QGIS Python API pro tvorbu vlastních nástrojů

Seminář GIS Ostrava 2022

Jan Caha
jan.caha@outlook.com
www.cahik.cz

16. 3. 2022

Motivace

- **zjednodušení pracovních postupů**
- omezení možností nastavení nástrojů
- zřetězení opakujících se úloh
- vytvoření chybějících funkcí

Nástroje QGIS

- dostupné z okna **Processing Toolbox**
    - jedná se o nástroje zcela závislé na vstupních parametrech a fungujících i bez uživatelského rozhraní **QGIS** (např. skrze **qgis_process**)
- neuvažujeme interaktivní nástroje dostupné z nástrojových lišt
    - jedná se o funkčně odlišné nástroje, často postavané na interakci s uživatelem

Možnosti řešení

- **Model Designer** - nástroj pro tvorbu modelů skrze grafické rozhraní
  - snadné na vytváření, ale omezené možnosti
  - pro řadu úloh dostačující, zejména pokud jde o úlohy řetězící dostupné nástroje a upravující nastavení nástrojů
- skripty v **Pythonu**
  - náročnější na tvorbu, ale v podstatě bez omezení ve vztahu k možnostem
  - API QGIS (a dílčích knihoven) + knihovny Pythonu (pozor na dostupnost knihoven u uživatele!!!)

Ukázka nastavení umístění skriptů a modelů v QGIS

Ukázka – Model Designer

Vytvoření bufferu, se specifickým nastavením, rozdělení výsledku na jednoduché polygony (rozdělení případných multi polygonů).

Python skripty

- formálně objekty odovozené od třídy **`QgsProcessingAlgorithm`**, nebo z ní odvozené třídy **`QgsProcessingFeatureBasedAlgorithm`**
- tato třída specifikuje funkce, které je nutné definovat, aby vznikl funkční nástroj
  - **`displayName()`**, **`name()`**, **`groupId()`**, **`group()`**
  - **`createInstance()`**
  - **`processAlgorithm()`**, **`initAlgorithm()`**
- doplňkové funkce:
  - funkčnost **`postProcessAlgorithm()`**, **`checkParameterValues()`**
  - nápověda **`shortHelpString()`**, **`helpUrl()`**
- dokumentace tříd, prvků atd.
- QGIS user guide

Ukázka – vytvoření skriptu v QGIS

Ukázka šablony skriptu v QGIS.

Funkce určující název a zařazení skriptu

- funkce by měly vracet text
- texty pro uživatele - `group()`, `displayName()`
- texty sloužící jako identifikátor pro potřebu **QGIS** - `name()`, `groupId()`

Funkce s dokumentací

- funkce by měly vracet text, v případě nápovědy může být formátovaný jako HTML a zpracován specifickou funkcí
- dokumentace funkce - **`shortHelpString()`**
- odkaz na online dokumentaci - **`helpUrl()`**

Funkce pro interní potřebu QGIS

- `createInstance()` - nezbytná funkce pro fungování skriptu, měla by vracet objekt skriptu

Umožnění překladů textů v nástroji

- defince funkce **`tr()`**, lze pojmenovat i jinak, ale takto je to standard

```
def tr(self, string):
    return QCoreApplication.translate('Processing', string)
```

- použití

```
self.tr("Libovolný text použitý ve skriptu")
```

- např. názvy vstupních parametrů, název nástroje atd.

Definice vstupů a výstupů skriptu

- funkce `initAlgorithm(self, config=None)`
- v této funkci přidáváme vstupy pomocí `self.addParameter()` a výstupy `self.addOutput()`
- vstupní parametery `QgsProcessingParameter*()`, výstupní parametry `QgsProcessingOutput*()`
- některé specifické parametry, např. vrstva pro ukládání vektorových prvků, či rastrů, jsou vstupními parametry (takto se definují) ale zároveň i výstupy

Hlavní část skriptu

- funkce `processAlgorithm(self, parameters, context, feedback)`
- načtení vstupních parametrů do proměnných pomocí funkcí `self.parameterAs*(parameters, name, context)`
- následuje samotné zpracování
- vrací se pythonový typ `dictionary`
    - výstupy a jejich hodnoty

Předávání zpráv z nástroje uživateli

- skrze proměnnou `feedback` ve funkci `processAlgorithm()` (typu `QgsProcessingFeedback`)
- lze vypisovat zprávy, varování i chyby

Ukázka - Skript 1

Ze vstupních polygonů extrahujeme centroidy, kolem nich vyrobíme buffer o zadané velikosti a výslednou geometrii oříznememe do rozsahu původní geometrie.

Ukázka - Skript 2

Zadání shodné se skriptem 1.

Doplníme pouze řešení problému s velikostí bufferu v jednotkách vstupní vrstvy.

Verifikace vstupů před spuštěním skriptu

- funkce `checkParameterValues(self, parameters, context)`
- vstupy se musí načíst stejně jako u hlavní části skriptu (`self.parameterAs*(parameters, name, context)`)
- funkce vrací pythonový typ `tuple`
  - pokud je ok - `True, ""`
  - pokud je chyba - `False, "Algorithm error."`
- vůbec nespouští samotný algorithmus, pokud nejsou splněny podmínky nebo naopak nějaká podmínka platí
- pokus o spuštění algoritmu skončí chybovým oknem s vypsanou chybovou hláškou
- např. algoritmus nelze spustit, pokud vstupní vrstva nemá projektovaný souřadnicový systém

Ukázka - Skript 3

Zadání shodné se skriptem 2.

Doplníme kontroly před spuštěním, povolíme pouze vrstvy, které mají projektované CRS. Hodnota velikosti bufferu musí být větší než 100.

Upravíme formátování nápovědy.

Vracení jiných proměnných než vrstvev

- pro běžné použití často pouze doplňková informace, ale může se hodit
- užitečné zejména pro komplexnější workflow

Ukázka - Skript 4

Zadání shodné se skriptem 3.

Doplníme návratovou hodotu typu Number, kde uživateli vrátíme počet prvků ve vrstvě.

Doplňkové zpracování po dokončení algoritmu

- např. vyčitění zůstatků po běhu algoritmu, nebo nastavení vizualizace
- funkce `postProcessAlgorithm(self, context, feedback)`
- vrací se pythonový typ `dictionary`
    - výstupy a jejich hodnoty

Ukázka - Skript 5

Zadání shodné se skriptem 4.

Po dokončení algoritmu spustíme post processing, který ve vrstvě vyrobí nový dočasný atribut (typu `Expression`) s rozlohou prvku, na základě kterého výstup vizualizujeme jako kartogram.

Použití existujících nástrojů ve vlastním skriptu

- ve skriptu použijeme modul **`processing`**

```
from qgis import processing

processing.run("native:multiparttosingleparts",
               params,
               context=context,
               feedback=feedback)
```

- **`params`** je python **`dictonary`** se vstupními parametry

Ukázka získání paramterů pro volání QGIS nástrojů

Ukázka – Skript 6

Zadání shodné se skriptem 5.

Zpracování dat předřadíme volání nástroje **`Multipart to singleparts`**.

Algoritmy odvozené od `QgsProcessingFeatureBasedAlgorithm`

- implicitně předpokládají existenci vstupní a výstupní vektorové vrstvy, není nutné je specifikovat
- ve funkci `initParameters()` se specifikují pouze další parametry
- funkce `inputLayerTypes()` a `outputWkbType()` specifikují vstupní a výstupní vrstvu
- příprava a načtení parametrů probíhá ve funkci `prepareAlgorithm()`
- zpracování prvku probíhá ve funkci `processFeature(self, feature, context, feedback)`

Algoritmy odvozené od `QgsProcessingFeatureBasedAlgorithm`

- není nutné řešit iterace přes prvky, to algoritmus předpokládá
- předdefinovaný vstup a výstup
- tereoticky jednodušší než použití kompletního algoritmu, ale v praxi se používá méně

Kde brát inspiraci

- většina existujících pluginů má kód dostupný na githubu (případně ho lze dohledat ve složce)
- klíčové pojmy QGIS API se dájí poměrně dobře vyhledávat na GitHubu
- dokumentace a user guide

Děkuji za pozornost.

Dotazy???