

O'REILLY®



WYKORZYSTAJ POTENCJAŁ HTML5 NA URZĄDZENIACH MOBILNYCH!



HELION

Estelle Weyl

Tytuł oryginału: Mobile HTML5

Tłumaczenie: Tomasz Walczak

ISBN: 978-83-246-8912-5

© 2014 Helion S.A.

Authorized Polish translation of the English edition Mobile HTML5,
ISBN 9781449311414 © 2014 Estelle Weyl.

This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls all rights to publish and sell the same.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

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)

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<http://helion.pl/user/opinie/htm5sm>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

Wprowadzenie	11
1. Przygotowania do nauki interfejsów API HTML5, CSS3 i JavaScriptu	25
CubeeDoo — gra na urządzenia przenośne napisana w HTML5	26
Narzędzia programistyczne	28
Edytor tekstu	28
Przeglądarka	28
Narzędzia do debugowania	29
Debugery stacjonarne	30
Zdalne debugowanie	32
Narzędzia do testowania	37
Emulatory i symulatory	38
Narzędzia dostępne w internecie	39
Telefony	40
Testy zautomatyzowane	42
2. Przechodzenie na HTML5	43
Składnia języka HTML5	43
Elementy	44
Atrybuty	45
Atrybuty globalne i atrybuty internacjonalizacji	45
Atrybuty z HTML-a 4, które stały się globalne w HTML5	48
Nowość w HTML5 — globalne atrybuty związane z dostępnością i elementami interaktywnymi	50
Składnia elementów i atrybutów w HTML-u	54
Elementy samozamykające	55
Najlepsze praktyki	56
Wymagane komponenty	57
Elementy ze znacznika <head>	62

Element <meta> — dodawanie metadanych	63
Znaczniki meta dla aplikacji mobilnych	64
Wartości charakterystyczne dla producentów urządzeń przenośnych	66
Znacznik <base> dla strony internetowej	67
Znacznik <link> jest przydatny nie tylko dla arkuszy stylów	67
3. Nowe elementy w HTML5	75
Elementy sekcyjne w HTML5	75
Element <section>	77
Element <article>	77
Element <section> a element <article>	78
Element <nav>	78
Element <aside>	79
Element <header>	79
Element <footer>	80
Nagłówek i stopka w grze CubeeDoo	80
Stary, ale rzadko używany element <address>	81
Grupowanie treści — inne nowe elementy HTML5	81
Element <main>	82
Elementy <figure> i <figcaption>	82
Element <hr>	83
Zmiany atrybutów elementów i	83
Nowe tekstowe elementy semantyczne w HTML5	83
Element <mark>	84
Element <time>	85
Elementy <rp>, <rt> i <ruby>	85
Element <bdi>	86
Element <wbr>	86
Zmodyfikowane tekstowe semantyczne elementy	86
Element <a>	86
Zmiany w elementach tekstowych w porównaniu z HTML-em 4	88
Niezmodyfikowane elementy	89
Elementy osadzone	90
Zmiany w elementach osadzanych	90
Elementy interaktywne	92
Elementy <details> i <summary>	92
Elementy <menu> i <menuitem>	94
Elementy XHTML-a niedostępne w HTML5	95
Wnioski	96

4. Formularze internetowe w HTML5 97

Atrybuty elementu <code><input></code> oraz innych elementów formularzy	99
Atrybut <code>type</code>	99
Atrybut <code>required</code>	99
Wartości minimalne i maksymalne — atrybuty <code>min</code> i <code>max</code>	100
Atrybut <code>step</code>	101
Atrybut <code>placeholder</code>	101
Atrybut <code>pattern</code>	102
Atrybut <code>readonly</code>	104
Atrybut <code>disabled</code>	104
Atrybut <code>maxlength</code>	105
Atrybut <code>size</code>	105
Atrybut <code>form</code>	105
Atrybut <code>autocomplete</code>	106
Atrybut <code>autofocus</code>	107
Typy i atrybuty elementu <code><input></code>	107
Ponowne wprowadzenie do typów, o których myślisz, że je znasz	108
Tekst — <code><input type="text"></code>	108
Hasła — <code><input type="password"></code>	109
Pole wyboru — <code><input type="checkbox"></code>	109
Przycisk opcji — <code><input type="radio"></code>	110
Przycisk wysyłania — <code><input type="submit"></code>	111
Przycisk resetowania — <code><input type="reset"></code>	112
Plik — <code><input type="file"></code>	112
Ukryte pole — <code><input type="hidden"></code>	114
Rysunki — <code><input type="image"></code>	114
Przycisk — <code><input type="button"></code>	114
Określanie stylów dla elementów <code><input></code> różnego typu	114
Nowe typy elementu <code><input></code>	115
Adres e-mail — <code><input type="email"></code>	116
Adres URL — <code><input type="url"></code>	117
Telefon — <code><input type="tel"></code>	118
Liczby — <code><input type="number"></code>	119
Przedziały — <code><input type="range"></code>	121
Wyszukiwanie — <code><input type="search"></code>	122
Kolory — <code><input type="color"></code>	122
Elementy <code><input></code> związane z datą i godziną	123
Daty — <code><input type="date"></code>	123
Data i godzina — <code><input type="datetime"></code>	125
Lokalna data i godzina — <code><input type="datetime-local"></code>	125
Miesiąc — <code><input type="month"></code>	125

Godzina — <code><input type="time"></code>	125
Tydzień — <code><input type="week"></code>	126
Sprawdzanie poprawności formularzy	127
Łatwe poprawianie interfejsu użytkownika za pomocą stylów CSS	130
Nowe elementy formularzy	132
Element <code><datalist></code> i atrybut <code>list</code>	132
Element <code><output></code>	134
Element <code><meter></code>	135
Element <code><progress></code>	136
Element <code><keygen></code>	137
Inne elementy formularzy	137
Element <code><form></code>	137
Elementy <code><fieldset></code> i <code><legend></code>	138
Elementy <code><select></code> , <code><option></code> i <code><optgroup></code>	138
Element <code><textarea></code>	138
Element <code><button></code>	139
Element <code><label></code>	139
Wnioski	139
5. Elementy <code>svg</code>, <code>canvas</code>, <code>audio</code> i <code>video</code>	141
Multimedialne interfejsy API w HTML5	141
SVG	141
Dołączanie grafiki SVG do dokumentów	143
Technika Clown Car — SVG i dynamicznie dopasowywana grafika pierwszego planu	144
Nauka SVG	145
Format SVG w grze CubeeDoo	146
Element <code>canvas</code>	148
Element <code><canvas></code> a element <code><svg></code>	152
Elementy <code><audio></code> i <code><video></code>	154
Typy plików multimedialnych	154
Dodawanie elementu <code><video></code> do witryny	155
Atrybuty elementów <code><video></code> i <code><audio></code>	155
Elementy <code><video></code> i <code><audio></code> a JavaScript	159
Określanie stylu elementu <code><video></code>	161
6. Inne interfejsy API z HTML5	165
Aplikacje internetowe działające w trybie offline	165
Czy urządzenie jest podłączone do sieci?	165
Pamięć podręczna aplikacji	166
Pamięć lokalna i pamięć sesji	170
Pamięć oparta na SQL-u i bazach danych	179

Wzbogacanie user experience	183
Geolokalizacja	183
Sieciowe wątki robocze	186
Mikrodane	188
Przekazywanie komunikatów między dokumentami	190
Specyfikacja ARIA	191
Dostępność	191
Wnioski	194
7. Przechodzenie na CSS3	195
CSS — definicje i składnia	196
Składnia CSS	196
Używanie zewnętrznych arkuszy stylów — jeszcze o elemencie <link>	198
Zapytania media	200
Najlepsze praktyki związane ze stylami CSS	202
Selektory CSS	206
Podstawowe selektory	207
Inne selektory CSS3	210
Selektory ogólne	210
Stosowanie selektorów	211
Selektory relacyjne — reguły oparte na kolejności kodu	212
Selektory atrybutów	215
Pseudoklasy	219
Pseudoklasy określające stan	222
Pseudoklasy strukturalne	223
Obliczenia dla pseudoklas nth	224
Inne pseudoklasy	227
Pseudoelementy	229
Inne selektory — model Shadow DOM	232
Specyficzność ważniejsza od kaskadowości — specyficzność w stylach CSS	233
Wnioski	234
8. Dodatkowe możliwości dzięki wartościom z CSS3	235
Wartości kolorów w CSS3	235
Wartości szesnastkowe	236
Składnia rgb()	237
Dodawanie przezroczystości w formacie RGBA	238
Odcień, nasycenie i jasność — HSL()	239
CMYK	240
Nazwane kolory	240
Słowo kluczowe currentColor	240
Wartości kolorów w przeglądarkach	241

Jednostki miar w CSS-ie	244
Wartości określające długość w CSS-ie	244
Kąty, czas i częstotliwość	247
Kąty w CSS-ie	247
Czas	249
Częstotliwość	249
Porządek GPDŁ — skrócone deklaracje właściwości i wartości	249
Wnioski	252
9. CSS3 — moduły, modele i grafika	253
Właściwości modelu pudełkowego z CSS-a	254
Obramowanie	255
Właściwość border-style	256
Właściwość border-color	256
Właściwość border-width	257
Model pudełkowy z CSS-a	257
Właściwość box-sizing	259
Nauka CSS3	260
Właściwość border-radius	261
Gradienty w CSS-ie	264
Typy gradientu: liniowy lub kołowy	264
Gradienty kołowe	265
Gradienty liniowe	265
Właściwość background-size	274
Gradienty z paskami	277
Powtarzanie gradientów liniowych	278
Cienie	282
Cienie dla tekstu	283
Ustawianie tekstu za pomocą właściwości width, overflow i text-overflow	284
Cienie pól	286
Łączenie wszystkich właściwości w grze CubeeDoo	288
10. Transformacje, przejścia i animacje w CSS3	293
Przejścia w CSS-ie	294
Właściwość transition-property	295
Właściwość transition-duration	298
Właściwość transition-timing-function	298
Właściwość transition-delay	299
Właściwość skrócona transition	300
Różne przejścia	301
Transformacje w CSS3	302
Właściwość transform-origin	303
Właściwość transform	304

Łączenie wielu transformacji	308
Stosowanie przejść dla transformacji	309
Funkcje transformacji trójwymiarowych	309
Inne właściwości transformacji trójwymiarowych	311
Łączenie wszystkich elementów	313
Animacje w CSS3	315
Klatki kluczowe	317
Przejścia, animacje i wydajność	323
11. CSS w projektowaniu RWD	325
Zapytania media, punkty graniczne i płynny układ	325
Kilka kolumn	326
Grafika obramowania	328
Ustawianie grafiki obramowania	329
Model flexbox	334
Właściwość flex	337
Wykrywanie funkcji za pomocą reguły @supports	339
Dynamiczne dostosowywanie do wymiarów ekranu	340
Udostępnianie grafiki	340
Maski w CSS-ie — tworzenie przezroczystych plików JPEG	346
Nagłówek Client-Hints	346
12. Projektowanie aplikacji mobilnych	349
O czym warto pomyśleć przed rozpoczęciem pracy?	350
Kwestie projektowe	351
Aplikacje zwiększające produktywność	352
Rozrywka — wciągające aplikacje	353
Narzędzia	354
Jaki typ aplikacji będzie odpowiedni dla Ciebie?	355
Platforma mobilna — bogactwo możliwości	356
Mały ekran	356
Mniejsza ilość pamięci	356
Jedno okno i jedna aplikacja naraz	358
Minimalna dokumentacja	359
Rozważania związane z programowaniem	359
Tworzenie rozwiązań dla mobilnych przeglądarek z silnikiem WebKit	360
Pasek stanu	360
Pasek nawigacji	361
Rysunek startowy	363
Ikony na stronie głównej urządzenia	364

Minimalizowanie ilości danych wprowadzanych z klawiatury	365
Zachowaj zwięzłość	365
Stosuj oczywiste rozwiązania	365
Minimalizuj ilość wprowadzanych danych	365
Minimalizuj ilość tekstu	365
Inne kwestie związane z user experience	366
13. Urządzenia przenośne i dotyk	367
Dostosowywanie stron do małych ekranów	367
Reguła @viewport	368
Dotknij mnie	368
Obszary reagujące na dotyk	369
Zdarzenia związane z myszą i dotykiem	369
Fikcyjne i rzeczywiste zdarzenia związane z kliknięciem	372
Dostęp do sprzętu	375
Ruch i kierunek poruszania telefonu	375
Stan urządzenia	376
Natywne aplikacje sieciowe, aplikacje w formie pakietów i rozwiązania hybrydowe	378
Testy	379
14. Wydajność w środowisku mobilnym	381
Czas pracy na baterii	381
Stosuj ciemne kolory	382
Stosuj format JPEG	382
Ogranicz ilość kodu w JavaScriptcie	383
Eliminuj liczbę przesyłanych żądań	384
Akceleracja sprzętowa	385
Opóźnienie	387
Zmniejszanie liczby żądań HTTP	388
Zmniejszanie wielkości żądań	392
Pamięć	395
Optymalizowanie grafiki	396
Szybkość reagowania interfejsu użytkownika	401
Zdarzenia związane z dotykiem	401
Animacje	402
Wnioski	403
A Selektory CSS-a i specyficzność	405
Skorowidz	413

Elementy svg, canvas, audio i video

Wcześniej omówiłam większość nowych elementów z HTML5. Wyjątkiem są elementy powiązane bezpośrednio z tworzonymi dopiero interfejsami API, a także dobrze obsługiwane elementy multimedialne `svg`, `canvas`, `audio` i `video`. Elementy z pierwszej z tych grup mogą jeszcze zostać zmodyfikowane, dlatego nie omawiam ich w tej książce. Elementy z drugiej grupy opisuję w tym rozdziale.

Przedstawiam tu najważniejsze funkcje, które programiści frontonów aplikacji sieciowych prawdopodobnie będą stosować w codziennej pracy. Dzięki temu przy tworzeniu kodu dla przeglądarek mobilnych będziesz mógł korzystać z nowych mechanizmów. Wszystkie nowe przeglądarki mobilne (z wyjątkiem Opery Mini) obsługują elementy `<canvas>`, `<video>` i `<audio>`, a także sieciowe interfejsy API związane z geolokalizacją, pamięcią lokalną, aplikacjami sieciowymi działającymi w trybie offline itd.

Na każdy z tematów omawianych w tym rozdziale można napisać całą książkę — i dla wielu opisywanych zagadnień takie książki już są dostępne. Przetawiam tu wystarczająco dużo informacji, abyś mógł uznać, że chcesz przeczytać książkę na dany temat, lub stwierdzić, że danym zagadnieniem nie jesteś zainteresowany. Choć nie opisuję szczegółowo poszczególnych kwestii, ten rozdział zapewni Ci wiedzę potrzebną do rozpoczęcia pracy. Co ważniejsze, zrozumiesz zalety i wady omawianych tu technologii w kontekście urządzeń przenośnych.

Multimedialne interfejsy API w HTML5

Pierwotna specyfikacja HTML-a dotyczyła wyłącznie materiałów tekstowych. Nie obejmowała nawet elementu `<img>`. Od tamtego czasu dużo się zmieniło. HTML5 umożliwia tworzenie skalowalnej grafiki wektorowej za pomocą formatu SVG i pustego obszaru, po którym można rysować (odpowiada mu element `<canvas>`). Oprócz grafiki w HTML5 obsługiwane są elementy `<video>` i `<audio>`, które działają bez konieczności stosowania niezależnych wtyczek.

SVG

Format SVG umożliwia tworzenie złożonej *skalowalnej grafiki wektorowej*. Ten format został wprowadzony w 2001 r. i jest otwartym standardem definiowania dwuwymiarowej grafiki wektorowej. Skalowalność w SVG oznacza, że ta sama grafika będzie równie ostra na dużych monitorach, jak i na małych wyświetlaczach urządzeń przenośnych. Nie wymaga to wprowadzania żadnych modyfikacji.

W specyfikacji SVG jest zdefiniowana oparta na XML-u gramatyka tworzenia kształtów, linii, krzywych, rysunków i tekstu. Obsługiwane są też różne funkcje, takie jak: przezroczystość, dowolne kształty geometryczne, filtry (cienie, efekty świetlne itd.), skrypty oraz animacje.

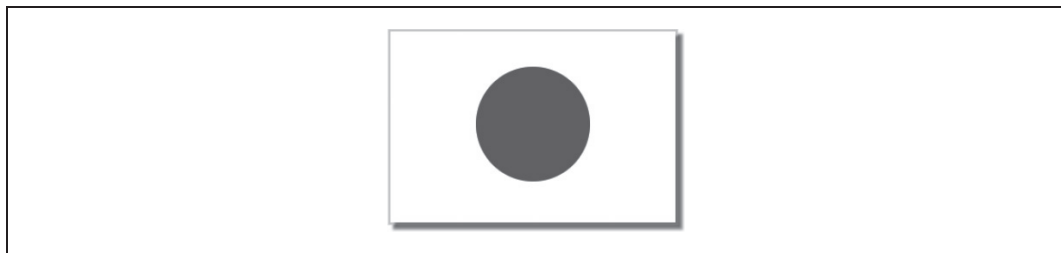
Ponieważ SVG to format graficzny oparty na teście, pliki mogą być bardzo małe. Ponieważ w tym formacie używany jest model obiektowy, grafiki można modyfikować za pomocą skryptów. Ponieważ jest to format wektorowy, można skalować go bez negatywnych efektów takich jak pikselizacja lub nierówne krawędzie. Ten format jest zrozumiały (dzięki temu, że jest deklaratywny) i obsługuje animacje.

Istnieje kilka wersji formatu SVG. Są one obsługiwane w przeglądarkach w różnym stopniu. Podstawowa obsługa samodzielnych plików .svg jest dostępna we wszystkich urządzeniach przenośnych i nowych przeglądarkach. W Androidzie pliki tego typu są obsługiwane od wersji 3. Grafikę w formacie SVG można stosować jako źródło elementów od systemów iOS 3.2 i Android 3.0 oraz od mobilnej wersji Internet Explorera 8.

Pliki w formacie SVG jako wartość właściwości background-image w stylach CSS są obsługiwane od systemów Android 3 i iOS 3.2. Ponadto od dawna można ich używać w Operze Mobile. Element <svg> na stronach HTML5 można stosować od systemów iOS 5 i Android 3 oraz od przeglądarki Internet Explorer 9 (a także we wszystkich innych nowych przeglądarkach). Tylko w systemach Android 2.3.3 i starszych, Amazon Silk i nieużywanym już WebOS firmy HP przeglądarki mobilne nie zapewniają obsługi formatu SVG. Statyczne pliki SVG są obsługiwane nawet w Operze Mini (podobnie jak element <canvas>, który jest obsługiwany we wszystkich przeglądarkach mobilnych), przy czym nie można ich animować, ponieważ poziom obsługi JavaScriptu w tej przeglądarce to uniemożliwia.

Ponieważ SVG jest oparty na XML-u, głównym elementem plików w tym formacie jest nie <html>, a <svg>. Pliki SVG, podobnie jak wszystkie dokumenty XML, rozpoczynają się od prologu XML-a i deklaracji DTD dla formatu SVG. Element główny <svg> zawiera całą treść dokumentu. Pliki w formacie SVG nie obejmują elementów <head> i <body>. Cała treść, w tym zagnieżdżone elementy <svg>, znajduje się w głównym elemencie <svg>.

Dobrym wprowadzeniem do formatu SVG jest japońska flaga. Jest to biały prostokątny obszar z czerwonym kołem (słońcem) w środkowej części, co przedstawia rysunek 5.1.



Rysunek 5.1. Japońska flaga wygenerowana za pomocą pliku SVG

```
1 <!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.0//EN"
2   "http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">
3 <svg xmlns="http://www.w3.org/2000/svg" height="220" width="320" version="1.0">
4   <title>Japońska flaga</title>
5   <desc>Czerwone koło na białym prostokącie</desc>
6   <rect x="10" y="10" width="300" height="200"
7     style="fill: #ffffff; stroke: #e7e7e7;"/>
8   <circle cx="160px" cy="107px" r="60px" fill="#d60818"/>
9 </svg>
```

Cóż ten kod oznacza? Wiersze od 1 do 3 to deklaracja DTD dla plików SVG i główny element `<svg>`. Warto zauważyć, że w głównym elemencie zadeklarowana jest wielkość danej grafiki wektorowej. Aby w stylach CSS móc używać grafiki w formacie SVG we właściwości `background-position`, trzeba zadeklarować wymiary tej grafiki. Jest to ważne, gdy tworzysz sprite'y w formacie SVG.

Można dodać element `<title>` (wiersz 4), jeśli plik SVG jest używany niezależnie od innych zasobów. Element `<desc>` (wiersz 5) zawiera tekstowy opis, który normalnie się nie wyświetla, gdy pokazywany jest plik SVG. Umieszczenie odpowiednich informacji w elementach `<desc>` i `<title>` zwiększa dostępność stron. Ponieważ nie wszystkie czytniki ekranu obsługują format SVG, dodanie atrybutu `aria-label` pozwala poprawić dostępność stron.

Element `<rect>` (wiersz 6) oznacza prostokąt. Dostępne kształty i linie to: `<path>`, `<rect>`, `<circle>`, `<ellipse>`, `<line>`, `<polyline>` i `<polygon>`. Tu podane są cztery atrybuty: `x`, `y`, `width` i `height`. Określają one współrzędne `x` i `y` oraz szerokość i wysokość prostokąta. Używany jest też atrybut `style`.

W dokumentach SVG, podobnie jak w dokumentach HTML, można używać stylów CSS do określania stylu elementów. Możesz zadeklarować style wewnątrzwierszowo za pomocą atrybutu `style`, jak to zrobiłam w przedstawionym przykładzie, albo dołączyć osadzony lub zewnętrzny arkusz stylów i wskazywać w nim elementy przy użyciu selektorów (ta technika jest stosowana w plikach HTML).

Nazwy właściwości są tu nieco odmienne niż w stylach CSS, do których jesteś przyzwyczajony, jednak są zrozumiałe. Właściwość `fill` działa podobnie jak właściwość `background` — tu określa kolor tła. Właściwość `stroke` pełni funkcję podobną do właściwości `border` w stylach CSS. Można też określić gradient lub wzór.

Choć w plikach SVG można stosować większość właściwości i wartości ze stylów CSS, ze względu na bezpieczeństwo niektórzy producenci przeglądarek¹ blokują importowanie grafiki rastrowej i skryptów w plikach SVG zapisanych jako grafika tła w elemencie `<img>`.

Element `<circle>` (wiersz 8) tworzy koło o czerwonym tle. Zamiast wysokości i szerokości w tym elemencie podawany jest atrybut `r`, który oznacza promień. Pozycja nie jest tu podawana na podstawie górnego lewego rogu (jak to się odbywa w elemencie `<rect>`), ale za pomocą środka koła. Atrybut `cx` określa współrzędną `x` środka koła, natomiast `cy` to współrzędna `y` środka.

Jeśli przyjrzyysz się atrybutom elementu `<circle>`, zauważysz, że jednym z nich jest `fill`, natomiast w elemencie `<rect>` `fill` jest podany jako właściwość stylów CSS.

Dołączanie grafiki SVG do dokumentów

Możesz dodać plik SVG bezpośrednio do dokumentu. Umożliwiają to znaczniki `<img>`, `<object>` i `<embed>`:

```

```

lub:

```
<embed type="image/svg+xml" src="flag.svg" width="320" height="220"/>
```

lub:

```
<object data="flag.svg" type="image/svg+xml" width="320" height="220"></object>
```

¹ Obecnie silniki WebKit i Mozilla blokują importowanie skryptów i grafiki rastrowej do plików SVG ze znaczników `<img>`. Dzieje się tak nawet wtedy, gdy grafika rastrowa pochodzi z tego samego źródła.

Warto zauważyć, że choć elementy `<embed>` i `<object>` nie mają atrybutu `alt`, grafika SVG może być dostępna. Aby zwiększyć dostępność strony, należy opisać grafikę za pomocą elementów `<desc>` lub `<title>` i dodać atrybut `aria-label` o wartości odpowiadającej tytułowi grafiki. Gdy podasz wysokość i szerokość w pliku `<svg>`, nie musisz tego robić w elementach `<img>`, `<embed>` i `<object>`, natomiast te wartości są potrzebne w stylach CSS.

Technika Clown Car — SVG i dynamicznie dopasowywana grafika pierwszego planu

Format SVG można wykorzystać do tworzenia i udostępniania dynamicznie dopasowywanej grafiki. Możesz wykorzystać obsługę formatu SVG w przeglądarkach oraz obsługę zapytań media i grafiki wektorowej w formacie SVG, aby tworzyć dynamicznie dopasowywane rysunki. Aby pobierać odpowiednią grafikę, należy zastosować zapytania media w pliku SVG.

Wiadomo, że dzięki grafice tła ze stylów CSS możliwe jest pobieranie tylko potrzebnych rysunków. Podobnie aby zapobiec pobieraniu w plikach SVG wszystkich dołączanych obrazów, można wykorzystać grafikę tła ze stylów CSS zamiast grafiki pierwszego planu. W dynamicznie dopasowywanych plikach SVG należy dołączać wszystkie potrzebne rysunki i na podstawie zapytań media wyświetlać tylko odpowiedni z nich (szczegółowe omówienie zapytań media znajdziesz w rozdziale 7).

```
<svg xmlns="http://www.w3.org/2000/svg"
  viewBox="0 0 300 329" preserveAspectRatio="xMidYMid meet">

<title>Tu wpisz wartość atrybutu alt</title>

<style>
  svg {
    background-size: 100% 100%;
    background-repeat: no-repeat;
  }

  @media screen and (max-width: 400px) {
    svg {
      background-image: url(images/small.png);
    }
  }

  @media screen and (min-width: 401px) and (max-width: 700px) {
    svg {
      background-image: url(images/medium.png);
    }
  }

  @media screen and (min-width: 701px) and (max-width: 1000px) {
    svg {
      background-image: url(images/big.png);
    }
  }

  @media screen and (min-width: 1001px) {
    svg {
      background-image: url(images/huge.png);
    }
  }
</style>
</svg>
```

Aby zachować proporcje elementów i zapewnić ich jednolite skalowanie, zastosowałam atrybuty `viewbox` i `preserveAspectRatio`. Wartość atrybutu `viewbox` to lista czterech rozdzielonych odstępami lub przecinkami liczb: minimalnej wartości `x`, minimalnej wartości `y`, szerokości i wysokości. Zdefiniowanie szerokości i wysokości okna roboczego pozwala określić proporcje grafiki SVG.

Z uwagi na związane z bezpieczeństwem problemy z importowaniem grafiki rastrowej za pomocą elementu `<img>` i plików SVG należy używać elementu `<object>`, aby dodawać do witryny dynamicznie dopasowywane rysunki. Element `<object>` umożliwia traktowanie zewnętrznych zasobów jak grafiki.

```
<object data="awesomefile.svg" type="image/svg+xml"></object>
```

Element `<object>` domyślnie jest tak szeroki jak element nadrzędny. Jednak, podobnie jak przy dodawaniu grafiki, można określić szerokość i wysokość elementu za pomocą atrybutów `width` i `height` albo przy użyciu właściwości `width` i `height` w stylach CSS. Z uwagi na umieszczone w przykładowym pliku SVG deklaracje `viewbox` i `preserveAspectRatio` element `<object>` domyślnie zachowa zadeklarowane proporcje, jeśli podany zostanie tylko jeden wymiar (szerokość lub wysokość).

Ponieważ w opisaney technice używany jest element `<object>` zamiast `<img>`, nie występuje tu atrybut `alt`. Aby ta technika była dostępna dla czytelników ekranu z obsługą formatu SVG², należy umieścić zawartość atrybutu `alt` w elemencie `<title>` w pliku SVG.

W elemencie `<object>` osadzany jest plik SVG. Ten plik pobiera grafikę tła, która pasuje do zapytania media. Robi to na podstawie wymiarów elementu `<object>`, a nie okna roboczego. Dla pokazanego wcześniej kodu zgłaszane są dwa żądania HTTP. Jedno dotyczy pliku SVG, a drugie odpowiedniej grafiki. Aby wystarczyło jedno żądanie HTTP, w atrybucie `data` elementu `<object>` zapisz identyfikator URI danych (musisz zastosować w nim znaki ucieczki³).

Omawianą technikę nazywam Clown Car (czyli: samochód klauna), ponieważ polega na umieszczaniu wielu dużych rysunków (klaunów) w jednym małym pliku graficznym SVG (w samochodzie).

Nauka SVG

Opisałam tu tylko podstawy formatu SVG. Pliki SVG mogą być dostępne, skalują się do ekranów o dowolnej rozdzielczości, a także obsługują animacje oparte na składni SVG lub skryptach JavaScriptu. Pełną kontrolę nad każdym elementem animacji można uzyskać za pomocą interfejsu API modelu SVG DOM. Za pomocą formatu SVG można zrobić bardzo wiele rzeczy, jednak ich omawianie wykracza poza zakres tej książki. W specyfikacji tego formatu w witrynie W3C (<http://www.w3.org/TR/SVG/>) znajdziesz więcej informacji na temat wszystkich elementów, atrybutów i interfejsu API do obsługi animacji.

² Zob. <http://www.i18n.com/just-how-accessible-is-svg/>.

³ Znaki ucieczki w identyfikatorze URI danych są potrzebne w Internet Explorerze 9 i nowszych wersjach tej przeglądarki. Tu tylko pokrótce omawiam technikę Clown Car. Więcej szczegółów i przykładów wraz z rozwiązaniami rezerwowymi dla przeglądarek, które nie obsługują formatu SVG, znajdziesz pod adresem <https://github.com/estelle/clowncar>.

Plik z japońską flagą to bardzo prosty dokument SVG. Pliki SVG mogą być bardzo skomplikowane. Jeśli znasz program Adobe Illustrator, może zauważyłeś, że rysunki można wyeksportować do formatu SVG. Choć jest to bardzo dobry sposób tworzenia skomplikowanych plików SVG, powstaje w ten sposób bardzo długi kod, a wspomniany program jest drogi.

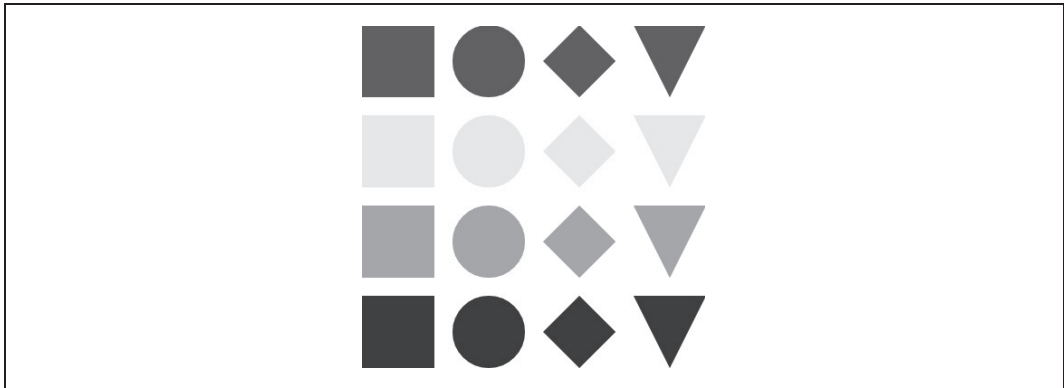
Amaya to bezpłatny program obsługujący podzbiór formatu SVG. Obsługiwane są m.in.: proste kształty, tekst, grafika, zewnętrzne obiekty, przezroczystość oparta na kanale alfa, transformacje oraz animacje. Program Amaya (<http://www.w3.org/Amaya/>) możesz pobrać bezpośrednio z witryny W3C. Amaya jest przydatna w nauce formatu SVG, ponieważ wygenerowany kod źródłowy można badać i edytować. Możesz też wypróbować program Inkscape (<http://inkscape.org/en/>). Jest to edytor grafiki wektorowej. Ten edytor jest programem o otwartym dostępie do kodu źródłowego i oferuje możliwości podobne do aplikacji Illustrator, CorelDraw i Xara. Inkscape używa plików SVG w formacie zgodnym ze standardem W3C.

Format SVG w grze CubeeDoo

W CubeeDoo używamy formatu SVG w dwóch miejscach. Grafiką tła w motywie z kształtami jest sprite w formacie SVG, a dla ikony wyciszenia używamy identyfikatora URI danych wskazującego plik SVG.

W grze dostępnych jest kilka motywów, m.in.: liczby, kolory i kształty. Do tworzenia kształtów służą proste sprite'y w formacie SVG. Poniżej znajduje się kod do tworzenia sprite'u SVG używanego na przednich stronach kart (rysunek 5.2).

```
1 <!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.0//EN"
2   "http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">
3 <svg xmlns="http://www.w3.org/2000/svg" height="400" width="400" version="1.0">
4   <desc>Sprite z kwadratami, kółkami, rombami i trójkątami</desc>
5
6   <!-- Kolorowe kwadraty -->
7   <rect x="10" y="10" width="80" height="80" style="fill: #d60818;"/>
8   <rect x="10" y="110" width="80" height="80" style="fill: #ffff33;"/>
9   <rect x="10" y="210" width="80" height="80" style="fill: #00FF00;"/>
10  <rect x="10" y="310" width="80" height="80" style="fill: #0000FF;"/>
11
12  <!-- Kolorowe kółka -->
13  <circle cx="150" cy="50" r="40" style="fill: #d60818;"/>
14  <circle cx="150" cy="150" r="40" style="fill: #ffff33;"/>
15  <circle cx="150" cy="250" r="40" style="fill: #00FF00;"/>
16  <circle cx="150" cy="350" r="40" style="fill: #0000FF;"/>
17
18  <!-- Romby -->
19  <polygon points="250,10 210,50 250,90 290,50" style="fill: #d60818;"/>
20  <polygon points="250,110 210,150 250,190 290,150" style="fill: #FFFF33;"/>
21  <polygon points="250,210 210,250 250,290 290,250" style="fill: #00FF00;"/>
22  <polygon points="250,310 210,350 250,390 290,350" style="fill: #0000FF;"/>
23
24  <!-- Trójkąty -->
25  <polygon points="310,10 350,90 390,10" style="fill: #d60818;"/>
26  <polygon points="310,110 350,190 390,110" style="fill: #FFFF33;"/>
27  <polygon points="310,210 350,290 390,210" style="fill: #00FF00;"/>
28  <polygon points="310,310 350,390 390,310" style="fill: #0000FF;"/>
29 </svg>
```

Rysunek 5.2. Sprite SVG z kształtami

Wiersz 1 zawiera deklarację DTD. W wierszu 3 deklarowany jest element główny. Podana jest tu też wysokość i szerokość grafiki SVG. Choć zgodnie ze specyfikacją nie trzeba podawać tych atrybutów, są one niezbędne, jeśli chcesz używać rysunku SVG jako grafiki tła. W wierszu 4 znajduje się opis, co zwiększa dostępność stron i pomaga w ich optymalizacji ze względu na wyszukiwarki.

W wierszach od 7 do 10 znajdują się deklaracje czterech kwadratów. Wiersz 9 oznacza: utwórz prostokąt w odległości 10 pikseli od lewej krawędzi i 210 pikseli od górnej krawędzi; prostokąt ma być szeroki na 80 pikseli i wysoki na 80 pikseli; wypełnij utworzony kształt kolorem #00FF00.

```
<rect x="10" y="210" width="80" height="80" style="fill: #00FF00;"/>
```

W wierszach od 13 do 16 zdefiniowane są cztery kółka. Wiersz 16 oznacza: znajdź punkt oddalony o 150 pikseli od lewej krawędzi i 350 pikseli od górnej krawędzi; ustaw ten punkt jako środek kółka o promieniu 40 pikseli wypełnionego kolorem #0000FF.

```
<circle cx="150" cy="350" r="40" style="fill: #0000FF;"/>
```

Wiersz od 18 do 28 zawierają osiem wielokątów: cztery romby i trzy trójkąty. Wielokąty deklaruje się przez podanie ich narożników. Wiersz 19 oznacza: ten kształt ma cztery narożniki; pierwszy punkt jest oddalony o 250 pikseli od lewej krawędzi i 10 pikseli od górnej krawędzi; drugi punkt jest oddalony o 210 pikseli od lewej krawędzi i 50 pikseli od górnej krawędzi; trzeci punkt jest oddalony o 250 pikseli od lewej krawędzi i 90 pikseli od górnej krawędzi; czwarty punkt jest oddalony o 290 pikseli od lewej krawędzi i 50 pikseli od górnej krawędzi; obszar między tymi punktami należy wypełnić kolorem #d60818 (jest to odcień czerwonego).

```
<polygon points="250,10 210,50 250,90 290,50" style="fill: #d60818;"/>
```

Pokazany kod tworzy kwadraty, kółka, romby i trójkąty obrócone podstawą do góry.

Te małe rysunki można też zapisać jako identyfikatory URI danych bezpośrednio w pliku CSS lub udostępnić jako grafikę pierwszego planu. Możesz np. dodać zakodowany plik SVG jako identyfikator URI danych:

```
background-image: url(data:image/svg+xml,%3Csvg%20xmlns%3D%22http%3A%2F%2Fwww.w3.org%2F2000%2Fsvg%22%20version%3D%221.0%22%3E%3Crect%20x%3D%20%22%20y%3D%220%22%20fill%3D%22%23abcdef%22%20width%3D%22100%25%22%20height%3D%22100%25%22%20f%3E%3C%2Fsvg%3E%3E);
```

W CubeeDoo pojawia się także ikona wyciszenia dźwięku. Identyfikator URI danych dla tej ikony wygląda tak:

```
background-image:
  background-image:
    url("data:image/svg+xml;utf8,%3Csvg%20xmlns=
'http://www.w3.org/2000/svg'%20width='100'%20height='100'%3E
%3Cpolygon%20points='39,13%2022,28%206,28%206,47%2022,48%2039,63
%2039,14'%20style='stroke:#111111;stroke-width:5;stroke-linejoin:round;
fill:#111111;%20/%3E%3Cpath%20d='M%2048,50%2069,26'%20%20style='fill:none;
stroke:#111111;stroke-width:5;stroke-linecap:round'%20/%3E%3Cpath%20
%20d='M%2069,50%2048,26'%20style='fill:none;stroke:#111111;stroke-width:5;
stroke-linecap:round'%20/%3E%3C/svg%3E");
```

Dane w tym przykładzie są mało czytelne dla ludzi. Utworzono jest za pomocą programu Amaya. Składnia powinna jednak wyglądać znajomo. Używamy tu właściwości `background-image` ze stylów CSS. Zamiast wywołania `url(path/mute.jpg)` lub nawet `url(path/mute.svg)` stosujemy instrukcję `url("data:image/svg+xml;utf8,<svg... /></svg>")`; Cały plik SVG ze znakami ucieczki jest ujęty w cudzysłów.

Zgodnie ze specyfikacją dla wersji Internet Explorera obsługujących format SVG (obecnie są to wersje 9 i 10) w identyfikatorach URI danych należy stosować znaki ucieczki.

Element canvas

Specyfikacja HTML5 Canvas dotyczy przeznaczonego do tworzenia rysunków interfejsu API JavaScriptu. Ten interfejs umożliwia zdefiniowanie na stronie HTML obiektu płótna w postaci elementu `<canvas>`, po którym można rysować. Rysunki utworzone w ten sposób można nawet wstawiać w stylach CSS jako grafikę tła.

Rysować można w przestrzeni dwu- lub trójwymiarowej (do tworzenia rysunków trójwymiarowych służy technologia WebGL). Rysunki dwuwymiarowe są obsługiwane we wszystkich nowych przeglądarkach internetowych. Technologia WebGL dopiero zyskuje popularność w świecie mobilnym. Ze względu na wydajność należy ją stosować tylko w urządzeniach ze sprzętową akceleracją grafiki.

W wersji dwuwymiarowej dostępny jest prosty, ale dający duże możliwości interfejs API do szybkiego rysowania na dwuwymiarowych bitmapach. Nie występuje tu konkretny format plików, a rysować można tylko za pomocą skryptów. Nie są dostępne węzły modelu DOM odpowiadające poszczególnym rysowanym kształtom. Element `<canvas>` służy do rysowania pikseli, a nie wektorów. Używanie tylko jednego węzła sprawia, że element `<canvas>` wydaje się wygodny w obsłudze dla przeglądarek mobilnych, jednak wysokie zapotrzebowanie na cykle procesora ze strony animacji JavaScriptu może szybko wyczerpywać baterie urządzeń przenośnych (choć dzięki akceleracji sprzętowej zużycie baterii jest niższe).

Pierwszy element `<canvas>`

Ponieważ jest to bardzo podstawowe wprowadzenie do elementu `<canvas>`, omawiam tu tylko proste kształty i linie. Jeśli nie znasz JavaScriptu, składnia może być początkowo mało zrozumiała. Jeżeli jednak znasz ten język, powinieneś rozumieć przedstawiony kod.

Pierwszy krok polega na dodaniu do dokumentu elementu `<canvas>`. W HTML-u jest to jedyna potrzebna operacja.

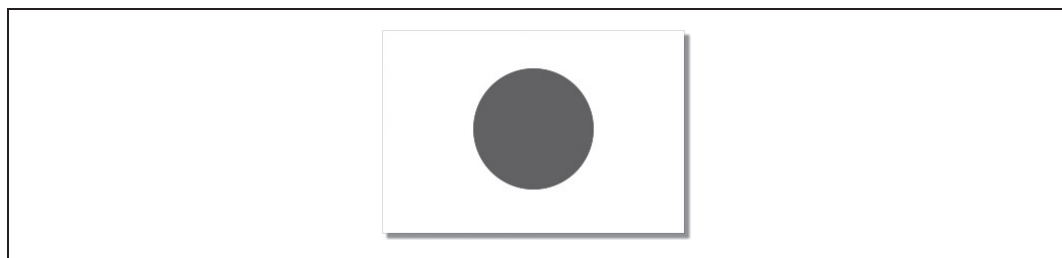
```
<canvas id="flag" width="320" height="220">
  Przeglądarka nie obsługuje elementu <canvas>. Gdyby obsługiwała, zobaczyłbyś flagę.
</canvas>
```

To już wszystko, co trzeba zrobić w HTML-u, aby dodać element `<canvas>`. Można napisać tylko `<canvas></canvas>`. Atrybut `id` jest używany, aby ułatwić wskazywanie elementu w JavaScriptcie, choć można też go wskazać za pomocą modelu DOM. Dodałam również zastępczy tekst dla użytkowników, których przeglądarka nie obsługuje elementu `<canvas>` lub którzy z innych przyczyn nie mogą zobaczyć tego elementu.



Element `<canvas>` w obecnej wersji to interfejs API, który nie zapewnia dostępności stron. Jedyna możliwość poprawienia dostępności to dodanie atrybutu `aria-label`.

Ten kod tworzy pustą tablicę do rysowania — płótno. Wszystkie pozostałe zadania wykonuje się w JavaScriptcie. Tu kod ponownie tworzy japońską flagę. Przedstawia ją rysunek 5.3.



Rysunek 5.3. Japońska flaga utworzona jako element `<canvas>`

Następny krok polega na utworzeniu rysunku na płótnie. Od tego momentu wszystko dzieje się w JavaScriptcie. Węzeł `<canvas>` można wskazać w prostym JavaScriptcie na trzy sposoby:

```
document.getElementById('flag')
document.getElementsByTagName('canvas')[0]
document.querySelector('#flag')
```

Potem należy zainicjować kontekst dwuwymiarowy i rozpocząć rysowanie za pomocą poleceń interfejsu API tego kontekstu. Kod rysuje tu japońską flagę.

```
1 <script>
2 var el= document.getElementById("flag");
3
4 if (el && el.getContext) {
5   var context = el.getContext('2d');
6   if (context) {
7     context.fillStyle = "#ffffff";
8     context.strokeStyle = "#cccccc";
9     context.lineWidth = 1;
10    context.shadowOffsetX = 5;
11    context.shadowOffsetY = 5;
12    context.shadowBlur = 4;
13    context.shadowColor = 'rgba(0, 0, 0, 0.4)';
14    context.strokeRect(10, 10, 300, 200);
15    context.fillRect(10, 10, 300, 200);
16    context.shadowColor='rgba(0,0,0,0)';
17    context.beginPath();
18    context.fillStyle = "#d60818";
19    context.arc(160, 107, 60, 0, Math.PI*2, false);
```

```

20     context.closePath();
21     context.fill();
22 }
23 }
24 </script>

```

Kod w wierszu 2 znajduje odpowiedni element `<canvas>` na podstawie atrybutu `id` tego elementu. Przed utworzeniem kontekstu dwuwymiarowego należy sprawdzić, że element `<canvas>` został odnaleziony *oraz* że przeglądarka obsługuje takie elementy. W tym celu w wierszu 4 kod sprawdza, czy dostępna jest metoda `getContext`.



Można wykorzystać skrypty do wykrywania funkcji (np. Modernizr — <http://modernizr.com/>) do sprawdzania, czy przeglądarka obsługuje element `<canvas>` i inne nowe mechanizmy. Modernizr pozwala wykrywać wszystkie funkcje lub tylko potrzebny w danym momencie mechanizm. Nie używam tu tego narzędzia, ponieważ chcę pokazać, jak bezpośrednio sprawdzać dostępność funkcji. Zwykle jednak warto stosować Modernizr, chyba że chcesz zminimalizować liczbę używanych zewnętrznych skryptów i żądań HTTP.

W wierszu 5 kod tworzy referencję do kontekstu, wywołując metodę `getContext(contextId)` elementu `<canvas>`. Właściwym kontekstem dla tego elementu jest `2d`. Jeśli tworzenie kontekstu kończy się powodzeniem (co kod sprawdza w wierszu 6), można wreszcie przejść do rysowania, za co odpowiada reszta skryptu.

Wprowadzie jest to technika eksperymentalna, ale jeśli chcesz w przeglądarce z silnikiem WebKit umieścić rysunek z elementu `<canvas>` jako tło za pomocą stylów CSS (<http://updates.html5rocks.com/2012/12/Canvas-driven-background-images>)⁴, to zamiast wywoływać element `<canvas>` w modelu DOM, możesz dodać rysunek jako grafikę tła. W stylach CSS należy zastosować następujący kod:

```
background: -webkit-canvas(theCanvas);
```

W JavaScriptcie umieść poniższe wywołanie:

```
var context = document.getCSSCanvasContext("2d", "theCanvas", 320, 220);
```

Drugi parametr jest tu nazwą używanego elementu `<canvas>`, którą bez cudzysłowu podano w stylach CSS.

Do wiersza 6, a nawet do wiersza 13 kod nic nie rysuje. Do wiersza 6 definiowany jest jedynie kontekst elementu `<canvas>`, który umożliwia rysowanie i modyfikowanie pikseli.

Przed narysowaniem kształtu trzeba zdefiniować jego wygląd i styl. W tym celu należy ustawić właściwości w obiekcie `context`. Tu kod definiuje wygląd obramowania (właściwości `stroke` i `linewidth`), kolor tła (właściwość `fill`) i cień (właściwości `shadowOffsetX`, `shadowOffsetY`, `shadowBlur` i `shadowColor`) dla pierwszego prostokąta, który jest rysowany za pomocą metody `strokeRect()` w wierszu 14. Przekazywane są tu te same parametry co w przykładowym pliku SVG (10, 10, 300, 200). Te cztery wartości określają współrzędne *x* i *y* oraz szerokość i wysokość.

⁴ Firefox 4 i nowsze wersje tej przeglądarki także obsługują element `<canvas>` w stylach CSS. Pozwala to dynamicznie tworzyć wirtualny element `<canvas>` za pomocą JavaScriptu i wywołania `-moz-element('#elementCanvas')` w stylach CSS.

Po wykonaniu polecenia skrypt „zapomina” o tym, co zrobił wcześniej, i przechodzi do następnego wiersza. Inaczej niż w przykładowym pliku SVG z wcześniejszego punktu, tu prostokąt jest rysowany w elemencie `<canvas>` i nie jest częścią modelu DOM. Właściwości `stroke`, `fill`, `linewidth` i `border` są zapamiętywane, jednak przeglądarka i skrypt nie potrafią określić, co zostało narysowane. Jeśli chcesz rejestrować, co i w którym miejscu jest rysowane w elemencie `<canvas>`, wykorzystaj metodę `getImageData()` kontekstu do zapisania wartości składowych czerwonej, zielonej i niebieskiej oraz kanału alfa pikseli.

W wierszu 15 kod wywołuje metodę `fillRect`, co prowadzi do narysowania drugiego prostokąta za pomocą ustawionej wcześniej właściwości `fillStyle`. Trzeba tu ponownie podać współrzędne, ponieważ pierwszy prostokąt nie jest zapamiętywany w modelu DOM (choć można uzyskać dostęp do informacji o pikselach).

W obu wywołaniach metod tworzących prostokąty (wiersze 14 i 15) używane są identyczne parametry — 10, 10, 300 i 200. Powoduje to narysowanie prostokąta z wypełnieniem bezpośrednim na prostokącie reprezentującym cień. Można utworzyć obiekt ze współrzędnymi i przekazać go do obu metod, jednak nie da się za pomocą elementu `<canvas>` uzyskać dostępu do współrzędnych pierwszego prostokąta i skopiować ich do drugiego.

Kod najpierw rysuje prostokąt, a potem go wypełnia. Gdyby zastąpić kolejność tych operacji, cień znajdowałby się nad kolorem tła. Ponieważ współrzędne są identyczne, a obramowanie jest ustawione na 1 piksel szerokości, ostatecznie ramka ma 0,5 piksela, ponieważ wypełnienie zajmuje wewnętrzną połowę jej szerokości.

Jak wcześniej wspomniałam, w momencie rozpoczęcia rysowania kółka (słońca) na fladze, w modelu DOM nie ma informacji o narysowanych wcześniej elementach. To prawda, JavaScript zapamiętuje ustawione wartości właściwości, np. `shadowColor`. Zapamiętuje też ostatnie operacje niezależnie od tego, czy dana figura została narysowana. Jednak piksele w elemencie `<canvas>` to tylko punkty o określonym kolorze. Ponieważ czerwone kółko nie ma mieć cienia, przed narysowaniem kółka trzeba ustawić właściwość `shadowColor` tak, aby była przezroczysta, co robi kod w wierszu 16.

Instrukcje dotyczące rysowania kółka rozpoczynają się od wywołania `beginPath()` (wiersz 17), a kończą wywołaniem `closePath()` (wiersz 20). Skrypt zapamiętuje operacje rysowania niezależnie od tego, czy element został już narysowany. Jeśli rysujesz kółko, a następnie bez otwierania i zamykania kontekstu dodasz kilka linii, to w momencie rysowania linii kółko nadal będzie w pamięci. Nowa linia może wtedy przechodzić przez kółko i przecinać je na pół. Tu unikam tego dzięki otwarciu i zamknięciu ścieżki za pomocą wywoływań `beginPath()` i `closePath()`.

Kółko jest definiowane za pomocą wywołania `context.arc(współrzędna_x, współrzędna_y, promień, kąt_początkowy, kąt_końcowy, niezgodnie_z_ruchem_wskazówek zegara)`. Powoduje ono dodanie punktów do krzywej, w wyniku czego powstaje wirtualny okrąg opisany za pomocą argumentów wywołania. Tu tym wywołaniem jest `context.arc(160, 107, 60, 0, Math.PI*2, false);`. Rysowanie zaczyna się od określonego kąta początkowego (tu jest to 0, czyli pozioma linia biegnąca w prawo) i kończy się podanym kątem końcowym. Rysowanie odbywa się zgodnie z określonym kierunkiem (tu zgodnie z ruchem wskazówek zegara). Gdyby *kąt końcowy* był mniejszy niż 2π , „kółko” byłoby przycięte — punkty początkowy i końcowy byłyby połączone prostą linią. Wartość π powoduje utworzenie półokręgu.

Ponadto w wierszu 18. zmieniany jest kolor wypełnienia (z białego na czerwony). Następnie kod wypełnia utworzone kółko za pomocą metody `fill()` (wiersz 21) według koloru ustawionego we właściwości `fillStyle`.

Nie opisałam tu nawet podstaw możliwości, jakie daje element `<canvas>`. Pod adresem <http://ie.microsoft.com/testdrive/Graphics/CanvasPad/Default.html> znajdziesz ciekawą stronę, która pomaga w nauce używania prostych kształtów, kolorów, cieni, tekstu, rysunków, transformacji, animacji i przesuwania kursora w elemencie `<canvas>`.

Przykładowy kod wykorzystujący element `<canvas>`

W przykładowej grze w celu zwiększenia trudności gry na wyższych poziomach zmieniamy tło planszy. W animacji pojawiają się kształty z przednich stron kart, przez co najwyższe poziomy są niezwykle trudne.

W internetowych materiałach do tego rozdziału (<http://www.standardista.com/mobile/>) znajduje się kilka przykładowych fragmentów kodu wykorzystujących element `<canvas>`. Spróbuj narysować japońską flagę, a następnie przekształcić ją w statyczny rysunek Pac-Mana (czerwone kółko zmień w żółte, a następnie dodaj buźkę, czarne kółko jako oczy, a obok kółka trzy małe kropki do zjedzenia). W przykładach z materiałów znajdziesz też tekst z kodem wykraczającym poza zakres tej książki. Opisana jest w nim funkcja zmieniająca kolory Pac-Mana. Pomoże Ci to nauczyć się, jak uzyskać dostęp do pikseli narysowanych już w elemencie `<canvas>`. Poznasz też inne metody rysowania na stronie.

Choć w tej książce podałam przykład wykorzystania elementu `<canvas>` w grze CubeeDoo, obecnie nie stosowałabym dynamicznie generowanych elementów `<canvas>` w produkcyjnej wersji mobilnej aplikacji sieciowej. Przy dzisiejszym stanie technologii zużycie baterii w wyniku wykonywania animacji w elemencie `<canvas>` za pomocą JavaScriptu byłoby dla użytkowników bardzo nieprzyjemne, jednak akceleracja sprzętowa dla elementu `<canvas>` jest coraz lepsza.

Element `<canvas>` a element `<svg>`

Elementy `<canvas>` i `<svg>` mają w HTML5 kilka podobnych cech. Często porównuje się ze sobą te elementy. Oba związane są z technologiami sieciowymi, które pozwalają tworzyć zaawansowane grafiki w przeglądarkach. Jednak między tymi technologiami występują poważne różnice.

W formacie SVG rysowanie odbywa się za pomocą XML-a. W elementach `<canvas>` używany jest do tego JavaScript. Przy używaniu tego elementu piksele są rysowane na płótnie, a po ich dodaniu każdy piksel jest zapominany. W SVG powstają węzły modelu DOM. Można do nich uzyskać dostęp do momentu ich usunięcia lub opuszczenia strony przez użytkownika. Obie te technologie mają wady i zalety.

Rysunki w formacie SVG dostosowują się do dowolnej rozdzielczości, dzięki czemu doskonale nadają się do interfejsów o dowolnych wymiarach (można skalować je pod kątem rozdzielczości każdego ekranu). Rysunki w tym formacie są zapisywane w plikach XML, co pozwala łatwo zwiększyć dostępność stron. Rysunki SVG można animować albo za pomocą składni deklaratywnej, albo przy użyciu JavaScriptu. Każdy narysowany element jest częścią modelu SVG DOM i można uzyskać dostęp do niego w JavaScriptcie. Jednak korzystanie z modelu DOM spowalnia działanie strony, co jest ważne i odczuwalne zwłaszcza w środowisku mobilnym.

W elementach `<canvas>` wszystko jest rysowane za pomocą pikseli. Zwiększenie rysunku może prowadzić do powstania efektu pikselizacji. Element `<canvas>` zmniejsza dostępność stron. Można jedynie dodać rezerwowy tekst pokazywany, gdy nie da się wyświetlić tego elementu. Aby umożliwić interakcje, trzeba ponownie narysować każdy piksel. Narysowane elementy nie są dostępne jako węzły DOM. Nie ma też interfejsu API do tworzenia animacji. Zamiast tego zwykle używa się zegarów lub wywołań `requestAnimationFrame`, aby aktualizować element `<canvas>` w krótkich odstępach czasu. Element `<canvas>` zapewnia powierzchnię, po której można rysować za pomocą interfejsu API wybranego kontekstu. Ten element doskonale nadaje się do edytowania grafiki, generowania grafiki rastrowej na potrzeby gier lub tworzenia fraktali, a także do wykonywania operacji wymagających manipulowania danymi na poziomie pikseli. Rysunki utworzone za pomocą interfejsu API elementu `<canvas>` można też eksportować.

Dwuwymiarowy kontekst elementu `<canvas>` jest dobrze obsługiwany we wszystkich przeglądarkach (od Internet Explorera 9). Także grafika SVG jest poprawnie interpretowana, ale w różnych formatach (od Internet Explorera 9 i Androida 3). Choć obie te technologie są poprawnie obsługiwane, obie mają też pewne wady.

Wydajność grafiki SVG bywa niska. Przeglądarki mobilne mają trudności z obsługiwaniem dużej liczby elementów modelu DOM. Każdy dodatkowy węzeł modelu DOM zajmuje pamięć i wymaga ponownego przetworzenia w momencie zmiany układu strony. Dlatego przy tworzeniu aplikacji na przeglądarki mobilne należy ograniczać liczbę dodawanych węzłów modelu DOM. Rysunki SVG składają się z węzłów modelu DOM. Większa liczba takich węzłów może prowadzić do spadku wydajności, a w skrajnych przypadkach nawet do zawieszenia się przeglądarki mobilnej. Z kolei element `<canvas>` przy stosowaniu animacji (w odróżnieniu od generowania pojedynczych rysunków) może szybko zużyć baterię urządzenia. Jednak we wszystkich popularnych przeglądarkach dostępna jest akceleracja sprzętowa dla tego elementu, dzięki czemu czas rysowania i aktualizowania jest znacznie krótszy, a zużycie baterii — mniejsze.

Sam uwzględnij wady i zalety obu technologii, zanim zdecydujesz się na zastosowanie jednej z nich (lub zrezygnowanie z obu).

WebGL

Obsługa grafiki trójwymiarowej i technologii WebGL jest dopiero wprowadzana. Obecnie występują poważne problemy z wydajnością, znacznym zużyciem baterii i ograniczoną obsługą w urządzeniach przenośnych. W czasie, gdy powstawała ta książka, najlepszą obsługę technologii WebGL zapewniał system BlackBerry 10. Niedawno obsługę tej technologii dodano także do systemu Firefox OS. Przy korzystaniu ze starszych urządzeń przenośnych warto dobrze się zastanowić nad stosowaniem tej technologii. Duże obciążenie procesora prowadzi do szybkiego zużycia baterii, a JavaScript mocno obciąża procesor. Nie chcesz przecież wyczerpać baterii. W urządzeniach obsługujących technologię WebGL (np. w urządzeniach z systemem BlackBerry 10) do przetwarzania grafiki w tym formacie używany jest procesor graficzny. Pozwala to zwiększyć wydajność i ograniczyć zużycie baterii w porównaniu z używaniem zwykłego procesora, jednak i tak waham się z zachęcaniem do korzystania z tej technologii. Jeśli zdecydujesz się użyć WebGL, zawsze pamiętaj o wydajności (np. o ilości zajmowanej pamięci i zużyciu baterii).

Elementy <audio> i <video>

Przed pojawieniem się HTML5 nie istniał standardowy sposób osadzania filmów wideo na stronach internetowych. Zamiast tego filmy z internetu wyświetlane były za pomocą niezależnych wtyczek, takich jak Flash lub QuickTime. Ponadto nie było łatwego sposobu na tworzenie dostępnych multimediów. Dlatego dołączone pliki wideo były często niedostępne dla osób z wadami wzroku i słuchu.

W HTML5 zdefiniowany jest standardowy sposób osadzania plików dźwiękowych i wideo na stronach internetowych. Służą do tego elementy <video> i <audio>. Oba są obsługiwane we wszystkich popularnych przeglądarkach mobilnych (z wyjątkiem Opery Mini), jednak na razie nie wszystkie przeglądarki obsługują te same formaty filmów. Przed omówieniem dołączania nagrań dźwiękowych i wideo do dokumentu trzeba opisać kodeki plików multimedialnych oraz obsługę multimediów w przeglądarkach. Jest to konieczne, ponieważ w różnych przeglądarkach należy udostępniać odmienne pliki multimedialne. Ponadto trzeba zapewniać rozwiązania rezerwowe dla przeglądarek, które nie obsługują multimediów określonego typu.

Typy plików multimedialnych

Gdy przeglądarki będą obsługiwać elementy <video> i <audio> z HTML5 oraz standardowe typy plików multimedialnych, nie trzeba będzie stosować niezależnych wtyczek do odtwarzania multimediów. Jednak na razie poszczególne przeglądarki obsługują różne kodeki audio i wideo. Jak prawdopodobnie wiesz, na iPadach i iPhone'ach nie działa Flash. Działają za to elementy <video> i <audio> oraz formaty H.264 (dla wideo) i AAC (dla dźwięku). Te formaty omawiam w następnym punkcie. Wszystkie nowe przeglądarki obsługują element <video> z HTML5, przy czym robią to dla różnych formatów. Internet Explorer 9, Safari, Chrome, Android oraz iOS obsługują format MPEG4/H.264 (pliki .mp4), natomiast Firefox 4 i jego nowsze wersje, Chrome, Opera oraz Android 2.3 i nowsze wersje poprawnie interpretują format WebM/VP8 (pliki .webm). Ten ostatni format działa także także w Internet Explorerze 9, jeśli w systemie zainstalowane są potrzebne kodeki. Listę obsługiwanych formatów znajdziesz w tabeli 5.1.

Tabela 5.1. Obsługa kodeków wideo w przeglądarkach (w przeglądarce Internet Explorer 9 można odrębnie zainstalować kodeki Ogg i WebM)

	iPhone i iPad	Android	BlackBerry	Opera Mobile	Opera Mini	Windows (Internet Explorer)	Chrome w Androidzie	Firefox w Androidzie
<video>	Tak	Tak	7	11		9	Tak	Tak
Ogg		2		11		(9*)		Tak
H.264	Tak	3.0 ^a	7			9	Tak	Tak*
WebM		2.3		14		(9*)	Tak	Tak

^a Zobacz stronę <http://www.broken-links.com/2010/07/08/making-html5-video-work-on-android-phones/>.

Istnieje kilka kodeków wideo. Najważniejsze z nich to: Theora/Ogg, VP8 i H.264. Theora/Ogg (pliki .ogg) to otwarty standard z wbudowaną obsługą w przeglądarkach Firefox 3.5, Chrome 4 i Opera 10.5 oraz nowszych. W Internet Explorerze ten format jest obsługiwany po zainstalowaniu wtyczki. WebM to nowszy format używany razem z kodekiem wideo VP8. Wbudowana obsługa tego formatu jest dostępna w najnowszych wersjach przeglądarek Chrome i Mozilla Firefox oraz w Operze 10.6 i nowszych.

Obecnie kodek VP8 jest dostępny bez opłat licencyjnych. Kodek ten jest opatentowany, jednak właściciel patentu (firma Google) udostępnia ten kodek bezpłatnie. Niestety, choć jest on dobrze obsługiwany w nowych przeglądarkach, prawa do patentów związanych z formatem WebM i kodekiem VP8 rości sobie także Nokia. Dlatego w najbliższej przyszłości ten format prawdopodobnie nie stanie się standardem sieciowym.

Format H.264 jest przeznaczony dla urządzeń o niskiej, umiarkowanej i wysokiej przepustowości. Można go odtwarzać za pomocą technologii Adobe Flash oraz w różnych urządzeniach przenośnych (m.in. w systemie Android i w iPhone'ach). Nie jest to jednak otwarty standard. Wykupienie licencji może być kosztowne. Producent przeglądarki Chrome zapowiedział, że zrezygnuje z obsługi tego formatu, jednak na razie tego nie zrobił. W Firefoksie ten format jest obsługiwany, jeśli kodek jest zainstalowany w używanym systemie operacyjnym. Do samo dotyczy Opery w urządzeniach przenośnych.

Na razie, co widać w tabeli 5.1, nie istnieje format działający we wszystkich przeglądarkach. Aby film można było odtwarzać wszędzie, trzeba zakodować go w więcej niż jednym formacie.

Obecnie przy tworzeniu witryn na telefony komórkowe w Stanach Zjednoczonych najlepiej jest używać formatu H.264. Trzeba jednak pamiętać, że poziom obsługi tego formatu może się zmienić. Jeśli docelowe urządzenia przenośne obejmują systemy GPS, konsole do gier itd., warto pamiętać, że najpopularniejszą przeglądarką mobilną na świecie jest Opera. Także w Stanach Zjednoczonych jest to najczęściej używana przeglądarka w urządzeniach innych niż telefony i komputery.

Dodawanie elementu <video> do witryny

Wprawdzie jeśli film docelowo ma być wyświetlany na smartfonach w Stanach Zjednoczonych, wystarczy wykorzystać format H.264, to aby uzyskać maksymalną zgodność z różnymi urządzeniami, trzeba przygotować dwie wersje nagrania. Utwórz wersje w formatach WebM (WP8 dla wideo i Vorbis dla audio) i MP4 (H.264 dla wideo i AAC dla audio). Odnośniki do obu plików wideo podaj za pomocą elementu <video> z HTML5 i podrzędnych znaczników <source>. Jako domyślny ustaw odtwarzacz oparty na Flashu.

Atrybuty elementów <video> i <audio>

Jest kilka atrybutów elementów <video> i <audio>, które służą do kontrolowania wyglądu i działania osadzanych multimediiów.

Oto atrybuty obsługiwane w elementach <video> i <audio>:

src

Atrybut src przyjmuje jako wartość adres URL pliku wideo lub audio. Można go zastąpić kilkoma podrzędnymi znacznikami <source>.

autoplay

Atrybut logiczny autoplay informuje przeglądarkę, że odtwarzanie nagrania rozpoczyna się automatycznie (bez oczekiwania na wciśnięcie przycisku *Play* przez użytkownika). Ten atrybut warto stosować tylko na tych stronach, których główną zawartością jest odtwarzany film.

loop
Atrybut logiczny `loop` powoduje, że nagranie filmowe lub dźwiękowe jest odtwarzane w pętli. Gdy zostanie uruchomione, będzie odtwarzane w kółko do czasu wstrzymania lub zatrzymania. Jeśli ten atrybut jest używany, nagranie po dotarciu do końca jest uruchamiane ponownie.

controls
Jeśli używany jest ten atrybut logiczny, przeglądarka powinna wyświetlać kontrolki do sterowania multimediami (przyciski do przeskakiwania do różnych miejsc nagrania, włączania go, wstrzymywania itd.).

preload
Atrybut `preload` informuje przeglądarkę, jak dużą część filmu powinna pobrać przed rozpoczęciem jego odtwarzania. Jeśli nie podasz tego atrybutu lub ustawisz jego wartość na `none`, przeglądarka nie będzie wstępnie pobierać danych. Jeżeli ten atrybut jest używany i ma wartość `auto`, przeglądarka wstępnie pobiera plik. Ustawienie `metadata` powoduje, że przeglądarka pobiera wymiary, długość i inne metadane dotyczące nagrania, natomiast nie ściąga samego pliku multimedialnego.

Poniższe atrybuty dotyczą tylko elementu `<video>` (nie są obsługiwane w elemencie `<audio>`):

poster
Atrybut `poster` przyjmuje adres URL rysunku używanego jako graficzny wypełniacz do momentu rozpoczęcia odtwarzania filmu. Jeśli nie podasz tego atrybutu, odtwarzacz wyświetli pierwszą klatkę z filmu (zwykle jest to czarny prostokąt).

width
Wartość atrybutu `width` to szerokość (w pikselach) pola z filmem.

height
Wartość atrybutu `height` to wysokość (w pikselach) pola z filmem.

Oto przykładowa deklaracja elementu `<video>` (opis poszczególnych komponentów przedstawia tabela 5.2):

```
<video autoplay controls loop poster="poster.jpg" preload="metadata"
src="video.mp4" height="360" width="480">Tekst zastępczy</video>
```

Tabela 5.2. Komponenty przykładowej deklaracji elementu `<video>`

Komponent	Opis
<code><video></code>	Znacznik <code><video></code>
<code>autoplay</code>	Jeśli jest ustawiony, odtwarzanie filmu rozpoczyna się w momencie wczytania strony
<code>controls</code>	Jeśli jest ustawiony, dostępny jest pasek sterowania
<code>loop</code>	Jeśli jest ustawiony, video jest odtwarzane w pętli
<code>poster="img/poster.jpg"</code>	Jeśli jest ustawiony, wyświetlany jest podgląd nagrania
<code>preload="metadata"</code>	Przyjmuje wartości <code>none</code> , <code>metadata</code> i <code>auto</code>
<code>src="/video/video.mp4"</code>	Odnosnik do pliku wideo
<code>height="360"</code>	Wysokość obrazu w wideo
<code>width="480"></code>	Szerokość obrazu w wideo
Tekst zastępczy	Dowolny poprawny kod w HTML-u (standardowo podawany jest odnośnik do danego filmu)
<code></video></code>	Wymagany zamykający znacznik <code></video></code>

Elementy `<audio>` i `<video>` z HTML5 umożliwiają powiązanie etykiet z osadzonymi multimediami. W HTML5 te elementy są częścią modelu DOM, co umożliwia określanie dla nich stylów CSS i zapewnia rozbudowany interfejs API, który daje programistom kontrolę nad odtwarzaniem filmu przy użyciu licznych nowych metod i właściwości JavaScriptu (takich jak: `play()`, `pause()`, `muted` i `ended`).

Gdy element `<video>` z HTML5 będzie w pełni obsługiwany, a wszystkie przeglądarki będą udostępniać ten sam kodek, wystarczy następujący kod:

```
<video src="myVideo.mp4" width="400" height="300"
  controls poster="myImage.jpg">
  Przeglądarka nie obsługuje HTML5, ale możesz pobrać plik
  <a href="myVideo.ogv"> pod tym adresem</a>.
</video>
```

Niestety, na razie ten kod zadziała tylko w niektórych przeglądarkach. Jak wcześniej wspomniałam, nie wszystkie przeglądarki obsługują ten sam kodek, dlatego dla poszczególnych przeglądarek trzeba udostępnić różne pliki.

Aby było to możliwe, w HTML5 dostępny jest element `<source>`. Ten element pozwala wskazać więcej niż jeden plik multimedialny. Element `<source>` ma trzy atrybuty (oprócz atrybutów globalnych): `src`, `type` i `media`.



Aby dynamicznie zmieniać odtwarzane multimedia, zmodyfikuj atrybut `src` w znaczniku `<video>` lub `<audio>`. Nie wystarczy zmienić wartości atrybutu `src` w elemencie `<source>`. Aby wybrać typ plik obsługiwany przez przeglądarkę, wykorzystaj metodę `canPlayType()`.

Atrybut `type` określa typ zasobów multimedialnych, dzięki czemu przeglądarka przed pobraniem pliku może ustalić, czy obsługuje dany typ. Wartością atrybutu `type` musi być poprawny typ MIME.

Nawet w sytuacji, gdy nie wszystkie przeglądarki obsługują ten sam kodek, kod nie jest zanedbanie skomplikowany. W przykładowej grze mogliśmy dodać film z instrukcjami do gry. Nie zrobiliśmy tego, ale było to możliwe. Gdybyśmy to zrobili, kod zapewniający obsługę wideo we wszystkich przeglądarkach powinien wyglądać tak:

```
<video width="400" height="300" preload="none" poster="posterImg.jpg"
  controls>
  <source src="myVideo.mp4" type="video/mp4; codecs=avc1.42E01E, mp4a.40.2"/>
  <source src="myVideo.webm" type="video/webm; codecs=vp8, vorbis"/>
  <source src="myVideo.ogv" type="video/ogg; codecs=dirac, speex"/>
  <object width="400" height="324" type="application/x-shockwave-flash"
    data="myVideo.swf"/>
    <param name="movie" value="myVideo.swf"/>
    <param name="flashvars"
      value="image=posterImg.jpg&amp;file=myVideo.mp4"/>
  <!-- Rozwiązanie rezerwowe -->
  <a href="linktovideo">
    
  </a>
</object>
</video>
```

Jeśli przeglądarka obsługuje element `<video>` z HTML5, zostanie on użyty. Jeżeli przeglądarka nie obsługuje pierwszego typu multimediiów, sprawdzi następny. Gdy element `<video>` z HTML5 nie jest obsługiwany, przeglądarka używa technologii Adobe Flash. Jeśli brakuje obsługi zarówno Flasha, jak i elementu `<video>`, wyświetlana jest zastępcza grafika. Możesz też dodać odnośnik umożliwiający pobranie nagrania.



Film z pliku we Flashu jest zadeklarowany jako wyższy o 24 piksele od innych wersji. Wynika to z tego, że kontrolki Flasha zajmują 24 piksele w pionie pod oknem z filmem, natomiast w formatach wideo w HTML5 kontrolki są nakładane na obraz.

Jeśli aplikacja jest przeznaczona tylko na nowe urządzenia przenośne, można pominąć Flasha i dodać elementy `<track>` (opis tej techniki znajdziesz w podpunkcie „Element `<track>`”).

```
<video width="400" height="300" preload="none" poster=
"posterImg.jpg" controls>
<source src="myVideo.mp4" type="video/mp4;
  codecs=avc1.42E01E, mp4a.40.2"/>
<source src="myVideo.webm" type="video/webm;
  codecs=vp8, vorbis"/>
<source src="myVideo.ogv" type="video/ogg;
  codecs=theora, vorbis"/>

<track kind="subtitles" label="English" src="en.vtt"
  srclang="en" default/></track>
<track kind="subtitles" label="Deutsche" src="de.vtt"
  srclang="de"/></track>
</video>
```

Pliki wideo zwykle obejmują ścieżki audio i wideo. Ścieżki audio zawierają znaczniki umożliwiające synchronizację dźwięku z obrazem. Poszczególne ścieżki mogą zawierać metadane określające np. proporcje obrazu lub język ścieżki dźwiękowej. Także w kontenerach mogą znajdować się metadane (np. tytuł filmu, okładka płyty z filmem, numer odcinka serialu itd.).

Do dokumentu można też dodać element `<audio>`:

```
<audio id="sound">
<source src="music.mp3" type="audio/mp3"/>
<source src="music.ogg" type="audio/ogg"/>
<!-- Można też dodać wersję we Flashu na potrzeby przeglądarek bez obsługi elementu <audio> -->
</audio>
```

W artykule w serwisie Dev.Opera (<http://dev.opera.com/articles/view/everything-you-need-to-know-about-html5-video-and-audio/>) znajdziesz bardzo szczegółowe instrukcje na temat wykrywania obsługi elementów.

Element `<track>`

Aby pliki wideo i audio były dostępne dla użytkowników z wadami słuchu, a nawet dla osób, które nie rozumieją języka z nagrania, można dodać do filmu podpisy, wskazując plik z nimi w elemencie `<track>`.

Atrybut `src` w elemencie `<track>` (umieszcza się go w elementach `<video>` i `<audio>`) prowadzi do ścieżki z informacjami o czasie. Atrybut `kind` określa, jakiego rodzaju dane są wskazywane w atrybucie `src`. Wartości atrybutu `kind` to: `subtitles`, `captions`, `descriptions`, `chapters` i `metadata`.

W elemencie multimedialnym można zapisać wiele ścieżek, jednak każda z nich musi być unikatową kombinacją typu i języka. Oto typy ścieżek:

subtitles

Jest to wartość domyślna atrybutu `kind`. Określa, że ścieżka zawiera tłumaczenie dialogów, które domyślnie jest wyświetlane razem z filmem lub nagraniem dźwiękowym. Ten typ jest najprzydatniejszy, gdy rozmowy są niesłyszalne lub prowadzone w obcym języku.

captions

Określa ścieżkę z transkrypcją lub tłumaczeniem dialogów. Przypomina typ `subtitles`, ale obejmuje transkrypcję efektów dźwiękowych, uwagi na temat muzyki i inne informacje na temat dźwięku, które pozwalają w pełni zastąpić ścieżkę dźwiękową, jeśli jest niedostępna. Ten typ jest najprzydatniejszy, gdy dźwięk jest wyłączony lub gdy użytkownik ma wadę słuchu.

descriptions

Ścieżki tego typu to opis obrazu z odtwarzanych multimedii. Ten typ jest przeznaczony dla syntezytatorów dźwięku, gdy obraz jest niedostępny. Przydaje się dla osób z wadami wzroku i innych użytkowników, którzy nie mogą zobaczyć filmu lub odczytać tekstu ścieżki.

chapters

Określa ścieżkę z tytułami rozdziałów przeznaczoną do poruszania się po nagraniu.

metadata

Określa ścieżkę przeznaczoną dla skryptów (niewyświetlaną użytkownikom).

Plik ze ścieżką podaje się w atrybucie `src`. Atrybut `srclang` określa język tekstu danych z elementu `<track>`. Atrybut `label` to czytelny dla użytkownika tytuł elementu `<track>`. Przeglądarka używa tego tytułu do wyświetlania w interfejsie użytkownika ścieżek z napisami, transkrypcją i opisem obrazu.

Atrybut `default` informuje, że przeglądarka ma domyślnie zastosować dany element `<track>`, jeśli preferencje nie określają, iż bardziej odpowiedni jest inny element tego rodzaju. Jako domyślny można ustawić tylko jeden element `<track>`.

Elementy `<audio>` i `<video>` z HTML5 umożliwiają powiązanie napisów z osadzonymi multimediami. Te elementy są częścią modelu DOM dla HTML5, co pozwala dodawać do nich style CSS i zapewnia rozbudowany interfejs API zapewniający programistom kontrolę nad odtwarzaniem filmów. Dostępne są tu liczne nowe metody i właściwości JavaScriptu (np.: `play()`, `pause()`, `muted` i `ended`).

Elementy `<video>` i `<audio>` a JavaScript

Jeśli zamierzasz używać JavaScriptu do kontrolowania elementów `<audio>` i `<video>`, powinieneś wykrywać dostępne funkcje. W ten sposób upewnisz się, że potrzebne elementy są obsługiwane, oraz unikniesz błędów JavaScriptu.

```
if (createElement('audio').canPlayType) { /* Element audio jest obsługiwany */ }
```

Możesz dodać wbudowane kontrolki lub utworzyć własne. Elementy `<audio>` i `<video>` obsługują metody `play()` i `pause()`. Aby utworzyć własne kontrolki, możesz dodać odpowiadający im kod w HTML-u i użyć JavaScriptu do uruchamiania i wstrzymywania nagrań. Potrzebny kod może wyglądać tak:

```

<div id="controls" style="display: none">
  <button id="playButton">Start</button>
  <button id="pauseButton">Pauza</button>
</div>
<script>
  if (document.createElement('audio').canPlayType) {
    if (document.createElement('audio').canPlayType('audio/mp3') ||
        (document.createElement('audio').canPlayType('audio/ogg'))) {
      // Element <audio> z HTML5 i podany typ audio są obsługiwane
      document.getElementById('player').style.display = 'block';
    } else {
      ... Tu możesz umieścić nagrania we Flashu lub inne ...
    }
  }
</script>

```

W celu utworzenia własnych kontrolerek możesz zastosować następujący kod:

```

var videoClip = document.querySelector('#clip');
var playButton = document.querySelector('#playButton');
var pauseButton = document.querySelector('#pauseButton');

playButton.addEventListener('touchEnd', function() {
  playVideo();
});
pauseButton.addEventListener('touchEnd', function() {
  pauseVideo();
});

function playVideo() {
  // Odtwarzanie filmu
  videoClip.play();
  // Aktualizowanie kontrolerek
  playButton.disabled = true;
  pauseButton.disabled = false;
}

function pauseVideo() {
  // Wstrzymywanie filmu
  videoClip.pause();
  // Aktualizowanie kontrolerek
  playButton.disabled = false;
  pauseButton.disabled = true;
}

function MuteUnMute() {
  // Zmiana wartości przycisku
  document.getElementById('mute').value =
    videoClip.muted ? 'Wyłącz dźwięk' : 'Włącz dźwięk';
  // Zmiana stanu filmu
  videoClip.muted = videoClip.muted ? false : true;
}

```

CubeeDoo

W omawianej tu przykładowej grze użyliśmy kilku dźwięków. Oprócz odtwarzania w tle opcjonalnej irytującej muzyki gra generuje dźwięki informujące o powodzeniu lub porażce (np. po przejściu na następny poziom, dopasowaniu kart, nieudanym dopasowaniu itd.).

Muzykę odtwarzaną w tle należy umieścić w elemencie `<audio>`, ponieważ jest kontrolowana przez użytkownika. Dźwięki informacyjne zależą od poczynań użytkownika i powodzenia lub porażki, dlatego są dynamicznie generowane za pomocą JavaScriptu.

Oto dwie techniki, które można stosować do dodawania dźwięku. Dźwięk można dołączyć bezpośrednio w HTML-u:

```
<audio id="nonmatchsound" preload src="notmatch.mp3"></audio>
<audio id="matchsound" preload src="match.mp3"></audio>
```

Program wstępnie wczytuje nagranie, ale nie odtwarza plików dźwiękowych automatycznie ani nie odgrywa ich w pętli. Zamiast tego kod w JavaScriptcie uruchamia odtwarzanie dźwięków informujących o dopasowanej lub niedopasowanej karcie:

```
playSound: function(matched) {
  if (qbdoo.mute) {
    return false;
  }
  if (matched) {
    qbdoo.matchfound.play();
  } else {
    qbdoo.failedmatch.play();
  }
},
```

Dźwięku nie trzeba dołączać w HTML-u. Zamiast tego można dodać go do modelu DOM za pomocą JavaScriptu bez dołączania plików dźwiękowych na stronie:

```
playSound: function(matched) {
  // Jeśli dźwięk jest wyłączony, należy przejść dalej
  if (qbdoo.mute) {
    return false;
  }
  // Jeśli węzeł audio nie istnieje, należy go utworzyć
  if (!qbdoo.audio) {
    qbdoo.audio = document.createElement('audio')
  }
  if (matched) {
    qbdoo.audio.src = qbdoo.matchedSound;
  }
  else {
    qbdoo.audio.src = qbdoo.failedMatchSound;
  }
  qbdoo.audio.play();
},
```

W grze dodaliśmy dźwięk tylko w celu zademonstrowania posługiwania się elementem `<audio>`. Nigdy nie należy automatycznie odtwarzać muzyki — irytuje to użytkowników. Zauważ, że gra zawiera przycisk wyciszania. Jeśli udostępniasz dźwięk i domyślnie jest on włączony, to gdy użytkownik zdecyduje się go wyłączyć, zapamiętaj to ustawienie w pamięci lokalnej (pamięć lokalną omawiam w rozdziale 6.).

Określanie stylu elementu `<video>`

Element `<video>` jest elementem HTML-a. Można więc określić jego styl (podobnie jak wszystkich elementów). Możesz wykorzystać style CSS do zdefiniowania wysokości i szerokości obrazu. Możesz też ukryć fragment nagrania, dodać zaokrąglone narożniki, a nawet powielić obraz. Za pomocą elementu `<canvas>` można pobrać piksele i zmienić ich kolor (ten sam efekt można osiągnąć za pomocą filtrów CSS).

Dynamiczne dopasowywanie wielkości obrazu

Co ważniejsze, można zmieniać wymiary obrazu na podstawie wielkości urządzenia i proporcji ekranu. Thierry Koblentz zaproponował skuteczną metodę (<http://alistapart.com/article/creating-intrinsic-ratios-for-video>) umożliwiającą przeglądarkom określanie wymiarów obrazu na podstawie szerokości bloku zawierającego film (lub szerokości strony) i pierwotnej wielkości. Zmiana szerokości, np. w wyniku zmiany orientacji urządzenia, prowadzi do ponownego wyznaczenia wysokości. To umożliwia zmianę wielkości obrazu i pozwala skalować filmy w taki sam sposób jak rysunki.

Aby utworzyć film pozwalający na zmianę wymiarów, należy przygotować pole o zmiennej wielkości i odpowiednich proporcjach (4:3, 16:9 itd.). Następnie film trzeba rozciągnąć, aby dopasować go do wymiarów pola. Można przy tym wykorzystać marginesy wewnętrzne (właściwości z rodziny padding), procentowe wartości wymiarów i bezwzględne określanie pozycji. Marginesy wewnętrzne są zwykle ustawiane na 56,25% lub 75% wielkości okna (w zależności od proporcji). W tym modelu można bezwzględnie określić pozycję elementu `<video>` w taki sposób, aby zajmował całą wysokość i szerokość obszaru wyznaczanego przez wewnętrzne marginesy.

Jeśli chcesz dodać filmy o zmiennej wielkości do witryny z dynamicznie dopasowywanym rozmiarem, możesz zastosować następujący kod:

```
.wrapper {
  position: relative;
  height: 0;
  width: 100%;
  padding-bottom: 56.25%;
  /* * Lub */
  padding-bottom: 75%;
}
video {
  position: absolute;
  width: 100%;
  height: 100%;
  left: 0;
  top: 0;
}
```

Co warto wiedzieć o stosowaniu elementu `<video>`?

W odróżnieniu od Flasha, który jest obsługiwany za pomocą wtyczki kontrolowanej przez jedną firmę, dlatego wszędzie działa w ten sam sposób, element `<video>` funkcjonuje nieco inaczej w poszczególnych przeglądarkach i systemach operacyjnych. Na iPhone'ach oraz w systemach Android i Windows Phone 8 filmy zawsze są odtwarzane w trybie pełnoekranowym. Na iPadach dostępna jest kontrolka trybu pełnoekranowego, a wielkość obrazu można zmieniać także za pomocą gestu „szczypania”. W systemach iOS i Windows do odtwarzania filmów używany jest procesor graficzny, natomiast w Androidzie do wersji 4 filmy były wyświetlane przy użyciu głównego procesora.

- Upewnij się, że używany serwer obsługuje potrzebne typy MIME wideo. Brak takiej obsługi może prowadzić do problemów w Firefoksie. Dodaj wiersz `AddType video/ogg.ogv` i podobne wpisy do pliku `.htaccess`, jeśli ich w nim nie ma.
- Urządzenia iPhone i iPad nie odtwarzają automatycznie filmów, nawet jeśli dodasz atrybut `autoplay`.

- Wygląd kontrolek zależy od natywnych kontrolek przeglądarki. Jak wcześniej wspomniałam, wygląd i styl kontrolek można zmienić za pomocą JavaScriptu. Jeśli chcesz dodać skórki dla kontrolek, zajrzyj na stronę <http://www.videojs.com/>.
- Jeżeli chcesz zacząć udostępniać własne filmy, istnieje otwarty, wieloplatformowy transkoder wideo z obsługą wielowątkowości. Jego nazwa to Handbrake (<http://handbrake.fr/>) i jest dostępny na licencji GPL w systemach Mac OS X, Linux i Windows.

Pamiętaj, że odtwarzanie nagrań audio i wideo szybko wyczerpuje baterie. Choć multimedia tego typu są obsługiwane we wszystkich smartfonach, większe możliwości wymagają więcej odpowiedzialności. Powinieneś zadbać o to, aby tworzone przez Ciebie aplikacje nie zużywały baterii w urządzeniach użytkowników. Zachowaj ostrożność przy stosowaniu funkcji wyczerpujących baterie.

- !important, 206
 - specyficzność, 233
- #id, 211
- #myParent a {}, 212
- .appcache, 167
- .copyright .urgent {}, 212
- :transform
 - matrix(), 309
- @keyframes, 317
- @media, 69
 - aspect-ratio, 200
 - device-aspect-ratio, 200
 - device-height, 200
 - device-width, 200
 - height, 200
 - orientation, 200
 - width, 200
 - właściwości dla urządzeń przenośnych, 200
 - wybór skrajnych wymiarów, 201
 - wykrywanie
 - obsługi animacji i przejść, 339
 - wybranych funkcji, 339
- @supports, 201
 - tworzenie układu opartego na modelu flexbox, 340
 - wykrywanie funkcji, 339
- @viewport, 368
- <a>, 86, 95
 - download, 87
 - media, 87
 - name, 87
 - ping, 87
 - target, 87
- <abbr>, 89, 95
- <address>, 75, 81
- <area>, 55, 92
- <article>, 77, 78
- <aside>, 79, 80
- <audio>, 154
 - a JavaScript, 159
 - atrybuty, 155, 156
 - powiązanie etykiet z osadzonymi multimediami, 157
- , 88
- <base>, 55, 67
- <bdi>, 86
- <bdo dir="rtl">, 86
- <bdo>, 89
- <body>, 57, 60, 61, 75
 - atrybuty, 61
 - odbiorniki zdarzeń click i touchend, 374
 - z elementami strony, 72
-
, 55, 89
- <button>, 114, 139
 - button, 139
 - reset, 139
 - submit, 139
- <canvas>, 148, 152
 - aktualizowanie, 153
 - dodanie do dokumentu, 148
 - dwuwymiarowy kontekst, 153
 - id, 149, 150
- <circle>, 143
- <cite>, 89
- <code>, 89
- <col>, 55, 95
- <colspan>, 95
- <command>, 55
- <datalist>, 132, 134
 - łagodna degradacja, 133
- , 89, 95
- <desc>, 143, 144
- <details>, 92
 - open, 92
- <dfn>, 89
- <div>, 77, 115
- , 89

- <embed>, 55, 92, 143
 - src, 92
 - type, 92
- <fieldset>, 138
- <figcaption>, 82
- <figure>, 82
- <footer>, 76, 80
- <form>, 97, 111, 137
 - autocomplete, 138
 - onreset, 112
- <h1>, 75
- <head>, 57, 60, 62
- <header>, 76, 79
 - dla bloku <section>, 79
 - podstawowy, 79
- <hr>, 55, 83
- <html>, 57, 59
 - manifest, 60, 166, 170
 - wymagane atrybuty na stronach różnego typu, 59
- <i>, 88
- <iframe>, 90
 - sandbox, 91
 - seamless, 91
 - srcdoc, 90
- , 55, 91, 114, 143
- <input>, 55, 98, 99, 108, 115, 128, 232
 - atrybuty, 99
 - autocomplete, 106
 - autofocus, 107
 - color, 103
 - data i godzina, 123
 - disabled, 104, 105
 - email, 132
 - form, 105, 106
 - list, 132
 - max, 100
 - maxlength, 105
 - min, 100
 - nowe typy, 115
 - określanie stylów dla elementów, 114
 - pattern, 102
 - placeholder, 101, 102
 - readonly, 104, 105
 - required, 99
 - size, 105
 - step, 101
 - text, 132
 - type, 98, 99, 115
 - typy i atrybuty elementu, 107, 126
 - url, 132
 - userid, 106
- <ins>, 89
- <ką>, 266
- <kbd>, 89
- <keygen>, 55, 137
 - autofocus, 137
 - challenge, 137
 - disabled, 137
 - form, 137
 - keytype, 137
 - name, 137
- <label>, 134, 139
 - dostępność formularzy, 102
 - for, 139
 - id, 139
- <legend>, 138
-
 - value, 83
- <link>, 55, 67, 360
 - atrybuty, 68
 - dla arkuszy stylów, 68
 - niedostępne atrybuty, 69
 - media, 68, 199
 - all, 198
 - rel, 69, 364
 - shortcut icon, 68
 - stylesheet, 198
 - src, 198
 - type, 198
- <main>, 82
- <mark>, 84
- <menu>, 94
 - id, 94
 - label, 94
 - type, 94
- <menuitem>, 94
 - label, 94
 - title, 94
 - type, 94
- <meta>, 55, 57, 63, 378
 - apple-mobile-web-app-capable, 66
 - apple-mobile-web-app-status-bar-style, 66
 - charset="UTF-8", 63
 - content, 63, 361
 - description, 63
 - dla aplikacji mobilnych, 64
 - format-detection, 67
 - http-equiv, 63
 - keyword, 64
 - name, 63, 64
 - styl paska stanu, 361
 - viewport, 64, 367, 368
 - content, 367
 - web-app-capable, 361

<meter>, 135, 136
 high, 135
 low, 135
 max, 135
 min, 135
 optimum, 135, 136
 <nav>, 78
 <noscript>, 72
 <object>, 91, 92, 95, 143, 145
 plik SVG, 145
 , 95
 reversed, 83
 type, 83
 <optgroup>, 138
 <option>, 132, 134, 138
 <output>, 114, 134, 135
 for, 134
 form, 134
 name, 134
 onchange, 134
 onformchange, 134
 onforminput, 134
 <p>, 134
 <param>, 55, 92
 <progress>, 136
 max, 136
 value, 136
 <q>, 89
 <rect>, 143
 <rp>, 85
 <rt>, 85
 <ruby>, 85
 <s>, 88
 <samp>, 89
 <script>, 71
 async, 72
 defer, 72
 dodawanie do stron internetowych, 71
 src, 71
 type, 71
 wydajność JavaScriptu, 71
 <section>, 77, 78
 <select>, 132, 134, 138
 <small>, 89
 <source>, 55
 atributy, 157
 type, 157
 , 89, 134, 193
 , 89, 95
 <style>, 70, 197
 media, 70
 scoped, 70
 <sub>, 89
 <summary>, 92
 odbiornik zdarzeń, 94
 <sup>, 89
 <svg>, 142
 porównanie z <canvas>, 152
 <table>, 95
 <td>, 95
 <textarea>, 138
 cols, 138
 hard, 138
 rows, 138
 soft, 138
 wrap to hard, 138
 <th>, 95
 <thead>, 95
 <time>, 85
 datetime, 85
 <title>, 57, 60, 143, 144
 <tr>, 95
 <track>, 55, 158
 default, 159
 kind, 158
 label, 159
 src, 158
 srclang, 159
 <u>, 89
 , 189
 <var>, 89
 <video>, 154
 a JavaScript, 159
 atributy, 155
 autoplay, 155
 controls, 156
 dodawanie do witryny, 155
 funkcjonowanie w przeglądarkach
 i systemach operacyjnych, 162
 height, 156
 komponenty przykładowej deklaracji, 156
 loop, 156
 określanie stylu elementu, 161
 poster, 156
 powiązanie etykiet z osadzonymi
 multimediami, 157
 preload, 156
 src, 155
 width, 156
 <wbr>, 55, 86

A

AAC, 154
 Aardwolf, 37
 Accessible Rich Internet Applications, 18, 51, 191
 active-matrix organic light-emitting diode, 382

- ADB, 32
- addEventListener, 368
- Adobe Edge Inspect, 35, 391
- Adobe PhoneGap Build, 378
- adres URL, plik manifestu, 60
- akceleracja sprzętowa, 385
 - dla elementu <canvas>, 153
 - obliczanie układu elementów, 386
 - ponowne wyświetlanie, 386
- akcelerometr, 375
- allow-forms, 91
- allow-pointer-lock, 91
- allow-popups, 91
- allow-same-origin, 91
- allow-scripts, 91
- allow-top-navigation, 91
- Amaya, 146
- Ambient Light Events, 377
- analizowanie żądań sieciowych, 391
- Android, 41
- Android 2.3, 27
- Android Debug Bridge, 32
- Android Debug Monitor, 33
 - Android Virtual Device Manager, 34
- animacje, 315
 - akceleracja sprzętowa, 385
 - cykl, 316
 - fade, 322
 - klatki kluczowe, 317, 318
 - opóźnienie odtwarzania, 316
 - płynność, 402
 - skrótowy zapis właściwości, 316
 - stosowanie, 318
 - właściwości, 315
 - punkt środkowy, 319
 - wstrzymane, 316
 - wydajność, 323, 402
 - wykorzystywane ustawienia przed i po zakończeniu animacji, 316
 - z odbijającą się kulą, 319
 - z wykorzystaniem sprite'ów, 320
 - zanikanie, 322
- animation, 316
- animation-delay, 316
- animation-direction, 316
 - alternate, 319, 323
 - normal, 319
- animation-duration, 316
- animation-fill-mode, 316
 - both, 319
- animation-iteration-count, 316
- animation-iterations, 323
- animation-name, 315
- animation-play-state, 316
- animation-timing-function, 316, 319, 320
 - sprite'y, 344
 - step(), 320
- antywzorce z obszaru wydajności, 394
- Apache Cordova, 378
- aplikacje
 - hybrydowe, 378
 - narzędziowe, 354
 - rozrywkowe, 353
 - typu mash-up, 190
 - udostępnianie w wersji produkcyjnej, 28
 - w formie pakietów, 378
 - właściwości, 351
 - zwiększające produktywność, 352
- aplikacje działające w trybie offline, 165, 360
 - pamięć lokalna, 170
 - pamięć oparta na SQL-u i bazach danych, 179
 - pamięć podręczna, 166
 - pamięć sesji, 170
 - sprawdzenie połączenia do sieci, 165
- aplikacje mobilne
 - dostosowywanie stron do małych ekranów, 367
 - kategorie, 352
 - projektowanie, 349
 - testowanie, 379
 - tworzenie rozwiązań dla mobilnych przeglądarek z sinikiem WebKit, 360
 - zarządzanie pamięcią, 357
- aplikacje natywne, 12
 - aktualizacja, 12
 - dostępność, 13
 - falszywe, 378
 - historia, 15
 - proces zatwierdzania, 12
 - przeglądarka, 356
 - testowanie, 40
 - użytkownicy, 351
 - wady, 16
 - zalety, 16
- aplikacje sieciowe, 12
 - aktualizowanie, 13
 - debugowanie, 32
 - dostępność, 13
 - historia, 15
 - możliwości, 12
 - na urządzenia przenośne, 11
 - narzędzia do tworzenia, 29
 - natywne, 378
 - testy zautomatyzowane, 42
 - tryb pełnoekranowy, 66, 361
 - wady, 16
 - zalety, 16
 - zmiany, 12

- AppCache, 166
- Appcelerator Titanium, 379
- apple-mobile-web-app-capable, 66
- apple-mobile-web-app-status-bar-style, 66
- apple-touch-icon, 70
- apple-touch-startup-image, 70
- ARIA, 18, 51, 100, 191
 - aria-labeledby, 132
 - aria-required, 132
 - atrybuty, 192
 - dostępność, 191, 192
 - role, 192, 193
 - właściwości, 192
- arkusze klienta, 204
- arkusze stylów
 - kolejność klas, 208
 - osadzanie, 394
 - oznaczenie bezwzględnych odnośników, 217
 - przesyłanie, 68
 - tekst zastępczy, 101
 - text/css, 68
 - zewnętrzne, 198, 203
- aspect-ratio, 200
- async, 72
- atrybuty, 45
 - accesskey, 49
 - alt, 145
 - aria-, 45, 51
 - ARIA, 100, 132
 - aria-atomic, 51
 - aria-busy, 51
 - aria-describedby, 52
 - aria-disabled, 52
 - aria-expanded, 52
 - aria-haspopup, 52
 - aria-hidden, 52
 - aria-label, 143, 149
 - aria-live, 51
 - aria-relevant, 51
 - async, 72
 - autocomplete, 106
 - autofocus, 97, 107
 - autoplay, 155
 - bandwidth, 376
 - charset, 57, 63
 - class, 45, 46, 61, 188, 189, 208
 - content, 63, 66, 367
 - contenteditable, 50
 - contextmenu, 50
 - controls, 156
 - data-, 45
 - dataaddress, 219
 - data-position, 52
 - data-value, 52, 219, 231
 - datetime, 85
 - default, 159
 - defer, 72
 - dir, 45, 48, 61
 - disabled, 104, 105
 - download, 87
 - draggable, 51
 - dropzone, 51
 - for, 46, 134, 135
 - form, 105, 106, 111, 134
 - globalne, 45
 - height, 156
 - hidden, 50, 189
 - href, 68, 69
 - hreflang, 69
 - http-equiv, 63, 64
 - id, 45, 46, 61, 132, 212
 - internacjonalizacji, 45
 - itemid, 53
 - itemprop, 53, 188
 - itemref, 53, 188
 - itemscope, 53, 188, 190
 - itemtype, 53, 188
 - kind, 158
 - label, 94, 159
 - lang, 45, 47, 60, 61
 - language, 71
 - list, 132
 - logiczne, 45, 54, 57
 - loop, 156
 - manifest, 60, 166, 167
 - max, 100, 129
 - maxlength, 105, 129
 - media, 68, 70, 87, 199
 - metered, 376
 - mikrodanych, 45, 53
 - min, 100
 - multiple, 113
 - name, 63, 134
 - navigator.connection.type, 376
 - niestandardowe atrybuty danych, 45, 52
 - open, 92
 - pattern, 102, 104, 120, 128, 129
 - ping, 87
 - placeholder, 101, 102
 - poster, 156
 - preload, 156
 - preserveAspectRatio, 145
 - profile, 60
 - properties, 190
 - readonly, 104, 105
 - rel, 68, 69, 198

- atrybuty
 - required, 99, 129
 - reversed, 83
 - role, 51, 52, 192, 193
 - sandbox, 91
 - scoped, 70
 - seamless, 91
 - selektory atrybutów, 215
 - size, 105
 - sizes, 69
 - składnia, 54
 - spellcheck, 51
 - src, 71, 92, 155, 157, 158
 - srcdoc, 90
 - srclang, 159
 - step, 101
 - style, 45, 47, 143, 197
 - tabindex, 48, 220
 - target, 87
 - title, 45, 46, 94, 103, 129, 132
 - type, 69, 71, 83, 92, 98, 99, 128
 - umieszczenie w elemencie, 54
 - ustawianie wartości, 56
 - value, 83, 108
 - viewbox, 145
 - width, 156
 - z HTML-a 4, które stały się globalne w HTML5, 48
 - związane z modulem ARIA, 51, 52
- auto, 48
- autocomplete, 106
- autofocus, 107
- automatyczne wykrywanie numerów telefonu, 67
- autoplay, 155
- autouzupełnianie, 106

B

- background-color, 220, 281
 - padding-box, 332
- background-image, 142
- background-position, 143, 231, 251, 273, 274
 - animacja, 345
- background-repeat, 274, 278
 - no-repeat, 204, 275
 - repeat-x, 283
- background-size, 274
 - auto, 274, 276
 - contain, 274, 276
 - cover, 274
 - kilka grafik tła, 281
 - słowa kluczowe, 274
 - wada, 276
 - wartość DPI, 276

- backwards, 316
- Basic UI, 285
- bateria, 377
- bazy danych, 179
 - IndexedDB, 165, 179, 183
 - sieciowe bazy danych, 179
 - Web SQL, 179
- bezpieczeństwo, 191
 - origin, 191
- biblioteki
 - jQuery, 107, 383
 - kodu JavaScript i CSS dla iPhone'a, 15
 - Node.js, 35
 - o otwartym dostępie do kodu źródłowego, 26
- Bing, 394
- BlackBerry 10, 37, 41
 - debuger, 37
 - WebGL, 153
- BlackBerry Browser, 37
- black-translucent, 361
- Blink, 29
- blok
 - deklaracji stylu, 197
 - nawigacyjny dokumentu, 78
- blokowanie
 - kliknięcia, 374
 - przesuwania strony, 373
 - przewijania, 374
- border-color, 256
 - currentColor, 256
- border-image, 328
 - border-width, 257
 - składnia, 329
 - właściwość skrócona, 333
- border-image-outset, 332
- border-image-repeat, 330, 332
 - repeat, 332
 - round, 333
 - space, 333
 - stretch, 332
- border-image-slice, 330, 331
 - fill, 330
- border-image-source, 330
- border-image-width, 331
- border-radius, 261
 - bottomleft, 262
 - bottomright, 262
 - efekt pikselizacji, 261
 - grafika tła, 264
 - kolejność narożników, 262
 - promień, 261
 - wartości w modelu DOM, 262
 - składnia, 261

- skrót dla właściwości, 261
- topleft, 262
- topright, 262
- border-style, 255, 256, 329
 - dotted, 255
 - hidden, 256
 - none, 256
 - wartości właściwości, 256
- border-style-right: dashed, 255
- border-width, 257, 331
 - model pudełkowy, 332
 - słowa kluczowe, 257
- both, 316
- box-shadow, 286
 - inset, 286
 - kolejność wyświetlania, 287
- box-sizing, 259
 - border-box, 259
- breakpoint, 325
- Browser Testing and Tools, 32
- browsing context, 91

C

- CACHE, 168
- Cache-Control, 388
- canvas, 18
- captions, 159
- capture, 113
- Cascading Style Sheets, 196
- CasperJS, 42
- cele użytkowników, 21
- change, 377
- changedTouches, 371
- chapters, 159
- charset, 57, 63
- Chrome, 14
 - okno robocze, 31
 - panel Overrides, 31
 - śledzenie zużycia pamięci, 357
 - zarządzanie pamięcią, 399
 - zdalne debugowanie, 32
 - zmiana współrzędnych dla geolokalizacji, 31
- Chrome Canary, 28
- ciało dokumentu, 57
- cienie, 282
 - dla tekstu, 283, 284
 - format RGBA, 239
 - pól, 286
- class, 45, 46, 61, 208
- Client-Hints, 346
- Clown Car, 91, 144
- column, 326
- column-count, 326

- column-fill, 328
- column-gap, 326, 328
- column-rule, 326, 327
- column-rule-width, 328
- column-span, 328
- column-width, 326, 328
- compassneeds calibration, 375
- console.log(), 33
- content, 63, 66
- contenteditable, 50
- context.arc, 151
- contextmenu, 50
- controls, 156
- cookie, 170, 171, 172, 204
 - identyfikator, 172
 - ograniczenie, 171
- CORS, 191
- Crossfire, 32
- cross-origin resource sharing, 191
- CSS, 45, 196
 - circle, 109
 - czas wyświetlania, 398
 - definicje, 196
 - disc, 109
 - elementy pasujące do selektorów pseudoklas, 221
 - formaty deklarowania kolorów, 236
 - gradienty, 264
 - kaskadowość, 233
 - maski, 346
 - none, 109
 - nowe elementy, 19
 - obsługa plików w formacie SVG, 142
 - odstęp i znaki przestankowe, 213
 - poprawianie interfejsu użytkownika, 130
 - reguły, 196
 - selektory, 206
 - składnia, 196
 - specyficzność, 233
 - square, 109
 - w projektowaniu RWD, 325
 - wskazywanie elementów, 197
 - wydajność, 397
 - wygląd i styl natywnych aplikacji z iPhone'a, 253, 260
 - zapisywanie plików, 198
 - zwiększanie użyteczności stron, 131
- CSS 2, 195
 - ograniczenia, 20
- CSS Object Model, 201
- CSS3, 19, 97, 195
 - animacje, 293, 315, 323
 - cienie, 282
 - dodatkowe możliwości, 235

- CSS3
 - funkcje, 19
 - grafika tła, 280
 - moduły, 195, 253
 - interfejsu użytkownika, 222
 - nauka, 260
 - nowe selektory, 210
 - ograniczenie liczby węzłów modelu DOM, 253
 - przejścia, 293, 294, 323
 - przygotowania do nauki, 25
 - selektory, 405
 - wartości kolorów, 235
 - wskazywanie węzła dokumentu, 207
 - wydajność, 253
 - zapytania media, 200
 - CSS4
 - selektory, 410
 - CSSOM, 201
 - CubeeDoo, 26, 246
 - alterAValue(), 175
 - alterValue, 173
 - ASC, 181
 - cardinfo, 176
 - currentLevel, 176
 - debugger, 178
 - defaultvalues, 174
 - DESC, 181
 - gameDuration, 177
 - gbdoo, 174
 - gbdoo.storageType, 180
 - getItem(), 174, 175, 177
 - getLocation(), 186
 - insert, 181
 - nth-of-type(), 226
 - pauseGame, 173, 176
 - playGame, 173, 174, 176, 177
 - px, 246
 - rem, 246
 - renderHighScores(), 182
 - reset, 173
 - select, 181
 - sessionStorage, 174
 - setupGame, 177
 - storageType, 181
 - storeValues(), 173, 174, 175, 176
 - timeLeft, 177
 - touchcancel, 371
 - vh, 246
 - vm, 246
 - Web SQL, 180
 - currentColor, 240, 256
 - dla tekstu, 284
 - czas, 247, 249
 - czas pracy na baterii, 381
 - akceleracja sprzętowa, 385
 - format JPEG, 382
 - ilość kodu w JavaScriptcie, 383
 - kolory, 382
 - liczba przesyłanych żądań, 384
 - obliczanie układu elementów, 386
 - ponowne wyświetlanie, 386
 - sieć, 384
 - czas wczytywania, 400
 - dla zdarzeń, 401
 - porównywanie, 401
 - czcionki
 - bezszerzyfowe, 19
 - dostosowane do sieci, 19
 - z ikonami, 345, 390
 - zamiast ikon graficznych, 390
 - częstotliwość, 247, 249
 - czytelność tekstu, 284
 - czytniki ekranu
 - aplikacje RIA, 51
 - atrybut lang, 47
- ## D
- dane, bez połączenia z siecią, 18
 - dataaddress, 219
 - data-position, 53
 - data-value, 53, 219, 231
 - datetime, 85
 - debugery, 30
 - debugowanie zdalne, 32
 - Adobe Edge Inspect, 35
 - BlackBerry 10, 37
 - Ghoshlab, 35
 - kodu w JavaScriptcie za pomocą Aardwolfa, 37
 - narzędzia diagnostyczne na Androida, 32
 - weinre, 34
 - za pomocą portu USB, 32
 - default, 159
 - defer, 72
 - definicja produktu, 359
 - definiowanie konkretnego czasu lub daty, 85
 - deklaracja typu dokumentu, 57, 58
 - dla plików SVG, 143
 - description, 63
 - descriptions, 159
 - device-aspect-ratio, 200
 - device-height, 65, 200
 - devicemotion, 375
 - deviceorientation, 375, 376
 - device-width, 65, 200
 - dh, 346
 - diagram wodospadu, 391, 393

- dir, 45, 48, 61
- disabled, 104, 105
- display
 - flex, 335, 338
 - inline-flex, 335
 - przedrostki, 336
- doctype, 58
- Document Object Model, 29
- document type declaration, 57
- dodawanie metadanych, 63
- dokument HTML5, 57
 - <body>, 61
 - <head>, 60
 - <html>, 59
 - <title>, 60
 - deklaracja DTD, 58
 - dodawanie stylów, 70
 - nagłówki dokumentu, 62
- DOM, 97, 186
 - classList, 176
 - minimalizowanie modelu, 399, 401
 - node.classList.add(), 176
 - node.classList.contains(), 176
 - node.classList.remove(), 176
 - node.classList.toggle(), 176
 - rysunki SVG, 152
- DOMContentLoaded, 400
- domyślna kolejność znaczników, 49
- dostęp do sprzętu, 375
- dostosowywanie stron do małych ekranów, 367
- dots per inch, 341
- dotyk, 367, 368
 - fikcyjne i rzeczywiste zdarzenia związane z kliknięciem, 372
 - reagujące obszary, 369
 - wielodotyk, 368
 - zdarzenia związane z myszą i dotykiem, 369
- download, 87
- dpi, 246
- DPI, 341
- dppx, 246
- dpr, 346
- dpx, 246
- draggable, 51
- Dragonfly, 30, 32
- Dreamweaver, 28
- dropzone, 51
- DTD, 57, 58
- dw, 346
- dymek z treścią, 256
- dynamiczne dostosowywanie do wymiarów ekranu, 340
- dynamiczne klawiatury, 117

- dynamicznie dopasowujące się witryny, 325
 - kolumny, 328
 - model flexbox, zapytania media, 338
 - układy wielokolumnowe, 334
- dźwięki, 18
 - dodawanie, 161
 - informacyjne, 160

E

- E ~ F, 214
- E + F, 214
- E > F, 213
- E F, 212
- Eclipse, 28
- edytor tekstu, 28
- efekty
 - dotknięcia, 372
 - typograficzne, 79
- ekran
 - AMOLED, 382
 - dostosowywanie stron do małych ekranów, 367
 - dotknięcia w celu przewijania, 375
 - obszary reagujące na dotyk, 369
 - przechwytywanie dotknięcia, 373
 - szerokość, 69
 - wielodotyk, 368
 - wysokość, 69
 - zdarzenie kliknięcia, 369
- elastyczny kontener, 335, 336
 - kierunek układu, 336
 - wielkość elementów, 337
- elementy, 44
 - blokowe, 75
 - canvas, 12, 18
 - definiowanie sekcji, 47
 - dodawanie do strony internetowej, 54
 - dopasowanie semantyczne, 57
 - formularzy, atrybuty, 99
 - grupujące, 81
 - identyfikatory, 46, 211
 - interaktywne, 92
 - internacjonalizacja, 47
 - liniowe, 75
 - zmiana wyglądu, 45
 - multimedialne, 90, 159
 - nieistotne, 50
 - odpowiedzialne za warstwę prezentacji, 196
 - określanie semantycznej struktury dla elementów o zmienionym przeznaczeniu, 51
 - osadzone, 90
 - pływające, 285
 - podrzędne, selektory CSS, 208

- elementy
 - powiązanie ze znacznikami, 50
 - przeciąganie, 51
 - przynależność do klas, 46
 - puste, 55
 - samodzielny selektor uniwersalny, 210
 - samozamykające, 55
 - sekcyjne, 75
 - struktura, 76
 - zasięg nagłówków i stopek, 76
 - selektory, relacyjne, 212
 - składnia, 54
 - składniki, 44
 - sprawdzanie pisowni i gramatyki, 51
 - stosowanie, 45
 - styl tekstu, 47
 - tekstowe elementy semantyczne, 83
 - we wcześniejszych wersjach HTML-a, 45
 - wewnątrztrójkątowe określenie stylów dla jednego wystąpienia, 47
 - wskazywanie, 224
 - przy użyciu atrybutów, 217
 - wykorzystanie niezgodnie z przeznaczeniem, 52
 - zagnieżdżanie, 55, 56
 - zamykanie, 54
 - ze znacznika <head>, 62
 - zmodyfikowane tekstowe semantyczne elementy, 86
 - związane z warstwą prezentacji, 88
- emulator, 38, 403
 - Android Debug Monitor, 34
 - Androida, 38
 - Androida 4.2.2 działający w systemie OS X, 33
 - Firefox OS Simulator, 39
 - Opera Mobile Emulator, 39
 - prrowadzenie testów, 38
 - urządzeń mobilnych, 379
 - Windows Phone Emulator, 39
 - WVGA 512 MB, 39
- Error Console, 30
- etykiety, 111
- Expires, 388

F

- F12, 30
- FALLBACK, 168
- filmy, 18, 154
- Firefox, 30
- Firefox, 14
 - zdalne debugowanie, 32
- Firefox OS
 - emulator, 39
 - Simulator, 39

- Flash, 191, 293
 - animacje, 293
 - Flash Player 11.1, 293
- flex, 335, 337
- flex-basis, 337
- flexbox, 334
- flex-direction, 336
 - column, 338
 - row-reverse, 336
- flex-flow, 336
- flex-grow, 337
- flex-shrink, 337
- flex-wrap, 336
- for, 46, 101
- form, 105
- format-detection, 67
- formaty
 - AAC, 154
 - H.264, 154, 155
 - JPEG, 382
 - MPEG4, 154
 - PNG8, 392
 - SVG, 141
 - VP8, 154
 - WebM, 154
 - wektorowy, 142
- formaty kolorów
 - cmyk(), 236, 240
 - hsl(), 236, 239
 - hsla(), 236, 239
 - nazwane kolory, 235, 236, 240
 - RGB, 236
 - rgb(), 235, 236
 - rgba(), 236
 - RRGGBB, 236
 - wartości szesnastkowe, 235, 236
- formularze, 17
- formularze internetowe, 97
 - adres e-mail, 105, 116
 - adres URL, 105, 117
 - autouzupełnianie, 106
 - bieżąca wartość, 135
 - blok do wprowadzania tekstu, 138
 - dane pofne, 106
 - dostępne listy wartości, 132
 - elementy związane z datą i godziną, 123, 125
 - generator par kluczy, 137
 - komunikaty o błędach, 127
 - kontrolki
 - automatycznie aktywowane, 107
 - etykiety, 139
 - grupowanie, 138
 - liczba opcji i znaków, 105

- nieedytowalne, 104
- opcje, 133
- styl, 131
- liczby, 119
- menu z opcjami do wyboru, 138
- nowe elementy, 132
- paleta wyboru kolorów, 122
- pliki, 112
- pola
 - <input>, 108
 - daty, 125
 - do wprowadzania danych, 108
 - godziny, 125
 - ukryte, 114
 - wyboru, 109
 - wymagane, 99
 - wyszukiwania, 122
- poprawianie interfejsu użytkownika, 130
- postęp, 136
- powiązanie elementów, 105
- przedziały, 121
- przyciski, 110, 111, 112, 114
- rysunki, 114
- sprawdzanie poprawności, 127
- style elementów, 114
- suwak, 101, 121
- tekst zastępczy, 101
- telefon, 118
- wskazówki lub instrukcje dotyczące
 - oczekiwanych typów danych, 101
- wyłączenie elementu formularza, 104
- wyrażenia regularne, 102
- wyświetlanie siły hasła, 135
- zakres dozwolonych wartości, 100
- żądanie hasła, 109

forwards, 316

funkcje

- cubic-bezier, 297, 298
- ease-in, 297, 319
- ease-out, 297, 319
- matrix(), 309
- perspective(), 311
- repeating-linear-gradient, 278
- rotate(), 306
- rotate3d(), 311
- rotateX(), 306
- rotateY(), 307
- scale(), 305
- scale3d(), 310
- scaleX(), 305
- scaleY(), 306
- scaleZ(), 311
- skew(), 307

- skewX(), 307
- skewY(), 308
- transformacji trójwymiarowych, 309
- translate(), 304
- translate3d(), 310
- translateX(), 305
- translateY(), 305
- translateZ(), 310
- z rodziny step, 299

G

generowanie treści, 219, 230

- content, 230
- tworzenie motywów, 230
- wykorzystanie, 231
- wyświetlanie pływających elementów stron, 230
- wzbogacanie odnośników, 219, 231

geolokalizacja, 18, 183

- accuracy, 185
- altitude, 185
- altitudeAccuracy, 185
- geotagowanie, 183
- heading, 185
- latitude, 185
- longitude, 185
- speed, 185
- watchCurrentPosition(), 184

gesty, 369

- przewodnik po gestach dotykowych, 371

getLocation(), 186

gęstość pikseli, 341

Ghostlab, 35, 36

główna treść strony, 82

główny element HTML, 57

GPD, 249

gradienty, 264

- kolory, 269
- kołowe, 264, 265, 398
- narzędzia do tworzenia, 281
- określanie, 264
- przezroczystość, 271, 273
- przycisków, 273
- stretch, 333
- szerokość, 279
- tła, 275
- tworzone za pomocą pikseli, 278
- wielokrotne wykorzystanie, 273
- wydajność, 398
- z paskami, 277

gradienty liniowe, 264

- 315deg, 268
- kąt ścieżki, 265
- kąty, 266

- gradienty liniowe
 - kierunek zmiany koloru, 266
 - powtarzanie, 278
 - punkt początkowy, 265
 - składnia, 265, 268
 - ścieżka gradientu, 267, 272
 - w iPhone, 271

- grady, 248

- grafika

- format JPEG, 382
 - kompresja, 389
 - nagłówka, 340
 - obramowania, 328
 - obsługa, 153
 - optymalizowanie, 396
 - pierwszego planu
 - dynamicznie dopasowana, 144
 - zapisywanie rysunku, 390
 - przetwarzana z akceleracją sprzętową, 385
 - rysowanie, 148
 - rysunki w formacie SVG, 152
 - SVG, dołączanie do dokumentów, 143
 - udostępnianie, 340
 - uniemożliwienie zaznaczenia, 230
 - wydajność w przeglądarkach, 153
 - zapisywana wewnątrzwerszowo, 389

- grafika tła, 91, 144

- formaty danych, 330
 - kolor tła, 271
 - niepowtarzana, 275
 - powtarzana, 278
 - przypisanie kilku grafik do jednego węzła modelu DOM, 280
 - skalowanie do mniejszego rozmiaru, 274
 - ścieżka gradientu, 268
 - wartości bezwzględne i względne, 274
 - wielkość, 274, 281
 - zniekształcenia, 274

- grupowanie

- semantyczne, 17
 - treści, 81

H

- H.264, 154, 155
- Handbrake, 163
- height, 156, 200, 255
 - auto, 340
- Helvetica, 19
- hidden, 50, 189
- hover, 372
- href, 68, 69
- hreflang, 69
 - fr, 217

- HSL, 235, 239

- jasność, 239
 - składnia, 239
 - wartość odcieni, 239

- HSLA, 19, 235

- kanał alfa, 239

- HTML 4, deklaracje DTD, 59

- HTML5, 11, 17

- atrybuty, 45
 - globalne z HTML 4, 48
 - globalne związane z dostępnością i elementami interaktywnymi, 50
 - niestandardowe, 52
 - deklaracja DTD, 58
 - elementy, 44
 - koncentracja na szczegółach, 44
 - niedostępne elementy XHTML-a, 95
 - niezmodyfikowane elementy, 89
 - nowe elementy, 75
 - przechodzenie na HTML5, 43
 - składnia, 43, 56
 - specyfikacja, 44
 - sprawdzanie danych, 98
 - stan, 165
 - wartości charakterystyczne dla producentów urządzeń przenośnych, 66
 - wymagane komponenty, 57
 - zmiany w elementach tekstowych w porównaniu z HTML-em 4, 88
 - zmodyfikowane elementy i atrybuty, 43

- HTML5 Canvas, 148

- http-equiv, 63, 64

- hue, 235

I

- IcoMoon, 390

- id, 45, 46, 61, 212

- IDE, 28

- identyfikatory, 46, 205, 209

- identyfikatory URI

- danych, 343
 - dodawanie w CSS-ie, 389

- ikony

- na stronie głównej urządzenia, 364
 - serwisów społecznościowych, 401
 - z czcionek, 345
 - z tańczącą postacią, 344

- ImageAlpha, 392, 396

- ImageOptim, 396

- image-set(), 345

- importScripts(), 187

- indeterminate, 109

- informacje zwrotne, 401

- initial-scale, 65
- instrukcje
 - @media, 201
 - @supports, 201
 - console.log(), 33
 - margin: 0, 204
 - padding: 0, 204
 - removeAttribute(), 106
 - setAttribute(), 106
 - weinre, 35
- integracja z treściami wyświetlanymi za pomocą niezależnych wtyczek, 92
- integrated development environment, 28
- interfejs API, 16, 165
 - animacji, 324
 - ARIA, 192
 - Battery Status, 377
 - dataset, 173
 - niestandardowe atrybuty danych, 53
 - Debug API, 32
 - dla mobilnych aplikacji sieciowych, 377
 - dla pamięci podręcznej i pamięci sesji, 165
 - do geolokalizacji, 18
 - do obsługi pamięci, 18
 - do wymiany komunikatów między dokumentami, 191
 - DOM dla mikrodanych, 190
 - JavaScript, 97
 - mikrodanych, 18, 190
 - multimedialny, 141
 - Network, 376
 - nowe standardy, 16
 - przygotowania do nauki, 25
 - systemu geolokalizacji, 184
 - validity, 129
- interfejs użytkownika
 - aplikacje rozrywkowe, 353
 - Metro, 41
 - Sencha Touch, 379
 - szybkość reagowania, 401
- Internet Explorer 10, 14
- iOS, 40
 - aktywowanie pola tekstowego, 107
 - aplikacja działająca w trybie offline, 360
 - autokorekta tekstu, 51
 - pasek stanu, 361
 - tło aplikacji dla systemu iOS, 277
 - url, 117
- iOS Simulator, 38
- iPhone, 15
- iPhone Bingo, 15
- iPhone SDK, 13, 15
- isTouchEnabled, 372
- itemid, 53

- itemprop, 53, 188
- itemref, 53, 188
- itemscope, 53, 188
- itemtype, 53, 188
- iUI, 15

J

- Jasmine, 42
- jasność, 239
- JavaScript
 - animacje, 293
 - dołączanie dźwięków, 161
 - kontrolowanie elementów <audio> i <video>, 159
 - kreska, 262
 - minimalizowanie kodu, 389
 - obsługa w przeglądarkach, 72
 - pobieranie osadzonych skryptów, 203
 - przygotowania do nauki, 25
 - rezygnacja z platform, 393
 - rysowanie, 152
 - sprawdzanie
 - danych, 98
 - typu połączenia, 342
 - wspólny plik z kodem, 388
 - wydajność, 71
 - zużycie pamięci i energii, 383
- jawne etykiety, 46
- jednostki miar, 244
 - %, 245
 - ch, 245
 - cm, 245
 - czas, 247, 249
 - częstotliwość, 247, 249
 - em, 245
 - ex, 245
 - grady, 248
 - in, 245
 - kąty, 247
 - kąty pełne, 249
 - mm, 245
 - pc, 245
 - piksele, 245
 - pt, 245
 - px, 245
 - radiany, 248
 - rem, 245
 - stopnie, 247
 - vh, 245
 - vmax, 245
 - vmin, 245
 - vw, 245
 - wartości określające długość, 244

- jednostki rozdzielczości, 246
 - dpc, 246
 - dpi, 246
 - dppx, 246
- JSON, 172, 174, 175

K

- kaskadowość, 233
- kąty, 247, 248
- kierunek tekstu, 48
 - zmiana kierunku wybranego fragmentu, 86
- kind, 158
- Kindle Fire, 42
- klasy
 - .active, 221
 - .clearfix, 230
 - current, 323
 - enlargen, 308
 - flipped, 322
 - hover, 222
 - matched, 322
 - no-js, 60
- klatka kluczowa, 294
- klawiatura numeryczna, wyświetlenie, 119
- kliknięcia, 368
- kod
 - dostępu, 36
 - składnia XHTML, 59
 - testowanie, 40
 - w HTML5, 56
- kodeki wideo, 154
- kodowanie UTF-8, 63
- kolory
 - activeBorder, 241
 - activeCaption, 241
 - appWorkspace, 241
 - background, 241
 - buttonFace, 241
 - buttonHighlight, 241
 - buttonShadow, 241
 - buttonText, 241
 - captionText, 241
 - grayText, 241
 - highlight, 241
 - highlightText, 241
 - inactiveBorder, 241
 - inactiveCaption, 241
 - inactiveCaptionText, 242
 - infoBackground, 242
 - infoText, 242
 - match, 242
 - menu, 242
 - menuText, 242

- nazwane, 240
- scrollbar, 242
- specyficzne, 242
- tła, 271
- threeDDarkShadow, 242
- threeDFace, 242
- threeDHighlight, 242
- threeDLightShadow, 242
- threeDShadow, 242
- w przeglądarkach, 241
- wartości, 235
 - szesnastkowe, 236
- windowFrame, 242
- windowText, 242
- wybór składni, 244
- kolumny, 326
 - dokładna szerokość, 328
 - elementy biegnące przez wszystkie kolumny, 328
 - linie, 327
 - marginesy, 327
 - w dynamicznie dopasowującym się układzie, 328
 - zapełnianie w całości, 328
- kombinatory, 210
 - dziecka, 213
 - ogólny kombinator brata, 214
 - potomka, 212
 - przyległego brata, 214
 - specyficzność, 233
- komentarze ruby, 85
 - nawiasy wokół tekstowego komponentu, 86
- komponenty sieciowe, 232
- kompresowanie plików binarnych, 392
- kontekst przeglądania, 91
- kontrolki, 159
 - wygląd, 163
 - formularza menu, 94
- kotwice
 - umieszczenie na stronie, 87
 - wewnętrzne, 46
 - wskazywanie elementów, 46
- krzywa Béziera, 298
- kształty, 143
 - definiowanie wyglądu i stylu, 150
 - i linie, 143
 - koło, 151
 - prostokąt, 150

L

- label, 94, 159
- lang, 45, 47, 60, 61
- language, 71

- legenda, 83
 - dla treści nadrzędnego elementu, 93
- leniwe wczytywanie, 400
- li strong {}, 212
- light, 235
- linewidth, 150
- list, 132
- listy
 - poleceń lub kontrolek, 94
 - pseudoklas dla interfejsu użytkownika, 221
 - selektorów typu, 207
- Load, 400
- localStorage, 203
- loop, 156
- ltr, 48

M

- magnetometr, 375
- mailto:, 87
- manifest, 60
- margin, 263
- marginesy, 258, 259
 - akapitów w kolumnach, 327
 - dwóch przyległych elementów, 259
 - szerokość, 258
 - wewnętrzne, 162, 258
- maskowanie, 346
- max, 100
- maximum-scale, 65
- maxlength, 105
- mechanizmy sprawdzania poprawności, 104, 128
- media, 68, 70, 87, 199
 - all, 199
 - braille, 199
 - dopuszczalne wartości, 69
 - embossed, 199
 - handheld, 199
 - print, 69, 199
 - projection, 199
 - screen, 69, 199
 - speech, 199
 - tty, 199
 - tv, 199
- MediaStream Recording, 377
- menu, 94
 - podręczne, pozycja, 94
- metadane
 - dodawanie, 63
 - strony, 72
- metadata, 159
- metody
 - addEventListener(), 368, 384
 - applicationCache, 168
 - applicationCache.update(), 169
 - beginPath(), 151
 - canPlayType(), 157
 - clear(), 171
 - clearInterval(), 187
 - clearTimeout(), 187
 - close(), 187
 - closePath(), 151
 - document.getItems(itemType), 190
 - executeSQL(), 180
 - fill(), 150, 152
 - fillRect, 151
 - fillStyle, 151, 152
 - GET, 109
 - getContext(contextId), 150
 - getCurrentPosition(), 184, 186
 - getImageData(), 151
 - getItem(), 171, 174, 175
 - getUserMedia(), 378
 - JSON.stringify(), 178
 - JSON.parse(), 175
 - key(), 171
 - obsługi onclick, 374
 - obsługi zdarzeń, 61
 - <body>, 73
 - urządzenia dotykowe, 221
 - onchange, 134
 - onformchange, 134
 - onforminput, 134
 - onload="wykonajZadanie();", 61
 - openDatabase(), 179
 - POST, 109
 - postMessage(), 186, 187
 - preventDefault(), 111, 371, 373, 374
 - querySelector(), 384
 - querySelectorAll(), 384
 - setCustomValidity(), 119, 130
 - setFocus(), 97
 - setInterval(), 187
 - setItem(), 171, 174, 175
 - setTimeout(), 187
 - stepDown(), 120
 - stepUp(), 120
 - stopPropagation, 374
 - stringify(), 174
 - strokeRect(), 150
 - terminate(), 187
 - transaction(), 180
 - updateready, 169
 - watchCurrentPosition(), 184
 - window.matchMedia, 201
- Metro, 41
- miejsce podziału wiersza, 86

- mikrodane, 18, 188, 190
 - a mikroformaty, 188
 - atrybuty, 188
 - interfejs API, 190
 - słowniki, 188
 - mikroformaty, 188
 - class, 188, 189
 - hCard, 189
 - MIME text/cache-manifest, 167
 - min, 100
 - minify, 389
 - minimum-scale, 65
 - mobilne systemy operacyjne, 39
 - mobiReady, 39
 - model flexbox, 334
 - elastyczny kontener, 335
 - elementy z bezwzględnie określoną pozycją, 336
 - kolejność wyświetlania elementów, 336
 - rozmieszczenie elementów, 335, 336
 - tworzenie kolumn, 335
 - wielkość elementów, 337
 - właściwości, 335
 - wykrywanie funkcji, 339
 - zapytania media, 338
 - zmiana układu bez modyfikowania kodu
 - w HTML, 335
 - model pudełkowy, 254
 - cienie, 286
 - komponenty, 257
 - marginesy, 258, 259
 - obramowanie, 255, 258
 - rozmiar zajmowany przez element, 259
 - szerokość, 258
 - treść, 258
 - właściwości, 254
 - wysokość, 258
 - z CCS-a, 257
 - model Shadow DOM, 113, 130, 232
 - Modernizr, 60, 150
 - Modify Headers, 40
 - moduły
 - specyfikacji CSS3, 195
 - zarządzania dostępnością, 51
 - MPEG4/H.264, 154
 - multimedia
 - dostępność dla użytkowników z wadami
 - słuchu, 158
 - dynamiczna zmiana, 157
 - dynamiczne dopasowywanie wielkości
 - obrazu, 162
 - dźwięki, 160, 161
 - etykiety, 157
 - filmy, 158, 162
 - kontrolowanie wyglądu i działania, 155
 - napisy, 159
 - osadzanie na stronach, 154
 - pliki, 154
 - wideo, 158
 - ścieżki, 158, 159
 - udostępnianie plików, 201
 - w układzie płynnym, 340
 - wydajność, 163
 - zasoby multimedialne, 157
 - multimedialne interfejsy API, 141
- ## N
- nagłówkek
 - Cache-Control, 388
 - Client-Hints, 346
 - dokumentu, 57
 - Expires, 388
 - odpowiedzi HTTP, 64
 - sekcji, 79
 - w układzie płynnym, 340
 - zasięg, 76
 - najlepsze praktyki, 56
 - name, 63
 - narożniki, zaokrąglone, 261
 - narzędzia, 352
 - diagnostyczne na Androida, 32
 - do testowania, 37
 - dostępne w internecie, 39
 - emulatory, 38
 - internetowe, 42
 - pracownie z przeglądarkami, 40
 - symulatory, 38
 - telefony, 40
 - testy zautomatyzowane, 42
 - do tworzenia gradientów, 281
 - npm, 35
 - programistyczne, 28
 - debugery stacjonarne, 30
 - do debugowania, 29
 - edytor tekstu, 28
 - przeglądarka, 28
 - zdalne debugowanie, 32
 - nasylenie, 239
 - natywne aplikacje sieciowe, 378
 - navigator.connection.bandwidth, 376
 - navigator.connection.metered, 376
 - navigator.connection.type, 342, 376
 - NETWORK, 168
 - New Exciting Web Technologies, 16
 - NEWT, 16
 - niestandardowe atrybuty danych, 45
 - z przedrostkiem data-, 52

- node package manager, 35
- no-js, 60
- none, 316
- notacja Wielbłądzia, 262
- notacja z podwójnym dwukropkiem, 231
- number, 100

O

obiekty

- classList, 176
- context, 150
- coords, 185
- documentFragment, 387
- gbdo, 181
- geolocation, 186
- localStorage, 172, 203
- MediaQueryList, 201
- navigator
 - onLine, 165
- navigator.battery, 377
- plótna, 148
- sessionStorage, 172, 174
- Touch, 371
- TouchEvent, 371
- validity, 129
- window, 179

obliczanie układu elementów, 386

obramowanie, 255, 258

- dane reprezentujące rysunek, 330
- definiowanie kolorów, 256
- dolne dla ostatniej pozycji listy, 263
- grafika, 328
- narzędzia, 334
- podział, 330
- powtarzanie i skalowanie fragmentów krawędzi, 332
- szerokość, 255, 257, 331
- tworzenie trójkątów, 256
- ustawianie, 329
- widoczne, 256
- wykraczanie poza obszar elementu, 332
- wysokość, 255
- zapis skrócony, 255

obsługa płótna, 12

obszary reagujące na dotyk, 369

odcień, 239

odnośniki, 79, 86

- dodawanie grafiki tła, 218
- obsługa w urządzeniach przenośnych, 87
- obsługiwane w sposób specjalny, 88
- oznaczenie za pomocą ikon, 217
- podkreślenie, 283
- pseudoklasy, 220

- selektor atrybutu kodu języka, 216
- tekst informujący o typie odnośnika, 217
- ustalenie typu, 219

ogólna treść dokumentu, 82

ogólny kombinator brata, 214

okno dialogowe

- do zapisywania i kopiowania grafiki, 373
- z operacją kopiowania i definiowania, 372

okno robocze, 64

- dostosowywanie stron do małych ekranów, 367
- obsługa gestów, 369
- skalowanie, 65
- szerokość, 65, 69, 202
- w przeglądarkach mobilnych, 30
- współczynnik przybliżenia, 65
- wydajność, 398
- wysokość, 65, 69
- zapytania media, 201

onclick, 374

onload, 97

onreset, 112

onsubmit, 111

onTouchEvent, 373

open, 92

Opera, 14

Opera Mini, 17, 41

Opera Mini Simulator, 39

Opera Mobile, 17

Opera Mobile Emulator, 39

opóźnienie, 387

- animacje, 402

- dwukrotne dotknięcie, 402

- pamięć, 395

- przy pobieraniu witryn mobilnych, 395

- sprite'y, 389

optymalizowanie grafiki, 396

- korzyści ze stosowania stylów CSS, 397

- minimalizowanie modelu DOM, 399

- okno robocze, 398

- procesor graficzny, 398

- wykorzystanie pamięci, 396

- zarządzanie pamięcią, 399

order, 336

orientation, 200

overflow, 284

P

- p:first-of-type + p {}, 214

- p > strong {}, 213

- p strong {}, 212

- PageSpeed, 396

- PageSpeed Google, 403

- pakiet SDK, 32
 - dostęp do narzędzia ADB, 32
 - pobieranie, 32
 - symulatory i emulatory, 38
 - udostępnienie, 15
 - Windows Phone SDK, 39
- pakowanie plików, 392
- paleta wyboru kolorów, 122
- pamięć
 - bez połączenia z siecią, 18
 - grafika, 396
 - oparta na SQL-u i bazach danych, 179
 - urządzeń przenośnych, 356
 - wydajność, 395
 - zarządzanie, 357, 395, 399
 - zmiany zużycia, 401
- pamięć lokalna, 170
 - baza IndexedDB, 170
 - baza Web SQL, 170
 - metody i właściwości, 171
 - zwiększanie wydajności aplikacji, 172
- pamięć podręczna, 171, 172
 - aktualizowanie, 168, 169
 - aplikacji, 166
 - plik manifestu, 167, 168
- pamięć sesji, 170
 - dane, 171
 - metody i właściwości, 171
- pasek dolny, 363
- pasek nawigacji, 361
 - deklarowanie pełnych kolorów, 272
 - gradienty, 271
 - kontrolki do zarządzania zawartością widoku, 362
 - przezroczystość, 273
 - punkt zmiany koloru, 272
 - ukrywanie, 362
 - user experience, 362
 - wielkość i kolor, 363
- pasek stanu
 - black-translucent, 361
 - projektowanie aplikacji, 360
 - wygląd, 66
- pattern, 102, 104
 - title, 103
 - wyrażenie regularne, 102
- pauseGame(), 175
- Phantom Limb, 372
- PhantomJS, 42
- PhoneGap, 34, 378
- Photoshop, HSB, 235
- Pickerview, 15, 26
- pierwsze żądanie, 172
- piksele, 245, 341
- pikselizacja, 153
- ping, 87
- placeholder, 101, 102, 108, 109
- platforma mobilna
 - ilość pamięci, 356
 - jedno okno i jedna aplikacja naraz, 358
 - mały ekran, 356
 - minimalna dokumentacja, 359
 - możliwości, 356
 - rozważania związane z projektowaniem, 359
- playGame(), 175
- pliki
 - .appcache, 167, 168, 169
 - .htaccess, 167
 - .mp4, 154
 - .ogv, 154
 - .webm, 154
 - audio, 158
 - binarne, kompresowanie, 392
 - cookie, 170, 204, 394
 - CSS, 203
 - do resetowania lub normalizowania stylów CSS, 204, 276
 - manifestu pamięci podręcznej, 167
 - multimedialne, 154
 - Normalize.css, 204
 - pakowanie, 392
 - PNG, 346
 - przezroczyste pliki JPEG, 346
 - wideo, 158
 - ze ścieżkami, 159
- plótno, 149
- płynny układ, 325
- tworzenie, 340
- PNGCrush, 392
- pobieranie niezależnych skryptów, 394
- podpis, 83
 - dla treści nadrzędnego elementu, 93
 - filmu, 158
- podział
 - tematyczny na poziomie akapitu, 83
 - wiersza, 86
- Pointer Lock, 377
- pointerenter, 221
- pointerleave, 221
- pojedynczy punkt krytyczny, 394
- pola
 - do wprowadzania danych, 108
 - email, 102
 - menu podręczne, 138
 - number, 100
 - password, 102

- range, 100
- search, 102
- tekstowe, 108
- telephone, 102
- text, 102
- ukryte, 114
- url, 102
- wyboru, 109
- połączenie sieciowe, 376
- ponowne wyświetlanie, 386
- porządek GPDŁ, 249, 250
- position: absolute, 290
- poster, 156
- powtarzane gradienty liniowe, 279
- pracownie z przeglądarkami, 40
- preload, 156
- preprocesory, 203
- preserveAspectRatio, 145
- procesor graficzny, 153, 398
 - odtworzenie filmów, 162
- profile, 60
- programowe aktywowanie elementów, 49
- projektowanie aplikacji mobilnych, 349
 - aplikacje narzędziowe, 354
 - aplikacje zwiększające produktywność, 352
 - dostęp do funkcji, 355, 358, 359
 - dynamiczne wczytywanie dodatkowych elementów, 366
 - ekran powitalny, 364
 - grupa docelowa, 350
 - ilość pamięci, 356
 - interfejs użytkownika, 359
 - jedno okno i jedna aplikacja naraz, 358
 - kwestie projektowe, 351
 - mały ekran, 356
 - menu rozwijane, 366
 - minimalizowanie ilości danych
 - wprowadzanych z klawiatury, 365
 - minimalna dokumentacja, 359
 - możliwości platformy mobilnej, 356
 - ograniczenia aplikacji, 397
 - poruszanie się po aplikacji, 359
 - ważne aplikacje rozrywkowe, 354
 - prostota, 355
 - przed rozpoczęciem pracy, 350
 - przyciski funkcyjne, 358
 - reguła 80 – 20, 355
 - rozrywka, 353
 - rozważania, 359
 - rysunek startowy, 363
 - standardy, 351
 - strony preferencji, 352
 - testowanie, 358
 - typy aplikacji, 352
 - ukrywanie treści, 366
 - user experience, 366
 - wezwanie do działania, 351
 - wybór stylu, 351
 - wybór typu, 355
 - zabawne aplikacje rozrywkowe, 354
 - zwięzłość, 365
- projekty dynamicznie dostosowywane do wielkości okna, 19
- protokoły
 - HTTPS, 109
 - SSL, 109
- przechwytywanie
 - dotknięcia, 373
 - zdarzeń, 374, 402
- przeglądarki
 - arkusze klienta, 204
 - badanie i debugowanie kodu źródłowego, 29
 - brakujące znaczniki dokumentu HTML5, 58
 - dostępne, 14
 - jako narzędzia programistyczne, 28, 29
 - mobilne, 350
 - przeciąganie, 51
 - sieciowe interfejsy API, 141
 - na potrzeby testów, 29
 - narzędzia dla programistów, 30
 - obsługujące nowe standardy sieciowe, 14
 - pobieranie dodatkowych elementów z serwera, 72
 - różne wersje na poszczególnych urządzeniach, 25
 - stacjonarne, 32, 35
 - starsze wersje, 14
 - strony HTML i style CSS, 21
 - ujednolicanie pracy, 204
 - urządzeń przenośnych, 13
 - w telefonach komórkowych, 20
 - wartości kolorów, 241
 - współczesne, 16
 - wyświetlanie żądanych stylów, 69
 - z wyłączoną obsługą JavaScriptu, 72
 - zewnętrzne arkusze stylów, referencje, 203
 - zgodne z HTML5, 18
- przejścia, 294
 - all, 301
 - czas, 298
 - dla grafiki tła, 297
 - dla transformacji, 309
 - dotyczące więcej niż jednego ustawienia, 301
 - hover, 294
 - opóźnienia, 299
 - różne, 301

- przejścia
 - tempo, 298
 - transition, 294
 - transition-delay, 294, 299
 - transition-duration, 294, 298
 - transition-property, 294, 295
 - transition-timing-function, 294, 298
 - właściwości, 295
 - wydajność, 323
- przekazywanie komunikatów pomiędzy dokumentami, 190
- przewijanie, 375
 - nieskończone, 399
 - usprawnienie, 384
- przewodnik po gestach dotykowych, 371
- przezroczystość, 273
 - dobawanie w formacie RGBA, 238
- przybliżanie, 402
- przyciski
 - atrakcyjne, 281
 - Enter, 122
 - Go, 122
 - gradient, 273, 275
 - grafika obramowania, 328
 - informacyjne, 355
 - o natywnym wyglądzie z iPhone'a, 262
 - opcji, 110
 - podwójny cień, 288
 - polecenia, 94
 - resetowania, 112
 - search_cancel_button, 122
 - standardowe, 362
 - trójwymiarowy wygląd, 282
 - tworzenie, 282
 - Wstecz, 362
 - wysyłania, 111
 - z systemu iOS, 275
- przytaczanie tytułów, 89
- pseudoelementy, 229
 - ::after, 230
 - ::before, 230
 - ::first-letter, 229
 - ::first-line, 229
 - ::selection, 230
 - ::webkit-validation-bubble, 130
 - ::webkit-validation-bubble-arrow, 130
 - ::webkit-validation-bubble-arrow-clipper, 130
 - ::webkit-validation-bubble-message, 131
 - notacja z podwójnym dwukropkiem, 231
- pseudoklasy, 219
 - ::after, 346
 - ::before, 346
 - :active, 220, 221
 - :checked, 110, 220, 221
 - :default, 223
 - :disabled, 105, 221
 - :empty, 223
 - :enabled, 221
 - :first-child, 223
 - :first-of-type, 223, 226
 - :focus, 220, 221
 - :hover, 220, 221, 294
 - :indeterminate, 221
 - :in-range, 223
 - :invalid, 100, 104, 116, 223
 - :lang(L), 227, 228
 - :last-child, 223
 - :last-of-type, 223, 226, 263
 - :link, 220, 221
 - :not(s), 108, 227, 228
 - specyficzność, 229, 233
 - :nth-child, 223, 225
 - :nth-last-child(n), 223
 - :nth-last-of-type(n), 223
 - :nth-of-type(n), 223, 225
 - :only-child, 223
 - :only-of-type, 223
 - :optional, 223
 - :out-of-range, 223
 - :placeholder-shown, 102
 - :read-only, 223
 - :read-write,
 - :required, 100, 223
 - :root, 223
 - :target, 227
 - :valid, 104, 223
 - :visited, 220, 221
 - dla interfejsu użytkownika, 220, 222
 - dla odnośników, 220
 - dla operacji użytkowników, 220
 - inne, 227
 - języka, 228
 - negacji, 228
 - nth, 224, 225
 - określające stan, 222
 - strukturalne, 223
 - związane z interakcją z użytkownikami, 220
- punkty
 - graniczne, 325
 - na cal, 341
- punkty zmiany koloru, 269, 272
 - szerokość gradientu, 279
 - twarde, 270, 281
 - background-size, 275
 - grafika tła, 277
 - tworzenie
 - paska nawigacji, 272
 - pasków, 277

R

- radiansy, 248
- ramki iFrame, 190
- range, 100
- readonly, 104, 105
- reguły, 196
 - @keyframes, 317
 - @supports, 339
 - @viewport, 368
 - 80 – 20, 355
 - kaskadowość, 233
 - specyficzność, 233
- rel, 68, 69, 198
 - alternate, 70
 - apple-touch-icon, 70
 - apple-touch-startup-image, 70
 - contents, 70
 - copyright, 70
 - glossary, 70
 - help, 70
 - icon, 70
 - index, 70
 - next, 70
 - prev, 70
 - stylesheet, 70, 198
 - wartości i ich definicje, 70
- requestAnimationFrame, 153
- required, 99
- responsive web design, 19, 381
- Retina, 341
- Retina Display, 341
- reversed, 83
- rgb(), 235, 236
 - składnia, 237
- RGBA, 19, 235, 238
 - cienie elementów, 239
 - dodawanie przezroczystości, 238
 - kanal alfa, 238
- Roboto, 19
- role, 51, 52
 - navigation, 79
 - WAI-ARIA, 45
- rozdzielczość, 341
 - rysunków, 341
 - wysoka, 276
- rozwiązania hybrydowe, 378
- rtl, 48
- RWD, 19
- rysunek startowy, 363

S

- Safari, 14, 30
 - ukrywanie paska nawigacji, 362
 - wykrywanie numerów telefonów, 88
 - wyróżnianie elementów, 84
- Safari 4.0, 12
- Safari WebKit, 13
- samochód klauna, 145
- samozamykający ukośnik, 55, 56
- sandbox, 91
- Sass, 205
- saturation, 235
- scoped, 70
- ScreenQueries, 31
- SDK, 15
- seamless, 91
- sekcje, 78
 - nagłówkek, 79
 - ogólna, 77
 - spis treści, 79
- selektory, 197
 - elementów, 207, 211
 - id, 46
 - klas, 208, 211
 - model Shadow DOM, 232
 - ogólne, 210
 - oparte na identyfikatorach, 205
 - typów, 207, 211
 - uniwersalne, 210
- selektory atrybutów, 215
 - kodu języka, 216
 - nazwy atrybutów, 218
- selektory CSS, 206, 408
 - .copyright, 209
 - p.copyright, 209
 - podstawowe, 207
 - specyficzność, 209, 409
- selektory CSS3, 19, 210, 405
 - #myParent a {}, 212
 - a[hreflang |= fr], 216
 - div > p {}, 213
 - E[atr\$=war], 217
 - E[atr*=war], 217
 - E[atr^=war], 217
 - li strong {}, 212
 - nazwa elementu, 211
 - p:first-of-type + p {}, 214
 - p > strong {}, 213
 - p strong {}, 212
 - proste, 228

- selektory CSS3
 - reguły oparte na kolejności kodu, 212
 - specyficzność, 233
 - stosowanie, 211
- selektory CSS4, 410
- selektory identyfikatorów, 209, 211
 - stosowanie, 212
- selektory relacyjne, 212
 - definicje i przykłady, 215
 - kombinator dziecka, 213
 - kombinator potomka, 212
 - kombinator przyległego brata, 214
 - ogólny kombinator brata, 214
- selektory strukturalne, 224
 - dolne obramowanie, 226
- semantyczne nazwy sekcji, 76
- Sencha Touch, 379
- Sencha.io, 396
- Sencha.io Src, 393
- SETTINGS, 168
- Shadow DOM
 - pseudoelementy, 130
 - style elementów, 232
- shadowBlur, 150
- shadowColor, 150, 151
- shadowOffsetX, 150
- shadowOffsetY, 150
- shortcut, 68
- sieci CDN, 191, 389
- sieciowe wątki robocze, 18, 72, 186
 - onmessage, 187
- silniki
 - Blink, 29
 - WebKit, 20, 32
- single point of failure, 394
- Sinon.JS, 42
- sizes, 69
- skalowalna grafika wektorowa, 141
- skrótowe deklaracje właściwości i wartości, 249
- skrypty, osadzanie, 394
- slider-horizental, 115
- słowa kluczowe
 - !important, 206
 - all, 295, 301
 - allow-forms, 91
 - allow-pointer-lock, 91
 - allow-popups, 91
 - allow-same-origin, 91
 - allow-scripts, 91
 - allow-top-navigation, 91
 - color-stop, 272
 - currentColor, 240
 - dla strony, 63
 - ease, 298
 - ease-in, 298
 - ease-in-out, 299
 - ease-out, 299
 - ellipsis, 285
 - even, 224
 - fill, 330
 - from, 317
 - inset, 286
 - linear, 298
 - odd, 224
 - off, 106
 - on, 106
 - rgb, 237
 - ścieżki gradientu, 267
 - to, 266, 267
 - transparent, 238, 240
- sms:, 87
- specyficzność, 233, 405
 - reguły, 233
 - selektory CSS, 409
 - style wewnętrzzwierszowe, 233
 - w kontekście kaskadowego dodawania stylów, 234
- specyfikacja
 - Basic UI, 285
 - WAI-ARIA, 51
- spellcheck, 51
- spinner, 100, 101
- SPOF, 394
- sprawdzanie poprawności, 128
 - automatyczne, 137
 - danych, 100
 - element.validity.customError, 130
 - element.validity.patternMismatch, 129
 - element.validity.rangeOverflow, 129
 - element.validity.rangeUnderflow, 129
 - element.validity.stepMismatch, 129
 - element.validity.tooLong, 129
 - element.validity.typeMismatch, 129
 - element.validity.valid, 130
 - element.validity.valueMissing, 129
 - validity, 129
- sprite, 344
 - a identyfikatory URI, 390
 - animacje, 320
 - używany w animacji postaci, 345
 - w formacie SVG, 146
 - z przyciskami, 264
 - z rysunkami, 389
- SQL, 179
- src, 71, 92, 155, 157
- srcdoc, 90

- srcLang, 159
 - stan urządzenia, 376
 - bateria, 377
 - interfejsy API, 377
 - połączenie sieciowe, 376
 - standardy aplikacji, 351
 - step, 101
 - steps(), 344
 - stopka, 80
 - zasięg, 76
 - stopnie, 247
 - storeValues(), 174, 175
 - stroke, 150
 - strony internetowe
 - dodawanie grafiki z podpisem, 82
 - interakcje z zawartością strony, 94
 - osadzanie plików dźwiękowych i wideo, 154
 - zwiększenie dostępności, 144
 - strony preferencji, 352, 354
 - strony ustawień, 352
 - style, 45, 47, 143, 197
 - background, 143
 - dla urządzeń i przeglądarek, 200
 - dodawanie
 - do dokumentu, 70
 - do witryny, 71
 - fill, 143
 - osadzane, 198, 203
 - projektowanie aplikacji, 351
 - stroke, 143
 - tekstu, 47
 - wewnątrzzwierszowe, 206
 - style CSS
 - bloki deklaracji, 197
 - najlepsze praktyki, 202
 - normalizowanie, 204
 - osadzanie w nagłówku dokumentu, 197
 - przekazywane kaskadowo, 208
 - resetowanie, 204
 - specyficzne dla sekcji, 205
 - specyficzność, 233
 - stosowanie identyfikatorów, 205
 - udostępnianie, 200
 - umieszczanie w zewnętrznym arkuszu stylów, 197
 - umieszczanie wewnątrzzwierszowo, 197
 - ustawione za pomocą selektora klasy, 208
 - używanie stylów ogólnych, 204
 - w wersji produkcyjnej, 203
 - Sublime Text, 28
 - subtitles, 159
 - suwaki, 101, 121
 - svg, 18, 141, 152
 - SVG, 141
 - <object>, 145
 - <title>, 143
 - alt, 145
 - Amaya, 146
 - deklaracja DTD, 143
 - dołączanie grafiki do dokumentów, 143
 - dynamicznie dopasowywana grafika
 - pierwszego planu, 144
 - nauka, 145
 - pliki, 142
 - właściwości, 145
 - rysowanie, 152
 - sprite'y, 143, 146
 - style CSS, 143
 - symulator, 38
 - iOS Simulator, 38
 - Opera Mini Simulator, 39
 - przewodzenie testów, 38
 - urządzeń BlackBerry, 39
 - systemy operacyjne
 - Android, 41
 - iOS, 40
 - mobilne, 39
 - Symbian OS, 42
 - WebOS, 42
 - Windows, 41
 - szybkość
 - pobierania i wysyłania danych, 387
 - połączenia, 342
- ## Ś
- ścieżki gradientu, 266, 272
 - kąt, 268
 - środowisko IDE, 28
 - środowisko mobilne, 349
 - ograniczenia, 350
 - przesyłanie dużych zasobów, 354
 - udostępnianie funkcji wersji stacjonarnej, 355
 - wprowadzanie danych, 365
 - wydajność, 381
- ## T
- tabele, naprzemienne kolorowanie wierszy, 224
 - tabindex, 48, 220
 - kolejność znaczników w kodzie strony, 49
 - tap-highlight-color, 230, 372
 - target, 87
 - techniki
 - Clown Car, 144
 - CORS, 191
 - oparte na grafice, 261
 - wykrywanie klienta, 347

- technologia WebGL, 148
- tekst
 - cienie, 283
 - kolumny, 326
 - prycinanie, 285
 - ustawianie, 284
 - wielokropek, 285
- tekstowe elementy semantyczne, 83
- tel., 87
- telefony, 40
 - Android, 41
 - iOS, 40
 - Nokia, 42
- telephone="no", 67
- testy
 - Chrome, 379
 - lokalne, 67
 - narzędzia specyficzne dla poszczególnych systemów operacyjnych, 358
 - przeglądarka stacjonarna, 379
 - urządzenia testowe, 380
 - w urządzeniach mobilnych, 379
 - zautomatyzowane, 42
- TextMate, 28
- text-overflow, 284, 285
 - ellipsis, 285
- text-shadow, 283
- Theora/Ogg, 154
- Tilt, 15
- Timeline, 399
- title, 45, 46, 94
- Touch, 371
- touch-action, 230
 - none, 373
- touch-callout, 230
 - none, 373
- touchcancel, 371
- touchend, 221, 371, 374
 - odbiornik dla zdarzenia, 402
- TouchEvent, 371
- TouchList, 371
- touchmove, 371
- touchstart, 221, 371
- transform, 304
 - łączenie właściwości, 308
- transformacje, 293, 302
 - backface-visibility, 312
 - iluzja głębi, 311
 - łączenie
 - wielu, 308
 - wszystkich elementów, 313
 - modyfikacja układu współrzędnych, 304
 - perspective, 311
 - przekrzywienie elementu, 307
 - przesuwanie elementu, 304, 310
 - rotowanie elementu, 306, 311
 - skalowanie względem osi, 310
 - stosowanie przejść, 309
 - transform, 303, 304
 - transform-origin, 303, 311
 - transform-style, 312
 - trójwymiarowe, 310
 - funkcje, 309
 - właściwości, 311
 - ustawianie punktu transformacji, 303, 311
 - widoczność odwróconego elementu, 312
 - wyświetlanie w przestrzeni trójwymiarowej, 312
 - zmiana wyglądu elementu, 304
- transition, 294
 - transition-delay, 294, 299
 - transition-duration, właściwości, 298
 - transition-property, 294, 295, 309
 - background-position, 297
 - background-size, 297
 - lista właściwości, 295
 - przykład właściwości, 297
 - visibility, 297
 - transition-timing-function, 294
 - właściwości, 298
- translate(), 304
- treści, 258
 - dodawanie ograniczeń, 91
 - generowanie, 230
 - za pomocą stylów, 219
 - małym drukiem, 89
 - podsumowanie, 93
 - powiązane, 79
 - usunięte z dokumentu, 88
- turns, 247
- twardy koniec wiersza, 83
- Twitter
 - identyfikatory URI, 343
- type, 69, 71, 83, 92, 98, 99, 115, 128
 - button, 114
 - checkbox, 109
 - color, 102, 122
 - pattern, 123
 - date, 102, 123, 125
 - pattern, 124
 - datetime, 125
 - datetime-local, 125
 - email, 99, 116, 129
 - autocomplete, 117
 - autofocus, 117
 - disabled, 117
 - form, 117
 - list, 117
 - maxlength, 117
 - multiple, 116
 - name, 117

- pattern, 117
- placeholder, 117
- readonly, 117
- required, 117
- size, 117
- file, 112
 - accept, 113
 - capture, 113
 - multiple, 113
 - value, 113
- hidden, 114
- image, 114
 - alt, 114
 - src, 114
- month, 125
 - step, 125
- number, 119, 120, 129
 - max, 119, 120
 - min, 119, 120
 - pattern, 120
 - spinner, 120
 - step, 119, 120
- password, 109
- radio, 110
 - name, 110, 111
 - value, 110
- range, 121
 - max, 121
 - min, 121
 - step, 121
 - value, 121
- reset, 112
- search, 122
- submit, 111
 - disabled, 111
 - name, 111
 - value, 111
- tel, 115, 118
 - pattern, 119
 - placeholder, 119
- text, 108
- text/css, 198
- time, 125
 - max, 125
 - min, 125
 - step, 126
- url, 117, 129
 - pattern, 117
- value, 108
 - accept, 113
- week, 126
- typy MIME, 198
- tytuł dokumentu, 57

U

- UA string, 347
- udostępnianie grafiki, 340
 - background-size, 343
 - czcionki z ikonami, 345
 - grafika tła dla wyświetlaczy o różnej gęstości, 345
 - identyfikatory URI danych, 343
 - image-set(), 345
 - sprite'y, 344
 - w zależności od szybkości połączenia, 342
- układy
 - dostosowujące się do wymiarów ekranu, 340
 - elastyczny, 334
 - flexbox, 334
 - płynne, 340
 - wielokolumnowe, 327
- url(), 389
- urządzenia przenośne
 - Android, 41
 - aplikacje w formie pakietów, 378
 - automatyczne wykrywanie numerów telefonu, 67
 - BlackBerry, 41
 - charakterystyczne zagadnienia, 20
 - dostęp do sprzętu, 375
 - dostosowanie witryny, 39
 - dostosowywanie stron do małych ekranów, 367
 - dotyk, 367, 368
 - interfejs użytkownika, 21
 - iOS, 40
 - Kindle, 42
 - natywne aplikacje sieciowe, 378
 - Nokia, 42
 - obsługa
 - odnośników, 87
 - samodzielnych plików .svg, 142
 - okno przeglądarki, 64
 - pamięć, 356
 - pasek stanu, 66
 - pliki CSS, 203
 - rozwiązania hybrydowe, 378
 - ruch i kierunek poruszania telefonu, 375
 - stan urządzenia, 376
 - testowanie aplikacji, 40, 379
 - użytkownicy, 21, 350
 - wartości charakterystyczne dla producentów, 66
 - wielozadaniowość, 358
 - Windows, 41
 - wydajność, 381
 - wykrywanie orientacji, 202
 - wyświetlacze o wysokiej rozdzielczości, 341

- User Agent switcher, 30
- user experience, 183, 362, 366
- userid, 106
- user-scalable, 65
- user-scalable=no | yes, 65
- user-select, 372
 - none, 230
- ustawienia, 352
- UTF-8, 63
- użytkownicy
 - aplikacji narzędziowych, 354
 - aplikacji zwiększających produktywność, 352
 - mobiloni, 403
 - projektowanie aplikacji mobilnych, 350
 - typowi, 350
 - urządzeń przenośnych, 21
 - wciągających aplikacji, 353

V

- valid, 130
- validityStateObject, 129
- value, 83, 108
 - podawanie instrukcji, 108
- Vibration, 377
- viewbox, 145
- VP8, 154, 155

W

- W3C, 145
- W3C mobileOK Checker, 39
- WAI-ARIA, 51, 193
- warstwy
 - określające działanie witryny, 196
 - prezentacji, 196, 368
 - rozdzielanie, 198
 - treści, 196
 - węzły, 386
- wartości, 197
 - DPI, 276
 - kolorów w przeglądarkach, 241
 - skrócone deklaracje, 249
 - szesnastkowe, 236
- watchCurrentPosition(), 184
- wciągające aplikacje, 353
- Web Inspector, 30, 37
 - Shadow DOM, 232
- web inspector remote, 34
- Web Intents, 377
- Web SQL, 179
 - drop, 182
 - insert, 181
 - metody, 179
 - select, 181
- Web.app, 378

- WebGL, 148
 - obsługa, 153
 - procesor graficzny, 153
- webkit/moz/ms-user-select, 220
- webkit-tap-highlight-color, 220
- webkit-touch-callout, 220
- webkit-user-select, 372
- WebM, 154
- WebOS, 42
- WebPageTest.org, 391
- WebStorm, 28
- weinre, 34, 391
 - biblioteki, 34
 - instalacja, 35
 - serwer debugera, 35
 - używanie debugera, 35
 - zakładki, 35
- wewnętrzne kotwice, 46
- węzły
 - pula węzłów wielokrotnego użytku, 399
 - wydajność, 399
- white-space, 285
 - clip, 285
- width, 156, 200, 255, 284, 328
- width=<liczba> | device-width, 65
- widżety
 - informacyjne, 92
 - style, 71
- wielkość ekranu, 20
 - ScreenQuer.es, 31
- wielkość obrazu
 - dynamiczne dopasowywanie, 162
- wieloplatformowy transkoder wideo, 163
- window.matchMedia, 201
- window.navigator.standalone, 66
- Windows Phone, 41
 - kafelki, 70
- Windows Phone Emulator, 39
- witryny
 - dynamicznie dopasowywanych rysunków, 145
 - filmy o zmiennej wielkości, 162
 - mobilne, 21
 - poruszanie się po witrynach, 48
 - upraszczanie witryn mobilnych, 349
 - wygląd i styl, 196
- właściwości, 197
 - appearance, 114
 - color, 290
 - content, 230
 - customError, 130
 - length, 171
 - navigator.battery.charging, 377
 - navigator.battery.chargingTime, 377
 - navigator.battery.dischargingTime, 377
 - onmessage, 187

- origin, 191
- rangeOverflow, 129
- rangeUnderflow, 129
- skrótowe, deklaracje, 249
- sliderthumb-horizontal, 115
- stepMismatch, 129
- tła, 275
- tooLong, 129
- visibility, 319
- Worker(), 187
- WP8, 154
- wtyczka ADB, 32
- wybór składni do podawania kolorów, 244
- wycieki pamięci, 357
- wydajność
 - animacje, 323
 - antywzorce, 394
 - czas pracy na baterii, 381
 - liczba żądań, 384
 - minimalizowanie modelu DOM, 399, 401
 - opóźnienie, 387
 - procesor graficzny, 398
 - przejścia, 323
 - szybkość reagowania interfejsu użytkownika, 401
 - w środowisku mobilnym, 381
- wykrywanie
 - funkcji, 372
 - klienta, 347
- wymagane komponenty HTML5, 57
- wyodrębnianie tekstu, 88
- wyrażenia
 - liczbowe, 225
 - regularne, 102
 - metaznaki, 103
- wyróżnik tekstu, 84
- podkreślenie, 89
- selektor atrybutu kodu języka, 216
- stylistyczny, 88
- wyszukiwarki, indeksowanie danych na
 - podstawie języka, 47
- wyświetlacze, gęstość, 341
- wzbogacenie user experience, 183
- wzorce projektowe, 351

X

- XHTML, 43
 - deklaracje DTD, 59
 - komponenty wymagane do poprawności
 - strony, 57
- XSS, 190

Y

- YSlow, 396
- YSlow Yahoo!, 403

Z

- zagnieżdżanie elementów, 55
- zaokrąglone narożniki, 261
- zapytania
 - media, 19, 69, 144, 200, 325
 - supports, 201
- zarządzanie
 - pamięcią, 357, 395, 399
 - zużyciem energii, 382
- zasięg nagłówków i stopek, 76
- zasoby tekstowe, 392
- zdalne debugowanie, 32
- zdarzenia
 - czas wczytywania, 401
 - przechwytywanie, 374
 - zmiany ustawień klienta, 202
 - związane z dotykiem, 369, 370, 401
 - związane z kliknięciami, 369, 402
 - związane z myszą, 369
 - związane z oknami, 375
 - związane z połączeniem, 165
 - związane z urządzeniami wskazującymi, 370
- zewnętrzne arkusze stylów, 197
 - atrybut media, 199
 - korzyści, 203
 - przeznaczenie plików, 199
 - używanie, 198
- zmienne
 - gameDuration, 177
 - isTouchEnabled, 372
 - storageType, 181
 - validityStateObject, 129
- zmniejszanie liczby żądań HTTP, 388
 - CSS3, 253
 - pamięć lokalna, 204
 - style CSS, 202, 397
- zmniejszanie wielkości żądań, 392
 - style CSS, 397
- znaczniki
 - do grupowania semantycznego, 17
 - domyślna kolejność, 49
 - meta dla aplikacji mobilnych, 64
 - opcjonalnych atrybutów, 44
 - otwierające, 44, 54
 - zagnieżdżanie, 56
 - zamykające, 44
- znaki niealfanumeryczne, 218

Ż

- żądania HTTP, 202
- żądania sieciowe
 - analizowanie, 391
 - szczegóły, 392
- żyroskop, 376

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA WYDAWNICZA

 **Helion SA**

HTML5. Strony mobilne



HTML5 to prawdziwa rewolucja w świecie stron WWW i aplikacji internetowych. Jego potencjał oraz możliwości zaskoczyły niejedną osobę. Wykorzystywany jest do tworzenia atrakcyjnych, intuicyjnych i dynamicznych stron WWW. To nie wszystko! Dzięki HTML5 i CSS3 możesz także stworzyć aplikację działającą na urządzeniach z systemami Android lub iOS. Chcesz zobaczyć, jak to zrobić?

Jeśli tak, to trafiłeś na świetną książkę, która wprowadzi Cię w świat nowoczesnych stron i aplikacji. W trakcie lektury będziesz poznawać składnię oraz możliwości HTML, przygotujesz środowisko pracy oraz zdobędziesz potrzebne narzędzia. Kolejne rozdziały to obowiązkowa dawka informacji na temat tworzenia formularzy oraz korzystania z elementów svg, canvas, audio i video. HTML5 dostarcza wielu nowych narzędzi, pozwalających m.in. sprawdzić stan połączenia z siecią, przechowywać dane na komputerze użytkownika lub uzyskać informację o jego lokalizacji. Dzięki tej książce opanujesz je w mig. W tym podręczniku znajdziesz również obszerny opis kaskadowych arkuszy stylów w wersji 3. Pozwolą Ci one błyskawicznie wprowadzać atrakcyjne dla oka efekty. Książka ta jest genialnym źródłem informacji dla wszystkich pasjonatów tworzenia stron i aplikacji internetowych.

Dzięki tej książce:

- poznasz składnię HTML5 i CSS3
- zaznajomisz się z możliwościami API
- zbudujesz zaawansowaną stronę lub mobilną aplikację internetową

Sprawdź, jak stworzyć aplikację internetową działającą na platformie Android lub iOS!

helion.pl
księgarnia
internetowa

Nr katalogowy: 19756



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/novosci>

Helion SA

ul. Kościuszki 1c, 44-100 Gliwice

tel.: 32 230 98 63

e-mail: helion@helion.pl

<http://helion.pl>

sięgnij po **WIECEJ**



KOD KORZYŚCI

ISBN 978-83-246-8912-5



Cena 69,00 zł