Узлы проекта AgavaSCADA/AgavaPLC

Редакция от 19.10.2026 · docs.kb-agava.ru

Содержание

1 Узлы проекта 4
2 Проект 5
3 Станция 6
4 Транспортная система 6
4.1 Транспорт 7
5 Система визуализации 7
5.1 Фрейм 8
5.1.1 Позиция и размеры (XCoord, YCoord, XSize, YSize) 9
5.1.2 Окна (Windows) 9
5.1.3 Номер монитора (DisplayNo) 9
5.1.4 Тип границы (BorderType) 9
5.1.5 Наследование привилегий 10
5.2 Окна 10
5.3 Форма 10
5.3.1 Модальность (Modality) 11
5.3.2 Тип границы (BorderType) 11
5.3.3 Управление заголовком (TitleEnabled, CloseEnabled) 12
5.3.4 Фон и затемнение (BackgroundColor, NeedDarkenApplication, ColorDarkenApplication) 12
5.3.5 Интервал обновления (UpdateInterval) 12
5.3.6 Обработчики событий (ActionsOnShow, ActionsOnClose) 12
5.4 Окно с композицией 12
5.4.1 Композиция (CompositionId) 13
5.4.2 Описание текущей композиции (CurrentCompositionDesc) 13
5.5 Композиция 13
5.5.1 Размеры (Size) 14
5.5.2 Кэширование (Cacheable) 14
5.5.3 Источник интервала обновления (UpdateIntervalSource) 14
5.5.4 Интервал обновления (UpdateInterval) 15
5.5.5 Обработчик событий ActionsOnLoad 15
5.6 Слой 15
5.6.1 Z-Порядок (ZOrder) 15
5.6.2 Прозрачность для действий (ActionTransparency) 16
5.6.3 Видимость (Visible) 16
5.7 Виджет 16
5.8 Меню 16
5.9 Создание многомониторных систем визуализации 17
5.10 Виджеты 17
5.10.1 Виджет «Текстовая метка» 17
5.10.2 Виджет «Условная метка» 19
5.10.3 Виджет «Изображение» 20
5.10.4 Виджет «Условное изображение» 20
5.10.5 Виджет «Движущееся изображение» 20
5.10.6 Виджет «Кнопка» 20
5.10.7 Виджет «Гистограмма» 21

5.10.8 Виджет «Исторический график» ... 21
5.10.9 Виджет «График кусочно-линейной функции» ... 23
5.10.10 Виджет «Список событий» ... 24
5.10.11 Виджет «Ввод значения» ... 25
5.10.12 Виджет «Выбор значения из списка» ... 26
5.10.13 Виджет «Флаг» ... 27
5.10.14 Виджет «Отчет» ... 27
6 Действия ... 28
6.1 Действие Воспроизведение звука ... 28
6.2 Действие Смена композиции ... 28
6.3 Действие Отображение окна ... 29
6.4 Действие Закрытие окна ... 29
6.5 Действие Переключение видимости окна ... 30
6.6 Действие Запись значения ... 30
6.7 Действие Создание события ... 30
6.7.1 Форматтеры для текста событий ... 31
6.7.2 Примеры ... 32
6.8 Действие Команда узлу ... 32
6.9 Действие Запуск процесса ... 32
6.10 Действие Авторизация (проверка подлинности) пользователя ... 33
6.11 Действие Изменение курсора ... 33
6.12 Действие Восстановление курсора ... 33
6.13 Действие Выполнение скрипта ... 34
6.14 Действие Выполнение программы ... 34
6.15 Действие Отображение меню ... 34
6.16 Действие Вызов метода экземпляра ... 35
6.17 Действие Смена пароля пользователя ... 35
7 Система тревог ... 36
7.1 Группа событий ... 36
7.2 Монитор событий ... 36
7.3 Порядок настройки системы ... 36
7.4 Окно тревог ... 37
8 Система архивирования ... 37
8.1 Состав системы ... 38
8.1.1 Узлы проекта ... 38
8.1.2 Доступные архиваторы ... 38
8.2 Порядок настройки системы ... 38
8.3 Общие свойства архиватора ... 39
8.4 Сбор и сохранение ... 40
8.5 Режим выдачи значений ... 40
8.6 Коннектор БД ... 41
9 Система журналирования ... 41
10 Система проверки прав пользователей ... 41
10.1 Роли и пользователи ... 42
10.2 Привилегии ... 42
10.3 Порядок настройки системы ... 43
11 Сигналы, источники и операции ... 43
11.1 Сигнал ... 44
11.1.1 Сигнал простой ... 44
11.1.2 Постоянная ... 44
11.1.3 Массив ... 44
11.1.4 Сигнал комплексный ... 44

11.2 Источник ... 44
11.2.1 Регистр Modbus ... 45
11.2.2 Источник индексный ... 45
11.3 Операция ... 45
11.3.1 Операция «Сложение» ... 45
11.3.2 Операция «Вычитание» ... 45
11.3.3 Операция «Умножение» ... 45
11.3.4 Операция «Деление» ... 45
11.3.5 Операция «Условие» ... 46
11.3.6 Операция «Пара условий» ... 46
11.3.7 Операция «Логическое ИЛИ» ... 46
11.3.8 Операция «Логическое И» ... 46
11.3.9 Операция «Логическое НЕ» ... 46
11.3.10 Операция «Таймер» ... 46
11.3.11 Операция «Мультиплексор» ... 47
11.3.12 Операция «Демультиплексор» ... 47
11.3.13 Операция «Генератор случайного числа» ... 47
11.3.14 Операция «Скрипт C++» ... 47
11.4 Объект ... 48
12 Задачи ... 48
12.1 Задача ... 49
12.1.1 Назначение ... 49
12.1.2 Свойства задачи ... 49
12.1.3 Режимы выполнения ... 49
12.1.4 Настройка ... 49
12.2 Обработчик события ... 49
12.2.1 Назначение ... 49
12.2.2 Настройка задачи ... 50
12.2.3 Сигнатуры функций обработки ... 50
12.2.4 Типы и коды исключений ... 51
12.2.5 Порядок выполнения ... 52
12.2.6 Продолжение работы рантайма ... 52
13 Устройства ... 53
14 Отчеты ... 53
14.1 Создание нового отчета ... 54
14.2 Редактирование макета отчета ... 54
14.2.1 Окно «Инспектор объектов» ... 55
14.2.2 Окно «Инспектор данных» ... 55
14.2.3 Окно «Инспектор скриптов» ... 55
14.2.4 Окно «Структура отчета» ... 55
14.3 Элементы отчета ... 55
14.3.1 Текст ... 55
14.3.2 Изображение ... 57
14.3.3 Фигура ... 57
14.3.4 Горизонтальная компоновка (Layout) ... 57
14.4 Разделы отчета (бэнды) ... 58
14.5 Функции ... 59
14.5.1 Агрегатные функции ... 59
14.5.2 Числовые функции ... 60

1 Узлы проекта

Узел - элементарная сущность проекта, осуществляющая хранение информации или реализующая определенный функционал. Каждый узел обладает набором свойств. Все узлы, объединенные в иерархическую структуру, образуют проект.

2 Проект

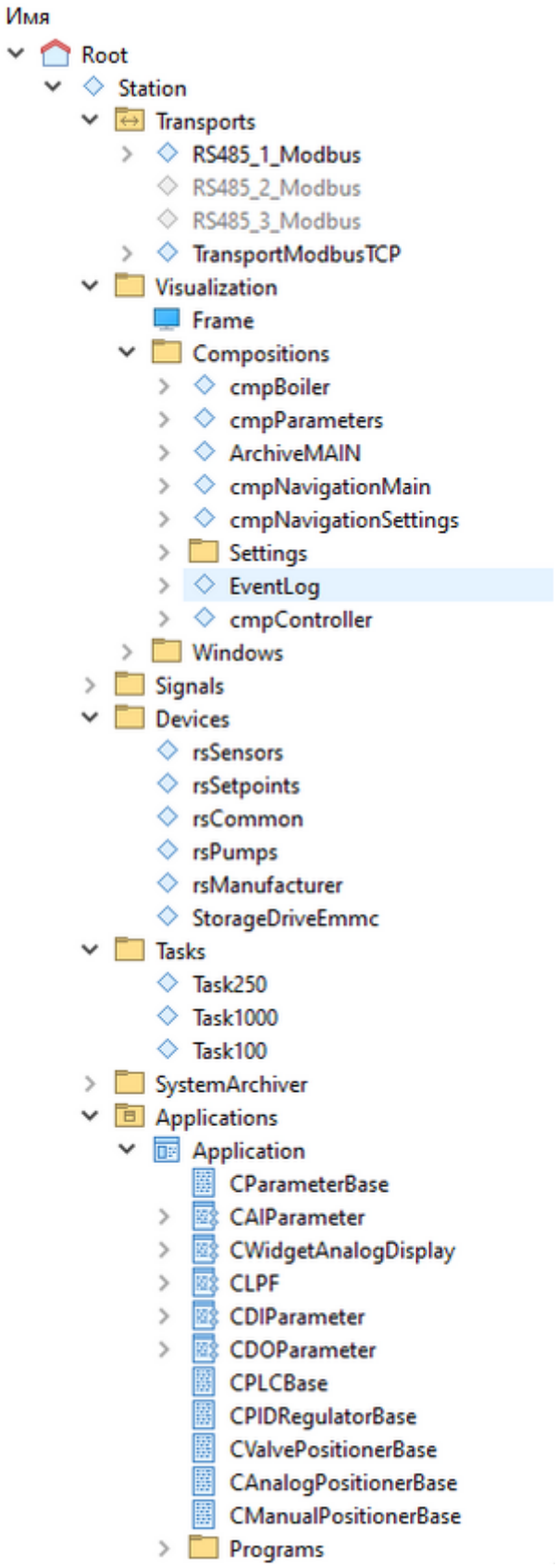


Рисунок 21 – Дерево проекта

Узел "Проект" ("Корневой узел", "Корень") предназначен для хранения общей информации о проекте и связывания других узлов проекта.

3 Станция

Станция – важнейший узел, управляющий работой собственных систем и выполняющий обмен данными с другими станциями. Физически станция представляет собой устройство, на котором исполняется проект (компьютер, панель оператора, МПР-60 или другое устройство).

Для использования в проекте доступны несколько типов станций. например:

- **Станция** - общий тип станции. Проекты с такой станцией могут выполняться на большинстве устройств - контроллеры, панели, персональные компьютеры. Чаще всего используется для проектов, предназначенных для выполнения на персональном компьютере.
- **Панель ПО-40** - специализированный тип для проектов на панелях оператора ПО-40. Станция такого типа по отношению к общему дополнительно предоставляет возможность управления оборудованием панели оператора (светодиоды индикации и др.), использования корзины субмодулей.
- **Панель ПО-50** - специализированный тип для проектов на панелях оператора ПО-50. Станция такого типа по отношению к общему дополнительно предоставляет возможность управления оборудованием панели оператора (светодиоды индикации и др.).
- **Контроллер МПР-60** - специализированный тип для проектов на программируемых реле МПР-60. Станция такого типа по отношению к общему дополнительно предоставляет возможность управления оборудованием устройства (светодиоды индикации и др.), использования корзины субмодулей.
- **Контроллер ПЛК-70.А** - специализированный тип для проектов, предназначенных для контроллеров АГАВА ПЛК-70.А. Станция такого типа по отношению к общему дополнительно предоставляет возможность управления оборудованием контроллера (светодиоды индикации и др.), использования корзины субмодулей.

При построении больших проектов станций в проекте может быть несколько, и они могут выполнять разные задачи. Например, станция-сервер выполняет сбор и регистрацию данных, клиентские станции выполняют визуализацию данных на рабочих местах и т. д.

Станции подчиняются узлы типа "Система" - Транспортная система, Система архивирования, Система визуализации и т. д.

4 Транспортная система

Транспортная система управляет передачей данных по различным каналам связи. Система управляет работой "Транспортов" - узлов, выполняющих обмен данными с устройствами по определенному протоколу.

С одной стороны, транспортная система обращается непосредственно к сигналам и каналам устройства, а с другой стороны, представляет их в виде неких универсальных единиц – узлов-источников. Каждый источник однозначно соответствует определенному сигналу или параметру устройства. При изменении сигнала меняется и значение источника. При записи данных в источник производится подача команды устройству на установку определенного значения конкретного параметра.

Транспортная система допускает использование нескольких транспортов, что дает возможность одновременного обмена данными с большим количеством устройств по разным протоколам.

Подробное описание транспортной системы приведено в отдельном документе Система транспортная AgavaSCADA/AgavaPLC.

4.1 Транспорт

Транспорт предназначен для обмена данными с устройствами по определенному протоколу. Один транспорт, в зависимости от типа, позволяет обмениваться данными с одним или несколькими устройствами.

Доступны для использования следующие виды транспортов:

- Транспорт Modbus-TCP;
- Транспорт Modbus-RTU;
- Транспорт SNMP;
- Транспорт OPC UA;
- Транспорт OPC DA (только для Windows);
- Транспорт MQTT;
- Последовательный транспорт (устаревший);
- TCP/IP транспорт (устаревший).

Подробное описание транспортов приведено в отдельном документе "Транспорты AgavaSCADA/AgavaPLC".

5 Система визуализации

	Система визуализации недоступна в проектах для контроллеров МПР

Задачей этой системы является отображение данных в виде набора мнемонических схем (экранов, мнемосхем), а также предоставление пользователю возможности управления.

Каждый экран позволяет в схематичном, понятном пользователю виде представить участок технологической схемы с отображением необходимых параметров технологического процесса.

Изображение на экране строится с помощью окон, на которых размещаются объекты визуализации – виджеты. Виджетами могут быть текстовые метки, изображения, в том числе анимированные, элементы управления, графики и т. д. Виджеты можно объединять в слои для удобного управления отображением виджетов.

Виджеты могут быть статическими и динамическими. Динамические виджеты привязаны к одному или нескольким сигналам и их отображение зависит от значений этих сигналов. Статические виджеты не имеют привязки к сигналам и их изображение всегда постоянно.

Для перехода по экранам, изменения значений параметров и иного управления служат специальные виджеты: кнопки управления, поля ввода и другие.

Для повышения быстродействия и увеличения комфортности работы с системой система визуализации хранит в памяти кэш из нескольких последних открытых экранов. Когда при последовательном открытии экранов этот кэш заполняется, наиболее старые экраны удаляются из него. Это связано с тем, что, как правило, в работе используются несколько одних и тех же экранов, которые отображают сводную информацию о техпроцессе и

оператор в штатном режиме переключается только между ними. Сохраняя их в памяти, система ускоряет переключение между ними, поскольку при этом не требуется каждый раз создавать экран, все находящиеся на нем объекты и запускать процессы обновления параметров.

Когда экран переходит в кэш, обновление параметров на нем прекращается для снижения нагрузки на центральный процессор. Если все же необходимо, чтобы экран всегда производил обновление данных (по разным причинам), даже находясь в кэше, необходимо в его настройках включить опцию «Постоянно обновляемый».

Подчиненные типы узлов:

- фрейм;
- окно;
- композиция.

5.1 Фрейм

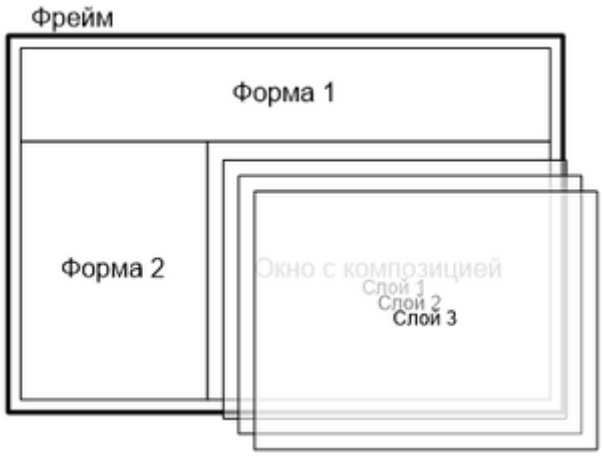


Рисунок 23 – Пример структуры сложного фрейма, состоящего из двух форм и одного окна с многослойной композицией

Фрейм – узел, предназначенный для группировки окон, отображаемых на определенном дисплее. Обеспечивает организацию окон в пространстве экрана, управление мониторами и границами отображения.

Если необходимо отобразить информацию на нескольких дисплеях одновременно, то для каждого из них нужно использовать отдельный фрейм. Окон во фрейме может быть несколько, однако они не могут накладываться друг на друга.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Позиция X	XCoord	int	Координата X левого верхнего угла фрейма	Целое число
Позиция Y	YCoord	int	Координата Y левого верхнего угла фрейма	Целое число
Ширина	XSize	int	Ширина фрейма в пикселях	Целое число > 0
Высота	YSize	int	Высота фрейма в пикселях	Целое число > 0

Окна	Windows	NodesArray	Список окон, принадлежащих фрейму	Узлы типа: Window
Номер монитора	DisplayNo	int	Номер монитора для отображения фрейма	0 - по умолчанию ≥1 - конкретные мониторы
Тип границы	BorderType	enum	Тип границы фрейма	0: Не определен (ошибка) 1: Отсутствует 2: Фиксированная

5.1.1 Позиция и размеры (XCoord, YCoord, XSize, YSize)

- **Тип:** int.
- **Описание:** Определяют положение и размеры фрейма на экране.
- **Значения по умолчанию:** (0, 0, 640, 480).
- **Применение:** Задание области экрана, которую занимает фрейм.

5.1.2 Окна (Windows)

Тип: NodesArray.

Описание: Коллекция окон, которые управляются данным фреймом. Окна могут добавляться только из существующих узлов.

Особенности:

- Поддерживаются только узлы типа Window.

5.1.3 Номер монитора (DisplayNo)

Тип: int.

Описание: Определяет, на каком мониторе системы отображается фрейм. Значение 0 соответствует монитору по умолчанию.

Использование:

- 0 - основной монитор.
- 1, 2, ... - дополнительные мониторы.
- Свойство важно для многомониторных конфигураций.

5.1.4 Тип границы (BorderType)

Тип: enum.

Описание: Определяет тип обрамления фрейма. Влияет на внешний вид и поведение границ.

Значения:

- **Отсутствует** - фрейм без видимых границ.
- **Фиксированная** - стандартная граница окна.

- **Не определен** - используется только для обработки ошибок.

5.1.5 Наследование привилегий

Фрейм автоматически добавляет доступные привилегии для работы с окнами:

- `OpenWindow` - привилегия на открытие окон.
- `CloseWindow` - привилегия на закрытие окон.

5.2 Окна

Окно – узел, осуществляющий отображение видеокадра (мнемонической схемы).

Существуют несколько видов окон:

1. «Форма», предназначенная для вывода простого набора данных без необходимости управления видимостью или внешним видом его элементов. Для вывода информации в такое окно кроме создания окна и добавления к нему виджетов ничего более не требуется.

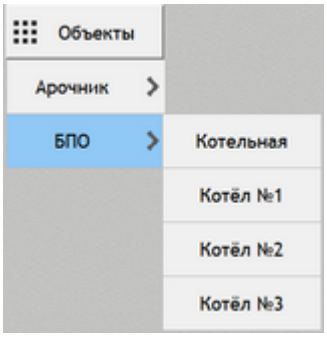


Рисунок 24 — Внешний вид меню в среде исполнения

2. «Окно с композицией», предназначенное для вывода большого количества информации и позволяющее организовать группировку элементов по слоям, управлять внешним видом элементов, видимостью слоев и отдельных элементов, а также дающее возможность регулирования работы пользователя с элементами. Для вывода информации в такое окно нужно к окну привязать композицию, содержащую минимум один слой, в который уже добавляются виджеты.

5.3 Форма

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения	Режим доступа
Модальность	Modality	enum	Режим модальности окна	0: Не определен (ошибка) 1: Немодальное 2: Модальное (Фрейм) 3: Модальное (Система)	Чтение/ Запись

Тип границы	BorderType	enum	Тип границы окна	0: Не определен (ошибка) 1: Фиксированная 2: Изменяемая 3: Отсутствует	Чтение/ Запись
Отображать строку заголовка	TitleEnabled	bool	Отображать строку заголовка	true/false	Чтение/ Запись
Отображать кнопку "Закрыть"	CloseEnabled	bool	Отображать кнопку "Закрыть"	true/false	Чтение/ Запись
Цвет фона	BackgroundColor	NColor	Цвет фона окна	Цвет в формате RGBA	Чтение/ Запись
Затемнение	NeedDarkenApplication	bool	Затемнение других окон приложения при открытии этого окна	true/false	Чтение/ Запись
Цвет затемнения	ColorDarkenApplication	NColor	Цвет затемнения	Цвет в формате RGBA	Чтение/ Запись
Интервал обновления	UpdateInterval	int	Интервал обновления виджетов, мс	0 - без задержки >0 - интервал в мс	Чтение/ Запись
Действия OnShow	ActionsOnShow	NodesContainer	Действия при показе окна	Узлы типа: Action	Чтение/ Запись
Действия OnClose	ActionsOnClose	NodesContainer	Действия при закрытии окна	Узлы типа: Action	Чтение/ Запись

5.3.1 Модальность (Modality)

Тип: enum.

Описание: Определяет поведение окна относительно других окон приложения.

Значения:

- **Немодальное** - позволяет взаимодействовать с другими окнами.
- **Модальное (Фрейм)** - блокирует взаимодействие с окнами в том же фрейме.
- **Модальное (Система)** - блокирует все окна приложения.

5.3.2 Тип границы (BorderType)

Тип: enum.

Описание: Определяет тип и поведение границ окна.

Значения:

- **Фиксированная** - окно нельзя изменять размер.
- **Изменяемая** - пользователь может изменять размер окна.
- **Отсутствует** - окно без видимых границ.

5.3.3 Управление заголовком (TitleEnabled, CloseEnabled)

Тип: bool.

Описание: Контролируют отображение элементов заголовка окна.

5.3.4 Фон и затемнение (BackgroundColor, NeedDarkenApplication, ColorDarkenApplication)

Тип: NColor, bool, NColor.

Описание: Управляют визуальным оформлением окна и его влиянием на другие окна.

Особенности:

- Затемнение применяется только при NeedDarkenApplication = true.
- Цвет затемнения по умолчанию: черный с прозрачностью 50%.

5.3.5 Интервал обновления (UpdateInterval)

Тип: int.

Описание: Определяет частоту обновления виджетов в окне.

Рекомендации:

- 0 - 100 мс для композиций с критичным временем отклика.
- 100 - 500 мс хорошо подходит для большинства случаев.
- более 1000 мс - для статических или редко меняющихся данных.

5.3.6 Обработчики событий (ActionsOnShow, ActionsOnClose)

Тип: NodesContainer.

Описание: Коллекции действий, выполняемых при показе и закрытии окна.

5.4 Окно с композицией

Данный узел имеет те же свойства, что и форма, а также собственные:

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения	Режим доступа
Композиция	CompositionId	NodeLink	Основная композиция окна	Узлы типа: CompositionLayered, CompositionTemplated	Чтение/ Запись

Описание текущей композиции	CurrentCompositionDesc	string	Описание текущей загруженной композиции	Автоматически формируется	Только чтение

5.4.1 Композиция (CompositionId)

Тип: NodeLink.

Описание: Ссылка на композицию, отображаемую в окне.

Поддерживаемые типы: CompositionLayered, CompositionTemplated.

5.4.2 Описание текущей композиции (CurrentCompositionDesc)

Тип: string.

Описание: Автоматически обновляемое описание загруженной композиции.

Особенности: Свойство только для чтения, обновляется системой автоматически.

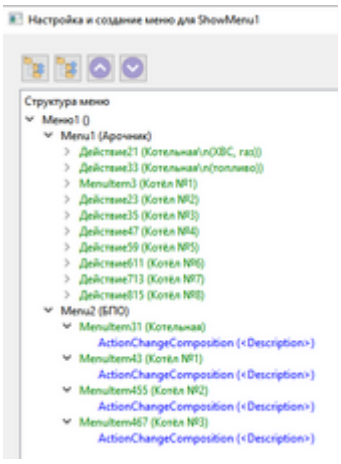


Рисунок 25 — Настройка меню в среде разработки

5.5 Композиция

Композиция – узел, представляющий составленную при разработке проекта мнемоническую схему в виде набора элементов визуализации (виджетов), сгруппированных в один или несколько слоев.

Композиция дает возможность организовать виджеты в слои, видимостью которых можно управлять во время работы проекта.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения	Режим доступа
Размеры	Size	NSize	Размеры композиции	Структура {ширина, высота}	Чтение/ Запись

Кэширование	Cacheable	bool	Кэширование данных композиции для быстрого повторного открытия	true/false	Чтение/ Запись
Источник интервала обновления	UpdateIntervalSource	enum	Определяет источник интервала обновления виджетов	0: Окно 1: Композиция	Чтение/ Запись
Интервал обновления	UpdateInterval	int	Интервал обновления виджетов в миллисекундах	Целое число ≥ 0	Чтение/ Запись
Действия OnLoad	ActionsOnLoad	NodesContainer	Действия при загрузке композиции	Узлы типа: Action	Чтение/ Запись

5.5.1 Размеры (Size)

Тип: NSize (ширина, высота).

Описание: Определяет размеры композиции в пикселях. По умолчанию установлены значения 800x600.

5.5.2 Кэширование (Cacheable)

Тип: bool.

Описание: Включает или отключает кэширование данных композиции. При включенном кэшировании композиция сохраняется в памяти для быстрого повторного открытия.

Преимущества кэширования:

- Быстрое переключение между композициями.
- Снижение нагрузки на систему при частом использовании.
- Сохранение состояния элементов.

Рекомендации: Включать для часто используемых композиций, отключать для редко используемых или ресурсоемких.

5.5.3 Источник интервала обновления (UpdateIntervalSource)

Тип: enum.

Описание: Определяет, откуда виджеты композиции получают интервал обновления.

Значения:

- **Окно** - интервал обновления наследуется от родительского окна.
- **Композиция** - используется собственный интервал обновления композиции.

Применение: Централизованное управление производительностью через окно или индивидуальная настройка для каждой композиции.

5.5.4 Интервал обновления (UpdateInterval)

Тип: int.

Описание: Задает интервал обновления виджетов в миллисекундах. При значении 0 отрисовка виджетов выполняется без задержки (максимальная частота).

Особенности:

- 0 - обновление без задержек (может создать высокую нагрузку).
- > 0 - обновление с заданным интервалом. Учитывается только при UpdateIntervalSource = "Композиция"

Рекомендации:

- 0 - 100 мс для композиций с критичным временем отклика.
- 100 - 500 мс хорошо подходит для большинства случаев.
- более 1000 мс - для статических или редко меняющихся данных.

5.5.5 Обработчик событий ActionsOnLoad

Тип: NodesContainer.

Описание: Коллекции действий, выполняемых после загрузки композиции `AS 1.6.27+`.

5.6 Слой

Слой – набор виджетов, объединенных каким-то общим структурным признаком. В многослойных композициях слой обеспечивает организацию виджетов в z-порядке, управление видимостью и взаимодействием с пользователем. С помошью свойства "Видимость" слоя можно управлять видимостью виджетов, размещенных на слое.

Свойства

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения	Режим доступа
Z-Порядок	ZOrder	int	Порядок слоя в композиции	Целое число (чем выше значение, тем выше слой)	Чтение/ Запись
Прозрачность для действий	ActionTransparency	enum	Определяет поведение слоя при взаимодействии с пользователем	0: Не определен (ошибка) 1: Непрозрачный 2: Прозрачный	Чтение/ Запись
Видимость	Visible	bool	Видимость слоя и всех его виджетов	true/false	Чтение/ Запись

5.6.1 Z-Порядок (ZOrder)

Тип: int

Описание: Определяет порядок отображения слоя относительно других слоев в композиции. Слои с большим значением ZOrder отображаются поверх слоев с меньшим значением.

Пример использования:

- Фоновый слой: ZOrder = 1.
- Основной контент: ZOrder = 2.
- Контекстные виджеты: ZOrder = 3.

5.6.2 Прозрачность для действий (ActionTransparency)

Тип: enum

Описание: Управляет взаимодействием слоя с пользовательскими действиями (клики, наведения и т.д.)

Значения:

- **Непрозрачный** - слой полностью перехватывает пользовательские действия, элементы под ним недоступны, даже если они видимы.
- **Прозрачный** - пользовательские действия проходят сквозь слой к элементам ниже.

Обратите внимание: если на слое размещается изображение с прозрачной областью, то такие виджеты не позволяют пользователю взаимодействовать с видимыми и размещенными под ним виджетами.

5.6.3 Видимость (Visible)

Тип: bool

Описание: Определяет видимость всего слоя и всех его дочерних виджетов. При значении false слой и все его содержимое становится невидимым.

Применение: Временное скрытие виджетов или переключение их видимости.

5.7 Виджет

Виджет – элемент визуализации, предназначенный для отображения информации в заданном виде.

5.8 Меню

Меню – элемент визуализации, сочетающий в себе несколько виджетов «Кнопка». Используется для более удобного управления проектом (см. Рисунок 24).

Для добавления меню, необходимо в среде разработки назначить действие «Отображение меню» для кнопки (или любого другого виджета). После добавления действия необходимо выделить его и нажать на кнопку настроек на панели вверху окна действий. Открывшееся окно (см. Рисунок 25) содержит дерево со структурой меню, в который можно добавлять элементы двух типов:

- меню – элемент, который позволяет группировать элементы меню, выводя их в виде раскрывающегося списка. На рисунках 24 и 25 этим элементом являются кнопки «Арочник» и «БПО».

- элемент меню – элемент, который предоставляет функционал, аналогичный виджету «Кнопка». Для добавления действия, выполняющегося при нажатии на элемент меню в среде исполнения, необходимо нажать на него ПКМ в дереве структуры меню и выбрать "Создать действие". На рисунках 24 и 25 этим элементом являются кнопки «Котельная», «Котёл №1», «Котёл №2» и «Котёл №3».

5.9 Создание многомониторных систем визуализации

AgavaSCADA предоставляет возможность реализации систем визуализации в составе проектов, использующих для отображения информации более одного дисплея.

Для вывода информации в окнах на нескольких дисплеях в проект необходимо внести следующие корректировки относительно проекта, использующего только один дисплей:

1. Добавить дополнительные фреймы, чтобы их количество в проекте соответствовало количеству дисплеев. Каждому фрейму назначить порядковый номер, начиная с 1. Фреймы связываются с дисплеями по номеру.
2. Добавить дополнительное окно (или окна, если в исходном фрейме их несколько или дополнительных фреймов больше одного).
3. Добавить в дополнительные фреймы созданные на этапе 2 окна, чтобы повторить структуру основоного фрейма.

Базовая настройка на этом закончена. После запуска такого проекта на двух или более дисплеях должны отображаться созданные окна.

Если в проекте используется навигация по окнам, то необходимо в каждом окне, включенном в фрейм, скорректировать действия, размещаемые на кнопках, меню или другх виджетах таким образом, чтобы действия открывали окна или загружали композиции в нужном окне.

5.10 Виджеты

5.10.1 Виджет «Текстовая метка»

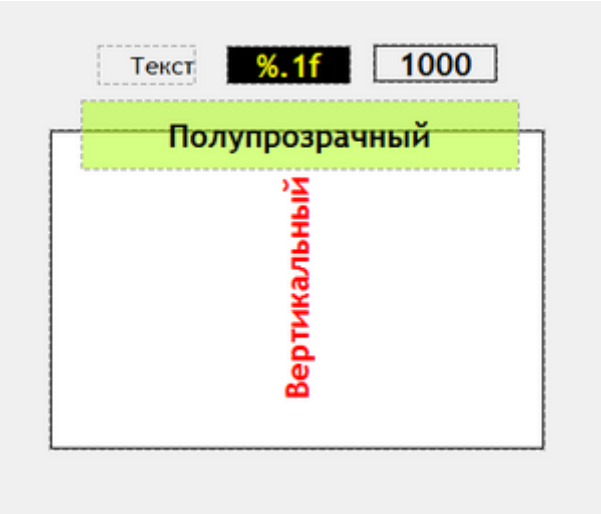


Рисунок 26 – Примеры оформления виджетов типа "Текстовая метка"

Виджет «Текстовая метка» – наиболее часто применяемый виджет, с помощью которого можно реализовать разные способы отображения текстовой информации.

Виджет «Текстовая метка» может использоваться для отображения статичного текста, значений параметров (одного или нескольких) и их комбинаций.

При необходимости отображения значений параметров они указываются в свойстве «Источник». Для задания определенного формата отображения используются спецификаторы формата в стиле языка «Си»:

- %f – значение в виде числа с плавающей точкой;
- %s – строка;
- %i – целое число со знаком;
- %u – целое число без знака;
- %d – целое число со знаком в десятичной системе исчисления.

Для разных типов свойств необходимо использовать строго определенные для них спецификаторы, с другими спецификаторами значения выводиться не будут.

Для отображения знака «%» необходимо указать его дважды: «%%».

5.10.1.1 Отображение нескольких значений

Одна текстовая метка может использоваться для отображения нескольких значений. Для этого все значения, которые нужно отобразить, указываются в свойстве «Источник», а в свойстве «Текст» указываются последовательно несколько спецификаторов формата соответственно порядку перечисления значений в свойстве «Источник».

Пример: текстовая метка, отображающая текущую позицию в очереди:

- Текст: `Позиция %i из %i`
- Источник: два узла из проекта.

5.10.1.2 Модификаторы спецификаторов формата

5.10.1.2.1 Модификатор минимальной ширины поля

Целое число, расположенное между знаком % и кодом формата, играет роль модификатора минимальной ширины поля. Если указан модификатор минимальной ширины поля, то, чтобы ширина поля вывода была не меньше указанной минимальной величины, вывод при необходимости будет дополнен пробелами. Если же выводятся строки или числа, которые длиннее указанного минимума, то они все равно будут отображаться полностью. По умолчанию для дополнения используются пробелы. А если для этого надо использовать нули, то перед модификатором ширины поля следует поместить 0. Например, %05i означает, что любое число, количество цифр которого меньше пяти, будет дополнено таким количеством нулей, чтобы состоять из пяти цифр.

Примеры

Значение	Спецификатор формата	Выводимое значение
10.123	%i	10
10.123	%d	10
10.123	%05i	00010
10.123	%f	10.123000
10.123	%.1f	10.1
10.123	%10f	<пробел>10.123000

10.123	%012f	00010.123000

5.10.1.2.2 Модификатор точности

Модификатор точности следует за модификатором минимальной ширины поля (если таковой имеется). Он состоит из точки и расположенного за ней целого числа. Значение этого модификатора зависит от типа данных, к которым его применяют.

Когда модификатор точности применяется к данным с плавающей точкой, для преобразования которых используются спецификаторы преобразования %f, %e или %E, то он определяет количество выводимых десятичных разрядов. Например, %10.4f означает, что ширина поля вывода будет не менее 10 символов, причем для десятичных разрядов будет отведено четыре позиции.

Если модификатор точности применяется к %g или %G, то он определяет количество значащих цифр.

Примененный к строкам, модификатор точности определяет максимальную длину поля. Например, %5.7s означает, что длина выводимой строки будет составлять минимум пять и максимум семь символов. Если строка окажется длиннее, чем максимальная длина поля, то конечные символы выводиться не будут.

Если модификатор точности применяется к целым типам, то он определяет минимальное количество цифр, которые будут выведены для каждого из чисел. Чтобы получилось требуемое количество цифр, добавляется некоторое количество ведущих нулей.

Примеры:

Значение	**Спецификатор формата**	**Выводимое значение**
10.123	%f	10.123000
10.123	%.1f	10.1
10.123	%10.4f	<пробел><пробел><пробел>10.1230
10.123	%010.3f	000010.123
Это простая проверка	%10.15s	Это простая про

5.10.2 Виджет «Условная метка»



Рисунок 27 – Список сопоставлений виджета "Условная метка"

Виджет «Условная метка» позволяет изменять выводимую строку в соответствии с изменениями привязанной к ней переменной.

Задать соответствие текста необходимому условию можно в свойстве «Список сопоставлений».

5.10.3 Виджет «Изображение»

Виджет «Изображение» предназначен для отображения на окнах статичных изображений форматов png, gif, jpg, bmp или анимированных изображений в формате gif. Отображаемая картинка выбирается в разделе «Оформление» - «Изображение». Она должна находиться в папке проекта.

5.10.4 Виджет «Условное изображение»

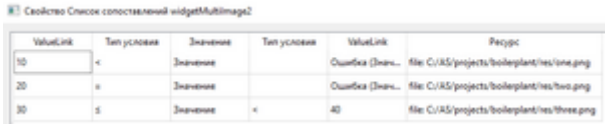


Рисунок 28 – Список сопоставлений виджета "Условное изображение"

Виджет «Условное изображение» позволяет изменять отображаемое изображение в соответствии с изменениями привязанной к нему переменной.

Задать соответствие изображения необходимому условию можно в свойстве «Список сопоставлений».

5.10.5 Виджет «Движущееся изображение»

Виджет предназначен для отображения на окнах изображений, перемещающихся в пределах области виджета. В основном используются на экране спящего режима.

Отображаемая картинка выбирается в разделе «Оформление» - «Изображение». Она должна находиться в папке проекта.

5.10.6 Виджет «Кнопка»

Виджет «Кнопка» предназначен для реализации функций управления:

- ввод данных;
- передача управляющих воздействий;
- смена режима работы АРМ или иного устройства, исполняющего проект.

Выполнение всех функций виджета обеспечивается с помощью узлов типа «Действие», добавляемых в соответствующее свойство (см. раздел #Действия).

5.10.7 Виджет «Гистограмма»

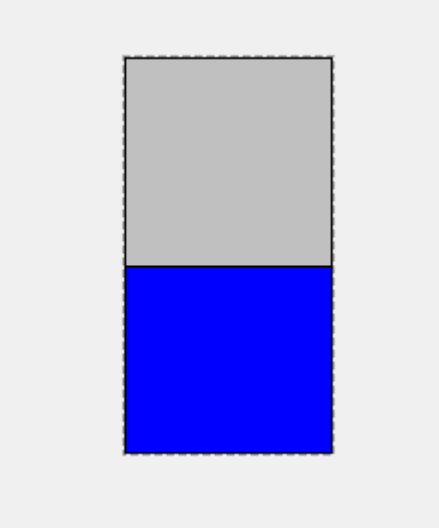


Рисунок 29 – Виджет "Гистограмма"

Виджет предназначен для отображения значений сигналов в виде столбчатой диаграммы.

Виджету необходимо задать источник (регистр/сигнал, с которого он будет получать данные), а также минимальное и максимальное значения.

5.10.8 Виджет «Исторический график»

Виджет «Исторический график» предназначен для отображения графиков параметров на двухкоординатной области изображения с привязкой к осям времени и значений.

Виджет может работать с данными, архивируемыми в БД, и напрямую с сигналами, значения которых вычисляются в ходе работы.

При работе с архивируемыми данными интервал обновления графика зависит от режима выдачи значения у базы данных: при выборе сохраняемых в БД он будет соответствовать интервалу архивирования, а при сохраненных в БД - интервалу сохранения БД на диск. Независимо от режима выдачи, график не может обновляться меньше чем за 300 мс.

При работе напрямую с сигналами график обновляется при изменении значения сигнала, но также не чаще чем раз в 300 мс.

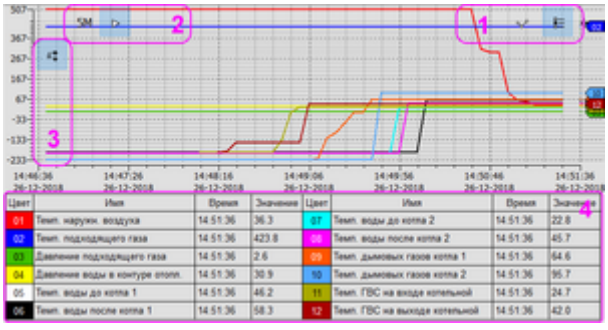


Рисунок 30 – Виджет «Исторический график». Области кнопок-пиктограмм. В нижней части отображена легенда

Виджет «Исторический график» состоит из нескольких областей:

- область осей X, Y;

- область легенды, содержащей список кривых и их параметры (цвет кривой, наименование кривой, текущее значение, время последнего зарегистрированного значения);
- область вывода данных.

На виджете имеются несколько областей, содержащих кнопки-пиктограммы управления внешним видом графика (см. Рисунок 28):

1. область пиктограмм графика;
2. область пиктограмм оси времени (X);
3. область пиктограмм оси значений (Y);
4. область легенды.

В области пиктограмм графика слева направо расположены следующие кнопки-пиктограммы:

- кнопка переключения режима отображения точек на всех кривых;
- кнопка переключения видимости легенды;
- кнопка включения режима визирной линии.

В области пиктограмм оси времени (X) слева направо расположены следующие кнопки-пиктограммы:

- кнопка открытия окна выбора интервала времени для оси X, текст кнопки отображает текущий выбранный интервал, например: 5М – 5 минут, 6Ч – 6 часов, 30М – 30 минут;
- кнопка-индикатор включения следящего режима.

В области пиктограмм оси значений (Y) расположена кнопка-пиктограмма «A» – кнопка переключения режима автоматического масштабирования значений.

5.10.8.1 Легенда

Цвет	Имя	Время	Значение	Цвет	Имя	Время	Значение
01	Темп. наружн. воздуха	15:07:38	36.3	07	Темп. воды до котла 2	15:07:38	22.8
02	Темп. подходящего газа	15:07:38	423.8	08	Темп. воды после котла 2	15:07:38	45.7
03	Давление подходящего газа	15:07:38	2.6	09	Темп. дымовых газов котла 1	15:07:38	64.6
04	Давление воды в контуре отопл.	15:07:38	30.9	10	Темп. дымовых газов котла 2	15:07:38	95.7
05	Темп. воды до котла 1	15:07:38	46.2	11	Темп. ГВС на входе котельной	15:07:38	24.7
06	Темп. воды после котла 1	15:07:38	58.3	12	Темп. ГВС на выходе котельной	15:07:38	42.0

Рисунок 31 – Легенда с четырьмя выключенными параметрами

Легенда – специальная область в нижней части экрана архива, предназначенная для отображения параметров (наименование, текущее значение и т. д.) выводимых графиков в виде таблицы. Для ознакомления с примерным видом легенды см. Рисунок 31. Информация о каждом сигнале в легенде выводится в несколько колонок:

- «Цвет» – индицирует цвет, которым выводится соответствующая кривая на графике, колонка «Цвет» позволяет нажатием переключать видимость соответствующей кривой на графике (см. Рисунок 29). Также в колонке "Цвет" отображается номер пера, которым выводится кривая параметра на графике;
- «Имя» («Наименование») – содержит наименование параметра;
- «Время» – содержит время последнего зарегистрированного значения параметра;
- «Значение» – содержит последнее зарегистрированное значение параметра.

5.10.8.2 Режимы просмотра графиков

Пользователю доступно два режима просмотра графиков:

- следящий режим;

- режим просмотра истории.

В следящем режиме на графике всегда отображаются актуальные значения параметров, тренд периодически дочитывает поступившие данные из базы и обновляет графики.

В режиме просмотра истории на графике отображаются исторические значения параметров. Просмотр истории технологического процесса осуществляется с помощью кнопок перемещения по осям и кнопок изменения масштаба.

Режим просмотра истории включается автоматически при прокрутке графика назад по времени.

5.10.8.3 Визирная линия

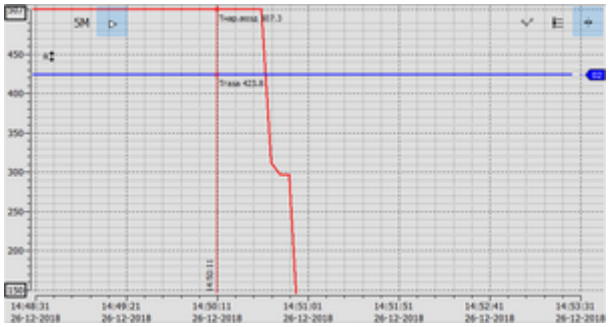


Рисунок 32 – График в режиме просмотра истории с установленной визирной линией

Для просмотра значений параметров в нужный момент времени служит визирная линия, отображаемая на графике как вертикальная линия.

При щелчке левой кнопкой мыши (ЛКМ) в области графиков визирная линия будет отображена в выбранном месте графика.

В точке пересечения визирной линии и кривых параметров отображаются значения этих параметров. Также в нижней части области графиков, возле оси времени, визирная линия отображает время.

5.10.9 Виджет «График кусочно-линейной функции»

Виджет предназначен для отображения графика зависимости двух параметров, расположенном на осях X и Y.

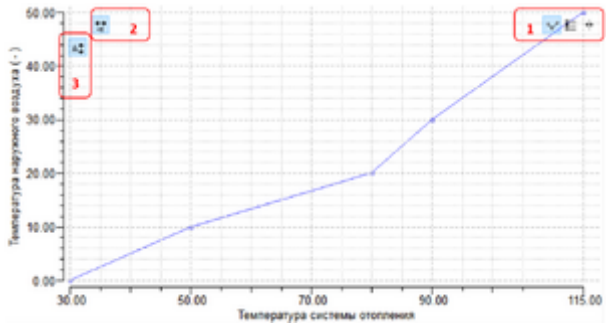


Рисунок 33 – Виджет «График кусочно-линейной функции». Области кнопок-пиктограмм.

Виджет состоит из нескольких областей:

- область осей X, Y;

- область легенды, содержащей список кривых и их параметры (цвет кривой, наименование кривой, текущее значение, время последнего зарегистрированного значения);
- область вывода графика.

На области вывода графика виджета имеются несколько областей, содержащих кнопки-пиктограммы управления внешним видом графика (см. Рисунок 33):

1. Область пиктограмм графика.
2. Область пиктограмм оси первого параметра (X).
3. Область пиктограмм оси второго параметра (Y).

В области пиктограмм графика расположены следующие кнопки-пиктограммы:

- кнопка переключения режима отображения точек на всех кривых;
- кнопка переключения видимости легенды;
- кнопка включения режима визирной линии.

В области пиктограмм оси первого (X) и второго (Y) параметров расположены кнопки-пиктограммы «A» – кнопка переключения режима автоматического масштабирования значений.

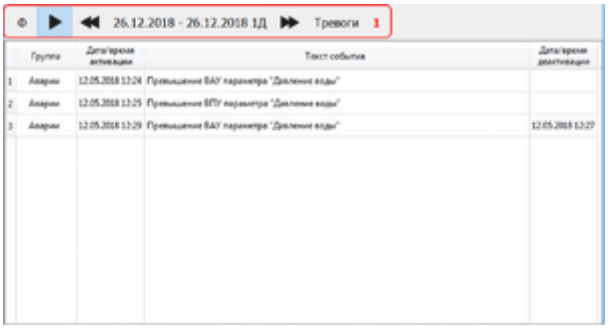


Рисунок 34 – Виджет "Список событий"

5.10.10 Виджет «Список событий»

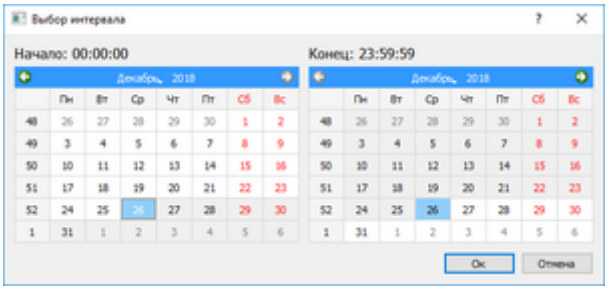


Рисунок 35 – Окно выбора интервала

Виджет предназначен для отображения регистрируемых событий в работе.

В верхней части виджета расположена панель управления (1).

Панель управления содержит органы управления журналом:

- кнопка отображения фильтра событий;
- кнопка включения следящего режима журнала;
- кнопка изменения интервала на один день назад;
- кнопка-индикатор выбора интервала, за который отображаются события;
- кнопка изменения интервала на один день вперед;

- кнопка вызова окна со списком активных тревог.

Ниже панели управления, в основном поле экрана, расположена таблица, содержащая зарегистрированные события.

Заголовок таблицы позволяет изменять порядок столбцов и включать сортировку по столбцам. Например, возможно включение сортировки событий сначала по времени, затем по группе. Заголовок таблицы имеет возможность отключения / включения отображения любого столбца.

При большом количестве записей в правой части таблицы появляется полоса прокрутки, позволяющая пролистывать список событий.

Нажатие на кнопку-индикатор выбора интервала вызывает окно выбора интервала (см. Рисунок 35):

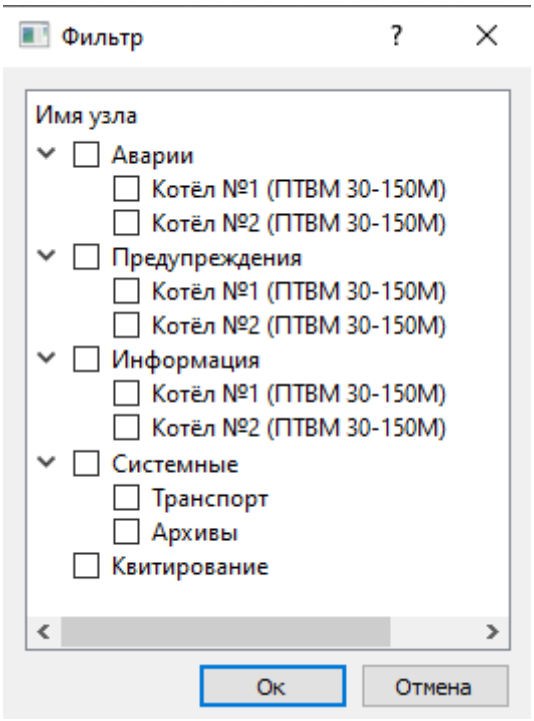


Рисунок 36 – Окно "Фильтр"

После выбора нужного интервала для сохранения выбора необходимо нажать кнопку «OK». После сохранения будут отображены зарегистрированные события за выбранный интервал.

Нажатие на кнопку отображения окна фильтра вызывает соответствующее окно. Окно фильтра позволяет изменить содержимое основного окна журнала событий путем выбора групп и подгрупп событий, перечисленных в окне фильтра (см. Рисунок 36).

После выбора нужных групп и подгрупп для сохранения выбора необходимо нажать кнопку «OK». После сохранения будут отображены зарегистрированные события, принадлежащих выбранным группам.

5.10.11 Виджет «Ввод значения»

Виджет «Ввод значения» предназначен для отображения значения привязанного сигнала и его редактирования.

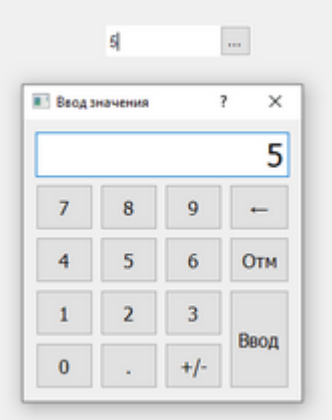


Рисунок 37 – Виджет «Ввод значения» с отображением экранной цифровой клавиатуры

После выбора узла для чтения/записи и соответствующего типа значения необходимо выбрать нужный тип ввода в разделе «Оформление».

Доступны следующие типы ввода:

- кнопки изменения – при нажатии на виджет появляются кнопки увеличения и уменьшения на заданный шаг изменения;
- кнопка редактирования – при нажатии на виджет появляется кнопка, при клике на которую открывается экранная цифровая клавиатура;

Виджет имеет возможность проверки вводимых значений, при активации которой запись будет осуществляться только тогда, когда записываемое значение будет находиться в диапазоне между заданными минимальным и максимальными значениями.

5.10.12 Виджет «Выбор значения из списка»

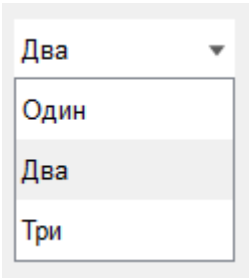


Рисунок 38 – Виджет «Выбор значения из списка»

Виджет «Выбор значения из списка» предназначен для отображения значения привязанного сигнала и его изменения в удобном для пользователя текстовом виде (см. Рисунок 37).

С помощью свойства «Список сопоставлений» задается соответствие между значением сигнала и выводимой в виджете строкой.

При нажатии на кнопку в правой части виджета отображается список всех сопоставлений. При выборе нужного элемента в списке в установленный узел производится запись нового значения.

5.10.13 Виджет «Флаг»

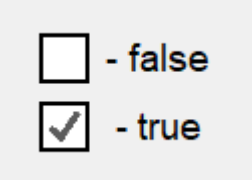


Рисунок 39 – Варианты внешнего вида виджета «Флаг»

Виджет «Флаг» предназначен для отображения значения и изменения булевой переменной (true/false). Состояние true отображается в виде галочки (см. Рисунок 38).

При нажатии на виджет значение переменной изменяется на противоположное.

В случае, если флаг необходимо использовать только для отображения информации, поставьте галочку в свойстве «Отключен»

5.10.14 Виджет «Отчет»

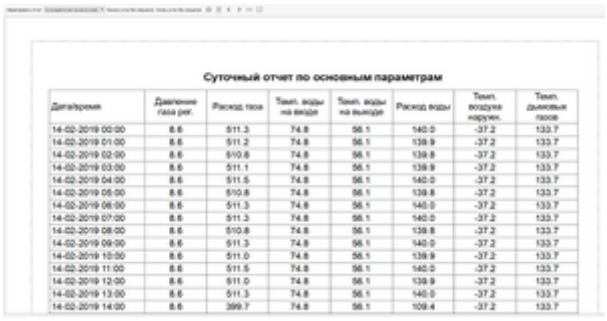


Рисунок 40 – Виджет «Отчет» в режиме отображения сформированного отчета

Виджет «Отчет» предназначен для формирования отчетов по заранее спроектированному макету.

Подробный порядок создания и редактирования отчетов приведен в разделе #Отчеты.

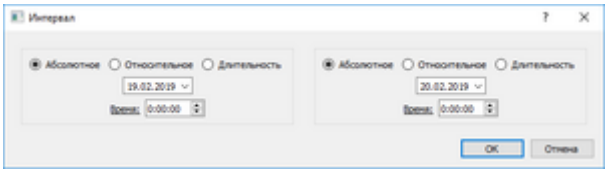


Рисунок 41 – Окно выбора интервала отчета

В верхней части виджета размещена панель управления, содержащая кнопки вызова команд:

- печать сформированного отчета на установленном в системе принтере;
- сохранение сформированного отчета в файле формата PDF;
- переход к предыдущей странице отчета;
- переход к следующей странице отчета;
- масштабирование экрана по ширине листа отчета;
- масштабирование экрана по высоте листа отчета.

При формировании отчета для изменения интервала времени, за который будет формироваться отчет, необходимо на панели управления щелкнуть кнопку выбора интервала (на ней отображается текущий выбранный интервал, например, «Начало суток без смещения – Конец суток без смещения»). Далее появится окно «Интервал», в котором можно выбрать начало и конец интервала (см. Рисунок 41).

В качестве начала и конца интервала можно выбрать как конкретную дату по календарю, так относительную, например, «Начало дня», "Конец недели". После выбора желаемого интервала необходимо нажать кнопку «OK».

Выбранный интервал отобразится на панели управления отчетом. Для формирования отчета за выбранный интервал необходимо нажать кнопку «Сформировать отчет». Через некоторое время, после загрузки данных из БД и их обработки, экран обновится и сформированный отчет за выбранный интервал будет отображен.

6 Действия

Узел типа `Действие` - законченный алгоритм, выполняющий определенный функционал.

Действия размещаются в обработчиках событий виджетов и в мониторах событий.

При размещении действий в свойствах типа `Обработка событий` виджетов, они исполняются при возникновении соответствующих событий:

- OnLeftButtonDown – нажатие левой кнопки мыши при наличии фокуса ввода на виджете.
- OnLeftButtonUp – отпускание левой кнопки мыши при наличии фокуса ввода на виджете.
- OnFocusOut – потеря фокуса ввода.
- OnButtonPress – нажатие левой кнопки мыши или кнопок `Пробел`, `Enter` на виджете типа `Кнопка`.
- OnButtonRelease – отпускание левой кнопки мыши или кнопок `Пробел`, `Enter` на виджете типа `Кнопка`.

При размещении действий в свойствах типа `Действия` мониторов событий, они исполняются при срабатывании контролируемых условий.

6.1 Действие Воспроизведение звука

Действие «Воспроизведение звукового файла» предназначено для воспроизведения указанного звукового файла через подключенный громкоговоритель или встроенную аудиосистему. Выбранный файл должен располагаться в папке проекта и быть загруженным в контроллер.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Файл	File	ResourceLink	Путь к звуковому файлу	Любой допустимый звуковой файл

6.2 Действие Смена композиции

Действие предназначено для смены текущей отображаемой композиции в указанном окне на другую. Используется для динамического изменения содержимого окон.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Номер дисплея	DisplayNo	int	Номер дисплея	Целое число ≥ 0
Окно	WindowId	NodeLink	Ссылка на окно	Узлы типа: WindowComposed
Композиция	CompositionId	NodeLink	Ссылка на композицию	Узлы типа: CompositionLayered

6.3 Действие Отображение окна

Действие предназначено для отображения окна. Окно можно отобразить по центру дисплея либо по заданным координатам.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Окно	WindowId	NodeLink	Ссылка на окно	Узлы типа: Window
Режим задания координат	CoordMode	enum	Режим задания координат	0: Не определен (ошибка) 1: Абсолютные 2: По центру дисплея
Координаты	Origin	NPoint	Координаты левого верхнего угла	Структура {x, y}
Номер дисплея	DisplayNo	int	Номер дисплея	Целое число ≥ 0
Открытие нескольких экземпляров	MultipleInstances	enum	Открытие нескольких экземпляров	0: Не определен (ошибка) 1: Запрещено на заданном дисплее 2: Запрещено 3: Разрешено

6.4 Действие Закрытие окна

Действие предназначено для закрытия окна.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Окно	WindowId	NodeLink	Ссылка на окно	Узлы типа: Window
Номер дисплея	DisplayNo	int	Номер дисплея	-1: на всех дисплеях 0: текущий активный ≥1: конкретный дисплей

Текущее	Current	bool	Закрывать текущее активное окно	true/false

6.5 Действие Переключение видимости окна

Действие предназначено для переключения видимости окна. Действие сочетает в себе действия по открытию и закрытию окна. Если окно было изначально закрыто, оно будет отображено, если окно было открыто – оно будет закрыто.

Все свойства аналогичны действию `Отображение окна`.

6.6 Действие Запись значения

Действие предназначено для записи значения в указанный узел (сигнал, источник или свойство).

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Сигнал/Источник	SignalId	NodeLink	Целевой узел для записи	Узлы типов: Source, Signal
Значение	Value	ValueLink	Значение для записи	Любое допустимое значение

6.7 Действие Создание события

Действие предназначено для создания нового события. События используются для вывода информационных сообщений или активации тревог, которые передаются в подсистемы тревог и журналирования.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Группа событий	EventGroup	NodeLink	Группа событий	Узлы типа: EventGroup
Текст сообщения активации	EventTextOn	string	Текст активации с форматтерами	Текст с поддержкой форматтеров
Текст сообщения деактивации	EventTextOff	string	Текст деактивации	Текст с поддержкой форматтеров

Тип события	EventType	enum	Тип события	0: Не определен (ошибка) 1: Информационное 2: Активация тревоги 3: Деактивация тревоги 4: Квитирование тревоги
Уровень события	EventLeve	enum	Уровень события	0: Не определен (ошибка) 1: Фатальный 2: Ошибка 3: Предупреждение 4: Замечание 5: Информация 6: Трассировка 7: Отладка
Деактивировать при сбросе	DeactivateOnFalse	bool	Деактивировать при сбросе	true/false
Группа событий для деактивации	EventGroupDeact	NodeLink	Группа для деактивации	Узлы типа: EventGroup
Звуковой файл активации	ActivationSound	ResourceLink	Звуковой файл активации	Любой допустимый звуковой файл
Звуковой файл деактивации	DeactivationSound	ResourceLink	Звуковой файл деактивации	Любой допустимый звуковой файл

6.7.1 Форматтеры для текста событий

В свойствах `EventTextOn` и `EventTextOff` действия `Создание события` поддерживаются следующие форматтеры:

- `$$nn` - имя узла;
- `$$nd` - описание узла;
- `$$ns` - шифр узла;
- `$$nv` - значение узла;
- `$$sv` - значение уставки;
- `$$nt` - время узла;
- `$$un` - имя пользователя;
- `$$ud` - полное имя пользователя;
- `$$pn` - имя родительского узла;

- `$$pd` - описание родительского узла.

6.7.2 Примеры

1. Контроль значений уставок:
    - В монитор событий добавляются несколько узлов, хранящих уставки параметров.
    - В монитор событий добавляется условие `Изменение значения`.
    - Также добавляется действие `Создание события` с текстом: `Параметр $$nd изменил значение на $$nv`.
2. Контроль границ изменения параметра:
    - В монитор событий добавляются несколько узлов, хранящих значения параметров.
    - В монитор событий добавляется условие `Сравнение значений`, определяющее условие для вхождения значения параметра в допустимые границы изменения.
    - Также добавляется действие `Создание события` с текстом: `Параметр $$nd вышел за допустимые границы!`.

6.8 Действие Команда узлу

Действие предназначено для передачи определенной команды (`Запуск`, `Останов` и др.) указанному узлу.

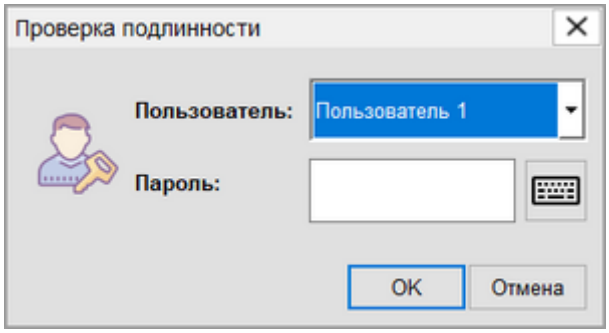
Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Узел	NodeId	NodeLink	Целевой узел	Узлы типа: Window
Команда	Command	enum	Команда узлу	0: Не определено (ошибка) 1: Запуск 2: Останов 3: Сброс

6.9 Действие Запуск процесса

Действие предназначено для запуска указанного приложения в операционной системе на целевом устройстве.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Путь до файла	FilePath	string	Путь к исполняемому файлу	Корректный путь к файлу

6.10 Действие Авторизация (проверка подлинности) пользователя



Окно "Проверка подлинности"

Действие предназначено для проверки подлинности реквизитов пользователя. При выполнении действия производится регистрация в системе указанного пользователя как текущего.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Пользователь	User	NodeLink	Пользователь для авторизации	Узлы типа: AuthUser
Запрашивать пароль	RequestPassword	bool	Запрашивать пароль	true/false

6.11 Действие Изменение курсора

Действие предназначено для изменения формы курсора мыши на указанную.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Форма курсора	CursorShape	enum	Форма курсора	0: ArrowCursor 1: CrossCursor 2: WaitCursor 3: PointingHandCursor 4: ForbiddenCursor 5: WhatsThisCursor 6: BusyCursor 7: OpenHandCursor 8: ClosedHandCursor

6.12 Действие Восстановление курсора

Действие предназначено для восстановления предыдущую форму курсора если она была изменена с помощью действия Изменение курсора.

Действие не имеет свойств.

6.13 Действие Выполнение скрипта

Действие предназначено для выполнения указанного скрипта на языке C++.

Действие выполняет предварительно созданный скрипт с передачей аргументов.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Скрипт	Script	NodeLink	Ссылка на скрипт	Узлы типа: OperationScript
Аргументы	Inputs	ValueArray	Входные аргументы	Массив значений

6.14 Действие Выполнение программы

Действие предназначено для выполнения указанной программы с передачей аргументов.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Программа	Program	NodeLink	Ссылка на программу	Узлы типов: ProgramFunctionCpp, ProgramMethodCpp
Аргументы	Arguments	ValueArray	Входные аргументы	Массив значений

6.15 Действие Отображение меню

Действие предназначено для отображения меню с настраиваемым внешним видом.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Меню	Menu	NodesContainer	Контейнер элементов меню	Узлы типа: Menu
Цвет текста	TextColor	NColor	Цвет текста	Цвет в формате RGBA
Шрифт	FontName	NFont	Шрифт	Описание шрифта
Цвет фона	BackgroundColor	NColor	Цвет фона	Цвет в формате RGBA
Цвет фона элемента	BackgroundItemColor	NColor	Цвет фона элемента	Цвет в формате RGBA
Цвет фона выделенного элемента	SelectedBackgroundItemColor	NColor	Цвет выделенного элемента	Цвет в формате RGBA
Цвет рамки	FrameColor	NColor	Цвет рамки	Цвет в формате RGBA
Ширина рамки	FrameWidth	int	Ширина рамки	Целое число ≥ 0

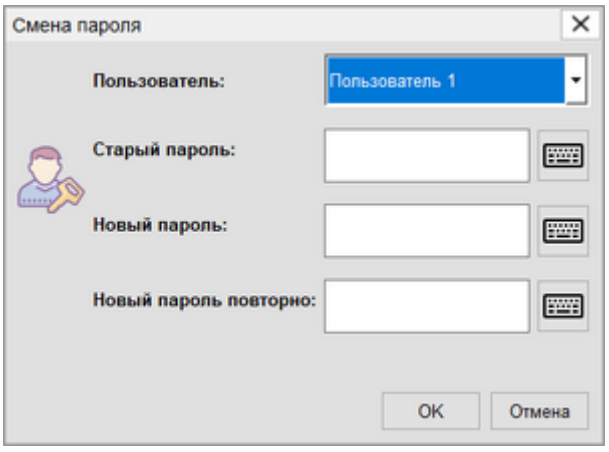
Выравнивание горизонтальное	AlignmentHoriz	enum	Выравнивание по горизонтали	0: Влево 1: По центру 2: Вправо
Выравнивание вертикальное	AlignmentVert	enum	Выравнивание по вертикали	0: Вверх 1: По центру 2: Вниз
По нажатию	ClickPos	bool	Открытие по позиции клика	true/false

6.16 Действие Вызов метода экземпляра

Действие предназначено для вызова метода экземпляра указанного объекта (класса) с возможной передачей аргументов.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Метод для вызова	CallMethod	MethodLink	Ссылка на метод экземпляра	Корректная ссылка на метод
Аргументы	Arguments	ValueArray	Входные аргументы	Массив значений

6.17 Действие Смена пароля пользователя



Окно "Смена пароля"

Действие предназначено для смены пароля пользователя. При исполнении действие отображает окно, в котором можно изменить пароль для текущего или любого пользователя.

Наименование свойства	Идентификатор	Тип	Описание	Допустимые значения
Пользователь	User	NodeLink	Пользователь для смены пароля	Узлы типа: AuthUser
Текущий	Current	bool	Смена пароля для текущего пользователя	true/false

7 Система тревог

Система тревог предназначена для оповещения пользователя об обнаруженных нарушениях.

Система содержит два типа узлов:

- Группа событий.
- Контроллер событий.

7.1 Группа событий

Группа событий – узел, предназначенный для взаимодействия с виджетом «Список событий». Группы событий позволяют группировать события по какому-то признаку, например по уровню тревоги (авария, предупреждение, квитирование и т.д.), технологическому участку и т.д.

Для каждой группы можно выбрать свой цвет шрифта и фона, которые будут использоваться для вывода событий, принадлежащих этой группе в виджете "Список событий".

В свойстве «Описание» группы событий указывается текст, который будет отображаться в фильтре списка событий.

7.2 Монитор событий

Монитор событий (ранее - Контроллер событий) – узел, реализующий логику проверки соответствия выбранных сигналов контролируемым условиям. Контроллеры событий выполняют всю работу по проверке значений контролируемых параметров и выполнению соответствующих действий.

Монитор событий проверяет значения заданных узлов на соответствие заданным условиям и при их совпадении выполняет заданные действия.

В качестве действий можно указать не только действие "Создание события", но и другие (см. раздел #Действия).



Система тревог

7.3 Порядок настройки системы

Порядок настройки системы тревог следующий:

1. Добавить в проект систему тревог. Для этого нажать ПКМ по узлу «Станция» в дереве проекта, затем выбрать «Добавить узел» - «Система» - «Система тревог».

2. Добавить узел «События» в систему тревог. Для этого нажать ПКМ по системе тревог в дереве проекта, затем выбрать «Добавить узел» - «Группа» - «События». При необходимости настроить цвет шрифта и фона, которые будут отображаться у события в окне тревог или журнале.
3. Добавить нужные группы событий в узел «События». Для этого нажать ПКМ по узлу «События» в дереве проекта, затем выбрать «Добавить узел» - «Группа событий».
4. Добавить контроллеры событий в систему тревог. Для этого нажать ПКМ по системе тревог в дереве проекта, затем выбрать «Добавить узел» - «Контроллер событий».
5. Добавить в свойство «Сигналы» контроллеров событий узлы, состояние которых необходимо контролировать.
6. Задать для этих узлов контролируемые условия в соответствующем свойстве контроллера событий.
7. Настроить действия, которые будут выполняться при выполнении контролируемого условия. Для вывода аварий, предупреждений или информационных сообщений необходимо добавить действие «Создание события».
8. Настроить действие «Создание события». В его свойстве «Группа событий» указать группу из созданных в пункте 2. Ввести тексты сообщения активации и деактивации (при необходимости), указать уровень события. Теперь, при срабатывании контролируемого условия, будет появляться окно тревог с сообщением активации.

7.4 Окно тревог

Окно тревог рпедназначено для отображения состояния активных тревог и управления ими.

Данное окно отображается при любом изменении состояния обрабатываемых тревог:

- появилось событие - отобразилось окно;
- квитировали событие - изменилось отображение события в окне;
- устранили первопричину - события пропали из окна.

Если события были квитированы и окно тревог было закрыто, то при снятии причины тревоги окно будет отображено, индицируя факт изменения состояния тревог и их отсутствие в текущий момент.

8 Система архивирования

Система архивирования предназначена для сохранения истории изменения технологических параметров и последующего получения данных за выбранный интервал времени. Архивные значения используются для построения графиков, анализа технологического процесса и формирования отчётов.

В дереве проекта система имеет наименование типа **«Система архивации»** и размещается внутри станции проекта. Сохранение выполняют подчинённые ей архиваторы. Каждый архиватор определяет место хранения, состав параметров, периодичность сбора и условия обработки данных.

Связь архиватора с параметром проекта задаётся узлом **«Коннектор БД»**. Коннектор также используется при подключении архивных данных к графику или выборке отчёта.

Для регистрации сообщений, тревог и действий пользователей применяется система журналирования. Система архивирования сохраняет историю значений параметров.

8.1 Состав системы

8.1.1 Узлы проекта

Узел	Назначение
Система архивации	Объединение архиваторов станции и представление суммарных показателей их объёма.
Архиватор	Настройка хранения, периодичности сбора, срока хранения и состава архивируемых параметров.
Коннектор БД	Связь одного архивируемого параметра с архиватором и потребителями его истории.
Сигнал или источник	Узел, текущее значение которого требуется сохранять.

Типовая структура имеет следующий вид:

```
Станция проекта
└── Система архивации
    └── Архиватор
        └── Коннекторы БД
            ├── Коннектор температуры → сигнал температуры
            ├── Коннектор давления → сигнал давления
            └── Коннектор состояния → сигнал состояния
```

В одной системе можно использовать несколько архиваторов, например для разных групп параметров, периодов опроса или мест хранения. Каждый архиватор настраивается отдельно.

8.1.2 Доступные архиваторы

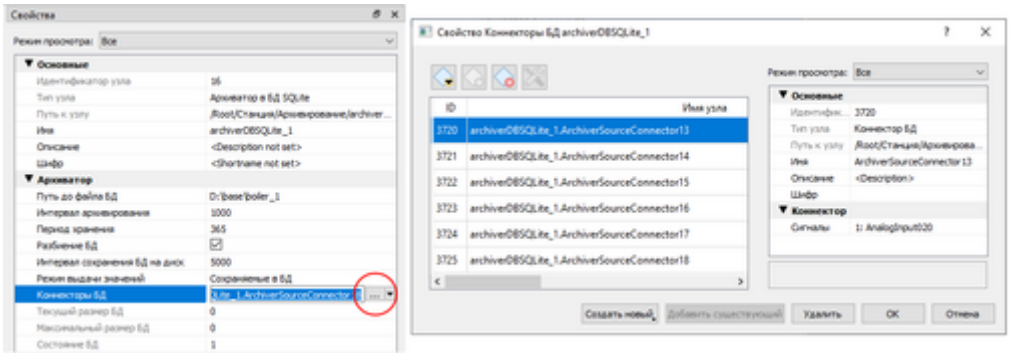
Тип узла	Основное применение	Особенности настройки
Архиватор в БД SQLite	Локальное хранение в файле или наборе суточных файлов на устройстве исполнения.	Путь к архиву, интервалы сбора и записи, срок хранения, прореживание.
Архиватор в БД PostgreSQL	Хранение на локальном или удалённом сервере PostgreSQL.	Подключение к серверу, интервальный сбор или сбор по изменению, время, разбиение, прореживание.
Архиватор в БД Firebird	Хранение в Firebird с возможностью использования срезов данных.	Путь к архиву, интервалы, срок хранения, срезы и разбиение.

Доступность конкретного архиватора определяется установленными расширениями среды разработки и среды исполнения. Для серверной БД также требуется доступный и настроенный сервер соответствующего типа.

Полный перечень и описание использования архиваторов приведено в документе Архиваторы AgavaSCADA/AgavaPLC.

8.2 Порядок настройки системы

Для сохранения истории параметров необходимо:



Настройка архиватора

1. Подготовить сигналы или источники, значения которых требуется архивировать, и проверить их обновление при выполнении проекта.
2. Добавить в проект систему архивации. Для этого необходимо открыть контекстное меню станции и выбрать **«Добавить узел» → «Система» → «Система архивации»**.
3. Добавить в систему архиватор требуемого типа.
4. Указать путь к файлу или каталогу архива либо параметры подключения к серверу БД.
5. Настроить интервал архивирования, интервал сохранения, период хранения и разбиение.
6. Открыть свойство архиватора **«Коннекторы БД»**, выбрать **«Создать новый» → «Коннектор БД»** и задать имя коннектора.
7. В свойстве коннектора **«Сигналы»** выбрать один существующий сигнал или источник.
8. Для остальных параметров создать отдельные коннекторы.
9. При необходимости настроить прореживание. Для PostgreSQL дополнительно выбрать режим архивирования и тип времени.
10. Проверить, что у участвующих узлов снят флаг **«Выключен»**, выполнить проверку проекта и устранить ошибки настройки.
11. Сохранить проект, загрузить его на целевое устройство и запустить.
12. Проверить состояние БД, появление архивных значений и их получение на графике или в отчёте.

Пути и сетевые адреса относятся к устройству, на котором выполняется среда исполнения. Путь к файлу на компьютере разработки не определяет место хранения архива на контроллере.

8.3 Общие свойства архиватора

Основные свойства расположены в группе **«Архиватор»**. Идентификаторы приведены для однозначного сопоставления со свойствами DevEnv. Набор свойств и их применение уточняются для каждого типа архиватора.

Свойство	Идентификатор	Назначение
Путь до файла БД	`FilePath`	Файл или каталог архива для файловых архиваторов. У PostgreSQL это поле отсутствует.
Интервал архивирования	`Interval`	Период получения текущих значений при интервальном сборе, мс. Начальное значение — 1000 мс.
Интервал сохранения БД на диск	`StoringInterval`	Интервал выполнения накопленных операций записи, мс. Начальное значение — 5000 мс. В PostgreSQL свойство называется «Интервал сохранения».

Период хранения	`Retention`	Продолжительность хранения, дни. Начальное значение — 1 день. Применение зависит от способа разбиения и очистки архива.
Разбиение БД	`DBSplitting`	Использование отдельных файлов или таблиц вместо единого хранилища.
Режим выдачи значений	`ValueReturnMode`	Выбор момента предоставления новых точек потребителям архива.
Коннекторы БД	`Signals`	Список связей архиватора с параметрами проекта.
Текущий размер БД	`DBCurrentSize`	Текущий учитываемый объём архива, Мб; только чтение.
Максимальный размер БД	`DBMaxSize`	Расчётная оценка объёма архива; только чтение. Не является ограничителем размера.
Состояние БД	`DBStatus`	Диагностическое состояние операций с БД; только чтение. Значение 0 соответствует отсутствию зарегистрированной ошибки.

Для интервалов и периода хранения следует задавать положительные значения. Нулевой интервал не используется как способ отключения сбора или сохранения.

8.4 Сбор и сохранение

«Интервал архивирования» определяет, как часто архиватор получает выборку параметра. **«Интервал сохранения»** определяет период обработки подготовленных данных для записи в БД. Эти значения могут различаться.

Например, при интервале архивирования 1000 мс и интервале сохранения 5000 мс значения собираются примерно раз в секунду, а операции записи выполняются группами. Частота сбора не увеличивается при уменьшении интервала сохранения.

Слишком редкий сбор может пропускать кратковременные изменения. Слишком частый сбор увеличивает объём архива и нагрузку на устройство. Выбирать интервал следует с учётом скорости изменения параметра и требуемой детализации истории.

Для SQLite и PostgreSQL архивная метка времени назначается при получении выборки архиватором. Она не сохраняет автоматически исходную метку времени устройства, например принятую вместе с телеметрией МЭК-60870. В PostgreSQL дополнительно выбирается локальное время или UTC.

8.5 Режим выдачи значений

Значение свойства	**Поведение**
Сохраняемые в БД	Новые точки предоставляются потребителям после подготовки к записи. Это начальный режим архиватора.
Сохраненные в БД	Новые точки предоставляются после обработки записи в БД.

Режим влияет, в частности, на поступление новых точек текущего графика. Он не изменяет интервал сбора и не отменяет прореживание.

В режиме **«Сохраняемые в БД»** появление точки на графике само по себе не подтверждает её запись в БД. При ненормальном завершении работы часть подготовленных данных может не сохраниться. Проверку долговременного хранения следует выполнять повторной выборкой истории, в том числе после перезапуска проекта.

8.6 Коннектор БД

Коннектор связывает архиватор с одним параметром. В его группе **«Коннектор»** предусмотрено свойство **«Сигналы»** с идентификатором `Signals`, позволяющее выбрать существующие сигналы и источники.

Несмотря на множественное число в названии свойства, в текущей версии архивируется **первый элемент списка**. Для каждого параметра необходимо создавать отдельный коннектор и включать в его список один узел.

Рекомендуется назначать коннектору имя, отражающее назначение параметра, например `TemperatureArchive` или `PressureArchive`. Наименование коннектора используется при выборе архивных данных в других узлах проекта.

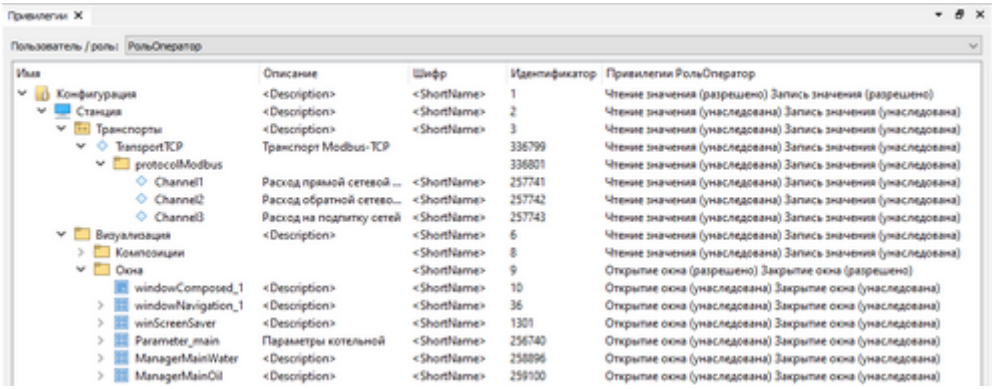
Источник коннектора должен существовать и предоставлять требуемое значение. Замена источника меняет связь коннектора; она не переносит автоматически историю прежнего параметра в историю нового.

9 Система журналирования

Система журналирования предназначена для управления регистрацией и хранением событий в журналах событий или других хранилищах. В журнал записываются все системные события и события, генерируемые с помощью системы тревог, которые соответствуют выбранному уровню сообщений.

Полное описание системы журналирования и всех узлов, которыми она управляет, приведено на странице описания.

10 Система проверки прав пользователей



Окно привилегий

Система проверки прав пользователей предназначена для разграничения доступа к значениям и окнам прикладного проекта. Она позволяет определить, какие операции доступны наблюдателю, оператору, наладчику и другим категориям пользователей.

Система реализует механизм контроля доступа к узлам проекта и их функциях на основе ролей. Узлы при выполнении действий проверяют наличие установленных привилегий у авторизованного в данный момент пользователя.

В дереве проекта используется узел **«Система проверки прав пользователей»**, размещаемый внутри станции. В системе создаются пользователи и роли. Ролям назначаются привилегии на узлы проекта, а пользователям — одна или несколько ролей.

При входе выбирается текущий пользователь. Его роли определяют результат последующих проверок доступа. Настройки относятся к выполняемому прикладному проекту; они не создают учётные записи операционной системы и не задают права редактирования проекта в DevEnv.

Полное описание системы размещено в документе "Система проверки прав пользователей AgavaSCADA/AgavaPLC".

API системы описано на странице `Класс SystemAuthentication`.

10.1 Роли и пользователи

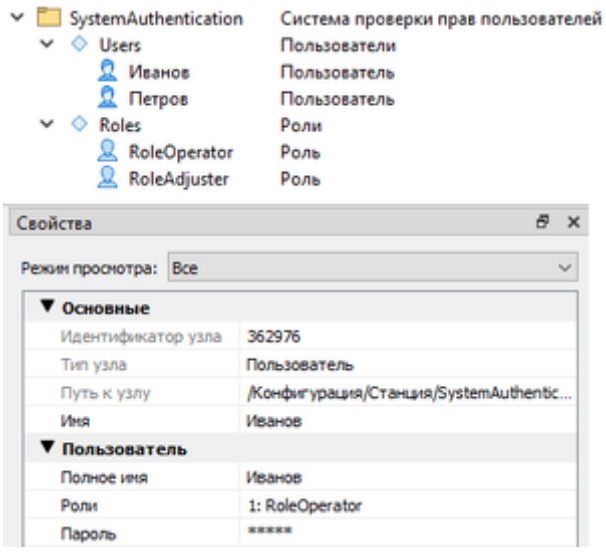
Система содержит в себе две подсистемы:

- Пользователи. Подсистема определяет пользователей, каждому из которых назначается логин/пароль и одна или несколько ролей.
- Роли. Роли хранят наборы доступных привилегий.

Отношения между пользователями, ролями и привилегиями можно описать следующим образом:

- Один пользователь может иметь несколько ролей.
- Одну роль могут иметь несколько пользователей.
- Одна роль может иметь несколько привилегий.
- Одна привилегия может принадлежать нескольким ролям.

10.2 Привилегии



Система проверки прав пользователей

Для узлов системы доступна установка следующих привилегий:

Все узлы:

- Чтение значения.
- Запись значения.

Окна и фреймы:

- Открытие окна.
- Закрытие окна.

При установке привилегий на какой-либо узел, все подчиненные узлы получают те же привилегии с флагом "унаследована".

Если привилегия не установлена, то считается что действие, регулируемое привилегией, запрещено.

Значение унаследованных привилегий можно переопределить с помощью явного задания привилегии.

10.3 Порядок настройки системы

Привилегии настраиваются в отдельном окне привилегий, доступном в меню "Вид".

Порядок настройки системы проверки прав следующий:

1. Добавить в проект систему проверки прав пользователей. Для этого нажать ПКМ по узлу «Станция» в дереве проекта, затем выбрать «Добавить узел» - «Система» - «Система проверки прав пользователей».
2. Добавить в систему проверки прав пользователей узел «Роль». Для этого нажать ПКМ по узлу «Система проверки прав пользователей» в дереве проекта, затем выбрать «Добавить узел» - «Роли».
3. Добавить в узел «Роли» необходимые роли. Для этого нажать ПКМ по узлу «Роли» в дереве проекта, затем выбрать «Добавить узел» - «Роль».
4. Для каждой роли настроить привилегии в окне привилегий. Для этого необходимо нажать «Вид» - «Открыть привилегии». В открывшемся окне в выпадающем списке «Пользователь / роль» выбрать нужную роль. Затем, нажимая ПКМ по узлам и выбирая «Добавить привилегию» настроить все необходимые ограничения для данной роли.
5. Добавить в систему проверки прав пользователей узел «Пользователи». Для этого нажать ПКМ по узлу «Система проверки прав пользователей» в дереве проекта, затем выбрать «Добавить узел» - «Пользователи».
6. Добавить в узел «Пользователи» необходимых пользователей. Для этого нажать ПКМ по узлу «Пользователи» в дереве проекта, затем выбрать «Добавить узел» - «Пользователи».
7. Для каждого пользователя указать имя и пароль, которые будут использоваться при авторизации, и выбрать одну или несколько ролей.

11 Сигналы, источники и операции

Регистры и другие узлы, добавленные в группу «Сигналы», являются внутренними для проекта и могут менять свои значения только с помощь скриптов или других узлов, с которыми они связаны.

Сигналы содержат в себе несколько подгрупп.

11.1 Сигнал

Сигнал - специальный тип узла, хранящий значение узла типа "источник". Сигналы могут использоваться для хранения значений нескольких источников, модифицировать их, обладать дополнительными полями и т. д.

Сигналы некоторых типов позволяют модифицировать исходное значение источников. Принцип обработки значения источника или сигнала с помощью операций:

Первичное значение поступает на вход операции, производится вычисление. Если есть еще операции, то вычисленное значение после первой операции передается на вход второй. И так далее по всем операциям. Если операций больше нет, вычисленное значение устанавливается в качестве значения узла.

Все операции выполняются в порядке, в котором они заданы в соответствующих свойствах.

11.1.1 Сигнал простой

Узел типа "Сигнал простой" предназначен для хранения и обработки значения источника, или другого сигнала или операции. Позволяет произвести какие-либо действия с этим значением с помощью добавления нужных операций в свойство «Операции после чтения».

11.1.2 Постоянная

Узел типа "Постоянная" предназначен для хранения данных выбранного формата и позволяет изменять их в ходе выполнения программы. Связывание с источником не предусмотрено. Изменение осуществляется с помощью специализированных функций или блока установки значения в задаче ПЛК.

11.1.3 Массив

Массив - совокупность каких-либо объектов, указанных в свойстве «Входные значения». Нумерация объектов в массиве начинается с 0. Может использоваться для взаимодействия с демультиплексором.

11.1.4 Сигнал комплексный

Сигнал используется для взаимодействия с виджетом «Цифровой индикатор». Позволяет контролировать входное значение и генерировать события переходу входного значения через уставки.

Содержит в себе свойства для настройки следующих параметров:

- ВАУ – верхняя аварийная уставка;
- ВПУ - верхняя предаварийная уставка;
- НПУ - нижняя предаварийная уставка;
- НАУ – нижняя аварийная уставка;
- ФНЧ – фильтр низких частот.

11.2 Источник

Источник - специальный тип узлов, предназначенных для передачи данных от внешних устройств.

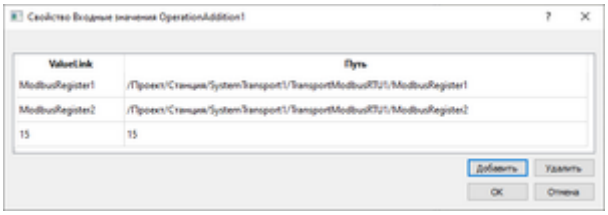
11.2.1 Регистр Modbus

Регистр Modbus используется для приема и передачи данных по протоколам Modbus-RTU, Modbus-TCP. Данный тип узов также может использоваться для организации работы некоторых алгоритмов программы (например для взаимодействия с контроллером событий).

11.2.2 Источник индексный

Узел типа "Источник индексный" используется для обработки данных, полученных по протоколу Upd, а так же в массивах для индексации значений.

11.3 Операция



Входные значения операций

Узлы типа "Операция" используются для реализации вычислительных алгоритмов в задаче ПЛК или отдельно, при их размещении в группе "Сигналы".

В качестве входных значений операций выбираются константные числа, другие узлы группы «Сигналы», регистры или блоки задачи ПЛК.

11.3.1 Операция «Сложение»

Выходным значением является сумма входных значений. В качестве входных значений можно выбирать как узлы, так и конкретные числа формата int или double.

11.3.2 Операция «Вычитание»

Выходным значением является разность первого входного значения и суммы всех последующих значений. В качестве входных значений можно выбирать как узлы, так и конкретные числа формата int или double.

11.3.3 Операция «Умножение»

Выходным значением является произведение значений всех входных значений. В качестве входных значений можно выбирать как узлы, так и конкретные числа формата int или double.

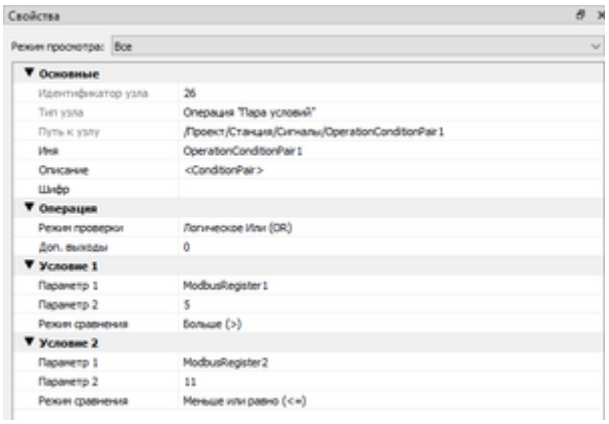
11.3.4 Операция «Деление»

Выходным значением является результат от деления первого входного значения на произведение всех последующих значений. В качестве входных значений можно выбирать как узлы, так и конкретные числа формата int или double.

11.3.5 Операция «Условие»

Узел типа "Операция условие" предназначен для сравнения значений двух узлов. Для условия задаются сравниваемые параметры и выбирается режим сравнения. В качестве входных значений можно выбирать как узлы, так и конкретные числа формата int или double.

11.3.6 Операция «Пара условий»



Операция «Пара условий»

Операция содержит в себе два условия. Для каждого условия задаются индивидуальные сравнимаемые параметры и выбирается режим сравнения.

Выходное значение операции зависит от ее режима проверки – при выбранном режиме «Логическое ИЛИ» выходное значение равно логической «1» если одно из условий равно «1», при «Логическом И»- если оба условия равны «1».

11.3.7 Операция «Логическое ИЛИ»

Если одно из входных значений становится равно логической «1» (контакты замкнуты), то выходное значение становится логической «1» (выход включен).

11.3.8 Операция «Логическое И»

Если все входные значения становятся равны логической «1», выходное значение становится логической «1» (выход включен).

11.3.9 Операция «Логическое НЕ»

Если входное значение блока становится равно логическому «0», то выходное значение становится логической «1», и наоборот.

11.3.10 Операция «Таймер»

Таймер предназначен для выполнения каких-либо действий через одинаковые промежутки времени (например запись значения в узел). Промежуток между исполнениями действий задается в свойстве «Интервал срабатывания», а сами действия в свойстве «Действия OnTimer». Для работы операции ее необходимо добавить в свойство «Узлы» задачи с периодическим или разовым режимом выполнения.

11.3.11 Операция «Мультиплексор»

Мультплексор осуществляет перевод нескольких булевых переменных в целочисленное значение по формуле: «выход = вход1 + вход2*2^1 + вход3*2^2 + ...».

Например если в качестве входных значений указаны два регистра, то выходное значение мультиплексора будет формироваться следующим образом:

Значение регистра 1	Значение регистра 2	Выходное значение
0	0	0
1	0	1
0	1	2
1	1	3

11.3.12 Операция «Демультиплексор»

Демультиплексор осуществляет перевод целочисленных значений из массива в несколько булевых переменных, при этом включается выход, номер которого соответствует числу из массива (например первый элемент массива равен 5, на пятом выходе появится значение true). Нумерация выходов начинается с нуля.

11.3.13 Операция «Генератор случайного числа»

Выходным значением является случайное число в диапазоне между указанным минимальным и максимальным значениями. Для работы операции ее необходимо добавить в свойство «Узлы» задачи с периодическим или разовым режимом выполнения.

11.3.14 Операция «Скрипт C++»

Операция «Скрипт C++» предназначена для реализации разнообразных алгоритмов на языке C++. Операция предусматривает реализацию одной или нескольких функций, выполняющих необходимые пользователю действия.

Для выполнения скрипта, можно добавить в свойство «Узлы» задачи с периодическим или разовым режимом выполнения. Другим способом вызова срабатывания скрипта является добавление его в действие «Выполнение скрипта».

Пример скрипта, выполняющего получение значения определенного узла и запись измененного значения в другой узел:

```
float val()
{
    float f = GetNodeValueAsFloat("/Конфигурация1/stationSimple1/Signals1/Random2");
    float f1 = f*100.0;
    SetNodeValueAsFloat("/Конфигурация1/stationSimple1/Signals1/Constant3", f1);
    return f*10.0;
}
```

Приведенный выше скрипт возвращает исходное значение, умноженное на 10. Возвращаемое таким образом значение записывается в выход операции для возможности считывания в других операциях, скриптах и т. д.

Скрипты могут возвращать значения следующих типов:

- void;

- float;
- double;
- int.

Подробное описание реализации языка C++ в AgavaSCADA/AgavaPLC приведено на странице "Описание языка C++ в AgavaPLC".

11.4 Объект

Узлы типа "Объект" предназначены для взаимодействия с аппаратными компонентами контроллера. Для всех объектов, являющихся субмодулями, необходимо указать слот, в котором они установлены. Доступ к значениям входов/выходов субмодуля осуществляется через его вложенные узлы.

ПЛК процессорный модуль

Предоставляет возможность регулирования подсветки экрана; включения и отключения индикаторов работы, аварии и программы.

ПЛК субмодуль DI

Позволяет получать доступ к значениям входов субмодуля DI.

ПЛК субмодуль R

Позволяет управлять значениями выходов субмодуля R.

ПЛК субмодуль SIM

Позволяет управлять значениями выходов субмодуля SIM.

ПЛК субмодуль DO6

Позволяет управлять значениями выходов субмодуля DO6.

ПЛК субмодуль AI

Позволяет получать доступ к значениям входов субмодуля AI и задавать их тип.

ПЛК субмодуль AIO

Позволяет получать доступ к значениям входов субмодуля AIO, управлять значениями выходов и задавать тип входов и выходов.

ПЛК субмодуль TMP

Позволяет получать доступ к значениям входов субмодуля TMP и задавать их тип.

12 Задачи

Группа «Задачи» предназначена для группировки узлов типа «Задача», которые используются для циклического или разового выполнения созданных пользователем алгоритмов.

12.1 Задача

12.1.1 Назначение

Узел «Задача» предназначен для разового или периодического выполнения программ на языках `AS` (C++), `ST` и `FBD/CFC`.

Задача определяет режим выполнения включённых в неё программ и период их вызова при периодическом режиме.

12.1.2 Свойства задачи

Свойство	Идентификатор	Назначение
Режим выполнения	`ExecMode`	Определяет способ выполнения программ: «Разовый» или «Периодический».
Период обработки	`ProcessPeriod`	Задаёт период выполнения в миллисекундах. Применяется в периодическом режиме.
Узлы	`ExecutableNodes`	Содержит ссылки на исполняемые узлы в порядке их вызова.

12.1.3 Режимы выполнения

В разовом режиме после запуска задачи выполняется один цикл вызова включённых в неё программ. По завершении цикла задача переходит в состояние останова. Повторное выполнение требует повторного запуска задачи.

В периодическом режиме циклы выполнения повторяются до остановки задачи. Период задаётся свойством «Период обработки» и определяет планируемый интервал между началами последовательных циклов. Время выполнения программ входит в этот интервал.

Программы одной задачи вызываются последовательно. При выборе периода необходимо учитывать суммарную длительность их выполнения: если она превышает заданный период, фактический интервал между циклами увеличивается.

12.1.4 Настройка

1. Создать узел «Задача» в группе задач проекта.
1. Добавить исполняемые узлы в свойство «Узлы» в требуемом порядке.
1. Выбрать значение «Разовый» или «Периодический» в свойстве «Режим выполнения».
1. Для периодического режима задать значение свойства «Период обработки» в миллисекундах.
1. Обеспечить запуск задачи в среде исполнения.

12.2 Обработчик события

12.2.1 Назначение

Задача «Обработчик события» предназначена для вызова пользовательских функций при возникновении исключения в приложении. Функции обработки могут быть реализованы на языках AgavaScript (AS) и Structured Text (ST).

Обработчик позволяет определить реакцию приложения на исключение, например зарегистрировать его тип или изменить диагностические переменные. При успешном выполнении хотя бы одной функции обработки исходное исключение не приводит к завершению рантайма. Работа прерванной программы или функции не возобновляется; рантайм продолжает выполнение последующих вызовов и циклов задач.

12.2.2 Настройка задачи

Для организации обработки исключений необходимо выполнить следующие действия:

1. Создать в группе задач узел типа «Обработчик события».
2. Установить для свойства «Тип» значение «Исключение».
3. В свойстве «Приложение» указать приложение, исключения которого подлежат обработке.
4. Создать в указанном приложении одну или несколько функций с сигнатурой, приведённой ниже.
5. Добавить ссылки на эти функции в свойство «Узлы» в требуемом порядке вызова.
6. Обеспечить запуск задачи. Задача, приложение и функции обработки должны быть включены.

Свойство	Идентификатор	Тип и назначение
Тип	`EventType`	Перечисление, содержащее единственное значение «Исключение».
Приложение	`Application`	Ссылка на приложение, для которого выполняется обработка исключений.
Узлы	`ExecutableNodes`	Упорядоченный список ссылок на AS- и ST-функции выбранного приложения.

В список «Узлы» допускается включать только функции выбранного приложения, удовлетворяющие требованиям к сигнатуре. Программы, методы, функции другого приложения, а также функции с дополнительными или выходными аргументами не допускаются.

12.2.3 Сигнатуры функций обработки

Функция обработки не возвращает значение и принимает ровно один входной аргумент типа `ExceptionType`. Имя функции и имя аргумента выбираются разработчиком.

12.2.3.1 AgavaScript

При передаче аргумента по значению используется следующая сигнатура:

```
void Handler(ExceptionType exceptionType)
{
   // Действия, соответствующие типу исключения.
}
```

Допускается передача аргумента по входной константной ссылке:

```
void Handler(const ExceptionType &in exceptionType)
{
   // Действия, соответствующие типу исключения.
}
```

Для сравнения с конкретным типом исключения используется выражение вида `exceptionType == ExceptionType::RTSEXCPT_DIVIDEBYZERO`. Параметры с направлениями `&out` и `&inout` не допускаются.

12.2.3.2 Structured Text

В свойстве функции «Тип результата» необходимо указать `VOID`. Единственный аргумент объявляется в секции `VAR_INPUT`:

```
FUNCTION Handler : VOID

VAR_INPUT

    exceptionType : ExceptionType;

END_VAR

(* Действия, соответствующие типу исключения. *)

END_FUNCTION
```

Для сравнения используется выражение вида `exceptionType = ExceptionType#RTSEXCPT_DIVIDEBYZERO`. Объявление аргумента в секциях `VAR_OUTPUT` и `VAR_IN_OUT` не допускается.

12.2.4 Типы и коды исключений

Коды исключений, передаваемые функции обработки, представлены значениями перечисления `ExceptionType`.

В сигнатурах функций AS и ST используется имя типа `ExceptionType`.

В таблице приведён полный перечень кодов, которые текущая реализация передаёт функции обработки, и соответствующие краткие имена. Каждое краткое имя обозначает то же значение перечисления, что и соответствующее имя `RTSEXCPT_*`. Числовые коды записаны в шестнадцатеричной форме ST.

Значение	Имя CODESYS	Краткое имя	Код
Неопределённый тип исключения	RTSEXCPT_UNKNOWN	Unknown	16#FFFFFFFF
Недостаток памяти	RTSEXCPT_OUT_OF_MEMORY	OutOfMemory	16#0000001C
Обращение по нулевому указателю	RTSEXCPT_ACCESS_VIOLATION	NullPointer	16#00000051
Переполнение стека	RTSEXCPT_STACK_OVERFLOW	StackOverflow	16#00000054
Выход за допустимые границы	RTSEXCPT_ARRAYBOUNDS	ArrayBounds	16#00000101
Деление на ноль	RTSEXCPT_DIVIDEBYZERO	DivideByZero	16#00000102
Переполнение	RTSEXCPT_OVERFLOW	Overflow	16#00000103
Недопустимая операция	RTSEXCPT_FPU_INVALID_OPERATION	InvalidOperation	16#00000154

Если тип исключения определить невозможно, передаётся `RTSEXCPT_UNKNOWN`.

Дополнительно в перечислении `ExceptionType` определены следующие константы:

Назначение	Имя CODESYS	Код
Отсутствие исключения; краткое имя — `NoException`	RTSEXCPT_NOEXCEPTION	16#00000000
Базовое значение кодов исключений изготовителя	RTSEXCPT_VENDOR_EXCEPTION_BASE	16#00002000
Базовое значение кодов пользовательских исключений	RTSEXCPT_USER_EXCEPTION_BASE	16#00010000

Указанные константы доступны в AS и ST, однако текущий механизм классификации исключений не передаёт их функции обработки.

`RTSEXCPT_UNKNOWN` имеет знаковое значение `-1`; его 32-разрядное битовое представление соответствует `16#FFFFFFFF`. Это различие необходимо учитывать при сохранении и передаче кода через целочисленные переменные.

Для исключения зависимости от числового представления рекомендуется использовать именованные константы.

12.2.5 Порядок выполнения

Обработка инициируется, если выполнение AS- или ST-кода вызвало исключение, и функция, в которой зарегистрировано исключение, принадлежит модулю выбранного приложения.

Функции обработки вызываются синхронно в потоке, выполнявшем скриптовый контекст, в порядке их расположения в свойстве `ExecutableNodes`.

Задача-обработчик не создаёт периодический поток выполнения.

Время выполнения функций обработки увеличивает длительность обработки исходного вызова; это следует учитывать при выборе выполняемых ими действий.

Вызовы выполняются только для запущенной и включённой задачи при включённом приложении. Отключённые функции пропускаются. Порядок вызова определён для функций внутри одной задачи; порядок между несколькими задачами обработки исключений не регламентируется.

12.2.6 Продолжение работы рантайма

Исходное исключение признаётся обработанным, если хотя бы одна подходящая функция обработки завершила выполнение с результатом `asEXECUTION_FINISHED`. Для функции с результатом `void` или `VOID` это означает штатное завершение её тела либо выполнение оператора возврата. Дополнительное возвращаемое значение для подтверждения обработки не требуется.

Для обработанного исключения не показывается диалог с сообщением об исключении и завершение рантайма не производится. Рантайм сохраняет возможность выполнять последующие вызовы программ и циклы задач в соответствии с их режимами выполнения. При повторном возникновении исключения процедура обработки выполняется вновь.

Условие	Поведение среды исполнения

Хотя бы одна подходящая функция обработки завершилась успешно; исключений внутри обработчиков не возникло	Исходное исключение считается обработанным. Диалог завершения рантайма не отображается; последующие вызовы и циклы задач могут выполняться.
Подходящая задача отсутствует, остановлена или отключена	Исключение считается необработанным; сохраняется штатный механизм диагностики и завершения рантайма.
Все функции обработки отключены, недоступны, имеют недопустимую сигнатуру либо не могут быть подготовлены к выполнению	Успешная обработка отсутствует; сохраняется штатный механизм диагностики и завершения рантайма.
Исключение возникло внутри функции обработки	Вложенное исключение передаётся штатному механизму диагностики необработанных исключений и может привести к завершению рантайма.

Исключение внутри функции обработки не инициирует повторный вызов обработчиков в том же потоке до завершения текущей обработки. Успешная обработка исходного исключения не подавляет диагностику вложенного исключения.

13 Устройства

Группа «Устройства» предназначена для группировки узлов типа "Устройство", выполняющие разнообразные задачи, например работа с внешними накопителями и ПЗУ.

Retain-накопитель дает возможность хранить значения регистров, постоянных и сигналов в постоянной памяти контроллера, что позволяет им сохранять свои значения после отключения питания.

Для сохранения переменных необходимо перечислить их в свойстве «Узлы» и указать путь сохранения на выбранном накопителе.

14 Отчеты



Данный узел доступен только при исполнении проекта на настольном компьютере. На панелях оператора и контроллерах МПР этот узел не доступен.

Отчет – технологический документ, содержащий информацию о ходе производственного процесса, обработанную по заложенным в отчет алгоритмам. Отчеты формируются на основе макетов (статической части отчетов) путем добавления в них свойств и значений заданных параметров технологического процесса, которые формируются (выбираются из базы данных технологических параметров и проходят соответствующую обработку) по алгоритмам, размещенным в этих шаблонах.

Данные для отчета предоставляет выборка, осуществляющая получение данных и их обработку из базы данных.

14.1 Создание нового отчета

Создание любого отчета производится в два этапа.

1. Создание необходимых узлов в конфигурации: сам отчет и необходимое количество выборок.
2. Составление макета отчета.

Создание узла типа «Отчет» в конфигурации и выборок для него производится аналогично созданию узлов других типов. К станции добавляется узел типа «Отчеты» с помощью команды контекстного меню «Добавить узел», далее к узлу типа «Отчеты» добавляется отчет и к отчету добавляется необходимое количество выборок.

Выборка – специальный узел, предназначенный для выборки данных из архива и их обработки для последующего отображения в отчете.

Перед переходом к редактированию макета желательно проделать следующие операции:

- в свойстве отчета «Путь к файлу отчета» установить имя файла, в котором будет храниться макет отчета;
- в свойстве «Источники» выборки (выборок) отчета установить перечень архивных сигналов (коннекторов), которые будут выводиться в отчете.

14.2 Редактирование макета отчета

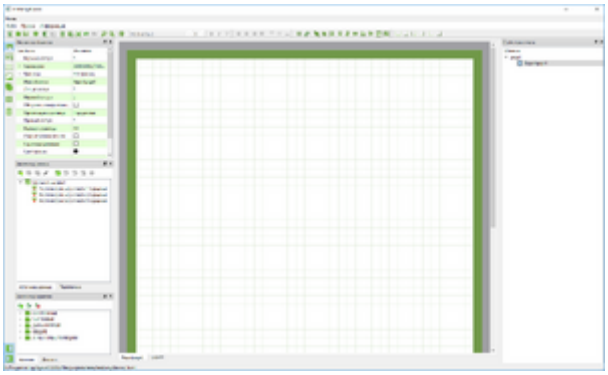


Рисунок 42 – Редактор макета отчета

Для редактирования макета отчета необходимо переключиться из редактора конфигурации в редактор макета отчета с помощью команды «Открыть редактор макета отчета», доступной из контекстного меню дерева конфигурации, отображаемого при щелчке правой кнопкой мыши на узле типа «Отчет».

В окне редактора расположены:

- панели инструментов;
- панель объектов отчета;
- окно «Инспектор объектов»;
- окно «Инспектор данных»;
- окно «Инспектор скриптов»;
- область построения макета отчета;
- окно «Структура отчета».

Редактор макета позволяет вызывать команды сочетаниями клавиш:

- Ctrl + N – новый макет;

- Ctrl + O – загрузить макет;
- Ctrl + S – сохранить макет;
- Ctrl + Shift + S – сохранить макет под новым именем;
- Ctrl + P – предварительный просмотр;
- Ctrl + Z – отмена изменений;
- Ctrl + Shift + Z – возврат отмененных изменений;
- Ctrl + C – копировать объект;
- Ctrl + X – вырезать объект;
- Ctrl + V – вставить объект;
- Ctrl + Стрелки – переместить выделенный объект;
- Shift + Стрелки – изменить размер выделенных объектов;
- Del – удалить выделенные объекты;
- Shift + левая кнопка мыши – создать область выделения.

14.2.1 Окно «Инспектор объектов»

Окно «Инспектор объектов» содержит список свойств выделенного объекта и предназначено для их редактирования.

14.2.2 Окно «Инспектор данных»

Окно «Инспектор данных» содержит список доступных для использования источников данных (выборок) и предназначено для заполнения блока «Данные» элементами соответствующего типа.

14.2.3 Окно «Инспектор скриптов»

Окно «Инспектор скриптов» содержит список доступных для использования в отчете скриптов.

14.2.4 Окно «Структура отчета»

Окно «Структура отчет» отображает структуру отчета в виде дерева связанных элементов.

14.3 Элементы отчета

Для составления макета отчета доступны следующие элементы:

- текст;
- данные;
- изображение;
- фигура;
- компоновка (Layout).

14.3.1 Текст

Элемент «Текст» служит для вывода надписей или содержимого полей источников данных.

Объект «Текст» обладает очень широкими возможностями. Он умеет отображать текст, рамку, заливку. Текст может быть отображен любым шрифтом любого размера, цвета и стиля. Большинство настроек делаются визуально с помощью панелей инструментов.

Одна из самых главных особенностей этого универсального объекта – это возможность отображения не только статичного текста, но и текста с применением вычисляемых выражений. Причем, выражения могут располагаться в объекте вперемешку с текстом. Рассмотрим простой пример – поместим в объект «Текст» следующую строку: «Привет! Сегодня $S{now()}».

Если запустить отчет на построение, мы увидим приблизительно следующее: «Привет! Сегодня 20180803». В процессе построения отчета выражение в тексте, заключенное $S{}, было вычислено и полученное значение вставлено обратно в текст. Объект «Текст» может содержать любое количество выражений, смешанных с обычным текстом. В скобки можно заключать и одиночные переменные, и выражения, например, $S{1+2*(3+4)}. В выражениях можно использовать константы, переменные ($V{}), функции, поля БД ($D{}).

Свойства элемента «Текст»:

Наименование свойства	**Описание**
alignment	Выравнивание текста по вертикали и горизонтали
angle	Поворот надписи
autoHeight	Автоматический подбор высоты
autoWidth	Автоматический подбор ширины
backgroundColor	Цвет заливки
backgroundMode	Режим заливки
backgroundOpacity	Прозрачность заливки
borders	Границы
content	Содержимое объекта «Текст»
font	Шрифт
fontColor	Цвет шрифта
foregroundOpacity	Прозрачность шрифта
geometry	Размер и расположение объекта
itemLocation	Расположение объекта (страница или бэнд)
margin	Отступы
objectName	Имя объекта
stretchToMaxHeight	Установка значения высоты самого высокого объекта на бэнде
trimValue	Удаление пробелов в начале и в конце надписи

Для редактирования свойства «Content» используется диалоговое окно «Text Item Editor», вызов которого осуществляется двойным щелчком левой кнопки мыши на элементе «Текст».

Свойство «Content» может содержать:

- текст;
- значения переменных;
- значения полей из набора данных;
- значения, формируемые посредством исполнения скрипта.

Для вывода значений переменных используется синтаксис «$V{имяПеременной}» (без кавычек), для вывода значения поля данных – «$D{имяНабораДанных.ИмяПеременной}», для вывода результата исполнения скрипта – «$S{телоСкрипта}».

14.3.2 Изображение

Элемент для вывода изображений:

Наименование свойства	Описание
autoSize	Подгонять размер под размер изображения
borders	Границы
content	
datasource	Источник данных
field	Поле данных
geometry	Размер и расположение
image	Изображение
itemLocation	Расположение объекта (страница или бэнд)
objectName	Имя объекта
stretchToMaxHeight	Устанавливать значение высоты самого высокого объекта на бэнде

14.3.3 Фигура

Элемент для вывода фигур:

Наименование свойства	Описание
borders	Границы
geometry	Размер и расположение
itemLocation	Расположение объекта (страница или бэнд)
lineWidth	Толщина линии
objectName	Имя объекта
opacity	Прозрачность
penStyle	Стиль линии
shape	Фигура
shapeBrush	Стиль заливки
shapeBrushColor	Цвет заливки
shapeColor	Цвет фигуры
stretchToMaxHeight	Устанавливать значение высоты самого высокого объекта на бэнде

14.3.4 Горизонтальная компоновка (Layout)

Элемент, позволяющий объединять несколько элементов в группу:

Наименование свойства	Описание
borders	Границы
geometry	Размер и расположение
itemLocation	Расположение объекта (страница или бэнд)
objectName	Имя объекта
stretchToMaxHeight	Устанавливать значение высоты самого высокого объекта на бэнде

14.4 Разделы отчета (бэнды)

Бэнд предназначен для размещения других элементов отчета. Бэнд может быть нескольких типов:

- Report Header – заголовок отчета;
- Report Footer – завершение отчета;
- Page Header – верхний колонтитул страницы отчета;
- Page Footer – нижний колонтитул страницы отчета;
- Data – данные отчета;
- SubDetail – подчиненные данные отчета;
- SubDetailHeader – заголовок подчиненных данных;
- SubDetailFooter – завершение подчиненных данных;
- GroupHeader – заголовок группы;
- GroupFooter – завершение группы.

Бэнды применяются для логической группировки объектов. К примеру, размещение объекта на элементе «Данные» типа «Page Header», приводит к отображению этого объекта в верхней части каждой страницы готового отчета. Аналогичным образом «Page Footer» выводится внизу каждой страницы со всеми лежащими на нем объектами.

Общие для всех элементов «Данные» свойства:

Наименование свойства	**Описание**
autoHeight	Автоматический подбор высоты
backgroundColor	Цвет заливки бэнда
borders	Границы
geometry	Размер
keepBottomSpace	Сохранение отступа от нижней границы бэнда
objectName	Имя объекта
printIfEmpty	Отображение пустого бэнда
splittable	Разделение бэнда, если он не влезает на страницу

Report Footer:

Наименование свойства	**Описание**
maxScalePercent	Максимальный процент, на который можно уменьшить бенд, если он не влезает на страницу. Если бэнд даже после сжатия не влезает на страницу, он будет перенесен полностью или частично – в зависимости от настроек бэнда

Data:

Наименование свойства	**Описание**
datasource	Источник данных. Data-бэнд будет сформирован для каждой строки в источнике данных

keepFooterTogether	Если Report Footer не влезает на страницу, то он будет перенесен на следующую страницу совместно с последним экземпляром Data-бэнда
sliceLastRow	Указывает генератору отчетов на то, можно ли разрезать последний экземпляр Data-бэнда или его нужно перенести целиком

Бэнды данного типа предназначены для вывода на печать данных из наборов. Набор данных - это заранее неизвестное количество строк (записей), каждая из которых содержит определенное количество колонок (полей).

SubDetail:

Наименование свойства	**Описание**
datasource	Источник данных. SubDetail-бэнд будет сформирован для каждой строки в источнике данных

SubDetailHeader, SubDetailFooter:

Наименование свойства	**Описание**
printAlways	Печатать, даже если SubDetail пустой

GroupHeader:

Наименование свойства	**Описание**
groupFieldName	Поле, по которому осуществляется группировка. Экземпляр GroupHeader будет формироваться при смене значения в этом поле

14.5 Функции

14.5.1 Агрегатные функции

Агрегатные функции предназначены для обработки значений, содержащихся в резделах Data и SubDetail.

14.5.1.1 AVG - вычисление среднего значения

Пример использования:

```
$S{AVG("TextItem4","DataBand1");}
```

14.5.1.2 SUM - вычисление суммы значений

Пример использования:

```
$S{SUM("TextItem8","DataBand1");}
```

14.5.1.3 MIN

14.5.1.4 MAX

14.5.1.5 COUNT

14.5.2 Числовые функции

14.5.2.1 numberFormat - форматирование числа

Синтаксис:

```
numberFormat(значение, 'формат', точность)
```

Форматы:

- `'f'` — фиксированная запись (например, `"123.45"`);
- `'g'` — общий формат (автоматически выбирает между `'f'` и `'e'`);
- `'e'` — экспоненциальная запись (например, `"1.2345e+02"`);

Пример использования:

```
$S{numberFormat($D{fetcharchiverdata1.ArchiversSourceConnector0 (среднее)}, 'f', 3)}       ->
123.495
```