Projekt TKOM - Dokumentacja wspępna

Konrad Miziński

15 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 elmentów pliku wraz z ich nazwami, wartościami i atrybutami (reprezentowanymi przez ich nazwę i wartość).

3 Wymagania funkconalne

- Pliki xml są reprezentowane przez klasy impelementujące interfejs XMLFile, ale alementy dzrzewa xml przez klasy implementujące interfejs XMLElement. Kody źródłowe tych interfejsów zastały zamieszczone w dalszej częci dokumentacji.
- Dostęp do poszczególnych elementów pliku xml odbywa się na zasadzię drzewa, tzn dostęp do dowoleno elementu różnego od elementu nadzrzędnego, możliwy jest jedynie poprzez jego rodzica w drzewie xml.
- Na każdym poziomie drzewa xml istnieje możliwośc pobrania danego elementu w formie tekstowej (metoda getContent())
- Dostęp do klasy reprezentującej plik xml można uzyskać za pomocą obiektu typu "Factory", udostępniającego statyczne motody tworzące obiekty typu XMLFile.
- Obiekty typy 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, rzucany jest zdefininiowany uprzednio wyjatek typu XMLParseException. Klasy XMLFile i XMLElement nie rzucają już żadnych wyjątków.
- Do biblioteki dołączona jest dokumentacja wygenerowana przez javadoc.

4 Wymiagania niefunkcjonalne

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

5 Algorytm parsowania plików xml

Pliki xml parsowane bedą za pomocą funkcji o następujących prototypach:

Funcja parsujElement:

1. Znajdź początek najbiższego taga("<").
2. Znajdć koniec najibliższego taga(">").
3. Wywołaj funkcję parsujNazweIAtrybuty.
4. Jeśli pusty element ("<.../>") to zakończ działanie funkcji.
5. W przeciwnym wypdaku znajdź tag kończący element (</...>)
6. Jeśli pusty element (<...></...>) to zakończ działanie funkcji.
7. W przeciwnym wypadku jeśli element zawiera nieotagowany tekst to zapamietaj wartosć elementu i zakończ działanie funkcji.
8. W przeciwnym wypadku wykonuj funkcję parsujElement aż do osiądniecia końcowego taga rozpatrywanego elemntu.

Funcja parsujNazweIAtrybuty:

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

        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();
}
```