Объектная модель AgavaSCADA/AgavaPLC

Редакция от 11.11.2025 · docs.kb-agava.ru

Содержание

1 Введение 2
2 Узлы 2
  2.1 Приложение 2
    2.1.1 Типы вложенных узлов 2
    2.1.2 Класс 2
    2.1.3 Структура 2
    2.1.4 Функция 3
    2.1.5 Программа 3
    2.1.6 Секция 3
3 Наследование 3
  3.1 Иерархия классов/объектов/узлов 3
  3.2 Взаимодействие с узлами проекта. 3
  3.3 Описание свойств, полей и методов базовых классов 3
  3.4 Использование в дереве проекта узлов, унаследованных от базовых классов 4
4 Связывание свойств экземпляров объектов в дереве проекта 4
  4.1 Свойства элементарных типов 4
    4.1.1 Примеры 4
  4.2 Свойства типов объектной модели 4
    4.2.1 Примеры 5
5 Разработка приложения 5

1 Введение

Данный документ описывает объектную модель AgavaSCADA/AgavaPLC версии 1.5 и старше (новее).

2 Узлы

2.1 Приложение

Узел для организации алгоритмов.

Позволяет хранить внутри себя алгоритмы и структуры, а также узлы (окна, формы, регистры и т. д.).

2.1.1 Типы вложенных узлов

- Класс (C++).
- Структура *(C++).*
- Функция (C++).
- Программа (C++).
- Секция (C++).

2.1.2 Класс

Класс – это основополагающая сущность объектно-ориентированного программирования.

Узлы типа «Класс» позволяют реализовывать создание объектов определённого типа, описывая их структуру (набор полей и их начальное состояние) и определять алгоритмы (функции или методы) для работы с этими объектами.

Класс можно использовать для объявления экземпляров в функции, программе или методе другого класса. Возможно объявление экземпляров класса в дереве проекта и доступ к его полям, методам и свойствам из программ и функций.

Аналог класса в языках МЭК 61131 – функциональный блок.

2.1.2.1 Вложенные узлы

- Метод. Функция, принадлежащая классу и имеющая доступ к его полям и методам.
- Свойство. Специальный метод, предоставляющий доступ к полям через функции-сеттеры/геттеры.
- Действие. Специальный метод, доступный для использования в проекте как другие узлы типа "Действие".

2.1.3 Структура

Структура – композитный тип данных, инкапсулирующий без сокрытия набор значений различных типов.

Структуру можно использовать для объявления в функции, процедуре и классе. Возможно объявление экземпляров структуры в дереве проекта и доступ к ее полям из программ и функций..

2.1.4 Функция

Функция – фрагмент программного алгоритма, к которому можно обратиться из другого алгоритма - программы, метода или функции.

Функцию можно использовать для вызова в другой функции, процедуре или методах класса.

2.1.5 Программа

Программа – новый элемент объектной модели, отсутствующий в языке C++. Наиболее близок к функции, имеет возможность объявления внутренних объектов (аналог блока VAR), которые сохраняют свое значение при циклическом выполнении алгоритма. Позволяет задавать входные переменные (аналог блока VAR_IN). В программе возможно использование всех декларативных типов узлов (функций, структур и классов). Аналог в Codesys – программа. Возможно добавление дополнительных вложенных функций, не доступных вне процедуры.

2.1.6 Секция

Секция позволяет разместить в приложении часть программного кода без создания классов и функций в приложении. Секция компилируется вместе с приложением и все определенные в ней элементы будут доступны в других узлах приложения.

3 Наследование

3.1 Иерархия классов/объектов/узлов

- BasicNode – базовый класс узла.
- BasicSource – базовый класс узла-источника.
- BasicSignal – базовый класс узла-сигнала.
- BasicWindow – базовый класс узла-окна.
- BasicWidget – базовый класс узла-виджета.

и т.д.

3.2 Взаимодействие с узлами проекта.

В функции, программы и методы можно передавать ссылки и указатели на узлы проекта и вызывать методы, определенные в узлах.

3.3 Описание свойств, полей и методов базовых классов

Описание свойств, полей и методов базовых классов AgavaSCADA/AgavaPLC приведено в описании базовых классов AgavaSCADA/AgavaPLC.

3.4 Использование в дереве проекта узлов, унаследованных от базовых классов

При создании в программе собственного класса, основанного на переопределении базового, появляется возможность создания и использования экземпляров этого класса в дереве проекта. При этом логика работы экземпляра такого класса определяется реализацией его собственных методов, а не методов базового класса.

4 Связывание свойств экземпляров объектов в дереве проекта

При связывании свойств экземпляров объектов в дереве проекта с другими узлами действует описанная ниже следующая логика.

4.1 Свойства элементарных типов

Свойства элементарных типов (bool, int, float, string и т. д.), у экземпляров объектов могут иметь значения этих типов.

При установке значения свойства элементарного типа как значения элементарного типа это значение однократно устанавливается в свойстве при инициализации объекта. Таким образом в свойстве можно установить постоянное значение.

При установке значения свойства элементарного типа как ссылки на узел проекта одного из типов объектной модели AgavaSCADA/AgavaPLC производится связывание свойства с узлом. В этом случае при обновлении значения в узле производится автоматический вызов сеттера свойства с обновленным значением узла.

4.1.1 Примеры

4.1.1.1 Свойство типа float, связанное с узлом в дереве проекта

```
get
{
    return m_fValue;
}
```

```
set
{
    m_fValue = value;
    PropertyValueUpdated("Value");  // Value - имя свойства
    Repaint();
}
```

При установке значения такого свойства как ссылки на узел соответствующего типа в дереве проекта, сеттер свойства будет вызываться при каждом изменении значения узла.

4.2 Свойства типов объектной модели

При связывании свойства типа одного из классов объектной модели производится однократная установка указателя на связываемый узел в свойстве при инициализации объекта.

4.2.1 Примеры

4.2.1.1 Свойство типа BasicSource_t@, связанное с узлом в дереве проекта

```
get
{
    return m_nSource;  //декларация поля: BasicSource@ m_nSource;
}
```

```
set
{
    // Свойство нужно для хранения связи с источником, из которого можно получить имя,
описание и пр.
    @m_nSource = BasicSource(value);  //Сохраняем умный указатель на переданный узел в поле.
}
```

Таким образом в свойстве запомнен умный указатель на один из узлов проекта, к которому можно обратиться в любом месте класса, которому принадлежит приведенное выше свойство.

5 Разработка приложения

AgavaSCADA/AgavaPLC предоставляет пользователю возможность разработки приложений с использованием объектно-ориентированного подхода несколькими способами:

1. Написание программ, классов, функциональных блоков с помощью текстовых языков (C++, ST). Все создаваемые алгоритмы размещаются в секциях, программах и функциях. Если требуется передача данных в виджеты или другие части проекта, используются соответствующие методы и функции.
2. Создание классов и функциональных блоков со свойствами и методами в дереве проекта. При таком способе появляется возможность визуального связывания виджетов и других визуальных частей проекта с размещенными в дереве проекта экземплярами созданных ФБ или классов.