Lekce 1: Úvod do automatizace GIS pomocí Model Builderu

Programování pro GIS

Fakulta životního prostředí, ČZU Praha

2025-01-01

Table of contents

Informace o lekci

Časová dotace: 90 minut (1,5 hodiny)
Předpoklady: Základní znalost ArcGIS Pro a Model Builderu (GIS 1, GIS 2)
Materiály: ArcGIS Pro projekt ke stažení na [odkaz]

1 Úvod do kurzu

1.1 Co se v tomto kurzu naučíte

Tento kurz vás provede cestou od vizuálního programování v Model Builderu k psaní vlastních Python skriptů pro automatizaci GIS úloh.

1.1.1 Týdny 1-2

Model Builder → Python

- Úvod do automatizace
- První pohled na Python kód
- Export modelů
- Iterátory a limity Model Builderu

1.1.2 Týdny 3-5

Základy programování v Pythonu

- Proměnné, cykly, funkce
- Práce se soubory a CSV
- Algoritmické myšlení

- Třídící algoritmy

1.1.3 Týdny 6-9

ArcPy - Python pro GIS

- Automatizace analýz
- Tabulkové operace
- Vektorové a rastrové analýzy
- Kurzory a geometrie

1.1.4 Týdny 10-11

Pokročilé techniky

- Práce s geometriemi
- Python Toolboxy
- Tvorba nástrojů s GUI
- Optimalizace kódu

1.1.5 Týden 12

Závěrečný projekt

- Vlastní nástroj
- Řešení reálného problému
- Dokumentace

1.2 Proč tento kurz?

Už umíte pracovat s ArcGIS Pro - znáte nástroje, dokážete provádět analýzy, vytvářet mapy. **Nyní se naučíte GIS automatizovat a programovat.**

Příklady z praxe

Opakující se úlohy
"Každý měsíc musíme aktualizovat mapy dostupnosti zdravotnických zařízení pro 14 krajů."
→ **Řešení:** Napsat skript, který to udělá automaticky za 5 minut.
Velké množství dat
"Potřebujeme zpracovat 500 rastrových snímků z družice."
→ **Řešení:** Python skript běžící přes noc, vy ráno kontrolujete výsledky.
Složité analýzy
"Chceme optimalizovat umístění větrných elektráren na základě 10 různých kritérií."
→ **Řešení:** Vlastní nástroj s GUI, který může používat kdokoli v týmu.

1.3 Struktura výuky

Každý týden:

- 2× 1,5 hodiny praktických cvičení
- Kombinace výkladu + samostatné práce
- Volitelné úkoly k procvičení

Hodnocení:

- Aktivita na cvičeních
- Průběžné úkoly (malé, týdenní)
- **Závěrečný projekt** (hlavní část hodnocení)
  - Funkční Python nástroj
  - Řeší reálný GIS problém
  - S dokumentací

Nástroje:

- ArcGIS Pro (máte nainstalované?)
- Python 3.x (součást ArcGIS Pro)
- Textový editor (Notepad++, VS Code)
- Později: Jupyter Notebook

2 Praktická úloha

2.1 Zadání

> ! Analytická úloha
>
> **Jaké je zastoupení lesů v ochranném pásmu 500 metrů kolem železnic v okrese Jindřichův Hradec?**

2.1.1 Proč tato úloha?

- **Realistická** - ochranná pásma, hlukové mapování, dostupnost
- **Jednoduchá** - pochopitelné kroky
- **Rozšiřitelná** - později přidáme iterátory a Python
- **Ukáže limity** Model Builderu → motivace pro Python

2.2 Data

Všechna data jsou připravena v projektu ArcGIS Pro, který si stáhnete z [odkaz].

2.2.1 Okresy

`okresy` - polygony okresů ČR

- Souřadnicový systém: EPSG:3035
- Klíčové pole: NAZ_LAU1 (název okresu)

2.2.2 Železnice

`zeleznice` - linie železnic ČR

- Souřadnicový systém: EPSG:3035
- Geometrie: polyline

2.2.3 Krajinný pokryv

`clc_2018` - rastr Corine Land Cover

- Souřadnicový systém: EPSG:3035
- Hodnoty:
  - 1xx = Urbanizované plochy
  - 2xx = Zemědělská půda
  - 3xx = Lesy
  - 4xx = Mokřady
  - 5xx = Vodní plochy

> **Note**
>
> **Poznámka:** Data jsou už transformována do jednotného souřadnicového systému ETRS89 LAEA (EPSG: 3035).

2.3 Analytický postup

Cíl: Zjistit, kolik procent plochy v pásmu 500m od železnic v okrese Jindřichův Hradec tvoří lesy.

```
flowchart TD
    A[Okresy] --> B[Vybrat JH]
    B --> C[Okres JH]
    D[železnice] --> E[Oříznout okresem]
    C --> E
    E --> F[železnice v JH]
    F --> G[Buffer 500m]
    G --> H[Pásmo 500m]
    I[CLC rastr] --> J[Binární rastr lesů]
```

```
    H --> K[Zónální statistika]
    J --> K
    K --> L[Výsledek]

    style C fill:#6baed6
    style F fill:#6baed6
    style H fill:#6baed6
    style J fill:#74c476
    style L fill:#fd8d3c
```

2.4 Krok za krokem

2.4.1 Příprava

1. Otevřete ArcGIS Pro projekt `Lekce1_AutomatizaceGIS.aprx`
2. Prohlédněte si data v mapě
3. Vytvořte nový toolbox:
    - Pravý klik v Catalog Pane → New → Toolbox
    - Pojmenujte: `Lekce1_Tools.atbx`
4. Vytvořte nový model:
    - Pravý klik na toolbox → New → Model
    - Pojmenujte: `Analyza_Lesu_v_Pasmu`

2.4.2 Krok 1: Výběr okresu Jindřichův Hradec

Nástroj: Make Feature Layer

> Proč Make Feature Layer?
>
> Nástroj *Select* vytváří novou datovou sadu na disku. *Make Feature Layer* vytváří pouze dočasnou vrstvu v paměti, což je rychlejší a efektivnější.

Postup:

1. V modelu: Insert → Tool → vyhledat "Make Feature Layer"
2. Přetáhněte vrstvu `okresy` do modelu
3. Propojte `okresy` s nástrojem Make Feature Layer
4. Dvojklik na nástroj → nastavit parametry:
    - **Input Features:** okresy
    - **Output Layer:** `okres_jh_layer`
    - **Expression:** Klikněte SQL

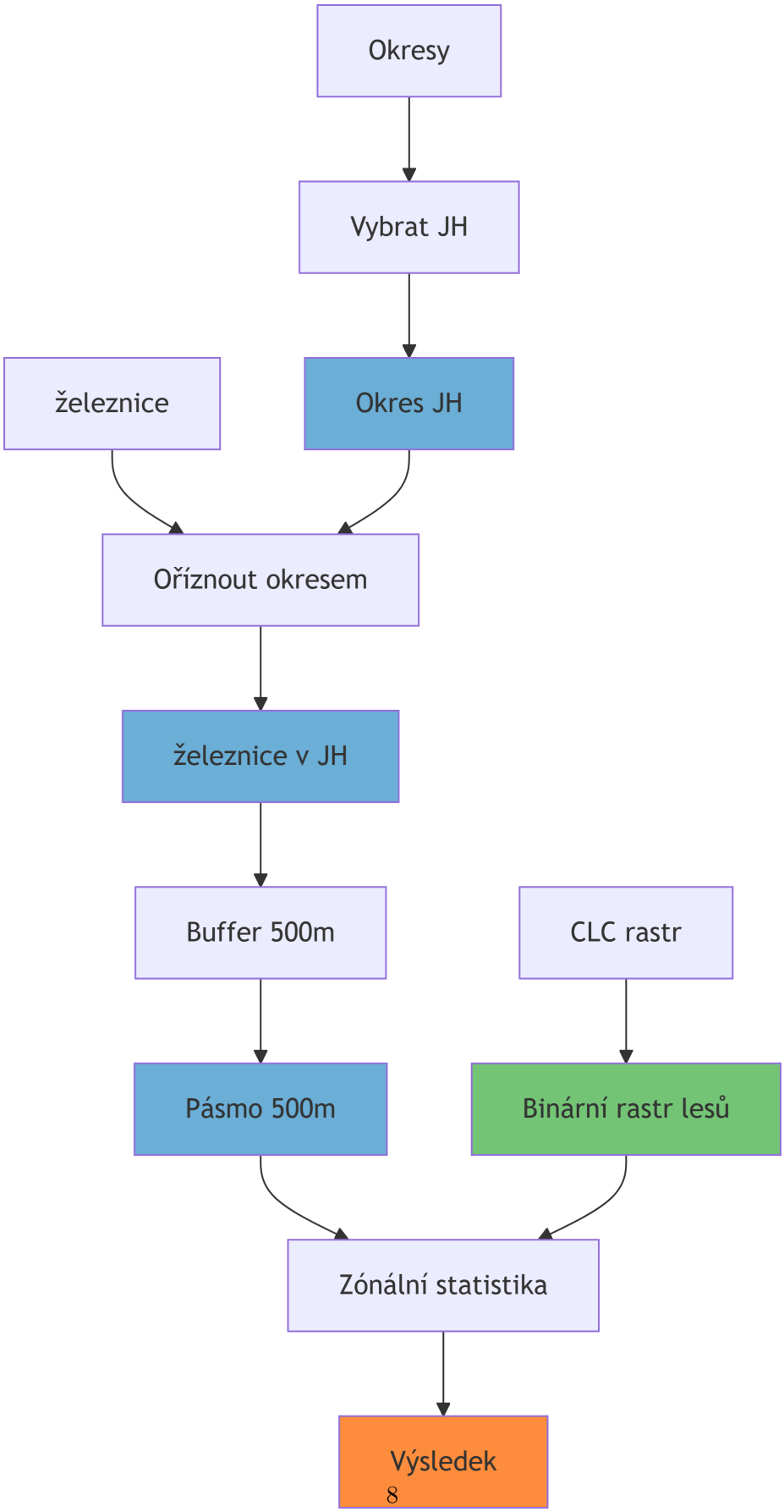


Figure 1: Workflow analytického postupu

```
NAZ_LAU1 = 'Jindřichův Hradec'
```

5. OK

Kontrola: `okresy` → `Make Feature Layer` → `okres_jh_layer`

2.4.3 Krok 2: Ořezání železnic na okres

Nástroj: Clip

Účel: Z celé vrstvy železnic chceme jen úseky, které jsou v okrese JH.

Postup:

1. Insert → Tool → "Clip"
2. Přetáhněte vrstvu `dalnice` do modelu
3. Propojte:
   - `dalnice` → Clip (jako Input Features)
   - `okres_jh_layer` → Clip (jako Clip Features)
4. Dvojklik na Clip → parametry:
   - **Input Features:** dalnice
   - **Clip Features:** okres_jh_layer
   - **Output:** `dalnice_clip`
5. OK

2.4.4 Krok 3: Vytvoření ochranného pásma (buffer)

Nástroj: Buffer

> ❗ Důležité nastavení
>
> **Dissolve Type = ALL** (spojí všechny buffery do jednoho)

Postup:

1. Insert → Tool → "Buffer"
2. Propojte `dalnice_clip` → Buffer
3. Dvojklik na Buffer → parametry:
   - **Input Features:** dalnice_clip
   - **Output:** `buffer_500m`
   - **Distance:** 500 Meters
   - **Dissolve Type: ALL** ← důležité!
   - Side Type: FULL

- End Type: ROUND

4. OK

> **Proč Dissolve ALL?**
>
> Bez dissolve bychom měli desítky překrývajících se bufferů (jeden pro každý úsek železnice). S ALL se všechny spojí do jednoho (multi)polygonu.
> Díky tomu dostaneme v zonální statistice přímo jeden výsledek - jedno číslo představující průměr z celého pásma.

2.4.5 Krok 4: Vytvoření binárního rastru lesů

Nástroj: Equal To (Spatial Analyst)

Účel: Z CLC rastru (hodnoty 1,2,3,4,5) vytvořit rastr s hodnotami 0/1, kde 1 = les.

Postup:

1. Insert → Tool → "Equal To" (v kategorii Spatial Analyst → Math → Logical)
2. Přetáhněte rastr `clc_2018` do modelu
3. Propojte `clc_2018` → Equal To
4. Dvojklik na Equal To:
   - **Input raster:** clc_2018
   - **Input value:** 3 (kód pro lesy)
   - **Output:** `lesy_binarni`
5. OK

> **Co se stane?**
>
> Rastr bude mít hodnotu 1 tam, kde je les (CLC=3), a hodnotu 0 všude jinde.

2.4.6 Krok 5: Zonální statistika

Nástroj: Zonal Statistics as Table

Účel: Spočítat průměr z binárního rastru v rámci bufferu.

> **Matematický trik**
>
> Průměr z nul a jedniček = podíl jedniček = relativní plocha lesů!
> Pokud je průměr 0.35, znamená to, že 35% pixelů má hodnotu 1 (les).

Postup:

1. Insert → Tool → "Zonal Statistics as Table"

2. Propojte:
    - `buffer_500m` → Zonal Statistics (jako Input Zone Data)
    - `lesy_binarni` → Zonal Statistics (jako Input Value Raster)
3. Dvojklik na Zonal Statistics:
    - **Input Zone Data:** buffer_500m
    - **Zone Field:** OBJECTID
    - **Input Value Raster:** lesy_binarni
    - **Output Table:** `vysledek_lesy.dbf`
    - **Statistics Type:** MEAN (průměr)
    - **Ignore NoData:** zaškrtnuto
4. OK

2.4.7 Uložení a spuštění modelu

2.4.8 1. Uložit

File → Save (Ctrl+S)

2.4.9 2. Ověřit

Model → Validate Entire Model

Pokud je vše OK, všechny nástroje budou barevné (ne šedé)

2.4.10 3. Spustit

Klikněte na Run

2.4.11 4. Počkat

Model běží, sledujte progress

2.4.12 5. Výsledek

Otevřete `vysledek_lesy.dbf`

2.5 Interpretace výsledku

V tabulce `vysledek_lesy.dbf` najdete sloupec **MEAN**.

Význam:

- MEAN = 0.354 → **35.4% plochy v pásmu tvoří lesy**
- MEAN = 0.205 → **20.5% plochy v pásmu tvoří lesy**

> **Proč to funguje?**
>
> Průměr z binárního rastru (0/1) v dané zóně = podíl pixelů s hodnotou 1 = relativní plocha lesů. Díky nastavení **Dissolve ALL** v bufferu máme jeden (multi)polygon, takže dostaneme přímo jedno číslo - procento lesů v celém pásmu kolem železnic v okrese.

2.5.1 A co kdyby více vzdáleností?

Situace:
"Váš šéf říká: 'Chci vidět, jak se to mění s vzdáleností. Spočítej to pro 100m, 300m, 500m a 1000m.' "

> **Problém**
>
> **Co byste museli udělat?**
>
> 1. Změnit Buffer distance na 100m → spustit
> 2. Změnit na 300m → spustit
> 3. Změnit na 500m → spustit (už máme)
> 4. Změnit na 1000m → spustit
>
> **= 4× ručně spustit model, pokaždé změnit parametr**
> **A pak:** 4 samostatné tabulky → jak je dát dohromady pro porovnání?

> **Řešení**
>
> **Příští týden:** Naučíme se **ITERÁTORY** - automatické procházení různých hodnot
> **Za měsíc:** Naučíme se **PYTHON** - elegantní řešení s vnořenými cykly

3 Export modelu do Pythonu

3.1 Proč exportovat?

Model Builder je skvělý pro vizualizaci workflow, ale má limity:

- Těžko se verzuje (Git, SVN)

- Složité sdílení (musíte sdílet celý toolbox)
- Omezené možnosti logiky (podmínky, cykly)

Python nám dává:

- Textový soubor (snadno sdílitelný, verzovatelný)
- Možnost úprav v textovém editoru
- Přidání vlastní logiky
- Spuštění mimo ArcGIS Pro (automatizace)

3.2 Jak exportovat

1. V Model Builderu: **Model → Export → To Python Script**
2. Uložit jako: `model_export.py`
3. Vybrat lokaci a uložit

3.3 Prohlédnutí Python kódu

Otevřete exportovaný soubor v textovém editoru:

- **Notepad++** (doporučeno - zvýrazňuje syntax)
- **VS Code** (pokud máte)
- **Poznámkový blok** (funguje, ale bez barev)

3.3.1 Struktura kódu

3.4 Co vidíme v kódu?

3.4.1 1. Import ArcPy

```
import arcpy
```

> **Note**
>
> **Význam:** "Chci použít nástroje ArcGIS v Pythonu"
> **Analogie:** Jako když v ArcGIS Pro otevřete ArcToolbox - získáte přístup k nástrojům.

3.4.2 2. Proměnné

```
okresy = "okresy"
vzdalenost = "500 Meters"
```

Tip

Výhoda proměnných: Můžeme snadno změnit na jednom místě:

```
vzdalenost = "1000 Meters"  # Změna parametru!
```

3.4.3 3. Volání nástroje

```
arcpy.Buffer_analysis(
    in_features=dalnice_clip,
    out_feature_class=buffer_500m,
    buffer_distance_or_field="500 Meters",
    dissolve_option="ALL"
)
```

Srovnání s Model Builderem:

Table 1: Srovnání Model Builder vs. Python

Model Builder	Python
Žlutý obdélník "Buffer"	`arcpy.Buffer_analysis()`
Dialog s parametry	Parametry v závorkách
Propojení šipkou	Proměnné jako parametry
Kliknutí na Run	`python script.py`

Klíčové poznání

Je to STEJNÉ, jen jinak zapsané!

3.5 Interaktivní prozkoumání

Úkol 1: Najděte v kódu

Kde je napsáno "Jindřichův Hradec"?

```
where_clause="NAZ_LAU1 = 'Jindřichův Hradec'"
```

Úkol 2: Najděte v kódu

Kde je vzdálenost bufferu?

```
buffer_distance_or_field="500 Meters"
```

Úkol 3: Najděte v kódu

Kde se vytváří binární rastr lesů?

```
arcpy.gp.EqualTo_sa(
    in_raster_or_constant1=clc_2018,
    in_raster_or_constant2="3",  # ← tady je kód pro lesy
    out_raster=lesy_binarni
)
```

3.6 Co by se dalo snadno změnit v Pythonu?

3.6.1 Změna vzdálenosti

```
# Místo:
buffer_distance_or_field="500 Meters"

# Můžeme:
vzdalenost = 1000  # metry
buffer_distance_or_field=f"{vzdalenost} Meters"
```

3.6.2 Jiný okres

```
# Místo:
where_clause="NAZ_LAU1 = 'Jindřichův Hradec'"

# Můžeme:
okres = "Praha-východ"
where_clause=f"NAZ_LAU1 = '{okres}'"
```

3.6.3 Více vzdáleností najednou

```
# V Pythonu bychom mohli:
vzdalenosti = [100, 300, 500, 1000]
```

```python
vysledky = []

for vzd in vzdalenosti:
    # Buffer
    buffer = arcpy.Buffer_analysis(..., f"{vzd} Meters")

    # Zonal Statistics
    vysledek = arcpy.ZonalStatisticsAsTable(...)

    # Uložit výsledek
    vysledky.append(vysledek)

# Hotovo! Všechny vzdálenosti v jednom běhu!
```

Vidíte?

V Pythonu můžeme snadno:

- Měnit parametry
- Přidávat výpočty
- Automatizovat opakování
- **Spojovat výsledky do jedné struktury!**

3.7 Výhody Pythonu vs. Model Builder

Table 2: Srovnání Model Builder vs. Python

Aspekt	Model Builder	Python
Vizualizace	Výborná	Žádná (jen text)
Rychlé vytvoření	Drag & drop	Musíte psát
Sdílení	Toolbox soubor	Textový .py soubor
Verzování (Git)	Binární formát	Textový formát
Podmínky (IF)	Omezené	Plná podpora
Cykly (FOR)	Jen iterátory	Plná flexibilita
Výpočty	Calculate Field	Jakékoli operace
Debugging	Obtížné	Snadné
Rychlost běhu	Pomalejší	Rychlejší
Spojování výsledků	Velmi složité	Jednoduché (seznamy)

! Závěr

Model Builder = skvělý start, vizuální, rychlý pro jednoduché úlohy
Python = mocný nástroj pro opakování, složitou logiku, automatizaci

4 Možnosti rozšíření

4.1 Různé vzdálenosti - motivace pro iterátory

Scénář: Chceme analyzovat 4 různé vzdálenosti: 100m, 300m, 500m, 1000m

4.1.1 V Model Builderu bez iterátoru

Museli byste:

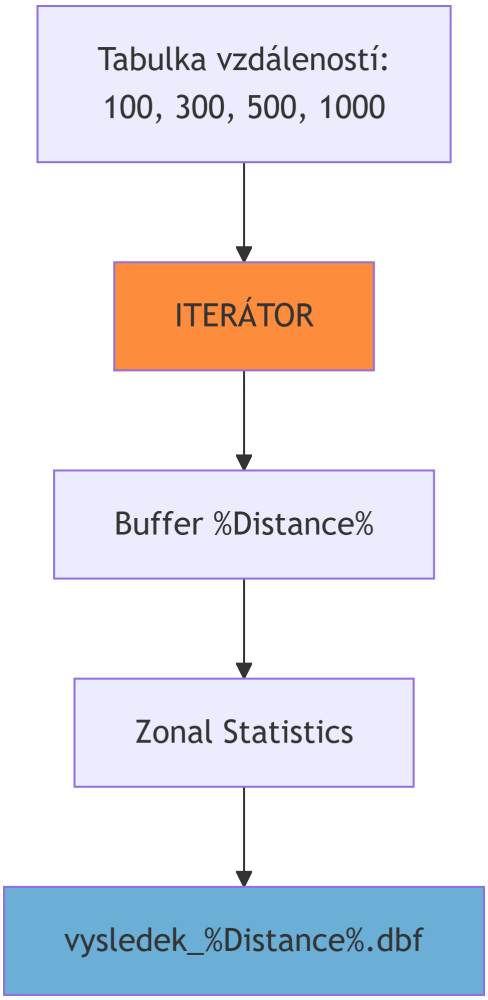
1. Změnit Buffer distance na 100m → Spustit → `vysledek_100.dbf`
2. Změnit na 300m → Spustit → `vysledek_300.dbf`
3. Změnit na 500m → Spustit → `vysledek_500.dbf`
4. Změnit na 1000m → Spustit → `vysledek_1000.dbf`

Čas: 15-20 minut
Problém: Nudné, náchylné k chybě, 4 samostatné tabulky

4.1.2 S iterátorem (příští týden)

```
flowchart TD
    A[Tabulka vzdáleností:<br/>100, 300, 500, 1000] --> B[ITERÁTOR]
    B --> C[Buffer %Distance%]
    C --> D[Zonal Statistics]
    D --> E[vysledek_%Distance%.dbf]

    style B fill:#fd8d3c
    style E fill:#6baed6
```



Model s iterátorem

> **Výhoda**
>
> Spustíte **jednou**, iterator automaticky projde všechny vzdálenosti!

> **Ale…**
>
> Dostanete **4 samostatné tabulky**:
>
> - vysledek_100.dbf → MEAN = 0.42
> - vysledek_300.dbf → MEAN = 0.38
> - vysledek_500.dbf → MEAN = 0.35
> - vysledek_1000.dbf → MEAN = 0.31
>
> **Jak je spojíte do jedné pro porovnání?**

V Model Builderu složité (Add Field + Calculate Field + Merge pro každou tabulku).
V Pythonu jednoduché (seznam)!

4.2 Kombinace parametrů - motivace pro Python

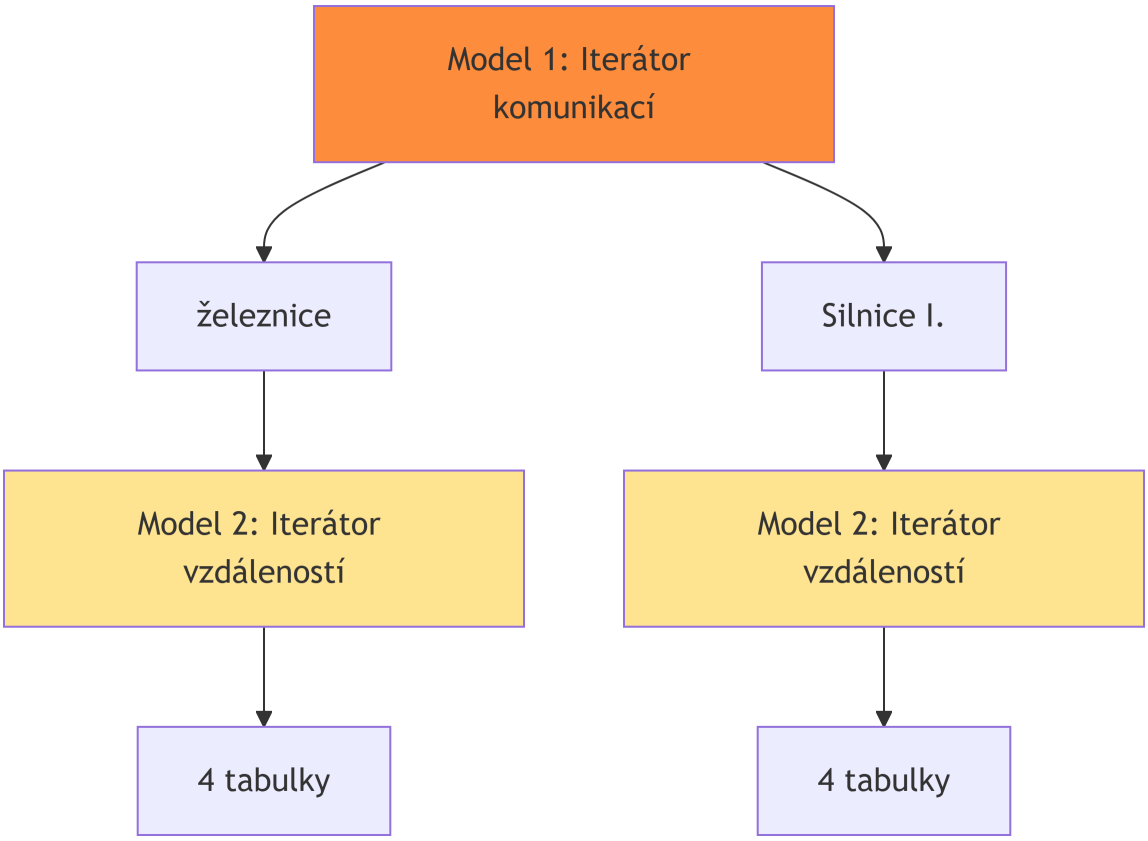
Scénář: Chceme analyzovat:

- **2 typy komunikací** (železnice, silnice I. třídy)
- × **4 vzdálenosti** (100, 300, 500, 1000)
- = **8 kombinací**

4.2.1 V Model Builderu

```
flowchart TD
    A[Model 1: Iterátor komunikací] --> B[železnice]
    A --> C[Silnice I.]
    B --> D[Model 2: Iterátor vzdáleností]
    C --> E[Model 2: Iterátor vzdáleností]
    D --> F[4 tabulky]
    E --> G[4 tabulky]

    style A fill:#fd8d3c
    style D fill:#fee391
    style E fill:#fee391
```



Vnořené modely (složité!)

> ⚠ Problémy
>
> 1. Potřebujete **2 vnořené modely** (složité nastavení!)
> 2. Model 1 (vnější) volá Model 2 (vnitřní)
> 3. Výsledek: **8 samostatných tabulek**
> 4. Jak je spojit? Velmi složité…

4.2.2 V Pythonu

```
komunikace = ['dalnice', 'silnice1']
vzdalenosti = [100, 300, 500, 1000]

vysledky = []

for kom in komunikace:
    for vzd in vzdalenosti:
        vysledek = analyzuj(kom, vzd)
```

```
        vysledky.append({
            'Komunikace': kom,
            'Vzdalenost': vzd,
            'Procento_lesu': vysledek
        })

# Jedna tabulka, 8 řádků!
uloz_tabulku(vysledky, 'vsechny_vysledky.csv')
```

! Vidíte rozdíl?

Model Builder: 2 modely, 8 tabulek, složité spojování
Python: Vnořený for cyklus (5 řádků), hotovo!

5 Shrnutí

5.1 Co jsme se dnes naučili

Přehled lekce

Struktura kurzu

- 12 týdnů od Model Builderu k Pythonu
- Praktické příklady motivace

Praktická úloha

- Analýza krajinného pokryvu v ochranných pásmech
- 5 kroků: výběr → clip → buffer → binární rastr → zonální statistika
- Výsledek: procento lesů v pásmu 500m kolem železnic

Export do Pythonu

- Model = Python kód
- První pohled na Python syntax
- Srovnání Model Builder vs. Python

Limity Model Builderu

- Opakování = ruční spouštění nebo iterátory
- Více tabulek = složité spojování
- Vnořené cykly = velmi složité
- **Motivace pro Python!**

5.2 Co nás čeká příště (Lekce 2)

5.2.1 První část - Iterátory

- **ITERÁTORY** v Model Builderu
- Iterate Field Values - procházení různých vzdáleností
- Automatické opakování
- **Problém:** 4 samostatné tabulky - jak spojit?

5.2.2 Druhá část - Model Tool

- Z modelu vytvoříme **nástroj s GUI**
- Parametry: uživatel si vybere vzdálenost
- Nástroj můžete sdílet s kolegy
- Použití v dalších modelech

5.2.3 Třetí část - Vnořené modely

- Model volá jiný model
- Způsob, jak obejít "max 1 iterátor"
- **Ukáže limity** Model Builderu
- **Motivace** pro Python vnořené cykly

5.3 Klíčové pojmy

Table 3: Klíčové pojmy z lekce

Pojem	Význam
Automatizace	Opakované spouštění úloh bez lidského zásahu
Model Builder	Nástroj pro vizuální tvorbu workflow
Workflow	Posloupnost kroků vedoucí k výsledku
Iterator	Mechanismus pro automatické opakování (příště)
ArcPy	Python knihovna pro ArcGIS
Zonální statistika	Výpočet statistik v definovaných zónách
Binární rastr	Rastr s hodnotami 0/1 (ano/ne)
Dissolve	Spojení více prvků do jednoho

5.4 Domácí úkol (volitelný)

Procvičte si látku pomocě volitelných úkolů níže.

Doporučení

- Začněte **Úkolem 1** (lehký) - určitě zvládnete!
- Pokud vás to baví, zkuste **Úkol 2** nebo **3**
- **Úkol 4** je pro prozkoumání Python kódu
- **Úkol 5 (BONUS)** je záměrně velmi těžký - ukáže vám limity MB

6 Volitelné úkoly

6.1 Úkol 1: Změna parametrů modelu

Obtížnost: Lehká

Cíl: Naučit se měnit parametry v modelu a vidět, jak to ovlivní výsledky.

Zadání:
Upravte svůj model tak, aby analyzoval **pásmo 300 metrů** (místo 500m) kolem železnic v okrese Jindřichův Hradec.

Očekávaný výsledek:

- Upravený model s bufferem 300m
- Nová výsledná tabulka
- Porovnání: je procento lesů v pásmu 300m vyšší nebo nižší než v 500m? Proč?

Postup

1. Otevřete svůj model `Analyza_Lesu_v_Pasmu`
2. Dvojklik na nástroj Buffer
3. Změňte Distance: `500 Meters` → `300 Meters`
4. Změňte název výstupu: `buffer_500m` → `buffer_300m`
5. Změňte název výsledné tabulky: `vysledek_lesy.dbf` → `vysledek_lesy_300m.dbf`
6. Uložte a spusťte model
7. Porovnejte výsledky (MEAN hodnoty)

Otázky k zamyšlení:

- Je procento lesů v užším pásmu (300m) jiné než v širším (500m)?
- Jak byste to vysvětlili? (Nápověda: rozmístění lesů vs. železnic)

Bonus

Vytvořte tabulku v Excelu s porovnáním:

Vzdálenost	Procento lesů
300m	X.X%
500m	Y.Y%

Vytvořte graf závislosti procenta lesů na vzdálenosti od železnic.

6.2 Úkol 2: Jiný okres

Obtížnost: Střední

Cíl: Pochopit, jak změnit atributový dotaz v modelu.

Zadání:
Upravte model tak, aby analyzoval okres **Praha-východ** (místo Jindřichův Hradec).

Očekávaný výsledek:

- Model fungující pro okres Praha-východ
- Výsledná tabulka s procentem lesů
- Porovnání: má Praha-východ více nebo méně lesů v pásmech kolem železnic než JH?

Postup

1. Nejdřív zjistěte přesný název okresu:
   - Otevřete atributovou tabulku vrstvy `okresy`
   - Najděte pole NAZ_LAU1
   - Najděte řádek s Prahou-východ (může být “Praha-východ” nebo “Praha - východ”)
2. V modelu: dvojklik na Make Feature Layer
3. Změňte Expression:

   ```
   NAZ_LAU1 = 'Praha-východ'
   ```

   (Pozor na přesný zápis!)
4. Změňte názvy výstupů, aby bylo jasné, že jde o jiný okres
5. Spusťte model

Nápověda

- Pokud model hlásí "0 features selected", zkontrolujte přesný název okresu v datech
- Může být potřeba použít `LIKE` místo `=`:

```
NAZ_LAU1 LIKE '%Praha%východ%'
```

Bonus

Vytvořte srovnávací tabulku pro 3-5 různých okresů:

Okres	Procento lesů
Jindřichův Hradec	35.4%
Praha-východ	?
Prachatice	?
...	...

Který okres má nejvíce lesů kolem železnic?

6.3 Úkol 3: Jiný typ krajinného pokryvu

Obtížnost: Střední

Cíl: Naučit se analyzovat různé kategorie dat změnou jednoho parametru.

Zadání:
Analyzujte zastoupení **zemědělské půdy** (CLC kód 2) místo lesů v pásmu 500m kolem železnic v okrese JH.

Očekávaný výsledek:

- Model analyzující zemědělskou půdu
- Porovnání: je v pásmu více lesů nebo zemědělské půdy?

Postup

1. V modelu: dvojklik na nástroj Equal To
2. Změňte Input value: `3` → `2`
3. Změňte názvy výstupů:
   - `lesy_binarni` → `zempuda_binarni`
   - `vysledek_lesy.dbf` → `vysledek_zempuda.dbf`
4. Spusťte model

Rozšíření

Vytvořte tabulku se všemi typy krajinného pokryvu:

CLC kód	Typ	Procento
1	Urbanizované plochy	?
2	Zemědělská půda	?
3	Lesy	35.4%
4	Mokřady	?
5	Vodní plochy	?

(Musíte spustit model 5× s různými kódy)

Otázka k zamyšlení

Je tento postup efektivní? Co kdybyste chtěli 10 kategorií? 50?
Odpověď: Proto se naučíme iterátory (příště) a Python (za měsíc)!

6.4 Úkol 4: Export a zkoumání Python kódu

Obtížnost: Lehká

Cíl: Seznámit se s Python syntaxí na vašem vlastním modelu.

Zadání:
Exportujte váš model do Pythonu a prozkoumejte kód.

Postup

1. V Model Builderu: Model → Export → To Python Script
2. Uložte jako `muj_model.py`
3. Otevřete v textovém editoru (Notepad++, VS Code, nebo Poznámkový blok)

Úkoly v kódu:

6.4.1 A) Najděte

Najděte a zvýrazněte:

- Řádek s importem arcpy
- Řádek, kde se vytváří buffer 500m
- Řádek s SQL dotazem pro okres
- Řádek s hodnotou pro lesy (3)

6.4.2 B) Experimentujte

Experimentujte (bez spouštění!):

- Zkuste změnit `"500 Meters"` na `"1000 Meters"` - na kterém řádku?
- Najděte místo, kde byste změnili okres na jiný
- Kolik řádků by bylo potřeba změnit, abyste změnili vzdálenost? A v modelu?

6.4.3 C) Přemýšlejte

Přemýšlejte:

- Je kód čitelný? Rozumíte alespoň trochu, co dělá?
- Které části jsou jasné, které ne?
- Vidíte výhody textové podoby vs. grafické?
- Jak by se v Pythonu řešilo 4 různé vzdálenosti?

> ⚠ Warning
>
> **Poznámka:** Kód zatím nespouštějte - to se naučíme příště. Teď jen pozorujte strukturu.

6.5 Úkol 5 (BONUS): Čtyři vzdálenosti - ukázka problému

> ⚠ Obtížnost: Velmi těžká
>
> **VAROVÁNÍ:** Tento úkol je záměrně obtížný! Jeho cílem je ukázat vám problém, který v příští lekci vyřešíme iterátorem, a za měsíc elegantně v Pythonu.
> Nebojte se, pokud se vám to nepodaří - právě proto se učíme Python!

Zadání:
Spočítejte procento lesů pro **4 různé vzdálenosti**: 100m, 300m, 500m, 1000m

Očekávaný výsledek:
Tabulka (v Excelu nebo jako poznámky):

Vzdálenost	Procento lesů
100m	?
300m	?
500m	35.4%
1000m	?

6.5.1 Přístup A (manuální)

Přístup A (nejjednodušší, ale nudný):

1. Spusťte model s 100m → zapište výsledek
2. Změňte na 300m, spusťte → zapište výsledek
3. Změňte na 500m, spusťte → zapište výsledek (už máte)
4. Změňte na 1000m, spusťte → zapište výsledek

Měřte čas: Kolik celkem trvalo všech 4 spuštění?

6.5.2 Přístup B (preview iterátoru)

Přístup B (pokročilý - preview na příští týden):

Pokud se chcete pokusit o iterátor již nyní:

1. Vytvořte v Excelu nebo jako DBF tabulku se vzdálenostmi:

   ```
   Distance
   100
   300
   500
   1000
   ```

2. Přidejte do modelu **Iterate Field Values**
3. Zkuste propojit s bufferem…

Poznámka: Pravděpodobně narazíte na problémy! To je v pořádku - příští týden to společně vyřešíme.

6.5.3 Reflexe (důležitější než řešení!)

Po dokončení napište:

1. **Kolik času vám to zabralo?**
2. **Kolik chyb jste udělali?** (zapomenuté změny parametru, špatné názvy…)
3. **Jak byste se cítili, kdyby zadání bylo 20 vzdáleností?**
4. **Vidíte potřebu automatizace?**

❗ Ukázka Python řešení (jen se podívejte)

```python
# V Pythonu by to vypadalo takto (nemusíte rozumět detailům):

vzdalenosti = [100, 300, 500, 1000]
vysledky = []

for vzd in vzdalenosti:
    # Buffer
    buffer = arcpy.Buffer_analysis(
        dalnice_clip,
        f"buffer_{vzd}m",
        f"{vzd} Meters",
        dissolve_option="ALL"
    )

    # Equal To
    binary = arcpy.sa.EqualTo(clc_2018, 3)

    # Zonal Statistics
    stats = arcpy.sa.ZonalStatisticsAsTable(
        buffer, "OBJECTID", binary,
        f"stats_{vzd}.dbf", statistics_type="MEAN"
    )

    # Přečíst výsledek
    with arcpy.da.SearchCursor(stats, ["MEAN"]) as cursor:
        mean_value = next(cursor)[0]

    # Uložit
    vysledky.append({
        'Vzdalenost': vzd,
        'Procento': mean_value * 100
    })

# Vytvoř jednu tabulku se všemi výsledky
import pandas as pd
df = pd.DataFrame(vysledky)
df.to_csv('vysledky_vsechny.csv')

print("Hotovo! Všechny 4 vzdálenosti zpracovány.")
print(df)
```

Výstup:

```
   Vzdalenost  Procento
0         100      42.3
1         300      38.1
2         500      35.4
3        1000      31.2

Hotovo! Všechny 4 vzdálenosti zpracovány.
```

Ponaučení: Vidíte, proč se učíme Python?

7 Další zdroje

7.1 Dokumentace

ArcGIS Pro

- Model Builder dokumentace
- Geoprocessing nástroje
- Iterators in ModelBuilder

ArcPy (Python)

- ArcPy dokumentace
- ArcPy Get Started

7.2 Corine Land Cover

- Corine Land Cover - dokumentace
- CLC Nomenclature

7.3 Tipy na další studium

1. **Procvičujte:** Čím víc modelů vytvoříte, tím lépe pochopíte workflow
2. **Experimentujte:** Zkuste různé nástroje a parametry
3. **Dokumentujte:** Pište si poznámky k modelům (Description v properties)
4. **Připravte se na iterátory:** Přemýšlejte, co by se dalo automatizovat
5. **Sledujte Python kód:** I když mu ještě nerozumíte, zvykejte si na syntax

8 Kontakt a dotazy

Kontaktní informace

Vyučující: Vojtěch Barták
Email: [email]
Konzultační hodiny: [čas a místo]
Otázky k lekci:

- Pište na email s předmětem "GIS-L1: [vaše otázka]"
- Nebo přijďte na konzultace

Sdílení úkolů (volitelné):

- Pokud chcete zpětnou vazbu, odevzdejte přes [systém/email]
- Deadline: [datum] (ale není povinné!)

9 Poznámky pro další lekci

Co si přinést příště

- Funkční ArcGIS Pro
- Uložený toolbox s modelem z dnešní lekce
- Případně vyřešené volitelné úkoly (pokud chcete ukázat)

Na co se těšit v Lekci 2

ITERÁTORY - automatické opakování: - Iterate Field Values - projde 4 vzdálenosti automaticky! - Problém: 4 tabulky - jak spojit?
MODEL TOOL - nástroj s GUI: - Z modelu uděláme nástroj - Parametry, které může nastavit uživatel - Sdílení s kolegy
VNOŘENÉ MODELY: - Model volá jiný model - Obchází limit "1 iterátor na model" - Uvidíte limity → motivace pro Python!

Gratuluji!

Úspěšně jste dokončili první lekci. Vytvořili jste funkční model, exportovali ho do Pythonu, a pochopili základy automatizace GIS úloh.
Viděli jste: - Jak Model Builder funguje - Jak vypadá Python kód - Kde jsou limity Model Builderu - **Proč se budeme učit Python!**

Next step: Lekce 2 - Iterátory, Model Tools a vnořené modely

Listing 1 `model_export.py`

```
# -*- coding: utf-8 -*-
# ---------------------------------------------------------------------------
# model_export.py
# Created on: 2025-01-15
# Description: Analýza lesů v pásmu kolem železnic
# ---------------------------------------------------------------------------

# Import knihovny ArcPy
import arcpy

# Lokální proměnné (cesty k datům)
okresy = "okresy"
dalnice = "dalnice"
clc_2018 = "clc_2018"
okres_jh_layer = "okres_jh_layer"
dalnice_clip = "C:\\Data\\dalnice_clip.shp"
buffer_500m = "C:\\Data\\buffer_500m.shp"
lesy_binarni = "C:\\Data\\lesy_binarni.tif"
vysledek_lesy = "C:\\Data\\vysledek_lesy.dbf"

# PROCES 1: Make Feature Layer - výběr okresu
arcpy.MakeFeatureLayer_management(
    in_features=okresy,
    out_layer=okres_jh_layer,
    where_clause="NAZ_LAU1 = 'Jindřichův Hradec'"
)

# PROCES 2: Clip - ořezání železnic
arcpy.Clip_analysis(
    in_features=dalnice,
    clip_features=okres_jh_layer,
    out_feature_class=dalnice_clip
)

# PROCES 3: Buffer - ochranné pásmo
arcpy.Buffer_analysis(
    in_features=dalnice_clip,
    out_feature_class=buffer_500m,
    buffer_distance_or_field="500 Meters",
    dissolve_option="ALL"
)

# PROCES 4: Equal To - binární rastr lesů
arcpy.gp.EqualTo_sa(
    in_raster_or_constant1=clc_2018,
    in_raster_or_constant2="3",
    out_raster=lesy_binarni
)

# PROCES 5: Zonal Statistics as Table
```