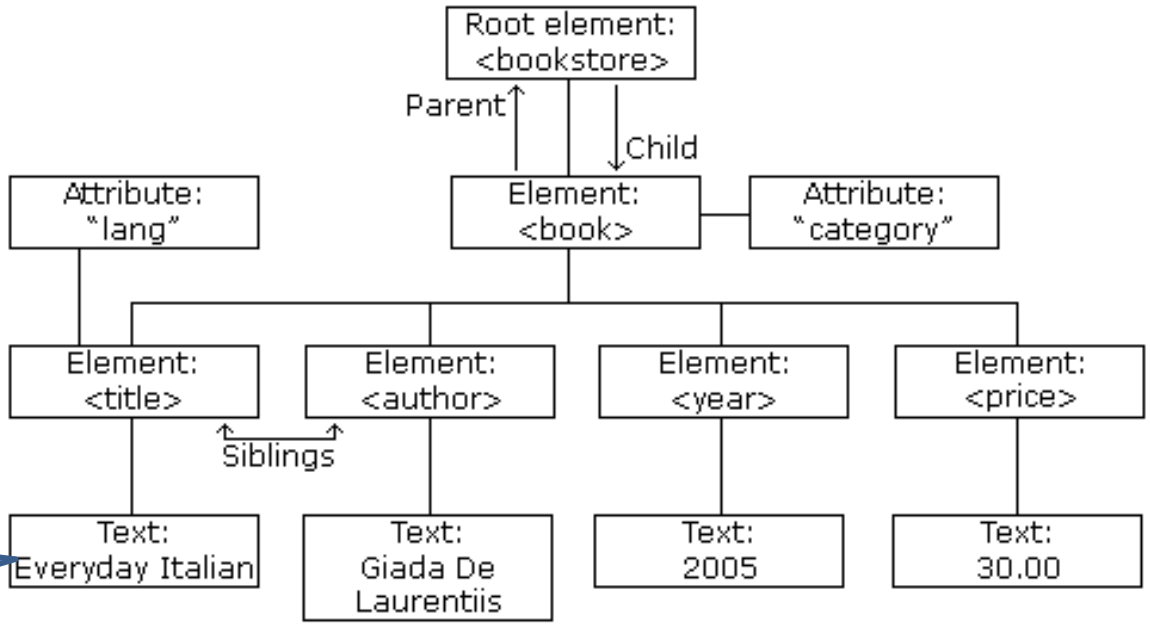
Obsługa XML
DOM - model

Definicja

- **Obiektowy model dokumentu** (*Document Object Model*, **DOM**) – sposób reprezentacji złożonych dokumentów XML i HTML w postaci modelu obiektowego. Model ten jest niezależny od platformy i języka programowania.
- Standard W3C DOM definiuje zespół klas i interfejsów, pozwalających na dostęp do struktury dokumentów oraz jej modyfikację poprzez tworzenie, usuwanie i modyfikację tzw. *węzłów* (ang. *nodes*).
- Dla większości języków programowania istnieją biblioteki obsługujące DOM dla plików XML. Standard W3C definiuje interfejsy DOM tylko dla języków JavaScript i Java.

Źródło: https://pl.wikipedia.org/wiki/Obiektowy_model_dokumentu

Struktura drzewiasta DOM XML



Tekst w węźle jest osobnym węzłem

Podstawowe właściwości / metody

Właściwości XML DOM

- Typowe właściwości DOM:
- x.getNodeName - nazwa x
- x.setNodeValue - wartość x
- x.getParentNode – rodzic x
- x.getChildNodes - dzieci x
- x.getAttributes - atrybuty x

Uwaga
Tekst w Elemencie jest osobnym węzłem, więc jeśli chcemy odczytać wartość tekstową to należy
x.setTextContent(napis)
x.getTextContent()

Metody XML DOM

- x.getElementsByTagName(*name*) – Zwraca wszystkie elementy jako węzły (Node) które mają określoną nazwę
- x.appendChild(*node*) – Wstawia węzeł dziecięcy
- x.removeChild(*node*) – Usuwa węzeł dziecięcy
- x.getNextSibling() – Zwraca następnego z rodzeństwa

Tworzenie dokumentu

1. Stworzenie dokumentu za pomocą klasy **DocumentBuilder**
2. Dodawanie zawartości XML za pomocą klasy **Element**
3. Konwersja XML do OutputStream za pomocą klasy **Transformer**
4. Uwaga: Większość klas budujących pobieramy z fabryk

Przykład

```
<biblioteka>
        <ksiazka id="1">
                <tytul>Przykladowy tytul</tytul>
                <autorzy>
                        <autor>
                                <imie>Adrian</imie>
                                <nazwisko>Drabina</nazwisko>
                        </autor>
                        <autor>
                                <imie>Jan</imie>
                                <nazwisko>Kowalski</nazwisko>
                        </autor>
                </autorzy>
                <rokwydania>2017</rokwydania>
        </ksiazka>
</biblioteka>
```

```java
public static void main(String[] args) {
    try {
        //Przygotowanie do stworzenia dokumewntu
        DocumentBuilderFactory docFactory = DocumentBuilderFactory.newInstance();
        DocumentBuilder docBuilder = docFactory.newDocumentBuilder();
        //Stworzenie dokumentu
        Document doc = docBuilder.newDocument();
        //Tworzenie zawartości dokumentu
        Element rootElement = doc.createElement("biblioteka");
        doc.appendChild(rootElement);
        //Dodanie elementow do biblioteki
        Element ksiazka = doc.createElement("ksiazka");
        rootElement.appendChild(ksiazka);
        //Dodanie atrybutu do elementu
        Attr attr = doc.createAttribute("id");
        attr.setValue("1");
        ksiazka.setAttributeNode(attr);
        //Dodanie pozostałych elementów
        Element autorzy = doc.createElement("autorzy");
        autorzy.appendChild(doc.createTextNode("Jan Kowalski"));
        ksiazka.appendChild(autorzy);
        Element rokWydania = doc.createElement("rokwydania");
        rokWydania.setTextContent("2017");
        ksiazka.appendChild(rokWydania);
        //Zapisanie dokumentu
        TransformerFactory transformerFactory = TransformerFactory.newInstance();
        Transformer transformer = transformerFactory.newTransformer();
        DOMSource source = new DOMSource(doc);
        //StreamResult result = new StreamResult(new File("example.xml"));
        StreamResult result = new StreamResult(System.out);
        transformer.transform(source, result);
    } catch (ParserConfigurationException | TransformerException ex) {
        ex.printStackTrace();
    }
}
```

Stworzenie Buildera

Stworzenie Dokumentu

Stworzenie Zawartości

Dodanie atrybutu do elementu zawartości

Stworzenie transformera i wysłanie do OutputStream

Wczytanie XML’a

```
public class ReadXML {
    public static void main(String[] args) {
        try {
            //Przygotowanie do stworzenia dokumewntu
            DocumentBuilderFactory docFactory = DocumentBuilderFactory.newInstance();
            DocumentBuilder docBuilder = docFactory.newDocumentBuilder();
            Document doc = docBuilder.parse(new File("biblioteka.xml"));

            TransformerFactory transformerFactory = TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();
            //Zapisanie dokumentu
            DOMSource source = new DOMSource(doc);
            //StreamResult result = new StreamResult(new File("example.xml"));
            StreamResult result = new StreamResult(System.out);
            transformer.transform(source, result);
        } catch (ParserConfigurationException | SAXException | IOException | TransformerException ex) {
            ex.printStackTrace();
        }

    }
}
```

Przetwarzanie XML'a

```
try {
    //Przygotowanie do stworzenia dokumewntu
    DocumentBuilderFactory docFactory = DocumentBuilderFactory.newInstance();
    DocumentBuilder docBuilder = docFactory.newDocumentBuilder();
    Document doc = docBuilder.parse(new File("biblioteka_collection.xml"));
    //Pobranie głównego elementu
    Element element = doc.getDocumentElement();
    //Wyszukanie wszystkich elementów typu ksiazka
    NodeList nodeList = element.getElementsByTagName("ksiazka");
    //Iterujemy po ksiazkach
    for(int i=0; i<nodeList.getLength(); i++){
        Node n = nodeList.item(i); //Pobranie k-tego elementu
        if (n.hasAttributes()){ //Sprawdzamy czy element ma atrybuty
            NamedNodeMap m = n.getAttributes(); //Pobranie listy atrybutów
            System.out.println("Attrubtes");
            for (int j=0; j<m.getLength(); j++){ //Iterujemy po atrybutach
                Node na = m.item(j);
                Attr attr = (Attr)na;
                System.out.println(attr.getName() + " " + attr.getValue());
            }
            System.out.println("End of Attrubtes");
        }
        System.out.println(n.getNodeName() + " " //Odczytujemy zawartość elementów
                + n.getNodeValue() + " "
                + n.getTextContent());
        //System.out.println(n);
        System.out.println("----------------------------------");
    }
} catch (ParserConfigurationException | SAXException | IOException  ex) {
    ex.printStackTrace();
}
```

Przydatne narzędzia

```
StringWriter writer = new StringWriter();
transformer.transform(new DOMSource(doc), new StreamResult(writer));
String output = writer.toString();
```

XML to String

```
Node nn = ns.item(i);
transformer.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "yes");
StringWriter writer = new StringWriter();
transformer.transform(new DOMSource(nn), new StreamResult(writer));
String output = writer.toString();
```

Node to Document oraz Node to String

```
String s = formatter.format(o);
Document doc1 = docBuilder.newDocument();
Document doc2 = docBuilder.parse(new ByteArrayInputStream(s.getBytes()));
Element e = doc2.getDocumentElement();
Node node = doc1.importNode(e, true);
rootElement.appendChild(node);
```

Importowanie węzła z jednego dokumentu do drugiego