

```

1 #utils.py
2
3 # coding: utf-8
4
5 import os
6 import config
7 from PyQt5 import QtWidgets, QtCore, QtGui
8 import json
9
10
11 class AbstractFunction(object):
12     def move_to_center(self):
13         qtRectangle = self.frameGeometry()
14         centerPoint = QtWidgets.QDesktopWidget().availableGeometry().center()
15         qtRectangle.moveCenter(centerPoint)
16         self.move(qtRectangle.topLeft())
17
18     @classmethod
19     def show_warning_message(
20         cls,
21         message: str,
22         title: str = "Warning",
23         detail: str = None,
24         extra: str = None,
25         parent=None,
26         only_yes: bool = False,
27     ):
28         """show warning msg"""
29         msg_box = QtWidgets.QMessageBox(parent=parent)
30         msg_box.setIcon(QtWidgets.QMessageBox.Warning)
31         msg_box.setText(message)
32         msg_box.setWindowTitle(title)
33         if isinstance(extra, str) and detail:
34             msg_box.setInformativeText(extra)
35         if isinstance(detail, str) and detail:
36             msg_box.setDetailedText(detail)
37         if only_yes is True:
38             msg_box.setStandardButtons(QtWidgets.QMessageBox.Yes)
39             btn_yes = msg_box.button(QtWidgets.QMessageBox.Yes)
40             btn_yes.setText("Yes")
41         else:
42             msg_box.setStandardButtons(QtWidgets.QMessageBox.Yes |
QtWidgets.QMessageBox.No)
43             btn_yes = msg_box.button(QtWidgets.QMessageBox.Yes)
44             btn_yes.setText("Yes")
45             btn_no = msg_box.button(QtWidgets.QMessageBox.No)
46             btn_no.setText("Ignore")
47
48             msg_box.setDefaultButton(QtWidgets.QMessageBox.Yes)
49             msg_box.setEscapeButton(QtWidgets.QMessageBox.No)
50             msg_box.setTextInteractionFlags(QtCore.Qt.TextSelectableByMouse) #
QtCore.Qt.NoTextInteraction
51             r = msg_box.exec_()
52             if r == QtWidgets.QMessageBox.Yes:
53                 return True
54             else:
55                 return False
56
57     @classmethod
58     def show_info_message(
59         cls,

```

```

60     message: str,
61     title="Notification",
62     detail: str = None,
63     extra: str = None,
64     parent=None,
65     only_yes: bool = False,
66 ):
67     """show notification msg"""
68     msg_box = QtWidgets.QMessageBox(parent=parent)
69     msg_box.setIcon(QtWidgets.QMessageBox.Information)
70     msg_box.setText(message)
71     msg_box.setWindowTitle(title)
72     if isinstance(extra, str) and detail:
73         msg_box.setInformativeText(extra)
74     if isinstance(detail, str) and detail:
75         msg_box.setDetailedText(detail)
76     if only_yes is True:
77         msg_box.setStandardButtons(QtWidgets.QMessageBox.Yes)
78         btn_yes = msg_box.button(QtWidgets.QMessageBox.Yes)
79         btn_yes.setText("Yes")
80     else:
81         msg_box.setStandardButtons(QtWidgets.QMessageBox.Yes |
QtWidgets.QMessageBox.No)
82         btn_yes = msg_box.button(QtWidgets.QMessageBox.Yes)
83         btn_yes.setText("Yes")
84         btn_no = msg_box.button(QtWidgets.QMessageBox.No)
85         btn_no.setText("No")
86
87     msg_box.setDefaultButton(QtWidgets.QMessageBox.Yes)
88     msg_box.setEscapeButton(QtWidgets.QMessageBox.No)
89     msg_box.setTextInteractionFlags(QtCore.Qt.TextSelectableByMouse) #
QtCore.Qt.NoTextInteraction
90     r = msg_box.exec_()
91     if r == QtWidgets.QMessageBox.Yes:
92         return True
93     else:
94         return False
95
96 @classmethod
97 def get_last_directory(cls):
98     data = cls.__load_default_config()
99     return data.get("last_dir", config.base_dir)
100
101 @classmethod
102 def save_last_directory(cls, dir_path):
103     if not isinstance(dir_path, str):
104         raise TypeError
105     if not os.path.exists(dir_path):
106         raise FileNotFoundError
107     if not os.path.isdir(dir_path):
108         raise ValueError
109     data = cls.__load_default_config()
110     data["last_dir"] = dir_path
111     cls.__save_default_config(data=data)
112
113 @classmethod
114 def __save_default_config(cls, data: dict):
115     if not isinstance(data, dict):
116         raise TypeError
117     with open(config.app_config_fp, "w", encoding="utf-8") as f:
118         json.dump(data, f)
119

```

```
120 | @classmethod
121 | def __load_default_config(cls):
122 |     data = {}
123 |     if not os.path.exists(config.app_config_fp):
124 |         return data
125 |     if not os.path.isfile(config.app_config_fp):
126 |         return data
127 |     with open(config.app_config_fp, "r", encoding="utf-8") as f:
128 |         data = json.load(f)
129 |     return data
130 |
```