



Министерство науки и высшего образования Российской Федерации

федеральное государственное бюджетное образовательное  
учреждение высшего профессионального образования  
«Санкт-Петербургский государственный технологический  
институт (технический университет)»

---

Кафедра систем автоматизированного проектирования и  
управления

О.В. Ершова, А.М. Полякова

**ИНТЕГРИРОВАННАЯ КОМПОНЕНТНАЯ СРЕДА  
РАЗРАБОТКИ WONDERWARE DEVELOPMENT STUDIO.  
СОЗДАНИЕ ПРИЛОЖЕНИЙ INTOUCH NMI**

Часть 1

Методические указания к  
лабораторным работам

Санкт-Петербург  
2018

## **УДК 004.9SCADA: 681.5 (075.4)**

О. В. Ершова, А. М. Полякова Интегрированная компонентная среда разработки Wonderware Development Studio. Создание приложений InTouch HMI. Часть 1. Метод. указания. – СПб.: СПбГТИ(ТУ), 2018. – 58 с.

Методические указания предназначены для выполнения цикла лабораторных работ «Изучение SCADA-системы InTouch».

Рассматриваются основные функции SCADA HMI InTouch: события в системе, алармовые ситуации, тренды реального времени, аналитические кривые, обмен данными по DDE-протоколу и предлагаются упражнения для их освоения.

Методические указания предназначены для студентов направления подготовки 09.04.01 «Информатика и вычислительная техника» при изучении дисциплины «Интегрированные системы проектирования и управления»; соответствуют рабочей программе дисциплины.

Ил. 75, библиогр. 6 назв.

Рецензент: Е.А. Гордина, начальник Учебного центра «Севзапмотажавтоматика».

Утверждены на заседании учебно-методической комиссии факультета информационных технологий и управления « 24 » января 2018 г.

Рекомендованы к изданию РИСо СПбГТИ(ТУ).

## Содержание

|   |    |
|---|----|
| ВВЕДЕНИЕ.....   | 4  |
| ЛАБОРАТОРНАЯ РАБОТА 1 .....   |    |
| 1.1 Основные компоненты SCADA-системы InTouch.....                          | 6  |
| 1.2 Среда разработки Window Maker и среда исполнения<br>Window Viewer ..... | 7  |
| 1.3 Работа с окнами и графическими объектами .....                          | 9  |
| 1.4 Запуск приложения на исполнение .....                                   | 10 |
| 1.5 Цель работы .....   |    |
| 1.6 Выполнение работы .....   |    |
| ЛАБОРАТОРНАЯ РАБОТА 2 .....   |    |
| 2.1 Словарь переменных.....   | 26 |
| 2.2 Анимационные связи .....  | 30 |
| 2.3 Цель работы .....   | 32 |
| 2.4 Выполнение работы .....   | 32 |
| ЛАБОРАТОРНАЯ РАБОТА 3 .....   |    |
| 3.1 Скрипты .....   |    |
| 3.2 Цель работы .....   |    |
| 3.3 Выполнение работы .....   |    |
| ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ .....  | 56 |
| СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ .....  | 57 |

## ВВЕДЕНИЕ

Wonderware Development Studio (WDS) предоставляет комплексный набор инструментов для быстрой разработки и развертывания приложений Wonderware для промышленной автоматизации. Ядром Development Studio служит интегрированная компонентная среда разработки (IDE), в рамках которой инженерный персонал может заниматься разработкой, созданием, тестированием и развертыванием любых промышленных приложений автоматизации. Не покидая среды разработки IDE, пользователи могут создавать эффективные и сложные графические представления и включать их в HMI, SCADA, MES приложения или системы производственного интеллекта.

Основные характеристики WDS:

- единая среда разработки централизованного управления и удаленного развертывания HMI, SCADA, MES и EMI приложений;
- шаблонно-ориентированная разработка объектов и графических элементов;
- полная и настраиваемая библиотека графических элементов
- возможность использования технических наработок, созданных в предыдущих проектах;
- возможности расширения при разработке на базе Microsoft .NET и использовании Wonderware API;
- быстрое распространение изменений по сетевым узлам;
- многоуровневая защита и журналы регистрации событий;
- управление устройствами ввода-вывода, архивом, алармами, событиями, графикой, сценариями и др. с помощью одного инструмента.

Программное обеспечение InTouch HMI предоставляет широкие возможности графической визуализации, что позволяет перевести на качественно новый уровень оптимизацию, контроль и управление производственными

процессами. InTouch – это один из компонентов пакета для полной многоуровневой автоматизации Wonderware FactorySuite, который является одной из самых распространенных систем создания человеко-машинных интерфейсов (HMI) для операционных систем класса Windows.

Таким образом, WDS обеспечивает процесс сбора информации реального времени с удаленных объектов для обработки, анализа и возможного управления. InTouch-приложения охватывают огромное многообразие вертикальных рынков, включая переработку нефти и газа, химическую, фармацевтическую, автомобильную, бумажную и другие отрасли промышленности.

В первой части методических указаний рассматриваются вопросы, касающиеся созданию приложений с помощью InTouch HMI – окон и графических объектов, разработка базы данных, изучаются функции импорта окон из других приложений, настройка анимационных связей и разработка скриптов различных типов.

# ЛАБОРАТОРНАЯ РАБОТА 1

## 1.1 Основные компоненты InTouch HMI

С помощью InTouch можно создавать мощные полнофункциональные приложения, опирающиеся на такие возможности Windows, как ActiveX-технология, OLE, графика, сетевые средства и т.д. InTouch легко дополняется специализированными ActiveX-компонентами, wizard-средствами, обобщенными объектами, а также создаваемыми расширениями InTouch QuickScript.

Программное обеспечение InTouch состоит из следующих основных компонентов: проводник приложений **Application Manager**, среда разработки **Window Maker**, среда исполнения **Window Viewer**, а также программа диагностики **Wonderware Logger**.

С помощью проводника приложений **Application Manager** можно управлять созданными приложениями, задавать параметры разработки сетевых приложений для клиентских и серверных архитектур.

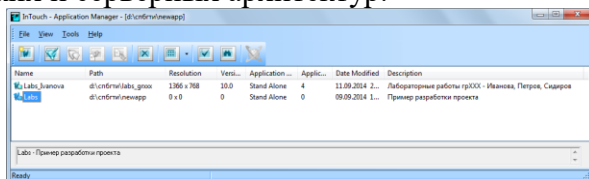


Рисунок 1

**Window Maker** – это среда разработки для создания анимированных сенсорных окон. Созданные дисплейные окна могут быть соединены с промышленным контроллером ввода/вывода и другими Windows-приложениями.

**Window Viewer** – это среда исполнения, которая позволяет работать с созданными в среде разработки Window Maker дисплейными окнами, выполняет скрипты, осуществляет регистрацию и генерацию отчетов, а также может работать в качестве клиента или сервера с протоколами DDE и SuiteLink.

## 1.2 Среда разработки Window Maker и среда исполнения Window Viewer

Среда разработки **Window Maker** поддерживает стандартные графические интерфейсы пользователя и легко настраивается под требования пользователя. По умолчанию при первом запуске Window Maker в окне этой программы автоматически отображаются все имеющиеся средства, включая все панели инструментов, как показано на рисунке 2.

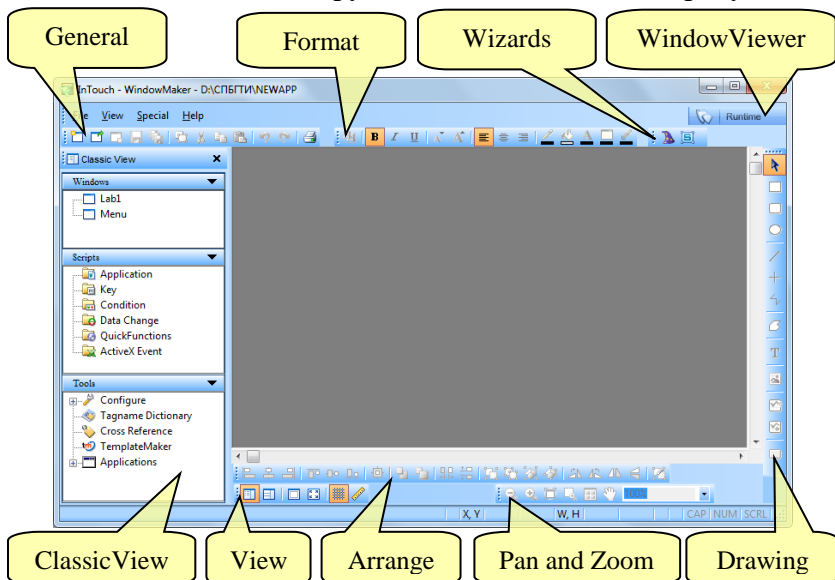


Рисунок 2

**Classic View** является иерархическим и графическим средством просмотра разрабатываемого приложения. В нем показаны основные объекты приложения – окна, скрипты, словарь тегов (переменных), дополнительные приложения и настройки, тем самым обеспечивается быстрый доступ к разрабатываемому проекту, командам и функциям Window Maker.

На панели инструментов **General** сгруппированы инструменты, выполняющие большинство команд управления окнами, которые располагаются в меню **File**, и

стандартные инструменты управления буфером обмена меню **Edit**.

На панели инструментов **Format** сгруппированы инструменты, выполняющие команды форматирования текста, а также инструменты цветовой палитры для изменения цветов линии, текста, заливки, фона и прозрачных объектов.

На панели инструментов **Drawing** сгруппированы все инструменты, используемые для рисования простых графических объектов (прямоугольники, эллипсы, линейные или текстовые объекты), растровых изображений, графов реального времени, исторических графов, трехмерных кнопок с надписями.

На панели инструментов **View** сгруппированы инструменты, которые используются для управления состоянием окна Window Maker

На панели инструментов **Arrange** сгруппированы инструменты, выполняющие команды компоновки объектов в окне.

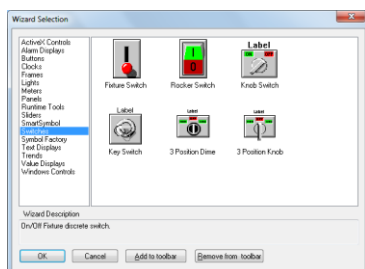


Рисунок 3

Панель инструментов **Wizards**, представленная на рисунке 3, позволяет использовать ряд готовых стандартных объектов, настраивать их анимационные функции и редактировать по своему усмотрению.

На панели инструментов **Pan and Zoom** расположены инструменты масштабирования.

В нижнем правом углу окна расположена строка состояния, в которой при выборе объекта отображаются координаты верхнего левого угла этого объекта, а также его ширина и высота в пикселях.



### 1.3 Работа с окнами и графическими объектами

Приложение, разработанное в SCADA-системе InTouch, состоит из многочисленных окон, созданных в среде разработки Window Maker, отображающих графические и текстовые объекты. При создании нового окна на экране появляется окно с соответствующими настройками – цвет фона окна, тип окна, тип рамки, строка заголовка, возможность изменения размеров окна, координаты расположения и размеры окна, как показано на рисунке 4. К окну можно привязать Quick-сценарии, которые будут выполняться в момент открытия, отображения или закрытия окна.

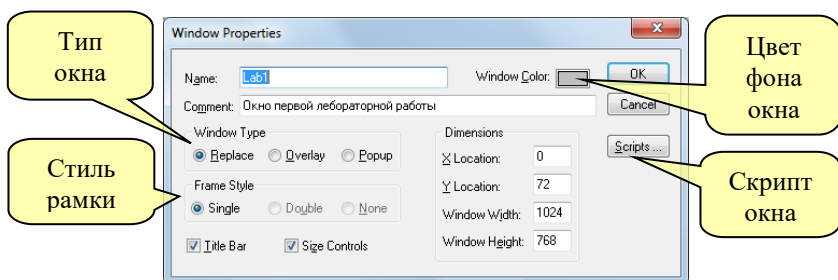


Рисунок 4

#### Тип окна:

- **replace** (заменяющий тип) - автоматически закрывает любые окна, с которыми оно пересекается на экране, включая окна всплывающего типа и другие заменяющие;
- **overlay** (перекрывающий тип) - появляется поверх любых ранее открытых окон и может полностью перекрывать их. После закрытия перекрывающего окна все другие окна, которые за ним скрывались, будут снова видны. Если нажать на любую видимую часть окна, скрытого за перекрывающим окном, это окно перейдет на передний план и станет активным;
- **popup** (всплывающий тип) – аналогично перекрывающему типу за тем исключением, что такое окно всегда остается поверх других открытых окон (даже при

нажатию на другое окно). Для закрытия всплывающего окна обычно требуется ответное действие пользователя.

### Стиль рамки:

- **single** (одинокая) рамка используется для окон с трехмерной рамкой, при необходимости, возможно настроить отображение строки заголовка окна (**title bar**) и управлять его размером (**size controle**);
- **double** (двойная) рамка используется для окон с трехмерной рамкой без строки заголовка и без возможности управления размером окна;
- **none** рамка используется для окон без трехмерной рамки и без возможности изменять размеры окна.

## 1.4 Запуск приложения на исполнение

Из среды Window Maker для проверки и отладки разработанного приложения существует возможность запуска приложения на исполнение в среде **Window Viewer**. Для этого необходимо нажать кнопку Runtime в верхнем правом углу SCADA-системы InTouch, представленную на рисунке 5.

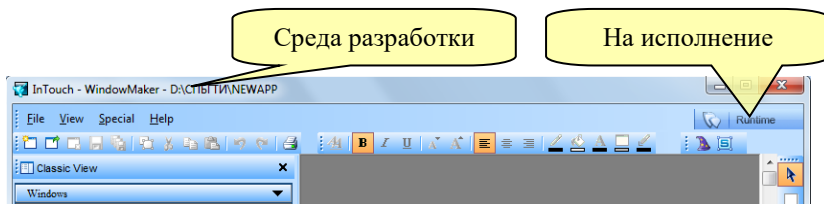


Рисунок 5

Для возврата из среды исполнения Window Viewer в среду разработки Window Maker необходимо нажать кнопку Development, представленную на рисунке 6.

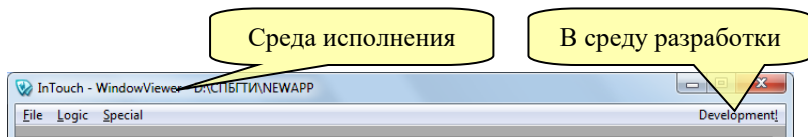


Рисунок 6

## 1.5 Цель работы

В результате выполнения работы необходимо:

- создание нового приложения;
- создание новых окон;
- создание графических объектов;
- использование Wizards-объектов;
- импортирование окна из проекта.

## 1.6 Порядок выполнения работы

Запуск SCADA-системы InTouch

1 В меню Пуск операционной системы Windows выберите: Все программы → Wonderware → InTouch.

При первом запуске InTouch HMI на вашем компьютере необходимо сослаться (или создать, в случае если его нет) на каталог с разрабатываемыми приложениями. Поэтому при первом запуске на экране появится окно, представленное на рисунке 7. Нажмите кнопку «Далее» для перехода к следующему шагу.

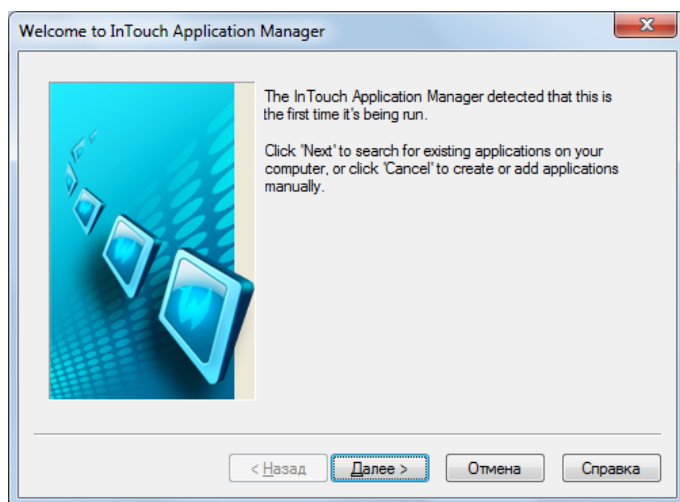


Рисунок 7

2 На экране появится окно, представленное на рисунке 8, в котором необходимо указать каталог хранения разрабатываемых приложений на данном компьютере и нажать кнопку **Готово**.

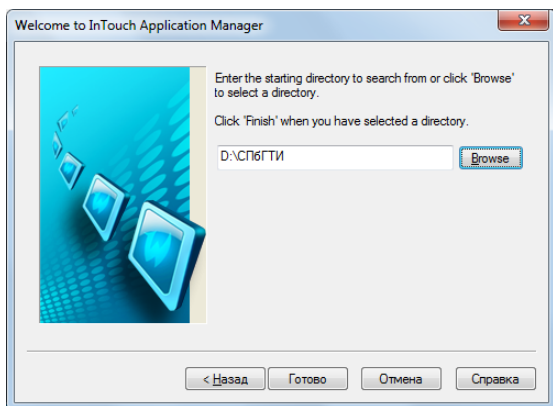
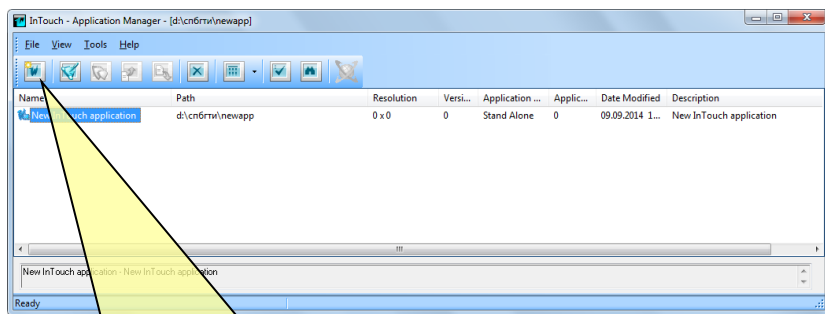


Рисунок 8

3 Если на компьютере уже разрабатывались приложения в SCADA-системе InTouch, то при запуске программы на экране сразу появится окно, представленное на рисунке 9.



Кнопка для создания нового приложения

Рисунок 9

Для создания нового приложения выберите в системном меню **File** → **New** или нажмите на кнопку **New** на панели инструментов (рисунок 9).

3 На экране появится окно, представленное на рисунке 10, в котором необходимо выбрать каталог хранения Вашего приложения и нажать кнопку **Далее**.

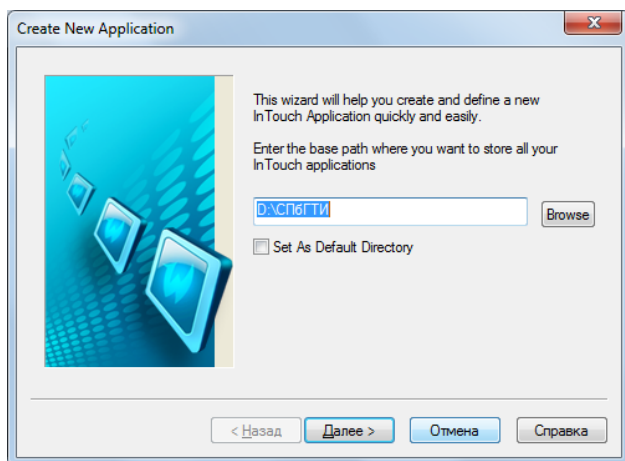


Рисунок 10

4 На экране появится окно, представленное на рисунке 11, в котором необходимо ввести имя каталога хранения ваших лабораторных работ.

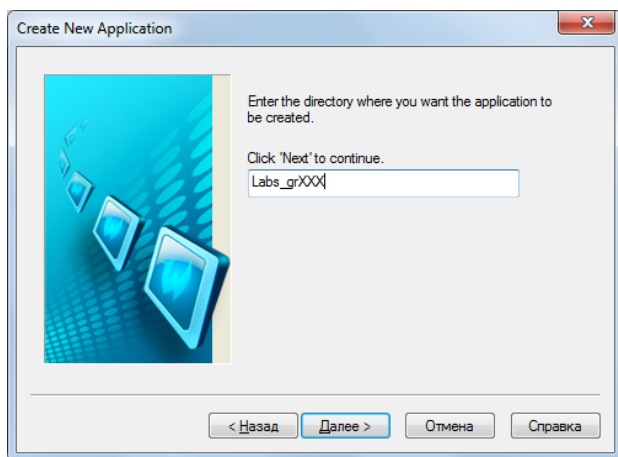


Рисунок 11

Нажмите кнопку **Далее**.

5 На экране появится окно, представленное на рисунке 12. В этом диалоговом окне укажите имя приложения в поле **Name** (которое будет использоваться в списке **Application Manager**) и описание в поле **Description** (для поля **Description** в списке приложений). Нажмите кнопку **Готово**.

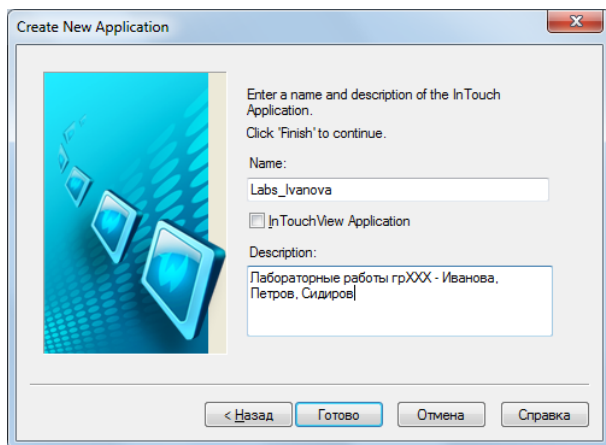


Рисунок 12

6 На экране появится окно **Application Manager**, как показано на рисунке 13. Для перехода в среду разработки **Window Maker** дважды кликните на строке с наименованием Вашего приложения или выберите ваше приложение в представленном списке, а затем в функциональной строке нажмите кнопку **Window Maker**.

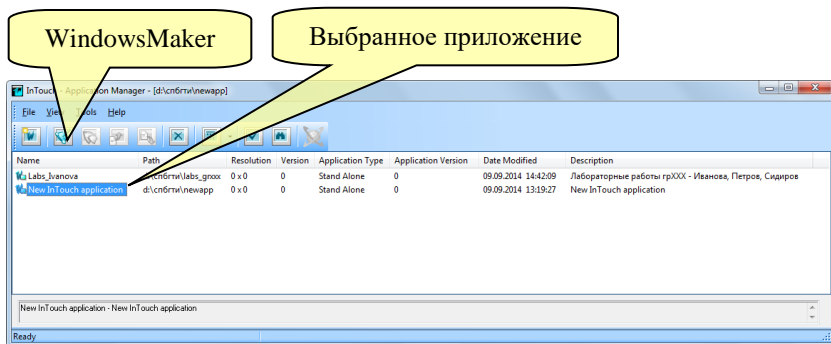


Рисунок 13

\* \* \*

Создание окон приложения и графических объектов  
7 На экране появится окно среды разработки **Window Maker**, представленное на рисунке 14.

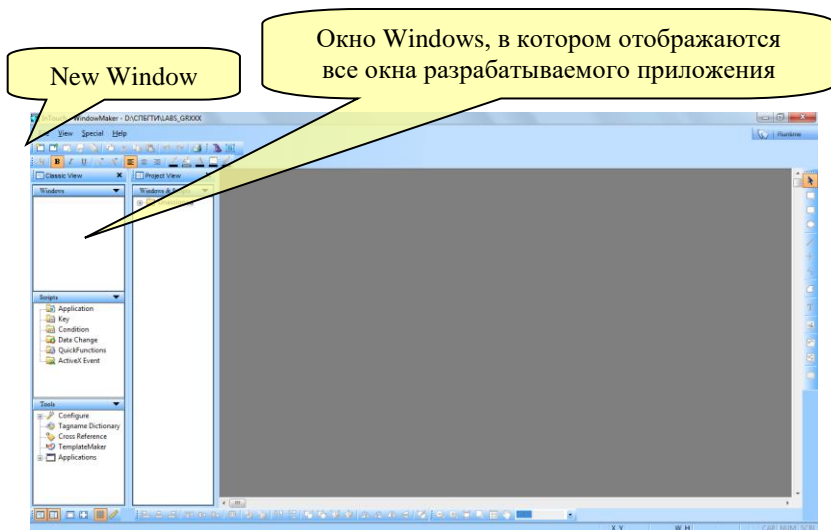


Рисунок 14

Для создания нового окна необходимо нажать кнопку **New Windows** на панели инструментов **General**, или выбрать в системном меню **File** → **New Window**, или в окне **Windows** щелкнуть левой клавишей мыши на пустом поле и в открывшемся контекстном меню выбрать пункт **New**.

Появится диалоговое окно **Window Properties**, представленное на рисунке 15, в котором необходимо указать свойства создаваемого окна.

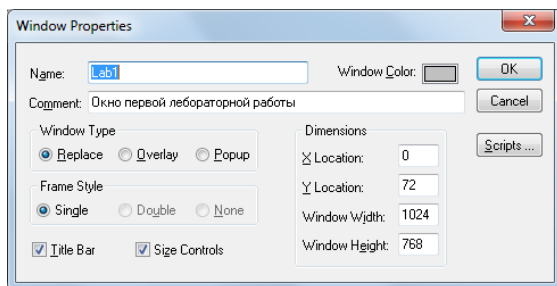


Рисунок 15

Сконфигурируйте следующие свойства окна: в поле **Name** укажите заголовок окна **Lab1**, **Window Type** (тип окна) - replace, **Frame Style** (стиль рамки) – single, **Window Color** (цвет окна), **Dimensions** (размер) – в качестве координат укажите в полях **X Location = 0** , **Y Location = 72**, а ширину и высоту окна **Window Width=1024**, **Window Height=768**.

После задания всех необходимых свойств окна нажмите кнопку **ОК** (рисунок 15).

**8** В окне **Lab1** необходимо нарисовать вентиль.

Используя средство **Polygon** панели инструментов **Drawin** сначала нарисуйте треугольник (рисунок 16) – для этого щелкните курсором по полю окна и, удерживая кнопку мыши, нарисуйте одну сторону треугольника (смена направления осуществляется щелчком мыши). Для создания треугольника достаточно нарисовать две стороны и дважды щелкнуть левой клавишей мыши.

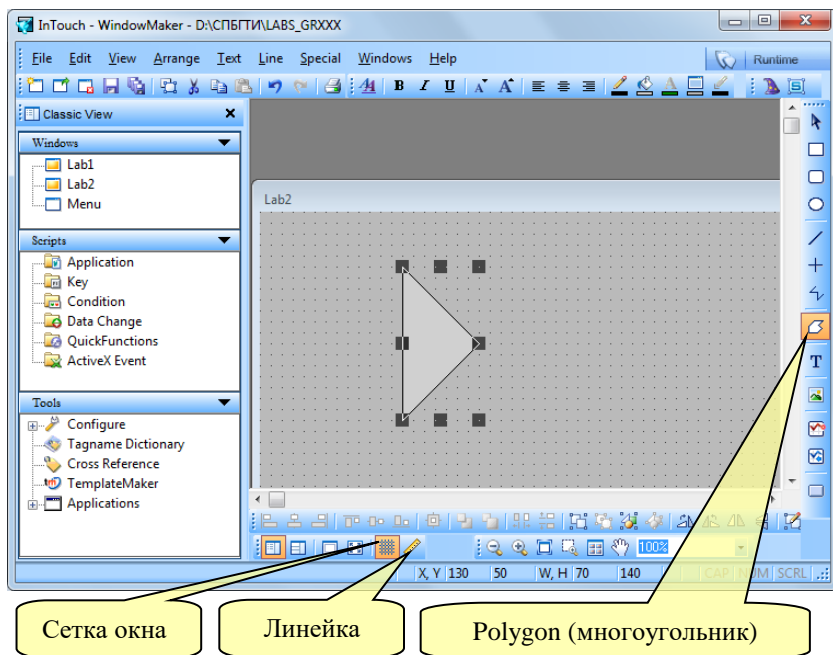


Рисунок 16



Для перехода от одного инструмента к другому в панели инструментов Drawing воспользуйтесь указателем перехода (объект со стрелкой).

**Внимание!** Для удобства рисования объектов в окне можно использовать привязку объектов к сетке окна, для этого в меню **Arrange** выберите пункт **Snap to Grid**.

Выделите треугольник и вызовите контекстное меню команд для операций над ним, щелкнув по нему правой клавишей мыши, в появившемся меню выберите пункт **Duplicate**. Данное действие позволит продублировать выделенный объект (рисунок 17), при этом копируются не только сам объект, но и все свойства и скрипты, связанные с этим объектом.

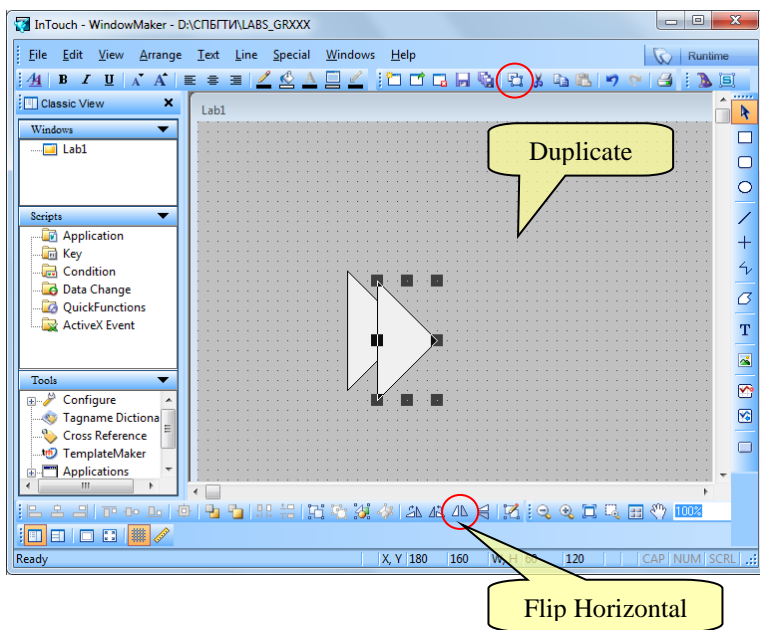


Рисунок 17

**9** Выделите второй треугольник (рисунок 17) и нажмите кнопку **Flip Horizontal** (отобразить объект по вертикали), расположенную на панели инструментов **Arrange**. Соедините два треугольника так чтобы получился вентиль, как показано на рисунке 18.

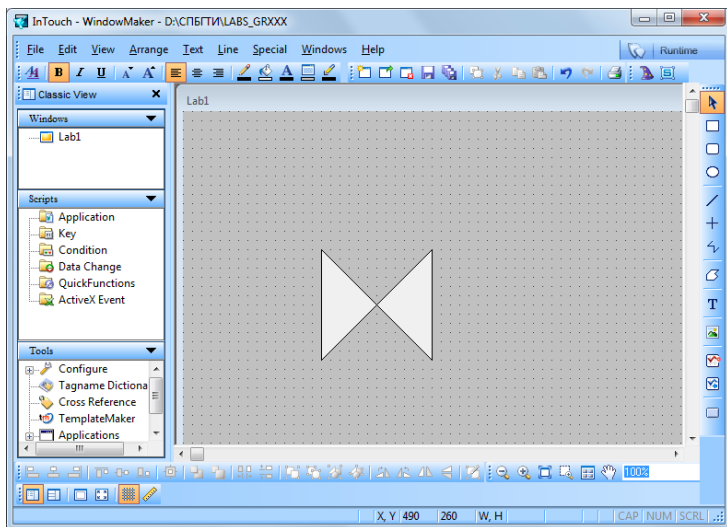


Рисунок 18

**10** Добавьте шток вентиля с помощью инструмента **Rectangle** панели **Drawing**, как показано на рисунке 19. Расположите его в фоновом слое командой **Send to Back** панели **Arrange**.

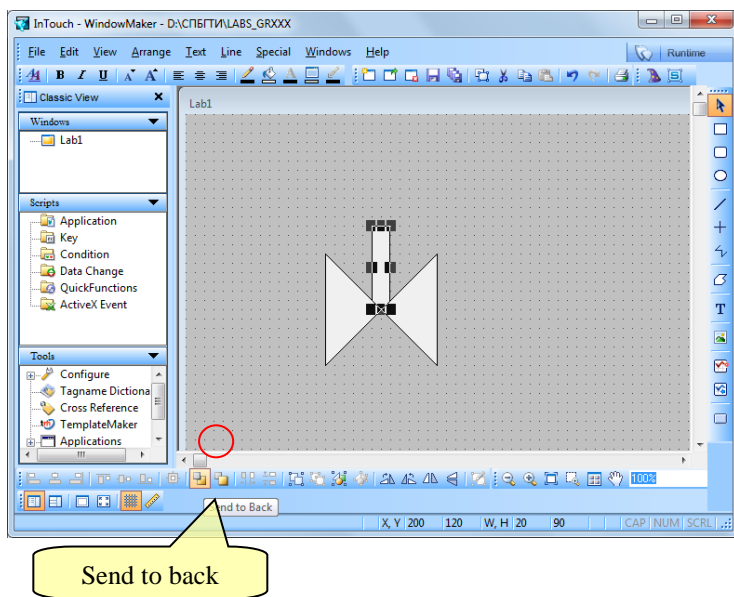


Рисунок 19

**11 Нарисуйте ручку вентиля с помощью инструмента **Ellipse** панели **Drawing**, как показано на рисунке 20.**

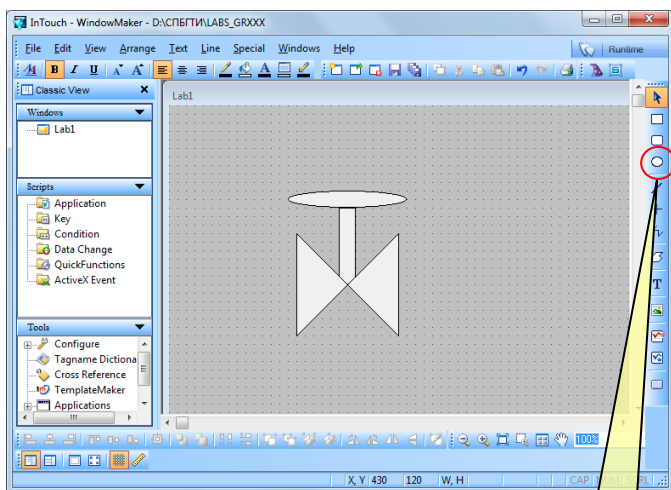


Рисунок 20

Ellipse

**12 Используя кнопку **Text** панели **Drawing**, добавьте под вентиля надпись **Вентиль 1** как показано на рисунке 21.**

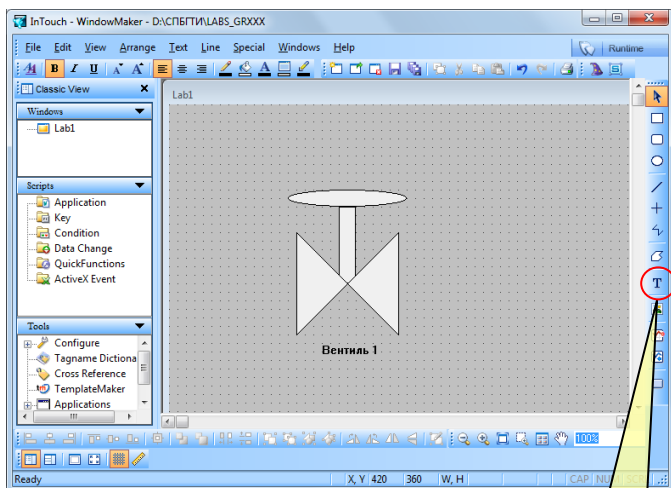


Рисунок 21

Text

**13** Таким образом, с помощью простых геометрических объектов был создан вентиль, состоящий из пяти отдельных элементов. Чтобы сделать из них единый объект, необходимо выделить все объединяемые элементы и нажать кнопку **Make Symbol** панели **Arrange**, как показано на рисунке 22.

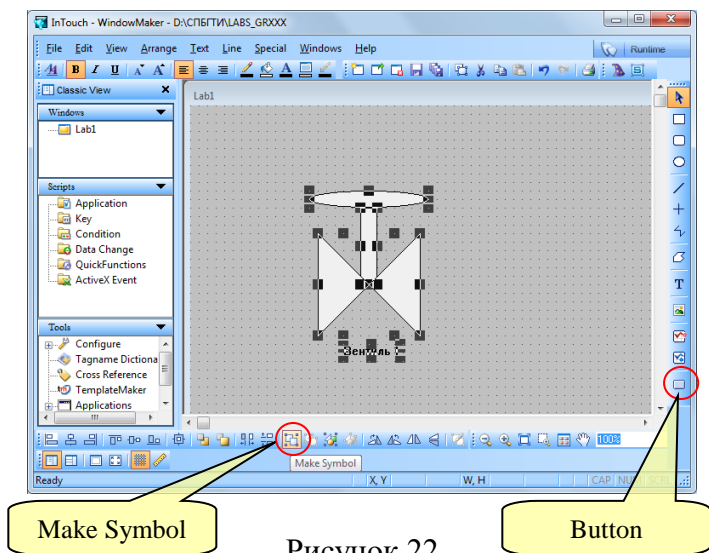


Рисунок 22

Используя панель инструментов **Format** у созданного объекта можно изменить цвет заполнения, толщину линий, а также цвет и тип шрифта надписи.

**14** Добавьте под вентилем кнопку **Button**. Для редактирования надписи кнопки выделите ее, затем выберите из системного меню **Special** → **Substitute**. На экране появится диалоговое окно **Substitute Strings**, в котором необходимо ввести **DiscTag1**, как показано на рисунке 23. Нажмите **OK**

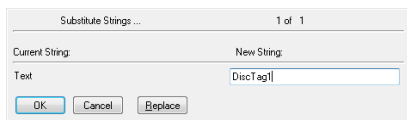


Рисунок 23

**15** Сохраните и закройте окно с именем Lab1.

\* \* \*

Разработка меню для навигации по выполняемым лабораторным работам, которое позволит переключаться между окнами Вашего приложения.

**16** Создайте новое окно с наименованием **Menu** со следующими координатами **X = 0** и **Y = 0**, **Width = 1024** и **Height = 72** и свойствами **Popup**, **Single**, сняв свойства **Title Bar** и **Size Control**. На рисунке 24 представлены настройки окна **Menu**.

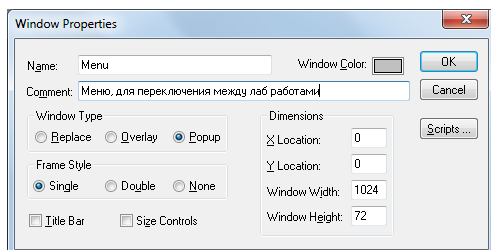


Рисунок 24

**Внимание!** По умолчанию настройки в этом диалоговом окне будут отражать настройки открытого или ранее созданного окна в WindowMaker. Если к активному окну привязан сценарий окна, появится сообщение с вопросом о том, хотите ли вы скопировать этот сценарий в новое окно.

В дальнейшем окно **Menu** будет постоянно открыто для обеспечения навигации по окнам операторского интерфейса. При этом необходимо рассчитать высоту окна таким образом, чтобы оно не перекрывало остальные окна. Поэтому для вновь создаваемых окон в следующих лабораторных работах устанавливайте координату **Y = 72**, чтобы окна **Lab1-Lab6** располагались под окном **Menu** в среде исполнения и не перекрывали его.

**17** В окне **Menu** с помощью инструмента **Button** создайте шесть кнопок с наименованиями: **Lab1 - Lab6**, т.е. по одной кнопке для каждой лабораторной работы, как показано на рисунке 25.

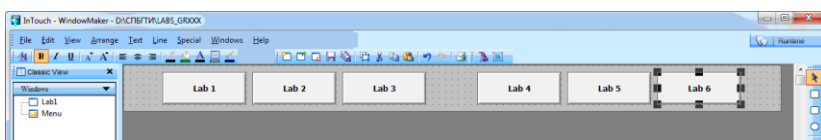


Рисунок 25

**18** Для настройки анимационных связей кнопки, то есть привязки к ней каких-либо действий, дважды щелкните по ней левой клавишей мыши или по правой кнопке мыши выберите из контекстного меню пункт **Animation Links**. На экране появится окно, представленное на рисунке 26.

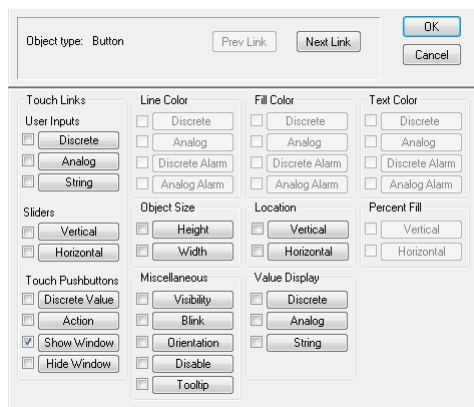


Рисунок 26

**19** В окне **Animation Links** (рисунок 26) выберите опцию **Show Windows**, щелкнув по соответствующей кнопке Show Windows левой клавишей мыши, на экране появится окно представленное на рисунке 27

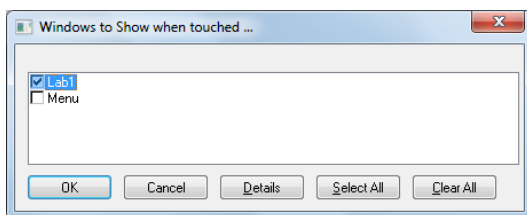


Рисунок 27

**20** В появившемся окне свяжите данную кнопку с окном **Lab1**, выбрав его из списка (рисунок 27). После выбора окна, которое будет открываться по кнопке нажмите кнопку OK.

В результате этих действий при нажатии в окне Menu кнопки Lab 1 будет открываться соответствующее окно.

## Wizards объекты

**21** Добавьте в окно Меню возможность отображения текущих времени и даты, для этого нажмите кнопку **Wizard**, представленную на рисунке 28.

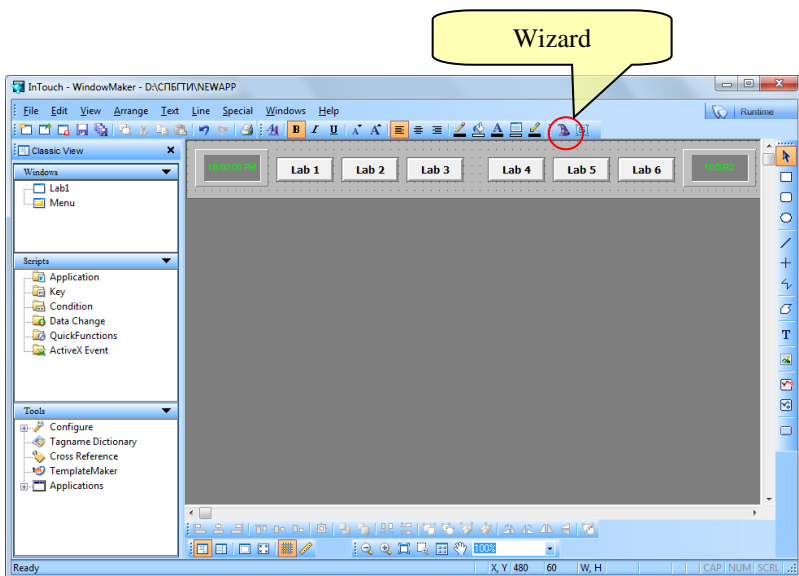


Рисунок 28

**22** На экране появится окно **Wizard Selection**, представленное на рисунке 29, в котором необходимо категорию **Clocks**, а затем выбрать объект **Digital Time/Date with Frame**. Нажмите кнопку OK.

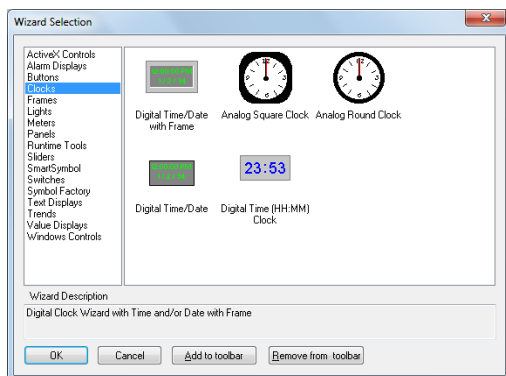


Рисунок 29

23 Расположите выбранный объект в левой части окна Menu. Двойным щелчком левой клавишей мыши вызовите окно Digital Clock Wizard, в котором необходимо выбрать пункт **Show Time** – показать время, как показано на рисунке 30

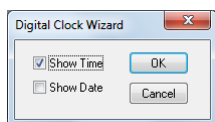


Рисунок 30

24 Создайте копию созданного объекта в пункте 23 и поместите его в правой части окна. Дважды щелкните на нем левой клавишей мыши и выберите **Show Date** – показать дату.

25 Вызовите окно с Wizards-объектами (рисунок 29). Выберите категорию **Symbol Factory** и нажмите кнопку **Add to toolbar**. После чего закройте окно Wizard Selection.

В результате на панели в среде разработки Window Maker появится кнопка **Symbol Factory**, как показано на рисунке 31.

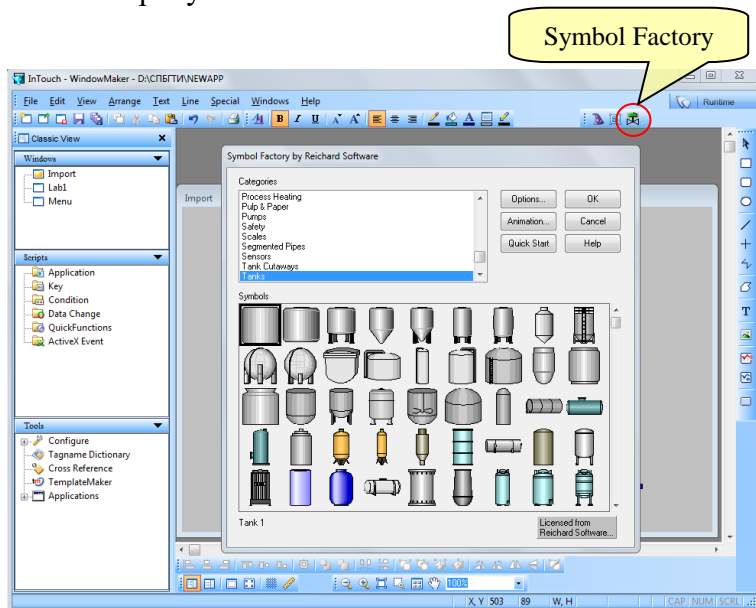


Рисунок 31



С помощью кнопки **Symbol Factory** организован быстрый доступ к дополнительной библиотеке с Wizards-объектами (рисунок 31). Ознакомьтесь с библиотекой **Symbol Factory**.

Создайте новое окно с именем **Wizard** и попробуйте, используя готовые элементы, составить упрощенную мнемосхему технологического процесса. Пример такой мнемосхемы представлен на рисунке 32.

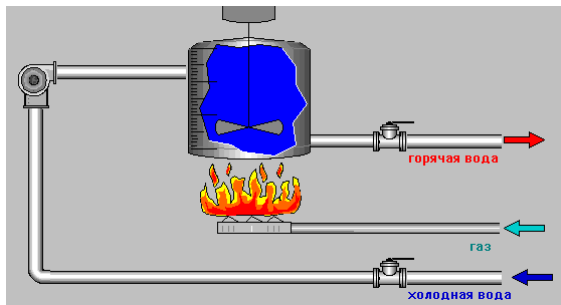


Рисунок 32

**26** Сохраните проект и оставьте на экране открытым только окно **Menu**.

Нажав на **Runtime!** в правом верхнем углу среды **Window Maker** перейдите в среду исполнения **Window Viewer**. Проверьте работоспособность Вашего приложения и вернитесь в среду разработки, нажав на **Development!** в правом верхнем углу среды исполнения.

\* \* \*

Импорт из другого проекта

**27** Закройте среду исполнения **Window Viewer** и все окна в среде разработки **Window Maker**.

**28** Выберите в меню **File → Import**. Следуя инструкциям в диалоговых окнах, укажите путь к приложению, из которого будете импортировать окна.

**Внимание!** В рамках данной лабораторной работы импорт осуществляется из текущего проекта. Поэтому для импортирования укажите путь к расположению Вашего проекта.

**29** На экране появится окно, представленное на

рисунке 33, в котором необходимо выбрать опцию Select Windows.

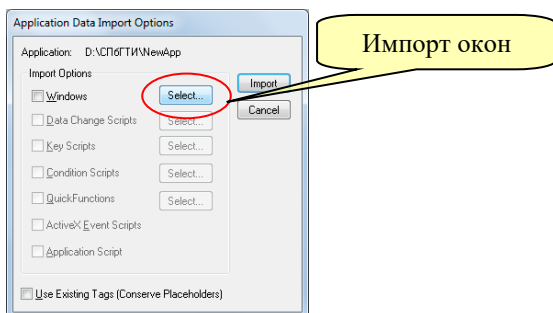


Рисунок 33

**30** Выберите окно с наименованием Lab1 для импорта. Нажмите кнопку Import.

Если при импортировании возникнет конфликтная ситуация, на экране появится окно, представленное на рисунке 34.

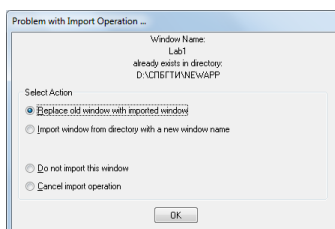


Рисунок 34

Импортируйте окно **Lab1** с новым именем **Lab1\_new**. Таким образом, в проекте появится еще одно окно **Lab1\_new**, идентичное окну **Lab1**. При этом все объекты в импортированном окне будут иметь связь фиктивными переменными, поэтому для работы скриптов и анимирования функций в импортированном окне необходимо преобразовать фиктивные переменные к действительным. Сейчас этого делать не нужно, так как импортированное окно не содержит такой информации.

**Результаты выполнения работы представляются преподавателю на дисплее в среде Viewer.**

## ЛАБОРАТОРНАЯ РАБОТА 2

### 2.1 Словарь переменных

В основе любого приложения разработанного в InTouch HMI лежит база данных технологических параметров, которая содержит информацию обо всех объявленных переменных. База данных представляет собой словарь переменных (**Tagname Dictionary**), или так называемых **тэгов**, представленный на рисунке 35. Именно переменные, связанные с объектами мнемосхемы, позволяют получить реальную динамическую картину объекта и в условиях технологического процесса отслеживать ситуацию на объекте в режиме реального времени. Таким образом, создание любого проекта целесообразно начинать не с разработки графических дисплеев (мнемосхем), а с базы данных.

The screenshot shows the 'Tagname Dictionary' window with the following details:

- Tab:** Details & Alarms (selected)
- Buttons:** New, Restore, Delete, Save, <<, Select..., >>, Cancel, Close
- Tagname:** IntTag
- Type:** Memory Integer
- Group:** \$System
- Read/Write:** Read/Write (selected)
- Comment:** (empty)
- Log Data:** (unchecked)
- Log Events:** (unchecked)
- Retentive Value:** (unchecked)
- Retentive Parameters:** (unchecked)
- Initial Value:** 250
- Min Value:** 0
- Deadband:** 0
- Eng Units:** (empty)
- Max Value:** 500
- Log Deadband:** 0
- ACK Model:** Condition (selected), Event Oriented, Expanded Summary
- Alarm Comment:** (empty)
- Alarm Settings Table:**

|  | Alarm Value | Priority | Alarm Inhibitor   |                               | Alarm Value | Priority         | Alarm Inhibitor | Value | Deadband |
|--|-------------|----------|---|-------------------------------|-------------|------------------|-----------------|-------|----------|
| <input type="checkbox"/> LoLo            | 0           | 1        |   | <input type="checkbox"/> High | 0           | 1                |                 |       | 0        |
| <input type="checkbox"/> Low             | 0           | 1        |   | <input type="checkbox"/> HIHi | 0           | 1                |                 |       |          |
| <input type="checkbox"/> Minor Deviation | 0           |          | 1   |                               |             |                  |                 |       |          |
| <input type="checkbox"/> Major Deviation | 0           |          | 1   |                               |             |                  |                 |       | 0        |
| <input type="checkbox"/> Rate of Change  | 0           | % per:   | <input type="radio"/> Sec <input checked="" type="radio"/> Min <input type="radio"/> Hr | Priority:                     | 1           | Alarm Inhibitor: |                 |       |          |

Рисунок 35

Основная информация, содержащаяся в словаре переменных, следующая:

- уникальное имя;
- группа, к которой относится данная переменная;
- тип переменной;

- значение при инициализации приложения;
- минимальное и максимальное значения переменной;
- предаварийные (Low и High) и аварийные (LoLo и HiHi) значения переменной;
- комментарий (служит для отображения и фиксации переменной в протоколах).

Неизменной составляющей словаря переменных являются системные теги, которые создаются автоматически при создании приложения и помечаются в словаре символом \$ в начале имени тэга. Системные теги обрабатываются приложением на определенные события, происходящие в системе. Системные теги могут использоваться в приложении для анимации объектов или сценариях.

При объявлении переменной в словаре переменных ей присваивается тип в соответствии с ее назначением. Тег **внешнего (I/O – Input/Output)** типа служит для получения информации о значении переменной из других приложений Windows или с/на сервера ввода/вывода. Для этого в **Tagname Dictionary** существует возможность настройки имени доступа **Access Name**, т.е. источника информации (например, контроллеров). **Внутренние (Memory)** теги хранятся внутри приложения InTouch HMI и используются для расчета параметров по математической модели, создания констант или анимации объектов.

Существуют следующие типы внутренних и внешних переменных:

- дискретный Discrete, который может принимать значения 0 или 1;
- целый Integer – 32-битное целое число;
- вещественный Real – с плавающей точкой;
- текстовый Message – текстовая строка до 131 символа.

По умолчанию все теги словаря переменных настроены на чтение и запись (опция **Read Write**), но для внешних переменных существует возможность настройки получения значения путем чтения переменной (опция **Read Only**).

В словаре переменных существуют дополнительные опции для создания и ведения архивных данных о состоянии переменных:

- для автоматической записи переменных в архивный файл, которые задействованы в суточных расчётах и в построении трендов, необходимо в словаре переменных отметить опцию **Log Data**;

- опция **Log Events** позволяет записывать и архивировать события в журнале алармов, при этом каждому событию необходимо определить приоритет;

- опция **Retentive Value** позволяет записывать последнее значение переменной в базе данных (при выходе из режима исполнения);

- опция **Retentive Parameters** позволяет записывать уставки аналоговых переменных.

В InTouch HMI реализованы функции алармов для оповещения пользователей о состоянии определенных процессов или системы. В словаре переменных представлены четыре уровня срабатывания сигнализаций: низкий (**Lo**), ниже нижнего (**LoLo**), верхний (**High**), выше верхнего (**HiHi**). Сигнализации нижнего (**Lo**) и верхнего (**High**) пределов срабатывания относятся к предупредительным сигнализациям, а сигнализации типа **LoLo** и **HiHi** – к аварийным и блокировочным сигнализациям. В связи с этим важно ввести понятие приоритета срабатывания сигнализации. Приоритет – это показатель важности сигнализации.

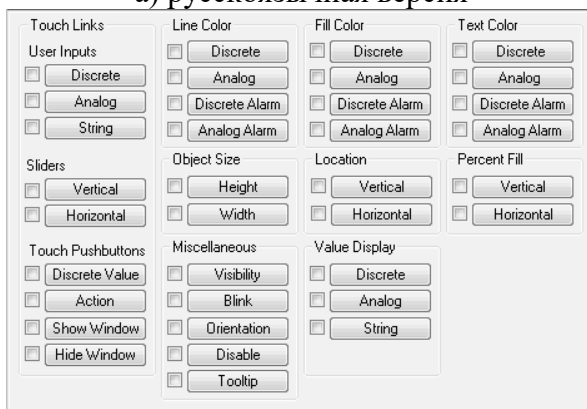
## 2.2 Анимационные связи

Созданный графический объект или символ может быть «оживлен» с помощью анимационных связей. Путем присоединения связей анимации можно заставить объект или символ изменять свой внешний вид при изменении значения тэга или выражения. Подобные специальные эффекты создаются путем определения связей анимации для объектов или символов.

Для объекта или символа можно определить несколько связей анимации. На рисунке 36 представлено окно настройки анимационных связей.



а) русскоязычная версия



б) англоязычная версия

Рисунок 36

InTouch поддерживает два основных типа связей: связи по нажатию и связи отображения. Связи по нажатию обеспечивают ввод со стороны оператора в систему. Связи

отображения обеспечивают вывод информации для оператора. Примерами связей по нажатию являются ползунки и кнопки. Эффекты цветовой заливки, положения или мерцания представляют собой связи отображения.

Путем сочетания различных связей на экране могут быть созданы практически любые анимационные эффекты, которые только можно вообразить. Можно заставить объекты изменять свой цвет, размер, положение, степень видимости, уровень заливки и т.д.

С помощью динамических связей, представленных в левом столбце диалога (**Touch Links** - связи по нажатию), осуществляется ввод информации в систему. Свойства **Touch Links** предоставляют оператору возможность открыть/закрыть клапан, запустить на выполнение скрипт, ввести новое значение переменной (задание), запустить распечатку отчета, перейти в другое окно и т. д.

Все остальные динамические связи предназначены для вывода информации на дисплей (**Display Links** - дисплейные связи).

**Связи по нажатию** используются для объектов или символов, которые должны быть "сенсорными" при выполнении приложения. Такие объекты позволяют оператору вводить информацию в систему. Например, оператор может включить или выключить клапан, задать новый пороговый уровень аларма, запустить сложный логический сценарий или процедуру входа в систему с использованием текстовых строк и т.д.

Связи по нажатию типа **Ввод пользователем** позволяют создать сенсорные объекты для ввода данных оператором в систему. Например, кнопки для изменения дискретных состояний, установки аналоговых значений или входа в систему.

Все переменные, имена которых будут использованы при конфигурировании динамических связей, должны быть определены в словаре переменных.

## 2.3 Цель работы

В результате выполнения работы необходимо:

- создание новых геометрических объектов;
- объявление переменных в словаре тегов;
- привязка анимационных связей к объектам;
- отображение значений переменной.

## 2.4 Выполнение работы

В результате выполнения лабораторной работы у Вас должно получиться окно со следующими объектами: вентиль из предыдущей работы, анимация которого происходит по нажатию соответствующей кнопки; шкала индикатора и круг, размер радиуса которого зависит от перемещения ползунка (т.е. значения переменной RealTag); транспортер с движущимися ступеньками.

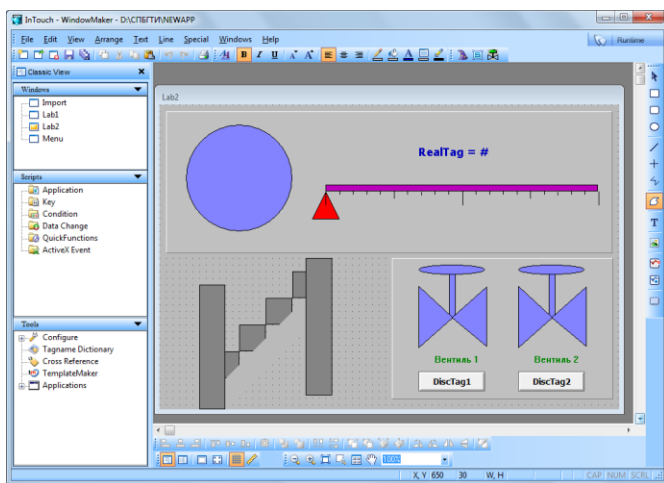


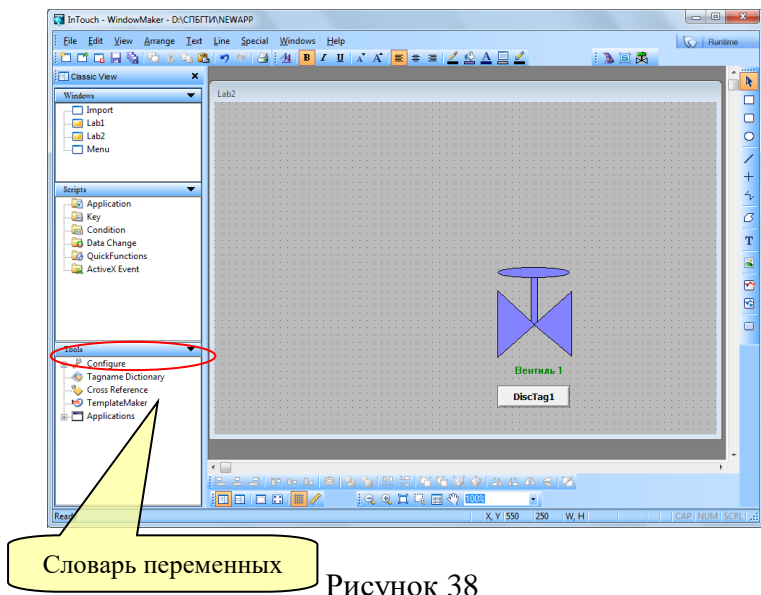
Рисунок 37

**Внимание!** Оформление лабораторной работы должно соответствовать современным эргономическим требованиям, при этом внешний вид и расположение объектов можно изменить по своему усмотрению.

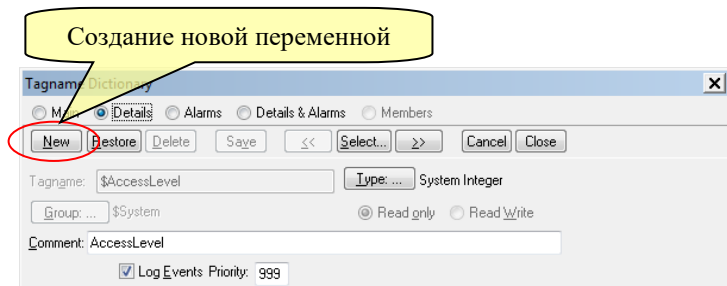


1 Создайте новое окно с наименованием **Lab2**, свойства которого соответствуют окну Lab1 из предыдущей лабораторной работы.

2 Скопируйте из окна **Lab1** клапан с дискретной кнопкой и разместите его в правом нижнем углу окна Lab2, как показано на рисунке 38.



3 Откройте **Tagname Dictionary** (словарь переменных) из меню **Special** → **Tagname Dictionary** или из панели **Tools** (рисунок 38). На экране появится окно **Tagname Dictionary**, представленное на рисунке 39.



#### 4 Создайте новые переменные с именами **DiscTag1** и **DiscTag2**.

В окне **Tagname Dictionary** нажмите кнопку **New** (рисунок 3) и заполните следующие поля: имя переменной **DiscTag1**, тип **Memory Discrete**, начальное значение **Off**, как показано на рисунке 40. После задания всех свойств переменной **DiscTag1** нажмите кнопку **Save**.

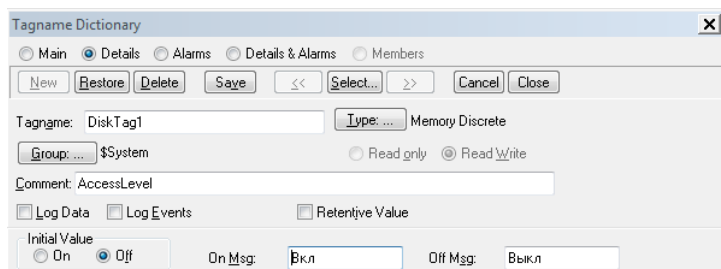


Рисунок 40

Создайте переменную **DiscTag2**, свойства которой подобны переменной **DiscTag1**.

5 Для привязки к объекту **Вентиль 1** анимационных функций вызовите меню настройки анимационных связей, дважды щелкнув левой клавишей мыши на объекте **Вентиль 1**. На экране появится окно **Object type**, представленное на рисунке 41.

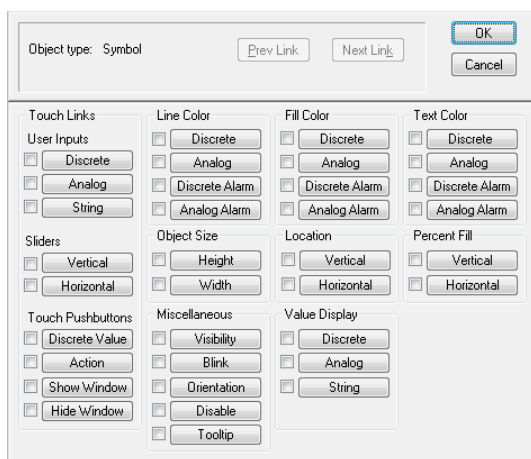


Рисунок 41

6 Поочередно установите динамические связи **Line Color**, **Fill Color** и **Text Color** для дискретной переменной **DiscTag1**. Все связи установите типа **Discrete**, что соответствует типу переменной **DiscTag1**, как показано на рисунке 42.

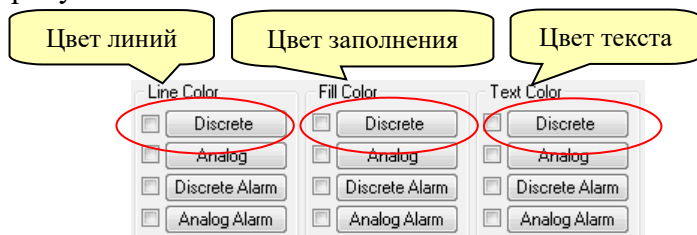


Рисунок 42

Для настройки анимационных связей с дискретной переменной **DiscTag1** необходимо у соответствующего свойства нажать кнопку **Discrete**. В соответствующих диалоговых окнах для **0, FALSE, Off** выберите красный цвет, а для **1, TRUE, On** – светло-зеленый, как показано на рисунке 43.

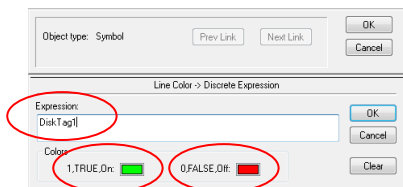


Рисунок 43

7 Для задания анимационных функций кнопки вентили **DiscTag1** дважды щелкните на ней левой клавишей мыши. На экране появится окно **Object type** (рисунок 41), в котором необходимо в категории **Touch Pushbutton** определить ее поведение как **Discrete Value**, т.е. как дискретное нажатие кнопки.

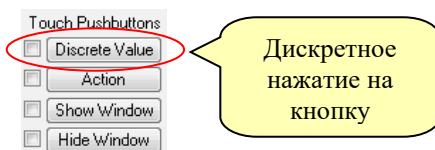


Рисунок 44

8 На экране появится окно, в котором необходимо связать нажатие кнопки **DiscTag1** с переменной **DiscTag1**, как показано на рисунке 45, а также задать тип нажатия на кнопку **Direct** (выполнять действие пока держим кнопку).

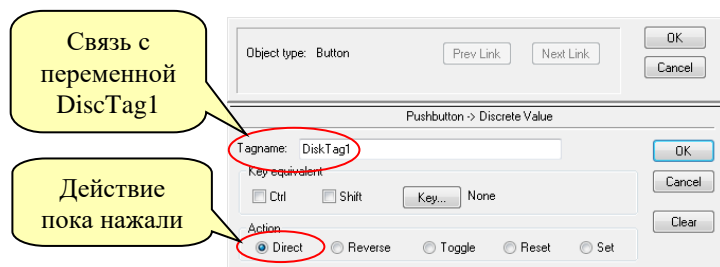


Рисунок 45

9 После анимирования объекта **Вентиль 1** с помощью функции **Duplicate** создайте его копию. Расположите объекты рядом, как это показано на рисунке 46. С помощью команды системного меню **Special → Substitute Strings** (Ctrl+L) отредактируйте надписи второго объекта – **Вентиль 2** и **DiscTag2**.

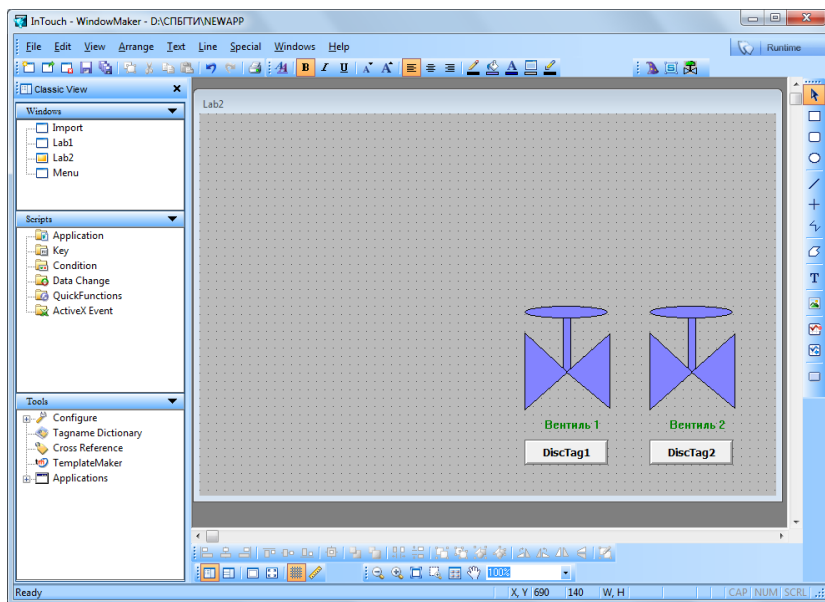


Рисунок 46

**10** Дважды щелкните на втором клапане для вызова меню анимационных связей. Поочередно установите динамические связи **Line Color**, **Fill Color** и **Text Color** для переменной с именем **DiscTag2** типа **Discrete**, как описано в пункте 5.

**11** При настройке анимационных связей для **0, FALSE, Off** выберите черный цвет, а для **1, TRUE, On** – синий, как показано на рисунке 47.

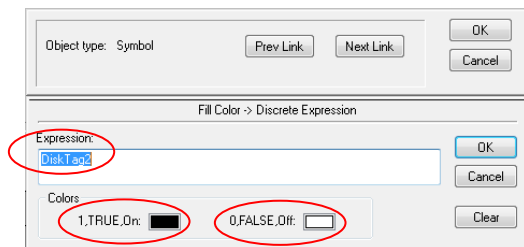


Рисунок 47

**12** Определите кнопку **DiscTag2** как дискретную пользовательского ввода, для этого дважды щелкните левой клавишей мыши на кнопке **DiscTag2** и в категории **User Input** выберите **Discrete**, как показано на рисунке 48.

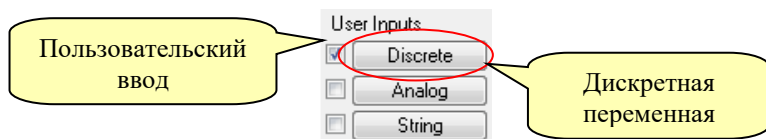


Рисунок 48

**13** На экране появится окно, в котором необходимо сделать привязку к тегу **DiscTag2**, как показано на рисунке 49.

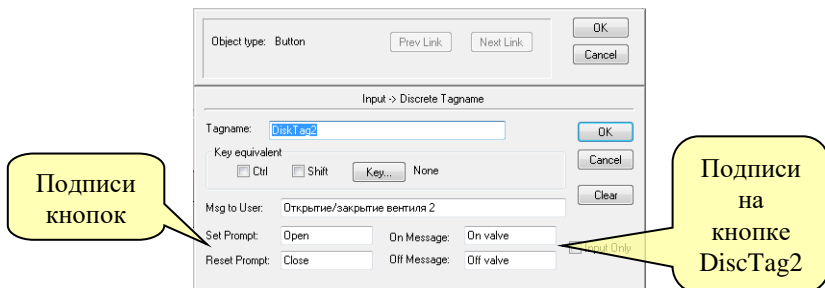


Рисунок 49

Задайте текст приглашений о состоянии вентиля в соответствующем диалоговом окне **Open** и **Closed**, а также сообщений отображаемых на кнопке **On Valve** и **Off Valve**; укажите в поле **Msg to User** комментарий **Открытие/Закрытие вентиля 2** (рисунок 49).

**14** Перейдите в среду исполнения **Window Viewer** и проверьте работу разработанного приложения.

**Внимание!** Вентили должны изменять цвет в зависимости от состояния дискретных переменных **DiscTag1** и **DiscTag2**, при этом первая кнопка **Замыкающего** типа, вторая **Дискретного пользовательского ввода**.

\* \* \*

Следующий этап работы – разработка анимированного ползунка, который можно перемещать относительно горизонтальной шкалы, отображение числового значения переменной связанной с положением ползунка и анимирование круга, радиус которого связано с этой переменной.

**15** В левой части окна с помощью инструментов **Rectangle** и **Polygon** нарисуйте прямоугольник (шкала) и треугольник (движок регулятора). Запомните ширину шкалы регулятора.

**16** В словаре переменных **Tagname Dictionary** объявите новую переменную (тег) **RealTag** типа **Memory Real**. В окне, представленном на рисунке 50, выберите опцию **Details** и определите начальное значение переменной **Initial Value = 250**, **Min Value = 0**, и **Max Value = 500**.

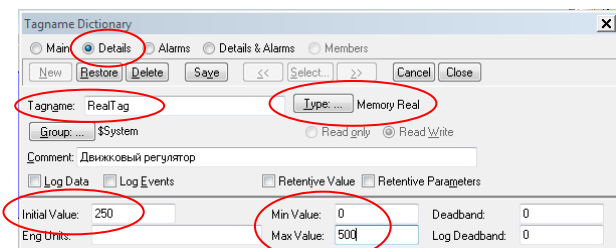


Рисунок 50

После определения свойств переменной необходимо записать ее в базу тэгов командой **Save**. Для выхода из словаря переменных нажмите **Close**.

**17** Анимируйте движок регулятора, для этого в окне анимационные связи (рисунок 41) определите его как **Horizontal Slider** (горизонтальный движок), нажатием на соответствующую кнопку, представленную на рисунке 51.

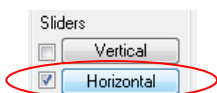


Рисунок 51

**18** В появившемся окне свяжите перемещение регулятора с переменной **RealTag**, определив следующие параметры: поля **At Left End = 0** и **At Right End = 500** (что соответствует минимальному и максимальному значениям переменной **RealTag**); поля **To Left = 0** и **To Right = 300** (что соответствует размеру прямоугольника или шкалы индикатора); **Reference Location = Left**.

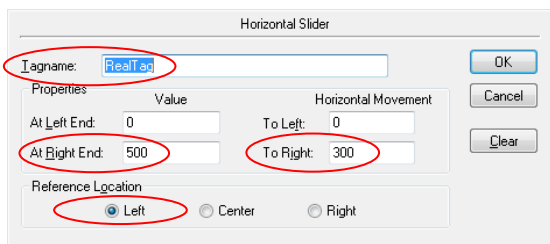


Рисунок 52

**19** Создайте текстовый объект для отображения значения переменной **RealTag**, как показано на рисунке 53.

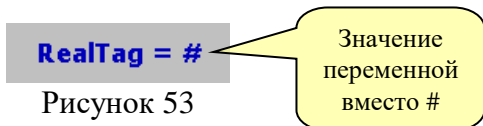


Рисунок 53

При этом в среде исполнения **Window Viewer** значение переменной будет отображено вместо значка решетки (#).

**20** Дважды щелкните левой клавишей мыши на текстовом объекте и выберите аналоговый вывод на экран,

как показано на рисунке 54. Далее свяжите текстовый объект с переменной RealTag.

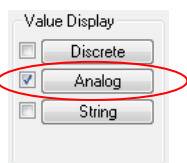


Рисунок 54

**21** Дважды кликните на прямоугольнике (шкала индикатор) и назначьте ему анимационную функцию **Horizontal Percent Fill**, как показано на рисунке 55.

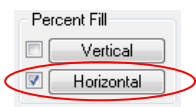


Рисунок 55

Свяжите свойство горизонтальное заполнение прямоугольника с переменной **RealTag**, указав для свойств **Value at Min Fill = 0** и **Value at Max Fill = 500** (минимальное и максимальные значения переменной RealTag) и выбрав цвет фона **Background Color** на свое усмотрение, как показано на рисунке 56.

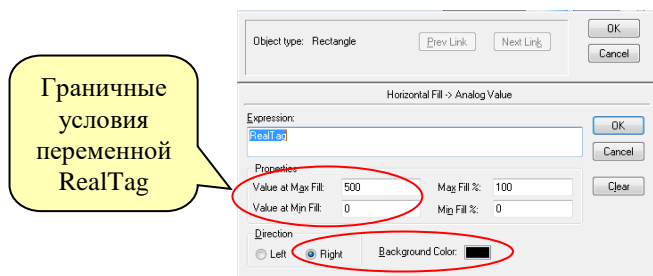


Рисунок 56

**22** Рядом со шкалой индикатора нарисуйте **круг**, радиус которого будет изменяться в зависимости от значения переменной RealTag.

**23** Определите кругу анимационное свойство изменение размера объекта по высоте и ширине, как показано на рисунке 57.



Свойства **Object Size Height** и **Object Size Width** свяжите с переменной RealTag таким образом, чтобы максимальная высота / ширина этих параметров соответствовала максимальному значению переменной, как показано на рисунке 58.

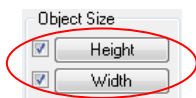


Рисунок 57

Граничные условия переменной RealTag

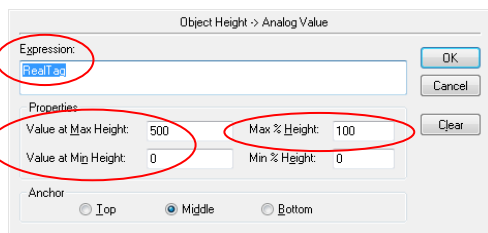


Рисунок 58

**24** Задайте для круга анимационное свойство **заполнение цветом** и свяжите его с аналоговой переменной RealTag, для этого в категории **Color Fill** выберите **Analog**, как показано на рисунке 59.

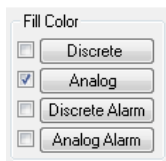


Рисунок 59

В окне определите диапазоны изменения цвета для переменной RealTag, пример представлен на рисунке 60.

Диапазоны изменения переменной RealTag



Рисунок 60

**25** Перейдите в среду исполнения и проверьте работу приложения.

\* \* \*

Разработка анимированного транспортера.

**26** С помощью инструмента **Rectangle** нарисуйте транспортер, который состоит из стоек, направляющей и **только одной** ступени, как показано на рисунке 61.

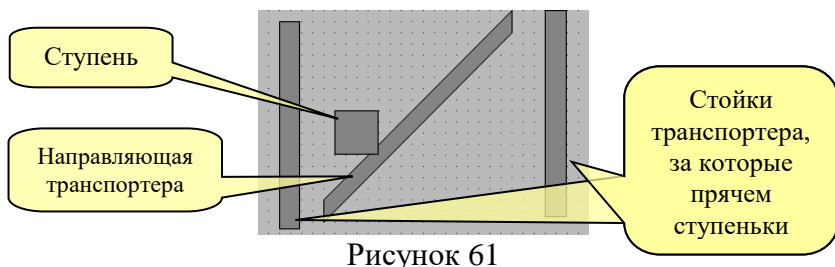


Рисунок 61

Задайте ступени анимационное свойство перемещение по вертикали **Location Vertical** со следующими значениями: **\$Second MOD 10**, **At Top Value = 10**, **At Bottom Value = 0**, **Up Vertical Movement = 20** и **Down Vertical Movement = 0**.

Затем определите анимационное свойство перемещение по горизонтали **Location Horizontal** со следующими значениями **\$Second MOD 10**, **At Left End Value = 0**, **At Right End Value = 10**, **To Left Horizontal Movement = 0**, **To Right Horizontal Movement = 20**.

Перейдите в среду исполнения и проверьте работу ступеньки, которая должна перемещаться циклически вправо и вверх.

**27** Продублируйте ступень и расположите объекты в виде транспортера (лестницы). Направляющую и стойки расположите поверх ступенек командой **Send To Front**.

**Для самостоятельной работы:** подберите такие значения для ступенек транспортера, чтобы создавалось впечатление непрерывности движения.

Перейдите в среду исполнения. Ступени транспортера должны перемещаться в соответствии с изменением значения системной переменной **\$Second**. Окончательно окно примет вид, как представлено на рисунке 37.

**28** Внесите изменения в окно **Menu**, для этого в окно добавьте текстовую надпись **#Mb**, как показано на рисунке 62.



Рисунок 62

**29** Назначьте текстовому объекту **# Mb** анимационную функцию аналогового вывода Analog из категории Value Display, которая возвращает каждую секунду число байт свободной дисковой памяти. Для этого необходимо в поле Expression ввести: **InfoDisk("C", 2, \$Second)/1000000**, как показано на рисунке 63.

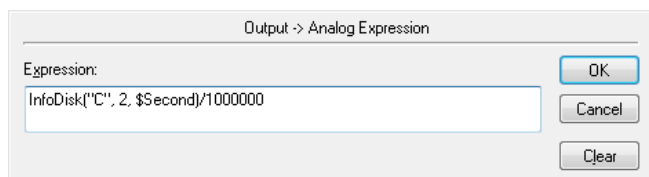


Рисунок 63

**30** Запустите Windows Viewer и проверьте работу вашего приложения.

## ЛАБОРАТОРНАЯ РАБОТА 3

### 3.1 Скрипты

Встроенные языки программирования – мощное средство SCADA-систем, предоставляющее разработчику гибкий инструмент для разработки сложных приложений. В современных версиях SCADA-систем функциональные возможности языков становятся существенно богаче. Явно выделяются два подхода:

- ориентация встроенных языков программирования на технологов. Функции в таких языках являются высокоуровневыми, не требующими профессиональных навыков программирования при их использовании;
- ориентация на системного интегратора (обычно это VBasic - подобные языки).

В разрабатываемом приложении создаются программные фрагменты, состоящие из операторов и функций языка, которые выполняют некоторую последовательность действий. Эти программные фрагменты связываются с разнообразными событиями в приложении, такими как нажатие кнопки, открытие окна, выполнение логического условия ( $a + b > c$ ). Каждое из событий ассоциируется с графическим объектом, окном, таймером, открытием/закрытием приложения. Когда приложение содержит сотни окон, тысячи различных графических объектов, а с каждым из них связано несколько событий, в приложении может "работать" огромное количество отдельных программных фрагментов. **Внимание!** При их разработки велика вероятность их "одновременной" активизации.

Скрипты в InTouch - это программные фрагменты, которые позволяют исполнять определенные последовательности команд и логических операций после выполнения некоторых событий. Событием может быть изменение данных, соблюдение условия, нажатие клавиши мыши, изменение значения переменной, открытие окна,

активизация таймера и т.д. Порядок выполнения скриптов зависит от приложения.

Основные типы скриптов вызываются из специальной панели Scripts, представленной на рисунке 64.

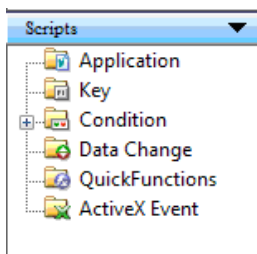


Рисунок 64

Существующие типы скриптов InTouch HMI:

- **Application Scripts** – скрипты, относящиеся ко всему приложению и используемые для запуска других приложений, имитации технологических процессов, вычисления значений переменных и т.д. Могут исполняться один раз в момент запуска или завершения приложения или периодически через заданные интервалы времени;

- **Window Scripts** – скрипты, связанные с конкретным окном. Исполняются один раз в момент открытия или закрытия окна или периодически через заданные интервалы времени пока открыто окно;

- **Key Scripts** – клавишные скрипты, которые привязываются к какой-либо клавише или комбинации клавиш клавиатуры. Скрипты могут исполняться один раз в момент нажатия на клавишу или ее отпускания, а также периодически пока нажата клавиша;

- **Touch Pushbutton Action Scripts** – скрипты, которые связаны с исполнительными кнопками. Эти скрипты запускаются при каждом нажатии на объект-кнопку;

- **Condition Scripts** – скрипты по изменению логического выражения связываются с логической переменной или выражением, которое будет принимать значения либо «истина», либо «ложь»;

- **Data Change Scripts** - скрипты по изменению данных связаны либо с переменной, либо с полем переменной. Эти скрипты исполняются только один раз, когда значение переменной либо поля меняется на величину, превышающую значение допуска, заданного в словаре переменных;

- **ActiveX Event** – скрипты событий, запускающиеся в **WindowViewer** во время исполнения приложения.

- **Quick Function** – скрипты, которые могут вызываться из других скриптов и использоваться в выражениях при определении динамических свойств объектов.

Диалоги редактора, открываемые при создании скриптов различных типов, имеют небольшие отличия. Вызов диалога редактора скриптов в окне WindowMaker осуществляется командой Special/Scripts с последующим выбором типа создаваемого или редактируемого скрипта.

В пакете InTouch имеется набор встроенных функций, которые могут быть связаны с командами или использованы в скриптах для выполнения самых различных задач. Все встроенные функции разбиты на четыре группы:

- String... – для обработки различных символьных строк и переменных;

- Math... – математические функции;

- System... – системные функции;

- Misc... – функции для работы с алармами распределенных систем, трендами, печатью и др.

InTouch HMI использует встроенный язык программирования Cicode системы Citect, созданный специально для мониторинга и управления приложениями. Это структурированный язык, похожий на Visual Basic или С. Применение Cicode предоставляет пользователю доступ к данным проекта в режиме реального времени, а также ко всем переменным, алармам, трендам, отчетам и т. д. Cicode поддерживает многозадачность и удаленный вызов процедур.

### 3.2 Цель работы

В результате выполнения работы необходимо:

- разработка мнемосхемы технологического процесса;
- разработка скриптов для пересчета параметров процесса;
- разработка скриптов для анимирования геометрических объектов.

### 3.3 Выполнение работы

В результате выполнения лабораторной работы получится окно со следующими объектами:

- бункер, внутри которого расположен элемент для отображения текущего уровня жидкости;
- транспортер, который включается нажатием на кнопку, при этом происходит заполнение бункера;
- синий указатель для отображения текущего значения уровня;
- красный указатель для задания максимального значения уровня;
- крышка бункера, которая открывается при достижении максимального значения уровня, происходит слив жидкости (т.е. текущий уровень уменьшается).

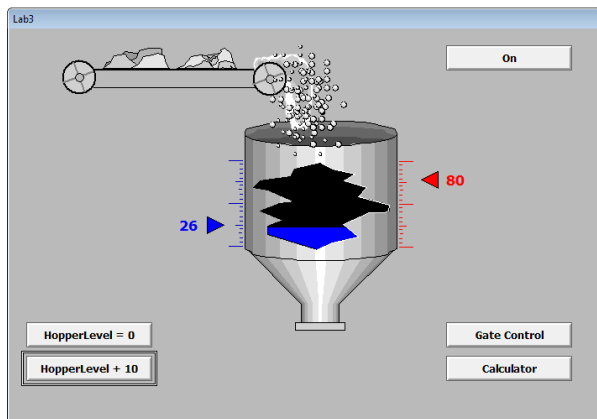


Рисунок 65

1 Создайте новое окно с наименованием **Lab3**.

2 Определите в **Tagname Dictionary** следующие переменные:

- **Conveyor\_on:**  
Initial Value – Off, Type - Memory Discrete;
- **Gate:**  
Initial Value – Off, Type - Memory Discrete;
- **GateCount:**  
Initial Value – 0, Memory Integer, 0 - 30;
- **HopperLevel:**  
Initial Value – 0, Type - Memory Real, 0 - 100;
- **HopperSetpoint:**  
Initial Value – 80, Type - Memory Real, 0 - 100;

3 Используя стандартные геометрические объекты и объекты библиотеки Wizards, разработайте мнемосхему технологического процесса.

3.1 Бункер высотой 200 пикселей.

3.2 Внутри бункера расположен объект для отображения текущего уровня жидкости, высотой 100. Присвойте объекту свойство **Percent Fill** с заполнением по вертикали **Vertical**. Привяжите объект к переменной **HopperLevel**, значение которой изменяется от 0 до 100. Укажите направление заполнения бункера **Direction**, а также цвет фона **Background Color**.

**Внимание!** Цвет объекта в Window Maker и цвет фона не должны совпадать.

3.3 Красный указатель – это максимальный уровень жидкости в бункере, значение которого выставляется оператором. Анимлируйте указатель с помощью анимационного свойства **Vertical Slider**, связав его с переменной **HopperSetpoint**. Определите границы изменения переменной **At Top** (0) и **At Bottom** (100), а также границы перемещения самого движка **Up** (0) и **Down** (100). Таким образом, удастся связать между собой переменную **HopperSetpoint** и перемещение движка.



Рядом с красным указателем расположен текстовый объект привязанный к аналоговой (**Analog**) переменной **HopperSetpoint** анимационными свойствами **Value Display** для отображения значения переменной и **Vertical Location** для перемещения текстовой строки вместе с указателем.

**3.4** Синий указатель – это текущий уровень жидкости в бункере. Анимлируйте указатель с помощью анимационного свойства **Vertical Location**, связав его с переменной **HopperLevel**. Определите границы изменения переменной **At Top** (0) и **At Bottom** (100), а также границы перемещения самого движка **Up** (0) и **Down** (100).

Рядом с указателем анимируйте текстовый объект для отображения значения переменной **HopperLevel**.

**3.5** Вращение колес анимируется привязкой свойства **Orientation** группы **Miscellaneous** к системной переменной **\$Second**.

Оси колес видны только при включенном конвейере, состояние которого отображает дискретная переменная **Conveyor\_on**. Для этого анимационному свойству **Visibility** группы **Miscellaneous** необходимо задать условие отображения **Conveyor\_on** = = 1 (если включен).

**3.6** С помощью свойства **Visibility** анимируйте вещество, которое поступает с конвейера в бункер.

**4** Задайте кнопке **Conveyor\_on Toggle** анимационное свойство **Discrete** пользовательского ввода **User Inputs** с привязкой к дискретной переменной **Conveyor\_on**.

**5** Перейдите в среду исполнения и проверьте работу анимационных функций.

\* \* \*

Разработка скриптов

**6** Анимирование кнопки **HopperLevel** = 0, нажатие которой приводит к обнулению переменной **HopperLevel** (т.е. к сбросу уровня в бункере).

Задайте кнопке **HopperLevel = 0** анимационное свойство **Action** категории **Touch Pushbutton**. В результате на экране появится окно создания скрипта, представленное на рисунке 66. В окне выберите действие **Key Down** (нажатие на кнопку) и напишите сценарий для обнуления уровня: **HopperLevel=0;** (рисунок 66).

Проверка синтаксиса сценария осуществляется нажатием кнопки **Convert**. Сохраните сценарий и закройте окно нажатием на кнопку **OK**.

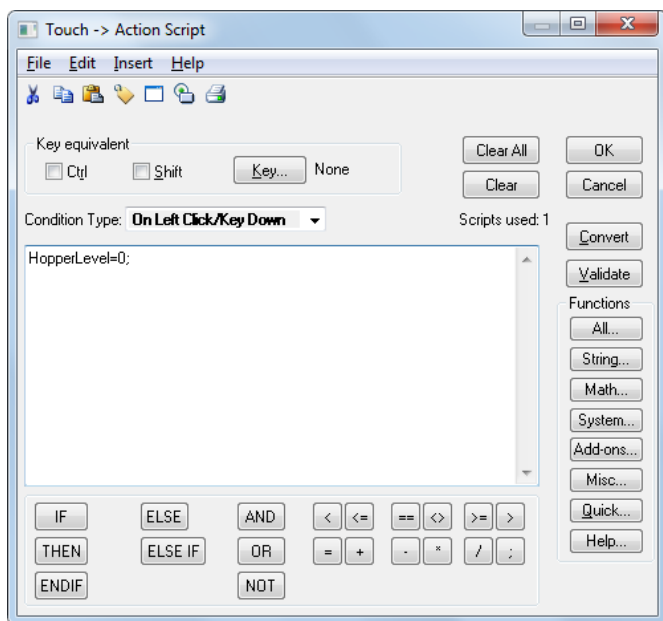


Рисунок 66

**7** Создайте для кнопки **HopperLevel + 10** скрипт **Action** типа **Key Down**, который выполняется по однократному нажатию левой клавишей мыши на кнопку, как показано на рисунке 67. Однократное нажатие кнопки приводит к увеличению переменной **HopperLevel**, при этом необходимо предусмотреть чтобы значение переменной было не больше верхнего граничного значения.

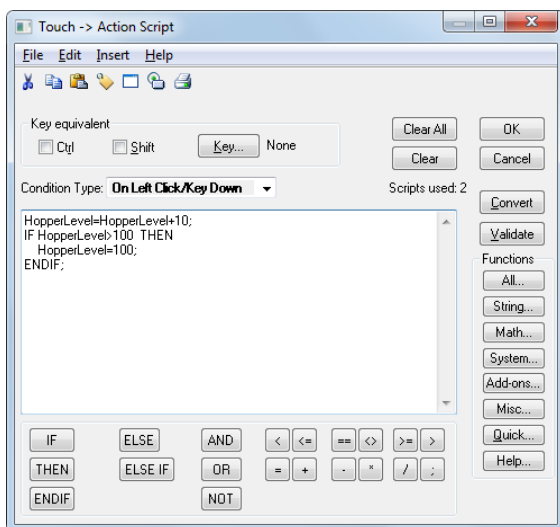


Рисунок 67

Разработайте для кнопки **HopperLevel + 10** второй скрипт, который будет выполняться по удержанию **While Down** на ней левой клавишей мыши, как показано на рисунке 68.

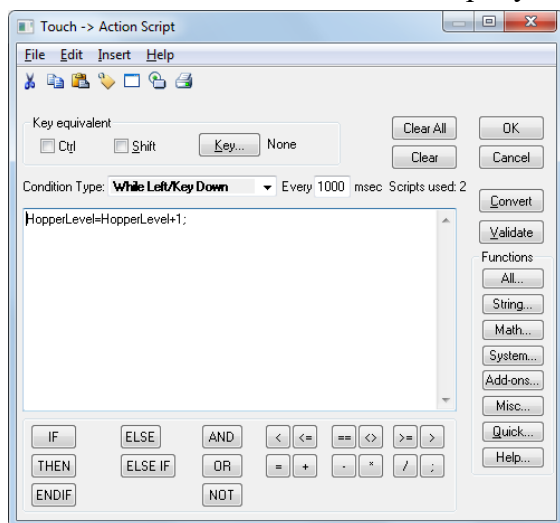


Рисунок 68

**8** Перейдите в среду исполнения и проверьте работу кнопок.

**9** Заполнение бункера осуществляется при включенном конвейере (**Conveyor\_on == 1**) с помощью скрипта **Condition**, выполняемого по условию.

В окне **Scripts** выберите пункт **Condition**, щелкнув правой клавишей мыши по соответствующей строке, выберите New. На экране появится окно создания скрипта по условию, как показано на рисунке 69.

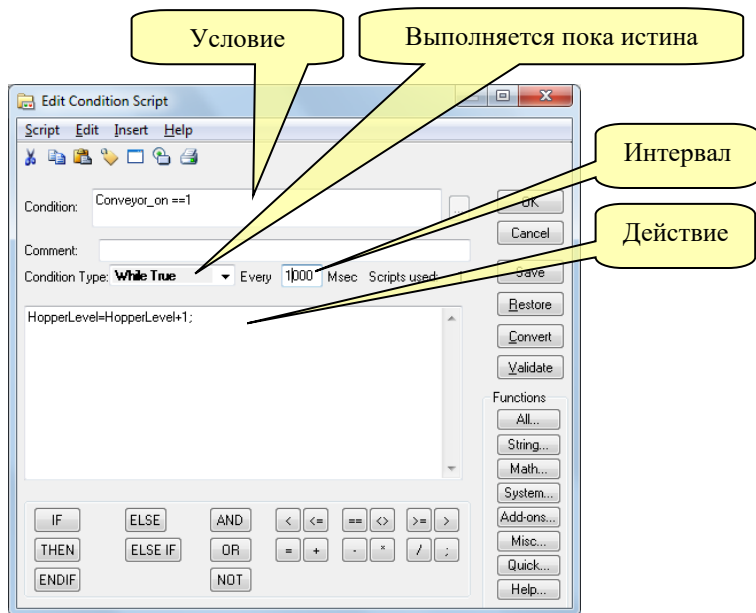


Рисунок 69

**10** Согласно предыдущему скрипту, пока включен конвейер происходит заполнение бункера. При этом необходимо учесть, что текущий уровень **HopperLevel** не должен превышать максимально заданный **HopperSetpoint**.

**Самостоятельно!** Разработайте скрипт по условию, который бы не допускал переполнение бункера.

**11** Перейдите в среду исполнения и проверьте работу разработанных скриптов.

**Внимательно проверяйте работу скриптов, неправильное сочетание условий может привести к разгласованию всего приложения.**

12 Нажатие на кнопку **GateCount** приводит к открытию/закрытию крышки бункера. Для этого вручную разработан скрипт **Action** по нажатию **Touch Pushbuttons** кнопки **GateCount**, как показано на рисунке 70.

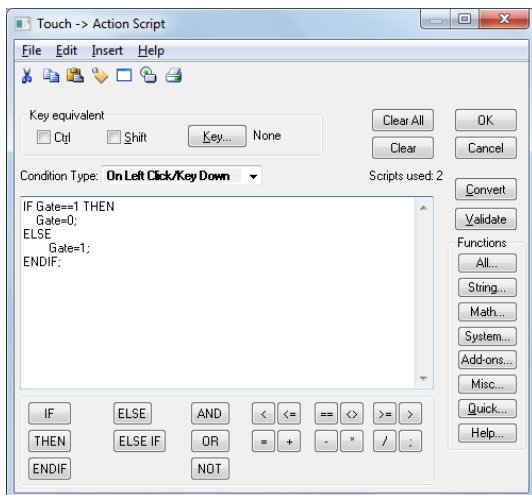


Рисунок 70

Также крышка бункера будет открываться при достижении текущего уровня **HopperLevel** максимального значения **HopperSetpoint**. Для этого разработан скрипт **Condition**, представленный на рисунке 71.

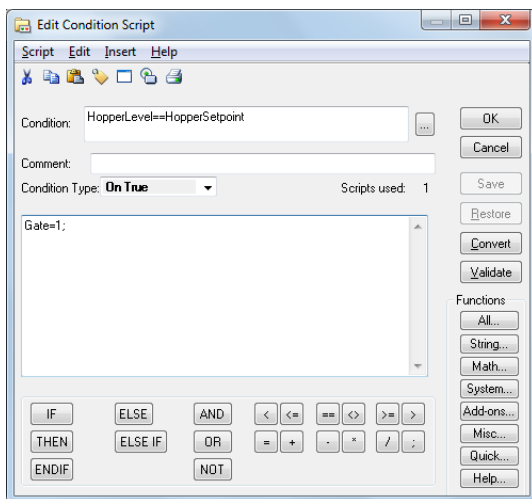


Рисунок 71

**13** Перемещение крышки бункера осуществляется привязкой переменной **GateCount** к анимационному свойству **Horizontal Location** (граничные значения переменной указываются для свойств At Left End = 0, At Right End = 30).

Создайте скрипт, который при соблюдении определенных условий будет постепенно изменять значение переменной **GateCount**. Выполнение скрипта привяжите к окну Lab3. Для этого из всплывающего меню окна Lab3, вызываемого по правой кнопке мыши, выберете пункт **Window Scripts**, как показано на рисунке 72.

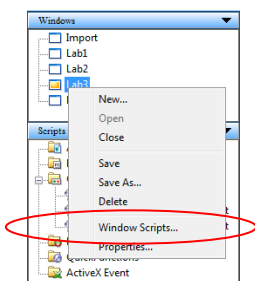


Рисунок 72

Создайте скрипт для перемещения крышки бункера, который выполняется пока активно окно **Lab3** с помощью свойства **While Showing**, как показано на рисунке 73.

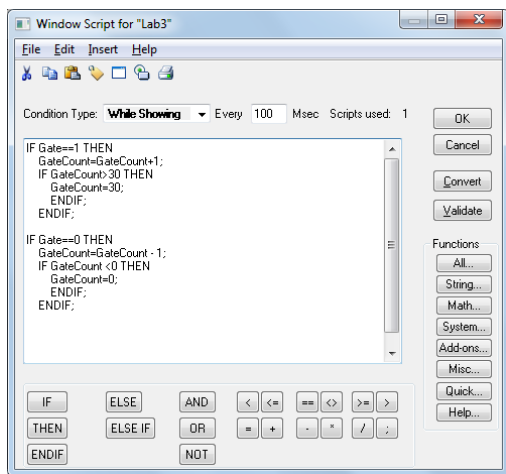


Рисунок 73

**14** При открытии окна Lab3 создайте скрипт типа **On Show** (когда открыли окно) для обнуления переменных **Gate**, **GateCount**, а также текущее значение уровня в бункере **HopperLevel**, как показано на рисунке 74.

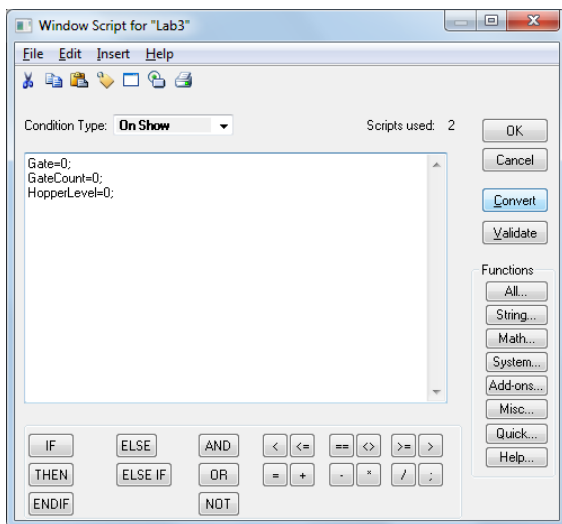


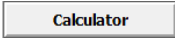
Рисунок 74

**15 Самостоятельно!** Разработайте скрипт для уменьшения значения текущего уровня в бункере **HopperLevel**, когда крышка бункера **Gate = 1** открыта. Крышка бункера открывается при достижении текущего уровня **HopperLevel** максимального значения **HopperSepoint**, а также при открытии крышки бункера нажатием кнопки **Gate Control**.

**HopperLevel=HopperLevel-1**

**16 Самостоятельно!** Подправьте скрипт, который позволяет при удержании кнопки **HopperLevel + 10** постепенно увеличивать значение **HopperLevel**, так как его выполнение может привести к ситуации когда значение **HopperLevel > HopperSepoint**.

**17** Перейдите в среду исполнения **Window Viewer** и проверьте работу приложения. Протестируйте работу скриптов для различных ситуаций.

**18** Создайте скрипт **Touch Pushbutton Action** типа **On Key Down** для кнопки , при нажатии на которую вызывается программа **Calculator**, как показано на рисунке 75.

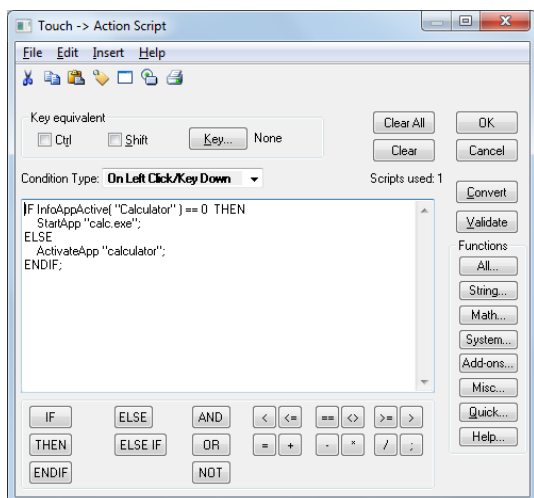


Рисунок 75

Функция **InfoAppActive** используется для проверки активности программы **Calculator**. Если приложение **Calculator** не активно, то скрипт запустит его; иначе скрипт будет работать в фоновом режиме.

**19** Свяжите окно **Lab3** с кнопкой **Lab3 окна Menu**, для того чтобы Ваше меню отображало данное окно на экране при нажатии на неё.

**20** Запустите **Windows Viewer** и проверьте работу вашего приложения.



## ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

- 1 Назовите основные компоненты InTouch HMI?
- 2 Для чего служат среды Window Maker и Window Viewer?
- 3 Какие типы объектов существуют в InTouch HMI?
- 4 Какие свойства окон есть в InTouch HMI?
- 5 Что такое Wizards-объекты?
- 6 Что такое словарь переменных и для чего он необходим?
- 7 Что такое тег?
- 8 Какие типы тегов существуют?
- 9 Какие типы тегов использовались в лабораторных работах?
- 10 Что такое анимационные связи объекта InTouch HMI?
- 11 Типы анимационных связей?
- 12 Какие типы анимационных связей использовались в лабораторных работах?
- 13 Что такое скрипты InTouch HMI?
- 14 В каких анимационных связях используются скрипты?
- 15 Типы кнопок в InTouch HMI?
- 16 Типы скриптов в InTouch HMI?

## СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1 Wonderware FactorySuite 2002. InTouch. Руководство пользователя. М.: Klinkmann, 2000. – 466с.

2 SCADA-система InTouch 10.0 фирмы Wonderwar: Методическое пособие по учебному курсу. М.: НОУ «Учебный центр РТСофт», 2012. – 191с.

3 Ершова О.В. Изучение SCADA – системы InTouch, часть 1: Метод. указания. СПб.: СПбГТИ(ТУ), 2005. – 42с.

4 Ершова О.В. Изучение SCADA – системы InTouch, часть 2: Метод. указания. СПб.: СПбГТИ(ТУ), 2005. – 40с.

5 Официальный сайт фирмы Wonderware Russia [Электронный ресурс]: программные решения для успешного управления в режиме реального времени. URL: <http://www.wonderware.ru> (дата доступа 06.10.2017).

6 Официальный сайт фирмы Klinkmann [Электронный ресурс]: официальный авторизованный дистрибьютор Wonderware разработчика самого популярного промышленного программного обеспечения в мире (системы АСУ ТП, SCADA, MES, решения для управления качеством, анализа эффективности производства и др.). URL: <http://www.klinkmann.ru> (дата доступа 06.10.2017).