

XIV Konferencja PLOUG
Szczyrk
Październik 2008

XForms

Tomasz Traczyk
Politechnika Warszawska

e-mail: ttraczyk@ia.pw.edu.pl

Abstrakt. Referat przedstawia wciąż mało znany, lecz stale rozwijający się XML-owy standard opisu za-awansowanych formularzy, działających w przeglądarkach internetowych. Standard ten daje znacznie większe możliwości od powszechnie znanych formularzy HTML. Formularze XForms do swego działania nie wymagają programowania, a jedynie obecności odpowiedniego rozszerzenia przeglądarki.

Informacja o autorze. Dr inż. Tomasz Traczyk jest docentem w Instytucie Automatyki i Informatyki Stosowanej na Wydziale Elektroniki i Technik Informatycznych Politechniki Warszawskiej.

Formularze w aplikacjach internetowych

W ostatnich latach daje się zauważyć dominujący trend do zastępowania specjalizowanych środowisk do wykonywania aplikacji formularzowych przez technologie internetowe, wykorzystujące po stronie klienta standardową przeglądarkę internetową. Rozpowszechnieniu się aplikacji internetowych nie towarzyszy jednak większy rozwój technologii tworzenia formularzy. Większość najpopularniejszych technologii korzysta z możliwości języka HTML, wzbogaconych skryptami. Bardziej złożone technologie, jak AJAX, są jedynie obejściami uzupełniającymi braki HTML. Technologia apletów języka Java nie przyjęła się (do wyjątków należy tu Oracle Forms). Bardziej złożone funkcjonalnie aplikacje są często realizowane za pomocą rozwiązań firmowych, np. Adobe Flash; rozwiązania te wiążą jednak twórcę z technologią jednej firmy, a od użytkownika wymagają instalacji odpowiednich wtyczek.

Standardowe możliwości formularzy HTML zupełnie nie wystarczają do budowy typowych aplikacji transakcyjnych. Brakuje wielu niezbędnych cech, np.

- walidacji danych, zarówno co do ich typu jak i co zgodności z bardziej złożonymi *business rules*; porządniejsze sprawdzenie poprawności danych wymaga interakcji z serwerem, co jest niewygodne, czasochłonne i obciąża zarówno serwer jak i łącze sieciowe;
- możliwości dynamicznego zmieniania wyglądu, treści i zachowania formularzy (np. wyliczania wartości, uaktywniania pól w zależności od zawartości innych pól, dynamicznego kolorowania danych itp.);

Te braki próbuje się uzupełnić oprogramowując formularze w językach skryptowych (najczęściej w języku Java Script). Prowadzi to jednak do powstawania stron obszernych, o nieczytelnym kodzie, trudnych w utrzymaniu oraz często nie w pełni przenośnych między przeglądarkami.

Do tego dochodzą jeszcze problemy wynikające z bezstanowości interakcji w WWW:

- trudności z zachowaniem wartości wpisanych formularzach składających się na interakcje wielostronicowe: cofnięcie się do poprzednich stron (zwłaszcza za pomocą przycisku „Wstecz” przeglądarki) powoduje istotne problemy w wielu aplikacjach, nierzadko powodując wprowadzenie błędnych danych;
- ryzyko wielokrotnego wykonania operacji: ponieważ wiele akcji wymaga interakcji z serwerem, zaś prędkość odpowiedzi serwera jest ograniczona, niecierpliwi użytkownicy przyciskają ten sam przycisk wielokrotnie, co często powoduje wielokrotne zarejestrowanie przez serwer tego samego żądania;
- przeładowywanie całych stron po każdej interakcji z serwerem; w przypadku stron rozbudowanych czy zawierających dużo grafiki jest to czasochłonne, a zawsze jest irytujące dla użytkownika.

Także te trudności obchodzi się używając języków skryptowych – przykładem jest tu technologia AJAX. Jest to jednak technicznie niełatwe, a przecież problemy te występują powszechnie.

Odpowiedzią na wszystkie te problemy jest wbudowanie w przeglądarki takiego aparatu obsługującego formularze, który zapewniłby realizację wszystkich cech niezbędnych do tworzenia typowych formularzy. By użycie tego aparatu było łatwe i możliwie mało narażone na błędy, trzeba stworzyć deklaracyjny język opisu formularzy, obejmujący wszystkie funkcjonalności potrzebne w typowych aplikacjach transakcyjnych.

Propozycją takiego właśnie zaawansowanego deklaracyjnego języka opisu formularzy jest XForms. Jest to dialekt XML, stworzony przez W3C (*World Wide Web Consortium*), przeznaczony

ny do integracji z innymi językami internetowymi (np. XHTML, SVG), a w intencji mający zastąpić niedoskonałe formularze HTML.

W stosunku do HTML nowy język opisu formularzy oferuje liczne przewagi:

- oddzielenie modelu danych i samych danych od prezentacji;
- możliwość operowania na złożonych danych zapisanych w XML oraz przechowywanie aktualnego stanu tych danych po stronie klienta;
- łatwe realizowanie złożonych formularzy, w tym także wieloliniowych, o strukturze *master-detail* itp.;
- możliwość dynamicznego sterowania sposobem prezentacji: kolorami, wyświetlaniem i aktywnością pól itp.;
- rozbudowane możliwości walidacji danych, zarówno co do ich zgodności z zadeklarowanym typem danych jak i co do spełniania przez nie zadeklarowanych reguł (*business rules*);
- możliwość określenia sposobu reagowania na zdarzenia bez potrzeby dodatkowego programowania;
- możliwość wykonywania obliczeń i wyświetlania ich rezultatów oraz wstawiania tych rezultatów do wynikowych danych.

Wprowadzenie do XForms

Tworząc nowy język opisu formularzy kierowano się następującymi założeniami:

- język powinien używać XML-a do opisu formularza i do zapisu danych, zarówno dostarczanych do formularza jak i wysyłanych przez formularz;
- dane powinny być oddzielone od określenia sposobu ich prezentacji;
- wypełniony formularz powinien dawać się łatwo przesyłać między programami i użytkownikami, by ułatwić budowanie rozwiązań typu *workflow*;
- możliwości języka powinny być na tyle duże, by w znaczącej większości przypadków nie trzeba było ich uzupełniać programowaniem w innych językach;
- język powinien zapewniać typowe funkcjonalności potrzebne na formularzach, jak walidacja czy obliczenia;
- formularze powinny działać nie tylko w typowych przeglądarkach HTML, ale także na urządzeniach przenośnych, głosowych itp.;
- wszystkie możliwości formularzy HTML powinny być zachowane, choć nie zakłada się syntaktycznej zgodności języków.

By opis formularzy był łatwy w interpretacji i odporny na błędy programowania, język jest w możliwie dużym stopniu deklaratywny. Zawiera jednak możliwości obsługi zdarzeń i wykonywania pewnych akcji.

Opis formularza w XForms zanurza się w języku macierzystym opisującym stronę. Takim językiem najczęściej jest XHTML, ale może być to dowolny dialekt XML, np. SVG. Do odróżnienia elementów XForms stosuje się przestrzenie nazw, co pokazano w poniższym przykładzie.

```
<html xmlns = "http://www.w3.org/1999/xhtml"
      xmlns:xf = "http://www.w3.org/2002/xforms"
      xmlns:xs = "http://www.w3.org/2001/XMLSchema">
<head>
```

```

<title>Prosty przykład</title>
<style>xf\:input xf\:label {width: 8ex}</style>
<xf:model id="model" xmlns="">
  <xf:instance id="dane">
    <dane>
      <napis>tekst</napis>
      <liczba>7</liczba>
    </dane>
  </xf:instance>
  <xf:instance id="temp">
    <temp>
      <dlugosc/>
    </temp>
  </xf:instance>

  <xf:bind nodeset="liczba" type="xs:integer"/>
  <xf:bind id="i_dl" nodeset="instance('temp')/dlugosc"
    calculate="string-length(instance('dane')/napis)"/>
  <xf:submission id="zapis" action="file:w.xml" method="put"
replace="none"/>
</xf:model>
</head>
<body>
  <h3>Prosty przykład</h3>
  <xf:input ref="napis"><xf:label>Napis:</xf:label></xf:input>
  &nbsp;
  <xf:output bind="i_dl"><xf:label>długość:&nbsp;</xf:label></xf:output> zn.
  <br/>
  <xf:input ref="liczba"><xf:label>Liczba:</xf:label></xf:input>
  <br/><hr/>
  <xf:submit submission="zapis"><xf:label>Submit</xf:label></xf:submit>
</body>
</html>

```

Wynik działania tego przykładu przedstawia rysunek 1.

Prosty przykład

Napis:	<input type="text" value="tekst"/>	długość: 5 zn.
Liczba:	<input type="text" value="7"/>	
<input type="button" value="Submit"/>		

Rys. 1. Wynik działania prostego formularza

Po naciśnięciu przycisku *Submit* wprowadzone dane są zapisywane w postaci dokumentu XML:

```

<?xml version="1.0"?>
<dane>
  <napis>tekst</napis>
  <liczba>7</liczba>
</dane>

```

Model danych

Model danych w XForms jest oddzielony od opisu samego formularza i wyrażony jest z XML, z użyciem przestrzeni nazw. Każdy formularz XForms musi korzystać z przynajmniej jednego takiego modelu (może też z wielu). Model danych może być zapisany w dokumencie definiującym formularz (zwykle w części <head>) lub w dokumencie zewnętrznym. Model zawiera strukturę danych zapisaną w XML (może ona być wypełniona danymi początkowymi), specyfikacje przypisań, nadające poszczególnym danym identyfikatory, wartości początkowe i wyliczane, typy danych i inne ograniczenia oraz określenie przeznaczenia danych wynikowych.

W powyższym prostym przykładzie model zawiera dwie tzw. instancje danych. Pierwsza – domyślna – jest użyta do zdefiniowania „prawdziwych” danych, które będą zapisane w wyniku operacji `submit`; druga zawiera dane tymczasowe. Pierwsza z instrukcji `bind` przypisuje typ danych (pochodzi on ze specyfikacji XML Schema); zgodność wpisanych danych z zadeklarowanym typem będzie sprawdzana przez formularz i w przypadku błędu dane nie dadzą się zapisać. Druga instrukcja `bind` definiuje sposób wyliczania wartości danej pomocniczej oraz dodatkowy identyfikator tej danej. Instrukcja `submit` określa sposób wysłania danych wyjściowych; w tym wypadku skorzystano z nowej możliwości, pozwalającej dane wyjściowe zapisać w lokalnym pliku XML; przydaje się to zwłaszcza w czasie testowania formularzy.

Układ formularza

Układ formularza zapisuje się za pomocą języka macierzystego, w którym zanurzone są znaczniki XForms. Znaczniki XForms definiują poszczególne pola formularza oraz sposób ich powiązania z modelem danych. Dostępne rodzaje pól są dość podobne do znanych z HTML; przykłady pokazano na rysunku 2.

The image shows a screenshot of an XForms form with several input fields and a submit button. The fields are arranged in a grid-like structure:

- input**: A text input field containing the word "tekst".
- secret**: A password input field with masked characters ".....".
- input (data)**: A date picker showing "2008-08-11". A calendar popup is visible, showing the month of August 2008 with the current date highlighted.
- range**: A slider control with the label "output 30%".
- textarea**: A text area input field containing the word "tekst".
- select1 (full)**: A radio button group with options "Pierwszy" (selected) and "Drugi".
- select1 (minimal)**: A dropdown menu showing "Pierwszy".
- select**: A dropdown menu showing "Jeden Trzy".
- Submit**: A button at the bottom left of the form.

Rys. 2. Wybrane rodzaje pól w XForms

Pola zachowywać się mogą różnie w zależności od zadeklarowanego typu danych; na powyższym rysunku przedstawiono np. możliwe zachowanie pola `input` dla danych typu `data`. Niektóre pola mają także kilka odmian wizualnych, np. listy mogą pojawiać się w postaci pełnej, skróconej lub minimalnej. Nowością jest pole typu `output`, które służy do wyświetlania wartości danych (lub wyniku obliczeń) i może być dowolnie wplatanie *in-line* w treść dokumentu. Nowe jest także wymaganie, by każde z pól (oprócz pól `output`) musiało mieć jawnie zdefiniowaną przypisaną sobie etykietę.

Poszczególne elementy i atrybuty danych mogą być adresowane w instrukcjach XForms na dwa sposoby: przez podanie (zwykle w atrybucie `ref` lub `nodeset`) ścieżki XPath odnajdującej określone dane w modelu danych albo przez podanie wartości identyfikatora (w atrybucie `bind`).

W podanym wcześniej prostym przykładzie użyto dwóch pól typu `input` do wprowadzania danych oraz pola typu `output` do wyświetlenia danej wyliczanej. Przycisk `submit` służy do wywołania operacji `submission` zdefiniowanej w modelu danych.

Pola formularza można łączyć w grupy. Grupa może mieć tworzyć kontekst w stosunku do modelu danych, co upraszcza pisanie odwołań XPath w poszczególnych polach.

Jeśli w modelu danych pewne elementy powtarzają się, to przypisane do nich pola także mogą być powtórzone; do budowania takich formularzy służy specjalna konstrukcja `repeat`.

Możliwości XForms

Pobieranie i wysyłanie danych

Dane prezentowane przez formularz są umieszczone w strukturze danych określonej przez model. Mogą one być po prostu literalnie zapisane w definicji modelu: w dokumencie zawierającym definicję formularza lub w osobnym dokumencie wskazanym przez adres URI. Mogą także być wyliczone przez formularz, zwykle na podstawie innych danych. Możliwe jest zarówno wyliczenie wartości początkowych, jak i dynamiczne wyliczanie wartości po każdej zmianie danych w formularzu. Dane można także załadować z podanej lokalizacji za pomocą wywołania akcji `load` albo z pliku lokalnego, wybranego w wyniku wypełnienia pola typu `upload`. Można także przywrócić pierwotny stan danych (z chwili po załadowaniu formularza) za pomocą akcji `reset`.

Formularz może zawierać kilka modeli danych, a każdy z modeli – kilka instancji danych. Dane dostępne w formularzu mogą pochodzić z kilku źródeł; podobnie dane wynikowe mogą być wysyłane w kilka miejsc.

Operacja wysyłania danych (*submission*) może wykorzystywać różne metody protokołu HTTP: *get*, *post*, a także *put*, możliwy jest też zapis do lokalnych plików oraz wysyłka danych pocztą elektroniczną. Domyślnie dane wysyłane są w formacie XML wynikającym ze struktury zdefiniowanej w modelu danych. Można jednak także zażądać zmiany na inne formaty, w tym także zgodne z postacią, w jakiej dane wysyłają formularze HTML – oznacza to, że formularze XForms mogą współpracować z rozwiązaniami serwerowymi napisanymi dla formularzy HTML (np. programami typu CGI).

W HTML strona odesłana przez serwer jako odpowiedź na wysłanie danych zawsze jest wyświetlana i zastępuje formularz wysyłający dane. W XForms możemy sterować sposobem traktowania takiej odpowiedzi z serwera: może ona zastąpić formularz (zapewne będzie to wówczas strona w HTML z komunikatem na temat powodzenia lub niepowodzenia wysyłki), może być wczytana do struktury danych formularza, zastępując wysłaną instancję danych (wtedy powinny to być oczywiście dane w XML, np. przekształcone przez serwer) lub może po prostu zostać zignorowana.

Walidacja danych

Formularze XForms zapewniają kilka rodzajów walidacji danych. Element danych może być oznaczony jako obowiązkowy – nie akceptujący wartości pustej, może też mieć określony typ danych (w sensie prostego typu XML Schema). Procesor XForms automatycznie sprawdza zgodność wprowadzonych danych z zadeklarowanym typem; od tego typu może także zależeć zachowanie niektórych pól. Cały model może być przypisany do schematu XML. W takim wypadku procesor stale sprawdza zgodność struktury danych ze schematem, uwzględniając zarówno wbudowane typy danych, jak i typy zdefiniowane przez użytkownika, np. zawierające dodatkowe ograniczenia, jak zgodność z wyrażeniem regularnym. Wreszcie do każdego z elementów modelu (zarówno do „liści” w strukturze jak i do elementów strukturalnych) można przypisać wyrażenie logiczne (zapisane w języku XPath), sprawdzające spełnianie określonej reguły przez daną strukturę. W ten sposób można np. wymusić pewne relacje między wartościami różnych danych lub ograniczyć liczbę wystąpień pewnych danych powtarzających się.

Obliczenia

Na formularzach można prowadzić obliczenia, zarówno dotyczące wartości początkowych, jak i wartości bieżących. Wyliczane wartości mogą być jedynie wyświetlane lub też mogą być podstawiane do struktury danych. Wyliczenia mogą być automatyczne, w reakcji na zmianę danych, albo wymuszone przez zdarzenia. Obliczane wyrażenie zapisuje się w języku XPath, korzystając

z funkcji tego języka, uzupełnionych przez dodatkowe funkcje XForms, takie jak obliczenie sumy, średniej, najmniejszej i największej wartości, policzenie liczby niepustych elementów, podanie aktualnej daty i czasu itd.

Dynamiczne kształtowanie prezentacji

Język XForms oferuje znaczne możliwości dynamicznego zmieniania cech wizualnych i zawartości formularzy. Możliwe jest włączanie i wyłączanie widoczności poszczególnych pól oraz całych fragmentów formularza. O widoczności pola, jego aktywności czy możliwości zmiany wartości decydują atrybuty, które mogą być wyrażeniami – wartości takich wyrażen obliczane są na bieżąco i dynamicznie decydują o wyglądzie i zachowaniu formularza. Dzięki temu łatwo jest zbudować takie formularze, które same odpowiednio się dostosowują do wcześniej wprowadzonych danych.

Listy wartości mogą być dynamicznie wypełniane na podstawie danych zawartych w modelu. Wyświetlane opisy mogą być inne od podstawianych wartości danych, można więc w ten sposób budować listy wartości o funkcjonalności podobnej do podstawiania wartości klucza obcego na podstawie jego deskryptora.

Wygląd pól określa się w definicji formularza dość ogólnie, pozostawiając szczegóły przeglądarce oraz ewentualnemu formatowaniu za pomocą CSS. Pełne możliwości formatowania formularzy XForms zapewni CSS3, niestety jeszcze nie wspierany przez typowe przeglądarki. Korzystając z CSS2 można jednak nadać formularzom estetyczny wygląd, posługując się specjalnymi pseudoklasami i pseudoelementami.

Zdarzenia i akcje

XForms korzysta z modelu zdarzeń zdefiniowanego w propozycji standardu XML Events [W3C03], uzupełniając zdarzenia związane z modelem DOM o dodatkowe zdarzenia charakterystyczne dla formularzy, np. po załadowaniu formularza, po zmianie danych, przed i po wysłaniu danych z formularza itp.

Zdarzenia mogą wywoływać akcje. W XForms zdefiniowano wiele akcji, realizujących takie funkcje jak: ładowanie danych do formularza, wysłanie formularza lub jego *reset*, dodawanie i usuwanie rekordów oraz ustawienie bieżącej linii w wielorekordowych częściach formularza, ustawienie kursora w określonym polu, wyświetlanie komunikatów, wymuszenie walidacji, wyliczeń lub odświeżenia formularza.

Akcje mogą być za pomocą instrukcji `action` grupowane w sekwencje przypisywane do jednego zdarzenia.

Za pomocą XForms może także porządnie zorganizować komunikację z operatorem. Dostępne są akcje służące do wyświetlania komunikatów. Komunikaty można wywoływać w reakcji na zdarzenia, można też je np. przypisać do błędów walidacji. Za pomocą kombinacji XForms i CSS można dynamicznie wyróżnić wizualnie dane błędne lub mające określone cechy. Każdemu polu można deklaratywnie przypisać podpowiedź oraz pełny tekst pomocy. Teksty te mogą zawierać formatowanie oraz być pobierane z dokumentów zewnętrznych; umożliwia to np. łatwe wykonanie wielu wersji językowych.

Przykład – formularz *master-detail*

Niżej opisany przykład pokazuje, jak można za pomocą XForms uzyskać funkcjonalność podobną do typowych formularzy stosowanych w aplikacjach z bazami danych. Rysunek 3 przedstawia wygląd tego formularza. Formularz ma służyć do wprowadzania przez studentów deklaracji zapisów na przedmioty. Ma on typową strukturę *master-detail*, odpowiednią dla wielu zastosowań.

Deklarowane przedmioty muszą być wybrane z listy przedmiotów dostępnych, wybiera się także jeden z dwóch sposobów zaliczania.

Zapisy

Nr albumu:	<input type="text" value="123456"/>
Nazwisko:	<input type="text" value="Abacki"/>
Imiona:	<input type="text" value="Alfred"/>

Id przedmiotu:	<input type="text" value="BD2 - Bazy danych 2"/>	Zaliczanie:	<input type="text" value="E-egzamin"/>	doc. dr inż. Tomasz Traczyk
Id przedmiotu:	<input type="text" value="TUL - Teoria układów logicznych"/>	Zaliczanie:	<input type="text" value="B-zaliczenie"/>	ECTS: 3 Teoria układów logicznych
Id przedmiotu:	<input type="text" value="BD1 - Bazy danych 1"/>	Zaliczanie:	<input type="text" value="B-zaliczenie"/>	ECTS: 2 Bazy danych 1

Rys. 3. Formularz *master-detail*

Model danych

Formularz korzysta z dwóch źródeł danych, będących dokumentami XML o strukturze przedstawionej w tabeli. Dane studenta powinny być dynamicznie wygenerowane przez serwer, dane dostępnych przedmiotów mogą także być dynamiczne albo mogą być udostępniane z pliku.

Tabela 1. Źródła danych dla formularza *master-detail*

dane.xml	przedmioty.xml
<pre><dane> <student nr_albumu="123456"> <imiona>Alfred</imiona> <nazwisko>Abacki</nazwisko> </student> <zapisy> <zapis id_predmiotu=""> <typ_zaliczania/> </zapis> </zapisy> </dane></pre>	<pre><przedmioty> <przedmiot id_predmiotu="BD1" ects="2"> <nazwa>Bazy danych 1</nazwa> <prowadzacy> mgr inż. Kazimierz Kowalski </prowadzacy> </przedmiot> ... </przedmioty></pre>
Dane studenta oraz szablon struktury do wpisywania deklaracji zapisów.	Dane przedmiotów

Dane te pobierane są do formularza przez umieszczenie w modelu danych definicji dwóch instancji danych:

```
<xf:instance id="dta" src="dane.xml"/>
<xf:instance id="dct" src="przedmioty.xml"/>
```

Dane studenta zostały oznaczone jako niemodyfikowalne, np.:

```
<xf:bind nodeset="student/@nr_albumu" readonly="true()"/>
```

Układ formularza

Część *master* formularza została zbudowana z pól typu `input`, podobnie jak w poprzednim przykładzie. Natomiast do budowy części *detail* trzeba było użyć specjalnej konstrukcji `repeat`:

```
<xf:repeat id="deta1" nodeset="zapisy/zapis">
  <xf:select1 ref="@id_predmiotu">
    <xf:label>Id przedmiotu:</xf:label>
```

```

<xf:itemset nodeset="instance('dct')//przedmiot">
  <xf:label ref="concat(@id_przedmiotu, ' - ', nazwa)"/>
  <xf:value ref="@id_przedmiotu"/>
</xf:itemset>
<xf:rebuild ev:event="xforms-value-changed"/>
</xf:select1>
&nbsp;
<xf:select1 ref="typ_zaliczania">
  <xf:label>Zaliczanie:</xf:label>
  <xf:item><xf:label>E-egzamin</xf:label><xf:value>E</xf:value></xf:item>
  <xf:item><xf:label>B-zaliczenie</xf:label><xf:value>B</xf:value></xf:item>
</xf:select1>
<xf:switch id="przesuw">
  <xf:case id="lewy">
    &nbsp;
    <xf:output ref="instance('dct')//przedmiot[@id_przedmiotu =
      current()//@id_przedmiotu]/@ects">
      <xf:label>ECTS:</xf:label>
    </xf:output>
    &nbsp;
    <xf:output ref="instance('dct')//przedmiot[@id_przedmiotu =
      current()//@id_przedmiotu]/nazwa"/>
  </xf:case>
  <xf:case id="prawy">
    &nbsp;
    <xf:output ref="instance('dct')//przedmiot[@id_przedmiotu =
      current()//@id_przedmiotu]/prowadzacy"/>
  </xf:case>
</xf:switch>
</xf:repeat>

```

Instrukcja `repeat` powoduje powtórzenie odpowiedniego zbioru pól tyle razy, ile jest wystąpień elementu `<zapis>` w strukturze danych. Przyciski *Dodaj* i *Usuń* wywołują specjalne akcje, związane ze sterowaniem powtarzającymi się danymi:

```

<xf:trigger>
  <xf:label>Dodaj</xf:label>
  <xf:insert nodeset="//zapisy/zapis" ev:event="DOMActivate"/>
</xf:trigger>
<xf:trigger>
  <xf:label>Usuń</xf:label>
  <xf:delete nodeset="//zapisy/zapis" at="index('detal')"
    ev:event="DOMActivate"/>
</xf:trigger>

```

Bieżący wiersz w części *detail* wyróżniono, definiując w arkuszu stylu sposób formatowania specjalnej pseudoklasy:

```
.repeat-index { background-color: silver; }
```

W części *detail* formularza zastosowano dwie listy rozwijane: jedną statyczną, z predefiniowanymi opcjami, drugą zaś dynamiczną, za pomocą instrukcji `itemset` wypełnianą danymi z instancji zawierającej informacje odczytane z dokumentu `przedmioty.xml`. Opisy pojawiające się na tej liście zawierają nazwy przedmiotów (deskryptory), zaś wartości zwracane – jedynie identyfikatory.

Pola pokazujące liczbę punktów ECTS, nazwę przedmiotu i prowadzącego są wyliczane na bieżąco; że zaś trzeba je odświeżyć po zmianie przedmiotu, wywołano akcję `rebuild`, przypisując ją do zdarzenia zmiany danych w polu identyfikatora przedmiotu.

Ponieważ wszystkie dane części *detail* mogłyby nie mieścić się na szerokości ekranu, zastosowano przełącznik `switch`, który powoduje, iż w wybranym wierszu pokazuje się jedna z dwóch grup danych: punkty ECTS i nazwa przedmiotu albo dane prowadzącego zajęcia. Przyciski zmie-

niające widoczną grupę wywołują akcję `toggle`, dedykowaną do sterowania wyświetlaniem jednej z części przełącznika `switch`, np.:

```
<xf:trigger>
  <xf:label>&lt;</xf:label>
  <xf:toggle ev:event="DOMActivate" case="lewy"/>
</xf:trigger>
```

W podobny sposób można łatwo budować formularze wielostronicowe.

Specyfikacje i implementacje

Standard XForms, podobnie jak inne standardy XML-owe, jest tworzony przez *World Wide Web Consortium* (W3C). Obecnie obowiązuje wersja 1.0 standardu [W3C07a], przygotowywana jest wersja 1.1 [W3C07b].

Jak to zwykle bywa w przypadku nowych standardów XML-owych, barierą dla ich rozpowszechnienia się jest brak obsługi w popularnych przeglądarkach. Zwykle wbudowanie odpowiednich rozwiązań do przeglądarek poprzedzone jest pojawieniem się odpowiednich rozszerzeń (*plug-ins*). Tak też jest i w przypadku XForms. Istnieje wiele rozszerzeń do różnych przeglądarek (patrz np. [FFac], [FPlay], [MozXF]), realizujących obszerne fragmenty specyfikacji XForms. Niektóre z tych rozwiązań (np. FormFaces [FFac]) nie wymagają instalacji żadnego oprogramowania po stronie przeglądarki, pobierając niezbędny kod na bieżąco. Obiecujące są także implementacje na urządzenia przenośne¹.

Istnieją także narzędzia wspierające projektowanie formularzy XForms, np. IBM XML Forms Generator, IBM Visual XForms Designer czy popularny edytor XMLSpy firmy Altova.

XForms ma stać się obowiązkowym modulem przygotowywanego standardu XHTML 2.0, zastępując dotychczasowe formularze HTML. Dopiero wejście w życie tego standardu spowoduje zapewne powszechne wsparcie dla XForms w przeglądarkach.

Podsumowanie

Standard XForms oferuje projektantom formularzy niemal wszystkie środki potrzebne do budowy zaawansowanych aplikacji. Jak się wydaje, w ogromnej większości typowych aplikacji, w tym i systemów z bazami danych, jego możliwości są wystarczające. Gdyby powstały stabilne i dostępne implementacje obejmujące całość standardu, niemal zupełnie znikłaby potrzeba stosowania języków skryptowych w aplikacjach internetowych. Włączenie XForms do projektu standardu XHTML 2.0 daje nadzieję na pojawienie się takich implementacji.

Projektowanie formularzy XForms jest stosunkowo proste dla osób, które miały do czynienia z projektowaniem aplikacji z bazami danych oraz znają podstawowe technologie XML-owe. Główne trudności sprowadzają się do „walki” z niedoskonałościami istniejących implementacji standardu.

Współdziałanie formularzy XForms z bazami danych nie powinno nastęczać większych trudności. Obecnie we wszystkich liczących się bazach danych istnieją wbudowane narzędzia pozwalające generować dane w formacie XML, odpowiednim do dostarczania danych do formularzy XForms. Podobnie, istnieją także narzędzia do składowania i przetwarzania danych dostarczonych w języku XML.

¹ Wśród nich jest także próbna implementacja firmy Oracle, wchodząca w skład OracleAS Wireless Client 10g R2 Preview Release

Sam standard XForms należy ocenić bardzo wysoko. Jego upowszechnienie może znacząco uprościć budowanie systemów informacyjnych korzystających z technologii internetowych. Obecny stan implementacji nie zasługuje niestety na równie wysoką ocenę, można jednak mieć nadzieję, że wkrótce ulegnie to poprawie.

Bibliografia

- [Dub03] Dubinko M.: XForms Essentials, O'Reilly, 2003.
- [FFac] FormFaces Web Site. <http://www.formfaces.com>
- [FPlay] FormsPlayer Web Site. <http://www.formsplayer.com>
- [MozXF] Mozilla XForms Project Web Site. <http://developer.mozilla.org/pl/docs/XForms>
- [W3C03] XML Events. An Events Syntax for XML. W3C Recommendation, October 2003. <http://www.w3.org/TR/xml-events>
- [W3C07a] XForms 1.0 (Third Edition) W3C Recommendation, October 2007. <http://www.w3.org/TR/2007/REC-xforms-20071029>
- [W3C07b] XForms 1.1 W3C Candidate Recommendation, November 2007. <http://www.w3.org/TR/xforms11>
- [W3S] W3Schools XForms Tutorial. <http://www.w3schools.com/xforms/default.asp>
- [Wiki] Wikibooks XForms Tutorial and Cookbook. <http://en.wikibooks.org/wiki/XForms>
- [XFInst] XForms Institute Web Site. <http://xformsinstitute.com>