CODE SNIPPET MANAGER

Aplikacja zaprojektowana i zaprogramowana przez Jana Solicha

Opis

SnippetManager („menedżer kawałków kodu”) to program napisany w C# i zaprojektowany do zbierania fragmentów kodu (ang. „snippet”) z różnych języków programowania do celów informacyjnych/nauki różnych funkcjonalności wybranego języka. Na przykład, dla C# byłoby kilka fragmentów pokazujących, jak używać delegatów lub pisać metody rozszerzeniowe, dla Javy przykłady użycia strumieni („streams”), dla C++ przykłady użycia klasy bitset, a dla Pythona przykłady użycia funkcji „list comprehension”.

Funkcjonalności

* Graficzny interfejs użytkownika
* Dodawanie, edytowanie, usuwanie fragmentów kodu
* Klonowanie fragmentów kodu w celu testowania ich różnych wariacji
* Podświetlanie składni kodu dla wszystkich wspieranych języków
* Możliwość zapisania danych kawałków kodu w pliku i odczytania ich z tego pliku w dwóch trybach: zastępowania (wszystkie snippety z listy zostaną zastąpione tymi z pliku) i dodawania (snippety z pliku zostaną dodane do istniejących, a istniejące zostaną).
* Wsparcie dla wielu języków programowania: Lua, Python, C#, C++, Java
* Kategoryzowanie fragmentów według tematu, języka programowania, poziomu zaawansowania
* Niektóre fragmenty mogą być uruchamiane bezpośrednio w aplikacji, aby natychmiast sprawdzić wynik i zachowanie, szczególnie po zmodyfikowaniu fragmentu (wspierane języki: Lua, Python, C#).
* W językach dynamicznych (Lua, Python) istnieje możliwość zachowania stanu interpretera między kolejnymi uruchamianymi snippetami. To znaczy np., że zmienne globalne zdefiniowane w snippecie będą dostępne i będą miały taką samą wartość w każdym innym snippecie w tym języku (oczywiście dopóki nie zostaną zmienione).
* Rozszerzony opis dla niektórych fragmentów (wraz z linkami do stron, gdzie omawiany jest temat), dostępny po kliknięciu przycisku
* Cheat-sheets, czyli skondensowany, ale długi kod z komentarzami pokazującymi dużo informacji na temat danego języka programowania, np. składnię, wbudowane funkcje, biblioteka standardowa itp.
* Sortowanie tabeli wyświetlającej informacje na temat snippetów przez kliknięcie na nagłówki kolumn

Różne uwagi

* Cały projekt będzie podzielony na bibliotekę implementującą większość najbardziej rozbudowanych funkcjonalności oraz aplikację GUI korzystającą z tej biblioteki.
* Interfejs GUI powinien mieć górne menu, jak w większości Windowsowych programów, i używać ikon, np. na przyciskach lub ikony pliku wykonywalnego programu.
* Program powinien używać jakiejś bazy danych do przechowywania fragmentów kodu (może to być po prostu JSON/XML).
* Należy zrobić przeglądarkę/edytor fragmentów kodu wspierającą języki programowania, w których może być potrzebnych wiele plików dla pojedynczego fragmentu (C++).

Implementacja

Biblioteka główna dostarcza klasy „CodeSnippet” i „SnippetList” reprezentujące główne elementy aplikacji. Ta pierwsza zawiera methody np. do uruchamiania kawałków kodu w wspieranych językach. Druga zawiera methody np. do filtrowania listy kawałków kodu według różnych kryteriów i zapisywania/odczytywania z pliku. Serializacja jest zrealizowana za pomocą wbudowanej klasy JsonSerializer.

Aplikacja GUI (WinForms) korzystająca z biblioteki zawiera dwie klasy okien („formularzy”). Jedno główne umożliwia użycie większości funkcjonalności (filtrowanie snippetów, prosta edycja [tylko kod snippetu], uruchamianie, usuwanie, klonowanie, odczyt i zapis do pliku…). Drugie pomocnicze umożliwia dodawanie/dokładną edycję kawałków kodu, umożliwiając dostosowanie praktycznie każdej właściwości snippetu.

Zawarta jest też klasa rozszerzająca klasę Scintilla z pakietu NuGet Scintilla5.NET. Scintilla to mocno zaawansowany edytor kodu napisany w C i używający Windowsowych komunikatów do zarządzania; Scintilla5.NET to wrapper .NET do tego edytora, umożliwiający korzystanie ze znacznej większości funkcjonalności w łatwy sposób i uwalniający od niskopoziomowych detali jak np. sterowanie komunikatami Windows.

Moja klasa dziedzicząca po klasie z pakietu dostarcza kilka przydatnych funkcjonalności, np. dynamiczna zmiana języka (w tym wypadku trzeba zrobić kilka rzeczy, które są „opakowane” przez jedną funkcję).

Kod programu

Do przechowywania kodu używam systemu kontroli wersji „Git” na platformie GitHub. Link do repozytorium: <https://github.com/Eksekk/SnippetManager>

Działająca wersja programu do natychmiastowego uruchomienia jest załączona w pliku .zip. !!!!!!! TODO !!!!!!!