Baza iNaturalist — występowanie gatunków zagrożonych na danym obszarze

projekt z Pracowni informatycznej

Natalia Okopna
nr albumu: 123454

prowadzący: dr hab. Wojciech Jakubowski

Marzec 2022

Spis treści

Kod aplikacji

Na bazie projektu interface'u oraz specyfikacji zaczynamy tworzyć aplikację. Interface aplikacji tworzymy korzystając z biblioteki *Pyqt:* `https://pythonpyqt.com/`.

1 Struktura przechowywania danych

Pliki przechowujemy w następującej strukturze:

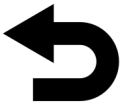
- inaturalist_base
  - scripts
    - data
      - about_app
      - back.jpg
      - data.json
      - data_bk.json
      - icon.png
    - points.py
    - saving.py
    - window.py
  - README

2 Ikony

Ikona jest taka sama dla każdego okna.

Rysunek 1: Ikona z loga nazwy bazy iNaturalist

Mamy również ikonę powrotu do poprzedniego okna. Powrót jest dostępny z każdego okna.



Rysunek 2: Ikona powrotu do poprzedniego okna

3 Odczyt i zapisywanie danych — saving.py

W pliku *saving.py* importujemy potrzebne biblioteki, następnie tworzymy funkcje obsługujące odczyt zapisywanie danych, tworzenie kopii itp.

4 Walidacja danych — validation.py

W pliku *validation.py* mamy funkcje do walidacji danych.

5 Interface — window.py

Kod potrzebny do zbudowania interface'u będziemy mieli w pliku *window.py*. Poniżej zobaczymy jego opis.

5.1 Potrzebne biblioteki

Wśród importowanych bibliotek mamy:

- sys — wbudowaną bibliotekę potrzebną do działania okna aplikacji,
- PyQt — biblioteka do tworzenia interface'u,
- folium — biblioteka do obsługi mapy.

Ponadto importujemy niektóre funkcje z plików *points.py* oraz *saving.py* potrzebne do obsługi danych.

5.2 Zmienne globalne

Następnie deklarujemy zmienne globalne: słownik z danymi obszaru i obserwacji *DATA* oraz słownik informacji o błędach *ERRORS*.

5.3 Funkcje

Kolejnie deklarujemy funkcję *error_show* potrzebną do wyświetlania informacji o błędach.

5.4 Klasa Menu

Następnie przystępujemy do deklaracji klas. Zaczniemy od deklaracji klasy głównego okna — *Menu*. Generator *__init__()* inicjuje klasę. Następnie w *initUI* ustawiamy ikonę aplikacji i inne parametry okna aplikacji, informacje o wersji aplikacji, umieszczamy wszystkie guziki i definiujemy funkcje otwierające odpowiednie okna po naciśnięciu każdego z nich.

5.5 Klasa NewArea

Deklarujemy klasę okna wprowadzania nowego obszaru *NewArea*. Ponadto ustawiamy Layout, aby wraz ze zmianą rozmiaru okna, elementy dopasowywały się do jego wymiarów.

Ponadto został stworzony obszar do wprowadzania danych geolokalizacyjnych (przykładowy input: 1,2;2,2;1,1), mapa, miejsce na listę gotowych punktów. Po zatwierdzeniu wczytania, następuje działający proces walidacji. Wyświetlane są okna informujące o błędach. Jeżeli obszar został poprawnie wprowadzony, na mapie pojawia się obszar wyznaczony przez użytkowania. Następnie użytkownik może zapisać obszar lokalnie, na dysku w pliku *scripts/data/data.json*. Nazwa obszaru zostaje również sprawdzana pod względem poprawności (akceptowalne są jedynie znaki alfanumeryczne).

5.6 Pozostałe klasy

Następnie deklarujemy wstępne wersje klas: okien obserwacji *Observations*, czytania z bazy *GenerateFromBase* i z dysku ReadFromDisk oraz okna wyświetlającego informacje o aplikacji *AboutApp*.

5.7 Wywołanie aplikacji

Po klauzuli *if __name__ == '__main__':* generujemy aplikację, ustawiamy opcje stylu czcionki, wyświetlamy aplikację.