cslsi-07-mueller

- Name: Simon Müller
- Mail: s6simue2@uni-bonn.de
- Group: Me, Myself and I


```
import sys
from PyQt5.QtWidgets import QWidget, QLabel, QLineEdit, QTextEdit, QListView, Q
ListWidget, QPushButton, QSpinBox, QRadioButton, QButtonGroup
from PyQt5.QtWidgets import QApplication, QDialog, QFileDialog
from PyQt5.QtWidgets import QGridLayout, QHBoxLayout, QVBoxLayout, QFormLayout

class SubjectDialog(QDialog):

    def __init__(self, parent, data):
        super().__init__(parent)
        self.name_edit = QLineEdit()
        self.sym_edit = QLineEdit()

        self.age_box = QSpinBox()
        self.age_box.setRange(0, 150)
        self.radio_group = QButtonGroup()
        self.radio1 = QRadioButton('m')
        self.radio2 = QRadioButton('f')
        self.radio3 = QRadioButton('?')

        okay_button = QPushButton('Accept')
        cancel_button = QPushButton('Cancel')

        hbox = QHBoxLayout()
        hbox_radio = QHBoxLayout()
        vbox = QVBoxLayout()
        formbox = QFormLayout()

        hbox.addWidget(okay_button)
        hbox.addWidget(cancel_button)
```

```
        hbox_radio.addWidget(self.radio1)
        hbox_radio.addWidget(self.radio2)
        hbox_radio.addWidget(self.radio3)

        formbox.addRow('Name: ', self.name_edit)
        formbox.addRow('Symptomes: ', self.sym_edit)
        formbox.addRow('Gender: ', hbox_radio)
        formbox.addRow('Age: ', self.age_box)
        formbox.addRow(hbox)

        vbox.addLayout(formbox)

        okay_button.clicked.connect(self.accept)
        cancel_button.clicked.connect(self.reject)

        self.setLayout(vbox)
        self.setGeometry(300, 300, 350, 300)
        self.setWindowTitle('Subject')
        self.show()

    def setData(self, subject):
        # TODO fill out the GUI elements
        self.name_edit.setText(subject[0])
        self.sym_edit.setText(subject[3])
        age = subject[1] if type(subject[1]) == int else 2018
        self.age_box.setValue(2018 - age)

        if subject[2] == 'm':
            self.radio1.toggle()
        elif subject[2] == 'f':
            self.radio2.toggle()
        else:
            self.radio3.toggle()


    def getData(self):
        gender = '?'
```

```
        if self.radio1.isChecked():
            gender = 'm'
        elif self.radio2.isChecked():
            gender = 'f'
        return [self.name_edit.text(), 2018 - self.age_box.value(), gender, sel
f.sym_edit.text()]

class ListWindow(QWidget):
    def __init__(self):
        super().__init__()

        # subject data stored as a list
        #     each subject is also a list containing
        #         name (string)
        #         year of birth (number)
        #         gender (string 'm', 'f' or '?')
        #         symptoms (string)
        self.data = [['Dummy Name', 1900, 'm', 'some symptoms'],
                     ['Other Dummy Name', 1970, 'f', 'other symptoms']]

        # some buttons
        button_add = QPushButton('Add')
        button_edit = QPushButton('Edit')
        button_delete = QPushButton('Delete')
        button_save = QPushButton('Save')
        button_load = QPushButton('Load')

        # horizontal box for the buttons
        hbox = QHBoxLayout()
        hbox.addWidget(button_add)
        hbox.addWidget(button_edit)
        hbox.addWidget(button_delete)
        hbox.addWidget(button_save)
        hbox.addWidget(button_load)

        # TODO connect functions to the buttons,
        # so they are called when the user clicks a button
```

```
    button_edit.clicked.connect(self.onEditClicked)
    button_delete.clicked.connect(self.onDeleteClicked)
    button_add.clicked.connect(self.onAddClicked)
    button_save.clicked.connect(self.onSaveClicked)
    button_load.clicked.connect(self.onLoadClicked)

    # list of the subject's names
    self.list = QListWidget()
    self.updateList()

    # vertical layout: buttons below the list
    vbox = QVBoxLayout()
    vbox.addWidget(self.list)
    vbox.addLayout(hbox)

    self.setLayout(vbox)

    self.setGeometry(300, 300, 350, 300)
    self.setWindowTitle('Subject List')
    self.show()

def updateList(self):
    self.list.clear()

    for d in self.data:
        self.list.addItem(d[0])

def onEditClicked(self):

    # which row is selected/current?
    current_row = self.list.currentRow()
    if current_row < 0:
        return

    # create the subject dialog box
    dlg = SubjectDialog(self, self.data)
```

```
        # fill the dialog box with our data
        dlg.setData(self.data[current_row])

        # don't allow any user input as long as the dialog box is visible
        self.setEnabled(False)
        dlg.setEnabled(True)

        # run the dialog
        if dlg.exec_() == dlg.Accepted:

            # get the edited data from the dialog box
            self.data[current_row] = dlg.getData()
            self.updateList()

        # ok, user input again
        self.setEnabled(True)

    def onAddClicked(self):

        # which row is selected/current?
        current_row = self.list.currentRow()
        if current_row < 0:
            return

        # create the subject dialog box
        dlg = SubjectDialog(self, self.data)

        # fill the dialog box with our data
        dlg.setData(['', '', '', ''])

        # don't allow any user input as long as the dialog box is visible
        self.setEnabled(False)
        dlg.setEnabled(True)

        # run the dialog
        if dlg.exec_() == dlg.Accepted:
            # get the edited data from the dialog box
```

```
            self.data.append(dlg.getData())
            self.updateList()

        # ok, user input again
        self.setEnabled(True)

    def onDeleteClicked(self):
        idx = self.list.selectedItems()
        name = ''
        if len(idx) > 0:
            name = idx[0].text()

        self.data = [d for d in self.data if d[0] != name]
        self.updateList()

    def onSaveClicked(self):
        filename = QFileDialog.getSaveFileName()[0]
        print('Saving to ' + filename)
        assert not 'list.py' in filename

        with open(filename, 'w') as f:
            for entry in self.data:
                entry[1] = str(entry[1])
                f.write(','.join(entry))
                f.write('\n')

    def onLoadClicked(self):
        filename = QFileDialog.getOpenFileName()[0]
        print('Loading from' + filename)
        assert not 'list.py' in filename

        with open(filename, 'r') as f:
            self.data = []
            lines = f.read().split('\n')[:-1]
            for line in lines:
                elms = line.split(',')
                elms[1] = int(elms[1])
```

```
                self.data.append(elms)
        self.updateList()

if __name__ == '__main__':

    app = QApplication(sys.argv)
    ex = ListWindow()
    app.exec_()
```