```
'''
Created on May 28, 2011

@author: neugebauer
'''
from PyQt4 import QtGui, QtCore
from projectCreatorUI import Ui_MainWindow as Dlg
from ProjectAdministrator import AtomProject, pRange, ProjectDatabase, JobDatabase
#from databaseUtilities import *
from KMCHamilton import atomKMC
from job_queue import JobQueue
import utilities as u
import os


class MainWindow(QtGui.QMainWindow, Dlg):
    def __init__(self, project, parent = None):
        print "parent", parent
        QtGui.QMainWindow.__init__(self,parent)
        self.setupUi(self)

        self.StepSpinBox.setMinimum(1)
        self.StepSpinBox.setValue(1)
        self.StartSpinBox.setMaximum(10000.)
        self.StopSpinBox.setMaximum(10000.)
        for i in range(1,3):
            self.NumParaBox.insertItem(i, str(i))



        self.project = project
        if self.project.hamilton == None:
            self.HamiltonInfoLabel.setText("None")
            self.ProjectTypeBox.enabled = False
            self.qDataBox.enabled = False
        else:
            ham = self.project.hamilton
            self.HamiltonInfoLabel.setText(ham.Type+ " "+ham.Version)
            pars = ham.control.paraDict.keys()
            for i,par in enumerate(pars):
                # TODO: select only useful ones, split up arrays, ... (should be done in
                Hamiltonian class)
                #       provide predefined values for start, stop, steps
                self.ParaBox.insertItem(i, par)

            self.ProjectTypeBox.enabled = True
            self.qDataBox.enabled = True

            #TODO: should be connected to a database so that it can be easily extended without
            programming
            if ham.Type == "KMC":
                self.ProjectTypeComboBox.insertItem(1, "KMC")

        self.ProjectViewerButton.setEnabled(False)
```

```
# connections
        self.connect(self.closeButton, QtCore.SIGNAL("clicked(bool)"),
                     self.exitAction)
        self.connect(self.submitButton, QtCore.SIGNAL("clicked(bool)"),
                     self.submit)
        self.connect(self.StartQueueButton, QtCore.SIGNAL("clicked(bool)"),
                     self.startQueue)
        self.connect(self.ShowQueueButton, QtCore.SIGNAL("clicked(bool)"),
                     self.showQueue)
        self.connect(self.InspectHamiltonButton, QtCore.SIGNAL("clicked(bool)"),
                     self.showStructure)
#TODO: relace by Hamilton creator form
        self.connect(self.CreateHamiltonButton, QtCore.SIGNAL("clicked(bool)"),
                     self.createStructure)
        self.connect(self.DBViewerButton, QtCore.SIGNAL("clicked(bool)"),
                     self.showDatabase)
        self.connect(self.ProjectViewerButton, QtCore.SIGNAL("clicked(bool)"),
                     self.showProjectViewer)

    def showDatabase(self, status):
        import DatabaseViewerGUI
        projectDB = ProjectDatabase()
        self.dialog = DatabaseViewerGUI.MainWindow(projectDB)
        self.dialog.show()

    def showProjectViewer(self, status):
        print "show project viewer"
        import ProjectViewerGUI

#         if self.project.projectDB == None:
#             self.projectDB = ProjectDatabase()

        jobDB = JobDatabase("KMC" + str(self.projectID))
        self.dialog = ProjectViewerGUI.MainWindow(jobDB)
        self.dialog.show()

    def showStructure(self, status):
        print "show structure viewer"
        import StructureInspectorGUI
        self.dialog = StructureInspectorGUI.MainWindow(self.project.hamilton.structure)
        self.dialog.show()


    def createStructure(self,status):
        print "create structure button"
        import StructureCreatorGUI
        # must be changed later
        self.dialog = StructureCreatorGUI.MainWindow(self.project.hamilton.structure)
        self.dialog.show()

    def checkQueueDeamon(self):
        #TODO: this routine should be implemented (e.g. by checking for a specific file)
        print "check"
```

```
        try:
            f = open(u.srcDir("queueRunning"),"r")
        except:
            print "No activity (queuing deamon not running)"
            return False

        txt = f.read()
        f.close()
        if txt == "running":
            return True
        else:
            return False

    def stopQueueDaemon(self):
        if self.checkQueueDeamon():
            os.remove(u.srcDir("queueRunning"))
            print "stop", self.checkQueueDeamon()

    def exitAction(self):
#         print "exit ProjectCreator"
        self.stopQueueDaemon()  # TODO: should be called by a separate queue manager
        self.close()
#         QtGui.qApp.quit()


    def submit(self, status):
        steps = self.StepSpinBox.value()
        start = self.StartSpinBox.value()
        stop = self.StopSpinBox.value()

        ind1 = self.ProjectTypeComboBox.currentIndex()
        projectType = self.ProjectTypeComboBox.itemData(ind1,2).toString()
        ind2 = self.ParaBox.currentIndex()
        parameter = str(self.ParaBox.itemData(ind2,2).toString())
        print "pRange: ", start,stop, steps, projectType, parameter, type (parameter)

        projectID = self.project.run(task = projectType,
                                     parameters = pRange(start,stop,steps,symbol=parameter))
        self.projectID = projectID
        self.ProjectViewerButton.setEnabled(True)



    def showQueue(self, status):
        import manageTable as db

        queue = JobQueue()
        queue.dbTable.setTimer(1)

        prog = db.app(0,queue.dbTable)
        prog.MainLoop()

        print "show queue"
```

```
    def startQueue(self,status):
# TODO: before starting queueDeamon check whether it is already active
# otherwise threads will be created that never finish!!
        self.pid = u.runCommand("python " + u.srcDir("runQueueDemon.py"))

#from hamilton import AtomHamilton
if __name__ == '__main__':
    import pickle
    from KMCHamilton import KMC_control

#    kmc = atomKMC(u.DumpDir("kmcTest.dat"))
    kmc = atomKMC(dictProject = {"Name" : "kmcTest.dat"})
    project = AtomProject(kmc, name = "kmcTest.dat") # just a dummy name is needed, should be
    improved
    project.showGUI()
```