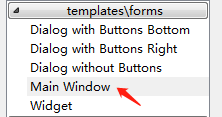
# Electric curtains

## 一、Development steps

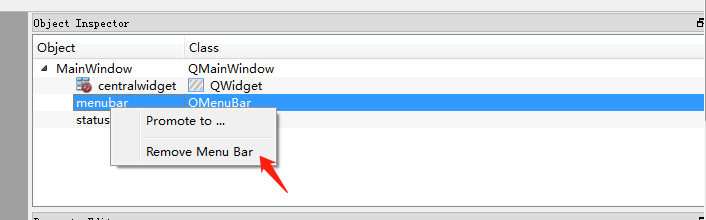
## Use Qt Designer to develop UI

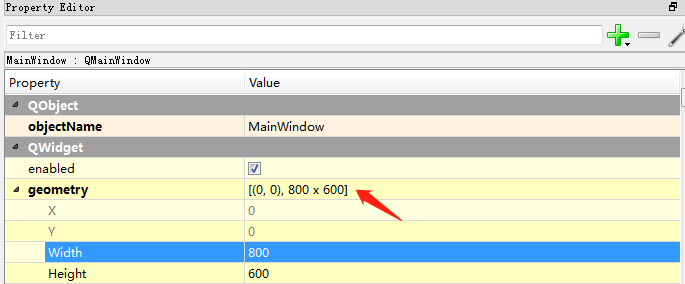
1. Create ui file

Open Qt Designer and create the file named ElectricCurtainUI.ui, its type is MainWindow , its path is C:\Users\Administrator\Desktop\Demo\ElectricCurtainUI.ui



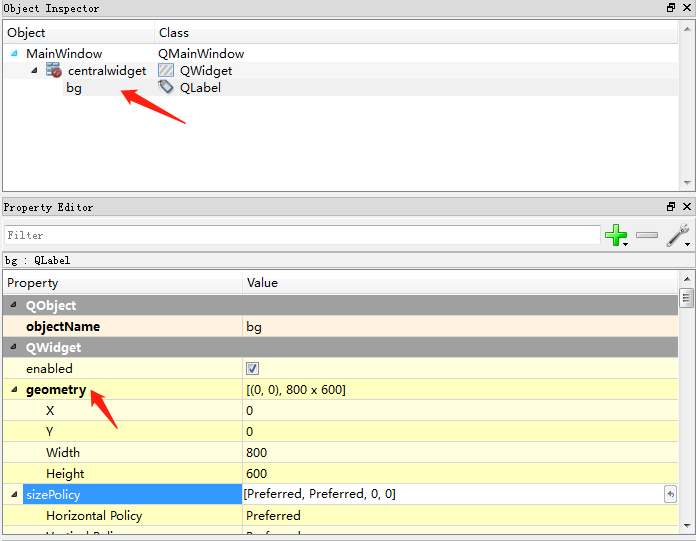
n the Object Inspector panel on the right, select menubar, right-click Remove Menu Bar, and remove menubar. Similarly, right-select statusbar and delete it.

Select MainWindow and find the geometry property in the Property Editor panel Set Width and Height to 800 and 600 respectively.

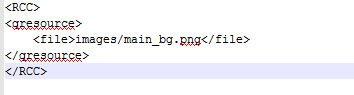


2.Add background image

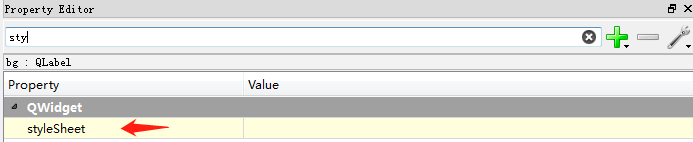
Find the Label object in the WidgetBox panel, drag it to the MainWindow form, set the Label name bg in the Object Inspector panel, and set the Width and Height of the geometry property in the Property Editor to 800 and 600 respectively. X and Y Both are 0, setting the Text property to null.



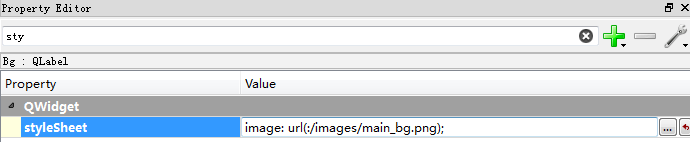
Create an APPResources.qrc file in the directory where ElectricCurtainUI is located, create a folder named images, and copy the downloaded main\_bg.png background image to the images directory. Right-click on the APPResources.qrc file with text editor and enter the following code:



Select the Label named bg in Qt Designer and find the styleSheet property in the Property Editor panel.

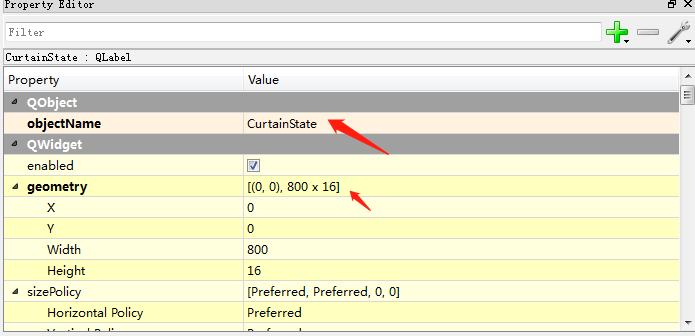


Set the value of the styleSheet property as shown:



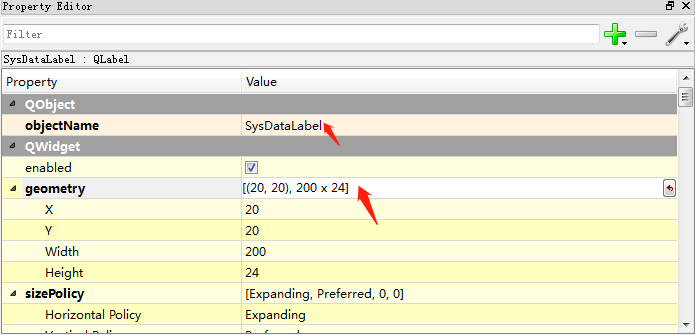
1. Curtain status information prompt box

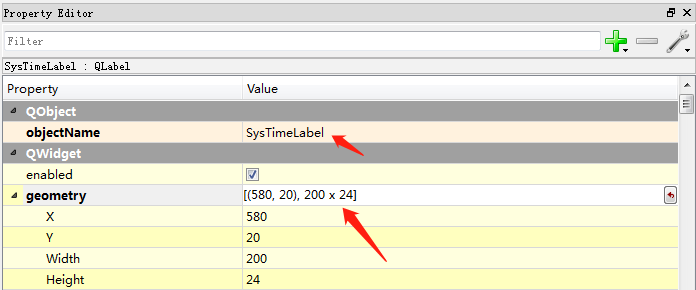
Drag and drop a Label from the Widget Box to the main window. It is used to display the current status of the window shade. Set the property editor panel properties as follows:



1. Date and time prompt box

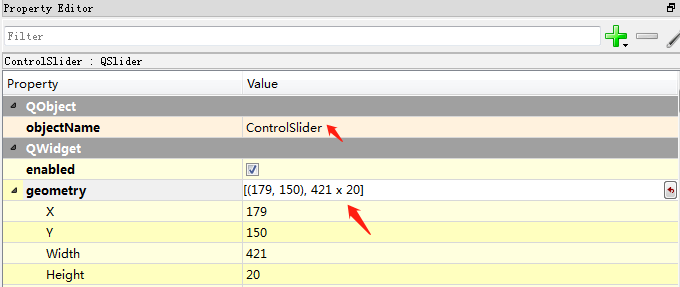
Drag and drop a Label from the Widget Box to the main window. It is used to display the system date. Set the Property Editor panel properties as follows:

Drag and drop a Label from the Widget Box to the main window. It is used to display the system date. Set the Property Editor panel properties as follows:

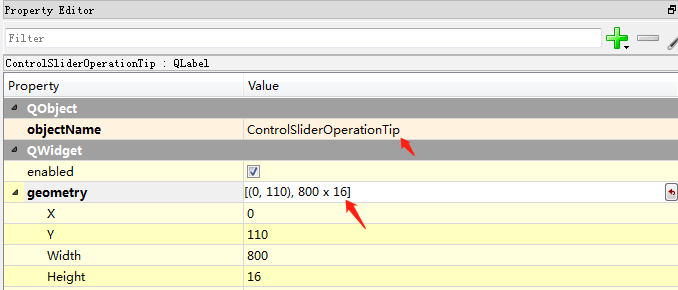


1. Curtain position progress bar

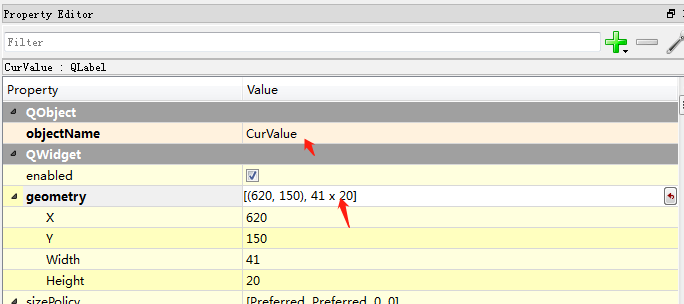
It is used to display and set the current curtain position. Drag and drop a Horizontal Slider from the Widget Box to the main window.



Drag and drop a Label from the Widget Box to the main window. It is used to display curtain operation tip. Set the Property Editor panel properties as follows:

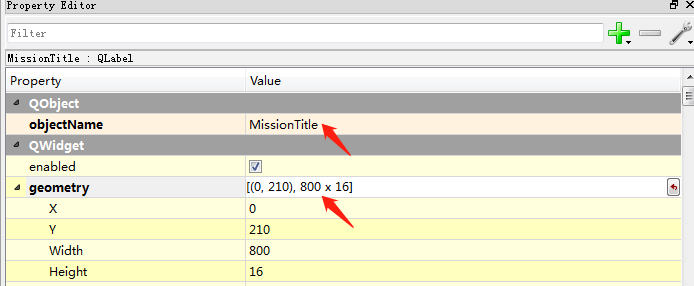


Drag and drop a Label from the Widget Box to the main window. It is used to display the current position of the curtain. Set the Property Editor panel properties as follows:

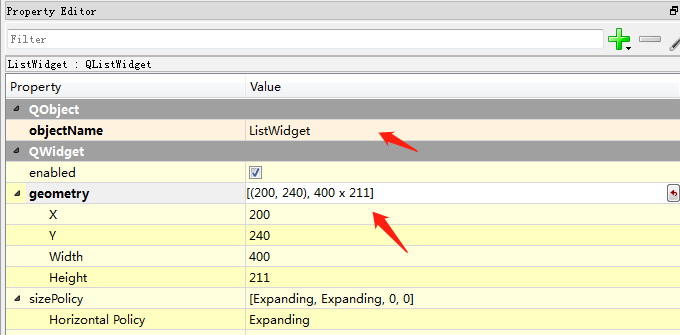


1. Task list

Drag and drop a Label from the Widget Box to the main window. It is used to display the tip of current task list. Set the Property Editor panel properties as follows:

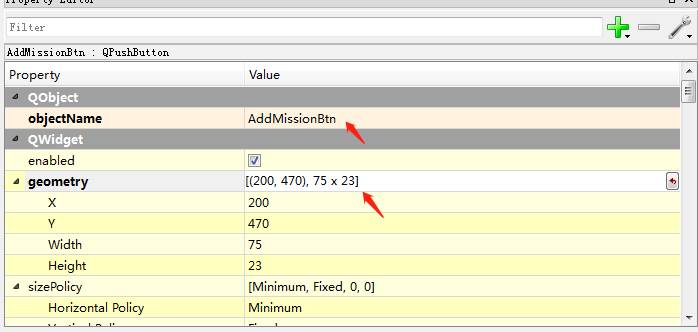


Drag and drop a List Widget from the Widget Box to the main window. It is used to display the current task list. Set the Property Editor panel properties as follows:

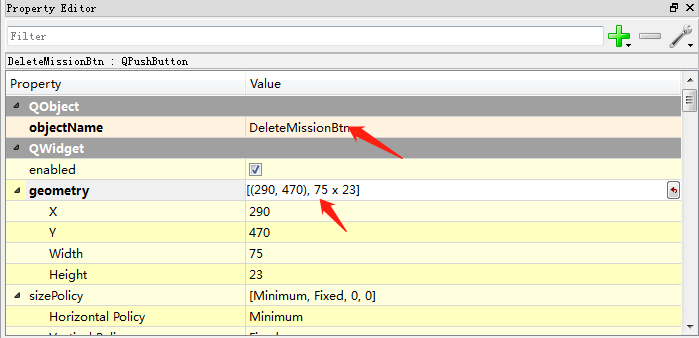


1. Task list operation panel

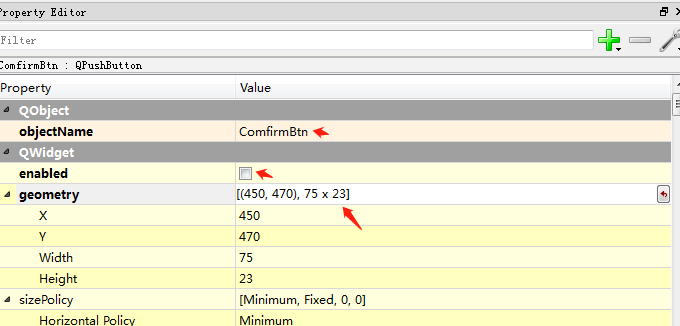
Drag and drop a Push Button from the Widget Box to the main window. It is used to add task. Set the Property Editor panel properties as follows:



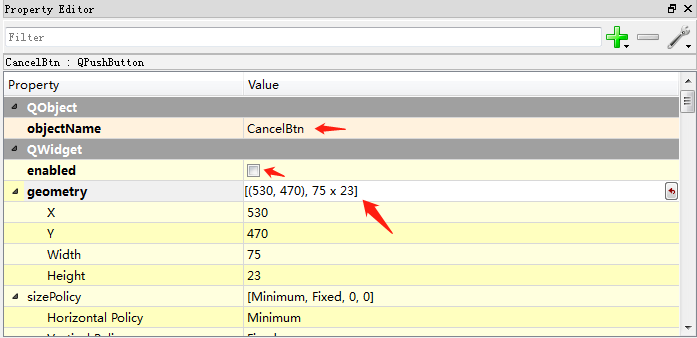
Drag and drop a Push Button from the Widget Box to the main window. It is used to delete task. Set the Property Editor panel properties as follows:



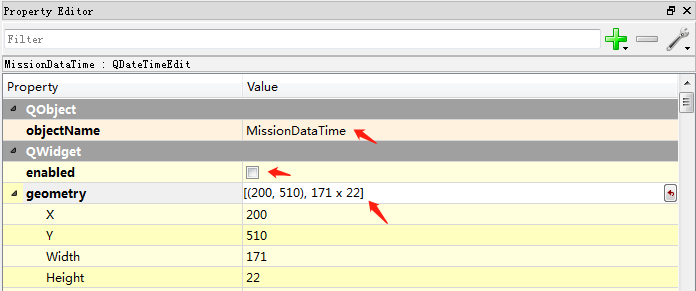
Drag and drop a Push Button from the Widget Box to the main window. It is used to confirm task. Set the Property Editor panel properties as follows:



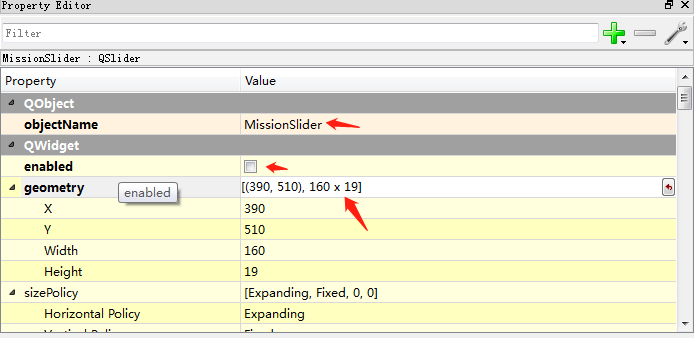
Drag and drop a Push Button from the Widget Box to the main window. It is used to cancel task. Set the Property Editor panel properties as follows:



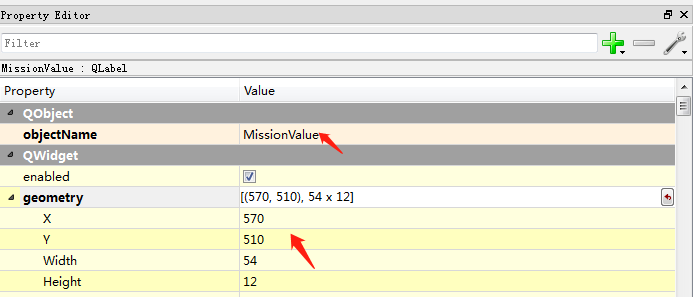
Drag and drop a DataTimeEdit from the Widget Box to the main window. It is used to set the task’s date and time . Set the Property Editor panel properties as follows:



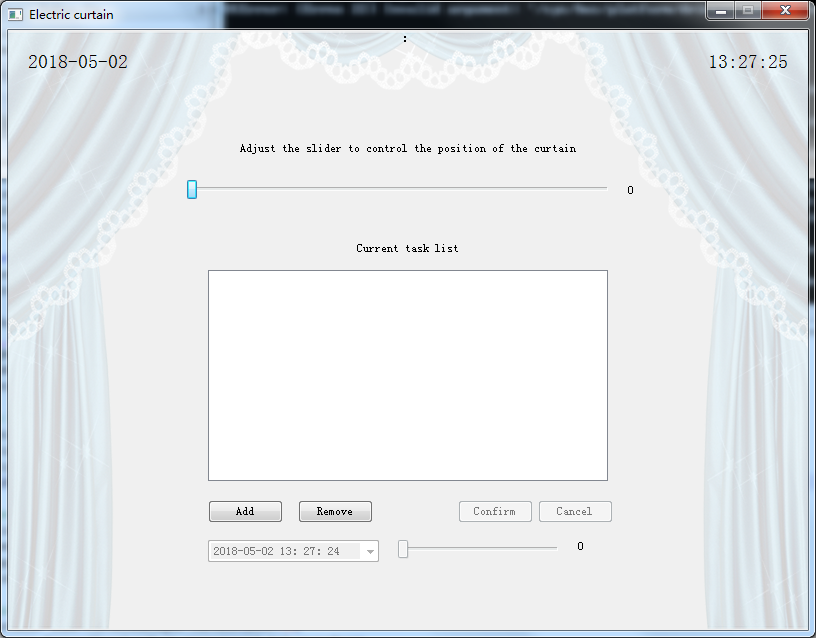
Drag and drop a Horizontal Slider from the Widget Box to the main window. It is used to set the task’s position of the curtain . Set the Property Editor panel properties as follows:



Drag and drop a Label from the Widget Box to the main window. It is used to display the curtain position of the task setting. Set the Property Editor panel properties as follows:



8.The final UI operation is as follows:

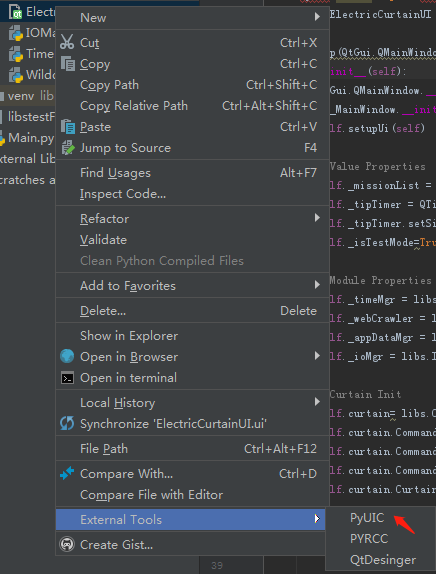


## **Use PyCharm to develop functional modules**

1. Turn the UI file into a Python file

Open PyCharm and create a project directory. The default directory is C:\Users\Administrator\PycharmProjects\ElectricCurtain. Create a libs folder in the directory and copy the Qt Designer saved UI files to the project directory. The default file name is ElectricCurtainUI. .ui.

Right-click on ElectricCurtainUI.ui to find the External Tool and select PyUIC. After that, the ElectricCurtainUI.py file is generated in the same directory as the ElectricCurtainUI.ui file.

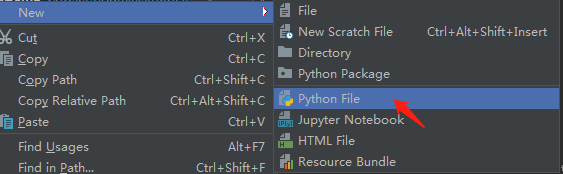


Then copy the APPResources.qrc file and the images folder referenced in Qt Designer to libs. Right-click on the PYRCC button under ExternalTools in the APPResources.qrc file, and then generate the APPResources\_rc.py file in the libs folder.

Double-click to open the ElectricCurtainUI.py file and modify import APPResources\_rc code to import libs.APPResources\_rc

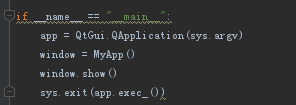
2.Create Main.py file and load UI

Right click a New Python script in the Project window, named Main, as the main program entry for the Demo.



In the Main.py file, first import the generated ui module from the libs.ElectricCurtainUI import Ui\_MainWindow and define the MyApp class. This class inherits QtGui.QMainWindow, Ui\_MainWindow, and implements QMainWindow and Ui\_MainWindow respectively in MyApp's \_\_init\_\_ method. Initialize the job.

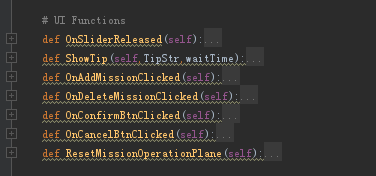
Then add the following code in the Main.py file as a program entry for Demo:



On a window7 64-bit PC, run the cmd program to enter the python plus Main.py file path to run the program.

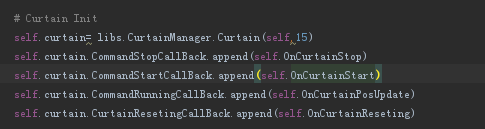
On the Qualcomm 410c development board, open the system terminal program and enter the python3 plus Main.py file path to run the program.

UI-related event logic is defined in Main.py, including curtain sliders, four buttons, and event handling of the balloon.



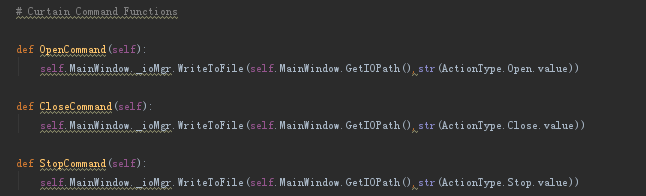
1. Curtain module

In PyCharm, create the CurtainManager.py file and define the Curtain class in the file. It implements the method logic related to curtain movement, including the four actions of curtain reset, opening, closing, and stopping movement. Define the callback method for the four actions.

In the Main.py file, the Curtain class was imported via import libs.CurtainManager, and the callback method for the four actions of the shade was implemented in the MyApp class, and the \_\_init\_\_ method was defined. Curtain object, and registered callback methods.

1. IO module

Because the Demo client operates the blinds by reading and writing numeric values in the system file, the IOManager.py file is created as the IO management of the Demo, and the IOMgr variable is defined in the Main.py file to provide the write and Read both methods. self.\_ioMgr = libs.IOManager.IOMgr() , the \_ioMgr variable is used in the Curtain class to implement the function of operating curtains.



Open the terminal and enter:

chmod 777 /sys/bus/platform/drivers/machine/soc:machine/value

1. Time module

Create the TimeManager.py file, define the TimeMgr class, and provide methods for getting the system time and timestamp.

1. AppData module

Create AppDataManager.py file to save the data in Demo program, including UI text, prompt text, and curtain-related data.

## 二、API

## **1. Curtain module（Curtain Class）**

|  |  |
| --- | --- |
| Attributes | Description |
| self.Position | Curtain location |
| self.CurAction | Curtain operating status type |
| self.CurtainRunTime | The time from when the curtain is closed to opening |
| self.MainWindow | Window object in Main.py |
| self.RunningTimer | The timer during the movement of the curtain is executed once every second to calculate the current curtain position. |
| self.CommandTimer | Curtain movement time |
| self.CommandStartCallBack | Curtains start moving callback method |
| self.CommandStopCallBack | Curtains stop moving callback method |
| self.CommandRunningCallBack | Callback method in curtain movement, performed once per second |
| self.CurtainResetingCallBack | Curtain position reset callback method |
| ActionType(Enum) | Curtain operating status type |
|  | |
| Method | Description |
| ControlCurtainWithTargetPos(self,targetPos) | Move the curtain to the specified position targetPos |
| ResetCurtain(self) | Reset curtain position |
| OpenCurtain(self,openLevel) | Open the curtains and move within the openLevel time |
| CloseCurtain(self,closelevel) | Close the curtains and move within the openLevel time |
| StopCurtain(self) | Curtains stop moving |
| OnCurtainRunning(self) | The curtain is moving |
| StopAllTimers(self) | Stop all timers |
| DoCommandStartCallBack(self) | Perform curtains to start moving callback methods |
| DoCommandStopCallBack(self) | Perform a blind stop move callback method |
| DoCammandRunningCallBack(self) | Perform callback method during curtain move |
| DoCurtainResetingCallBack(self) | Perform curtain reset callback method |
| OpenCommand(self) | Operation file execution to open curtain |
| CloseCommand(self) | Operation file execution to close curtain |
| StopCommand(self) | Operation file execution to stop curtain move |

## **2. IO module（IOMgr class）**

|  |  |
| --- | --- |
| Method | Description |
| WriteToFile(self,path,str) | Write str value into the path file |
| PrintFile(self,path) | Read the value of the path file |

## **3. Time module（TimeMgr class）**

|  |  |
| --- | --- |
| Method | Description |
| GetSystemDataTime(self) | Get the current system date and time |
| GetSystemDataTimeStamp(self) | Get the current system date and time stamp |
| StartUpdateDataTime(self,callBackFunc) | Start updating system time and executes callBackFunc callback method every second |
| StopUpdateDataTime(self) | Stop updating system date and time |
| updateTimerCallBackFunc(self) | Perform callback method of updating system date and time |

## **4. APPData module（DataMsg class）**

|  |  |
| --- | --- |
| Attributes | Description |
| self.DataFormat | System date display format |
| self.TimeFormat | System time display format |
| self.DataTimeFormat | System date and time display format |
| self.MissionIntervalTime | Define task interval |
| self.TestIOFilePath | IO path in test environment |
| self.IOFilePath | IO path in release environment |
| self.LocalizationMsg | Current localized message type |
| self.AddMissionError | Add task failed message |
| self.CurtainResetTipStr | Curtain resetting state |
| self.CurtaionStopTipStr | Curtain stop moving |
| self.CurtaionRunningTipStr | The curtain is moving |
| self.ControlSliderOperationTipStr | Control Curtain Slider Tips |
| self.MissionTitleStr | Current task list tips |
| self.AddMissionBtnStr | Add task button information |
| self.DeleteMissionBtnStr | Delete task button information |
| self.ComfirmBtnStr | Confirm button information |
| self.CancelBtnStr | Cancel button information |

## **5. Main module（MyApp class）**

|  |  |
| --- | --- |
| Attributes | Description |
| self.\_missionList | Current task list |
| self.\_tipTimer | Demo prompt text timer |
| self.\_isTestMode | Is the program a test mode |
| self.\_timeMgr | Time module instance |
| self.\_appDataMgr | AppData module instance |
| self.\_ioMgr | IO module instance |
| self.curtain | Curtain module instance |
|  | |
| Method | Description |
| UpdateDataTimeLabel(self,dataTime): | Update system date and time |
| OnSliderReleased(self) | Control callback method performed after controlling the curtain slider release |
| ShowTip(self,TipStr,waitTime) | After waiting for waitTime, TipStr is displayed |
| OnAddMissionClicked(self) | Add task button to perform actions |
| OnDeleteMissionClicked(self) | Delete task button to perform actions |
| OnConfirmBtnClicked(self) | Confirm button action |
| OnCancelBtnClicked(self) | Cancel button to perform actions |
| ResetMissionOperationPlane(self) | Reset the task dashboard |
| AddMission(self) | Add task |
| RemoveMission(self) | Delete task |
| OnCurtainReseting(self) | Curtain reset callback method |
| OnCurtainStart(self) | Curtain start moving callback method |
| OnCurtainStop(self) | Curtain stop moving callback method |
| OnCurtainPosUpdate(self,pos) | Curtain position update callback method |
| GetIOPath(self) | Get current IO file path |