First. the development steps

1. Environmental construction

(1) Pyqt environment and python environment to build

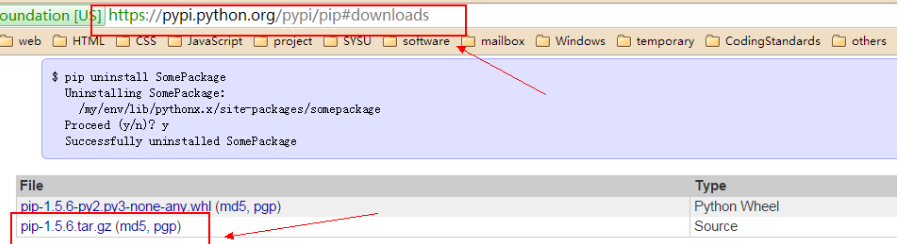
(2) Wilddog-python environment to build

In the development environment of Windows 7 and the running environment of the 410c development board, the development environment of wilddog-python needs to be built as follows:

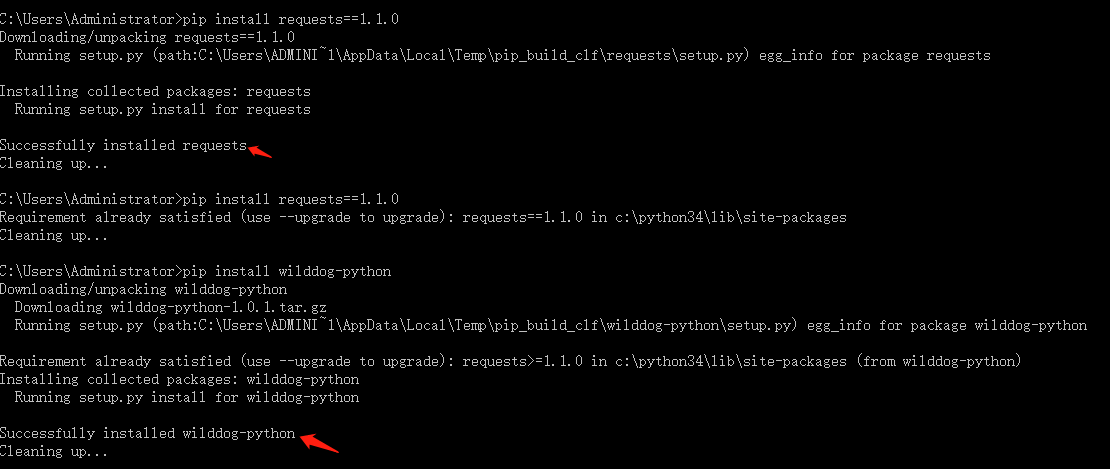
1. Open the cmd command line (open the terminal in the 410c development board), enter the pip list, and the following image appears to indicate that the pip is installed.



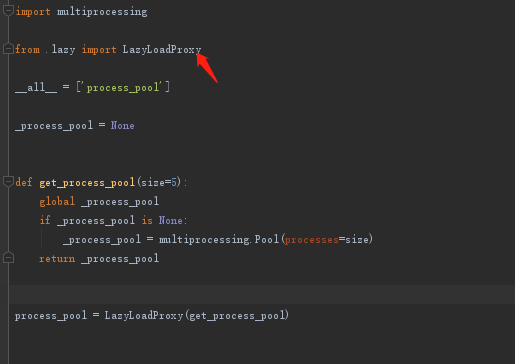
2. If the above image does not appear, you need to reinstall PIP. First download the corresponding version of the PIP file in the https://pypi.python.org/pypi/pip#downloads link, extract the PIP installation package to a folder, and then open it. Terminal, enter the decompressed directory, enter python setup.py install, and then add C:\ Python34 \ Scripts to the computer environment variable;



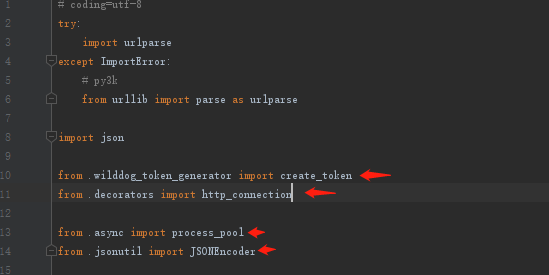
3. After Pip confirms the installation, enter pip install pygatt, pip install requests == 1.1.0 and pip install wilddog-python in the terminal. The following image is displayed indicating that the installation is complete.



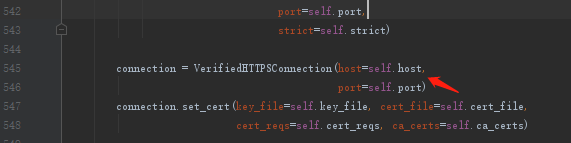
4. Open the async.py file in the path C:\Python34\lib\site-packages\wilddog. Modify from lazy import LazyLoadProxy to from .lazy import LazyLoadProxy



5. Open the wilddog.py file in the path C:\Python34\lib\site-packages\wilddog\, modify from wilddog\_token\_generator import create\_token to from .wilddog\_token\_generator import create\_token, modify from decorators import http\_connection to from .decorators import http\_connection, modify from Async import process\_pool is from .async import process\_pool, modify from jsonutil import JSONEncoder to from .jsonutil import JSONEncoder



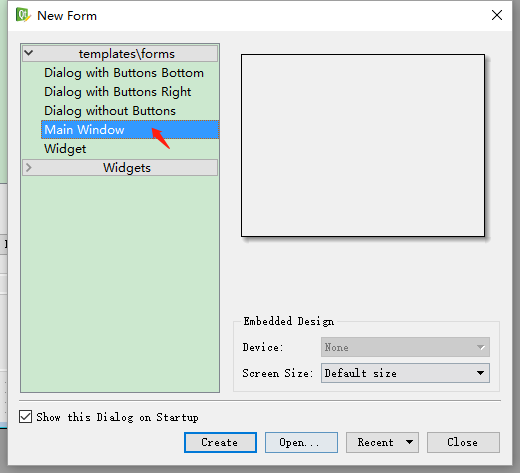
6. Open the connectionpool.py file in the path C:\Python34\lib\site-packages\requests\packages\urllib3\ and modify connection = VerifiedHTTPSConnection (host=self.host,port=self.port,strict=self.strict ) for connection = VerifiedHTTPSConnection (host=self.host,port=self.port)



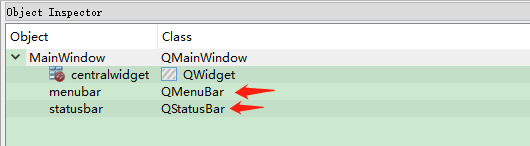
2. Develop UI with Qt Designer

(1) Ui file creation

Open the QtDesigner tool, click FIle->New, select the Main Window option in the pop-up panel, click the Creat button, create a UI file, name it UIControlCenter.ui, and save it to the project directory ControlCenter410c/UI path.

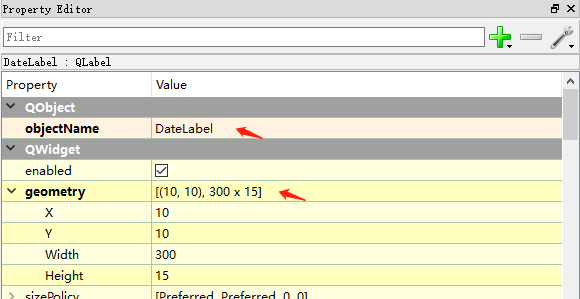


After creating the ui file, right-click on the menubar and statusbar in the ObjectInspector panel and click on the corresponding remove to delete the two objects.

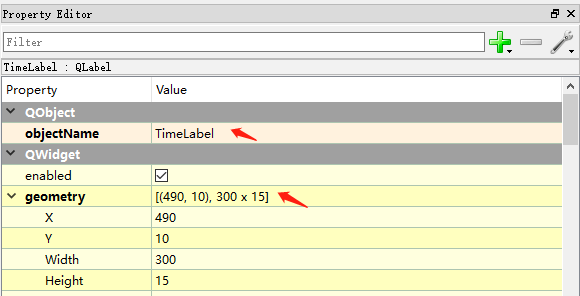


(2) Time and date panel

Locate the Label component in the WidgetBox panel, drag it to the ui interface, select the created Label component in the Object Inspector, and in the PropertyEidtor panel, modify the Label name to DateLabel and modify the geometry as shown below.

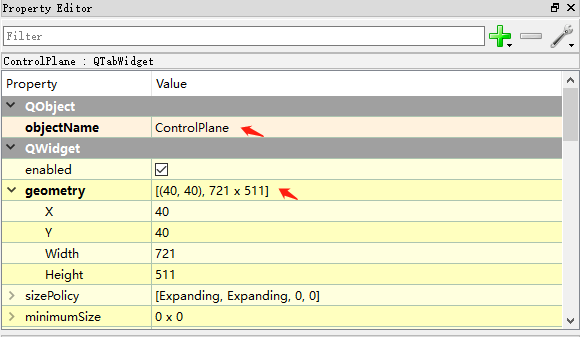


Locate the Label component in the WidgetBox panel, drag it to the ui interface, select the created Label component in the Object Inspector, and in the PropertyEidtor panel, modify the Label name to TimeLabel and modify the geometry as shown below.

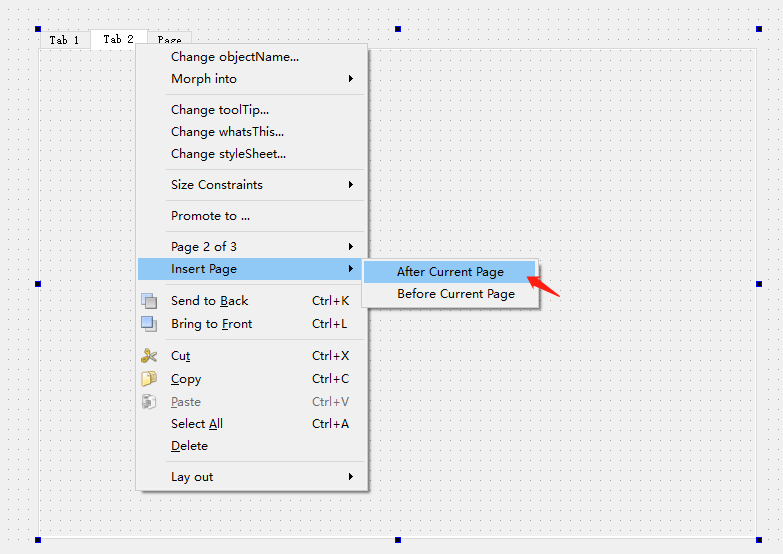


(3) Function module TabWidget panel

Locate the Tab Widget component in the Widget Box panel, drag it into the ui file, find the TabWidget component in the Inspector panel, set its objectName and geometry property values in the Property Editor panel as follows:



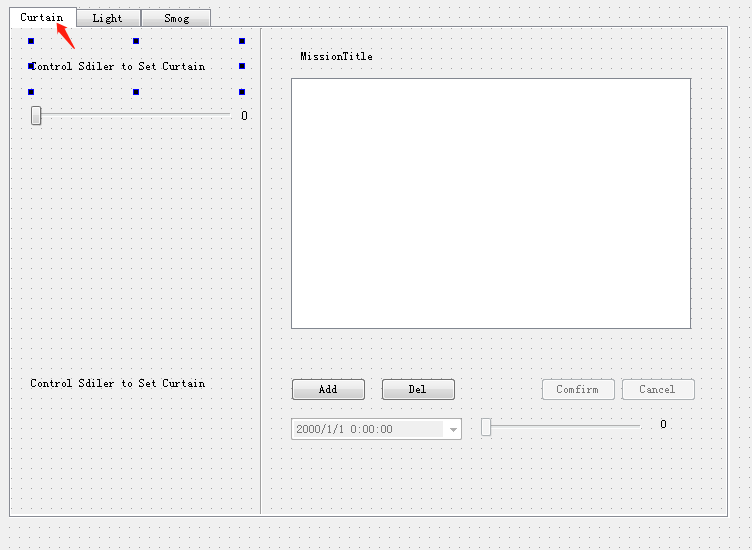
Right-click on the Tab2 button in the ui window and find Insert Package->After Current Package in the pop-up panel.



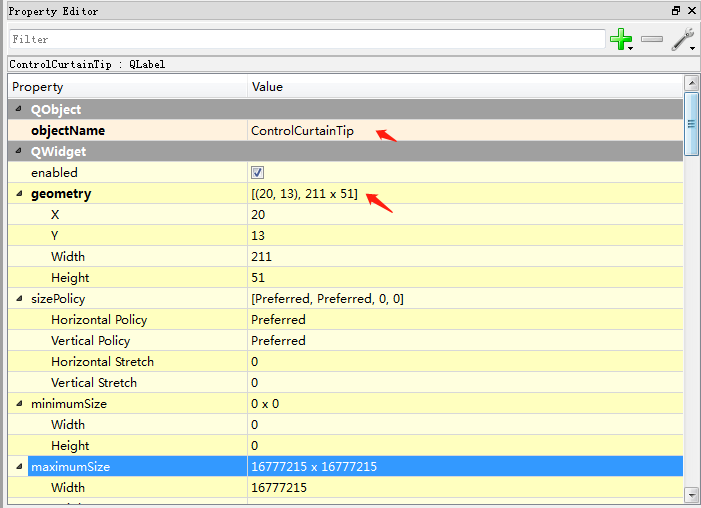
Expand the ControlPlane object in the Object Inspector panel and select the three sub-objects under ControlPlane. In the Property Editor panel, modify the objectname properties to Curtain, Light, and Smog.

(4) Curtain module panel

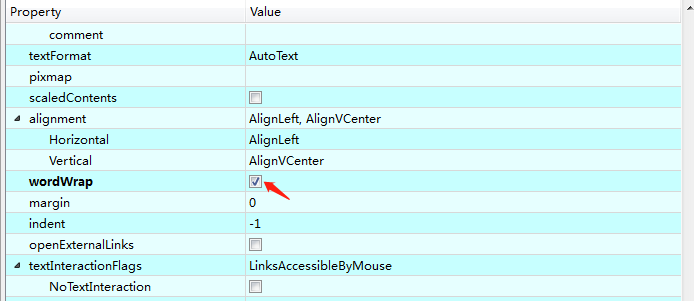
Select the Curtain button Tab form in the Ui form



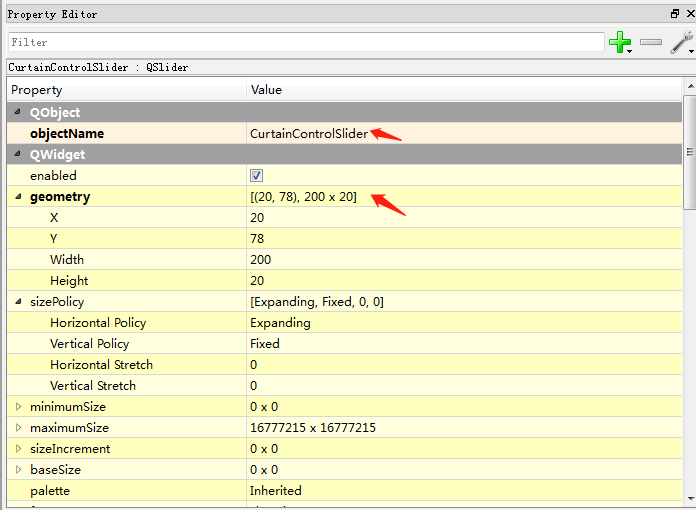
Locate the Label component in the WidgetBox panel and drag it into the Curtain panel. Set its PropertyEditor property as follows:



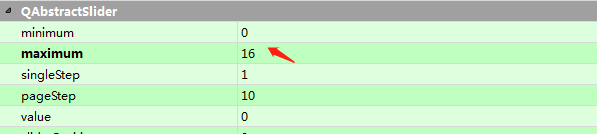
Set the wordWrap property to True



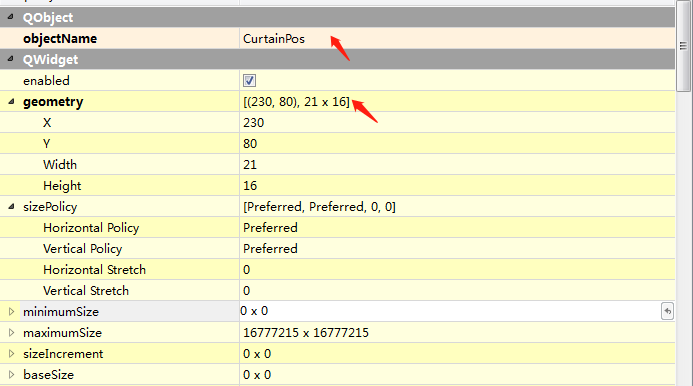
Find the Horizontal Scroll Bar component from the Widget box panel, drag it into the Curtain form, and set its properties as follows:



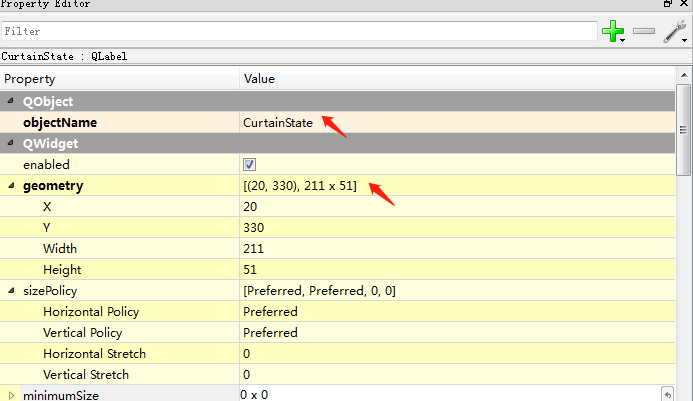
Set the maximum property to 16



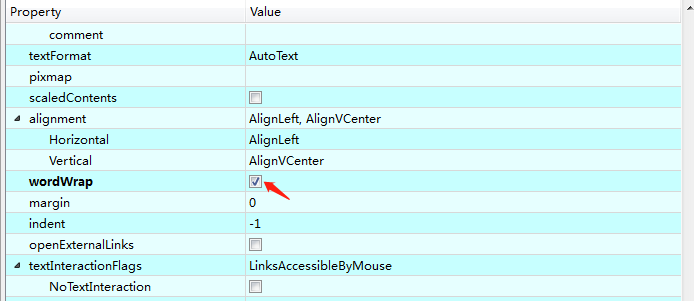
Locate the Label component in the WidgetBox panel, drag and drop it into the Curtain form, and set the properties as follows:



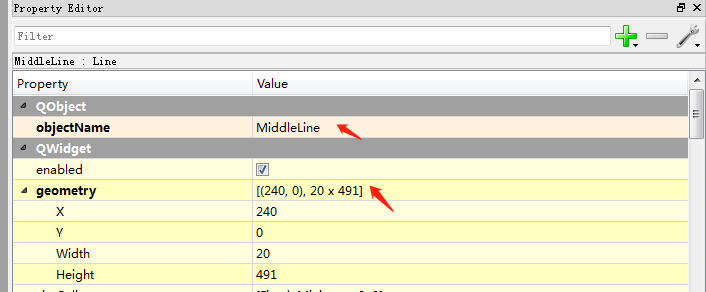
Locate the Label component in the WidgetBox panel, drag and drop it into the Curtain form, and set the properties as follows:



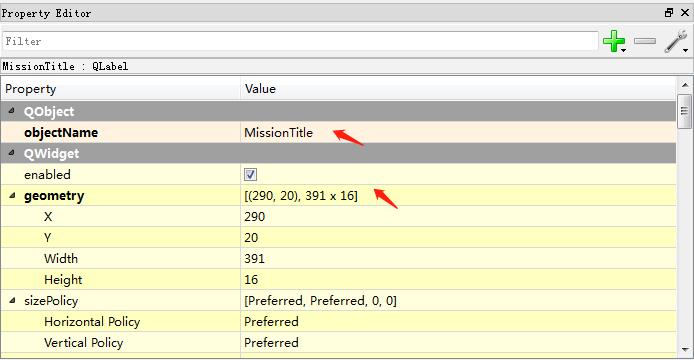
Set the wordWrap property to True



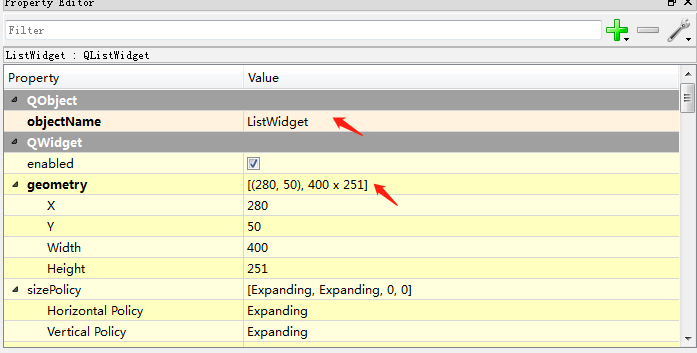
Find the Vertical Line component in the Widget Box panel, drag it into the Curtain form, and set its properties to



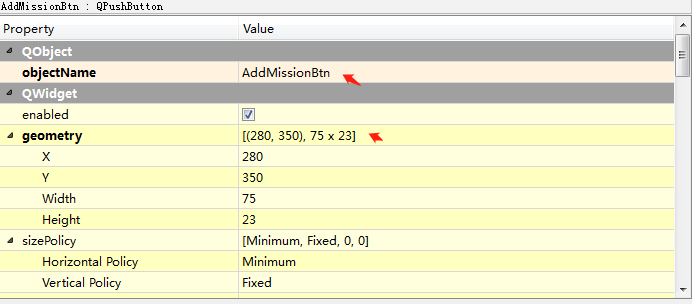
Find the Label component in the Widget Box panel, drag and drop it into Curtain and set its properties to:



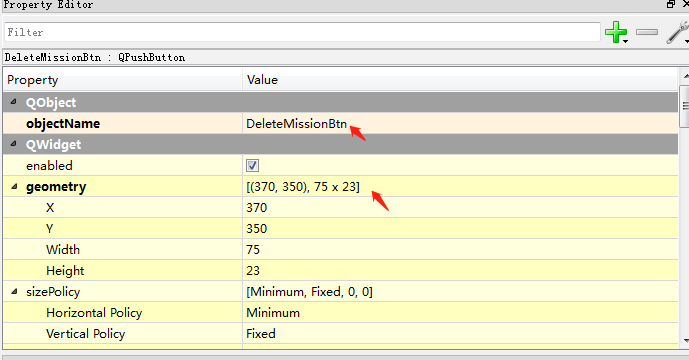
Locate the List Widget component in the Widget Box panel, drag it into the Curtain panel, and set its properties as follows:



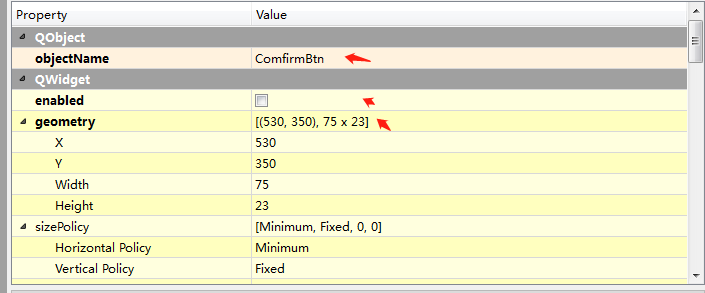
Drag and drop the PushButton component into the Curtain panel in the Widget box panel and set its properties to:



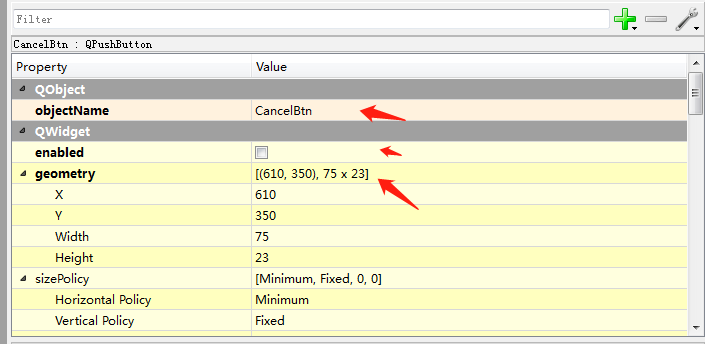
Drag and drop the PushButton component into the Curtain panel in the Widget box panel and set its properties to:



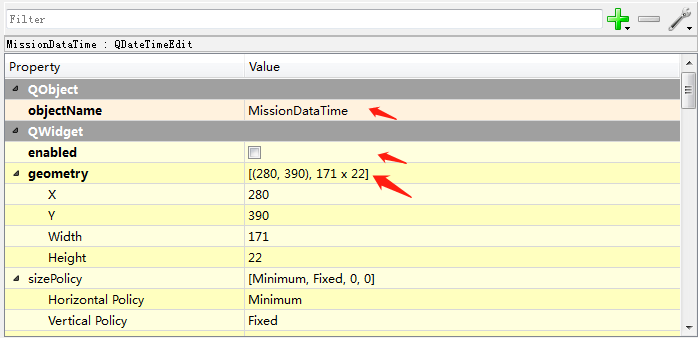
Drag and drop the PushButton component into the Curtain panel in the Widget box panel and set its properties to:



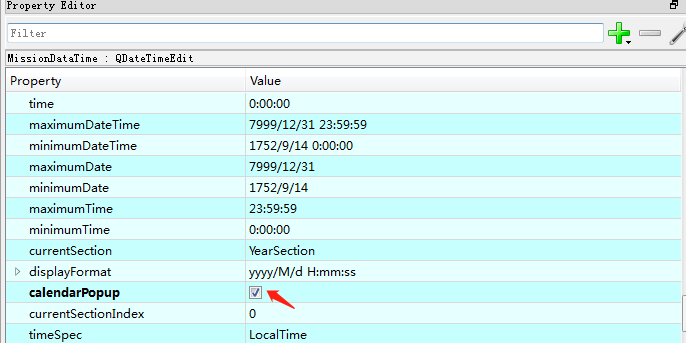
Drag and drop the PushButton component into the Curtain panel in the Widget box panel and set its properties to:



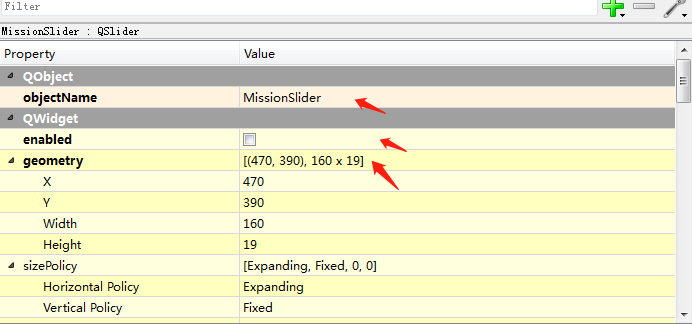
Find the Date/Time Edit component in the Widget Box panel, drag it into the Curtain panel and set its properties as follows:



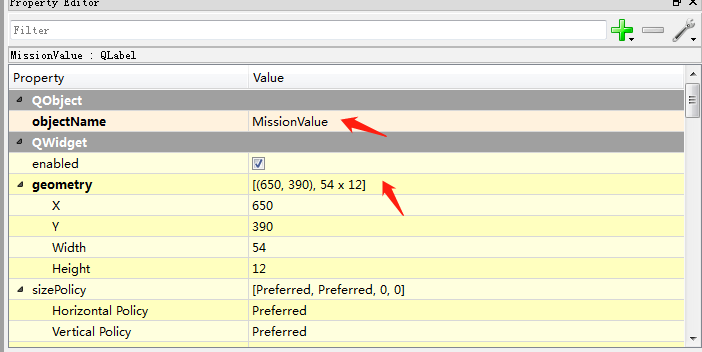
Set its calendarpopup property to True



Find the Horizontal Slider component in the Widget box panel, drag it into the Curtain panel and set its properties as follows:

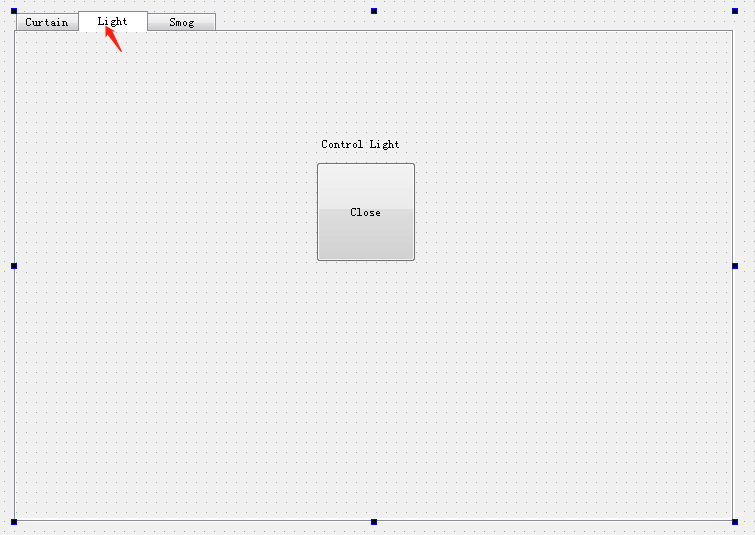


Locate the Label component in the Widget box panel, drag it into the Curtain panel, and set its properties to:

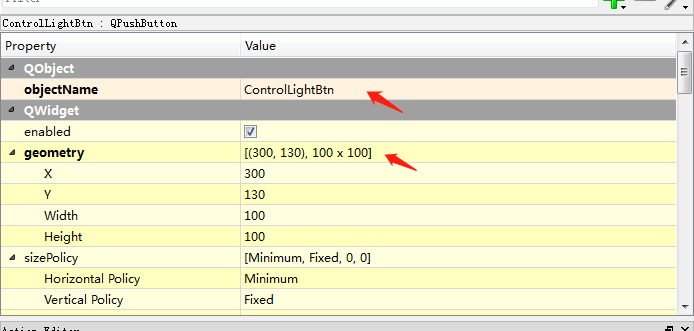


(5) SmartLight module panel

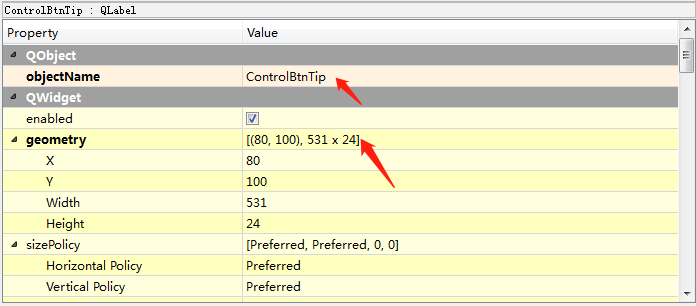
Select the Light button in the UI form and switch to the Light panel.



Find the Push Button component in the Widget box panel, drag and drop it into the Light panel, and set its properties as follows:



Locate the Label component in the Widget Box panel, drag it into the Light panel, and set its properties as follows:

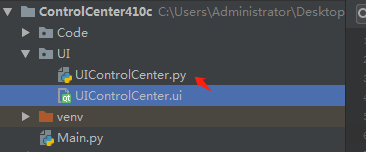


3. Develop functional modules using PyCharm

(1) UI module function development

Open the PyCharm software, open the ControlCenter410c project directory, right click on the ControlCenter410c directory, select New->Directory, create a folder, and modify the folder name to Code.

You can find the created UIControlCenter.ui file in the ControlCenter410c/UI directory, right-click on External Tools->PyUIC, and then generate UIControlCenter.py file in this directory.



(2) Basic module function development

Right-click the ControlCenter410c/Code directory and select New->Directory to create the Common, Curtain, and Light directories.

The Common directory is the control center base module, right-click the Common directory, select New-> Python File, and create AppDataManager.py, IOManager.py, TimeManager.py, and WilddogManager.py files respectively.

The AppDataManager is responsible for managing the data needed in the application.

The IOManager is responsible for the IO related function logic in the application.

TimeManager is responsible for the function logic related to the Time module in the application.

The WilddogManager is responsible for applying the relevant function logic of the Wilddog SDK module.

(3) Curtain related function development

Right-click the Curtain directory and select New->Python File to create the CurtainManager.py and CurtainModule.py files respectively.

CurtainManager is responsible for processing the logic code related to Curtain control.

CurtainModule is responsible for managing the core logic related to Curtain in the application.

(4) Development of Light related function modules

Right-click on the Light directory and select New->Python File to create the BleDeviceManager.py and LightManager.py files respectively.

BleDeviceManager handles the Ble communication related function logic in the Light module.

The LIghtManager handles the related functions of the Light module in the application.

(5) Main module

Right-click the ControlCenter410c directory and select New->Python File to create Main.py. Its main function is to process events in the UI file, manage each module, and provide an entry point for the program to run.

Second. API

1. Common module

(1)AppDataManager

|  |  |
| --- | --- |
| Attributes | Effect |
| self.DataFormat | System date display format |
| self.TimeFormat | System time display format |
| self.DataTimeFormat | System date and time display format |
| self.MissionIntervalTime | Define task time interval |
| self.TestIOFilePath | IO path in the test environment |
| self.IOFilePath | IO path under actual operation |
| self.LocalizationMsg | Currently localized message type |
| self.AddMissionError | Add task failure message |
| self.CurtainResetTipStr | Curtain reset state |
| self.CurtaionStopTipStr | Curtain stop motion |
| self.CurtaionRunningTipStr | Curtain movement |
| self.ControlSliderOperationTipStr | Control curtain slider tips |
| self.MissionTitleStr | Current task list prompt |
| self.AddMissionBtnStr | Add task button information |
| self.DeleteMissionBtnStr | Delete task button information |
| self.ComfirmBtnStr | Confirm button information |
| self.CancelBtnStr | Cancel button information |
| self.LightBtnTextClosed | Light button off status text message |
| self.LightBtnTextOpened | Light button on status text message |
| self.LightTipTextClosed | Light off status message |
| self.LightTipTextOpened | Light on status message |

(2)IOManager

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

(3)TimeManager

|  |  |
| --- | --- |
| Method | Effect |
| GetSystemDataTime(self) | Get current system time |
| GetSystemDataTimeStamp(self) | Get the current system timestamp |
| StartUpdateDataTime(self,callBackFunc) | Start updating system time, execute callBackFunc callback method every second |
| StopUpdateDataTime(self) | Stop updating system time |
| updateTimerCallBackFunc(self) | Callback method for performing update time |

(4) WilddogManager

|  |  |
| --- | --- |
| Method | Effect |
| AddWilddogCallback(self,func) | Add wilddog information callback method |
| RemoveWilddogCallback(self,func) | Delete wilddog information callback method |
| UpdateWilddogNodeValue(self,nodeurl,value) | Update the value of the Wilddog node |
| DeleteWilddogNodeValue(self,nodeurl) | Delete the value of the Wilddog node |
| StartReadUrlLoop(self, loopTime) | Start loop reading the wilddog value |
| StopReadUrlLoop(self) | Stop loop reading the wilddog value |
| \_\_GetUrl(self) | Execute the wilddog callback method |

1. Curtain module

(1)CurtainManager

|  |  |
| --- | --- |
| Attributes | Effect |
| self.Position | Current curtain position |
| self.CurAction | The type of action performed by the current curtain |
| self.CurtainRunTime | The time it takes for the curtain to be closed to open |
| self.MainWindow | Window object in Main.py |
| self.RunningTimer | The timer during the movement of the curtain is executed once per second to calculate the current curtain position. |
| self.CommandTimer | The time it takes for the curtain to perform the action |
| self.CommandStartCallBack | Curtain starts moving callback method |
| self.CommandStopCallBack | Curtain stop moving callback method |
| self.CommandRunningCallBack | Callback method in curtain movement, executed once per second |
| self.CurtainResetingCallBack | Curtain position reset callback method |
| ActionType(Enum) | Curtain execution type |
|  | |
| Method | Effect |
| ControlCurtainWithTargetPos(self,targetPos) | Move the curtain to the specified position targetPos |
| ResetCurtain(self) | Curtain position reset |
| OpenCurtain(self,openLevel) | Open the curtain and move the time of openLevel |
| CloseCurtain(self,closelevel) | Close the curtain and move the time of closelevel |
| StopCurtain(self) | Curtain stops moving |
| OnCurtainRunning(self) | The curtain is moving |
| StopAllTimers(self) | Stop all timers |
| DoCommandStartCallBack(self) | Execute the curtain to start running callback method |
| DoCommandStopCallBack(self) | Perform curtain stop motion callback method |
| DoCammandRunningCallBack(self) | Perform callback method during curtain running |
| DoCurtainResetingCallBack(self) | Perform curtain reset callback method |
| OpenCommand(self) | Operation file execution open curtain command |
| CloseCommand(self) | Operation file execution close curtain command |
| StopCommand(self) | Operation file execution curtain stop motion command |

(2)CurtainModule

|  |  |
| --- | --- |
| Attributes | Effect |
| self.missionList | Current task list |
| self.\_\_controlValueUrl | Wilddog saves the node path of the Curtain location |
| self.\_\_missionListUrl | Wilddog saves the Curtain task list path |
| self.\_\_missionTimeKey | Wilddog saves Curtain task time node |
| self.\_\_missionValueKey | Wilddog saves Curtain task value node |
| Self.\_\_lastMissionList | Locally saved existing task list |
| self.\_\_appDataMgr | AppData module instance |
| self.\_\_wilddogMgr | Wilddog module instance |
| self.\_\_timeMgr | Time module instance |
| self.\_\_curtain | Curtain module instance |
| self.myApp | Main module instance |
|  | |
| Method | Effect |
| GetCurtain(self) | Get the current Curtain instance |
| WilddogCallBack(self,wilddogInfo) | Wilddog module information callback processing |
| OnSliderReleased(self) | Control action after the slider is released |
| ControlCurtainWithTargetPos(self,targetPos) | Move Curtain to targetPos position |
| OnAddMissionClicked(self) | Add task button to perform action |
| OnDeleteMissionClicked(self) | Delete task button to perform action |
| OnConfirmBtnClicked(self) | Confirm button execution |
| OnCancelBtnClicked(self) | Cancel button execution |
| ResetMissionOperationPlane(self) | Reset task operation panel |
| AddMission(self) | Add task |
| RemoveMission(self) | Delete task |
| OnCurtainReseting(self) | Curtain reset callback method |
| OnCurtainStart(self) | Curtain starts moving callback method |
| OnCurtainStop(self) | Curtain stop moving callback method |
| OnCurtainPosUpdate(self,pos) | Curtain position update callback method |

3. Light module

(1)BleDeviceManager

|  |  |
| --- | --- |
| Attributes | Effect |
| Self.device | Current ble device |
| self.adapter | Current Ble device adapter |
|  | |
| Method | Effect |
| stop(self) | Stop adapter |
| scan(self, timeout=5) | Scan the current Ble device |
| connect\_name(self, name, devices=None) | Connect a ble device named name |
| connect(self, address) | Connect the device according to the Ble device address |
| discover\_characteristics(self, device=None) | Find the attribute value of the device Ble communication |
| read\_characteristics(self, uuid, device=None) | Read Ble device attribute values according to uuid |
| write\_characteristics(self, str, uuid, device=None) | Write device attribute values based on uuid |

(2)LightManager

|  |  |
| --- | --- |
| Attributes | Effect |
| self.lightIsOpen | Whether the current light is on |
| self.lightStateCallBack | Light state change callback |
| self.\_\_controlValueUrl | Wilddog saves light state value node |
| self.\_\_myApp | Main module MyApp instance |
| self.\_\_wilddogMgr | Wilddog module instance |
|  | |
| Method | Effect |
| WilddogCallBack(self,wilddogInfo) | Handling Wilddog module callback information |
| ChargeLightControllerState(self) | Switch light status |
| InitLight(self) | Initialize the light state |
| OpenLight(self) | Turn on the light |
| CloseLight(self) | Turn off the light |

4.Main module

(1)Main

|  |  |
| --- | --- |
| Attributes | Effect |
| Self.\_isTestMode | Whether the current app is in test mode |
| self.\_\_timeMgr | Time module instance |
| self.\_\_appDataMgr | AppData module instance |
| self.\_\_sysDataTime | Current system dataTime |
| self.\_\_ioMgr | IOManager module instance |
| self.\_\_wilddogMgr | Wilddog module instance |
| self.\_\_lightModule | LightManager module instance |
| self.\_\_curtainModule | MyCurtainModule module instance |
| self.\_tipTimer | Tip prompt text timer |
|  | |
| Method | Effect |
| GetSysDataTime(self) | Returns the current system DataTime |
| GetAppDataMgr(self) | Returns the current AppData instance |
| GetTimeMgr(self) | Returns the current TimeManager module instance |
| GetIOMgr(self) | Returns the current IOManager module instance |
| GetWilddogMgr(self) | Return to the Wilddog module instance |
| GetIOPath(self) | Return the current IO path |
| OnCurtainSliderReleased(self) | Curtain control slider sliding execution method |
| OnCurtainAddMissionClicked(self) | Add task button execution method |
| OnCurtainDeleteMissionClicked(self) | Delete task button execution method |
| OnCurtainMissionComfirmClicked(self) | Confirm button execution method |
| OnCurtainMissionCancelClicked(self) | Cancel button execution method |
| OnLightControlBtnClicked(self) | Light button execution method |
| ShowTip(self, TipStr, waitTime=3) | Show prompt text |
| UpdateDataTimeLabel(self, dataTime) | Update the date and time of the prompt text |
| UpdateLightState(self,lightIsOpen) | Perform light state monitoring callback |
| if \_\_name\_\_ == "\_\_main\_\_": | Main program entry method |