**Algorithm: ScheduleParser**

Import 🡪 PyQt5 🡪 QtCore, QtWidgets, QtGui

Import 🡪 json

Class 🡨 ScheduleParser

Function 🡨 \_\_init\_\_(self, table, data):

self.table 🡨 table

header 🡨 [['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday']]

OPEN json\_file 🡨 (‘timeslots.json’)

WITHIN json\_file:

self.timeslots 🡨 timeslots 🡨 json.load(json\_file)['timeslots']

END json\_file

self.settings 🡨 settings 🡨 **getSettings()**

header 🡨 *append* 🡨 timeslots[settings['starting\_time']:settings['ending\_time'] + 1]

temporaryData 🡨 []

FOR i in range (0 and settings['ending\_time'] + 1 - settings['starting\_time']):

temporaryData 🡨 append 🡨 ['', '', '', '', '', '']

END FOR loop

self.model 🡨 ScheduleParserModel(header, temporaryData)

table.setModel(self.model)

table.setFocusPolicy(QtCore.Qt.NoFocus)

table.setSelectionMode(QtWidgets.QAbstractItemView.NoSelection)

table.verticalHeader().setSectionResizeMode(QtWidgets.QHeaderView.Fixed)

self.**parseData(data)**

Function 🡨 parseData(self, data):

view 🡨 self.table

model 🡨 self.model

FOR entry in data :

entry['color'] = **colorGenerator()**

FOR instance in entry['instances']:

index = model.index(instance[1], instance[0])

view.setSpan(instance[1], instance[0], instance[2] - instance[1], 1)

item 🡨 QtGui.QStandardItem(entry['text'])

item.setBackground(QtGui.QBrush(QtGui.QColor(\*entry['color'])))

item.setForeground(QtGui.QBrush(QtGui.QColor(\*Utilities.textColor  
 (entry['color']))))

model.setData(index, item)

Function 🡨 subjectGenerator(self):

Print 🡨 self.settings['starting\_time']

END CLASS

Class 🡨 ScheduleParserModel(TableModel.TableModel)

Function 🡨 \_\_init\_\_(self, header, data):

super().\_\_init\_\_(header, data)

Function 🡨 setData(self, index, value, role=None):

IF (index not valid)

THEN Return 🡨 False

ELSE IF (role 🡨 None)

THEN self.data[index.row()][index.column()] 🡨 value

END ELSE IF

self.dataChanged.emit(index, index)

Return 🡨 True

Function 🡨 data(self, index, role):

IF (index not valid or not self.data[index.row()][index.column()])

THEN Return 🡨 QtCore.QVariant()

ELSE IF (role equals QtCore.Qt.TextAlignmentRole)

THEN Return 🡨 QtCore.Qt.AlignCenter

ELSE IF (role equals QtCore.Qt.BackgroundRole)

THEN Return 🡨 self.data[index.row()][index.column()].background()

ELSE IF (role equals QtCore.Qt.ForegroundRole)

THEN Return 🡨 self.data[index.row()][index.column()].foreground()

ELSE IF (role not equals QtCore.Qt.DisplayRole)

THEN Return 🡨 QtCore.QVariant()

END ELSE IF

Return 🡨 self.data[index.row()][index.column()].text()