

HTML5 i CSS3

Standardy przyszłości

**Kompletny przewodnik po nowej, rewolucyjnej specyfikacji
najstynniejszej technologii internetowej!**

- Opanuj nowe atrybuty, selektory, funkcje formularzy i znaczniki strukturalne
- Zobacz, jak osadzać grafikę, pliki audio i wideo bez użycia Flasha
- Poznaj niezbędne mechanizmy HTML5 i twórz przyjazne aplikacje działające po stronie klienta



Brian P. Hogan

» Idź do

- Spis treści
- Przykładowy rozdział
- Skorowidz

» Katalog książek

- Katalog online
- Zamów drukowany katalog

» Twój koszyk

- Dodaj do koszyka

» Cennik i informacje

- Zamów informacje o nowościach
- Zamów cennik

» Czytelnia

- Fragmenty książek online

» Kontakt

Helion SA
ul. Kościuszki 1c
44-100 Gliwice
tel. 32 230 98 63
e-mail: helion@helion.pl
© Helion 1991–2011

HTML5 i CSS3. Standardy przyszłości

Autor: Brian P. Hogan
Tłumaczenie: Daniel Kaczmarek
ISBN: 978-83-246-3028-8
Tytuł oryginału: [HTML5 and CSS3: Develop with Tomorrow's Standards Today](#)
Format: 158×235, stron: 304



Kompletny przewodnik po nowej, rewolucyjnej specyfikacji najżywniejszej technologii internetowej!

- Opanuj nowe atrybuty, selektory, funkcje formularzy i znaczniki strukturalne
- Zobacz, jak osadzać grafikę, pliki audio i wideo bez użycia Flasha
- Poznaj niezbędne mechanizmy HTML5 i twórz przyjazne aplikacje, działające po stronie klienta

Stało się! Choć jeszcze niedawno HTML5 i CSS3 wydawały się technologiami wciąż odległej przyszłości, ich specyfikacje już zostały zaimplementowane w takich przeglądarkach, jak Google Chrome, Safari, Firefox czy Opera. Standardy HTML5 i CSS3, mimo że jeszcze w stadium rozwoju, nie przestają wzbudzać zachwyty administratorów sieci i twórców stron WWW, a ich opublikowanie hucznie zapowiedziało nową generację aplikacji internetowych. Teraz można je łatwiej wdrażać i utrzymywać, a także wyjść naprzeciw potrzebom użytkowników. Mało? Język HTML5 wprowadza także nowe elementy służące do definiowania struktury witryny oraz osadzania w niej treści, a CSS3 udostępnia zaawansowane selektory, rozszerzenia graficzne oraz zapewnia lepszą obsługę czcionek. W świecie internetu, gdzie pod względem rozprzestrzeniania się nowości trzy miesiące to cała epoka, już jutro ta technologia będzie obecna wszędzie. Nie trzeba więc nikogo przekonywać, że aby ten postęp bez zadyszki dogonić, trzeba już dziś... wziąć się za lekturę!

Przed Tobą doskonały podręcznik, w którym przedstawione zostały wszystkie dostępne sposoby korzystania z nowych możliwości standardów HTML5 i CSS3, także tych, które nie są jeszcze obsługiwane przez część przeglądarek. Każdy rozdział skupia się na określonej grupie problemów i zawiera listę niezbędnych znaczników, funkcji lub mechanizmów. Na początek dokładnie poznasz standardy HTML5 i CSS3, nowe atrybuty i znaczniki strukturalne. Przeczytasz co nieco na temat formularzy i odkryjesz, jak wykorzystać niektóre nowe pola i funkcje. Poznasz rewolucje wprowadzone w CSS3 – nowe selektory, cienie, gradienty, transformacje i sposoby korzystania z czcionek. Dowiesz się, jak HTML5 obsługuje dane audio i wideo oraz nauczysz się używać kanw do rysowania rozmaitych kształtów. A potem stworzysz przykładowe aplikacje działające po stronie klienta, a także zobaczysz, jak HTML5 umożliwia przesyłanie komunikatów i danych między domenami.

- Wprowadzenie do HTML5 i CSS3
- Nowe atrybuty i znaczniki strukturalne
- Tworzenie przyjaznych formularzy internetowych
- Tworzenie lepszych interfejsów użytkownika z użyciem CSS3
- Zwiększanie dostępności witryn internetowych
- Rysowanie na kanwach
- Osadzanie danych audio, wideo oraz grafiki wektorowej
- Wykorzystanie cieni, gradientów, transformacji i czcionek
- Przetwarzanie danych po stronie klienta
- Korzystanie z interfejsów API

Jesteś gotowy? Doskonale! Zacznij już dziś poznawać standardy przyszłości!

Spis treści

Przedmowa	11
Rozdział 1. Ogólne informacje na temat HTML5 i CSS3	17
1.1. Platforma tworzenia aplikacji internetowych	17
1.2. Zgodność wsteczna	21
1.3. Droga do przyszłości jest wyboista	23
 Część I Ulepszanie interfejsu użytkownika	
Rozdział 2. Nowe atrybuty i znaczniki strukturalne	31
Porada 1. Zmiana definicji bloga przy użyciu znaczników semantycznych	34
Porada 2. Tworzenie okien wywoływanych z atrybutami danych użytkownika	47
Rozdział 3. Tworzenie przyjaznych formularzy internetowych	53
Porada 3. Opisywanie danych za pomocą nowych pól wejściowych	56
Porada 4. Przechodzenie do pierwszego pola formularza przy użyciu atrybutu autofocus	65
Porada 5. Wyświetlanie wskazówek w postaci tekstu zastępczego	67
Porada 6. Edytowanie danych bezpośrednio na stronie przy użyciu atrybutu contenteditable	74

Rozdział 4. Tworzenie lepszych interfejsów użytkownika z użyciem CSS3	81
Porada 7. Definiowanie stylu tabeli przy użyciu pseudoklas	83
Porada 8. Umieszczanie docelowych adresów łączy przy użyciu :after	93
Porada 9. Tworzenie stron z wieloma kolumnami tekstu	97
Porada 10. Tworzenie interfejsów aplikacji na urządzenia przenośne przy użyciu zapytań o media	103
Rozdział 5. Zwiększanie dostępności	107
Porada 11. Definiowanie wskazówek nawigacji za pomocą ról ARIA	109
Porada 12. Tworzenie obszaru o uaktualnianej zawartości dostępnego dla programów odczytujących zawartość ekranu	115
 Część II Nowe widoki i dźwięki	
Rozdział 6. Rysowanie na kanwach	123
Porada 13. Rysowanie logo	125
Porada 14. Tworzenie wykresów statystyk przy użyciu biblioteki RGraph	132
Rozdział 7. Osadzanie danych audio i wideo	141
7.1. Kilka słów o historii	143
7.2. Kontenery i kodeki	145
Porada 15. Udostępnianie plików audio	150
Porada 16. Osadzanie plików wideo	154
Rozdział 8. Radość dla oczu	163
Porada 17. Zaokrąglanie rogów	165
Porada 18. Wykorzystanie cieni, gradientów i transformacji	173
Porada 19. Wykorzystanie własnych czcionek	184
 Część III Wychodzimy poza HTML5	
Rozdział 9. Przetwarzanie danych po stronie klienta	193
Porada 20. Zapisywanie preferencji przy użyciu obiektu localStorage	197
Porada 21. Przechowywanie danych w relacyjnej bazie danych po stronie klienta	204
Porada 22. Praca w trybie offline	217

Rozdział 10. Korzystanie z innych interfejsów API	221
Porada 23. Utrzymywanie historii	223
Porada 24. Komunikowanie się między domenami	226
Porada 25. Komunikacja przy użyciu gniazdek internetowych	233
Porada 26. Sprawdzanie własnej pozycji. Geolokalizacja	241
Rozdział 11. Co będzie dalej?	247
11.1. Transformacje CSS3	249
11.2. Wątki robocze	253
11.3. Wbudowana obsługa mechanizmu „przeciągnij i upuść”	255
11.4. WebGL	262
11.5. Indexed Database API	263
11.6. Weryfikacja formularza po stronie klienta	264
11.7. Cała naprzód!	266
 Dodatki	
Dodatek A Skrócony przegląd funkcji	269
A.1. Nowe elementy	269
A.2. Atrybuty	270
A.3. Formularze	270
A.4. Atrybuty pól formularzy	271
A.5. Dostępność	272
A.6. Multimedia	272
A.7. CSS3	273
A.8. Przechowywanie danych po stronie klienta	275
A.9. Dodatkowe API	276
Dodatek B Podstawowe informacje na temat biblioteki jQuery ..	277
B.1. Ładowanie biblioteki jQuery	278
B.2. Podstawy biblioteki jQuery	278
B.3. Metody, które służą do zmieniania zawartości	279
B.4. Tworzenie elementów	282
B.5. Zdarzenia	282
B.6. Funkcja document.ready	283

Dodatek C Kodowanie danych audio i wideo	285
C.1. Kodowanie danych audio	285
C.2. Kodowanie danych wideo dla sieci WWW	286
Dodatek D Zasoby	289
D.1. Zasoby dostępne w internecie	289
Dodatek E Bibliografia	291
Skorowidz	293

Rozdział 3.

Tworzenie przyjaznych formularzy internetowych

Jeżeli kiedykolwiek tworzyłeś skomplikowany interfejs użytkownika, na pewno wiesz, jak bardzo ograniczone jest działanie podstawowych kontrolki formularzy HTML. Programista jest ograniczony jedynie do pól tekstowych, menu wyboru, przycisków opcji, pól wyboru i czasami korzysta też z jeszcze bardziej prymitywnych **list wielokrotnego wyboru**, których sposób wykorzystania trzeba użytkownikom nieustannie tłumaczyć („Naciśnij klawisz *Ctrl* i klikaj myszą pozycje, które chcesz zaznaczyć, a na komputerze Mac zamiast klawisza *Ctrl* trzymaj wciśnięty klawisz *Cmd*”).

W takiej sytuacji robi się to, co robią wszyscy dobrzy programiści — korzysta się z biblioteki Prototype albo jQuery bądź też implementuje się własne kontrolki i funkcje przy użyciu HTML, CSS i JavaScript. Jednak gdy spojrzy się potem na formularz, który składa się z suwaków, kontrolki kalendarza, pokręteł, pól uzupełnianych automatycznie i wizualnych edytorów, szybko dochodzi się do wniosku, że jego utrzymanie będzie koszmarem. Trzeba będzie bowiem zapewnić, że kontrolki umieszczane na stronie nie będą popadać w konflikt z innymi kontrolkami na niej obecnymi

ani z żadną biblioteką kodu JavaScript używaną na stronie. Można spędzić długie godziny na implementowaniu kontrolki kalendarza, a na koniec dowiedzieć się, że w bibliotece Prototype występuje teraz problem wynikający z przejęcia funkcji `$()` przez bibliotekę jQuery. Pierwszym pomysłem jest wtedy użycie metody `noConflict()`, lecz od razu okazuje się z kolei, że kontrolka wyboru koloru przestała działać, ponieważ obsługująca ją wtyczka nie została prawidłowo zaimplementowana.

Jeśli się teraz uśmiechasz, to znak, że już przez to przechodziłeś. Jeśli się wściekasz, przyczyna jest pewnie ta sama. Jest jednak dla nas nadzieja. W tym rozdziale utworzymy kilka formularzy internetowych, które będą zawierać nowe pola formularzy. Zaimplementujemy także mechanizm autofokusu i tekst zastępczy. Na koniec natomiast pokażę, jak używa się nowego atrybutu `contenteditable`, aby za jego pomocą zmienić dowolne pole HTML w kontrolkę na dane wejściowe.

W niniejszym rozdziale opisane zostaną następujące rozwiązania¹:

Pole adresu poczty elektronicznej [`<input type="email">`]

Wyświetla pole formularza do wpisywania adresów poczty elektronicznej. [O10.1, IOS]

Pole adresu URL [`<input type="url">`]

Wyświetla pole formularza do wpisywania adresów URL. [O10.1, IOS]

Pole numeru telefonu [`<input type="tel">`]

Wyświetla pole formularza do wpisywania numerów telefonów. [O10.1, IOS]

Pole wyszukiwania [`<input type="search">`]

Wyświetla pole formularza do wpisywania poszukiwanych słów kluczowych. [C5, S4, O10.1, IOS]

Suwak [`<input type="range">`]

Wyświetla kontrolkę suwaka. [C5, S4, O10.1]

¹ W opisach przedstawionych poniżej wskazano zakres obsługi poszczególnych elementów przy użyciu skrótów umieszczonych w nawiasach kwadratowych. Poszczególne skróty oznaczają: C — Google Chrome; F — Firefox; IE — Internet Explorer; O — Opera; S — Safari; IOS — urządzenia iOS z przeglądarką Mobile Safari; A — przeglądarka Android.

Liczba [`<input type="number">`]

Wyświetla pole formularza przeznaczone do wpisywania liczb, często w postaci kontrolki pokrętła. [C5, S5, O10.1, IOS]

Pole daty [`<input type="date">`]

Wyświetla pole formularza do wpisywania dat. Obsługuje następujące typy: `date`, `month` oraz `week`. [C5, S5, O10.1]

Daty wraz z godzinami

Wyświetla pole formularza do wpisywania dat wraz z godzinami. Obsługuje następujące typy: `datetime`, `datetime-local`, `time`. [C5, S5, O10.1]

Kolor [`<input type="color">`]

Wyświetla pole formularza do wskazywania kolorów. [C5, S5] (Chrome 5 i Safari 5 rozpoznają pole `color`, lecz nie wyświetlają żadnego konkretnego koloru).

Obsługa autofokusu [`<input type="text" autofocus>`]

Obsługa mechanizmu ustawiania fokusu na konkretnym elemencie formularza. [C5, S4]

Obsługa tekstów zastępczych [`<input type="email" placeholder="ja@przyklad.com">`]

Obsługa mechanizmu wyświetlania tekstu zastępczego w polu formularza. [C5, S4, F4]

Obsługa edycji wewnątrz strony [`<p contenteditable>lorem ipsum</p>`]

Obsługa mechanizmu edytowania danych z poziomu przeglądarki. [C4, S3.2, IE6, O10.1]

Najpierw poznamy najbardziej przydatne typy pól formularza.

Porada 3. Opisywanie danych za pomocą nowych pól wejściowych

W HTML5 wprowadzono kilka nowych typów pól wejściowych, za pomocą których można bardziej precyzyjnie opisać dane wejściowe wpisywane przez użytkowników. Oprócz standardowych pól tekstowych, przycisków opcji i pól wyboru można definiować nowe elementy, takie jak pola na adresy poczty elektronicznej, kalendarze, kontrolki wyboru koloru, pokrętła i suwaki. Na podstawie tak zdefiniowanych pól przeglądarki mogą udostępniać użytkownikom lepsze kontrolki bez konieczności wykonywania kodu języka JavaScript. W urządzeniach przenośnych oraz w wirtualnych klawiaturach dla tabletów i ekranów dotykowych na podstawie typów pól można wyświetlać klawiatury w różnych układach. Na przykład przeglądarka Mobile Safari obecna w iPhone'ie odpowiednio dostosowuje układ klawiatury, gdy użytkownik wpisuje dane w polach typu URL i email. W wyświetlanym wówczas układzie klawiatury łatwiej dostępne stają się klawisze takie jak @, ., : i [

Ulepszenie formularza na stronie firmy Pełen Wypas S.A.

Firma Pełen Wypas S.A. opracowuje nową aplikację internetową do zarządzania projektami. Witryna ma za zadanie ułatwić programistom i menedżerom monitorowanie postępów prac przy licznych projektach, które są realizowane w firmie. Dla każdego projektu definiuje się nazwę, adres poczty elektronicznej osoby kontaktowej oraz roboczy adres URL strony, na której można śledzić postęp w rozwoju witryny. Dodatkowo na stronie znajdują się pola na datę początku projektu, jego priorytet i szacowaną liczbę godzin roboczych, które trzeba poświęcić na prace projektowe. Menedżer projektu powinien mieć również możliwość przypisywania projektowi określonego koloru, który pozwoli mu szybko zidentyfikować poszczególne projekty na raportach.

W następnym punkcie opracujemy szablon strony do definiowania preferencji użytkownika. Strona będzie zawierać nowe pola HTML5.

Projekt prostego formularza

Najpierw utworzymy prosty formularz HTML, który będzie przesyłany metodą POST. Ponieważ pole, w którym wpisuje się nazwę projektu, nie musi posiadać żadnych szczególnych cech, stworzymy je na podstawie standardowego pola text.

```
html5_formularze/index.html
```

```
<form method="post" action="/projects/1">

  <fieldset id="personal_information">
    <legend>Informacje na temat projektu</legend>
    <ol>
      <li>
        <label for="name">Nazwa</label>
        <input type="text" name="name" autofocus id="name">
      </li>
      <li>
        <input type="submit" value="Wyślij">
      </li>
    </ol>
  </fieldset>

</form>
```

Zwróćmy uwagę, że kontrolki formularza są opisywane przy użyciu etykiet wchodzących w skład listy uporządkowanej. Etykiety mają kluczowe znaczenie, gdy tworzy się przyjazne formularze. Atrybut `for` etykiety odwołuje się do identyfikatora `id` powiązanego z nią elementu formularza. Dzięki temu rozwiązaniu programy odczytujące zawartość ekranu mogą zidentyfikować pola obecne na stronie. Lista uporządkowana pozwala ułożyć pola bez konieczności definiowania skomplikowanej tabeli ani struktur `div`. Jednocześnie za pomocą listy można wyznaczyć kolejność, w której użytkownicy powinni wypełniać kolejne pola formularza.

Tworzenie suwaka dla zakresu wartości

Suwaki wykorzystuje się po to, by umożliwić użytkownikom zmniejszanie lub zwiększanie wartości liczbowej. Za pomocą tej kontrolki umożliwimy menedżerom szybkie i proste wizualizowanie i zmienianie priorytetu dla projektu. Suwak implementuje się z zastosowaniem typu `range`.

```
html5_formularze/index.html
```

```
<label for="priority">Priorytet</label>
<input type="range" min="0" max="10"
       name="priority" value="0" id="priority">
```

Element suwaka należy dodać do formularza wewnątrz elementu `li`, tak samo jak poprzednio zdefiniowane pole nazwy projektu.

Zarówno Chrome, jak i Opera obsługują kontrolkę suwaka, która prezentuje się w sposób widoczny na rysunku 3.1.



Rysunek 3.1. Kontrolka suwaka wyświetlona w przeglądarce Chrome

Jak widać w definicji suwaka, zdefiniowano dla niego również wartości min i max reprezentowanego zakresu. W ten sposób ograniczyliśmy zakres wartości, jakie można wybrać przy użyciu kontrolki.

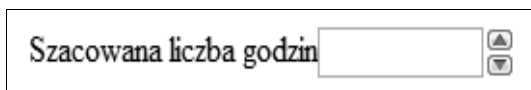
Obsługa liczb za pomocą pola pokręta

W formularzach często trzeba podawać liczby. Wprawdzie wpisywanie liczb nie jest niczym trudnym, jednak czynność tę można jeszcze nieco ułatwić. Pole pokręta to kontrolka z przyciskami strzałek, które zwiększają lub zmniejszają wartość widoczną w polu tekstowym. Pola pokręta użyjemy do wpisywania szacowanej liczby godzin do przepracowania przy projekcie. W ten sposób łatwo będzie zmieniać liczbę godzin.

```
html5_formularze/index.html
```

```
<label for="estimated_hours">Szacowana liczba godzin</label>  
<input type="number" name="estimated_hours"  
      min="0" max="1000"  
      id="estimated_hours">
```

Pole pokręta jest obsługiwane przez przeglądarki Opera i Chrome. Wygląd pola pokręta w przeglądarce Opera przedstawiono na rysunku 3.2.



Rysunek 3.2. Pole pokręta wyświetlone w przeglądarce Opera

W polu pokręta domyślnie można także wpisywać liczby z klawiatury. Ponadto podobnie jak dla suwaków, dla pola pokręta można definiować wartości minimalną i maksymalną. Jednak wartość minimalna i maksymalna nie będzie brana pod uwagę, jeżeli wartość zostanie wpisana w polu z klawiatury.

Warto zaznaczyć, że za pomocą parametru `step` można zdefiniować postępowanie wartości w polu — wartość parametru wskazuje, o ile ma się jednorazowo zmienić wartość obecna w polu. Domyślnie postępowanie ma wartość 1, lecz można mu przypisać dowolną inną wartość liczbową.

Daty

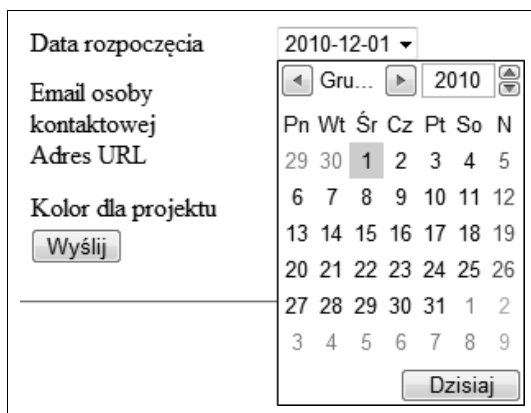
Istotną sprawą jest wskazanie daty rozpoczęcia projektu. Chcemy, by było to jak najprostsze. Doskonale do tego celu nadaje się pole typu `date`.

```
html5_formularze/index.html
```

```
<label for="start_date">Data rozpoczęcia</label>  
<input type="date" name="start_date" id="start_date"  
value="2010-12-01">
```

W trakcie powstawania tej książki jedyną przeglądarką, która w pełni obsługiwała kontrolkę kalendarza, była przeglądarka Opera.

Na rysunku 3.3 przedstawiono przykładową implementację kontrolki kalendarza w przeglądarce Opera.



Rysunek 3.3. Kontrolka kalendarza wyświetlona w przeglądarce Opera

Przeglądarka Safari 5.0 wyświetla zamiast kontrolki kalendarza pole podobne do pola typu `number`, z przyciskami ze strzałkami, za pomocą których można ustawić datę. Jeżeli pole jest puste, przyjmowana jest domyślna wartość „1582”. Pozostałe przeglądarki zamiast kontrolki kalendarza wyświetlają zwykłe pole tekstowe.

Adres poczty elektronicznej

Specyfikacja HTML5 stanowi, że pole wejściowe typu `email` jest przeznaczone do wpisywania jednego adresu poczty elektronicznej lub listy takich adresów. Pole to posłuży nam zatem do wpisywania w formularzu adresu poczty elektronicznej osoby kontaktowej.

```
html5_formularze/index.html
```

```
<label for="email">Email osoby kontaktowej</label>  
<input type="email" name="email" id="email">
```

Największą korzyść z zastosowania pola typu `email` uzyskuje się na urządzeniach przenośnych, w których układ wirtualnej klawiatury zmienia się tak, aby łatwiej dostępne były klawisze ze znakami niezbędnymi do wpisanie adresu poczty elektronicznej.

Adres URL

Dostępne jest także pole specjalnie przystosowane do wpisywania adresów URL. Sprawdza się ono zwłaszcza wówczas, gdy użytkownik wywołujący stronę korzysta z iPhone'a, ponieważ urządzenie zmienia wówczas układ klawiatury na taki, w którym znajdują się przyciski ułatwiające szybkie wpisywanie adresów internetowych. Jest to rozwiązanie podobne do klawiatury wyświetlanej w trakcie wpisywania adresu URL w pasku adresów w przeglądarce Mobile Safari. W celu umieszczenia na formularzu pola na adres URL wystarczy użyć następującej definicji:

```
html5_formularze/index.html
```

```
<label for="url">Adres URL</label>  
<input type="url" name="url" id="url">
```

Również wirtualne klawiatury zmieniają swój układ zależnie od typu pola wejściowego.

Kolory

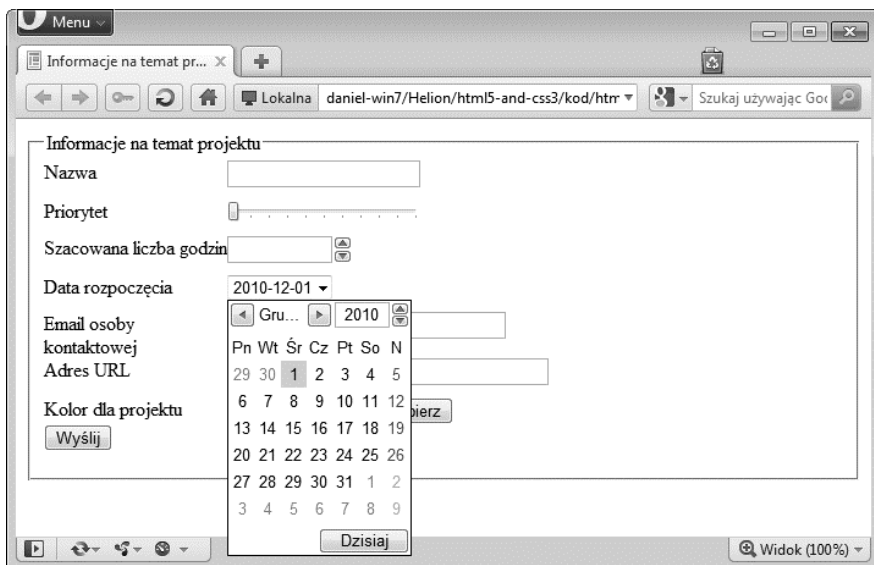
W ostatnim polu formularza należy podać kolor dla projektu. Posłuży do tego pole z typem `color`.

```
html5_formularze/index.html
```

```
<label for="project_color">Kolor dla projektu</label>  
<input type="color" name="project_color" id="project_color">
```

W trakcie powstawania książki żadna przeglądarka nie obsługiwała pola wejściowego typu `color`, jednak nie jest to powód, by w ogóle z niego zrezygnować. Odpowiedni znacznik precyzyjnie opisuje przeznaczenie pola, co przyda się w przyszłości, zwłaszcza gdy zajdzie potrzeba opracowania rozwiązania zastępczego.

Obecnie większość z przedstawionych nowych kontrolki jest obsługiwana przez przeglądarkę Opera, co widać na rysunku 3.4. Jednak gdy tę samą stronę otworzy się w przeglądarce Firefox, Safari albo Google Chrome, będzie widać tylko niewielkie różnice. I te właśnie różnice trzeba będzie wyeliminować.

**Rysunek 3.4.**

Niektóre kontrolki formularza są już obsługiwane przez przeglądarkę Opera

Rozwiązanie zastępcze

Przeglądarki, które nie potrafią rozpoznać pól wejściowych nowych typów, prezentują je jako pola typu `text`. Formularz będzie więc nadal zdalny do użycia. Na chwilę obecną do takich pól można podpinąć widżety interfejsu użytkownika z biblioteki jQuery UI albo YUI, które odpowiednio takie pole przekształca. W miarę upływu czasu coraz więcej przeglądarek będzie obsługiwać pola nowych typów, więc będzie można stopniowo usuwać zastępcze rozwiązania oparte na kodzie JavaScript.

Zastąpienie kontrolki do wskazywania koloru

Kontrolkę do wskazywania koloru można bez trudu zastąpić przy użyciu biblioteki jQuery oraz selektorów atrybutów obecnych w CSS3. Wystarczy w tym celu zlokalizować pole `input` typu `color` i przypisać mu plugin biblioteki jQuery o nazwie `SimpleColor`.

```
html5_formularze/index.html
```

```
if (!hasColorSupport()){  
    $('input[type=color]').simpleColor();  
}
```

Ponieważ w kodzie znaczników użyto nowych typów dla pól formularzy, nie trzeba definiować dodatkowych nazw klas ani innych znaczników, które identyfikowałyby kontrolkę do wskazywania koloru. Selektory atrybutów i HTML5 w zupełności wystarczą.

Jeżeli przeglądarka obsługuje kontrolkę nowego typu, nie ma potrzeby korzystania z pluginu. Należy więc przy użyciu kodu JavaScript sprawdzić, czy przeglądarka obsługuje pola wejściowe z atrybutem `type` o wartości `color`.

```
html5_formularze/index.html
```

```
1 function hasColorSupport(){
2   input = document.createElement("input");
3   input.setAttribute("type", "color");
4   var hasColorType = (input.type !== "text");
5   // obsługa częściowej implementacji Safari/Chrome
6   if(hasColorType){
7     var testString = "foo";
8     input.value=testString;
9     hasColorType = (input.value !== testString);
10  }
11  return(hasColorType);
12 }
```

Najpierw w kodzie JavaScript tworzony jest nowy element, a jego atrybutowi `type` przypisywana jest wartość `color`. Następnie odczytujemy atrybut `type`, aby sprawdzić, czy przeglądarka pozwoliła przypisać mu potrzebną nam wartość. Jeżeli odczytaną wartością atrybutu `type` jest `color`, oznacza to, że przeglądarka prawidłowo obsługuje kontrolkę do wskazywania koloru. Jeżeli wartość atrybutu jest inna, wówczas trzeba użyć pluginu.

Ciekawie przedstawia się zwłaszcza szósty wiersz przedstawionego kodu. Safari 5 i Google Chrome 5 częściowo obsługują już typ `color`. Mówiąc bardziej precyzyjnie, przeglądarki obsługują pola tego typu, lecz nie wyświetlają właściwej kontrolki do wskazywania koloru, a zamiast niej umieszczają na formularzu standardowe pole tekstowe. Dlatego w metodzie, która sprawdza, czy kontrolka jest obsługiwana, przypisujemy wartość polu wejściowemu, a następnie sprawdzamy, czy wartość ta zostaje zapamiętana. Jeżeli wartość nie zostanie zapamiętana, jest to znak, że przeglądarka w pełni obsługuje kontrolkę do wskazywania koloru, ponieważ pole wejściowe nie zachowuje się jak standardowe pole tekstowe.

Kod, który zastępuje kontrolkę do wskazywania koloru widżetem, przedstawia się następująco:

```
html5_formularze/index.html
```

```
if (!hasColorSupport()){
  $('input[type=color]').simpleColor();
}
```


Tak skonstruowane rozwiązanie działa, lecz jest dość ograniczone. Jest ono przeznaczone tylko dla niektórych przeglądarek i rozwiązuje problem jedynie z kontrolką do wskazywania koloru. Z innymi kontrolkami wiążą się inne problemy, o których warto wiedzieć. Na szczęście istnieje rozwiązanie alternatywne.

Modernizr

Biblioteka Modernizr² potrafi rozpoznawać, czy dostępna jest obsługa większości funkcji obecnych w HTML5 i CSS3. Jeżeli przeglądarka nie obsługuje danej funkcji, biblioteka jej w tym nie zastępuje, lecz udostępnia szereg rozwiązań podobnych do zaimplementowanego przez nas w poprzednim punkcie rozwiązania rozpoznającego pola formularza. Co więcej, rozwiązania dostępne w bibliotece Modernizr są znacznie bardziej niezawodne.

Zanim zaczniemy wykorzystywać Modernizr we własnych projektach, należy najpierw poświęcić trochę czasu na zrozumienie, jak biblioteka działa. Programista jest bowiem odpowiedzialny za kod użyty w projekcie bez względu na to, czy jest jego autorem, czy nie. Modernizr nie potrafi jeszcze rozpoznawać częściowej obsługi pola do wskazywania koloru, obecnej w przeglądarce Safari. Gdy udostępniona zostanie nowa wersja przeglądarki Chrome lub Firefox, być może trzeba będzie samodzielnie opracować nowe rozwiązanie. Kto wie, może Twoje rozwiązanie stanie się częścią biblioteki Modernizr?

Rozwiązania zastępcze dla kontrolki takich jak kontrolki kalendarza oraz suwaki implementuje się w podobny sposób. Suwaki i kontrolki kalendarza to komponenty dostępne w bibliotece jQuery UI³. Bibliotekę jQuery UI należy dołączyć do strony, sprawdzić, czy przeglądarka sama obsługuje wybraną kontrolkę, a jeżeli tej obsługi brakuje, użyć implementacji analogicznej kontrolki w języku JavaScript.

Za jakiś czas będzie można całkowicie zrezygnować z kontrolki JavaScript i korzystać wyłącznie z mechanizmów zaimplementowanych w przeglądarkach. Biblioteka Modernizr jest o tyle pomocna, że zdejmuje z programisty obowiązek tworzenia kodu rozpoznającego obsługę poszczególnych typów kontrolki, który zwykle jest dość skomplikowany. Tym niemniej

² <http://www.modernizr.com/>

³ <http://jqueryui.com/>

w dalszej części książki nadal będziemy implementować kod rozpoznający obecność poszczególnych funkcji w przeglądarce, aby pokazać istotę jego działania.

Oprócz nowych typów pól formularza HTML5 definiuje kilka nowych atrybutów pól formularzy, które zwiększają ich użyteczność. W następnym punkcie omówię atrybut `autofocus`.

Porada 4. Przechodzenie do pierwszego pola formularza przy użyciu atrybutu autofocus

Czynność wypełniania formularza można znacznie przyspieszyć, gdy w momencie ładowania strony umieści się kursor w pierwszym polu formularza. W wielu wyszukiwarkach internetowych dokonuje się tego za pomocą odpowiedniego kodu JavaScript. Obecnie natomiast HTML5 udostępnia podobną funkcję jako element języka.

Całe rozwiązanie sprowadza się do tego, by do wybranego pola formularza dodać atrybut `autofocus` w taki sam sposób, jak zrobiliśmy to już na stronie profilu utworzonej w poradzie „Opisywanie danych za pomocą nowych pól wejściowych” na stronie 56.

```
html5_formularze/index.html
```

```
<label for="name">Nazwa</label>
<input type="text" name="name" autofocus id="name">
```

Aby atrybut `autofocus` działał poprawnie, powinien występować tylko jeden raz na stronie. Jeżeli atrybut wystąpi więcej niż jeden raz, przeglądarka umieści kursor w ostatnim polu z tym atrybutem.

Rozwiązanie zastępcze

Jeżeli przeglądarka użytkownika nie obsługuje atrybutu `autofocus`, można ten fakt rozpoznać za pomocą odpowiednio sformułowanego kodu JavaScript. Następnie należy użyć biblioteki jQuery i umieścić kursor w odpowiednim polu formularza. Jest to najprawdopodobniej najprostsze rozwiązanie zastępcze, które znajduje się w książce.

```
html5_formularze/autofocus.js
```

```
function hasAutofocus() {
    var element = document.createElement('input');
    return 'autofocus' in element;
}

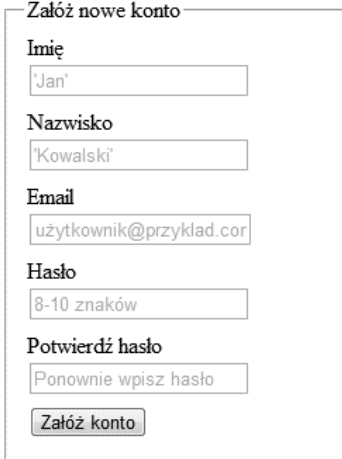
$(function(){
    if(!hasAutofocus()){
        $('input[autofocus=true]').focus();
    }
});
```

Wystarczy umieścić zaprezentowany kod JavaScript na stronie, aby uzyskać odpowiednią obsługę atrybutu `autofocus`.

Autofokus ułatwia użytkownikom wypełnianie formularza. Jeszcze większym ułatwieniem będzie jednak udostępnienie informacji na temat tego, jakich konkretnie danych oczekujemy w poszczególnych polach formularza. Dlatego w następnym punkcie zajmiemy się atrybutem `placeholder`.

Porada 5. Wyświetlanie wskazówek w postaci tekstu zastępczego

Tekst zastępczy stanowi wskazówkę dla użytkownika, która mówi, jakie dane należy wpisywać w poszczególnych polach. Na rysunku 3.5 przedstawiono formularz do tworzenia nowego konta, w którym zastosowano tekst zastępczy. Formularz ten zdefiniujemy w następnym punkcie.



Założ nowe konto

Imię
Jan

Nazwisko
Kowalski

Email
uzytkownik@przyklad.cor

Hasło
8-10 znaków

Potwierdź hasło
Ponownie wpisz hasło

Założ konto

Rysunek 3.5.

Teksty zastępcze wskazują użytkownikowi, jakie dane powinien wpisać w poszczególnych polach

Formularz zakładania nowego konta

Na stronie firmy Pelen Wypas S.A. użytkownik musi najpierw założyć nowe konto. Poważnym problemem, który zawsze pojawia się w takim przypadku, jest to, że użytkownicy podają hasła łatwe do odgadnięcia. Dlatego użyjemy tekstu zastępczego, aby wskazać użytkownikowi wymagania co do postaci hasła. Dla zachowania spójności tekst zastępczy umieścimy także w pozostałych polach tekstowych.

Aby umieścić w polu formularza tekst zastępczy, wystarczy do każdego takiego pola dodać atrybut `placeholder` w następujący sposób:

```
html5_tekstzastepczy/index.html
```

```
<input id="email" type="email"  
      name="email" placeholder="uzytkownik@przyklad.com">
```

Kod znaczników całego formularza przedstawiono na poniższym listingu. W każdym polu formularza umieszczono tekst zastępczy.

```
html5_tekstzastepczy/index.html
```

```
<form id="create_account" action="/signup" method="post">
  <fieldset id="signup">
    <legend>Założ nowe konto</legend>
    <ol>
      <li>
        <label for="first_name">Imię</label>
        <input id="first_name" type="text"
          autofocus="true"
          name="first_name" placeholder="'Jan'">
      </li>
      <li>
        <label for="last_name">Nazwisko</label>
        <input id="last_name" type="text"
          name="last_name" placeholder="'Kowalski'">
      </li>
      <li>
        <label for="email">Email</label>
        <input id="email" type="email"
          name="email" placeholder="uzytkownik@przyklad.com">
      </li>
      <li>
        <label for="password">Hasło</label>
        <input id="password" type="password" name="password" value=""
          autocomplete="off" placeholder="8-10 znaków" />
      </li>
      <li>
        <label for="password_confirmation">Potwierdź hasło</label>
        <input id="password_confirmation" type="password"
          name="password_confirmation" value=""
          autocomplete="off" placeholder="Ponownie wpisz hasło" />
      </li>
      <li><input type="submit" value="Założ konto"></li>
    </ol>
  </fieldset>
</form>
```

Wyłączanie automatycznego uzupełniania

Nietrudno zauważyć, że do pól formularza, w których wpisuje się hasło, dodano atrybut `autocomplete`. Atrybut ten jest nowością w HTML5 i wskazuje on przeglądarkom, że nie powinny automatycznie wypełniać tego pola danymi. Niektóre przeglądarki pamiętają bowiem dane wpisane w przeszłości przez użytkownika, a w niektórych sytuacjach pożądane jest, by przeglądarki nie wstawiały ponownie tych samych danych.

Ponieważ również w tym formularzu pola są zdefiniowane w ramach listy uporządkowanej, można dodać kaskadowy arkusz stylów, aby nadać stronie bardziej przyjazny wygląd.

```
html5_tekstzastepczy/style.css
```

```
fieldset{
  width: 216px;
}

fieldset ol{
  list-style: none;
  padding:0;
  margin:2px;
}

fieldset ol li{
  margin:0 0 9px 0;
  padding:0;
}

/* Pola wejściowe mają znajdować się w oddzielnych wierszach */
fieldset input{
  display:block;
}
```

Dzięki takiemu rozwiązaniu użytkownicy przeglądarek Safari, Opera i Chrome będą widzieć wskazówki w postaci tekstu zastępczego widocznego w polach wejściowych. Pozostało nam tylko opracować rozwiązanie zastępcze, które da ten sam efekt w przeglądarkach Firefox i Internet Explorer.

Rozwiązanie zastępcze

Za pomocą krótkiego kodu języka JavaScript w polach formularzy można umieszczać teksty zastępcze. W tym celu należy sprawdzić zawartość każdego pola formularza, a jeżeli jest ono puste, trzeba przypisać mu tekst zastępczy. Gdy w polu formularza zostanie umieszczony kursor, tekst zastępczy się usuwa, a gdy kursor zostanie przeniesiony do innego pola, ponownie należy sprawdzić jego zawartość. Jeżeli zawartość pola jest inna niż tekst zastępczy, pozostawia się ją bez zmian. Jeżeli natomiast pole jest puste, wstawia się w nim tekst zastępczy.

Aby sprawdzić, czy przeglądarka obsługuje tekst zastępczy, należy wykorzystać rozwiązanie podobne do tego, za pomocą którego sprawdzaliśmy obsługę autofokusu.

```
html5_tekstzastepczy/index.html
```

```
function hasPlaceholderSupport() {  
    var i = document.createElement('input');  
    return 'placeholder' in i;  
}
```

Teraz pozostaje już tylko napisać kod JavaScript, który będzie obsługiwał zmiany tekstu. Do tego celu wykorzystamy rozwiązanie, które opracował Andrew January⁴ wraz z zespołem. Za pomocą skryptu wszystkie pola formularza będziemy wypełniać tekstem zastępczym przechowywanym w atrybucie `placeholder`. Gdy użytkownik umieści kursor w polu, umieszczony w nim tekst zostanie usunięty. Całość kodu zaimplementujemy w postaci pluginu biblioteki jQuery, aby łatwo było go wykorzystać w formularzu. Więcej informacji na temat sposobu działania pluginów znajduje się w ramce na stronie 72.

```
html5_tekstzastepczy/jquery.placeholder.js
```

```
1 (function($){  
2  
3     $.fn.placeholder = function(){  
4  
5         function valueIsPlaceholder(input){  
6             return $(input).val() == $(input).attr("placeholder");  
7         }  
8         return this.each(function() {  
9  
10            $(this).find(":input").each(function(){  
11  
12                if($(this).attr("type") == "password"){  
13  
14                    var new_field = $("<input type='text'>");  
15                    new_field.attr("rel", $(this).attr("id"));  
16                    new_field.attr("value", $(this).attr("placeholder"));  
17                    $(this).parent().append(new_field);  
18                    new_field.hide();  
19  
20                    function showPasswordPlaceholder(input){  
21                        if( $(input).val() == "" || valueIsPlaceholder(input) ){  
22                            $(input).hide();  
23                            $('input[rel=' + $(input).attr("id") + ']').show();  
24                        };  
25                    };  
26                }  
27            }  
28        }  
29    }  
30 };
```

⁴ Skrypt w oryginalnej postaci jest dostępny na stronie pod adresem <http://www.morethannothing.co.uk/wp-content/uploads/2010/01/placeholder.js>. Skrypt ten nie obsługuje jednak pól haseł w Internet Explorerze.


```
26
27     new_field.focus(function(){
28         $(this).hide();
29         $('input#' + $(this).attr("rel")).show().focus();
30     });
31
32     $(this).blur(function(){
33         showPasswordPlaceholder(this, false);
34     });
35
36     showPasswordPlaceholder(this);
37
38 }else{
39
40     // Zastąpienie wartości tekstem zastępczym.
41     // Opcjonalny parametr reload pozwala na zapisywanie wartości pól
42     // w pamięci podręcznej przez FF i IE.
43     function showPlaceholder(input, reload){
44         if( $(input).val() == "" ||
45             ( reload && valueIsPlaceholder(input) ) ){
46             $(input).val($(input).attr("placeholder"));
47         }
48     };
49
50     $(this).focus(function(){
51         if($(this).val() == $(this).attr("placeholder")){
52             $(this).val("");
53         };
54     });
55
56     $(this).blur(function(){
57         showPlaceholder($(this), false)
58     });
59
60     showPlaceholder(this, true);
61 };
62 });
63
64 // Zapobiega wysłaniu formularza z wartościami domyślnymi
65 $(this).submit(function(){
66     $(this).find(":input").each(function(){
67         if($(this).val() == $(this).attr("placeholder")){
68             $(this).val("");
69         }
70     });
71 });
72 });
73
74 });
75 };
76
77 })(jQuery);
```

W przedstawionym kodzie źródłowym pluginu znajduje się kilka interesujących rozwiązań, na które warto zwrócić uwagę. W wierszu 45. tekst zastępczy zostaje umieszczony w polu, jeżeli nie ma ono żadnej wartości, ale również wówczas, gdy nastąpiło ponowne załadowanie strony. Firefox i inne przeglądarki zachowują wartości obecne w polach formularza. Skrypt przypisuje atrybutowi `value` pól teksty zastępcze, ponieważ nie chcemy, by pozostała w nim wartość wcześniej wpisana przez użytkownika. W momencie ładowania strony do metody `showPlaceholder()` przekazujemy wartość `true`, co widać w wierszu 61.

Pluginy jQuery

Bibliotekę jQuery można rozszerzać przez implementowanie własnych pluginów. Gdy zaimplementowaną samodzielnie metodę doda się do funkcji jQuery, plugin stanie się automatycznie dostępny dla innych programistów, którzy dołączą do swojego kodu tak rozszerzoną bibliotekę. Oto najprostszy przykład, który powoduje wyświetlenie okna komunikatu JavaScript:

```
jQuery.fn.debug = function() {  
    return this.each(function(){  
        alert(this.html());  
    });  
};
```

Aby wyświetlić okno dla każdego akapitu na stronie, wystarczy wywołać funkcję w następujący sposób:

```
$("#p").debug();
```

Pluginy biblioteki jQuery są konstruowane w taki sposób, że przechodzą przez kolekcję obiektów jQuery oraz zwracają taką kolekcję obiektów. Można więc z uzyskanych w ten sposób obiektów tworzyć łańcuch. Na przykład dzięki temu, że przedstawiony powyżej plugin o nazwie `debug` również zwraca kolekcję jQuery, można wykorzystać metodę `css` biblioteki i zmienić kolor tekstu wszystkich akapitów jednym wierszem kodu.

```
$("#p").debug().css("color", "red");
```

W książce jeszcze co najmniej kilka razy skorzystamy z pluginów biblioteki jQuery, aby utrzymać odpowiednią organizację kodu źródłowego stanowiącego rozwiązanie zastępcze. Więcej informacji na temat pluginów jQuery można znaleźć na stronie z dokumentacją biblioteki*.

* <http://docs.jquery.com/Plugins/Authorin>

Pola przeznaczone na wpisywanie haseł zachowują się inaczej niż pozostałe pola formularza, dlatego też trzeba je inaczej potraktować. Spójrzmy na wiersz 12. Najpierw rozpoznajemy w nim obecność pola hasła. Potem trzeba zmienić jego typ na zwykłe pole tekstowe, aby jego wartość nie została ukryta pod postacią gwiazdek. Niektóre przeglądarki zwracają błąd, gdy próbuje się przekształcić pola haseł, dlatego musimy zastąpić pole hasła zwykłym polem tekstowym. Pole hasła i pole tekstowe będziemy zamieniać ze sobą nawzajem zależnie od czynności wykonywanych przez użytkownika.

Opracowane rozwiązanie zmienia wartości pól na formularzu, dlatego należy zapewnić, że teksty zastępcze nie zostaną przesłane do serwera jako dane formularza. Ponieważ kod do obsługi tekstów zastępczych jest wykonywany jedynie wówczas, gdy w przeglądarce włączona jest obsługa języka JavaScript, możemy przy użyciu tego języka wykryć próbę przesłania formularza na serwer i usunąć z niego wartości, które są identyczne jak teksty zastępcze. W wierszu 66. rozpoznawane jest zdarzenie zatwierdzenia formularza, po czym kod usuwa wartości z wszystkich pól wejściowych, w których znajduje się tekst zastępczy.

Dzięki temu, że nasze rozwiązanie zaimplementowaliśmy w postaci pluginu, możemy je teraz wywołać na stronie przez podłączenie pluginu do formularza w następujący sposób:

```
html5_tekstzastepczy/index.html
```

```
$(function(){  
  function hasPlaceholderSupport() {  
    var i = document.createElement('input');  
    return 'placeholder' in i;  
  }  
  
  if(!hasPlaceholderSupport()){  
    $("#create_account").placeholder();  
    // KONIEC rozwiązania zastępczego  
  
    $('input[autofocus=true]').focus();  
  
  };  
});
```

W ten sposób powstało całkiem eleganckie rozwiązanie, które pozwala umieszczać teksty zastępcze w formularzach aplikacji internetowych w dowolnej przeglądarce.

Porada 6. Edytowanie danych bezpośrednio na stronie przy użyciu atrybutu contenteditable

Zawsze warto szukać sposobów, dzięki którym użytkownikom łatwiej będzie korzystać z aplikacji. Czasami warto pozwolić użytkownikom na edytowanie danych na ich temat bez konieczności odsyłania ich do dedykowanego formularza. Standardowo mechanizm edycji danych bezpośrednio na stronie implementuje się w taki sposób, że najpierw rozpoznaje się kliknięcia myszą w regionie, w którym znajduje się tekst, a następnie w miejsce regionu wstawia się pole tekstowe. Zmieniona zawartość takiego pola tekstowego jest potem przesyłana za pośrednictwem technologii Ajax z powrotem do serwera. Natomiast obecny w HTML5 znacznik `contenteditable` automatycznie udostępnia możliwość wpisywania danych bezpośrednio na stronie. Również w przypadku zastosowania znacznika `contenteditable` trzeba zaimplementować kod JavaScript, który będzie przysyłał zmienione dane na serwer w celu ich zapisania, lecz nie trzeba już wówczas tworzyć ani przełączać ukrytych pól formularzy.

Celem jednego z projektów realizowanych w firmie Pelen Wypas S.A. jest umożliwienie użytkownikom przeglądania danych na temat ich konta. Na przeznaczony do tego stronie WWW znajduje się nazwisko użytkownika, miasto, województwo, kod pocztowy oraz adres poczty elektronicznej. Dodamy więc możliwość edytowania tych danych bezpośrednio na stronie. W efekcie powinien powstać interfejs widoczny na rysunku 3.6.

Informacje o użytkowniku

Imię i nazwisko	Daniel Kaczmarek
Miasto	<input type="text" value="Dowolne"/>
Województwo	Śląskie
Kod pocztowy	44-100
Email	boss@pelenwypas.com

Rysunek 3.6. Łatwe edytowanie danych bezpośrednio na stronie

Zanim przystąpimy do implementacji, chcę zaznaczyć, że implementowanie jakiegokolwiek mechanizmu opartego na kodzie JavaScript bez uprzedniego zaimplementowania kodu działającego na serwerze jest wbrew wszelkim znanym mi dobrym praktykom tworzenia aplikacji internetowych. Przyjąłem jednak taki tryb pracy tylko dlatego, że chcę skupić się na przedstawieniu działania znacznika `contenteditable`, oraz dlatego, że

nie będzie to kod na środowisko produkcyjne. Zawsze, i to naprawdę **zawsze**, najpierw należy zaimplementować tę część, która nie wymaga kodu JavaScript, a dopiero **potem** tworzyć wersję rozwiązania, która opiera się na skryptach JavaScript. Na koniec powinno się także zaimplementować automatyczne testy dla obydwóch wersji rozwiązania, aby zwiększyć prawdopodobieństwo zidentyfikowania wszystkich potencjalnych błędów, które mogą zająć zwłaszcza wówczas, gdy zmianie ulegnie tylko jedna z wersji.

Formularz edycji danych o koncie

W HTML5 udostępniono znacznik `contenteditable`, który jest dostępny niemal dla każdego elementu. Dodanie znacznika `contenteditable` spowoduje, że element przekształci się w pole dostępne do edycji.

```
html5_edytowaniedanych/show.html
```

```
<h1>Informacje o użytkowniku</h1>
<div id="status"></div>
<ul>
  <li>
    <b>Imię i nazwisko</b>
    <span id="name" contenteditable="true">Daniel Kaczmarek</span>
  </li>
  <li>
    <b>Miasto</b>
    <span id="city" contenteditable="true">Dowlne</span>
  </li>
  <li>
    <b>Województwo</b>
    <span id="state" contenteditable="true">Śląskie</span>
  </li>
  <li>
    <b>Kod pocztowy</b>
    <span id="postal_code" contenteditable="true">44-100</span>
  </li>
  <li>
    <b>Email</b>
    <span id="email" contenteditable="true">boss@pelenwypas.com</span>
  </li>
</ul>
```

Tak skonstruowany formularz można rozszerzyć o kaskadowy arkusz stylów. Do zidentyfikowania pól dostępnych do edycji wykorzystamy selektory CSS3. Pola te będą zmieniać kolor, gdy użytkownik umieści na nich kursor myszy albo je zaznaczy.

```
html5_edytowaniedanych/show.html
```

```
1 ul{list-style:none;}
2
3 li{clear:both;}
```

```
4
5 li>b, li>span{
6   display: block;
7   float: left;
8   width: 100px;
9 }
10
11 li>span{
12   width:500px;
13   margin-left: 20px;
14 }
15
16 li>span[contenteditable=true]:hover{
17   background-color: #ffc;
18 }
19
20 li>span[contenteditable=true]:focus{
21   background-color: #ffa;
22   border: 1px shaded #000;
23 }
```

To już wszystko, jeśli chodzi o część widoczną dla użytkownika. Użytkownicy mogą już teraz łatwo zmieniać dane bezpośrednio na stronie. Pozostaje je tylko zapisać na serwerze.

Zapisywanie danych

Użytkownicy mogą już zmieniać dane, jednak na razie zmiany te zostaną utracone, gdy nastąpi ponowne wywołanie strony bądź też gdy użytkownik przejdzie do innej strony internetowej. Trzeba zapewnić możliwość przesłania wprowadzonych zmian do serwera. Najprościej jest wykorzystać do tego celu bibliotekę jQuery. Jeżeli kiedykolwiek miałeś styczność z technologią Ajax, zastosowane rozwiązanie nie będzie dla Ciebie niczym nowym.

```
html5_edytowaniedanych/show.html
```

```
$(function(){
  var status = $("#status");
  $("#span[contenteditable=true]").blur(function(){
    var field = $(this).attr("id");
    var value = $(this).text();
    $.post("http://localhost:4567/users/1",
      field + "=" + value,
      function(data){
        status.text(data);
      }
    );
  });
});
```

Do każdej sekcji `span`, która znajduje się na stronie i zawiera atrybut `contenteditable` o wartości `true`, dodany został odbiornik zdarzeń. Potem następuje już tylko przesłanie danych do skryptu działającego na serwerze.

Rozwiązanie zastępcze

W zastosowanym rozwiązaniu użyto szeregu rozwiązań, które w niektórych okolicznościach nie będą działać poprawnie. Przede wszystkim wysłanie zmienionych danych do serwera jest wykonywane przez kod JavaScript, co w zasadzie jest złym pomysłem. Poza tym użyliśmy pseudoklasy `focus`, aby wyróżnić pola, w których znajduje się kursor, a pseudoklasa ta nie jest obsługiwana przez niektóre wersje przeglądarki Internet Explorer. Zajmijmy się więc najpierw działaniem strony, a potem jej wyglądem.

Strona do edycji danych

Zamiast wymyślać różne okoliczności, w których użytkownik może nie mieć możliwości skorzystania z zaimplementowanego przez nas rozwiązania, najlepiej jest po prostu dać mu możliwość przejścia do oddzielnej strony z formularzem. Oczywiście wymaga to napisania dodatkowego kodu, jednak weźmy pod uwagę okoliczności, jakie mogą mieć miejsce:

- ◆ Użytkownik nie ma włączonej obsługi języka JavaScript i używa Internet Explorera 7.
- ◆ Użytkownik korzysta z przeglądarki, która nie obsługuje HTML5.
- ◆ Użytkownik korzysta z najnowszej wersji przeglądarki Firefox, która obsługuje HTML5, lecz nie ma włączonej obsługi języka JavaScript, bo po prostu tego języka nie lubi (to może się zdarzyć zawsze, zdecydowanie częściej, niż można by sobie wyobrazić).

Gdy zastanowimy się nad opisanymi okolicznościami, najbardziej sensownym rozwiązaniem zastępczym okaże się stworzenie formularza, który metodą `POST` wykona tę samą czynność, co zaimplementowany przez nas kod Ajax odpowiedzialny za zapisanie zmian. Sposób wykonania rozwiązania zastępczego zależy tylko od nas, jednak wiele dostępnych platform programistycznych umożliwia wykrywanie typu żądania przez sprawdzenie zawartości nagłówków `accept`. Na tej podstawie można się dowiedzieć, czy żądanie zostało wykonane zwykłą metodą `POST`, czy za pomocą `XMLHttpRequest`.

Kod pozostanie wówczas zgodny z zasadą DRY⁵. Łącze do formularza będzie ukrywane, jeżeli przeglądarka będzie obsługiwać znacznik `contenteditable` i język JavaScript.

Utworzymy więc nową stronę o nazwie `edit.html`, która będzie zawierać standardowy formularz edycji danych. Dane będą przesyłane do tego samego adresu, z którego korzysta zaimplementowany przez nas kod Ajax.

```
html5_edytowaniedanych/edit.html
```

```
<!DOCTYPE html>
<html lang="pl-PL">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>Edycja danych o koncie</title>
    <link href="style.css" rel="stylesheet" media="screen">
  </head>
  <body>
    <form action="/users/1" method="post" accept-charset="utf-8">
      <fieldset id="your_information">
        <legend>Informacje o użytkowniku</legend>
        <ol>
          <li>
            <label for="name">Imię i nazwisko</label>
            <input type="text" name="name" value="" id="name">
          </li>
          <li>
            <label for="city">Miasto</label>
            <input type="text" name="city" value="" id="city">
          </li>
          <li>
            <label for="state">Województwo</label>
            <input type="text" name="state" value="" id="state">
          </li>
          <li>
            <label for="postal_code">Kod pocztowy</label>
            <input type="text" name="postal_code" value="" id="postal_code">
          </li>
          <li>
            <label for="email">Email</label>
            <input type="email" name="email" value="" id="email">
          </li>
        </ol>
      </fieldset>
      <p><input type="submit" value="Zapisz"></p>
    </form>
  </body>
</html>
```

⁵ DRY to skrót angielskich słów *Don't Repeat Yourself*, czyli „nie powtarzaj się”. Określenie to zostało sformułowane przez Dave'a Thomasa i Andy'ego Hunta w książce *The Pragmatic Programmer*. [HT00]


```
</form>

</body>
</html>
```

Następnie na stronie *show.html* należy dodać łącze do nowej strony *edit.html*.

```
html5_edytowaniedanych/show.html
```

```
<h1>Informacje o użytkowniku</h1>
<section id="edit_profile_link">
  <p><a href="edit.html">Edycja danych o użytkowniku</a></p>
</section>
<div id="status"></div>
```

Po dodaniu łącza trzeba nieco zmodyfikować skrypt. Łącze do strony edycji danych ma być ukrywane, a kod Ajax wykonywany jedynie wówczas, gdy przeglądarka będzie obsługiwać znacznik `contenteditable`.

```
html5_edytowaniedanych/show.html
```

```
if(document.getElementById("edit_profile_link").contentEditable != null){
```

Skrypt rozszerzony o mechanizm wykrywania obsługi znacznika `contenteditable` ma następującą postać:

```
html5_edytowaniedanych/show.html
```

```
$(function(){
  if(document.getElementById("edit_profile_link").contentEditable != null){
    $("#edit_profile_link").hide();
    var status = $("#status");
    $("#span[contenteditable=true]").blur(function(){
      var field = $(this).attr("id");
      var value = $(this).text();
      $.post("http://localhost:4567/users/1",
        field + "=" + value,
        function(data){
          status.text(data);
        }
      );
    });
  }
});
```

Dzięki takiej implementacji użytkownicy mogą korzystać ze standardowego interfejsu edycji danych bądź też z szybszego trybu edycji danych bezpośrednio na stronie. Wiesz już, w jaki sposób zaimplementować taki interfejs, pamiętaj więc, aby najpierw stworzyć rozwiązanie zastępcze. W odróżnieniu bowiem od rozwiązań zastępczych dla innych przypadków, brak rozwiązania zastępczego w tej sytuacji ograniczy zbiór funkcji dostępnych dla użytkownika.

Przyszłość

Jeśli obecnie na stronie dodana zostanie kontrolka kalendarza zaimplementowana w języku JavaScript, użytkownicy będą musieli poznać sposób jej działania. Jeżeli kiedyś kupowałeś w internecie bilety na samolot i rezerwowałeś pokój w hotelu, zdajesz sobie sprawę, jak różnorodne są sposoby implementowania formularzy na stronach WWW. Podobnie jest z bankomatami, których interfejsy różnią się między sobą czasem tak znacznie, że różnice te mogą spowalniać czynności korzystających z nich osób.

Wyobraźmy sobie jednak, że na każdej witrynie znajduje się pole daty zaimplementowane w HTML5, a odpowiedzialność za stworzenie odpowiedniego interfejsu spoczywa na przeglądarce. Na każdej witrynie odwiedzanej przez użytkowników kontrolka kalendarza miałyby identyczną postać. W oprogramowaniu odczytującym zawartość ekranu można by wręcz zaimplementować standardowy mechanizm, który ułatwiłby osobom niewidomym łatwe wpisywanie dat. Zwróćmy też uwagę, jak bardzo przydatne dla użytkowników staną się teksty zastępcze i autofokus, gdy będą powszechnie obsługiwane. Na podstawie tekstów zastępczych oprogramowanie odczytujące zawartość ekranu może wyjaśniać użytkownikom przeznaczenie poszczególnych pól formularzy, zaś dzięki mechanizmowi autofokusu łatwiej będzie takim osobom nawigować po formularzu bez użycia myszy. Będzie to znaczne ułatwienie zarówno dla osób niewidomych i niedowidzących, jak również dla tych użytkowników, którym ograniczenia ruchowe nie pozwalają na korzystanie z myszy lub je utrudniają.

Możliwość przekształcania dowolnego elementu w region dostępny do edycji znacznie ułatwia edytowanie danych bezpośrednio na stronie. Jednocześnie może to doprowadzić do zmiany sposobu tworzenia interfejsów dla systemów do zarządzania treścią.

Współczesny internet skupia się na interakcji, a formularze są tej interakcji kluczowym elementem. Rozszerzenia dostępne w HTML5 dają programistom cały zbiór nowych narzędzi, dzięki którym można znacznie ułatwić życie użytkownikom.

Skorowidz

@font-face, 185

<!DOCTYPE HTML>, 36

A

A, 33

AAC, 147

abbr, 26

accept, 77

Accessibility for Reach Internet

Applications, 107

acronym, 26

Address, 225

addToNotesList, 210, 212

Adres poczty elektronicznej, 59

Advanced Audio Coding, 147

align, 26

alink, 26

Andrew January, 70

Android, 33, 194, 214, 217, 219

animacja, 251

animowane elementy interfejsu, 248

Apache, 219, 227

API, 158, 221, 226, 254

historii, 223

Static Map, 241

aplikacja odczytująca zawartość ekranu,
107

applet, 26

application, 110

aria-atomic, 108, 118

aria-live, 108, 118

article, 33, 39, 40, 112

artykuł, 33, 39, 40

struktura, 39

aside, 33, 40, 42

assertive, 118

atrybuty

danych, 33

użytkownika, 32, 33

audio, 24, 141

loops, 150

autocomplete, 68

autofocus, 65

Autofokus, 55

AVI, 145

B

background-color, 164

background-image, 181

banner, 110

basefont, 25

beginPath, 128

bgcolor, 26

big, 25

bind, 231, 258
blog, 34
body, 26
border, 26
border-radius, 164, 171
box-shadow, 164, 176

C

C, 33
canPlayType, 153
canvas, 24, 123, 136
Canvas API, 131
cellpadding, 26
cellspacing, 26
center, 25
choroba Divitis, 31
Chrome, 27, 44, 57, 58, 60, 62, 69, 91,
93, 105, 146, 148, 150, 154, 167,
171, 175, 179, 200, 214, 217, 225,
231, 237, 241, 255, 263
Chrome Frame, 215, 254
cieniowanie
obrazków, 21
tekstu, 21
cieniowanie tekstu, 177
cień, 176
elementu, 164
class, 47
click, 49, 210
ClientLocation, 244
closePath, 129
color, 60, 62
Columnizer, 101
complementary, 110
contenteditable, 31, 74, 75, 77
contentinfo, 110, 111
cookie, 194, 200, 201
Corner, 168
Create, Retrieve, Update and Delete, 204
Cross-document Messaging, 193, 226

cross-site scripting, 19
CRUD, 204
CSS3, 17
cubic-bezier, 251
cytat, 40
czat, 233, 234
czcionka, 184, 185
zastępcza, 189

D

dane użytkownika
atrybuty, 47
data, 59
data-, 47
data-name, 135
data-remote=true, 51
dataStorage, 258
date, 59
datetime, 55
datetime-local, 55
definicja pojęcia, 112
definition, 112
deklaracja typu dokumentu, 21, 23, 34
diagram, 40
dir, 26
directory, 112
DirectX, 179, 180
div, 31, 32, 34, 39
dobra praktyka tworzenia aplikacji, 74
doctype, 21
document, 112, 113
DOMAssistant, 91
dygresja, 40

E

ease-in, 251
ease-in-out, 251
ease-out, 251
ease-out-in, 251

edycja wewnątrz strony, 55
edytowanie danych na stronie, 74, 80
ekran
 orientacja, 103
 rozdzielczość, 103
 szerokość, 103
 wysokość, 103
email, 59
embed, 143
Embedded OpenType, 187
Embedded Open-Type, 185
EOT, 185, 187
errata, 15
even, 86
event, 231
ExplorerCanvas, 131, 139

F

F, 33
fetchNotes, 211
fill, 129
fillText, 128
film wideo, 145
filmy dla dorosłych, 159
Firefox, 27, 33, 44, 60, 69, 91, 93, 105,
 115, 146, 148, 150, 154, 165, 167,
 171, 175, 179, 200, 214, 217, 220,
 225, 231, 241, 255, 263
Flash, 143, 155, 177, 238
Flash Player, 146, 147, 148
Flash Socket Policy, 239
Flowplayer, 155
focus, 77
font, 25
Font API†, 185
FontSquirrel, 186, 187
footer, 33, 37, 38
for, 57
form, 51
format czcionki, 186

formularz, 56, 66
frame, 25
frameset, 25
funkcje czasu, 250

G

geolocation, 241
Geolocation, 193
geolokalizacja, 193, 222, 241
getAttribute, 33
getContext, 125
getScript, 232, 244
gniazdka internetowe, 193, 222, 233
gniazdka sieciowe, 19
Google Chrome, 33, 147
gradient, 21, 130, 164, 175
Gradient, 181
grafika wektorowa, 124

H

H.264, 145
h1, 36
h2, 36
h3, 36
handheld, 104
hasBorderRadius, 168
head, 26, 228
header, 110
heading, 112
height, 26
hidden, 119
historia przeglądarki, 222
hover, 83, 249, 260
href, 48
hspace, 26
HTML5, 12, 17, 23
HTML5Shiv, 46

I

id, 32, 37, 57
ID, 36
IE, 33
IE-CSS3, 91
iframe, 26, 226, 229
Illustrator, 177
img, 26, 112, 123
implementowanie własnych pluginów, 72
indeksowana baza danych, 248
Indexed Database API, 263
IndexedDB, 205
Inkscape, 177
input type
 autofocus, 55
 color, 55
 date, 55
 email, 54, 55
 number, 55
 placeholder, 55
 range, 54
 serach, 54
 tel, 54
 text, 55
 url, 54
insertid, 212
insertNote, 212
Internet Explorer, 23, 33, 36, 44, 46, 69,
 77, 90, 91, 93, 94, 101, 105, 115,
 131, 146, 147, 151, 154, 165, 167,
 171, 179, 187, 200, 214, 217, 225,
 231, 254, 255
Internetowe aplikacje offline, 196
internetowe bazy danych SQL, 193, 196
iOS, 33, 146, 217
iTunes Store, 147

J

JAWS, 108
jQuery, 49, 72, 91, 95, 101, 105, 118,
 134, 135, 168, 170, 218, 225, 228,
 232, 261
jQuery UI, 61

K

kanwa, 125
 zapisanie stanu, 129
kanwy 3D, 248
kaskadowy arkusz stylów, 42, 94
Keith Clark, 91
klient czatu, 233
kody źródłowe, 15
kolor, 60, 61
Kolor, 55
kolor
 linii, 130
 tła, 170, 250
 wypełnienia, 130
komentarz warunkowy, 44
komunikacja
 między dokumentami, 193
 między domenami, 193
 w czasie rzeczywistym, 233
kontener, 148
 MP4, 148
 OGG, 148
 WebM, 148
kontrolka
 kalendarza, 59, 63
 wyszukiwania, 110
konwertowanie czcionki, 186

L

landmark roles, 109
last-child, 88
Leslie Michaela Orchard, 260
li, 57
Liczba, 55
linear-gradient, 164
link, 26
list, 112
lista nieuporządkowana, 32, 38
lista odniesień, 112
lista uporządkowana, 57
listitem, 112
litery
 kolor, 85
load_settings, 199
loadNote, 211
localStorage, 194, 195, 197, 198, 201,
 202, 219
lokalizacja punktu dostępowego, 241
lokalizacja użytkownika, 242
longdesc, 26

M

main, 110
map, 135
margines ostatniego akapitu, 88
math, 112
Matrix, 180
max-width, 105
media queries, 103
meter, 33, 45
metoda, 129
migracja do HTML5, 26
min-width, 105
Mobile Safari, 33, 56, 194
Modernizr, 63
month, 55
Mozilla, 147

mp3, 27
MP3, 148
MPEG, 145

N

nagłówek, 36
 sekcji, 33
 strony, 33
 sekcji strony, 112
 wiersza w tabeli, 112
naprzemiennie pokolorowane wiersze, 85
nav, 33, 42, 111
navigation, 109, 110, 111
nawigacja, 33, 37, 110
noConflict, 54
noframes, 25
notatka
 odczytanie, 205
 usunięcie, 205
 utworzenie, 205
note, 112
nth-child, 86
nth-last-child, 89
nth-of-type, 85
null, 101
number, 59

O

O, 33
object, 26, 143
oblanie tekstem, 43
obracanie elementów, 178
obrazek, 112
obrazki wektorowe, 24
obrót, 164
odczytanie zmienionego obszaru, 118
odd, 86
odstęp między komórkami tabeli, 84
offline, 217

ogg, 27
OGG, 148
okno odbiera komunikat, 231
okno wywoływane, 47
on drag, 257
onafterprint, 94
onbeforeprint, 94
onclick, 47, 48
onclose, 236, 237
ondragend, 257
ondragenter, 257, 259
ondragleave, 257, 259
ondragover, 257
ondragstart, 257
ondrop, 257
onmessage, 231, 236, 254
onopen, 236
OpenType, 187
Opera, 33, 57, 58, 59, 60, 69, 91, 105,
146, 147, 148, 214
opis obrazka, 112
originalement, 258
originalEvent, 231
overflow, 235
OVG, 187

P

p contenteditable, 55
pasek boczny, 40, 42
szerokość, 44
pasek nawigacyjny
poziomy, 43
pasek postępu, 45
pattern, 265
placeholder, 67
play, 159
plik manifestu, 217
plugin dostępny dla innych
programistów, 72

początek strony, 110
podwodne patenty, 146
podział tekstu na kolumny, 102
Pole adresu
poczty elektronicznej, 54
URL, 54
Pole daty, 55
pole dostępne do edycji, 75
pole
numeru telefonu, 54
pokrętła, 58
wyszukiwania, 54
polite, 118
połączenie stanowe, 233
połączenie z serwerem, 236
popup, 49
window, 47
POST, 56, 77
postęp wartości w polu, 58
postęp względem wartości docelowej, 33
postMessage, 228, 232, 253
posts, 43
powiadamanie o zmianach w treści
strony, 118
praca bez dostępu do sieci, 193
prawidłowość danych wejściowych, 264
prawo do rozpowszechniania, 185
użytkowania, 185
presentation, 112
preventDefault, 49
prezentowanie pomocy na temat aplikacji,
47
print, 94
profile, 26
progress, 33, 45
Prototype, 91
przechowywanie danych na komputerach
klientów, 197
przechowywanie danych w internecie, 193

przeciąganie plików, 260
przeciągnij i upuść, 248, 255
przezroczystość, 164, 181
pseudoklasa, 83
punkt początkowy rysunku, 129
pushState, 224

Q

Quirksmode, 201

R

ramka boczna, 111
ramki, 25
range, 57
receiveMessage, 232
redistribution rights, 185
rel, 47
relative, 137
Remy Sharp, 46
required, 264
restore, 129
rgb, 178
RGB, 164
rgba, 178, 181
RGraph, 133, 134, 139
 Bar, 134
rodzaj czcionki, 128
role, 108, 109, 110
 obszarów, 109
 struktury dokumentu, 112
rotate, 164
row, 112
rowheader, 112
Ruby, 239
rysowanie linii, 128

S

s, 25
S, 33
Safari, 27, 33, 44, 59, 60, 62, 69, 91,
 93, 105, 115, 146, 148, 150, 154,
 165, 167, 175, 179, 200, 214, 217,
 225, 231, 237, 241, 255
sample, 153
save, 129
Scalable Vector Graphics, 187
screen, 94
screencast, 150
scrolling, 26
search, 110, 113
section, 33, 39, 40, 42, 207
sekcja, 39, 40
selektory, 20
send, 237
serwer gniazdek internetowych, 240
sessionStorage, 194, 195, 202
setData, 258
showLocation, 242, 244
showPlaceholder, 72
sIFR, 185
source, 156
span, 77
spis treści, 112
step, 58
stopka, 37, 38
 sekcji, 33
 strony, 33
 styl, 44
stos czcionek, 189
strike, 25
stroke, 129
struktura dokumentu, 112
strumieniowe przesyłanie plików wideo,
 158

styl, 42
 czcionki, 43
style.css, 42
submit, 237
suwak, 54, 57, 63
SVG, 187
szerokość nagłówka, 43

T

table, 26
tablica z wartościami procentowymi, 135
target, 26
tekst w kolumnach, 97
tekst zastępczy, 55, 67
testowanie kodu, 14
text, 26, 56, 61
Theora, 145, 146
time, 55
title, 212
tło
 kolor, 85
Tony Amoya, 170
transform, 164
transformacje CSS3, 248
transition-timing-function, 250
treść
 główna, 111
 wtracona, 33
TrueType, 187
tryb standardowy, 36
tt, 25
TTF, 187
type, 62
Typekit, 185

U

u, 25
UI, 261
ukrywanie obszarów, 118
ul, 26

ułatwienie dla niewidomych, 80
update, 213
updateNote, 212
URL, 60
urządzenia przenośne, 103
usage rights, 185
uzupełnianie automatyczne, 68
użytkownik
 niesłyszący, 160
 niewidomy, 160

V

valign, 26
value, 72
video, 24, 141, 156
visible, 119
vlink, 26
VoiceOver, 108
Vorbis, 148
VP8, 145, 147, 148
vspace, 26

W

W3C Validator, 26
wartość należąca do zakresu, 33
wątki robocze, 248, 253
Web Open Font, 187
web sockets, 19
Web Sockets, 193, 233
Web SQL Databases, 193, 194, 205,
 219
Web SQL Storage, 204, 220
Web Storage, 193, 194
Web Workers, 253
WebGL, 262
web-socket-js, 238
week, 55
weryfikacja poprawności formularzy, 248
WIA-ARIA, 107, 115
width, 26

wiersz komórek w tabeli, 112
window.localStorage, 198
window.onpopstate, 224, 225
window.open, 50
window.openDatabase, 208
WindowsEyes, 108
WOFF, 187
współrzędnych GPS, 241
wtrącenie, 40, 112
wykres, 132
wykres słupkowy, 137
wymiana komunikatów między dokumentami, 222
wyrażenie matematyczne, 112
wyrównywanie
 pionowe tekstu, 128
 tekstu w kolumnie, 86
wysyłanie komunikatu, 228
wyśrodkowanie treści, 43
wyświetlenie dodatkowych opcji, 47

X

Xiph.Org Foundation, 146
XMLHttpRequest, 77

Y

YUI, 61

Z

zaokrąglenie rogów, 164, 165
zapisywanie danych, 76
zapytania o media, 103
zasada DRY, 78
zawartość strony, 110, 112
znacznik semantyczny, 37, 40
znaczniki
 opisowe, 20
 samozamykające, 22
znaczniki unieważnione, 24
zwiększanie dostępności witryn, 107

Ź

źródło pliku wideo, 156

HTML5 i CSS3 Standardy przyszłości

Stało się! Choć jeszcze niedawno HTML5 i CSS3 wydawały się technologiami wciąż odległej przyszłości, ich specyfikacje już zostały zaimplementowane w takich przeglądarkach, jak Google Chrome, Safari, Firefox czy Opera. Standardy HTML5 i CSS3, mimo że jeszcze w stadium rozwoju, nie przestają wzbudzać zachwytu administratorów sieci i twórców stron WWW, a ich opublikowanie hucznie zapowiedziało nową generację aplikacji internetowych. Teraz można je łatwiej wdrażać i użytkować, a także wyjść naprzeciw potrzebom użytkowników. Mało? Język HTML5 wprowadza także nowe elementy służące do definiowania struktury witryny oraz osadzania w niej treści, a CSS3 udostępni zaawansowane selektory, rozszerzenia graficzne oraz zapewnia lepszą obsługę czcionek. W świecie internetu, gdzie pod względem rozprzestrzenienia się nowości trzy miesiące to cała epoka, już jutro ta technologia będzie obecna wszędzie. Nie trzeba więc nikogo przekonywać, że aby ten postęp bez zadyszki dogonić, trzeba już dziś... wziąć się za lekturę!

Przed Tobą doskonały podręcznik, w którym przedstawione zostały wszystkie dostępne sposoby korzystania z nowych możliwości standardów HTML5 i CSS3, także tych, które nie są jeszcze obsługiwane przez część przeglądarek. Każdy rozdział skupia się na określonej grupie problemów i zawiera listę niezbędnych znaczników, funkcji lub mechanizmów. Na początek dokładnie poznasz standardy HTML5 i CSS3, nowe atrybuty i znaczniki strukturalne. Przeczytasz co nieco na temat formularzy i odkryjesz, jak wykorzystać niektóre nowe pola i funkcje. Poznasz rewolucję wprowadzoną w CSS3 – nowe selektory, cienie, gradienty, transformacje i sposoby korzystania z czcionek. Dowiesz się, jak HTML5 obsługuje dane audio i wideo, oraz nauczysz się używać kanw do rysowania rozmaitych kształtów. A potem stworzysz przykładowe aplikacje działające po stronie klienta, a także zobaczysz, jak HTML5 umożliwia przesyłanie komunikatów i danych między domenami.

- Wprowadzenie do HTML5 i CSS3
- Nowe atrybuty i znaczniki strukturalne
- Tworzenie przyjaznych formularzy internetowych
- Tworzenie lepszych interfejsów użytkownika z użyciem CSS3
- Zwiększanie dostępności witryn internetowych
- Rysowanie na kanwach
- Osadzanie danych audio, wideo oraz grafiki wektorowej
- Wykorzystanie cieni, gradientów, transformacji i czcionek
- Przetwarzanie danych po stronie klienta
- Korzystanie z interfejsów API

Jesteś gotowy? Doskonale! Zacznij już dziś poznawać standardy przyszłości!

Pragmatic
Bookshelf

helion.pl
księgarnia
internetowa

Nr katalogowy: **5874**



Księgarnia internetowa:
<http://helion.pl>



Zamówienia telefoniczne:
0 801 339900
0 601 339900



Helion

Sprawdź najnowsze promocje:

🔗 <http://helion.pl/promocje>

Książki najchętniej czytane:

🔗 <http://helion.pl/bestsellery>

Zamów informacje o nowościach:

🔗 <http://helion.pl/nowosci>

Helion SA

ul. Kościuszki 1c, 44-100 Gliwice
tel.: 32 230 98 63
e-mail: helion@helion.pl
<http://helion.pl>

Cena 49,00 zł

ISBN 978-83-246-3028-8



9 788324 630288

Informatyka w najlepszym wydaniu