

# Projekt TKOM - Dokumentacja wspólna

Konrad Miziński

1 grudnia 2012

## 1 Treść projektu

Biblioteka java do obsługi plików XML.

## 2 Opis funkcjonalności

Dostarczona biblioteka powinna zawierać klasy umożliwiające dostęp do całej treści pliku xml, tzn. do wszystkich elementów pliku wraz z ich nazwami, wartościami i atrybutami (reprezentowanymi przez ich nazwę i wartość).

## 3 Wymagania funkcjonalne

- Pliki xml są reprezentowane przez klasy implementujące interfejs XMLFile, ale elementy drzewa xml przez klasy implementujące interfejs XMLElement. Kody źródłowe tych interfejsów zastały zamieszczone w dalszej części dokumentacji.
- Dostęp do poszczególnych elementów pliku xml odbywa się na zasadzie drzewa, tzn dostęp do dowolnego elementu różnego od elementu nadrzędnego, możliwy jest jedynie poprzez jego rodzica w drzewie xml.
- Na każdym poziomie drzewa xml istnieje możliwość pobrania danego elementu w formie tekstowej (metoda getContent())
- Dostęp do klasy reprezentującej plik xml można uzyskać za pomocą obiektu typu "Factory", udostępniającego statyczne metody tworzące obiekty typu XMLFile.
- Obiekty typu XMLFile mogą być tworzone na podstawie ścieżki do pliku.xml lub obiektu typu java.io.File ze standardowej biblioteki Javy.
- W przypadku niepowodzenia utworzenia obiektu typu, rzucony jest zdefiniowany uprzednio wyjątek typu XMLParseException. Klasy XMLFile i XMLElement nie rzucają już żadnych wyjątków.
- Do biblioteki dołączona jest dokumentacja wygenerowana przez javadoc.

## 4 Wymagania niefunkcjonalne

- Biblioteka dostarczona zostanie w postaci archiwum typu JAR.
- Wraz z biblioteką dostarczony zostanie prosty program demonstrujący możliwości biblioteki. Program będzie pracował w trybie tekstowym.

## 5 Algorytm parsowania plików xml

Pliki xml parsowane będą za pomocą funkcji o następujących prototypach:

### Funcja parsujElement:

1. Znajdź początek najbliższego taga("<").
2. Znajdź koniec najbliższego taga(">").
3. Wywołaj funkcję parsujNazweIAtrybuty.
4. Jeśli pusty element ("<.../>") to zakończ działanie funkcji.
5. W przeciwnym wypadku znajdź tag kończący element (</...>)
6. Jeśli pusty element (<...></...>) to zakończ działanie funkcji.
7. W przeciwnym wypadku jeśli element zawiera nieotagowany tekst to zapamiętaj wartość elementu i zakończ działanie funkcji.
8. W przeciwnym wypadku wykonuj funkcję parsujElement aż do osiągnięcia końcowego taga rozpatrywanego elementu.

### Funcja parsujNazweIAtrybuty:

1. Znajdź pierwszy biały znak.
2. Tekst przed pierwszym białym znakiem potraktuj jako nazwę elementu.
3. Pomiń kolejne białe znaki
4. Jeśli napotkano koniec taga(">" lub ">") to zakończ działania funkcji.
5. W przeciwnym wypadku znajdź element "="
6. Tekst od ostatnio pominiętego białego znaku do znaku równości potraktuj jako nazwę atrybutu.
7. Pobierz znak następny po znaku "=" (" lub ' - początek wartości atrybutu).
8. Znajdź następny znak taki jak znak początku wartości atrybutu (odpowiednio " lub ')
9. Test pomiędzy znakami " (lub ') potraktuj jako wartość atrybutu.
10. Powrót do kroku 3.

## 6 Postawowe intefejsy

```
package pl.waw.mizinski.xml;

public interface XMLFile
{
    XMLElement getRootElement ();

    String getContent ();
}

package pl.waw.mizinski.xml;

import java.util.List;
import java.util.Map;

public interface XMLElement
{
    String getName ();

    boolean isComplexElement ();

    boolean isEmpty ();

    String getAttribute (String name);

    Map<String , String> getAttributes ();

    List<String> getChildElementsNames ();

    List<XMLElement> getChildElements ();

    int getChildElementsCount (String name);

    XMLElement getChildElement (String name);

    XMLElement getChildElement (String name, int number);

    String getValue ();

    String getChildElementValue ( String attribute );

    String getContent ();
}
```