ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 2](#_Toc105616957)

[1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 3](#_Toc105616958)

[1.1 Автоматизированные системы управления технологическими процессами 3](#_Toc105616959)

[1.1.1 Принцип построения системы АСУ ТП 4](#_Toc105616960)

[1.1.2 Аппаратные и программные составляющие АСУ ТП 5](#_Toc105616961)

[1.2 Программируемые логические контроллеры 6](#_Toc105616962)

[1.2.1 Типы ПЛК 7](#_Toc105616963)

[1.2.2 Виды языков программирования для ПЛК 9](#_Toc105616964)

[2 ПРАКТИЧЕСКАЯ ЧАСТЬ 11](#_Toc105616965)

[2.1 Отчёт №1 «Основы работы с графикой в CODESYS 3.5». 11](#_Toc105616966)

[2.1.1 Цель работы 11](#_Toc105616967)

[2.1.2. Создание стандартного проекта в Codesys 3.5 11](#_Toc105616968)

[2.1.3. Выполнение практической работы 11](#_Toc105616969)

[2.1.4. Запуск программы 15](#_Toc105616970)

[2.1.5 Результат работы программы 17](#_Toc105616971)

[2.2 Отчёт №2 «Типичные параметры графических примитивов». 19](#_Toc105616972)

[2.2.1 Цель работы 19](#_Toc105616973)

[2.2.2 Выполнение практической работы. 19](#_Toc105616974)

[2.2.3 Результат работы программы 24](#_Toc105616975)

[2.3 Отчёт №3 «Работа с Фреймами». 26](#_Toc105616976)

[2.3.1 Цель работы 26](#_Toc105616977)

[2.3.2 Выполнение практической работы 26](#_Toc105616978)

[2.3.3 Результат работы программы 27](#_Toc105616979)

[2.4 Отчёт №4 «Комбинированное окно» 29](#_Toc105616980)

[2.4.1 Цель работы 29](#_Toc105616981)

[2.4.2 Выполнение практической работы 29](#_Toc105616982)

[2.4.3 Результат работы программы 34](#_Toc105616983)

[ЗАКЛЮЧЕНИЕ 35](#_Toc105616984)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 36](#_Toc105616985)

ВВЕДЕНИЕ

Автоматизация — одно из направлений научно-технического прогресса, применение саморегулирующих технических средств, экономико-математических методов и систем управления, освобождающих человека от участия в процессах получения, преобразования, передачи и использования энергии, материалов или информации, существенно уменьшающих степень этого участия или трудоёмкость выполняемых операций. Требует дополнительного применения датчиков (сенсоров), устройств ввода, управляющих устройств (контроллеров), исполнительных устройств, устройств вывода, использующих электронную технику и методы вычислений, иногда копирующие нервные и мыслительные функции человека. Наряду с термином автоматический, используется понятие автоматизированный, подчеркивающий относительно большую степень участия человека в процессе.

Цель автоматизации — повышение производительности труда, улучшение качества продукции, оптимизация управления, устранение человека от производств, опасных для здоровья, повышение надежности и точности производства, увеличение конвертируемости и уменьшение времени обработки данных.

1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ
   1. ****Автоматизированные системы управления технологическими процессами****

**АСУ ТП –** это комплекс программных и технических средств, предназначенных для создания систем автоматизации управления технологическим оборудованием и производственными процессами на предприятиях (автоматизация производства). АСУ ТП – комплексное решение, обеспечивающее автоматизацию основных технологических операций на производстве в целом или каком-то его участке, выпускающем относительно завершенный продукт (промышленная автоматизация). АСУ ТП может состоять из отдельных систем автоматического управления (САУ) и комплексных устройств, объединенных единым решением для автоматизации технологических процессов с целью обеспечения максимальной эффективности решения производственных задач.

**Система АСУ ТП позволяет:**

* сделать работу оборудования более эффективной;
* исключить простои и сбои в функционировании оборудования;
* добиться более удобного управления необходимыми процессами, предусмотренными определенной технологией;
* осуществлять контроль и мониторить параметры процесса;
* устранить ошибки, допущенные из-за человеческого фактора, когда осуществляется управление.

Система АСУ ТП включает не только технические и программные средства. В её состав входят разные виды обеспечения, в частности, метрологическое, эргономическое, информационное, организационное. При автоматизации управлении значительно облегчается работа человека, когда требуется контролировать, стабилизировать, управлять процессами производства. Но данная система не исключают человеческий фактор.

* + 1. Принцип построения системы АСУ ТП

[АСУ ТП](https://www.tadviser.ru/index.php/%D0%90%D0%A1%D0%A3_%D0%A2%D0%9F) система обычно строится по трехуровневому принципу:

* **Нижний уровень**. Уровень оборудования (входов/выходов – Input/Output level). Это уровень датчиков, измерительных устройств, контролирующих управляемые параметры, а также исполнительных устройств, воздействующих на эти параметры процесса, для приведения их в соответствие с заданием. На этом уровне осуществляется согласование сигналов датчиков с входами устройства управления, а вырабатываемых команд с исполнительными устройствами.
* **Средний уровень.** Уровень управления оборудованием – Control level. Это уровень контроллеров (ПЛК – PLC – Programable Logic Controller). ПЛК получает информацию с контрольно-измерительного оборудования и датчиков о состоянии технологического процесса и выдает команды управления, в соответствии с заранее запрограммированным алгоритмом управления, на исполнительные механизмы.
* **Верхний уровень**. Уровень промышленного сервера, сетевого оборудования, уровень операторских и диспетчерских станций. На этом уровне идет контроль хода производства: обеспечивается связь с нижними уровнями, откуда осуществляется сбор данных, визуализациия и диспетчеризациия (мониторинг) хода технологического процесса.  Это уровень HMI, SCADA. На этом уровне задействован человек, т.е. оператор (диспетчер). Он осуществляет локальный контроль технологического оборудования через так называемый человеко-машинный интерфейс (HMI - Human Machine Interface). К нему относятся: мониторы, графические панели, которые устанавливаются локально на пультах управления и шкафах автоматики. Для осуществления контроля за распределенной системой машин, механизмов и агрегатов применяется SCADA (Supervisory Control And Data Acqusition - диспетчерское управление и сбор данных) система. Эта система представляет собой порграммное обеспечение, которое настраивается и устанавливается на диспетчерских компьютерах. Она обеспечивает сбор, архивацию, визуализацию, важнейших данных от ПЛК. При получении данных система самостоятельно сравнивает их с заданными значениями управляемых параметров (уставками) и при отклонении от задания  уведомляет оператора с помощью тревог(Alarms), позволяя ему предпринять необходимые действия. При этом система записывает все происходящее, включая действия оператора, обеспечивая контроль действий оператора в случае аварии или другой нештатной ситуации. Таким образом, обеспечивается персональная ответственность управляющего оператора.
  + 1. Аппаратные и программные составляющие АСУ ТП

Аппаратное обеспечение АСУ ТП — это комплекс технических средств, обеспечивающих выполнение всех функций АСУ ТП, а также обеспечивающих взаимодействие персонала с техническими средствами системы и с технологическим процессом.

**Аппаратные средства автоматизированных систем включают в себя:**

* контроллеры;
* операторские станции, сервера, сети;
* модули цифрового интерфейса;
* систему управления диспетчером;
* измерительные преобразователи;
* счетчики и сигнализаторы;
* исполнительные механизмы.

Программное обеспечение АСУ ТП— совокупность программ, обеспечивающих функционирование всех цифровых вычислительных средств АСУ ТП (контроллеры, серверы, рабочие и инженерные станции, программаторы, панели оператора), а также решающих все функциональные задачи на этапах разработки, наладки, тестирования и эксплуатации системы.

**Программное обеспечение АСУ ТП делится на две категории:**

* Общее программное обеспечение, включающее операционные системы, SCADA-системы, пакеты программ для программирования контроллеров, компиляторы, редакторы и т.п. Общее программное обеспечение АСУ ТП не привязано к конкретному объекту автоматизации.
* Специальное программное обеспечение — это программы, разработанные для конкретной АСУ ТП. К этой категории относятся программы для контроллеров, реализующие определенные функциональные задачи обработки информации и управления; программы, сгенерированные в среде SCADA-системы для визуализации, архивирования данных конкретного технологического процесса.
  1. Программируемые логические контроллеры

ПЛК – это унифицированная цифровая управляющая электронная система, специально разработанная для использования в производственных условиях. ПЛК постоянно контролирует состояние устройств ввода и принимает решения на основе пользовательской программы для управления состоянием выходных устройств.

ПЛК широко применяются в сфере промышленной автоматизации разнообразных технологических процессов на больших и малых предприятиях. Их применение значительно упрощает создание и эксплуатацию как сложных автоматизированных систем, так и отдельных устройств. ПЛК позволяет сократить этап разработки, упрощает процесс монтажа и отладки за счет стандартизации отдельных аппаратных и программных компонентов, а также обеспечивает повышенную надежность в процессе эксплуатации, удобный ремонт и модернизацию при необходимости.

ПЛК имеет три основные секции:

* Входная секция ПЛК обеспечивает ввод в центральную секцию состояния переключателей, датчиков и смарт-устройств.
* Центральная секция содержит центральный процессор (ЦП), память и систему коммуникаций. Она выполняет обработку данных, принимаемых от входной секции данных, и передает результаты обработки в выходную секцию.
* Выходная секция ПЛК управляет внешними исполнительными устройствами.

Система программирования является одной из примечательных и полезных особенностей ПЛК, она обеспечивает упрощенный подход к разработке управляющих программ для специалистов различного профиля.

В качестве основного режима работы ПЛК выступает его длительное автономное использование, зачастую в неблагоприятных условиях окружающей среды, без серьёзного обслуживания и практически без вмешательства человека.

* + 1. Типы ПЛК

Классифицировать программируемые логические контроллеры (ПЛК) можно по следующим основным критериям:

**Количество каналов ввода/вывода.** По количеству каналов ввода/вывода ПЛК подразделяются на следующие классы:

* нано контроллеры (часто с встроенными функциями), имеющие до 16 входов/выходов;
* малые контроллеры, рассчитанные на 16-100 входов/выходов;
* средние контроллеры, рассчитанные примерно на 100-300 входов/выходов;
* большие контроллеры, рассчитанные примерно на 300-2000 входов/выходов;
* сверхбольшие контроллеры, имеющие примерно от 2000 и более входов/выходов.

**Расположение модулей ввода/вывода.** По расположению модулей ввода/вывода ПЛК различаются на:

* моноблочные — в которых устройство ввода/вывода не может быть отделено от контроллера или заменено на другое.
* модульные — состоящие из общей корзины (шасси), в которой располагаются модуль центрального процессора и сменные модули ввода/вывода.
* распределенные (с удаленными модулями ввода/вывода).

**Функциональное назначение.** По функциональному назначению ПЛК делятся на следующие типы:

* универсальные общепромышленные ПЛК;
* коммуникационные ПЛК;
* ПИД-контроллеры;
* специализированные ПЛК.

**Способ крепления.** По способу крепления и конструктивному исполнению промышленные контроллеры подразделяются на:

* Панельные – монтаж осуществляется на лицевой панели шкафа(щита);
* DIN-реечные — монтаж осуществляется внутри шкафа(щита) на DIN-рейку;
* Стоечные – монтаж осуществляется в стойке;
* Бескорпусные – применяется производителями оборудования ОЕМ (Original Equipment Manufacturer) в специализированных конструктивных исполнениях.

**Используемые языки программирования.** ПЛК могут программироваться на следующих языках:

* на языках МЭК61131-3 (IEC61131-3);
* на классических алгоритмических языках (C, С#, Visual Basic).
  + 1. Виды языков программирования для ПЛК

Для программирования ПЛК используются стандартизированные языки МЭК (Международная Электротехническая Комиссия) стандарта IEC61131-3. IEC61131-3 – раздел международного стандарта IEC61131, описывающий языки программирования для ПЛК.

Структурно в IEC61131-3 среда исполнения представляет собой набор ресурсов. Ресурс предоставляет возможность исполнять задачи. Задачи представляют собой набор программ. Задачи могут вызываться циклически, по событию, с максимальной частотой.

Графические языки программирования:

* LD (Ladder) – это среда разработки, которая представляет собой подобие релейной схемы. Является самым распространённым языком для программирования ПЛК.
* FBD (Диаграмма Функциональных Блоков) определяет собой некую множественность функциональных блоков, которые имеют соединения между cобой (вход и выход). Данные связи являются переменными и выполняют пересылку между блоками. Каждый блок в отдельности может представлять определенную операцию (триггер, логическое “или” и т.д.). Переменные задаются с помощью определенных блоков, а цепи выхода могут иметь связи с конкретными выходами контроллера или связи с глобальными переменными.
* SFC (Sequential Function Chart). Принцип его построения близок к образу конечного автомата, данное условие относит его к самым мощным языкам программирования. Технологические процессы, в данном языке, построены по типу определенных шагов. Структура шагов состоит из вертикали, которая идет сверху вниз. Каждый шаг – это конкретные операции.
* CFC (Continuous Flow Chart) – относится к языкам высокого уровня. Процесс проектирования состоит из использования готовых блоков и размещения их на экране. Далее происходит их настройка и размещения соединений между ними. Каждый блок – это управление определенным технологическим процессом. Здесь идет основной уклон на технологический процесс.

Текстовые языки программирования:

* ST (Структурированный Текст) – относится к языкам высокого уровня и имеет много сходного с Pascal и Basic. ST позволяет интерпретировать более шестнадцати типов данных и имеет возможность работать с логическими операциями, циклическими вычислениями и т.д.
* IL (Список Команд) – язык подобен Ассемблеру, обычно используется для кодировки блоков по отдельности. Данные блоки имеют большую скорость работы и низкую требовательность к ресурсам.

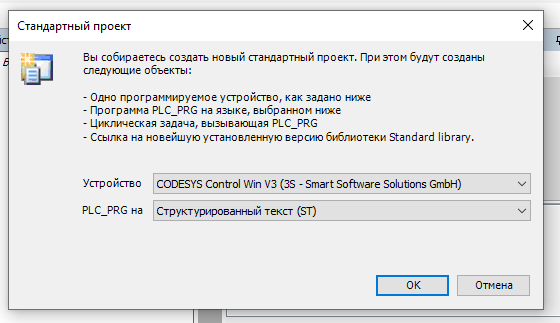
1. ПРАКТИЧЕСКАЯ ЧАСТЬ
   1. Отчёт №1 «Основы работы с графикой в CODESYS 3.5».

### **2.1.1 Цель работы**

Изучение простейших графических примитивов в графическом редакторе Codesys 3.5 и их основных параметров, ознакомиться с графическим примитивом Кнопка.

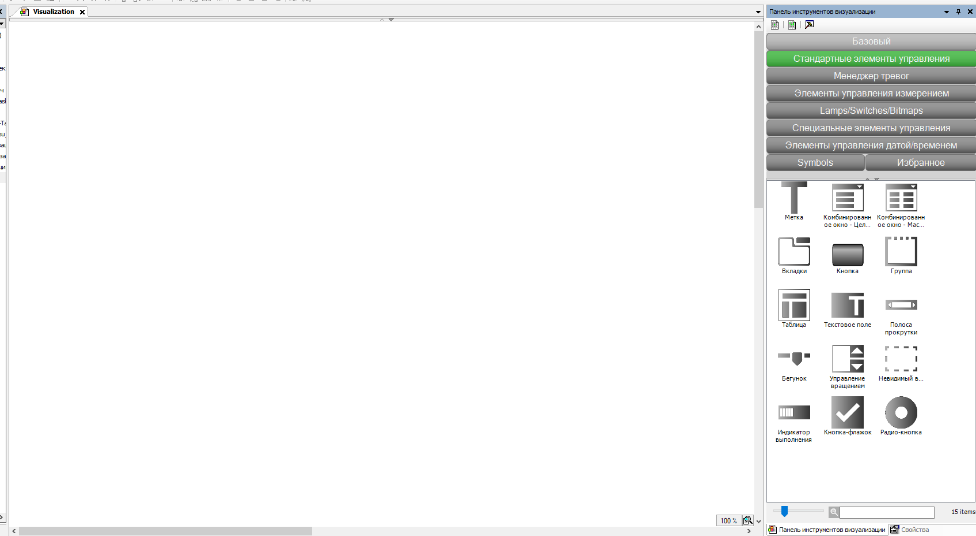
### **2.1.2. Создание стандартного проекта в Codesys 3.5**

Выбираем устройство, как на Рис. 1. Язык программирования ST.

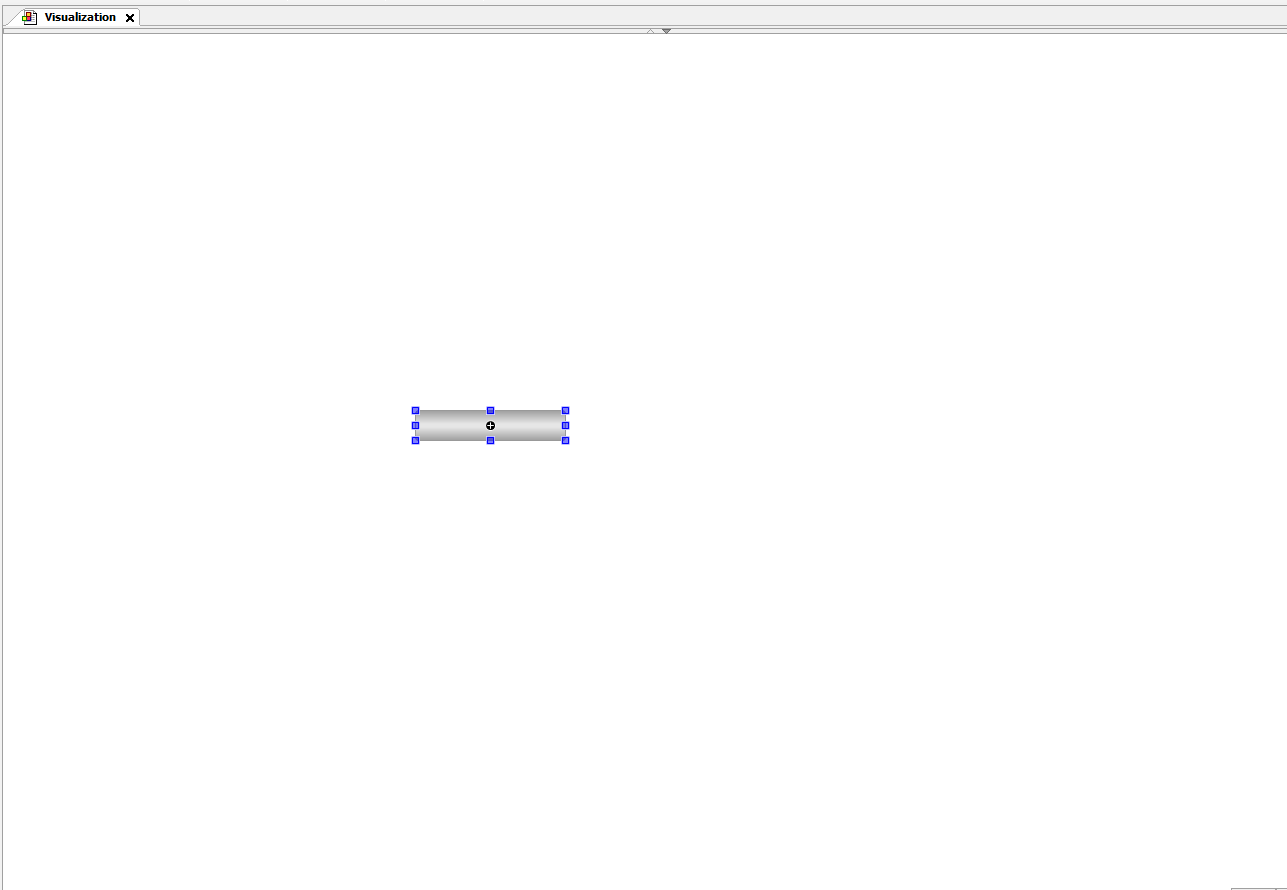
  
Рисунок 2.1 — **Первичные параметры стандартного проекта**

### **2.1.3. Выполнение практической работы**

После добавления визуализации в проекте появляется пустая вкладка «Visualisation» (Рис. 2).

  
Рисунок 2.2— **Вид пустой вкладки Визуализация**

Панели инструментов визуализации – выберем примитив Кнопка и перетащим Кнопку на рабочую область (Рис. 3).

  
Рисунок 2.3 - **Добавление примитива Кнопка**

Зададим Имя Кнопки (Рис. 4).

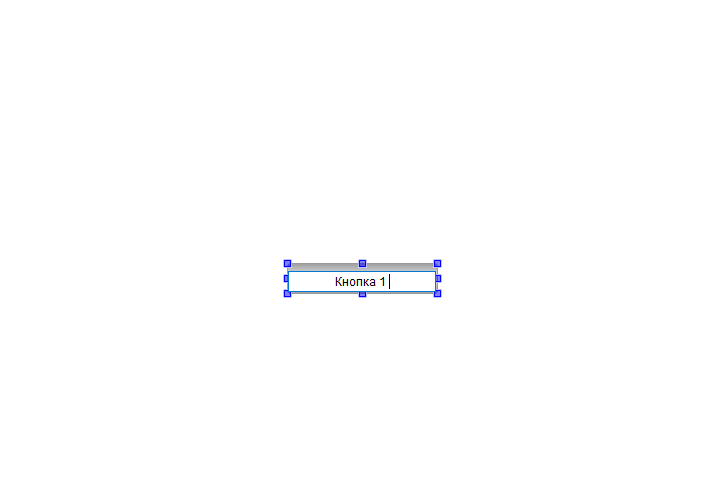
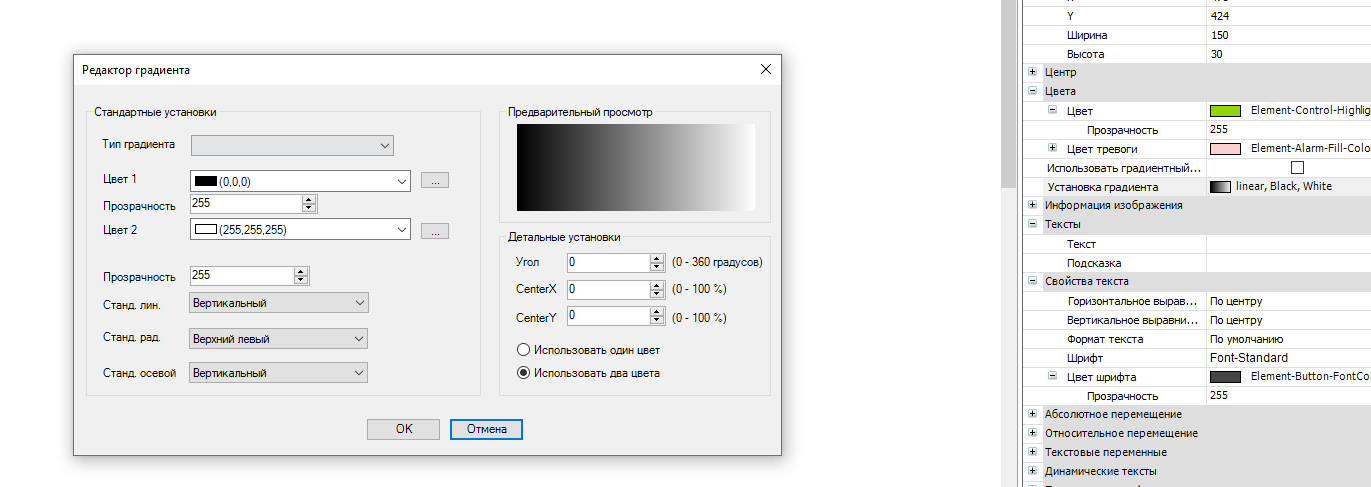


Рисунок 2.4 - **Изменение имени Кнопки**

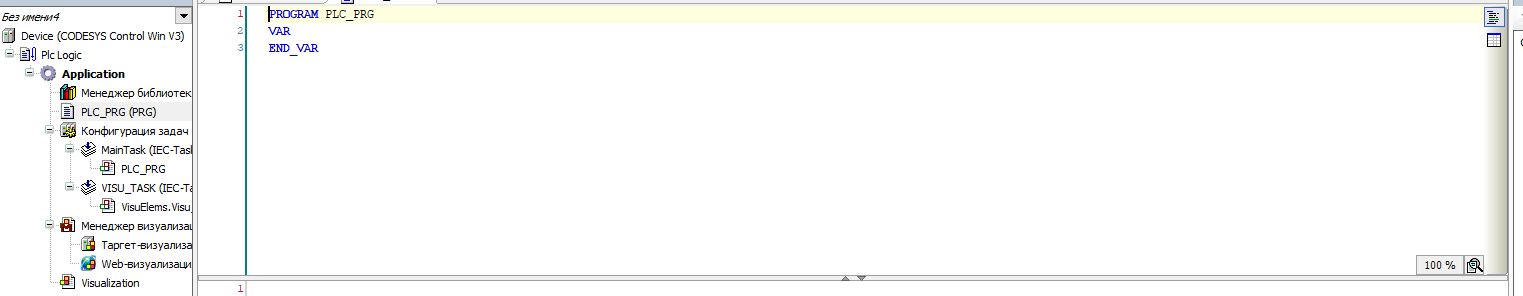
Выберем примитив Кнопка1. Активируем Дополнительные свойства. Выберем нужный цвет Кнопки (Рис. 5).

  
Рисунок 2.5 — **Выбор цвета Кнопки**

Установим Градиент цвета (Рис. 6)

  
Рисунок 2.6 — **Установка градиента цвета**

Далее добавим скрипт на ранее добавленную кнопку на языке ST. При нажатии на кнопку счетчик должен прибавлять единицу и выводить результат.

  
Рисунок 2.7 — **Добавление плейсхолдера**

iClicks: INT;

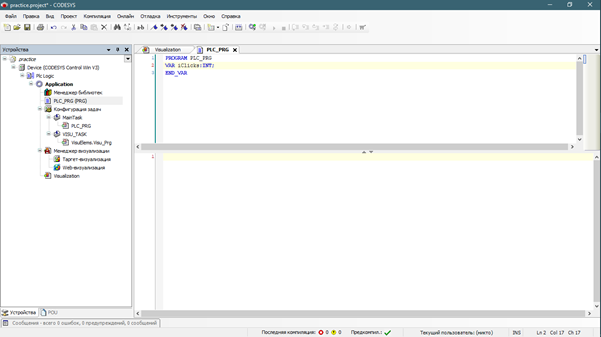


Рисунок 2.8 - **Объявление переменной типа INT**

Привязываем переменную к Кнопке (Рис. 9). После добавления переменной – она отобразиться в свойствах Кнопки (Рис. 10).

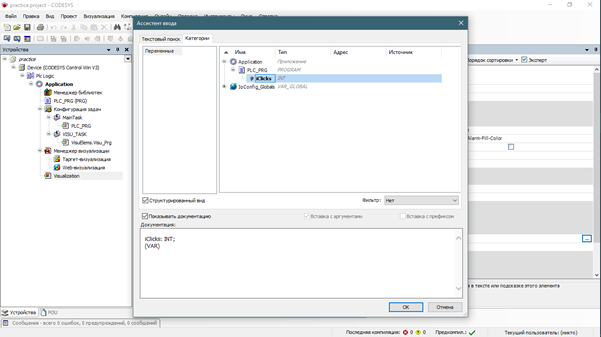
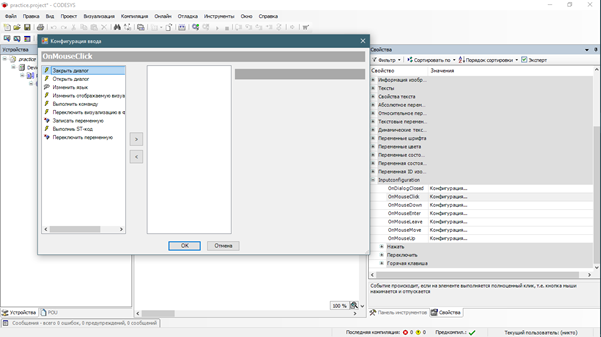


Рисунок 2.9 — **Добавление переменной в свойства кнопки**

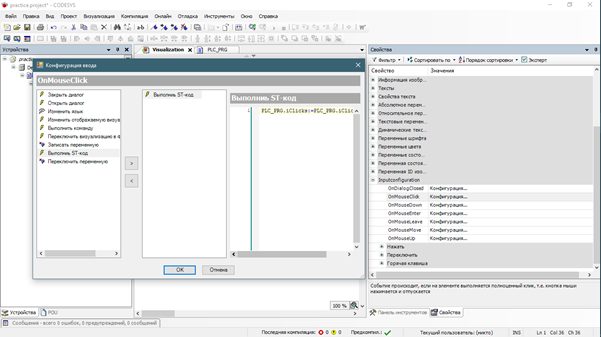
  
Рисунок 2.10 — **Отображение переменной в свойствах Кнопки**

Теперь во время исполнения программы вместо плейсхолдера будет выводиться значение переменной.

Выбираем действие при нажатии мышки на кнопку (Рис. 11).

  
Рисунок 2.11 - **Конфигурация действия при нажатии на кнопку**

Вводим код, который будет выполняться при нажатии на кнопку (Рис. 12).

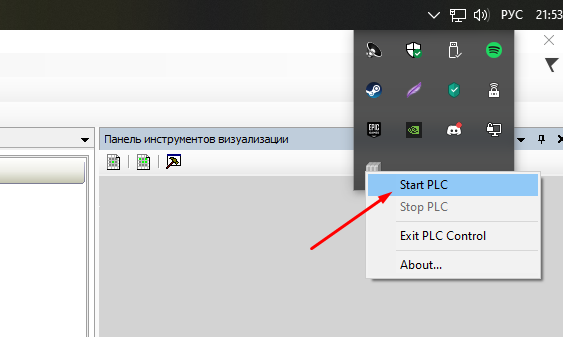
  
Рисунок 2.12 - **Код программы (действия) Кнопки**

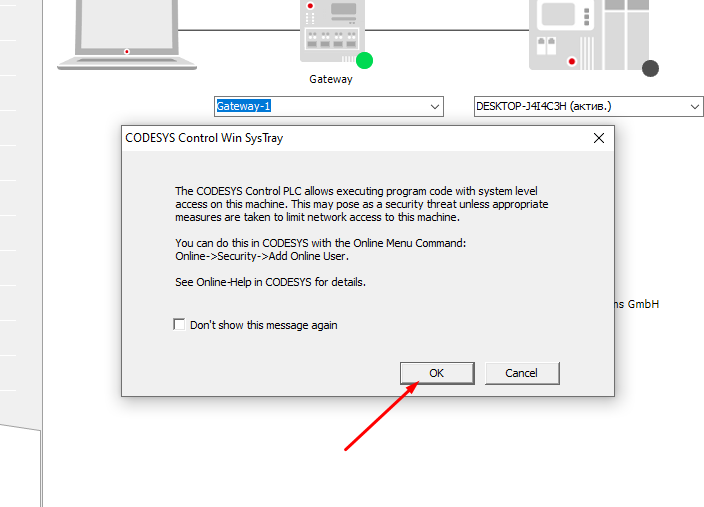
После всех настроек – код действия отобразиться в свойствах Кнопки (Рис. 13).

  
Рисунок 12.13 - **Отображение кода действия в свойствах Кнопки**

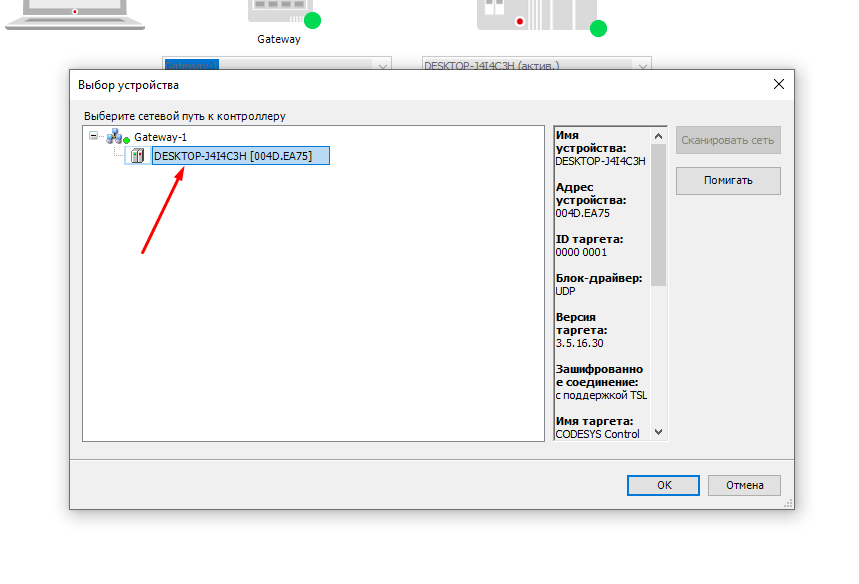
### **2.1.4. Запуск программы**

Для запуска программы требуется запустить SoftPLC (Рис. 14-15).

  
Рисунок 2.14 - **Старт Soft PLC**

  
Рисунок 2.15 - **Подтверждение старта Soft PLC**

После запуска SoftPLC требуется просканировать сеть и убедится в том, что ПЛК был запущен (Рис. 16).

  
Рисунок 2.16 - **Сканирование сети**

### **2.1.5 Результат работы программы**

Конечный результат работы программы представлен на рисунках 17 и 18. По нажатию на кнопку изменяется счётчик нажатия.



Рисунок 2.17 - **Увеличение счетчика, при нажатии на кнопку (1)**

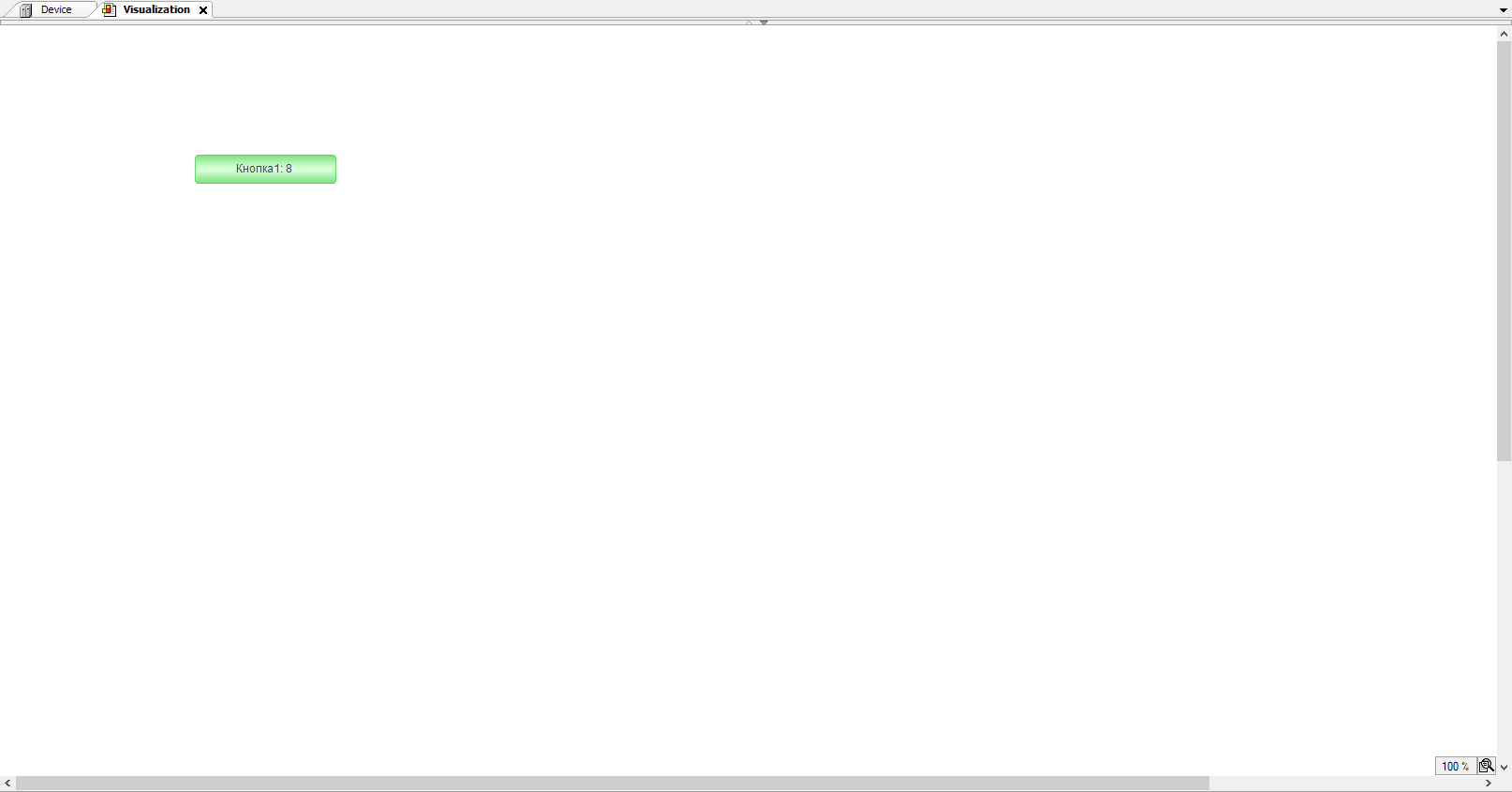


Рисунок 2.18 - **Увеличение счетчика, при нажатии на кнопку (2)**

* 1. Отчёт №2 «Типичные параметры графических примитивов».

### **2.2.1 Цель работы**

Изучение простейших графических примитивов в графическом редакторе Codesys 3.5 и их основных параметров, ознакомится с графическими примитивами: Прямоугольник, Список текстов, Управление вращением, Полоса прокрутки, Бегунок, Метка, Текстовое поле, Радиокнопка, Клавишный выключатель, Переключатель изображения, Индикатор.

### **2.2.2 Выполнение практической работы.**

Вначале реализуется новый базовый проект CODESYS с названием Example\_Rectangle и базовыми установками: с таргетом – CODESYS Control Win V3x64 и языком программирования PLC\_PRG – ST.

В программе PLC\_PRG описываются новые переменные (рисунки 19-21)

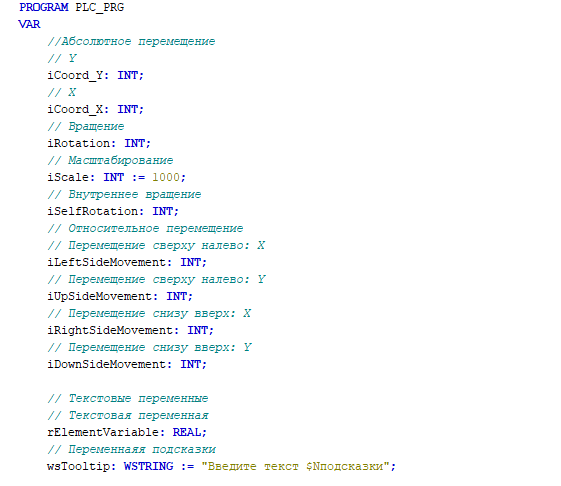


Рисунок 2.19— **Описание переменных.**



Рисунок 2.20 — **Описание переменных.**

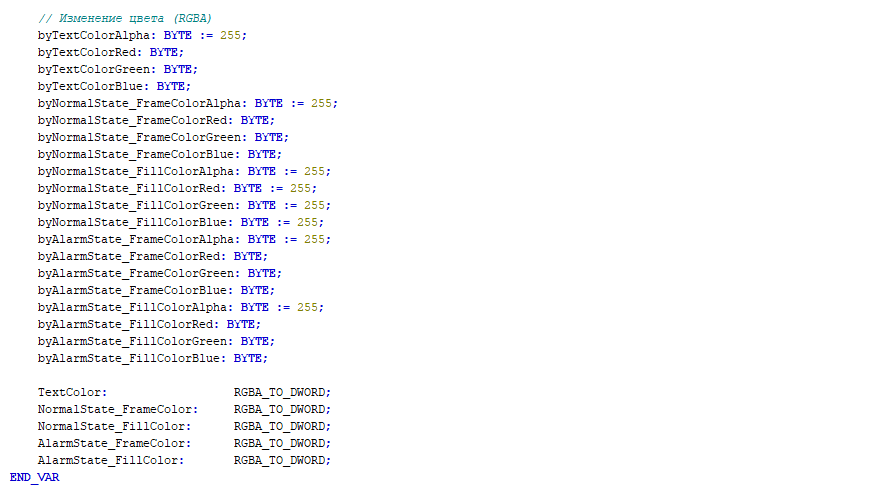


Рисунок 2.21 — **Описание переменных.**

Для управления цветом реализуется функциональный блок RGBA\_TO\_DWORD (рисунок 22)

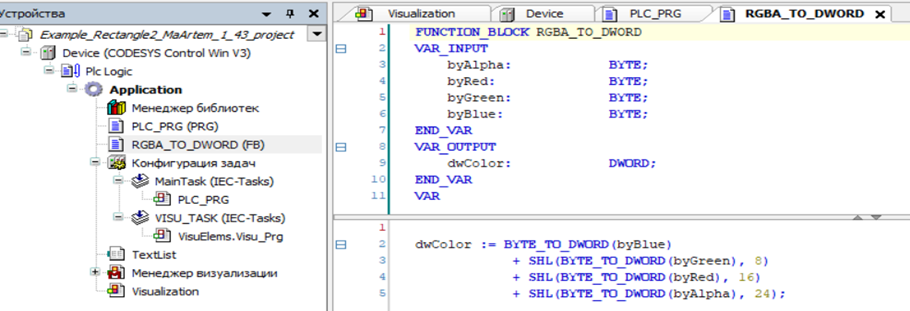


Рисунок 2.22 —  **Реализация функционального блока**

Далее публикуются в PLC\_RPG экземпляры созданного нами функционального блока (рисунок 23)

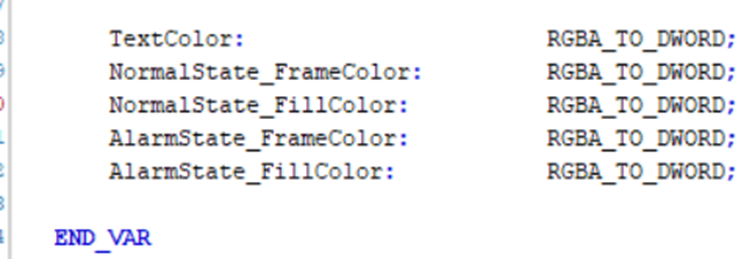


Рисунок 2.23 —  **Публикация переменных функционального блока.**

Программный код для PLC\_PRG реализуется следующим образом (рисунок 24)

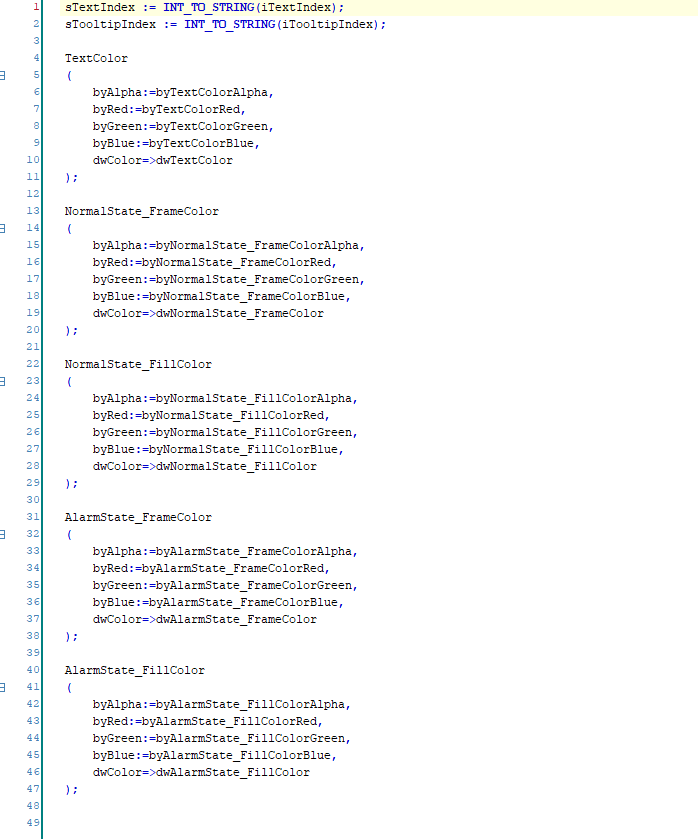


Рисунок 2.24 — **Программный код PLC\_PRG**

Далее создается окно визуализации Visualisation и добавляется список текстов в проект TextList.

В окне визуализации создаются и располагаются группы и примитивы согласно методическим указаниям (рисунок 25)

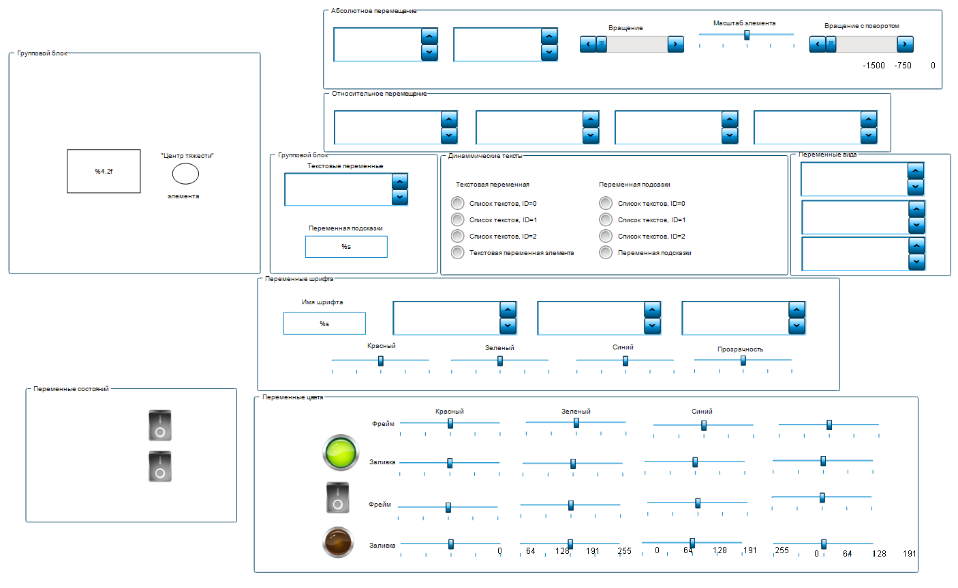


Рисунок 2.25 —  **Расположение групп и примитивов**

По группам задаются параметры примитивов:

1. Абсолютное перемещение: два примитива Управление вращением, которые используют параметры iCoordX и iCoordY, Полоса прокрутки, изменяющая параметр iRotation и *Бегунок* с параметром iScale.
2. Относительное перемещение: четыре примитива Управление вращением, которые задают параметры iLeftSideMovement, iUpSideMovement, iRightSideMovement, iDownSideMovement.
3. Текстовые переменные: примитив Управление вращением, изменяющий переменную rElementVariable и примитив Текстовое полев который вводится и записывается параметр wsTooltip.
4. Динамические тексты: два примитива Радио-кнопкас параметрами iTextIndex и iTooltipIndex.
5. Переменные вида: три примитива Управление вращением, меняющие значения iLineThickness, dwFillEnable, dwLineStyle.
6. Переменные шрифта: примитив Текстовое полес текстовой переменной iFontName, два примитива Управление вращением с переменными iTextSize и iTextAligment, три примитива Бегунок, меняющие значения byTextColorRed, byTextColorGreen, byTextColorBlue.
7. Переменная цвета: 16 примитивов Бегунокс переменными byNormalState\_FrameColorRed, byNormalState\_FrameColorGreen, byNormalState\_FrameColorBlue, byNormalState\_FrameColorAlpha, byNormalState\_FillColorRed, byNormalState\_FillColorGreen, byNormalState\_FillColorBlue, byNormalState\_FillColorAlpha, byAlarmState\_FrameColorRed, byAlarmState\_FrameColorGreen, byAlarmState\_FrameColorBlue, byAlarmState\_FrameColorAlpha,
8. byAlarmState\_FillColorRed, byAlarmState\_FillColorGreen, byAlarmState\_FillColorBlue, byAlarmState\_FillColorAlpha; примитив Клавишный выключатель,задающий значения для bSwitchStateColor, примитивы Переключатель изображения и Индикатор*,* зависящие от значения bSwitchStateColor.
9. Переменная состояний: два примитива Клавишный выключательс переменными bInvisible и bNoTouchReaction.
10. Создаётся примитив Прямоугольник, в который записываются все выше перечисленные параметры из групп. Код для перезагрузки всех значений представлен в следующем виде (рисунок 26)



Рисунок 2.26— **Код для перезагрузки значений**

### **2.2.3 Результат работы программы**

Конечный результат работы программы представлен на рисунке 27.

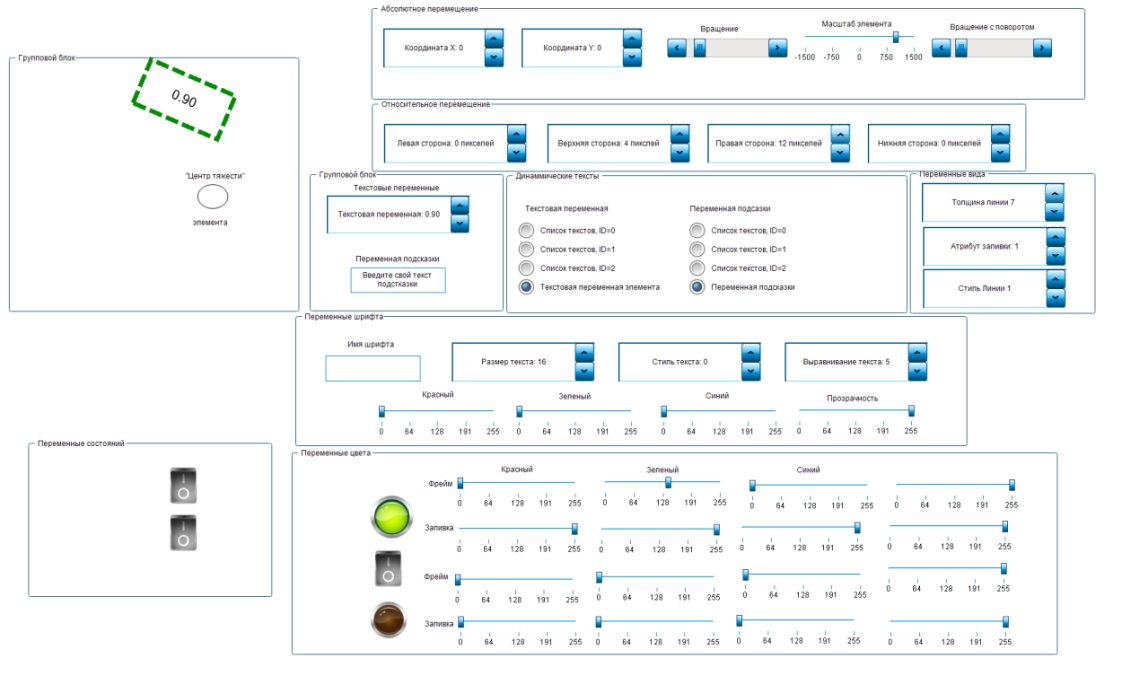


Рисунок 2.27 — **Результат работы программы**

* 1. Отчёт №3 «Работа с Фреймами».

### **2.3.1 Цель работы**

Изучение графического примитива Фрейм в графическом редакторе Codesys 3.5, который позволяет открывать в плоскости одних экранов визуализацию других экранов.

### **2.3.2 Выполнение практической работы**

Был создан новый стандартныйпроект CODESYS с именем «Example\_Frame» и настроен по умолчанию (рис 28).

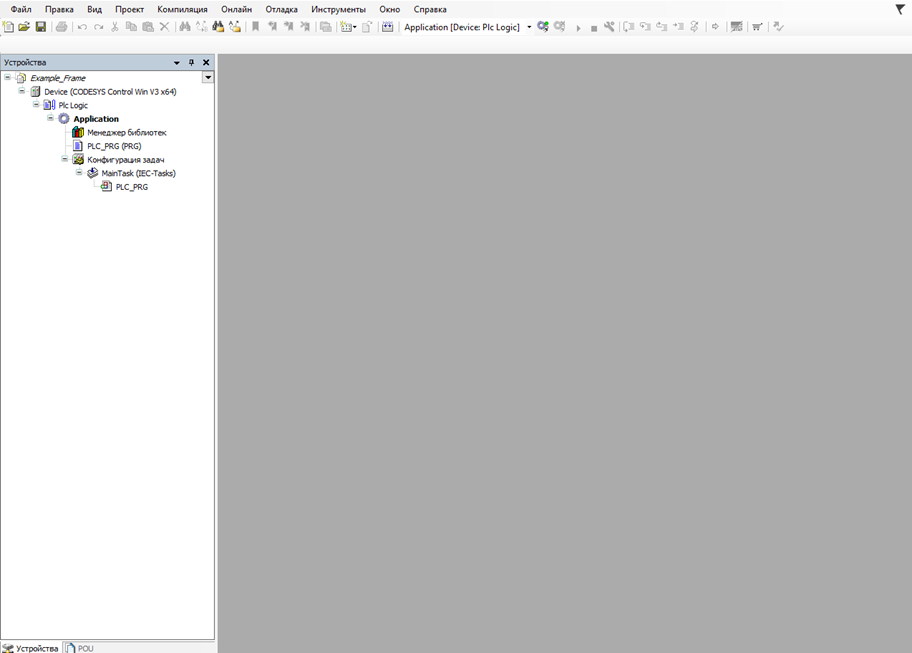


Рисунок 2.28 — **Создание проекта «Example\_Frame»**

В программе PLC\_PRG были объявлены следующие переменные (рисунок 29).

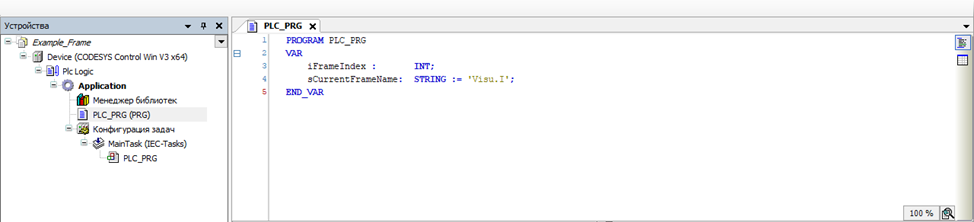


Рисунок 2.29 — **Объявление переменных программы PLC\_PRG**

Добавлен экран визуализации Frame\_0 в проект.

Добавлен экран Frame\_1 и Frame\_2 в проект визуализации (Frame\_1 – синий, Frame\_2 – красный).

Добавлен экран визуализации *Visu* в проект (рисунок 30).

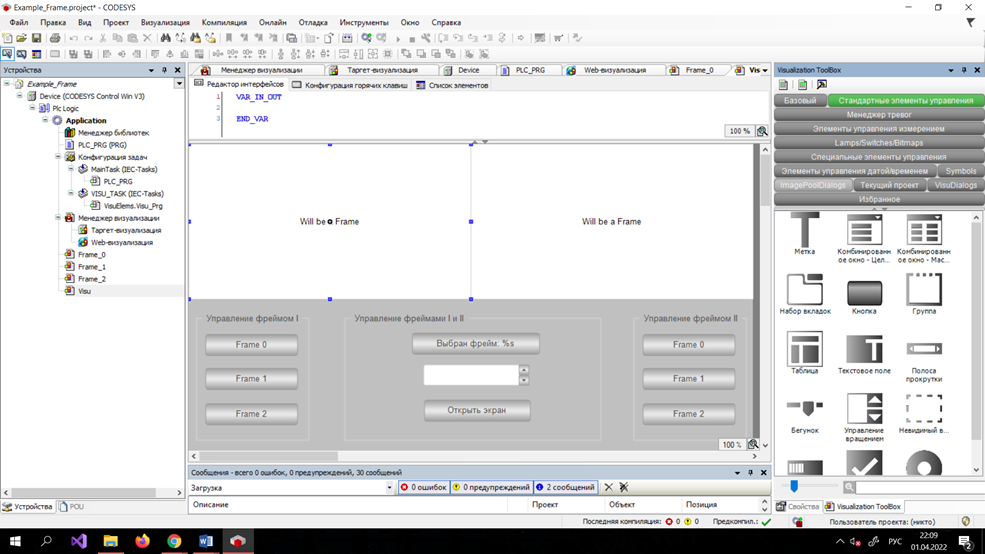


Рисунок 2.30 — **Экран визуализации *Visu (фреймы без конфигурации)***

Экран содержит следующие примитивы:

* два примитива *Фрейм* с пустой конфигурацией;
* вкладку *Управление фреймом I* (примитив *Группа*) с тремя примитивами Кнопка;
* вкладку *Управление фреймом II* (примитив *Группа*) с тремя примитивами Кнопка;
* вкладку *Управление фреймами I и II* (примитив *Группа*) с двумя примитивами *Кнопка* и одним примитивом *Управление вращением*.

Заданы параметры первого примитива *Фрейм*. Были настроены конфигурации примитивов. Были настроены примитивы кнопок, привязаны *фреймы.* Была параметризирована панель «Управление фреймом I и II». Параметризирован примитив «*Управление вращением*». При помощи него будет производиться выбор индекса экрана, открываемого в фрейме. Был параметризирован компонент «*Менеджер визуализации*».

### **2.3.3 Результат работы программы**

Конечный результат работы программы представлен на рисунке 31.

Нажатие на кнопки Frame\_0, Frame\_1, Frame\_2 панели «*Управление фрей-мом I*» активизирует соответствующий экран в этом фрейме.

Нажатие на кнопки Frame\_0, Frame\_1, Frame\_2 панели «*Управление фрей- мом II*» активизирует соответствующий экран в этом фрейме.

Панель «Управление фреймами I и II» работает следующим образом:

Сначала необходимо нажать на кнопку «*Выбран элемент*» и ввести путь к элементу в виде «*Имя экрана визуализации. Имя примитива* (соответственно, в этом случае это Visu.I - для первого фрейма и Visu.II - для второго)».

Затем необходимо выбрать индекс фрейма 3.

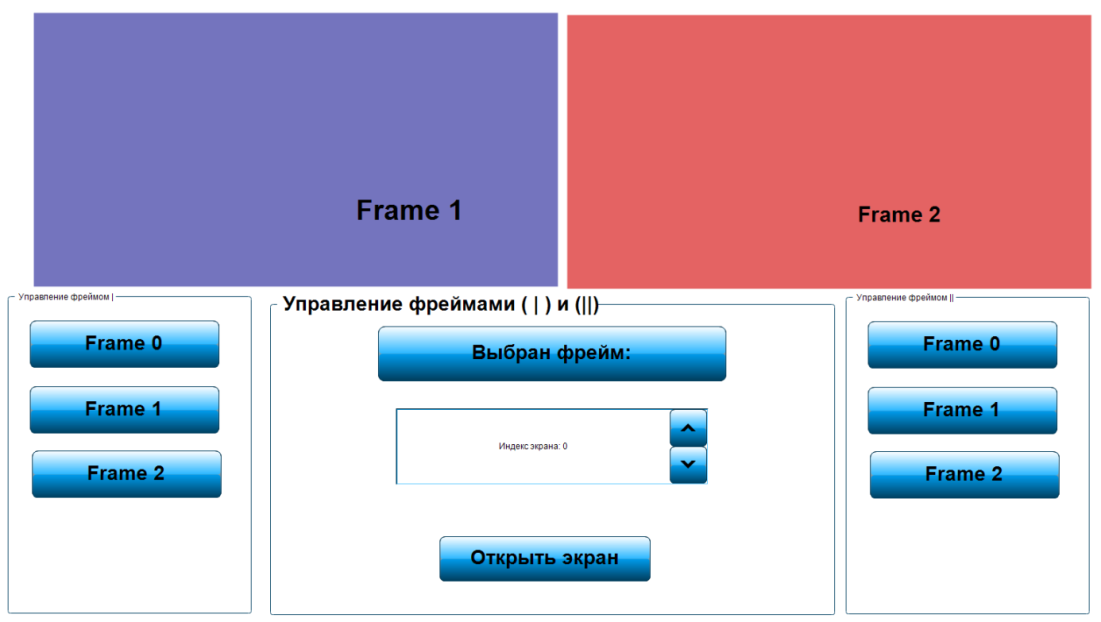


Рисунок 2.31 —  **Конечный результат работы программы**

## 2.4 Отчёт №4 «Комбинированное окно»

### **2.4.1 Цель работы**

Изучение графического примитива Фрейм в графическом редакторе Codesys 3.5, который позволяет открывать в плоскости одних экранов визуализацию других экранов.

### **2.4.2 Выполнение практической работы**

Создать базовый проект CODESYS. В программе PLC\_PRG обозначить следующие переменные (рисунок 32)

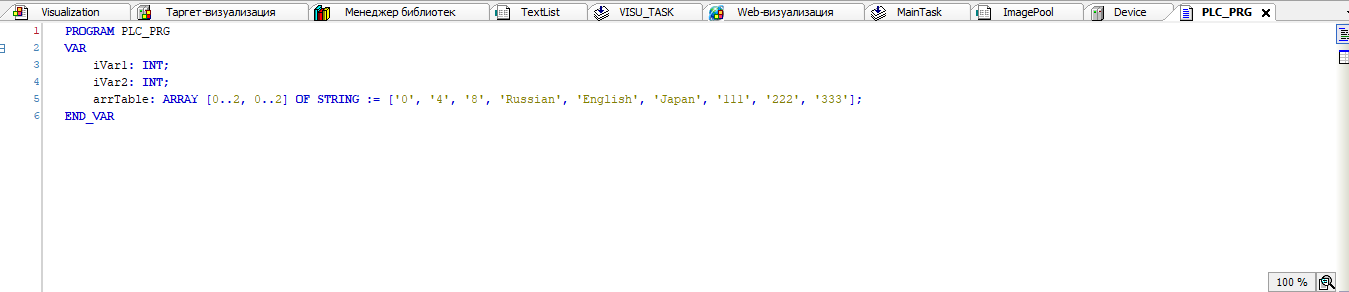


Рисунок 2.32 – **Обозначение переменных**

Необходимо добавить в проект список изображений с названием «ImagePool». Добавить в него три картинки с ID 0, 4 и 8 (Рисунок 33).

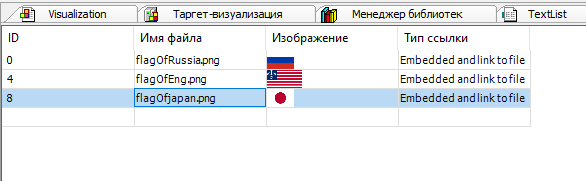


Рисунок 2.33 – **Cодержимое списка изображений ImagePool**

Добавить в проект список текстов с названием «TextList». Его ID будут идентичны ID из списка изображений (Рисунок 34).

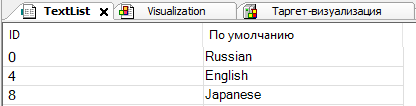


Рисунок 2.34 – **Содержимое списка текстов «TextList»**

Добавить в проект экран визуализации с названием «Visualization». В его параметрах указать размер 800 × 480. Добавить на экран два примитива «Группа», примитив «Комбинированное окно – Целочисленное», примитив «Комбинированное окно – Массив» и два примитива «Прямоугольник» (рисунок 35).

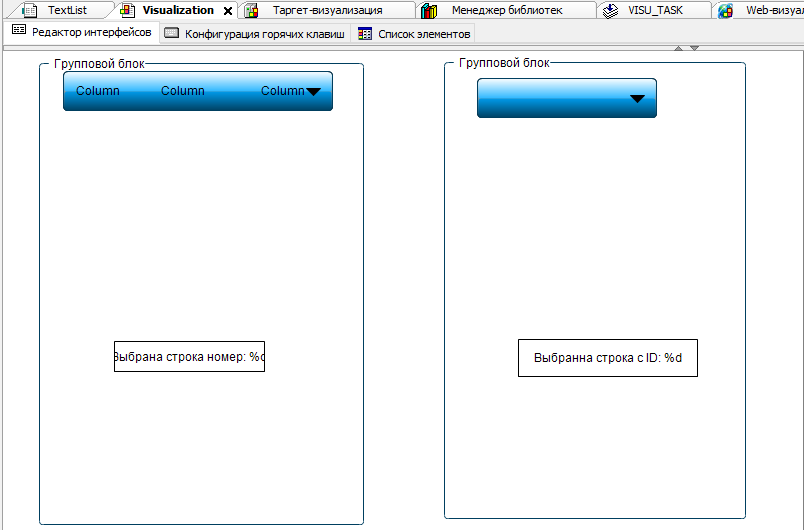


Рисунок 2.35 - **Содержимое экрана «Visaulization»**

Необходимо спараметризировать элемент «Комбинированное окно – Целочисленное» (рисунок 36).

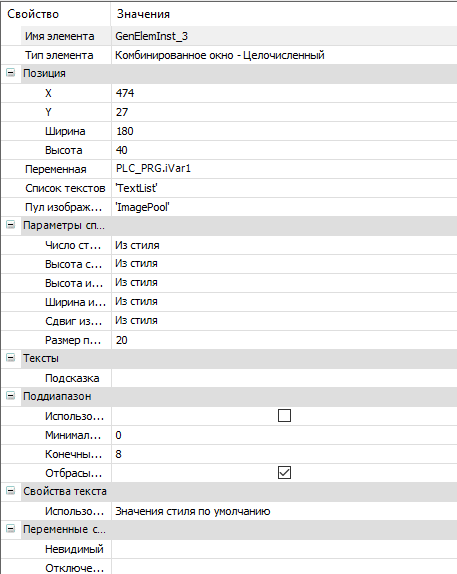


Рисунок 2.36 – **Параметры примитива «Комбинированное окно – Целочисленное»**

Настроить примитив «Комбинированное окно – Массив» (рисунок 37).

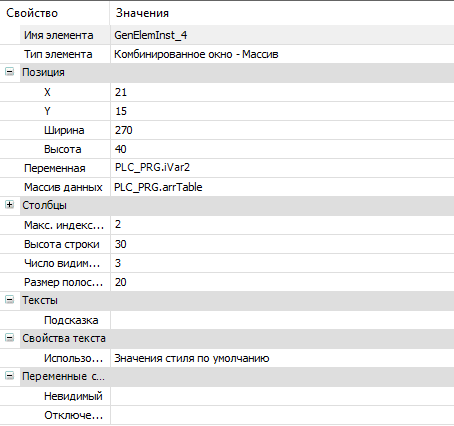


Рисунок 2.37 – **Параметры примитива «Комбинированное окно – Массив»**

Необходимо настроить примитивы «Прямоугольник». Они будут различаться переменными iVar1 для первого iVar2 для второго (рисунок 38).

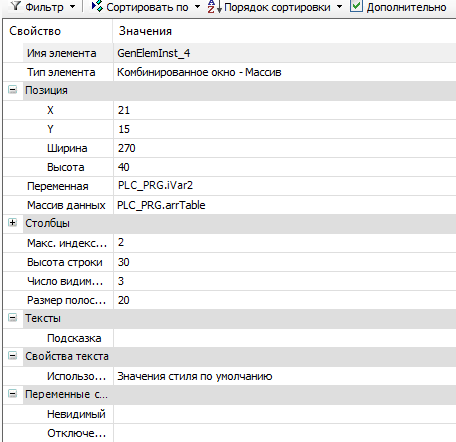


Рисунок 2.38 – **параметры примитива «Прямоугольник II»**

Необходимо параметризировать «Менеджер визуализации». В его параметрах выбрать галочку «Использовать строки Unicode» (Рисунок 39).

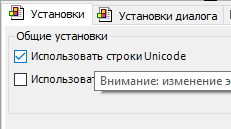


Рисунок 2.39 **– Параметризация Unicode**

Параметры таргет и web визуализации представлены на рисунках 40-41.

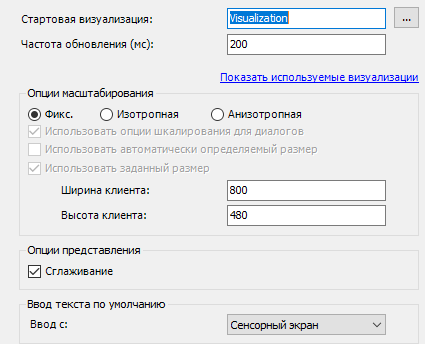


Рисунок 2.40 – **Параметры таргет-визуализации**

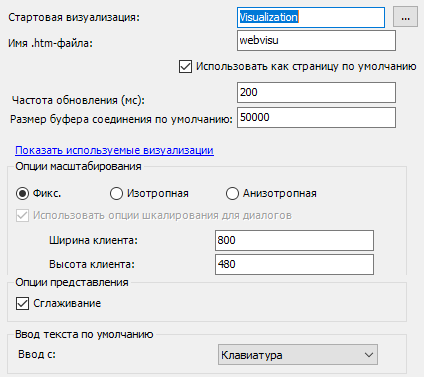


Рисунок 2.41 – **Параметры web-визуализации**

### **2.4.3 Результат работы программы**

Результат работы программы представлен на рисунке 42:

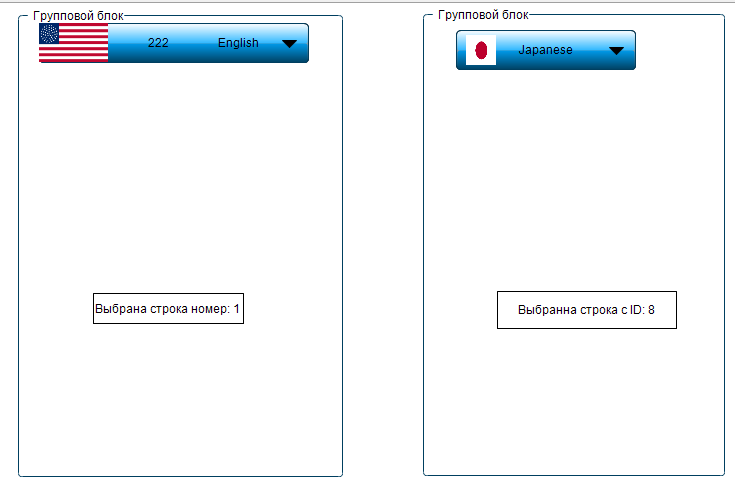


Рисунок 2.42 – **Конечный результат работы программы**

ЗАКЛЮЧЕНИЕ

В ходе практики был приобретён опыт с разработкой в интегрированной среде разработки (IDE) приложений для программируемых контроллеров CODESYS, а также создании интерфейсов и работой с примитивами. В теоретической части были изучены особенности ПЛК и АСУ ТП.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Что такое АСУ ТП? URL: "https://www.o-matic.ru/about/41-about-asu.html" (Дата обращения: 15.05.2022)
2. Компэл. Введение в ПЛК: что такое программируемый логический компьютер. URL:"https://www.compel.ru/lib/95591" (Дата обращения: 15.05.2022)
3. Уровни АСУ ТП URL:"http://opiobjektid.tptlive.ee/Automatiseerimine/9\_\_\_.html" (Дата обращения: 15.05.2022
4. ASUTPP. Языки программирования PLC: LD, FBD, SFC, ST, IL, CFC URL: "https://www.asutpp.ru/yazyki-programmirovaniya-plc.html" (Дата обращения: 15.05.2022)
5. Типы ПЛК URL:" http://plcbox.ru/plctype/" (Дата обращения: 15.05.2022)
6. [Информационно-библиотечный центр МИРЭА - Российского технологического университета](Информационно-библиотечный%20центр%20МИРЭА%20-%20Российского%20технологического%20университета). Основы разработки графических интерфейсов в среде Codesys 3.5. Ч.1. [URL: "https://library.mirea.ru/share/4407](URL:%20%22https://library.mirea.ru/share/4407)" (Дата обращения: 15.05.2022)