Platforma .NET

Zajęcia laboratoryjne

Ćwiczenie nr 2

1. Język C# (a) Napisać klasę Zegar, która zgłasza zdarzenie co 1 sekundę.
   Wskazówka: W klasie zgłaszającej zdarzenie zdefiniować delegat delZgloszenieZegara i utworzyć obiekt delegata zdarzenie. Funkcja delegata nie zwraca niczego, przyjmuje jako parametry referencję na klasę zgłaszającą(Zegar) i referencję na obiekt klasy TimeEventArgs.
   W klasie Zegar zaimplementować funkcję Run, która po uruchomieniu wykonuje się w pętli nieskończonej, odczytuje czas systemowy (DateTime.Now) i co sekundę zgłasza zdarzenie, tzn. uruchamia obiekt delegata (zdarzenie). Należy sprawdzić, czy obiekt delegata nie jest null!

   (b) Napisać klasę TimeEventArgs dziedziczącą po klasie EventArgs, przechowującą informację o czasie (godzinę, minutę i sekundę).

   (c) Napisać klasę Sluchacz, która nasłuchuje na zdarzenie generowane przez Zegar. Po wystąpieniu zdarzenia słuchacz wypisuje swoją nazwę i czas na konsoli.
   Wskazówka: Napisać funkcję subskrybuj, która przyjmuje jako parametr referencję na obiekt Zegara; funkcja przypisuje delegatowi klasy zgłaszającej zdarzenie metodę obsługi zdarzenia.

   (d) W klasie Sluchacz napisać funkcję obsługi zdarzenia, która przyjmuje jako parametry referencję na obiekt zgłaszającą zdarzenie (Zegar) i referencję na obiekt klasy TimeEventArgs.

   (e) W funkcji głównej uruchomić zegar i 3 słuchacze

2. Stwórz dwie metody (zwracające string)
   (a) Metodę wykonującą się 2 sekundy zwracającą na ekran informację na temat swojego działania oraz nr wątku na którym operuje (wykorzystaj Thread.CurrentThread.ManagedThreadId, Thread.Sleep) -> nazwa metody WolnaMetoda
   (b) Metodę wykonującą się szybko zwracającą na ekran informację na temat swojego działania oraz nr wątku na którym operuje (wykorzystaj Thread.CurrentThread.ManagedThreadId)-> nazwa metody -> SzybkaMetoda
   (c) Wypisz na ekran aktualny wątek (Thread.CurrentThread.ManagedThreadId) następnie uruchom metodę wolną przed metodą szybką, obserwuj efekty
   (d) Wykorzystaj funkcjonalność Task Parallel Library (TPL) aby uruchamiać wolną metodę (wykorzystaj Task oraz Task.Factory.StartNew<string>, Operation)
   (e) Uruchom program, obserwuj efekty
   (f) Stwórz asynchroniczną wersję wolnej metody wykorzystując operatory async / await oraz zastąp Thread.Sleep metodą Task.Delay (wywoływaną z operatorem await, która będzie reprezentować długotrwałą operację np. serwisu WCF wywoływaną w sposób asynchroniczny)
   (g) Uruchom, porównaj różnice

(h) Obserwowane różnice należy zamieścić w komentarzu w kodzie