Craig Sharkie, Andrew Fisher

Responsywne strony WWW

TECHNOLOGIA NA START!

Strony WWW
na każde urządzenie!

Helion

Tytuł oryginału: Jump Start Responsive Web Design

Tłumaczenie: Piotr Rajca
ISBN: 978-83-246-9237-8

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: helion@helion.pl
WWW: http://helion.pl (księgarnia internetowa, katalog książek)

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:
ftp://ftp.helion.pl/przyklady/reswww.zip

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie/reswww
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

Spis treści

Rozdział 4

Zrozumieć zapytania medialne

Zapytania medialne są trzecim filarem metodologii RWD i stanowią rozszerzenie języka HTML5 pozwalające, by na dostarczanie arkuszy CSS do konkretnego urządzenia miały wpływ cechy jego ekranu, zgodnie z definicją zawartą w specyfikacji modułu CSS3[1]. Innymi słowy: istnieje możliwość wykrycia, że strona jest wyświetlana na urządzeniu przenośnym z ekranem o szerokości 320 pikseli i działającym w układzie poziomym, i dostarczenia do niej innego arkusza stylów niż do tej samej strony wyświetlanej na komputerze stacjonarnym z ekranem o obszarze prezentacyjnym o szerokości 1024 pikseli. Tradycyjnie różnice w stylach ograniczałyby się jedynie do innego układu, tła oraz obrazów, jednak w rzeczywistości istnieje możliwość dostarczenia całkowicie odrębnego zestawu stylów.

Arkusze stylów określane na podstawie zapytań medialnych mogą być podawane na trzy sposoby, podobnie jak w przypadku typów mediów. Po pierwsze, przy użyciu elementu `link` języków HTML i XHTML:

```
<link rel="stylesheet" type="text/css" media="all and
   (color)" href="/style.css">
```

Po drugie, przy użyciu kodu XML:

```
<?xml-stylesheet media="all and (color)" rel="stylesheet"
   href="/style.css" ?>
```

Ostatnim sposobem jest użycie reguł `@import` w arkuszach stylów:

```
@import url("/style.css") all and (color);
```

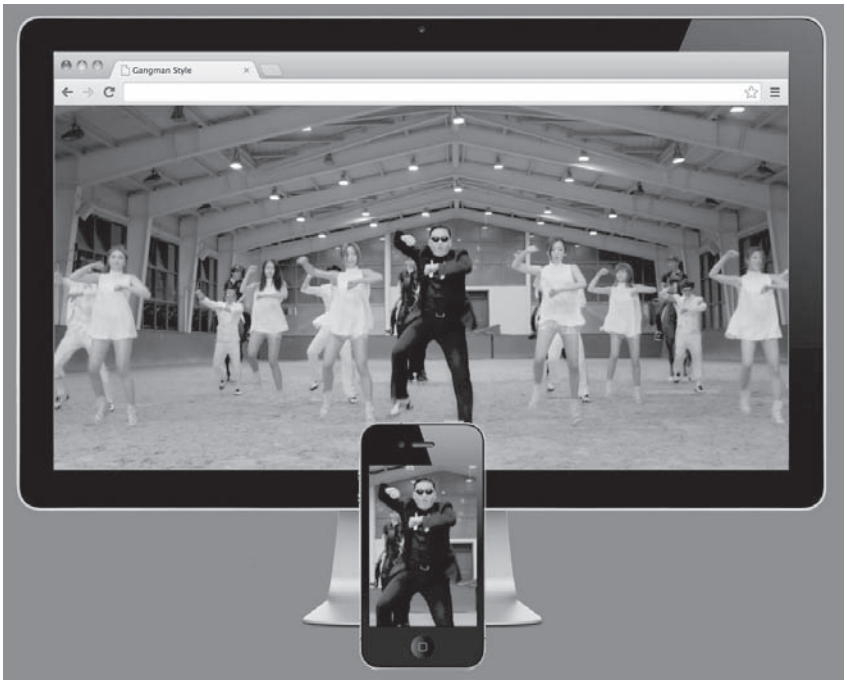
[1] *http://www.w3.org/TR/css3-mediaqueries/*.

lub reguł `@media`:

```
@media all and (color) { /* jeden lub kilka zestawów reguł... */ }
```

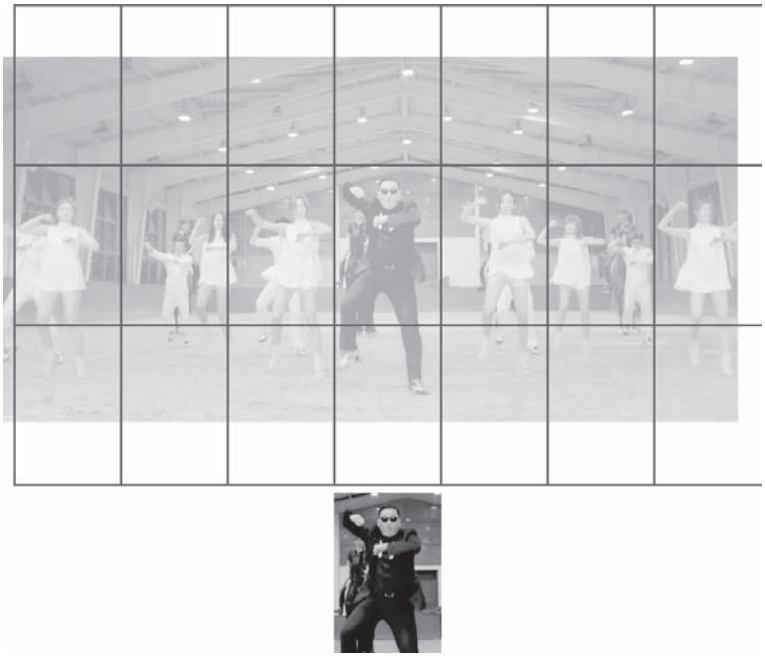
Także w tym przypadku metodologia RWD udostępnia nam narzędzia do tworzenia prostych rozwiązań umożliwiających budowanie niezawodnych układów. Zapewnia również możliwość skoncentrowania uwagi na treści i na zachęceniu użytkowników do ponownego skorzystania z aplikacji.

Jak wyraźnie widać na rysunku 4.1, nowoczesne urządzenia mają bardzo różne możliwości oraz wymagania związane z wykorzystywanymi materiałami i stylami. Na szczęście składnia zapytań medialnych jest bardzo prosta, jak również powszechnie obsługiwana i stosowana — w rzeczywistości pojawiła się już w każdym rozdziale tej książki!



Rysunek 4.1. Różne obrazy na docelowych urządzeniach

Zazwyczaj staramy się ograniczać style CSS dostarczane do każdego z urządzeń docelowych. Zaczynając od stylów używanych zawsze, dochodzimy do tych, które są przeznaczone dla konkretnych urządzeń. Niemniej jednak nie tylko same style są dostosowywane do urządzeń — to samo dotyczy także materiałów używanych i dostarczanych przez te style. Rysunek 4.2 pokazuje, że jeśli do naszego urządzenia z ekranem o szerokości 320 pikseli (pokazanego na rysunku 4.1) przekażemy tło o szerokości 1920 pikseli, to wyślemy do niego obraz 14 razy większy niż to konieczne. Takiego niepotrzebnego marnowania przepustowości i obciążenia aplikacji można całkiem łatwo uniknąć.



Rysunek 4.2. Nasz mały obraz mieści się w dużym obrazie około 14 razy

Przyjrzyjmy się zapytaniu medialnemu stworzonemu z myślą o telefonie iPhone 5:

```
@media only screen and (min-device-width: 640px) and
   (max-device-width: 1136px) and
   (-webkit-min-device-pixel-ratio: 2) {
      /* tylko dla iPhone'a 5... przynajmniej na razie */
}
```

Teraz przynajmniej składnia ta nie wygląda już zupełnie obco. Cechę `device-width` widzieliśmy już w elemencie `meta viewport`, a `device-pixel-ratio` — w adaptacyjnych obrazach tworzonych przy użyciu elementu `picture`. Gdyby tego było mało, zapytania medialne są kuzynami typów mediów wprowadzonych w maju 1998 roku jako jeden z elementów rekomendacji CSS 2.0[2].

Poznawanie możliwości zapytań medialnych

Zapytania medialne zapewniają całkiem sporo możliwości, nawet jeśli będziemy się posługiwali wyłącznie cechą `device-width`. Trzeba jednak pamiętać o tym, że można sprawdzać także inne cechy, a nie jedynie szerokość ekranu urządzenia. Obecnie specyfikacja[3] wymienia

[2] *http://www.w3.org/TR/2008/REC-CSS2-20080411/.*

[3] *http://www.w3.org/TR/css3-mediaqueries/#contents.*

13 takich cech: `width`, `height`, `device-width`, `device-height`, `orientation`, `aspect-ratio`, `device-aspect-ratio`, `color`, `color-index`, `monochrome`, `resolution`, `scan` oraz `grid`. Mogą one, z wyjątkiem `orientation`, `scan` oraz `grid`, być także poprzedzane prefiksami `min-` oraz `max-`. Poziom kontroli, jaki zapewniają nam zapytania medialne, jest inspirujący, choć jednocześnie budzi respekt!

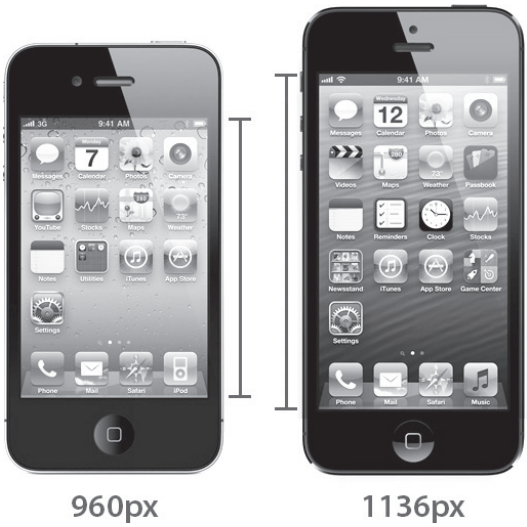
Jeśli wciąż jesteśmy pełni szacunku dla liczby możliwości, jakie na nas czekają, to musimy zdać sobie sprawę z tego, że zapytania medialne są kontrolowane przez CSS, a twórcy przeglądarek mogą dodawać do tej listy nowe możliwości, poprzedzając je odpowiednimi prefiksami — przykładem jest cecha `-webkit-device-min-pixel-ratio` stanowiąca dodatek wprowadzony przez firmę Apple i umożliwiająca tworzenie stylów przeznaczonych dla urządzeń z ekranami Retina. Trzeba także pamiętać, że oprócz prefiksu `-webkit-` jest także wiele innych, których można używać. Oto kilka przykładów:

```
@media only screen and (-webkit-min-device-pixel-ratio: 2)
      and (min-width: 320px),
   only screen and (min—moz-device-pixel-ratio: 2)
      and (min-width: 320px),
   only screen and (-o-min-device-pixel-ratio: 2/1)
      and (min-width: 320px),
   only screen and (min-device-pixel-ratio: 2)
      and (min-width: 320px) {
         /* style dla małych ekranów Retina lub AMOLED */
      }
```

Podobnie jak w przypadku standardowych prefiksów przeglądarek, także i tutaj ostatnia właściwość nie ma żadnego prefiksu — została ona podana w nadziei, że kiedyś prefiksy te nie będą potrzebne i do określenia docelowej grupy urządzeń wystarczy sama właściwość.

Dawno już minęły czasy, gdy rozwiązaniem wszelkich problemów było używanie typu `handheld`. Obecnie trzeba uwzględniać i stosować wszystkie możliwe sposoby określania urządzeń docelowych oraz uważnie planować budżet dostępny na stworzenie projektu i samej aplikacji. Zważywszy na ciągłą tendencję do stosowania przesadnie szczegółowych zapytań medialnych, analiza ta staje się coraz to bardziej złożona i trudna. Równocześnie z wprowadzeniem na rynek telefonu iPhone 5 z systemem iOS6 pojawiła się nowa grupa urządzeń, których ekrany mają dodatkowe 176 pikseli wysokości. Jednak jest raczej mało prawdopodobne, byśmy starali się tworzyć style przeznaczone dla tych urządzeń, stosując zapytania `max-device-height: 960px`. W takim przypadku pominęlibyśmy bowiem wszystkie telefony iPhone 5S. Przyjrzyjmy się przykładowi przedstawionemu na rysunku 4.3.

To prawda, że dziś możemy odróżniać telefony iPhone 5 od ich poprzedników, używając w tym celu cechy `device-height`, jednak trzeba by też uwzględnić zagrożenie, że stopka, tak skrupulatnie umieszczona na samym dole ekranu iPhone'a 4S i wcześniejszych modeli tego telefonu, teraz znajdzie się 176 pikseli powyżej dolnej krawędzi ekranu. Oprócz tego skoro iPhone 5 zyskał dodatkowe 176 pikseli wysokości ekranu, to czego możemy oczekiwać po jego kolejnej generacji? Albo po jej następcach? Trzeba wciąż pamiętać o tym, co jest



Rysunek 4.3. Porównanie wielkości ekranów telefonów iPhone 4 oraz iPhone 5

naszym celem — zaufanie, że nasze style będą działać na wszystkich urządzeniach: zarówno tych obecnych, jak i przyszłych, i to bez konieczności przepisywania kodu dla każdej nowej wielkości ekranów.

W takim razie warto rzucić okiem na składnię używaną do wybierania różnych cech. Lista zapytań medialnych jest łańcuchem znaków, którego fragmenty są od siebie oddzielone przecinkami i który zawiera serię zapytań medialnych precyzyjnie określających cechy docelowego urządzenia:

```
<link rel="stylesheet" media="screen and (color),
   projection and (color)" rel="stylesheet"
   href="example.css">
```

Jeśli przynajmniej jedno zapytanie umieszczone na tej liście będzie prawdziwe, to przyjmuje się, że cała lista będzie prawdziwa, a wszystkie spełnione cechy zostaną uwzględnione. Na przykład jeśli na liście takiej jak `"screen and (color), projection and (color)"` zapytanie `"screen and (color)"` będzie prawdziwe, to także cała lista zostanie uznana za prawdziwą.

Przecinki oddzielające poszczególne zapytania medialne odpowiadają logicznej alternatywie (`OR`), a każde zapytanie może używać słowa kluczowego `AND` reprezentującego logiczną koniunkcję. Aby zapytanie z logiczną alternatywą zostało spełnione, musi być dostępna którakolwiek z wymienionych cech, natomiast w przypadku użycia logicznej koniunkcji muszą być dostępne wszystkie wymienione cechy. Poprzednie zapytanie zadziała w sytuacji, gdy aplikacja zostanie uruchomiona na ekranie lub projektorze zdolnym do wyświetlania obrazów w kolorze. Jeśli używane urządzenie spełnia te kryteria, to zostanie zastosowany arkusz stylów podany w elemencie `link`.

Aby odrzucić wybraną cechę, należy użyć słowa kluczowego `not`. Na przykład poniższe zapytanie zadziała na telefonach iPhone, lecz jedynie na tych, które NIE są wyposażone w ekrany Retina; innymi słowy: zadziała na iPhonie 3, lecz nie na iPhonie 5:

```
@media only screen (min-device-width: 640px) and
   not (-webkit-min-device-pixel-ratio: 2) {
      /* style dla urządzeń z ekranami o niskiej gęstości */
}
```

Ogólnie rzecz biorąc, składnia zapytań medialnych jest bardzo prosta, czytelna i można ją łatwo zrozumieć, trzeba tylko uważać na te niejawne logiczne alternatywy. Gdyby W3C oraz WHATWG zdecydowały się na przyjęcie elementu `picture`, a nie atrybutu `srcset`, to dodatkowe możliwości zastosowania, jakie zyskałyby zapytania medialne, sprawiłyby, że ich poznawanie byłoby jeszcze bardziej warte zachodu.

Obsługa cech

Z powodzeniem można wiele osiągnąć, stosując wyłącznie zapytania medialne testujące cechy związane z wymiarami ekranów — `width`, `height`, `device-width` oraz `device-height`. Zapewniają one dwojakie korzyści: są łatwe do zrozumienia i powszechnie obsługiwane przez przeglądarki. Oczywiście łatwo je zrozumieć, jeśli tylko będziemy pamiętać, że `width` określa szerokość przeglądarki, a `device-width` — szerokość ekranu zgłaszaną przez urządzenie (rysunek 4.4).

Rysunek 4.4. Porównanie cech width oraz device-width

Przyjrzyjmy się także pozostałym dostępnym cechom, aby zorientować się, jak można by ich używać.

orientation	określenie układu, w jakim działa urządzenie, pozwala na podanie wartości `portrait` (układ pionowy) lub `landscape` (układ poziomy)
aspect-ratio	akceptuje takie wartości, jak 16:9 lub 4:3
resolution	określa rozdzielczość wyrażoną jako liczba punktów na cal (DPI — *Dots Per Inch*) lub liczba punktów na centymetr (DPCM — *Dots Per Centimetre*)
scan	typ ekranów pozwalający na stosowanie stylów przeznaczonych dla odbiorników telewizyjnych używających skanowania progresywnego
grid	odpowiada ekranom Teletype lub urządzeniom używającym tylko jednej czcionki
monochrome	sprawdza liczbę pikseli na piksel w monochromatycznym buforze ramki; w przypadku wartości 0 warunek jest uznawany za niespełniony
color	sprawdza liczbę bitów reprezentujących kolor; w przypadku wartości 0 warunek jest uznawany za niespełniony
color-index	sprawdza liczbę komórek w tablicy kolorów używanej na danym urządzeniu; w przypadku wartości 0 warunek jest uznawany za niespełniony

Aby dokładnie opanować te cechy, trzeba szczegółowo poznać proporcje ekranu urządzenia, jego rozdzielczość, układ, w jakim w tej chwili działa, oraz to, czy jest w stanie wyświetlać kolory. Są to zagadnienia doskonale znane osobom, które tworzyły grafiki na potrzeby stron WWW. Także w tym przypadku znów znaleźliśmy się na dobrze znanym gruncie, gdyż specyfikacja HTML5 posługuje się tymi wszystkimi terminami[4], opisując standard języka. Dzięki temu krzywa trudności nauki zapytań medialnych jest znacznie łagodniejsza.

Co się stało z typami mediów?

Koniecznie trzeba pamiętać, że choć w tej książce interesują nas zagadnienia tworzenia responsywnych stron WWW, to jednak są także inne czynniki, które mogą (i powinny) mieć wpływ na nasze aplikacje. Sam fakt stosowania metodologii RWD nie oznacza, że mamy nie zwracać uwagi na zapewnianie wysokiej dostępności stron, na tworzenie różnych wersji językowych witryny czy też zapominać o sprawdzeniu, jak będzie się ona prezentować na ekranach, które nie są zazwyczaj używane do przeglądania stron WWW.

W efekcie wykorzystywane zapytania medialne będą zwykle używać typu `screen`, choć trzeba pamiętać, że są także inne możliwości[5]!

Wprawdzie obsługa poszczególnych cech może być różna, ale nawet te, które nie są związane z wymiarami, mogą być z powodzeniem używane, gdyż stosowana terminologia jest oczywista w przemyśle, do którego jest skierowana. Nie szkodzi, że cechy `scan` oraz `grid`

[4] *https://developer.mozilla.org/en/docs/CSS/Media_queries*.

[5] *http://dev.w3.org/csswg/css3-mediaqueries/#background*.

mogą nie być dobrze znane projektantom tworzącym strony przeznaczone dla komputerów stacjonarnych, są one bowiem dedykowane dla innych urządzeń i będą doskonale znane osobom tworzącym rozwiązania zaprojektowane właśnie dla nich. Na przykład jeśli nasza praca jest związana z tworzeniem aplikacji prezentowanych na odbiornikach telewizyjnych, to będziemy doskonale wiedzieli, czego dotyczy cecha `scan` oraz czym jest skanowanie progresywne[6].

Używanie zapytań medialnych oraz list zapytań jest najlepszym sposobem zrozumienia, jak należy się nimi posługiwać, zatem przejdźmy do ćwiczeń praktycznych.

Zapytania medialne w działaniu

Strona Web Directions South (WDS), która wielokrotnie pojawiała się jako przykład w tej książce, od samego początku istnienia używała pewnych zapytań medialnych. Rysunek 4.5 przedstawia nagłówkowe grafiki, używane w przeglądarkach mających więcej oraz mniej niż 800 pikseli szerokości.

Rysunek 4.5. Obrazy o wymiarach 1565×690 oraz 800×650 pikseli udostępniane przy wykorzystaniu zapytań medialnych

Grafiki te gwarantują, że witryna będzie mieć spójny wygląd, a co ważniejsze, że przeglądarki będą pobierać możliwie jak najmniej danych. A co jest naszym celem? Spójrzmy na rysunek 4.6.

[6] *http://pl.wikipedia.org/wiki/Skanowanie_progresywne*.

Rysunek 4.6. Nasz cel: stopka witryny WDS w przeglądarkach o szerokości 1440, 960, 800 oraz 480 pikseli

Na witrynie Web Directions South zapytania medialne są stosowane przy wykorzystaniu kombinacji elementów `link` oraz arkuszy CSS. Rozwiązanie to zapewnia, że pierwszą lokalizacją, w której rozpoczyna się określanie stylów CSS, jest sekcja `head` dokumentu, czyli miejsce, gdzie tradycyjnie już są umieszczane odwołania do plików CSS. Na razie wszystkie te odwołania umieścimy w komentarzach:

rozdzial04/wds/index.html (*fragment*)

```
<!-- <link rel="stylesheet" media="all and (max-width: 800px)"
   href="stylesheets/799max.min.css"> -->
<!-- <link rel="stylesheet" media="all and (min-width: 800px)"
   href="stylesheets/800plus.min.css"> -->
<!-- <link rel="stylesheet" media="all and (min-width: 800px)"
   href="stylesheets/panels.min.css"> -->
```

Jednym z zagadnień, o których należy pamiętać, stosując tę metodę, jest to, że przeglądarka będzie pobierać nawet te arkusze stylów, w których warunki określane przez zapytania medialne nie zostały spełnione[7]. Arkusze te nie zostaną jednak zastosowane, a przeglądarka nie pobierze żadnych zasobów, które są w nich używane. Niemniej jednak same arkusze stylów zostaną pobrane, będą zatem mieć wpływ na wydajność działania aplikacji oraz zużycie przepustowości.

[7] *http://scottjehl.github.com/CSS-Download-Tests/*.

Mamy właśnie zamiar usunąć z naszej przykładowej strony wszystkie używane na niej arkusze stylów — jest to rozwiązanie dosyć radykalne, jednak ułatwi nam zmianę struktury stylów. Skoro zapytania medialne używane na produkcyjnej witrynie Web Directions South są umieszczone w nagłówku strony, a dla przeglądarek o mniejszej szerokości wykorzystywana jest po prostu właściwość `display: none`, skoncentrujemy się na sekcji sponsorów umieszczonej w stopce strony. Później wrócimy do usuniętych arkuszy CSS, by odszukać początkowe style określające postać sekcji sponsorów i nie tylko. Chcemy jedynie zmienić strukturę stylów, a nie opracowywać je od nowa.

rozdzial04/wds/index.html (*fragment*)

```
<link rel="stylesheet" href="stylesheets/reset.css">
<link rel="stylesheet" href="stylesheets/sheet.css">
<link rel="stylesheet" href="stylesheets/query.css">
```

Zapytania medialne wykorzystamy w pliku *query.css*, w pliku z domyślnymi stylami *sheet.css*, jak również w ostatnim arkuszu stylów *reset.css*. Dzięki temu wprowadzimy możliwie najmniej zmian w pliku HTML i spróbujemy ułatwić utrzymanie witryny dzięki zastosowaniu sensownych nazw plików. Więcej informacji na temat ostatniego z używanych plików CSS można znaleźć w rozdziale 6., w punkcie „Neutralizacja i normalizacja”, dlatego też w tym rozdziale skoncentrujemy się na stylach umieszczonych w dwóch pierwszych plikach.

Jak wiadomo, połączenie zapytań medialnych z domyślnym stylem oznacza, że w efekcie zupełnie za darmo uzyskujemy w naszym domyślnym stylu dodatkowy punkt graniczny (ang. *breakpoint*). Używamy dwunastu stylów wykorzystujących w selektorze identyfikator `sponsors`, na szczęście w celu zastosowania zapytań medialnych będziemy musieli zmodyfikować tylko dwa z nich.

```
#sponsors {
   margin: 2em auto 0 auto;
   width: 800px;
}

#sponsors ul li {
   background-color: #FFF;
   display: inline-block;
   float: left;
   height: auto;
   margin: 0 0 1.7em 0;
   min-height: 120px;
   vertical-align: top;
   width: 30%;
}
```

W tych dwóch stylach interesować nas będą wyłącznie deklaracje związane z właściwościami `width` oraz `margin`, gdyż to właśnie one mają wyznaczać punkty graniczne naszego responsywnego układu:

```
#sponsors {
   margin: 2em auto 0 auto;
   width: 800px;
}

#sponsors ul li {
   ...
   margin: 0 0 1.7em 0;
   ...
   width: 30%;
}
```

W pierwszej kolejności zmienimy dotychczasowe style określające postać elementów stopki, zgodnie z założeniem, by najpierw tworzyć rozwiązanie przeznaczone dla urządzeń mobilnych. Efekt tych zmian przedstawia rysunek 4.7.

Rysunek 4.7. Stopka strony wyświetlona w przeglądarce o szerokości 480 pikseli

rozdzial04/wds/stylesheets/sheet.css (*fragment*)

```
#sponsors {
   margin: 2em auto 0 auto;
   width: 480px;
}
#sponsors ul li {
   ...
   margin: 0 5px 1.7em 5px;
   ...
   width: 470px;
}
```

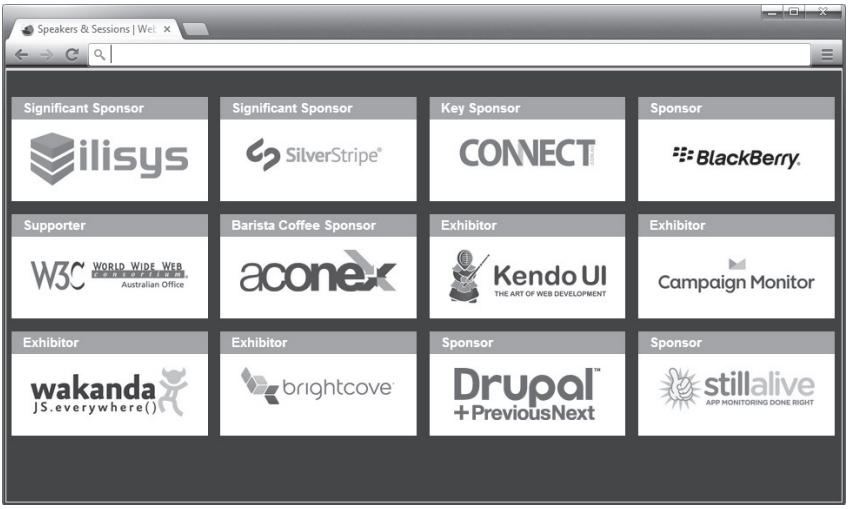
Jesteśmy na dobrej drodze. Naszym pierwszym zadaniem jest określenie elementu `#sponsors` w taki sposób, by w razie wyświetlenia strony na urządzeniu mobilnym zawsze miał on 480

pikseli szerokości. Właśnie z tego powodu podaliśmy stałe wymiary szerokości oraz marginesów elementów `li`. Zastosujemy responsywne rozwiązanie, określając punkty graniczne, jednak teraz uzyskamy pewność, że w naszym domyślnym stylu elementy zawsze będą miały stałą szerokość, a stała szerokość oznacza, że przeglądarka pozycjonując takie elementy, będzie miała łatwiejsze zadanie. Po wprowadzeniu powyższych zmian w pliku *sheet.css* zajmiemy się dodaniem do arkusza *query.css* pierwszego punktu granicznego:

rozdzial04/wds/stylesheets/query.css (*fragment*)

```
@media only screen and (min-width: 480px) and
   (max-width: 960px) {
      #sponsors {
         max-width: 960px;
         width: 100%;
      }
      #sponsors ul li {
         margin: 0 0.4% 1em 0.8%;
         width: 31.5%;
      }
}
```

Ze względu na to, że do określenia docelowej grupy urządzeń w zapytaniu medialnym użyliśmy składni `@media`, dwa selektory zostały odpowiednio wcięte i zagnieżdżone wewnątrz reguły `@media`. Lista zapytań rozpoczyna się od `only screen and`, a następnie określa cechy docelowej grupy urządzeń: `min-width: 480px` oraz `max-width: 960px`. Uzyskane efekty przedstawia rysunek 4.8.



Rysunek 4.8. Stopka strony wyświetlona w przeglądarce o szerokości 960 pikseli

To jest nasz średni punkt graniczny. Będziemy mieli rację, sądząc, że można by pominąć określenie typu mediów oraz cechy `min-width` i pomimo to uzyskać takie same rezultaty, jednak podanie obu tych cech znacznie ułatwi nam zrozumienie działania punktu granicz-

nego w przyszłości, kiedy będziemy chcieli poprawić lub zmodyfikować style. Powyższą listę zapytań można by skrócić do następującej postaci:

```
@media (max-width: 960px) {
   ...
}
```

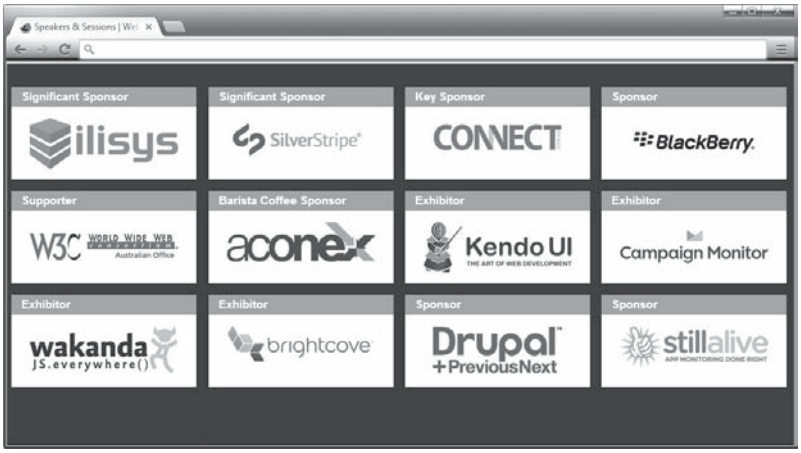
Jednak zaleta, jaką jest skrócona postać zapisu, staje się wadą, gdy spojrzymy na powyższy kod pod kątem łatwości utrzymania i modyfikacji. Nieco bardziej rozbudowana forma jest jednocześnie bardziej korzystna:

```
#sponsors {
   max-width: 960px;
   width: 100%;
}

#sponsors ul li {
   margin: 0 0.4% 1em 0.8%;
   width: 31.5%;
}
```

Same deklaracje są zwyczajnym, niezawodnym kodem CSS. Nie ma potrzeby modyfikowania właściwości `margin` w elemencie zawierającym logo sponsorów, dlatego też w arkuszu nie ma deklaracji z tą właściwością; zamiast niej dodaliśmy deklarację `max-width: 960px`, a właściwości `width` przypisaliśmy wartość `100%`. Jak wiemy z wcześniejszych rozważań dotyczących siatek, takie rozwiązanie sprawia, że element staje się płynny i na wszystkich ekranach o szerokości pomiędzy 480 i 960 pikseli stopka będzie zajmowała całą dostępną szerokość.

Aby wykorzystać ten fakt, zapewnimy także responsywność elementów `li` — w tym celu określimy ich szerokość, używając wartości procentowych, i trochę się z nimi pobawimy, by zmaksymalizować uzyskiwany efekt. Chcemy, by elementy były możliwie jak najszersze, a jednocześnie by nie pojawiały się błędy zaokrągleń opisane w rozdziale 2., w uwadze „Diabeł tkwi w szczegółach”. Jeśli zsumujemy szerokość elementu oraz wielkości jego poziomych marginesów, to uzyskamy 31,5 + 0,4 + 0,8 = 32,7 procent. Dysponujemy zatem „buforem bezpieczeństwa” o szerokości 0,3 procent, który umożliwi nam zniwelowanie ewentualnych różnic w zaokrągleniach w różnych przeglądarkach. Koniecznie trzeba pamiętać, że — z wyjątkiem czytelników tej książki oraz naszego zespołu — mało kto będzie próbował modyfikować szerokość okna przeglądarki, by sprawdzić, jak zmieni się wtedy układ strony. Większość osób zadowoli się wysiłkami, które podjęliśmy, by zapewnić, że wyświetlana strona będzie możliwie najlepsza.

Na przykład gdyby ktoś powiększył okno przeglądarki, nadając mu szerokość przekraczającą 960 pikseli (co pokazuje rysunek 4.9), to zostałby zastosowany ostatni z trzech określonych przez nas punktów granicznych.



Rysunek 4.9. Stopka wyświetlona w przeglądarce o szerokości 1024 pikseli

rozdzial04/wds/stylesheets/query.css (*fragment*)

```
@media only screen and (min-width: 960px) {
   #sponsors {
      max-width: 1280px;
      width: 100%;
   }
   #sponsors ul li {
      margin: 0 0.5% 1em 0.6%;
      width: 23.5%;
   }
}
```

Także w tym przypadku zastosowane style zapewniają elastyczność układu i elementów, choć pojawiły się w nich pewne korzystne różnice. Przede wszystkim zamiast nadawać elementowi `sponsors` szerokość odpowiadającą maksymalnej szerokości naszego punktu granicznego, nadaliśmy mu stałą szerokość 1280 pikseli. Oznacza to, że jeśli szerokość okna przeglądarki nie przekroczy 1280 pikseli, element `sponsors` będzie zajmował całą dostępną szerokość strony. Kiedy jednak okno przeglądarki przekroczy 1280 pikseli szerokości, to szerokość elementu `sponsors` nie będzie się dalej powiększać, a marginesy określone w pliku *sheet.css* spowodują, że zostanie on wyśrodkowany (jak pokazaliśmy na rysunku 4.10).

Druga zmiana polega na tym, że zrezygnowaliśmy z wyświetlania trzech sponsorów w jednym wierszu na rzecz wyświetlania czterech, choć wciąż korzystamy z tego samego responsywnego rozwiązania. Sumaryczna szerokość elementu wynosi 23,5 + 0,5 + 0,6 = 24,6 procent, co daje nam bufor bezpieczeństwa wielkości 0,4 procent. Nieliczni spośród użytkowników, którzy zdecydują się na zmianę szerokości okna przeglądarki, będą mieli okazję zauważyć, że nasz układ zmienia się dynamicznie od jednej kolumny z logo sponsorów, poprzez trzy kolumny aż do czterech kolumn w oknach o największej szerokości. W ten sposób wszyscy są zadowoleni — sponsorzy, gdyż przeznaczamy na prezentację ich logo maksymalny obszar ekranu, i użytkownicy, dlatego że niezależnie od szerokości okna przeglądarki, czytelnie prezentowane logo stanowią doskonały obiekt zainteresowania.

Rysunek 4.10. Stopka wyświetlona w przeglądarce o szerokości 1440 pikseli

Dzięki zdecydowaniu się na wykorzystanie kombinacji punktów granicznych oraz płynnego układu strony udało się nam zoptymalizować jej interfejs pod kątem możliwie największej liczby czynników. Udało nam się również spełnić oczekiwania zarówno sponsorów, jak i użytkowników, a zastosowane rozwiązanie korzystające z metodologii RWD zapewnia także tę zaletę, że nie musimy projektować odrębnych układów dla każdego z istniejących ekranów. Choć punkty graniczne mogą się wydawać sposobem na to, by projektanci ponownie odzyskali kontrolę nad układem na niebezpiecznym gruncie bezustannie rozwijanych urządzeń, zdecydowaliśmy się w pełni wykorzystać ich możliwości, a nie jedynie ograniczyć się do serii stałych układów strony.

Skoro dotarliśmy już do tego miejsca, to zanim nasze modyfikacje będą gotowe do wdrożenia na produkcyjnej witrynie, zostaje nam do zrobienia jeszcze jedna rzecz. Okazuje się, że choć dzięki zapytaniom medialnym nasza witryna może się podobać tak szerokiemu gronu użytkowników, to jednak istnieje pewna grupa, która jest traktowana po macoszemu. Jeśli użytkownicy wciąż korzystają z przeglądarki Internet Explorer w wersji wcześniejszej niż dziewiąta, to nie będą mogli skorzystać z efektów zastosowania zapytań medialnych, ponieważ obsługa wybranych możliwości CSS3[8] została dodana dopiero w przeglądarce IE9. Jak sobie poradzić z tym problemem?

Oczywiście możemy zachęcać naszych użytkowników do zainstalowania nowszej wersji przeglądarki, jednak gdyby to było możliwe, zapewne już by to zrobili.

Moglibyśmy uzupełnić możliwości przeglądarki, stosując rozwiązanie skryptowe, takie jak skrypt *Respond.js*[9] Scotta Jehla, zapewniający możliwość stosowania zapytań medialnych testujących minimalną i maksymalną szerokość nawet w starszych przeglądarkach. Oczywistą wadą takiego rozwiązania jest dodatkowe wydłużenie czasu pobierania stron w przeglądarkach, które i tak nie należą do najszybszych. Zważywszy na to, że biblioteka Modernizr

[8] *http://caniuse.com/css-mediaqueries*.

[9] *https://github.com/scottjehl/Respond*.

zaczynając od wersji 2.5, zrezygnowała z użycia skryptu *Respond.js*, także i my zrezygnujemy z niego na rzecz innego rozwiązania.

Nasze rozwiązanie opiera się na pomyśle, że starsze wersje przeglądarki Internet Explorer mogą działać wyłącznie na komputerach stacjonarnych i laptopach. Możemy zatem skorzystać z komentarzy warunkowych Internet Explorera, by zastosować w tych przeglądarkach układy o stałej szerokości i wykorzystać ich zalety:

rozdzial04/wds/index.html (*fragment*)

```
<!--[if lt IE 9]>
   <link rel="stylesheet" type="text/css"
      href="stylesheets/ielt9.css">
<![endif]-->
```

A to zawartość arkusza stylów:

rozdzial04/wds/stylesheets/ielt9.css (*fragment*)

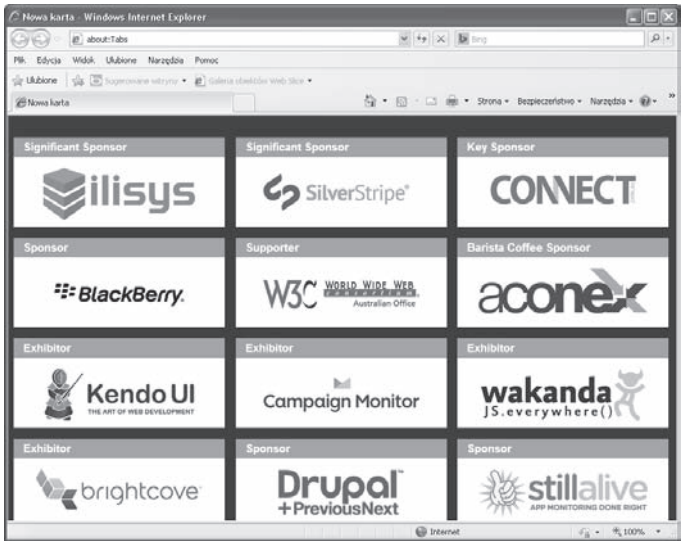
```
#sponsors {
   width: 960px;
}

#sponsors ul {
   padding-left: 3px;
}

#sponsors ul li {
   margin: 0 6px 15px 6px;
   width: 302px;
}
```

Style zastosowane w tych przeglądarkach opierają się na płynnym układzie o szerokości 960 pikseli, choć sam układ został zmodyfikowany tak, by miał stałą szerokość. W tym celu usunęliśmy wartości procentowe w szerokościach i marginesach, zastępując je konkretnymi wielkościami podanymi w pikselach. Odpowiada to możliwościom tych starszych przeglądarek (na przykład Internet Explorera przedstawionego na rysunku 4.11), zatem dostarczamy ich użytkownikom stronę o najlepszej postaci, jaką jesteśmy w stanie przygotować. Reagujemy na możliwości, jakimi dysponuje użytkownik, a w tym przypadku reakcja ta polega na wybraniu układu o stałej szerokości.

Dodawanie punktów granicznych

Zgodnie z informacjami opublikowanymi na witrynie Responsive Images Community Group[10], punkt graniczny jest logicznym odzwierciedleniem cech medium, które aktualizuje style używane na stronie: „Pojedynczy punkt graniczny reprezentuje regułę (lub grupę

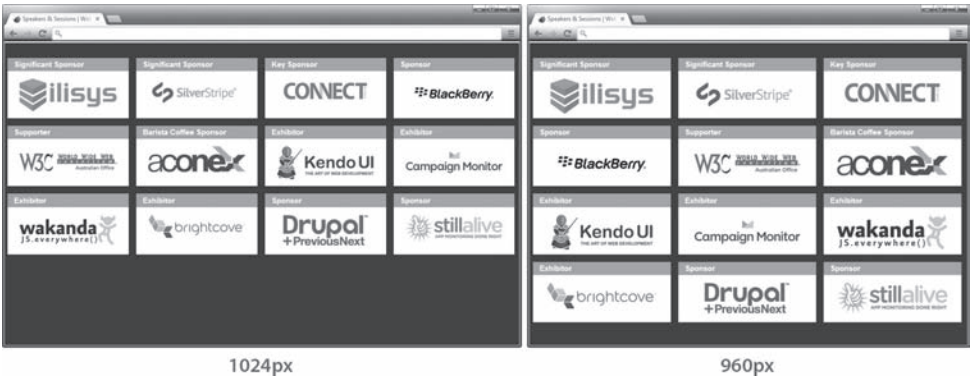
[10] *http://usecases.responsiveimages.org/#design-breakpoints.*



Rysunek 4.11. Stopka naszej witryny wyświetlona w starej przeglądarce Internet Explorer

reguł) określającą punkt, w którym zawartość danego zapytania medialnego zostaje zastosowana w układzie strony". Mimo że nie jest to oficjalna specyfikacja, zrozumienie jej wcale nie jest proste.

Najszybciej będziemy w stanie pojąć, o co w tym chodzi, jeśli wyobrazimy sobie punkty graniczne jako miejsca, w których nasz projekt ulega zmianie. Jeszcze raz wróćmy do przykładu witryny Web Directions South. Jak widać na rysunku 4.12, gdy szerokość okna przeglądarki zmniejszy się do 960 pikseli, czterokolumnowy układ strony ulega zmianie — punkt graniczny zostaje przekroczony, a strona zaczyna korzystać z układu trójkolumnowego.



Rysunek 4.12. W momencie przekroczenia punktu granicznego i zmniejszenia szerokości do 960 pikseli układ czterokolumnowy zostaje zastąpiony układem trójkolumnowym

Oczywiście w tym przypadku używana jest tylko jedna reguła, jednak dodawanie kolejnych punktów granicznych nie sprawia, że stają się one trudniejsze do zrozumienia — powiększa tylko ich liczbę. Poza tym może nasza witryna będzie używała znacznie więcej stylów

przeznaczonych dla konkretnych urządzeń niż tylko jeden? Wprost przeciwnie! Nawet jeśli ograniczymy się wyłącznie do sprawdzania szerokości, może nas kusić, by zastosować aż sześć punktów granicznych: 320px, 480px, 600px, 768px, 1024px oraz 1280px.

Korzystając wyłącznie z tych punktów, bylibyśmy w stanie obsłużyć szeroką gamę urządzeń, posługując się przy tym obecnie używanymi, standardowymi szerokościami (wyrażonymi w pikselach).

320px	urządzenia mobilne działające w układzie pionowym
480px	urządzenia mobilne działające w układzie poziomym
600px	małe tablety
768px	tablety działające w układzie pionowym
1024px	tablety działające w układzie pionowym, netbooki, komputery stacjonarne
1280px i więcej	duże komputery stacjonarne

A zatem gdybyśmy chcieli wykorzystać wszystkie te szerokości, nasze elementy `link` mogłyby mieć następującą postać:

```
<link rel="stylesheet" media="only screen
   and (max-device-width: 320px)" href="tiny.css">

<link rel="stylesheet" media="only screen
   and (max-device-width: 480px)" href="small.css">

<link rel="stylesheet" media="only screen
   and (max-device-width: 960px)" href="medium.css">

<link rel="stylesheet" media="only screen
   and (max-device-width: 1024px)" href="large.css">

<link rel="stylesheet" media="only screen
   and (min-device-width: 1280px)" href="extralarge.css">
```

Dobór odpowiednich punktów granicznych może być trudny: jeśli wybierzemy ich zbyt wiele, to utrzymanie witryny mogłoby się stać zbyt kłopotliwe, z kolei jeśli wybierzemy ich zbyt mało, to w efekcie strona nie będzie wyglądać dostatecznie dobrze na urządzeniach, których nie uwzględnialiśmy podczas projektowania. Jako absolutne minimum należy stosować dwa punkty graniczne: mały (przeznaczony dla wszystkich urządzeń poniżej 480 pikseli szerokości) oraz duży (dla wszystkich pozostałych przypadków). Właśnie takie postępowanie jest przykładem projektowania z myślą o urządzeniach mobilnych.

Kiedy już uda się nam dopracować te dwa układy, pozostaje pytanie: co zrobić z większymi ekranami? Przekonajmy się, w jaki sposób podzieliliśmy je na średnie, duże i bardzo duże.

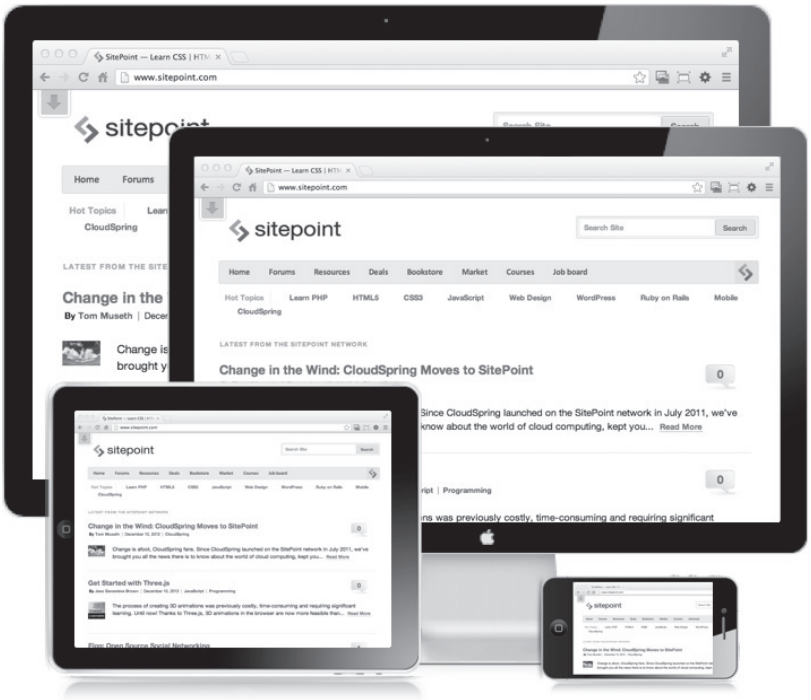
Układ dla ekranów średnich obejmuje zakres szerokości od 480 do 960 pikseli — jego górną granicę znamy z wcześniejszych rozważań dotyczących siatek. W dużym stopniu układ

ten będzie używany w przeglądarkach, które nie działają w trybie pełnoekranowym, jak również na wielu urządzeniach przenośnych, zwłaszcza tabletach, takich jak iPady lub urządzenia Kindle. W tych przypadkach niezwykle pomocny okazuje się element `meta viewport`, pozwalający dostosować układ o szerokości 960 pikseli do pracy na urządzeniach o mniejszych ekranach, takich jak iPady działające w układzie pionowym, których ekran ma szerokość 768 pikseli:

```
<meta name="viewport" content="width=device-width,
   initial-scale=1.0">
```

Kolejny, duży punkt graniczny obejmuje większość komputerów stacjonarnych, laptopów oraz tablety działające w układzie poziomym. Zaczyna się on na 960 pikselach szerokości, a kończy na 1280 pikselach. W ten sposób ostatni punkt graniczny — bardzo duży — obejmuje szerokości powyżej 1280 pikseli. Do tej kategorii zaliczają się nowoczesne monitory komputerów stacjonarnych o wielkości powyżej 20 cali, wysokiej klasy laptopy o szerokich ekranach i najnowsze tablety.

Te trzy punkty graniczne definiowane przy użyciu zapytań medialnych, w połączeniu z czwartym punktem korzystającym z domyślnych stylów przeznaczonych dla urządzeń mobilnych, wyznaczają cztery grupy ekranów zilustrowane na rysunku 4.13. Zapewniają one solidną obsługę niemal wszystkich naszych użytkowników.



Rysunek 4.13. Nasze cztery punkty graniczne i urządzenia, z myślą o których były tworzone

Powyżej tej granicy przekraczamy punkt malejących dochodów i stajemy wobec problemu określenia równowagi pomiędzy kosztami a zyskami z inwestycji, jakim jest dodawanie kolejnych punktów granicznych. Moglibyśmy dodać kolejny, bardzo mały punkt graniczny dla ekranów o szerokości poniżej 320 pikseli, lub jeszcze jeden punkt — pośredni dla szerokości 768 pikseli, który pozwoliłby dostosować witrynę do potrzeb siedmiocalowych tabletów. Jednak w związku z wykorzystywaniem w tabletach ekranów o coraz większej gęstości i coraz większych możliwościach tworzenie tak szczegółowych punktów granicznych jedynie zwiększyłoby nakłady pracy konieczne do utrzymania witryny, nie dając w zamian dużych korzyści. Niemniej jednak jeśli tworzone aplikacje są przeznaczone dla bardzo szerokiego grona użytkowników urządzeń mobilnych (wśród których mogą się także znaleźć posiadacze urządzeń starych lub o gorszej specyfikacji), to wykorzystanie tych dwóch dodatkowych punktów granicznych (bardzo małego i pośredniego) może być uzasadnione.

Dobór punktów granicznych

Należy dokładnie określić docelową grupę urządzeń, z myślą o których będziemy tworzyć naszą aplikację, i oszacować, czy zyski wynikające ze sposobu, w jaki użytkownicy będą z niej korzystać, pozwolą zrównoważyć koszty. To bardzo ważne, nawet jeśli zdecydujemy się używać tylko paru punktów granicznych lub jakiejś kombinacji pięciu punktów wymienionych powyżej: bardzo małego, małego, średniego, dużego i bardzo dużego.

Dokładna analiza opłacalności staje się jeszcze ważniejsza, gdy zdecydujemy się porzucić bezpieczeństwo, jakie dają nam punkty graniczne, i w jeszcze większym stopniu dostosować naszą aplikację do konkretnych urządzeń.

W bardzo wielu, lub nawet w przeważającej większości przypadków wykorzystanie trzech spośród pięciu wymienionych wcześniej punktów granicznych zapewni nam dostatecznie duży zakres obsługi użytkowników. Nie oznacza to jednak, że zastosowanie bardziej szczegółowego rozwiązania nie gwarantuje naszej aplikacji korzyści. Przyjrzyjmy się zatem solidnemu zestawowi zapytań medialnych, który będzie w stanie zapewnić obsługę szerokiego zakresu urządzeń, i to nawet bez zbytniego angażowania się w obsługę urządzeń z ekranami Retina i AMOLED. Opracowanie stylów i zaprojektowanie układów strony dla każdego z przedstawionych poniżej 12 punktów granicznych okazałoby się zadaniem tytanicznym, a ich późniejsze utrzymanie byłoby równie onieśmielające:

```
/* NIEWIELKIE TELEFONY Z SYSTEMEM ANDROID, TELEFONY SPECJALISTYCZNE – UKŁAD
  POZIOMY */
@media only screen and (max-width: 240px) {
      ...
}

/* SMARTFONY – UKŁAD PIONOWY I POZIOMY */
@media only screen and (min-device-width: 320px)
   and (max-device-width: 480px),
   only screen and (min-width: 320px) and (max-width: 480px) {
      ...
}
```

```
/* SMARTFONY – UKŁAD POZIOMY */
@media only screen and (min-width: 321px) {
      ...
}

/* SMARTFONY – UKŁAD PIONOWY */
@media only screen and (max-width: 320px) {
      ...
}

/* IPADY, TABLETY – UKŁAD PIONOWY I POZIOMY */
@media only screen and (min-device-width: 768px)
   and (max-device-width: 1024px),
   only screen and (min-width: 768px) and (max-width: 1024px) {
      ...
}

/* IPADY, TABLETY – UKŁAD POZIOMY */
@media only screen and (min-device-width: 768px)
   and (max-device-width: 1024px)
   and (orientation: landscape) {
      ...
}

/* IPADY, TABLETY – UKŁAD PIONOWY */
@media only screen and (min-device-width: 768px)
   and (max-device-width: 1024px) and (orientation:
   portrait) {
      ...
}

/* TABLETY, KOMPUTERY STACJONARNE I LAPTOPY */
@media only screen and (min-width: 960px) {
      ...
}

/* KOMPUTERY STACJONARNE I LAPTOPY */
@media only screen and (min-width: 1024px) {
      ...
}

/* KOMPUTERY STACJONARNE I LAPTOPY */
@media only screen and (min-width: 1224px) {
      ...
}

/* DUŻE EKRANY */
@media only screen and (min-width: 1824px) {
      ...
}

/* EKRANY RETINA I AMOLED */
@ media only screen and (-webkit-min-device-pixel-ratio:
   1.5), only screen and (min-device-pixel-ratio: 1.5) {
      ...
}
```

Zastosowanie powyższych punktów granicznych zapewniłoby naszemu zespołowi zajęcie na czas dłuższy, a poza tym stanowiłoby całkiem poważne wyzwanie. Znacznie lepszym rozwiązaniem byłoby wykorzystanie mniejszej liczby punktów i aktywne poszukiwanie tych, które można by uznać za przestarzałe i odrzucić, gdyż koszty ich utrzymania byłyby większe od zysków. Gdyby się okazało, że wymienione wcześniej punkty graniczne nie spełniają naszych oczekiwań i potrzeb, to nic nie stoi na przeszkodzie, by wybrać inny podzbiór wymienionych 12 punktów.

Jeśli koncentrujemy się na rynku urządzeń mobilnych, to prawdopodobnie moglibyśmy dokładniej określić zestaw niezbędnych punktów granicznych, jednak ograniczenie ich liczby jest równie ważne jak używanie składni zapytań medialnych. Istnieje tylko jedna żelazna reguła dotycząca liczby stosowanych punktów granicznych — zaleca ona, by przestać dodawać kolejne, gdy tylko zadowoli nas liczba użytkowników aplikacji.

Sprostać zadaniu

Spośród wszystkich technicznych filarów metodologii RWD zapytania medialne są najbardziej rozpowszechnione i najlepiej obsługiwane. Co więcej, z punktu widzenia projektowania aplikacji zapewniają one rozsądne zyski z poniesionych nakładów i możliwość użycia nawet w już istniejących aplikacjach w celu poprawienia ich atrakcyjności.

Choć faktycznie obsługa zapytań medialnych w starszych przeglądarkach jest raczej kiepska, należy pamiętać, że przeglądarki te są stosowane w coraz to mniejszym gronie użytkowników komputerów stacjonarnych.

Zapytania medialne są oparte na CSS i mogą być używane w przeglądarkach obsługujących tę technologię, dlatego eksperymenty przeprowadzane przez twórców przeglądarek doprowadziły do uzyskania takiego poziomu kontroli, który wyraźnie pokazuje ogromne możliwości, jakie daje stosowanie prefiksów przeglądarek. Oczywiście taki poziom kontroli sprawia, że trudno jest uniknąć pokusy zastosowania zapytań medialnych w celu dodawania coraz to nowych punktów granicznych. Jednak opierając się pędowi do nadużywania punktów granicznych i stosując wybrane punkty w stałych układach, projektanci mogą uzyskać możliwości tworzenia fantastycznych rozwiązań zaprojektowanych pod kątem użytkowników.

Sami będziemy sobie wdzięczni, jeśli opanujemy chęć nadużywania punktów granicznych i skupimy się na wykorzystaniu tylko wybranych spośród nich. Dzięki temu łatwiej będzie zarządzać stosowanymi zapytaniami medialnymi i skoncentrować się na tworzeniu wspaniałych stron dla tych punktów granicznych, których zdecydowaliśmy się użyć.

Skorowidz

H

I

J

K

L

M

N

O

P

R

S

T

U

W

Y

Z

Ż

Responsywne strony WWW potrafią dopasowywać się do rozmiaru ekranu użytkownika. Niezależnie od tego, czy korzysta on ze stacjonarnego komputera z dużym monitorem, z laptopa czy ze smartfona, zawsze może bez utraty kluczowej zawartości przeglądać ulubione witryny. Responsywność była do niedawna tylko dodatkiem do tworzonych stron — miała po prostu miło zaskoczyć użytkowników. Obecnie to standard usług WWW.

Z tej unikalnej książki dowiesz się, jak tworzyć responsywne serwisy. Nauczysz się stosować płynne siatki oraz korzystać z tych gotowych: 960 Grid System, 320 and Up oraz Bootstrap. Zrozumiesz, w jaki sposób działają responsywne strony, które nie tylko dostosowują swój układ do rozmiaru ekranu, lecz także adaptują do niego materiały multimedialne, głównie zdjęcia. W kolejnych rozdziałach znajdziesz obszerne omówienie tych zagadnień. Dowiesz się też, jak stworzyć własny elastyczny szablon oraz jakie są dostępne gotowe rozwiązania. Ta książka jest obowiązkową lekturą każdego dewelopera stron WWW, który chce być na czasie oraz korzystać z dobrodziejstw responsywnych serwisów!

Dzięki tej książce:

- **poznasz potencjał responsywnych stron WWW**
- **stworzysz własną płynną siatkę**
- **zaserwujesz użytkownikowi pliki multimedialne dostosowane do jego urządzenia**
- **poznasz szablony — Bootstrap, Foundation, Skeleton i inne**
- **zbudujesz responsywną stronę**

Obowiązkowa lektura każdego dewelopera stron WWW!

ISBN 978-83-246-9237-8

Cena: 32,90 zł