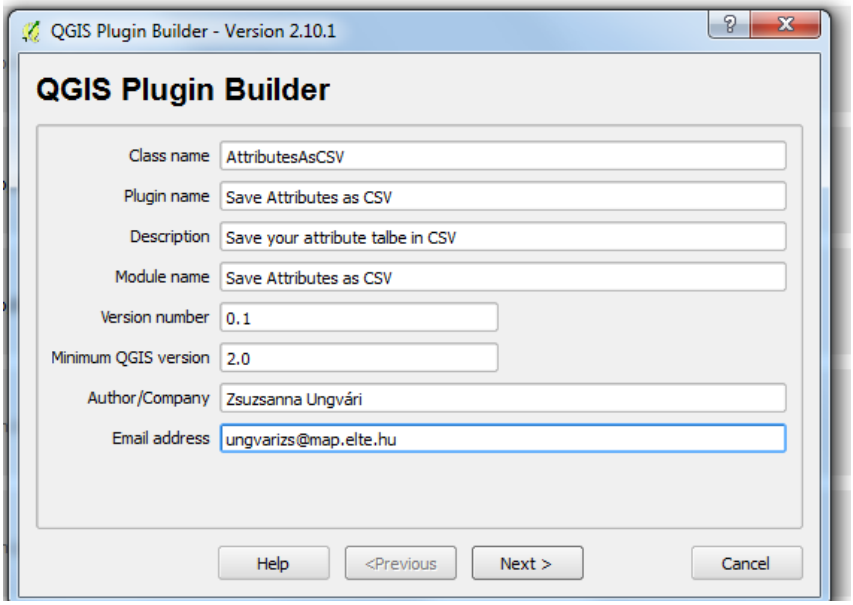
23.Modul készítése QGIS alá Python-nal

Készítsünk saját modult QGIS alá. Használjuk ehhez a Plugin Builder modult. Az új modul segítségével az egyes rétegek attribútum tábláit CSV fájlként tudjuk menteni.
Ez a fejezet Ujaval Gandhi leckéje alapján készült.
http://www.qgistutorials.com/en/docs/building_a_python_plugin.html

Hogyan programozzunk QGIS plugin-t. https://plugins.qgis.org/

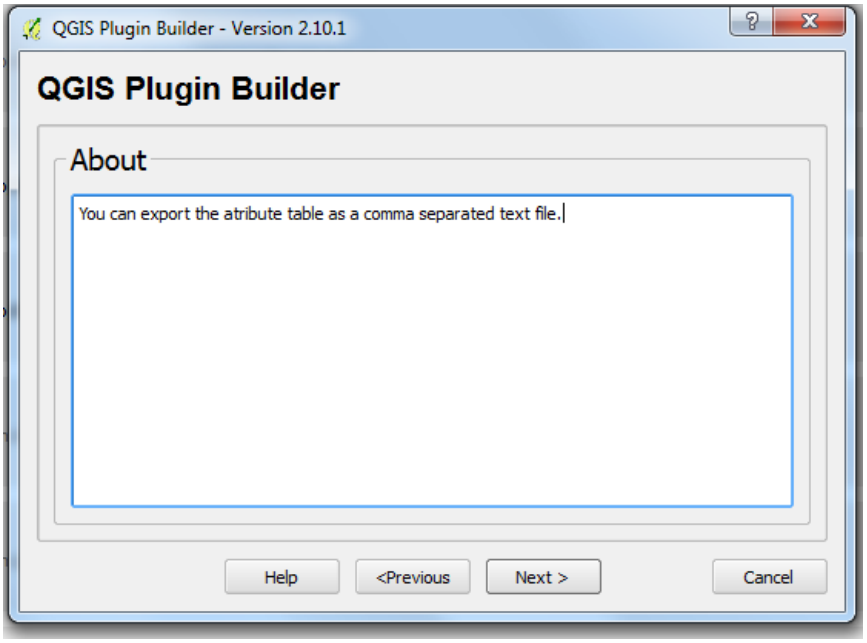
Megoldás

Telepítsük a Plugin Builder modult. Nyissuk meg a Plugins→Manage and Install Plugin menüt. Keressük meg az All-nál a Plugin Builder-t. Telepítsük, majd indítsuk el.
Töltsük ki a párbeszédpanelt.

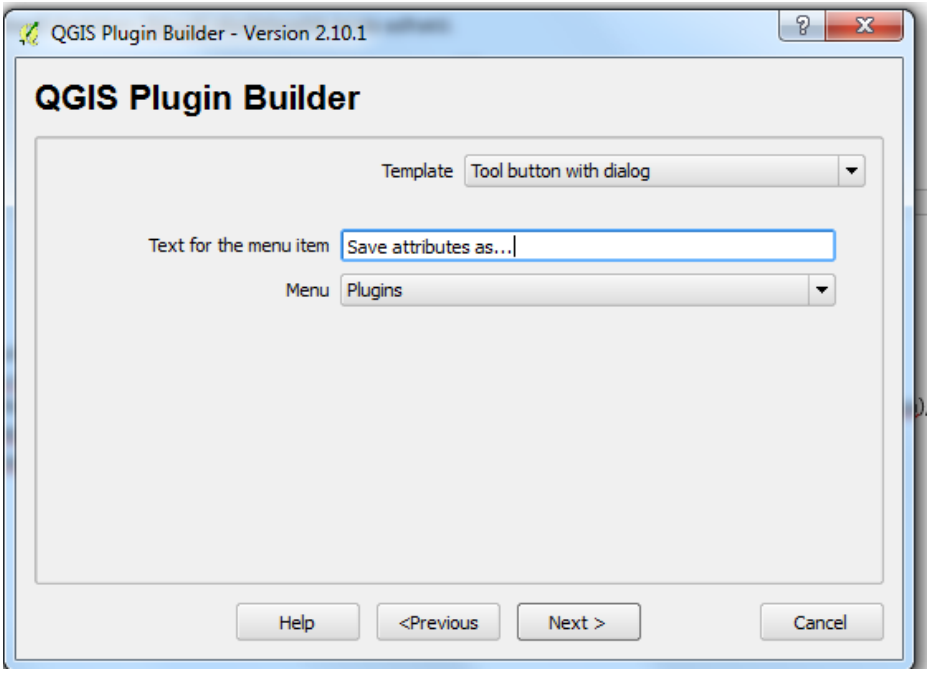


Megadhatjuk a plugin nevét (ez szerepel majd a fejlécben), az osztály nevét (Python osztály neve), a modul nevét, ebben ne legyen lehetőleg se ékezet, se szóköz. Ezen kívüé egy rövid leírást, a plugin verziószámát, a minimális QGIS verziót, a szerző nevét és e-mail címért.

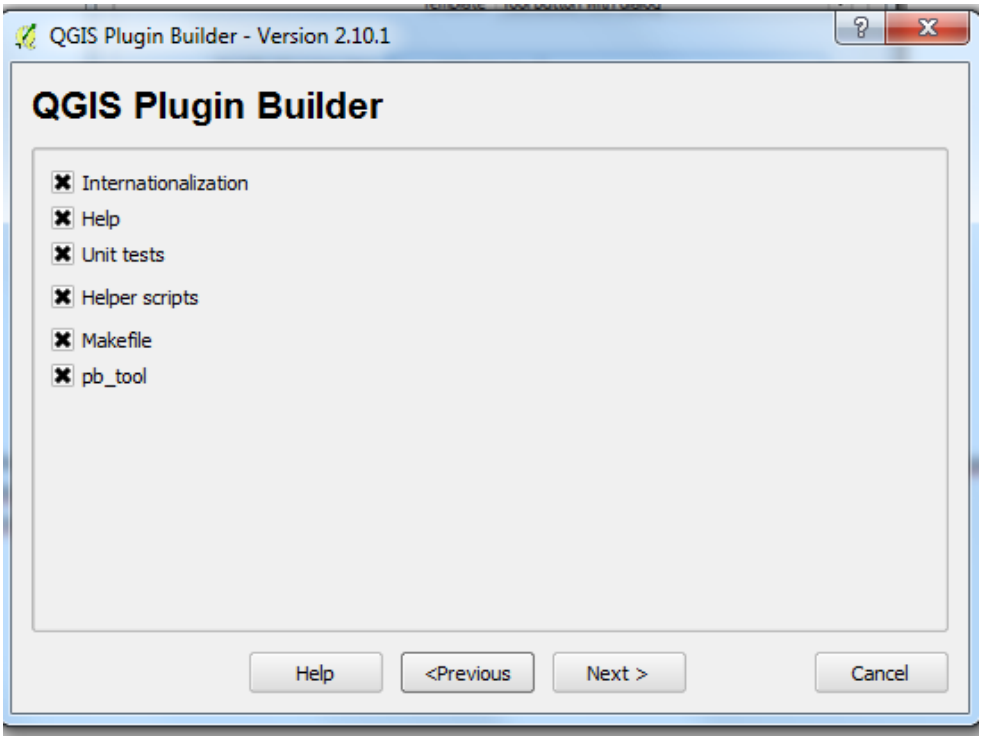
A következő lépésben (About) részletesebb leírás adható.



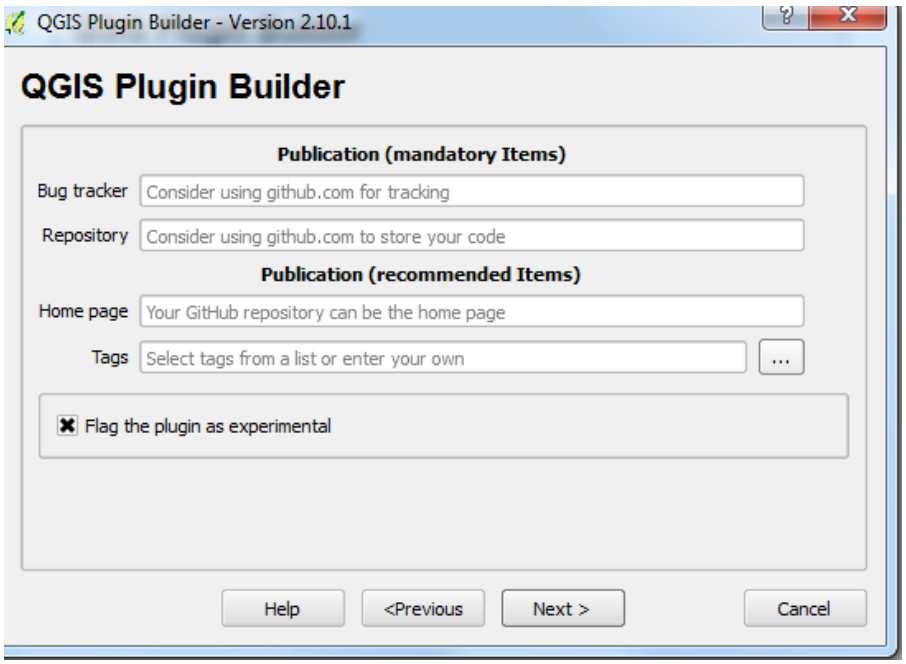
Válasszuk Template-nek a Tool button with dialog módot. Ezzel egy egyszerű párbeszédpanel jön létre.
A Menu-nél kiválasztható, melyik Főmenübe kerüljön bele, legyen ez most a Plugin.



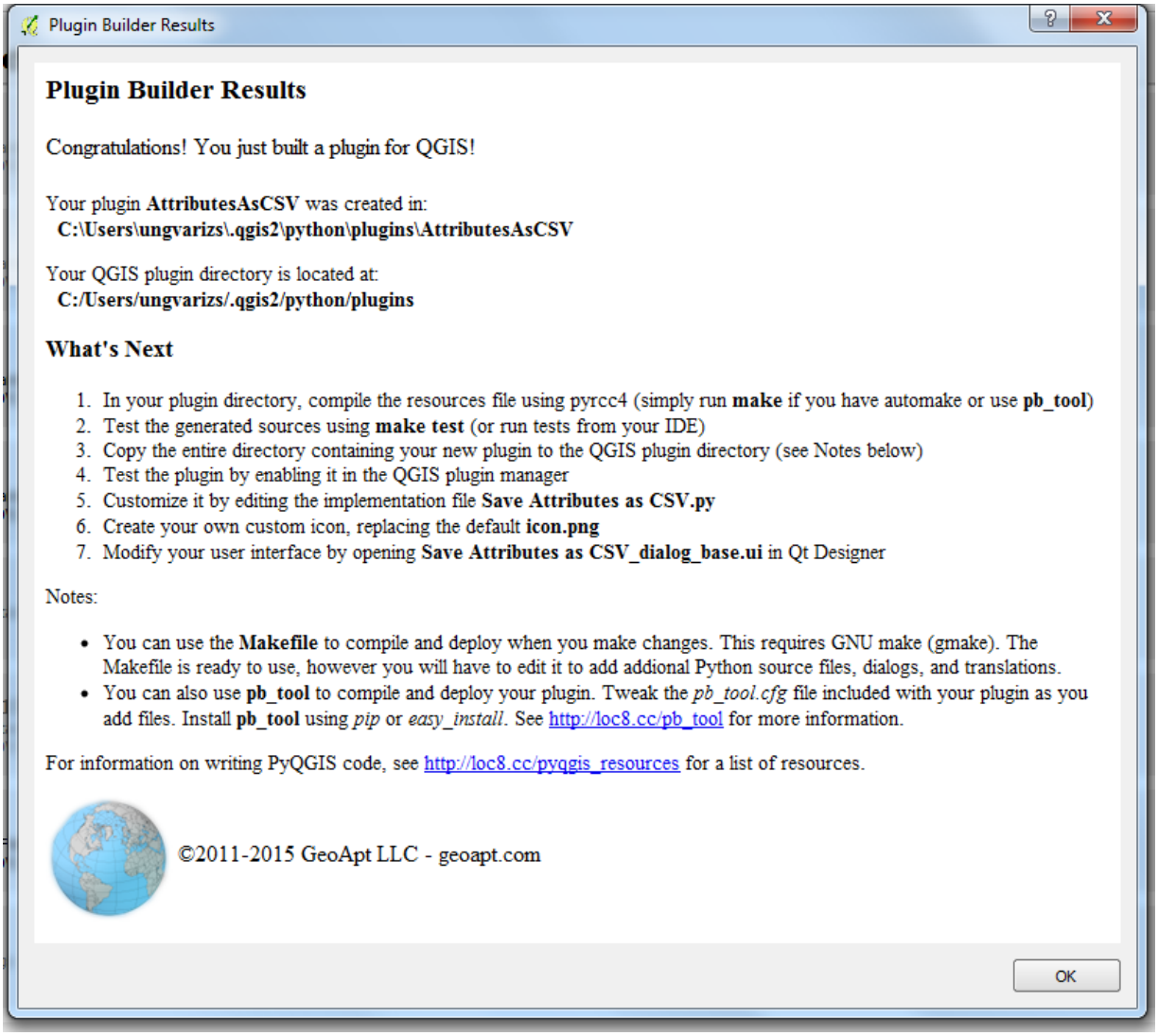
A következő lépésben megadható, milyen fájlokat készítsen el. (Internationalization: többnyelvűsítés, Help : Súgó létrehozása), Unit tests (tesztelés), Helper scripts, Makefile: a modult futtatásra előkészítő szkript, pb_tool).



A Bug tracker-be, és a Repository-ba lehet majd feltölteni a kész kódot. Ha nem írunk be semmit, általában nem enged továbblépni. Később ez is módosítható a létrejött metadata.txt fájlban.
A Flag the plugin as experimental azt jelenti, hogy a modul a kísérleti modulok közé kerül majd be. Ebben az esetben ne felejtsük el bekapcsolni a Manage and Install Plugin menüben a Settings-nél a Show also experimental plugin lehetőséget.

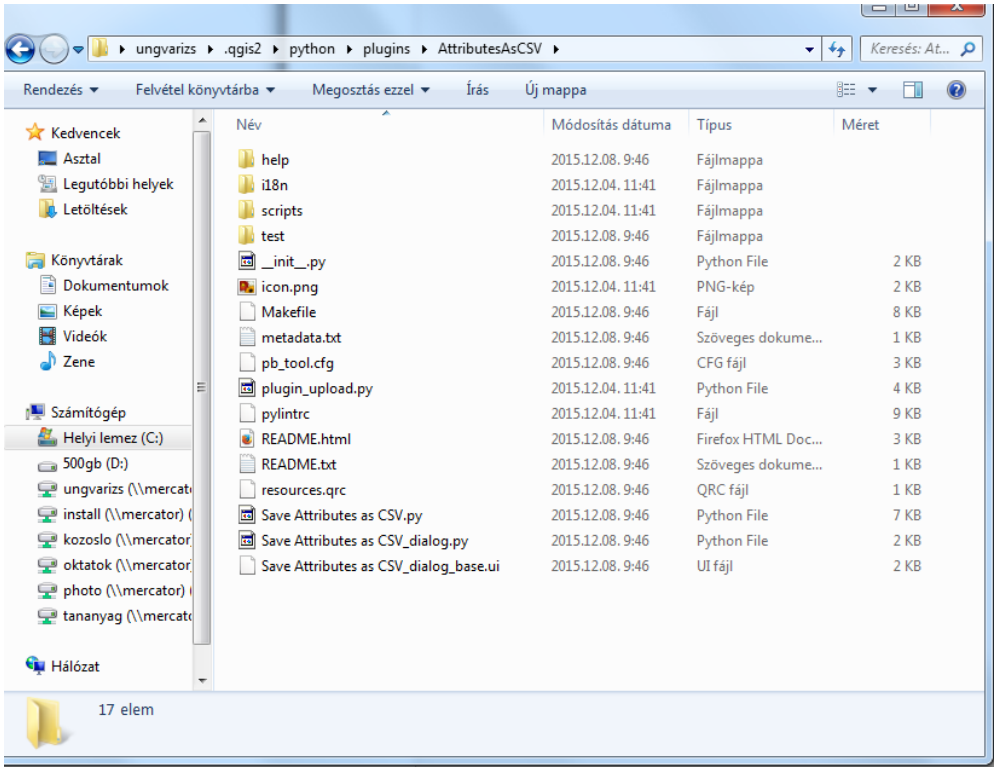


Következő lépésben ki kell választani a plugin helyét. A QGIS a plugineket a C:/Users/[sajatfelhasznalonev]/.qgis2/python/plugin mappába teszi, illetve innen olvassa be.

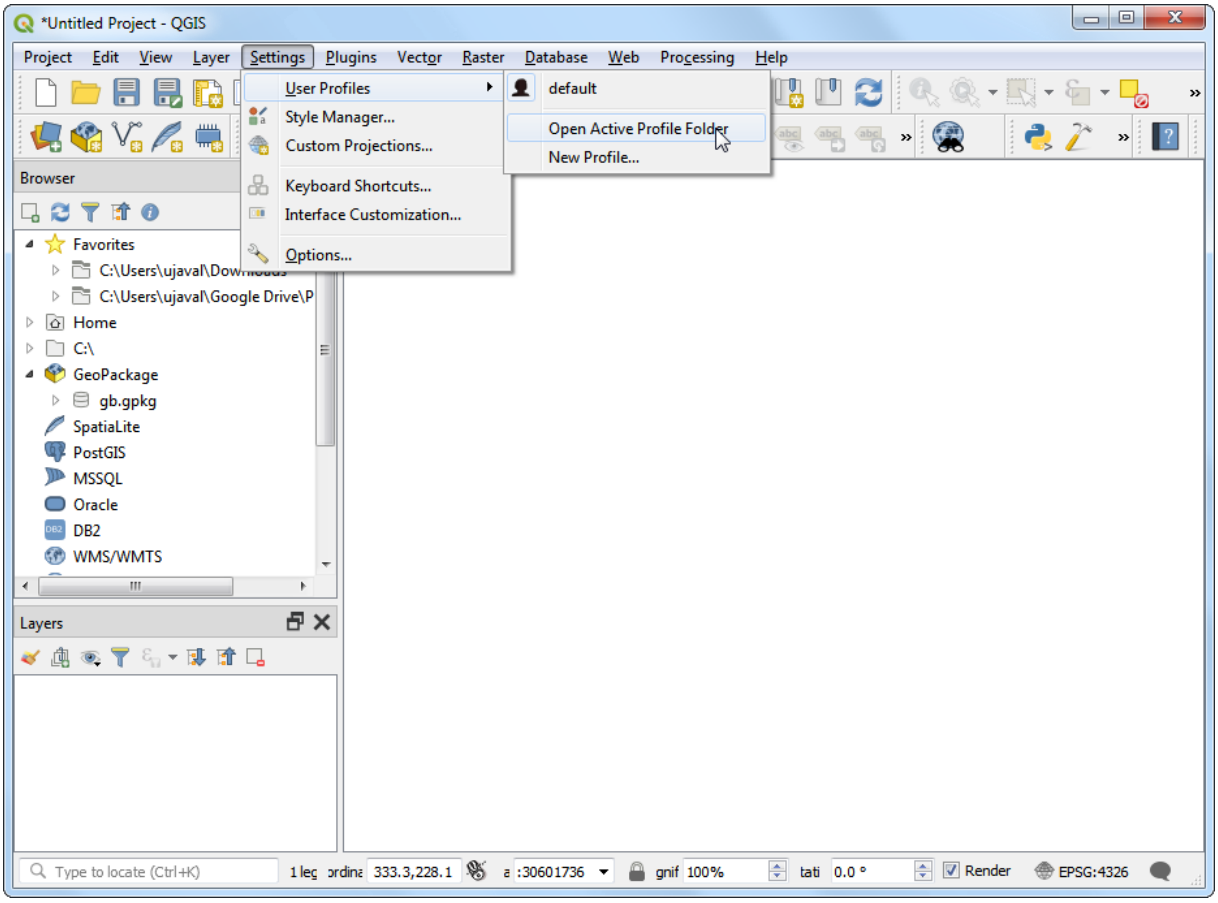


A fenti kép a sikeres létrehozást illusztrálja.
Most le kell fordítanunk a `RESOURCES.QRC` fájlt `RESOURCES.PY`-já, hogy a QGIS a létrejött modult be tudja olvasni.
Windows alatt létre kell hoznunk egy batch fájlt (kötegelt állomány, formailag szöveges fájl, amely parancsok sorozatát tartalmazza).
Bármi lehet a fájl neve, legyen pl. Compile.bat. Tartalma. Legyen a modullal azonos mappában, akkor egy az egyben lefuttatható. Futtasuk is le.

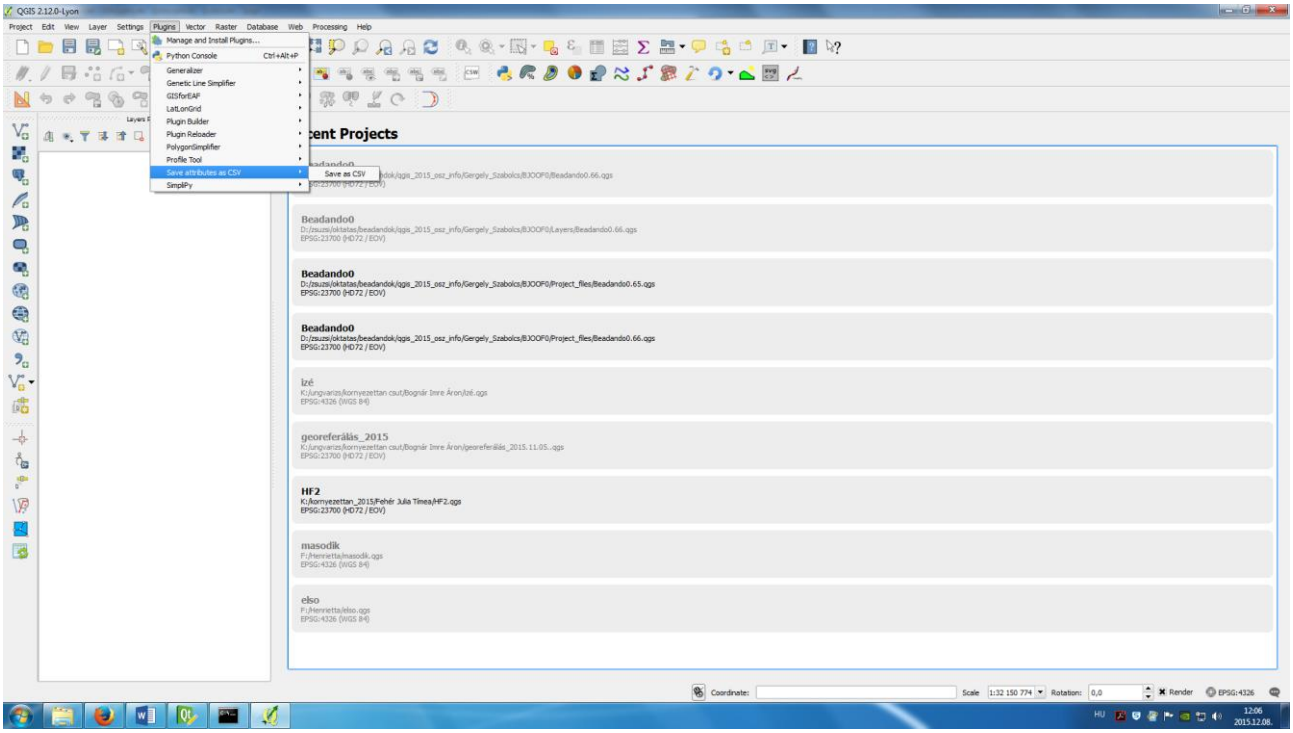
```
@echo off
call "C:\OSGeo4W64\bin\o4w_env.bat"
call "C:\OSGeo4W64\bin\qt5_env.bat"
call "C:\OSGeo4W64\bin\py3_env.bat"

@echo on
pyrcc5 -o resources.py resources.qrc
```



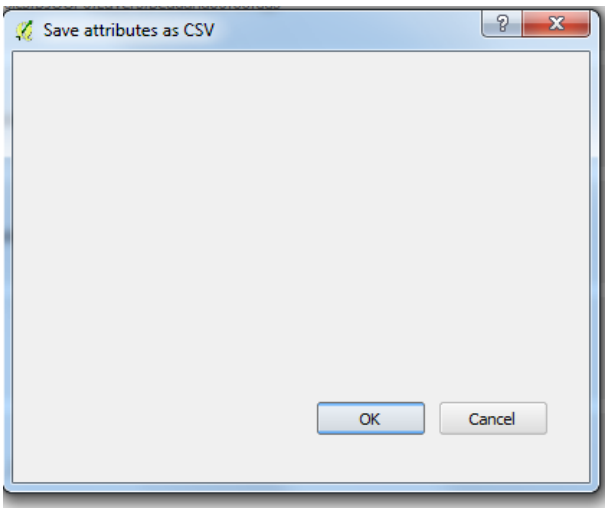
A valahová elmentett modult át kellene másolnunk abba a mappába, ahol a QGIS a modulokat „gyűjti”.
A mappa helye:
C:\Users\SAJÁTUSERNAME\AppData\Roaming\QGIS\QGIS3\profiles\default\python\plugin
Vagy ha nagyon nem sikerül, ezt a mappát a QGIS-ből is meghívhatjuk, a képen látható módon (Settings menü> user profile> open active profile folder)



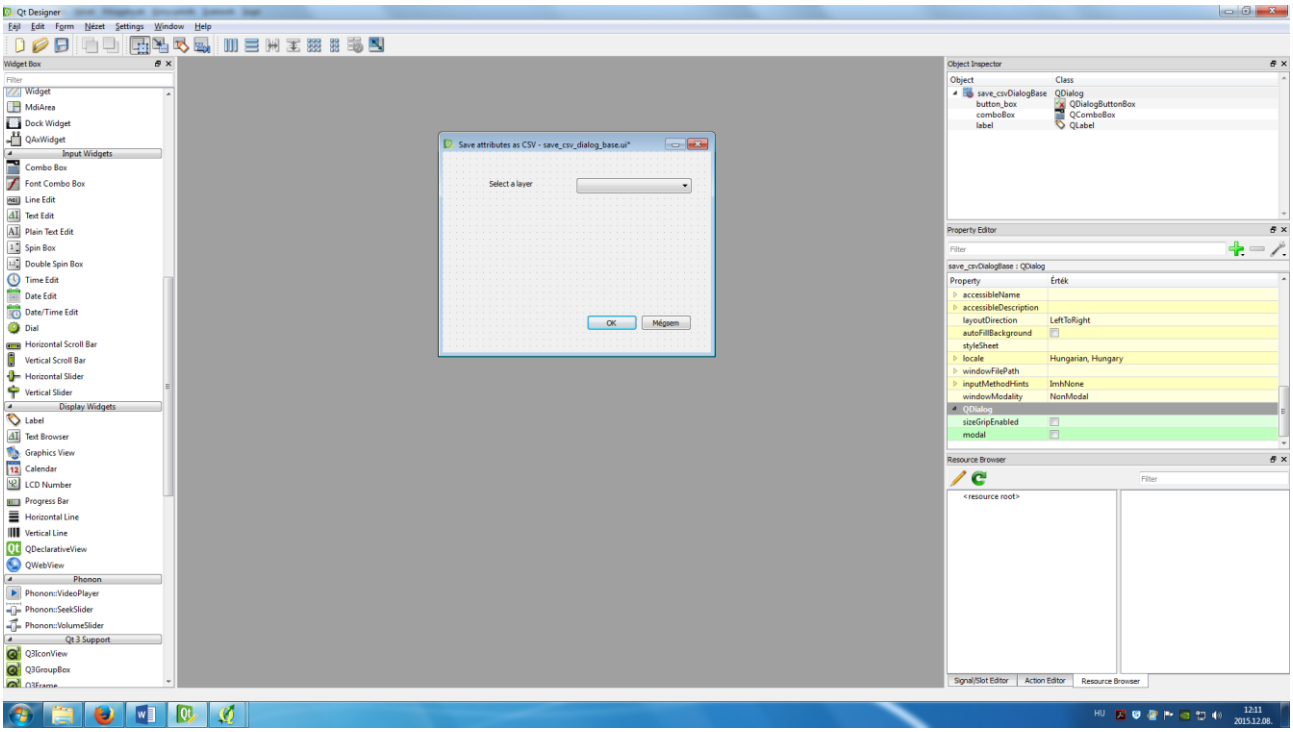
Na és az utolsó lépés, mielőtt az új pluginünk váza láthatóvá válik QGIS-ben.
Nyissuk meg a Plugin> Manage and Install Plugint. Keressünk rá a SaveAttributes-ra (vagyis a pluginünk nevére). Pipáljuk ki. Ezután keressük meg a menüben, amibe tettük. Én a pluginba tettem.



Így néz ki, egyelőre még nem csinál semmit. Most kell hozzá megadni a funkciókat.

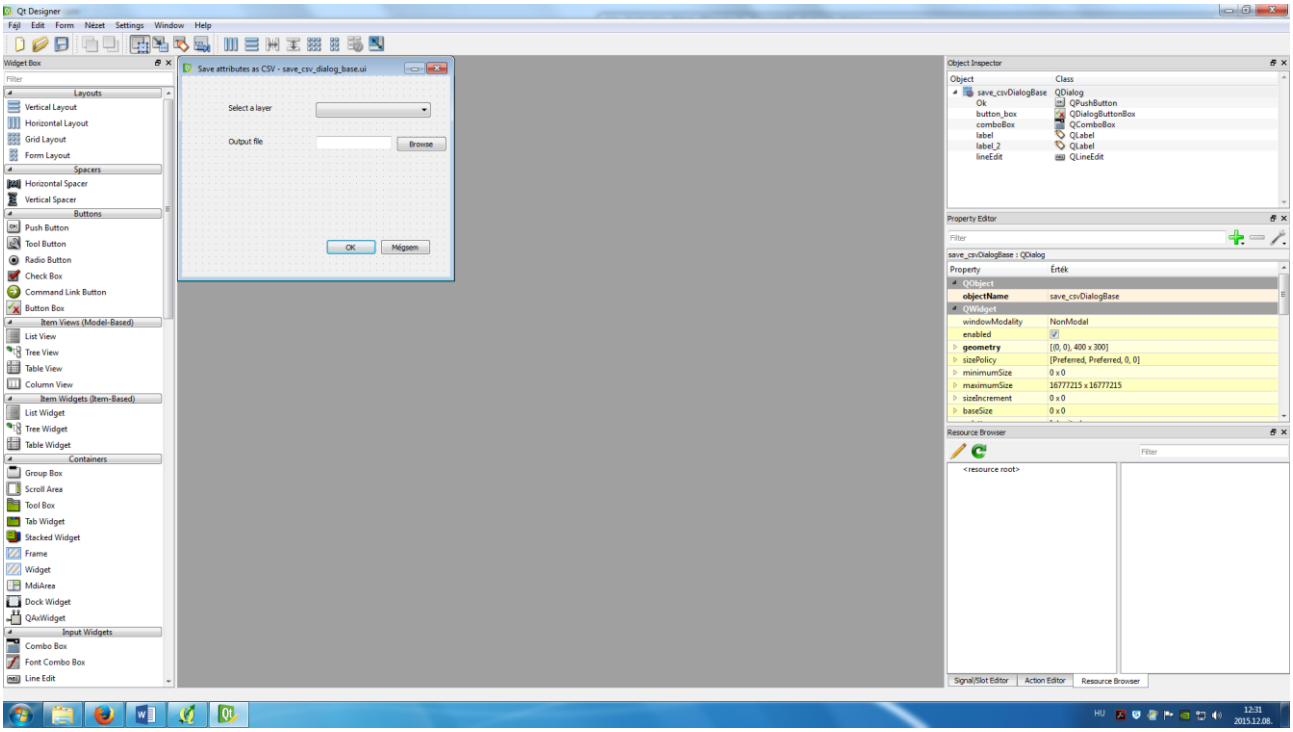


Először is nyissuk meg Qt Designer programot a Start menüből. Ez az OSGEO4W installerrel települ.
Ebben a programban tudjuk a plugin „kinézetét" egyszerűen megszerkeszteni, a program maga egy user interface designer, vagyis felhasználói felület tervező.
Nyissuk meg a modul mappájában lévő `save_csv_dialog_base.ui` UI kiterjesztésű fájlt.



Tegyünk fel egy comboBox-ot, amellyel a felhasználó a behívott rétegek közül tud választani. Ehhez tegyünk fel egy Text labelt is (szövege: Select layer), amelyre magyarázatokat fűzhetünk. Mindkettő az Input Widget-ben található, és csak rá kell húzni a modul felületére a bal egérgombbal.
Majd adjunk hozzá egy újabb Text labelt (Output file name felirattal), egy LineEditet, és

hozzá „Browse” feliratú pushButton gombot. A Line editbe írható be majd az elérési út. A gomb segítségével pedig kiválaszthatjuk a mentés helyét, és a fájl nevét.



A modul felhasználói felülete ezzel készen van, most meg kell írnunk azokat a kódrészleteket, amelyek a létrehozott objektumokat vezérlik.
Nyissuk meg a modul nevével azonos `.PY` kiterjesztésű fájlt a modul mappájából, vagyis az esetemben a `SAVE_ATTRIBUTES.PY` fájlt.
A Python kódok bármilyen szövegszerkesztővel megnyithatók (pl. Notepad++ stb.), de erre a célra ajánlom az IDLE-t, ez a Python beépített szerkesztője.

A következő kódrészleteket kell beszúrni:
A fájl elején az importálandó moduloknál tegyük be a QFileDialog-ot, ez az objektum a PyQt.QtWidgets modulban található.

```
from qgis.PyQt.QtWidgets import QAction, QFileDialog
"""
from qgis.PyQt.QtCore import QSettings, QTranslator, QCoreApplication
from qgis.PyQt.QtGui import QIcon
from qgis.PyQt.QtWidgets import QAction, QFileDialog
from qgis.core import QgsProject
```

A run(self) függvény elé szúrjuk be ezt a kódrészletet, ezzel elérhetjük, hogy beolvassa a rétegeket a comboBoxba:

```
    def select_output_file(self):
        filename, _filter = QFileDialog.getSaveFileName(
            self.dlg, "Select   output file ","", '*.csv')
        self.dlg.lineEdit.setText(filename)
```

```
    def unload(self):
        """Removes the plugin menu item and icon from QGIS GUI."""
        for action in self.actions:
            self.iface.removePluginMenu(
                self.tr(u'&Save Attributes'),
                action)
            self.iface.removeToolBarIcon(action)

    def select_output_file(self):
        filename, _filter = QFileDialog.getSaveFileName(
            self.dlg, "Select   output file ","", '*.csv')
        self.dlg.lineEdit.setText(filename)

    def run(self):
        """Run method that performs all the real work"""


self.dlg.pushButton.clicked.connect(self.select_output_file)


    def run(self):
        """Run method that performs all the real work"""

        # Create the dialog with elements (after translation) and keep reference
        # Only create GUI ONCE in callback, so that it will only load when the plugin is started
        if self.first_start == True:
            self.first_start = False
            self.dlg = saDialog()
            self.dlg.pushButton.clicked.connect(self.select_output_file)
if result:
                filename = self.dlg.lineEdit.text()
                with open(filename, 'w') as output_file:
                  selectedLayerIndex = self.dlg.comboBox.currentIndex()
                  selectedLayer = layers[selectedLayerIndex].layer()
                  fieldnames = [field.name() for field in
selectedLayer.fields()]
                  # write header
                  line = ','.join(name for name in fieldnames) + '\n'
                  output_file.write(line)
                  # wirte feature attributes
                  for f in selectedLayer.getFeatures():
                    line = ','.join(str(f[name]) for name in fieldnames) + '\n'
                    output_file.write(line)
    # See if OK was pressed
    if result:
        filename = self.dlg.lineEdit.text()
        with open(filename, 'w') as output_file:
          selectedLayerIndex = self.dlg.comboBox.currentIndex()
          selectedLayer = layers[selectedLayerIndex].layer()
          fieldnames = [field.name() for field in selectedLayer.fields()]
          # write header
          line = ','.join(name for name in fieldnames) + '\n'
          output_file.write(line)
          # wirte feature attributes
          for f in selectedLayer.getFeatures():
            line = ','.join(str(f[name]) for name in fieldnames) + '\n'
            output_file.write(line)
```

Készen vagyunk, teszteljük a modult!

Megjegyzés, kiegészítés

- A Python kis- és nagybetű érzékeny.
- Elgépeléskor, hibánál előfordulhat, hogy a modul nem töltődik be, nem látszik a

menüsoron.
- A Python-ban szigorúan ügyelni kell a behúzásokra (tabulátorral vagy négy szóközzel), ugyanis ez jelzi, hogy az adott blokk a ciklus v. elágazás része-e. Egyes szövegszerkesztőkben ez nem követhető jól, ezért ajánlom az IDLE-t, ez a Python beépített szövegszerkesztője.
- A QtDesigner-ben a modulra felhelyezett objektumok (lineEdit, comboBox stb.) neve, értéke, megjelenése módosítható. Ehhez jelöljük ki az elemet, és a jobb oldali sávon, a Property Editorban változtassuk pl. a gomb szövegét a text tulajdonságnál, és az objectName-nél az objektum neve.

A példa Ujaval Gandhi példáján alapul:
https://www.qgistutorials.com/en/docs/3/building_a_python_plugin.html