
Vývoj geografických aplikací v GIS

Hana Stanková

Lekcia 7

Tvorba vlastného nástroja

Čistenie kódu

- príkazy **try, except, finally**
 - používajú sa, ak chceme zachytiť výnimky v programe (napr. vstupná trieda objektov nemá žiadne objekty)
 - najprv sa vykonajú príkazy za kľúčovým slovom **try**:
 - ak sa neobjaví výnimka, dokončí sa vykonanie príkazov
 - ak sa objaví výnimka, zvyšok príkazov sa preskočí a výnimka sa porovnáva s výnimkami za kľúčovým slovom **except**: (môže ich byť aj viac)
-

Čistenie kódu

- ak sa výnimka zhoduje, vykonajú sa príkazy za `except` a pokračuje sa ďalej za príkazom `try`
 - ak sa výnimka nezhoduje (nezachytí sa), pokračuje sa vo vykonávaní vonkajšieho príkazu `try` alebo sa program zastaví
 - za kľúčové slovo **finally**: môžeme umiestniť príkazy, ktoré chceme aby sa vykonali bez ohľadu na to, či sa výnimka objaví alebo nie (napr. vymazať kurzor)
-

Čistenie kódu

```
Python
>>> try:
...     raise NameError('Ja som vynimka')
... except NameError:
...     print 'Niekde sa stala chyba'
...
Niekde sa stala chyba
>>>
```

Čistenie kódu

- kľúčové slovo **except**: zachytáva výnimky až po najvyššiu úroveň (systémové)
 - príkaz **raise** umožňuje tvorbu vlastných výnimiek (musia byť inštanciami alebo podtriedami triedy Exception)
 - ak sa zadá príkaz **raise** bez argumentov, znovu sa získa výnimka, ktorá je práve aktívna, a vypíše sa chyba (bez tohto by sa len vykonali príkazy za except)
-

Čistenie kódu

```
Python
>>> try:
...     raise NameError('Ja som vynimka')
... except:
...     raise
...
Runtime error <type 'exceptions.NameError'>: Ja som
vynimka
>>>
```

Chyby a správy

- na zachytávanie správ pri spúšťaní nástroja slúži funkcia **arcpy.GetMessages()**
 - na zobrazenie chýb a správ používateľovi slúžia funkcie:
 - **arcpy.AddMessage()** - všeobecné a informatívne správy (závažnosť 0)
 - **arcpy.AddWarning()** - varovné správy (závažnosť 1)
 - **arcpy.AddError()** - chybové hlásenia (závažnosť 2)
-

Chyby a správy

- výpis správ v interaktívnom okne:
print arcpy.GetMessages()
 - funguje to len v Python okne mimo ArcGIS
-

Tvorba nového nástroja

- ArcCatalog
- New - Toolbox
- Add - Script

Add Script

Name: CostDistanceMatrix

Label: Cost Distance Matrix

Description: Vytvorí novú tabuľku, ktorá obsahuje medzi každým prvkom zo zdrojovej a každým prvkom z cieľovej vrstvy (D Nákladová vzdialenosť sa počíta podľa (Cost Raster) po najmenej nákladnej dvojicou zdrojových a cieľových prvkov.

Stylesheet:

Store relative path names (instead of full paths)

Always run in foreground

Add Script

Script File: E:\Data\python_arcgis\exercis

Show command window when script runs

Run Python script in process

Add Script

Display Name	Data Type
Origin Features	Feature Layer
Destination Features	Feature Layer
Cost Raster	Raster Layer
Output Cost Distanc...	Table
@	

Click any parameter above to see its properties below.

Parameter Properties

Property	Value
Type	Required
Direction	Input
MultiValue	No
Default	
Environment	
Filter	None
Obtained from	
Symbology	

To add a new parameter, type the name into an empty row in the name column, click in the Data Type column to choose a data type, then edit the Parameter Properties.

< Zpět Finish Storno

Nastavovanie parametrov

- základné vlastnosti:
 - **Display Name**
 - **Data Type** - Feature Class, Shapefile, File, Text File, String, Table, Raster ...
 - ďalšie vlastnosti:
 - **Type** - Required, Optional, Derived (len výstupné)
 - **Direction** - Input, Output
 - **MultiValue** - Y (zoznam hodnôt), N
-

Nastavovanie parametrov

- ďalšie vlastnosti:
 - **Default** - prednastavená hodnota
 - **Filter** - 6 typov filtrov
 - **Obtained from** - pre vstupné a odvodené výstupné parametre
 - **Symbology** - symbolika (.lyr) pre zobrazovanie výstupov
-

Odvozené parametre

- keď chceme aktualizovať vstupné údaje:

The screenshot shows the 'Update Field Values Properties' dialog box with the 'Parameters' tab selected. The 'Parameters' table lists 'Input features' and 'Output features', both of type 'Feature Class'. The 'Output features' parameter is selected. Below the table, the 'Parameter Properties' section shows the following values:

Property	Value
Type	Derived
Direction	Output
MultiValue	No
Default	
Environment	
Filter	None
Obtained from	Input_features
Symbology	

Two callout boxes provide instructions:

- If you are updating the input data, set Type to Derived ...
- ... and Obtained from to the input data parameter.

Získané parametre

- keď chceme vybrať atribútové pole zo vstupnej vrstvy:

The screenshot shows the 'Add Script' dialog box with a table of parameters. The 'Field to use' parameter is selected. Below the table is the 'Parameter Properties' section, which is expanded to show the 'Obtained from' property set to 'Input_table'.

Display Name	Data Type
Input table	Table
@ Field to use	Field

Click any parameter above to see its properties below.

Parameter Properties

Property	Value
Type	Required
Direction	Input
MultiValue	No
Default	
Environment	
Filter	None
Obtained from	Input_table
Symbology	

For a field data type, Obtained From is the name of the parameter from which to obtain the list of fields.

To add a new parameter, type the name into an empty row in the...

Filtrovanie parametrov

- typy filtrov:
 - **Value List** - zoznam predvolených hodnôt (číselných alebo textových)
 - **Range** - povolený rozsah hodnôt
 - **Feature Class** - zoznam povolených typov (Point, ...)
 - **File** - zoznam povolených koncoviek súborov (.txt, ...)
 - **Field** - zoznam povolených typov atribútov
 - **Workspace** - zoznam povolených typov prostredí
-

Validácia parametrov

- kontroluje, či sú všetky parametre zadané korektne, a ak nie, podá o tom nejakým spôsobom správu
 - vykonáva sa pomocou bloku Python kódu
 - dva druhy validácie:
 - 1. interná (základná) validácia** - ArcGIS ju vykonáva automaticky
 - 2. vlastná validácia** - môžeme vytvoriť vlastný kód
-

Interná validácia

- kontroluje:
 - či sú zadané povinné parametre
 - či je zadaná hodnota správneho typu
 - či zadané hodnoty zodpovedajú filtru
 - či existujú vstupné vrstvy
 - či existujú výstupné vrstvy (podľa nastavenia `overwriteOutput`)
 - generuje predvolenú cestu pre výstupné vrstvy
-

Vlastná validácia

- pomocou vlastnej validácie môžeme:
 - aktualizovať hodnoty filtrov na základe iných parametrov (napr. vstupnej vrstvy)
 - aktivovať/deaktivovať parametre
 - počítat predvolené hodnoty
 - ... a ďalšie
-

Vlastná validácia

- **Script Properties - Validation**
 - trieda **ToolValidator** obsahuje tieto metódy:
 - **__init__** - inicializuje triedu, importuje knižnice, inicializuje objekt (self)
 - **initializeParameters** - volá sa, keď sa prvý krát otvorí dialógové okno (alebo sa nástroj spustí z prík. riadku)
 - **update Parameters** - volá sa pri každej zmene parametra, následne sa vykoná interná validácia
-

Vlastná validácia

- trieda **ToolValidator** obsahuje tieto metódy:
 - **updateMessages** - volá sa po skončení internej validácie
 - metódy môžeme editovať cez Python editor
 - na začiatku obsahujú iba príkaz return
 - musia byť všetky tri prítomné, aby bol trieda ToolValidator validná
-

Vlastná validácia

The screenshot shows the 'Script Properties' dialog box with the 'Validation' tab selected. The code for the 'ToolValidator' class is displayed, with several lines highlighted in blue. A yellow callout box points to the class definition and its methods. Another yellow callout box points to the 'Edt...' button at the bottom right of the dialog.

```
class ToolValidator:
    """Class for validating a tool's parameter values
    and the behavior of the tool's dialog."""

    def __init__(self):
        """Setup arcpy and the list of tool parameters"""
        import arcpy
        self.params = arcpy.GetParameterInfo()

    def initializeParameters(self):
        """Define the properties of a tool's parameters
        called when the tool is opened."""
        return

    def updateParameters(self):
        """Modify the values and properties of parameter
        validation is performed. This method is called
        has been changed."""
        return

    def updateMessages(self):
        """Modify the messages created by internal validation
        parameter. This method is called after internal
        return
```

Default ToolValidator class and methods

Click the Edit button to open the Python editor.

The code is being edited. Apply changes by clicking OK or Apply after you save your changes and close your Python editor.

Edt...

Vlastná validácia

- príklad - aktivovanie/deaktivovanie parametrov:

```
def updateParameters(self):  
    # If the option to use a weights file is selected (the user chose  
    # "Get Spatial Weights From File"), enable the parameter for specifying  
    # the file, otherwise disable it  
    #  
    if self.params[3].value == "Get Spatial Weights From File":  
        self.params[8].enabled = 1  
    else:  
        self.params[8].enabled = 0
```

Preberanie parametrov

- dva spôsoby preberania parametrov:
 1. funkcia **arcpy.GetParametersAsText()**
 2. metóda **sys.argv** - limitovaný počet znakov
sys.argv[0] - názov súboru so skriptom

```
# Read the parameter values:  
# 1: input workspace  
# 2: input clip features  
# 3: output workspace  
#  
inWorkspace = arcpy.GetParameterAsText(0)  
clipFeatures = arcpy.GetParameterAsText(1)  
outWorkspace = arcpy.GetParameterAsText(2)  
env.workspace = inWorkspace
```

```
# Read the parameter values:  
# 1: input workspace  
# 2: input clip features  
# 3: output workspace  
#  
inWorkspace = sys.argv[1]  
clipFeatures = sys.argv[2]  
outWorkspace = sys.argv[3]  
env.workspace = inWorkspace
```

Dokumentácia nástroja

- Tool - Item Description

