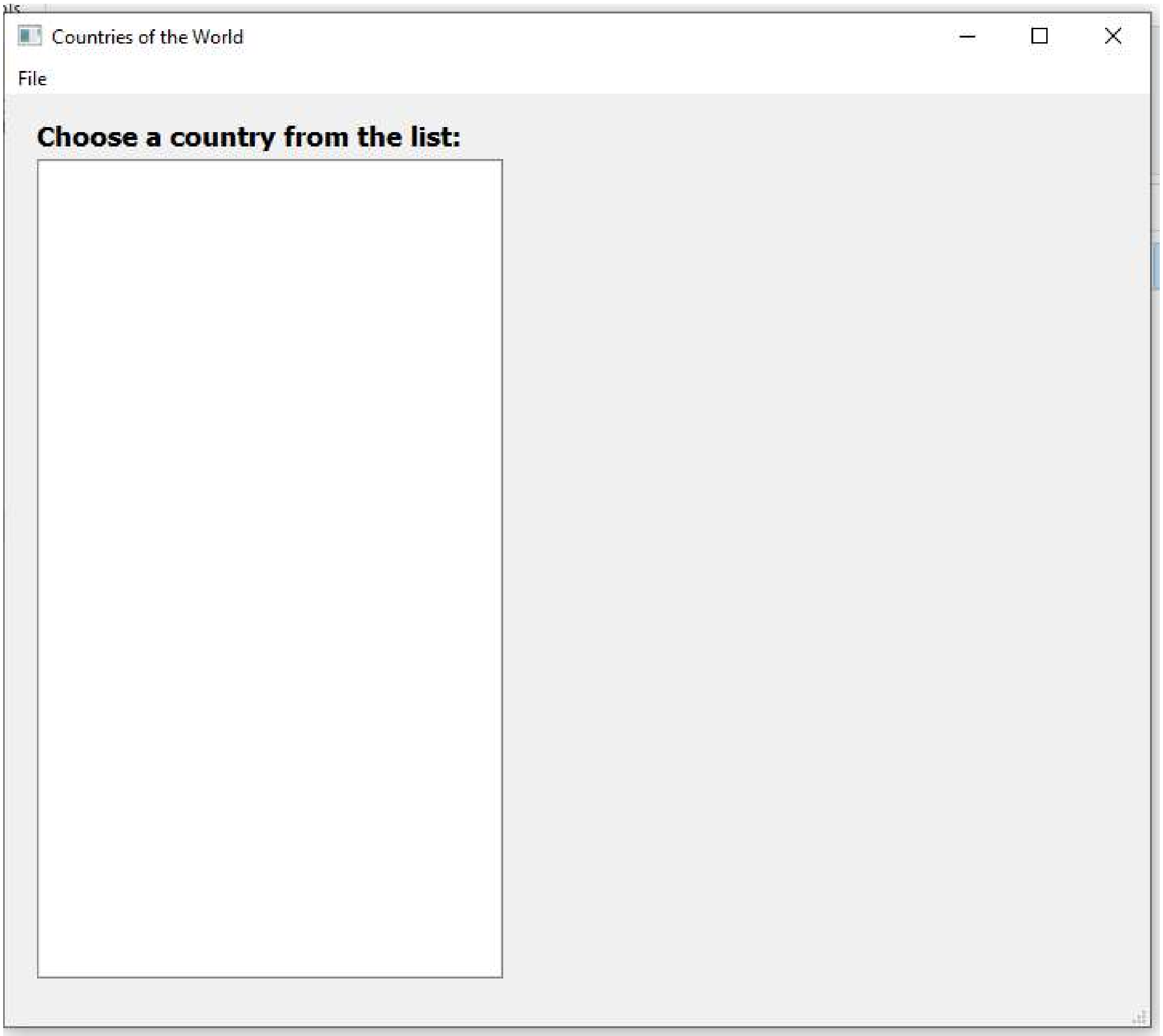
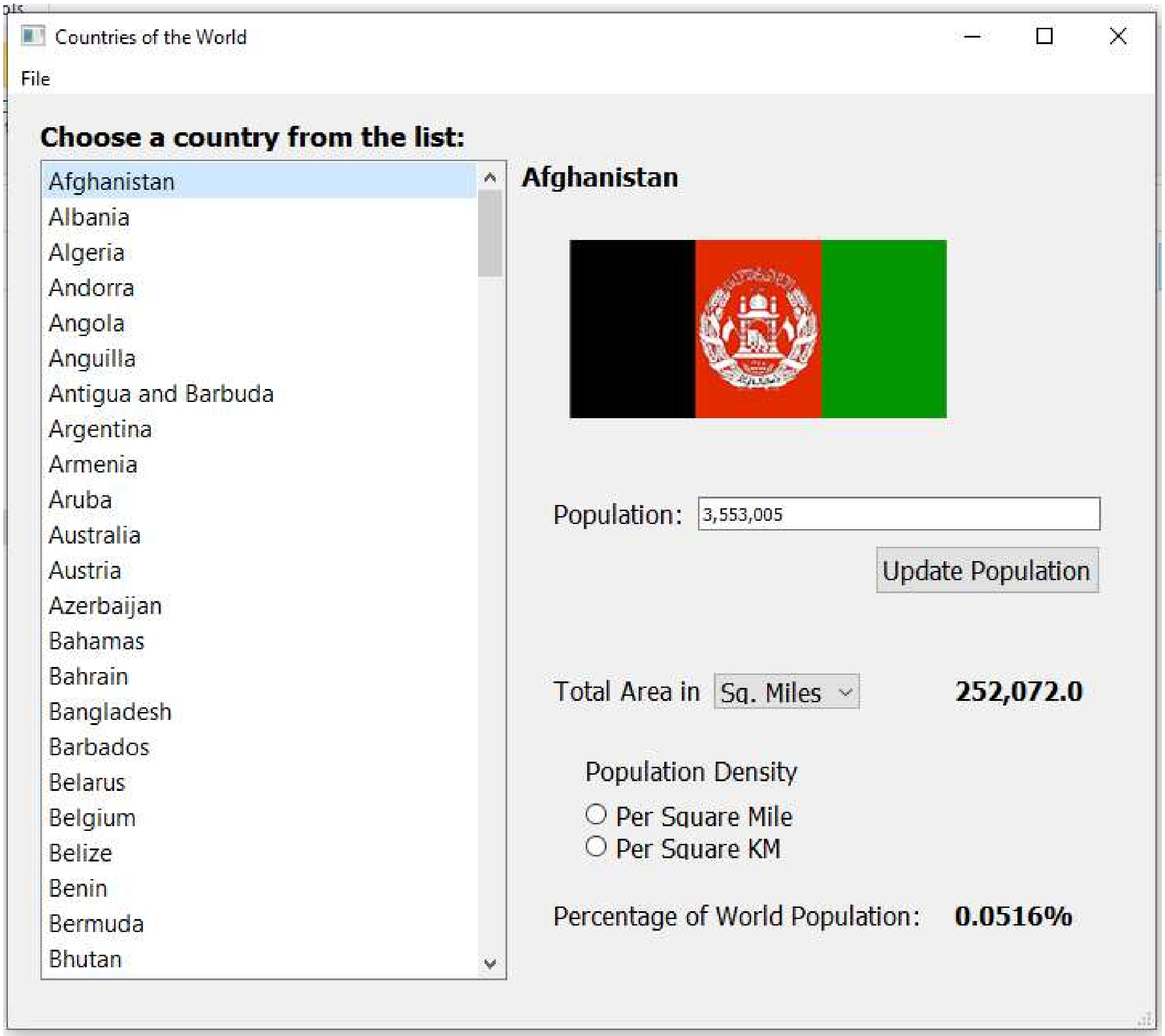


Choose a country from the list:

This program was made with PYQT5. It loads several countries, their population, their total area, their population density, and percentage of the world population. It will also display a flag of the country being viewed.



Choose a country from the list:

- Afghanistan
- Albania
- Algeria
- Andorra
- Angola
- Anguilla
- Antigua and Barbuda
- Argentina
- Armenia
- Aruba
- Australia
- Austria
- Azerbaijan
- Bahamas
- Bahrain
- Bangladesh
- Barbados
- Belarus
- Belgium
- Belize
- Benin
- Bermuda
- Bhutan

Afghanistan

Population: 3,553,005

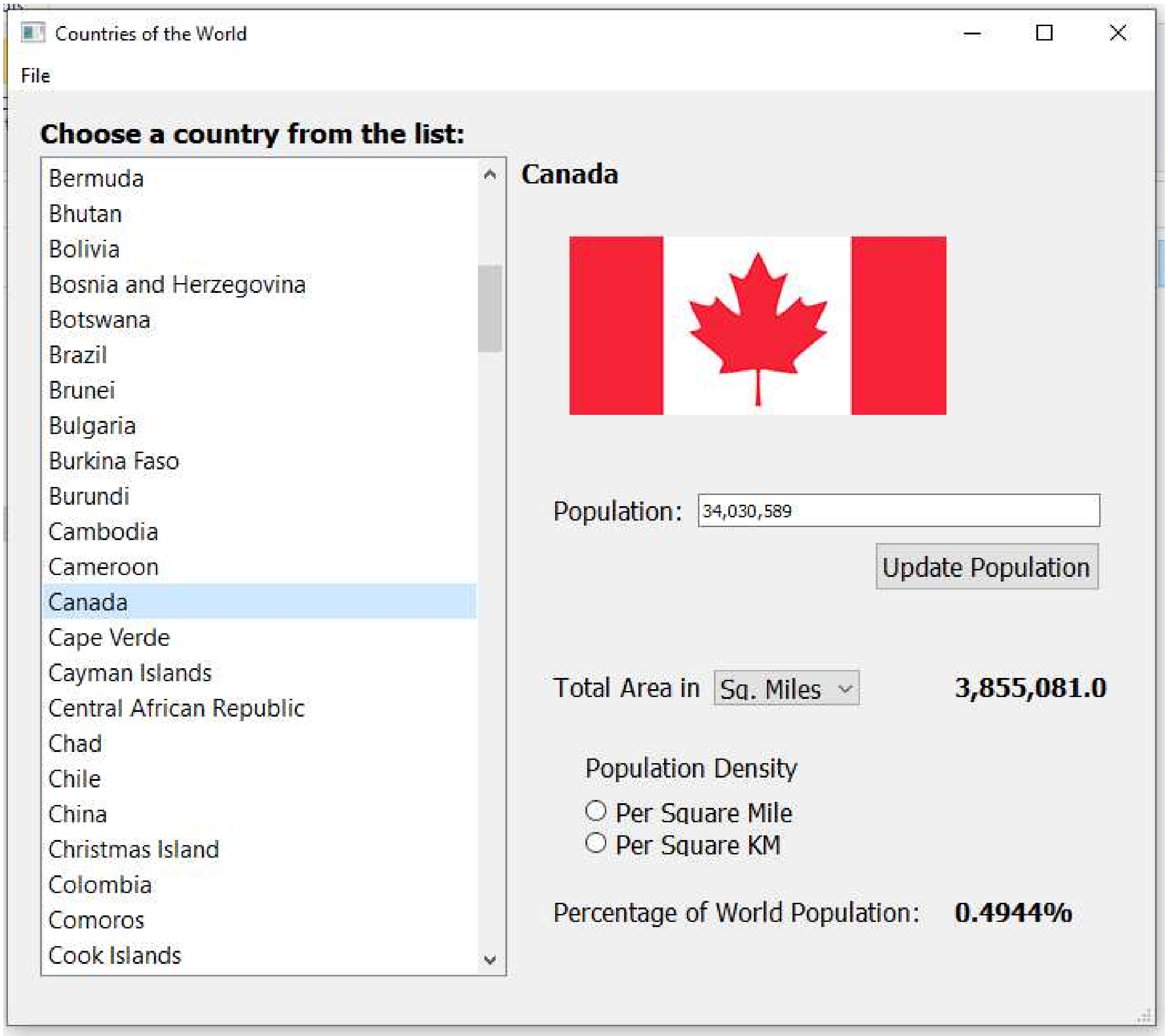
Total Area in Sq. Miles **252,072.0**

Population Density

○ Per Square Mile
○ Per Square KM

Percentage of World Population: **0.0516%**

Once loaded you have the option to change the data between square miles and square kilometers.



Choose a country from the list:

Bermuda
Bhutan
Bolivia
Bosnia and Herzegovina
Botswana
Brazil
Brunei
Bulgaria
Burkina Faso
Burundi
Cambodia
Cameroon
Canada
Cape Verde
Cayman Islands
Central African Republic
Chad
Chile
China
Christmas Island
Colombia
Comoros
Cook Islands

Canada

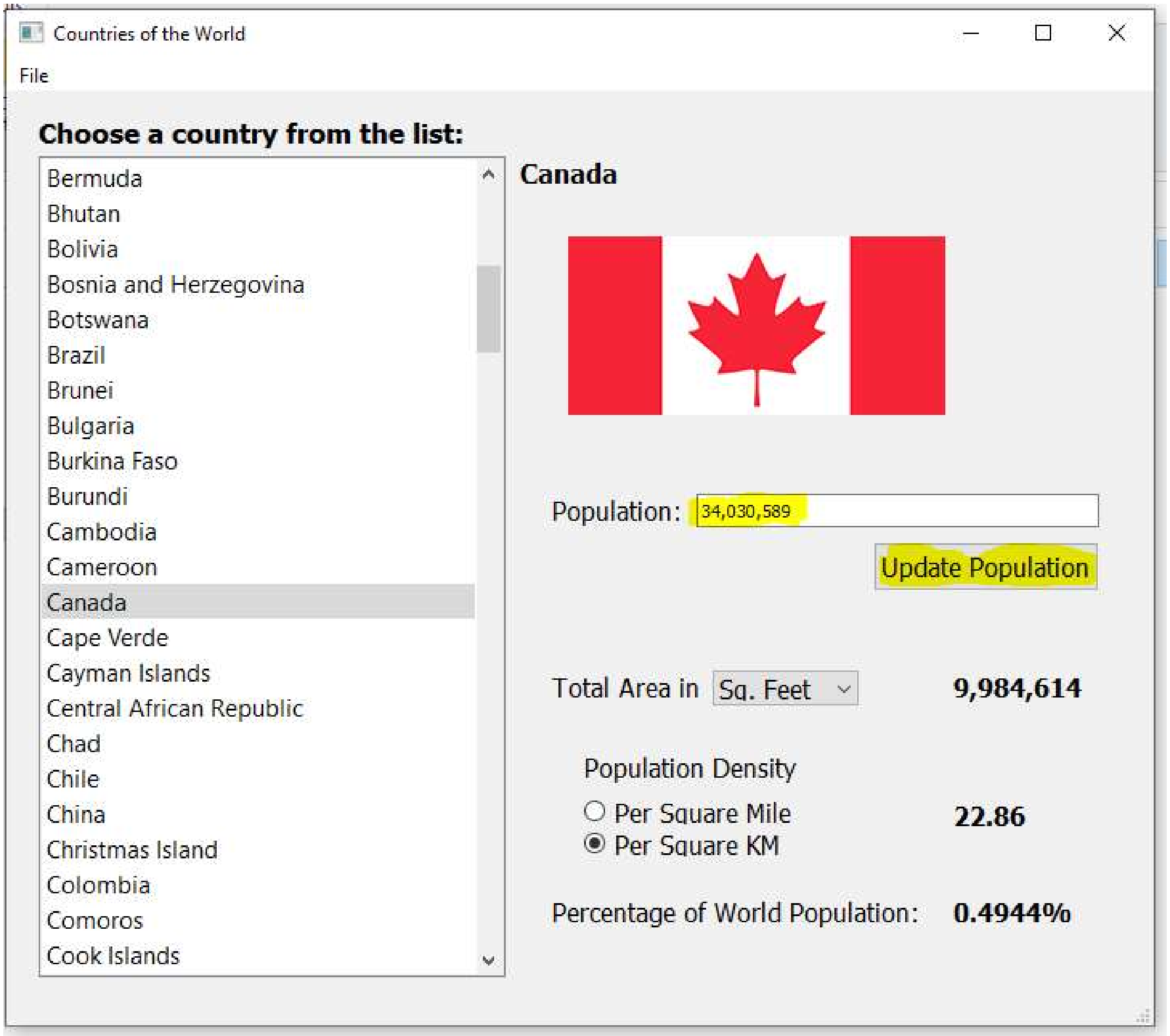
Population: 34,030,589

Total Area in Sq. Miles **3,855,081.0**

Population Density
○ Per Square Mile
○ Per Square KM

Percentage of World Population: **0.4944%**

This screen is showing the data loading with a second country.



Canada

Population: 34,030,589

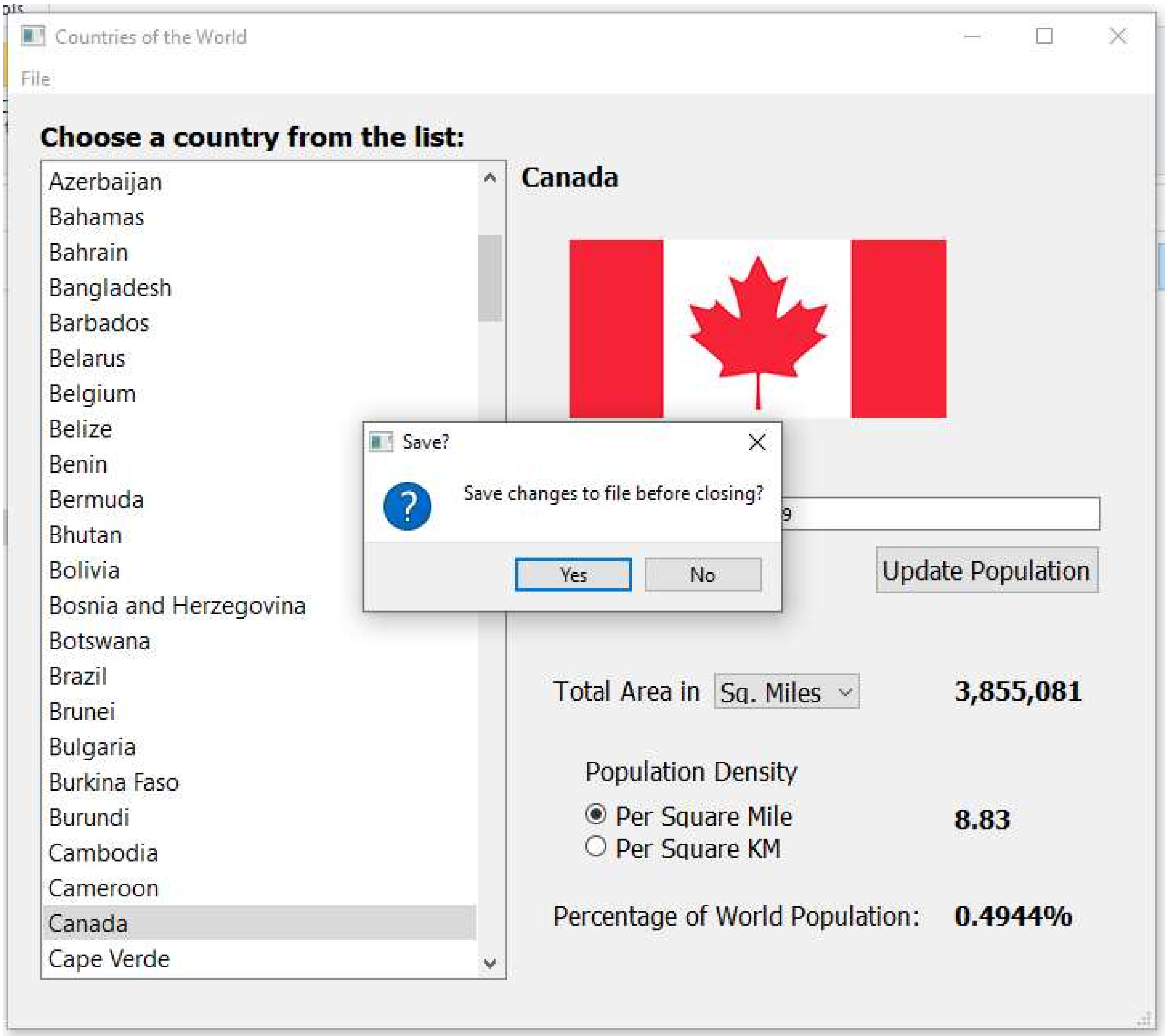
Total Area in Sq. Feet 9,984,614

Population Density

○ Per Square Mile
● Per Square KM 22.86

Percentage of World Population: 0.4944%

This program also fives you the option to update the program data. It will take the information you enter and add commas to the number as required. You can click Update Population to save the data to memory.



Choose a country from the list:

Azerbaijan
Bahamas
Bahrain
Bangladesh
Barbados
Belarus
Belgium
Belize
Benin
Bermuda
Bhutan
Bolivia
Bosnia and Herzegovina
Botswana
Brazil
Brunei
Bulgaria
Burkina Faso
Burundi
Cambodia
Cameroon
Canada
Cape Verde

Canada

Save changes to file before closing?

Total Area in Sq. Miles 3,855,081

Population Density

● Per Square Mile 8.83
○ Per Square KM

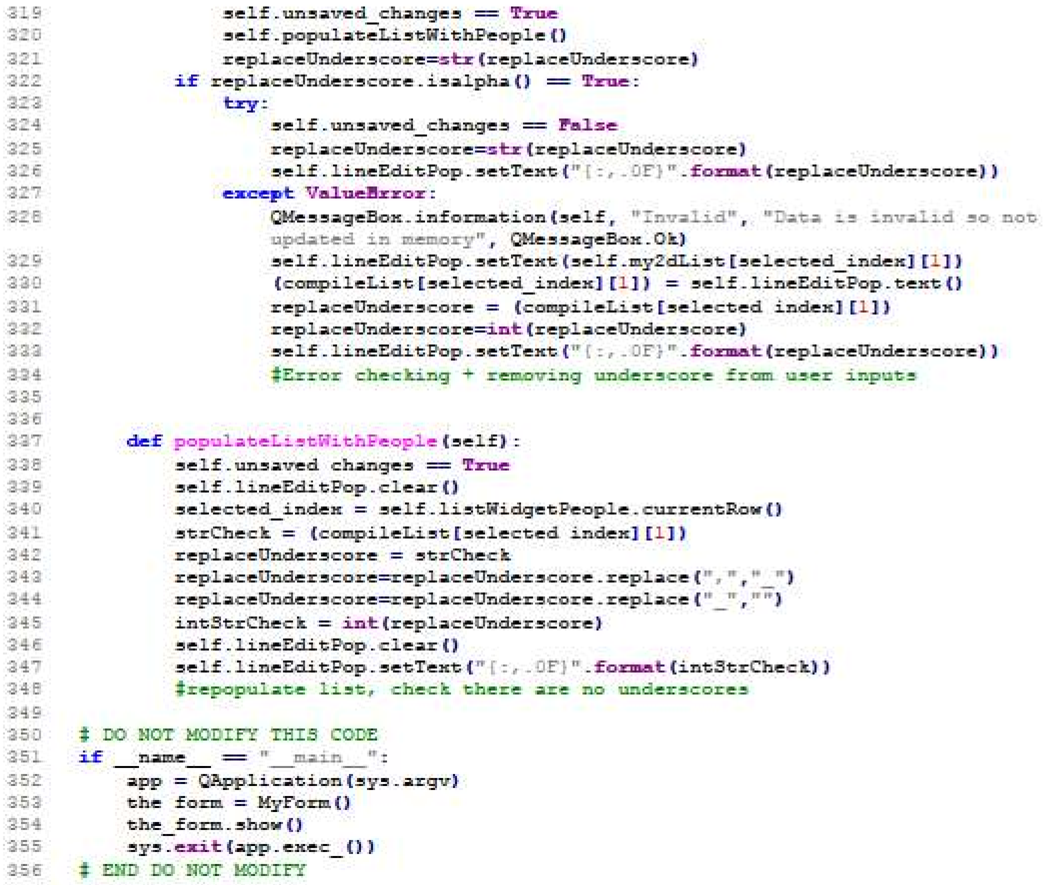
Percentage of World Population: 0.4944%

When you are ready, you can commit the changes to the program permanently, either by going to File > Save or by exiting the program. You can also choose to cancel.

PYW Code:
This program was from the very first programming class I ever took, there for some of this code was provided in the form of a template by my instructor which I then had to modify to make work. The sections my instructor added are commented in the code.

```python
"""
1. Create UI
2. Compile UI as MainWindow, import it.
3. Read data from countries.txt, save it as countries.csv
4. Create a 2dlist out of the CSV values
5. Column 0 is country name, column 1 is country population and column 2 is country square miles.
   Create a menu that reads the CSV file the values from Column 0 in the different rows
6. Store data in memory.
7. Create labels/text boxes to display the country name, population and link it to the proper index.
8. Create a label for Flags and use the ID of the country name from the 2Dlist as a variable to get the
   name of the country. Create a variable and use it when calling the file path. Replacing spaces with underscores
9. Calculate conversion from square miles to square KM and display on screen when selected via radio button.
10. Calculate population percentage by taking the sum of all column 1 values and doing (current pop X 100) / sum of col 1
11. Enable save option in menu to save from menu to countries.txt
12. Add confirmation messages and save/exit confirmation.
Student Name:    Steven Crosby
Program Title:  Countries of the World
Description:    Final Project: Phase 3
"""

import sys

from PyQt5.QtWidgets import QApplication, QMainWindow, QMessageBox
from PyQt5.QtGui import QPixmap

#ADD IMPORT STATEMENT FOR YOUR GENERATED UI.PY FILE HERE
import Ui_FinalProject
#      ^^^^^^^^^^^ Change this!

#CHANGE THE SECOND PARAMETER (Ui ChangeMe) TO MATCH YOUR GENERATED UI.PY FILE
class MyForm(QMainWindow, Ui_FinalProject.Ui_MainWindow):
#                         ^^^^^^^^^^^   Change this!
    def LoadCountriesFromFile(newCSVFile):
        try:#load the file, convert to CSV and create a 2DLIST.
            fileName = "Files/countries.txt"
            accessMode = "r"

            with open(fileName, accessMode) as myTXTFile:
                myTXT=(myTXTFile.read())

            fileName = "Files/countries.csv"
            accessMode = "w"

            with open(fileName, accessMode) as newCSVFile:
                newCSVFile.write(myTXT)
        except FileNotFoundError:
            fileName = "Files/countries.txt"
            accessMode = "w"

            with open(fileName, accessMode) as myTXTFile:
                data = "Error Reading from File Path,0,0"
                myTXTFile.write(data)
                myTXT = data

                fileName = "Files/countries.csv"
                accessMode = "w"
            with open(fileName, accessMode) as newCSVFile:
                 newCSVFile.write(myTXT)
                 return myTXT
    newCSVFile = ""
    myTXT = LoadCountriesFromFile(newCSVFile)
    people = []
    my2dList = []
    unsaved_changes = False
    # DO NOT MODIFY THIS CODE
    def __init__(self, parent=None): #Runs when form loads, before user sees anything
        super(MyForm, self).__init__(parent)
        self.setupUi(self)
    # END DO NOT MODIFY


        # ADD SLOTS HERE, indented to this level (ie. inside def __init__)
        self.frameCountries.setVisible(False)
        self.listWidgetPeople.currentRowChanged.connect(self.DisplayCountryData)
        self.listWidgetPeople.currentRowChanged.connect(self.ShowImage)
        self.actionLoadCountries.triggered.connect(self.LoadCountriesListBox)
        self.actionSavetoFile.triggered.connect(self.saveNoExit)
        self.actionExit.triggered.connect(self.exitApplication)
        self.pushButtonUpdate.clicked.connect(self.updatePeople)
        self.comboBoxMiFt.currentIndexChanged.connect(self.CalculateTotalWorldPopulation)
        self.comboBoxMiFt.addItem("Sq. Miles")
        self.comboBoxMiFt.addItem("Sq. Feet")
        self.radioButtonMile.clicked.connect(self.radioButtonMileClicked)
        self.radioButtonKM.clicked.connect(self.radioButtonKMClicked)

    # ADD SLOT FUNCTIONS HERE
    # These are the functions your slots will point to
    # Indent to this level (ie. inside the class, at same level as def __init__)
    def LoadCountriesListBox(self):
        self.frameCountries.setVisible(True)
        if len(self.my2dList) == 0:
            self.LoadCountriesFromFile()

        self.listWidgetPeople.clear()
        for row in self.my2dList:
            self.listWidgetPeople.addItem(row[0])
            #Add country names in 2Dlist to list widget
        self.sumList = []
        self.sumofSumList = []
        global compileList
        compileList = self.my2dList
        selectedCountryIndex = self.listWidgetPeople.currentRow()
        updatePop = (str(compileList[selectedCountryIndex][1]))
        try:#Check for errors in the list/if file is non existant
            updatePop = int(updatePop)
        except ValueError:
            QMessageBox.information(self, "Error", "Error reading from the file path. Closing program.", QMessageBox.Ok)
            exit()
        except:
            QMessageBox.information(self, "Error", "Error reading from the file path. Closing program.", QMessageBox.Ok)
            exit()

        self.lineEditPop.setText("{:,.0F}".format(updatePop))
        total = 0
        indexLoop = 0
        for row in compileList:
            self.sumList.append(compileList[indexLoop][1])
            indexLoop = indexLoop + 1
        # appending the values of column 1 to its own list to sum
        for counter in range(0, len(self.sumList)):
            total = total + int(str(self.sumList[counter]))
            self.popTotal=str(total)

    def CalculateTotalWorldPopulation(self):
        fileName = "Files/countries.csv"
        accessMode = "r"
```

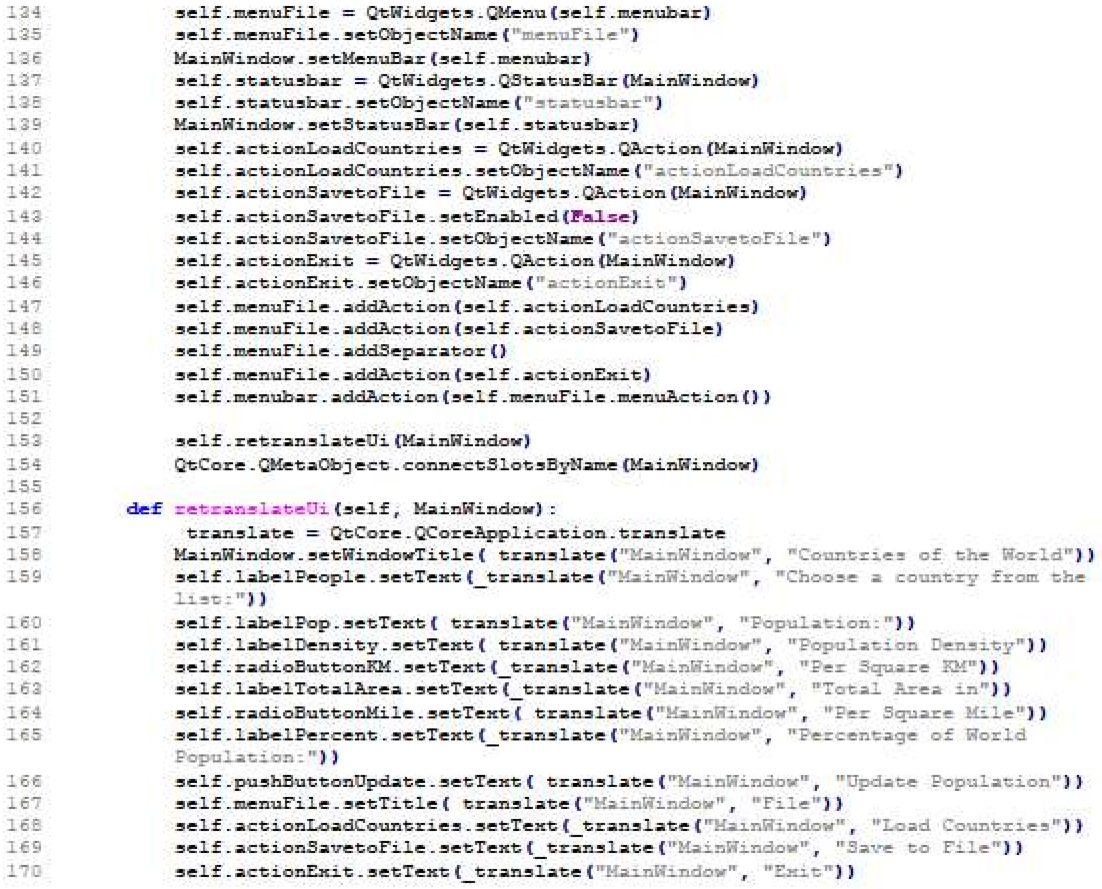
```python

        with open(fileName, accessMode) as myCSVFile:
            import csv
            fileData = csv.reader(myCSVFile)

            self.my2dList = []
            for row in fileData:
                self.my2dList.append(row)
            comboIndex = self.comboBoxMiFt.currentIndex()
            try:
                if comboIndex == 1:
                    selectedCountryIndex = self.listWidgetPeople.currentRow()
                    convertKM1 = float(self.my2dList[selectedCountryIndex][2])
                    convertKM2 = float(2.58998811)
                    convertTotal= (convertKM1 * convertKM2)
                    self.labelEditSqm.setText("{:,.0F}".format(convertTotal))
                elif comboIndex == 0:
                    selectedCountryIndex = self.listWidgetPeople.currentRow()
                    convertTotal = float(self.my2dList[selectedCountryIndex][2])
                    self.labelEditSqm.setText("{:,.0F}".format(convertTotal))
            except ValueError:
                convertTotal = 1
                self.labelEditSqm.setText("{:,.0F}".format(convertTotal))
                #Square Mile to KM conversion + error checking



    def DisplayCountryData(self, selectedCountryIndex):
        OneHundred = 100
        convertPop=(compileList[selectedCountryIndex][1])
        try:
            convertPop = float(convertPop)
            self.lineEditPop.clear()
            self.lineEditPop.setText("{:,.0F}".format(convertPop))
            self.lineEditPop.update()
        except ValueError:
            value = self.listWidgetPeople.currentItem()
            if value == "Error Reading from File Path":
                QMessageBox.information(self, "Invalid", "Data is invalid so not updated in memory", QMessageBox.Ok)
                exit()
            else:
                QMessageBox.information(self, "Invalid", "Data is invalid so not updated in memory", QMessageBox.Ok)
        try:
            popLabel = (convertPop)/float(self.popTotal)*float(OneHundred)
        except AttributeError:
            exit()
        except TypeError:
            QMessageBox.information(self, "Invalid", "Data is invalid so not updated in memory", QMessageBox.Ok)
        except ZeroDivisionError:
            QMessageBox.information(self, "Error", "Error reading from the file path. Closing program.", QMessageBox.Ok)
            exit()
        self.labelCountry.clear()
        self.labelCountry.setText(str(compileList[selectedCountryIndex][0]))
        self.labelCountry.update()
        self.labelEditSqm.clear()
        MiFeetUpdate= (compileList[selectedCountryIndex][2])
        try:
            MiFeetUpdate = float(MiFeetUpdate)
        except ValueError:
            MiFeetUpdate = 0.0
            QMessageBox.information(self, "Error", "Error reading from the file path. Closing program.", QMessageBox.Ok)
            exit()
        self.labelEditSqm.setText("{:,.1F}".format(MiFeetUpdate))
        self.labelEditSqm.update()
        MiFeetUpdate = str(MiFeetUpdate)
        MiFeetUpdate = MiFeetUpdate.replace(","," ")
        MiFeetUpdate = MiFeetUpdate.replace("_","")
        self.labelEditPopulation.clear()
        try:
            self.labelEditPopulation.setText("{:.4f}%".format(popLabel))
            self.labelEditPopulation.update()
        except UnboundLocalError:
            QMessageBox.information(self, "Invalid", "Data is invalid so not updated in memory", QMessageBox.Ok)
            #population percentage calculations and program \
            # refresh after value update


    def radioButtonMileClicked(self, enabled):
        selectedCountryIndex = self.listWidgetPeople.currentRow()
        squareMi1 = float(compileList[selectedCountryIndex][1])
        squareMi2 = float(compileList[selectedCountryIndex][2])
        squareMiTotal= (squareMi1) / (squareMi2)
        if enabled:
            self.comboBoxMiFt.currentIndexChanged.connect(self.populateListWithPeople)
            self.labelEditDensity.clear()
            self.labelEditDensity.setText("{:.2f}".format(squareMiTotal))
            self.listWidgetPeople.currentRowChanged.connect(self.radioButtonMileClicked)
            self.listWidgetPeople.currentRowChanged.connect(self.radioButtonKMClicked)
        #Radio button conversion
    def radioButtonKMClicked(self, enabled):
        selectedCountryIndex = self.listWidgetPeople.currentRow()
        squareKM1 = float(compileList[selectedCountryIndex][1])
        squareKM2 = float(2.58998811)
        squareKM3 = float(compileList[selectedCountryIndex][2])
        squareKMTotal= (squareKM1 * squareKM2) / (squareKM3)
        if enabled:
            self.labelEditDensity.clear()
            self.labelEditDensity.setText("{:.2f}".format(squareKMTotal))
            self.listWidgetPeople.currentRowChanged.connect(self.radioButtonMileClicked)
            self.listWidgetPeople.currentRowChanged.connect(self.radioButtonKMClicked)

    #ADD HELPER FUNCTIONS HERE
    # These are the functions the slot functions will call, to
    # contain the custom code that you'll write to make your progam work.
    # Indent to this level (ie. inside the class, at same level as def __init__)

    def ShowImage(self):
        fileName = "Files/countries.csv"
        accessMode = "r"
        with open(fileName, accessMode) as imageCSVFile:
            import csv
            imageData = csv.reader(imageCSVFile)     #KIND OF gives me a 2d list of the file's contents

            self.imageList = []
            for row in imageData:
                self.imageList.append(row)
        index = self.listWidgetPeople.currentRow()
        printIndex=(self.imageList[index][0])
        printIndex = printIndex.replace(" ", "_")
        image=QPixmap("Files/Flags/"+str(printIndex)+".png")
        self.labelPicture.clear()
        self.labelPicture.setPixmap(image)
        #get proper flag image loaded

    def closeEvent(self, event):
        self.unsaved_changes == True
        quit_msg = "Save changes to file before closing?"
        reply = QMessageBox.question(self, 'Save?',
                quit_msg, QMessageBox.Yes, QMessageBox.No)

        if reply == QMessageBox.Yes:
            self.unsaved_changes == False
            self.actionSavetoFile.setEnabled(False)
            myFile=""
            self.SaveCountriesToFile(myFile)
            QMessageBox.information(self, 'Saved', 'Changes were saved to the file',
            QMessageBox.Ok)
        else:
            exit()
            #Closing (click the X)

    def saveNoExit(self, event):
        self.unsaved_changes == True
        quit_msg = "Save changes to file?"
        reply = QMessageBox.question(self, 'Save?',
                quit_msg, QMessageBox.Yes, QMessageBox.No)
        if reply == QMessageBox.Yes:
            myFile=""
            self.SaveCountriesToFile(myFile)
            self.actionSavetoFile.setEnabled(False)
        if reply == QMessageBox.No:
            self.unsaved_changes == False
            pass
        #(Save menu)

    def exitApplication(self, event):
        self.unsaved_changes == True
        quit_msg = "Save changes to file before closing?"
        reply = QMessageBox.question(self, 'Save?',
                quit_msg, QMessageBox.Yes, QMessageBox.No)
        if reply == QMessageBox.Yes:
            myFile=""
            self.SaveCountriesToFile(myFile)
            exit()
        if reply == QMessageBox.No:
            self.unsaved_changes == False
            exit()
            pass
        # File > Exit

    def SaveCountriesToFile(self, myFile):
        with open("Files/countries.txt", "w") as myFile:
            for person in compileList:
                finalPrint = person[1].replace(" ","")
                selected_index = self.listWidgetPeople.currentRow()
                finalPrint=compileList[selected_index][1]
                finalPrint = self.lineEditPop.text()
                myFile.write(",".join(person) + "\n")
                self.unsaved_changes == False
                #Saving function

    def updatePeople(self):
        selected_index = self.listWidgetPeople.currentRow()
        replaceUnderscore = self.lineEditPop.text()
        replaceUnderscore=replaceUnderscore.replace(",","_")
        replaceUnderscore=replaceUnderscore.replace("_","")
        if replaceUnderscore.isalpha() != True:
            self.lineEditPop.setText(replaceUnderscore)
            (compileList[selected_index][1]) = self.lineEditPop.text()
            replaceUnderscore=int(replaceUnderscore)
            self.lineEditPop.setText("{:,.0F}".format(replaceUnderscore))
            QMessageBox.information(self, 'Updated', "Data has been updated in memory,
            but hasn't been updated in the file yet.", QMessageBox.Ok)
            self.actionSavetoFile.setEnabled(True)
```



```
            self.unsaved_changes == True
            self.populateListWithPeople()
            replaceUnderscore=str(replaceUnderscore)
        if replaceUnderscore.isalpha() == True:
            try:
                self.unsaved_changes == False
                replaceUnderscore=str(replaceUnderscore)
                self.lineEditPop.setText("{:,.0F}".format(replaceUnderscore))
            except ValueError:
                QMessageBox.information(self, "Invalid", "Data is invalid so not updated in memory", QMessageBox.Ok)
                self.lineEditPop.setText(self.my2dList[selected_index][1])
                (compileList[selected_index][1]) = self.lineEditPop.text()
                replaceUnderscore = (compileList[selected index][1])
                replaceUnderscore=int(replaceUnderscore)
                self.lineEditPop.setText("{:,.0F}".format(replaceUnderscore))
                #Error checking + removing underscore from user inputs


    def populateListWithPeople(self):
        self.unsaved changes == True
        self.lineEditPop.clear()
        selected_index = self.listWidgetPeople.currentRow()
        strCheck = (compileList[selected index][1])
        replaceUnderscore = strCheck
        replaceUnderscore=replaceUnderscore.replace(",","_")
        replaceUnderscore=replaceUnderscore.replace("_","")
        intStrCheck = int(replaceUnderscore)
        self.lineEditPop.clear()
        self.lineEditPop.setText("{:,.0F}".format(intStrCheck))
        #repopulate list, check there are no underscores

# DO NOT MODIFY THIS CODE
if __name__ == "__main__":
    app = QApplication(sys.argv)
    the form = MyForm()
    the_form.show()
    sys.exit(app.exec_())
# END DO NOT MODIFY
```

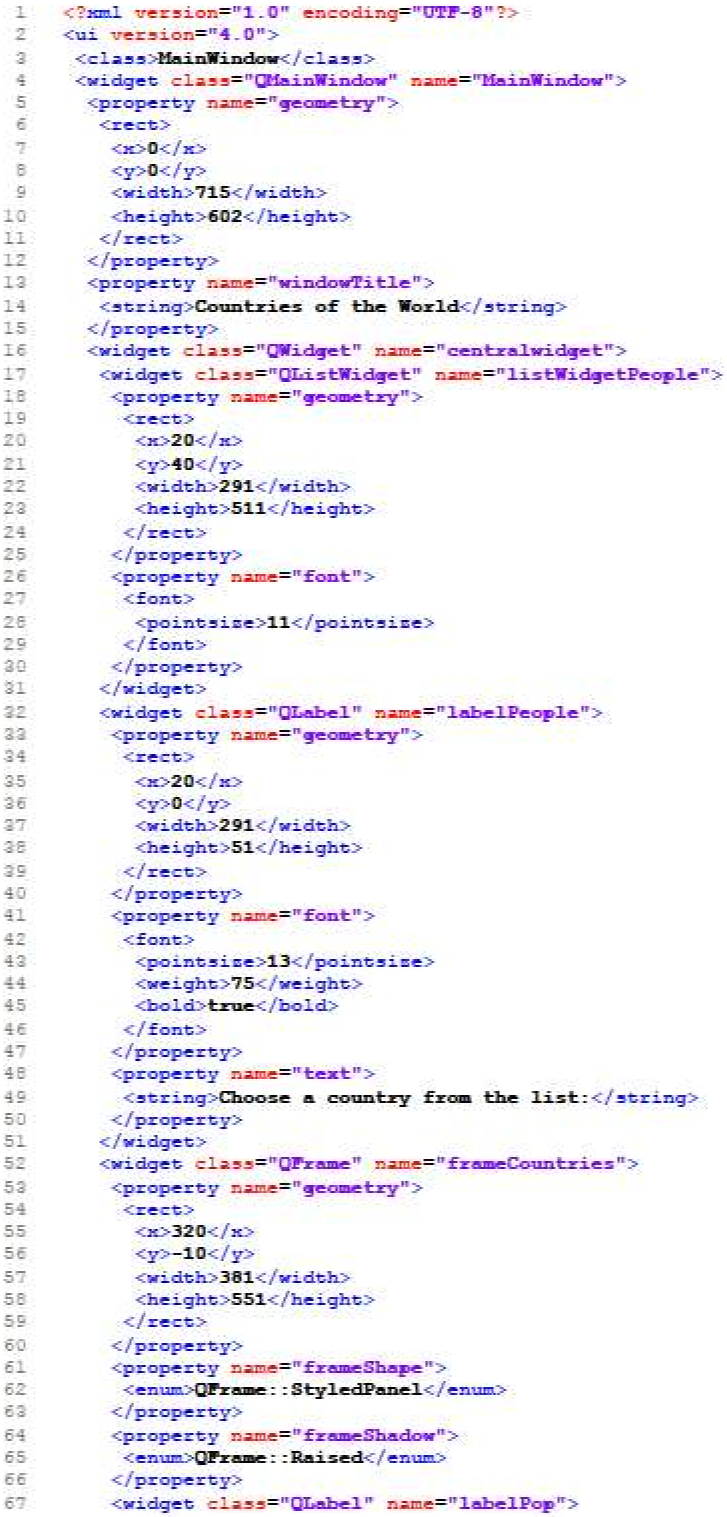
PY code, created by: PyQt5 UI code generator 5.15.1

```python
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'c:\PROG1700 SourceCode\w0218814 CrosbyS\GUI\FinalProject\FinalProject.ui'
#
# Created by: PyQt5 UI code generator 5.15.1
#
# WARNING: Any manual changes made to this file will be lost when pyuic5 is
# run again.  Do not edit this file unless you know what you are doing.


from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(715, 602)
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.listWidgetPeople = QtWidgets.QListWidget(self.centralwidget)
        self.listWidgetPeople.setGeometry(QtCore.QRect(20, 40, 291, 511))
        font = QtGui.QFont()
        font.setPointSize(11)
        self.listWidgetPeople.setFont(font)
        self.listWidgetPeople.setObjectName("listWidgetPeople")
        self.labelPeople = QtWidgets.QLabel(self.centralwidget)
        self.labelPeople.setGeometry(QtCore.QRect(20, 0, 291, 51))
        font = QtGui.QFont()
        font.setPointSize(13)
        font.setBold(True)
        font.setWeight(75)
        self.labelPeople.setFont(font)
        self.labelPeople.setObjectName("labelPeople")
        self.frameCountries = QtWidgets.QFrame(self.centralwidget)
        self.frameCountries.setGeometry(QtCore.QRect(320, -10, 381, 551))
        self.frameCountries.setFrameShape(QtWidgets.QFrame.StyledPanel)
        self.frameCountries.setFrameShadow(QtWidgets.QFrame.Raised)
        self.frameCountries.setObjectName("frameCountries")
        self.labelPop = QtWidgets.QLabel(self.frameCountries)
        self.labelPop.setGeometry(QtCore.QRect(20, 250, 121, 41))
        font = QtGui.QFont()
        font.setPointSize(12)
        self.labelPop.setFont(font)
        self.labelPop.setObjectName("labelPop")
        self.comboBoxMiFt = QtWidgets.QComboBox(self.frameCountries)
        self.comboBoxMiFt.setGeometry(QtCore.QRect(120, 370, 91, 22))
        font = QtGui.QFont()
        font.setPointSize(12)
        self.comboBoxMiFt.setFont(font)
        self.comboBoxMiFt.setObjectName("comboBoxMiFt")
        self.labelEditSqm = QtWidgets.QLabel(self.frameCountries)
        self.labelEditSqm.setGeometry(QtCore.QRect(270, 360, 101, 41))
        font = QtGui.QFont()
        font.setPointSize(12)
        font.setBold(True)
        font.setWeight(75)
        self.labelEditSqm.setFont(font)
        self.labelEditSqm.setText("")
        self.labelEditSqm.setObjectName("labelEditSqm")
        self.labelDensity = QtWidgets.QLabel(self.frameCountries)
        self.labelDensity.setGeometry(QtCore.QRect(40, 410, 141, 41))
        font = QtGui.QFont()
        font.setPointSize(12)
        self.labelDensity.setFont(font)
        self.labelDensity.setObjectName("labelDensity")
        self.radioButtonKM = QtWidgets.QRadioButton(self.frameCountries)
        self.radioButtonKM.setGeometry(QtCore.QRect(40, 470, 141, 16))
        font = QtGui.QFont()
        font.setPointSize(12)
        self.radioButtonKM.setFont(font)
        self.radioButtonKM.setObjectName("radioButtonKM")
        self.labelEditDensity = QtWidgets.QLabel(self.frameCountries)
        self.labelEditDensity.setGeometry(QtCore.QRect(270, 440, 91, 41))
        font = QtGui.QFont()
        font.setPointSize(12)
        font.setBold(True)
        font.setWeight(75)
        self.labelEditDensity.setFont(font)
        self.labelEditDensity.setText("")
        self.labelEditDensity.setObjectName("labelEditDensity")
        self.labelTotalArea = QtWidgets.QLabel(self.frameCountries)
        self.labelTotalArea.setGeometry(QtCore.QRect(20, 360, 121, 41))
        font = QtGui.QFont()
        font.setPointSize(12)
        self.labelTotalArea.setFont(font)
        self.labelTotalArea.setObjectName("labelTotalArea")
        self.radioButtonMile = QtWidgets.QRadioButton(self.frameCountries)
        self.radioButtonMile.setGeometry(QtCore.QRect(40, 450, 141, 16))
        font = QtGui.QFont()
        font.setPointSize(12)
        self.radioButtonMile.setFont(font)
        self.radioButtonMile.setObjectName("radioButtonMile")
        self.lineEditPop = QtWidgets.QLineEdit(self.frameCountries)
        self.lineEditPop.setGeometry(QtCore.QRect(110, 260, 251, 21))
        self.lineEditPop.setObjectName("lineEditPop")
        self.labelPicture = QtWidgets.QLabel(self.frameCountries)
        self.labelPicture.setGeometry(QtCore.QRect(30, 100, 241, 111))
        self.labelPicture.setText("")
        self.labelPicture.setObjectName("labelPicture")
        self.labelEditPopulation = QtWidgets.QLabel(self.frameCountries)
        self.labelEditPopulation.setGeometry(QtCore.QRect(270, 500, 91, 41))
        font = QtGui.QFont()
        font.setPointSize(12)
        font.setBold(True)
        font.setWeight(75)
        self.labelEditPopulation.setFont(font)
        self.labelEditPopulation.setText("")
        self.labelEditPopulation.setObjectName("labelEditPopulation")
        self.labelPercent = QtWidgets.QLabel(self.frameCountries)
        self.labelPercent.setGeometry(QtCore.QRect(20, 500, 261, 41))
        font = QtGui.QFont()
        font.setPointSize(12)
        self.labelPercent.setFont(font)
        self.labelPercent.setObjectName("labelPercent")
        self.labelCountry = QtWidgets.QLabel(self.frameCountries)
        self.labelCountry.setGeometry(QtCore.QRect(0, 40, 441, 41))
        font = QtGui.QFont()
        font.setPointSize(12)
        font.setBold(True)
        font.setWeight(75)
        self.labelCountry.setFont(font)
        self.labelCountry.setText("")
        self.labelCountry.setObjectName("labelCountry")
        self.pushButtonUpdate = QtWidgets.QPushButton(self.frameCountries)
        self.pushButtonUpdate.setGeometry(QtCore.QRect(220, 290, 141, 31))
        font = QtGui.QFont()
        font.setPointSize(12)
        self.pushButtonUpdate.setFont(font)
        self.pushButtonUpdate.setObjectName("pushButtonUpdate")
        MainWindow.setCentralWidget(self.centralwidget)
        self.menubar = QtWidgets.QMenuBar(MainWindow)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 715, 21))
        self.menubar.setObjectName("menubar")
```

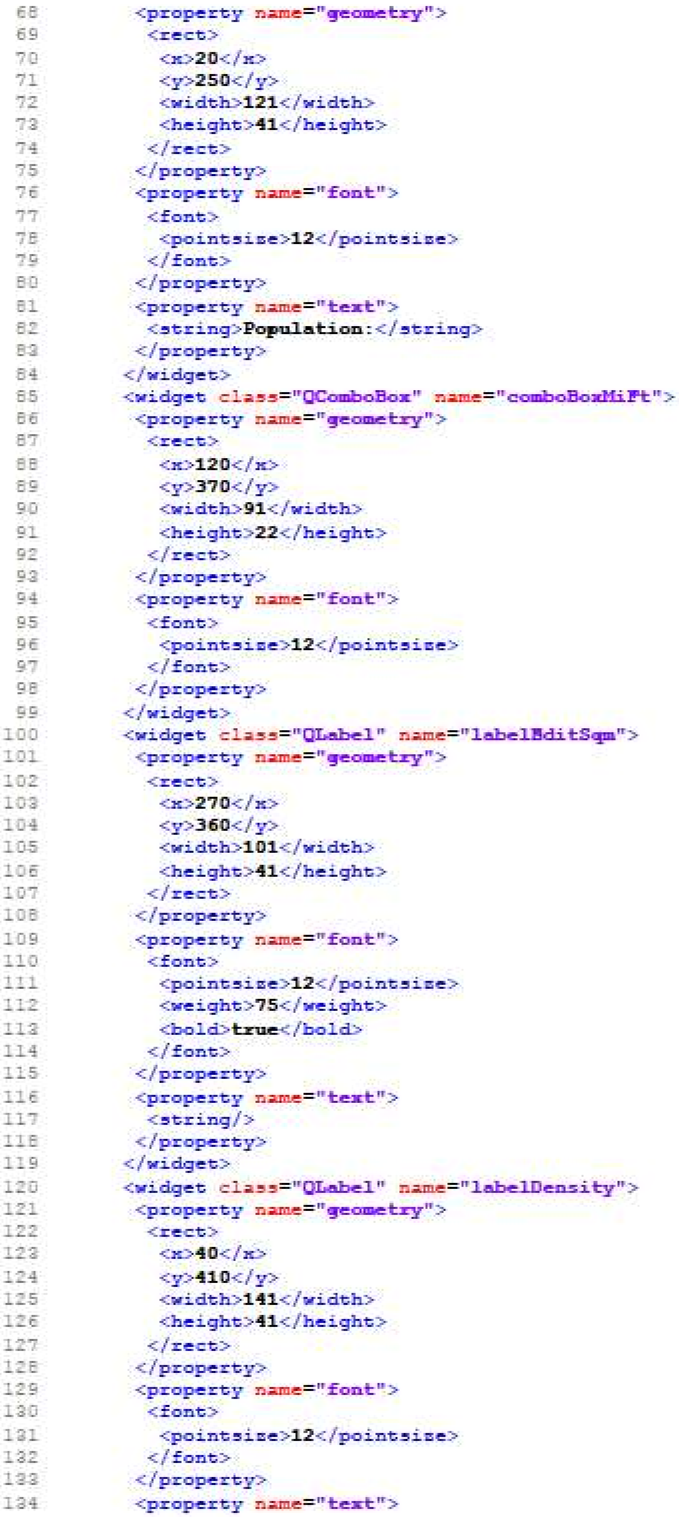


```
        self.menuFile = QtWidgets.QMenu(self.menubar)
        self.menuFile.setObjectName("menuFile")
        MainWindow.setMenuBar(self.menubar)
        self.statusbar = QtWidgets.QStatusBar(MainWindow)
        self.statusbar.setObjectName("statusbar")
        MainWindow.setStatusBar(self.statusbar)
        self.actionLoadCountries = QtWidgets.QAction(MainWindow)
        self.actionLoadCountries.setObjectName("actionLoadCountries")
        self.actionSavetoFile = QtWidgets.QAction(MainWindow)
        self.actionSavetoFile.setEnabled(False)
        self.actionSavetoFile.setObjectName("actionSavetoFile")
        self.actionExit = QtWidgets.QAction(MainWindow)
        self.actionExit.setObjectName("actionExit")
        self.menuFile.addAction(self.actionLoadCountries)
        self.menuFile.addAction(self.actionSavetoFile)
        self.menuFile.addSeparator()
        self.menuFile.addAction(self.actionExit)
        self.menubar.addAction(self.menuFile.menuAction())

        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow", "Countries of the World"))
        self.labelPeople.setText(_translate("MainWindow", "Choose a country from the
        list:"))
        self.labelPop.setText(_translate("MainWindow", "Population:"))
        self.labelDensity.setText(_translate("MainWindow", "Population Density"))
        self.radioButtonKM.setText(_translate("MainWindow", "Per Square KM"))
        self.labelTotalArea.setText(_translate("MainWindow", "Total Area in"))
        self.radioButtonMile.setText(_translate("MainWindow", "Per Square Mile"))
        self.labelPercent.setText(_translate("MainWindow", "Percentage of World
        Population:"))
        self.pushButtonUpdate.setText(_translate("MainWindow", "Update Population"))
        self.menuFile.setTitle(_translate("MainWindow", "File"))
        self.actionLoadCountries.setText(_translate("MainWindow", "Load Countries"))
        self.actionSavetoFile.setText(_translate("MainWindow", "Save to File"))
        self.actionExit.setText(_translate("MainWindow", "Exit"))
```

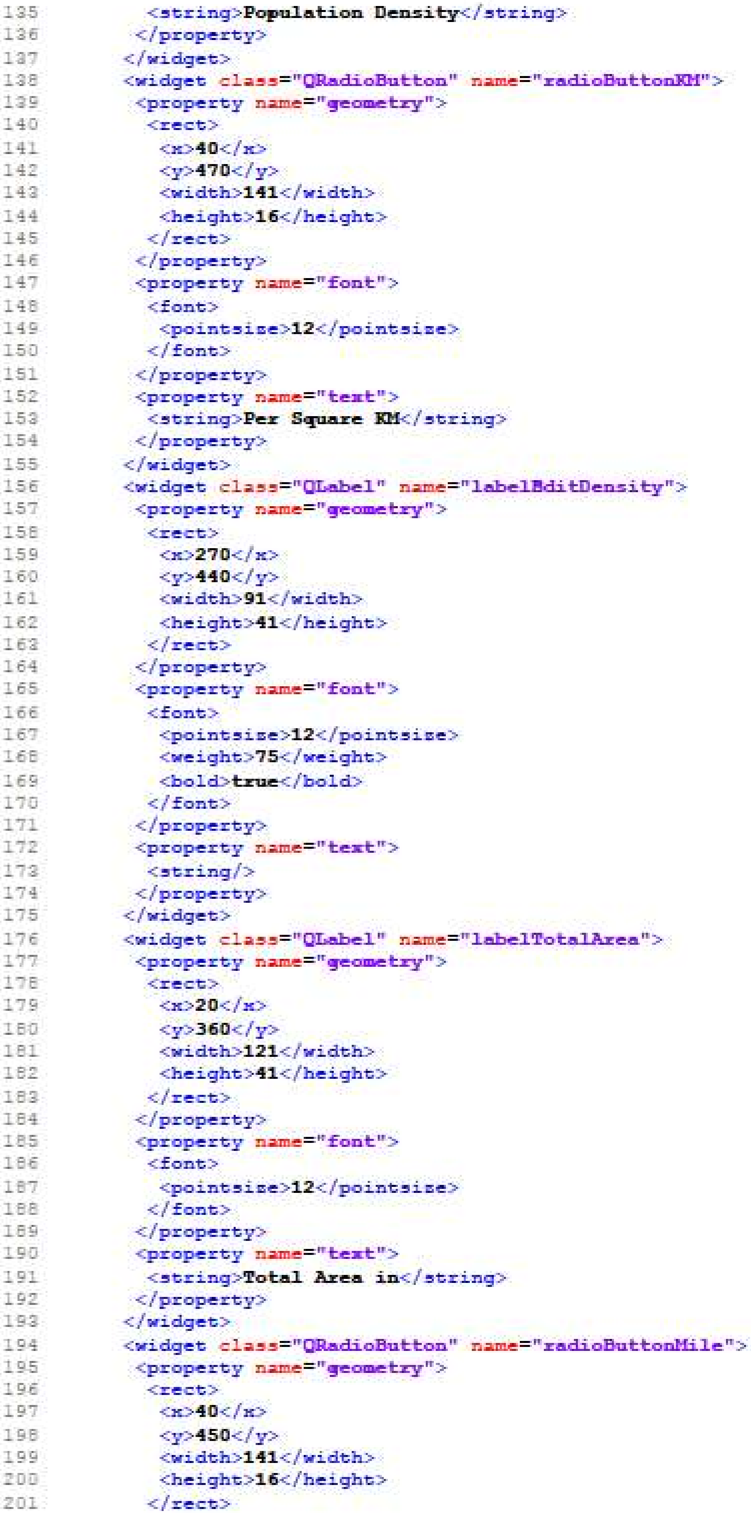
The is the text version of the code generated by QTDesigner from within PYQT5



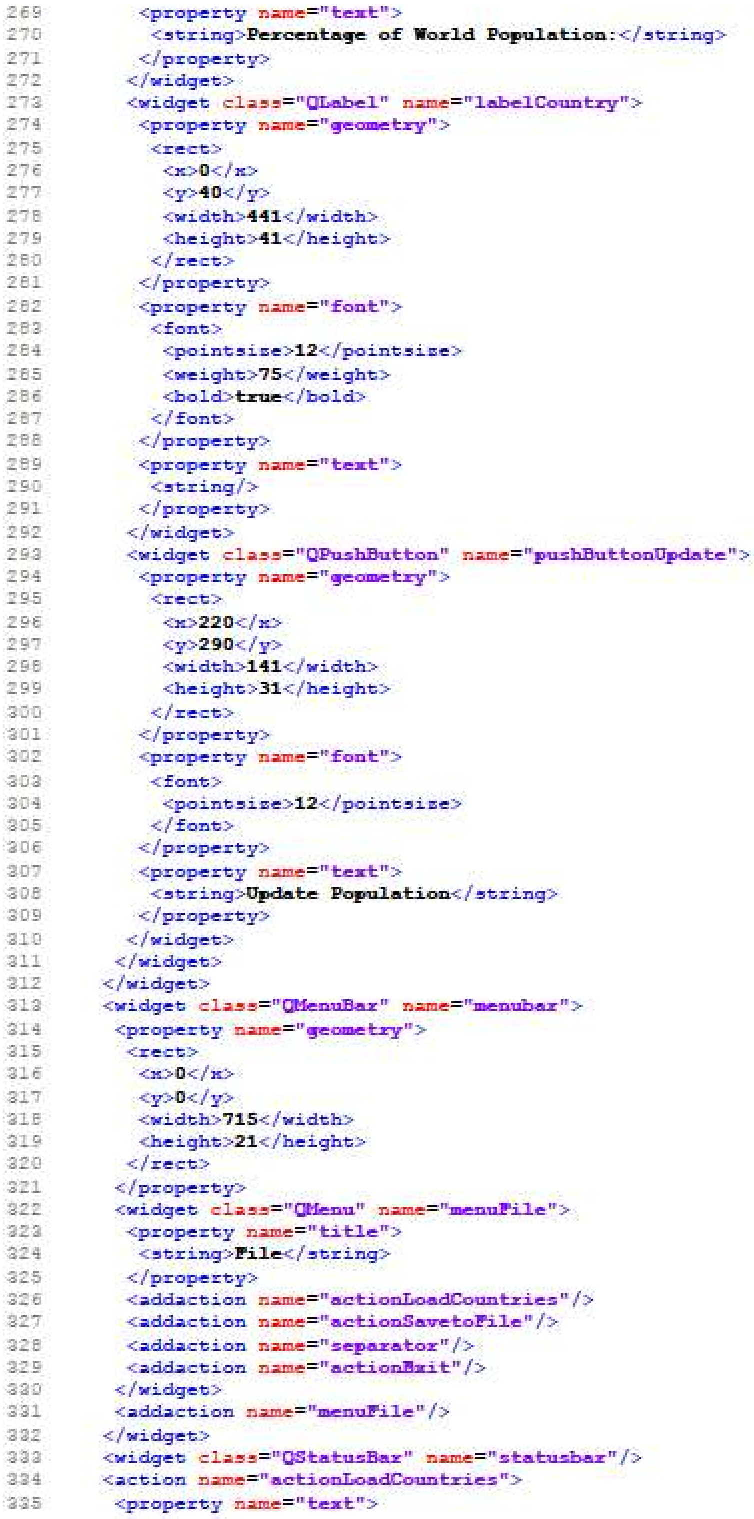
```xml
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>715</width>
    <height>602</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>Countries of the World</string>
  </property>
  <widget class="QWidget" name="centralwidget">
   <widget class="QListWidget" name="listWidgetPeople">
    <property name="geometry">
     <rect>
      <x>20</x>
      <y>40</y>
      <width>291</width>
      <height>511</height>
     </rect>
    </property>
    <property name="font">
     <font>
      <pointsize>11</pointsize>
     </font>
    </property>
   </widget>
   <widget class="QLabel" name="labelPeople">
    <property name="geometry">
     <rect>
      <x>20</x>
      <y>0</y>
      <width>291</width>
      <height>51</height>
     </rect>
    </property>
    <property name="font">
     <font>
      <pointsize>13</pointsize>
      <weight>75</weight>
      <bold>true</bold>
     </font>
    </property>
    <property name="text">
     <string>Choose a country from the list:</string>
    </property>
   </widget>
   <widget class="QFrame" name="frameCountries">
    <property name="geometry">
     <rect>
      <x>320</x>
      <y>-10</y>
      <width>381</width>
      <height>551</height>
     </rect>
    </property>
    <property name="frameShape">
     <enum>QFrame::StyledPanel</enum>
    </property>
    <property name="frameShadow">
     <enum>QFrame::Raised</enum>
    </property>
    <widget class="QLabel" name="labelPop">
```



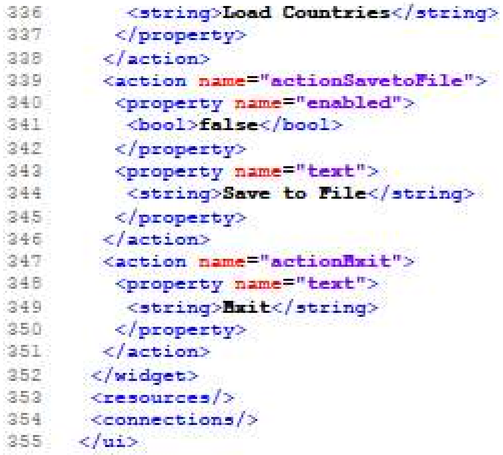
```
    <property name="geometry">
     <rect>
      <x>20</x>
      <y>250</y>
      <width>121</width>
      <height>41</height>
     </rect>
    </property>
    <property name="font">
     <font>
      <pointsize>12</pointsize>
     </font>
    </property>
    <property name="text">
     <string>Population:</string>
    </property>
   </widget>
   <widget class="QComboBox" name="comboBoxMiFt">
    <property name="geometry">
     <rect>
      <x>120</x>
      <y>370</y>
      <width>91</width>
      <height>22</height>
     </rect>
    </property>
    <property name="font">
     <font>
      <pointsize>12</pointsize>
     </font>
    </property>
   </widget>
   <widget class="QLabel" name="labelEditSqm">
    <property name="geometry">
     <rect>
      <x>270</x>
      <y>360</y>
      <width>101</width>
      <height>41</height>
     </rect>
    </property>
    <property name="font">
     <font>
      <pointsize>12</pointsize>
      <weight>75</weight>
      <bold>true</bold>
     </font>
    </property>
    <property name="text">
     <string/>
    </property>
   </widget>
   <widget class="QLabel" name="labelDensity">
    <property name="geometry">
     <rect>
      <x>40</x>
      <y>410</y>
      <width>141</width>
      <height>41</height>
     </rect>
    </property>
    <property name="font">
     <font>
      <pointsize>12</pointsize>
     </font>
    </property>
    <property name="text">
```



```
     <string>Population Density</string>
    </property>
   </widget>
   <widget class="QRadioButton" name="radioButtonKM">
    <property name="geometry">
     <rect>
      <x>40</x>
      <y>470</y>
      <width>141</width>
      <height>16</height>
     </rect>
    </property>
    <property name="font">
     <font>
      <pointsize>12</pointsize>
     </font>
    </property>
    <property name="text">
     <string>Per Square KM</string>
    </property>
   </widget>
   <widget class="QLabel" name="labelEditDensity">
    <property name="geometry">
     <rect>
      <x>270</x>
      <y>440</y>
      <width>91</width>
      <height>41</height>
     </rect>
    </property>
    <property name="font">
     <font>
      <pointsize>12</pointsize>
      <weight>75</weight>
      <bold>true</bold>
     </font>
    </property>
    <property name="text">
     <string/>
    </property>
   </widget>
   <widget class="QLabel" name="labelTotalArea">
    <property name="geometry">
     <rect>
      <x>20</x>
      <y>360</y>
      <width>121</width>
      <height>41</height>
     </rect>
    </property>
    <property name="font">
     <font>
      <pointsize>12</pointsize>
     </font>
    </property>
    <property name="text">
     <string>Total Area in</string>
    </property>
   </widget>
   <widget class="QRadioButton" name="radioButtonMile">
    <property name="geometry">
     <rect>
      <x>40</x>
      <y>450</y>
      <width>141</width>
      <height>16</height>
     </rect>
     </property>
     <property name="font">
      <font>
       <pointsize>12</pointsize>
      </font>
     </property>
     <property name="text">
      <string>Per Square Mile</string>
     </property>
    </widget>
    <widget class="QLineEdit" name="lineEditPop">
     <property name="geometry">
      <rect>
       <x>110</x>
       <y>260</y>
       <width>251</width>
       <height>21</height>
      </rect>
     </property>
    </widget>
    <widget class="QLabel" name="labelPicture">
     <property name="geometry">
      <rect>
       <x>30</x>
       <y>100</y>
       <width>241</width>
       <height>111</height>
      </rect>
     </property>
     <property name="text">
      <string/>
     </property>
    </widget>
    <widget class="QLabel" name="labelEditPopulation">
     <property name="geometry">
      <rect>
       <x>270</x>
       <y>500</y>
       <width>91</width>
       <height>41</height>
      </rect>
     </property>
     <property name="font">
      <font>
       <pointsize>12</pointsize>
       <weight>75</weight>
       <bold>true</bold>
      </font>
     </property>
     <property name="text">
      <string/>
     </property>
    </widget>
    <widget class="QLabel" name="labelPercent">
     <property name="geometry">
      <rect>
       <x>20</x>
       <y>500</y>
       <width>261</width>
       <height>41</height>
      </rect>
     </property>
     <property name="font">
      <font>
       <pointsize>12</pointsize>
      </font>
     </property>
```



```
     <property name="text">
      <string>Percentage of World Population:</string>
     </property>
    </widget>
    <widget class="QLabel" name="labelCountry">
     <property name="geometry">
      <rect>
       <x>0</x>
       <y>40</y>
       <width>441</width>
       <height>41</height>
      </rect>
     </property>
     <property name="font">
      <font>
       <pointsize>12</pointsize>
       <weight>75</weight>
       <bold>true</bold>
      </font>
     </property>
     <property name="text">
      <string/>
     </property>
    </widget>
    <widget class="QPushButton" name="pushButtonUpdate">
     <property name="geometry">
      <rect>
       <x>220</x>
       <y>290</y>
       <width>141</width>
       <height>31</height>
      </rect>
     </property>
     <property name="font">
      <font>
       <pointsize>12</pointsize>
      </font>
     </property>
     <property name="text">
      <string>Update Population</string>
     </property>
    </widget>
   </widget>
  </widget>
  <widget class="QMenuBar" name="menubar">
   <property name="geometry">
    <rect>
     <x>0</x>
     <y>0</y>
     <width>715</width>
     <height>21</height>
    </rect>
   </property>
   <widget class="QMenu" name="menuFile">
    <property name="title">
     <string>File</string>
    </property>
    <addaction name="actionLoadCountries"/>
    <addaction name="actionSavetoFile"/>
    <addaction name="separator"/>
    <addaction name="actionExit"/>
   </widget>
   <addaction name="menuFile"/>
  </widget>
  <widget class="QStatusBar" name="statusbar"/>
  <action name="actionLoadCountries">
   <property name="text">
```



```
    <string>Load Countries</string>
   </property>
  </action>
  <action name="actionSavetoFile">
   <property name="enabled">
    <bool>false</bool>
   </property>
   <property name="text">
    <string>Save to File</string>
   </property>
  </action>
  <action name="actionExit">
   <property name="text">
    <string>Exit</string>
   </property>
  </action>
 </widget>
 <resources/>
 <connections/>
</ui>
```