

Tworzenie animacji z użyciem języka JavaScript

WPROWADZENIE DO TECHNIK ANIMACJI

Julian Shapiro

Helion 

Tytuł oryginału: Web Animation using JavaScript: Develop & Design

Tłumaczenie: Mirosław Gołda

ISBN: 978-83-283-1572-3

Authorized translation from the English language edition, entitled: WEB ANIMATION USING JAVASCRIPT: DEVELOP & DESIGN; ISBN 0134096665; by Julian Shapiro; published by Pearson Education, Inc, publishing as Peachpit Press.

Copyright © 2015 by Julian Shapiro.

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 Pearson Education, Inc.

Polish language edition published by HELION S.A. Copyright © 2016.

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)

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:

<ftp://ftp.helion.pl/przyklady/twanjs.zip>

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

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

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

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

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

SPIS TREŚCI

Wstęp	7
Wprowadzenie	9
ROZDZIAŁ 1 ZALETY ANIMACJI JAVASCRIPT	10
Animacje JavaScript a animacje CSS	12
Wspaniała wydajność	14
Możliwości	15
Proces łatwy w utrzymaniu	17
Podsumowanie	18
ROZDZIAŁ 2 TWORZENIE ANIMACJI ZA POMOCĄ VELOCITY.JS	20
Typy bibliotek animacji JavaScript	22
Instalacja jQuery i Velocity	23
Praca z Velocity: podstawy	24
Praca z Velocity: opcje	28
Praca z Velocity: dodatkowe funkcje	34
Praca z Velocity: bez jQuery (średnio zaawansowane)	37
Podsumowanie	39
ROZDZIAŁ 3 TEORIA PROJEKTOWANIA RUCHU	40
Projekt ruchu poprawia wrażenia użytkownika	42
Użyteczność	45
Elegancja	50
Podsumowanie	55
ROZDZIAŁ 4 PRACA Z ANIMACJAMI	56
Praca z animacjami CSS	58
Technika kodowania: rozdzielanie stylowania od logiki	61
Technika kodowania: organizacja sekwencji animacji	66
Technika kodowania: dodawanie efektów	70
Techniki projektowania	73
Podsumowanie	77

ROZDZIAŁ 5	ANIMACJA TEKSTU	78
	Standardowe podejście do animacji tekstu	80
	Przygotowujemy elementy tekstu do animacji za pomocą Blast.js	82
	Wprowadzanie tekstu do widoku i wyprowadzanie go z niego	89
	Zmiana indywidualnych elementów tekstu	92
	Bardziej wyszukane zmiany tekstu	93
	Floresy tekstowe	94
	Podsumowanie	96
ROZDZIAŁ 6	PODSTAWOWE INFORMACJE O SVG	98
	Tworzenie obrazków za pomocą kodu	100
	Język znaczników SVG	101
	Stylowanie SVG	103
	Wsparcie dla SVG	104
	Animacje SVG	105
	Przykład implementacji: animowane logo	108
	Podsumowanie	109
ROZDZIAŁ 7	WYDAJNOŚĆ ANIMACJI	110
	Realia wydajności stron internetowych	112
	Technika: zapobieganie uszkodzeniu układu strony	115
	Technika: seryjne dodawanie elementów do DOM	119
	Technika: unikanie wpływu na sąsiednie elementy	122
	Technika: unikanie równoległego wczytywania	124
	Technika: nie reaguj w sposób ciągly na zdarzenia przewijania i zmiany rozmiaru przeglądarki ...	126
	Technika: redukovanie generowanego obrazu	127
	Technika: degradacja animacji na starszych przeglądarkach	129
	Odnajdź właściwy próg wydajności na wczesnym etapie	131
	Podsumowanie	135
ROZDZIAŁ 8	POKAZ ANIMACJI	136
	Zachowanie	138
	Struktura kodu	140
	Sekcja kodu: Konfiguracja animacji	142
	Sekcja kodu: Tworzenie kół	143
	Sekcja kodu: Animacja kontenera	144
	Sekcja kodu: Animacja kół	147
	Podsumowanie	151
	SKOROWIDZ	153

ROZDZIAŁ 1.

Zalety animacji JavaScript

W tym rozdziale porównamy możliwości języka JavaScript oraz stylów CSS w zakresie tworzenia animacji i poznamy unikalne możliwości i korzyści w zakresie organizacji pracy przemawiające za zastosowaniem języka JavaScript.

Mówiąc w skrócie, przedstawię tu kontekst potrzebny do zrozumienia wszystkiego, czego nauczysz się z tej książki.

ANIMACJE JAVASCRIPT A ANIMACJE CSS

W społeczności webdeveloperskiej istnieje fałszywe przekonanie, że jedynym wydajnym rozwiązaniem w zakresie tworzenia animacji na stronach internetowych jest stosowanie animacji CSS. To nieporozumienie zaowocowało tym, że wielu deweloperów porzuciło całkowicie animacje oparte na języku JavaScript, co zmusiło ich do:

- Zarządzania całym interfejsem użytkownika za pomocą arkuszy stylów, co szybko staje się trudne w utrzymaniu.
- Rezygnacji z kontroli nad upływem czasu, dostępnej jedynie za pośrednictwem JavaScriptu. (Kontrola nad czasem potrzebna jest przy projektowaniu animacji w interfejsach odpowiadających na przeciąganie elementów, spotykanych na przykład w aplikacjach mobilnych).
- Rezygnacji z fizyki ruchu, która pozwala obiektom zachowywać się w sposób charakterystyczny dla realnego świata.
- Utraty wsparcia dla starszych wersji przeglądarek, które są wciąż popularne na świecie.

Animacje oparte na języku JavaScript są często równie szybkie jak animacje CSS. Animacjom CSS przypisywana jest niesłusznie duża przewaga szybkości w porównaniu z animacjami JavaScript. Wynika to z faktu częstego bezpośredniego utożsamiania animacji JavaScript z animacjami wprowadzonymi w bibliotecę jQuery, które są rzeczywiście bardzo wolne. Alternatywne rozwiązania w postaci bibliotek JavaScript, które pomijają całkowicie wykorzystanie jQuery, potrafią być niesamowicie wydajne poprzez usprawnienie interakcji ze stroną.



UWAGA: Jedną z bibliotek, z której będziemy korzystać w tej książce, jest *Velocity.js*. Jest ona lekka, ale bardzo rozbudowana i odwzorowuje składnię animacji jQuery, aby ułatwić naukę.

Animacje CSS sprawdzają się jednak doskonale przy animacjach wyświetlanych w wyniku ruchu myszką (na przykład przy zmianie koloru łącza po najechaniu kursorem na kolor niebieski). Przekształcenia CSS wpasowują się świetnie w istniejące arkusze stylów, co pozwala deweloperom uniknąć zaśmiecania strony niepotrzebnymi bibliotekami JavaScript. Co więcej, animacje CSS są bardzo szybkie.

W tej książce pokażę jednak, dlaczego JavaScript jest często lepszym rozwiązaniem przy animacjach bardziej złożonych niż proste zmiany stanu.

Nie utożsamiaj JavaScriptu z jQuery

WSPANIAŁA WYDAJNOŚĆ

JavaScript i jQuery są mylnie utożsamiane ze sobą. Animacje JavaScript są szybkie, a jQuery je spowalnia. Pomimo niesamowitej mocy tej biblioteki nie została ona zaprojektowana jako wysokowydajny silnik animacji. Nie posiada mechanizmu obronnego przed „uszkodzeniem układu strony”, co skutkuje przeciążeniem przeglądarki zadaniami przetwarzania układu podczas odtwarzania animacji.

Co więcej, z uwagi na fakt, że kod bazowy jQuery służy nie tylko animacjom, ale także wielu innym celom, zużywa on dużo pamięci i wywołuje mechanizmy czyszczenia pamięci w przeglądarce, co doprowadza do przestojów w działaniu animacji. A do tego, w wyniku podjętych przez zespół jQuery decyzji mających ułatwić początkującym użytkownikom tworzenie przyzwoitej jakości kodu interfejsu użytkownika, jQuery zrezygnowało z zalecanej praktyki wykorzystywania funkcji `requestAnimationFrame`, która pozwala na znaczne zwiększenie liczby wyświetlanych klatek w animacji internetowej.

Biblioteki animacji JavaScript, które nie korzystają wcale z mechanizmów jQuery, oferują fenomenalną wydajność dzięki zastosowaniu przepływów interakcji na stronie. Jedną z bibliotek, którą wykorzystamy w tej książce, jest `Velocity.js`. Biblioteka ta jest lekka, ale też bogata w funkcjonalności, a jej składnia odwzorowuje składnię animacji jQuery, co znacznie ułatwia naukę.

Tematem wydajności zajmiemy się w rozdziale 7., „Wydajność animacji”. Poznając niuanse wydajności renderowania w przeglądarce, zyskasz podstawy wiedzy, dzięki którym zbudujesz animacje działające prawidłowo na wszystkich przeglądarkach i urządzeniach, niezależnie od ich szybkości przetwarzania.

MOŻLIWOŚCI

Szybkość nie jest oczywiście jedynym powodem, dla którego powinniśmy zdecydować się na użycie języka JavaScript — równie istotne jest bogactwo dostępnych funkcjonalności.

PRZEWIJANIE STRONY

Przewijanie strony jest jednym z najczęstszych zastosowań dla animacji opartej na języku JavaScript. Najnowszym trendem w zakresie projektowania stron internetowych jest tworzenie długich stron, które po przewinięciu w dół są rozbudowywane o nowe treści za pomocą animacji.

Biblioteki animacji JavaScript, takie jak Velocity, udostępniają proste funkcje, które umożliwiają dodanie treści do strony:

```
$element.velocity("scroll", 1000);
```

Rezultatem działania tego kodu jest przewinięcie przeglądarki do górnej krawędzi elementu `$element` w ciągu 1000 milisekund. Warto zauważyć, że składnia Velocity jest niemal identyczna ze składnią funkcji `$.animate()` języka jQuery, którą zajmujemy się w dalszej części tego rozdziału.

ODWRÓCENIE ANIMACJI

Odwróceniem animacji nazywamy w skrócie odwrócenie efektów poprzedniej animacji na wybranym elemencie. Polecenie `reverse` instruuje element, że ma za pomocą animacji wykonać powrót do stanu sprzed animacji. Powszechnym zastosowaniem jest w tym przypadku okno dialogowe, które najpierw się pojawia, a następnie chowa po zamknięciu przez użytkownika.

Nieoptymalizowany proces odwracania animacji mógłby polegać na śledzeniu szeregu właściwości elementów poddawanych ostatnio animacji, których efekty mogą zostać odwrócone. Utrzymywanie tych wszystkich stanów animacji w kodzie interfejsu użytkownika stałoby się jednak szybko trudne w utrzymaniu. Polecenie `reverse` biblioteki Velocity wykonuje tę pracę za nas.

Składnia polecenia `reverse` przypomina składnię polecenia `scroll`. Wywołujemy je, przekazując jako pierwszy argument ciąg `"reverse"`:

```
// Pierwsza animacja: Zmniejszamy wartość opaciy
// (nieprzezroczystość) do zera
$element.velocity({ opacity: 0 });
// Druga animacja: Odwracamy poprzedni efekt
// i zwiększamy wartość opaciy do 1
$element.velocity("reverse");
```

Kontrola nad czasem trwania animacji oferowana przez JavaScript nie polega jedynie na jej odwracaniu: JavaScript umożliwi również globalne spowolnienie lub przyspieszenie wszystkich wykonywanych aktualnie animacji. Więcej na ten temat dowiesz się z rozdziału 4., „Praca z animacjami”.

RUCH WYKORZYSTUJĄCY PRAWA FIZYKI

Wykorzystanie fizyki w projektowaniu animacji jest czynnikiem umożliwiającym uzyskanie wspaniałych doznań użytkownika (UX) korzystającego z Twojej strony: interfejsów, które w naturalny sposób reagują na jego działania. Inaczej mówiąc — interfejsów naśladowujących ruch obiektów w prawdziwym świecie.

Jako proste, a zarazem rozbudowane wprowadzenie w możliwości Velocity w zakresie fizyki ruchu pokażę efekt wygładzania ruchu (ang. *easing*) bazujący na fizyce sprężyny. (Koncepcją wygładzania ruchu zajmę się w następnym rozdziale). W celu wykorzystania standardowych ustawień wygładzania przekazujemy ciąg znaków odpowiadający zdefiniowanej krzywej wygładzania (na przykład "ease" lub "easeInOutSine"). Typ wygładzania fizyki sprężyny przyjmuje z kolei dwuelementową tablicę.

```
// Animujemy szerokość elementu do "500px" z wykorzystaniem
// łagodzenia w fizyce sprężyny, korzystając z jednostek
// 500 nacisku i 20 jednostek tarcia
$element.velocity({ width: "500px" }, { easing: [ 500, 20 ] });
```

Elementy tablicy reprezentują kolejno napięcie i tarcie symulowanej sprężyny. Wyższe wartości nacisku zwiększają ogólną szybkość i sprężystość animacji. Niższe wartości zwiększają szybkość wibracji w końcowym etapie animacji. Modyfikacja tych wartości pozwala nadać animacjom niepowtarzalny profil ruchu, co z kolei pozwala uwydatnić ich indywidualne zachowania.

PROCES ŁATWY W UTRZYMANIU

Projektowanie animacji jest eksperymentalnym procesem wymagającym wielokrotnych modyfikacji wartości czasu i wygładzania w celu uzyskania unikalnego efektu na stronie. Niemal pewne jest, że właśnie wtedy, gdy już maksymalnie dopracujesz swój projekt, klient zażąda wprowadzenia jakichś znaczących zmian. W takich momentach najważniejsze jest posiadanie kodu, który jest łatwy w utrzymaniu.

Rozwiązanie tego problemu w JavaScriptcie jest bardzo eleganckie i zostanie opisane dokładnie w rozdziale 4., „Praca z animacjami”. W tej chwili przedstawię tylko krótkie wyjaśnienie: istnieją techniki łączenia ze sobą pojedynczych animacji JavaScript — o różnym czasie trwania, wygładzaniu itd. — w których czas działania jednej animacji nie wpływa na pozostałe. Oznacza to, że mamy możliwość zmiany pojedynczych czasów trwania bez konieczności wykonywania przeliczeń i możemy w prosty sposób modyfikować ustawienia tak, aby działały równolegle lub w ustalonej kolejności.

PODSUMOWANIE

Tworząc animacje z wykorzystaniem CSS, ograniczeni jesteśmy do funkcji udostępnianych przez specyfikację CSS. Język JavaScript, dzięki specyficie języków programowania, umożliwia tworzenie bibliotek, które posiadają nieograniczone możliwości kontroli. Silniki animacji wykorzystują te możliwości do przygotowywania rozbudowanych funkcjonalności, które znacznie usprawniają pracę i poszerzają możliwości interakcji tworzonych animacji. To właśnie jest tematem tej książki: projektowanie pięknych animacji w jak najefektywniejszy sposób.

W następnym rozdziale wyjaśnię, jak korzystać z *Velocity.js*, silnika animacji, który wybrałem na potrzeby tej książki. Aby nabrać biegłości w stosowaniu tego silnika, musimy zrozumieć, jak wykorzystać funkcje już poznane i wiele, wiele innych.

SKOROWIDZ

A

Adobe After Effects, 80
Adobe Photoshop, 100
animacja, 9, 22, 66
 3D, 144
 akcja
 hierarchia ważności, 47
 o nieodwracalnych konsekwencjach, 47
 równoległa, 47, 124
 CSS, 12, 58, 61
 wady, 60
 zalety, 59
 czas trwania, 28, 48, 50, 54, 73
 efekt wygładzania ruchu, 16, 28, 29, *Patrz też:*
 wygładzanie
 JavaScript, 12, 14, 59
 optymalizacja, 62, 67, 68, 69, 70
 przewijanie strony, 15
 zalety, 60, 64, 68, 69, 70
 jQuery, 61
 kod, *Patrz:* kod
 kolejkowanie, 30
 koloru elementu, 46
 multimedia, 127
 odbijania, 46
 odbijanie, 48, 50
 odwrócenie, 15, 34
 ograniczanie, 48, 50
 opóźnienie, 32, 73
 pokaz, 138
 punkt początkowy, 52
 redukcja różnorodności, 47, 48
 rozbicie na etapy, 51
 rozciąganie w czasie, 51, 90, 124
 separacja logiki, 61, 64
 SVG, 105, 106, 108, 109
 tekstu, *Patrz:* tekst animacja
 teoria ruchu, *Patrz:* teoria ruchu
 wczytywania strony, 66
 właściwości, 25
 wpływ na elementy sąsiednie, 122

wydajność, 112, 129, 131
 wykorzystanie praw fizyki, 16, 42
argument, 24, 25, 67
atrybut, 104, 109
 x, 106
 cx, 106
 height, 101
 pozycyjny, 106
 prezentacji, *Patrz:* atrybut prezentacji
 prezentacyjny, 105
 width, 101
 y, 106

B

Bézierra krzywa, *Patrz:* krzywa Bézierra
biblioteka
 animacji, 22
 Blast, *Patrz:* Blast
 GSAP, *Patrz:* GSAP
 jQuery, *Patrz:* jQuery
 Snap.svg, 104
 Underscore, 126
 Velocity, *Patrz:* Velocity
Blast, 82, 83, 84, 88
 cofanie działania, 87
 instalacja, 84
 opcja
 customClass, 85
 delimiter, 84
 generateValueClass, 85, 86
 tag, 87
blok tekstu, 80

C

canvas, *Patrz:* płótno
CSS
 animacja, *Patrz:* animacja CSS
 gradient, 128
 selektor nth-child, 92

D

debouncing, *Patrz:* uchwyt zdarzenia odbijanie
DOM, 115, 126
element
 dodawanie, 116
 dodawanie seryjne, 119, 120, 143
 ustawianie struktury, 116, 119

E

easing, *Patrz:* animacja efekt wygładzania ruchu
element
 DOM, 116, 119, 143
 img, 128
 SVG, *Patrz:* grafika SVG
 węzeł, *Patrz:* węzeł elementu
event handler, *Patrz:* uchwyt zdarzenia

F

flores, 94
force-feeding, *Patrz:* właściwość wymuszanie
 wartości
format
 PNG, 52
 SVG, 52, 100, 101, 103, 108, 109
funkcja
 keyframes, 58
 wartości, 147
 zwrotna, *Patrz:* wywołanie zwrotne

G

grafika, 128
 gradient, 128
 PNG, 52
 SVG, 52, 100, 101, 106, 108, 109
 atrybut prezentacji, 101
 stylowanie, 103, 104, 109

I

Inkscape, 100
inline status indication, *Patrz:* wskaźnik stanu
 w postaci wstawki

interfejs użytkownika, 9
 animacja, 22
 projektowanie, 42
 wydajność, 112, 115, 116
Internet Explorer, 129, 131

J

jednostki
 %, 26
 deg, 26
 em, 26
 px, 26, 27
 rem, 26
 vh, 26
 vw, 26
JEO, 116
język
 LESS, 73
 SASS, 73
jQuery, 12, 14, 84
 funkcja
 animate, 21, 61
 eq, 92
 hide, 72
 show, 72
 instalacja, 23
 obiekt elementu, *Patrz:* JEO
jQuery element objects, *Patrz:* JEO

K

kod
 ekspresyjny, 59
 łatwy w utrzymaniu, 59
 związły, 59
komponent niskiego poziomu, 80
krzywa Béziera, 29

L

layout trashing, *Patrz:* uszkodzenie układu
LESS, 73

Ł

łącze, 80

M

monitora wymiary, 142
multimedia, 127
 animacja, *Patrz:* animacja multimedia
multiplikator czasu, 73

O

operator
 matematyczny, 27
 wartości, 27

P

pakiet UI, *Patrz:* UI
pętla, 30
plik, *Patrz:* format
 obrazu, 127
 PNG, *Patrz:* format PNG
 SVG, *Patrz:* format SVG
 wideo, 127
plótno, 101
polecenie
 reverse, 15, 34, 149, 150
 scroll, 34, 35

S

SASS, 73
sequence running, *Patrz:* UI uruchamianie sekwencji
Sketch, 100
staggering, *Patrz:* animacja rozciągnięcie

T

tabela, 80
tekst, 80
 animacja, 89, 90, 93, 94
 flores, *Patrz:* flores
 separacja, 82
 zmiana indywidualnych elementów, 92
 zmiana widoczności, 80
teoria ruchu, 16, 42
 elegancja, 42, 44, 50
 użyteczność, 44, 45, 50

U

uchwyt zdarzenia odbijanie, 126
UI, 66, 70, 82, 90
 uruchamianie sekwencji, 66
uszkodzenie układu, 115, 116, 126

V

Velocity, 12, 14, 21, 22, 82, 84, 90, 104
 argument, *Patrz:* argument
 efekt, 70
 3D, 93
 rejestrwanie, 70, 72
 transition.fadeIn, 90, 91, 93
 transition.perspectiveDownIn, 93
 transition.shrinkIn, 93
 wygaszania elementu, 70
 wywoływanie, 70
instalacja, 23
opcja, 28, 30
 backwards, 91
 begin, 28, 30, 34
 complete, 28, 30, 34
 container, 35
 delay, 28, 32, 33, 73, 146
 display, 28, 32, 33
 duration, 28, 73
 easing, 28, 29, *Patrz też:* wygładzanie
 loop, 30
 stagger, 90, 91, 124
 visibility, 32, 33
przekształcenie, 36
składnia wieloargumentowa, 67
UI, *Patrz:* UI
 właściwość
 mock, 74
 scale, 70
 współpraca z jQuery, 24
 wywołanie, 27
Velocity Motion Designer, *Patrz:* VMD
VMD, 74, 76

W

węzeł

- elementu, 83
- tekstowy, 80, 83

właściwość, 25

- box-shadow, 26
- left, 123
- margin, 26
- nazwa, 26
- opacity, 123
- padding, 25
- perspective, 144, 145
- prospective-origin, 144, 145
- przekształceń, 122, 123
- rotateX, 122, 144
- rotateY, 122, 144
- rotateZ, 122
- scaleX, 122
- scaleY, 122
- text-shadow, 26
- top, 123
- transform, 26
- transition, 59
- translateX, 122
- translateY, 122, 123

translateZ, 144

wymuszanie wartości, 117, 118, 145

wskaźnik stanu

- tradycyjny, 46
- w postaci wstawki, 46

wygładzanie

- CSS, 29
 - ease, 29
 - ease-in, 28, 29
 - ease-in-out, 29
 - ease-out, 28, 29
 - easeOutQuad, 150
 - fizyka sprężyny, 29
 - liniowe, 28
 - spring, 29
 - trygonometryczne, 29
 - w oparciu o krzywą Béziera, 29
- wywołanie zwrotne, 30

Z

Zepto, 84

znacznik

- div, 80
- span, 80, 83
- svg, 101

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>

Twój unikalny przewodnik po świecie animacji JavaScript!

Do niedawna umieszczenie zaawansowanej animacji na stronie WWW wymagało zastosowania technologii flashowej opracowanej przez firmę Adobe. Takie rozwiązanie powodowało wiele zagrożeń, a ponadto konieczna była instalacja dodatkowego oprogramowania. Ciągły rozwój możliwości przeglądarek, języka JavaScript oraz HTML umożliwił w końcu tworzenie nawet bardzo zaawansowanych animacji bez potrzeby korzystania z zewnętrznych narzędzi.

Jeżeli chcesz się nauczyć tworzyć animacje, czerpiąc z możliwości JavaScript, trafieś na właściwy podręcznik. W trakcie lektury kolejnych rozdziałów zdobędziesz bezcenną wiedzę na temat biblioteki Velocity.js oraz zasad projektowania ruchu. Potem przejdziesz do praktycznych aspektów pracy z animacjami — dowiesz się, jak rozdzielić style i logikę, jak zorganizować różne sekwencje animacji oraz jak dodawać efekty. Następnie nauczysz się animować teksty i korzystać z grafiki wektorowej w formacie SVG oraz zadbasz o wydajność animacji. Ta książka stanowi doskonałą lekturę, po której zaskoczysz użytkowników Twoich stron WWW atrakcyjnymi animacjami.

DZIĘKI TEJ KSIĄŻCE:

- Poznasz bibliotekę Velocity.js
- Zrozumiesz teoretyczne podstawy projektowania ruchu
- Zaznajomisz się z najlepszymi technikami kodowania animacji
- Przygotujesz animację tekstu
- Poznasz format SVG
- Wykonasz atrakcyjny pokaz dla Twoich klientów

JULIAN SHAPIRO — założyciel startupu oraz deweloper. Obecnie skupia uwagę na poprawie prezentacji ruchu w internecie. Autor biblioteki Velocity.js, najczęściej używanej do tworzenia animacji JavaScript na stronach WWW. Velocity.js jest stosowana przez takie serwisy jak WhatsApp oraz Tumblr.

Helion	
37666	numer katalogowy
księgarnia internetowa	
http://helion.pl	
zamówienia telefoniczne	
	0 801 339900
	0 601 339900
Sprawdź najnowsze promocje: • http://helion.pl/promocje Książki najczęściej czytane: • http://helion.pl/bestsellery Zamów informacje o nowościach: • http://helion.pl/nowosci	
Helion SA ul. Kosciuszki 1c, 44-100 Gliwice tel.: 32 230 98 63 e-mail: helion@helion.pl http://helion.pl	
ISBN 978-83-283-1572-3	
9 788328 315723	
cena: 34,90 zł	

