Programowanie pod Windows

Zestaw 5, C# + System.Windows.Forms

5 maja 2003 roku

1. Rozwiązać zadania 1-4 i 8 z zestawu 1 oraz zadania 1 i 2 z zestawu 2, nie korzystając (chyba że jest to niezbędne) z funkcji Win32API, tylko z bibliotek .NET.

2. Napisać prosty menedżer plików na wzór programu **Total Commander** (dawniej **Windows Commander**). Program powinien pokazywać dwie listy plików w wybranych folderach (użyć kontroli **ListView**). Użytkownik powinien móc nawigować po obu listach i wykonywać proste operacje na plikach, takie jak kopiowanie, przenoszenie, zmiana nazwy, usuwanie, uruchamianie, podgląd, tworzenie katalogu.

3. Napisać program będący odpowiednikiem menedżera zadań. Program powinien pokazać status aktywnych w systemie procesów i pozwolić na zabicie wybranego procesu.

4. Przedstawiony w skrypcie program rysujący w oknie bieżący czas przerobić na wzór zegarka systemowego Windows, to znaczy tak, żeby bieżąca godzina była przedstawiana na tarczy zegara analogowego a nie cyfrowego. Wykorzystać oczywiście funkcje do rysowania z GDI+.

5. Napisać program będący przeglądarką do obrazków z możliwością konwersji między popularnymi formatami graficznymi (wykorzystać możliwości obiektu **Image**).

6. Programiści stykający się z biblioteką **System.Windows.Forms** pytają często o to, w jaki sposób uzyskać taki styl menu jak ten z Visual Studio czy SharpDevelopa (styl ten istotnie różni się od standardowego menu).

   W związku z tym przedstawiony w skrypcie kod do rysowania elementów menu uzupełnić o rysowanie pozycji rozwijalnych z głównego menu, wzorując się na kształcie menu z Visual Studio .NET czy Sharp Develop. Dodać możliwość umieszczania ikon obok pozycji menu.

7. Programiści stykający się z biblioteką **System.Windows.Forms** pytają również często o to, w jaki sposób zmienić styl komponentu **Pro-**

gressBar na gładki, w przeciwieństwie do "kafelkowego", który jest dostępny standardowo.

Napisać w związku z tym własny komponent **SmoothProgressBar**. Komponent taki powinien zachowywać się jak zwykły **ProgressBar** (pasek postępu). Komponent powinien mieć tylko 3 propercje: **Min**, **Max** i **Value**, pozwalające określić odpowiednio minimalną, maksymalną i bieżącą wartość paska postępu.

Mając te informacje, **SmoothProgressBar** powinien rysować gładki pasek postępu o odpowiedniej długości. Do tego celu użyć odpowiednich funkcji z GDI+.