



Katedra Systemów Geoinformatycznych

Programowanie urządzeń mobilnych

Laboratorium

**Środowisko .NET Compact
Framework – omówienie
architektury, realizacja
podstawowej aplikacji GUI**



Wstęp

W ramach realizacji ćwiczenia student zapoznaje się z podstawowymi umiejętnościami umożliwiającymi stworzenie prostej aplikacji przeznaczonej na urządzenia mobilne z systemem operacyjnym Windows Mobile z wykorzystaniem środowiska .NET Compact Framework (.NET CF). Realizacja zadań w ćwiczeniu polega na zapoznaniu się :

- z architekturą .NET CF
- ze środowiskiem programistycznym Visual Studio 2008 (VS 2008)
- problematyką tworzenia projektów w VS 2008
- techniką uruchamiania aplikacji na emulatorze urządzenia mobilnego wraz z rodzajami emulatorów
- techniką uruchamiania aplikacji na wybranym urządzeniu mobilnym
- tworzeniem GUI użytkownika w aplikacjach .NET Compact Framework na przykładzie prostej aplikacji do zapisu danych osobowych studentów

Platforma .NET Compact Framework

Technologia .NET Framework jest produktem firmy Microsoft przeznaczonym do tworzenia i uruchamiania aplikacji na komputery klasy PC z wykorzystaniem języków programowania C++, C#, Visual Basic lub Java (Java#). .NET Framework umożliwia także korzystanie z wielu narzędzi wspomagających tworzenie kodu lub działanie aplikacji takich jak XML, technologia Web-service, SOAP, HTML i innych. Platforma .NET Framework składa się z dwóch głównych modułów:

- środowiska uruchomieniowego – Common Language Runtime (CLR)
- zintegrowanego zbioru bibliotek takich (ASP .NET, Windows Forms, ADO .NET i innych)

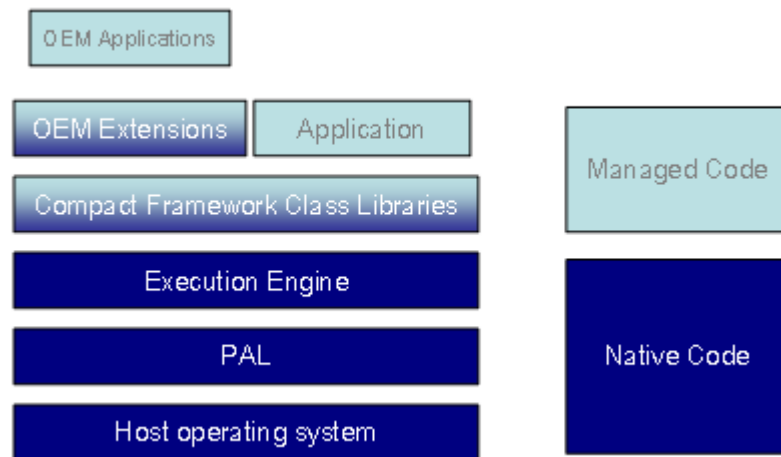
.NET Compact Framework (.NET CF) jest środowiskiem przeznaczonym do tworzenia aplikacji na urządzenia przenośne (ang. smart device) umożliwiającą wykorzystywanie tych samych technik programistycznych co w .NET Framework. Jej funkcjonalność jest zbliżona do funkcjonalności platformy .NET Framework i dostarcza programistom zbliżonych możliwości..NET CF jest platformą specjalnie przystosowaną do tworzenia aplikacji na urządzenia przenośne takie jak PDA, smartfony, która upraszcza proces tworzenia i uruchamiania aplikacji na urządzeniu przenośnym, umożliwiając tym samym, programiście w pełni wykorzystać funkcjonalność platformy sprzętowej.

Kod zarządzany (ang. managed code)

Kod źródłowy programu napisany z wykorzystaniem platformy .NET CF jest tzw. kodem zarządzalnym. Oznacza to, że środowisko uruchomieniowe platformy, NET CF (CLR) wprowadza szereg mechanizmów i zabezpieczeń umożliwiających sprawne, szybkie i bezpieczne wykonywanie kodu, a mianowicie:

- zarządzanie wskaźnikami,
- zarządzanie pamięcią,
- usuwanie wycieków pamięci.

Kod zarządzany podczas kompilacji programu jest transformowany do tzw. kodu Microsoft Intermediate Language (MSIL), czyli zbioru instrukcji wykonywalnych przez CLR oraz danych typu metadata stanowiących dodatkowy opis programu.



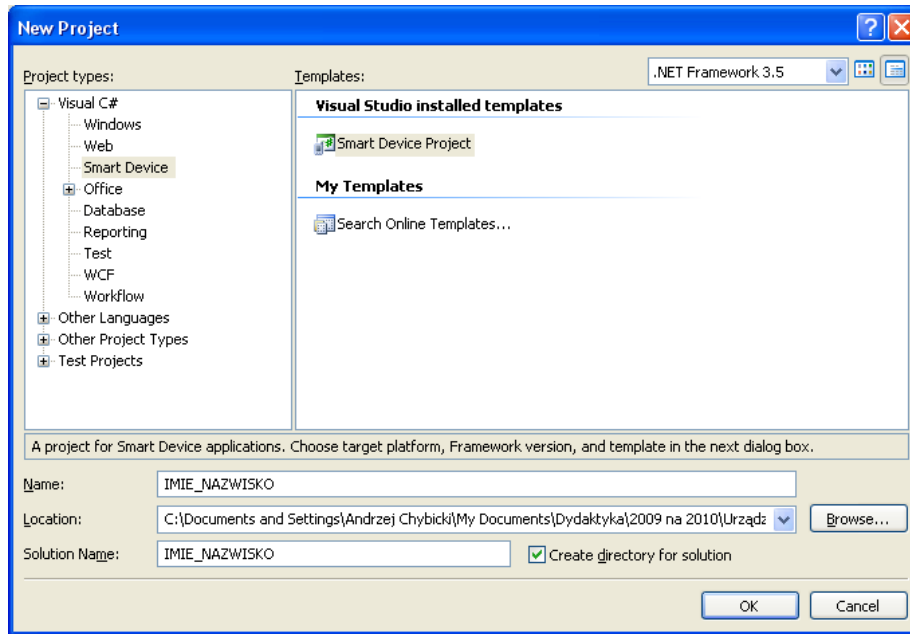
Rys. 1. Architektura platformy .NET Compact Framework.

Zadania do wykonania

ZADANIE 1

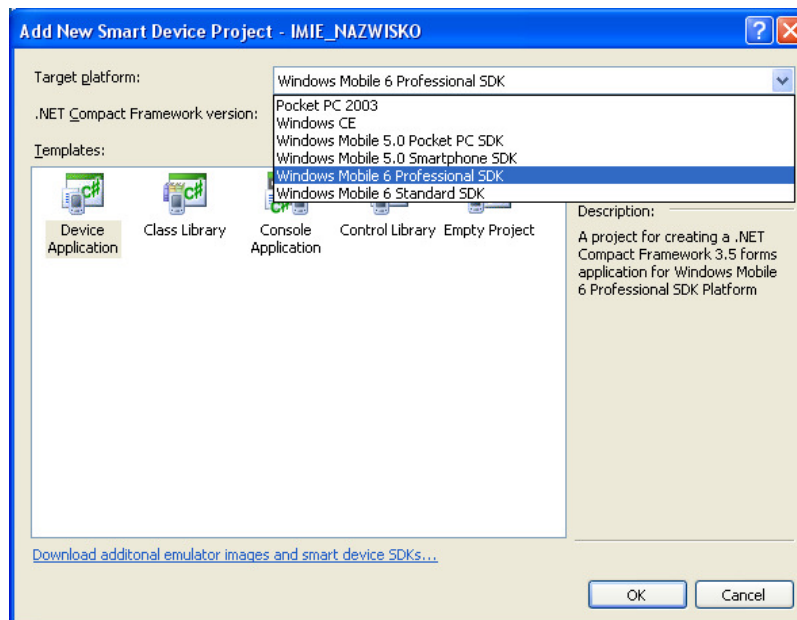
Zadanie 1 polega na zapoznaniu się ze środowiskiem Visual Studio 2008, stworzeniu projektu przeznaczonego do wykonywania na urządzeniach mobilnych oraz napisaniu prostej aplikacji GUI. Opis szczegółów zadań poniżej.

Tworzenie aplikacji GUI zaczyna się od stworzenia projektu w Visual studio. W tym celu w menu VS należy wybrać opcję **File->New->Project (Ctrl – Shift –N)** - pojawi się nam okno tworzenia nowego projektu (Rys. 2)



Rys. 2. Okno tworzenia nowego projektu.

Jako rodzaj projektu wybieramy *Smart Device Project*. W nazwie projektu w polu *Name* wpisujemy nazwę pochodzącą od naszego imienia i nazwiska przedzielonego podkreślnikiem napisaną dużymi literami jak na przykładzie (Rys. 2). Następnie pojawi się nam okno wyboru rodzaju projektu (Rys. 3).



Rys. 3. Okno wyboru rodzaju projektu.

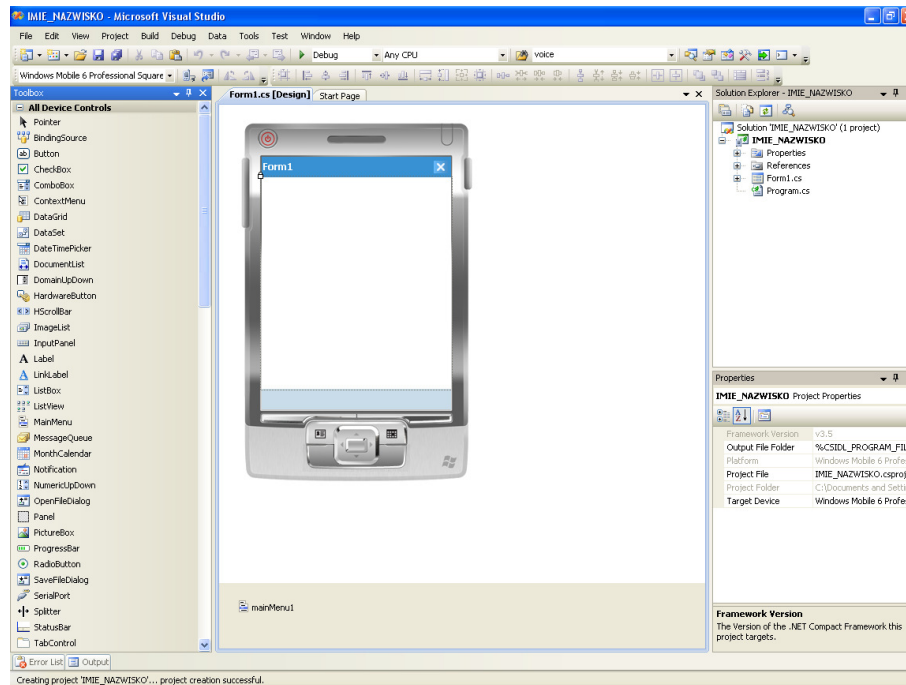
Jako platformę docelową (Target Platform) wybieramy Windows Mobile 6 Professional SDK. W polu *Templates* wybieramy Device Application. Dzięki zaznaczonym opcjom ustawienia naszego projektu będzie przystosowane tak, aby w wyniku kompilacji kodu stworzyć aplikację okienkową (Windows.Forms) na urządzenia mobilne wyposażone w system Windows Mobile w wersji 6 lub wyższej. Wybór innych opcji w tym okienku umożliwia nam stworzenie programu w innej wersji wykonywalnej np. dll lub CAB (plik instalatora na urządzenia mobilne)

Co powinieneś także wiedzieć

- Stosowanie Windows Mobile (WM) 6 SDK nie wyklucza faktu wdrażania (ang. deploy) aplikacji na urządzenia mobilne wyposażone w system WM 5. WM6 SDK zawiera szereg komponentów i bibliotek które nie muszą być obsługiwane w starszych wersjach systemu. Jeśli projektant aplikacji nie zdecyduje się na ich użycie program powinien bez problemu uruchamiać na WM 5 lub starszych
- Zmiana środowiska docelowego aplikacji jest możliwa także na etapie edycji projektu – można to zrobić klikając lewym przyciskiem myszy na zakładkę projektu w oknie *Solution Explorer*
- Aby sprawdzać działanie aplikacji nie trzeba jej od razu uruchamiać na urządzeniu mobilnym podłączonym do komputera – robi się to najczęściej z wykorzystaniem tzw. emulatora. W VS 2008 dostępny jest szereg emulatorów dla każdej z wersji systemów obsługiwanych przez .NET CF

Po naciśnięciu **OK** pojawi się okno Visual Studio (Rys. 4) z uruchomionym projektem, w którym dalej można programować aplikację. Wygląd i układ okienek może różnić w przypadku zastosowania innego emulatora oraz innych ustawień domyślnych. W tym przypadku, w głównym panelu widzimy tzw. okno designera umożliwiające projektowanie samego GUI aplikacji. Designer jest komponentem Visual Studio umożliwiającym dodawanie, usuwanie i zarządzanie komponentami GUI bez bezpośredniego tworzenia kodu – kod źródłowy jest generowany automatycznie do plików **Form1.cs** oraz **Form1.Designer.cs**, gdzie **Form1** jest klasą dziedziczącą po klasie **Windows.Form**. Komponenty GUI, które można dodać z poziomu

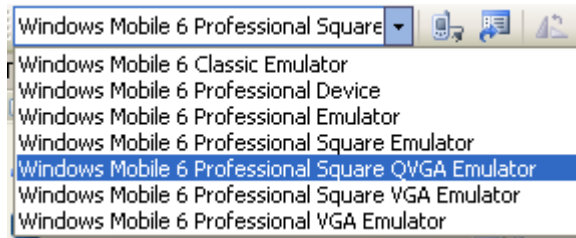
designera są przechowywane w magazynie **toolbox**, którego okno znajduje się po lewej stronie designera. Po prawej stronie ekranu widoczne jest także okno **Solution Explorera** umożliwiające zarządzanie całością projektu oraz, poniżej, okno właściwości (**Properties**) aktualnie zaznaczonego w designerze komponentu GUI. Programista oczywiście może dostosować sobie układ okien do swoich potrzeb wykorzystując np. zakładkę **Menu -> View**.



Rys. 4. Domyślny układ okien dla projektów Smart Device w Visual Studio 2008.

Aby uniknąć ciągłego wdrażania tworzonej aplikacji na urządzenie mobilne Microsoft dostarcza programiście narzędzi do testowania wytworzonego oprogramowania bez urządzenia mobilnego – narzędziem tym jest tzw. emulator.


Emulator jest osobnym programem, który zachowuje się dokładnie tak jak telefon komórkowy. W laboratorium dostępny jest szereg emulatorów urządzeń mobilnych spełniających standardy Windows Mobile. Wybór emulatora jest możliwy w ListBox w prawym górnym rogu ekranu (Rys. 5).



Rys. 5. Listbox wyboru emulatora.


W naszym projekcie wybieramy **Windows Mobile 6 Classic Emulator**. Z emulatora można korzystać tak samo jak z telefonu komórkowego – oznacza to, że można na nim instalować i uruchamiać zewnętrzne programy przeznaczone na urządzenia mobilne, korzystać z lokalnych zasobów (procesor, pamięć, karty pamięci) oraz wykonywać zapytania sieciowe – wszystko to oczywiście pod warunkiem wcześniejszej – odpowiedniej konfiguracji emulatora, o której będziemy mówić w ramach ćwiczeń laboratoryjnych.

PODSTAWY KONFIGURACJI EMULATORA

Emulator można włączać z poziomu VS bez konieczności debugowania lub uruchamiania tworzonej przez nas aplikacji. Robi się to za pomocą przycisku . Konfiguracja emulatora jest możliwa po wyborze opcji **File->Menu->Configure** z okna emulatora.

Sprawdź, jakie opcje można wybrać z menu konfiguracji emulatora?


CO JESZCZE POWINIENES WIEDZIEĆ

- emulator jest osobnym programem i działa tak jak telefon komórkowy,
- emulatora nie należy wyłączać za każdym razem kiedy chcesz wyłączyć aplikację – służy do tego przycisk Stop , który pojawi się po uruchomieniu aplikacji na emulatorze w głównym oknie Visual Studio,
- na emulatorze, tak samo jak na urządzeniu mobilnym, można instalować programy oraz kopiować pliki.
- po odpowiedniej konfiguracji emulatora, można także korzystać z dodatkowych urządzeń w które są standardowo wyposażone smartfony takich jak (wbudowana

kamera, aparat cyfrowy, GPS i inne) – sposób ich obsługi poznasz na następnych ćwiczeniach


URUCHOMIENIE APLIKACJI NA EMULATORZE

Aby sprawdzić czy w kodzie, który stworzyłeś nie ma błędu kompilacji, nie jest potrzebne uruchamianie aplikacji – samo przekompilowanie projektu jest możliwe po naciśnięciu opcji **Build Solution (F6)** w **Menu-Build**.

Aby uruchomić aplikację na wybranym przez nas wcześniej emulatorze używamy przycisku  znajdującego się w górnej części ekranu. Spowoduje on przekompilowanie kodu źródłowego, uruchomienie emulatora (lub urządzenia), na które ma być wdrażany projekt, oraz uruchomienie samego programu.

Domyślny projekt, który do tej pory został przez Ciebie stworzony po przekompilowaniu utworzy aplikację okienkową o nazwie **Form1**.

URUCHOMIENIE APLIKACJI NA URZĄDZENIU MOBILNYM

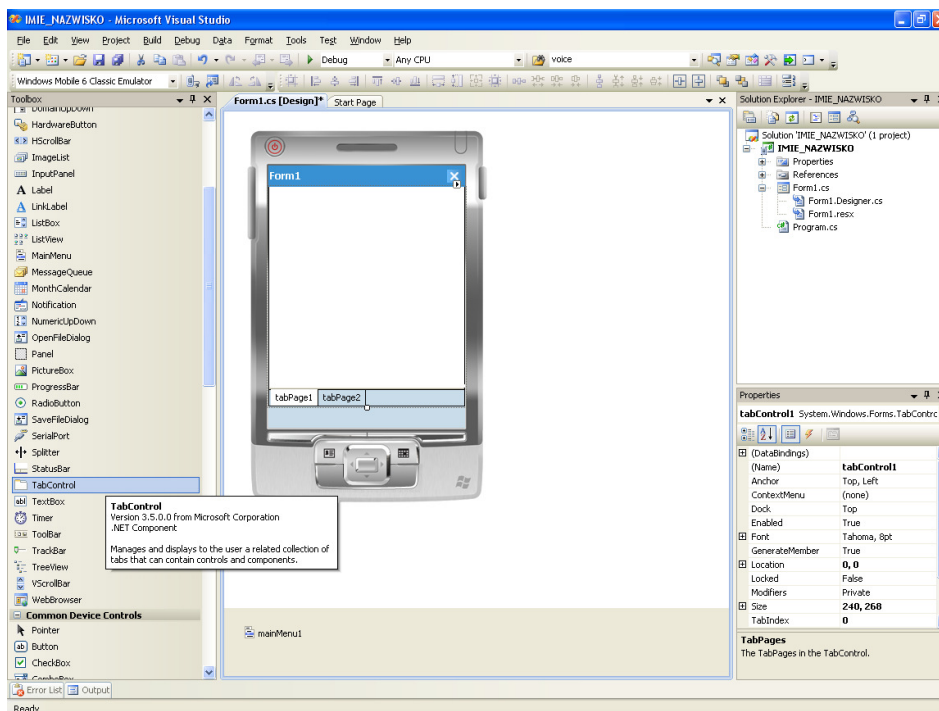
Aby uruchomić stworzoną przez Ciebie aplikację na urządzeniu mobilnym po naciśnięciu przycisku  należy wybrać urządzenie mobilne (Windows Mobile 6 Device). Należy wcześniej upewnić się, że urządzenie mobilne jest podłączone do komputera PC za pomocą kabla (USB – mini-USB). Aby prawidłowo wdrożyć aplikację na urządzenie mobilne wymagany jest także program ActiveSync, który można pobrać ze strony Microsoft (jest zainstalowany na komputerze laboratoryjnym). ActiveSync w tym przypadku jest odpowiedzialny za komunikację Visual Studio z komputerem oraz prawidłowe załadowanie zasobów na urządzeniu mobilnym.

Uruchomioną aplikację na urządzeniu mobilnym pokaż prowadzącemu – jest to koniec zadania 1

ZADANIE 2

Zadanie 2 polega na stworzeniu prostego interfejsu do programu, który był przedmiotem zad. 1. Zadanie to zrealizujemy tworząc prostą aplikację do zarządzania danymi osobowymi studentów, którzy robią zadanie laboratoryjne.

1. W pierwszym kroku dodaj komponent TabControl do głównego formularza aplikacji



Rys. 6. Dodawanie komponentu TabControl.

Pierwszą zakładkę nazwij „Dodaj dane” drugą zakładkę „Zarządzanie danymi”

2. W pierwszej zakładce stworzymy interfejs do dodawania danych. Dane osobowe, które nas będą interesować to:
 - ID (GUID),
 - imię (string),

- nazwisko (string),
- data urodzenia (DateTime),
- ocena– wybierane na podstawie enumeratora (Enum),
- zdjęcie (opcjonalnie- Image).

W nawiasie obok zmiennych podano typ, który będzie je reprezentował.

3. Dane osobowe będziemy przechowywać w klasie *Osoba* w pliku *Osoba.cs*. Aby dodać nową klasę naciskamy prawy przycisk myszy na naszym projekcie w oknie Solution Explorera i z menu kontekstowego wybieramy **Add->Class**.

Następnie wypełniamy ciało klasy odpowiednimi atrybutami klasy *Osoba*, pamiętając że typ Enum trzeba zdefiniować poza ciałem klasy. Definicja naszego enumeratora wygląda następująco:

```
public enum Ocena : byte {  
    niedostateczny,  
    dostateczny,  
    dobry,  
    bardzo_dobry  
}
```

Wszystkie atrybuty klasy z wyjątkiem atrybutu *zdjecie* powinny być publiczne. Z uwagi na to, iż atrybut *zdjecie* jest opcjonalny należy go traktować jako prywatny i ustawić dla niego właściwość (Property).

```
public Image Zdjecie  
{  
    get { return zdjecie; }  
}
```

```
set { zdjecie = value; }  
}
```

4. Stworzenie interfejsu użytkownika

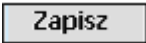
Po stworzeniu klasy *Osoba* przystępujemy do stworzenia samego interfejsu umożliwiającego wypełnienie danych. Korzystając z komponentów *label* (opisy pól), *textBox* (wpisy pól), *DateTimePicker* (data laboratorium), *PictureBox* (zdjęcie) oraz *Button* stwórz interfejs taki jak pokazany na Rys. 7.



Rys. 7. Interfejs programu



ZADANIE 3

Zadanie 3 polega na stworzeniu mechanizmu zapisywania i odczytywania informacji otrzymywanych od użytkownika poprzez interfejs. Dane powinny być zapisywane do pliku XML

o nazwie pochodzącej od numeru indeksu studenta. Zapis powinien się rozpoczynać po naciśnięciu przycisku  w GUI użytkownika.

Przeglądanie danych powinno być możliwe po przejściu na zakładkę „Zarządzaj danymi” . Należy wykorzystać komponent DocumentList. Po naciśnięciu na wybrany plik XML interfejs powinien uaktywnić pierwszą zakładkę wraz z wypełnionymi danymi pochodzącymi z wybranego pliku XML. Komponent DocumentList powinien pokazywać tylko pliki XML!!

WSKAZÓWKI:

- aby stworzyć funkcję która będzie uruchamiana po naciśnięciu przycisku  należy na niego dwa razy kliknąć w oknie designera. VS automatycznie stworzy odpowiednie delegaty i funkcje przypisując je do odpowiednich zasobów,
- aby przechwycić zdarzenie wyboru dokumentu w DocumentList należy przechwycić zdarzenie **DocumentActivated** – można to wykonać poprzez wykorzystanie zakładki Properties obiektu klasy DocumentList (podzakładka Events ) ,
- do zapisu danych do pliku XML należy wykorzystać namespace System.Xml. Krótki przykład kodu do zapisu i odczytu XML znajduje się poniżej:

```
XmlTextWriter xmlWriter = new XmlTextWriter(@"\Storage
Card\nazwa_pliku.xml", null);
    // Otwarcie dokumentu
    xmlWriter.WriteStartDocument();

    //komentarz
    xmlWriter.WriteComment("Opcjonalny komentarz");

    // Pierwszy element
    xmlWriter.WriteStartElement("Student");

    // Element zagnieżdżony
    xmlWriter.WriteStartElement("Imie", "");
    xmlWriter.WriteString("Andrzej");
    xmlWriter.WriteEndElement();
```

```
// Drugi element zagniezdzony
xmlWriter.WriteStartElement("Nazwisko", "");
xmlWriter.WriteString("Chybicki");
xmlWriter.WriteEndElement();

// Koniec elementu Student
xmlWriter.WriteEndDocument();

// Zamkniecie strumienia
xmlWriter.Close();
```

oraz czytanie takiego samego pliku XML:

```
XmlTextReader textReader = new XmlTextReader(@"\Storage
Card\nazwa_pliku.xml");

// Pierwszy element
textReader.ReadStartElement("Student");

// Element zagniezdzony
textReader.ReadStartElement("Imie", "");
imieTextBox.Text = textReader.ReadString(); //Andrzej
textReader.ReadEndElement();

// Drugi element zagniezdzony
textReader.ReadStartElement("Nazwisko", "");
nazwiskoTextBox.Text = textReader.ReadString(); //Chybicki
textReader.ReadEndElement();
```