WellinTech Product Training

Luna Kan

lunakyh@gmail.com
• Section 1 – Overview Features
• Section 2 – Create a new project
  - To know KingSCADA
  - Create a solution
  - Create a new project
• Section 3 – Connect with I/O device (IOServer)
  - Overview
  - Features
  - Build a project
  - Connect KingSCADA and IOServer
  - Create tags (data model)
• Section 4 – Graphic System
  – Graphy Editor (Development Environment)
  – Scripting
  – Animation
  – Extension elements (X-Y Curve, Trend curves, Alarm Windows, etc.)
  – Zoom in development environment and run time

• Section 5 – Genius
  – Genius feature
  – How to make genius

• Section 6 – Graphic Model

• Section 7 – History and Alarm
- Section 9 – Report
- Section 10 – Redundancy
  - Redundancy solution
  - Redundancy configuration
- Section 11 – Network
- Section 12 – Web Publishing
- Section 13 – Security and User management
• Section 1 – Overview Features
  - Training objectives
  - General introduction
  - Features
1. **To know KingSCADA (Introduction)**
   SCADA software that normally applies to the field, on-site monitoring of local instruments and devices, while providing centralized control for total solution.

2. **To use KingSCADA (Application)**
   KingSCADA is easy-to-learn and easy-to-use, to apply KingSCADA have several steps to make an available project to supervise and control the devices on-site.

3. **To compare KingSCADA and other SCADA software (Comparison)**
   KingSCADA has its own features better than others, a comparison is useful for us to know the differences of the popular SCADA software on the market.
What is SCADA?

SCADA stands for *Supervisory Control And Data Acquisition*. It features:

- Remote project deployment, monitoring, debugging and diagnosis via Internet
- High-speed data acquisition
- Real-time online monitoring
- Easy operation, information integration
- Reliable, robust
KingSCADA Perspectives

- Better presenting of the field – advanced graphical development technology
- Faster data acquisition – independent block data collection, rapid and efficient
- More convenient project development – model creation, high component reusability
- Simpler debug and maintenance – easy-to-learn and easy-to-use, modulated functions
- More robust system configuration – complete redundancy solution, ensure data integrity
- More flexible network deployment – Flexible network architecture, scalable
- More open information platform – Open interfaces for third-party access and integration into enterprise information portal
KingSCADA 3.0 is a revolutionary HMI/SCADA product aiming at middle to high end market. It features **vivid** graphics, highly reusable graphics model and data model, block-based fast data acquisition, **advanced redundant technology**, and **powerful web publishing** that leads to higher productivity, reliability, and scalability.

The rich open APIs and convenient web portal support make it easier for enterprises to build integrated intelligent information platform.
KingSCADA Features

Applications

IOServer

KingSCADA

Key Features

- Powerful graphics animation & beautiful true-color display
- Construct graphics model & data model, highly reusable
- Block-based data collection & supports redundancy
- Scalable network architecture
- Powerful web publishing
KingSCADA: Resourceful IDE

- Visualized graphics development environment
- Convenient toolkits
- Object-based development method
- Rich drawing objects
- Various configuration windows:
  - Property Window
  - Object Window
  - Animation Window
  - Content Window

Features

- Layout Toolbar
- Insert image
- Pen brush setting window
- Property Window
- Object Window
- Animation Window
- Content Window

2010-11-18 WellinTech Product Training — Yuehong Kan
KingSCADA: Sophisticated Graphics Technology
KingSCADA: Sophisticated Graphics Development
KingSCADA: Model Construction and Reuse

- Revolutionary new concept
- Build model, unlimited reuse
- Modification propagation
- Customized standard, consistent, accumulate enterprise resources
KingSCADA: Modulated Functions, Independent Data Collection

- Online/Offline Editing
- Provide drivers for 3500 popular hardware
- Efficient data acquisition
- Reliable
- Performance monitoring
KingSCADA: Complete Redundancy

Data Redundancy
- Real-time data
- Alarm data
- Historical data
- IOServer redundancy

Data Integrity
- Real-time data synchronization
- Data buffering for recovery
KingSCADA: Reliable Redundancy, Rapid Switch

- Redundant status detection channel
- Dedicated network card
- Dedicated serial port
Features

- Flexible deployment
- Load balancing
- Data integrity
KingSCADA: Open Platform

Open interfaces support:
- ActiveX Control
- OPC
- DDE
- API

Third-party access to alarm data, historical data.

Easily integrate KingSCADA data into other information system.
KingSCADA: Building Enterprise Information Portal

- Web publishing without any conversion, build B/S architecture
- Help build enterprise portal
KingSCADA: Much More…

- Strong software development team, over 100 experienced and dedicated developers
- Internationalization of project, breaks language barriers, fits for global market needs
- Modulated product management, great for creating OEM software and provide solutions
- Provide customized device drivers
- Excellent customer service, provide A+ services
Section 2 – Create a new project
- Installation & System requirement
- Create a solution
- Create a project
KingSCADA: Hardware requirement

- Processor Pentium III
- CPU speed 1GHz
- 32 or 64 bit CPU
- 512MB RAM
- 2G HDD

KingSCADA: System requirement

- Supports Windows 32 or 64 bit operation system (English, Chinese, Japanese, Korean, Chinese-traditional characters)
- Windows Xp, sp2
- Windows Vista Ultimate
- Windows 7 Ultimate
- Windows 2003 server, sp2 (or R2)
KingSCADA: Process of Installation

- Install KingSCADA
- Install KingSCADA Drivers
- Install Key Drivers
- Install KingSCADA
Application: To know KingSCADA

KingSCADA has several parts:

- **KSMaker**: development environment
- **KSView**: runtime application
- **IOServer**: for acquiring data from I/O devices
- **DriverSetup**: for installing new drivers
- **User's guide**: for help
- …
To know KingSCADA: Development Environment (KingMaker)

- **Menu bar**: basic functions to operate projects
- **Tool bar**: shortcuts of menu commands
- **Tree-structured directory**: constituent parts of the project; the entrance of every function
- **Editing area**: view area of each function
- **Information display area**: output information
To know KingSCADA: Graphic Editor

Create a New Project

1. Menu bar
2. Toolbar
3. Toolbox
4. Editor
5. Property configuration tab
6. Links tab
7. Content tab
8. Object browse tab
KingSCADA: To create a new project

Steps of building new project:
Start-up the KingMake, select “new” icon in the “file” menu, then pop-up the dialogue box as below:
KingSCADA: To create a new project

Two projects have been created in the new solution:

Click the solution name in the directory tree on the left, the information of these projects can be shown on the right.

The information of the solution and projects.
Practice & Discussion

1. About KingSCADA features?
2. About KingMaker?
• Section 3 – Connect with I/O device (IOServer)
  - General introduction
  - Performance features
  - Supervisory features
  - Build a project
  - Connect KingSCADA and IOServer
  - Create tags (data model)
KingSCADA and external devices

KingSCADA communicates with I/O device through IOServer.

IOServer is used to communicate with the device on-site, and acquire real time data and control on-site data of the modules.

IOServer supports popular PLC, intelligent module, intelligent instrument, transducer and data acquisition boards, etc. both domestic and abroad.

IOServer can communicate with the devices through standard communicate interface to transfer data.

Engineers don’t need to concern about the complicate code and device communication protocol but need to know how to connect with I/O device and create tags corresponding with the I/O variables.
IOServer: What is IOServer?

- Data acquisition and transmission
- Data sources: Field devices, software, files, manual input, … etc.
- Provide standard interface to third-party software; or directly implement the interface of third-party software
- Configuration mode; Support data cache, redundancy etc.
IOServer: System architecture
IOServer: Support more than 3500 drivers…

Summary by 2009-03 (total: 3540)

- Module: 1046
- Instruments: 961
- PLC: 265
- Interface Card: 413
- Other: 759
- VFD: 96
Define data block to enhance the efficiency:

- **Block**: the minimal communication unit to the device. One block determines a couple of fixed command to device and expected response from it.

- **Block properties**: Register, start/end address, acquisition times and order in a circle.
**Features**

**Show the channel’s acquisition status:**

- Actual costing
- Order
- Frequency of block
IOServer: Diagnosis

Show the channel’s acquisition status:

- Actual costing
- Order
- Frequency of block
The process of creating new IOServer:

Click menu “File -> New IOServer”, the window for new IOServer setting will be shown as below:

Fill the parameters of the new IOServer, and click “OK”.
Process of creating new channel:

1. Build a New Project
2. Set the channel connection timeout here.
   - Timeout range: 0-65535 ms
3. Set the serial port settings.
   - Ensure it matches the device settings.
   - Options include:
     - Port:COM
     - Baud Rate: 9600
     - Data Bit: 8
     - Stop Bit: 1
     - Parity: None
     - Flow: RTS

Channel Name: IOServer1
Device Driver: ModbusMaster
Channel Type: Serial Port
**IOServer: New Channel**

**Redundancy channel setting:**

If there is another channel for this redundancy setting, “Use Double-IOServer Host standby” can be chosen.

Specify the Slave or Master server by “Name”, “IP”, and set the heart beat interval time, etc.

Also, if “Use Special Netcard for Status Check” has been chose, it can detect heartbeat between the “Slave” and the “Master”.

![Build a New Project](image-url)
**IOServer: New Device**

**Process of creating new device:**

Note: If “Assign Redundant Secondary Device” have been checked, there will detect whether there is another one for the redundancy, if not, the redundancy is unavailable for the device.
**IOServer: New Data Block**

**Process of creating new data block:**

Right click the blank area of “IOserver1-Modbus1”, and select “New Block…”; Configure the setting of new block, like the “name”, “Size” (block size), “Acquire” (Acquire frequency), and so on…
The process of creating new IOServer:

Click menu “File -> New IOServer”, the window for new IOServer setting will be shown as below:

Fill the parameters of the new IOServer, and click “OK”.

Practice
IOServer: New IOServer From OPC

IOServer From OPC Configuration:

Practise
IOServer: New IOServer From OPC

IOServer From OPC Configuration:

[Image of software interface showing configuration settings]

WellinTech Product Training — Yuehong Kan
IOServer: New IOServer From OPC

IOServer From OPC Configuration: Define a tag from new IOServer From OPC

![Tag Properties Window]

- Tag Name: JOTagDisc0
- Data Type: IODisc
- Description:

- Access Name: newopc.TextOPC
- Data Block: BlockRandom
- Item: 300001

- Read/Write: Read Only
- Data Convert: None

- Collect Settings: Enabled

2010-11-18 WellinTech Product Training — Yuehong Kan
Practice & Discussion

1. About IOServer features?
2. About IOServer application?
3. About IOServer from OPC?
Step 1: Build a new project: Create a new directory to store the documents associated with the project.

Step 2: Configure the hardware: Configure the hardware settings of the equipment used in the project.

Step 3: Define variables: Define global variables including memory variables and I/O devices.

Step 4: Create graphics: Draw monitoring pictures according to the project requirements.

Step 5: Define animation links: According to the on-site monitoring requirements, the fifth step is to make the static pictures produce animation effects simulating process control objects.

Step 6: Write an event script: Created in order to complete the complex control process.

Step 7: Configure of other necessary functions: Networks, recipes, SQL access, WEB browsing etc.

Step 8: Operate and debug the project.

Step 9: After finishing the above steps, a simple project has been created.
KingSCADA: Connect with the hardware you want to access

Connect with IOServer:

Create IO Server in KingSCADA. This IO Server can be connected with IOServer by “IP Address” and “Port”.

![Connection Diagram]
KingSCADA: Create tags

Tag can be created in KingSCADA:
KingSCADA has many types of tag, like I/O tag, Memory tag, etc.

Tag type in KingSCADA:

- **System tag**: Basic tag, each tag is an individual object and cannot be modified.
- **Basic tag**: Basic tag, each tag is an individual object.
- **Structure tag**: The tag exists in the format of structure and is the collection of several basic tags.
- **Pointer tag**: Using a group of tags to substitute tags all of which have the same data type.

Data type of basic tags:

- **I/O tag**: Boolean, real number, integer type and string
- **Memory tag**: Boolean, real number, integer type and string
KingSCADA: Create tags

Step 1: “Database” → “Tag Dictionary”, and click “New” upon the right area where can display tags.

Tag configuration window will be shown as below:

**Tag Properties:**

- **General:** set initial value, save value or parameter, security, etc.
- **IO:** set I/O connection, read/write mode, data converting, collect settings.
- **Alarm:** set alarm settings, like limits, alarm type, etc.
- **History:** set how to store the historical data for the tag.

![Tag Properties Window](image.png)
### KingSCADA: Create tags for the training project

#### Step 2: define the tags which the training project.

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag type</th>
<th>Data type</th>
<th>Description</th>
<th>Initial value</th>
<th>Sensitivity</th>
<th>Max</th>
<th>Min</th>
<th>Save value</th>
</tr>
</thead>
<tbody>
<tr>
<td>temperature</td>
<td>basic</td>
<td>Mem float</td>
<td>reactor temperature</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>selected</td>
</tr>
<tr>
<td>liquid_level</td>
<td>basic</td>
<td>Mem float</td>
<td>reactor liquid level</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>selected</td>
</tr>
<tr>
<td>valve_gate1</td>
<td>basic Disc</td>
<td>Mem</td>
<td>reactor feed-in valve 1</td>
<td>closed</td>
<td>0</td>
<td></td>
<td></td>
<td>selected</td>
</tr>
<tr>
<td>water_pump</td>
<td>basic</td>
<td>Mem Disc</td>
<td>reactor discharging water pump</td>
<td>closed</td>
<td>0</td>
<td></td>
<td></td>
<td>selected</td>
</tr>
<tr>
<td>motor</td>
<td>basic</td>
<td>Mem Disc</td>
<td>reactor stir motor</td>
<td>closed</td>
<td>0</td>
<td></td>
<td></td>
<td>selected</td>
</tr>
<tr>
<td>valve_gate2, valve_gate3</td>
<td>basic Disc</td>
<td>Mem</td>
<td>reactor feed-in valve 2 (3)</td>
<td>closed</td>
<td>0</td>
<td></td>
<td></td>
<td>selected</td>
</tr>
</tbody>
</table>
KingSCADA: Create data model

Create data model: define the tags which the training project. It can include variables and script algorithms.

Name: the name of data model, can be "Control"

Description: to describe this data model

Version: show the data model ID to verify the version

Author: note the author of the data model

Security: password can be set for the data model, when it is used by somebody, a password is needed.
KingSCADA: Create data model

Create data model: define the tags which the training project. It includes tags, parameters and script algorithms.

Create parameters: create parameters for the training project, “HiLimit”, “LoLimit”. 

![DataModel Definition](image1)
![Parameter](image2)
KingSCADA: Create data model

Create tags: define the tags in data model for training project, liquid_level, valve_gate, water_pump.

Name: **liquid_level**  
Data type: Mem Float  
Description: liquid level  
Initial value: 0  
Sensitivity: 0  
Min: 0  
Max: 100  
Save value: selected  
Locked properties

Name: **valve_gate**  
Data type: Mem Disc

Name: **water_pump**  
Data type: Mem Disc
KingSCADA: Create data model

Create script: define 3 conditional type of script in data model, for the training project.

1 Write in condition:
   liquid_level>liquid_level.HiLimit
Write in entering:
   valve_gate=0;
   water_pump=0;
2 Write in condition:
   liquid_level<liquid_level.LoLimit
Write in entering:
   valve_gate=1;
   water_pump=1;
3 Write in condition:
   valve_gate==1
Write in existing:
   liquid_level=liquid_level+0.78;
KingSCADA: Create object instantiated from data model

Instantiation: click “New” for “Data Model Instance”, configure the object instantiated from data model.

Name: set data model instance’s name
Data Model: choose the data model
Pointer: create only pointer object, not create real tags
Parameter Settings: the parameters predefined in data model can be changed here, fill current value in the blank corresponding.

For the training project, create 2 objects derived from this data model:
1 activator
2 material
Practice & Discussion

1. About datamodel?
2. About the process of creating datamodel?
3. About the instantiation from datamodel to data model instance?
• Section 4 – Graphic System
  – Graphy Editor (Development Environment)
  – Scripting
  – Animation
  – Extension elements (X-Y Curve, Trend curves, Alarm Windows, etc.)
  – Zoom in development environment and run time

• Section 5 – Genius
  – Genius feature
  – How to make genius

• Section 6 – Graphic Model

• Section 7 – History and Alarm
Section 4 – Graphic System

- Graphy Editor (Development Environment)
- Scripting
- Animation
- Extension elements (X-Y Curve, Trend curves, Alarm Windows, etc.)
- Zoom in development environment and run time
To know KingSCADA: Graphiy Editor

- Menu bar
- Toolbar
- Toolbox
- Editor
- Property configuration tab
- Links tab
- Content tab
- Object browse tab
To know KingSCADA: Graphic Editor — Menu bar

The menu bar is located at the top of the Graphy editor system. It provides basic functions for project and picture operations.

- **File**: mainly for the commands of the projects, like “New”, “Open”, “Import”, “Export”, “Save”, “Save As”, etc.
- **Edit**: mainly for the commands of developing the graphics, like “Undo”, “Redo”, “Cut”, “Copy”, “Paste”, etc.
- **View**: to control the visible of the function windows, for example, click “Toolbar”, it can be hidden or shown.
- **Object**: commands to insert selected objects into the picture, for example, click “OCX Control” means to insert an OCX control into current picture.
- **Draw**: commands to layout the graphics on the picture, like “Group”, “Ungroup”, “Align”, “Arrange”, “Turn”, etc.
- **Window**: provides operations for documents, such as closing current document or all the documents.
- **Help**: You can get the version number of graphy editor with Help menu.
To know KingSCADA: Graphic Editor — Tool Bar

Tool bar provide the shortcuts of the command of the menu. Most shortcut keys in the Tool Bar of Graphy Editor are explained in the menu command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Function description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Page Up" /> <img src="image2" alt="Page Down" /></td>
<td>Switchover pictures with the two button, the former is equal to Page Up, while the latter is equal to Page down</td>
</tr>
<tr>
<td><img src="image3" alt="Compile" /></td>
<td>Click the command to compile the editing picture, if no error exists, compiling successful dialog box is popped up</td>
</tr>
<tr>
<td><img src="image4" alt="Open View" /></td>
<td>Click the command to open KingSCADA View system</td>
</tr>
<tr>
<td><img src="image5" alt="Show Grid Line" /></td>
<td>Click the command to show grid line in the picture, click again, grid line is disappeared</td>
</tr>
<tr>
<td><img src="image6" alt="Snap To Grid" /></td>
<td>Click the command to enable the Snap To Grid function, click again the function will be invalid</td>
</tr>
<tr>
<td><img src="image7" alt="Show Ruler" /></td>
<td>Click the command to show dividing ruler in current picture, click again, dividing ruler is disappeared</td>
</tr>
<tr>
<td><img src="image8" alt="Zoom In" /></td>
<td>Click the command to zoom in the current picture without limitation</td>
</tr>
<tr>
<td><img src="image9" alt="Zoom Out" /></td>
<td>Click the command to zoom out the current picture without limitation</td>
</tr>
<tr>
<td><img src="image10" alt="Auto Zoom" /></td>
<td>Click the command, the size of the window will be adjusted automatically according to the proportion of the picture, so that the whole picture can be displayed in the window, when the button is pressed down, move the mouse into the picture and select an area to be zoomed in, then display the zoomed area in the current window</td>
</tr>
<tr>
<td><img src="image11" alt="Reset" /></td>
<td>After the picture is zoomed in or out, click the button to get the picture in original size</td>
</tr>
<tr>
<td><img src="image12" alt="Zoom Scale" /></td>
<td>Click the command and select a zoom scale in the drop-down list for the picture, the zoom scale can be: 20%, 50%, 100%, 150% and so on</td>
</tr>
</tbody>
</table>
To know KingSCADA: Picture

Create a new picture in Graphy Editor, “File->New->Picture...”, configure the picture setting as below:
To know KingSCADA: Picture

Configuration for the picture:

<table>
<thead>
<tr>
<th>Picture view</th>
</tr>
</thead>
<tbody>
<tr>
<td>Window location:</td>
</tr>
<tr>
<td>X: 0</td>
</tr>
<tr>
<td>Y: 0</td>
</tr>
</tbody>
</table>

- **Window location**: the location of the window in the runtime screen.
- **Window size**: the size of display of the picture in runtime.
- **Picture size**: the real size of a picture.
- **Allow using scroll bar**: use scroll bar when the window size is smaller than the picture in runtime environment.

<table>
<thead>
<tr>
<th>Picture style</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show title bar</td>
</tr>
<tr>
<td>Border style:</td>
</tr>
<tr>
<td>Window style:</td>
</tr>
</tbody>
</table>

- **Show title bar**: the location of the window in the runtime screen.
- **Sizable**: the size of display of the picture in runtime.
- **Closebox**: the real size of a picture.
- **Allow using scroll bar**: use scroll bar when the window size is smaller than the picture in runtime environment.

<table>
<thead>
<tr>
<th>Advance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Always open while running</td>
</tr>
</tbody>
</table>
To know KingSCADA: Graphic Editor — Toolbox

The toolbox of Graphy editor is composed of three parts: Basic tools, Extended tools and Windows UI controls.
To know KingSCADA: Graphic Editor — Content window

The Content window is used to add custom properties and custom variables, and it is composed of two parts: properties and variables.

**Custom Properties:**
properties defined by users and can be used by multiple different graphic objects after definition, so that different objects can have a common property. When the property is changed, property value of the object that has this custom property will be changed accordingly.

**Custom Variables:**
variables defined by users, they are private variables and can be used to manipulate pictures, and the manipulations include opening a picture, viewing a picture, closing a picture and so on.
To know KingSCADA: Graphic Editor — Object window

Object window can display all objects on current picture, the name, the thumbnail, object visibility state of all the objects and other information about the objects.

- **Edit visible**: set the visible status under edit environment
- **Edit lock**: set the edit availability under edit environment
- **Member access**: set the accessibility in the script
- **Cut**
- **Copy**
- **Rename**: rename the object
- **Find**: find specific object
- **Replace**
- **Expand all**: to expand all the sub-objects that are included in the selected object
- **Collapse all**: collapse all the sub-objects that are included in the selected object
To know KingSCADA: Graphic Editor — Links window

Animation link properties of the object can be set in Links window of the Graphy Editor, properties such as Fill, Shift, Scale, Mouse Trigger and so on can be set in Links window.
Create object on picture: Round rectangle

**Rectangle**: set properties through property window;

**Properties**: 

**Line style setting:**

**Brush setting:**
Create object on picture: Text

Text: for example, set text context with “## m”, if the value is 2.4, it will be displayed as “2.4 m”. Set animation with tags defined before.

Note: Format using “#” for analogy values
1 “# ton” -> “20.4 ton”
2 “The weight is # ton” -> “The weight is 20.4 ton”

Practice: set animation to text using data model tag.
1 Set text content as “Activator's Liquid level: #”

2 Set “AnalogOutput” using tag “\local\activator.liquid_level” and set parameters as the picture on the right.
Create object on picture: Pipe

Pipe: this element designed to show the pipe-like status, like water or liquid flowing inside.

Properties:

1. OuterWidth: set the width of the pipe, in pixels;
2. OuterColor: set the outside colour of the pipe;
3. InnerWidth: set inner width of the pipe, in pixels;
4. InnerColor: set inner colour of the pipe;
5. StartPoint: location of the start point of the pipe;
6. EndPoint: location of the end point of the pipe;

<table>
<thead>
<tr>
<th>Style</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>OuterWidth</td>
<td>16</td>
</tr>
<tr>
<td>OuterColor</td>
<td>#FF000000</td>
</tr>
<tr>
<td>InnerWidth</td>
<td>10</td>
</tr>
<tr>
<td>InnerColor</td>
<td>#FFFF0000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>StartPoint</td>
<td>331.0, 153.0</td>
</tr>
<tr>
<td>EndPoint</td>
<td>592.0, 234.0</td>
</tr>
</tbody>
</table>
Create object on picture: Pipe

Pipe: this element designed to show the pipe-like status, like water or liquid flowing inside.

Practice:

1. Draw a pipe on the picture

2. Set the animation of the pipe with tag \"\local\valve\_gate1\". 

The speed of the flowing, the exact time of the frequency of the refreshment is defined in system configuration.
Create object on picture: Button

**Button**: it is a button has two styles, it can be common button and poly-button which has several state to display.

Properties:
1. **FlatStyle**: set the appearance;
2. **Font**: set the font of the button text;
3. **Background**: set button’s background;
4. **Foreground**: set button’s foreground;
5. **PressedBackground**: set the background color of the button when the button is pressed down;
6. **FocusedBackground**: set the background color of the button when the button is focused;
7. **DisabledBackground**: set the background color of the button when the button can not be manipulated that is the Enable property is set to false;
8. **Caption**: the text displayed on the button;
9. **ShapeStyle**: set the shape of the button;
10. **Icon**: the icon displayed on the button;
11. **Style**: normal or poly, for poly is that the button can have several states can be displayed.
Create object on picture: Button

**Button**: it is a button has two styles, it can be common button and poly-button which has several state to display.

Practice:
1. Create a button on the picture;
2. Set the properties as the picture on the right;
3. Set animation “MouseTrigger->LeftDown” to this button for the training project as below:

```java
if(!!local\valve\_gate1==false)
{
   \local\valve\_gate1=true;
}
else
{
   \local\valve\_gate1=false;
}
```
Practice & Discussion

1 About Graphy Editor?
2 About animation?
3 About objects?
Section 5 – Genius

- Genius feature
- How to make genius
- Practice
**Genius**: a group of objects with animations, variables, script and properties to represent to simple devices, like switch or valve etc.

**Components of genius:**

1. **Graphics**: support almost all kinds of objects in the toolbox, the same as the pictures development.

2. **Custom properties**: to be used as a interface to set the properties of genius when use the genius on the picture.

3. **Custom variables**: to be used inside the genius for animation and script, and will be connected with real tags when the genius being used on the picture.

4. **Links (animations)**: the animations defined inside of the genius, it can’t be seen when it is used on the picture.
Genius: a group of objects with animations, variables, script and properties to represent to simple devices, like switch or valve etc.

Practice: create a genius

1. Create a genius in Graphy Editor. (“File->New-> Genius”);
2. Draw objects for the genius;
3. Define variables and properties as need;
4. Links (Animations) as need.
**Genius**: a group of objects with animations, variables, script and properties to represent to simple devices, like switch or valve etc.

**Practice**: create a genius

3 Define variables and properties as need;

For example:
1. **BackColor**: brush;
2. **fluidColor**: brush;
3. **Max**: float;
4. **Min**: float;
**Genius:** a group of objects with animations, variables, script and properties to represent to simple devices, like switch or valve etc.

**Practice:** create a genius

3 Define variables and properties as needed;

For example: define `fluidValue(float)`: 

---

![Variable configuration dialog box](image)
Genius: a group of objects with animations, variables, script and properties to represent to simple devices, like switch or valve etc.

Practice: create a genius

1. Use “Max” to configure the text of “100”;
2. Use “Min” to configure the text of “0”;
3. Use “BackColor” to configure the object rectangle A’s background;
4. Use “fluidColor” to configure the object rectangle D’s background;
**Genius:** a group of objects with animations, variables, script and properties to represent to simple devices, like switch or valve etc.

**Practice:** create a genius

4 Define links (animations) as need;

For example:

1 Add “VerticalScale” animation to rectangle D, using inner tag “fluidValue”;
2 Use “Min” to configure the parameter “At Min Height”;
3 Use “Max” to configure the parameter “At Max Height”;
Genius: a group of objects with animations, variables, script and properties to represent to simple devices, like switch or valve etc.

Practice: save a genius

5 Save the genius to library.

Note: A new folder can be created under the system folder.
**Genius:** A group of objects with animations, variables, script and properties to represent to simple devices, like switch or valve etc.

**Practice: save a genius**

5 Save the genius to library.

Note: A new folder can be created under the system folder.
Use genius: insert genius into a picture and configure it.

Practice:
1. Find the genius needed;
2. Configure the genius' properties;
Use genius: insert genius into a picture and configure it.

Practice:

3 Link a real tag to the genius.
Practice & Discussion

1 About genius creation?
2 About genius variables?
3 About genius properties usage?
4 About genius management?
5 About genius application?
Section 6 – Graphic Model
- General introduction
- How to make Graphic Model
- Practice
Graphic Model:

- Graphic model has graphics, variables, links, script and other items;
- It can be combined by basic graphic elements, ActiveX controls, and even genius;
- Feature: The change of model can change the objects generated from it.

Note:

Difference from genius: “Infection”
Graphic Model:

Practice: make a graphic model.
1. Create a graphic model in KingMake: “New” -> fill the information of the graphic model.

Note: Also can be created in Graphy Editor, “New” -> ”Graphy model”
Graphic Model:

Practice: make a graphic model.

2 Add graphics into graphic model, as the picture on the right, part A, part B and part C;

3 Insert Genius “TankGenius” created before into this graphic model;
Graphic Model:

Practice: make a graphic model.

4 Define properties being used to configure genius and:
- Tank
- Pan
- Indic
- Text
- Max
- Min

5 Configure:

- TankColor (Brush)
- PanelColor (Brush)
- IndicatorColor (Brush)
- TextColor (Color)
- TextFont (Font)
- Max (Float, 100)
- Min (Float, 0)
Graphic Model:

Practice: make a graphic model.
4 Add animation to objects in graphic model, connect tag “fluidValue” to “Text1” (#)’s Analogoutput animation and the genius’ link;
Graphic Model:

Practice: use script in graphic model.

6 In graphic model, there are OnOpen, WhileOpen, OnClose. Use script to change Text1’s color when it is exceed alarm limit.

1> Create 2 brush properties, “NormalTextColor”, “AlarmTextColor” as below:

2> Set “true” the property “MemberAccess”: of Text1, so that it can be accessed in script.
Graphic Model:

Practice: use script in graphic model.

6 In graphic model, there are OnOpen, WhileOpen, OnClose. Use script to change Text1’s color when it is exceed alarm limit.

3> Set script in “OnOpen” and “WhileOpen” for graphic model to change text color when the alarm happens.

```plaintext
float HAlarm; float LAlarm; HAlarm=Max*0.9; LAlarm=Min*0.1; if(fluidValue<LAlarm||fluidValue>HAlarm)   Text1.TextBrush=AlarmTextColor; else   Text1.TextBrush=NormalTextColor;
```

Graphic Model:

Practice: insert graphic model into the picture.

7 Link tag “\local\activator.liquid_level “to the graphic model.
Practice & Discussion

1 About graphic model?
2 About usage of graphic model?
3 Practice: the “Infection” feature using the graphic model.
Section 7 – History & Alarm

- Historical Record Setting
- Historical Trend Curve
- Alarm Record Setting
- Alarm Window
**Historical data record:**

KingSCADA has its own historical database.

1. Configure “History Record Setting”;

2. Advanced settings for buffer.

---

![Historical Record Setting](image1)

![Advanced Record Setting](image2)
Historical Trend Curve: KingSCADA uses History Window to display the trend curve.

History Window:
1 Insert “History Window” from toolbox into a picture;
2 Set the properties of History Window
Historical Trend Curve: KingSCADA uses History Window to display the trend curve.

History Window:

3 TrendMode: can be “Historian Mode” or “Realtime Mode”;

4 Edit inner objects: right click the trend curve, click “Edit”, enter the inner of History Window;

5 Add Curve: right click grid area, “Add Curve” can add a new curve into the window;

6 DataSource: click the Curve, set the property of “DataSource”, choose historical tag as data source;

7 TimeAxis: there can be 2 time axis for comparison of two curves;

8 DataAxis: there can be 4 data axis at maximum;

9 ChartAreaCount: there can be 4 areas at maximum;

….See more in the user’s guide.
Historical Trend Curve: KingSCADA uses History Window to display the trend curve.

**History Window functions:**

1. **Query data to display curves:**
   - For example: Use a button and script: `TrendChart1.SetTimeAxis("TimeAxis1");`

2. **Print curves:**
   - For example: Use a button and script: `TrendChart1.PrintChart();`

3. **Get average value of one curve:**
   - For example: Use a button and script: `GetCurveValue("Curve1", "AverageValue");`

....See more in the user’s guide.
Alarm Window: used to display and query alarm information.

Alarm System settings:

- Alarm and Event Setting
- Store Setting: Alarm DB Setting
  - Record Setting
    - Record Selection
      - Record alarm event
      - Record operate event
      - Record enter event
      - Record station event
  - Alarm Group and Level
  - Alarm Group
  - Alarm Priority
- Alarm Buffer: 1000
- Alarm Duration: 30
- Mode of alarm-removing
  - Alarm manual delete
  - Alarm automatic delete
  - Alarm automatic delete
- Record DB timeout: 30

Email Setting

- Server Info
  - POP3: [Input]
  - SMTP: [Input]
  - User: [Input]
  - Password: [Input]
  - Port: 110
  - Port: 25
Alarm Window: used to display and query alarm information.

Alarm Window settings:
Alarm Window: used to display and query alarm information.

Alarm Window settings:
Alarm Window: used to display and query alarm information.

Alarm Window settings:
Build conditions: condition setting can build sql statement to query alarm information.

Use SQL query alarm information:
Practice & Discussion

1. Historical record??
2. About alarm setting?
3. About alarm querying?
Section 9 – Report

- Report is Excel-based in KingSCADA
- Functions
Report: excel-based report, can make real time report and historical report.

Report properties:
Report: excel-based report, can make real time report and historical report.

Report functions:

1. SetCellValue: Set value to specific cell of the report, for example “SetCellValue(2,4,\local\ Temperature); “

2. SetHistData: query data and input the result into the specific column, for example “SetHistData(“\local\liquid_level”, 1);”
Section 10 – Redundancy

- Redundancy introduction
- For types of redundancy
- Redundancy configuration
Redundancy System: to ensure the reliability and to reduce the data loss.

- There are 4 types of redundancy in KingSCADA.
  - **Double-machine hot standby**: primary machine and the secondary machine are connected via the TCP/IP network, and 2 machines are redundant.
  - **Double-network redundant**: use two network cables to build the network communication between the two machines, and 2 network is redundant.
  - **Double-facility redundant**: redundancy between 2 devices, like 2 PLC can be set to be redundant, and KingSCADA can support this redundancy.
  - **Double-IOServer redundant**: redundancy between 2 IOServers.
Double-machine hot standby: primary machine and the secondary machine are connected via the TCP/IP network, and 2 machines are redundant.
Double-machine hot standby: primary machine and the secondary machine are connected via the TCP/IP network, and 2 machines are redundant.

- Real-time data server redundancy
- Historical server redundancy
- Alarm server redundancy
Double-machine hot standby:

Configure double-machine host standby:

1 network configurations on master machine:

1> Set Online mode
2> Set local station information
3> Double-machine Hoststandby settings
4> Redundancy state detect channel

<table>
<thead>
<tr>
<th>Network Config</th>
<th>Server Mode Setting</th>
<th>Client Mode Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Network Parameter</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Stand-alone</strong></td>
<td><strong>Online</strong></td>
<td></td>
</tr>
<tr>
<td>Local Station Name: PrimopPC</td>
<td>Local Station IP: 192.168.1.1</td>
<td></td>
</tr>
<tr>
<td>Local Station Name: SecondaryPC</td>
<td>Slave Station Backup Net Card IP:</td>
<td></td>
</tr>
<tr>
<td>Slave Station Name: SecondaryPC</td>
<td>Slave Station Backup Net Card IP:</td>
<td></td>
</tr>
<tr>
<td>Slave Station Name: SecondaryPC</td>
<td>Slave Station Backup Net Card IP:</td>
<td></td>
</tr>
<tr>
<td><strong>Double-machine Hoststandby</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use Double-machine Hoststandby</td>
<td>Local is Master Station</td>
<td>Local is Slave Station</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Redundancy State Check Channels</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Serial Port</td>
<td>Serial Port Name: COM1</td>
<td></td>
</tr>
<tr>
<td>Network Card</td>
<td>Redundancy Station IP: 100.100.100.1</td>
<td></td>
</tr>
<tr>
<td>Other Parameter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Heartbeat Test Time:</td>
<td>5 s</td>
<td>Heartbeat Test Time:</td>
</tr>
</tbody>
</table>

OK | Cancel
Double-machine hot standby:

Configure double-machine host standby

2 master node type configuration:
1> Set Logon server
2> Set Data server
3> Set synchronized server and settings

3 Copy the network project
Double-machine Hot Standby

Configuring Double-machine Hot Standby:

4 Network Configuration on the Slave Machine

- Stand-alone
- Online

Local Station Name: SecondaryPC
Local Station IP: 192.168.1.2

Use Double-machine Hot Standby
- Local is Master Station
- Local is Slave Station

Master Station Name: PrimaryPC
Master Station IP: 192.168.1.1

Redundancy State Check Channels
- Serial Port
- Network Card

Serial Port Name: COM1
Redundancy Station IP: 10.100.100.1

Heartbeat Test Time: 5
Heartbeat Test: 2 Times
Double-machine hot standby:

System variables in double-machine hot standby state:

1 Master machine status monitor: system variable $RedundantStatus to monitor the primary’s status:
   - $RedundantStatus=1, the master machine is in active state at present.
   - $RedundantStatus=2, the master machine is in back-up state at present.

2 Slave machine status monitor: use system variable $RedundantStatus to monitor the slave’s status:
   - $RedundantStatus=1, the slave machine is in back-up state at present.
   - $RedundantStatus=2, the slave machine is in active state at present.

3 Transform the Status Manually: activate the primary or secondary machine using $RedundantStatus.

Note: only can be done on the machine in active state.

4 Have a try using the system tag $RedundantStatus.
Double-network redundancy: use two network cable to make the network communication between 2 machines.

Requires:
1> 2 network cards should be installed on each site in the network
2> 2 network cards should be installed in two different network segments
3> When the cable of the master network is interrupted, the network communication is switched to the secondary network automatically to ensure that the communication links are not interrupted and the operation enables the system to run steadily and reliably.
Double-network redundancy:

Configurations:

1 Network configurations on the master machine:

- Local Station Name: PrimaryPC
- Local Station IP: 192.168.1.1
- Local Backup Net Card IP: 200.200.200.1

2 Network configurations on the slave machine:

- Slave Station Name: SecondaryPC
- Slave Station IP: 192.168.1.2
- Slave Station Backup Net Card IP: 200.200.200.2

Redundancy State Check Channels:

- Serial Port
- Network Card

Other Parameters:

- Heartbeat Test Time: 5 seconds
- Heartbeat Test: 3 Times
Double-network redundancy:

Configurations:

2 master machine node type settings:
### Double-network Redundancy

#### Configurations

1. **Copy the network project**
2. **4 network configurations**
   on the slave machine

---

#### Configuration

**Network Config**

<table>
<thead>
<tr>
<th>Network Parameter</th>
<th>Server Node Setting</th>
<th>Client Node Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Stand-alone</strong></td>
<td><strong>Online</strong></td>
</tr>
</tbody>
</table>

- **Local Station Name**: SecondaryPC
- **Local Station IP**: 192.168.1.2
- **Local Backup Net Card IP**: 192.168.200.2
- **Local Station Backup Net Card IP**: 192.168.200.1

**Double-machine Hotstandby**

- **Use Double-machine Hotstandby**
  - Local is Master Station
  - Local is Slave Station

- **Master Station Name**: PrimaryPC
- **Master Station IP**: 192.168.1.1
- **Master Station Backup Net Card IP**: 192.168.200.1

**Redundancy Status Check Channels**

- **Serial Port Name**: COM1
- **Redundancy Station IP**: 192.168.100.1
- **Redundancy Heartbeat Setting**

**Other Parameters**

- **Heartbeat Test Time**: 5s
- **Heartbeat Test**: 3 Times
IOServer double-facility redundancy:

● IOServer double-facility redundancy refers to the redundancy between facilities, and it refers to the mutual redundancy between two facilities which have the same configuration.

● For data acquisition system which is rather important to users, users can collect data with two devices that are exactly the same at the same time, and communicate with IOServer, to realize the function of double-facility redundancy.

● In normal circumstances, IOServer communicates with the master facility and collects data at the same time, if the communication with the master facility breaks down, IOServer will switch to the slave facility automatically, and continue the data acquisition.
IOServer double-facility redundancy:

Configurations:

1 Set up the IOServer slave facility:
IOServer double-facility redundancy:

Configurations:

2 Set up the IOServer master facility:
IOServer double-facility redundancy:

Configurations:

3 Set up the IOServer master facility:
Pairs of IO Server Redundancy Setting:

Configurations:

1. Set up the IOServer, and create a new project:

   ![Image of New IOServer window]

   - Project Name: test
   - Location: C:\(Documents and Settings)\xuan\wu
   - Type: IOServer From Driver

2. Edit IOServer properties as the picture on the right: set as master.

   ![Image of Edit IOServer Properties window]
Pairs of IO Server Redundancy Setting:

Configurations:

3 Slave IO Server setting:

4 IO Server redundancy setting in KingSCADA3.0

Note: Please select the master IO Server. There is no need to set the slave IO Server in KingSCADA3.0, and when the master IO Server goes wrong, the slave one will realize the functions of the master one completely.
Practice & Discussion

1. About Double-machine hot standby?
2. About Double-network redundant?
3. About Double-facility redundant?
4. About Double-IOServer redundant?
Section 11 – Network

- Section 11 – Network
Section 12 – Web Publishing

- Web Publishing Configuration
- IIS and Browser Configuration
- Unavailable Functions through Web Publishing
Web publishing:

**Web Server**
- **Name**: ForTrainingWeb
- **Path**: C:\Program Files\KingSCADA\Projects\WebCab
- **Communication mode**: TCP/IP

**Station IP Map Setting**
- OriginalIP: 127.0.0.1
- MappedIP: 218.240.44.106
- StationName: local

**站点IP映射设置**
- | 原始IP  | 映射IP  | 站点名   |
- |--------|--------|---------|
- | 172.16.2.126 | 218.240.44.120 | 普通   |
- | 172.16.2.187 | 218.240.44.210 | NetServerA |
Web publishing
Web publishing:
Web publishing:
For publishing preparation:
1 IIS must be installed;
2 Configure IIS
Web publishing:

For publishing preparation:

1 IIS must be installed;

2 Configure IIS

3 MIME type configuration: after the IIS has been installed on win2003 server, vista and win7, you need to configure the MIME type for default web sites in IIS, add extension types such as .kzip, .cfg and .dat.
Web publishing:

Browser setting:
1. Configure browser security
2. Browse picture via IE
Web publishing:

Functions that KingSCADAWeb do not support are as follows:

- Systematic scripts and all the custom scripts are not supported by KingSCADAWeb, only picture scripts are supported by KingSCADAWeb (except the write operation for KingSCADA project variables)
- SQL database function
- File operation function
- Recipe
- Reference variable
- Hotkey
- Virtual keyboard
- The function Alarm Window.Print( ) which is used to print the alarm window can not be called.
Section 13 – Security and User management
User management:

Management interface:
User management:

Create new user and configure it:
User management:

Role configurations:
User management:

Runtime to use system logon, logoff:

![User Logon Window](image1)

![KingSCADA Run-Time System](image2)
User management:

Functions provided:

1. Logon functions: The logon functions are as follows:
   - **LogOn() function:** Carry out the logon operation, the logon dialog box will be popped up.
   - **LogOnNoInput (string strUserName, string strPassword, bool bNetLogMode)**
     - **Function:** Carry out the logon operation without the logon dialog box; input the password directly to log in the system.
     - **Parameter specifications are as follows:**
       - a) **strUserName:** string, the name of the logon user
       - b) **strPassWord:** string, the password of the logon user
       - c) **bNetLogMode:** bool, specify whether the logon user is a net user or a local user, True: net user, False: local user
     - **Script as:** LogOnNoInput(“user1”, “user1”, Flase);

2. Logoff function: LogOff()
Practice & Discussion

1 About web publishing?

2 About KingSCADA anything else?