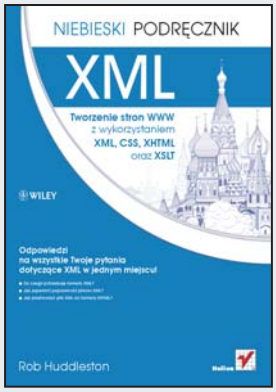
Helion SA
ul. Kościuszki 1c
44-100 Gliwice
tel. 032 230 98 63
e-mail: helion@helion.pl

XML. Tworzenie stron WWW z wykorzystaniem XML, CSS, XHTML oraz XSLT. Niebieski podręcznik



Autor: Rob Huddleston
Tłumaczenie: Andrzej Grażyński
ISBN: 978-83-246-1738-8
Tytuł oryginału: XML: Your visual blueprint for building expertwebsites with XML, CSS, XHTML, and XSLT (Visual Blueprint)
Format: 170×230, stron: 320

Odpowiedzi na wszystkie Twoje pytania dotyczące XML w jednym miejscu!

- Do czego potrzebuję formatu XML?
- Jak zapewnić poprawność plikom XML?
- Jak przetworzyć plik XML do formatu XHTML?

XML znaczy tyle, co „rozszerzalny język znaczników". Prawdopodobnie słowo „rozszerzalny" w tym skrócie jest najważniejsze. Oznacza ono, że jako użytkownik możesz zdefiniować tyle znaczników, ile tylko jest Ci potrzebnych w danym przypadku. XML jest uniwersalnym językiem formalnym, za pomocą którego można reprezentować dane w usystematyzowany, strukturalny sposób. Niezależność od platformy jest niewątpliwie tą cechą, która pozwoliła mu na zdobycie ogromnej popularności – stał się on wręcz swego rodzaju standardem. W dodatku język ten pozwala na tworzenie stron internetowych przy użyciu XHTML, czyli przedstawienie HTML w postaci XML. Mało? Dzięki zastosowaniu XSLT możesz także przekształcić dowolny dokument XML do postaci XHTML!

Dzięki książce „XML. Tworzenie stron WWW z wykorzystaniem XML, CSS, XHTML oraz XSLT. Niebieski podręcznik" poznasz podstawy XML i sposób przetwarzania plików XML za pomocą języka XSLT oraz dowiesz się, jak wybrać dobry edytor, pozwalający na wygodną pracę z dokumentami. W kolejnych rozdziałach zdobędziesz wiedzę na temat sposobów wykorzystania atrybutów i przestrzeni nazw oraz stworzysz swój pierwszy dokument XML. Ponadto nauczysz się definiować strukturę dokumentu za pomocą schematów i zobaczysz, jak łatwo można oceniać poprawność dokumentów XML. Autor książki zaprezentuje Ci również technikę generowania dokumentów XML na podstawie istniejących danych, między innymi w programach Access i Excel, oraz technikę przekształcania plików XML do formatu XHTML za pomocą języka XSLT. Jeżeli interesuje Cię format XML i formaty pokrewne, trzymasz w ręku książkę, której poszukiwania zajęły Ci tak dużo czasu!

Wykorzystaj wszystkie możliwości XML w swoich projektach!

Spis treści

Wprowadzenie do schematów

Jak wiadomo, twórca dokumentu XML ma bardzo dużą swobodę pod względem definiowania elementów. Aby jednak tworzony dokument był zrozumiały dla innych ludzi i dla parserów, konieczne staje się dostarczenie informacji na temat jego struktury, a konkretnie informacji o tym, jakie elementy będą w dokumencie używane, w jakiej kolejności powinny się pojawiać, które z nich muszą być obowiązkowe użyte i (ewentualnie) jakie atrybuty posiadać może (lub musi) każdy z tych elementów.

Definicja schematu W3C

XML — i pokrewne mu technologie — opracowany został przez World Wide Web Consortium, w skrócie W3C. Dostrzegając konieczność dostarczenia metody definiowania XML-a, konsorcjum to opracowało język o nazwie XML Schema Language, który stał się standardem definiowania poprawnych dokumentów XML. Język ten, sam oparty na XML-u, to język zarówno skomplikowany, jak i bardzo funkcjonalny.

Schemat dokumentu XML jest jego planem; samo słowo *schemat* wywodzi się z języka greckiego i oznacza właśnie kształt lub plan. Niekiedy schematy dokumentów XML określane są mianem ich „gramatyki".

Schemat schematów

No dobrze, ale sama koncepcja schematu XML wyrażona została przez W3C w postaci dokumentu XML, co przypomina błędne koło w rodzaju: „Co było pierwsze: kura czy jajko?". Jak bowiem zdefiniować schemat dokumentu, którego treścią jest właśnie definicja schematu?

Odpowiedź kryje się pod postacią głównego (*master*) dokumentu nazywanego „schematem schematów", opracowanego przez W3C i stanowiącego punkt wyjścia do definiowania jakichkolwiek schematów. Definiując więc schemat — w odróżnieniu od „zwykłego" dokumentu XML — możemy wykorzystywać jedynie znaczniki zdefiniowane w „schemacie schematów"

XML Data Reduced Schema firmy Microsoft

Konsorcjum W3C z zasady działa dość wolno, wobec konieczności zbierania materiałów i opinii od programistów z całego świata, dyskutowania pojawiających się problemów i wypracowywania rekomendacji (które znowu muszą być szeroko dyskutowane). Pod koniec lat 90. ubiegłego wieku Microsoft przygotowywał się do wypuszczenia serii produktów wykorzystujących XML i w związku z tym potrzebował narzędzia do definiowania schematów. Ponieważ standard W3C nie był jeszcze gotowy, Microsoft opracował własny standard w tym względzie, niestety — jak się ostatecznie okazało — całkowicie odmienny od W3C Schema. Microsoft (sam wchodzący w skład W3C) zgodził się więc całkowicie zarzucić własny standard po ukazaniu się W3C Schema, i tak się też faktycznie stało. Ponieważ jednak ów zarzucony standard był dostępny publicznie, można natknąć się na schematy definiowane na jego podstawie; w takim przypadku warto rozważyć przepisanie schematu do postaci zgodnej z W3C Schema, bowiem ewentualne przyszłe wsparcie dla standardu Microsoftu stoi pod znakiem zapytania.

Co jest definiowane w ramach schematu?

Mimo iż wielu początkujących twórców skłonnych jest traktować poszczególne dokumenty XML jako niezależne całości, to często zachodzi potrzeba stworzenia szeregu dokumentów różnych, co prawda, ale opartych na tej samej strukturze. Koronnym przykładem takiej sytuacji jest tworzenie witryny WWW, której poszczególne strony przechowują dane w postaci XML — wielce prawdopodobnym jest, że na każdej ze stron dane te podporządkowane są pewnemu wspólnemu schematowi.

Co jest definiowane w ramach schematu? *(kontynuacja)*

Dokładniej rzecz biorąc, w definicji schematu określone są następujące aspekty struktury dokumentu:

- nazwa elementu nadrzędnego (*root*) i sposób różnicowania wielkości liter w tej nazwie,
- nazwy i sposoby różnicowania wielkości liter dla elementów zagnieżdżonych bezpośrednio w elemencie *root*,
- nazwy i sposoby różnicowania wielkości liter dla elementów zagnieżdżonych na niższych poziomach,
- określenie, które elementy mogą zawierać dane i w których elementach można zagnieżdżać inne elementy,
- dla elementów, które mogą zawierać dane — dozwolony typ tych danych (na przykład tekst lub liczba),
- określenie, które elementy muszą wystąpić obowiązkowo w dokumencie, które są opcjonalne i które mogą występować wielokrotnie w dokumencie,
- określenie, które elemen ty mogą posiadać atrybuty, wraz z bliższą informacją o tychże atrybutach: rodzaju, obowiązkowości lub opcjonalności wystąpienia, rodzaju reprezentowanej informacji,
- kolejność, w jakiej poszczególne elementy powinny pojawiać się w dokumencie.

Tworzenie schematów

Ponieważ definicja schematu jest niczym innym jak dokumentem XML, można ją tworzyć za pomocą tych samych narzędzi, które służą do tworzenia „regularnych" dokumentów XML. Jeżeli jakaś aplikacja posiada funkcję eksportu danych w formacie XML — do wykorzystania w innych aplikacjach — prawdopodobnie udostępnia też funkcję eksportu schematu dla tychże danych. Wiele zaawansowanych edytorów XML, jak XMLSpy, oferuje możliwość automatycznego generowania definicji schematu na podstawie przykładowego pliku; istnieją ponadto dedykowane narzędzia do tworzenia schematów.

Co najpierw?

Logicznie rzecz biorąc, wydaje się oczywiste, że nie sposób stworzyć dokumentu XML bez podstawowej wiedzy o jego strukturze; wychodzi więc na to, że najpierw trzeba zdefiniować schemat dokumentu. W praktyce jednak kwestia ta rozstrzygana jest nieco mniej rygorystycznie: autorzy, mając jedynie zarys koncepcji struktury, tworzą na jego podstawie zaczątek przykładowego dokumentu, a dopiero potem przystępują do regularnego definiowania schematu, weryfikując w ten sposób uprzednio przyjęte założenia.

Publikowanie schematów

Niektóre schematy przeznaczone są jedynie dla wąskich, specyficznych potrzeb, niektóre jednak są na tyle uniwersalne, że prawdopodobne jest, iż staną się podstawą masowego tworzenia dokumentów XML, i z tego względu warto je uczynić publicznie dostępnymi. Wiele popularnych technologii, w tym RSS, miało właśnie takie początki.

Wykorzystywanie istniejących schematów

Wiele organizacji i korporacji opublikowało mnóstwo standaryzowanych schematów XML dla swych gałęzi przemysłu, warto więc poszukać gotowych rozwiązań przed podjęciem decyzji o samodzielnym definiowaniu takiego czy innego schematu. Można w tym celu użyć popularnych wyszukiwarek bądź skontaktować się bezpośrednio z przedstawicielem odpowiedniej firmy.

Przestrzenie nazw schematów

Schemat jest dokumentem XML, którego strukturę określa „schemat schematów" narzucający predefiniowany zestaw elementów. Aby jednak ułatwić programistom rozszerzanie koncepcji schematu, owe predefiniowane elementy „zanurzone" zostały w swej własnej przestrzeni nazw, zatem muszą być specyfikowane z wykorzystywaniem (konsekwentnym) prefiksu identyfikującego tę przestrzeń. Mimo iż może to wydawać się wielce uciążliwe, jest konieczne do tego, by standard XML był tak elastyczny, jak to tylko jest możliwe. Wiele zaawansowanych edytorów XML uwalnia zresztą użytkowników (w dużym stopniu) od tej uciążliwości, m.in. przez automatyczne dodawanie znaczników zamykających opatrzonych właściwymi prefiksami.

Podobnie jak przestrzenie nazw w ogólności, tak i przestrzeń nazw schematu definiowana jest w elemencie nadrzędnym (*root*) dokumentu. W definicji tej konieczne jest odwołanie do oryginalnej definicji „schematu schematów" znajdującej się pod adresem *http://www.*
↪w3.org/2001/XMLSchema, przykładowy element nadrzędny mógłby więc rozpoczynać się następującym znacznikiem:

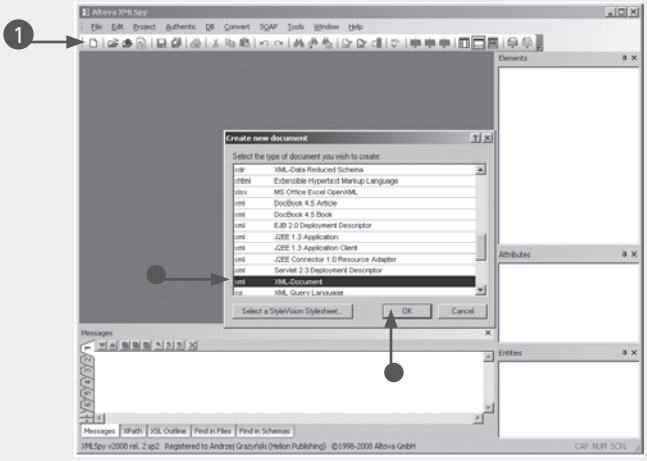
```
<xs:schema xmlns:xs="http://www.w3.org/2001/
↪XMLSchema">
```

Zwróćmy uwagę na znany już atrybut `xmlns` oraz frazę `:schema` po definicji przedrostka, która oznacza, że przestrzeń nazw dotyczy właśnie schematu; zdarza się, że początkujący programiści zapominają o tych szczegółach.

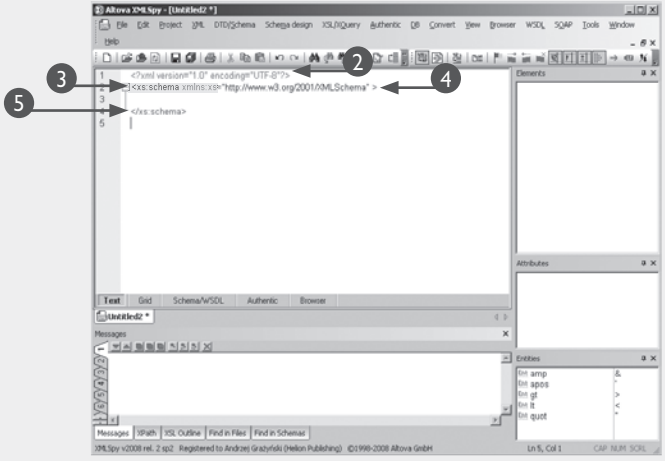
Zgodnie z wymogami „schematu schematów" nazwy elementów zapisywane są z użyciem mieszanej wielkości liter, a dokładniej nazwa rozpoczyna się małą literą, zaś początek każdego jej „słowa składowego" sygnalizowany jest wielką literą. Dotyczy to zarówno nazw samych elementów, jak i nazw i wartości atrybutów.

Wykorzystywanie przestrzeni nazw

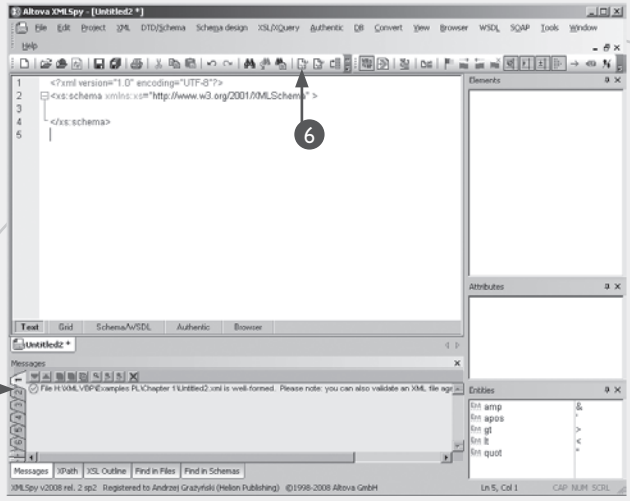
1. Otwórz nową stronę w edytorze XML.
   - W edytorze XMLSpy kliknij w tym celu przycisk *New Document* i wybierz z wyświetlonej listy pozycję *XML Document*.



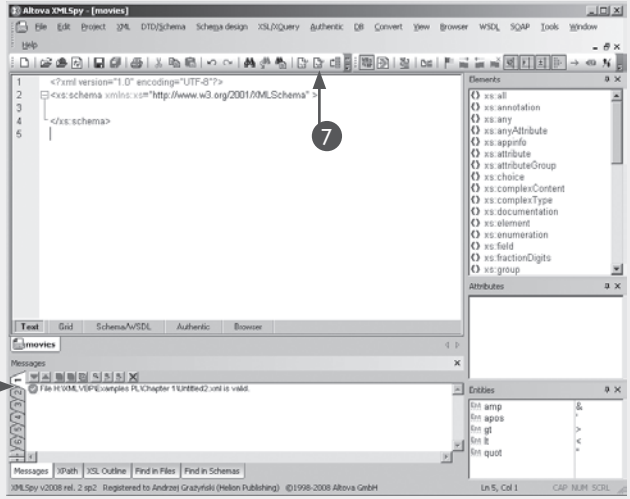
2. Dodaj prolog XML.
3. Dodaj element *root* schematu.
4. Utwórz odwołanie do przestrzeni nazw identyfikowanej przez prefiks `xs`.
5. Dodaj znacznik zamykający element *root*.



6. Kliknij przycisk *Check Well-Formedness*, by sprawdzić poprawność składniową dokumentu.
   - Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.



7. Kliknij przycisk *Check Validity* w celu sprawdzenia zgodności dokumentu ze schematem.
   - Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.
8. Zapisz dokument w pliku z rozszerzeniem *.xsd*.



Wskazówka

Technicznie rzecz biorąc, prefiks przestrzeni nazw schematu może mieć dowolną postać, określoną przez atrybut `xmlns`, na przykład:

```
<myschema:schema xmlns:myschema ="http://www.w3.org/2001/XMLSchema">
```

Gdy jednak weźmie się pod uwagę fakt, że prefiks ten będzie wykorzystywany w treści dokumentu dość intensywnie, staje się oczywiste, iż powinien on składać się z raczej niewielkiej liczby liter. W niniejszej książce konsekwentnie używamy przedrostka `xs`, który stał się już powszechnie przyjętym standardem w tej roli.

Konsorcjum World Wide Web realizuje swój serwis internetowy na bazie serwerów uniksowych. Jak wiadomo, UNIX, w odróżnieniu od Windows, jest systemem wrażliwym na wielkość liter w nazwach plików. Dla przeciętnego użytkownika Internetu fakt ten nie ma znaczenia o tyle, że pliki zawierające dane powiązane ze stronami WWW opatrywane są zazwyczaj nazwami składającymi się wyłącznie z małych liter. Konsorcjum W3C wyłamuje się jednak z tej konwencji, stosując różnicowanie wielkości liter — a skoro tak, to wszelkie odwołania do nazw plików muszą być zgodne z faktycznymi nazwami tychże plików *także co do wielkości liter*, w przeciwnym razie przeprowadzenie walidacji dokumentu opisującego schemat okaże się niemożliwe.

Elementy złożone

W dokumencie XML mogą występować dwa typy elementów. Elementy *proste* to takie, które zawierają jedynie dane i pozbawione są atrybutów. Elementy nie posiadające tej własności, w szczególności zagnieżdżające w sobie inne elementy i (lub) wykorzystujące atrybuty, to elementy *złożone* — w typowym dokumencie jest ich więcej niż elementów prostych.

W definicji schematu opis struktury elementu rozpoczyna się od znacznika `xs:element`, w którym specyfikuje się nazwę odnośnego elementu. Opis struktury elementu złożonego jest natomiast określony (w definicji schematu) za pomocą elementu `xs:complexType`. Jeżeli docelowy element zagnieżdżać będzie w sobie inne elementy (co jest prawdą w przypadku wszystkich niemal elementów *root*), fakt ten odzwierciedla się w definicji schematu za pomocą elementu `xs:sequence`. Każdy z zagnieżdżonych elementów (docelowych) opisywany jest z kolei (w definicji schematu) przez osobny element `xs:element`, zagnieżdżony wewnątrz elementu `xs:sequence`; opis ten sygnalizuje jednak tylko fakt zagnieżdżania elementu potomnego, bez opisywania tegoż. Oto przykładowa definicja elementu `kolekcjaFilmow`, zagnieżdżającego w sobie element (lub sekwencję elementów) `film`:

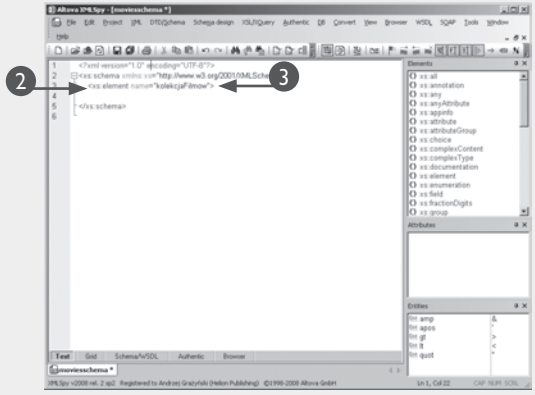
```
<xs:element name="kolekcjaFilmow">
<xs:complexType>
    <xs:sequence>
        <xs:element name="film" />
    </xs:sequence>
</xs:complexType>
</xs:element>
```

Alternatywą dla deklaracji `<xs:sequence>` jest deklaracja `<xs:choice>`, dająca możliwość wyboru użytego elementu:

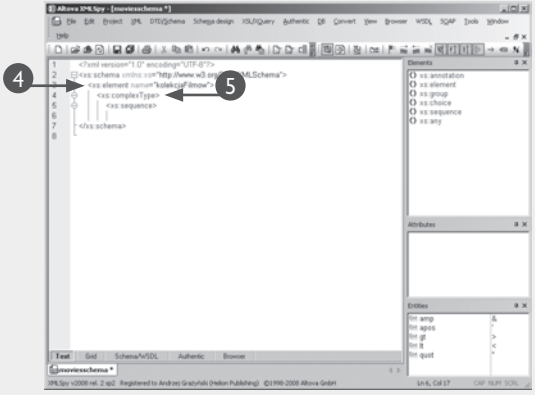
```
<xs:element name="kolekcjaFilmow">
<xs:complexType>
    <xs:choice>
        <xs:element name="film" />
        <xs:element name="zapowiedz" />
    </xs:choice>
</xs:complexType>
</xs:element>
```

Dodawanie elementów złożonych

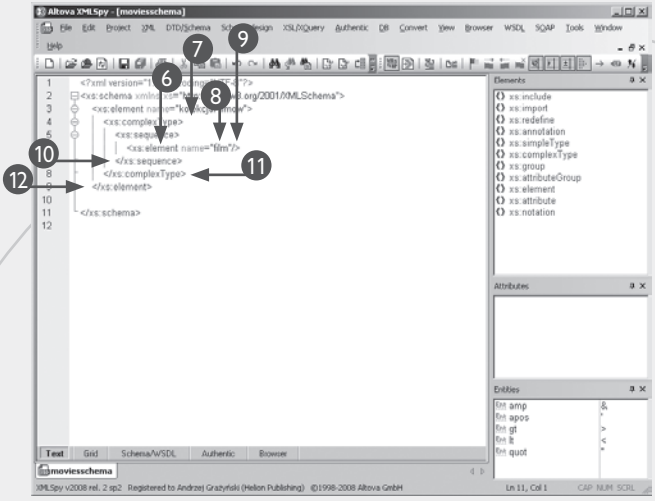
1. Otwórz dokument schematu XML.
2. Dodaj znacznik otwierający `<xs:element>`.
3. Przypisz nazwę do atrybutu `name` elementu *root*.



4. Dodaj znacznik otwierający `<xs:complexType>`.
5. Dodaj znacznik otwierający `<xs:sequence>`.



6. Dodaj znacznik otwierający `<xs:element>`.
7. Dodaj atrybut `name` elementu.
8. Przypisz temu atrybutowi nazwę elementu potomnego.
9. Uczyń znacznik `<xs:element>` samozamykającym.
10. Dodaj znacznik zamykający element `<xs:sequence>`.
11. Dodaj znacznik zamykający element `<xs:complexType>`.
12. Dodaj znacznik zamykający element `<xs:element>`.



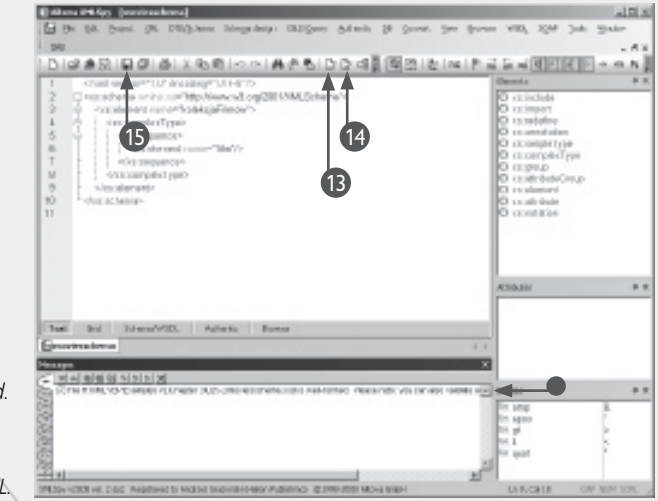
13. Kliknij przycisk *Check Well-Formedness*, by sprawdzić poprawność składniową dokumentu.

    Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.

14. Kliknij przycisk *Check Validity* w celu sprawdzenia zgodności dokumentu ze schematem.
    - Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.

15. Zapisz dokument w pliku z rozszerzeniem *.xsd*.

Uwaga: *Dokument powinien zostać zapisany w pliku z rozszerzeniem .xsd, aby mógł być bezbłędnie rozpoznawany jako schemat XML.*



Zastosuj to

W ramach elementu `<xs:sequence>` możliwe jest określenie sekwencji *następstwa* elementów — pod pojęciem „następstwa" rozumiemy określone elementy występujące w określonej kolejności, na przykład:

```
<xs:element name="film">
    <xs:complexType name="film">
        <xs:sequence>
            <xs:element name="tytul" />
            <xs:element name="recenzja" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

Typy danych

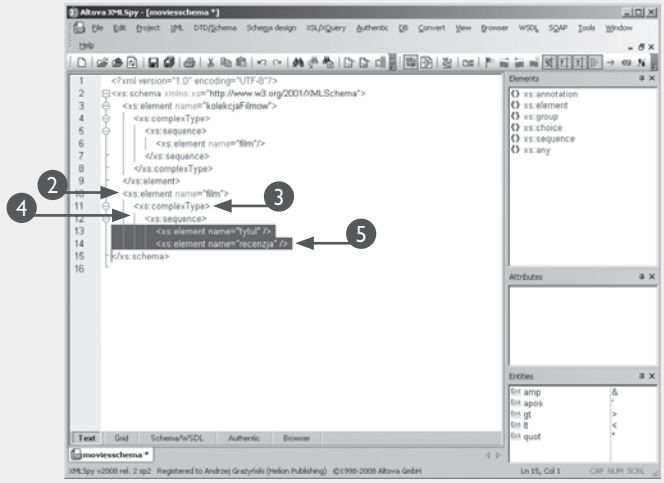
Jedną z podstawowych cech aplikacji bazodanowych (i większości języków programowania) jest definiowanie, różnicowanie i kontrolowanie typów danych — specyfikując pole bazy danych, związujemy z nim zwykle dane odpowiedniego typu: tekst, liczbę całkowitą, liczbę dziesiętną, datę itp. W podobny sposób schematy dają możliwość określania typów danych występujących w elementach XML. Dopuszczalnych jest wiele typów elementów, najczęściej jednak wykorzystywane są: `string` dla zwykłego tekstu, `int` dla liczb całkowitych, `decimal` dla liczb zawierających część ułamkową i `boolean` dla logiki dwuwartościowej `true/false`.

Zgodność postaci danych elementu ze specyfikowanym ich typem jest jednym z warunków poprawności dokumentu XML — parser zakwestionuje wszystkie przypadki, w których brak jest takiej zgodności.

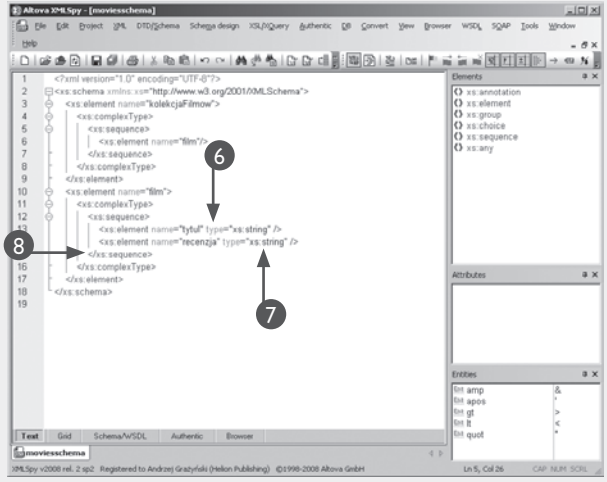
Miejscem, w którym definiuje się typ danych odnośnego elementu, jest atrybut `type` znacznika `<xs:element>` otwierającego definicję tegoż elementu. W przypadku zagnieżdżonej hierarchii elementów (z użyciem znaczników `xs:choice` lub `xs:sequence`) dotyczy to elementów na najniższym poziomie zagnieżdżenia. W przypadku elementu złożonego, który elementem złożonym jest jedynie z racji posiadania atrybutu, typ danych specyfikujemy za pomocą atrybutu `type` znacznika `xs:extension`. Ponieważ dopuszczalne typy danych predefiniowane są w „schemacie schematów", wszelkie wystąpienia ich identyfikatorów muszą być poprzedzone prefiksem identyfikującym przestrzeń nazw schematu.

Dodawanie typów danych

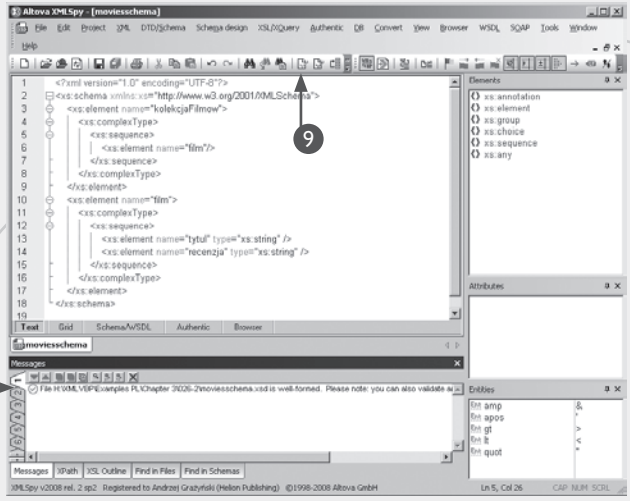
1. Otwórz dokument schematu XML.
2. Dodaj znacznik otwierający `<xs:element>`.
3. Dodaj znacznik otwierający `<xs:complexType>`.
4. Dodaj znacznik otwierający `<xs:sequence>` lub `<xs:choice>`.
5. Dodaj jeden lub kilka znaczników otwierających `<xs:element>`.



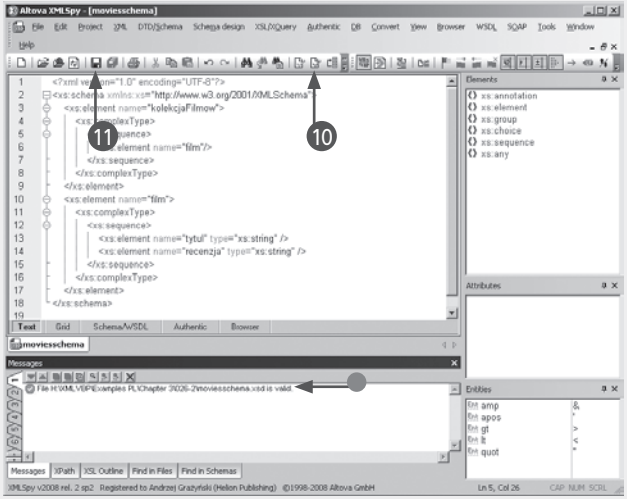
6. Dodaj do każdego z tych znaczników atrybut `type`.
7. Przypisz każdemu z tych atrybutów stosowną wartość.
8. Zamknij wszystkie elementy otwarte w punkcie 5.



9. Kliknij przycisk *Check Well-Formedness*, by sprawdzić poprawność składniową dokumentu.
   - Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.



10. Kliknij przycisk *Check Validity* w celu sprawdzenia zgodności dokumentu ze schematem.
    - Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.
11. Kliknij przycisk *Save*. Dokument zostanie zapisany.



Wskazówka

Należy starannie upewnić się, że istotnie wybrano właściwy typ danych dla elementu. Jednym z najczęstszych błędów jest kwalifikowanie jako typy numeryczne danych takich jak numer telefonu, kod pocztowy, numer NIP itp. Owszem, dane te zawierają cyfry, lecz cyfrom tym towarzyszą dodatkowe znaki w rodzaju nawiasów, myślników itp., nieakceptowane w polu numerycznym. Należy przyjąć zasadę, iż typy numeryczne rezerwujemy wyłącznie dla danych podlegających obliczeniom matematycznym, takich jak cena czy ilość. „Schemat schematów" definiuje rozmaite odmiany typów numerycznych, umożliwiających reprezentowanie liczb całkowitych i dziesiętnych, wraz z ograniczeniem zakresu lub długości części ułamkowej (liczby cyfr dziesiętnych).

W związku z powyższym należy regularnie przeprowadzać walidację dokumentów definiujących schematy. W przeciwieństwie do „zwykłych" dokumentów XML, które co prawda mogą, lecz wcale nie muszą odwoływać się do schematów, każda definicja schematu powołuje się z konieczności na „schemat schematów" i jako taka nieuchronnie podlega walidacji.

Wiele edytorów XML, m.in. Altova XMLSpy, oferuje wbudowane walidatory, które znacząco upraszczają ten proces.

Atrybuty

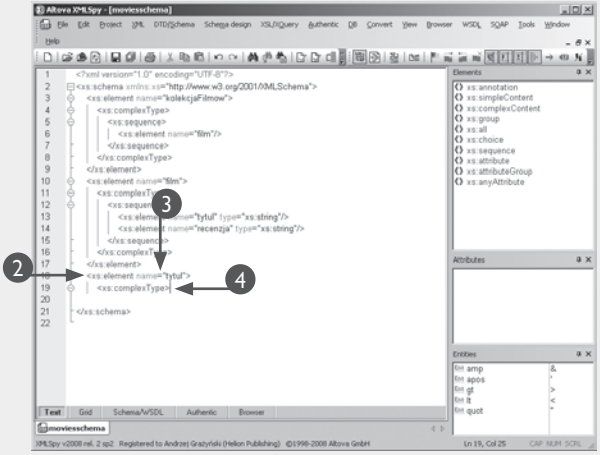
Wiele elementów posiada atrybuty, których rolą jest dokładniejszy opis zawartości. W schemacie atrybuty definiowane są w ramach definicji elementu złożonego. Jeśli ów docelowy element posiada jedynie dane, fakt ten odzwierciedlony zostaje przez deklarację `xs:simpleContent`; element zagnieżdżający inne elementy deklarowany jest jako `xs:complexContent`. W obydwu przypadkach typ atrybutu odnośnego elementu deklarowany jest przez atrybut `base` znacznika `xs:extension`, najczęściej jako `xs:string` (dla tekstów) lub `xs:int` (dla liczb całkowitych). Nazwa atrybutu definiowana jest (w schemacie) przez atrybut `name` elementu `xs:attribute` zagnieżdżonego wewnątrz elementu `xs:extension`. W elemencie `xs:attribute` można także określić, za pomocą atrybutu `use`, sposób traktowania atrybutu: `required` (obowiązkowy), `optional` (nieobowiązkowy) albo `prohibited` (niedozwolony).

Opisane zagnieżdżanie definicji może wydawać się dość przytłaczające, lecz staje się coraz bardziej zrozumiałe w miarę nabywania doświadczenia w pracy ze schematami. Czytając opisaną wyżej definicję, można w naturalny sposób wywnioskować, że mamy do czynienia z elementem złożonym (`complexType`), o prostej zawartości (`simpleContent`), a typ jego danych rozszerzany jest (`extension`) przez atrybut o wartości określonego typu. Młodsze od XML-a języki służące do definiowania danych nie mają już co prawda tak „rozgadanego" charakteru, lecz jednocześnie brak im elastyczności właściwej schematom XML.

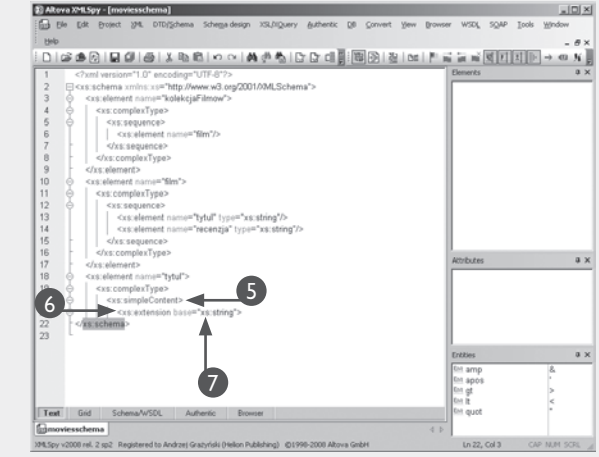
Użycie elementu `xs:extension` jest konieczne, ponieważ definicja elementów prostych (`xs:simpleType`) nie pozwala na definiowanie typów danych. Istotą `xs:extension` jest rozszerzenie (*extension*) typu danych na *atrybut* elementu, nie na sam element. To notabene jedna z takich subtelności, czyniących prawdziwe zrozumienie schematów XML niezwykle trudnym.

Dodawanie atrybutów

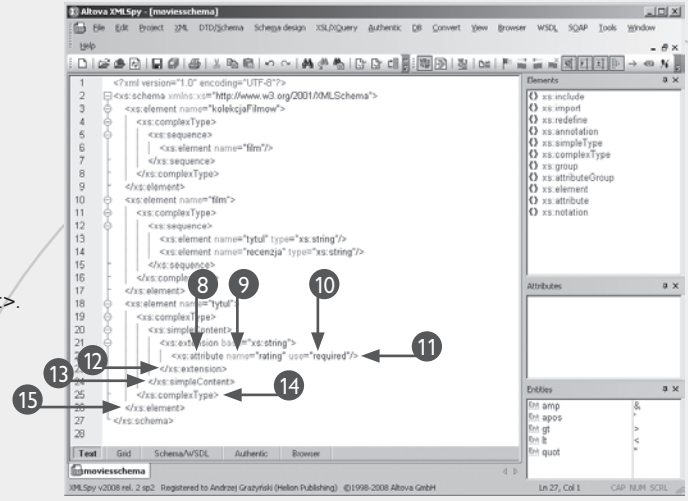
1. Otwórz dokument schematu XML.
2. Dodaj znacznik otwierający `<xs:element>`.
3. Nadaj elementowi nazwę, przypisując ją do atrybutu `name`.
4. Dodaj znacznik otwierający `<xs:complexType>`.



5. Dodaj znacznik otwierający `<xs:simpleContent>`.
6. Dodaj znacznik otwierający `<xs:extension>`.
7. Przypisz atrybutowi `base` ww. znacznika odpowiedni typ danych.



8. Dodaj znacznik otwierający `<xs:attribute>`.
9. Określ nazwę atrybutu docelowego.
10. Określ obowiązkowy charakter atrybutu docelowego.
11. Uczyń znacznik `<xs:attribute>` samozamykającym.
12. Zamknij element `<xs:extension>`.
13. Zamknij element `<xs:simpleContent>`.
14. Zamknij element `<xs:complexType>`.
15. Zamknij element `<xs:element>`.



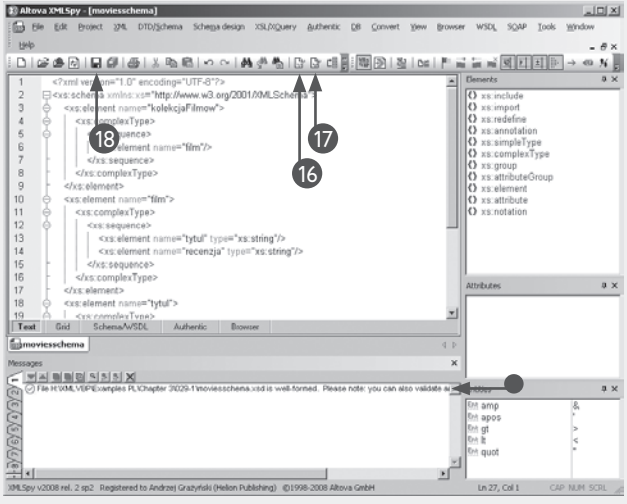
16. Kliknij przycisk *Check Well-Formedness*, by sprawdzić poprawność składniową dokumentu.

    Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.

17. Kliknij przycisk *Check Validity* w celu sprawdzenia zgodności dokumentu ze schematem.
    - Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.

18. Kliknij przycisk *Save*.

    Dokument zostanie zapisany.



Zastosuj to

Możliwe jest nadanie globalnego charakteru definicjom atrybutów, na początku definicji schematu, i odwoływanie się do tychże definicji w całym dokumencie. Robi się to za pomocą znacznika `xs:attributeGroup`:

```
<xs:attributeGroup name="fileattrs">
    <xs:attribute name="path" type="xs:string" />
</xs:attributeGroup>
<xs:element name="filename">
    <xs:simpleContent>
       <xs:extension base="xs:string">
           <xs:attributeGroup ref="fileattrs" />
       </xs:extension>
    </xs:simpleContent>
</xs:element>
```

Elementy proste

W typowym dokumencie XML można znaleźć przynajmniej kilka elementów *prostych* — czyli takich, które nie zagnieżdżają w sobie innych elementów i nie zawierają atrybutów, posiadając co najwyżej dane. W schemacie element prosty deklarowany jest za pomocą samozamykającego znacznika `xs:element`:

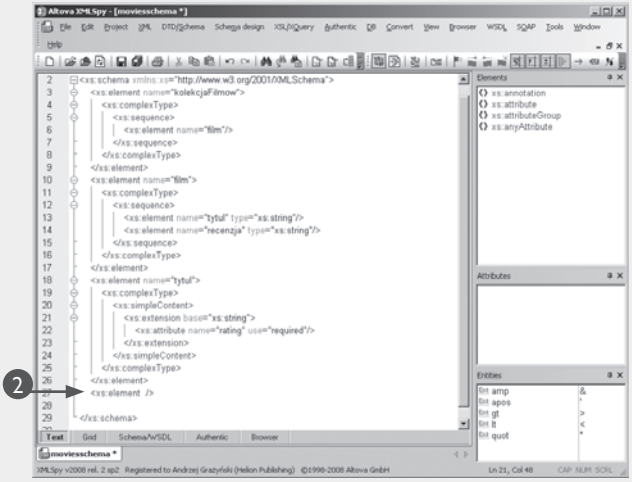
```
<xs:element name="autorRecenzji"
↪type="xs:string" />
```

Jedynym wymaganym atrybutem tego znacznika jest `name`, określający nazwę elementu; niemal zawsze stosuje się także atrybut `type`, ten bowiem określa typ danych posiadanych przez odnośny element. Ponadto za pomocą atrybutów `minOccurs` oraz `maxOccurs` można określić dozwoloną, odpowiednio: minimalną i maksymalną liczbę wystąpień elementu.

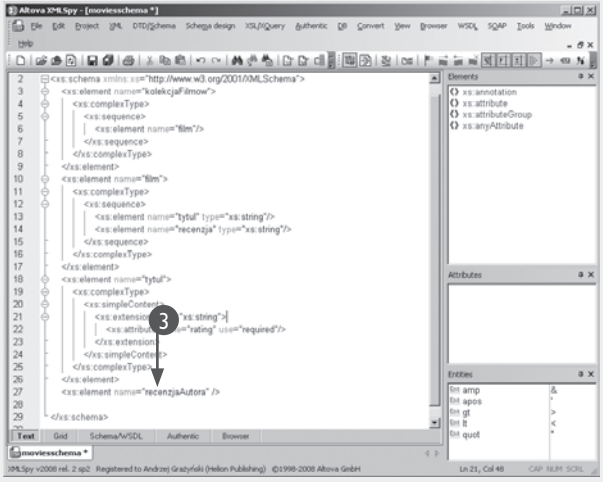
Z wyjątkiem przypadku, gdy elementy deklarowane są w ramach sekwencji `xs:sequence`, kolejność ich występowania w dokumencie jest dowolna, niezależna od kolejności ich deklarowania. Niektórzy programiści zwykli deklarować (w schemacie) najpierw elementy złożone, potem elementy proste, inni z kolei deklarują elementy w kolejności, jakiej oczekiwaliby w docelowym dokumencie. Wszystko to jest kwestią gustu, nie rodzącą żadnych konsekwencji pod adresem dokumentu docelowego.

Dodawanie elementów prostych

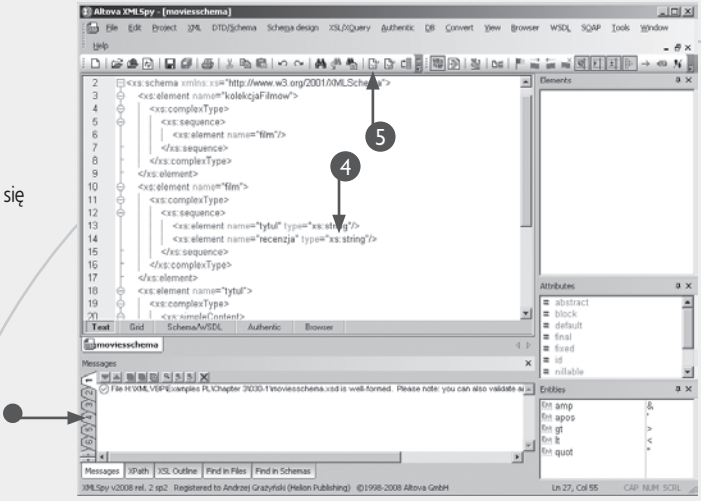
1. Otwórz dokument schematu XML.
2. Dodaj element zagnieżdżony bezpośrednio w `<xs:schema>`.



3. Przypisz atrybutowi `name` nazwę definiowanego elementu.

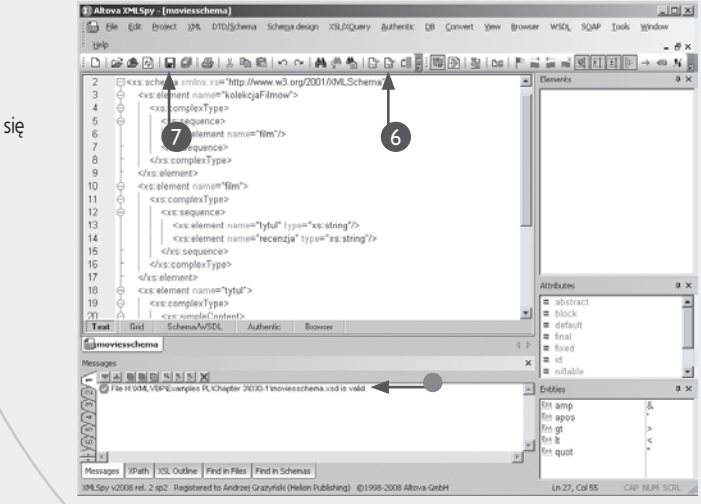


4. Dodaj atrybut `type` i przypisz mu odpowiednią wartość.
5. Kliknij przycisk *Check Well-Formedness*, by sprawdzić poprawność składniową dokumentu.
   - Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.



6. Kliknij przycisk *Check Validity* w celu sprawdzenia zgodności dokumentu ze schematem.
   - Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.
7. Kliknij przycisk *Save*.

   Dokument zostanie zapisany.



Wskazówka

Dodatkowe wymogi pod adresem deklarowanego elementu można określić za pomocą tzw. *faset* (ang. *facets*). „Schemat schematów" oferuje osiem takich faset, spośród których najczęściej używane są te ograniczające długości łańcuchów, definiujące zbiory wartości domyślnych i ograniczające zakres danych liczbowych. Wszystkie one definiowane są pod postacią elementu `xs:restrictions`.

W szczególności `xs:minLength` i `xs:MaxLength` mogą być użyte do określenia minimalnej i maksymalnej długości łańcucha, a za pomocą `xs:enumeration` specyfikuje się zestaw możliwych wartości do wyboru. Wreszcie, dopuszczalny zakres wartości liczbowych określić można za pomocą faset `xs:minInclusive`, `xs:maxInclusive`, `xs:minExclusive` i `xs:maxExclusive`.

Oto przykład użycia faset:

```
<xs:element name = "password">
<xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:minLength value="8"  />
      <xs:maxLength value="16" />
    </xs:restriction>
</xs:simpleType>
</xs:element>
```

Elementy mieszane

Elementy złożone to elementy zagnieżdżające w sobie inne ele menty i nie posiadające danych. Jedno nie wyklucza jednak drugiego: elementy zawierające zarówno dane, jak i elementy potomne, noszą nazwę elementów *mieszanych*.

Rozpatrzmy przykładową recenzję filmu: choć może być ona zwykłym tekstem, to równie dobrze może mieć strukturę bardziej skomplikowaną, na przykład w celu uwidocznienia cytowanego tytułu:

```
<recenzja>
  <para>
    <tytul>Rozmowy kontrolowane</tytul>
to kontynuacja losów Ryszarda Ochódzkiego
znanego z komedii
    <tytul>Miś</tytul>.
  </para>
</recenzja>
```

W powyższym przykładzie element `recenzja` zagnieżdża w sobie element potomny `para`, który z kolei zagnieżdża w sobie zarówno dwa elementy potomne, jak i dane w postaci otartego tekstu. Element `para` powinien zatem zostać zdefiniowany w schemacie jako element mieszany.

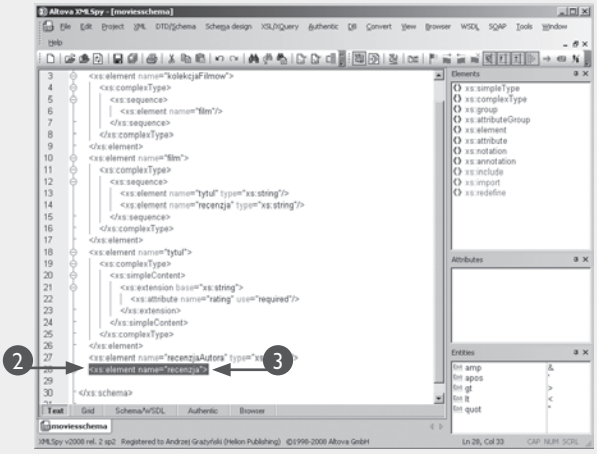
Definicja elementu mieszanego jest w istocie definicją elementu złożonego, używającą atrybutu `mixed`:

```
<xs:element name="para">
  <xs:complexType mixed="true">
    <xs:choice>
      <xs:element name="title" />
    </xs:choice>
  </xs:complexType>
</xs:element>
```

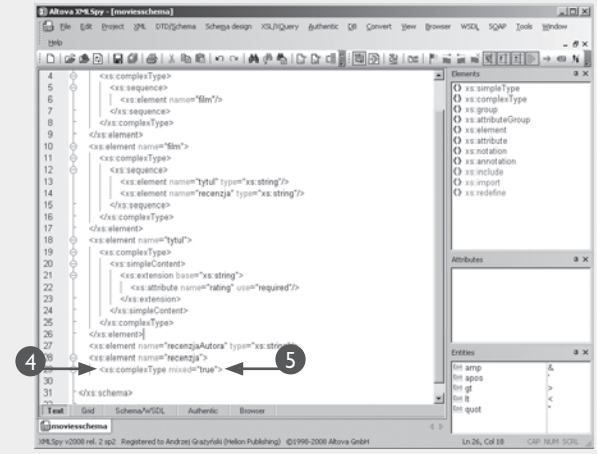
W definicji takiej deklarowany jest zarówno wybór (`choice`) innych elementów, jak i typ danych posiadanych przez element.

Dodawanie elementów złożonych

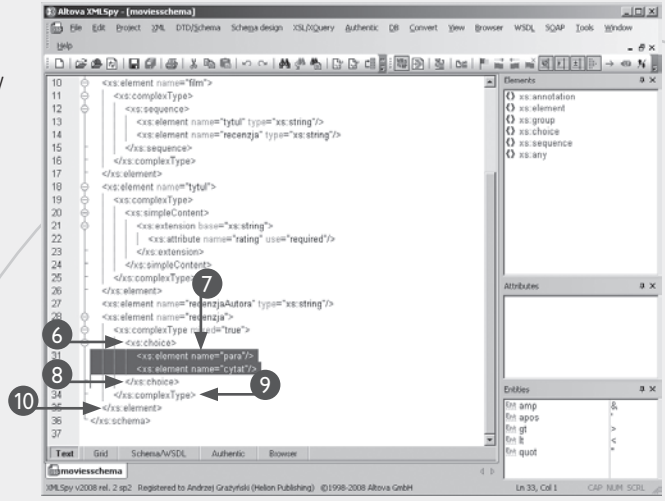
1. Otwórz dokument schematu XML.
2. Dodaj znacznik otwierający `<xs:element>`.
3. Określ nazwę elementu.



4. Dodaj znacznik otwierający `<xs:complexType>`.
5. Dodaj atrybut `mixed="true"`.



6. Dodaj znacznik otwierający `<xs:choice>`.
7. Dodaj dwa lub więcej znaczników `<xs:element>` reprezentujących elementy potomne.
8. Zamknij element `<xs:choice>`.
9. Zamknij element `<xs:complexType>`.
10. Zamknij element `<xs:element>`.



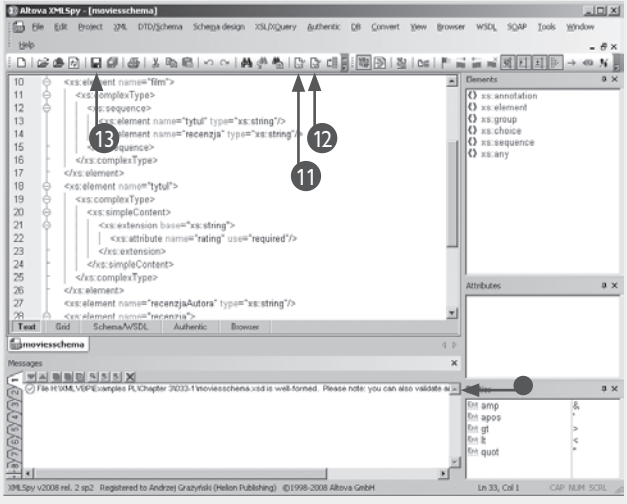
11. Kliknij przycisk *Check Well-Formedness*, by sprawdzić poprawność składniową dokumentu.

    Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.

12. Kliknij przycisk *Check Validity* w celu sprawdzenia zgodności dokumentu ze schematem.
    - Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.

13. Kliknij przycisk *Save*.

    Dokument zostanie zapisany.



Wskazówka

Oprócz `xs:choice` dostępne są jeszcze dwie formy specyfikacji zagnieżdżanych elementów: `xs:sequence` i `xs:all`. Jak wiadomo, `xs:choice` specyfikuje listę wzajemnie wykluczających się elementów, z których wystąpić musi dokładnie jeden; `xs:sequence` specyfikuje listę elementów, które wystąpić muszą w ściśle określonej kolejności; `xs:all` specyfikuje elementy, z których każdy wystąpić musi dokładnie raz, jednak kolejność wystąpienia poszczególnych elementów jest dowolna. Jeśli wykorzystywane jest `xs:all`, musi pojawić się na początku dokumentu schematu i najczęściej pojawia się w definicji elementu *root*.

Każdy z trzech wymienionych selektorów może, choć nie musi pojawić się w definicji elementu złożonego (i faktycznie często się pojawia), w szczególności elementu *root*. Jedyną cechą, która odróżnia element mieszany od elementu złożonego, jest obecność atrybutu `mixed` w znaczniku `xs:complexType`. Wybór między użyciem elementu złożonego a użyciem elementu mieszanego jest zawsze kwestią konkretnego zastosowania i osobistego upodobania programisty.

Ograniczenie występowania elementów

Wymienione wcześniej selektory stwarzają dość dużą elastyczność w zakresie używania elementów potomnych, często jednak konieczne jest określenie wymagań jeszcze bardziej rygorystycznych. Przykładowo, lista zawodników meczu piłki nożnej składać się musi z dokładnie 11 elementów.

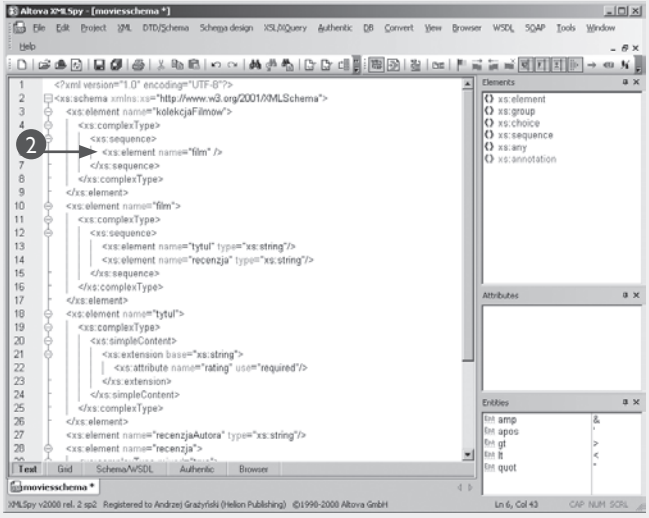
Schematy XML dają możliwość określenia zarówno minimalnej, jak i maksymalnej liczby wystąpień danego elementu; w opisanym przypadku obydwie te wartości, specyfikowane przez atrybuty `minOccurs` i `maxOccurs` znacznika `xs:element`, muszą mieć wartość 11.

Przez przypisanie atrybutowi `minOccurs` wartości `0` sprawiamy, że odnośny element staje się opcjonalny, czyli nie musi w ogóle występować. Zdjęcie *górnego* ograniczenia na liczbę występowania elementu odbywa się przez przypisanie atrybutowi `maxOccurs` wartości `unbounded`. Nadanie obydwu atrybutom tej samej wartości oznacza żądanie wystąpienia elementów w ściśle określonej liczbie — 11 w przytoczonym przykładzie drużyny piłkarzy.

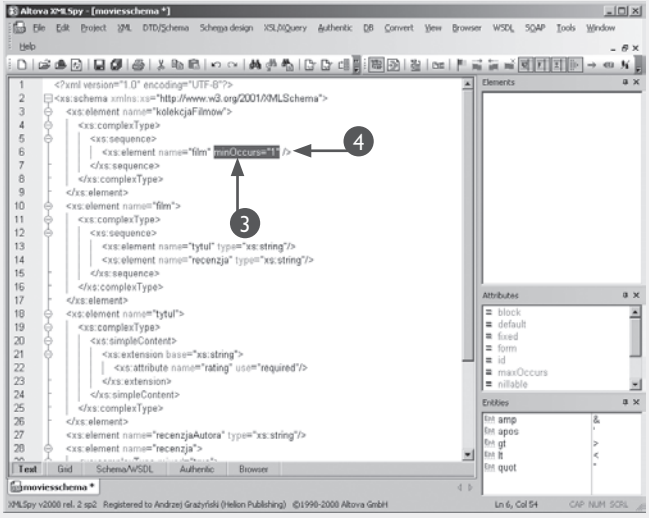
Domyślną wartością każdego z atrybutów `minOccurs` i `maxOccurs` jest 1 — gdy atrybuty te nie zostaną jawnie określone, odnośny element wystąpić musi dokładnie jeden raz. Ponieważ z natury wymaganie takie narzucone jest na element *root*, wspomniane atrybuty nie mogą wystąpić w jego definicji.

Definiowanie dopuszczalnej liczby wystąpień elementów

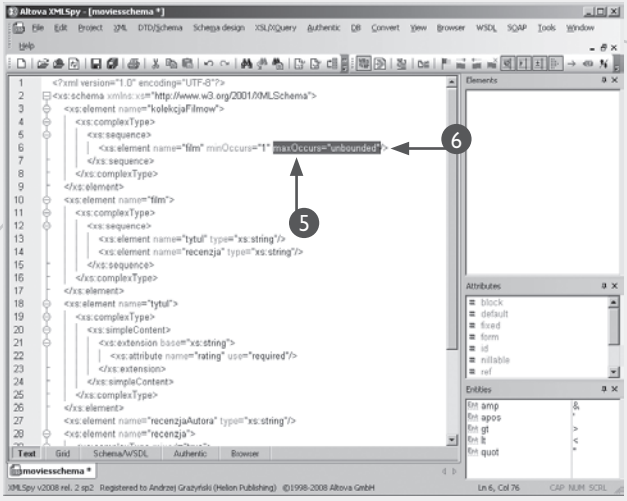
1. Otwórz dokument schematu XML.
2. Dodaj nowy (lub wykorzystaj istniejący) znacznik otwierający `<xs:element>`.



3. Dodaj atrybut `minOccurs`.
4. Przypisz atrybutowi wartość 0 oznaczającą, że użycie elementu jest opcjonalne.



5. Dodaj atrybut `maxOccurs`.
6. Przypisz atrybutowi wartość `"unbounded"` lub konkretną liczbę.

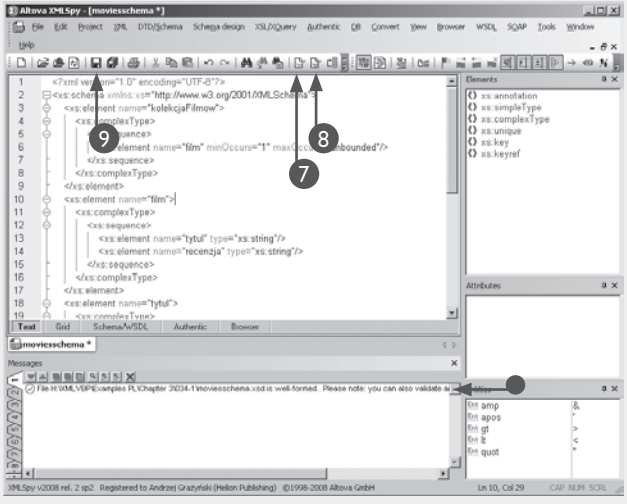


7. Kliknij przycisk *Check Well-Formedness*, by sprawdzić poprawność składniową dokumentu.

   Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.

8. Kliknij przycisk *Check Validity* w celu sprawdzenia zgodności dokumentu ze schematem.
   - Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.

9. Zapisz dokument w pliku z rozszerzeniem *.xsd*.



Wskazówka

Staranne zaplanowanie jest kluczem do stworzenia właściwego schematu. Dokumenty XML wrażliwe są na wielkość liter; chociaż plik schematu posiada specyficzne rozszerzenie, jest dokumentem XML i wszelkie użyte w nim nazwy elementów, atrybutów itp. muszą być tożsame także co do wielkości liter z odpowiednimi nazwami w dokumencie docelowym. Z tego względu niezwykle ważną rzeczą jest opracowanie spójnej metody różnicowania wielkości liter („kapitalizacji") i konsekwentne jej stosowanie zarówno w schemacie, jak i dokumentach na tym schemacie opartych. Alternatywą dla różnicowania wielkości liter w identyfikatorach „wielosłowowych" jest używanie myślników i znaków podkreślenia — jeśli się na to decydujemy, także to powinniśmy robić w sposób spójny.

Należy ponadto zwracać szczególną uwagę na wartości przypisywane atrybutom `minOccurs` i `maxOccurs`, rozważając przypadki, kiedy to dopuszczalne jest wystąpienie dokumentu o jeden raz więcej lub o jeden raz mniej od „typowej" wartości.

Co do rygoryzmu kontroli wynikającej ze schematu: zbyt restrykcyjne schematy okazują się na tyle krępujące (dla twórców docelowych dokumentów), że w praktyce okazać się mogą nieużyteczne; z drugiej jednak strony schemat powinien być na tyle restrykcyjny, by dawać jak najlepszą gwarancję kontroli poprawności danych.

Kojarzenie dokumentu XML ze schematem

Aby walidator mógł przeprowadzić kontrolę poprawności dokumentu względem pewnego schematu, musi uzyskać informację o lokalizacji tegoż schematu. Z pojedynczym schematem może być skojarzonych wiele dokumentów.

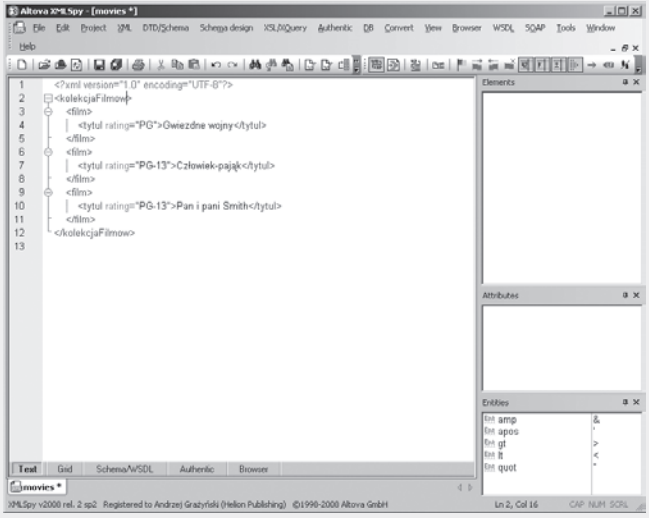
Informacji o lokalizacji schematu dostarcza atrybut `noNamespaceSchemaLocation` w znaczniku otwierającym element *root*. Bezpośrednie użycie tego atrybutu jest jednak niedopuszczalne, bowiem wywodzi się on z własnej przestrzeni nazw, a konkretnie z przestrzeni `XMLSchemaInstance`. Aby zatem poprawnie połączyć dokument z odpowiednim schematem, atrybut ów należy poprzedzić prefiksem identyfikującym wspomnianą przestrzeń — zwykle używa się w tym celu przedrostka `xsi`.

Sama ścieżka wskazująca położenie schematu może być absolutną lub relatywną względem położenia samego dokumentu, może też mieć postać URL-a wskazującego zdalny zasób w sieci (absolutnego lub relatywnego). Oto przykład poprawnego wskazania schematu, którego plik znajduje się w tym samym katalogu co kontrolowany dokument:

```
<myRoot xmlns:xsi=
↪"http://www.w3.org/2001/XMLSchema-instance"
xsi:NoNamespaceSchemaLocation="mySchema.xsd" >
```

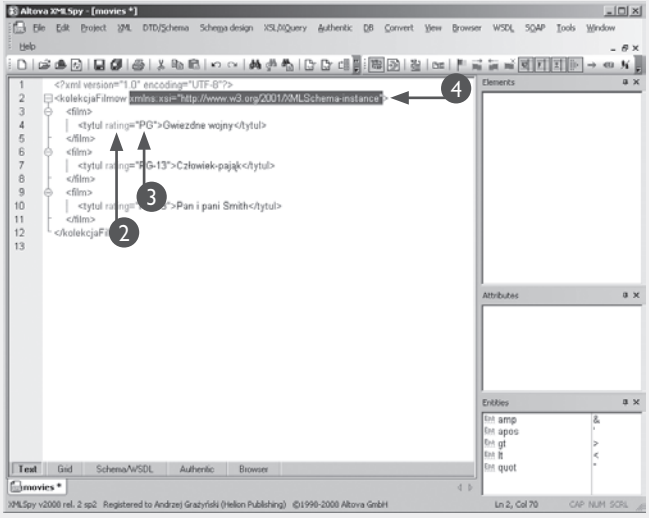
Definiowanie skojarzenia dokumentu XML ze schematem

1. Otwórz dowolny dokument XML bazujący na schemacie.



2. Dodaj atrybut `xmlns` do elementu *root*.
3. Dodaj prefiks `xsi`.
4. Przypisz atrybutowi `xmlns:xsi` wartość **http://www.w3.org/2001/↪XMLSchema-instance**.

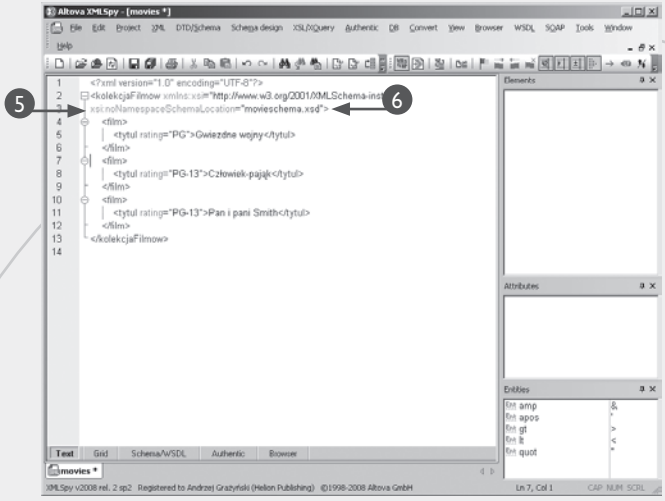
Uwaga: *Zwróć uwagę na różnicowanie wielkości liter.*



5. Dodaj atrybut `xsi:noNamespaceSchemaLocation`.

Uwaga: *Ponownie zwróć uwagę na wielkość liter.*

6. Przypisz powyższemu atrybutowi ścieżkę do pliku `.xsd` zawierającego dokument schematu.

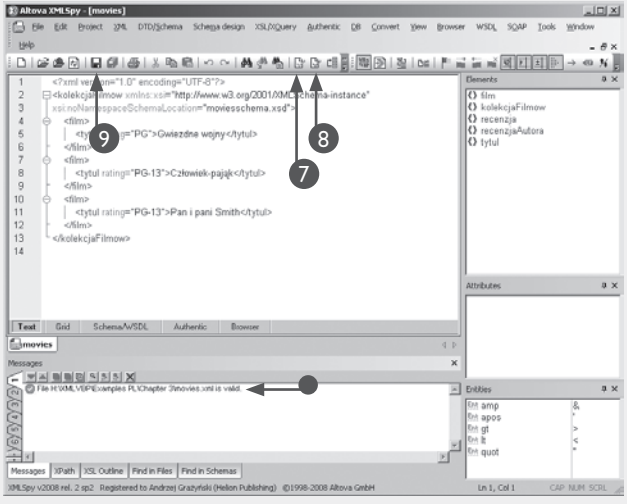


7. Kliknij przycisk *Check Well-Formedness*, by sprawdzić poprawność składniową dokumentu.

   Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.

8. Kliknij przycisk *Check Validity* w celu sprawdzenia zgodności dokumentu ze schematem.
   - Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.

9. Zapisz dokument w pliku z rozszerzeniem *.xsd*.



Zastosuj to

Gdy dokument XML odwołuje się do elementów pochodzących z innej przestrzeni adresowej, odwołanie do schematu związanego z tą przestrzenią powinno odbywać się za pośrednictwem atrybutu `xsi:schemaLocation`, zamiast (lub w uzupełnieniu do) atrybutu `xsi:noNamespaceSchemaLocation`, na przykład tak:

```
<kolekcjaFilmow xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceLocation="movieListSchema.xsd"
xsi:SchemaLocation="http://www.someothersite.com/a-schema.xsd" >
```

Walidacja dokumentu

Należy zwrócić uwagę na fakt, że o ile wszystkie parsery domyślnie kontrolują poprawność syntaktyczną („poprawne sformowanie") dokumentu, niektóre tylko przeprowadzają jego walidację, nawet jeśli w dokumencie jawnie podano odwołanie do schematu.

Różne parsery prezentują przy tym nieco odmienne podejście do samej walidacji. XMLSpy wykonuje tę czynność na żądanie (w wyniku kliknięcia odpowiedniego przycisku), lecz wykonuje ją także w czasie zapisywania dokumentu, wyświetlając stosowny komunikat ostrzegawczy w przypadku stwierdzenia błędu; mimo iż komunikat ten oferuje możliwość zapisania błędnego dokumentu, wskazane jest natychmiastowe poprawienie sygnalizowanych błędów zamiast odkładania tej czynności „na później". Ani Internet Explorer, ani Firefox nie posiadają wbudowanych walidatorów, te jednak dostępne są powszechnie pod postacią rozmaitych rozszerzeń. Inne parsery, jak te wbudowane w aplikacje MS Office, generalnie pozwalają na pracę z dowolnymi dokumentami XML, proponując później przeprowadzenie ich walidacji.

Mimo iż parsery zwykle są odporne na błędy walidacji, niepoprawne dokumenty mogą okazać się kłopotliwe w przetwarzaniu, przykładowo ich transformacja z użyciem XSLT może się załamywać lub dawać nieoczekiwane wyniki. Jest więc niezwykle istotne konsekwentne przeprowadzanie walidacji (i poprawianie napotkanych błędów „na bieżąco") każdorazowo przed użyciem dokumentu.

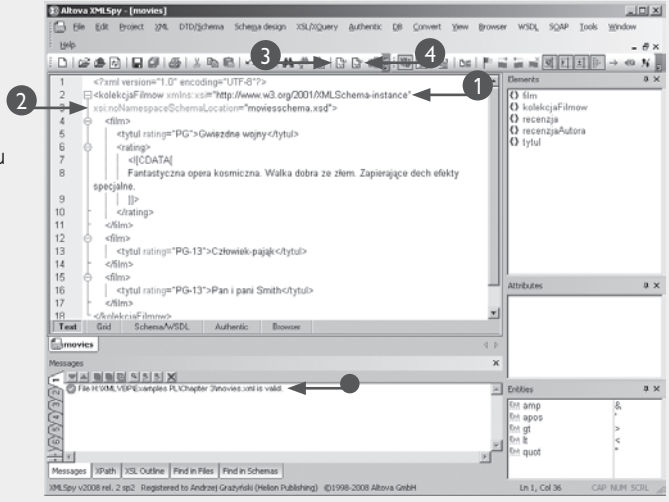
Rozmaite bywa także traktowanie samych błędów walidacji przez poszczególne parsery — niektóre z nich ograniczają się do wyświetlania komunikatów diagnostycznych, inne natomiast traktują błędy jako krytyczne i odmawiają dalszego przetwarzania dokumentu.

Walidacja dokumentu

W edytorze XMLSpy

1. Jeśli to konieczne, zadeklaruj przestrzeń nazw `XMLSchema-instance`.
2. Jeśli to konieczne, użyj atrybutu `xsi:noNamespaceSchemaLocation` w celu odwołania się do schematu.
3. Kliknij przycisk *Check Well-Formedness*, by sprawdzić poprawność składniową dokumentu.

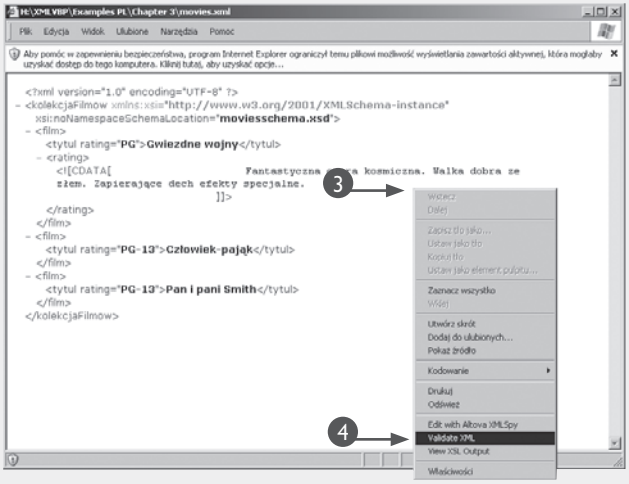
   Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.
4. Kliknij przycisk *Check Validity* w celu sprawdzenia zgodności dokumentu ze schematem.
   - Informacja o poprawności (albo niepoprawności) dokumentu pojawi się w obszarze walidacji.



W Internet Explorerze

1. Pobierz i zainstaluj *Internet Explorer Tools for Validating XML and Viewing XSLT Output* ze strony *www.microsoft.com/downloads/details.aspx?FamilyId=D23C1D2C-1571-4D61-BDA8-ADF9F6849DF9&displaylang=en.*
2. Otwórz dokument XML zawierający odwołanie do schematu.
3. Kliknij dokument prawym przyciskiem myszy.
4. Z menu kontekstowego wybierz opcję *Validate XML.*

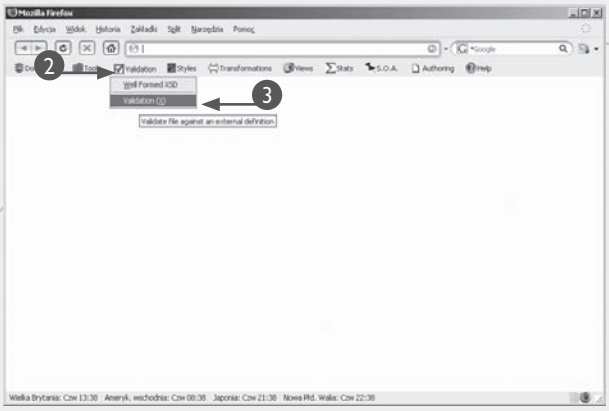
   Wyświetlony zostanie komunikat potwierdzający poprawność (albo niepoprawność) dokumentu.



Walidacja dokumentu w Firefoksie

1. Uruchom Firefox, wejdź na stronę *https://addons.mozilla.org/en-US/firefox/addon/2897* i zainstaluj *XML Developer Toolbar*. Zaakceptuj komunikat informujący o ponownym uruchomieniu Firefoksa.
2. *Na pasku XML Developer kliknij opcję Validation.*
3. *Z wyświetlonego menu kontekstowego wybierz opcję Validation.*

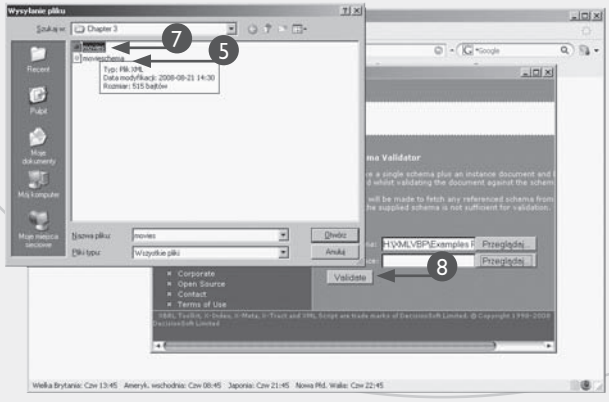
   Uruchomiony zostanie XSD Validator firmy DecisionSoft.



4. *Kliknij przycisk Przeglądaj obok pola XML Schema.*
5. *Wskaż plik zawierający definicję schematu.*
6. *Kliknij przycisk Przeglądaj obok pola XML instance.*

7. *Wskaż plik zawierający weryfikowany dokument.*
8. *Kliknij przycisk Validate.*

   Po zakończeniu walidacji kliknij opcję Click here, aby zobaczyć wyniki.



Zastosuj to

Dostępnych jest wiele narzędzi walidacyjnych dla Internet Explorera i Firefoksa, a także dla innych przeglądarek, jak Safari czy Opera. Większość z nich ogranicza jednak swą funkcjonalność do weryfikacji dokumentu pod kątem zgodności z W3C Schema lub DTD. Walidacja pod kątem zgodności z innymi schematami, jak RELAX NG, oferowana jest jedynie przez wybrane narzędzia.

SourceForge.net, witryna stworzona w celu udostępniania oprogramowania open-source, oferuje wiele walidatorów XML, a także wiele edytorów i parserów, napisanych w różnych językach i przeznaczonych do wielu różnych celów. Mimo iż dostępne jest w tej kolekcji także oprogramowanie komercyjne, większość produktów dostępna jest za darmo. Sama witryna posiada wbudowaną wyszukiwarkę, oferującą zaawansowane funkcje, m.in. system rankingu oceniający stopień zgodności wyszukanych pozycji pod kątem kryterium wyszukiwania. Niektóre produkty oferowane przez SourceForge.net mogą być kłopotliwe w instalowaniu i konfigurowaniu — wówczas pomocne może okazać się bardzo aktywne forum użytkowników.

Wizualne tworzenie schematu za pomocą XMLSpy

XMLSpy jest wysoce funkcjonalną aplikacją pozwalającą radykalnie zredukować czas potrzebny na tworzenie dokumentów XML, arkuszy stylów XSL i schematów. Gdy załaduje się do programu plik schematu, program automatycznie uruchamia widok *Design View* umożliwiający budowanie schematu w sposób wizualny, podczas gdy XMLSpy automatycznie tworzy kod tegoż schematu. Ta i inne funkcje tego rodzaju to zasadniczy powód, dla którego XMLSpy cieszy się takim powodzeniem wśród programistów używających go w codziennej pracy.

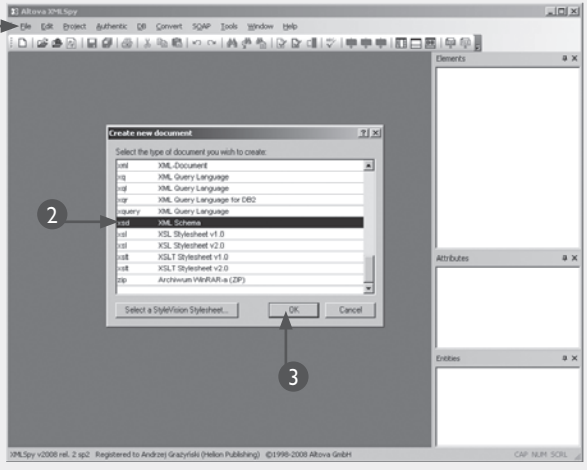
Widok *Design View* może być wyświetlany w dwóch trybach: *Schema Overview* i *Content Model View*. W pierwszym z nich podzielony jest na sekcje — w górnej połowie okna widnieje lista wszystkich elementów definiowanych w schemacie, u dołu natomiast widoczne są wszystkie atrybuty wybranego elementu i związane z nim ograniczenia. Grupa paneli po prawej stronie ekranu umożliwia przeglądanie szczegółów elementów schematu, wraz z zastosowanymi do tych elementów fasetami.

Design View posiada jeszcze szereg innych opcji upraszczających proces tworzenia schematów, między innymi zmianę kolejności elementów za pomocą operacji „przeciągnij i upuść" oraz kopiowanie i wklejanie elementów za pomocą menu kontekstowego, co daje możliwość łatwego kopiowania i przenoszenia elementów między różnymi dokumentami schematów.

W trybie *Content Model View* widoczny jest prosty diagram ukazujący zależności między elementami definiowanymi w schemacie, co znacznie ułatwia analizę złożonych schematów. Diagram ten umożliwia zarówno łatwe nawigowanie po elementach, jak i wizualne edytowanie zależności między nimi.

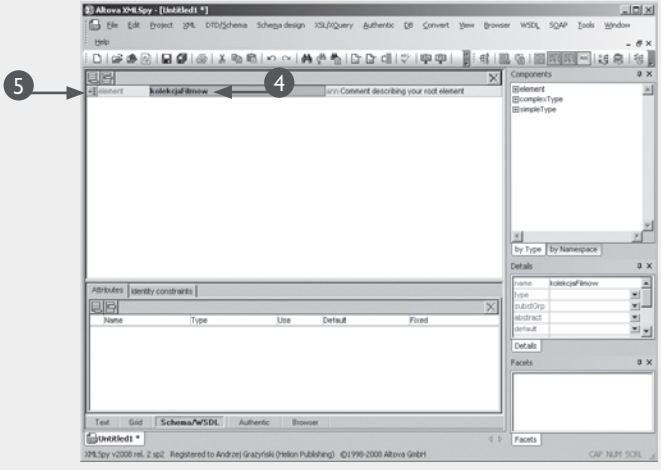
Wizualne tworzenie schematu za pomocą XMLSpy

1. W edytorze XMLSpy wybierz opcję *File/New*. Wyświetlone zostanie okno dialogowe *Create New Document*.
2. Z listy obsługiwanych typów plików wybierz *xsd XML Schema*.
3. Kliknij przycisk *OK*.



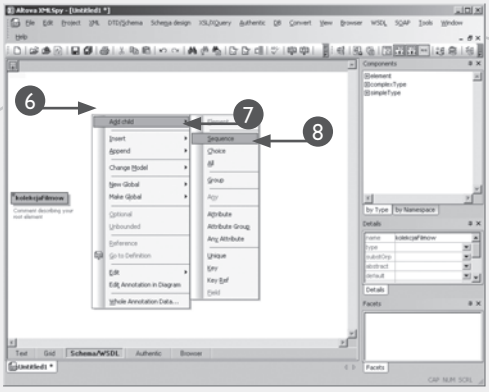
Utworzony zostanie nowy dokument schematu.

4. Zastąp tekst `ENTER_NAME_OF_ROOT_ELEMENT_HERE` wybraną przez Ciebie nazwą elementu *root*.
5. Kliknij ikonę *Content Model View*.



Wyświetlone zostanie okno dialogowe *Content Model View*.

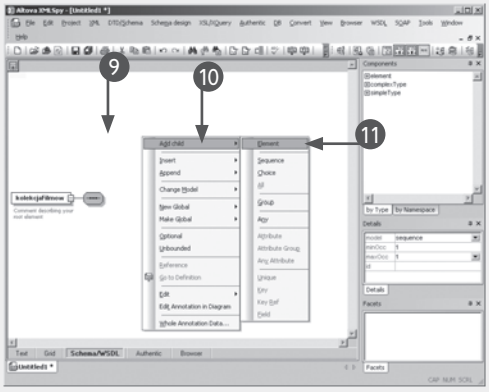
6. Kliknij ww. okno prawym przyciskiem myszy.
7. Z menu kontekstowego wybierz opcję *Add Child*.
8. Z kolejnego menu kontekstowego wybierz opcję *Sequence*.



- Pojawi się kompozytor sekwencji.

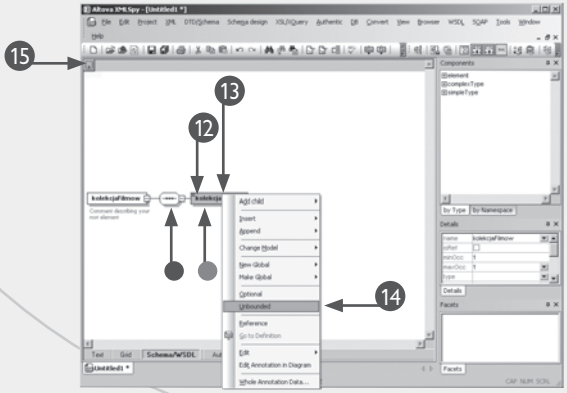
9. Kliknij wyświetlone okno prawym przyciskiem myszy.
10. Z menu kontekstowego wybierz opcję *Add Child*.
11. Kliknij opcję *Element*.

    Zostanie utworzony nowy element.



12. Wprowadź nazwę dla elementu potomnego.
13. Kliknij element prawym przyciskiem myszy.
14. Wybierz opcję *Unbounded*.
    - Widok zostanie zaktualizowany, wskazując, że liczba wystąpień elementu jest nieograniczona.
15. Kliknij ikonę *Show Globals*.

    Wyświetlony zostanie widok *Schema Design*.



Wskazówka

Użytkownicy XMLSpy z łatwością spostrzegą, że edytor ten dodaje do schematu elementy `xs:annotation` i `xs:documentation`. Elementy te nie są przetwarzane przez parser w procesie walidacji, lecz umożliwiają umieszczanie opisowych notatek dotyczących schematu — bezpośrednio w tymże schemacie.

Użycie wspomnianych elementów jest bezpieczniejsze niż umieszczanie komentarzy, te ostatnie bowiem bywają usuwane ze schematów przez niektóre parsery. Elementom takie usuwanie nie grozi, a przy okazji powstaje mechanizm umożliwiający automatyczne generowanie dokumentacji.

Każdy niemal element schematu może posiadać element `xs:annotation` jako element potomny, ten z kolei może zagnieżdżać element `xs:documentation`. Tekst w ramach tego elementu — jako jego dane — powinien w sposób czytelny dla człowieka adekwatnie opisywać odnośny element, wraz z takimi szczegółami jak atrybuty (obowiązkowe i opcjonalne), typ danych i informację o powtarzalności.

cdn.

Wizualne tworzenie schematu za pomocą XMLSpy (ciąg dalszy)

Widok *Schema Overview* umożliwia wybór globalnie zdefiniowanych elementów i ich atrybutów z prostej listy. Program wyświetla atrybuty i ograniczenia wybranego elementu w pomocniczym oknie poniżej listy, ułatwiając definiowanie typów danych, narzucanie ograniczeń oraz nadawanie atrybutom wartości stałych lub domyślnych. W tabeli widocznej w górnej sekcji ekranu można w łatwy sposób wstawiać, dołączać i usuwać elementy globalne oraz typy złożone. Wybierając element lub typ, można zmieniać jego atrybuty w oknie pomocniczym.

W trybie *Content Model view* możliwe jest także edytowanie (w oknie pomocniczym) właściwości wybranych elementów; chociaż w danej chwili dostępny do takiej edycji jest tylko pojedynczy element, to jednak bardzo łatwo można się przemieszczać między poszczególnymi elementami, jak również przenosić i kopiować ich właściwości przy użyciu schowka.

Definiowanie typu złożonego, obarczonego ograniczeniami, jeżeli odbywa się bezpośrednio w kodzie źródłowym, jest o tyle utrudnione, że bardzo łatwo można pomylić się w definiowaniu owych ograniczeń. Wizualny edytor XMLSpy znacznie ułatwia to zadanie, przez wizualizację zastosowanych ograniczeń i propozycje dodawania kolejnych.

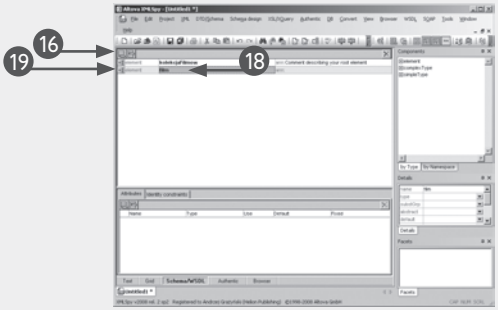
Sposób wyświetlania schematu jest wysoce konfigurowalny: użytkownik ma możliwość określenia, które z parametrów będą wyświetlane — te, których na pewno nie użyje, można z łatwością ukryć.

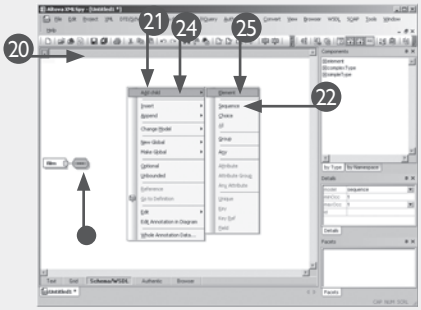
Całości dopełniają funkcje importowania definicji z innych przestrzeni nazw, globalnego redefiniowania komponentów schematu, generowanie schematów na podstawie DTD lub innych schematów oraz generowanie relacyjnych baz danych.

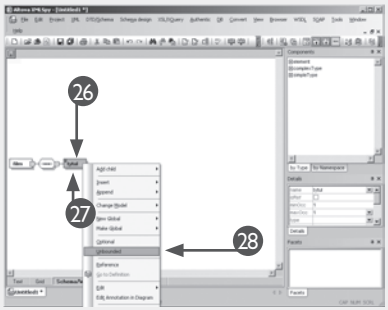
Wizualne tworzenie schematu za pomocą XMLSpy *(ciąg dalszy)*

16. W wyświetlanym dialogu *Design View* kliknij ikonę *Append*.
17. Kliknij opcję *Element*. Zostanie utworzony nowy element.
18. Wpisz nazwę elementu potomnego, który utworzyłeś w punkcie 12.
19. Kliknij ikonę towarzyszącą nowemu elementowi.

    Wyświetlony zostanie widok *Content Model View*.
20. Kliknij prawym przyciskiem myszy wyświetlone okno.
21. Wybierz opcję *Add Child*.
22. Wybierz opcję *Sequence*.
    - Pojawi się kompozytor sekwencji.
23. Kliknij prawym przyciskiem myszy wyświetlone okno.
24. Wybierz opcję *Add Child*.
25. Wybierz opcję *Element* w celu dodania nowego elementu.
26. Wprowadź nazwę dla utworzonego elementu potomnego.
27. Kliknij element prawym przyciskiem myszy.
28. Wybierz opcję *Unbounded*.







Widok zostanie zaktualizowany, wskazując, że liczba wystąpień elementu jest nieograniczona.

29 W oknie pomocniczym *Details entry* kliknij opcję *type* i wybierz z listy pozycję `xs:string`.

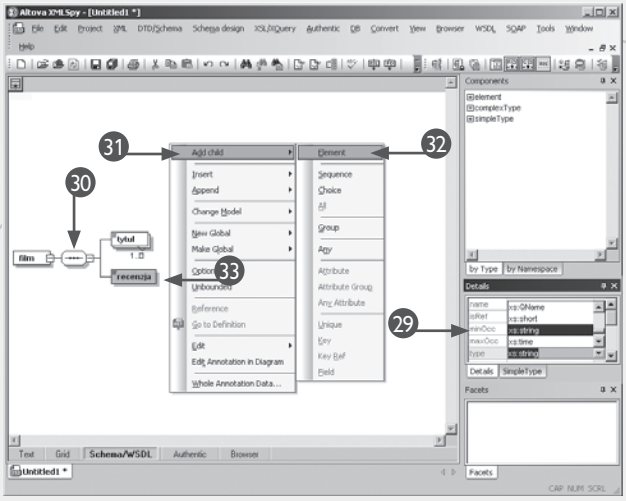
30 Kliknij prawym przyciskiem myszy okno kompozytora sekwencji.

31 Wybierz opcję *Add Child*.

32 Kliknij opcję *Element*.

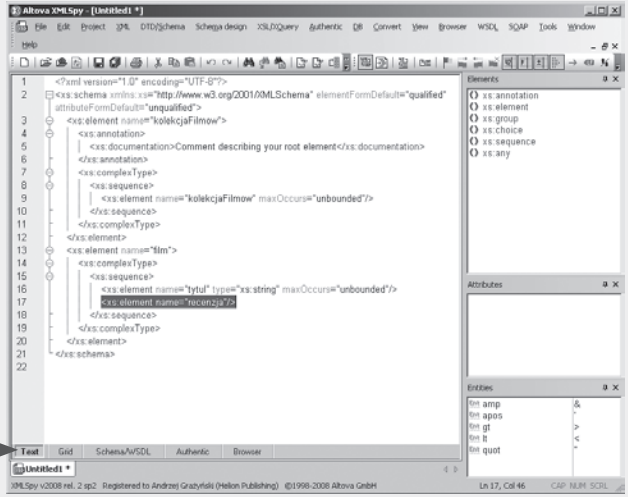
Zostanie utworzony nowy element.

33 Wprowadź nazwę dla utworzonego elementu potomnego.



34 Kliknij zakładkę *Text*.

Zostanie wyświetlony tekstowy widok schematu.



Wskazówka

Elementy `xs:annotation` i `xs:documentation`, automatycznie dodawane przez XMLSpy, nie tylko ułatwiają posługiwanie się schematami, lecz także wykorzystywane są do generowania szczegółowej dokumentacji na temat tych schematów. Dokumentacja taka zawsze jest przydatna jako taka, jednak szczególnie wartościowa okazuje się w przypadku publikowania schematów do użytku ogólnego.

Oferowany przez XMLSpy automatyczny generator dokumentacji, uruchamiany za pośrednictwem prostego okna dialogowego, umożliwia szczegółowe określenie listy elementów (globalnych i lokalnych) oraz ich grup, typów danych (prostych i złożonych) oraz atrybutów, na temat których utworzony zostanie szczegółowy opis wyjaśniający sposoby używania ich w tworzonym dokumencie docelowym. Dla każdej pozycji zażądać można wygenerowania diagramu graficznego, podobnego do tego wyświetlanego w widoku *Content View*, a także dołączenia związanego z daną pozycją fragmentu kodu źródłowego. Dokumentacja może być generowana zarówno w formacie HTML, jak i w formacie obsługiwanym przez Microsoft Word.