LABORATORIUM SYSTEMÓW MOBILNYCH

STWORZENIE MOBILNEJ APLIKACJI, WYŚWIETLAJĄCEJ AKTUALNĄ POZYCJĘ UŻYTKOWNIKA, LISTĘ WIDOCZNYCH SATELITÓW ORAZ ICH POZYCJĘ

I. Temat ćwiczenia

Stworzenie mobilnej aplikacji, wyświetlającej na ekranie Pocket PC aktualną pozycję użytkownika, listę widocznych satelitów oraz ich pozycję (w formie graficznej).
Aplikacja ma odczytywać dane bezpośrednio z portu **COM** (lub z pliku) i poddawać analizie wykorzystując stworzony na poprzednich laboratoriach moduł **GPS** do parsowania danych **NMEA-0183**.

II. Wymagania

Znajomość podstaw tworzenia aplikacji mobilnej na Pocket PC.
Skończony moduł GPS z poprzednich zajęć.

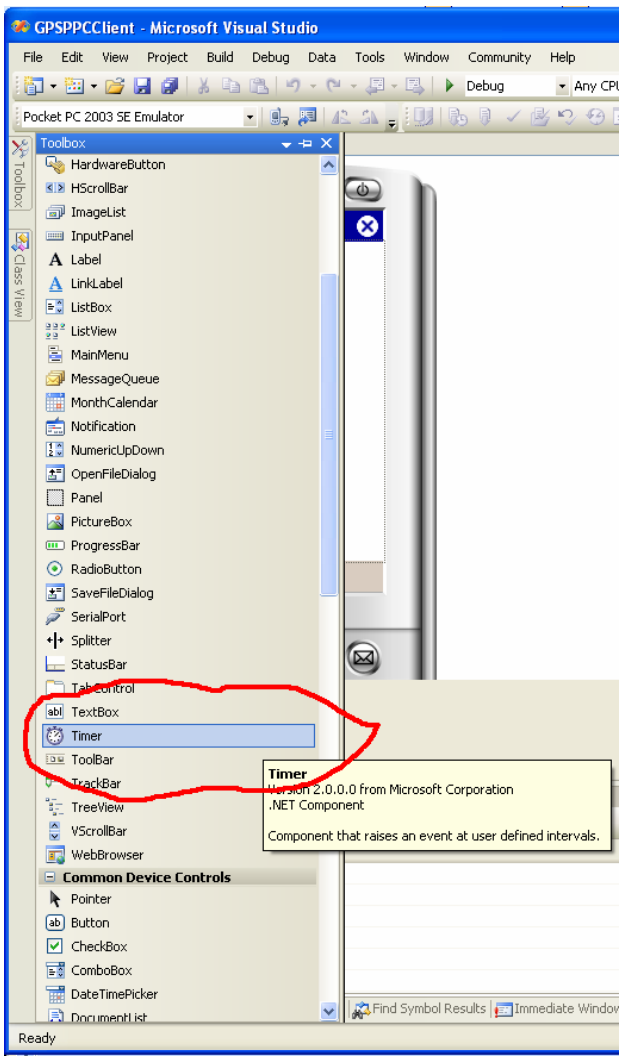
III. Ćwiczenie

1. Stworzenie nowego projektu.

Projekt powinien obejmować urządzenie typu **Pocket PC 2003**. Należy pamiętać o dodaniu możliwości ustalenia konfiguracji używanego portu szeregowego i odpowiednich przyciskach umożliwiających otwieranie i zamykanie portu.

Po otwarciu portu szeregowego, należy za pomocą timera lub w osobnym wątku odczytywać, co jakiś czas dane przychodzące do portu **COM** i poddawać je parsowaniu wykorzystując stworzony na poprzednich zajęciach moduł **GPS**.

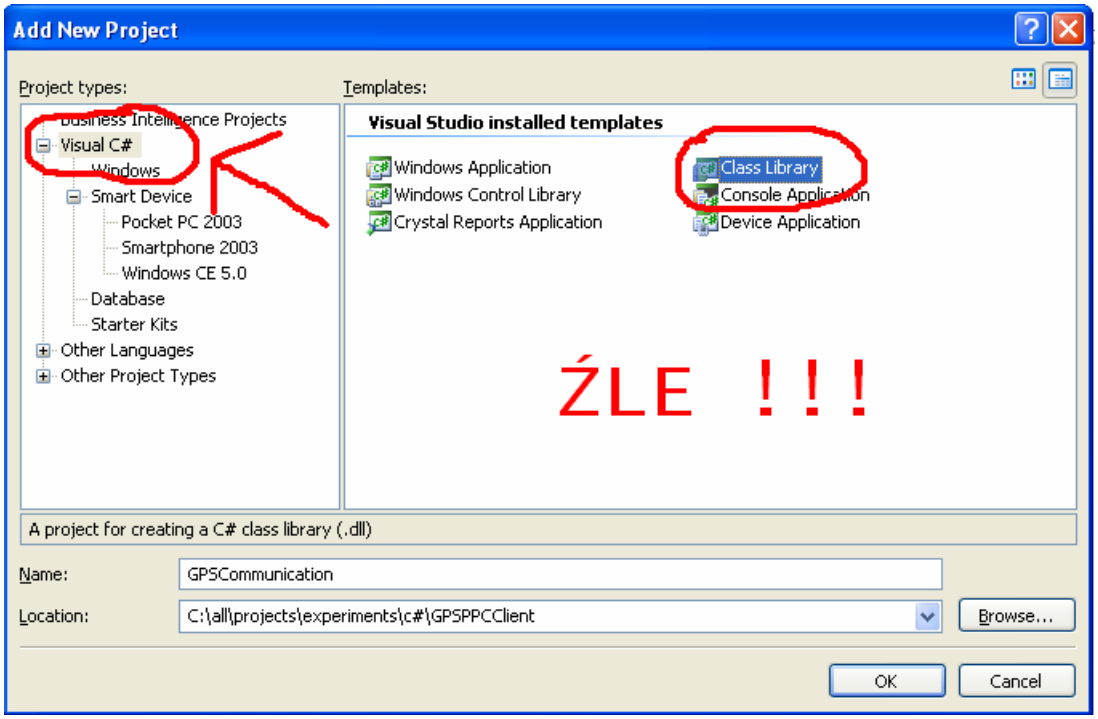
Należy wykorzystać opcje **OnTick**, timera systemowego, który należy dodać do aplikacji, znajdujący się na zakładce **Properties**:

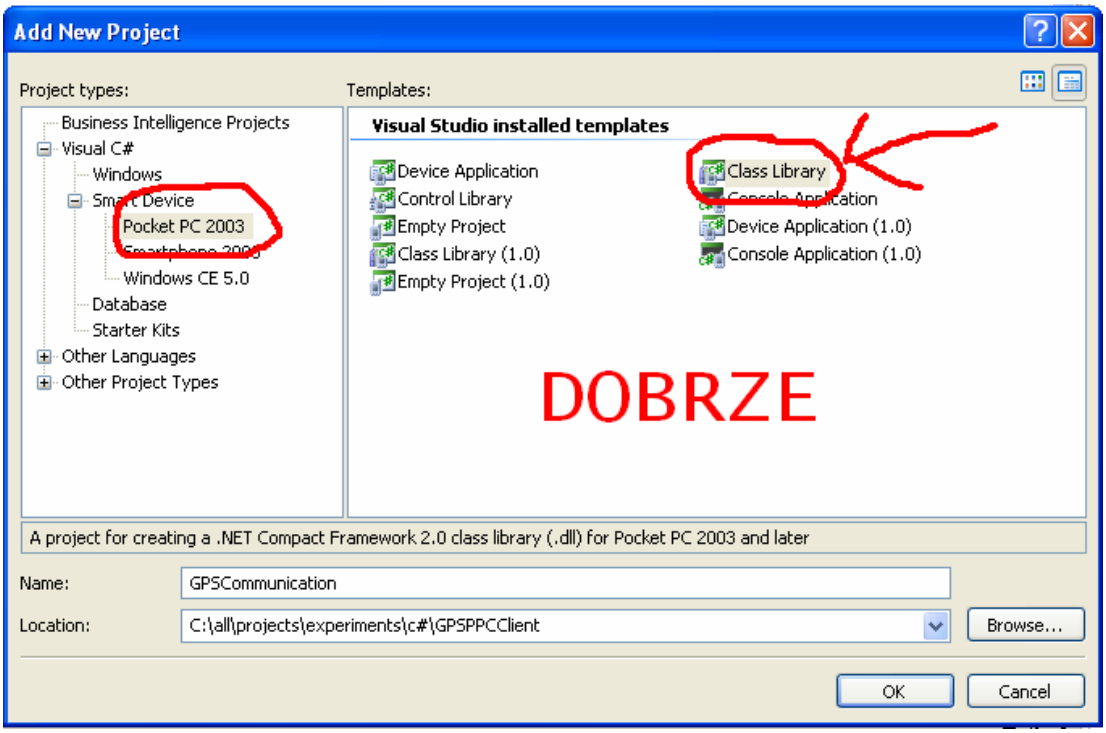


Przykładowa implementacja zdarzenia **OnTick** wygląda następująco:

```
private void timer1_Tick(object sender, EventArgs e)
{
      // odczytaj dane
      byte[] dane = GPSComm.Read();
      if (dane != null)
      {
            // poddaj je parsowaniu
            parser.Parse(dane);
            // uaktualnij formatke i wyswietl biezace dane
            UpdateView();
      }
}
```

Stworzony na poprzednich zajęciach moduł **GPS** (jeżeli został stworzony przy wykorzystaniu opcji **Class Library** bezpośrednio z zakładki **Visual C#**) będzie musiał zostać przerobiony na moduł **Class Library** (z zakładki **Pocket PC 2003**).





Sposób stworzenia odpowiedniego modułu do komunikacji z portem szeregowym i odczytywania danych z portu szeregowego został przedstawiony poniżej.

2. Stworzenie modułu komunikacyjnego z portem szeregowym COM

Należy stworzyć klasę odpowiedzialną za odczytywanie i monitorowanie portu **COM**.

Przy tworzeniu modułu, który powinien działać na **Pocket PC**, konieczne jest stworzenie go w odpowiedni sposób (należy wybrać opcje **Class Library** z zakładki **Pocket PC 2003**) – tak jak pokazano w poprzednim punkcie.

Jest to na tyle ważne, iż inaczej aplikacja **Pocket PC** nie skompiluje się !

W celu odczytywania danych z portu **COM** należy skorzystać z namespacu:

```
using System.IO.Ports;
```

Klasa **GPSCommunication** powinna posiadać prywatną zmienną typu **SerialPort**.

```
private SerialPort port = new SerialPort();
```

Klasa powinna zawierać 3 metody:

- Publiczną metodę otwierającą dany port COM (z określonymi jako argumenty konfiguracjami port – tzn. szybkość w bodach, ilość bitów stopu, kontrola parzystości itp.) – argumenty konfiguracyjne portu mogą być przekazywane do kreatora nowo tworzonego obiektu,
- Metodę odczytującą dane z portu COM,
- Metodę zamykającą port COM.

Przykładowe metody są następujące:

```
public bool Open(string portName,
      Parity parity, int baudRate, int dataBits, StopBits stopBits)
{
      port.PortName = portName;
      port.Parity = parity;
      port.BaudRate = baudRate;
      port.StopBits = stopBits;
      port.DataBits = dataBits;
      try
      {
            port.Open();
      }
      catch (Exception ex)
      {
            System.Diagnostics.Trace.WriteLine(
                  "Problem z otwarciem portu! " + ex.ToString());
            return false;
      }
      return true;
}
```

```
public void Close()
{
      try
      {
            // zamykam port
```

```
            if (port.IsOpen)
                  port.Close();
      }
      catch (Exception ex)
      {
            System.Diagnostics.Trace.WriteLine(
                  "Problem z zamknieciem portu! " + ex.ToString());
      }
}
```

```
public byte[] Read()
{
      byte[] bData = new byte[256];

      try
      {
            port.Read(bData, 0, 256);

            return bData;
      }
      catch (Exception e)
      {
            System.Diagnostics.Debug.WriteLine(e.ToString());
            return null;
      }
}
```

3. Wyświetlanie informacji o położeniu i informacji o satelitach

Informacje o bieżącym położeniu mogą być wyświetlanie na odpowiedniej zakładce w odpowiednich **Label**-ach lub **Textbox**-ach.

4. Wyświetlanie graficzne pozycji satelitów

W celu wyświetlenia pozycji satelitów w formie graficznej należy wykorzystać kontrolkę **PictureBox** z zakładki Properties.

Należy stworzyć metodę ShowSatelites wywoływaną z wnętrza metody UpdateView (która jest wywoływana z wnętrza Timer-a i odpowiada za odświeżenie informacji na formatce).

Wzorzec funkcji rysującej Satelity zamieszczono poniżej:

```
private void ShowSatelites()
{
      Pen circlePen = new Pen(System.Drawing.Color.DarkBlue,1);

      Graphics g = pictureBox1.CreateGraphics();

      int centerX = pictureBox1.Width/2;
      int centerY = pictureBox1.Height/2;

      double maxPromien =
            (Math.Min(pictureBox1.Height,pictureBox1.Width)-20) / 2;

      // rysuj okregi
      double[] okregi = new double[] {0,Math.PI/2, Math.PI/3 ,Math.PI / 6};
```

```
      foreach(double okrag in okregi)
      {
            double promien = (double) System.Math.Cos(okrag) * maxPromien;
            g.DrawEllipse(circlePen,(int)(centerX - promien),
                  (int)(centerY - promien),
                  (int)(2 * promien),
                  (int)( 2*  promien));
      }

      // zaznacz kreske co 90 stopni
      g.DrawLine(circlePen,centerX-3,centerY,centerX + 3,centerY);
      g.DrawLine(circlePen,centerX,centerY-3,centerX,centerY+3);

      Pen satellitePen =
            new Pen(System.Drawing.Color.LightGoldenrodYellow,4);

      // tutaj w petli powinny byc rysowane poszczegolne satelity
}
```

Następnie należy za pomocą odczytanych i sparsowanych danych z modułu GPS wyświetlić aktualny obraz widocznych satelitów.

Dla każdej satelity, można wywołać następującą funkcje rysującą:

```
// rysuj satelite
double h =
(double)System.Math.Cos((sat.Elevation * Math.PI) / 180) * maxPromien;

int satX = (int)(centerX + h * Math.Sin((sat.Azimuth * Math.PI) / 180));
int satY = (int)(centerY - h * Math.Cos((sat.Azimuth * Math.PI) / 180));

g.DrawRectangle(satellitePen, satX, satY, 4, 4);

g.DrawString(sat.Id.ToString(),
      new Font("Verdana", 8, FontStyle.Regular),
      new System.Drawing.SolidBrush(Color.Black),
      new Point(satX + 5, satY + 5));
```

Wykorzystanie komponentu PictureBox daje dodatkowo możliwość zapisania do pliku aktualnego ustawienia satelitów w celach zbierania historii.

5. Zadanie

Stworzoną aplikację wyświetlającą satelity systemu nawigacji należy rozszerzyć o funkcjonalność skalowania oraz obrotu otrzymanego obrazu rozmieszczenia poszczególnych satelitów. Ponadto przy każdej wyrysowanej satelicie powinna się pojawić informacja na temat jej współrzędnych położenia.