Programowanie lokalnych aplikacji .NET

2019/2020

Instrukcja projektowa cz.3

Wielozadaniowość w Windows

Prowadzący: Tomasz Goluch

Wersja: 8.0

I. Zadania projektowe – 03.

Cel: Utrwalenie wiedzy zdobytej podczas wykładu.

Projekt będzie pisany w dwóch częściach. Pierwsza z wykorzystaniem wątków a w szczególności kolejki wątków (ThreadPool). Proszę o ustawienie wersji framework'a .NET na wersję 3.5.

Druga część projektu będzie wykorzystywać zadania (klasa Task), a w szczególności pulę zadań (biblioteka TPL – Task Parallel Library). Proszę o ustawienie wersji framework'a .NET na wersję 4.0 bądź wyższą.

Zadaniem każdej w części będzie napisanie prostego programu przetwarzającego pliki graficzne/wideo lub inne umieszczone w wybranym folderze. Program będzie przetwarzał pliki po uruchomieniu oraz w momencie dodania nowego pliku do monitorowanego folderu.

Każda część projektu powinna składać się z głównego interfejsu napisanego w WPF (GUI). Oraz jednego bądź większej liczby pomocniczych procesów konsola/WPF (worker). Liczba worker'ów zależy o liczby wybranych transformacji wybranych przez studenta. Ponadto GUI powinno pozwalać na wybranie folderu roboczego zawierającego obrabiane pliki. Po wybraniu folderu przetwarzanie powinno uruchomić się automatycznie, bądź powinniśmy mieć taką możliwość. Wybrany folder będzie dodatkowo monitorowany pod kątem operacji dodania nowych plików. W tym celu należy wykorzystać klasę `FileSystemWatcher`. Po dodaniu nowego pliku/plików do folderu, np. kopiując go/je przy pomocy eksploratora plików GUI sprawdza czy jest już uruchomiony proces konwertera, jeśli nie to uruchamia go i przekazuje mu informację o plikach wymagających przetworzenia. Jeśli proces konwertera istniał w momencie dodania nowych plików GUI jedynie informuje go o nowych zadaniach. Operacje uruchamiane przez GUI nie powinny go blokować.

Wspomniane procesy uruchamiane przez GUI to programy konsolowe przetwarzające pliki graficzne w pewien określony sposób. Rodzaj przetwarzania/konwersji jest dowolny, najlepiej jednak gdyby był relatywnie pracochłonny tak aby można było zauważyć pracę programu. Może to być zmiana rozmiaru pliku graficznego, zmiana rodzaju kompresji, zastosowanie pewnego efektu. W tym celu można wykorzystać np. bibliotekę AForge.NET. Dostępną w postaci pakietów nuget:

```
Install-Package AForge.Imaging
Install-Package AForge.Video
```

Przetwarzanie plików powinno być wykonane przy pomocy puli wątków bądź zadań/wątków. Worker powinien wyświetlać swój stan a w szczególności nazwę przetwarzanego pliku. Przetworzone pliki nie powinny być nadpisywane a kopiowane do innego folderu wynikowego.

W wersji minimalnej (50% oceny laboratoryjnej) w skład obydwu części programu powinien wchodzić proces GUI i worker'a. Przetwarzane powinny być zdarzenia związane z dodaniem nowego pliku do monitorowanego folderu. W celu podniesienia oceny wymagane są dodatkowe funkcjonalności, takie jak:

- (20%) dodanie kolejnego/kolejnych workerów powodującego że będą przetwarzane w wielu krokach. To czy wykonywane czynności będą wymagały odpowiedniej

kolejności oraz nadzór nad ich wykonaniem w określony sposób zależy od wybranych transformacji i powinno być dookreślone przez autora.

- (30%) wyświetlanie w GUI nazw plików czekających w kolejce do przetworzenia oraz nazwy aktualnie przetwarzanego pliku.
- (30%) możliwość wstrzymania/aktywacji lub przerwania aktualnie wykonywanych operacji przez wątki/zadania.
- (20%) możliwość dynamicznej zmiany wielkości parametru obrabianych plików np. rozmiaru, rodzaju kompresji itp… Aktualnie przetwarzany pliki powinien zostać przetworzony wg starych ustawień a kolejne już wg nowych.
- (10%) obsługa wyjątków powstałych podczas przetwarzania plików w puli.
- (? %) funkcjonalności zaproponowane przez studenta, w szczególności wymagające nietrywialnych mechanizmów synchronizacji.

Liczba punktów będzie zależeć od jakości wykonanego zadania oraz dodatkowych funkcjonalności. Podane wartości są jedynie oszacowaniem. Przykładowe GUI programu: