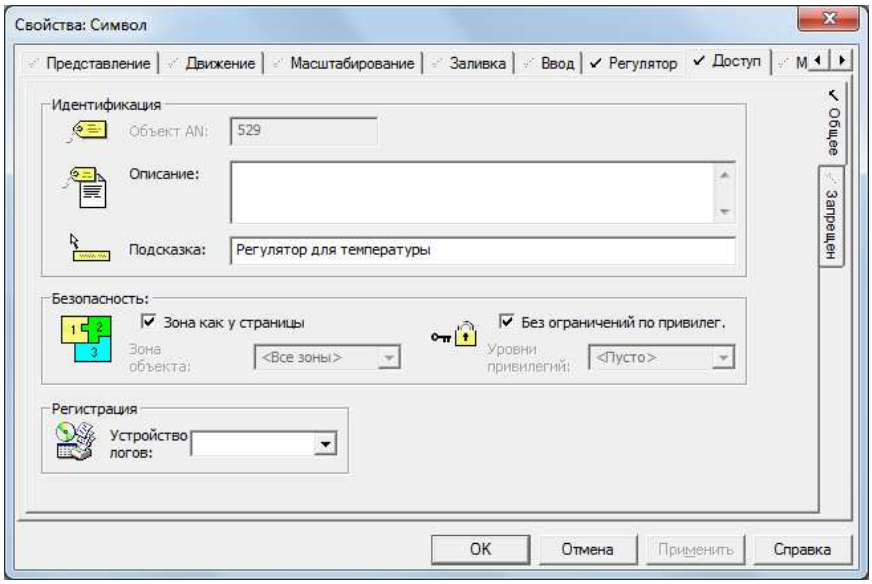


Рис. 7.4. Конфигурирование всплывающей подсказки для ползункового переключателя

7.2. Команды ввода с помощью мыши (Touch commands)

Команды ввода с помощью мыши (**Touch commands**) позволяют оператору выполнить одну команду или последовательность команд с помощью левой клавиши мыши, предварительно поместив курсор мыши на изображение графического объекта. Для графического объекта можно определить одновременно несколько команд — когда левая кнопка мыши нажата, либо отпущена, либо нажата в течение длительного времени (более одной или двух секунд). Этого можно достичь путем соответствующего задания свойств графического объекта во вкладке **Ввод** (**Касание**).

Пример использования команд ввода с помощью мыши приведен ранее в упр. 3.4 — 3.5.

7.3. Команды ввода с помощью клавиатуры (Keyboard commands)

Команды ввода с помощью клавиатуры (клавиатурные команды) представляют собой комбинацию клавиш, вводимую оператором, в ответ на которую выполняется заданная команда или последовательность команд.

Клавиатурные команды можно определить таким образом, чтобы они действовали на все графические страницы (**System keyboard commands — Системные клавиатурные команды**), или только при отображении конкретной графической страницы (**Page keyboard commands — Страничные клавиатурные команды**), или только при размещении указателя мыши над конкретным объектом на графической странице (**Object keyboard commands — Объектные клавиатурные команды**). Если для разных клавиатурных команд определена одна и та же клавиатурная комбинация, то конфликт разрешается путем исполнения клавиатурной команды, имеющей более высокий приоритет (системные клавиатурные команды имеют наивысший приоритет, а клавиатурные команды графического объекта имеют самый низкий приоритет).

Для доступа к клавиатурной комбинации (в частном случае к отдельной клавише) ее следует определить (снабдить некоторым именем). Например, клавишу **End** можно назвать именем Shutdown, клавишу **Home** — именем Home, клавишу **F11** — именем Info и т. д. Некоторые клавиши и клавиатурные комбинации в системе Vijeo Citect являются предопределенными.

> ***Совет***
> Подробнее об этом см. в [2], тема Technical Reference | Vijeo Citect Reference Information | Specifications | Predefined Commands.

Для *определения системной клавиши или клавиатурной комбинации* можно в среде приложения **Проводник Citect** в поле **Список проектов** раскрыть выбранный проект, "кликнуть" левой кнопкой мыши по элементу **Система** и в поле **Содержимое Система** дважды "кликнуть" левой кнопкой мыши по элементу **Клавиши клавиатуры**. Этой же цели можно достигнуть в среде приложения **Редактор проектов Citect** с помощью команды **Система | Клавиши клавиатуры**, что удобнее.

Для *определения системной клавиатурной команды или их последовательности* можно в среде приложения **Проводник Citect** в поле **Список проектов** раскрыть выбранный проект, "кликнуть" левой кнопкой мыши по элементу **Система** и в поле **Содержимое Система** дважды "кликнуть" левой кнопкой мыши по элементу **Клавиатурные команды**. Этой же цели можно достигнуть в среде приложения **Редактор проектов Citect** с помощью команды **Система | Клавиатурные команды**, что также удобнее (рис. 7.5). Системная клавиатурная команда или последовательность таких команд доступна во всех частях проекта (действует на все графические страницы).

Клавиатурные команды страницы напоминают системные клавиатурные команды, но они доступны только на тех графических страницах, для которых они определены. На одной графической странице может быть определено несколько клавиаткрных команд. Для *определения клавиатурной команды страницы* можно в среде приложения **Построитель графики Citect** открыть требуемую страницу (например, страницу **Oven**) и выполнить команду **Файл | Свойства**. В появившемся окне **Oven Properties** следует выбрать **Клавиатурные команды** (рис. 7.6).

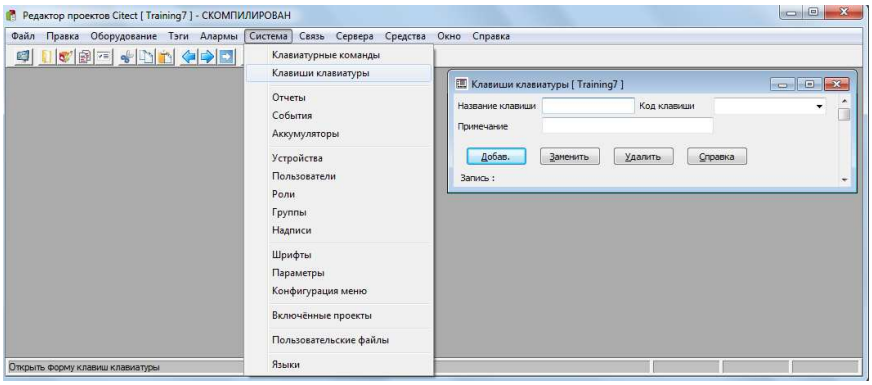


Рис. 7.5. Ввод с клавиатуры уровня системы

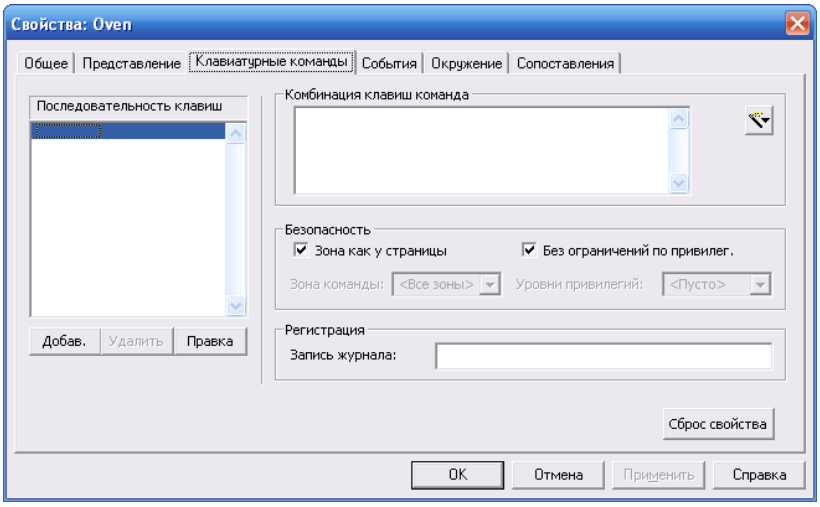


Рис. 7.6. Ввод с клавиатуры уровня графической страницы

Любой графический объект страницы может воспринимать ввод с клавиатуры точно так же, как и от мыши. Для *определения клавиатурной команды графического объекта* можно воспользоваться его свойством **Ввод (Клавиатурные команды)** (рис. 7.7).

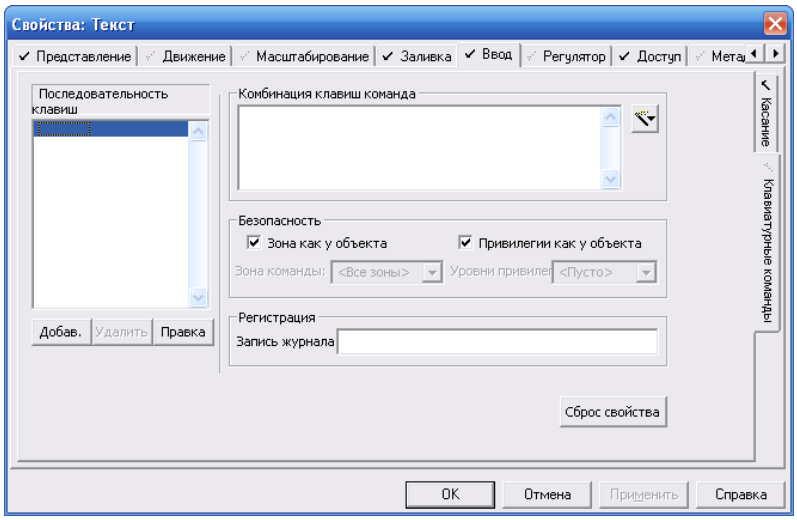


Рис. 7.7. Ввод с клавиатуры уровня графического объекта

7.3.1. Ввод с клавиатуры уровня графического объекта

Система Vijeo Citect не позволяет непосредственно присваивать значение, вводимое с помощью клавиатуры, пользовательскому тегу. Для этой цели используется служебный тег **Argvalue1**, являющийся внутренней переменной, в которой формируется значение, вводимое с помощью клавиатуры. В процессе формирования выполняется проверка корректности вводимых значений. Как только оператор нажимает клавишу **Enter**, введенное в служебный тег **Argvalue1** значение копируется в пользовательский тег. Пример клавишной последовательности: **###ENTER**. При этом в поле команды может стоять оператор **A = Argvalue1**.

Ввести несколько значений можно с помощью внутренних переменных-тегов **Arg1**, **Arg2**, … **Arg8**. Пример клавишной последовательности: **###,###ENTER**. При этом в поле команды могут стоять операторы: **A = Arg1; B = Arg2**. Но при вводе нескольких значений, в отличие от предыдущего случая, не производится проверка корректности вводимых значений.

Упражнение 7.2. Ввод с клавиатуры уровня графического объекта. Проект Training8

В графической странице **Oven** проекта **Training7** для изменения значения аналогового тега **Oven_Temp** создайте объект, поддерживающий клавиатурный ввод, и протестируйте его работу. В среде приложения **Проводник Citect** выберите проект **Training7**, активизируйте приложение **Построитель графики Citect** и откройте графическую страницу **Oven**. Пользуясь **Окном объектов** приложения **Построитель**

графики Citect (см. приведенный ранее рис. 3.17) разместите в графической странице **Oven** объект **Текст** так, как это показано на рис. 7.8. Для текстового объекта задайте только отображаемый текст.

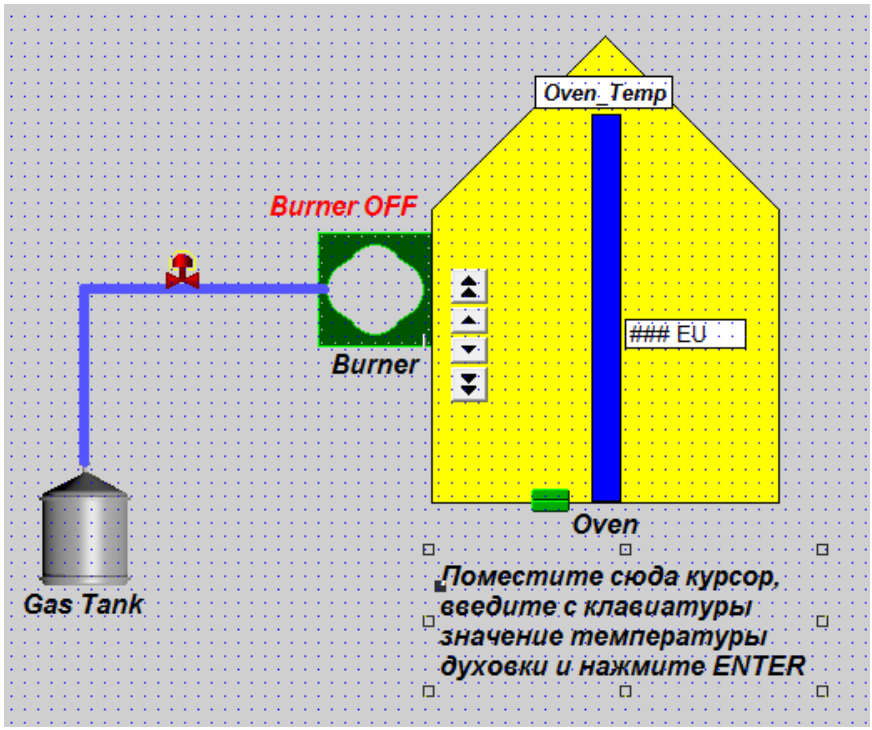


Рис. 7.8. Объект для клавиатурного ввода значения пользовательского тега **Oven_Temp**

Для обеспечения ввода значения тега **Oven_Temp** с помощью клавиатуры созданный текстовый объект настройте в соответствии с рис. 7.9 и нажмите клавишу **OK**. Теперь для изменения значения тега **Oven_Temp** достаточно переместить курсор мыши на текстовый объект, размещенный в рамке, ввести требуемое значение тега (вводимое значение будет отображаться по мере ввода во всплывающем окне) и нажать клавишу **Enter**.

Сохраните графическую страницу **Oven**, выполните компиляцию, запустите проект и протестируйте сделанные изменения. Завершите работу приложения.

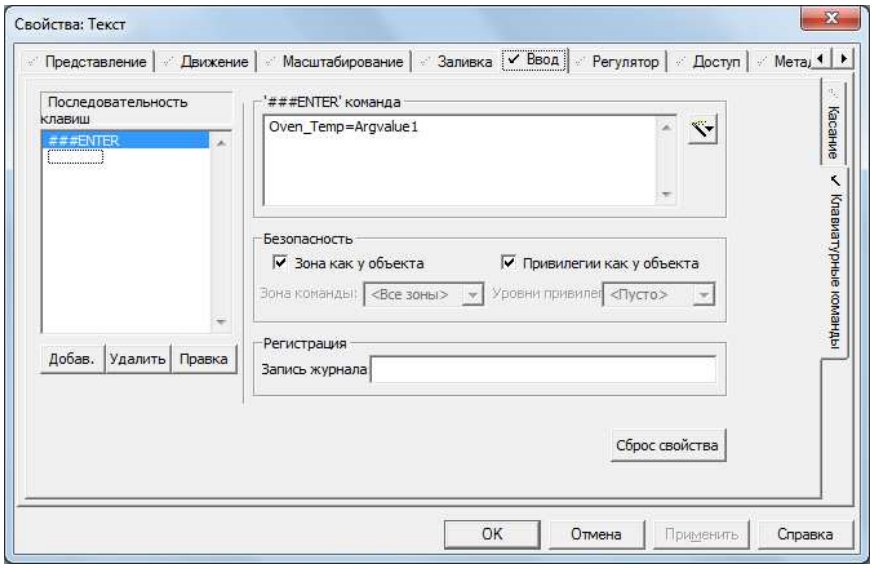


Рис. 7.9. Конфигурирование объекта **Поместите сюда курсор, введите с клавиатуры значение температуры духовки и нажмите ENTER** для клавиатурного ввода значения пользовательского тега **Oven_Temp**

Совет

Сохраните проект **Training7** на магнитном диске. Для этого в среде приложения **Проводник Citect** достаточно выбрать в окне **Список проектов** этот проект и выполнить команду **Создание резервной копии…** его контекстного меню. В появившемся окне **Создание резервной копии проекта** следует указать требуемые параметры и нажать кнопку **OK**. Проект сохраните под именем **Training8.** В дальнейшем работайте с проектом **Training8**. Для этого достаточно выбрать **Мои проекты** в окне **Список проектов** и выполнить команду **Восстановление…** его контекстного меню. В появившемся окне **Восстановить проект** следует указать требуемые параметры и нажать кнопку **OK** (новый проект, имя проекта **Training8**).

Примечание

Системные клавиатурные команды и клавиатурные команды страницы будут рассмотрены далее.

Глава 8. "Продвинутая" графика. Анализатор процессов (Process Analyst)

> ***Совет***
> Материал, относящийся к **Анализатору процессов** (**Process Analyst**) и его использованию можно найти в файле справки ..\Program Files\Schneider Electric\ Vijeo Citect\Vijeo Citect 7.30\Bin\ProcessAnalyst.chm.

Анализатор процессов системы Vijeo Citect является управляющим приложением ActiveX, которое позволяет операторам наблюдать данные трендов с сервера трендов, данные сигналов тревог с сервера тревог, *теги переменных и локальные переменные*. Анализатор процессов предоставляет визуальные средства для анализа и сравнения данных трендов (реального времени или архивных) и данных сигналов тревог в более понятной форме, чем шаблоны трендов и шаблоны отображения сигналов тревог.

8.1. Что представляет собой анализатор процессов?

Анализатор процессов может быть помещен в любую графическую страницу, причем в простейшем случае нет необходимости в его дополнительной настройке перед размещением в графической странице. Анализатор процессов способен выводить данные трендов, сигналов тревог, тегов переменных и локальных переменных в одном и том же окне (рис. 8.1).

Поскольку он использует информацию существующих серверов трендов и тревог, что и страницы трендов и сигналов тревог, то нет необходимости в дополнительной настройке, кроме создания соответствующих тегов. Вместе с тем, некоторые свойства анализатора процессов можно задавать и в период разработки.

Кратко рассмотрим назначение кнопок панелей инструментов анализатора процессов, начав с *главной панели инструментов*, расположенной в верхней части анализатора процессов:

Все кнопки панелей инструментов снабжены краткими всплывающими подсказками. Кнопки **Сохранить представление** (Save View) и **Загрузить представление** (Load View) позволяют соответственно сохранить текущую конфигурацию анализатора процессов в файле **.pav** на магнитном диске или восстановить сохраненный ранее вид. Кнопка **Печать** (Print) позволяет напечатать текущий вид анализатора процессов. Кнопки **Копировать в буфер обмена** (Copy to Clipboard) и **Копировать в файл** (Copy to File) позволяют соответственно скопировать текущее изображение анализатора в буфер промежуточного хранения или в файл на магнитном диске. Кнопка **Добавить кривые** (Add Pens) позволяет добавить новое

перо (перья). Кнопка **Удалить кривую** (Remove Pen) позволяет удалить выбранное перо. Кнопка **Сцепить/Расцепить кривые** (Lock/Unlock Pens) позволяет заблокировать или разблокировать вертикальное или горизонтальное перемещение перьев друг относительно друга, которое применяется для сравнения процессов. Кнопка **Показать/Скрыть точки** (Show/Hide Points) позволяет включить или скрыть точечное отображение процесса. Кнопка **Показать/Скрыть курсор** (Show/Hide Cursor) позволяет включить или скрыть отображение курсора. Кнопка **Показать/Скрыть курсорные метки** (Show/Hide Cursor Labels) позволяет включить или скрыть отображение меток курсора. Кнопка **Отображение списка кривых** (Toggle Object View) позволяет скрывать или отображать панель инструментов объектов. Кнопка **Показать свойства** (Show Properties) открывает окно конфигурирования анализатора процессов. Кнопка **Справка** (Help) вызывает справку анализатора процессов.

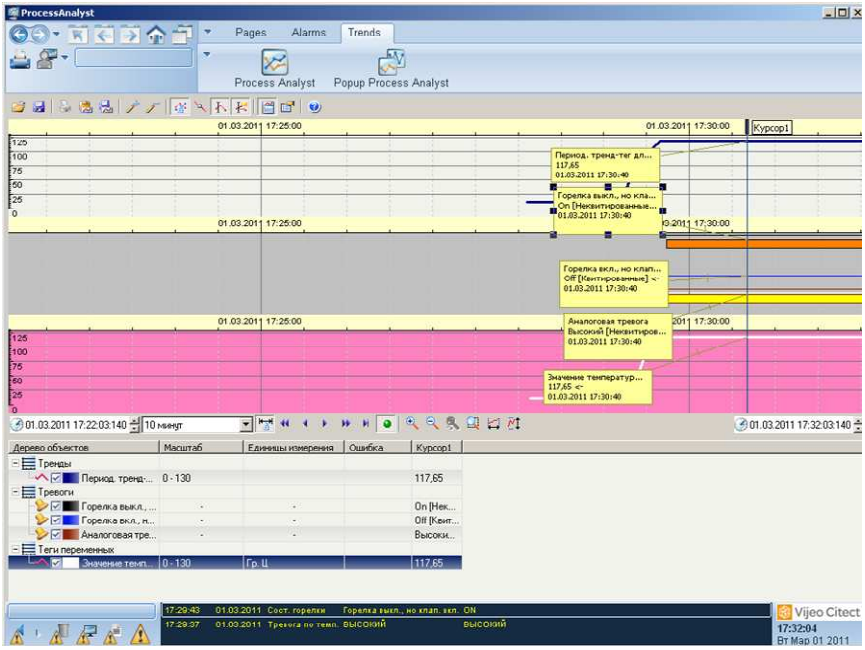


Рис. 8.1. Возможный вид **Анализатора процессов**

Совет

Для автоматического восстановления конфигурации **Анализатора процессов** используйте стандартные функции языка Cicode. С этой целью задайте свойства графической страницы, на которой размещен **Анализатор процессов**, в соответствии с рис. 8.2. Функция языка Cicode **LoadProcessAnalyst()** имеет вид, представленный на листинге 8.1.

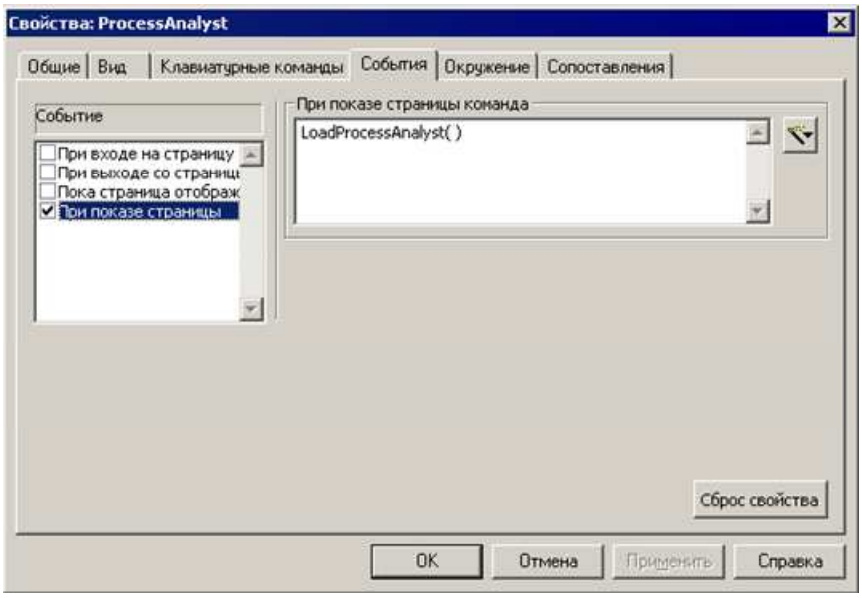


Рис. 8.2. Конфигурирование графической страницы **Oven** проекта **Training8** для автоматического восстановления конфигурации **Анализатора процессов**

Листинг 8.1. Функция, обеспечивающая автоматическое восстановление конфигурации анализатора процессов

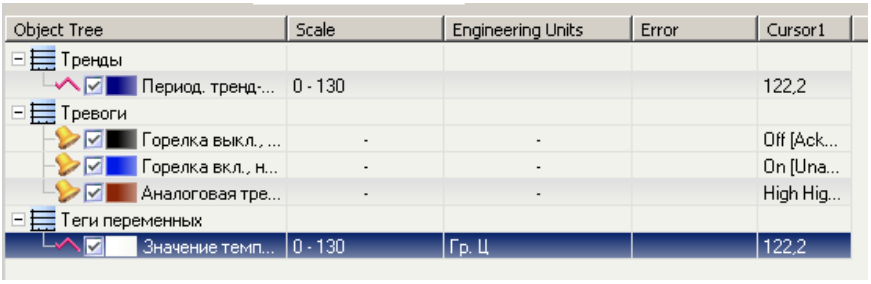
```
/* Восстановление ранее сохраненной в файле на магнитном диске
   конфигурации анализатора процессов */
FUNCTION LoadProcessAnalyst( )
    /* Получение дескриптора анализатора процессов: AN502 -
       анимационный номер анализатора процессов */
    OBJECT hProcessAnalyst = ObjectByName( "AN502" );
    /* Восстановление конфигурации анализатора процессов из файла
       pa.pav, помещенного ранее в подпапку Analyst Views папки
       проекта */
    _ObjectCallMethod( hProcessAnalyst, "LoadFromFile","pa.pav", 1 );
END
```

Теперь рассмотрим *навигационную панель инструментов*, расположенную под панелями отображения анализатора процессов:

В полях **Начальный момент времени** (Start Time) и **Конечный момент времени** (End Time), расположенных по краям панели, отображаются соответственно дата и время левой и правой границ временной оси анализатора процессов. В поле

Временной интервал (Time Span) отображается диапазон оси времени, который можно изменить с помощью кнопки, расположенной в правой части поля. Кнопка **Назад на один интервал** (Back One Span) обеспечивает переход по временной оси назад на интервал, указанный в поле **Временной интервал** (Time Span). Кнопка **Назад на полинтервала** (Back Half a Span) действует аналогично, но выполняет переход на половину временного интервала. Действие кнопок **Вперед на один интервал** (Forward One Span) и **Вперед на полинтервала** (Forward Half a Span) аналогично, но переход выполняется вперед. Кнопка **Синхронизировать** (Sinchronise to Now) инициализирует поле **Конечный момент времени** (End Time) текущим временем. Кнопка **Переключение автоматической прокрутки** (Toggle Auto-Scrolling) останавливает или включает режим прокрутки. Кнопки **Увеличить на 50%** (Zoom in 50%) и **Уменьшить на 50%** (Zoom out 50%) соответственно увеличивают или уменьшают масштаб по оси ординат на 50%. Кнопка **Установка длительности** (Edit Span) позволяет задать любое значение интервала временной оси, не используя фиксированный список интервалов. Кнопка **Изменить масштаб по вертикальной оси** (Edit Vertical Scale) позволяет изменить масштаб вертикальной оси (только для аналоговых перьев).

И, в заключение, несколько полезных замечаний о панели инструментов объектов, расположенной под навигационной панелью инструментов:



В столбце **Дерево объектов** (Object Tree) с помощью отмечаемых кнопок можно включать или скрывать перья. В столбце **Курсор1** (Cursor1), который появляется при включении курсора, отображается текущее значение пера.

Для добавления в графическую страницу анализатора процессов можно открыть приложение **Построитель графики Citect** и выполнить команду **Правка | Вставить анализатор процессов**.

Упражнение 8.1. Добавление анализатора процессов в графическую страницу

В графической странице **Oven** проекта **Training8** разместите управляющий элемент ActiveX **Анализатор процессов** для наблюдения изменений значений тегов трендов и сигналов тревог. В среде приложения **Проводник Citect** выберите проект **Training8**, активизируйте приложение **Построитель графики Citect** и откройте графическую страницу **Oven**. Пользуясь **Окном объектов** приложения **Построитель**

графики Citect (см. приведенный ранее рис. 3.17) воспользуйтесь кнопкой **Анализатор процессов** и разместите в графической странице **Oven** анализатор процессов. Появится диалоговое окно **Свойства: Citect Process Analyst Control**, которое пока закройте кнопкой **Отмена** и разместите **Анализатор процессов** в соответствии с рис. 8.3. Сохраните графическую страницу **Oven** и выполните компиляцию проекта.

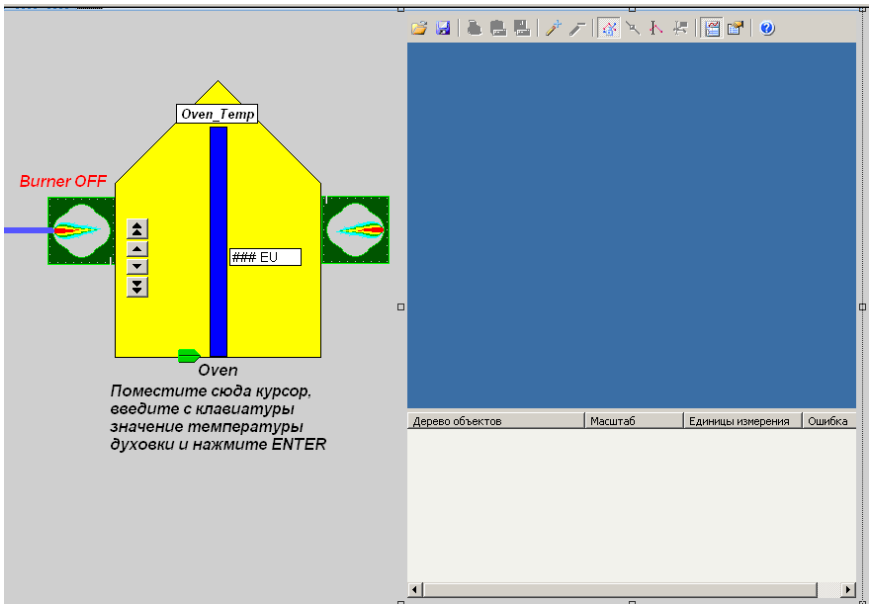


Рис. 8.3. Размещение **Анализатора процессов** в графической странице **Oven**

8.2. Свойства анализатора процессов. Отображение трендов, сигналов тревог и тегов переменных

Пользователи могут конфигурировать требуемые свойства анализатора процессов как в процессе работы (см. приведенный ранее рис. 8.1), так и при конструировании.

Упражнение 8.2. Настройка анализатора процессов при конструировании и выполнении. Отображение трендов, аналоговых, дискретных тревог и тегов переменных. Проект Training9

В графической странице **Oven** проекта **Training8** настройте анализатор процессов для наблюдения изменения значения тренд-тега и протестируйте его работу. В среде приложения **Проводник Citect** выберите проект **Training8**, перейдите в среду **Построителя графики Citect** и откройте графическую страницу **Oven**. Настройте анализатор процессов на отображение тренд-тега температуры в соответствии с рис. 8.4.

Дальнейшую настройку **Анализатора процессов** выполните в режиме исполнения. Для этого сохраните графическую страницу **Oven**, выполните компиляцию, запустите проект **Training8** и перейдите на графическую страницу **Oven**. В результате вы будете наблюдать график тренда температуры. Для отображения сигналов тревог на отдельной панели **Анализатора процессов** нажмите кнопку **Показать свойства** (Show Properties) на главной панели инструментов. Аналогично тому, как это было указано ранее на рис. 8.4, создайте панель **Тревоги**. Для отображения тревог на панели навигации (главной панели инструментов) **Анализатора процессов** нажмите кнопку **Добавить кривые** (Add Pens) и появится диалоговое окно **Добавление новых кривых** (Add New Pen(s)). Настройте диалоговое окно в соответствии с рис. 8.5. Для отображения тега переменной на отдельной панели **Анализатора процессов** нажмите кнопку **Показать свойства** (Show Properties) на главной панели инструментов. Аналогично тому, как это было указано ранее на рис. 8.4, создайте панель **Теги переменных**. Для отображения тега переменной на панели навигации (главной панели инструментов) **Анализатора процессов** нажмите кнопку **Добавить кривые** (Add Pens) и появится диалоговое окно **Добавление новых кривых** (Add New Pen(s)). Настройте диалоговое окно в соответствии с рис. 8.6. С помощью кнопки **Сохранить представление** (Save View) на главной панели инструментов **Анализатора процессов** сохраните конфигурацию в файле **Oven.pav**. Это позволит в дальнейшем при переходе на страницу **Oven** легко восстановить сохраненную конфигурацию **Анализатора процессов**, используя кнопку **Загрузить представление** (Load View) на главной пенели инструментов.

Рассмотрим более подробно работу с диалоговым окном **Добавление новых кривых** (Add New Pen(s)). В поле **Тип:** (Type) можно выбрать **Алармы** (Alarms), **Тренды** (Trends), **Переменные теги** (Variable Tags) или **Локальные переменные** (Local Variables). Кнопка **Поиск** (Search) позволяет выполнить поиск доступных трендов или сигналов тревог, результаты поиска помещаются в поле **Результаты поиска** (Search Results) в виде списка, в котором можно выбрать нужный тренд или сигнал тревоги для его конфигурирования. В поле **Добавить кривые в:** (Add Pens to:) можно выбрать панель, в которой будет отображаться выбранный тренд, сигнал тревоги, тег переменной или локальная переменная. В поле **Тип кривой:** (Pen Type:) можно выбрать тип пера — аналоговый, цифровой или для сигнала тревоги. В поле

Название кривой: (Pen Name:) можно задать название пера. Кнопка **Добавить** (Add) позволяет перенести выбранные в поле **Результаты поиска** (Search Results) элементы в поле **Выбранные элементы** (Selected Items).

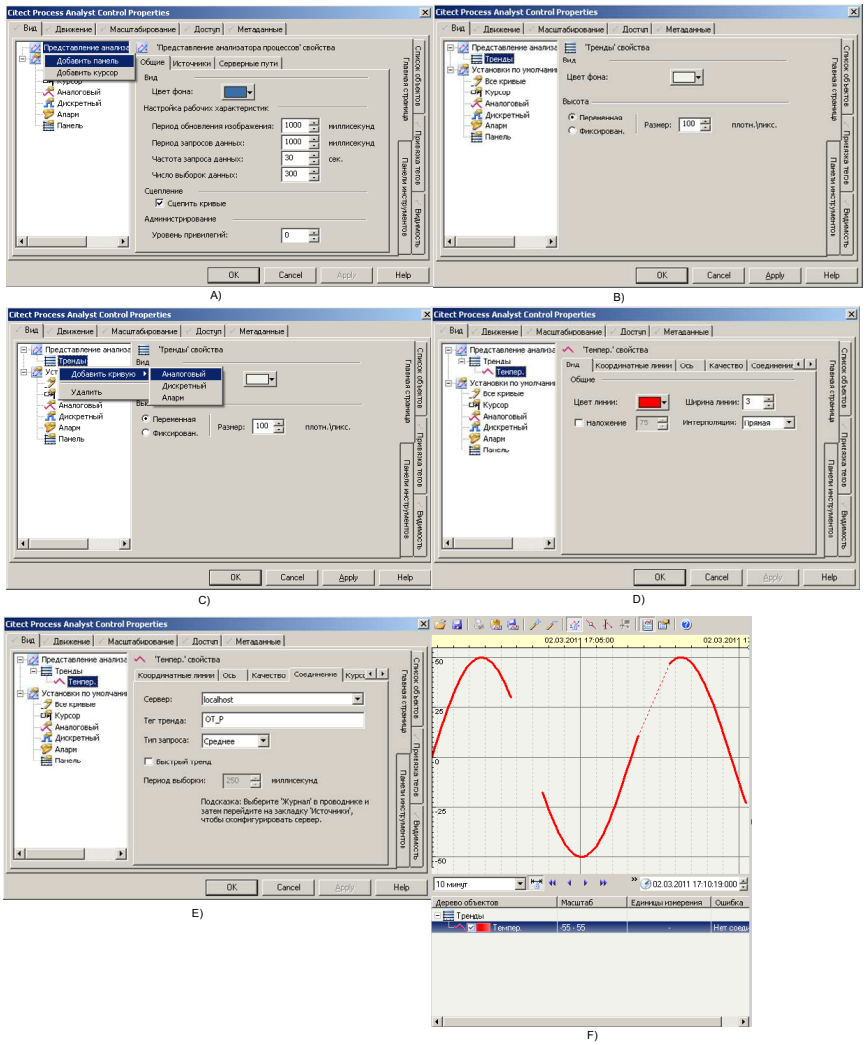


Рис. 8.4. Настройка **Анализатора процессов** в режиме конструирования (отображение тренд-тега температуры на отдельной панели)

С помощью кнопки **Справка** (Help) можно вызвать справку анализатора процессов, а с помощью кнопки **OK** тренды или сигналы тревог, представленные списком в окне **Выбранные элементы** (Selected Items) включаются для отображения в анализаторе процессов.

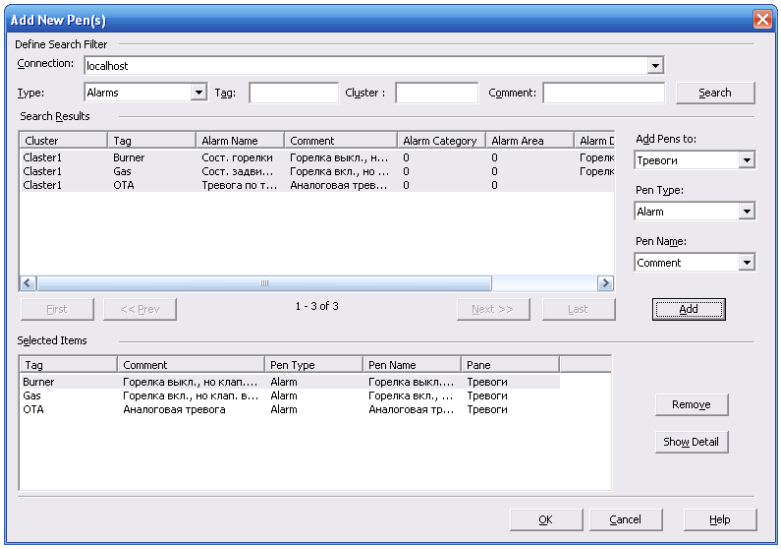


Рис. 8.5. Диалоговое окно добавления тревог в **Анализатор процессов**

В окне **Добавление новых кривых** (Add New Pen(s)), см. приведенный ранее рис. 8.5) нажмите кнопку **OK.** Изменяйте значение аналогового тега **Oven_Temp**, например, с помощью ползункового переключателя, состояния горелки и вентиля подачи газа и пронаблюдайте поведение отображаемых тревог на **Анализаторе процессов**. Возможный вид **Анализатора процессов** после выполнения указанных действий представлен на рис. 8.7.

Поэкспериментируйте, нажимая различные кнопки на панели инструментов навигации (**Временной интервал**, **Фиксация интервала**, **Назад на один интервал**, **Назад на полинтервала**, **Вперед на полинтервала** и др.) и оцените реакцию на их нажатие.

Поэкспериментируйте, нажимая различные кнопки на главной панели инструментов (**Показать/Скрыть точки**, **Показать/Скрыть курсор**, **Показать/Скрыть курсорные метки**, **Отображение списка кривых**, **Показать свойства** и др) и оцените реакцию на их нажатие. Перейдите в область отображения **Анализатора процессов**, нажмите левую кнопку мыши и не отпуская ее попробуйте передвигать мышку влево или вправо. Синхронно с этим область отображения будет также передвигаться по шкале времени. Таким образом, вы сможете просматривать архивные данные.

На графической странице **Oven** в меню **Trends** (см. приведенный ранее рис. 8.1) имеется две команды: **Process Analyst** и **Popup Process Analyst**.

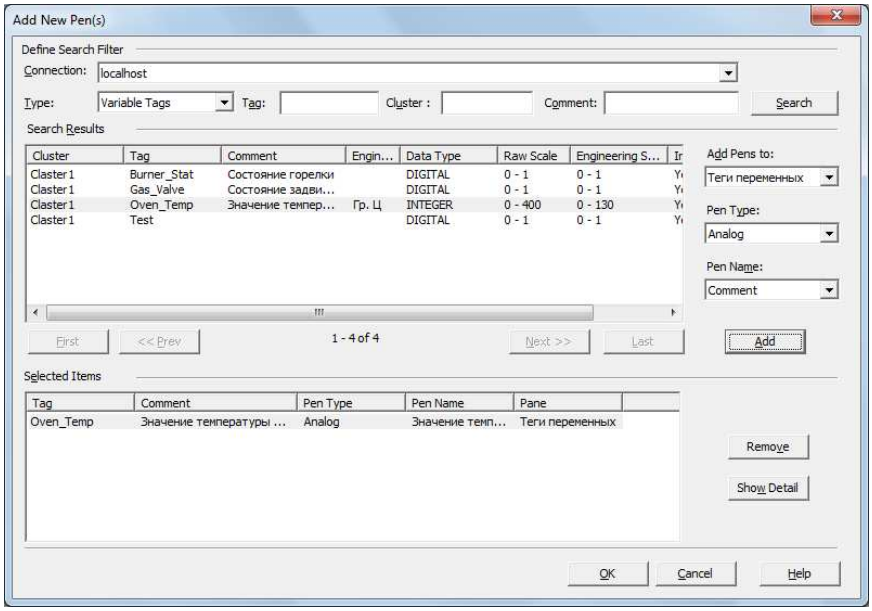


Рис. 8.6. Диалоговое окно добавления тега переменной в **Анализатор процессов**

Команда **Trends | Process Analyst** предназначена для работы **Анализатора процессов** в полностраничном режиме. Чтобы эта команда стала актуальной, в проекте **Training8** создайте графическую страницу на основе шаблона **normal** (стиль **tab_style_1**, разрешение **XGA**), разместите на ней **Анализатор процессов**, заняв всю площадь графической страницы, сохраните графическую страницу под именем **ProcessAnalyst**, выполните компиляцию и запустите проект **Training8.** На графической странице **Oven** выполните команду **Trends | Process Analyst**. В появившейся странице **ProcessAnalyst** настройте **Анализатор процессов** точно так же, как это было сделано ранее для страницы **Oven**, и сохраните полученную конфигурацию (см. приведенный ранее рис. 8.1) в файле **pa.pav**. Остановите работу проекта. Способом, указанным ранее на рис. 8.2 и листинге 8.1 обеспечьте автоматическое восстановление конфигурации **Анализатора процессов** при переходе на графическую страницу **ProcessAnalyst** и протестируйте внесенные изменения. Для этого сохраните графическую страницу **ProcessAnalyst**, выполните компиляцию, запустите проект, перейдите в графическую страницу **Oven** и выполните команду **Trends | Process Analyst**. В результате вы должны наблюдать состояние **Анализатора процессов**, аналогичное показанному ранее на рис. 8.1.

Команда **Trends | Popup Process Analyst** предназначена для работы **Анализатора процессов** во всплывающем окне.

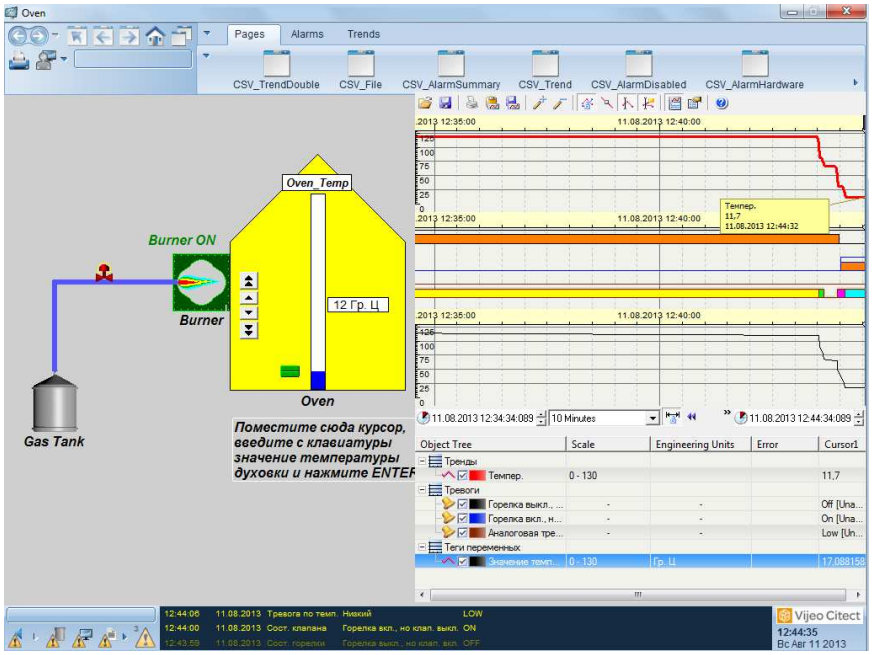


Рис. 8.7. Отображение трендов, аналоговых, дискретных тревог и тегов переменных в **Анализаторе процессов**

Чтобы эта команда стала актуальной, в проекте **Training8** создайте графическую страницу на основе шаблона **poppa** (стиль **tab_style_1**, разрешение **XGA**), сохраните графическую страницу под именем **!ProcessAnalystPopup**, выполните компиляцию и запустите проект **Training8.** На графической странице **Oven** выполните команду **Trends | Popup Process Analyst**. В появившейся всплывающей странице **!ProcessAnalystPopup** настройте **Анализатор процессов** точно так же, как это было сделано ранее для страницы **Oven**, и сохраните полученную конфигурацию (см. приведенный ранее рис. 8.1) в файле **poppa.pav**. Возможный вид **Анализатора процессов** во всплывающем окне **!ProcessAnalystPopup** иллюстрирует рис. 8.8. Остановите работу проекта. Повторно запустите проект, перейдите в графическую страницу **Oven** и выполните команду **Trends | Popup Process Analyst**. С помощью кнопки **Загрузить представление** (Load View), расположенной на главной панели инструментов, восстановите конфигурацию **Анализатора процессов**. В результате вы должны наблюдать состояние **Анализатора процессов**, аналогичное показанному ранее на рис. 8.8.

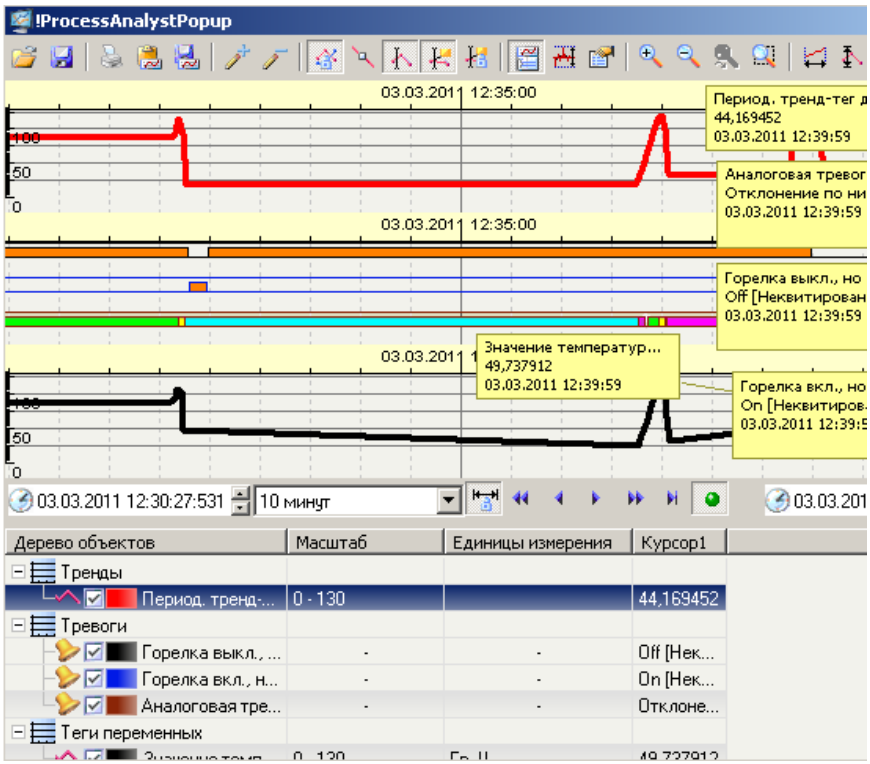


Рис. 8.8. Отображение трендов, аналоговых, дискретных тревог и тегов переменных в **Анализаторе процессов**, работающем во всплывающем окне

Завершите работу проекта.

Совет

Сохраните проект **Training8** на магнитном диске. Для этого в среде приложения **Проводник Citect** достаточно выбрать в окне **Список проектов** этот проект и выполнить команду **Создание резервной копии…** его контекстного меню. В появившемся окне **Создание резервной копии проекта** следует указать требуемые параметры и нажать кнопку **OK**. Проект сохраните под именем **Training9.** В дальнейшем работайте с проектом **Training9**. Для этого достаточно выбрать **Мои проекты** в окне **Список проектов** и выполнить команду **Восстановление…** его контекстного меню. В появившемся окне **Восстановить проект** следует указать требуемые параметры и нажать кнопку **OK** (новый проект, имя проекта **Training9**).

8.3. Типы перьев анализатора процессов и настройка свойств анализатора процессов в период исполнения

Перья анализатора процессов служат для отображения данных как трендов, так и сигналов тревог, тегов переменных и локальных переменных. Анализатор процессов поддерживает три типа данных — аналоговые, цифровые и перья сигналов тревог.

Каждое перо имеет собственное графическое представление. Настраивать большинство свойств пера можно во время исполнения.

Анализатор процессов обычно использует *аналоговые перья* для отображения данных, не являющихся двоичными. По этой причине только аналоговые перья имеют вертикальную ось значений, относительно которой строятся графики данных (см., например, приведенный ранее рис. 8.1).

> ***Совет***
> Для получения дополнительной информации об аналоговых перьях смотрите тему справки Process Analyst for Operators | Understanding Process Analyst Pens | Pen Types | Analog Pens.

Анализатор процессов обычно использует *цифровые перья* для отображения двоичных данных. Значения такого пера заключены в диапазоне от 0 до 1. Заливка цветом используется для индикации данных, имеющих единичное значение.

> ***Совет***
> Для получения дополнительной информации о цифровых перьях смотрите тему справки Process Analyst for Operators | Understanding Process Analyst Pens | Pen Types | Digital Pens.

Анализатор процессов использует *перья сигналов тревог* для графического отображения текущих значений или истории сигналов тревог в течение какого-либо времени. Включение, выключение сигналов тревог и их подтверждение представляются графически (рис. 8.9).

> ***Совет***
> Для получения дополнительной информации о перьях сигналов тревог смотрите тему справки Process Analyst for Operators | Understanding Process Analyst Pens | Pen Types | Alarm Pens.

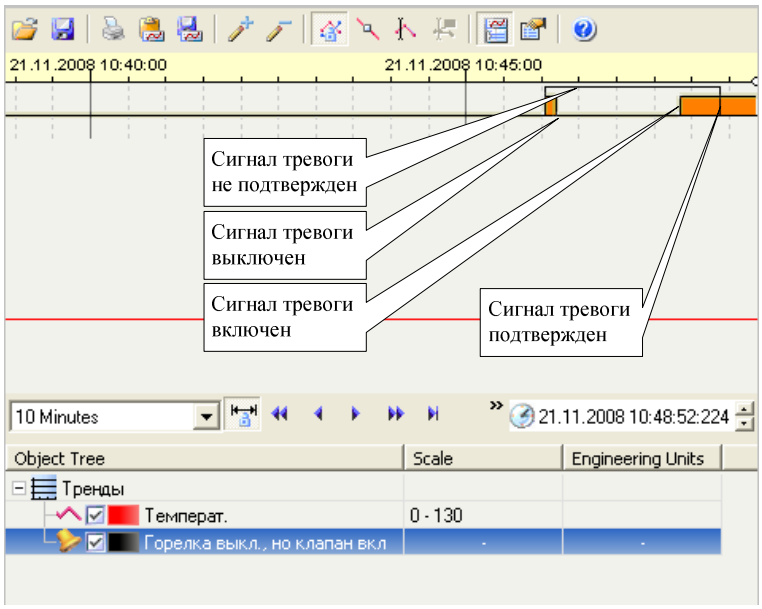


Рис. 8.9. Отображение дискретной тревоги **Горелка выкл., но клапан вкл.** проекта **Training9** в **Анализаторе процессов**

Для настройки свойств анализатора процессов во время исполнения следует нажать кнопку **Показать Свойства** (Show Properties) на главной панели инструментов. В результате появится диалоговое окно **Свойства: Process Analyst Control**, представленное на рис. 8.10.

Это окно содержит три вкладки. Вкладка **Главная страница** (Main Page) позволяет добавлять и конфигурировать панели отображения и курсоры, удалять панели и курсоры, добавлять и конфигурировать перья, удалять перья и задавать свойства перечисленных элементов, используемых по умолчанию. Вкладка **Панели инструментов** (Toolbars) позволяет определять состав кнопок на главной и навигационной панелях инструментов. Вкладка **Список объектов** (Object View) позволяет конфигурировать состав столбцов в списке объектов.

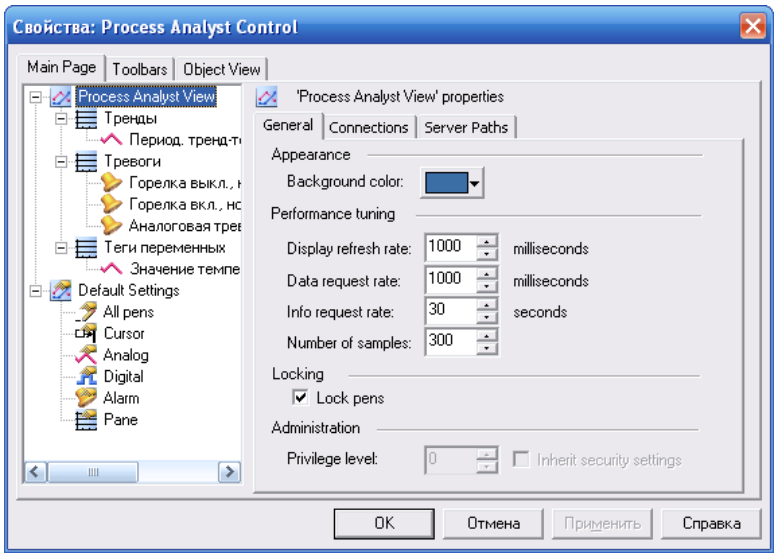


Рис. 8.10. Задание свойств **Анализатора процессов** в период исполнения

Глава 9. "Продвинутая" графика. Изображения, джины (Genie) и всплывающие страницы

Для завершения обсуждения графики системы Vijeo Citect последовательно рассмотрим такие "продвинутые" средства графики, как импорт графических изображений, настройка цветов, использование графических изображений в качестве фона страницы, создание пользовательских джинов и всплывающих страниц.

9.1. Графическая страница. Импорт графики, настройка цветов и использование графических изображений в качестве фона

> ***Совет***
> Материал, относящийся к данной теме можно найти в [2], тема Using Vijeo Citect | Defining and Drawning Graphics Pages | Understanding the Drawning Environment; [3], слайды 96 — 102.

"Продвинутая" графика системы Vijeo Citect позволяет оператору импортировать изображения с цифровых камер, программного обеспечения сторонних производителей изображений и из других источников без потерь данных. В графических изображениях проекта можно заменять одни цвета и оттенки другими (команда **Средства | Заменить цвета…**), изменять яркость, насыщенность и цветовой диапазон (команда **Средства | Настроить цвета…**) и добавлять в палитру часто используемые нестандартные оттенки (команда **Средства | Редактировать предпочтительные цвета…**) для облегчения доступа к ним в пределах проекта.

В любой графической странице можно использовать изображение из внешних источников путем импорта разнообразных типов графических файлов, включая файлы с расширениями **`.bmp`**, **`.dxf`**, **`.dib`**, **`.rle`**, **`.dcx`**, **`.dxf`**, **`.eps`**, **`.img`**, **`.jpg`**, **`.jge`**, **`.jfi`**, **`jff`**, **`.pcd`**, **`.pcx`**, **`.png`**, **`.tga`**, **`.tif`**, **`.wpg`**, **`.wmf`**. можно импортировать также и текстовые файлы с расширением **`.txt`**. Цвета в импортированном файле можно настроить, а файл использовать в качестве фонового изображения графической страницы или в качестве пиктограммы кнопки на панели инструментов.

Упражнение 9.1. Импорт графики и замена цвета

В среде приложения **Проводник Citect** выберите проект **Training9**, перейдите в среду **Построителя графики Citect** и, используя кнопку **Новый** на панели инструментов, создайте новую графическую страницу **Lighting** с параметрами,

указанными по умолчанию. Для созданной страницы используйте белый фон. Воспользуйтесь командой **Файл | Импорт…** для импорта файла ..\Демонстрационные примеры и сопутствующие файлы\FloorPlant.bmp, отображающего план завода, и нажмите кнопку **Open** (Открыть). Воспользуйтесь командой **Средства | Заменить цвета…** для замены пурпурного на прозрачный цвет (рис. 9.1). Для этого в окне **Замена цвета** кликните левой кнопкой мыши по раскрывающемуся списку в поле **Из**, нажмите кнопку в левом нижнем углу с пиктограммой в виде пипетки, переместите курсор на фон импортируемого рисунка, нажмите левую кнопку мыши и еще раз "кликните" по раскрывающемуся списку в поле **Из**. В поле **До** в качестве прозрачного цвета аналогично выберите белый цвет, совпадающий с фоновым цветом графической страницы. Нажмите кнопку **OK**.

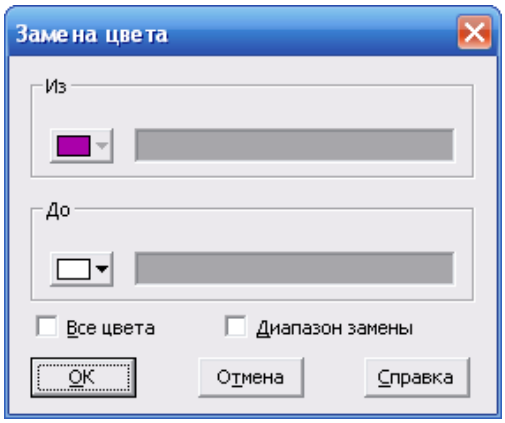


Рис. 9.1. Замена пурпурного цвета на прозрачный в файле, импортированном в графическую страницу **Lighting**

Сохраните созданную графическую страницу, выполните компиляцию проекта и запустите проект. Посмотрите созданную графическую страницу и завершите работу проекта.

В реальной графике используется около 16.7 млн. разных оттенков и настройка диапазона цветов по одному за раз является слишком трудоемкой. В этом случае удобно использовать операцию **Средства | Настроить цвета…**, которая дает пользователю возможность выбирать полный диапазон настраиваемых оттенков, а также изменять яркость и насыщенность избранных оттенков и всего изображения.

Упражнение 9.2. Осветление синих оттенков

На графической странице **Lighting** проекта **Training9** выберите изображение **FloorPlant**. Для настройки цветов воспользуйтесь командой **Средства | Настроить цвета…** и сконфигурируйте настройку цветов в соответствии с рис. 9.2. Нажмите кнопку **OK** и синяя административная часть на рисунке станет

ярче, а разделительные линии будут более различимыми. Сохраните созданную графическую страницу, выполните компиляцию проекта и запустите проект. Посмотрите созданную графическую страницу и завершите работу проекта.

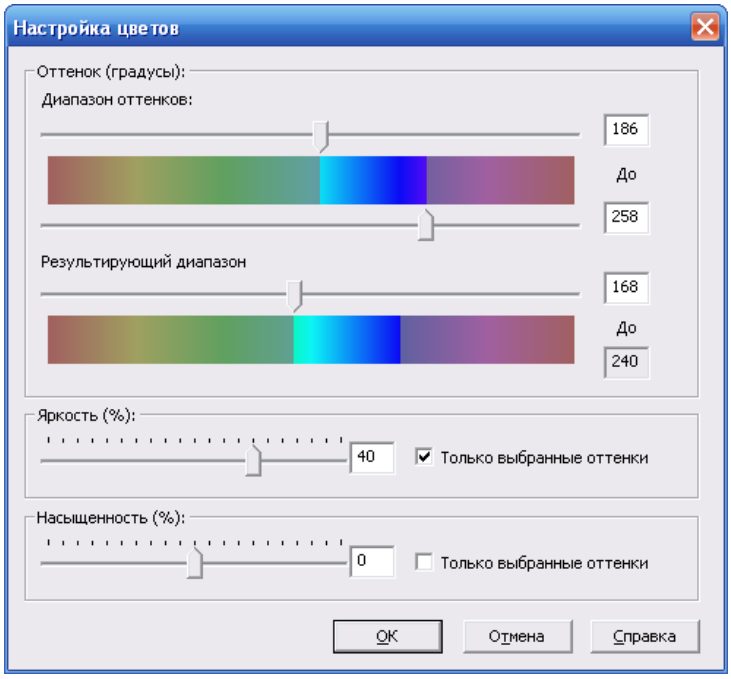


Рис. 9.2. Настройка цветов изображения **FloorPlant** графической страницы **Lighting** проекта **Training9**

При использовании изображения в качестве статического фона бывает полезно его заблокировать, чтобы его невозможно было выбрать в среде приложения **Построитель графики Citect**. Это позволит оператору работать с другими графическими объектами поверх фонового изображения, не боясь случайно выбрать само фоновое изображение.

Упражнение 9.3. Изображение как фон графической страницы. Проект Training10

На графической странице **Lighting** проекта **Training9** выберите изображение **FloorPlant**. Для фиксации изображения в качестве фона графической страницы воспользуйтесь командой **Правка | Фиксировать объект**.

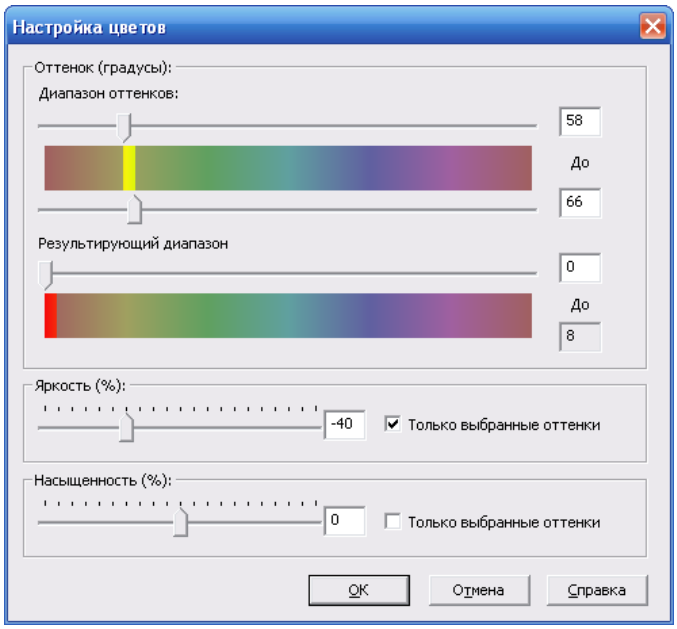


Рис. 9.3. Настройка цветов изображения **FloorPlant** графической страницы **Lighting** проекта **Training9**

Замечание

Для отмены фиксации изображения в качестве фона следует выполнить команду **Правка | Не учитывать фиксацию**.

Сохраните созданную графическую страницу, выполните компиляцию проекта и запустите проект. Посмотрите созданную графическую страницу и завершите работу проекта.

Совет

Сохраните проект **Training9** на магнитном диске. Для этого в среде приложения **Проводник Citect** достаточно выбрать в окне **Список проектов** этот проект и выполнить команду **Создание резервной копии…** его контекстного меню. В появившемся окне **Создание резервной копии проекта** следует указать требуемые параметры и нажать кнопку **OK**. Проект сохраните под именем **Training10.** В дальнейшем работайте с проектом **Training10**. Для этого достаточно выбрать **Мои проекты** в окне **Список проектов** и выполнить команду **Восстановление…** его контекстного меню. В появившемся окне **Восстановить проект** следует указать требуемые параметры и нажать кнопку **OK** (новый проект, имя проекта **Training10**).

9.2. Джины (Genies)

Совет

Материал, относящийся к данной теме можно найти в [2], тема Using Vijeo Citect | Genies and Super Genies; [4], слайды 125 — 139.

Обычно каждый графический объект графической страницы конфигурируется отдельно. *Джины (Genies)* позволяют скомпоновать несколько связанных между собой объектов в группу и сохранять группу в библиотеке genie, подобной библиотеке символов. После этого джин можно использовать как единый объект (вставлять, перемещать, изменять размер и т. д.), а элементы джина конфигурируются в совокупности друг с другом.

Например, можно определить джин для устройства пуска/останова с кнопками пуска и останова и индикаторной лампочкой и применять один и тот же джин для разнообразного оборудования — насосов, конвейеров и т. п. При использовании джинов нужно лишь указать информацию, уникальную для данного конкретного насоса или конвейера (тег переменной).

Включаемые проекты, входящие в комплект базовой установки системы Vijeo Citect, имеют несколько библиотек с джинами, которые предназначены для использования в проектах. Обычно джины определяются без конкретных тегов переменных, что позволяет применять их как в данном проекте, так и в других проектах. При вставке джина на графическую страницу появится диалоговое окно с запросом на ввод одного или нескольких тегов переменных и комментариев.

Замечание

Пример использования джина из библиотеки включаемых проектов для изменения значения температуры духовки был рассмотрен ранее в упр. 4.9 (проект **Training3**).

В графических страницах можно не только использовать стандартные графические объекты **Джины**, но и создавать и использовать собственные джины.

Упражнение 9.4. Создание пользовательского джина для управления задвижкой подачи газа

В среде приложения **Построитель графики Citect** при выбранном проекте **Training10** нажмите кнопку **Новый** на панели инструментов и в появившемся диалоговом окне **Новый** для создания пользовательского **Джина** нажмите кнопку **Джин**. В появившуюся страницу поместите набор символов (**Набор образов**) и сконфигурируйте его в соответствии с рис. 9.4 — 9.6.

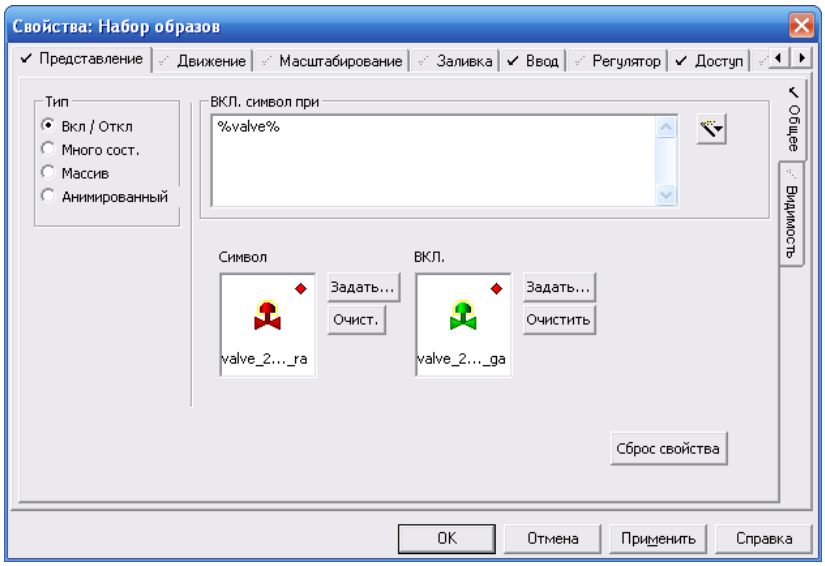


Рис. 9.4. Свойство **Вид (Общие)** набора символов

В эту же страницу поместите, справа от набора символов, графический объект **Число** и сконфигурируйте его в соответствии с рис. 9.7. Справа от графического объекта **Число** поместите графический объект **Текст** и сконфигурируйте его в соответствии с рис. 9.8 — 9.9. Над размещенными графическими объектами поместите графический объект **Текст** и сконфигурируйте его в соответствии с рис. 9.10. Сгруппируйте все графические объекты с помощью графического объекта **Прямоугольник** (рис. 9.11) и выполните команду **Порядок | Группировать объекты**. Левый верхний угол сгруппированного объекта совместите с точкой привязки, расположенной в центре графической страницы.

Командой **Файл | Сохранить как…**сохраните графический объект **Джин**. Для этого в окне **Сохранить как** нажмите кнопку **Новый**. В появившемся окне **Новая библиотека** в поле **Название:** укажите **Training**, нажмите кнопку **OK**, а в поле **Джин:** окна **Сохранить как** укажите **ValveControl** и нажмите кнопку **OK**. Разместите графический объект **Training.ValveContol** в графической странице **Oven**. В окне его свойств поместите **Gas_Valve** и нажмите кнопку **OK**. Сохраните графическую страницу **Oven**, выполните компиляцию, запустите проект, посмотрите работу графического объекта **Genie.ValveContol** и завершите работу проекта.

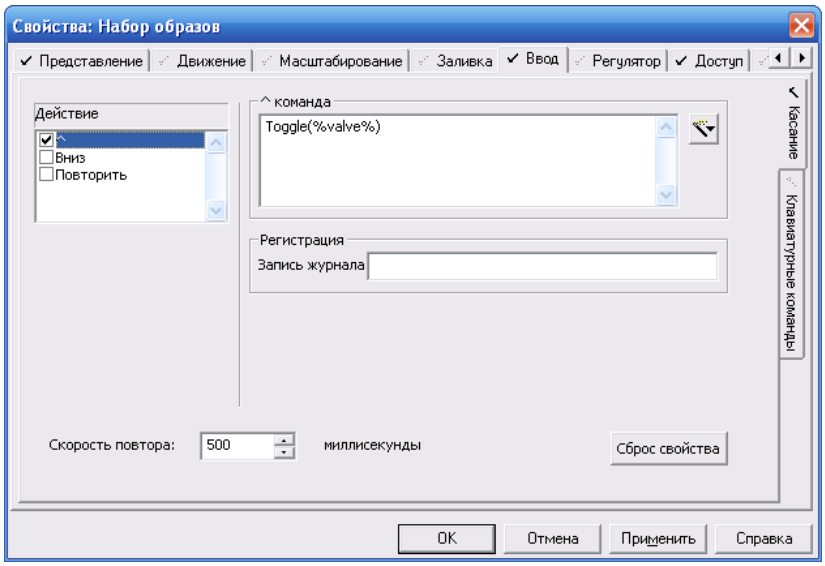


Рис. 9.5. Свойство **Ввод (Касание)** набора символов

В созданный джин можно в любое время внести изменения. Если джин был вставлен на страницы проекта, то перед изменением джина убедитесь, что все страницы проекта сохранены. После внесения изменений сохраните джин и выполните команду **Средства | Обновить страницы**, чтобы обновить все вставленные экземпляры джинов в проекте.

Как следует из выполненного упражнения, в любом месте джина текст или имена тегов переменных могут быть заменены именем подстановки, для которого используется синтаксис **%valve%**. При вставке джина на графическую страницу пользователю предлагается заменить в данном экземпляре джина имя подстановки на имя тега переменной или текст (в рассмотренном упражнении в качестве такового использован тег переменной **Gas_Valve**).

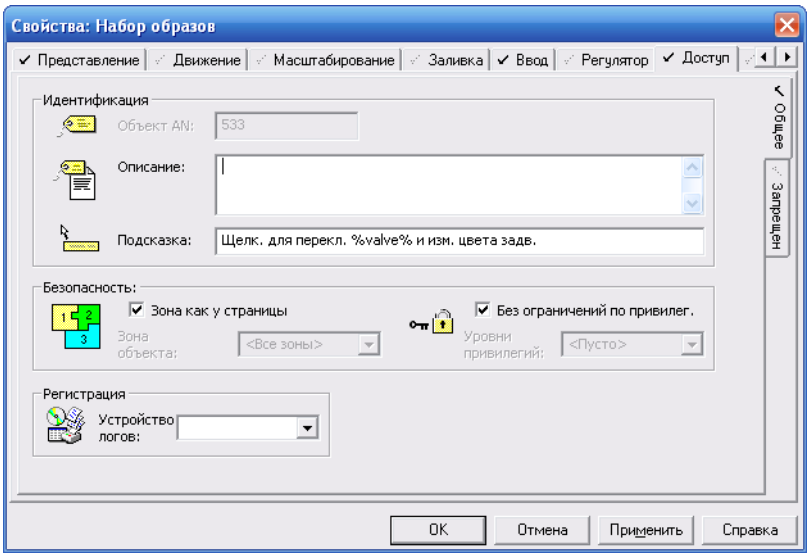


Рис. 9.6. Свойство **Доступ (Общие)** набора символов

Примечание

В рассмотренном упражнении имя подстановки **%valve%** используется 7 раз, но вопрос о подстановке выдается пользователю один раз. При каждой вставке экземпляра джина на графическую страницу все экземпляры **%Valve%** в этой копии будут заменяться указанным именем (в упражнении таким именем являлось имя тега **Gas_Valve**).

Упражнение 9.5. Просмотр свойств джина на графической странице. Проект Training11

Поместите курсор мыши на любой элемент джина, размещенного в рассмотренном ранее упр. 9.5, затем нажмите и удерживайте клавишу **Ctrl** и выполните **Двойной щелчок** левой кнопкой мыши. Для выбранного фрагмента джина откроется окно свойств в режиме только для чтения. Таким образом просмотрите конфигурацию джина и имена тегов переменных.

Совет

Сохраните проект **Training10** на магнитном диске. Для этого в среде приложения **Проводник Citect** достаточно выбрать в окне **Список проектов** этот проект и выполнить команду **Создание резервной копии…** его контекстного меню. В появившемся окне **Создание резервной копии проекта** следует указать требуемые параметры и нажать кнопку **OK**. Проект сохраните под именем **Training11.** В дальнейшем работайте с проектом **Training11**. Для этого достаточно выбрать **Мои проекты** в окне **Список проектов** и выполнить команду **Восстановление…** его

контекстного меню. В появившемся окне **Восстановить проект** следует указать требуемые параметры и нажать кнопку **OK** (новый проект, имя проекта **Training11**).

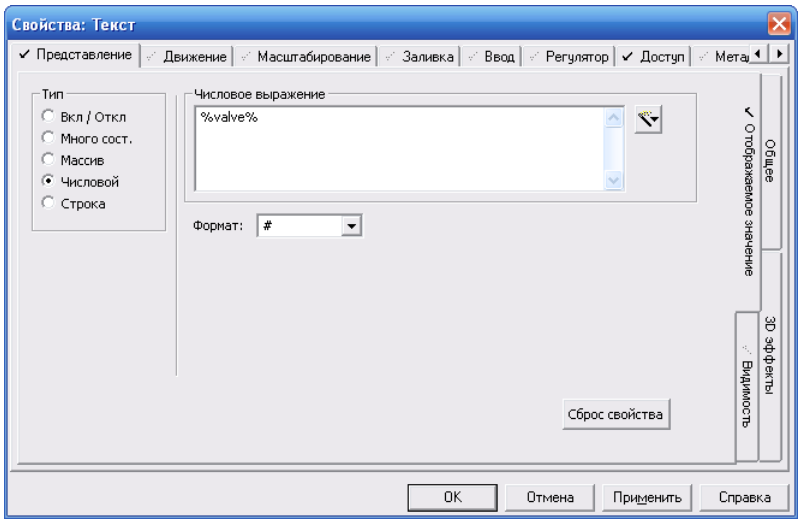


Рис. 9.7. Свойство **Вид (Отображаемое Значение)** графического объекта **Число**

9.3. Всплывающие страницы (окна)

Совет

Материал, относящийся к данной теме можно найти в [2], тема Using Vijeo Citect | Defining and Drawning Graphics Pages; [3], слайды 71 — 72, 141, 144, 145, 147, 149, 150.

Одной из удобных разновидностей графических страниц являются всплывающие страницы. Их отличительной особенностью является то, что они, как поплавки, все время находятся на переднем плане, пока не будут закрыты.

Всплывающие страницы часто используют для отображения всплывающих органов управления процессами или отдельными узлами промышленного оборудования. Одно и та же всплывающая страница может быть использована многократно с разными наборами тегов. В большинстве случаев всплывающая страница используется совместно с суперджином, реже — совместно с джином.

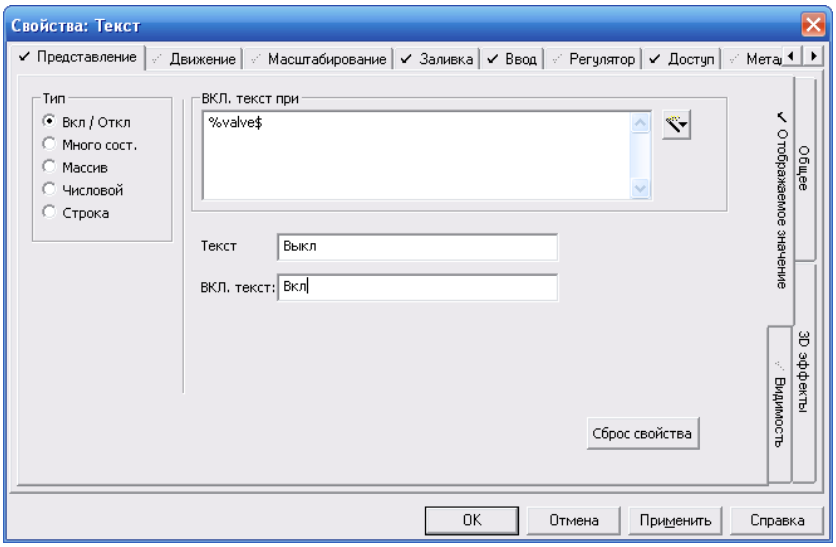


Рис. 9.8. Свойство **Вид (Отображаемое значение)** графического объекта **Текст**

Упражнение 9.6. Создание всплывающей страницы (Popup Window). Проект Training12

В среде приложения **Построитель графики Citect** при выбранном проекте **Training11** нажатием кнопки **Новый** на панели инструментов создайте новую графическую страницу. В появившемся окне **Новый** нажмите кнопку **Страница** и сконфигурируйте страницу в соответствии с рис. 9.12 и нажмите кнопку **OK**. Выполните команду **Свойства страницы…** контекстного меню созданной графической страницы и задайте ее свойства в соответствии с рис. 9.13 и нажмите кнопку **OK**. Разместите в окне графический объект **Genie.TempControl** из библиотеки **training**, в появившемся окне **TempControl** укажите имя тега **Gas_Valve**, нажмите кнопку **OK** и командой **Файл | Сохранить как…** сохраните всплывающую страницу под именем **PopUp** (задайте это имя в поле **Страница** и нажмите кнопку **OK**).

Откройте графическую страницу **Oven**, разместите на ней графический элемент **Кнопка**, задайте его свойства в соответствии с рис. 9.14 и нажмите кнопку **OK**. Сохраните графическую страницу, выполните компиляцию, запустите проект и протестируйте работу кнопки **Управление задвижкой** страницы **Oven** (в результате должно появиться созданное в проекте всплывающее окно). Завершите работу проекта. Сохраните проект под новым именем **Training12**, затем восстановите проект под этим именем и, в дальнейшем, работайте с проектом **Training12**.

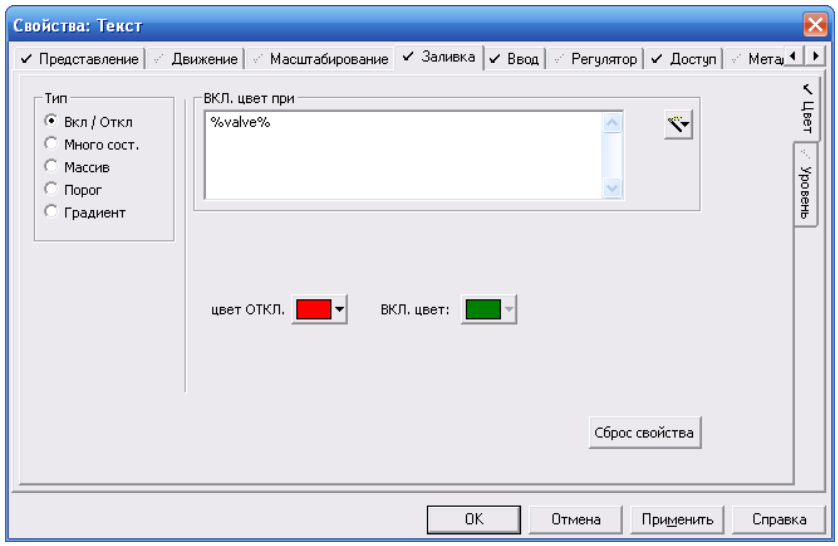


Рис. 9.9. Свойство **Заливка (Цвет)** графического объекта **Текст**

Глава 10. Устройства. Конфигурирование устройства для регистрации команд оператора

Совет
Материал, относящийся к данной теме можно найти в [2], тема Using Vijeo Citect | Using Devices; [3], слайды 171 — 179.

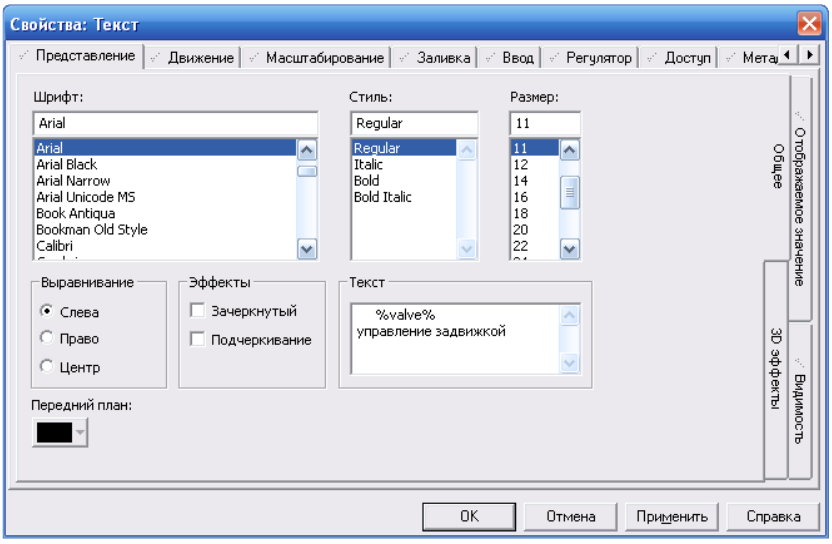


Рис. 9.10. Свойство **Вид (Общие)** графического объекта **Текст**

Рис. 9.11. Группировка графических объектов

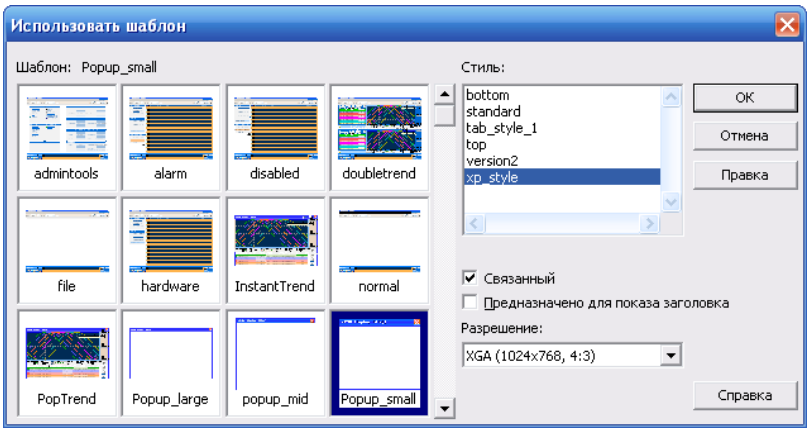


Рис. 9.12. Свойства всплывающей страницы

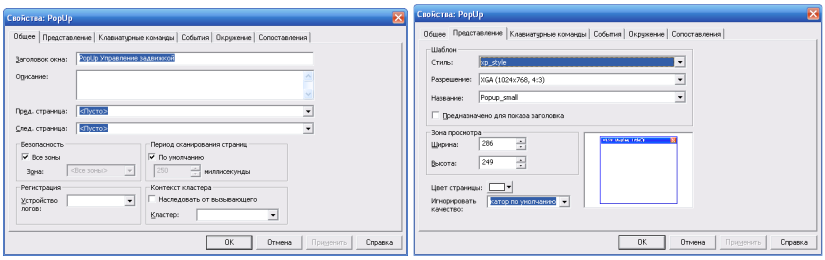


Рис. 9.13. Свойства всплывающей страницы

Устройство представляет собой средство для передачи данных высокого уровня (отчетов, журналов регистрации команд и сигналов тревог) между системой Vijeo Citect и такими элементами, как принтер, базы данных, файлы на магнитном диске различных форматов. С помощью устройств данные можно не только записывать, но и читать из файлов или баз данных.

Каждое устройство в системе Vijeo Citect имеет отдельную запись, которая определяет формат передаваемых или принимаемых данных, тип и имя устройства. Для определения устройства в среде приложения **Проводник Citect** в поле **Список проектов** выберите проект, раскройте его, выберите папку **Система** и выполните двойной щелчок левой кнопкой мыши на значке **Устройства** в поле **Содержимое Система**. Другим способом определения устройства является переход в среду приложения **Редактор проектов Citect** и выполнение команды **Система | Устройства**. Прежде, чем перейти к выполнению упражнения, рассмотрите материал, представленный в [3] на слайдах 171 — 179.

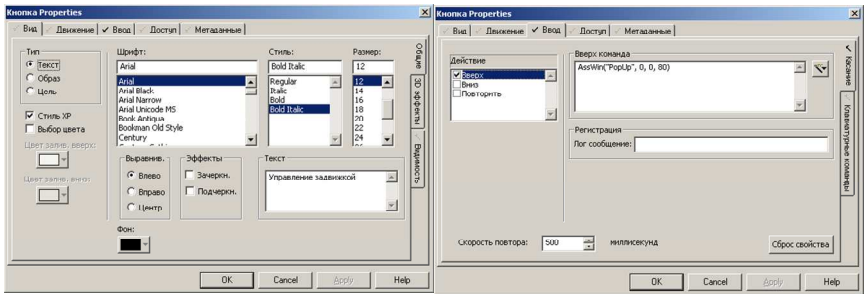


Рис. 9.14. Свойства графического элемента **Кнопка** на странице **Oven** ("PopUp" — имя всплывающего окна; 0, 0 — горизонтальная и вертикальная координаты левого верхнего угла всплывающего окна; 80 = 16 (нет заголовка) + 64 (всплывающее окно); "Oven_Temp" — имя тега для графического объекта Genie)

Упражнение 10.1. Конфигурирование и использование устройства для регистрации команд оператора. Проект Training13

Перейдите в среду **Редактор проектов Citect** демонстрационного проекта **Training12**, выполните команду **Система | Устройства**, добавьте, сконфигурируйте устройство для регистрации команд оператора в соответствии с рис. 10.1 и нажмите кнопку **Добав**. В поле **Формат** 15, 10 и 64 соответственно задают количество позиций на отображение даты, времени и текста сообщения о выполненной оператором команде. Синтаксис допустимых форматов можно посмотреть в [2], тема Using Vijeo Citect | Using Devices | Using Command Fields или по кнопке **Справка** окна, представленного ранее на рис. 10.1. В поле **Формат** можно также использовать табуляторы и произвольный неформатируемый текст. В поле **Имя файла** переменная **[DATA]:** задает путь к файлу для регистрации команд оператора (см. секцию **[CTEDIT]** файла **citect.ini**, для его просмотра используйте приложение, запускаемое командой **Средства | Редактор конфигурирования компьютера**). Значение **-1** в поле **Число файлов** означает, что запись будет проводиться в один и тот же файл в режиме добавления в конец файла.

Перейдите в среду приложения **Построитель графики Citect**, откройте графическую страницу **Oven**, модифицируйте свойства графического объекта **Набор образов** (задвижка на трубопроводе) в соответствии с рис. 10.2 и нажмите кнопку **OK**. В графической странице **Oven** модифицируйте свойства графического объекта **Текст** (Burner ON/OFF) в соответствии с рис. 10.3 и нажмите кнопку **OK**. В графической странице **Oven** модифицируйте свойства графического объекта **Кнопка** с надписью **Управление задвижкой** в соответствии с рис. 10.4 и нажмите кнопку **OK**. Сохраните графическую страницу **Oven**, выполните компиляцию и запустите проект. Перейдите на страницу **Oven**, выполните изменение состояния клапана, горелки и с помощью кнопки создайте всплывающее окно. Завершите работу с проектом. Посмотрите содержимое файла ..\ProgramData\Schneider Electric\Vijeo

Citect 7.30\Data\Com_Log_11_08_2013.txt (дата в имени файла будет соответсвовать дате выполения проекта).

Архивируйте проект под именем **Training13**, восстановите проект **Training13** и, в дальнейшем, работайте с ним.

Совет
Изучите также демонстрационный проект ComLogSuper.

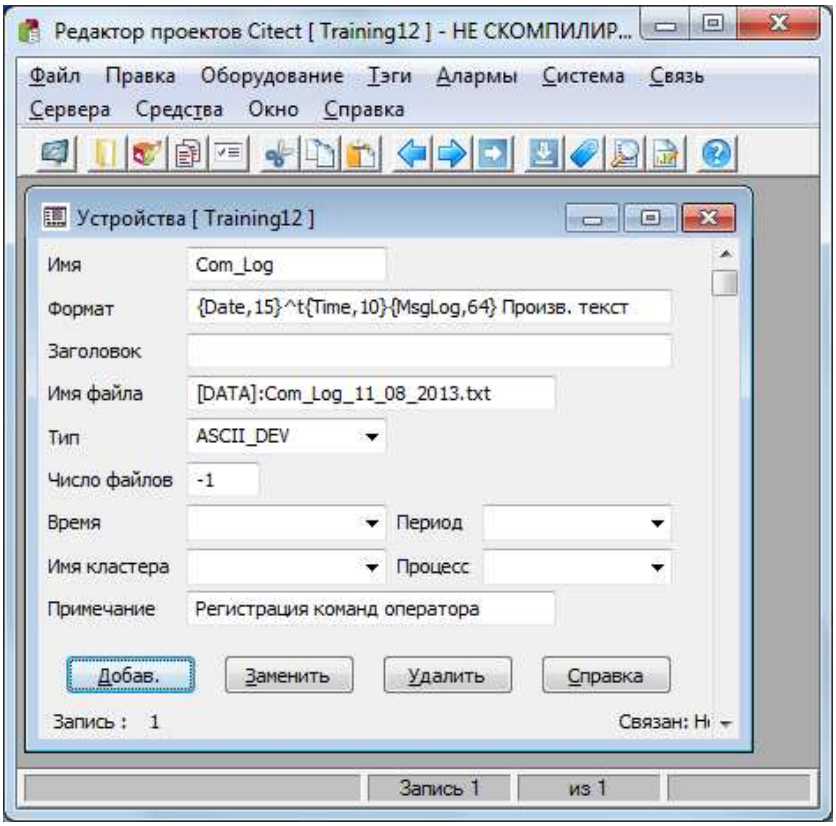


Рис. 10.1. Конфигурирование устройства для регистрации команд оператора

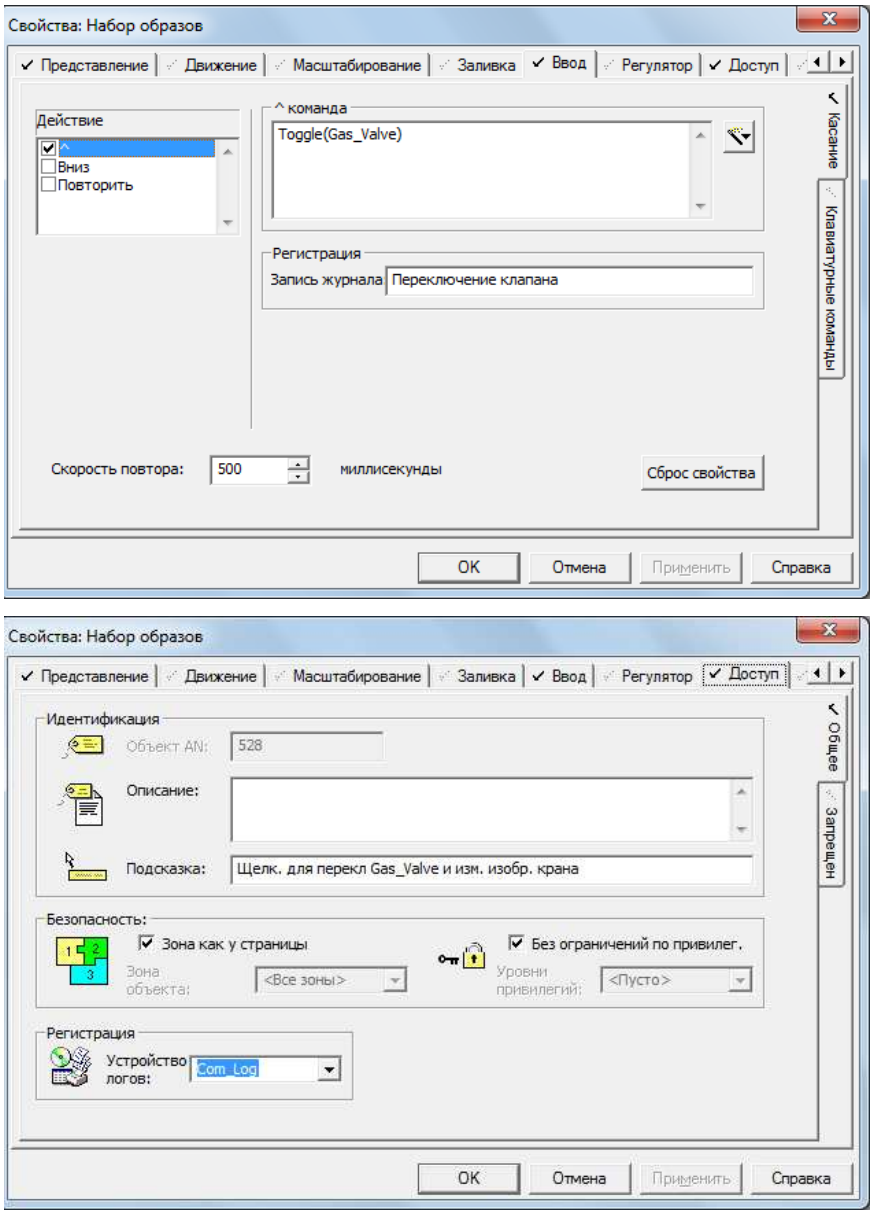


Рис. 10.2. Модификация свойств графического объекта **Набор образов** страницы **Oven**

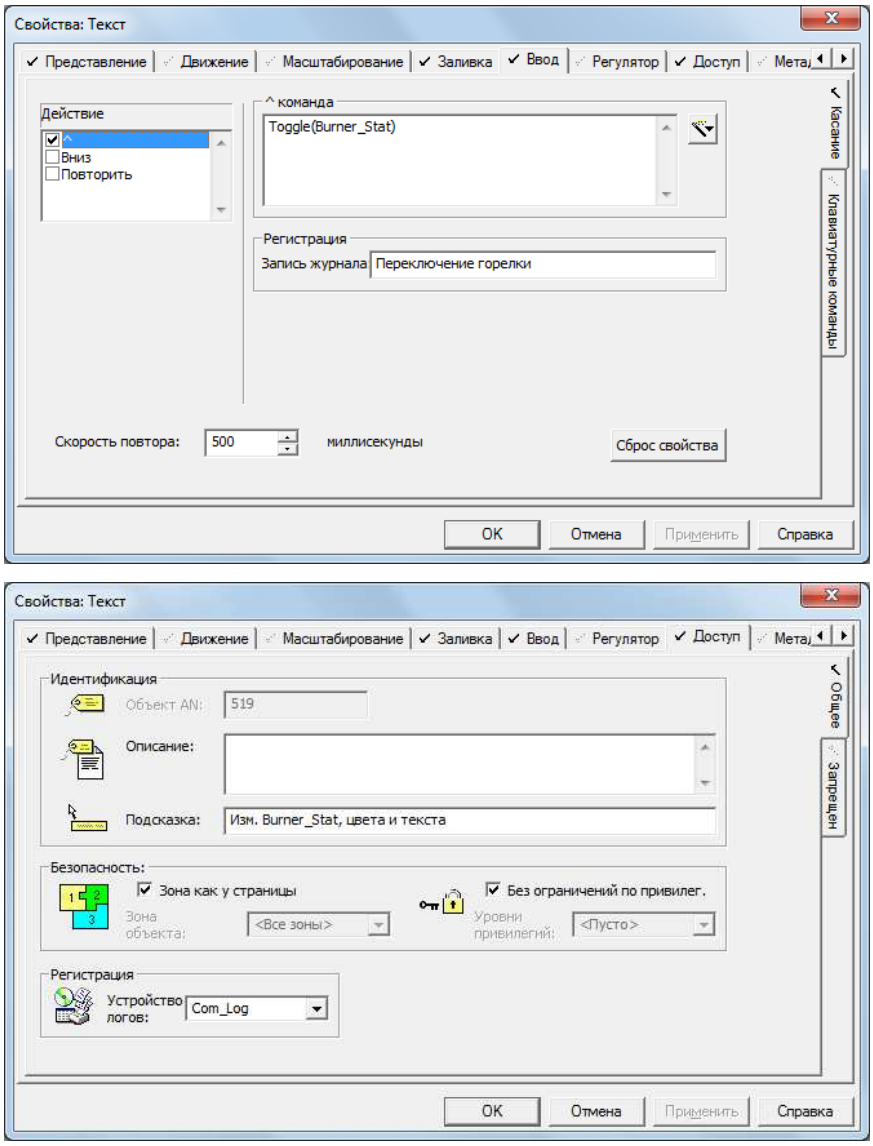


Рис. 10.3. Модификация свойств графического объекта **Текст** страницы **Oven**

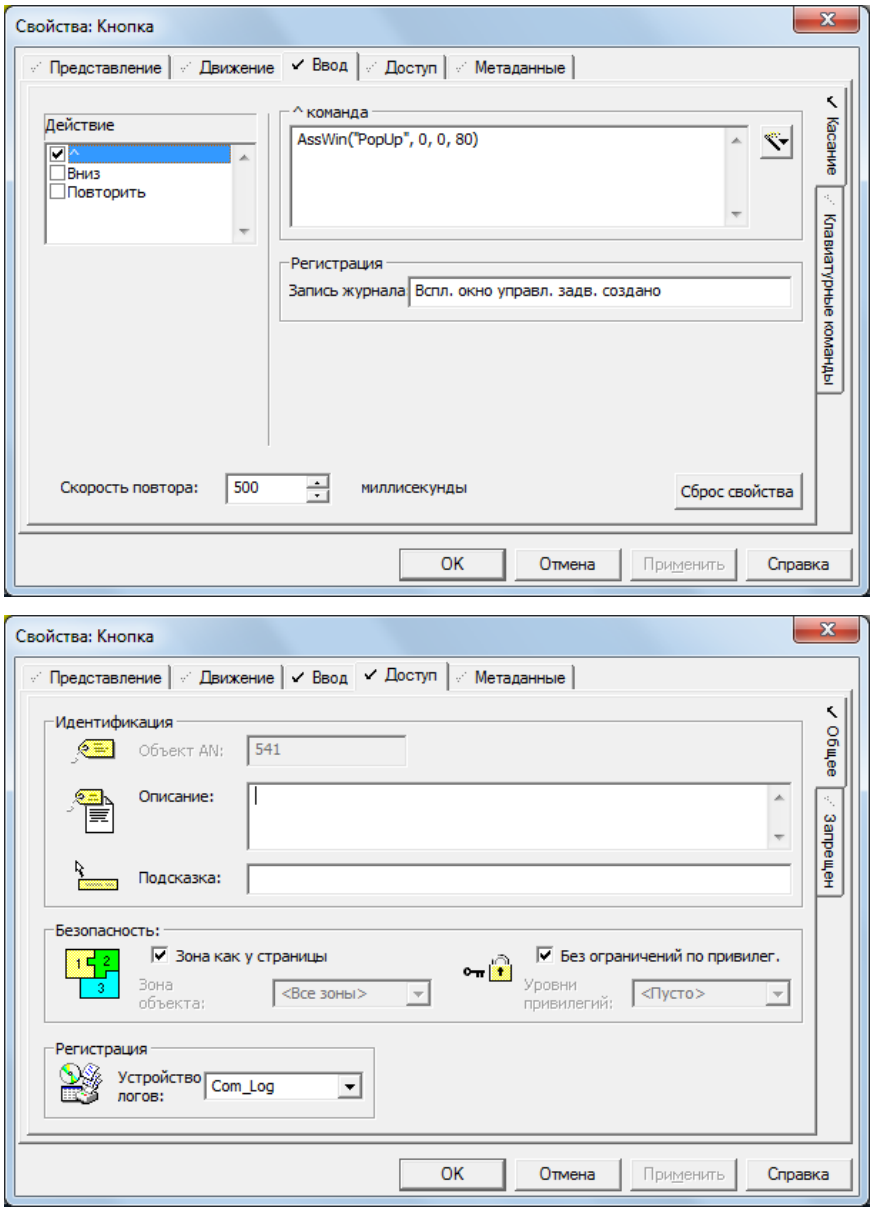


Рис. 10.4. Модификация свойств графического объекта **Кнопка** страницы **Oven**

Глава 11. События. Определение и обработка событий

> ***Совет***
> Материал, относящийся к данной теме можно найти в [2], тема Using Vijeo Citect | Configuring Events; [3], слайды 180 — 189.

События могут запускать некоторое действие, например, команду или набор команд. По событию можно извещать оператора, когда некоторый процесс завершился или выполнять некоторую последовательность команд, когда, например, процесс достиг некоторой точки. Выполнять команды при наступлении события можно следующими способами — автоматически в указанное время или период времени, автоматически при возникновении некоторого события или автоматически при возникновении события в заданное время или промежуток времени. События определяются в проекте и сохраняются в базе данных. Для исполнения команд, связанных с событием, оно должно быть активировано (разрешено) с помощью **Мастера конфигурирования компьютера** (полная установка, окно **Настройка событий**). Можно обрабатывать события на любом компьютере системы Vijeo Citect, но только каждое событие на одном компьютере.

События ***не обязательно*** должны иметь уникальные имена. Так, если система Vijeo Citect работает на нескольких компьютерах в сети и требуется запустить событие на всех компьютерах, то нужно использовать **Global** в качестве имени события.

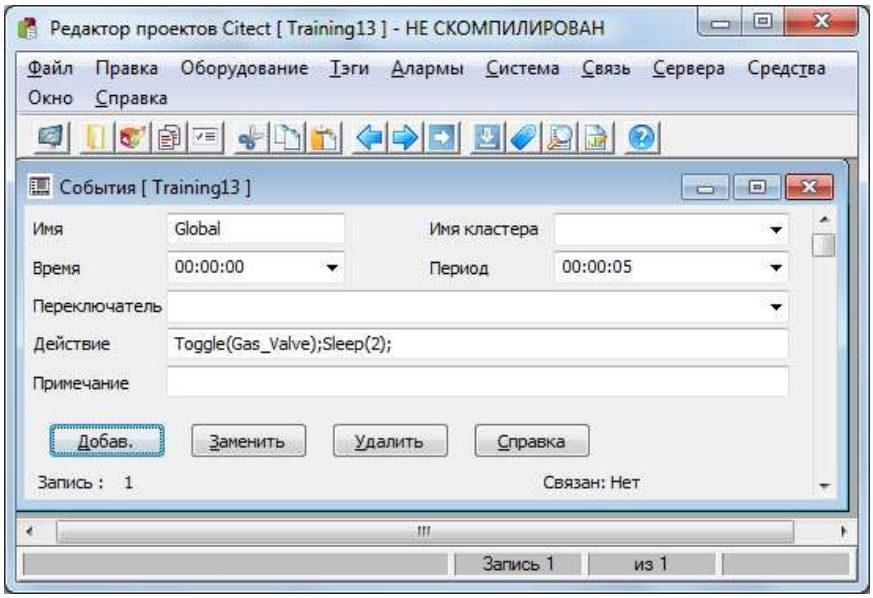
Для определения события в среде приложения **Проводник Citect** в поле **Список проектов** выберите проект, раскройте его, выберите папку **Система** и выполните двойной щелчок левой кнопкой мыши на значке **События** в поле **Содержимое Система**. Другим способом определения события является переход в среду приложения **Редактор проектов Citect** и выполнение команды **Система | События**.

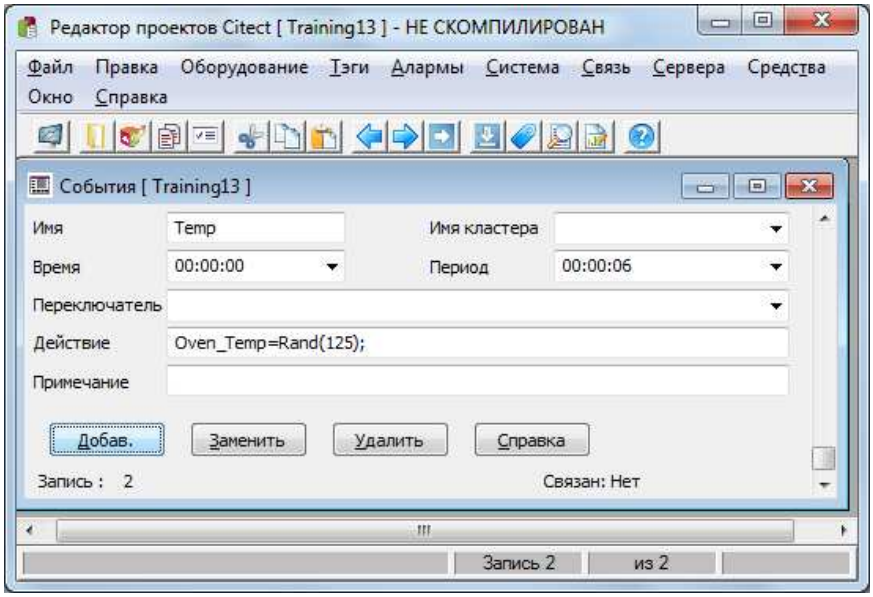
Для активизации (разрешения) события следует любым способом запустить **Мастер конфигурирования компьютера** (полная установка) и в окне **Настройка событий** включить обработку всех событий.

Упражнение 11.1. Определение и разрешение событий. Проект Training14

Перейдите в среду приложения **Редактор проектов Citect** демонстрационного проекта **Training13**, выполните команду **Система | События** и добавьте события в соответствии с рис. 11.1, всякий раз нажимая после добавления очередного события кнопку **Добав**.

Выполните компиляцию.


Редактор проектов Citect [Training13] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства
Окно Справка
События [Training13]
Имя
Global
Имя кластера
Время
00:00:00
Период
00:00:05
Переключатель
Действие
Toggle(Gas_Valve);Sleep(2);
Примечание
Добав.
Заменить
Удалить
Справка
Запись : 1
Связан: Нет
Запись 1
из 1


Редактор проектов Citect [Training13] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства
Окно Справка
События [Training13]
Имя
Temp
Имя кластера
Время
00:00:00
Период
00:00:06
Переключатель
Действие
Oven_Temp=Rand(125);
Примечание
Добав.
Заменить
Удалить
Справка
Запись : 2
Связан: Нет
Запись 2
из 2

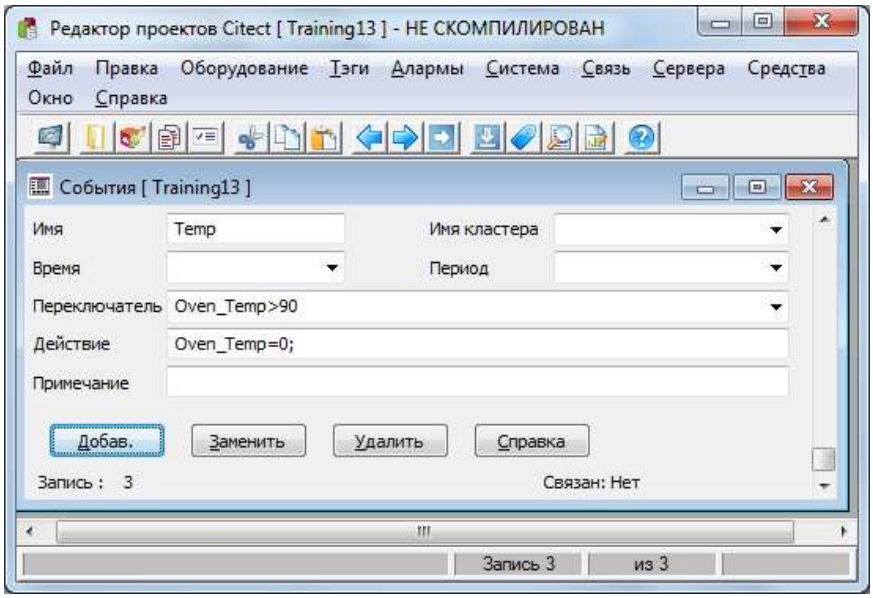


Рис. 11.1. Добавление и конфигурирование событий

Замечание
Используйте систему помощи для получения сведений о стандартных функциях Cicode.

Запустите **Средства | Мастер настройки компьютера** в режиме **Индивидуальная настройка** и настройте окно **Настройка событий** в соответствии с рис. 11.2.

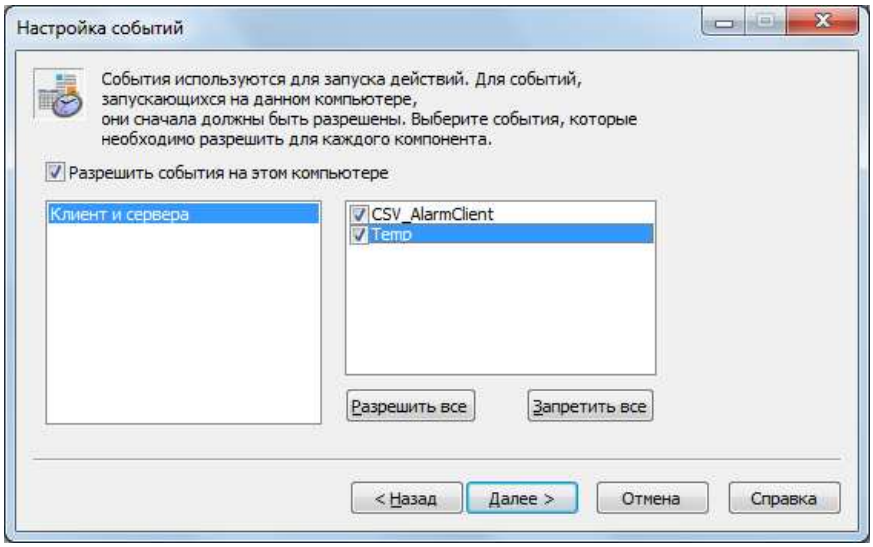


Рис. 11.2. Разрешение событий

Откомпилируйте и запустите проект. Перейдите в графическое окно **Oven** и пронаблюдайте за изменением состояния клапана и значения температуры. Завершите работу проекта. Архивируйте проект под именем **Training14**, восстановите проект **Training14** и, в дальнейшем, работайте с ним.

Глава 12. Сигналы тревог: категории, группы и звуковое оформление

Примечание

Материал, относящийся к данной теме можно найти в [2], тема Using VijeoCitect | Using Alarms in SCADA; [3], слайды 196 — 202, 204 — 214.

Сигналы тревог можно включать в определенные категории. Для каждой категории можно задать формат отображения (шрифт и тип страницы), детали регистрации (принтер или файл), некоторое действие при срабатывании сигнала тревоги, отнесенного к данной категории (например, звуковой сигнал), приоритет, порядок отображения и др.

Можно сконфигурировать до 16376 категорий (категория 0 по умолчанию; категория 254 зарезервирована для итоговой информации по сигналам тревоги, создаваемой пользователем; категория 255 зарезервирована для аппаратных сигналов тревог). Если определить сигнал тревоги и задать для него категорию, то тем самым будет задан характер поведения при обработке сигнала тревоги. Чтобы определить, куда будет направлена информация о сигнале тревоги, задайте устройство. Можно регистрировать сигналы тревоги более, чем одним устройством, и выбирать для регистрации группы сигналов тревог. Каждая категория имеет связанный с ней приоритет, который используется для определения порядка отображения сигналов тревог, предоставляя удобные для оператора средства фильтрации.

Совет

Разумно включать сигналы тревог разных типов в различные категории, чтобы назначить каждому типу собственный формат отображения и способ их обработки.

Для определения категории в среде приложения **Проводник Citect** в поле **Список проектов** выберите проект, раскройте его, выберите компоненту **Алармы** и выполните двойной щелчок левой кнопкой мыши на значке **Категории алармов** в поле **Содержимое Алармы**. Другим способом определения категории является переход в среду приложения **Редактор проектов Citect** и выполнение команды **Алармы | Категории алармов**.

Упражнение 12.1. Категории сигналов тревог. Проект Training15

Перейдите в среду **Редактора проектов Citect** демонстрационного проекта **Training14**. Выполните команду **Алармы | Дискретные алармы** и сконфигурируйте категорию 1 сигналов тревог (на приведенном ранее рис. 5.4 в поле **Категория** укажите 1). Всякий раз после внесения изменения нажимайте кнопку **Заменить**. Далее выполните команду **Алармы | Аналоговые алармы** и сконфигурируйте категорию 2 сигналов тревог (на приведенном ранее рис. 5.3 в поле **Категория**

укажите 2). После внесения изменения нажмите кнопку **Заменить**. Выполните команду **Система | Устройства** и добавьте еще одно устройство — для архивного файла журнала тревог (рис. 12.1), нажмите после заполнения формы кнопку **Добавить.** Более подробные сведения о форматах, используемых в поле **Формат** приведены в [2], темы Using Vijeo Citect | Using Devices | Using Command Fields и Technical Reference | Vijeo Citect Reference Information | Format fields. Здесь **{Name,16}** — имя сигнала тревоги и ширина поля вывода, **{Desc,32}** — описание сигнала тревоги и ширина поля вывода, **{OnTime,11}** — время активации сигнала тревоги и ширина поля вывода, **{LogState,11}** — текущее состояние сигнала тревоги и ширина поля вывода. Значение **-1** в поле **Число файлов** означает запись в один файл. Для создания категорий тревог выполните команду **Алармы | Категории алармов** и добавьте две категории тревог в соответствии с рис. 12.2, нажимая каждый раз кнопку **Добавить.**

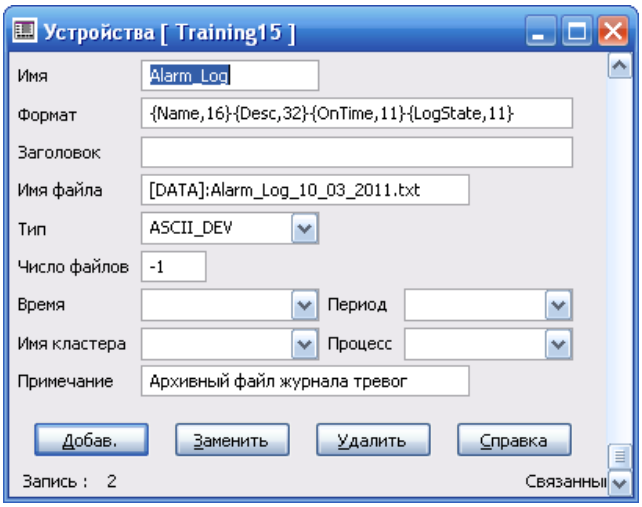
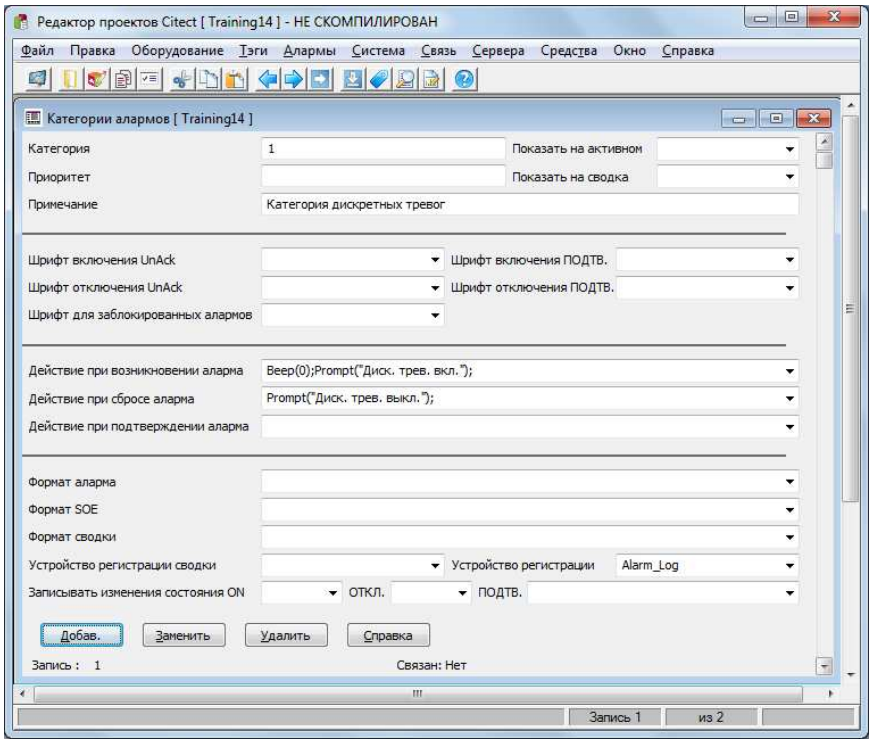


Рис. 12.1. Конфигурирование устройства для создания архивного файла журнала тревог

Замечание

Более подробные сведения о содержимом полей можно получить с помощью кнопки **Справка**, расположенной на соответствующей форме.

Выполните компиляцию, запустите проект и зарегистрируйтесь как привилегированный пользователь. Откройте страницу **Oven** и убедитесь, что с помощью обработки событий создаются аналоговые и дискретные тревоги (должны слышаться акустические сигналы, в поле **Prompt**, расположенном в левом нижнем углу страницы **Oven**, должны появляться сообщения о дискретных и аналоговых тревогах).


Редактор проектов Citect [Training14] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Категории алармов [Training14]
Категория
1
Показать на активном
Приоритет
Показать на сводка
Примечание
Категория дискретных тревог
Шрифт включения UnAck
Шрифт включения ПОДТВ.
Шрифт отключения UnAck
Шрифт отключения ПОДТВ.
Шрифт для заблокированных алармов
Действие при возникновении аларма
Beep(0);Prompt("Диск. трев. вкл.");
Действие при сбросе аларма
Prompt("Диск. трев. выкл.");
Действие при подтверждении аларма
Формат аларма
Формат SOE
Формат сводки
Устройство регистрации сводки
Устройство регистрации
Alarm_Log
Записывать изменения состояния ON
ОТКЛ.
ПОДТВ.
Добав.
Заменить
Удалить
Справка
Запись : 1
Связан: Нет
Запись 1
из 2

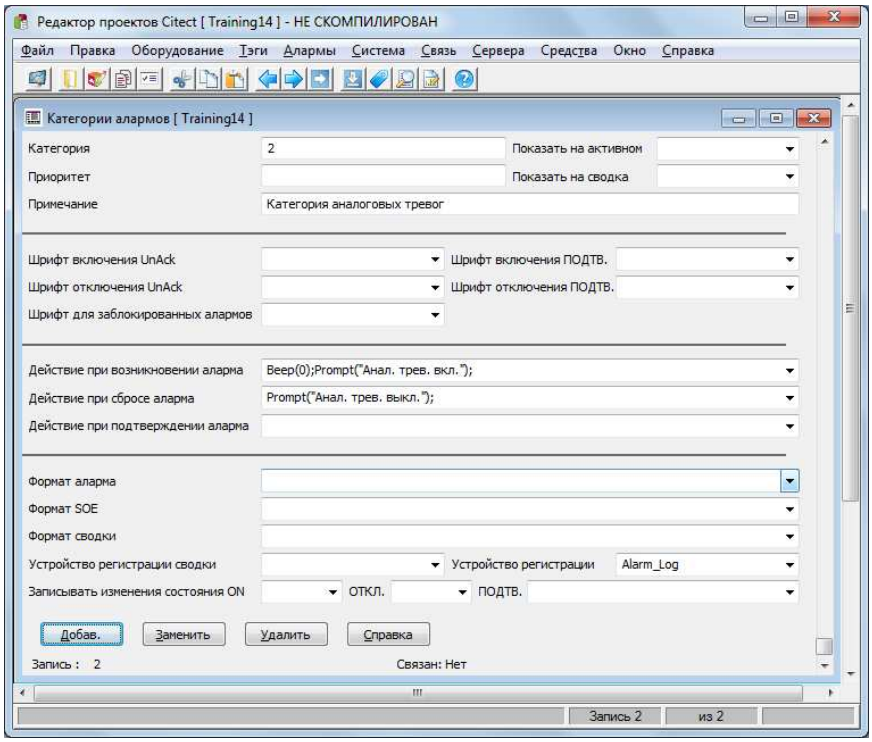


Рис. 12.2. Свойства категорий сигналов тревог

С помощью команды **Alarms | Active Alarms** перейдите на страницу активных тревог и подробно изучите действие кнопок на расположенных в левой части окна (рис. 12.3).

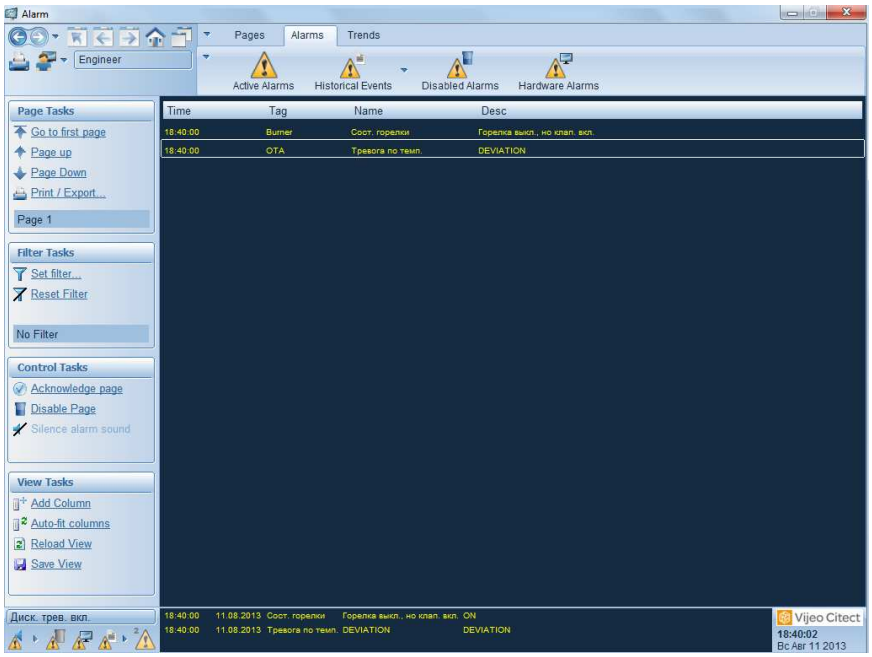


Рис. 12.3. Вид окна отображения активных тревог

Завершите работу проекта и посмотрите журнал тревог в файле ..\Vijeo Citect 7.30\ Data\ Alarm_Log_11_08_2013.txt. Архивируйте проект под именем **Training15**, восстановите проект **Training15** и, в дальнейшем, работайте с ним.

Упражнение 12.2. Звуковое оформление сигналов тревог. Проект Training16

Перейдите в среду приложения **Редактор проектов Citect** демонстрационного проекта **Training15**, выполните команду **Алармы | Категории алармов** и назначьте приоритеты всем категориям тревог (для дискретных тревог — приоритет 2, для аналоговых тревог — приоритет 3, приоритет задается в поле **Приоритет**, см. приведенный ранее рис. 12.2). В корневую папку диска **D:** поместите файлы **DigAlarm.wav** и **AnaAlarm.wav**. Запустите приложение **Редактор настройки Компьютера**. К разделу **[ALARM]** добавьте два параметра — **Sound2** и **Sound3** (рис. 12.4). Выполните компиляцию, запустите приложение **Мастер конфигурирования компьютера** в режиме **Полная установка** и убедитесь, что в окне **Настройка событий** разрешено событие **CSV_AlarmClient**. Запустите проект, прослушайте звуки, завершите работу проекта.

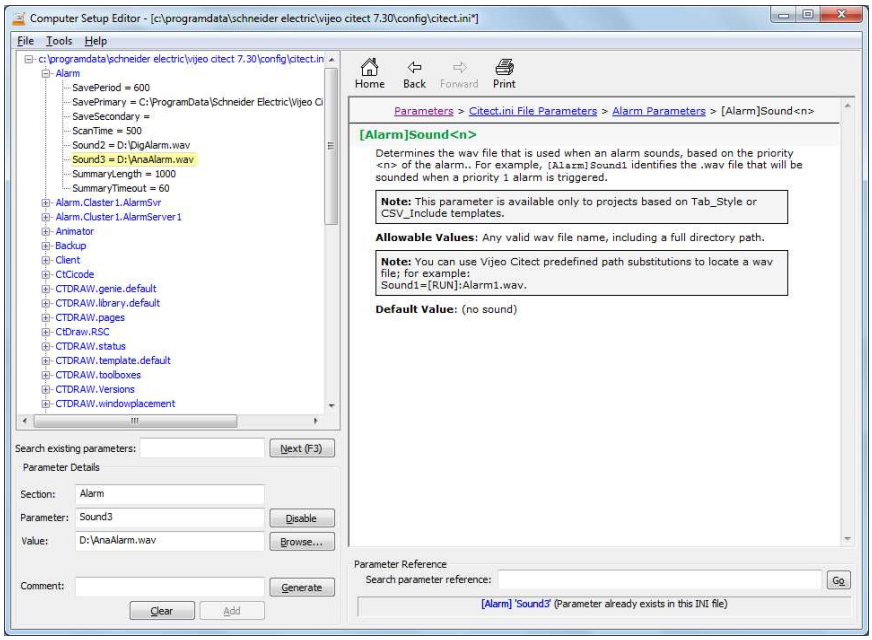


Рис. 12.4. Модификация раздела **[ALARM]** файла **citect.ini**

Сохраните проект **Training15** под именем **Training16**, восстановите проект **Training16** и, в дальнейшем, работайте с ним.

Глава 13. Система навигации

Замечание

Материал, относящийся к данной теме можно найти в [2], тема Using the CSV_Include Project | Creating a New Project | Creating Custom Menus | Building custom menus; [3], слайды 267 — 282.

Главным средством навигации в проекте системы Vijeo Citect является строка меню, находящаяся непосредственно под строкой заголовка графической страницы. С помощью команд меню можно перемещать пользователя на заданную страницу или вызывать функцию, написанную на языке Cicode.

Конфигурирование меню в Vijeo Citect 7.30 можно выполнить, используя более удобные новые средства (команда **Система | Конфигурация меню** в **Редакторе проектов Citect**) или с помощью устаревших средств — панели конфигурирования в странице **CSV_AdminTools** (поддерживается включаемым проектом **CSV_Include**). В первом случае после модификации меню его новая конфигурпция сохраняется в файле **PageMenu.dbf** в папке проекта. В последнем случае после модификации меню его новая конфигурация сохраняется в файле **Menu.dbf** в папке проекта. Эти файлы можно портировать в другие проекты путем копирования в соответствующие папки проектов. Рассмотрим последовательно обе эти возможности.

13.1. Конфигурирование меню средствами Vijeo Citect 7.30

По умолчанию, если конфигурирование меню командой **Система | Конфигурация меню** в **Редакторе проектов Citect** не выполняется, в графических страницах, основанных на большинстве шаблонов стиля **tab_style_1**, наследуется строка меню, представленная ранее на рис. 8.1 и 12.3. Объясняется это тем, что большинство шаблонов стиля **tab_style_1**, в свою очередь, основывается на одном и том же общем шаблоне **_base**.

Команду конфигурирования системы меню можно активизировать двумя способами. Во-первых, в среде **Проводника Citect** в окне **Список проектов** можно раскрыть дерево проекта, выбрать компонент **Система** и кликнуть по элементу **Конфигурация меню** в окне **Содержимое Система**. Во-вторых, для этой цели можно выполнить команду **Система | Конфигурация меню** в **Редакторе проектов Citect**.

Упражнение 13.1. Создание и русификация общих меню графических страниц

Сконфигурируйте общие меню графических страниц, представленные на рис. 13.1. С этой целью используйте проект **Training16** и сконфигурируйте меню в соответствии с рис. 13.2 и 13.3. Выполните компиляцию, запустите проект и протестируйте работу меню на нескольких графических страницах, основанных на шаблоне **Normal** стиля **tab_style_1**.

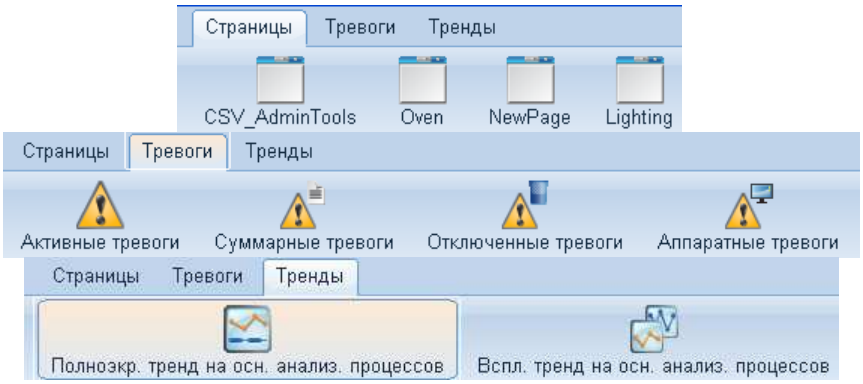
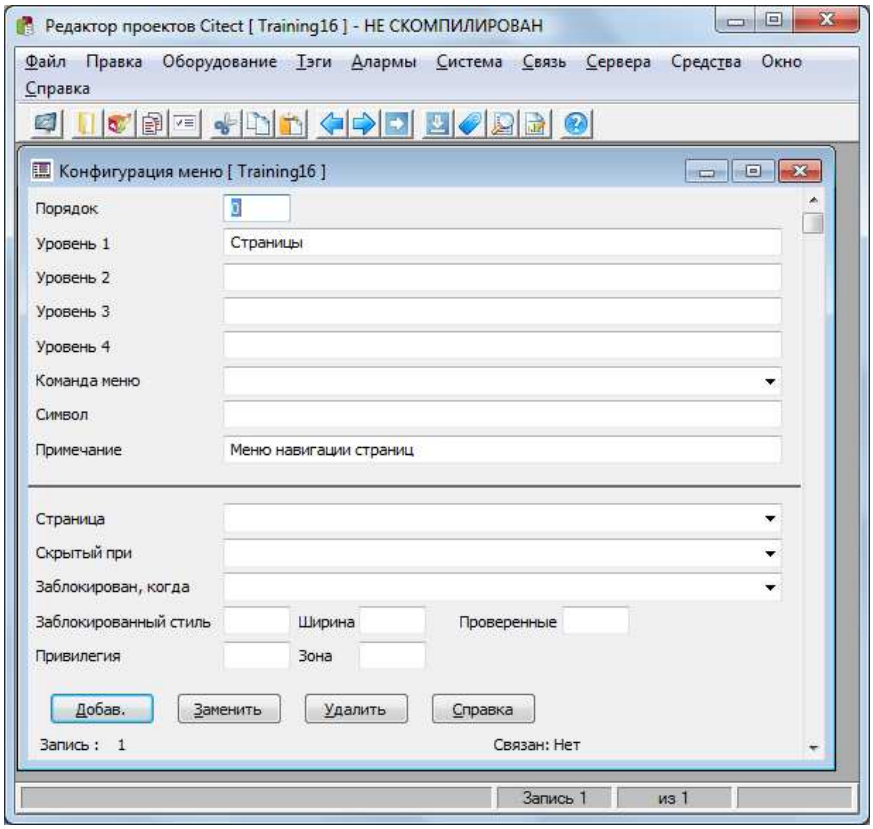
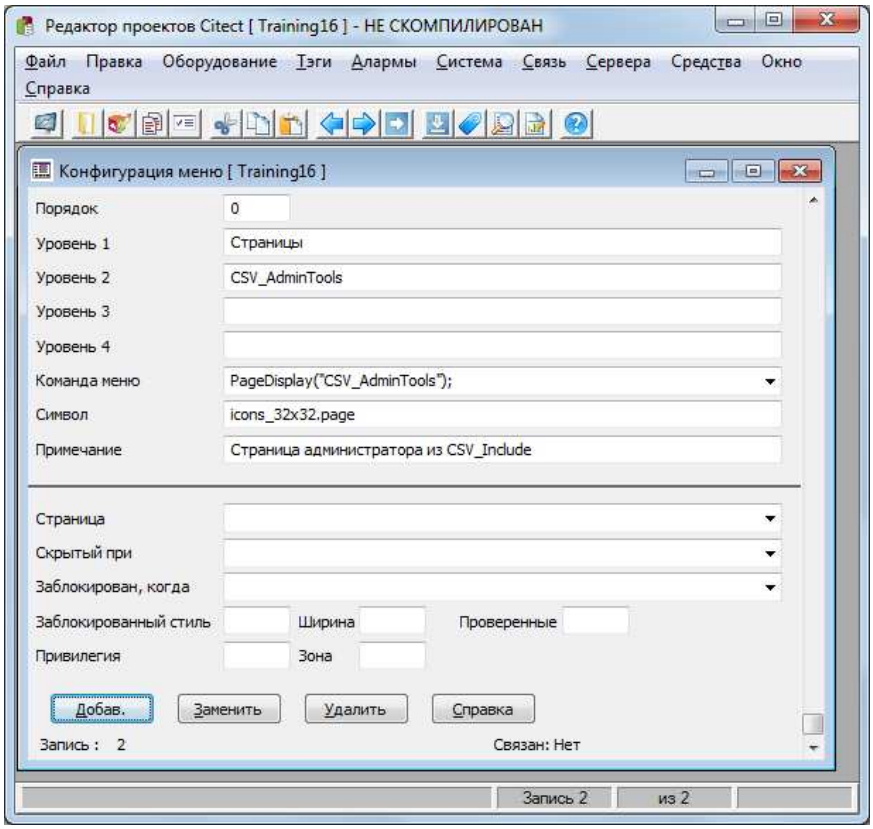
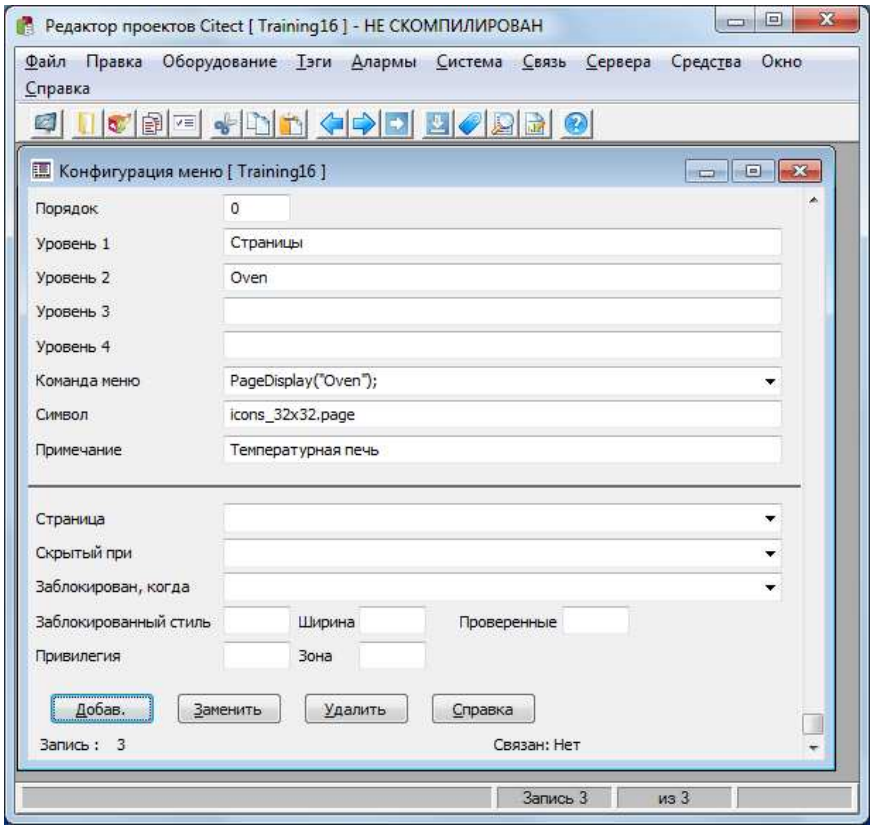
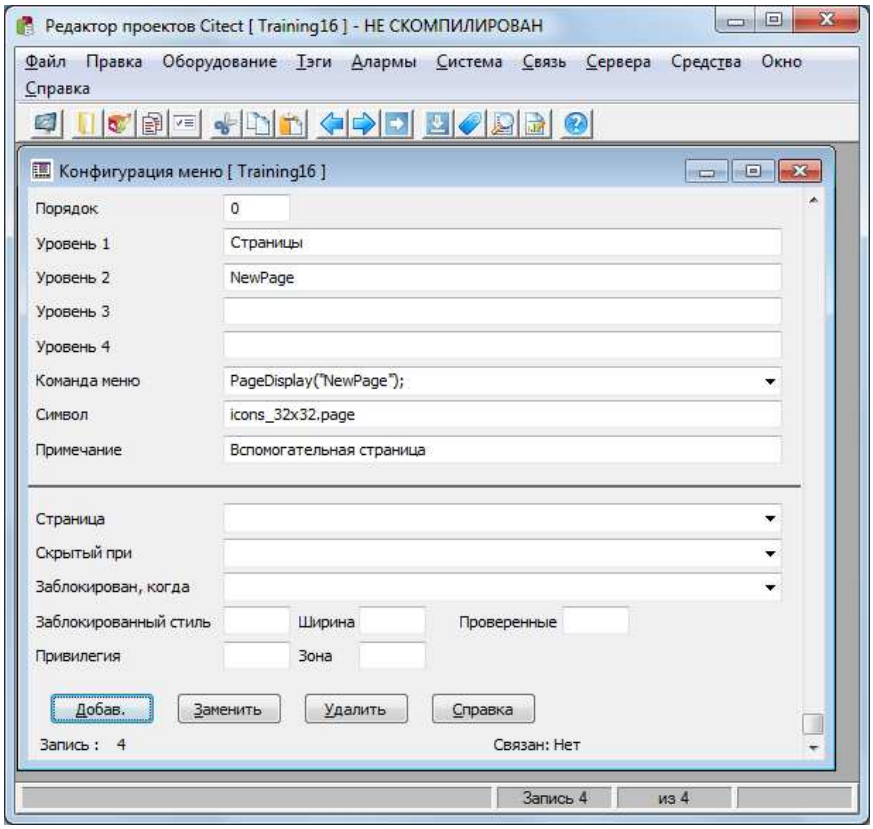


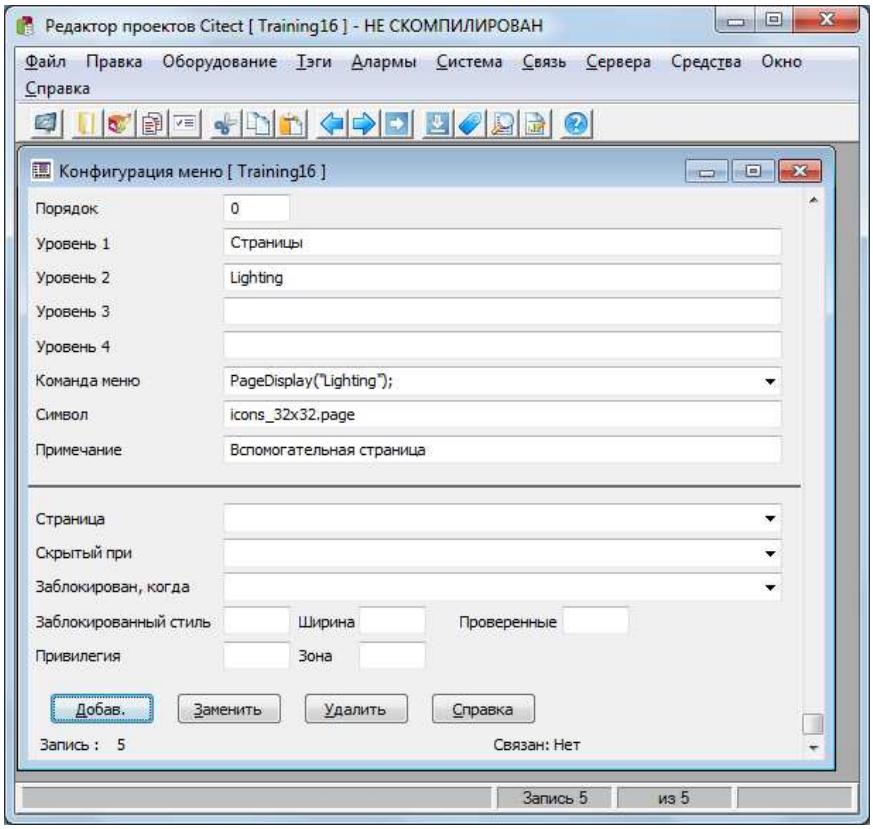
Рис. 13.1. Вид меню и его команд


Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок
Уровень 1
Страницы
Уровень 2
Уровень 3
Уровень 4
Команда меню
Символ
Примечание
Меню навигации страниц
Страница
Скрытый при
Заблокирован, когда
Заблокированный стиль
Ширина
Проверенные
Привилегия
Зона
Добав.
Заменить
Удалить
Справка
Запись : 1
Связан: Нет
Запись 1
из 1


Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок 0
Уровень 1 Страницы
Уровень 2 CSV_AdminTools
Уровень 3
Уровень 4
Команда меню PageDisplay("CSV_AdminTools");
Символ icons_32x32.page
Примечание Страница администратора из CSV_Include
Страница
Скрытый при
Заблокирован, когда
Заблокированный стиль Ширина Проверенные
Привилегия Зона
Добав. Заменить Удалить Справка
Запись : 2 Связан: Нет
Запись 2 из 2


Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок
0
Уровень 1
Страницы
Уровень 2
Oven
Уровень 3
Уровень 4
Команда меню
PageDisplay("Oven");
Символ
icons_32x32.page
Примечание
Температурная печь
Страница
Скрытый при
Заблокирован, когда
Заблокированный стиль
Ширина
Проверенные
Привилегия
Зона
Добав.
Заменить
Удалить
Справка
Запись : 3
Связан: Нет
Запись 3
из 3


Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок 0
Уровень 1 Страницы
Уровень 2 NewPage
Уровень 3
Уровень 4
Команда меню PageDisplay("NewPage");
Символ icons_32x32.page
Примечание Вспомогательная страница
Страница
Скрытый при
Заблокирован, когда
Заблокированный стиль
Ширина
Проверенные
Привилегия
Зона
Добав.
Заменить
Удалить
Справка
Запись : 4
Связан: Нет
Запись 4
из 4


Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок
0
Уровень 1
Страницы
Уровень 2
Lighting
Уровень 3
Уровень 4
Команда меню
PageDisplay("Lighting");
Символ
icons_32x32.page
Примечание
Вспомогательная страница
Страница
Скрытый при
Заблокирован, когда
Заблокированный стиль
Ширина
Проверенные
Привилегия
Зона
Добав.
Заменить
Удалить
Справка
Запись : 5
Связан: Нет
Запись 5
из 5

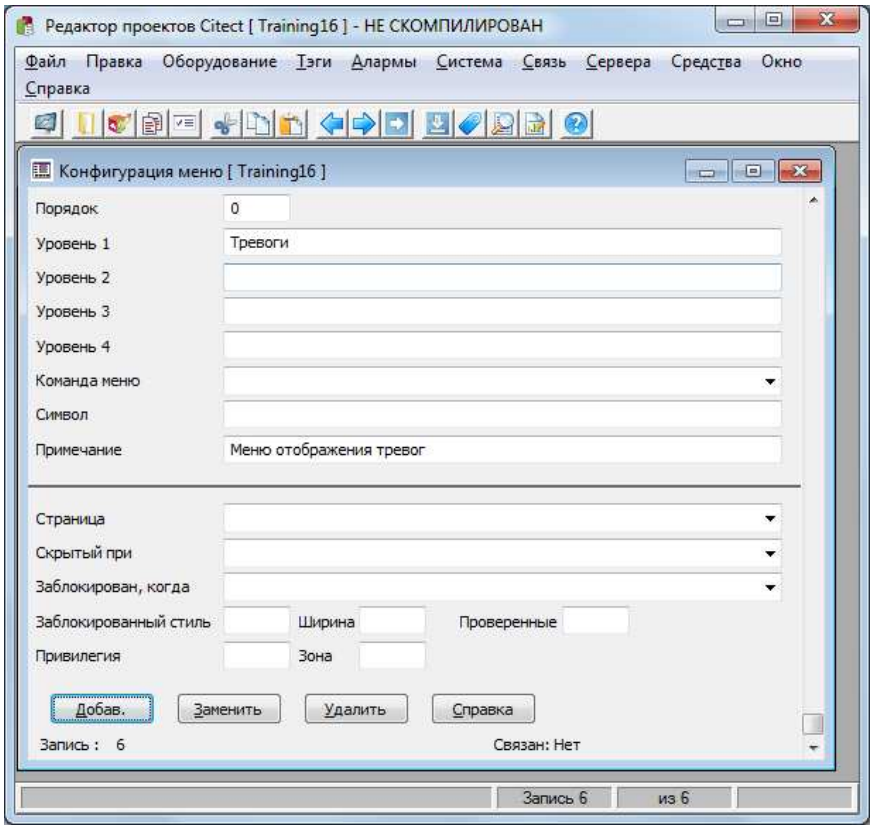
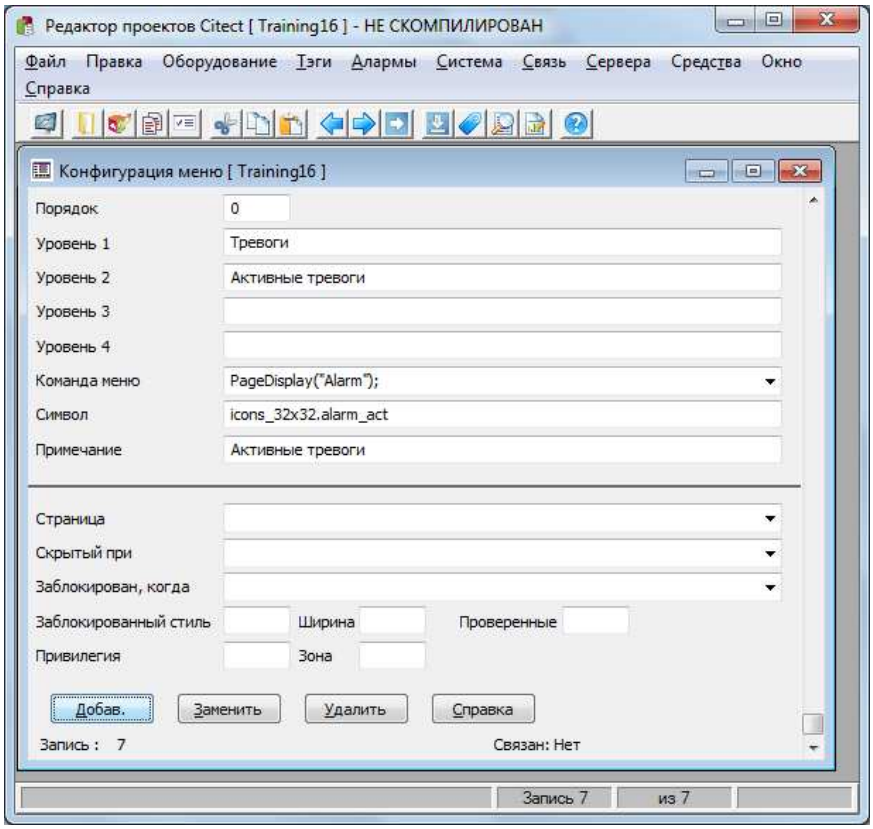
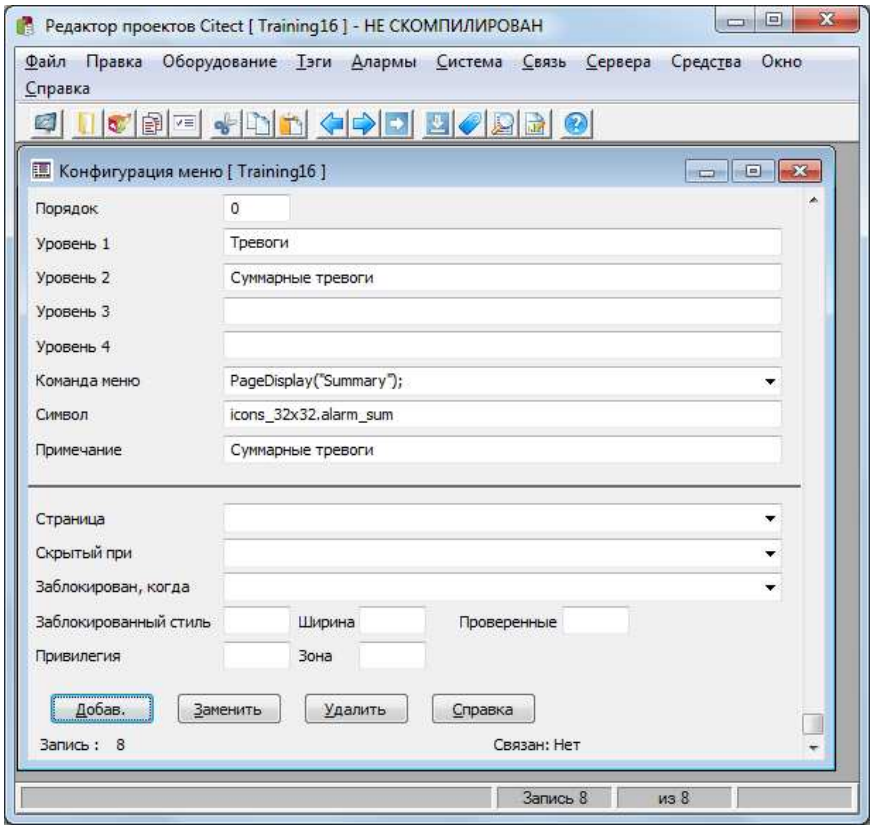
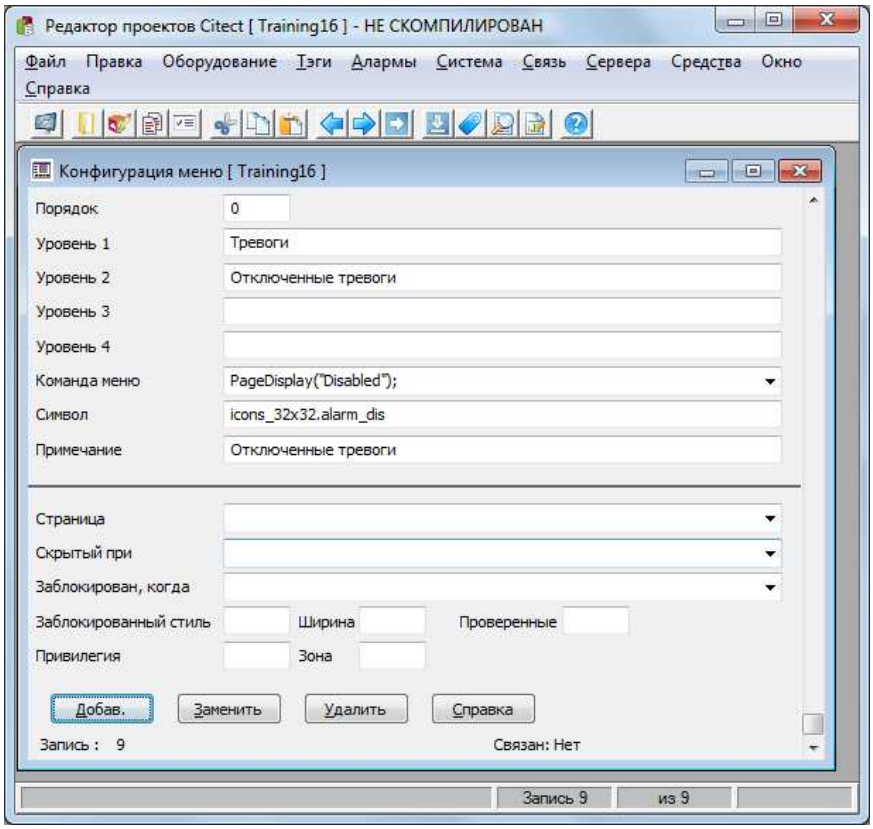
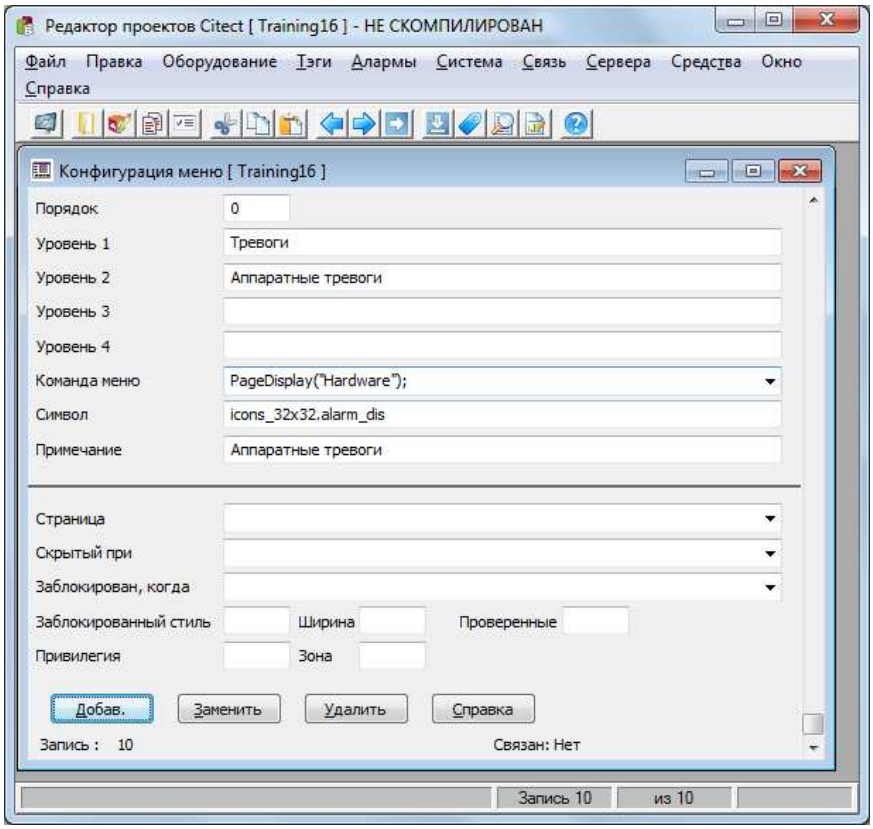


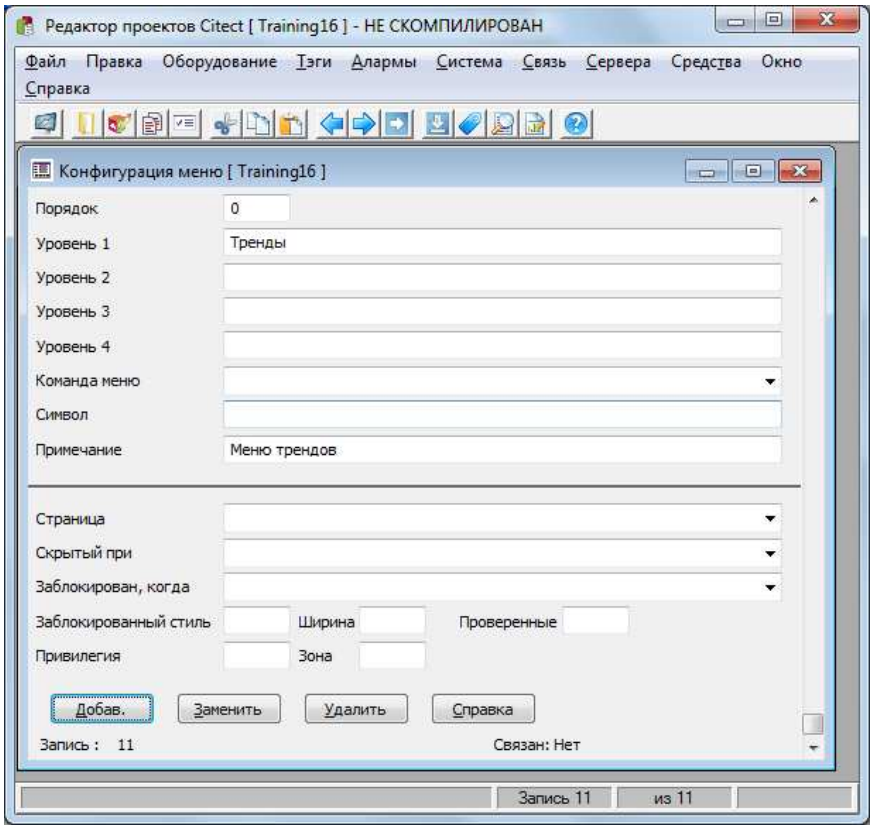
Рис. 13.2. Конфигурирование меню

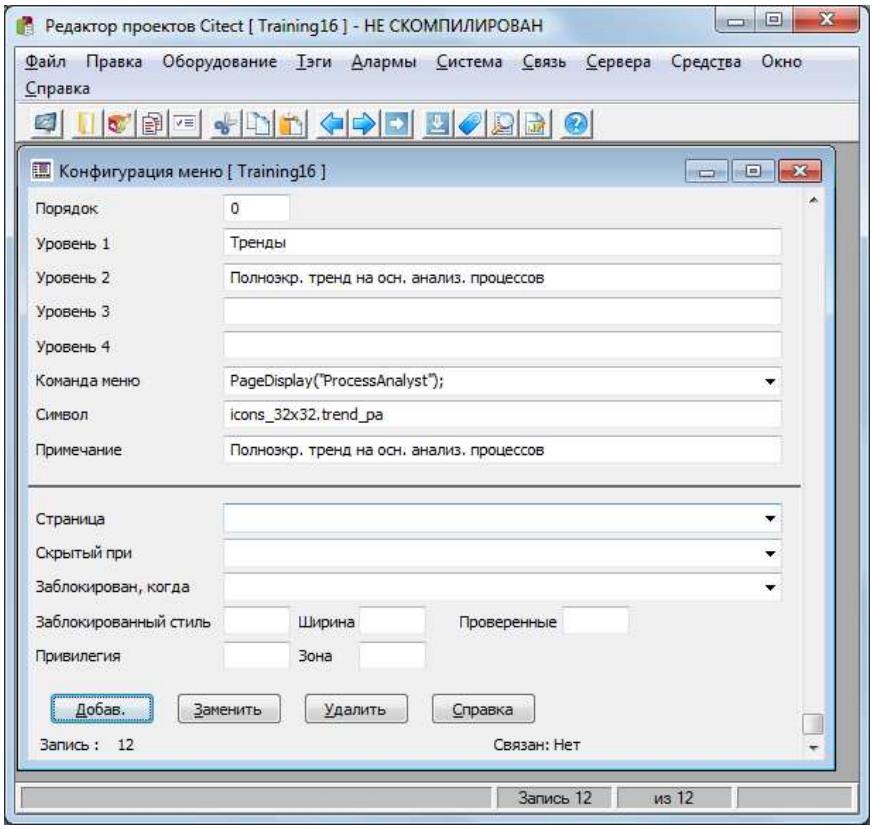

Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок
0
Уровень 1
Тревоги
Уровень 2
Активные тревоги
Уровень 3
Уровень 4
Команда меню
PageDisplay("Alarm");
Символ
icons_32x32.alarm_act
Примечание
Активные тревоги
Страница
Скрытый при
Заблокирован, когда
Заблокированный стиль
Ширина
Проверенные
Привилегия
Зона
Добав.
Заменить
Удалить
Справка
Запись : 7
Связан: Нет
Запись 7
из 7


Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок 0
Уровень 1 Тревоги
Уровень 2 Суммарные тревоги
Уровень 3
Уровень 4
Команда меню PageDisplay("Summary");
Символ icons_32x32.alarm_sum
Примечание Суммарные тревоги
Страница
Скрытый при
Заблокирован, когда
Заблокированный стиль Ширина Проверенные
Привилегия Зона
Добав. Заменить Удалить Справка
Запись : 8 Связан: Нет
Запись 8 из 8


Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок 0
Уровень 1 Тревоги
Уровень 2 Отключенные тревоги
Уровень 3
Уровень 4
Команда меню PageDisplay("Disabled");
Символ icons_32x32.alarm_dis
Примечание Отключенные тревоги
Страница
Скрытый при
Заблокирован, когда
Заблокированный стиль Ширина Проверенные
Привилегия Зона
Добав. Заменить Удалить Справка
Запись : 9 Связан: Нет
Запись 9 из 9


Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок 0
Уровень 1 Тревоги
Уровень 2 Аппаратные тревоги
Уровень 3
Уровень 4
Команда меню PageDisplay("Hardware");
Символ icons_32x32.alarm_dis
Примечание Аппаратные тревоги
Страница
Скрытый при
Заблокирован, когда
Заблокированный стиль Ширина Проверенные
Привилегия Зона
Добав. Заменить Удалить Справка
Запись : 10
Связан: Нет
Запись 10 из 10


Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок 0
Уровень 1 Тренды
Уровень 2
Уровень 3
Уровень 4
Команда меню
Символ
Примечание Меню трендов
Страница
Скрытый при
Заблокирован, когда
Заблокированный стиль
Ширина
Проверенные
Привилегия
Зона
Добав.
Заменить
Удалить
Справка
Запись : 11
Связан: Нет
Запись 11
из 11


Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок 0
Уровень 1 Тренды
Уровень 2 Полноэкр. тренд на осн. анализ. процессов
Уровень 3
Уровень 4
Команда меню PageDisplay("ProcessAnalyst");
Символ icons_32x32.trend_pa
Примечание Полноэкр. тренд на осн. анализ. процессов
Страница
Скрытый при
Заблокирован, когда
Заблокированный стиль Ширина Проверенные
Привилегия Зона
Добав. Заменить Удалить Справка
Запись : 12 Связан: Нет
Запись 12 из 12

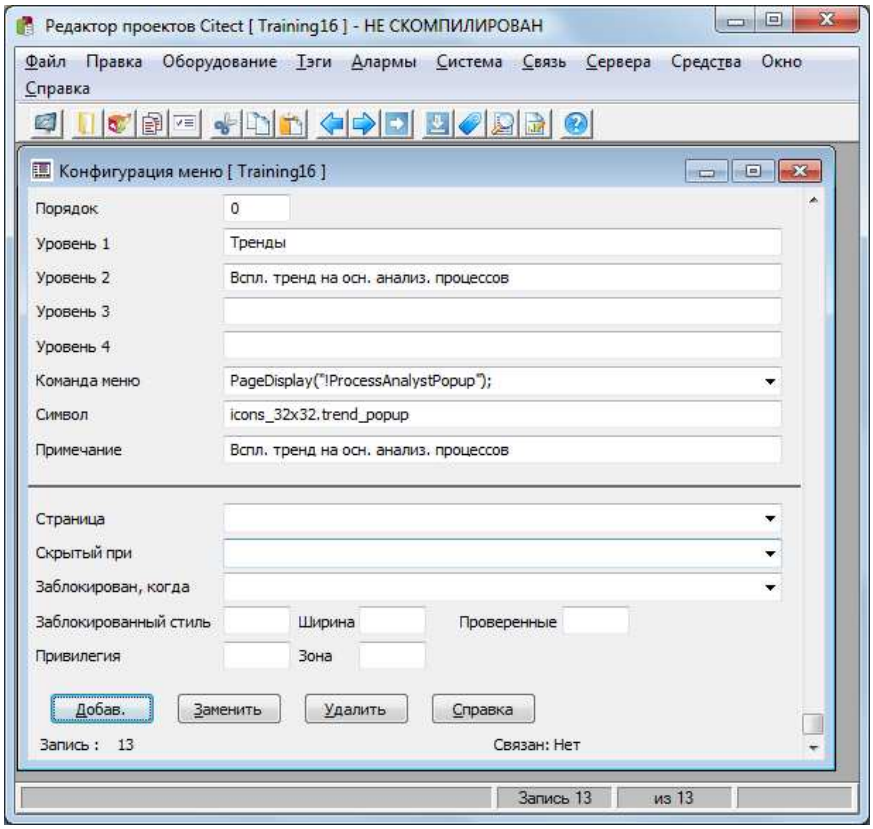
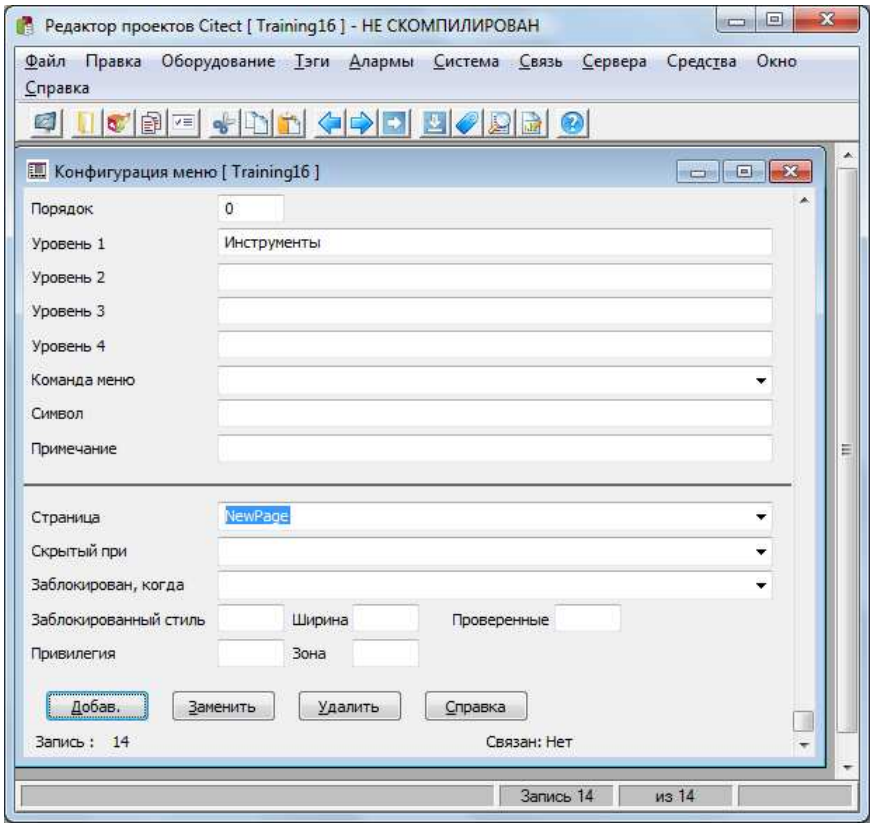


Рис. 13.3. Конфигурирование меню

Упражнение 13.2. Создание и русификация индивидуальных меню графических страниц

Для добавления индивидуальных меню графичеких страниц следует воспользоваться расширенной формой конфигурирования меню, которая появляется по нажатию клавиши **F2**.

В качестве упражнения на графическую страницу **NewPage** добавьте индивидуальное меню **Инструменты** с единственной командой перехода на страницу администратора **CSV_AdminTools** из включаемого проекта **CSV_Include**. Для этого в **Редакторе проектов Citect** выполните команду **Система | Конфигурация меню** и в появившуюся форму добавьте две записи в соответствии с рис. 13.4.


Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок
0
Уровень 1
Инструменты
Уровень 2
Уровень 3
Уровень 4
Команда меню
Символ
Примечание
Страница
NewPage
Скрытый при
Заблокирован, когда
Заблокированный стиль
Ширина
Проверенные
Привилегия
Зона
Добав.
Заменить
Удалить
Справка
Запись : 14
Связан: Нет
Запись 14
из 14

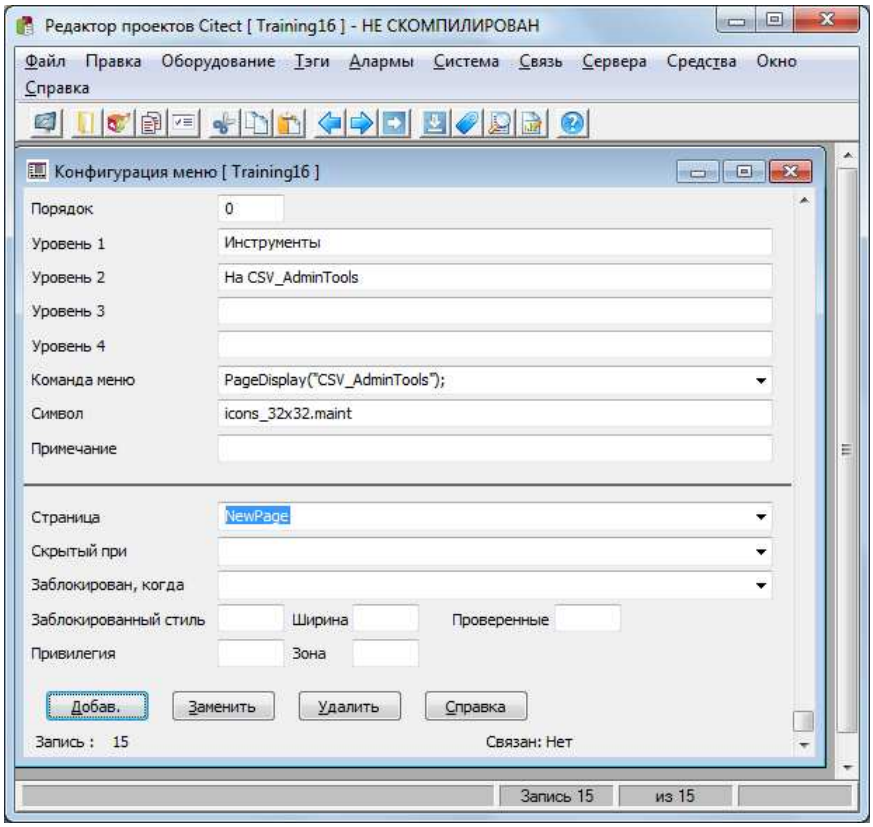


Рис. 13.4. Добавление индивидуального меню в графическую страницу **NewPage**

Примечание

В поле **Страница** расширенной формы конфигурирования меню на рис. 13.4 указана страница, для которой добавленное меню является индивидуальным. Если в этом поле ничего не указано, то, по умолчанию, меню и команда добавляются во все страницы. Если же в поле **Страница** будет указано **Template**, то, аналогично, во все страницы будет добавлено общее всплывающее меню (об этом будет сказано далее).

Выполните компиляцию, запустите проект и убедитесь, что меню страницы **NewPage** имеет вид, показанный на рис. 13.5.

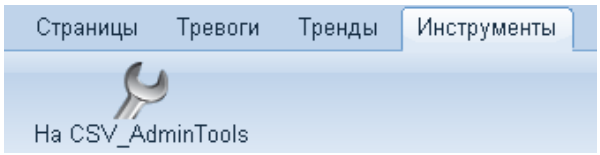


Рис. 13.5. Вид добавленного меню на графической странице **NewPage**

Протесируйте работу добавленной команды меню и остановите работу включаемого проекта.

Упражнение 13.3. Создание и русификация всплывающего меню графических страниц

По умолчанию, если всплывающее меню графичеких страниц не конфигурируется, то оно имеет вид, показанный на рис. 13.6 (для его активации следует нажать на кнопку **List Login Options**, расположенную слева от строки меню).

В качестве упражнения создайте всплывающее меню, общее для всех графических страниц, имеющее вид, представленный на рис. 13.7.

Рис. 13.6. Всплывающее меню графических страниц (формируется по умолчанию)

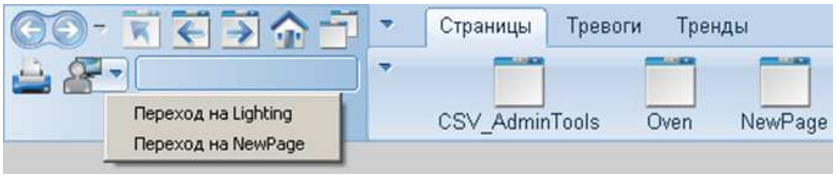


Рис. 13.7. Вид всплывающего меню, общего для всех графических страниц

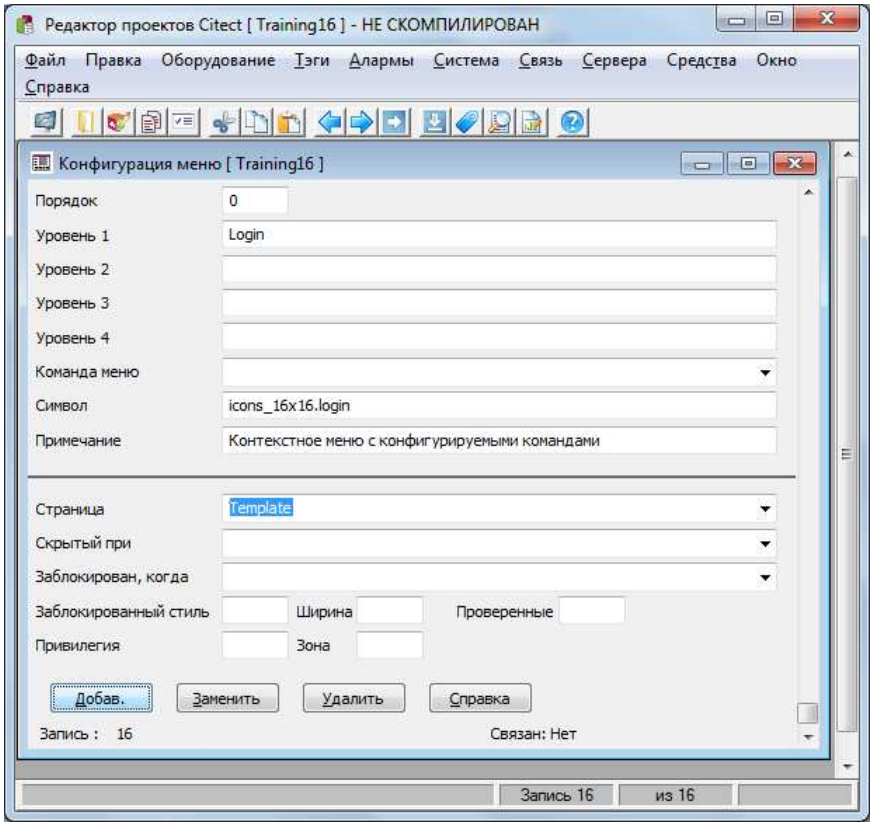
С этой целью в форму конфигурирования меню добавьте еще три записи (рис. 13.8). Выполните компиляцию, запустите проект, протестируйте работу всплывающего меню и завершите работу проекта.

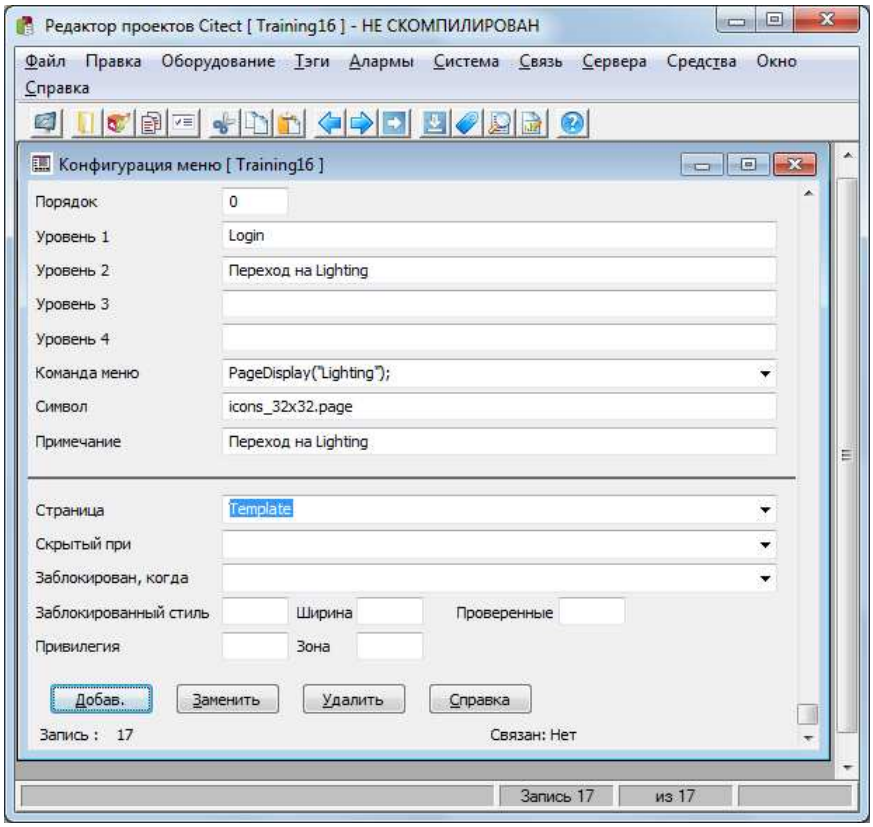
Дополнительные возможности навигации предоставляют кнопки панели инструментов, расположенные в левом верхнем углу окна (см. приведенный ранее рис. 13.7).

Кнопки  (**Go Back a Page** и **Go Forward a Page**) позволяют выполнять откат назад или переход вперед на одну графическую страницу.

Кнопка (**List Recent Pages**) показывает список всех графических страниц данного проекта и позволяет перейти на требуемую страницу (рис. 13.9).

Кнопки  (**Display Previous Page** и **Display Next Page**) позволяют перейти на графические страницы, указанные в свойствах данной графической страницы (рис. 13.10, поля **Пред. страница** и **След. страница**).




Редактор проектов Citect [Training16] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Конфигурация меню [Training16]
Порядок 0
Уровень 1 Login
Уровень 2 Переход на Lighting
Уровень 3
Уровень 4
Команда меню PageDisplay("Lighting");
Символ icons_32x32.page
Примечание Переход на Lighting
Страница Template
Скрытый при
Заблокирован, когда
Заблокированный стиль Ширина Проверенные
Привилегия Зона
Добав. Заменить Удалить Справка
Запись : 17
Связан: Нет
Запись 17 из 17

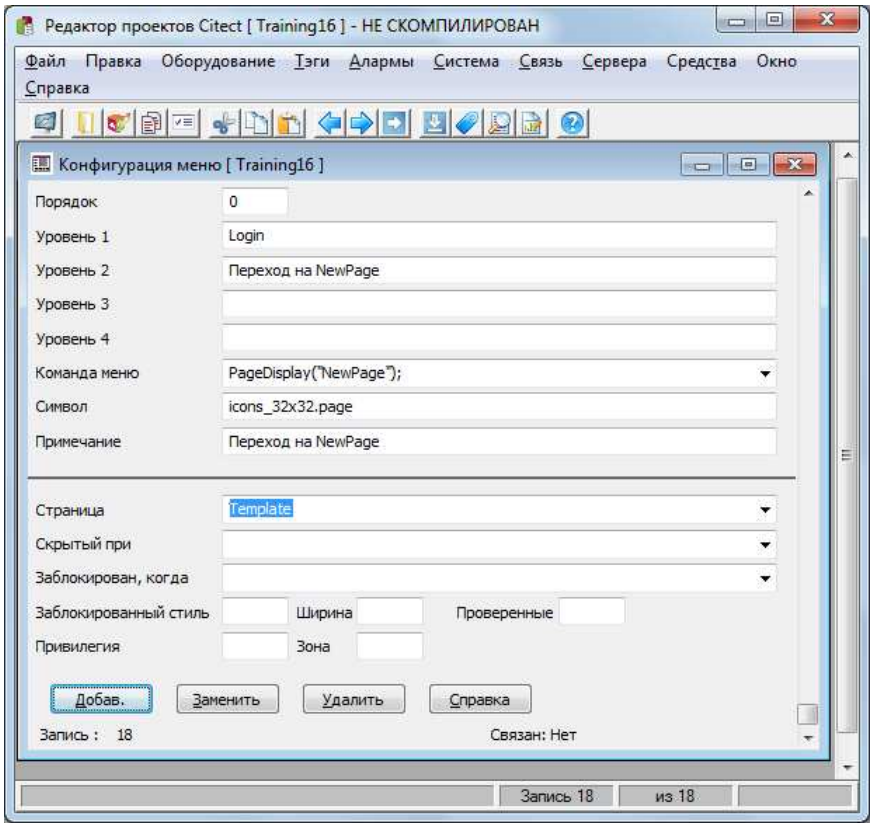


Рис. 13.8. Конфигурирование всплывающего меню, общего для всех графических страниц

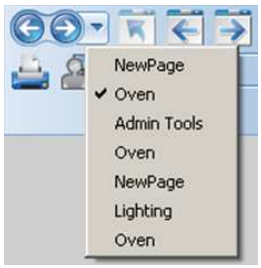


Рис. 13.9. Список для навигации на графические страницы проекта

Кнопка  (**Display Home Page**) обеспечивает навигацию на стартовую (домашнюю страницу).

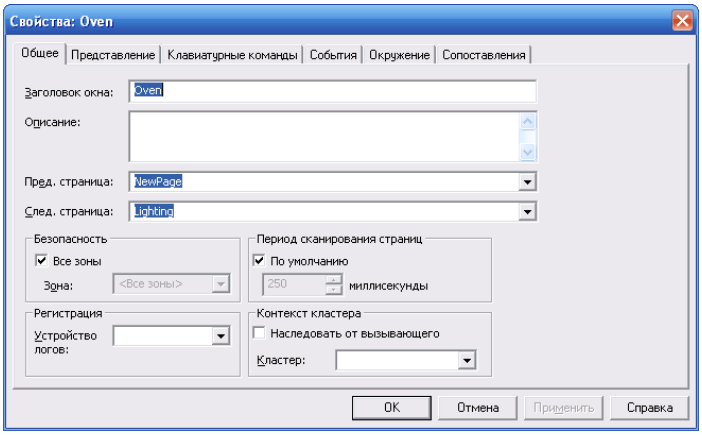


Рис. 13.10. Свойства графической страницы **Oven**

Кнопка  (**Display Page Select Dialog**) позволяет выбрать страницу из списка и по кнопке **OK** выполнить навигацию на выбранную страницу.

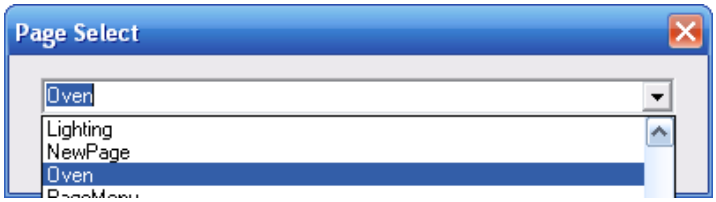


Рис. 13.11. Действие кнопки **Display Page Select Dialog**

13.2. Конфигурирование меню средствами включаемого проекта CSV_Include

Система меню имеет четыре уровня. Поле уровня **Страница** определяется либо словом **Generig** (общее), либо конкретным именем страницы внутри проекта. Имя **Generig** указывает, что меню связано со всеми графическими страницами проекта, а конкретное имя страницы указывает, что меню появляется только на данной странице.

Примечание
Ранее, в расширенной форме конфигурирования меню в поле **Страница** вместо **Generig** ничего не указывалось.

На уровне **Меню** определяются имена главных меню. На уровне **Вложенное меню** определяются меню, расположенные внутри другого меню. На уровне **Команда меню** определяются команды главного меню или подменю.

Упражнение 13.4. Конфигурирование меню. Проект Training17

Запустите демонстрационный проект **Training16** и зарегистрируйтесь как привилегированный пользователь. В качестве имени и пароля используйте **Engineer**. Перейдите на графическую страницу **CSV_AdminTools** (команда **Страницы | CSV_AdminTools**), кнопкой **Configure Menu** запустите средство конфигурирования меню и раскройте узел **Generic** в дереве меню. Выберите меню **Tools** и выполните команду **New Item** его контекстного меню. Измените сгенерированное имя команды **New Menu Item** на **Активные тревоги**. Для добавленной команды вызовите контекстное меню и выполните команду **Edit Item**. В поле **Action** появившегося диалогового окна **Edit Menu Item** скопируйте содержимое соответствующего поля команды **Alarms | Active Alarms**, остальные поля не меняйте и нажмите кнопку **OK**. Русифицируйте названия меню и их команд (кроме меню **Pages** и его команд). Выполните просмотр файла **[DATA]:\Com_Log_14_10_2009.txt**, предназначенного для регистрации команд оператора и созданного в упр. 10.1. Для этого, как и ранее, выберите меню **Инструменты** и выполните команду **New Item** его контекстного меню. Измените сгенерированное имя команды **New Menu Item** на **Команды оператора**. Для добавленной команды вызовите контекстное меню и выполните команду **Edit Item**.

В поле **Action** появившегося диалогового окна **Edit Menu Item** поместите текст **?CSV_Nav_File Команды_оператора, [DATA]:\Com_Log_11_08_2013.txt,9**, остальные поля не меняйте и нажмите кнопку **OK**.

Примечание
Текст ?CSV_Nav_File Команды оператора,[DATA]:\Com_Log_11_08_2013.txt,9 представляет собой запись вызова функции CSV_Nav_File("Команды_оператора", "[DATA]:\Com_Log_11_08_2013.txt",9); включаемого проекта CSV_Include в виде командной строки, принятую в системе навигации. Исходный код и описание этой функции Вы можете найти в файле CSV_Navigation.ci включаемого проекта CSV_Include.

Для сохранения изменений выполните команду **Save** контекстного меню и нажмите кнопку **Да**. Закройте диалог **Menu Configuration** и, после выполнения команды **Home Page** с помощью кнопки на панели инструментов, протестируйте выполненные изменения. Завершите работу проекта. Архивируйте проект под именем **Training17**, восстановите проект **Training17** и, в дальнейшем, работайте с ним.

Упражнение 13.5. "Продвинутое" конфигурирование меню. Проект Training18

В среде приложения **Редактор проектов Citect** демонстрационного проекта **Training17** с помощью команды **Система | Пользователи** создайте еще одного привилегированного пользователя и именем и паролем **admin** и нажмите кнопку **Добавить**.

Скомпилируйте, запустите проект и зарегистрируйтесь как привилегированный пользователь. Командой **Средства | Страница администратора** откройте страницу администратора и запустите средство конфигурирования меню (**Configure Menu**). Вместо привязки создаваемого меню к конкретной графической странице, привяжем его к графическому объекту **Login**, имеющемся в шаблоне **Normal** графической страницы. Нажмите правую кнопку мыши в левом поле окна **Menu Configuration** и вызовите команду **New Page**. Назовите новую страницу **Template**. Нажмите правую кнопку мыши на **Template** и выполните команду **New Button** контекстного меню. Назовите кнопку **Login**. Нажмите правую кнопку мыши на **Login** и выполните команду **New Item** контекстного меню. Назовите команду **Привилегированный пользователь 1**. Нажмите правую кнопку мыши на **Привилегированный пользователь 1** и выполните команду **Edit Item** контекстного меню, в поле **Action** введите **?Login admin, admin** и нажмите кнопку **OK**. Аналогично добавьте еще три команды (табл. 13.1).

Таблица 13.1. *Добавление команд*

Item Name	**Action**
Привилегированный пользователь 2	?Login Engineer, Engineer
Регистрация	?LoginForm
Разрегистрация	?Logout

Сохраните изменения. Для этого нажмите правую кнопку мыши в левом поле окна **Menu Configuration** и выполните команду **Save** контекстного меню. В ответ на запрос нажмите кнопку **Да** и закройте окно **Menu Configuration**. Поэкспериментируйте с кнопкой, расположенной справа от кнопки **Login/Logout** на панели инструментов. Убедитесь, что такая возможность имеется на всех графических страницах включаемого проекта **CSV_Include**. Завершите работу проекта. Архивируйте проект под именем **Training18**, восстановите проект **Training18** и, в дальнейшем, работайте с ним.

Другой способ настройки навигации в проекте заключается в использовании секции **[Navigation]** в файле **citect.ini**.

Упражнение 13.5. Параметры навигации. Проект Training19

Данное упражнение демонстрирует, в числе прочего, использование клавиатурного ввода системного уровня.

В среде приложения **Построитель графики Citect** демонстрационного проекта **Training18** откройте страницу **CSV_Start** проекта **CSV_Include**. Сохраните ее в проекте **Training18** под именем **Home**. Модифицируйте размещенный на странице текст и добавьте новый в соответствии с рис. 13.12.

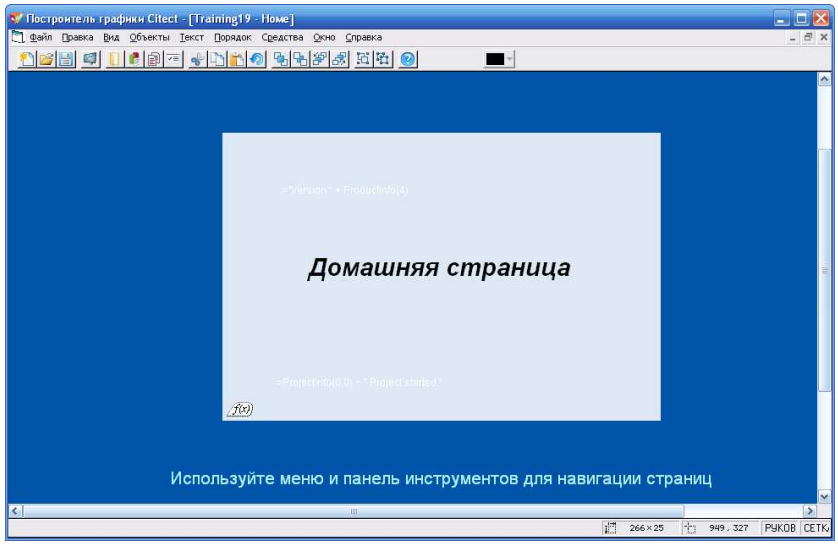


Рис. 13.12. Домашняя страница **Home** проекта **Training18**

Сохраните внесенные изменения. С помощью клавиатурных команд **Система | Клавиши клавиатуры** и **Система | Клавиатурные команды** системного уровня обеспечьте отображение графической страницы **Home** при нажатии клавиши **Home**. С этой целью в среде **Редактор проектов Citect** выполните команду **Система | Клавиши клавиатуры**, задайте параметры в соответствии с рис. 13.13 и нажмите кнопку **Добавить**.

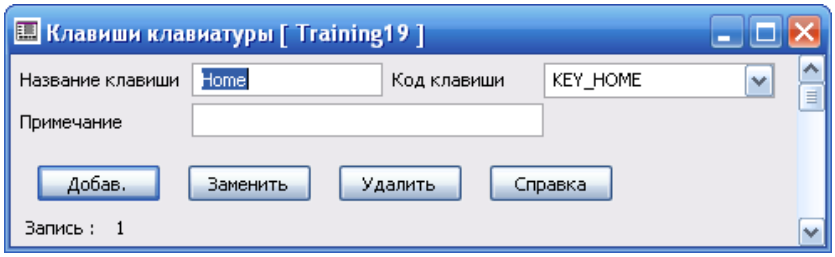


Рис. 13.13. Конфигурирование клавиатурного ввода системного уровня

Примечание
В поле **Название клавиши**, в принципе, можно использовать любой другой идентификатор.

Далее выполните команду **Система | Клавиатурные команды**, задайте параметры в соответствии с рис. 13.14 и также нажмите кнопку **Добавить**.

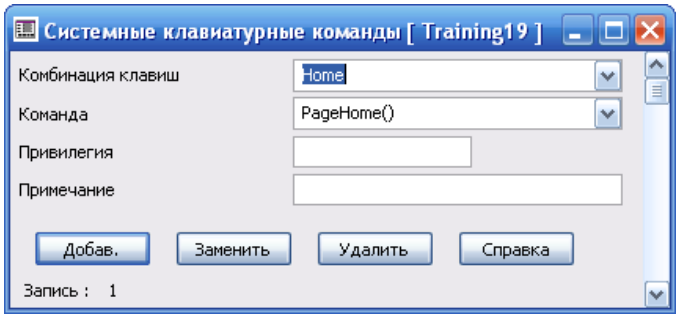


Рис. 13.14. Конфигурирование клавиатурного ввода системного уровня

Запустите **Редактор настройки компьютера**, настройте его в соответствии с рис. 13.15, сохраните конфигурацию и закройте **Редактор настройки компьютера**.

Выполните компиляцию, запустите среду приложения **Мастера конфигурирования компьютера** в режиме **Полная установка** и в диалоговом окне **Общие настройки** укажите в качестве первой (**Начальная страница**) страницу **Home**.

Запустите проект. С помощью команды **Pages | New Page** откройте страницу **New Page**. Нажатием клавиши **Home** откройте страницу **Home**. Завершите работу проекта. Архивируйте проект под именем **Training19**, восстановите проект **Training19** и, в дальнейшем, работайте с ним.

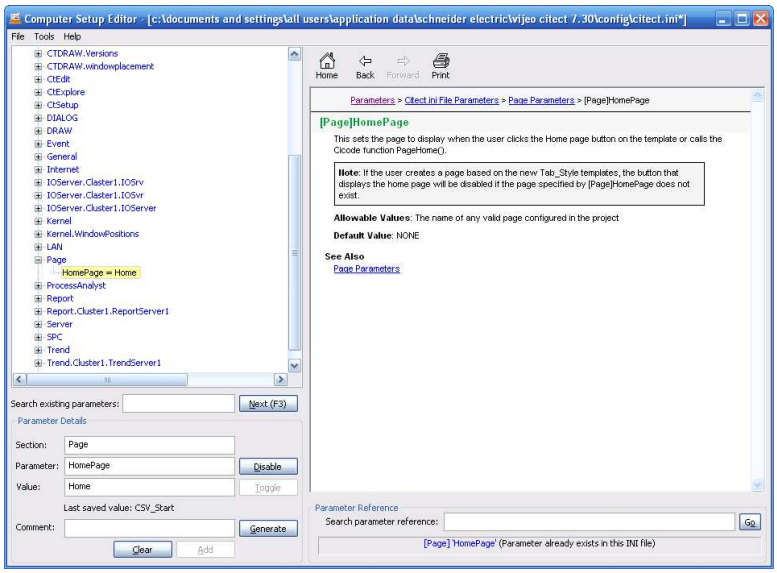


Рис. 13.15. Параметры секции **[PAGE]** файла **citect.ini**

Глава 14. Отчеты: определение, создание и просмотр отчета

Замечание
Материал, относящийся к данной теме можно найти в [2], тема Using Vijeo Citect | Reporting Information; [3], слайды 283 — 293.

Пользователь может затребовать регулярные отчеты о состоянии всего производства или отчеты, представляющие информацию о возникновении особых ситуаций в работе оборудования. Отчеты могут создаваться на основе запросов оператора, в указанное время и/или при возникновении определенных событий. *Отчеты могут содержать выражения языка Cicode, которые исполняются при создании отчета.*

Отчеты, подобно событиям, могут создаваться периодически или по некоторому условию (или и то и другое). Кроме того, отчеты могут создаваться в любое время по запросу оператора с помощью функции **Report()** языка Cicode. Форма отчета указывается в файле формата отчета, а сам отчет выводится на определенное устройство.

Таким образом, отчет является средством архивации событий и процессов и представления соответствующей информации оперативному персоналу. При работе с отчетом его нужно определить (сконфигурировать), создать отчет и посмотреть созданный отчет.

Для определения отчета в среде приложения **Проводник Citect** в поле **Список проектов** выберите проект, раскройте его, выберите папку **Система** и выполните двойной щелчок левой кнопкой мыши на значке **Отчеты** в поле **Содержимое Система**. Другим способом определения события является переход в среду приложения **Редактор проектов Citect** и выполнение команды **Система | Отчеты**.

Упражнение 14.1. Определение (конфигурирование) отчета

В среде приложения **Редактор проектов Citect** демонстрационного проекта **Training19** для работы с отчетом добавьте новое устройство. Для этого выполните команду **Система | Устройства**, сконфигурируйте устройство в соответствии с рис. 14.1 и нажмите кнопку **Добавить**. Командой **Сервера | Сервера отчетов** сконфигурируйте сервер отчетов в соответствии с рис. 14.2 и нажмите кнопку **Добавить**. Добавьте отчет в формате **RTF**. Для этого выполните команду **Система | Отчеты**, сконфигурируйте отчет в соответствии с рис. 4.3 и нажмите кнопку **Добавить**. В окне, показанном на рис. 14.3, для определения формата отчета нажмите кнопку **Правка** и задайте формат отчета в соответствии с рис. 14.4. Сохраните заданный формат, закройте окно **WordPad** и нажмите кнопку **Заменить**.

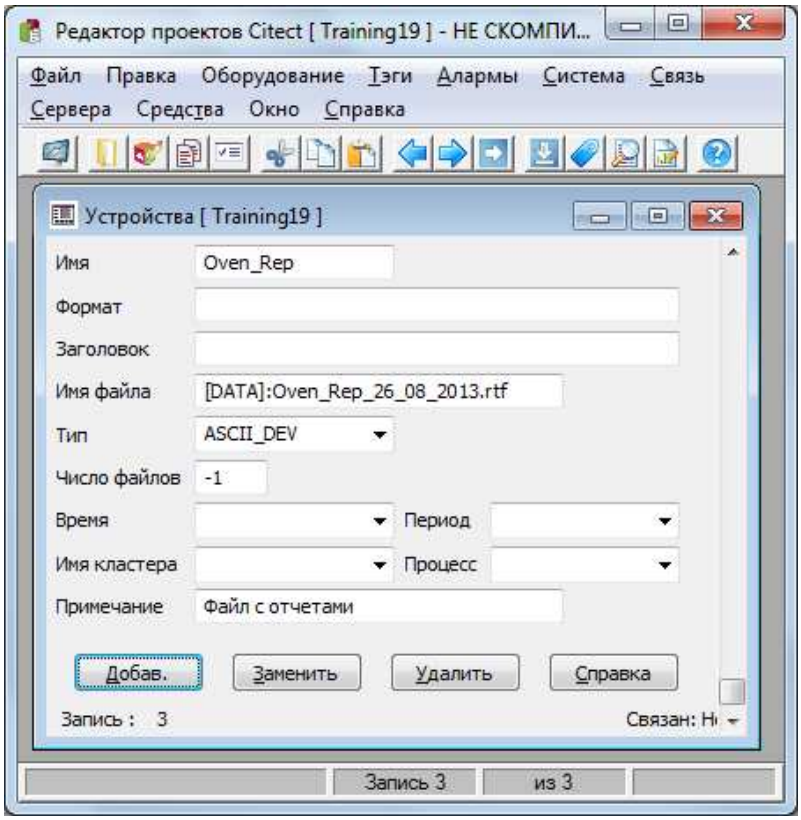


Рис. 14.1. Добавление и конфигурирование устройства для работы с отчетом

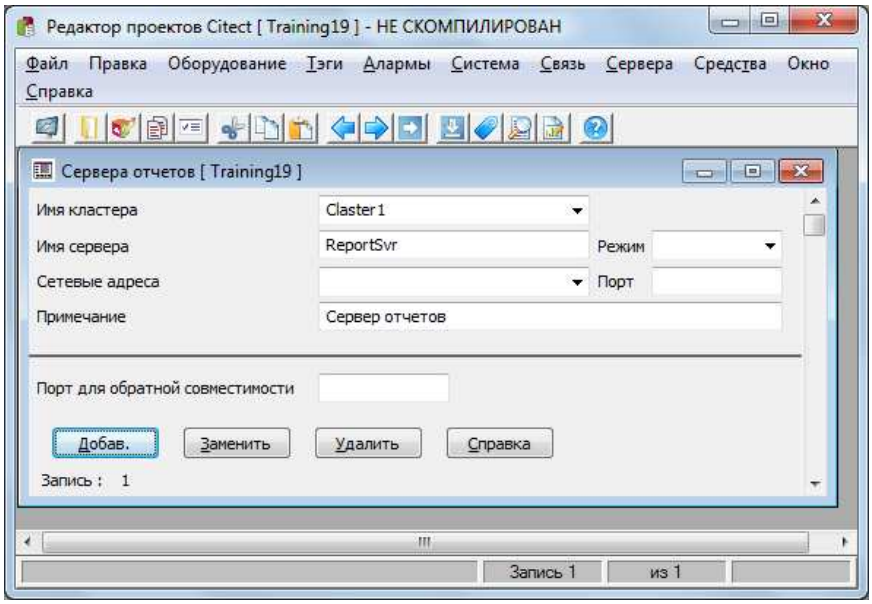


Рис. 14.2. Конфигурирование сервера отчетов

Кнопка **Edit** на приведенном ранее рис. 14.3 предназначена для задания формата отчета. По умолчанию при форматировании используется стандартный редактор **Wordpad**. Нажатие этой кнопки открывает существующий файл форматирования отчета или пустой файл в случае нового отчета. При сохранении файла форматирования отчета следует указать одно из трех расширений в зависимости от требуемого формата вывода (**.rtf** — форматирование, цвет и графика; **.txt** — неформатированный текст в кодировке ASCII, **.dbf** — файл базы данных, dbase III).

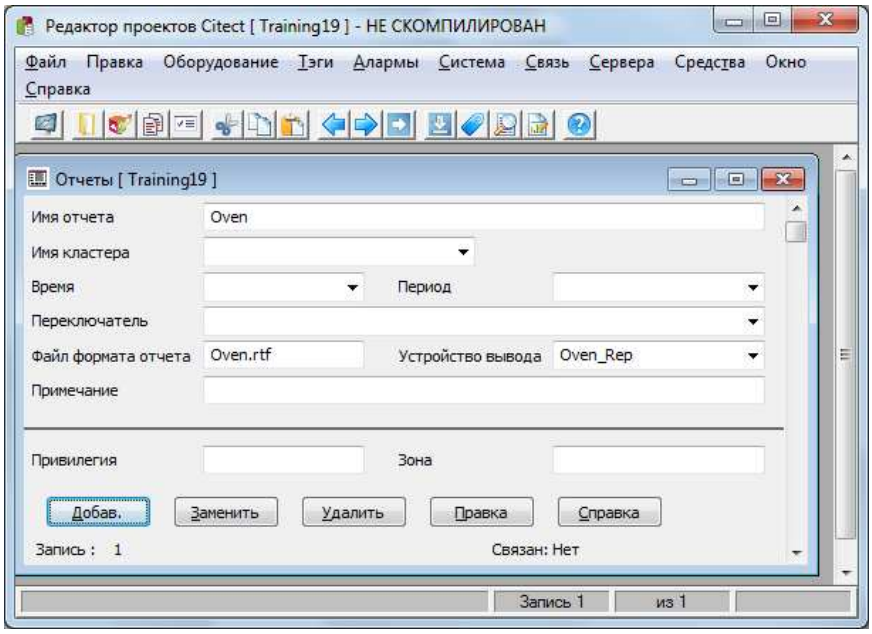


Рис. 14.3. Добавление отчета

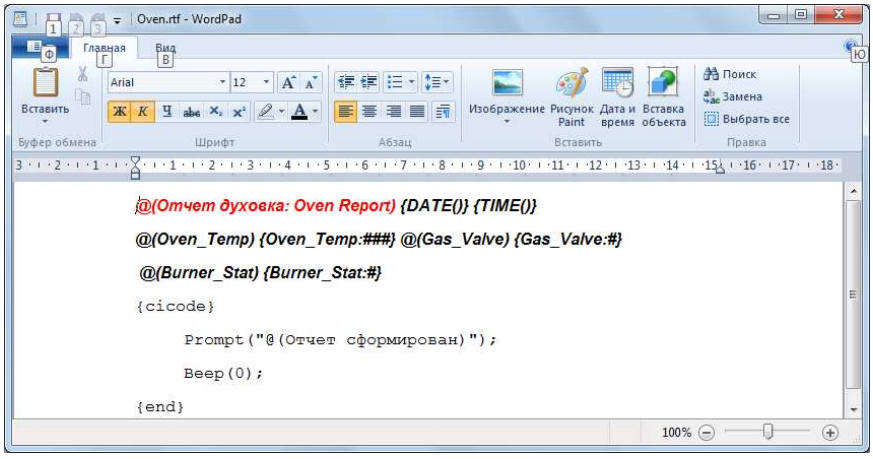


Рис. 14.4. Определение формата отчета **Oven**

Файл формата отчета может содержать различную информацию, в том числе простой текст, команды форматирования, выражения на языке Cicode и значения тегов переменных.

Совет

Очень важно! Если в файле форматирования в элементах форматирования используется текст на русском языке, то для его просмотра в нужной кодировке в операционной системе Windows XP настройте реестр следующим образом. Выполните команду **Пуск | Выполнить**, в поле **Открыть** задайте **regedit** и нажмите кнопку **OK**. Появится окно редактора реестра. Последовательно раскройте компоненты **HKEY_LOCAL_MACHINE, SYSTEM, CurrentControlSet, Control, Nls**. Выберите **CodePage**, дважды щелкните левой кнопкой мыши по компоненте **1252**, в появившемся окне **Изменение строкового параметра** в поле **Значение** задайте **c_1251.nls** и нажмите кнопку **OK**. Закройте окно редактора реестра и обязательно перезагрузите компьютер. В Windows Vista/7/8 данный метод не работает, там необходимо выполнить следующие действия:

1) Перейти в папку **%windir%\system32** и найти там два файла кодировок **C_1251.NLS и C_1252.NLS**.
2) Задать полный доступ к файлу **%windir%\system32\C_1252.NLS**. Для этого кликнуть правой кнопкой мыши по файлу **C_1252.NLS**, в появившемся контекстном меню выбрать **Свойства**, вкладку **Безопасность**, далее **Дополнительно**, вкладка **Владелец**, сменить владельца на себя, нажать **ОК**, изменить разрешения по кнопке **Изменить** на **Полный доступ** для себя (установить флажок напротив) и **Применить**.
3) После того, как мы получили полный доступ к требуемому файлу, переименовываем его с C**_1252.NLS** на **C_1252.NLS.BAK.**
4) Далее скопировать файл **C_1251.NLS** в любую удобную для вас папку, переименовать его в **C_1252.NLS**, после переименования вернуть его в папку **%windir%\system32**. Все права безопасности вернуть в исходное состояние.

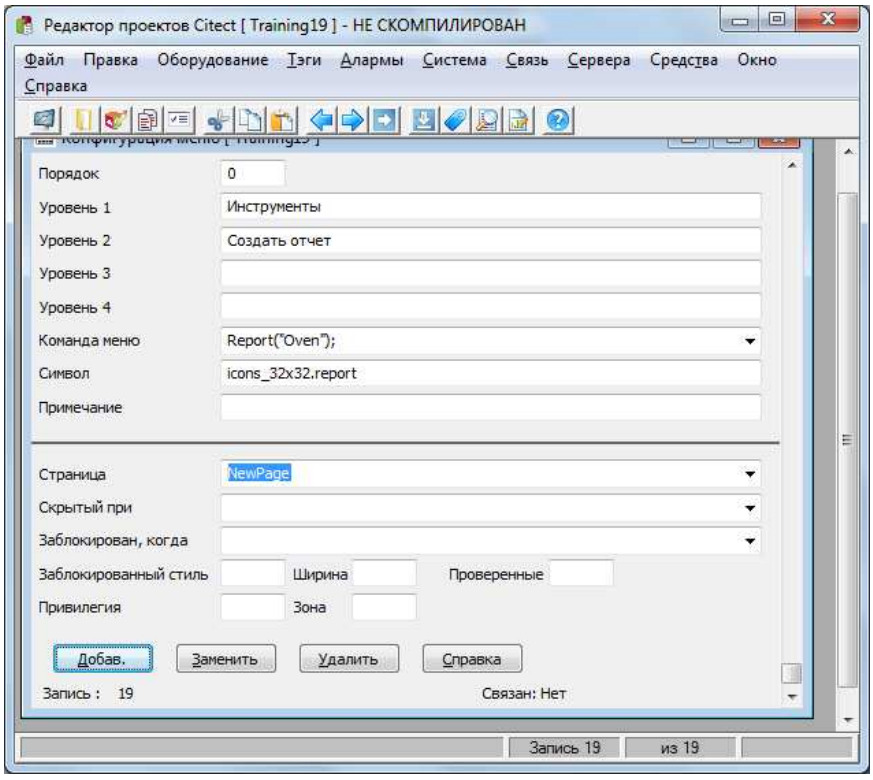
Замечание

Более подробные сведения о средствах форматирования отчетов можно найти в [2], тема Using Vijeo Citect | Reporting Information | Report Format File.

Выполните компиляцию проекта.

Упражнение 14.2. Создание и просмотр отчетов. Проект Training20

На графическую страницу **NewPage** проекта **Training19** добавьте в индивидуальное меню **Инструменты** две команды, сконфигурированные в соответствии с рис. 14.5.


Редактор проектов Citect [Training19] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно
Справка
Порядок 0
Уровень 1 Инструменты
Уровень 2 Создать отчет
Уровень 3
Уровень 4
Команда меню Report("Oven");
Символ icons_32x32.report
Примечание
Страница NewPage
Скрытый при
Заблокирован, когда
Заблокированный стиль Ширина Проверенные
Привилегия Зона
Добав. Заменить Удалить Справка
Запись : 19 Связан: Нет
Запись 19 из 19

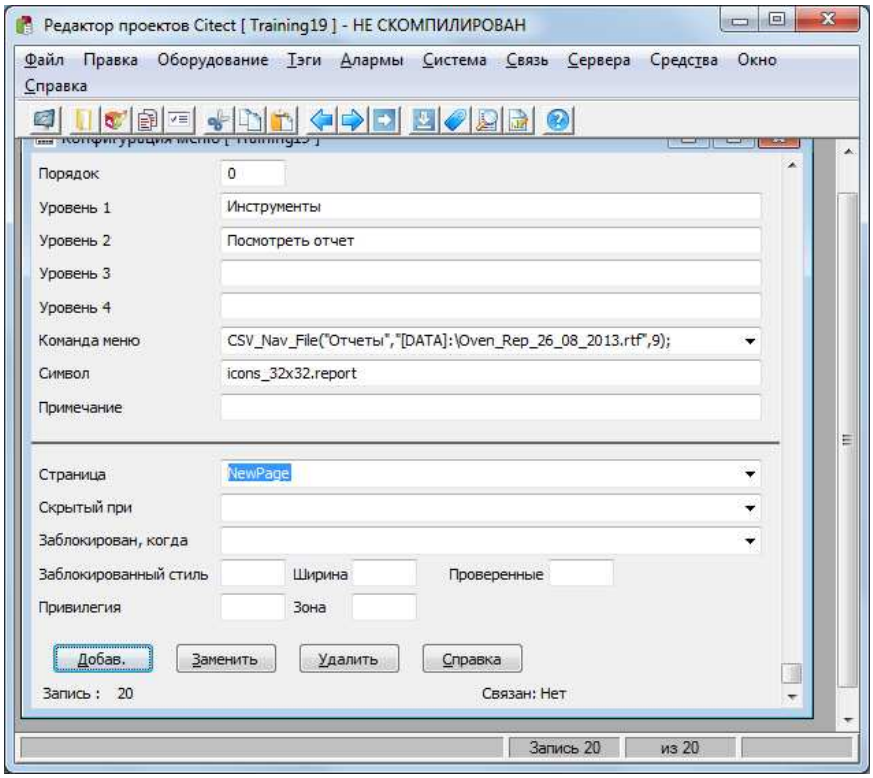


Рис. 14.5. Конфигурирование команд для создания и просмотра отчета

Выполните компиляцию, запустите проект, перейдите на графическую страницу **NewPage** и протестируйте созданные команды и завершите работу проекта. Архивируйте проект под именем **Training20**, восстановите проект **Training20** и, в дальнейшем, работайте с ним.

Глава 15. Выполнение процессов в реальном масштабе времени. Безопасность

Далее последовательно рассматриваются выполнение процессов в нежестком реальном времени и планирование безопасности предприятия, управляемого с использованием проекта системы Vijeo Citect.

15.1. Выполнение процессов в реальном масштабе времени

Любой процесс, реализованный с помощью Cicode-функции (Cicode-функций) пользователя, можно выполнять в реальном масштабе времени (РМВ). Для этой цели можно использовать обработку событий.

Упражнение 15.1. Выполнение Cicode-функции пользователя в РМВ с периодом исполнения в секундах. Проект Training21

В среде приложения **Редактор проектов Citect** демонстрационного проекта **Training20** запустите приложение **Редактор Cicode**, создайте файл **ChangeTemp** и добавьте функцию в соответствии с рис. 15.1.

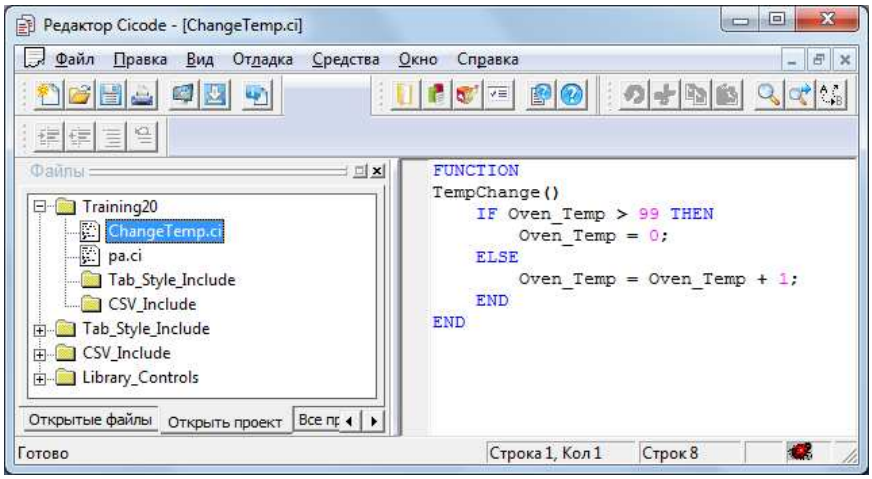


Рис. 15.1. Cicode-функция пользователя, выполняемая в PMB

В среде приложения **Редактор проектов Citect** добавьте обычное событие. Для этого выполните команду **Система | События** и сконфигурируйте событие в соответствии с рис. 15.2. Скомпилируйте проект. Запустите приложение **Мастер конфигурирования компьютера** в режиме **Полная установка,** включите событие **RunFunRealTime** и выключите событие **Temp**. Запустите проект и пронаблюдайте за изменением тега **Oven_Temp**. Повторите эксперимент, задав большую периодичность события. Обратите внимание на уменьшение темпа вызовов пользовательской функции. Завершите работу проекта. Архивируйте проект под именем **Training21**, восстановите проект **Training21** и, в дальнейшем, работайте с ним.

Замечание

Минимальное значение для периода события составляет одну секунду. Если в упражнении вместо одного события сконфигурировать два события с разными именами и периодом 1 секунда, но с одинаковыми остальными параметрами, то пользовательская функция будет вызываться с вдвое меньшими интервалами (через 0.5 секунды).

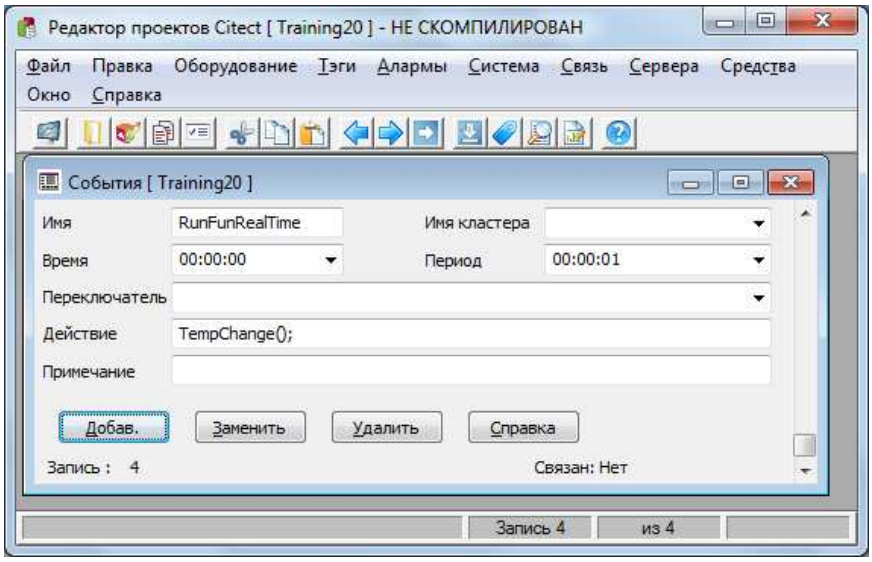


Рис. 15.2. Конфигурирование события для выполнения функции пользователя в PMB

Упражнение 15.2. Выполнение Cicode-функции пользователя в PMB с периодом исполнения в секундах и использованием нескольких событий. Проект Training22

Для выбранного проекта **Training21** в среде приложения **Редактор проектов Citect** добавьте еще одно обычное событие с другим именем и настройте его в соответствии с приведенным ранее рис. 15.2. Скомпилируйте проект. Запустите приложение **Мастер настройки компьютера** в режиме **Индивидуальная настройка** и включите добавленное событие. Запустите проект и пронаблюдайте за изменением тега **Oven_Temp**. Убедитесь в том, что интенсивность изменения тега **Oven_Temp** увеличилась в два раза.

Повторите это упражнение, использовав четыре периодических события с интервалом возникновения 1 секунда каждый. Завершите работу проекта. Архивируйте проект под именем **Training22**, восстановите проект **Training22** и, в дальнейшем, работайте с ним.

15.2. Безопасность

Система Vijeo Citect для обеспечения безопасности управления предприятием с использованием проекта Vijeo Citect предоставляет богатый набор средств. Поэтому при реализации любого варианта безопасности следует очень тщательно спланировать какие действия разрешается выполнять каждому из операторов, какие

участки разрешается видеть каждому из операторов и какие команды можно выполнять без ограничений.

В большинстве случаев оператор должен вводить команды по мере необходимости. Но могут существовать некоторые команды, выполнение которых определенными операторами следует ограничить. Данную функцию безопасности обеспечивают роли и регистрационные записи пользователей.

Безопасность может быть также основана на выделении зон, к которым оператор имеет исключительный доступ и которые указаны при определении пользователя. Наряду с этим оператору можно запретить работу и просмотр частей предприятия (зон), доступ к которым для него ограничен или запрещен. Если ни одна из зон и привилегий не определена для графической страницы или ее графического объекта, то для данного элемента по умолчанию будет определена зона 0 с привилегиями 0. Такая графическая страница или расположенный на ней графический объект не будут иметь ограничений безопасности и будут доступны для любого оператора (пользователя). Одновременное использование зон и привилегий предоставляет чрезвычайно высокий уровень безопасности.

Наряду с планированием безопасности работы предприятия, разумно обеспечить безопасность для самой системы Vijeo Citect. Например, после запуска проекта Vijeo Citect можно запретить оператору переключаться на другие приложения Windows.

15.3. Безопасность. Зоны и привилегии

При настройке проекта системы Vijeo Citect, с помощью которого выполняется управление предприятием, можно использовать зоны, привилегии или зоны и привилегии одновременно.

> ***Примечание***
> Важно! Любой пользователь, имеющий глобальные привилегии любого уровня, будет автоматически иметь доступ ко всем зонам.

Зоны и/или привилегии можно назначать следующим компонентам проекта:

1. В среде **Построителя графики Citect**.

- ❑ Графической странице (свойство **Общие**, группа **Безопасность**; свойство **Клавиатурные команды**, группа **Безопасность**).
- ❑ Объекту графической страницы (свойство **Ввод (Клавиатурные команды)**, группа **Безопасность**; свойство **Доступ (Общие)**, группа **Безопасность**; свойство **Доступ (Запрещен)**).
- ❑ Объекту **Джин** графической страницы стилей **controls**, **faceplat**, **faceplate**, **keyentry**, **tabcontrols** и **valves** (свойство **Privilege**).

2. В среде **Редактора проектов Citect**.

- ❑ Расширенная форма команды **Теги | Дескрипторы SPC** (поля **Привилегии** и **Зона**).

- Нижняя часть расширенной формы команд **Алармы | Дискретные алармы**, **Алармы | Хронологические алармы**, **Алармы | Аналоговые алармы**, **Алармы | Расширенные алармы**, **Алармы | Хронологические дискретные алармы**, **Алармы | Хронологические аналоговые алармы**, **Алармы | Мульти-дискретные алармы** (поля **Привилегии** и **Зона**).
- Нижняя часть расширенной формы команд **Система | Клавиатурные команды** (поля **Привилегии** и **Зона**), **Система | Отчеты**, **Система | Аккумуляторы**, **Система | Роли** (поле **Глобальные привилегии**), **Система | Конфигурация меню.**

Чтобы ограничить доступ к конкретной графической странице, для нее следует назначить зону. Чтобы ограничить доступ к графическому объекту страницы, следует назначить как зону, так и уровень привилегий для этого объекта.

> ***Замечание***
> Привилегии могут назначаться как исключительно (независимо), так и иерархически (например, привилегия уровня 3 имеет доступ к уровням 1 и 2). В последнем случае привилегии задаются в виде списка; 1,2,3.

При назначении зон и привилегий следует начинать с графических страниц, а затем переходить к их графическим объектам.

Чтобы назначить зону для графической страницы следует выполнить команду **Свойства страницы…** ее контекстного меню, в появившемся окне **Свойства: ИмяСтраницы** выбрать вкладку **Общее** и заполнить нужным образом поля в группе **Безопасность**.

Чтобы назначить зону и/или привилегии клавиатурной команде графической страницы следует выполнить команду **Свойства страницы…** ее контекстного меню, в появившемся окне **Свойства: ИмяСтраницы** выбрать вкладку **Клавиатурные команды** и заполнить нужным образом поля этой вкладки.

Чтобы назначить зону и/или привилегии графическому объекту страницы следует выполнить команду **Свойства…** его контекстного меню, в появившемся окне **Свойства: ИмяОбъекта** выбрать окно свойства **Доступ (Общее)** и заполнить нужным образом поля в группе **Безопасность**.

Чтобы назначить зону и/или привилегии клавиатурной команде графического объекта страницы следует выполнить команду **Свойства…** его контекстного меню, в появившемся окне **Свойства: ИмяОбъекта** выбрать окно свойства **Ввод (Клавиатурные команды)** и заполнить нужным образом поля в группе **Безопасность**.

Графический объект страницы может иметь **Стиль запрета** (стиль отображения в заблокированном состоянии), если оператор не имеет достаточных прав на зону или привилегий на работу с графическим объектом. Выбранный стиль блокировки графического объекта определяет, как этот объект будет отображаться, когда доступ к нему запрещен. Допустимыми стилями блокировки являются **Рельефный**, **Серый** и **Скрытый**.

Совет

Рекомендуем не назначать блокированным графическим объектам стиль скрытый, чтобы операторы, забывшие зарегистрироваться с достаточными правами доступа, могли бы, по крайней мере, знать о существовании этих графических объектов.

Чтобы определить стиль блокировки графического объекта страницы следует выполнить команду **Свойства…** его контекстного меню, в появившемся окне **Свойства: ИмяОбъекта** выбрать окно свойства **Доступ (Запрещен)** и выбрать нужный стиль блокировки графического объекта.

Упражнение 15.3. Назначение зон и привилегий графическим страницам и графическим объектам

Архивируйте проект **Training19** под именем **Security**, восстановите проект **Security** и, в дальнейшем, работайте с ним.

Создайте две зоны.

В зону 1 включите графическую страницу **Oven** и расположенный на ней графический объект, управляющий подачей топлива по трубопроводу. С этой целью перейдите в среду приложения **Построитель графики Citect**, откройте графическую страницу **Oven**, выполните команду **Свойства страницы…** ее контекстного меню, выберите вкладку **Общие**, модифицируйте параметры страницы в соответствии с рис. 15.3 и нажмите кнопку **OK**.

Для назначения зоны и привилегий графическому объекту, представляющему задвижку на трубе подачи топлива, выполните команду **Свойства…** его контекстного меню, в окне **Свойства: Набор образов** настройте свойства **Доступ (Общие)** и **Доступ (Запрещен)** в соответствии с рис. 15.4 и нажмите кнопку **OK**.

В зону 2 включите графическую страницу **NewPage** и расположенные на ней кнопки и графический объект **Текст**. С этой целью в среде приложения **Построитель графики Citect** откройте графическую страницу **NewPage**, выполните команду **Свойства страницы…** ее контекстного меню, последовательно выберите вкладки **Общие** и **Клавиатурные команды**, модифицируйте параметры страницы в соответствии с рис. 15.5 и нажмите кнопку **OK**.

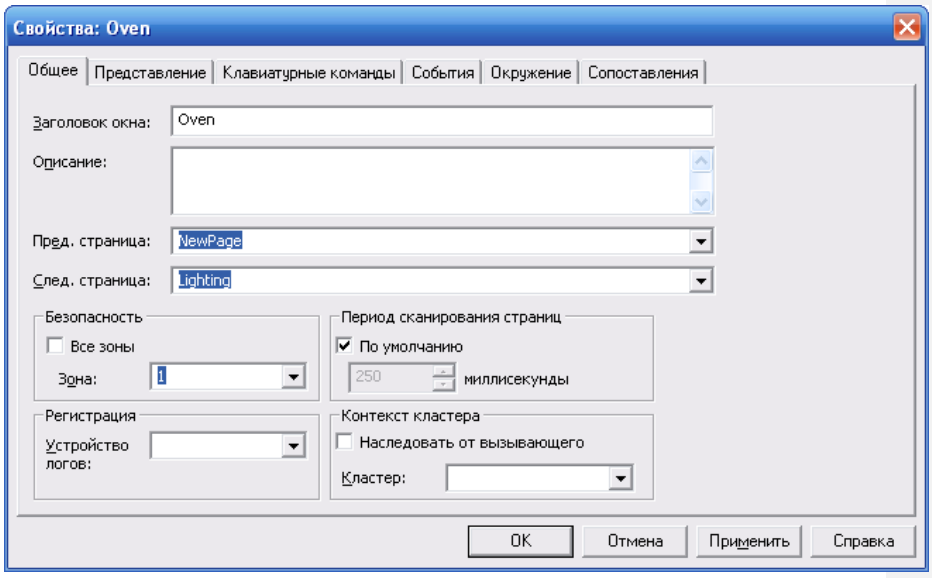
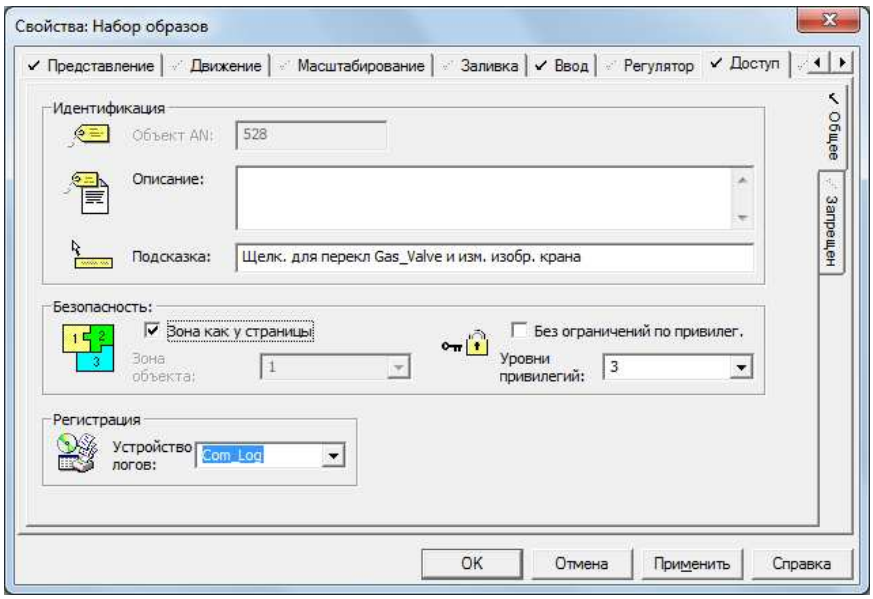


Рис. 15.3. Включение графической страницы **Oven** в зону 1



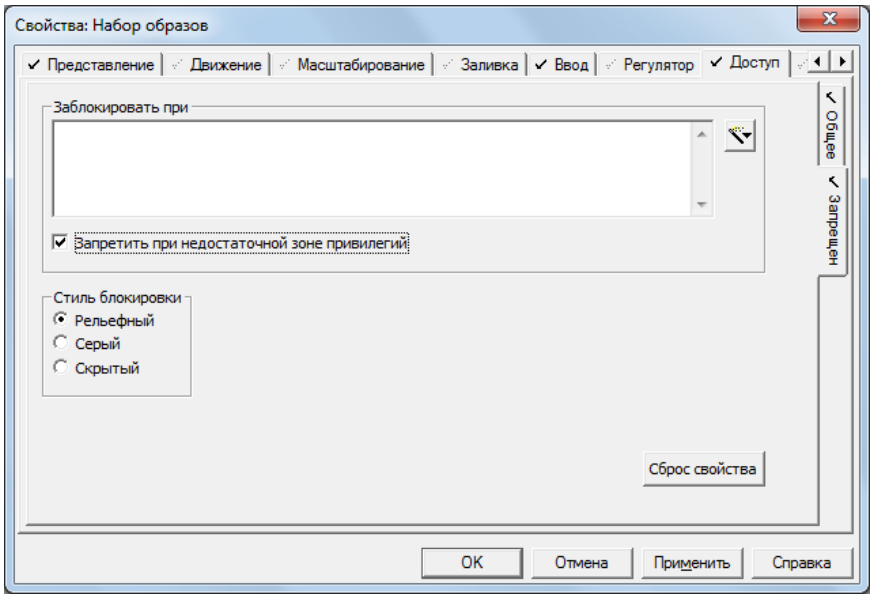


Рис. 15.4. Назначение зоны и привилегий графическому объекту **Набор образов**

Обратите внимание, что страница **NewPage** предусматривает использование клавиатурного ввода уровня страницы (по нажатию клавиши **Down** выполняется переход на графическую страницу **Home**. Для назначения зоны и привилегий графическому объекту **Задание Test=0** выполните команду **Свойства…** его контекстного меню, в окне **Свойства: Кнопка** настройте свойства **Доступ (Общие)** и **Доступ (Запрещен)** в соответствии с рис. 15.6 и нажмите кнопку **OK**.

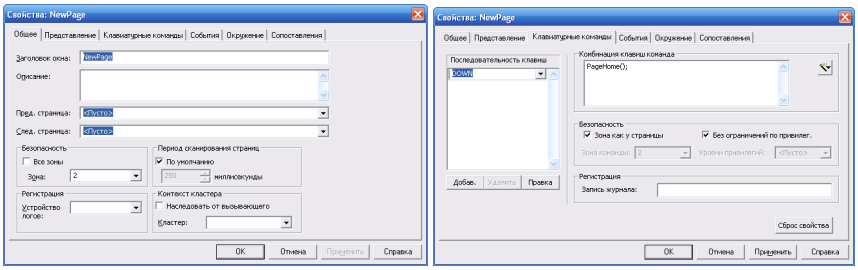


Рис. 15.5. Включение графической страницы **NewPage** в зону 2

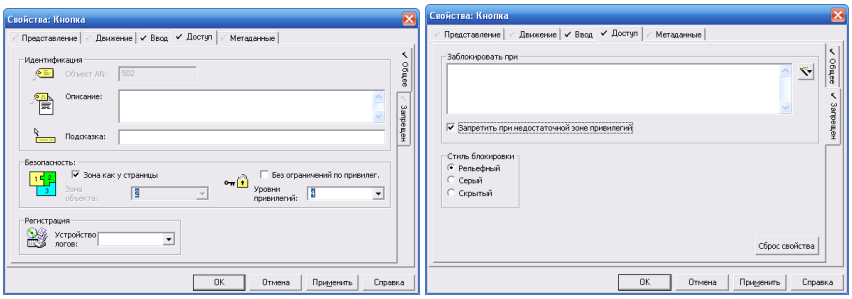


Рис. 15.6. Назначение зоны и привилегий графическому объекту-кнопке **Задание Test=0**

Аналогичным образом выполните назначение зоны и привилегий графическому объекту **Задание Test=1**. В отличие от предыдущего случая используйте стиль блокировки **Серый**. Для назначения зоны и привилегий графическому объекту **Текст** выполните команду **Свойства…** его контекстного меню, в окне **Свойства: Текст** настройте свойства **Доступ (Общие)** и **Доступ (Запрещен)** в соответствии с рис. 15.7 и нажмите кнопку **OK**. Сохраните графические страницы **Oven** и **NewPage** выполните компиляцию.

15.4. Безопасность. Добавление пользователей

Для каждого потенциального пользователя (оператора) проекта системы Vijeo Citect необходимо добавить в базу данных запись, определяющую его возможности в период выполнения проекта (указанные возможности зависят от используемой пользователем роли). Каждый такой предопределенный пользователь может во время выполнения проекта зарегистрироваться и получить соответствующий доступ к зонам (графическим страницам и графическим объектам).

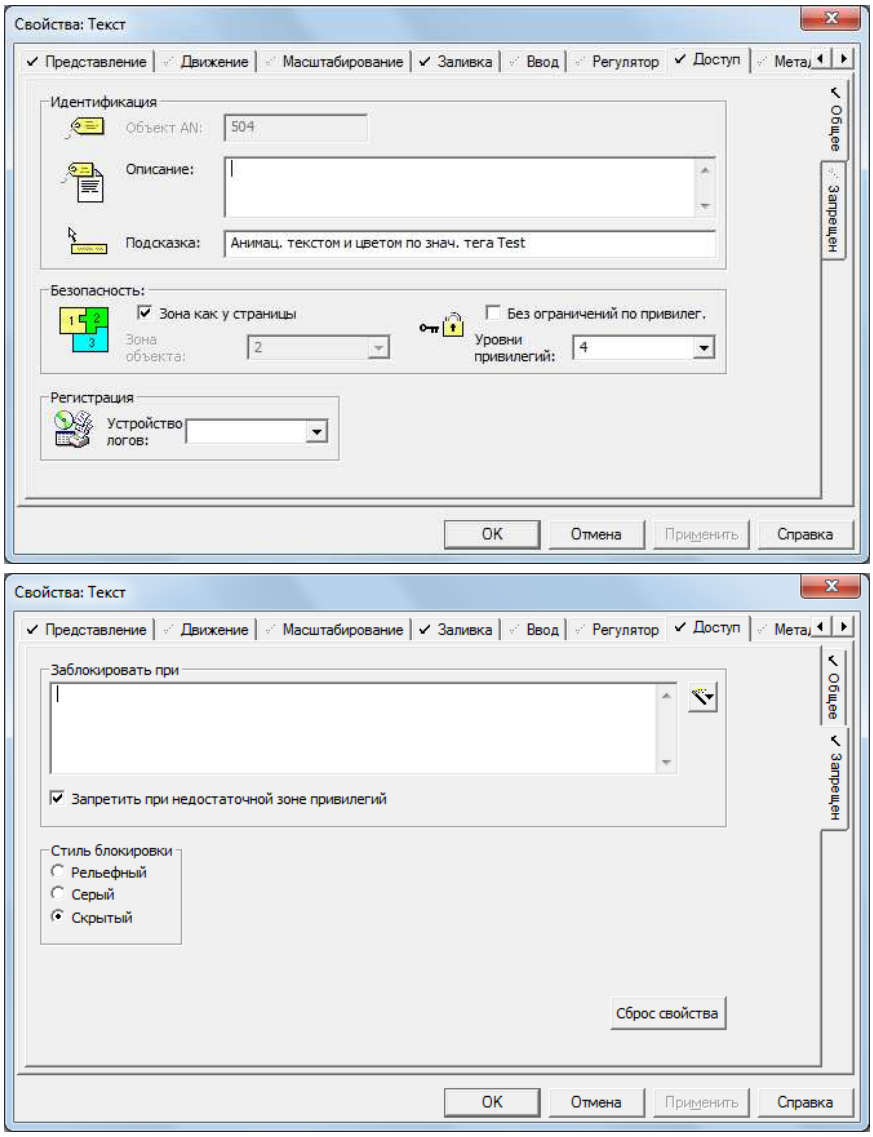


Рис. 15.7. Назначение зоны и привилегий графическому объекту **Текст**

Замечание

Во время выполнения проекта для управления операторами можно использовать ряд стандартных функций языка Cicode: `LoginForm()` — регистрация оператора, `Logout()` — разрегистрация оператора, `UserInfo()` — получение информации о текущем операторе, `UserCreateForm()` — создание нового оператора, `UserpasswordForm()` — изменение пароля оператора. Первые три функции использовались в ранее рассмотренных учебных проектах.

Перед заданием новых пользователей добавим в проект несколько новых ролей. Для этого достаточно в среде **Редактора проектов Citect** выполнить команду **Система | Роли** и добавить роли для вновь создаваемых пользователей в соответствии с рис. 15.8, каждый раз нажимая кнопку **Добавить**.

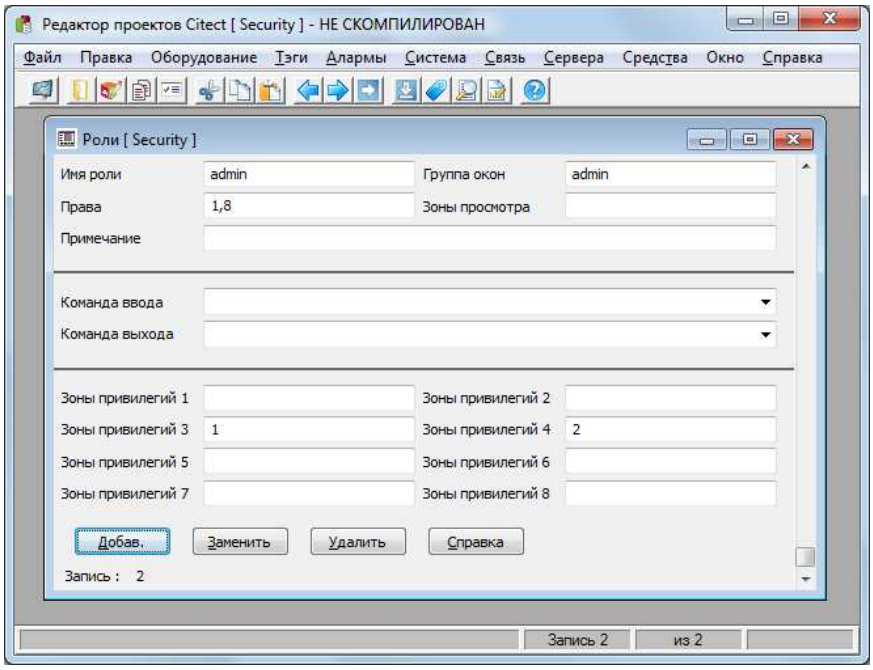
Для создания оператора (пользователя) в среде приложения **Проводник Citect** в поле **Список проектов** выберите проект, раскройте его, выберите папку **Система** и выполните двойной щелчок левой кнопкой мыши на значке **Пользователи** в поле **Содержимое Система**. Другим способом определения оператора является переход в среду приложения **Редактор проектов Citect** и выполнение команды **Система | Пользователи**.

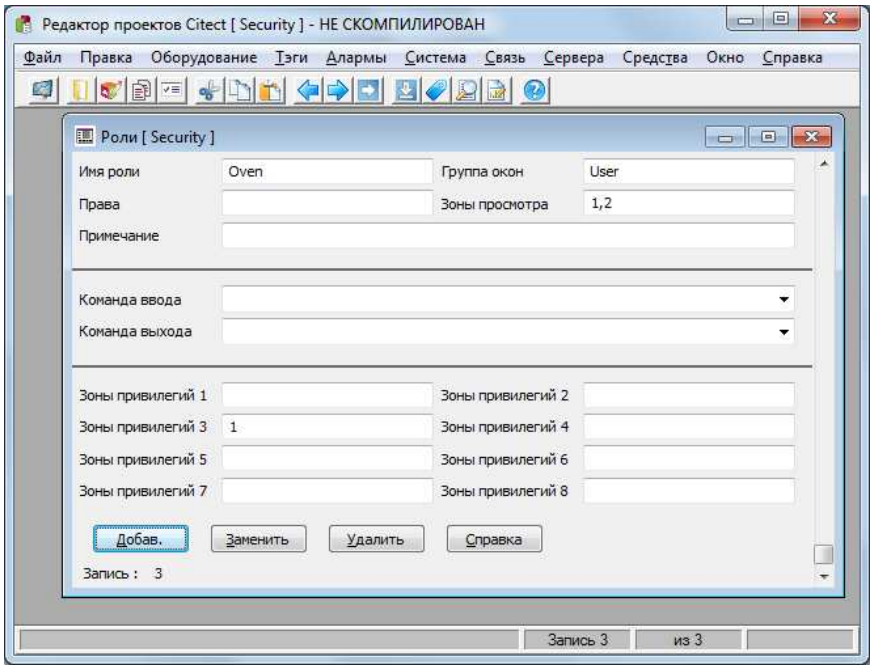
Упражнение 15.4. Модификация параметров пользователей. Добавление пользователей

Для реализации политики безопасности создайте четырез новых пользователей. С этой целью перейдите в среду приложения **Редактор проектов Citect**, выполните команду **Система | Пользователи** и сконфигурируйте новых пользователей в соответствии с рис. 15.9 (пароли добавляемых пользователей совпадают с их именами), каждый раз нажимая кнопку **Добавить**.

Примечание

Обратите внимание на два момента. В поле **Зоны доступные для просмотра** для пользователей **Engineer**, **admin**, **dvg** и **NewPage** ничего не указано, т. к. эти пользователи имеют в используемых ролях глобальные привилегии и, следовательно, автоматически имеют доступ ко всем зонам. Содержимое полей **Зоны для привилегий 3** и **Зоны для привилегий 4** коррелированно с уровнями приоритетов графических страниц и графических объектов зон 1 и 2 (см. приведенные ранее рис. 15.4 — 15.7).


Редактор проектов Citect [Security] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Роли [Security]
Имя роли
admin
Группа окон
admin
Права
1,8
Зоны просмотра
Примечание
Команда ввода
Команда выхода
Зоны привилегий 1
Зоны привилегий 2
Зоны привилегий 3
1
Зоны привилегий 4
2
Зоны привилегий 5
Зоны привилегий 6
Зоны привилегий 7
Зоны привилегий 8
Добав.
Заменить
Удалить
Справка
Запись : 2
Запись 2
из 2


Редактор проектов Citect [Security] - НЕ СКОМПИЛИРОВАН
Файл Правка Оборудование Тэги Алармы Система Связь Сервера Средства Окно Справка
Роли [Security]
Имя роли
Oven
Группа окон
User
Права
Зоны просмотра
1,2
Примечание
Команда ввода
Команда выхода
Зоны привилегий 1
Зоны привилегий 2
Зоны привилегий 3
1
Зоны привилегий 4
Зоны привилегий 5
Зоны привилегий 6
Зоны привилегий 7
Зоны привилегий 8
Добав.
Заменить
Удалить
Справка
Запись : 3
Запись 3
из 3

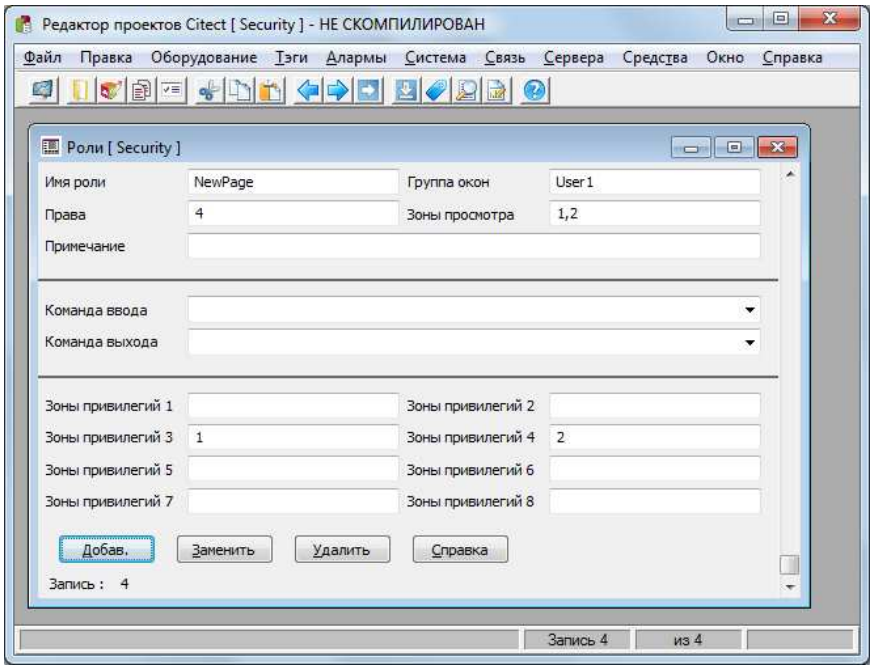


Рис. 15.8. Добавление новых ролей

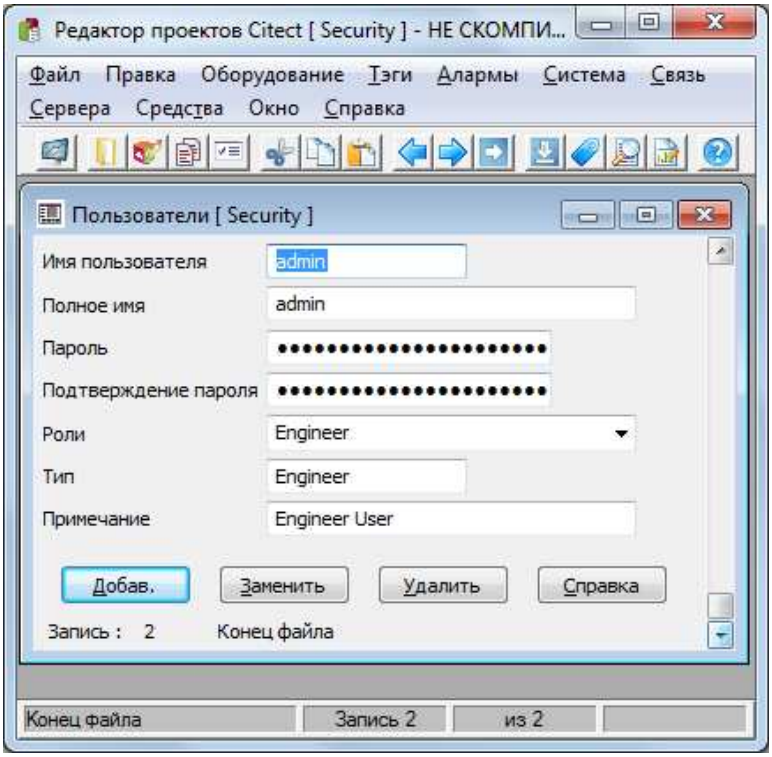
Для проверки выполните компиляцию проекта.

Упражнение 15.5. Тестирование созданной политики безопасности

Запустите проект **Security** и протестируйте политику безопасности с правами доступа по умолчанию для стартовой страницы **Lighting**. Для работы в зоне 1 выполните команду **Страницы | Oven**. Обратите внимание, что перехода на графическую страницу **Oven** не происходит, а в поле **Prompt** текущей графической страницы появляется сообщение **Страница вне вашей зоны**. Это означает, что по умолчанию работа в зоне 1 запрещена. Тестирование работы в зоне 2 путем выполнения команды **Страницы | NewPage** дает аналогичные результаты.

Далее протестируйте политику безопасности с правами привилегированных пользователей. Зарегистрируйтесь как привилегированный пользователь с именем **Engineer**. Для работы в зоне 1 выполните команду **Pages | Oven**. Произойдет переход в графическую страницу **Oven**. "Кликните" левой кнопкой мыши по клапану на трубопроводе — произойдет переключение клапана. Это означает, что работа оператора **Engineer** в зоне 1 не имеет ограничений. Тестирование работы в зоне 2

путем выполнения команды **Pages | NewPage** дает аналогичные результаты (кнопки и текстовый объект в графической странице **NewPage** полностью доступны, клавиатурный ввод уровня графической страницы — переход по клавише **Down** на графическую страницу **Home** происходит). Тестирование политики безопасности для привилегированных пользователей **dvg** и **admin** дает такие же результаты.



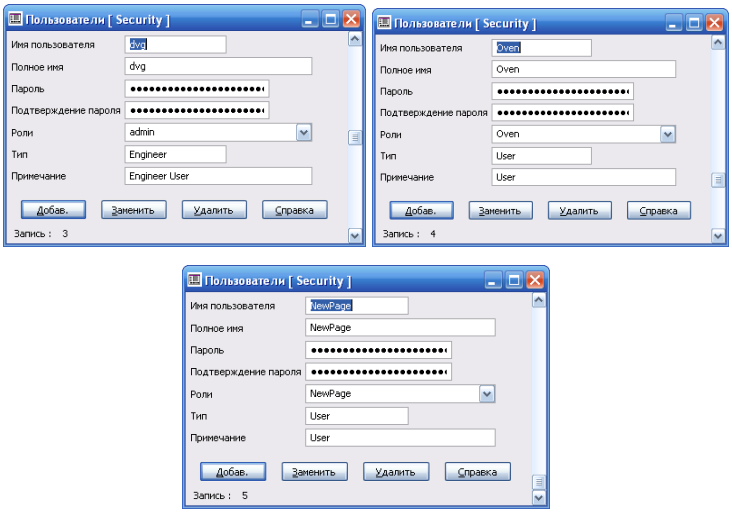


Рис. 15.9. Создание новых операторов

В заключение протестируйте политику безопасности с правами пользователей **Oven** (работа в зоне 1) и **NewPage** (работа в зоне 2).

Зарегистрируйтесь как пользователь с именем **Oven**. Для работы в зоне 1 выполните команду **Pages | Oven**. Произойдет переход в графическую страницу **Oven**. "Кликните" левой кнопкой мыши по клапану на трубопроводе — произойдет переключение клапана. Это означает, что работа оператора **Oven** в зоне 1 не имеет ограничений. Для работы в зоне 2 выполните команду **Pages | NewPage**. Обратите внимание, что перехода на графическую страницу **NewPage** не происходит, а в поле **Prompt** текущей графической страницы появляется сообщение **Страница вне вашей зоны**. Это означает, что работа пользователя **Oven** в зоне 2 запрещена.

Зарегистрируйтесь как пользователь с именем **NewPage**. Для работы в зоне 1 выполните команду **Pages | Oven**. Произойдет переход в графическую страницу **Oven** (область видимости для пользователя **NewPage** включает зоны 1 и 2). "Кликните" левой кнопкой мыши по клапану на трубопроводе — переключение клапана не произойдет, клапан блокирован. Для работы в зоне 2 выполните команду **Pages | NewPage**. Обратите внимание, что переход на графическую страницу **NewPage** имеет место, кнопки и текстовый объект в графической странице **NewPage** полностью доступны, клавиатурный ввод уровня графической страницы — переход по клавише **Down** на графическую страницу **Home** происходит. Это означает, что работа пользователя **NewPage** в зоне 2 разрешена. Остановите работу проекта.

Чтобы посмотреть, как выглядят стили отображения графических объектов **Рельефный**, **Серый** и **Скрытый**, измените роль **Oven**, указав в поле **Зоны, доступные для просмотра** значение 1,2, нажмите кнопку **Изменить**.

Выполните компиляцию, запустите проект, зарегистрируйтесь как пользователь **Oven**, перейдите на графическую страницу **NewPage** и посмотрите на вид графических объектов. Завершите работу проекта и архивируйте его под прежним именем.

Глава 16. Пользовательские шаблоны, точки анимации и пользовательские меню

Если страницы вашего проекта содержат элементы, являющиеся общими для всех графических страниц, например, такие как панели инструментов и строки состояния, вы можете создать собственный шаблон для использования в качестве основы страниц. В дальнейшем вы сможете создавать страницы на основе шаблона, добавляя к ним индивидуальные элементы.

Если впоследствии вы решите удалить или изменить расположение элемента или добавить новый общий элемент, вам не придется изменять каждую страницу — достаточно внести изменения в шаблон. Система Vijeo Citect автоматически внесет изменения во все страницы, построенные на основе шаблона.

16.1. Шаблоны Vijeo Citect

Начиная работать над новым проектом, можно создавать страницы таким образом, чтобы они выглядели в соответствии с определенными требованиями. В основе простейшей страницы лежит шаблон **Blank**, представляющий собой пустое окно (точнее — почти пустое окно). Разработчик может добавлять в это окно графические объекты и функции, а также разрабатывать новые шаблоны для страниц своего проекта.

Для разработчиков, ограниченных сжатыми сроками проектирования, а также для новых пользователей Vijeo Citect существует множество предварительно разработанных шаблонов, которые помогают быстро создавать нужные страницы. Стандартные шаблоны в предварительно сконфигурированных включаемых проектах **CSV_Include** и **Tab_Style_Include** предназначены для надежного выполнения всех основных функций, используемых в проекте.

Упражнение 16.1

Познакомьтесь с предопределенными шаблонами включаемых проектов **CSV_Include** и **Tab_Style_Include**. Для этого запустите приложение **Проводник Citect** и перейдите в среду приложения **Построитель графики Citect**. Для просмотра очередного шаблона нажмите кнопку **Открыть** на панели инструментов, в появившемся окне **Открыть** выберите вкладку **Шаблон**, в поле **Проект:** выберите включаемый проект **CSV_Include** и **Tab_Style_Include**, поле **Предназначено для показа заголовка** отметьте для проекта **Tab_Style_Include** или снимите отметку для проекта **CSV_Include**, в поле **Разрешение:** выберите **XGA**, в полях **Стиль:** и **Шаблон:** выберите необходимый шаблон и нажмите кнопку **OK**. По окончании просмотра шаблона закройте его окно и аналогичным образом

просмотрите все предопределенные шаблоны включаемых проектов **CSV_Include** и **Tab_Style_Include**.

Проект **Tab_Style_Include** отличается от включаемого проекта **CSV_Include** тем, что является предварительно сконфигурированным проектом, который инсталлируется со SCADA-системой Vijeo Citect версии 7.30. Он предназначен для сокращения времени, необходимого для конфигурирования нового проекта, и содержит предопределенные шаблоны и страницы, выполненные в стиле операционной системы Microsoft Windows XP.

При создании нового проекта проект **Tab_Style_Include** автоматически входит в новый проект как включаемый проект (проект **CSV_Include**, при необходимости, нужно включить вручную). Это означает, что все шаблоны и другие возможности включаемого проекта **Tab_Style_Include** доступны в новом проекте при создании графических страниц. Наряду с шаблоном **Normal** стандартной графики для создания технологических графических страниц, в проект **Tab_Style_Include** включены предопределенные страницы трендов и тревог, страница средств администрирования, страницы для представления текстовых файлов, а также разнообразные всплывающие окна. Все они оснащены обычными меню и панелями инструментов навигации, которые обеспечивают согласованную функциональность и внешний вид в пределах всего проекта. Включаемый проект поддерживает и мультимониторное отображение, что позволяет одновременно отображать несколько графических страниц на нескольких мониторах компьютера.

16.2. Создание собственных шаблонов

В большинстве проектов создаются шаблоны, разработанные специально для конкретного участка. Собственный шаблон, создаваемый далее, частично будет основываться на стандартных шаблонах.

Создать новый шаблон можно несколькими способами. Во-первых, в среде приложения **Проводник Citect** можно в поле **Список проектов** выбрать проект, последовательно раскрыть проект и компонент **Графика**, выбрать компонент **Шаблоны** и в поле **Содержимое Шаблоны** "кликнуть" дважды левой кнопкой мыши по элементу **Создать новый шаблон**. Во-вторых, в среде приложения **Построитель графики Citect** можно нажать кнопку **Новый** на панели инструментов или, в-третьих, выполнить команду **Файл | Новый...**. В появившемся окне **Новый** нажмите кнопку **Шаблон**.

Упражнение 16.2. Создание собственного шаблона. Проект TmpltAndMenu

Сохраните проект **Training22** под именем **TmpltAndMenu**, восстановите проект **TmpltAndMenu** и, в дальнейшем, работайте с этим проектом.

Создайте новый шаблон **MyNormal** для страниц вашего проекта **TmpltAndMenu** (рис. 16.1) и нажмите кнопку **OK**. Командой **Файл | Сохранить как…** сохраните новый шаблон под именем **MyNormal** в проекте **TmpltAndMenu** (рис. 16.2) и

нажмите кнопку **OK**. В появившемся окне (рис. 16.3) нажмите кнопку **Да**, а в окне, показанном на рис. 16.4, задайте имя нового стиля **MyTraining**, для сохранения нового стиля нажмите кнопку **OK**, а для сохранения нового шаблона еще раз нажмите кнопку **OK**. Вы увидите предупреждающее сообщение (рис. 16.5), нажмите кнопку **OK.**

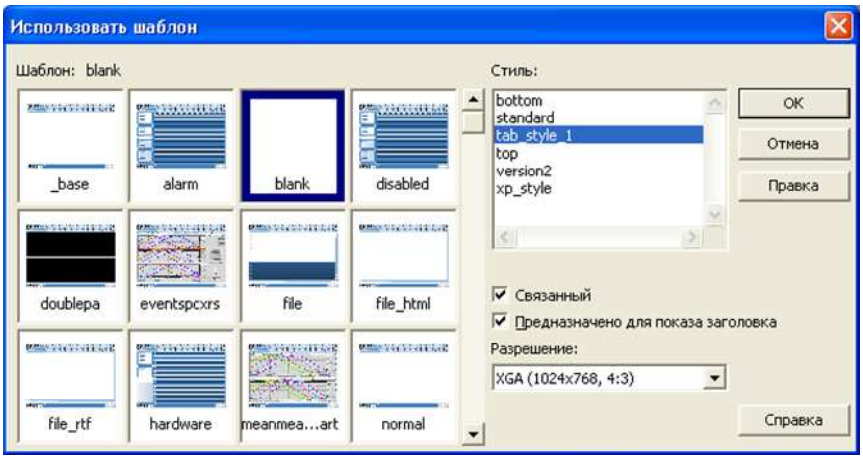


Рис. 16.1. Свойства нового шаблона

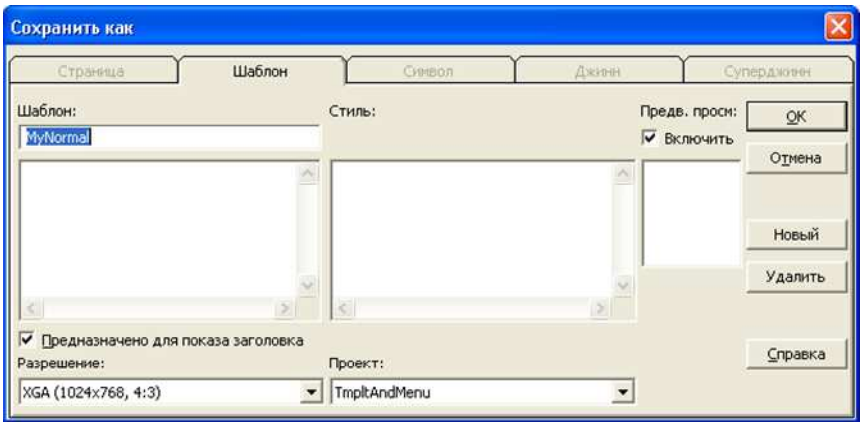


Рис. 16.2. Сохранение нового шаблона

Командой **Файл | Свойства** вызовите окно свойств шаблона, в поле **Цвет фона** задайте белый цвет (рис. 16.6) и нажмите кнопку **OK**. Сохраните проект под именем **TmpltAndMenu**.

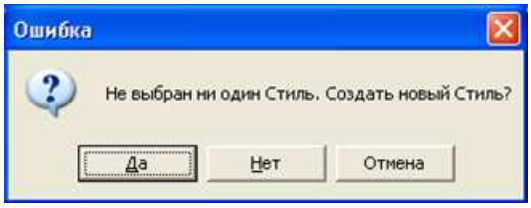


Рис. 16.3. Сообщение о необходимости создания нового стиля шаблона

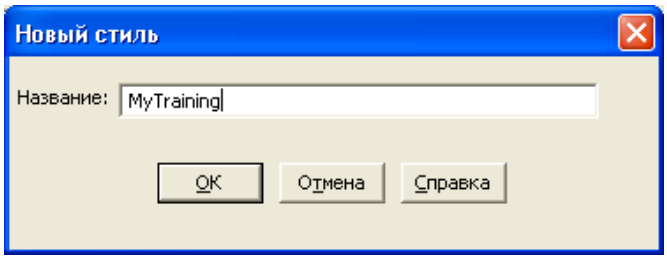


Рис. 16.4. Задание имени нового стиля шаблона

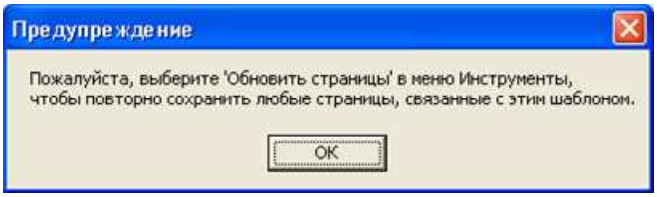


Рис. 16.5. Предупреждение о необходимости обновления страниц, связанных с данным шаблоном

16.3. Настройка собственных шаблонов

Стандартные шаблоны имеют множество свойств, которые вы можете использовать в своих шаблонах. Стандартные шаблоны во включенном проекте обеспечивают вас примерами, которые вы можете использовать в шаблонах своих собственных проектов. Вы можете копировать данные свойства из поставляемых шаблонов в свои собственные разработки.

Упражнение 16.3. Настройка созданного шаблона MyNormal. Создание строки заголовка

Добавьте строку заголовка к вашему шаблону — закрашенный прямоугольник в верхней части страницы созданного шаблона.

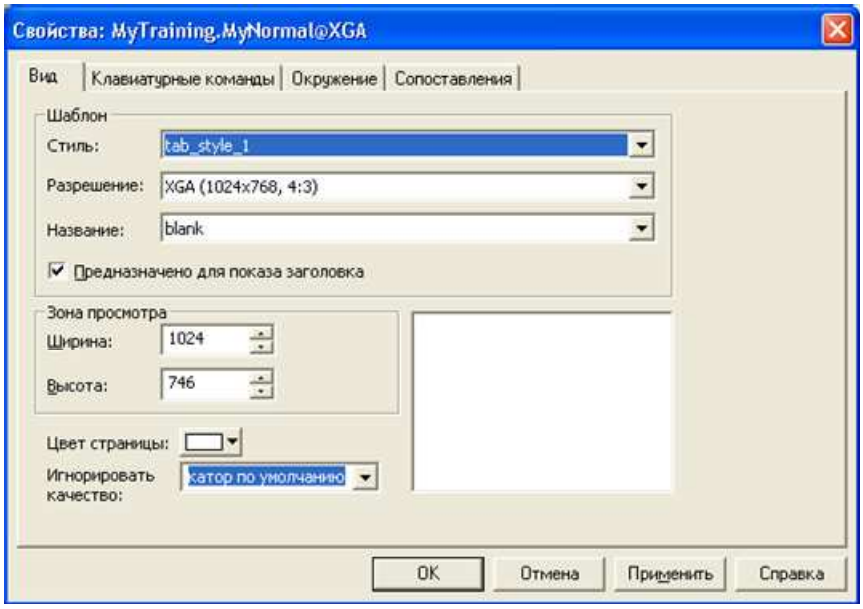


Рис. 16.6. Задание цвета фона созданного шаблона

Добавьте в левую часть строки заголовка текстовый графический объект **Text** для отображения имени графической страницы со свойствами, представленными на рис. 16.7 (см. также справку по функции `PageInfo()`).

Добавьте в правую часть строки заголовка рисованную кнопку, используемую для завершения работы проекта. С этой целью просто скопируйте аналогичную кнопку (белый крестик на красном фоне в строке заголовка шаблона **Normal** стиля **xp_style** из включаемого проекта **CSV_Include)** со свойствами, представленными на рис. 16.8 (см. также справку по функции `Shutdown()`). В результате строка заголовка созданного пользовательского шаблона приобретает вид, представленный на рис. 16.9.

Сохраните пользовательский шаблон под прежним именем. Выполните компиляцию проекта и сохраните его под прежним именем.

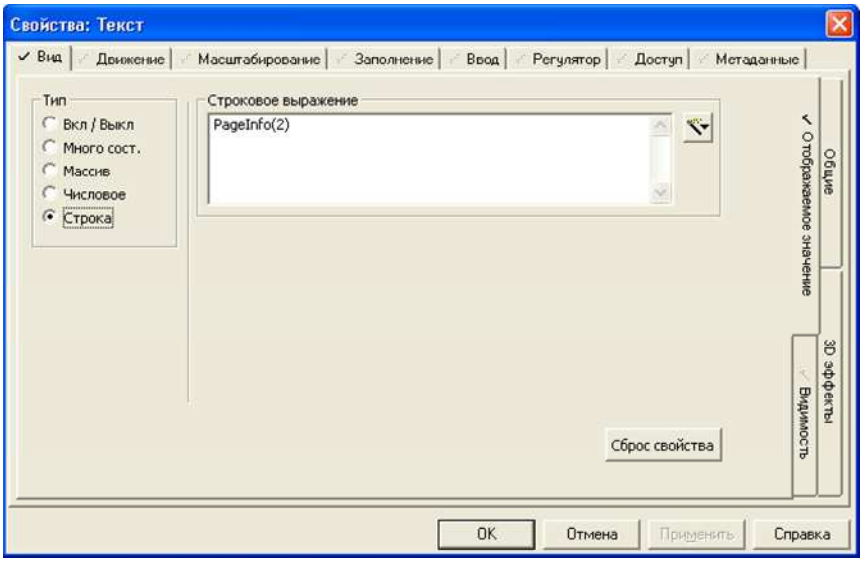


Рис. 16.7. Свойства графического объекта **Текст**, используемого в строке заголовка созданного шаблона

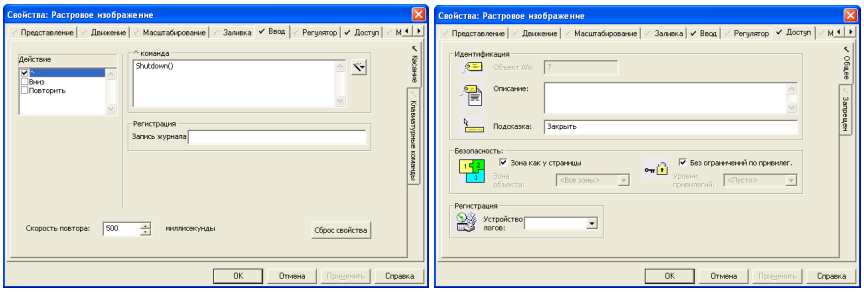


Рис. 16.8. Свойства рисованной кнопки, используемой в строке заголовка созданного шаблона

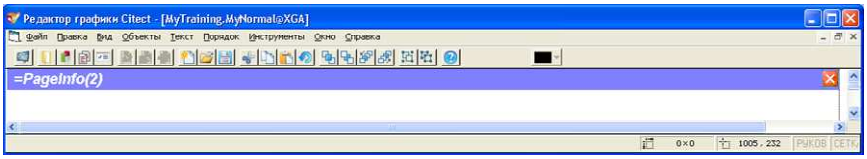


Рис. 16.9. Строка заголовка пользовательского шаблона

Упражнение 16.4. Настройка созданного шаблона MyNormal. Создание панелей инструментов

В среде приложения **Построитель графики Citect** нажмите кнопку **Открыть** на панели инструментов и откройте шаблон **Normal** стиля **xp_style** из включаемого проекта **CSV_Include**. Фрагменты открытого шаблона представлены на рис. 16.10.

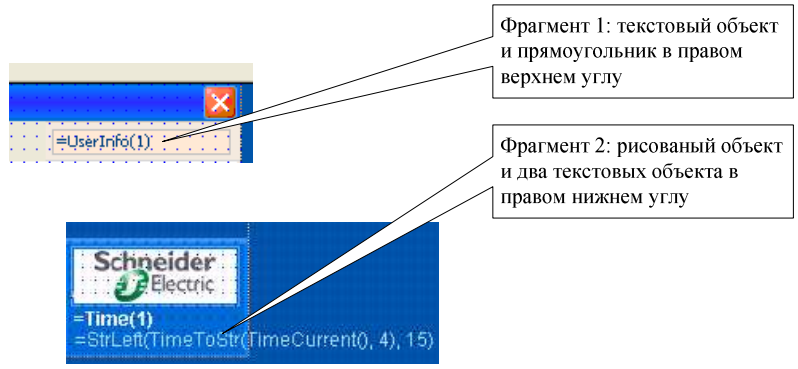


Рис. 16.10. Фрагменты шаблона **Normal** стиля **xp_style** из включаемого проекта **CSV_Include**

Скопируйте фрагмент 1 (см. рис. 16.10) в пользовательский шаблон **MyNormal** проекта **TmpltAndMenu**, расположив его аналогично расположению фрагмента 1 на рис. 16.10. Настройте скопированные графические объекты в соответствии с рис. 16.11.

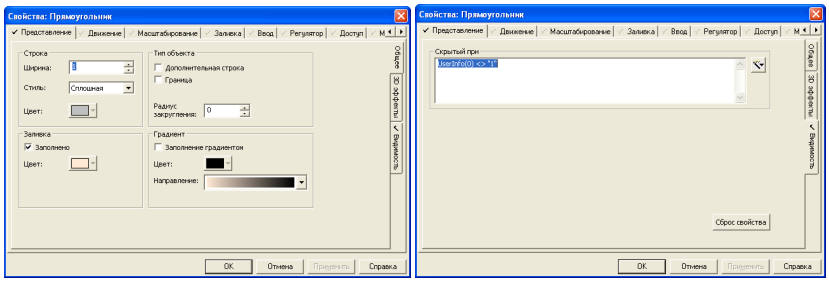


Рис. 16.11а. Конфигурирование скопированных графических объектов (фрагмент 1, объект **Прямоугольник**)

Скопируйте фрагмент 2 (см. рис. 16.10) в пользовательский шаблон **MyNormal** проекта **TmpltAndMenu**, расположив его аналогично расположению фрагмента 2 на рис. 16.10. Настройте скопированные графические объекты в соответствии с рис. 16.12. Сохраните пользовательский шаблон **MyNormal**.

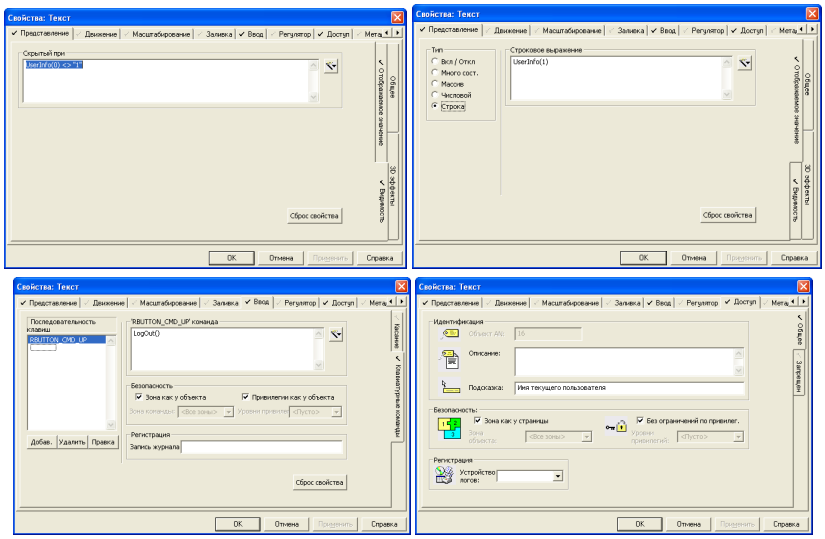


Рис. 16.11b. Конфигурирование скопированных графических объектов (фрагмент 1, объект **Текст**)

16.4. Точки вывода анимации (AN)

Каждая точка графической страницы, на которой отображается графический объект, называется точкой вывода анимации (AN — Animation Number). При добавлении в графическую страницу графического объекта (текста, символа, трубы и т. п.) Vijeo Citect автоматически присваивает номер точке вывода анимации (см. свойство **Доступ (Общие)** графического объекта, поле **Объект AN:**). Количество объектов, которые можно использовать, ограничивается только быстродействием компьютера, хотя это редко является проблемой. Хорошим стилем является использование в графической странице не более 3000 графических объектов (точек вывода анимации).

SCADA-система Vijeo Citect всегда использует две первых точки вывода анимации для отображения системной информации, т. е. эти точки являются зарезервированными. В некоторых графических страницах зарезервированными являются и некоторые другие точки вывода анимации с начальными номерами.

Когда вы создаете свои шаблоны, рекомендуется резервировать для каждого шаблона некоторое количество точек анимации. Это даст возможность добавлять в свои шаблоны в процессе их модификации новые графические объекты. Когда объект будет добавляться к графической странице, основанной на вашем шаблоне, он будет использовать следующий доступный номер точки вывода анимации.

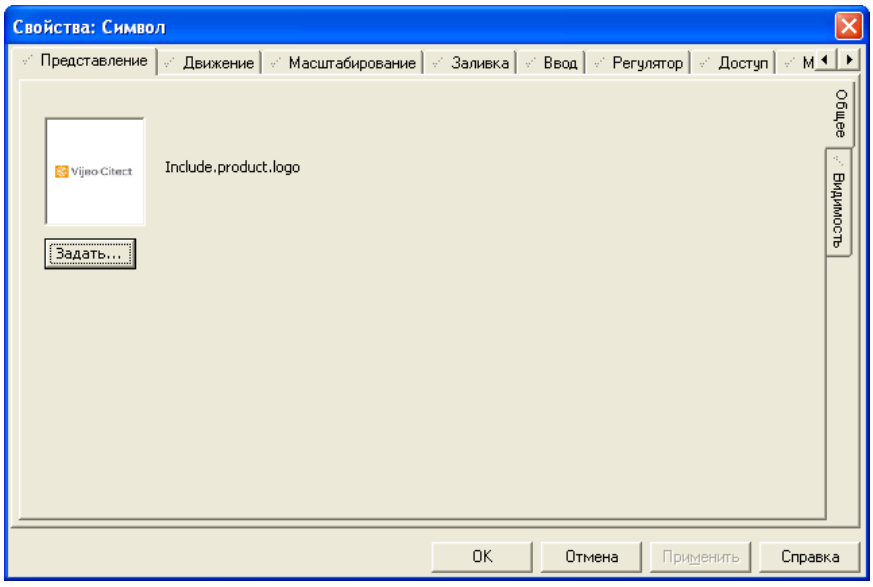


Рис. 16.12a. Конфигурирование графических объектов (фрагмент 2, объект **Образ**)

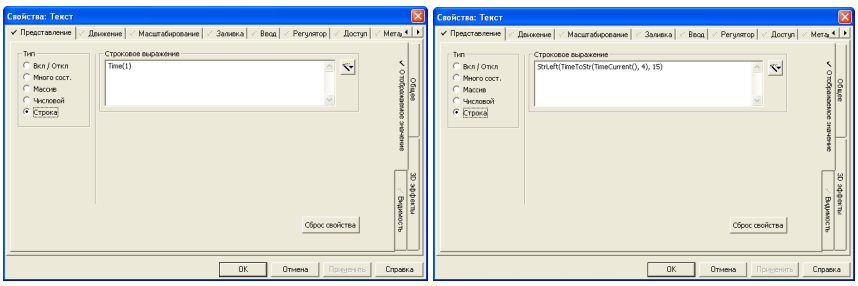


Рис. 16.12b. Конфигурирование графических объектов (фрагмент 2, объекты **Текст**)

Если на одном из последующих этапов вы решите внести изменения в свой шаблон, то любой включаемый в шаблон графический объект будет использовать следующую точку вывода анимации. При этом два номера точек вывода анимации (в шаблоне и графической странице) будут конфликтовать. Для устранения конфликта резервируйте номера точек вывода анимации для каждого своего шаблона. В случае, когда на одном из последующих этапов понадобится добавить в шаблон графический объект, достаточно предварительно удалить один из номеров точек вывода анимации, зарезервированных для данного шаблона.

Упражнение 16.5. Настройка созданного шаблона MyNormal. Использование зарезервированных точек вывода анимации

В среде приложения **Построитель графики Citect** добавьте в пользовательский шаблон **MyNormal** проекта **TmpltAndMenu** панель инструментов, имеющую вид прямоугольника, расположите ее в верхней части шаблона по ширине окна под графическим объектом **=UserInfo(1)** и настройте ее в соответствии с рис. 16.13 (левый верхний угол панели инструментов имеет координаты 0 – 50, а правый нижний — координаты 1024 – 75).

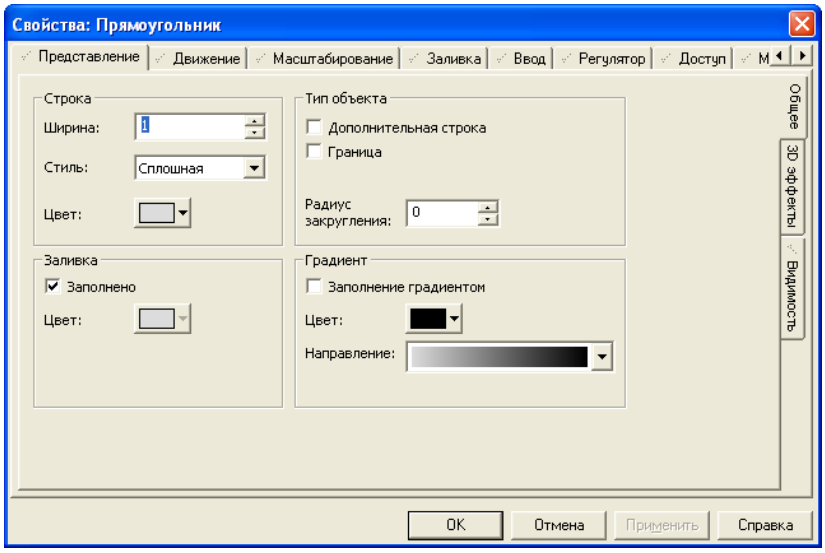


Рис. 16.13. Конфигурирование панели инструментов

Точки вывода анимации **AN1** и **AN2** расположите в пределах панели инструментов соответственно в точках с координатами (340, 62) и (510, 62) как это показано на рис. 16.14. С этой целью откройте шаблон **tab_style_1.blank** проекта **Tab_Style_Include**, выполните команду **Средства | Перейти к объекту…,** в появившемся окне **Перейти к объекту** выберите **Точка анимации AN1** и нажмите кнопку **OK**. В появившемся окне **Точка анимации** в поле **Координата по X** введите 340, в поле **Координата по Y** введите 62 и нажмите кнопку **OK**. Аналогично задайте координаты для **AN2**. С помощью кнопки **Сохранить** на панели инструментов сохраните измененный шаблон **Include – Standard.blank@-XGA**, в появившемся окне нажмите кнопку **OK** и закройте окно шаблона. Чтобы точки вывода анимации в шаблоне **MyNormal** проекта **TmpltAndMenu**, скрытые под панелью инструментов, стали видны, выберите прямоугольник панели инструментов и выполните команду

Порядок | На задний план. Для улучшения видимости точек вывода анимации добавьте позади них темные прямоугольники (см. рис. 16.14). Точка вывода анимации 1 отображает вводимые с клавиатуры символы, а точка вывода анимации 2 отображает **Prompt**(подсказку).

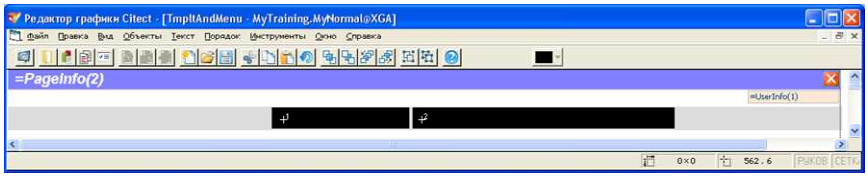


Рис. 16.14. Точки вывода анимации в панели инструментов

Зарезервируйте для пользовательского шаблона **MyNormal** проекта **TmpltAndMenu** двадцать точек вывода анимации. Для этого выполните команду **Средства | Настройки...** и в появившемся окне **Параметры** выберите опцию **Инструменты версий 3.xx/ 4.xx** и нажмите кнопку **OK**. Появится еще одна панель объектов, представленная на рис. 16.15, обеспечивающая работу с точками вывода анимации (инструменты панели снабжены всплывающими подсказками).

Рис. 16.15. Дополнительное дерево объектов, поддерживающее работу с точками вывода анимации

В этой панели объектов выберите инструмент **Точка анимации** и с его помощью зарезервируйте для пользовательского шаблона еще 20 точек вывода анимации. Расположите точки вывода анимации за одной из внешних границ видимой части страницы (рис. 16.16).

Сохраните шаблон **MyNormal** проекта **TmpltAndMenu** под прежним именем.

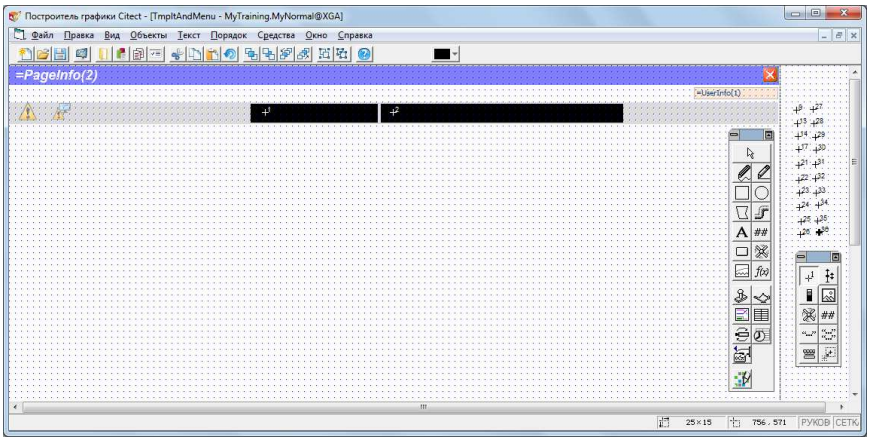


Рис. 16.16. Вид пользовательского шаблона **MyNormal**

16.5. Отображение в шаблоне сигналов тревог

Вы можете принять решение о выводе самых последних сигналов тревог на всех страницах. Функция `AlarmDsp()` отображает список сигналов тревог, начиная с определенной точки вывода анимации и до последующих точек.

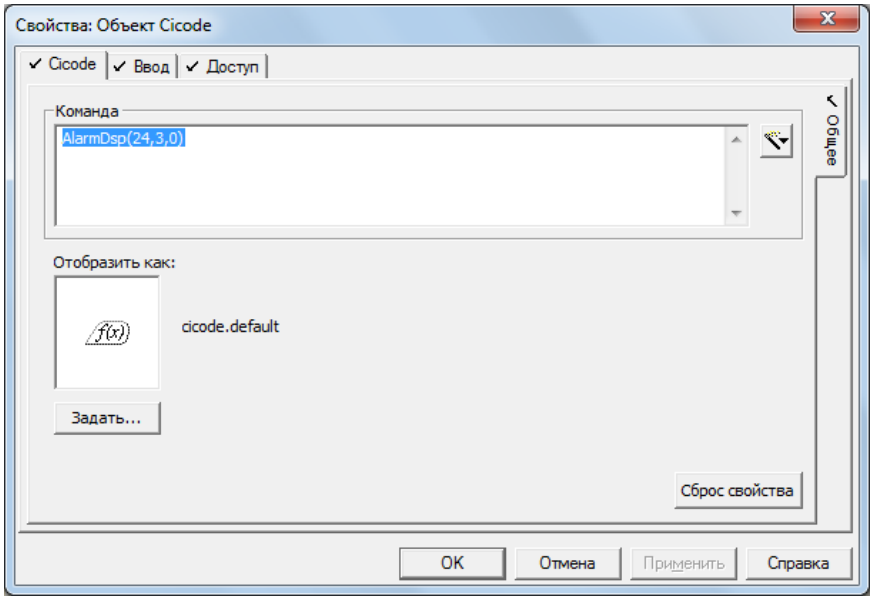
С помощью расположенных друг под другом объектов **Cicode** вы можете задать тип и количество сигналов тревог (см. подробнее в справке по функции `AlarmDsp()`). Перед вызовом данной функции для шаблона следует добавить к шаблону нужное количество точек вывода анимации из числа зарезервированных по числу одновременно отображаемых сигналов тревог. Используемые точки вывода анимации должны иметь последовательные номера. Вы можете проверить номер точки вывода анимации в поле **Объект ТА** свойства **Доступ (Общие)** объекта **Cicode**.

Упражнение 16.6. Настройка созданного шаблона MyNormal. Отображение сигналов тревог

Освободите четыре точки вывода анимации с последовательными номерами из числа зарезервированных ранее. Для этого выберите первую точку и выполните команду **Вырезать** ее контекстного меню. Аналогично освободите еще три точки анимации. Нарисуйте в нижней части пользовательского шаблона **MyNormal** проекта **TmpltAndMenu** панель инструментов, имеющую вид черного прямоугольника. Расположите в левой части прямоугольника друг под другом три объекта **Cicode** и настройте верхний объект в соответствии с рис. 16.17.

Аналогичным образом настройте остальные объекты Cicode и убедитесь, что точки вывода анимации имеют последовательные номера, которые используются в качестве

первого аргумента в вызове функции `AlarmDsp()`. Второй и третий аргументы в вызове функции `AlarmDsp()` означают, что в данном случае будут отображаться до трех последних активных сигналов тревог.



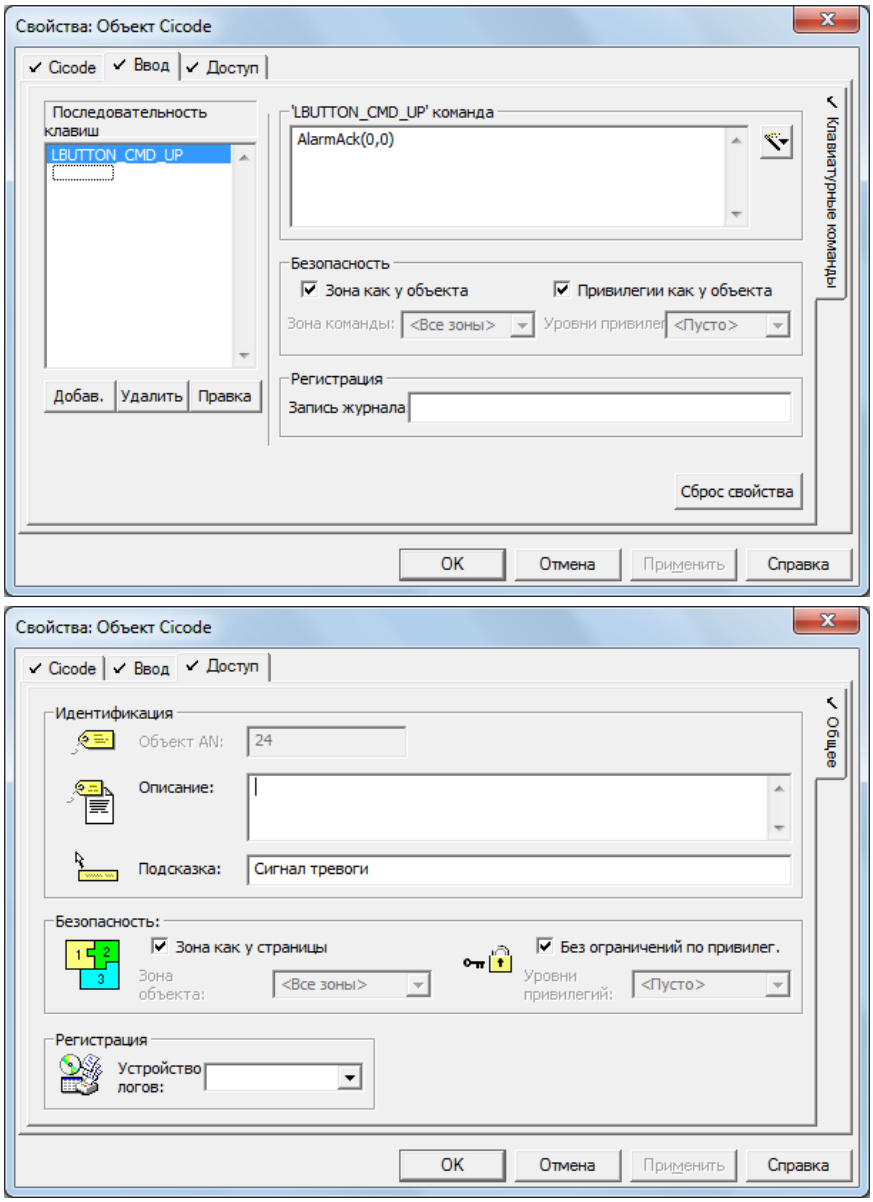


Рис. 16.17. Свойства первого объекта **Cicode**

Добавьте два набора символов в левый верхний угол панели инструментов, в которую вы ранее добавили точки вывода анимации **AN1** и **AN2**. Используйте параметры символов, представленные на рис. 16.18 и 16.19.

> ***Примечание***
> Функции `PageAlarm()`,`PageHardware()` и `PageSummary()` по умолчанию открывают страницы с именами **Alarm**, **Hardware** и **Summary** соответственно.

Сохраните пользовательский шаблон **MyNormal**. Текущий вид шаблона представлен на рис. 16.20.

16.6. Навигация (перемещение) в шаблоне

Большие проекты Vijeo Citect часто содержат много графических страниц. Для перехода пользователя к другим страницам часто используются боковые панели инструментов и меню.

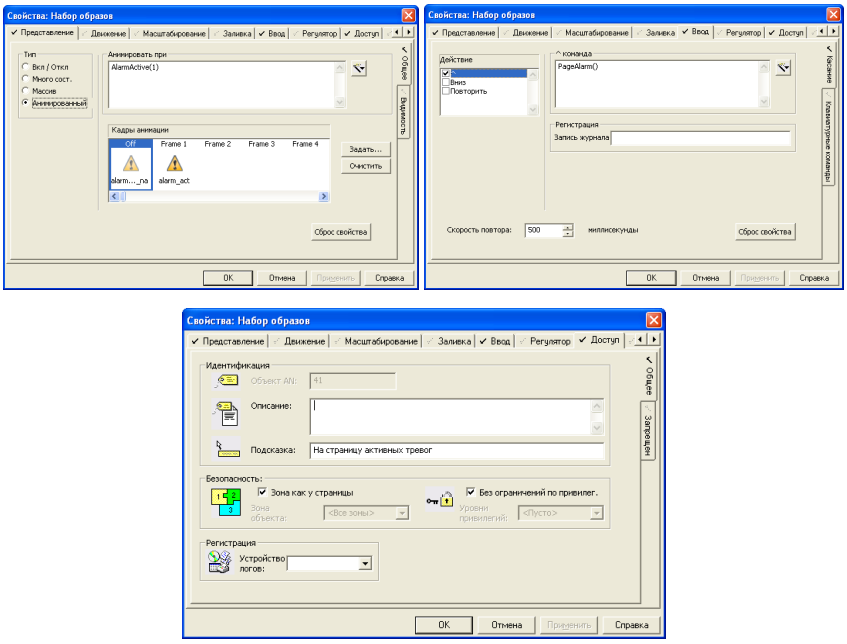


Рис. 16.18. Свойства набора символов для отображения активных тревог

Упражнение 16.7. Настройка созданного шаблона MyNormal. Боковые панели инструментов навигации

Разместите в верхней левой части пользовательского шаблона **MyNormal** проекта **TmpltAndMenu** боковую панель инструментов навигации, содержащую три кнопки, графический объект **Текст** с поясняющим текстом и прямоугольник для группирования (рис. 16.21). Настройте кнопки навигации в соответствии с рис. 16.22 — 16.24. Окончательный вид пользовательского шаблона иллюстрирует рис. 16.25. Сохраните пользовательский шаблон **MyNormal**.

16.7. Использование пользовательского шаблона

На основе созданного пользовательского шаблона **MyNormal** можно создавать графические страницы, обладающие общим набором свойств, заложенным в шаблон **MyNormal**.

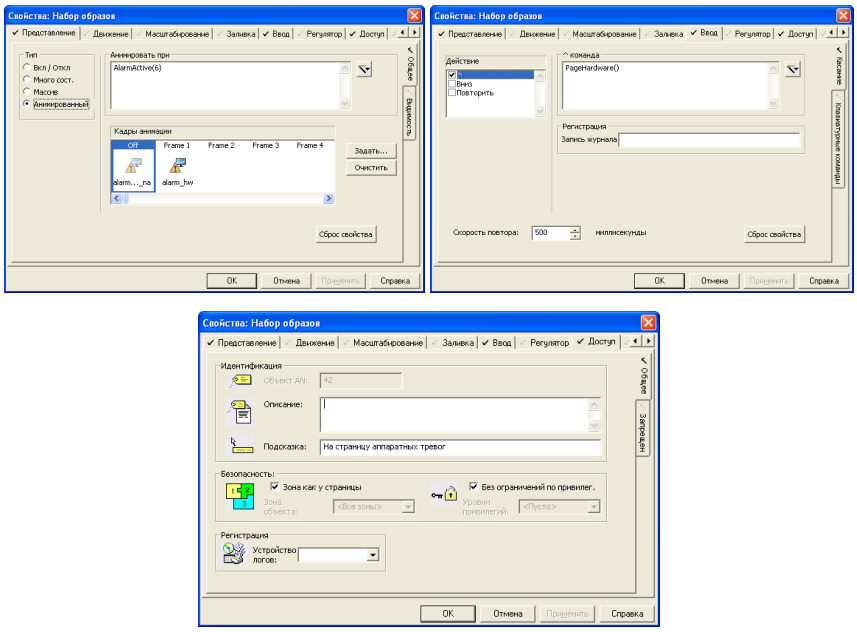


Рис. 16.19. Свойства набора символов для отображения аппаратных тревог

Упражнение 16.8. Использование созданного шаблона MyNormal

Запустите приложение **Проводник Citect**, выберите проект **TmpltAndMenu** и перейдите в среду приложения **Построитель графики Citect**.

Создайте новую графическую страницу, основанную на созданном пользовательском шаблоне. С этой целью нажмите кнопку **Новый** на панели инструментов, в появившемся окне **Новый** нажмите кнопку **Страница** и появится окно **Использовать шаблон**. Для выбора созданного пользовательского шаблона в поле **Стиль:** выберите **mytraining**, в поле **Шаблон:** выберите **MyNormal**, отметьте поле **Связанный** и поле **Предназначено для показа заголовка**, в поле **Разрешение:** выберите **XGA** и нажмите кнопку **OK**.

Сохраните созданную страницу под именем **MyPage**. Для этого выполните команду **Файл | Сохранить как…**, в поле **Страница:** появившегося окна **Сохранить как** укажите имя **MyPage** и нажмите кнопку **OK**.

Для тестирования пользовательского шаблона **MyNormal** выполните компиляцию проекта, запустите **Мастер настройки компьютера** в режиме выборочной настройки (задайте в качестве стартовой страницы **Home**), запустите проект, зарегистрируйтесь как привилегированный пользователь и перейдите на страницу **MyPage**.

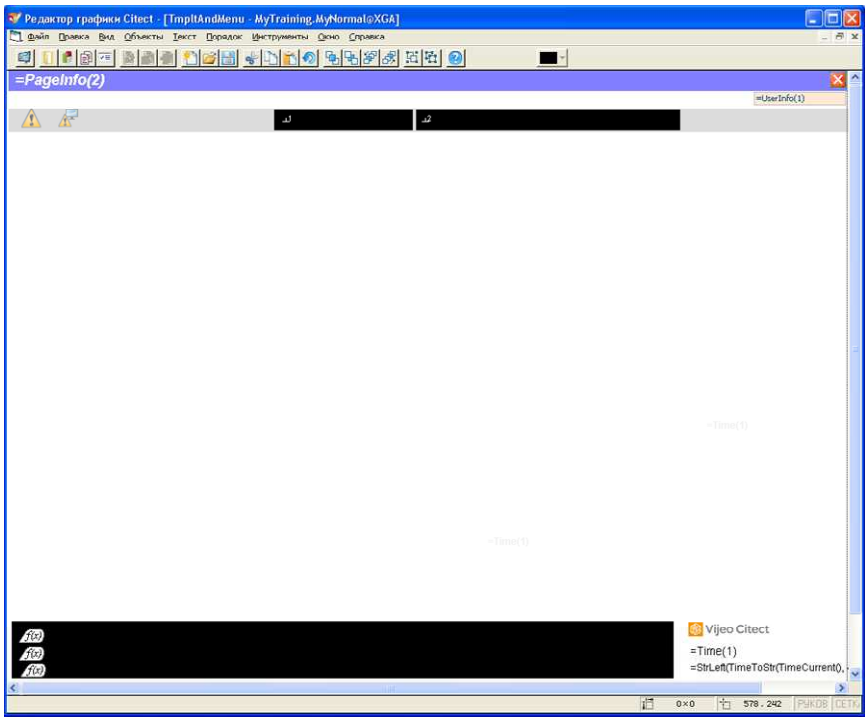


Рис. 16.20. Вид пользовательского шаблона **MyNormal**

Обратите внимание на следующие фрагменты окна **MyPage**. В левой части строки заголовка отображается имя графической страницы. Если поместить указатель мыши над кнопкой, расположенной в строке заголовка справа, то появится всплывающая подсказка **Закрыть**. Справа под строкой заголовка отображается имя зарегистрированного пользователя (в данном случае это привилегированный пользователь). В правой части панели инструментов, расположенной сверху, в точке вывода анимации **AN2** отображается информация о дискретных тревогах (информация задается аргументом в вызове функции `Prompt()`). В левой части панели инструментов, расположенной снизу, отображается информация об активных тревогах. И, наконец, в правой части этой же панели инструментов отображаются логотип, текущие значения времени и даты.

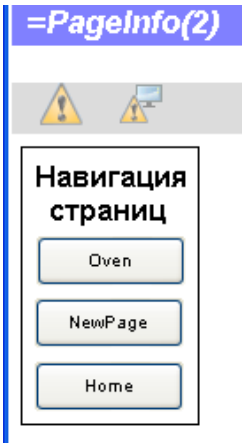


Рис. 16.21. Боковая панель инструментов навигации пользовательского шаблона **MyNormal**

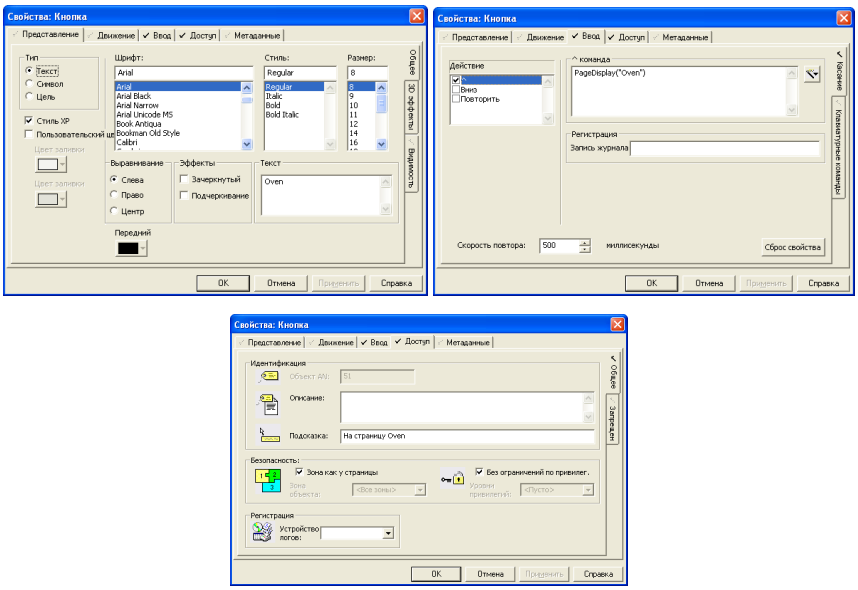


Рис. 16.22. Свойства кнопки навигации

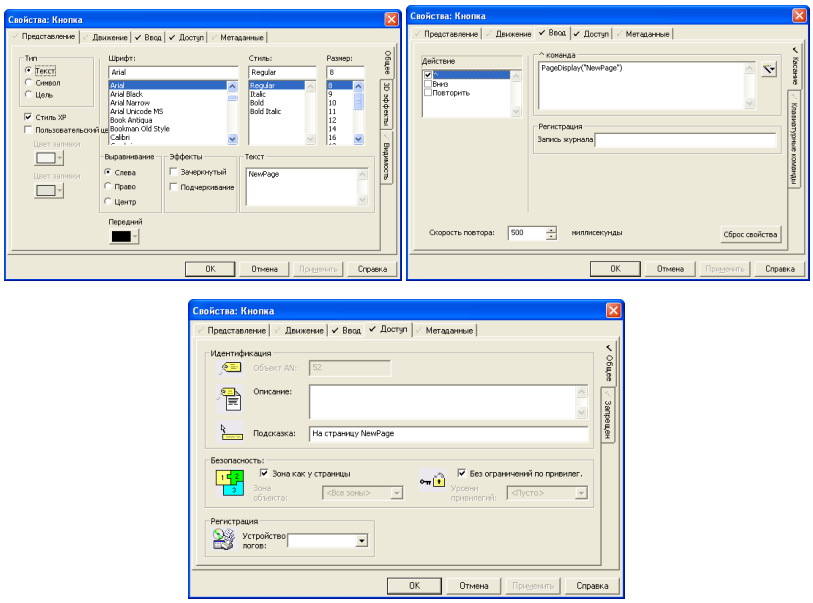


Рис. 16.23. Свойства кнопки навигации

Протестируйте возможности работы с сигналами тревог. Для этого поместите, например, курсор мыши на верхний элемент списка сигналов активных тревог, расположенного на нижней панели инструментов. Появится всплывающая подсказка **Сигнал тревоги**. Для квитирования (подтверждения) выбранного сигнала тревоги "кликните" левой кнопкой мыши и цвет текста станет менее ярким, а подтвержденный сигнал окажется внизу списка. Обратите внимание, что квитирование сигнала тревоги доступно только привилегированному пользователю. Переместите указатель на символ, расположенный в левой части верхней панели инструментов. Появится всплывающая подсказка **На страницу активных тревог**. Нажмите этот символ и произойдет переход на страницу активных тревог. После просмотра страницы снова вернитесь на страницу **MyPage**. Переместите указатель на второй слева символ, расположенный в верхней панели инструментов. Появится всплывающая подсказка **На страницу аппаратных тревог**. Нажмите этот символ и произойдет переход на страницу аппаратных тревог. После просмотра страницы снова вернитесь на страницу **MyPage**. Выберите объект, расположенный справа под строкой заголовка, и "кликните" по нему правой кнопкой мыши. Произойдет разрегистрация текущего пользователя и объект перестанет отображаться.

Протестируйте боковую панель инструментов. Для этого поместите, например, курсор мыши на кнопку **Oven**, расположенную на боковой панели инструментов. Появится всплывающая подсказка **Переход на страницу Oven**. После просмотра

страницы снова вернитесь на страницу **MyPage**. Аналогичным образом протестируйте действие кнопок **NewPage** и **Home**.

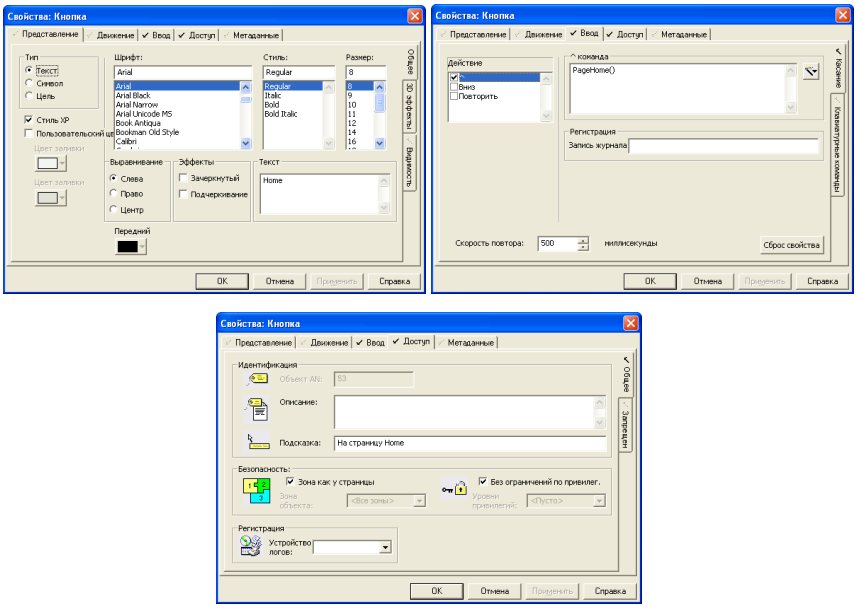


Рис. 16.24. Свойства кнопки навигации

В заключение протестируйте использование точки вывода анимации **AN1**. С этой целью нажмите какую-либо произвольную клавишу компьютера, например, клавишу **g**, действие которой в проекте не определено, и пронаблюдайте за информацией, выводимой в области **AN1** и **AN2**. Далее нажмите клавишу **Home** компьютера, действие которой в проекте определено, и пронаблюдайте за информацией, выводимой в области **AN1** и **AN2**. В результате произойдет переход на страницу **Home**.

Завершите работу проекта и сохраните его.

16.8. Пользовательские меню

Команды меню позволяют вызвать функцию Cicode или выполнить переход к требуемой графической странице. Это дает пользователю возможность работать с проектом выбирая и выполняя команды меню в стиле Windows.

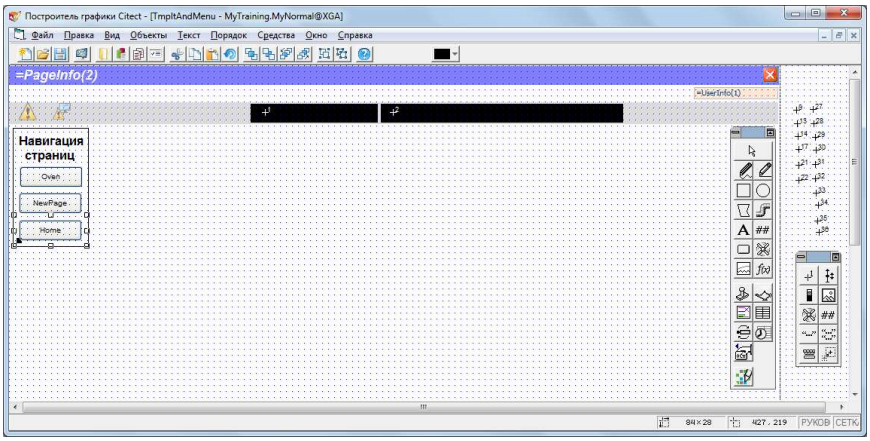


Рис. 16.25. Окончательный вид пользовательского шаблона **MyNormal**

Функция `DspPopupMenu()` создает раскрывающееся меню, содержащее команды. Многочисленные вызовы этой функции позволяют создавать и иерархические меню с командами. Команды меню могут отображаться как нормальные, отмеченные или отключенные и могут разделяться сепараторами. Рассмотрим пример создания и работы с раскрывающимся меню (листинг 16.1).

Листинг 16.1

```
' Первый вызов функции создает команды меню:
'   команда Oven будет отображена в нормальном виде и отделена от
'   последующих команд сепаратором (см. два подряд идущих разделителя
'   ,,);
'   команда NewPage будет отображена как отключенная (см. префикс !);
'   команда Home будет отображена как отмеченная (см. префикс ~)
DspPopupMenu(0,"Oven,,!NewPage,~Home");
' Второй вызов функции отображает меню на экране в позиции (0,50) и
'   возвращает номер выбранной команды:
'   1, если выбрана команда Oven;
'   0, если выбрана команда NewPage (команда отключена, иначе будет
'      возвращено значение 2);
'   3, если выбрана команда Home;
RetVal= DspPopupMenu(-1,"",0,50);    ' RetVal – локальная переменная
If RetVal =1 Then
    PageDisplay("Oven");             ' Переход на страницу Oven
End
If RetVal =2 Then
```

```
    PageDisplay("NewPage");          ' Переход на страницу NewPage
End
If RetVal =3 Then
    PageHome();                      ' Переход на страницу Home
End
```

Упражнение 16.9. Создание и применение пользовательского меню

Запустите приложение **Проводник Citect**, выберите проект **TmpltAndMenu** и перейдите в среду приложения **Потсроитель графики Citect**.

Откройте графическую страницу **MyPage** и разместите в ней кнопку в соответствии с рис. 16.26.

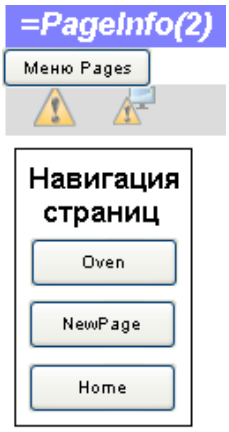


Рис. 16.26. Размещение кнопки раскрывающегося меню в графической странице **MyPage**

Задайте свойства кнопки в соответствии с рис. 16.27. Полный текст, введенный в поле **Up command**, приведен ранее в листинге 16.1.

Сохраните графическую страницу **MyPage**, выполните компиляцию, запустите проект и перейдите в страницу **MyPage**.

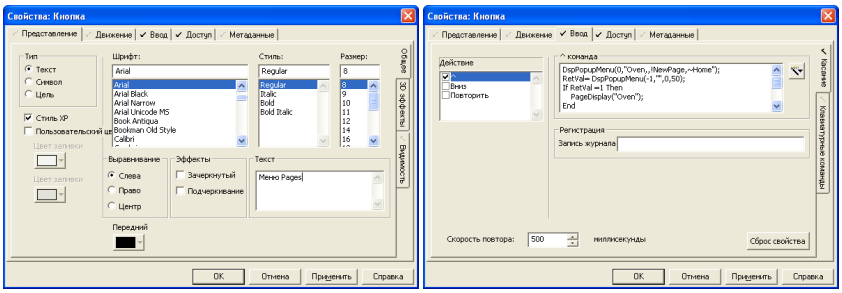


Рис. 16.27. Свойства кнопки раскрывающегося меню в графической странице **MyPage**

Чтобы раскрыть меню нажмите кнопку **Меню Pages** (рис. 16.28). Команда **Oven** отображается в нормальном виде и отделена от остальных команд меню сепаратором, команда **NewPage** недоступна, а команда **Home** отмечена.

Рис. 16.28. Вид раскрывающегося меню в графической странице **MyPage**

Протестируйте команды меню. Для этого выполните команду **Oven** меню **Меню Pages** и будет выполнен переход на графическую страницу **Oven**. Вернитесь на страницу **MyPage** и попытайтесь выполнить команду **NewPage**. Ничего не произойдет, т. к. эта команда недоступна. Выполните команду **Home** и будет выполнен переход на графическую страницу **Home**.

Завершите работу проекта и сохраните проект.

Глава 17. Всплывающие окна и суперджины. ActiveX элементы

Всплывающие окна и суперджины являются динамическими страницами, которые можно использовать для передачи информации, когда страница отображается в процессе работы проекта (приложения). Их обычно используют для отображения всплывающих органов управления процессами или отдельными узлами промышленного оборудования. *Одну и ту же всплывающую страницу можно многократно использовать с различными наборами тегов для суперджинов.* Напомним, что в случае всплывающих страниц с джинами, для каждого джина требуется своя всплывающая страница. Если специалист по интеграции работает с множеством различных систем, то использование суперджинов позволяет сохранить их в файле с расширением **имя_библиотеки.ctm** и легко скопировать в другие проекты.

17.1. Создание всплывающей графической страницы и функции работы с суперджинами

Для создания всплывающей графической страницы сначала создайте и сохраните новую пустую страницу. Эта страница может быть связана с джином (суперджином) или вызываться непосредственно из другой графической страницы. В любом случае, для открытия всплывающей страницы во время работы проекта необходим вызов одной из функций работы с суперджинами.

> ***Замечание***
> Справка по функциям работы с суперджинами содержится в теме Cicode Programming Reference | Cicode Function Categiries |Super Genie Functions.

Существует множество функций, которые можно использовать для вызова и модификации суперджина. Эти функции можно вызывать из графической страницы или из джина во время работы проекта. Многие из таких функций содержат список имен тегов для передачи их суперджину. Примерами таких функций являются `AssPopUp()` и `AssWin()`.

17.2. Синтаксис суперджина

Имена тегов переменных будут подставлены во время работы проекта на место серии последовательных шаблонов имен со следующим синтаксисом:

```
?type number?
```

Здесь `number` является позицией имени в списке аргументов функции, вызванной для открытия страницы суперджина. Использование ключевого слова `type` в синтаксисе суперджина не является обязательным. Но, если это ключевое слово используется, то

вместо слова `type` следует подставить соответствующий тегу переменной тип данных (`string`, `int`, `real` или `digital`).

Примечание
Тип строковой переменной указывать обязательно.

Упражнение 17.1. Создание всплывающей графической страницы для управления горелкой и краном трубопровода на странице Oven. Проект SuperGenie1

Сохраните проект **TmpltAndMenu** под именем **SuperGenie1**, восстановите проект **SuperGenie1** и, в дальнейшем, работайте с этим проектом.

Для создания нового всплывающего окна, содержащего суперджин, перейдите в среду приложения **Построитель графики Citect** и нажмите кнопку **Новый** на панели инструментов. В появившемся окне **Новый** нажмите кнопку **Страница**, в следующем окне **Использовать шаблон** выберите шаблон **tab_style_1.blank**, отметьте **Связанный** и **Предназначено для показа заголовка**, в окне **Разрешение:** выберите **XGA** и нажмите кнопку **OK**. Задайте белый цвет фона страницы.

Разместите в левом верхнем углу созданной графической страницы три графических объекта так, как это показано на рис. 17.1, и задайте их свойства в соответствии с рис. 17.2 — 17.4.

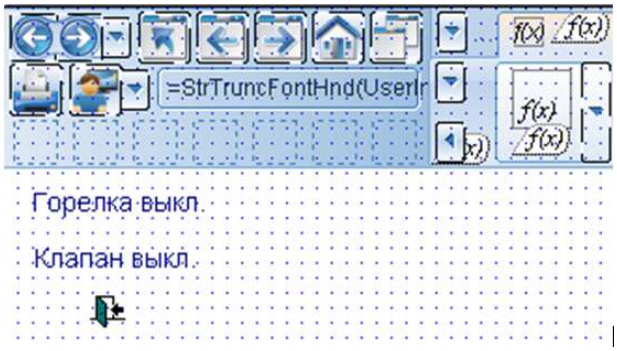


Рис. 17.1. Графические объекты суперджина

Примечание
Созданный суперджин имеет два параметра подстановки (`?digital 1?` и `?digital 2?`), которые используются в нескольких местах суперджина.

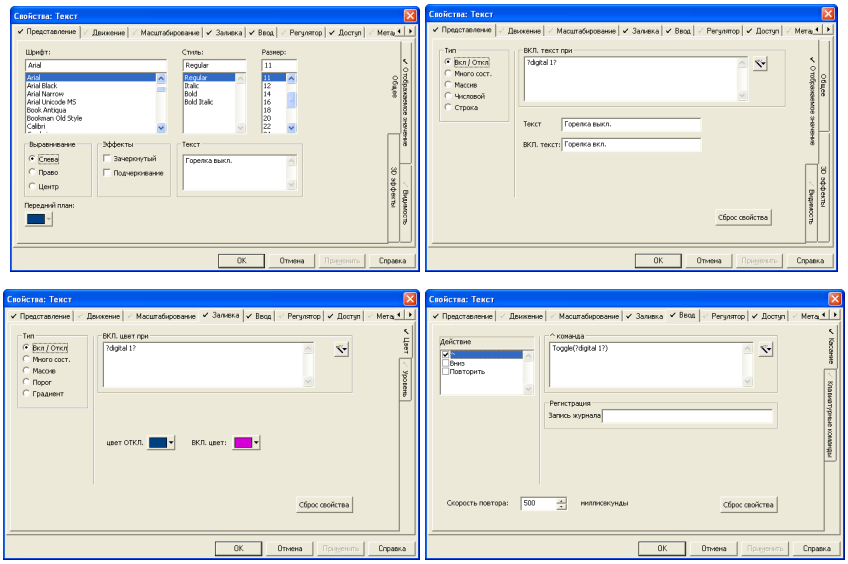


Рис. 17.2. Свойства графического объекта суперджина для управления горелкой

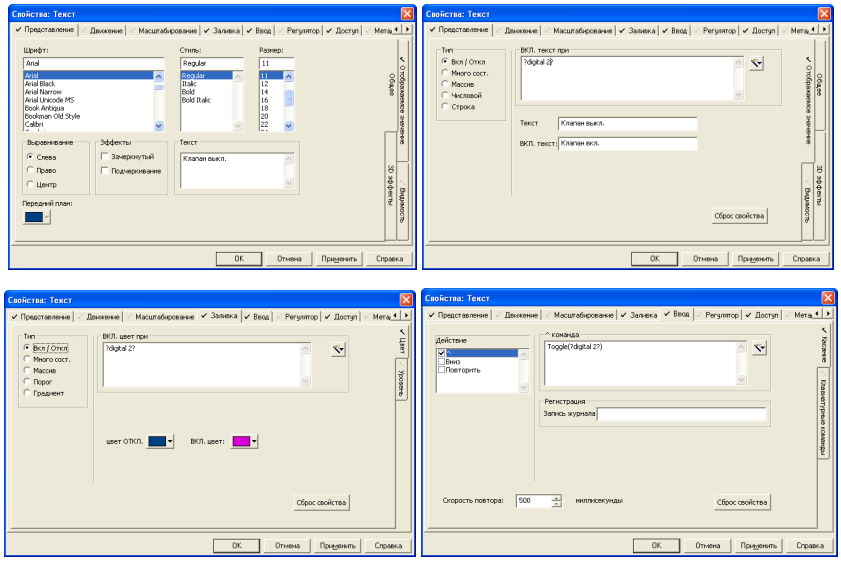


Рис. 17.3. Свойства графического объекта суперджина для управления краном трубопровода

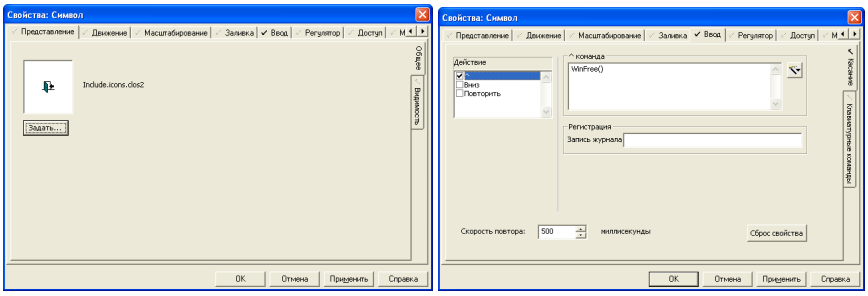


Рис. 17.4. Свойства графического объекта для закрытия всплывающего окна суперджина

Настройте размеры. Расположите курсор в правом нижнем углу под графическими объектами страницы и он будет указывать необходимый размер всплывающей страницы (в строке состояния внизу экрана будут отображаться координаты курсора относительно верхнего левого угла страницы — в нашем случае 130, 100, где 130 является шириной, а 100 — высотой окна). Выполните команду **Файл | Свойства**, в появившемся окне в поле **Ширина:** укажите **130**, в поле **Высота:** укажите **100** и нажмите кнопку **OK**.

Нажмите кнопку **Сохранить** на панели инструментов и сохраните страницу под именем **!BurnerValve**.

> ***Замечание***
> Восклицательный знак в начале имени страницы означает, что эту страницу нельзя будет выбрать из диалога `Select Page` (появляется после вызова функции `PageSelect()`) или в меню **Page** во время работы проекта.

Упражнение 17.2. Вызов из графического объекта всплывающей графической страницы с суперджином

При выбранном проекте **SuperGenie1** в среде приложения **Построитель графики Citect** перейдите на графическую страницу **Oven**.

Добавьте графический объект **Кнопка** и задайте его свойства в соответствии с рис. 17.5. Два последних аргумента в вызове функции `AssWin("!BurnerValve",0,0,1+8+512,"Burner_Stat","Gas_Valve")` являются подстановками для шаблонов имен `?digital 1?` и `?digital 2?` соответственно.

> ***Примечание***
> Имена тегов в вызове функции `AssWin()` указываются в кавычках. Если кавычки опустить, то вместо имени тега функции будет передано значение тега.

Сохраните графическую страницу **Oven**, скомпилируйте и запустите проект **SuperGenie1**. Перейдите на графическую страницу **Oven**.

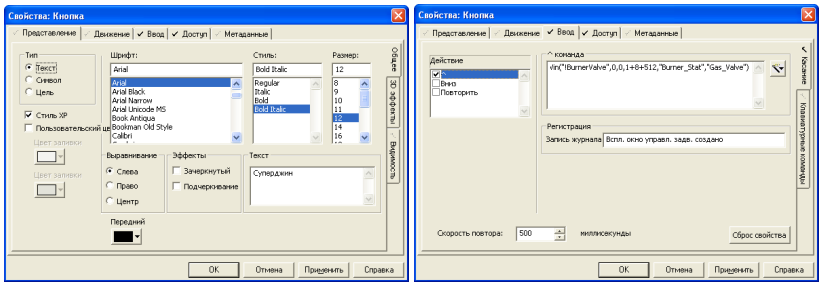


Рис. 17.5. Свойства графического объекта **Кнопка** для вызова всплывающей страницы с суперджином

Для вызова всплывающей страницы с суперджином нажмите кнопку **Суперджин** (рис. 17.6). Протестируйте действие графических объектов суперджина путем последовательного нажатия кнопок **Горелка вкл./выкл.** и **Клапан вкл./выкл**. Для закрытия всплывающего окна с суперджином нажмите нижнюю кнопку суперджина.

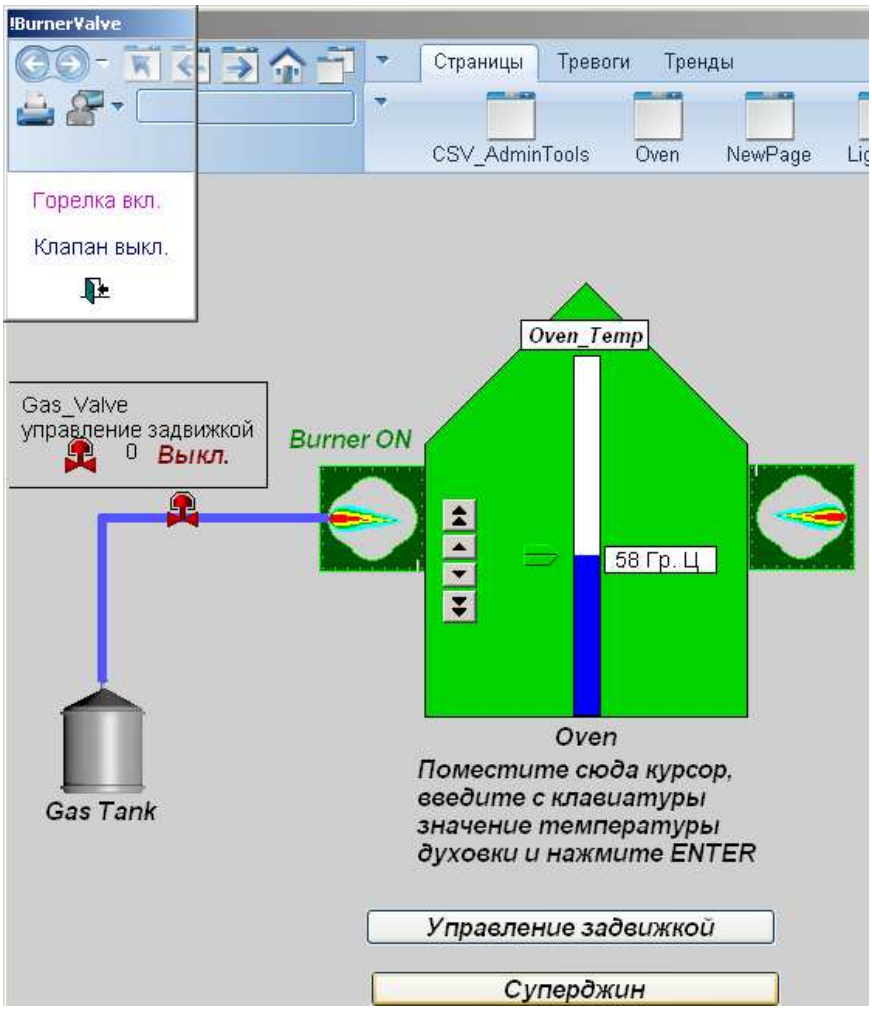


Рис. 17.6. Вызов и тестирование всплывающей страницы с суперджинном

Завершите работу проекта и сохраните проект.

17.3. Объекты ActiveX и Vijeo Citect

Vijeo Citect позволяет включать в графические страницы проектов объекты **ActiveX**. Это дает возможность пользоваться объектами **ActiveX**, разработанными как в рамках Vijeo Citect, так и вне Vijeo Citect.

Важно иметь в виду, что поведение объекта **ActiveX** в Vijeo Citect в значительной мере определяется свойствами самого объекта. Функциональность, надежность и пригодность объекта к использованию с Vijeo Citect будут зависеть от способа создания этого объекта разработчиком.

Проще всего вставить объект **ActiveX** в проект Vijeo Citect с помощью **Потсроителя графики Citect**. Воспользовавшись кнопкой **ActiveX** в окне графических объектов можно выбирать и вставлять объекты **ActiveX** в свои графические страницы (рис. 17.7). Это действие во многом похоже на вставку обычных графических объектов. Как и другие графические объекты, их можно перемещать, копировать и изменять размеры.

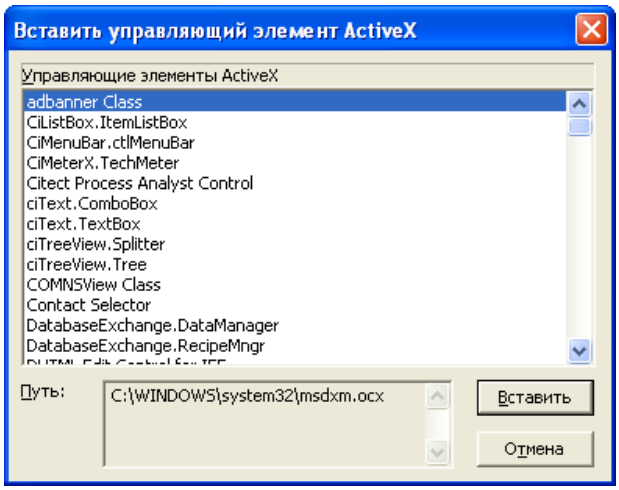


Рис. 17.7. Размещение объекта **ActiveX** на графической странице

Упражнение 17.3. Использование объектов ActiveX

Перейдите в среду **Проводника Citect** и выберите проект **Example**. Перейдите в среду **Построителя графики Citect** и откройте графическую страницу **ActiveX** (рис. 17.8). Страница содержит сгруппированный графический объект на основе слайдера (Vijeo Citect, не **ActiveX** объект, по свойствам напоминающий последний) и пять **ActiveX** объектов **CiMeterX.TechMeter** для Vijeo Citect.

Рис. 17.8. Вид графической страницы **ActiveX**

Поэкспериментируйте с этими объектами в ручном и автоматическом режимах и завершите работу проекта.

Изучите свойства графических объектов, входящих в состав сгруппированного объекта **Citect Objects Based Slider**, а также свойства пяти **ActiveX** объектов **CiMeterX.TechMeter** — это очень полезно. Используйте эти элементы в своих проектах.

Для работы с **ActiveX** объектами язык Cicode имеет ряд функций, описание которых содержится в разделе Cicode Programming Reference | Cicode Functions Categories | ActiveX Functions справки.

Большинство объектов **ActiveX** сопровождается поясняющей документацией. Некоторые из них, например, **Анализатор процессов**, имеют файл помощи. Другие могут иметь простые текстовые подсказки, кратко поясняющие каждое свойство. Это зависит от того, что разработчик решил включить в комплект поставки.

Совет

Изучите также демонстрационный проект ActX.

Часть 2. SCADA-система Vijeo Citect. Язык Cicode. Базовый курс (модуль AUT24)

Глава 18. Язык Cicode: назначение языка, структура программы, данные

Глава 19. Операторы языка Cicode

Глава 20. Функции языка Cicode

Глава 21. Структура Cicode-файлов (*.ci). Использование комментариев

Глава 22. Интегрированная среда разработки и отладки Cicode-программ

Глава 23. Использование Cicode-файлов, Cicode-команд и Cicode-функций в системе Vijeo Citect

Далее последовательно приведено краткое описание языка Cicode, рассмотрены интегрированная среда разработки (ИСР) Cicode-программ и особенности использования языка Cicode в системе Vijeo Citect.

Глава 18. Язык Cicode: назначение языка, структура программы, данные

Язык Cicode является специализированным языком, предназначенным для использования в системе Vijeo Citect. Его отличительной особенностью является наличие обширной библиотеки стандартных функций, обеспечивающих полную программную поддержку различных средств Vijeo Citect. Стандартные функции сгруппированы следующим образом (названия групп стандартных функций перечислены в алфавитном порядке):

- Accumulator Functions;
- ActiveX Functions;
- Alarm and Alarm Filter Functions;
- Clipboard Functions;
- Cluster Functions;
- Color Functions;
- Communication Functions;
- Dynamic Data Exchange Functions;
- Device Functions;
- Display Functions;
- DLL Functions;
- Equipment Database Functions;
- Event Functions;
- Error Functions;
- File Functions;
- Form Functions;
- Format Functions;
- FTP Functions;
- FuzzyTech Functions;
- Group Functions;
- I/O Device Functions;
- Keyboard Functions;
- Mail Functions;
- Math/Trigonometry Functions;
- Menu Functions;

- Miscellaneous Functions;
- Page Functions;
- Plot Functions;
- Process Analyst Functions;
- Quality Functions;
- Report Functions;
- Scheduler Functions;
- Security Functions;
- Sequence of Events Functions;
- Server Functions;
- Statistical Process Control Functions;
- SQL Functions;
- String Functions;
- Super Genie Functions;
- Tag Functions;
- Task Functions;
- Table (Array) Functions;
- Time and Date Functions;
- Timestamp Functions;
- Trend Functions;
- Window Functions.

Характеристика групп функций и отдельных функций будет приведена далее и имеется справочной системе в [2], тема Cicode Programming Reference. Язык Cicode использует средства, присущие языкам C и Visual Basic, но не является языком объектно-ориентированного программирования. Язык интегрирован в систему Vijeo Citect в виде приложения **Редактор Cicode**, которое, по сути, представляет собой интегрированную среду разработки, отладки и выполнения Cicode-программ, CitectVBA-программ и их фрагментов.

18.1. Структура Cicode-программы

Cicode-программа проекта представляет собой один или несколько Cicode-файлов с расширением **.ci** и/или включаемых Cicode-файлов с расширением **.cii**. Эти файлы располагаются в компоненте **Файлы Cicode** проекта (рис. 18.1). Кроме того, фрагменты Cicode-программы могут использоваться в диалогах настройки свойств графических объектов приложения (рис. 18.2).

Никлаус Вирт — создатель языка Паскаль, дал следующее определение программы:

Программа = Структуры данных + Алгоритмы

Структуры данных представляют то, что обрабатывает программа (данные — переменные, массивы), а *алгоритмы* определяют, каким образом данные будут обработаны (представлены операторами и функциями языка). В соответствии с Н. Виртом, далее рассмотрим сначала данные, а затем операторы и функции языка Cicode.

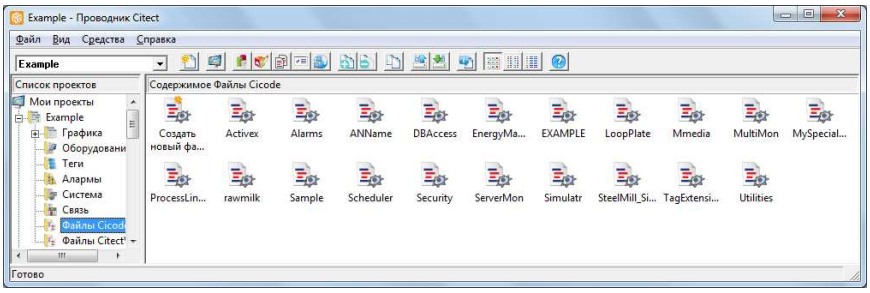


Рис. 18.1. Файловая структура Cicode-программы проекта **Example**

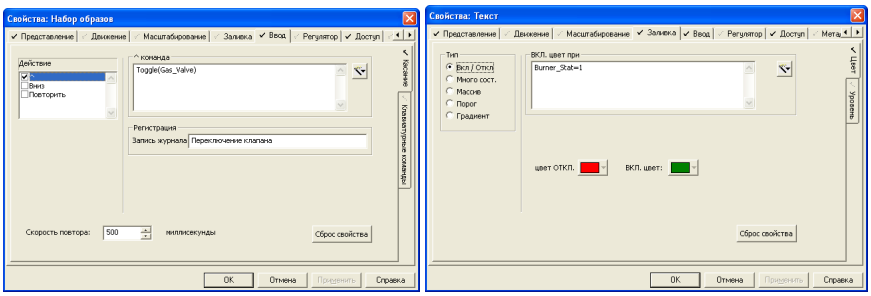


Рис. 18.2. Использование фрагментов Cicode-программы в диалогах настройки свойств графических объектов программы (поля ^ **команда** и **ВКЛ. цвет при**)

18.2. Данные языка Cicode

В качестве данных в Cicode-программе можно использовать константы, переменные, массивы, локальные переменные и теги переменных, которые могут иметь различные типы.

18.2.1. Типы данных

Типы данных языка Cicode представлены в табл. 18.1.

Замечание
При вводе служебного слова с использованием строчных символов по окончании ввода служебного слова строчные символы автоматически заменяются на прописные, а служебное слово выделяется синим цветом.

Таблица 18.1. *Типы данных языка Cicode*

Служебное слово (прописные символы, выделение синим цветом)	Тип данного (формат)	Диапазон значений
INT	Целое (32 бита)	От -2 147 483 648 до +2 147 483 647
REAL	С плавающей точкой (32 бита)	От -3.4E38 до 3.4E38
STRING	Текстовая строка (не более 256 символов, включая нуль-символ; не более 128 — для глобальных и модульных строк)	ASCII с завершающим нулевым символом
OBJECT	Управляющий элемент ActiveX (32 бита)	
QUALITY	Представляет качество Vijeo Citect	QUAL_GOOD, QUAL_BAD, QUAL_UNCR
TIMESTAMP	64-битовая значение, представляющее количество интервалов длительностью 100 наносекунд, прошедших с 1 января 1601	

Совет
Если необходимо определить тип данных **DIGITAL**, используемый в тегах переменных, то можно для этого использовать тип **INT** (при этом **TRUE** экв. 1, а **FALSE** экв 0).

18.2.2. Определение переменных

Определение переменной, в общем случае, может содержать до шести полей, из которых три поля не являются обязательными (они заключены в квадратные скобки):

[<Область_действия>]<Тип><Имя>[=<Начальное_значение>]<;>[<Комментарий>]

Необязательное поле **Область действия** может иметь значение **GLOBAL** (переменная доступна во всех файлах Cicode-программы), **MODULE** (переменная доступна только в одном файле Cicode-программы, в котором она определена на внешнем уровне) или **LOCAL** (по умолчанию, переменная доступна только в одной функции Cicode-программы, внутри которой она определена).

Обязательное поле **Тип переменной** может содержать **INT** (32 бита), **REAL** (32 бита), **STRING** (до 255 символов для **LOCAL** или до 127 для остальных областей действия), **OBJECT** (32 бита), **QUALITY** или **TIMESTAMP**.

Обязательное поле **Имя переменной** содержит идентификатор переменной, первые 32 символа которого должны быть уникальными. Только первые 32 символа и будут восприниматься компилятором.

В необязательном поле **Начальное значение** может содержать соответствующую константу.

В необязательном поле **Комментарий** помещают текст, поясняющий назначение переменной. Поле комментария может начинаться символом <**!**> и тогда оставшаяся часть строки будет комментарием.

18.2.2.1. Правила именования переменных

Рекомендуется использовать следующие правила именования переменных, сведенные в табл. 18.2.

Таблица 18.2. *Правила именования переменных*

Тип	Префикс идентификатора переменной	Область применения переменной
INT (32 бита)	i	Для индексов и счетчиков циклов
INT и **OBJECT** (32 бита)	h	Для дескрипторов
INT (32 бита)	b	Для булевских переменных (**TRUE**/**FALSE**)
REAL (32 бита)	r	
STRING (до 127 или 255 символов)	s	

Замечание

Для переменных с областью действия **GLOBAL** и **MODULE** перед префиксами, указанными в табл. 18.2, рекомендуется использовать соответственно префиксы **g** и **m**.

18.2.2.2. Примеры определения переменных

Примеры определения переменных и вид ИСР Cicode-программы приведены на рис. 18.3.

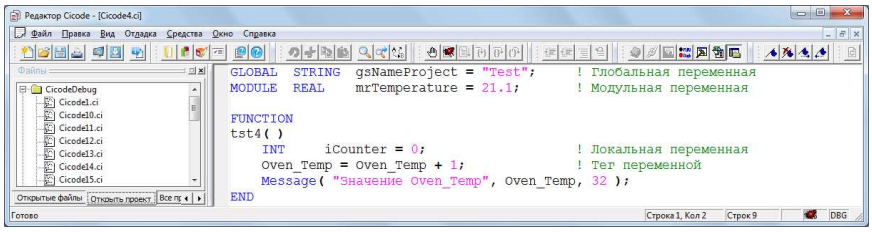


Рис. 18.3. Примеры определения переменных и вид ИСР Cicode-программы

Совет

Обратите внимание на ступенчатую запись, оформление комментариев и цветовые выделения в ИСР.

18.2.3. Определение массивов

Определение массивов имеет такой же синтаксис, что и определение переменных, но имеются отличия в заполнении трех полей.

В поле **Область действия** можно использовать только служебные слова **GLOBAL** и **MODULE** (по умолчанию). Объясняется это тем, что не предусмотрено определение массива внутри функции.

В поле **Имя** указывается не только имя массива, но и его размеры. Каждый из размеров массива указывается внутри пары квадратных скобок, как это будет показано далее. Для одномерного массива (вектора) внутри пары квадратных скобок задается только его единственный размер, означающий количество элементов массива. Для двумерного массива (матрицы) после имени массива используется две пары квадратных скобок — внутри первой задается число строк матрицы, а внутри второй — число столбцов матрицы. Для трехмерного массива, который можно интерпретировать как вектор матриц, после имени массива следуют три пары квадратных скобок, внутри, каждой из которых, задается один из размеров массива. Массивы большей размерности не предусмотрены.

В поле **Начальное значение** задается не одно начальное значение, а их список, в котором начальные значения элементов массива отделены друг от друга запятыми. Если это поле отсутствует, то выполняется автоматическая инициализация значений элементов массива нулями (пустыми строками). Примеры определения массивов приведены на рис. 18.4.

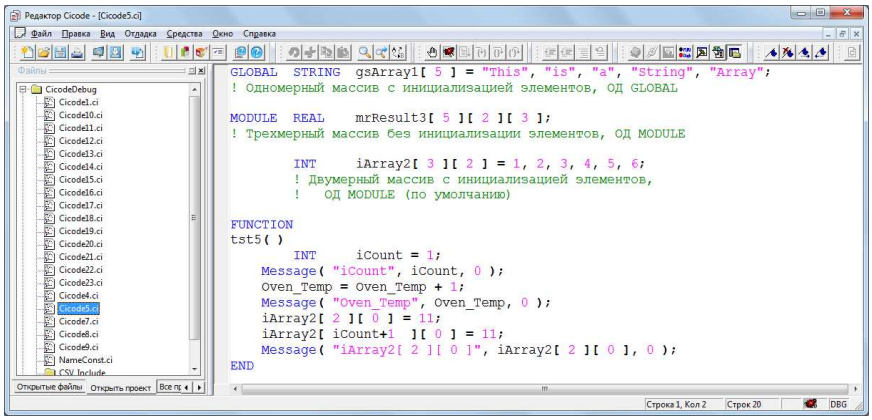


Рис. 18.4. Примеры определения массивов

Как следует из приведенных примеров, обращение к элементам массивов выполняется с помощью индексных выражений, записываемых внутри квадратных скобок, помещаемых после идентификатора массива. Как и в языке C, индексация начинается с нуля. Следовательно, максимальное значение индексного выражения должно быть на единицу меньше соответствующего размера массива.

> ***Совет***
> Обратите внимание на использование функции **Message()**, которая использована для отладочного вывода.

В примере, приведенном на рис. 18.4, определение одномерного массива строк **gsArray1** совмещено с инициализацией его элементов. При этом элементы массива получают следующие значения:

```
gsArray1[ 0 ] = "This"
gsArray1[ 1 ] = "is"
gsArray1[ 2 ] = "a"
gsArray1[ 3 ] = "String"
gsArray1[ 4 ] = "Array"
```

Примеры определения двумерного и трехмерного массивов с инициализацией его элементов иллюстрирует верхняя часть рис. 18.5. Двумерные массивы (матрицы) располагаются в памяти построчно и в таком же порядке инициализируются. Аналогичным образом выполняется и обработка трехмерных массивов.

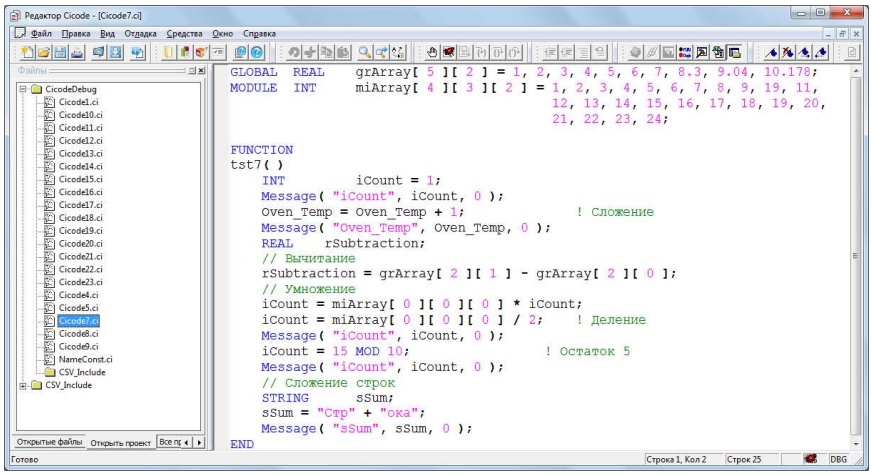


Рис. 18.5. Примеры определения двумерного и трехмерного массивов с инициализацией элементов

18.3. Константы языка Cicode

Константы языка Cicode можно записывать в виде *литералов* или в форме *именованных констант*.

Приведем примеры *констант-литералов*:

`1.0e-3, -2.4` (тип **REAL**); `255, -100` (тип **INT**); `0xFF` (шестнадцатеричная константа, тип **INT**); **TRUE/FALSE** (булевские константы, тип **INT**); `"Строка"` (строковая константа)

Константы можно записывать также в виде *именованных констант*. В этом случае идентификатор константы должен иметь префикс **c**, а константа определена и инициализирована в начале файла Cicode:

```
MODULE REAL          cmrPI = 3.14;          ! Именованная константа
! Именованная константа
GLOBAL STRING cgsProjectName = "Generic";

FUNCTION
tst( )
       Message( "cmrPI", cmrPI, 0 );
       Message( "cgsProjectName", cgsProjectName, 0 );
END
```

Замечание

К сожалению, приходится констатировать, что в языке Cicode, в отдичие от других языков программирования, разрешено изменять значения именованных констант. Префикс **c** при этом является для программиста только индикатором, что именованная

константа является, по-сути, переменной "только для чтения" и менять ее значение не следует.

Глава 19. Операторы языка Cicode

Далее — о второй "стороне медали". Перейдем к рассмотрению операторов языка и их составных частей — выражений.

19.1. Выражения языка Cicode

В состав многих операторов языка входят *выражения*, которые мы и рассмотрим далее. Выражения строятся из *операндов* (участников выражения) и *знаков операций*. В качестве *операндов* выражения можно использовать константы, переменные, в том числе локальные переменные и теги переменных, элементы массивов и указатели функций, имеющих возвращаемые значения. *Знаки операций* можно разделить на знаки арифметических, битовых операций, операций отношений и логических операций.

19.1.1. Арифметические операции

Арифметические операции предназначены для арифметических вычислений над целыми операндами или операндами с плавающей точкой. При этом левый и правый операнды могут иметь неодинаковые типы (исключение — операция **MOD** требует целых операндов). Знаки арифметических операций представлены в табл. 19.1.

Таблица 19.1. *Арифметические операции языка Cicode*

Обозначение	Назначение
+	Сложение
-	Вычитание
*	Умножение
/	Деление
MOD	Получение остатка от деления целого на целое

Операцию сложения можно использовать также для сложения двух строк. Примеры использования арифметических операций иллюстрирует рис. 19.1. Обратите внимание на возможность использования в Cicode-программе комментариев в стиле языка C++.

19.1.2. Операции над битами

С помощью битовых операций можно сравнивать соответствующие биты левого и правого операндов. Битовые операции требуют целых операндов. Знаки битовых операций представлены в табл. 19.2. Примеры использования битовых операций приведены на рис. 19.2.

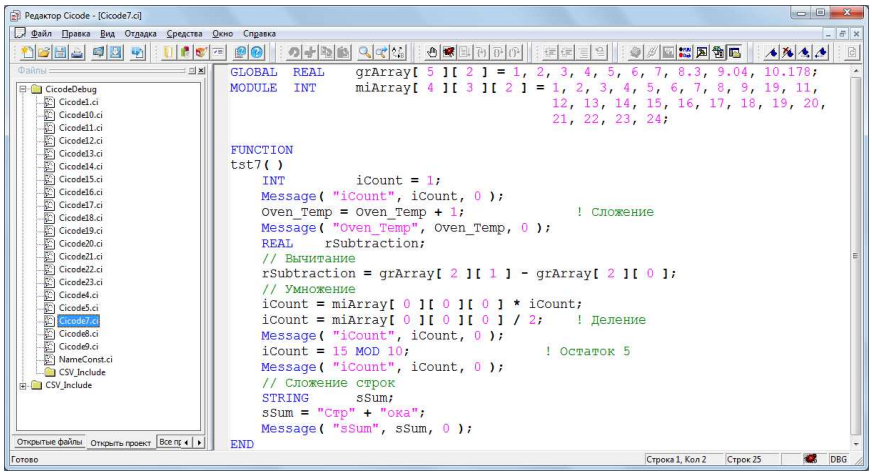


Рис. 19.1. Примеры использования арифметических операций

Таблица 19.2. *Операции над битами языка Cicode*

Обозначение	Назначение
BITAND	Умножение пары битов (**AND**)
BITOR	Сложение пары битов (**OR**)
BITXOR	Исключающее сложение пары битов (**XOR**)

19.1.3. Операции отношений

С помощью операций отношений можно сравнивать значения операндов, в том числе операндов строкового типа. В случае сравнения строк оба операнда должны быть строками. При сравнении арифметических значений операнды могут быть любого арифметического значения (**INT** или **REAL**). Знаки операций отношений представлены в табл. 19.3. Результатом вычисления операции отношения является значение логического (целого) типа — **TRUE** (1) или **FALSE** (0). Примеры использования операций отношений приведены на рис. 19.3.

19.1.4. Логические операции

С помощью логических операций можно проверить выполнение или невыполнение совокупности условий (двух или более). Знаки логических операций представлены в табл. 19.4. Результатом вычисления логической операции является значение логического (целого) типа — **TRUE** (1) или **FALSE** (0). Примеры использования логических операций приведены на рис. 19.4.

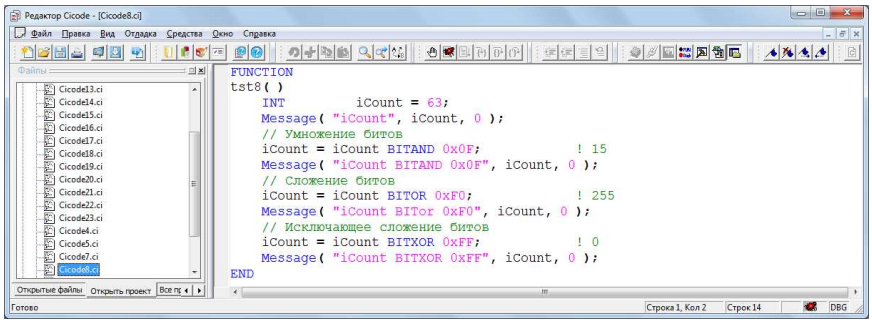


Рис. 19.2. Примеры использования операций над битами

Таблица 19.3. *Операции отношений языка Cicode*

Обозначение	Назначение
=	Проверка на равенство
<>	Проверка на неравенство
<	Проверка на "меньше"
>	Проверка на "больше"
<=	Проверка на "меньше или равно"
>=	Проверка на "больше или равно "

19.1.5. Приоритеты (порядок выполнения) операций

Приоритеты (порядок выполнения) рассмотренных операций показаны в табл. 19.5.

Каждая строка таблицы содержит операции, имеющие одинаковый приоритет, причем, чем выше расположена строка таблицы, тем выше приоритет операций соответствующей строки. Если выражение содержит две или более операции одинакового приоритета, то операции выполняются в том порядке, в каком они встречаются в записи выражения при его просмотре слева направо.

Операторы языка Cicode можно разделить на три группы — *операторы присваивания*, *операторы ветвлений* (условный **IF … THEN** и переключатель **SELECT CASE**) и *циклические операторы* (**FOR … DO** и **WHILE … DO**). Этого достаточно для программирования любых алгоритмов.

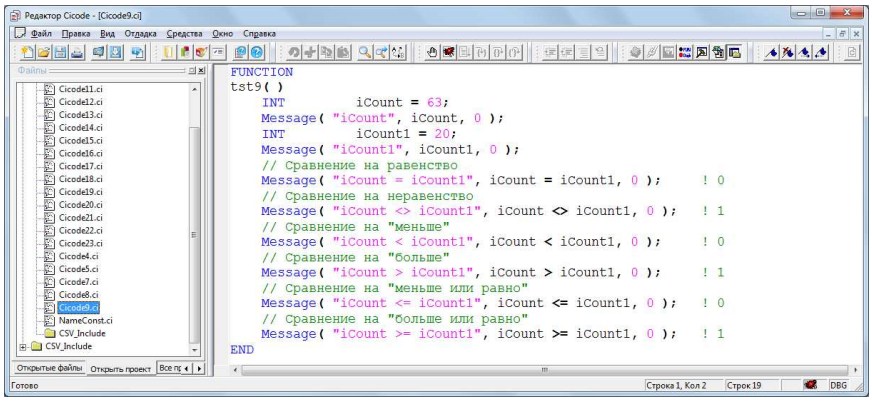


Рис. 19.3. Примеры использования операций отношений

Таблица 19.4. *Логические операции языка Cicode*

Обозначение	Назначение
AND	Логическое умножение
OR	Логическое сложение
NOT	Логическое отрицание

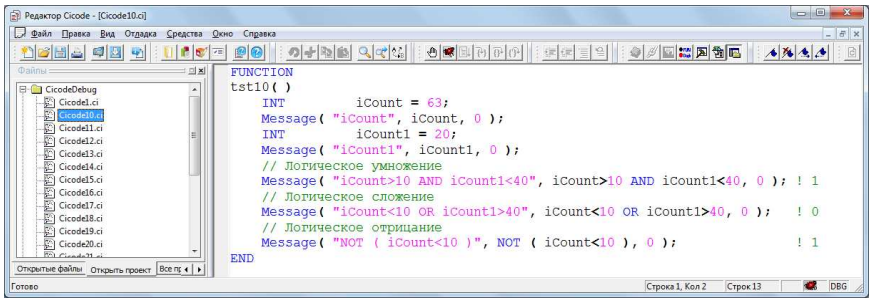


Рис. 19.4. Примеры использования логических операций

Таблица 19.5. *Приоритеты (порядок выполнения) операций языка Cicode*

Обозначение	Назначение
( )	Круглые скобки, самый высокий приоритет
NOT	Логическое отрицание, унарная операция
*** / MOD**	Арифметические мультипликативные операции (бинарные)
:	Операция форматирования (см. об этом далее в разд. "Операторы")
+ -	Арифметические аддитивные операции (бинарные)
< > <= >=	Операции отношений (бинарные)
= <>	Операции отношений (бинарные)
AND	Логическое умножение, бинарная операция
OR	Логическое сложение, бинарная операция
BITAND BITOR BITXOR	Операции над битами (бинарные), самый низкий приоритет

19.2. Оператор присваивания

Оператор присваивания имеет следующую структуру:

```
Переменная или элемент_массива = выражение;
```

Оператор присваивания обрабатывается следующим образом. Вначале вычисляется значение выражения, стоящего справа от знака присваивания. В частном случае выражение может иметь вид переменной, тега переменной или константы и тогда вычисление не требуется. Затем полученное значение приводится к типу переменной или элемента массива, записанному слева от знака присваивания. На последнем этапе преобразованное значение присваивается переменной или элементу массива. Примеры использования операторов присваивания приведены на рис. 19.5.

> ***Совет***
> Обратите внимание на то, что использование цепочек присваиваний в одном операторе не разрешено, и на преобразования вычисленных значений выражений.

Язык Cicode предоставляет четыре стандартных функции для преобразования целых и вещественных значений в строки и наоборот — **IntToStr** (преобразование переменной или выражения целого типа в строку), **RealToStr** (преобразование переменной или выражения вещественного типа в строку), **StrToInt** (преобразование переменной или выражения строкового типа в целое) и **StrToReal** (преобразование переменной или выражения строкового типа в вещественное значение).

> ***Замечание***
> Выполнять преобразования данных можно автоматически и без использования перечисленных стандартных функций (см. приведенные ранее примеры на рис. 19.5).

Но, чтобы обеспечить полный контроль над процессом преобразования, все же предпочтительнее использование перечисленных ранее стандартных функций. И еще одно важное замечание — переменные с типом **OBJECT** нельзя преобразовать ни к какому другому типу.

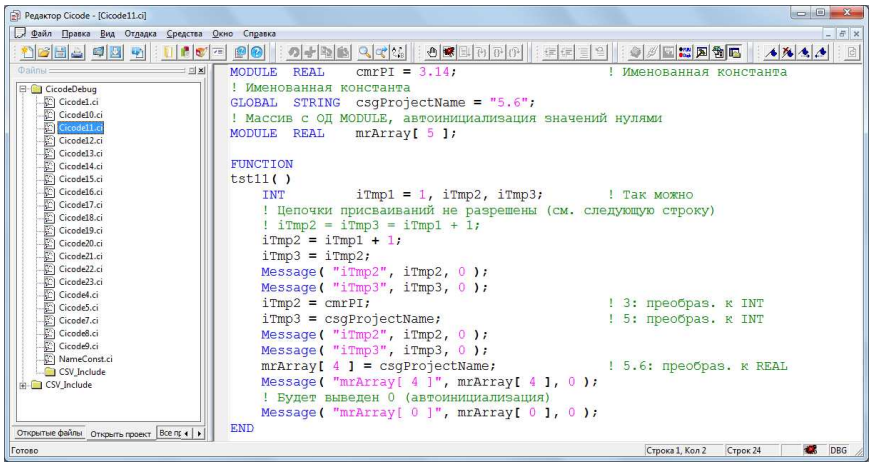


Рис. 19.5. Примеры использования операторов присваивания

Различные примеры преобразования целого значения в строку приведены на рис. 19.6.

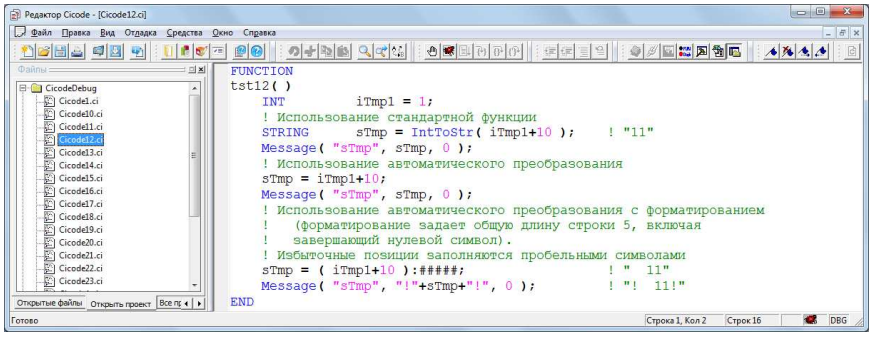


Рис. 19.6. Примеры преобразования целого значения в строку

Совет

Обратите внимание на форматирование результирующей строки (:#####) и комментарий на рис. 19.6, относящийся к форматированию.

Различные примеры преобразования вещественного значения в строку приведены на рис. 19.7.

> ***Совет***
> Обратите внимание на смысл аргументов в вызове функции **RealToStr** и средства форматирования результирующей строки при автоматическом преобразовании (см. приведенный ранее рис. 19.7).

Различные примеры преобразования строки в целое значение приведены на рис. 19.8.

> ***Совет***
> Внимательно изучите приведенные примеры — это очень полезно.

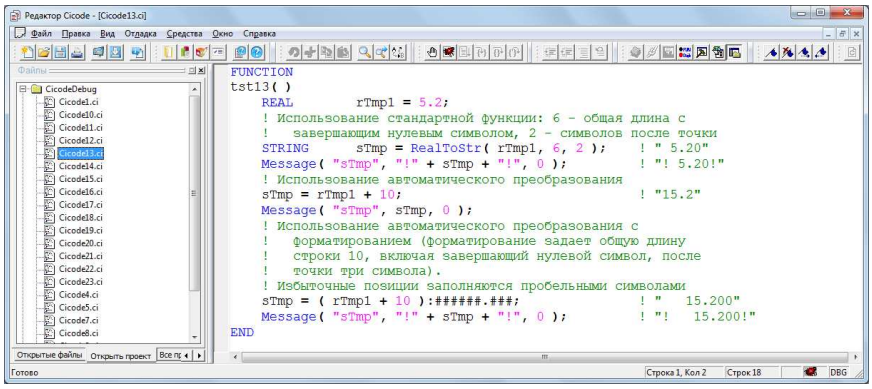


Рис. 19.7. Примеры преобразования вещественного значения в строку

Различные примеры преобразования строки в вещественное значение приведены на рис. 19.9.

19.2.1. Форматирование текстовых строк. Использование Escape-последовательностей

Строки можно форматировать, используя операцию сложения (+) строк и Escape-последовательности. Различные примеры подобного форматирования строк приведены на рис. 19.10. Обратите внимание на использование в одном из примеров Escape-последовательности. Полный перечень Escape-последовательностей приведен в табл. 19.6.

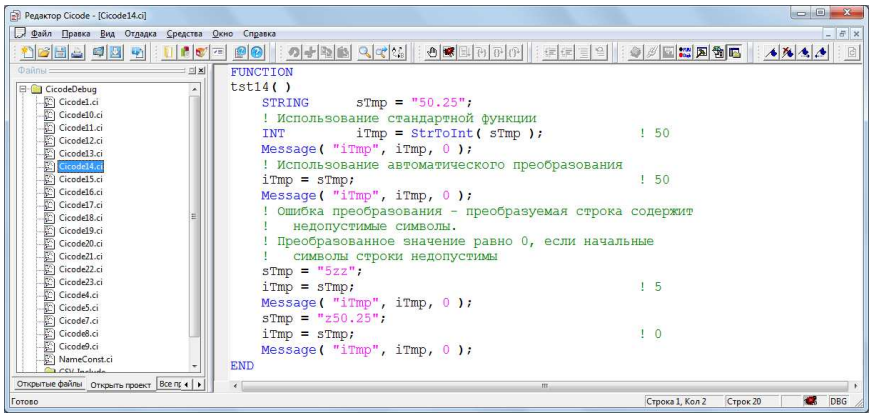


Рис. 19.8. Примеры преобразования строки в целое значение

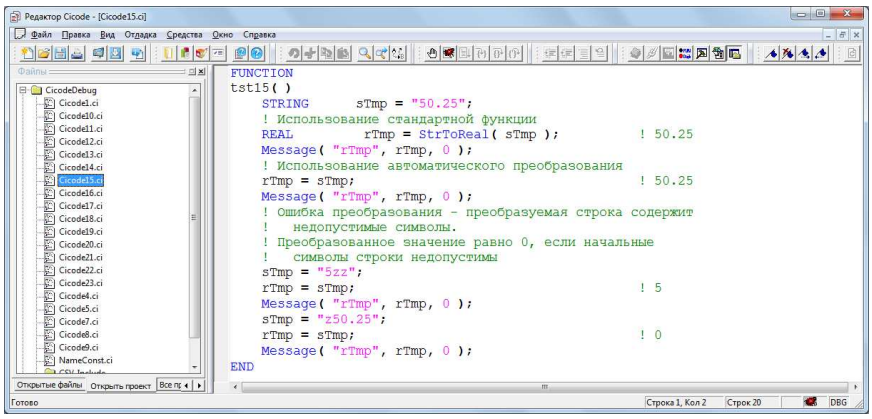


Рис. 19.9. Примеры преобразования строки в вещественное значение

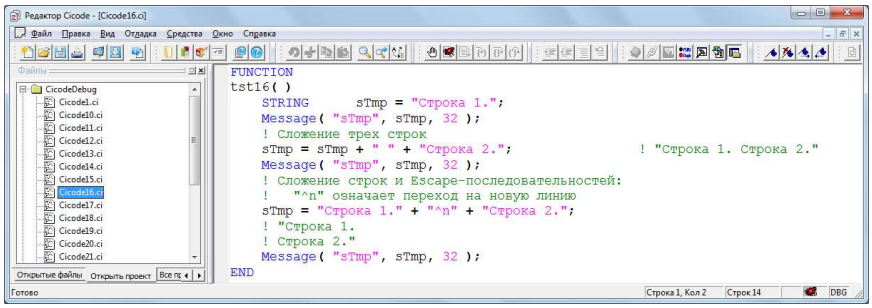


Рис. 19.10. Примеры форматирования строк

Таблица 19.6. *Escape-последовательности языка Cicode*

Обозначение	Назначение
^b	Backspace
^f	form feed
^n	new line
^t	horizontal tab
^v	vertical tab
^'	single quote
^"	double quote
^^	Caret
^r	carriage return
^0x*hh*	where *hh* is a hexadecimal number (for example, ^0x1A)

19.3. Операторы ветвлений

К операторам ветвлений относятся *условный оператор*, который можно записывать в полной или сокращенной форме, и *переключатель*.

19.3.1. Условный оператор

Условный оператор **IF** обеспечивает выполнение одного или нескольких операторов в зависимости от значения проверяемого выражения. Оператор **IF** можно использовать в *полной* (**IF THEN ELSE**) или *сокращенной* (**IF THEN**) формах:

```
! Сокращенная форма
IF <выражение> THEN
    оператор(ы); ! Выполняются, если проверяемое <выражение>
```

```
                ! имеет ненулевое значение
END
! Полная форма
IF <выражение> THEN
    оператор(ы); ! Выполняются, если проверяемое <выражение>
                 ! имеет ненулевое значение
ELSE
    оператор(ы); ! Выполняются, если проверяемое <выражение>
                 ! имеет нулевое значение
END
```

Здесь в качестве `оператора(ов)` можно использовать *любые* операторы языка, в том числе условный оператор **IF** в полной или сокращенной форме и тогда получаем гнездо вложенных операторов **IF**.

Примеры использования условного оператора **IF** приведены соответственно на рис. 19.11 — 19.13.

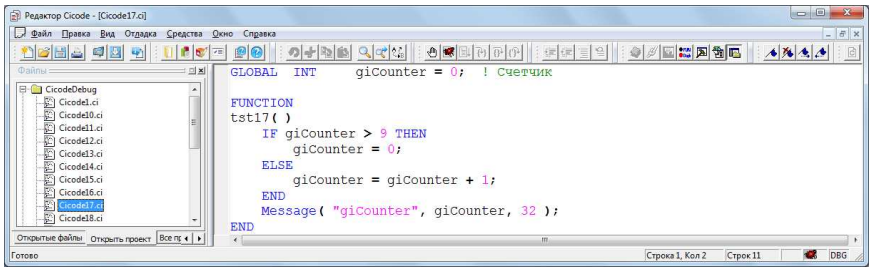


Рис. 19.11. Пример использования условного оператора **IF** в полной форме

> ***Совет***
> Если внешний оператор **IF** в гнезде условных операторов записан в полной форме, то вложенный оператор **IF** помещайте в ветвь **ELSE**. В этом случае код получается более понятным и его проще сопровождать. Обратите внимание на ступенчатую запись условных операторов **IF** и их построчное расположение.

19.3.2. Переключатель

Переключатель выполняет одну из групп операторов в зависимости от значения проверяемого выражения. Переключатель является более эффективным по сравнению с гнездом операторов **IF** способом записи кода и его использование предпочтительнее. Переключатель можно использовать всегда, когда проверяемое выражение единственное. В отличие от других языков программирования проверяемое выражение может быть любого типа, кроме типа **OBJECT**.

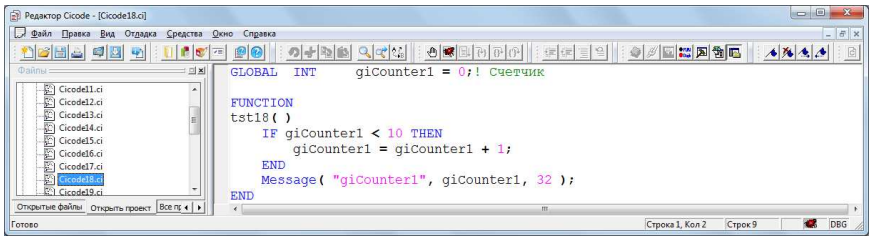


Рис. 19.12. Пример использования условного оператора **IF** в сокращенной форме

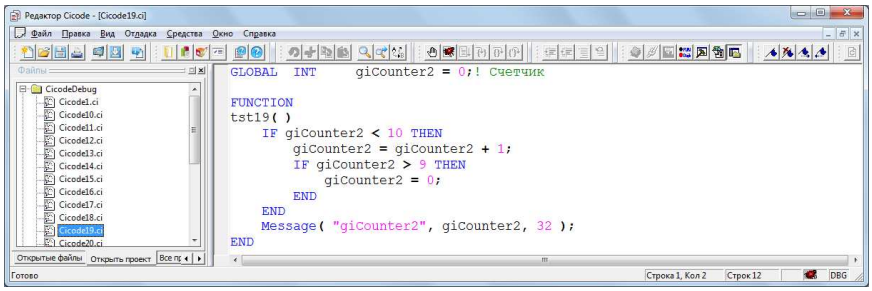


Рис. 19.13. Пример использования гнезда условных операторов **IF**

В общем случае переключатель имеет следующий формат:

```
SELECT CASE Проверяемое_выражение
        CASE Выражение_варианта1, Выражение_варианта2
            Оператор(ы);
        CASE Выражение_варианта3 TO Выражение_варианта4
            Оператор(ы);
        CASE IS < Выражение_варианта5, IS > Выражение_варианта6
            Оператор(ы);
        CASE ELSE
            Оператор(ы);
END SELECT
```

Выражение варианта может иметь вид `выражение` или `выражение` **`TO`** `выражение`. В выражении варианта со служебным словом **`IS`** может использоваться любая из операций отношений. И, наконец, в каждом из вариантов можно использовать список любых разновидностей выражений вариантов, разделенных запятыми.

Примеры использования переключателей иллюстрирует рис. 19.14. Переключатель имеет синтаксис и семантику такие же, как в языке Microsoft Visual Basic.

Совет

Внимательно изучите примеры, приведенные на рис. 19.14, поэкспериментируйте с ними, задавая различные значения проверяемого выражения, протестируйте работу всех вариантов переключателей.

19.4. Циклические операторы

В языке Cicode предусмотрены два циклических оператора — цикл **FOR … DO** с заранее заданным количеством повторений группы операторов, называемой телом цикла, и итерационный цикл **WHILE … DO** с заранее неизвестным количеством повторений тела цикла. Оба циклических оператора реализуют циклы с предусловием, отличающиеся тем, что тело цикла может выполняться любое количество раз, в том числе, в частном случае, ни одного раза. *Цикл с постусловием, имеющийся в большинстве языков программирования высокого уровня, в языке Cicode отсутствует.*

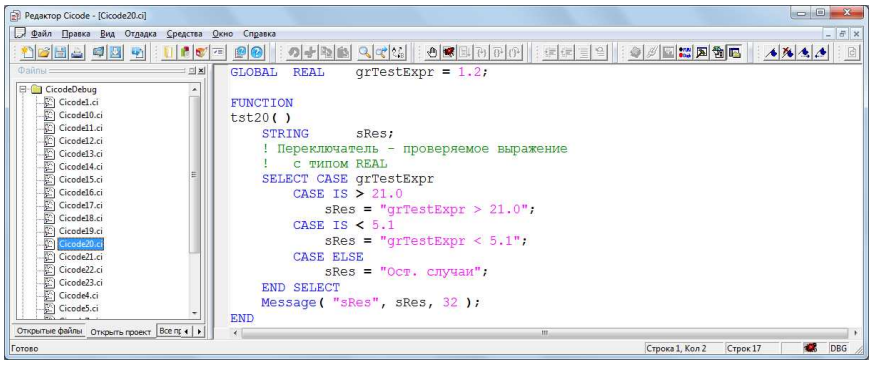


Рис. 19.14. Примеры использования переключателя

19.4.1. Цикл FOR … DO

Цикл **FOR … DO** имеет следующую структуру:

```
FOR Переменная_цикла = Выражение1 TO Выражение2 DO
    Оператор(ы);                    ! Тело цикла
END
```

В частном случае, когда значение `Выражения1` превышает значение `Выражения2`, тело цикла не будет выполнено ни разу. `Переменная_цикла` в процессе работы принимает последовательные значения от значения `Выражения1` до значения `Выражения2` с шагом единица и должна иметь целый тип.

В качестве операторов в теле цикла могут использоваться *любые* другие операторы. В частности, в качестве таких операторов могут использоваться циклические операторы, и тогда мы получаем гнездо циклических операторов.

Пример использования цикла **FOR … DO** приведен на рис. 19.15.

> ***Совет***
> Обратите внимание на стандартные функции **Prompt** (вывод на графическую страницу) и **Sleep** (программная задержка). Функцией **Sleep** пользуйтесь осмотрительно — ее использование существенно загружает центральный процессор.

19.4.2. Цикл WHILE … DO

Цикл **WHILE … DO** имеет следующую структуру:

```
WHILE Проверяемое_выражение DO
    Оператор(ы);                    ! Тело цикла
END
```

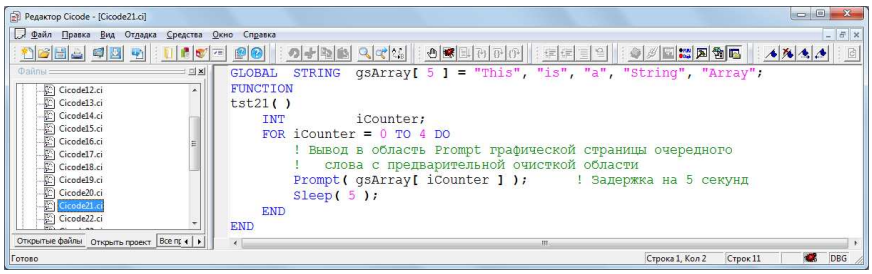


Рис. 19.15. Пример использования цикла **FOR … DO**

В частном случае, когда начальное значение `Проверяемого_выражения` нулевое, тело цикла не будет выполнено ни разу. В процессе работы выполнение тела цикла будет повторяться до тех пор, пока значение `Проверяемого_выражения` не станет нулевым.

Пример использования цикла **WHILE … DO** приведен на рис. 19.16. Нетрудно заметить, что для приведенного примера использование цикла **FOR … DO** предпочтительнее. Однако, когда заранее неизвестно количество повторений тела цикла, применение цикла **WHILE … DO** является единственной возможностью.

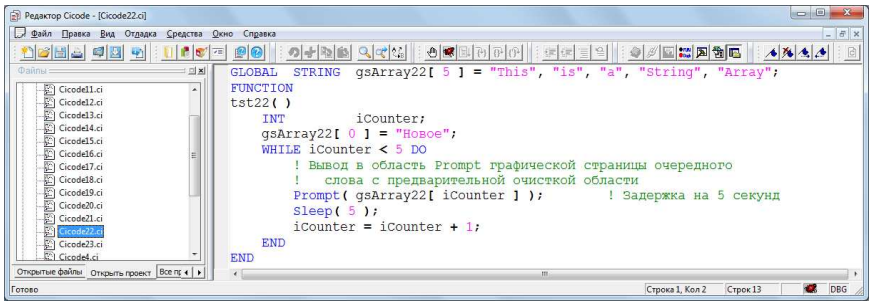


Рис. 19.16. Пример использования цикла **WHILE … DO**

Глава 20. Функции языка Cicode

Функции языка Cicode поддерживают *технологию модульного программирования*, в соответствии с которой решаемая задача путем декомпозиции разбивается на более простые функционально законченные подзадачи. Каждая из таких задач записывается в виде функции языка Cicode.

Психологами установлено, что каждый человек в среднем в состоянии хорошо воспринимать не более семи объектов (правило семи). В соответствии с этим правилом функции, реализующие подзадачи должны соответствовать следующим ограничениям. Число операторов внутри функции не должно превышать семи, а запись оператора не должна содержать более семи строк. Следовательно, общее количество строк в записи функции не должно превышать 50 (7*7). Аналогично, количество параметров функции не должно превышать семи.

20.1. Синтаксис и семантика определения функции

В приводимом далее определении синтаксиса функции, как и ранее, элементы ее определения, заключенные в квадратные скобки, не являются обязательными и могут отсутствовать. Функция языка Cicode имеет следующий синтаксис:

```
[ Область_действия ]
[ Тип_возвращаемого_значения ]
FUNCTION
Имя_функции( [ Список_параметров ] )
    Оператор;
...
    Оператор;
    [ RETURN Возвращаемое_значение; ]
END
```

Область_действия — необязательный элемент определения функции. Может принимать значения **PRIVATE** или **PUBLIC** (по умолчанию **PUBLIC**) и записывается в отдельной строке. После служебного слова точка с запятой не ставится. PUBLIC-функция доступна во всех Cicode-файлах, графических страницах и базах данных системы Vijeo Citect (например, в Alarm.dbf). PRIVATE-функция доступна только в файле, где она определена.

Тип_возвращаемого_значения — необязательный элемент определения функции. Может принимать значения **INT**, **REAL**, **STRING** или **OBJECT** (умолчание отсутствует) и записывается в отдельной строке. После служебного слова точка с запятой не ставится.

FUNCTION — обязательное служебное слово, размещаемое в отдельной строке, после которого точка с запятой не ставится. Отмечает начало функции.

Имя_функции — обязательный элемент определения функции, может содержать до 32 символов, уникальный идентификатор, регистр любой. Размещается в отдельной строке, в конце точка с запятой не ставится. Название функции должно включать, по крайней мере, следующую информацию — от трех до пяти символов описания назначения функции (например, `Trend, Plot, Win`); одно или два слова, описывающих обрабатываемые данные (например, `Info, ClientInfo, Mode`), и одно слово, характеризующее выполняемое действие (например, `Get, Set, Init, Read`). Например, приемлемыми именами функций могут быть `PlotInit,TrendClientOpen,TrendClientInfoRead`.

Список_параметров — необязательный элемент определения функции. При его отсутствии сразу же за именем функции следует пара круглых скобок, внутри которых ничего нет. При наличии параметров они помещаются внутри круглых скобок и разделяются запятыми. Круглые скобки следуют сразу же за именем функции в той же самой строке. Можно использовать параметры с типами **INT**, **REAL** или **STRING**. Параметры могут иметь значения по умолчанию. При необходимости, для удобства восприятия, параметры можно записывать не в одной, а в нескольких строках. После закрывающей круглой скобки точка с запятой не ставится. Параметр функции имеет следующий синтаксис:

```
Тип_параметра Имя_параметра [ = Значение_по_умолчанию ]
```

Имя_параметра может содержать до 32 символов, уникальный идентификатор без пробелов внутри, регистр любой. **Значение_по_умолчанию** должно быть константой того же самого типа, что и тип параметра. Это значение будет использовано, если в вызове функции не будет указан соответствующий аргумент. Параметры с умалчиваемыми значениями должны располагаться в конце списка параметров.

RETURN `Возвращаемое_значение;` — необязательный оператор в определении функции. Но он должен обязательно присутствовать, если функция имеет возвращаемое значение. В качестве **Возвращаемого_значения** можно использовать константу, переменную или выражение, тип которого должен совпадать с типом значения, возвращаемого функцией. После **Возвращаемого_значения** должна следовать точка с запятой.

Оператор; — каждый оператор должен размещаться в отдельной строке.

END — обязательный элемент определения функции, отмечающий конец функции. Должен размещаться в отдельной строке. Является служебным словом, после которого не должно быть точки с запятой.

На рис. 19.16, приведенном ранее, дан пример определения функции без возвращаемого значения с областью действия **PUBLIC** по умолчанию, а на рис. 20.1 приведен исчерпывающий набор и других примеров определения функций.

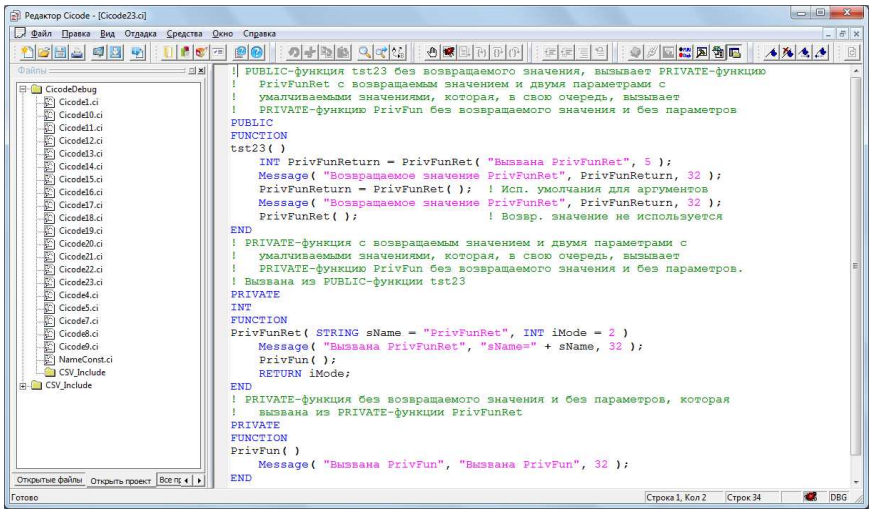


Рис. 20.1. Примеры определения функций

20.2. Синтаксис и семантика вызова функции

Синтаксис вызова функции зависит от того, имеет функция возвращаемое значение или нет.

Если функция не имеет возвращаемого значения, то она вызывается с помощью указателя:

```
Имя_функции( [ Список_аргументов ] );
```

Список_аргументов является необязательным элементом вызова функции. Он отсутствует, если функция не имеет параметров, или может отсутствовать, если все параметры имеют значения по умолчанию. В качестве примера см. вызовы функций **PrivFunRet** и **PrivFun** в примерах, приведенных ранее на рис. 20.1.

Если функция имеет возвращаемое значение, то она вызывается с помощью оператора присваивания:

```
Переменная = Имя_функции( [ Список_аргументов ] );
```

В качестве **Переменной** можно использовать переменную, элемент массива или тег переменной, причем их тип должен совпадать с типом значения, возвращаемого функцией. Все сказанное ранее относительно **Списка_аргументов** справедливо и в данном случае. Если возвращаемое значение не нужно, то вызов функции можно выполнять не с помощью оператора присваивания, а с помощью указателя. В качестве примера см. вызовы функций **PrivFunRet** в примерах, приведенных ранее на рис. 20.1.

Глава 21. Структура Cicode-файлов (*.ci). Использование комментариев

Как указывалось ранее, в любом проекте системы Vijeo Citect в составе Cicode-программы могут использоваться Cicode-файлы кода с расширениями **.ci** и включаемые Cicode-файлы с расширениями **.cii**. (они будут рассмотрены позднее).

> ***Замечание***
> В частном случае проект может не содержать Cicode-файлов.

Далее рассмотрим варианты оформления комментариев, структуру файлов с расширениями **.ci** и рациональное использование комментариев в заголовках файлов и в заголовках определения функций.

21.1. Синтаксис комментариев

Разумное использование комментариев во всех Cicode-файлах соответствует хорошему стилю программирования. Комментарии позволяют вам (или другому разработчику) быстро понять назначение переменной или работу функции, что важно при модификации Cicode-программы. Компилятор языка Cicode, кроме собственного однострочного комментария, начинающегося с символа **!**, распознает комментарии в силе языков C и C++:

```
! Однострочный комментарий Cicode
WHILE DevNext( hDev ) DO
    Counter = Counter + 1;    ! Локальный комментарий Cicode
END
/* Многострочный комментарий в стиле языка C (в частном
   случае он может занимать одну строку или часть строки).
   Обратите внимание, что между символами, отмечающими начало
   и конец комментария, пробел отсутствует.
   Такое оформление комментария целесообразно использовать
   для многострочных комментариев
*/
// Однострочный комментарий в стиле языка C++
Variable = 42;    // Локальный комментарий в стиле языка C++
```

21.2. Структура и использование комментариев в заголовке файла с расширением .ci

Исходные файлы (файлы с расширениями **.ci**, содержащие Cicode) должны содержать заголовочный комментарий с кратким обзором содержимого файла.

Структуру и заголовочный комментарий к файлу целесообразно отформатировать следующим образом:

```
/*
   FILE:        <name of file.CI>

   DESCRIPTION: <description of basically what is in the
                 file>

   FUNCTIONS:   PUBLIC
                <list of the PUBLIC functions contained
                 in this file>
                PRIVATE
                <list of the PRIVATE functions contained
                 in this file>
*/

! ***************** GLOBAL CONSTANTS ***********************
<global constants>          // <comments (purpose)>

! ***************** GLOBAL VARIABLES ***********************
<global variables>          // <comments (purpose)>

! ***************** MODULE CONSTANTS ***********************
<module constants>          // <comments (purpose)>

! ***************** MODULE VARIABLES ***********************
<module variables>          // <comments (purpose)>

! ***************** GLOBAL FUNCTIONS ***********************
<global functions>

! ***************** PRIVATE FUNCTIONS ***********************
<private functions>
```

Замечание

Имейте в виду, что каждая глобальная константа или глобальная переменная должна определяться только в одном экземпляре в одном из файлов в соответствующей секции. Не забывайте инициализировать глобальные и модульные переменные при их определении.

Приведем пример оформления заголовка исходного файла:

```
/*
   FILE:        Recipe.CI
```

```
   DESCRIPTION: contains all functions for gathering recipe
                data

   FUNCTIONS:   PUBLIC
                OpenRecipeDatabase
                CloseRecipeDatabase
                ReadRecipeData
                WriteRecipeData
                GatherRecipeData
                RecipeForm
                OpenRecipeDatabase

                PRIVATE
                ButtonCallback
*/

! ***************** MODULE CONSTANTS ***********************
MODULE INT cmiRecipeMax = 100; //Maximum number of recipes

! ***************** MODULE VARIABLES ***********************
MODULE INT miRecipeNumber = 0; //Minimum number of recipes
```

21.3. Использование комментариев в заголовке определения функции

Каждая функция, в соответствии с хорошим стилем программирования, должна иметь заголовочный комментарий, отформатированный следующим образом:

```
/*
   FUNCTION :      <name of function>

   DESCRIPTION:    <suggests the operation, application
                    source and multi-tasking issues>

   REVISION          DATE   BY       COMMENTS
   <revision number> <date> <author> <comments about the
                                      change>

   ARGUMENTS:      <argument description>

   RETURNED VALUE: < description of possible return values>

   NOTES:
```

```
*/
```

Приведем пример оформления заголовка и определения функции:

```
/*
   FUNCTION :        OpenRecipeDatabase

   DESCRIPTION:      opens the specified database

   REVISION          DATE      BY    COMMENTS
   1                 28/09/97  BS    Original
   2                 05/10/97  SFA   Added INI checking

   ARGUMENTS:
       STRING sName   Name of the recipe database
       INT    dwMode  Mode to open the recipe database (0 for
                      read only, 1 for read/write)

   RETURNED VALUE:    handle if successful, otherwise -1

   NOTES:
*/
FUNCTION
OpenRecipeDatabase( STRING sName, INT dwMode)
    ...
END
```

Глава 22. Интегрированная среда разработки и отладки Cicode-программ

Интегрированная среда разработки и отладки Cicode-программ (ИСР, **Редактор Cicode**) может быть активизирована различными способами. Можно выполнить команду **Средства | Редактор Cicode** или нажать на панели инструментов кнопку **Редактор Cicode**, причем это можно выполнить в любом из приложений **Проводник Citect**, **Редактор проектов Citect** или **Построитель графики Citect**. И, наконец, наиболее удобный способ, заключается в использовании приложения **Проводник Citect**, в котором следует выбрать проект (например, **CicodeDebug**), открыть папку **Файлы Cicode** и "кликнуть" дважды левой кнопкой мыши по требуемому Cicode-файлу (рис. 22.1, например, по файлу Cicode1.ci).

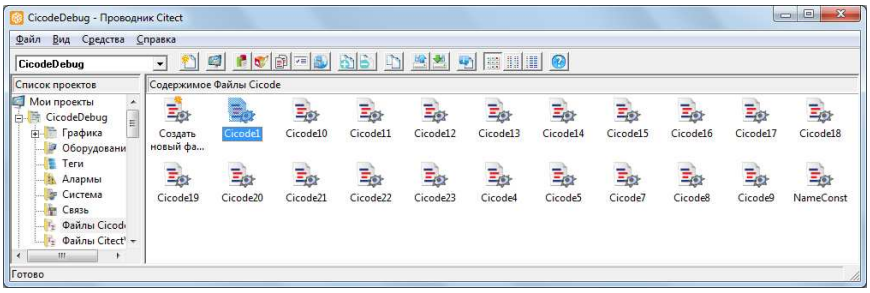


Рис. 22.1. Запуск ИСР Cicode-программ

Возможный вид ИСР показан на рис. 22.2. Строка заголовка, строка меню и окно редактирования видимы всегда, а строку состояния, остальные окна и панели инструментов можно скрывать или показывать по желанию пользователя.

Так переключение *строки состояния* выполняется командой **Вид | Строка состояния**. Если ни одна из панелей инструментов не отображается, то для конфигурирования панелей инструментов и окон следует выполнить команду **Отладка | Настройки…** или активизировать акселератор **Ctrl+T**. В появившемся диалоге **Настройки…** выбрать вкладку **Окна и панели**, установить флажки для тех панелей инструментов и окон, которые требуется сделать видимыми и нажать кнопку **OK** (рис. 22.3).

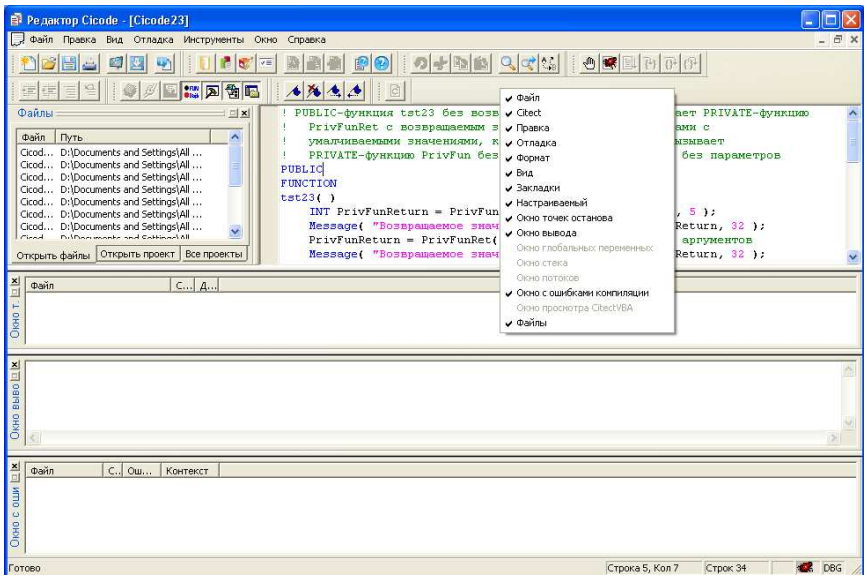


Рис. 22.2. Возможный вид ИСР Cicode-программ

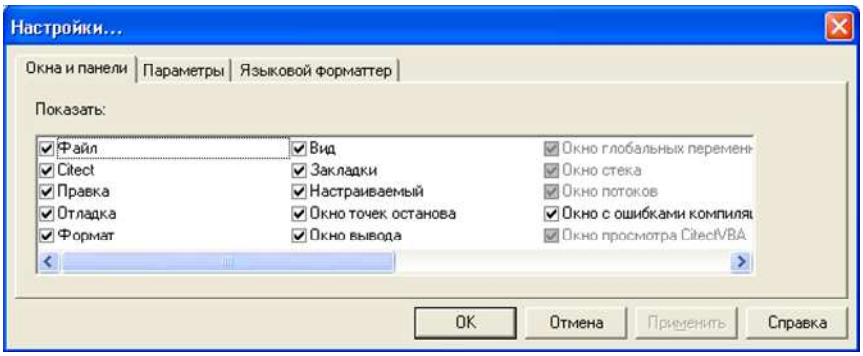


Рис. 22.3. Конфигурирование панелей инструментов и окон ИСР

Если же хотя бы одна из панелей инструментов отображается, то удобнее поместить указатель мыши на свободное место рядом с панелью инструментов и правой кнопкой мыши вызвать контекстное меню (см. приведенный ранее рис. 22.2). Далее установить флажки для тех панелей инструментов и окон, которые требуется сделать видимыми и "кликнуть" левой кнопкой мыши по любому месту вне контекстного меню.

22.1. Основные приемы работы в ИСР Редактор Cicode

ИСР можно использовать для ввода, редактирования и отладки Cicode-программы. Текстовый редактор ИСР аналогичен таким средствам редактирования кода, как в среде Microsoft Visual Studio. Он содержит соответствующие интеллектуальные средства — панели инструментов и окна, синтаксическую подсветку текста, автоотступы и ступенчатую запись, всплывающие окна с прототипами функций, автоматическое исправление ключевых слов, поддержку работы с клавиатурными комбинациями и т. п.

22.1.1. Изменение умолчания для текстового редактора ИСР

ИСР позволяет вместо встроенного и используемого по умолчанию текстового редактора Cicode, использовать любой текстовый редактор, поддерживаемый операционными системами (ОС) Windows.

Для задания другого текстового редактора следует выполнить команду **Средства | Редактор проектов** или нажать кнопку **Редактор проектов** на панели инструментов **Citect**. В появившейся среде приложения **Редактор проектов Citect** выполните команду **Средства | Параметры** и появится диалог **Параметры** (рис. 22.4).

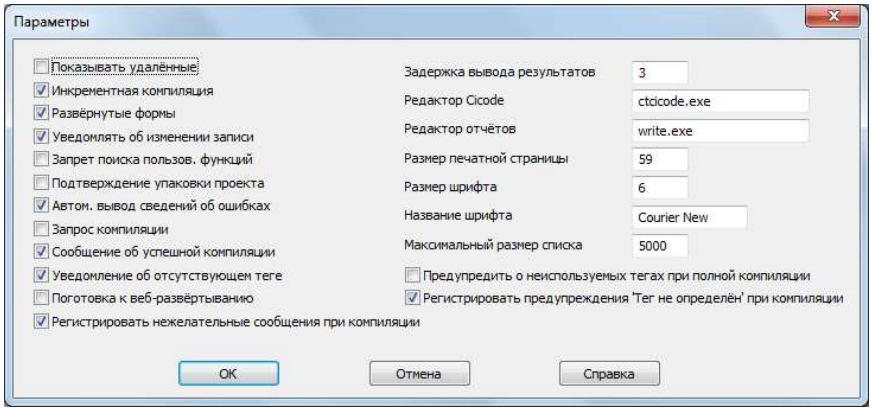


Рис. 22.4. Выбор текстового редактора для ИСР

Введите в поле **Редактор Cicode** имя приложения для текстового редактора (по умолчанию **ctcicode.exe**, для других текстовых редакторов **имя.расширение** для приложения следует вводить, указывая полный путь) и нажмите кнопку **OK** для принятия изменения или кнопку **Отмена** для отказа от изменения.

22.1.2. Создание и сохранение Cicode-файла

Для создания Cicode-файла целесообразно выполнить следующие действия. Перейдите в среду приложения **Проводник Citect**, в поле **Список проектов** выберите проект, в который требуется включить создаваемый файл (например, **CicodeDebug**), откройте папку проекта, выберите папку **Файлы Cicode** и в поле **Содержимое Файлы Cicode** дважды "кликните" левой кнопкой мыши по пиктограмме с надписью **Создать новую файл Cicode**. В окне редактирования ИСР введите текст файла (о структуре и оформлении файла см. приведенную ранее гл. 21). Для сохранения введенного файла выполните команду **Файл | Сохранить как…**. В появившемся диалоге **Сохранить как** в поле **Имя файла:** введите требуемое имя файла и нажмите кнопку **Сохранить** (рис. 22.5).

В результате введенный файл Cicode1.ci появится в папке проекта ..\Schneider Applications \ User \ CicodeDebug.

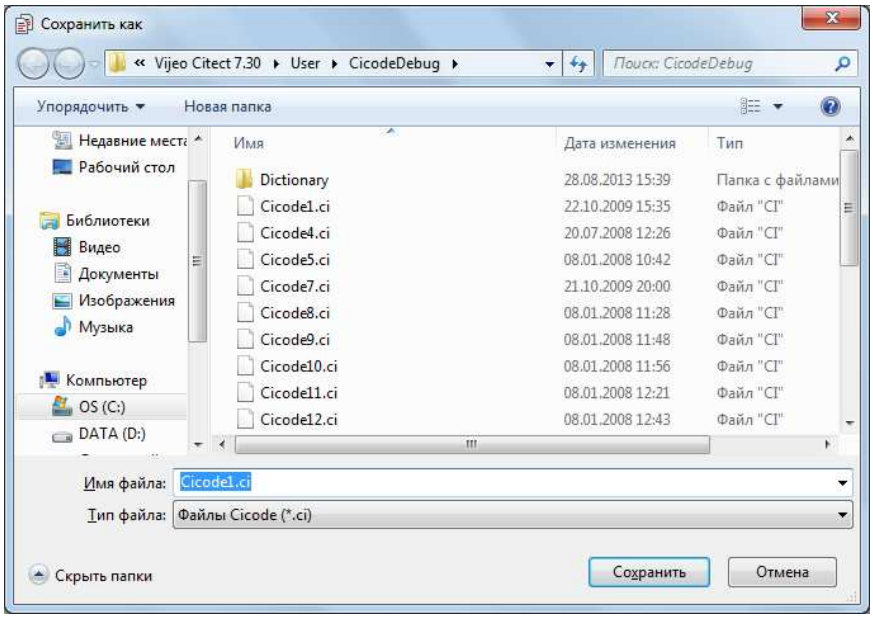


Рис. 22.5. Сохранение введенного файла

Примечание

После добавления в проект нового файла ИСР необходимо перезагрузить — иначе во вкладке **Открыть проект** окна **Файлы** добавленный файл не будет отображаться.

22.1.3. Открытие существующего Cicode-файла

Для открытия существующего Cicode-файла целесообразно выполнить следующие действия. Перейдите в среду приложения **Проводник Citect**, в поле **Список проектов** выберите проект, в котором имеется файл, подлежащий открытию (например, **CicodeDebug**), откройте папку проекта, выберите папку **Файлы Cicode** и в поле **Содержимое Файлы Cicode** дважды "кликните" левой кнопкой мыши по пиктограмме с надписью, содержащей **имя** открываемого файла (например, по файлу **Cicode1**, см. приведенный ранее рис. 22.1). В окне редактирования ИСР появится текст открытого файла.

22.1.4. Удаление существующего Cicode-файла

Для удаления существующего Cicode-файла целесообразно выполнить следующие действия. В ИСР выполните команду **Файл | Открыть...** или нажмите кнопку **Открыть** на панели инструментов **Файл**, если она не скрыта. В появившемся диалоге **Открыть** выберите файл, подлежащий удалению, и нажмите кнопку **Delete** на клавиатуре компьютера. В появившемся диалоге **Подтверждение удаления файла** нажмите кнопку **Да**. Нажмите кнопку **Отмена** для закрытия диалога **Открыть**. В окне **Файлы** ИСР выберите удаленный файл и выполните команду **Файл | Закрыть**.

22.1.5. Поиск текста в Cicode-файле

Для поиска текста в Cicode-файле целесообразно выполнить следующие действия. В ИСР выполните команду **Правка | Найти…**, или активизируйте акселератор **Ctrl+F**, или нажмите кнопку **Найти** на панели инструментов **Правка**, если она не скрыта. В появившемся диалоге **Поиск** (рис. 22.6) в поле **Найти:** введите искомый текст (например, **FUNCTION**) и задайте нужное состояние флагов поиска только слов (**Только целые слова**) и поиска с учетом регистра (**Учитывать регистр**). При нажатии кнопки **Найти далее** произойдет поиск текста, начиная с текущей позиции курсора, и если искомый фрагмент текста будет обнаружен, то он будет выделен (рис. 22.7). Для поиска следующего фрагмента текста достаточно на клавиатуре компьютера нажать клавишу **F3**.

При нажатии кнопки **Отметить все** произойдет отметка строк файла, содержащих искомый текст (рис. 22.8). При необходимости снятия отметки строк нажмите кнопку **Убрать закладки** на панели инструментов **Закладки**. При нажатии кнопки **Отмена** поиск отменяется.

Рис. 22.6. Поиск текста в файле

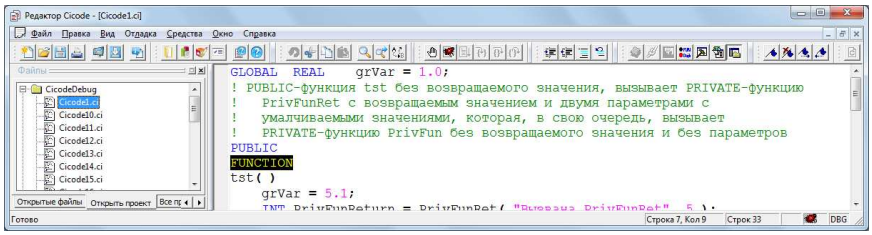


Рис. 22.7. Результаты поиска заданного текста

22.1.6. Компиляция Cicode-файла и просмотр информации об ошибках

Для компиляции Cicode-файла сделайте видимым окно **Окно с ошибками компиляции** (команда **Вид | Ошибки компиляции**), выполните команду **Файл | Компилировать** или активизируйте акселератор **Alt+F10**. При наличии ошибок, информация о них может иметь вид, представленный на рис. 22.9 и 22.10.

22.1.7. Режимы ИСР

Настройка режимов отладки выполняется с помощью диалога **Настройки…** (см. приведенный ранее рис. 22.3), который появляется при выполнении команды **Отладка | Настройки…**. С помощью этого диалога можно, в частности, определять или изменять поведение ИСР при возникновении ошибок, определять, когда отладка должна начаться и поведение отладчика в режиме отладки.

Диалог **Настройки…** имеет три вкладки для задания свойств ИСР — вкладка **Окна и панели** (см. приведенный ранее рис. 22.3), о которой говорилось ранее, вкладка **Параметры** (рис. 22.11) и вкладка **Язык форматирования** (рис. 22.12).

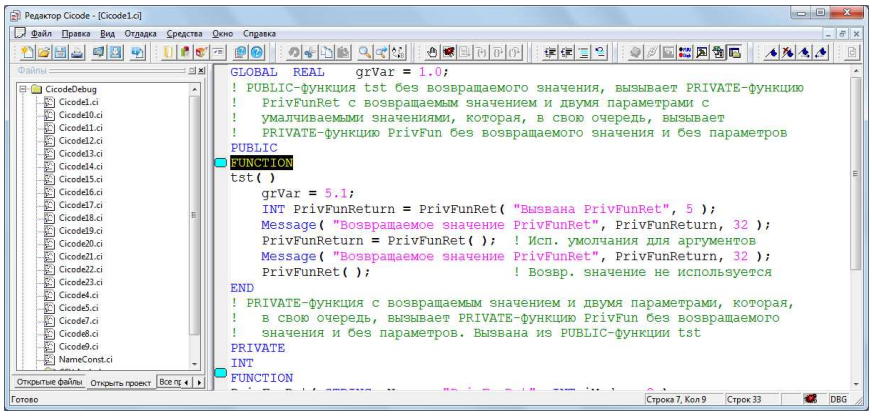


Рис. 22.8. Отметка строк с заданным текстом

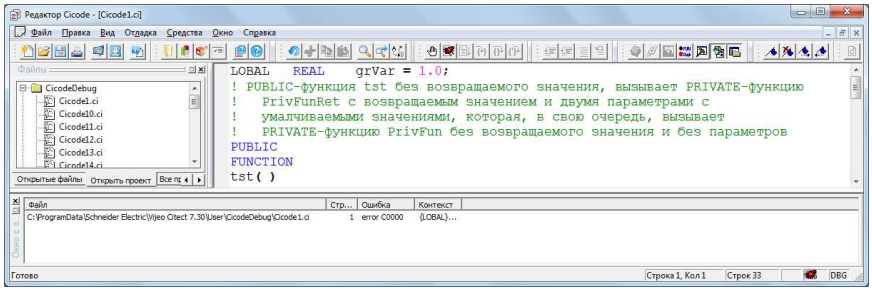


Рис. 22.9. Информация об ошибках компиляции в ИСР

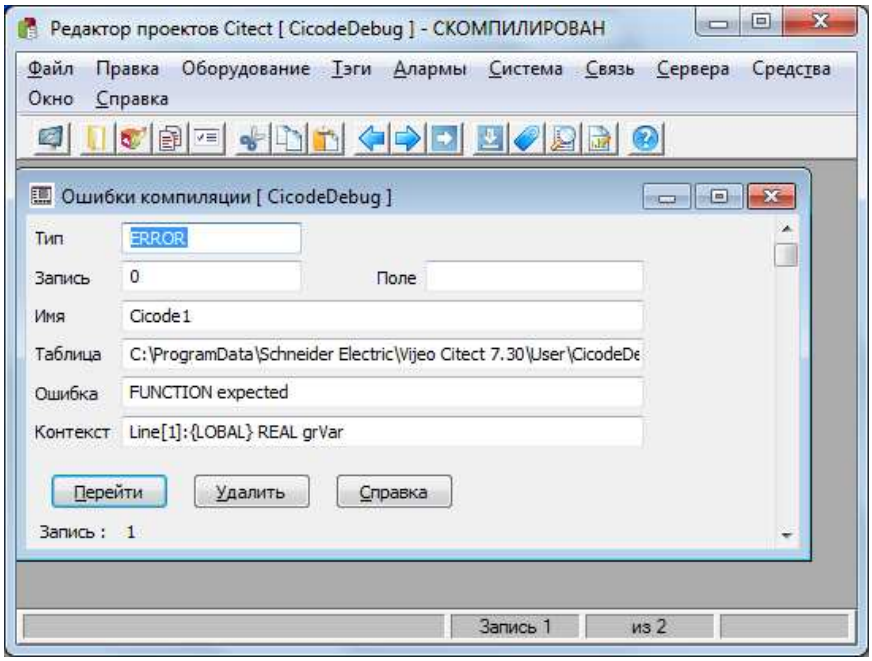


Рис. 22.10. Информация об ошибках компиляции в **Citect Project Editor**

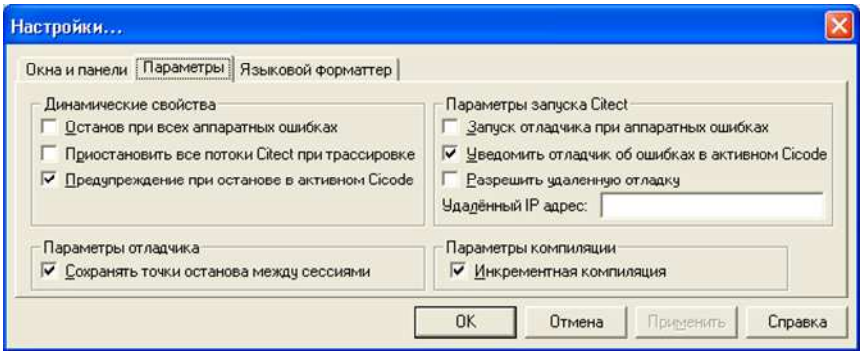


Рис. 22.11. Конфигурирование режимов ИСР

22.1.8. Размещение окон и панелей инструментов ИСР

Окна просмотра и панели инструментов ИСР могут быть либо "приклеены" к любой стороне фрейма главного окна ИСР, либо размещены в виде плавающего окна в произвольном месте экрана.

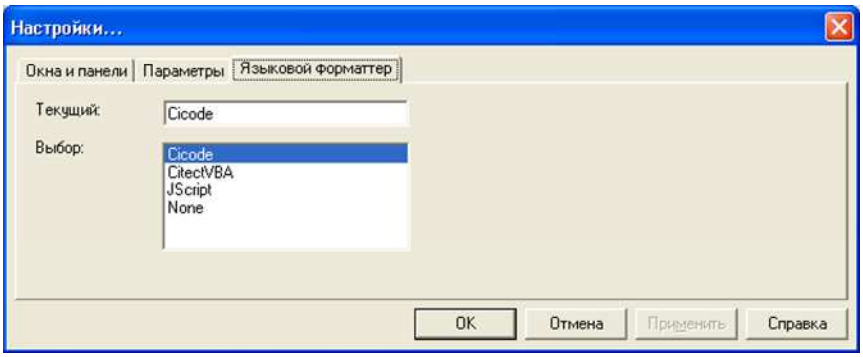


Рис. 22.12. Конфигурирование режимов форматирования ИСР

По умолчанию панели инструментов размещаются в предназначенной для них области в верхней части главного окна ИСР, окна просмотра располагаются под ними: окно **Файлы** — слева, окно редактирования — справа, а остальные окна просмотра ниже их (см. приведенный ранее рис. 22.2).

Размеры "приклеенных" (docked) окон автоматически выбираются так, чтобы они полностью размещались рядом друг с другом в пределах главного окна ИСР и не перекрывали друг друга.

Панели инструментов и окна просмотра можно, по желанию пользователя, размещать в любом месте экрана, в том числе и вне области, занятой ИСР. Чтобы панель инструментов или окно просмотра перевести в "плавающий" (floating) режим и разместить в произвольном месте экрана, достаточно поместить указатель мыши на заголовок панели инструментов или окна просмотра (рис. 22.13), нажать левую клавишу мыши, перемещая мышь, выбрать новое положение панели инструментов или окна просмотра и отпустить клавишу мыши (рис. 22.14).

Аналогичным образом можно панель инструментов или окно просмотра вернуть в прежнее положение (состояние Docking).

22.1.9. Назначение и использование панелей инструментов ИСР

ИСР предусматривает использование восьми панелей инструментов — **Файл**, **Citect**, **Правка, Отладка, Формат, Вид, Закладки** и **Настраиваемый**. Каждая из панелей инструментов, по желанию пользователя, может быть показана или скрыта.

Использование кнопок панелей инструментов, по сравнению с аналогичными командами меню, позволяет более оперативно выполнить требуемое действие. Назначение кнопок панелей инструментов легко определить с помощью всплывающих подсказок. Для получения всплывающей подсказки достаточно поместить курсор мыши на пиктограмму кнопки. Всплывающее окно содержит текст о назначении кнопки, который отображается в течение нескольких секунд.

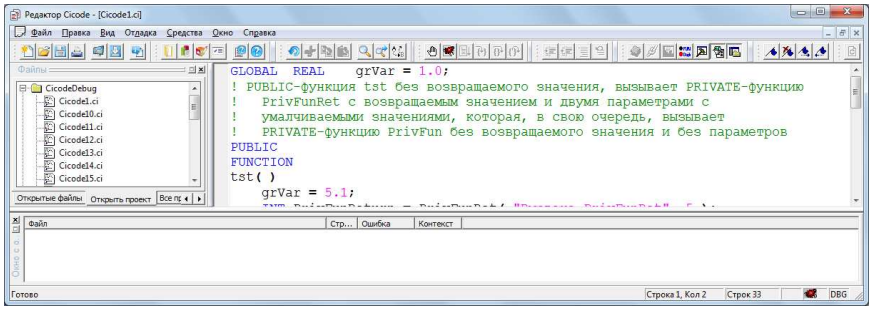


Рис. 22.13. Местоположение заголовков панелей инструментов и окон просмотра

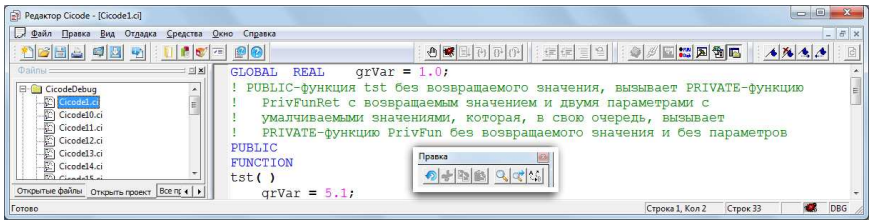


Рис. 22.14. Панель инструментов **Правка** в "плавающем" состоянии

Панель инструментов Файл предназначена для оперативного выполнения наиболее часто выполняемых файловых операций. Панель содержит расположенные слева-направо кнопки **Новый** (создать новый файл), **Открыть** (открыть существующий файл), **Сохранить** (сохранить текущий, активный файл), **Печать** (напечатать текущий файл), **Компилировать и запустить** (компилировать и запустить проект), **Компилировать** (компилировать проект) и **Предпочтения** (посмотреть или изменить свойства отображаемых файлов). Команды, выполняющие аналогичные действия, имеются в меню **Файл** и **Вид**, но их использование менее оперативно.

Панель инструментов Citect предназначена для оперативного запуска приложений системы Vijeo Citect и получения справочной информации. Панель содержит расположенные слева-направо кнопки **Проводник Citect** (переход в среду

проводника проектов), **Редактор проектов** (переход в среду редактора проектов), **Построитель графики** (переход в среду графического редактора), **Мастер конфигурирования компьютера** (запуск мастера настройки компьютера), **Справка по функциям Cicode** и **Разделы справки** (получение справки). Команды, выполняющие аналогичные действия, имеются в меню **Средства** и **Справка**, но их использование менее оперативно.

Панель инструментов Правка предназначена для оперативного редактирования текущего файла. Панель содержит расположенные слева-направо кнопки **Откатить** (отменить последнюю операцию редактирования), **Вырезать** (вырезать выделенный текст и поместить его в буфер промежуточного хранения), **Копировать** (копировать выделенный текст в буфер промежуточного хранения), **Вставить** (вставить текст из буфера промежуточного хранения в место, отмеченное курсором), **Найти** (найти заданный фрагмент текста в текущем файле), **Найти далее** (найти следующий фрагмент текста в текущем файле) и **Заменить** (заменить заданный фрагмент текста новым фрагментом). Команды, выполняющие аналогичные действия, имеются в меню **Правка**.

Панель инструментов Отладка предназначена для отладки текущего файла. Панель содержит расположенные слева-направо кнопки **Поставить/Убрать точку останова** (переключить состояние точки останова), **Запустить/Остановить отладку** (переключить состояние режима отладки), **Продолжить** (продолжить выполнение с текущей точки), **Шагнуть в** (выполнить текущую строку, если она не содержит вызов функции или, иначе, войти внутрь функции), **Шагнуть через** (выполнить текущую строку, даже если она содержит вызов функции) и **Выйти** (выйти из текущей функции). Команды, выполняющие аналогичные действия, имеются в меню **Отладка**.

Панель инструментов Формат предназначена для форматирования и комментирования текущего файла. Панель содержит расположенные слева-направо кнопки **Отступ** (сдвинуть строки с выделенным текстом на табулятор вправо), **Выступ** (сдвинуть строки с выделенным текстом на табулятор влево), **Комментировать** (закомментировать строки с выделенным текстом) и **Раскомментировать** (раскомментировать строки с выделенным текстом). *Команды, выполняющие аналогичные действия, в меню отсутствуют.*

Панель инструментов Вид предназначена для переключения окон просмотра. Панель содержит расположенные слева-направо кнопки **Показать/Скрыть стек** (переключить **Окно стека**), **Показать/Скрыть потоки** (переключить

Окно потоков), **Показать/Скрыть глобальные переменные** (переключить **Окно глобальных переменных**), **Показать /Скрыть точки останова** (переключить **Окно точек останова**), **Показать/Скрыть вывод** (переключить **Окно вывода**), **Показать/Скрыть ошибки компиляции** (переключить **Окно с ошибками компиляции**) и **Показать/Скрыть файлы** (переключить **Окно файлов**). Команды, выполняющие аналогичные действия, имеются в меню **Вид**.

Панель инструментов Закладки предназначена для работы с отмеченными строками (см. приведенный ранее рис. 22.8). Панель содержит расположенные слева-направо кнопки **Поставить/Убрать закладку** (переключить отметку текущей строки), **Убрать закладки** (снять отметку всех строк), **Следующая закладка** (перейти на следующую отмеченную строку), и **Предыдущая закладка** (перейти на предыдущую отмеченную строку). Команды, выполняющие аналогичные действия, имеются в меню **Правка**.

22.1.10. Назначение и использование окон просмотра ИСР

ИСР предусматривает использование семи окон просмотра — **Окно точек останова**, **Окно вывода**, **Окно глобальных переменных**, **Окно стека**, **Окно потоков**, **Окно с ошибками компиляции** и окно **Файлы**.

Окно точек останова. Это окно используется для отображения списка всех точек останова, имеющихся в данный момент в проекте (рис. 22.15). Для любой из строк окна отображения точек останова с помощью правой кнопки мыши можно вызвать контекстное меню (рис. 22.16), с помощью которого точку останова можно удалить или переключить ее состояние. Если в **Окне точек останова** выбрать какую-либо строку и дважды "кликнуть" левой кнопкой мыши, то в окне редактирования отобразится соответствующий файл с выделенной строкой, соответствующей точке останова. В столбце **Файл** окна **Точек останова** отображается полный путь, имя и расширение файла, содержащего точку останова. В столбце **Строка** отображается номер строки файла, отмеченной как точка останова, а в столбце **Доступна** — указывается, включена точка останова (**Yes**) или нет (**No**).

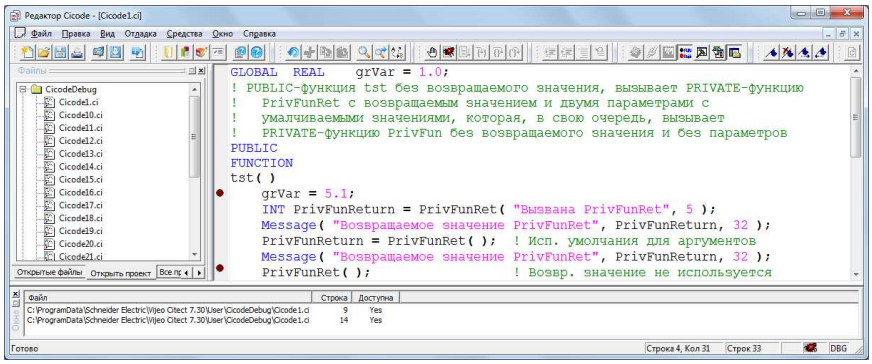


Рис. 22.15. Окно отображения точек останова проекта **CicodeDebug**

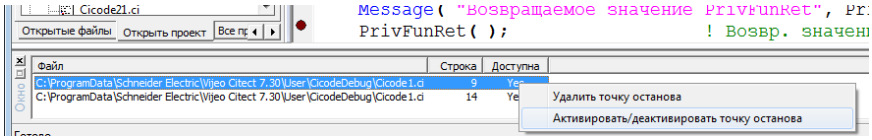


Рис. 22.16. Контекстное меню точки останова **Окна точек останова**

Окно вывода. Это окно используется для отображения списка сообщений, посылаемых CitectSCADA в *процессе отладки* (рис. 22.17). В нем размещаются сведения о начале и завершения потоков в том порядке, в котором потоки активизируются, а также сведения о прерываниях. **Окно вывода** отображает сообщения, посылаемые функцией `TraceMsg( )`.

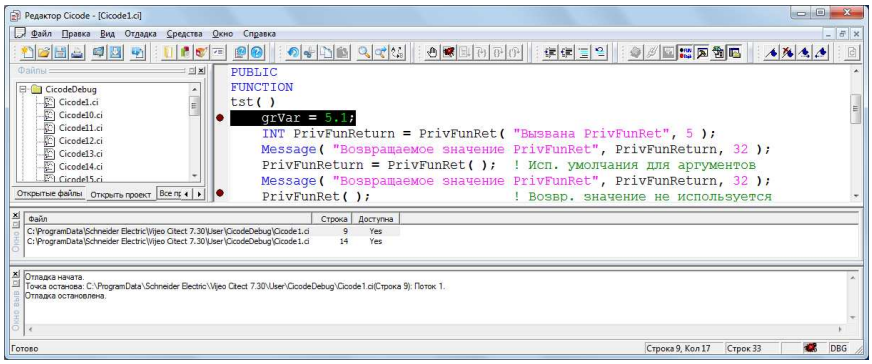


Рис. 22.17. **Окно вывода** ИСР

Замечание

Имейте в виду, что сообщения в **Окно вывода** выдаются, только если включен режим отладки (например, при выполнении команды **Отладка | Запустить отладку** или при активизации акселератора **Shift+F5**).

Окно глобальных переменных. Это окно используется для отображения списка значений глобальных и модульных переменных. Глобальная или модульная переменная добавляется в список *только при включенном режиме отладки после первого присвоения ей значения* (рис. 22.18).

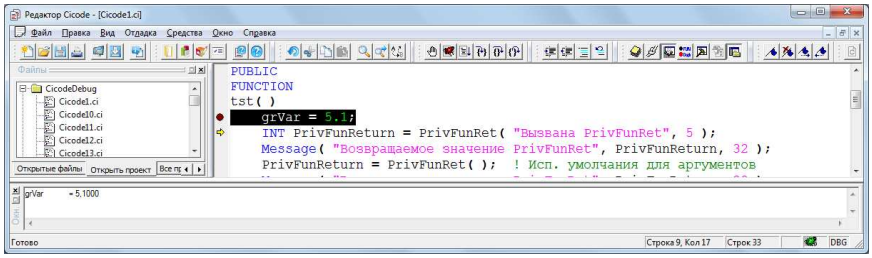


Рис. 22.18. Окно глобальных переменных ИСР

При последующем изменении значения глобальной переменной новое значение также обновляется в соответствующей строке списка.

Окно стека. Это окно используется для отображения списка значений из стека для текущего потока. Стек содержит вызовы функций с указанием значений их аргументов, значения всех переменных, используемых в функциях и возвращаемые функциями значения (рис. 22.19). **Окно стека** показывается и обновляется *только при включенном режиме отладки (построчная трассировка).*

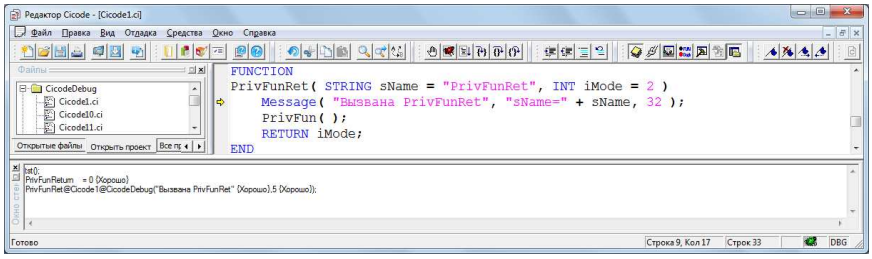


Рис. 22.19. **Окно стека** ИСР

Окно потоков. Это окно имеет вид, представленный на рис. 22.20.

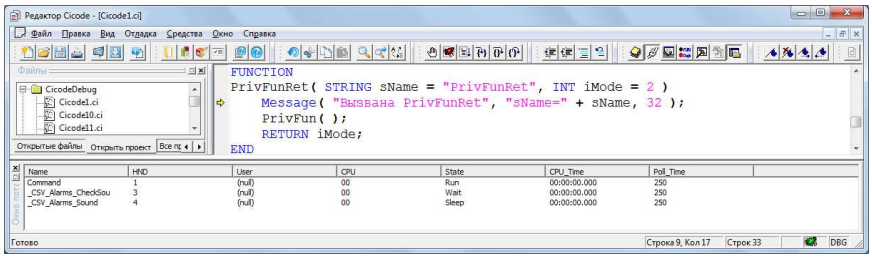


Рис. 22.20. **Окно потоков** ИСР

Окно потоков показывается и обновляется *только при включенном режиме отладки*. Окно имеет следующие столбцы. Столбец **Name** отображает имя потока — имя функции, запустившей поток (эта функция может быть вызвана, например, из функции **`TaskNew( )`**). Если вы "кликните" левой кнопкой мыши по имени потока, то переведете фокус отладчика на этот поток и отладчик отобразит источник этого потока.

> ***Примечание***
> Если поток не был запущен из функции `TaskNew( )`, то имя в столбце **Name** отобразится как **Command**.

Столбец **HND** отображает дескриптор потока. Столбец **CPU** отображает процент загрузки потоком центрального процессора. Код Cicode эффективен, если это значение не превышает 25%. Большие значения могут означать, что в циклах следует использовать функцию **`Sleep( )`**. Столбец **State** отображает состояние потока — состояние **Ready** означает готовность потока к запуску, состояние **Sleep** означает приостанов потока с помощью функции **`Sleep( )`**, а состояние **Run** означает выполнение потока. Столбец **CPU_Time** отображает суммарное время центрального процессора, использованное потоком.

Окно с ошибками компиляции. Это окно имеет вид, представленный ранее на рис. 22.9. Окно отображает список ошибок, выявленных в процессе компиляции. Если "кликнуть"два раза левой кнопкой мыши по строке списка ошибок, то в окне редактирования отобразится файл, содержащий ошибку, в котором будет выделена ошибочная строка.

Окно файлов. Это окно имеет вид, представленный ранее на рис. 22.2, и содержит три вкладки. Вкладка **Все проекты** отображает иерархическое дерево всех проектов и их Cicode- и VBA-файлов. Вкладка **Открыть проект** отображает иерархическое дерево текущего выбранного проекта и всех включенных проектов. Вкладка **Открытые файлы** отображает названия всех файлов, открытых в настоящее время для редактирования. Щелчок по любому из этих файлов приведет к его загрузке в окно редактирования и сделает его активным.

22.2. Отладка фрагментов и функций Cicode-программы

Чтобы определить местонахождение ошибок Cicode, следует в ИСР включить режим отладки. Для этого достаточно в ИСР выполнить команду **Отладка | Запустить отладку**, или активизировать акселератор **Shift+F5**, или нажать кнопку **Запустить/Остановить отладку** на панели инструментов **Отладка**, если она не скрыта.

> ***Замечание***
> Если при переходе в режим отладки текущий проект не выполнялся, то автоматически выполняется компиляция и запуск проекта. В правом нижнем углу окна ИСР пиктограмма режима отладки имеет зеленый цвет, а при выключенном режиме отладки — красный (см. приведенный ранее рис. 22.20).

Аналогично, для выключения режима отладки следует в ИСР выполнить команду **Отладка | Остановить отладку**, или активизировать акселератор **Shift+F5**, или нажать кнопку **Запустить/Остановить отладку** на панели инструментов **Отладка**, если она не скрыта.

Для отладки Cicode-фрагмента или Cicode-функции можно применить, например, следующий прием. В какую-либо графическую страницу текущего проекта (например, в графическую страницу **Debug** проекта **CicodeDebug**) для целей отладки можно поместить кнопку (рис. 22.21) и сконфигурировать ее в соответствии с рис. 22.22.

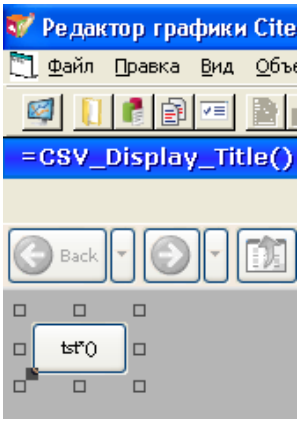


Рис. 22.21. Кнопка, используемая для отладки Cicode-фрагментов и функций

Теперь отлаживаемый Cicode-фрагмент достаточно поместить для отладки в тело глобальной функции **tst( )** и установить точку останова на первом же операторе в теле функции. Аналогично, для отладки другой функции достаточно в свойствах

кнопки заменить имя функции требуемым именем, определение функции поместить в Cicode-файл проекта **Cicode*.ci** (см., например, приведенный ранее рис. 22.20) и, как и в предыдущем случае, установить точку останова на первом же операторе в теле функции.

> ***Совет***
> Перед включением в ИСР режима отладки не забудьте выполнить компиляцию и устранить возможные синтаксические ошибки. Для запуска компиляции можно либо выполнить команду **Файл | Компилировать**, либо активизировать акселератор **Alt+F10**, либо нажать кнопку **Компилировать** на панели инструментов **Файл**, если она не скрыта.

Если теперь включить режим отладки, то проект автоматически запустится. Для выполнения отладки функции **`tst( )`** теперь достаточно нажать кнопку **tst*()**, добавленную в графическую страницу **Debug** проекта **CicodeDebug** для целей отладки и функция **`tst( )`** будет выполнена до точки останова.

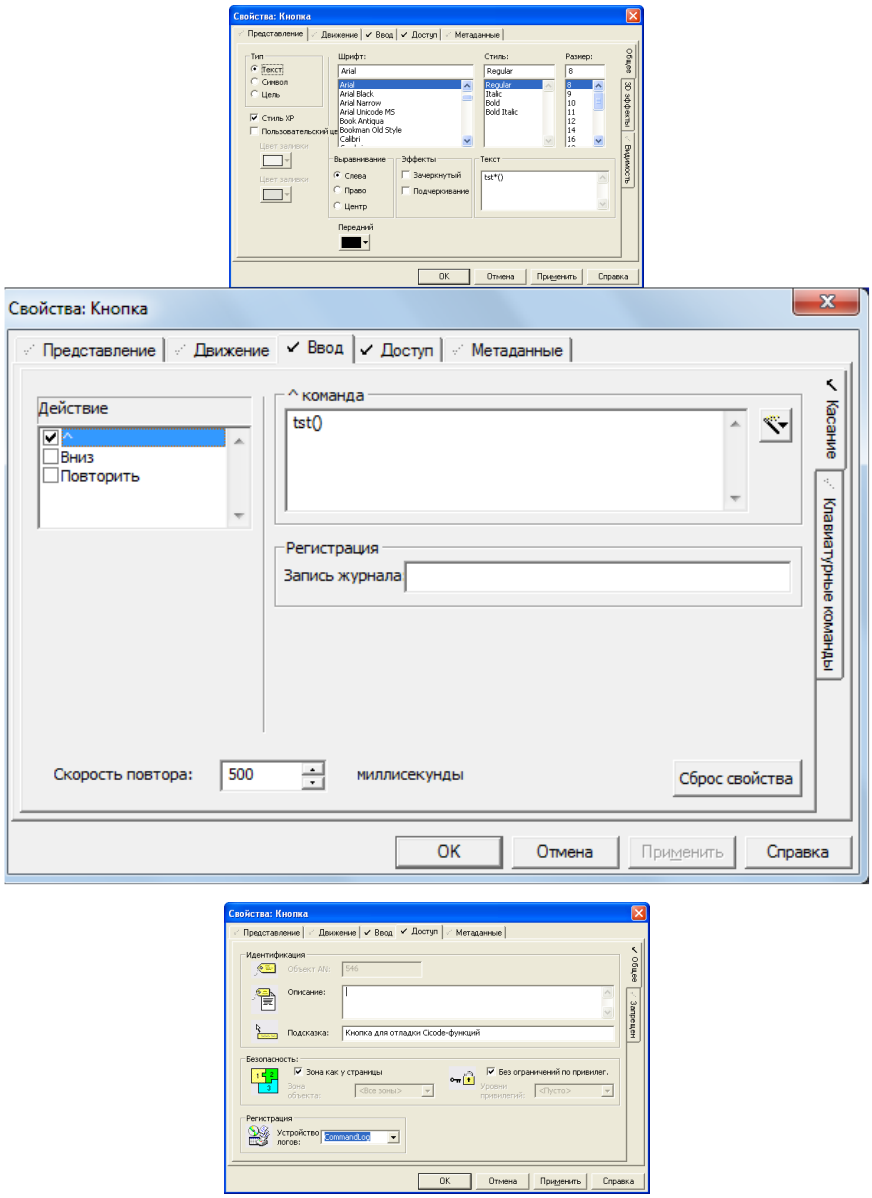


Рис. 22.22. Свойства кнопки, используемой для отладки Cicode-фрагментов и функций

Далее можно выполнить построчную трассировку функции, контролируя результаты, полученные после выполнения очередной строки. Для трассировки, как было указано ранее, можно использовать панель инструментов **Отладка** или команды меню **Отладка**. Для контроля значений глобальных и модульных переменных в точках останова можно использовать **Окно глобальных переменных**, а изменения значений локальных переменных функции можно наблюдать в **Окне стека**. Именно таким образом тестировались все приведенные ранее фрагменты и функции Cicode.

Глава 23. Использование Cicode-файлов, Cicode-команд и Cicode-функций в системе Vijeo Citect

Cicode — язык программирования, специально разработанный в системе Vijeo Citect для мониторинга и управления оборудованием завода. Используя язык Cicode, вы можете получить доступ в реальном времени ко всем данным (переменным) проекта Vijeo Citect и всем средствам обслуживания Vijeo Citect, таким, как теги переменных, сигналы тревог, тренды, отчеты и т. п. Вы можете использовать язык Cicode в качестве интерфейсного средства для объединения различных средств обслуживания компьютера, например операционной системы и коммуникационных портов. Язык Cicode поддерживает также ряд дополнительных средств.

23.1. Использование Cicode-файлов

Вы можете помещать определения функций в Cicode-файлах, сохраняемых на магнитном диске, и имеющих расширение `.ci`. Для минимизации потенциальных будущих проблем поддержания Cicode-файлов следует придерживаться Cicode-стандарта программирования (см. подробнее в справочной системе, тема Cicode Programming Reference | Using Cicode Programming Standards). О структуре и оформлении Cicode-файлов было сказано ранее в гл. 21.

Когда выполняется компиляция проекта системы Vijeo Citect, компилятор обрабатывает также все функции, глобальные и модульные переменные, определенные в файлах Cicode. В результате определенные в этих файлах пользовательские функции становятся доступными наравне с предопределенными (стандартными) функциями языка. Можно использовать столько файлов Cicode, сколько необходимо. Файлы Cicode размещаются в той же самой папке, что и файлы проекта. Когда выполняется сохранение проекта, то сохраняются и все исходные файлы Cicode.

23.2. Использование Cicode-команд

Команды языка Cicode расширяют возможности управляющих элементов в системе мониторинга и управления Vijeo Citect. Команды используются для управления системой Vijeo Citect и, тем самым, для управления производственным процессом.

Каждая команда имеет механизм для ее активации. Команда может быть запущена вручную, оператором, вводящим ключевую последовательность или нажимающим кнопку или объект на графической странице. Можно также так сконфигурировать команду, чтобы автоматически выполнить ее, когда оператор регистрируется в процессе работы или выполняет противоположное действие, когда графическая

страница открывается или закрывается, когда возникает сигнал тревоги или когда возникает некоторое событие.

Чтобы определить команду языка Cicode, следует ввести выражение, оператор или группу операторов, разделенных точкой с запятой в поле **^ команда** вкладки **Ввод (Касание)** соответствующего графического объекта (рис. 23.1).

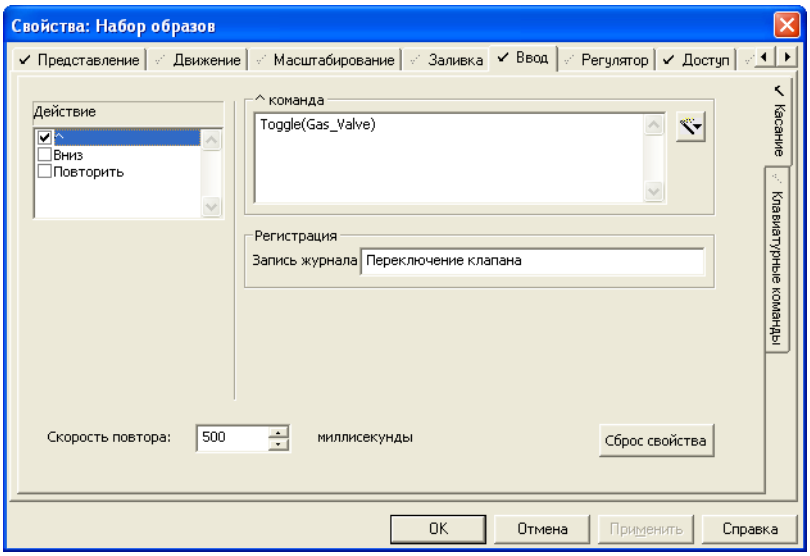


Рис. 23.1. Пример определения команды языка Cicode графического объекта **Набор образов**

Каждое выражение или оператор команды обычно выполняет отдельную задачу, например, присваивание переменной значения, вычисление значения или отображение сообщения на экране. Если нужно оценить состояние производственного процесса вместо управления им, в поле **Команда** следует записать соответствующее выражение или его частную разновидность, например, переменную. Можно задать значение переменной в поле **Команда**, в поле **Выражение** или в вызываемой функции Cicode путем использования оператора присваивания:

```
<VAR_TAG> = Val;
```

Здесь `<VAR_TAG>` является именем переменной, которой присваивается новое значение, а `Val` задает новое значение.

> ***Замечание***
> В качестве `Val` можно использовать не только константу, но и переменную и даже выражение.

В поле **Команда** графического объекта можно задавать не один, а несколько операторов, например, как это показано на рис. 23.2.

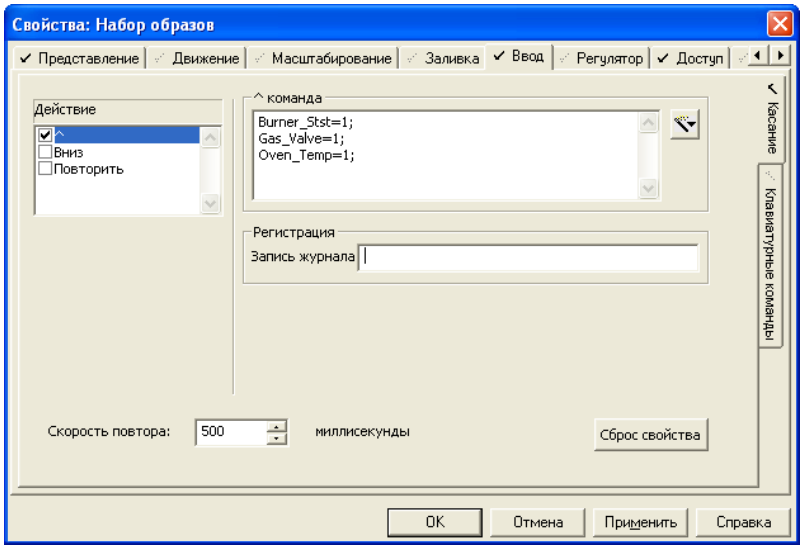


Рис. 23.2. Пример задания нескольких операторов в поле **Команда** (кнопка **tst*()**, графическая страница **Debug** проекта **CicodeDebug**)

В приведенном примере в поле **Команда** использованы три оператора, разделенные точкой с запятой. Операторы представляют собой операторы присваивания тегам переменных требуемых значений. В процессе работы проекта **CicodeDebug** при нажатии кнопки **tst*()**, расположенной на графической странице **Debug**, последовательно выполняются указанные в поле **Команда** операторы присваивания.

> ***Указание***
> Обязательно разделяйте операторы точкой с запятой. При невыполнении этого требования нельзя правильно обнаружить конец оператора, что приведет к ошибке компиляции. После последнего оператора точку с запятой можно опускать.

Число операторов, которые можно ввести в поле **Команда**, ограничено только размерами этого поля (71 строка и 207 символов в строке; в одной строке можно располагать любое число операторов, что определяется длиной строки). Однако, для наглядности, не следует использовать слишком много операторов (в соответствии с "правилом семи" не используйте более семи операторов). Для выполнения этого требования, при необходимости, оформляйте эти операторы в виде функции (функций) и в поле **Команда** помещайте только вызов функции (функций). ***Другой альтернативой является использование включаемых файлов, содержащих необходимые операторы, о чем будет сказано далее.***

Как указывалось ранее, количество операторов и выражений, которые можно ввести в поле **Команда** или **Выражение**, ограничивается перечисленными в предыдущем абзаце размерами поля. Если размеры поля недостаточны, то можно использовать включаемый файл с расширением **.cii** и поместить в него выражения и операторы Cicode-команды. Включаемый файл представляет собой обычный текстовый ASCII-файл, который можно создать с помощью текстового редактора, встроенного в ИСР Cicode-программ. Как и файлы с расширением **.ci**, включаемые файлы хранятся в папке проекта и сохраняются при сохранении проекта.

> ***Указание***
> Имейте в виду, что во включаемый файл нельзя помещать вызовы функций. При использовании в поле **Команда** включаемого файла нельзя одновременно в это поле помещать вызовы функций. Вместе с тем, при отсутствии включаемого файла в поле **Команда** можно одновременно помещать вызовы функций, выражения и операторы. Не путайте включаемые файлы и включаемые проекты — это совершенно разные вещи.

В поле **Команда** имя включаемого файла следует вводить в следующем формате:

```
@<filename.cii>
```

Здесь `filename` задает имя файла с расширением **.cii** по правилам DOS, а использование угловых скобок является обязательным.

> ***Примечание***
> Расширение **.cii** не является обязательным. Если включаемый файл находится вне папки проекта, то следует указывать полный путь к этому файлу.

Пример использования включаемого файла при задании свойств графического объекта иллюстрируют рис. 23.3 и 23.4.

23.3. Работа с обычно используемыми функциями

Язык Cicode содержит множество стандартных функций, реализующих решение разнообразных задач. Многие из них используются при создании комплексных проектов системы Vijeo Citect. Использование некоторого множества стандартных функций рассмотрено ранее. Наиболее часто используемые стандартные функции можно разделить на пять категорий — функции для работы с сигналами тревог (**Alarm Functions**), с графическими страницами (**Page Functions**), с отчетами (**Report Functions**), со временем и датой (**Time/date Functions**) и прочие функции (**Miscellaneous Functions**).

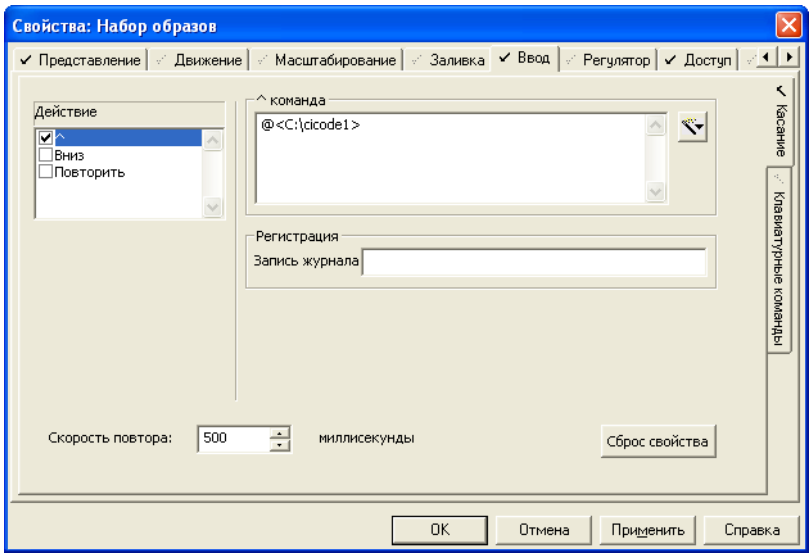


Рис. 23.3. Пример использования включаемого файла в поле **Команда** (кнопка **tst*()**, графическая страница **Debug** проекта **CicodeDebug**)

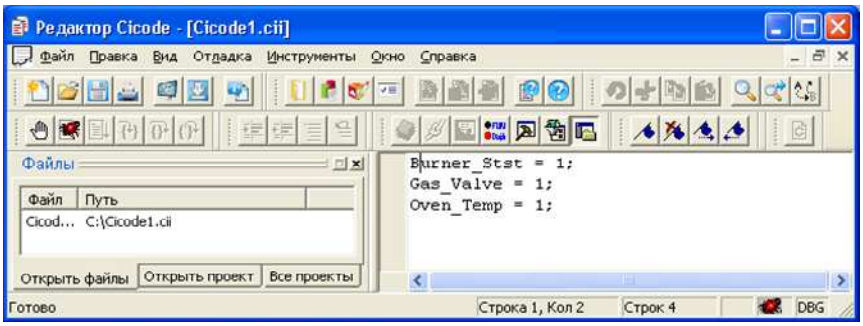


Рис. 23.4. Текст включаемого файла

23.3.1. Функции для работы с сигналами тревог (Alarm Functions)

Вы можете использовать функции работы с сигналами тревог для отображения тревог и связанных с ними страниц помощи, для квитирования (подтверждения), включения и выключения тревог. Вы можете ограничить доступ к каждой команде, которая использует функции для работы с сигналами тревог, тем самым гарантировать, что только оператор с соответствующими правами доступа может

выполнить соответствующие команды. Однако такая возможность существует только в том случае, если ограничение доступа не выполнено на индивидуальные тревоги.

В этой группе, в числе прочих, имеются следующие стандартные функции.

Функция **`AlarmAck( )`** предназначена для квитирования (подтверждения приема) сигнала тревоги. Если команда, запускающая эту функцию, выполнена, то позиционирование курсора на тревогу в списке квитирует ее. Таким образом, можно квитировать не только одну, но и несколько тревог (показать на примере проекта **Training9**, команда **Alarms | Active Alarms** и проекта **TmpltAndMenu**, шаблон графической страницы **mytraining.MyNormal**). Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска AlarmAck. Пример использования функции имеется также в файле **CSV_Alarms.ci** проекта **CSV_Include**.

Функция **`AlarmComment( )`** предназначена для добавления комментария к выбранному элементу списка страницы **Alarm Summary** в процессе работы проекта. Комментарий отображается, когда курсор позиционируется на соответствующий элемент списка сигналов тревог, если ранее была выполнена команда, активизирующая функцию **`AlarmComment( )`**. При этом аргумент функции, задающий текст комментария, вводится с клавиатуры и его длина не может превышать 128 символов. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска AlarmComment. Пример использования функции имеется в файле **Alarms.ci** проекта **CSV_Example**.

Функция **`AlarmDisable( )`** предназначена для отключения сигнала тревоги. Эту функцию можно использовать только привилегированному пользователю для отключения многократно повторяющихся тревог (показать на примере проекта **Training9**, команда **Alarms | Active Alarms**, команда **Disable** контекстного меню для выбранной тревоги). Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска AlarmDisable. Пример использования функции имеется в файле **Alarms.ci** проекта **CSV_Example**.

Функция **`AlarmEnable( )`** предназначена для подключения ранее выключенного сигнала тревоги. Эту функцию можно использовать привилегированному пользователю для подключения многократно повторяющихся тревог (показать на примере проекта **Training9**, команда **Alarms | Disabled Alarms**, команда **Enable** контекстного меню для выбранной тревоги). Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска AlarmEnable.

Функция **`AlarmDsp( )`** предназначена для отображения сигналов тревог, определяемых списком аргументов (показать на примере проекта **TmpltAndMemu**, шаблон графической страницы **mytraining.MyNormal**). Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска AlarmDsp.

Функция **`AlarmActive( )`** предназначена для отображения сигналов тревог, категория тревог определяется списком аргументов. Пример использования функции имеется в проекте **TmpltAndMenu**, шаблон графической страницы **mytraining.MyNormal**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска AlarmActive.

Информацию о других функциях этой группы можно получить в [2], тема Cicode Programming Reference | Cicode Function Categories | Alarm and Alarm Filter Functions.

23.3.2. Функции для работы с графическими страницами (Page Functions)

Вы можете использовать функции работы с графическими страницами (далее — страничные функции) для отображения созданных вами графических страниц и стандартных страниц сигналов тревог.

Функция **`PageInfo( )`** предназначена для получения информации о текущей странице. Вид получаемой информации определяется аргументом функции. Пример использования функции имеется в проекте **TmpltAndMenu**, шаблон графической страницы **mytraining.MyNormal**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска PageInfo.

Функция **`PageAlarm( )`** предназначена для отображения сконфигурированных в проекте категорий активных сигналов тревог в странице тревог (показать на примере проекта **Training9**, проекта **OvenTraining23**, графическая страница **Home** и проекта **TmpltAndMenu**). Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска PageAlarm.

> ***Замечание***
> Для правильной работы функции необходимо предварительно создать страницу **Alarm** на основе шаблона **alarm** для отображения активных тревог (это было реализовано в процессе упражнений, выполненных ранее). Если требуется, чтобы страница имела другое имя, в секцию **[Page]** файла инициализации **Citect.ini** добавить параметр **AlarmPage=MyAlarm**. При этом также необходимо предварительно создать страницу **MyAlarm** на основе шаблона **alarm**.

Функция **`PageDisplay( )`** отображает на экране новую страницу. Имя страницы задается аргументом функции (используйте функцию **`PageLast( )`** для отображения предыдущей страницы). Показать на примере проекта **Training9** и проекта **OvenTraining23**, графическая страница **Home**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска PageDisplay.

Функция **`PageGoto( )`** отображает на экране новую страницу. Имя страницы задается аргументом функции. Функция аналогична функции **`PageDisplay( )`**, но вызов функции **`PageLast( )`** после вызова функции **`PageGoto( )`** не обеспечивает отображения предыдущей страницы. Показать на примере проекта **OvenTraining23**, графическая страница **Home**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска PageGoto.

Функция **`PageHardware( )`** отображает аппаратные сигналы тревог на сконфигурированной в проекте странице аппаратных тревог (показать на примере проекта **Training9** и проекта **OvenTraining23**, графическая страница **Home**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска PageHardware.

Замечание
Для правильной работы функции необходимо предварительно создать страницу **Hardware** на основе шаблона **hardware** для отображения аппаратных тревог (это было реализовано в процессе упражнений, выполненных ранее). Если требуется, чтобы страница имела другое имя, в секцию **[Page]** файла инициализации **Citect.ini** добавить параметр **HardwarePage=MyHardware**. При этом также необходимо предварительно создать страницу **MyHardware** на основе шаблона **hardware**.

Функция **`PageLast( )`** отображает графическую страницу, которая показывалась перед текущей страницей. С ее помощью можно выполнить "откат" на отображение десяти последних страниц. Показать на примере проекта **OvenTraining23**, графическая страница **Home**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска PageLast.

Функция **`PageNext( )`** отображает следующую графическую страницу, определенную в свойстве **След. страница** формы графической страницы. Показать на примере проекта **OvenTraining23**, графическая страница **Home**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска PageNext.

Функция **`PagePrev( )`** отображает графическую страницу, определенную в свойстве **Пред. страница** формы графической страницы. Показать на примере проекта **OvenTraining23**, графическая страница **Home**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска PagePrev.

Функция **`PageSummary( )`** отображает общую информацию о сигналах тревог на странице тревог, сформированной в проекте. Показать на примере проекта **OvenTraining23**, графическая страница **Home**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска PageSummary.

Замечание
Для правильной работы функции необходимо предварительно создать страницу **Summary** на основе шаблона **summary** для отображения суммарных тревог (это было реализовано в процессе упражнений, выполненных ранее). Если требуется, чтобы страница имела другое имя, в секцию **[Page]** файла инициализации **Citect.ini** добавить параметр **SummaryPage=MySummary**. При этом также необходимо предварительно создать страницу **MySummary** на основе шаблона **summary**.

Функция **`PageTrend( )`** отображает стандартную страницу трендов. Показать на примере проекта **OvenTraining23**, графическая страница **Home**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска PageTrend. Обязательно посмотрите эту справочную информацию.

Информацию о других функциях этой группы можно получить в [2], тема Cicode Programming Reference | Cicode Function Categories | Page Functions.

23.3.3. Функции для работы с отчетами (Report Functions)

Функция **`Report( )`** запускает создание отчета на сервере отчетов. Аргументом в вызове этой функции является имя отчета. Показать на примере проекта

OvenTraining23, графическая страница **Home**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска Report.

Информацию о других функциях этой группы можно получить в [2], тема Cicode Programming Reference | Cicode Function Categories | Report Functions.

23.3.4. Функции для работы со временем и датой (Time/date Functions)

Функции для работы со временем и датой возвращают текущее время и дату.

Функция **`Date( )`** возвращает текущую дату в виде строки. Показать на примере проекта **OvenTraining23**, графическая страница **Home**, графический объект **Текст** с надписью **=Date(3)**, свойство **Вид (Отображаемое значение)**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска Date. Обратите внимание на возможные значения аргументов функции.

Функция **`Time( )`** возвращает текущее время в виде строки. Показать на примере проекта **OvenTraining23**, графическая страница **Home**, графический объект **Текст** с надписью **=Time(1)**, свойство **Вид (Отображаемое Значение)**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска Time.

Функция **`SysTime( )`** возвращает внутреннее время системы Vijeo Citect в миллисекундах в виде целого значения. Эту функцию удобно использовать для измерения величины интервала времени между двумя вызовами функции. Показать на примере проекта **OvenTraining23**, графическая страница **Home**, графический объект **Текст** с надписью **=SysTime( )**, свойство **Вид (Отображаемое Значение)**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска SysTime.

Функция **`TimeCurrent( )`** возвращает текущие время/дату. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска TimeCurrent. Функция **`TimeToStr( )`** преобразует время/дату в строку. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска TimeToStr. Пример использования этих функций имеется в проекте **TmpltAndMenu** (шаблон графической страницы **mytraining.MyNormal**).

Информацию о других функциях этой группы можно получить в [2], тема Cicode Programming Reference |Cicode Function Categories | Time and Date Functions.

23.3.5. Разные функции (Miscellaneous Functions)

Функция **`Beep( )`** формирует акустический сигнал с помощью внутреннего динамика или встроенной звуковой карты. Аргумент функции может принимать значения -1, 0, 1, 2, 3 или 4, что соответствует различным акустическим сигналам. Если используется встроенный динамик компьютера, то функция завершается только после окончания воспроизведения звука. Если используется звуковая карта, то функция завершается немедленно, а звук воспроизводится в фоновом режиме. Показать на примере проекта **OvenTraining23**, команда

Алармы | Категории алармов в среде **Редактора проектов Citect**, поле **Действие при возникновении алармов**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска Beep.

Функция **`FullName( )`** возвращает полное имя текущего зарегистрированного в системе пользователя. Показать на примере проекта **OvenTraining23**, графическая страница **Home**, графический объект **Текст** с надписью **=FullName()**, свойство **Вид (Отображаемое значение)**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска FullName.

Функция **`Login( )`** регистрирует пользователя в системе и определяет права доступа пользователя к ресурсам системы Vijeo Citect. Показать на примере проекта **OvenTraining23**, графическая страница **Home**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска Login.

Функция **`LoginForm( )`** отображает диалоговое окно, с помощью которого пользователь регистрируется в системе. Показать на примере проекта **OvenTraining23**, команда **Регистрация** всплывающего меню. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска LoginForm.

Функция **`Logout( )`** выполняет разрегистрацию пользователя в системе. Показать на примере проекта **OvenTraining23**, команда **Разрегистрация** всплывающего меню. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска Logout.

Функция **`Name( )`** возвращает имя текущего зарегистрированного в системе пользователя. Показать на примере проекта **OvenTraining23**, графическая страница **Home**, графический объект **Текст** с надписью **=Name()**, свойство **Вид (Отображаемое значение)**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска Name.

Функция **`UserInfo( )`** возвращает информацию о текущем зарегистрированном в системе пользователе. Вид возвращаемой информации определяется значением аргумента функции. Показать на примере проекта **OvenTraining23**, графическая страница **Home**, графический объект **Текст** с надписью **=UserInfo(0)**, свойство **Вид (Отображаемое значение)**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска UserInfo.

Функция **`Prompt( )`** отображает на экране сообщение, заданное аргументом функции. Показать на примере проекта **OvenTraining23**, Среда **Редактора проектов Citect**, команда **Алармы | Категории алармов**, поля **Действие при возникновении аларма** и **Действие при сбросе аларма**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска Prompt.

Функция **`Shutdown( )`** завершает работу проекта системы Vijeo Citect. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска Shutdown.

Функция **ShutdownForm( )** показывает диалоговое окно, позволяющее пользователю завершить работу проекта системы Vijeo Citect. Показать на примере проекта **OvenTraining23**, графическая страница **Home**, рисованный графический объект, свойства **Доступ (Общие)** и **Ввод (Касание)**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска ShutdownForm.

Функция **Sleep( )** приостанавливает текущую задачу Cicode, из которой вызвана функция, на количество секунд, заданное аргументом функции. При этом функция не влияет на выполнение других задач Cicode. Показать на примере проекта **OvenTraining23**, среда приложения **Редактор проектов Citect** , команда **Система | События**, диалоговое окно **События**, поле **Действие**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска Sleep.

Функция **SleepMS( )** аналогично функции **Sleep( )**, но ее аргумент задает время приостановки в миллисекундах. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска SleepMS.

Функция **Rand(n)** возвращает псевдослучайное число между 0 и значением (n-1), используя равномерный закон распределения. Значение аргумента функции должно быть в диапазоне от 2 до 32767 включительно. Показать на примере проекта **OvenTraining23**, среда приложения **Редактор проектов Citect**, команда **Система | События**, диалоговое окно **События**, событие **Temp**, поле **Действие**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска Rand.

Функция **Toggle( )** изменяет значение дискретного тега, указанного в качестве аргумента функции, на противоположное. Показать на примере проекта **OvenTraining23**, среда приложения **Редактор проектов Citect**, команда **Система | События**, диалоговое окно **События**, событие **Global**, поле **Action**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска Toggle.

Функция **WinNewAt( )** отображает заданную страницу в новом окне, расположенном в указанном месте. Созданное окно может быть позже закрыто с помощью функции **WinFree( ).** Функция имеет следующий синтаксис:

```
WinNewAt(Page, X, Y, Mode);
```

Здесь *Page* — имя или номер страницы, отображаемой в окне; *X* — горизонтальная координата в пикселях левого верхнего угла окна; *Y* — вертикальная координата в пикселях левого верхнего угла окна; *Mode* — режим отображения окна, который может комбинироваться из перечисленных далее элементарных режимов.

- *0* — обычная страница.
- *1* — "страничное" дочернее окно. Окно закрывается, когда отображается новая страница, например, когда происходит вызов функции **PageDisplay( )** или **PageGoto( )**. Родительским является текущее активное окно.
- *2* — "оконное" дочернее окно. Окно закрывается автоматически, когда родительское окно "освобождается" с помощью функции **WinFree()**. Родительским является текущее активное окно.

- *4* — запрет изменения размера. Окно отображается с тонкой рамкой и без иконок "Максимизировать/минимизировать".
- *8* — отсутствие иконок. Окно отображается с тонкой рамкой и без иконок "Максимизировать/минимизировать" и системного меню. Нельзя изменять размер окна.
- *16* — отсутствие заголовка. Окно отображается с тонкой рамкой, без заголовка и без иконок "Максимизировать/минимизировать" и системного меню. Нельзя изменять размер окна.
- *32* — разрешено эхо. В этом случае весь ввод с клавиатуры, подсказки, сообщения об ошибках отображаются в родительском окне. Этот режим следует использовать при работе с дочерними окнами (режимы 1 и 2).
- *64* — всегда на переднем плане (всплывающее окно).
- *128* — открыть уникальное окно. Этот режим предотвращает открытие окна два или более раз.
- *256* — открыть окно целиком. Этот режим обеспечивает такое отображение, когда ни одна часть окна не выходит за пределы экрана.
- *512* — открыть уникальное окно **Суперджина**. Этот режим предотвращает открытие **Суперджина** одновременно два или более раз, но тот же **Суперджин**, связанный с другим содержимым, может быть открыт.
- *1024* — запрет динамического изменения размера окна. Имеет более высокий приоритет по сравнению с параметром **[Page]DynamicSizing**.
- *4096* — позволяет окну быть измененным, не поддерживая текущий формат изображения. Формат изображения определяет соотношение между шириной и высотой окна, что означает, что это позволяет растягивать или сжимать окно к любым пропорциям. Имеет более высокий приоритет по сравнению с параметром **[Page]MaintainAspectRatio**.
- *8192* — текст на странице будет изменен в пропорции с максимальным изменением масштаба к измененному окну. Имеет более высокий приоритет по сравнению с параметром **[Page]ScaleTextToMax**.

Показать на примере проекта **OvenTraining23**, графическая страница **PopUp**, графический объект **Вставить джин**, библиотека **training**, Джин: **sg_time**, кнопка **Правка**, свойство **Ввод (Касание)**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска WinNewAt.

Функция **`AssWin( )`** связывает до восьми тегов переменных с **Суперджином** и отображает **Суперджин** в новом окне. Результат действия этой функции такой же, как при восьмикратном вызове функции **`Ass( )`** или функции **`AssTag( )`** с последующим вызовом функции **`WinNewAt( )`**. Если требуется связать с **Суперджином** более восьми тегов переменных, то следует вызвать функцию **`AssVarTags( )`**, **`AssTag( )`** или **`Ass( )`** для выполнения нужного связывания перед вызовом функции **`AssWin( )`**. Функция имеет следующий синтаксис:

```
AssWin(sPage, X, Y, Mode, sTag1..8);
```

Здесь *sPage* — имя страницы **Суперджина**; *X* — горизонтальная координата в пикселях левого верхнего угла окна; *Y* — вертикальная координата в пикселях левого верхнего угла окна; *Mode* — режим отображения окна, который может комбинироваться из перечисленных далее элементарных режимов:

- *0* — обычная страница.
- *1* — "страничное" дочернее окно. Окно закрывается, когда отображается новая страница, например, когда происходит вызов функции **PageDisplay( )** или **PageGoto( )**. Родительским является текущее активное окно.
- *2* — "оконное" дочернее окно. Окно закрывается автоматически, когда родительское окно "освобождается" с помощью функции **WinFree()**. Родительским является текущее активное окно.
- *4* — запрет изменения размера. Окно отображается с тонкой рамкой и без иконок "Максимизировать/минимизировать".
- *8* — отсутствие иконок. Окно отображается с тонкой рамкой и без иконок "Максимизировать/минимизировать" и системного меню. Нельзя изменять размер окна.
- *16* — отсутствие заголовка. Окно отображается с тонкой рамкой, без заголовка и без иконок "Максимизировать/минимизировать" и системного меню. Нельзя изменять размер окна.
- *32* — разрешено эхо. В этом случае весь ввод с клавиатуры, подсказки, сообщения об ошибках отображаются в родительском окне. Этот режим следует использовать при работе с дочерними окнами (режимы 1 и 2).
- *64* — всегда на переднем плане (всплывающее окно).
- *128* — открыть уникальное окно. Этот режим предотвращает открытие окна два или более раз.
- *256* — открыть окно целиком. Этот режим обеспечивает такое отображение, когда ни одна часть окна не выходит за пределы экрана.
- *512* — открыть уникальное окно **Суперджина**. Этот режим предотвращает открытие **Суперджина** одновременно два или более раз, но тот же **Суперджин**, связанный с другим содержимым, может быть открыт.
- *1024* — запрет динамического изменения размера окна. Имеет более высокий приоритет по сравнению с параметром **[Page]DynamicSizing**;
- *sTag1..8* — первые восемь физических тегов для связывания с **Super Genie**.

Показать на примере проекта **OvenTraining23**, графическая страница **Oven**, графический объект **Кнопка** с надписью **Управление задвижкой**, свойство **Ввод (Касание)**. Более подробные сведения о функции см. в [2], вкладка "Указатель", ключевое слово для поиска AssWin.

Функция **DspPopupMenu( )** позволяет создавать всплывающие пользовательские меню, которые могут содержать подменю и команды. Пример использования этой

функции рассмотрен ранее в главе 16, проект **TmpltAndMrnu**, графическая страница **MyPage**, графический объект **Кнопка** с надписью **Меню Pages**.

Примечание
Хорошими способами изучения стандартных функций Cicode являются изучение демонстрационного проекта **Example** и графических объектов шаблонов графических страниц включаемых проектов.

23.4. Категории стандартных функций Cicode и их краткое описание

Язык Cicode включает 944 стандартные функции, разбитые на 48 категорий (табл. 23.1).

Таблица 23.1. *Категории стандартных функций языка Cicode*

Категория	Назначение
Accumulator Functions (8 функций)	Операции с аккумуляторами
ActiveX Functions (12 функций)	Операции с объектами ActiveX
Alarm and Alarm Filter Functions (91 функция)	Операции с сигналами тревог
Clipboard Functions (5 функций)	Операции с буфером промежуточного хранения
Cluster Functions (10 функций)	Операции с кластерами
Color Functions (7 функций)	Операции с цветами
Communication Functions (6 функций)	Операции с коммуникациями
DDE Functions (13 функций)	Взаимодействие приложений с использованием DDE
Device Functions (31 функция)	Управление доступом к устройствам
Display Functions (92 функции)	Управление отображением и обработкой графических страниц и объектов
DLL Functions (4 функции)	Операции с DLL
Equipment Database Functions (17 функций)	Операции с базами данных оборудования
Event Functions (5 функций)	Операции с событиями
Error Functions (12 функций)	Операции с ошибками (проверка статуса других функций)
Cicode Errors (3 функции)	
File Functions (23 функции)	Операции с файлами

Категория	Назначение
Form Functions (35 функций)	Операции с формами
Format Functions (11 функций)	Операции конвертирования данных
FTP Functions (5 функций)	Мониторинг FTP коммуникаций и файлов
FuzzyTech Functions (7 функций)	Поддержка управления с нечеткой логикой
Group Functions (10 функций)	Управление группами областей, категориями сигналов тревог и т.п.
I/O Device Functions (4 функции)	Операции с устройствами ввода/вывода
Keyboard Functions (17 функций)	Операции с клавиатурой
Mail Functions (5 функций)	Операции с использованием почты
Math/Trigonometry Functions (25 функций)	Математические и тригонометрические функции
Menu Functions (19 функций)	Операции с меню
Miscellaneous Functions (54 функции)	Прочие функции
Page Functions (37 функций)	Операции с графическими страницами
Plot Functions (12 функций)	Операции с графиками и диаграммами
Process Analyst Functions (5 функций)	Операции с анализатором процессов
Quality Functions (10 функций)	Операции с качеством
Report Functions (4 функции)	Операции с отчетами
Scheduler Functions (38 функций)	
Security Functions (24 функции)	Операции с пользователями и их правами
Sequence of Events Functions (4 функции)	
Server Functions (13 функций)	Операции с серверами
SPC Functions (12 функций)	Обработка статистики процессов управления
SQL Functions (37 функций)	Операции с SQL-базами данных
String Functions (35 функций)	Операции со строками
Super Genie Functions (21 функция)	Операции с Super Genie
Task Functions (36 функции)	Управление задачами
Table (Array) Functions (3 функции)	Получение максимума, минимума, математического ожидания или отклонения для значений элементов таблицы или массива
Tag Functions (38 функций)	Управление тегами

Категория	Назначение
Task Functions (36 функций)	
Time and Date Functions (22 функции)	Получение значения даты или времени
Timestamp Functions (12 функций)	Операции с метками времени
Trend Functions (84 функций)	Операции с трендами
Window Functions (32 функции)	Управление отображением окон

Более подробные сведения об отдельных категориях стандартных функций и их составе см. в [2], тема Cicode Programming Reference | Cicode Functions Categories | Название_категории. Подробные сведения об отдельной стандартной функции см. в [2] , вкладка "Указатель", ключевое слово для поиска Имя_функции.

Приложение 1. Инсталляция, конфигурирование и тестирование драйвера связи с контроллером Twido

Если драйвер связи контроллера Twido и персонального компьютера (**Schneider Modbus Serial Driver**) еще не был установлен, то по окончании установки драйвера кабеля связи необходимо установить драйвер связи контроллера и компьютера. Самораспаковывающийся файл для установки драйвера находится в папке **Drivers** установочного компакт-диска **TwidoSoft Programming software v3.5** (рис. П1.1).

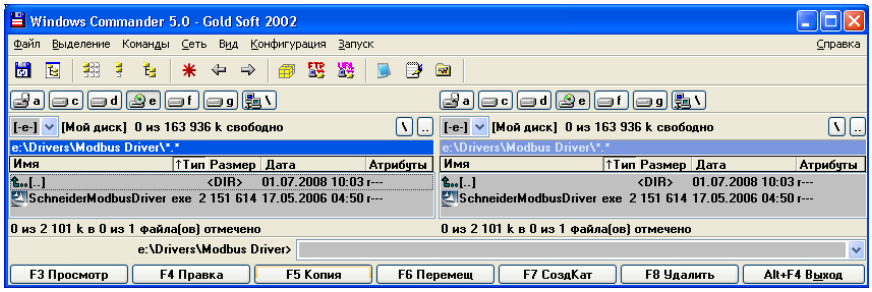


Рис. П1.1 Самораспаковывающийся файл для установки драйвера **Schneider Modbus Serial Driver**

После двойного щелчка по загрузочному файлу появится окно, предупреждающее, что далее будет производиться установка данного драйвера (рис. П1.2). После нажатия кнопки **Next**, начинается процесс установки драйвера (рис. П1.3). По его окончанию будет предложено перегрузить компьютер (рис. П1.4). Выбираем строку для перегрузки компьютера, т. е. **Yes, I want to restart my computer now** и далее нажимаем кнопку **Next**.

После установки драйвера и перегрузки компьютера для конфигурирования драйвера и его тестирования запускаем приложение **Drivers management** (рис. П1.5). В результате открывается окно **Свойства: SCHNEIDER Drivers management**, показанное на рис. П1.6. В поле группы **Drivers** выбираем **MODBUS** и переходим на вкладку **MODBUS SERIAL Driver** (рис. П1.7). Далее для открытия окна конфигурации драйвера нажимаем кнопку **[1]Configuration** (см. приведенный ранее рис. П1.7). В окне **MODBUS Driver — MODBUS01** выбираем вкладку **Configuration** и устанавливаем параметры соединения в соответствии с рис. П1.8. Остальные вкладки оставляем без изменений и нажимаем кнопку **OK**.

На этом конфигурирование драйвера заканчивается и необходимо провести его тестирование. Перед тестированием следует убедиться, что контроллер подключён по кабелю **USB (TSXPCX3030)** к компьютеру и на него подано питающее

напряжение. Для тестирования связи в окне **Свойства: SCHNEIDER Drivers management**, выбираем вкладку **MODBUS Test** (рис. П1.9).

Рис. П1.2. Предупреждение об установке драйвера

Для тестирования связи на выбранной вкладке в поле **Protocol** выбираем протокол **Modbus Serial[1]** и для проверки соединения нажимаем кнопку **Connect** (в строке **State** должно отобразиться наличие соединения — **Connected)**. Для проверки информационного обмена нажимаем кнопку **Start** (начнутся запросы и ответы, которые можно будет наблюдать в строке **Request,** рис. П1.10).

Для завершения тестирования последовательно нажимаем кнопки **Stop** и **Disconnect** и закрываем диалоговое окно нажатием кнопки **OK**.

Приложение 2. Конфигурирование и программирование контроллера Twido. Импорт-экспорт программ

Перед разработкой новой программы контроллер Twido следует переконфигурировать соответственно его реальной комплектации.

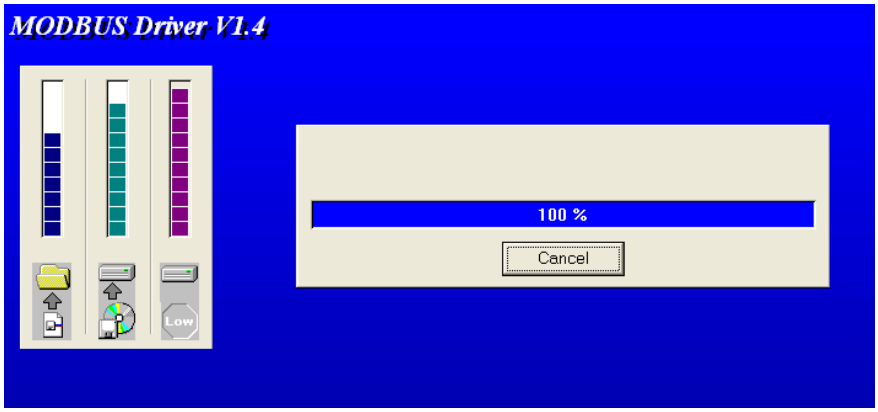


Рис. П1.3. Процесс установки драйвера

Примечание

При загрузке в интегрированную среду разработки (ИСР) **TwidoSoft** ранее разработанного проекта переконфигурирование контроллера не требуется — в составе загружаемого в ИСР проекта содержится не только программа, но и соответствующая ей конфигурация контроллера.

П2.1. Конфигурирование контроллера, ввод новой программы и ее тестирование

После запуска ИСР **TwidoSoft** выполняем команду **File | New** или нажимаем кнопку **New** на панели инструментов и создаём новый проект (рис. П2.1). Появившееся окно **Functional Level Management** оставляем без изменения и нажимаем на кнопку **OK.** В левом поле окна ИСР в виде дерева проекта представлена предлагаемая по умолчанию информация о структуре проекта (в том числе конфигурация контроллера **Twido)**. Если дерево проекта не отображается, то следует в меню **View** выбрать **Application browser** (установить слева галочку). Конфигурацию следует модифицировать в соответствии с реальным составом модулей контроллера (базовый

контроллер TWDLMDA20DTK, модули расширения — дискретных входов TWDDDI8DT, дискретных выходов TWDDRA8RT, аналоговых входов TWDAMI4LT, аналоговых выходов TWDAVO2HT и модуль коммуникации Modbus). С помощью команды **File | Save as…** сохраняем созданный проект под именем **Example**.

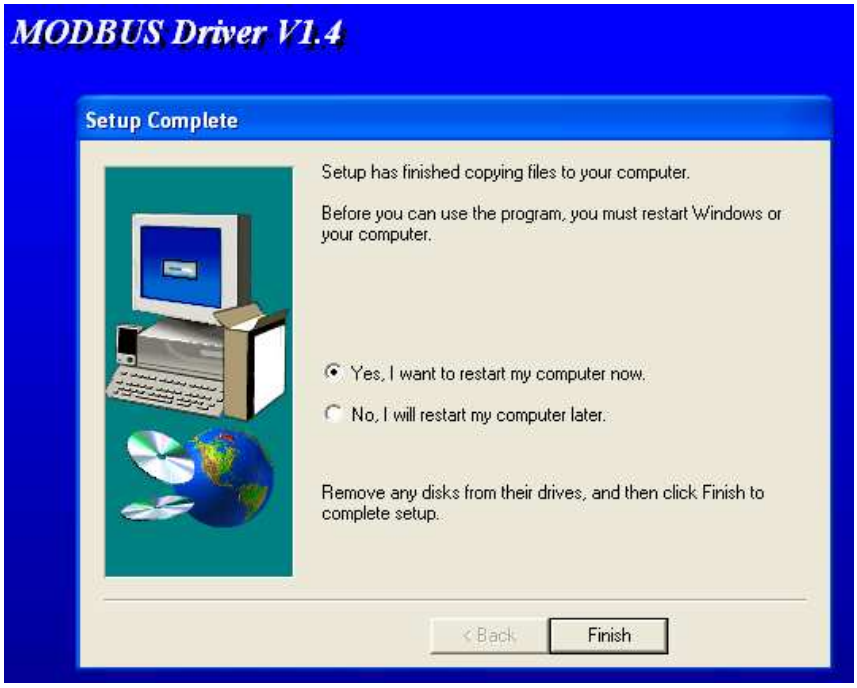


Рис. П1.4. Окончание установки драйвера связи

Рис. П1.5. Запуск **Drivers management**

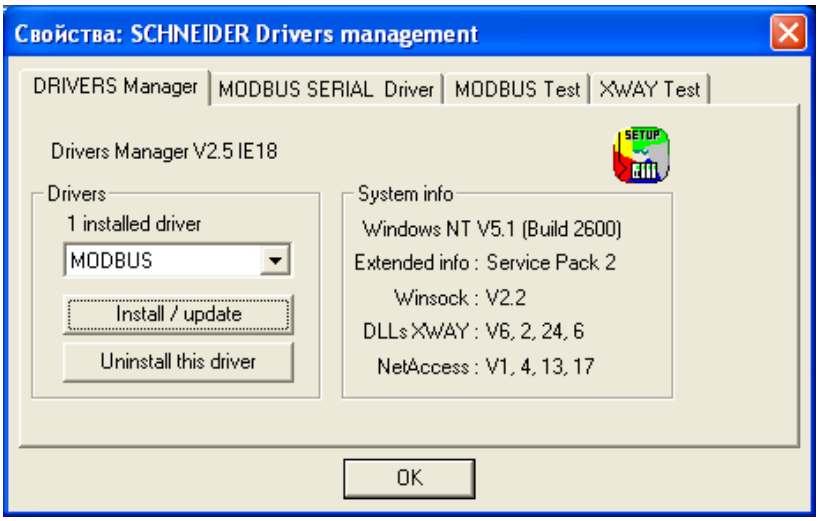


Рис. П1.6. Конфигурирование драйвера связи

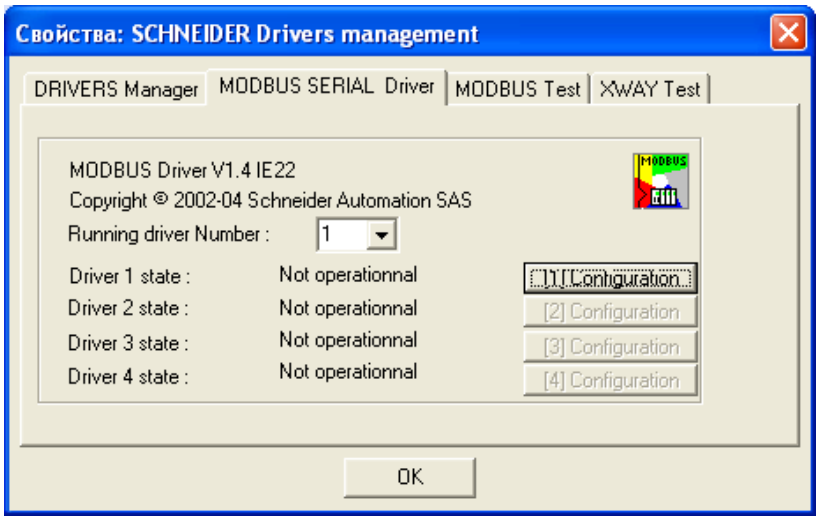


Рис. П1.7. Вкладка **MODBUS SERIAL Driver**

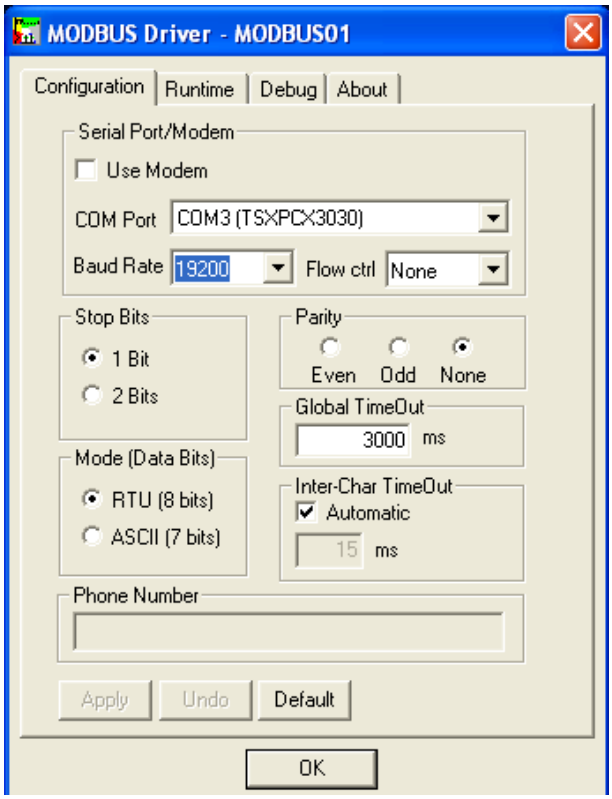


Рис. П1.8. Окно конфигурирования драйвера связи

Для модификации конфигурации контроллера в дереве проекта вначале выбираем имя контроллера по умолчанию (**TWDLMDA40DTK**), выполняем команду **Change Base Controller...** его контекстного меню, в поле **Controller:** появившегося окна **Change Base Controller** выбираем **TWDLMDA20DTK** и нажимаем кнопку **Change** (рис. П2.2). Далее в дереве проекта выбираем **Port 1: Remote link, 1**, выполняем команду **Edit Controller Com Setup…** его контекстного меню, в появившемся окне **Controller Communication Setup** в поле **Type:** выбираем **Modbus** (рис. П2.3) и нажимаем кнопку **OK**. Для конфигурирования модулей расширения контроллера (**Expansion Bus**) в дереве объектов проекта выбираем **Expansion Bus**, выполняем команду **Add a module…** его контекстного меню, в появившемся окне **Add module** в поле **Module:** выбираем **TWDDDI8DT** и нажимаем кнопку **Add** (рис. П2.4). Аналогичным образом добавляем модули расширений **TWDDRA8RT, TWDAMI4LT** и **TWDAVO2HT**. После добавления последнего модуля расширения

нажимаем кнопку **Done**. Результирующая конфигурация контроллера показана на рис. П2.5.

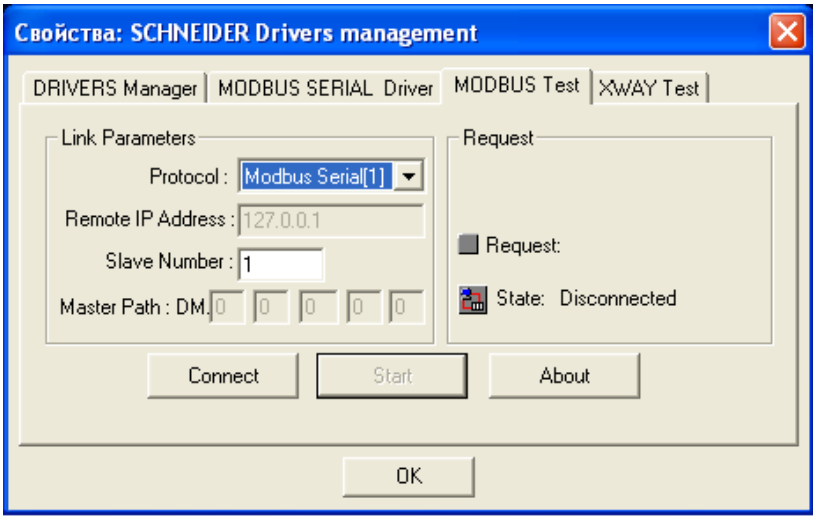


Рис. П1.9. Окно тестирования драйвера связи

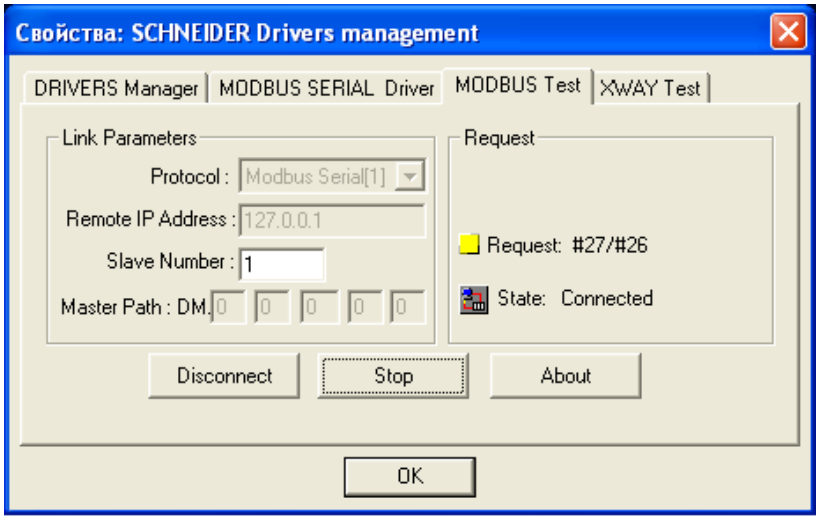


Рис. П1.10. Тестирование драйвера связи

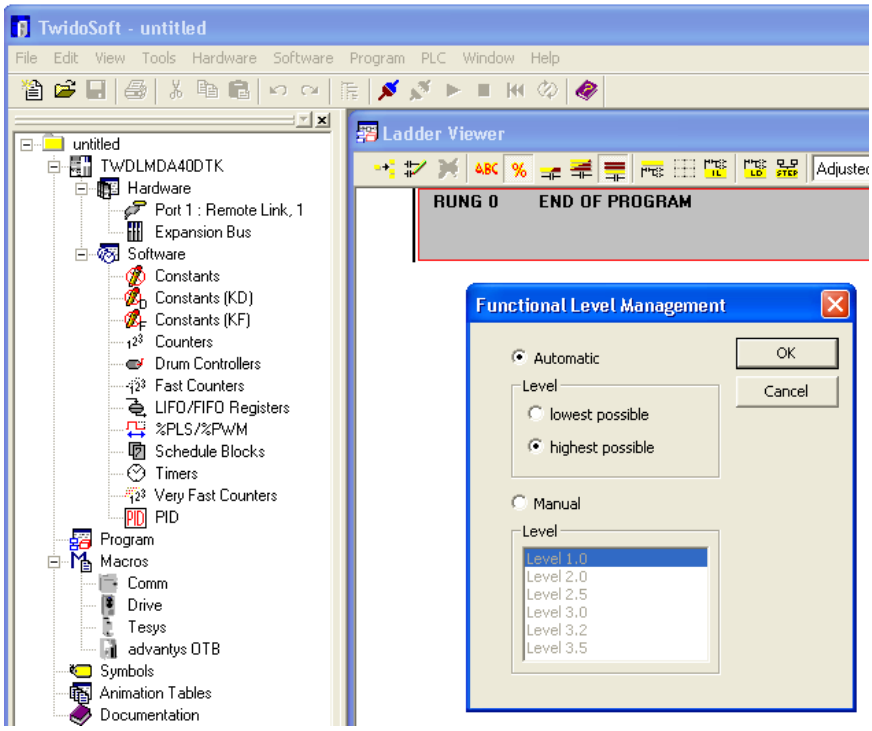


Рис. П2.1. Создание нового проекта в ИСР **TwidoSoft**

В заключение необходимо настроить используемые входы/выходы модулей аналогового ввода (TWDAMI4LT) и вывода (TWDAVO2HT). Для конфигурирования используемых аналоговых входов в дереве проекта выбираем модуль аналогового ввода **TWDAMI4LT**, выполняем команду **Configure…** его контекстного меню (рис. П2.6), в появившемся окне **Configure Module — TWDAMI4LT [Position 3]** настраиваем используемый вход в соответствии с рис. П2.7 и нажимаем кнопку **OK**. Для конфигурирования используемых аналоговых выходов в дереве проекта аналогично выбираем модуль аналогового вывода **TWDAVO2HT**, выполняем команду **Configure…** его контекстного меню, в появившемся окне **Configure Module — TWDAVO2HT [Position 4]** настраиваем используемый вход в соответствии с рис. П2.8 и нажимаем кнопку **OK**.

Теперь контроллер полностью сконфигурирован и можно вводить новую программу. Для ввода программы воспользуемся языком Ladder Language. Вводимая программа будет содержать три ступени (Rung).

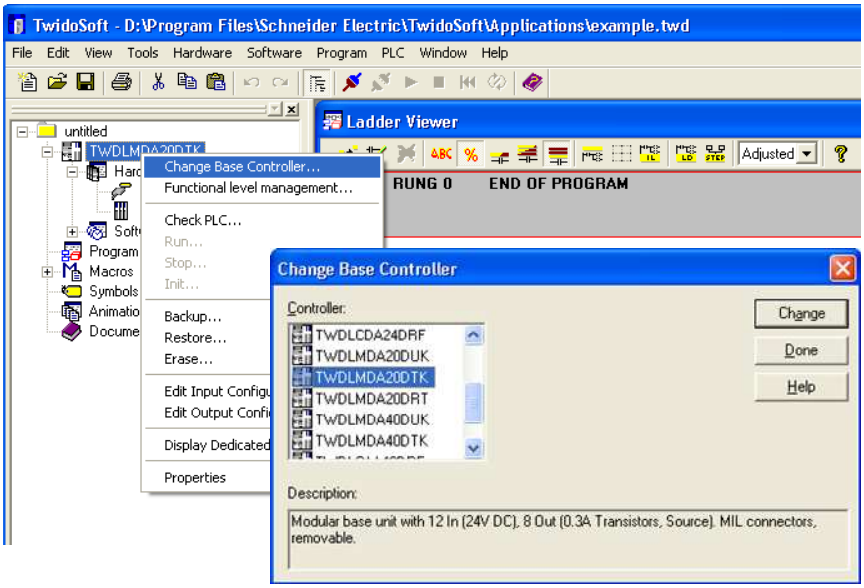


Рис. П2.2. Выбор модуля контроллера

Для ввода первой ступени программы выполняем команду **Tools | Rung** или активизируем акселератор **Ins**. В появившемся окне **Ladder Editor — Insert Rung** для удобства дальнейшей работы с помощью кнопки **Toggle Grid**, расположенной на главной панели инструментов включаем отображение сетки. С помощью кнопки **Contact** (акселератор **F2**), расположенной на вспомогательной панели инструментов под главной панелью, размещаем в левой ячейке поля **Test Zone** первой ступени (рис. П2.9) блок дискретного ввода. Для выбора модуля расширения и адреса входа выполняем над вставленным элементом двойной щелчок левой кнопкой мыши и в появившемся поле вводим **%I1.0** (% — признак переменной или константы контроллера, I — признак модуля дискретных входов, 1 — позиционный номер модуля, 0 — адрес входа) и нажимаем клавишу **ENTER**. С помощью кнопки **Coil** (акселератор **Schift+F2**) размещаем в правой ячейке поля Action Zone (см. приведенный ранее рис. П2.9) блок дискретного вывода. Для выбора модуля расширения и адреса выхода выполняем над вставленным элементом двойной щелчок левой кнопкой мыши и в появившемся поле вводим **%Q2.0** (% — признак переменной или константы контроллера, Q — признак модуля дискретных выходов, 2 — позиционный номер модуля, 0 — адрес выхода) и нажимаем клавишу **ENTER**. Для соединения блоков между собой нажимаем левой кнопкой мыши кнопку **Horizontal connector** (акселератор **F11**), отпускаем кнопку, перемещаем мышь на ячейку рядом с блоком дискретного ввода и нажимаем клавишу мыши. В результате создания первой ступени программы дискретный выход с адресом 0 модуля

расширения дискретных выходов отслеживает состояние дискретного входа с адресом 0 модуля расширения дискретных входов.

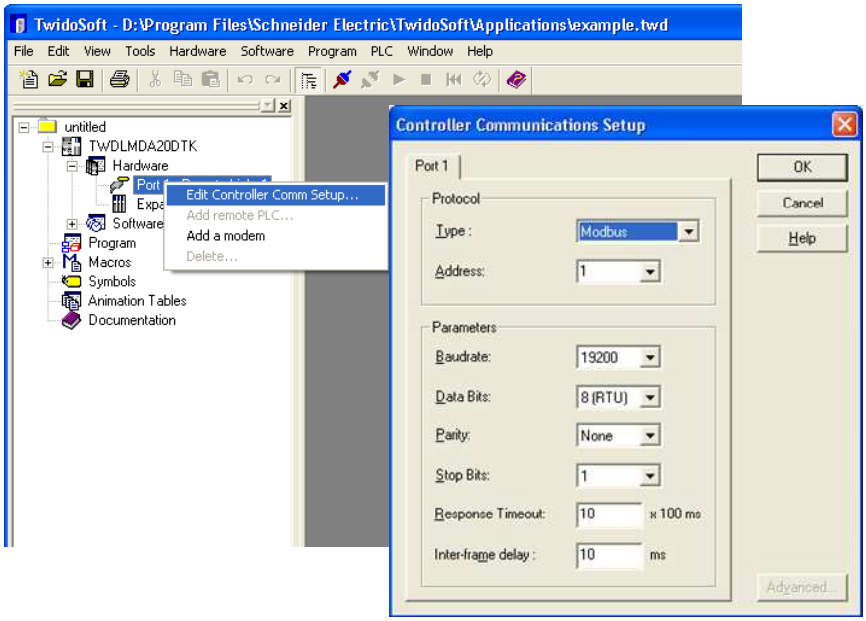


Рис. П2.3. Конфигурирование модуля коммуникации

Для ввода второй ступени программы выполняем команду **Tools | New Rung**. В окне **Ladder Editor — Insert Rung** на панели инструментов, расположенной под главной панелью, для установки блока дискретного входа левой кнопкой мыши нажимаем кнопку **Connect** (акселератор **F2**), перемещаем мышь в левую ячейку сетки зоны **Test Zone** создаваемой ступени и вновь нажимаем левую кнопку мыши (см. приведенный ранее рис. П2.9).

Для задания позиционного номера и адреса входа модуля дискретных входов над установленным блоком выполняем двойной щелчок левой кнопкой мыши, в появившемся поле вводим **%I1.1** (% — признак внутренней переменной или константы контроллера, I — признак модуля дискретных входов, 1 — позиционный номер модуля, 1 — адрес входа модуля) и нажимаем клавишу **ENTER**. На панели инструментов, расположенной под главной панелью, для установки блока битовой переменной контроллера левой кнопкой мыши нажимаем кнопку **Coil** (акселератор **Schift+F2**), перемещаем мышь в правую ячейку сетки зоны Action Zone создаваемой ступени и вновь нажимаем левую кнопку мыши (см. приведенный ранее рис. П2.9). Для задания битовой переменной контроллера и ее адреса над установленным блоком выполняем двойной щелчок левой кнопкой мыши, в появившемся поле

вводим **%M1** (% — признак внутренней переменной или константы контроллера, M — признак размещения переменной в оперативной памяти контроллера, 1 — адрес битовой переменной) и нажимаем клавишу **ENTER**.

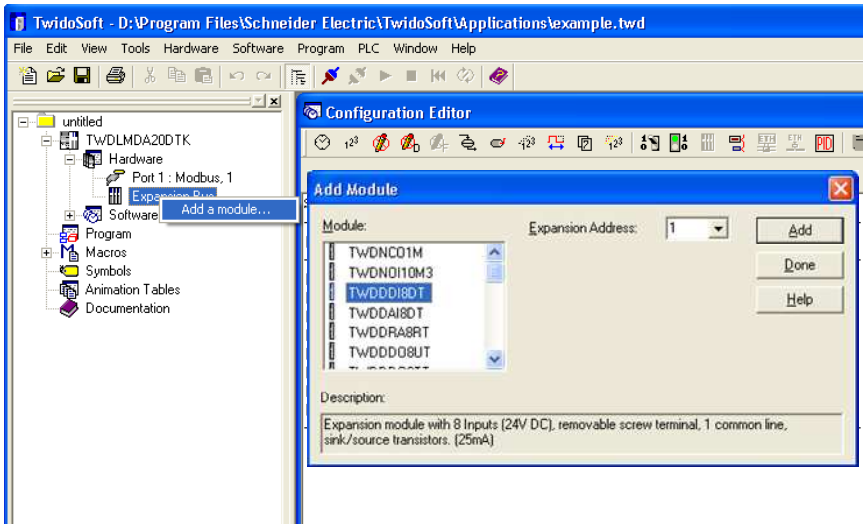


Рис. П2.4. Конфигурирование модуля расширения

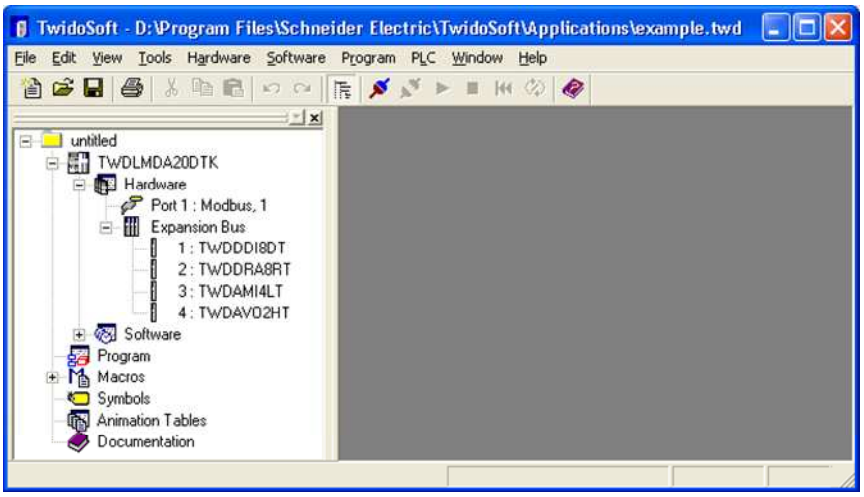


Рис. П2.5. Конфигурация контроллера после модификации

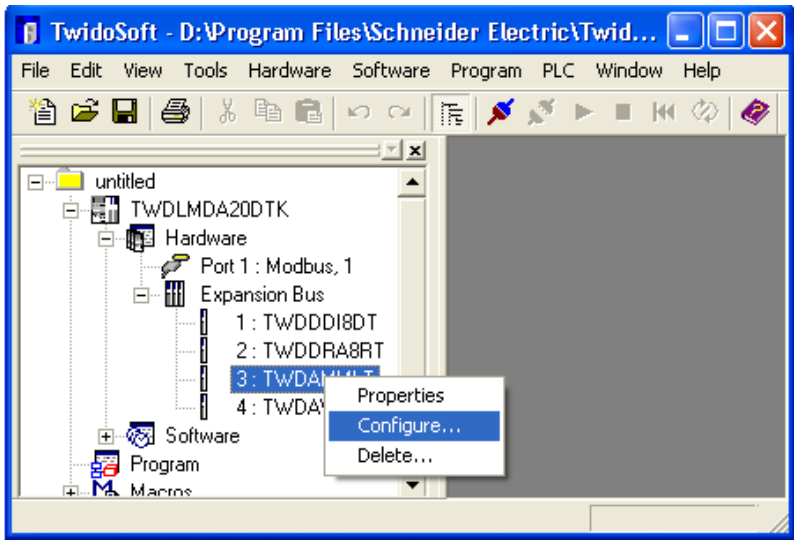


Рис. П2.6. Конфигурирование модуля аналогового ввода

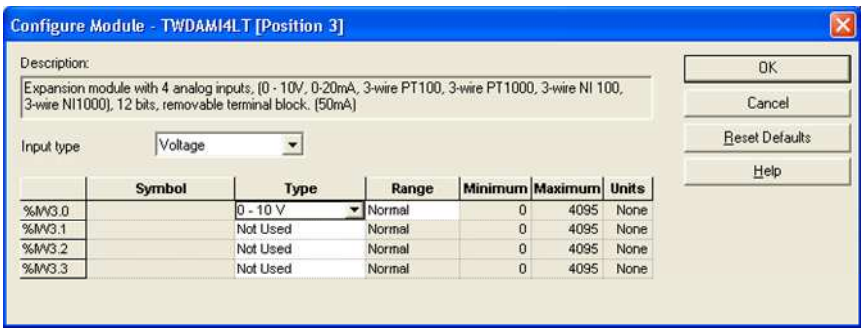


Рис. П2.7. Настройка входов модуля аналогового ввода

Для соединения блоков ступени на панели инструментов левой кнопкой мыши нажимаем кнопку **Horizontal connector fill** (акселератор **F11**), перемещаем мышь на ячейку, расположенную справа от блока дискретного входа, и еще раз нажимаем левую кнопку мыши (см. приведенный ранее рис. П2.9). С помощью созданной ступени битовая переменная контроллера с адресом 1 будет отслеживать состояние дискретного входа с адресом 1 модуля дискретных входов.

Для ввода третьей ступени программы выполняем команду **Tools | New Rung**. В окне **Ladder Editor — Insert Rung** на панели инструментов, расположенной под главной

панелью, для установки блока дискретного входа левой кнопкой мыши нажимаем кнопку **Connect** (акселератор **F2**), перемещаем мышь в левую ячейку сетки зоны Test Zone создаваемой ступени и вновь нажимаем левую кнопку мыши (см. приведенный ранее рис. П2.9).

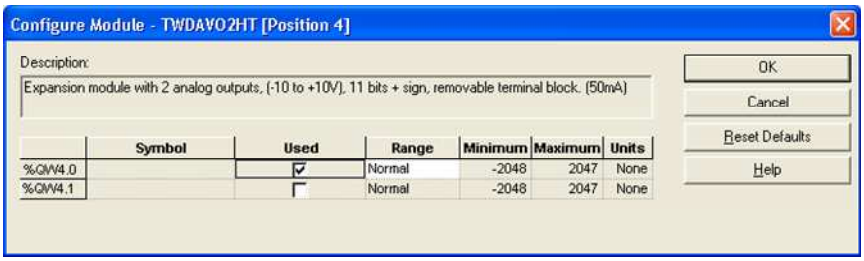


Рис. П2.8. Настройка выходов модуля аналогового вывода

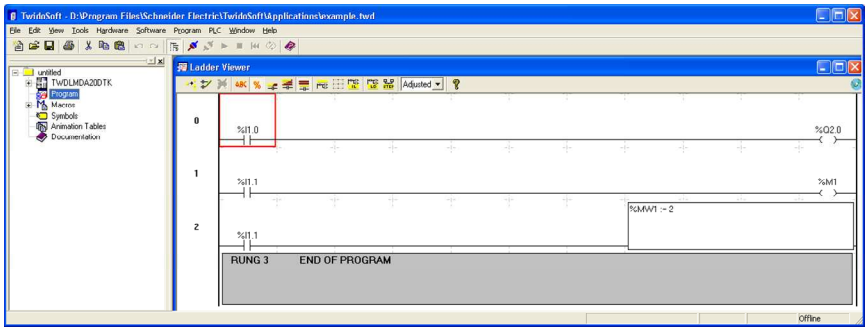


Рис. П2.9. Вид новой программы

Для задания позиционного номера и адреса входа модуля дискретных входов над установленным блоком выполняем двойной щелчок левой кнопкой мыши, в появившемся поле вводим **%I1.1** (% — признак внутренней переменной или константы контроллера, I — признак модуля дискретных входов, 1 — позиционный номер модуля, 1 — адрес входа модуля) и нажимаем клавишу **ENTER**. На панели инструментов, расположенной под главной панелью, для добавления операционного блока, содержащего оператор присваивания, левой кнопкой мыши нажимаем кнопку **Operate block** (акселератор **Schift+F8**), перемещаем мышь в правую ячейку сетки зоны Action Zone создаваемой ступени и вновь нажимаем левую кнопку мыши (см. приведенный ранее рис. П2.9). Для задания оператора присваивания над установленным блоком выполняем двойной щелчок левой кнопкой мыши, в появившемся поле вводим **%MW1:=2** (% — признак внутренней переменной или константы контроллера, M — признак размещения переменной в оперативной памяти контроллера, W — признак слова, 1 — адрес переменной) и нажимаем клавишу **ENTER**.

Для соединения блоков ступени на панели инструментов левой кнопкой мыши нажимаем кнопку **Horizontal connector fill** (акселератор **F11**), перемещаем мышь на ячейку, расположенную справа от блока дискретного входа, и еще раз нажимаем левую кнопку мыши (см. приведенный ранее рис. П2.9). С помощью созданной ступени переменная контроллера, имеющая формат слова, с адресом 0 будет принимать значение 2 всякий раз, когда на дискретном входе с адресом 1 модуля дискретных входов будет появляться единичное значение. Для проверки правильности созданной программы выполняем команду **Program | Analyze Program**. Результаты проверки представлены на рис. П2.10.

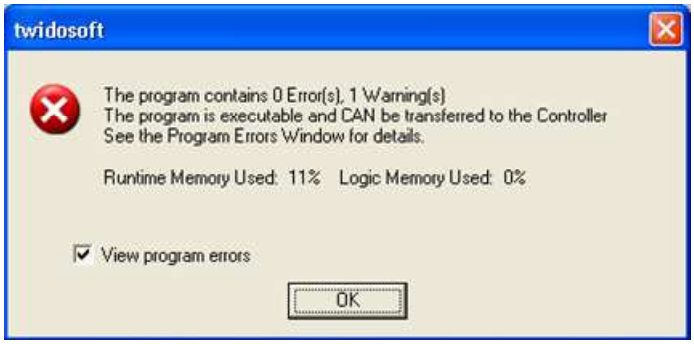


Рис. П2.10. Результаты проверки правильности программы

Для сохранения созданной программы и проекта в целом выполняем команду **File | Save As…** и сохраняем проект в папке **Applications**, расположенной в установочном каталоге TwidoSoft, под именем **Example**.

Для закрытия ИСР TwidoSoft выполняем команду **File | Exit**.

П2.2. Экспорт-импорт проектов между компьютером и контроллером

После запуска ИСР TwidoSoft v3.5 с помощью кнопки **Open** на панели инструментов открываем готовый проект **Example.twd** (рис. П2.11 и приведенный ранее рис. П2.9). Перед загрузкой готового проекта в контроллер нужно удостовериться, что контроллер Twido подсоединен к персональному компьютеру и на него подано питающее напряжение. Для загрузки проекта в контроллер сначала устанавливаем связь с ним. Для этого с помощью команды **PLC | Select a connection** выбираем тип соединения **USB** (рис. П2.12).

Далее с помощью команды **PLC | Connect…** устанавливаем соединение с контроллером. Если до этого момента в контроллер уже был загружен другой проект, то появится окно **Connect**, где будут предложены различные действия с проектом, который уже был загружен (рис. П2.13). Если же до этого момента в контроллер

ничего не загружалось, то окно **Connect** не появится, а в правом нижнем углу окна среды **TwidoSoft** будет мигать надпись об установлении соединения (рис. П2.14).

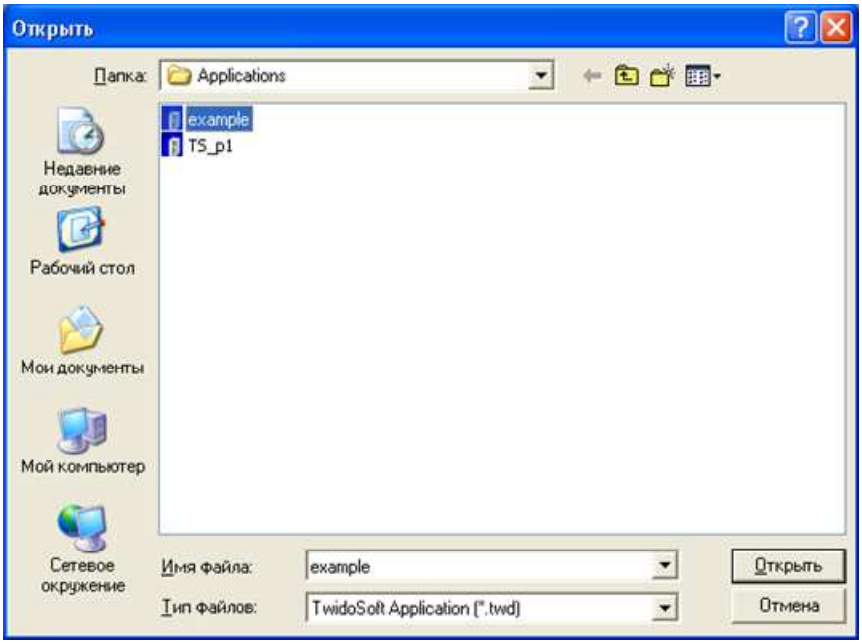


Рис. П2.11. Открытие существующего проекта

В окне **Connect** нажимаем кнопку **PC => Controller**. Последовательно появятся окна, представленные на рис. П2.15 и П2.16 (каждый раз нажимаем кнопку **OK**). По окончании в правом нижнем углу окна среды **TwidoSoft** будет мигать надпись об установлении соединения (см. приведенный ранее рис. П2.14).

С помощью команды **PLC | Run(RUN)** запускаем проект на выполнение. Появится запрос, представленный на рис. П2.17, нажимаем кнопку **OK**. Одновременно С помощью команды **PLC | Toggle animation'Ctrl+F7** можно наблюдать изменения переменных программы во время её выполнения (рис. П2.18).

В ИСР TwidoSoft можно создать символьный файл описания переменных, предназначенный для обеспечения информационной связи контроллера с проектом системы Vijeo Citect с помощью OPC Factory Server. Для этого в дереве проекта следует кликнуть два раза левой кнопкой мыши на элементе **Symbols**. В результате откроется окно **Symbol Editor** (рис. П2.19).

В первой колонке **Symbol** пишется имя переменной, во второй её адрес и в третьей — комментарии для этой переменной (третья колонка необязательна для заполнения). Пример заполнения символьного файла описания переменных представлен на

приведенном ранее рис. П2.19. После заполнения таблицы в дереве проекта выбираем **Symbols** и с помощью команды **Tools | Save symbol table** его контекстного меню сохраняем символьный файл описания переменных (файл сохраняется с тем же именем что и проект, но с расширением **.csv)**.

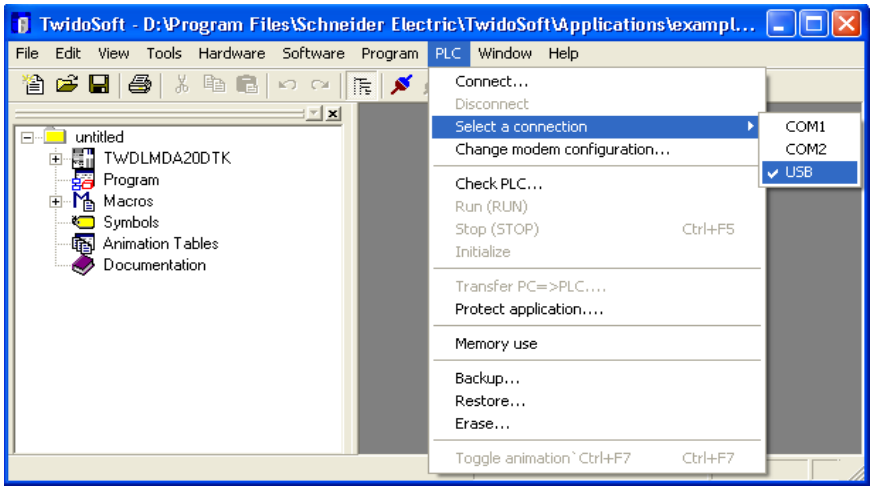


Рис. П2.12. Выбор соединения с контроллером

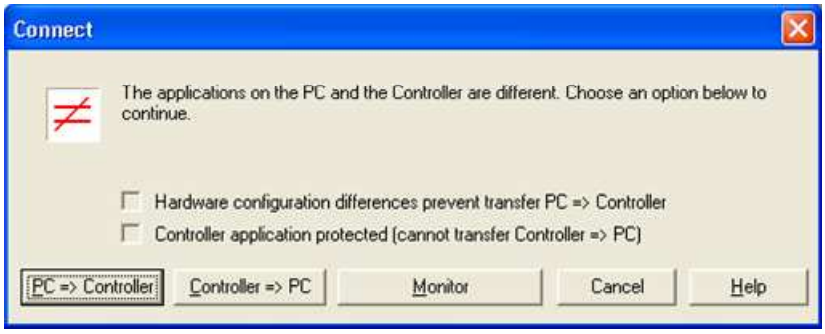


Рис. П2.13. Выбор действия с проектом (**PC => Controller** - перезаписать проект, который уже в контроллере, новым из среды **TwidoSoft**; **Controller => PC** - открыть проект, который уже был загружен в контроллер в среде **TwidoSoft**)

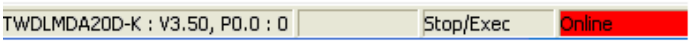


Рис. П2.14. Соединение установлено

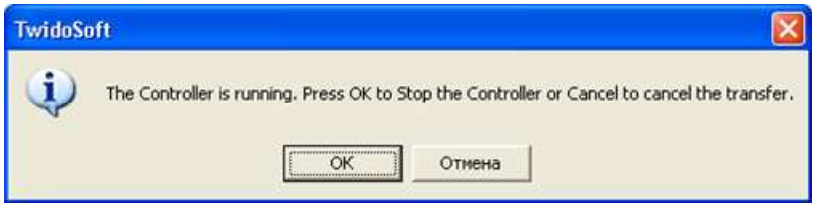


Рис. П2.15. Запрос на останов работы программы при загрузке проекта в контроллер

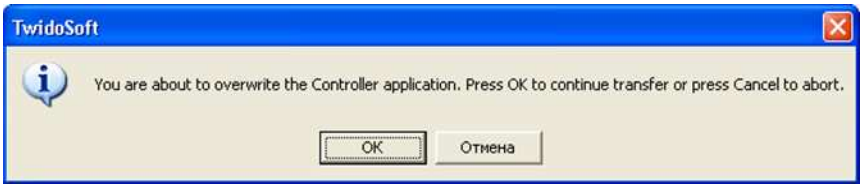


Рис. П2.16. Запрос на перезапись программы при загрузке проекта в контроллер

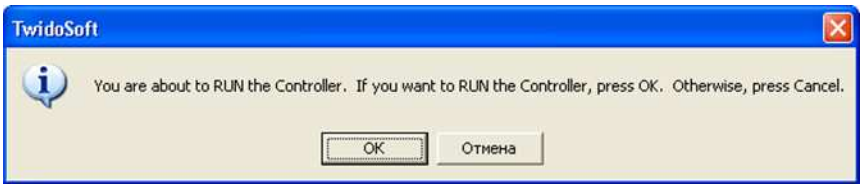


Рис. П2.17. Запрос на запуск контроллера

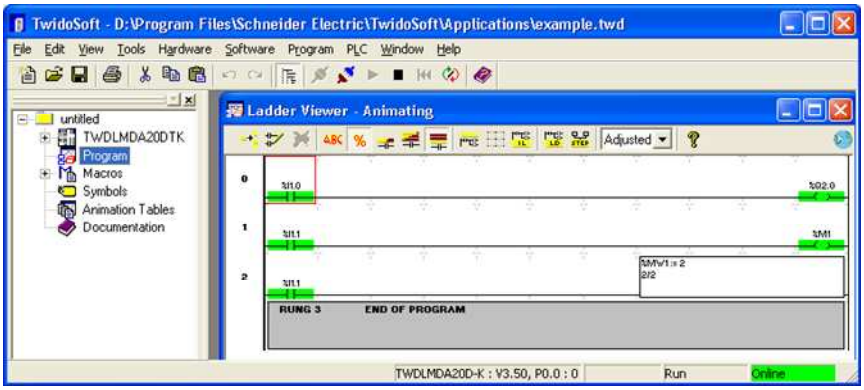


Рис. П2.18. Анимация программы при ее выполнении

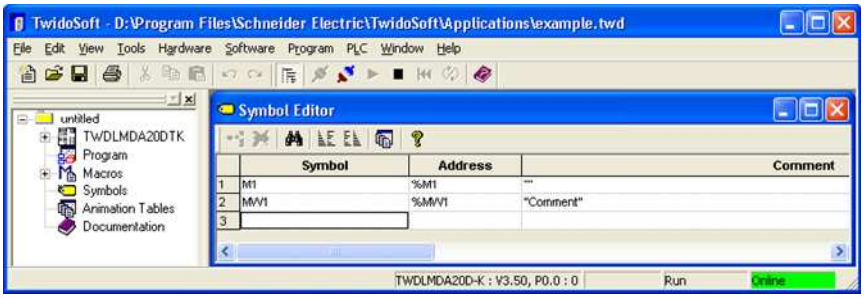


Рис. П2.19. Символьный файл описания переменных проекта **Example**

Далее следует с помощью любого текстового редактора выполнить модификацию полученного файла **Example.csv**. Содержимое символьного файла описания переменных до его модификации:

```
M1;%M1;""
MW1;%MW1;"Comment"
```

Содержимое символьного файла описания переменных поле его модификации для его использования в сервере **OFS**:

```
%M1,M1,Comment
%MW1,MW1
```

Перед завершением работы в ИСР TwidoSoft выполняем команду **PLC | Disconnect**, на появившийся запрос отвечаем подтверждением **Да**, а для завершения работы выполняем команду **File | Exit**. ИСР TwidoSoft закрывается, но проект в контроллере продолжает выполняться.

Приложение 3. Описание диска (папки "Инсталлятор", "Учебное пособие AUT20 и AUT24")

Папки, файлы диска и их назначение приведены в табл. П3.1.

Таблица П3.1. *Структура диска*

Папка или файл	Часть, глава, прилож., раздел	Описание
Папка “Инсталлятор”		
TwidoSoft 3.5	Часть 1, глава 3, приложения 1, 2	Интегрированная среда разработки проектов для контроллера Twido: папка с файлами для установки (для старта установки следует запустить файл setup.exe)
Файл Communication Drivers\ SchneiderModbusDriverSuite.exe	Часть 1, глава 3, приложения 1, 2	Файл с драйверами для автономного подключения контроллера Twido к компьютеру через USB в операционной системе Windows 7
Файл Communication Drivers\ EngSetupDrv.pdf	Часть 1, глава 3, приложения 1, 2	Руководство по установке файла с драйверами для автономного подключения контроллера Twido к компьютеру через USB в операционной системе Windows 7
VijeoCitect730\Vijeo Citect V7.30	Части 1, 2	SCADA-система Vijeo Citect 7.30: папка с файлами для установки 5 (для старта установки следует запустить файл Launch.exe)
Важно! В папке **Инсталлятор\VijeoCitect730** есть и другие полезные материалы. Настоятельно рекомендуем ознакомиться с ними		
VijeoCitect-7.30-Service-Pack-1	Части 1, 2	Пакет обновления 1 для Vijeo Citect 7.30
Папка «Учебное пособие AUT20 и AUT24\Учебное пособие»		
Vijeo Citect 7.30 SP1. Базовый курс.doc		Учебное пособие AUT20 и AUT24
Аннотация.doc		Аннотация учебного пособия

Папка или файл	Часть, глава, прилож., раздел	Описание
Учебное пособие\Vijeo Basics RU to print.ppt	Части 1 и 2	Файл презентации на русском языке для Vijeo Citect версии 6.1
Учебное пособие\Vijeo Configuration Course v7.ppt	Части 1 и 2	Файл презентации на английском языке для Vijeo Citect версии 7.0
Шаблон		Папка с шаблоном учебного пособия и инструкцией его подключения
Папка «Учебное пособие AUT20 и AUT24\Новые описания и презентации»		
Vijeo Citect\Catalog\36400-ru (web).pdf		Программное обеспечение для диспетчерского управления и сбора данных (SCADA) Vijeo Citect – иформация о лицензиях, архитектура, комплекты поставки, ключи, драйверы, конвертирование приложений сторонних производителей, обучение
Vijeo Citect\Vijeo Compatibility and Interoperability Matrix (Ru).pdf		Матрица совместимости для программных продуктов Vijeo Citect, Vijeo Historian, Ampla и Vijeo Citect Batch Msnager с различными версиями операционных систем Microsoft Windows, пакетов программ Microsoft Office и программного обеспечения обмена данными
Vijeo Citect\Presentations\Vijeo Citect Overview.pdf		Презентация для Vijeo Citect (Шестаков А.А.)
Vijeo Citect\Presentations\Vijeo Citect benefits.pdf		Преимущества Vijeo Citect 7.30, презентация Шестакова А.А.
Vijeo Citect\Presentations\Vijeo Citect 7.30 & Vijeo Historian. What's new. 2nd Ed.pdf		Новые продукты и опции, 2013 год
Vijeo Citect\QuickStarting Tutorial (подпапка с файлами)		Создание нового проекта, его инициализация, создание графической страницы, графические объекты и их анимация, алармы, тренды, их просмотр, безопасность, управление проектами. Импорт оборудования, джинны и

Папка или файл	Часть, глава, прилож., раздел	Описание
		суперджинны, качество тегов. Архивы двух демонстрационных проектов
Vijeo Citect\Vijeo Citect V7.30 Documents\Vijeo_Citect_Tech_Overview_hires.pdf		Обзор Vijeo Citect
Vijeo Citect\Vijeo Citect V7.30 Documents\Equipment-Template-Guide.pdf		Шаблоны оборудования
Vijeo Citect\Vijeo Citect V7.30 Documents\ Vijeo Citect - What's New.pdf		Новое и Vijeo Citect 7.30

Важно! В папке **Учебное пособие AUT20 и AUT24\Новые описания и презентации** есть и другие полезные материалы. Настоятельно рекомендуем ознакомиться с ними

Папка «Учебное пособие AUT20 и AUT24\Демонстрационные проекты и сопутствующие файлы		
Архив проекта ActX.ctz	Часть 1, подраздел 17.3	Работа с ActX-объектами
Файлы AnaAlarm.wav, DigAlarm.wav	Глава 12	Звуковое оформление сигналов тревог
Архив проекта CicodeDebug.ctz	Глава 22	Отладка фрагментов и функций Cicode-программы
Архив проекта ComLogSuper.ctz	Глава 10	Устройства. Регистрация команд оператора (расширенные возможности форматирования выводимой информации)
Архив проекта EditTagExcel	Глава 3	Редактирование тегов с помощью приложения Microsoft Excel
Файл example.csv	Глава 3, раздел 3.5 и приложение 2	Файл описания символьных переменных проекта example.twd
Файл TwidoSoft 3.5\Example\example.twd	Приложение 2	Пример файла проекта для контроллера Twido
Файл FloorPlan.bmp	Глава 9	Импорт графики и настройка цветов
Архив проекта OvenSecurity.ctz	Глава 15	Безопасность: конфигурирование зон,

Папка или файл	Часть, глава, прилож., раздел	Описание
		привилегий и пользователей
Архив проекта OvenTraining	Глава 2	Создание нового проекта. Включение проектов
Архив проекта OvenTraining1	Глава 3	Связь сервера с устройством. Добавление тегов в проект
Архив проекта OvenTraining2	Глава 3	Создание привилегированного пользователя. Изменение и просмотр значений тегов
Архив проекта OvenTraining3	Глава 4	Создание и просмотр графических страниц
Архив проекта OvenTraining4	Глава 4	Анимация графических объектов
Архив проекта OvenTraining5	Глава 4	Динамическая анимация графических объектов
Архив проекта OvenTraining6	Глава 5	Сигналы тревог и их отображение
Архив проекта OvenTraining7	Глава 6	Тренды и их отображение
Архив проекта OvenTraining8	Глава 7	Команды и средства управления
Архив проекта OvenTraining9	Глава 8	Анализатор процессов (графический объект ActiveX)
Архив проекта OvenTraining10 и файл FloorPlan.bmp	Глава 9	Импорт графики и настройка цветов
Архив проекта OvenTraining11	Глава 9	Джинны (Genies)
Архив проекта OvenTraining12	Глава 9	Всплывающие страницы
Архив проекта OvenTraining13	Глава 10	Устройства. Регистрация команд оператора
Архив проекта OvenTraining14	Глава 11	Определение, разрешение и просмотр событий
Архив проекта OvenTraining15	Глава 12	Категории сигналов тревог
Архив проекта OvenTraining16	Глава 12	Звуковое оформление сигналов тревог
Архив проекта OvenTraining17	Глава 13	Система навигации. Конфигурирование меню
Архив проекта OvenTraining18	Глава 13	Система навигации. "Продвинутое" конфигурирование меню
Архив проекта OvenTraining19	Глава 13	Параметры навигации

Папка или файл	Часть, глава, прилож., раздел	Описание
Архив проекта OvenTraining20	Глава 14	Определение, создание и просмотр отчетов
Архив проекта OvenTraining21	Глава 15	Выполнение процессов в реальном масштабе времени
Архив проекта OvenTraining22	Глава 15	Выполнение процессов в реальном масштабе времени
Архив проекта OvenTraining23	Глава 23	Использование стандартных Cicode-функций
Архив проекта TemplateAndMenu	Глава 16	Пользовательские шаблоны, точки анимации и пользовательские меню
Архив проекта SuperGenie	Глава 17	Всплывающие окна и суперджины
Архив проекта TwidoOFS	Глава 3	Связь контроллера Twido со SCADA-системой Vijeo Citect (операционная система Windows XP)

Литература

1. Vijeo Citect. V 7.0. Vijeo Citect User Guide. November 2008 (файл ..\Program Files\Schneider Electric\Documentation\Vijeo Citect User Guide.pdf, англ. язык).
2. Файл CitectSCADA.chm (CitectSCADA Help, встроенная справка, английский язык).
3. Файл Vijeo Basics RU to print.ppt (Vijeo Citect v6.1, базовый курс, презентация на русском языке).
4. Файл Vijeo Configuration Course v7.ppt (Vijeo Citect v7, базовый курс, презентация на английском языке).
5. Давыдов В.Г. Vijeo CitectSCADA. Базовый курс: Учеб. пособие. СПб. 2008. 208 с.
6. Конфигурация Vijeo Citect. Методическое руководство. Версия 7.0. Август 2007. Выпуск 1.

Предметный указатель

O

P

T

V

Я