

УДК 004.4:004.02
Ф88



Оригінальна назва твору: HEAD FIRST. JAVASCRIPT PROGRAMMING

Усі права збережено.

Жодна частина цієї книжки не може бути відтворена в будь-якій формі без письмового дозволу власників авторських прав.

Authorized Ukrainian translation of the English edition
of Head First JavaScript Programming ISBN 9781449340131.

Даний переклад публікується та продається з дозволу O'Reilly Media, Inc,
які володіють або контролюють всі права на публікацію та продаж
того ж самого перекладу.

Фрімен Ерік

Ф88 Head First. Програмування на JavaScript / Фрімен Ерік, Робсон Елізабет / Пер. з англ.
Г. Якубовська. — Харків : ВД «Фабула», 2022. — 672 с.

ISBN 978-617-522-047-4

Ця книжка познайомить вас з особливостями JavaScript — основної мови програмування Всесвітньої мережі, що дозволяє визначити розширену поведінку на вебсторінках. Ви отримаєте глибоке розуміння того, як працює ця чудова мова, назавжди забудете про сухі, нудні, статичні сторінки, що просто займають місце на екрані, навчитеся взаємодіяти з користувачами, реагувати на події, отримувати і використовувати дані з Інтернету, виводити графіку та багато іншого. І все це дозволить вам перейти від основ до найцікавіших концепцій сучасного комп'ютерного програмування і розв'язання проблем експертного рівня.

Як і завжди у книжках від Head First, матеріал викладений у цікавій, розважальній та дуже змістовній формі. Це по-справжньому серйозний посібник — та, однак, його легко і приємно читати і вивчати.

УДК 004.4:004.02

Copyright © 2014 Eric Freeman, Elisabeth Robson.
All rights reserved.
© Г. Якубовська, пер. з англ., 2022
© ВД «Фабула», 2022

ISBN 978-617-522-047-4

Відгуки про «Програмування на JavaScript» від Head First

Увага: в жодному разі не читайте «Програмування на JavaScript» від Head First, допоки не вирішите вивчити основи програмування на JavaScript у цікавій, розважальній та змістовній формі. Тут може виникнути побічний ефект: ви дійсно зможете дізнатись більше про JavaScript, ніж після прочитання типових посібників і довідників.

Джессі Палмер, старший розробник програмного забезпечення Gamnett Digital

Серія Head First використовує досягнення сучасної теорії навчання, щоби швидко ввести читачів у курс справи. Авторі довели цією книжкою, що матеріал експертного рівня можна засвоїти швидко та ефективно. Жодних сумнівів — це по-справжньому серйозний посібник, та, проте, його приємно читати і вивчати!

Френк Мур, вебдизайнер і розробник

Шукаєте книжку, що зацікавить вас (і розсмішить), одночасно навчаючи серйозним навичкам програмування? «Програмування на JavaScript» від Head First — саме те, що вам треба!

Тім Вільямс, підприємець у галузі програмного забезпечення

Додайте цю книжку до своєї бібліотеки незалежно від рівня навичок у програмуванні!

Кріс Фузельє, консультант з інженерно-технічних питань

Робсон і Фрімен знову це зробили! Використовуючи той самий веселий та насичений інформацією стиль, як і в попередніх книжках серії Head First, «Програмування на JavaScript» дозволяє читачеві сформувати міцний фундамент сучасного програмування на JavaScript, яким можна в подальшому широко користатися для розв'язання найсерйозніших проблем.

Рассел Аллен-Віллемс, цифровий археолог у DiachronicDesign.com

Найкращий стиль подання матеріалу для внутрішньо розкутого експерта-розробника, який захований у кожному з нас. Відмінний довідник з практичних стратегій розробки — мій мозок працює, не відволікаючись на докучливий і застарілий академічний жаргон.

Тревіс Каланік, генеральний директор Uber

Купити книгу на сайті kniga.biz.ua >>>

Програмування на JavaScript ■

Вражаюча зрозумілість цієї книжки, її гумор та значний рівень дотепності допоможе навіть програмістам-початківцям ефективно мислити під час розв'язання непростих задач.

*Корі Доктороу, друга редакторка журналу Boing Boing,
письменниця-фантаст*

У мене таке відчуття, ніби я щойно осягнув тисячу фунтів інших книжок.

Ворд Каннінгем, творець Wiki

Одна з небагатьох книжок із програмування, що залишає переконливе враження, ніби вона є незамінною. Насправді я би нарахував не більше десятка подібних книжок з цієї галузі.

*Девід Гелентер, професор комп'ютерних технологій
Ельського університету*

Я не можу уявити кращих екскурсоводів, ніж Ерік та Елізабет.

*Міко Мацумура, віцепрезидент із маркетингу та зв'язків із розробниками у Hazelcast,
колишній головний спеціаліст Sun Microsystems із Java*

Візуальний підхід та послідовне викладення матеріалу точно відтворює найкращий спосіб вивчення цього матеріалу.

Денні Гудман, автор книжки Dynamic HTML: The Definitive Guide

Ерік та Елізабет добре знаються на своїй справі. Інтернет-технології стають складнішими, і творче створення вебсторінок набуває все більшого значення. Елегантна архітектура посідає центральне місце в кожному розділі, кожна концепція викладається з рівною дозою прагматизму і дотепності.

*Кен Голдстейн, колишній генеральний директор Shop.com, автор книжки
This is Rage: A Novel of Silicon Valley and Other Madness*

Head First Програмування на JavaScript

От було би чудово,
якби існувала книжка із JavaScript,
яка була би приємнішою за відвідини
стоматолога, і зрозумілішою,
ніж форма податкової декларації!
Мабуть, про це залишається
тільки мріяти...



**Ерік Фрімен
Елізабет Робсон**

O'REILLY®

Пекін • Кембридж • Кельн • Себастьян (Каліфорнія) • Токіо

Купити книгу на сайті kniga.biz.ua >>>

*Присвячується мові JavaScript.
Ти не народилася зі срібною ложкою в роті,
але перевірила всі мови,
що намагалися конкурувати з тобою у браузерах.*

Купити книгу на сайті kniga.biz.ua >>>

Про авторів «Програмування на JavaScript» big Head First



← Ерік Фрімен

Ерік, за словами Кеті Сьєрра, співавторки серії Head First, є «однією з тих рідкісних особистостей, які добре знаються на мові, практиці та культурі найрізноманітніших галузей — від техно-хакінгу до віцепрезидента корпорації, інженера, аналітика».

Майже десять років Ерік працював на посаді технічного директора *Disney Online & Disney.com* у компанії «*The Walt Disney Company*». Зараз він присвячує переважну частину свого часу «*WickedlySmart*» — стартапу, заснованому разом з Елізабет.

За освітою Ерік є фахівцем із комп'ютерних технологій; він займався дослідженнями разом із корифеєм галузі Девідом Гелернтером ще під час роботи над дисертацією в Ельському університеті. Його дисертація вважається однією з фундаментальних праць у галузі альтернатив для інтерфейсів, що реалізують метафору робочого столу, а також першою реалізацією концепції потоків активності, розробленої ним спільно з доктором Гелернтером.

У вільний час Ерік серйозно займається музикою; останній проект Еріка, створений ним у співдружності з одним із піонерів музичного стилю ембієнт Стівом Роучем, доступний в магазині *iPhone App Store* під назвою «*Immersion Station*».

Ерік живе з дружиною і маленькою донькою в Остіні (штат Техас). Його донька часто навідується до студії Еріка; їй дуже подобається крутити верньєри синтезаторів і генераторів аудіоефектів. Пишіть йому за адресою eric@wickedlysmart.com або відвідайте його сайт ericfreeman.com



↙ Елізабет Робсон

Елізабет — розробниця, письменниця і викладачка. Вона закохана у свою роботу ще із часів навчання в Ельському університеті, де отримала ступінь магістра в галузі комп'ютерних технологій, а також створила паралельну візуальну мову програмування і архітектуру програмного забезпечення.

Елізабет брала участь у створенні популярного сайту *The Ada Project* — одного з перших, що допомагають жінкам знайти інформацію про роботу та освіту в галузі комп'ютерних технологій.

Вона стала однією із засновників «*WickedlySmart*» — компанії, що працює у сфері інтернет-освіти на базі вебтехнологій. Тут вона пише книжки і статті, створює відеокурси тощо. Раніше Елізабет обіймала посаду директора спеціальних проєктів в «*O'Reilly Media*» і розробляла семінари та курси дистанційного навчання з різних технічних питань, що допомагають людям розібратися в новітніх технологіях. До приходу в «*O'Reilly*» Елізабет працювала в «*The Walt Disney Company*», де керувала дослідженнями і розробками у сфері цифрових мультимедійних технологій.

Коли Елізабет не сидить за комп'ютером, вона займається велоспортом і веслуванням, фотографує або готує вегетаріанські страви.

Із нею можна зв'язатися за адресою beth@wickedlysmart.com або відвідати її блог <http://elisabethrobson.com>.

Зміст (коротко)

Вступ	25
1. Беремося до справи. Швидке занурення у JavaScript	39
2. Наступний крок. Створення реального коду	81
3. Початок роботи. Знайомство із функціями	115
4. Наведення ладу в даних. Масиви	159
5. Подорож до Об'єктивіля. Розуміння об'єктів	205
6. Знайомство з DOM. Взаємодія з вебсторінкою	259
7. Серйозні типи. Типи, рівність, перетворення — і весь цей джаз	293
8. Побудова застосунку. Збираємо все до купи	343
9. Обробка подій. Асинхронне програмування	407
10. Звільнені функції. Функції першого класу	453
11. Анонімні функції, області видимості та замикання. Серйозні функції	499
12. Створення об'єктів. Розширене конструювання об'єктів	541
13. Використання прототипів. Надпотужні об'єкти	581
Додаток: Усе, що залишилося. Десять тем, яких ми ще не торкалися	641

Зміст (докладно)

Вступ

Ваш мозок і JavaScript. Коли ви намагаєтеся щось вивчити, ваш мозок намагається надати вам послугу, переконуючи, що все це не варто уваги. Він думає: «Краще перейматися важливішими речами, наприклад, небезпечними дикими тваринами або тим, що катання голяка на сноуборді — погана ідея». Як же переконати свій мозок у тому, що ваше життя залежить від знання програмування на JavaScript?

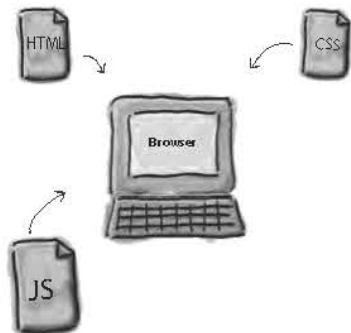
Для кого ця книжка?	26
Ми знаємо, про що ви подумали	27
Ми вважаємо читача «Head First Java» учнем	28
Метапізнання: мислення про мислення	29
Ось що зробили МИ	30
Ось що ви можете зробити, аби змусити свій мозок підкорятися	31
Read Me	32
Технічні консультанти	35
Подяки*	36



1 Беремося до справи Швидко занурення у JavaScript

JavaScript подарує вам суперсилу! JavaScript, основна мова програмування Всесвітньої мережі, дозволяє визначати розширену поведінку на вебсторінках. Забудьте про сухі, нудні, статичні сторінки, що просто займають місце на екрані,— із JavaScript ви будете взаємодіяти з користувачами, реагувати на події, отримувати і використовувати дані з Інтернету, виводити графіку та багато іншого. Коли ж ви оволодієте JavaScript, то зможете навіть створювати абсолютно нові моделі поведінки для ваших користувачів.

- Принцип роботи JavaScript..... 40
- Як пишеться код JavaScript 41
- Як розмістити код JavaScript на сторінці 42
- JavaScript, ти пройшла довгий шлях, дитинко..... 44
- Як описати твердження 48
- Змінні і значення 49
- Тримайтеся подалі від клавіатури! 50
- Акуратніше з виразами! 53
- Багаторазове виконання операцій 55
- Як працює цикл while 56
- Прийняття рішень у JavaScript 60
- А якщо потрібно ухвалити БАГАТО рішень? 61
- Залучайте користувача до взаємодії зі сторінкою 63
- Ближче знайомство з console.log 65
- Як відкрити консоль..... 66
- Пишемо серйозний застосунок на JavaScript 67
- Як додати код на сторінку? (Перераховуємо способи)..... 70
- Нам доведеться вас розділити..... 71

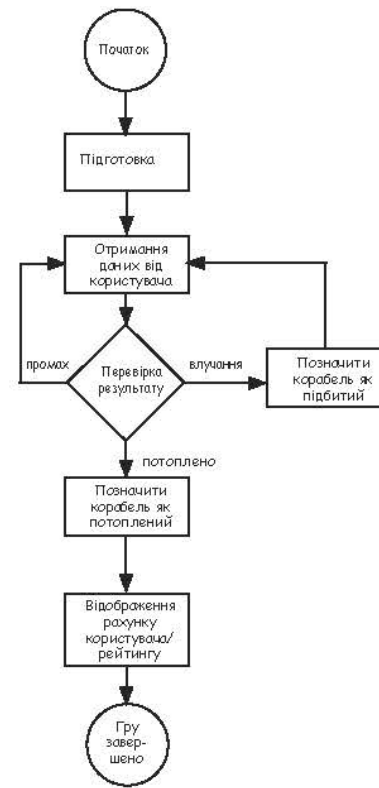


2 Наступний крок Створення реального коду



Ви вже знаєте, що таке змінні, типи, вирази тощо. Отже, ви вже дещо знаєте про JavaScript. Ба більше: цих знань достатньо, щоби почати писати справжні програми, які роблять щось настільки цікаве, що хтось захоче ними користуватися. Втім, вам не вистачає практичного досвіду написання коду, і ми прямо зараз почнемо розв'язувати цю проблему. Яким чином? Просто візьмемося за створення нескладної гри, повністю реалізованої на JavaScript. Мета амбіційна, але ми будемо рухатися до неї поступово, крок за кроком. Тож — до справи, а якщо вам раптом захочеться використати нашу розробку у своїх стартапах — ми не проти, користуйтеся цим кодом, як заманеться.

- Давайте реалізуємо гру «Морський бій» 82
- Наша перша спроба 82
- Починаємо із проектування..... 83
- Ще трохи деталей..... 84
- Робота через псевдокод..... 85
- Стоп! Перш ніж рухатися далі, згадайте про HTML!..... 87
- Пишемо код спрощеної версії «Морського бою»..... 88
- Час переходити до реалізації логіки гри 89
- Крок перший: налаштування циклу,
отримання деяких вхідних даних 90
- Як працює функція prompt..... 91
- Перевірка припущення користувача (змінна guess)..... 92
- Ну що, влучив? 94
- Додавання коду перевірки влучення у ціль..... 95
- Гей, ти потопив мій корабель! 95
- Здійснить аналіз після гри..... 96
- Реалізація логіки готова!..... 98
- Трохи про контроль якості (Quality Assurance)..... 99
- А чи можна стисліше? 103
- Завершення спрощеної версії «Морського бою» 104
- Як отримати випадкову позицію 105
- Всесвітньо відомий рецепт генерування випадкових чисел 105
- Повертаємося до контролю якості 107
- Вітаємо, ви створили свою першу програму на JavaScript!
А тепер декілька слів про повторне використання коду..... 109

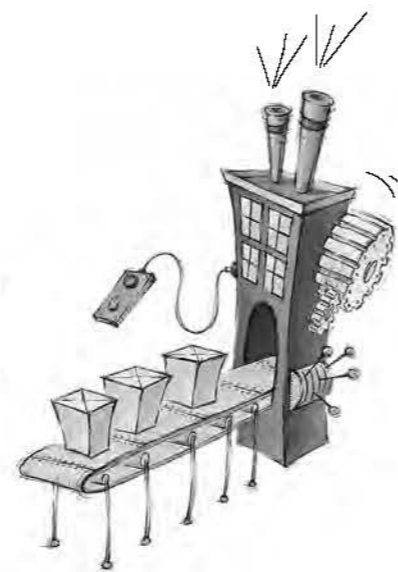


3 Початок роботи Знайомство із функціями

Приготуйтеся опанувати ваші перші суперздібності! Ви вже дещо знаєте про програмування, тож час зробити наступний крок та освоїти роботу із **функціями**. Функції дозволяють писати код, що можна застосувати в різних ситуаціях, код, який можна **повторно використовувати**; код, істотно простіший у **супроводі**; код, який можна **абстрагувати** і привласнити йому просте ім'я, щоби забути про рутинні подробиці та зайнятися по-справжньому важливими справами. Ви переконаєтеся, що функції не лише відкривають шлях до майстерності розробника, але й відіграють ключову роль у стилі програмування JavaScript. У цьому розділі ми почнемо з основ: механіки і всіх тонкощів роботи функцій, а в подальшому будемо вдосконалювати свої навички роботи із функціями. Отже, почнемо закладати надійну основу просто зараз.



То чим поганий цей код?	117
До речі, а ми часом не згадували про ФУНКЦІЇ?.....	119
Гаразд, але як усе це працює?	120
Що можна передати функції?	125
У JavaScript використовується передача параметрів за значенням ...	128
Загадкові функції	130
А ще функції можуть повертати значення.....	131
Простежимо, як виконується функція з командою return.....	132
Глобальні та локальні змінні.....	135
Знайомство з областю дії локальних і глобальних змінних	137
Коротке життя змінних	138
Не забувайте оголошувати локальні змінні!.....	139



4 Наведення лагу в даних Масиви



JavaScript має справу не лише з числами, рядками і логічним типом даних. Дотепер ви писали код на JavaScript виключно із **примітивами** — простими рядками, числами і логічним типом даних на кшталт `Fido`, `23` і `true`. Із примітивними типами можна зробити багато чого, але в якусь мить виникне необхідність у **більшій кількості даних**. Наприклад, для подання всіх позицій у кошику покупок, усіх пісень у плей-листі, групи зірок та їхніх зоряних величин або каталогу продуктів. Подібні завдання вимагають серйозніших засобів. Типовим інструментом для подання таких однорідних даних є масив JavaScript, і у цьому розділі ми дізнаємося, як розмістити дані в **масив**, як передавати їх і працювати з ними. У наступних розділах будуть розглянуті інші способи **структурування даних**, але почнемо ми з масивів.



Чи можете ви допомогти «Bubbles-R-Us»?	160
Як представити декілька значень у JavaScript	161
Як працюють масиви	162
Як звернутися до елемента масиву	163
Оновлення значення в масиві	163
Скільки ж елементів у масиві?	164
Генератор красивих фраз.....	166
Тим часом у фірмі «Bubbles-R-Us».....	169
Як перебрати елементи масиву	172
Але стривайте, існує зручніший спосіб перебору масивів!.....	174
Що, знову?.. Чи не можна стисліше?	180
Доопрацювання циклу <code>for</code> з оператором постфіксного збільшення	181
Створення порожнього масиву (і додавання елементів до нього) ...	185
А переможцями стають... ..	189
Стислий огляд коду	191
Створення функції <code>printAndGetHighScore</code>	192
Рефакторинг коду за допомогою функції <code>printAndGetHighScore</code>	193
Збираємо все до купи... ..	195

5 Подорож до Об'єктвіля Розуміння об'єктів

Донеper ви використовували примітиви і масиви у своєму кодi. І при цьому застосовувалася методологія процедурного програмування із простими операторами, умовами, циклами for/while і функціями — такий підхід був далекий від принципів об'єктно-орієнтованого програмування. Власне, він узагалі не мав нічого спільного з об'єктно-орієнтованим програмуванням. Ми іноді використовували об'єкти, причому ви про це навіть не знали, але ще не написали жодного власного об'єкта. Настав час покинути нудне процедурне містечко і зайнятися створенням власних об'єктів. У цьому розділі ви дізнаєтеся, чому об'єкти відчутно покращують наше життя — принаймні в області програмування (насправді ми не можемо допомогти вам і зрозуміти віяння моди, і одночасно оволодіти навичками JavaScript в одному посібнику). Просто зятямте: звикнувши до об'єктів, ви вже не схочете повертатися назад. І не забудьте надіслати листівку, коли обживетеся.

- Хтось сказав «об'єкти»?! 206
- Детальніше про властивості 207
- Як створити об'єкт 209
- Що таке «об'єктно-орієнтований підхід»? 212
- Як працюють властивості 213
- Як об'єкт зберігається в змінній? Допитливі голови цікавляться... 218
- Порівняння примітивів з об'єктами. 219
- Об'єкти здатні на більше... 220
- Попередня перевірка 221
- І що вийшло?..... 221
- Перевірка крок за кроком..... 222
- Ще трохи поговоримо про передачу об'єктів у функції..... 224
- Удосконалення методу drive 231
- Чому метод drive не знає про властивості started?..... 234
- Як працює this..... 236
- Як поведінка впливає на стан. Додамо трохи газу!..... 242
- Тепер давайте вплинемо станом на поведінку 243
- Вітаємо з першими об'єктами! 245
- Уявіть: вас оточують суцільні об'єкти!
(І вони спрощують вашу роботу) 246

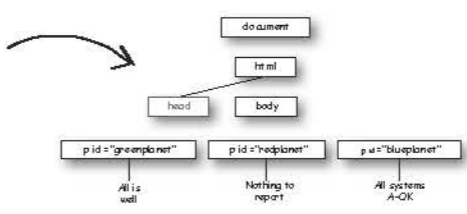


6 Знайомство з DOM Взаємодія з вебсторінкою

Ви значно просунулися у вивченні JavaScript. Фактично, із новачка в області скриптового програмування ви перетворилися на... розробника. Утім, дечого ще бракує: для повноцінного використання ваших навичок поводження із JavaScript слід навчитися взаємодіяти з вебсторінкою, у якій розташовується ваш код. Тільки у цьому випадку ви зможете писати динамічні сторінки, що реагують та відповідають на дії користувача й самі оновлюються після завантаження. Отже, як взаємодіяти зі сторінкою? Із допомогою об'єктної моделі документа DOM (Document Object Model). У цьому розділі ми детально розглянемо DOM і подивимося, як можна її використовувати в поєднанні із JavaScript, а також спробуємо навчити вашу сторінку декільком новим трюкам.

- У попередньому розділі ми запропонували вам виконати невелике завдання. Йдеться про випробування «Злам коду»..... 260
- То що ж робить цей код? 261
- Як JavaScript насправді взаємодіє з вашою сторінкою..... 263
- Як створити власну модель DOM 264
- DOM: перші враження 265
- Отримання елемента методом getElementById 270
- Що саме я отримую від DOM?..... 271
- У пошуках внутрішнього HTML 272
- Що відбувається при внесенні змін до DOM..... 274
- І не здумайте виконувати мій код до того, як сторінка буде завантажена!..... 279
- Ви кажете: «Обробник подій», а я кажу: «Функція зворотного виклику»... 280
- Навіщо зупинятися? Давайте зробимо наступний крок..... 284
- Як задати атрибут методом setAttribute 285
- Веселощі з атрибутами тривають! 286
- Що станеться, якщо мій атрибут не існує в елементі? 286
- Не забувайте, що getElementById також може повертати null!..... 286
- А тим часом у далекій Сонячній системі... 287
- Що ще можна зробити з DOM? 288

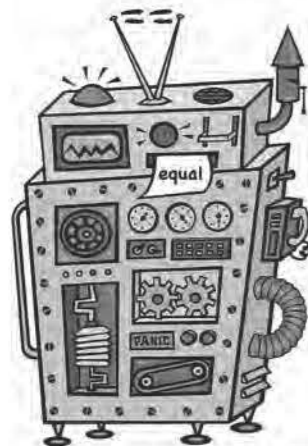
А я, браузер, читаю сторінку і створюю з неї DOM.



7 Серйозні типи

Типи, рівність, перетворення — і весь цей джаз

Настав час серйозно поговорити про типи. Одна із чудових особливостей JavaScript полягає в тому, що початківець може доволі далеко просунути, не заглиблюючись в подробиці мови. Але щоби дійсно **оволодіти мовою**, отримати підвищення по роботі і зайнятися чимось таким, що вам до душі, слід добре розбиратися в **типах**. Пам'ятаєте, що ми говорили про JavaScript? У цієї мови не було такої розкоші, як академічне визначення, що пройшло експертну оцінку. Так, це правда, але відсутність академічного підґрунтя не завадила Стіву Джобсу і Біллу Гейтсу; не завадила вона і JavaScript. Це означає, що система типів JavaScript... м'яко кажучи, не продумана, і в ній трапляється чимало **дивацтв**. Але не турбуйтеся, у цьому розділі ми все розберемо, і незабаром ви навчитеся успішно оминати всі ці неприємні моменти з типами.



- Істина десь поруч... ..294
- Будьте обережні, бо можете несподівано зіткнутися з undefined... ..296
- Як застосовувати null... ..299
- Робота з NaN301
- А далі ще дивніше301
- Ми маємо вам зізнатися303
- Оператор рівності (також відомий як ==)304
- Як відбувається перетворення операндів (усе не так страшно, як може видатися)305
- Як перевірити сувору рівність308
- Ще більше перетворень типів314
- Як перевірити два об'єкти на рівність317
- Правдиве твердження десь поруч... ..319
- Що саме JavaScript вважає «хибним»320
- Таємне життя рядків322
- Рядок може виглядати і як примітив, і як об'єкт323
- П'ятихвилинний огляд методів (і властивостей) рядків325
- Битва за крісло329

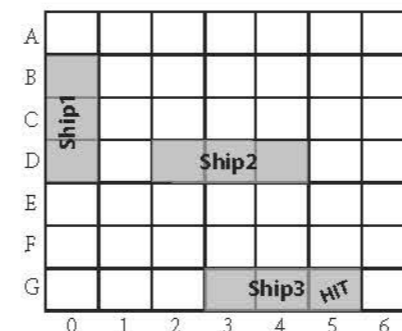
8 Побудова застосунку

Збираємо все до купи

Підготуйте свій інструментарій до роботи. Тобто ваш інструментарій з усіма новими навичками кодування, вашими знаннями DOM і навіть деякими знаннями HTML та CSS. У цьому розділі ми об'єднаємо все це разом, аби створити наш перший повноцінний **вебзастосунок**. Віднині жодних **примітивних ігор** з одним кораблем, що розміщується в одному рядку. У цьому розділі ми побудуємо **повну версію гри**: яскраве велике поле, кілька кораблів та введення даних користувачем безпосередньо на вебсторінці. Ми також створимо структуру сторінки для гри в розмітці HTML, візуально оформимо гру засобами CSS і напишемо код JavaScript, що визначить поведінку гри. Приготуйтеся: у цьому розділі ми займемося повноцінним, серйозним програмуванням і напишемо цілком дієздатний код.



- Цього разу давайте побудуємо СПРАВЖНЮ гру «Морський бій» 344
- Крок назад — до HTML і CSS 345
- Створення сторінки HTML: загальна картина 346
- Додамо ще трохи стильового оформлення 350
- Використання класів hit і miss 353
- Як спроектувати гру 355
- Реалізація представлення 357
- Як працює метод showMessage 357
- Реалізація showMessage 358
- Як працюють методи displayHit і displayMiss 359
- Реалізація displayHit і displayMiss 360
- Як ми будемо представляти кораблі 364
- Реалізація моделі 367
- Роздуми про метод fire 368
- Реалізація контролера 375
- Обробка пострілу гравця 376
- Планування коду 377
- Реалізація методу parseGuess 378
- Підрахунок та обробка пострілів 381
- Як додати обробник подій до кнопки «Fire!» 385
- Передача введених даних контролеру 386
- Як розміщувати кораблі 390
- Метод generateShip 391
- Створення початкової позиції для нового корабля 392
- Завершення методу generateShip 393
- Дві останні зміни 395
- Вітаємо, настав час запуску! 397



9

Обробка подій

Асинхронне програмування

Засвоївши цей розділ, ви зрозумієте, що ви більше не в Канзасі. Дотепер ви писали код, що зазвичай виконується згори вниз. Звісно, ваш код міг бути трошки складнішим, бо в ньому застосовувалися функції, об'єкти і методи, але виконання йшло заздалегідь наміченим курсом. Шкода, що доводиться повідомляти подібну новину лише у другій половині книжки, але **така структура коду не характерна для JavaScript**. Велика частина коду JavaScript пишеться для **обробки подій**. Яких саме? Ну, скажімо, чогось на кшталт клацання користувача на сторінці, надходження даних із мережі, спрацьовування таймера у браузері, змін, що відбуваються в DOM,— і це лише декілька прикладів. Ба більше, у браузері **повсякчас** відбуваються події, що залишаються переважно непоміченими. У цьому розділі ми переглянемо свій підхід до програмування і дізнаємося, для чого потрібно писати код, що реагує на події.

Що таке «подія» (event)?	409
Що таке «обробник події»?	410
Створюємо ваш перший обробник подій.....	411
Розбираємося в подіях, створюючи гру	414
Реалізація гри	415
Додамо ще кілька зображень	420
Тепер варто призначити один обробник усім властивостям onclick усіх зображень	421
Як повторно використовувати один обробник для всіх зображень.....	422
Як працює об'єкт події.....	425
Запуск роботи об'єкта події	427
Черги і події	430
Ще більше подій.....	433
Як працює setTimeout	434
Завершення образу гри	438



10

Звільнені функції

Функції першого класу

Вивчайте функції та сайт, як зірки. У кожній справі є ключовий принцип, саме за ним відрізняються виконавці «середньої ланки» від зірок-віртуозів. І якщо йдеться про JavaScript, ознакою професіоналізму тут є достеменно **розуміння** функцій. Функції відіграють фундаментальну роль у JavaScript, і багато прийомів, що застосовуються при **проектуванні та організації** коду, засновані саме на досконалому знанні функцій і вмінні застосовувати їх. Шлях до вивчення функцій на такому рівні цікавий і непростий, тож приготуйтеся... Цей розділ трохи нагадує екскурсію шоколадною фабрикою Віллі Вонка: під час вивчення функцій JavaScript ви побачите чимало дивного, божевільного і чудового.

Таємниче подвійне життя ключового слова function.....	454
Оголошення функцій і функціональні вирази	455
Аналіз оголошення функції	456
Що далі? Браузер виконує код.....	457
Рухаємося далі. Перевірка умови	458
І наостанок... ..	459
Як функції є також значеннями	463
Чи ми вже згадували, що функції є повноправними громадянами JavaScript?	466
Польоти першим класом	467
Написання коду для обробки і перевірки пасажирів.....	468
Перебір пасажирів.....	470
Передача функції іншій функції	471
Тестовий політ	471
Повернення функцій із функцій	474
Код замовлення стюардесі.....	475
Код замовлення стюардесі: інший підхід	476
Стривайте, одного напою недостатньо!	477
Замовлення напоїв із використанням функції першого класу	478
«Вебвіль-Кола»	481
Як працює метод для сортування масивів sort ...	483
Збираємо все до купи	484
Тим часом у компанії «Webville Cola»	485



11

Анонімні функції, області видимості та замикання Серйозні функції

Ви дізналися чимало нового про функції, але це ще далеко не все. У цьому розділі ми підемо далі, і це буде справжній хардкор! Ми збираємося продемонструвати, як **насправді варто обробляти** функції. Розділ буде не дуже розлогим, але з дуже інтенсивним викладенням матеріалу, і наприкінці його виразність вашого коду JavaScript перевершить усі очікування. Ви також будете готові взятися за аналіз коду колеги або зосередитися на вивченні бібліотеки JavaScript із відкритим кодом, бо ми збираємося охопити деякі поширені ідіоми та угоди, пов'язані з використанням функцій. А якщо ви ніколи не чули про **анонімні функції** і **замикання**, то ви потрапили саме туди, куди треба.

Подивимося на функції з іншого боку	500
Як використовувати анонімну функцію	501
Варто ще раз поговорити про багатослівність	503
Коли визначається функція? Це залежить від... ..	507
Що сталося? Чому функція <code>foo</code> не визначена?	508
Як створюються вкладені функції	509
Як вкладення впливає на область видимості	510
Коротко про лексичну область видимості	512
Чим цікава лексична область видимості	513
Знову до функцій	515

Виклики функцій (знову)	516
Що, до біса, значить «замикання»?	519
Оточення для функції	520
Використання замикань для реалізації чарівного лічильника	522
Погляд за куліси	523
Створення замикання шляхом передачі функціонального виразу як аргумент ...	525
Замикання містить безпосереднє оточення, а не його копію	526
Створення замикання за допомогою обробника події	527
Як працює замикання в нашій програмі	530

От дідько!
Джуді знову
має рацію.

Стривайте хвилинку...
Джуді щось згадувала про
«замикання»? Здається,
це якось пов'язане з тим,
що ми робимо. Давайте
вивчимо цю тему,
щоби остаточно
розібратися з нею.



12

Створення об'єктів Розширене конструювання об'єктів

Досі ми створювали об'єкти вручну. Для кожного об'єкта використовувався **об'єктний літерал**, що задавав усі без винятків властивості. Для невеликих програм це припустимо, але для серйозного коду потрібно щось краще. Саме тут стають у пригоді **конструктори об'єктів**. Конструктори спрощують створення об'єктів, до того ж ми можемо створювати об'єкти за єдиним **шаблоном** — інакше кажучи, конструктори дозволяють створювати серії об'єктів, що мають однакові властивості й містять однакові методи. Код, написаний із використанням конструкторів, виходить значно **компактнішим** і знижує ризик виникнення помилок при створенні великої кількості об'єктів. Запевняємо, що після вивчення цього розділу ви будете користуватися конструкторами так, ніби ви родом з Об'єктвіля.





Створення об'єктів із використанням об'єктних літералів	542
Використання угод для об'єктів	543
Представляємо конструктори об'єктів	545
Як створити конструктор	546
Як користуватися конструктором	547
Як працюють конструктори	548
У конструктори також можна додати методи	550
Настав час виробництва!	556
Не поспішайте прощатися з об'єктними літералами	559
Перетворення аргументів на об'єктний літерал	560
Перетворення конструктора <code>Car</code>	561
Розберемося з екземплярами об'єкта	563
Навіть сконструйовані об'єкти можуть мати незалежні властивості	566
Конструктори в реальному світі	568
Об'єкт <code>Array</code>	569
Ще більше задоволення від вбудованих об'єктів	571



13 Використання прототипів

Надпотужні об'єкти

Опанувати створення об'єктів — лише початок. Настав час «підкачати м'язи» на наших об'єктах. Нам знадобиться більше способів визначення **відносин** між об'єктами та організація **спільного використання коду**. До того ж, нам потрібні механізми розширення і покращення існуючих об'єктів. Інакше кажучи, нам необхідно розширити свій інструментарій роботи з об'єктами. У цьому розділі ви побачите, що у JavaScript реалізована доволі потужна **об'єктна модель**, але вона трохи відрізняється від моделі традиційних об'єктно-орієнтованих мов. Замість типових об'єктно-орієнтованих систем на базі класів, JavaScript надає перевагу моделі **прототипів** — об'єктів, здатних наслідувати і розширювати поведінку інших об'єктів. Яка у цьому користь? Незабаром дізнаєтесь. Отже, до справи...

	Object	Агов, перед тим як ми розпочнемо, пропонуємо кращий спосіб подання об'єктів на діаграмах!	583
	toString() hasOwnProperty() // and more	Знову про конструктори: код використовується повторно, але наскільки ефективно?	584
		Чи дійсно дублювання методів створює проблеми?	586
		Що таке «прототип»	587
		Наслідування від прототипу	588
		Як працює наслідування	589
		Перевизначення прототипу	591
	Dog Prototype	Як налаштувати прототип	594
	species: "Canine"	Прототипи — динамічні	600
	bark() run() wag()	Більш цікава реалізація методу sit	602
		Ще раз: як працює властивість sitting	603
		Як підійти до оформлення виставки собак	607
		Створення ланцюжка прототипів	609
		Як працює наслідування в ланцюжку прототипів	610
	ShowDog Prototype	Створення прототипу show dog	612
	league: "Webville"	Створення екземпляра виставкової собаки	616
	stack() bait() gait() groom()	Остаточне наведення ладу серед виставкових собак	620
		Виклик Dog.call: крок за кроком	622
		Ланцюжок не закінчується на собаці	625
		Застосовуємо наслідування у власних інтересах, перевизначаючи вбудовану поведінку	626
		Застосовуємо наслідування у власних інтересах, розширюючи вбудований об'єкт	628
	ShowDog	Теорія великого об'єднання (ну, не ВСЬОГО, але точно у JavaScript)	630
	name: "Scotty" breed: "Scottish Terrier" weight: 15 handler: "Cookie"	Покращення життя завдяки об'єктам	630
		Складаємо все до купи	631
		Що далі?	631

Додаток: Усе, що залишилося

Десять тем, яких ми ще не торкалися

Ми охопили багато питань, а ви майже завершили знайомство із цією книжкою. Ми, автори, сумувати-мемо за вами, але наостанок усе ж таки маємо провести невелику додаткову підготовку, щоби насолоджуватися довершеним почуттям виконаного обов'язку. Ми не можемо вмістити у цей порівняно невеликий розділ усе, що вам колись знадобиться. Власне, спочатку ми спробували включити сюди те, що ще необхідно знати про програмування на JavaScript (із того, що не було розглянуто в попередніх розділах), утрічі зменшивши розмір шрифту. І справді: вся необхідна інформація помістилася, але жодна людина не змогла її прочитати. Тоді ми викинули більшість інформації і зберегли лише найкращі фрагменти для цього Додатку. Це дійсно кінець книжки. За винятком, звісно, покажчика (що є обов'язковим для ознайомлення!).



1. jQuery	642
2. Робимо більше за допомогою DOM	644
3. Об'єкт Window	645
4. Об'єкт Arguments	646
5. Обробка винятків	647
6. Додання обробників подій за допомогою addEventListener	648
7. Регулярні вирази	650
8. Рекурсія	652
9. JSON	654
10. JavaScript на стороні сервера	655

Наостанок 656

Покажчик 657