Programowanie pod Windows
Zestaw 7

System.Windows.Forms (2)

2025-04-08

Liczba punktów do zdobycia: **8/58**
Zestaw ważny do: 2025-04-29

1. (**3p**) Przygotować aplikację, która wykorzystuje omówiony na wykładzie podsystem GDI+ do rysowania w oknie zegara analogowego prezentującego bieżący czas, zgodny z zegarem systemowym.

   Rysowany widok powinien poprawnie dostosowywać się do wielkości okna podczas zmiany jego rozmiarów przez użytkownika.

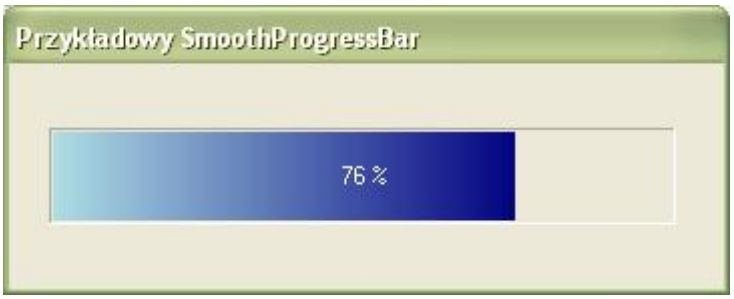
2. (**2p**) Zaimplementować własny komponent `SmoothProgressBar`, który będzie imitować zachowanie standardowego komponentu `ProgressBar` (pasek postępu).

   Komponent powinien mieć co najmniej 3 właściwości: `Min`, `Max` i `Value`, pozwalające określić odpowiednio minimalną, maksymalną i bieżącą wartość paska postępu. Mając te informacje, `SmoothProgressBar` w zdarzeniu `Paint` powinien rysować gładki (w przeciwieństwie do oryginalnego, który jest złożony z "kafelków") pasek postępu o odpowiedniej długości (według zadanych proporcji).

3. (**2p**) Zademonstrować w praktyce działanie klasy `BackgroundWorker` oraz jej zdarzenia `ProgressChanged` do delegowania długiego zadania do przetwarzania w tle.

   Formalnie - wątek w tle niech testuje jakiś zakres liczb na przykład testem pierwszości. Zakres dobrać tak, aby obliczenia trwały nie krócej niż kilkanaście sekund. Postęp obliczeń powinien być raportowany w interfejsie użytkownika za pomocą formantu paska postępu.

   Dla porównania - przygotować wersję która zamiast `BackgroundWorker` użyje po prostu wątka (obiekt typu `Thread`) który w trakcie obliczeń spróbuje aktualizować pasek postępu. Jaka trudność pojawia się w tym drugim podejściu (jest to również odpowiedź na pytanie co wnosi `BackgroundWorker` w stosunku do takiego naiwnego podejścia)?



Rysunek 1: Przykładowy `SmoothProgressBar`

4. (**1p**) Zaprezentować na niewielkim przykładzie zastosowanie rozszerzeń języka w obszarze programowania asynchronicznego (`async/await`).

   Bardziej formalnie - pokazać że w aplikacji okienkowej użycie nieblokującej metody asynchronicznej `HttpClient::ReadStringAsync` do pobrania zawartości z zewnętrznego zasobu sieciowego nie spowoduje zablokowania wątka głównego aplikacji, w którym przetwarzana jest pętla obsługi komunikatów. W tej samej aplikacji zademonstrować synchroniczne, blokujące wywołanie metody `WebClient::DownloadString`.

Wiktor Zychla