

- Przedmowa (11)
- Wstęp (13)
- Podziękowania (15)
- O książce (17)

CZĘŚĆ I. WPROWADZENIE DO APLIKACJI SPA (21)

Rozdział 1. Pierwsza aplikacja jednostronicowa (23)

- 1.1. Definicja, trochę historii oraz nasze cele (24)
 - o 1.1.1. Trochę historii (24)
 - o 1.1.2. Dlaczego musieliśmy tak długo czekać na aplikacje SPA JavaScript? (25)
 - o 1.1.3. Cele (28)
- 1.2. Budowanie pierwszej aplikacji SPA (29)
 - o 1.2.1. Definiowanie celu (30)
 - o 1.2.2. Zainicjowanie struktury plików (30)
 - o 1.2.3. Konfigurowanie narzędzi dla programistów przeglądarki Google Chrome (31)
 - o 1.2.4. Projektowanie kodów HTML i CSS (31)
 - o 1.2.5. Dodawanie kodu JavaScript (33)
 - o 1.2.6. Sprawdzanie działania aplikacji za pomocą narzędzi dla programistów przeglądarki Google Chrome (38)
- 1.3. Korzyści dla użytkownika z dobrze napisanej aplikacji SPA (42)
- 1.4. Podsumowanie (43)

Rozdział 2. JavaScript wraca do łask (45)

- 2.1. Zakres zmiennej (47)
- 2.2. Wynoszenie zmiennej (49)
- 2.3. Zaawansowane wynoszenie zmiennej i obiekt kontekstu wykonywania (51)
 - o 2.3.1. Wynoszenie (51)
 - o 2.3.2. Kontekst wykonywania i obiekt kontekstu wykonywania (52)
- 2.4. Łańcuch zakresów (56)
- 2.5. Obiekty i łańcuch prototypów języka JavaScript (59)
 - o 2.5.1. Łańcuch prototypów (62)
- 2.6. Głębsze spojrzenie na funkcje (67)
 - o 2.6.1. Funkcje oraz funkcje anonimowe (68)
 - o 2.6.2. Samowykonujące się funkcje anonimowe (68)
 - o 2.6.3. Wzorzec modułu - wprowadzanie do języka JavaScript zmiennych prywatnych (71)
 - o 2.6.4. Domknięcia (76)
- 2.7. Podsumowanie (79)

CZĘŚĆ II. KLIENT APLIKACJI SPA (81)

Rozdział 3. Projektowanie powłoki (83)

- 3.1. Gruntowne zrozumienie powłoki (84)
- 3.2. Konfigurowanie plików i przestrzeni nazw (85)
 - o 3.2.1. Tworzenie struktury plików (85)
 - o 3.2.2. Pisanie kodu HTML aplikacji (86)

- o 3.2.3. Tworzenie głównej przestrzeni nazw CSS (87)
 - o 3.2.4. Tworzenie głównej przestrzeni nazw JavaScript (89)
- 3.3. Tworzenie kontenerów funkcji (90)
 - o 3.3.1. Wybór strategii (90)
 - o 3.3.2. Pisanie kodu HTML powłoki (91)
 - o 3.3.3. Pisanie kodu CSS powłoki (92)
- 3.4. Renderowanie kontenerów funkcji (95)
 - o 3.4.1. Konwersja HTML na JavaScript (95)
 - o 3.4.2. Dodawanie szablonu HTML do JavaScript (96)
 - o 3.4.3. Pisanie arkusza stylów powłoki (97)
 - o 3.4.4. Nakazanie aplikacji, aby korzystała z powłoki (99)
- 3.5. Zarządzanie kontenerami funkcji (100)
 - o 3.5.1. Pisanie metody rozwijania i zwijania suwaka czatu (101)
 - o 3.5.2. Dodanie procedury obsługi zdarzeń kliknięcia suwaka czatu (103)
- 3.6. Zarządzanie stanem aplikacji (107)
 - o 3.6.1. Zachowania przeglądarki oczekiwane przez użytkowników (107)
 - o 3.6.2. Wybór strategii zarządzania kontrolkami historii (108)
 - o 3.6.3. Zmiana kotwicy przy wystąpieniu zdarzenia historii (109)
 - o 3.6.4. Użycie kotwicy do sterowania stanem aplikacji (111)
- 3.7. Podsumowanie (116)

Rozdział 4. Dodawanie modułów funkcji (119)

- 4.1. Strategia modułu funkcji (120)
 - o 4.1.1. Porównanie z modułami zewnętrznymi (121)
 - o 4.1.2. Moduły funkcji i fraktalowy wzorec MVC (123)
- 4.2. Konfigurowanie plików modułu funkcji (125)
 - o 4.2.1. Planowanie struktury plików (125)
 - o 4.2.2. Zapełnianie plików (126)
 - o 4.2.3. Co mamy napisane (132)
- 4.3. Projektowanie interfejsów API metod (132)
 - o 4.3.1. Wzorec interfejsu kotwicy (133)
 - o 4.3.2. Interfejsy API konfiguracji Czatu (134)
 - o 4.3.3. Interfejs API inicjowania Czatu (135)
 - o 4.3.4. Interfejs API metody setSliderPosition Czatu (136)
 - o 4.3.5. Kaskada konfiguracji i inicjowania (137)
- 4.4. Implementowanie interfejsu API funkcji (139)
 - o 4.4.1. Arkusze stylów (139)
 - o 4.4.2. Modyfikacja Czatu (144)
 - o 4.4.3. Czyszczenie powłoki (149)
 - o 4.4.4. Prześledzenie wykonywania aplikacji (154)
- 4.5. Dodanie często potrzebnych metod (156)
 - o 4.5.1. Metoda removeSlider (157)
 - o 4.5.2. Metoda handleResize (158)
- 4.6. Podsumowanie (162)

Rozdział 5. Budowanie Modelu (163)

- 5.1. Czym jest Model (164)
 - o 5.1.1. Co zamierzamy zbudować (165)

- o 5.1.2. Zadania Modelu (166)
 - o 5.1.3. Czego Model nie robi (167)
- 5.2. Konfigurowanie Modelu oraz innych plików (168)
 - o 5.2.1. Planowanie struktury plików (168)
 - o 5.2.2. Wypełnianie plików (169)
 - o 5.2.3. Zastosowanie ujednoliconej biblioteki do obsługi wprowadzania danych za pomocą dotyku i myszy (174)
- 5.3. Projektowanie obiektu people (175)
 - o 5.3.1. Projektowanie obiektów person (176)
 - o 5.3.2. Projektowanie interfejsu API obiektu people (178)
 - o 5.3.3. Dokumentowanie interfejsu API obiektu people (181)
- 5.4. Budowanie obiektu people (182)
 - o 5.4.1. Tworzenie listy symulowanych osób (182)
 - o 5.4.2. Rozpoczęcie budowy obiektu people (185)
 - o 5.4.3. Dokończenie obiektu people (189)
 - o 5.4.4. Testowanie interfejsu API obiektu people (195)
- 5.5. Implementacja logowania i wylogowania w powłoce (197)
 - o 5.5.1. Projektowanie wrażeń użytkownika podczas logowania (198)
 - o 5.5.2. Aktualizacja kodu JavaScript powłoki (199)
 - o 5.5.3. Aktualizacja arkusza stylów powłoki (201)
 - o 5.5.4. Testowanie procesów logowania i wylogowania za pomocą interfejsu użytkownika (202)
- 5.6. Podsumowanie (203)

Rozdział 6. Dokończenie budowy modułów modelu i danych (205)

- 6.1. Projektowanie obiektu chat (206)
 - o 6.1.1. Projektowanie metod i zdarzeń (206)
 - o 6.1.2. Dokumentowanie interfejsu API obiektu chat (209)
- 6.2. Budowanie obiektu chat (210)
 - o 6.2.1. Rozpoczęcie budowy obiektu chat przez utworzenie metody join (210)
 - o 6.2.2. Aktualizacja Atrapy, aby reagowała na metodę chat.join (213)
 - o 6.2.3. Testowanie metody chat.join (214)
 - o 6.2.4. Dodanie do obiektu chat obsługi przesyłania wiadomości (216)
 - o 6.2.5. Aktualizacja Atrapy w celu emulowania komunikatów (220)
 - o 6.2.6. Testowanie obsługi komunikatów dla obiektu chat (222)
- 6.3. Dodanie obsługi Awatara do Modelu (224)
 - o 6.3.1. Dodanie obsługi Awatara do obiektu chat (224)
 - o 6.3.2. Modyfikacja Atrapy w celu emulacji awatarów (225)
 - o 6.3.3. Testowanie obsługi awatarów (226)
 - o 6.3.4. Technika projektowania sterowanego testami (227)
- 6.4. Dokończenie modułu funkcji czatu (229)
 - o 6.4.1. Aktualizacja kodu JavaScript Czatu (230)
 - o 6.4.2. Aktualizacja arkuszy stylów (237)
 - o 6.4.3. Testowanie interfejsu użytkownika Czatu (241)
- 6.5. Tworzenie modułu funkcji awatara (242)
 - o 6.5.1. Utworzenie pliku JavaScript Awatara (243)
 - o 6.5.2. Tworzenie arkusza stylów Awatara (248)
 - o 6.5.3. Aktualizacja powłoki i dokumentu przeglądarkowego (249)
 - o 6.5.4. Testowanie modułu funkcji awatara (250)

- 6.6. Wiązanie danych oraz jQuery (250)
- 6.7. Tworzenie modułu danych (252)
- 6.8. Podsumowanie (255)

CZĘŚĆ III. SERWER APLIKACJI SPA (257)

Rozdział 7. Serwer WWW (259)

- 7.1. Rola serwera (259)
 - o 7.1.1. Uwierzytelnianie i autoryzacja (260)
 - o 7.1.2. Walidacja (260)
 - o 7.1.3. Przechowywanie i synchronizacja danych (261)
- 7.2. Node.js (261)
 - o 7.2.1. Dlaczego Node.js? (262)
 - o 7.2.2. Tworzenie aplikacji "Witaj, świecie!" za pomocą serwera Node.js (262)
 - o 7.2.3. Instalowanie frameworku Connect i korzystanie z niego (266)
 - o 7.2.4. Dodawanie funkcji pośredniczących frameworku Connect (267)
 - o 7.2.5. Instalowanie frameworku Express i korzystanie z niego (268)
 - o 7.2.6. Dodawanie funkcji pośredniczących frameworku Express (271)
 - o 7.2.7. Korzystanie z różnych środowisk za pomocą frameworku Express (272)
 - o 7.2.8. Serwowanie plików statycznych za pomocą frameworku Express (273)
- 7.3. Zaawansowany routing (274)
 - o 7.3.1. Trasy operacji CRUD obiektu user (275)
 - o 7.3.2. Ogólny routing operacji CRUD (281)
 - o 7.3.3. Umieszczenie routingu w osobnym module Node.js (284)
- 7.4. Dodanie uwierzytelniania i autoryzacji (288)
 - o 7.4.1. Uwierzytelnianie podstawowe (288)
- 7.5. Technologia WebSocket i biblioteka Socket.IO (289)
 - o 7.5.1. Prosta aplikacja Socket.IO (290)
 - o 7.5.2. Socket.IO oraz serwery wymiany komunikatów (293)
 - o 7.5.3. Aktualizacja kodu JavaScript z wykorzystaniem Socket.IO (294)
- 7.6. Podsumowanie (297)

Rozdział 8. Baza danych serwera (299)

- 8.1. Rola bazy danych (300)
 - o 8.1.1. Wybór magazynu danych (300)
 - o 8.1.2. Eliminacja transformacji danych (300)
 - o 8.1.3. Przeniesienie logiki w wymagane miejsce (301)
- 8.2. Wprowadzenie do MongoDB (302)
 - o 8.2.1. Przechowywanie dokumentowe (303)
 - o 8.2.2. Dynamiczna struktura dokumentów (303)
 - o 8.2.3. Pierwsze kroki z MongoDB (304)
- 8.3. Użycie sterownika MongoDB (305)
 - o 8.3.1. Przygotowanie plików projektu (306)
 - o 8.3.2. Instalacja i połączenie do MongoDB (307)
 - o 8.3.3. Zastosowanie metod CRUD bazy danych MongoDB (308)
 - o 8.3.4. Dodanie operacji CRUD do aplikacji serwera (311)
- 8.4. Walidacja danych klienta (315)
 - o 8.4.1. Walidacja typu obiektu (316)
 - o 8.4.2. Walidacja obiektu (317)

- 8.5. Tworzenie osobnego modułu CRUD (325)
 - o 8.5.1. Przygotowanie struktury plików (326)
 - o 8.5.2. Przeniesienie logiki operacji CRUD do osobnego modułu (329)
- 8.6. Budowanie modułu czatu (334)
 - o 8.6.1. Rozpoczęcie budowy modułu czatu (334)
 - o 8.6.2. Tworzenie procedury obsługi komunikatu adduser (337)
 - o 8.6.3. Tworzenie procedury obsługi komunikatu updatechat (341)
 - o 8.6.4. Tworzenie procedur obsługi komunikatów rozłączania (343)
 - o 8.6.5. Tworzenie procedury obsługi komunikatu updateavatar (345)
- 8.7. Podsumowanie (348)

Rozdział 9. Przygotowanie aplikacji SPA do pracy w środowisku produkcyjnym (349)

- 9.1. Optymalizacja naszej aplikacji SPA dla wyszukiwarek (350)
 - o 9.1.1. Jak Google indeksuje aplikację SPA (350)
- 9.2. Chmura i usługi zewnętrzne (354)
 - o 9.2.1. Analityka stron WWW (354)
 - o 9.2.2. Rejestrowanie błędów po stronie klienta (356)
 - o 9.2.3. Systemy CDN (358)
- 9.3. Buforowanie i pomijanie pamięci podręcznej (359)
 - o 9.3.1. Możliwości buforowania (359)
 - o 9.3.2. Web Storage (360)
 - o 9.3.3. Buforowanie HTTP (362)
 - o 9.3.4. Buforowanie serwera (365)
 - o 9.3.5. Buforowanie zapytań bazy danych (371)
- 9.4. Podsumowanie (372)

DODATKI (375)

Dodatek A Standard kodowania JavaScript (377)

- A.1. Dlaczego potrzebujemy standardu kodowania (378)
- A.2. Układ kodu i komentarze (379)
- A.3. Nazwy zmiennych (389)
- A.4. Deklarowanie i przypisywanie zmiennych (398)
- A.5. Funkcje (400)
- A.6. Przestrzenie nazw (402)
- A.7. Nazwy i układ plików (403)
- A.8. Składnia (404)
- A.9. Walidacja kodu (408)
- A.10. Szablon dla modułów (411)
- A.11. Podsumowanie (413)

Dodatek B Testowanie aplikacji SPA (415)

- B.1. Konfiguracja trybów testowych (416)
- B.2. Wybór frameworku testowego (420)
- B.3. Konfiguracja frameworku nodeunit (421)
- B.4. Tworzenie zestawu testów (422)
- B.5. Dostosowanie modułów SPA do testów (441)
- B.6. Podsumowanie (444)

Skorowidz (445)