

Jun 30, 2025 16:02

disk_main.py

Page 1/4

```
1  # controller of disk database program
2  #it is the graphical interface
3  # this will be the overall control of the calory program
4  # it creates a meun, reads in the configuration and
5  # drives the program
6
7
8  import config_disk_db as CD          # get the configuration classs
9
10 import disk_db as DB
11
12
13
14 import os
15 import sys
16 import platform
17 from loguru import logger
18
19 # to import from subdir, create __init__.py in subdir and then you can import
20 # in this case we import module database_connect.py in subdir ui_files
21
22 from ui_files.database_connect import Ui_db_dialog
23
24 from PySide6.QtWidgets import (QApplication,
25                                QFileDialog,
26                                QDialog,
27                                QLabel,
28                                QMainWindow,
29                                QMenu,
30                                QPushButton,
31                                QVBoxLayout,
32                                QWidget)
33 from PySide6.QtGui import QAction, QIcon
34 from PySide6.QtUiTools import QUiLoader
35
36
37
38 loader = QUiLoader()
39 basedir = os.path.dirname(__file__)
40
41
42 class database_dialog(QDialog, Ui_db_dialog):
43     def __init__(self):
44         super().__init__()
45         self.setupUi(self)
46
47
48
49
50
51 class CalMain(QMainWindow):
52     def __init__(self, config_file = None):
53         super().__init__()
54
55
56
57
```

Jun 30, 2025 16:02

disk_main.py

Page 2/4

```
58
59     self.setWindowTitle("Database Control")
60     myLabel = QLabel("Disk Database program vs 1.0")
61     myCloseButton = QPushButton("Close")
62     myCloseButton.clicked.connect(self.close_app)
63     layout= QVBoxLayout()
64     layout.addWidget(myLabel)
65     layout.addWidget(myCloseButton)
66
67
68     widget = QWidget()
69     widget.setLayout(layout)
70
71
72
73     self.setCentralWidget(widget)
74     self.show()
75
76     self.config_file = config_file
77
78     #setup menu
79     menu = self.menuBar()
80
81     file_menu = menu.addMenu("&Action")
82
83
84     # Create a " config" action
85     config_action = QAction(" Config File", self)
86     config_action.setShortcut("Ctrl+F")
87     config_action.setStatusTip("change config file")
88     config_action.triggered.connect(self.SetupConfigNew)
89     file_menu.addAction(config_action)
90
91     # Create a " connect db" action
92     db_action = QAction(" Connect Database", self)
93     db_action.setStatusTip("connect database ")
94     db_action.triggered.connect(self.connect_db)
95     file_menu.addAction(db_action)
96
97     #instantiate configuration
98
99     self.SetupConfig()
100
101     self.SetupLogger()
102
103
104
105
106     #         self.log_level = self.CM.log_level
107     #         self.ingred_table = self.CM.ingred_table
108
109
110     #instantiate the disk_db class
111
112
113     def connect_db(self):
114         db_name = self.CD.db_name
```

Jun 30, 2025 16:02

disk_main.py

Page 3/4

```
115         db_user      = self.CD.db_user
116         db_system     = self.CD.db_address
117         db_pwd        = self.CD.db_pwd
118
119
120         return
121
122     def close_app(self):
123         """Closes the program"""
124
125         logger.info("closing down")
126         self.close()
127
128
129     def SetupConfig(self):
130         mysystem = platform.system()
131
132         # get the config filename
133         # here we do a filedialog
134         if(self.config_file == None or not os.path.isfile(self.config_file)):
135             self.config_file , filter = QFileDialog.getOpenFileName(self,
136                 self.tr("Open Config file"), "~", self.tr("*.json"))
137
138
139         logger.info("config file %s" % self.config_file)
140
141
142
143         self.CD = CD.MyConfig(self.config_file)
144         self.log_level = self.CD.log_level
145         self.log_output = self.CD.log_output
146         #reset self.config_file, so we can change it through the menu
147         #self.config_file = None
148
149
150     def SetupConfigNew(self):
151         """ gest called when one ants to read in a new config file"""
152         mysystem = platform.system()
153
154         # get the config filename
155         # here we do a filedialog
156         self.config_file = None
157         if(self.config_file == None or not os.path.isfile(self.config_file)):
158             self.config_file , filter = QFileDialog.getOpenFileName(self,
159                 self.tr("Open Config file"), "~", self.tr("*.json"))
160
161
162         logger.info("config file %s" % self.config_file)
163
164
165
166         self.CD = CD.MyConfig(self.config_file)
167         self.log_level = self.CD.log_level
168         self.log_output = self.CD.log_output
169         #reset self.config_file, so we can change it through the menu
170         #self.config_file = None
171
```

Jun 30, 2025 16:02

disk_main.py

Page 4/4

```
172
173
174     def SetupLogger(self):
175
176         logger.remove(0)
177         #now we add color to the terminal output
178         logger.add(sys.stdout,
179                     colorize = True, format="<green>{time}</green> {function} {line} {level} <level>{message}</level>" ,
180                     level = "DEBUG")
181
182
183
184
185         fmt = "{time} - {name} - {function} - {line} - {level} - {message}"
186         logger.add('info.log', format = fmt , level = 'INFO', rotation="1 day")
187
188
189         # set the colors of the different levels
190         logger.level("INFO", color = '<black>')
191         logger.level("WARNING", color = '<green>')
192         logger.level("ERROR", color = '<red>')
193         logger.level("DEBUG", color = '<blue>')
194
195         return
196
197
198
199
200
201 if __name__ == "__main__":
202     app = QApplication([])
203     config_file = '/Users/klein/git/diskdb/config/config_disk_db.json'
204     window = CalMain(config_file = config_file )
205     window.show()
206     app.exec()
```